PENGEMBANGAN SISTEM PENYEWAAN KENDARAAN KOTA MALANG BERBASIS ANDROID MENGGUNAKAN METODE PENGEMBANGAN MASAM (*MOBILE* APPLICATION SOFTWARE AGILE METHODOLOGY)

SKRIPSI

Untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun oleh:
Meita Putri Aidha
NIM: 135150201111290

PROGRAM STUDI TEKNIK INFORMATIKA
JURUSAN TEKNIK INFORMATIKA
FAKULTAS ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA
MALANG
2018

PENGESAHAN

PENGEMBANGAN SISTEM PENYEWAAN KENDARAAN KOTA MALANG BERBASIS ANDROID MENGGUNAKAN METODE PENGEMBANGAN MASAM (*MOBILE* APPLICATION SOFTWARE AGILE METHODOLOGY)

SKRIPSI

Diajukan untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun Oleh :
Meita Putri Aidha
NIM: 135150201111290

Skripsi ini telah diuji dan dinyatakan lulus pada
18 Mei 2018
Telah diperiksa dan disetujui oleh:

Dosen Pembimbing I	Dosen Pembimbing II
Adam Hendra Brata, S.Kom., M.T., M.Sc.	Komang Candra Brata, S.Kom., M.T., M.Sc.
NIK: 2016079001051001	NIK: 2016078907111001

Mengetahui
Ketua Jurusan Teknik Informatika

Tri Astoto Kurniawan, S.T, M.T, Ph.D
NIP: 197105182003121001

IDENTITAS TIM PENGUJI

No.	Nama Dosen Penguji	NIK
1.	Wibisono Sukmo Wardhono, S.T, M.T (penguji ke I / ketua majelis)	201008 820404 1 001
2.	Ratih Kartika Dewi, S.T., M.Kom (penguji ke II)	201503 890520 2 001

PERNYATAAN ORISINALITAS

Saya menyatakan dengan sebenar-benarnya bahwa sepanjang pengetahuan saya, di dalam naskah skripsi ini tidak terdapat karya ilmiah yang pernah diajukan oleh orang lain untuk memperoleh gelar akademik di suatu perguruan tinggi, dan tidak terdapat karya atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain, kecuali yang secara tertulis disitasi dalam naskah ini dan disebutkan dalam daftar pustaka.

Apabila ternyata didalam naskah skripsi ini dapat dibuktikan terdapat unsur-unsur plagiasi, saya bersedia skripsi ini digugurkan dan gelar akademik yang telah saya peroleh (sarjana) dibatalkan, serta diproses sesuai dengan peraturan perundang-undangan yang berlaku (UU No. 20 Tahun 2003, Pasal 25 ayat 2 dan Pasal 70).

Malang, 4 Juni 2018

Meita Putri Aidha

NIM: 135150201111290

DAFTAR RIWAYAT HIDUP

Data Pribadi

Nama	: Meita Putri Aidha
Jenis Kelamin	: Perempuan
Tempat/Tgl Lahir	: Tangerang, 10 Mei 1995
Agama	: Islam
Status	: Belum Menikah
Alamat	: Jl. Baja Raya No.45 Perumnas II Kota Tangerang
Telp / Hp	: 082234652726

Latar Belakang Pendidikan

A. Pendidikan Formal
   1. Taman Kanak-Kanak Islam Gunung Jati Kota Tangerang
   2. Sekolah Dasar Negeri Karawaci Baru 3 Kota Tangerang
   3. Sekolah Menengah Pertama Negeri 9 Kota Tangerang
   4. Sekolah Menengah Atas Negeri 5 Kota Tangerang

B. Pendidikan Non Formal
   1. 2016 LIA English Course, Intermediate Class
   2. 2013 Bimbingan Belajar Einstein College

Pengalaman Kerja

1. Sekretaris UKM Street Matcha
2. Anggota Kelompok Program Kreativitas Mahasiswa oleh Kemenristek Dikti
3. Web Programmer Sistem Monitoring Rencana Kerja pada PT. PLN TJBTB

Pengalaman Organisasi

1. 2013, Wakil Ketua Umum Ekstrakulikuler Tari Saman - SMAN 5 Kota Tangerang
2. 2014, Anggota Divisi Acara Pengenalan Kehidupan Kampus Mahasiswa Baru. FILKOM - Universitas Brawijaya
3. 2015 – 2016, Bendahara Umum Unit Aktivitas Band Universitas Brawijaya

Skripsi Ini Saya
Persembahkan Untuk Keluarga
Dan Para Sahabat...
Terimakasih Sudah Selalu
Memberikan Dukungan
Semangat Dan Motivasi Untuk
Penulis.

KATA PENGANTAR

Puji syukur kehadirat Tuhan YME yang telah melimpahkan rahmat-Nya sehingga laporan skripsi yang berjudul "Pengembangan Sistem Penyewaan Kendaraan Kota Malang Berbasis Android Menggunakan Metode Pengembangan MASAM (*Mobile* Application Software Agile Methodology)" ini dapat terselesaikan.

Penulis menyadari bahwa skripsi ini tidak akan berhasil tanpa bantuan dari beberapa pihak. Oleh karena itu, penulis ingin menyampaikan rasa hormat dan terima kasih kepada:

1. Papa, Mama, Mas dan Mba atas doa dan segala nasihat, perhatian, dan kesabarannya dalam membesarkan dan mendidik penulis.
2. Bapak Adam Hendra Brata, S.Kom., M.T., M.Sc. dan Bapak Komang Candra Brata, S.Kom., M.T., M.Sc. selaku dosen pembimbing skripsi yang telah dengan sabar membimbing dan mengarahkan penulis sehingga dapat menyelesaikan skripsi ini.
3. Bapak Tri Astoto Kurniawan, S.T, M.T, Ph.D. selaku Ketua Jurusan Teknik Informatika Fakultas Ilmu Komputer Universitas Brawijaya.
4. Arya Yudha Mahendra yang senantiasa membantu, memberikan kritik, saran dan solusi kepada penulis selama penulis menyusun skripsi ini
5. Ayu dan Ike teman merantau di kota Malang dan teman satu atap selama 4 tahun lebih yang senantiasa menemani penulis.
6. Para sahabat di tangerang yang selalu memberikan semangat kepada penulis untuk menyelesaikan skripsi ini.
7. Semua pihak dan teman-teman lainnya ikut membantu dalam proses penyelesaian skripsi dan selalu memberikan semangat kepada penulis.

Penulis menyadari bahwa dalam penyusunan skripsi ini masih banyak kekurangan, sehingga saran dan kritik yang membangun sangat penulis harapkan. Akhir kata penulis berharap skripsi ini dapat membawa manfaat bagi semua pihak yang menggunakannya.

Malang, 4 Juni 2018

Penulis

ABSTRAK

Kebutuhan kendaraan bermotor di Indonesia sekarang ini semakin meningkat, karena sangat membantu aktivitas manusia. Hal ini ditunjukkan 111 juta penduduk Indonesia telah menggunakan kendaraan bermotor. Penetrasi penggunaan jasa penyewaan kendaraan roda empat di Indonesia diprediksikan mengalami peningkatan dengan prediksi mencapai 1,7% pada tahun 2017 dan 2,3% pada tahun 2021. Dari survey yang dilakukan penulis terhadap 31 responden didapatkan nilai 96,8% responden membutuhkan sistem penyewaan kendaraan di Kota Malang untuk memudahkan dalam melakukan pencarian kendaraan tersedia serta melakukan penyewaan. Karena pencarian dan penyewaan kendaraan yang ada sekarang ini masih menggunakan layanan telepon, SMS dan Media Sosial. Maka dari itu didalam penelitian ini penulis membangun sebuah sistem penyewaan kendaraan. *Platform* yang digunakan untuk mengembangkan sistem ini adalah android, karena sebanyak 76,46% masyarakat indonesia menggunakan sistem operasi android. Untuk mengembangkan sistem tersebut penulis memutuskan untuk menggunakan metode pengembangan MASAM yang merupakan salah satu turunan dari metode pengembangan agile. Metode MASAM mendukung pengembangan yang cepat dan menghindari metode pengembangan yang rumit dengan memanfaatkan *domain knowledge* dan *UI Prototyping* untuk mengetahui keinginan pengguna. Hasil pengujian *usability* yang didapatkan dengan menggunakan metode SUS didapatkan skor 79.4 dari aplikasi pelanggan dan 72.5 dari aplikasi rental. Sehingga dapat disimpulkan sistem ini dapat diterima dan mudah digunakan oleh pengguna.

Kata kunci: android, agile, masam

ABSTRACT

Vehicle usage in Indonesia is increasingly, because the function of vehicle is very helpful for indonesian people. This is indicated by 111 million Indonesians already using vehicles. Penetration of vehicle rental services in Indonesia is predicted to increase with prediction 1.7% in 2017 and 2.3% in 2021. From survey collected by writer to 31 respondents, 96,8% of respondents need vehicle rental system in Malang City to facilitate them to search available vehicle and rent vehicle. Because to find available vehicle and rent vehicle, they still use telephone, SMS and social media. In this study the authors build a vehicle rental system. The platform used to develop this system is android, because 76.46% of Indonesian people use android. To develop the system, author decided to use MASAM as a development method which is one of the derivatives of the agile development method. The MASAM method supports rapid development and avoids complex development methods by utilizing the domain knowledge and UI Prototyping to find out the user desires. The results of usability testing obtained by using SUS method is 79.4 from customer and 72.5 from rental. So it can be concluded this system is acceptable and easy to use by the user.

Keyword : android, agile, masam

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

DAFTAR LAMPIRAN

BAB 1 PENDAHULUAN

1.1 Latar belakang

Kebutuhan kendaraan bermotor di Indonesia sekarang ini semakin meningkat, karena sangat membantu aktivitas manusia dalam kehidupannya. Berdasarkan data Badan Pusat Statistik tahun 2015, jumlah penduduk di Indonesia yaitu sebesar 255 juta dengan 185 juta penduduk berumur 15-65 tahun keatas yang diantaranya 13 juta penduduk indonesia menggunakan kendaraan roda empat dan 98 juta menggunakan kendaraan roda dua (Badan Pusat Statistik, 2015). Hal ini menunjukkan sekitar 111 juta penduduk sudah menggunakan kendaraan bermotor dan 74 juta diantara penduduk Indonesia berumur 15-65 tahun keatas masih belum menggunakannya. Maka dengan itu, penduduk di Indonesia memanfaatkan transportasi umum atau penyewaan kendaraan untuk memenuhi kebutuhannya.

Dikutip dari Statista (2017), penetrasi penggunaan jasa penyewaan kendaraan roda empat di Indonesia diprediksikan mencapai 1,7% pada tahun 2017 dan 2,3% pada tahun 2021. Sedangkan prediksi pendapatan dari bisnis penyewaan kendaraan roda empat di Indonesia akan mengalami peningkatan sebanyak 12,5% pada tahun 2017. Hal ini menunjukkan perkembangan pembukaan bisnis penyewaan kendaraan roda empat di Indonesia mengalami perkembangan yang pesat dan salah satunya di Kota Malang. Berdasarkan survei yang penulis lakukan penulis terhadap 31 responden yang pernah melakukan penyewaan kendaraan di Kota Malang, penulis mendapatkan sebanyak 64,5% responden pernah menyewa kendaraan roda empat dan 58,1% pernah menyewa kendaraan roda dua.

Sekarang ini sistem pencarian kendaraan yang tersedia dan untuk melakukan penyewaan kendaraan di kota Malang berdasarkan survey yang penulis lakukan, sebanyak 48,4% responden menggunakan telepon, 64,5% menggunakan Media Sosial dan 22,6% menggunakan SMS. Sehingga masih sedikit yang memanfaatkan suatu aplikasi sebagai sarana pencarian dan penyewaan kendaraan di Kota Malang. Seperti yang terlihat pada rental kendaraan Grand Trans Malang dalam websitenya hanya memberikan informasi mengenai nama rental, nomor telepon rental, jenis kendaraan yang disewakan beserta harga sewa (Grand Trans Malang, 2016). Untuk mendapatkan informasi mengenai ketersediaan kendaraan dan melakukan penyewaan, calon penyewa harus menghubungi rental kendaraan satu persatu melalui telepon, media sosial atau pun SMS. Dari survey yang telah dilakukan terdapat beberapa pertimbangan yang dilakukan oleh responden dalam memilih kendaraan yang akan disewa, yaitu 96.8% responden memilih harga sewa, 52.6% mempertimbangkan lokasi pengambilan kendaraan, 51.6% memilih fasilitas dan kondisi kendaraan serta sebanyak 45.2% memilih tipe kendaraan. Berdasarkan pengalaman tersebut, 16.1% responden merasa mudah, 51.6% responden biasa saja dan 32.3% responden masih merasa kesulitan dalam melakukan pencarian dan penyewaan kendaraan yang tepat dan sesuai. Sehingga diperlukan suatu sistem yang dapat memudahkan dalam pencarian dan

penyewaan kendaraan di Kota Malang yang dibuktikan sebanyak 96,8% responden menyatakan memerlukan sistem tersebut.

Berdasarkan permasalahan yang diuraikan diatas, maka penulis berencana mengembangkan suatu sistem penyewaan kendaraan Kota Malang yang akan dikembangkan pada platform *mobile.* Hal ini dikarenakan penggunaan teknologi informasi aplikasi berbasis *mobile* semakin mengalami pertumbuhan yang pesat sekaligus menyediakan performa yang cepat dan mempunyai mobilitas yang tinggi. Sedangkan untuk pemilihan sistem operasi pada platform *mobile*, peneliti memilih sistem operasi android yang dimana jumlah pengguna sistem operasi android di Indonesia mencapai 76,46% (Statista, 2017).

Untuk mengembangkan suatu aplikasi *mobile* yang baik, pada penelitian ini akan digunakan suatu metode pengembangan perangkat lunak berbasis *mobile* yaitu *Mobile* Application Software Agile Methodology (MASAM). MASAM merupakan turunan dari metode pengembangan Agile, yang dimana Agile memungkinkan adanya perubahan perangkat lunak yang sedang dikembangkan yang dimana perubahan tersebut merupakan *feedback* dari calon pengguna perangkat lunak demi menghasilkan suatu perangkat lunak yang baik dan sesuai dengan keinginan pengguna (Leau, et al., 2012). MASAM mendukung pengembangan yang cepat dan menghindari metode pengembangan yang rumit dengan memanfaatkan *domain knowledge* yang terdiri dari *architecture pattern*, *design pattern* dan *UI pattern*. Selain itu didalam MASAM terdapat penggunaan prototipe antarmuka untuk mengetahui keinginan pengguna dalam segi antarmuka aplikasi (Jeong, et al., 2008). Keterkaitan penggunaan MASAM dengan Pengembangan Sistem Penyewaan Kendaraan Kota Malang Berbasis Android adalah dibutuhkannya *feedback* atau tanggapan pengguna pada proses pengembangan sistem ini yang didapatkan dari proses iterasi. Sehingga sistem yang dikembangkan dapat memuat keinginan pengguna serta dapat menyediakan pencarian dan alur penyewaan kendaraan sesuai dengan keinginan pengguna.

Pada penelitian ini penulis akan mengembangkan Sistem Penyewaan Kendaraan Kota Malang berbasis Android menggunakan Metode Pengambangan MASAM *(Mobile Agile Software Application Methodology)*. Dengan harapan sistem ini dapat membantu masyarakat Kota Malang dalam melakukan penyewaan kendaraan di Kota Malang.

1.2 Rumusan masalah

1. Bagaimana hasil analisis kebutuhan pada sistem penyewaan kendaraan Kota Malang berbasis Android dengan menggunakan Metode Pengembangan MASAM?
2. Bagaimana hasil perancangan pada sistem penyewaan kendaraan Kota Malang berbasis Android dengan menggunakan Metode Pengembangan MASAM?
3. Bagaimana hasil implementasi pada sistem penyewaan kendaraan Kota Malang berbasis Android dengan menggunakan Metode Pengembangan MASAM?

4. Bagaimana hasil pengujian pada sistem penyewaan kendaraan Kota Malang berbasis Android dengan menggunakan Metode Pengembangan MASAM?
5. Apakah sistem penyewaan kendaraan Kota Malang dapat membantu memudahkan pengguna dalam pencarian dan penyewaan rental kendaraan?

1.3 Tujuan

Tujuan penelitian ini adalah :

1. Mengembangkan sistem penyewaan kendaraan Kota Malang berbasis Android dengan menggunakan metode pengembangan MASAM.
2. Mengembangkan sistem penyewaan kendaraan kota Malang yang dapat membantu pengguna dalam pencarian dan penyewaan kendaraan.

1.4 Manfaat

Manfaat penelitian dari penelitian ini adalah :

1. Peneliti dapat memahami penerapan metode pengembangan MASAM sebagai salah satu metode pengembangan perangkat lunak berbasis *mobile*.
2. Sistem yang dikembangkan dapat memberikan suatu informasi dan memberikan kemudahan kepada masyarakat dalam pencarian dan penyewaan kendaraan di Kota Malang.

1.5 Batasan masalah

Berdasarkan permasalahan diatas, maka batasan masalah dalam penelitian ini adalah :

1. Sistem ini hanya sebatas prototipe penelitian.
2. Penelitian ini tidak menerapkan beberapa tahapan dalam metode MASAM yang berhubungan dengan *pair programming*, manajemen tim dan rilis publik.
3. Pengembangan sistem pada sisi pengguna yaitu berbasis android dengan menggunakan bahasa pemrograman java.
4. Sistem ini menggunakan *firebase real-time database* sebagai media manajemen basis data.
5. Iterasi yang dilakukan dalam penelitian ini hanya dilakukan sebanyak 2 kali.
6. Penelitian ini tidak berfokus pada pembayaran.

1.6 Sistematika pembahasan

Dalam penyusunan skripsi ini terdapat penulisan yang terstruktur sebagai berikut :

BAB 1 PENDAHULUAN

Bab ini menguraikan masalah umum terkait dengan penelitian yang bersistematis. Penulisan dimulai dari latar belakang, rumusan masalah, tujuan, manfaat, batasan masalah dan sistematika penulisan.

BAB 2 LANDASAN KEPUSTAKAAN

Bab ini membahas tentang kajian pustaka dan dasar teori berdasarkan referensi-referensi yang saling berkaitan dan mendukung penelitian pengembangan sistem penyewaan kendaraan kota Malang menggunakan metode pengembangan MASAM.

BAB 3 METODOLOGI PENELITIAN

Bab ini menguraikan tentang metode dan langkah kerja yang digunakan dalam penulisan penelitian ini yang terdiri dari studi literatur, Analisa Kebutuhan, Perancangan Sistem, Implementasi Sistem, Pengujian dan Analisis Sistem dan Penarikan Kesimpulan.

BAB 4 REKAYASA KEBUTUHAN

Bab ini membahas tentang rekayasa kebutuhan dari sistem penyewaan kendaraan kota Malang dengan menggunakan metode pengembangan MASAM dengan dasar teori dan literatur yang ada.

BAB 5 PERANCANGAN DAN IMPLEMENTASI

Bab ini membahas tentang perancangan dan implementasi sistem penyewaan kendaraan kota Malang dengan menggunakan metode pengembangan MASAM. Seperti bagaimana pengimplementasian antarmuka dan alur program sistem.

BAB 6 PENGUJIAN

Bab ini membahas tentang pengujian sistem penyewaan kendaraan kota Malang dengan menggunakan metode pengembangan MASAM dengan menggunakan beberapa metode pengujian.

BAB 7 PENUTUP

Bab ini membahas kesimpulan dari apa yang telah didapatkan dari penelitian ini dan saran-saran yang diharapkan bermanfaat untuk kemajuan dan perbaikan sistem.

BAB 2 LANDASAN KEPUSTAKAAN

Pada bab landasan kepustakaan berisi kajian pustaka terhadap penelitian-penelitian terkait yang berhubungan dengan penelitian dan pembahasan tentang dasar teori yang dapat mendukung dalam penyusunan penelitian ini. Kajian pustaka membahas tentang penelitian-penelitian yang telah dilakukan sebelumnya yang akan digunakan dalam proses penyusunan skripsi.

2.1 Kajian Pustaka

Pada bagian kajian pustaka akan menampilkan penelitian terdahulu yang berhubungan dengan pengembangan sistem penyewaan kendaraan kota malang berbasis android dengan menggunakan metode pengembangan MASAM. Terdapat satu penelitian tentang metodologi MASAM, namun didalam bagian ini juga akan dijelaskan kajian pustaka lainnya yang berhubungan dengan metode pengembangan sistem berbasis *mobile*. Kajian pustaka pada penelitian ini dapat dilihat pada Tabel 2.1.

Tabel 2.1 Kajian Pustaka Penelitian Terkait

No.	Judul Penelitian	Topik Penelitian	Objek Penelitian	Keunikan Penelitian
1.	*Development Process of Mobile Application SW Based on Agile Methodology* (Jeong, et al., 2008)	Rekayasa Perangkat Lunak	Metode Pengembangan untuk sistem berbasis *mobile*	Memaparkan sebuah metode pengembangan baru dalam pengembangan aplikasi perangkat lunak berbasis *mobile* MASAM
2.	*Agile Development Methodology and Testing for Mobile Applications - A Survey* (Hamsini, et al., 2016)	Rekayasa Perangkat Lunak	Metodologi pengembangan agile pada aplikasi *mobile*.	Memaparkan karakterisitik dari pengembangan aplikasi *mobile*.

3.	*Software Architecture Design Pattern in Java* (Kuchana, 2004).	Rekayasa Perangkat Lunak	Pola-pola perancangan dalam pengembangan sistem menggunakan bahasa pemrograman Java.	Memaparkan pola-pola perancangan atau *design pattern* yang dapat digunakan dalam bahasa pemrogramana java.

Kajian penelitian yang pertama, dilakukan oleh Jeong pada tahun 2008 dengan judul penelitian *"Development Process of Mobile Application SW Based on Agile Methodology"* yang diberi nama MASAM. Metode pengembangan MASAM membahas mengenai suatu metode pengembangan baru yang dapat diimplementasikan dalam pengembangan suatu aplikasi perangkat bergerak dengan menggunakan pendekatan pengembangan Agile. Metode ini diyakini dapat menangani perubahan-perubahan mengenai kebutuhan dari suatu perangkat lunak yang cepat berubah maupun bertambah. Karakteristik dari metode pengembangan ini adalah pengembangan yang cepat dan berbasis pengetahuan atau *domain knowledge* (Jeong, et al., 2008). *Domain knowledge* pada pengembangan MASAM, digunakan untuk mempermudah penggunaan kembali pola-pola dari pengembangan sistem yang akan di kembangkan. Selain itu, metode pengembangan masam dibuat untuk mengembangkan pengembangan yang cepat karena metode MASAM dibuat untuk menghindari metode pengembangan yang rumit. Menurut Jeong, aplikasi *mobile* berhubungan erat dengan lingkungan komunikasi dan memiliki fitur-fitur yang didalamnya dibutuhkan reaksi langsung dari para calon pengguna sehinggan keterlibatan pengguna sangatlah penting. Pada metode MASAM juga membutuhkan *UI (User Interface) Prototyping* untuk mendapatkan keinginan pengguna secara cepat.

Kajian penelitian yang kedua menjelaskan tentang karakteristik dari pengembangan aplikasi *mobile*. Terdapat beberapa karakteristik dari pengembangan aplikasi *mobile* yaitu siklus hidup yang pendek, siklus pengembangan yang pendek, dapat menerima perubahan dari calon pengguna, dapat secara mudah diperbarui dan dapat di unggah secara cepat (Hamsini, et al., 2016). Menurut Hamsini,et al., berdasarkan karakteristik tersebut muncul beberapa metode pengembangan aplikasi *mobile* yang juga mempunyai karakteristik yang berbeda-beda. Diantaranya yaitu *Mobile* D, HME (Hybrid Method Engineering), MASAM (*Mobile* Application Software Agile Methodology), SleSS dan Scrum.

Kajian penelitian yang ketiga dilakukan oleh Kunchana membahas tentang pola – pola perancangan yang terdapat pada pengambangan sistem menggunakan bahasa pemrograman java. Menurut Kunchana, pola desain atau pola perancangan merupakan suatu praktik terbaik dan terdokumentasi untuk suatu

solusi yang telah diterapkan di banyak lingkungan untuk memecahkan masalah yang muncul kembali dalam rangkaian situasi tertentu. Pola desain membahas tentang suatu cara yang ditemukan selama suatu studi atau selama melakukan pengembangan suatu sistem perangkat lunak dan dapat digunakan kembali untuk memecahkan suatu permasalahan yang mungkin muncul pada pengembangan sistem selanjutnya (Kuchana, 2004).

Berdasarkan kajian pustaka yang dijelaskan diatas, pada penelitian Pengembangan Sistem Penyewaan Kendaraan Kota Malang Berbasis Android akan digunakan metode pengembangan MASAM, karena pada penelitian ini dibutuhkan suatu karakteristik pengembangan aplikasi yang mempunyai siklus pengembangan yang pendek dan pada penelitian ini hanya terdiri dari satu sumber daya manusia.

2.2 *Mobile Application Software Agile Methodologies* (MASAM)

Didalam penelitiannya, Jeong et al. (2008) mengusulkan suatu metodologi pengembangan perangkat lunak berbasis *mobile* yaitu Mobile Application Software Agile Methodology (MASAM) yang merupakan suatu metode yang menyediakan pengembangan secara cepat dengan menerapkan metode pendekatan Agile. Metode ini diyakini dapat menangani perubahan-perubahan mengenai kebutuhan dari suatu perangkat lunak yang cepat berubah maupun bertambah. Karakteristik dari metode pengembangan ini adalah pengembangan yang cepat dan berbasis pengetahuan atau *domain knowledge* (Jeong, et al., 2008). *Domain knowledge* pada pengembangan MASAM, digunakan untuk mempermudah penggunaan kembali pola-pola dari pengembangan sistem yang akan di kembangkan. Selain itu, metode pengembangan masam dibuat untuk mengembangkan pengembangan yang cepat karena metode MASAM dibuat untuk menghindari metode pengembangan yang rumit. Menurut Jeong, aplikasi *mobile* berhubungan erat dengan lingkungan komunikasi dan memiliki fitur-fitur yang didalamnya dibutuhkan reaksi langsung dari para calon pengguna sehingga keterlibatan pengguna sangatlah penting. Pada metode MASAM juga membutuhkan *UI (User Interface) Prototyping* untuk mendapatkan keinginan pengguna secara cepat. Didalam metodologi MASAM, pengujian kinerja dilakukan dengan menggunakan TDD (*Test Driven Development*). Berikut ini merupakan karakteristik dari MASAM :

1. Salah satu metode pengembangan aplikasi *mobile*
2. Perkembangan yang cepat dengan menggunakan pendekatan agile
3. Pemanfaatan *domain knowledge.*

MASAM menyediakan proses pengembangan aplikasi perangkat lunak pada platform *mobile* dengan berdasarkan pada pada Extreme Programming (XP), Agile Unified Process (AUP), Rational Unified Process (RUP) dan Software Systems Process Engineering Meta-model (SPEM). Pada Metode pengembangan MASAM, Extreme Programming digunakan pada tahap pengembangan sistem yang membahas tentang siklus pengembangan sistem serta iterasi yang dikerjakan.

Sedangkan Software Systems Process Engineering Meta-model (SPEM) menjelaskan tiga macam aset proses yang ada dalam pengembangan MASAM, yaitu :

Kind	Description	Name
Role	It defines a set of related skills, competencies and responsibilities of an individual(s).	Planner, Manager, UI designer, Developer, Development team, Initial development team, Tester, User
Task	It is an assignable unit of work assigned to a specific role. The granularity of a task is generally a few hours to a few days and usually affects one or only a small number of work products.	Product Summary, Initial Planning, User Definition, Initial Analysis, Select Resource, Select Process, Establish Environment, Write Story Card, UI Design, Define Architecture, Planning, Iteration plan, Face-to-face Meeting, Incremental Design, TDD, Refactoring, Release Plan, Feedback, Pattern Manage, Pair Programming, Integration, Acceptance Test, User Test Figure
Work Product	It is a general term for task inputs and outputs. There are three types of work product.	Product Summary, Project Planner, UI Sample, UI Model, UI pattern, Architecture Pattern, Application Pattern, Story Card, Task Card, Architecture Model, Component Model, Test Case.

Tabel 2.2 Aset Proses Metode MASAM

Sumber : (Jeong, et al., 2008)

Didalam MASAM, tujuan dari penggunaan SPEM adalah untuk menentukkan proses sesuai dengan konteks perusahaan. Sedangkan Rational Unified Language (RUP) yang diterapkan dalam MASAM terdapat dalam tahapan untuk menentukkan *domain knowledge,* yang terdiri dari *architecture pattern, design pattern* dan *UI pattern*. Dalam MASAM tahapan tersebut termasuk dalam *task pattern management* dalam aktivitas architecting yang berada di fase *embodiment.*

MASAM mengusulkan empat tahap pengembangan dalam pengembangan suatu aplikasi *mobile* yaitu : *Development Preparation Phase, Embodiment Phase, Product developing Phase,* dan *Commercialization Phase* yang dimana masing-masing dari tahapan ini terbagi lagi menjadi beberapa tahapan. Berikut ini merupakan gambaran dari tahapan methodology MASAM yang terlihat pada Tabel 2.3.

Phase	Activity	Task
Preparation Phase	Grasping Product	Product summary
		Pre-planning
	Product Concept Sharing	User definition
		Initial product analysis
	Project Set-up	Development process coordination
		Project resouce coordination
		Pre study
Embodiment Phase	User Need Understanding	Story card workshop
		UI design
	Architecting	Non-functional requirement analysis
		Architecture definition
		Pattern management
Development Phase	Implementation & Preparation	Environment setup
		Development planning
	Release Cycle	Release Planning
		Iteration Cycle
		Release
Commercialization Phase	System Test	Acceptance Test
		User Test
	Product Selling	Launching Test
		Product Launching

Tabel 2.3 Metode Pengembangan MASAM

Sumber : (Jeong, et al., 2008)

Namun pada penelitian ini tidak menerapkan semua tahapan yang ada dalam metode MASAM. Penelitian ini hanya mengadopsi beberapa *activity* dan *task* sesuai dengan kebutuhan penelitian. Tabel 2.4 menunjukkan *activity* dan *task* dari metode MASAM yang di adopsi pada penelitian ini.

Phase	Activity	Task
Preparation Phase	Grasping Product	Product summary
		Pre-planning
	Product Concept Sharing	User definition
Embodiment Phase	User Need Understanding	Story card workshop
		UI design
	Architecting	Non-functional requirement analysis
		Architecture definition
		Pattern management
Development Phase	Implementation & Preparation	Environment setup
	Release Cycle	Release Planning
		Iteration Cycle
		Release
Commercialization Phase	System Test	Acceptance Test

Tabel 2.4 Hasil Adopsi Metode Pengembangan MASAM

2.2.1 *Preparation phase*

Didalam metode pengembangan MASAM, dialog dengan calon pengguna sangatlah dibutuhkan selama tahap ini. Para pengembang harus memahami bagaimana tanggapan calon pengguna terhadap produk yang akan dikembangkan. Bagian penting dari tahapan ini adalah visi dan konsep dari produk harus disampaikan kepada calon pengguna dengan baik. Pada fase *preparation,* terdapat beberapa bagian *activity* yang dimana setiap *activity* di bagi menjadi beberapa *task.* Tabel 2.5 menunjukkan aktivitas fase *preparation*.

Preparation Phase	Grasping Product	Product summary
		Pre-planning
	Product Concept Sharing	User definition

Tabel 2.5 Aktivitas Fase *Preparation*

2.2.1.1 Grasping Product

Didalam aktivitas *Grasping Product* terdapat beberapa *task* yang dilakukan, yaitu *Product Summary dan Pre-planning. Product Summary* bertujuan untuk menggambarkan produk atau gambaran umum dari sistem yang akan dikembangkan secara ringkas. Sedangkan *preplanning* merupakan perencanaan awal terhadap produk yang akan dikembangkan. Didalam *Pre-planning,* sebelum pengembang melakukan sebuah proyek beberapa pertimbangan dalam proses perencanaan pengembangan produk, menentukan *stakeholder,* dan manajemen resiko yang mungkin muncul (Greer & Conradi, 2009) .

2.2.1.2 Product Concept Sharing

Tahapan aktivitas ini terdiri dua *task* yaitu *User Definition* dan *Intial Product Analysis. Task User Definition* menjelaskan tentang siapa saja yang akan menggunakan sistem yang akan dibangun. Pada *task* ini, calon pengguna akan didefisinikan.

2.2.2 *Embodiment phase*

Pada fase *embodiment*, penggalian akan kebutuhan mulai digali dari calon pengguna. Pada fase ini keterlibatan calon pengguna sangatlah diperlukan. Untuk penggalian kebutuhan menggunakan *story card.* Selain itu, pada tahap ini pelanggan diberikan suatu prototipe UI atau simulasi *User Interface* dan memeriksa apakah prototipe dirancang sesuai dengan yang calon pengguna inginkan.

MASAM menggunakan pengetahuan domain yang dimana pengetahuan domain terdiri dari pola arsitektur (*architecture pattern)*, pola desain (*design pattern)*, dan pola *User Interface (UI pattern).*

Embodiment Phase	User Need Understanding	Story card workshop
		UI design
	Architecting	Non-functional requirement analysis
		Architecture definition
		Pattern management

Tabel 2.6 Aktivitas Fase *Embodiment*

2.2.2.1 User Need Understanding

Tahap ini merupakan tahapan dari elisitasi kebutuhan dari calon pengguna. Elisitasi kebutuhan dilakukan dengan menggunakan *story card* dan UI desain. *Story card* merupakan sebuah dokumen yang dimana merekam kebutuhan-kebutuhan *stakeholder* dalam bentuk cerita (Onions & Patel, 2009). Menurut Cohn (2004) didalam Onions & Patel (2009), *story card* harus dapat berdiri sendiri, ternilai, dapat diuji, dapat di negosiasi dan mengandung iterasi yang kecil.

Didalam metodologi MASAM, calon pengguna diberikan suatu simulasi *User Interface* dan memeriksa apakah prototipe dirancang sesuai dengan yang calon pengguna inginkan. Prototipe antarmuka akan terus dilakukan sampai memenuhi keinginan calon pengguna. Maka dari itu, pada tahapan ini terdapat beberapa iterasi untuk dapat memenuhi penambahan atau perubahan kebutuhan *mayor*. Apabila terdapat perubahan ketika memberikan perancangan antarmuka kepada pengguna, juga akan memperngaruhi spesifikasi kebutuhan fungsional.

2.2.2.2 Architecting

Dalam tahapan *architecting*, terbagi menjadi tiga *task* yaitu *non-functional requirement analysis, architecture definition,* dan *pattern management*. *Non-functional requirement* merupakan sebuah tahapan yang dilakukan para pengembang untuk mengetahui apa saja kebutuhan non-fungsional yang akan terdapat didalam sistem yang akan dibangun. Menurut Sommervile (2011) dalam Alashqar, et al. (2015), kebutuhan non-fungsional merupakan suatu layanan atau fungsi yang ditawarkan dalam sistem seperti kegunaan, fleksibilitas, efisiensi, dan berlaku untuk sistem secara keseluruhan.

Architecture definition merupakan suatu *task* yang menggambarkan tentang arsitektur dari sistem yang akan dibangun. Salah satu karakteristik dari metodologi MASAM adalah memanfaatkan *domain knowledge. Domain knowledge* terdiri dari beberapa *pattern type* yaitu pola arsitektur (*architecture pattern)*, pola desain (*design pattern)*, dan pola *User Interface (UI pattern).* Pola arsitektur membahas tentang bagaimana strategi menyangkut komponen skala besar, sifat global dan mekanisme sistem. Sedangkan menurut (Edwin, 2014), *design pattern* atau pola desain merupakan solusi umum yang dapat digunakan kembali dalam permasalahan sering terjadi dalam desain *software*. Pola desain berorientasi objek biasanya menunjukkan hubungan dan interaksi antara kelas atau objek. Sedangkan *UI pattern* adalah solusi yang bertujuan untuk memecahkan masalah-

masalah yang ada dalam perancangan desain antarmuka suatu perangkat lunak. Ketiga pola pattern ini termasuk kedalam tahapan *pattern management.*

2.2.3 *Development phase*

Didalam fase *development,* sistem akan secara cepat di bangun lalu di rilis kepada calon pengguna untuk memastikan apakah sistem yang dibangun sudah sesuai dengan keinganan pengguna. Namun terkadang, ketika suatu sistem telah dibangun dan sudah di rilis kepada calon pengguna, tidak menutup kemungkinan adanya penambahan suatu kebutuhan. Maka dari itu didalam fase ini, terbagi menjadi dua buah activity yaitu *Implementation & Preparation dan Release Cycle*.

Development Phase	Implementation & Preparation	Environment setup
	Release Cycle	Release Planning
		Iteration Cycle
		Release

Tabel 2.7 Aktivitas Fase *Development*

Pada metode pengembangan MASAM, pada akitivitas *release cycle* terdapat beberapa *task* didalamnya. Tabel 2.8 akan menunjukkan *task* dari *activity release cycle* yang terdapat metode pengembangan MASAM.

Activity	Task -1	Task - 2	Task - 3
Implementation Preparation	Environment Set up		
	Development Planning		
Release Cycle	Release Planning		
	Iteration Cycle	Iteration Planning	
		Implementation Cycle	Face-to Metting
			Incremental Design
			TDD
			Refactoring
			Pair Programming
			Continuous Integration
		Feed back	
	Release	Acceptance Test	
		Feed back	

Tabel 2.8 Aktivitas Fase *Development*

Namun pada penelitian ini tidak semua *task* yang ada di *activity release cycle* digunakan pada penelitian ini. Sehingga hanya beberapa *task* saja yang di adopsi.

Tabel 2.9 menunjukkan *task* apa saja yang diimplementasikan pada *activity release cycle*.

Activity	Task-1	Task-2	Task-3
Release Cycle	Release Planning		
	Iteration Cycle	Iteration Planning	
		Implementation on Cycle	TDD (Test Driven Development)
	Release	Acceptance Test	
		Feed back	

Tabel 2.9 Hasil Adopsi *Activity Release Cycle* Metode Pengembangan MASAM

2.2.3.1 Implementation & Preparation

Pada tahapan *implementation & preparation* membahas tentang bagaimana keadaan lingkungan dan alat yang dibutuhkan oleh sisi pengembang dalam proses pengembangan sistem dan juga membahas tentang perencanaan dari pengembangan.

2.2.3.2 Release Cycle

Pada tahapan *release cycle* akan dibagi menjadi tiga buah *task, yaitu release planning, iteration cycle* dan *release. Release planning* merupakan perencaan dari sisi pengembang tentang waktu rilis dari sistem yang telah dikembangkan kepada calon pengguna dan juga membahas tentang berapa kali iterasi yang dibutuhkan untuk melaksanakan pengembangan sistem. Setelah *release planning*, selanjutnya terdapat *task Iteration cycle* yang dimana tahapan ini merupakan tahapan untuk melakukan implementasi sistem sesuai dengan kebutuhan fungsional yang didapatkan dari hasil elisitasi kebutuhan. *Iteration cycle* merupakan tahapan yang juga berfungsi untuk menentukan kebutuhan fungsional apa saja yang didapatkan dari *story card* untuk diimplementasikan pada setiap iterasi. Didalam *iteration cycle* terdapat 2 buah bagian *task*, yaitu *task* 2 yang terdiri dari *iteration planning*, *implementation cycle*. Selanjutnya task-3 terdapat dalam implementation cycle yang terdiri dari TDD (test driven development).

Test Driven Development (TDD) merupakan prosedur penulisan tes sederhana sebelum dilakukannya implementasi. TDD merupakan suatu pengujian yang dilakukan dalam lingkungan pengembangan sistem yang berjangkan pendek. Sebelum pihak pengembang melakukan implementasi kode program, akan dilakukan pembuatan *test case* atau kasus uji untuk memastikan kode program yang akan diimplementasikan akan menghasilkan hasil yang lebih baik karena dirancangan sesuai dengan kode tes yang di rancang sebelumnya. Menurut Ambler (2013), terdapat dua buah level pada TDD, yaitu *Acceptance Test Driven Development* (ATDD) dan *Developer Test Driven Development* (TDD).

Acceptance Test Driven Development (ATDD) merupakan suatu tes pengujian yang dilakukan dengan anggota tim dengan perspektif yang berbeda seperti perspektif sisi pengembang, customer dan pengujian yang saling berkolaborasi

untuk membuat *acceptance test* sebelum melakukan implementasi terkait fungsi yang akan di kembangkan didalam sistem. ATDD juga disebut sebagai *Behavior Driven Development* (BDD). Untuk melakukan ATDD dapat digunakan salah satu teknik pengujian *Black-box testing* yang selanjutnya akan dijelaskan pada subbab pengujian perangkat lunak. Sedangkan level lainnya yaitu *Developer Test Driven Development* (TDD) dapat menggunakan teknik pengujian *White-box testing*.

Setelah iterasi dilakukan, selanjutnya sistem akan dirilis kepada calon pengguna. Tahapan tersebut adalah *task release.* Dalam *task release*, terdapat pengujian *Acceptance* yang merupakan pengujian validasi yang dilakukan oleh sisi pengembang, untuk melakukan validasi kebutuhan atau fitur-fitur yang sudah diimplementasikan. Setelah itu sistem yang sudah di validasi akan di berikan kepada pengguna untuk mengetahui tanggapan dari pengguna. Hal ini sangat memungkinkan adanya penambahan maupun pengurangan kebutuhan. Jika terdapat perubahan atau penambahan fitur, selanjutnya pengembang akan melakukan perencaan untuk kembali melakukan pengembangan dan kembali lagi ke tahapan *iteration planning.*

2.2.4 *Commercialization phase*

Pada fase terakhir dari metode pengembangan MASAM yang di adopsi pada penelitian ini terdapat *System test. System test* membahas tentang pengujian dari sistem yang telah dikembangkan. Pengujian ini meliputi *Acceptance Test* yang berhubungan pada pengguna sistem. Tabel 2.10 akan menjelaskan tentang activity dan *task* pada fase *commercialization*.

Commercialization Phase	System Test	Acceptance Test

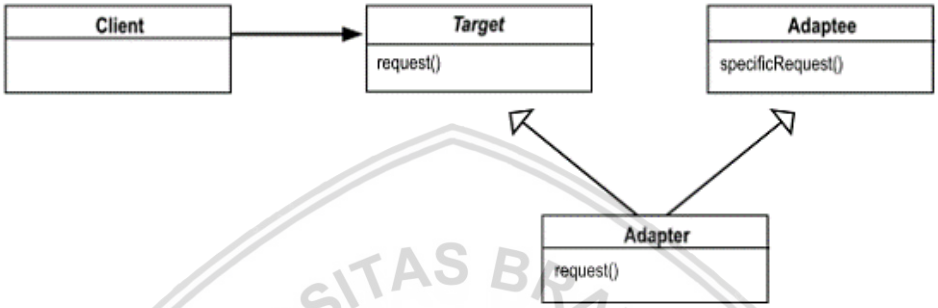
Tabel 2.10 Aktivitas Fase *Commercialization*

2.3 *Design Pattern*

Design pattern atau pola perancangan merupakan suatu praktik terbaik dan terdokumentasi untuk suatu solusi yang telah diterapkan di banyak lingkungan untuk memecahkan masalah yang muncul kembali dalam rangkaian situasi tertentu. Pola desain membahas tentang suatu cara yang ditemukan selama suatu studi atau selama melakukan pengembangan suatu sistem perangkat lunak dan dapat digunakan kembali untuk memecahkan suatu permasalahan yang mungkin muncul pada pengembangan sistem selanjutnya (Kuchana, 2004). Menurut Luedke (2015), penggunaan *design pattern* pada pengembangan aplikasi Android terbagi menjadi beberapa kategori yaitu *Creational patterns* yang terdiri dari builder, dependency injection dan singleton, Structural patterns yang terdiri dari adapter dan façade, lalu Behavioral pattern yang terdiri dari Command, Observer, Model View Controller dan Model View ViewModel. Pada penelitian ini akan digunakan dua buah *design pattern* yang umum digunakan yaitu adapter pattern dan command pattern.

2.3.1 Adapter Pattern

Adapter pattern merupakan salah satu *pattern* yang termasuk ke dalam *structural pattern*. Pola perancangan adapter digunakan untuk merubah antarmuka sebuah klas menjadi klas lain yang kompatibel dan dapat digunakan untuk menyampaikan informasi kepada pengguna. Adapter juga disebut sebagai wrapper atau pembungkus. Gambar 2.10 akan menunjukkan desain umum dari adapter pattern.



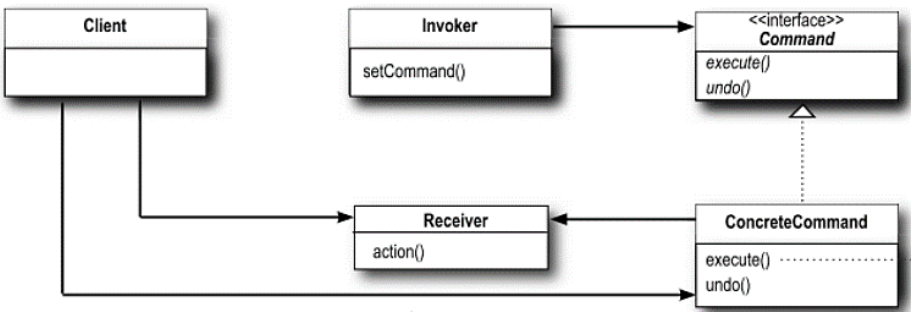
Gambar 2.1 Desain Umum *Adapter Pattern*

Sumber : (Gamma, et al., 1994)

Berdasarkan gambar 2.1 terdapat empat buah komponen, yaitu *client, target, adapter* dan *adaptee*. Client merupakan komponen yang bekerja sama dengan objek yang sesuai dengan antarmuka target. Sedangkan target mendefinisikan antarmuka spesifik yang digunakan oleh *client*. Lalu *adapter* merupakan komponen yang menyesuaikan antarmuka *Adaptee* untuk antarmuka *target*. Komponen yang terakhir adalah adaptee untuk mendefinisikan antarmuka yang sudah ada dan perlu beradaptasi.

2.3.2 Command Pattern

Command Pattern merupakan bagian dari behavioral pattern yang berfungsi untuk mendeklarasikan suatu class interface yang bertujuan untuk mengeksekusi suatu operasi. Gambar 2.2 merupakan desain umum dari *command pattern*.



Gambar 2.2 Desain Umum *Command Pattern*

Sumber : (Gamma, et al., 1994)

Gambar 2.2 menunjukkan bahwa terdapat lima buah komponen didalam desain *pattern command pattern*, yaitu *client, invoker, receiver, command* dan *concreate command*. Komponen *client* merupakan object yang meminta dilakukannya operasi. *Invoker* merupakan komponen yang digunakan untuk menjalankan *command* atau sebagai penerima atas permintaan *client*. Selanjutnya *receiver* berfungsi sebagai objek yang siap menerima operasi. *Command* merupakan interface untuk mengeksekusi operasi. Lalu *concreate command* sebagai pengubung antara command dan receiver.

2.4 Skala Likert

Skala likert adalah skala pengukuran yang dikembangkan oleh Likert (1932). Skala likert mempunyai empat atau lebih butir-butir pertanyaan yang dikombinasikan sehingga membentuk sebuah skor/nilai yang merepresentasikan sifat individu, misalkan pengetahuan, sikap, dan perilaku. Dalam proses analisis data, komposit skor, biasanya jumlah atau rataan, dari semua butir pertanyaan dapat digunakan. Penggunaan jumlah dari semua butir pertanyaan valid karena setiap butir pertanyaan adalah indikator dari variabel yang direpresentasikannya.

Skala yang paling mudah digunakan adalah skala likert. Skala likert menggunakan beberapa butir pertanyaan untuk mengukur perilaku individu dengan merespon 5 titik pilihan pada setiap butir pertanyaan, sangat setuju, setuju, cukup, tidak setuju, dan sangat tidak setuju (Likert 1932). Pada penelitian ini akan di gunakan 5 pilihan jawaban pada setiap pertanyaan yang di sediakan. Menurut Sugiyono (2011), terdapat penilaian dalam setiap masing-masing titik pilihan yang di sediakan skala likert. Skor penilaian untuk pilihan sangat setuju diberikan nilai 5, pilihan setuju diberikan nilai 4, pilihan cukup diberikan nilai 3, pilihan tidak setuju diberikan nilai 3 dan pilihan jawaban sangat tidak setuju akan diberikan nilai 1. Skor ideal merupakan skor yang digunakan untuk menghitung skor untuk menentukan *rating scale* dan jumlah seluruh jawaban. Untuk menghitung jumlah skor ideal (kriterium) dari seluruh item, digunakan rumus :

Skor Kriterium = Nilai skala x Jumlah responden

Semua hasil skor kriterium dijumlahkan dan selanjutnya akan ditentukan variabel X dan Y yang dimana X merupakan Skor tertinggi likert dikalikan jumlah responden (Skor Tertinggi Likert adalah 5), sedangkan Y merupakan Skor terendah likert dikalikan jumlah responden (Skor Terendah Likert adalah 1). Setelah didapatkan X dan Y maka untuk mendapatkan penilaian interpretasi dari responden akan di gunakan rumus berikut :

$$\textbf{Rumus Index \%} = \frac{\textbf{Total Skor Kriterium}}{Y} \textbf{ x 100\%}$$

Sehingga akan didapatkan nilai persentase penilaian responden. Untuk menyimpulkan hasil penilaian tersebut digunakan Tabel persentase penilaian dari Skala Likert yang digambarkan pada Tabel 2.12.

Tabel 2.11 Persentase Nilai Skala Likert

Jawaban	Keterangan
0% - 19,99%	Sangat Tidak Setuju
20% - 39,99%	Tidak Setuju
40% - 59,99%	Cukup
60% - 79,99%	Setuju
80% - 100%	Tidak Setuju

2.5 Populasi dan Sampel

Populasi dalam penelitian ini adalah masyarakat kota Malang yang pernah melakukan penyewaan kendaraan di Kota Malang. Untuk menentukan banyaknya sampel yang dibutuhkan pada penelitian ini mengacu pada ukuran sampel menurut Roscoe (Sugiyono, 2011) seperti berikut ini :

1. Ukuran sampel sebanyak 30 dan kurang dari 500 adalah tepat untuk kebanyakan penelitian.
2. sampel dipecah ke dalam subsampel (pria/wanita, junior/senior, dan sebagainya), ukuran sampel minimum 30 untuk tiap kategori adalah tepat.
3. Dalam penelitian *mutivariate* (termasuk analisis regresi berganda), ukuran sampel sebaiknya 10x lebih besar dari jumlah variabel dalam penelitian.
4. Untuk penelitian eksperimental sederhana dengan kontrol eskperimen yang ketat, penelitian yang sukses adalah mungkin dengan ukuran sampel kecil antara 10 sampai dengan 20.

Dalam penelitian ini, peneliti mengambil sampel sebanyak 31 responden yang dimana mengacu pada ukuran sampel menurut Roscoe. Sedangkan mengenai teknik pengambilan sampe pada penelitian ini akan digunakan teknik *Purposive Sampling*.

2.5.1 Purposive Sampling

Teknik *Purposive Sampling* termasuk ke dalam pengambilan sampel secara tidak acak (*Nonprobability/Nonrandom Sampling*). Dengan menggunakan teknik ini sampel akan dipilih dengan maksud dan tujuan tertentu karena peneliti menganggap sampel mempunyai informasi tertentu yang diperlukan untuk terlaksananya penelitian peneliti (Sugiyono, 2011).

Dalam penelitian ini, sampel yang akan dipilih adalah masyarakat yang pernah melakukan penyewaan kendaraan di Kota Malang untuk mendapatkan informasi

berupa pengalaman apa yang dirasakan dan didapatkan dalam sistem penyewaan kendaraan yang ada sekarang ini.

2.6 UML (*Unified Modelling Language*)

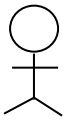
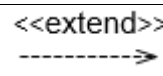
Unified Modelling Language adalah bahasa spesifikasi standar atau bahasa pemodelan yang digunakan dalam spesifikasi, visualisasi, membangun, dan mendokumentasikan artifak dari suatu sistem perangkat lunak. UML digunakan untuk memberikan pemahaman tentang suatu sistem yang akan dikembangkan (Rambaugh, et al., 2005).

UML juga dapat diartikan sebagai bahasa grafik standar yang bertujuan untuk memodelkan perangkat lunak berorientasi objek. James Rambaugh, Grady Booch dan Ivar Jacobson pertama kali mengembangkan bahasa pemodelan UML pada tahun 1990, yang dimana mereka mengembakan notasi mereka sendiri. UML juga disebut sebagai bahasa pemodelan yang digunakan untuk menggambarkan suatu perancangan perangkat lunak pada pengembangan perangkat lunak yang menggunakan *Object Oriented* (OO) (Fowler, 2003)

2.6.1 Use Case Diagram

Use case diagram adalah diagram yang digunakan untuk menggambarkan sekumpulan *use case*, aktor yang terlibat dalam sistem serta hubungan antara aktor dengan *use case*. *Use case* merupakan sekumpulan skenario untuk mendeskripsikan bagaimana suatu sistem bekerja pada sebuah situasi untuk mencapai tujuannya. Berikut ini merupakan simbol yang terdapat dalam *use case diagram*:

Tabel 2.12 Simbol pada Use case Diagram

Simbol	Deskripsi
	***Use case*** merupakan sekumpulan dari skenario untuk mendeskripsikan tentang cara kerja sistem pada sebuah situasi untuk mencapai tujuan.
	**Aktor** adalah manusia atau sistem lain yang dapat berinteraksi dengan sistem yang akan dibangun.
	Asosiasi berfungsi untuk menggambarkan interaksi antara *use case* dengan aktor
	Generalisasi digunakan ketika terdapat satu *use case* yang mirip dengan *use case* lain atau aktor yang mirip dengan aktor lain. Arah panah mengarah pada *use case* atau aktor yang menggeneralisasinya.
	***Extend*** adalah relasi *use case* tambahan ke sebuah *use case* dimana *use case* yang

	ditambahkan dapat berdiri sendiri. Arah panah menghadap pada *use case* yang ditambahkan.
<<include>> ---------->	***Include*** adalah relasi *use case* tambahan ke sebuah *use case* dimana *use case* yang ditambahkan memerlukan *use case* atau aktor yang menjadi generalisasinya.

2.6.2 Sequence diagram

Sequence diagram merupakan diagram yang menggambarkan alur kontrol diantara objek-objek yang terdapat dalam *use case*. Alur kontrol tersebut mendeskripsikan operasi-operasi yang berjalan diantara objek tersebut. Berikut ini merupakan simbol yang terdapat pada *sequence diagram* :

Tabel 2.13 Simbol pada Sequence Diagram

Simbol	Deskripsi
	**Aktor** adalah manusia atau sistem lain yang dapat berinteraksi dengan sistem yang akan dibangun.
	Pesan berfungsi untuk menyatakan suatu obyek memanggil operasi yang ada pada objek lain atau objek itu sendiri.
	**Objek** menyatakan objek yang berinteraksi.
	Return berfungsi untuk menyatakan nilai kembalian dari sebuah operasi yang dijalankan pada suatu objek dan juga menyatakan pembentukkan objek baru.
	Lifeline adalah menyatakan kehidupan dari suatu objek.

2.6.3 *Class diagram*

Class diagram digunakan untuk mendeskripsikan objek-objek dan relasi diantara objek-objek tersebut. Didalam *Class diagram* terdapat atribut-atribut dan operasi-operasi yang ada pada suatu objek. Berikut ini merupakan simbol yang terdapat pada *class diagram*:

Tabel 2.14 Simbol pada *Class diagram*

Simbol	Deskripsi
	**Nama_kelas** adalah objek pada sistem yang memiliki atribut dan operasi. **Atribut** berfungsi untuk memberikan karakterisitik atau ciri yang mendeskripsikan kelas. **Operasi** adalah perilaku yang mencerminkan suatu kelas.
	Asosiasi adalah merupakan hubungan antar kelas yang dimana suatu kelas memanggil objek dari kelas lainnya.
	Dependensi merupakan hubungan antar kelas yang dimana method suatu kelas membutuhkan objek dari kelas lain untuk dijadikan parameter.
	Generalisasi berfungsi untuk menunjukkan hubungan *inheritance* antara anak kelas dengan induk kelas.
	**Komposisi** merupakan hubungan antar kelas yang saling bergantung satu sama lain.
	**Agregasi** merupakan hubungan antar kelas yang dimana kelas tersebut merupakan bagian dari kelas lain tetapi tidak saling bergantung.

2.7 Android

Android adalah suatu arsitektur *open source* yang mencakup sistem operasi, kerangka aplikasi, kernel Linux, middleware dan aplikasi bersama dengan suatu API (Application Programming Interface) untuk menulis aplikasi *mobile* yang dapat memberikan suatu tampilan, fitur-fitur yang dapat digunakan oleh pengguna. Sekarang ini, para pengembang aplikasi *mobile* dapat memperluas *platform* Android untuk meningkatkan keandalan, kegunaan dan fitur dari suatu produk (Singh, et al., 2016). Tanpa kesulitan dan kompleksitas, pengembang Android dapat dengan mudah menulis kode aplikasi yang dapat membuat *hardware* ponsel lebih berguna dan *user friendly*. Android menggunakan sistem operasi Linux.

Terdapat beberapa sistem operasi perangkat *mobile* yaitu Android, iOS, Windows Phone, Blackberry, Symbian dan lain-lain. Menurut survei Statista (2017)

tentang penggunaan sistem operasi *mobile* di Indonesia, di dominasi dengan penggunaan sistem operasi Android sebesar 76,46% dan iOS 4,09%.

2.8 Firebase

Basis data *real-time* merupakan *cloud-hosted database* (basis data komputasi awan) yang disimpan dalam format JSON dan disinkronisasi secara terus menerus untuk selalu mendapat keterhubungan dengan klien (KhedKar & Thube, 2017). Firebase merupakan salah satu basis data yang bersifat *real-time* dan basis data *backend* untuk *platform* android, iOS dan aplikasi web. Firebase merupakan suatu layanan *BaaS (Backend as a Service)* yang disediakan oleh Google. Pada Firebase data disimpan sebagai JSON dan disinkronkan secara *real-time* ke setiap klien yang terhubung. Ketika membuat aplikasi lintas-platform dengan SDK Android, iOS, dan JavaScript, semua klien akan berbagi sebuah *instance Realtime Database* dan menerima *update* data terbaru secara otomatis.

Ketika Firebase API di implementasikan pada aplikasi android, iOS atau web, firebase dapat memberikan fitur dengan baris kode yang sederhana. Dalam penelitian ini akan digunakan Firebase dikarenakan penelitian ini mempunyai fungsi utama untuk melakukan pencarian kendaraan yang tersedia secara *real-time.* Firebase menyediakan beberapa fitur seperti Autentifikasi, *Hosting, Messaging, Analytics, Storage, Real-time Database, Crash Reporting, App Indexing* dan *Admob.* Namun pada penelitian ini hanya menggunakan beberapa fitur dari *firebase real-time database*, yaitu :

2.8.1 Autentifikasi

Autentifikasi merupakan fitur yang disediakan firebase untuk memberikan izin hanya kepada pengguna yang sudah diberikan izin untuk mengakses aplikasi. Firebase menyediakan fitur login dan registrasi melalui Gmail, Github, twitter, facebook dan juga memberikan layanan bagi pengembang untuk membuat autentifikasi sesuai dengan keinginan pengembang.

2.8.2 Storage

Berfungsi sebagai media penyimpanan dalam basis data *real-time*. *Storage* dapat menyimpan dan menampilkan konten seperti gambar, video dan audio secara langsung dari SDK klien, Unggah dan Unduh juga dilakukan pada proses *background.* Data yang disimpan hanya dapat diakses oleh pengguna yang telah diberikan akses pada aplikasi tersebut.

2.8.3 Real-time Database

Firebase Realtime Database adalah *database* yang di-*host* pada *cloud*. Data disimpan dalam bentuk JSON dan disinkronkan secara *real-time* ke setiap klien yang terhubung. Ketika pengembang membuat aplikasi lintas platform dengan SDK Android, iOS, dan JavaScript, semua klien akan berbagi sebuah *instance Realtime Database* dan menerima *update* data terbaru secara otomatis.

2.8.4 Crash Reporting

Firebase Crash Reporting membuat laporan mendetail mengenai error di aplikasi. Error dikelompokkan ke dalam masalah berdasarkan pelacakan tumpukan yang serupa, dan diurutkan berdasarkan tingkat keparahan dampaknya terhadap pengguna. Selain laporan otomatis, pengembang dapat membuat log khusus untuk membantu menemukan masalah yang menyebabkan aplikasi tidak bekerja.

2.9 Pengujian Perangkat Lunak

Suatu pengujian perangkat lunak bertujuan untuk menunjukkan bahwa program yang telah dibangun melakukan apa yang dimaksudkan dan untuk menemukan apakah ada kekurangan atau cacat dalam program sebelum digunakan oleh pengguna. Pengujian perangkat lunak digunakan untuk mengetahui kelengkapan, ketepatan dan kualitas serta mutu dari perangkat lunak yang dibangun. Dalam perangkat lunak yang berorientasi objek, pengujian dilakukan untuk menghasilkan sekumpulan layered subsystem yang membungkus atau mengenkapsulasi kelas-kelas yang saling berkolaborasi. Elemen sistem yang terdiri dari subsitem dan kelas dapat melakukan fungsi yang membantu untuk mencapai kebutuhan sistem. Hal ini sangat penting untuk menguji sebuah sistem object-oriented dan menemukan kesalahan-kesalahan yang mungkin terjadi dalam hubungan antar kelas dan komunikasi subsistem yang melewati layer arsitektur sistem.

Menurut Pressman (2010), perangkat lunak diuji dari dua buah perspektif yang berbeda, yaitu logika program internal dilaksanakan dengan menggunakan teknik kasus-uji *white box* dan kebutuhan perangkat lunak tersebut dilakukan dengan menggunakan teknik kasus uji *black box.* Diagram Use case dapat membantu mengungkapkan kesalahan pada tingkat validasi perangkat lunak. Selain pengujian *white box* dan *black box* terdapat beberapa pengujian yang dapat dilakukan dalam membangun suatu perangkat lunak, diantaranya yaitu pengujian *acceptance* dan *Test-Driven Development*.

2.9.1 Pengujian Unit

Pengujian uni merupakan tingkat pengujian yang digunakan untuk menguji tingkat terkecil yaitu suatu unit atau komponen dalam suatu perangkat lunak. Pengujian ini berfungsi untuk mengetahui apakah setiap unit atau komponen yang telah di implementasikan dalam sistem sesuai dengan perancangan yang sudah dilakukan, hal ini juga dikarenakan pengujian unit yang lebih berfokus pada sisi implementasi. Untuk melakukan pengujian unit, digunakan suatu teknik pengujian yaitu teknik pengujian *White box.*

Menurut Pressman (2010), Pengujian *White box* merupakan metode yang digunakan untuk perancangan kasus uji dengan menggunakan struktur kontrol dari perancangan prosedural untuk memperoleh kasus uji atau *test case.* Pengujian *white box* dilakukan pada potongan program yang dikembangkan. Pengujian whitebox juga disebut sebagai *Glass Box Testing, Code Base Testing* dan

Structural Testing. Dengan menggunakan pengujian *White box* dapat memperoleh kasus uji yang dapat menjamin bahwa semua jalur independen dalam modul pernah dieksekusi walaupun hanya satu kali, dapat mengerjakan semua keputusan secara logika pada sisi *true* dan *false*, dapat melakukan eksekusi pada semua perulangan yang sesuai dengan batasannya dan dapat mengerjakan seluruh struktur data internal untuk memastikan validitasnya. Didalam pengujian *White box* terdapat beberapa teknik diantaranya yaitu *Basis Path Testing.*

2.9.1.1 Basis Path Testing

Basis Path Testing merupakan teknik pengujian yang memungkinkan mendapatkan ukuran kompleks logical dari perancangan prosedural dan menggunakan ukuran ini untuk mendefinisikan himpunan jalur yang akan diuji. Teknik pengujian yang dilakukan oleh Tom McCabe ini menggunakan turunan kasus uji untuk mengeksekusi setiap pernyataan dalam kode programnya setidaknya satu kali selama pengujian. *Basis Path* menggunakan notasi g*raph* atau *flow graph* atau program *graph* untuk menggambarkan aliran kontrol. Selain *flow graph,* didalam *Basis path testing* juga terdapat *Independent Program Paths* yang merupakan jalur program yang menunjukkan setidaknya satu proses atau kondisi baru dalam program.

2.9.2 Pengujian Integrasi

Pengujian integrasi merupakan pengujian yang digunakan untuk menguji suatu kumpulan unit atau komponen yang saling berinteraksi dalam perangkat lunak. Tujuan dari dilakukan pengujian integrasi adalah untuk mengetahui apakah terdapat kesalahan antara unit atau komponen dalam sistem yang berinteraksi. Pada pengujian integrasi terdapat beberapa pendekatan pengujian yang dapat dilakukan *yaitu Top Down Testing, Bottom Up Testing* dan *Big Bang Testing*. (Singh & Khan, 2012). Sedangkan teknik yang digunakan pada pengujian integrasi adalah dengan menggunakan pengujian *whitebox* seperti yang telah dijelaskan pada sub bab pengujian unit.

2.9.3 Pengujian Validasi

Pengujian validasi adalah suatu pengujian yang bertujuan untuk mengetahui apakah sistem yang dikembangkan sudah memenuhi kebutuhan yang dibutuhkan. Pengujian validasi menggunakan teknik pengujian *Black box*, karena pengujian validasi lebih berfokus pada proses jalannya program serta berfokus pada input dan output apa yang di hasilkan oleh sistem.

Metode pengujian *Black box* yang juga diketahui sebagai pengujian behavioralmerupakan pengujian yang berfokus pada kebutuhan fungsional dari perangkat lunak. Didalam pengujian *black box*, penguji dapat mendefinisikan kumpulan kondisi input dan melakukan pengetesan pada spesifikasi fungsional program (Mustaqbal, et al., 2015). *Black box* Testing bukanlah solusi alternatif dari *White box* Testing tapi lebih merupakan pelengkap untuk menguji hal-hal yang tidak dicakup oleh *White box* Testing. *Black box* testing bertujuan untuk mengidentifikasikan bug-bug yang ada pada hasil, kinerja dan juga perilaku sistem.

Pengujian ini biasanya dilakukan oleh pihak penguji ketika integration test, system test, dan acceptance test, tetapi juga berguna untuk tahap yang lebih awal untuk membantu membangun unit test case dan component test case yang lebih baik.

Pengujian *black box* berfungsi untuk menemukan kesalahan kesalahan dalam fungsi yang salah atau hilang, kesalahan antarmuka sistem, kesalahan dalam struktur data atau akses database eksternal, kesalahan kinerja dan kesalah inisialisasi dan terminasi. Functional testing juga dapat disebut sebagai *black box* testing karena tidak ada pengetahuan dari logika internal sistem yang digunakan untuk membuat test case. Biasanya dalam pengujian fungsional, teknik validasi lebih digunakan untuk melakukan pengujian. Pada umumnya, pengujian *Black box* dilakukan pada tahap akhir pengujian. Hal ini dikarenakan pengujian black-box memperhatikan struktur kontrol, sehingga lebih berfokus pada domain informasi.

2.9.4 Pengujian *Acceptance*

Pengujian Acceptance atau pengujian penerimaan merupakan suatu pengujian yang dilakukan oleh calon pengguna dalam upaya untuk melaksanakan semua fitur dan fungsi yang diperlukan (Pressman, 2010). Menurut Perry (2006), Pengujian *Acceptance* dirancang untuk menentukan apakah perangkat lunak sesuai dengan pengguna untuk digunakan. Konsep ini dibangun untuk mencocokan desain yang telah dibangun kedalam proses bisnis pengguna dan proses uji yang bertujuan untuk mengetahui tingkat kecocokan sistem dengan pengguna. Didalam pengujian *Acceptance* terdapat empat komponen untuk mengukur kecocokan/kesesuaian tersebut, yaitu *Data, People, Strcuture* dan *Rules*. Didalam pengujian penerimaan ini, terdapat kebutuhan sistem yang harus dipenuhi. Hal ini dibagi menjadi empat kategori yaitu *Functionality, Performance, Interface Quality* dan *Overall Software Quality*. Untuk melakukan pengujian *Acceptance* dibutuhkan 5 orang responden yang dimana menurut Nielsen tidak harus membutuhkan jumlah pengguna untuk pengujian yang besar karena hal tersebut dapat menimbulkan pemborosan sumber daya (Nielsen, 2000).

2.9.4.1 Pengujian *Usability*

Pengujian usability merupakan suatu metode pengujian yang bertujuan untuk mengetahui kekurangan yang ada dalam sebuah produk dengan mengetahui pertimbangan dari pengguna. Pada pengujian *usability*, pengguna akan diberikan suatu tugas untuk menjalankan produk untuk mengetahui apakah pengguna menemukan suatu kesalahan ketika menggunakan produk tersebut (Alluri, 2012). Pengujian *usability* mengacu pada efektivitas, efisiensi dan kepuasan pengguna.

Menurut Nielsen, *usability* merupakan atribut kualitas yang dapat menilai tentang bagaimana kemudahan yang didapatkan oleh pengguna ketika menggunakan suatu produk. *Usability* didefinisikan melalui 5 komponen kualitas, yaitu :

1. *Learnability* : berfungsi untuk mengukur kemudahan yang didapatkan pengguna ketika menggunakan suatu produk untuk pertama kali.

2. *Efficiency* : berfungsi untuk mengukur secepat apa pengguna dapat melakukan tugasnya.
3. *Memorability* : berfungsi untuk mengukur ingatan pengguna tentang proses yang dilakukan pada produk yang digunakan untuk mencapain tujuannya.
4. *Error* : berfungsi untuk mengetahui error yang di lakukan oleh pengguna, sejauh mana akibat dari error tersebut, serta bagaimana pengguna dapat mengatasi error tersebut.
5. *Satisfaction* : berfungsi untuk mengetahui tanggapan atau perasaan pengguna ketika menggunakan produk secara keseluruhan.

Pada penelitian menggunakan metode *System Usability Scale* (SUS) pada pengujian *usability*. Berikut ini akan dijelaskan mengenai *System Usability Scale* (SUS).

a. System Usability Scale

Untuk melakukan pengujian *usability*, terdapat beberapa metode yang dapat dilakukan salah satunya adalah *System Usability Scale* (SUS). SUS merupakan salah satu metode *usability* yang paling banyak digunakan sekarang ini (Sharfina & Santoso, 2016). SUS pertama kali dikembangkan oleh John Broke pada tahun 1986. Gambar 2.3 menunjukkan *item* asli dari SUS.

No.	Original Item
1	I think that I would like to use this system.
2	I found the system unnecessarily complex.
3	I thought the system was easy to use.
4	I think that I would need the support of a technical person to be able to use this system.
5	I found the various functions in the system were well integrated.
6	I thought there was too much inconsistency in this system.
7	I would imagine that most people would learn to use this system very quickly.
8	I found the system very cumbersome to use.
9	I felt very confident using the system.
10	I needed to learn a lot of things before I could get going with this system.

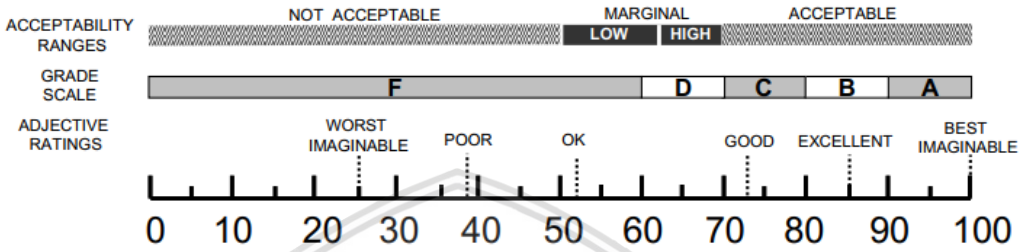
Gambar 2.3 *Original Item* SUS

Sumber : (Sharfina & Santoso, 2016)

Penilaian skor yang didapatkan dengan menggunakan SUS yaitu dengan memberikan skor untuk item nomor ganjil 1, 3, 5, 7 dan 9 dengan skor kontribusi posisi skala dikurangi dengan 1. Lalu untuk item nomor genap 2, 4, 6, 8 dan 10 skor kontribusinya adalah 5 dikurangi dengan nilai skala yang didapatkan pada tiap item genap. Kemudian dilakukan penjumlahan dari setiap skor item lalu dikalikan

dengan 2,5 sehingga akan menghasilkan skor SUS. Selanjutnya total skor SUS dibagi dengan jumlah responden, sehingga didapatkan nilai rata-rata skor SUS.

Didalam SUS, terdapat *range* penilaian untuk menentukan apakah sistem yang telah dikembangkan diterima oleh pengguna atau tidak. Skor SUS dengan rentang penilaian 0 – 50 termasuk kedalam kategori *"Not Acceptable",* skor SUS 51 – 70 termasuk dalam kategori *"Marginal"* dan skor SUS dengan rentang 71 – 100 termasuk dalam kategori *"Acceptable".*

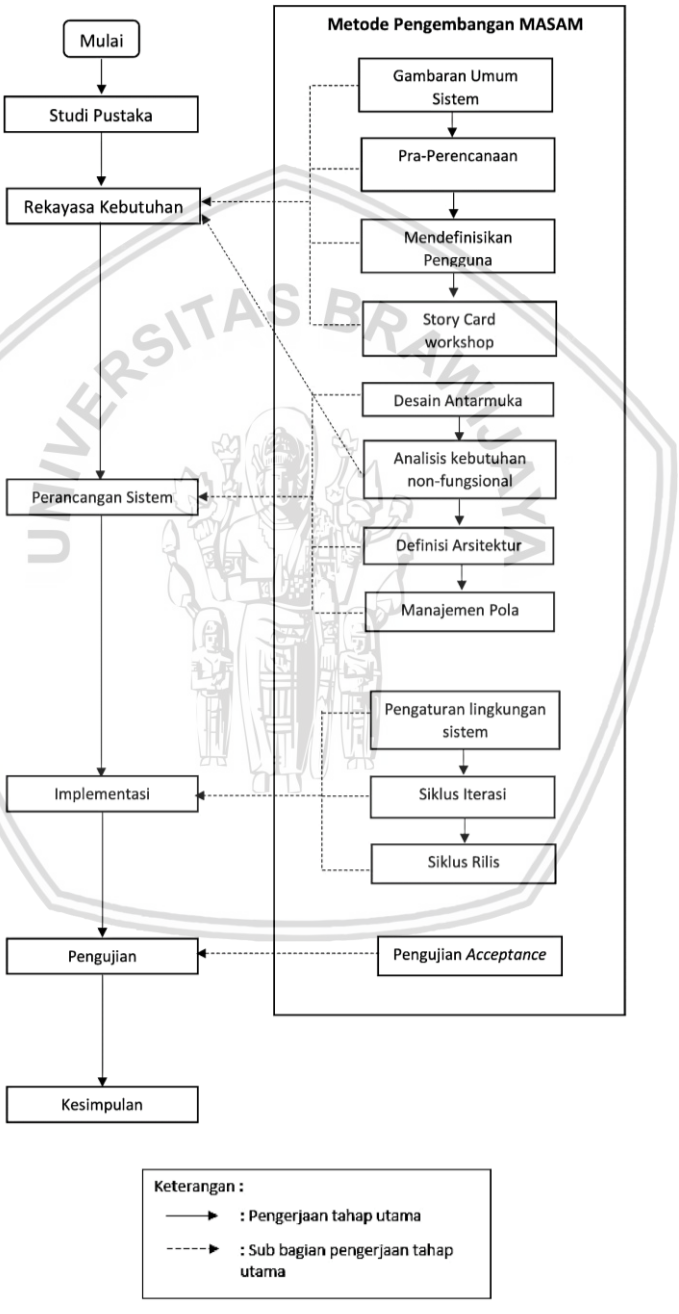


Gambar 2.4 Penilaian SUS

Sumber : (Bangor, et al., 2009)

BAB 3 METODOLOGI

Pada bab ini akan dijelaskan mengenai langkah-langkah yang akan dilakukan untuk mengembangkan suatu sistem penyewaan kendaraan kota Malang berbasis *mobile* dengan menggunakan metode pengembangan MASAM. Diagram alur metodologi penelitian menggunakan metode pengembangan MASAM diilustrasikan pada Gambar 3.1.



Gambar 3.1 Diagram Alur Metodologi Penelitian

3.1 Studi Pustaka

Studi Pustaka mengandung landasan teori yang terkait dengan penelitian. Hal ini penting dilakukan agar pengetahuan dasar untuk membangun suatu sistem dapat terpenuhi dengan baik. Pustaka dan teori yang berkaitan dengan penelitian ini adalah :

1. *Mobile* Application Software Agile Methodologies (MASAM)
    a. *Preparation phase*
    b. *Embodiment phase*
    c. *Development phase*
    d. *Commercialization phase*
2. *Design Pattern*
3. UML (*Unified Modelling Language*)
    a. *Use Case Diagram*
    b. *Sequence Diagram*
    c. *Class diagram*
4. Android
5. Firebase
6. Pengujian Perangkat Lunak
    a. Pengujian Unit
    b. Pengujian Integrasi
    c. Pengujian Validasi
    d. Pengujian *Acceptance*

3.2 Rekayasa Kebutuhan

Rekayasa kebutuhan merupakan sebuah tahapan untuk mengidentifikasi kebutuhan dari sistem penyewaan kendaraan kota malang. Penggalian kebutuhan dilakukan untuk mengetahui batasan dan prioritas dari kebutuhan-kebutuhan fungsionalitas sistem. Pada penelitian ini, di adopsi beberapa *task* dari metode pengembangan MASAM yang berhubungan dengan rekayasa kebutuhan. *Task* yang di adopsi pada bagian rekayasa kebutuhan adalah, gambaran umum (p*roduct summary*) dan pra-perencanaan (*pre-planning*) dari *activity grasping product*. Lalu *task* definisi pengguna (*user definition*) yang akan dijelaskan pada sub bab identifikasi aktor. Selanjutnya *task* yang di adopsi adalah *story card workshop* dari *activity user need understanding* pada fase *user need understanding. Story card workshop* akan dijelaskan pada sub bab elisitasi kebutuhan.

Gambaran umum (*product summary*) sistem akan dijelaskan pada bab rekayasa kebutuhan untuk memberikan penjelasan singkat tentang sistem yang

akan dibangun. Lalu pra-perencanaan (*pre-planning*) akan membahas tentang siapa saja *stakeholder* yang akan terlibat dalam pengembangan sistem ini. Selanjutnya akan didefinisikan siapa saja aktor dalam sistem yang akan di buat, hal ini termasuk dalam *task user definition*.

Elisitasi kebutuhan akan dilakukan dengan wawancara kepada calon 30 pengguna yang dimana akan dibagi menjadi 25 untuk sisi pelanggan dan 5 untuk sisi rental. Wawancara tersebut akan membahas tentang kebutuhan apa saja yang dibutuhkan oleh pengguna ketika akan dikembangkan suatu sistem penyewaan kendaraan Kota Malang. Serta membahas tentang urutan prioritas dari masing-masing kebutuhan untuk dapat mengetahui kebutuhan yang lebih penting. Lalu kebutuhan akan dituliskan ke dalam *story card*. Bagian ini akan di jelaskan pada sub bab spesifikasi kebutuhan. Gambar 3.3 menunjukkan templete dari *story card.*

Story Card

Number / ID	Type	Difficulty		Effort		Priority	Notes
		Before	After	Estimate	Spent		
	New Fix Enhance	Easy Moderate Hard	Easy Moderate Hard				
Description							
Date	**Status**		**Comment**				
	Defined						
	Implementing						
	Done						
	Verified						
	Postponed / Cancelled / Merged						

Gambar 3.2 *Template Story Card*

Namun pada penelitian ini tidak mengadopsi semua bagian yang ada dalam *story card.* Bagian *story card* yang akan digunakan dalam penelitian ini yaitu nomor *story*, judul *story,* priotitas kebutuhan dan deskripsi kebutuhan. Skala prioritas kebutuhan yang digunakan didalam penelitian ini adalah skala 1 sampai dengan 5 yang dimana menurut Lant (2010), untuk mengetahui prioritas suatu kebutuhan dapat menggunakan skala 1 sampai 5 dengan mempertimbangankan urgensi atau kepentingan dari setiap kebutuhan. Hal ini dimulai dari skala 1 yang mempunyai urgensi yang rendah sampai skala 5 yang mempunyai urgensi yang tinggi.

Setelah dilakukan elisitasi dan spesifikasi kebutuhan menggunakan *story card,* selanjutnya dapat dibuat pemodelan kebutuhan dengan menggunakan salah satu diagram UML yaitu *use case diagram.* Setelah itu, dilakukan pembuatan *use case scenario.* Pemodelan ini bertujuan untuk mengetahui kebutuhan apa saja yang akan dibangun dalam sistem lalu siapa saja yang akan berinteraksi dalam sistem serta mengetahui cara kerja sistem.

Pada tahap ini terdapat analisis kebutuhan non-fungsional dari sistem yang akan dibangun. Tahapan ini termasuk dalam fase *embodiment* dengan aktivitas *architecting* dalam metode pengembangan MASAM. Kebutuhan non-fungsional

dapat menggambarkan batasan-batasan apa saja yang terdapat dalam sistem penyewaan kendaraan Kota Malang.

3.3 Perancangan

Salah satu karakteristik metode pengembangan MASAM adalah, *UI prototyping* yang bertujuan untuk mendapatkan antarmuka yang sesuai dengan keinginan pengguna. Setelah tahapan rekayasa kebutuhan selesai dilakukan, selanjutnya adalah membuat perancangan antarmuka atau simulasi antarmuka. Alat bantu yang di gunakan untuk membuat simulasi antarmuka adalah *mocking bot*. Apabila pengguna menginginkan suatu penambahan fitur atau perubahan fitur, tahap pengembangan akan kembali ke tahapan *story card workshop* untuk menambahkan fitur yang ingin ditambah atau diubah. Maka dari itu alur antara *task story card workshop* dengan *task UI design* dapat ditunjukkan oleh Gambar 3.3.

Gambar 3.3 Siklus Iterasi *UI Prototyping*

Pada bagian *story card workshop* dan desain antarmuka akan dilakukan sebanyak dua kali iterasi dimana iterasi bertujuan untuk mendapatkan kebutuhan yang dibutuhkan oleh pengguna dari sisi pelanggan dan pemilik rental. Pada iterasi pertama berfungsi untuk mengetahui apakah ada penambahan atau pengurangan fitur maupun tampilan dari elisitasi kebutuhan yang sebelumnya telah dilakukan. Sedangkan iterasi kedua dilakukan untuk mengetahui tanggapan dari hasil perbaikan, penambahan atau pengurangan kebutuhan maupun tampilan dari iterasi pertama.

Perancangan sistem merupakan tahapan dimana peneliti melakukan perancangan mengenai kebutuhan fungsional sistem yang telah didapatkan dari tahapan Analisis kebutuhan. Perancangan sistem pada penelitian ini menerapkan beberapa *activity* yang ada dalam fase *embodiment* pada metode pengembangan MASAM yang terdiri dari definisi arsitektur (*Architecture definition)* dan Manajemen Pola (*Pattern Management)*. Salah satu karakteristik MASAM adalah adanya *domain knowledge* seperti yang sudah dijelaskan sebelumnya. Namun pada penelitian ini, bagian *domain knowledge* yang digunakan hanya *design pattern* atau pola perancangan. Penjelasan tentang definisi arsitektur sistem

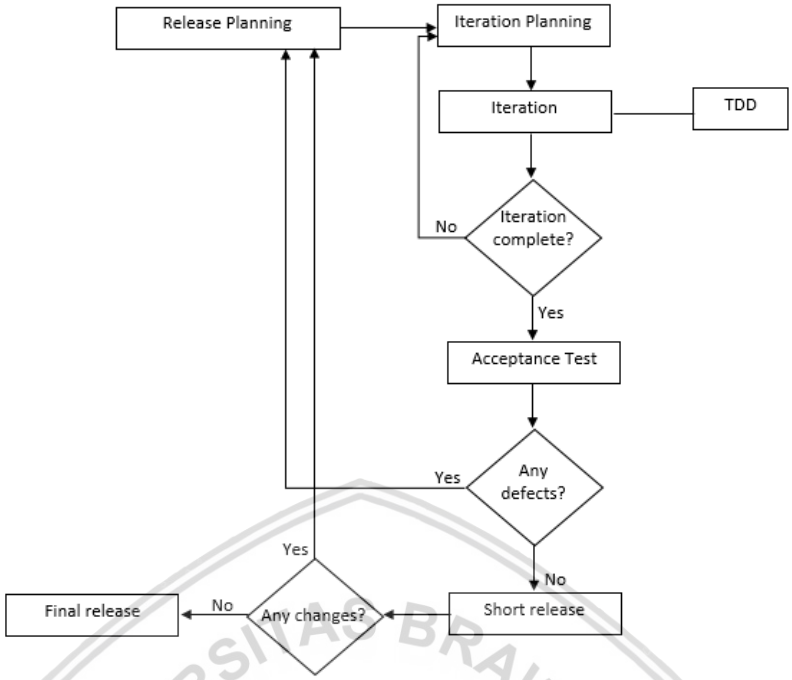
(*architecture definition*) akan di jelaskan pada sub bab perancangan arsitektural. Sedangkan penjelasan tentang manajemen pola (*pattern management*) akan dijelaskan pada sub bab manajemen pola perancangan.

Selain itu pada tahapan perancangan ini akan dilakukan perancangan sistem berdasarkan *object-oriented analysis* dengan menggunakan pemodelan UML (*Unified Modelling Language)*. Berikut merupakan urutan dari perancangan pada penelitian ini, yaitu :

1. Perancangan Arsitektural merupakan perancangan mengenai arsitektur perangkat lunak dari sistem yang akan dikembangkan.
2. Perancangan Basis Data merupakan perancangan struktur basis data untuk menetukan bagaimana struktur penyimpanan data dalam sistem.
3. Perancangan Diagram *Sequence* merupakan diagram yang digunakan untuk menggambarkan interaksi antar obyek serta komunikasi diantara obyek-obyek tersebut didalam suatu sistem.
4. Perancangan Diagram *Class* merupakan diagram yang digunakan untuk menujukkan kelas dan keterhubungannya di dalam suatu sistem. didalam perancangan Diagram *Class* juga terdapat perancangan *design pattern* yang merupakan perancangan yang bertujuan untuk mendesain kelas dan mengetahui bagaimana interaksi yang terjadi antar kelas sehingga kelas yang akan dibangun akan lebih *reusable.*
5. Manajemen Pola Perancangan menjelaskan tentang pola perancangan apa saja yang akan di implementasikan pada penelitian ini serta menjelaskan tentang kegunaannya pada pengembangan sistem.
6. Perancangan Antarmuka merupakan perancangan yang menjelaskan antarmuka dari sistem yang akan dikembangkan.
7. Perancangan Algoritme merupakan perancangan yang bertujuan untuk menjelaskan tentang langkah-langkah yang di lakukan untuk mengimplementasikan suatu kebutuhan fungsional.

3.4 Implementasi

Implementasi sistem merupakan tahapan pembangunan sistem yang mengacu pada perancangan sistem. Kebutuhan fungsional yang akan di implementasikan adalah daftar spesifikasi kebutuhan terakhir yang didapatkan apabila ada penambahan atau perubahan kebutuhan fungsional. Pada tahapan ini merupakan bagian dari fase *development* yang terdapat pada metode MASAM. Terdapat *task environment setup* dari *activity implementation preparation*. Dan beberapa *task* yang di adopsi dari *activity release cycle* diantarnya yaitu*, release planning, iteration cycle* (siklus iterasi) dan rilis. Gambar 3.4 menunjukkan siklus iterasi dan siklus rilis dari metode pengambangan MASAM yang di adopsi untuk penelitian ini.



Gambar 3.4 Siklus Iterasi dan Siklus Rilis

Implementasi perangkat lunak diawali dengan spesifikasi apa saja yang akan dibutuhkan dalam membangun sistem tersebut. Didalam fase Implementasi mengadopsi fase pengembangan yang ada dalam metode MASAM yaitu *Environment setup*. *Environment Set up* merupakan tahapan yang akan menjelaskan kebutuhan apa saja yang diperlukan dalam proses pengembangan sistem. Sehingga pencapaian kebutuhan sistem haruslah mencukupi untuk mendukung berjalannya penelitian yang optimal. Adapun sistem yang dibutuhkan berupa perangkat keras, perangkat lunak dan server. Dalam melakukan implementasi digunakan suatu metode pemrograman yaitu *Object Oriented Programming (*OOP). Sistem pada sisi client/pengguna dibuat dalam bentuk aplikasi perangkat lunak berbasis *mobile.* Sedangkan software yang digunakan untuk mendukung tahapan implementasi ini adalah Android Studio. Dan bagian server dan basis data, penelitian ini menggunakan *firebase real-time database*.

Setelah melakukan tahapan *environment set up* yang sudah dijelaskan sebelumnya, selanjutnya adalah melakukan perencanaan rilis untuk merancang ada berapa kali iterasi yang nantinya akan dilakukan dalam pembangunan sistem penyewaan kendaraan kota Malang berbasis Android. Pada penelitian ini iterasi akan dilakukan sebanyak dua kali iterasi. Selanjutnya adalah perencanaan iterasi yang membahas dan memilih tentang *story* apa saja yang akan di implementasikan terlebih dahulu sesuai dengan prioritasnya dan melakukan implementasi sesuai *story* tersebut. Selanjutnya pada iterasi yang akan dikerjakan, sebelum melakukan implementasi program akan dilakukan pengujian terhadap perancangan algoritme dengan menggunakan teknik pengujian *white-box* untuk mengetahui apakah ada *bug/error* sebelum melakukan implementasi program yang dimana bagian ini merupakan implementasi dari *Test Driven Development* (TDD). Setelah itu maka

akan dilakukan implementasi kode program berdasarkan *story* atau kebutuhan yang telah diprioritaskan. Selanjutnya, apabila semua iterasi sudah selesai dilakukan dan seluruh *story* sudah berhasil dilakukan pengujian dari sisi pengembang dengan menggunakan pengujian *acceptance* dengan tipe pengujian *alpha test* yang dilakukan oleh sisi pengembang dengan menggunakan teknik pengujian *blackbox*. Apabila tidak ada cacat/error maka akan dilakukan *short release* kepada pengguna untuk mengetahui tanggapan/*feedback* dari pengguna. Apabila ada perubahan, penambahan atau pengurangan fitur akan kembali ke *task release planning*, apabila tidak ada akan dilakukan *final release* untuk selanjutnya di lakukan pengujian akhir.

3.5 Pengujian dan Analisis

Tahap pengujian ini merupakan tahapan selanjutnya setelah implementasi selesai dilakukan. Pengujian dilakukan untuk mengetahui apakah sistem yang dikembangkan terdapat kesalahan dan juga untuk mengetahui apakah kebutuhan-kebutuhan pengguna sudah tepenuhi dengan baik. Tahapan pengujian dan analisis dilakukan setelah iterasi kedua telah selesai di implementasikan.

Didalam tahapan pengujian terdapat beberapa metode pengujian yang di adopsi dari metode pengembangan MASAM, *yaitu Acceptance test*. Selain itu, pengujian unit, pengujian integrasi dan pengujian validasi juga dilakukan pada pengujian sistem penyewaan kendaraan Kota Malang.

1. Pengujian Unit merupakan salah satu pengujian yang digunakan untuk menguji setiap klas yang didapatkan dari hasil perancangan. Pada penelitian ini pengujian unit dilakukan dengan pengujian *white box.* Jenis pengujian yang dilakukan adalah dengan menggunakan *basis path testing* untuk menguji kode program berdasarkan algoritma pada setiap metode yang ada di klas.
2. Pengujian Integrasi merupakan salah satu pengujian yang digunakan untuk menguji klas-klas yang saling berhubungan dalam sistem. Pada penelitian ini pengujian integrasi dilakukan dengan pengujian *white box.* Jenis pengujian yang dilakukan adalah dengan menggunakan *basis path testing* untuk menguji kode program berdasarkan algoritma pada setiap metode yang ada di klas.
3. Pengujian Validasi merupakan salah satu pengujian untuk memastikan bahwa sistem yang dikembangkan sudah memenihi kebutuhan pengguna. Pada pengujian ini akan menggunakan teknik pengujian *black box*.
4. Pengujian *Acceptance* merupakan salah satu pengujian yang digunakan untuk menguji seluruh kebutuhan fungsional sistem apakah sudah sesuai dengan kebutuhan yang telah didapatkan sebelumnya. Pengujian *acceptance* yang dilakukan pada pengujian ini adalah dengan parameter *usability* yang merupakan salah satu metode pengujian yang dilakukan oleh pengguna dengan tujuan untuk mengetahui kepuasan pengguna terhadap sistem yang telah dikembangkan dan juga untuk mengetahui

apakah sistem dapat diterima oleh pengguna atau tidak. Pengujian ini dilakukan terhadap 5 responden. Pengujian *usability* yang dilakukan pada penelitian ini adalah dengan menggunakan metode *System Usability Scale* (*SUS*), peneliti akan memberikan kuisioner berdasarkan *System Usability Scale* (*SUS*) untuk mengetahui tingkat kepuasaan pengguna terhadap sistem yang telah dikembangkan.

Dengan dilakukannya analisis dari hasil pengujian yang telah dilakukan akan didapatkan apa saja kekurangan dari sistem yang telah dikembangkan dan juga dapat mengetahui apa saja yang harus diperbaiki dalam sistem, sehingga hasil dari pengujian yang telah dilakukan akan dijadikan sebagai kesimpulan atau sebagai evaluasi untuk membuat perbaikan sistem.

3.6 Pengambilan Kesimpulan dan Saran

Tahap pengambilan kesimpulan merupakan tahap akhir dari penelitian ini. Kesimpulan diambil dari hasil pengujian dan analisis. Dengan adanya kesimpulan diharapkan dapat menjawab perumusan permasalahan yang telah dirumuskan. Sedangkan saran digunakan untuk memperbaiki kesalahan – kesalahan yang terjadi pada penelitian serta menjadi tolak ukur pertimbangan atas penelitian selanjutnya.

BAB 4 REKAYASA KEBUTUHAN

Bab ini menjelaskan fase analisis kebutuhan dari pengembangan sistem penyewaan kendaraan Kota Malang. Pada tahapan pertama akan dilakukan analisis kebutuhan yang terdiri dari beberapa langkah yang dimana langkah-langkah ini juga di adopsi dari metode pengembangan MASAM, yaitu menggambarkan gambaran umum sistem (*product summary*) yang akan dikembangkan, menentukan *stakeholder* yang merupakan *task* pra-perencanaan (*pra-planning*), melakukan proses identifikasi aktor (*user definition*), melakukan penggalian kebutuhan (*story card workshop*), melakukan spesifikasi kebutuhan dan melakukan pemodelan untuk menggambarkan kebutuhan dengan diagram *use case* serta melakukan analisis kebutuhan non-fungsional.

4.1 Analisis kebutuhan

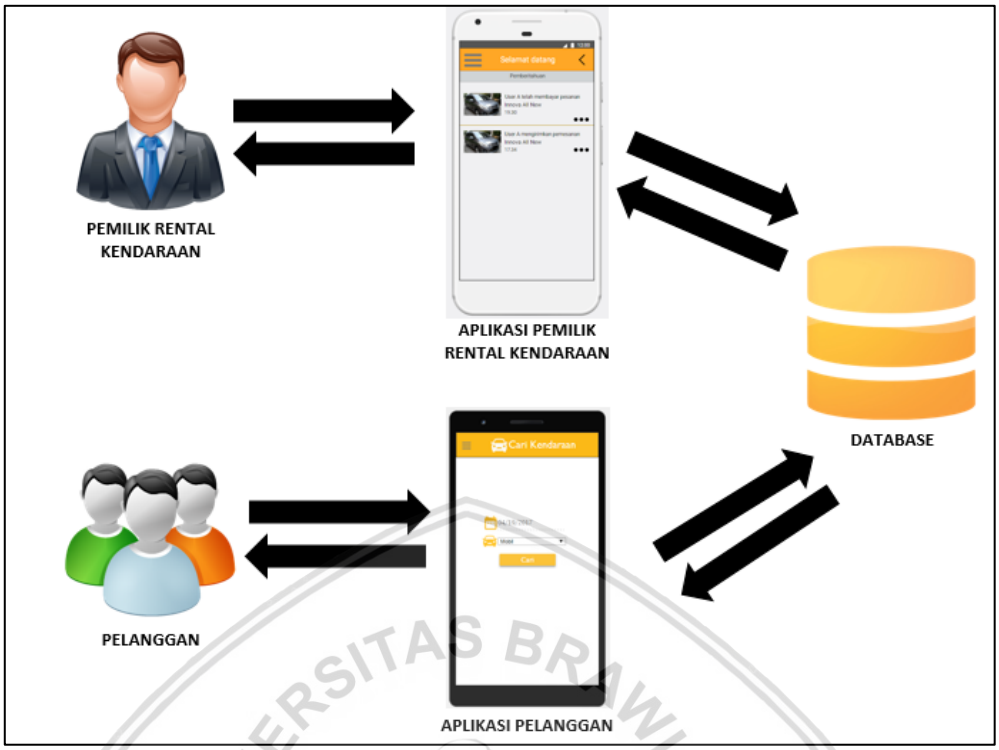
Proses analisis kebutuhan mengacu kepada gambaran umum sistem yang akan dikembangkan dan hasil pengumpulan serta penetapan kebutuhan sistem. Proses analisis kebutuhan ini diawali dengan menggambarkan gambaran umum sistem yang akan dikembangkan, melakukan proses identifikasi aktor, melakukan penggalian kebutuhan dengan menggunakan *story card*, melakukan spesifikasi dan melakukan pemodelan untuk menggambarkan kebutuhan dengan diagram *use case* serta melakukan analisis kebutuhan non-fungsional. Tujuan dari dilakukannya analisis kebutuhan adalah untuk mengetahui kebutuhan-kebutuhan yang harus disediakan oleh sistem untuk dapat memenuhi kebutuhan pengguna.

4.1.1 Gambaran Umum Sistem Penyewaan Kendaraan

Pembahasan gambaran umum sistem akan membahas mengenai sistem yang akan dikembangkan. Pembahasan ini terdiri dari bagaimana cara penggunaan sistem penyewaan kendaraan. Pada penelitian ini akan dibangun suatu sistem penyewaan kendaraan di Kota Malang yang diharapkan mampu memberikan kemudahan bagi masyarakat kota Malang maupun wisatawan dalam melakukan penyewaan kendaraan yang sesuai dengan keinginan dan ketersedian. Dan juga diharapkan sistem ini juga dapat membantu pemilik rental kendaraan dalam melakukan pengelolaan terhadap kendaraan yang disewakan.

4.1.1.1 Cara Penggunaan Sistem Penyewaan Kendaraan

Sistem penyewaan kendaraan memiliki urutan langkah kerja seperti yang terlihat pada Gambar 4.1.



Gambar 4.1 Urutan Langkah Kerja Sistem

Dari gambar 4.1 dapat dijelaskan bahwa pelanggan dan pemilik rental kendaraan dapat saling berinteraksi. Pelanggan dapat melakukan pencarian atas kendaraan yang tersedia dan dapat melakukan penyewaan atas kendaraan tersebut. Lalu sistem akan melakukan pengecekkan ke dalam *database* sistem untuk mengecek kendaraan yang tersedia dan juga untuk memuat segala informasi yang tersimpan an juga yang dibutuhkan oleh pelanggan. Ketika pelanggan berhasil melakukan penyewaan maka data penyewaan tersebut akan tersimpan didalam *database* sistem dan sisi pemilik rental dapat melihat data penyewaan tersebut yang selanjutnya rental akan melakukan konfirmasi atas penyewaan tersebut.

4.1.2 Pra-perencanaan

Bagian ini akan menjelaskan tentang siapa saja *stakeholder* yang akan terlibat. Pada *story card workshop* akan dilakukan penggalian kebutuhan awal pada masyarakat kota Malang yang sudah pernah melakukan penyewaan kendaraan di Kota Malang. Penggalian kebutuhan ini menggunakan wawancara dengan menyediakan penilaian 1-5, dengan nilai 1 sangat tidak setuju, nilai 2 tidak setuju, nilai 3 biasa saja, nilai 4 setuju dan nilai 5 sangat tidak setuju. Wawancara ini akan dilakukan kepada 30 calon pengguna dengan 25 untuk sisi pelanggan dan 5 untuk sisi pemilik rental.

Sedangkan penentuan *stakeholder* yang akan di wawancara untuk membahas spesifikasi kebutuhan dan perbaikan pada tahapan siklus iterasi *UI prototyping* dan siklus iterasi pengembangan, hanya mencakup 5 orang responden untuk mengurangi sumber daya yang ada (Nielsen, 2000).

4.1.3 Identifikasi Aktor

Pada bagian ini merupakan *task* definisi pengguna (*user definition*) yang terdapat pada metode pengembangan MASAM. Tahap ini merupakan tahapan untuk melakukan identifikasi aktor yang akan berinteraksi dalam sistem yang akan dibangun. Tabel 4.1 akan memperlihatkan aktor-aktor yang terlibat berserta penjelasannya.

Tabel 4.1 Identifikasi Aktor

Aktor	Deskripsi
Pengguna	Pengguna dapat menggunakan sistem penyewaan kendaraan Kota Malang melalui *smartphone* android masing-masing dan harus mendaftar sebagai pelanggan dalam sistem untuk dapat menggunakan layanan sistem.
Pelanggan	Pelanggan merupakan pengguna dari sistem penyewaan kendaraan Kota Malang yang telah terdaftar dalam sistem dan dapat menggunakan layanan yang disediakan dalam sistem yaitu melakukan pencarian kendaraan, melihat informasi kendaraan, melakukan penyewaan, pencarian rental terdekat dan menggunakan rute perjalanan menuju rental.
Pemilik rental kendaraan	Pemilik rental kendaraan merupakan pengguna dari sistem penyewaan kendaraan Kota Malang yang juga telah terdaftar dalam sistem dan dapat menggunakan layanan yang disediakan sebagai pemilik rental. Layanan yang disediakan yaitu dapat melihat penyewaan masuk, manajemen data kendaraan, melihat pemberitahuan/pemberitahuan dan manajemen profil rental.

4.1.4 Analisis data

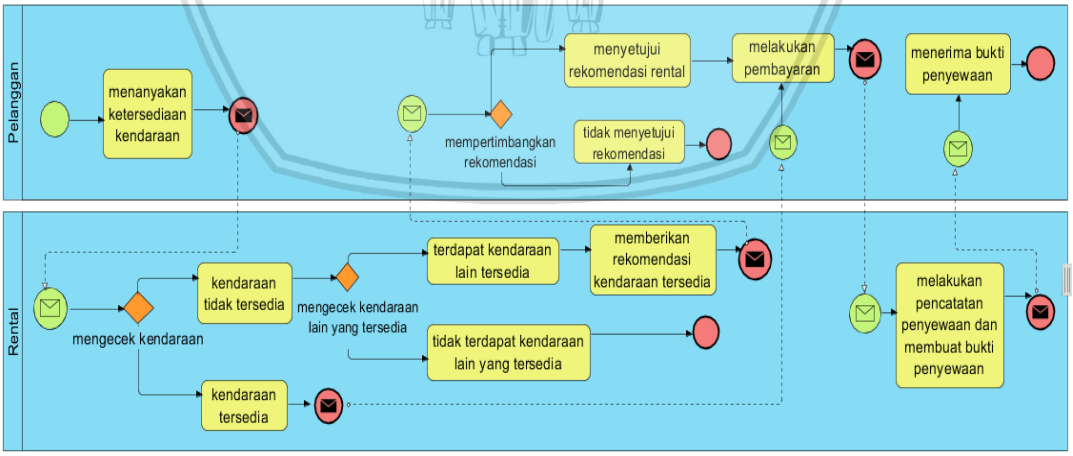
Analisis data bertujuan untuk mengetahui tentang bagaimana struktur penyimpanan data yang dibutuhkan sistem penyewaan kendaraan Kota Malang. Berikut ini merupakan struktur penyimpanan data berdasarkan analisis data pada sistem penyewaan Kendaraan Kota Malang :

1. Data pengguna berupa email dan *password* yang digunakan untuk registrasi dan *login*.
2. Data pelanggan berupa foto profil, NIK, nama lengkap, nomor telepon, alamat dan email.

3. Data rental kendaraan berupa nama rental kendaraan, nomor kontak, alamat rental kendaraan, email rental, kebijakan pemakaian, kebijakan pembatalan, kebijakan kelebihan waktu dan persyaratan penyewaan.
4. Data kendaraan berupa nama kendaraan, tipe kendaraan, harga sewa kendaraan, tipekendaraan, fasilitas kendaraan, foto kendaraan, jumlah penumpang dan jumlah kendaraan.
5. Data pengguna berupa nomor telepon yang digunakan untuk melakukan registrasi dan login dimana hal ini dihasilkan dari iterasi 2 dengan adanya perubahan kebutuhan. Alasan perubahan ini dilakukan oleh peneliti karena pemilihan nomor telepon atau nomor ponsel sebagai data yang digunakan untuk registrasi dan login dianggap lebih aman untuk melakukan verifikasi kebenaran data atau identitas pengguna. Hal ini dikarenakan nomor telepon saat ini telah diverifikasi oleh pemerintah sesuai dengan NIK penduduk Indonesia seperti yang tercantum dalam Peraturan Menteri Kominfo Nomor 14 Tahun 2017 (PERMENKOMINFO, 2017).

4.1.5 Elisitasi Kebutuhan

Pada bagian ini akan membahas tentang elisitasi kebutuhan dari sistem penyewaan kendaraan Kota Malang. Pada metode pengembangan MASAM, tahapan ini merupakan bagian yang terdapat pada *task story card workshop*. Proses pengumpulan kebutuhan perangkat lunak didapatkan dari hasil wawancara terhadap 30 responden yang dimana responden dibagi menjadi 25 sisi pelanggan yang melakukan penyewaan dan 5 pemilik rental kendaraan. Untuk mendapatkan kebutuhan awal yang akan terdapat pada sistem, pengembang melakukan definisi kebutuhan awal berdasarkan pada alur atau proses bisnis dari penyewaan kendaraan yang ada pada saat ini yang digambarkan oleh Gambar 4.2.



Gambar 4.2 *Business Process Model and Notation* Penyewaan Kendaraan

Selanjutnya pengguna akan memberikan nilai dari setiap definisi kebutuhan untuk mendapatkan nilai akhir untuk dijadikan pengukuran atas prioritas atau kepentingan tiap kebutuhan. Hasil elisitasi akan dihitung untuk mendapatkan nilai akhir. Nilai akhir tersebut akan dijadikan acuan untuk menentukan prioritas kebutuhan dan sebagai tolak ukur apakah kebutuhan tersebut dibutuhkan atau

tidak. Nilai akhir dari elisitasi kebutuhan ini didapatkan dengan menggunakan perhitungan Skala Likert. Tabel persentase penilaian Skala Likert yang terdapat pada Bab 2 akan dijadikan acuan pada penelitian ini, nilai akhir dari kebutuhan yang kurang dari 60% tidak akan diimplementasikan pada penelitian ini. Kebutuhan akan di tuliskan dengan menggunakan *story card* dan pengguna akan memberikan tingkat kepentingan (prioritas) dari kebutuhan tersebut. Hasil dari *story card workshop* dapat dilihat pada lampiran 1. Berikut ini merupakan tabel 4.2 yang menjelaskan tentang hasil elisitasi kebutuhan dari sisi pelanggan dan tabel 4.3 menjelaskan tentang hasil elisitasi kebutuhan dari sisi pemilik rental kendaraan.

Tabel 4.2 Hasil Elisitasi Kebutuhan Pelanggan

No	Nama Kebutuhan	Prioritas Kebutuhan					Nilai Akhir
		1	2	3	4	5	
1.	Saya ingin ditampilkan kolom pencarian kendaraan	0	0	1	10	14	90%
2.	Saya ingin ada filter untuk pencarian	0	0	1	13	11	88%
3.	Saya ingin ditampilkan informasi kendaraan	0	0	2	8	15	90%
4.	Saya ingin disediakan fitur penyewaan kendaraan	0	0	2	8	15	90%
5.	Saya ingin disediakan fitur pembayaran	0	0	2	8	15	90%
6.	Saya ingin ditampilkan peta lokasi rental	0	0	6	9	10	83%
7.	Saya ingin ditampilkan rute perjalanan menuju rental	0	0	3	14	8	84%
8.	Saya ingin ditampilkan lokasi rental terdekat	0	0	3	10	12	87%
9.	Saya ingin ditampilkan profil rental	0	0	8	10	7	79%
10.	Saya ingin disediakan fitur manajemen akun saya	0	1	11	10	3	72%
11.	Saya ingin disediakan fitur untuk memberi penilaian	1	0	7	11	6	76%

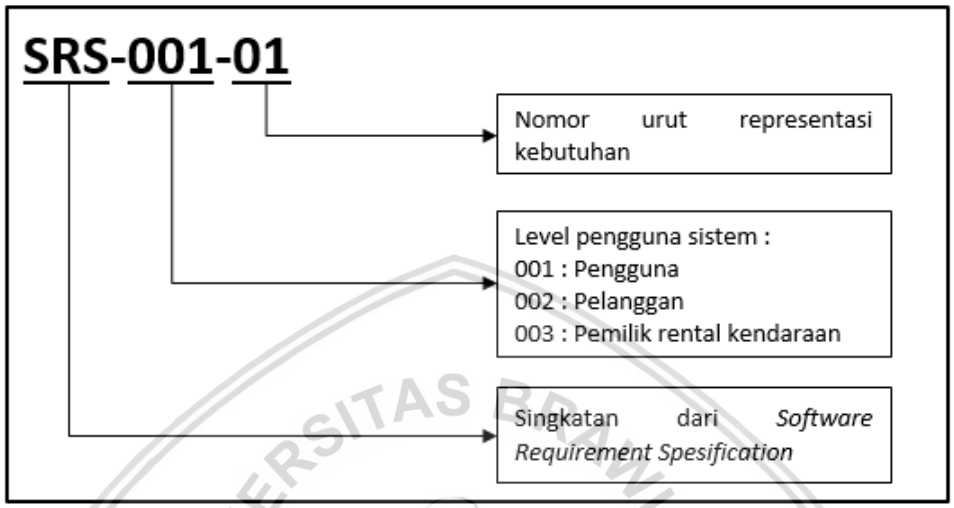
Tabel 4.3 Hasil Elisitasi Kebutuhan Pemilik Rental

No	Nama Kebutuhan	Prioritas Kebutuhan					Nilai Akhir
		1	2	3	4	5	
1.	Saya ingin ditampilkan daftar permintaan penyewaan	0	0	0	5	0	80%
2.	Saya ingin ditampilkan informasi detail penyewaan	0	0	0	5	0	80%
3.	Saya ingin disediakan fitur untuk melakukan konfirmasi pembayaran	0	0	0	5	0	80%
4.	Saya ingin disediakan fitur untuk melakukan konfirmasi pembatalan.	0	0	0	5	0	80%
5.	Saya ingin ditampilkan status penyewaan kendaraan	0	0	0	5	0	80%
6.	Saya ingin disediakan fitur *tracking* kendaraan yang sedang saya sewakan	0	3	0	2	0	56%
7.	Saya ingin disediakan fitur manajemen data kendaraan yang saya sewakan	0	0	0	5	0	80%
8.	Saya ingin disediakan fitur manajemen profil rental kendaraan saya	0	0	0	5	0	80%

Berdasarkan hasil dari elisitasi kebutuhan sisi pelanggan didapatkan nilai akhir dari semua kebutuhan lebih dari 60%, sehingga semua kebutuhan fungsional dari sisi pelanggan akan diimplementasikan. Sedangkan hasil elisitasi dari pemilik rental, terdapat kebutuhan yang mempunyai nilai akhir kurang dari 60%. Sehingga kebutuhan untuk melakukan *tracking* kendaraan tidak dimplementasikan pada sistem ini. Hal ini dikarenakan menurut 3 pengguna merasa tidak memerlukan fungsionalitas tersebut. Karena fitur pemantauan kendaraan sudah diterapkan sebelumnya oleh rental kendaraan.

4.1.6 Spesifikasi Kebutuhan

Pada tahap ini akan membahas tentang daftar kebutuhan yang terdiri dari spesifikasi kebutuhan fungsional dan kebutuhan non-fungsional yang terdapat dalam sistem. Berikut ini merupakan aturan penomoran kode fungsi yang digunakan dalam spesifikasi kebutuhan fungsional sistem pada Gambar 4.3.



Gambar 4.3 Aturan Penomoran

4.1.6.1 Spesifikasi Kebutuhan Fungsional

Spesifikasi kebutuhan akan dijelaskan sesuai dengan kebutuhan yang dibutuhkan oleh masing-masing aktor. Didalam penulisan kebutuhan fungsional terdapat tiga tahapan penulisan, yang terdiri dari penulisan pertama mengacu hasil elisitasi kebutuhan fungsional lalu penulisan kedua mengacu pada hasil iterasi analisis kebutuhan pertama dan penulisan ketiga mengacu pada hasil dari iterasi analisis kebutuhan kedua. Iterasi pertama berfungsi untuk mengetahui apakah ada penambahan atau pengurangan fitur maupun tampilan dari elisitasi kebutuhan yang sebelumnya telah dilakukan. Sedangkan iterasi kedua dilakukan untuk mengetahui tanggapan dari hasil perbaikan, penambahan atau pengurangan kebutuhan maupun tampilan dari iterasi pertama. Selanjutnya ketiga tahapan tersebut didapatkan dengan melakukan wawancara dan menuliskannya kedalam *story card* untuk mengetahui prioritas atau kepentingan dari setiap kebutuhan fungsional.

Spesifikasi kebutuhan fungsional pengguna ditunjukkan pada Tabel 4.4 dan Tabel 4.5 akan menjelaskan kebutuhan fungsional sisi pengguna dari hasil iterasi kedua.

Tabel 4.4 Spesifikasi Kebutuhan Fungsional Pengguna

Kode Fungsi	Nama Fungsi	Deskripsi	Use Case
SRS_001_01	Registrasi	Sistem harus menyediakan fasilitas untuk registrasi sehingga pengguna dapat menjadi	Registrasi

		pelanggan dan bisa menggunakan fungsionalitas sistem.	
SRS_001_02	Login	Sistem harus menyediakan fasilitas login sehingga pengguna dapat masuk kedalam sistem.	Login
SRS_001_03	Mengisi biodata	Sistem harus menyediakan fasilitas mengisi biodata pengguna dan menyimpannya dalam sistem.	Mengisi biodata

Tabel 4.5 Spesifikasi Kebutuhan Fungsional Pengguna Iterasi 2

Kode Fungsi	Nama Fungsi	Deskripsi	Use Case
SRS_001_01	Registrasi	Sistem harus menyediakan fasilitas untuk registrasi sehingga pengguna dapat menjadi pelanggan dan bisa menggunakan fungsionalitas sistem.	Registrasi
SRS_001_02	Login	Sistem harus menyediakan fasilitas login sehingga pengguna dapat masuk kedalam sistem.	Login
SRS_001_04	Autentifikasi Dengan Nomor Telepon	Sistem menyediakan fasilitas registrasi dan login dengan menggunakan nomor telepon pengguna yang aktif.	Autentifikasi Dengan Nomor Telepon

Pada Siklus *UI prototyping* iterasi 1 pada sisi pengguna tidak ada perubahan atau penambahan kebutuhan. Tetapi ketika dilakukan siklus *UI protoyping* iterasi 2 dari sisi pengembang terdapat penambahan dan pengurangan kebutuhan fungsional. SRS_001_04 merupakan penambahan kebutuhan fungsional Autentifikasi Dengan Nomor Telepon yang berfungsi untuk melakukan registrasi dan login ke dalam sistem hanya dengan menggunakan nomor telepon yang aktif dan pengguna hanya perlu mengisi kode yang di kirimkan ke ponsel pengguna melalui SMS. Hal ini berdampak pada pengurangan kebutuhan fungsional SRS_001_01 dan SRS_001_02 yang di hapus dari spesifikasi kebutuhan sistem.

Selanjutnya tabel 4.6 akan menjelaskan tentang kebutuhan fungsional sisi pelanggan yang didapatkan dari hasil elisitasi kebutuhan pelanggan. Dan tabel 4.7 akan menjelaskan tentang penambahan maupun perubahan kebutuhan fungsional yang didapatkan dari hasil iterasi pertama untuk menyempurnakan kebutuhan yang sebelumnya didapatkan dari hasil elisitasi kebutuhan.

Tabel 4.6 Spesifikasi Kebutuhan Fungsional Pelanggan

Kode Fungsi	Nama Fungsi	Deskripsi	Use Case
SRS_002_01	Melakukan pencarian kendaraan	Sistem harus menyediakan fasilitas pencarian kendaraan berdasarkan tanggal penyewaan dan kategori kendaraan.	Melakukan pencarian kendaraan
SRS_002_02	Melihat daftar kendaraan	Sistem harus menyediakan fasilitas untuk melihat daftar seluruh kendaraan yang tersedia dan sesuai dengan pencarian pelanggan.	Melihat daftar kendaraan
SRS_002_03	Melihat detail informasi kendaraan	Sistem harus menyediakan fasilitas untuk melihat informasi kendaraan yang telah dipilih oleh pelanggan.	Melihat informasi kendaraan
SRS_002_04	Melakukan penyewaan kendaraan	Sistem harus menyediakan fasilitas untuk melakukan penyewaan kendaraan terhadap kendaraan yang tersedia dan sesuai dengan pencarian pelanggan.	Melakukan penyewaan kendaraan
SRS_002_05	Melihat informasi detail penyewaan	Sistem harus menyediakan fasilitas untuk melihat informasi detail penyewaan kendaraan.	Melihat informasi detail penyewaan
SRS_002_06	Melakukan pembayaran	Sistem harus menyediakan fasilitas untuk melakukan pembayaran terhadap kendaraan yang di pesan pelanggan.	Melakukan pembayaran
SRS_002_07	Unggah bukti pembayaran	Sistem harus menyediakan fasilitas untuk melakukan unggah bukti pembayaran.	Unggah bukti pembayaran

SRS_002_08	Melakukan pembatalan	Sistem harus menyediakan fasilitas untuk melakukan pembatalan atas penyewaan kendaraan.	Melakukan pembatalan
SRS_002_09	Melihat status penyewaan	Sistem harus menyediakan fasilitas untuk melihat informasi mengenai status penyewaan kendaraan.	Melihat status penyewaan
SRS_002_10	Melihat rental kendaraan terdekat	Sistem harus menyediakan fasilitas untuk melihat rental kendaraan terdekat dari pelanggan.	Melihat rental kendaraan terdekat
SRS_002_11	Melakukan *filter* pencarian	Sistem harus menyediakan fasilitas untuk melakukan *filter* berdasarkan rentang harga.	Melakukan *filter* pencarian
SRS_002_12	Melihat peta lokasi rental	Sistem harus menyediakan fasilitas untuk melihat peta lokasi rental kendaraan.	Melihat peta lokasi rental
SRS_002_13	Melihat rute perjalanan menuju rental	Sistem harus menyediakan fasilitas untuk mengetahui rute perjalanan menuju rental.	Melihat rute perjalanan menuju rental
SRS_002_14	Melihat profil rental	Sistem harus menyediakan fasilitas untuk melihat profil rental kendaraan.	Melihat profil rental
SRS_002_15	Memberikan penilaian	Sistem harus menyediakan fasilitas untuk memberikan penilaian kepada rental kendaraan berupa *rating* dan ulasan.	Memberikan penilaian
SRS_002_16	Melihat profil pelanggan	Sistem harus menyediakan fasilitas untuk melihat profil pelanggan kendaraan.	Melihat profil pelanggan
SRS_002_17	Memperbarui profil pelanggan	Sistem harus menyediakan fasilitas untuk memperbarui data pada profil pelanggan.	Memperbarui profil pelanggan
SRS_002_18	Logout	Sistem harus menyediakan fasilitas untuk logout yang	Logout

		berfungsi untuk keluar dari sistem.	

Tabel 4.7 Spesifikasi Kebutuhan Pelanggan Iterasi 1

Kode Fungsi	Nama Fungsi	Deskripsi	Use Case
SRS_002_01	Melakukan Pencarian Kendaraan	Sistem harus menyediakan fasilitas pencarian kendaraan berdasarkan tanggal sewa, tanggal kembali penyewaan, kategori kendaraan dan jumlah kendaraan yang diinginkan oleh pelanggan.	Melakukan Pencarian Kendaraan
SRS_002_10	Melihat rental kendaraan terdekat	Sistem harus menyediakan fasilitas untuk melihat rental kendaraan terdekat dari pelanggan.	Melihat rental kendaraan terdekat
SRS_002_11	Melakukan *Filter* Pencarian	Sistem harus menyediakan fasilitas untuk melakukan *filter* berdasarkan ketersediaan kendaraan yang di lengkapi fasilitas dengan sopir, tanpa sopir, dengan bahan bakar atau tanpa bahan bakar.	Melakukan *Filter* Pencarian
SRS_002_19	Melihat Penilaian dan Ulasan Rental	Sistem harus menyediakan fasilitas untuk melihat penilaian dan ulasan terhadap rental kendaraan.	Melihat Penilaian dan Ulasan Rental

Berdasarkan tanggapan 5 calon pengguna terhadap *UI prototyping* iterasi pertama didapatkan perubahan dan penambahan kebutuhan fungsional. Pada SRS_002_01 terdapat perubahan pada kebutuhan fungsional pencarian kendaraan untuk melakukan pencarian kendaraan berdasarkan tanggal sewa, tanggal kembali penyewaan, kategori kendaraan dan jumlah kendaraan yang diinginkan oleh pelanggan yang dimana sebelumnya kebutuhan fungsional ini

hanya menggunakan pencarian berdasarkan tanggal sewa dan kategori pencarian. Perubahan ini didasarkan pada 4 calon pengguna menginginkan penambahan fungsionalitas yang bertujuan untuk memudahkan pencarian apabila pengguna ingin menyewa kendaraan lebih dari satu hari dan juga lebih dari satu kendaraan.

Selanjutnya adalah spesifikasi kebutuhan SRS_002_02 yang merupakan penambahan fitur dalam kebutuhan fungsional pelanggan. Penambahan kebutuhan fungsional ini di karenakan sebanyak 3 pengguna membutuhkan fitur untuk menampilkan hasil penilaian dan ulasan dari pengguna yang telah melakukan penyewaan pada rental tertentu, sehingga penambahan fitur ini diperlukan di dalam sistem.

Penambahan fitur pada kebutuhan fungsional SRS_002_19 dilakukan berdasarkan 3 pengguna menginginkan adanya penambahan kategori *filter* yaitu dapat melakukan filter terhadap hasil pencarian kendaraan yang hanya menyediakan fasilitas dengan sopir atau tanpa sopir dan dengan bahan bakar atau tanpa bahan bakar. Sehingga dengan adanya penambahan fitur ini dapat memudahkan pengguna dalam mengetahui fasilitas kendaraan apa saja yang ada dan sesuai dengan keinginan serta kebutuhan pengguna.

Sedangkan kebutuhan fungsional dengan kode SRS_002_10 terdapat perubahan skenario atau alur dari sisi pengembang. Hal ini dikarenakan untuk sekarang ini firebase belum menyediakan *library* untuk mengurutkan suatu tempat terdekat dari posisi pengguna berdasarkan *latitude* dan *longitude* dari keberadaan tempat tersebut. Tetapi firebase menyediakan suatu *library* bernama geofire yang bersifat *open source* untuk dapat melihat suatu tempat terdekat berdasarkan radius (km) tertentu didalam tampilan suatu peta. Sehingga kebutuhan fungsional SRS_002_10 akan mengalami perubahan dalam skenario *usecase.*

Tabel 4.8 Spesifikasi Kebutuhan Pelanggan Iterasi 2

Kode Fungsi	Nama Fungsi	Deskripsi	Use Case
SRS_002_20	Menerima Pemberitahuan	Sistem menampilkan pesan pemberitahuan pada kondisi tertentu.	Menampilkan Pemberitahuan

Pada iterasi kedua tidak terdapat penambahan atau perubahan dari calon pelanggan. Penambahan fungsi tersebut adalah SRS_002_20 dengan nama fungsi menerima pemberitahuan. Fungsi ini bertujuan untuk memudahkan pelanggan untuk mengetahui apabila terdapat konfirmasi penyewaan dari rental kendaraan dan kondisi lainnya.

Selanjutnya adalah Tabel 4.9 yang akan menjelaskan spesifikasi kebutuhan pemilik rental dari hasil elisitasi kebutuhan. Lalu tabel 4.10 akan menjelaskan spesifikasi kebutuhan pemilik rental dari hasil iterasi kedua tentang penambahan maupun perubahan yang terdapat kebutuhan fungsional pemilik rental.

Tabel 4.9 Spesifikasi Kebutuhan Pemilik Rental

Kode Fungsi	Nama Fungsi	Deskripsi	Use Case
SRS_003_01	Menambah data kendaraan	Sistem harus menyediakan fasilitas untuk menambah data kendaraan yang disewakan.	Menambah data kendaraan
SRS_003_02	Melihat data kendaraan	Sistem harus menyediakan fasilitas untuk melihat data kendaraan yang disewakan.	Melihat data kendaraan
SRS_003_03	Memperbarui data kendaraan	Sistem harus menyediakan fasilitas untuk memperbarui data kendaraan uang disewakan.	Memperbarui data kendaraan
SRS_003_04	Menghapus data kendaraan	Sistem harus menyediakan fasilitas untuk menghapus kendaraan yang disewakan.	Menghapus data kendaraan
SRS_003_05	Melihat daftar penyewaan pelanggan	Sistem harus menyediakan fasilitas untuk melihat penyewaan sewa kendaraan yang dikirimkan dari pelanggan terhadap pemilik rental.	Melihat daftar penyewaan pelanggan
SRS_003_06	Melihat informasi detail penyewaan pelanggan	Sistem harus menyediakan fasilitas untuk melihat informasi detail penyewaan kendaraan.	Melihat informasi detail penyewaan pelanggan
SRS_003_07	Melakukan konfirmasi pembayaran	Sistem harus menyediakan fasilitas untuk melakukan konfirmasi pembayaran terhadap penyewaan sewa kendaraan yang telah dibayarkan pelanggan.	Melakukan konfirmasi pembayaran
SRS_003_08	Melakukan konfirmasi penyewaan selesai	Sistem harus menyediakan fasilitas untuk melakukan konfirmasi apabila penyewaan sedang berlangsung.	Melakukan konfirmasi penyewaan selesai

SRS_003_09	Melakukan konfirmasi pembatalan	Sistem harus menyediakan fasilitas untuk melakukan konfirmasi pembatalan terhadap penyewaan sewa yang ingin dibatalkan oleh pelanggan.	Melakukan konfirmasi pembatalan
SRS_003_10	Melihat status penyewaan	Sistem harus menyediakan fasilitas untuk melihat informasi mengenai status penyewaan kendaraan.	Melihat status penyewaan
SRS_003_11	Melihat profil pelanggan	Sistem harus menyediakan fasilitas untuk melihat profil pelanggan.	Melihat profil pelanggan
SRS_003_12	Melihat profil rental kendaraan	Sistem harus menyediakan fasilitas untuk melihat profil pemilik rental kendaraan.	Melihat profil rental kendaraan
SRS_003_13	Memperbarui profil rental	Sistem harus menyediakan fasilitas untuk memperbarui profil pemilik rental.	Memperbarui profil rental
SRS_003_14	Logout	Sistem harus menyediakan fasilitas untuk logout yang berfungsi untuk keluar dari sistem.	Logout

Tabel 4.10 Spesifikasi Kebutuhan Pemilik Rental Iterasi 1

Kode Fungsi	Nama Fungsi	Deskripsi	Use Case
SRS_003_15	Melihat Penilaian dan Ulasan	Sistem harus menyediakan fasilitas untuk melihat penilaian dan ulasan terhadap renltal kendaraan.	Melihat Penilaian dan Ulasan Rental
SRS_003_16	Menambah Rekening Pembayaran	Sistem harus menyediakan fasilitas untuk menambah rekening pembayaran rental kendaraan.	Menambah Rekening Pembayaran
SRS_003_17	Mengubah Rekening Pembayaran	Sistem harus menyediakan fasilitas untuk mengubah rekening pembayaran rental kendaraan.	Mengubah Rekening Pembayaran

SRS_003_18	Menghapus Rekening Pembayaran	Sistem harus menyediakan fasilitas untuk menghapus rekening pembayaran rental kendaraan.	Menghapus Rekening Pembayaran

Hasil iterasi pertama didapatkan dari tanggapan yang diberikan 5 pengguna sisi pemilik rental terhadap UI *prototyping* pertama. Selanjutnya adalah SRS_003_15 merupakan penambahan kebutuhan fungsional Melihat Penilaian dan Ulasan. Penambahan kebutuhan fungsional ini didasarkan pada permintaan pengguna sebanyak 5 membutuhkan fitur ini. Sedangkan SRS_003_16, SRS_003_17 dan SRS_003_18 merupakan kebutuhan fungsional yang dibutuhkan oleh pemilik rental untuk mengelola rekening pembayaran yang tersimpan dalam sistem. Sebanyak 5 pengguna membutuhkan ketiga fitur ini untuk menambah, mengubah dan menghapus rekening pembayaran pada sistem.

Tabel 4.11 Spesifikasi Kebutuhan Pemilik Rental Iterasi 2

Kode Fungsi	Nama Fungsi	Deskripsi	Use Case
SRS_003_19	Menerima Pemberitahuan	Sistem menampilkan pesan pemberitahuan pada kondisi tertentu.	Menampilkan Pemberitahuan
SRS_003_20	Melakukan penolakkan pembayaran	Sistem harus menyediakan fasilitas untuk melakukan penolakkan pembayaran	Melakukan penolakkan pembayaran

Pada iterasi kedua terdapat satu penambahan kebutuhan fungsional dari calon pengguna sisi pemilik rental, yaitu SRS_003_20 dengan nama fungsi melakukan penolakkan pembayaran. Fungsi ini berfungsi bagi pemilik rental untuk melakukan penolakkan pembayaran apabila terdapat kondisi tertentu. Sedangkan dari sisi pengembang terdapat beberapa penambahan kebutuhan fungsional. Penambahan fungsi yang tersebut adalah SRS_002_19 dengan nama fungsi menerima pemberitahuan. Fungsi ini bertujuan untuk memudahkan rental untuk mengetahui apabila terdapat permintaan penyewaan baru dan kondisi lainnya.

4.1.6.2 Spesifikasi Kebutuhan Non-Fungsional

Spesifikasi kebutuhan Non-fungsional dari sistem penyewaan kendaraan Kota Malang dijelaskan pada Tabel 4.12.

Tabel 4.12 Spesifikasi Kebutuhan Non-Fungsional

Parameter	Deskripsi Kebutuhan

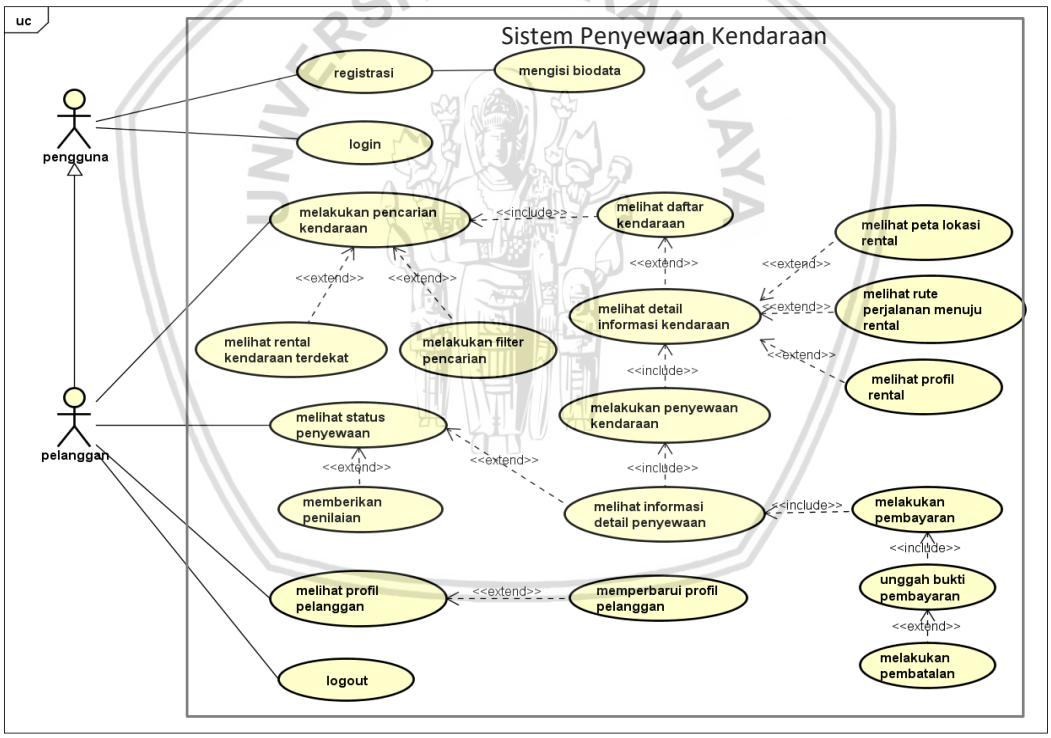
Usability	Tampilan antarmuka sistem dirancang semudah mungkin sehingga pengguna sistem tidak mengalami kesulitan dalam menggunakannya.

4.1.7 Diagram *Use Case*

Diagram *Use Case* merupakan diagram yang digunakan untuk menggambarkan perilaku aktor dengan sistem. Tujuan dari pembuatan diagram *use case* adalah untuk menggambarkan seluruh kebutuhan sistem yang akan dikembangkan sehingga siapa pun yang melihat dapat memahaminya. Diagram *use case* ini dibuat berdasarkan dari kebutuhan fungsional sistem yang telah didapatkan.

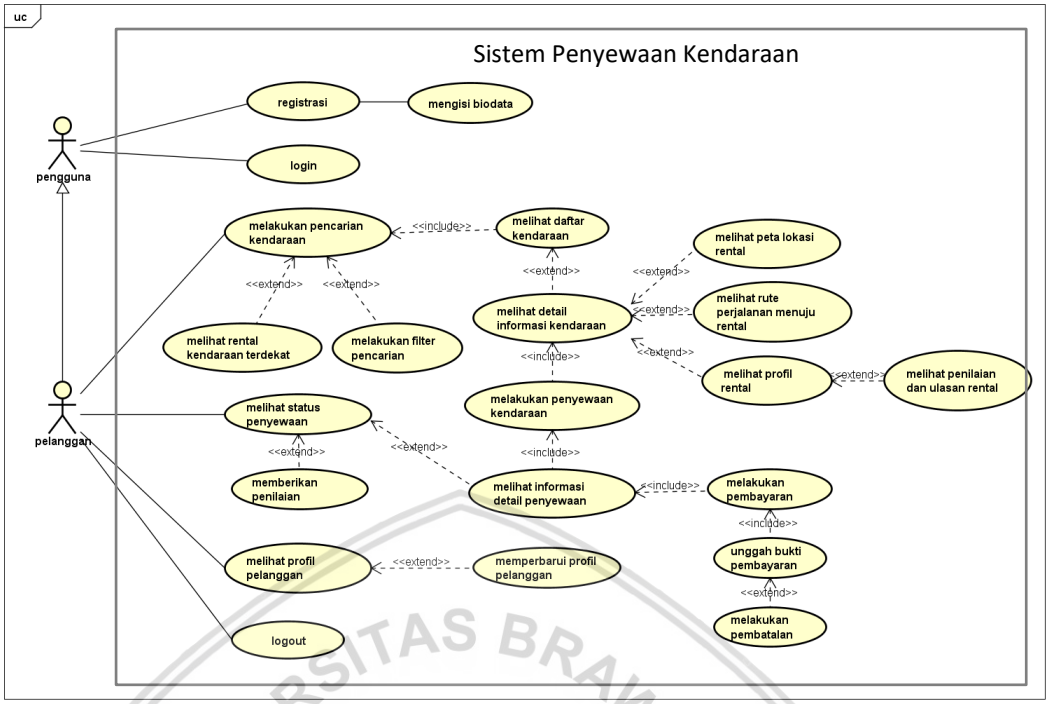
4.1.7.1 Diagram *Use Case* Pelanggan

Pada diagram *use case* terdapat dua buah diagram *use case* sesuai dengan banyaknya iterasi pada penelitian ini. Pelanggan merupakan sebagai aktor utama dalam sistem yang dapat melakukan kebutuhan fungsional yang disediakan oleh sistem. Diagram *use case* pelanggan ditunjukkan oleh Gambar 4.4.



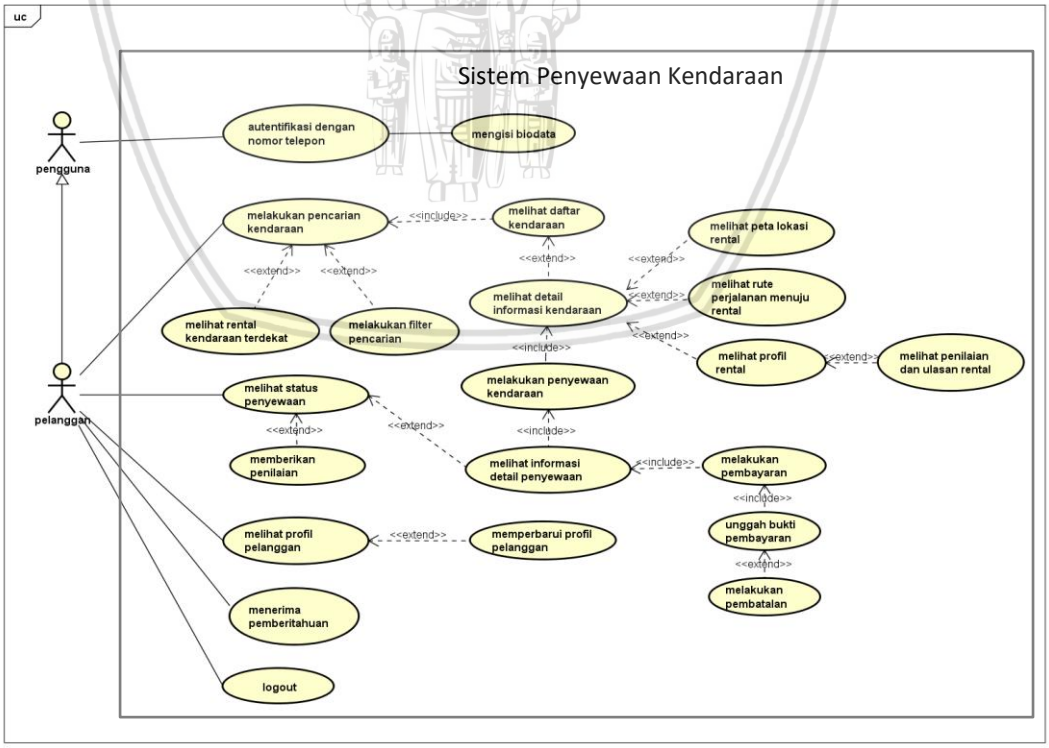
Gambar 4.4 Diagram *Use Case* Pelanggan

Sedangkan pada Gambar 4.5 menggambarkan Diagram *use case* dari hasil iterasi pertama. Pada diagram tersebut terdapat penambahan *use case* melihat penilaian dan ulasan rental.



Gambar 4.5 Diagram *Use Case* Pelanggan Iterasi 1

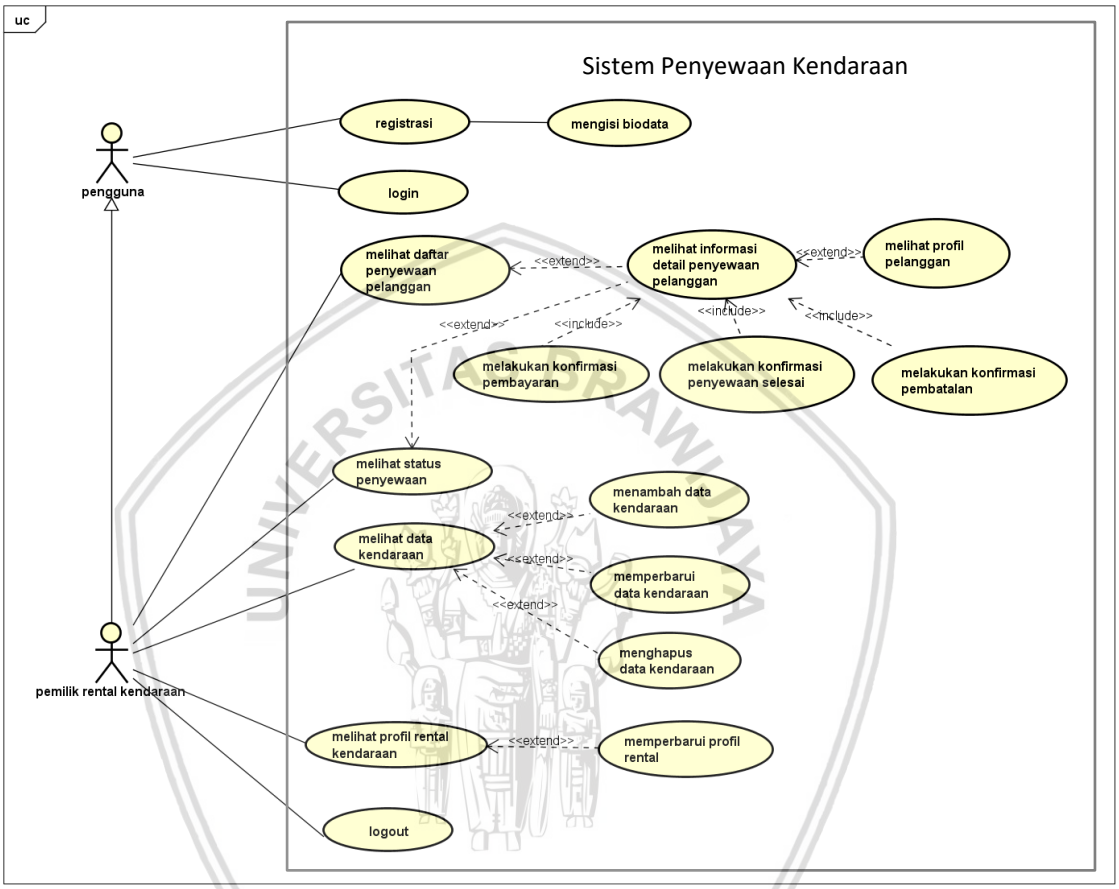
Pada iterasi kedua terdapat penambahan satu buah use case baru, yaitu use case untuk menampilkan pemberitahuan seperti yang di tunjukkan oleh Gambar 4.6.



Gambar 4.6 Diagram *Use Case* Pelanggan Iterasi 2

4.1.7.2 Diagram *Use Case* Pemilik rental kendaraan

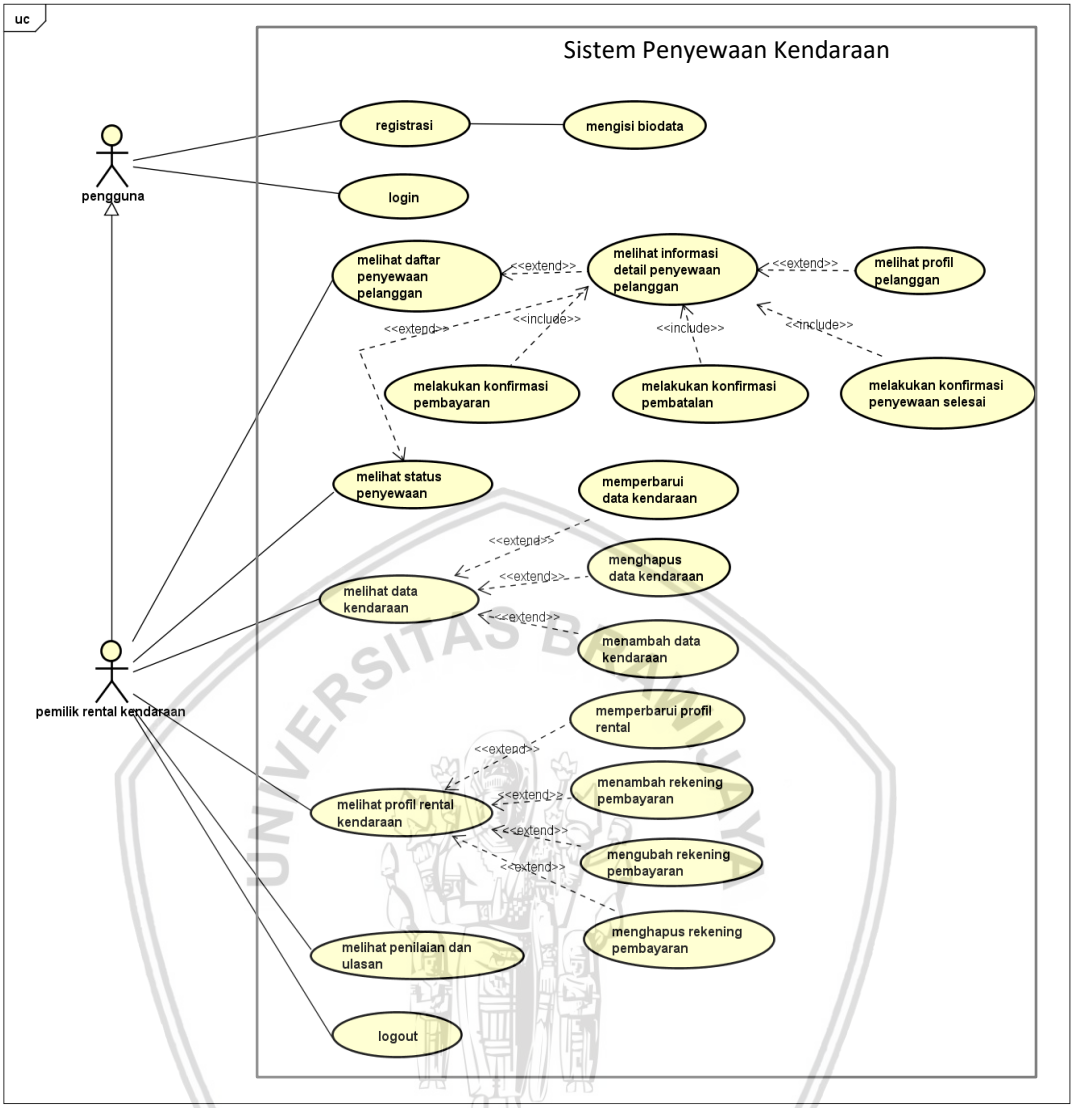
Pada diagram *use case* pemilik rental kendaraan terdapat dua buah diagram *use case* sesuai dengan banyaknya iterasi pada penelitian ini. Pemilik rental merupakan aktor utama dalam sistem yang dapat melakukan kebutuhan fungsional yang disediakan oleh sistem. Diagram *use case* pemilik rental kendaraan ditunjukkan oleh Gambar 4.7.



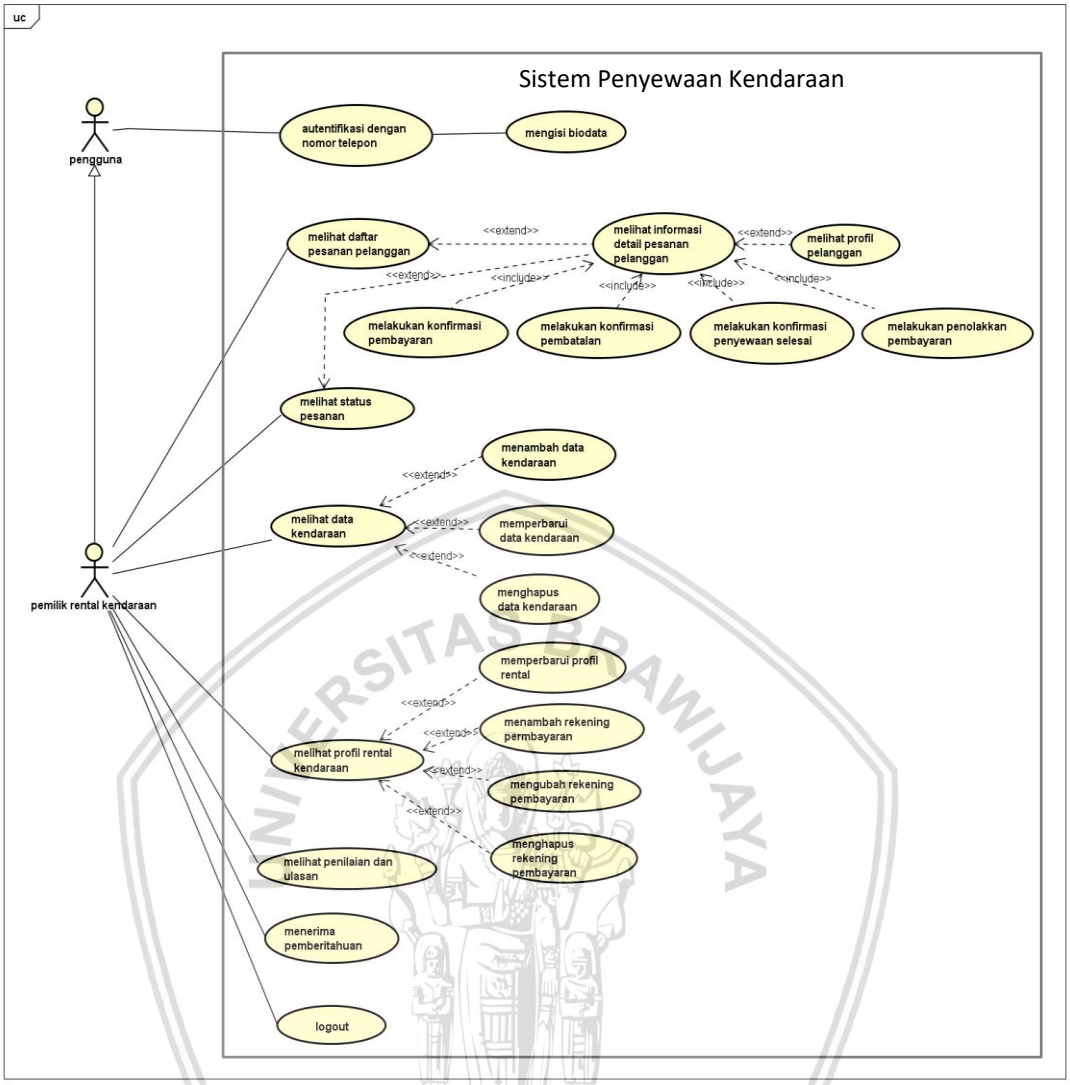
Gambar 4.7 Diagram *Use Case* Pemilik Rental Kendaraan

Sedangkan Gambar 4.8 menjelaskan tentang Diagram *use case* sisi pemilik rental pada iterasi pertama. Terdapat penambahan *use case* didalamnya. *Use case* yang ditambahkan yaitu melihat penilaian dan ulasan, menambah rekening pembayaran, mengubah rekening pembayaran dan menghapus rekening pembayaran. Diagram *use case* pemilik rental iterasi pertama ditunjukkan oleh Gambar 4.8.

Pada iterasi kedua sisi pemilik rental terdapat penambahan satu buah use case baru, yaitu use case untuk menampilkan pemberitahuan seperti yang di tunjukkan oleh Gambar 4.9.



Gambar 4.8 Diagram *Use Case* Pemilik Rental Kendaraan Iterasi 1



Gambar 4.9 Diagram *Use Case* Pemilik Rental Kendaraan Iterasi 2

4.1.8 Skenario *Use Case*

Skenario *use case* merupakan urutan alur dan penjelasan lebih lengkap dari diagram *use case* yang telah didefinisikan. Pada bagian skenario *Use Case* juga akan dijelaskan beberapa urutan alur dan penjelasan dari setiap kebutuhan yan telah di spesifikasikan sebelumnya. Pada bagian ini juga terdapat tiga bagian penulisan untuk masing-masing pengguna, pelanggan dan pemilik rental. Bagian pertama akan menjelaskan skenario *use case* dari spesifikasi kebutuhan yang didapatkan dari hasil elisitasi. Sedangkan bagian kedua akan menjelaskan skenario *use case* yang didapatkan dari hasil iterasi pertama. Lalu bagian ketiga akan menjelaskan skenario *use case* yang didapatkan dari hasil iterasi kedua.

4.1.8.1 Skenario Use Case Pengguna

Tabel 4.13 Skenario *Use Case* Registrasi

Skenario Kasus pada Sistem	
Nomor *Use Case*	SRS_001_01
Nama	Registrasi
Tujuan	Untuk melakukan pendaftaran menjadi pelanggan pada sistem.
Deskripsi	*Use* case ini menjelaskan bagaimana pengguna melakukan registrasi untuk menjadi pelanggan pada sistem.
Aktor	Pengguna
Skenario Utama	
Kondisi Awal	Sistem menampilkan form login.
Aksi Aktor	Reaksi Sistem
1. Pengguna menekan tombol registrasi pada halaman login.	2. Sistem menampilkan halaman registrasi.
3. Pengguna memasukkan email dan password lalu menekan tombol daftar.	4. Sistem akan melakukan pengecekkan terhadap data yang telah diisikan oleh pengguna. Apabila data belum ada di dalam sistem, maka data akan disimpan dalam sistem.
Skenario Alternatif 1 : Jika data sudah terdaftar	
	5. Sistem akan menampilkan pesan peringatan bahwa data yang dimasukkan sudah terdapat dalam sistem.
Skenario Alternatif 2 : Jika data yang dimasukkan belum lengkap	
	6. Sistem akan menampilkan pesan peringatan bahwa data yang dimasukkan dalam halaman isian registrasi belum lengkap.

Kondisi Akhir	Sistem kan menampilkan pesan bahwa registrasi berhasil dilakukan dan menampilkan halaman isi biodata pengguna.

Tabel 4.14 Skenario *Use Case* Login

Skenario Kasus pada Sistem	
Nomor *Use Case*	SRS_001_02
Nama	Login
Tujuan	Agar pengguna bisa masuk kedalam sistem.
Deskripsi	*Use case* ini menjelaskan tentang bagaimana pengguna melakukan login untuk masuk kedalam sistem sebagai pelanggan.
Aktor	Pengguna
Skenario Utama	
Kondisi Awal	Sistem menampilkan *form* login.
Aksi Aktor	Reaksi Sistem
1. Pengguna memasukkan data login (email dan password) lalu menekan tombol Login.	2. Sistem akan melakukan pengecekkan terhadap data yang telah dimasukan oleh pelanggan.
Skenario Alternatif 1 : jika email dan password kosong	
	3. Sistem akan menampilkan pesan peringatan bahwa email atau password belum dimasukkan.
Skenario Alternatif 2 : jika email dan password tidak terdaftar	
	4. Sistem akan menampilkan pesan peringatan bahwa emal tidak terdaftar dalam sistem.
Kondisi Akhir	Sistem menampilkan halaman utama sistem yaitu halaman pencarian kendaraan.

Tabel 4.15 Skenario *Use Case* Mengisi Biodata Pengguna

Skenario Kasus pada Sistem	
Nomor *Use Case*	SRS_001_02
Nama	Mengisi biodata pengguna
Tujuan	Agar pengguna dapat menyimpan biodata.
Deskripsi	*Use case* ini menjelaskan tentang bagaimana pengguna mengisi biodata dan menyimpannya dalam sistem.
Aktor	Pengguna
Skenario Utama	
Kondisi Awal	Sistem menampilkan *form* isian biodata dan pengguna sudah terdaftar atau berhasil registrasi.
Aksi Aktor	Reaksi Sistem
1. Pengguna memasukkan data lengkap pengguna kedalam *form* isian.	2. Sistem akan melakukan pengecekkan terhadap data yang telah dimasukan oleh pengguna.
Skenario Alternatif 1 : jika terdapat data yang masih kosong	
	3. Sistem akan menampilkan pesan peringatan untuk memenuhi semua *form* isian.
Kondisi Akhir	Sistem menyimpan biodata pengguna dan menampilkan halaman utama sistem yaitu halaman pencarian kendaraan.

4.1.8.2 Skenario Use Case Pengguna Iterasi 2

Tabel 4.16 Skenario *Use Case* Autentifikasi dengan Nomor Telepon

Skenario Kasus pada Sistem	
Nomor *Use Case*	SRS_001_04
Nama	Autentifikasi dengan Nomor Telepon
Tujuan	Untuk melakukan pendaftaran dan login sebagai pelanggan pada sistem.

Deskripsi	*Use* case ini menjelaskan bagaimana pengguna melakukan registrasi dan login untuk menjadi pelanggan pada sistem.
Aktor	Pengguna
Skenario Utama	
Kondisi Awal	Sistem menampilkan form untuk memasukkan nomor telepon.
Aksi Aktor	Reaksi Sistem
1. Pengguna mengisi nomor telepeon ponsel yang aktif.	
2. Pengguna menekan tombol selanjutnya.	3. Sistem mengirimkan SMS berupa kode verifikasi.
4. Pengguna memasukkan kode verifikasi.	
5. Pengguna memasukkan kode verifikasi yang dikirimkan lewat SMS.	6. Sistem akan melakukan pengecekkan terhadap kode verifikasi yang telah diisikan oleh pengguna. Apabila kode verifikasi benar dan data pengguna belum ada di dalam sistem, maka data pengguna akan disimpan dalam sistem dan mengarahkan ke halaman input biodata pengguna.
Skenario Alternatif 1 : Jika kode verifikasi yang dimasukkan salah	
	7. Sistem menampilkan pesan peringatan
Skenario Alternatif 2 : Jika data sudah terdaftar	
	8. Sistem akan mengerahkan ke halaman utama sistem.
Kondisi Akhir	Sistem akan mengarahkan ke halaman input biodata pengguna.

4.1.8.3 Skenario Use Case Pelanggan

Tabel 4.17 Skenario *Use Case* Melakukan pencarian kendaraan

Skenario Kasus pada Sistem	
Nomor *Use Case*	SRS_002_01
Nama	Melakukan pencarian kendaraan
Tujuan	Agar pelanggan bisa melakukan pencarian kendaraan yang tersedia dan sesuai dengan tanggal penyewaan yang diinginkan.
Deskripsi	*Use case* ini menjelaskan tentang bagaimana proses pencarian kendaraan yang disewakan dan tersedia di Kota Malang.
Aktor	Pelanggan
Skenario Utama	
Kondisi Awal	Pelanggan telah berhasil login
Aksi Aktor	Reaksi Sistem
1. Pelanggan memilih tanggal penyewaan dan kategori kendaraan pada halaman awal pencarian kendaraan.	2. Sistem menampilkan daftar kendaraan yang tersedia sesuai dengan tanggal penyewaan dan kategori kendaraan.
Skenario Alternatif 1 : data kendaraan tidak tersedia	
	3. Sistem akan menampilkan pesan bahwa data tidak tersedia.
Kondisi Akhir	Sistem menampilkan daftar kendaraan yang sesuai dengan pencarian pelanggan.

Tabel 4.18 Skenario *Use Case* Melihat Daftar Kendaraan

Skenario Kasus pada Sistem	
Nomor *Use Case*	SRS_002_02
Nama	Melihat daftar kendaraan

Tujuan	Agar pengguna bisa melihat daftar kendaraan yang tersedia sesuai dengan pencarian yang dilakukan pelanggan.
Deskripsi	*Use case* ini menjelaskan tentang bagaimana proses menampilkan daftar kendaraan yang tersedia dan sesuai dengan pencarian pelanggan.
Aktor	Pelanggan
Skenario Utama	
Kondisi Awal	Pelanggan berhasil melakukan pencarian kendaraan.
Aksi Aktor	Reaksi Sistem
1. Pelanggan menekan tombol Cari pada halaman pencarian.	2. Sistem menampilkan daftar kendaraan yang tersedia sesuai dengan pencarian pelanggan.
Kondisi Akhir	Sistem menampilkan daftar kendaraan yang tersedia.

Tabel 4.19 Skenario *Use Case* Melihat detail informasi kendaraan

Skenario Kasus pada Sistem	
Nomor *Use Case*	SRS_002_03
Nama	Melihat detail informasi kendaraan
Tujuan	Agar pengguna bisa melihat informasi kendaraan secara detail
Deskripsi	*Use case* ini menjelaskan tentang bagaimana proses menampilkan informasi penyewaan kendaraan di Kota Malang.
Aktor	Pelanggan
Skenario Utama	
Kondisi Awal	Pelanggan telah berhasil melihat hasil pencarian daftar kendaraan yang tersedia.
Aksi Aktor	Reaksi Sistem

1. Pelanggan memilih kendaraan yang diinginkan dari daftar kendaraan hasil pencarian.	2. Sistem menampilkan halaman detail informasi kendaraan.
Kondisi Akhir	Sistem menampilkan detail informasi kendaraan

Tabel 4.20 Skenario *Use Case* Melakukan penyewaan kendaraan

Skenario Kasus pada Sistem	
Nomor *Use Case*	SRS_002_04
Nama	Melakukan penyewaan kendaraan
Tujuan	Agar pelanggan bisa melakukan penyewaan pada kendaraan yang ingin disewa.
Deskripsi	*Use case* ini menjelaskan tentang proses penyewaan kendaraan yang ingin disewa pada sistem.
Aktor	Pelanggan
Skenario Utama	
Kondisi Awal	Pelanggan telah berhasil melihat detail informasi kendaraan yang ingin disewa.
Aksi Aktor	Reaksi Sistem
1. Pelanggan menekan tombol sewa kendaraan pada halaman detail informasi kendaraan yang telah dipilih.	2. Sistem akan menampilkan rincian penyewaan kendaraan.
3. Pelanggan menekan tombol lanjutkan untuk melanjutkan proses penyewaan kendaraan.	4. Sistem akan menampilkan informasi penyewaan kendaraan yang terdiri dari biodata penyewa dan sistem menampilkan *form* isian mengenai waktu dan lokasi penjemputan atau waktu pengambilan kendaraan.

5. Pelanggan memasukkan waktu dan lokasi penjemputan atau pengambilan.	
6. Pelanggan menekan tombol buat penyewaan untuk melanjutkan proses buat penyewaan.	7. Sistem akan melakukan pengecekkan jika kendaraan masih tersedia atau tidak. Jika tersedia sistem akan mengarahkan ke halaman pembayaran.
Skenario alternatif 1 : data kendaraan yang dipilih tidak tersedia	
	8. Sistem akan menampilkan pesan peringatan bahwa kendaraan yang dipilih sudah tidak tersedia.
	9. Sistem akan menampilkan halaman pencarian.
Skenario alternatif 2 : jika informasi penyewaan belum lengkap	
	10. Sistem akan menampilkan peringatan untuk melengkapi kolom isian.
Kondisi Akhir	Sistem menyimpan data penyewaan pada basis data.

Tabel 4.21 Skenario *Use Case* Melihat informasi detail penyewaan

Skenario Kasus pada Sistem	
Nomor *Use Case*	SRS_002_05
Nama	Melihat informasi detail penyewaan
Tujuan	Agar pelanggan bisa melihat informasi detail penyewaan kendaraan yang telah dilakukan.
Deskripsi	*Use case* ini menjelaskan tentang bagaimana proses melihat informasi detail penyewaan pada sistem.
Aktor	Pelanggan
Skenario Utama	

Kondisi Awal	Pelanggan telah berhasil melakukan proses penyewaan kendaraan yang ingin disewa.
Aksi Aktor	Reaksi Sistem
1. Menekan tombol menu pada sistem.	
2. Menekan menu kelola penyewaan.	3. Sistem akan menampilkan daftar penyewaan yang telah dilakukan oleh pelanggan.
4. Pelanggan memilih *tab menu* status penyewaan yang ingin dilihat dan memilih penyewaan dari daftar penyewaan ingin yang ditampilkan.	5. Sistem akan menampilkan informasi detail penyewaan kendaraan yang telah dilakukan oleh pelanggan.
Kondisi Akhir	Sistem menampilkan halaman detail informasi penyewaan kendaraan.

Tabel 4.22 Skenario *Use Case* Melakukan pembayaran

Skenario Kasus pada Sistem	
Nomor *Use Case*	SRS_002_06
Nama	Melakukan pembayaran
Tujuan	Agar pelanggan bisa melakukan pembayaran atas penyewaan kendaraan yang telah dilakukan pelanggan.
Deskripsi	*Use case* ini menjelaskan tentang bagaimana proses pembayaran yang dilakukan oleh pelanggan.
Aktor	Pelanggan
Skenario Utama	
Kondisi Awal	Pelanggan telah berhasil melakukan penyewaan.

Aksi Aktor	Reaksi Sistem
	1. Sistem menampilkan daftar rekening pembayaran yang tersedia dalam sistem.
2. Pelanggan memilih rekening pembayaran.	3. Sistem akan menampilkan rincian penyewaan, informasi mengenai petunjuk pembayaran dan batas maksimal melakukan pembayaran.
4. Pelanggan melakukan pembayaran melalui transfer rekening ke rekening yang dituju sesuai dengan nominal total pembayaran.	
Kondisi Akhir	Sistem menampilkan halaman mengenai informasi pembayaran.

Tabel 4.23 Skenario *Use Case* Unggah bukti pembayaran

Skenario Kasus pada Sistem	
Nomor *Use Case*	SRS_002_07
Nama	Unggah bukti pembayaran
Tujuan	Agar pelanggan bisa mengunggah bukti pembayaran atas penyewaan kendaraan yang telah dilakukan.
Deskripsi	*Use case* ini menjelaskan tentang proses unggah bukti pembayaran.
Aktor	Pelanggan
Skenario Utama	
Kondisi Awal	Pelanggan telah melakukan pembayaran melalui metode pembayaran yang telah dipilih.
Aksi Aktor	Reaksi Sistem

1. Pelanggan memlih unggah bukti pembayaran sekarang pada halaman informasi pembayaran.	2. Sistem akan menampilkan halaman unggah bukti pembayaran.
3. Pelanggan memasukkan data bukti pembayaran.	
4. Pelanggan menekan tombol unggah.	5. Sistem akan menyimpan data tersebut pada basis data firebase.
Skenario alternatif 2 : jika pelanggan memilih unggah bukti pembayaran nanti	
	6. Sistem akan mengarahkan ke halaman kelola penyewaan dengan status belum bayar.
	7. Sistem akan menampilkan penyewaan dengan status belum bayar.
8. Pelanggan menekan daftar penyewaan yang diinginkan.	9. Sistem akan menampilkan halaman detail dan menampilkan tombol unggah bukti pembayaran.
10. Pelanggan menekan tombol unggah bukti pembayaran.	11. Sistem akan menampilkan halaman unggah bukti pembayaran.
12. Pelanggan memasukkan informasi mengenai pembayaran yang telah dilakukan dan menekan tombol unggah bukti pembayaran.	13. Sistem akan menyimpan data tersebut pada basis data firebase.
Skenario Alternatif 2 : jika data bukti pembayaran tidak lengkap	

	14. Sistem akan menampilkan pesan peringatan bahwa data yang diisikan belum lengkap.
Kondisi Akhir	Sistem akan menyimpan data bukti pembayaran, mengubah status penyewaan menjadi menunggu konfirmasi pembayaran dan menampilkan daftar penyewaan dengan status menunggu konfirmasi pembayaran.

Tabel 4.24 Skenario *Use Case* Melakukan pembatalan

Skenario Kasus pada Sistem	
Nomor *Use Case*	SRS_002_08
Nama	Melakukan pembatalan
Tujuan	Agar pelanggan bisa melakukan pembatalan atas penyewaan yang sudah dilakukan.
Deskripsi	Use case ini menjelaskan tentang proses pembatalan penyewaan.
Aktor	Pelanggan
Skenario Utama	
Kondisi Awal	Pelanggan telah melakukan pembayaran atas penyewaan kendaraan dan telah di konfirmasi oleh pemilik rental sehingga status penyewaan berhasil.
Aksi Aktor	Reaksi Sistem
1. Pelanggan menekan tombol menu.	
2. Pelanggan menekan menu kelola penyewaan.	
3. Pelanggan memilih *tab menu* status berhasil lalu memilih	4. Sistem akan menampilkan daftar penyewaan yang telah berhasil dilakukan oleh pelanggan.

penyewaan dari daftar penyewaan yang diinginkan.	
5. Pelanggan menekan daftar penyewaan yang diinginkan.	6. Sistem akan menampilkan detail penyewaan dan tombol ajukan pembatalan.
7. Pelanggan menekan tombol ajukan pembatalan.	8. Sistem akan memproses permintaan pembatalan penyewaan dan mengirimkan permintaan pembatalan pelanggan kepada pemilik rental.
Kondisi Akhir	Sistem akan mengubah status penyewaan menjadi pengajuan pembatalan dan menampilkan daftar penyewaan dengan status pengajuan pembatalan.

Tabel 4.25 Skenario *Use Case* Melihat status penyewaan

Skenario Kasus pada Sistem	
Nomor *Use Case*	SRS_002_09
Nama	Melihat status penyewaan
Tujuan	Agar pelanggan bisa melihat status penyewaan kendaraan yang telah dilakukan.
Deskripsi	Use case ini menjelaskan tentang proses melihat status penyewaan.
Aktor	Pelanggan
Skenario Utama	
Kondisi Awal	Pelanggan telah melakukan penyewaan kendaraan.
Aksi Aktor	Reaksi Sistem
1. Pelanggan menekan tombol menu.	

2. Pelanggan menekan menu kelola penyewaan.	3. Sistem akan menampilkan daftar penyewaan yang telah dilakukan oleh pelanggan.
4. Pelanggan memilih status yang diinginkan pada *tab menu* status.	5. Sistem akan menampilkan status penyewaan pengguna.
Kondisi Akhir	Sistem akan menampilkan status penyewaan pelanggan.

Tabel 4.26 Skenario *Use Case* Melihat rental kendaraan terdekat

Skenario Kasus pada Sistem	
Nomor *Use Case*	SRS_002_10
Nama	Melihat rental kendaraan terdekat
Tujuan	Agar pengguna dapat mendapat informasi rental kendaraan terdekat dari lokasinya saat ini.
Deskripsi	*Use case* ini menjelaskan tentang bagaimana proses menampilkan informasi rental kendaraan terdekat dari keberadaan pelanggan.
Aktor	Pelanggan
Skenario Utama	
Kondisi Awal	Pelanggan telah berhasil melakukan pencarian
Aksi Aktor	Reaksi Sistem
1. Pelanggan menekan tombol terdekat.	2. Sistem mengambil data koordinat pelanggan.
	3. Sistem menghitung rental kendaraan terdekat dari pelanggan berdasarkan koordinat, mengurutkannya lalu menampilkannya.
Kondisi Akhir	Sistem menampilkan daftar kendaraan sesuai dengan rental terdekat.

Tabel 4.27 Skenario *Use Case* Melakukan *filter* pencarian

Skenario Kasus pada Sistem	
Nomor *Use Case*	SRS_002_11
Nama	Melakukan *filter* pencarian
Tujuan	Agar pengguna bisa melakukan penyaringan atau *filter* pada daftar kendaraan yang telah muncul pada halaman hasil pencarian.
Deskripsi	*Use case* ini menjelaskan tentang bagaimana proses melakukan *filter* pencarian pada daftar kendaraan.
Aktor	Pelanggan
Skenario Utama	
Kondisi Awal	Pelanggan telah berhasil melakukan pencarian kendaraan dan data kendaraan tersedia.
Aksi Aktor	Reaksi Sistem
1. Pelanggan menekan tombol *filter* pada halaman daftar hasil pencarian kendaraan.	2. Sistem menampilkan halaman pemilihan *filter* yang diinginkan.
3. Pelanggan memilih rentang harga yang diinginkan.	
4. Pelanggan menekan tombol terapkan *filter*.	5. Sistem akan menampilkan daftar pencarian hasil *filter.*
Kondisi Akhir	Sistem menampilkan daftar pencarian hasil *filter.*

Tabel 4.28 Skenario *Use Case* Melihat peta lokasi rental

Skenario Kasus pada Sistem	
Nomor *Use Case*	SRS_002_12
Nama	Melihat peta lokasi rental

Tujuan	Agar pengguna bisa melihat peta lokasi rental kendaraan.
Deskripsi	*Use case* ini menjelaskan bagaimana proses menampilkan peta lokasi rental kendaraan yang dipilih pelanggan.
Aktor	Pelanggan
Skenario Utama	
Kondisi Awal	Pelanggan telah berhasil melihat informasi detail kendaraan.
Aksi Aktor	Reaksi Sistem
1. Pelanggan memilih kendaraan yang diinginkan dari daftar kendaraan.	2. Sistem menampilkan informasi kendaraan.
3. Pelanggan menekan tombol informasi detail.	4. Sistem menampilkan lokasi rental kendaraan pada halaman informasi detail kendaraan.
Kondisi Akhir	Sistem menampilkan peta lokasi rental.

Tabel 4.29 Skenario *Use Case* Melihat rute perjalanan menuju rental

Skenario Kasus pada Sistem	
Nomor *Use Case*	SRS_002_13
Nama	Melihat rute perjalanan menuju rental
Tujuan	Agar pengguna mendapat rute perjalanan menuju rental.
Deskripsi	*Use case* ini menjelaskan bagaimana proses menampilkan rute ke rental kendaraan.
Aktor	Pelanggan
Skenario Utama	
Kondisi Awal	Pelanggan telah berhasil melihat informasi detail kendaraan.
Aksi Aktor	Reaksi Sistem

1. Pelanggan menekan tombol panah pada peta lokasi rental.	2. Sistem akan mengarahkan ke halaman *google maps*.
Kondisi Akhir	Sistem mengarahkan ke halaman *google maps* dan google maps menampilkan rute perjalanan menuju rental.

Tabel 4.30 Skenario *Use Case* Melihat profil rental

Skenario Kasus pada Sistem	
Nomor *Use Case*	SRS_002_14
Nama	Melihat profil rental
Tujuan	Agar dapat melihat profil rental kendaraan.
Deskripsi	*Use case* ini menjelaskan tentang bagaimana proses melihat profil rental kendaraan.
Aktor	Pelanggan.
Skenario Utama	
Kondisi Awal	Pelanggan telah berhasil melihat informasi detail kendaraan.
Aksi Aktor	Reaksi Sistem
1. Pelanggan menekan tombol lihat profil pada halaman informasi detail kendaraan.	2. Sistem akan menampilkan halaman profil rental kendaraan.
Kondisi Akhir	Sistem menampilkan profil rental kendaraan.

Tabel 4.31 Skenario *Use Case* Memberikan penilaian

Skenario Kasus pada Sistem	
Nomor *Use Case*	SRS_002_15
Nama	Memberikan penilaian

Tujuan	Agar pelanggan dapat memberikan penilaian terhadap penyewaan kendaraan yang telah dilakukan.
Deskripsi	*Use case* ini menjelaskan tentang bagaimana proses memberikan penilaian yang dilakukan oleh pelanggan terhadap rental kendaraan.
Aktor	Pelanggan
Skenario Utama	
Kondisi Awal	Pelanggan telah selesai melakukan penyewaan kendaraan
Aksi Aktor	Reaksi Sistem
1. Pelanggan menekan tombol menu.	
2. Pelanggan menekan menu kelola penyewaan.	
3. Pelanggan memilih daftar penyewaan dengan status selesai pada *tab menu* status.	4. Sistem akan menampilkan daftar penyewaan yang telah dilakukan oleh pelanggan dengan status selesai.
5. Pelanggan menekan daftar penyewaan yang diinginkan.	6. Sistem akan menampilkan detail penyewaan dan menampilkan tombol berikan penilaian.
7. Pelanggan menekan tombol berikan penilaian.	
8. Pelanggan memberikan penilaian dan ulasan yang sesuai dengan pengalaman penyewaannya.	
9. Pelanggan menekan tombol simpan.	10. Sistem akan mengecek apakah form isian penilaian sudah lengkap atau belum. Jika sudah lengkap maka sistem akan

	menampilkan pesan pemberitahuan bahwa penilaian telah tersimpan.
Skenario alternatif 1 : form isian penilaian belum lengkap	
	11. Sistem akan menampilkan pesan bahwa form isian penilaian belum lengkap.
Kondisi Akhir	Sistem menyimpan penilaian yang diberikan oleh pelanggan dan menampilkannya.

Tabel 4.32 Skenario *Use Case* Melihat profil pelanggan

Skenario Kasus pada Sistem	
Nomor *Use Case*	SRS_002_16
Nama	Melihat profil pelanggan
Tujuan	Agar dapat melihat profil pelanggan.
Deskripsi	*Use case* ini menjelaskan tentang bagaimana proses melihat profil pelanggan.
Aktor	Pelanggan.
Skenario Utama	
Kondisi Awal	Pelanggan telah berhasil login.
Aksi Aktor	Reaksi Sistem
1. Pelanggan menekan tombol menu.	
2. Pelanggan menekan menu profil.	3. Sistem menampilkan profil pelanggan.
Kondisi Akhir	Sistem menampilkan profil pelanggan.

Tabel 4.33 Skenario *Use Case* Memperbarui profil

Skenario Kasus pada Sistem	
Nomor *Use Case*	SRS_002_17
Nama	Memperbarui profil

Tujuan	Agar dapat memperbarui profil pelanggan.
Deskripsi	*Use case* ini menjelaskan tentang bagaimana proses memperbarui profil pelanggan.
Aktor	Pelanggan.
Skenario Utama	
Kondisi Awal	Pelanggan telah berhasil login.
Aksi Aktor	Reaksi Sistem
1. Pelanggan menekan tombol menu.	
2. Pelanggan menekan menu profil.	3. Sistem menampilkan profil pelanggan dan menampilkan tombol ubah profil.
4. Pelanggan menekan tombol edit profil.	5. Sistem menampilkan form isian ubah biodata.
6. Pelanggan mengisi form isian ubah profil.	
7. Pelanggan menekan tombol simpan	8. Sistem akan mengecek apakah form isian sudah lengkap atau belum, jika sudah lengkap maka sistem akan menyimpannya.
Skenario alternatif 1 : jika form isian ubah profil belum lengkap	
	9. Sistem akan menampilkan pesan bahwa kolom isian ubah profil belum lengkap.
Kondisi Akhir	Sistem menampilkan profil pelanggan.

Tabel 4.34 Skenario *Use Case* Logout

Skenario Kasus pada Sistem	
Nomor *Use Case*	SRS_002_18
Nama	Logout
Tujuan	Agar pelanggan dapat keluar dari sistem.

Deskripsi	*Use case* ini menjelaskan tentang bagaimana proses logout yang dilakukan pelanggan.
Aktor	Pelanggan
Skenario Utama	
Kondisi Awal	Pelanggan telah berhasil login.
Aksi Aktor	Reaksi Sistem
1. Pelanggan menekan tombol menu.	
2. Pelanggan menekan menu logout.	3. Sistem menampilkan halaman login.
Kondisi Akhir	Sistem menampilkan halaman login.

4.1.8.4 Skenario Use Case Pelanggan Iterasi 1

Tabel 4.35 Skenario *Use Case* Melakukan Pencarian Kendaraan

Skenario Kasus pada Sistem	
Nomor *Use Case*	SRS_002_01
Nama	Melakukan pencarian kendaraan
Tujuan	Agar pelanggan bisa melakukan pencarian kendaraan yang tersedia dan sesuai dengan tanggal sewa, tanggal kembali penyewaan, kategori kendaraan dan jumlah kendaraan.
Deskripsi	*Use case* ini menjelaskan tentang bagaimana proses pencarian kendaraan yang disewakan dan tersedia di Kota Malang.
Aktor	Pelanggan
Skenario Utama	
Kondisi Awal	Pelanggan telah berhasil login
Aksi Aktor	Reaksi Sistem

1. Pelanggan memilih tanggal penyewaan, tanggal kembali, kategori kendaraan dan jumlah kendaraan pada halaman awal pencarian kendaraan.	2. Sistem menampilkan daftar kendaraan yang tersedia sesuai dengan tanggal penyewaan, tanggal kembali, jumlah kendaraan dan kategori kendaraan.
Skenario Alternatif 1 : data kendaraan tidak tersedia	
	3. Sistem akan menampilkan pesan bahwa kendaraan tidak tersedia.
Kondisi Akhir	Sistem menampilkan daftar kendaraan yang sesuai dengan pencarian pelanggan.

Tabel 4.36 Skenario *Use Case* Melihat Rental Kendaraan Terdekat

Skenario Kasus pada Sistem	
Nomor *Use Case*	SRS_002_10
Nama	Melihat rental kendaraan terdekat
Tujuan	Agar pengguna dapat mendapat informasi rental kendaraan terdekat dari lokasinya saat ini.
Deskripsi	*Use case* ini menjelaskan tentang bagaimana proses menampilkan informasi rental kendaraan terdekat berdasarkan radius tertentu dari keberadaan pelanggan.
Aktor	Pelanggan
Skenario Utama	
Kondisi Awal	Pelanggan telah berhasil melakukan pencarian
Aksi Aktor	Reaksi Sistem
1. Pelanggan menekan tombol terdekat.	2. Sistem menampilkan pilihan untuk memilih jumlah radius yang diinginkan.

3. Pelanggan memilih radius yang diinginkan.	
4. Pelanggan menekan tombol terapkan.	5. Sistem menampilkan titik lokasi rental kendaraan berdasarkan radius yang telah diterapkan pelanggan.
Kondisi Akhir	Sistem menampilkan titik lokasi rental kendaraan.

Tabel 4.37 Skenario *Use Case* Melakukan Filter Pencarian

Skenario Kasus pada Sistem	
Nomor *Use Case*	SRS_002_11
Nama	Melakukan filter pencarian
Tujuan	Agar pengguna bisa melakukan penyaringan atau *filter* pada daftar kendaraan yang telah muncul pada halaman hasil pencarian.
Deskripsi	*Use case* ini menjelaskan tentang bagaimana proses melakukan filter pencarian pada daftar kendaraan.
Aktor	Pelanggan
Skenario Utama	
Kondisi Awal	Pelanggan telah berhasil melakukan pencarian kendaraan dan data kendaraan tersedia
Aksi Aktor	Reaksi Sistem
1. Pelanggan menekan tombol *filter* pada halaman daftar hasil pencarian kendaraan.	2. Sistem menampilkan halaman pemilihan filter yang diinginkan.
3. Pelanggan memilih rentang harga yang diinginkan atau memilih dengan sopir atau tanpa sopir, dengan	

bahan bakar atau tanpa bahan bakar.	
4. Pelanggan menekan tombol terapkan.	5. Sistem akan menampilkan daftar pencarian yang telah dilakukan penyaringan/*filter* data.
Kondisi Akhir	Sistem menampilkan daftar kendaraan yang telah dilakukan *filter*.

Tabel 4.38 Skenario *Use Case* Melihat Penilaian dan Ulasan Rental

Skenario Kasus pada Sistem	
Nomor *Use Case*	SRS_002_19
Nama	Melihat Penilaian dan Ulasan Rental
Tujuan	Agar pelanggan bisa melakukan melihat penilaian dan ulasan terhadap rental kendaraan.
Deskripsi	*Use case* ini menjelaskan tentang bagaimana proses melihat penilaian dan ulasan terhadap rental kendaraan.
Aktor	Pelanggan
Skenario Utama	
Kondisi Awal	Pelanggan telah berhasil melihat detail kendaraan.
Aksi Aktor	Reaksi Sistem
1. Pelanggan menekan tombol lihat profil rental pada halaman detail kendaraan.	2. Sistem akan menampilkan halaman profil rental.
	3. Sistem akan menampilkan penilaian pada halaman profil rental dan menampilkan tombol lihat ulasan.
4. Pelanggan menakan tombol lihat ulasan.	5. Sistem akan menampilkan daftar ulasan.

Kondisi Akhir	Sistem menampilkan penilaian dan ulasan.

4.1.8.5 Skenario Use Case Pelanggan Iterasi 2

Tabel 4.39 Skenario *Use Case* Menerima Pemberitahuan

Skenario Kasus pada Sistem	
Nomor *Use Case*	SRS_002_20
Nama	Menerima Pemberitahuan
Tujuan	Agar pelanggan bisa melakukan menerima pemberitahuan pada kondisi tertentu.
Deskripsi	*Use case* ini menjelaskan tentang bagaimana proses menerima pemberitahuan.
Aktor	Pelanggan
Skenario Utama	
Kondisi Awal	Pelanggan telah berhasil login dan sudah melakukan penyewaan.
Aksi Aktor	Reaksi Sistem
	1. Sistem menampilkan pemberitahuan pada ponsel pelanggan.
2. Pelanggan memilih notifikasi yang ingin dilihat.	3. Sistem menampilkan halaman sesuai dengan isi pemberitahuan.
Kondisi Akhir	Sistem menampilkan pemberitahuan.

4.1.8.6 Skenario Use Case Pemilik Rental

Tabel 4.40 Skenario *Use Case* Menambah data kendaraan

Skenario Kasus pada Sistem	
Nomor *Use Case*	SRS_003_01
Nama	Menambah data kendaraan
Tujuan	Agar pemilik rental dapat menambah data kendaraan yang disewakan kedalam sistem.

Deskripsi	*Use case* ini menjelaskan proses untuk melakukan penambahan data kendaraan yang disewakan dalam sistem.
Aktor	Pemilik rental
Skenario Utama	
Kondisi Awal	Pemilik rental berhasil login.
Aksi Aktor	Reaksi Sistem
1. Pemilik rental menekan tombol menu.	
2. Pemilik rental menekan menu manajemen kendaraan.	3. Sistem menampilkan halaman manajemen kendaraan.
4. Pemilik rental menekan tombol + atau tambah data kendaraan.	5. Sistem menampilkan *form* isian data kendaraan.
6. Pemilik rental mengisi *form* isian data kendaraan.	
7. Pemilik rental menekan tombol simpan.	8. Sistem akan mengecek apakah form isian data kendaraan sudah lengkap atau belum. Jika sudah lengkap maka sistem akan menampilkan pesan pemberitahuan bahwa data telah tersimpan.
Skenario alternatif 1 : form isian penilaian belum lengkap	
	9. Sistem akan menampilkan pesan bahwa form isian data kendaraan belum lengkap.
Kondisi Akhir	Sistem menyimpan data kendaraan yang ditambahkan.

Tabel 4.41 Skenario *Use Case* Melihat Data Kendaraan

Skenario Kasus pada Sistem	
Nomor *Use Case*	SRS_003_02
Nama	Melihat data kendaraan
Tujuan	Agar pemilik rental kendaraan bisa melihat data kendaraan yang disewakan.
Deskripsi	Use case ini menjelaskan tentang proses melihat data kendaraan yang tersimpan dalam sistem.
Aktor	Pemilik rental kendaraan
Skenario Utama	
Kondisi Awal	Pemilik rental mempunyai data kendaraan yang sudah tersimpan pada sistem.
Aksi Aktor	Reaksi Sistem
1. Pelanggan menekan tombol menu.	
2. Pelanggan menekan menu manajemen kendaraan.	3. Sistem akan menampilkan daftar kendaraan yang tersimpan dalam sistem.
Kondisi Akhir	Sistem akan menampilkan daftar kendaraan yang tersimpan dalam sistem.

Tabel 4.42 Skenario *Use Case* Memperbarui data kendaraan

Skenario Kasus pada Sistem	
Nomor *Use Case*	SRS_003_03
Nama	Memperbarui data kendaraan
Tujuan	Agar pemilik rental dapat memperbarui data kendaraan yang sudah tersimpan dalam sistem.
Deskripsi	*Use case* ini menjelaskan tentang bagaimana proses memperbarui data kendaraan.
Aktor	Pemilik rental

Skenario Utama	
Kondisi Awal	Pemilik rental telah berhasil login.
Aksi Aktor	Reaksi Sistem
1. Pemilik rental tombol menu.	
2. Pemilik rental menekan menu manajemen kendaraan.	3. Sistem menampilkan daftar kendaraan yang tersimpan pada halaman manajemen kendaraan.
4. Pemilik rental memilih kendaraan yang akan diubah.	5. Sistem menampilkan detail informasi kendaraan dan tombol edit.
6. Pemilik rental menekan tombol edit data kendaraan.	7. Sistem menampilkan form isian edit data kendaraan.
8. Pemilik rental mengisi form isian edit data kendaraan.	
9. Pemilik rental menekan tombol simpan	10. Sistem akan mengecek apakah form isian edit kendaraan sudah lengkap atau belum, jika sudah lengkap maka sistem akan menyimpannya.
Skenario alternatif 1 : jika form isian edit data kendaraan belum lengkap	
	11. Sistem akan menampilkan pesan bahwa form isian edit kendaraan belum lengkap.
Kondisi Akhir	Sistem menyimpan data kendaraan yang sudah diubah.

Tabel 4.43 Skenario *Use Case* Menghapus data kendaraan

Skenario Kasus pada Sistem	
Nomor *Use Case*	SRS_003_04

Nama	Menghapus data kendaraan
Tujuan	Agar pemilik rental dapat mengapus data kendaraan yang sudah tersimpan dalam sistem.
Deskripsi	*Use case* ini menjelaskan tentang bagaimana proses menghapus data kendaraan.
Aktor	Pemilik rental
Skenario Utama	
Kondisi Awal	Pemilik rental telah berhasil login.
Aksi Aktor	Reaksi Sistem
1. Pemilik rental tombol menu.	
2. Pemilik rental menekan menu manajemen kendaraan.	3. Sistem menampilkan halaman manajemen kendaraan.
4. Pemilik rental memilih kendaraan yang akan diubah.	5. Sistem menampilkan detail informasi kendaraan dan tombol hapus.
6. Pemilik rental menekan tombol hapus data kendaraan.	7. Sistem menampilkan pesan konfirmasi apakah data ingin dihapus.
8. Pemilik rental menekan tombol konfirmasi hapus.	9. Sistem akan menghapus data kendaraan dari database sistem.
10. Pemilik rental menekan tombol simpan	11. Sistem akan mengecek apakah form isian edit kendaraan sudah lengkap atau belum, jika sudah lengkap maka sistem akan menyimpannya.
Skenario alternatif 1 : jika pemilik rental menekan tombol konfirmasi batal	
	12. Sistem tidak akan menghapus data kendaraan dan kembali kehalaman menajemen data kendaraan.
Kondisi Akhir	Sistem menghapus data kendaraan.

Tabel 4.44 Skenario *Use Case* Melihat daftar penyewaan

Skenario Kasus pada Sistem	
Nomor *Use Case*	SRS_003_05
Nama	Melihat daftar penyewaan pelanggan
Tujuan	Agar pemilik rental dapat mengetahui penyewaan kendaraan yang dikirimkan pelanggan.
Deskripsi	*Use case* ini menjelaskan tentang bagaimana proses melihat penyewaan dalam sistem.
Aktor	Pemilik rental
Skenario Utama	
Kondisi Awal	Pemilik rental telah berhasil login dan terdapat penyewaan pada rental.
Aksi Aktor	Reaksi Sistem
1. Pemilik rental menekan tombol menu.	
2. Pemilik rental menekan tombol kelola penyewaan.	3. Sistem menampilkan halaman daftar penyewaan.
Kondisi Akhir	Sistem menampilkan daftar penyewaan.

Tabel 4.45 Skenario *Use Case* Melihat informasi detail penyewaan pelanggan

Skenario Kasus pada Sistem	
Nomor *Use Case*	SRS_003_06
Nama	Melihat informasi detail penyewaan pelanggan
Tujuan	Agar pemilik rental kendaraan dapat mengetahui informasi detail penyewaan kendaraan.
Deskripsi	*Use case* ini menjelaskan tentang bagaimana proses melihat informasi detail penyewaan.
Aktor	Pemilik rental
Skenario Utama	

Kondisi Awal	Pemilik rental mempunyai daftar permintaan penyewaan yang dikirimkan oleh pelanggan.
Aksi Aktor	Reaksi Sistem
1. Pemilik rental menekan tombol menu kelola penyewaan.	2. Sistem menampilkan daftar penyewaan yang tersimpan.
3. Pelanggan memilih *tab menu* status penyewaan yang ingin dilihat dan memilih penyewaan dari daftar penyewaan ingin yang ditampilkan.	4. Sistem menampilkan informasi detail penyewaan.
Kondisi Akhir	Sistem menampilkan informasi detail penyewaan

Tabel 4.46 Skenario *Use Case* Melakukan konfirmasi pembayaran

Skenario Kasus pada Sistem	
Nomor *Use Case*	SRS_003_07
Nama	Melakukan konfirmasi pembayaran
Tujuan	Agar pemilik rental dapat melakukan konfirmasi pembayaran terhadap penyewaan.
Deskripsi	*Use case* ini menjelaskan tentang bagaimana proses melakukan konfirmasi pembayaran terhadap penyewaan penyewaan kendaraan.
Aktor	Pemilik rental
Skenario Utama	
Kondisi Awal	Pemilik rental mempunyai daftar permintaan penyewaan yang dikirimkan oleh pelanggan dengan status menunggu konfirmasi pembayaran.
Aksi Aktor	Reaksi Sistem

1. Pemilik rental menekan tombol kelola penyewaan	
2. Pemilik rental memilih *tab menu* dengan status menunggu konfirmasi pembayaran.	3. Sistem menampilkan daftar penyewaan dengan status menunggu konfirmasi pembayaran.
4. Pemilik rental menekan penyewaan yang ingin di konfirmasi.	5. Sistem menampilkan halaman detail penyewaan dan menampilkan tombol konfirmasi penyewaan.
6. Pemilik rental menekan tombol konfirmasi pembayaran pada halaman detail informasi penyewaan.	7. Sistem akan mengubah status penyewaan menjadi penyewaan berhasil.
Kondisi Akhir	Sistem akan mengubah status penyewaan menjadi berhasil dan menampilkan daftar penyewaan dengan status berhasil

Tabel 4.47 Skenario *Use Case* Melakukan konfirmasi penyewaan selesai

Skenario Kasus pada Sistem	
Nomor *Use Case*	SRS_003_08
Nama	Melakukan konfirmasi penyewaan selesai
Tujuan	Agar pemilik rental dapat melakukan konfirmasi penyewaan kendaraan telah selesai dilakukan.
Deskripsi	*Use case* menjelaskan tentang proses melakukan konfirmasi penyewaan kendaraan telah selesai.
Aktor	Pemilik rental
Skenario Utama	

Kondisi Awal	Penyewaan sudah selesai dilakukan dan dalam status penyewaan berhasil.
Aksi Aktor	Reaksi Sistem
1. Pemilik rental menekan tombol menu kelola penyewaan.	
2. Pemilik rental memilih tab status sedang dalam penyewaan.	3. Sistem menampilkan daftar penyewaan dengan status sedang dalam penyewaan.
4. Pemilik rental menekan penyewaan yang ingin di konfirmasi.	5. Sistem menampilkan halaman detail penyewaan dan menampilkan tombol konfirmasi selesai.
6. Pemilik rental menekan tombol penyewaan selesai.	7. Sistem akan memproses dan mengubah status penyewaan menjadi telah selesai.
Kondisi Akhir	Sistem mengubah status penyewaan menjadi selesai dan mengerahkan ke halaman daftar penyewaan dengan status selesai.

Tabel 4.48 Skenario *Use Case* Memperbarui konfirmasi pembatalan

Skenario Kasus pada Sistem	
Nomor *Use Case*	SRS_003_09
Nama	Melakukan konfirmasi pembatalan
Tujuan	Agar pemilik rental dapat melakukan konfirmasi pembatalan.
Deskripsi	*Use case* ini menjelaskan tentang proses konfirmasi pembatalan atas penyewaan kendaraan yang diajukan pelanggan.
Aktor	Pemilik rental
Skenario Utama	
Kondisi Awal	Pemilik rental telah berhasil melakukan konfirmasi pembayaran dan memiliki daftar

	penyewaan dengan status pengajuan pembatalan.
Aksi Aktor	Reaksi Sistem
1. Pemilik rental menekan tombol kelola penyewaan	
2. Pemilik rental memilih *tab menu* dengan status pengajuan pembatalan.	3. Sistem menampilkan daftar penyewaan dengan status pengajuan pembatalan.
4. Pemilik rental memilih penyewaan yang ingin di konfirmasi.	5. Sistem menampilkan halaman detail penyewaan dan menampilkan tombol konfirmasi pembatalan.
6. Pemilik rental menekan tombol konfirmasi pembatalan.	7. Sistem menampilkan halaman unggah bukti pengembalian pembayaran.
8. Pemilik rental mengisi informasi pengembalian dana.	
9. Pemilik rental menekan tombol unggah.	10. Sistem akan menyimpan dan menampilkan pesan peringatan pembatalan berhasil
Skenario Alternatif 1 : jika data bukti pembayaran tidak lengkap	
	15. Sistem akan menampilkan pesan peringatan bahwa data yang diisikan belum lengkap.
Kondisi Akhir	Sistem menyimpan data pengembalian dana, mengubah status penyewaan menjadi batal dan menampilkan halaman kelola penyewaan dengan status batal.

Tabel 4.49 Skenario *Use Case* Melihat status penyewaan

Skenario Kasus pada Sistem	
Nomor *Use Case*	SRS_002_10
Nama	Melihat status penyewaan
Tujuan	Agar pemilik rental kendaraan bisa melihat status penyewaan kendaraan yang telah dilakukan.
Deskripsi	Use case ini menjelaskan tentang proses melihat status penyewaan.
Aktor	Pemilik rental kendaraan
Skenario Utama	
Kondisi Awal	Pemilik rental mempunyai data penyewaan yang tersimpan dalam sistem.
Aksi Aktor	Reaksi Sistem
1. Pelanggan menekan tombol menu.	
2. Pelanggan menekan menu kelola penyewaan.	3. Sistem akan menampilkan daftar penyewaan beserta status penyewaan.
Kondisi Akhir	Sistem akan menampilkan status penyewaan penyewaan kendaraan.

Tabel 4.50 Skenario *Use Case* Melihat profil pelanggan

Skenario Kasus pada Sistem	
Nomor *Use Case*	SRS_003_11
Nama	Melihat profil pelanggan
Tujuan	Agar dapat melihat profil pelanggan yang mengirimkan permintaan penyewaan.
Deskripsi	*Use case* ini menjelaskan tentang bagaimana proses melihat profil pelanggan.
Aktor	Pemilik rental
Skenario Utama	

Kondisi Awal	Pemilik rental telah mempunyai permintaan penyewaan.
Aksi Aktor	Reaksi Sistem
1. Pemilik rental menekan tombol lihat detail penyewaan pada daftar penyewaan.	2. Sistem menampilkan informasi detail dari penyewaan.
3. Pemilik rental menekan tombol lihat profil pelanggan.	4. Sistem menampilkan profil pelanggan.
Kondisi Akhir	Sistem menampilkan profil pelanggan.

Tabel 4.51 Skenario *Use Case* Melihat profil rental kendaraan

Skenario Kasus pada Sistem	
Nomor *Use Case*	SRS_003_12
Nama	Melihat profil rental kendaraan
Tujuan	Agar dapat melihat profil Pemilik rental.
Deskripsi	*Use case* ini menjelaskan tentang bagaimana proses melihat profil Pemilik rental.
Aktor	Pemilik rental.
Skenario Utama	
Kondisi Awal	Pemilik rental telah berhasil login.
Aksi Aktor	Reaksi Sistem
1. Pemilik rental menekan tombol menu.	
2. Pemilik rental menekan menu profilku.	3. Sistem menampilkan profil Pemilik rental.
Kondisi Akhir	Sistem menampilkan profil Pemilik rental.

Tabel 4.52 Skenario *Use Case* Memperbarui profil

Skenario Kasus pada Sistem	
Nomor *Use Case*	SRS_003_13
Nama	Memperbarui profil
Tujuan	Agar dapat memperbarui profil Pemilik rental.
Deskripsi	*Use case* ini menjelaskan tentang bagaimana proses memperbarui profil Pemilik rental.
Aktor	Pemilik rental.
Skenario Utama	
Kondisi Awal	Pemilik rental telah berhasil login.
Aksi Aktor	Reaksi Sistem
1. Pemilik rental menekan tombol menu.	
2. Pemilik rental menekan menu profilku.	3. Sistem menampilkan profil Pemilik rental dan menampilkan tombol edit profil.
4. Pemilik rental menekan tombol edit profil.	5. Sistem menampilkan form isian edit biodata.
6. Pemilik rental mengisi form isian edit biodata.	
7. Pemilik rental menekan tombol simpan	8. Sistem akan mengecek apakah form isian sudah lengkap atau belum, jika sudah lengkap maka sistem akan menyimpannya.
Skenario alternatif 1 : jika form isian edit profil belum lengkap	
	9. Sistem akan menampilkan pesan bahwa form edit profil belum lengkap.
Kondisi Akhir	Sistem menampilkan profil Pemilik rental.

Tabel 4.53 Skenario *Use Case* Logout

Skenario Kasus pada Sistem	
Nomor *Use Case*	SRS_003_14
Nama	Logout
Tujuan	Agar Pemilik rental dapat keluar dari sistem.
Deskripsi	*Use case* ini menjelaskan tentang bagaimana proses logout yang dilakukan Pemilik rental.
Aktor	Pemilik rental
Skenario Utama	
Kondisi Awal	Pemilik rental telah berhasil login.
Aksi Aktor	Reaksi Sistem
1. Pemilik rental menekan tombol menu.	
2. Pemilik rental menekan menu logout.	3. Sistem menampilkan halaman login.
Kondisi Akhir	Sistem menampilkan halaman login.

4.1.8.7 Skenario Use Case Pemilik Rental Iterasi 1

Tabel 4.54 Skenario *Use Case* Melihat Penilaian dan Ulasan

Skenario Kasus pada Sistem	
Nomor *Use Case*	SRS_003_15
Nama	Melihat Penilaian dan Ulasan
Tujuan	Agar pemilik rental bisa melihat penilaian dan ulasan dari pelanggan.
Deskripsi	*Use case* ini menjelaskan tentang bagaimana proses pemilik rental dapat melihat penilaian dan ulasan dari pelanggan.
Aktor	Pemilik rental
Skenario Utama	

Kondisi Awal	Pemilik rental telah berhasil login
Aksi Aktor	Reaksi Sistem
1. Pemilik rental menekan tombol menu.	
2. Pemilik rental memilik menu Ulasan.	3. Sistem akan menampilkan daftar penilaian dan ulasan dari pelanggan terhadap penyewaan yang pernah dilakukan.
Kondisi Akhir	Sistem menampilkan daftar penilaian dan ulasan.

Tabel 4.55 Skenario *Use Case* Menambah Rekening Pembayaran

Skenario Kasus pada Sistem	
Nomor *Use Case*	SRS_003_16
Nama	Menambah Rekening Pembayaran
Tujuan	Agar pemilik rental dapat melakukan penambahan rekening pembayaran.
Deskripsi	*Use case* ini menjelaskan tentang bagaimana proses pemilik rental dapat melakukan penambahan rekening pembayaran dalam sistem.
Aktor	Pemilik rental
Skenario Utama	
Kondisi Awal	Pemilik rental telah berhasil login
Aksi Aktor	Reaksi Sistem
1. Pemilik rental menekan tombol menu.	
2. Pemilik rental menekan menu profil.	3. Sistem menampilkan profil pemilik rental.
4. Pemilik rental menekan tombol pengaturan profil.	5. Sistem menampilkan tombol ubah profil dan pengaturan rekening rental.

6. Pemilik rental menakan tombol pengaturan rekening rental.	7. Sistem menampilkan daftar rekening pembayaran dan tombol tambah.
8. Pemilik rental menekan tombol tambah.	9. Sistem menampilkan kolom isian penambahan rekening.
10. Pemilik rental mengisi kolom isian data rekening.	
11. Pemilik rental menekan tombol tambah.	12. Sistem melakukan cek kolom isian. Apabila semua sudah terisi akan melakukan menyimpan rekening pembayaran.
Skenario alternatif 1 : jika kolom isian masih kosong	
	13. Sistem menampilkan pesan peringatan untuk melengkapi kolom isian.
Kondisi Akhir	Sistem menyimpan rekening pembayaran.

Tabel 4.56 Skenario *Use Case* Mengubah Rekening Pembayaran

Skenario Kasus pada Sistem	
Nomor *Use Case*	SRS_003_17
Nama	Mengubah Rekening Pembayaran
Tujuan	Agar pemilik rental dapat melakukan perubahan pada rekening pembayaran tang sebelumnya tersimpan.
Deskripsi	*Use case* ini menjelaskan tentang bagaimana proses pemilik rental dapat melakukan perubahan rekening pembayaran dalam sistem.
Aktor	Pemilik rental
Skenario Utama	
Kondisi Awal	Pemilik rental telah berhasil masuk halaman pengaturan rekening pembayaran.
Aksi Aktor	Reaksi Sistem

1. Pemilik rental memilih rekening yang ingin diubah.	2. Sistem menampilkan detail rekening pembayaran, tombol edit dan hapus.
3. Pemilik rental menekan tombol edit.	4. Sistem menampilkan halaman ubah rekening pembayaran.
5. Pemilik rental melakukan perubahan terhadap data yang ingin diubah.	
6. Pemilik rental menekan tombol simpan.	7. Sistem melakukan cek kolom isian. Apabila semua sudah terisi akan melakukan menyimpan rekening pembayaran yang diubah.
Skenario alternatif 1 : jika kolom isian masih kosong	
	8. Sistem menampilkan pesan peringatan untuk melengkapi kolom isian.
Kondisi Akhir	Sistem menyimpan rekening pembayaran yang berhasil diubah.

Tabel 4.57 Skenario *Use Case* Menghapus Rekening Pembayaran

Skenario Kasus pada Sistem	
Nomor *Use Case*	SRS_003_18
Nama	Melihat Penilaian dan Ulasan
Tujuan	Agar pemilik rental dapat melakukan hapus rekening pembayaran yang sebelumnya tersimpan.
Deskripsi	*Use case* ini menjelaskan tentang bagaimana proses pemilik rental dapat melakukan hapus rekening pembayaran dalam sistem.
Aktor	Pemilik rental
Skenario Utama	

Kondisi Awal	Pemilik rental telah berhasil masuk halaman pengaturan rekening pembayaran.
Aksi Aktor	Reaksi Sistem
1. Pemilik rental memilih rekening yang ingin dihapus.	2. Sistem menampilkan detail rekening pembayaran, tombol edit dan hapus.
3. Pemilik rental menekan tombol hapus.	4. Sistem menampilkan pesan konfirmasi hapus.
5. Pemilik rental menekan tombol hapus.	6. Sistem menghapus rekening pembayaran.
Kondisi Akhir	Sistem menghapus rekening pembayaran.

4.1.8.8 Skenario Use Case Pemilik Rental Iterasi 2

Tabel 4.58 Skenario *Use Case* Menerima Pemberitahuan

Skenario Kasus pada Sistem	
Nomor *Use Case*	SRS_003_19
Nama	Menerima Pemberitahuan
Tujuan	Agar pemilik rental bisa melakukan menerima pemberitahuan pada kondisi tertentu.
Deskripsi	*Use case* ini menjelaskan tentang bagaimana proses menerima pemberitahuan.
Aktor	Pemilik rental
Skenario Utama	
Kondisi Awal	Pemilik rental telah berhasil login dan terdapat penyewaan.
Aksi Aktor	Reaksi Sistem
	1. Sistem menampilkan pemberitahuan pada ponsel pelanggan.
2. Pelanggan memilih notifikasi yang ingin dilihat.	3. Sistem menampilkan halaman sesuai dengan isi pemberitahuan.

Kondisi Akhir	Sistem menampilkan pemberitahuan.

Tabel 4.59 Skenario *Use Case* Melakukan konfirmasi pembayaran

Skenario Kasus pada Sistem	
Nomor *Use Case*	SRS_003_07
Nama	Melakukan konfirmasi pembayaran
Tujuan	Agar pemilik rental dapat melakukan konfirmasi pembayaran terhadap penyewaan.
Deskripsi	*Use case* ini menjelaskan tentang bagaimana proses melakukan konfirmasi pembayaran terhadap penyewaan penyewaan kendaraan.
Aktor	Pemilik rental
Skenario Utama	
Kondisi Awal	Pemilik rental menerima pemberitahuan jika ada pembayaran yang dikirimkan.
Aksi Aktor	Reaksi Sistem
1. Pemilik rental menekan tombol *builder* pemberitahuan.	2. Sistem mengarahkan ke halaman status penyewaan dengan status menunggu konfirmasi pembayaran.
	3. Sistem menampilkan daftar penyewaan dengan status menunggu konfirmasi pembayaran.
4. Pemilik rental menekan penyewaan yang ingin di konfirmasi.	5. Sistem menampilkan halaman detail penyewaan dan menampilkan tombol konfirmasi penyewaan.
6. Pemilik rental menekan tombol konfirmasi pembayaran pada halaman detail informasi penyewaan.	7. Sistem akan mengubah status penyewaan menjadi penyewaan berhasil.

Kondisi Akhir	Sistem akan mengubah status penyewaan menjadi berhasil dan menampilkan daftar penyewaan dengan status berhasil

Tabel 4.60 Skenario *Use Case* Melakukan konfirmasi pembatalan

Skenario Kasus pada Sistem	
Nomor *Use Case*	SRS_003_09
Nama	Melakukan konfirmasi pembatalan
Tujuan	Agar pemilik rental dapat melakukan konfirmasi pembatalan.
Deskripsi	*Use case* ini menjelaskan tentang proses konfirmasi pembatalan atas penyewaan kendaraan yang diajukan pelanggan.
Aktor	Pemilik rental
Skenario Utama	
Kondisi Awal	Pemilik rental menerima pemberitahuan pengajuan pembatalan
Aksi Aktor	Reaksi Sistem
1. Pemilik rental menekan *builder* pemberitahuan.	2. Sistem mengarahkan ke halaman status penyewaan dengan status menunggu konfirmasi pembayaran.
	3. Sistem menampilkan daftar penyewaan dengan status pengajuan pembatalan.
4. Pemilik rental memilih penyewaan yang ingin di konfirmasi.	5. Sistem menampilkan halaman detail penyewaan dan menampilkan tombol konfirmasi pembatalan.
6. Pemilik rental menekan tombol konfirmasi pembatalan.	7. Sistem menampilkan halaman unggah bukti pengembalian pembayaran.
8. Pemilik rental mengisi informasi	

pengembalian dana.	
9. Pemilik rental menekan tombol unggah.	10. Sistem akan menyimpan dan menampilkan pesan peringatan pembatalan berhasil
Skenario Alternatif 1 : jika data bukti pembayaran tidak lengkap	
	16. Sistem akan menampilkan pesan peringatan bahwa data yang diisikan belum lengkap.
Kondisi Akhir	Sistem menyimpan data pengembalian dana, mengubah status penyewaan menjadi batal dan menampilkan halaman kelola penyewaan dengan status batal.

Tabel 4.61 Skenario *Use Case* Melakukan Penolakkan Pembayaran

Skenario Kasus pada Sistem	
Nomor *Use Case*	SRS_003_20
Nama	Melakukan penolakkan pembayaran
Tujuan	Agar pemilik rental dapat melakukan penolakkan pembayaran atas penyewaan dari pelanggan.
Deskripsi	*Use case* ini menjelaskan tentang proses konfirmasi penolakkan pembayaran atas penyewaan kendaraan yang diajukan pelanggan.
Aktor	Pemilik rental
Skenario Utama	
Kondisi Awal	Pemilik rental menerima pemberitahuan terdapat penyewaan baru dengan status menunggu konfirmasi pembayaran.
Aksi Aktor	Reaksi Sistem
1. Pemilik rental menekan *builder* pemberitahuan.	2. Sistem mengarahkan ke halaman status penyewaan dengan status menunggu konfirmasi pembayaran.

	3. Sistem menampilkan daftar penyewaan dengan status menunggu konfirmasi pembayaran.
4. Pemilik rental memilih penyewaan yang ingin di tolak.	5. Sistem menampilkan halaman detail penyewaan dan menampilkan tombol tolak pembayaran.
6. Pemilik rental menekan tombol tolak pembayaran.	7. Sistem menampilkan halaman input alasan penolakkan pembayaran.
8. Pemilik rental memilih alasan penolakkan.	
Skenario Alternatif 1 : jika alasan penolakkan karena kendaraan tidak tersedia	
9. Pemilik rental menekan tombol konfirmasi penolakkan.	10. Sistem mengarahkan ke halaman pengembalian dana untuk melakukan unggah bukti pembayaran pengembalian dana.
Skenario Alternatif 2 : jika alasan penolakkan karena pembayaran kurang	
	11. Sistem akan mengarahkan ke halaman kelola penyewaan dengan status menunggu sisa pembayaran.
Kondisi Akhir	Sistem menyimpan data pengembalian dana, mengubah status penyewaan menjadi batal atau menunggu sisa pembayaran.

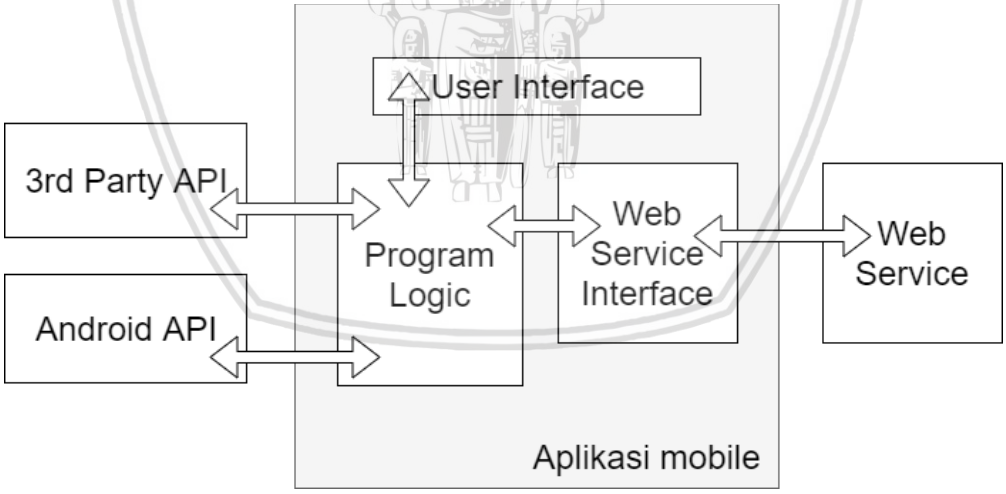
BAB 5 PERANCANGAN DAN IMPLEMENTASI

5.1 Perancangan Sistem

Perancangan sistem dilakukan setelah proses rekayasa kebutuhan dilakukan. Tahapan perancangan ini juga mengadopsi beberapa tahapan yang ada dalam metode pengembangan MASAM yaitu desain antarmuka (*UI Design*) yang akan dijelaskan pada sub bab perancangan antarmuka, definisi arsitektur (a*rchitecture definition*) yang akan dijelaskan pada sub bab perancangan arsitektural dan manajemen pola perancangan (*pattern management*) yang akan dijelaskan pada sub bab manajemen pola perancangan. Selain itu didalam tahapan perancangan sistem ini terdapat perancangan diagram *class,* perancangan basis data dan perancangan diagram *sequence*.

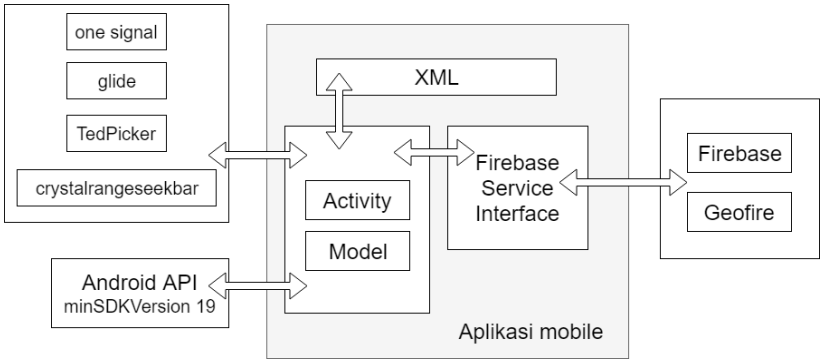
5.1.1 Perancangan Arsitektural

Pada penelitian ini, sistem yang akan dikembangkan menggunakan arsitektur *front-end* dan *back-end.* Sehingga pengolahan antarmuka (*front-end)* dengan pengolahan data (*back-end)* diolah secara terpisah. Pada perancangan arsitektural di bedakan menjadi dua bagian, yang pertama adalah arsitektur dasar dari pengembangan suatu aplikasi android dan arsitektur kedua adalah arsitektur yang menggambarkan arsitektur pada penelitian ini. Gambar 5.1 menunjukkan arsitektur dasar dan Gambar 5.2 menunjukkan arsitektur Sistem Penyewaan Kendaraan Kota Malang.



Gambar 5.1 Arsitektur Dasar Sistem

Sumber : (Stapic, 2013)



Gambar 5.2 Arsitektur Sistem Penyewaan Kendaraan Kota Malang

5.1.2 Perancangan Basis Data

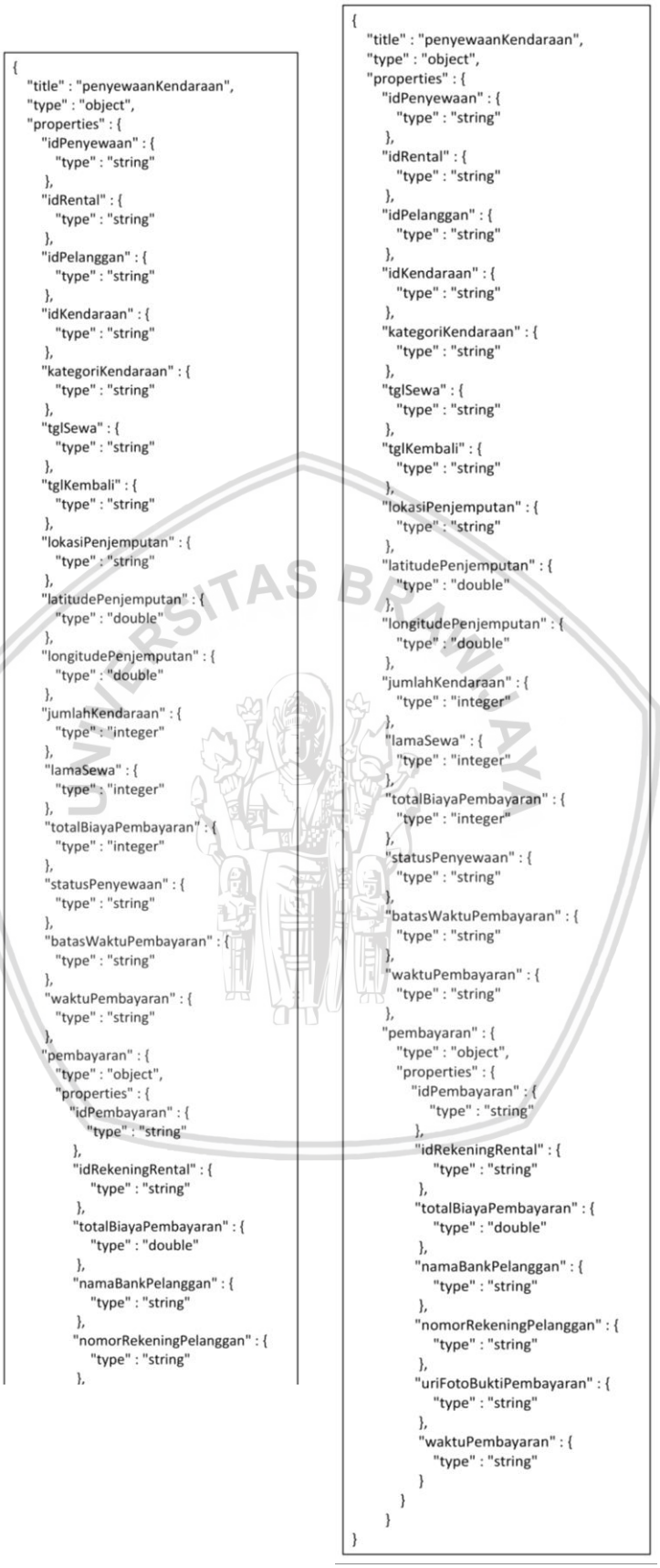
Perancangan basis data dilakukan dengan menggunakan skema diagram JSON *Schema* karena pada penelitian ini basis data yang digunakan adalah *Firebase Real-time database* berbasis NoSQL. Sehingga pada perancangan basis data tidak terdapat relasi. Perancangan basis data ditunjukkan oleh Gambar 5.3.

```
{
  "title" : "pelanggan",
  "type" : "object",
  "properties" : {
    "idPelanggan" : {
      "type" : "string"
    },
    " noIdentitas" : {
      "type" : "string"
    },
    " namaLengkap" : {
      "type" : "string"
    },
    " alamat" : {
      "type" : "string"
    },
    " noTelp" : {
      "type" : "string"
    },
    "uriFotoProfil" : {
      "type" : "string"
    },
    "deviceToken" : {
      "type" : "string"
    }
  }
}
```

```
{
  "title" : "kendaraan",
  "type" : "object",
  "properties" : {
    "idKendaraan" : {
      "type" : "string"
    },
    "idRental" : {
      "type" : "string"
    },
    "kategoriKendaraan" : {
      "type" : "string"
    },
    " tipeKendaraan" : {
      "type" : "string"
    },
    " hargaSewa" : {
      "type" : "double"
    },
    "jumlahKendaraan" : {
      "type" : "Integer"
    },
    "jumlahPenumpang" : {
      "type" : "string"
    },
    "fasilitas" : {
      "type" : "string"
    },
    "areaPemakaian" : {
      "type" : "string"
    },
    "durasiPenyewaan" : {
      "type" : "string"
    },
    "supir" : {
      "type" : "boolean"
    },
    "bahanBakar" : {
      "type" : "boolean"
    },
    "uriFotoKendaraan" : {
      "type" : "array",
      "items" : {
        "type" : "string"
      }
    }
  }
}
```

```
{
  "title" : "cekSisaKendaraan",
  "type" : "object",
  "properties" : {
    "idCekSisa" : {
      "type" : "string"
    },
    "idKendaraan" : {
      "type" : "string"
    },
    "sisaKendaraan" : {
      "type" : "integer"
    },
    "tglSewa" : {
      "type" : "string"
    },
    "tglKembali" : {
      "type" : "string"
    }
  }
}
```

```
{
  "title" : "penilaian",
  "type" : "object",
  "properties" : {
    "idPenilaian" : {
      "type" : "string"
    },
    "idPenyewaan" : {
      "type" : "string"
    },
    "idRental" : {
      "type" : "string"
    },
    "idPelanggan" : {
      "type" : "string"
    },
    "idKendaraan" : {
      "type" : "string"
    },
    "penilaian" : {
      "type" : "number"
    },
    "ulasan" : {
      "type" : "string"
    }
  }
}
```



Gambar 5.3 Perancangan Basis Data

5.1.3 Perancangan Diagram *Sequence*

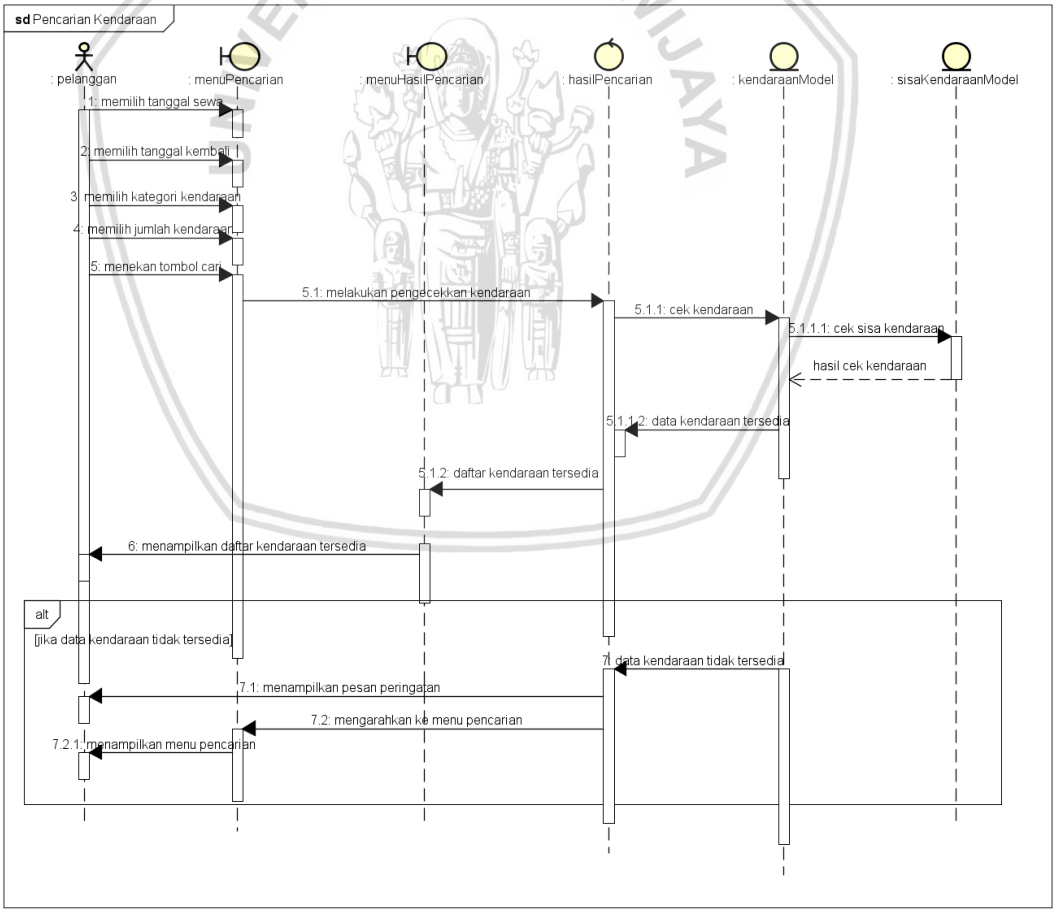
Diagram *sequence* merupakan representasi dari proses interaksi antar objek sesuai dengan urutan prosesnya. Diagram *Sequence* menunjukkan bagaimana pertukaran serangkaian pesan antar objek-objek yang melakukan suatu aksi tertentu. Diagram *Sequence* digunakan untuk menggambarkan arus pekerjaan, pesan yang disampaikan dan elemen-elemen yang bekerja sama didalamnya dari waktu ke waktu untuk mencapai suatu tujuan tertentu.

5.1.3.1 Diagram *Sequence* pada sisi Pelanggan

Pada bagian diagram *sequence* sisi pelanggan akan dijelaskan dua buah diagram *sequence* yaitu diagram *sequence* melakukan pencarian kendaraan dan diagram *sequence* melakukan penyewaan kendaraan.

a. Diagram *Sequence* Melakukan Pencarian Kendaraan

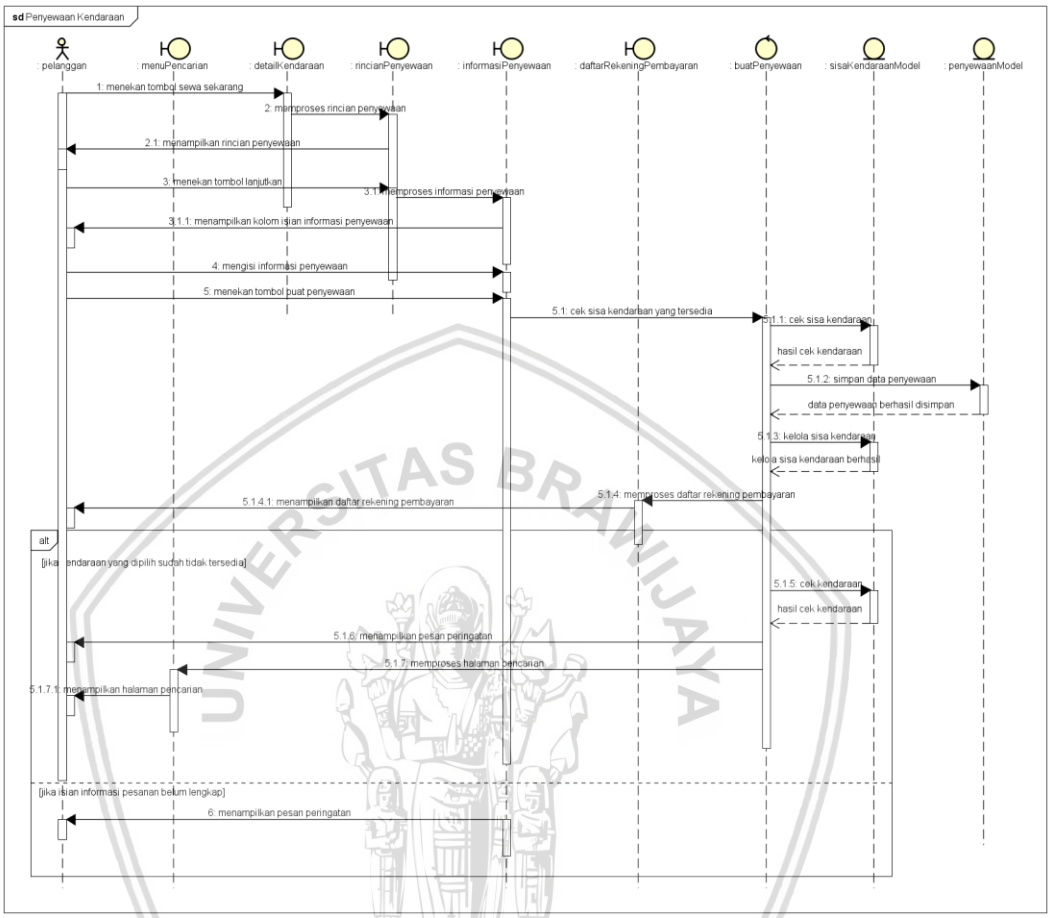
Gambar 5.4 dibawah ini merupakan diagram *sequence* yang menggambarkan interaksi ketika pelanggan melakukan proses pencarian kendaraan didalam sistem.



Gambar 5.4 Diagram *Sequence* Melakukan Pencarian Kendaraan

b. Diagram *Sequence* Melakukan Penyewaan Kendaraan

Gambar 5.5 merupakan diagram *sequence* yang menggambarkan interaksi ketika pelanggan melakukan proses penyewaan kendaraan didalam sistem.



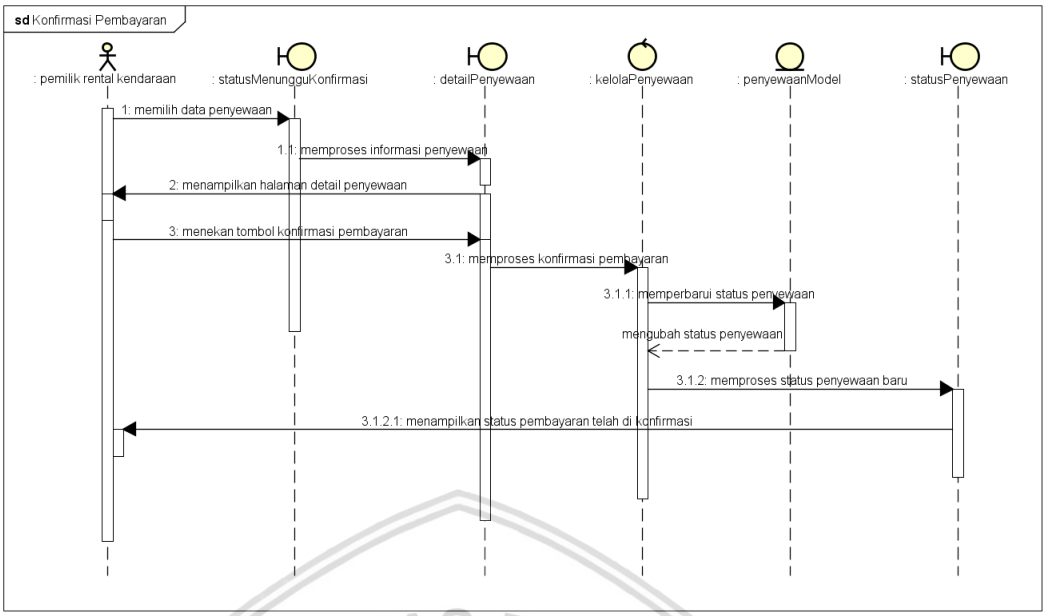
Gambar 5.5 Diagram *Sequence* Melakukan Penyewaan Kendaraan

5.1.3.2 Diagram Sequence pada sisi Pemilik rental kendaraan

Pada bagian diagram *sequence* sisi pemilik rental kendaraan akan dijelaskan dua buah diagram s*equence* yaitu diagram *sequence* melakukan konfirmasi pembayaran dan diagram *sequence* menambah data kendaraan.

a. Diagram *Sequence* Melakukan Konfirmasi Pembayaran

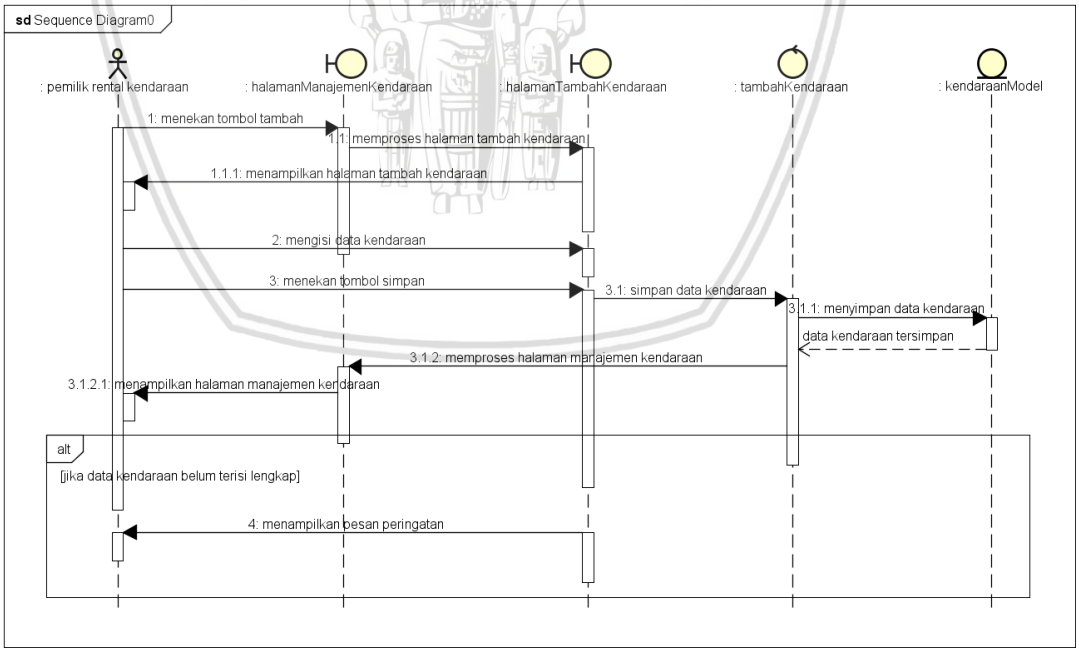
Gambar 5.6 merupakan diagram *sequence* yang menggambarkan interaksi ketika pemilik rental kendaraan melakukan proses konfirmasi pembayaran.



Gambar 5.6 Diagram *Sequence* Melakukan Konfirmasi Pembayaran

b. Diagram *Sequence* Menambah Data Kendaraan

Gambar 5.7 dibawah ini merupakan diagram *sequence* yang menggambarkan interaksi ketika pemilik rental kendaraan melakukan proses penambahan data kendaraan.



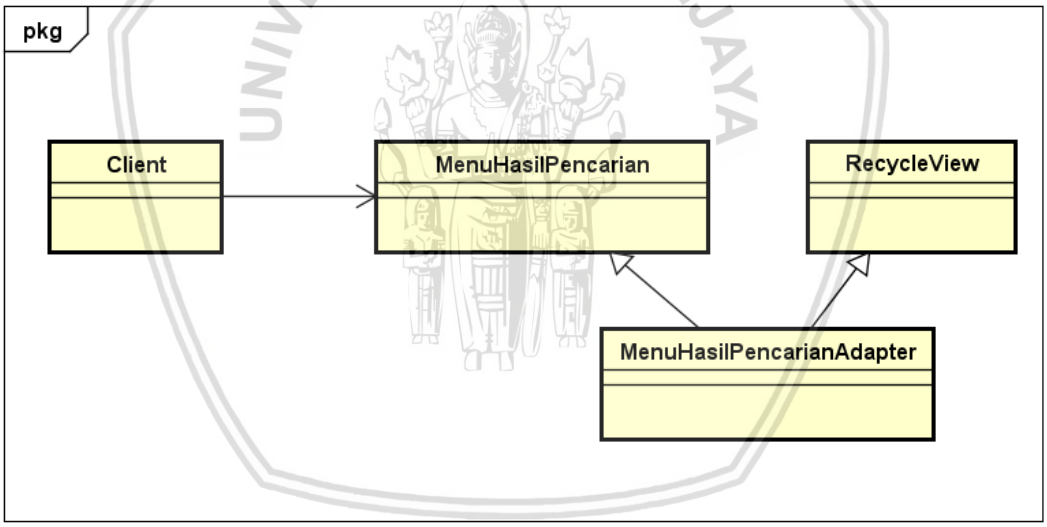
Gambar 5.7 Diagram Sequence Menambah Data Kendaraan

5.1.4 Manajemen Pola Perancangan

Pada *Embodiment Phase* yang terdapat pada fase pengembangan metode MASAM, terdapat aktivitas *Architecturing* yang dimana didalamnya terdapat *task pattern management* atau manajemen pola perancangan. Tahapan ini merupakan salah satu karakteristik pada metode pengembangan MASAM, karena diharapkan dengan adanya manajemen pola perancangan dapat memudahkan pengembangan sistem penyewaan kendaraan kota Malang. Pada pengembangan sistem ini terdapat beberapa pola perancangan yang akan diimplementasikan yaitu pola perancangan *adapter* dan *command.* Pada bagian ini akan dijelaskan peran masing-masing pola perancangan dalam pengembangan sistem ini. Selanjutnya gambaran tentang pola perancangan tersebut akan di gambarkan dalam bagian ini.

5.1.4.1 Pola Perancangan *Adapter*

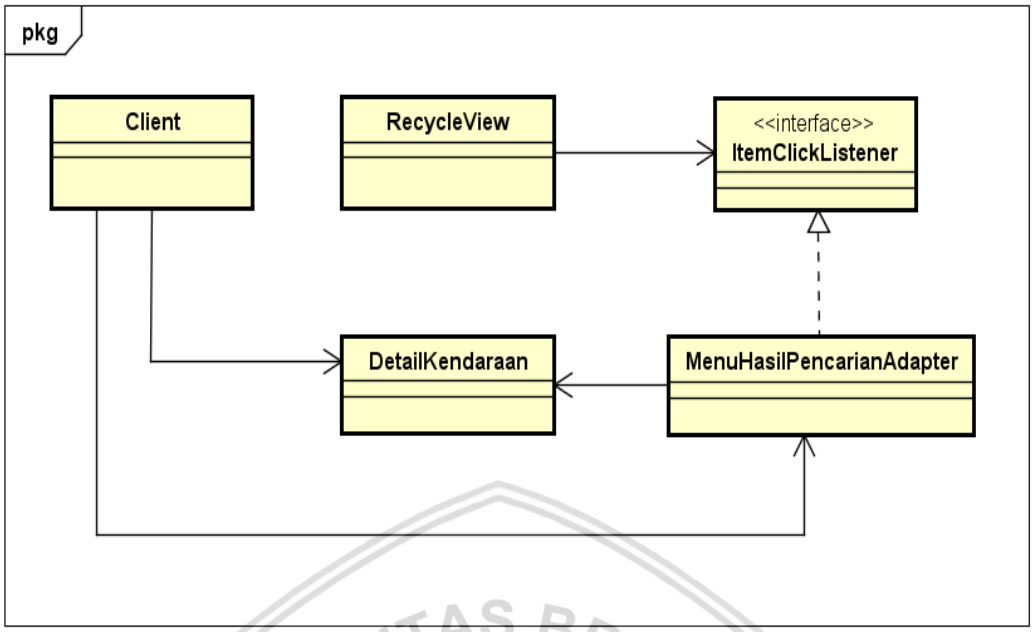
Pada sistem ini, pola perancangan *adapter* digunakan untuk menampilkan suatu data dalam bentuk *list*, seperti untuk menampilkan daftar kendaraan. Gambar 5.8 akan menunjukkan perancangan dari penggunaan pola adapter pada penelitian ini.



Gambar 5.8 Perancangan *Adapter Pattern*

5.1.4.2 Pola Perancangan *Command*

Pada sistem ini, pola perancangan *command* digunakan untuk melakukan fungsi `ItemClickListener` dari sebuah adapter. Gambar 5.9 menunjukkan perancangan *Command* pada penelitian ini.



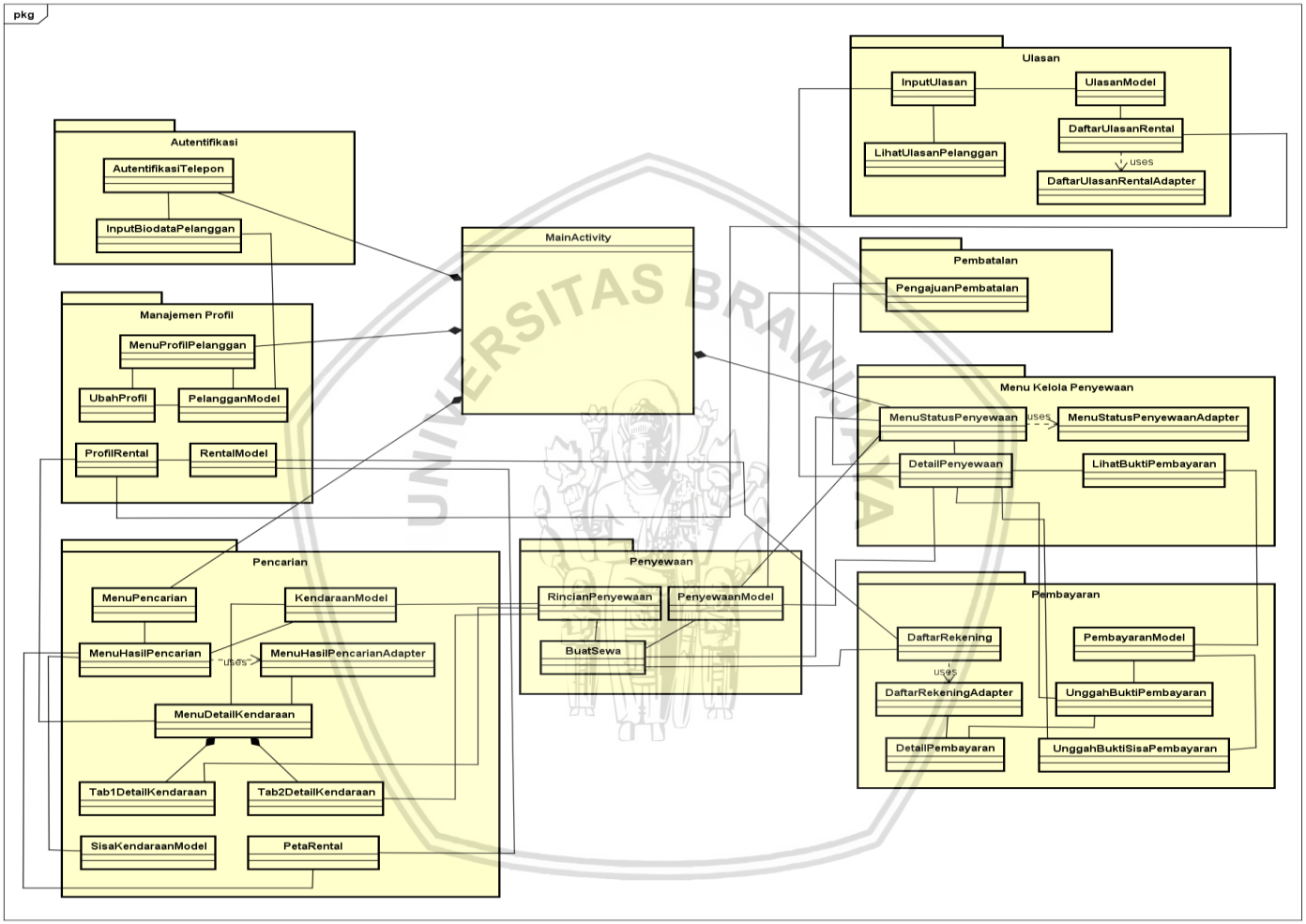
Gambar 5.9 Perancangan *Command Pattern*

5.1.5 Perancangan *Class Diagram*

Perancangan *Class diagram* merupakan diagram yang digunakan untuk menujukkan kelas dan keterhubungannya di dalam suatu sistem. *Class* yang digunakan untuk membangun sistem ini dikelompokkan menjadi beberapa paket berdasarkan fungsi-fungsi yang akan diimplementasikan didalam sistem. *Class diagram* juga terbagi menjadi dua bagian yaitu *class diagram* pelanggan dan *class diagram* pemilik rental. Bagian ini merupakan kelanjutan dari penjelasan perancangan arsitektural pada bagian program logic yang terdiri dari activity dan model. Penjabaran tentang activity dan model akan dijelaskan pada bagian ini.

5.1.5.1 Perancangan *Class Diagram* Pelanggan

Pada *class diagram* pelanggan terbagi menjadi beberapa paket yaitu `paket autentifikasi`, `paket manajemen profil`, `paket pencarian`, `paket kelola penyewaan`, `paket pembayaran`, `paket status penyewaan`, `paket pembatalan` dan `paket ulasan`. Berikut ini *class diagram* dari sisi pemilik rental beserta relasi antar *class* yang ditunjukkan pada Gambar 5.10.

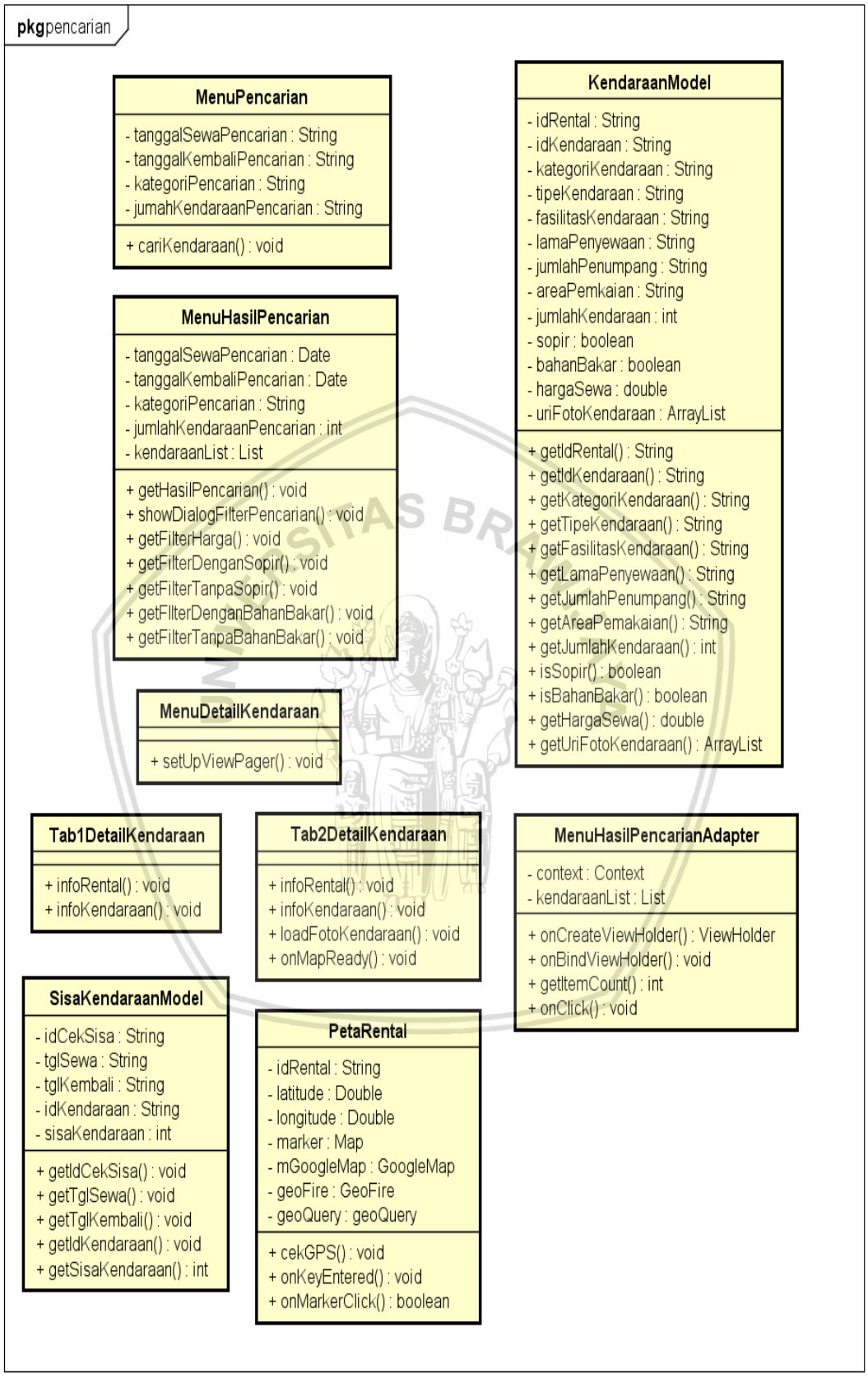


Gambar 5.10 *Class Diagram* Pelanggan

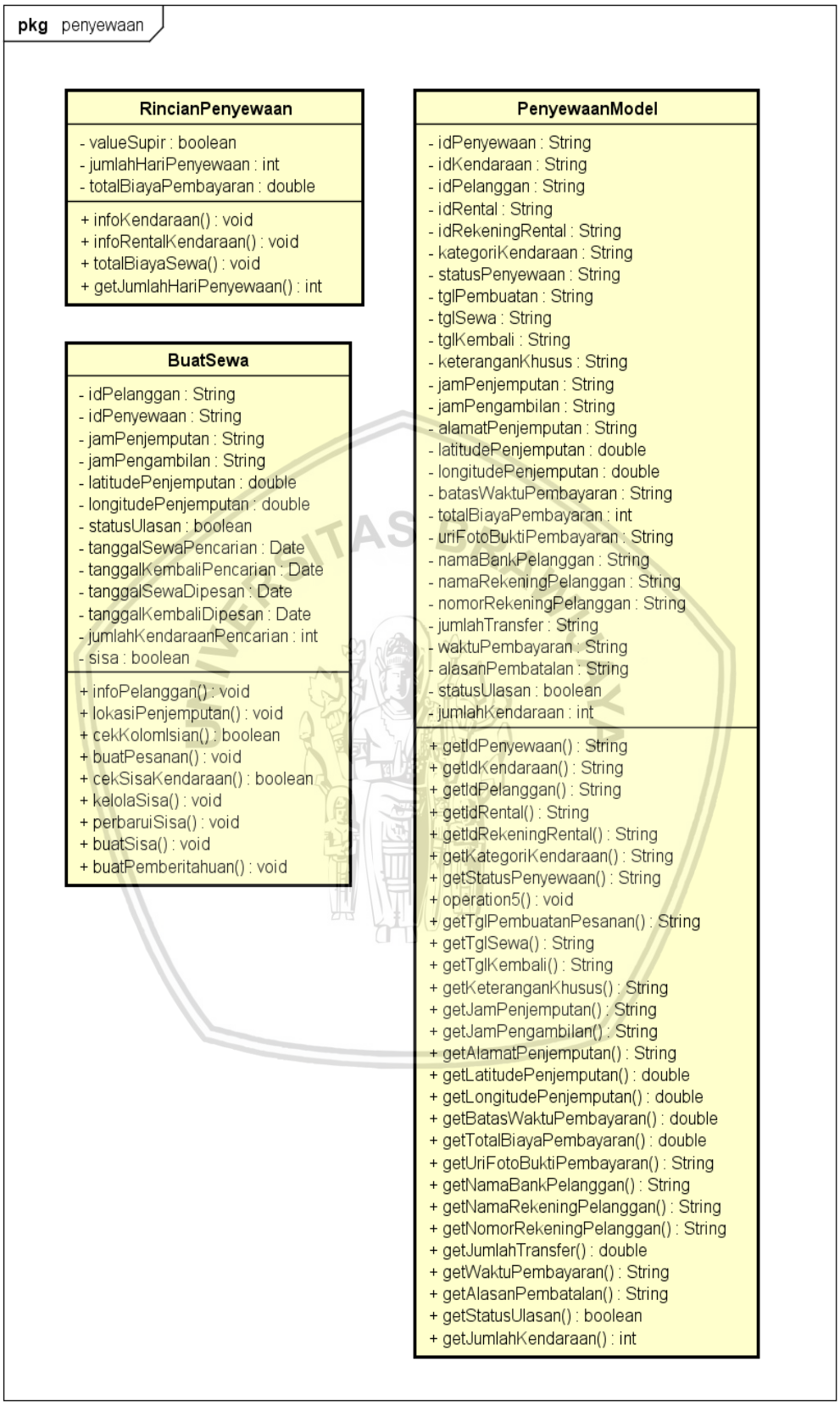
Dari delapan paket pada *class diagram* pelanggan akan dijelaskan tiga paket utama beserta atribut dan operasi yang ada di dalam *class* tersebut. Paket utama yang akan di jelaskan adalah `paket pencarian`, `paket penyewaan` dan `paket pembayaran`.

Paket pencarian terdiri dari tujuh *class* yaitu *class* `MenuPencarian`, `HasilPencarian`, `Hasil Pencarian Adapter`, `DetailKendaraan`, `Tab1DetailKendaraan`, `Tab2DetailKendaraan`, `PetaRental`, `KendaraanModel` dan `SisaKendaraanModel`. *Class* `MenuPencarian kendaraan` merupakan *class* yang berfungsi bagi pelanggan untuk dapat melakukan pencarian kendaraan yang tersedia berdasarkan pada tanggal sewa, tanggal kembali, jumlah kendaraan dan kategori kendaraan. *Class* `HasilPencarian` merupakan *class* yang berfungsi untuk melakukan pencarian kendaraan sesuai dengan variabel pencarian yang dimasukkan oleh pelanggan pada `MenuPencarian`. Sedangkan `HasilPencarianAdapter` merupakan *class* yang berfungsi mengatur tampilan hasil pencarian kedalam bentuk ListView yang dimana *class* HasilPencarian menggunakan *class* `HasilPencarianAdapter`. Selanjutnya adalah `DetailKendaraan` merupakan suatu *class* yang menginisialisasi dua buah fragment yaitu *class* `Tab1DetailKendaraan` dan `Tab2DetailKendaraan` untuk digunakan sebagai tab menu untuk memuat informasi detail kendaraan. *class* `PetaRental` merupakan suatu *class* yang berfungsi untuk menampilkan lokasi rental dalam bentuk peta dengan radius atau jarak tertentu dari lokasi pengguna. *Class* `KendaraanModel` dan `SisaKendaraanModel` merupakan suatu *class* yang menyediakan operasi *getter* data kendaraan dan juga berfungsi sebagai *entity class.* Atribut dan operasi dari tujuh buah *class* tersebut dapat dilihat dalam *class diagram* pada Gambar 5.11.

Paket penyewaan terdiri dari tiga buah *class* yaitu *class* `RincianPenyewaan`, `BuatPenyewaan` dan `PenyewaanModel`. *Class* `RincianPenyewaan` merupakan activity *class* yang berfungsi untuk menampilkan rincian penyewaan yang dilakukan oleh pelanggan, melakukan kalkulasi lama penyewaan kendaraan dan juga melakukan kalkulasi total pembayaran atas penyewaan yang dilakukan oleh pelanggan. *Class* `BuatPenyewaan` merupakan activity *class* yang berfungsi untuk melakukan penyewaan kendaraan dan menyimpannya dalam sistem. *Class* ini juga menyediakan halaman isian bagi pelanggan untuk menambah informasi pemesan apabila dibutuhkan. Selanjutnya adalah *class* `PenyewaanModel` yang menyediakan operasi *getter* data penyewaan dan juga berfungsi sebagai *entity class.* Atribut dan operasi dari tiga buah *class* tersebut dapat dilihat dalam *class diagram* pada Gambar 5.12.

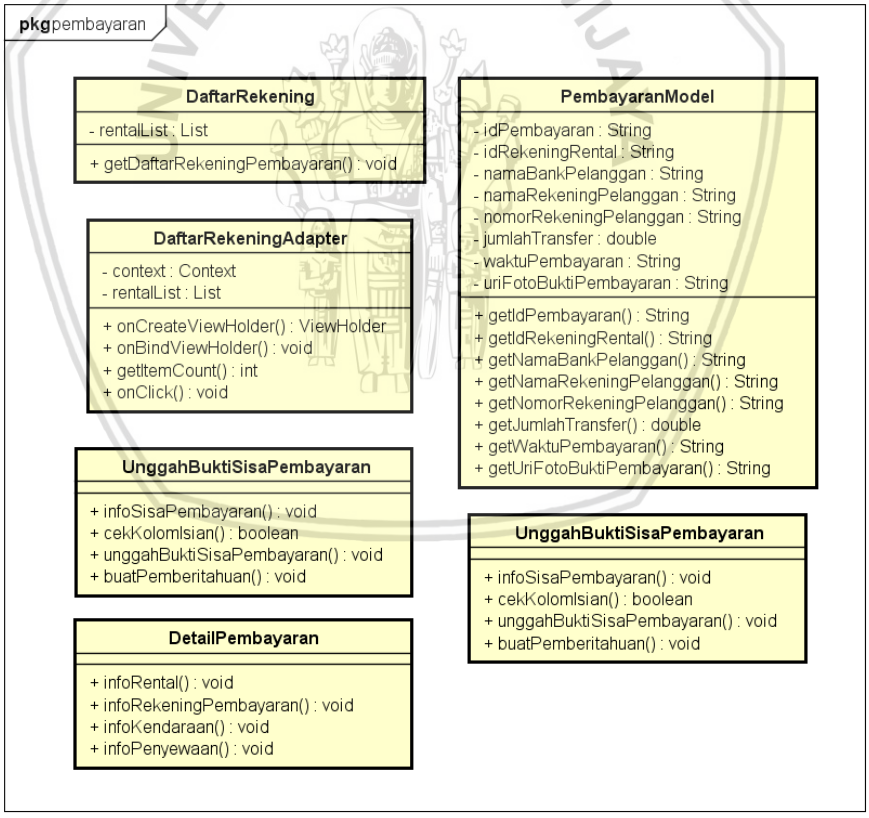


Gambar 5.11 *Class Diagram* Paket Pencarian



Gambar 5.12 *Class Diagram* Paket Penyewaan

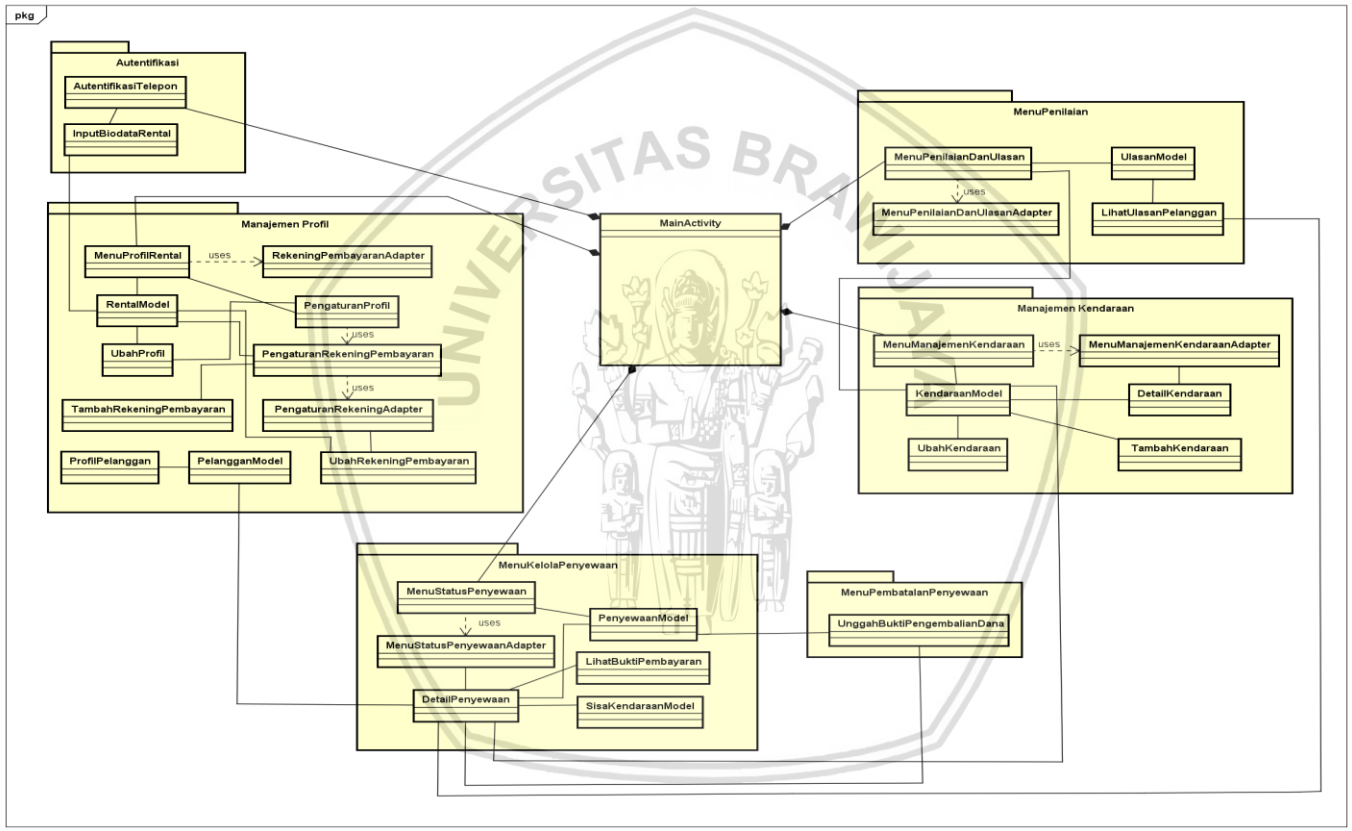
Paket pembayaran terdiri dari enam buah *class* yaitu *class* `DaftarRekening`, `DaftarRekeningAdapter`, `UnggahBuktiPembayaran`, `UnggahBuktiSisaPembayaran`, `DetailPembayaran` dan `PembayaranModel`. *Class* `DaftarRekening` merupakan *class* yang berfungsi untuk menampilkan rekening pembayaran rental kendaraan yang dapat digunakan untuk melakukan transfer pembayaran. Sedangkan *class* `DaftarRekeningAdapter` merupakan suatu *class* adapter yang digunakan oleh *class* `DaftarRekening` untuk menampilkan daftar rekening pembayaran rental. Selanjutnya adalah *class* `UnggahBuktiPembayaran` dan `UnggahBuktiSisaPembayaran` merupakan suatu activity *class* yang berfungsi bagi pelanggan untuk melakukan unggah bukti pembayaran atas penyewaan yang dilakukan. *Class* `DetailPembayaran` berfungsi sebagai activity class untuk menampilkan informasi pembayaran yang harus dilakukan pelanggan atas penyewaan yang dilakukan. *Class* `PembayaranModel` merupakan *class* yang menyediakan operasi *getter* data pembayaran dan juga berfungsi sebagai *entity class.* Atribut dan operasi dari tiga buah *class* tersebut dapat dilihat dalam *class diagram* pada Gambar 5.13.



Gambar 5.13 *Class Diagram* Paket Pembayaran

5.1.5.2 Perancangan *Class Diagram* Pemilik Rental

Pada *class diagram* pemilik rental terbagi menjadi enam buah paket. Berikut ini *class diagram* dari sisi pemilik rental beserta relasi antar *class* yang ditunjukkan pada Gambar 5.14.



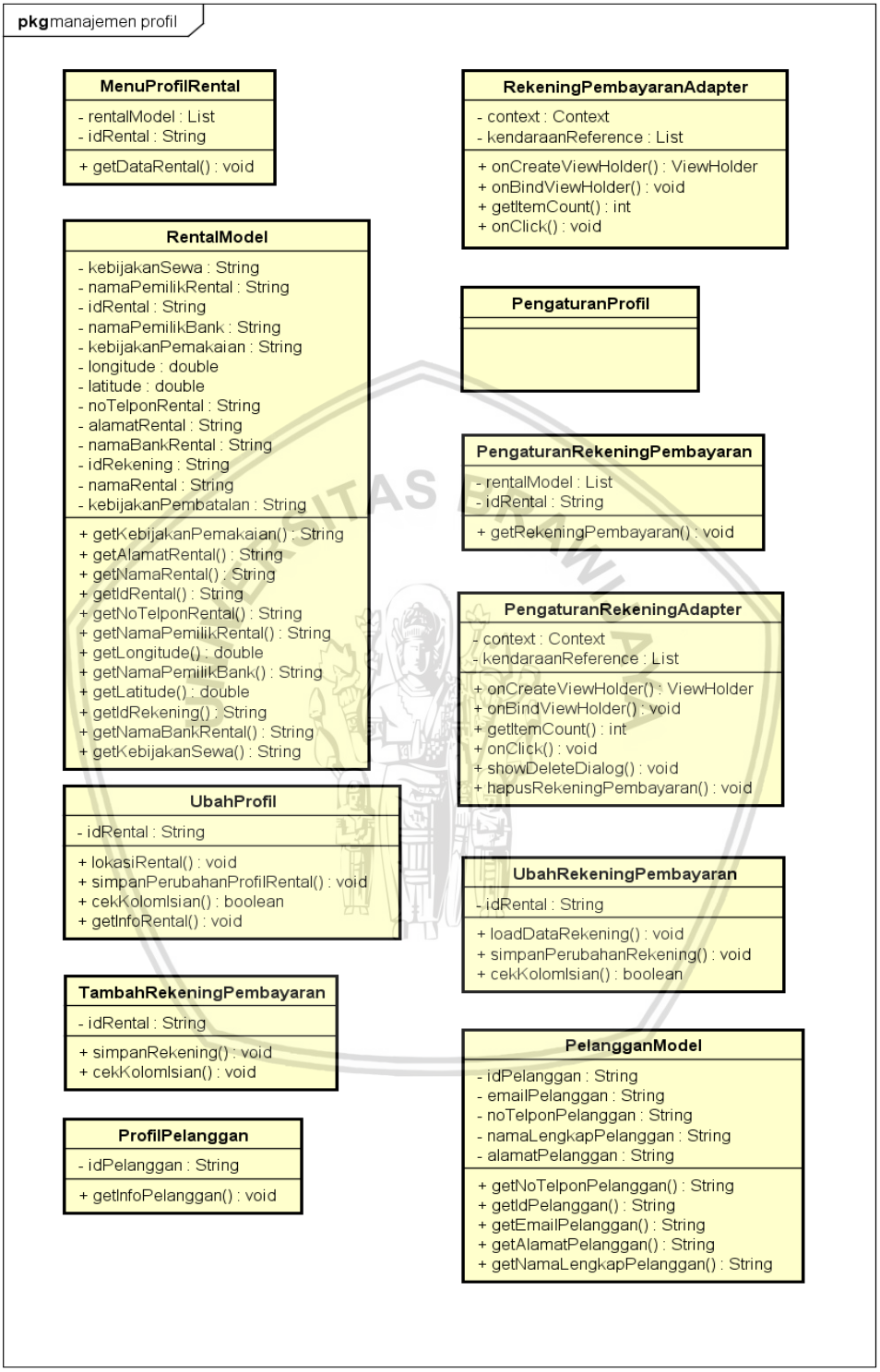
Gambar 5.14 *Class Diagram* Pemilik Rental

Dari lima paket pada *class diagram* pemilik rental akan dijelaskan tiga paket utama beserta atribut dan operasi yang ada di dalam *class* tersebut. Paket utama yang akan di jelaskan adalah `paket manajemen profil`, `paket kelola penyewaan` dan `paket manajemen kendaraan`.

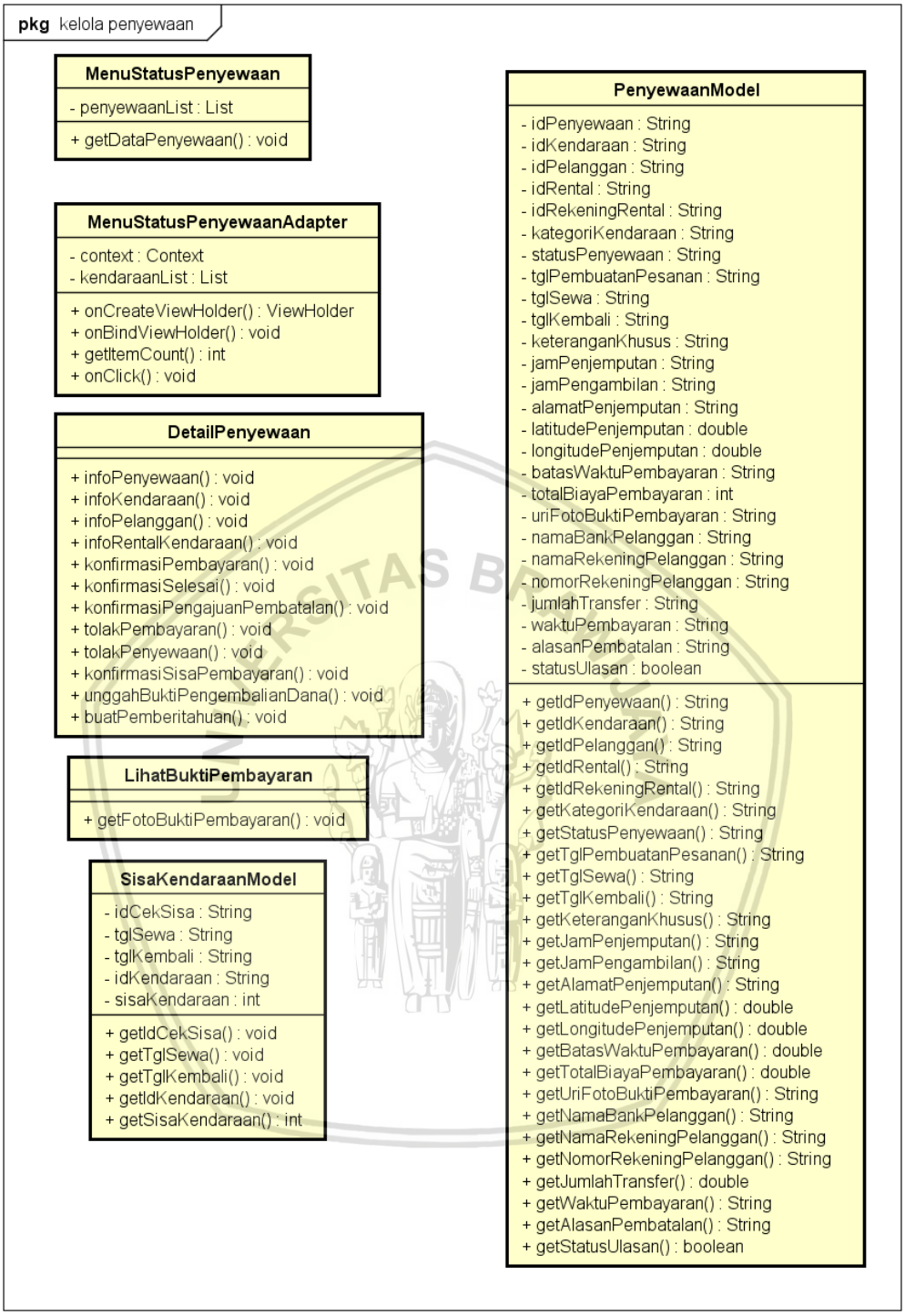
Paket manajemen profil terdiri dari sebelas class yaitu `class MenuProfilRental`, `UbahProfil`, `RentalModel`, `ProfilPelanggan`, `PelangganModel`, `RekeningPembayaranAdapter`, `PengaturanProfil`, `PengaturanRekeningPembayaran`, `PengaturanRekeningAdapter`, `TambahRekeningPembayaran` dan `UbahRekeningPembayaran`. *Class* `MenuProfilRental` merupakan *fragment class* yang berfungsi menampilkan profil rental kendaraan. *C*lass `UbahProfil` merupakan *class* yang berfungsi untuk mengubah profil rental yang sebelumnya telah terdaftar dalam sistem. Lalu *class* `RentalModel` merupakan merupakan suatu *class* yang menyediakan operasi *getter* data kendaraan dan juga berfungsi sebagai *entity class. Class* `ProfilPelanggan` merupakan class yang berfungsi bagi pemilik rental untuk melihat profil pelanggan yang melakukan penyewaan dan *class* `PelangganModel` merupakan *class* menyediakan operasi *getter* data pelanggan. *Class* `RekeningPembayaranAdapter` berfungsi untuk menampilkan daftar rekening pembayaran yang tersimpan dalam sistem. *Class* `PengaturanProfil` berfungsi untuk menampilkan pilihan ubah profil atau ubah rekening pembayaran. *Class* `PengaturanRekeningPembayaran` berfungsi untuk menampilkan daftar rekening pembayaran yang ingin di ubah dengan menggunakan *class* `PengaturanRekeningAdapter`. *Class* `TambahRekeningPembayaran` merupakan *class* yang digunakan untuk menambahkan rekening pembayaran pada sistem. Sedangkan *class* `UbahRekeningPembayaran` merupakan *class* yang berfungsi mengubah data rekening pembayaran yang sebelumnya telah tersimpan dalam sistem. Atribut dan operasi dari lima buah *class* tersebut dapat dilihat dalam *class diagram* pada Gambar 5.15.

Paket kelola penyewaan terdiri dari enam buah class yaitu *class* `MenuStatusPenyewaan`, `MenuStatusPenyewaanAdapter`, `DetailPenyewaan`, `LihatBuktiPembayaran` dan `PenyewaanModel` dan `SisaKendaraanModel`. `Paket kelola penyewaan` berfungsi untuk melakukan pengelolaan terhadap penyewaan dan juga untuk melihat status penyewaan. *Class* `MenuStatusPenyewaan` berfungsi untuk menampilkan daftar penyewaan yang masuk kepada pemilik rental. Untuk menampilkannya *class* `MenuStatusPenyewaan` menggunakan *class* `MenuStatusPenyewaanAdapter`. Sedangkan *class* `DetailPenyewaan` digunakan untuk menampilkan informasi penyewaan dan juga untuk melakukan pengelolaan terhadap penyewaan. Sedangkan *class* `LihatBuktiPembayaran` merupakan *class* yang berfungsi untuk menampilkan foto bukti pembayaran penyewaan. *Class* `PenyewaanModel` dan `SisaKendaraanModel` merupakan *class* yang menyediakan operasi *getter* data penyewaan dan juga

berfungsi sebagai *entity class.* Atribut dan operasi dari lima buah *class* tersebut dapat dilihat dalam *class diagram* pada Gambar 5.16.



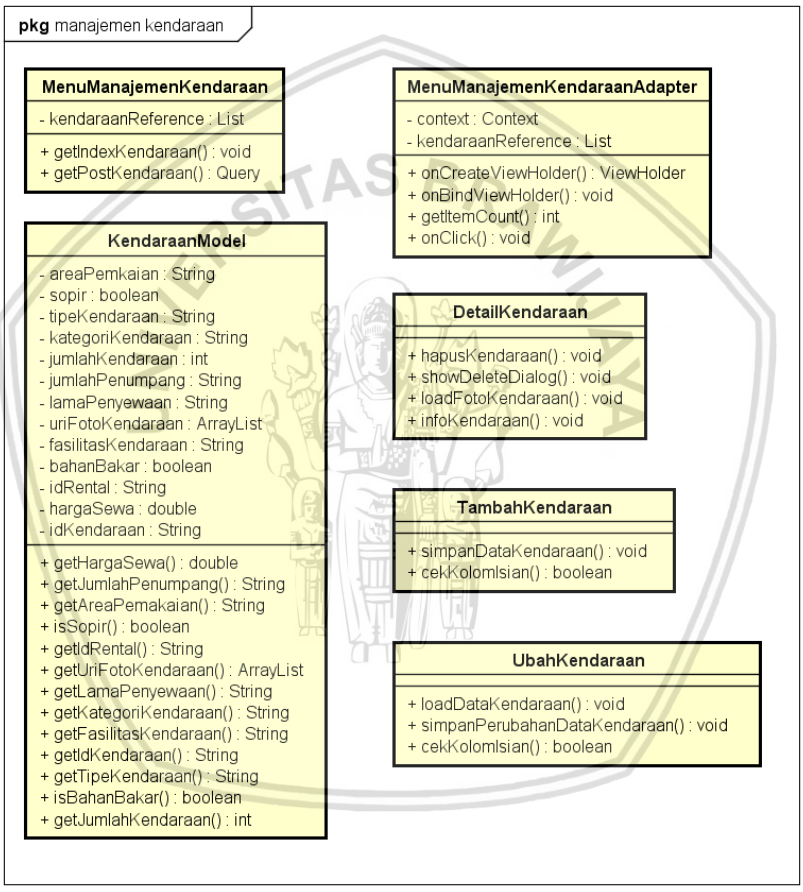
Gambar 5.15 Class Diagram Paket Manajemen Profil



Gambar 5.16 *Diagram Class* Paket Kelola Penyewaan

`Paket manajemen kendaraan` terdiri dari enam buah *class* yaitu *class* `MenuManajemenKendaraan`, `MenuManajemenKendaraanAdapter`, `DetailKendaraan`, `TambahKendaraan`, `UbahKendaraan` dan `KendaraanModel`. `Paket manajemen kendaraan` berfungsi untuk melakukan pengelolaan terhadap data kendaraan rental. *Class* `MenuManajemenKendaraan` merupakan *class* yang berfungsi menampilkan

daftar kendaraan rental yang telah tersimpan dalam sistem. Sedangkan `MenuManajemenKendaraanAdapter` merupakan *class* yang digunakan oleh `MenuManajemenKendaraan` untuk menampilkan daftar kendaraan rental. *Class* `DetailKendaraan` berfungsi untuk menampilkan informasi detail dari kendaraan. Sedangkan *class* `TambahKendaraan` berfungsi bagi pemilik rental untuk dapat menambah dan kendaraan dan *class* `UbahKendaraan` berfungsi untuk mengubah data kendaraan yang sebelumnya telah tersimpan dalam sistem. *Class* `KendaraanModel` merupakan suatu *class* yang berfungsi menyediakan operasi *getter* data kendaraan dan juga berfungsi sebagai *entity class.* Atribut dan operasi dari enam buah *class* tersebut dapat dilihat dalam *class diagram* pada Gambar 5.17.



Gambar 5.17 *Class Diagram* Paket Manajemen Kendaraan

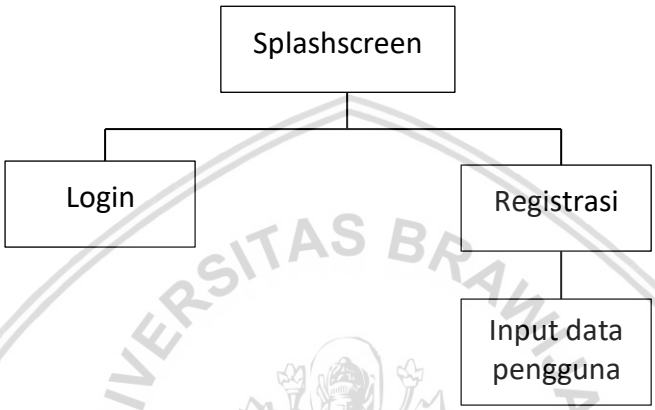
5.1.6 Perancangan Antarmuka

Bagian ini menjelaskan tentang perancangan antarmuka dari sistem penyewaan kendaraan Kota Malang. Perancangan antarmuka berisi tentang tampilan aplikasi yang akan digunakan oleh calon pengguna aplikasi. *Screenflow* digunakan untuk menggambarkan keseluruhan antarmuka sistem yang sesuai dengan kebutuhan aktor yang terbagi menjadi pengguna, pelanggan dan pemilik rental. Pada metode pengembangan MASAM, terdapat suatu *task UI Design* yang dimana pada penelitian ini akan dijelaskan pada bagian perancangan antarmuka.

Maka dengan itu, pada bagian ini akan menggambarkan kebutuhan-kebutuhan fungsional yang akan di implementasikan kedalam sistem sesuai dengan spesifikasi kebutuhan pengguna. Sesuai dengan iterasi pada penelitian ini maka terdapat 2 iterasi pada bagian perancangan antarmuka.

5.1.6.1 Perancangan Antarmuka pada sisi pengguna

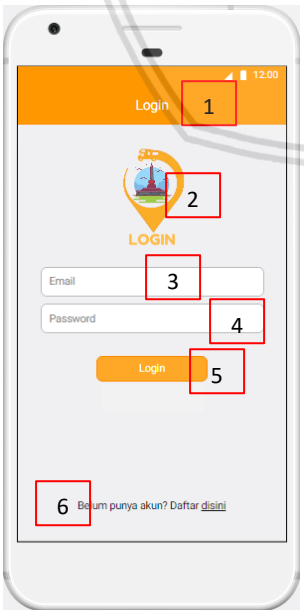
Antarmuka pada sisi pengguna berupa halaman yang disediakan bagi pengguna untuk dapat masuk kedalam sistem. Antarmuka untuk sisi pengguna terdiri dar antarmuka halaman *splashscreen*, halaman *login* dan halaman registrasi. *Screenflow* aplikasi pada sisi pengguna ditunjukkan oleh Gambar 5.18.



Gambar 5.18 *Screenflow* menu sisi pengguna

a. Halaman *Login*

Halaman *login* merupakan salah satu antarmuka pengguna yang berfungsi untuk masuk atau *login* kedalam sistem. Antarmuka login ditunjukkan oleh Gambar 5.19 dibawah ini.



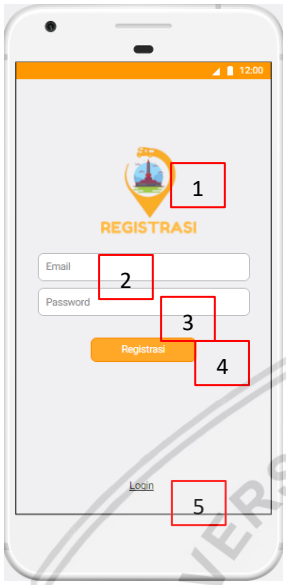
Keterangan :

1. *Text View* judul halaman.
2. *Image View* untuk menampilkan logo aplikasi.
3. *Edit text* untuk mengisi email.
4. *Edit text* untuk mengisi *password.*
5. Tombol Login untuk menjalankan proses login kedalam sistem.
6. *Link* untuk mengarahkan pengguna ke halaman registrasi

Gambar 5.19 Tampilan Antarmuka Halaman Login

b. Halaman Registrasi

Halaman registrasi merupakan salah satu antarmuka pengguna yang berfungsi bagi pengguna untuk mendaftar sebagai pelanggan pada aplikasi. Tampilan antarmuka halaman registrasi ditunjukkan oleh Gambar 5.20.



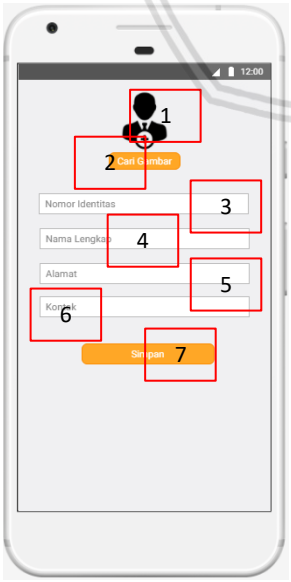
Keterangan :

1. *Imageview* untuk menampilkan logo aplikasi.
2. *Edit text* untuk mengisi email.
3. *Edit text* untuk mengisi *password.*
4. Tombol Registrasi untuk menjalankan proses registrasi kedalam sistem.
5. *Link* untuk mengarahkan pengguna ke halaman login.

Gambar 5.20 Halaman Registrasi

c. Halaman Input Data Pengguna

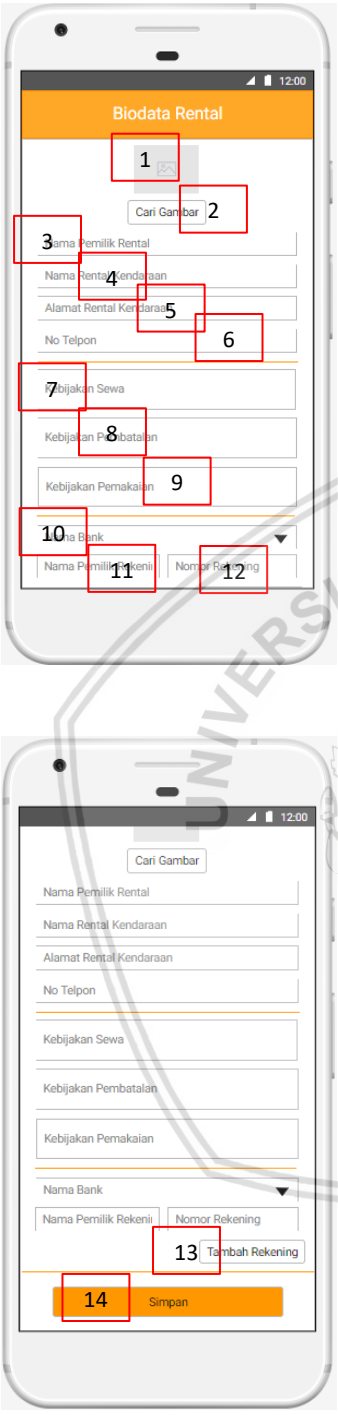
Halaman input data pengguna terbagi menjadi dua buah bagian, yang pertama digunakan pada aplikasi sisi pelanggan dan yang kedua digunakan pada aplikasi sisi pemilik rental. Untuk input biodata pengguna pada aplikasi pelanggan ditunjukkan oleh Gambar 5.21, sedangkan input biodata pengguna pada aplikasi pemilik rental ditunjukkan oleh Gambar 5.22.



Keterangan :

1. *Imageview* untuk menampilkan gambar yang dipilih pengguna.
2. Tombol cari gambar untuk menjalankan proses cari gambar.
3. *Edit text* untuk mengisi nomor identitas.
4. *Edit text* untuk mengisi nama lengkap.
5. *Edit text* untuk mengisi alamat.
6. *Edit text* untuk mengisi kontak.
7. Tombol simpan untuk menyimpan data pengguna kedalam sistem.

Gambar 5.21 Halaman Input Data Pengguna Sisi Pelanggan



Keterangan :

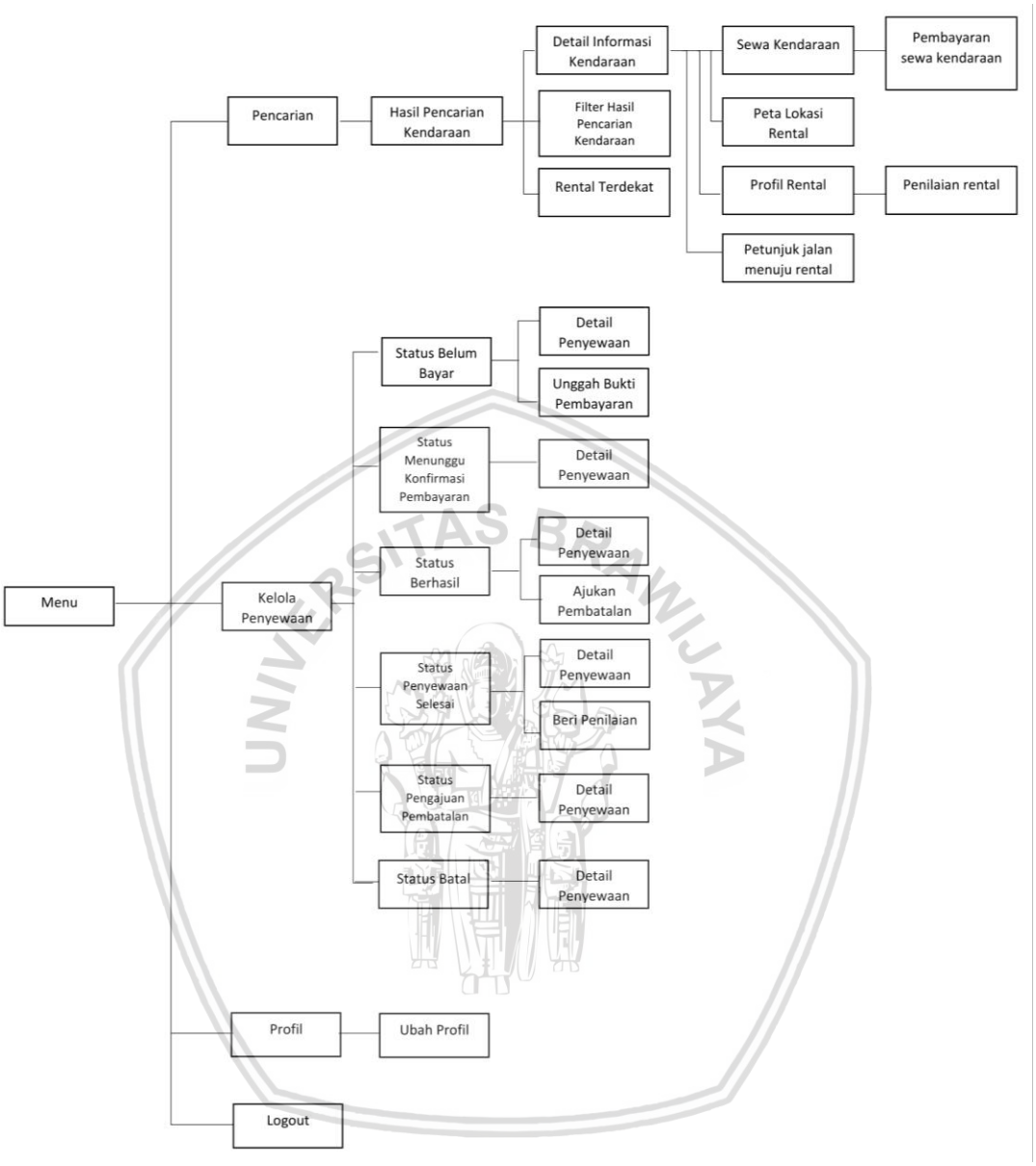
8. *Imageview* untuk menampilkan gambar yang dipilih pengguna.
9. Tombol cari gambar untuk menjalankan proses cari gambar.
10. *Edit text* untuk mengisi nama pemilik rental
11. *Edit text* untuk mengisi nama rental.
12. *Edit text* untuk mengisi alamat.
13. *Edit text* untuk mengisi kontak.
14. *Edit text* untuk mengisi kebijakan sewa rental.
15. *Edit text* untuk mengisi kebijakan pembatalan rental.
16. *Edit text* untuk mengisi kebijakan pemakaian rental.
17. *Spinner* untuk memilih nama bank.
18. *Edit text* untuk mengisi nama pemilik rekening.
19. *Edit text* untuk mengisi nomor rekening.
20. Tombol untuk menambahkan rekening pembayaran.
21. Tombol simpan untuk menyimpan data rental kedalam sistem.

Gambar 5.22 Halaman Input Data Pengguna Sisi Pemilik Rental

5.1.6.2 Perancangan Antarmuka pada sisi pelanggan

Antarmuka pada sisi pelanggan terdiri dari beberapa halaman yaitu, halaman pencarian, halaman kelola penyewaan, halaman pemberitahuan, halaman profil pelanggan dan logout. Didalam halaman tersebut terdapat beberapa halaman

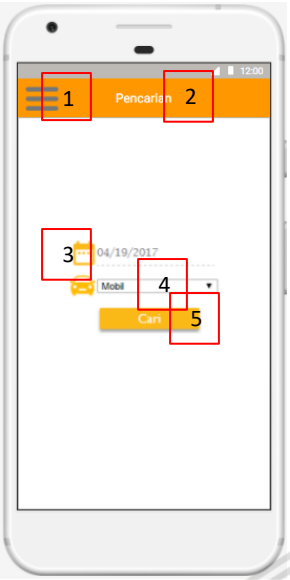
yang saling berhubungan. *Screenflow* aplikasi pada sisi pelanggan ditunjukkan oleh Gambar 5.23.



Gambar 5.23 *Screenflow* Menu Pada Sisi Pelanggan

a. Halaman Pencarian

Halaman pencarian merupakan salah satu antarmuka pelanggan yang berfungsi untuk melakukan pencarian kendaraan sesuai dengan keinginan pelanggan. Pelanggan dapat melakukan pencarian kendaraan yang tersedian berdasarkan tanggal penyewaan dan kategori kendaraan yang diinginkan pengguna. Tampilan antarmuka halaman pencarian ditunjukkan oleh Gambar 5.24.

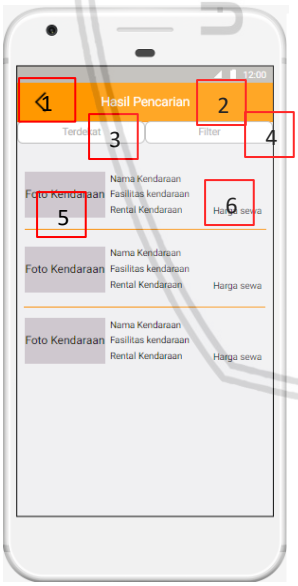


Keterangan :

1. *Menu drawer* untuk menampilkan menu utama.
2. *Text View* judul halaman.
3. *Date picker* untuk memilih tanggal penyewaan.
4. *Dropdown menu* untuk memilih kategori kendaraan.
5. Tombol Cari untuk menjalankan proses pencarian kendaraan.

Gambar 5.24 Tampilan Antarmuka Halaman Pencarian

Halaman hasil pencarian merupakan salah satu antarmuka pelanggan yang berfungsi menampilkan hasil pencarian kendaraan yang telah dilakukan oleh pengguna. Tampilan antarmuka halaman hasil pencarian ditunjukkan oleh Gambar 5.25.



Keterangan :

1. Tombol kembali untuk kembali ke halaman sebelumnya.
2. *Text View* judul halaman.
3. Tombol ganti tanggal untuk mengarahkan pelanggan ke menu pencarian.
4. Tombol terdekat untuk melakukan pencarian kendaraan berdasarkan rental terdekat.
5. Tombol filter untuk melakukan penyaringan atas pencarian kendaraan.
6. *Image View* untuk menampilkan foto kendaraan
7. *Text View* untuk menampilkan informasi kendaraan

Gambar 5.25 Tampilan Antarmuka Halaman Hasil Pencarian

Halaman hasil pencarian terdekat merupakan salah satu antarmuka pelanggan yang berfungsi untuk menampilkan hasil pencarian berdasarkan rental kendaraan dengan lokasi terdekat dengan posisi pelanggan. Tampilan antarmuka hasil pencarian terdekat ditunjukkan oleh Gambar 5.26.

Keterangan :

1. Tombol kembali untuk kembali ke halaman sebelumnya.
2. *Text View* judul halaman.
3. Tombol terdekat untuk melakukan pencarian kendaraan berdasarkan rental terdekat.
4. Tombol filter untuk melakukan penyaringan atas pencarian kendaraan.
5. *Image View* untuk menampilkan foto kendaraan
6. *Text View* untuk menampilkan informasi kendaraan
7. *Text View* untuk menampilkan jarak antara rental terdekat dengan lokasi pelanggan

Gambar 5.26 Tampilan Antarmuka Halaman Hasil Pencarian Terdekat

Halaman detail kebijakan sewa merupakan salah satu halaman yang termasuk kedalam menu detail kendaraan. Halaman detail kebijakan sewa berfungsi untuk memberikan informasi kepada pelanggan tentang kebijakan penyewaan yang berlaku. Tampilan antarmuka halaman detail kebijakan sewa ditunjukkan oleh Gambar 5.27.

Keterangan :

1. Tombol kembali untuk kembali ke halaman sebelumnya.
2. *Text View* judul halaman.
3. *Tab menu* Kebijakan Sewa untuk menampilkan halaman kebijakan sewa.
4. *Tab menu* Detail Kendaraan untuk menampilkan halaman detail kendaraan.
5. *Image View* untuk menampilkan foto kendaraan.
6. *Text View* untuk menampilkan informasi kendaraan.
7. *Text View* untuk menampilkan informasi kebijakan sewa.
8. Tombol Pesan untuk menjalankan proses penyewaan kendaraan.

Gambar 5.27 Tampilan Antarmuka Halaman Detail Kebijakan Sewa

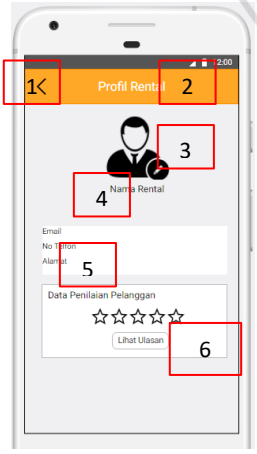
Sedangkan Gambar 5.28 akan menampilkan informasi kendaraan yang tersedia sesuai dengan kendaraan yang dipilih pelanggan dari halaman hasil pencarian kendaraan yang tersedia.

Keterangan :

1. Tombol kembali untuk kembali ke halaman sebelumnya.
2. *Text View* judul halaman
3. *Tab menu* Kebijakan Sewa untuk menampilkan halaman kebijakan sewa.
4. *Tab menu* Detail Kendaraan untuk menampilkan halaman detail kendaraan.
5. *Image View* untuk menampilkan foto kendaraan.
6. *Text View* untuk menampilkan informasi kendaraan.
7. *Text View* untuk menampilkan informasi rental kendaraan
8. Tombol Lihat Profil Rental untuk menampilkan halaman profil rental kendaraan.
9. Peta untuk menampilkan lokasi rental.
10. Tombol petunjuk jalan untuk menjalankan proses dan menampilkan halaman petunjuk jalan menuju rental.
11. Tombol Pesan untuk menjalankan proses penyewaan kendaraan.

Gambar 5.28 Tampilan Antarmuka Halaman Detail Kendaraan

Halaman antarmuka profil rental merupakan salah satu antarmuka pada sisi pelanggan yang berfungsi untuk mengatahui informasi dari rental. Dalam halaman ini, pelanggan juga dapat melihat penilaian dari rental kendaraan. Tampilan antarmuka halaman detail kendaraan ditunjukkan pada Gambar 5.29.

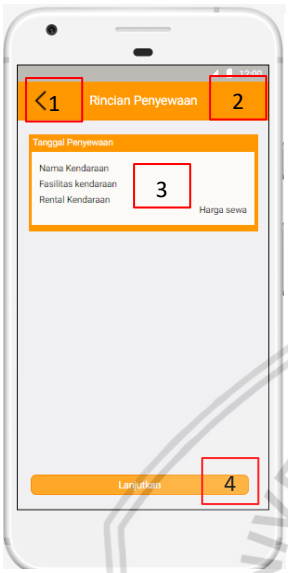


Keterangan :

1. Tombol kembali untuk mengarahkan ke halaman sebelumnya.
2. *Text View* judul halaman.
3. *Image View* untuk menampilkan gambar atau foto profil rental.
4. *Text View* untuk menampilkan nama rental.
5. *Text View* untuk menampilkan informasi rental.
6. Text View untuk menampilkan penilaian rental.

Gambar 5.29 Tampilan Antarmuka Halaman Profil Rental

Halaman antarmuka rincian penyewaan merupakan halaman yang berfungsi untuk memberikan rincian informasi mengenai penyewaan yang dilakukan oleh pelanggan. Informasi yang ditampilkan mengacu pada kendaraan yang dipilih pelanggan sesuai dengan tanggal penyewaan dan kategori yang diinginkan pelanggan. Tampilan antarmuka halaman rincian penyewaan ditampilkan pada Gambar 5.30.



Keterangan :

1. Tombol kembali untuk mengarahkan ke halaman sebelumnya.
2. *Text View* judul halaman.
3. *Text View* untuk menampilkan informasi rincian penyewaan.
4. Tombol Lanjutkan untuk mengarahkan pelanggan kehalaman proses penyewaan selanjutnya.

Gambar 5.30 Tampilan Antarmuka Halaman Rincian Penyewaan

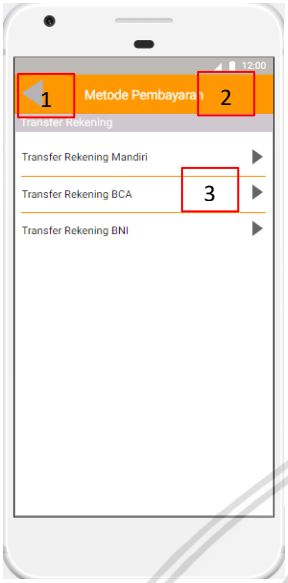
Halaman antarmuka input informasi penyewaan merupakan halaman yang berfungsi bagi pelanggan untuk melengkapi informasi tambahan mengenai penyewaan yang dibuat. Tampilan antarmuka halaman input informasi penyewaan ditunjukkan oleh Gambar 5.31.

Keterangan :

1. Tombol kembali untuk mengarahkan ke halaman sebelumnya.
2. *Text View* judul halaman.
3. *Text View* untuk menampilkan informasi biodata pemesan.
4. *Text Input* untuk mengisi keterangan waktu penjemputan/pengambilan kendaraan.
5. *Text Input* untuk mengisi keterangan lokasi penjemputan
6. Tombol Lanjutkan untuk mengarahkan pelanggan kehalaman proses penyewaan selanjutnya.

Gambar 5.31 Tampilan Antarmuka Halaman Input Informasi Penyewaan

Halaman antarmuka pilih metode pembayaran merupakan halaman yang berfungsi bagi pelanggan untuk memilih pembayaran yang akan digunakan. Tampilan antarmuka pilih metode pembayaran ditunjukkan oleh Gambar 5.32.



Keterangan :

1. Tombol kembali untuk mengarahkan ke halaman sebelumnya.
2. *Text View* judul halaman.
3. *List View* untuk menampilkan pilihan kategori kendaraan yang akan ditambahkan.

Gambar 5.32 Tampilan Antarmuka Halaman Pilih Metode Pembayaran

Halaman antarmuka pembayaran merupakan halaman yang berfungsi untuk memberikan informasi terhadap pelanggan tentang pembayaran atas penyewaannya. Tampilan antarmuka pembayaran ditunjukkan oleh Gambar 5.33.

Keterangan :

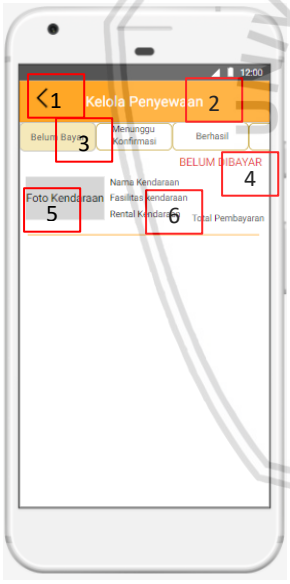
1. Tombol kembali untuk mengarahkan ke halaman sebelumnya.
2. *Text View* judul halaman.
3. *Text View* untuk menampilkan informasi pembayaran.
4. *Text View* menampilkan informasi batas waktu pembayaran.
5. *Text View* untuk menampilkan rincian penyewaan.
6. Tombol Unggah Bukti Pembayaran Sekarang untuk mengarahkan ke halaman unggah bukti pembayaran.
7. Tombol Unggah Bukti Pembayaran Nanti untuk mengarahkan ke halaman kelola penyewaan.

Gambar 5.33 Tampilan Antarmuka Halaman Pembayaran

b. Halaman Kelola Penyewaan

Halaman kelola penyewaan merupakan salah satu antarmuka yang berfungsi bagi pelanggan untuk melihat status penyewaan dan melakukan pengelolaan terhadap penyewaan yang telah dilakukan. Pengelolaan tersebut meliputi melihat detail penyewaan, melakukan unggah bukti pembayaran, melihat bukti pembayaran, melakukan pengajuan pembatalan dan memberikan penilaian. Semua fungsi pengelolaan tersebut dapat dilakukan tergantung pada status penyewaan pelanggan. Namun pada bagian ini hanya akan di tampilkan rancangan antarmuka untuk tiga status yaitu status belum bayar, menunggu konfirmasi pembayaran dan berhasil. Hal ini dikarenakan seluruh status mempunyai tampilan yang secara keseluruhan sama tetapi mempunya perbedaan pada halaman detail yang akan menampilkan tombol konfirmasi yang berbeda-beda. Status yang tidak ditampilkan yaitu status selesai yang dimana pada halaman detailnya terdapat tombol untuk memberikan penilaian, lalu status pengajuan pembatalan dan batal mempunyai tampilan detail yang sama namu status yang ditampilkan berbeda.

Halaman status belum bayar merupakan halaman yang memuat semua penyewaan pelanggan yang belum dibayar. Tampilan antarmuka halaman status belum bayar ditunjukkan oleh Gambar 5.34.



Keterangan :

1. Tombol kembali untuk mengarahkan ke halaman sebelumnya.
2. *Text View* judul halaman.
3. *Tab Menu* status penyewaan untuk menampilan tab status penyewaan yang sedang aktif.
4. *Text View* status penyewaan untuk menampilkan status penyewaan
5. *Image View* untuk menampilkan foto kendaraan yang dipesan.
6. *Text View* untuk menampilkan informasi kendaraan dan informasi penyewaan.

Gambar 5.34 Tampilan Antarmuka Halaman Status Belum Bayar

Pada halaman status belum bayar terdapat halaman detail penyewaan yang memuat informasi mengenai penyewaan yang dilakukan oleh pelanggan. Tampilan antarmuka halaman detail penyewaan dengan status belum bayar ditunjukkan oleh Gambar 5.35.

Keterangan :

1. Tombol kembali untuk mengarahkan ke halaman sebelumnya.
2. *Text View* judul halaman.
3. *Text View* status penyewaan untuk menampilkan status penyewaan.
4. *Text View* informasi penyewaan dan biodata pelanggan untuk memberikan informasi penyewaan kepada pelanggan.
5. Tombol bayar sekarang untuk mengarahkan ke halaman unggah bukti pembayaran.

Gambar 5.35 Tampilan Antarmuka Halaman Detail Penyewaan Dengan Status Belum Bayar

Pada halaman status belum bayar terdapat halaman unggah bukti pembayaran yang berfungsi memfasilitasi pelanggan untuk mengunggah bukti pembayaran atas penyewaannya. Tampilan antarmuka halaman unggah bukti pembayaran ditunjukkan oleh Gambar 5.36.

Keterangan :

1. Tombol kembali untuk mengarahkan ke halaman sebelumnya.
2. *Text View* judul halaman.
3. *Text View* untuk menampilkan informasi transfer pembayaran.
4. *Text View* untuk menampilkan informasi batas waktu pembayaran.
5. *Text View* untuk menampilkan total pembayaran penyewaan.
6. Tombol untuk melakukan pencarian gambar.
7. *Edit text* untuk mengisi informasi pembayaran.
8. Tombol untuk melakukan unggah bukti pembayaran kedalam sistem.

Gambar 5.36 Tampilan Antarmuka Halaman Unggah Bukti Pembayaran

Halaman status menunggu konfirmasi pembayaran merupakan halaman yang memuat semua penyewaan pelanggan dengan status menunggu konfirmasi pembayaran dari rental kendaraan. Tampilan antarmuka halaman status menunggu konfirmasi pembayaran ditunjukkan oleh Gambar 5.37.

Keterangan :

1. Tombol kembali untuk mengarahkan ke halaman sebelumnya.
2. *Text View* judul halaman.
3. *Tab Menu* status penyewaan untuk menampilan tab status penyewaan yang sedang aktif.
4. *Text View* status penyewaan untuk menampilkan status penyewaan
5. *Image View* untuk menampilkan foto kendaraan yang dipesan.
6. *Text View* untuk menampilkan informasi kendaraan dan informasi penyewaan.

Gambar 5.37 Tampilan Antarmuka Halaman Status Menunggu Konfirmasi Pembayaran

Pada halaman status menunggu konfirmasi pembayaran terdapat halaman detail penyewaan yang memuat informasi mengenai penyewaan yang dilakukan oleh pelanggan. Tampilan antarmuka halaman detail penyewaan dengan status menunggu konfirmasi pembayaran ditunjukkan oleh Gambar 5.38.

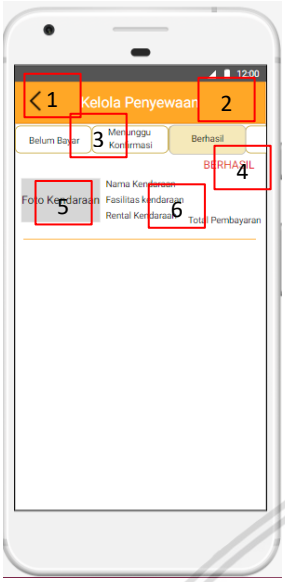
Keterangan :

1. Tombol kembali untuk mengarahkan ke halaman sebelumnya.
2. *Text View* judul halaman.
3. *Text View* status penyewaan untuk menampilkan status penyewaan.
4. *Text View* informasi penyewaan dan biodata pelanggan untuk memberikan informasi penyewaan kepada pelanggan.

Gambar 5.38 Tampilan Antarmuka Halaman Detail Penyewaan Dengan Status Menunggu Konfirmasi Pembayaran

Halaman status berhasil merupakan halaman yang memuat semua penyewaan pelanggan dengan pembayaran yang telah berhasil dan telah di

konfirmasi rental. Tampilan antarmuka halaman status berhasil ditunjukkan oleh Gambar 5.39.



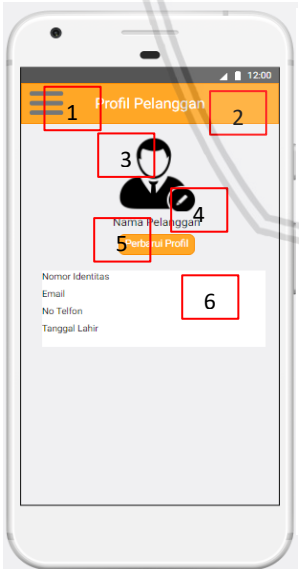
Keterangan :

1. Tombol kembali untuk mengarahkan ke halaman sebelumnya.
2. *Text View* judul halaman.
3. *Tab Menu* status penyewaan untuk menampilan tab status penyewaan yang sedang aktif.
4. *Text View* status penyewaan untuk menampilkan status penyewaan
5. *Image View* untuk menampilkan foto kendaraan yang dipesan.
6. *Text View* untuk menampilkan informasi kendaraan dan informasi penyewaan.

Gambar 5.39 Tampilan Antarmuka Halaman Status Berhasil

c. Halaman Profil

Halaman Profil merupakan salah satu antarmuka yang berfungsi untuk menampilkan profil pelanggan dan menyediakan antarmuka untuk melakukan perubahan profil pelanggan. Tampilan antarmuka halaman profil pelanggan ditunjukkan oleh Gambar 5.40.

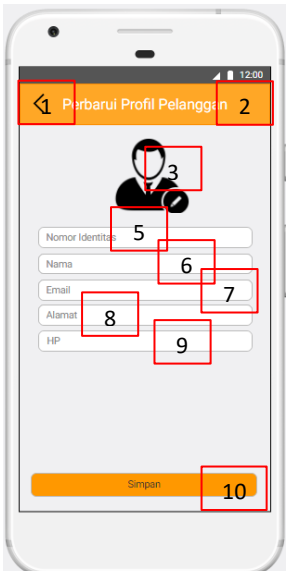


Keterangan :

1. Tombol kembali untuk mengarahkan ke halaman sebelumnya.
2. *Text View* judul halaman.
3. *Image View* untuk menampilkan foto profil pelanggan.
4. *Text View* untuk menampilkan nama pelanggan.
5. Tombol Ubah profil untuk mengarahkan ke halaman Ubah profil.
6. *Text View* untuk menampilkan informasi biodata pelanggan.

Gambar 5.40 Tampilan Antarmuka Profil

Tampilan antarmuka halaman ubah profil pelanggan ditunjukkan oleh Gambar 5.41.



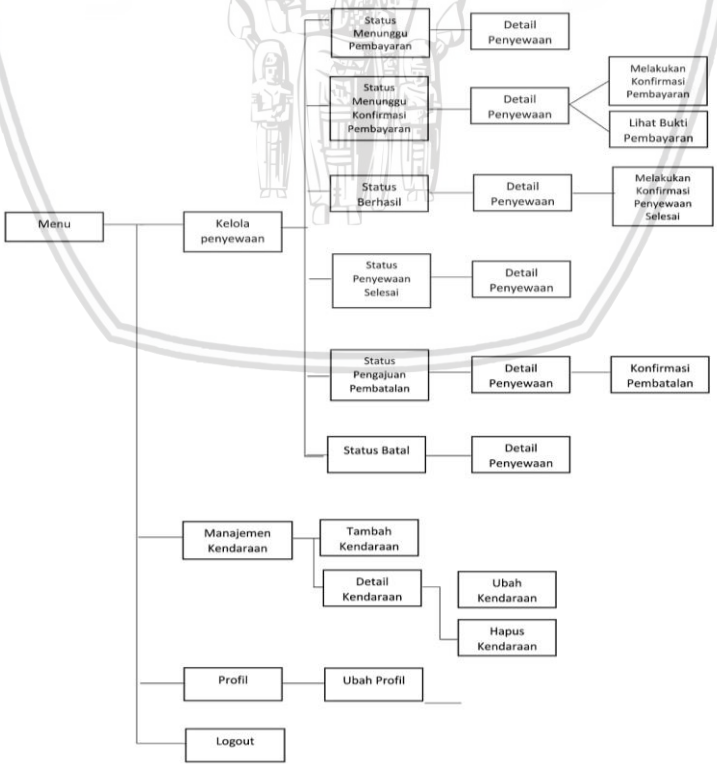
Keterangan :

1. Tombol kembali untuk mengarahkan ke halaman sebelumnya.
2. *Text View* judul halaman.
3. *Image Button* untuk melakukan pencarian gambar.
4. *Edit text* untuk mengubah nomor identitas.
5. *Edit text* untuk mengubah nama.
6. *Edit text* untuk mengubah email.
7. *Edit text* untuk mengubah alamat.
8. *Edit text* untuk mengubah nomor kontak.
9. Tombol untuk menyimpan perubahan profil.

Gambar 5.41 Tampilan Antarmuka Halaman Ubah Profil

5.1.6.3 Perancangan Antarmuka pada sisi pemilik rental

Antarmuka pada sisi pemilik rental terdiri dari beberapa halaman yaitu, halaman pemberitahuan, halaman penyewaanku, halaman manajemen data kendaraan, halaman profil dan logout. *Screenflow* aplikasi pada sisi pemilik rental ditunjukkan oleh Gambar 5.42.

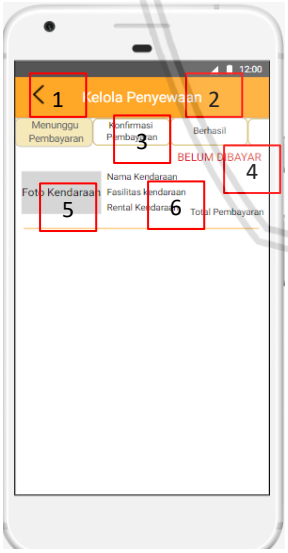


Gambar 5.42 *Screenflow* menu pada sisi pemilik rental

a. Menu Kelola Penyewaan

Halaman kelola penyewaan merupakan salah satu antarmuka yang berfungsi bagi pemilik rental untuk melihat status penyewaan dan melakukan pengelolaan terhadap penyewaan yang masuk. Pengelolaan tersebut meliputi melihat detail penyewaan, melakukan konfirmasi pembayaran, melihat bukti pembayaran, melakukan konfirmasi pembatalan, melihat penilaian dan melakukan unggah bukti pengembalian dana ketika dilakukan pembatalan. Semua fungsi pengelolaan tersebut dapat dilakukan tergantung pada status penyewaan pelanggan. Pada bagian perancangan antarmuka ini hanya tiga status yang di tunjukkan yaitu status menunggu pembayaran, menunggu konfirmasi pembayaran, dan status berhasil. Hal ini dikarenakan tampilan dari seluruh status penyewaan mempunyai tampilan yang hampir sama hanya berbeda pada halaman detail saja. Pada halaman detail penyewaan dengan status selesai akan terdapat tombol untuk melihat penilaian yang telah diberikan pelanggan terhadap rental. Sedangkan pada halaman detail pengajuan pembatalan terdapat tombol untuk melakukan konfirmasi pengajuan pembatalan dan tombol tersebut akan mengarahkan ke halaman unggah bukti pengembalian dana pembayaran. Kemudian tampilan detail status batal mempunyai tampilan yang sama dengan lainnya tanpa ada tombol konfirmasi apapun.

Menu kelola penyewaan bertujuan untuk melihat tentang status penyewaan terhadap pemilik rental. Halaman status menunggu pembayaran merupakan halaman yang memuat semua penyewaan pelanggan terhadap rental yang belum dibayar oleh pelanggan. Tampilan antarmuka halaman status belum bayar ditunjukkan oleh Gambar 5.43.



Keterangan :

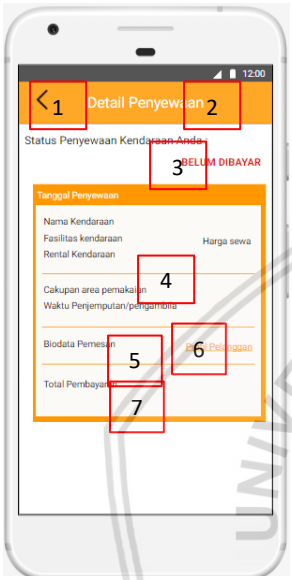
1. Tombol kembali untuk mengarahkan ke halaman sebelumnya.
2. *Text View* judul halaman.
3. *Tab Menu* status penyewaan untuk menampilan tab status penyewaan yang sedang aktif.
4. *Text View* status penyewaan untuk menampilkan status penyewaan
5. *Image View* untuk menampilkan foto kendaraan yang dipesan.
6. *Text View* untuk menampilkan informasi kendaraan dan informasi penyewaan.

Gambar 5.43 Tampilan Antarmuka Halaman Status Menunggu Pembayaran

Pada halaman status menunggu pembayaran terdapat halaman detail penyewaan yang memuat informasi mengenai penyewaan yang dilakukan oleh pelanggan kepada rental. Tampilan antarmuka halaman detail penyewaan dengan status menunggu pembayaran ditunjukkan oleh Gambar 5.44.

Halaman status konfirmasi pembayaran merupakan halaman yang memuat penyewaan yang membutuhkan aksi dari rental untuk mengkonfirmasi pembayaran yang telah dilakukan pelanggan. Tampilan antarmuka halaman status menunggu konfirmasi pembayaran ditunjukkan oleh Gambar 5.45.

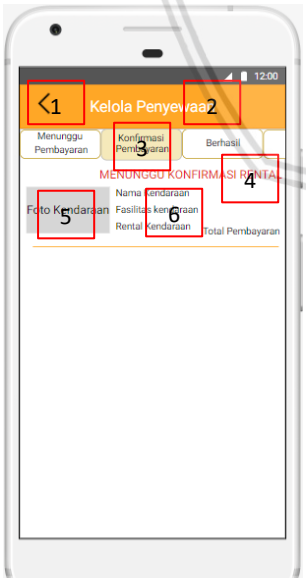
Pada halaman status konfirmasi pembayaran terdapat halaman detail penyewaan yang memuat informasi mengenai penyewaan yang dilakukan oleh pelanggan kepada rental. Tampilan antarmuka halaman detail penyewaan dengan status konfirmasi pembayaran ditunjukkan oleh Gambar 5.46.



Keterangan :

1. Tombol kembali untuk mengarahkan ke halaman sebelumnya.
2. *Text View* judul halaman.
3. *Text View* status penyewaan untuk menampilkan status penyewaan.
4. *Text View* informasi kendaraan yang dipesan.
5. *Text View* informasi biodata pelanggan yang melakukan penyewaan.
6. *Link* untuk mengarahkan ke halaman profil pelanggan.
7. *Text View* menampilkan informasi total pembayaran penyewaan.

Gambar 5.44 Tampilan Antarmuka Halaman Detail Penyewaan dengan Status Menunggu Pembayaran



Keterangan :

1. Tombol kembali untuk mengarahkan ke halaman sebelumnya.
2. *Text View* judul halaman.
3. *Tab Menu* status penyewaan untuk menampilan tab status penyewaan yang sedang aktif.
4. *Text View* status penyewaan untuk menampilkan status penyewaan
5. *Image View* untuk menampilkan foto kendaraan yang dipesan.
6. *Text View* untuk menampilkan informasi kendaraan dan informasi penyewaan.

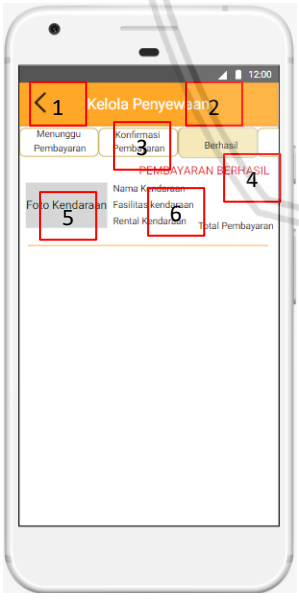
Gambar 5.45 Tampilan Antarmuka Halaman Status Konfirmasi Pembayaran

Keterangan :

1. Tombol kembali untuk mengarahkan ke halaman sebelumnya.
2. *Text View* judul halaman.
3. *Text View* status penyewaan untuk menampilkan status penyewaan.
4. *Text View* informasi kendaraan yang dipesan.
5. *Text View* informasi biodata pelanggan yang melakukan penyewaan.
6. *Link* untuk mengarahkan ke halaman profil pelanggan.
7. *Text View* menampilkan informasi total pembayaran penyewaan.
8. *Link* untuk mengarahkan ke halaman lihat foto bukti pembayaran.
9. Tombol untuk melkukan konfirmasi pembayaran

Gambar 5.46 Tampilan Antarmuka Halaman Detail Penyewaan dengan Status Konfirmasi Pembayaran

Halaman status berhasil merupakan halaman yang memuat penyewaan yang sudah berhasil di konfirmasi oleh rental. Tampilan antarmuka halaman status berhasil ditunjukkan oleh Gambar 5.47.



Keterangan :

1. Tombol kembali untuk mengarahkan ke halaman sebelumnya.
2. *Text View* judul halaman.
3. *Tab Menu* status penyewaan untuk menampilan tab status penyewaan yang sedang aktif.
4. *Text View* status penyewaan untuk menampilkan status penyewaan
5. *Image View* untuk menampilkan foto kendaraan yang dipesan.
6. *Text View* untuk menampilkan informasi kendaraan dan informasi penyewaan.

Gambar 5.47 Tampilan Antarmuka Halaman Status Berhasil

Pada halaman status berhasil terdapat halaman detail penyewaan yang memuat informasi detail mengenai penyewaan yang dilakukan oleh pelanggan

kepada rental. Tampilan antarmuka halaman detail penyewaan dengan status berhasil ditunjukkan oleh Gambar 5.48.

Keterangan :

1. Tombol kembali untuk mengarahkan ke halaman sebelumnya.
2. *Text View* judul halaman.
3. *Text View* status penyewaan untuk menampilkan status penyewaan.
4. *Text View* informasi kendaraan yang dipesan.
5. *Text View* informasi biodata pelanggan yang melakukan penyewaan.
6. *Link* untuk mengarahkan ke halaman profil pelanggan.
7. *Text View* enampilkan informasi total pembayaran penyewaan.
8. *Link* untuk mengarahkan ke halaman lihat foto bukti pembayaran.
9. Tombol untuk melakukan cek persyaratan.

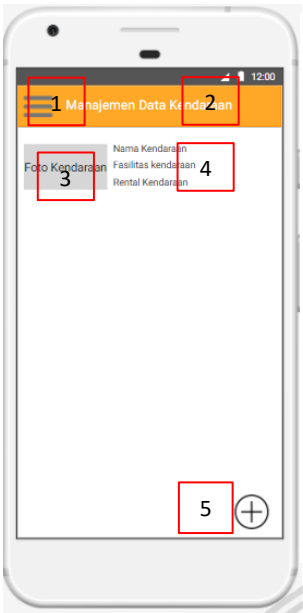
Gambar 5.48 Tampilan Antarmuka Halaman Detail Penyewaan dengan Status Berhasil

b. Menu Manajemen Kendaraan

Menu manajemen kendaraan merupaka menu yang digunakan pemilik rental untuk melakukan pengelolaan kendaraan seperti melihat daftar kendaraan, menambah kendaraan, mengubah kendaraan dan menghapus kendaraan. Halaman manajemen kendaraan merupakan halaman yang berfungsi menampilkan daftar kendaraan dari suatu rental yang telah tersimpan dalam sistem. Tampilan antarmuka halaman manajemen kendaraan di gambar oleh Gambar 5.49.

Halaman tambah data kendaraan merupakan halaman yang berfungsi bagi pemilik rental untuk menambahankan data kendaraan kedalam sistem. Tampilan antarmuka halaman tambah kendaraan di gambarkan oleh Gambar 5.50.

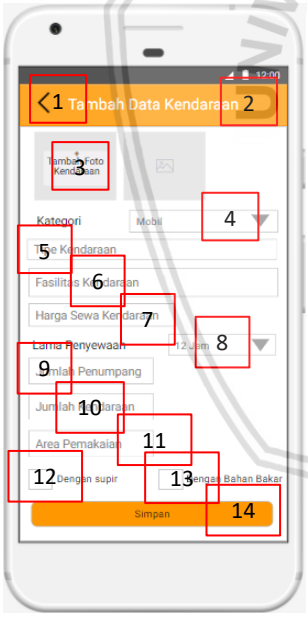
Halaman ubah kendaraan bertujuan untuk mengubah data kendaraan yang sebelumnya telah tersimpan dalam sistem. Tampilan antarmuka halaman ubah kendaraan ditunjukkan oleh Gambar 5.51.



Keterangan :

1. *Menu Drawer* untuk menampilkan menu utama pada sistem.
2. *Text view* judul halaman.
3. *Image view* untuk menampilkan judul halaman.
4. *Text view* untuk menampilkan informasi kendaraan
5. Tombol untuk menambah data kendaraan.

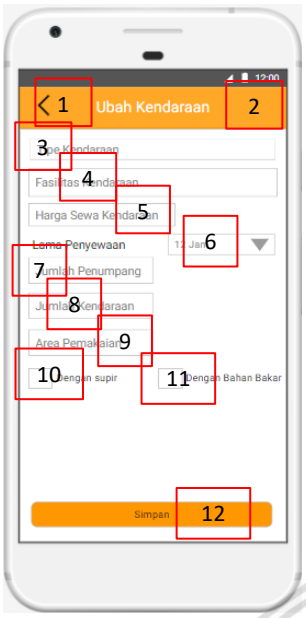
Gambar 5.49 Tampilan Antarmuka Halaman Manajemen Kendaraan



Keterangan :

1. Tombol untuk mengarahkan ke halaman sebelumnya.
2. *Text view* judul halaman.
3. Button untuk menambahkan foto kendaraan.
4. Dropdown untuk memilih kategori kendaraan.
5. *Edit text* untuk mengisi tipe kendaraan.
6. *Edit text* untuk mengisi fasilitas kendaraan.
7. *Edit text* untuk mengisi harga sewa kendaraan.
8. Dropdown untuk memilih lama penyewaan kendaraan.
9. *Edit text* untuk mengisi jumlah penumpang.
10. *Edit text* untuk mengisi jumlah kendaraan.
11. *Edit text* untuk mengisi area pemakaian.
12. Checkbox fasilitas sopir.
13. Checkbox fasilitas bahan bakar.
14. Tombol untuk menyimpan data kendaraan.

Gambar 5.50 Tampilan Antarmuka Halaman Tambah Kendaraan



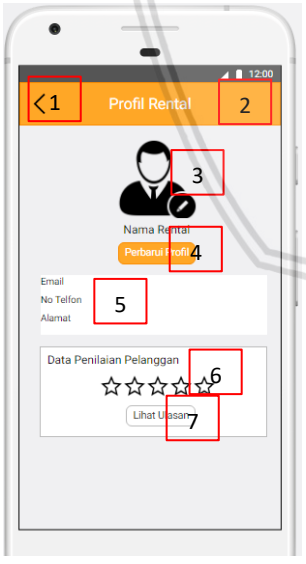
Keterangan :

1. Tombol untuk mengarahkan ke halaman sebelumnya.
2. *Text view* judul halaman.
3. *Edit text* untuk mengisi tipe kendaraan.
4. *Edit text* untuk mengisi fasilitas kendaraan.
5. *Edit text* untuk mengisi harga sewa kendaraan.
6. Dropdown untuk memilih lama penyewaan kendaraan.
7. *Edit text* untuk mengisi jumlah penumpang.
8. *Edit text* untuk mengisi jumlah kendaraan.
9. *Edit text* untuk mengisi area pemakaian.
10. Checkbox fasilitas sopir.
11. Checkbox fasilitas bahan bakar.
12. Tombol untuk menyimpan perubahan.

Gambar 5.51 Tampilan Antarmuka Halaman Ubah Kendaraan

c. Menu Profil

Menu profil merupakan menu yang digunakan untuk melihat profil dan melakukan ubah profil. Halaman antarmuka profil rental digambarkan oleh Gambar 5.52.

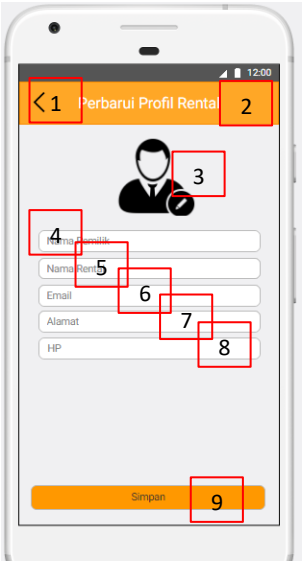


Keterangan :

1. Tombol untuk mengarahkan ke halaman sebelumnya.
2. *Text view* judul halaman.
3. *Image view* untuk menampilkan foto profil rental.
4. Tombol untuk mengarahkan ke halaman ubah profil.
5. *Rating bar* untuk menampilkan penilaian pelanggan.
6. Tombol untuk menampilkan daftar ulasan.

Gambar 5.52 Tampilan Antarmuka Halaman Profil Rental

Halaman ubah profil rental berfungsi untuk melakukan perubahan terhadap profil rental yang tersimpan dalam sistem. Tampilan antarmuka halaman ubah profil rental di tunjukkan oleh Gambar 5.53.



Keterangan :

1. Tombol untuk mengarahkan ke halaman sebelumnya.
2. *Text view* judul halaman.
3. *Image view* untuk menampilkan foto profil rental.
4. *Edit text* untuk mengubah nama pemilik rental.
5. *Edit text* untuk mengubah nama rental.
6. *Edit text* untuk mengubah email rental.
7. *Edit text* untuk mengubah alamat rental.
8. *Edit text* untuk mengubah kontak.

Gambar 5.53 Tampilan Antarmuka Halaman Ubah Profil Rental

5.1.6.4 Perancangan Antarmuka pada sisi pelanggan iterasi 1

Pada iterasi pertama perancangan antarmuka pelanggan, terdapat beberapa penambahan kebutuhan fungsional yang juga membutuhkan penambahan perancangan antarmuka. Penambahan antarmuka tersebut adalah cari kendaraan, daftar penilaian rental, filter pencarian dan lihat penilaian pelanggan.

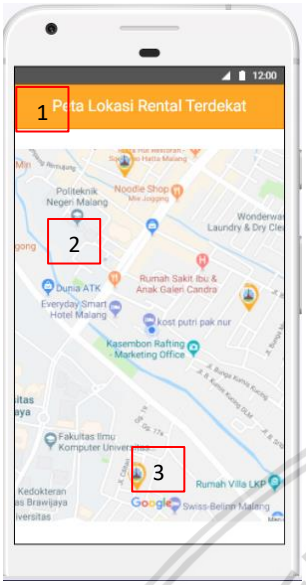
Tampilan antarmuka cari kendaraan pada iterasi pertama mengalami perubahan, dimana pada antarmuka ini terdapat penambahan spinner jumlah kendaraan. Tampilan antarmuka halaman cari kendaraan hasil iterasi pertama ditunjukkan oleh Gambar 5.54.

Keterangan :

1. Tombol untuk menampilkan menu *drawer.*
2. Text view judul halaman.
3. Date picker untuk memilih tanggal sewa.
4. Date picker untuk memilih tanggal kembali.
5. Spinner untuk memilih kategori kendaraan.
6. Spinner untuk memilih jumlah kendaraan.
7. Tombol untuk melakukan proses pencarian kendaraan.

Gambar 5.54 Tampilan Antarmuka Halaman Pencarian Kendaraan Iterasi 1

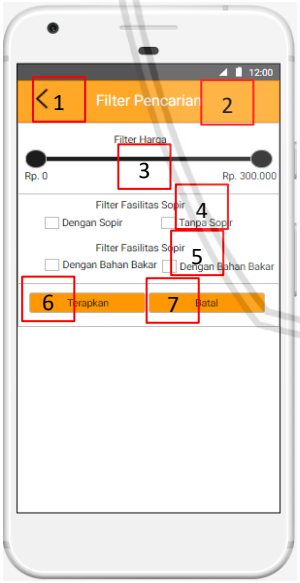
Pada iterasi pertama juga didapatkan suatu perubahan pada kebutuhan fungsional melihat rental lokasi terdekat. Perancangan antarmuka dari kebutuhan tersebut akan ditunjukkan oleh Gambar 5.55.



Keterangan :
1. Text view judul halaman.
2. Peta Lokasi.
3. Titik atau *marker* lokasi rental kendaraan berdasarkan radius terdekat.

Gambar 5.55 Tampilan Antarmuka Halaman Rental Kendaraan Terdekat

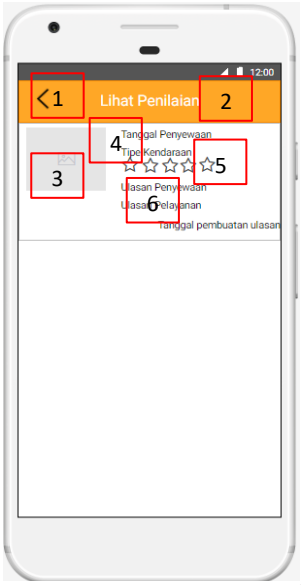
Penambahan kebutuhan fungsional filter pencarian kendaraan yang dihasilkan pada iterasi pertama akan digambarkan pada Gambar 5.56.



Keterangan :
1. Tombol kembali untuk mengarahkan ke halaman sebelumnya.
2. *Text view* judul halaman.
3. *Range seek bar* untuk memilih *range* harga.
4. *Checkbox* untuk memilih filter fasilitas sopir.
5. *Checkbox* untuk memilih filter fasilitas bahan bakar.
6. Tombol terapkan untuk menerapkan *filter* yang telah dipilih.
7. Tombol batal untuk membatalakan pemilihan *filter*.

Gambar 5.56 Tampilan Antarmuka Halaman Filter Pencarian Iterasi 1

Penambahan kebutuhan fungsional lihat penilaian pelanggan yang dihasilkan pada iterasi pertama akan ditampilkan pada Gambar 5.57.



Keterangan :

1. Tombol kembal untuk mengarahkan ke halaman sebelumnya.
2. Text view judul halaman.
3. Imageview untuk menampilkan foto kendaraan.
4. Textview untuk menampilkan informasi penyewaan yang telah selesai dan telah dinilai.
5. Ratingbar untuk menampilkan penilaian pelanggan.
6. Textview untuk menampilkan informasi ulasan pelanggan.

Gambar 5.57 Tampilan Antarmuka Halaman Lihat Ulasan Pelanggan Iterasi 1

Penambahan kebutuhan fungsional lihat daftar penilaian rental merupakan antarmuka yang berfungsi untuk menampilkan seluruh penilaian yang ditujukan pada rental tertentu. Halaman ini merupakan bagian dari halaman profil rental. Tampilan antarmuka halaman daftar penilaian rental yang dihasilkan pada iterasi pertama akan ditampilkan pada Gambar 5.58.

Keterangan :

1. Tombol kembal untuk mengarahkan ke halaman sebelumnya.
2. Text view judul halaman.
3. Imageview untuk menampilkan foto kendaraan.
4. Textview untuk menampilkan informasi penyewaan yang telah selesai dan telah dinilai.
5. Ratingbar untuk menampilkan penilaian pelanggan.
6. Textview untuk menampilkan informasi ulasan pelanggan.

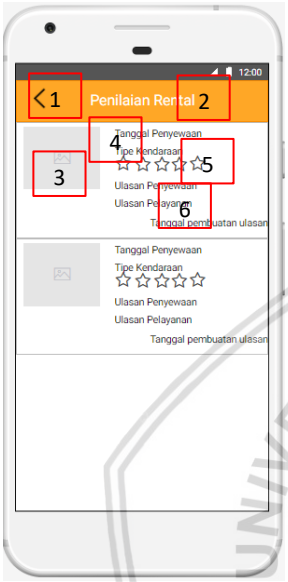
Gambar 5.58 Tampilan Antarmuka Halaman Daftar Penilaian Rental Iterasi 1

5.1.6.5 Perancangan Antarmuka pada sisi pemilik rental iterasi 1

Pada sisi pemilik rental juga terdapat penambahan kebutuhan fungsional yang didapatkan dari iterasi pertama, penambahan tersebut adalah melihat daftar ulasan, menambah rekening pembayaran, ubah rekening pembayaran dan

menghapus rekening pembayaran. Hal ini juga mempengaruhi adanya penambahan perancangan antarmuka pada penelitian ini.

Penambahan kebutuhan fungsional melihat daftar ulasan rental. Kebutuhan fungsional ini berdampak pada perancangan antarmuka untuk menambahkan menu melihat ulasan. Menu ini terdiri dari halaman daftar ulasan yang menampilkan seluruh ulasan yang masuk pada penyewaan rental. Tampilan antarmuka halaman daftar ulasan rental ditunjukkan oleh Gambar 5.59.



Keterangan :

1. Tombol kembal untuk mengarahkan ke halaman sebelumnya.
2. Text view judul halaman.
3. Imageview untuk menampilkan foto kendaraan.
4. Textview untuk menampilkan informasi penyewaan yang telah selesai dan telah dinilai.
5. Ratingbar untuk menampilkan penilaian pelanggan.
6. Textview untuk menampilkan informasi ulasan pelanggan.

Gambar 5.59 Tampilan antarmuka halaman daftar ulasan rental

Tampilan antarmuka yang terakhir dihasilkan dari iterasi pertama adalah halaman pengaturan profil, daftar rekening pembayaran, tambah rekening pembayaran, ubah rekening pembayaran dan hapus rekening pembayaran. Halaman pengaturan profil merupakan halaman yang berfungsi bagi pemilik rental untuk mengarahkan ke halaman ubah profil atau halaman pengaturan rekening pembayaran. Tampilan antarmuka halaman pengaturan profil ditunjukkan oleh Gambar 5.60.

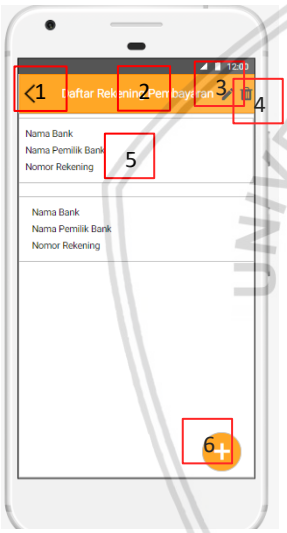
Halaman daftar rekening pembayaran merupakan halaman yang diarahkan oleh tombol pengaturan pembayaran yang ada pada halaman pengaturan profil diatas. Halaman daftar rekening pembayaran berfungsi untuk menampilkan daftar rekening rental yang tersimpan pada sistem. Tampilan antarmuka halaman daftar kendaraan ditunjukkan oleh Gambar 5.61.

Pada halaman daftar rekening pembayaran terdapat tombol tambah untuk menambahkan rekening pembayaran. Halaman tambah rekening pembayaran berfungsi untuk menambahkan rekening pembayaran dan menyimpan dalam siste. Tampilan antarmuka halaman tambah rekening pembayaran ditunjukkan oleh Gambar 5.62.

Keterangan :

1. Tombol kembal untuk mengarahkan ke halaman sebelumnya.
2. Text view judul halaman.
3. Tombol untuk mengarahkan ke halaman ubah profil.
4. Tombol untuk mengarahkan ke halaman pengaturan rekening pembayaran.

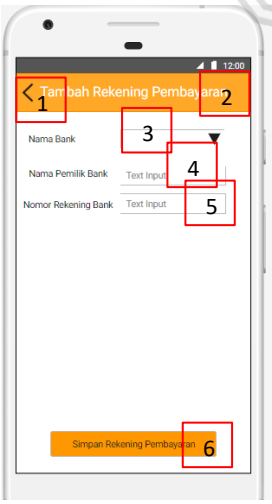
Gambar 5.60 Halaman Antarmuka Pengaturan profil



Keterangan :

1. Tombol kembal untuk mengarahkan ke halaman sebelumnya.
2. Text view judul halaman.
3. Tombol untuk mengarahkan ke halaman ubah rekening pembayaran.
4. Tombol untuk mengarahkan ke halaman hapus rekening pembayaran.
5. Text view untuk menampilkan informasi pembayaran.
6. Tombol untuk mengarahkan ke halaman tambah rekening pembayaran.

Gambar 5.61 Tampilan antarmuka daftar rekening pembayaran

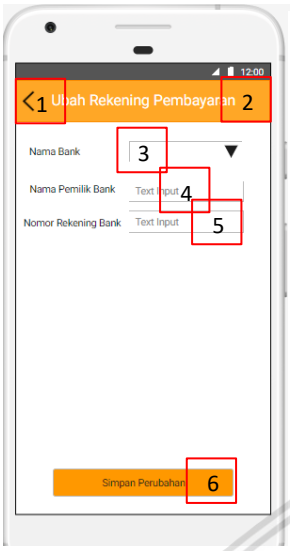


Keterangan :

1. Tombol kembali untuk mengarahkan ke halaman sebelumnya.
2. Text view judul halaman.
3. Spinner untuk memilih nama bank.
4. Edit text untuk mengisi nama pemilik rekening bank.
5. Edit text untuk mengisi nomor rekening bank.
6. Tombol untuk menyimpan rekening pembayaran.

Gambar 5.62 Tampilan Antarmuka Halaman Tambah Rekening

Sedangkan halaman untuk mengubah rekening pembayaran ditunjukkan oleh Gambar 5.63.



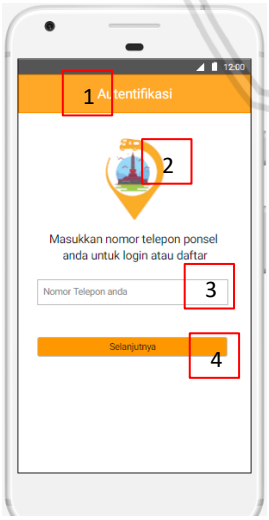
Keterangan :

1. Tombol kembali untuk mengarahkan ke halaman sebelumnya.
2. Text view judul halaman.
3. Spinner untuk memilih nama bank.
4. Edit text untuk mengubah nama pemilik rekening bank.
5. Edit text untuk mengubah nomor rekening bank.
6. Tombol untuk menyimpan perubahan rekening pembayaran.

Gambar 5.63 Tampilan Antarmuka Halaman Ubah Rekening Pembayaran

5.1.6.6 Perancangan Antarmuka pada sisi pengguna iterasi 2

Pada sub bab ini akan dijelaskan tentang perancangan antarmuka yang terdapat pada iterasi 2 pada sisi pengguna. Pada iterasi 2, pengguna hanya memiliki satu penambahan kebutuhan fungsional yaitu Autentifikasi dengan menggunakan nomor telepon. Kebutuhan fungsional tersebut menggantikan Registrasi dan Login. Sehinga kebutuhan fungsional tersebut penting untuk pengguna agar dapat masuk dan mendaftar sebagai pelanggan pada sistem. Gambar 5.64 akan menjelaskan tentang perancangan Antarmuka Autentifikasi dengan nomor telepon.



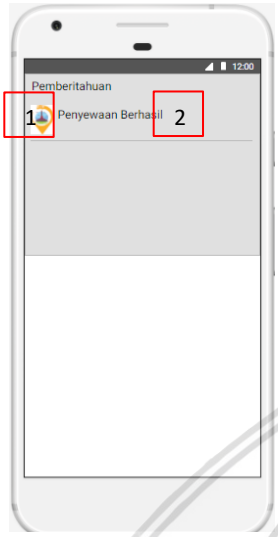
Keterangan :

1. *Text view* Judul Halaman.
2. *Image view* untuk menampilkan logo aplikasi.
3. *Edit text* untuk mengisi nomor telepon.
4. Tombol untuk mengarahkan ke halaman selanjutnya.

Gambar 5.64 Tampilan Antarmuka Halaman Auntentifikasi Nomor Telepon

5.1.6.7 Perancangan Antarmuka pada sisi pelanggan iterasi 2

Pada iterasi kedua, pelanggan hanya memiliki satu penambahan kebutuhan fungsional yaitu menerima pemberitahuan. Gambar 5.65 akan menjelaskan tentang perancangan Antarmuka Menerima Pemberitahuan.



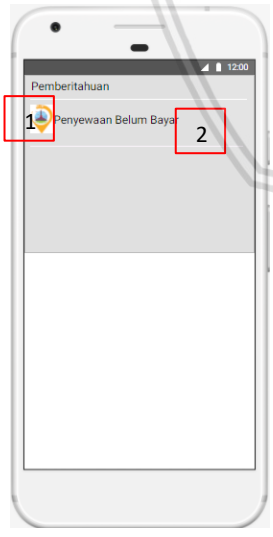
Keterangan :

1. Icon untuk menampilkan logo aplikasi.
2. *Text view* untuk pesan pemberitahuan.

Gambar 5.65 Tampilan Antarmuka Halaman Pemberitahuan Pelanggan

5.1.6.8 Perancangan Antarmuka pada sisi pemilik rental iterasi 2

Pada iterasi kedua, sisi pemilik rental memiliki dua penambahan kebutuhan fungsional yang salah satunya adalah kebutuhan fungsional menerima pemberitahuan. Gambar 5.66 akan menjelaskan tentang perancangan Antarmuka Menerima Pemberitahuan.



Keterangan :

1. Icon untuk menampilkan logo aplikasi.
2. *Text view* untuk pesan pemberitahuan.

Gambar 5.66 Tampilan Antarmuka Halaman Pemberitahuan Pemilik Rental

5.1.7 Perancangan Algoritme

Perancangan algoritme merupakan perancangan yang berfungsi untuk menjelaskan langkah-langkah atau urutan untuk melakukan suatu fungsi tertentu. Pada bagian ini akan dijelaskan beberapa perancangan algoritme yaitu perancangan algoritme pencarian kendaraan, jumlah hari penyewaan dan total biaya sewa.

5.1.7.1 Algoritme Pencarian Kendaraan

Algoritme pencarian kendaraan merupakan algoritme yang berfungsi untuk melakukan proses pencarian kendaraan yang tersedia sesuai dengan tanggal sewa, tanggal kembali, kategori kendaraan dan jumlah kendaraan yang diinginkan pengguna. Algoritme pencarian kendaraan akan ditunjukkan oleh Gambar 5.67.

```
Algoritme pencarian kendaraan

Deklarasi :

   - Date → tanggalSewaPencarian, tanggalKembaliPencarian,
     tanggalSewaDipesan, tanggalKembaliDipesan
   - Integer → jumlahKendaraan, jumlahKendaraanPencarian,
     jumlahKendaraanDipesan
   - String → kategoriKendaraanPencarian
Deskripsi :

   - Masukkan : tanggalSewaPencarian, tanggalKembaliPencarian,
     jumlahKendaraanPencarian, kategoriKendaraanPencarian
   - Proses :
     1. Menyimpan nilai tanggal sewa pencarian, tanggal kembali
        pencarian, jumlah kendaraan pencarian dan kategori
        kendaraan pencarian.
     2. Membaca data kendaraan pada child kendaraan sesuai dengan
        kategoriKendaraanPencarian, jika data tersedia akan
        dilakukan langkah berikut :
        a. Melakukan perulangan sesuai dengan banyaknya data
           kendaraan yang tersimpan dan menyimpan index data
           kendaraan yang sedang di eksekusi.
            i. Membaca seluruh variabel data kendaraan yang
               sedang di eksekusi sesuai index dan menyimpannya
               didalam variabel.
           ii. Melakukan seleksi kondisi, dengan membaca data
               kendaraan yang sedang di eksekusi pada child
               cekSisaKendaraan. Jika data kendaraan terdapat
               pada child cekSisaKendaraan, maka akan dilakukan
               langkah-langkah berikut :
                  1. Melakukan perulangan sesuai dengan banyaknya
                     data kendaraan di eksekusi yang terdapat
                     pada child cekSisaKendaraan dan menyimpan
                     index data cekSisaKendaraan yang sedang di
                     eksekusi.
```

2. Membaca seluruh variabel data penyewaan yang sedang di eksekusi.
3. Menyimpan variabel dari data penyewaan berupa tanggal sewa yang sudah dipesan, tanggal kembali yang sudah dipesan dan jumlah kendaraan yang sudah dipesan.
4. Melakukan seleksi kondisi, jika tanggal sewa pencarian sebelum tanggal kembali yang sudah dipesan, atau tanggal sewa pencarian sama dengan tanggal kembali yang sudah dipesan dan tanggal kembali pencarian sesudah tanggal sewa yang sudah dipesan, atau tanggal kembali pencarian sama dengan tanggal sewa tanggal sewa yang sudah dipesan, atau tanggal sewa pencarian sama dengan tanggal sewa yang sudah dipesan dan tanggal kembali pencarian sama dengan tanggal kembali yang sudah dipesan, maka akan dilakukan langkah-langkah berikut :
   a. Melakukan pengecekkan apakah jumlah sisa kendaraan lebih besar atau sama dengan jumlah kendaraan pencarian, jika ya maka akan dilakukan :
      i. Menambahkan data kendaraan ke dalam array list.
   b. Jika tidak akan dilakukan :
      i. Menghapus data kendaraan dari array.
5. Jika tanggal tidak sama, maka akan dilakukan langkah berikut :
   a. Melakukan pengecekkan apakah jumlah kendaraan lebih besar atau sama dengan jumlah kendaraan pencarian, jika ya maka akan dilakukan :
      i. Menambahkan data kendaraan ke dalam array .
   b. Jika tidak akan dilakukan :
      i. Menghapus data kendaraan dari array.
6. Perulangan berhenti.

```
            iii.  Jika data kendaraan tidak terdapat pada tabel
                  cekSisaKendaraan, maka akan dilakukan seleksi
                  kondisi :
                     1. jumlah kendaraan lebih besar atau sama
                        dengan jumlah kendaraan pencarian, jika
                        ya maka akan dilakukan :
                          a.  Menambahkan data kendaraan ke
                              array.
                     2. jika tidak maka akan dilakukan :
                          a.  Menghapus data kendaraan dari array.
              iv.   Menampilkan data kendaraan.
         b. Perulangan berhenti.
      3.  Jika tidak tersedia akan ditampilkan pesan peringatan.
```

Gambar 5.67 Algoritme Pencarian Kendaraan

5.1.7.2 Algoritme Jumlah Hari Penyewaan

Algoritme jumlah hari penyewaan merupakan algoritme yang berfungsi untuk mengetahui banyaknya hari atau jumlah hari penyewaan atas kendaraan yang diinginkan pengguna. Algoritme jumlah hari penyewaan akan ditunjukkan oleh Gambar 5.68.

```
Algoritme jumlah hari penyewaan

Deklarasi :

   -  String → tanggalSewaPencarian, tanggalKembaliPencarian
   -  Integer → jumlahHariPenyewaan

Deskripsi :

   -  Masukkan : -
   -  Proses :
      1. Menyimpan seluruh nilai variabel tanggalSewaPencarian dan
         tanggalKembaliPencarian.
      2. Melakukan seleksi kondisi, jika variabel
         tanggalSewaPencarian sama dengan tanggalKembaliPencarian
         maka akan dilakukan :
          a.  Menyimpan nilai variabel jumlahHariPenyewaan dengan
              nilai 1.
      3. Jika tidak maka akan dilakukan :
          a.  Melakukan perubahan format data variabel
              tanggalSewaPencarian dan tanggalKembaliPencarian
              kedalam tipe data Date.
          b.  Melakukan perhitungan jumlah hari penyewaan dengan
              menggunakan perhitungan Date.
          c.  Menyimpan nilai variabel jumlahHariPenyewaan dengan
              nilai hasil perhitungan jumlah hari penyewaan ditambah
              1.
```

Gambar 5.68 Algoritme Jumlah Hari Penyewaan

5.1.7.3 Algoritme Total Biaya Sewa

Algoritme total biaya sewa merupakan algoritme untuk mengetahui total biaya sewa yang harus dibayarkan oleh pengguna sesuai dengan lama hari penyewaan, fasilitas kendaraan dan harga kendaraan. Algoritme total biaya sewa ditunjukkan oleh Gambar 5.69.

```
Algoritme Total Biaya Sewa

Deklarasi :

   - String → idKendaraan, kategoriKendaraanPencarian,
     jumlahKendaraanPencarian, lamaPenyewaan, fasilitasSupir,
     durasiSewa, jumlahHariPenyewaan

Deskripsi :

   - Masukkan : -
   - Proses :
     1. Menyimpan seluruh nilai variabel idKendaraan,
        kategoriKendaraanPencarian, jumlahKendaraanPencarian.
     2. Membaca data kendaraan yang sesuai
        kategoriKendaraanPencarian dan menyimpan variabel
        lamaPenyewaan, fasilitasSupir, durasiSewa.
     3. Mencoba untuk melakukan beberapa langkah berikut :
          a. Melakukan seleksi kondisi jika variabel
             lamaPenyewaan sama dengan durasiSewa maka akan
             dilakukan :
              i.   Mengkalikan nilai variabel
                   jumlahHariPenyewaan dengan hargaSewa dan
                   jumlahKendaraan dan menyimpan hasilnya
                   kedalam variabel totalBiayaPembayaran.
          b. Melakukan seleksi kondisi jika variabel
             lamaPenyewaan sama dengan durasiSewa, variabel
             fasilitasSupir bernilai true dan
             jumlahHariPenyewaan bernilai 1 maka akan dilakukan
              i.   Mengkalikan nilai variabel
                   jumlahHariPenyewaan dengan hargaSewa dan
                   jumlahKendaraan dan menyimpan hasilnya
                   kedalam variabel totalBiayaPembayaran.
          c. Melakukan seleksi kondisi jika variabel
             lamaPenyewaan sama dengan durasiSewa, variabel
             fasilitasSupir bernilai true dan
             jumlahHariPenyewaan bernilai lebih dari 1 maka
             akan dilakukan :
              i.   Mengkalikan nilai variabel
                   jumlahHariPenyewaan dengan hargaSewa dan
                   jumlahKendaraan lalu dikalikan 2 dan
                   menyimpan hasilnya kedalam variabel
                   totalBiayaPembayaran.
          d. Jika tidak ada kondisi yang memenuhi maka akan
             dilakukan :
```

```
            i.  Mengkalikan nilai variabel
                jumlahHariPenyewaan dengan hargaSewa dan
                jumlahKendaraan dan menyimpan hasilnya
                kedalam variabel totalBiayaPembayaran.
    4. Jika terdapat kesalahan sistem maka akan menampilkan
       pesan peringatan.
```

Gambar 5.69 Algoritme Total Biaya Sewa

5.2 Implementasi

Bab ini merupakan tahapan yang membahas tentang implementasi Sistem Penyewaan Kendaraan Kota Malang yang didapatkan dari hasil analisis kebutuhan dan perancangan. Pada bagian ini akan dijelaskan beberapan pembahasan yang terdiri dari spesifikasi sistem, batasan implementasi, implementasi basis data, implementasi klas, implementasi kode program dan implemetasi antarmuka. Sesuai dengan metode pengembangan MASAM yang di adopsi, pada sub bab implementasi terdapat *task* pengaturan lingkungan sistem (*environment setup*) yang akan di jelaskan pada bagian spesifikasi sistem dan siklus iterasi (*iteration cycle*) akan dijelaskan pada bagian siklus iterasi.

5.2.1 Spesifikasi Sistem

Dalam pengembangan sistem penyewaan kendaraan kota malang ini, pengembangan dikembangkan dalam lingkungan implementasi yang terdiri dari dua buah perangkat, yaitu perangkat keras dan perangkat lunak.

5.2.1.1 Spesifikasi Perangkat Keras

Spesifikasi perangkat keras yang digunakan dalam proses pengembangan sistem penyewaan kendaraan kota malang ditunjukkan pada tabel 5.1.

Tabel 5.1 Spesifikasi Perangkat Keras komputer

Notebook Hp Pavilion 14-ab135tx	
Processor	Intel® Core™ i7-6500U CPU @2.50GHz 2.59GHz
Memory(RAM)	8 GB
Harddisk	1 TB 5400 rpm SATA
Graphic Card	NVIDIA GeForce 940M
Monitor	14" diagonal HD BrightView WLED-backlit (1366 x 768)

5.2.1.2 Spesifikasi Perangkat Lunak

Spesifikasi perangkat lunak yang digunakan dalam proses pengembangan sistem penyewaan kendaraan kota malang ditunjukkan pada tabel 5.2.

Tabel 5.2 Spesfikasi Perangkat Lunak Komputer

Notebook Hp Pavilion 14-ab135tx

Operating System	Microsoft Windows 10 Pro 64bit
Programming Language	Java
Programming Tool	Android Studio 3.0.1
Android SDK Tools	26.1.1
Gradle version	4.1
Database Management System	Firebase realtime database SDK 11.4.2

5.2.1.3 Spesifikasi Perangkat Bergerak

Spesifikasi perangkat bergerak yang digunakan dalam proses pengembangan sistem penyewaan kendaraan kota malang dijelaskan pada tabel 5.3.

Tabel 5.3 Spesifikasi Perangkat Bergerak

Smartphone Android Xiaomi Mi4c	
Operating System	Android nougat 7.0
CPU	Hexa-core (4x1.4 GHz Cortex-A53 & 2x1.8 GHz Cortex-A57)
GPU	Adreno 418
Memory(RAM)	3 GB
Chipset	Qualcomm MSM8992 Snapdragon 808
Display	5.0 inches

5.2.2 Batasan Implementasi

Batasan impelemntasi dalam pengembangan sistem penyewaan kendaraan kota malang adalah sebagai berikut :

1. Sistem Penyewaan Kendaraan Kota Malang dirancang dan diimplementasikan dengan menggunakan Android Studio.
2. Bahasa pemrograman yang digunakan adalah java.
3. Manajemen Basis Data yang digunakan adalah *Firebase Real Time Database.*

5.2.3 Siklus Iterasi

Pada bagian ini merupakan bagian yang di adopsi dari metode pengembangan MASAM yang dimana siklus iterasi (*release cycle)* dari fase *development* terdiri dari beberapa *task* yaitu *release planning, iteration cycle* dan rilis. Berikut ini akan dijelaskan bagaimana implementasi dari *task* tersebut pada implementasi penelitian ini.

5.2.3.1 *Release Planning*

Task release planning bertujuan untuk mengetahui tentang perencanaan iterasi yang akan dilakukan pada penelitian ini. Pada penelitian ini akan dilakukan sebanyak dua kali iterasi untuk melakukan implementasi dari kebutuhan-kebutuhan yang telah didapatkan kedalam kode program.

5.2.3.2 *Iteration cycle*

Task iteration cycle terdiri dari beberapa *task* yaitu *iteration planning*, *iteration* dan *Test Driven Development (TDD).* Pada *task iteration planning*, akan di pilih tentang story card apa saja yang akan di implementasikan pada iterasi pertama maupun iterasi kedua. Setelah memilih *story card* yang akan diimplementasikan berdasarkan prioritas, selanjutnya memilih kode kebutuhan apa saja yang akan diimplementasikan yang terdapat pada sub bab spesifikasi kebutuhan sesuai dengan urutan *story card* yang telah dipilih berdasarkan prioritas.

Tabel 5.4 akan menjelaskan tentang kebutuhan fungsional pengguna apa saja yang akan diimplementasikan pada iterasi pertama, kode kebutuhan yang tercantum pada Tabel 5.4 mengacu pada Tabel 4.4. Sedangkan Tabel 5.5 menjelaskan kebutuhan pelanggan yang mengacu pada tabel spesifikasi kebutuhan pelanggan Tabel 4.6 dan Tabel 5.6 menjelaskan kebutuhan pemilik rental yang akan diimplementasikan pada diiterasi pertama yang mengacu pada Tabel 4.9.

Tabel 5.4 Kebutuhan Pengguna pada Siklus Iterasi 1

No	Kode Kebutuhan
1	SRS_001_03
2	SRS_001_04

Tabel 5.5 Kebutuhan Pelanggan pada Siklus Iterasi 1

No	Kode Kebutuhan
1	SRS_002_01
2	SRS_002_02
3	SRS_002_03
4	SRS_002_04
5	SRS_002_05
6	SRS_002_06
7	SRS_002_07
8	SRS_002_08
9	SRS_002_09

10	SRS_002_10

Tabel 5.6 Kebutuhan Pemilik rental pada Siklus Iterasi 1

No	Kode Kebutuhan
1	SRS_003_01
2	SRS_003_02
3	SRS_003_03
4	SRS_003_04
5	SRS_003_05
6	SRS_003_06
7	SRS_003_07
8	SRS_003_08
9	SRS_003_09
10	SRS_003_10
11	SRS_003_11

Sedangkan pada iterasi kedua akan diimplementasikan kebutuhan fungsional yang akan dijelaskan pada Tabel 5.7 untuk sisi pelanggan dan Tabel 5.8 untuk sisi pemilik rental.

Tabel 5.7 Kebutuhan Pelanggan pada Siklus Iterasi 2

No	Kode Kebutuhan
1	SRS_002_11
2	SRS_002_12
3	SRS_002_13
4	SRS_002_14
5	SRS_002_15
6	SRS_002_16
7	SRS_002_17
8	SRS_002_18
9	SRS_002_19
10	SRS_002_20

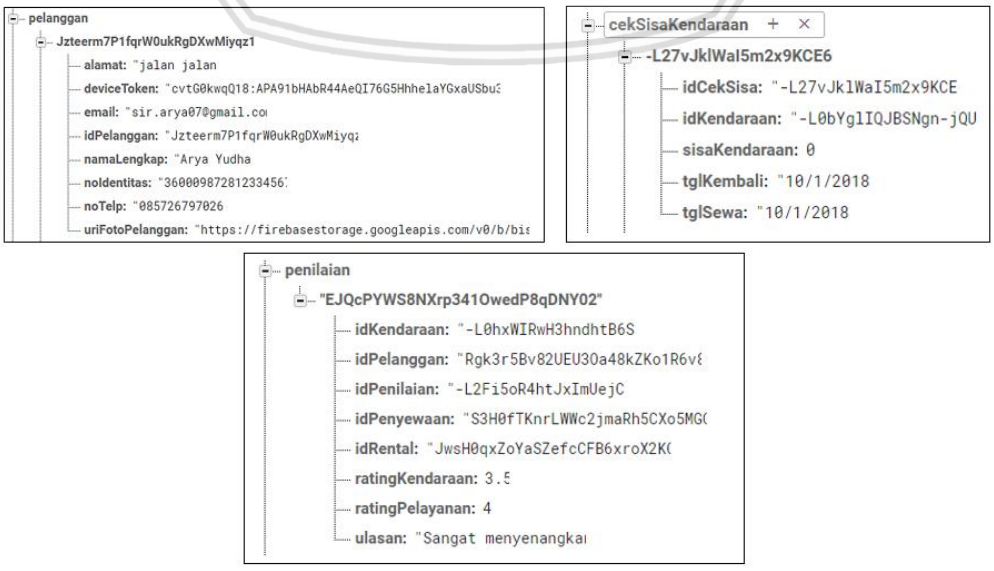
Tabel 5.8 Kebutuhan Pemilik rental pada Siklus Iterasi 2

No	Kode Kebutuhan
1	SRS_003_12
2	SRS_003_13
3	SRS_003_14
4	SRS_003_15
5	SRS_003_16
6	SRS_003_17
7	SRS_003_18
8	SRS_003_19
9	SRS_002_20

Setelah dilakukan pemilihan urutan pengerjaan kebutuhan, selanjutnya adalah melakukan tahapan implementasi kode program. Setelah iterasi sudah selesai semua diimplementasikan akan di lakukan rilis untuk mengetahui apakah ada perubahan atau penambahan yang dimana bagian ini merupakan bagian dari *task* rilis. Tetapi perubahan dan penambahan yang terdapat pada tahapan ini hanya bersifat kecil.

5.2.4 Implementasi Database

Manajemen basis data yang digunakan pada pengembangan sistem penyewaan kendaraan kota Malang adalah *Firebase Realtime Database*. Seperti yang dijelaskan sebelumnya, *Firebase Realtime Database* merupakan manajemen basis data yang menggunakan NoSQL sehingga penjelasan implementasi *database* akan digambarkan dengan operasi *tree* seperti pada Gambar 5.70.



Gambar 5.70 Implementasi *Database*

5.2.5 Implementasi *Class*

Setiap *class* yang telah dirancang pada proses perancangan direalisasikan pada sebuah file program dengan ekstensi *.java. Pasangan antara *class* dengan file program yang digunakan untuk implementasinya di tunjukan oleh tabel 5.9 untuk sisi pelanggan dan tabel 5.10 untuk sisi pemilik rental.

Tabel 5.9 implementasi *class* pada kode program *.java sistem sisi pelanggan

No.	Paket	Nama *class* atau *interface*	Nama File program
1	Pencarian	MenuPencarian	MenuPencarian.java
2	Pencarian	HasilPencarian	HasilPencarian.java
3	Pencarian	Hasil Pencarian Adapter	Hasil Pencarian Adapter.java
4	Pencarian	DetailKendaraan	DetailKendaraan.java

5	Pencarian	Tab1DetailKendaraan	Tab1DetailKendaraan.java
6	Pencarian	Tab2DetailKendaraan	Tab2DetailKendaraan.java
7	Pencarian	PetaRental	PetaRental.java
8	Pencarian	KendaraanModel	KendaraanModel.java
9	Pencarian	SisaKendaraanModel	SisaKendaraanModel.java
10	Penyewaan	RincianPenyewaan	RincianPenyewaan.java
11	Penyewaan	BuatPenyewaan	BuatPenyewaan.java
12	Penyewaan	PenyewaanModel	PenyewaanModel.java
13	Pembayaran	DaftarRekening	DaftarRekening.java
14	Pembayaran	DaftarRekeningAdapter	DaftarRekeningAdapter.java
15	Pembayaran	UnggahBuktiPembayaran	UnggahBuktiPembayaran.java
16	Pembayaran	PembayaranModel	PembayaranModel.java

Tabel 5.10 Implementasi *class* pada kode program *.java sistem sisi pemilik rental

No.	Paket	Nama *class* atau *interface*	Nama File program
1	ManajemenProfil	MenuProfilRental	MenuProfilRental.java
2	ManajemenProfil	UbahProfil	UbahProfil.java
3	ManajemenProfil	RentalModel	RentalModel.java
4	ManajemenProfil	ProfilPelanggan	ProfilPelanggan.java
5	ManajemenProfil	PelangganModel	PelangganModel.java
6	ManajemenProfil	RekeningPembayaranAdapter	RekeningPembayaranAdapter.java
7	ManajemenProfil	PengaturanProfil	PengaturanProfil.java
8	ManajemenProfil	PengaturanRekeningPembayaran	PengaturanRekeningPembayaran.java
9	ManajemenProfil	PengaturanRekeningAdapter	PengaturanRekeningAdapter.java
10	ManajemenProfil	TambahRekeningPembayaran	TambahRekeningPembayaran.java
11	ManajemenProfil	UbahRekeningPembayaran	UbahRekeningPembayaran.java

12	MenuKelolaPenyewaan	DetailPenyewaan	DetailPenyewaan.java
13	MenuKelolaPenyewaan	LihatBuktiPembayaran	LihatBuktiPembayaran.java
14	MenuKelolaPenyewaan	PenyewaanModel	PenyewaanModel.java
15	MenuManajamenKendaraan	MenuManajemenKendaraan	MenuManajemenKendaraan.java
16	MenuManajamenKendaraan	MenuManajemenKendaraanAdapter	MenuManajemenKendaraanAdapter.java
17	MenuManajamenKendaraan	DetailKendaraan	DetailKendaraan.java
18	MenuManajamenKendaraan	TambahKendaraan	TambahKendaraan.java
19	MenuManajamenKendaraan	UbahKendaraan	UbahKendaraan.java
20	MenuManajamenKendaraan	KendaraanModel	KendaraanModel.java

5.2.6 Implementasi Kode Program

Bagian Implementasi Kode Program akan menjelaskan tentang implementasi kode program utama pada sistem penyewaan kendaraan kota malang dengan menggunakan bahasa pemrograman java.

5.2.6.1 Implementasi *Method* getHariPenyewaan

Method `getHariPenyewaan` merupakan *method* yang digunakan untuk menghitung jumlah hari penyewaan kendaraan yang dimasukkan pelanggan pada halaman pencarian.

```
1.  public int getJumlahHariPenyewaan() throws ParseException {
2.          jumlahHariPenyewaan = 0;
3.          final String tglSewaPencarian =
4.  getIntent().getStringExtra("tglSewaPencarian");
5.          final String tglKembaliPencarian =
6.  getIntent().getStringExtra("tglKembaliPencarian");
7.
8.          if (tglSewaPencarian.equals(tglKembaliPencarian)) {
9.              jumlahHariPenyewaan = 1;
10.
11. textViewLamaSewaPemesanan.setText(String.valueOf(jumlahHariPenyew
12. aan));
13.         } else {
14.             SimpleDateFormat myFormat = new
15. SimpleDateFormat("dd/MM/yyyy");
16.             Date date1 = myFormat.parse(tglSewaPencarian);
17.             Date date2 = myFormat.parse(tglKembaliPencarian);
            long diff = date2.getTime() - date1.getTime();
```

```
18.             int dayCount = (int) diff / (24 * 60 * 60 * 1000);
19.             jumlahHariPenyewaan = dayCount + 1;
20.
21. textViewLamaSewaPemesanan.setText(String.valueOf(jumlahHariPenyew
22. aan));
23.         }
24.         return jumlahHariPenyewaan;
    }
```

Gambar 5.71 Implementasi Kode Program *Method* getJumlahHariPenyewaan

Pada baris pertama kode program yang digunakan untuk melakukan deklarasi *method* `getJumlahHariPenyewaan`. Pada baris 2 – 6 berfungsi untuk melakukan inisialisasi variabel `tglSewaPencarian` dan `tglKembaliPencarian` yang didapatkan dari halaman intent sebelumnya. Baris 7 merupakan percabangan dengan kondisi nilai variabel `tglSewaPencarian` sama dengan `tglKembaliPencarian`. Selanjutnya pada baris 8 - 11 merupakan pernyataan yang di eksekusi pada baris 7 apabila terpenuhi. Sedangkan pada baris 12 merupakan kondisi lain apabila baris 7 tidak memenuhi dan akan mengeksekusi pernyataan pada baris 13 – 21. Baris 13 – 14 merupakan inisialisasi format *date* atau tanggal ke dalam format tertentu, lalu pada baris 14 – 15 akan dilakukan perubahan nilai tglSewaPencarian dan tglKembaliPencarian yang semula mempunyai tipe data `String` menjadi `Date`. Lalu baris 16 – 18 merupakan kode program yang berfungsi untuk melakukan perhitungan jumlah hari penyewaan. Pada baris 20 – 21 digunakan untuk melakukan set nilai pada `textViewLamaSewaPemesanan`. Pada baris 23 merupakan kode program yang berfungsi untuk memberikan nilai kembalian.

5.2.6.2 Implementasi *Method* totalBiayaSewa

Method `totalBiayaSewa` merupakan method yang berfungsi untuk mengitung jumlah total pembayaran dari hasil kalkulasi jumlah hari penyewaan di kali kan dengan harga sewa kendaraan.

```
1.  public void totalBiayaSewa() throws ParseException  {
2.      final String idKendaraan =
3.  getIntent().getStringExtra("idKendaraan");
4.      final String kategoriKendaraan =
5.  getIntent().getStringExtra("kategoriKendaraan");
6.      final String jumlahKendaraanPencarian =
7.  getIntent().getStringExtra("jumlahKendaraanPencarian");
8.      final int jmlKendaraan =
9.  Integer.parseInt(jumlahKendaraanPencarian);
10.
11. mDatabase.child("kendaraan").child(kategoriKendaraan).child(idKen
12. daraan).addValueEventListener(new ValueEventListener() {
13.         @Override
14.         public void onDataChange(DataSnapshot dataSnapshot) {
15.             KendaraanModel kendaraan =
16. dataSnapshot.getValue(KendaraanModel.class);
17.             String lamaPenyewaan = kendaraan.getLamaPenyewaan();
18.             boolean fasilitasSupir = kendaraan.isSupir();
19.             double hargaSewa = kendaraan.getHargaSewa();
20.             String durasiPenyewaanKendaraan = "12 Jam";
21.             double total = 0;
22.             try {
23.                 if
```

```
24.  ((lamaPenyewaan.equals(durasiPenyewaanKendaraan) &&
25.  fasilitasSupir == true)) {
26.                      total = (getJumlahHariPenyewaan() *
27.  hargaSewa) * jmlKendaraan;
28.                      totalBiayaPembayaran = total;
29.                      textViewTotalPembayaran.setText("Rp." +
30.  BaseActivity.rupiah().format(total));
31.                  }
32.                  else if
33.  ((lamaPenyewaan.equals(durasiPenyewaanKendaraan) &&
34.  fasilitasSupir == false && getJumlahHariPenyewaan()==1)) {
35.                      total = (getJumlahHariPenyewaan() *
36.  hargaSewa) * jmlKendaraan;
37.                      totalBiayaPembayaran = total;
38.                      textViewTotalPembayaran.setText("Rp." +
39.  BaseActivity.rupiah().format(total));
40.                  }
41.                  else if
42.  ((lamaPenyewaan.equals(durasiPenyewaanKendaraan) &&
43.  fasilitasSupir == false && getJumlahHariPenyewaan()>1)) {
44.                      total = (getJumlahHariPenyewaan() *
45.  (hargaSewa*2)) * jmlKendaraan;
46.                      totalBiayaPembayaran = total;
47.                      textViewTotalPembayaran.setText("Rp." +
48.  BaseActivity.rupiah().format(total));
49.                  } else {
50.                      total = (getJumlahHariPenyewaan() *
51.  (hargaSewa)) * jmlKendaraan;
52.                      totalBiayaPembayaran = total;
53.                      textViewTotalPembayaran.setText("Rp." +
54.  BaseActivity.rupiah().format(total));
55.                  }
56.              } catch (ParseException e) {
57.                  Toast.makeText(getApplicationContext(), "Terdapat
58.  Kesalahan pada Sistem", Toast.LENGTH_LONG).show();
59.                  Intent intent = new Intent(RincianPenyewaan.this,
60.  MainActivity.class);
61.                  startActivity(intent);
62.              }
63.          }
64.
65.          @Override
66.          public void onCancelled(DatabaseError databaseError) {
67.
68.          }
69.      });
```

Gambar 5.72 Implementasi Kode Program *Method* totalBiayaSewa

Pada baris pertama *method* totalBiayaSewa berfungsi untuk melakukan deklarasi *method* `totalBiayaSewa`. Baris 2 – 9 berfungsi untuk melakukan insialisasi nilai variabel `idKendaraan`, `kategoriKendaraan`, `jumlahKendaraanPencarian` yang didapatkan dari halaman sebelumnya. Pada baris ke 10-11 berfungsi untuk mendapatkan data dari basis data firebase pada *child* kendaraan lalu *child* kategoriKendaraan. Baris 12 – 13 dan baris 64-65 merupakan deklarasi *method* `onDataChange` dan `onCancelled` yang dimana *method* tersebut *method* yang di *override* dari `addValueEventListener` untuk membaca dan mendeteksi perubahan pada konten. Baris 14 – 18 berfungsi untuk memanggil semua data dari basis data *firebase* yang sesuai dengan data yang ada di *class* `KendaraanModel`. Baris 19

dan 20 merupakan deklarasi variabel. Baris 27 merupakan *statement* `try` yang berfungsi untuk menangani apabila terdapat kode program yang ada didalam `try` mempunyai kesalahan/error dan selanjutnya apabila terdapat kesalahan akan dijalankan *statement* `catch` pada baris 55. Didalam *statement* `try` terdapat beberapa seleksi kondisi dengan menggunakan `if else` untuk menyeleksi kondisi dari penyewaan yang dilakukan berdasarkan durasi penyewaan kendaraan, fasilitas sopir dan jumlah hari penyewaan yang dilakukan pelanggan. Pada baris 22 merupakan seleksi kondisi apabila nilai variabel `lamaPenyewaan` kendaraan sama dengan variabel `durasiPenyewaanKendaraan` dan nilai `fasilitasSopir` adalah *true* maka akan dijalankan kode program baris 25 – 29. Lalu pada baru 31 merupakan seleksi kondisi apabila nilai variabel `lamaPenyewaan` kendaraan sama dengan variabel `durasiPenyewaanKendaraan` dan nilai `fasilitasSopir` adalah *false* dan nilai *return* dari method `getJumlahHariPenyewaan` sama dengan 1 maka akan dijalankan kode program baris 32 – 38. Selanjutnya pada baris ke 40 merupakan merupakan seleksi kondisi apabila nilai variabel `lamaPenyewaan` kendaraan sama dengan variabel `durasiPenyewaanKendaraan`, nilai `fasilitasSopir` adalah *false* dan nilai *return* dari method `getJumlahHariPenyewaan` lebih dari 1, maka akan dijalankan kode program baris 43 – 47. Dan yang terakhir apabila tidak ada yang memenuhi kondisi tersebut akan dijalankan baris program 49-53.

5.2.6.3 Implementasi *Method* cekTanggal

Method `cekTanggal` berfungsi untuk melakukan pengecekkan terhadap tanggal pencarian dengan tanggal penyewaan yang sebelumnya telah tersimpan didalam basis data.

```
1.  public boolean cekTanggal(String tanggalSewaPencarian, String
2.  tanggalKembaliPencarian, String tanggalSewaDipesan, String
3.  tanggalKembaliDipesan) {
4.      SimpleDateFormat format = new SimpleDateFormat("dd/MM/yyyy");
5.      try {
6.          tglSewaPencarian = format.parse(tanggalSewaPencarian) ;
7.          tglKembaliPencarian =
8.  format.parse(tanggalKembaliPencarian);
9.          tglSewaDipesan = format.parse(tanggalSewaDipesan);
10.         tglKembaliDipesan = format.parse(tanggalKembaliDipesan);
11.
12.         if ((tglSewaPencarian.before(tglKembaliDipesan) ||
13.                 tglSewaPencarian.equals(tglKembaliDipesan)) &&
14. (tglKembaliPencarian.after(tglSewaDipesan) ||
15.                 tglKembaliPencarian.equals(tglSewaDipesan)) ||
16.                 tglSewaPencarian.equals(tglSewaDipesan) &&
17. tglKembaliPencarian.equals(tglKembaliDipesan)) {
18.             cekTanggal = true;
19.         }  else {
20.             cekTanggal = false;
21.         }
22.     } catch (ParseException e) {
23.         Toast.makeText(getApplicationContext(), "Proses Pencarian
24. Gagal Karena Terdapat Kesalahan Sistem",
25. Toast.LENGTH_LONG).show();
26.     }
27.
```

```
28.     return cekTanggal;
    }
```

Gambar 5.73 Implementasi Kode Program *Method* cekTanggal

Pada baris pertama *method* `cekTanggal` terdapat kode program untuk deklarasi nama *method* dengan konstruktor `tanggalSewaPencarian`, `tanggalKembaliPencarian`, `tanggalSewaDipesan` dan `tanggalKembaliDiPesan`. Pada baris 4 merupakan inisialisasi variabel `format` dengan tipe data SimpleDateFormat. Pada baris 6 – 10 digunakan untuk mengubah variabel `tanggalSewaPencarian`, `tanggalKembaliPencarian`, `tglSewaDipesan` dan `tglKembaliDipesan` ke dalam tipe data *date* dengan menggunakan pernyataan *try* dan *catch* yang merupakan suatu pernyataan yang berfungsi untuk menangkap atau mendeteksi kesalahan pada program. Selanjutnya pada baris 12 - 17 merupakan kondisi yang digunakan untuk mengecek tanggal pencarian dengan tanggal yang dipesan. Jika kondisi tersebut memenuhi akan dijalankan pernyataan pada baris 18 dengan memberikan nilai variabel `cekTanggal` `true`. Namun apabila kondisi tidak memenuhi akan dieksekusi baris 20 dengan memberikan variabel `cekTanggal` `false`. Selanjutnya baris 23 merupakan pernyataan yang bertujuan untuk memberikan informasi bahwa terdapat kesalahan sistem. Sedangkan baris 27 untuk memberikan nilai kembalian *method* `cekTanggal`.

5.2.6.4 Implementasi *Method* getHasilPencarian

Method `getHasilPencarian` digunakan oleh sistem untuk melakukan pencarian kendaraan. *Method* `getHasilPencarian` ditunjukkan oleh Gambar 5.74.

```
1.  public void getHasilPencarian() {
2.      final String kategoriKendaraanPencarian =
3.  getIntent().getStringExtra("kategoriKendaraanPencarian");
4.      final String jumlahKendaraanPencarian =
5.  getIntent().getStringExtra("jumlahKendaraanPencarian");
6.      final String tanggalSewaPencarian =
7.  getIntent().getStringExtra("tglSewaPencarian");
8.      final String tanggalKembaliPencarian =
9.  getIntent().getStringExtra("tglKembaliPencarian");
10.     jmlKendaraanPencarian =
11. Integer.parseInt(jumlahKendaraanPencarian);
12.     final LinkedHashSet<KendaraanModel> lhs = new
13. LinkedHashSet<>();
14.
15. mDatabase.child("kendaraan").child(kategoriKendaraanPencarian).a
16. ddValueEventListener(new ValueEventListener() {
17.         @Override
18.         public void onDataChange(DataSnapshot dataSnapshot) {
19.             if (dataSnapshot.exists()) {
20.                 progressBar.setVisibility(View.GONE);
21.                 for (DataSnapshot postSnapshot :
22. dataSnapshot.getChildren()) {
23.                     final KendaraanModel kendaraan =
24. postSnapshot.getValue(KendaraanModel.class)
25.                     jmlKendaraan =
26. kendaraan.getJumlahKendaraan();
```

```
                    final String id =
kendaraan.getIdKendaraan();
                    final int jmlKendaraanModel = jmlKendaraan;

                    Firebase ref = new
Firebase("https://bismillahskripsi-
44a73.firebaseio.com/cekSisaKendaraan");
                    Query query =
ref.orderByChild("idKendaraan").equalTo(id);
                    query.addValueEventListener(new
com.firebase.client.ValueEventListener() {
                        @Override
                        public void
onDataChange(com.firebase.client.DataSnapshot dataSnapshot) {
                            if (dataSnapshot.exists()) {
                                for
(com.firebase.client.DataSnapshot postSnapshot :
dataSnapshot.getChildren()) {
                                    SisaKendaraanModel sisaModel
= postSnapshot.getValue(SisaKendaraanModel.class);
                                    idCekSisa =
sisaModel.getIdCekSisa();
                                    final int sisaKendaraan =
sisaModel.getSisaKendaraan();
                                    String tanggalSewaDipesan =
sisaModel.getTglSewa();
                                    String tanggalKembaliDipesan
= sisaModel.getTglKembali();

                                    if
(cekTanggal(tanggalSewaPencarian, tanggalKembaliPencarian,
tanggalSewaDipesan, tanggalKembaliDipesan)) {
                                        if (sisaKendaraan >=
jmlKendaraanPencarian || jmlKendaraanModel ==
jmlKendaraanPencarian) {

kendaraanModel.add(kendaraan);
                                        } else {

kendaraanModel.remove(kendaraan);
                                            break;
                                        }
                                    } else {
                                        if (jmlKendaraanModel >=
jmlKendaraanPencarian || jmlKendaraanModel ==
jmlKendaraanPencarian) {

kendaraanModel.add(kendaraan);
                                        } else {

kendaraanModel.remove(kendaraan);
                                            break;
                                        }
                                    }
                                }
                            } else {
                                if (jmlKendaraanModel >
jmlKendaraanPencarian || jmlKendaraanModel ==
jmlKendaraanPencarian) {

kendaraanModel.add(kendaraan);
                                } else {

kendaraanModel.remove(kendaraan);
                                }
```

```
92.                                 }
93.                                 lhs.addAll(kendaraanModel);
94.                                 kendaraanModel.clear();
95.                                 kendaraanModel.addAll(lhs);
96.                                 adapter = new
97.   MenuHasilPencarianAdapter(MenuHasilPencarian.this,
98.   kendaraanModel, tanggalSewaPencarian, tanggalKembaliPencarian,
99.   jumlahKendaraanPencarian);
100.                                recyclerView.setAdapter(adapter);
101.                            }
102.                            @Override
103.                            public void onCancelled(FirebaseError
104.  firebaseError) {
105.                            }
106.                        });
107.                    }
108.                } else {
109.                    progressBar.setVisibility(View.GONE);
110.
111.  linearLayoutListKendaraan.setVisibility(View.GONE);
112.
113.  kendaraanTidakTersedia.setVisibility(View.VISIBLE);
114.                }
115.            }
116.            @Override
117.            public void onCancelled(DatabaseError databaseError) {
118.            }
119.        });
120.  }
```

Gambar 5.74 Implementasi Kode Program *Method* getHasilPencarian

Pada baris pertama *method* `getHasilPencarian` berfungsi untuk melakukan deklarasi c*lass*. Pada baris 2 – 8 terdapat inisialisasi variabel yang didapatkan dari halaman sebelumnya. Lalu pada baris kode program 9 – 10 berfungsi untuk melakukan perubahan tipe data `jumlahKendaraanPencarian` yang semula bertipe data string menjadi integer. Selanjutnya baris 12 – 13 berfungsi untuk mendeklarasi `LinkedHashSet lhs` yang digunakan untuk mengecek list `kendaraanModel` apabila mempunyai data yang sama didalamnya. Baris 15 – 7 merupakan kode program yang berfungsi untuk mengambil data pada *child* `kendaraan` lalu *child* `kategoriKendaraanPencarian` dengan menggunakan *database reference* `mDatabase`. Ketika kode program tersebut dijalankan dengan menggunakan `addValueEventListener` akan otomatis di *override method* `onDataChange` pada baris 18 dan `onCancelled` pada baris 118 – 199. *Method* `onDataChange` berfungsi untuk medeteksi apabila ada perubahan dalam *child* yang kita panggil. Sedangkan *method* `onCancelled` berfungsi untuk membatalkan *listener* apabila terdapat kesalahan pada *server*. Selanjutnya pada baris 19 – 20 berfungsi untuk melakukan kondisi apabila terdapat data didalam *child* yang sudah kita definisikan. Apabila ada data maka akan di eksekusi kode program baris 20 – 108. Apabila tidak terdapat data, maka akan dijalankan pernyataan pada baris 110 – 115.

Kode program baris 20 – 108 berfungsi untuk melakukan pengecekkan antara *child* `kendaraan` dengan *child* `cekSisaKendaraan` untuk mengetahui kendaraan yang tersedia sesuai dengan tanggal pencarian, kategori dan jumlah

kendaraan yang diinginkan pelanggan. Baris 21 – 22 merupakan perulangan yang berfungsi untuk mendapatkan semua data kendaraan yang ada dalam *child* `kendaraan`. lalu pada baris 23 – 24 merupakan deklarasi variabel `kendaraan` yang digunakan untuk menyimpan semua variabel data kendaraan yang sudah di definisikan pada *class* `KendaraanModel` dan data tersebut sesuai dengan yang terdapat di basis data firebase. Baris 25 – 26 dan 27 – 28 berfungsi untuk melakukan inisialisasi variabel `jmlKendaraan` dan id dengan nilai `jumlahKendaraan` dan `idKendaraan` yang didapatkan dari basis data firebase. Selanjutnya pada baris 31 – 37 merupakan kode program yang berfungsi untuk menyeleksi apakah `idKendaraan` yang sedang di eksekusi pada baris 27 – 28 terdapat pada *child* `cekSisaKendaraan`. Apabila ada, dapat diartikan `idKendaraan` tersebut sudah di pesan. Apabila `idKendaraan` terdapat pada *child* `cekSisa` maka akan dijalankan kode program pada baris 42 – 108. Apabila tidak ada maka akan dijalankan 42 – 82 dan apabila tidak ada akan dijalan kode program pada baris 82 – 92. Langkah selanjutnya apabila `idKendaraan` terdapat pada *child* `cekSisaKendaraan` adalah melakukan seleksi tanggal dan menyeleksi apakah jumlah kendaraan masih tersedia atau tidak. Pada baris 42 – 44 berfungsi untuk mendapatkan seluruh data pada *child* `cekSisaKendaraan` yang mempunyai `idKendaraan` yang sama dengan baris 27 – 28. Selanjutnya baris 45 – 46 berfungsi untuk mendapatkan data dari `cekSisaKendaraan` yang ditampung pada variabel `sisaModel`. Sedangkan kode program baris 47 – 54 berfungsi untuk mendapatkan nilai dari basis data firebase. Pada baris 56 – 58 merupakan kode program yang berfungsi untuk menyeleksi tanggal pencariaan dengan tanggal yang ada di *child* `cekSisaKendaraan`, apabila tanggal sama maka akan djalankan kode program 60 – 68. Pada baris 60 – 68 juga terdapat seleksi kondisi, yaitu pada baris 60 – 61 apabila jumlah sisa kendaraan lebih besar dari jumlah pencarian kendaraan atau sama dengan jumlah pencarian kendaraan, maka kendaraan yang di seleksi tersebut masih tersedia dan ditambahkan ke dalam list `kendaraanModel` untuk ditampilkan. Namun apabila jumlah sisa kendaraan lebih kecil akan di hapus dari list `kendaraanModel`, hal ini terlihat pada baris 65 – 66. Apabila kode program pada baris 56 – 58 tidak memenuhi, maka dapat disimpulkan tanggal pencarian dengan tanggal yang dipesan berdasarkan `idKendaraan` tertentu tidak sama. Maka akan dijalankan kode program baris 70 yang dimana didalam kode program tersebut juga terdapat kondisi untuk mengecek apakah jumlah kendaraan yang tersimpan dalam *child* `kendaraan` lebih besar dari jumlah kendaraan pencarian atau sama dengan jumlah kendaraan pencarian, apabila sama maka akan ditambahkan ke dalam list `kendaraanModel`, apabila tidak akan di hapus dari *list* `kendaraanModel`. Kode program 82 – 92 akan dijalankan apabila kondisi `idKendaraan` yang sedang di cek tidak ada di *child* `cekSisaKendaraan`. Pernyataan dari kondisi ini juga terdiri dari kondisi pada baris 83 – 85 yang menyatakan apabila jumlah kendaraan yang tersimpan lebih besar dari jumlah kendaraan pencarian atau jumlah kendaraan sama dengan jumlah kendaraan pencarian, apabila memenuhi maka kendaraan akan di tambahkan ke list `kendaraanModel`, apabila tidak akan di hapus seperti yang terlihat pada baris 90. Selanjutnya adalah baris 93 –

100 berfungsi untuk menambahkan list `kendaraanModel` ke dalam adapter dan menyeleksi apakah terdapat data yang sama atau tidak, apabila terdapat data yang sama maka akan hanya ditampilkan salah satu saja.

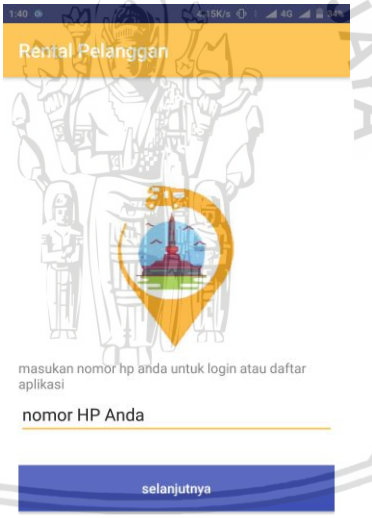
5.2.7 Implementasi Antarmuka

Implementasi antarmuka merupakan bagian yang menjelaskan tentang hasil tampila dari sistem penyewaan kendaraan Kota Malang yang sudah dikembangkan. Pada bagian implementasi antarmuka akan dijelaskan hasil tampilan antarmuka yang telah dihasilkan. Implementasi antarmuka akan di bagi menjadi tiga bagian, yaitu implementasi antarmuka pengguna, antarmuka pelanggan dan antarmuka pemilik rental.

5.2.7.1 Implementasi Antarmuka Sisi Pengguna

Antarmuka pada sisi pengguna berfungsi bagi pengguna untuk masuk atau mendaftar menjadi pelanggan atau pemilik rental. Pada antarmuka sisi pelanggan juga terdapat antarmuka yang berfungsi untuk melakukan pengisian biodata yang berbeda antara sisi pelanggan dan pemilik rental.

a. Halaman Autentifikasi dengan Nomor Telepon



Gambar 5.75 Implementasi Antarmuka Autentifikasi dengan Nomor Telepon

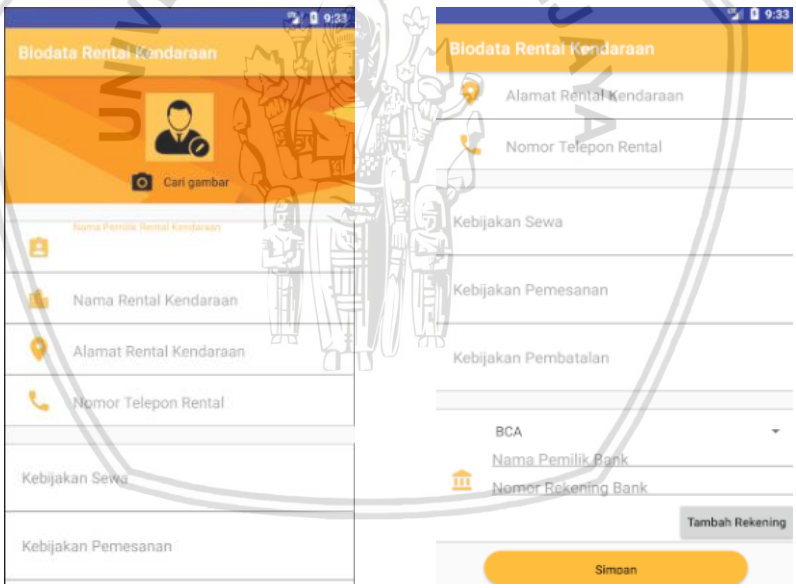
b. Halaman Isi Biodata Pelanggan

Halaman isi biodata pelanggan merupakan halaman yang berfungsi bagi pengguna untuk mengisi biodata sebagai pelanggan pada sistem. Implementasi antarmuka halaman isi biodata pelanggan akan ditunjukkan oleh Gambar 5.76.

Gambar 5.76 Implementasi Antarmuka Halaman Mengisi Biodata Pelanggan

c. Halaman Isi Biodata Rental

Halaman isi biodata rental merupakan halaman yang berfungsi bagi pengguna untuk mengisi biodata sebagai pemilik rental pada sistem. Implementasi antarmuka halaman isi biodata pemilik rental akan ditunjukkan oleh Gambar 5.77.



Gambar 5.77 Implementasi Antarmuka Halaman Mengisi Biodata Rental

5.2.7.2 Implementasi Antarmuka Sisi Pelanggan

Implementasi antarmuka pada sisi pelanggan akan menjelaskan halaman yang disediakan oleh sistem untuk pelanggan agar dapat menggunakan semua fungsionalitas yang telah disediakan. Implementasi antarmuka sisi pelanggan mengacu pada perancangan antarmuka yang sudah dijelaskan sebelumnya. Berikut ini merupakan tampilan antarmuka pada sisi pelanggan.

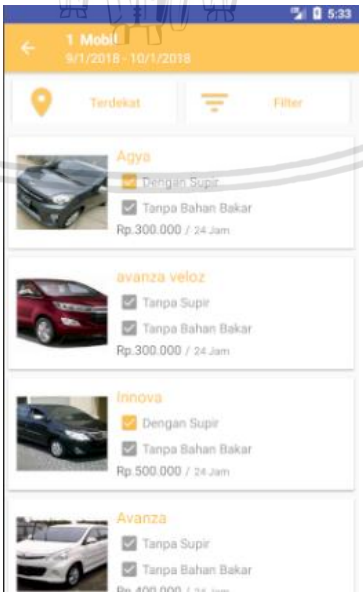
a. Halaman pencarian

Halaman pencarian merupakan halaman yang berfungsi bagi pelanggan untuk melakukan pencarian sesuai dengan tanggal, kategori dan jumlah kendaraan yang pelanggan butuhkan. Implementasi antarmuka halaman pencarian ditunjukkan oleh Gambar 5.78.

Gambar 5.78 Implementasi Antarmuka Pencarian

b. Halaman hasil pencarian

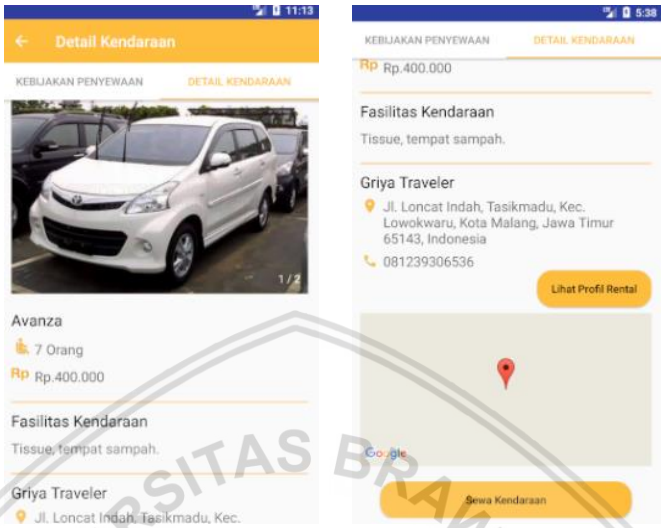
Halaman hasil pencarian merupakan halaman yang berfungsi untuk menampilkan daftar kendaraan yang tersedia sesuai dengan tanggal, kategori dan jumlah kendaraan yang dilakukan pencarian oleh pelanggan. Impelementasi antarmuka halaman hasil pencarian ditunjukkan oleh Gambar 5.79.



Gambar 5.79 Implementasi Antarmuka Halaman Hasil pencarian

c. Halaman detail kendaraan

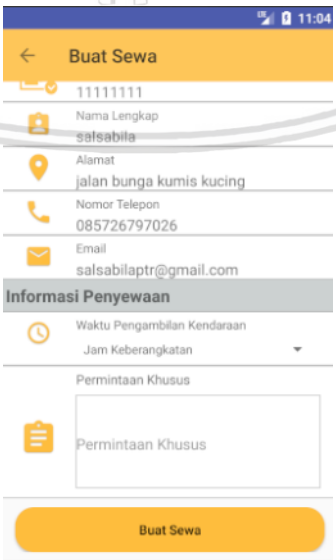
Halaman detail kendaraan merupakan halaman yang berfungsi bagi pelanggan untuk melihat informasi detail kendaraan yang dipilihnya. Implementasi antarmuka halaman detail kendaraan ditunjukkan oleh Gambar 5.80.



Gambar 5.80 Implementasi Antarmuka Halaman Detail Kendaraan

d. Halaman buat penyewaan

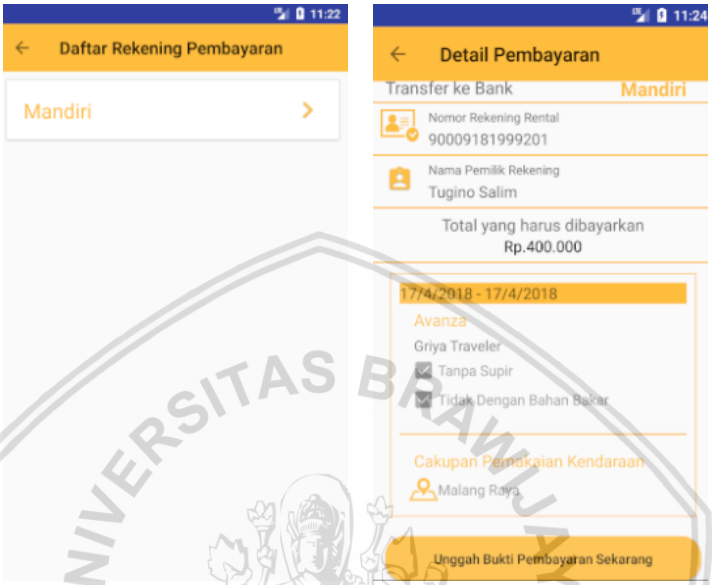
Halaman buat penyewaan merupakan halaman yang berfungsi bagi pelanggan untuk melakukan penyewaan kendaraan. pada halaman ini jug pelanggan akan ditampilkan sebuah *maps* untuk menunjukkan lokasi penjemputan apabila kendaraan yang dipilihnya menyediakan fasilitas sopir. Tapi apabila kendaraan yang dipilihnya tidak menyediakan fasilitas sopir maka input lokasi penjemputan tidak akan ditampilkan. Implementasi antarmuka halaman buat penyewaan ditunjukkan oleh Gambar 5.81.



Gambar 5.81 Implementasi Antarmuka Halaman Buat Penyewaan

e. Halaman pembayaran

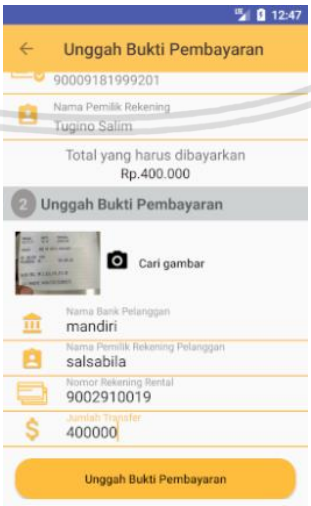
Halaman pembayaran merupakan halaman yang digunakan oleh pelanggan untuk melakukan pembayaran terhadap penyewaan yang dibuat. Pada halaman pembayaran pelanggan akan ditampilkan daftar rekening pembayaran yang tersedia dan setelah itu akan ditampilkan halaman detail pembayaran. Implementasi antarmuka halaman pembayaran ditunjukkan oleh Gambar 5.82.



Gambar 5.82 Implementasi Antarmuka Halaman Pembayaran

f. Halaman unggah bukti pembayaran

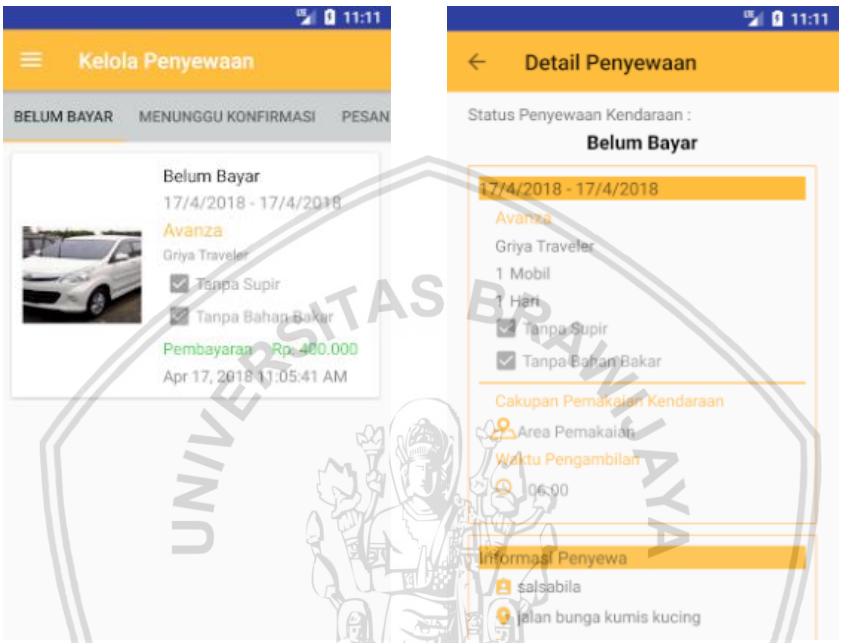
Halaman unggah bukti pembayaran berfungsi bagi pelanggan untuk dapat melakukkan unggah foto bukti pembayaran ke dalam sistem untuk di konfirmasi oleh pemilik rental. Implementasi halaman unggah bukti pembayaran ditunjukkan oleh Gambar 5.83.



Gambar 5.83 Implementasi Antarmuka Halaman Unggah Bukti Pembayaran

g. Halaman kelola penyewaan

Halaman kelola penyewaan merupakan halaman yang berfungsi bagi pelanggan untuk melihat status penyewaan, lihat detail penyewaan, unggah bukti pembayaran, melakukan pembatalan, memberikan penilaian dan melihat penilaian. Halaman kelola penyewaan terdiri dari dua buah halaman untuk menampilkan daftar penyewaan sesuai dengan status penyewaan masing-masing dan menampilkan informasi detail dari penyewaan pelanggan. Status penyewaan dibedakan kedalam *tab menu* seperti yang ditunjukkan oleh Gambar 5.84.



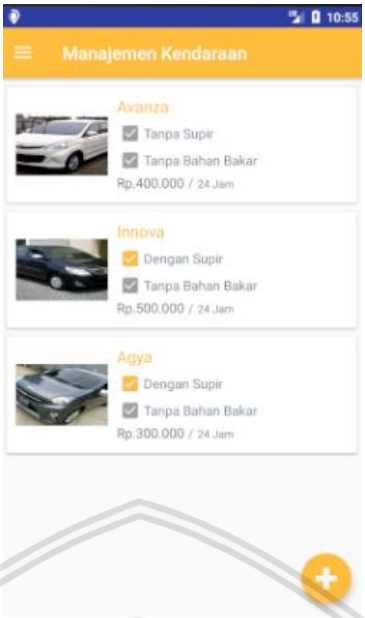
Gambar 5.84 Implementasi Antarmuka Halaman Kelola Penyewaan

5.2.7.3 Implementasi Antarmuka Sisi Pemilik rental

Implementasi antarmuka pada sisi pemilik rental akan menjelaskan halaman yang disediakan oleh sistem untuk pemilik rental agar dapat menggunakan semua fungsionalitas yang telah disediakan. Implementasi antarmuka sisi pemilik rental mengacu pada perancangan antarmuka yang sudah dijelaskan sebelumnya. Berikut ini merupakan tampilan antarmuka pada sisi pemilik rental.

a. Halaman manajemen kendaraan

Halaman manajemen kendaraan merupakan halaman yang menampilkan daftar kendaraan rental yang tersimpan dalam sistem. Implementasi antarmuka halaman manajemen kendaraan ditunjukkan oleh Gambar 5.85.



Gambar 5.85 Implementasi Antarmuka Halaman Manajemen Kendaraan

b. Halaman detail kendaraan

Halaman detail kendaraan merupakan halaman yang berfungsi bagi pemilik rental untuk melihat informasi detail kendaraan yang tersimpan. Implementasi antarmuka halaman detail kendaraan ditunjukkan oleh Gambar 5.86.

Gambar 5.86 Implementasi Antarmuka Halaman Detail Kendaraan

c. Halaman tambah kendaraan

Halamana tambah kendaraan merupakan halaman yang berfungsi bagi pemilik rental untuk melakukan penambahan kendaraan pda sistem. Implementasi antarmuka halaman tambah kendaraan ditunjukkan oleh Gambar 5.87.

Gambar 5.87 Implementasi Antarmuka Halaman Tambah Kendaraan

d. Halaman lihat profil

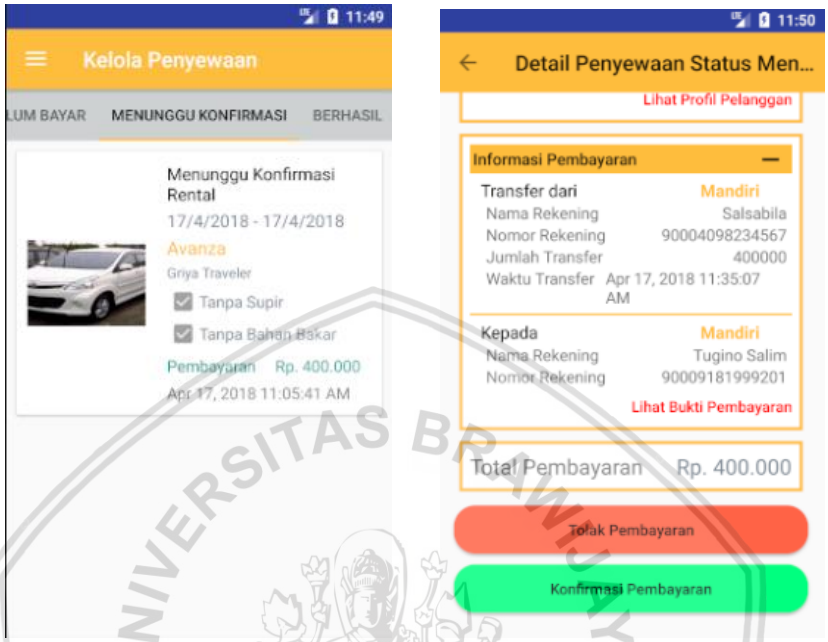
Halaman lihat profil merupakan halaman yang berfungsi bagi pemilik rental untuk melihat profilnya. Implementasi antarmuka halaman lihat profil ditunjukkan oleh Gambar 5.88.

Gambar 5.88 Implementasi Antarmuka Halaman Lihat Profil

e. Halaman kelola penyewaan

Halaman kelola penyewaan merupakan halaman yang berfungsi bagi pemilik rentl untuk melihat status penyewaan, lihat detail penyewaan, melakukan konfirmasi pembayaran, melakukan konfirmasi penyewaan selesai, unggah bukti

pengembalian dana, melakukan konfirmasi pembatalan, dan melihat penilaian. Halaman kelola penyewaan terdiri dari dua buah halaman untuk menampilkan daftar penyewaan sesuai dengan status penyewaan masing-masing dan menampilkan informasi detail dari penyewaan pelanggan. Status penyewaan dibedakan kedalam *tab menu* seperti yang ditunjukkan oleh Gambar 5.89.



Gambar 5.89 Implementasi Antarmuka Kelola Penyewaan

BAB 6 PENGUJIAN

Pada bab ini akan dibahas tentang tahapan pengujian sistem penyewaan kendaraan kota Malang yang telah dikembangkan. Pengujian yang dilakukan pada penelitian ini terdiri dari dua bagian yaitu pengujian fungsional dan pengujian non-fungsional. Pengujian fungsional merupakan pengujian yang dilakukan untuk menguji kebutuhan fungsional pada sistem. Sedangkan pengujian non-fungsional dilakukan untuk menguji kebutuhan non-fungsional pada sistem. Proses pengujian fungsional yang dilakukan pada penelitian ini meliputi empat strategi pengujian yaitu pengujian unit, pengujian integrasi, pengujian validasi dan pengujian *acceptence.*

Pada metode pengembangan MASAM terdapat *task* tentang pengujian sistem yang di adopsi, yaitu pada fase *commercialization* dengan aktivitas *system test* yang terdiri dari *task acceptance test*. Pada bagian ini akan dilakukan pengujian *acceptance* dengan parameter u*sability* yang berhubungan dengan kebutuhan non-fungsional sistem.

6.1 Pengujian Fungsional

Pengujian Fungsional bertujuan untuk menguji seluruh kebutuhan fungsional yang telah dikembangkan dalam sistem. Dalam pengujian fungsional terbagi menjadi pengujian unit, pengujian integrasi, pengujian validasi dan pengujian *acceptance*. Sesuai dengan metodologi penelitian yang di adopsi, pada tahapan pengujian unit dan integrasi akan dilakukan pengujian terhadap kebutuhan utama yang sesuai dengan pengembangan iterasi 1.

6.1.1 Pengujian Unit

Pengujian unit merupakan suatu teknik pengujian yang dilakukan pada penelitian ini dengan menggunakan teknik *basis path testing* untuk menguji kode program berdasarkan algoritma pada setiap metode yang ada di klas. Proses pengujian yang dilakukan adalah dengan memodelkan algoritma menjadi suatu *flow graph* dan mencari jumlah kompleksitas siklomatis (*cyclomatic complexity*) untuk menentukan kompleksitas logika dari sebuah program, menentukan jalur independen dan memberikan kasus uji.

6.1.1.1 Pengujian Unit cekTanggal

Method `cekTanggal` merupakan suatu *method* yang berfungsi untuk melakukan pengecekkan jika terdapat tanggal sewa dan tanggal kembali pencarian yang mempunyai nilai sama dengan tanggal sewa dan tanggal kembali penyewaan yang sudah ada. *Method* `cekTanggal` mempunyai nilai kembalian dengan tipe data boolean dengan nilai kembalian *true* atau *false*. Gambar 6.1 menunjukkan kode program dari *method* `cekTanggal`.


```
public boolean cekTanggal(String tanggalSewaPencarian,
String tanggalKembaliPencarian, String
tanggalSewaDipesan, String tanggalKembaliDipesan) {
    SimpleDateFormat format = new
SimpleDateFormat("dd/MM/yyyy");
    try {
        tglSewaPencarian =
format.parse(tanggalSewaPencarian) ;
        tglKembaliPencarian =
format.parse(tanggalKembaliPencarian);
        tglSewaDipesan =
format.parse(tanggalSewaDipesan);
        tglKembaliDipesan =
format.parse(tanggalKembaliDipesan);

        if ((tglSewaPencarian.before(tglKembaliDipesan)
||

tglSewaPencarian.equals(tglKembaliDipesan)) &&
(tglKembaliPencarian.after(tglSewaDipesan) ||

tglKembaliPencarian.equals(tglSewaDipesan)) ||
                tglSewaPencarian.equals(tglSewaDipesan)
&& tglKembaliPencarian.equals(tglKembaliDipesan)) {
            cekTanggal = true;
        } else {
            cekTanggal = false;
        }
    } catch (ParseException e) {
        Toast.makeText(getApplicationContext(), "Proses
Pencarian Gagal Karena Terdapat Kesalahan Sistem",
Toast.LENGTH_LONG).show();

    }
    return cekTanggal;
}
```

Gambar 6.1 Kode Program *Method* cekTanggal

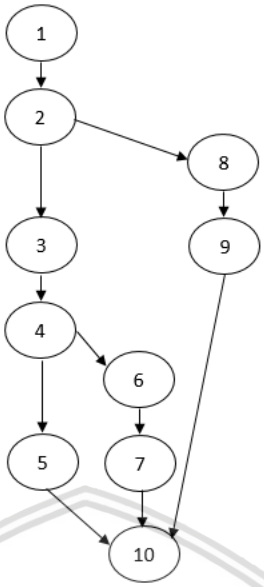
Berdasarkan proses pengujian tersebut, Gambar 6.2 menunjukkan *flow graph* dari *method* `cekTanggal` yang berfungsi menggambarkan alur dari method tersebut.

Tahapan selanjutnya dari pengujian unit *method* `cekTanggal` yaitu tahapan untuk menghitung *cyclomatic complexity*. Berdasarkan *flow graph method* `cekTanggal`, didapatkan perhitungan *cyclomatic complexity* dengan menggunakan persamaan V(G) = E – N + 2, dimana V(G) merupakan jumlah kompleksitas siklomatis, lalu E *(edge*) merupakan garis penghubung antar node dan N merupakan jumlah simpul *(node).*

V(G) = 11 – 10 + 2

= 1 + 2

= 3



Gambar 6.2 *Flow Graph Method* cekTanggal

Setelah mendapatkan nilai dari *cyclomatic complexity*, maka didapatkan tiga buah basis set jalur *independent*, yaitu :

Jalur 1 : 1 – 2 – 3 – 4 – 5 – 10

Jalur 2 : 1 – 2 – 3 – 4 – 6 – 7 – 10

Jallur 3 : 1 – 2 – 8 – 9 – 10

Berdasarkan hasil dari banyaknya jalur *independent*, maka dapat dilakukan pengujian terhadap tiap jalur tersebut dengan kasus uji yang ditunjukkan pada Tabel 6.1.

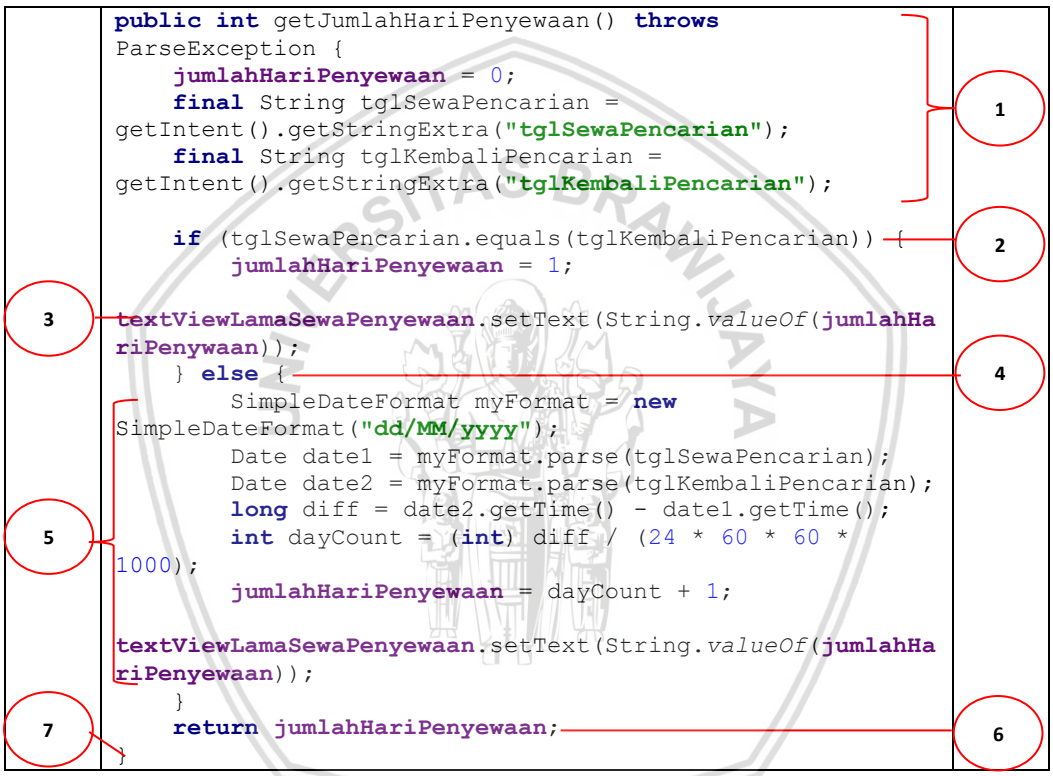
Tabel 6.1 Kasus Uji Pengujian Unit *Method* cekTanggal

Jalur	Kasus Uji	Hasil yang Diharapkan	Hasil yang Didapatkan
1	Ketika tanggal sewa pencarian mempunyai nilai yang sama dengan tanggal kembali pencarian.	*Method* menginisialisasi nilai kembalian variabel boolean cekTanggal dengan nilai true.	*Method* menginisialisasi nilai kembalian variabel boolean cekTanggal dengan nilai true.
2	Ketika tanggal sewa pencarian mempunyai nilai yang berbeda dengan tanggal kembali pencarian.	*Method* menginisialisasi nilai kembalian variabel boolean cekTanggal dengan nilai false.	*Method* menginisialisasi nilai kembalian variabel boolean cekTanggal dengan nilai false.

3	Ketika terdapat kesalahan sistem.	Menampilkan pesan peringatan bahwa proses pencarian Gagal.	Menampilkan pesan peringatan bahwa proses pencarian Gagal.

6.1.1.2 Pengujian Unit getJumlahHariPenyewaan

Method `getJumlahHariPenyewaan` merupakan suatu *method* yang digunakan dalam sistem untuk mengetahui jumlah hari penyewaan kendaraan yang didapatkan dari kalkukasi antara tanggal sewa dan tanggal kembali. Gambar 6.3 dibawah ini menunjukkan kode program dari *method* `getJumlahHariPenyewaan`.




```
public int getJumlahHariPenyewaan() throws
ParseException {
    jumlahHariPenyewaan = 0;
    final String tglSewaPencarian =
getIntent().getStringExtra("tglSewaPencarian");
    final String tglKembaliPencarian =
getIntent().getStringExtra("tglKembaliPencarian");

    if (tglSewaPencarian.equals(tglKembaliPencarian)) {
        jumlahHariPenyewaan = 1;

textViewLamaSewaPenyewaan.setText(String.valueOf(jumlahHa
riPenywaan));
    } else {
        SimpleDateFormat myFormat = new
SimpleDateFormat("dd/MM/yyyy");
        Date date1 = myFormat.parse(tglSewaPencarian);
        Date date2 = myFormat.parse(tglKembaliPencarian);
        long diff = date2.getTime() - date1.getTime();
        int dayCount = (int) diff / (24 * 60 * 60 *
1000);
        jumlahHariPenyewaan = dayCount + 1;

textViewLamaSewaPenyewaan.setText(String.valueOf(jumlahHa
riPenyewaan));
    }
    return jumlahHariPenyewaan;
}
```

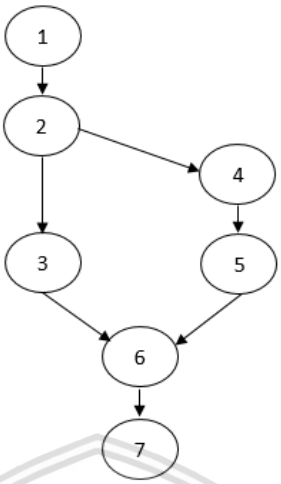
Gambar 6.3 Kode Program *Method* getJumlahHariPenyewaan

Berdasarkan proses pengujian tersebut, Gambar 6.4 menunjukkan *flow graph* dari *method* `getJumlahHariPenyewaan` yang berfungsi menggambarkan alur dari *method* tersebut.

Tahapan selanjutnya dari pengujian unit method getJumlahHariPenyewaan yaitu tahapan untuk menghitung cyclomatic complexity. Berdasarkan flow graph method getJumlahHariPenyewaan, didapatkan perhitungan cyclomatic complexity dengan menggunakan persamaan V(G) = E – N + 2, dimana V(G) merupakan jumlah kompleksitas siklomatis, lalu E *(edge*) merupakan garus penghubung antar node dan N merupakan jumlah simpul *(node).*

V(G) = 7 – 7 + 2

= 2



Gambar 6.4 *Flow Graph Method* getJumlahHariPenyewaan

Setelah mendapatkan nilai dari cyclomatic complexity, maka didapatkan dua buah basis set jalur *independent*, yaitu :

Jalur 1 : 1 – 2 – 3 – 6 – 7

Jalur 2 : 1 – 2 – 4 – 5 – 6 – 7

Berdasarkan hasil dari banyaknya jalur independent, maka dapat dilakukan pengujian terhadap tiap jalur tersebut dengan kasus uji yang ditunjukkan pada Tabel 6.2.

Tabel 6.2 Kasus Uji Pengujian Unit *Method* getJumlahHariPenyewaan

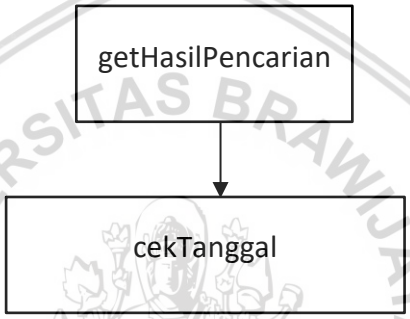
Jalur	Kasus Uji	Hasil yang Diharapkan	Hasil yang Didapatkan
1	Ketika tanggal sewa pencarian mempunyai nilai yang sama dengan tanggal kembali pencarian.	*Method* menginisialisasi nilai *return* variabel jumlahHariPenyewaan dengan nilai 1.	Method menginisialisasi nilai *return* variabel jumlahHariPenyewaan dengan nilai 1.
2	Ketika tanggal sewa pencarian mempunyai nilai yang berbeda dengan tanggal kembali pencarian.	Method melakukan perhitungan untuk mendapatkan nilai variabel jumlahHariPenyewaan dan melakukan *return* variable jumlahHariPenyewaan.	Method melakukan perhitungan untuk mendapatkan nilai variabel jumlahHariPenyewaan dan melakukan *return* variable jumlahHariPenyewaan.

6.1.2 Pengujian Integrasi

Pengujian integrasi merupakan pengujian yang dilakukan ketika ada proses integrasi dari beberapa *method* atau *class* untuk melakukan suatu operasi tertentu. Pada pengujian Integrasi akan digunakan pendeketan *top down* untuk melakukan pengujian integrasi.

6.1.2.1 Pengujian Integrasi method getHasilPencarian

Method `getHasilPencarian` merupakan salah satu *method* utama yang berfungsi untuk melakukan pencarian kendaraan berdasarkan tanggal sewa, tanggal kembali, kategori kendaraan dan jumlah kendaraan. *Method* ini terintegrasi dengan *method* `cekTanggal` yang sebelumnya sudah dilakukan pengujian pada sub bab pengujian unit. Diagram hirarki dari dua *method* yang saling terintegrasi tersebut digambarkan oleh Gambar 6.5.



Gambar 6.5 Hirarki Pengujian Integrasi *Method* getHasilPencarian

Strategi pengujian integrasi yang akan digunakan untuk menguji method integrasi `getHasilPencarian` adalah menggunakan pendekatan *Top Down*. Pada pendekatan ini pada *method* `getHasilPencarian` akan di buat sebuah *stub* yang berfungsi sebagai modul pengganti yang akan memanggil modul yang sedang di uji. Langkah – langkah dalam pengujian integrasi *method* `getHasilPencarian` seperti berikut ini :

Langkah 1 : IS = `getHasilPencarian` + `cekTanggal`

Pada pengujian integrasi *method* `getHasilPencarian` hanya terdiri dari 1 langkah saja. Hal ini karenakan hanya terdapat 1 *method* yang terintegrasi dengan *method* `getHasilPencarian`. Stub yang akan digunakan pada pengujian *method* `getHasilPencarian` diberikan nilai *true* atau *false* yang merupakan tipe data boolean. Gambar 6.6 merupakan kode program *method* `getHasilPencarian` yang menggunakan *stub* dengan nilai true sebagai modul pengganti nilai kembalian *method* `cekTanggal`.

```
public void getHasilPencarian() {
    final String kategoriKendaraanPencarian =
getIntent().getStringExtra("kategoriKendaraanPencarian");
    final String jumlahKendaraanPencarian =
getIntent().getStringExtra("jumlahKendaraanPencarian");
    final String tanggalSewaPencarian =
getIntent().getStringExtra("tglSewaPencarian");
    final String tanggalKembaliPencarian =
getIntent().getStringExtra("tglKembaliPencarian");
```

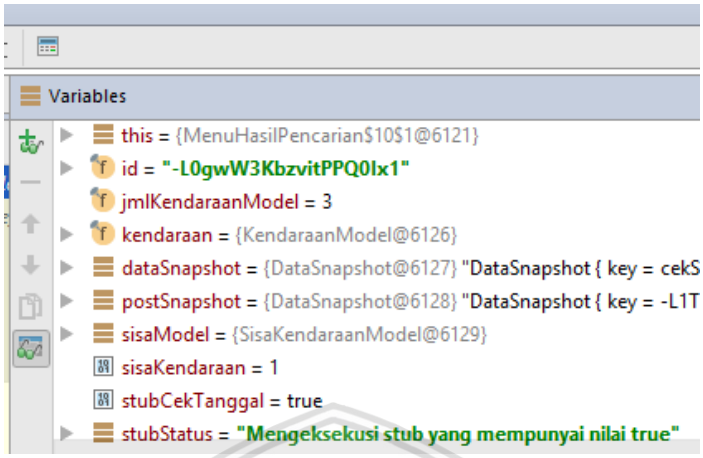
```
    jmlKendaraanPencarian =
Integer.parseInt(jumlahKendaraanPencarian);
    final LinkedHashSet<KendaraanModel> lhs = new
LinkedHashSet<>();
mDatabase.child("kendaraan").child(kategoriKendaraanPenca
rian).addValueEventListener(new ValueEventListener() {
            @Override
            public void onDataChange(DataSnapshot
dataSnapshot) {
                if (dataSnapshot.exists()) {
                    progressBar.setVisibility(View.GONE);
                    for (DataSnapshot postSnapshot :
dataSnapshot.getChildren()) {
                        final KendaraanModel kendaraan =
postSnapshot.getValue(KendaraanModel.class)
                        jmlKendaraan =
kendaraan.getJumlahKendaraan();
                        final String id =
kendaraan.getIdKendaraan();
                        final int jmlKendaraanModel =
jmlKendaraan;

                        Firebase ref = new
Firebase("https://bismillahskripsi-
44a73.firebaseio.com/cekSisaKendaraan");
                        Query query =
ref.orderByChild("idKendaraan").equalTo(id);
                        query.addValueEventListener(new
com.firebase.client.ValueEventListener() {
                            @Override
                            public void
onDataChange(com.firebase.client.DataSnapshot
dataSnapshot) {
                                if
(dataSnapshot.exists()) {
                                    for
(com.firebase.client.DataSnapshot postSnapshot :
dataSnapshot.getChildren()) {

SisaKendaraanModel sisaModel =
postSnapshot.getValue(SisaKendaraanModel.class);
                                        idCekSisa =
sisaModel.getIdCekSisa();
                                        final int
sisaKendaraan = sisaModel.getSisaKendaraan();


boolean stubCekTanggal = true;

                                    if (stubCekTanggal) {
                                        String
stubStatus = "Mengeksekusi stub yang mempunyai nilai
true";
                                        if
(sisaKendaraan >= jmlKendaraanPencarian ||
jmlKendaraanModel == jmlKendaraanPencarian) {

kendaraanModel.add(kendaraan);
                                        } else {

kendaraanModel.remove(kendaraan);
                                            break;
                                        }
                                    } else {
                                        String
```

```
stubStatus = "Mengeksekusi stub yang mempunyai nilai
false";
                                                    if
(jmlKendaraanModel >= jmlKendaraanPencarian ||
jmlKendaraanModel == jmlKendaraanPencarian) {

kendaraanModel.add(kendaraan);
                                                    } else {

kendaraanModel.remove(kendaraan);
                                                        break;
                                                    }
                                                }
                                            }
                                        } else {
                                            if (jmlKendaraanModel
> jmlKendaraanPencarian || jmlKendaraanModel ==
jmlKendaraanPencarian) {

kendaraanModel.add(kendaraan);
                                        } else {

kendaraanModel.remove(kendaraan);
                                        }
                                    }

lhs.addAll(kendaraanModel);
                                kendaraanModel.clear();

kendaraanModel.addAll(lhs);
                                adapter = new
MenuHasilPencarianAdapter(MenuHasilPencarian.this,
kendaraanModel, tanggalSewaPencarian,
tanggalKembaliPencarian, jumlahKendaraanPencarian);

recyclerView.setAdapter(adapter);
                            }


                            @Override
                            public void
onCancelled(FirebaseError firebaseError) {
                            }
                        });
                    }
                } else {
                    progressBar.setVisibility(View.GONE);

linearLayoutListKendaraan.setVisibility(View.GONE);

kendaraanTidakTersedia.setVisibility(View.VISIBLE);
                }
            }
            @Override
            public void onCancelled(DatabaseError
databaseError) {
            }
        });
```

Gambar 6.6 Kode Program *Method* getHasilPencarian dengan menggunakan Stub bernilai *True*

Hasil dari penggunaan *stub* dengan nilai variabel `stubCekTanggal` *true* adalah *method* `getHasilPencarian` berhasil mengeksekusi nilai String

`stubStatus` yang merupakan *statement* pada kondisi jika `stubCekTanggal` bernilai *true*. Hal ini ditunjukkan oleh Gambar 6.7.



Gambar 6.7 Hasil Penggunaan Stub pada *Method* getHasilPencarian

Selanjutnya adalah mengganti nilai `stubCekTanggal` dengan nilai false sebagai modul pengganti nilai kembalian *method* `cekTanggal`. Gambar 6.8 menunjukkan penggunaan *stub* dengan nilai *stub* sama dengan *false.*

```
public void getHasilPencarian() {
    final String kategoriKendaraanPencarian =
getIntent().getStringExtra("kategoriKendaraanPencarian");
    final String jumlahKendaraanPencarian =
getIntent().getStringExtra("jumlahKendaraanPencarian");
    final String tanggalSewaPencarian =
getIntent().getStringExtra("tglSewaPencarian");
    final String tanggalKembaliPencarian =
getIntent().getStringExtra("tglKembaliPencarian");
    jmlKendaraanPencarian =
Integer.parseInt(jumlahKendaraanPencarian);
    final LinkedHashSet<KendaraanModel> lhs = new
LinkedHashSet<>();
mDatabase.child("kendaraan").child(kategoriKendaraanPenca
rian).addValueEventListener(new ValueEventListener() {
            @Override
            public void onDataChange(DataSnapshot
dataSnapshot) {
                if (dataSnapshot.exists()) {
                    progressBar.setVisibility(View.GONE);
                    for (DataSnapshot postSnapshot :
dataSnapshot.getChildren()) {
                        final KendaraanModel kendaraan =
postSnapshot.getValue(KendaraanModel.class)
                        jmlKendaraan =
kendaraan.getJumlahKendaraan();
                        final String id =
kendaraan.getIdKendaraan();
                        final int jmlKendaraanModel =
jmlKendaraan;

                        Firebase ref = new
Firebase("https://bismillahskripsi-
44a73.firebaseio.com/cekSisaKendaraan");
                        Query query =
ref.orderByChild("idKendaraan").equalTo(id);
                        query.addValueEventListener(new
com.firebase.client.ValueEventListener() {
```

```
                                @Override
                                public void
onDataChange(com.firebase.client.DataSnapshot
dataSnapshot) {
                                    if
(dataSnapshot.exists()) {
                                        for
(com.firebase.client.DataSnapshot postSnapshot :
dataSnapshot.getChildren()) {

SisaKendaraanModel sisaModel =
postSnapshot.getValue(SisaKendaraanModel.class);
                                            idCekSisa =
sisaModel.getIdCekSisa();
                                            final int
sisaKendaraan = sisaModel.getSisaKendaraan();


boolean stubCekTanggal = false;

                                    if (stubCekTanggal) {
                                            String
stubStatus = "Mengeksekusi stub yang mempunyai nilai
true";
                                        if
(sisaKendaraan >= jmlKendaraanPencarian ||
jmlKendaraanModel == jmlKendaraanPencarian) {

kendaraanModel.add(kendaraan);
                                        } else {

kendaraanModel.remove(kendaraan);
                                            break;
                                        }
                                    } else {
                                            String
stubStatus = "Mengeksekusi stub yang mempunyai nilai
false";
                                        if
(jmlKendaraanModel >= jmlKendaraanPencarian ||
jmlKendaraanModel == jmlKendaraanPencarian) {

kendaraanModel.add(kendaraan);
                                        } else {

kendaraanModel.remove(kendaraan);
                                            break;
                                        }
                                    }
                                } else {
                                    if (jmlKendaraanModel
> jmlKendaraanPencarian || jmlKendaraanModel ==
jmlKendaraanPencarian) {

kendaraanModel.add(kendaraan);
                                    } else {

kendaraanModel.remove(kendaraan);
                                    }
                                }

lhs.addAll(kendaraanModel);
                                kendaraanModel.clear();
```

```
kendaraanModel.addAll(lhs);
                                    adapter = new
MenuHasilPencarianAdapter(MenuHasilPencarian.this,
kendaraanModel, tanggalSewaPencarian,
tanggalKembaliPencarian, jumlahKendaraanPencarian);

recyclerView.setAdapter(adapter);
                                }

                                @Override
                                public void
onCancelled(FirebaseError firebaseError) {
                                }
                            });
                        }
                    } else {
                        progressBar.setVisibility(View.GONE);

linearLayoutListKendaraan.setVisibility(View.GONE);

kendaraanTidakTersedia.setVisibility(View.VISIBLE);
                    }
                }
                @Override
                public void onCancelled(DatabaseError
databaseError) {
                }
            });
```

Gambar 6.8 Kode Program *Method* getHasilPencarian dengan menggunakan Stub bernilai *False*

Hasil dari penggunaan *stub* dengan nilai variabel `stubCekTanggal` *false* adalah *method* `getHasilPencarian` berhasil mengeksekusi nilai String `stubStatus` yang merupakan *statement* pada kondisi jika `stubCekTanggal` bernilai *false*. Hal ini ditunjukkan oleh Gambar 6.9.

Gambar 6.9 Hasil Penggunaan Stub pada *Method* getHasilPencarian

Selanjutnya adalah dengan mengintegrasikan *method* `getHasilPencarian` dengan *method* `cekTanggal`. Gambar 6.10 dibawah ini akan menjelaskan tentang kode program `getHasilPencarian` yang telah di integrasikan dengan *method* `cekTanggal`.

```
public void getHasilPencarian() {
    final String kategoriKendaraanPencarian =
getIntent().getStringExtra("kategoriKendaraanPencarian");
    final String jumlahKendaraanPencarian =
getIntent().getStringExtra("jumlahKendaraanPencarian");
    final String tanggalSewaPencarian =
getIntent().getStringExtra("tglSewaPencarian");
    final String tanggalKembaliPencarian =
getIntent().getStringExtra("tglKembaliPencarian");
    jmlKendaraanPencarian =
Integer.parseInt(jumlahKendaraanPencarian);
    final LinkedHashSet<KendaraanModel> lhs = new
LinkedHashSet<>();

mDatabase.child("kendaraan").child(kategoriKendaraanPenca
rian).addValueEventListener(new ValueEventListener() {

        @Override
        public void onDataChange(DataSnapshot
dataSnapshot) {
            if (dataSnapshot.exists()) {
                progressBar.setVisibility(View.GONE);
                for (DataSnapshot postSnapshot :
dataSnapshot.getChildren()) {
                    final KendaraanModel kendaraan =
postSnapshot.getValue(KendaraanModel.class);
                    jmlKendaraan =
kendaraan.getJumlahKendaraan();
                    final String id =
kendaraan.getIdKendaraan();
                    final int jmlKendaraanModel =
jmlKendaraan;

                    Firebase ref = new
Firebase("https://bismillahskripsi-
44a73.firebaseio.com/cekSisaKendaraan");
                    Query query =
ref.orderByChild("idKendaraan").equalTo(id);
                    query.addValueEventListener(new
com.firebase.client.ValueEventListener() {
                        @Override
                        public void
onDataChange(com.firebase.client.DataSnapshot
dataSnapshot) {
                            if (dataSnapshot.exists()) {
                                for
(com.firebase.client.DataSnapshot postSnapshot :
dataSnapshot.getChildren()) {
                                    SisaKendaraanModel
sisaModel =
postSnapshot.getValue(SisaKendaraanModel.class);
                                    idCekSisa =
sisaModel.getIdCekSisa();
                                    final int
sisaKendaraan = sisaModel.getSisaKendaraan();
                                    String
tanggalSewaDipesan = sisaModel.getTglSewa();
                                    String
tanggalKembaliDipesan = sisaModel.getTglKembali();


                                    if
(cekTanggal(tanggalSewaPencarian,
tanggalKembaliPencarian, tanggalSewaDipesan,
tanggalKembaliDipesan)) {
```

1
2
3
4
5
6
7
8
9
10

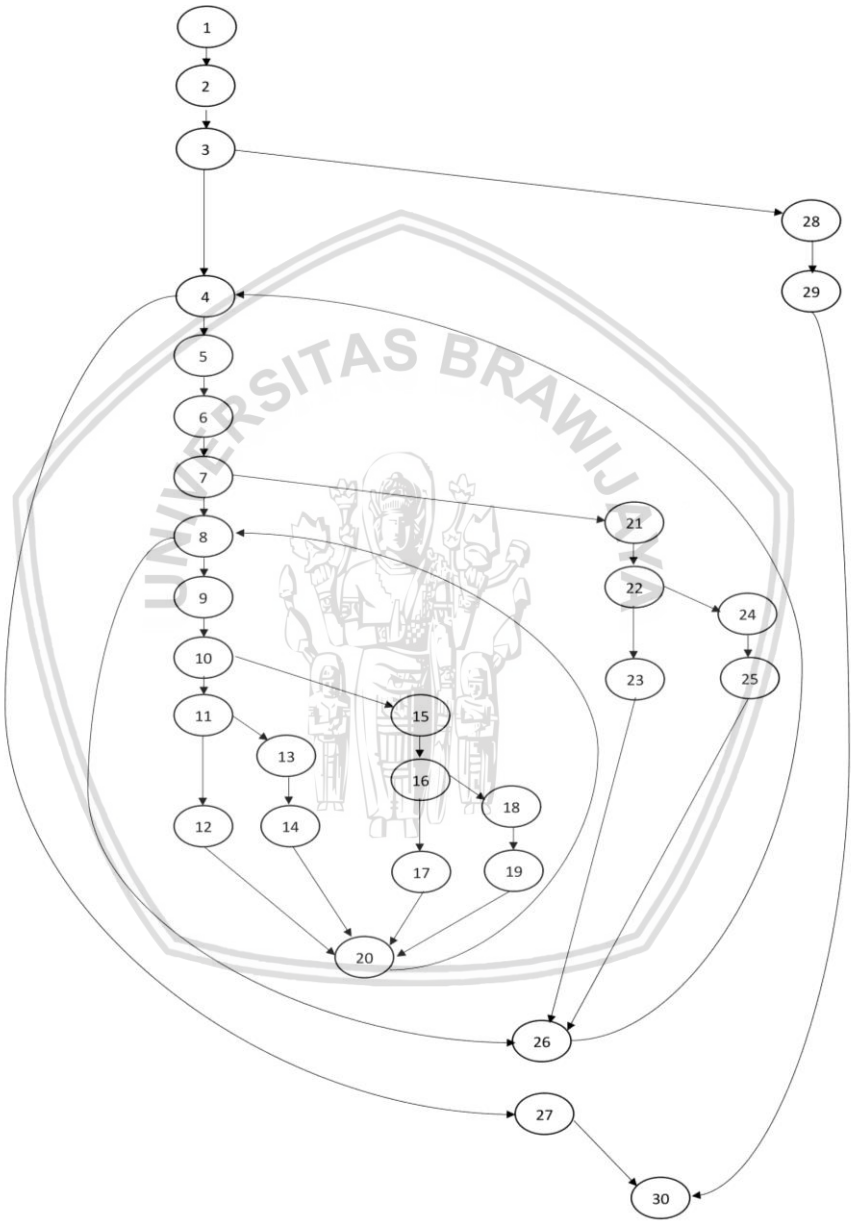
```
                                        if (sisaKendaraan
>= jmlKendaraanPencarian || jmlKendaraanModel ==
jmlKendaraanPencarian) {

kendaraanModel.add(kendaraan);
                                        } else {

kendaraanModel.remove(kendaraan);
                                            break;
                                        }
                                    } else {
                                        if
(jmlKendaraanModel >= jmlKendaraanPencarian ||
jmlKendaraanModel == jmlKendaraanPencarian) {

kendaraanModel.add(kendaraan);
                                    } else {

kendaraanModel.remove(kendaraan);
                                            break;
                                        }
                                    }
                                }
                            } else {
                                if (jmlKendaraanModel >
jmlKendaraanPencarian || jmlKendaraanModel ==
jmlKendaraanPencarian) {

kendaraanModel.add(kendaraan);
                                } else {

kendaraanModel.remove(kendaraan);
                                    }
                                }
                                lhs.addAll(kendaraanModel);
                                kendaraanModel.clear();
                                kendaraanModel.addAll(lhs);
                                adapter = new
MenuHasilPencarianAdapter(MenuHasilPencarian.this,
kendaraanModel, tanggalSewaPencarian,
tanggalKembaliPencarian, jumlahKendaraanPencarian);

recyclerView.setAdapter(adapter);
                            }

                            @Override
                            public void
onCancelled(FirebaseError firebaseError) {
                            }
                        });
                    }
                } else {
                    progressBar.setVisibility(View. GONE);

linearLayoutListKendaraan.setVisibility(View.GONE);

kendaraanTidakTersedia.setVisibility(View.VISIBLE);
                }
            }
            @Override
            public void onCancelled(DatabaseError
databaseError) {
            }
        });
```

11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29

	}	30

Gambar 6.10 Kode Program *Method* getHasilPencarian

Berdasarkan proses pengujian tersebut, Gambar 6.11 menunjukkan *flow graph* dari *method* `getHasilPencarian` yang berfungsi menggambarkan alur dari *method* tersebut.



Gambar 6.11 *Flow Graph Method* getHasilPencarian

Tahapan selanjutnya dari pengujian unit *method* `getHasilPencarian` yaitu tahapan untuk menghitung `cyclomatic complexity`. Berdasarkan *flow graph method* `getHasilPencarian`, didapatkan perhitungan *cyclomatic complexity* dengan menggunakan persamaan V(G) = E – N + 2, dimana V(G)

merupakan jumlah kompleksitas siklomatis, lalu E *(edge*) merupakan garis penghubung antar node dan N merupakan jumlah simpul *(node).*

V(G) = E – N + 2

= 37 – 30 + 2

= 9

Setelah mendapatkan nilai dari *cyclomatic complexity*, maka didapatkan sembilan buah basis set jalur *independent*, yaitu :

Jalur 1 : 1 – 2 – 3 – 4 – 27 – 30

Jalur 2 : 1 – 2 – 3 – 4 – 5 – 6 – 7 – 8 – 26 – 4 – 27 – 30

Jalur 3 : 1 – 2 – 3 – 4 – 5 – 6 – 7 – 8 – 9 – 10 – 11 – 12 – 20 – 8 – 26 – 4 – 27 - 30

Jalur 4 : 1 – 2 – 3 – 4 – 5 – 6 – 7 – 8 – 9 – 10 – 11 – 13 – 14 – 20 – 8 – 26 – 4 – 27 - 30

Jalur 5 : 1 – 2 – 3 – 4 – 5 – 6 – 7 – 8 – 9 – 10 – 15 – 16 – 17 – 20 – 8 – 26 – 4 – 27 - 30

Jalur 6 : 1 – 2 – 3 – 4 – 5 – 6 – 7 – 8 – 9 – 10 – 15 – 18 – 19 – 20 – 8 – 26 – 4 – 27 - 30

Jalur 7 : 1 – 2 – 3 – 4 – 5 – 6 – 7 – 21 – 22 – 23 – 26 – 4 – 27 - 30

Jalur 8 : 1 – 2 – 3 – 4 – 5 – 6 – 7 – 21 – 22 – 24 – 25 – 26 – 4 – 27 – 30

Jalur 9 : 1 – 2 – 28 – 29 – 30

Berdasarkan hasil dari banyaknya jalur *independent*, maka dapat dilakukan pengujian terhadap tiap jalur tersebut dengan kasus uji yang ditunjukkan pada Tabel 6.3.

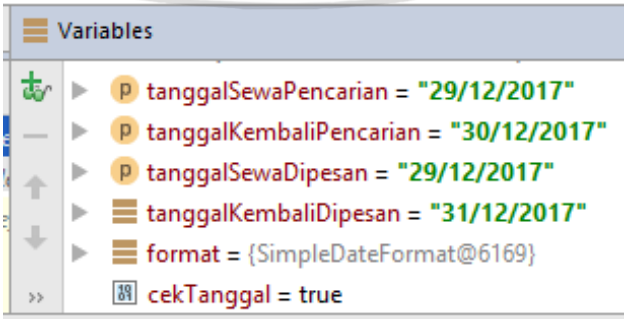
Tabel 6.3 Kasus Uji Pengujian Integrasi *Method* getHasilPencarian

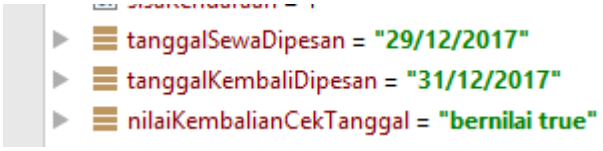
Jalur	Kasus Uji	Hasil yang Diharapkan	Hasil yang Didapatkan
1	Ketika tidak ada data kendaraan yang dapat di eksekusi lagi.	Data kendaraan tidak ada yang di seleksi.	Data kendaraan tidak ada yang di seleksi.
2	Ketika tidak ada data kendaraan di child cekSisa dan tidak ada lagi data kendaraan yang dapat di eksekusi lagi.	Data kendaraan tidak ada yang di seleksi.	Data kendaraan tidak ada yang di seleksi.
3	Melakukan pencarian kendaraan untuk idKendaraan yang telah tersimpan pada child	Data kendaraan tampil pada halaman hasil pencarian.	Data kendaraan tampil pada halaman hasil pencarian.

	cekSisakendaraan dengan kondisi nilai kembalian method cekTanggal bernilai true dan sisa kendaraan tidak lebih kecil dari pada jumlah kendaraan pencarian.		
4	Melakukan pencarian kendaraan untuk idKendaraan yang telah tersimpan pada child cekSisakendaraan dengan kondisi nilai kembalian method cekTanggal bernilai false dan sisa kendaraan lebih kecil dari pada jumlah kendaraan pencarian.	Data kendaraan tidak tampil pada halaman hasil pencarian.	Data kendaraan tidak tampil pada halaman hasil pencarian.
5	Melakukan pencarian kendaraan untuk idKendaraan yang telah tersimpan pada child cekSisakendaraan dengan tanggal pencarian yang berbeda pada child cekSisaKendaraan dan jumlah kendaraan pencarian tidak lebih besar dari jumlah kendaraan yang ada pada child kendaraan.	Data kendaraan tampil pada halaman pencarian.	Data kendaraan tampil pada halaman pencarian.
6	Melakukan pencarian kendaraan untuk idKendaraan yang telah tersimpan pada child cekSisakendaraan dengan tanggal pencarian yang berbeda pada child cekSisaKendaraan dan jumlah kendaraan pencarian lebih besar dari jumlah kendaraan yang ada pada child kendaraan.	Data kendaraan tidak tampil pada halaman pencarian.	Data kendaraan tidak tampil pada halaman pencarian.
7	Melakukan pencarian kendaraan untuk idKendaraan yang belum tersimpan pada child	Data kendaraan tampil pada halaman pencarian.	Data kendaraan tampil pada halaman pencarian.

	cekSisaKendaraan dan jumlah kencaraan pencarian tidak lebih besar dari jumlah kendaraan yang ada pada child kendaraan.		
8	Melakukan pencarian kendaraan untuk idKendaraan yang belum tersimpan pada child cekSisaKendaraan dan jumlah kencaraan pencarian lebih besar dari jumlah kendaraan yang ada pada child kendaraan.	Data kendaraan tidak tampil pada halaman pencarian.	Data kendaraan tidak tampil pada halaman pencarian.
9	Melakukan pencarian kendaraan dengan kondisi tidak ada data kendaraan yang tersimpan.	Sistem menampilkan pesan peringatan bahwa kendaraan tidak tersedia.	Sistem menampilkan pesan peringatan bahwa kendaraan tidak tersedia.

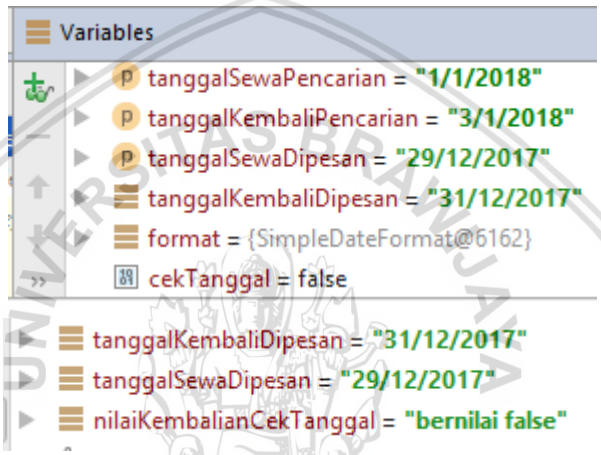
Selanjutnya adalah membandingkan nilai yang didapatkan sebelumnya menggunakan *stub* dengan nilai yang didapatkan dengan menggunakan *method* `cekTanggal` yang telah di integrasikan dengan *method* `getHasilPencarian`. Untuk mendapatkan nilai true dari nilai kembalian *method* `cekTanggal`, akan dilakukan simulasi dengan mengisi jumlah tanggal sewa dan tanggal kembali pencarian yang sama dengan tanggal sewa dan tanggal kembali yang terdapat pada *child* `cekSisaKendaraan` pada *firebase database*. Gambar 6.12 menunjukkan hasil integrasi antara *method* `getHasilPencarian` dan *method* `cekTanggal` dengan nilai kembalian dari *method* `cekTanggal` bernilai *true*.





Gambar 6.12 Hasil Pengujian Integrasi *Method* getHasilPencarian

Sedangkan untuk mendapatkan nilai *false* dari nilai kembalian *method* `cekTanggal`, akan dilakukan simulasi dengan mengisi jumlah tanggal sewa dan tanggal kembali pencarian yang berbeda dengan tanggal sewa dan tanggal kembali yang terdapat pada *child* `cekSisaKendaraan` pada *firebase database*. Gambar 6.13 menunjukkan hasil integrasi antara *method* `getHasilPencarian` dan *method* `cekTanggal` dengan nilai kembalian dari *method* `cekTanggal` bernilai *false*.

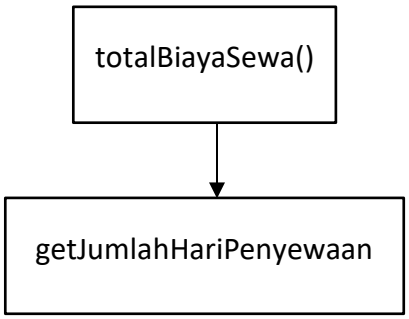


Gambar 6.13 Hasil Pengujian Integrasi *Method* getHasilPencarian

Hasil dari penggunaan *stub* dan penggunaan *method* `cekTanggal` yang sudah di integrasi kan dengan *method* `getHasilPencarian` menghasilkan hasil yang sama. Sehingga dapat disimpulkan bahwa *method* `cekTanggal` dan `getHasilPencarian` sudah berhasil terintegrasi.

6.1.2.2 Pengujian Integrasi method totalBiayaSewa

Method `totalBiayaSewa` merupakan suatu *method* integrasi yang berfungsi untuk melakukan perhitungan total biaya sewa yang harus di bayar oleh pelanggan. *Method* ini terintegrasi dengan *method* `getJumlahHariPenyewaan` yang sudah di uji pada pengujian unit. Diagram hirarki dari dua *method* yang saling terintegrasi tersebut digambarkan oleh Gambar 6.14.



Gambar 6.14 Hirarki Pengujian Integrasi *Method* totalBiayaSewa

Strategi pengujian integrasi yang akan digunakan untuk menguji method integrasi `totalBiayaSewa` adalah menggunakan pendekatan *Top Down*. Pada pendekatan ini pada *method* `getJumlahHariPenyewaan` akan di buat sebuah *stub* yang berfungsi sebagai modul pengganti yang akan memanggil modul yang sedang di uji. Langkah – langkah dalam pengujian integrasi *method* `totalBiayaSewa` seperti berikut ini :

Langkah 1 : IS = `totalBiayaSewa` + `getJumlahHariPenyewaan`

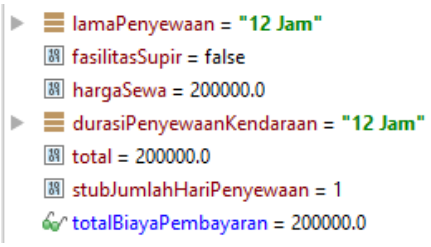
Pada pengujian integrasi *method* `totalBiayaSewa` hanya terdiri dari 1 langkah saja. Hal ini karenakan hanya terdapat 1 *method* yang terintegrasi dengan *method* `totalBiayaSewa`. *Stub* yang akan digunakan pada pengujian integrasi *method* `totalBiayaSewa` mempunyai nilai integer sebagai pengganti nilai kembalian dari *method* `getJumlahHariPenyewaan`. *Stub* yang digunakan di inisialisasi sebagai variabel `stubJumlahHariPenyewaan` dengan nilai awal sama dengan 1. Gambar 6.15 merupakan kode program *method* `totalBiayaSewa` yang menggunakan *stub* sebagai modul pengganti nilai kembalian *method* `getJumlahHariPenyewaan`.

```
public void totalBiayaSewa() throws ParseException  {
    final String idKendaraan =
getIntent().getStringExtra("idKendaraan");
    final String kategoriKendaraan =
getIntent().getStringExtra("kategoriKendaraan");
    final String jumlahKendaraanPencarian =
getIntent().getStringExtra("jumlahKendaraanPencarian");
    final int jmlKendaraan =
Integer.parseInt(jumlahKendaraanPencarian);
mDatabase.child("kendaraan").child(kategoriKendaraan).chi
ld(idKendaraan).addValueEventListener(new
ValueEventListener() {
        @Override
        public void onDataChange(DataSnapshot
dataSnapshot) {
            KendaraanModel kendaraan =
dataSnapshot.getValue(KendaraanModel.class);
            String lamaPenyewaan =
kendaraan.getLamaPenyewaan();
            boolean fasilitasSupir = kendaraan.isSupir();
            double hargaSewa = kendaraan.getHargaSewa();
            String durasiPenyewaanKendaraan = "12 Jam";
            double total = 0;
                try {
            // stub
            final int stubJumlahHariPenyewaan = 1;
```

```
            if
((lamaPenyewaan.equals(durasiPenyewaanKendaraan) &&
fasilitasSupir == true)) {
                total = (stubJumlahHariPenyewaan *
hargaSewa) * jmlKendaraan;
                totalBiayaPembayaran = total;
                textViewTotalPembayaran.setText("Rp." +
BaseActivity.rupiah().format(total));
            }
            else if
((lamaPenyewaan.equals(durasiPenyewaanKendaraan) &&
fasilitasSupir == false && stubJumlahHariPenyewaan==1)) {
                total = (stubJumlahHariPenyewaan *
hargaSewa) * jmlKendaraan;
                totalBiayaPembayaran = total;
                textViewTotalPembayaran.setText("Rp." +
BaseActivity.rupiah().format(total));
            }
            else if
((lamaPenyewaan.equals(durasiPenyewaanKendaraan) &&
fasilitasSupir == false && stubJumlahHariPenyewaan>1)) {
                total = (stubJumlahHariPenyewaan *
(hargaSewa*2)) * jmlKendaraan;
                totalBiayaPembayaran = total;
                textViewTotalPembayaran.setText("Rp." +
BaseActivity.rupiah().format(total));
            } else {
                total = (stubJumlahHariPenyewaan *
(hargaSewa)) * jmlKendaraan;
                totalBiayaPembayaran = total;
                textViewTotalPembayaran.setText("Rp." +
BaseActivity.rupiah().format(total));
            }
        } catch (Exception e) {
            Toast.makeText(getApplicationContext(),
"Terdapat Kesalahan pada Sistem",
Toast.LENGTH_LONG).show();
            Intent intent = new
Intent(RincianPenyewaan.this, MainActivity.class);
            startActivity(intent);
        }
    }
    @Override
    public void onCancelled(DatabaseError databaseError)
{

    }
});
```

Gambar 6.15 Kode Program Method totalBiayaSewa dengan menggunakan *Stub*

Dengan adanya *stub* pada *method* `totalBiayaSewa`, berhasil menjalankan fungsinya dengan sukses. Hal ini ditunjukkan dengan Gambar 6.16.



Gambar 6.16 Hasil Penggunaan *Stub* pada *Method* totalBiayaSewa

Selanjutnya adalah dengan mengintegrasikan *method* `totalBiayaSewa` dengan *getJumlahHariPenyewaan*. Gambar dibawah ini akan menjelaskan tentang kode program `totalBiayaSewa` yang telah di integrasikan dengan *method* `getJumlahHariPenyewaan`.

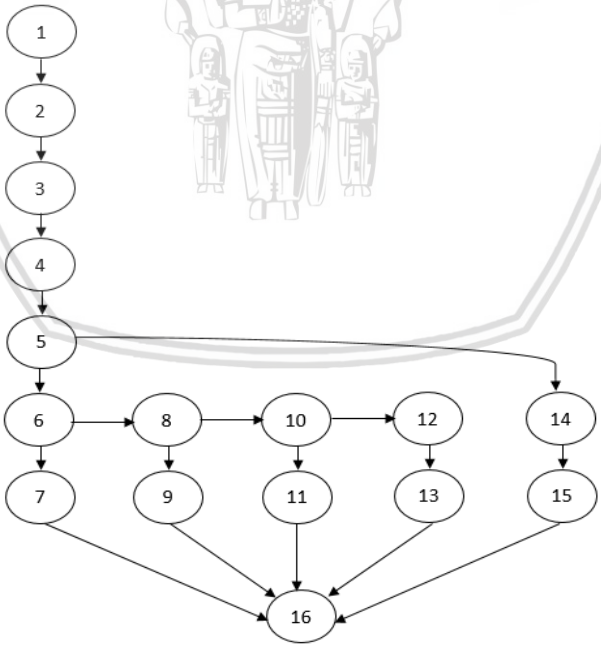
```
public void totalBiayaSewa() throws ParseException  {
    final String idKendaraan =
getIntent().getStringExtra("idKendaraan");
    final String kategoriKendaraan =
getIntent().getStringExtra("kategoriKendaraan");
    final String jumlahKendaraanPencarian =
getIntent().getStringExtra("jumlahKendaraanPencarian");
    final int jmlKendaraan =
Integer.parseInt(jumlahKendaraanPencarian);
mDatabase.child("kendaraan").child(kategoriKendaraan).chi
ld(idKendaraan).addValueEventListener(new
ValueEventListener() {
        @Override
        public void onDataChange(DataSnapshot
dataSnapshot) {
            KendaraanModel kendaraan =
dataSnapshot.getValue(KendaraanModel.class);
            String lamaPenyewaan =
kendaraan.getLamaPenyewaan();
            boolean fasilitasSupir = kendaraan.isSupir();
            double hargaSewa = kendaraan.getHargaSewa();
            String durasiPenyewaanKendaraan = "12 Jam";
            double total = 0;
            try {
                if
((lamaPenyewaan.equals(durasiPenyewaanKendaraan) &&
fasilitasSupir == true)) {
                    total = (getJumlahHariPenyewaan() *
hargaSewa) * jmlKendaraan;
                    totalBiayaPembayaran = total;
                    textViewTotalPembayaran.setText("Rp."
+ BaseActivity.rupiah().format(total));
                }
                else if
((lamaPenyewaan.equals(durasiPenyewaanKendaraan) &&
fasilitasSupir == false && getJumlahHariPenyewaan()==1))
{
                    total = (getJumlahHariPenyewaan() *
hargaSewa) * jmlKendaraan;
                    totalBiayaPembayaran = total;
                    textViewTotalPembayaran.setText("Rp."
+ BaseActivity.rupiah().format(total));
                }
                else if
((lamaPenyewaan.equals(durasiPenyewaanKendaraan) &&
fasilitasSupir == false && getJumlahHariPenyewaan()>1)) {
                    total = (getJumlahHariPenyewaan() *
```

1 2 3 4 5 6 7 8 9 10 11

```
(hargaSewa*2)) * jmlKendaraan;                                          11
                    totalBiayaPembayaran = total;
                    textViewTotalPembayaran.setText("Rp."
+ BaseActivity.rupiah().format(total));
12              } else {
                    total = (getJumlahHariPenyewaan() *
(hargaSewa)) * jmlKendaraan;                                            13
                    totalBiayaPembayaran = total;
                    textViewTotalPembayaran.setText("Rp."
+ BaseActivity.rupiah().format(total));
                }
14          } catch (ParseException e) {
                Toast.makeText(getApplicationContext(),
"Terdapat Kesalahan pada Sistem",
Toast.LENGTH_LONG).show();                                              15
                Intent intent = new
Intent(RincianPenyewaan.this, MainActivity.class);
                startActivity(intent);
            }
        }
        @Override
        public void onCancelled(DatabaseError
databaseError) {
        }
    });                                                                 16
}
```

Gambar 6.17. Kode Program *method* totalBiayaSewa

Berdasarkan kode program tersebut, Gambar 6.18 akan menunjukkan *flow graph* dari *method* `totalBiayaSewa` yang berfungsi menggambarkan alur dari method tersebut.



Gambar 6.18 *Flow Graph Method* totalBiayaSewa

Tahapan selanjutnya dari pengujian integrasi *method* `totalBiayaSewa` yaitu tahapan untuk menghitung *cyclomatic complexity*. Berdasarkan *flow graph method* `totalBiayaSewa`, didapatkan perhitungan *cyclomatic complexity* dengan menggunakan persamaan V(G) = E – N + 2.

V(G) = 19 – 16 + 2

= 5

Setelah mendapatkan nilai dari *cyclomatic complexity*, maka didapatkan dua buah basis set jalur *independent*, yaitu :

Jalur 1 : 1 – 2 – 3 – 4 – 5 – 6 – 7 – 16

Jalur 2 : 1 – 2 – 3 – 4 – 6 – 8 – 9 – 16

Jalur 3 : 1 – 2 – 3 – 4 – 6 – 8 – 10 – 11 – 16

Jalur 4 : 1 – 2 – 3 – 4 – 6 – 8 – 10 – 12 – 13 – 16

Jalur 5 : 1 – 2 – 3 – 4 – 5 – 14 – 15 – 16

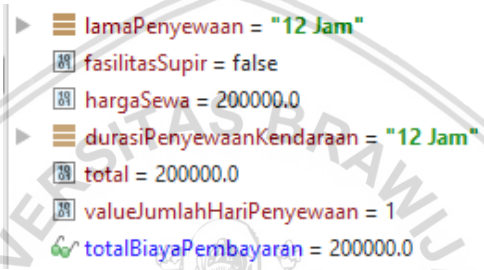
Berdasarkan hasil dari banyaknya jalur *independent*, maka dapat dilakukan pengujian terhadap tiap jalur tersebut dengan kasus uji yang ditunjukkan pada Tabel 6.4.

Tabel 6.4 Kasus Uji Pengujian Integrasi *Method* totalBiayaSewa

Jalur	Kasus Uji	Hasil yang Diharapkan	Hasil yang Didapatkan
1	Ketika nilai variabel lamaPenyewaan sama dengan nilai durasiPenyewaanKendaraan dan nilai variabel fasilitasSupir sama dengan true.	Menampilkan hasil kalkulasi biaya pembayaran.	Menampilkan hasil kalkulasi biaya pembayaran.
2	Ketika nilai variabel lamaPenyewaan sama dengan nilai durasiPenyewaanKendaraan, nilai variabel fasilitasSupir sama dengan true dan nilai kembalian getJumlahHariPenyewaan sama dengan 1.	Menampilkan hasil kalkulasi biaya pembayaran.	Menampilkan hasil kalkulasi biaya pembayaran.
3	Ketika nilai variabel lamaPenyewaan sama dengan nilai durasiPenyewaanKendaraan, nilai variabel fasilitasSupir sama dengan true dan nilai kembalian getJumlahHariPenyewaan lebih dari 1.	Menampilkan hasil kalkulasi biaya pembayaran.	Menampilkan hasil kalkulasi biaya pembayaran.
4	Ketika kondisi tidak ada yang memenuhi.	Menampilkan hasil kalkulasi biaya pembayaran.	Menampilkan hasil kalkulasi biaya pembayaran.

5	Ketika terdapat kesalahan sistem.	Menampilkan pesan peringatan bahwa terdapat kesalahan sistem.	Menampilkan pesan peringatan bahwa terdapat kesalahan sistem.

Selanjutnya adalah membandingkan nilai yang didapatkan sebelumnya menggunakan *stub* dengan nilai yang didapatkan dengan menggunakan *method* `getJumlahHariPenyewaan` yang telah di integrasikan dengan *method* `totalBiayaSewa`. Gambar 6.19 menunjukkan hasil yang didapatkan dengan menggunakan integrasi *method* `getJumlahHariPenyewaan`.



Gambar 6.19 Hasil Pengujian Integrasi totalBiayaSewa

Hasil dari penggunaan *stub* dan penggunaan *method* `getJumlahHariPenyewaan` yang sudah di integrasi kan dengan method `totalBiayaSewa` menghasilkan hasil yang sama. Sehingga dapat disimpulkan bahwa *method* `getJumlahHariPenyewaan` dan `totalBiayaSewa` sudah berhasil terintegrasi.

6.1.3 Pengujian Validasi

Pengujian validasi berfungsi untuk mengetahui apakah sistem yang telah dibangun sudah sesuai dengan kebutuhan fungsional yang sudah di rancang pada tahap perancangan. Metode pengujian yang digunakan untuk melakukan pengujian validasi adalah metode pengujian *Black box*. Hal ini karenakan pengujian validasi lebih berfokus untuk mengamati hasil eksekusi melalui kasus uji. Pada pengujian validasi akan terbagi menjadi tiga bagian yaitu, pengujian validasi sisi pengguna, pengujian validasi sisi pelanggan dan pengujian validasi sisi pemilik rental.

6.1.3.1 Pengujian Validasi Sisi Pengguna

a. Kasus Uji Autentifikasi Nomor Telepon Berhasil

Tabel 6.5 Kasus Uji Autentifikasi Nomor Telepon Berhasil

Nama Kasus Uji	Autentifikasi Nomor Telepon Berhasil

Tujuan Pengujian	Untuk membuktikan bahwa sistem dapat melakukan registrasi dengan menggunakan nomor telepon.
Prosedur Pengujian	1. Memasukkan nomor telepon. 2. Menekan tombol Selanjutnya. 3. Memasukkan kode verifikasi. 4. Menekan tombol Simpan.
Hasil yang Diharapkan	Sistem dapat melakukan verfikasi terhadap kode verfikasi yang dimasukkan pengguna, apabila verifikasi benar akan ditampilkan halaman untuk mengisi biodata pengguna dan menyimpan data autentifikasi pengguna ke dalam *firebase database.*

b. Kasus Uji Autentifikasi Nomor Telepon Jika kode verifikasi salah

Tabel 6.6 Kasus Uji Autentifikasi Nomor Telepon Jika kode verifikasi salah

Nama Kasus Uji	Autentifikasi Nomor Telepon Jika kode verifikasi salah
Tujuan Pengujian	Untuk membuktikan bahwa sistem dapat menampilkan pesan peringatan apabila kode verifikasi tidak sesuai.
Prosedur Pengujian	1. Memasukkan nomor telepon. 2. Menekan tombol Selanjutnya. 3. Memasukkan kode verifikasi yang tidak sesuai dengan kode verifikasi yang dikirimkan. 4. Menekan tombol Simpan.
Hasil yang Diharapkan	Sistem menampilkan pesan peringatan

c. Kasus Uji Autentifikasi Nomor Telepon Jika data sudah terdaftar

Tabel 6.7 Kasus Uji Autentifikasi Nomor Telepon Jika data sudah terdaftar

Nama Kasus Uji	Autentifikasi Nomor Telepon Jika data sudah terdaftar
Tujuan Pengujian	Untuk membuktikan bahwa sistem dapat melakukan login ke dalam sistem .
Prosedur Pengujian	1. Memasukkan nomor telepon. 2. Menekan tombol Selanjutnya. 3. Memasukkan kode verifikasi yang tidak sesuai dengan kode verifikasi yang dikirimkan.

	4. Menekan tombol Simpan.
Hasil yang Diharapkan	Sistem akan mengerahkan ke halaman utama sistem dan menamppilkan pesan berhasil login.

d. Kasus Uji Mengisi Biodata

Tabel 6.8 Kasus Uji Mengisi Biodata

Nama Kasus Uji	Mengisi Biodata
Tujuan Pengujian	Untuk membuktikan bahwa sistem dapat menyimpan biodata pengguna ke dalam sistem.
Prosedur Pengujian	1. Memilih foto profil. 2. Mengisi kolom isian biodata. 3. Menekan tombol simpan.
Hasil yang Diharapkan	Sistem dapat menjalankan fungsi simpan biodata pelanggan ke dalam firebase database dan menampilkan halaman utama sistem.

e. Kasus Uji Mengisi Biodata dengan kolom isian tidak lengkap

Tabel 6.9 Kasus Uji Mengisi Biodata dengan kolom isian tidak lengkap

Nama Kasus Uji	Mengisi Biodata
Tujuan Pengujian	Untuk membuktikan bahwa sistem dapat menyimpan biodata pengguna ke dalam sistem.
Prosedur Pengujian	1. Memilih foto profil. 2. Menekan tombol simpan.
Hasil yang Diharapkan	Sistem menampilkan pesan peringatan untuk mengisi semua kolom isian.

6.1.3.2 Pengujian Validasi Sisi Pelanggan

a. Kasus Uji Melakukan Pencarian Kendaraan

Tabel 6.10 Kasus Uji Melakukan Pencarian Kendaraan

Nama Kasus Uji	Melakukan Pencarian Kendaraan
Tujuan Pengujian	Untuk membuktikan bahwa sistem dapat menampilkan daftar kendaraan yang tersedia sesuai dengan kriteria pencarian pelanggan.
Prosedur Pengujian	1. Memilih tanggal sewa.

	2. Memilih tanggal kembali. 3. Memilih kategori kendaraan. 4. Memilih jumlah kendaraan.
Hasil yang Diharapkan	Sistem dapat melakukan proses pencarian kendaraan yang tersedia dan menampilkan daftar kendaraan.

b. Kasus Uji Melakukan Pencarian Kendaraan dengan data kendaraan tidak tersedia.

Tabel 6.11 Kasus Uji Melakukan Pencarian Kendaraan dengan data kendaraan tidak tersedia.

Nama Kasus Uji	Melakukan Pencarian Kendaraan
Tujuan Pengujian	Untuk membuktikan bahwa sistem dapat menampilkan pesan apabila kendaraan tidak tersedia.
Prosedur Pengujian	1. Memilih tanggal sewa yang sama dengan yang ada di basis data. 2. Memilih tanggal kembali yang sama dengan yang ada di basis data. 3. Memilih kategori kendaraan yang sama dengan yang ada di basis data. 4. Memilih jumlah kendaraan yang lebih besar dari sisa kendaraan.
Hasil yang Diharapkan	Sistem menampilkan pesan bahwa kendaraan tidak tersedia.

c. Kasus Uji Melihat Daftar Kendaraan

Tabel 6.12 Kasus Uji Melihat Daftar Kendaraan

Nama Kasus Uji	Melihat Daftar Kendaraan
Tujuan Pengujian	Untuk membuktikan bahwa sistem dapat menampilkan daftar kendaraan yang tersedia.
Prosedur Pengujian	1. Melakukan Pencarian Kendaraan
Hasil yang Diharapkan	Sistem menampilkan daftar kendaraan yang tersedia.

d. Kasus Uji Melihat Informasi Detail Kendaraan

Tabel 6.13 Kasus Uji Melihat Informasi Detail Kendaraan

Nama Kasus Uji	Melihat Informasi Detail Kendaraan
Tujuan Pengujian	Untuk membuktikan bahwa sistem dapat menampilkan informasi detail kendaraan.
Prosedur Pengujian	1. Memilih kendaraan yang diinginkan dengan menekan *listview* daftar kendaraan.
Hasil yang Diharapkan	Sistem menampilkan halaman untuk memberikan informasi detail kendaraan.

e. Kasus Uji Melakukan Penyewaan Berhasil

Tabel 6.14 Kasus Uji Melakukan Penyewaan Berhasil

Nama Kasus Uji	Melakukan Penyewaan Berhasil
Tujuan Pengujian	Untuk membuktikan bahwa sistem dapat melakukan penyewaan kendaraan.
Prosedur Pengujian	1. Menekan tombol pesan sekarang pada halaman detail informasi kendaraan. 2. Menekan tombol lanjutkan pada halaman rincian penyewaan. 3. Mengisi kolom keterangan penyewaan. 4. Menekan tombol buat penyewaan pada halaman informasi penyewaan.
Hasil yang Diharapkan	Sistem dapat melakukan proses penyewaan dan menyimpan data penyewaan ke dalam firebase *database* lalu menampilkan daftar rekening pembayaran.

f. Kasus Uji Melakukan Penyewaan Tidak Berhasil Karena Kendaraan Tidak Tersedia

Tabel 6.15 Kasus Uji Melakukan Penyewaan Tidak Berhasil Karena Kendaraan Tidak Tersedia

Nama Kasus Uji	Melakukan Penyewaan Penyewaan Tidak Berhasil Karena Kendaraan Tidak Tersedia
Tujuan Pengujian	Untuk membuktikan bahwa sistem dapat menampilkan pesan peringatan bahwa kendaraan sudah tidak tersedia.
Prosedur Pengujian	1. Menekan tombol pesan sekarang pada halaman detail informasi kendaraan.

	2. Menekan tombol lanjutkan pada halaman rincian penyewaan. 3. Mengisi kolom keterangan penyewaan. 4. Mengubah sisa kendaraan pada child cekSisaKendaraa di basis data menjadi 0. 5. Menekan tombol buat penyewaan pada halaman informasi penyewaan.
Hasil yang Diharapkan	Sistem menampilkan pesan peringatan bahwa kendaraan sudah tidak tersedia dan mengarahkan ke halaman pencarian.

g. Kasus Uji Melakukan Penyewaan Tidak Berhasil Karna Kolom Isian Kosong

Tabel 6.16 Kasus Uji Melakukan Penyewaan Tidak Berhasil Karna Kolom Isian Kosong

Nama Kasus Uji	Melakukan Penyewaan Tidak Berhasil Tidak Berhasil Karna Kolom Isian Kosong
Tujuan Pengujian	Untuk membuktikan bahwa sistem dapat menampilkan pesan peringatan untuk melengkapi kolom isian.
Prosedur Pengujian	1. Menekan tombol pesan sekarang pada halaman detail informasi kendaraan. 2. Menekan tombol lanjutkan pada halaman rincian penyewaan. 3. Menekan tombol buat penyewaan pada halaman informasi penyewaan.
Hasil yang Diharapkan	Sistem menampilkan pesan peringatan untuk melengkapi kolom isian informasi penyewaan.

h. Kasus Uji Melihat Informasi Detail Penyewaan

Tabel 6.17 Kasus Uji Melihat Informasi Detail Penyewaan

Nama Kasus Uji	Melihat Informasi Detail Penyewaan
Tujuan Pengujian	Untuk membuktikan bahwa sistem dapat menampilkan informasi detail penyewaan.
Prosedur Pengujian	1. Menekan tombol menu. 2. Memilih menu kelola penyewaan. 3. Memilih penyewaan yang ingin ditampilkan informasi detail.

Hasil yang Diharapkan	Sistem menampilkan halaman informasi detail penyewaan.

i. Kasus Uji Melakukan Pembayaran

Tabel 6.18 Kasus Uji Melakukan Pembayaran

Nama Kasus Uji	Melakukan Pembayaran
Tujuan Pengujian	Untuk membuktikan bahwa sistem dapat melakukan pembayaran.
Prosedur Pengujian	1. Memilih nama bank pada halaman daftar rekening bank.
Hasil yang Diharapkan	Sistem menampilkan informasi pembayaran yang harus dilakukan.

j. Kasus Uji Unggah Bukti Pembayaran Berhasil

Tabel 6.19 Kasus Uji Unggah Bukti Pembayaran Berhasil

Nama Kasus Uji	Unggah Bukti Pembayaran
Tujuan Pengujian	Untuk membuktikan bahwa sistem dapat melakukan unggah bukti foto pembayaran dan menyimpannya ke dalam sistem.
Prosedur Pengujian	1. Memilih unggah bukti pembayaran sekarang. 2. Memilih foto bukti pembayaran. 3. Memasukkan informasi transfer pembayaran. 4. Menekan tombol unggah.
Hasil yang Diharapkan	Sistem menyimpan foto bukti pembayaran dan informasi pembayaran ke dalam firebase *database* dan mengubah status penyewaan menjadi menunggu konfirmasi pembayaran.

k. Kasus Uji Unggah Bukti Pembayaran Tidak Berhasil

Tabel 6.20 Kasus Uji Unggah Bukti Pembayaran Tidak Berhasil

Nama Kasus Uji	Unggah Bukti Pembayaran Tidak Berhasil
Tujuan Pengujian	Untuk membuktikan bahwa sistem dapat menampilkan informasi untuk melengkapi kolom isian.
Prosedur Pengujian	1. Memilih unggah bukti pembayaran sekarang.

	2. Memilih foto bukti pembayaran. 3. Menekan tombol unggah.
Hasil yang Diharapkan	Sistem akan menampilkan pesan peringatan untuk mengisi kolom isian.

l. Kasus Uji Melakukan Pembatalan

Tabel 6.21 Kasus Uji Melakukan Pembatalan

Nama Kasus Uji	Melakukan Pembatalan
Tujuan Pengujian	Untuk membuktikan bahwa sistem dapat melakukan pembatalan penyewaan.
Prosedur Pengujian	1. Memilih menu kelola penyewaan. 2. Memilih tab status penyewaan berhasil. 3. Memilih penyewaan yang ingin dibatalkan. 4. Menekan tombol pengajuan pembatalan. 5. Mengisi alasan pembatalan. 6. Menekan tombol ajukan pembatalan.
Hasil yang Diharapkan	Sistem akan merubah status penyewaan menjadi ajukan pembatalan.

m. Kasus Uji Melihat Status Penyewaan

Tabel 6.22 Kasus Uji Melihat Status Penyewaan

Nama Kasus Uji	Melihat Status Penyewaan
Tujuan Pengujian	Untuk membuktikan bahwa sistem dapat menampilkan status penyewaan.
Prosedur Pengujian	1. Menekan tombol menu. 2. Menekan tombol menu kelola penyewaan.
Hasil yang Diharapkan	Sistem akan menampilkan status penyewaan.

n. Kasus Uji Melihat Rental Kendaraan Terdekat

Tabel 6.23 Kasus Uji Melihat Rental Kendaraan Terdekat

Nama Kasus Uji	Melihat Rental Kendaraan Terdekat
Tujuan Pengujian	Untuk membuktikan bahwa sistem dapat menampilkan rental terdekat sekitar pelanggan.
Prosedur Pengujian	1. Menekan tombol terdekat pada halaman hasil pencarian.

	2. Memilih radius yang diinginkan. 3. Menekan tombol terapkan.
Hasil yang Diharapkan	Sistem menampilkan titik lokasi rental terdekat.

o. Kasus Uji Melakukan *Filter* Pencarian

Tabel 6.24 Kasus Uji Melakukan *Filter* Pencarian

Nama Kasus Uji	Melakukan *Filter* Pencarian
Tujuan Pengujian	Untuk membuktikan bahwa sistem dapat melakukan *filter* terhadap pencarian kendaraan.
Prosedur Pengujian	1. Menekan tombol *filter* pada halaman hasil pencarian. 2. Memilih fi*lter* yang diinginkan. 3. Menekan tombol terapkan.
Hasil yang Diharapkan	Sistem menampilkan informasi kendaraan sesuai dengan *filter*.

p. Kasus Uji Melihat Peta Lokasi Rental

Tabel 6.25 Kasus Uji Melihat Peta Lokasi Rental

Nama Kasus Uji	Melihat Peta Lokasi Rental
Tujuan Pengujian	Untuk membuktikan bahwa sistem dapat menampilkan peta lokasi rental.
Prosedur Pengujian	1. Menekan tab menu detail kendaraan pada halaman informasi detail kendaraan.
Hasil yang Diharapkan	Sistem menampilkan informasi lokasi rental dalam bentuk peta.

q. Kasus Uji Kasus Uji Melihat Rute Perjalanan menuju Rental

Tabel 6.26 Kasus Uji Melihat Rute Perjalanan menuju Rental

Nama Kasus Uji	Melihat Rute Perjalanan menuju Rental
Tujuan Pengujian	Untuk membuktikan bahwa sistem dapat menampilkan rute perjalanan menuju rental.
Prosedur Pengujian	1. Menekan tombol panah pada peta lokasi rental pada halaman detail kendaraan.

Hasil yang Diharapkan	Sistem mengarahkan ke halaman google maps dan menampilkan rute perjalanan menuju rental.

r. Kasus Uji Melihat Profil Rental

Tabel 6.27 Kasus Uji Melihat Profil Rental

Nama Kasus Uji	Melihat Profil Rental
Tujuan Pengujian	Untuk membuktikan bahwa sistem dapat menampilkan profil rental.
Prosedur Pengujian	1. Menekan tombol lihat profil pada halaman detail kendaraan.
Hasil yang Diharapkan	Sistem menampilkan informasi profil rental.

s. Kasus Uji Memberikan Penilaian Berhasil

Tabel 6.28 Kasus Uji Memberikan Penilaian Berhasil

Nama Kasus Uji	Memberikan Penilaian Berhasil
Tujuan Pengujian	Untuk membuktikan bahwa sistem dapat menyimpan penilaian pelanggan.
Prosedur Pengujian	1. Menekan tombol berikan penilaian pada halaman detail informasi penyewaan dengan status penyewaan selesai. 2. Memasukkan penilaian dan ulasan. 3. Menekan tombol simpan.
Hasil yang Diharapkan	Sistem menyimpan penilaian pelanggan ke dalam firebase *database*.

t. Kasus Uji Memberikan Penilaian Tidak Berhasil

Tabel 6.29 Kasus Uji Memberikan Penilain Tidak Berhasil

Nama Kasus Uji	Memberikan Penilaian Tidak Berhasil
Tujuan Pengujian	Untuk membuktikan bahwa sistem dapat menampilkan pesan peringatan untuk melengkapi kolom isian penilaian.
Prosedur Pengujian	1. Menekan tombol berikan penilaian pada halaman detail informasi penyewaan dengan status penyewaan selesai. 2. Menekan tombol simpan.

Hasil yang Diharapkan	Sistem menampilkan pesan peringatan untuk melengkapi kolom isian penilaian.

u. Kasus Uji Melihat Profil Pelanggan

Tabel 6.30 Kasus Uji Melihat Profil Pelanggan

Nama Kasus Uji	Melihat Profil Pelanggan Berhasil
Tujuan Pengujian	Untuk membuktikan bahwa sistem dapat menampilkan profil pelanggan.
Prosedur Pengujian	1. Menekan tombol menu. 2. Memilih menu profil.
Hasil yang Diharapkan	Sistem menampilkan informasi profil pelanggan.

v. Kasus Uji Memperbarui Profil Pelanggan Berhasil

Tabel 6.31 Kasus Uji Memperbarui Profil Pelanggan Berhasil

Nama Kasus Uji	Memperbarui Profil Pelanggan Berhasil
Tujuan Pengujian	Untuk membuktikan bahwa sistem dapat memperbarui profil pelanggan.
Prosedur Pengujian	1. Menekan tombol perbarui profil pada halaman profil. 2. Mengisi data yang ingin diubah. 1. Menekan tombol simpan.
Hasil yang Diharapkan	Sistem menyimpan perubahan data pelanggan kedalam firebase *database*.

w. Kasus Uji Memperbarui Profil Pelanggan Tidak Berhasil

Tabel 6.32 Kasus Uji Memperbarui Profil Pelanggan Tidak Berhasil

Nama Kasus Uji	Memperbarui Profil Pelanggan Tidak Berhasil
Tujuan Pengujian	Untuk membuktikan bahwa sistem dapat menampilkan pesan peringatan untuk melengkapi kolom isian.
Prosedur Pengujian	3. Menekan tombol perbarui profil pada halaman profil. 4. Menekan tombol simpan.

Hasil yang Diharapkan	Sistem menampilkan pesan peringatan untuk mengisi kolom isian.

x. Kasus Uji Logout

Tabel 6.33 Kasus Uji Logout

Nama Kasus Uji	Logout
Tujuan Pengujian	Untuk membuktikan bahwa pelanggan dapat keluar dari sistem.
Prosedur Pengujian	1. Menekan tombol menu. 2. Memilih menu logout.
Hasil yang Diharapkan	Sistem menjalankan proses logout dan menampilkan halaman login.

y. Kasus Uji Melihat Penilaian dan Ulasan Rental

Tabel 6.34 Kasus Uji Melihat Penilaian dan Ulasan Rental

Nama Kasus Uji	Melihat Penilaian dan Ulasan Rental
Tujuan Pengujian	Untuk membuktikan bahwa sistem dapat menampilkan penilaian dan ulasan suatu rental.
Prosedur Pengujian	1. Menekan tombol lihat ulasan rental pada halaman profil rental.
Hasil yang Diharapkan	Sistem menampilkan daftar penilaian dan ulasan rental.

z. Kasus Uji Menerima Pemberitahuan

Tabel 6.35 Kasus Uji Menerima Pemberitahuan

Nama Kasus Uji	Menerima Pemberitahuan
Tujuan Pengujian	Untuk membuktikan bahwa sistem dapat mengirimkan pemberitahuan kepada pelanggan pada kondisi tertentu.
Prosedur Pengujian	1. Sisi rental melakukan konfirmasi pembayaran.
Hasil yang Diharapkan	Sistem menampilkan pemberitahuan pada sisi pelanggan.

6.1.3.3 Pengujian Validasi Sisi Pemilik Rental

a. Kasus Uji Melihat Daftar Penyewaan Pelanggan

Tabel 6.36 Kasus Uji Melihat Daftar Penyewaan Pelanggan

Nama Kasus Uji	Melihat Daftar Penyewaan Pelanggan
Tujuan Pengujian	Untuk membuktikan bahwa sistem dapat menampilkan daftar penyewaan yang masuk pada rental.
Prosedur Pengujian	1. Menekan tombol menu. 2. Memilih menu kelola penyewaan.
Hasil yang Diharapkan	Sistem menampilkan daftar penyewaan.

b. Kasus Uji Melihat Informasi Detail Penyewaan Pelanggan

Tabel 6.37 Kasus Uji Melihat Informasi Detail Penyewaan Pelanggan

Nama Kasus Uji	Melihat Informasi Detail Penyewaan Pelanggan
Tujuan Pengujian	Untuk membuktikan bahwa sistem dapat menampilkan informasi detail penyewaan pelanggan.
Prosedur Pengujian	1. Menekan tombol menu. 2. Memilih menu kelola penyewaan. 3. Memilih penyewaan yang ingin ditampilkan informasi detail.
Hasil yang Diharapkan	Sistem menampilkan halaman informasi detail penyewaan.

c. Kasus Uji Melakukan Konfirmasi Pembayaran

Tabel 6.38 Kasus Uji Melakukan Konfirmasi Pembayaran

Nama Kasus Uji	Melakukan Konfirmasi Pembayaran
Tujuan Pengujian	Untuk membuktikan bahwa sistem dapat membantu pemilik rental untuk melakukan konfirmasi pembayaran.
Prosedur Pengujian	1. Menekan tombol konfirmasi pembayaran pada halaman detail penyewaan dengan status menunggu konfirmasi.

Hasil yang Diharapkan	Sistem menampilkan pesan bahwa konfirmasi pembayaran berhasil dan mengubah status penyewaan menjadi berhasil.

d. Kasus Uji Melakukan Konfirmasi Pembatalan Berhasil

Tabel 6.39 Kasus Uji Melakukan Konfirmasi Pembatalan Berhasil

Nama Kasus Uji	Melakukan Konfirmasi Pembatalan
Tujuan Pengujian	Untuk membuktikan bahwa sistem dapat membantu pemilik rental untuk melakukan konfirmasi pembatalan.
Prosedur Pengujian	1. Menekan tombol konfirmasi pembatalan pada halaman detail penyewaan dengan status pengajuan pembatalan. 2. Memilih foto bukti pengembalian dana. 3. Mengisi informasi pengembalian dana. 4. Menekan tombol unggah
Hasil yang Diharapkan	Sistem mengarahkan ke halaman status batal dan mengubah status penyewaan menjadi batal.

e. Kasus Uji Melakukan Konfirmasi Pembatalan Tidak Berhasil

Tabel 6.40 Kasus Uji Melakukan Konfirmasi Pembatalan Tidak Berhasil

Nama Kasus Uji	Melakukan Konfirmasi Pembatalan Tidak Berhasil
Tujuan Pengujian	Untuk membuktikan bahwa sistem dapat menampilkan pesan peringatan untuk melengkapi kolom isian.
Prosedur Pengujian	1. Menekan tombol konfirmasi pembatalan pada halaman detail penyewaan dengan status pengajuan pembatalan. 2. Menekan tombol unggah.
Hasil yang Diharapkan	Sistem menampilkan pesan peringatan untuk melengkapi kolom isian.

f. Kasus Uji Melakukan Konfirmasi Penyewaan Selesai

Tabel 6.41 Kasus Uji Melakukan Konfirmasi Penyewaan Selesai

Nama Kasus Uji	Melakukan Konfirmasi Penyewaan Selesai
Tujuan Pengujian	Untuk membuktikan bahwa sistem dapat membantu pemilik rental untuk melakukan konfirmasi penyewaan selesai.
Prosedur Pengujian	1. Menekan tombol konfirmasi selesai pada halaman detail penyewaan dengan status berhasil.
Hasil yang Diharapkan	Sistem akan mengarahkan ke halaman daftar penyewaan dengan status selesai dan mengubah status penyewaan menjadi selesai.

g. Kasus Uji Melihat Status Penyewaan

Tabel 6.42 Kasus Uji Melihat Status Penyewaan

Nama Kasus Uji	Melihat Status Penyewaan
Tujuan Pengujian	Untuk membuktikan bahwa sistem dapat menampilkan status penyewaan.
Prosedur Pengujian	1. Menekan tombol menu. 2. Menekan tombol kelola penyewaan.
Hasil yang Diharapkan	Sistem akan menampilkan status penyewaan.

h. Kasus Uji Melihat Data Kendaraan

Tabel 6.43 Kasus Uji Melihat Data Kendaraan

Nama Kasus Uji	Melihat Data Kendaraan
Tujuan Pengujian	Untuk membuktikan bahwa sistem dapat menampilkan daftar kendaraan yang sudah tersimpan.
Prosedur Pengujian	1. Menekan tombol menu. 2. Memilih menu manajemen kendaraan.
Hasil yang Diharapkan	Sistem menampilkan daftar kendaraan yang sudah tersimpan.

i. Kasus Uji Menambah Data Kendaraan Berhasil

Tabel 6.44 Kasus Uji Menambah Data Kendaraan Berhasil

Nama Kasus Uji	Menambah Data Kendaraan Berhasil
Tujuan Pengujian	Untuk membuktikan bahwa sistem dapat menyediakan fungsi untuk menambahkan data kendaraan.
Prosedur Pengujian	1. Menekan tombol + (tambah) pada halaman manajemen kendaraan. 2. Memilih foto kendaraan. 3. Mengisi data kendaraan. 4. Menekan tombol simpan.
Hasil yang Diharapkan	Sistem memproses fungsi tambah kendaraan dan menyimpannya ke dalam firebase *database*.

j. Kasus Uji Menambah Data Kendaraan Tidak Berhasil

Tabel 6.45 Kasus Uji Menambah Data Kendaraan Tidak Berhasil

Nama Kasus Uji	Menambah Data Kendaraan Tidak Berhasil
Tujuan Pengujian	Untuk membuktikan bahwa sistem dapat menampilkan pesan peringatan untuk melengkapi kolom isian.
Prosedur Pengujian	1. Menekan tombol + (tambah) pada halaman manajemen kendaraan. 2. Menekan tombol simpan.
Hasil yang Diharapkan	Sistem menampilkan pesan peringatan untuk melengkapi kolom isian.

k. Kasus Uji Memperbarui Data Kendaraan

Tabel 6.46 Kasus Uji Memperbarui Data Kendaraan

Nama Kasus Uji	Memperbarui Data Kendaraan
Tujuan Pengujian	Untuk membuktikan bahwa sistem dapat memperbarui data kendaraan yang sebelumnya tersimpan.
Prosedur Pengujian	1. Memilih kendaraan yang ingin di perbarui pada halaman manajemen kendaraan. 2. Menekan tombol *edit*.

	3. Mengisi data yang ingin diperbarui. 4. Menekan tombol simpan.
Hasil yang Diharapkan	Sistem menjalankan fungsi simpan data kendaraan dan memperbarui data kendaraan yang sebelumnya tersimpan dalam firebase *database*.

l. Kasus Uji Menghapus Data Kendaraan

Tabel 6.47 Kasus Uji Menghapus Data Kendaraan

Nama Kasus Uji	Menghapus Data Kendaraan
Tujuan Pengujian	Untuk membuktikan bahwa sistem dapat melakukan fungsi untuk menghapus data kendaraan.
Prosedur Pengujian	1. Memilih kendaraan yang ingin di hapus pada halaman manajemen kendaraan. 2. Menekan tombol hapus. 3. Menekan tombol ya.
Hasil yang Diharapkan	Sistem akan menghapus data kendaraan yang sebelumnya tersimpan dalam firebase *database.*

m. Kasus Uji Melihat Profil Pelanggan

Tabel 6.48 Kasus Uji Melihat Profil Pelanggan

Nama Kasus Uji	Melihat Profil Pelanggan
Tujuan Pengujian	Untuk membuktikan bahwa sistem dapat menampilkan profil pelanggan.
Prosedur Pengujian	1. Menekan tombol kelola penyewaan 2. Memilih penyewaan. 3. Menekan tombol lihat profil pelanggan.
Hasil yang Diharapkan	Sistem menampilkan informasi profil pelanggan.

n. Kasus Uji Melihat Profil Rental

Tabel 6.49 Kasus Uji Melihat Profil Rental

Nama Kasus Uji	Melihat Profil Rental

Tujuan Pengujian	Untuk membuktikan bahwa sistem dapat menampilkan profil rental.
Prosedur Pengujian	1. Menekan tombol menu. 2. Memilih menu profil.
Hasil yang Diharapkan	Sistem menampilkan informasi profil rental.

o. Kasus Uji Memperbarui Profil Rental Berhasil

Tabel 6.50 Kasus Uji Memperbarui Profil Rental Berhasil

Nama Kasus Uji	Memperbarui Profil Rental Berhasil
Tujuan Pengujian	Untuk membuktikan bahwa sistem dapat memperbarui profil rental.
Prosedur Pengujian	1. Menekan tombol perbarui profil pada halaman profil. 2. Mengisi data yang ingin diubah. 3. Menekan tombol simpan.
Hasil yang Diharapkan	Sistem menyimpan perubahan data rental kedalam firebase *database*.

p. Kasus Uji Memperbarui Profil Rental Tidak Berhasil

Tabel 6.51 Kasus Uji Memperbarui Profil Rental Tidak Berhasil

Nama Kasus Uji	Memperbarui Profil Rental Tidak Berhasil
Tujuan Pengujian	Untuk membuktikan bahwa sistem dapat menampilkan pesan peringatan untuk melengkapi kolom isian.
Prosedur Pengujian	1. Menekan tombol perbarui profil pada halaman profil. 2. Menekan tombol simpan.
Hasil yang Diharapkan	Sistem menampilkan pesan peirngatan untuk melengkapi kolom isian.

q. Kasus Uji Logout

Tabel 6.52 Kasus Uji Logout

Nama Kasus Uji	Logout

Tujuan Pengujian	Untuk membuktikan bahwa pemilik rental dapat keluar dari sistem.
Prosedur Pengujian	1. Menekan tombol menu. 2. Memilih menu logout.
Hasil yang Diharapkan	Sistem menjalankan proses logout dan menampilkan halaman login.

r. Kasus Uji Melihat Penilaian dan Ulasan

Tabel 6.53 Kasus Uji Melihat Penilaian dan Ulasan

Nama Kasus Uji	Melihat Penilaian dan Ulasan
Tujuan Pengujian	Untuk membuktikan bahwa sistem dapat menampilkan daftar penilaian dan ulasan suatu rental.
Prosedur Pengujian	1. Menekan tombol menu. 2. Memilih menu daftar ulasan.
Hasil yang Diharapkan	Sistem menampilkan daftar penilaian dan ulasan dari rental kendaraan.

s. Kasus Uji Menambah Rekening Pembayaran Berhasil

Tabel 6.54 Kasus Uji Menambah Rekening Pembayaran Berhasil

Nama Kasus Uji	Menambah Rekening Pembayaran Berhasil
Tujuan Pengujian	Untuk membuktikan bahwa sistem dapat melakukan penambahan rekening pembayaran.
Prosedur Pengujian	1. Menekan tombol menu. 2. Memilih menu profil. 3. Menekan tombol pengaturan profil. 4. Menekan tombol pengaturan rekening pembayaran. 5. Menekan tombol tambah. 6. Mengisi data rekening pembayaran. 7. Menekan tombol simpan.
Hasil yang Diharapkan	Sistem melakukan proses penambahan rekening pembayaran dan menyimpannya di firebase *database.*

t. Kasus Uji Menambah Rekening Pembayaran Tidak Berhasil

Tabel 6.55 Kasus Uji Menambah Rekening Pembayaran Tidak Berhasil

Nama Kasus Uji	Menambah Rekening Pembayaran Tidak Berhasil
Tujuan Pengujian	Untuk membuktikan bahwa sistem dapat menampilkan pesan peringatan untuk melengkapi kolom isian.
Prosedur Pengujian	1. Menekan tombol menu. 2. Memilih menu profil. 3. Menekan tombol pengaturan profil. 4. Menekan tombol pengaturan rekening pembayaran. 5. Menekan tombol tambah. 6. Menekan tombol simpan.
Hasil yang Diharapkan	Sistem menampilkan pesan peringatan untuk melengkapi kolom isian.

u. Kasus Uji Mengubah Rekening Pembayaran

Tabel 6.56 Kasus Uji Mengubah Rekening Pembayaran

Nama Kasus Uji	Mengubah Rekening Pembayaran
Tujuan Pengujian	Untuk membuktikan bahwa sistem dapat melakukan fungsi mengubah rekening pembayaran.
Prosedur Pengujian	1. Menekan tombol menu. 2. Memilih menu profil. 3. Menekan tombol pengaturan profil. 4. Menekan tombol pengaturan rekening pembayaran. 5. Memilih rekening yang akan diubah. 6. Menekan tombol *edit.* 7. Mengisi data rekening pembayaran yang ingin diubah. 8. Menekan tombol simpan.
Hasil yang Diharapkan	Sistem menyimpan perubahan data rekening rental ke dalam firebase *database.*

v. Kasus Uji Menghapus Rekening Pembayaran

Tabel 6.57 Kasus Uji Menghapus Rekening Pembayaran

Nama Kasus Uji	Menghapus Rekening Pembayaran

Tujuan Pengujian	Untuk membuktikan bahwa sistem dapat menghapus data rekening pembayaran.
Prosedur Pengujian	1. Menekan tombol menu. 2. Memilih menu profil. 3. Menekan tombol pengaturan profil. 4. Menekan tombol pengaturan rekening pembayaran. 5. Memilih rekening yang akan diubah. 6. Menekan tombol hapus. 7. Menekan tombol konfirmasi hapus.
Hasil yang Diharapkan	Sistem menghapus data rekening pada firebase *database.*

w. Kasus uji menerima pemberitahuan

Tabel 6.58 Kasus Uji Menerima Pemberitahuan

Nama Kasus Uji	Menerima Pemberitahuan
Tujuan Pengujian	Untuk membuktikan bahwa sistem dapat mengirimkan pemberitahuan kepada pelanggan pada kondisi tertentu.
Prosedur Pengujian	1. Sisi pelanggan melakukan penyewaan kendaraan dengan status belum bayar.
Hasil yang Diharapkan	Sistem menampilkan pemberitahuan pada sisi pemilik rental dan pemilik rental akan menerima pemberitahuan.

x. Kasus Uji Melakukan Penolakkan Pembayaran Karena Kendaraan Tidak Tersedia

Tabel 6.59 Kasus Uji Melakukan Penolakkan Pembayaran Karena Kendaraan Tidak Tersedia

Nama Kasus Uji	Melakukan Penolakkan Pembayaran Karena Kendaraan Tidak Tersedia
Tujuan Pengujian	Untuk membuktikan bahwa sistem dapat melakukan penolakkan pembayaran karena kendaraan sudah tidak tersedia.
Prosedur Pengujian	1. Menekan tombol tolak pembayaran pada halaman kelola penyewaan dengan status menunggu konfirmasi pembayaran.

	2. Memilih alasan penolakkan.
Hasil yang Diharapkan	Sistem menampilkan halaman unggah bukti pembayaran pengembalian dana.

y. Kasus Uji Melakukan Penolakkan Pembayaran Karena Pembayaran Kurang

Tabel 6.60 Kasus Uji Melakukan Penolakkan Pembayaran Karena Pembayaran Kurang

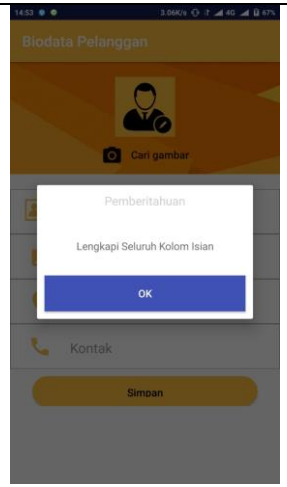
Nama Kasus Uji	Melakukan Penolakkan Pembayaran Karena Pembayaran Kurang
Tujuan Pengujian	Untuk membuktikan bahwa sistem dapat melakukan penolakkan pembayaran karena pembayaran kurang.
Prosedur Pengujian	1. Menekan tombol tolak pembayaran pada halaman kelola penyewaan dengan status menunggu konfirmasi pembayaran. 2. Memilih alasan penolakkan.
Hasil yang Diharapkan	Sistem menampilkan halaman kelola penyewaan dengan status penyewaan menunggu sisa pembayaran.

6.1.3.4 Hasil Pengujian Validasi Sisi Pengguna

Tabel 6.61 Hasil Pengujian Validasi Sisi Pengguna

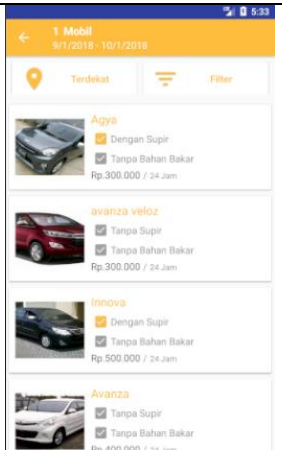
No	Kasus Uji	Hasil Yang Diharapkan	Hasil Yang Didapatkan	Hasil Uji	*Screenshoot* Validasi
1	Autentifikasi Nomor Telepon Berhasil	Sistem dapat melakukan verfikasi terhadap kode verfikasi yang dimasukkan pengguna, apabila verifikasi benar akan ditampilkan halaman untuk mengisi biodata pengguna dan menyimpan data autentifikasi pengguna ke dalam *firebase database.*	Sistem dapat melakukan verfikasi terhadap kode verfikasi yang dimasukkan pengguna, apabila verifikasi benar akan ditampilkan halaman untuk mengisi biodata pengguna dan menyimpan data autentifikasi pengguna ke dalam *firebase database.*	Valid	

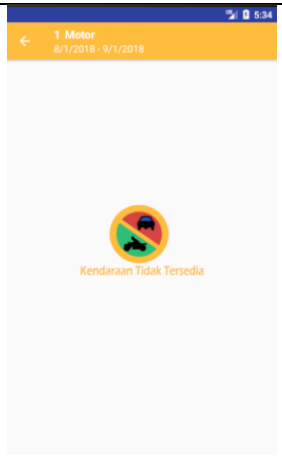
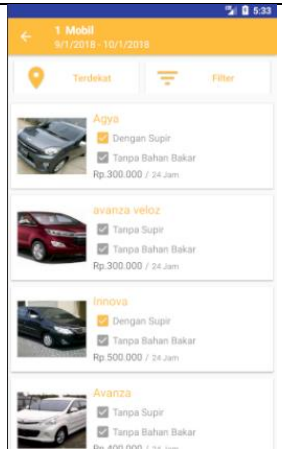
2	Autentifikasi Nomor Telepon Jika kode verifikasi salah	Pengguna memasukkan kode verifikasi yang tidak sesuai dengan kode yang dikirimkan lewat SMS.	Sistem menampilkan pesan peringatan	Valid	
3.	Autentifikasi Nomor Telepon Jika data sudah terdaftar	Pengguna memasukan nomor telepon yang sebelumnya sudah didaftarkan.	Sistem akan mengerahkan ke halaman utama sistem dan menamppilkan pesan berhasil login.	Valid	

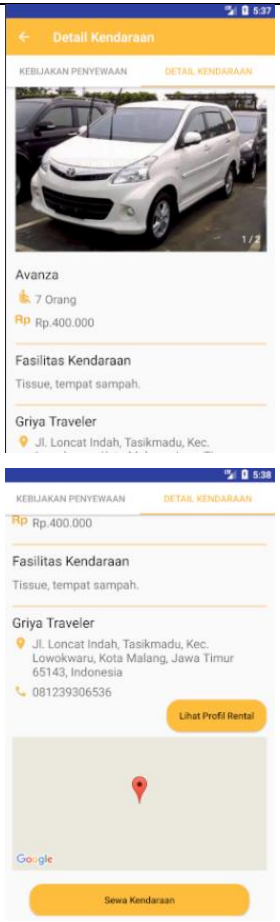
4.	Mengisi Biodata berhasil	Pengguna memasukkan biodata secara lengkap dan sesuai.	Sistem dapat menjalankan fungsi simpan biodata pelanggan ke dalam firebase database dan menampilkan halaman utama sistem.	Valid	
5.	Mengisi Biodata dengan kolom isian tidak lengkap	Pengguna memasukkan biodata tidak secara lengkap	Sistem menampilkan pesan peringatan untuk mengisi semua kolom isian.	Valid	

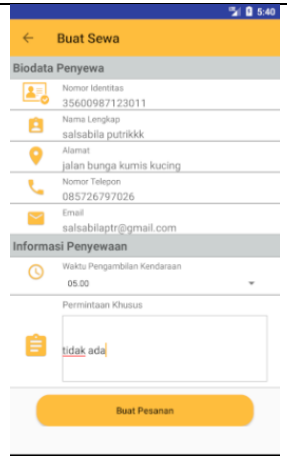
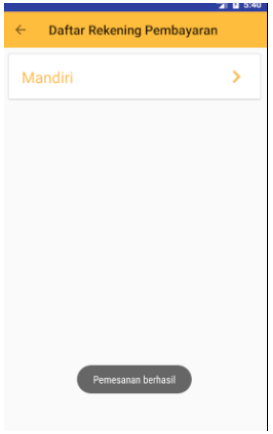
6.1.3.5 Hasil Pengujian Validasi Sisi Pelanggan

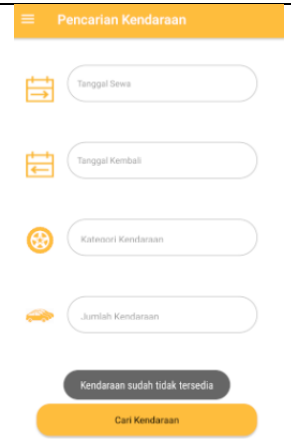
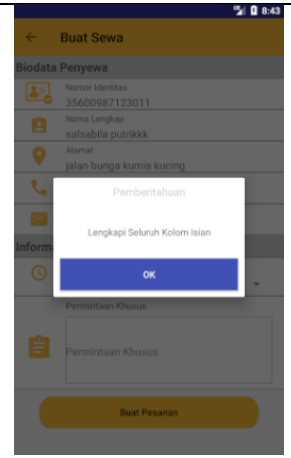
Tabel 6.62 Hasil Pengujian Validasi Sisi Pelanggan

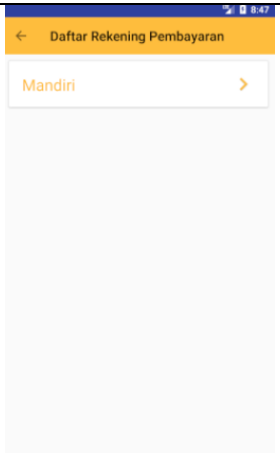
No	Kasus Uji	Hasil Yang Diharapkan	Hasil Yang Didapatkan	Hasil Uji	*Screenshoot* Validasi
1	Melakukan Pencarian Kendaraan berhasil	Sistem dapat melakukan proses pencarian kendaraan yang tersedia dan menampilkan daftar kendaraan.	Sistem dapat melakukan proses pencarian kendaraan yang tersedia dan menampilkan daftar kendaraan.	Valid	

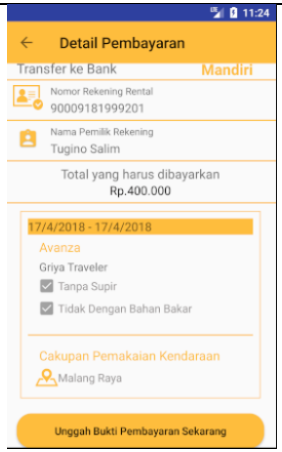
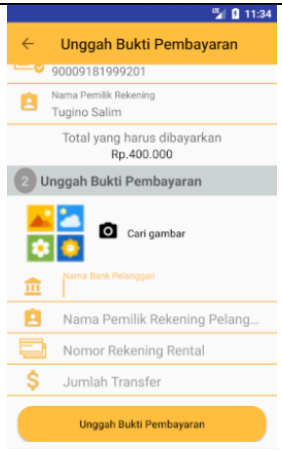
2	Melakukan Pencarian Kendaraan dengan data kendaraan tidak tersedia.	Sistem menampilkan pesan bahwa kendaraan tidak tersedia.	Sistem dapat melakukan proses pencarian kendaraan yang tersedia dan menampilkan daftar kendaraan.	Valid	
3.	Melihat Daftar Kendaraan	Sistem menampilkan halaman untuk memberikan informasi detail kendaraan.	Sistem menampilkan halaman untuk memberikan informasi detail kendaraan.	Valid	

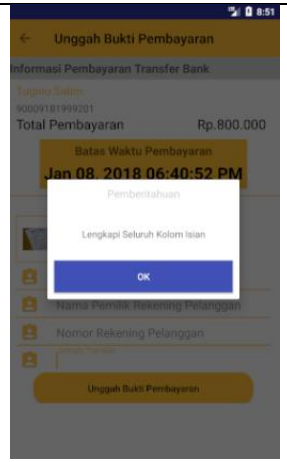
4.	Melihat Informasi Detail Kendaraan	Sistem menampilkan halaman untuk memberikan informasi detail kendaraan.	Sistem menampilkan halaman untuk memberikan informasi detail kendaraan.	Valid	

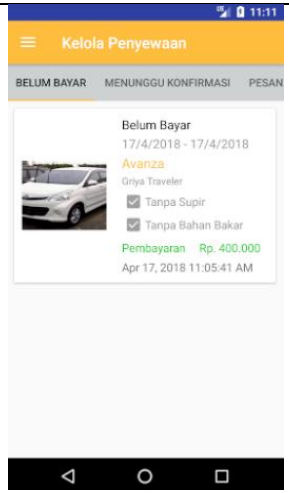
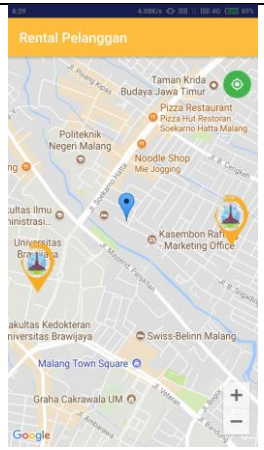
5.	Melakukan Penyewaan Berhasil	Sistem dapat melakukan proses penyewaan dan menyimpan data penyewaan ke dalam firebase *database* lalu menampilkan daftar rekening pembayaran.	Sistem dapat melakukan proses penyewaan dan menyimpan data penyewaan ke dalam firebase *database* lalu menampilkan daftar rekening pembayaran.	Valid	 

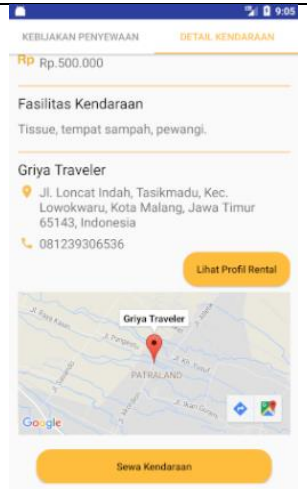
6.	Melakukan Penyewaan Tidak Berhasil Karena Kendaraan Tidak Tersedia	Sistem menampilkan pesan bahwa kendaraan sudah tidak tersedia dan mengarahkan ke halaman pencarian.	Sistem menampilkan pesan bahwa kendaraan sudah tidak tersedia dan mengarahkan ke halaman pencarian.	Valid	
7.	Melakukan Penyewaan Tidak Berhasil Karna Kolom Isian Kosong	Sistem menampilkan pesan peringatan untuk melengkapi kolom isian informasi penyewaan.	Sistem menampilkan pesan peringatan untuk melengkapi kolom isian informasi penyewaan.	Valid	

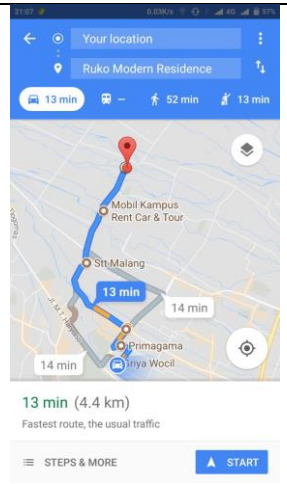
8.	Melihat Informasi Detail Penyewaan	Sistem menampilkan halaman informasi detail penyewaan.	Sistem menampilkan halaman informasi detail penyewaan.	Valid	
9.	Melakukan Pembayaran	Sistem menampilkan informasi pembayaran yang harus dilakukan.	Sistem menampilkan informasi pembayaran yang harus dilakukan.	Valid	

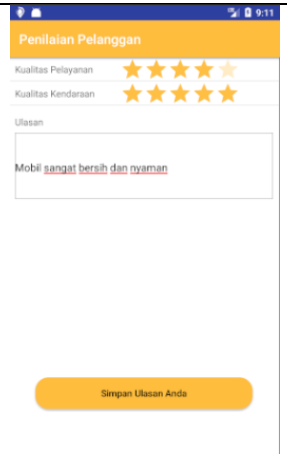
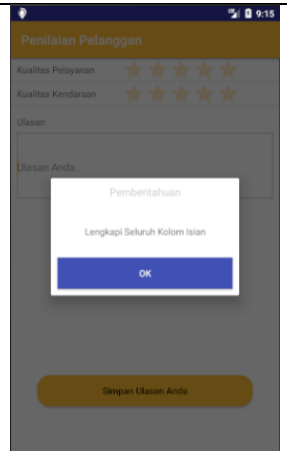
					
10.	Unggah Bukti Pembayaran Berhasil	Sistem menyimpan foto bukti pembayaran dan informasi pembayaran ke dalam firebase *database* dan mengubah status penyewaan menjadi menunggu konfirmasi pembayaran.	Sistem menyimpan foto bukti pembayaran dan informasi pembayaran ke dalam firebase *database* dan mengubah status penyewaan menjadi menunggu konfirmasi pembayaran.	Valid	

11.	Unggah Bukti Pembayaran Tidak Berhasil	Sistem akan menampilkan pesan peringatan untuk mengisi kolom isian.	Sistem akan menampilkan pesan peringatan untuk mengisi kolom isian.	Valid	
12.	Melakukan Pembatalan	Sistem akan merubah status penyewaan menjadi ajukan pembatalan.	Sistem akan merubah status penyewaan menjadi ajukan pembatalan.	Valid	

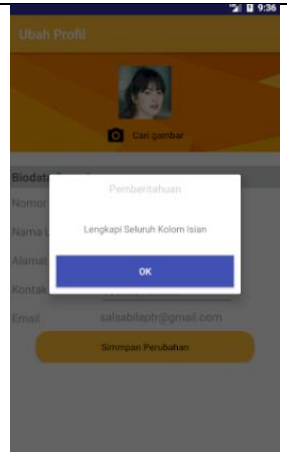
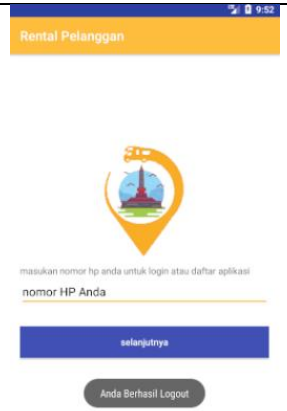
13.	Melihat Status Penyewaan	Sistem akan menampilkan status penyewaan.	Sistem akan menampilkan status penyewaan.	Valid	
14.	Melihat Rental Kendaraan Terdekat	Sistem menampilkan informasi lokasi rental terdekat berdasarkan radius.	Sistem menampilkan informasi lokasi rental terdekat berdasarkan radius.	Valid	

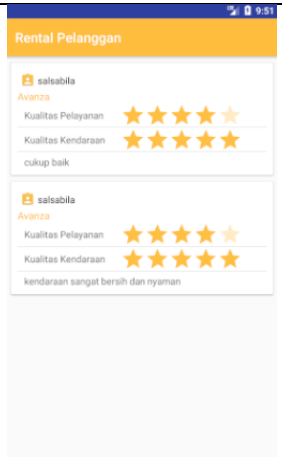
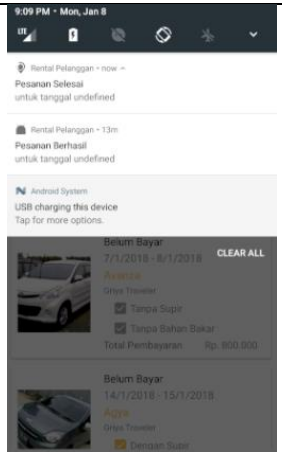
15.	Melakukan *Filter* Pencarian	Sistem menampilkan informasi kendaraan sesuai dengan *filter*.	Sistem menampilkan informasi kendaraan sesuai dengan *filter*.	Invalid	
16.	Melihat Peta Lokasi Rental	Sistem menampilkan informasi lokasi rental dalam bentuk peta.	Sistem menampilkan informasi lokasi rental dalam bentuk peta.	Valid	

17.	Melihat Rute Perjalanan menuju Rental	Sistem mengarahkan ke halaman google maps dan menampilkan rute perjalanan menuju rental.	Sistem mengarahkan ke halaman google maps dan menampilkan rute perjalanan menuju rental.	Valid	
18.	Melihat Profil Rental	Sistem menampilkan informasi profil rental.	Sistem menampilkan informasi profil rental.	Valid	

19.	Memberikan Penilaian Berhasil	Sistem menyimpan penilaian pelanggan ke dalam firebase *database*.	Sistem menyimpan penilaian pelanggan ke dalam firebase *database*.	Valid	
20.	Memberikan Penilaian Tidak Berhasil	Sistem menampilkan pesan peringatan untuk melengkapi kolom isian penilaian.	Sistem menampilkan pesan peringatan untuk melengkapi kolom isian penilaian.	Valid	

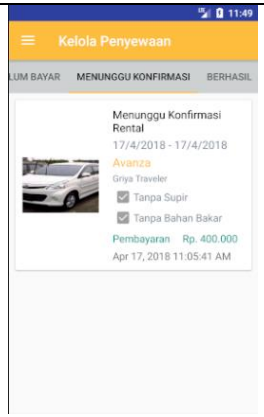
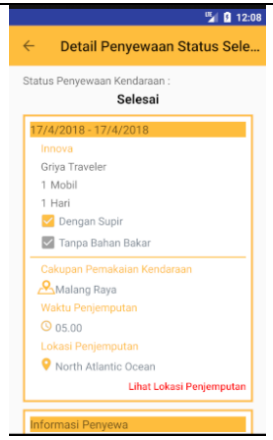
21.	Melihat Profil Pelanggan	Sistem menampilkan informasi profil pelanggan.	Sistem menampilkan informasi profil pelanggan.	Valid	
22.	Memperbarui Profil Pelanggan Berhasil	Sistem menyimpan perubahan data pelanggan kedalam firebase *database*.	Sistem menyimpan perubahan data pelanggan kedalam firebase *database*.	Valid	

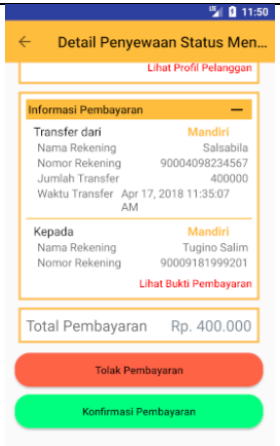
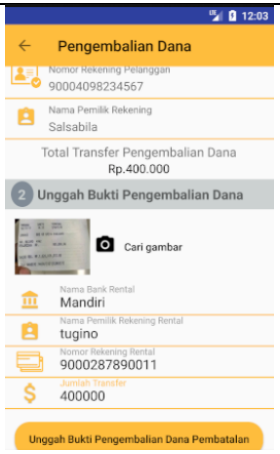
23.	Memperbarui Profil Pelanggan Tidak Berhasil	Sistem menampilkan pesan peringatan untuk mengisi kolom isian.	Sistem menampilkan pesan peringatan untuk mengisi kolom isian.	Valid	
24.	Logout	Sistem menjalankan proses logout dan menampilkan halaman login.	Sistem menjalankan proses logout dan menampilkan halaman login.	Valid	

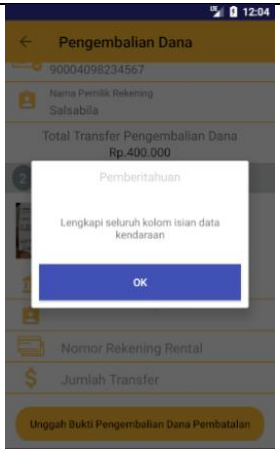
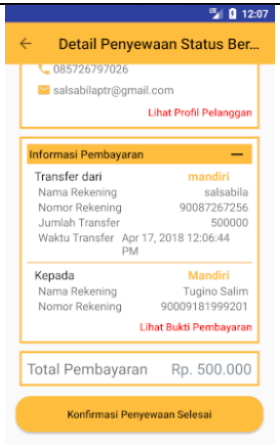
25.	Melihat Penilaian dan Ulasan Rental	Sistem menampilkan daftar penilaian dan ulasan rental.	Sistem menampilkan daftar penilaian dan ulasan rental.	Valid	
26.	Menerima Pemberitahuan	Sistem menampilkan pemberitahuan pada sisi pelanggan.	Sistem menampilkan pemberitahuan pada sisi pelanggan.	Valid	

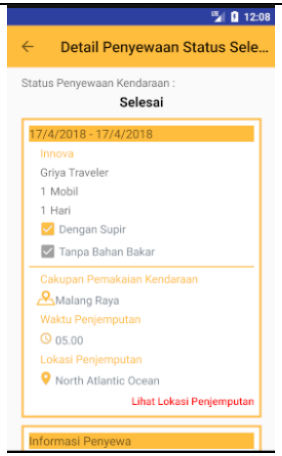
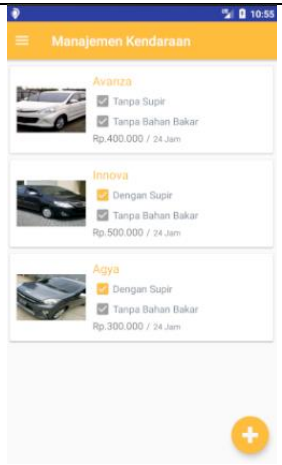
6.1.3.6 Hasil Pengujian Validasi Sisi Pemilik Rental

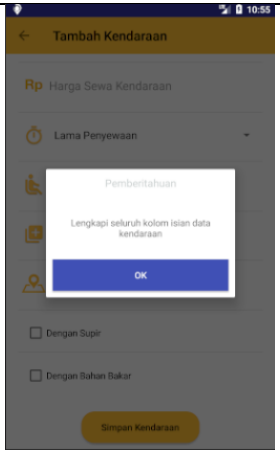
Tabel 6.63 Hasil Pengujian Validasi Sisi Pemilik Rental

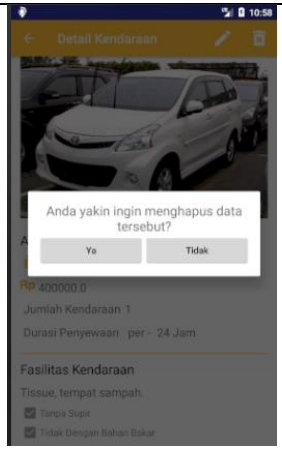
No	Kasus Uji	Hasil Yang Diharapkan	Hasil Yang Didapatkan	Hasil Uji	*Screenshoot* Validasi
1	Melihat Daftar Penyewaan Pelanggan	Sistem menampilkan daftar penyewaan.	Sistem menampilkan daftar penyewaan.	Valid	
2	Melihat Informasi Detail Penyewaan Pelanggan	Sistem menampilkan halaman informasi detail penyewaan.	Sistem menampilkan halaman informasi detail penyewaan.	Valid	

3.	Melakukan Konfirmasi Pembayaran	Sistem menampilkan pesan bahwa konfirmasi pembayaran berhasil dan mengubah status penyewaan menjadi berhasil.	Sistem menampilkan pesan bahwa konfirmasi pembayaran berhasil dan mengubah status penyewaan menjadi berhasil.	Valid	
4.	Melakukan Konfirmasi Pembatalan Berhasil	Sistem mengarahkan ke halaman status batal dan mengubah status penyewaan menjadi batal.	Sistem mengarahkan ke halaman status batal dan mengubah status penyewaan menjadi batal.	Valid	

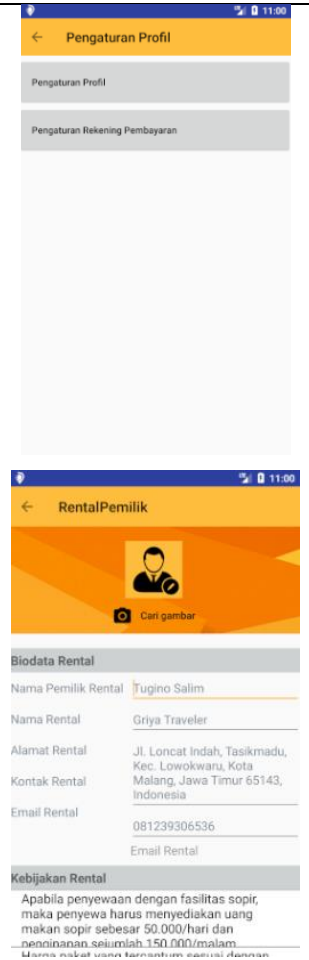
5.	Melakukan Konfirmasi Pembatalan Tidak Berhasil	Sistem menampilkan pesan peringatan untuk melengkapi kolom isian.	Sistem menampilkan pesan peringatan untuk melengkapi kolom isian.	Valid	
6.	Melakukan Konfirmasi Penyewaan Selesai	Sistem akan mengarahkan ke halaman daftar penyewaan dengan status selesai dan mengubah status penyewaan menjadi selesai.	Sistem mengarahkan ke halaman daftar penyewaan dengan status selesai dan mengubah status penyewaan menjadi selesai.	Valid	

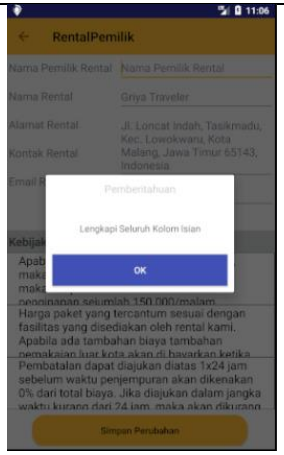
7.	Melihat Status Penyewaan	Sistem akan menampilkan status penyewaan.	Sistem menampilkan status penyewaan.	Valid	
8.	Melihat Data Kendaraan	Sistem menampilkan daftar kendaraan yang sudah tersimpan.	Sistem menampilkan daftar kendaraan yang sudah tersimpan.	Valid	

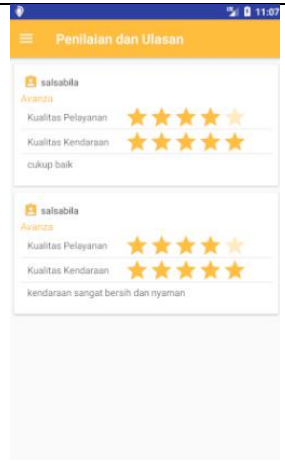
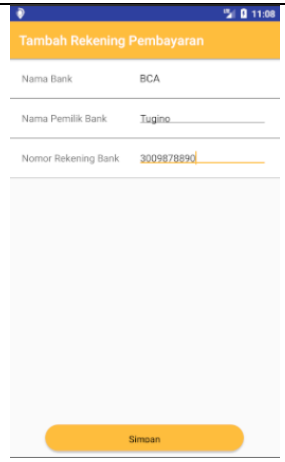
9.	Menambah Data Kendaraan Berhasil	Sistem memproses fungsi tambah kendaraan dan menyimpannya ke dalam firebase *database*.	Sistem memproses fungsi tambah kendaraan dan menyimpannya ke dalam firebase *database*.	Valid	
10.	Menambah Data Kendaraan Tidak Berhasil	Sistem menampilkan pesan peringatan untuk melengkapi kolom isian.	Sistem menampilkan pesan peringatan untuk melengkapi kolom isian.	Valid	

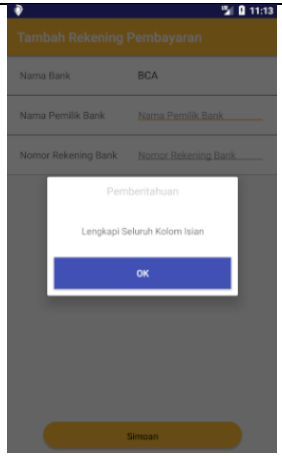
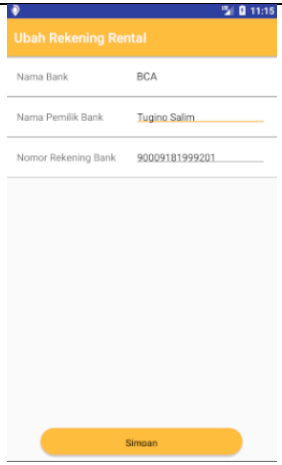
11.	Memperbarui Data Kendaraan	Sistem menjalankan fungsi simpan data kendaraan dan memperbarui data kendaraan yang sebelumnya tersimpan dalam firebase *database*.	Sistem menjalankan fungsi simpan data kendaraan dan memperbarui data kendaraan yang sebelumnya tersimpan dalam firebase *database*.	Valid	
12.	Menghapus Data Kendaraan	Sistem akan menghapus data kendaraan yang sebelumnya tersimpan dalam firebase *database.*	Sistem akan menghapus data kendaraan yang sebelumnya tersimpan dalam firebase *database.*	Valid	

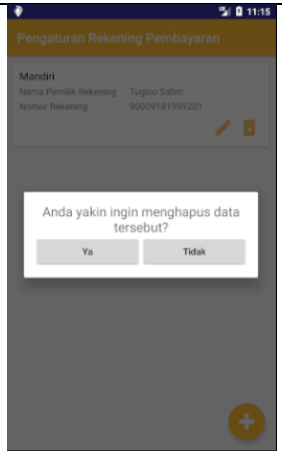
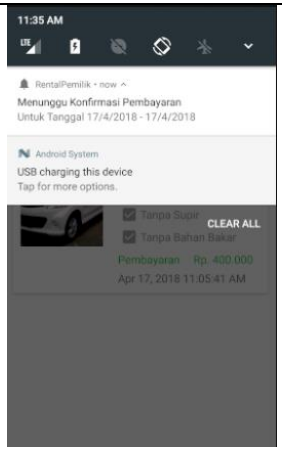
13.	Melihat Profil Pelanggan	Sistem menampilkan informasi profil pelanggan.	Sistem menampilkan informasi profil pelanggan.	Valid	
14.	Melihat Profil Rental	Sistem menampilkan informasi profil rental.	Sistem menampilkan informasi profil rental.	Valid	

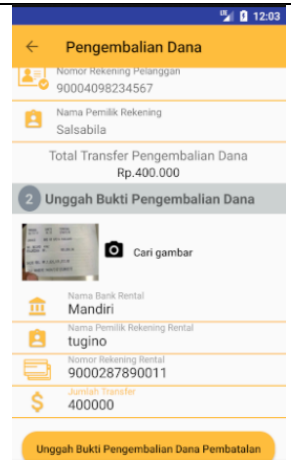
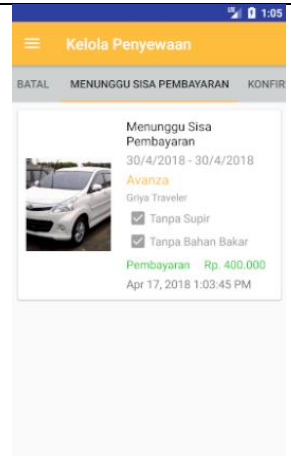
15.	Memperbarui Profil Rental Berhasil	Sistem menyimpan perubahan data rental kedalam firebase *database*.	Sistem menyimpan perubahan data rental kedalam firebase *database*.	Valid	

16.	Memperbarui Profil Rental Tidak Berhasil	Sistem menampilkan pesan peirngatan untuk melengkapi kolom isian.	Sistem menampilkan pesan peirngatan untuk melengkapi kolom isian.	Valid	
17.	Logout	Sistem menjalankan proses logout dan menampilkan halaman login.	Sistem menjalankan proses logout dan menampilkan halaman login.	Valid	

18.	Melihat Penilaian dan Ulasan	Sistem menampilkan daftar penilaian dan ulasan dari rental kendaraan.	Sistem menampilkan daftar penilaian dan ulasan dari rental kendaraan.	Valid	
19.	Menambah Rekening Pembayaran Berhasil	Sistem melakukan proses penambahan rekening pembayaran dan menyimpannya di firebase *database.*	Sistem melakukan proses penambahan rekening pembayaran dan menyimpannya di firebase *database.*	Valid	

20.	Menambah Rekening Pembayaran Tidak Berhasil	Sistem menampilkan pesan peringatan untuk melengkapi kolom isian.	Sistem menampilkan pesan peringatan untuk melengkapi kolom isian.	Valid	
21.	Mengubah Rekening Pembayaran	Sistem menyimpan perubahan data rekening rental ke dalam firebase *database.*	Sistem menyimpan perubahan data rekening rental ke dalam firebase *database.*	Valid	

22.	Menghapus Rekening Pembayaran	Sistem menghapus data rekening pada firebase *database.*	Sistem menghapus data rekening pada firebase *database.*	Valid	
23.	Menerima pemberitahuan	Sistem menampilkan pemberitahuan pada sisi pemilik rental dan pemilik rental akan menerima pemberitahuan.	Sistem menampilkan pemberitahuan pada sisi pemilik rental dan pemilik rental akan menerima pemberitahuan.	Valid	

24.	Melakukan Penolakkan Pembayaran Karena Kendaraan Tidak Tersedia	Sistem menampilkan halaman unggah bukti pembayaran pengembalian dana.	Sistem menampilkan halaman unggah bukti pembayaran pengembalian dana.	Valid	
25.	Melakukan Penolakkan Pembayaran Karena Pembayaran Kurang	Sistem menampilkan halaman kelola penyewaan dengan status penyewaan menunggu sisa pembayaran.	Sistem menampilkan halaman kelola penyewaan dengan status penyewaan menunggu sisa pembayaran.	Valid	

6.2 Pengujian Non-fungsional

Pada penelitian ini akan dilakukan pengujian terhadap kebutuhan non-fungsional. Kebutuhan non-fungsional yang akan dilakukan pengujian adalah kebutuhan non-fungsional *usability*.

6.2.1 Pengujian *Usability*

Pengujian *usability* merupakan pengujian yang dilakukan untuk mengetahui apakah sistem yang telah dikembangkan dapat digunakan dengan mudah atau tidak oleh pengguna dan juga pengujian *usability* bertujuan untuk mengetahui tingkat kepuasan pengguna terhadap penggunaan sistem ini. Pada penelitian ini akan digunakan metode pengujian *System Usability Scale* (SUS).

6.2.1.1 Identifikasi Pengguna

Untuk melakukan pengujian *usability* dibutuhkan 5 orang responden yang dijelaskan sebelumnya tidak harus membutuhkan pengujian pengguna yang besar berkaitan dengan pemborosan sumber daya (Nielsen, 2000). Kelima responden tersebut juga sebelumnya telah ikut serta dalam proses iterasi untuk mengetahui tanggapan dari hasil akhir pengembangan sistem penyewaan kendaraan Kota Malang. Dari kelima responden tersebut akan dibagi menjadi 4 responden bagi sisi pelanggan dan 1 responden bagi sisi pemilik rental.

6.2.1.2 *System Usability Scale*

a. Skenario Pengujian

Pada metode SUS terdapat 10 kriteria pertanyaan yang terbagi menjadi nomor ganji dan genap, dimana nomor ganjil akan memberikan pernyataan positif dan nomor genap akan memberikan pernyataan negatif. Tahap pengujian *usability* akan di mulai dengan memberikan penjelasan mengenai sistem yang akan di uji lalu responden akan menjalan sistem yang sudah dikembangkan dan memberikan penilaian dengan menggunakan kuisioner yang telah diberikan. Daftar Pertanyaan dibagi menjadi dua bagian, yaitu daftar pertanyaan untuk sisi pelanggan dan daftar pertanyaan untuk sisi pemilik rental. Tabel 6.64 akan menjelaskan daftar pertanyaan untuk sisi pelanggan dan Tabel 6.65 akan menjelaskan daftar pertanyaan untuk sisi pemilik rental.

Tabel 6.64 Daftar Pernyataan Pengujian *Usability* Sisi Pelanggan

No	Pernyataan Kuisioner
1	Saya akan menggunakan Sistem Penyewaan Kendaraan Kota Malang berbasis Android untuk melakukan penyewaan kendaraan di Kota Malang.
2	Saya menilai sistem ini menyediakan alur yang kompleks.
3	Saya menilai sistem ini mudah untuk digunakan.
4	Saya membutuhkan bantuan orang lain untuk menggunakan sistem ini.

5	Saya menilai sistem ini menyediakan fungsi yang saling berintegrasi dengan baik.
6	Saya menilai sistem ini mengandung banyak hal yang tidak konsisten.
7	Saya merasa banyak orang yang akan dengan mudah menggunakan sistem ini untuk melakukan penyewaan kendaraan.
8	Saya menilai sistem ini sangat rumit untuk digunakan.
9	Saya merasa dapat menggunakan sistem ini dengan baik.
10	Saya butuh belajar lebih banyak untuk menggunakan sistem ini dengan baik.

Tabel 6.65 Daftar Pernyataan Pengujian *Usability* Sisi Pemilik Rental

No	Pernyataan Kuisioner
1	Saya akan menggunakan Sistem Penyewaan Kendaraan Kota Malang berbasis Android untuk melakukan mengelola penyewaan yang masuk pada rental kendaraan saya.
2	Saya menilai sistem ini menyediakan alur yang kompleks.
3	Saya menilai sistem ini mudah untuk digunakan.
4	Saya membutuhkan bantuan orang lain untuk menggunakan sistem ini.
5	Saya menilai sistem ini menyediakan fungsi yang saling berintegrasi dengan baik.
6	Saya menilai sistem ini mengandung banyak hal yang tidak konsisten.
7	Saya merasa banyak orang yang akan dengan mudah menggunakan sistem ini untuk melakukan penyewaan kendaraan.
8	Saya menilai sistem ini sangat rumit untuk digunakan.
9	Saya merasa dapat menggunakan sistem ini dengan baik.
10	Saya butuh belajar lebih banyak untuk menggunakan sistem ini dengan baik.

b. Hasil Pengujian

Hasil pengujian *usability* dengan menggunakan metode SUS dari sisi pelanggan maupun rental akan dikumpulkan untuk dihitung nilai skor sesuai dengan perhitungan metode SUS. Lalu skor tersebut dapat menjadi tolak ukur apakah sistem yang telah dikembangkan memilik tingkat kepuasan dan kemudahan yang tinggi atau tidak. Penilaian dari metode SUS adalah untuk item nomor ganjil 1, 3, 5, 7 dan 9 memilik skor kontribusi posisi skala dikurangi dengan 1. Lalu untuk item nomor genap 2, 4, 6, 8 dan 10 skor kontribusinya adalah 5 dikurangi dengan nilai

skala yang didapatkan pada tiap item genap. Kemudian dilakukan penjumlahan dari setiap skor item lalu dikalikan dengan 2,5 sehingga akan menghasilkan skor SUS. Selanjutnya total skor SUS dibagi dengan jumlah responden, sehingga didapatkan nilai rata-rata skor SUS.

Tabel 6.66 Hasil Rekapitulasi dan Perhitungan Kuisioner SUS Sisi Pelanggan

Responden	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	Skor SUS
R1	5	3	4	3	4	1	5	2	4	3	75
R2	4	2	4	3	4	3	3	2	4	3	65
R3	5	3	5	1	4	2	5	1	5	1	90
R4	3	3	4	2	5	1	4	1	5	1	87.5
Total Skor SUS											317.5
Rata-Rata Skor SUS											79.4

Tabel 6.67 Hasil Rekapitulasi dan Perhitungan Kuisioner SUS Sisi Pemilik Rental

Responden	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	Skor SUS
R1	4	2	4	2	3	2	4	2	4	2	72.5
Total Skor SUS											72.5
Rata-Rata Skor SUS											72.5

6.3 Analisis Hasil Pengujian

Tahapan analisis hasil pengujian bertujuan untuk mendapatkan kesimpulan dari hasil pengujian Sistem Penyewaan Kendaraan Kota Malang. Analisis akan dilakukan terhadap semua hasil pengujian yang dilakukan penelitian ini, meliputi analisis hasil pengujian unit, analisis hasil pengujian integrasi, analisis hasil pengujian validasi dan analisis pengujian *usability*.

6.3.1 Analisis Hasil Pengujian Unit

Analisis hasil dari pengujian unit yang telah dilakukan dengan melihat apakah fungsi atau kebutuhan fungsional yang telah diimplementasikan dalam unit modul yang telah diuji memiliki kesesuaian dengan perancangan sistem yang sebelumnya telah dirancang. Berdasarkan hal tersebut, dapat disimpulkan bahwa unit modul yang diuji sebanyak 2 *method* dari kode program sudah memiliki kesesuaian

dengan fungsi atau kebutuhan fungsional yang telah dirancang pada tahap perancangan.

6.3.2 Analisis Hasil Pengujian Integrasi

Analisis hasil dari pengujian integrasi yang telah dilakukan dengan melihat apakah fungsi atau kebutuhan fungsional yang telah diimplementasikan dalam unit modul dapat saling berintegrasi dengan baik dan juga dengan melihat apakah method integrasi yang diuji sudah memiliki kesesuaian dengan perancangan sistem yang sebelumnya telah dirancang. Pada bagian ini, 2 *method* yang saling berintegrasi diuji dan dapat disimpulkan bahwa method integrasi yang terdiri dari beberapa unit modul dari program sudah memenuhi kebutuhan fungsional seperti yang telah dirancangan pada tahap perancangan.

6.3.3 Analisis Hasil Pengujian Validasi

Analisis hasil pengujian validasi dilakukan dengan membandingkan antara hasil uji dengan perancangan sistem yang telah dilakukan pada tahap perancangan. Jika hasil uji mempunyai kesesuaian dengan perancangan sistem, maka sistem tersebut adalah valid karena telah mengimplementasikan fungsi atau kebutuhan fungsional pada tahapan perancangan. Apabila menghasil hasil yang tidak valid maka sistem tersebut belum memenuhi semua kebutuhan fungsional yang terdapat pada tahapan perancangan. Pada penelitian ini terdapat 42 kebutuhan fungsional, namun hanya didapatkan 41 kebutuhan fungsional yang valid dan 1 lainnya tidak valid yaitu kebutuhan fungsional melakukan *filter* pencarian pada sisi pelanggan dikarenakan keterbatasan penggunaan firebase *real-time database* saat ini yang belum mendukung adanya *query* dengan menggunakan *multiple where clause*.

6.3.4 Analisis Hasil Pengujian *Usability*

Analisis hasil pengujian *usability* dengan menggunakan metode SUS dapat disimpulkan dari nilai akhir atau skor SUS. Didalam SUS, terdapat *range* penilaian untuk menentukan apakah sistem yang telah dikembangkan diterima oleh pengguna atau tidak. Skor SUS dengan rentang penilaian 0 – 50 termasuk kedalam kategori *Not Acceptable*, skor SUS 51 – 70 termasuk dalam kategori *Marginal* dan skor SUS dengan rentang 71 – 100 termasuk dalam kategori *Acceptable*. Nilai rata-rata skor SUS yang didapatkan pada sisi pelanggan yaitu 79,4 dimana hasil penilaian tersebut di dominasi dari pertanyaan nomor 9 yang mempunyai nilai tertinggi. Sedangkan nilai akhir skor SUS pada sisi pemilik rental didapatkan sebesar 72,5 dimana pertanyaan nomor 1, 3, 7 dan 9 mempunyai nilai yang tinggi. Kedua penilaian tersebut disimpulkan dari pertanyaan ganjil yang memuat penilaian positif terhadap sistem. Sehingga dapat disimpulkan bahwa Sistem Penyewaan Kendaraan Kota Malang Berbasis Android mudah digunakan dan dapat diterima oleh pengguna.

BAB 7 PENUTUP

7.1 Kesimpulan

Berdasarkan hasil perancangan, implementasi dan pengujian yang telah dilakukan pada penelitian ini, dapat diambil kesimpulan sebagai berikut :

1. Pada hasil analisis kebutuhan atau rekayasa kebutuhan penelitian ini didapatkan kebutuhan fungsional sebanyak 42 kebutuhan fungsional yang terbagi menjadi beberapa sisi yaitu 2 kebutuhan fungsional pada sisi pengguna, 20 kebutuhan fungsional pada sisi pelanggan dan 20 kebutuhan fungsional pada sisi pemilik rental. Kebutuhan ini didapatkan dari spesifikasi hasil elisitasi kebutuhan dengan menggunakan *story card*. Kebutuhan fungsional tersebut didapatkan berdasarkan aktvitas iterasi pada siklus UI *prototyping* sebanyak 2 kali iterasi. Pada penelitian ini didapatkan kebutuhan non-fungsional *usability*.
2. Pada bagian perancangan penelitian ini dibagi menjadi perancangan arsitektural, perancangan basis data, perancangan diagram *sequence,* perancangan diagram *class,* manajemen pola perancangan, perancangan antarmuka dan perancangan algoritme. Perancangan basis data pada penelitian ini menggunakan skema diagram karena pada penelitian ini menggunakan firebase *real-time database* yang berbasis NoSQL. Perancangan antarmuka pada penelitian sesuai dengan metode pengembangan MASAM yaitu dengan menggunakan UI *prototyping* yang dimana perancangan antarmuka akan menggambarkan kebutuhan fungsional dari calon pengguna yang di dapatkan dari hasil *story card workshop* iterasi 1 dan iterasi 2.
3. Tahapan implementasi yang dilakukan pada penelitian ini dengan menggunakan IDE Android Studio dan menggunakan media penyimpanan basis data Firebase *real-time database*. Pada tahapan implementasi ini juga terdapat bagian yang di adopsi dari metode pengembangan MASAM, yaitu *Environment setup* atau pengaturan lingkungan, siklus rilis dan siklus iterasi. Dari tahapan siklus iterasi pada penelitian ini menerapkan dua kali iterasi yang dimana pada iterasi pertama mengerjakan 2 kebutuhan fungsional dari sisi pengguna, 10 kebutuhan fungsional dari sisi pelanggan dan 11 kebutuhan fungsional dari sisi pemilik rental. Selanjutnya pada iterasi kedua dikerjakan 10 kebutuhan fungsional dari sisi pelanggan dan 9 kebutuhan fungsional dari sisi pemilik rental.
4. Bagian terakhir dari penelitian ini adalah dilakukannya pengujian yang terbagi menjadi beberapa pengujian yaitu pengujian unit, integrasi, validasi dan *usability*. Pada pengujian unit dan integrasi sistem sudah berhasil melakukan semua jalur pengujian. Lalu pada pengujian validasi terhadap 42 kebutuhan fungsional dihasilkan 41 kebutuhan fungsional valid atau berhasil dan 1 kebutuhan fungsional invalid atau gagal.

5. Selanjutnya adalah pengujian *usability* yang termasuk kedalam pengujian a*cceptance* untuk mengetahui tanggapan dan kepuasan calon pengguna. Hasil dari pengujian SUS, didapatkan sebesar 79.4 untuk sisi pelanggan dan 72.5 untuk sisi pemilik rental. Maka dengan itu dapat disimpulkan bahwa sistem ini dapat diterima oleh pengguna dan dapat dengan mudah digunakan oleh pengguna untuk melakukan pencarian dan penyewaan kendaraan di Kota Malang.

7.2 Saran

Saran yang dapat diberikan dari hasil penelitian ini untuk mendapatkan suatu penelitian yang lebih baik kedepannya diantaranya sebagai berikut :

1. Pada fitur pembayaran akan lebih mudah apabila menyediakan fitur pembayaran yang otomatis dapat melakukan konfirmasi atas pembayaran sewa yang dilakukan oleh pelanggan.
2. Pada fitur *filter* hasil pencarian masih menghasilkan hasil yang *invalid* dikarenakan didalam penggunaan basis data firebase *real-time* belum mendukung fitur untuk membaca data dengan lebih dari 1 kondisi (*multiple where clause*), untuk menghasilkan hasil yang valid dapat digunakan firestore yang merupakan salah satu layanan yang disediakan firebase karena firestore mendukung fitur untuk membaca data dengan lebih dari 1 kondisi yang disebut dengan *compound query*.

DAFTAR PUSTAKA

Alashqar, A. M., Elfetouh, A. A. & El-Bakry, H. M., 2015. Requirement Engineering for Non-Functional Requirements. *International Journal of Information and Communication Technology Research,* Volume V, pp. 21-27.

Alluri, A., 2012. *Usability Testing of Android Applications.* San Diego: s.n.

Ambler, S., 2013. *Introduction to Test Driven Development (TDD).* [Online] Available at: http://agiledata.org/essays/tdd.html [Diakses 09 11 2017].

Badan Pusat Statistik, 2015. *Perkembangan Jumlah Kendaraan Bermotor Menurut Jenis, 1949-2015.* [Online] Available at: https://www.bps.go.id/linkTableDinamis/view/id/1133 [Diakses 1 March 2017].

Bangor, A., Kortum, P. & Miller, J., 2009. Determining What Individual SUS Scores Mean: Adding an Adjective Rating Scale. *Journal Of Usability Studied,* 4(3), pp. 114-123.

Cohn, M., 2004. *User Stories Applied: For Agile Software Development.* Addison: Wesley Professional.

Edwin, N. M., 2014. Software Frameworks, Architectural and Design Patterns. *Journal of Software Engineering and Applications,* Issue 7, pp. 670-678.

Fowler, M., 2003. *UML Distilled Third Edition A Brief Guide to the Standart Object Modelling Language.* Third penyunt. Boston: Addison Wesley.

G. P., Hamsini, R. & R, S. G., 2016. Agile Development Methodology and Testing for Mobile Applications - A Survey. *International Journal of New Technology and Research (IJNTR),* 2(9), pp. 98-101.

Grand Trans Malang, 2016. *Grand Trans Malang.* [Online] Available at: https://grandrentcarmalang.com/ [Diakses 2 April 2017].

Greer, D. & Conradi, R., 2009. Software Project Initiation and Planning - An Empirical Study - an empirical study. *The Institution of Engineering and Technology ,* III(5), pp. 356-368.

Jeong, Y. J., Lee, J. H. & Shin, G. S., 2008. Development Process of Mobile Application SW Vased on Agile Methodology. *International Conference on Advanced Communications Technology,* pp. 362-366.

KhedKar, S. & Thube, S., 2017. Real Time Databases for Application. *International Research Journal of Engineering and Technology,* 04(06), pp. 2078 - 2082.

Kuchana, P., 2004. *Software Architecture Design Patterns in Java.* U.S: s.n.

Lant, M., 2010. *Software Development, Agile Methods and the Intersection of People Process and Technology.* [Online]

Available at: https://michaellant.com/2010/05/21/how-to-easily-prioritize-your-agile-stories/
[Diakses 16 May 2017].

Leau, Y. b., Loo, W. K., Tham, W. Y. & Tan, S. F., 2012. Software Development Life Cycle AGILE vs Traditional Approaches. *International Conference on Information and Network Technology,* Volume 37, pp. 162-167.

Luedke, M., 2015. *Common Design Patterns for Android.* [Online] Available at: https://www.raywenderlich.com/109843/common-design-patterns-for-android
[Diakses 6 12 2017].

Mustaqbal, M. S., Firdaus, R. F. & Rahmadi, H., 2015. PENGUJIAN APLIKASI MENGGUNAKAN BLACK BOX TESTING BOUNDARY VALUE ANALYSIS. *Jurnal Ilmiah Teknologi Informasi Terapan,* Volume I, pp. 31-37.

Nielsen, J., 2000. *Why You Only Need to Test with 5 Users.* [Online] Available at: https://www.nngroup.com/articles/why-you-only-need-to-test-with-5-users/
[Diakses 22 12 2017].

Onions, P. & Patel, C., 2009. Enterprise SoBA: Large-scale Implementation of Acceptance Test Driven Story Cards. *IEEE International Conference on Information Reuse and Intergration (IRI),* pp. 105-109.

PERMENKOMINFO, 2017. *Peraturan Menteri Komunikasi dan Informatika Nomor 12 Tahun 2016 tanggal 4 Agustus 2016.* [Online] Available at: https://jdih.kominfo.go.id/produk_hukum/view/id/541/t/peraturan+menteri+komunikasi+dan+informatika++nomor+12+tahun+2016+tanggal+4++agustus+2016
[Diakses 3 12 2017].

Perry, W. E., 2006. *Effective Methods for Software Testing includes Complete Guidelines and Checklists.* 3rd penyunt. Canada: Wiley Publishing Inc.

Pressman, R. S., 2010. *Software Engineering Practitioner Approach.* 7th penyunt. New York: McGraw-Hill.

Rambaugh, J., Jacobson, I. & Booch, G., 2005. *The Unified Modeling Language Reference Manual.* 2nd penyunt. Canada: Addison-Wesley.

Sharfina, Z. & Santoso, H. B., 2016. An Indonesian Adaptation of the System Usability Scale (SUS). *ICACSIS,* pp. 145 - 148.

Singh, A., Sharma, S. & Singh, S., 2016. Android Application Development using Android Studio and PHP Framework. *International Journal of Computer Applications,* pp. 5-8.

Sommerville, I., 2011. *Software Engineering.* 9 penyunt. Boston: s.n.

Stapic, Z., 2013. *A PROPOSAL OF AN ONTOLOGY-BASED METHODOLOGICAL FRAMEWORK FOR MULTI-PLATFORM MOBILE APPLICATIONS DEVELOPMENT.*

Alcalá de Henares (Madrid): University of Alcalá Computer Science Department, Postgraduate School Doctoral program “Information and Knowledge Engineering”.

Statista, 2016. *Stastia Car Rentals.* [Online] Available at: https://www.statista.com/outlook/270/120/car-rentals/indonesia# [Diakses 3 April 2017].

Statista, 2017. *Market share held by mobile operating systems in Indonesia from January 2012 to February 2017.* [Online] Available at: https://www.statista.com/statistics/262205/market-share-held-by-mobile-operating-systems-in-indonesia/ [Diakses 10 April 2017].

Sugiyono, 2011. *Metode Penelitian Kuantitatif Kualitatif DAN R&D.* Bandung: Alfabeta.