

KATA PENGANTAR

Puji dan syukur Penulis panjatkan kehadiran Allah SWT, karena hanya dengan rahmat dan bimbingannya Penulis dapat menyelesaikan tugas akhir dengan judul “Rancang Bangun Aplikasi Penginapan Backpacker” dengan baik. Tanpa rahmat dan bimbingan dari Tuhan Yang Maha Esa, maka niscaya Penulis tidak akan dapat menyelesaikan tugas akhir ini dengan baik dan tepat waktu.

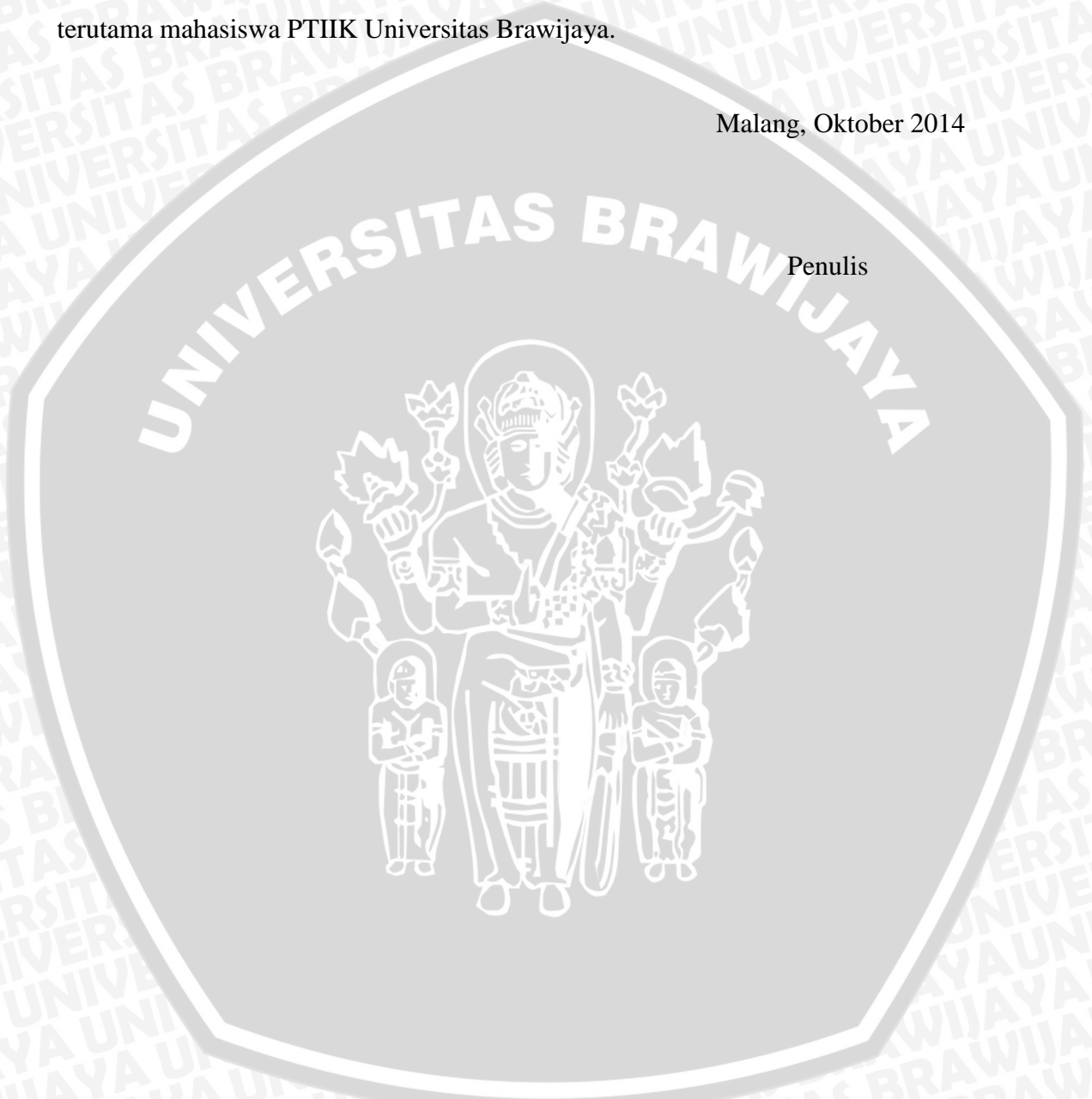
Penulisan dan penyusunan laporan skripsi ini dapat terlaksana dengan baik karena adanya bantuan secara langsung maupun tidak langsung dari pihak tertentu diantaranya:

1. Bapak Dr. Eng Herman Tolle, ST., MT. selaku dosen pembimbing I yang telah banyak memberikan ilmu dan saran untuk laporan skripsi ini.
2. Bapak Eriq Muh. Adams Jonemaro, ST., M.Kom_ selaku dosen pembimbing II yang juga banyak memberikan ilmu dan saran untuk laporan skripsi ini.
3. Kedua orang tua Endik Yulianto S.Pd., M.Pd. dan Hidayati Hanis S.Pd. yang telah memberi motivasi, kasih sayang serta dukungan moril dan materil. Kakak Filan Yurid Eden serta Rizky Fitri Lestari yang telah memberikan semangat dari awal sampai akhir pengerjaan skripsi ini.
4. Seluruh rekan kerja dan supervisor BPTIK PTIIK UB, terima kasih untuk segala pengalaman dan pelajaran selama bergabung bersama kalian.
5. Semua teman-teman PTIIK, khususnya Informatika/Illmu Komputer 2010 terima kasih atas segala bantuan dan dukungannya selama ini.
6. Segenap dosen dan karyawan PTIIK Universitas Brawijaya yang telah membantu pelaksanaan skripsi ini.
7. Semua teman-teman kontrakan manggar yang telah membantu dalam pengumpulan data skripsi.
8. Semua pihak yang tidak dapat penulis sebutkan satu per satu yang terlibat baik secara langsung maupun tidak langsung demi terselesaikannya tugas akhir ini.

Semoga jasa dan amal baik mendapatkan balasan dari Allah SWT. Dengan segala kerendahan hati, penulis menyadari sepenuhnya bahwa skripsi ini masih jauh dari sempurna karena keterbatasan materi dan pengetahuan yang dimiliki penulis. Akhirnya semoga skripsi ini dapat bermanfaat dan berguna bagi pembaca terutama mahasiswa PTIIK Universitas Brawijaya.

Malang, Oktober 2014

Penulis



ABSTRAK

Afif Fahmi Brilian. 2014. Rancang Bangun Aplikasi Penginapan Backpacker.

Program Teknologi Informasi dan Ilmu Komputer, Universitas Brawijaya, Malang. Dosen Pembimbing Dr. Eng Herman Tolle, ST., MT. dan Eriq Muh. Adams Jonemaro, ST., M.Kom.

Backpacking adalah suatu aktivitas berpergian menuju berbagai daerah tertentu atau yang biasa disebut *traveling* dengan membawa barang dan akomodasi yang terbatas dengan prinsip *educated* dan *travel cheap*. Tren *backpacking* yang meningkat dengan cepat sayangnya tidak diikuti dengan kemudahan para wisatawan dalam mencari tempat menginap yang sesuai. Sedangkan orang yang ingin menawarkan tempat tinggalnya tidak menemukan media yang tepat untuk menginformasikan tempat tinggalnya. Penelitian ini bermaksud untuk memberikan fasilitas kemudahan informasi bagi kedua belah pihak melalui media yang mudah diakses. Media tersebut adalah aplikasi pada sebuah perangkat *mobile* yang dapat diakses dengan mudah dimana saja dengan hanya menggunakan koneksi internet. Dalam penelitian ini sistem yang dibangun adalah sebuah aplikasi dengan *framework* phonegap menggunakan konsep *hybrid mobile application* dengan pada perangkat Android. Konsep *hybrid* adalah pembuatan aplikasi menggunakan HTML5, CSS3 dan javascript dimana ketiganya saling melengkapi dan secara maksimal bekerja dalam JQuery Mobile. Teknik penyimpanan dilakukan menggunakan media penyimpanan secara terpusat yaitu server sehingga perputaran informasi secara dinamis dapat terjadi. Dari hasil penelitian yang dilakukan, aplikasi dapat menjadi sebuah solusi sebagai media penghubung antara backpacker dan orang yang menawarkan ruangan tempat tinggalnya.

Kata kunci: *Backpack, mobile, tempat menginap, hybrid.*

ABSTRACT

Afif Fahmi Brilian. 2014. Rancang Bangun Aplikasi Penginapan Backpacker.

Information Technology and Computer Science Program, Brawijaya University, Malang. *Advisor:* Dr. Eng Herman Tolle, ST., MT. and Eriq Muh. Adams Jonemaro, ST., M.Kom.

Backpacking is an activity to go to various specific areas or commonly referred to traveling with carrying limited accommodation with the principles of educated and travel cheap. This trend is increasing rapidly, but unfortunately not followed by the convenience of travelers in finding an appropriate place to stay. Meanwhile, people who want to offer their home do not find the right media to inform the residence. This study intends to facilitate ease of information for both parties through accessible media. The media is an application on a mobile device that can be easily accessed anywhere using just an internet connection. The system which build based by this study use PhoneGap framework using the concept of a hybrid mobile application on Android device. Hybrid concept is the creation of applications using HTML5, CSS3, and JavaScript where all three are complementary and work optimally in the jQuery Mobile. Storage techniques performed using the centralized storage media servers so that the information can be easily distributed. From the research that is done, the application can be a solution as a liaison media between the backpacker and the people who offer the room.

Keywords: Backpack, mobile, places to stay, hybrid.

DAFTAR ISI

KATA PENGANTAR	i
ABSTRAK	iii
ABSTRACT	iv
DAFTAR ISI	v
DAFTAR GAMBAR	viii
DAFTAR TABEL	xi
BAB I	1
PENDAHULUAN	1
1.1 Latar Belakang	1
1.2 Rumusan Masalah	2
1.3 Batasan Masalah	3
1.4 Tujuan	3
1.5 Manfaat Penelitian	3
BAB II	4
KAJIAN PUSTAKA DAN DASAR TEORI	4
2.1 Kajian Pustaka	4
2.2 HyperText Markup Language (HTML)	6
2.3 Cascading Style Sheet (CSS)	8
2.4 JQuery	10
2.5 JQuery Mobile	11
2.6 Hypertext Preprocessor (PHP)	12
2.7 Aptana Studio 3	13
2.8 XAMPP	14
2.9 PhoneGap	15
2.10 Java Script Object Notation (JSON)	17
2.11 Web Service REST	17
BAB III	18
METODE PENELITIAN	18
3.1 Studi Literatur	18
3.2 Analisis Kebutuhan	20
3.3 Perancangan Konsep dan Desain Aplikasi	21

3.4	Implementasi.....	22
3.5	Pengujian dan Analisis	22
3.6	Pengambilan Kesimpulan dan Saran	23
BAB IV		24
PERANCANGAN DAN IMPLEMENTASI		24
4.1	Perancangan	24
4.1.1	Perancangan Umum Sistem	24
4.1.2	Analisis Kebutuhan	27
4.1.3	Perancangan perangkat Lunak.....	52
4.2	Implementasi.....	71
4.2.1	Spesifikasi Sistem	72
4.1.2.1	Spesifikasi Perangkat Keras	72
4.1.2.2	Spesifikasi Perangkat Lunak.....	73
4.2.2	Batasan – Batasan Implementasi.....	74
4.2.3	Implementasi Basis Data.....	75
4.2.4	Implementasi Class dan Assets Pada File Program	79
4.2.5	Implementasi Kode Program.....	81
4.2.5.1	Algoritma Proses Lihat Data	81
4.2.5.2	Algoritma Proses Ubah data	83
4.2.5.3	Algoritma Proses Hapus data	86
4.2.6	Implementasi Antarmuka Aplikasi.....	88
4.2.6.1	Halaman <i>Splash Screen</i>	88
4.2.6.2	Halaman Menu Utama.....	88
4.2.6.3	Halaman <i>Form</i>	89
4.2.6.4	Halaman Pencarian Berdasarkan Peta	90
4.2.6.5	Halaman Detail Penawaran Tempat Menginap	90
4.2.7	Implementasi Fitur Mobile.....	92
4.2.7.1	<i>File</i>	92
a.	Upload File	92
b.	Download File	94



4.2.7.2 *Geolocation*..... 97

BAB V101

 PENGUJIAN DAN ANALISIS101

 5.1 Pengujian101

 5.1.1 Pengujian Validasi.....102

 5.1.2 Pengujian Kompabilitas116

 5.1.3 Pengujian UAT (*User Acceptance Test*)117

 5.1.4 Pengujian Unit.....118

 5.2 Analisis Hasil Pengujian.....135

 5.2.1 Analisis Hasil Pengujian Validasi135

 5.2.2 Analisis Hasil Pengujian Kompabilitas.....135

 5.2.3 Analisis Hasil Pengujian UAT (*User Acceptance Test*).....135

 5.2.4 Analisis Hasil Pengujian Unit139

BAB VI141

 PENUTUP141

 6.1 Kesimpulan.....141

 6.2 Saran142

DAFTAR PUSTAKA DP-1

LAMPIRAN.....L-1

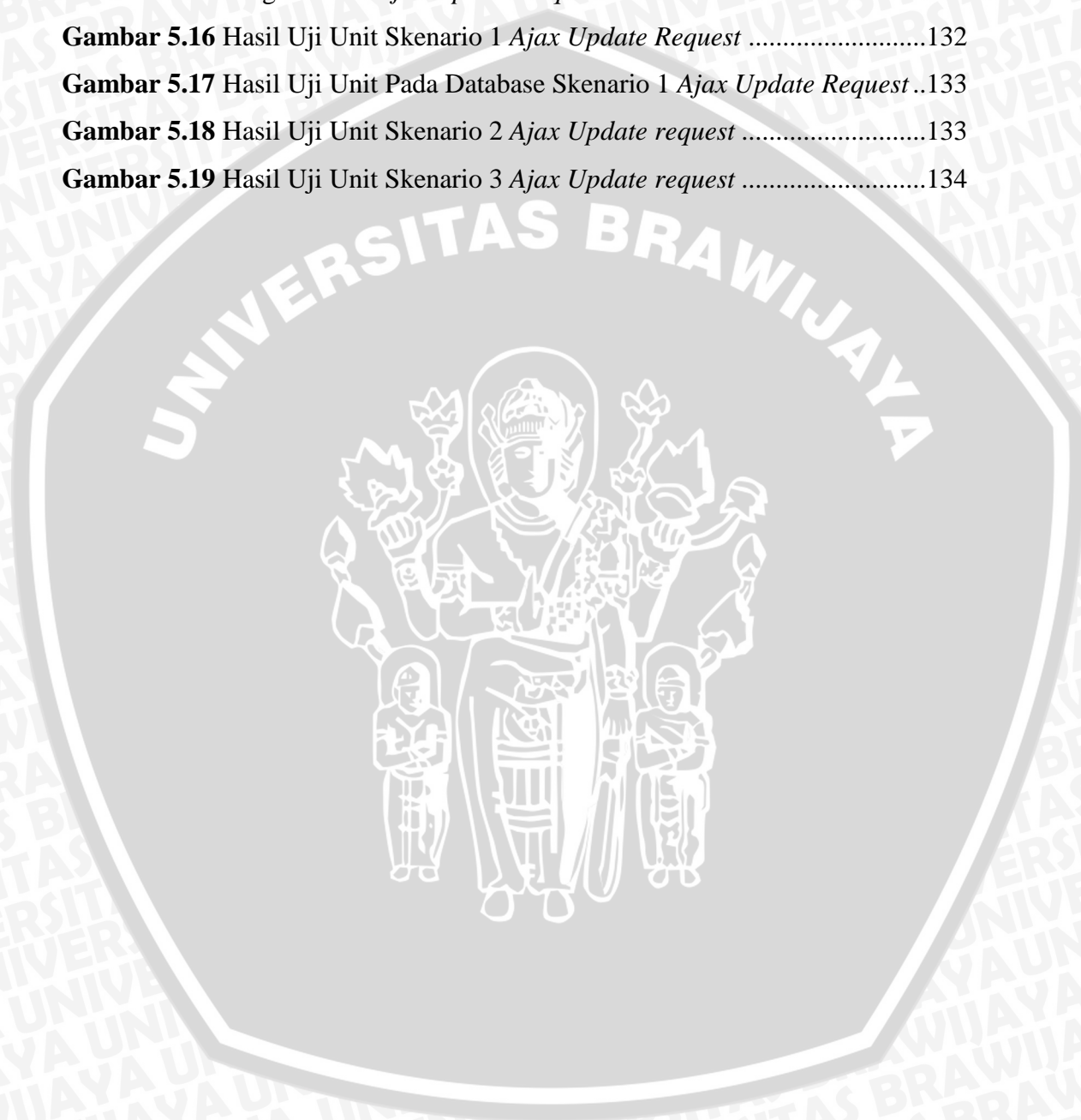


DAFTAR GAMBAR

Gambar 2.1 Grafik StaticCounter Global Stats	6
Gambar 2.2 Tampilan Aptana studio 3	13
Gambar 3.1 Flowchart tahapan kegiatan penelitian	18
Gambar 3.2 Diagram Alir Pemodelan <i>Waterfall</i>	19
Gambar 4.1 Diagram Pohon Perancangan	25
Gambar 4.2 Diagram Blok Perancangan Umum Sistem	26
Gambar 4.3 Diagram <i>Use case</i> Pengguna	32
Gambar 4.4 Diagram <i>Use case</i> Admin.....	32
Gambar 4.5 Perancangan Arsitektur Sistem Aplikasi	53
Gambar 4.6 <i>Entity Relational Diagram</i>	55
Gambar 4.7 <i>Class Diagram</i> Aplikasi	61
Gambar 4.8 <i>Sequence Diagram</i> Login dan Kelola Profil	63
Gambar 4.9 <i>Sequence Diagram</i> Kelola Penawaran Tempat Tinggal.....	64
Gambar 4.10 <i>Sequence Diagram</i> Lihat Tempat Menginap Melalui Pencarian Berdasarkan Parameter dan Kelola Komentar	65
Gambar 4.11 <i>Sequence Diagram</i> Lihat Deskripsi Aplikasi	66
Gambar 4.12 <i>Sequence Diagram</i> Kelola Permintaan Menginap.....	66
Gambar 4.13 <i>Sequence Diagram</i> Kelola Data pada Admin.....	67
Gambar 4.14 <i>Sequence Diagram</i> Hapus Data pada Admin	68
Gambar 4.15 Struktur Pohon Menu Aplikasi.....	68
Gambar 4.16 Antar Muka Halaman Splash Screen.....	69
Gambar 4.17 Antar Muka Halaman Menu Utama (<i>Home</i>)	69
Gambar 4.18 Antar Muka Halaman Form.....	70
Gambar 4.19 Antar Muka Halaman Pencarian berdasarkan Peta	71
Gambar 4.20 Halaman Detail Penawaran Tempat Menginap	71
Gambar 4.21 Diagram Pohon Implementasi	72
Gambar 4.22 Diagram Konseptual <i>Entity Relationship</i>	76
Gambar 4.23 Implementasi Query <i>Create</i> tabel <i>tbl_post</i>	77
Gambar 4.24 Implementasi Query <i>Create</i> tabel <i>tbl_request</i>	77

Gambar 4.25 Implementasi Query <i>Create</i> tabel <i>tbl_user</i>	78
Gambar 4.26 Implementasi Query <i>Create</i> tabel <i>tbl_rating</i>	78
Gambar 4.27 Implementasi Algoritma Proses Lihat Data.....	83
Gambar 4.28 Implementasi Algoritma Proses Ubah data	85
Gambar 4.29 Implementasi Algoritma Proses Hapus data JS	87
Gambar 4.30 Implementasi Algoritma Proses Hapus data PHP	87
Gambar 4.31 Tampilan <i>Splash Screen</i>	88
Gambar 4.32 Tampilan Menu Utama.....	89
Gambar 4.33 Tampilan Pencarian	89
Gambar 4.34 Tampilan Login	89
Gambar 4.35 Tampilan Pencarian Berdasarkan Peta	90
Gambar 4.36 Tampilan Tab General Information.....	91
Gambar 4.37 Tampilan Tab Rute	91
Gambar 4.38 Tampilan Tab Gambar Pendukung.....	91
Gambar 4.39 Tampilan Tab Rating	91
Gambar 4.40 Implementasi Algoritma Upload File.....	93
Gambar 4.41 Tampilan Upload File.....	94
Gambar 4.42 Implementasi Algoritma Download File	96
Gambar 4.43 Tampilan Download File.....	96
Gambar 4.44 Implementasi Geolocation.....	100
Gambar 4.45 Tampilan Map dengan Geolocation	100
Gambar 5.1 Diagram Pohon Pengujian dan Analisis	101
Gambar 5.2 Kasus Uji Unit <i>Ajax Load Data</i>	118
Gambar 5.3 Diagram Alir <i>Ajax Load Data</i>	120
Gambar 5.4 Hasil Uji Unit Skenario 1 <i>Ajax Load Data</i>	121
Gambar 5.5 Hasil Uji Unit Skenario 2 <i>Ajax Load Data</i>	122
Gambar 5.6 Hasil Uji Unit Skenario 3 <i>Ajax Load Data</i>	123
Gambar 5.7 Hasil Uji Unit Skenario 4 <i>Ajax Load Data</i>	124
Gambar 5.8 Kasus Uji Unit <i>Ajax Create Request</i>	124
Gambar 5.9 Diagram Alir <i>Ajax Create Request</i>	126
Gambar 5.10 Hasil Uji Unit Skenario 1 <i>Ajax Create Request</i>	127
Gambar 5.11 Hasil Uji Unit Pada Database Skenario 1 <i>Ajax Create Request</i>	127

Gambar 5.12 Hasil Uji Unit Skenario 2 <i>Ajax Create request</i>	128
Gambar 5.13 Hasil Uji Unit Skenario 3 <i>Ajax Create request</i>	129
Gambar 5.14 Kasus Uji Unit <i>Ajax Update Request</i>	130
Gambar 5.15 Diagram Alir <i>Ajax Update Request</i>	131
Gambar 5.16 Hasil Uji Unit Skenario 1 <i>Ajax Update Request</i>	132
Gambar 5.17 Hasil Uji Unit Pada Database Skenario 1 <i>Ajax Update Request</i> ..	133
Gambar 5.18 Hasil Uji Unit Skenario 2 <i>Ajax Update request</i>	133
Gambar 5.19 Hasil Uji Unit Skenario 3 <i>Ajax Update request</i>	134



DAFTAR TABEL

Tabel 4.1 Identifikasi Aktor	28
Tabel 4.2 Spesifikasi Kebutuhan Fungsional Pengguna	29
Tabel 4.3 Spesifikasi Kebutuhan Fungsional Admin.....	31
Tabel 4.4 Skenario <i>Use Case</i> Kelola Profil Pribadi	33
Tabel 4.5 Skenario <i>Use Case</i> Kelola Penawaran Tempat Menginap.....	34
Tabel 4.6 Skenario <i>Use Case</i> Cari Data Dengan Berbagai Parameter.....	36
Tabel 4.7 Skenario <i>Use Case</i> Cari Data Melalui Peta.....	38
Tabel 4.8 Skenario <i>Use Case</i> Cari Data Berdasarkan Lokasi Terdekat.....	40
Tabel 4.9 Skenario <i>Use Case</i> Lihat Detail <i>Post</i>	41
Tabel 4.10 Skenario <i>Use Case</i> Memberikan Komentar	43
Tabel 4.11 Skenario <i>Use Case</i> Lihat Deskripsi Aplikasi	44
Tabel 4.12 Skenario <i>Use Case</i> Kelola Permintaan Menginap	45
Tabel 4.13 Skenario <i>Use Case</i> Kelola Penawaran Tempat Menginap.....	47
Tabel 4.14 Skenario <i>Use Case</i> Kelola Akun Pengguna	49
Tabel 4.15 Skenario <i>Use Case</i> Kelola Komentar.....	51
Tabel 4.16 Spesifikasi kebutuhan non-fungsional	52
Tabel 4.17 Struktur Tabel <i>tbl_province</i>	56
Tabel 4.18 Struktur Tabel <i>tbl_city</i>	56
Tabel 4.19 Struktur Tabel <i>tbl_user</i>	56
Tabel 4.20 Struktur Tabel <i>tbl_rating_content</i>	57
Tabel 4.21 Struktur Tabel <i>tbl_post</i>	57
Tabel 4.22 Struktur Tabel <i>tbl_post_picture</i>	58
Tabel 4.23 Struktur Tabel <i>tbl_comment</i>	58
Tabel 4.24 Struktur Tabel <i>tbl_rating</i>	59
Tabel 4.25 Struktur Tabel <i>tbl_level</i>	59
Tabel 4.26 Struktur Tabel <i>tbl_forgot_password</i>	60
Tabel 4.27 Struktur Tabel <i>tbl_request</i>	60
Tabel 4.28 Spesifikasi Perangkat Keras Komputer.....	72
Tabel 4.29 Spesifikasi Perangkat Keras <i>Smartphone</i>	73
Tabel 4.30 Spesifikasi Perangkat Lunak Komputer.....	74

Tabel 4.31 Spesifikasi Perangkat Lunak <i>Smartphone</i>	74
Tabel 4.32 Implementasi class pada kode program *.java.....	79
Tabel 4.33 Implementasi assets pada kode program *.html	79
Tabel 4.34 Implementasi assets pada kode program *.js	80
Tabel 5.1 Kasus Uji Validasi Lihat Profil Pribadi	102
Tabel 5.2 Kasus Uji Validasi Edit Profil Pribadi	103
Tabel 5.3 Kasus Uji Validasi Lihat Penawaran Tempat Menginap	104
Tabel 5.4 Kasus Uji Validasi Tambah Penawaran Tempat Menginap.....	105
Tabel 5.5 Kasus Uji Validasi Edit Penawaran Tempat Menginap	106
Tabel 5.6 Kasus Uji Validasi Lihat Penawaran Tempat Menginap Melalui Pencarian Parameter.....	107
Tabel 5.7 Kasus Uji Validasi Lihat Penawaran Tempat Menginap Melalui Pencarian Peta	108
Tabel 5.8 Kasus Uji Validasi Lihat Penawaran Tempat Menginap Berdasarkan Lokasi Terdekat	109
Tabel 5.9 Kasus Uji Validasi Lihat Detail Penawaran Tempat Menginap	110
Tabel 5.10 Kasus Uji Validasi Lihat Komentar	111
Tabel 5.11 Kasus Uji Validasi Tambah Komentar.....	112
Tabel 5.12 Kasus Uji Validasi Lihat Deskripsi Aplikasi	113
Tabel 5.13 Kasus Uji Validasi Lihat <i>Request</i> Penawaran Tempat Tinggal	114
Tabel 5.14 Kasus Uji Validasi Tambah <i>Request</i> Penawaran Tempat Tinggal....	115
Tabel 5.15 Kasus Uji dan Hasil Pengujian Kompatibilitas Android dan iOS.....	116
Tabel 5.16 Hasil Pengujian Usability.....	117
Tabel 5.17 Interpretasi Skor Likert	136
Tabel 5.18 Persentase Keseluruhan Jawaban	136
Tabel 5.19 Status Pengujian UAT	138



BAB I

PENDAHULUAN

1.1 Latar Belakang

Berpergian menuju berbagai daerah tertentu atau yang biasa disebut *traveling* telah menjadi sebuah agenda favorit bagi berbagai kalangan masyarakat. Tujuan dari *traveling* sangat bermacam-macam mulai dari berwisata untuk mencari hiburan dan kesenangan hingga mencari sebuah pengetahuan baru ataupun keduanya. *Traveling* dilakukan dengan cara yang berbeda tergantung dari orang itu sendiri, cara yang berbeda tersebut dipengaruhi oleh anggaran biaya dan pilihan seseorang dalam menentukan bagaimana dia akan berwisata. Seseorang yang tidak ingin repot akan cenderung memilih menggunakan jasa agen untuk berwisata, tetapi beberapa kalangan memiliki pilihan yang berbeda dan lebih memilih untuk berwisata tanpa membawa barang-barang yang memberatkan dan tidak perlu tidur di hotel tetapi cukup di suatu tempat yang dapat dijadikan untuk beristirahat atau tidur [CCO-06]. *Traveling* dengan cara tersebut biasa disebut dengan *backpacking*.

Tren *backpacking* sudah semakin berkembang dan telah menjadi salah satu pilihan utama sebagai cara untuk berpergian yang tidak hanya dilakukan oleh wisatawan lokal melainkan juga wisatawan internasional. *Backpacking* memiliki beberapa prinsip penting sehingga kegiatan berwisata tersebut termasuk sebagai *backpacking*, yaitu *educated* (bersifat mendidik) dan *travel cheap* (berwisata dengan cara dan biaya yang murah) [ASN-13]. Poin penting tersebut dapat menarik perhatian masyarakat untuk saling bertukar informasi dan menambah pengetahuan dengan cara menyediakan sebuah tempat untuk tinggal secara sementara.

Meningkatnya tren *backpacking* sayangnya tidak diikuti dengan kemudahan dalam mencari informasi dari tempat yang ingin dituju terutama sebuah tempat untuk menginap. Aplikasi pencari penginapan yang ada hanya menyediakan fitur informasi hotel atau penginapan yang sudah ada. Dibutuhkan

informasi tentang penginapan lain terutama dari masyarakat yang ingin tempatnya untuk ditinggali. Dari interaksi kedua belah pihak yaitu orang menyewakan tempatnya dan orang yang mencari tempat akan terjadi hubungan saling menguntungkan secara edukatif dan juga dari segi aplikasi [DMZ-06]. Dari interaksi tersebut akan memperlancar update informasi secara terus-menerus dan membantu pengguna dalam mencari informasi tentang tempat tinggal yang dibutuhkan.

Dengan latar belakang seperti yang telah dipaparkan diatas maka dilakukan pengembangan aplikasi yang berjalan pada perangkat bergerak dengan basis *mobile* OS Android. Dengan pengembangan pada perangkat bergerak maka akses informasi bagi para *backpacker* akan semakin mudah dan cepat. Pengembangan aplikasi akan dilakukan dengan menggunakan pendekatan *hybrid app* dengan alasan pengembangan untuk semua platform terutama *mobile* OS Android dan iOS yang merupakan platform yang paling banyak digunakan di dunia. Aplikasi ini diharapkan dapat menyediakan berbagai macam informasi tempat menginap bagi para *backpacker* yang langsung didapat dari orang yang mempromosikan tempat tinggalnya. Aplikasi diharapkan dapat menyediakan fitur pencarian dalam berbagai bentuk seperti informasi peta (map) berdasarkan lokasi *backpacker* saat menggunakan aplikasi, dan fitur pencarian berdasarkan berbagai macam parameter dalam aplikasi ini. Selain itu, diharapkan terdapat fitur untuk mempromosikan tempat tinggal yang akan disewakan dengan berbagai fitur dalam penentuan lokasinya. Aplikasi ini diharapkan dapat digunakan dimanapun dengan koneksi internet sehingga *backpacker* dapat dengan mudah mencari informasi tempat menginap dari manapun dan kapanpun.

1.2 Rumusan Masalah

Berdasarkan pada permasalahan yang telah dijelaskan pada bagian latar belakang, rumusan masalah dari penelitian ini yaitu:

1. Bagaimana membuat suatu aplikasi *mobile* yang dapat menyediakan dan mencari informasi tempat menginap sementara selain hotel.
2. Sejauh mana kemudahan pengguna dalam mencari tempat menginap sementara saat menggunakan aplikasi yang dibangun.

3. Bagaimana kompatibilitas aplikasi untuk *platform* yang berbeda terutama Android dan iOS pada aplikasi *hybrid* yang dibangun.

1.3 Batasan Masalah

Batasan masalah yang digunakan pada penelitian ini adalah lokasi pengambilan data yang digunakan yaitu pada daerah Malang raya dan sekitarnya. Aplikasi dibangun dengan menggunakan bahasa pemrograman HTML5, CSS3 dan JavaScript. Pada sisi server aplikasi dibangun menggunakan bahasa pemrograman PHP.

1.4 Tujuan

Tujuan dari dilaksanakannya penelitian yaitu membangun sebuah aplikasi Sistem pencarian tempat menginap untuk *backpacker* pada perangkat *mobile*. Dimana aplikasi tersebut dapat membantu penggunanya dalam melakukan pencarian tempat menginap berdasarkan berbagai macam kriteria atau menawarkan ruangan tempat tinggal.

1.5 Manfaat Penelitian

Manfaat serta keandalan yang didapatkan setelah penelitian ini selesai dilaksanakan antara lain:

1. Memberikan tempat bagi pengguna yang memiliki tempat untuk disewakan dan pengguna yang mencari tempat menginap untuk saling bertemu.
2. Sebagai sumber informasi tempat dan lokasi untuk menginap yang lengkap dan terus diperbarui.
3. Memberikan keuntungan dari segi pengetahuan karena kedua belah pihak yang saling bertemu pasti berasal dari tempat yang berbeda. Selain itu juga dapat memberikan keuntungan dari segi finansial bagi pengguna yang mematok harga untuk tempat yang disediakan.

BAB II

KAJIAN PUSTAKA DAN DASAR TEORI

Pada bab ini akan dibahas mengenai kajian pustaka dan dasar teori yang digunakan penulis dalam menyusun skripsi. Kajian pustaka didapatkan berdasarkan dari penelitian penelitian yang berhubungan dengan kegiatan backpacking. Dasar teori digunakan sebagai dasar pembuatan perangkat lunak seperti HTML, CSS, javascript, PHP, JQuery mobile dan phonegap.

2.1 Kajian Pustaka

Kajian pustaka pertama pada skripsi ini yaitu berdasarkan tinjauan dari penelitian – penelitian sebelumnya yaitu “*Backpackers ethnography*” oleh Anders Sørensen, “*Backpackers’ motivations The Role of Culture and Nationality*”, dan “*From drifter to gap year, mainstreaming Backpacker Travel*” oleh Elsevier. Kajian pustaka yang kedua didapatkan dari data yang diperoleh dari situs gs.statcounter.com.

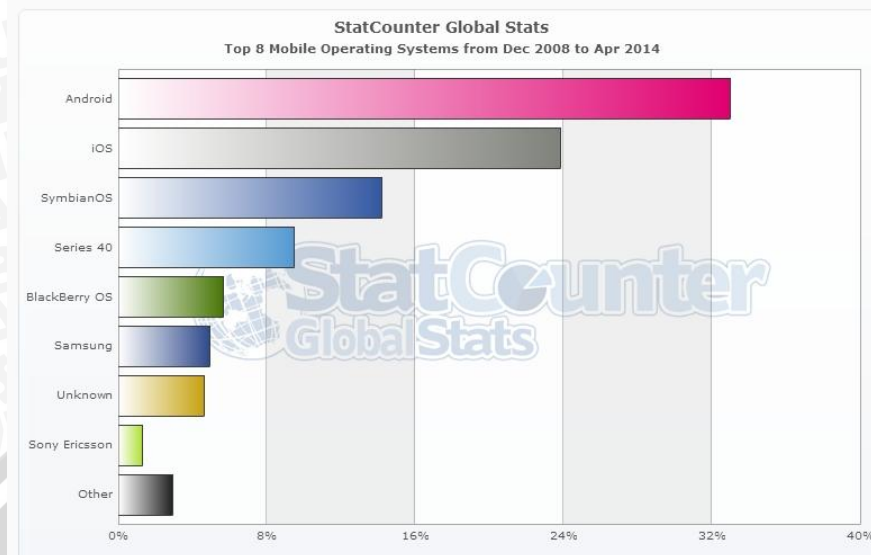
Pada referensi tersebut dijelaskan bahwa tren *backpacking* sudah semakin berkembang dan telah menjadi salah satu pilihan utama sebagai cara untuk berpergian yang tidak hanya dilakukan oleh wisatawan lokal melainkan juga wisatawan internasional. *Backpacking* memiliki beberapa prinsip penting sehingga kegiatan berwisata tersebut termasuk sebagai *backpacking*, yaitu *educated* (bersifat mendidik) dan *travel cheap* (berwisata dengan cara dan biaya yang murah). Masyarakat yang melakukan *backpacking* rata – rata berumur dari 18 – 33 tahun, dan *backpacker* mewakili tingkat edukasi yang tinggi dari negaranya masing – masing. Hal tersebut sesuai dengan prinsip *educated* yang secara tidak langsung komunikasi antara penduduk local dan *backpacker* akan meningkatkan informasi antara dua belah pihak yang berbeda [ASN-13]. Prinsip *travel cheap* sendiri didasarkan pada motivasi *backpacker* yang sebenarnya yaitu tujuan untuk mendapatkan jati diri yang sesungguhnya melalui perjalanan jauh yang dilakukan sendiri. Pada studi yang lebih lanjut motivasi dari *backpacker*

sendiri meluas yaitu untuk melakukan rekreasi dan mengunjungi berbagai tempat dengan sebisa mungkin mengurangi kebutuhan dan menekan pengeluaran [DMZ-06]. Melakukan rekreasi dengan cara yang baru seperti *backpacker* telah banyak menarik minat masyarakat yang ingin mencari cara dan tantangan baru dalam berpariwisata.

Meningkatnya tren *backpacking* yang didasari dari motivasi *backpacker*, sayangnya tidak diikuti dengan kemudahan dalam mencari informasi dari tempat yang ingin dituju terutama sebuah tempat untuk menginap. Aplikasi yang ada hanya menyediakan fitur informasi hotel atau penginapan yang sudah ada seperti pada beberapa contoh aplikasi dan website yang ada seperti <http://www.backpackingsouthafrica.co.za/> atau aplikasi yang terdapat pada Android google playstore. Dari fitur informasi tersebut tentu *backpacker* belum mendapatkan informasi yang cukup yaitu informasi tentang penginapan. Tentu *backpacker* tidak mungkin mencari penginapan yang berharga mahal karena prinsip dari *backpacking* itu sendiri. Maka dibutuhkanlah sebuah tempat penginapan yang berharga murah terutama dari masyarakat yang ingin tempatnya untuk ditinggali. Dari interaksi kedua belah pihak yaitu orang menyewakan tempatnya dan orang yang mencari tempat akan terjadi hubungan saling menguntungkan secara edukatif dan juga dari segi aplikasi [DMZ-06]

Dari situs gs.statcounter.com diperoleh data yang perhitungan persentase data statistik penggunaan smartphone di dunia yang didapatkan dari desember 2008 hingga april 2014. Data tersebut menunjukkan tingkat pengguna *mobile OS* mulai dari yang tertinggi sampai terendah yang digambarkan dalam sebuah grafik.

Di dalam Gambar 2.1 di bawah ini menunjukkan grafik dari 9 *mobile OS* yang tertinggi berdasarkan data statistik yang diperoleh



Gambar 2.1 Grafik StatCounter Global Stats

2.2 HyperText Markup Language (HTML)

HyperText Markup Language (HTML) adalah bahasa untuk memarkup (memformat) yang untuk membuat halaman *web* dan informasi lain yang dapat ditampilkan melalui *web browser*. HTML akan memberitahu *web browser* bagaimana menampilkan konten dengan kode bahasa tertentu. HTML dibuat oleh kolaborasi Caillau TIM dengan Berners-lee Robert ketika mereka bekerja di CERN pada tahun 1989 (CERN adalah lembaga penelitian fisika energi tinggi di Jenewa). HTML merupakan standar Internet yang didefinisikan dan dikendalikan penggunaannya oleh *World Wide Web Consortium* (W3C) [ABH-06][SNW-10][DML-04].

Kegunaan dari HTML yaitu [UNK-11]:

- Mengintegrasikan gambar dengan tulisan.
- Membuat Pranala.
- Mengintegrasikan berkas suara dan rekaman gambar hidup.
- Membuat *form* interaktif.

HTML mempunyai ciri utama sebagai penanda bahasa yaitu tag dan elemen. Tag merupakan perintah HTML yang biasa digunakan untuk membuat

tampilan dari dokumen HTML, sedangkan elemen adalah bagian-bagian yang terstruktur dalam HTML itu sendiri [UNK-11][ABH-06].

Cara penulisan HTML adalah dengan menggunakan tag HTML di awal dan di akhir, seperti contoh <title>Penulisan HTML</title> [UNK-11][SNW-10]. Namun tidak semua tag pembuka diakhiri dengan tag penutup, contohnya
<HR> [UNK-11]. Pada HTML terdapat tag utama yang digunakan yaitu <HTML> </HTML>, <HEAD> </HEAD>, <BODY> </BODY>. Dengan struktur disetiap dokumn HTML adalah sebagi berikut [UNK-11][ABH-06][SNW-10]:

```
<HTML>
  <HEAD><!--Merupakan header HTML--></HEAD>
  <BODY><!--Merupakan isi HTML--></BODY>
</HTML> [W3C-01]
```

Selama tahun 90-an telah muncul berbagai versi HTML dengan berbagai versi HTML dengan berbagai macam fituranya. Versi HTML yang ada tersebut antara lain:

1. Versi 1.0

Merupakan versi pertama setelah lahirnya HTML dan mempunyai kemampuan seperti untuk heading, paragraph, hypertext, list, cetak tebal (*bold*), dan cetak miring (*italic*) pada teks. Selain itu juga dukungan peletakan image pada dokumen dengan tidak memperbolehkan peletakan teks di sekelilingnya (*wrapping*).

2. Versi 2.0

Dirilis pada tanggal 14 Januari 1996 dengan beberapa kemampuan tambahan seperti penambahan form.

3. Versi 3.0

Dirilis pada tanggal 18 Desember 1997, yang dikenal dengan HTML+. Versi ini memiliki beberapa fitur tambahan seperti fitur table dalam paragraph. 30

4. Versi 3.2

Dirilis pada bulan Mei 1998, merupakan versi terbaru penyempurnaan dari versi 3.0.

5. Versi 4.0

Dirilis pada tanggal 24 Desember 1999 dengan penambahan beberapa fitur seperti adanya link, imagemaps, image, dan lain-lain.

6. Versi 5.0

Dirilis pada tahun 2009 yang menjadi standar baru untuk HTML, XHTML, dan DOM HTML. HTML5 ini merupakan proyek kerja sama antara W3C (*World Wide Web Consortium*) dan WHATG (*Web Hypertext Application Technology Working Group*).

2.3 Cascading Style Sheet (CSS)

Cascading Style Sheets (CSS) adalah bahasa *style sheet* yang digunakan untuk memisahkan struktur desain dari dokumen bahasa *markup*. Penerapan yang paling umum adalah untuk *style* dari *web page* yang ditulis dalam HTML. CSS diatur oleh *World Wide Web Consortium* (W3C) [ASS-10][ASR-13].

Fungsi utama CSS adalah untuk memisahkan konten yang akan ditampilkan dengan tampilannya. HTML sebagai template untuk membuat sebuah aplikasi sebenarnya sudah cukup, tetapi dengan adanya CSS dimungkinkan konten yang ditampilkan dapat diperbagus sedemikian rupa. File penyimpanan style ini dikenal sebagai *Cascading Style Sheet* (CSS).

CSS dirancang untuk pemisahan konten dari presentasi dokumen, termasuk seperti *layout*, warna, dan huruf. CSS ini dapat digunakan untuk beberapa dokumen HTML, selama HTML tersebut tersambung dengan CSS ini [ASS-10]. Pemisahan ini dapat meningkatkan aksesibilitas konten, memberikan fleksibilitas, dan lebih banyak kendali untuk spesifikasi dari karakteristik penyajian, memungkinkan beberapa halaman untuk berbagi *format* yang sama, dan mengurangi kompleksitas dan pengulangan konten struktural [CSS-13].

Terdapat beberapa cara untuk memasang CSS pada dokumen HTML yaitu :

- *External Style Sheet* (file CSS berbeda dari file HTML),
- *Internal Style Sheet* (Kode CSS dipasang di dalam tag head HTML)

- dan *Inline Style Sheet* (Kode CSS langsung dipasang di tag HTML)

2.3.1 Sintaks Cascading Style Sheet (CSS)

Sintaks CSS dibuat atas tiga bagian, yaitu selector, property, dan value.

Susunannya terlihat seperti berikut:

```
Selector: {property:value;}
```

Selector adalah elemen atau tag html yang ingin diberi style. Property adalah style atau attribute yang akan dipakai oleh selector. Value merupakan nilai yang akan diterapkan pada atribut. Penulisan antara property dan value dipisahkan dengan tanda titik dua (:), dan dibuka dengan tanda { dan ditutup dengan }.

Contoh :

```
h3{color:red}
```

pada kode diatas maka tag h3 akan berwarna merah selama CSS masih tersambung dengan file HTML. Jika atribut yang diberi style lebih dari satu maka disisipkan tanda titik koma (;) untuk memisahkannya.

2.3.2 Penggunaan CSS 3

Penggunaan CSS3 pada penelitian ini yaitu pada desain yang lebih responsive terutama penggunaan CSS *webkit* seperti berikut :

```
-webkit-background-size: cover;
```

```
-moz-background-size: cover;
```

```
-o-background-size: cover;
```

```
background-size: cover;
```

penggunaan background size : cover digunakan untuk membuat sebuah background yang berupa gambar agar secara otomatis menjadi background

yang menyeluruh pada layar. *Style* tersebut penyederhanaan dari CSS sebelumnya yang lebih panjang menjadi lebih sederhana dengan cukup menggunakan sintaks diatas.

```
input::-webkit-input-placeholder
```

sintak *selector* css diatas merupakan fitur baru yang terdapat pada CSS3 yang digunakan pada penelitian ini. Selector diatas memungkinkan peneliti untuk merubah *style* dari *placeholder* sebuah input melalui css.

2.4 JQuery

JQuery merupakan suatu *framework (library) Javascript* yang merupakan perantara ataupun interaksi antara Javascript dan HTML. JQuery memungkinkan digunakan tanpa menggunakan teknik *binding event* pada HTML (*unobstrusive*), dan dapat memodifikasi CSS [BLE-10]. Pada perkembangannya JQuery memiliki kehandalan dan kelebihan yang cukup banyak. Hal tersebut menyebabkan banyak developer *web* menggunakannya. JQuery memiliki slogan “*Write less, do more*” yang kurang lebih maksudnya adalah kesederhanaan dalam penulisan *code*, tapi dengan hasil yang lebih banyak. JQuery ini secara *standart* memerlukan kode yang cukup panjang, namun dengan dibantunya Javascript, maka penggunaan JQuery tersebut lebih mudah dilakukan [MUI-11].

Berapa contoh kemampuan dan keunggulan dari jquery [BLE-10]:

- a. Mengakses bagian halaman tertentu dengan mudah.
- b. Mengubah tampilan bagian halaman tertentu.
- c. Mengubah isi dari halaman.
- d. Merespon interaksi *user* dalam halaman.
- e. Menambahkan animasi ke halaman.
- f. Mengambil informasi dari *server* tanpa *me-refresh* seluruh halaman.
- g. Menyederhanakan penulisan Javascript biasa.

Cara sederhana penggunaan JQuery dapat menggunakan nama tag yang ada pada HTML, dengan memberikan tanda '\$' pada *element* tipe *selector*, contohnya:

```
Var nama = $("Budi");
```

Atau dapat dengan menggunakan selector ID dari HTML maupun CSS, contohnya:

```
Document.getElementById("nama");
```

atau

```
var nama = $("#nama);
```

2.5 JQuery Mobile

JQuery Mobile merupakan suatu *framework web* yang dikembangkan oleh tim proyek dari jQuery. Pengembangan yang dikerjakan difokuskan pada pembuatan suatu *framework* yang sesuai dengan berbagai macam smartphone dan pc tablet. *Framework* jQuery Mobile memiliki kompatibilitas dengan *framework* aplikasi mobile lainnya seperti PhoneGap, Worklight dan lainnya [JQM-14].

Beberapa fitur dari jQuery Mobile yaitu:

- Kompatibel dengan semua *platform mobile* utama serta semua *desktop browser* utama, termasuk iOS, Android, Blackberry, WebOS, Symbian, Windows Phone, dan lainnya.
- Dibangun berdasarkan jQuery sehingga membantu pengguna yang sudah akrab dengan perintah jQuery.
- *Theming framework* yang memungkinkan pembuatan tema sesuai keinginan.
- Dependensi terbatas dan ringan untuk mengoptimalkan kecepatan.
- Basis kode yang sama secara otomatis akan menyesuaikan ntuk skala ulayar manapun.
- HTML5 yang berbasis konfigurasi untuk membuat halaman dengan scripting minimal.

2.6 Hypertext Preprocessor (PHP)

Hypertext Preprocessor (PHP) merupakan bahasa berbentuk skrip yang ditempatkan pada sisi *server* dan diproses di *server* atau dikenal dengan istilah bahasa pemrograman *server-side scripting*. PHP ini berupa sintak atau *code* yang dapat ditanamkan atau disisipkan ke dalam *HyperText Markup Language* (HTML) [SNW-10]. Pada awalnya PHP merupakan kependekan dari *Personal Home Page* (Situs personal) dan dibuat pertama kali oleh Rasmus Lerdorf pada tahun 1995. Pada waktu itu PHP masih bernama *Form Interpreted* (FI), yang wujudnya berupa sekumpulan skrip yang digunakan untuk mengolah data *formulir* dari *web* [ASS-10].

PHP saat ini banyak dipakai untuk memrogram situs *web* dinamis, karena PHP dapat berinteraksi dengan *database*, *file* dan *folder* sehingga PHP dapat digunakan untuk membangun sebuah CMS. PHP adalah bahasa yang *cross-platform* yaitu dapat berjalan ataupun digunakan pada system operasi yang berbeda-beda [TRY-07].

Cara kerja PHP bermula dari adanya permintaan suatu halaman *web* dari *web* browser kepada *web server* yang dituju. Berdasarkan URL (*Universal Resource Locater*) yang *web* browser minta kepada *web server*, *web server* akan mengirimkan hasil dari pencarian kepada *web* browser untuk diterjemahkan berupa kode HTML dan akan ditampilkan di layar client [SNW-10][DAS-07].

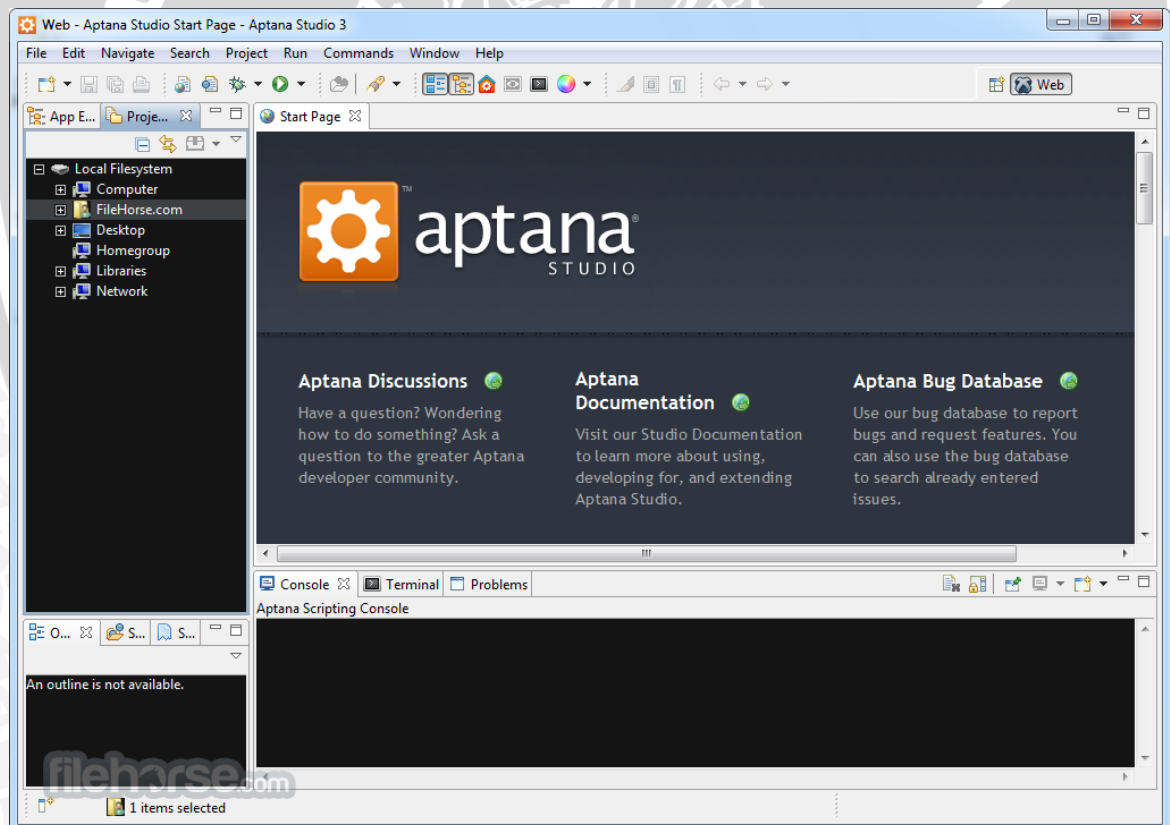
Berikut ini adalah beberapa kelebihan PHP dibandingkan dengan Bahasa Pemrograman *server-side scripting* yang lain adalah [DAS-07][SNW-10]:

1. PHP mampu dikembangkan diberbagai *Operating System*.
2. PHP dapat dihubungkan dengan berbagai macam *database*.
3. PHP mampu beroperasi pada multi *server*.
4. PHP merupakan *server-side scripting* yang *open source*.
5. PHP dapat diintegrasikan dengan dokumen HTML.

PHP ini memiliki ciri utama sebagai tag yang digunakan dalam menulisannya. Cara penulisan PHP adalah dengan menggunakan tag PHP di awal dan di akhir, seperti contoh `<?php echo "Sistem Informasi";?>` [SNW-10].

2.7 Aptana Studio 3

Aptana Studio adalah aplikasi desain dan pengembangan *web*. *Aptana Studio* mendukung berbagai macam jenis bahasa pemrograman *web* seperti ASP, ASP .NET, JSP, PHP, dan lain sebagainya. *Aptana Studio* merupakan program yang menyediakan *editor WYSIWYG visual* (bahasa sehari-hari yang disebut sebagai *Design view*) dan kode *editor* dengan fitur standar seperti *syntax highlighting*, *code completion*, dan *code collapsing* serta fitur lebih canggih seperti *real-time syntax checking* dan *code introspection* untuk menghasilkan petunjuk kode untuk membantu pengguna dalam menulis kode.



Gambar 2.2 Tampilan Aptana studio 3

2.8 XAMPP

XAMPP merupakan singkatan dari X(empat *Operating System*), Apache, MySQL, PHP, Perl. XAMPP adalah sebuah paket aplikasi *open source platform* aplikasi yang terdiri dari sekumpulan aplikasi pokok atau aplikasi utama yaitu Apache, MySQL, PHP, Perl [ASR-13].

Dari semua aplikasi pokok tersebut dirancang menjadi satu sehingga memungkinkan *end user* mudah untuk mengembangkan aplikasi berbasis *web* yang baru melalui sebuah *interface* yang mudah dipahami. XAMPP dapat dijalankan pada beragam sistem operasi seperti Linux, Microsoft Windows, Apple Mac, Sun Solaris maupun OpenSolaris dsb [ASR-13].

- a. Apache, yaitu sebuah aplikasi *server web open source*.

Apache bersifat *open source*, artinya setiap orang boleh menggunakannya, mengambil dan bahkan mengubah kode programnya. Tugas utama Apache adalah untuk menghasilkan *web* yang benar sesuai dengan kode PHP yang dituliskan [PTX-13].

- b. MySQL, yaitu sebuah aplikasi *server database open source*.

MySQL adalah perangkat lunak sistem manajemen basis data (DBMS) yang *multiuser* dan *multi operation system* yang artinya dapat digunakan oleh banyak orang dalam satu waktu dan dapat digunakan seluruh OS secara gratis. MySQL juga dapat menjalankan perintahperintah *Structured Query Language* (SQL) untuk mengelola *database-database* yang ada di dalamnya [PTX-13][ASS-10].

- c. PHP, yaitu sebuah aplikasi pemrograman berbasis *web*.

Hypertext Preprocessor (PHP) merupakan bahasa pemrograman *server-side scripting* yang dapat ditanamkan atau disisipkan ke dalam *HyperText Markup Language* (HTML) [SNW-10]. PHP dipakai untuk memprogram situs *web* dinamis, karena PHP dapat berinteraksi dengan *database*, *file* dan *folder*. PHP dapat berjalan ataupun digunakan pada *operation system* yang berbeda-beda [TRY-07].

- d. Perl, yaitu sebuah aplikasi pemrograman *system*.

Perl adalah bahasa pemrograman untuk segala keperluan, dalam artian Perl adalah penanganan teks dan berbagai jalan pintasi untuk menyelesaikan persoalan-persoalan umum. Perl sering digunakan dalam program CGI (*Common Gateway Interface*) dan berbagai *protocol* internet lainnya [PTX-13].

2.9 PhoneGap

PhoneGap merupakan suatu framework produksi Nitobi yang digunakan dalam mengembangkan *mobile*. Nitobi sendiri telah diakuisisi oleh *Adobe Systems* pada tahun 2011. PhoneGap memungkinkan programmer perangkat lunak dapat membangun aplikasi untuk perangkat *mobile* menggunakan JavaScript, HTML5, dan CSS3, yang bukan merupakan bahasa spesifik suatu perangkat seperti Objective-C. Aplikasi yang dihasilkan adalah *hybrid*, yang berarti bahwa aplikasi tersebut tidak benar-benar *native* (karena seluruh pembuatan layout dilakukan melalui tampilan web dan bukan framework UI asli dari platform) atau bukan juga aplikasi murni berbasis web (karena aplikasi tersebut tidak hanya aplikasi web, tetapi dikemas sebagai aplikasi untuk distribusi dan memiliki akses ke API (*Application Programming Interface*) dari perangkat *native*). Bahkan dari versi 1.9 dan seterusnya memungkinkan adanya penggunaan potongan kode *native* dan *hybrid* secara bebas [PHG-14].

Perangkat lunak yang mendasari PhoneGap adalah *Apache Cordova*. Perangkat lunak ini sebelumnya hanya disebut "*PhoneGap*", kemudian "*Apache Callback*". *Apache Cordova* adalah perangkat lunak *open source*. Pada versi awal PhoneGap orang yang membuat aplikasi pada iOS harus memiliki komputer Apple, dan orang yang membuat aplikasi *Windows Mobile* untuk memiliki komputer yang menjalankan Windows. Setelah September 2012, PhoneGap memberikan layanan yang memungkinkan programmer untuk meng-upload HTML, CSS dan kode JavaScript ke "*cloud compiler*" yang menghasilkan aplikasi untuk setiap platform yang didukung. Pada saat ini PhoneGap mendukung pengembangan untuk beberapa sistem operasi yaitu Apple iOS, BlackBerry, Google Android, LG webOS, Microsoft Windows Phone (7 and 8), Nokia Symbian OS, Tizen (SDK 2.x), Bada, Firefox OS, dan Ubuntu Touch [PHG-14].

Penggunaan phonegap pada aplikasi berupa sebuah sintak javascript yang akan memudahkan akses sensor dan fitur khusus pada *handphone*. Contoh penggunaan phonegap pada aplikasi yaitu fungsi :

a. `navigator.app.exitApp()`

Fungsi tersebut digunakan untuk keluar dari aplikasi pada *handphone*. Pada aplikasi ini dijalankan fungsi tersebut saat terjadi *event backbutton* pada *handphone*.

b. `navigator.camera.getPicture`

Fungsi tersebut digunakan untuk mengambil gambar dan memiliki dua pilihan dalam menentukan sumber yang akan dipilih yaitu CAMERA dan PHOTOLIBRARY. Pilihan CAMERA akan menjalankan fungsi phonegap yaitu menjalankan kamera pada *handphone* dan menghentikan aplikasi secara sementara hingga fungsi kamera telah selesai digunakan. Fungsi PHOTOLIBRARY digunakan untuk membuka *image gallery* pada *handphone* dan menghentikan aplikasi secara sementara hingga gambar telah dipilih dan aplikasi akan dilanjutkan kembali.

c. `FileTransfer()`

Filetransfer banyak digunakan pada penelitian ini karena fungsinya dalam hal pengolahan gambar antara aplikasi dan *handphone*. Fungsi inilah menjadi pengganti ajax post pada Android yang memiliki versi 2.9 kebawah. Dua fungsi utama yang digunakan pada penelitian ini adalah *upload* dan *download*, *upload* akan mengunggah file yang dipilih ke media penyimpanan luar yaitu server sedangkan *download* akan mengunduh file yang dipilih ke media penyimpanan *handphone* yaitu *sdcard*.

d. `Contacts()`

Fungsi *contact* digunakan pada penelitian ini karena fungsinya yang dibutuhkan dalam hal penyimpanan *contact* seseorang yang menawarkan tempat tinggal dan fungsinya yang bisa melakukan panggilan telepon secara langsung. Penggunaan *save contact* dilakukan saat fungsi `navigator.contacts.create()` sebagai media

menambahkan parameter *contact* dipanggil dan `contact.save()` akan melakukan penyimpanan pada handphone.

2.10 Java Script Object Notation (JSON)

JSON adalah sebuah format dalam pertukaran data yang terjadi pada sistem. Aplikasi utamanya terdapat pada pemrograman aplikasi web AJAX dengan berperan sebagai alternatif terhadap penggunaan tradisional format XML.

Kelebihan JSON adalah ukuran lebih kecil dibanding XML, efeknya adalah transfer data lebih cepat dan lebih hemat. JSON adalah format data bawaan di Javascript, artinya jika data dari server di kirim ke client, dan client menggunakan javascript, maka tidak perlu library tambahan untuk memprosesnya. Dibanding XML, format JSON lebih sederhana.

Format JSON yang digunakan pada penelitian ini adalah:

```
[
  {"data": "result", "data": "result"},
  { "data": "result", "data": "result"},
  { "data": "result", "data": "result"}
]
```

2.11 Web Service REST

REST adalah sebuah arsitektur yang memudahkan penggunaanya untuk mendesain *web service* yang fokus pada sumber daya sistem, yaitu bagaimana sumber daya dikirim melalui HTTP oleh berbagai klien yang ditulis dalam bahasa yang berbeda [ARO-08].

Karakteristik utama dari REST adalah penggunaan metode pada HTTP secara eksplisit yang dapat dipetakan seperti proses CRUD yaitu *create*, *retrieve*, *update* dan *delete*. Untuk membuat sumber daya / *resource* pada server maka digunakanlah metode POST. Untuk mengambil sumber daya / *resource* pada server maka digunakan metode GET. Untuk merubah sumber daya / *resource* pada server maka digunakan metode PUT dan untuk menghapus sumber daya / *resource* pada server digunakan metode DELETE.

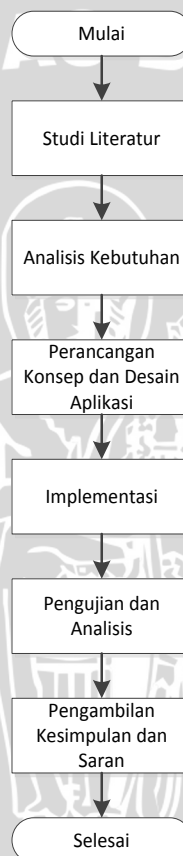
Server bertipe RESTful akan meningkatkan performa dari aplikasi karena server bertipe REST akan merubah protokol HTTP yang awalnya berupa *stateful* menjadi *stateless*. Struktur *stateless* akan mempercepat pengiriman data karena tidak diperlukan jaminan data akan berhasil dikirim seperti pada struktur *stateful*. REST memiliki beberapa keuntungan yang pertama yaitu cepat, karena tidak dibutuhkan pemrosesan data lebih lanjut pada server. Kedua tidak dibutuhkan tools lain untuk berinteraksi dengan web service karena REST dapat mengirim data dengan berbagai macam bentuk tidak seperti pada struktur SOAP yang harus menggunakan XML.



BAB III

METODE PENELITIAN

Pada bab ini dijelaskan langkah-langkah yang dilakukan dalam proses perancangan, implementasi dan pengujian dari aplikasi perangkat lunak yang akan dikembangkan. Diagram alir dari langkah-langkah tersebut yaitu:



Gambar 3.1 Flowchart tahapan kegiatan penelitian

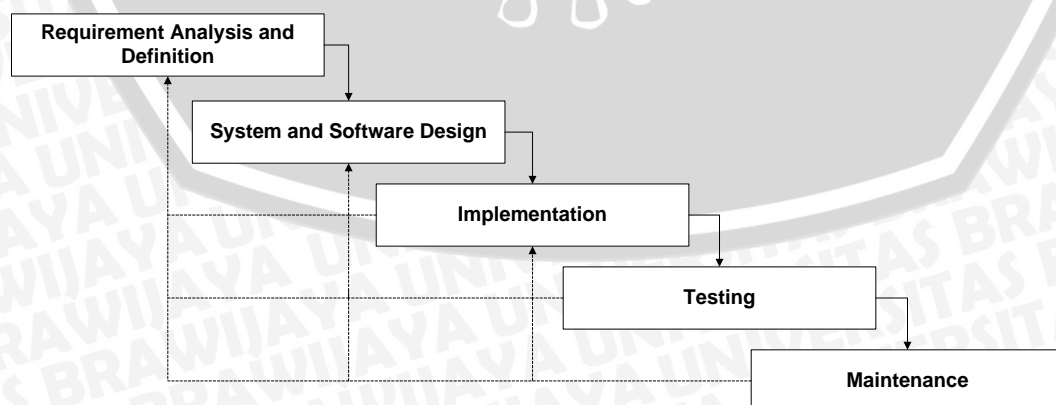
3.1 Studi Literatur

Studi Literatur adalah metode praktek yang dilakukan dengan mempelajari dan mengumpulkan data-data dari berbagai literatur yang dapat berasal dari buku, paper dan dokumentasi dari internet yang berkaitan dengan teori yang akan diterapkan dalam penelitian. Hal ini menjelaskan dasar teori yang digunakan dalam penyusunan skripsi. Teori-teori pendukung tersebut meliputi:

- Kebutuhan *backpacker* dan alasan *backpacking*

- *HyperText Markup Language (HTML)*
- *Cascading Style Sheets (CSS)*
- *javascript*
- JQuery
- JQuery Mobile
- *Hypertext Preprocessor (PHP)*
- Aptana Studio 3
- XAMPP
- PhoneGap
- JSON

Model proses perangkat lunak yang diimplementasikan adalah model sekuensial linier (*classic life cycle* atau *waterfall model*) merupakan paradigma rekayasa perangkat lunak yang paling banyak dipakai dan digunakan. Model ini mengusulkan sebuah pendekatan perkembangan perangkat lunak yang sistematis dan sekuensial yang dimulai pada tingkat dan kemajuan sistem pada seluruh analisis, desain, kode, pengujian dan pemeliharaan. Kelebihan dari *waterfall model* adalah prosesnya mudah dipahami, struktur sistem jelas, sebuah fase dijalankan setelah fase lainnya selesai [BYA-11]. Proses pengembangan ini digunakan karena kebutuhan pengguna telah dipahami dan kemungkinan terjadi perubahan kebutuhan pengguna kecil. Tahapan-tahapan dalam pembuatan *waterfall model* dijelaskan dalam Gambar 3.2.



Gambar 3.2 Diagram Alir Pemodelan *Waterfall*

Penjelasan mengenai gambar pemodelan *waterfall* dilakukan sebagai berikut:

1. *Requirement analysis and definition*

Pada tahapan ini dilakukan pengumpulan kebutuhan sistem pencari penginapan untuk backpacker yaitu kebutuhan pengguna, kebutuhan perangkat keras, perangkat lunak dan basis data.

2. *System and Software Design*

Pada tahap ini mulai dilakukan perancangan sistem sesuai dengan analisis kebutuhan pengguna. Hasilnya berupa diagram UML seperti *use case*, *class diagram*, *sequential diagram* dan diagram hubungan entitas (ERD)

3. *Implementation*

Pada tahap ini dibuatlah aplikasi *mobile* berdasarkan perancangan yang telah dibuat sesuai dengan kebutuhan pengguna. Aplikasi dibuat menggunakan pendekatan *hybrid* menggunakan HTML, javascript dan JQuery mobile serta melakukan integrasi database dengan *web service* menggunakan format JSON dan teknologi *service REST*.

4. *Testing*

Pada tahap ini dilakukan pengujian sebelum sistem dipublikasikan. Pengujian dilakukan menggunakan metode *black box* dan *white box*. Uji *black box* yang digunakan adalah pengujian validasi, komparabilitas dan *user acceptance test*, sementara uji *white box* akan digunakan uji unit.

3.2 Analisis Kebutuhan

Analisis kebutuhan bertujuan untuk mendapatkan semua kebutuhan yang diperlukan dari sistem yang akan dibangun. Analisis kebutuhan dilakukan dengan mengidentifikasi semua kebutuhan (*requirements*) sistem dan siapa saja yang terlibat di dalamnya. Analisis juga dilakukan untuk mengetahui kondisi lapangan yang ada sehingga dapat diketahui implementasi perangkat lunak yang akan digunakan. Metode analisis yang digunakan adalah *Object Oriented Analysis*

dengan menggunakan bahasa pemodelan UML (*Unified Modeling Language*). *Use case diagram* digunakan untuk mendeskripsikan kebutuhan-kebutuhan dan fungsionalitas sistem dari perspektif *end-user*. Analisis kebutuhan dilakukan dengan mengidentifikasi semua kebutuhan (*requirements*) perangkat lunak yang kemudian akan dimodelkan dalam *use case diagram*.

Tahap analisis kebutuhan terdiri atas empat langkah yaitu melakukan penjabaran tentang gambaran umum aplikasi, melakukan proses identifikasi aktor yang terlibat dalam aplikasi, membuat daftar kebutuhan pengguna menggunakan pemodelan diagram *use case*. Proses analisis kebutuhan ini diawali dengan penjabaran gambaran umum aplikasi *rooamate*, identifikasi aplikasi *rooamate*, penjabaran tentang daftar kebutuhan dan kemudian memodelkannya ke dalam diagram *use case*. Analisis kebutuhan ini bertujuan untuk menggambarkan kebutuhan-kebutuhan yang harus disediakan oleh sistem agar dapat memenuhi kebutuhan pengguna.

Aplikasi yang akan dibangun diarahkan menjadi *generic product* sesuai kebutuhan pengguna yang telah dianalisa. *Generic product* adalah *software* yang dapat melakukan banyak fungsi dan tidak dibatasi untuk kebutuhan khusus pengguna. Keuntungan pengembangan yang mengarah kepada *generic product* adalah mudah dalam implementasinya dan tidak dibutuhkan perubahan khusus pada aplikasi.

3.3 Perancangan Konsep dan Desain Aplikasi

Perancangan perangkat lunak dilakukan setelah semua kebutuhan perangkat lunak didapatkan melalui tahap analisis kebutuhan. Pada proses perancangan aplikasi terdapat 3 (tiga) tahapan yaitu, tahap pertama adalah perancangan umum sistem, proses analisis kebutuhan dan proses perancangan perangkat lunak. Tahap perancangan umum sistem menjelaskan mengenai gambar proses kerja sistem secara umum. Tahap proses analisis kebutuhan akan menjelaskan tentang gambaran umum aplikasi, identifikasi aktor, analisis kebutuhan fungsional dan non-fungsional. Sedangkan pada proses perancangan perangkat lunak memiliki empat langkah, yaitu perancangan arsitektural, perancangan basis data yang direpresentasikan

menggunakan *Entity Relationship Diagram* (ERD), pemodelan *sequence diagram* dan perancangan antarmuka pengguna dari aplikasi.

3.4 Implementasi

Sistem akan dibangun menggunakan bahasa pemrograman HTML, CSS dan JavaScript dengan memanfaatkan beberapa perangkat lunak yaitu:

- a. *Aptana Studio 3* : Sebagai perangkat lunak yang digunakan untuk mengembangkan aplikasi
- b. *DBMS MySQL* : Sebagai *basis data* manajemen sistem dari aplikasi yang dibuat
- c. *XAMPP 1.7.7* : Menghubungkan antara aplikasi yang dibuat dengan *basis data MySQL*
- d. *jQuery Mobile 1.4.0* : Sebagai *framework* yang digunakan dalam mengembangkan aplikasi
- e. *PhoneGap* : Sebagai *framework* untuk aplikasi *mobile* yang *cross platform* dengan menggunakan bahasa HTML5, Javascript dan CSS.
- f. Pada tahap akhir dilakukan implementasi simulasi pada *hardware* secara langsung dari Eclipse menggunakan *Android Develepmoent Tools (ADT)*.

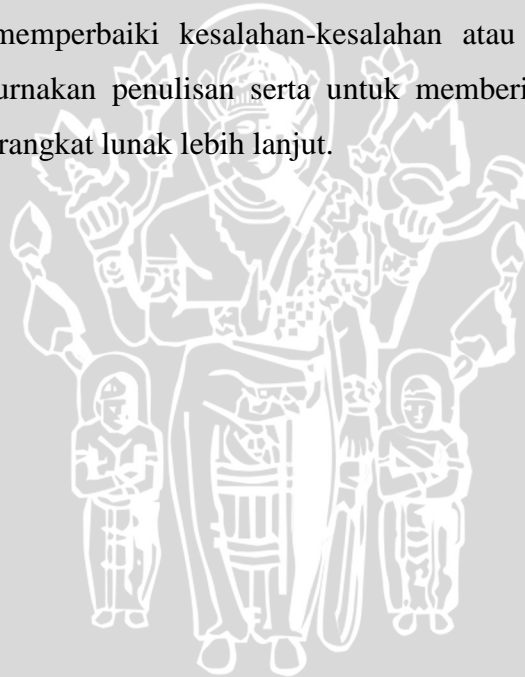
3.5 Pengujian dan Analisis

Pengujian merupakan bagian yang penting dalam siklus pembangunan perangkat lunak. Tujuan dari pengujian adalah untuk menjamin perangkat lunak yang dibangun sesuai dengan spesifikasi yang dibutuhkan oleh pengguna, yaitu mampu mempresentasikan kajian pokok dari spesifikasi, analisis, perancangan dan pengkodean dari perangkat lunak itu sendiri. Pengujian dilakukan dengan menggunakan metode *Black-box testing* dan metode *White-box testing*. Metode *Black-box testing* digunakan pada tahap pengujian validasi dan kompatibilitas aplikasi yang telah dibuat. Pengujian kompatibilitas dilakukan untuk mengetahui bagaimana kompatibilas perangkat lunak ketika berjalan pada Android dan iOS. Pada proses pengujian *White-box testing* dilakukan pengujian unit dengan menggunakan tools QUnit. Dengan menggunakan QUnit maka akan dilakukan pengujian pada setiap detail fungsi yang ada secara terperinci dan akan didapatkan hasil berupa sukses atau tidaknya tiap modul atau fungsi javascript yang ada pada

aplikasi. Dilakukan juga pengujian UAT (*User Acceptance Test*) pada tahap terakhir dilakukan untuk melihat kebutuhan *usability* pada sistem apakah dapat diterima oleh pengguna dengan baik. Analisis juga dilakukan untuk mengetahui hasil dari pengujian perangkat lunak sehingga dapat didapatkan kesimpulan dari pengembangan perangkat lunak yang telah dilakukan.

3.6 Pengambilan Kesimpulan dan Saran

Pengambilan kesimpulan dilakukan setelah semua tahapan perancangan perangkat lunak, implementasi perangkat lunak, dan pengujian perangkat lunak telah selesai dilakukan. Kesimpulan diambil dari hasil pengujian dan analisis terhadap sistem yang dibangun. Tahap terakhir dari penulisan adalah saran yang dimaksudkan untuk memperbaiki kesalahan-kesalahan atau kekurangan yang terjadi dan menyempurnakan penulisan serta untuk memberikan pertimbangan atas pengembangan perangkat lunak lebih lanjut.



BAB IV

PERANCANGAN DAN IMPLEMENTASI

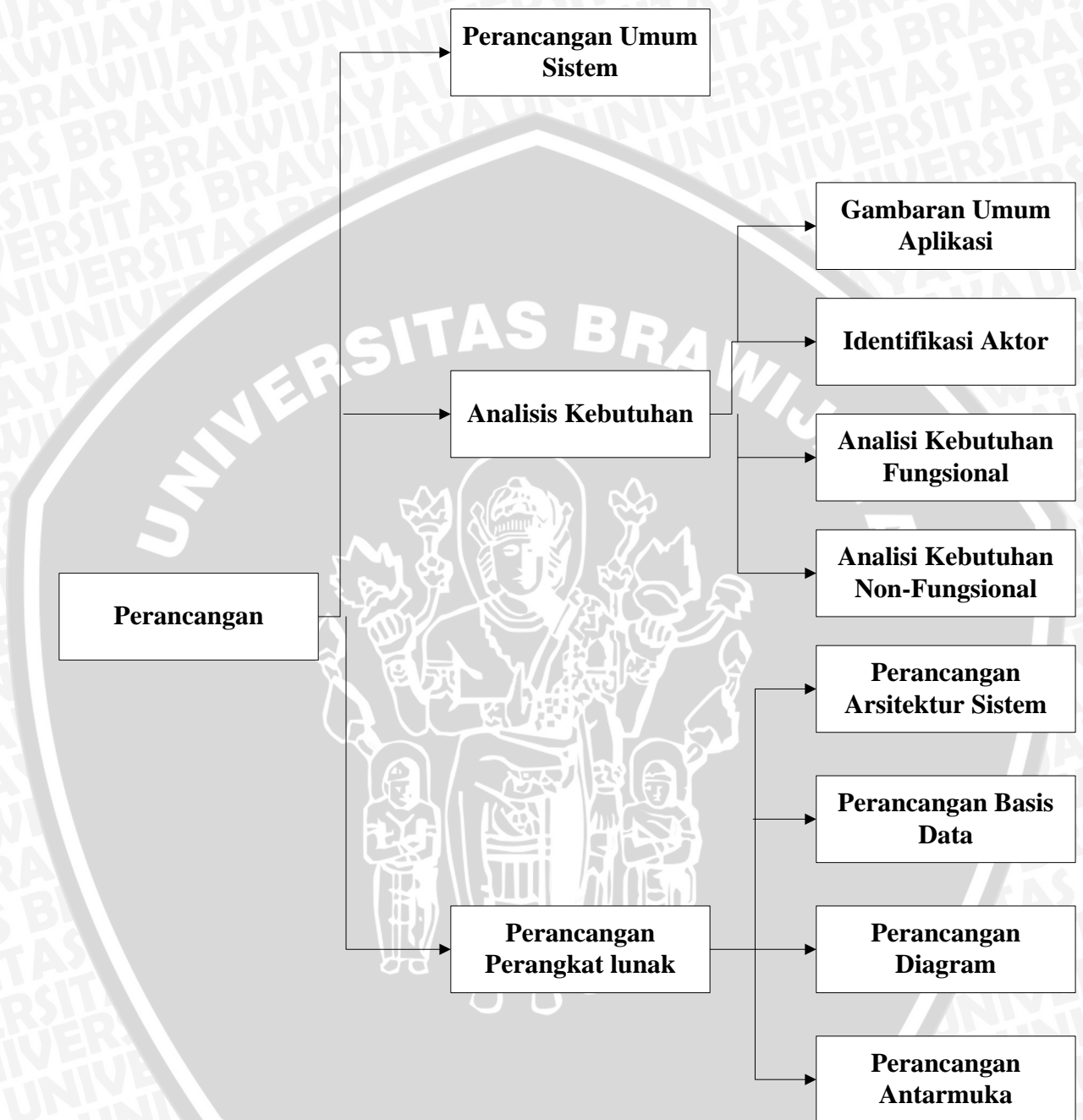
Bab ini membahas mengenai perancangan dan implementasi aplikasi. Perancangan yang dilakukan meliputi 3 (tiga) tahap yaitu perancangan umum sistem, proses analisis kebutuhan dan perancangan perangkat lunak. Sedangkan pembahasan implementasi terdiri atas penjelasan tentang spesifikasi lingkungan implementasi, batasan-batasan dalam implementasi, implementasi basis data, implementasi *class* dan *assets* pada file program, implementasi algoritma dan implementasi antarmuka aplikasi.

4.1 Perancangan

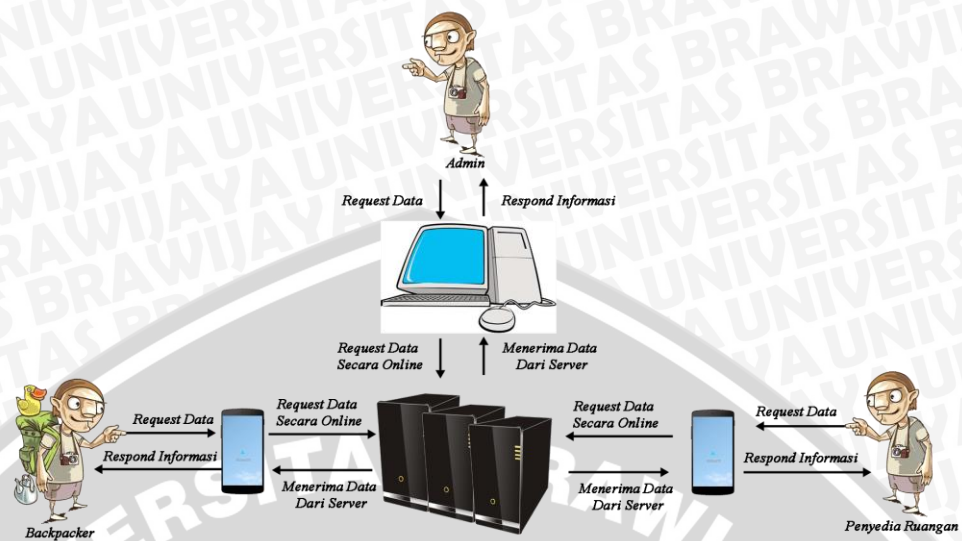
Proses perancangan aplikasi dibagi menjadi 3 (tiga) bagian, tahap pertama adalah perancangan umum sistem, tahap kedua adalah proses analisis kebutuhan dan tahap ketiga adalah perancangan perangkat lunak. Tahap perancangan umum sistem menjelaskan mengenai gambar proses kerja sistem secara umum. Tahap analisis kebutuhan terdiri atas empat langkah yaitu melakukan penjabaran tentang gambaran umum aplikasi, melakukan proses identifikasi aktor yang terlibat dalam aplikasi, membuat daftar kebutuhan pengguna dengan menganalisa kebutuhan fungsional dan kebutuhan non fungsional. Pada proses perancangan perangkat lunak memiliki empat langkah, yaitu perancangan arsitektural, pemodelan *sequence diagram*, perancangan basis data dan perancangan antarmuka pengguna dari aplikasi. Tahapan perancangan ditunjukkan dalam Gambar 4.1.

4.1.1 Perancangan Umum Sistem

Perancangan umum sistem merupakan tahapan awal dari perancangan perangkat lunak yaitu pendefinisian dari kebutuhan-kebutuhan fungsional yang menggambarkan bagaimana suatu sistem dibentuk. Perancangan sistem dilakukan untuk merepresentasikan arsitektur sistem yang akan dibuat secara umum. Gambar 4.2 berikut ini akan menunjukkan perancangan umum dari sistem yang akan dibuat.



Gambar 4.1 Diagram Pohon Perancangan



Gambar 4.2 Diagram Blok Perancangan Umum Sistem

Sistem akan berjalan menggunakan sebuah webservice sebagai media untuk melakukan pertukaran data. Webservice menggunakan server bertipe RESTful yang menggunakan protokol HTTP untuk menyediakan data. Request dikirim menuju server yang bertipe RESTful dengan metode POST dan akan mengirim kembali data informasi dalam bentuk JSON.

Server bertipe RESTful akan meningkatkan performa dari aplikasi karena server bertipe REST akan merubah protokol HTTP yang awalnya berupa *stateful* menjadi *stateless*. Struktur *stateless* akan mempercepat pengiriman data karena tidak diperlukan jaminan data akan berhasil dikirim seperti pada struktur *stateful*. REST memiliki beberapa keuntungan yang pertama yaitu cepat, karena tidak dibutuhkan pemrosesan data lebih lanjut pada server. Kedua tidak dibutuhkan tools lain untuk berinteraksi dengan web service karena REST dapat mengirim data dengan berbagai macam bentuk tidak seperti pada struktur SOAP yang harus menggunakan XML. Ketiga REST adalah struktur yang paling mendekati teknologi web yang lain terutama pada desain.

Arsitektur jaringan yang digunakan berupa 3-tier yaitu hubungan pertukaran data antara klien dan database dijembatani oleh sebuah pihak ketiga. Klien adalah tampilan GUI pada user dan pihak ketiga yang digunakan untuk

mengolah data dan berhubungan dengan server digunakan bahasa PHP. Pihak ketiga akan menambah performa aplikasi dan menambah tingkat keamanan dari sistem.

4.1.2 Analisis Kebutuhan

Proses analisis kebutuhan ini diawali dengan penjabaran gambaran umum aplikasi, identifikasi aktor, penjabaran tentang kebutuhan fungsional yang akan di modelkan dalam bentuk *use case diagram* serta kebutuhan non fungsional. Analisis kebutuhan ini bertujuan untuk menggambarkan kebutuhan-kebutuhan yang harus disediakan oleh sistem agar dapat memenuhi kebutuhan pengguna.

4.1.2.1 Gambaran Umum Aplikasi

Pembahasan gambaran umum aplikasi terdiri atas dua bagian, yaitu deskripsi umum aplikasi dan lingkungan aplikasi. Aplikasi yang dirancang akan diberi nama aplikasi *mobile rooamate* yang diambil dari nama aplikasi yang akan dibuat yaitu *rooamate*.

4.1.2.1.1 Deskripsi Aplikasi

Aplikasi *mobile rooamate* adalah aplikasi yang dapat digunakan sebagai media atau tempat berinteraksinya pengguna sebagai *backpacker* atau pencari tempat menginap dan pengguna yang ingin menyewakan tempat tinggalnya. Tempat tinggal yang ingin disewakan dalam hal ini adalah sebuah ruangan atau kamar. Bagi pengguna yang mencari tempat tinggal atau *backpacker*, diharapkan dapat mencari tempat tinggal dengan fitur-fitur seperti pencarian berdasarkan daerah dan parameter lain, pencarian berdasarkan peta dengan lokasi pengguna secara *real time*, dan pencarian tempat tinggal terdekat berdasarkan jarak dari posisi pengguna. Bagi pengguna yang menawarkan tempat tinggalnya, diharapkan dapat menawarkan tempat tinggalnya dengan kemudahan menambahkan posisi berdasarkan posisi saat itu ataupun berdasarkan peta. Pengguna yang telah terdaftar diharapkan dapat memberikan penilaian kepada sebuah penawaran tempat tinggal. Kedua pengguna diharapkan dapat saling bertukar pendapat melalui komentar yang ada disetiap penawaran tempat tinggal. Pengguna diharapkan dapat melihat detail dari penawaran tempat tinggal yang dipilih seperti

informasi umum penawaran tempat tinggal, gambar tempat tinggal, peta yang langsung menunjukkan arah ke tempat tujuan dan rating.

4.1.2.1.2 Lingkungan Aplikasi

Aplikasi *mobile roommate* ini membutuhkan suatu lingkungan yang digunakan sebagai tempat berjalannya aplikasi. Secara keseluruhan aplikasi *mobile* ini berbasis *hybrid mobile application*, sehingga membutuhkan sebuah *device* untuk menjalankan aplikasi tersebut. Dengan penggunaan teknologi *mobile smartphone Android*, aplikasi *mobile* ini bisa dijalankan di semua lingkungan *device* Android. Lingkungan lain yang dibutuhkan yaitu koneksi internet, koneksi tersebut dapat melalui jaringan *wifi* ataupun *packet mobile data* yang langsung dari *handphone*. Dibutuhkan juga setting untuk aktifasi *GPS location* sesuai *device* Android masing-masing yang digunakan untuk mendapatkan lokasi pengguna. Koneksi internet digunakan untuk mendapatkan data-data yang dibutuhkan untuk ditampilkan kepada pengguna.

4.1.2.2 Identifikasi Aktor

Tahap ini adalah tahap untuk melakukan identifikasi terhadap aktor-aktor yang akan berinteraksi dengan aplikasi. Pada Tabel 4.1 memperlihatkan aktor-aktor yang terlibat beserta penjelasannya.

Tabel 4.1 Identifikasi Aktor

Aktor	Deskripsi
Pengguna	Terdapat 2 pengguna yaitu backpacker dan orang yang memiliki ruangan untuk disewakan. Backpacker dan pemilik tempat tinggal dapat menggunakan aplikasi untuk melihat dan menggunakan fitur pencarian berdasarkan daerah dengan berbagai parameter, berdasarkan lokasi terkini dan tempat tinggal terdekat berdasarkan lokasi terkini. Pengguna juga dapat memberikan <i>vote</i> dan memberikan komentar pada <i>post</i> yang dipilih. Pengguna yang telah memiliki akun dapat menawarkan tempat tinggalnya.
Administrator	Administrator atau admin adalah orang yang dapat melakukan pengelolaan data pada aplikasi. Seorang administrator dapat membuat, mengubah dan menghapus data dari pengguna

	sehingga data aplikasi tetap terjaga dan terbebas dari konten – konten yang tidak diinginkan. Admin juga dapat melakukan tindakan pada akun pengguna seperti melakukan blokir pada pengguna tertentu. Dengan adanya administrator maka aplikasi dapat berjalan dengan optimal dan moderasi aplikasi dapat dilakukan dengan mudah.
--	---

4.1.2.3 Analisis Kebutuhan Fungsional

Daftar kebutuhan terdiri dari kebutuhan fungsional dan kebutuhan non-fungsional. Pada daftar kebutuhan fungsional akan dispesifikasikan yaitu spesifikasi kebutuhan fungsional pengguna yang di tunjukkan pada Tabel 4.2 dengan pernomer menggunakan SRS (*Software Requirement Spesifikation*).

Tabel 4.2 Spesifikasi Kebutuhan Fungsional Pengguna

Nomor SRS	Kebutuhan	Use case
SRS_001_01	Aplikasi harus menyediakan fasilitas untuk menampilkan data pengguna dan fasilitas untuk merubah data profil pengguna, sehingga membantu dalam hal pengolahan data profil pribadi	Kelola profil pribadi
SRS_002_01	Aplikasi harus menyediakan fasilitas untuk lihat tempat menginap dan fasilitas untuk merubah data tempat menginap, sehingga membantu pengguna untuk melakukan perubahan data yang diperluka	Kelola penawaran tempat menginap
SRS_003_01	Aplikasi harus menyediakan fasilitas untuk mencari data – data tempat menginap melalui fasilitas pencarian cepat ataupun pencarian yang lebih detail, sehingga pengguna dapat dengan mudah menemukan data dari tempat tinggal yang diinginkan	Lihat tempat menginap melalui pencarian berdasarkan parameter
SRS_003_02	Aplikasi harus menyediakan fasilitas untuk mencari data – data tempat	Lihat tempat menginap melalui pencarian

	menginap melalui fasilitas pencarian pada peta berdasarkan lokasi pengguna, sehingga pengguna dapat dengan mudah menemukan data dari tempat tinggal yang diinginkan	berdasarkan peta
SRS_003_03	Aplikasi harus menyediakan fasilitas untuk mencari data – data tempat menginap melalui fasilitas pencarian berdasarkan lokasi terdekat dari pengguna, sehingga pengguna dapat dengan mudah menemukan data dari tempat tinggal yang diinginkan	Lihat tempat menginap melalui pencarian berdasarkan lokasi terdekat
SRS_004_01	Aplikasi harus menyediakan fasilitas untuk melihat detail dari tempat menginap yang telah dipilih oleh pengguna yaitu detail biaya, alamat, nomer telepon, gambar, detail lokasi, rating hingga komentar yang ada, sehingga membantu pengguna dalam memperoleh informasi yang dibutuhkan untuk memilih tempat menginap	Lihat detail tempat menginap
SRS_004_02	Aplikasi harus menyediakan fasilitas untuk melihat komentar yang ditinggalkan oleh pengguna lain maupun pengguna juga dapat memberikan komentar terhadap tempat tinggal yang ada, sehingga pengguna lain dapat dengan mudah mengetahui <i>feedback</i> dari pengguna lain terhadap tempat tinggal tersebut	Kelola komentar
SRS_005_01	Aplikasi harus menyediakan fasilitas untuk melihat deskripsi aplikasi sehingga pengguna paham kegunaan aplikasi	Lihat deskripsi aplikasi
SRS_006_01	Aplikasi harus menyediakan fasilitas untuk melakukan permintaan pemesanan pada penawaran tempat tinggal tertentu berdasarkan tanggal, bulan dan tahun, dan juga fasilitas untuk melakukan pengelolaan permintaan untuk menginap	Kelola permintaan / <i>request</i> tanggal pemesanan tempat tinggal

Tabel di bawah ini akan menunjukkan spesifikasi kebutuhan fungsional admin yang di tunjukkan pada Tabel 4.3 dengan pernomer menggunakan SRS (*Software Requirement Spesification*).

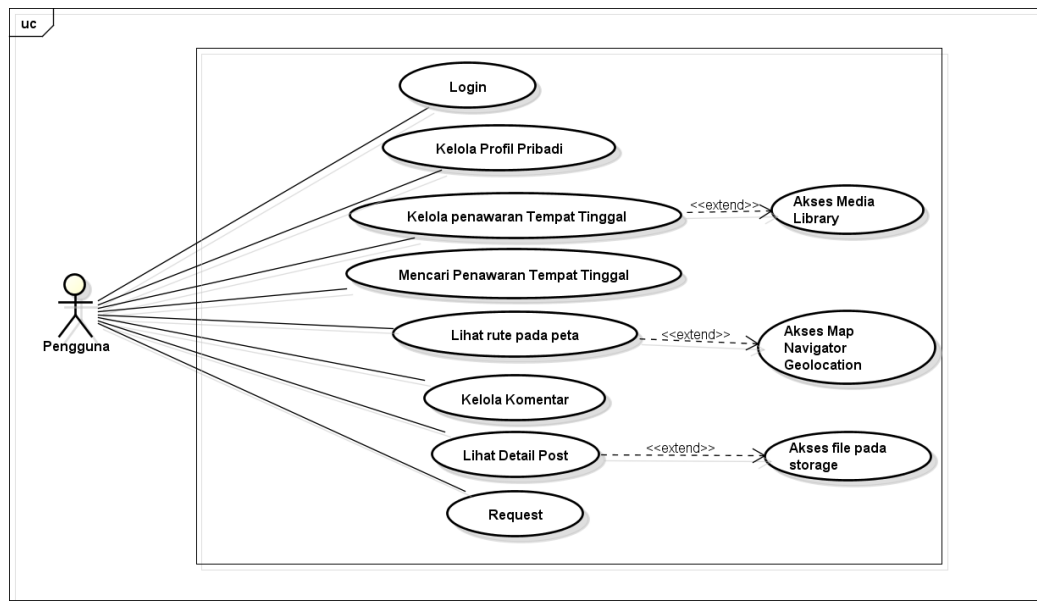
Tabel 4.3 Spesifikasi Kebutuhan Fungsional Admin

Nomor SRS	Kebutuhan	Use case
SRS_001_01	Aplikasi harus menyediakan fasilitas untuk lihat tempat menginap, fasilitas untuk merubah data tempat menginap, dan melakukan hapus data tempat menginap sehingga membantu admin untuk melakukan perubahan data yang diperlukan	Kelola penawaran tempat menginap
SRS_002_01	Aplikasi harus menyediakan fasilitas untuk lihat data akun pengguna, fasilitas untuk merubah data aku pengguna, dan melakukan pemblokiran akun sehingga membantu admin untuk melakukan moderasi pada akun pengguna	Kelola akun pengguna
SRS_003_01	Aplikasi harus menyediakan fasilitas untuk lihat komentar, dan melakukan hapus data komentar pada tempat menginap sehingga membantu admin untuk melakukan moderasi pada komentar	Kelola komentar

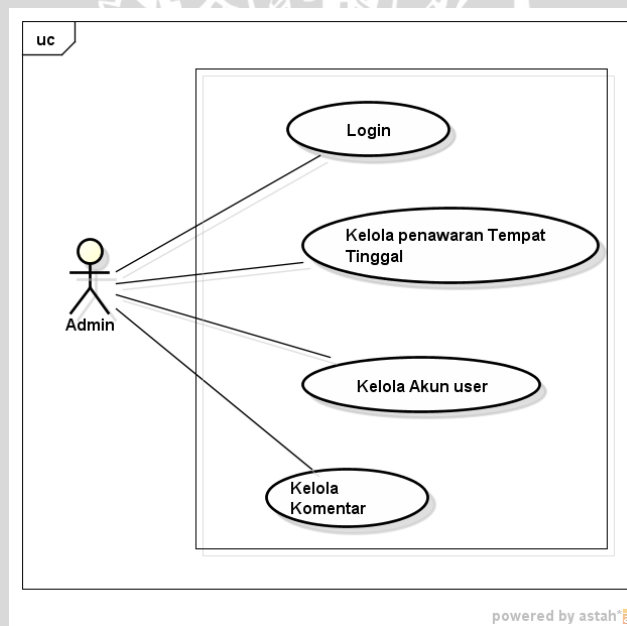
Selanjutnya daftar kebutuhan fungsional akan lebih dijabarkan menggunakan diagram *use case*.

1. Diagram Use Case

Diagram *use case* adalah salah satu diagram untuk memodelkan aspek perilaku sistem. Diagram *use case* menunjukkan sekumpulan *use case*, aktor, dan hubungannya. *Use case* merupakan fungsionalitas dari sistem yang diinisialisasi oleh aktor, Diagram *use case* untuk pengguna dan admini aplikasi ditunjukkan dalam Gambar 4.3 dan Gambar 4.4 berikut ini:



Gambar 4.3 Diagram Use case Pengguna



Gambar 4.4 Diagram Use case Admin

2. Skenario Use Case

Masing-masing *use case* yang terdapat pada diagram *use case* akan dijabarkan lebih mendetail dalam skenario *use case*. Di dalam skenario *use case*, akan diberikan uraian nama *use case*, aktor yang berhubungan dengan *use case* tersebut, tujuan dari *use case*, deskripsi global tentang *use case*, kondisi awal

yang harus dipenuhi dan kondisi akhir yang diharapkan setelah berjalannya fungsional *use case*. Selain itu juga akan diberikan ulasan yang berkaitan dengan tanggapan dari sistem atas suatu aksi yang diberikan oleh aktor.

a. Skenario *Use Case* Pengguna

Berikut akan dijabarkan mengenai skenario *use case* dari pengguna berdasarkan *use case* pengguna:

a. Skenario *Use Case* Kelola Profil Pribadi

Kebutuhan fungsional yang disediakan pada sistem adalah kebutuhan untuk mengelola profil pribadi pada sistem. Kebutuhan tersebut direpresentasikan oleh *use case* kelola profil pribadi pengguna, Tabel 4.4 merupakan skenario *use case* kelola profil pribadi.

Tabel 4.4 Skenario *Use Case* Kelola Profil Pribadi

Nama	Kelola Profil Pribadi
Kode SRS	SRS_001_01
Tujuan	Mengelola profil pribadi
Deskripsi (Brief Description)	<i>Use case</i> ini memungkinkan pengguna untuk mengelola data profil pribadi pada sistem. Mengelola yang dimaksudkan pada <i>use case</i> adalah dapat melihat dan mengubah informasi data profil pribadi dari sistem.
Aktor	Pengguna
Kondisi Awal (Pre-Conditions)	Aktor harus membuka dan melakukan login pada aplikasi terlebih dahulu sebelum <i>use case</i> dimulai. Aktor harus masuk pada menu profil pribadi
Flow of Events	
Alur Utama (Basic Flow)	
<ol style="list-style-type: none"> 1. Aktor memilih fungsi yang akan dikerjakan yaitu ubah profil pribadi. 2. Setelah aktor memilih pilihan informasi maka <i>sub flow</i> “Ubah Profil Pribadi” 	

dijalankan.	
Alur Bagian (Sub Flow)	
Ubah Profil Pribadi	<ol style="list-style-type: none"> 1. Sistem menampilkan informasi data profil pribadi 2. Aktor membuat perubahan yang diinginkan pada informasi data profil pribadi. Informasi sesuai dengan yang ada pada profil pribadi 3. Setelah aktor mengubah informasi yang diperlukan, sistem mengubah data profil pribadi.
Kondisi Akhir (Post-Conditions)	Informasi data profil pribadi pengguna akan berubah. Jika tidak, keadaan sistem tidak berubah.

b. Skenario Use Case Kelola Penawaran Tempat Menginap

Kebutuhan fungsional yang disediakan pada sistem adalah kebutuhan untuk mengelola penawaran tempat menginap pada sistem. Kebutuhan tersebut direpresentasikan oleh *use case* kelola penawaran tempat menginap pengguna, Tabel 4.5 merupakan skenario *use case* kelola penawaran tempat menginap.

Tabel 4.5 Skenario *Use Case* Kelola Penawaran Tempat Menginap

Nama	Kelola penawaran tempat menginap
Kode SRS	SRS_002_01
Tujuan	Mengelola penawaran tempat menginap
Deskripsi (Brief Description)	<i>Use case</i> ini memungkinkan pengguna untuk mengelola data penawaran tempat menginap pada sistem. Mengelola yang dimaksudkan pada <i>use case</i> adalah dapat melihat dan mengubah informasi data tempat menginap dari sistem diantaranya judul, alamat, nomor telepon, harga, lokasi dan gambar yang berhubungan.

Aktor	Pengguna
Kondisi Awal (Pre-Conditions)	Aktor harus membuka dan melakukan login pada aplikasi terlebih dahulu sebelum <i>use case</i> dimulai. Aktor harus masuk pada menu <i>offer place</i> untuk melakukan penawaran dan <i>edit post</i> untuk melakukan perubahan.
Flow of Events	
Alur Utama (Basic Flow)	
<p><i>Use case</i> ini dimulai ketika aktor ingin menambah, mengubah, atau menghapus informasi dari penawaran tempat tinggal</p> <ol style="list-style-type: none"> 1. Aktor melakukan penambahan atau mengubah penawaran tempat tinggal. 2. Setelah aktor memilih pilihan informasi maka <i>sub flow</i> dijalankan. <ol style="list-style-type: none"> a. Jika aktor memilih “Tambah penawaran tempat tinggal” maka <i>sub flow</i> Tambah penawaran tempat tinggal dijalankan. b. Jika aktor memilih “Ubah penawaran tempat tinggal” maka <i>sub flow</i> Ubah penawaran tempat tinggal dijalankan. 	
Alur Bagian (Sub Flow)	
Tambah Penawaran Tempat Tinggal	<ol style="list-style-type: none"> 1. Sistem mengizinkan pengguna memasukkan informasi data penawaran tempat tinggal. Informasi yang dimasukkan antara lain judul, harga, deskripsi, alamat, nomor telepon, lokasi tempat tinggal melalui peta ataupun lokasi pengguna, dan gambar pendukung. 2. Setelah pengguna memberikan informasi yang diminta, sistem memberikan kode yang unik untuk data penawaran yang baru. Data ini akan di tambahkan ke dalam sistem. 3. Sistem menyediakan data penawaran tempat tinggal dengan kode yang baru.
Ubah Penawaran Tempat Tinggal	<ol style="list-style-type: none"> 1. Sistem Meminta aktor untuk memilih penawaran tempat tinggal yang ada. 2. Sistem menampilkan detail data dan aktor harus menekan tombol <i>edit</i> untuk melakukan perubahan data. 3. Sistem menampilkan informasi penawaran dan aktor melakukan

	perubahan sesuai yang diinginkan. Informasi sesuai dengan yang ada pada <i>sub flow</i> tambah penawaran tempat tinggal. 4. Setelah aktor mengubah informasi yang diperlukan, sistem mengubah data penawaran tempat tinggal.
Kondisi Akhir (Post-Conditions)	Informasi data penawaran tempat tinggal akan berubah. Jika tidak, keadaan sistem tidak berubah.

c. Skenario *use Case* Lihat Tempat Menginap Melalui Pencarian Berdasarkan Parameter

Kebutuhan fungsional yang disediakan pada sistem adalah kebutuhan untuk melihat tempat menginap yang ditawarkan pada sistem berdasarkan pencarian dari beberapa parameter. Kebutuhan tersebut direpresentasikan oleh *use case* cari data dengan berbagai parameter, Tabel 4.6 merupakan skenario *use case* cari data dengan berbagai parameter.

Tabel 4.6 Skenario *Use Case* Cari Data Dengan Berbagai Parameter

Nama	Cari Data Dengan Berbagai Parameter
Kode SRS	SRS_003_01
Tujuan	Mencari tempat menginap melalui pencarian cepat ataupun detail berdasarkan parameter yang disediakan sistem
Deskripsi (Brief Description)	<i>Use case</i> ini memungkinkan pengguna untuk mencari data tempat tinggal pada sistem. Mencari data dari berbagai parameter yang dimaksudkan pada <i>use case</i> adalah dapat mencari semua tempat tinggal yang diinginkan oleh pengguna melalui pencarian pada sistem
Aktor	Pengguna
Kondisi Awal (Pre-Conditions)	Aktor harus membuka aplikasi terlebih dahulu sebelum <i>use case</i> dimulai. Aktor harus masuk pada menu utama sistem

Flow of Events	
Alur Utama (Basic Flow)	
<p><i>Use case</i> ini dimulai ketika aktor ingin mencari informasi tempat tinggal pada sistem.</p> <ol style="list-style-type: none"> 1. Aktor mencari data tempat tinggal berdasarkan parameter. 2. Ketika aktor memilih salah satu penawaran tempat tinggal maka <i>sub flow</i> “Lihat detail <i>post</i>” dijalankan. 	
Alur Bagian (Sub Flow)	
Lihat Detail Post	<ol style="list-style-type: none"> 1. Sistem menampilkan informasi penawaran tempat tinggal sesuai penawaran tempat tinggal yang dipilih oleh aktor yaitu informasi judul, harga, deskripsi, alamat, nomor telepon, lokasi dan gambar pendukung. 2. Sistem menyediakan fasilitas komentar dan rating bagi aktor yang belum melakukan rating. 3. Setelah aktor selesai melihat detail post, <i>use case</i> berakhir.
Alur Alternatif (Alternative Flows)	
Informasi Data Penawaran Tempat Tinggal Tidak Tersedia	<p>Jika pada <i>Basic Flow</i> sistem tidak dapat menemukan informasi data penawaran tempat tinggal, maka sistem akan menampilkan pesan informasi tidak ditemukan penawaran tempat tinggal.</p>
Kondisi Akhir (Post-Conditions)	<p>Informasi data penawaran tempat tinggal akan muncul. Jika tidak, keadaan sistem tidak berubah.</p>

d. Skenario *Use Case* Lihat Tempat Menginap Melalui Pencarian Berdasarkan Peta

Kebutuhan fungsional yang disediakan pada sistem adalah kebutuhan untuk melihat tempat menginap yang ditawarkan berdasarkan pencarian dari peta yang terdapat pada sistem. Kebutuhan tersebut direpresentasikan oleh *use case* cari data berdasarkan peta, Tabel 4.7 merupakan skenario *use case* cari data berdasarkan peta.

Tabel 4.7 Skenario *Use Case* Cari Data Melalui Peta

Nama	Cari Data Melalui Peta
Kode SRS	SRS_003_02
Tujuan	Mencari tempat menginap melalui peta yang disediakan sistem
Deskripsi (Brief Description)	<i>Use case</i> ini memungkinkan pengguna untuk mencari data tempat tinggal pada sistem. Mencari data melalui peta yang dimaksudkan pada <i>use case</i> adalah dapat mencari semua tempat tinggal yang diinginkan oleh pengguna melalui peta yang menggunakan <i>google maps</i> berdasarkan lokasi aktual aktor.
Aktor	Pengguna
Kondisi Awal (Pre-Conditions)	Aktor harus membuka aplikasi terlebih dahulu sebelum <i>use case</i> dimulai. Aktor harus masuk pada menu utama sistem
Flow of Events	
Alur Utama (Basic Flow)	

Use case ini dimulai ketika aktor ingin mencari informasi tempat tinggal pada sistem.

1. Aktor mencari data tempat tinggal berdasarkan peta.
2. Ketika aktor memilih salah satu penawaran tempat tinggal maka *sub flow* "Lihat detail *post*" dijalankan.

Alur Bagian (*Sub Flow*)

Lihat Detail <i>Post</i>	<ol style="list-style-type: none"> 1. Sistem menampilkan informasi penawaran tempat tinggal sesuai penawaran tempat tinggal yang dipilih oleh aktor yaitu informasi judul, harga, deskripsi, alamat, nomor telepon, lokasi dan gambar pendukung. 2. Sistem menyediakan fasilitas komentar dan rating bagi aktor yang belum melakukan rating. 3. Setelah aktor selesai melihat detail <i>post</i>, <i>use case</i> berakhir.
---------------------------------	--

Alur Alternatif (*Alternative Flows*)

Informasi Data Penawaran Tempat Tinggal Tidak Tersedia	Jika pada <i>Basic Flow</i> sistem tidak dapat menemukan informasi data penawaran tempat tinggal, maka sistem akan menampilkan pesan informasi tidak ditemukan penawaran tempat tinggal.
Kondisi Akhir (<i>Post-Conditions</i>)	Informasi data penawaran tempat tinggal akan muncul. Jika tidak, keadaan sistem tidak berubah.

e. Skenario *Use Case* Lihat Tempat Menginap Melalui Pencarian Berdasarkan Lokasi Terdekat

Kebutuhan fungsional yang disediakan pada sistem adalah kebutuhan untuk melihat tempat menginap yang ditawarkan berdasarkan pencarian dari lokasi terdekat yang terdapat pada sistem. Kebutuhan tersebut direpresentasikan oleh *use case* cari data berdasarkan lokasi terdekat, Tabel 4.8 merupakan skenario *use case* cari data berdasarkan lokasi terdekat.

Tabel 4.8 Skenario *Use Case* Cari Data Berdasarkan Lokasi Terdekat

Nama	Cari Data Berdasarkan Lokasi Terdekat
Kode SRS	SRS_003_03
Tujuan	Mencari tempat menginap melalui lokasi terdekat aktor yang telah disediakan sistem
Deskripsi (Brief Description)	<i>Use case</i> ini memungkinkan pengguna untuk mencari data tempat tinggal pada sistem. Mencari data berdasarkan lokasi terdekat yang dimaksudkan pada <i>use case</i> adalah dapat mencari semua tempat tinggal yang diinginkan oleh pengguna berdasarkan lokasi terdekat berdasarkan lokasi aktual aktor.
Aktor	Pengguna
Kondisi Awal (Pre-Conditions)	Aktor harus membuka aplikasi terlebih dahulu sebelum <i>use case</i> dimulai. Aktor harus masuk pada menu utama sistem
Flow of Events	
Alur Utama (Basic Flow)	
<p><i>Use case</i> ini dimulai ketika aktor ingin mencari informasi tempat tinggal pada sistem.</p> <ol style="list-style-type: none"> 1. Aktor mencari data tempat tinggal berdasarkan lokasi terdekat. 2. Ketika aktor memilih salah satu penawaran tempat tinggal maka <i>sub flow</i> “Lihat detail <i>post</i>” dijalankan. 	

Alur Bagian (<i>Sub Flow</i>)	
Lihat Detail Post	<ol style="list-style-type: none"> 1. Sistem menampilkan informasi penawaran tempat tinggal sesuai penawaran tempat tinggal yang dipilih oleh aktor yaitu informasi judul, harga, deskripsi, alamat, nomor telepon, lokasi dan gambar pendukung. 2. Sistem menyediakan fasilitas komentar dan rating bagi aktor yang belum melakukan rating. 3. Setelah aktor selesai melihat detail post, <i>use case</i> berakhir.
Alur Alternatif (<i>Alternative Flows</i>)	
Informasi Data Penawaran Tempat Tinggal Tidak Tersedia	Jika pada <i>Basic Flow</i> sistem tidak dapat menemukan informasi data penawaran tempat tinggal, maka sistem akan menampilkan pesan informasi tidak ditemukan penawaran tempat tinggal.
Kondisi Akhir (<i>Post-Conditions</i>)	Informasi data penawaran tempat tinggal akan muncul. Jika tidak, keadaan sistem tidak berubah.

f. Skenario *Use Case* Lihat detail tempat menginap

Kebutuhan fungsional yang disediakan pada sistem adalah kebutuhan untuk melihat detail informasi dari penawaran tempat tinggal yang disediakan sistem. Kebutuhan tersebut direpresentasikan oleh *use case* lihat detail *post*, Tabel 4.9 merupakan skenario *use case* lihat detail *post*.

Tabel 4.9 Skenario *Use Case* Lihat Detail *Post*

Nama	Lihat Detail <i>Post</i>
Kode SRS	SRS_004_01
Tujuan	Melihat informasi yang terdapat pada penawaran tempat tinggal yang dipilih oleh aktor

Deskripsi (<i>Brief Description</i>)	<i>Use case</i> ini memungkinkan pengguna untuk melihat data dan informasi tempat tinggal pada sistem. Detail informasi yang tersedia pada sistem yaitu judul, harga, deskripsi tempat tinggal, alamat, nomor telepon, lokasi tempat tinggal yang langsung menunjukkan rutenya dari lokasi aktor dan gambar pendukung tempat tinggal.
Aktor	Pengguna
Kondisi Awal (<i>Pre-Conditions</i>)	Aktor harus membuka aplikasi terlebih dahulu dan memilih penawaran tempat tinggal melalui beberapa fitur pencarian sebelum <i>use case</i> dimulai.
Flow of Events	
Alur Utama (<i>Basic Flow</i>)	
<p><i>Use case</i> ini dimulai ketika aktor ingin melihat informasi detail tempat tinggal pada sistem.</p> <ol style="list-style-type: none"> 1. Aktor memilih penawaran tempat tinggal yang didapatkan setelah melakukan pencarian melalui fitur pencarian yang ada pada sistem. 2. Ketika aktor memilih tempat tinggal sistem akan menampilkan detail dari tempat tinggal mulai dari judul, harga, deskripsi, alamat, nomor telepon, rute pada map dan gambar – gambar pendukung dari tempat tinggal yang dipilih. <i>Use case</i> berakhir setelah aktor keluar dari detail <i>post</i>. 	
Kondisi Akhir (<i>Post-Conditions</i>)	Informasi detail tempat tinggal akan muncul. Jika tidak, keadaan sistem tidak berubah.

g. Skenario Use Case Kelola Komentar

Kebutuhan fungsional yang disediakan pada sistem adalah kebutuhan untuk melakukan interaksi melalui komentar yang terdapat pada detail informasi tempat tinggal yang disediakan sistem. Kebutuhan tersebut direpresentasikan oleh *use case* memberikan komentar, Tabel 4.10 merupakan skenario *use case* memberikan komentar.

Tabel 4.10 Skenario *Use Case* Memberikan Komentar

Nama	Memberikan Komentar
Kode SRS	SRS_004_02
Tujuan	Melihat komentar pada detail tempat tinggal yang dipilih dan dapat memberikan <i>feedback</i> melalui fasilitas komentar
Deskripsi (Brief Description)	<i>Use case</i> ini memungkinkan pengguna untuk melihat dan memberikan komentar pada detail tempat tinggal pada sistem. Aktor dapat melihat komentar apa saja yang terdapat pada <i>post</i> yang dipilih, dan aktor dapat memberikan komentar pada <i>post</i> tersebut.
Aktor	Pengguna
Kondisi Awal (Pre-Conditions)	Aktor harus membuka aplikasi terlebih dahulu dan memilih penawaran tempat tinggal melalui beberapa fitur pencarian sebelum <i>use case</i> dimulai.
Flow of Events	
Alur Utama (Basic Flow)	
<p><i>Use case</i> ini dimulai ketika aktor ingin melihat dan memberikan komentar pada detail tempat tinggal pada sistem.</p> <ol style="list-style-type: none"> 1. Aktor memilih penawaran tempat tinggal yang didapatkan setelah melakukan pencarian melalui fitur pencarian yang ada pada sistem. 2. Ketika aktor memilih tempat tinggal sistem akan menampilkan komentar yang ada pada <i>post</i> tempat tinggal yang dipilih. 3. Ketika aktor akan memberikan komentar, sistem telah menyediakan <i>form</i> komentar dengan data yaitu nama, email dan konten dari komentar. Pada aktor yang telah melakukan <i>login</i> nama dan email akan secara otomatis terisi dan ditambah dengan <i>profile picture</i> dari aktor yang bersangkutan. <i>Use case</i> berakhir setelah aktor keluar dari detail <i>post</i>. 	

Kondisi Akhir (<i>Post-Conditions</i>)	Informasi komentar pada detail tempat tinggal akan muncul. Jika tidak, keadaan sistem tidak berubah.
--	--

h. Skenario Use Case Lihat Deskripsi Aplikasi

Kebutuhan fungsional yang disediakan adalah informasi tentang sistem. Kebutuhan tersebut direpresentasikan oleh *use case* lihat deskripsi aplikasi, Tabel 4.11 merupakan skenario *use case* lihat deskripsi aplikasi.

Tabel 4.11 Skenario *Use Case* Lihat Deskripsi Aplikasi

Nama	Lihat Deskripsi Aplikasi
Kode SRS	SRS_005_01
Tujuan	Melihat deskripsi dan kegunaan dari sistem
Deskripsi (<i>Brief Description</i>)	<i>Use case</i> ini memungkinkan pengguna untuk melihat kegunaan dan fungsi dari sistem ini.
Aktor	Pengguna
Kondisi Awal (<i>Pre-Conditions</i>)	Aktor harus membuka aplikasi terlebih dahulu dan memilih fitur <i>about us</i> sebelum <i>use case</i> dimulai.
Flow of Events	
Alur Utama (<i>Basic Flow</i>)	
<i>Use case</i> ini dimulai ketika aktor ingin melihat informasi sistem.	
1. Aktor memilih menu lihat deskripsi dan kegunaan dari sistem.	
Kondisi Akhir (<i>Post-Conditions</i>)	Informasi profil sistem akan muncul. Jika tidak, keadaan sistem tidak berubah.

i. Skenario Use Case Kelola Permintaan Menginap

Kebutuhan fungsional yang disediakan adalah informasi tentang sistem. Kebutuhan tersebut direpresentasikan oleh *use case* kelola permintaan menginap, Tabel 4.12 merupakan skenario *use case* kelola permintaan menginap.

Tabel 4.12 Skenario *Use Case* Kelola Permintaan Menginap

Nama	Kelola Permintaan Menginap
Kode SRS	SRS_006_01
Tujuan	Mengelola permintaan menginap
Deskripsi (Brief Description)	<i>Use case</i> ini memungkinkan pengguna untuk melakukan permintaan menginap dan melakukan pengelolaan pada permintaan tersebut.
Aktor	Pengguna
Kondisi Awal (Pre-Conditions)	Aktor harus membuka aplikasi dan melakukan <i>login</i> terlebih dahulu sebelum <i>use case</i> dimulai.
Flow of Events	
Alur Utama (Basic Flow)	
<p>Use case ini dimulai ketika aktor ingin melakukan permintaan dan melakukan moderasi dari permintaan tersebut.</p> <ol style="list-style-type: none"> 1. Aktor melakukan permintaan menginap, atau melakukan moderasi pada permintaan menginap. 2. Ketika aktor memilih melakukan permintaan menginap maka <i>sub flow</i> “request tempat menginap” dijalankan. 3. Ketika aktor memilih melakukan moderasi permintaan tempat menginap maka <i>sub flow</i> “moderasi tempat menginap” dijalankan. 4. Ketika aktor selesai melihat daftar tempat tinggal, maka <i>use case</i> berakhir 	
Alur Bagian (Sub Flow)	
Request Tempat Menginap	<ol style="list-style-type: none"> 1. Sistem menampilkan informasi penawaran tempat tinggal sesuai penawaran tempat tinggal yang dipilih oleh aktor lalu aktor memilih pada <i>request</i> tempat menginap. 2. Sistem menyediakan fasilitas permintaan menginap dengan sistem meminta aktor memasukan tanggal sebagai parameter. 3. Setelah aktor selesai melakukan permintaan maka <i>use</i>

	<i>case</i> berakhir.
Moderasi Tempat Menginap	<ol style="list-style-type: none"> 1. Sistem menampilkan informasi pengguna siapa saja yang melakukan permintaan menginap. 2. Sistem menyediakan fasilitas moderasi yaitu melakukan <i>approve</i> atau <i>decline</i> pada permintaan menginap disertai dengan pesan singkat. 3. Setelah aktor selesai melakukan moderasi maka <i>use case</i> berakhir.
Alur Alternatif (<i>Alternative Flows</i>)	
Informasi Penawaran Tinggal Tersedia Data Tempat Tidak Tersedia	Jika pada <i>Basic Flow</i> sistem tidak dapat menemukan informasi permintaan menginap, maka sistem akan menampilkan pesan informasi tidak ditemukan penawaran tempat tinggal.
Kondisi Akhir (<i>Post-Conditions</i>)	Informasi profil sistem akan muncul. Jika tidak, keadaan sistem tidak berubah.

b. Skenario *Use Case Admin*

Berikut akan dijabarkan mengenai scenario *use case* dari pengguna berdasarkan *use case* admin:

a. Skenario *Use Case Kelola Penawaran Tempat Menginap*

Kebutuhan fungsional yang disediakan pada sistem adalah kebutuhan untuk mengelola penawaran tempat menginap pada sistem. Kebutuhan tersebut direpresentasikan oleh *use case* kelola penawaran tempat menginap admin, Tabel 4.13 merupakan skenario *use case* kelola penawaran tempat menginap.

Tabel 4.13 Skenario *Use Case* Kelola Penawaran Tempat Menginap

Nama	Kelola penawaran tempat menginap
Kode SRS	SRS_001_01
Tujuan	Mengelola penawaran tempat menginap
Deskripsi (Brief Description)	<i>Use case</i> ini memungkinkan admin untuk mengelola data penawaran tempat menginap pada sistem. Mengelola yang dimaksudkan pada <i>use case</i> adalah dapat melihat, mengubah dan melakukan hapus data tempat menginap dari sistem. Lihat dan mengubah diantaranya diantaranya judul, alamat, nomor telepon, harga, lokasi dan gambar yang berhubungan.
Aktor	Admin
Kondisi Awal (Pre-Conditions)	Aktor harus membuka dan melakukan login pada aplikasi terlebih dahulu sebelum <i>use case</i> dimulai. Aktor harus masuk pada menu <i>posts</i> untuk melakukan <i>write new</i> , <i>edit post</i> ataupun <i>delete</i> untuk melakukan perubahan.
Flow of Events	
Alur Utama (Basic Flow)	
<p><i>Use case</i> ini dimulai ketika aktor ingin menambah, mengubah, atau menghapus informasi dari penawaran tempat tinggal</p> <ol style="list-style-type: none"> 1. Aktor melakukan tambah penawaran tempat tinggal atau ubah penawaran tempat tinggal atau delete penawaran tempat tinggal. 2. Setelah aktor memilih pilihan informasi maka <i>sub flow</i> dijalankan. <ol style="list-style-type: none"> a. Jika aktor memilih “Tambah penawaran tempat tinggal” maka <i>sub flow</i> Tambah penawaran tempat tinggal dijalankan. b. Jika aktor memilih “Ubah penawaran tempat tinggal” maka <i>sub flow</i> Ubah penawaran tempat tinggal dijalankan. c. Jika aktor memilih “Hapus penawaran tempat tinggal” maka <i>sub flow</i> Hapus penawaran tempat tinggal dijalankan. 	

Alur Bagian (Sub Flow)	
Tambah Penawaran Tempat Tinggal	<ol style="list-style-type: none"> 1. Aktor memasukkan informasi data penawaran tempat tinggal. Informasi yang dimasukkan antara lain judul, harga, deskripsi, alamat, nomor telepon, lokasi tempat tinggal melalui peta ataupun lokasi pengguna, dan gambar pendukung. 2. Setelah aktor memberikan informasi yang diminta, sistem memberikan kode yang unik untuk data penawaran yang baru. Data ini akan di tambahkan ke dalam sistem. 3. Sistem menyediakan data penawaran tempat tinggal dengan kode yang baru.
Ubah Penawaran Tempat Tinggal	<ol style="list-style-type: none"> 1. Aktor untuk memilih penawaran tempat tinggal yang ada. 2. Sistem menampilkan detail data dan aktor harus menekan tombol <i>edit</i> untuk melakukan perubahan data. 3. Sistem menampilkan informasi penawaran dan aktor melakukan perubahan sesuai yang diinginkan. Informasi sesuai dengan yang ada pada <i>sub flow</i> tambah penawaran tempat tinggal. 4. Setelah aktor mengubah informasi yang diperlukan, sistem mengubah data penawaran tempat tinggal.
Hapus Penawaran Tempat Tinggal	<ol style="list-style-type: none"> 1. Aktor untuk memilih penawaran tempat tinggal yang ada. 2. Sistem menampilkan detail data dan aktor harus menekan tombol <i>delete</i> untuk melakukan hapus data. 3. Sistem menampilkan konfirmasi penghapusan penawaran tempat tinggal. Jika aktor mengkonfirmasi untuk melakukan hapus data, penawaran tempat tinggal akan hilang dari database.
Kondisi Akhir (Post-Conditions)	<p>Jika <i>use case</i> berhasil, informasi data penawaran tempat tinggal akan diperbarui dari sistem. Jika tidak, keadaan sistem tidak berubah.</p>

b. Skenario Use Case Kelola Akun Pengguna

Kebutuhan fungsional yang disediakan pada sistem adalah kebutuhan untuk mengelola akun pengguna pada sistem. Kebutuhan tersebut direpresentasikan oleh *use case* kelola akun pengguna, Tabel 4.14 merupakan skenario *use case* kelola penawaran tempat menginap.

Tabel 4.14 Skenario *Use Case* Kelola Akun Pengguna

Nama	Kelola akun pengguna
Kode SRS	SRS_002_01
Tujuan	Mengelola akun pengguna
Deskripsi (Brief Description)	<i>Use case</i> ini memungkinkan admin untuk mengelola data pengguna pada sistem. Mengelola yang dimaksudkan pada <i>use case</i> adalah dapat melihat, mengubah dan melakukan blokir akun pengguna. Lihat dan mengubah diantaranya judul, alamat, nomor telepon, harga, lokasi dan gambar yang berhubungan. Blokir pengguna akan menyebabkan pengguna tidak dapat menggunakan akunnnya lagi
Aktor	Admin
Kondisi Awal (Pre-Conditions)	Aktor harus membuka dan melakukan login pada aplikasi terlebih dahulu sebelum <i>use case</i> dimulai. Aktor harus masuk pada menu <i>users</i> untuk melakukan <i>add new</i> , <i>edit user</i> ataupun <i>delete</i> untuk mekakukan perubahan.
Flow of Events	
Alur Utama (Basic Flow)	
<p><i>Use case</i> ini dimulai ketika aktor ingin menambah, mengubah, atau memblokir akun pengguna</p> <ol style="list-style-type: none"> 1. Aktor melakukan perubahan, perubahan, atau memblokir akun pengguna. 2. Setelah aktor memilih pilihan informasi maka <i>sub flow</i> dijalankan. 	



- a. Jika aktor memilih “Tambah akun pengguna” maka *sub flow* Tambah akun pengguna dijalankan.
- b. Jika aktor memilih “Ubah akun pengguna” maka *sub flow* Ubah akun pengguna dijalankan.
- c. Jika aktor memilih “Blokir akun” maka *sub flow* Blokir akun dijalankan.

Alur Bagian (Sub Flow)

Tambah Akun	<ol style="list-style-type: none"> 1. Aktor memasukkan informasi data pengguna. Informasi yang dimasukkan antara lain <i>avatar</i>, nama, email, <i>username</i>, <i>password</i>, bio, jenis kelamin, status dan level pengguna. 2. Setelah aktor memberikan informasi yang diminta, sistem memberikan kode yang unik untuk data akun yang baru. Data ini akan di tambahkan ke dalam sistem. 3. Sistem menyediakan akun pengguna dengan kode yang baru.
Ubah Akun	<ol style="list-style-type: none"> 1. Aktor untuk memilih akun pengguna yang ada. 2. Sistem menampilkan detail data dan aktor harus menekan tombol <i>edit</i> untuk melakukan perubahan data. 3. Sistem menampilkan informasi akun pengguna dan aktor melakukan perubahan sesuai yang diinginkan. Informasi sesuai dengan yang ada pada <i>sub flow</i> tambah akun pengguna. 4. Setelah aktor mengubah informasi yang diperlukan, sistem mengubah data akun pengguna.
Blokir Akun	<ol style="list-style-type: none"> 1. Aktor untuk memilih akun pengguna yang ada. 2. Sistem menampilkan detail data dan aktor harus menekan tombol <i>delete</i> untuk melakukan blokir akun pengguna. 3. Sistem menampilkan konfirmasi pemblokiran akun. Jika aktor mengkonfirmasi untuk melakukan hapus data, maka status akun akan menjadi tidak aktif pada database.
Kondisi Akhir (Post-Conditions)	Informasi data akun pengguna akan diperbarui dari sistem. Jika tidak, keadaan sistem tidak berubah.

c. Skenario Use Case Kelola Komentar

Kebutuhan fungsional yang disediakan pada sistem adalah kebutuhan untuk mengelola komentar pada sistem. Kebutuhan tersebut direpresentasikan oleh *use case* kelola komentar, Tabel 4.15 merupakan skenario *use case* kelola komentar.

Tabel 4.15 Skenario *Use Case* Kelola Komentar

Nama	Kelola komentar
Kode SRS	SRS_003_01
Tujuan	Mengelola komentar
Deskripsi (Brief Description)	<i>Use case</i> ini memungkinkan admin untuk mengelola komentar pada sistem. Mengelola yang dimaksudkan pada <i>use case</i> adalah dapat melihat komentar pada sistem melakukan <i>delete</i> pada komentar yang tidak sesuai.
Aktor	Admin
Kondisi Awal (Pre-Conditions)	Aktor harus membuka dan melakukan login pada aplikasi terlebih dahulu sebelum <i>use case</i> dimulai. Aktor harus masuk pada menu <i>comments</i> untuk melakukan lihat komentar dan <i>delete</i> komentar.
Flow of Events	
Alur Utama (Basic Flow)	
<p><i>Use case</i> ini dimulai ketika menghapus komentar pada sistem.</p> <ol style="list-style-type: none"> 1. Aktor melihat atau menghapus komentar. 2. Jika aktor memilih data komentar, detail akan langsung terlihat pada halaman tersebut. 3. Jika aktor memilih “hapus data” maka akan muncul konfirmasi penghapusan komentar. Jika aktor memberikan konformasi maka data akan dihapus pada sistem. 	
Kondisi Akhir (Post-Conditions)	Informasi data komentar akan diperbarui dari sistem. Jika tidak, keadaan sistem tidak berubah.



4.1.2.4 Analisis Kebutuhan Non-Fungsional

Analisis kebutuhan non fungsional adalah analisis untuk mengetahui spesifikasi yang dibutuhkan oleh sistem agar dapat berjalan secara sempurna. Kebutuhan non fungsional tersebut adalah *Usability* dan *Compability*.

Tabel 4.16 Spesifikasi kebutuhan non-fungsional

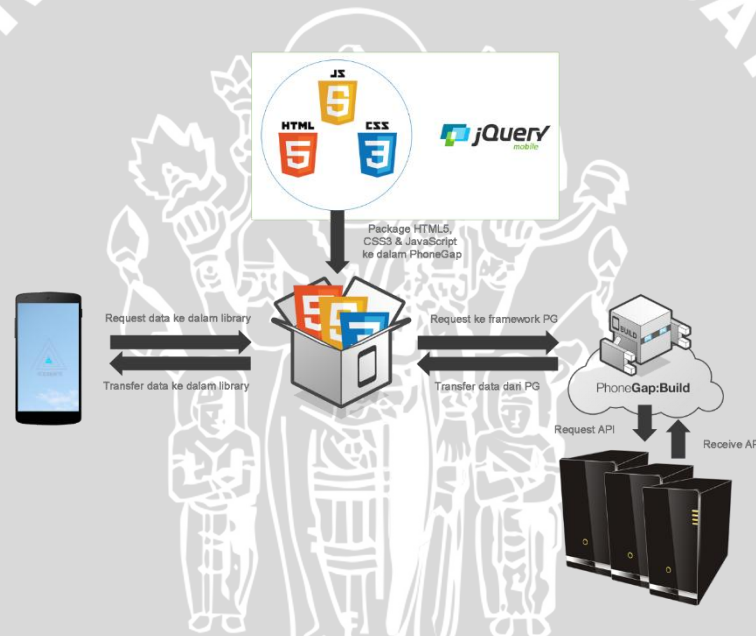
Parameter	Deskripsi Kebutuhan
<i>Compatibility</i>	Aplikasi harus dapat dijalankan di berbagai sistem operasi seperti Android dan iOS.
<i>Usability</i>	Tampilan antar muka pada aplikasi dirancang sedemikian rupa hingga memudahkan penggunaan karena antarmuka sistem dirancang dalam bentuk mobile. Bahasa yang digunakan pada sistem adalah bahasa inggris yang <i>familiar</i> oleh pengguna dalam negeri ataupun luar negeri, sehingga pengguna dapat dengan mudah membaca berbagai informasi yang ada pada sistem. Fitur – fitur pada aplikasi dibuat ringkas mungkin sehingga memudahkan penggunaan sistem.

4.1.3 Perancangan perangkat Lunak

Perancangan aplikasi perangkat lunak dilakukan dalam lima tahap yaitu, perancangan arsitektural, perancangan basis data, pemodelan *class diagram* untuk menggambarkan perancangan struktur *class – class* yang menyusun aplikasi, perancangan *sequence diagram* untuk menggambarkan interaksi antar objek, dan perancangan antar muka aplikasi. Perancangan aplikasi pada skripsi ini menggunakan pendekatan desain berorientasi objek yang direpresentasikan dengan menggunakan UML (*Unified Modelling Language*).

4.1.3.1 Perancangan Arsitektur Sistem

Perancangan perangkat lunak aplikasi *mobile roamte* dibangun dengan menggunakan konsep *hybrid mobile application* dengan pada perangkat Android. Konsep *hybrid* adalah pembuatan aplikasi menggunakan HTML5, CSS3 dan javascript dimana ketiganya saling melengkapi dan secara maksimal bekerja dalam JQuery Mobile. Penulis menggunakan *framework* phonegap yang berguna untuk mengkonversikan aplikasi yang dibuat melalui JQuery mobile sehingga dapat berjalan secara *native*. Teknik penyimpanan dilakukan menggunakan media penyimpanan secara terpusat yaitu server, dimana database berjalan dengan MySQL. Berikut rancangan arsitektur hubungan sistem akan ditunjukkan dalam Gambar 4.5.



Gambar 4.5 Perancangan Arsitektur Sistem Aplikasi

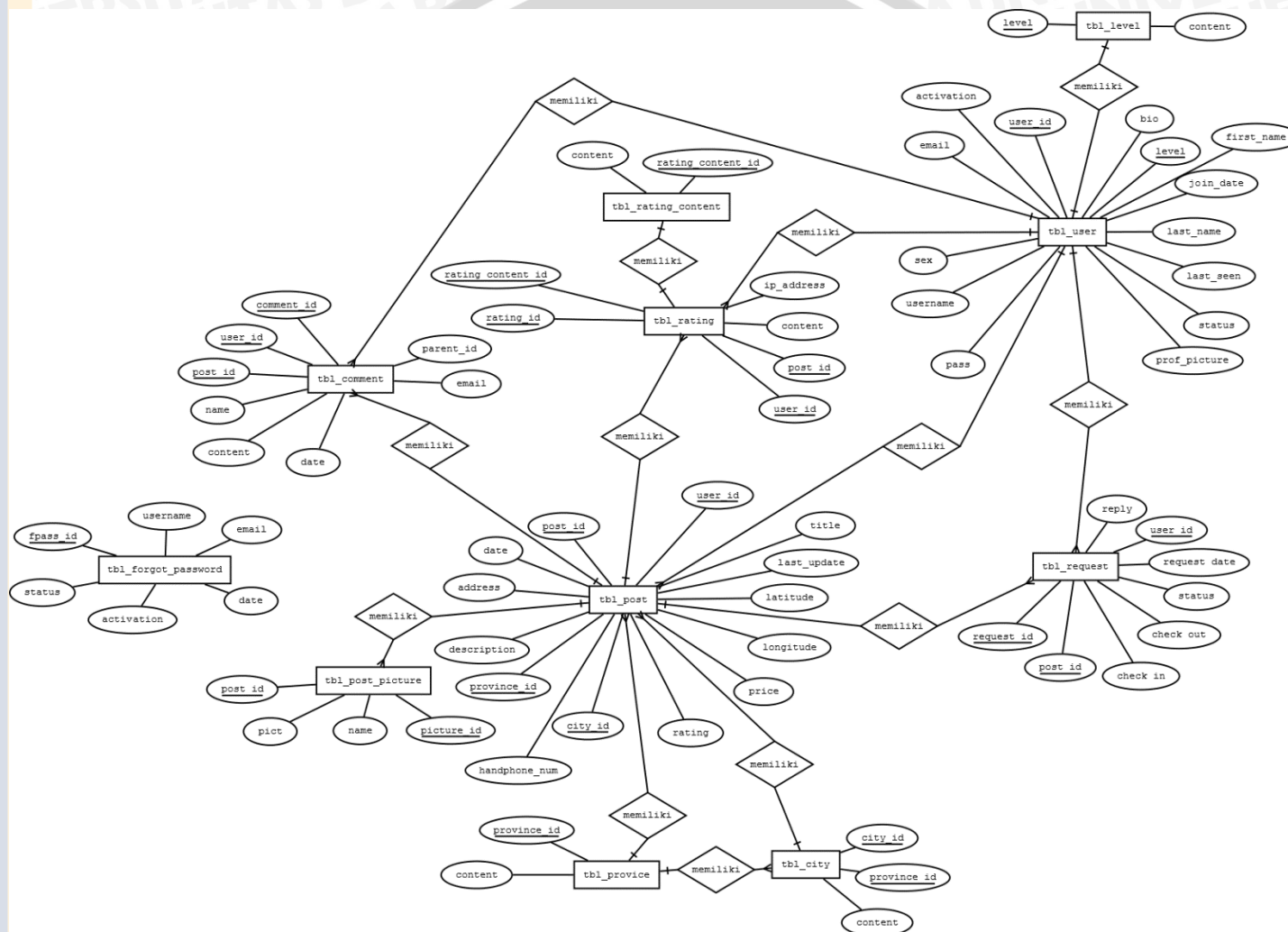
4.1.3.2 Perancangan Basis Data

Basis data berfungsi sebagai media penyimpanan data, pada penelitian ini penyimpanan data dilakukan secara terpusat pada server. Perancangan basis data direpresentasikan dalam bentuk *Entity Relationship Diagram* (ERD). ERD berisi komponen-komponen himpunan entitas dan himpunan relasi yang masing-masing dilengkapi dengan beberapa atribut yang mempresentasikan seluruh fakta yang ditinjau dari keadaan yang nyata dan aplikasi-aplikasi pencari penginapan yang

sudah ada. Aplikasi yang dimaksud adalah “*Hostelworld*” yang dapat di unduh melalui *Google Play Store*.

Terdapat 10 buah tabel yaitu tabel provinsi , tabel kota, tabel *user*, tabel *post*, tabel *post picture*, tabel *rating content*, tabel *rating*, tabel komentar, tabel level dan tabel *forgot password*. Berdasarkan aplikasi acuan tabel *post* dibutuhkan beberapa atribut diantaranya *title* atau judul, *address* atau alamat, *description* atau keterangan, provinsi, kota, dan posisi dari tempat menginap yang berupa *latitude* dan *longitude*. Untuk tabel *rating* dibutuhkan tambahan atribut karena perbedaan sistem *rating*, pada penelitian ini dibutuhkan atribut *user id* karena pengguna yang melakukan proses *rating* berdasarkan parameter pada tabel *rating content*. Tabel komentar memiliki penyesuaian atribut sesuai fitur *reviews* pada aplikasi acuan karena secara umum memiliki fungsi yang sama, yaitu *content* atau isi, *user id*, *date* atau tanggal, dan *post id*.

Dalam Gambar 4.6 terdapat relasi yang menghubungkan antar tabel. Tabel *tbl_user* dengan 5 tabel (*tbl_level*, *tbl_post*, *tbl_request*, *tbl_rating*, dan *tbl_comment*) mempunyai hubungan dimana satu user dapat memiliki banyak data dalam kelima tabel tersebut, sehingga relasinya adalah *tbl_user* (one): *tbl_level*, *tbl_post*, *tbl_request*, *tbl_rating*, dan *tbl_comment* (many). Tabel *tbl_post* dan *tbl_post_picture* mempunyai hubungan dimana satu post dapat memiliki banyak gambar sesuai atribut pada tabel *tbl_post_picture*, sehingga relasinya adalah *tbl_post* (one): *tbl_post_picture* (many). Tabel *tbl_post* dan *tbl_request* mempunyai hubungan dimana satu post dapat memiliki banyak permintaan untuk menginap di penawaran tempat tinggal yang dipilih, sehingga relasinya adalah *tbl_post* (one): *tbl_request* (many). Terdapat *tbl_user* dan *tbl_level* yang memiliki hubungan dimana satu pengguna memiliki satu level, sehingga relasinya adalah *tbl_user* (one): *tbl_level* (one). Terdapat tabel *tbl_forgot_password* yang tidak memiliki relasi karena berfungsi sebagai tabel log, yaitu tabel yang mencatat permintaan lupa password berdasarkan email yang dimasukkan, dan jika tidak terdapat konfirmasi setelah 1 hari maka permintaan tersebut sudah tidak dapat digunakan atau *expired*. ERD dari penelitian ini dapat dilihat dalam Gambar 4.6 dibawah ini.



Gambar 4.6 Entity Relational Diagram

Berikut struktur tabel dan keterangan dari masing-masing field yang ada pada database.

Entitas *tbl_province* mewakili tabel *tbl_province* yang berisi data-data provinsi dari seluruh Indonesia. Tabel *tbl_province* berisi *province id* dan *content*. Struktur tabel *province* ditunjukkan pada Tabel 4.17.

Tabel 4.17 Struktur Tabel *tbl_province*

No.	Nama Field	Type Data	Keterangan
1	Province_id (pk)	Integer	Kode dari provinsi (1,2,3,...)
2	Content	Varchar	Nama provinsi (Jawa Timur, Jawa Barat,...)

Entitas *tbl_city* mewakili tabel *tbl_city* yang berisi data-data kota dari seluruh Indonesia. Tabel *tbl_city* berisi *city id*, *province id* dan *content*. Struktur tabel *tbl_city* ditunjukkan pada Tabel 4.18.

Tabel 4.18 Struktur Tabel *tbl_city*

No.	Nama Field	Type Data	Keterangan
1	City_id (pk)	Integer	Kode dari kota (1,2,3,...)
2	Province_id (fk)	Integer	Kode dari provinsi (1,2,3,...)
3	Content	Varchar	Nama kota (Malang, Jakarta,...)

Entitas *tbl_user* mewakili tabel *tbl_user* yang berisi data-data *user*. Tabel *tbl_user* berisi *user id*, *first name*, *last name*, *email*, *username*, *password*, *profile picture*, *sex*, *join date*, *bio* dan *last seen*. Struktur tabel *tbl_user* ditunjukkan pada Tabel 4.19.

Tabel 4.19 Struktur Tabel *tbl_user*

No.	Nama Field	Type Data	Keterangan
1	User_id (pk)	Integer	Kode dari user (1,2,3,...)
2	First_name	Varchar	Nama pertama dari user (afif)
3	Last_name	Varchar	Nama terakhir dari user (fahmi)
4	Email	Varchar	Email user (af.brilian@gmail.com)
5	Username	Varchar	Username (admin)

6	Pass	Varchar	Password (admin)
7	Prof_picture	Varchar	Lokasi gambar user (aset/pict/1.png)
8	Sex	Varchar	Jenis kelamin user (L/P)
9	Join_date	Date	Tanggal, bulan dan tahun bergabung user (2014 – 03 – 10)
10	Bio	Text	Bio dari user
11	Last_seen	Datetime	Waktu terakhir user menggunakan aplikasi (03:17:47)
12	activation	Varchar	Kode aktivasi akun
13	status	Enum	Status pengguna aktif atau tidak (0,1)
14	Level (fk)	Int	Level / hak akses pengguna (admin)

Entitas *tbl_rating_content* mewakili tabel *tbl_rating_content* yang berisi pertanyaan-pertanyaan yang digunakan sebagai parameter untuk melakukan rating. Tabel *tbl_rating_content* berisi *rating_content id* dan *content*. Struktur tabel *tbl_rating_content* ditunjukkan pada Tabel 4.20.

Tabel 4.20 Struktur Tabel *tbl_rating_content*

No.	Nama Field	Type Data	Keterangan
1	Rating_content_id (pk)	Integer	Kode dari rating (1,2,3,...)
2	Content	Varchar	Isi Rating (apakah bagus)

Entitas *tbl_post* mewakili tabel *tbl_post* yang berisi data-data dari penawaran tempat tinggal. Tabel *tbl_post* berisi *post id*, *province id*, *city id*, *user id*, *title*, *price*, *date*, *address*, *handphone number*, *description*, *rating*, *longitude*, *langitude* dan *last update*. Struktur tabel *tbl_post* ditunjukkan pada Tabel 4.21.

Tabel 4.21 Struktur Tabel *tbl_post*

No.	Nama Field	Type Data	Keterangan
1	Post_id (pk)	Integer	Kode dari kota (1,2,3,...)
2	City_id (fk)	Integer	Kode dari kota (1,2,3,...)
3	Province_id (fk)	Integer	Kode dari provinsi (1,2,3,...)
4	User_id (fk)	Integer	Kode dari user (1,2,3,...)
5	Title	Varchar	Judul dari <i>post</i> (kamar kosong)

6	Price	Varchar	Harga dari tempat tinggal (100000)
7	Date	Datetime	Tanggal <i>post</i> di unggah (2014-1-1)
8	Address	Varchar	Alamat dari tempat tinggal (malang)
9	Handphone_num	Varchar	Nomor telepon (0341418876)
10	Desc	Longtext	Deskripsi dari tempat tinggal
11	Rating	Double	Rating tempat tinggal (1,2,3,...)
12	Longitude	Varchar	Longitude dari tempat tinggal (-8.298360)
13	Latitude	Varchar	Latitude dari tempat tinggal (111.738604)
14	Last_update	Datetime	<i>Update</i> terakhir pada <i>post</i> (2014-03-23 03:17:47)

Entitas *tbl_post_picture* mewakili tabel *tbl_post_picture* yang berisi gambar-gambar pendukung dari penawaran tempat tinggal yang ada pada tabel *tbl_post*. Tabel *tbl_post_picture* berisi *picture id*, nama, dan *pict*. Struktur tabel *tbl_post_picture* ditunjukkan pada Tabel 4.22.

Tabel 4.22 Struktur Tabel *tbl_post_picture*

No.	Nama Field	Type Data	Keterangan
1	Picture_id (pk)	Integer	Kode dari gambar (1,2,3,...)
2	Post_id (fk)	Integer	Kode dari post (1,2,3,...)
3	Name	Varchar	Nama gambar (gambar)
4	Pict	Varchar	Lokasi disimpannya gambar (aset/pict/1.png)

Entitas *tbl_comment* mewakili tabel *tbl_comment* yang berisi komentar-komentar dari penawaran tempat tinggal yang ada pada tabel *tbl_post*. Tabel *comment* berisi *comment id*, nama, *content*, *date*, *parent id* dan *email*. Struktur tabel *tbl_comment* ditunjukkan pada Tabel 4.23.

Tabel 4.23 Struktur Tabel *tbl_comment*

No.	Nama Field	Type Data	Keterangan
1	comment_id (pk)	Integer	Kode dari komentar (1,2,3,...)
2	User_id (fk)	Varchar	Kode dari <i>user</i> (1,2,3,...)

3	Post_id (fk)	Varchar	Kode dari <i>post</i> (1,2,3,...)
4	Name	Varchar	Nama (brilian)
5	Content	Varchar	Isi komentar (post bagus)
6	Date	Datetime	Tanggal komentar (2014-03-23 03:17:47)
7	Parent_id	Varchar	Jika komentar adalah balasan dari komentar yang sudah ada (1,2,3,...)
8	Email	Varchar	Email pengirim (reyourday@gmail.com)

Entitas *tbl_rating* mewakili tabel *tbl_rating* yang berisi kumpulan rating dari penawaran tempat tinggal yang ada pada tabel *tbl_post*. Tabel *tbl_rating* berisi *rating id*, *content* dan alamat ip. Struktur tabel *tbl_rating* ditunjukkan pada Tabel 4.24.

Tabel 4.24 Struktur Tabel *tbl_rating*

No.	Nama Field	Type Data	Keterangan
1	Rating_id (pk)	Integer	Kode dari rating (1,2,3,...)
2	User_id (fk)	Integer	Kode dari <i>user</i> (1,2,3,...)
3	Post_id (fk)	Integer	Kode dari <i>post</i> (1,2,3,...)
4	Rating_content_id	Integer	Kode dari tabel <i>rating content</i> (1,2,3,...)
5	Content	Integer	Nilai rating (1,2,3,...)
6	IP Address	Varchar	IP pengguna yang melakukan rating (172.21.0.00)

Entitas *tbl_level* mewakili tabel *tbl_level* yang berisi level atau hak akses dari pengguna aplikasi. Tabel *tbl_level* berisi level dan *content*. Struktur tabel *tbl_level* ditunjukkan pada Tabel 4.25.

Tabel 4.25 Struktur Tabel *tbl_level*

No.	Nama Field	Type Data	Keterangan
1	level (pk)	Integer	Kode dari level (1,2,3,...)
2	Content	Varchar	Keterangan level (admin, user)

Entitas `tbl_forgot_password` mewakili tabel `tbl_forgot_password` yang berisi data dari pengguna yang tidak dapat mengingat passwordnya. Tabel `tbl_forgot_password` berisi `fpass_id`, `username`, `email`, `activation`, `status`, `done`. Struktur tabel `tbl_forgot_password` ditunjukkan pada Tabel 4.26.

Tabel 4.26 Struktur Tabel `tbl_forgot_password`

No.	Nama Field	Type Data	Keterangan
1	<code>fpass_id</code> (pk)	Integer	Kode tabel forgot password (1,2,3,...)
2	Username	Integer	Username pengguna (Segesta)
3	Email	Integer	Email pengguna (reyourday@gmail.com)
4	Activation	Integer	Kode aktivasi (124fvev5)
5	Status	Integer	Status aktivasi (0,1)
6	Done	Datetime	Tanggal <i>expired</i> dari permintaan (2014-03-23 03:17:47)

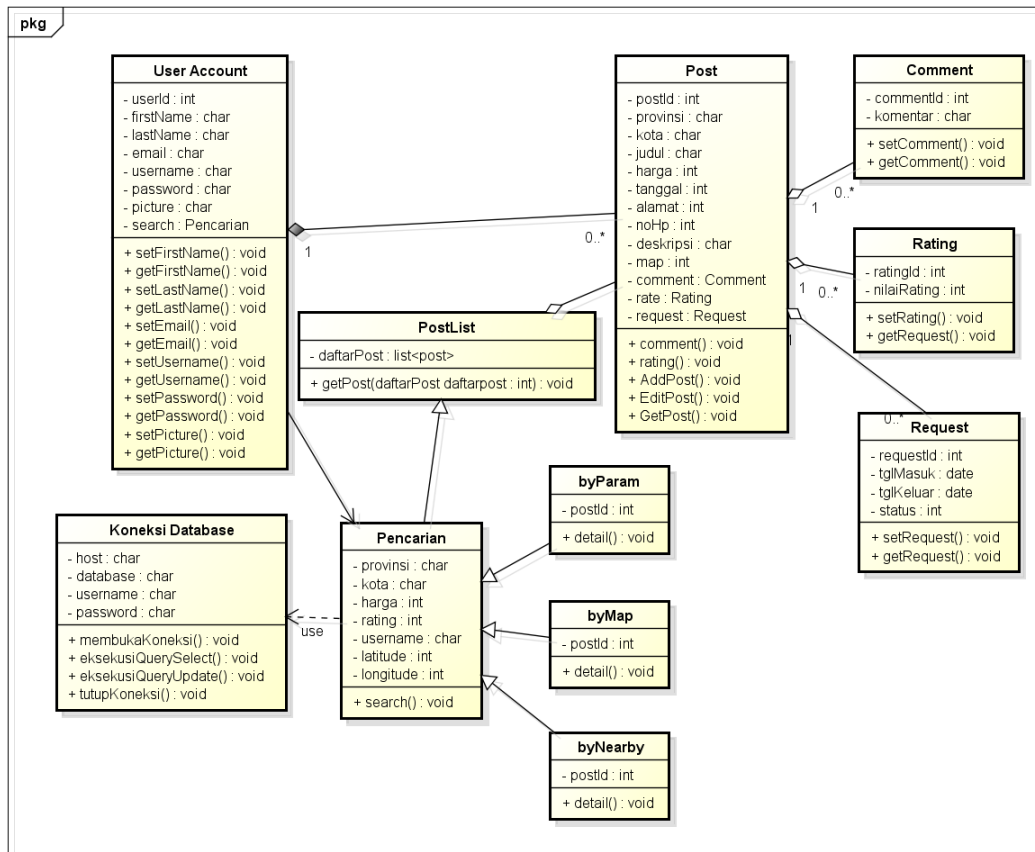
Entitas `tbl_request` mewakili tabel `tbl_request` yang berisi data dari pengguna yang melakukan permintaan menginap. Tabel `tbl_request` berisi `requests_id`, `post_id`, `check in`, `check out`, `status`, `request date`, `reply`, `user_id`. Struktur tabel `tbl_request` ditunjukkan pada Tabel 4.27.

Tabel 4.27 Struktur Tabel `tbl_request`

No.	Nama Field	Type Data	Keterangan
1	<code>request_id</code> (pk)	Integer	Kode tabel <i>request</i> (1,2,3,...)
2	<code>post_id</code> (fk)	Integer	Kode post (1,2,3,...)
3	<code>check_in</code>	Date	Tanggal permintaan masuk (2014-01-01)
4	<code>check_out</code>	Date	Tanggal permintaan keluar (2014-01-01)
5	Status	Enum	Status permintaan (0,1)
6	<code>request_date</code>	Date	Tanggal <i>request</i> dari permintaan (2014-01-01)
7	<code>reply</code>	Varchar	Jawaban dari pengguna (ya, tidak)
8	<code>user_id</code> (fk)	Integer	User_id pengguna (1,2,3,...)

4.1.3.3 Perancangan Class Diagram

Class diagram adalah kumpulan objek-objek yang mempunyai struktur umum, behavior umum, dan relasi yang umum. Class diagram memberi gambaran (diagram statis) tentang sistem/perangkat lunak dan relas-relasi yang ada didalamnya.



Gambar 4.7 Class Diagram Aplikasi

Dalam Gambar 4.7 ditunjukkan object, method, dan relasi antar class diagram. Class diagram tersebut menunjukkan pengguna dapat memiliki banyak penawaran tempat menginap dengan relasi komposisi dimana penawaran tempat tinggal tidak bisa berdiri sendiri tanpa pengguna. Pengguna dapat melakukan pencarian yang ditunjukkan dengan relasi asosiasi dimana pencarian dapat dilakukan melalui parameter, peta dan lokasi terdekat dengan pengguna. Penawaran tempat tinggal memiliki komentar, rating dan request dengan relasi



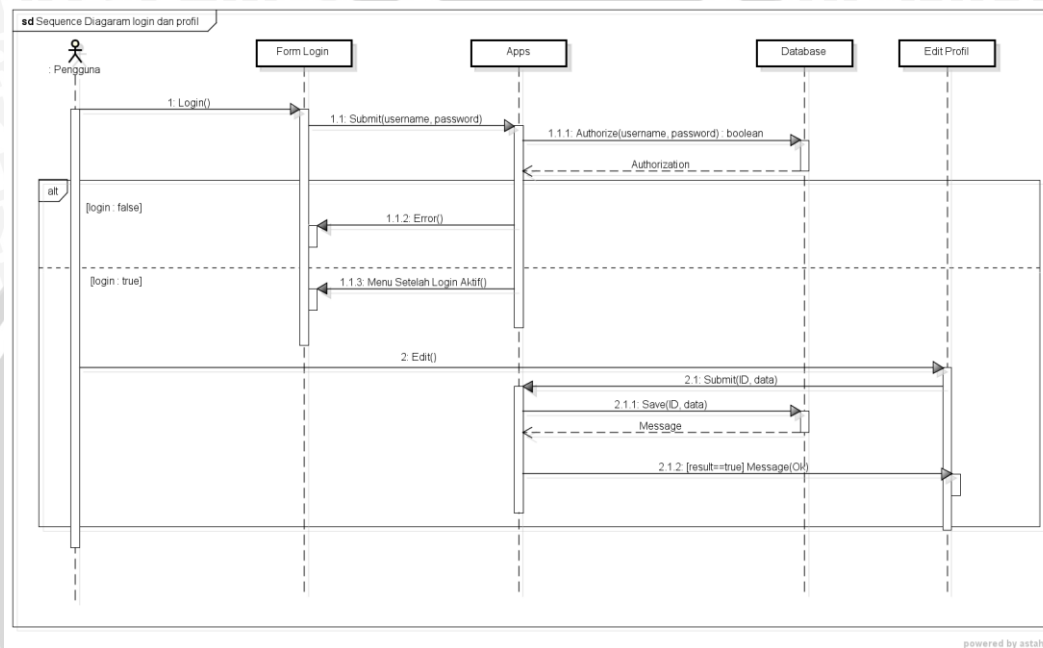
agregasi dimana penawaran tempat tinggal dapat berdiri sendiri tanpa komentar, rating dan request.



4.1.3.4 Perancangan *Sequence Diagram*

Sequence diagram adalah suatu diagram yang menggambarkan interaksi antar obyek dan mengindikasikan komunikasi diantara obyek-obyek tersebut, dimana interaksi tersebut didasarkan pada diagram *use case*.

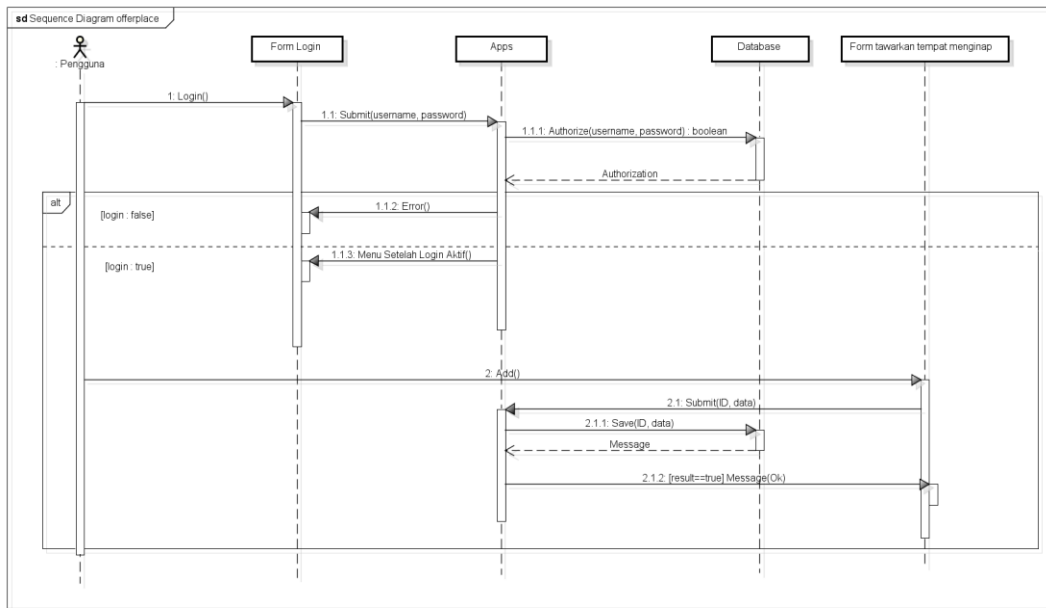
1. *Sequence Diagram* Login dan Kelola Profil Pribadi



Gambar 4.8 *Sequence Diagram* Login dan Kelola Profil

Dalam Gambar 4.8 ditunjukkan aktifitas yang dilakukan oleh pengguna, antarmuka pengguna dan sistem basis data. *Sequence diagram* dalam Gambar menunjukkan urutan langkah pengguna dalam melakukan kelola profil pribadi. Pengguna melakukan login terlebih dahulu dengan mengisi form login dan menekan tombol login, lalu sistem akan memproses pada database dan sistem akan menjalankan aktifitas selanjutnya sesuai kembalian dari database. Pengguna melakukan edit profil dan menekan tombol save sehingga data akan diperbarui pada database.

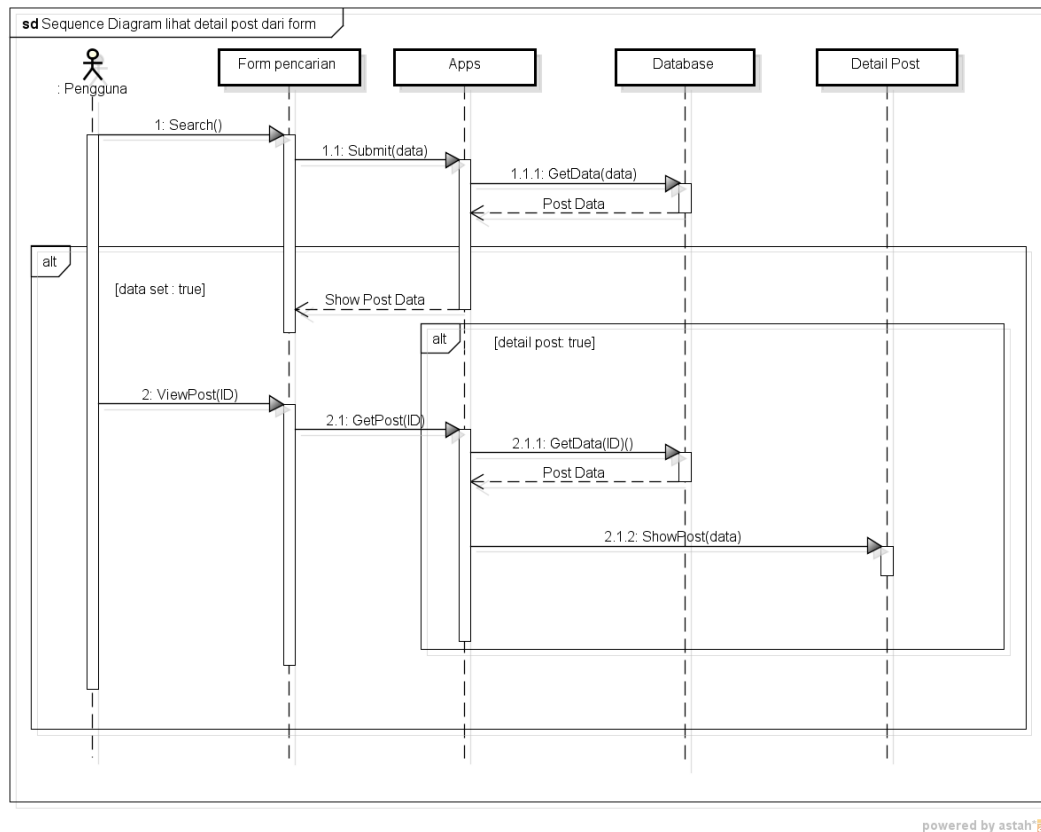
2. Sequence Diagram Kelola Penawaran Tempat Tinggal



Gambar 4.9 Sequence Diagram Kelola Penawaran Tempat Tinggal

Dalam Gambar 4.9 ditunjukkan aktifitas yang dilakukan oleh pengguna, antarmuka pengguna dan sistem basis data. *Sequence diagram* dalam Gambar menunjukkan urutan langkah pengguna dalam melakukan kelola penawaran tempat tinggal. Pengguna melakukan login terlebih dahulu dengan mengisi form login dan menekan tombol login, lalu sistem akan memproses pada database dan sistem akan menjalankan aktifitas selanjutnya sesuai kembalian dari database. Pengguna dapat menambahkan penawaran tempat tinggal dengan mengisi form penawaran tempat tinggal dan setelah itu melakukan penyimpanan ke dalam database dengan menekan tombol save.

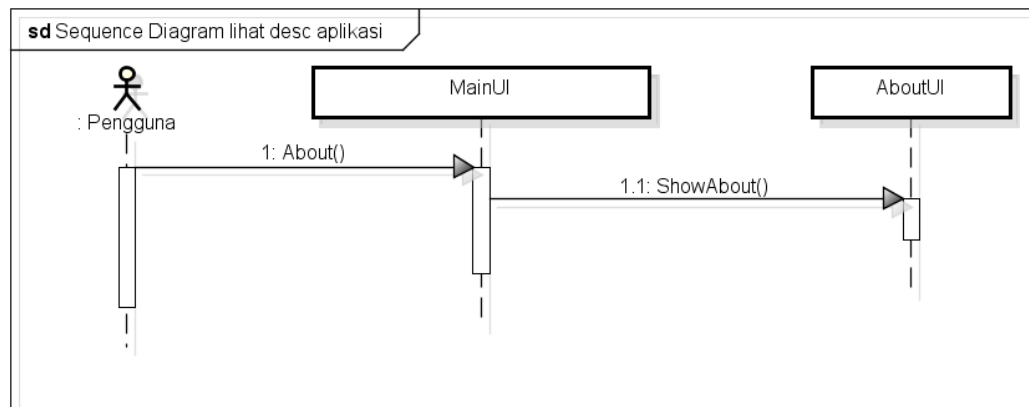
3. *Sequence Diagram* Lihat Tempat Menginap Melalui Pencarian Berdasarkan Parameter



Gambar 4.10 *Sequence Diagram* Lihat Tempat Menginap Melalui Pencarian Berdasarkan Parameter

Dalam Gambar 4.10 ditunjukkan aktifitas yang dilakukan oleh pengguna, antarmuka pengguna dan sistem basis data. *Sequence diagram* dalam Gambar menunjukkan urutan langkah pengguna dalam melakukan proses pencarian tempat tinggal berdasarkan parameter. Pengguna melakukan pencarian melalui *form* pencarian lalu menekan tombol proses. Setelah itu database akan mengembalikan data – data pencarian sesuai kriteria pengguna. Langkah selanjutnya pengguna memilih *post* dan sistem akan mengambil data sesuai pilihan pengguna dan ditampilkan pada *detail post*.

4. **Sequence Diagram Lihat Deskripsi Aplikasi**

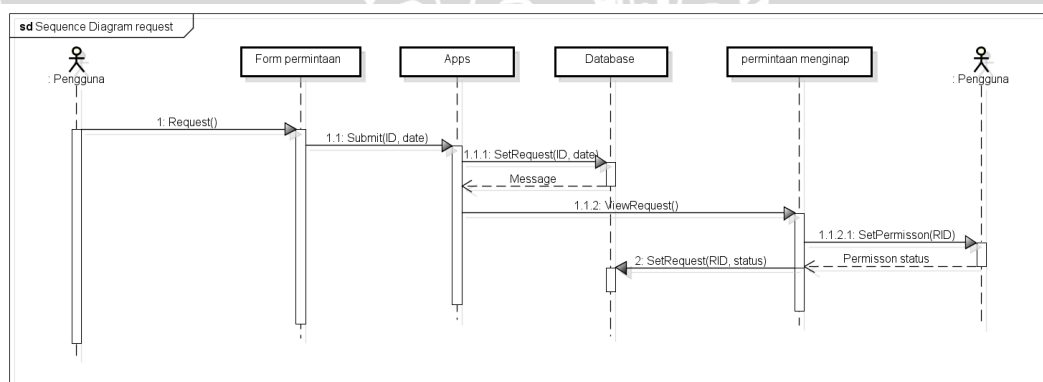


powered by astah

Gambar 4.11 *Sequence Diagram* Lihat Deskripsi Aplikasi

Dalam Gambar 4.11 ditunjukkan aktifitas yang dilakukan oleh pengguna dan antarmuka. *Sequence diagram* dalam Gambar menunjukkan urutan langkah pengguna dalam melakukan proses melihat deskripsi aplikasi. Pengguna menekan tombol lihat deskripsi aplikasi lalu sistem akan menampilkan informasi deskripsi aplikasi melalui antarmuka.

5. **Sequence Diagram Kelola Permintaan Menginap**



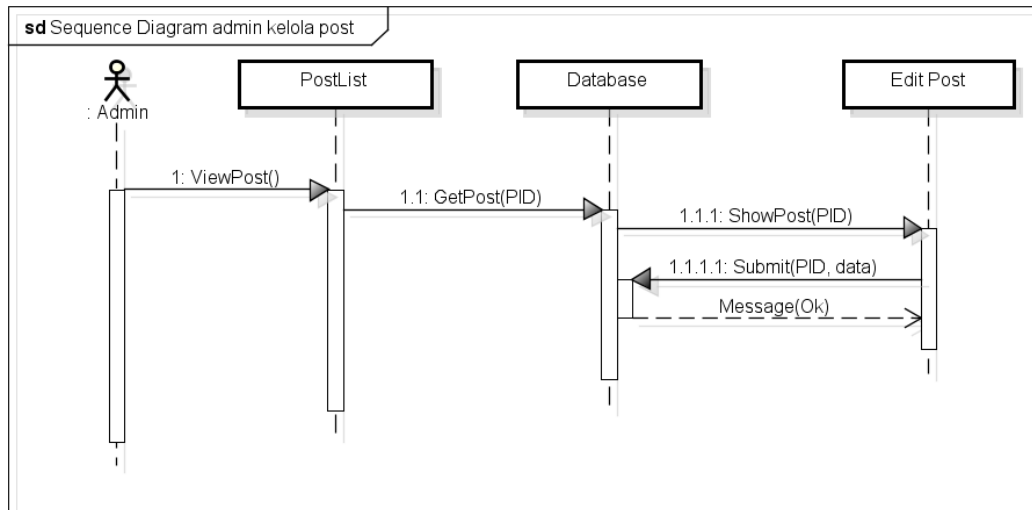
powered by astah

Gambar 4.12 *Sequence Diagram* Kelola Permintaan Menginap

Dalam Gambar 4.12 ditunjukkan aktifitas yang dilakukan oleh pengguna yang melakukan permintaan dan pengguna yang menyediakan tempat menginap pada antarmuka. *Sequence diagram* dalam Gambar menunjukkan urutan langkah pengguna pertama dalam melakukan proses permintaan data pada sistem. Pengguna melakukan permintaan menginap lalu pengguna yang menawarkan

tempat menginap akan melakukan proses *approve* atau *decline* dari permintaan tersebut.

6. **Sequence Diagram Kelola Data pada Admin**



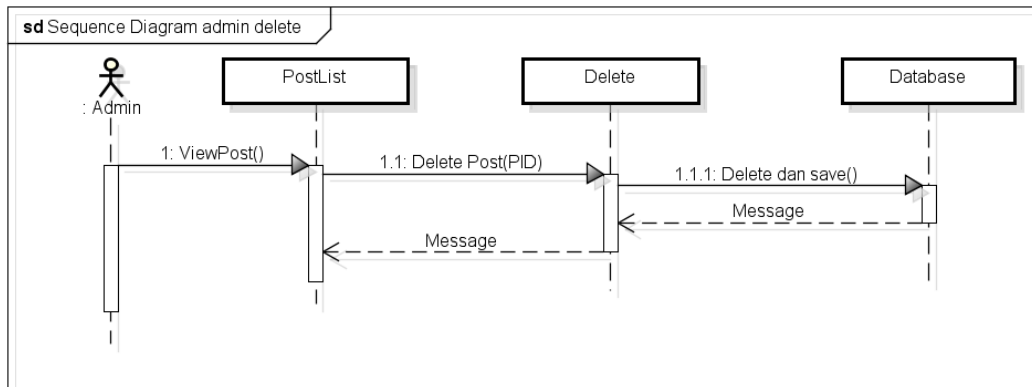
powered by astah

Gambar 4.13 *Sequence Diagram* Kelola Data pada Admin

Dalam Gambar 4.13 ditunjukkan aktifitas yang dilakukan oleh admin dan antarmuka. *Sequence diagram* dalam Gambar menunjukkan urutan langkah admin dalam melakukan proses pengelolaan data pada sistem. *Sequence diagram* diatas mewakili pengelolaan data penawaran tempat tinggal dan pengelolaan akun pengguna. Admin melakukan pengelolaan dengan menampilkan data terlebih dahulu, lalu admin dapat melakukan perubahan melalui form edit dan langkah selanjutnya adalah melakukan penyimpanan kedalam database.



7. **Sequence Diagram Hapus Data pada Admin**



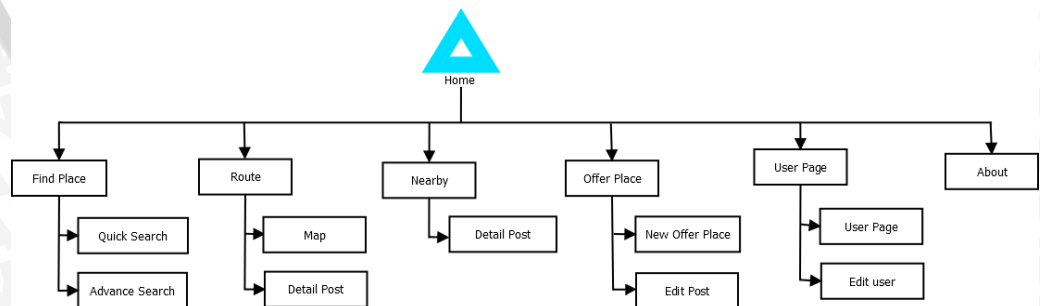
powered by astah

Gambar 4.14 Sequence Diagram Hapus Data pada Admin

Dalam Gambar 4.14 ditunjukkan aktifitas yang dilakukan oleh admin dan antarmuka. *Sequence diagram* dalam Gambar menunjukkan urutan langkah admin dalam melakukan proses penghapusan data pada sistem. Sequence diagram diatas mewakili pengelolaan data penawaran tempat tinggal, pengelolaan akun pengguna dan pengelolaan komentar. Admin melakukan pengelolaan dengan menampilkan data terlebih dahulu, lalu admin dapat melakukan perubahan dengan menekan tombol delete lalu data pada database akan terhapus.

4.1.3.5 **Perancangan Antar Muka**

Pada bagian ini akan dijelaskan mengenai tampilan antar muka aplikasi *rooamate*. Berikut adalah Struktur pohon menu dari aplikasi *rooamate*.

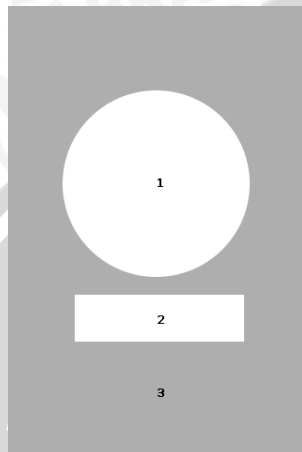


Gambar 4.15 Struktur Pohon Menu Aplikasi



a. Halaman *Splash Screen*

Halaman *splash screen* merupakan tampilan awal ketika pengguna membuka aplikasi. *Splash screen* memiliki durasi selama 3 detik yang kemudian akan langsung masuk ke halaman home. *Splash screen* berisi logo aplikasi dan nama aplikasi.



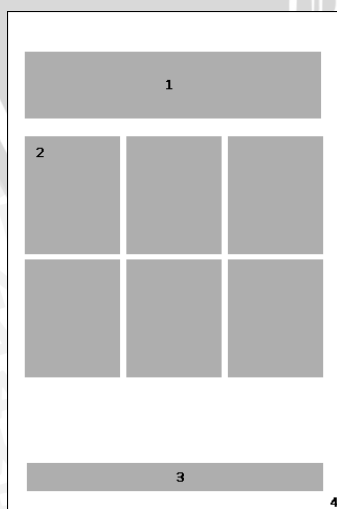
Keterangan :

1. Logo aplikasi
2. Nama aplikasi
3. Background

Gambar 4.16 Antar Muka Halaman *Splash Screen*

b. Halaman Menu Utama (*Home*)

Halaman *home* merupakan tampilan awal setelah antar muka *splash screen* selesai berjalan. Halaman home terdiri dari menu-menu utama yang disediakan oleh aplikasi. Gambar 4.17 adalah gambar rancangan antarmuka halaman *home*.



Keterangan :

1. Logo aplikasi
2. Menu utama aplikasi
3. link facebook, twitter, g+
4. background aplikasi

Gambar 4.17 Antar Muka Halaman Menu Utama (*Home*)

c. Halaman Form

Halaman form akan ditampilkan saat pengguna menjalankan aksi pencarian berdasarkan parameter, login, kelola profil pribadi dan kelola penawaran tempat tinggal. Gambar 4.18 adalah gambar rancangan antarmuka halaman form.

The wireframe shows a form layout with the following elements:

- 1**: Header area at the top.
- 2**: A form area containing two rows of labels and text boxes:

Label 1	Text 1
Label 2	Text 2
- 3**: A dark grey rectangular button.
- 4**: A list of data items, each with a right-pointing arrow:

Search...
Data 1
Data 2
Data 3
etc..

Keterangan :

1. Header aplikasi
2. Form
3. Tombol submit
4. List data

Gambar 4.18 Antar Muka Halaman Form

d. Halaman Pencarian Berdasarkan Peta

Halaman pencarian berdasarkan peta akan ditampilkan saat pengguna menjalankan aksi pencarian berdasarkan peta. Gambar 4.19 adalah gambar rancangan antarmuka halaman Pencarian berdasarkan Peta.

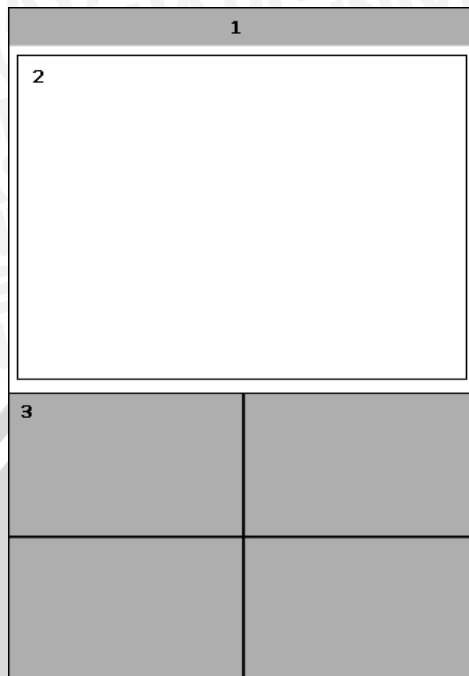
The wireframe shows a map search interface with the following elements:

- 1**: Header area at the top.
- 2**: A large map area.
- 3**: A downward-pointing triangle icon, likely representing the user's location.
- 4**: A diamond-shaped icon, likely representing a search location or marker.

Keterangan :

1. Header aplikasi
2. Map
3. lokasi pengguna
4. lokasi tempat tinggal

Gambar 4.19 Antar Muka Halaman Pencarian berdasarkan Peta



3. Footer pada detail

e. Halaman Detail Penawaran

Tempat Menginap

Halaman detail penawaran tempat menginap akan ditampilkan saat pengguna memilih *post* setelah melakukan pencarian. Gambar 4.20 adalah gambar rancangan antarmuka halaman detail penawaran tempat menginap. **Keterangan :**

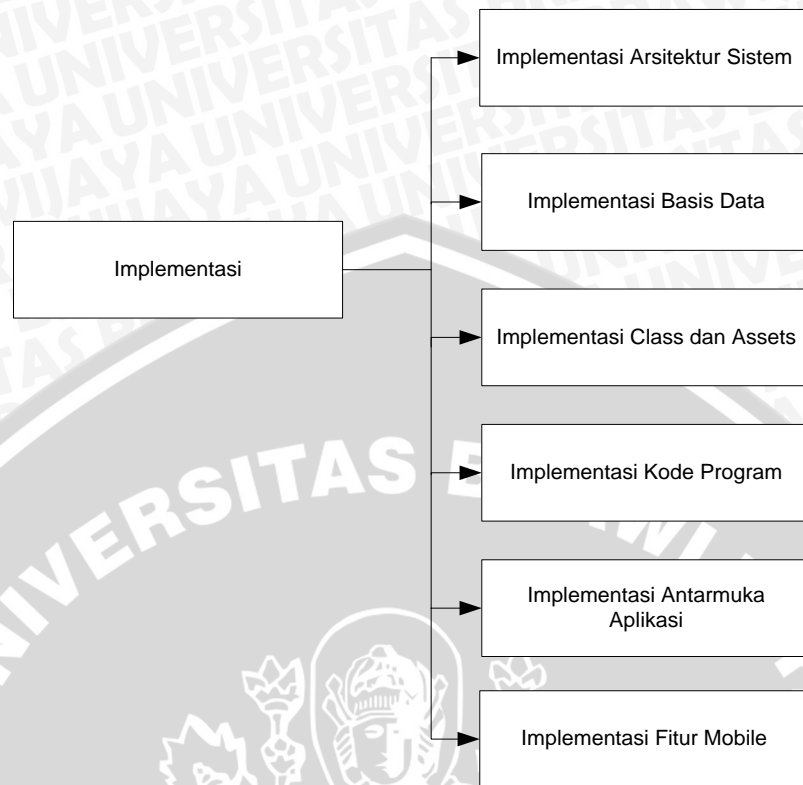
1. Header aplikasi

2. Isi detail

Gambar 4.20 Halaman Detail Penawaran Tempat Menginap

4.2 Implementasi

Pada bab ini dibahas mengenai implementasi perangkat lunak berdasarkan hasil yang telah didapatkan dari analisis kebutuhan dan perancangan perangkat lunak. Pembahasan bab implementasi terdiri dari spesifikasi sistem, batasan-batasan dalam implementasi, implementasi *class* dan *assets*, implementasi algoritma dan kode program dan implementasi antarmuka aplikasi.



Gambar 4.21 Diagram Pohon Implementasi

4.2.1 Spesifikasi Sistem

Hasil dari analisis kebutuhan dan perancangan lunak yang telah dijelaskan pada tahap analisis kebutuhan dan tahap perancangan sistem menjadi dasar untuk dilakukan implementasi menjadi sebuah aplikasi Android yang dapat berfungsi sesuai dengan kebutuhan. Spesifikasi sistem diimplementasikan pada spesifikasi perangkat keras dan perangkat lunak.

4.1.2.1 Spesifikasi Perangkat Keras

Dalam pengembangan aplikasi *rooamate* ini menggunakan sebuah komputer dengan spesifikasi *processor*, *memory*, dan *display* yang akan dijelaskan pada Tabel 4.28 berikut.

Tabel 4.28 Spesifikasi Perangkat Keras Komputer

Nama Komponen	Spesifikasi
<i>System Model</i>	Acer Aspire 4741G
<i>Processor</i>	Intel (R) Core (TM) i5-450M CPU @ 2.40GHz (4 CPUs), ~2.4GHz
<i>Memory</i>	6144 MB RAM
<i>Display</i>	NVIDIA GeForce 310M

Dalam melakukan proses instalasi dan pengujian, perangkat yang digunakan adalah perangkat bergerak smartphone Android dengan spesifikasi perangkat keras yang ditunjukkan pada Tabel 4.29 berikut.

Tabel 4.29 Spesifikasi Perangkat Keras *Smartphone*

Nama Komponen	Spesifikasi
<i>System Model</i>	GALAXY Y GT-S5360
<i>Processor</i>	830 MHz ARMv6
<i>Memory</i>	microSD, up to 32 GB 180 MB, 290 MB RAM
<i>Display</i>	240 x 320 pixels, 3.0 inches (~133 ppi pixel density)
Spesifikasi Fisik	97.5 g (3.42 oz) weight and 104 x 58 x 11.5 mm dimension

4.1.2.2 Spesifikasi Perangkat Lunak

Dalam proses pengembangan aplikasi *mobile rooamate* ini menggunakan perangkat lunak dengan spesifikasi yang akan dijelaskan pada Tabel 4.30 berikut.

Tabel 4.30 Spesifikasi Perangkat Lunak Komputer

Nama Komponen	Spesifikasi
<i>Operating System</i>	Windows 8 Pro with Media Center 64-bit (6.2, Build 9200)
<i>Bahasa Pemrograman</i>	Java, HTML5, CSS dan Javascript
<i>IDE (Integrated Development Environmet)</i>	Eclipse Juno dengan ADT (Android Development Tools) plug in

Dalam proses implementasi dan pengujian dari pengembangan aplikasi *mobile roamate* menggunakan perangkat lunak pada smartphone dengan spesifikasi seperti pada Tabel 4.31 berikut

Tabel 4.31 Spesifikasi Perangkat Lunak *Smartphone*

Nama Komponen	Spesifikasi
<i>Platform</i>	Android 4.1.1 (Jelly Bean) Android Browser HSDPA 14.4/ HSPA 5.76

4.2.2 Batasan – Batasan Implementasi

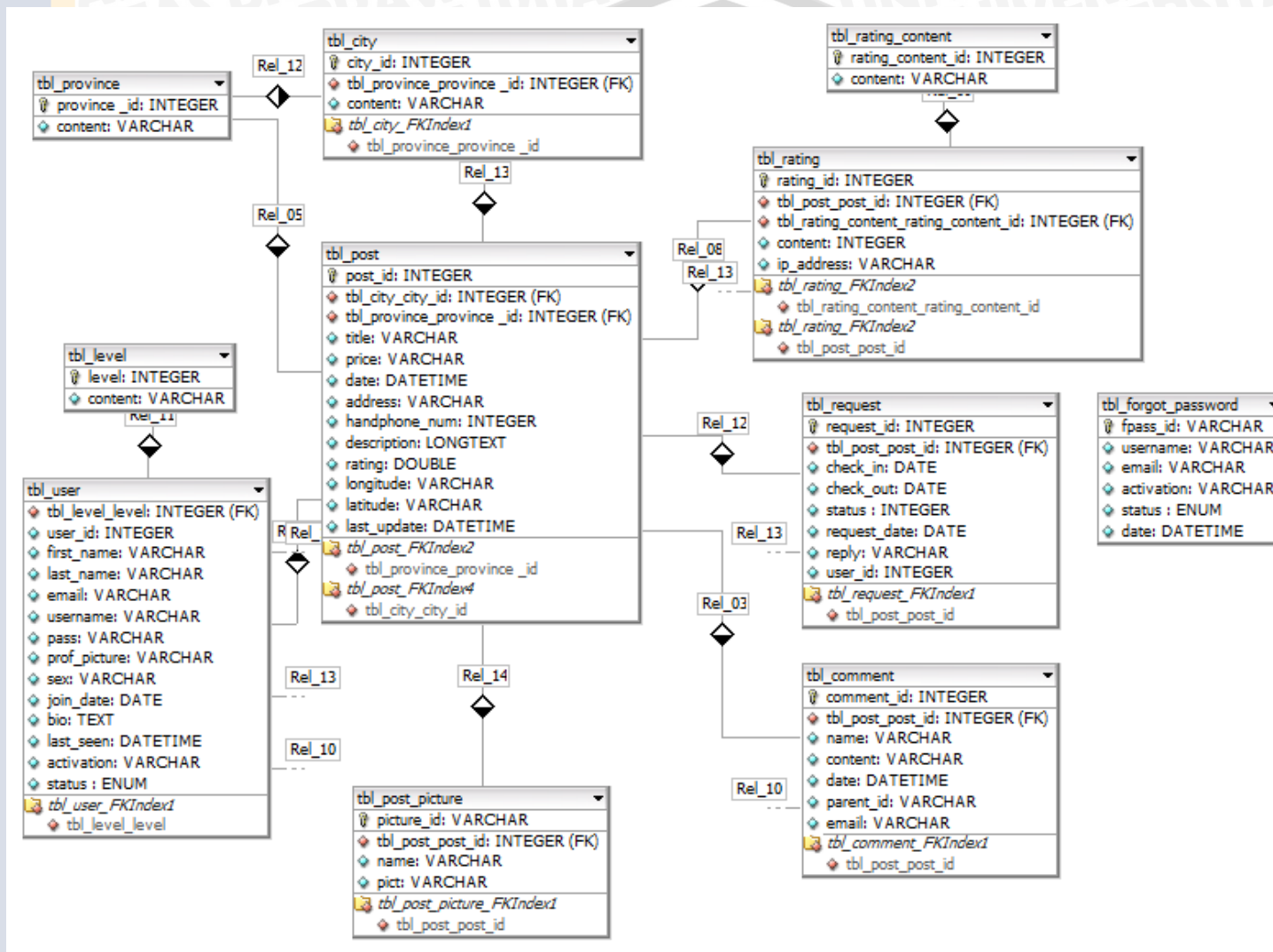
Pada implementasi perangkat lunak aplikasi *mobile roamate* terdapat batasan-batasan dalam proses yaitu sebagai berikut :

1. Aplikasi *mobile roamate* dirancang untuk dijalankan pada smartphone Android menggunakan konsep hybrid.
2. Aplikasi harus berjalan dengan terkoneksi internet sebagai media pertukaran data karena aplikasi menggunakan konsep *online* dengan media penyimpanan terpusat yaitu server.
3. Aplikasi menggunakan framework PhoneGap dalam pembuatan aplikasi mobile hybrid

4. Komunikasi data antara aplikasi dengan *online* storage diimplementasikan menggunakan pertukaran data dengan bahasa pemrograman javascript.
5. Penyimpanan data pada media penyimpanan terpusat yaitu server menggunakan MySQL yang dihubungkan oleh *framework* PhoneGap.
6. Pembuatan layout pada user interface aplikasi menggunakan bahasa pemrograman HTML, CSS, dan Javascript.
7. Pembuatan modul administrator menggunakan *framework* code igniter (CI) dengan tambahan bootstrap sebagai layout CSS.

4.2.3 Implementasi Basis Data

Implementasi pada penyimpanan data dilakukan pada media penyimpanan terpusat yaitu pada server. Hasil implementasi penyimpanan data ini berupa database dengan format extension file SQL. Hasil implementasi database MySQL pada database ini dimodelkan dalam diagram konseptual entity relationship. Pada konseptual entity relationship terdapat hubungan relasi antar tabel. Dalam Gambar 4.21 akan menggambarkan diagram konseptual entity relationship dari aplikasi *mobile roamate*. Implementasi query akan ditunjukkan dalam Gambar 4.22, 4.23, 4.24, dan 4.25.



Gambar 4.22 Diagram Konseptual Entity Relationship

```
1 CREATE TABLE tbl_post (  
2     post_id INTEGER UNSIGNED NOT NULL AUTO_INCREMENT,  
3     city_id INTEGER UNSIGNED NOT NULL,  
4     province_id INTEGER UNSIGNED NOT NULL,  
5     title VARCHAR NULL,  
6     price VARCHAR NULL,  
7     date DATETIME NULL,  
8     address VARCHAR NULL,  
9     handphone_num INTEGER UNSIGNED NULL,  
10    description LONGTEXT NULL,  
11    rating DOUBLE NULL,  
12    longitude VARCHAR NULL,  
13    latitude VARCHAR NULL,  
14    last_update DATETIME NULL,  
15    PRIMARY KEY(post_id),  
16    FOREIGN KEY ("province_id") REFERENCES "tbl_province"  
17    ("province_id"),  
18    FOREIGN KEY ("city_id") REFERENCES "tbl_city" ("city_id")  
19 );
```

Gambar 4.23 Implementasi Query *Create* tabel *tbl_post*

```
1 CREATE TABLE tbl_request (  
2     request_id INTEGER UNSIGNED NOT NULL AUTO_INCREMENT,  
3     post_id INTEGER UNSIGNED NOT NULL,  
4     check_in DATE NULL,  
5     check_out DATE NULL,  
6     status INTEGER UNSIGNED NULL,  
7     request_date DATE NULL,  
8     reply VARCHAR NULL,  
9     user_id INTEGER UNSIGNED NULL,  
10    PRIMARY KEY(request_id),  
11    FOREIGN KEY ("post_id") REFERENCES "tbl_post" ("post_id")  
12 );
```

Gambar 4.24 Implementasi Query *Create* tabel *tbl_request*

```
1 CREATE TABLE tbl_user (  
2     level INTEGER UNSIGNED NOT NULL,  
3     user_id INTEGER UNSIGNED NOT NULL AUTO_INCREMENT,  
4     first_name VARCHAR NULL,  
5     last_name VARCHAR NULL,  
6     email VARCHAR NULL,  
7     username VARCHAR NULL,  
8     pass VARCHAR NULL,  
9     prof_picture VARCHAR NULL,  
10    sex VARCHAR NULL,  
11    join_date DATE NULL,  
12    bio TEXT NULL,  
13    last_seen DATETIME NULL,  
14    activation VARCHAR NULL,  
15    status ENUM NULL,  
16    FOREIGN KEY ("level") REFERENCES "tbl_level" ("level")  
17 );
```

Gambar 4.25 Implementasi Query *Create* tabel `tbl_user`

```
1 CREATE TABLE tbl_rating (  
2     rating_id INTEGER UNSIGNED NOT NULL AUTO_INCREMENT,  
3     post_id INTEGER UNSIGNED NOT NULL,  
4     rating_content_id INTEGER UNSIGNED NOT NULL,  
5     content INTEGER UNSIGNED NULL,  
6     ip_address VARCHAR NULL,  
7     PRIMARY KEY(rating_id),  
8     FOREIGN KEY ("rating_content_id") REFERENCES  
9     "tbl_rating_content" ("rating_content_id"),  
10    FOREIGN KEY ("post_id") REFERENCES "tbl_post" ("post_id")  
11 );
```

Gambar 4.26 Implementasi Query *Create* tabel `tbl_rating`

4.2.4 Implementasi Class dan Assets Pada File Program

Class dan assets direalisasikan kedalam bentuk file program, diantaranya adalah implementasi class default dengan menggunakan format java (.java) dan implementasi assets dengan menggunakan format html (.html) serta javascript (.js). Hasil implementasi akan ditunjukkan pada Tabel 4.32, 4.33 dan 4.34 berikut.

Tabel 4.32 Implementasi class pada kode program *.java

Folder	Package	Nama	Nama File Program
src	Com.roamate	Main Activity	MainActivity.java

Tabel 4.33 Implementasi assets pada kode program *.html

Folder	Nama	Nama File program
assets/www	Index	index.html
assets/www	Main	main.html
assets/www	Findplace	findplace.html
assets/www	Advance search	advancesearch.html
assets/www	Route	route.html
assets/www	Nearby	nearby.html
assets/www	Offer place	offerplace.html
assets/www	Login	login.html
assets/www	Signup	signup.html
assets/www	User	user.html
assets/www	Edit password	editpassword.html
assets/www	Detail	detail.html
assets/www	About	about.html

Tabel 4.34 Implementasi assets pada kode program * .js

Folder	Nama	Nama File program
assets/www/js	Pencarian melalui parameter	app.js app-css.js
assets/www/js	Pencarian melalui peta	app.js app-css.js map.js jquery.ui.map.js jquery.ui.map.extensions.js jquery.ui.map.service.js
assets/www/js	Pencarian melalui lokasi terdekat	app.js app-css.js map.js jquery.ui.map.js jquery.ui.map.extensions.js jquery.ui.map.service.js
assets/www/js	Kelola penawaran tempat tinggal	app.js app-css.js map.js jquery.ui.map.js jquery.ui.map.extensions.js jquery.ui.map.service.js
assets/www/js	Kelola profil pribadi	app.js app-css.js
assets/www/js	Detail post	app.js app-css.js map.js jquery.ui.map.js jquery.ui.map.extensions.js jquery.ui.map.service.js
assets/www/js	About	app-css.js

4.2.5 Implementasi Kode Program

Aplikasi *mobile* ini mempunyai beberapa proses (*function*) utama yang terdapat pada satu file javascript. Pada penulisan skripsi ini akan dicantumkan beberapa algoritma utama pada javascript. Algoritma proses yang dicantumkan antara lain adalah lihat, ubah, dan hapus data.

4.2.5.1 Algoritma Proses Lihat Data

Algoritma proses lihat data dijalankan saat pengguna memilih menu pencarian. Sistem aplikasi akan memanggil proses ke dalam sistem basis data dan menampilkannya ke dalam antarmuka pengguna. Algoritma ini akan digunakan berulang pada beberapa fitur aplikasi yang berelasi dengan basis data. Algoritma ini disimpan dalam file javascript (*.js).

Implementasi algoritma proses lihat data akan ditunjukkan pada kode program dalam Gambar 4.27 berikut.

```
1 function load_data(i, province_id_, city_id_, pricefrom_,
2 priceto_, user_, counter, userid, rating_){
3 if(pricefrom_=='')pricefrom_=0;
4 $.ajax({
5 url: urlRequest+'search_room.php',
6 type: 'POST',
7 async: 'true',
8 dataType: 'json',
9 beforeSend: function() { $('#cust-spin').html('<center><div
10 class="custom-spinner"></div></center>'); }, //Show spinner
11 complete: function() { $('#cust-spin').empty(); },
12 //Hide spinner
13 cache: false,
14
15 data: {
16     province_id : province_id_,
17     city_id      : city_id_,
18     pricefrom   : pricefrom_,
19     priceto     : priceto_,
20     user        : user_,
21     rating      : rating_,
```

```
22     counter      : counter,
23     userid       : userid
24   },
25   success: function (result) {
26     var cek = JSON.stringify(result);
27     if(cek=="No rows found"){
28       $('#span-room-list').empty();
29       $('#span-room-list').html('<p align="center">No
30 room available</p>');
31       $('#end-data-msg').html('<a href="#" class="ui-btn
32 ui-btn-custom"><bodytext><p class="pbutton">NO ROOM
33 AVAILABLE</p></bodytext></a>');
34       if(i=='remove')$('#room-list' + ' li').remove();
35     }
36     else{
37       var output = '';
38       $.each(result, function(i, f){
39         output += '<li data-icon="false">';
40         output += '<a class="a-li-cust" href="detail.html"
41 onclick="get_id(\''+ f.post_id +'\', \''+ f.latitude +'\',
42 \''+ f.longitude +'\', \''+ f.user_id +'\', \''\')" data-
43 ajax="false" rel="external" >';
44         output += ' <h2>' +f.title+ '</h2><p>';
46         if(f.price == 0){ output += 'Free'; }
47         else{ output += 'Rp.'+f.price; }
48         output += '<span style="float: right; position:
49 relative;font-size: 11px;"><b>by</b> '+f.username+',
50 &nbsp;'+(f.date).substr(0,10)+'<span></p>';
51         output += '<span id="'+f.post_id+'-notif"
52 class="new_com_notif label-danger"></span>';
53         output += '</a></li>';
54       });
55       if(i=='remove')$('#room-list' + ' li').remove();
56       $('#room-list').append(output);
57       $('#room-list').listview('refresh');
58       com_notif();//-- CALL IF THERE IS NEW COMMENT ---
59     }
```

```

60     }
61     }); //-----END AJAX-----
62 } //-----END START LOAD DATA POST FUNCTION-----
63 -----

```

Gambar 4.27 Implementasi Algoritma Proses Lihat Data

Gambar di atas menunjukkan implementasi dari algoritma proses untuk melihat atau menampilkan data pada pengguna. Pada baris ke 4 fungsi ajax akan mengirimkan data data dengan berbagai parameter seperti pada baris 16 – 25. Data – data tersebut akan dikirim menuju server dan ajax akan menerima kembalian data dalam bentuk JSON. Selanjutnya JSON akan diolah seperti pada baris 26 – 65 untuk ditampilkan pada pengguna.

4.2.5.2 Algoritma Proses Ubah data

Algoritma proses ubah data dijalankan saat pengguna memilih menu *edit pada* penawaran tempat tinggal dan profil. Sistem aplikasi akan memanggil proses ke dalam sistem basis data dan menampilkannya ke dalam antarmuka pengguna. Algoritma ini disimpan dalam file javascript (*.js).

Implementasi algoritma proses ubah data akan ditunjukkan pada kode program dalam Gambar 4.28 berikut.

```

1  function detail_info(post_id){
2  $.ajax({
3      url: urlRequest+'search_room.php',
4      type: 'POST',
5      async: 'true',
6      dataType: 'json',
7      beforeSend: function() { $('#.cust-
8  spin').html('<center><div class="custom-
9  spinner"></div></center>'); }, //Show spinner
10     complete: function() { $('#.cust-spin').empty(); },
11     //Hide spinner
12     cache: false,
13     data: {
14         post_id : post_id,
15         param : sessionStorage.param

```



```
16     },
17     success: function (result) {
18         var cek = JSON.stringify(result);
19         if(cek=="No rows found"){
20             $('#end-data-msg').html('<a href="#" class="ui-btn
21 ui-btn-custom"><bodytext><p class="pbutton">NO ROOM
22 AVAILABLE</p></bodytext></a>');
23             // alert("gagal");
24         }
25     } else{
26         var output = '';
27         if(sessionStorage.param=='edit'){
28             var delay=100;//1 seconds
29             $.each(result, function(i, f){
30                 $.each(f.post, function(i, f){
31                     $('#hidId').val(f.hid_id);
32                     $('#title').val(f.title);
33                     $('#price').val(f.price);
34                     $('#desc').val(f.description);
35                     $('#address').val(f.address);
36                     $('#phone').val(f.handphone_num);
37                     $('#lat').val(f.latitude);
38                     $('#long').val(f.longitude);
39                     setTimeout(function(){
40                         $('#province').val(f.province_id);
41                         $('#province option[value="' + f.province_id +
42 "']").prop('selected', true);
43                         $('#province').selectmenu('refresh');
44                     },delay);
45                     setTimeout(function(){
46                         $('#city').val(f.city_id);
47                         $('#city option[value="' + f.city_id +
48 "']").prop('selected', true);
49                         $('#city').selectmenu('refresh');
50                     },delay);
51                 });
52             $.each(f.pict, function(i, f){
53                 output += "<li id="+f.picture_id+"><a class='a-li-
```

```
54 cust' onclick='deletepict("+f.picture_id+")' ><img
55 width='80px' height='80px'
56 src='"+urlRequest+f.pict+"/><p>"+f.picture_id"</p></a></
57 li>";
58     $('#uploadedfilelist').html(output);
59     $('#uploadedfilelist').listview('refresh');
60     });
61     });
62 //-----HIDE UPLOAD FORM-----
63     var uploadedfile =
64     $("#uploadedfilelist").children().length;
65     if(uploadedfile==4||uploadedfile>4){$('#form-
66 upload').hide();}
67 //-----HIDE UPLOAD FORM-----
68     }else{
69     $.each(result, function(i, f){
70     $('#.title').html((f.title).toUpperCase());
71     if(f.price=='0'){$('#.price').html("Free");}
72     else {$('#.price').html(f.price);}
73     $('#.address').html(f.address);
74     $('#.hp').html(f.handphone_num);
75     $('#.desc').html(f.description);
76     $('#.post-by').html('<b>by</b> <a href="user.html"
77 onclick="get_view_user_id(\''+ f.user_id +'\')" data-
78 ajax="false" rel="external">'+f.username+'</a>,
79 &nbsp;'+(f.date).substr(0,10));
80     var lastupdate = (f.last_update).substr(0,10);
81     if(lastupdate!='0000-00-00'){$('#.last-
82 update').html('Last update : '+lastupdate);}
83     });
84     $('#end-data-msg').empty();
85     }
86     }
87 }
88 });//-----END AJAX-----
89 };
90
```

Gambar 4.28 Implementasi Algoritma Proses Ubah data

Gambar di atas menunjukkan implementasi dari algoritma proses untuk merubah data. Pada baris ke 2 fungsi ajax akan mengirimkan data data dengan berbagai parameter seperti pada baris 13 – 16. Data – data tersebut akan dikirim menuju server dan ajax akan menerima kembalian data dalam bentuk JSON. Selanjutnya JSON akan diolah seperti pada baris 29 – 87 untuk ditampilkan pada pengguna, dimana data yang ditampilkan adalah data informasi penawarakan tempat tinggal dari pengguna.

4.2.5.3 Algoritma Proses Hapus data

Algoritma hapus data dijalankan saat pengguna menekan tombol *delete*. Sistem aplikasi akan mengirim data menuju sistem basis data dan menyimpan data yang dikirim. Algoritma ini akan digunakan berulang pada beberapa fitur admin yang berelasi dengan basis data. Algoritma ini disimpan dalam file javascript (*.js) dan PHP (*.php).

Implementasi algoritma *delete* pada javascript dan PHP akan ditunjukkan pada kode program dalam Gambar 4.29 dan 4.30 berikut.

```
1 $(document).on('click', '.delete-post', function() {
2     var x = confirm('Are you sure delete this post?');
3     if(x){
4         var postid = $(this).data("postdel");
5         $.ajax({
6             url: base_url+'conf/c_post/delete_post',
7             type: 'POST',
8             async: 'true',
9             dataType: 'html',
10            cache: false,
11            data: {
12                postid : postid
13            },
14            success: function (result) {
15                if(result=='success'){
16                    $('#'+postid).fadeOut();
17                };
18            }
19        });
20    }
21 }
```

```

19         });
20     }
21 });

```

Gambar 4.29 Implementasi Algoritma Proses Hapus data JS

```

1  function delete($param=NULL, $id=NULL){
2  if($param=='post'){
3  $sqlc = "DELETE FROM tbl_comment WHERE
4  MID(MD5(tbl_comment.`post_id`),6,6) = '". $id."'";
5  $sqlp = "DELETE FROM tbl_post_picture WHERE
6  MID(MD5(tbl_post_picture.`post_id`),6,6) = '". $id."'";
7  $sqlr = "DELETE FROM tbl_rating WHERE
8  MID(MD5(tbl_rating.`post_id`),6,6) = '". $id."'";
9  $this->db->query($sqlc);
10 $this->db->query($sqlp);
11 $this->db->query($sqlr);
12 $sql = "DELETE FROM tbl_post WHERE
13 MID(MD5(tbl_post.`post_id`),6,6) = '". $id."'";
14 if($this->db->query($sql)){
15     return "success";
16 }else return "failed";
17 }//----POST-----
18 elseif($param=='comment'){
19 $sql = "DELETE FROM tbl_comment WHERE
20 MID(MD5(tbl_comment.`comment_id`),6,6) = '". $id."'";
21 if($this->db->query($sql)){
22     return "success";
23 }else return "failed";
24 }//----COMMENT-----
25 elseif($param=='pict'){
26 $sql = "DELETE FROM tbl_post_picture WHERE
27 tbl_post_picture.`picture_id` = '". $id."'";
28 if($this->db->query($sql){ return "success";
29 }else return "failed";
30 }//----COMMENT-----
31 }

```

Gambar 4.30 Implementasi Algoritma Proses Hapus data PHP

Gambar di atas menunjukkan implementasi dari algoritma proses untuk menghapus data. Fungsi delete hanya bisa dilakukan oleh seorang admin menggunakan bahasa pemrograman berbentuk PHP. Pada baris ke 5 fungsi ajax akan mengirimkan data sebagai parameter untuk menghapus data seperti pada baris 12 dalam gambar 4.28. Data tersebut akan dikirim dan diolah dalam PHP yang menggunakan framework codeigniter seperti pada baris 1 – 44 dalam gambar 4.29.

4.2.6 Implementasi Antarmuka Aplikasi

Pada implementasi antarmuka aplikasi akan ditampilkan hasil implementasi antarmuka aplikasi *mobile rooamate*.

4.2.6.1 Halaman *Splash Screen*

Halaman *splash screen* merupakan halaman yang pertama kali dijalankan sebelum aplikasi masuk ke menu *home*.



Gambar 4.31 Tampilan *Splash Screen*

4.2.6.2 Halaman Menu Utama

Halaman menu utama merupakan halaman yang dijalankan setelah *splash screen* dan halaman ini adalah halaman utama untuk melanjutkan ke fungsi lainnya. Terdapat 6 submenu yaitu *find place*, *route*, *nearby*, *offerplace*, *login* dan

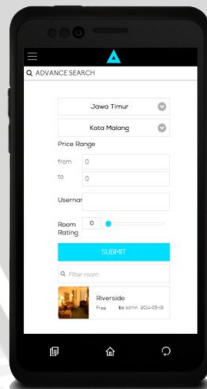
jika pengguna telah melakukan *login* akan berubah menjadi *user* dan yang terakhir adalah *about*.



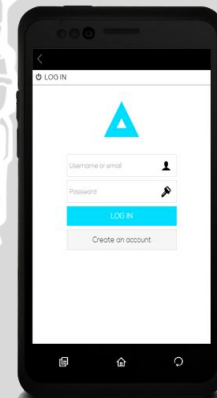
Gambar 4.32 Tampilan Menu Utama

4.2.6.3 Halaman Form

Terdapat beberapa halaman form, pada penulisan skripsi ini dicantumkan beberapa implementasi dari form yaitu pencarian dan login



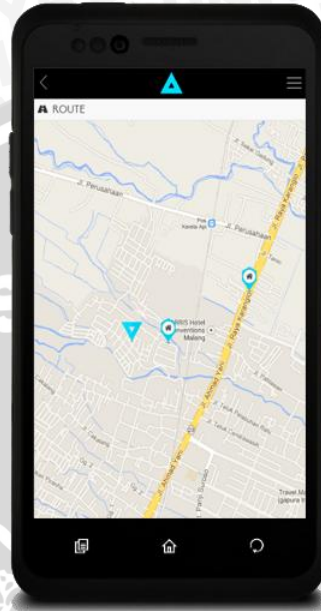
Gambar 4.33 Tampilan Pencarian



Gambar 4.34 Tampilan Login

4.2.6.4 Halaman Pencarian Berdasarkan Peta

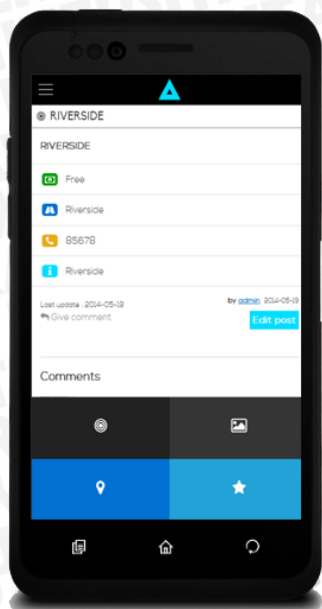
Terdapat beberapa halaman form, pada penulisan skripsi ini dicantumkan beberapa implementasi dari form yaitu pencarian dan login.



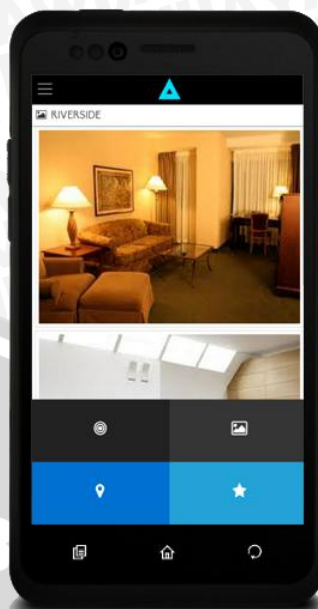
Gambar 4.35 Tampilan Pencarian Berdasarkan Peta

4.2.6.5 Halaman Detail Penawaran Tempat Menginap

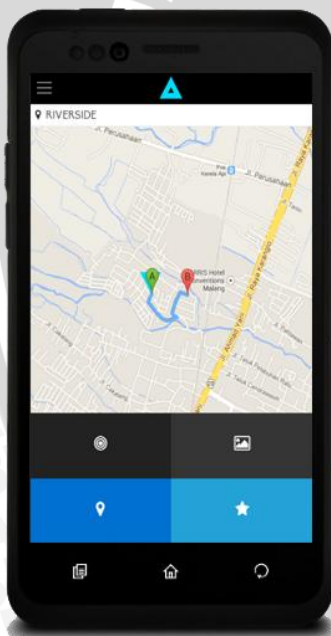
Terdapat beberapa tampilan dari detail penawaran tempat menginap, yaitu *tab general information*, *tab gambar pendukung*, *tab rute* dan *tab rating*.



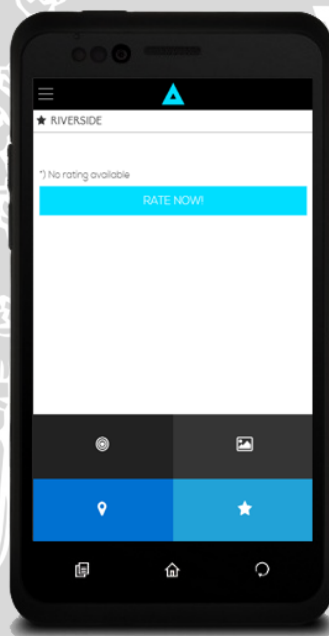
Gambar 4.36 Tampilan Tab General Information



Gambar 4.38 Tampilan Tab Gambar Pendukung



Gambar 4.37 Tampilan Tab Rute



Gambar 4.39 Tampilan Tab Rating

4.2.7 Implementasi Fitur Mobile

Fitur khusus yang dapat dijalankan pada aplikasi *mobile* yang terdapat pada penelitian ini disediakan oleh *framework* phonegap sebagai media yang dapat mengakses fitur *handphone* dengan pendekatan *hybridapp*. Implementasi fitur khusus yang ada pada perangkat *mobile* pada aplikasi ini adalah fitur *geolocation* dan *file*.

4.2.7.1 File

File pada phonegap adalah fitur yang memperbolehkan aplikasi untuk mengakses *storage handphone* secara bebas. Akses storage yang digunakan pada aplikasi yaitu membaca *file*, transfer *file*, membuat folder, *upload file* dan *download file*. Fitur – fitur tersebut digunakan pada saat melihat halaman *detail page* pada aplikasi, dimana pengguna dapat melakukan berbagai macam aktifitas berkaitan dengan file yang berupa gambar. Membaca, *upload file* dan transfer *file* digunakan saat pengguna akan memasukan gambar sedangkan membuat folder, *download file* dan transfer *file* digunakan saat pengguna mengunduh gambar yang diinginkan.

a. Upload File

Algoritma proses membaca, *upload file* dan transfer *file* akan dijalankan bersamaan saat pengguna akan melakukan upload gambar kedalam sistem. Algoritma ini disimpan dalam file javascript (*.js).

Implementasi algoritma *upload file* akan ditunjukkan pada kode program dalam Gambar 4.40 berikut.

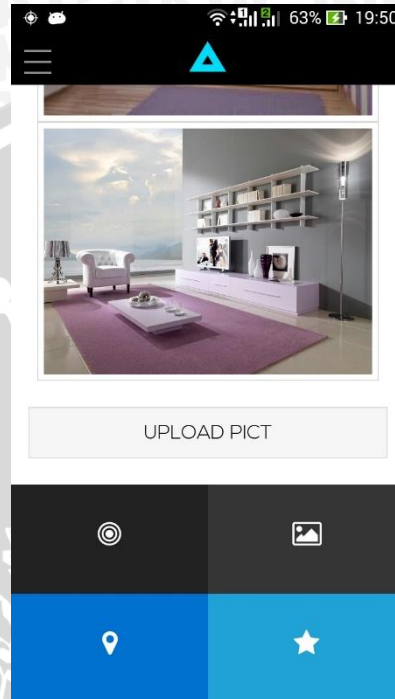
```
1 function photos(param) {
2   if(localStorage.user_id){
3     navigator.camera.getPicture(uploadPhoto,function(message) {
4       alert('get picture failed'); },
5         {quality : 50, destinationType :
6           navigator.camera.DestinationType.FILE_URI, sourceType :
7             navigator.camera.PictureSourceType.PHOTOLIBRARY });
8     function uploadPhoto(imageURI) {
9       $('loading-place').css('visibility', 'visible');
```

```
10     var options = new FileUploadOptions();
11         options.fileKey="file";
12     options.fileName=imageURI.substr(imageURI.lastIndexOf('/')+
13 1);
14         options.mimeType="image/jpeg";
15     var params = {};
16     if(param=='profpic'){
17         params.param = 'profpic';
18         params.userid = localStorage.user_id;
19     }else if(param=='postpic'){
20         params.param = 'postpic';
21         params.postid = sessionStorage.post_id;
22     }
23     options.params = params;
24     var ft = new FileTransfer();
25     ft.upload(imageURI,
26     encodeURI(urlRequest+"uploadpict.php"), win, fail,
27     options);
28 }
29 function win(r) {
30     $('.loading-place').css('visibility', 'hidden');
31     alert(r.response);
32     location.reload();
33 }
34 function fail(error) {
35     $('.loading-place').css('visibility', 'hidden');
36     alert("Failed");
37     location.reload();
38 }
39 }
40 };
50
```

Gambar 4.40 Implementasi Algoritma Upload File

Gambar di atas menunjukkan implementasi dari beberapa fitur phonegap yang menggunakan fitur *file* untuk *upload*. Pada baris ke 3 – 13 fungsi *navigator* akan menampilkan *library handphone* sesuai *setting* sumbernya. Pada baris 20 – 34

sistem akan melakukan seleksi dan mengirim gambar yang dipilih untuk di-*upload* menggunakan *file transfer*.



Gambar 4.41 Tampilan Upload File

b. Download File

Algoritma proses membuat folder, *download file* dan transfer *file* akan dijalankan bersamaan saat pengguna akan melakukan *download* gambar kedalam *storage handphone*. Algoritma ini disimpan dalam file javascript (*.js).

Implementasi algoritma *download file* akan ditunjukkan pada kode program dalam Gambar 4.42 berikut.

```

1 function download(URL, File_Name){
2   var Folder_Name = 'Rooamate';
3   var x = confirm("download this picture ? ");
4   if(x){window.requestFileSystem(LocalFileSystem.PERSISTENT, 0,
5     fileSystemSuccess, fileSystemFail);
6   function fileSystemSuccess(fileSystem) {
7     $('loading-place').css('visibility', 'visible');
8     var download_link = encodeURI(URL);

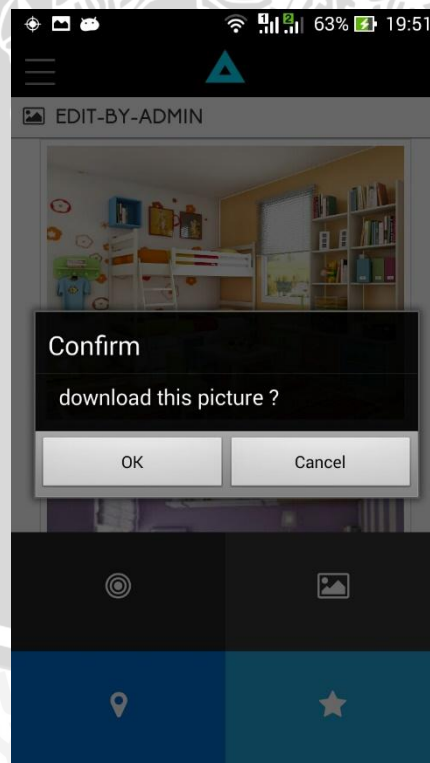
```

```
9  ext = download_link.substr(download_link.lastIndexOf('.') +
10  1); //Get extension of URL
11  var directoryEntry = fileSystem.root; // to get root path of
12  directory
13  directoryEntry.getDirectory(Folder_Name, { create: true,
14  exclusive: false }, onDirectorySuccess, onDirectoryFail);
15  // creating folder in sdcard
16  var rootdir = fileSystem.root;
17  var fp = rootdir.fullPath; // Returns Fullpath of local
18  directory
19
20  fp = fp + "/" + Folder_Name + "/" + File_Name + "." + ext; //
21  fullpath and name of the file which we want to give
22  // download function call
23  filetransfer(download_link, fp);
24  }
25  function onDirectorySuccess(parent) {
26  // Directory created successfully
27  }
28  function onDirectoryFail(error) {
29  //Error while creating directory
30  alert("Unable to create new directory: " + error.code);
31  $('.loading-place').css('visibility', 'hidden');
32  }
33  function fileSystemFail(evt) {
34  //Unable to access file system
35      alert(evt.target.error.code);
36      $('.loading-place').css('visibility', 'hidden');
37  }
38  function filetransfer(download_link, fp) {
39  var fileTransfer = new FileTransfer();
40  // File download function with URL and local path
41  fileTransfer.download(download_link, fp, function (entry) {
42  alert("download complete: " + entry.fullPath);
43  $('.loading-place').css('visibility', 'hidden');
44  },
45  function (error) {
46  //Download abort errors or download failed errors
```

```
47 alert("download error source " + error.source);
48 //alert("download error target " + error.target);
49 //alert("upload error code" + error.code);
50 $('.loading-place').css('visibility', 'hidden');
51     }
52     );
53     }
54     }
55 };
56
```

Gambar 4.42 Implementasi Algoritma Download File

Gambar di atas menunjukkan implementasi dari beberapa fitur phonegap yang menggunakan fitur *file* untuk *download*. Pada baris ke 5 – 20 sistem melakukan *setting* mulai dari mendapatkan *url* gambar dan *setting path* atau lokasi destinasi gambar yang akan di-*download*. Pada baris 47 – 66 sistem akan mengirim gambar yang dipilih untuk di-*download* menggunakan *file transfer*.



Gambar 4.43 Tampilan Download File

4.2.7.2 Geolocation

Geolocation pada phonegap adalah fitur yang memperbolehkan aplikasi untuk mengakses lokasi pengguna secara bebas. Fitur ini akan mengakses perangkat gps pada handphone sehingga pengguna dapat mendapatkan lokasinya secara tepat dalam bentuk *longitude* dan *latitude*.

Algoritma proses *geolocation* akan dijalankan bersamaan saat pengguna mengakses halaman yang mengakses *map*. Algoritma ini disimpan dalam file javascript (*.js). Implementasi algoritma akan ditunjukkan pada kode program dalam Gambar 4.44 berikut.

```
1 $(document).on("pageinit", "#route_page", function() {
2     navigator.geolocation.getCurrentPosition(locSuccess,
3     locError, {enableHighAccuracy: true});
4     document.addEventListener("backbutton", onBackKeyDown,
5     false);
6 });
7 function locSuccess(position) {
8     initialize(position.coords.latitude,
9     position.coords.longitude);
10 }
11 function initialize(lat, lon)
12 {
13     directionsDisplay = new
14     google.maps.DirectionsRenderer();
15     directionsService = new
16     google.maps.DirectionsService();
17     currentPosition = new google.maps.LatLng(lat, lon);
18     map = new
19     google.maps.Map(document.getElementById('map'), {
20         zoom: 15,
21         center: currentPosition,
22         mapTypeId: google.maps.MapTypeId.ROADMAP
23     });
24     directionsDisplay.setMap(map);
25     get_address(currentPosition);
26     var currentPositionMarker = new google.maps.Marker({
```

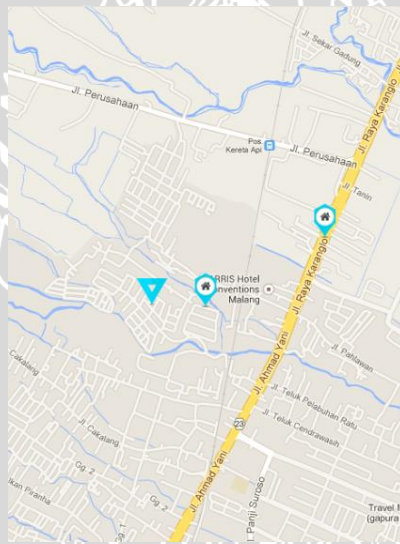
```
27     position: currentPosition,
28     map: map,
29     title: 'You are here!',
30     icon: 'img/pin/pin-user.png'
31   });
32
33   var infowindow = new google.maps.InfoWindow();
34   google.maps.event.addListener(currentPositionMarker,
35 'click', function() {
36   // infowindow.setContent("Current position: latitude: " +
37 lat +" longitude: " + lon);
38   infowindow.setContent('<div class = "MarkerPopUp" style=
39 "width: 80px;height: 20px" ><center>You are
40 here</center></div>');
41     infowindow.open(map, currentPositionMarker);
42   });
43   //-----
44   -----
45   $.getJSON( urlRequest+'search_room.php',
46 function(data) {
47     $.each( data, function(i, marker) {
48       roomMarker = new google.maps.Marker({
49         position: new google.maps.LatLng(marker.latitude,
50 marker.longitude),
51         map: map,
52         icon: 'img/pin/pin-room.png'});
53       if(marker.price==0){marker.price = "free";}
54       var info = (
55         '<div class="MarkerPopUp">'+
56         '<bodytext><table>'+
57         '<tr>'+
58         '<td style="vertical-align:top;"><span class="icon-
59 detail" style="background: #c60d0d;"><i class="fa fa-
60 bullseye"></i></span></td>'+
61         '<td>'+marker.title+'</td>'+
62         '</tr>'+
63         '<tr>'+
64         '<td style="vertical-align:top;"><span class="icon-
```

```
65 detail" style="background: #009900;"><i class="fa fa-
66 money"></i></span></td>' +
67   '<td>' + marker.price + '</td>' +
68   '</tr>' +
69   '<tr>' + '<td style="vertical-align:top;"><span
70 class="icon-detail" style="background: #0071D1;"><i
71 class="fa fa-road"></i></span></td>' +
72   '<td>' + marker.address + '</td>' +
73   '</tr>' +
74   '<tr>' +
75   '<td style="vertical-align:top;"><span class="icon-detail"
76 style="background: #e8aa17;"><i class="fa fa-
77 phone"></i></span></td>' +
78   '<td>' + marker.handphone_num + '</td>' +
79   '</tr>' +
80   '</table>' +
81   '<a href="#" class="ui-btn ui-link-icon"
82 onclick="calculateRoute(' + marker.latitude + ', ' + marker.longi
83 tude + ')" style="background: #0071D1;"><span style="color:
84 white;"><i class="fa fa-road"></i></span></a>' +
85   '<a href="detail.html" class="ui-btn ui-link-icon"
86 onclick="get_id(\' + marker.post_id + \', \' +
87 marker.latitude + \', \' + marker.longitude + \', \' +
88 marker.user_id + \', \'\')" style="background: #c6d0d0;"
89 data-ajax="false" rel="external"><span style="color:
90 white;"><i class="fa fa-bullseye"></i></span></a>' +
91   '</bodytext></div>'
92 );
93   infowindowroom = new google.maps.InfoWindow({
94     content: info
95   });
96   google.maps.event.addListener(roomMarker, 'click',
97   function() {
98     infowindowroom.setContent(info);
99     infowindowroom.open(map, this);
100   });
101
102   });
```


103	});
104	//-----
105	-----

Gambar 4.44 Implementasi Geolocation

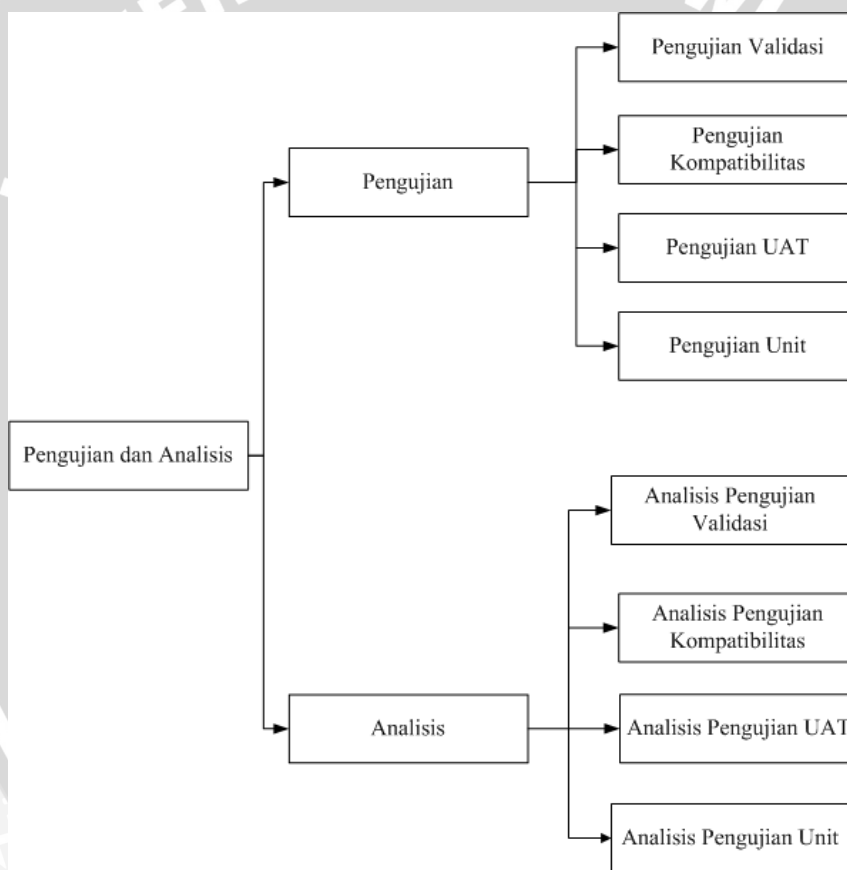
Gambar di atas menunjukkan implementasi dari fitur phonegap yang menggunakan fitur *geolocation*. Pada baris ke 1 – 11 sistem melakukan *request* untuk akses lokasi melalui fitur yang ada pada handphone. Pada baris 19 – 34 sistem akan menampilkan lokasi dan posisi pengguna dalam bentuk *marker* berdasarkan *longitude* dan *latitude* yang telah didapatkan pada proses sebelumnya. Pada baris 47 – 100 sistem akan memasukan *marker* dari lokasi penawaran tempat menginap ke dalam *map* sehingga pengguna dapat melihat lokasi tersebut.



Gambar 4.45 Tampilan Map dengan Geolocation

BAB V PENGUJIAN DAN ANALISIS

Pada bab ini dilakukan proses pengujian dan analisis terhadap aplikasi *mobile* yang telah dibangun. Proses pengujian dilakukan melalui dua tahapan (strategi) yaitu pengujian validasi dan pengujian kompatibilitas. Pada pengujian validasi dan kompatibilitas digunakan teknik pengujian *Black Box (Black Box Testing)* dan dilakukan juga pengujian unit (*White Box Testing*).



Gambar 5.1 Diagram Pohon Pengujian dan Analisis

5.1 Pengujian

Proses pengujian dilakukan melalui tiga tahapan (strategi) yaitu pengujian validasi, pengujian kompatibilitas dan pengujian unit.



5.1.1 Pengujian Validasi

Pengujian validasi digunakan mengetahui apakah sistem yang dibangun sudah benar sesuai dengan yang dibutuhkan. Daftar kebutuhan yang telah dirumuskan dan merupakan hasil analisis kebutuhan akan menjadi acuan untuk melakukan pengujian validasi. Pengujian validasi menggunakan metode pengujian *Black Box*, karena tidak diperlukan konsentrasi terhadap alur jalannya algoritma program dan lebih ditekankan untuk menemukan kesesuaian antara kinerja sistem dengan daftar kebutuhan. Pada pengujian validasi hanya dilakukan pengujian dengan kondisi yang normal dan tidak normal. Pengujian validasi dilakukan pada perangkat Android dan iOS.

1. Kasus Uji Validasi Lihat Profil Pribadi

Tabel 5.1 Kasus Uji Validasi Lihat Profil Pribadi

Nama Kasus Uji	Kasus uji lihat profil pribadi
Objek Uji	Kebutuhan Fungsional (SRS_001_01)
Tujuan Pengujian	Pengujian dilakukan untuk memastikan bahwa aplikasi dapat memenuhi kebutuhan fungsional dalam menyediakan fasilitas untuk melihat profil pribadi pengguna.
Prosedur Uji	<ol style="list-style-type: none"> 1. Buka aplikasi 2. Masuk pada halaman <i>home</i> 3. Pilih menu <i>profile</i> pada halaman <i>home</i>
Hasil yang Diharapkan	Aplikasi dapat menampilkan informasi pribadi
Status Validitas (Android)	Valid
Status Validitas (iOs)	Valid

2. Kasus Uji Validasi Edit Profil Pribadi

Tabel 5.2 Kasus Uji Validasi Edit Profil Pribadi

Nama Kasus Uji	Kasus uji edit profil pribadi
Objek Uji	Kebutuhan Fungsional (SRS_001_01)
Tujuan Pengujian	Pengujian dilakukan untuk memastikan bahwa aplikasi dapat memenuhi kebutuhan fungsional dalam menyediakan fasilitas untuk melakukan perubahan data profil pribadi pengguna.
Prosedur Uji	<ol style="list-style-type: none"> 1. Buka aplikasi 2. Masuk pada halaman <i>home</i> 3. Pilih menu <i>profile</i> pada halaman <i>home</i> 4. Pilih menu <i>edit</i> pada profil pribadi 5. Sistem akan menampilkan halaman form ubah profil pribadi 6. Masukkan informasi berbeda pada halaman form 7. Pilih menu simpan 8. Sistem akan menyimpan informasi data yang baru
Hasil yang Diharapkan	Aplikasi dapat mengubah informasi pribadi profil pribadi sesuai data yang dimasukkan pengguna
Status Validitas (Android)	Valid

Status Validitas (iOs)	Valid
-------------------------------	-------

3. Kasus Uji Validasi Lihat Penawaran Tempat Menginap

Tabel 5.3 Kasus Uji Validasi Lihat Penawaran Tempat Menginap

Nama Kasus Uji	Kasus uji lihat penawaran tempat menginap
Objek Uji	Kebutuhan Fungsional (SRS_002_01)
Tujuan Pengujian	Pengujian dilakukan untuk memastikan bahwa aplikasi dapat memenuhi kebutuhan fungsional dalam menyediakan fasilitas untuk melihat penawaran tempat menginap.
Prosedur Uji	<ol style="list-style-type: none"> 1. Buka aplikasi 2. Masuk pada halaman <i>home</i> 3. Pilih menu <i>find place</i> pada halaman <i>home</i> 4. Masukkan parameter pencarian dan pilih penawaran tempat menginap 5. Sistem akan menampilkan informasi penawaran tempat menginap
Hasil yang Diharapkan	Aplikasi dapat menampilkan informasi Penawaran Tempat Menginap
Status Validitas (Android)	Valid
Status Validitas (iOs)	Valid

4. Kasus Uji Validasi Tambah Penawaran Tempat Menginap

Tabel 5.4 Kasus Uji Validasi Tambah Penawaran Tempat Menginap

Nama Kasus Uji	Kasus uji tambah penawaran tempat menginap
Objek Uji	Kebutuhan Fungsional (SRS_002_01)
Tujuan Pengujian	Pengujian dilakukan untuk memastikan bahwa aplikasi dapat memenuhi kebutuhan fungsional dalam menyediakan fasilitas untuk menambah penawaran tempat menginap.
Prosedur Uji	<ol style="list-style-type: none"> 1. Buka aplikasi 2. Masuk pada halaman <i>home</i> 3. Pilih menu <i>offer place</i> pada halaman <i>home</i> 4. Masukkan informasi sesuai dengan daftar isian yang diminta 5. Pilih menu <i>simpan</i> 6. Sistem akan menyimpan data baru
Hasil yang Diharapkan	Aplikasi dapat menyimpan informasi data penawaran tempat menginap dan ditampilkan pada saat pengguna melakukan pencarian penawaran tempat menginap
Status Validitas (Android)	Valid
Status Validitas (iOs)	Valid

5. Kasus Uji Validasi Edit Penawaran Tempat Menginap

Tabel 5.5 Kasus Uji Validasi Edit Penawaran Tempat Menginap

Nama Kasus Uji	Kasus uji edit penawaran tempat menginap
Objek Uji	Kebutuhan Fungsional (SRS_002_01)
Tujuan Pengujian	Pengujian dilakukan untuk memastikan bahwa aplikasi dapat memenuhi kebutuhan fungsional dalam menyediakan fasilitas untuk melakukan perubahan data penawaran tempat menginap.
Prosedur Uji	<ol style="list-style-type: none"> 1. Buka aplikasi 2. Masuk pada halaman <i>home</i> 3. Pilih menu <i>find place</i> pada halaman <i>home</i> 4. Masukkan parameter pencarian dan pilih penawaran tempat menginap 5. Sistem akan menampilkan informasi penawaran tempat menginap 6. Pilih menu <i>edit</i> pada penawaran tempat menginap 7. Sistem akan menampilkan halaman form ubah penawaran tempat menginap 8. Masukkan informasi berbeda pada halaman form 9. Pilih menu simpan 10. Sistem akan menyimpan informasi data yang baru
Hasil yang Diharapkan	Aplikasi dapat mengubah informasi penawaran tempat menginap sesuai data yang dimasukkan pengguna

Status Validitas (Android)	Valid
Status Validitas (iOs)	Valid

6. Kasus Uji Validasi Lihat Penawaran Tempat Menginap Melalui Pencarian Parameter

Tabel 5.6 Kasus Uji Validasi Lihat Penawaran Tempat Menginap Melalui Pencarian Parameter

Nama Kasus Uji	Kasus uji lihat penawaran tempat menginap melalui pencarian parameter
Objek Uji	Kebutuhan Fungsional (SRS_003_01)
Tujuan Pengujian	Pengujian dilakukan untuk memastikan bahwa aplikasi dapat memenuhi kebutuhan fungsional dalam menyediakan fasilitas untuk melihat penawaran tempat menginap.
Prosedur Uji	<ol style="list-style-type: none"> 1. Buka aplikasi 2. Masuk pada halaman <i>home</i> 3. Pilih menu <i>find place</i> pada halaman <i>home</i> 4. Pilih menu <i>advance search</i> 5. Masukkan parameter pencarian dan pilih penawaran tempat menginap 6. Sistem akan menampilkan informasi penawaran tempat menginap
Hasil yang	Aplikasi dapat menampilkan informasi

Diharapkan	Penawaran Tempat Menginap
Status Validitas (Android)	Valid
Status Validitas (iOs)	Valid

7. Kasus Uji Validasi Lihat Penawaran Tempat Menginap Melalui Pencarian Peta

Tabel 5.7 Kasus Uji Validasi Lihat Penawaran Tempat Menginap Melalui Pencarian Peta

Nama Kasus Uji	Kasus uji lihat penawaran tempat menginap melalui pencarian peta
Objek Uji	Kebutuhan Fungsional (SRS_003_02)
Tujuan Pengujian	Pengujian dilakukan untuk memastikan bahwa aplikasi dapat memenuhi kebutuhan fungsional dalam menyediakan fasilitas untuk melihat penawaran tempat menginap.
Prosedur Uji	<ol style="list-style-type: none"> 1. Buka aplikasi 2. Masuk pada halaman <i>home</i> 3. Pilih menu <i>route</i> pada halaman <i>home</i> 4. Sistem akan menampilkan <i>map</i> dengan <i>marker</i> lokasi penawaran tempat menginap 5. Pilih salah satu <i>marker</i> dan pilih menu <i>detail</i> 6. Sistem akan menampilkan informasi penawaran tempat menginap
Hasil yang	Aplikasi dapat menampilkan <i>map</i> dan informasi

Diharapkan	Penawaran Tempat Menginap
Status Validitas (Android)	Valid
Status Validitas (iOs)	Valid

8. Kasus Uji Validasi Lihat Penawaran Tempat Menginap Berdasarkan Lokasi Terdekat

Tabel 5.8 Kasus Uji Validasi Lihat Penawaran Tempat Menginap Berdasarkan Lokasi Terdekat

Nama Kasus Uji	Kasus uji lihat penawaran tempat menginap berdasarkan lokasi terdekat
Objek Uji	Kebutuhan Fungsional (SRS_003_03)
Tujuan Pengujian	Pengujian dilakukan untuk memastikan bahwa aplikasi dapat memenuhi kebutuhan fungsional dalam menyediakan fasilitas untuk melihat penawaran tempat menginap.
Prosedur Uji	<ol style="list-style-type: none"> 1. Buka aplikasi 2. Masuk pada halaman <i>home</i> 3. Pilih menu <i>Nearby</i> pada halaman <i>home</i> 4. Sistem akan menampilkan penawaran tempat menginap berdasarkan lokasi pengguna 5. Pilih salah satu penawaran tempat menginap 6. Sistem akan menampilkan informasi penawaran tempat menginap

Hasil yang Diharapkan	Aplikasi dapat menampilkan informasi Penawaran Tempat Menginap
Status Validitas (Android)	Valid
Status Validitas (iOs)	Valid

9. Kasus Uji Validasi Lihat Detail Penawaran Tempat Menginap

Tabel 5.9 Kasus Uji Validasi Lihat Detail Penawaran Tempat Menginap

Nama Kasus Uji	Kasus uji lihat detail tempat menginap
Objek Uji	Kebutuhan Fungsional (SRS_004_01)
Tujuan Pengujian	Pengujian dilakukan untuk memastikan bahwa aplikasi dapat memenuhi kebutuhan fungsional dalam menyediakan fasilitas untuk melihat detail penawaran tempat menginap.
Prosedur Uji	<ol style="list-style-type: none"> 1. Buka aplikasi 2. Masuk pada halaman <i>home</i> 3. Pilih menu <i>Nearby</i> pada halaman <i>home</i> 4. Sistem akan menampilkan penawaran tempat menginap berdasarkan lokasi pengguna 5. Pilih salah satu penawaran tempat menginap 6. Sistem akan menampilkan detail informasi penawaran tempat menginap
Hasil yang	Aplikasi dapat menampilkan detail informasi



Diharapkan	Penawaran Tempat Menginap
Status Validitas (Android)	Valid
Status Validitas (iOs)	Valid

10. Kasus Uji Validasi Lihat Komentar

Tabel 5.10 Kasus Uji Validasi Lihat Komentar

Nama Kasus Uji	Kasus uji lihat komentar
Objek Uji	Kebutuhan Fungsional (SRS_004_02)
Tujuan Pengujian	Pengujian dilakukan untuk memastikan bahwa aplikasi dapat memenuhi kebutuhan fungsional dalam menyediakan fasilitas untuk melihat komentar.
Prosedur Uji	<ol style="list-style-type: none"> 1. Buka aplikasi 2. Masuk pada halaman <i>home</i> 3. Pilih menu <i>Nearby</i> pada halaman <i>home</i> 4. Sistem akan menampilkan penawaran tempat menginap berdasarkan lokasi pengguna 5. Pilih salah satu penawaran tempat menginap 6. Sistem akan menampilkan detail informasi penawaran tempat menginap 7. Sistem akan menampilkan komentar
Hasil yang Diharapkan	Aplikasi dapat menampilkan informasi komentar

Status Validitas (Android)	Valid
Status Validitas (iOs)	Valid

11. Kasus Uji Validasi Tambah Komentar

Tabel 5.11 Kasus Uji Validasi Tambah Komentar

Nama Kasus Uji	Kasus uji tambah komentar
Objek Uji	Kebutuhan Fungsional (SRS_004_02)
Tujuan Pengujian	Pengujian dilakukan untuk memastikan bahwa aplikasi dapat memenuhi kebutuhan fungsional dalam menyediakan fasilitas untuk menambah komentar.
Prosedur Uji	<ol style="list-style-type: none"> 1. Buka aplikasi 2. Masuk pada halaman <i>home</i> 3. Pilih menu <i>Nearby</i> pada halaman <i>home</i> 4. Sistem akan menampilkan penawaran tempat menginap berdasarkan lokasi pengguna 5. Pilih salah satu penawaran tempat menginap 6. Sistem akan menampilkan detail informasi penawaran tempat menginap 7. Sistem akan menampilkan komentar 8. Masukkan informasi sesuai dengan daftar isian yang diminta 9. Pilih menu simpan 10. Sistem akan menyimpan data komentar baru

Hasil yang Diharapkan	Aplikasi dapat menyimpan informasi data komentar dan ditampilkan pada saat pengguna melihat detail penawaran tempat menginap
Status Validitas (Android)	Valid
Status Validitas (iOs)	Valid

12. Kasus Uji Validasi Lihat Deskripsi Aplikasi

Tabel 5.12 Kasus Uji Validasi Lihat Deskripsi Aplikasi

Nama Kasus Uji	Kasus uji lihat deskripsi aplikasi
Objek Uji	Kebutuhan Fungsional (SRS_005_01)
Tujuan Pengujian	Pengujian dilakukan untuk memastikan bahwa aplikasi dapat memenuhi kebutuhan fungsional dalam menyediakan fasilitas untuk melihat deskripsi.
Prosedur Uji	<ol style="list-style-type: none"> 1. Buka aplikasi 2. Masuk pada halaman <i>home</i> 3. Pilih menu <i>About</i> pada halaman <i>home</i> 4. Sistem akan menampilkan informasi deskripsi aplikasi
Hasil yang Diharapkan	Aplikasi dapat menampilkan informasi deskripsi aplikasi

Status Validitas (Android)	Valid
Status Validitas (iOs)	Valid

13. Kasus Uji Validasi Lihat *Request* Penawaran Tempat Tinggal

Tabel 5.13 Kasus Uji Validasi Lihat *Request* Penawaran Tempat Tinggal

Nama Kasus Uji	Kasus uji lihat <i>request</i> penawaran tempat tinggal
Objek Uji	Kebutuhan Fungsional (SRS_006_01)
Tujuan Pengujian	Pengujian dilakukan untuk memastikan bahwa aplikasi dapat memenuhi kebutuhan fungsional dalam menyediakan fasilitas untuk melihat <i>request</i> penawaran tempat tinggal.
Prosedur Uji	<ol style="list-style-type: none"> 1. Buka aplikasi 2. Masuk pada halaman <i>home</i> 3. Pilih menu <i>Nearby</i> pada halaman <i>home</i> 4. Sistem akan menampilkan penawaran tempat menginap berdasarkan lokasi pengguna 5. Pilih salah satu penawaran tempat menginap 6. Sistem akan menampilkan detail informasi penawaran tempat menginap 7. Sistem akan menampilkan <i>request</i> penawaran tempat tinggal
Hasil yang Diharapkan	Aplikasi dapat menampilkan informasi <i>request</i> penawaran tempat tinggal

Status Validitas (Android)	Valid
Status Validitas (iOs)	Valid

14. Kasus Uji Validasi Tambah *Request* Penawaran Tempat Tinggal

Tabel 5.14 Kasus Uji Validasi Tambah *Request* Penawaran Tempat Tinggal

Nama Kasus Uji	Kasus uji tambah <i>request</i> penawaran tempat tinggal
Objek Uji	Kebutuhan Fungsional (SRS_006_01)
Tujuan Pengujian	Pengujian dilakukan untuk memastikan bahwa aplikasi dapat memenuhi kebutuhan fungsional dalam menyediakan fasilitas untuk menambah <i>request</i> penawaran tempat tinggal.
Prosedur Uji	<ol style="list-style-type: none"> 1. Buka aplikasi 2. Masuk pada halaman <i>home</i> 3. Pilih menu <i>Nearby</i> pada halaman <i>home</i> 4. Sistem akan menampilkan penawaran tempat menginap berdasarkan lokasi pengguna 5. Pilih salah satu penawaran tempat menginap 6. Sistem akan menampilkan detail informasi penawaran tempat menginap 7. Sistem akan menampilkan <i>request</i> 8. Masukkan informasi sesuai dengan daftar isian yang diminta 9. Pilih menu simpan 10. Sistem akan menyimpan data <i>request</i> baru

Hasil yang Diharapkan	Aplikasi dapat menyimpan informasi data <i>request</i> penawaran tempat tinggal dan ditampilkan pada saat pengguna melihat detail penawaran tempat menginap
Status Validitas (Android)	Valid
Status Validitas (iOs)	Valid

5.1.2 Pengujian Kompabilitas

Pengujian kompatibilitas digunakan untuk mengetahui kompatibilitas antarmuka sistem pada perangkat bergerak Android dan iOS. Pengujian kompatibilitas dilakukan untuk mengetahui validitas kinerja fitur-fitur yang disediakan oleh sistem terkait dengan antarmuka sistem. Tabel 5.15 menjelaskan prosedur dan hasil kasus uji pengujian kompatibilitas pada sistem operasi Android dan iOS.

Tabel 5.15 Kasus Uji dan Hasil Pengujian Kompabilitas Android dan iOS

Nama Kasus Uji	Pengujian kompatibilitas Android dan iOS
Objek Uji	Kebutuhan Nonfungsional
Tujuan Pengujian	Pengujian dilakukan untuk mengetahui validitas kinerja dari sistem dalam menyediakan fitur-fitur dan antarmuka pengguna
Prosedur Uji	Membuka setiap halaman sesuai dengan spesifikasi kebutuhan sistem
Hasil yang diharapkan	Sistem dapat menampilkan fitur-fitur dan antarmuka sesuai dengan implementasi antarmuka system
Hasil yang didapatkan	Sistem dapat menampilkan fitur-fitur dan antarmuka sesuai dengan implementasi antarmuka system

Status Validitas (Android)	Valid
Status Validitas (iOS)	Valid

5.1.3 Pengujian UAT (*User Acceptance Test*)

Pengujian UAT dilakukan dengan melibatkan pengguna, pengujian dilakukan dengan cara pengisian kuisisioner dengan jumlah pertanyaan 8 butir serta jumlah korespondensi 20 orang secara acak. Pengujian UAT bertujuan untuk melihat tingkat penerimaan pengguna terhadap aplikasi, disamping itu terdapat beberapa variabel penilaian yaitu kemudahan penggunaan aplikasi (*perceived ease of use*) dan pencapaian kegunaan (*usefulness*). Pada Tabel 5.16 dijelaskan pernyataan-pernyataan yang berhubungan dengan variabel penilaian serta hasil dari pengujian. Pernyataan dan hasil akan disertakan sebagai lampiran.

Tabel 5.16 Hasil Pengujian Usability

No	Pernyataan	STS	TS	N	S	SS	Total
1	Saya merasa navigasi di dalam aplikasi mudah bagi saya	0	0	3	14	3	20
2	Saya merasa interaksi di dalam aplikasi jelas dan mudah di mengerti	0	1	4	11	4	20
3	Saya merasa sangat mudah menggunakan aplikasi ini	0	0	5	9	6	20
4	Saya merasa aplikasi ini dapat merespons informasi dengan cepat	0	0	1	11	8	20
5	Saya merasa aplikasi ini memungkinkan saya untuk mendapatkan informasi yang ingin saya dapatkan	0	0	3	11	6	20
6	Saya merasa aplikasi ini dapat membantu saya untuk mengumpulkan informasi lebih banyak	0	2	2	11	5	20
7	Saya merasa aplikasi ini memberikan respon kinerja yang cepat ketika saya	0	2	1	13	4	20

	operasikan						
8	Saya dapat terbantu dengan adanya sistem ini untuk mengetahui informasi lokasi dan rute menuju lokasi tempat menginap.	0	0	0	8	12	20

Keterangan:

- STS : Sangat Tidak Setuju S : Setuju
- TS : Tidak Setuju SS : Sangat Setuju
- N : Netral

5.1.4 Pengujian Unit

Pengujian unit dilakukan untuk melihat apakah algoritma berjalan dengan benar. Pengujian dilakukan dengan menggunakan *tools* QUnit dengan melakukan pengujian algoritma javascript. Dalam pengujian unit ini tidak dilakukan pada semua algoritma, penulis membatasi pengujian pada algoritma penting yang digunakan secara umum pada aplikasi yaitu *Create, Retrieve* dan *Update*. Pengujian yang dilakukan pada algoritma menampilkan data, *insert data request* dan *update data request*.

Kasus uji pertama adalah menguji fungsi *ajax load data* pada sistem. Pengujian dilakukan dengan memberikan data sebagai parameter yang selanjutnya akan diolah oleh server dengan pengembalian data berupa data penawaran tempat menginap. Hasil yang diharapkan dari pengujian ini adalah berjalannya fungsi *ajax* yang berjalan secara asinkronus dan sistem akan mengembalikan data berupa data penawaran tempat menginap.

Kasus uji unit fungsi *ajax load data* terdapat pada Gambar 5.2 berikut.

Gambar 5.2 Kasus Uji Unit *Ajax Load Data*

```

1 module("Rooamate");
2 asyncTest("Function Load data", function() {
3     var urlRequest =
4     ("http://www.rooamate.com/backpack/server/");
5     $.ajax({
6         url: urlRequest+'search_room.php',
7         dataType: "json",
8         type: "POST",

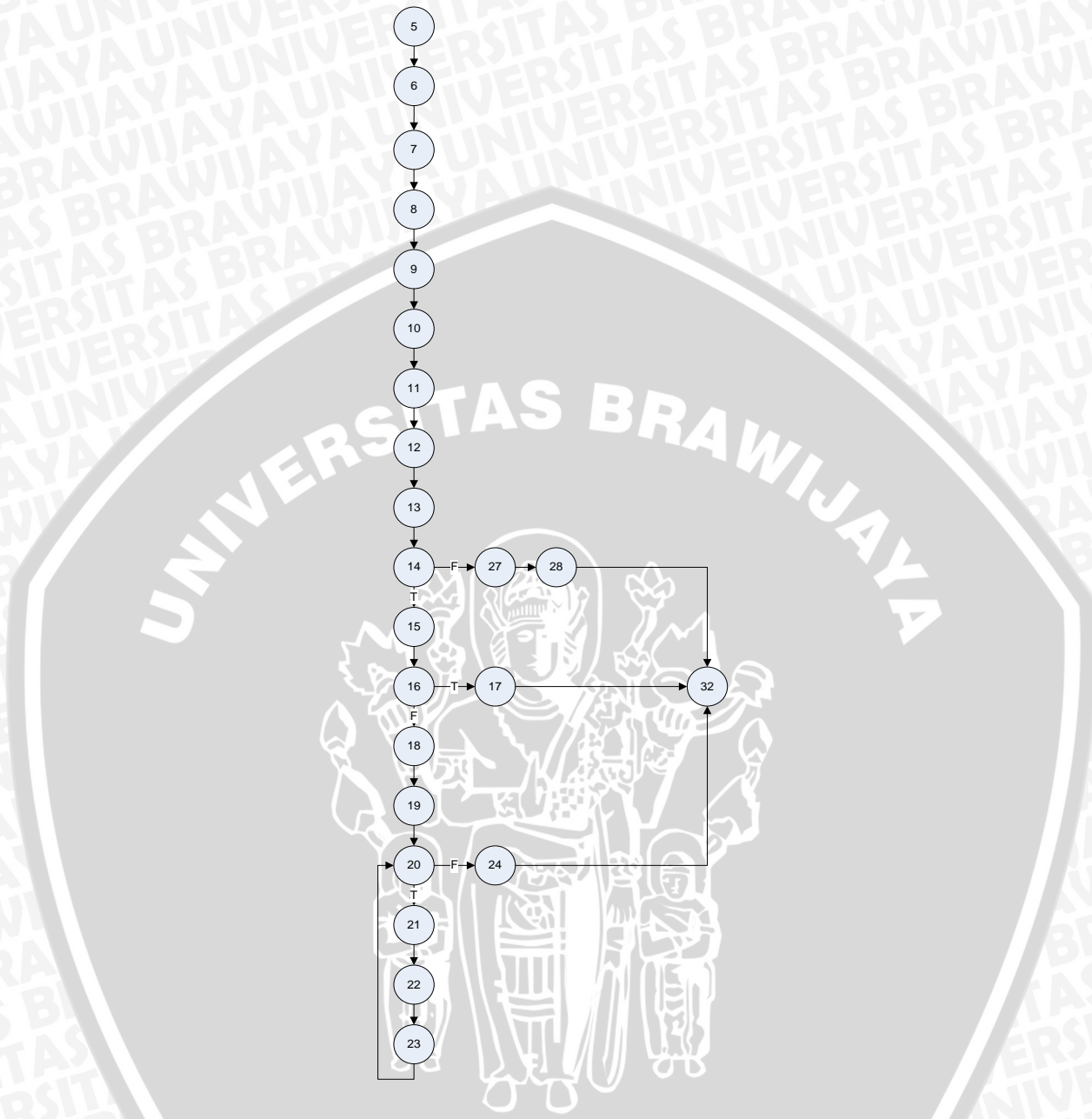
```

```
9     data: {
10         province_id : '13000015',
11         city_id     : '1300257'
12     },
13     timeout: 1000
14 }).done(function(data) {
15     var cek = JSON.stringify(data);
16     if(cek=="No rows found"){
17         ok(true, "No Rows Found");
18     }else{
19         var output = '';
20         $.each(data, function(i, f){
21             ok(true, "loaded");
22             ok(data.length>1, "array size");
23             ok(f.username == 'admin', "username data = True");
24         });
25     }
26
27 }).fail(function(x, text, thrown) {
28     ok(false, "ajax failed: " + text);
29 }).always(function(){
30     start();
31 });
32 });
```

Pada Gambar 5.2 kasus uji unit fungsi *ajax load data* menjelaskan algoritma dalam bentuk kode yang akan dilakukan *testing* pada pengujian unit.

1. Pada baris 1 – 9 menjelaskan bahwa test yang akan dilakukan adalah test asinkronus menggunakan ajax dengan data kembalian bertipe JSON dan tipe ajax berupa POST.
2. Pada baris 10 – 13 adalah parameter data yang akan dikirim.
3. Pada baris 16 – 26 adalah hasil kembalian dari test tersebut ketika hasil diketahui sama antara data kembalian dengan syarat kembalian maka hasil yang diharapkan tercapai.

Langkah pengujian yang pertama adalah menggambarkan diagram alir dari kode program diatas yang ditampilkan pada Gambar 5.3 dibawah ini



Gambar 5.3 Diagram Alir *Ajax Load Data*

Dari diagram alir pada Gambar 5.3 diatas maka akan didapatkan jumlah dari *cyclomatic complexity* yaitu:

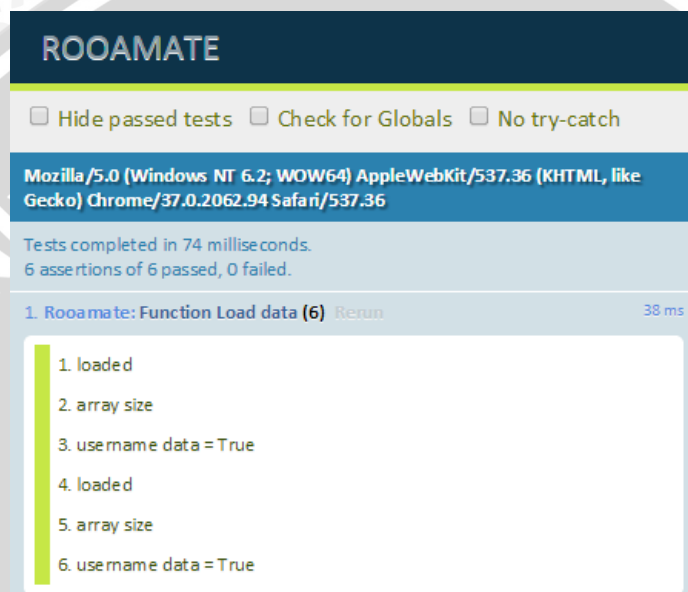
Perhitungan diatas menjelaskan bahwa dibutuhkan 4 skenario *test case* untuk menguji kode program fungsi *ajax load data*.



No	Nama Skenario	Field Provinsi	Field Kota
1	Data muncul	valid	valid

Skenario pengujian yang pertama dijalankan dengan mengisi *field* provinsi dan kota secara benar dan diharapkan data yang valid akan muncul.

Hasil pengujian unit pada kasus uji pertama dijelaskan dalam Gambar 5.4 di bawah ini



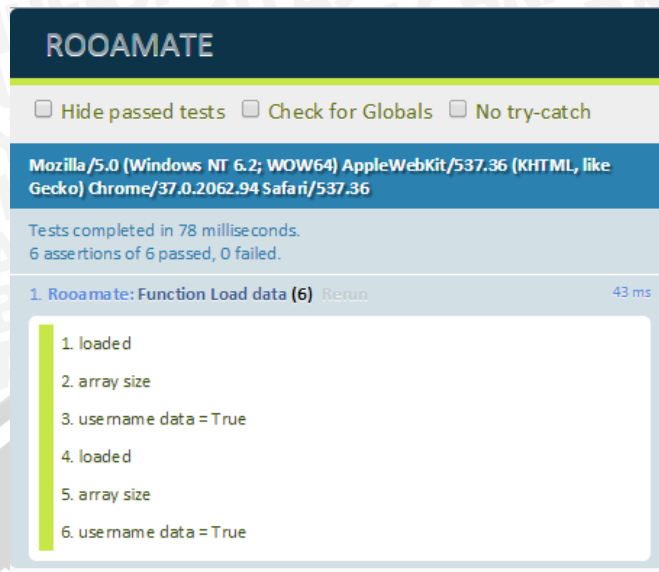
Gambar 5.4 Hasil Uji Unit Skenario 1 *Ajax Load Data*

Dalam Gambar 5.4 di atas menjelaskan hasil pengujian terhadap kasus uji yang dilakukan pada Gambar 5.2 dengan skenario yang telah disebutkan. Pengujian pada kasus *test case* pertama di dapatkan hasil alokasi waktu test sebesar 38 ms dengan 6 dari 6 pernyataan dinyatakan diterima (*passed*).

No	Nama Skenario	Field Provinsi	Field Kota
2	Data muncul, <i>field</i> kota kosong	valid	invalid

Skenario pengujian yang kedua dijalankan dengan mengisi *field* provinsi saja secara benar dan diharapkan data yang valid akan muncul.

Hasil pengujian unit pada kasus uji kedua dijelaskan dalam Gambar 5.5 di bawah ini



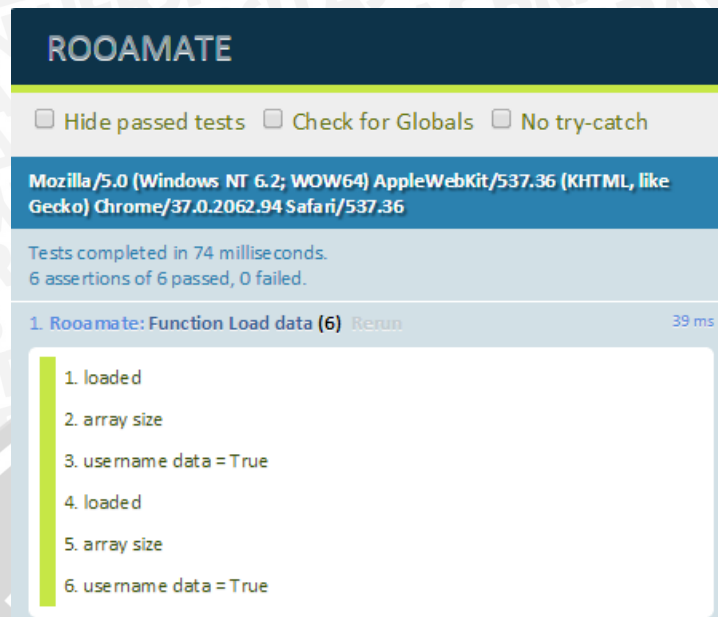
Gambar 5.5 Hasil Uji Unit Skenario 2 *Ajax Load Data*

Dalam Gambar 5.5 di atas menjelaskan hasil pengujian terhadap kasus uji yang dilakukan pada Gambar 5.2 dengan skenario yang telah disebutkan. Pengujian pada kasus *test case* kedua di dapatkan hasil alokasi waktu test sebesar 43 ms dengan 6 dari 6 pernyataan dinyatakan diterima (*passed*).

No	Nama Skenario	Field Provinsi	Field Kota
3	Data muncul, <i>field</i> Provinsi kosong	invalid	valid

Skenario pengujian yang ketiga dijalankan dengan mengisi *field* kota saja secara benar dan diharapkan data yang valid akan muncul.

Hasil pengujian unit pada kasus uji ketiga dijelaskan dalam Gambar 5.6 di bawah ini



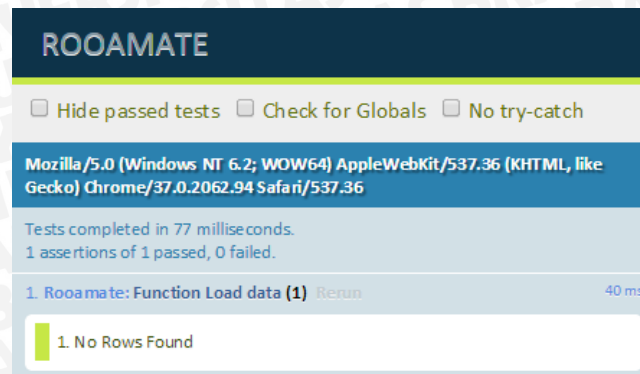
Gambar 5.6 Hasil Uji Unit Skenario 3 *Ajax Load Data*

Dalam Gambar 5.6 di atas menjelaskan hasil pengujian terhadap kasus uji yang dilakukan pada Gambar 5.2 dengan skenario yang telah disebutkan. Pengujian pada kasus *test case* ketiga di dapatkan hasil alokasi waktu test sebesar 39 ms dengan 6 dari 6 pernyataan dinyatakan diterima (*passed*).

No	Nama Skenario	Field Provinsi	Field Kota
4	Data tidak muncul, <i>field</i> Provinsi dan kota kosong	invalid	invalid

Skenario pengujian yang keempat dijalankan dengan tidak mengisi *field* provinsi dan kota sehingga data tidak muncul.

Hasil pengujian unit pada kasus uji keempat dijelaskan dalam Gambar 5.7 di bawah ini



Gambar 5.7 Hasil Uji Unit Skenario 4 *Ajax Load Data*

Dalam Gambar 5.7 di atas menjelaskan hasil pengujian terhadap kasus uji yang dilakukan pada Gambar 5.2 dengan skenario yang telah disebutkan. Pengujian pada kasus *test case* keempat di dapatkan hasil alokasi waktu test sebesar 40 ms dengan 1 dari 1 pernyataan dinyatakan diterima (*passed*).

Kasus uji kedua adalah menguji fungsi *ajax create request* pada sistem. Pengujian dilakukan dengan memberikan data sebagai parameter yang selanjutnya akan diolah oleh server dengan pengembalian data berupa keterangan sukses atau gagal. Hasil yang diharapkan dari pengujian ini adalah berjalannya fungsi *ajax* yang berjalan secara asinkronus dan terjadi penambahan data pada database dan kembalian *ajax* yaitu notifikasi “*Success*”.

Kasus uji unit fungsi *ajax create request* terdapat pada Gambar 5.8 berikut.

Gambar 5.8 Kasus Uji Unit *Ajax Create Request*

```

1 module("Rooamate");
2 asyncTest("Function Add request", function() {
3     var urlRequest = ("http://
4     www.rooamate.com/backpack/server/");
5     $.ajax({
6         url: urlRequest+'request.php',
7         dataType: "json",
8         type: "POST",
9         data: {
10            postid : "9bb137",
11            checkin : "06/07/2014",
12            checkout : "06/09/2014",
13            userid : "2014060002"
14        },
15        timeout: 1000
16    }).done(function(result) {

```

```

17     if(result=='Success'){
18         ok(true, "loaded");
19         ok(result == 'Success', "Insert Success");
20     }
21     }).fail(function(x, text, thrown) {
22         ok(false, "ajax failed: " + text);
23     }).always(function(){
24         start();
25     });
26 });

```

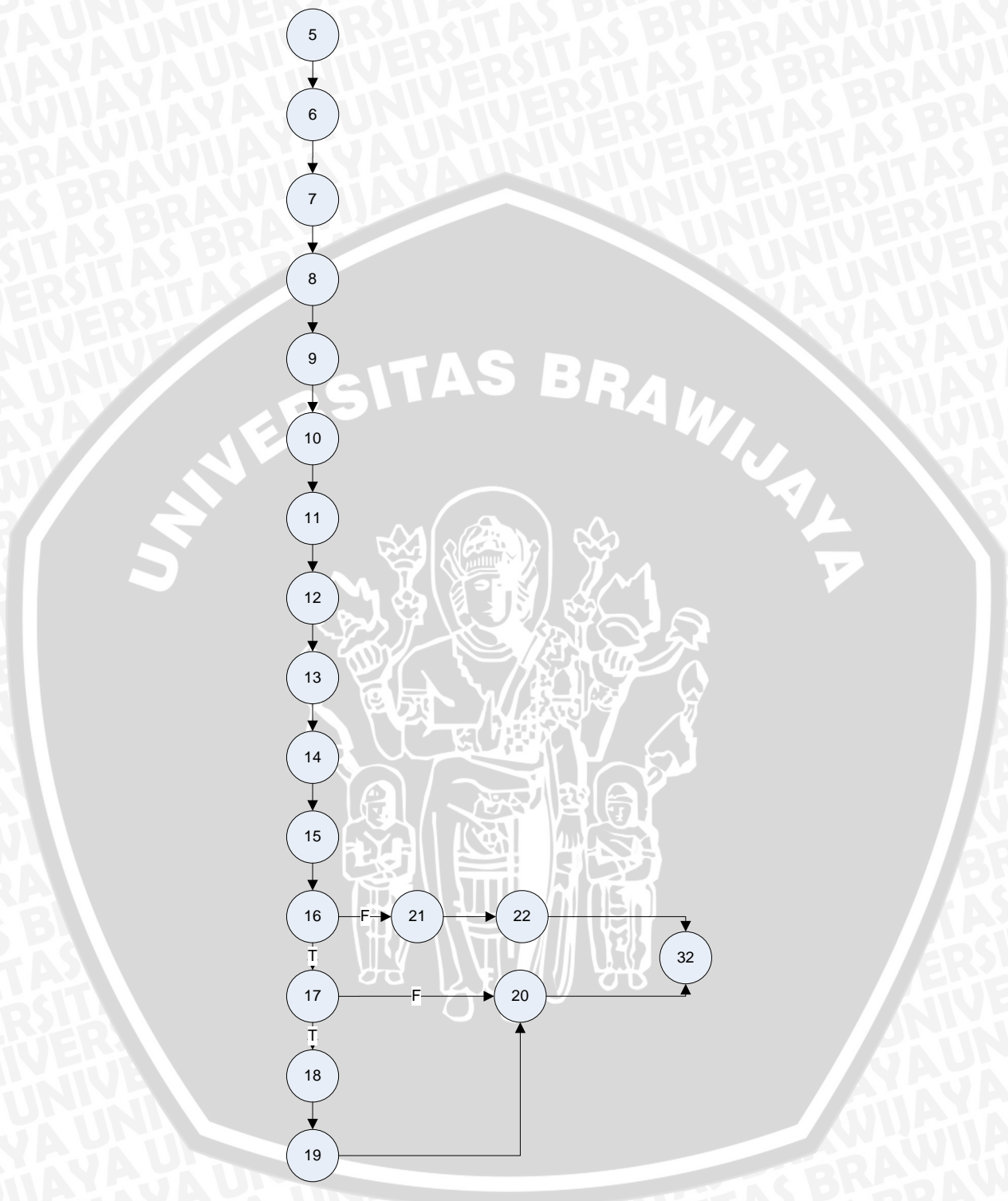
Pada Gambar 5.8 kasus uji unit fungsi *ajax create request* menjelaskan algoritma dalam bentuk kode yang akan dilakukan *testing* pada pengujian unit.

1. Pada baris 2 – 8 menjelaskan bahwa test yang akan dilakukan adalah test asinkronus menggunakan ajax dengan data kembalian bertipe JSON dan tipe ajax berupa POST.
2. Pada baris 9 – 14 adalah parameter data yang akan dikirim.
3. Pada baris 16 – 23 adalah hasil kembalian dari test tersebut ketika hasil diketahui sama antara data kembalian dengan syarat kembalian maka hasil yang diharapkan tercapai.

Langkah pengujian yang pertama adalah menggambarkan diagram alir dari kode program diatas yang ditampilkan pada Gambar 5.9.

Dari diagram alir pada Gambar 5.9 maka akan didapatkan jumlah dari *cyclomatic complexity* yaitu:

Perhitungan diatas menjelaskan bahwa dibutuhkan 3 skenario *test case* untuk menguji kode program fungsi *ajax create request*



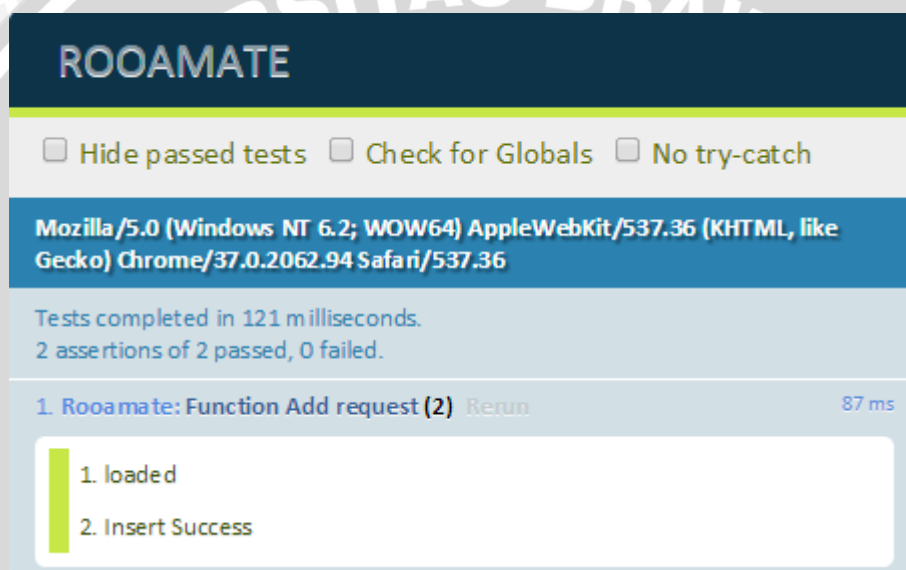
Gambar 5.9 Diagram Alir Ajax Create Request



No	Nama Skenario	Field Post id	Field Checkin	Field Checkout	Field User id
1	Data masuk	valid	valid	valid	valid

Skenario pengujian yang pertama dijalankan dengan mengisi *field* checkin, dan checkout secara benar dan diharapkan data yang valid akan masuk ke database.

Hasil pengujian unit pada kasus uji pertama dijelaskan dalam Gambar 5.10 di bawah ini.



Gambar 5.10 Hasil Uji Unit Skenario 1 *Ajax Create Request*

Dalam Gambar 5.10 di atas menjelaskan hasil pengujian terhadap kasus uji yang dilakukan pada Gambar 5.8 dengan skenario yang telah disebutkan. Pengujian pada kasus *test case* pertama di dapatkan hasil alokasi waktu test sebesar 87 ms dengan 2 dari 2 pernyataan dinyatakan diterima (*passed*). Hasil dari test tersebut dapat diketahui melalui data yang masuk kedalam database yang ditunjukkan dalam Gambar 5.11 dibawah ini.

	request_id	post_id	check_in	check_out	status	request_date	reply	user_id
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	2014090001	2014050003	2014-06-07	2014-06-09	2	2014-09-07		2014060002

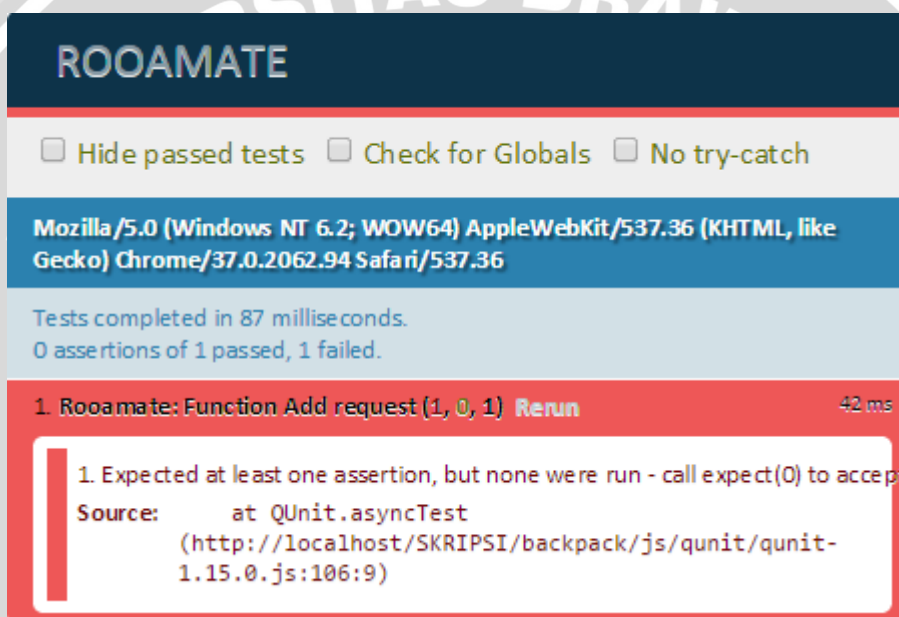
Gambar 5.11 Hasil Uji Unit Pada Database Skenario 1 *Ajax Create Request*



No	Nama Skenario	Field Post id	Field Checkin	Field Checkout	Field User id
2	Data tidak masuk	valid	invalid	invalid	valid

Skenario pengujian yang kedua dijalankan dengan tidak mengisi *field* checkin, dan checkout atau tidak mengisi salah satu secara benar dan diharapkan data tidak masuk ke dalam database.

Hasil pengujian unit pada kasus uji kedua dijelaskan dalam Gambar 5.12 di bawah ini.



Gambar 5.12 Hasil Uji Unit Skenario 2 *Ajax Create request*

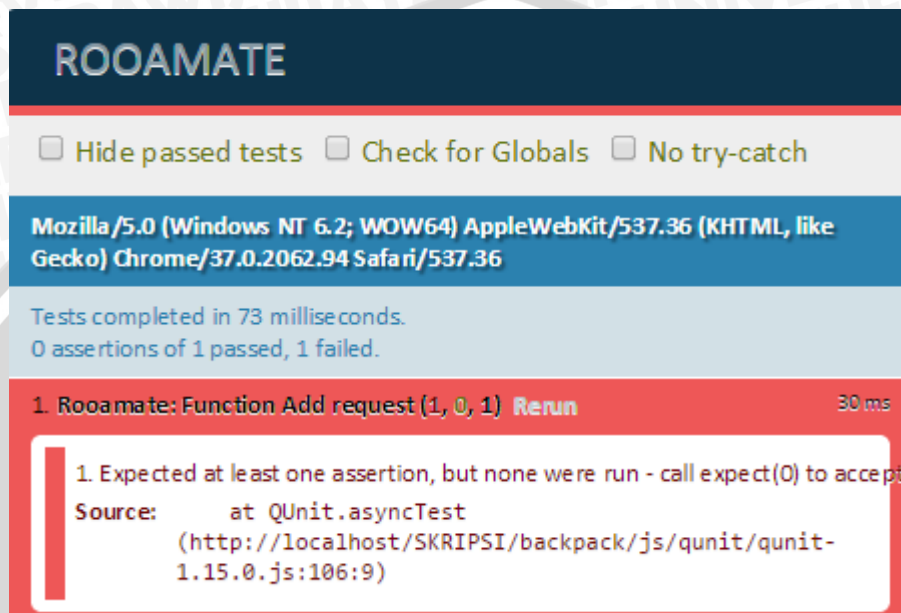
Dalam Gambar 5.12 di atas menjelaskan hasil pengujian terhadap kasus uji yang dilakukan pada Gambar 5.8 dengan skenario yang telah disebutkan. Pengujian pada kasus *test case* kedua di dapatkan hasil alokasi waktu test sebesar 42 ms dengan 1 dari 1 pernyataan dinyatakan gagal (*failed*).

No	Nama Skenario	Field Post id	Field Checkin	Field Checkout	Field User id
3	Data tidak masuk	invalid	invalid	invalid	invalid



Skenario pengujian yang ketiga dijalankan dengan tidak mengisi semua *field* secara benar dan diharapkan data tidak masuk ke dalam database.

Hasil pengujian unit pada kasus uji ketiga dijelaskan dalam Gambar 5.13 di bawah ini.



Gambar 5.13 Hasil Uji Unit Skenario 3 *Ajax Create request*

Dalam Gambar 5.13 di atas menjelaskan hasil pengujian terhadap kasus uji yang dilakukan pada Gambar 5.8 dengan skenario yang telah disebutkan. Pengujian pada kasus *test case* ketiga di dapatkan hasil alokasi waktu test sebesar 30 ms dengan 1 dari 1 pernyataan dinyatakan gagal (*failed*).

Kasus uji ketiga adalah menguji fungsi *ajax Update request* pada sistem. Pengujian dilakukan dengan memberikan data sebagai parameter yang selanjutnya akan diolah oleh server dengan pengembalian data berupa keterangan sukses atau gagal. Hasil yang diharapkan dari pengujian ini adalah berjalannya fungsi *ajax* yang berjalan secara asinkronus dan terjadi penambahan data pada database dan kembalian *ajax* yaitu notifikasi “*Success*”.

Kasus uji unit fungsi *ajax edit request* terdapat pada Gambar 5.14 berikut.

Gambar 5.14 Kasus Uji Unit *Ajax Update Request*

```

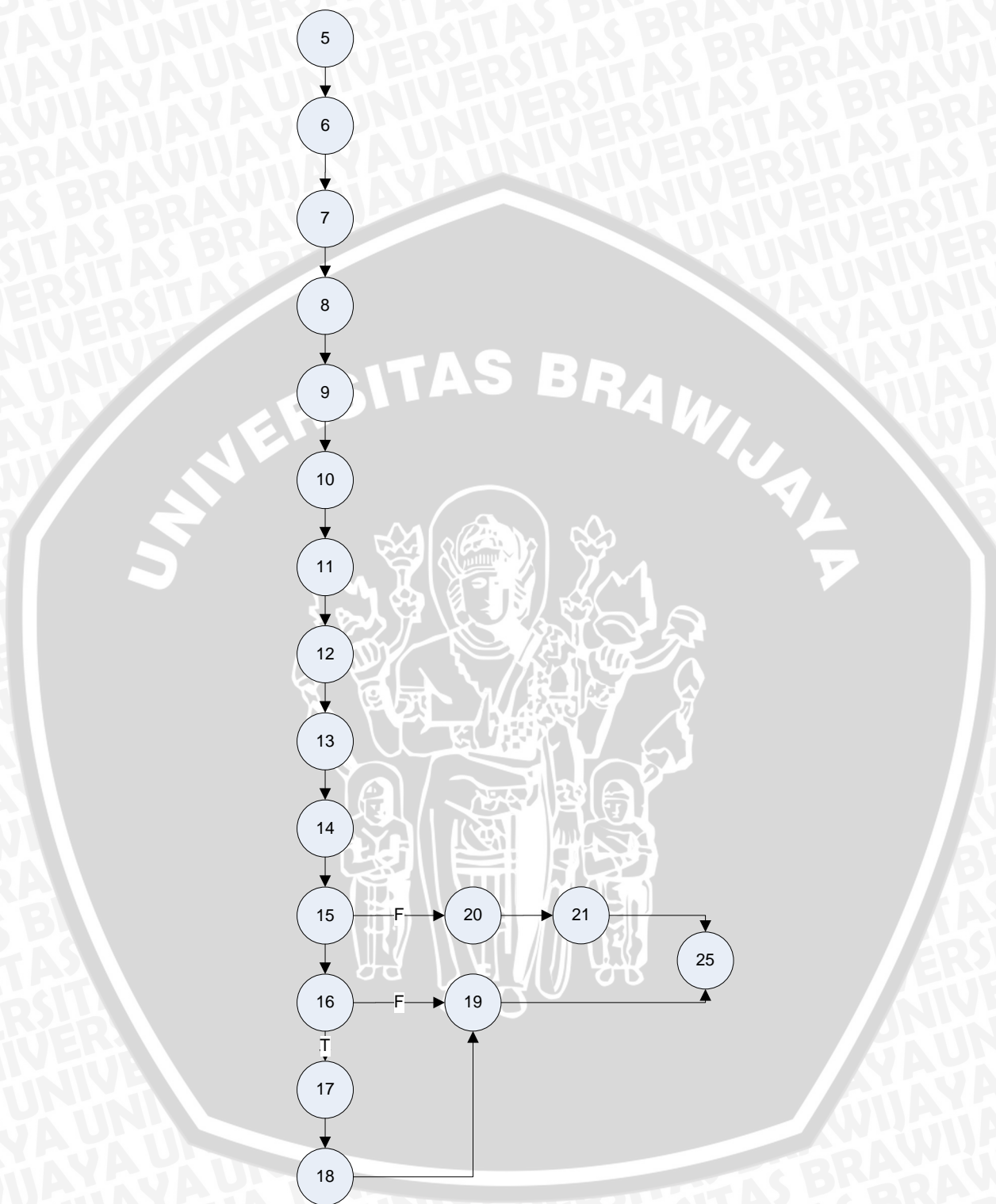
1 module("Rooamate");
2 asyncTest("Function Update request", function() {
3     var urlRequest = ("http://
4     www.rooamate.com/backpack/server/");
5     $.ajax({
6         url: urlRequest+'request.php',
7         dataType: "json",
8         type: "POST",
9         data: {
10             hidReqId : "0eef9b",
11             status : "1",
12             reply : "Approved"
13         },
14         timeout: 1000
15     }).done(function(result) {
16         if(result=='Success'){
17             ok(true, "loaded");
18             ok(result == 'Success', "ubah data Success");
19         }
20     }).fail(function(x, text, thrown) {
21         ok(false, "ajax failed: " + text);
22     }).always(function(){
23         start();
24     });
25 });

```

Pada Gambar 5.14 kasus uji unit fungsi *ajax update request* menjelaskan algoritma dalam bentuk kode yang akan dilakukan *testing* pada pengujian unit.

1. Pada baris 2 – 7 menjelaskan bahwa test yang akan dilakukan adalah test asinkronus menggunakan ajax dengan data kembalian bertipe JSON dan tipe ajax berupa POST.
2. Pada baris 8 – 12 adalah parameter data yang akan dikirim.
3. Pada baris 14 – 25 adalah hasil kembalian dari test tersebut ketika hasil diketahui sama antara data kembalian dengan syarat kembalian maka hasil yang diharapkan tercapai.

Langkah pengujian yang pertama adalah menggambarkan diagram alir dari kode program diatas yang ditampilkan pada Gambar 5.15 di bawah ini.



Gambar 5.15 Diagram Alir Ajax Update Request



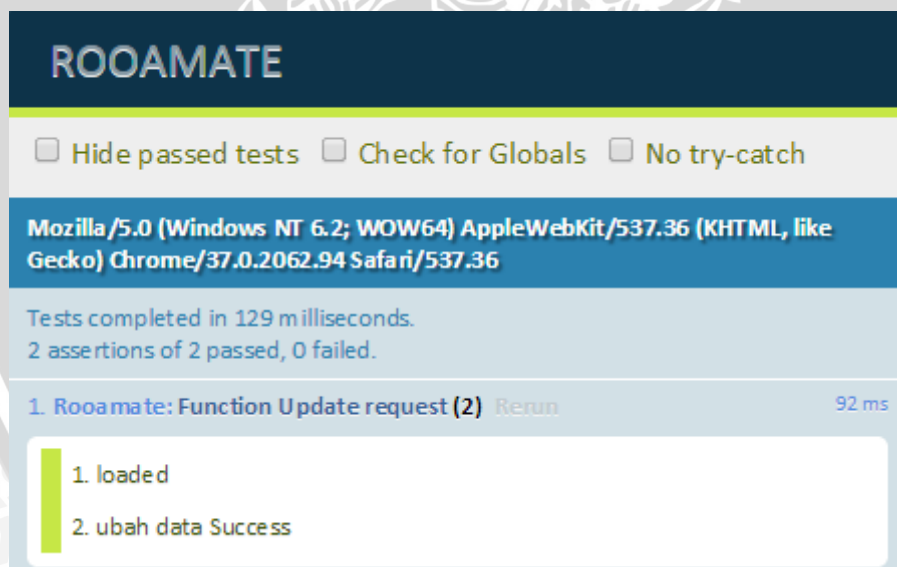
Dari diagram alir pada Gambar 5.15 diatas maka akan didapatkan jumlah dari *cyclomatic complexity* yaitu:

Perhitungan diatas menjelaskan bahwa dibutuhkan 3 skenario *test case* untuk menguji kode program fungsi *ajax update data*.

No	Nama Skenario	Field Request id	Field Status	Field Reply
1	Data berubah	valid	valid	valid

Skenario pengujian yang pertama dijalankan dengan mengisi *field status*, dan reply secara benar dan diharapkan data yang valid akan masuk ke database.

Hasil pengujian unit pada kasus uji pertama dijelaskan dalam Gambar 5.16 di bawah ini.



Gambar 5.16 Hasil Uji Unit Skenario 1 *Ajax Update Request*

Dalam Gambar 5.16 di atas menjelaskan hasil pengujian terhadap kasus uji yang dilakukan pada Gambar 5.14 dengan skenario yang telah disebutkan. Pengujian pada kasus *test case* pertama di dapatkan hasil alokasi waktu test sebesar 92 ms dengan 2 dari 2 pernyataan dinyatakan diterima (*passed*). Hasil dari

test tersebut dapat diketahui melalui data yang berubah pada database yang ditunjukkan dalam Gambar 5.17 dibawah ini.

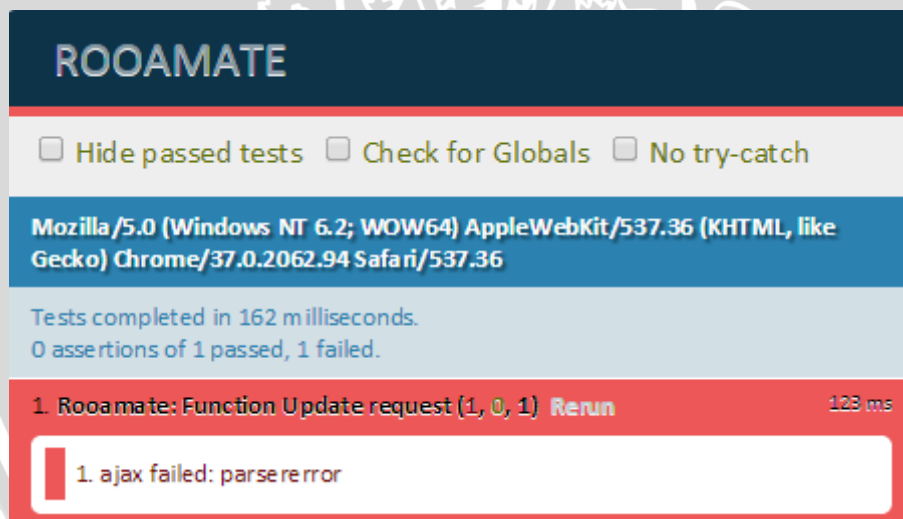
request_id	post_id	check_in	check_out	status	request_date	reply	user_id
2014090001	2014050003	2014-06-07	2014-06-09	1	2014-09-07	Approved	2014060002

Gambar 5.17 Hasil Uji Unit Pada Database Skenario 1 *Ajax Update Request*

No	Nama Skenario	Field Request id	Field Status	Field Reply
2	Data tidak berubah	valid	invalid	invalid

Skenario pengujian yang kedua dijalankan dengan tidak mengisi *field* status, dan *reply* secara benar dan diharapkan data tidak masuk ke database.

Hasil pengujian unit pada kasus uji kedua dijelaskan dalam Gambar 5.18 di bawah ini.



Gambar 5.18 Hasil Uji Unit Skenario 2 *Ajax Update request*

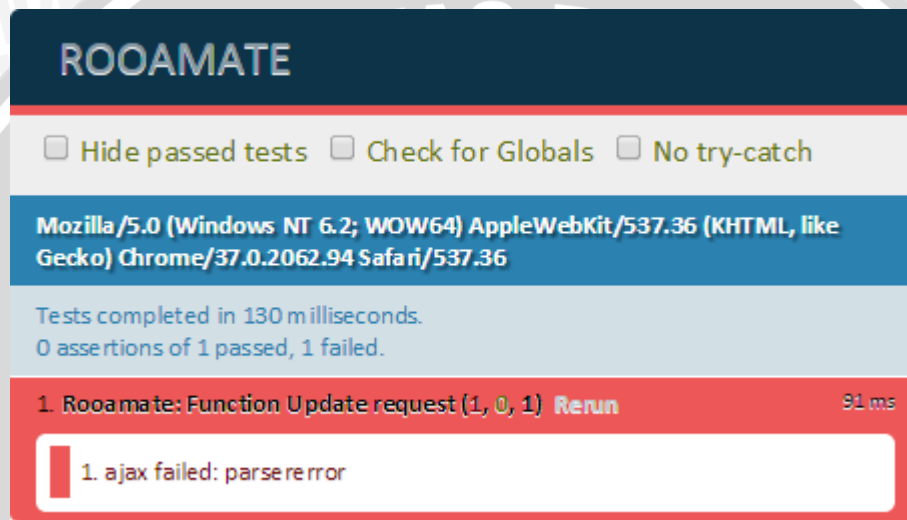
Dalam Gambar 5.18 di atas menjelaskan hasil pengujian terhadap kasus uji yang dilakukan pada Gambar 5.14 dengan skenario yang telah disebutkan. Pengujian pada kasus *test case* kedua di dapatkan hasil alokasi waktu test sebesar 123 ms dengan 1 dari 1 pernyataan dinyatakan gagal (*failed*).



No	Nama Skenario	Field Request id	Field Status	Field Reply
3	Data tidak berubah	invalid	invalid	invalid

Skenario pengujian yang ketiga dijalankan dengan tidak mengisi *field* status, *request id* dan *reply* secara benar dan diharapkan data tidak masuk ke database.

Hasil pengujian unit pada kasus uji ketiga dijelaskan dalam Gambar 5.19 di bawah ini.



Gambar 5.19 Hasil Uji Unit Skenario 3 *Ajax Update request*

Dalam Gambar 5.19 di atas menjelaskan hasil pengujian terhadap kasus uji yang dilakukan pada Gambar 5.14 dengan skenario yang telah disebutkan. Pengujian pada kasus *test case* ketiga di dapatkan hasil alokasi waktu test sebesar 91 ms dengan 1 dari 1 pernyataan dinyatakan gagal (*failed*).

5.2 Analisis Hasil Pengujian

Proses analisis terhadap hasil pengujian dilakukan untuk mendapatkan kesimpulan dari hasil pengujian aplikasi *mobile rooamate* yang telah dilakukan. Proses analisis mengacu pada hasil pengujian yang didapatkan. Analisis dilakukan terhadap hasil pengujian di setiap tahap pengujian. Proses analisis yang dilakukan meliputi analisis hasil pengujian validasi, dan analisis hasil pengujian komparabilitas.

5.2.1 Analisis Hasil Pengujian Validasi

Proses analisis terhadap hasil pengujian validasi dilakukan dengan melihat kesesuaian antara hasil kinerja sistem dengan daftar kebutuhan. Terdapat empat belas (14) kasus uji validasi yang dilakukan terhadap perangkat *mobile* Android dan iOS. Berdasarkan hasil pengujian validasi dengan kasus uji kelola profil pribadi, kelola penawaran tempat tinggal, pencarian tempat tinggal, kelola komentar, dan kelola request, dapat disimpulkan bahwa implementasi dan fungsionalitas aplikasi *mobile rooamate* telah memenuhi kebutuhan yang telah dijabarkan pada tahap analisis kebutuhan yang diantaranya adalah kebutuhan pengguna yang dapat menyediakan dan mencari informasi tempat menginap sementara.

5.2.2 Analisis Hasil Pengujian Komparabilitas

Proses analisis terhadap hasil pengujian komparabilitas yang dilakukan pada aplikasi *mobile* terhadap sistem operasi Android dan iOS dilakukan untuk melihat kesesuaian antara hasil kinerja sistem dengan daftar kebutuhan. Berdasarkan hasil pengujian komparabilitas dapat disimpulkan bahwa implementasi dan fungsionalitas aplikasi *mobile rooamate* telah memenuhi kebutuhan yang telah dijabarkan pada tahap analisis kebutuhan.

5.2.3 Analisis Hasil Pengujian UAT (*User Acceptance Test*)

Proses analisis terhadap hasil pengujian UAT dilakukan dengan menggunakan skala Likert. Pengujian UAT layak diterima jika perhitungan keseluruhan jawaban dari setiap pernyataan lebih dari 50%. Interpretasi skor Likert atau persentase dari setiap skor Likert ditunjukkan pada Tabel 5.17. Hasil

perhitungan persentase keseluruhan jawaban dari setiap pernyataan ditunjukkan pada Tabel 5.18 dan hasil status pengujian UAT ditunjukkan pada Tabel 5.19.

Tabel 5.17 Interpretasi Skor Likert

Skor Likert	Interpretasi skor dengan interval = 20	Pilihan
1	0% - 19.99%	Sangat Tidak Setuju
2	20% - 39.99%	Tidak Setuju
3	40% - 59.99%	Netral
4	60% - 79.99%	Setuju
5	80% - 100%	Sangat Setuju

Keterangan:

Interval = 20 didapatkan dari pembagian nilai 100 dengan jumlah skor Likert

Tabel 5.18 Persentase Keseluruhan Jawaban

No	Pernyataan	STS	TS	N	S	SS	Total Skor	PKJ (%)
1	Saya merasa navigasi di dalam aplikasi mudah bagi saya	0	0	3	14	3	80	80%
2	Saya merasa interaksi di dalam aplikasi jelas dan mudah di mengerti	0	1	4	11	4	78	78%
3	Saya merasa sangat mudah menggunakan aplikasi ini	0	0	5	9	6	81	81%
4	Saya merasa aplikasi ini dapat merespons informasi dengan cepat	0	0	1	11	8	87	87%
5	Saya merasa aplikasi ini	0	0	3	11	6	83	83%

	memungkinkan saya untuk mendapatkan informasi yang ingin saya dapatkan							
6	Saya merasa aplikasi ini dapat membantu saya untuk mengumpulkan informasi lebih banyak	0	2	2	11	5	79	79%
7	Saya merasa aplikasi ini memberikan respon kinerja yang cepat ketika saya operasikan	0	2	1	13	4	79	79%
8	Saya dapat terbantu dengan adanya sistem ini untuk mengetahui informasi lokasi dan rute menuju lokasi tempat menginap.	0	0	0	8	12	92	92%

Keterangan:

STS : Sangat Tidak Setuju **S** : Setuju

TS : Tidak Setuju **SS** : Sangat Setuju

N : Netral

$$\text{Total Skor} = S_{STS} \times 1 + S_{TS} \times 2 + S_N \times 3 + S_S \times 4 + S_{SS} \times 5$$

$$\text{Persentase Keseluruhan Jawaban (\%)} = (\text{Total Skor} / Y) \times 100$$

Y : Skor Likert tertinggi x Jumlah audien

Berdasarkan Tabel 5.19 pada pernyataan 1 didapatkan hasil presentase keseluruhan jawaban sebesar 80%. Karena kebanyakan koresponden dapat dengan mudah mengerti navigasi pada aplikasi. Seperti navigasi pada halaman utama dan menu sidebar, navigasi telah diimplementasikan menggunakan *icon* dan tulisan yang jelas. Pernyataan 2 didapatkan hasil presentase keseluruhan jawaban sebesar 78%. Karena interaksi yang ada pada aplikasi dibuat sederhana seperti interaksi

antara halaman satu dengan yang lain tidak terlalu berbeda. Sehingga pengguna dapat langsung mengerti interaksi yang ada pada aplikasi tersebut. Pernyataan 3 didapatkan hasil presentase keseluruhan jawaban sebesar 81%. Karena navigasi dan interaksi pada aplikasi mudah untuk digunakan oleh pengguna. Sehingga aplikasi juga dapat digunakan dengan mudah oleh pengguna. Pernyataan 4 didapatkan hasil presentase keseluruhan jawaban sebesar 87%. Karena koneksi internet yang digunakan saat menjalankan aplikasi menggunakan koneksi 3G, HSDPA, dan *wifi* sehingga aplikasi dapat merespon fungsi-fungsinya dengan cepat. Pernyataan 5 didapatkan hasil presentase keseluruhan jawaban sebesar 83%. Karena informasi yang ditampilkan pada aplikasi terdiri dari informasi tentang deskripsi lokasi, harga, dan informasi rute menuju lokasi secara detail. Informasi yang ditampilkan singkat tetapi jelas. Sehingga pengguna dapat memahami informasi tersebut. Pernyataan 6 didapatkan hasil presentase keseluruhan jawaban sebesar 79%. Karena aplikasi menyediakan banyak informasi tentang penawaran tempat menginap pada pengguna. Pernyataan 7 didapatkan hasil presentase keseluruhan jawaban sebesar 79%. Karena aplikasi tidak menggunakan banyak gambar yang berukuran besar dan aplikasi dibuat sederhana, selain itu internet yang digunakan cukup stabil, sehingga banyak pengguna yang setuju dengan respon yang cepat pada aplikasi tersebut. Pernyataan 8 didapatkan hasil presentase keseluruhan jawaban sebesar 92%. Karena pada aplikasi pencarian lokasi ditampilkan secara detail mulai dari informasi hingga rute pada peta, sehingga pengguna dapat dengan mudah mengetahui arah untuk menuju lokasi penawaran tempat menginap.

Tabel 5.19 Status Pengujian UAT

Aspek Penilaian	Persentase (%)	Status
Navigasi mudah	80%	Memuaskan
Interaksi di dalam aplikasi jelas dan mudah di mengerti	78%	Memuaskan
Mudah menggunakan aplikasi	81%	Memuaskan

Kecepatan respons terhadap fungsi-fungsinya	87%	Memuaskan
Informasi jelas dan mudah dipahami	83%	Memuaskan
Informasi yang didapatkan banyak	79%	Memuaskan
Memberikan respon kinerja yang cepat	79%	Memuaskan
Dapat membantu dalam mendapatkan informasi rute	92%	Memuaskan

Perhitungan persentase rata-rata dari semua aspek penilaian

= jumlah total presentase (%) / jumlah pernyataan

= $659 / 8 = 82.375\% = 82\%$

Kesimpulan yang dapat diambil dari hasil perhitungan persentase keseluruhan jawaban pada setiap pernyataan dan status pengujian UAT ialah aplikasi layak diterima atau memuaskan dengan persentase rata-rata dari semua aspek penilaian sebesar 82%.

5.2.4 Analisis Hasil Pengujian Unit

Proses analisis terhadap hasil pengujian unit dilakukan dengan melihat kesesuaian fungsi dari implementasi unit fungsi yang diuji dengan hasil perancangan perangkat lunak yang telah dirancang sebelumnya. Terdapat 3 fungsi utama dari aplikasi yang diambil untuk diuji menggunakan tools Qunit, yaitu fungsi *ajax load data*, fungsi *ajax create request* dan fungsi *ajax edit request*. Tahap pengujian yang pertama adalah menggambar diagram alir program untuk mendapatkan *cyclomatic complexity* yang akan menghasilkan jumlah kasus uji yang harus dilakukan pada satu fungsi. Pada fungsi *ajax load data* didapatkan 4 buah kasus uji yang harus dilakukan dimana dari empat pengujian tersebut 1 kasus uji dinyatakan berhasil sedangkan 3 dinyatakan gagal. Pada fungsi *ajax create request* didapatkan 3 buah kasus uji yang harus dilakukan dimana dari empat pengujian tersebut 1 kasus uji dinyatakan berhasil sedangkan 2 dinyatakan gagal.

Pada fungsi *ajax edit request* didapatkan 3 buah kasus uji yang harus dilakukan dimana dari empat pengujian tersebut 1 kasus uji dinyatakan berhasil sedangkan 2 dinyatakan gagal. Kasus uji yang dinyatakan berhasil dikarenakan pengkondisian input data yang benar sehingga sistem dapat memproses data tersebut dan mengembalikan hasil yang sesuai, sedangkan kasus uji yang gagal di karenakan pengkondisian input data yang salah sehingga sistem mengembalikan data yang tidak sesuai.

Berdasarkan hal tersebut, maka dapat diambil kesimpulan bahwa unit fungsi yang diuji dari program sudah memenuhi kebutuhan fungsional yang telah dirancang pada tahap perancangan.



BAB VI

PENUTUP

6.1 Kesimpulan

Berdasarkan hasil analisa, perancangan, implementasi dan pengujian yang dilakukan, maka diambil kesimpulan sebagai berikut:

1. Perancangan aplikasi *mobile rooamate* telah dibuat sesuai dengan spesifikasi kebutuhan dan perancangan yang telah dianalisa dan dapat digunakan sebagai salah satu media yang dapat menyediakan dan mencari informasi tempat menginap sementara.
2. Berdasarkan dari hasil pengujian *User Acceptance Testing*, sistem telah dapat diterima oleh pengguna. Kebutuhan non fungsionalitas terhadap kemudahan kegunaan dan pencapaian kegunaan aplikasi dapat terpenuhi dengan persentase rata-rata dari semua aspek penilaian sebesar 82% yang menunjukkan bahwa aplikasi dapat diterima atau dapat digunakan dengan mudah oleh pengguna.
3. Berdasarkan dari hasil pengujian validasi, kompatibilitas dan uji unit, sistem dapat dinyatakan berjalan dengan baik pada perangkat *mobile* Android dan iOS, dan memenuhi spesifikasi kebutuhan yang telah dianalisa tanpa perlu melakukan perubahan pada kode program.

6.2 Saran

Saran yang dapat diberikan untuk pengembangan aplikasi *rooamate* selanjutnya antara lain adalah:

1. Untuk pembangunan lebih lanjut, aplikasi ini dapat dikembangkan dengan menambahkan fitur *push notification* sebagai *reminder*.
2. Untuk pembangunan lebih lanjut dapat ditambahkan fitur kompas pada peta sehingga aplikasi dapat menunjukkan menghadap kemanakah arah pengguna saat menggunakan fitur *route* agar pengguna dapat lebih mudah menentukan arah saat mencari lokasi tempat menginap.
3. Dilakukan pengembangan untuk beberapa *mobile os* lain yaitu windows phone dan Symbian agar dapat menjangkau pengguna yang menggunakan perangkat selain Android dan iOS.



DAFTAR PUSTAKA

- [ASN-13] Sørensen, Anders. 2003, "BACKPACKER ETHNOGRAPHY", Elsevier Ltd, Great Britain.
- [CCO-06] O'Reilly, Camille Caprioglio. 2006, "FROM DRIFTER TO GAP YEAR TOURIST Mainstreaming Backpacker Travel", Elsevier Ltd, Great Britain.
- [DMZ-06] Maoz, Darya. 2006, "BACKPACKERS' MOTIVATIONS The Role of Culture and Nationality", Elsevier Ltd, Great Britain.
- [BYA-11] Bassil, Youssef. 2011, "A Simulation Model for the Waterfall Software Development Life Cycle", Lebanese Association for Computational Sciences, Lebanon.
- [ABH-06] Hanan, Abdul. 2006. Belajar PHP. Akses dari <http://www.belajar-gratis.com> Tanggal 05 April 2014.
- [SVT-01] NATIVE, HTML5, HYBRID. WHAT TO CHOOSE?. Akses dari <http://svitla.com/blog/native-html5-hybrid-what-to-choose/> tanggal 25 Maret 2014.
- [SNW-10] Wahyuni, Sri Ngudi. 2010. Wireless Application Protocol. Yogyakarta : STMIK AMIKOM.
- [UNK-11] Unikom. 2011. Modul Praktikum Pengantar Komputer dan Internet, HTML (Hyper Text Markup Language) part 1. Akses dari <http://elib.unikom.ac.id/files/disk1/468/jbptunikompp-gdl-andisugand-23388-1-html1.pdf> Tanggal 24 Juli 2014.
- [DML-04] Pengenalan HTML. Akses dari <https://developer.mozilla.org/id/docs/Web/Guide/HTML/Pengenalan> tanggal 30-04-2014
- [W3C-01] HTML Introduction. Akses dari http://www.w3schools.com/html/html_intro.asp Tanggal 30-04-2014

- [ASS-10] Siregar, Astuan Saputra. 2010. Sistem Informasi Pengolahan Data Pegawai dan Surat Berbasis *Web* Dinas Pendidikan Kecamatan Subang. Bandung : UNIKOM.
- [ASR-13] Syamsul, Asep. 2013. Sistem informasi simpan pinjam di Koperasi Pegawai Republik Indonesia-Koperasi Inspeksi Pendidikan Agama Islam Sumedang. Bandung : UNIKOM.
- [CSS-13] Wikipedia. 2013. Cascading Style Sheets (CSS). Akses dari http://en.wikipedia.org/wiki/Cascading_Style_Sheets Tanggal 21 April 2014.
- [W3C-13] Akses dari <http://www.w3schools.com/> Tanggal 04 Mei 2014
- [BLE-10] Panggabean, Benny. 2010. Pembangunan situs Penjualan Online Pada CC.Cipta Rupa Pratama Dengan Menggunakan Framework CakePHP Dan Framework jQUERY. Bandung : UNIKOM.
- [MUI-11] Iqbal, Muhammad. 2011. Pengembangan *Web* Portal Universitas Menggunakan JQuery Pada STMIK Islam Internasional Jakarta. Jakarta : UIN Syarif Hidayatullah.
- [JQM-14] JQuery Mobile. 2014. Akses dari http://en.wikipedia.org/wiki/JQuery_Mobile Tanggal 7 April 2014.
- [TRY-07] Yuliano, Triswansyah. 2007. Pengenalan PHP. Akses dari Ilmu Komputer.com Tanggal 15 Februari 2014.
- [DAS-07] Suhartono, Dwi Agung, dkk. 2007. Aplikasi Sistem Informasi Penjualan Pada Gerai (Outlet) Ponsel Berbasis *Web*. Semarang : Universitas Diponegoro.
- [ASR-13] Syamsul, Asep. 2013. Sistem informasi simpan pinjam di Koperasi Pegawai Republik Indonesia-Koperasi Inspeksi Pendidikan Agama Islam Sumedang. Bandung : UNIKOM.
- [PTX-13] Pengertian tentang XAMPP. 2013. Akses dari

<http://blog.duniascript.com/pengertian-tentang-xampp.html> Tanggal 30 Januari 2014.

[XAM-13] XAMPP 1.7.7. 2013. Akses dari <http://merahmarunluck.blogspot.com/2011/10/xampp-for-windows-177.html> Tanggal 22 April 2014.

[PHG-14] PhoneGap. 2014. Akses dari <http://en.wikipedia.org/wiki/PhoneGap> Tanggal 7 April 2014.

[ARO-08] Rodriguez, Alex. 2008. RESTful Web services: The basics. IBM.

