PENGEMBANGAN APLIKASI RESERVASI DAN MANAJEMEN *MARKETPLACE HOMESTAY* BERBASIS WEB (STUDI KASUS: KOTA PEKANBARU)

SKRIPSI

Untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun oleh:
Habib Yafi Ardi
NIM: 155150207111136

PROGRAM STUDI TEKNIK INFORMATIKA
JURUSAN TEKNIK INFORMATIKA
FAKULTAS ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA
MALANG
2019

PENGESAHAN

PENGEMBANGAN APLIKASI RESERVASI DAN MANAJEMEN *MARKETPLACE* *HOMESTAY* BERBASIS WEB (STUDI KASUS: KOTA PEKANBARU)
SKRIPSI

Untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun Oleh :
Habib Yafi Ardi
NIM: 155150207111136

Skripsi ini telah diuji dan dinyatakan lulus pada
13 September 2019
Telah diperiksa dan disetujui oleh:

Dosen Pembimbing I	Dosen Pembimbing 2
Bayu Priyambadha, S.Kom, M.Kom	Arief Andy Soebroto, S.T., M.Kom.
NIP: 19820909 200812 1 004	NIP: 19720425 199903 1 002

Mengetahui
Ketua Jurusan Teknik Informatika

Tri Astoto Kurniawan, S.T., M.T., Ph.D.
NIP: 19710518 200312 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 unsurunsur 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, 6 Juli 2019

Habib Yafi Ardi

NIM: 155150207111136

PRAKATA

Puji syukur kehadirat Allah SWT yang telah melimpahkan rahmat, taufik dan hidayah-Nya sehingga laporan skripsi yang berjudul "Pengembangan Aplikasi Reservasi Dan Manajemen *Marketplace Homestay* Berbasis Web (Studi Kasus: Kota Pekanbaru)" 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. Allah SWT yang selalu memberikan kemudahan dalam setiap proses penulisan skripsi ini.
2. Kedua orang tua dan keluarga atas doa dan dukungan yang diberikan kepada penulis selama proses studi.
3. Bapak Bayu Piyambadha, S.Kom., M.Kom. dan Bapak Arief Andy Soebroto, S.T., M.Kom. selaku dosen pembimbing penulis yang telah sabar mengarahkan dan membimbing penulis dalam pengerjaan skripsi ini,.
4. Bapak Agus Wahyu Widodo, S.T, M.Cs. selaku Ketua Program Studi Teknik Informatika.
5. Bapak Tri Astoto Kurniawan, S.T, M.T, Ph.D selaku Ketua Jurusan Teknik Informatika.
6. Seluruh Bapak dan Ibu dosen Fakultas Ilmu Komputer Universtas Brawijaya atas segala bimbingan dan ilmu yang diberikan kepada penulis.
7. Teman-teman seperjuangan teknik informatika 2015 atas bantuan, motivasi dan meluangkan waktu untuk berdiskusi selama ini.
8. Dan semua teman penulis yang silih berganti menemani perjuangan penulis.

Penulis menyadari bahwa dalam pengerjaan skripsi ini masih banyak kekurangan, sehingga saran dan kritik yang membangun sangat penulis harapkan.

Akhir kata penulis berharap skripsi ini dapat memberikan manfaat bagi semua pihak yang membacanya maupun menggunakannya.

Malang, 6 Juli 2019

Penulis

habibyafi45@gmail.com

ABSTRAK

Habib Yafi Ardi, Pengembangan Aplikasi Reservasi Dan Manajemen *Marketplace Homestay* Berbasis Web (Studi Kasus: Kota Pekanbaru).

Pembimbing: Bayu Priyambadha, S.Kom., M.Kom. dan Arief Andy Soebroto,S.T., M.Kom

Pada Provinsi Riau, Kota Pekanbaru, pada kategori akomodasi yaitu penyewaan tempat tinggal di tahun 2012, mengalami pesatnya perkembangan akibat dampak dari Pekan Olahraga Nasional (PON) 2012, masyarakat berlomba-lomba untuk membangun properti seperti hotel, *homestay, kost,* kondominium, dan wisma untuk memenuhi kebutuhan para pendatang dari luar provinsi Riau. Namun pada tahun 2013 penggunaan dari properti tersebut berubah alih dari yang sebelumnya diperuntukkan kepada atlet, panitia, dan penonton diluar provinsi yang bersifat masif menjadi orang-orang yang ingin bekerja di Pekanbaru. Sehingga properti-properti kecil seperti *homestay* dan *kost* menjadi kalah bersaing dengan hotel-hotel yang sudah memiliki pasar tersendiri. Berdasarkan masalah-masalah yang telah dirumuskan diatas, maka dikembangkanlah sebuah perangkat lunak yang berbasis web yaitu Homie. Tujuan dari perangkat lunak ini adalah untuk menciptakan suatu pasar bagi para pemilik properti persewaan tempat tinggal dimana pemilik dan pelanggan penyewa tempat tinggal dapat dipertemukan. Dari hasil penelitian ini didapatkan dua puluh dua kebutuhan fungsional dan satu kebutuhan non-fungsional dari tahap analisis kebutuhan, kemudian perancangan arsitektur, perancangan komponen, perancangan basisdata, dan perancangan antarmuka yang digunakan untuk menghasilkan implementasi sistem. Kemudian pada tahap pengujian diperolehlah hasil tiga pengujian unit, satu pengujian integrasi, dan dua puluh dua pengujian validasi yang memberikan nilai 100% valid. Kemudian pada pengujian kompatibilitas didapatkan hasil bahwa aplikasi Homie dapat berjalan di 8 jenis perambah diantaranya edge, firefox, safari, google chrome, perangkat ios, dan perangkat android.

Kata kunci: *marketplace, homestay,* laravel*,* react*,* javascript*, payment gateway*

ABSTRACT

Habib Yafi Ardi, ***Development of Web Based Marketplace Application for Homestay Reservation and Management (Case Study: Pekanbaru City).***

***Supervisors*: Bayu Priyambadha, S.Kom., M.Kom. dan Arief Andy Soebroto,S.T., M.Kom**

In Riau Province, Pekanbaru City, in the accommodation category of housing rentals in 2012, experiencing rapid development due to the impact of the 2012 Pekan Olahraga Nasional (PON), communities competed to build properties such as hotels, homestays, boarding houses, condominiums, and guesthouses to meet the needs of migrants from outside Riau province. But in 2013 the use of these properties changed over from previously intended athletes, organizers, and spectators outside of the province which became massive people who wanted to work in Pekanbaru. So that small properties such as homestays and boarding houses are unable to compete with hotels that already have their own markets. Based on the problems that have been formulated above, a web-based software is developed, namely Homie. The purpose of this software is to create a market for the owners of residential rental properties where owners and customers of residential tenants can be reunited. From the results of this study twenty-two functional requirements and one non-functional requirement were obtained from the needs analysis stage, then architectural design, component design, database design, and interface design used to generate system implementation. Then in the testing phase the results of three unit tests, one integration test, and twenty-two validation tests that provide a 100% valid value are obtained. Then on compatibility testing the results are that Homie applications can run on 8 types of browsers including edge, firefox, safari, google chrome, iOS devices, and android devices.

Keywords: *marketplace, homestay, laravel, react, javascript, payment gateway*

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

BAB 1 PENDAHULUAN

1.1 Latar Belakang

Perkembangan ekonomi tanpa migas di provinsi Riau tepatnya pada kategori penyediaan akomodasi dan makan minum yang mana didalamnya terdapat jasa penyewaan tempat tinggal seperti *homestay, kost,* dan juga kondominium selalu meningkat dari tahun 2015 sampai dengan 2017. Pada pertumbuhan yang naik pada kategori tersebut, terdapat pula pertumbuhan Produk Domestik Regional Bruto (PDRB) yang turun drastis pada kategori tersebut. Penurunan yang drastis tersebut terjadi pasca kegiatan besar nasional yaitu Pekan Olahraga Nasional (PON) pada tahun 2012 yang mana tuan rumah dari kegiatan tersebut adalah provinsi Riau. Pada tahun 2012, laju pertumbuhan PDRB di provinsi Riau adalah 10.36% kemudian pada tahun 2013 laju pertumbuhan PDRB di provinsi Riau menjadi 5.93, dapat dibandingkan penurunan hampir mencapai 50% (BPS, 2017).

Pada kategori akomodasi yaitu penyewaan tempat tinggal pada tahun 2012, seiring pesatnya perkembangan kota pekanbaru akibat dampak dari Pekan Olahraga Nasional (PON) 2012, masyarakat berlomba-lomba untuk membangun properti seperti hotel, *homestay, kost,* kondominium, dan wisma untuk memenuhi kebutuhan para pendatang dari luar provinsi Riau (Kompas, 2010). Namun pada tahun 2013 penggunaan dari bisnis properti tersebut berubah alih dari yang sebelumnya diperuntukkan kepada atlet, panitia, dan penonton diluar provinsi yang bersifat masif menjadi orang-orang yang ingin bekerja ataupun hanya singgah di Pekanbaru. Setelah PON berlalu, kebutuhan akan tempat penginapan menurun drastis tidak terkecuali untuk *homestay* (Detik, 2012).

Homestay-homestay kecil yang bergerak dibidang penyewaan tempat tinggal tersebut memiliki kelemahan yang sangat vital dibandingkan dengan hotel-hotel yang ada, diantaranya tidak memiliki resepsionis yang siap siaga selama 24 jam, manajemen *homestay* yang tidak terkelola dengan baik seperti pencatatan pemasukan keuangan dan pengaturan ketersediaan kamar, dan promosi yang masih bersifat konvensional dengan cara memasang papan-papan kecil didepan bangunan. Resepsionis disini berperan penting sebagai penunjang untuk transaksi yang bersifat kontan atau tunai, kemudian manajemen *homestay* untuk menjaga keberlangsungan bisinis, dan promosi untuk mempermudah pendatang mendapatkan informasi akan bisnis tersebut dan memperkuat profit dari bisnis tersebut.

Berdasarkan pendefinisian masalah yang dirumuskan sebelumnya, maka dikembangkanlah sebuah perangkat lunak yang berbasis web yaitu Homie. Tujuan dari perangkat lunak ini adalah untuk menciptakan suatu pasar bagi para pemilik *homestay* persewaan tempat tinggal dimana pemilik dan pelanggan penyewa tempat tinggal dapat dipertemukan, sehingga permasalahan seperti sulitnya mencari *homestay-homestay* kecil dapat diselesaikan. Aplikasi Homie diharapkan mampu membantu permasalahan-permasalahan yang telah dipaparkan melalui fitur-fitur yang ada didalamnya sebagai solusi yang dapat menguntungkan pemilik

homestay penyewaan tempat tinggal dan pelanggan yang ingin menyewa tempat tinggal tersebut.

Fitur-fitur yang akan dihadirkan oleh perangkat lunak ini berupa pemesanan secara *online* dan pembayaran *cashless* atau nontunai sehingga tidak memerlukan peran resepsionis dalam melakukan transaksi. Pada fitur pembayaran nontunai akan dibantu dengan aplikasi pihak ketiga yaitu *payment gateway*. Selanjutnya perangkat lunak ini akan memiliki fitur manajemen kepada para pemilik *homestay* yang telah terdaftar didalam sistem yang mana didalamnya terdapat fitur-fitur yang lebih spesifik dalam mengelola bisnis *homestay* tersebut diantaranya laporan statistik pemasukan, manajemen kamar, penentuan harga, dan sebagainya. Perangkat lunak ini akan dapat diakses secara *online* di internet sehingga siapapun dapat mendapatkan informasi-informasi terkait persewaan tempat tinggal ini di internet.

Pengembangan aplikasi reservasi dan manajemen *marketplace homestay* akan menggunakan teknologi ReactJs. Performa (*speed*) dari aplikasi merupakan faktor utama pada *design* dalam mencapai kesuksesan suatu aplikasi (Turban et al., 2015). Mengacu terhadap penjelasan Turban pada bukunya, maka penggunaan ReactJs cocok diterapkan dalam pengembangan aplikasi ini. Alasan dari baiknya performa yang ditawarkan olerh ReactJs adalah Bahasa pemrograman yang digunakan, yaitu Javascript dan konsep pengembangan *reusable component*. Selain itu, kompatibilitas merupakan salah satu faktor utama yang mempengaruhi pengadopsian dari *e-commerce* (Alam et al., 2007) sehingga pada bagian kebutuhan non-fungsional akan didefinisikan terkait kompatibilitas sistem dan juga akan dilakukan pengujian kompatibilitas.

1.2 Rumusan Masalah

Dengan masalah yang terdapat latar belakang diatas, maka dirumuskan bahwa permasahalan yang akan dibahas adalah sebagai berikut:

1. Bagaimanakah hasil analisis dan spesifikasi kebutuhan dalam mengembangkan perangkat lunak Homie untuk mewadahi bisnis persewaan *homestay*?
2. Bagaimanakah rancangan aplikasi Homie yang dapat memenuhi hasil dari analisis kebutuhan dan spesifikasi kebutuhan?
3. Bagaimanakah implementasi perangkat lunak Homie yang sesuai dengan rancangan sistem yang telah dilakukan sebelumnya?
4. Bagaimanakah hasil pengujian aplikais Homie agar kebutuhan dapat dipenuhi dari aplikasi yang telah dikembangkan?

1.3 Tujuan

Dengan masalah yang terdapat latar belakang diatas, didapatkanlah beberapa poin tujuan penelitian ini diantaranya:

1. Menganalisis dan menyusun spesifikasi kebutuhan untuk mengembangkan perangkat lunak Homie.

2. Merancang aplikasi Homie yang dapat memenuhi hasil dari analisis kebutuhan dan spesifikasi kebutuhan.
3. Mengimplementasikan sistem perangkat lunak sesuai dengan hasil perancangan yang telah dilakukan sebelumnya.
4. Menguji aplikasi Homie agar kebutuhan yang dirumuskan dapat terpenuhi.

1.4 Manfaat

Penelitian ini memiliki beberapa poin yang diharapkan menjadi manfaat, adalah sebagai berikut:

1. Sebagai sarana promosi penyewaan tempat tinggal.
2. Memudahkan pemilik bisnis penyewaan tempat tinggal dalam hal promosi, pemesanan dan transaksi.
3. Memudahkan pemilik bisnis penyewaan tempat tinggal dalam menjaga keberlangsungan bisnis tersebut dengan manajemen yang benar.
4. Memudahkan penyewa tempat tinggal mencari informasi dan melakukan pemesanan beserta transaksi yang lebih praktis.

1.5 Batasan Masalah

Terdapat beberapa poin yang memberikan batasan terhadap penelitian ini agar terhindar dari penyimpangan, yaiut sebagai berikut:

1. Fokus wilayah yang diterapkan dalam penelitian ini adalah Kota Pekanbaru.
2. Fokus sistem yang dikembangkan adalah pemesanan secara *online* dan transaksi nontunai menggunakan *payment gateway*.
3. Aplikasi dibangun hanya dengan PHP dan Javascript sebagai Bahasa pemrograman, HTML, dan *styling* menggunakan CSS.
4. Aplikasi dibangun menggunakan manajemen basis data MySQL.
5. Aplikasi hanya dapat dijalankan perambah edge, firefox, safari, google chrome, perangkat ios, dan perangkat android.
6. Penelitian ini akan menggunakan *Software Development Life Cycle* (SDLC) model *waterfall* yang berfokus pada tahap rekayasa kebutuhan, perancangan, implementasi, dan pengujian.

1.6 Sistematika Pembahasan

Sistematika pembahasan laporan bertujuan untuk memaparkan alur pengerjaan secara umum, berikut uraiannya:

BAB 1: PENDAHULUAN

Bagian ini akan menjelaskan latar belakang penelitian, rumusan masalah penelitian, tujuan, manfaat yang didapatkan, batasan masalah dalam pengembangan dan bagaimana sistematika penulisannya dalam pengembangan Aplikasi Reservasi dan Manajemen *Marketplace Homestay* Berbasis Web.

BAB 2: LANDASAN KEPUSTAKAAN

Bagian ini akan menjelaskan mengenai dasar ilmu yang menyangkut dalam penelitian sebagai acuan dalam pengembangan Aplikasi Reservasi dan Manajemen *Marketplace Homestay* Berbasis Web diantaranya *payment gateway,* rekayasa perangkat lunak, model pengembangan yang diperlukan dalam penelitian ini.

BAB 3: METODOLOGI

Bagian ini akan membahas mengenai langkah yang terstruktur dalam melakukan penelitian, urutan tersebut bermula pada elisitasi kebutuhan dan rekayasa kebutuhan, merancang perancangan berdasarkan kebutuhan, melakukan implementasi sistem dengan acuan sesuai dengan perancangan yang telah didefinisika.

BAB 4: REKAYASA KEBUTUHAN

Bagian ini akan dijelaskan proses analisa terhadap kebutuhan kebutuhan dan memodelkan kebutuhan untuk penerapan pengembangan Aplikasi Reservasi dan Manajemen *Marketplace Homestay* Berbasis Web diantaranya kebutuhan fungsiobal dan non-fungsional, kemudian pemodelan kebutuhab.

BAB 5: PERANCANGAN DAN IMPLEMENTASI SISTEM

Bagian ini akan menjelaskan perihal proses dalam merancang dan memaparkan implementasi perangkat lunak pada Aplikasi Reservasi dan Manajemen *Marketplace Homestay* Berbasis Web. Poin-poin yang terdapat dalam perancangan aplikasi diantaranya melakukan perancangan terhadap arsitektur aplikais, komponen aplikasi, database aplikasi, dan antarmuka aplikasi. Kemudian untuk segmen implementasi akan menghasilkan spesifikasi sistem, melakukan implementasi terhadap diantaranya kode program, database, dan antarmuka.

BAB 6: PENGUJIAN SISTEM

Bagian ini akan dijelaskan mengenai pengerjaan pengujian apliaksi diantaranya pengujian unit, integrasi, validasi, dan kompatibilitas terhadap Aplikasi Reservasi dan Manajemen *Marketplace Homestay* Berbasis Web yang dikembangkan.

BAB 7: PENUTUP

Pada Bagian ini akan dijelaskan perihal kesimpulan yang telah didefinisikan melalui tiap-tiap tahap pengerjaan aplikasi, dan juga terdapat saran untuk dapat dipergunakan dalam penelitian kedepannya terhadap pengembangan perangkat lunak Aplikasi Reservasi dan Manajemen *Marketplace Homestay* Berbasis Web.

BAB 2 LANDASAN KEPUSTAKAAN

Landasan kepustakaan memiliki dua sub bab umum. Kajian pustaka merupakan pendefinisian akan teori dari penelitian sebelumnya namun akan digunakan dalam penelitian saat ini. Dasar teori merupakan pendefinisian perihal teori-teori yang terkait dalam perencanaan perangkat lunak. Dasar teori yang akan didefinisikan diantaranya adalah teknik dalam melakukan integrasi *payment gateway* dengan Aplikasi Reservasi dan Manajemen *Marketplace Homestay* Berbasis Web. Sleanjutnya teori dalam pengembangan perangkat lunak yaitu model *waterfall*, dan pendekatan *object oriented*. Kemudian akan dilanjutkan dengan mendefinisikan teknologi yang digunakan diantaranya, Midtrans, Laravel, *library* ReactJS, DBMS MySQL, dan SortSite.

2.1 Kajian Pustaka

Bab ini akan memamaprkan penjelasan perihal penelitian yang telah dilakukan sebelumnya dan menjadi rujukan dalam peneilitan ini. Penelitian yang dibahas adalah penelitian yang berada dalam lingkup sistem informasi, *marketplace*, dan *paymeny gateway*. Terdapat tiga penelitian sebelumnya dengan batas penelitian dalam lima tahun. Penelitian-penelitain tersebut telah dipaparakan pada Tabel 2.1 dengan penjelasan berupa judul penelitian, objek penelitian, dan perbedaan dari penelitian sebelumnya kemudian renacana penelitian sekarang berdasarkan objek penelitian.

Penelitian yang pertama dilakukan oleh Aryanto (2017) memiliki studi kasis di Hotel Kesawan. Hotel ini adalah salah satu aktor dalam pergerakan bisnis properti didaerah itu. Cara pemesanan kamar yang dilakukan pada hotel ini masih bersifat konvensional yaiti dengan telepon, fax, agen travel, atau dengan cara mendatangi hotel tersebut. Karena metode yang digunakan masih konvensional, masyarakat atau tamu yang akan memesan kamar merasakan kesulitan dalam melakukan pengecekan terhadap ketersediaan kamar dan proses pemesenan yang dapat dibilang kurang efisien. Dengan melihat teknologi yang ada saat ini, sangat disayangkan jika hotel tersebut tidak memanfaatkan teknologi yang ada saat ini. Hal tersebut merupakan bagian yang sangat penting dalam dunia perhotelan agar dapat melakukan persebaran infomrasi yang lebih luas kepada masyarakat dan juga mempermudah dalam melakukan reservasi serta promosi-promosi. Penerapan sistem reservasi kamar dengan menggunakan web pada hotel Kesawan akan memiliki beberapa kelebihan seperti mempermudah proses pemesanan kamar dari customer, proses pengecekan ketersediaan kamar yang mudah, meningkatkan prestise perusahaan dan menghemat biaya promosi dari hotel Kesawan.

Penelitian yang kedua dilakukan oleh Suryanto (2018) Penerapan E-*Marketplace* Pada Distro Silve Squad. Penerapan aplikasi E-*marketplace* memberikan dampak yang sangat besar. Dengan model transaksi yang lebih fleksibel, ketika customer bisa melakukan pencarian dan pembelian pakaian dari banyaknya penjual dan melakukan proses jual beli dan terdapatnya fitur yang

dapat sangat mempermudah penggunanya serta didalam implementasi dilapangan dapat dilihat secara langsung proses jual beli dengan internet, melakukan transaksi melalui media elektronik yang berperan sebagai penghubung dari sis penjual dan pembeli.

Kemudian penelitian yang terakhir dilakukan oleh Febriyanto, Rahardja, dan Alnabawi (2018) yang menerapkan integrase pihak ketiga berupa *paymeny gateway* yaitu midtrans. iPanda adalah tempat menjual hosting dan doamain yang bersifat online dan dapat diakses dengan mudah melalui website dengan berbaai perambah. Pada umumnya digunakan untuk mahasiswa tingkat akhir yang sedang dalam proses pengerjaan dan membutuhkan hosting dan domain untuk tugas mereka yang membutuhkan koneksi daring. Berhubung dengan pesatnya transaksi yang terjadi maka sistem harus dapat membantu proses transaksi secara efektif dan efisien. Sehingga diterapkan metode pembayaran yang menggunakan *paymeny gateway* yang bernama Midtrans untuk membantu porses bisnis tersebut. Berbagai metode pembayaran telah difasilitasi oleh midtrans sehingga iPanda memiliki proses bisnis transaksi yang lebih efektif dan efisien.

Tabel 2.1 Kajian Pustaka

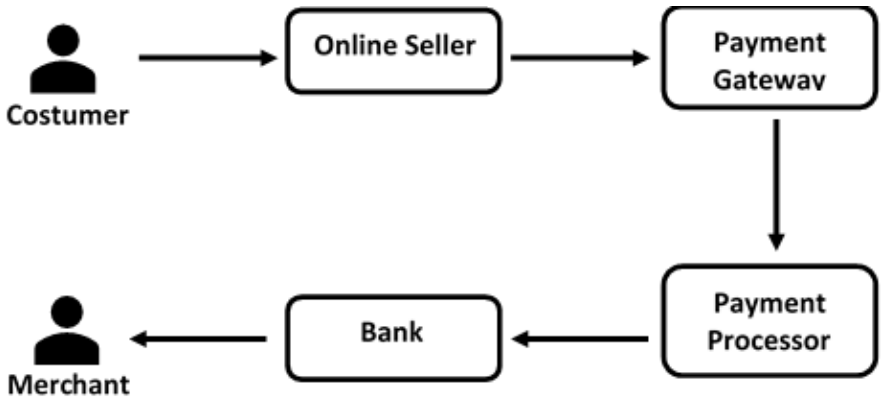
No.	Judul Penelitian	Objek Penelitian	Perbedaan	
			Penelitian Terdahulu	Rencana Penelitian
1	Penerapan dan Perancangan Sistem Informasi Pemesanan Kamar Hotel Berbasis Web (Studi Kasus pada Hotel Kesawan) (Aryanto, 2017).	Menerapkan reservasi kamar dan pengecekan sisa kamar hotel secara *online* dalam sebuah sistem informasi berbasis web.	Menerapkan reservasi kamar dan pengecekan sisa kamar hotel secara *online* dalam sebuah sistem informasi berbasis web.	Menerapkan pemesanan kamar pada tiap-tiap penyewaan tempat tinggal dan ketersediaan kamar secara *online* pada sebuah sistem informasi berbasis web.
2	Penerapan E-Marketplace pada Distro Silver Squad (Suryanto, 2018).	Menerapkan konsep *marketplace* pada Distro Silver Squad sehingga customer dapat melakukan pencarian dan	Menerapkan konsep *marketplace* pada Distro Silver Squad sehingga customer dapat melakukan pencarian dan	Menerapkan konsep *marketplace* pada *Homestay* dengan studi kasus Kota Pekanbaru sehingga customer dapat melakukan

		pembelian barang dari sekian banyak penjual.	pembelian barang dari sekian banyak penjual.	pencarian dan memesan kamar dari sekian banyak *homestay*.
3	Penerapan Midtrans sebagai Sistem Verifikasi Pembayaran pada Website iPanda (Febriyanto, Rahardja and Alnabawi, 2018).	Penggunaan midtrans dalam mekanisme transaksi pada website iPanda.	Penggunaan midtrans dalam mekanisme transaksi pada website iPanda.	Penggunaan midtrans dalam mekanisme transaksi pada aplikasi Homie.

2.2 *Payment Gateway*

Berdasarkan ketetapan yang telah dikeluarkan Bank Indonesia Nomor 18/40/PBI/2016 dalam melaksanakan proses transaksi pembayaran, yaitu *paymeny gateway* adalah layanan elektronik yang berperan dalam melakukan pembayaran melalui beberapa alat pembayaran. Layanan ini berperan besar dalam membantu proses pembayaran para pembeli kepada penjual dari sisi efisiensi. Banyak *e-commerce* ataupun toko online yang telah menggunaka layanan ini untuk membantu proses bisnis transaksi merekan. Dalam menjaga integritas layanan, terdapat beberapa syarat penting yang harus dipenuhi oleh pengguna sebelum melakukan aktivasi akun. Sehingga toko-toko online ataupun e-commerce yang telah terdaftar dapat dipastikan aman dan memberikan kenyamanan bagi *costumer* dari pengguna layanan ini (Cermati, 2017).

Proses-proses yang terjadi pada *paymeny gateway* yang pertama adalah pengiriman informasi mengenai detail transaksi setelah pembeli menentukan produk apa yang ingin dibeli ataupun jasa yang ingin digunakan ke penyedia layanan *paymeny gateway*. Setelah penyedia layanan *paymeny gateway* menerima informasi transaksi, maka akan diteruskan kepada sistem mekanisme pembayaran dari bank penjual. Kemudian mekanisme pembayaran pada bank akan melanjutkan dengan mendapatkan informasi terkait media apa yang digunakan. Setelahnya, request request akan diterima oleh bank untuk dilakukan validasi terkait status transaksi. Kemudian, mekanisme transaksi bank akan sebuah response berupa informasi-informasi terkait transaksi kepada penyedia layanan *paymeny gateway*. Response tersebut akan menjadi response lanjutan kepada clien-client yang menggunakan layanan *paymeny gateway* tersebut. Mekanisme kerja dari *paymeny gateway* dapat dilihat pada Gambar 2.1.



Gambar 2.1 Mekanisme Kerja *Payment Gateway*

Sumber: (Cermati, 2017)

2.3 E-Marketplace

Pasar elektronik berperan penting dalam era ekonomi digital saat ini. Peran-perannya antara lain adalah memfasilitasi pertukaran informasi, barang, jasa, dan pembayaran. Perdagangan pada *marketplace* memberikan nilai ekonomi terhadap berbagai kalangan diantaranya pembeli, penjual, perantara pasar, dan juga masyarakat secara luas. *Marketplace* memiliki empat fungsi utama yaitu sebagai wadah atau tempat bertemu antara penjual dan pembeli untuk memungkinkan terjadinya suatu transaksi, memungkinkan untuk terjadinya alur informasi-informasi yang relevan, menyediakan suatu layanan yang terkait dengan transaksi seperti *payment*, dan menyediakan layanan tambahan keamanan dan audit. Kemudian dari segi komponen, terdapat tiga komponen utama agar suatu *marketplace* dapat berdiri. Komponen-komponen tersebut adalah kostumer, penjual, dan produk ataupun jasa (Turban et al., 2015).

2.4 Rekayasa Perangkat Lunak

Bagian ini berisi tahapan-tahapan lengkap mulai dari spesifikasi sistem hingga perawatan sistem. Tahapan-tahapan tersebut tidak terlepas dari seluruh aspek produksi perangkat lunak. (Sommerville, 2011). Pada pendefinisiannya terbagi menjadi dua poin penting yaitu:

1. Engineering discipline

Engineer harus berada dalam batasan lingkungan dari segi organisasi dan keuangan yang bertujuan agar *engineer* dapat secara selektif memilah solusi yang efektif dan efisien dalam keterbatasan teori pendukung yang sedikit. Dalam keterbatasan itu juga *engineer* dapat membuat sebuah alat yang berdasarkan teori-teori dan metode kemudian dibantu dengan alat bantu yang cocok untuk permasalahannya. (Sommerville, 2011).

2. All aspects of software production

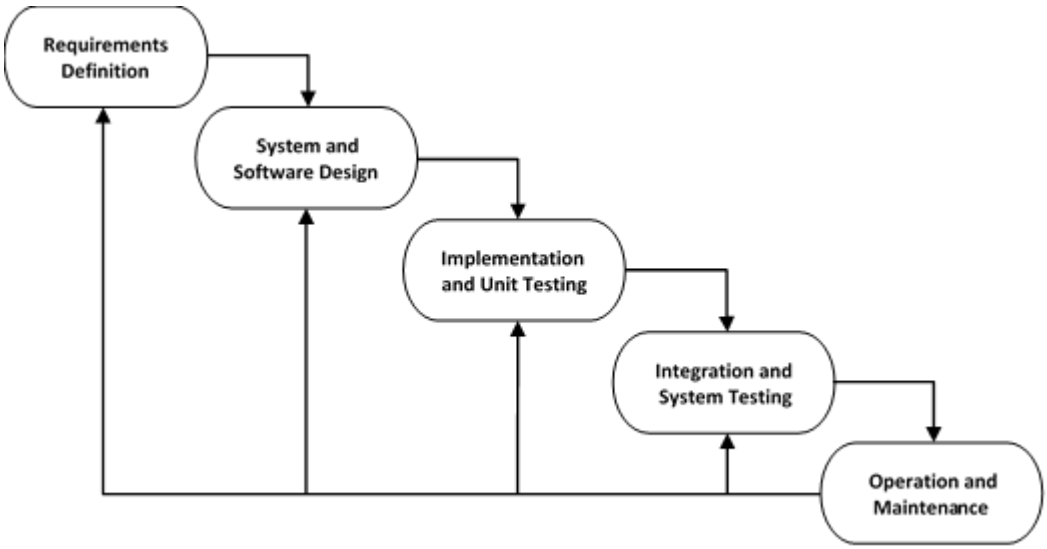
Produksi perangkat lunak merupakan tujuan utama dari rekayasa perangkat lunak. Sehingga seluruh aspek harus diperhatikan mulai dari teknis pengembangan, manajemen proyek dan pengembangan kakas bantu, serta metode-metode dan juga teori terkait.

2.5 Pengembangan Perangkat Lunak

Pada prosesnya, terdapat *Software Development Life Cycle* (SDLC) yang merupakan siklus hidup pengembangan suatu perangkat lunak. SDLC merupakan acuan umum dalam pengembangan suatu perangkat lunak. SDLC dapat dibilang sebagai *Software Process Model* yang dapat memproyeksikan proses-proeses yang ada dalam pengembangan perangkat lunak (Sommerville, 2011). Beberapa model SDLC yang cukup banyak digunakan diantaranya model *waterfall*, *iterative*, *prototypng*, *incremental*, *spiral*, RAD, dan *extreme programming*.

2.5.1 Model *Waterfall*

Model *waterfall* atau *classic life cycle* merupakan model yang pertama hadir dalam SDLC. Model ini memiliki karakteristik yang jelas dimana seluruh proses dilakukan secara berurutan mulai dari spesifikasi kebutuhan hingga perawatan dan operasi suatu perangkat lunak (Pressman, 2009). Model *waterfall* terpecah kedalam beberapa tahap sekuensial diantaranya *Requirments Definition*, *System and Software Design*, *Implementation and Unit Testing*, *Integration and System Testing*, dan *Operation and Maintenance* (Sommerville, 2011). Pada penelitian ini, tahap *Operation and Maintenance* tidak dilakukan karena tahapan yang diterapkan pada penelitian ini hanya sampai dengan tahap pengujian sesuai dengan batasan masalah. Tahapan-tahapan tersebut digambarkan dan dapat dilihat pada Gambar 2.2.



Gambar 2.2 Model *Waterfall*

Sumber: (Sommerville, 2011)

1. *Requirements Definition*

   Pendefinisian kebutuhan merupakan awalan dari proses pertama SDLC model *waterfall*. Pada tahap ini akan dilakukan analisa untuk memahami kebutuhan secara benar berdasarkan permasalahan yang ada. Kemudian tahapan ini juga bertujuan untuk mengidentifikasi apa saja yang dibutuhkan oleh sistem. Bentuk umum yang dihasilkan dari proses ini ialah Software Requirement Specification (SRS) (Kumar, Zadgaonkar and Shukla, 2013).

2. *System and software design*

   Pada tahap ini akan dipaparkan struktur-struktur dalam perancangan suatu perangkat lunak. Terdapat empat poin yang akan dipenuhi, yaitu:

   a. Perancangan arsitektural

   Pada perancangan arsitektural dilakukan identifikasi struktur dari keseluruhan sistem, identifikasi komponen utama (sub-sistem atau *modules*), kemudian juga melakukan identifikasi relasi-relasi antar tiap-tiap komponen pada sistem dan bagaimana relasi didistribusikan.

   b. Perancangan komponen

   Memaparkan pendefinisan detail dari komponen yang teradapat dalam suatu perangkat lunak. Pada proses ini harus didefinisikan algoritme suatu komponen dan struktur datanya. Perincian algoritme diproyeksikan kedalam bentuk *pseudocode*. *Pseudocode* merupakan bahasa pemrograman yang dekat dengan bahasa manusia dan tidak diperuntukkan dalam penggunaan bahasa mesin, sifat dari *pseudocode* juga fleksibel dan informal. Tujuan utama *pseudocode* adalah untuk merancang pemikirian sistem perangkat lunak sebelum dieksekuis menjadi kode program (Pressman, 2009).

   c. Perancangan antarmuka

   Memaparkan bagaimana cara berkomunikasi suatu perangkat lunak dengan sistem diluarnya ataupun dengan pengguna manusia. Pada penerapan perancangan antarmuka, akan digunakan prinsip-prinsip dasar yang dapat meningkatkan *usability* dari sebuah antarmuka khususnya *web*. Prinsip dasar dari perancangan ini adalah *Ten Good Deeds in Web Design* yang yang didasari oleh *Jakob's Law of Internet User Experience*. Inti dari teori ini adalah menyatakan bahwa pengguna lebih banyak mendapatkan pengalaman-pengalaman dari situs lain dari pada situs yang akan dibuat (Nielsen, 1999).

   d. Perancangan basis data

   Melakukan perancangan basis data dengan acuan model berupa *Physical Data Model* (PDM) yang nantinya dirujuk dalam proses implementasian (Foster and Godbole, 2016). *Physical Data Model* (PDM) merupakan pemaparan data yang lebih detail dan spesifik daripadai contextual data model maupun logical data model. Hal itu dikarenakan pada tiap entitas tabel

akan didefinisikan variabel beserta tipe data dan juga relasi-relasi antar entitas jika ada. Elemen-elemen didalamnya antara lain adalah tabel-tabel beserta kolom-kolomnya yang berisi nama entitas dan juga tipe datanya (A. S and Shalahuddin, 2013).

3. *Implementation and Unit Testing*

   Tahapan ini akan merealisasikan perancangan yang telah didefinisikan sebelumnya. Diantaranya perancangan basis data, perancangan arsitektural, perancangan antarmuka, dan perancangan komponen. Kemudian pada implementasinya akan dilakukan juga pengujian unit untuk memastikan bahwa komponen yang diimplementasikan sudah benar dan tidak terdapat error ataupun bug (Pressman, 2009). Pada tahap implementasi, didalamnya terdapat proses implementasi *database* menggunakan perintah *Structured Query Language* (SQL) jenis *Data Definition Language* (DDL). *Data Definition Language* adalah perintah yang digunakan untuk mendefinisikan struktur dari suatu *database*, yaitu *database* dan *table*. Terdapat empat perintah dasar pada *Data Definition Language* (Solichin, 2010) diantaranya:

   1) CREATE, perintah ini digunakan dalam tiap-tiap pembuatan suatu *database* ataupun *table.*
   2) ALTER, perintah ini digunakan untuk melakukan perubahan struktur dari *database* ataupun *table* yang sudah didefinisikan*.*
   3) RENAME, perintah ini digunakan untuk merubah nama dari *database* ataupun *table* yang sudah didefinisikan.
   4) DROP, perintah ini digunakan untuk menghapus *database* ataupun *table* yang sudah didefinisikan.

4. *Integration and System Testing*

   Tahapan ini akan melakukan pengujian integrasi antar komponen modul dan pengujian sistem. Tujuannya adalah untuk melihat apakah komponen-komponen utama dalam sistem sudah berjalan sesuai dengan kebutuhan yang telah didefinisikan sebelumnya.

5. *Operation and Maintenance*

   Pada tahapan ini, perangkat lunak sudah berada pada kondisi penggunaan yang sebenarnya. Tahap *operation* dan *maintenance* ini akan memerlukan durasi yang lebih panjang dalam pengerjaannya. Perangkat lunak yang telah di *deploy* akan dilakukan perawatan terus menerus untuk menjaga perangkat lunak terbebas dari bug maupun error yang tidak ada pada masa pengembangan. Kemudian tahap ini juga bertujuan untuk memenuhi kebutuhan-kebutuhan tambahan dari pengguna.

2.6 Pendekatan Berorientasi Objek

Perangkat lunak yang menggunakan konsep *Object Oriented* (OO) akan memiliki tahapan-tahapan yang bersifat *object oriented* pula diantaranya, *Object*

Oriented Analysis (OOA) yang berperan untuk melakukan identifikasi kebuthan yang berasal dari permasalahan-permasalahan. Setelah proses OOA dilakukan maka diteruskan ke tahap selanjutnya yaitu *Object Oriented Design* (OOD). OOD merupakan tahap perancangan yang pada prosesnya merujuk terhadap hasil dari OOA. Proses OOD sendiri memiliki empat poin yang harus dibuat diantaranya perancangan arsitektur, komponen, data, dan antarmuka. Kemudian dari hasil proses OOD akan dialkukan proses pembentukan kode program yang nantinya sebagai kode eksekusi terhadap mesin yang disebut *Object Oriented* Programming (OOP). Kemudian langkah akhir adalah *Object Oriented Testing* (OOT) yang akan melakukan pengujian terhadap hasil dari perancangan arsitektur, antarmuka, dan komponen (Pressman, 2009).

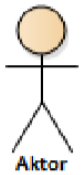
2.7 Pemodelan Berorientasi Objek

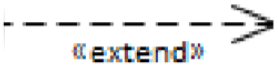
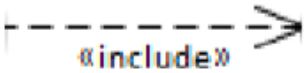
Unified Modelling Language (UML) merupakan pemodelan yang digunakan dalam OOA dan OOD (Pressman, 2009). Visualisasi, perancangan, dan dokumentasi dari model UML ini menjadikannya sebagai standar dalam pengembangan perangkat lunak (Booch, Rumbaugh and Jacobson, 1998). UML memiliki tiga diagram dalam proses merepresentasikannya, yaitu *use case diagram, sequence diagram, dan class diagram*. Penjelasan lebih lanjut terkait diagram yang telah disebutkan dapat dilihat pada sub bab 2.6.1 sampai dengan sub bab 2.6.3.

2.7.1 Use Case Diagram

Diagram ini akan menggambarkan interaksi antara sistem dan aktor. Dalam penggunaannya akan digambarkan bagaimana suatu aktor berinteraksi dengan fungsi yang terdapat pada sistem secara umum. Tujuannya adalah agar dapat lebih mudah dipahami dan dimengerti untuk proses yang lebih lanjut (Booch, Rumbaugh and Jacobson, 1998). *Use case diagram* memiliki notasi-notasi dalam penggunaannya diantaranya notasi actor, *use case*, dan relasi dimana terdapat pula empat jenis notasi pada relasi yaitu asosiasi (*association*), ekstensi (*extend*), *include,* dan generalisasi (*generalization*) (Whitten and Bentley, 2007). Notasi-notasi yang telah disebutkan akan dijelaskan lebih rinci pada Tabel 2.2.

Tabel 2.2 Notasi *Use Case Diagram*

Notasi	Penjelasan Notasi
Aktor 	Aktor merupakan *external users* yang melakukan *trigger* atau inisialisasi terhadap aktifitas sistem, yaitu *use case*. Tujuannya adalah untuk mengerjakan proses bisnis yang ada pada sistem dan menghasilkan suatu nilai yang terukur.
Use Case	*Use case* mendeskripsikan fungsi sistem dari perspektif *external users* sesuai dengan pemahamannya.

	
Asosiasi / *Association* 	Relasi asosiasi merupakan penghubung antara aktor dan *use case* yang menandakan bahwa terdapat suatu interaksi antara aktor dan *use case*.
Ekstensi / *Extend* 	Relasi *extend* menandakan bahwa *use case* yang melakukan *extend* memilik perpanjangan fungsi terhadap *use case* yang diekstensi dimana *use case* yang diekstensi tersebut hanya dapat dipanggil oleh *use case* yang melakukan *extend*.
Include 	Relasi *include* menandakan bahwa suatu *use case* tersedia untuk digunakan fungsionalitasnya oleh *use case* lain yang mereferensinya.
Generalisasi / *Generalization* 	Relasi generalisasi merupakan relasi antara aktor induk (umum) dengan aktor turunan (spesifik) lainnya dimana aktor turunan akan memiliki hak akses atas fungsionalitas-fungsionalitas sistem yang lebih luas.

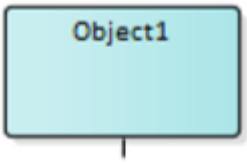
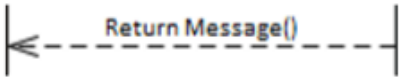
Sumber: (Ambler, 2005)

2.7.2 Sequence Diagram

Diagram ini akan menggambarkan bagaimana interaksi antar tiap-tiap objek yang ada didalam sistem dan diluar sistem dengan cara melihat alur pesan yang dikirimkan dari tiap-tiap objek dan keluaran dari objek tersebut (Booch, Rumbaugh and Jacobson, 1998). Interaksi-interaksi yang terjadi menggambarkan suatu operasi yang terjadi pada fitur sebuah perangkat lunak. Terdapat notasi-notasi dalam pemodelan *sequence diagram* diantaranya aktor, *lifeline, object, activation bar, input message, output message,* dan *frame* (Whitten and Bentley, 2007). Notasi-notasi yang telah disebutkan akan dijelaskan lebih rinci pada Tabel 2.3.

Tabel 2.3 Notasi *Sequence Diagram*

Notasi	Penjelasan Notasi

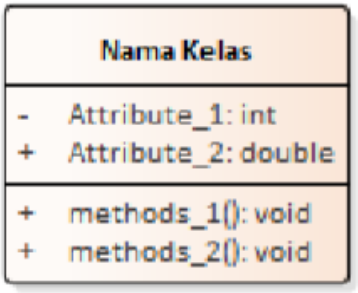
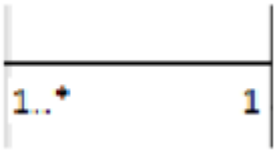
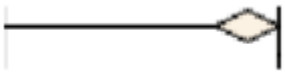
Aktor 	Merupakan aktor yang menginisialisasi suatu *use case* yang direpresentasikan kedalam bentuk simbol aktor.
Lifeline	Merupakan garis lurus dari aktor dan simbol sistem hingga kebawah yang terputus-putus, menandakan periode hidup dari *sequence*.
Object 	Merupakan representasi dari objek yang memiliki peran terhadap sistem, penggunaan *semicolon* (:) menandakan hasil instansiasi dari sistem.
Activation bar 	Merupakan sebuah balok vertikal yang berada pada *lifeline,* notais ini menandakan bahwa sedang terjadi suatu interaksi aktif antar model dalam sistem.
Input message 	Merupakan panah yang membentang secara horizontal dari pengirim pesan ke penerima pesan, bertujuan untuk mengirimkan pesan berdasarkan parameter yang dibawa.
Output message 	Merupakan panah putus-putus yang membentang secara horizontal dari penerima pesan ke pengirim pesan, bertujuan mengirimkan pesan balasan atau *return*.
Frame 	Merupakan kotak yang memberikan pesan tambahan untuk menunjukkan adanya suatu alternative tindakan atau pilihan.

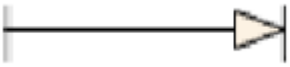
Sumber: (Whitten and Bentley, 2007)

2.7.3 Class Diagram

Diagram ini memaparkan bentuk struktur dari tiap-tiap *class* beserta relasi-relasi antar *class*. *Class* merupakan suatu *blueprint* atau cetakan dalam instansiasi objek yang mana merupakan inti dari pengembangan berorientasi objek (Booch, Rumbaugh and Jacobson, 1998). Terdapat dua bentuk notasi pada *class diagram,* pertama adalah *class* dimana detail *class* tersebut diantarnya *atribute*, *operation*, *data type atribute*, *data type operation*, *class type* apakah *class* biasa atau interface, kemudian *modifier* dari tiap-tiap *atribute* dan *class*. Sedangkan yang kedua adalah relasi, *class* dapat dinotasikan berdasarkan tipe relasinya, diantaranya asosiai, agregasi, komposisi, dan generalisasi (Whitten and Bentley, 2007). Notasi-notasi yang telah disebutkan akan dijelaskan lebih rinci pada Tabel 2.4.

Tabel 2.4 Notasi *Class Diagram*

Notasi	Penjelasan Notasi
Kelas / *Class* 	*Class* merupakan sebuah kotak wadah yang terdiri dari tiga bagian yaitu bagian pertama untuk nama *class*, bagian kedua untuk *attribute class* beserta *modifier* dan tipe datanya, dan bagian ketiga untuk *method class* beserta *modifier* dan tipe datanya.
Asosiasi / *Association* 	Relasi asosiasi menandakan bahwa *class-class* yang dihubungkan dengan relasi ini saling mengetahui satu sama lain. Pada relasi ini dapat didefinisikan suatu *multiplicity.*
Komposisi / *Composition* 	Relasi komposisi menandakan bahwa hubungan yang dimiliki sangat kuat sehingga ketiadaan *class part* akan sangat mempengaruhi *class whole* dimana *class whole* tidak akan bisa berdiri sendiri tanpa *class part*.
Agregasi / *Agregation* 	Relasi agregasi hanya menandakan bahwa suatu *class part* merupakan bagian dari *class whole* lainnya namun tidak memiliki hubungan yang kuat seperti relasi komposisi sehingga ketiadaan dari *class part* pada relasi agregasi tidak akan mempengaruhi *class whole*.

Generalisasi / *Generalization* 	Relasi generalisasi menandakan bahwa *class super* mewariskan *attribute-attribute* dan *method-method* yang dimilikinya kepada *class sub*.

Sumber: (Whitten and Bentley, 2007)

2.8 Teknologi Pengembangan Sistem

Dalam pengembangan aplikasi Homie diperlukan beberapa teknologi yang dapat menunjang dalam proses pengembangannya. Untuk penjelasan lebih lanjut akan dipaparkan pada sub bab 2.7.1 sampai dengan sub bab 2.7.5.

2.8.1 Midtrans

Midtrans adalah instansi yang menawarkan layanan *paymeny gateway*. Midtrans sendiri memiliki berbagai fitur yang dapat menunjang proses transaksi para penggunanya. Berbagai macam metode pembayaran disediakan oleh midtrans diantaranya *card payment*, *direct* debit, bank *transfer*, dan sebagainya. Optimalisasi transaksi midtrans berfokus pada ranah daring, dilain pihak pembeli atau *costumer* dari pebisnis yang menggunakan midtrans juga dapat terbantu dalam proses pembeliannya. Selain metode pembayaran dengan berbagai media yang beragam, Midtrans juga memberikan fasilitas yang dapat menunjang proses bisnis pengguna layanan *paymeny gateway* ini diantaranya analisa data , kelola risiko, dan fitur chat kepada para pelanggan dalam hal penambahan transaksi. Karena penggunaan midtrans yang sudah terbilang banyak, maka pengguna tidak perlu meragukan keamana dari midtrans. (Midtrans, 2015).

2.8.2 Laravel

Laravel merupakan frameword untuk mengembangkan aplikasi web yang menggunakan pola model-view-controller. Dalam pemakaiannya laravel meggunakan bahasa pemrograman web yaitu PHP. Tujuan utama laravel sendiri adalah untuk menunjang kualitas dari sebuah perangkat lunak yang berguna untuk mengurangi *cost* pengembangan dan perawatan perangkat lunak. Fitur-fitur yang dimiliki laravel juga sangat berperan dalam mempercepat waktu mengimplementasikannya. (McCool, 2012).

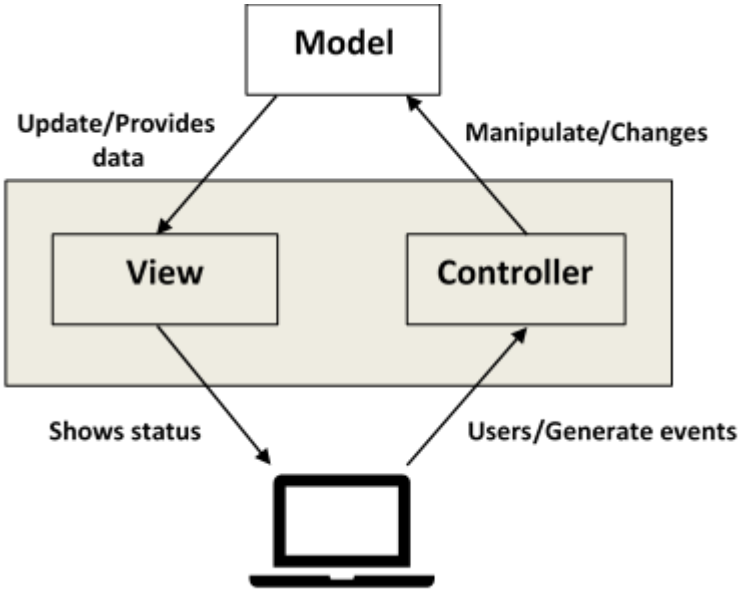
Karena pola yang digunakan pada laravel dalah MVC atau model-view-controller, maka dalam struktur laravel tersebut dilakukan pemisahan antara model, view, dan controller. Tujuan dari pemisahan model, view, dan controller adalah untuk menurunkan ikatan dari tiap komponen sehingga perubahan-perubahan yang terjadi tidak memiliki dampak yang terlalu besar. (Caytiles and Lee, 2014) dalam penelitiannya menyatakan bahwa MVC merupakan pola yang mana hasil dari penggunaannya akan mendapat sebuah aplikasi perangkat yang terbagi menjadi tiga objek diantarnya:

Model berperan dalam menangkap aksi dari domain permasalahan maupun antarmuka. Digunakannya juga untuk melakukan manipulasi data atau mengelola data beserta logika data dan aturan-aturan aplikasi

View memproyeksikan keluaran apapun dalam bentuk apapun ke antarmuka diantarnya teks, checkbox, grafik atau diagram, dan bentuk-bentuk lain yang serupa. Sehingga dalam satu view dapat terjadi penggabungan dari beberapa elemen.

Controller berperan sebagai penghubung antara model dan view. Dimana controller mengatur seluruh logika melalui inputan yang dilalukan user dan mengatur logika-logika dalam memanipulasi data.

Pemaparan model MVC dapat dilihat pada Gambar 2.3.



Gambar 2.3 Relasi Komponen MVC

Sumber: (Caytiles dan Lee, 2014)

2.8.3 ReactJS

ReactJS adalah pustaka JavaScript yang digunakan untuk membangun komponen UI yang dapat digunakan kembali. Menurut dokumentasi resmi React, React bertujuan untuk membangun antarmuka pengguna yang dapat dikompilasi. Ini mendorong terciptanya komponen UI yang dapat digunakan kembali, yang mana juga menyajikan data yang dapat berubah dari waktu ke waktu. React pada umumnya digunakan sebagai View pada model Model-View-Controller (MVC). React menawarkan model pemrograman yang lebih sederhana dan kinerja yang lebih baik. React juga bias melakukan render pada server menggunakan Node, dan itu juga berperan dalam menggerakkan aplikasi asli menggunakan React Native. React menerapkan satu arah aliran data reaktif, yang mengurangi *boilerplate* dan lebih mudah ditebak daripada pengikatan data tradisional (Wall et al., 2015).

1. Fitur React

JSX - JSX adalah ekstensi syntax JavaScript. Tidak ada keharusan dalam menggunakan JSX dalam pengembangan React, namun penggunaannya dianjurkan.
Komponen - React adalah kumpulan dari berbagai komponen sehingga semua tentang React adalah komponen. Dalam pengembangan menggunakan React, harus memikirikan terlebih dahulu bagaimana sistem tersebut dikelompokkan dalam komponen-komponen. Ini akan membantu mempertahankan kode ketika bekerja pada proyek dengan skala yang lebih besar.
Aliran data searah dan Flux - React mengimplementasikan aliran data satu arah yang mana membuatnya mudah memahami aplikasi. Flux adalah pola yang membantu menjaga data tetap searah.
Lisensi - React berlisensi di bawah Facebook Inc. Dokumentasi berlisensi di bawah CC BY 4.0.

2. Keuntungan React

React menggunakan virtual DOM yang merupakan objek JavaScript. Penggunaan virtual DOM akan meningkatkan kinerja aplikasi, karena JavaScript virtual DOM lebih cepat daripada DOM biasa. Kemudian react juga daapat digunakan pada sisi klien dan server serta dengan kerangka kerja lainnya. Komponen dan pola data juga membantu meningkatkan kemampuan membaca, yang mana juga dapat membantu menjaga keberlangsungan aplikasi yang lebih besar.

2.8.4 MySQL

MySQL merupakan aplikasi manajemen basis data SQL(*database management system*) atau DBMS. DBMS memiliki karakteristik berupa multithread, multi-user, dan telah memilik penginstalan aplikasi sebanyak enam juta lebih dunia. Dalam operasinya MySQL menggunakan Structured Query Language (SQL). MySQL memiliki banyak fitur yang mendukung dalam manajemen basis data. MySQL AB merupakan pemilik dan pemberi sponsor terhadap MySQL, dimana hak cipta sepenuhnya dipegang oleh MySQL AB (Solichin, 2010).

2.8.5 SortSite

SortSite memberikan layanan dalam pengujian perangkat lunak web yang bekerja dengan cara menguji tiap-tiap halaman dari sebuah web. Pengujian tersebut diantaranya adalah pengujian kualitas (PowerMapper, 2019). Dari banyak layanan yang tersedia, SortSite memiliki layanan untuk melakukan pengujian kompatibilitas dari sebuat web. Pengujian ini akan melakukan uji pada web yang akan diuji terhadap beberapa perambah yang umum digunakan. Jenis-jenis perambah yang dapat diuji oleh sortsite diantaranya edge, firefox, safari, google chrome, perangkat ios, dan perangkat android. Tiga klasifikasi masalah yang dipaparkan SortSite adalah *critical issues, major issues,* dan *minor issues*.

2.9 Pengujian Perangkat Lunak

Dalam melakukan pembentukan kumpulan test case maka dilakukanlah pengujian perangkat lunak yang dapat memberikan persentase yang tinggi akan ditemukannya error yang ada pada sistem (Pressman, 2009). Tujuan dari pengujian perangkat lain adalah menemukan error atau hal-hal yang tidak seharusnya ada pada sistem. Konten, *function, usability, navigability, performance,* kapasitas dan keamanan merupakan tempat error biasa ditemukan. Terdapat beberapa metode dalam melakukan pengujian perangkat lunak diantaranya *white-box testing*, *black-box testing*, dan *integration testing*. Kemudian untuk pengujian dari sisi kemudahan pengguna adalah *usability testing*.

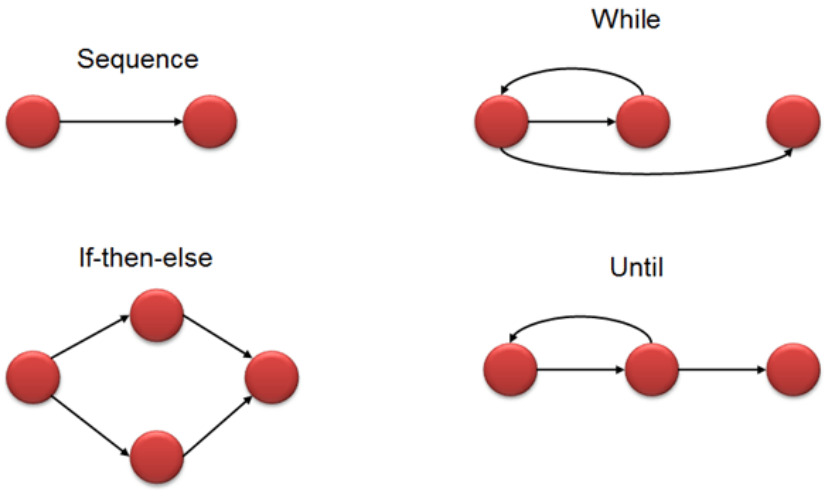
Dua teknik yang dapat digunakan dalam pengujian perangkat lunak adalah *white-box testing* dan *black-box testing* (Pressman, 2009):

2.9.1 White-box testing

White-box testing bertujuan untuk memastikan jalur-jalur logika telah didefinisikan dalam independent path dan minimal dieksekusi satu kali. Kemudian menjalankan seluruh kondisi logis dalam status benar atau salah. Selanjutnya menjalankan seluruh perulangan dalam batas-batasnya dan diluar batas-batasnya untuk memastikan bahwa syarat terpenuhi. Dalam penerapan *white-box testing* terdapat salah satu teknik yaitu *basis path testing*. Teknik ini berperan dalam pendefinisian test case melalui jalur yang ada (Pressman, 2009). *Basis path testing* akan menghasilkan nilai kompleksitas logika (Cyclomatic Complexity). Terdapat tiga tahapan dalam menerapkan metode *basis path testing* yaitu dengan membuat indpendent path, berdasarkan flowgraph yang telah digambarkan. Terdapat tiga komponen dalam pembuatan flowgraph, diantaranya:

1. *Node*
   Merupakan notasi satu atau lebih prosedur dengan bentuk lingkaran.
2. *Edge*
   Merupakan notasi alur *flowgraph* yang berbentuk anak panah.
3. *Predicate Node*
   Merupakan notasi node yang memiliki suatu kondisi tertentu yang digambarkan dengan memiliki dua *output edge*.

Pada Gambar 2.4 akan ditampilkan pendefinisian notasi dari flowgraph dimana pada penggambarannya akan membentuk dua atau lebih node yang pada ujung-ujungnya akan dihubungkan oleh edge, notasi-notasi tersebut diantaranya adalah model *flow sequence, while, if-then-else,* dan *until* (Guru99, 2019).



Gambar 2.4 Notasi *Flowgraph*

Sumber: (Guru99, 2019)

Sebuah metrik kompleksitas akan didapatkan dari hasil penggambaran flowgraph melalui perhitungan independent path, metrik kompleksitas tersebut adalah Cyclomatic Complexity V(G). Perhitungan untuk mendapatkan nilai Cyclomatic Complexity dapat dilakukan secara manual dengan menggunakan rumus seperti dibawah ini (Guru99, 2019):

V(G) = E – N + 2

Keterangan:
E = Jumlah seluruh *edge*
N = Jumlah seluruh *node*

V(G) = E – N + 2

Keterangan:
P = Jumlah seluruh *predicate node*

Kemudian jumlah *region* yang terdapat pada *flowgraph* juga dapat menentukan nilai V(G). Nilai kompleksitas dari V(G) memiliki beberapa tingkatan berdasarkan *range* angka yang diklasifikasikan menjadi empat level. Pendefinisian klasifikasi nilai cyclomatic complexity dapat dilihat pada Tabel 2.5.

Tabel 2.5 *Complexity Number*

Complexity Number	Meaning
1 – 10	Structured and well written code High Testability Cost and Effort is less
10 – 20	Complex Code Medium Testability Cost and effort is Medium

20 – 40	Very complex Code Low Testability Cost and Effort are high
> 40	Not at all testable Very high Cost and Effort

Sumber: (Guru99, 2019)

2.9.2 Black-box testing

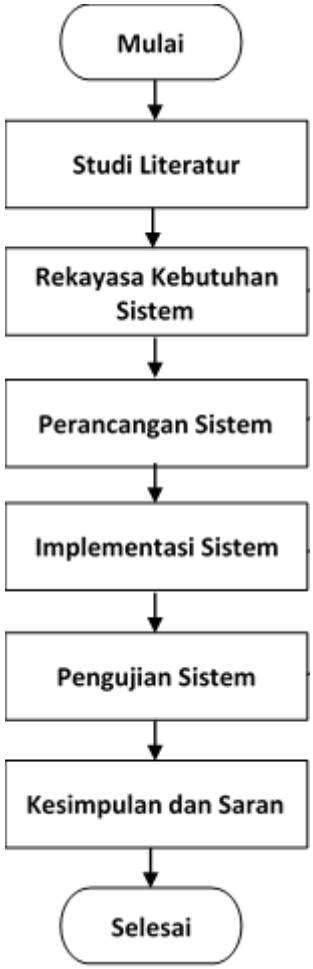
Black-box berutjuan untuk melakukan pengujian tanpa melihat kode program dan melakukan pengujian berdasarkan spesifikasi sistem. (Sommerville, 2011). Dikarenakan penggunaannya yang dikhususkan terhadap kebutuhan fungsional sistem, maka *black-box testing* dapat disebut juga dengan *behavioral testing* sistem. *Black-box testing* akan melihat apakah kebutuhan awal yang telah didefinisikan sesuai dengan implementasi sistem. Seperti pengujian yang lain, *black-box testing* bertujuan untuk mencar error pada sebauh sistem. Terdapat beberapa katergori dalam *Black-box*, diantarnya:

1. Fungsi-fungsi yang salah atau tidak ditemukan
2. Kesalahan antarmuka
3. Kesalahan pada struktur data atau pengaksesan database non-internal
4. Kesalahan proses
5. Kesalahan Inisialisasi dan terminasi.

Pengujian ini juga dapat digunakan untuk pengujian terhadap kebutuhan non-fungsional. Pada pengujian kebutuhan non-fungsional, aspek-aspek yang dilihat adalah yang tidak berkaitan dengan fungsi-fungsi tertentu ataupun aksi pengguna terhadap sistem. Pengujian kompatibilitas memiliki tujuan untuk menguji apakah perangkat lunak dapat dijalankan diberbagai lingkukangan dengan karakteristik yang berubah-ubah. Pemilihan terhadap konfigurasi lingkungan-lingkungan tertentu juga dilibatkan dalam pengujian kompatibilitas (Yoon et al., 2008). Kompatibilitas merupakan salah satu faktor utama yang mempengaruhi pengadopsian dari *e-commerce* (Alam et al., 2007).

BAB 3 METODOLOGI

Pada bab ini akan dijelaskan metode-metode yang dipergunakan pada pengembangan perangkat lunak Homie yaitu Aplikasi Reservasi dan Manajemen *Marketplace Homestay* Berbasis Web. Penelitian ini menerapkan pemodelan *waterfall* untuk proses pengembangan aplikasi Homie. Berdasarkan alirannya, maka tahap pertama dalam melakukan pengembangan aplikasi Homie adalah analisis kebutuhan, kemudian dilanjutkan hingga tahap pengujia. Pada Gambar 3.1 akan ditunjukkan diagram alir dari penelitian ini.



Gambar 3.1 Diagram Alir Penelitian

3.1 Studi Literatur

Teori-teori yang mendasari penelitian ini didasarkan oleh referensi yang didapatkan melalui artikel, buku, jurnal, konferensi, serta penelitian-penelitian yang terkait. Studi literatur berperan untuk pemahaman akan pengetahuan umum dalam melakukan perekayasaan kebutuhan, perancangan, implementasi, dan

pengujian sistem. Terdapat beberapa poin yang memberi lingkup terhadap studi literatur diantaranya:

1. Rekayasa Perangkat Lunak
2. *Payment Gateway*
3. Pengembangan Perangkat Lunak
4. Pendekatan Berorientasi Objek
5. Pemodelan Berorientasi Objek
6. Midtrans
7. Laravel
8. ReactJS
9. MySQL
10. SortSite
11. Pengujian Perangkat Lunak

3.2 Rekayasa Kebutuhan Sistem

Rekayasa kebutuhan pada sistem memiliki tujuan yaitu mengumpulkan seluruh kebutuhan-kebutuhan yang akan digunakan dalam pengembangan perangkat lunak. Wawancara digunakan sebagai praktik dalam melakukan elisitasi kebutuhan, wawancara akan dilakukan kepada pemilik *homestay* yang sekiranya cukup berkompeten dan mampu memberikan informasi terkait bisnis proses *homestay* tersebut. Bahasa pemodelan yang digunakan adalah Unified Modeling Language (UML) sedangkan untuk metode analisis menggunakan metode *Object Oriented Analysis* (OOA). Dalam pemodelannya akan digunakan *use case diagram* yang berguna untuk memaparkan interaksi antara user denagan fungsi-fungsi yang ada pada sistem. Proses rekayasa kebutuhan aplikasi Homie akan melakukan identifikasi terhadap seluruh kebutuhan user dalam mengembangkan perangkat lunak Aplikasi Reservasi dan Manajem *Marketplace Homestay* yang kemudian akan dimodelkan dalam bentuk *use case diagram*. Penjelasan lebih rinci terhadap fungsionalitas yang terdapat pada sistem akan dijelaskan lebih lanjut pada *use case scenario*. Selanjutnya hasil dari analisis tersebut akan dipaparkan kembali kepada pemangku kepentingan untuk dilakukan validasi dan verifikasi kebutuhan.

3.3 Perancangan Sistem

Perancangan sistem memiliki peran yang besar pada tahap selanjutnya yaitu implementasi dan pengujian. Tujuannya adalah sebagai acuan atau model pada tahap selanjutnya, yaitu implementasi dan pengujian. Terdapat beberapa poin perancangan yang dihasilkan dari tahap perancangan sistem diantaranya perancangan arsitektur, komponen, data, dan antarmuka. Kemudian untuk pembahasan lebih lanjut dan detail mengenai tahapan-tahapan perancangan sistem adalah sebagai berikut:

1. Perancangan Arsitektur
   Komponen penyusun dari perancangan aristektur adalah *sequence diagram* dan *class diagram*. *Sequence diagram* berperan dalam merepresentasikan interaksi yang terjadi pada objek, ditunjukkan dengan pengiriman dan pembelasan pesan dari objek dan oleh objek. Pada penelitian ini akan dilakukan pembuatan *sequence diagram* sebanyak tiga buah berdasarkan fungsionalitas utama yang ada pada perangkat lunak yang dikembangkan. Sedangkan *class diagram* berperan dalam memproyeksikan gambaran akan relasi-relasi antar *class* yang terdapat dalam sistem beserta komponen-komponen penyusun *class* diantaranya attribute dan method. Penggambaran *class diagram* tersebut akan mengikuti struktur *default* dari *framework* Laravel.
2. Perancangan Komponen
   Bagian pada penelitian ini akan menghasilakn tiga algoritme utama yang ada pada sistem yang akan dikembangkan. *Pseudocode* merupakan bahasa yang digunakan dalam penulisan algoritme-algoritme tersebut.
3. Perancangan Data
   Bagian pada penelitian ini akan menghasilkan suatu model yang menjadi acuan dalam implementasi basis data pada sistem yaitu *Physical Data Model* (PDM).
4. Perancangan Antarmuka
   Bagian pada penelitian ini akan menghasilkan berbagai rancangan antarmuka. Perancangan antarmuka tersebut akan menerapkan teori-teori dalam perancangan antarmuka sesuai dengan prinsip pada *Ten Good Deeds in Web Design*. Perancangan antarmuka yang dibuat akan menjadi acuan dalam tahap sleanjutnya yaitu implementasi antarmuka sistem.

3.4 Implementasi Sistem

Implementasi pada system merupakan tahap pengerjaan sistem yang mengikuti model-model yang telah dirancang pada tahap perancangan sistem. Terdapat beberapa poin implementasi sistem diantaranya adalah implementasi basis data dengan cara pendefinisian tabel menggunakan *Data Definition Language* (DDL) dan sedangkan untuk implementasinya akan memakai *Database Management System* (DMBS) MySQL, implementasi logika program dengan cara melampirkan hasil pemrograman menggunakan bahasa pemrograman yang digunakan, dan implementasi antarmuka dengan cara manampilkan hasil tangkapan layer. Implementasi sistem dilakukan dengan mengacu kepada spesifikasi sistem yang digunakan dalam pengembangan perangkat lunak ini. Kemudian untuk penjelasan lebih lanjut dalam tahapan-tahapan implementasi dapat dilihat pada bagian selanjutnya yaitu berikut:

1. Spesifikasi Sistem
   Spesifikasi sistem yang digunakan merupakan acuan dalam proses-proeses implementasi sistem selanjutnya pada pengembangan perangkat lunak ini.

2. Implementasi Logika Program
   Dalam melakukan implementasi kode program, model yang digunakan sebagai acuan impelemntasi adalah algoritme perancangan pada tahap perancangan sistem sebelumnya. Berdasarkan framework yang digunakan yaitu laravel, maka dalam implementasinya akan menggunakan bahasa pemrograman web yaitu PHP.
3. Implementasi Basis Data
   Dalam mengimplementasikan data base dilakukan proses mendefinisikan struktur tabel menggunakan Data Dictionary Language (DDL) yang akan dieksekusi untuk menciptakan struktur tabel pada database. Implementasi database akan dilakukan berdasarkan acuan dari *Physical Data Model* (PDM).
4. Implementasi Antarmuka
   Dalam melakukan implementasi antarmuka akan digunakan acuan berupa perancangan antarmuka pada tahap perancangan sistem sebelumnya, dimana perancangan sistem menerapkan teori-teori umum dalam perancangan antarmuka. Berdasarkan library yang digunakan yaitu ReactJS, maka bahasa yang digunakan adalah javascript dengan bantuan CSS pada sisi *styling*.

3.5 Pengujian Sistem

Pengujian pada sistem bertujuan untuk melihat apakah kebutuhan yang telah didefinisikan sebelumnya, yang mana kebutuhan tersebut sudah dilakukan implementasi terbebas dari error yang sekiranya masih tersisa didalam sistem perangkat lunak. Kemudian pengujian pada sistem juga bertujuan untuk melakukan minimalisir terjadinya error dan atau bug. Pada penelitian ini digunakan empat model pegujian yaitu pengujian unit dengan melakukan metode *basis path testing* (BPT) untuk mendapatkan *test case*, pengujian integrasi untuk megnetahui apakah komponen-komponen terkecil dalam sistem dapat terhubung secara baik dan benar, pengujian validasi untuk memvalidasi apakah hasil dari implementasi sudah dibuat berdasarkan pendefinisian kebutuhan pada tahap rekayasa kebutuhan sebelumnya, dan pengujian kompatibilitas untuk memastikan aplikasi dapat berjalan diberbagai perambah. Model-model pengujian yang telah didefinisikan akan dijelaskan lebih lanjut pada poin-poin dibawah ini diantaranya:

1. Pengujian Unit
   Pengujian unit bertujuan untuk melakukan *test* terhadapap komponen terkecil dalam sebuah sistem, komponen tersebut bisa berupa *class*, fungsi, ataupun objek. Karena pengujian ini akan melihat bagaimana cara kerja suatu komponen dalam *class* maka digunakanlah teknik pengujian *white-box testing* dan dalam mengumpulkan *test case-test case* yang akan diuji maka *basis path testing* akan menjadi metode dalam pengumpulan *test case*. Setelah *test case* didapatkan pengujian akan dilakukan dengan bantuan phpUnit dimana tiap *test case* akan didefinisikan terlebih dahulu dalam sebuah fungsi pada *class driver*. Selanjutnya, eksekusi pengujian unit menggunakan phpUnit dilakukan

dengan memasukkan perintah phpunit dan nama file pengujian sehingga phpUnit dapat melakukan *test* berdasarkan *test case* dan menampilkan hasil.

2. Pengujian Integrasi
   Pengujian integrase bertujuan untuk mengetahui apakah komunikasi antar modul berjalan dengan dan sudah terintegrasi dengan benar. Pada penelitian ini akan digunakan pendekatan top-down integration dalam melakukan pengujian integrasi. Pada pengerjaannya akan dibuat *class* stub yang bertujuan hanya untuk berkomunikasi pada *class* dengan level diatasnya. Sedangkan dari sisi teknik, pengujian ini akan menggunakan teknik *black-box testing*.

3. Pengujian Validasi
   Pengujian validasi bertujuan untuk menguji perangkat lunak, apakah perangkat lunak dan kebutuhan fungsional pada tahap rekayasa kebutuhan sudah selaras dalam implementasinya. Pengujian validasi dalam pengerjaannya akan melihat apakah kondisi masukan menghasilka kondisi keluaran yang diharapkan tanpa melihat bagian dalam sistem, sehingga pada pengujian ini akan digunakan teknik pengujian *black-box testing*.

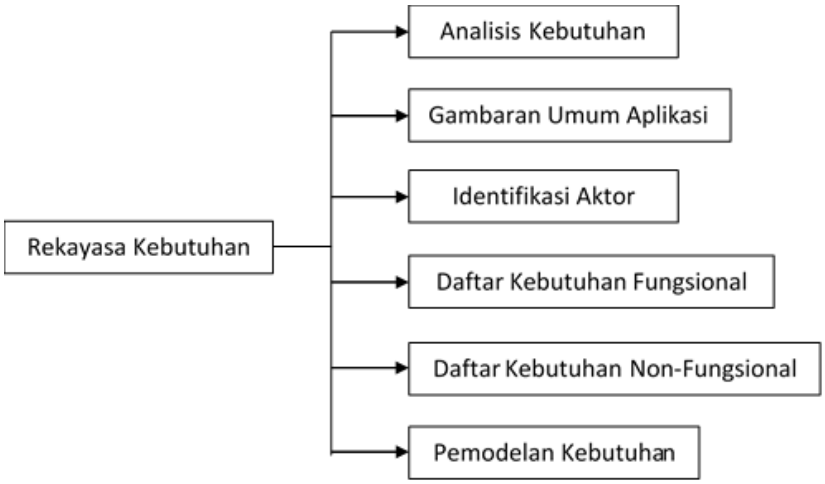
4. Pengujian Kompatibilitas
   Pengujian kompatibilitas bertujuan untuk mengeuji perangkat luank berdasarkan kebutuhan fungsional yaitu kompatibilitas perangkat lunak pada berbagai lingkungan atau perambah. Dalam penelitian ini, pengujian akan dilakukan dengan bantuan perangkat lunak SortSite yang memberikan layanan untuk melakukan pengujian kompatibilitas. SortSite akan menguji perangkat lunak dalam kemampuannya berjalan pada delapan jenis perambah.

3.6 Kesimpulan dan Saran

Pada bagian kesimpulan akan disimpulkan hasil dari penelitian Pengembangan Aplikasi Reservasi dan Manajemen *Marketplace Homestay* Berbasis Web dengan nama aplkasi Homie setelah melalui berbagai tahapan-tahapan penelitian diantaranya studi literatur, rekayasa kebutuhan, perancangan, implementasi, dan pengujian. Kemudian untuk mengetahui apakah masalah yang sudah dipaparkan sebelumnya dapat terselesaikan dengan adanya perangkat lunak ini, maka tahap pengujian harus dilakukan secara baik dan benar agar mendapatpak hasil yang valid pada kesimpulan ini. Selanjutnya adalah penulisan pada bagian saran, dimana informasi-informasi masukan yang berguna terkait pengembangan sistem selanjutnya akan dituliskan pada bagian ini.

BAB 4 REKAYASA KEBUTUHAN

Rekayasa kebutuhan pada penelitian ini merupakan proses yang pertama kali diterapkan. Tujuan dari rekayasa kebutuhan ini adalah untuk mencari tahu berbagai kebutuhan yang harus ada pada aplikasi Homie sehingga target yang akan dicapai dapat tercapai sesuai dengan hasil analisis kebutuhan. Proses rekayasa kebutuhan akan menghasilkan beberapa komponen penting diantaranya *use case diagram* dan *use case scenario.* Dua komponen tersebut bertujuan untuk mempermudah penulis dalam memahami berbagai kebutuhan perangkat lunak yang akan dikembangkan. Diagram *use case* sendiri berperan untuk mengidentifikasi para aktor yang terlibat dan interaksinya terhadap fungsionalitas-fungsionalitas sistem, sedangkan *use case scenario* berperan untuk melakukan identifikasi *scenario* pada perangkat lunak. Struktur dari bab rekayasa kebutuhan akan ditunjukkan pada Gambar 4.1.



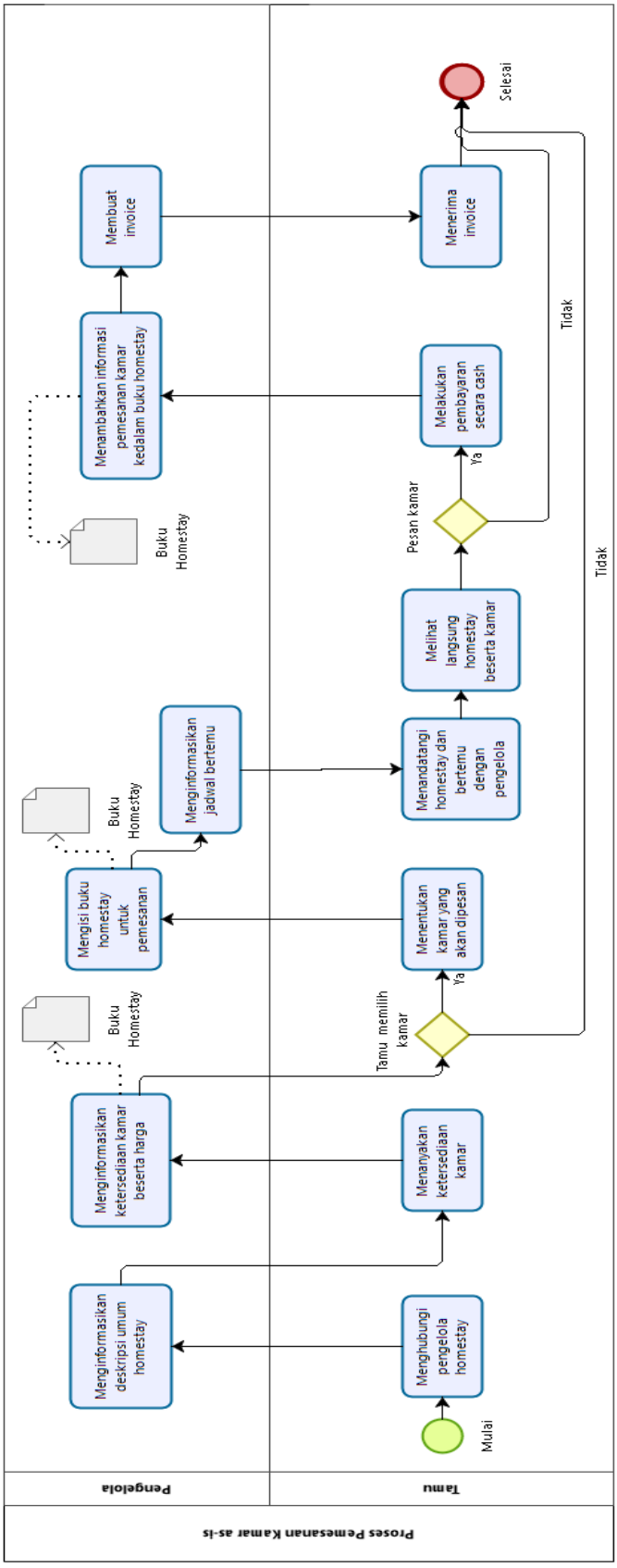
Gambar 4.1 Struktur Bab Rekayasa Kebutuhan

4.1 Analisis Kebutuhan

Sub bab ini bertujuan untuk mendefinisikan domain-domain masalah yang muncul dalam lingkup *homestay* pada Kota Pekanbaru. Metode yang dilakukan dalam analisis kebutuhan adalah wawancara. Wawacara lebih efektif dalam melakukan elisitasi data kualitatif dimana data kualitatif dapat lebih mudah ditransformasikan menjadi kebutuhan. Wawancara dilakukan kepada *homestay-homestay* yang ada di berbagai kecamatan yang ada di Kota Pekanbaru. Selanjutnya hasil dari analisis kebutuhan yang didapatkan akan dipaparkan kembali kepada pemangku kepentingan untuk dilakukan validasi dan verifikasi kebutuhan.

Berdasarkan hasil wawancara pada Lampiran B, dapat dilihat terjadi proses-proses yang memungkinkan terjadinya pembatalan tamu dalam melakukan pemesanan kamar *homestay*. Diantaranya ketika tamu harus menghubungi pengelola *homestay* hingga pengelola *homestay* menerima panggilan dari tamu

dan melanjutkan pertukaran informasi. Kemudian ketika tamu harus menyocokkan jadwal dengan jadwal milik pengelola *homestay* agar dapat bertemu untuk dapat melihat bentuk dari *homestay* dan kamar-kamar *homestay*. Selanjutnya tamu sudah harus memiliki dan membawa uang cash untuk melanjutkan ke proses transakasi. Lebih lanjut, jika dilihat dari sisi pengelola *homestay* maka dapat disimpulkan juga beberapa hal yang mengharuskan pengelola *homestay* harus siap siaga dalam menjalankan *homestay*-nya. Contohnya antara lain ketika tamu menghubungi, pengelola harus tahu jumlah kamar tersisa pada tanggal yang diinginkan oleh tamu, harga tiap-tiap kamar, fasilitas tiap-tiap kamar, dan mengetahui pesanan kamar yang belum melakukan transaksi atau sedang menunggu transaksi. Untuk pemaparan lebih lanjut tentang proses bisnis as-is dalam pemesanan kamar sebelum adanya aplikasi Homie ditunjukkan pada Gambar 4.2.

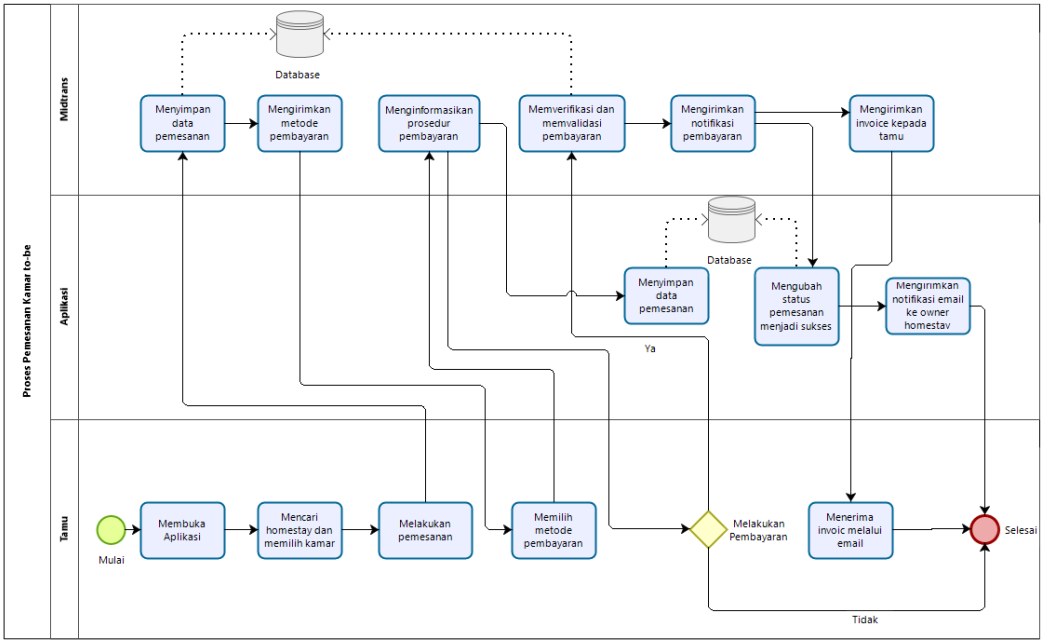


Gambar 4.2 Proses Bisnis As-Is Pemesanan Kamar

4.2 Gambaran Umum Sistem

Latar belakang dari pengembangan Aplikasi Reservasi dan Manajemen *Marketplace Homestay* Berbasis Web menghasilkan beberapa fitur-fitur penting yang dibutuhkan untuk menjadi solusi dalam permasalahan. Domain permasalahan berupa tentang bagaimana *homestay-homestay* yang ada dapat bersaing dengan bisnis-bisnis properti lainnya seperti hotel. Kemudian, bagaimana agar *homestay-homestay* mendapatkan kemudahan dalam mengorganisir dan menjaga kelancaran alur transaksi. Jika dijabarkan maka akan terdapat dua fitur utama yaitu manajemen *homestay dan* reservasi *homestay*. Fitur manajemen *homestay* bertujuan untuk memberikan kemampuan untuk melakukan manajemen terhadap *homestay* dan hanya bisa diakses dan dikelola oleh *owner homestay* itu sendiri, sedangkan reservasi *homestay* bertujuan untuk memberikan kemampuan terhadap *customer* dalam melakukan pemesanan kamar *homestay* secara *online*. Pada implementasiannya, sistem ini akan dikembangankan dan dibangun pada *platform* web agar penggunaan aplikasi tidak terbatas oleh device *costumer* yang beragam.

Pada Gambar 4.2 dipaparkan proses bisnis to-be pemesanan kamar setelah menggunakan aplikasi Homie. Dapat dilihat bahwa proses bisnis tersebut dapat memangkas alur-alur yang kurang efektif dan efisien. Kemudian tamu lebih leluasa dalam mencari *homestay*, memilih kamar, mengetahui informasi-informasi penting tentang *homestay* hingga melakukan pembayaran secara *cashless*.



Gambar 4.3 Proses Bisnis To-Be Pemesanan Kamar

4.3 Identifikasi Aktor

Para aktor yang nantinya akan bertinteraksi dengan sistem akan diidentifikasi dan dijelaskan perannya pada sub bab identifikasi aktor. Aktor-aktor tersebut diantaranya adalah *Admin*, *Owner*, dan *Guest*. Pada Tabel 4.1 akan dipaparkan hasil identifikasi para aktor beserta deskripsinya.

Tabel 4.1 Identifikasi Aktor

No.	Aktor	Deskripsi
1	*Guest*	*Guest* merupakan user yang belum *login* kedalam sistem.
2	*Owner*	*Owner* merupakan user yang sudah *login* kedalam sistem dan sudah terdaftar pada sistem.
3	*Admin*	*Admin* merupakan user yang sudah *login* dan memiliki akses penuh terhadap sistem yang dapat melakukan perubahan-perubahan.
4	Midtrans	Midtrans merupakan user yang memiliki hak akses melakukan perubahan status pemesanan.

4.4 Daftar Kebutuhan Fungsional

Kebutuhan-kebutuhan yang telah didapatkan akan diberi kode HOMIE-F-XX-YY. HOMIE menunjukkan nama dari Aplikasi yang akan dikembangkan. Pengkodean F merupakan kode yang menyatakan bahwa kebutuhan tersebut masuk kedalam kebutuhan fungsional, XX merupakan kode untuk penomoran kebutuhan fungsional, kemudian YY merupakan kode untuk penjabaran nomor spesifikasi kebutuhan. Berdasarkan hasil analisi, didapatkan dua puluh empat kebutuhan fungsional beserta spesifikasi kebutuhannya. Daftar kebutuhan fungsional beserta spesifikasinya akan dijabarkan dan dijelaskan pada Tabel 4.2.

Tabel 4.2 Daftar Kebutuhan Fungsional dan Spesifikasi Kebutuhan

No.	Kode	Kebutuhan	*Use Case*	Aktor
1	HOMIE-F-01	Sistem harus mampu menampilkan daftar *homestay* yang sudah terdaftar kedalam sistem kepada *guest* berdasarkan parameter pencarian.	Mencari *homestay*	*Guest*
	HOMIE-F-01-01	Form pencarian akan memiliki beberapa parameter berupa *field* diantaranya *location,*		

No.	Kode	Kebutuhan	*Use Case*	Aktor
		check-in date, dan *duration*.		
2	HOMIE-F-02	Sistem harus mampu menampilkan informasi yang lebih spesifik dan detail dari *homestay* yang sudah terdaftar kedalam sistem beserta daftar kamar yang tersedia kepada *guest.*	Melihat detail *homestay*	*Guest*
3	HOMIE-F-03	Sistem harus mampu menyediakan fungsi pemesanan kamar kepada *guest.*	Melakukan pemesanan kamar	*Guest*
	HOMIE-F-03-01	Form pemesanan akan memiliki beberapa parameter berupa *field* diantaranya name, *guest* name, *email* address, dan phone number.		
4	HOMIE-F-04	Sistem harus mampu menampilakan detail pemesanan kamar berdasarkan pilihan yang sudah dipilih sebelumnya kepada *guest.*	Melihat detail pemesanan	*Guest*
	HOMIE-F-04-01	Detail pemesanan akan menampilkan beberapa parameter berupa *textfield* diantaranya *homestay name* dan *address*, *checkin date, duration of stay and, checkout date, name*, *guest name*, *email* address, dan *phone number,* dan *price total*.		
5	HOMIE-F-05	Sistem harus mampu menyediakan fungsi pembayaran secara *cashless* kepada *guest.*	Melakukan pembayaran	*Guest*

No.	Kode	Kebutuhan	*Use Case*	Aktor
	HOMIE-F-05-01	Pembayaran secara *cashless* melalui *paymeny gateway* di fasilitasi oleh aplikasi pihak ketiga yaitu Midtrans yang akan memberikan metode pembayaran diantaranya credit card, ATM/Bank transfer, dan Go-Pay		
6	HOMIE-F-06	Sistem harus mampu menyediakan fungsi *login* kepada *guest* untuk masuk sebagai *owner* atau *admin.*	*Login*	*Guest*
	HOMIE-F-06-01	Form *login* akan memiliki beberapa parameter berupa *field* diantaranya *email* dan passowrd.		
7	HOMIE-F-07	Sistem harus mampu menyediakan fungsi registrasi kepada *guest* untuk mendaftar menjadi *owner.*	Registrasi	*Guest*
	HOMIE-F-07-01	Form *login* akan memiliki beberapa parameter berupa *field* diantaranya name, *email*, dan passowrd.		
8	HOMIE-F-08	Sistem harus mampu menambahkan *homestay* milik *owner* kedalam sistem.	Mendaftarkan *homestay.*	*Owner*
	HOMIE-F-08-01	Form mendaftarkan *homestay* akan memiliki beberapa parameter berupa *field* diantaranya name, location, address, facilities, number of rooms, description dan price.		

No.	Kode	Kebutuhan	*Use Case*	Aktor
9	HOMIE-F-09	Sistem harus mampu menampilkan *homestay* yang didaftarkan oleh *owner.*	Melihat *Homestay owner*	*Owner*
10	HOMIE-F-10	Sistem harus mampu mengubah informasi *homestay* milik *owner.*	Mengubah informasi *homestay*	*Owner*
	HOMIE-F-10-01	Form mengubah informasi *homestay* tidak akan dapat mengubah location, address dan number of rooms.		
11	HOMIE-F-11	Sistem harus mampu mengubah informasi yang ada pada kamar *homestay* sesuai dengan kamar yang dipilih oleh *owner*.	Mengubah informasi kamar	*Owner*
12	HOMIE-F-12	Sistem harus mampu mengubah ketersediaan kamar pada *owner homestay.*	Mengubah ketersediaan kamar	*Owner*
	HOMIE-F-12-01	Status-status ketersedian kamar diantaranya *Opened* dan *Closed*.		
13	HOMIE-F-13	Sistem harus mampu menampilkan ketersediaan kamar pada *owner homestay.*	Melihat ketersediaan kamar	*Owner*
14	HOMIE-F-14	Sistem harus mampu menyediakan fungsi *check-in* terhadap *guest* untuk *owner homestay.*	Melakukan *check-in*	*Owner*
15	HOMIE-F-15	Sistem harus mampu menyediakan fungsi *check-out* terhadap *guest* untuk *owner homestay.*	Melakukan *check-out*	*Owner*
16	HOMIE-F-16	Sistem harus mampu menampilkan daftar seluruh pemesanan kamar	Menampilkan daftar	*Owner*

No.	Kode	Kebutuhan	*Use Case*	Aktor
		yang berhasil kepada *owner homestay*.	pemesanan kamar	
	HOMIE-F-16-01	Parameter dalam daftar pesanan meliputi nama tamu, nomor kamar, tanggal *check-in*, durasi menginap, dan tanggal *check-out*.		
17	HOMIE-F-17	Sistem harus mampu menampilkan laporan pemasukan *homestay* dalam bentuk infografis kepada *owner homestay.*	Menampilkan infografis pemasukan	*Owner*
18	HOMIE-F-18	Sistem harus mampu menampilkan seluruh daftar *homestay* yang telah terdaftar pada sistem melalui *admin*.	Menampilkan daftar *homestay*	*Admin*
19	HOMIE-F-19	Sistem harus mampu menampilkan daftar seluruh akun *owner* yang telah terdaftar pada sistem melalui *admin*.	Menampilkan daftar *owner*	*Admin*
20	HOMIE-F-20	Sistem harus mampu menghapus *homestay* milik *owner* melalui *admin*.	Menghapus *homestay*	*Admin*
	HOMIE-F-20-01	Hapus *homestay owner* juga akan melakukan penghapusan terhadap kamar-kamar *homestay* pada *database*.		
21	HOMIE-F-21	Sistem harus mampu menghapus akun *owner* melalui *admin.*	Menghapus akun *owner*	*Admin*
	HOMIE-F-21-01	Hapus akun *owner* juga akan melakukan penghapusan terhadap *homestay owner* pada *database*.		

No.	Kode	Kebutuhan	*Use Case*	Aktor
22	HOMIE-F-22	Sistem harus mampu menampilkan seluruh daftar transaksi kepada *admin.*	Menampilkan seluruh transaksi	*Admin*
23	HOMIE-F-23	Sistem harus mampu menyediakan fungsi *logout.*	*Logout*	*Owner dan Admin*
24	HOMIE-F-24	Sistem harus mampu merubah status pemesanan.	Mengubah status pemesanan	Midtrans
	HOMIE-F-24-01	Status-status perubahan yang diubah diantaranya *capture*(sukses), *pending*, dan *expire*		

4.5 Daftar Kebutuhan Non-Fungsional

Pada tiap kebutuhan non-fungsional akan diberikan pengkodean yang serupa dengan pengkodean pada kebutuhan fungsional yaitu HOMIE-N-XX. HOMIE menunjukkan nama dari Aplikasi yang akan dikembangkan, N merupakan kode yang menyatakan bahwa kebutuhan tersebut masuk kedalam kebutuhan non-fungsional, XX merupakan kode untuk penomoran kebutuhan non-fungsional. Pada pengembangan perangkat lunak Homie membutuhkan satu kebutuhan non-fungsional. Kebutuhan non-fungsional tersebut adalah kompatibilitas sistem, dimana kebutuhan ini merupakan elemen penting dalam pengembangan aplikasi dengan model pasar elektronik. Daftar kebutuhan non-fungsional akan dijabarkan dan dijelaskan pada Tabel 4.3.

Tabel 4.3 Daftar Kebutuhan Non-Fungsional

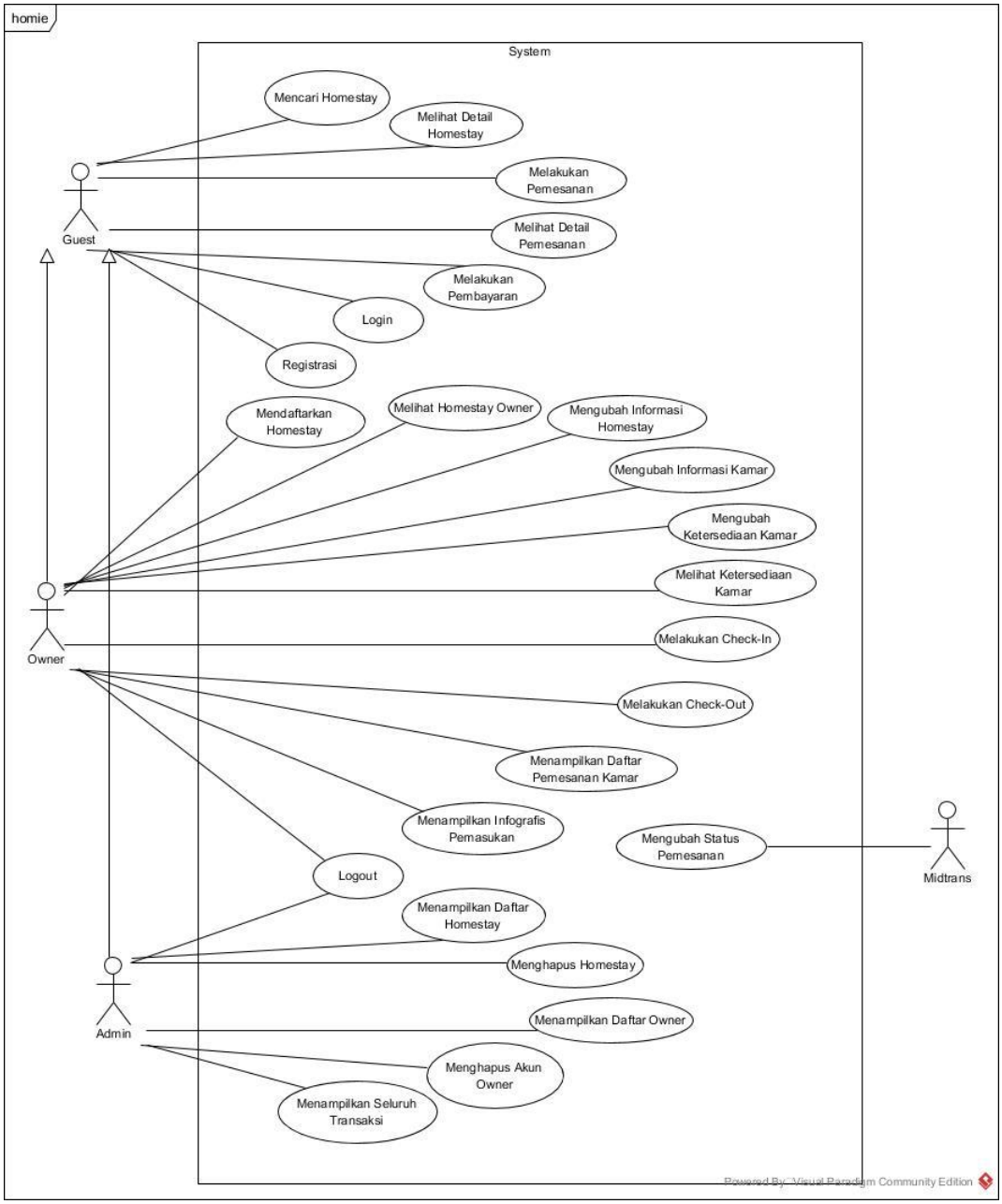
No.	Kode	Kebutuhan
1	HOMIE-N-01	Sistem dapat dijalankan pada berbagai jenis perambah seperti edge, firefox, safari, google chrome, perangkat ios, dan perangkat android.

4.6 Pemodelan Kebutuhan

Berdasarkan daftar kebutuhan yang telah didefinisikan, maka akan dipaparkan kedalam bentuk pemodelan kebutuhan. Terdapat dua sub bab dalam merepresentasikan kebutuhan dalam model kebutuhan diantaranya menggunakan *use case digram* dan *use case scenario*.

4.6.1 *Use Case Diagram*

Bagian ini akan menghasilkan *diagram* yang merepresentasikan komunikasi atau intetraksi antara sistem dan aktor eksternal sistem. Pada pembuatan *use case diagram* akan mengacu kepada kebutuhan fungsional yang telah didefinisikan pada bagian sebelumnya. Tujuan dari pembuatan *use case diagram* yang sesuai dengan kebutuhan fungsional adalah untuk memudahkan dalam melihat, membaca, dan memahami fungsionalitas dari sistem yang akan dibangun. *Use case diagram* sistem yang akan dikembangkan dapat dilihat pada Gambar 4.1.



Gambar 4.4 *Use Case Diagram* Aplikasi Manajemen dan Reservasi *Marketplace Homestay*

4.6.2 *Use Case Scenario*

Use case scenario bertujuan untuk menjabarkan dan menjelaskan informasi detail pada tiap-tiap kebutuhan fungsional dan langkah-langkah yang terlibat. *Use case scenario* dari tiap fungsionalitas akan dipaparkan pada Tabel 4.4 sampai dengan Tabel 4.27.

Pada Tabel 4.4 akan dijelaskan bagaimana aktor berinteraksi dengan *use case* mencari *homestay* berdasarkan skenario-skenario yang ada pada *use case* mencari *homestay*.

Tabel 4.4 *Use Case Scenario* Mencari *Homestay*

Flow of Events for mencari *homestay*	
Kode Kebutuhan	HOMIE-F-01
Objective	*Use case* ini bertujuan untuk menampilkan *list homestay* yan sudah terdaftar pada sistem sesuai dengan parameter pencarian.
Actor	*Guest*
Pre-condition	Aktor telah mengakses halaman utama sistem dan sistem telah menampilkan halaman utama yang berisi *search bar*.
Main Flow	1. Aktor menentukan nilai parameter pencarian *homestay* yaitu, lokasi, tanggal *check-in*, dan durasi kemudian menekan tombol *search.* 2. Sistem menampilkan daftar *homestay* sesuai dengan parameter pencarian yang telah ditentukan oleh aktor.
Alternative Flow	1. Apabila *homestay* yang dicari tidak sesuai denga *homestay* yang terdaftar pada sistem, maka akan ditampilkan pesan “Homestay Not Found” oleh sistem.
Post-condition	Aktor melihat daftar *homestay* yang terdaftar pada sistem sesuai dengan parameter pencarian yang telah ditentukan.

Pada Tabel 4.5 akan dijelaskan bagaimana aktor berinteraksi dengan *use case* melihat detail *homestay* berdasarkan skenario-skenario yang ada pada *use case* melihat detail *homestay*.

Tabel 4.5 *Use Case Scenario* Melihat Detail *Homestay*

Flow of Events for melihat detail *homestay*	
Kode Kebutuhan	HOMIE-F-02

Objective	*Use case* ini bertujuan untuk menampilkan penjelasan yang lebih detail terhadap *homestay,* seperti ketersediaan kamar, harga per malam, alamat, dan fasilitas beserta daftar kamar yang tersedia.
Actor	*Guest*
Pre-condition	Aktor telah melihat daftar *homestay* yang terdaftar pada sistem sesuai dengan parameter pencarian yang telah ditentukan.
Main Flow	1. Aktor memilih salah satu *homestay* dari daftar yang telah ditampilkan sistem. 2. Sistem menampilkan rincian harga per malam, ketersediaan kamar, alamat, dan fasilitas beserta daftar kamar.
Alternative Flow	-
Post-condition	Aktor melihat informasi detail dari *homestay* yang telah dipilih beserta daftar kamar *homestay* tersebut.

Pada Tabel 4.6 akan dijelaskan bagaimana aktor berinteraksi dengan *use case* melakukan pemesanan kamar berdasarkan skenario-skenario yang ada pada *use case* melakukan pemesanan kamar.

Tabel 4.6 *Use Case Scenario* Melakukan Pemesanan Kamar

Flow of Events for melakukan pemesanan kamar	
Kode Kebutuhan	HOMIE-F-03
Objective	*Use case* ini bertujuan untuk melakukan pemesanan kamar melalui sistem.
Actor	*Guest*
Pre-condition	Aktor telah memilih salah satu *homestay* dari daftar yang telah ditampilkan sistem dan sistem menampilkan rincian harga per malam, ketersediaan kamar, alamat, dan fasilitas.
Main Flow	1. Aktor memilih opsi pilih kamar pada daftar kamar yang tersedia. 2. Sistem menampilkan form pemesanan kamar. 3. Aktor mengisi form pemesanan kamar dan memilih opsi pesan. 4. Sistem menyimpan informasi pemesanan kamar oleh aktor.

Alternative Flow	1. Jika dalam mengisi form terdapat *field* kosong, maka sistem akan menampilkan pesan bahwa seluruh *field* harus diisi.
Post-condition	Aktor berhasil menyimpan informasi pemesanan kamar kedalam sistem.

Pada Tabel 4.7 akan dijelaskan bagaimana aktor berinteraksi dengan *use case* melihat detail pemesanan berdasarkan skenario-skenario yang ada pada *use case* melihat detail pemesanan.

Tabel 4.7 *Use Case Scenario* Melihat Detail Pemesanan

Flow of Events for melihat detail pemesanan	
Kode Kebutuhan	HOMIE-F-04
Objective	*Use case* ini bertujuan untuk melihat detail pemesanan berdasarkan pilihan yang sudah dipilih oleh aktor
Actor	*Guest*
Pre-condition	Aktor telah melakukan pemesanan dengan memilih *homestay*, kamar, dan mengisi form pemesanan kamar yang ditampilkan oleh sistem.
Main Flow	1. Aktor memilih opsi *confirm* 2. Sistem menampilkan rincian pesanan yaitu, total harga yang hatus dibayar, informasi kamar, alamat, dan fasilitas sesuai dengan pilihan aktor ketika memesan kamar.
Alternative Flow	-
Post-condition	Aktor berhasil melihat detail pemesanan kamar.

Pada Tabel 4.8 akan dijelaskan bagaimana aktor berinteraksi dengan *use case* melakukan pembayaran berdasarkan skenario-skenario yang ada pada *use case* melakukan pembayaran.

Tabel 4.8 *Use Case Scenario* Melakukan Pembayaran

Flow of Events for melakukan pembayaran	
Kode Kebutuhan	HOMIE-F-05
Objective	*Use case* ini bertujuan untuk melakukan pembayaran pesanan kamar *homestay* secara *cashless* melalui *window payment gateway*.
Actor	*Guest*

Pre-condition	Aktor telah melihat detail pemesanan kamar.
Main Flow	1. Aktor memilih opsi bayar. 2. Sistem menampilkan *pop up window payment gateway*. 3. Aktor memilih metode pembayaran pada *pop up window payment gateway* Midtrans dan melakukan pembayaran. 4. *Payment gateway* Midtrans mengirimkan notifikasi pembayaran kepada email pemesan.
Alternative Flow	-
Post-condition	Aktor melakukan pembayaran kamar yang telah dipesan.

Pada Tabel 4.9 akan dijelaskan bagaimana aktor berinteraksi dengan *use case login* berdasarkan skenario-skenario yang ada pada *use case login*.

Tabel 4.9 *Use Case Scenario Login*

Flow of Events for login	
Kode Kebutuhan	HOMIE-F-06
Objective	*Use case* ini bertujuan untuk memberikan akses kepada *guest* untuk dapat melakukan aksi sesuai otoritasnya.
Actor	*Guest*
Pre-condition	Aktor sudah mengakses halaman web sistem.
Main Flow	1. Aktor memilih opsi *login.* 2. Sistem menampilkan form *login.* 3. Aktor mengisi *field-field* pada form *login*, yaitu *username* dan *password* kemudian memilih opsi *login.* 4. Sistem memberikan izin kepada *guest* untuk masuk kedalam sistem sebagai *owner* atau *admin.*
Alternative Flow	1. Jika aktor tidak mengisikan *username* ataupun *password* ketika *login*, maka sistem akan menampilkan pesan bahwa *field* tidak boleh kosong. 2. Jika aktor memasukkan *username* atau *password* yang salah, maka sistem akan menampilkan pesan bahwa *username* atau *password* salah.

Post-condition	Aktor masuk kedalam sistem dan mengakses layanan-layanan sistem sesuai *role*-nya.

Pada Tabel 4.4 akan dijelaskan bagaimana aktor berinteraksi dengan *use case* registrasi berdasarkan skenario-skenario yang ada pada *use case* registrasi.

Tabel 4.10 *Use Case Scenario* Registrasi

Flow of Events for registrasi	
Kode Kebutuhan	HOMIE-F-07
Objective	*Use case* ini bertujuan untuk membuat akun *owner.*
Actor	*Guest*
Pre-condition	Aktor sudah mengakses halaman web sistem.
Main Flow	1. Aktor memilih opsi registrasi*.* 2. Sistem menampilkan form registrasi*.* 3. Aktor mengisi *field-field* pada form registrasi, yaitu *name, email address*, dan *password*, kemudian memilih opsi registrasi*.* 4. Sistem akan mendaftarkan akun baru ke *database.* 5. Sistem memberikan informasi bahwa akun berhasil didaftarkan kedalam sistem.
Alternative Flow	1. Jika aktor tidak mengisikan *username, password,* nama, alamat, nomor telfon, ataupun *e-mail* ketika registrasi, maka sistem akan menampilkan pesan bahwa *field* tidak boleh kosong.
Post-condition	Aktor berhasil mendaftarkan akunnya kedalam sistem.

Pada Tabel 4.11 akan dijelaskan bagaimana aktor berinteraksi dengan *use case* mendaftarkan *homestay* berdasarkan skenario-skenario yang ada pada *use case* mendaftarkan *homestay*.

Tabel 4.11 *Use Case Scenario* Mendaftarkan *Homestay*

Flow of Events for mendaftarkan *homestay*	
Kode Kebutuhan	HOMIE-F-08
Objective	*Use case* ini bertujuan untuk mendaftarkan *homestay* milik *owner* kedalam sistem.
Actor	*Owner*
Pre-condition	Aktor sudah *login.*

Main Flow	1. Aktor memilih opsi daftarkan *homestay* pada saat menekan logo profile. 2. Sistem menampilkan form pendaftaran *homestay.* 3. Aktor mengisi *field-field* pada form pemdaftaran *homestay*, yaitu nama *homestay*, alamat *homestay*, jumlah kamar beserta tipe-tipe kamar, harga per malam untuk tiap-tiap kamar, dan fasilitas kemudian memilih opsi daftarkan. 4. Sistem akan mendaftarkan *homestay* baru ke *database.*
Alternative Flow	1. Jika aktor tidak mengisi seluruh *field,* maka sistem akan menampilkan pesan bahwa seluruh *field* harus diisi.
Post-condition	Aktor berhasil mendaftarkan *homestay* kedalam sistem.

Pada Tabel 4.12 akan dijelaskan bagaimana aktor berinteraksi dengan *use case* melihat *homestay owner* berdasarkan skenario-skenario yang ada pada *use case* melihat *homestay owner*.

Tabel 4.12 *Use Case Scenario* Melihat *Homestay Owner*

Flow of Events for melihat *homestay owner*	
Kode Kebutuhan	HOMIE-F-09
Objective	*Use case* ini bertujuan *homestay* milik *owner* yang telah didaftarkan kedalam sistem.
Actor	*Owner*
Pre-condition	Aktor sudah *login.*
Main Flow	1. Aktor memilih opsi my *homestay* pada saat menekan logo profile. 2. Sistem menampilkan *homestay* yang dimiliki oleh *owner.*
Alternative Flow	1. Jika aktor belum mendaftarkan *homestay*, maka sistem akan secara otomatis berpindah ke halaman mendaftarkan *homestay*.
Post-condition	Aktor dapat melihat *homestay* miliknya yang telah didaftarkan kedalam sistem.

Pada Tabel 4.13 akan dijelaskan bagaimana aktor berinteraksi dengan *use case* mengubah informasi *homestay* berdasarkan skenario-skenario yang ada pada *use case* mengubah informasi *homestay*.

Tabel 4.13 *Use Case Scenario* Mengubah Informasi *Homestay*

Flow of Events for mengubah informasi *homestay*	
Kode Kebutuhan	HOMIE-F-10
Objective	*Use case* ini bertujuan untuk mengubah informasi *homestay* yang telah didaftarkan.
Actor	*Owner*
Pre-condition	Aktor sudah *login*.
Main Flow	1. Aktor memilih opsi *edit homestay* pada saat menekan logo profile. 2. Sistem menampilkan form *edit homestay.* 3. Aktor mengisi ulang *field-field* yang akan diubah kemudian memilih opsi simpan. 4. Sistem menyimpan informasi *homestay* sesuai dengan informasi yang telah diisikan.
Alternative Flow	-
Post-condition	Aktor berhasil mengubah informasi *homestay.*

Pada Tabel 4.14 akan dijelaskan bagaimana aktor berinteraksi dengan *use case* mengubah informasi kamar berdasarkan skenario-skenario yang ada pada *use case* mengubah informasi kamar.

Tabel 4.14 *Use Case Scenario* Mengubah Informasi Kamar

Flow of Events for mengubah informasi kamar	
Kode Kebutuhan	HOMIE-F-11
Objective	*Use case* ini bertujuan untuk mengubah informasi kamar *homestay* yang telah didaftarkan.
Actor	*Owner*
Pre-condition	Aktor sudah *login*.
Main Flow	1. Aktor memilih opsi *myhomestay* pada saat menekan logo profile. 2. Sistem menampilkan *homestay* milik *owner*. 3. Aktor memilih opsi edit kamar

	4. Sistem menampilkan daftar seluruh kamar yang ada pada *homestay* 5. Aktor memilih opsi edit kamar pada daftar kamar yang telah ditampilkan oleh sistem 6. Sistem menampilkan form *edit homestay.* 7. Aktor mengisi ulang *field-field* yang akan diubah kemudian memilih opsi simpan. 8. Sistem menyimpan informasi kamar sesuai dengan informasi yang telah diisikan
Alternative Flow	-
Post-condition	Aktor berhasil mengubah informasi kamar *homestay.*

Pada Tabel 4.15 akan dijelaskan bagaimana aktor berinteraksi dengan *use case* mengubah ketersediaan kamar berdasarkan skenario-skenario yang ada pada *use case* mengubah ketersediaan kamar.

Tabel 4.15 *Use Case Scenario* Mengubah Ketersediaan Kamar

Flow of Events for mengubah ketersediaan kamar	
Kode Kebutuhan	HOMIE-F-12
Objective	*Use case* ini bertujuan untuk mengubah ketersediaan kamar *homestay* yang telah didaftarkan.
Actor	*Owner*
Pre-condition	Aktor sudah *login*.
Main Flow	1. Aktor memilih opsi *myhomestay* pada saat menekan logo profile. 2. Sistem menampilkan *homestay* milik *owner*. 3. Aktor memilih opsi edit kamar. 4. Sistem menampilkan daftar seluruh kamar yang ada pada *homestay.* 5. Aktor memilih opsi buka atau tutup kamar pada daftar kamar yang telah ditampilkan oleh sistem. 6. Sistem menyimpan informasi ketersediaan kamar kedalam *database*.
Alternative Flow	-
Post-condition	Aktor berhasil mengubah informasi ketersediaan kamar *homestay.*

Pada Tabel 4.16 akan dijelaskan bagaimana aktor berinteraksi dengan *use case* melihat ketersediaan kamar berdasarkan skenario-skenario yang ada pada *use case* melihat ketersediaan kamar.

Tabel 4.16 *Use Case Scenario* Melihat Ketersediaan Kamar

Flow of Events for melihat ketersediaan kamar	
Kode Kebutuhan	HOMIE-F-13
Objective	*Use case* ini bertujuan untuk melihat ketersediaan kamar *homestay.*
Actor	*Owner*
Pre-condition	Aktor sudah *login.*
Main Flow	1. Aktor memilih opsi *checkin/checkout* pada saat menekan logo profile. 2. Sistem menampilkan halaman ketersediaan kamar.
Alternative Flow	-
Post-condition	Aktor melihat ketersediaan kamar.

Pada Tabel 4.17 akan dijelaskan bagaimana aktor berinteraksi dengan *use case* melakukan *check-in* berdasarkan skenario-skenario yang ada pada *use case* melakukan *check-in*.

Tabel 4.17 *Use Case Scenario* Melakukan *Check-In*

Flow of Events for melakukan *check-in*	
Kode Kebutuhan	HOMIE-F-14
Objective	*Use case* ini bertujuan untuk melakukan *check-in guest* terhadap kamar.
Actor	*Owner*
Pre-condition	Aktor sudah berada pada halaman melihat ketersediaan kamar.
Main Flow	1. Aktor memilih opsi *check-in* pada kamar yang memiliki informasi yang sesuai dengan *guest.* 2. Sistem menandakan kamar yang sudah dipilih sebagai terisi.
Alternative Flow	1. Apabila *check-in* dilakukan ketika transaction status tidak berstatus *capture*, maka sistem akan menampilkan pesan *transaction not found*.

Post-condition	Aktor berhasil melakukan *check-in guest* terhadap kamar dan mengubah status pesanan pada sistem.

Pada Tabel 4.18 akan dijelaskan bagaimana aktor berinteraksi dengan *use case* melakukan *check-out* berdasarkan skenario-skenario yang ada pada *use case* melakukan *check-out*.

Tabel 4.18 *Use Case Scenario* Melakukan *Check-Out*

Flow of Events for melakukan *check-out*	
Kode Kebutuhan	HOMIE-F-15
Objective	*Use case* ini bertujuan untuk melakukan *check-out guest* terhadap kamar.
Actor	*Owner*
Pre-condition	Aktor sudah berada pada halaman melihat ketersediaan kamar.
Main Flow	1. Aktor memilih opsi *check-out* pada kamar yang memiliki informasi yang sesuai dengan *guest.* 2. Sistem menandakan kamar yang sudah dipilih sebagai kosong.
Alternative Flow	1. Apabila *check-out* dilakukan ketika transaction status tidak berstatus *active*, maka sistem akan menampilkan pesan *transaction is not active*.
Post-condition	Aktor berhasil melakukan *check-out guest* terhadap kamar dan mengubah status pesanan pada sistem.

Pada Tabel 4.19 akan dijelaskan bagaimana aktor berinteraksi dengan *use case* menampilkan daftar pemesanan kamar berdasarkan skenario-skenario yang ada pada *use case* menampilkan daftar pemesanan kamar.

Tabel 4.19 *Use Case Scenario* Menampilkan Daftar Pemesanan Kamar

Flow of Events for menampilkan daftar pemesanan kamar	
Kode Kebutuhan	HOMIE-F-16
Objective	*Use case* ini bertujuan untuk menampilkan daftar pemesanan kamar.
Actor	*Owner.*
Pre-condition	Aktor sudah *login.*
Main Flow	1. Aktor memilih opsi tampilkan daftar pemesanan kamar pada saat menekan logo profile.

	2. Sistem menampilkan daftar pemesan kamar.
Alternative Flow	-
Post-condition	Aktor melihat daftar pemesanan kamar.

Pada Tabel 4.20 akan dijelaskan bagaimana aktor berinteraksi dengan *use case* menampilkan infografis pemasukan berdasarkan skenario-skenario yang ada pada *use case* menampilkan infografis pemasukan.

Tabel 4.20 *Use Case Scenario* Menampilkan Infografis Pemasukan

Flow of Events for menampilkan infografis pemasukan	
Kode Kebutuhan	HOMIE-F-17
Objective	*Use case* ini bertujuan untuk menampilkan infografis pendapatan dalam bentuk *chart*.
Actor	*Owner*
Pre-condition	Aktor sudah *login.*
Main Flow	1. Aktor memilih opsi report pada saat menekan logo profile. 2. Sistem menampilkan pemasukan dalam bentuk infografis berdasarkan pemesanan yang berhasil dibayar.
Alternative Flow	-
Post-condition	Aktor melihat laporan pemasukan dalam bentuk infografis *chart*.

Pada Tabel 4.21 akan dijelaskan bagaimana aktor berinteraksi dengan *use case* menampilkan daftar *homestay* berdasarkan skenario-skenario yang ada pada *use case* menampilkan daftar *homestay*.

Tabel 4.21 *Use Case Scenario* Menampilkan Daftar *Homestay*

Flow of Events for menampilkan daftar *homestay*	
Kode Kebutuhan	HOMIE-F-18
Objective	*Use case* ini bertujuan untuk menampilkan daftar seluruh *homestay* yang telah terdaftar kedalam sistem.
Actor	*Admin*
Pre-condition	Aktor sudah *login.*
Main Flow	1. Aktor memilih opsi daftar *homestay* pada saat menekan logo profile.

	2. Sistem menampilkan daftar dari seluruh *homestay* yang terdaftar pada sistem.
Alternative Flow	1. Apabila belum ada *homestay* yang terdaftar kedalam sistem, maka pesan "No Homestay Registered" akan ditampilkan oleh sistem.
Post-condition	Aktor melihat seluruh daftar *homestay* yang telah terdaftar kedalam sistem.

Pada Tabel 4.22 akan dijelaskan bagaimana aktor berinteraksi dengan *use case* menampilkan daftar *owner* berdasarkan skenario-skenario yang ada pada *use case* menampilkan daftar *owner*.

Tabel 4.22 *Use Case Scenario* Menampilkan Daftar *Owner*

Flow of Events for menampilkan daftar *owner*	
Kode Kebutuhan	HOMIE-F-19
Objective	*Use case* ini bertujuan untuk menampilkan daftar seluruh *owner* yang telah terdaftar kedalam sistem.
Actor	*Admin*
Pre-condition	Aktor sudah *login.*
Main Flow	1. Aktor memilih opsi daftar *owner* pada saat menekan logo profile. 2. Sistem menampilkan daftar dari seluruh *homestay* yang terdaftar pada sistem.
Alternative Flow	1. Apabila belum ada *homestay* yang terdaftar kedalam sistem, maka pesan "No Owner Registered" akan ditampilkan oleh sistem.
Post-condition	Aktor melihat seluruh daftar *owner* yang telah terdaftar kedalam sistem.

Pada Tabel 4.23 akan dijelaskan bagaimana aktor berinteraksi dengan *use case* meghapus *homestay* berdasarkan skenario-skenario yang ada pada *use case* menghapus *homestay*.

Tabel 4.23 *Use Case Scenario* Menghapus *Homestay*

Flow of Events for menghapus *homestay*	
Kode Kebutuhan	HOMIE-F-20
Objective	*Use case* ini bertujuan untuk menghapus *homestay* milik *owner* yang telah terdaftar didalam sistem.

Actor	*Admin*
Pre-condition	Aktor sudah berada pada halaman menampilkan daftar *homestay.*
Main Flow	1. Aktor memilih hapus *homestay* dari daftar *homestay* yang telah ditampilkan oleh sistem. 2. Sistem menghapus *homestay* milik *owner* dari sistem beserta kamar-kamar yang terdaftar dalam *homestay* tersebut.
Alternative Flow	-
Post-condition	Aktor berhasil menghapus *homestay* beserta kamar-kamar yang terdaftar didalam milik *owner* dari sistem.

Pada Tabel 4.24 akan dijelaskan bagaimana aktor berinteraksi dengan *use case* menghapus akun *owner* berdasarkan skenario-skenario yang ada pada *use case* menghapus akun *owner*.

Tabel 4.24 *Use Case Scenario* Menghapus Akun *Owner*

Flow of Events for menghapus akun *owner*	
Kode Kebutuhan	HOMIE-F-21
Objective	*Use case* ini bertujuan untuk menghapus akun *owner* dari sistem.
Actor	*Admin*
Pre-condition	Aktor sudah berada pada halaman menampilkan daftar *homestay* dan *owner.*
Main Flow	1. Aktor memilih hapus *owner* dari daftar *owner* yang telah ditampilkan oleh sistem. 2. Sistem menghapus akun *owner* dari sistem beserta *homestay* milik *owner* dan juga kamar-kamar yang terdaftar didalam *homestay* tersebut.
Alternative Flow	-
Post-condition	Aktor berhasil menghapus akun *owner* beserta *homestay* dan kamar-kamar milik *owner* dari sistem.

Pada Tabel 4.25 akan dijelaskan bagaimana aktor berinteraksi dengan *use case* menampilkan seluruh transaksi berdasarkan skenario-skenario yang ada pada *use case* menampilkan seluruh transaksi.

Tabel 4.25 *Use Case Scenario* Menampilkan Seluruh Transaksi

Flow of Events for menampilkan seluruh transaksi

Kode Kebutuhan	HOMIE-F-22
Objective	*Use case* ini bertujuan untuk melihat daftar transaksi yang telah terjadi pada seluruh *homestay.*
Actor	*Admin*
Pre-condition	Aktor sudah *login.*
Main Flow	1. Aktor memilih opsi daftar pemesanan pada saat menekan logo profile. 2. Sistem menampilkan daftar seluruh transaksi yang telah terjadi satu bulan lalu.
Alternative Flow	-
Post-condition	Aktor berhasil melihat seluruh transaksi yang telah terjadi pada sistem.

Pada Tabel 4.26 akan dijelaskan bagaimana aktor berinteraksi dengan *use case logout* berdasarkan skenario-skenario yang ada pada *use case logout*.

Tabel 4.26 *Use Case Scenario Logout*

Flow of Events for logout	
Kode Kebutuhan	HOMIE-F-23
Objective	*Use case* ini bertujuan untuk mengeluarkan *admin* atau *owner* dari sistem.
Actor	*Admin, Owner*
Pre-condition	Aktor sudah *login*.
Main Flow	1. Aktor memilih opsi *Logout.* 2. Sistem mengeluarkan akun yang sudah *login.* 3. Sistem menampilkan halaman utama dari web.
Alternative Flow	-
Post-condition	Aktor berhasil keluar dari *role*-nya dan menjadi *guest.*

Pada Tabel 4.27 akan dijelaskan bagaimana aktor berinteraksi dengan *use case* mengubah status pemesanan berdasarkan skenario-skenario yang ada pada *use case* mengubah status pemesanan.

Tabel 4.27 *Use Case Scenario* Mengubah Status Pemesanan

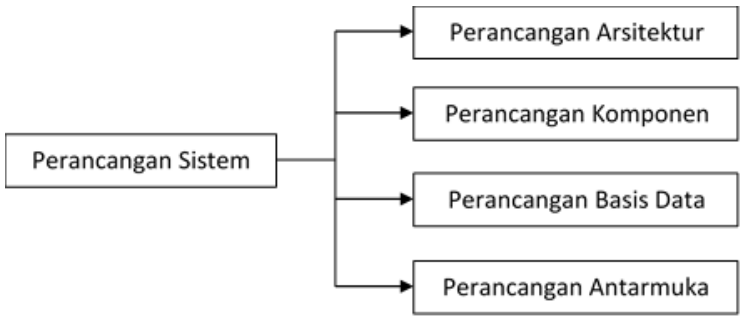
Flow of Events for mengubah status pemesanan	
Kode Kebutuhan	HOMIE-F-24

Objective	*Use case* ini bertujuan untuk mengubah status pemesanan yang telah dilakukan.
Actor	Midtrans
Pre-condition	Aktor telah mengetahui *end point* API ubah status pemesanan.
Main Flow	1. Aktor mengakses *end point* API ubah status pemesanan dengan mengirim data status pemesanan. 2. Sistem mengubah status pemesanan sesuai dengan data yang diterima. 3. Sistem mengirimkan *email* notifikasi kepada pemilik *homestay* ketika status *capture*(sukses).
Alternative Flow	-
Post-condition	Aktor berhasil melakukan perubahan status pemesanan pada sistem.

BAB 5 PERANCANGAN DAN IMPLEMENTASI SISTEM

5.1 Perancangan Sistem

Bagian ini dilakukan jika tahap analisis kebutuhan sudah diselesaikan maka tahap selanjutnya adalah perancangan. Tahap perancangan memiliki hubungan yang kuat dengan tahap rekayasa kebutuhan karena pada tahap perancangan menggunakan acuan berdasarkan hasil dari analisis kebutuhan. Tahap perancangan Aplikasi Reservasi dan Manajemen *Marketplace Homestay* akan dibagi menjadi beberapa bagian, diantaranya perancangan arsitektur, perancagan komponen, perancangan data, dan perancangan antarmuka. Struktur dari sub-bab perancangan sistem akan ditunjukkan pada Gambar 5.1.



Gambar 5.1 Struktur Sub-bab Perancangan Sistem

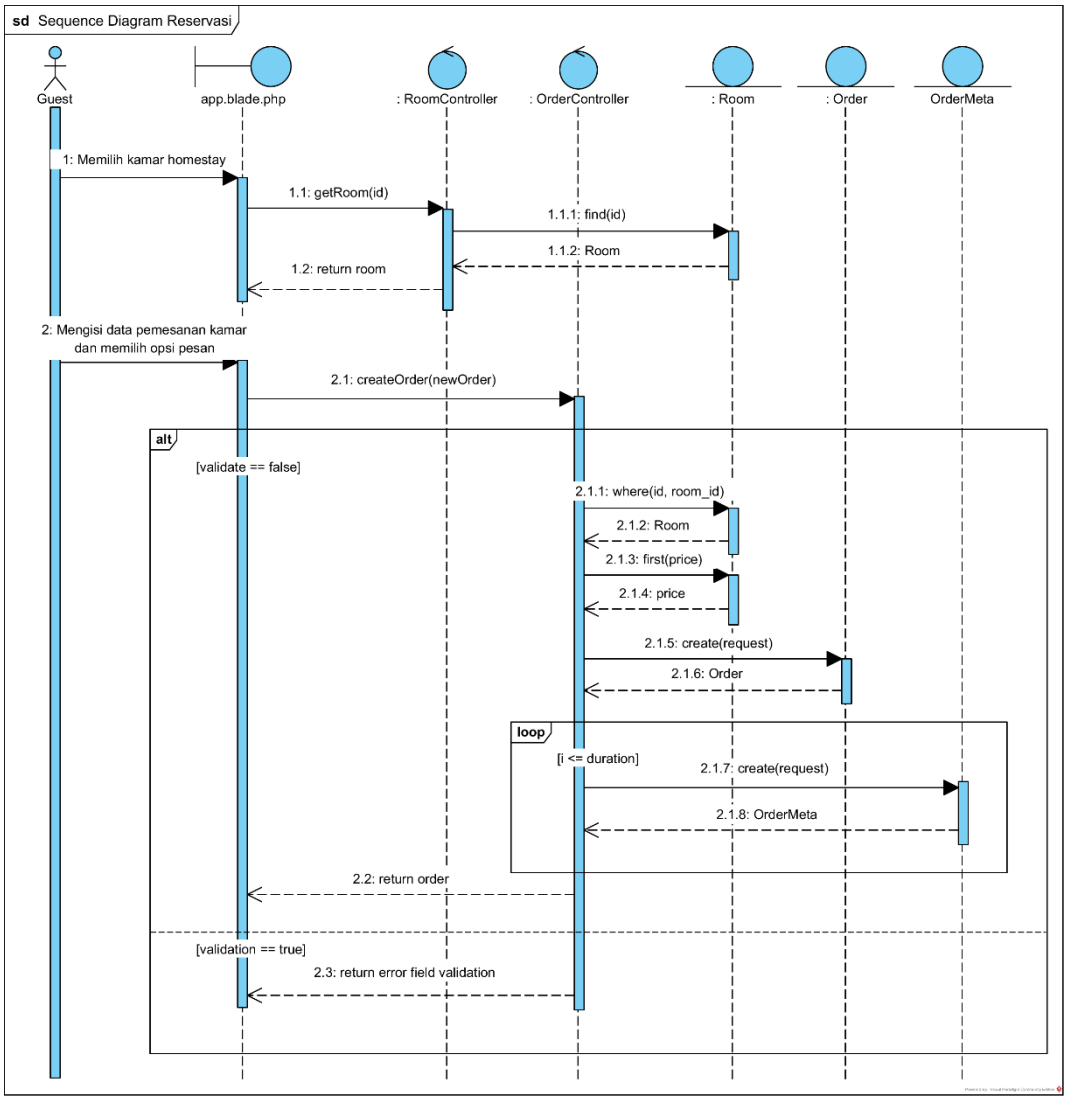
5.1.1 Perancangan Arsitektur

Pada perancangan arsitektur akan dipaparkan bagaimana suatu *class* berinteraksi dengan *class* yang lainnya dan bagaimana alur dari pertukaran data antar *object* yang direpresentasikan dalam bentuk *sequence diagram*. Pada tahap ini juga dijelaskan relasi antar *class* dalam sistem yang akan direpresentasikan dalam bentuk *class diagram*. Dalam pembuatan *sequence diagram* akan digunakan tiga fungsionalitas utama yaitu melakukan pemesanan kamar, melakukan *checkin,* dan melakukan *checkout*.

5.1.1.1 *Sequence Diagram* Melakukan Pemesanan Kamar

Pada Gambar 5.2 ditunjukkan *sequence diagram* melakukan pemesanan kamar. *Sequence diagram* ini memiliki tujuh notasi, diantaranya adalah aktor, *boundary* yaitu app.blade.php, *controller* yaitu RoomController dan OrderController, dan tiga buah *entity* yaitu Room, Order, dan OrderMeta. Dalam menjalankan *use case* melakukan pemesanan kamar, aktor harus terlebih dahulu opsi pilih kamar dengan melakukan klik pada tombol pilih kamar sehingga *method* showRoomList() pada *controller* RoomController dapat dipanggil dan menampilkan form pemesanan kembali ke *boundary* app.blade.php. Kemudian, aktor mengisikan dan memberikan data pemesanan kepada form dan memilih tombol pesan pada *boundary* app.blade.php untuk dapat memanggil *method* createOrder(newOrder) pada *controller* OrderController agar data pemesanan

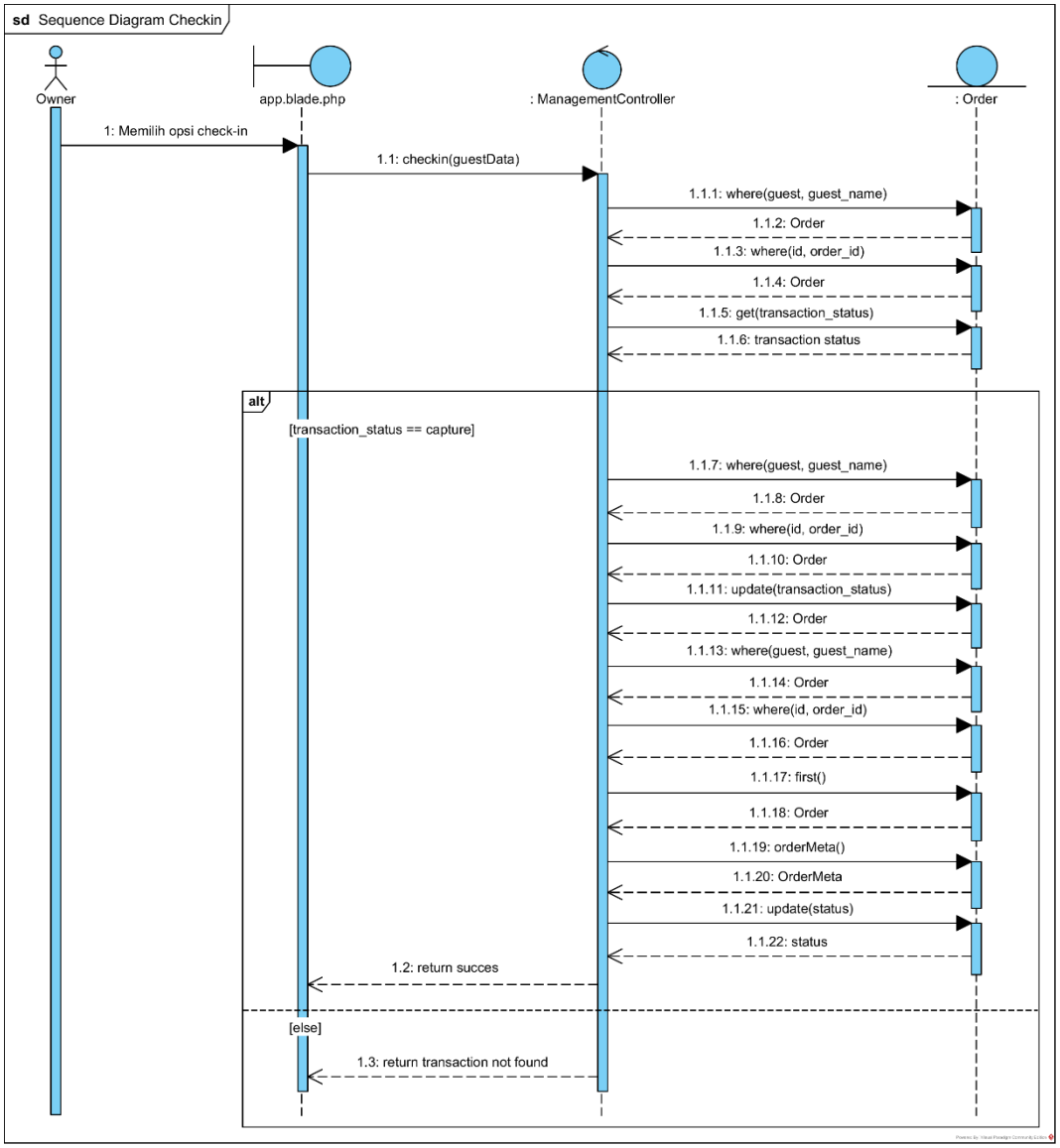
dapat disimpan kedalam *entity* Order dan *entity* OrderMeta yang nantinya akan digunakan dalam *use case* pembayaran.



Gambar 5.2 *Sequence Diagram* Melakukan Pemesanan Kamar

5.1.1.2 *Sequence Diagram* Melakukan *Check-In*

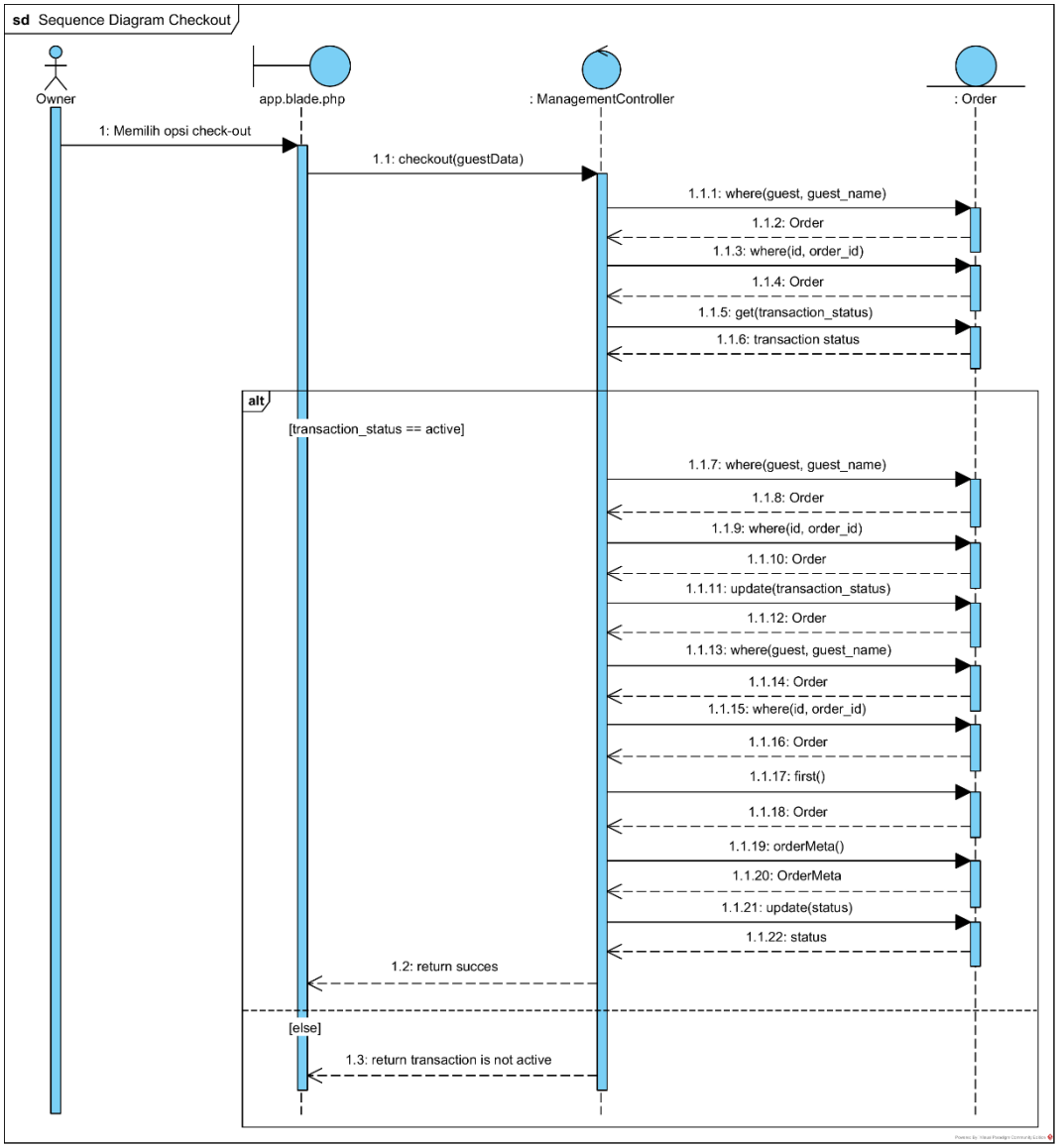
Pada Gambar 5.3 ditunjukkan *sequence diagram* melakukan *check-in*. *Sequence diagram* ini memiliki empat notasi, diantaranya adalah aktor, *boundary* yaitu app.blade.php, *controller* ManagementController, kemudian *entity* Order. Dalam menjalankan *use case* melakukan *check-in*, aktor memilih opsi *check-in* dengan cara melakukan klik pada tombol *check-in* agar dapat memanggil *method* checkin(guestData) pada *controller* ManagementController dan melakukan validasi pembayaran terhadap model *entity* Order, kemudian dilanjutkan dengan memanggil *method* update(transaction_status) untuk melakukan perubahan status telah *check-in* pada *entity* Order.



Gambar 5.3 *Sequence Diagram* Melakukan *Check-In*

5.1.1.3 *Sequence Diagram* Melakukan *Check-Out*

Pada Gambar 5.4 ditunjukkan *sequence diagram* melakukan *check-out*. *Sequence diagram* ini memiliki enam notasi, diantaranya adalah aktor, *boundary* yaitu app.blade.php, *controller* RoomController dan ManagementController, kemudian *entity* Room dan Order. Dalam menjalankan *use case* melakukan *check-out*, aktor memilih opsi *check-out* dengan cara melakukan klik pada tombol *check-out* agar dapat memanggil *method* checkout(guestData) pada *controller* ManagementController dan melakukan validasi pembayaran terhadap model *entity* Order, kemudian dilanjutkan dengan memanggil *method* update(transaction_status) untuk melakukan perubahan status telah *check-out* pada *entity* Order.



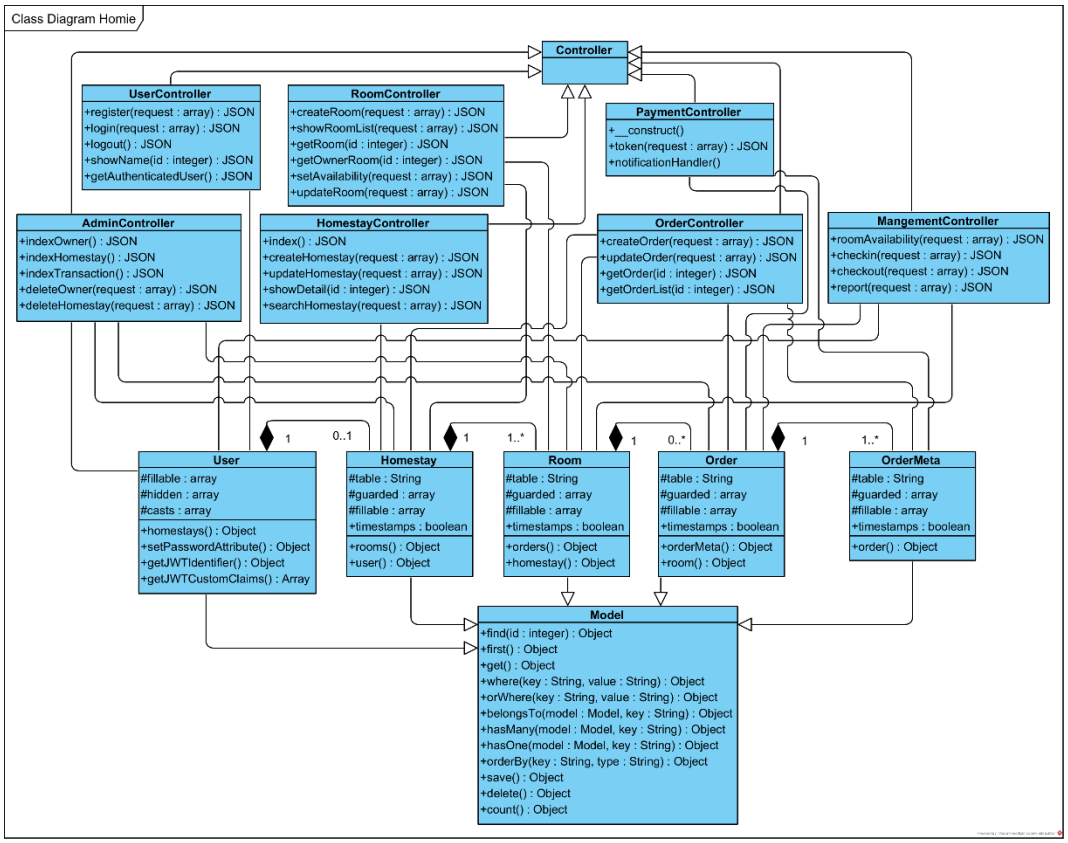
Gambar 5.4 *Sequence Diagram* Melakukan *Check-Out*

5.1.1.4 *Class Diagram*

Pada Gambar 5.5 ditunjukkan *class diagram* dari sistem. *Class diagram* ini bertujuan untuk memaparkan *class-class* pembentuk sistem apa saja yang terdapat didalam sistem ini dan juga untuk menjelaskan relasi atau hubungan-hubungan antar *class* dalam berinteraksi anatara satu dengan yang lainnya. Pada gambar *class diagram* Gambar 5.5 terdapat dua jenis *class* yaitu *class* yang memiliki peran sebagai *controller* dan *class* yang memiliki peran sebagai *model*. Pada *class model* akan menerapkan struktur *default* dari *framework* Laravel untuk melakukan *mass asignment* dimana pendefinisian atau pendeklarasian *attribute* dilakukan dalam array fillable, guarded, dan hidden. Tujuan dari *mass assignment* adalah untuk melakukan manipulasi data dalam jumlah banyak.

Pada jenis *class controller* terdapat satu *parent class* yaitu *class Controller* dan tujuh *class child* yaitu AdminController, UserController, HomestayController,

RoomController, OrderController, PaymentController, dan ManagementController. Selanjutnya pada jenis *class model* terdapat satu *parent class* yaitu *class model* dan lima *child class* yaitu User, Homestay, Room, Order, dan OrderMeta. Seluruh *class child* akan mewarasi *attribute* dan *method* dari *class parent*-nya.



Gambar 5.5 *Class Diagram* Aplikasi Reservasi dan Manajemen *Homestay*

5.1.2 Perancangan Komponen

Pada bagian ini dipaparkan penjelasan secara lebih rinci akan sub-sistem dari tiap-tiap komponen yang ada dalam pengembangan perangkat lunak aplikasi ini. Pada bagian ini juga akan didefinisikan secara rinci bagaimana proses algoritme yang terjadi dalam format bahasa *pseudocode.* Terdapat tiga *method* yang akan digunakan dalam perancangan komponen diantarnya *method createOrder, method checkin,* dan *method checkout*.

5.1.2.1 Perancangan Komponen *Method* CreateOrder

Pada Tabel 5.1 dipaparkan algoritme *method* Order yang berada pada *class* OrderController. Tujuan dari method ini adalah melakukan pemesanan kamar *homestay*. Algoritme ini akan melakukan inisialisasi data berdasarkan data yang didapat dari parameter method kemudian dialkukan validasi terhadap data yang dibawa, jika terdapat *field* kosong maka proses akan berakhir dan mengirimkan status sebagai *return value*. Jika berhasil maka akan dilanjutkan dengan validasi ketersedian kamar. Jika validasi ketersedian kamar berhasil maka akan dilanjutkan

dengan menambahkan data pesanan tersebut ke database. Jika proses penyimpanan ke database berhasil maka proses pada method ini akan berakhir dan dilanjutkan dengan mengirimkan status berhasil sebagai *return value*.

Nama *class*: OrderController

Nama *method*: createOrder

Tabel 5.1 *Pseudocode* Algoritme *Method* createOrder

```
Pseudocode algoritme method createOrder
1   BEGIN
2   orderData = Request
3
4   validate = [
5      orderData field name is required,
6      orderData field email is required,
7      orderData field phoneNumber is required,
8      orderData field rooms is required,
9      orderData field checkinDate is required,
10     orderData field duration is required,
11  ]
12
13  IF (validate is FALSE)
14     RETURN response to JSON (data with fail validate
15     field and validate fail status)
16  END IF
17
18  checkinDate = checkin_date of orderData formatted with
19  (Y-M-D) date format
20  checkoutDate = checkin_date of orderData + duration day
21  of orderData formatted with (Y-M-D) date format
22  roomPrice = get first price from database table room
23  where id equals room_id of orderData
24  priceTotal = price of roomPrice * duration of orderData
25
26  order = create data to database table Order with value
27  (orderData all attribute, checkoutDate, priceTotal)
28
29  FOR (from i = 0 until duration of orderData, increment
30  by 1){
31       order_id = id of orderData
32       stay_date  =  checkin_date  of  orderData  +  i
33  formatted with (Y-M-D) date format
34       orderMeta  =  create  data  to  database  table
35  OrderMeta with value (order_id, stay_date, status = no
36  status)
37  END FOR
38
39  RETURN response to JSON (order and store success status
40  201)
41  END
```

5.1.2.2 Perancangan Komponen *Method* Checkin

Pada Tabel 5.2 dipaparkan algoritme *method* CheckIn yang berada pada *class* ManagementController. Tujuan dari method ini adalah melakukan perubahan status kamar *homestay*. Algoritme ini akan melakukan inisialisasi data berdasarkan data yang didapat dari parameter method. Jika berhasil maka akan dilanjutkan dengan verifikasi kode pembayaran. Jika kode pembyaran yang diinputkan tidak sesuai dengan kode pembayaran yang ada pada *database* Transaksi maka akan dilanjutkan pada proses *return value* berupa *status fail paymentCode*. Jika verifikasi kode pembayaran berhasil maka status kamar pada *database* akan diubah menjadi *available*. Setelah semua proses berhasil dijalankan, maka akan dilanjutkan dengan proses *return value* berupa success status sebagai informasi tambahan bahwa fungsi *check-in* berhasil.

Nama *class*: ManagementController

Nama *method*: Checkin

Tabel 5.2 *Pseudocode* Algoritme *Method* Checkin

```
Pseudocode algoritme method checkin
1   BEGIN
2   data = request
3
4   transactionStatus = get transaction_status from
5   database table Order where guest equals guest_name of
6   data and where id equals order_id of data
7
8   IF (transaction_status of index 0 transactionStatus is
9   capture) {
10     transactionStatus = update data transaction_status
11  to active from database table Order where guest equals
12  guest_name of data and where id equals order_id of data
13     orderMetaUpdate = get first data from database table
14  Order where guest equals guest_name of data and where
15  id equals order_id of data
16     orderMetaUpdate update status to active of orderMeta
17     RETURN response to JSON (success with success status
18  code 200)
19  }
20
21  ELSE {
22     RETURN response to JSON (transaction not found with
23  not found status code 404)
24  }
25  END
```

5.1.2.3 Perancangan Komponen *Method* Checkout

Pada Tabel 5.3 dipaparkan algoritme *method* CheckOut yang berada pada *class* ManagementController. Tujuan dari method ini adalah melakukan perubahan status kamar *homestay*, algoritme method ini hampir sama dengan method

checkIn namun ada beberapa proses yang berbeda. Algoritme ini akan melakukan inisialisasi data berdasarkan data yang didapat dari parameter method. Jika berhasil maka akan dilanjutkan dengan perubahan status pada *database*. Jika data status yang dibawa memiliki nilai available, maka proses pengubahan status pada database akan dieksekusi. Pada tahap akhir method, proses *return value* berupa status sukses akan dieksekusi.

Nama *class*: ManagementController

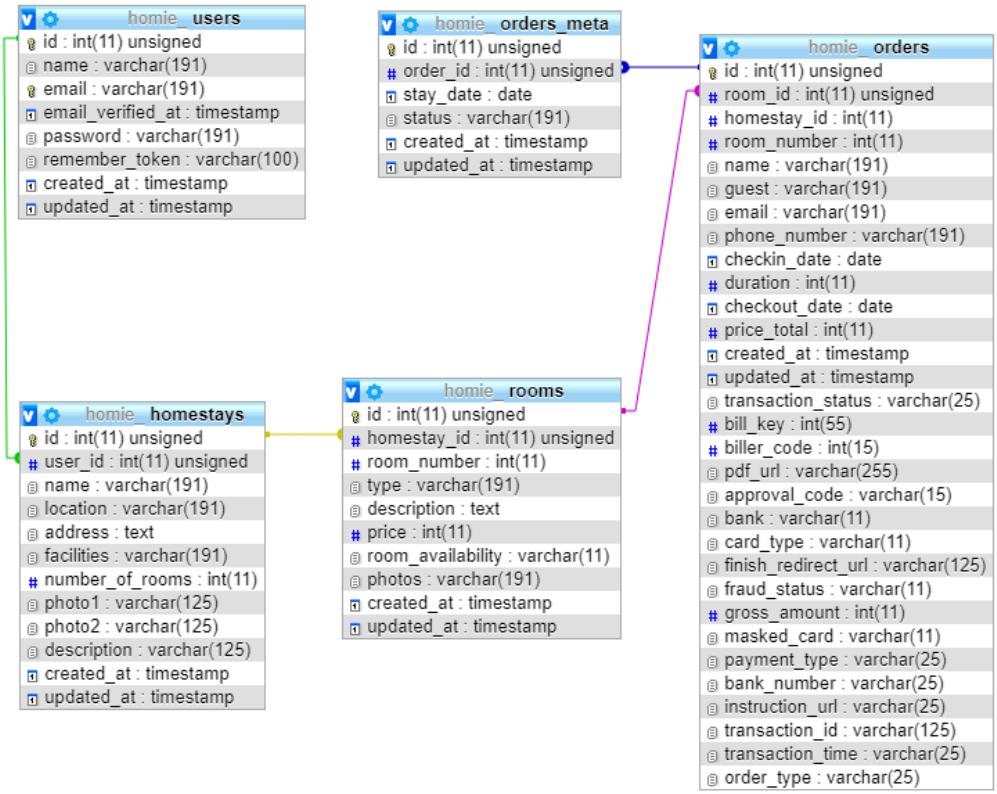
Nama *method*: Checkout

Tabel 5.3 *Pseudocode* Algoritme *Method* Checkout

Pseudocode algoritme method checkout	
1	BEGIN
2	data = request
3	
4	transactionStatus = get transaction_status from
5	database table Order where guest equals guest_name of
6	data and where id equals order_id of data
7	
8	IF (transaction_status of index 0 transactionStatus is
9	active) {
10	transactionStatus = update data transaction_status
11	to used from database table Order where guest equals
12	guest_name of data and where id equals order_id of data
13	orderMetaUpdate = get first data from database table
14	Order where guest equals guest_name of data and where
15	id equals order_id of data
16	orderMetaUpdate update status to used of orderMeta
17	RETURN response to JSON (success with success status
18	code 200)
19	}
20	
21	ELSE {
22	RETURN response to JSON (transaction is not active
23	with not found status code 404)
24	}
25	END

5.1.3 Perancangan Basis Data

Pada tahap perancangan basis data, menghasilkan suatu rancangan data dalam bentuk *Physical Data Model* (PDM) yang mana hasil dari rancangan tersebut diproyeksikan pada Gambar 5.6.



Gambar 5.6 Perancangan *Physical Data Model*

5.1.4 Perancangan Antarmuka

Pada tahap perancangan antarmuka akan dipaparkan rancangan tampilan dari sistem. Rancangan ini akan menjelaskan komponen-kompnen dalam tiap halaman. Hasil dari rancangan antarmuka ini akan menjadi penghubung interaksi sistem dengan pengguna. Perancangan antarmuka ini akan menjadi acuan dalam implementasi antarmuka sistem. Pada penelitian ini terdapat enam perancangan antarmuka utama yaitu, perancangan antarmuka *Homepage,* perancangan antarmuka *Homestay List*, perancangan antarmuka *Homestay Detail*, pearancangan antarmua *Order Form*, perancangan antarmuka *Booking Details*, dan perancangan antarmuka *Owner Room Management*.

Perancangan antarmuka yang dilakukan akan menerapkan prinsip-prinsip dasar dalam melakukan perancangan antarmuka. Prinsip-prinsip ini mengacu kepada *Ten Good Deeds in Web Design* yang didasari oleh teori *Jakob's Law of Internet User Experience*. Sepuluh poin tersebut diantaranya adalah:

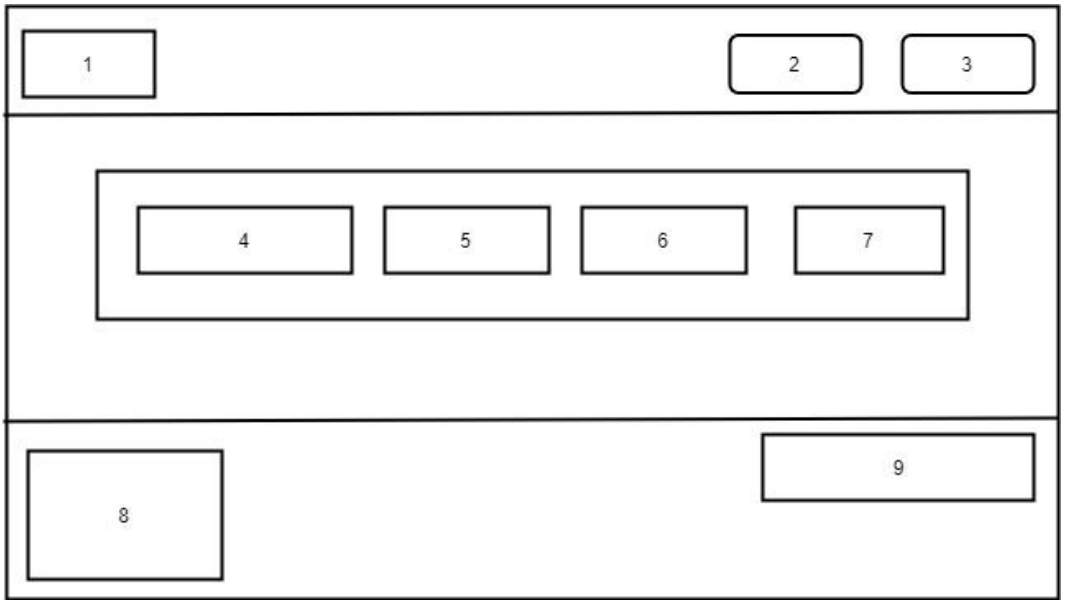
1. Memasang logo pada tiap-tiap halaman dimana logo tersebut akan mereferensi ke halaman utama.
2. Menyediakan fitur *search* apabila halaman mencapai 100 lebih.
3. Menulis judul halaman secara jelas, tepat sasaran, dan simple.

4. Membuat struktur halaman yang mempermudah pengguna dalam melakukan pencarian sehingga tidak terjadi penumpukan informasi dalam satu tampilan.
5. Gunakan *hypertext* untuk membuat struktur halaman agar yang ditampilkan hanyalah poin-poin penting dari produk dan halaman kedua menampilkan detail informasi produk, tujuannya adalah untuk menghindari detail informasi yang tidak dibutuhkan oleh pengguna.
6. Gunakan foto produk yang tidak berantakan dan bertumpuk-tumpuk.
7. Penggunaan gambar yang memiliki relevansi tinggi sehingga menampilkan gambar yang benar-benar sesuai.
8. Gunakan judul tautan untuk memudahkan pengguna mengetahui kemana arah halaman.
9. Pastikan halaman-halaman penting dapat diakses oleh pengguna dengan disabilitas.
10. Lakukan hal yang sama seperti orang lain lakukan, jika user website besar menerapkan sesuatu dengan cara tertentu maka ikutilah karena pengguna akan mengharapkan hal yang seperti itu pula pada website anda.

Perancangan antarmuka beserta penjelasannya akan diperjelas pada sub bab 5.1.4.1 sampai dengan sub bab 5.1.4.6.

5.1.4.1 Perancangan Antarmuka *Homepage*

Perancangan antarmuka *Homepage* ditunjukkan pada Gambar 5.7. Pada Gambar 5.7 dipaparkan rancangan antarmuka *Homepage* yang terdiri dari dua bagian utama yaitu *header* dan *body*. Total dari seluruh komponen yang ada pada halaman ini adalah sembilan. *Header* pada halaman ini bertujuan untuk memudahkan pengguna dalam menggunakan fungsi yang harus siap sedia dalam tiap halaman yang mana halaman ini disusun oleh tiga komponen diantaranya logo, *login*, dan, register. Sedangkan pada bagian *body* disusun berdasarkan konteks halaman itu sendiri dimana pada halaman *homepage,* konteks tersebut adalah pencarian *homestay* yang bertujuan untuk langsung mencari *homestay* sesuai kebutuhan dan keinginan pengguna tanpa harus menampilkan komponen-komponen yang tidak dan belum diperlukan oleh pengguna. Bagian pada *body* disusun oleh 6 komponen diantaranya, lokasi, tamu, kamar, *checkin*, durasi, dan *search*.



Gambar 5.7 Perancangan Antarmuka *Homepage*

Gambar 5.7 merepresentasikan gambar perancangan antarmuka halaman utama atau *homepage*. Penjelasan dari gambar tersebut terdapat pada Tabel 5.4.

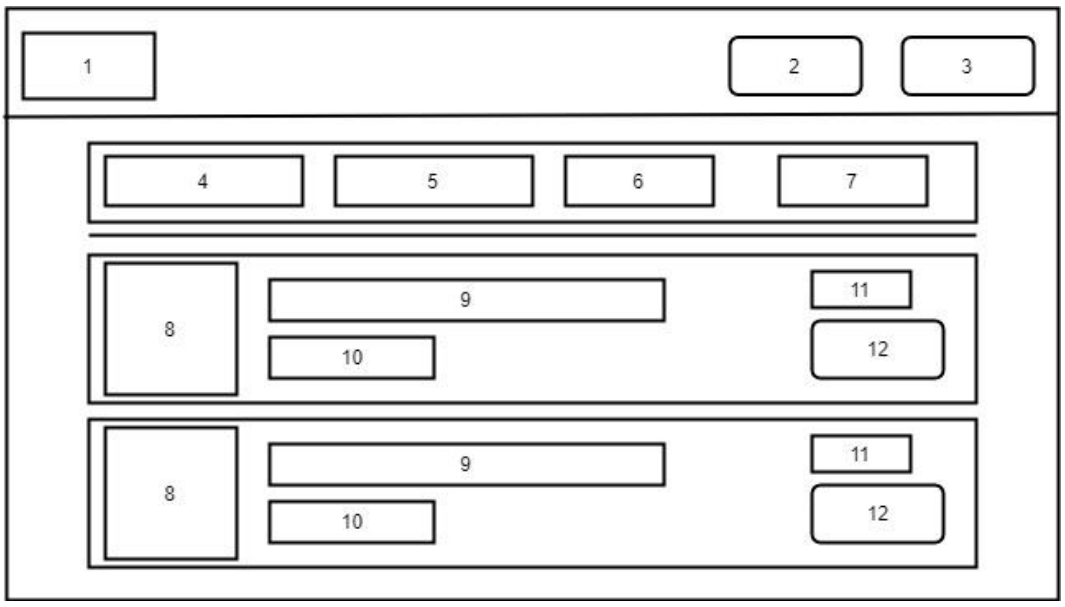
Tabel 5.4 Keterangan Gambar Perancangan Antarmuka *Homepage*

No	Nama Objek	Tipe	Keterangan
1	Logo	Gambar	Menampilkan logo aplikasi dalam bentuk gambar
2	Register	Tombol	Tombol untuk melakukan proses pendaftaran
3	*Login*	Tombol	Tombol untuk melakukan proses *login*
4	Lokasi	*Textfield*	Menerima inputan user berupa lokasi daerah
5	*Check-in*	*Textfield*	Menerima inputan user berupa tanggal melakukan *check-in*
6	Duration	*Textfield*	Menerima inputan user berupa lamanya user akan menginap
7	Search	Tombol	Tombol untuk melakukan pencarian berdasarkan informasi pesanan user dalam *textfield*
8	About Us	Label	Menampilkan informasi About Us pada aplikasi

9	Email	Label	Menampilkan nama aplikasi dan kontak *email* yang berperan sebagai *costumer service*

5.1.4.2 Perancangan Antarmuka *Homestay List*

Perancangan antarmuka *Homestay List* ditunjukkan pada Gambar 5.8. Pada Gambar 5.8 dipaparkan rancangan antarmuka *Homestay List* yang terdiri dari dua bagian utama yaitu *header* dan *body*. Total dari seluruh komponen yang ada pada halaman ini adalah tiga belas. *Header* pada halaman ini bertujuan untuk memudahkan pengguna dalam menggunakan fungsi yang harus siap sedia dalam tiap halaman yang mana halaman ini disusun oleh tiga komponen diantaranya logo, *login*, dan, register. Sedangkan pada bagian *body* disusun berdasarkan konteks halaman itu sendiri dimana pada halaman *homestay list,* konteks tersebut adalah daftar *homestay* berdasarkan parameter-parameter yang telah dimasukkan oleh pengguna. Bagian ini bertujuan untuk menampilkan *homestay* dalam bentuk daftar menurun kebawah yang mempermudah pengguna dalam memilih *homestay.* Informasi yang dipaparkan pada daftar tersebut dibuat cukup sederhana agar dapat menampilkan informasi-informasi penting saja dan tidak memenuhi layer pengguna. Pada bagian *body* juga terdapat *field* pencarian *homestay* yang bertujuan untuk langsung mencari *homestay* jika pengguna ingin melakukan pencarian ulang. Jumlah komponen penyusun pada bagian *body* halaman ini adalah sepuluh diantaranya, lokasi, tamu, kamar, *checkin*, durasi, dan *search*, gambar *homestay*, nama *homestay*, alamat *homestay*, dan tombol select beserta harga.



Gambar 5.8 Perancangan Antarmuka *Homestay List*

Gambar 5.8 merepresentasikan gambar perancangan antarmuka *homestay list*. Penjelasan dari gambar tersebut ditunjukkan pada Tabel 5.5.

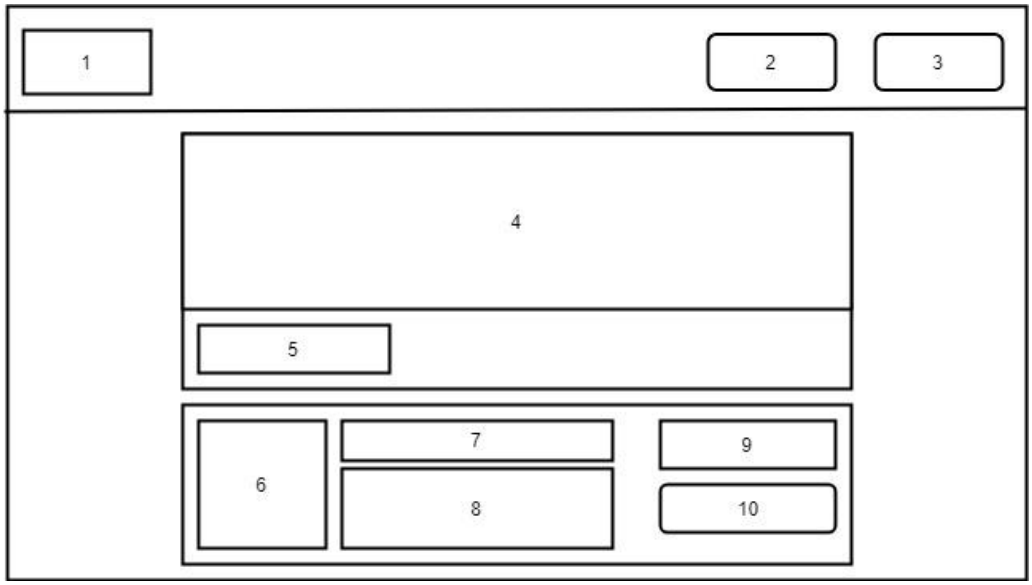
Tabel 5.5 Keterangan Gambar Perancangan Antarmuka *Homestay List*

No	Nama Objek	Tipe	Keterangan
1	Logo	Gambar	Menampilkan logo aplikasi dalam bentuk gambar
2	Register	Tombol	Tombol untuk melakukan proses pendaftaran
3	*Login*	Tombol	Tombol untuk melakukan proses *login*
4	Lokasi	*Textfield*	Menerima inputan user berupa lokasi daerah
5	*Check-in*	*Textfield*	Menerima inputan user berupa tanggal melakukan *check-in*
6	Duration	*Textfield*	Menerima inputan user berupa lamanya user akan menginap
7	Search	Tombol	Tombol untuk melakukan pencarian berdasarkan informasi pesanan user dalam *textfield*
8	Gambar *Homestay*	Gambar	Menampilkan gambar yang dimiliki oleh *homestay* tersebut
9	Nama	*Text*	Menampilkan nama *homestay* dalam bentuk text
10	Alamat	*Text*	Menampilkan alamat *homestay* dalam bentuk text
11	Harga	Text	Merupakan text untuk menampilkan harga terendah dari kamar yang tersedia
12	Select	Tombol	Merupakan Tombol untuk memasuki halaman detail *Homestay*

5.1.4.3 Perancangan Antarmuka *Homestay Detail*

Perancangan antarmuka *Homestay Detail* ditunjukkan pada Gambar 5.9. Pada Gambar 5.9 dipaparkan rancangan antarmuka *Homestay Detail* yang terdiri dari dua bagian utama yaitu *header* dan *body*. Total dari seluruh komponen yang ada pada halaman ini adalah tiga belas. *Header* pada halaman ini bertujuan untuk memudahkan pengguna dalam menggunakan fungsi yang harus siap sedia dalam tiap halaman yang mana halaman ini disusun oleh tiga komponen diantaranya logo, *login*, dan, register. Sedangkan pada bagian *body* disusun berdasarkan

konteks halaman itu sendiri dimana pada halaman *homestay detail,* konteks tersebut adalah informasi yang lebih mendalam tentang *homestay* yang telah dipilih oleh pengguna. Bagian ini bertujuan untuk menampilkan informasi *homestay* yang lebih lengkap sehingga pengguna mendapatkan informasi-informasi seperti gambar *homestay* yang lebih lengkap, fasilitas-fasilitas yang disediakan, dan daftar kamar *homestay* berdasarkan tipe-tipe yang disediakan.



Gambar 5.9 Perancangan Antarmuka *Homestay Detail*

Gambar 5.9 merepresentasikan gambar perancangan antarmuka *homestay detail*. Penjelasan dari gambar tersebut terdapat pada Tabel 5.6.

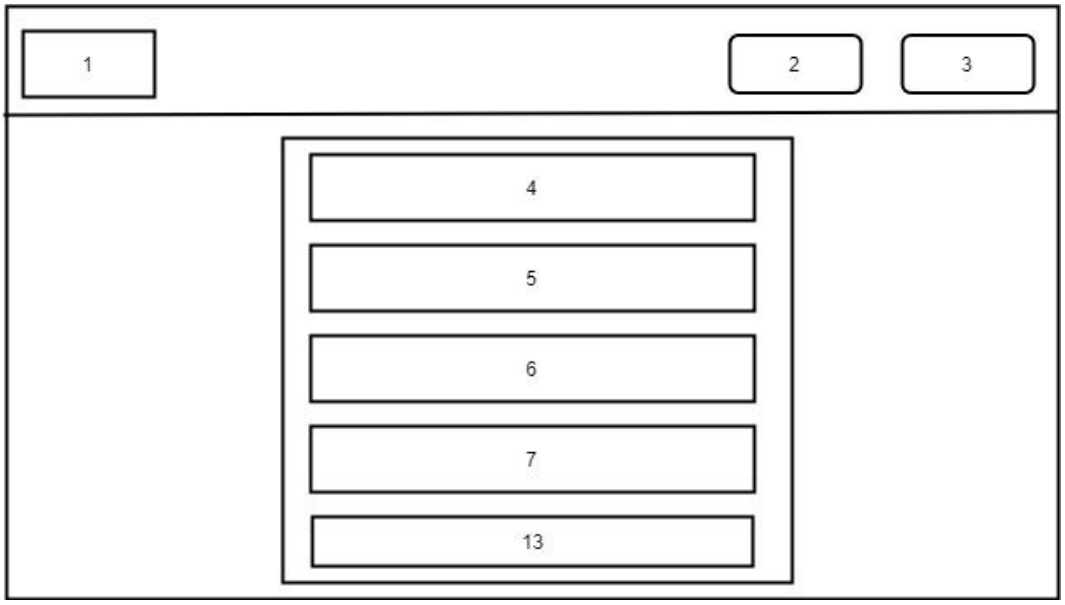
Tabel 5.6 Keterangan Gambar Perancangan Antarmuka *Homestay Detail*

No	Nama Objek	Tipe	Keterangan
1	Logo	Gambar	Menampilkan logo aplikasi dalam bentuk gambar
2	Register	Tombol	Tombol untuk melakukan proses pendaftaran
3	*Login*	Tombol	Tombol melakukan proses untuk *login*
4	Gambar *Homestay*	Gambar	Menampilkan gambar utama dari h*omestay* dalam ukuran yang besar
5	Detail *Homestay*	*Text*	Merupakan informasi yang lebih detail yang terdapat pada *homestay* dan ditampilkan dalam bentuk text

6	Gambar Kamar	Gambar	Menampilkan gambar kamar sesuai dengan tipe kamar
7	Nomor Kamar	*Text*	Menampilkan nomor kamar dalam bentuk text
8	Deskripsi	*Text*	Menampilkan deskripsi singkat tentang kamar
9	Harga Kamar	*Text*	Menampilkan harga kamar sesuai dengan tipe kamar
10	Pesan	Tombol	Merupkan tombol yang bertujuan untuk melakukan pemesanan kamar yang dipilih

5.1.4.4 Perancangan Antarmuka *Order Form*

Perancangan antarmuka *Order Form* ditunjukkan pada Gambar 5.10. Pada Gambar 5.10 dipaparkan rancangan antarmuka *Order Form* yang terdiri dari dua bagian utama yaitu *header* dan *body*. Total dari seluruh komponen yang ada pada halaman ini adalah tiga belas. *Header* pada halaman ini bertujuan untuk memudahkan pengguna dalam menggunakan fungsi yang harus siap sedia dalam tiap halaman yang mana halaman ini disusun oleh tiga komponen diantaranya logo, *login*, dan, register. Sedangkan pada bagian *body* disusun berdasarkan konteks halaman itu sendiri dimana pada halaman *order form,* konteks tersebut adalah form informasi pemesan dan detail pemesanan sesuai dengan pilihan yang telah dipilih oleh pengguna. Bagian ini memaparkan dua sub bagian yaitu form informasi pemesanan disebelah kiri dan detail pemesanan disebelah kanan. Tujuannya untuk tetap menampilkan detail pemesanan agar pengguna dapat secara tidak langsung maupun langsung melakukan pengecekan ulang terhadap pilihannya. Informasi yang dipaparkan pada detail pemesanan tersebut dibuat cukup sederhana namun memiliki poin-poin penting seperti nama dan alamat *homestay*, durasi kamar, tanggal *check-in*, tanggal *check-out* serta rincian harga. Jumlah komponen penyusun pada bagian *body* halaman ini adalah sepuluh komponen.



Gambar 5.10 Perancangan Antarmuka *Order Form*

Gambar 5.10 merepresentasikan gambar perancangan antarmuka *order form*. Penjelasan dari gambar tersebut terdapat pada Tabel 5.7.

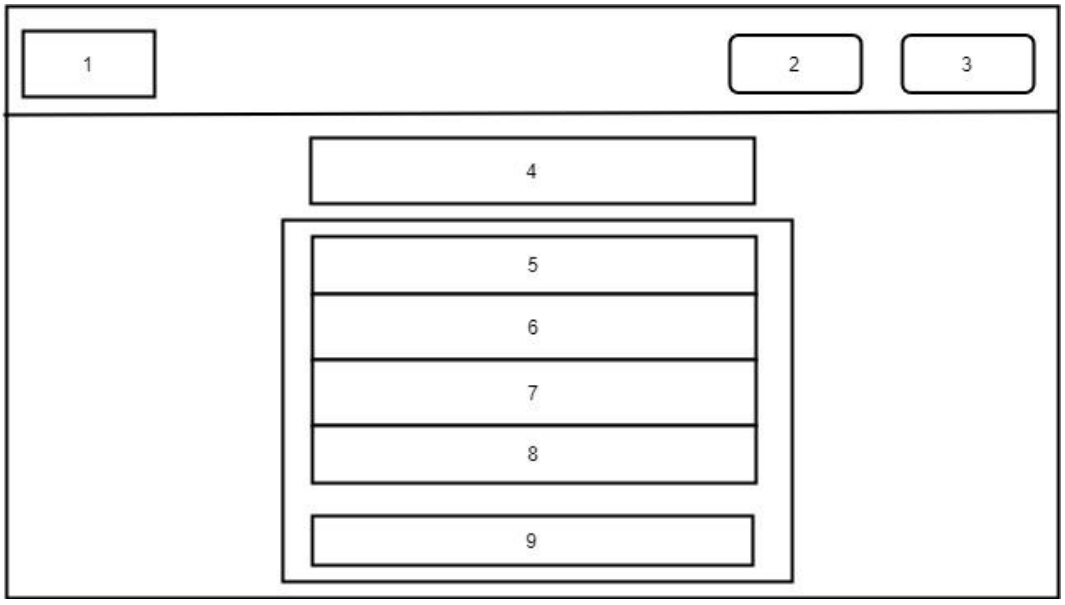
Tabel 5.7 Keterangan Gambar Perancangan Antarmuka *Order Form*

No	Nama Objek	Tipe	Keterangan
1	Logo	Gambar	Menampilkan logo aplikasi dalam bentuk gambar
2	Register	Tombol	Tombol untuk melakukan proses pendaftaran
3	*Login*	Tombol	Tombol melakukan proses untuk *login*
4	Nama Pemesan	*Textfield*	Menerima inputan user berupa nama dari customer yang memesan kamar
5	Nama Tamu	*Textfield*	Menerima inputan user berupa nama dari customer yang akan menjadi tamu
6	Nomor Telepon	*Textfield*	Menerima inputan user berupa nomor handphone dari customer yang memesan kamar
7	Email	*Textfield*	Menerima inputan user berupa *email* dari customer yang memesan kamar
8	Konfirmasi	Tombol	Merupakan tombol konfirmasi dan bertujuan untuk

			melanjutkan ke bagian pembayaran

5.1.4.5 Perancangan Antarmuka *Booking Details*

Perancangan antarmuka *Booking Details* ditunjukkan pada Gambar 5.11. Pada Gambar 5.11 dipaparkan rancangan antarmuka *Booking Details* yang terdiri dari dua bagian utama yaitu *header* dan *body*. Total dari seluruh komponen yang ada pada halaman ini adalah tiga belas. *Header* pada halaman ini bertujuan untuk memudahkan pengguna dalam menggunakan fungsi yang harus siap sedia dalam tiap halaman yang mana halaman ini disusun oleh tiga komponen diantaranya logo, *login*, dan, register. Sedangkan pada bagian *body* disusun berdasarkan konteks halaman itu sendiri dimana pada halaman *order detail,* konteks tersebut adalah form informasi pemesan dan detail pemesanan sesuai dengan pilihan yang telah dipilih oleh pengguna. Bagian ini memaparkan dua sub bagian yaitu form informasi pemesanan disebelah kiri dan detail pemesanan disebelah kanan. Tujuannya untuk tetap menampilkan detail pemesanan agar pengguna dapat secara tidak langsung maupun langsung melakukan pengecekan ulang terhadap pilihannya. Informasi yang dipaparkan pada detail pemesanan tersebut dibuat cukup sederhana namun memiliki poin-poin penting seperti nama dan alamat *homestay*, durasi kamar, tanggal *check-in*, tanggal *check-out* serta rincian harga. Jumlah komponen penyusun pada bagian *body* halaman ini adalah sepuluh komponen.



Gambar 5.11 Perancangan Antarmuka *Booking Details*

Gambar 5.11 merepresentasikan gambar perancangan antarmuka *booking details*. Penjelasan dari gambar tersebut ditunjukkan pada Tabel 5.8.

Tabel 5.8 Keterangan Gambar Perancangan Antarmuka *Booking Details*

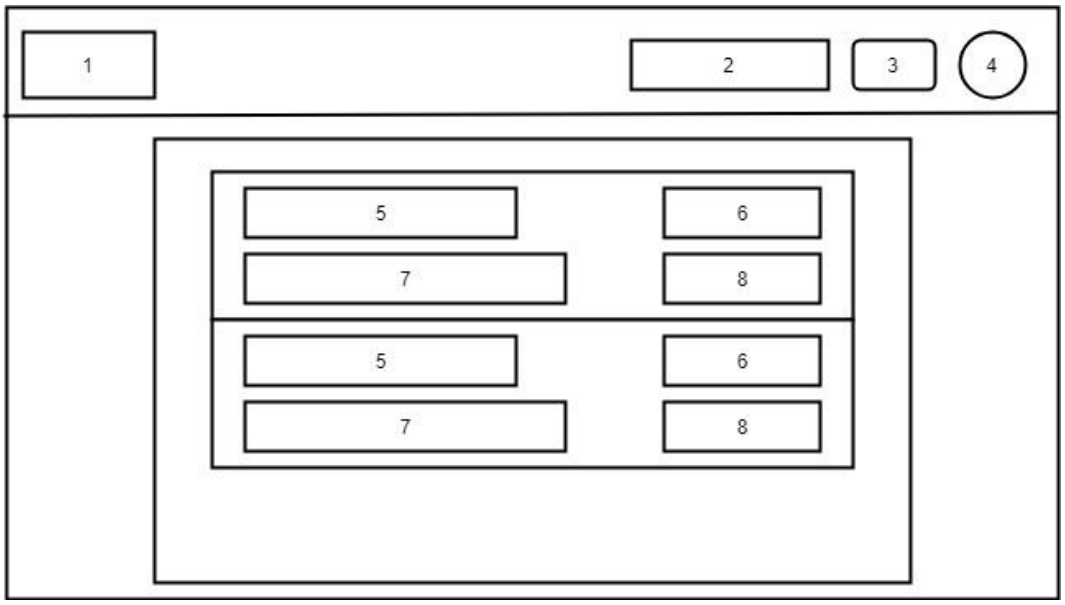
No	Nama Objek	Tipe	Keterangan

1	Logo	Gambar	Menampilkan logo aplikasi dalam bentuk gambar
2	Register	Tombol	Tombol untuk melakukan proses pendaftaran
3	*Login*	Tombol	Tombol melakukan proses untuk *login*
4	Booking Details	*Text*	Menampilkan Tulisan Booking Details dalam bentuk text
5	*Homestay* Detail	*Text*	Menampilkan informasi *homestay* yang dipilih dalam bentuk text
6	Stay Details	*Text*	Menampilkan informasi tanggal menginap dan lama menginap beserta tanggal keluar dalam bentuk text
7	Costumer Data	*Text*	Menampilkan informasi pemesan kamar dalam bentuk text
8	Total Price	*Text*	Menampilkan harga total dari pemesanan kamar dalam bentuk text
9	Pay	*Button*	Tombol untuk melanjutkan pembayran cashless menggunakan *paymeny gateway* midtrans

5.1.4.6 Perancangan Antarmuka *Owner Room Management*

Perancangan antarmuka *Owner Room Management* ditunjukkan pada Gambar 5.12. Pada Gambar 5.12 dipaparkan rancangan antarmuka *Owner Room Management* yang terdiri dari dua bagian utama yaitu *header* dan *body*. Total dari seluruh komponen yang ada pada halaman ini adalah tiga belas. *Header* pada halaman ini bertujuan untuk memudahkan pengguna dalam menggunakan fungsi yang harus siap sedia dalam tiap halaman yang mana halaman ini disusun oleh empat komponen diantaranya logo, *welcome message*, notifikasi yang berisi informasi pemesanan kamar yang telah dibayar, dan profile yang berisi menu *My Homestay*. Sedikit berbeda dengan rancangan *header* sebelumnya dikarenakan *header* ini sengaja dikhususkan untuk *owner homestay* agar mempermudah dalam mengelola kamar *homestay*. Sedangkan pada bagian *body* disusun tetap berdasarkan konteks halaman itu sendiri dimana pada halaman *owner room management,* konteks tersebut adalah daftar kamar yang dimiliki *homestay* berdasarkan status dari kamar tersebut. Bagian ini berisi informasi dari kamar yang

telah diisi atau *check-in*, diantaranya nama tamu, durasi menginap, tanggal *check-out*, dan kontak *handphone* tamu. Tujuan dari informasi tersebut untuk memudahkan pengelola atau *owner homestay* dalam menjaga penggunaan kamar. Kemudian terdapat sebuah tombol yang memiliki dua fungsi yaitu *check-in* jika kamar tersebut kosong dan *check-out* jika kamar tersebut telah diisi. Ukuran tombol tersebut dibuat cukup besar bertujuan juga sebagai informasi status kamar yang menandakan bahwa kamar tersebut terisi atau tidak. Jumlah komponen penyusun pada bagian *body* halaman ini adalah lima diantaranya, nama tamu, durasi menginap, tanggal *check-out*, nomor telepon tamu, dan tombol untuk melakukan *check-in* dan *check-out*.



Gambar 5.12 Perancangan Antarmuka *Owner Room Management*

Gambar 5.12 merepresentasikan gambar perancangan antarmuka halaman utama atau *homepage*. Rincian dari gambar tersebut ditunjukkan pada Tabel 5.9.

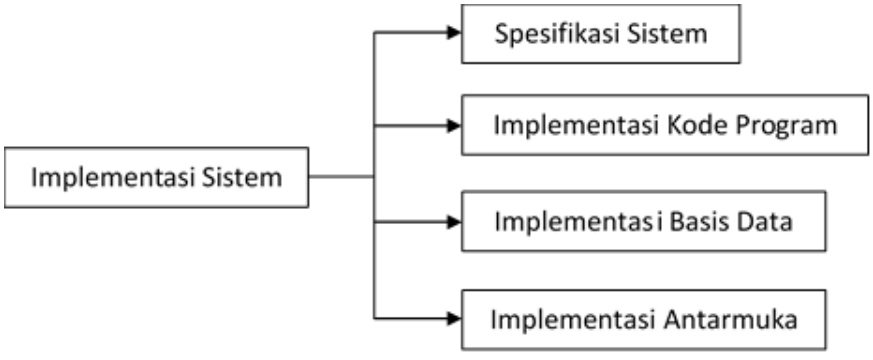
Tabel 5.9 Keterangan Gambar Perancangan Antarmuka *Owner Room Management*

No	Nama Objek	Tipe	Keterangan
1	Logo	Gambar	Menampilkan logo aplikasi dalam bentuk gambar
2	Welcome	*Text*	Menampilkan kata sambutan terhadap user yang telah *login* sesuai dengan nama akun user
3	*Logout*	*Button*	Tombol untuk melakukan fungsi *logout* yang akan mengeluarkan user yang telah *login* dan mengholangkan otoritasnya

4	Profile	*Dropdown*	Tombol untuk menampilkan pilihan menu milik *owner homestay* dalam bentuk dropdown
5	Room Number	*Text*	Menampilkan nomor kamar dalam bentuk text
6	*Check-in* dan *Check-out*	Tombol	Tombol untuk melakukan *check-in* terhadap kamar yang kosong dan *check-out* terhadap kamar yang terisi sesuai kamar yang dipilih
7	Guest Data	*Text*	Menampilkan nama tamu dan durasi hari tamu yang menginap pada kamar tersebut
8	*Check-out*	*Text*	Menampilkan tanggal *check-out* dari kamar tersebut

5.2 Implementasi Sistem

Implemetasi sistem akan dilakukan ketika proses perancangan sudah selesai. Implementasi sistem merupakan perpaduan dari tahap sebelumnya yaitu tahap analisis kebutuhan dan tahap perancangan sistem. Setiap kebutuhan-kebutuhan yang telah didefinisikan harus diimplementasikan kedalam sistem. Implementasi dari segi antarmuka juga harus sesuai dengan perancangan antarmuka pada tahap perancangan sistem. Struktur kelas pada *class diagram*, *pseudocode* algoritme, merupakan acuan dalam implementasi sistem. Implementasi sistem akan mendefinisikan dan menjelaskan spesifikasi dari sistem, implementasi kode program, implementasi data, dan implementasi antarmuka. Struktur dari sub-bab implementasi sistem akan ditunjukkan pada Gambar 5.13.



Gambar 5.13 Struktur Sub-bab Implementasi Sistem

Aplikasi Homie akan dihubungkan dengan internet dengan cara *hosting*. Tujuannya adalah agar aplikasi dapat diakses oleh pengguna dari manapun melalui internet. Kemudian juga bertujuan untuk memecahkan permasalahan pada latar belakang terkait promosi *homestay*. Domain yang digunakan dalam implementasi aplikasi ini adalah http://homie.4pps.web.id. Informasi terkait akun yang digunakan akan didefinisikan pada Tabel 5.10.

Tabel 5.10 Data Akun Aplikasi Reservasi dan Manajemen *Marketplace Homestay*

Email	Password	Keterangan
admin@admin.com	1234	Merupakan informasi data autentifikasi *admin*.
habib_yafi@ymail.com	1234	Merupakan informasi data autentifikasi *owner homestay*.

5.2.1 Spesifikasi Sistem

Pada sub bab ini akan dilakukan perincian terkait spesifikasi perangkat keras, perangkat lunak dan spesifikasi sistem operasi yang dipergunakan. Untuk spesifikasi perangkat keras akan dibagi menjadi enam komponen yaitu, *system model, processor, memory, storage, graphic card,* dan *display resolution*. Kemudian pada spesifikasi perangkat lunak akan dibagi menjadi enam komponen yaitu, editor dokumentasi, editor perancangan, editor pemrograman, bahasa pemrograman, *framework*, DBMS.

5.2.1.1 Spesifikasi Perangkat Keras

Spesifikasi perangkat keras yang digunakan dalam pengembangan perangkat lunak ini ditunjukkan pada Tabel 5.11.

Tabel 5.11 Spesifikasi Perangkat Keras

Nama Komponen	Spesifikasi
System Model	DELL 5459
Processor	i7-6500U CPU
Memory	12288MB RAM
Storage	1TB HDD

5.2.1.2 Spesifikasi Perangkat Lunak

Spesifikasi perangkat lunak yang digunakan dalam pengembangan perangkat lunak ini ditunjukkan pada Tabel 5.12.

Tabel 5.12 Spesifikasi Perangkat Lunak

Nama Komponen	Spesifikasi
Editor Dokumentasi	Microsoft Word 2016
Editor Perancangan	Visual Paradigm 15.2
Editor Pemrograman	Visual Studio Code 1.31.1
Bahasa Pemrograman	PHP 7.3.2, Javascript
Framework	Laravel 5.8.18, ReactJS 16.2
DBMS	MySQL 4.8.5

5.2.1.3 Spesifikasi Sistem Operasi

Spesifikasi sistem operasi yang digunakan dalam pengembangan perangkat lunak ini ditunjukkan pada Tabel 5.13.

Tabel 5.13 Spesifikasi Sistem Operasi

Nama Komponen	Spesifikasi
Operating System	Windows 10 64-bit

5.2.2 Implementasi Kode Program

Implementasi kode program mengacu kepada perancangan komponen yang telah dibuat pada bab perancangan. Algoritme-algoritme *pseudocode* yang telah didefinisikan pada bagian perancangan komponen akan diimplemetasikan kedalam bahasa pemrograman yang sesuai dengan bahasa pemrograman pada spesesifikasi perangkat lunak.

5.2.2.1 Implementasi Kode Program *Method* CreateOrder

Pada Tabel 5.14 dipaparkan kode program *method* Order yang berada pada *class* OrderController. Tujuan dari method ini adalah melakukan pemesanan kamar *homestay*. Algoritme ini akan melakukan inisialisasi data berdasarkan data yang didapat dari parameter method kemudian dialkukan validasi terhadap data yang dibawa, jika terdapat *field* kosong maka proses akan berakhir dan mengirimkan status sebagai *return value*. Jika berhasil maka akan dilanjutkan dengan validasi ketersedian kamar. Jika validasi ketersedian kamar berhasil maka akan dilanjutkan dengan menambahkan data pesanan tersebut ke database. Jika proses penyimpanan ke database berhasil maka proses pada method ini akan berakhir dan dilanjutkan dengan mengirimkan status berhasil sebagai *return value*.

Nama *class*: OrderController

Nama *method*: createOrder

Tabel 5.14 Implementasi Kode Program *Method* CreateOrder

Source code method createOrder

```
$this->validate($request, [
            'name'          => 'required',
            'guest'         => 'required',
            'email'         => 'required',
            'phone_number'  => 'required',
            'room_id'       => 'required',
        ]);
$checkinDate = date('Y-m-d', strtotime($request-
>checkin_date));
$checkoutDate = date('Y-m-d', strtotime($request-
>checkin_date. ' + '.$request->duration.' day'));
$roomPrice = Room::where('id', $request->room_id)-
>first(['price']);
$priceTotal = $roomPrice->price * $request->duration;

$order = Order::create(['name' => $request->name,
                        'guest' => $request->guest,
                        'email' => $request->email,
                        'phone_number' => $request-
>phone_number,
                        'room_id' => $request-
>room_id,
                        'room_number' => $request-
>room_number,
                        'homestay_id' => $request-
>homestay_id,
                        'checkin_date' =>
$checkinDate,
                        'duration' => $request-
>duration,
                        'checkout_date' =>
$checkoutDate,
                        'price_total' => $priceTotal,
                       ]);
for ($i=0; $i <= $request->duration; $i++) {
            $order_id = $order->id;
            $stay_date = date('Y-m-d',
strtotime($request->checkin_date. ' + '.$i.' day'));
            $orderMeta = OrderMeta::create([
                'order_id' => $order_id,
                'stay_date' => $stay_date,
                'status' => 'no status',
            ]);
}

return response()->json($order, 201);
```

5.2.2.2 Implementasi Kode Program *Method* Checkin

Pada Tabel 5.15 dipaparkan algoritme *method* CheckIn yang berada pada *class* ManagementController. Tujuan dari method ini adalah melakukan perubahan status kamar *homestay*. Algoritme ini akan melakukan inisialisasi data berdasarkan data yang didapat dari parameter method kemudian dilakukan validasi terhadap data yang dibawa, jika terdapat *field* kosong maka proses akan berakhir dan mengirimkan status sebagai *return value*. Jika berhasil maka akan dilanjutkan dengan verifikasi kode pembayaran. Jika kode pembyaran yang diinputkan tidak sesuai dengan kode pembayaran yang ada pada *database* Transaksi maka akan dilanjutkan pada proses *return value* berupa *status fail paymentCode*. Jika verifikasi kode pembayaran berhasil maka status kamar pada *database* akan diubah menjadi *available*. Setelah semua proses berhasil dijalankan, maka akan dilanjutakan dengan proses *return value* berupa success status sebagai informasi tambahan bahwa fungsi *check-in* berhasil.

Nama *class*: ManagementController

Nama *method*: Checkin

Tabel 5.15 Implementasi Kode Program *Method* Checkin

```
Source code method checkin
1   $this->validate($request,[
2               'guest_name'      => 'required',
3               // 'email'     => 'required',
4               // 'password'  => 'required|min:4',
5           ]);
6
7           $transactionStatus = Order::where('guest',
8   $request->guest_name)->where('id', $request-
9   >order_id)->get(['transaction_status']);
10          if ($transactionStatus[0]->transaction_status
11  == 'capture' ) {
12              $transactionStatus = Order::where('guest',
13  $request->guest_name)->where('id', $request-
14  >order_id)->update(['transaction_status' =>
15  'active']);
16              // $orderMetaUpdate =
17  OrderMeta::where('order_id', $id)->update(['status' =>
18  'used']);
19              $orderMetaUpdate = Order::where('guest',
20  $request->guest_name)->where('id', $request-
21  >order_id)->first();
22              $orderMetaUpdate->orderMeta()-
23  >update(['status' => 'active']);
24              return response()->json('Success', 200);
25          }
26          else {
27              return response()->json('transaction not
28  found', 404);
29          }
```

5.2.2.3 Implemetasi Kode Program *Method* Checkout

Pada Tabel 5.16 dipaparkan algoritme *method* CheckOut yang berada pada *class* ManagementController. Tujuan dari method ini adalah melakukan perubahan status kamar *homestay*, algoritme method ini hamper sama dengan method checkIn namun ada beberapa proses yang dikurangi. Algoritme ini akan melakukan inisialisasi data berdasarkan data yang didapat dari parameter method kemudian dilakukan validasi terhadap data yang dibawa, jika terdapat *field* kosong maka proses akan berakhir dan mengirimkan status sebagai *return value*. Jika berhasil maka akan dilanjutkan dengan perubahan status pada *database*. Jika data status yang dibawa memiliki nilai available, maka proses pengubahan status pada database akan dieksekusi. Pada tahap akhir method, proses *return value* berupa status sukses akan dieksekusi.

Nama *class*: ManagementController

Nama *method*: Checkout

Tabel 5.16 Implementasi Kode Program *Method* Checkout

```
Source code method checkout
1  $transactionStatus = Order::where('guest', $request-
2  >guest_name)->where('id',         $request->order_id)-
3  >get(['transaction_status']);
4          if  ($transactionStatus[0]->transaction_status
5  == 'active' ) {
6              $transactionStatus = Order::where('guest',
7  $request->guest_name)->where('id',           $request-
8  >order_id)->update(['transaction_status' => 'used']);
9              $orderMetaUpdate  =  Order::where('guest',
10 $request->guest_name)->where('id',           $request-
11 >order_id)->first();
12             $orderMetaUpdate->orderMeta()-
13 >update(['status' => 'used']);
14             return response()->json('Success', 200);
15         }
16         else {
17             return  response()->json('transaction  is
18 not active', 404);
           }
```

5.2.3 Implementasi Basis Data

Setelah tahap perancangan basis data dilakukan. Implementasi basis data akan dipaparkan dengan mengacu kepada perancangan data yang telah didefinisikan sebelumnya. Impelemntasi data pada aplikasi ini menghasilkan lima tabel. Query *Data Definition Language* (DDL) dari implementasi ke lima tabel tersebut akan dipaparkan pada sub bab ini. Query DDL basis data dapat dilihat pada Tabel 5.17 sampai 5.21.

5.2.3.1 Implemetasi Data Tabel Homestays

Implementasi data dari perancangan tabel homestays akan dipaparkan dalam bentuk query *Data Definition Language* (DDL). Query *Data Definition Language* (DDL) tabel homestays ditunjukkan pada Tabel 5.17.

Tabel 5.17 Implementasi Data Tabel Homestays

Implementasi Data Tabel Homestays	
1	CREATE TABLE `homestays` (
2	`id` int(11) UNSIGNED NOT NULL,
3	`user_id` int(11) UNSIGNED NOT NULL,
4	`name` varchar(191) COLLATE utf8mb4_unicode_ci NOT
5	NULL,
6	`location` varchar(191) COLLATE utf8mb4_unicode_ci
7	NOT NULL,
8	`address` text COLLATE utf8mb4_unicode_ci NOT NULL,
9	`facilities` varchar(191) COLLATE utf8mb4_unicode_ci
10	NOT NULL,
11	`number_of_rooms` int(11) NOT NULL,
12	`photo 1` varchar(125) COLLATE utf8mb4_unicode_ci
13	DEFAULT NULL,
14	`photo 2` varchar(125) COLLATE utf8mb4_unicode_ci
15	DEFAULT NULL,
16	`created_at` timestamp NULL DEFAULT NULL,
17	`updated_at` timestamp NULL DEFAULT NULL
18	) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4
19	COLLATE=utf8mb4_unicode_ci;

5.2.3.2 Implemetasi Data Tabel Rooms

Implementasi data dari perancangan tabel rooms akan dipaparkan dalam bentuk query *Data Definition Language* (DDL). Query *Data Definition Language* (DDL) tabel rooms ditunjukkan pada Tabel 5.18.

Tabel 5.18 Implemetasi Data Tabel Rooms

Implementasi Data Tabel Rooms	
1	CREATE TABLE `rooms` (
2	`id` int(11) UNSIGNED NOT NULL,
3	`homestay_id` int(11) NOT NULL,
4	`room_number` int(11) NOT NULL,
5	`type` varchar(191) COLLATE utf8mb4_unicode_ci NOT
6	NULL,
7	`description` text COLLATE utf8mb4_unicode_ci NOT
8	NULL,
9	`price` int(11) NOT NULL,
10	`room_availability` varchar(11) COLLATE
11	utf8mb4_unicode_ci NOT NULL,
12	`photos` varchar(191) COLLATE utf8mb4_unicode_ci NOT
13	NULL,
14	`created_at` timestamp NULL DEFAULT NULL,
15	`updated_at` timestamp NULL DEFAULT NULL
16	

```
17 ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4
18 COLLATE=utf8mb4_unicode_ci;
```

5.2.3.3 Implementasi Data Tabel Users

Implementasi data dari perancangan tabel users akan dipaparkan dalam bentuk query *Data Definition Language* (DDL). Query *Data Definition Language* (DDL) tabel users ditunjukkan pada Tabel 5.19.

Tabel 5.19 Implementasi Data Tabel Users

```
Implementasi Data Tabel Users
1  CREATE TABLE `users` (
2    `id` int(11) UNSIGNED NOT NULL,
3    `name` varchar(191) COLLATE utf8mb4_unicode_ci NOT
4  NULL,
5    `email` varchar(191) COLLATE utf8mb4_unicode_ci NOT
6  NULL,
7    `email_verified_at` timestamp NULL DEFAULT NULL,
8    `password` varchar(191) COLLATE utf8mb4_unicode_ci
9  NOT NULL,
10   `remember_token` varchar(100) COLLATE
11 utf8mb4_unicode_ci DEFAULT NULL,
12   `created_at` timestamp NULL DEFAULT NULL,
13   `updated_at` timestamp NULL DEFAULT NULL
14 ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4
15 COLLATE=utf8mb4_unicode_ci;
```

5.2.3.4 Implementsi Data Tabel Orders

Implementasi data dari perancangan tabel orders akan dipaparkan dalam bentuk query *Data Definition Language* (DDL). Query *Data Definition Language* (DDL) table orders ditunjukkan pada Tabel 5.20.

Tabel 5.20 Implementsi Data Tabel Orders

```
Implementasi Data Tabel Orders
1  CREATE TABLE `orders` (
2    `id` int(11) UNSIGNED NOT NULL,
3    `room_id` int(11) NOT NULL,
4    `homestay_id` int(11) NOT NULL,
5    `room_number` int(11) NOT NULL,
6    `name` varchar(191) COLLATE utf8mb4_unicode_ci NOT
7  NULL,
8    `guest` varchar(191) COLLATE utf8mb4_unicode_ci NOT
9  NULL,
10   `email` varchar(191) COLLATE utf8mb4_unicode_ci NOT
11 NULL,
12   `phone_number` varchar(191) COLLATE
13 utf8mb4_unicode_ci NOT NULL,
14   `checkin_date` date NOT NULL,
15   `duration` int(11) NOT NULL,
16   `checkout_date` date NOT NULL,
17   `price_total` int(11) NOT NULL,
```

```
18    `created_at` timestamp NULL DEFAULT NULL,
19    `updated_at` timestamp NULL DEFAULT NULL,
20    `transaction_status`       varchar(25)       COLLATE
21  utf8mb4_unicode_ci DEFAULT NULL,
22    `bill_key` int(25) DEFAULT NULL,
23    `biller_code` int(15) DEFAULT NULL,
24    `pdf_url`  varchar(255)  COLLATE  utf8mb4_unicode_ci
25  DEFAULT NULL,
26    `approval_code`          varchar(15)          COLLATE
27  utf8mb4_unicode_ci DEFAULT NULL,
28    `bank` varchar(11) COLLATE utf8mb4_unicode_ci DEFAULT
29  NULL,
30    `card_type`  varchar(11)  COLLATE  utf8mb4_unicode_ci
31  DEFAULT NULL,
32    `finish_redirect_url`      varchar(125)      COLLATE
33  utf8mb4_unicode_ci DEFAULT NULL,
34    `fraud_status`           varchar(11)          COLLATE
35  utf8mb4_unicode_ci DEFAULT NULL,
36    `gross_amount` int(11) DEFAULT NULL,
37    `masked_card` varchar(11) COLLATE utf8mb4_unicode_ci
38  DEFAULT NULL,
39    `payment_type`           varchar(25)          COLLATE
40  utf8mb4_unicode_ci DEFAULT NULL,
41    `bank_number` varchar(25) COLLATE utf8mb4_unicode_ci
42  DEFAULT NULL,
43    `instruction_url`         varchar(25)         COLLATE
44  utf8mb4_unicode_ci DEFAULT NULL,
45    `transaction_id`         varchar(125)         COLLATE
46  utf8mb4_unicode_ci DEFAULT NULL,
47    `transaction_time`        varchar(25)         COLLATE
48  utf8mb4_unicode_ci DEFAULT NULL,
49    `order_type` varchar(25) COLLATE utf8mb4_unicode_ci
50  DEFAULT NULL
51  )      ENGINE=InnoDB      DEFAULT      CHARSET=utf8mb4
52  COLLATE=utf8mb4_unicode_ci;
```

5.2.3.5 Implementasi Data Tabel Orders_Meta

Implementasi data dari perancangan tabel orders_meta akan dipaparkan dalam bentuk query *Data Definition Language* (DDL). Query *Data Definition Language* (DDL) table orders_meta ditunjukkan pada Tabel 5.21.

Tabel 5.21 Implementasi Data Tabel Orders_Meta

```
Implementasi Data Tabel Orders_Meta
1   CREATE TABLE `orders_meta` (
2     `id` int(11) UNSIGNED NOT NULL,
3     `order_id` int(11) UNSIGNED DEFAULT NULL,
4     `stay_date` date NOT NULL,
5     `status` varchar(191) COLLATE utf8mb4_unicode_ci NOT
6   NULL,
7     `created_at` timestamp NULL DEFAULT NULL,
8     `updated_at` timestamp NULL DEFAULT NULL
```

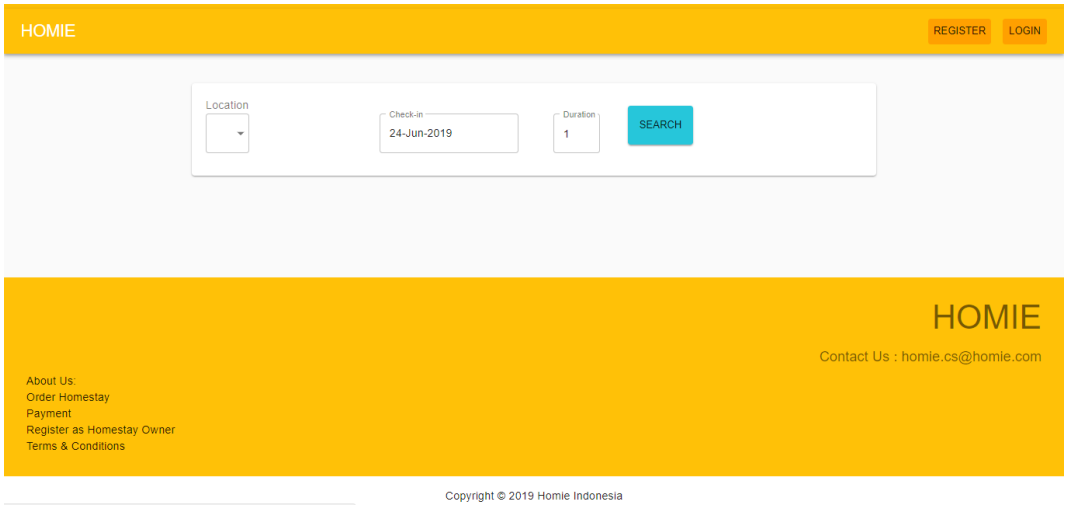
```
9  )       ENGINE=InnoDB       DEFAULT       CHARSET=utf8mb4
10 COLLATE=utf8mb4_unicode_ci;
```

5.2.4 Implemetasi Antarmuka

Pada tahap implemetasi antarmuka akan memaparkan hasil dari implementasi antarmuka. Hasil ini didasarkan pada perancangan antarmuka pada bab perancangan sebelumnya. Pada bagian ini, akan dipaparkan implementasi antarmuka homepage, *homestay* list, *homestay* detail, *order* detail, dan *owner room management*. Penjelasan lebih rinci akan dibahasa pada sub bab 5.2.4.1 sampai dengan sub bab 5.2.4.6.

5.2.4.1 Implementasi Antarmuka *Homepage*

Pada implementasi antarmuka *homepage* akan menampilkan 3 komponen utama yaitu header, body, dan footer. Pada bagian header terdapat logo aplikasi, tombol *login* dan register, dan tombol, profile jika *owner* sudah *login*. Pada bagian body terdapat beberapa komponen penyusun sesuai dengan konteks dari halaman ini yaitu homepage yang berisi *search bar*. *Search bar* ini memiliki *textfield* untuk lokasi, datepicker untuk tanggal check in, dan dropdown duration untuk lama menginap. Selanjutnya ada tombol *search* untuk menampilkan hasil pencarian. Pada bagian footer terdapat logo aplikasi, *email* untuk *costumer service*, dan about us. Hasil implementasi dari perancangan antarmuka *homepage* digambarkan pada Gambar 5.14.

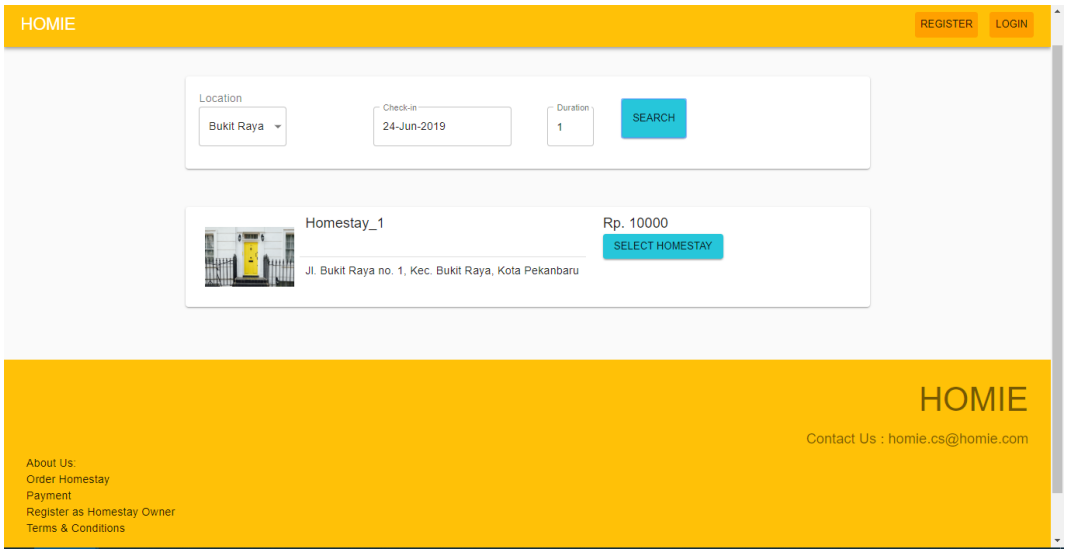


Gambar 5.14 Implementasi Antarmuka *Homepage*

5.2.4.2 Implementasi Antarmuka *Homestay List*

Pada implementasi antarmuka *homestay list* akan menampilkan 3 komponen utama yaitu header, body, dan footer. Pada bagian header terdapat logo aplikasi, tombol *login* dan register, dan tombol, profile jika *owner* sudah *login*. Pada bagian body terdapat beberapa komponen penyusun sesuai dengan konteks dari halaman ini yaitu home list yang berisi hasil pencarian. Search *bar* ini memiliki

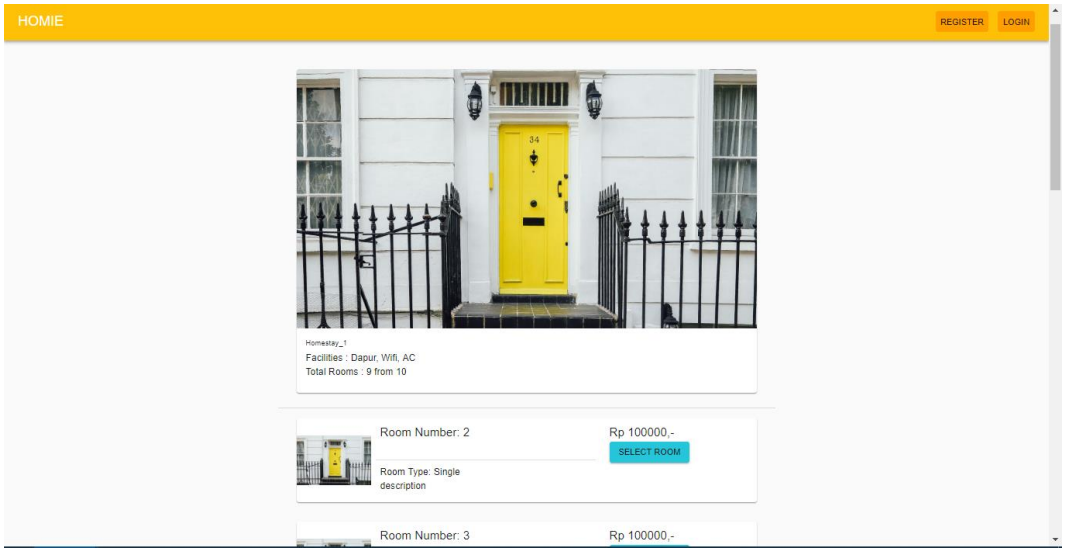
textfield untuk lokasi, datepicker untuk tanggal check in, dan dropdown duration untuk lama menginap. Selanjutnya ada tombol *search* untuk menampilkan hasil pencarian. Pada bagian footer terdapat logo aplikasi, *email* untuk *costumer service*, dan about us. Hasil implementasi dari perancangan antarmuka *homestay list* digambarkan pada Gambar 5.15.



Gambar 5.15 Implementasi Antarmuka *Homestay List*

5.2.4.3 Implemetasi Antarmuka *Homestay Detail*

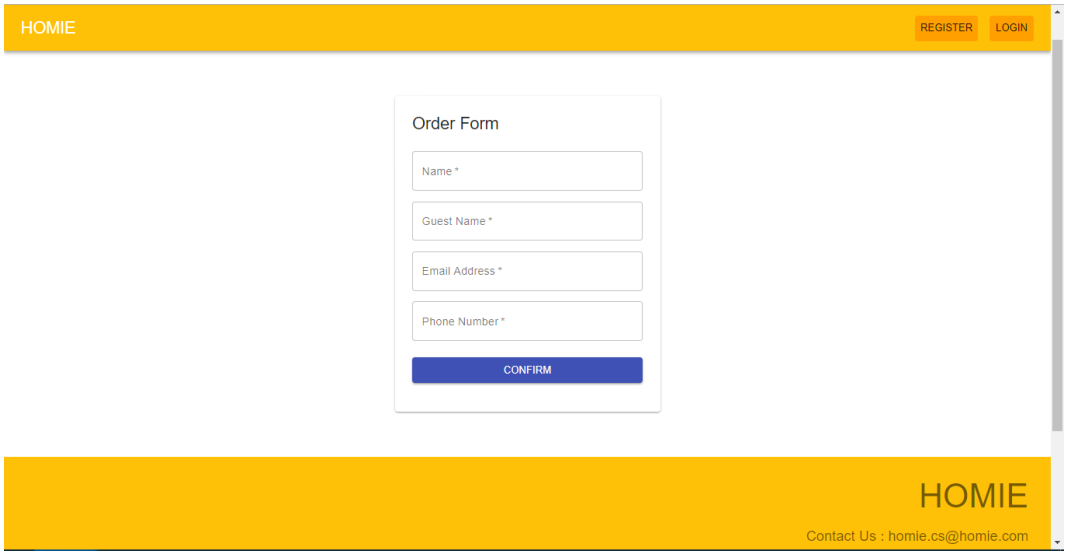
Pada implementasi antarmuka *homestay detail* akan menampilkan 3 komponen utama yaitu header, body, dan footer. Pada bagian header terdapat logo aplikasi, tombol *login* dan register, dan tombol, profile jika *owner* sudah *login*. Pada bagian body terdapat beberapa komponen penyusun sesuai dengan konteks dari halaman ini yaitu *homestay detail* yang berisi *search bar*. *Search bar* ini memiliki *textfield* untuk lokasi, datepicker untuk tanggal check in, dan dropdown duration untuk lama menginap. Selanjutnya ada tombol *search* untuk menampilkan hasil pencarian. Pada bagian footer terdapat logo aplikasi, *email* untuk *costumer service*, dan about us. Hasil implementasi dari perancangan antarmuka *homestay detail* digambarkan pada Gambar 5.16.



Gambar 5.16 Implementasi Antarmuka *Homestay Detail*

5.2.4.4 Implementasi Antarmuka *Order Form*

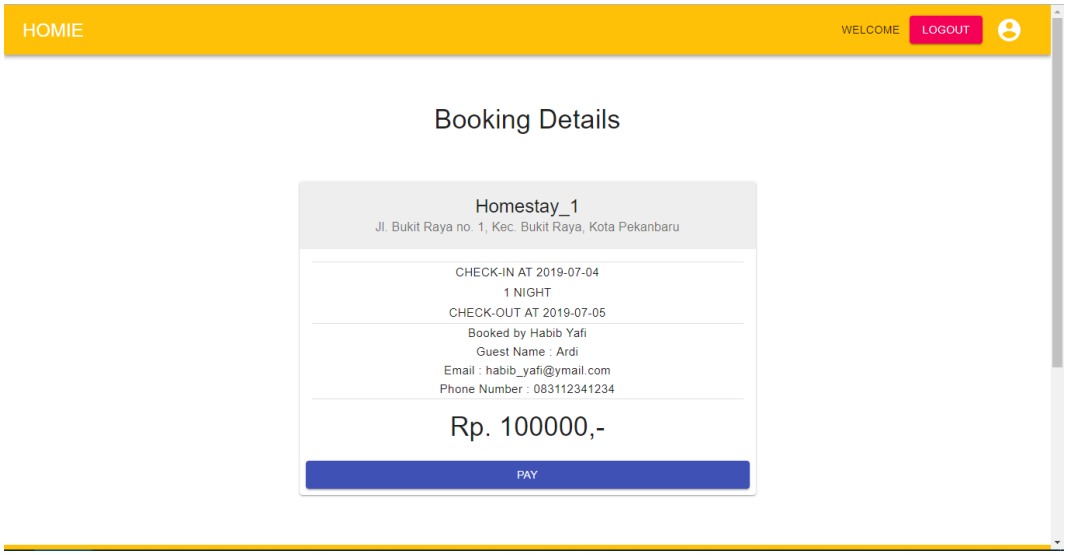
Pada implementasi antarmuka *order form* akan menampilkan 3 komponen utama yaitu header, body, dan footer. Pada bagian header terdapat logo aplikasi, tombol *login* dan register, dan tombol, profile jika *owner* sudah *login*. Pada bagian body terdapat beberapa komponen penyusun sesuai dengan konteks dari halaman ini yaitu *order form* yang berisi *search bar*. *Search bar* ini memiliki *textfield* untuk lokasi, datepicker untuk tanggal check in, dan dropdown duration untuk lama menginap. Selanjutnya ada tombol *search* untuk menampilkan hasil pencarian. Pada bagian footer terdapat logo aplikasi, *email* untuk *costumer service*, dan about us. Hasil implementasi dari perancangan antarmuka *order detail* digambarkan pada Gambar 5.17.



Gambar 5.17 Implementasi Antarmuka *Order Form*

5.2.4.5 Implementasi Antarmuka *Booking Details*

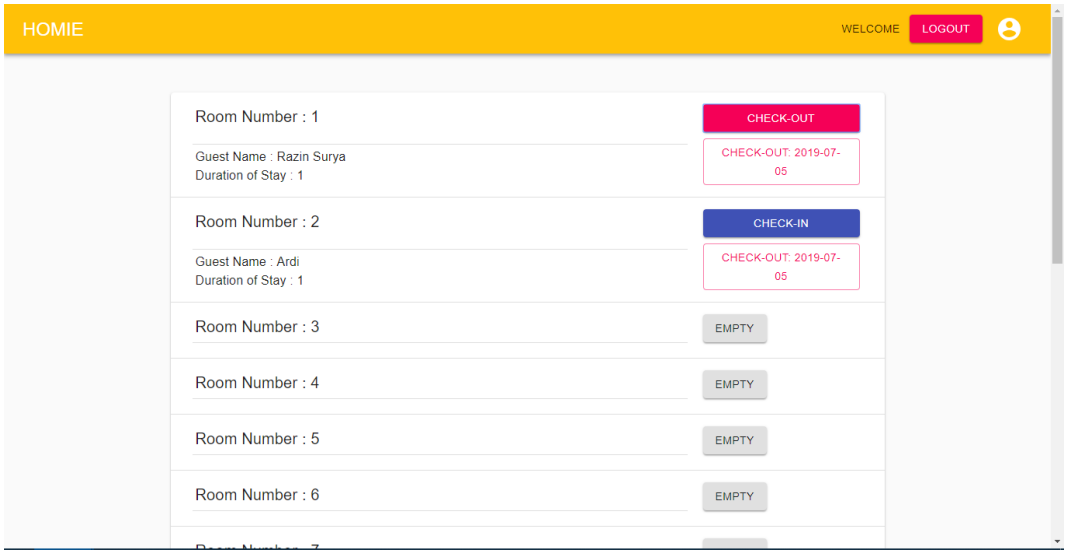
Pada implementasi antarmuka *booking details* akan menampilkan 3 komponen utama yaitu header, body, dan footer. Pada bagian header terdapat logo aplikasi, tombol *login* dan register, dan tombol, profile jika *owner* sudah *login*. Pada bagian body terdapat beberapa komponen penyusun sesuai dengan konteks dari halaman ini yaitu *booking details* yang berisi *search bar*. *Search bar* ini memiliki *textfield* untuk lokasi, datepicker untuk tanggal check in, dan dropdown duration untuk lama menginap. Selanjutnya ada tombol *search* untuk menampilkan hasil pencarian. Pada bagian footer terdapat logo aplikasi, *email* untuk *costumer service*, dan about us. Hasil implementasi dari perancangan antarmuka *order detail* digambarkan pada Gambar 5.18.



Gambar 5.18 Implementasi Antarmuka *Booking Details*

5.2.4.6 Implementasi Antarmuka *Owner Room Management*

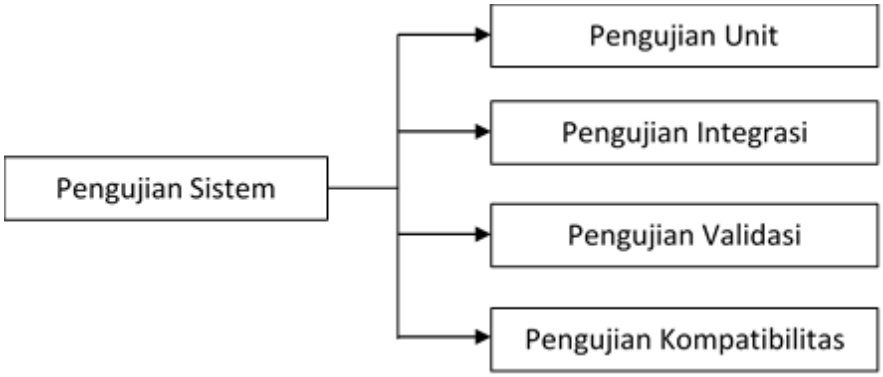
Pada implementasi antarmuka *owner room management* akan menampilkan 3 komponen utama yaitu header, body, dan footer. Pada bagian header terdapat logo aplikasi, tombol *login* dan register, dan tombol, profile jika *owner* sudah *login*. Pada bagian body terdapat beberapa komponen penyusun sesuai dengan konteks dari halaman ini yaitu *owner room management* yang berisi *search bar*. *Search bar* ini memiliki *textfield* untuk lokasi, datepicker untuk tanggal check in, dan dropdown duration untuk lama menginap. Selanjutnya ada tombol *search* untuk menampilkan hasil pencarian. Pada bagian footer terdapat logo aplikasi, *email* untuk *costumer service*, dan about us. Hasil implementasi dari perancangan antarmuka *owner room management* digambarkan pada Gambar 5.19.



Gambar 5.19 Implementasi Antarmuka *Owner Room Management*

BAB 6 PENGUJIAN SISTEM

Pada bagian pengujian sistem, akan dilakukan pengujian terhadap perangkat lunak yang telah diimplementasikan. Tujuannya adalah untuk memastikan bahwa hasil dari rekayasa kebutuhan, perancangan sistem, dan pengujian pada aplikasi Homie sesuai. Terdapat empat jenis pengujian yang akan dilakukan pada aplikasi Homie. Pengujian tersebut diantaranya pengujian unit, pengujian, integrasi, pengujian validasi, dan pengujian kompatibilitas. Struktur dari bab pengujian sistem akan ditunjukkan pada Gambar 6.1.



Gambar 6.1 Struktur Bab Pengujian Sistem

6.1 Pengujian Unit

Pada pengujian unit berfokus pada pelaksanaan uji untuk tiap-tiap unit ataupun komponen terkecil yang ada pada perangkat lunak. Tahap perancangan dan implemetasi sudah harus diselesaikan agar dapat melanjutkan kebagian pengujian. Metode yang digunakan adalah *whitebox testing*. Pada bagian ini akan diuji tiga *method* utama yaitu *method* createOrder, *method* checkin, dan *method* checkout. Dalam mengumpulkan *test case-test case* yang akan diuji maka *basis path testing* akan menjadi metode dalam pengumpulan *test case*. Setelah *test case* didapatkan pengujian akan dilakukan dengan bantuan phpUnit dimana tiap *test case* akan didefinisikan terlebih dahulu dalam sebuah fungsi pada *class driver*. Selanjutnya, eksekusi pengujian unit menggunakan phpUnit dilakukan dengan memasukkan perintah phpunit dan nama file pengujian sehingga phpUnit dapat melakukan *test* berdasarkan *test case* dan menampilkan hasil.

6.1.1 Pengujian Unit *Method* CreateOrder

1. *Pseudocode*

Nama *class*: OrderController

Nama *method*: createOrder

Pada pengujian unit *method* createOrder akan menggunakan *pseudocode* sebagai dasar dalam pembuatan *flowgraph*. *Pseudocode method* createOrder ditunjukkan pada Tabel 6.1.

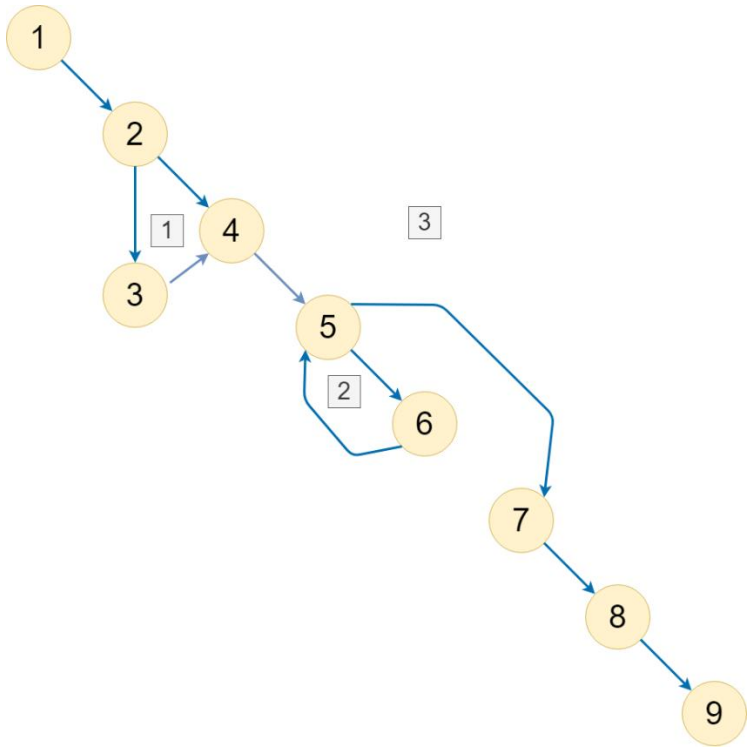
Tabel 6.1 *Pseudocode* Algoritme *Method* createOrder

Node	Pseudocode algoritme method createOrder
1	BEGIN
2	orderData = Request
2	
2	validate = [
2	orderData field name is required,
2	orderData field email is required,
2	orderData field phoneNumber is required,
2	orderData field rooms is required,
2	orderData field checkinDate is required,
2	orderData field duration is required,
2	]
2	
2	IF (validate is FALSE)
3	RETURN response to JSON (data with fail validate field and validate fail status)
4	END IF
5	
5	checkinDate = checkin_date of orderData formatted with (Y-M-D) date format
5	checkoutDate = checkin_date of orderData + duration day of orderData formatted with (Y-M-D) date format
5	roomPrice = get first price from database table room where id equals room_id of orderData
5	priceTotal = price of roomPrice * duration of orderData
5	
5	order = create data to database table Order with value (orderData all attribute, checkoutDate, priceTotal)
5	
6	FOR (from i = 0 until duration of orderData, increment by 1){
6	order_id = id of orderData
6	stay_date = checkin_date of orderData + i formatted with (Y-M-D) date format
6	orderMeta = create data to database table OrderMeta with value (order_id, stay_date, status = no status)
7	END FOR
8	
8	RETURN response to JSON (order and store success status 201)
9	END

2. *Basis Path Testing*

2.1 *Flow Graph*

Hasil dari penggambaran *flowgraph* berdasarkan *pseudocode* pada pengujian unit *method* createOrder ditunjukkan pada Gambar 6.2.



Gambar 6.2 *Flowgraph method* createOrder

2.2 *Cyclomatic Complexity*

- V(G) = jumlah region = 3
- V(G) = jumlah edge – jumlah node + 2 = 10 – 9 + 2 = 3
- V(G) = jumlah predicate node + 1 = 2 + 1 = 3

2.3 *Independent Path*

- Jalur 1: 1-2-4-5-7-8-9
- Jalur 2: 1-2-3-4-5-7-8-9
- Jalur 3: 1-2-4-5-6-5-7-8-9

Tabel 6.2 akan menampilkan kode program untuk driver yang digunakan dalam pengujian *method* createOrder.

Tabel 6.2 *Source Code Driver* OrderTest

OrderTest	
1	`class OrderTest extends TestCase`
2	`{`
3	`    public function testPath1()`
4	`    {`
5	`        $newOrder = [`
6	`            'name' => 'name test',`
7	`            'guest' => 'guest test',`

```
8               'email' => 'unitTest@gmail.com',
9               'phone_number' => '081312341234',
10              'room_id' => 19,
11              'room_number' => 1,
12              'homestay_id' => 77,
13              'checkin_date' => '2019-08-19',
14              'duration' => -1,
15          ];
16          $response = $this->json('POST', 'api/order',
17  $newOrder);
18          $response->assertStatus(500);
19          $this->assertTrue(true);
20      }
21
22      public function testPath2()
23      {
24          $newOrder = [
25              'name' => '',
26              'guest' => 'guest test',
27              'email' => 'unitTest@gmail.com',
28              'phone_number' => '081312341234',
29              'room_id' => '777',
30          ];
31          $response = $this->json('POST', 'api/order',
32  $newOrder);
33          $response->assertStatus(422);
34          $response->assertJson(['message' => 'The given
35  data was invalid.']);
36          $this->assertTrue(true);
37      }
38
39      public function testPath3()
40      {
41          $newOrder = [
42              'name' => 'name test',
43              'guest' => 'guest test',
44              'email' => 'unitTest@gmail.com',
45              'phone_number' => '081312341234',
46              'room_id' => 19,
47              'room_number' => 1,
48              'homestay_id' => 77,
49              'checkin_date' => '2019-08-19',
50              'duration' => 1,
51          ];
52          $response = $this->json('POST', 'api/order',
53  $newOrder);
54          $response->assertStatus(201);
55          $this->assertTrue(true);
56      }
57  }
```

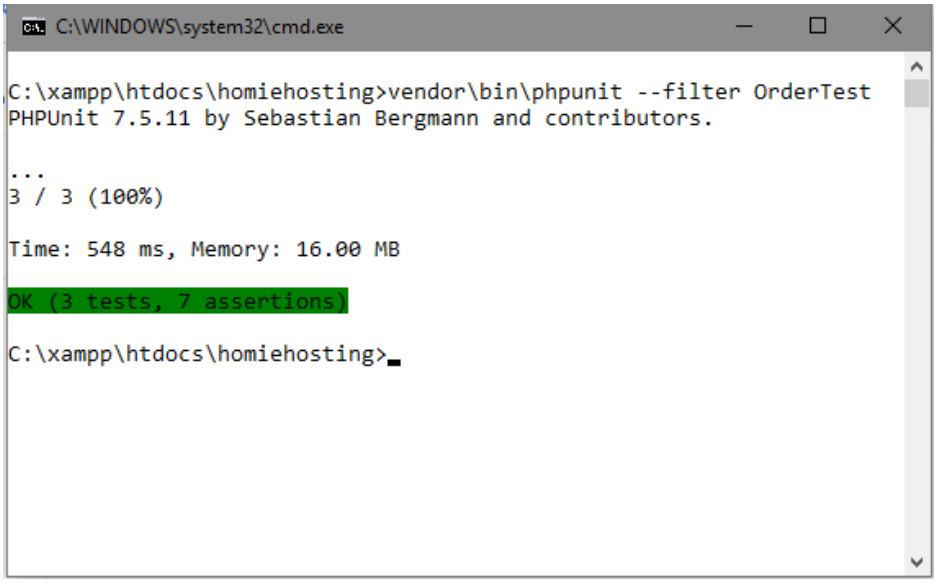
Hasil dari pengujian unit *method* createOrder akan dipaparkan lebih jelas pada Tabel 6.3.

Tabel 6.3 Hasil pengujian unit *method* createOrder

No.	Jalur	Prosedur Uji	Hasil yang Diharapkan	Hasil yang Didapatkan	Status
1	1-2-4-5-7-8-9	*Class driver* OrderTest method testPath1 dijalankan dengan memanggil *method* createOrder pada *class* OrderController.	Menjalankan *class* driver dan memanggil *method* createOrder kemudian perulangan tidak dilalui dan tidak terjadi penambahan data pada tabel order_meta di database	Menjalankan *class* driver dan memanggil *method* createOrder kemudian perulangan tidak dilalui dan tidak terjadi penambahan data pada tabel order_meta di database	Valid
2	1-2-3-4-5-7-8-9	*Class driver* OrderTest method testPath2 dijalankan dengan memanggil *method* createOrder pada *class* OrderController.	Menjalankan *class* driver dan memanggil *method* createOrder kemudian mengirim pesan error request failed dengan status 422 yang berisi pesan "The given data was invalid"	Menjalankan *class* driver dan memanggil *method* createOrder kemudian mengirim pesan error request failed dengan status 422 yang berisi pesan "The given data was invalid"	Valid
3	1-2-4-5-6-5-7-8-9	*Class driver* OrderTest method testPath3 dijalankan dengan memanggil *method* createOrder pada *class* OrderController.	Menjalankan *class* driver dan memanggil *method* createOrder kemudian perulangan dilalui dan terjadi penambahan data pada tabel order_meta di database sesuai dengan	Menjalankan *class* driver dan memanggil *method* createOrder kemudian perulangan dilalui dan terjadi penambahan data pada tabel order_meta di database sesuai dengan	Valid

			banyaknya nilai duration	banyaknya nilai duration	

Hasil eksekusi dari driver terhadap *method* checkin akan ditunjukkan pada Gambar 6.5. Pada gambar tersebut akan menampilkan status dari pengujian dan juga jumlah dari *test* yang dilakukan sesuai dengan *test case* yang telah didefinisikan sebelumnya.



Gambar 6.3 Hasil Pengujian Unit *Method* createOrder

6.1.2 Pengujian Unit *Method* Checkin

1. *Pseudocode*

Nama *class*: ManagementController

Nama *method*: checkin

Pada pengujian unit *method* checkin akan menggunakan *pseudocode* sebagai dasar dalam pembuatan *flowgraph*. *Pseudocode method* checkin ditunjukkan pada Tabel 6.4.

Tabel 6.4 *Pseudocode* Algoritme *Method* checkin

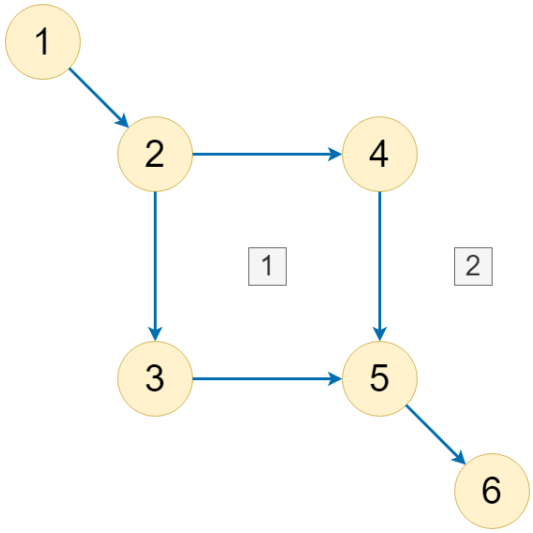
Pseudocode algoritme method checkin	
1	BEGIN
2	data = request
2	
2	transactionStatus = get transaction_status from
2	database table Order where guest equals guest_name of
2	data and where id equals order_id of data
2	
2	IF (transaction_status of index 0 transactionStatus is
3	capture) {
3	

```
3     transactionStatus = update data transaction_status
3  to active from database table Order where guest equals
3  guest_name of data and where id equals order_id of data
3     orderMetaUpdate = get first data from database table
3  Order where guest equals guest_name of data and where
3  id equals order_id of data
3     orderMetaUpdate update status to active of orderMeta
3     RETURN response to JSON (success with success status
4  code 200)
4  }
4
4  ELSE {
4     RETURN response to JSON (transaction not found with
5  not found status code 404)
6  }
   END
```

2. *Basis Path Testing*

2.1 *Flow Graph*

Hasil dari penggambaran *flowgraph* berdasarkan *pseudocode* pada pengujian unit *method* checkin ditunjukkan pada Gambar 6.4.



Gambar 6.4 *Flowgraph method* checkin

2.2 *Cyclomatic Complexity*

- V(G) = jumlah region = 2
- V(G) = jumlah edge – jumlah node + 2 = 6 – 6 + 2 = 2
- V(G) = jumlah predicate node + 1 = 1 + 1 = 2

2.3 *Independent Path*

- Jalur 1: 1-2-3-5-6
- Jalur 2: 1-2-4-5-6

Tabel 6.5 akan menampilkan kode program untuk driver yang digunakan dalam pengujian *method* checkin.

Tabel 6.5 *Source Code Driver* CheckinTest

CheckinTest

```
1   class CheckinTest extends TestCase
2   {
3       /**
4        * A basic unit test example.
5        *
6        * @return void
7        */
8       public function testPath1()
9       {
10          $checkin = [
11              'order_id' => 94,
12              'guest_name' => 'guest test'
13          ];
14          $response = $this->json('POST', 'api/checkin',
15  $checkin);
16          $response->assertStatus(404);
17          $this->assertTrue(true);
18      }
19
20      public function testPath2()
21      {
22          $checkin = [
23              'order_id' => 92,
24              'guest_name' => 'guest test'
25          ];
26          $response = $this->json('POST', 'api/checkin',
27  $checkin);
28          $response->assertStatus(200);
29          $this->assertTrue(true);
30      }
31
```

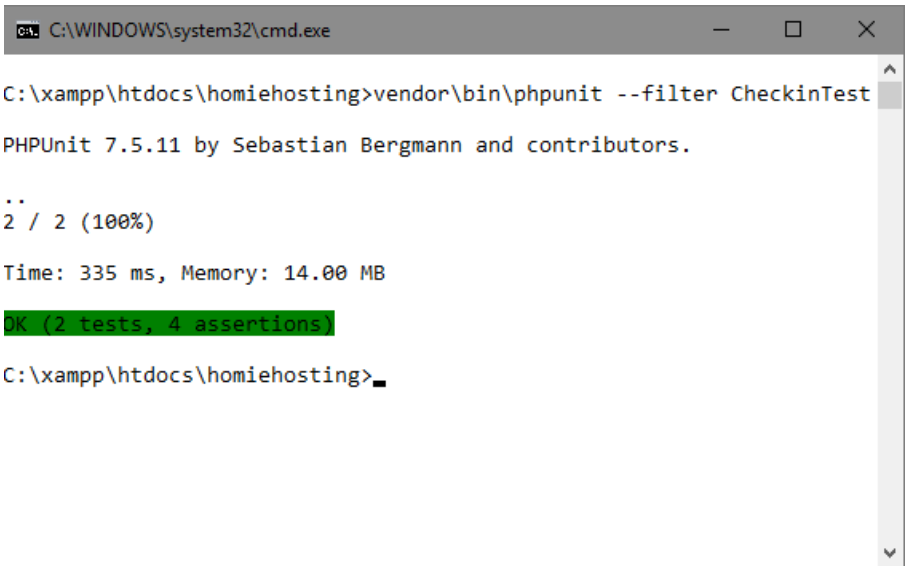
Hasil dari pengujian unit *method* checkin akan dipaparkan lebih jelas pada Tabel 6.6.

Tabel 6.6 Hasil pengujian unit *method* checkin

No.	Jalur	Prosedur Uji	Hasil yang Diharapkan	Hasil yang Didapatkan	Status
1	1-2-3-5-6	*Class driver* CheckinTest method	Menjalankan *class driver* dan memanggil	Menjalankan *class driver* dan	Valid

		testPath1 dijalankan dengan memanggil *method* checkin pada *class* Management Controller.	method checkin kemudian tidak melakukan perubahan terhadap tabel order maupun tabel order_meta pada database dan mengirimkan pesan error dengan kode status 404 yang berisi "transaction not found"	memanggil method checkin kemudian tidak melakukan perubahan terhadap tabel order maupun tabel order_meta pada database dan mengirimkan pesan error dengan kode status 404 yang berisi "transaction not found"	
2	1-2-4-5-6	*Class driver* CheckinTest method testPath2 dijalankan dengan memanggil *method* checkin pada *class* Management Controller.	Menjalankan *class driver* dan memanggil method checkin kemudian melakukan perubahan status terhadap tabel order dan tabel order_meta menjadi active dan mengirimkan pesan success dengan kode status 200 yang berisi "Success"	Menjalankan *class driver* dan memanggil method checkin kemudian melakukan perubahan status terhadap tabel order dan tabel order_meta menjadi active dan mengirimkan pesan success dengan kode status 200 yang berisi "Success"	Valid

Hasil eksekusi dari driver terhadap *method* checkin akan ditunjukkan pada Gambar 6.5. Pada gambar tersebut akan menampilkan status dari pengujian dan juga jumlah dari *test* yang dilakukan sesuai dengan *test case* yang telah didefinisikan sebelumnya.



Gambar 6.5 Hasil Pengujian Unit *Method* checkin

6.1.3 Pengujian Unit *Method* Checkout

1. *Pseudocode*

Nama *class*: ManagementController

Nama *method*: checkout

Pada pengujian unit *method* checkout akan menggunakan *pseudocode* sebagai dasar dalam pembuatan *flowgraph*. *Pseudocode method* checkout ditunjukkan pada Tabel 6.7.

Tabel 6.7 *Pseudocode* Algoritme *Method* checkout

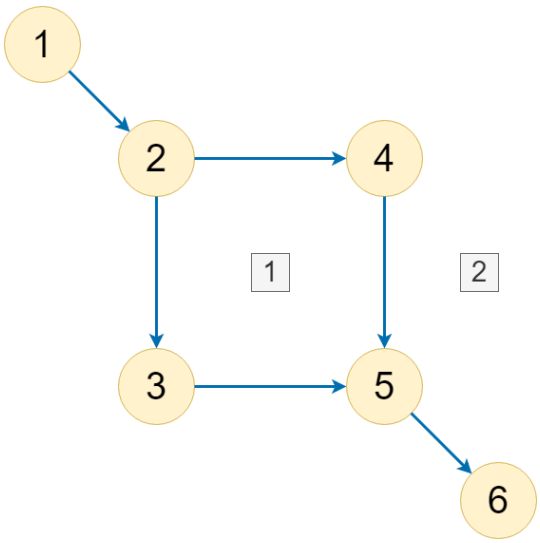
Pseudocode algoritme method checkout	
1	BEGIN
2	data = request
2	
2	transactionStatus = get transaction_status from database table Order where guest equals guest_name of data and where id equals order_id of data
2	
2	IF (transaction_status of index 0 transactionStatus is active) {
3	transactionStatus = update data transaction_status to used from database table Order where guest equals guest_name of data and where id equals order_id of data
3	orderMetaUpdate = get first data from database table Order where guest equals guest_name of data and where id equals order_id of data

```
3     orderMetaUpdate update status to used of orderMeta
3     RETURN response to JSON (success with success status
3  code 200)
4  }
4
4  ELSE {
4     RETURN response to JSON (transaction is not active
4  with not found status code 404)
5  }
6  END
```

2. *Basis Path Testing*

2.1 *Flow Graph*

Hasil dari penggambaran *flowgraph* berdasarkan *pseudocode* pada pengujian unit *method* checkout ditunjukkan pada Gambar 6.2.



Gambar 6.6 *Flowgraph method* checkout

2.2 *Cyclomatic Complexity*

- V(G) = jumlah region = 2
- V(G) = jumlah edge – jumlah node + 2 = 6 – 6 + 2 = 2
- V(G) = jumlah predicate node + 1 = 1 + 1 = 2

2.3 *Independent Path*

- Jalur 1: 1-2-3-5-6
- Jalur 2: 1-2-4-5-6

Tabel 6.8 akan menampilkan kode program untuk driver yang digunakan dalam pengujian *method* checkout.

Tabel 6.8 *Scource Code Driver* CheckoutTest

CheckoutTest

```
1   class CheckinTest extends TestCase
2   {
3       /**
4        * A basic unit test example.
5        *
6        * @return void
7        */
8       public function testPath1()
9       {
10          $checkin = [
11              'order_id' => 94,
12              'guest_name' => 'guest test'
13          ];
14          $response = $this->json('POST', 'api/checkin',
15  $checkin);
16          $response->assertStatus(404);
17          $this->assertTrue(true);
18      }
19
20      public function testPath2()
21      {
22          $checkin = [
23              'order_id' => 92,
24              'guest_name' => 'guest test'
25          ];
26          $response = $this->json('POST', 'api/checkin',
27  $checkin);
28          $response->assertStatus(200);
29          $this->assertTrue(true);
30      }
31
```

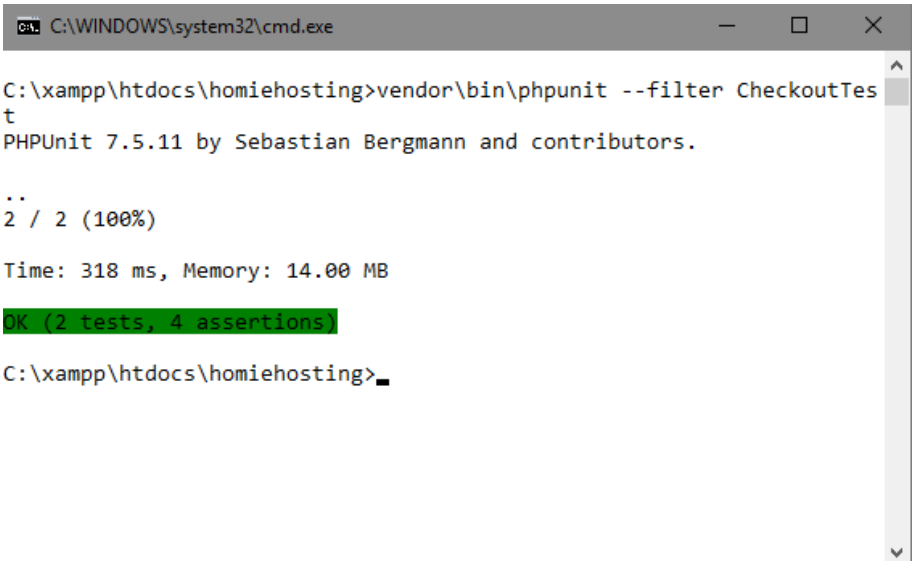
Hasil dari pengujian unit *method* checkout akan dipaparkan lebih jelas pada Tabel 6.9.

Tabel 6.9 Hasil pengujian unit *method* checkout

No.	Jalur	Prosedur Uji	Hasil yang Diharapkan	Hasil yang Didapatkan	Status
1	1-2-3-5-6	*Class driver* CheckoutTest method testPath1 dijalankan dengan memanggil *method* checkout	Menjalankan *class driver* dan memanggil method checkout kemudian tidak melakukan perubahan	Menjalankan *class driver* dan memanggil method checkout kemudian tidak	Valid

		pada *class* Management Controller.	terhadap tabel order maupun tabel order_meta pada database dan mengirimkan pesan error dengan kode status 404 yang berisi "transaction is not active"	melakukan perubahan terhadap tabel order maupun tabel order_meta pada database dan mengirimkan pesan error dengan kode status 404 yang berisi "transaction is not active"	
2	1-2-4-5-6	*Class driver* CheckoutTest method testPath2 dijalankan dengan memanggil *method* checkout pada *class* Management Controller.	Menjalankan *class driver* dan memanggil method checkout kemudian melakukan perubahan status terhadap tabel order dan tabel order_meta menjadi active dan mengirimkan pesan success dengan kode status 200 yang berisi "Success"	Menjalankan *class driver* dan memanggil method checkout kemudian melakukan perubahan status terhadap tabel order dan tabel order_meta menjadi active dan mengirimkan pesan success dengan kode status 200 yang berisi "Success"	Valid

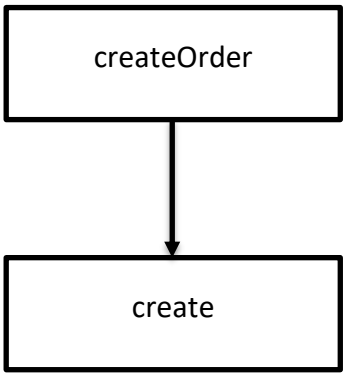
Hasil eksekusi dari driver terhadap *method* checkout akan ditunjukkan pada Gambar 6.7. Pada gambar tersebut akan menampilkan status dari pengujian dan juga jumlah dari *test* yang dilakukan sesuai dengan *test case* yang telah didefinisikan sebelumnya.



Gambar 6.7 Hasil Pengujian Unit *Method* checkout

6.2 Pengujian Integrasi

Pengujian ini difokuskan untuk konstruksi maupun desan dari sebuah aplikasi. Penerapan pada pengujian ini adalah melakukan suatu pekerjaan untuk mengintegrasikan fungsional dari satu atau lebih *class* ataupun *method* dalam melakukan suatu operasi. Pendekatan yang digunakan pada pengujian integrasi di sub bab ini adalah pendekatan top down. Pada Gambar 6.8 akan dipaparkan relasi antara *method* createOrder dan *method* create.



Gambar 6.8 Diagram Hierarki Pengujian Integrasi Method createOrder

Langkah uji yang akan dilakukan pada pengujian ini akan dijelaskan lebih spesifik pada Tabel 6.10.

Tabel 6.10 Langkah Uji Pengujian Integrasi

No.	Langkah Uji	Keterangan
1	createOrder() + create()	Method createOrder bertujuan melakukan penyimpanan percobaan yang di inputkan aktor. Method createOrder yang dideklarasikan dalam *class* controller

		dijalankan agar dapat melakukan pengujian dengan bantuan method stubCreate. Pemberian nilai dengan tipe string yang bertujuan untuk mengetahui create Order berhasil adalah “Order Success” dan kondisi create MetaOrder berhasil adalah “OrderMeta Success”.

Pada tahap awal pada pengujian integrasi antara method createOrder dan create menggunakan bantuan stub. Stub yang dipergunakan dalam pengujian ini adalah method stubCreate. Method stubCreate akan diberikan dua buah kondisi, antara lain kondisi create Order dengan masukan dengan tipe data string “Order Success” dan kondisi create OrderMeta dengan tipe data string “OrderMeta Success”. Kode program untuk method createOrder yang diintegrasikan dengan stubCreate dapat dilihat pada Tabel 6.11.

Tabel 6.11 *Source Code Method* createOrder

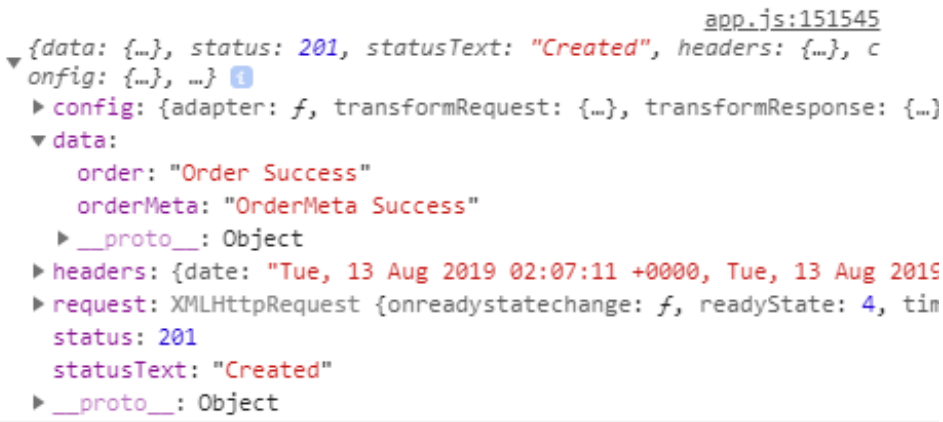
```
Source code
1   $this->validate($request, [
2               'name'          => 'required',
3               'guest'         => 'required',
4               'email'         => 'required',
5               'phone_number'  => 'required',
6               'room_id'       => 'required',
7           ]);
8           $checkinDate          =          date('Y-m-d',
9   strtotime($request->checkin_date));
10          $checkoutDate         =          date('Y-m-d',
11  strtotime($request->checkin_date.   '   +   '.$request-
12  >duration.' day'));
13          $roomPrice   =   Room::where('id',   $request-
14  >room_id)->first(['price']);
15          $priceTotal  =  $roomPrice->price  *  $request-
16  >duration;
17
18          $order = $this->stubCreate('Order Success');
19
20          for ($i=0; $i <= $request->duration; $i++) {
21              $order_id = $order->id;
22              $stay_date          =          date('Y-m-d',
23  strtotime($request->checkin_date. ' + '.$i.' day'));
24              $orderMeta  =  $this->stubCreate('OrderMeta
25  Success');
26
27          }
28          return       response()->json(compact('order',
29  'orderMeta'), 201);
```

Kode program *method* stubCreate yang dipergunakan pada pengujian integrasi, dipaparkan pada Tabel 6.12.

Tabel 6.12 *Source Code Method* stubCreate

Source code	
1	`public function stubCreate($createOrder) {`
2	`    return $createOrder;`
3	
4	`}`

Hasil uji didapatkan dengan menjalankan method createOrder. Pada method createOrder akan diuji integrase dari method tersebut dengan stub yang telah dibuat yaitu stubCreate. Ketika stubCreate dipanggil maka akan akan megirimkan pesan "Order Success" dan "OrderMeta Success". Hal tersebut menandakan bahwa method createOrder berhasil diintegrasikan dengan stubCreate. Hasil pengujian dibuktikan dengan Gambar 6.9.



Gambar 6.9 Hasil Pengujian *Method* createOrder Menggunakan stubCreate

6.3 Pengujian Validasi

Pada pengujian ini akan menggonakan model atau acuan berdasarkan skenario dan fungsional- fungsional sistem yang telah dibuat. Tujuan dari pengujian ini yaitu untuk memastikan kebutuhan fungsional beserta skenarionya selaras dengan hasil implementasi. Jika semua kasus uji telah memenuhi harapan yang ada pada pendefinisian kebutuhan dan pendefinisian skenario maka dapat dinyatakan bahwa kasus uji tersebut valid. Pengujian validasi terhadap aplikasi Homie akan diapaparkan pada Tabel 6.13 sampai dengan Tabel 6.45.

Hasil pengujian validasi untuk *use case* mencari *homestay* menghasilkan nilai valid. Nilai valid tersebut didapatkan ketika hasil yang didapatkan sesuai dengan hasil yang diharapkan. Penjelasan lebih rinci terkait pengujian validasi pada *use case* ini ditunjukkan pada Tabel 6.13.

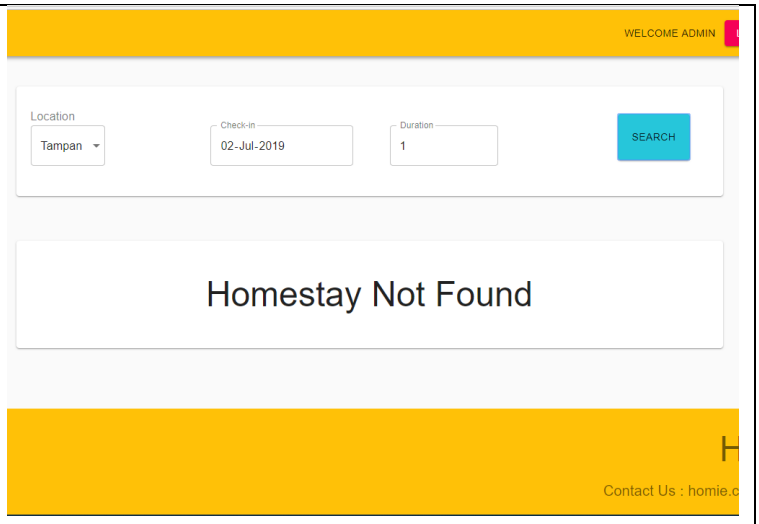
Tabel 6.13 Pengujian Validasi *Use Case* Mencari *Homestay*

Kode Kebutuhan	HOMIE-F-01
Nama Kasus Uji	Mencari *Homestay*
Prosedur	1. Memasuki halaman utama sistem. 2. Mengisi *form* pencarian berdasarkan lokasi yang dinginkan, tanggal checkin, dan durasi lamanya menginap. 3. Memilih opsi cari (hasil ditemukan).
Hasil yang Diharapkan	Sistem menampilkan daftar *homestay* yang terdaftar pada sistem sesuai dengan parameter pencarian yang telah ditentukan.
Hasil yang Didapatkan	Sistem menampilkan daftar *homestay* yang terdaftar pada sistem sesuai dengan parameter pencarian yang telah ditentukan.
Status	Valid

Hasil pengujian validasi untuk *use case* mencari *homestay* alternatif 1 menghasilkan nilai valid. Nilai valid tersebut didapatkan ketika hasil yang didapatkan sesuai dengan hasil yang diharapkan. Penjelasan lebih rinci terkait pengujian validasi pada *use case* ini ditunjukkan pada Tabel 6.14.

Tabel 6.14 Pengujian Validasi *Use Case* Mencari *Homestay* Alternatif 1

Kode Kebutuhan	HOMIE-F-01
Nama Kasus Uji	Mencari *Homestay*
Prosedur	1. Memasuki halaman utama sistem. 2. Mengisi *form* pencarian berdasarkan lokasi yang dinginkan, tanggal checkin, dan durasi lamanya menginap. 3. Memilih opsi cari (hasil tidak ditemukan).
Hasil yang Diharapkan	Sistem menampilkan pesan “Homestay Not Found”.
Hasil yang Didapatkan	Sistem menampilkan pesan “Homestay Not Found”.

	
Status	Valid

Hasil pengujian validasi untuk *use case* melihat detail *homestay* menghasilkan nilai valid. Nilai valid tersebut didapatkan ketika hasil yang didapatkan sesuai dengan hasil yang diharapkan. Penjelasan lebih rinci terkait pengujian validasi pada *use case* ini ditunjukkan pada Tabel 6.15.

Tabel 6.15 Pengujian Validasi *Use Case* Melihat Detail *Homestay*

Kode Kebutuhan	HOMIE-F-02
Nama Kasus Uji	Melihat Detail *Homestay*
Prosedur	1. Memasuki halaman detail *homestay*.
Hasil yang Diharapkan	Sistem menampilkan daftar *homestay* yang terdaftar pada sistem berdasarkan parameter pencarian yang telah ditentukan.
Hasil yang Didapatkan	Sistem menampilkan daftar *homestay* yang terdaftar pada sistem berdasarkan parameter pencarian yang telah ditentukan.
Status	Valid

Hasil pengujian validasi untuk *use case* melakukan pemesanan kamar menghasilkan nilai valid. Nilai valid tersebut didapatkan ketika hasil yang didapatkan sesuai dengan hasil yang diharapkan. Penjelasan lebih rinci terkait pengujian validasi pada *use case* ini ditunjukkan pada Tabel 6.16.

Tabel 6.16 Pengujian Validasi *Use Case* Melakukan Pemesanan Kamar

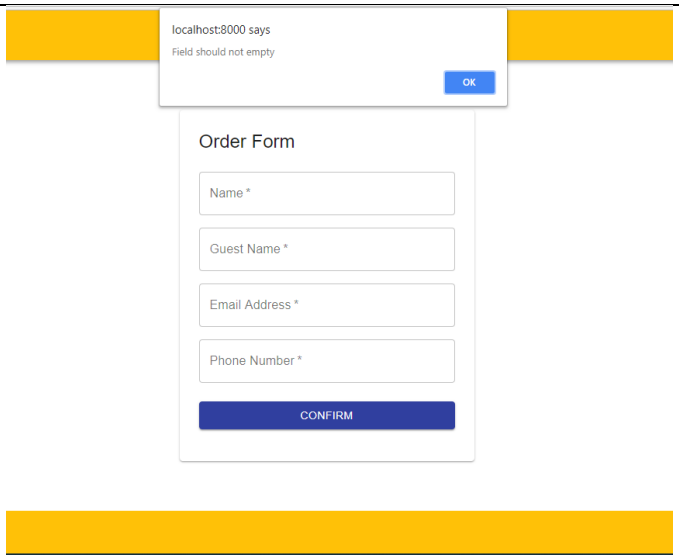
Kode Kebutuhan	HOMIE-F-03
Nama Kasus Uji	Melakukan Pemesanan Kamar

Prosedur	1. Memasuki halaman pemesanan kamar. 2. Mengisi *field-field* yang ada dalam *form* pemesanan, Nama = "Habib", Nama Tamu = "Habib", Email = "habibyafi45@gmail.com, dan nomor HP = "081312341234". 3. Memilih opsi pesan.
Hasil yang Diharapkan	Sistem menyimpan informasi pemesanan kamar berdasarkan nilai yang telah dimasukkan kedalam *field-field* yang ada didalam *form* pemesanan kamar dan kemudian menampilkan halaman detail pemesanan.
Hasil yang Didapatkan	Sistem menyimpan informasi pemesanan kamar berdasarkan nilai yang telah dimasukkan kedalam *field-field* yang ada didalam *form* pemesanan kamar dan kemudian menampilkan halaman detail pemesanan.
Status	Valid

Hasil pengujian validasi untuk *use case* melakukan pemesanan kamar alternatif 1 menghasilkan nilai valid. Nilai valid tersebut didapatkan ketika hasil yang didapatkan sesuai dengan hasil yang diharapkan. Penjelasan lebih rinci terkait pengujian validasi pada *use case* ini ditunjukkan pada Tabel 6.17.

Tabel 6.17 Pengujian Validasi *Use Case* Melakukan Pemesanan Kamar Alternatif 1

Kode Kebutuhan	HOMIE-F-03
Nama Kasus Uji	Melakukan Pemesanan Kamar
Prosedur	1. Memasuki halaman pemesanan kamar. 2. Tidak mengisikan *field-field* yang ada didalam *form* pemesanan. 3. Memilih opsi pesan
Hasil yang Diharapkan	Sistem menampilkan pesan informasi peringatan bahwa *field* tidak boleh kosong.
Hasil yang Didapatkan	Sistem menampilkan pesan informasi peringatan bahwa *field* tidak boleh kosong.

	
Status	Valid

Hasil pengujian validasi untuk *use case* melihat detail pemesanan menghasilkan nilai valid. Nilai valid tersebut didapatkan ketika hasil yang didapatkan sesuai dengan hasil yang diharapkan. Penjelasan lebih rinci terkait pengujian validasi pada *use case* ini ditunjukkan pada Tabel 6.18.

Tabel 6.18 Pengujian Validasi *Use Case* Melihat Detail Pemesanan

Kode Kebutuhan	HOMIE-F-04
Nama Kasus Uji	Melihat Detail Pemesanan
Prosedur	1. Memasuki halaman pemesanan kamar. 2. Mengisi *field-field* yang ada dalam *form* pemesanan kamar. 3. Menekan tombol pesan.
Hasil yang Diharapkan	Sistem menampilkan detail pemesanan kamar berdasarkan pilihan-pilihan yang sudah dipilih sebelumnya.
Hasil yang Didapatkan	Sistem menampilkan detail pemesanan kamar berdasarkan pilihan-pilihan yang sudah dipilih sebelumnya.
Status	Valid

Hasil pengujian validasi untuk *use case* melakukan pembayaran menghasilkan nilai valid. Nilai valid tersebut didapatkan ketika hasil yang didapatkan sesuai dengan hasil yang diharapkan. Penjelasan lebih rinci terkait pengujian validasi pada *use case* ini ditunjukkan pada Tabel 6.19.

Tabel 6.19 Pengujian Validasi *Use Case* Melakukan Pembayaran

Kode Kebutuhan	HOMIE-F-05
Nama Kasus Uji	Melakukan Pembayaran
Prosedur	1. Memasuki halaman detail pesanan. 2. Memilih opsi bayar 3. Memilih metode pembayaran yang ada pada *window payment gateway.* 4. Mengisikan informasi yang dibutuhkan oleh *window payment gateway.* 5. Mengikuti prosedur pembayaran yang diinformasikan lewat notifikasi email kepada pemesan oleh *payment gateway.*
Hasil yang Diharapkan	Sistem menyimpan informasi yang didapat kedalam sistem pada saat pembayaran menggunakan *payment gateway*.
Hasil yang Didapatkan	Sistem menyimpan informasi yang didapat kedalam sistem pada saat pembayaran menggunakan *payment gateway*.
Status	Valid

Hasil pengujian validasi untuk *use case login* menghasilkan nilai valid. Nilai valid tersebut didapatkan ketika hasil yang didapatkan sesuai dengan hasil yang diharapkan. Penjelasan lebih rinci terkait pengujian validasi pada *use case* ini ditunjukkan pada Tabel 6.20.

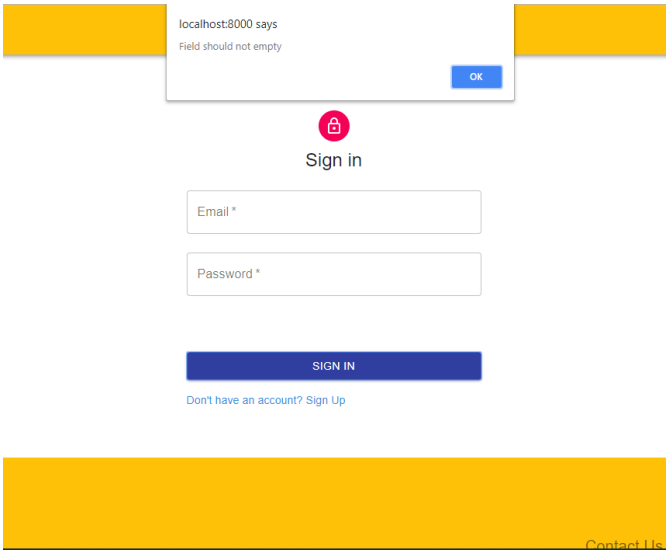
Tabel 6.20 Pengujian Validasi *Use Case Login*

Kode Kebutuhan	HOMIE-F-06
Nama Kasus Uji	*Login*
Prosedur	1. Memasuki halaman *login*. 2. Mengisikan *field-field* yang ada didalam *form login* dengan email dan password yang benar, email = “owner1@owner.com” dan password = “1234”. 3. Menekan tombol *login*.
Hasil yang Diharapkan	Sistem menampilkan halaman utama sistem.

Hasil yang Didapatkan	Sistem menampilkan halaman utama sistem.
Status	Valid

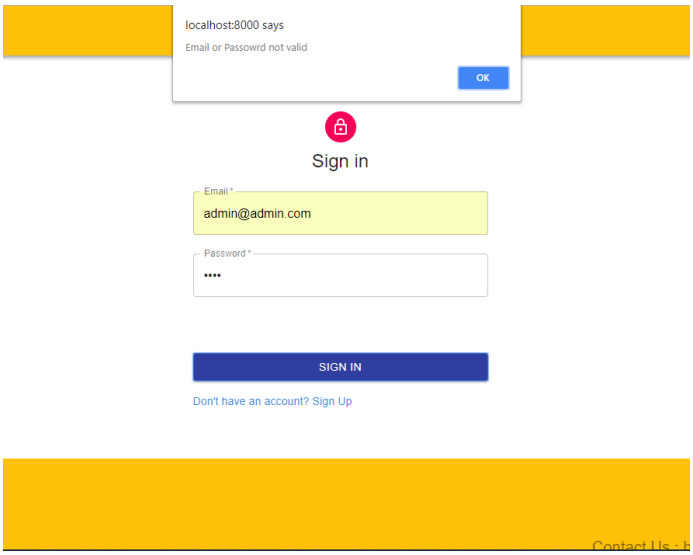
Hasil pengujian validasi untuk *use case login* alternatif 1 menghasilkan nilai valid. Nilai valid tersebut didapatkan ketika hasil yang didapatkan sesuai dengan hasil yang diharapkan. Penjelasan lebih rinci terkait pengujian validasi pada *use case* ini ditunjukkan pada Tabel 6.21.

Tabel 6.21 Pengujian Validasi *Use Case Login* Alternatif 1

Kode Kebutuhan	HOMIE-F-06
Nama Kasus Uji	*Login*
Prosedur	1. Memasuki halaman *login*. 2. Tidak mengisikan *field-field* yang ada didalam *form login*. 3. Menekan tombol *login*.
Hasil yang Diharapkan	Sistem menampilkan pesan informasi peringatan bahwa *field* tidak boleh kosong. 
Hasil yang Didapatkan	Sistem menampilkan pesan informasi peringatan bahwa *field* tidak boleh kosong
Status	Valid

Hasil pengujian validasi untuk *use case login* alternatif 2 menghasilkan nilai valid. Nilai valid tersebut didapatkan ketika hasil yang didapatkan sesuai dengan hasil yang diharapkan. Penjelasan lebih rinci terkait pengujian validasi pada *use case* ini ditunjukkan pada Tabel 6.22.

Tabel 6.22 Pengujian Validasi *Use Case Login* Alternatif 2

Kode Kebutuhan	HOMIE-F-06
Nama Kasus Uji	*Login*
Prosedur	1. Memasuki halaman *login*. 2. Mengisikan *field-field* yang ada didalam *form login* dengan email dan password yang salah, email = “salah@salah.com” dan password = “salah”. 3. Menekan tombol *login*.
Hasil yang Diharapkan	Sistem menampilkan pesan peringatan bahwa *email* atau *password* yang diinputkan salah.
Hasil yang Didapatkan	Sistem menampilkan pesan peringatan bahwa *email* atau *password* yang diinputkan salah. 
Status	Valid

Hasil pengujian validasi untuk *use case registrasi* menghasilkan nilai valid. Nilai valid tersebut didapatkan ketika hasil yang didapatkan sesuai dengan hasil yang diharapkan. Penjelasan lebih rinci terkait pengujian validasi pada *use case* ini ditunjukkan pada Tabel 6.23.

Tabel 6.23 Pengujian Validasi *Use Case* Registrasi

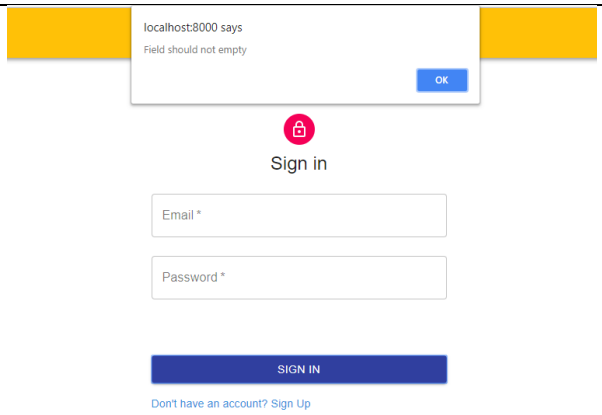
Kode Kebutuhan	HOMIE-F-07
Nama Kasus Uji	Registrasi
Prosedur	1. Memasuki halaman registrasi.

	2. Mengisikan *field-field* yang ada didalam *form* registrasi. 3. Menekan tombol registrasi.
Hasil yang Diharapkan	Sistem mendaftarkan akun baru kedalam sistem sesuai dengan nilai yang diinputkan pada *field-field* yang ada didalam *form* pendaftaran
Hasil yang Didapatkan	Sistem mendaftarkan akun baru kedalam sistem sesuai dengan nilai yang diinputkan pada *field-field* yang ada didalam *form* pendaftaran
Status	Valid

Hasil pengujian validasi untuk *use case* registrasi alternatif 1 menghasilkan nilai valid. Nilai valid tersebut didapatkan ketika hasil yang didapatkan sesuai dengan hasil yang diharapkan. Penjelasan lebih rinci terkait pengujian validasi pada *use case* ini ditunjukkan pada Tabel 6.24.

Tabel 6.24 Pengujian Validasi *Use Case* Registrasi Alternatif 1

Kode Kebutuhan	HOMIE-F-07
Nama Kasus Uji	Registrasi
Prosedur	1. Memasuki halaman registrasi. 2. Tidak mengisikan *field-field* yang ada didalam *form* registrasi. 3. Menekan tombol registrasi.
Hasil yang Diharapkan	Sistem menampilkan pesan informasi peringatan bahwa *field* tidak boleh kosong.
Hasil yang Didapatkan	Sistem menampilkan pesan informasi peringatan bahwa *field* tidak boleh kosong.

	
Status	Valid

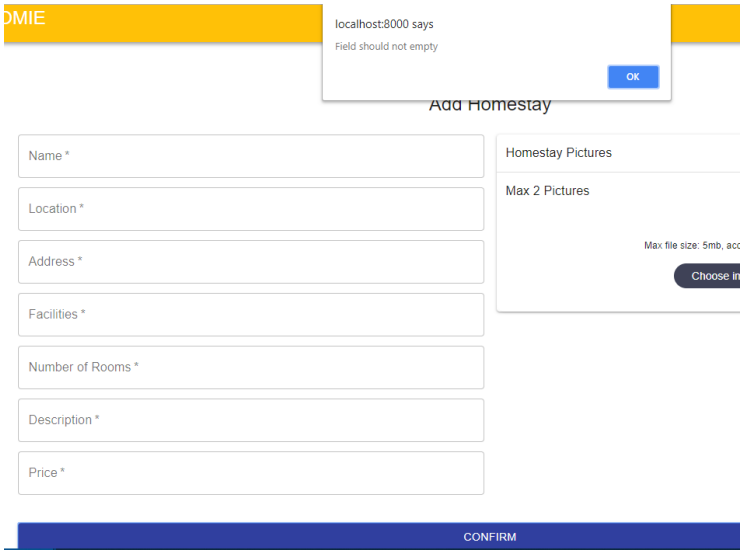
Hasil pengujian validasi untuk *use case* mendaftarkan *homestay* menghasilkan nilai valid. Nilai valid tersebut didapatkan ketika hasil yang didapatkan sesuai dengan hasil yang diharapkan. Penjelasan lebih rinci terkait pengujian validasi pada *use case* ini ditunjukkan pada Tabel 6.25.

Tabel 6.25 Pengujian Validasi *Use Case* Mendaftarkan *Homestay*

Kode Kebutuhan	HOMIE-F-08
Nama Kasus Uji	Mendaftarkan *Homestay*
Prosedur	1. Memasuki halaman *owner* mendaftarkan *homestay*. 2. Mengisi *field-field* yang ada pada *form* pendaftaran *homestay*. 3. Menekan tombol konfirmasi.
Hasil yang Diharapkan	Sistem mendaftarkan *homestay* dengan informasi-informaasi yang telah diberikan kedalam *field-field form* pendaftaran *homestay*.
Hasil yang Didapatkan	Sistem mendaftarkan *homestay* dengan informasi-informasi yang telah diberikan kedalam *field-field form* pendaftaran *homestay*.
Status	Valid

Hasil pengujian validasi untuk *use case* mendaftarkan *homestay* alternatif 1 menghasilkan nilai valid. Nilai valid tersebut didapatkan ketika hasil yang didapatkan sesuai dengan hasil yang diharapkan. Penjelasan lebih rinci terkait pengujian validasi pada *use case* ini ditunjukkan pada Tabel 6.26.

Tabel 6.26 Pengujian Validasi *Use Case* Mendaftarkan *Homestay* Alternatif 1

Kode Kebutuhan	HOMIE-F-08
Nama Kasus Uji	Mendaftarkan *Homestay*
Prosedur	1. Memasuki halaman *owner* mendaftarkan *homestay*. 2. Tidak Mengisi *field-field* yang ada pada *form* pendaftaran *homestay*. 3. Menekan tombol konfirmasi.
Hasil yang Diharapkan	Sistem menampilkan pesan informasi peringatan bahwa *field* tidak boleh kosong.
Hasil yang Didapatkan	Sistem menampilkan pesan informasi peringatan bahwa *field* tidak boleh kosong. 
Status	Valid

Hasil pengujian validasi untuk *use case* melihat *homestay owner* menghasilkan nilai valid. Nilai valid tersebut didapatkan ketika hasil yang didapatkan sesuai dengan hasil yang diharapkan. Penjelasan lebih rinci terkait pengujian validasi pada *use case* ini ditunjukkan pada Tabel 6.27.

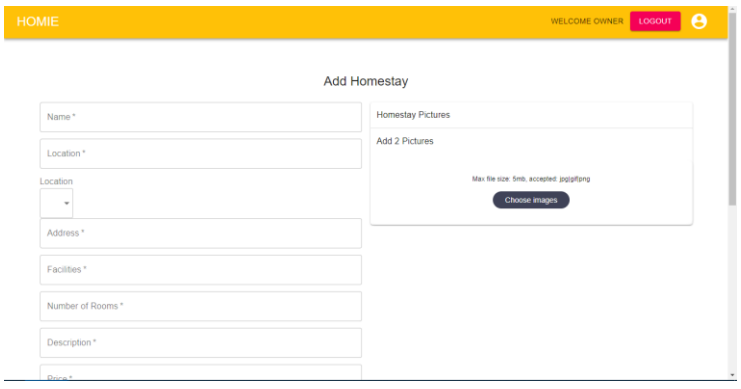
Tabel 6.27 Pengujian Validasi *Use Case* Melihat *Homestay Owner*

Kode Kebutuhan	HOMIE-F-09
Nama Kasus Uji	Melihat *Homestay Owner*
Prosedur	1. Memasuki halaman *owner myhomestay*.

Hasil yang Diharapkan	Sistem menampilkan *homestay* yang dimiliki oleh *owner* beserta informasi-informasi terkait *homestay* tersebut
Hasil yang Didapatkan	Sistem menampilkan *homestay* yang dimiliki oleh *owner* beserta informasi-informasi terkait *homestay* tersebut
Status	Valid

Hasil pengujian validasi untuk *use case* melihat *homestay owner* alternatif 1 menghasilkan nilai valid. Nilai valid tersebut didapatkan ketika hasil yang didapatkan sesuai dengan hasil yang diharapkan. Penjelasan lebih rinci terkait pengujian validasi pada *use case* ini ditunjukkan pada Tabel 6.28.

Tabel 6.28 Pengujian Validasi *Use Case* Melihat *Homestay Owner* Alternatif 1

Kode Kebutuhan	HOMIE-F-09
Nama Kasus Uji	Melihat *Homestay Owner*
Prosedur	1. Belum mendaftarkan *homestay* kemudian memasuki halaman *owner myhomestay*.
Hasil yang Diharapkan	Sistem akan mengarahkan ke halaman *add homestay*.
Hasil yang Didapatkan	Sistem akan mengarahkan ke halaman *add homestay*. 
Status	Valid

Hasil pengujian validasi untuk *use case* mengubah informasi *homestay* menghasilkan nilai valid. Nilai valid tersebut didapatkan ketika hasil yang didapatkan sesuai dengan hasil yang diharapkan. Penjelasan lebih rinci terkait pengujian validasi pada *use case* ini ditunjukkan pada Tabel 6.29.

Tabel 6.29 Pengujian Validasi *Use Case* Mengubah Informasi *Homestay*

Kode Kebutuhan	HOMIE-F-10
Nama Kasus Uji	Mengubah Informasi *Homestay*
Prosedur	1. Memasuki halaman *owner myhomestay*. 2. Memilih opsi edit *homestay*. 3. Megisi *form* dengan nilai terbaru 4. Memilih opsi simpan
Hasil yang Diharapkan	Sistem melakukan perubahan informasi *homestay* sesuai dengan nilai-nilai yang telah diinputkan oleh aktor ke tiap-tiap *field* yang terdapat dalam *form*.
Hasil yang Didapatkan	Sistem melakukan perubahan informasi *homestay* sesuai dengan nilai-nilai yang telah diinputkan oleh aktor ke tiap-tiap *field* yang terdapat dalam *form*.
Status	Valid

Hasil pengujian validasi untuk *use case* mengubah informasi kamar menghasilkan nilai valid. Nilai valid tersebut didapatkan ketika hasil yang didapatkan sesuai dengan hasil yang diharapkan. Penjelasan lebih rinci terkait pengujian validasi pada *use case* ini ditunjukkan pada Tabel 6.30.

Tabel 6.30 Pengujian Validasi *Use Case* Mengubah Informasi Kamar

Kode Kebutuhan	HOMIE-F-11
Nama Kasus Uji	Mengubah Informasi Kamar
Prosedur	1. Memasuki halaman *owner myhomestay*. 2. Memilih opsi edit kamar. 3. Memilih kamar yang akan diedit. 4. Megisi *form* dengan nilai terbaru 5. Memilih opsi simpan
Hasil yang Diharapkan	Sistem melakukan perubahan informasi kamar sesuai dengan nilai-nilai yang telah diinputkan oleh aktor ke tiap-tiap *field* yang terdapat dalam *form*.

Hasil yang Didapatkan	Sistem melakukan perubahan informasi kamar sesuai dengan nilai-nilai yang telah diinputkan oleh aktor ke tiap-tiap *field* yang terdapat dalam *form*.
Status	Valid

Hasil pengujian validasi untuk *use case* mengubah ketersediaan kamar menghasilkan nilai valid. Nilai valid tersebut didapatkan ketika hasil yang didapatkan sesuai dengan hasil yang diharapkan. Penjelasan lebih rinci terkait pengujian validasi pada *use case* ini ditunjukkan pada Tabel 6.31.

Tabel 6.31 Pengujian Validasi *Use Case* Mengubah Ketersediaan Kamar

Kode Kebutuhan	HOMIE-F-12
Nama Kasus Uji	Mengubah Ketersediaan Kamar
Prosedur	1. Memasuki halaman *edit rooms*. 2. Memilih opsi *open room* atau *close room*.
Hasil yang Diharapkan	Sistem mengubah status ketersediaan kamar *homestay*.
Hasil yang Didapatkan	Sistem mengubah status ketersediaan kamar *homestay*.
Status	Valid

Hasil pengujian validasi untuk *use case* melihat ketersediaan kamar menghasilkan nilai valid. Nilai valid tersebut didapatkan ketika hasil yang didapatkan sesuai dengan hasil yang diharapkan. Penjelasan lebih rinci terkait pengujian validasi pada *use case* ini ditunjukkan pada Tabel 6.32.

Tabel 6.32 Pengujian Validasi *Use Case* Melihat Ketersediaan Kamar

Kode Kebutuhan	HOMIE-F-13
Nama Kasus Uji	Melihat Ketersediaan Kamar
Prosedur	1. Memasuki halaman *owner check-in/check-out*.
Hasil yang Diharapkan	Sistem menampilkan ketersediaan kamar *homestay*.
Hasil yang Didapatkan	Sistem menampilkan ketersediaan kamar *homestay*.
Status	Valid

Hasil pengujian validasi untuk *use case* melakukan *check-in* menghasilkan nilai valid. Nilai valid tersebut didapatkan ketika hasil yang didapatkan sesuai dengan hasil yang diharapkan. Penjelasan lebih rinci terkait pengujian validasi pada *use case* ini ditunjukkan pada Tabel 6.33.

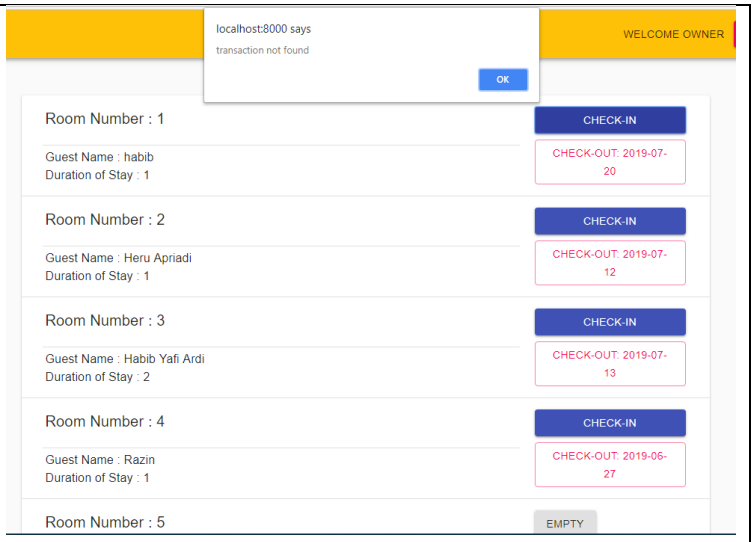
Tabel 6.33 Pengujian Validasi *Use Case* Melakukan *Check-In*

Kode Kebutuhan	HOMIE-F-14
Nama Kasus Uji	Melakukan Check-In
Prosedur	1. Memasuki halaman *owner check-in/check-out*. 2. Memilih opsi *check-in*.
Hasil yang Diharapkan	Sistem melakukan *check-in guest* terhadap kamar yang akan ditempati oleh *guest*.
Hasil yang Didapatkan	Sistem melakukan *check-in guest* terhadap kamar yang akan ditempati oleh *guest*.
Status	Valid

Hasil pengujian validasi untuk *use case* melakukan *check-in* alternatif 1 menghasilkan nilai valid. Nilai valid tersebut didapatkan ketika hasil yang didapatkan sesuai dengan hasil yang diharapkan. Penjelasan lebih rinci terkait pengujian validasi pada *use case* ini ditunjukkan pada Tabel 6.34.

Tabel 6.34 Pengujian Validasi *Use Case* Melakukan *Check-In* Alternatif 1

Kode Kebutuhan	HOMIE-F-14
Nama Kasus Uji	Melakukan Check-In
Prosedur	1. Memasuki halaman *owner check-in/check-out*. 2. Melakukan *check-in* terhadap pemesanan dengan status selain *capture*.
Hasil yang Diharapkan	Sistem menampilkan pesan *error transaction not found*.

	
Hasil yang Didapatkan	Sistem menampilkan pesan *error transaction not found*.
Status	Valid

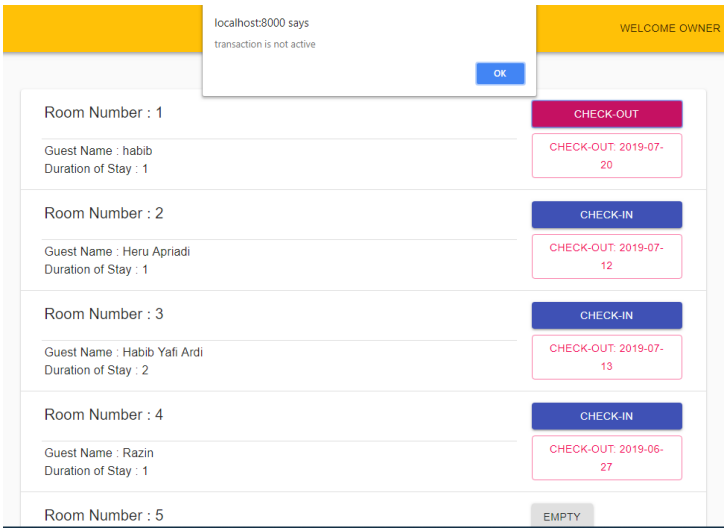
Hasil pengujian validasi untuk *use case* melakukan *check-out* menghasilkan nilai valid. Nilai valid tersebut didapatkan ketika hasil yang didapatkan sesuai dengan hasil yang diharapkan. Penjelasan lebih rinci terkait pengujian validasi pada *use case* ini ditunjukkan pada Tabel 6.35.

Tabel 6.35 Pengujian Validasi *Use Case* Melakukan *Check-Out*

Kode Kebutuhan	HOMIE-F-15
Nama Kasus Uji	Melakukan Check-Out
Prosedur	1. Memasuki halaman *owner check-in/check-out.* 2. Memilih opsi *check-out* .
Hasil yang Diharapkan	Sistem melakukan *check-out guest* terhadap kamar yang telah ditempati oleh *guest*.
Hasil yang Didapatkan	Sistem melakukan *check-out guest* terhadap kamar yang telah ditempati oleh *guest*.
Status	Valid

Hasil pengujian validasi untuk *use case* melakukan *check-out* alternatif 1 menghasilkan nilai valid. Nilai valid tersebut didapatkan ketika hasil yang didapatkan sesuai dengan hasil yang diharapkan. Penjelasan lebih rinci terkait pengujian validasi pada *use case* ini ditunjukkan pada Tabel 6.36.

Tabel 6.36 Pengujian Validasi *Use Case* Melakukan *Check-Out* Alternatif 1

Kode Kebutuhan	HOMIE-F-15
Nama Kasus Uji	Melakukan Check-Out
Prosedur	1. Memasuki halaman *owner check-in/check-out.* 2. Melakukan *check-out* terhadap pemesanan dengan status selain *active*.
Hasil yang Diharapkan	Sistem menampilkan pesan *error transaction is not active.* 
Hasil yang Didapatkan	Sistem menampilkan pesan *error transaction is not active.*
Status	Valid

Hasil pengujian validasi untuk *use case* menampilkan daftar pemesanan kamar menghasilkan nilai valid. Nilai valid tersebut didapatkan ketika hasil yang didapatkan sesuai dengan hasil yang diharapkan. Penjelasan lebih rinci terkait pengujian validasi pada *use case* ini ditunjukkan pada Tabel 6.37.

Tabel 6.37 Pengujian Validasi *Use Case* Menampilkan Daftar Pemesanan Kamar

Kode Kebutuhan	HOMIE-F-16
Nama Kasus Uji	Menampilkan Daftar Pemesanan Kamar
Prosedur	1. Memasuki halaman *owner* daftar order.
Hasil yang Diharapkan	Sistem menampilkan daftar pemesanan kamar yang berhasil dibayar pada *homestay* milik *owner*.

Hasil yang Didapatkan	Sistem menampilkan daftar pemesanan kamar yang berhasil dibayar pada *homestay* milik *owner*.
Status	Valid

Hasil pengujian validasi untuk *use case* menampilkan infografis pemasukan menghasilkan nilai valid. Nilai valid tersebut didapatkan ketika hasil yang didapatkan sesuai dengan hasil yang diharapkan. Penjelasan lebih rinci terkait pengujian validasi pada *use case* ini ditunjukkan pada Tabel 6.38.

Tabel 6.38 Pengujian Validasi *Use Case* Menampilkan Infografis Pemasukan

Kode Kebutuhan	HOMIE-F-17
Nama Kasus Uji	Melihat Detail *Homestay*
Prosedur	1. Menampilkan Infografis Pemasukan
Hasil yang Diharapkan	Sistem menampilkan infografis pendapatan dalam bentuk *chart*.
Hasil yang Didapatkan	Sistem menampilkan infografis pendapatan dalam bentuk *chart*.
Status	Valid

Hasil pengujian validasi untuk *use case* menampilkan daftar *homestay* menghasilkan nilai valid. Nilai valid tersebut didapatkan ketika hasil yang didapatkan sesuai dengan hasil yang diharapkan. Penjelasan lebih rinci terkait pengujian validasi pada *use case* ini ditunjukkan pada Tabel 6.39.

Tabel 6.39 Pengujian Validasi *Use Case* Menampilkan Daftar *Homestay*

Kode Kebutuhan	HOMIE-F-18
Nama Kasus Uji	Menampilkan Daftar *Homestay*
Prosedur	1. Memasuki halaman *admin* daftar *homestay*.
Hasil yang Diharapkan	Sistem menampilkan daftar seluruh *homestay* yang telah terdaftar kedalam sistem.
Hasil yang Didapatkan	Sistem menampilkan daftar seluruh *homestay* yang telah terdaftar kedalam sistem.
Status	Valid

Hasil pengujian validasi untuk *use case* menampilkan daftar *owner* menghasilkan nilai valid. Nilai valid tersebut didapatkan ketika hasil yang didapatkan sesuai dengan hasil yang diharapkan. Penjelasan lebih rinci terkait pengujian validasi pada *use case* ini ditunjukkan pada Tabel 6.40.

Tabel 6.40 Pengujian Validasi *Use Case* Menampilkan Daftar *Owner*

Kode Kebutuhan	HOMIE-F-19
Nama Kasus Uji	Menampilkan Daftar *Owner*
Prosedur	1. Memasuki halaman *admin* daftar *owner*.
Hasil yang Diharapkan	Sistem menampilkan daftar seluruh *owner* yang telah terdaftar kedalam sistem.
Hasil yang Didapatkan	Sistem menampilkan daftar seluruh *owner* yang telah terdaftar kedalam sistem.
Status	Valid

Hasil pengujian validasi untuk *use case* menghapus *homestay* menghasilkan nilai valid. Nilai valid tersebut didapatkan ketika hasil yang didapatkan sesuai dengan hasil yang diharapkan. Penjelasan lebih rinci terkait pengujian validasi pada *use case* ini ditunjukkan pada Tabel 6.41.

Tabel 6.41 Pengujian Validasi *Use Case* Menghapus *Homestay*

Kode Kebutuhan	HOMIE-F-20
Nama Kasus Uji	Menghapus *Homestay*
Prosedur	1. Memasuki halaman daftar *homestay*. 2. Memilih opsi hapus *homestay*.
Hasil yang Diharapkan	Sistem menghapus *homestay* beserta kamar-kamar yang terdaftar didalam milik *owner* dari sistem.
Hasil yang Didapatkan	Sistem menghapus *homestay* beserta kamar-kamar yang terdaftar didalam milik *owner* dari sistem.
Status	Valid

Hasil pengujian validasi untuk *use case* menghapus akun *owner* menghasilkan nilai valid. Nilai valid tersebut didapatkan ketika hasil yang didapatkan sesuai dengan hasil yang diharapkan. Penjelasan lebih rinci terkait pengujian validasi pada *use case* ini ditunjukkan pada Tabel 6.42.

Tabel 6.42 Pengujian Validasi *Use Case* Menghapus Akun *Owner*

Kode Kebutuhan	HOMIE-F-21
Nama Kasus Uji	Menghapus Akun *Owner*
Prosedur	1. Memasuki halaman daftar *owner*. 2. Memilih opsi hapus akun *owner*.
Hasil yang Diharapkan	Sistem menghapus akun *owner* beserta *homestay* dan kamar-kamar yang dipilih dari sistem.
Hasil yang Didapatkan	Sistem menghapus akun *owner* beserta *homestay* dan kamar-kamar yang dipilih dari sistem.
Status	Valid

Hasil pengujian validasi untuk *use case* menampilkan seluruh transaksi menghasilkan nilai valid. Nilai valid tersebut didapatkan ketika hasil yang didapatkan sesuai dengan hasil yang diharapkan. Penjelasan lebih rinci terkait pengujian validasi pada *use case* ini ditunjukkan pada Tabel 6.43.

Tabel 6.43 Pengujian Validasi *Use Case* Menampilkan Seluruh Transaksi

Kode Kebutuhan	HOMIE-F-22
Nama Kasus Uji	Menampilkan Seluruh Transaksi
Prosedur	1. Memasuki halaman *admin* daftar transaksi.
Hasil yang Diharapkan	Sistem menampilkan daftar seluruh pemesanan yang terjadi pada seluruh *homestay*.
Hasil yang Didapatkan	Sistem menampilkan daftar seluruh pemesanan yang terjadi pada seluruh *homestay*.
Status	Valid

Hasil pengujian validasi untuk *use case logout* menghasilkan nilai valid. Nilai valid tersebut didapatkan ketika hasil yang didapatkan sesuai dengan hasil yang diharapkan. Penjelasan lebih rinci terkait pengujian validasi pada *use case* ini ditunjukkan pada Tabel 6.44.

Tabel 6.44 Pengujian Validasi *Use Case Logout*

Kode Kebutuhan	HOMIE-F-23
Nama Kasus Uji	*Logout*
Prosedur	1. Menekan tombol *logout*.
Hasil yang Diharapkan	Sistem mengeluarkan aktor yang telah *login* kedalam sistem dan menjadi *guest*.
Hasil yang Didapatkan	Sistem mengeluarkan aktor yang telah *login* kedalam sistem dan menjadi *guest*.
Status	Valid

Hasil pengujian validasi untuk *use case* mengubah status pemesanan menghasilkan nilai valid. Nilai valid tersebut didapatkan ketika hasil yang didapatkan sesuai dengan hasil yang diharapkan. Penjelasan lebih rinci terkait pengujian validasi pada *use case* ini ditunjukkan pada Tabel 6.45.

Tabel 6.45 Pengujian Validasi *Use Case* Mengubah Status Pemesanan

Kode Kebutuhan	HOMIE-F-24
Nama Kasus Uji	Mengubah Status Pemesanan
Prosedur	1. Mengakses *end point* API ubah status pemesanan.
Hasil yang Diharapkan	Sistem melakukan perubahan terhadap status pemesanan berdasarkan data yang diberikan oleh aktor Midtrans dan mengirimkan email notifikasi kepada *owner homestay*.
Hasil yang Didapatkan	Sistem melakukan perubahan terhadap status pemesanan berdasarkan data yang diberikan oleh aktor Midtrans dan mengirimkan email notifikasi kepada *owner homestay*.
Status	Valid

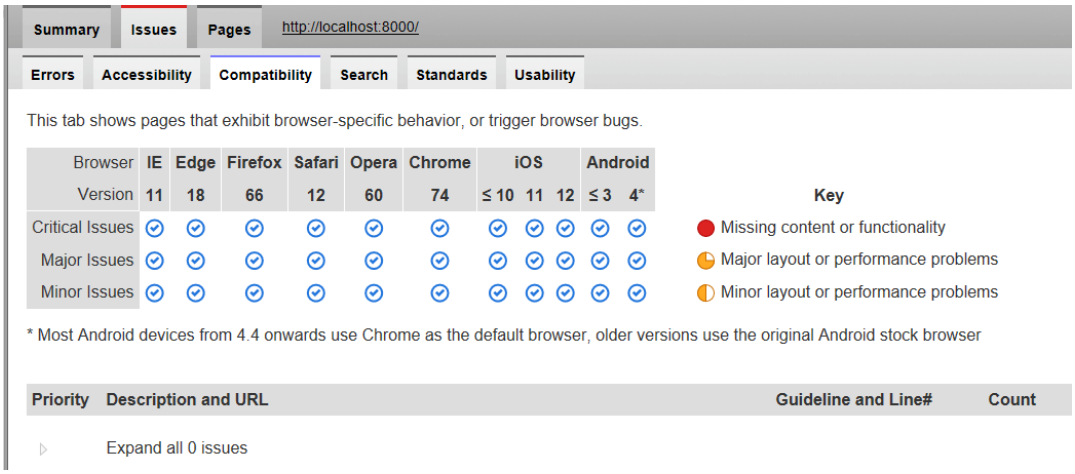
6.4 Pengujian Kompatibilitas

Pada tahap pengujian ini akan dilakukan pengujian berdasarkan kebutuhan non fungsional dengan kode kebutuhan HOMIE-N-01 yaitu sistem mampu berjalan di berbagai lingkungan perambah. Pengujian kompatibilitas pada penilitian ini akan menggunakan bantuan dari perangkat lunak SortSite yang memberikan layanan untuk melakukan pengujian kompatibilitas. Pengujian kompabilitas pada sub bab ini akan dijelaskan secara rinci pada Tabel 6.46.

Tabel 6.46 Pengujian Kompatibilitas

Kode Kebutuhan	HOMIE-N-01
Nama Kasus Uji	Sistem dapat dijalankan pada berbagai jenis perambah seperti edge, firefox, safari, google chrome, perangkat ios, dan perangkat android.
Prosedur	1. Membuka perangkat lunak dibeberapa *browser* yang berbeda seperti edge, firefox, safari, google chrome, perangkat ios, dan perangkat android.
Hasil yang Diharapkan	Sistem ditampilkan dengan baik terhadap *browser* seperti edge, firefox, safari, google chrome, perangkat ios, dan perangkat android.
Hasil yang Didapatkan	Sistem ditampilkan dengan baik terhadap *browser* seperti edge, firefox, safari, google chrome, perangkat ios, dan perangkat android.
Status	Valid

Pengujian kompatibilitas dilakukan dengan bantuan aplikasi SortSite. Aplikasi ini akan menjalankan aplikasi Homie di berbagai perambah. Perambah-perambah tersebut adalah IE, Edge, Firefox, Safari, Opera, Chrome, iOS, dan Android. Hasil dari pengujian tersebut mengasilkan nilai valid. Gambar 6.10 akan menunjukkan tangkapan layar dari hasil pengujian menggunakan SortSite.



Gambar 6.10 Hasil Pengujian Kompatibilitas

BAB 7 PENUTUP

7.1 Kesimpulan

Berdasarkan hasil pada tahap awal hingga akhir penelitian ini, maka didapatkanlah beberapa kesimpulan yaitu:

1. Pada tahap analisis kebutuhan yang telah dilakukan. Aplikasi Homie memiliki dua puluh empat kebutuhan fungsional dan satu kebutuhan non-fungsional yang membantu masyarakat khususnya masayarakat pada wilayah Kota Pekanbaru dan para masayarakat dari dalam dan luar kota yang berada di Kota Pekanbaru, untuk melakukan promosi kamar *homestay*, manajemen *homestay* bagi pemilik dan kemudian pemesanan kamar *homestay*, pembayaran secara online bagi calon penyewa kamar *homestay*. Proses studi literatur dan wawancara sehingga dapat mencakup permasalahan utama diantaranya, permasalahan dari segi promosi, permasalahan dari segi manajemen *homestay*, dan permasalahan dari segi efisiensi pelanggan dalam memesan kamar *homestay*. Selanjutnya pada sisi segmentasi pengguna didapatkan 3 aktor yang dapat berinteraksi dengan sistemn diantaranya *Guest, Owner,* dan *Admin*.
2. Pada tahap perancangan yang telah dilakukan. Aplikasi Homie menghasilkan beberapa poin perancangan, diantaranya perancangan arsitektur yang disusun oleh rancangan *sequence diagram* dan *class diagram*. Kemudian perancangan komponen yang berisi algoritme-algoritme utama yang digunakan dalam aplikasi Homie. Selanjutnya perancangan basisdata berupa *Physical Data Model* (PDM). Dan yang terakhir adalah perancangan antarmuka yang berisi rancangan *layout* dari *user interface* perangkat lunak.
3. Pada tahap implementasi yang telah dilakukan. Aplikasi Homie menghasilkan spesifikasi sistem, implementasi kode program berdasarkan rancangan algoritme, implementasi basis data, dan implementasi antarmuka berdasarkan rancangan antarmuka.
4. Pada tahap pengujian yang telah dilakukan, diperolehlah hasil dari tiga pengujian unit, satu pengujian integrasi, dan dua puluh empat pengujian validasi yang memberikan nilai 100% valid. Kemudian pada pengujian kompatibilitas didapatkan hasil bahwa aplikasi Homie dapat berjalan di delapan jenis perambah diantaranya edge, firefox, safari, google chrome, perangkat ios, dan perangkat android.

7.2 Saran

Saran yang perlu diberikan terhadap pengembangan lebih lanjut aplikasi Homie ini antara lain adalah:

1. Pemanfaatan aplikasi dengan cara melakukan ekspansi cakupan wilayah yang lebih luas. Mengingat pada peneilitian ini wilayah cakupan dibatasi oleh studi kasus wilayah, yaitu Kota Pekanbaru.
2. Pemanfaatan aplikasi yang sudah bersifat daring untuk memberikan pelayanan berupa *membership* keanggotaan terhadap aplikasi agar dapat meningkatkan efiesiensi proses bisnis.

DAFTAR PUSTAKA

A. S, R. and Shalahuddin, M., 2013. *Rekayasa Perangkat Lunak: Terstruktur dan Berorientasi Objek*. 4th ed. Bandung: Informatika Bandung.

Alam, S.S., Khatibi, A., Ahmad, M.I.S. and Ismail, H. Bin, 2007. Factors affecting e-commerce adoption in the electronic manufacturing companies in Malaysia. 17(1), pp.125–139.

Ambler, S.W., 2005. *The Elements of UML 2.0 Style*. New York: Cambridge University Press.

Aryanto, D., 2017. Penerapan dan Perancangan Sistem Informasi Pemesanan Kamar Hotel Berbasis Web (Studi Kasus pada Hotel Kesawan). 6(2), pp.46–51.

Booch, G., Rumbaugh, J. and Jacobson, I., 1998. *The Unified Modeling Language User Guide*. [online] *Techniques*, Available at: <http://portal.acm.org/citation.cfm?id=1088874>.

Caytiles, R.D. and Lee, S., 2014. A Review of an MVC Framework based Software Development. *International Journal of Software Engineering and Its Applications*, [online] 8(10), pp.213–220. Available at: <http://dx.doi.org/10.14257/ijseia.2014.8.10.19>.

Cermati, 2017. Mengenal Payment Gateway, Dari Cara Kerja sampai Keuntungannya buat Transaksi. [online] Tersedia di: <https://www.cermati.com/artikel/mengenal-payment-gateway-dari-cara-kerja-sampai-keuntungannya-buat-transaksi/> [Diakses 10 Juni 2019].

Detik, 2012. Hotel Melati Hingga Berbintang di Pekanbaru Laris karena Ada PON. [online] Tersedia di: https://finance.detik.com/properti/d-2015701/hotel-melati-hingga-berbintang-di-pekanbaru-laris-karena-ada-pon [Diakses 20 Juli 2019].

Febriyanto, E., Rahardja, U. and Alnabawi, N., 2018. Penerapan Midtrans sebagai Sistem Verifikasi Pembayaran pada Website iPanda. 4(2), pp.246–254.

Foster, E.C. and Godbole, S., 2016. *Database Systems*. [online] Available at: <http://link.springer.com/10.1007/978-1-4842-1191-5>.

Guru99, 2019. Learn Mccabe's Cyclomatic Complexity with Example. [online] Tersedia di: <https://www.guru99.com/cyclomatic-complexity.html> [Di akses 23 Maret 2018].

Kompas, 2010. Pembangunan Hotel di Pekanbaru Meningkat Tajam. [online] Tersedia di: <https://travel.kompas.com/read/2010/12/17/21492044/Pembangunan.Hotel.di.Pekanbaru.Meningkat.Tajam> [Diakses 20 Juli 2019].

Kumar, N., Zadgaonkar, A.S. and Shukla, A., 2013. Evolving a New Software

Development Life Cycle Model SDLC-2013 with Client Satisfaction. *International Journal of Soft Computing and Engineering(IJSCE)*, 3(1), pp.216–221.

McCool, S., 2012. *Laravel Starter - The definitive introduction to the Laravel PHP web development framework*.

Midtrans, 2015. Yuk Mengenal Payment Gateway Dan Berbagai Keuntungannya. [online] Tersedia di: <https://blog.midtrans.com/yuk-mengenalpayment-gateway-dan-keuntungan-bila-menggunakannya/> [Diakses 10 Juni 2019].

Nielsen, J., 1999. 10 Good Deeds in Web Design. [online] Tersedia di: <https://www.nngroup.com/articles/ten-good-deeds-in-web-design/> [Diakses 10 Juli 2019].

PowerMapper, 2019. *One Click Website Testing*. [online] Tersedia di: <https://www.powermapper.com/products/sortsite/> [Diakses 12 Juni 2019].

Pressman, R.S., 2009. *Software Engineering A Practitioner's Approach 7th Ed - Roger S. Pressman*. *Software Engineering A Practitioner's Approach 7th Ed - Roger S. Pressman*.

Solichin, A., 2010. *My SQL Dari Pemula Hingga Mahir*. Jakarta: Achmad Solichin.

Sommerville, I., 2011. *Software Engineering*. 9th ed. *Software Engineering*.

Suryanto, T., 2018. Penerapan E-Marketplace pada Distro Silver Squad. (1), pp.8–9.

Turban, E., King, D., Lee, J.K., Liang, T.-P. and Turban, D.C., 2015. *Electronic Com merce A Managerial and Social Networks Perspective*. 8th ed. New York: Springer.

Wall, L., Extraction, P., Language, R., Os, M., Scripting, P., Shell, U., Point, T., Point, T. and Point, T., 2015. ReactJS Tutorials Point Simply Easy Learning. p.2.

Whitten, J.L. and Bentley, L.D., 2007. *Systems Analysis and Design Methods*. New York: McGraw-Hill.

Yoon, I., Sussman, A., Memon, A. and Porter, A., 2008. Effective and Scalable Software Compatibility Testing. pp.63–73.

LAMPIRAN A MATERI WAWANCARA

Berikut merupakan lampiran dari materi wawancara untuk mendukung penelitian ini. Adapun materi wawancara ini dibagi menjadi 4 bagian yang terdiri dari **:**

1. **Gambaran Umum Tentang *Homestay* di Pekanbaru**

   Materi wawancara ini bertujuan untuk mengetahui gambaran umum tentang usaha penyawaan kamar di Pekanbaru.

2. **Sistem Promosi *Homestay* Penyewaan Kamar di Pekanbaru**

   Materi wawancara ini bertujuan untuk mengetahui apa saja upaya yang dilakukan dalam rangka meningkatkan jumlah tamu yang datang untuk peyewaan kamar (*homestay*) di Pekanbaru.

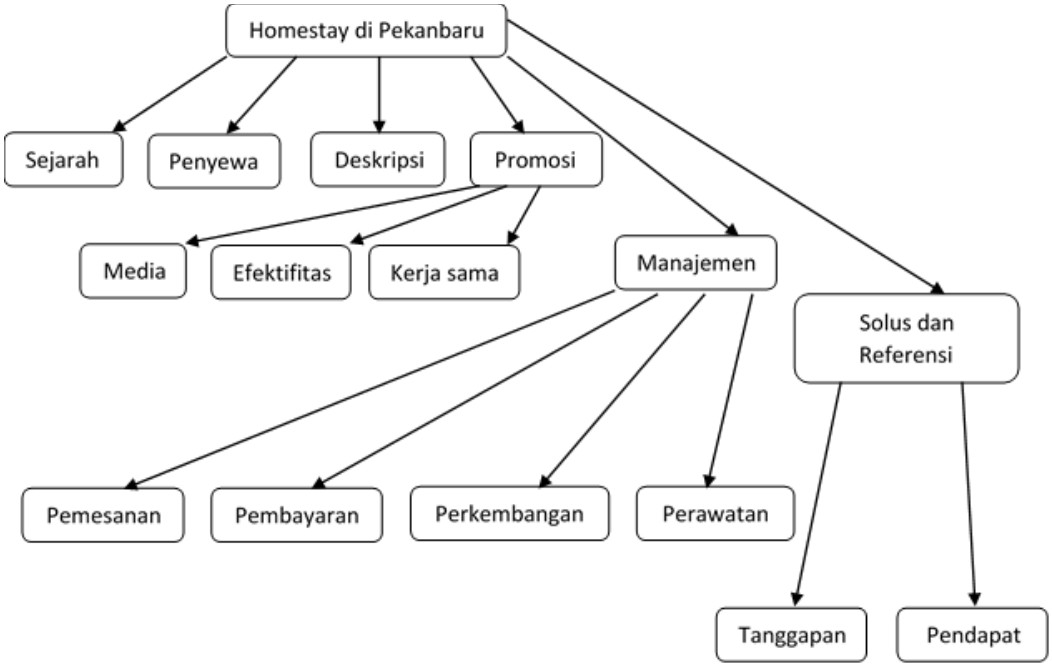
3. **Sistem Manajemen yang Dijalankan Oleh Pengelola/ Pemilik *Homestay* di Pekanbaru**

   Materi wawancara ini bertujuan untuk mengetahui sistem pengelolaan, tingkat perkembangan usaha kamar sewa (*homestay*) di Pekanbaru.

4. **Solusi dan Referensi**

   Materi wawancara ini bertujuan untuk mendapatkan tanggapan dan saran dari pemilik atau pengelola kamar sewa / *homestay* atas solusi dan referensi yang diberikan melalui penelitian PENGEMBANGAN APLIKASI RESERVASI DAN MANAJEMEN MARKET PLACE HOMESTAY BERBASIS WEB (STUDI KASUS: KOTA PEKANBARU).

Untuk susunan tree / work structured terkait materi wawancara dapat dilihat pada Gambar 7.1.



Gambar 7.1 *Work Structured Tree* mengenai materi wawancara

LAMPIRAN B HASIL WAWANCARA

Hari/Tanggal : Jumat, 29 Maret 2019

Waktu : 08.30 – 09.30 WIB

Tempat : Modern Room D19 (Jl. Muchtar Lutfi No.69, Simpang Baru, Kec. Tampan, Pekanbaru)

Narasumber : Bu Erni

A1. Gambaran umum tentang home stay di pekanbaru (penyewaan kamar) di Pekanbaru

Materi wawancara ini bertujuan untuk mengetahui gambaran umum tentang usaha penyawaan kamar di Pekanbaru.

No	Pertanyaan	Jawaban	Checklist
1	Sudah berapa lamakah usaha penyewaan kamar (home stay) ini dijalan kan?	Sudah 7 tahun lebih, yang pasti sebelum PON dan sudah rampung ketika PON 2012.	√
2	Berdasar kan jenis/ type penyewa, siapa sajakah yang memesan kamar di tempat anda ?	Penghuni homestay ini bervariasi namun yang paling sering adalah karyawan, terkadang untuk harian ada juga orang-orang yang singgah ke pekanbaru.	√
3	Untuk jangka waktu berapa lama biasanya tamu meyewa kamar di properti anda?	Tamu yang menyewa kamar ini bisa sewa secara harian, mingguan, dan bulanan.	√
4	Dari mana saja kah tamu berasal?	Luar provinsi riau. Dari Jakarta cukup banyak seperti karyawan-karyawan yang pindah tugas, atau manager-manager yang disuruh pegang wilayah pekanbaru tinggal disini.	√

5	Apa saja fasilitas yang tersedia	Lengkap, untuk umumnya ada dapur, air gratis, ruang tamu, parkir motor. Kalau dari kamarnya full furnish ya, seperti AC, kamar mandi dalam, kasur, TV, meja, lemari.	√
6.	Berapakan rentang harga sewa perkamar	Semua kamar modelnya sama. Yang harian 100.000 rupiah, mingguan 500.000 rupiah, dan yang bulanan 1,1 juta rupiah.	√

A2. Sistim promosi homestay penyewaan kamar di Pekanbaru

Materi wawancara ini bertujuan untuk mengetahui program apa saja upaya yang dilakukan dalam rangka meningkatkan jumlah tamu datang untuk meyewaan kamar (homestay) di Pekanbaru.

No	**Pertanyaan**	**Jawaban**	**Checklist**
1	Kegiatan apa saja yang dilakukan untuk meningkat jumlah tamu datang untuk menginap?	Pelayanan dari kaminya sendiri seperti cepat tanggap kalau ada komplain. Dan promosi-promosi secara online maupun offline.	√
2	Usaha apa saja yang dilakukan untuk mempromosikan homestay/ penyewaan kamar kepada orang2 yang berada di luar kota Pekanabru atau diluar pulau sumatera	Lewat sosial media seperti facebook dan whatsapp grup. Pernah juga promosi lewat OLX dulu terus sekarang sudah enggak karena sudah mulai berbayar OLX dan kurang efektif.	√
3	Seberapa efektifkan promosi melalui media sosial ini?	Kalau dari media sosial bisa terbilang efektif, karna	√

		banyak juga teman-teman saya di facebook yang dari luar kota, jadi bisa membantu penyebaran informasinya.	
4	Berapa persentasa tamu yang mengetahui home stay anda melalui media sosial dibanding papan pengumuman dan selebaran?	Yang paling besar itu dari temannya tamu yang pernah menginap disini, kalau dari traveloka atau airy gak terlalu besar ya. Malah kamar untuk traveloka dan airy lumayan sering kosong dan itu gak bisa kita pakai buat sewakan ke orang-orang yang tidak melalui airy atau traveloka.	√
5	Apakah ada dilakukan promosi melalui WEB?	Dulu OLX, sekarang sudah tidak gratis dan cukup mahal jadi tidak lanjut promosi di olxnya lagi.	√
6	Apakah homme stay anda sudah pernah mencoba bergabung dgn perusahaan penyedia market place seperti traveloka atau airy rooms dan yang lain nya?	Dengan airy rooms ada dengan traveloka juga ada. Tapi ya perjanjian-perjanjiannya harus kita penuhi seperti wajib menggunakan attribute airy dan logistik dari airy, mewajibkan ada air panas untuk kamar mandi, tidak boleh menggunakan kamar yang telah kerjasama dengan mereka untuk dipakai diluar	√

		sistem mereka, dan sebagainya.	

A3. Sistim manajemen yang dijalankan oleh pengelola/ pemilik homestay di Pekanbaru

Materi wawancara ini bertujuan untuk mengetahui sistem pengelolaan, tingkat perkembangan usaha kamar sewa (homestay) di Pekanbaru.

No	**Pertanyaan**	**Jawaban**	**Checklist**
1	Bagaimanakan sistim pemesanan kamar bagi tamu yang menyewa kamar?	Variasi ya, ada yang lewat airy rooms, ada yang lewat traveloka, ada yang telfon nomor, kalau sms tidak terlalu karna tidak pasti biasanya.	√
2	Bagaimana anda memperkenalkan fasilitas home stay anda kepada calon tamu?	Kita sampaikan langsung misal ditanya lewat telfon, atau juga kita suruh datang dlu ke lokasi ntar biar di tunjukkan oleh resepsionisnya. Kalau lewat traveloka atau airy rooms kan sudah jelas fasilitasnya disana.	√
3	Apakah ada resepsionis yang siap siagaa melayani tamu 24?	Kita punya resepsionis tapi tidak 24 jam, stand by nya dari pagi jam 7 sampai jam 5 sore, selanjutnya saya yang handle untuk jam 5 sore	√

		keatas paling sampai jam 11 malam.	
4	Bagaimana sitem pembayaran yg ditetpkan	Tidak ada ketetapan, bisa by cash atau transfer bank.	√
5	Apa ada kendala yang anda alami dengan sistem pembayaran by cash?	Harus rajin tulis detil tiap-tiap transaksinya.	√
6	Apakah ada kejadian dgn tamu dalam hal pembayaran konvesional ini?	Kadang kelupaan memberikan bukti pembayaran.	√
7	Bagaiman perkembangan usaha properti ini?	Biasa-biasa saja, tapi puncaknya dlu waktu PON 2012.	√
8	Bagaimanaa dengan rata rata omset yang diperoleh setiap tahun nya?	Waktu pon 2012 cukup besar karena kamar terisi semua setiap hari dan yang membutuhkan juga banyal.	√
9	Berapa kira-kira dana yang anda siapkan untuk biaya maintenance per tahun nya dan apakah terdata?	Tidak ada persiapan, kalau untuk listrik itu 1 juta perminggunya.	√
10	Apakah ada pembinaan dari pemerintah setempat tetang bagaimana kiat kiat meningkat kan usaha bisnis propeeti home stay ini?	Belum ada.	√

A 4. Solusi dan referensi

Materi wawancara ini bertujuan untuk mendapatkan tanggapan dan saran dari pemilik atau pengelola kamr sewa / home stay atas solusi dan referensi yang diberikan melalui penelitian PENGEMBANGAN APLIKASI RESERVASI DAN MANAJEMEN MARKET PLACE HOMESTAY BERBASIS WEB (STUDI KASUS: KOTA PEKANBARU)

No	Pertanyaan	Jawaban	Checklist
1	Bagaaimanaa tanggapan anda tentang menurun nya omset usaha penyewaan kamar ini?	Perlu adanya paltform untuk promosi yang baik dan gratis secara web promosi sebelumnya yang bersifat gratis kini sudah berbayar.	✓
2	apakah ada saran yang membangun apabila dibuatkan sebuah sistim yang dapat membantu dalam proses pengelolaan usaha home stay ini?	Pembukuan, dan pembayaran online tanpa harus bertemu pengelola dlu agar dapat melakukan pembayaran.	✓

Pekanbaru, 29 Maret 2019
Pengelola Homestay Kota Pekanbaru

(ERNI)

Hari/Tanggal : Rabu, 27 Maret 2019

Waktu : 11.00 – 12.00 WIB

Tempat : Liberty Homestay (Jl. Punai, Parit Indah, Kec. Bukit Raya, Pekanbaru)

Narasumber : Bu Melly Susanti

A1. Gambaran umum tentang home stay di pekanbaru (penyewaan kamar) di Pekanbaru

Materi wawancara ini bertujuan untuk mengetahui gambaran umum tentang usaha penyawaan kamar di Pekanbaru.

No	Pertanyaan	Jawaban	Checklist
1	Sudah berapa lamakah usaha penyewaan kamar (home stay) ini dijalan kan?	Sudah jalan 4 tahun lebih.	√
2	Berdasar kan jenis/ type penyewa, siapa sajakah yang memesan kamar di tempat anda ?	Dari berbagai kalangan bisa, namun karna tempat ini ada kerjasama dengan PT maka karyawan-karyawan PT yang lebih sering mengisi kamar disini.	√
3	Untuk jangka waktu berapa lama biasanya tamu meyewa kamar di properti anda?	Tamu yang menyewa kamar ini bisa sewa secara harian, mingguan, dan bulanan. Tapi biasanya pada ambil yang mingguan karena karyawan-karyawan PT.	√
4	Dari mana saja kah tamu berasal?	Penyewa kamar homestay disini biasanya kebanyakan dari berhubung mereka dibawah PT dan PT tersebut yang membawa kesini. Ada yang dari luar sumatra, tapi lebih banyak dari dumai.	√

5	Apa saja fasilitas yang tersedia	Fasilitas lengkap. Ada AC, kamar mandi dalam kamar, kasur spring bed, lemari, meja, kursi, kloset duduk, shower, TV, Wifi.	√
6.	Berapakan rentang harga sewa perkamar	Bervariasi sesuai model kamarnya, ada 3 tipe kamar. Yang pertama tipe standar 150.000 rupirah, tipe family 200.000 rupiah, dan untuk yang VIP 250.000 rupiah	√

A2. Sistim promosi homestay penyewaan kamar di Pekanbaru

Materi wawancara ini bertujuan untuk mengetahui program apa saja upaya yang dilakukan dalam rangka meningkatkan jumlah tamu datang untuk meyewaan kamar (homestay) di Pekanbaru.

No	**Pertanyaan**	**Jawaban**	**Checklist**
1	Kegiatan apa saja yang dilakukan untuk meningkat jumlah tamu datang untuk menginap?	Menyebarkan info lewat sosial media, brosur-brosur untuk tamu yang berkunjung dan juga menyebarkan selebaran kepada orang-orang di jalanan.	√
2	Usaha apa saja yang dilakukan untuk mempromosikan homestay/ penyewaan kamar kepada orang2 yang berada di luar kota Pekanabru atau diluar pulau sumatera?	Tidak terlalu karena kamar-kamar yang ada sudah terlalu ramai karena karyawan-karyawan PT.	√
3	Seberapa efektifkan promosi melalui media sosial ini?	Cukup efektif untuk mengisi beberapa kamar-kamar yang	√

		kosong tapi untuk jumlah banyak tidak terlalu	
4	Berapa persentasa tamu yang mengetahui home stay anda melalui media sosial dibanding papan pengumuman dan selebaran?	Lewat media sosial maupun selebaran ataupun papan informasi tidak terlalu besar persentasenya karena tidak terlalu gencar juga promosi yang dilakukan.	√
5	Apakah ada dilakukan promosi melalui WEB?	Tidak ada.	√
6	Apakah homme stay anda sudah pernah mencoba bergabung dgn perusahaan penyedia market place seperti traveloka atau airy rooms dan yang lain nya?	Belum pernah tapi pernah ditawarkan. Dan kami menolak karna kamar-kamar yang dibuat kerjasama harus tetap standby dan hanya boleh diisi oleh orang-orang yang memesan dari traveloka atau airy saja.	√

A3. Sistim manajemen yang dijalankan oleh pengelola/ pemilik homestay di Pekanbaru

Materi wawancara ini bertujuan untuk mengetahui sistem pengelolaan, tingkat perkembangan usaha kamar sewa (homestay) di Pekanbaru.

No	Pertanyaan	Jawaban	Checklist
1	Bagaimanakan sistim pemesanan kamar bagi tamu yang menyewa kamar?	Bisa menggunakan telfon atau sms namun jarang sekali yang menggunakan sms.Ada juga yang datang langsung kesini	√
2	Bagaimana anda memperkenalkan fasilitas home stay anda kepada calon tamu?	Jika melalui handphone akan dijelaskan secara detil. Jika penyewa kamar datang kesini kita berikan brosur dan juga sedikit menjelaskan perihal fasilitasnya	√
3	Apakah ada resepsionis yang siap siagaa melayani tamu 24?	Untuk resepsionis ada namun bergantian, atau shift-shift an. Kalau sudah malam biasanya diganti kan oleh security tapi tetap bisa menjadi resepsionis.	√
4	Bagaimana sitem pembayaran yg ditetpkan	Tidak ada ketetapan, penyewa bisa datang kesini bayar pakai uang cash atau via transfer	√
5	Apa ada kendala yang anda alami dengan sistem pembayaran by cash?	Tidak ada.	√

6	Apakah ada kejadian dgn tamu dalam hal pembayaran konvesional ini?	Sejauh ini belum ada.	
7	Bagaiman perkembangan usaha properti ini?	Pernah dulu mengalami penurunan, tapi itu dlu ketika masih manager yang lama. Jadi dapat disimpulkan tergantung managernya apakah bisa atau tidak untuk mengangkat perkembangan dari bisnis ini.	√
8	Bagaimanaa dengan rata rata omset yang diperoleh setiap tahun nya?	Setelah pergantian manager jadi lebih terasa peningkatan pendapatan tiap tahunnya yang meningkat.	√
9	Berapa kira-kira dana yang anda siapkan untuk biaya maintenance per tahun nya dan apakah terdata?	Kalau untuk dana maintenance pasti ada.	√
10	Apakah ada pembinaan dari pemerintah setempat tetang bagaimana kiat kiat meningkat kan usaha bisnis propeeti home stay ini?	Belum ada.	√

A 4. Solusi dan referensi

Materi wawancara ini bertujuan untuk mendapatkan tanggapan dan saran dari pemilik atau pengelola kamr sewa / home stay atas solusi dan referensi yang diberikan melalui penelitian PENGEMBANGAN APLIKASI RESERVASI DAN MANAJEMEN MARKET PLACE HOMESTAY BERBASIS WEB (STUDI KASUS: KOTA PEKANBARU)

No	Pertanyaan	Jawaban	Checklist
1	Bagaaimanaa tanggapan anda tentang menurun nya omset usaha penyewaan kamar ini?	Karena homestay ini sering digunakan oleh PT, jadi tidak terlalu kelihatan bagaimana omset dari tamu umum.	✓
2	apakah ada saran yang membangun apabila dibuatkan sebuah sistim yang dapat membantu dalam proses pengelolaan usaha home stay ini?	Mudah dioperasikan saja karena sistem yang sudah ada sekarang agak sulit digunakan dan kadang-kadang muncul error.	✓

Pekanbaru, 27 Maret 2019
Pengelola Homestay Kota Pekanbaru

Melly susanti
( )

Hari/Tanggal : Jumat, 29 Maret 2019

Waktu : 10.30 – 11.30 WIB

Tempat : Homestay Mandiri (Jl. Kutilang Sakti, Kec. Tampan, Pekanbaru)

Narasumber : Bu Reni

A1. Gambaran umum tentang home stay di pekanbaru (penyewaan kamar) di Pekanbaru

Materi wawancara ini bertujuan untuk mengetahui gambaran umum tentang usaha penyawaan kamar di Pekanbaru.

No	Pertanyaan	Jawaban	Checklist
1	Sudah berapa lamakah usaha penyewaan kamar (home stay) ini dijalan kan?	Sudah berjalan 2 tahun, sejak tahun 2017 pertama kali buat kamar hanya 2 saja.	√
2	Berdasar kan jenis/ type penyewa, siapa sajakah yang memesan kamar di tempat anda ?	Penghuni homestay ini bervariasi namun yang paling sering adalah karyawan-karyawan toko saya, terkadang ada juga orang tua mahasiswa yang anaknya akan wisuda.	√
3	Untuk jangka waktu berapa lama biasanya tamu meyewa kamar di properti anda?	Tamu yang menyewa kamar ini bisa sewa secara harian, bulanan, atau tahunan. Tapi yang paling sering adalah yang harian dan mingguan.	√
4	Dari mana saja kah tamu berasal?	Dari dalam kota, jarang dari luar kota.	√
5	Apa saja fasilitas yang tersedia	Fasilitas dari homestay ini standar seperti kamar, lemari, meja, kursi, sedangkan kamar mandi terletak diluar	√

6.	Berapakan rentang harga sewa perkamar	Harga sewa kamar tidak ada variasinya karna memang satu model semua jenis kamarnya. Untuk harian 100.000 rupiah sedangkan mingguan tidak ada harga khusus. Tetap dikali 7 hari.	√

A2. Sistim promosi homestay penyewaan kamar di Pekanbaru

Materi wawancara ini bertujuan untuk mengetahui program apa saja upaya yang dilakukan dalam rangka meningkatkan jumlah tamu datang untuk meyewaan kamar (homestay) di Pekanbaru.

No	**Pertanyaan**	**Jawaban**	**Checklist**
1	Kegiatan apa saja yang dilakukan untuk meningkat jumlah tamu datang untuk menginap?	Memasang papan informasi di gang depan sebelum masuk kesini agar kelihatan dari jalan.	√
2	Usaha apa saja yang dilakukan untuk mempromosikan homestay/ penyewaan kamar kepada orang2 yang berada di luar kota Pekanabru atau diluar pulau sumatera	Kalau untuk promosi diluar kota pekanbaru biasanya lewat kerabat-kerabat atau teman-teman yang berada diluar kota yang menginformasikan kalau kita disini punya homestay.	√
3	Seberapa efektifkan promosi melalui media sosial ini?	Tidak terlalu, tapi yang paling sering ya dari kerabat-kerabat atau teman-teman yang tahu kalau kita punya homestay.	√

4	Berapa persentasa tamu yang mengetahui home stay anda melalui media sosial dibanding papan pengumuman dan selebaran?	Untuk persentase tidak tahu pasti, namun beberapa penyewa kamar yang kami tanyakan menjawab bahwa mereka mendapatkan informasi dari orang-orang yang sudah pernah menginap disini dan juga ada yang mendapatkan informasi dari papan informasi yang ditempel didepan gang.	√
5	Apakah ada dilakukan promosi melalui WEB?	Sempat didaftarkan oleh kerabat tapi sudah gak dilanjutkan lagi karna kamar juga tidak terlalu banyak dan sering penuh.	√
6	Apakah homme stay anda sudah pernah mencoba bergabung dgn perusahaan penyedia market place seperti traveloka atau airy rooms dan yang lain nya?	Belum pernah. Dan juga tidak terlalu tertarik karena jumlah kamar yang ada masih terbilang sedikit dan kurang menguntungkan jika bekerjasama dengan traveloka ataupun airy. Dan juga banyak persyaratan dari traveloka maupun airy dalam hal kerjasama untuk menjaga kenyamanan penyewa homestay.	√

A3. Sistim manajemen yang dijalankan oleh pengelola/ pemilik homestay di Pekanbaru

Materi wawancara ini bertujuan untuk mengetahui sistem pengelolaan, tingkat perkembangan usaha kamar sewa (homestay) di Pekanbaru.

No	Pertanyaan	Jawaban	Checklist
1	Bagaimanakan sistim pemesanan kamar bagi tamu yang menyewa kamar?	Biasanya tamu yang akan memesan kamar menghubungi dlu lewat telfon atau sms. Para tamu langsung bilang kalau mau booking kamar untuk kapan dan sampai kapan tergantung kebutuhan mereka.	√
2	Bagaimana anda memperkenalkan fasilitas home stay anda kepada calon tamu?	Kalau lewat telfon langsung saya bilang fasilitasnya apa-apa saja dan kalau lewat sms saya balas apa-apa saja fasilitasnya karna informasi untuk fasilitas ini kita belum ada, hanya menampilkan informasi berupa No. handphone.	√
3	Apakah ada resepsionis yang siap siagaa melayani tamu 24?	Karena homestay ini bersebalahan dengan rumah saya maka praktisnya saya yang menjadi resepsionis namun tidak bisa juga stand by 24 jam.	√

4	Bagaimana sitem pembayaran yg ditetpkan	Dari segi pembayaran tidak ada ketetapan, jadi jika ada yang ingin bayar secara cash tidak masalah dan jika ada yang ingin bayar secara transfer bank juga tidak masalah.	√
5	Apa ada kendala yang anda alami dengan sistem pembayaran by cash?	Belum ada.	√
6	Apakah ada kejadian dgn tamu dalam hal pembayaran konvesional ini?	Belum ada juga.	√
7	Bagaiman perkembangan usaha properti ini?	Baik-baik saja tapi ya pasti ada jatuh bangunnya.	√
8	Bagaimanaa dengan rata rata omset yang diperoleh setiap tahun nya?	Tidak menentu, yang pasti tahun pertama lumayan menguntungkan pendapatannya.	√
9	Berapa kira-kira dana yang anda siapkan untuk biaya maintenance per tahun nya dan apakah terdata?	Tidak ada persiapan, jadi ketika misal ada yang rusak langusng saja pakai uang yang ada atau ditahan-tahan dlu baru panggil tukang.	√
10	Apakah ada pembinaan dari pemerintah setempat tetang bagaimana kiat kiat meningkat kan usaha bisnis propeeti home stay ini?	Belum ada.	√

A 4. Solusi dan referensi

Materi wawancara ini bertujuan untuk mendapatkan tanggapan dan saran dari pemilik atau pengelola kamr sewa / home stay atas solusi dan referensi yang diberikan melalui penelitian PENGEMBANGAN APLIKASI RESERVASI DAN MANAJEMEN MARKET PLACE HOMESTAY BERBASIS WEB (STUDI KASUS: KOTA PEKANBARU)

No	Pertanyaan	Jawaban	Checklist
1	Bagaaimanaa tanggapan anda tentang menurun nya omset usaha penyewaan kamar ini?	Belum bisa menanggapi juga karena kamar yang dimiliki tidak terlalu banyak dan pendapatan dari bisnis ini juga langsung dipakai.	✓
2	apakah ada saran yang membangun apabila dibuatkan sebuah sistim yang dapat membantu dalam proses pengelolaan usaha home stay ini?	Yang penting dapat mempermudah saja sudah cukup.	✓

Pekanbaru,29 Maret 2019
Pengelola Homestay Kota Pekanbaru

(PEMI)

Hari/Tanggal : Kamis, 28 Maret 2019

Waktu : 11.00 – 12.00 WIB

Tempat : Penginapan Harian dan Bulanan Pekanbaru (Jl. Dagang No. 2, Kec. Sukajadi, Pekanbaru)

Narasumber : Bu Hj. Suhaiti

A1. Gambaran umum tentang home stay di pekanbaru (penyewaan kamar) di Pekanbaru

Materi wawancara ini bertujuan untuk mengetahui gambaran umum tentang usaha penyawaan kamar di Pekanbaru.

No	Pertanyaan	Jawaban	Checklist
1	Sudah berapa lamakah usaha penyewaan kamar (home stay) ini dijalan kan?	Sudah sangat lama, dari tahun 1996.	√
2	Berdasar kan jenis/ type penyewa, siapa sajakah yang memesan kamar di tempat anda ?	Penyewa kamarnya dari berbagai kalangan seperti karyawan, ada juga 1 atau 2 mahasiswa, dan ada juga orang tua mahasiswa yang datang kesini buat menghadiri wisuda anaknya.	√
3	Untuk jangka waktu berapa lama biasanya tamu meyewa kamar di properti anda?	Tamu yang menyewa kamar ini bisa sewa secara harian, mingguan, bulanan, atau tahunan. Tapi yang paling sering adalah yang harian.	√
4	Dari mana saja kah tamu berasal?	Penyewa kamar homestay disini bervariasi, kalau dari luar kota seperti medan, sumatra barat, Jakarta dan malah ada yang pernah dari Malaysia.	√
5	Apa saja fasilitas yang tersedia	Bisa dibilang cukup, ada kasur, lemari, kamar mandi, TV, dan beberapa	√

		ada yang sudah dipasangi AC.	
6.	Berapakan rentang harga sewa perkamar	Harga sewa kamar ini untuk yang hariannya kisaran antara 50.000 rupiah sampai dengan 100.000 rupiah tergantung fasilitas tiap-tiap kamar.	√

A2. Sistim promosi homestay penyewaan kamar di Pekanbaru

Materi wawancara ini bertujuan untuk mengetahui program apa saja upaya yang dilakukan dalam rangka meningkatkan jumlah tamu datang untuk meyewaan kamar (homestay) di Pekanbaru.

No	Pertanyaan	Jawaban	Checklist
1	Kegiatan apa saja yang dilakukan untuk meningkat jumlah tamu datang untuk menginap?	Memberikan pelayanan yang maksimal biar yang menginap disini nyaman dan bisa direkomendasikan dengan teman-teman atau kerabatnya. Penah juga dlu diawal-awal promosi lewat koran.	√
2	Usaha apa saja yang dilakukan untuk mempromosikan homestay/ penyewaan kamar kepada orang2 yang berada di luar kota Pekanabru atau diluar pulau sumatera	Pernah dibantu dimasukkan ke internet, tapi sudah lupa dan bukan saya yang memasukan ke internet. Pernah juga share di media sosial seperti facebook dan grup-grup di whatsapp.	√
3	Seberapa efektifkan promosi melalui media sosial ini?	Tidak terlalu tapi patut dicoba. Kadang setiap saya tanyain yang mau menyewa kamar ini	√

		bilangnya dapat informasi dari teman-temannya. Jadi kita yakin ini promosi dari mulut ke mulut saja.	
4	Berapa persentasa tamu yang mengetahui home stay anda melalui media sosial dibanding papan pengumuman dan selebaran?	Bisa terbilang kecil ya, seperti yang sudah saya katakan tadi lebih banyak para penyewa kamar ini dapat informasi dari orang-orang yang sudah pernah tinggal disini. Kadang ada yang telfon mengaku temannya pernah tinggal disini dan dia mau menginap juga disini.	√
5	Apakah ada dilakukan promosi melalui WEB?	Yaitu tadi, dimasukkan ke internet tapi saya tidak tahu betul detilnya bagaimana	√
6	Apakah homme stay anda sudah pernah mencoba bergabung dgn perusahaan penyedia market place seperti traveloka atau airy rooms dan yang lain nya?	Tidak ada, kalau dilihat-lihat juga bisnis kami ini gak terlalu mewah kayak yang sudah kerjasama di traveloka atau airy rooms. Juga banyak persyaratan dan aturan-aturannya seperti kamar yang dipegang oleh mereka.	√

A3. Sistim manajemen yang dijalankan oleh pengelola/ pemilik homestay di Pekanbaru

Materi wawancara ini bertujuan untuk mengetahui sistem pengelolaan, tingkat perkembangan usaha kamar sewa (homestay) di Pekanbaru.

No	Pertanyaan	Jawaban	Checklist
1	Bagaimanakan sistim pemesanan kamar bagi tamu yang menyewa kamar?	Macam-macam modelnya, ada yang lewat telfon, ada yang lewat sms juga pernah, dna yang datang langsung juga ada.	√
2	Bagaimana anda memperkenalkan fasilitas home stay anda kepada calon tamu?	Misal yang lewat telfon dibilang langsung di telfon, yang lewat sms di balas lewat sms juga, kalau yang datang langsung lebih enak, bisa kita lihatkan langsung kamar dan fasilitasnya.	√
3	Apakah ada resepsionis yang siap siagaa melayani tamu 24?	Kalau resepsionisnya dari kita sendiri, keluarga ini, karna ini usaha keluarga kan. Kadang saya yang jadi resepsionisnya kadang anak saya atau keponakan saya yang lagi kosong waktunya tapi ya tidak sampai 24 jam juga.	√
4	Bagaimana sitem pembayaran yg ditetpkan	Dari segi pembayaran tidak ada ketetapan, jadi jika ada yang ingin bayar secara cash tidak masalah dan jika ada yang ingin bayar secara transfer bank juga tidak masalah.	√

5	Apa ada kendala yang anda alami dengan sistem pembayaran by cash?	Kadang uangnya susah dikumpulkan, soalnya langusng terpakai buat keperluan yang lain. Jadi kadang sulit juga kalau menghitung jumlah pemasukkannya.	√
6	Apakah ada kejadian dgn tamu dalam hal pembayaran konvesional ini?	Belum ada ya, paling yang mau perpanjang biasanya hutang dlu.	√
7	Bagaiman perkembangan usaha properti ini?	Menurun, tidak seperti tahun 90an dan waktu PON di riau.	√
8	Bagaimanaa dengan rata rata omset yang diperoleh setiap tahun nya?	Tidak menentu, kadang baik kadang tidak baik.	√
9	Berapa kira-kira dana yang anda siapkan untuk biaya maintenance per tahun nya dan apakah terdata?	Spontan saja, jadi misal ada yang rusak langsung panggil tukang buat perbaiki pakai uang yang ada.	√
10	Apakah ada pembinaan dari pemerintah setempat tetang bagaimana kiat kiat meningkat kan usaha bisnis propeeti home stay ini?	Tidak ada.	√

A 4. Solusi dan referensi

Materi wawancara ini bertujuan untuk mendapatkan tanggapan dan saran dari pemilik atau pengelola kamr sewa / home stay atas solusi dan referensi yang diberikan melalui penelitian PENGEMBANGAN APLIKASI RESERVASI DAN MANAJEMEN MARKET PLACE HOMESTAY BERBASIS WEB (STUDI KASUS: KOTA PEKANBARU)

No	Pertanyaan	Jawaban	Checklist
1	Bagaaimanaa tanggapan anda tentang menurun nya omset usaha penyewaan kamar ini?	Daerah sini sudah banyak orang yang bikin bisnis seperti ini, jadi bisa dibilang karna persaingan.	✓
2	apakah ada saran yang membangun apabila dibuatkan sebuah sistim yang dapat membantu dalam proses pengelolaan usaha home stay ini?	Bisa membantu bukan menyusahkan, dan juga kalau bisa ada yang mengatur keuangannya.	✓

Pekanbaru, 28 Maret 2019
Pengelola Homestay Kota Pekanbaru

(Hj. Suhaiti)