RANCANG BANGUN APLIKASI INFORMASI GEDUNGUNIVERSITAS BRAWIJAYA PADA *SMARTPHONE* ANDROID

SKRIPSI

Untuk memenuhi sebagian persyaratan untuk mencapai gelar Sarjana Komputer

Disusun Oleh :

FARID ANGGA PRIBADI

NIM. 0710683039

KEMENTERIAN PENDIDIKAN DAN KEBUDAYAAN

UNIVERSITAS BRAWIJAYA

PROGRAM TEKNOLOGI INFORMASI DAN ILMU KOMPUTER

PROGRAM STUDI TEKNIK INFORMATIKA

MALANG

2013

LEMBAR PERSETUJUAN
RANCANG BANGUN APLIKASI INFORMASI GEDUNGUNIVERSITAS BRAWIJAYA PADA *SMARTPHONE* ANDROID

SKRIPSI

Untuk memenuhi sebagian persyaratan untuk mencapai gelar Sarjana Komputer

Disusun Oleh :

FARID ANGGA PRIBADI

NIM.0710683039

Telah diperiksa dan disetujui oleh :

Dosen Pembimbing I	Dosen Pembimbing II
Ismiarta Aknuranda, S.T, M.Sc, Ph.D	**Wibisono Sukmo Wardhono, ST., MT.**
NIK. 740719 06 1 1 0079	**NIK. 820404 06 1 1 0091**

LEMBAR PENGESAHAN

RANCANG BANGUN APLIKASI INFORMASI GEDUNGUNIVERSITAS BRAWIJAYA PADA *SMARTPHONE* ANDROID

SKRIPSI

KONSENTRASI REKAYASA PERANGKAT LUNAK

Diajukan untuk memenuhi
persyaratan memperoleh gelar Sarjana Komputer

Disusun Oleh :

FARID ANGGA PRIBADI

NIM.0710683039

Skripsi ini telah diuji dan dinyatakan lulus pada
tanggal 11Januari 2013

Penguji I	Penguji II
Issa Arwani, S.Kom., M.Sc.	**Eriq M. Adams J., ST., M.Kom.**
NIK. 830922 06 1 1 0074	**NIK. 850410 06 1 1 0027**

Penguji III

Ir. Heru Nurwasito, M.Kom.

NIP. 19650402 199002 1 001

Mengetahui
Ketua Program Studi Teknik Informatika

Drs. Marji, MT.
NIP. 19670801 199203 1 001

PERNYATAAN
ORISINALITAS SKRIPSI

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

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

Malang, 23Januari 2013
Mahasiswa,

Farid Angga Pribadi
NIM 0710683039

KATA PENGANTAR

Dengan nama Allah SWT Yang Maha Pengasih dan Penyayang. Segala puji bagi Allah SWT karena atas rahmat dan hidayah-Nyalah penulis dapat menyelesaikan Skripsi yang berjudul "Rancang Bangun Aplikasi Informasi Gedung Universitas Brawijaya Pada *Smartphone* Android". Shalawat dan salam atas junjungan besar kita Nabi Muhammad S.A.W. beserta keluarga dan para sahabat sekalian. Skripsi ini disusun untuk memenuhi sebagian persyaratan memperoleh gelar Sarjana Komputer di Program Studi Teknik Informatika Program Teknologi Informasi dan Ilmu Komputer Universitas Brawijaya Malang.

Melalui kesempatan ini, penulis ingin menyampaikan rasa hormat dan terima kasih penulis yang sebesar-besarnya kepada semua pihak yang telah memberikan bantuan – bantuan baik lahir maupun batin selama penulisan tugas akhir ini. Oleh karena itu, pada kesempatan ini penulis ingin menyampaikan rasa hormat dan terima kasih penulis kepada :

1. Ayahanda Imam Suhadi, Ibunda Erna Fidiah S.Pd., Kakanda Dina Ika Pravitasari S.Si., dan seluruh keluarga besar atas segala nasehat, kasih sayang, perhatian dan kesabarannya di dalam membesarkan dan mendidik penulis, serta yang senantiasa tiada henti – hentinya memberikan doa dan semangat demi terselesaikannya skripsi ini.
2. Bapak Ir. Sutrisno, M.T, Bapak Ir. Heru Nurwasito, M.Kom, Bapak Himawat Aryadita, S.T, M.Sc, dan Bapak Eddy Santoso, S.Kom selaku Ketua, Wakil Ketua 1, Wakil Ketua 2 dan Wakil Ketua 3 Program Teknologi Informasi dan Ilmu Komputer Universitas Brawijaya.
3. Bapak Drs. Marji, M.T dan Bapak Issa Arwani, S.Kom, M.Sc selaku Ketua dan Sekretaris Program Studi Teknik Informatika Universitas Brawijaya.
4. Bapak Ismiarta Aknuranda, ST, M.Sc, Ph.D dan Bapak Wibisono Sukmo Wardhono, ST., MT. selaku dosen pembimbing skripsi yang telah dengan sabar membimbing dan mengarahkan penulis sehingga dapat menyelesaikan skripsi ini.
5. Bapak Suprapto, S.T, M.T selaku dosen penasehat akademik yang selalu memberikan nasehat kepada penulis selama menempuh masa studi.

6. Bapak Sabriansyah Rizqika Akbar, ST., M.Eng, M.T dan Bapak Eriq Muhammad Adams, ST., M.Kom yang telah banyak membantu dan memberi penulis inspirasi dalam penulisan skripsi ini.
7. Seluruh Dosen Teknik Informatika Universitas Brawijaya atas kesediaan membagi ilmunya kepada penulis.
8. Seluruh Civitas Akademika Teknik Informatika Universitas Brawijaya yang telah banyak memberi bantuan dan dukungan selama penulis menempuh studi di Teknik Informatika Universitas Brawijaya dan selama penyelesaian skripsi ini.
9. Adam Hendra Brata S.Kom, Fedrick Zulvasius S.Kom, Lutfi Fanani S.Kom, Firman Jati Pamungkas, Sativandi Putra, Deta Pratama, Anza Ansori, Komang Candra Brata dan semua sahabat – sahabat Angkatan 2007 yang selalu bertukar semangat dengan penulis selama menyelesaikan skripsi ini serta berbagi suka duka selama menempuh studi di Teknik Informatika Universitas Brawijaya.
10. Utami Wahyuning Kristi beserta Keluarga yang selalu memberikan semangat, dukungan dan doa.
11. Semua sahabatku di SMA atas doa, dukungan dan semangat yang selalu diberikan selama penulis menempuh studi di Teknik Informatika Universitas Brawijaya.
12. Semua pihak yang tidak dapat penulis sebutkan satu per satu yang terlibat baik secara langsung maupun yang tidak langsung demi terselesaikannya skripsi ini.

Hanya doa yang bisa penulis berikan semoga Allah SWT memberikan pahala serta balasan kebaikan yang berlipat. Penulis menyadari bahwa skripsi ini masih banyak kekurangan dan masih jauh dari sempurna. Untuk itu, saran dan kritik yang membangun sangat penulis harapkan. Semoga skripsi ini membawa manfaat bagi penyusun maupun pihak lain yang menggunakannya.

Malang, 23Januari 2013

Penulis

ABSTRAK

Farid Angga Pribadi. 2012. : Rancang Bangun Aplikasi Informasi Gedung Universitas Brawijaya Pada *Smartphone* Android. Skripsi Program Studi Teknik Informatika, Program Teknologi Informasi dan Ilmu Komputer, Universitas Brawijaya. Dosen Pembimbing : Ismiarta Aknuranda, ST, M.Sc, Ph.D dan Wibisono Sukmo Wardhono, ST., MT.

Informasi GedungUB merupakan aplikasi yang mengolah informasi yang sebelumnya belum digunakan secara maksimal menjadi informasi yang memiliki nilai lebih. Informasi utama yang diolah dalam aplikasi ini adalah letak gedung berupa koordinat. Aplikasi ini menunjukkan letak gedung-gedung yang ada di Universitas Brawijaya melalui *mobile device (smartphone* Android). Aplikasi ini memanfaatkan fungsi kamera untuk mendeteksi gedung-gedung dan menggunakan fungsi gps untuk mendeteksi koordinat dari gedung-gedung yang ada di Universitas Brawijaya. Informasi ditampilkan menggunakan teknologi *Augmented Reality (*AR*)*. Aplikasi ini diimplementasikan dengan pemrograman JAVA dan IDE Eclipse Helios dan dikembangkan dengan metode *Component Based Software Engineering* (CBSE). Metode CBSE bertujuan untuk menggabungkan komponen yang sudah ada dan dapat digunakan kembali. Aplikasi ini mengambil data dari *webserver* yang telah diuji pada jaringan intranet maupun internet. Pada saat pengujian unit, disimpulkan bahwa unit modul dari program sudah memenuhi kebutuhan fungsional yang telah dirancang pada tahap perancangan. Pada pengujian validasi dapat disimpulkan bahwa implementasi dan fungsionalitas perangkat lunak Informasi Gedung UBtelah memenuhi kebutuhan yang telah dijabarkan pada tahap analisis kebutuhan. Hasil pengujian performa menunjukkan bahwa pada performa server pada jaringan intranet memiliki performa lebih baik daripada performa server pada jaringan internet.

Kata Kunci : Informasi, *Augmented Reality* , Android, *GPS*, Koordinat, *Webserver*

ABSTRACT

Farid Angga Pribadi. 2012. : *Design Information Building Brawijaya University Application on* Android *Smartphone*. *Advisor* : Ismiarta Aknuranda, ST, M.Sc, Ph.D*and* Wibisono Sukmo Wardhono, ST., MT.

Informasi Gedung UB *is an application that processes the information that previously has not been used optimally to the information that has value. The main information that is processed in this application is the location of the building in the form of the coordinates. This application shows the location of the existing buildings at UB via mobile devices (Android smartphone). These applications take advantage of the camera function to detect the buildings and use the GPS function to detect the coordinates of the existing buildings at UB. Information displayed using the technology of Augmented Reality (AR).This application is implemented with JAVA programming and IDE Eclipse Helios and was developed by the method of Component Based Software Engineering (CBSE). CBSE method aims to combine existing components and reusable. These applications retrieve the data from the web server that has been tested on the intranet and internet.At the time of the test unit, it was concluded that the unit module of the program meets the functional requirements that have been designed at the design stage. In validation testing can be concluded that the implementation and functionality of the software* Informasi Gedung UB *has been meeting the needs that have been outlined in the requirements analysis phase.The performance test results show that the performance of the server on the intranet has performed better than the performance of a server on the internet.*

Keywords : *Information, Augmented Reality, Android, GPS, Coordinat, Webserver*

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

DAFTAR ISTILAH

Augmented Reality	Sebuah teknologi yang menggabungkan benda maya dua dimensi dan ataupun tiga dimensi ke dalam sebuah lingkungan nyata tiga dimensi lalu memproyeksikan benda -benda maya tersebut dalam waktu nyata.
CBSE	*Component-Based Software Engineering* (CBSE) adalah metode pengembangan perangkat lunak yang menekankan pada perancangan dan pembangunan perangkat lunak dengan menggunakan komponen perangkat lunak yang sudah ada dan bersifat dapat digunakan kembali.
GPS	Global Positioning System merupakan sistem navigasi satelit. Dalam sistem ini digunakan 24 satelit yang mengirimkan sinyal gelombang mikro ke Bumi . Sistem navigasi satelit mengirimkan data posisi (garis bujur dan lintang, dan ketinggian) dan sinyal waktu dari satelit, ke alat penerima di permukaan. Penerima di permukaan dapat mengetahui posisi, kecepatan, arah, dan waktu secara tepat.
Mixare	*Mix Augmented Reality Engine* adalah *open source augmented reality browser* yang diterbitkan dibawah GPLv3. Mixare tersedia untuk Android dan iPhone 3GS dan diatasnya. Mixare bekerja sebagai aplikasi yang sangat otonom dan tersedia untuk pengembangan dari implementasi sendiri.
MVC	*Model-View-Controller* atau MVC adalah sebuah metode untuk membuat sebuah aplikasi dengan memisahkan data (*Model*) dari tampilan (*View*) dan cara bagaimana memprosesnya (*Controller*).

UML	*Unified Modelling Language* (UML) adalah sebuah "bahasa" yg telah menjadi standar dalam industri untuk visualisasi, merancang dan mendokumentasikan sistem perangkat lunak.

BAB I
PENDAHULUAN

1.1 Latar Belakang

Informasi merupakan kebutuhan yang sangat penting pada masa sekarang ini. Banyak media yang digunakan dalam penyampaian informasi, mulai dari media cetak sampai media eletronik. Informasi-informasi yang ada sekarang telah diolah menggunakan sebuah teknologi yang biasa kita kenal dengan istilah teknologi informasi.

Teknologi informasi telah diterapkan di berbagai bidang kehidupan, mulai dari bidang kesehatan, pendidikan, pariwisata, bisnis, keamanan dan bidang-bidang lain. Informasi-informasi yang ada diolah dengan teknologi sehingga berguna dalam bidangnya masing-masing. Teknologi informasi berkembang secara terus menerus. Selalu ada hal baru yang muncul di setiap perkembangannya yang bersifat berkelanjutan.

Perkembangan teknologi informasi saat ini mengarah pada bagaimana menyajikan informasi sehingga pengakses informasi mendapatkan informasi secara mudah, cepat dan akurat dengan berbagai cara dan berbagai media, salah satunya adalah *Augmented Reality* (AR).

AR adalah teknologi yang menggabungkan benda maya dua dimensi dan ataupun tiga dimensi ke dalam sebuah lingkungan nyata lalu memproyeksikan benda-benda maya tersebut dalam waktu nyata. Benda-benda maya berfungsi menampilkan informasi yang tidak dapat diterima oleh manusia secara langsung. Contohnya,sebuah pola yang mengandung informasi tertentu yang hanya dapat dilihat oleh teknologi AR. Hal ini membuat AR berguna sebagai alat untuk membantu persepsi dan interaksi penggunanya dengan dunia nyata. Informasi yang ditampilkan oleh benda maya membantu pengguna melaksanakan kegiatan-kegiatan dalam dunia nyata.

Universitas Brawijaya (UB) Malang merupakan salah satu universitas negeri di kota Malang. UB memiliki luas area kampus 1.813.664 m²[UBM-12] dan tentunya memiliki banyak gedung. Setiap fakultas maupun jurusan memiliki gedung tersendiri dengan fungsi tersendiri pula. Seperti gedung kuliah, gedung birokrat, gedung kegiatan kemahasiswaan dan lain-lain. Untuk membantu

pengunjung UB mengenali gedung-gedung tersebut, dibutuhkan sebuah sistem informasi gedung yangmudah, cepat dan akurat.

Penulis mempunyai gagasan yang dapat memenuhi kebutuhan informasi yang disebutkan sebelumnya. Gagasan ini adalah merancang sebuah aplikasi*mobile* yang dapat mengolah informasi posisi (koordinat) gedung.Dengan mengunakan fungsi kamera pada perangkat *mobile*tersebut, aplikasi ini dapat mendeteksi gedung yang koordinatnya telah disimpan di basis data.Selanjutnya aplikasi ini akan menampilkan informasi yang berkaitan dengan gedung tersebut, misalnya informasi posisi gedung, nama gedung, fungsi utama gedung.Dengan demikian sistem ini akan memberikan informasi kepada pengunjung dengan mudah, cepat dan akurat.

Berdasarkan latar belakang tersebut penulis mengambil judul "Rancang Bangun Aplikasi Informasi GedungUniversitas Brawijaya pada *Smartphone*Android" dengan harapan sistem informasi ini nantinya dapat menjadi sebuah sistem informasi yang dapat memudahkan civitas akademik Universitas Brawijaya pada khususnya dan masyrakat luas pada umumnya.

1.2 Rumusan Masalah

Berdasarkan uraian latar belakang diatas, maka dapat dirumuskan permasalahnya yaitu:

1. Bagaimana merancang aplikasi informasi gedung Universitas Brawijaya pada smartphone Android?
2. Bagaimana mengimplementasikanaplikasi informasi gedung Universitas Brawijaya pada smartphone Android?
3. Bagaimana membangun sumber data aplikasi informasi gedung Universitas Brwaijaya pada *smartphone* Android?
4. Bagaimana mengujiaplikasi informasi gedung Universitas Brawijaya pada smartphone Android?

1.3 Batasan Masalah

Agar permasalahan yang dirumuskan lebih terfokus dan tidak terjadi pelebaran topik, maka penelitian tugas akhir ini dibatasi dalam hal:

1. Informasi yang ditampilkan adalah informasi posisi gedung, nama gedung, fungsi utama gedung tersebut.
2. Gedung yang terdeteksi adalah gedung-gedung utama seperti gedung universitas, gedung fakultas, gedung jurusan.
3. Perangkat lunak untuk sistem informasi gedung Universitas Brawijaya menggunakan *AR engine*mixare.
4. Sistem informasi gedung Universitas Brawijaya menggunakan perangkat *mobile* berbasis *Android*minimal versi 2.2 (*froyo*).
5. Sumber data yang digunakan adalah sumber data yang dibangun oleh penulis.
6. Pengujian performa difokuskan pada nilai *transfer rate* dan *request per second* sebagai *throughput* dari jumlah *request* yang dikirimkan ke *server*.
7. Performa penguncian posisi koordinat *user* tergantung performa GPS dari *device.*

1.4 Tujuan

Tujuan penulisan tugas akhir ini:

1. Merancang aplikasi informasi gedung Universitas Brawijaya pada smartphone Android.
2. Mengimplementasikan aplikasi informasi gedung Universitas Brawijaya pada smartphone Android.
3. Membangun sumber data aplikasi informasi gedung Universitas Brawijaya pada *smartphone* Android.
4. Melakukan pengujianaplikasi informasi gedung Universitas Brawijaya pada smartphone Android.

1.5 Manfaat

A. Bagi Pengguna Sistem

1. Mendapatkan informasi yang cepat, akurat dan *realtime*mengenai gedung-gedung yang ada di Universitas Brawijaya.

B. Bagi Penulis

1. Memenuhi syarat kelulusan dari kegiatan perkuliahan di Teknik Informatika Universitas Brawijaya.
2. Mengaplikasikan ilmu yang telah didapat selama perkuliahan.
3. Memperdalam ilmu pengetahuan tentang aplikasi *Augmented Reality*.
4. Memperdalam ilmu pengetahuan tentang sistem operasi Android.

1.6 Sistematika Penulisan

Sistematika penulisan laporan tugas akhir ini adalah:

Bab I Pendahuluan

Bab pendahuluan berisi tentang latar belakang, rumusan masalah, batasan masalah, tujuan penelitian, manfaat penelitian, dan sistematika penulisan.

Bab II Dasar Teori

Bagian ini membahas tentang dasar teori secara luas mengenai *AR (Augmented Reality)* dan teori pendukungnya untuk membangun sebuah sistem informasi dengan *AR* dengan media perangkat *mobile*.

Bab III Metodologi Peneletian

Pada bab ini dijelaskan langkah-langkah yang akan dilakukan dalam perancangan, implementasi dan pengujian dari aplikasi rancang bangun aplikasi informasi gedung Universitas Brawijaya pada smartphone Android.

Bab IVPerancangan

Bab ini berisi tentang perancangan dan analisis sistem yang dibangun.

Bab V Implementasi

Bab implementasimenjelaskan realisasi dari perancangan dan pengumpulan teori mengenai sistem informasi yang akan dibuat.

Bab VI Pengujian dan Analisis

Bab ini berisi pengujian pada perangkat *mobile* untuk menampilkan informasi-informasi gedung Universitas Brawijaya.

Bab VII Penutup

Bab ini berisi kesimpulan yang diambil berdasarkan analisa hal-hal penting, meliputi keunikan, kelebihan atau kekurangan, serta saran–saran untuk penyempurnaan sistem yang dibuat.

BAB II
LANDASAN TEORI

2.1 Pengertian Informasi

Secara umum informasi dapat didefinisikan sebagai hasil dari pengolahandatadalam suatu bentuk yang lebih berguna dan lebih berarti bagi penerimanyayang menggambarkan suatu kejadian-kejadian yang nyata yang digunakan untukpengambilan keputusan. Informasi merupakan data yang telah diklasifikasikanatau diolah atau diinterpretasi untuk digunakan dalam proses pengambilan keputusan.

2.2 *AR* (*Augmented Reality*)

Augmented Reality(AR)adalahpenggabungan benda-benda nyata dan maya di lingkungan nyata, berjalan secara interaktif dalam waktu nyata, dan terdapat integrasi antarbenda dalam tiga dimensi, yaitu benda maya terintegrasi dalam dunia nyata [AZU-97]. Penggabungan benda nyata dan maya dimungkinkan dengan teknologi tampilan yang sesuai, interaktivitas dimungkinkan melalui perangkat-perangkat input tertentu, dan integrasi yang baik memerlukan penjejakan yang efektif. Selain menambahkan benda maya dalam lingkungan nyata, realitas tertambah juga berpotensi menghilangkan benda-benda yang sudah ada. Menambah sebuah lapisan gambar maya dimungkinkan untuk menghilangkan atau menyembunyikan lingkungan nyata dari pandangan pengguna. Misalnya, untuk menyembunyikan sebuah meja dalam lingkungan nyata, perlu digambarkan lapisan representasi tembok dan lantai kosong yang diletakkan di atas gambar meja nyata, sehingga menutupi meja nyata dari pandangan pengguna.

AR dibedakan menjadi empat bagian menurut media yang digunakan yaitu *simple AR, MarkerBased AR, Markerless AR* dan *Augmented Vision.Markerlees AR* umumnya diterapkan dalam *mobile device* seperti *smartphone*. Sesuai dengan namanya *markerless AR* tidak membutuhkan marker yang terlihat secara fisik untuk mengetahui posisi suatu objek. Sebagai gantinya, digunakan informasi dari

GPS atau kompas dan cara ini dikenal dengan nama *geotagging* dan *geolocation*. Melalui *geotagging* dan *geolocation* yang bisa kita sebut sebagai marker yang tidak terlihat inilah konten seperti tulisan, video, maupun audio kemudian ditampilkan dilayar *mobile device* tersebut.

Salah satu metode AR yang saat ini sedang berkembang adalah metode "Markerless*Augmented Reality*", dengan metode ini pengguna tidak perlu lagi menggunakan sebuah marker untuk menampilkan elemen-elemen digital. Seperti yang saat ini dikembangkan oleh perusahaan *Augmented Reality* terbesar di dunia Total Immersion, mereka telah membuat berbagai macam teknik Markerless Tracking sebagai teknologi andalan mereka, seperti Face Tracking, 3D Object Tracking, Motion Tracking, dan GPS Based Tracking. Dalam skripsi ini, teknologi yang digunakan adalah GPS Based Tracking.

Teknik GPS Based Tracking saat ini mulai populer dan banyak dikembangkan pada aplikasi smartphone (IPhone dan Android). Dengan memanfaatkan fitur GPS dan Kompas yang ada didalam smartphone, aplikasi akan mengambil data dari GPS dan Kompas kemudian menampilkannya dalam bentuk arah yang kita inginkan secara realitme, bahkan ada beberapa aplikasi menampikannya dalam bentuk 3D. Salah satu pelopor GPS Based Tracking adalah aplikasi yang bernama Layar.

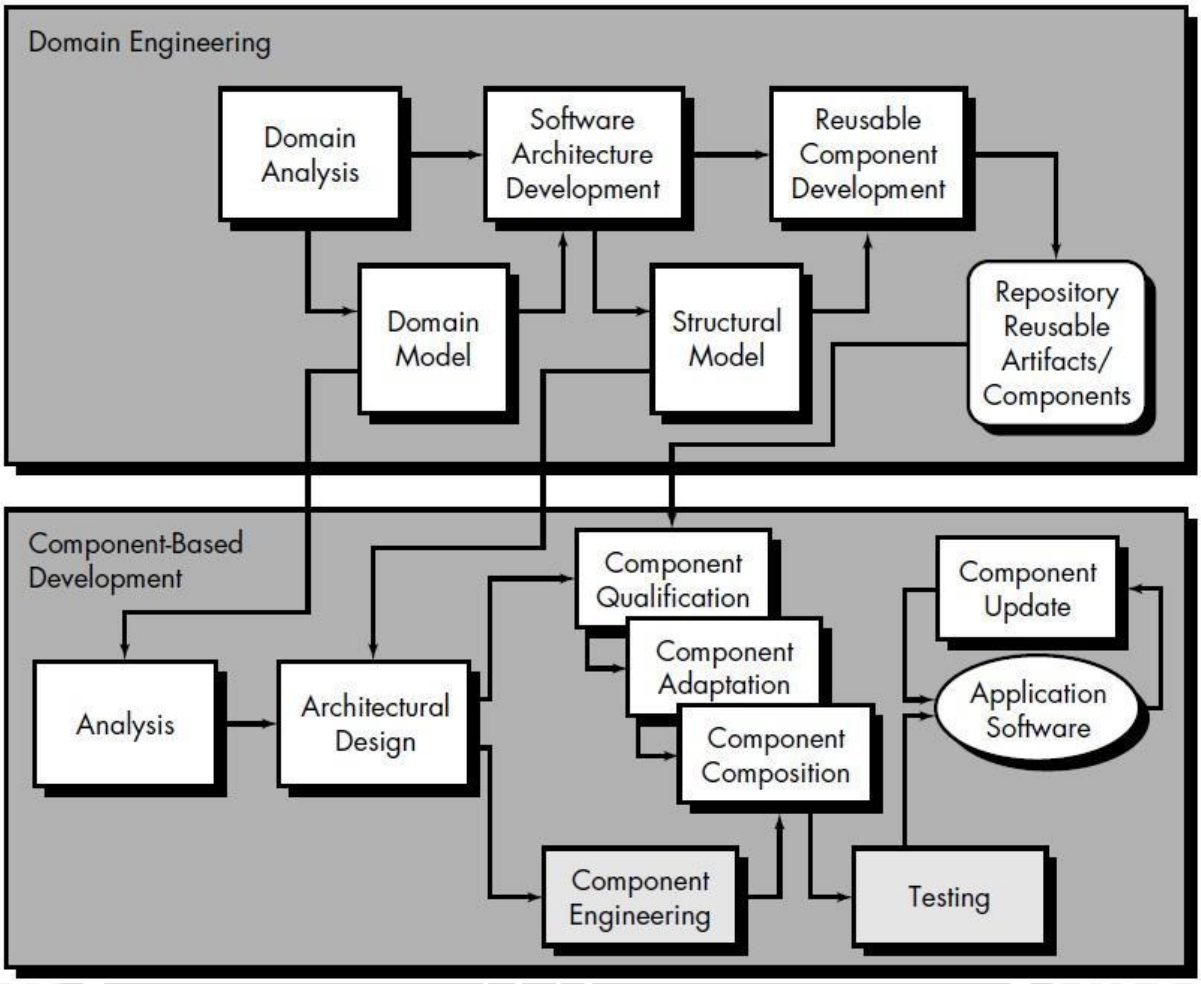
Gambar 2.1. Aplikasi *Markerless AR* pada *smartphone*

Sumber : [MIX-11]

2.3. *Component-Based Software Engineering*

Component-Based Software Engineering (CBSE) adalah metode pengembangan perangkat lunak yang menekankan pada perancangan dan

pembangunan perangkat lunak dengan menggunakan komponen perangkat lunak yang sudah ada dan bersifat dapat digunakan kembali. [PRE-01:721]. CBSE memiliki *software process* yang terdiri dari dua bagian utama yang berjalan paralel, yaitu *domain engineering* dan *component-based development*. Alur *software process* dari CBSE digambarkan pada Gambar 2.1.



Gambar 2.2*Software process* dari CBSE

Sumber : [PRE-01:725]

2.3.1. *Domain Engineering*

Domain Engineering bertujuan untuk mengidentifikasi, membangun, mengumpulkan, dan mengalokasikan sekumpulan komponen perangkat lunak yang dapat dipakai pada pengembangan perangkat lunak pada saat ini dan di masa yang akan datang.

1. Domain Analysis

Pada tahap *domain analysis* akan dilakukan penggalian informasi terkait perangkat lunak yang akan dikembangkan melalui metode analisis kebutuhan sesuai dengan teknik *software analysis*.Penerapan *domanin analysis* pada skripsi ini adalah pada *softwarerequirement spesification* (srs).

2. Domain Model

Dalam *Domain Model* akan dilakukan pemodelan terhadap hasil *domain analysis* yang telah dilakukan sebelumnya. Pemodelan dapat dilakukan dengan *unified modelling language* untuk analisis yang bersifat *object-oriented.Domain model* diwakili oleh penggunaan UML pada tahap perancangan.

3. Software Architecture Development

Software architecture development merupakan tahap dimana terjadi perancangan pola arsitektur perangkat lunak yang nantinya akan dipakai pada pengembangan perangkat lunak yang akan dibangun. Pada skripsi ini ditunjukkan dalam perancangan arsitektur perangkat lunak.

4. Structural Model

Structural model adalah tahapan dimana dilakukan analisis lebih lanjut tentang pemodelan struktural perangkat lunak berdasarkan perancangan pola arsitektur perangkat lunak yang telah dilakukan. Analisis ini umumnya dilakukan dengan melakukan *break down* modul – modul fungsional dari perangkat lunak yang akan dibangun.

5. Reusable Components Development

Pada tahap *reusable component development* dilakukan proses identifikasi dan pencarian kebutuhan *software component* yang akan digunakan dalam membangun perangkat lunak.Proses *reusable component development* diwakili oleh *deployment diagram* yang disusun oleh komponen-komponen yang digunakan untuk menyusun aplikasi Informasi Gedung UB .

6. Repository Reusable Artifacts / Components

Repository reusable artifacts / components adalah tahapan dimana komponen – komponen perangkat lunak dikumpulkan dan disimpan dalam bentuk *repository* sehingga siap diimplementasikan pada bagian *component-based development* atau dapat digunakan kembali di masa yang akan datang. Pada skripsi ini diwakili oleh penjelasan tentang komponen-komponen yang menyusun perangkat lunak yang digambarkan pada *deployment diagram.*

2.3.2. *Component-Based Development*

Component-based development bertujuan untuk melakukan pembangunan sistem perangkat lunak dengan menggunakan komponen – komponen perangkat lunak yang bersifat *reusable.*

1. *Analysis*

Pada tahap *analysis* ini akan dilakukan analisis kebutuhan lebih lanjut terkait dengan kebutuhan komponen – komponen yang dibutuhkan untuk membangun perangkat lunak berdasarkan analisis kebutuhan fungsional.

2. *Architectural Design*

Tahap *architectural design* merupakan tahapan perancangan arsitektural dari perangkat lunak yang akan dibangun. Perancangan arsitektural ini dilakukan berdasarkan pemodelan kebutuhan fungsionalitas perangkat lunak dan kebutuhan komponen yang telah dilakukan. Dalam perancangan arsitektural ini akan dijelaskan arsitektur perangkat lunak yang akan dibangun, seperti arsitektur penyimpanan data dan arsitektur *class* terkait fungsionalitas perangkat lunak. Umumnya, perancangan arsitektural ini dilakukan dengan *unified modelling language* apabila menggunakan *object-oriented analysis.*Dalam skripsi ini diwakili oleh *class diagram* dan perancangan basis data.

3. Component Qualification

Pada tahap *component qualification* akan dijelaskan tentang kualifikasi dari komponen perangkat keras dan komponen perangkat lunak yang dibutuhkan

dalam pengembangan perangkat lunak yang akan dibangun. Tahap kualifikasi komponen ini bertujuan untuk mengetahui apakah komponen yang digunakan telah sesuai dan memenuhi syarat - syarat dari kebutuhan perangkat lunak yang dibangun.Pada skripsi ini ditunjukkan dalam spesifikasi sistem.

4. Component Adaptation

Dalam tahap *component adaptation* dilakukan proses adaptasi dan modifikasi dari komponen – komponen perangkat lunak yang dipakai sehingga memenuhi kualifikasi kebutuhan komponen dari perangkat lunak yang dibangun.Penggunaan Mixare merupakan penerapan dari *component adaptation.*

5. Component Engineering

Tahap *component engineering* adalah tahap dimana komponen – komponen dari perangkat lunak yang dibangun, dikembangkan secara mandiri. Pengembangan komponen baru ini umumnya dilakukan dengan teknik *object-oriented programming.* Pada tahap ini, diwakili oleh pembuatan *webserver* secara mandiri.

6. Component Composition

Tahap *component composition* adalah tahap utama dimana dilakukan pembangunan perangkat lunak dengan menggunakan komponen- komponen yang sudah ada. Pada tahap ini dilakukan penggabungan komponen – komponen perangkat lunak menjadi satu perangkat lunak yang padu.Pada tahap ini dilakukan penggabungan komponen – komponen perangkat lunak menjadi satu perangkat lunak yang padu.

7. Testing

Tahap *testing* adalah tahap dilakukannya proses pengujian perangkat lunak yang dibangun menggunakan teknik dan metode pengujian perangkat lunak tertentu. Pengujian dilakukan untuk mengetahui apakah fungsionalitas perangkat lunak sudah benar dan dapat memenuhi kebutuhan pengguna. Tahapan ini diimplementasikan pada bab pengujian.

8. Application Software

Proses distribusi perangkat lunak dilakukan pada tahap *application software.* Perangkat lunak yang telah jadi dan telah diuji didistribusikan kepada pengguna atau klien untuk dapat dipakai sebagaimana mestinya.

9. Component Update

Pada tahap ini dilakukan pembaharuan (*update*) komponen perangkat lunak yang telah dibangun untuk meningkatkan kinerja dari perangkat lunak tersebut. Proses tersebut juga termasuk *maintenance* perangkat lunak yang telah didistribusikan.Tahap ini bisa dilihat pada bagian saran namun tidak diimplementasikan pada skripsi ini.

2.4 Sistem Operasi Android

2.4.1 Pengenalan Android

Android adalah *software* untuk perangkat *mobile* yang mencakup sistem operasi, *middleware* dan aplikasi kunci. SDK (*Software Development Kit*) Android menyediakan alat dan API (*Application Programming Interface*) yang diperlukan untuk mengembangkan aplikasi pada platform Android dengan menggunakan bahasa pemrograman Java [HER-11]. Android menyediakan platform terbuka bagi para pengembang untuk menciptakan aplikasi mereka sendiri untuk digunakan oleh bermacam *mobile devices*.

2.4.2Fitur Android

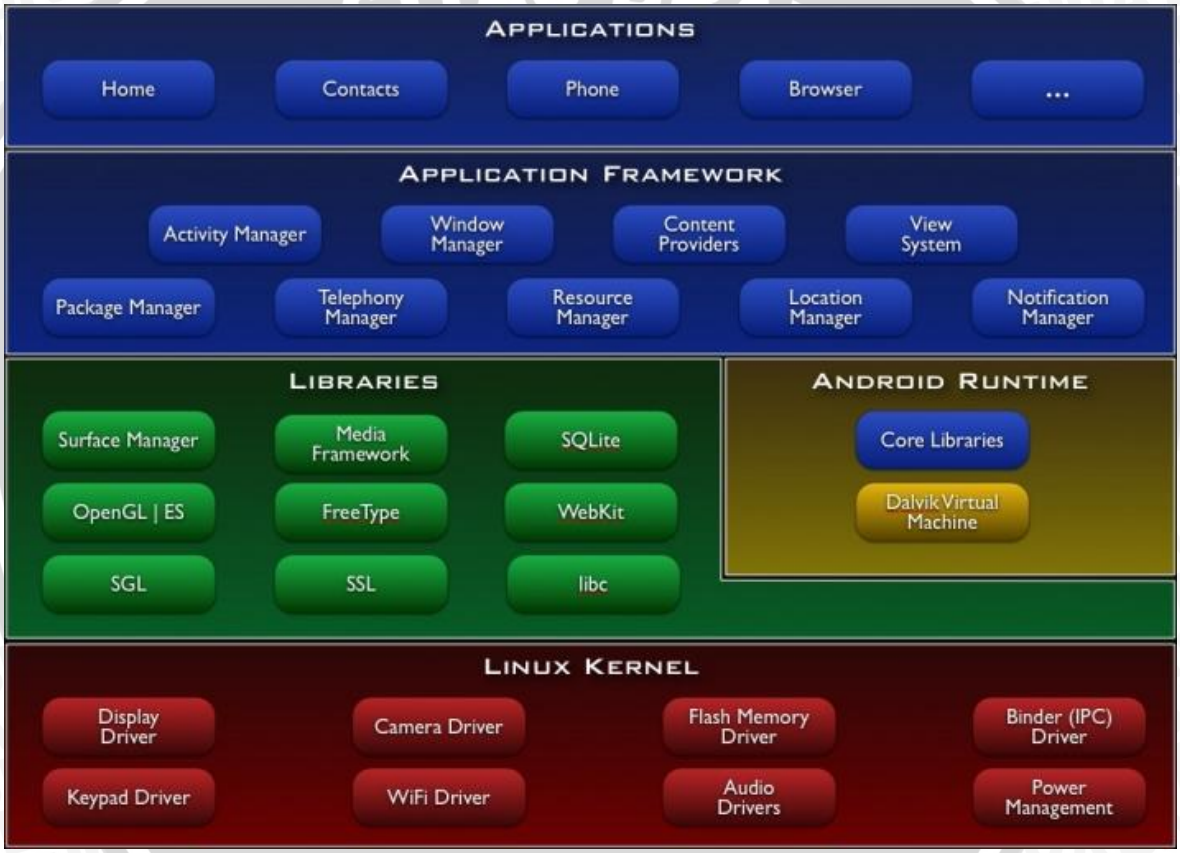
Android juga memiliki beberapa fitur sebagai berikut [HER-11]. :

1. ***Application framework*** memungkinkan penggunaan kembali dan penggantian komponen
2. ***Dalvik virtual machine*** optimalisasi untuk perangkat bergerak
3. ***Integrated browser*** berdasarkan open source WebKit mesin
4. ***Optimized graphics*** didukung *library*
   grafis 2D ; grafis 3D berbasis OpenGL ES 1.0
5. ***SQLite*** untuk penyimpanan data terstruktur
6. ***Media support*** untuk audio, video, dan gambar masih dalam format (MPEG4, H.264, MP3, AAC, AMR, JPG, PNG, GIF)

7. ***GSM Telephony***
8. ***Bluetooth, EDGE, 3G, and WiFi***
9. ***Camera, GPS, compass, and accelerometer***
10. ***Rich development environment*** termasuk *device emulator*, alat untuk *debugging*, *memory* dan *performance profiling*, dan *plugin* untuk Eclipse IDE

2.4.3 Arsitektur Android

Diagram Arsitektur Androidditunjukkan pada gambar 2.3.



Gambar 2.3 Arsitektur *Android*

Sumber : [HER-11]

Sesuai ilustrasi yang ditunjukkan pada gambar 2.3. Arsitektur Android dapat dibagi menjadi lima bagian besar, yaitu:

1. *Applications*

Android memuat seperangkat aplikasi inti mencakup *email client*, program SMS, kalender, peta, *browser*, kontak, dan lain lain. Semua aplikasi ditulis menggunakan bahasa pemrograman Java.

2. *Application Framework*

Android menawarkan kemampuan untuk membangun aplikasi yang sangat kaya dan inovatif. Pengembang bebas untuk mengakses perangkat keras, informasi akses lokasi, menjalankan *background services*, mengatur alarm, tambahkan pemberitahuan ke *statusbar*,dan banyak lagi.

Pengembang memiliki akses penuh ke *framework* API yang sama yang digunakan oleh aplikasi inti. Arsitektur aplikasi dirancang untuk menyederhanakan penggunaan kembali komponen, aplikasi apapun dapat mempublikasikan kemampuannya dan aplikasi lain kemudian dapat menggunakan kemampuan mereka (diatur oleh batasan keamanan yang diberlakukan oleh *framework*). Mekanisme ini mengizinkan *user* mengganti komponen.

Berikut adalah contoh servis dan system yang mendasari seluruh aplikasi Android:

- *View*: tampilan yang dapat digunakan untuk membangun aplikasi, termasuk didalamnya *list*, *grid*, kotak teks, tombol, dan sebagaianya.
- *Content Providers* : memungkinkan aplikasi untuk mengakses data dari aplikasi lain (misal Kontak), atau untuk membagi data yang dimiliki.
- *Resource Manager* : menyediakan akses ke *non-code resources* seperti grafik dan *layout* data.
- *Notification Manager*: memungkinkan aplikasi untuk menampilkan peringatan pada *status bar*.
- *Activity Manager*: mengatur *lifecycle* aplikasi.

3. *Libraries*

Libraries adalah *layer* dimana fitur-fitur Android berada, biasanya para pembuat aplikasi Android mengakses *libraries* untuk menjalankan aplikasinya. Android mencakup set *library* C / C++ yang digunakan oleh berbagai komponen dari sistem Android. Beberapa *library* tercantum seperti dibawah ini:

- Sistem C library : *library* standard C (libc)

- Media Libraries : *library* untuk mendukung memutar dan merekam berbagai format *audio* dan *video*, juga file gambar.
- Surface Manager : mengatur akses ke tampilan *subsistem* dan memadukan grafik 2D dan 3D pada beberapa aplikasi.
- LibWeb Core : *web browser engine* yang mendukung Android*browser*.
- SGL : mendasari *engine* grafik 2D.
- 3D libraries : implementasi OpenGL ES 1.0 APIs berfungsi mengoptimalkan fungsionalitas 3D.
- FreeType : *rendering bitmap* dan *font.*
- SQLite : *engine* basis data yang ampuh dan ringan yang tersedia untuk semua aplikasi.

4. *Android Runtime*

Android mencakup seperangkat *library* inti yang menyediakan sebagian besar fungsi yang tersedia di *library* inti dari bahasa pemrograman Java. Setiap aplikasi Android berjalan dalam prosesnya sendiri, dengan Dalvik *Virtual Machine*-nya (VM) sendiri. Dalvik VM mengeksekusi *file* dalam format Dalvix *excetubale* (.dex) yang dioptimalkan untuk meminimalisir jejak memori.

5. *Linux Kernel*

Android bergantung pada Linux versi 2.6 untuk layanan sistem inti seperti keamanan, manajemen memori, manajemen proses, jaringan *stack*, dan *driver model*.

2.4.4 Fundamental Aplikasi

Aplikasi Android ditulis dalam bahasa pemrograman java. Kode java dikompilasi bersama dengan data file resource yang dibutuhkan oleh aplikasi, dimana proses di-package oleh tools yang dinamakan "apt tools" ke dalam paket Android sehingga menghasilkan file dengan ekstensi apk. File apk itulah yang kita sebut dengan aplikasi, dan nantinya dapat diinstall diperangkat mobile.

Ada enam jenis komponen pada aplikasi Android, yaitu [SAF-11] :

1. *Activities*

Suatu *activity*akan menyajikan *user interface (*UI*)* pada pengguna, sehingga pengguna dapat melakukan interaksi. Sebuah aplikasi android bisa jadi hanya memiliki satu *activity*, tetapi umumnya aplikasi memiliki banyak *activity*

tergantung pada tujuan aplikasi dan desain dari aplikasi tersebut. Satu *activity* biasanya akan dipakai untuk menampilkan aplikasi atau yang bertindak sebagai *user interface* (UI) saat aplikasi diperlihatkan kepada *user*. Untuk pindah dari satu *activity* ke *activity* lain kita dapat melakukannya dengan satu *event*, misalnya klik tombol, memilih opsi atau menggunakan *triggers* tertentu. Secara hirarki sebuah windows *activity* dinyatakan dengan method *Activity.setContentView()*. *ContentView* adalah objek yang berada pada *root* hirarki.

2. *Services*

Service tidak memiliki *Graphic User Interface* (GUI), tetapi *service* berjalan secara *background*, sebagai contoh dalam memainkan musik, *service* mungkin memainkan musik atau mengambil data jaringan, tetapi *service* harus berada dalam kelas induknya. Misalnya, media player sedang memutar lagu dari *list* yang ada, aplikasi ini akan memiliki dua atau lebih *activity* yang memungkinkan *user* untuk memilih lagu misalnya, atau *sms* sambil *player* sedang jalan.

3. *Broadcast Receiver*

Broadcast receiver berfungsi menerima dan bereaksi untuk menyampaikan notifikasi. Contoh *broadcast* seperti notifikasi zona waktu berubah, baterai *low*, gambar telah selesai diambil oleh kamera, atau mengubah referensi bahasa yang digunakan. *Broadcast receiver* tidak memiliki *user interface* (UI), tetapi memiliki *activity* untuk merespon informasi yang mereka terima.

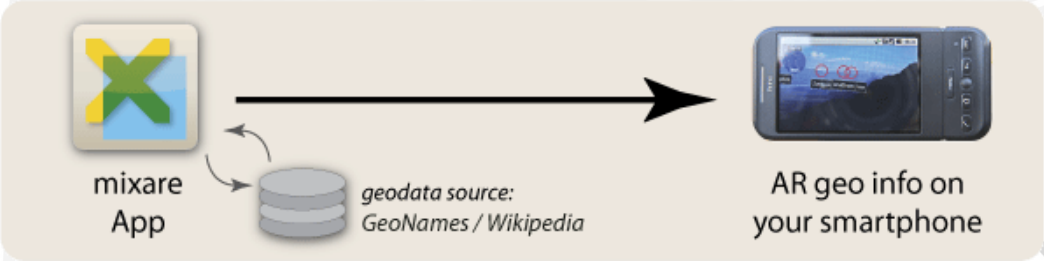
4. *Content Provider*

Content provider membuat kumpulan aplikasi data secara spesifik sehingga bisa digunakan oleh aplikasi lain. Data disimpan dalam *file* sistem seperti *SQLite*. *Content provider* menyediakan cara untuk mengakses data yang dibutuhkan olh suatu *activity*, misalnya ketikan kita menggunakan aplikasi yang membutuhkan peta (*Map*), atau aplikasi yang membutuhkan untuk mengakses data kontak dan navigasi, maka disinilah fungsi *content provider*.

2.5 *Mixare*

Mixare (Mix Augmented Reality Engine) adalah open source augmented reality browser yang diterbitkan dibawah GPLv3. Mixare tersedia untuk Android

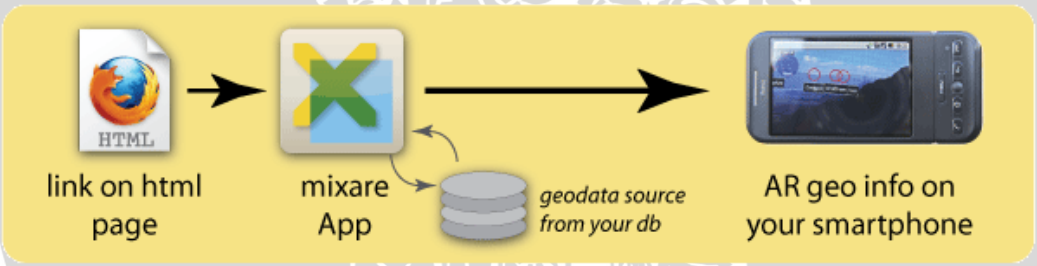
dan iPhone 3GS dan diatasnya. Mixare bekerja sebagai aplikasi yang sangat otonom dan tersedia untuk pengembangan dari implementasi sendiri.Ada beberapa cara dalam menggunakan mixare.Mixare adalah sebuah aplikasi otonom yang menampilkan POI Wikipedia terhadap lingkungannya. Mixare dijalankan di perangkat *mobile* secara langsung.



Gambar 2.4 Penggunaan Mixare Secara Langsung

Sumber : [MIX-11]

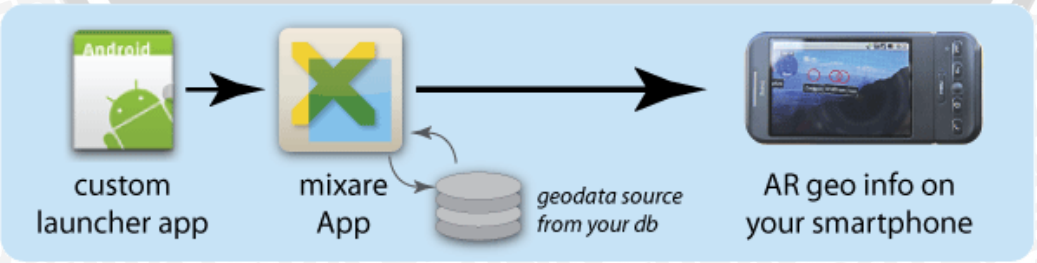
2. Mixare dapat diakses oleh link pada situs HTML, dimana sumber data ditransfer ke aplikasi.



Gambar 2.5 Penggunaan Mixare Melalui *LinkHTML*

Sumber : [MIX-11]

3. Mixare dapat diakses oleh launcher-app kita sendiri, data source ditransfer ke aplikasi yang kita buat.

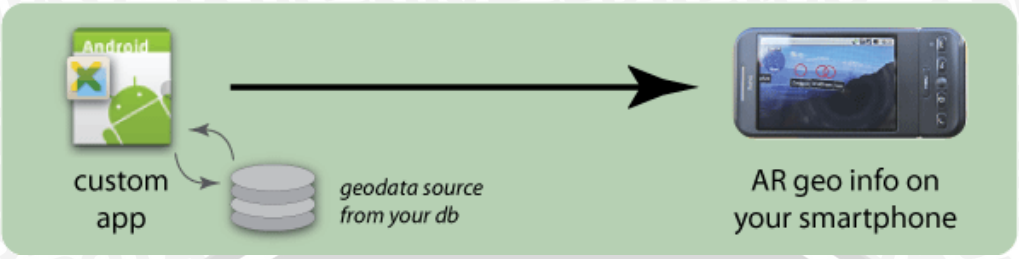


Gambar 2.6 Penggunaan Mixare dengan Launcher

Sumber : [MIX-11]

4. Mixare secara bebas di upgrade dan bahkan dapat dimodifikasi ke

aplikasi individu.



Gambar 2.7 Penggunaan Mixare dengan Pengembangan Sendiri

Sumber : [MIX-11]

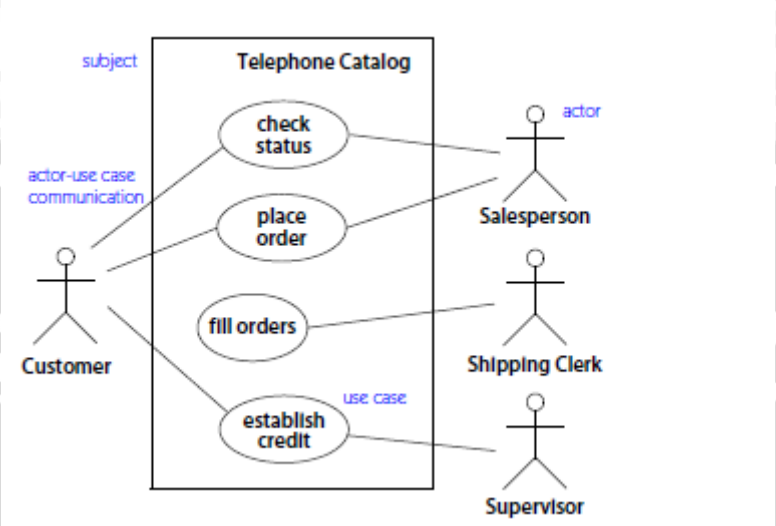
2.6 Unified Modeling Language

Unified modeling language (UML)adalah sebuah bahasa yangtelah menjadi standar dalam industri untuk visualisasi, merancang, dan mendokumentasikan sistem piranti lunak. UML menawarkan standar untuk merancang model sebuah sistem. Dengan menggunakan UML, kita dapat membuat model untuk semua jenis aplikasi piranti lunak, dimana aplikasi tersebut dapat berjalan pada piranti keras, sistem operasi, dan jaringan apapun, serta dapat juga ditulis dengan bahasa pemrograman apapun. Karena UML menggunakan *class* dan *operation* dalam implementasinya, maka lebih cocok dengan menggunakan bahasa berorientasi objek. Seperti C++, java, atau VB.NET. Tetapi dapat juga digunakan pada aplikasi yang bersifat procedural seperti VB atau C.

Seperti bahasa yang lain, UML memiliki notasi dan sematik. Notasi UML merupakan sekumpulan bentuk khusus untuk menggambarkan berbagai diagram piranti lunak. Setiap bentuk memilikai makna tertentu, dan UML mendefinisikan bagaimana bentuk-bentuk tersebut dapat dikombinasikan. Notasi UML terutama diturunkan dari 3 notasi yang telah ada sebelumnya. Grady Booch OOD (Object-Oriented Design), Jim Rumbaugh OMT (Object Modeling Technique), dan Ivar Jacobson OOSE (Object-Oriented Software Engineering). Dalam pemodelan UML, dapat digambarakan dengan diagram, dimana masing-masingnya memiliki fungsi yang berbeda. Beberapa diagram diantaranya adalah :

2.6.1 *Use Case Diagram*

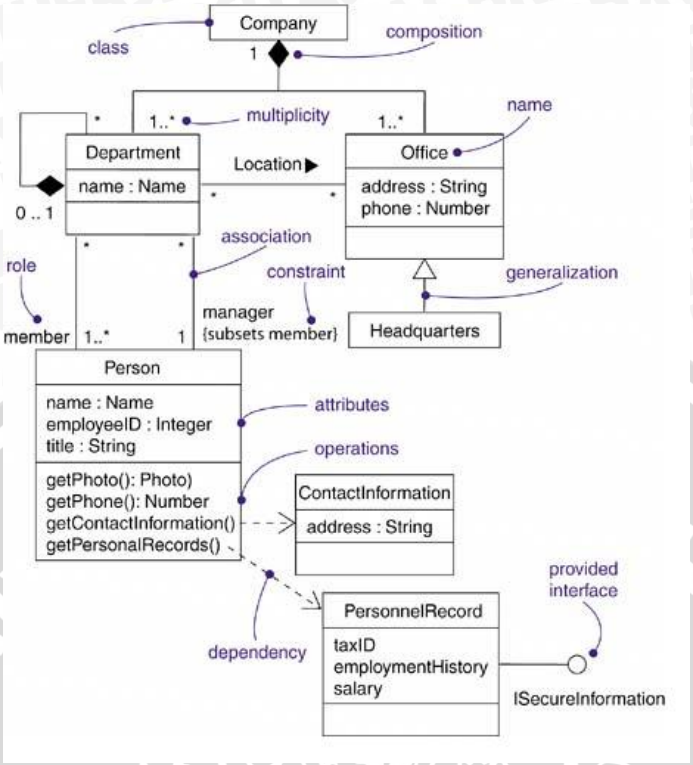
diagram *use case* adalah diagram yang menunjukkan satu set *use case* dan aktor dan hubungan mereka.[WES-05]. Gambar 2.8 merupakan contoh dari diagram*Use Case*.



Gambar 2.8 Contoh *Use Case Diagram*
Sumber: [WES-06]

2.6.2 *Class Diagram*

Sebuah diagram kelas menunjukkan satu set kelas, interface, dan kolaborasi dan hubungan mereka. Diagram kelas adalah diagram yang paling umum ditemukan dalam pemodelan sistem berorientasi objek. Anda menggunakan diagram kelas untuk menggambarkan tampilan desain statis dari sebuah sistem. Kelas diagram yang mencakup kelas aktif digunakan untuk mengatasi pandangan proses statis dari sebuah sistem. [WES-05]. Gambar 2.9 merupakan contoh dari class diagram.

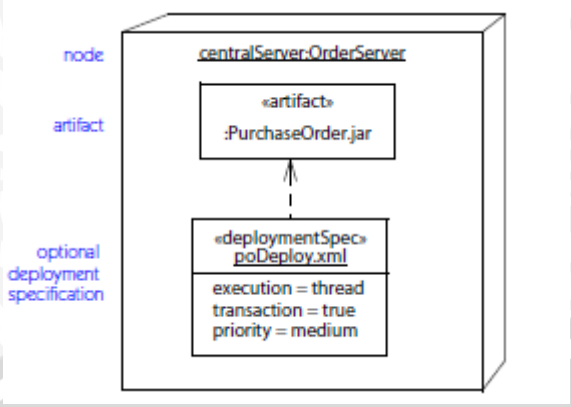


Gambar 2.9 Contoh *Class Diagram*

Sumber :[WES-05]

2.6.3 *Deployment Diagram*

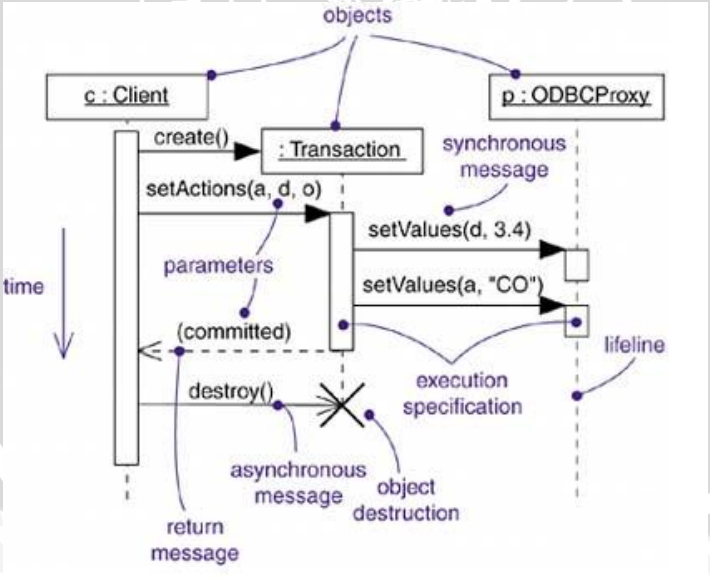
Deployment Diagram adalah diagram yang menunjukkan konfigurasi run-time node pemrosesan dan artefak yang hidup pada mereka. *Deploument diagram* banyak berada pada tingkat kelas atau tingkat *instance*[WES-05]. *Deployment/physical diagram* menggambarkan detail bagaimana komponen di-*deploy* dalam infrastruktur sistem, di mana komponen akan terletak (pada mesin, server atau piranti keras apa), bagaimana kemampuan jaringan pada lokasi tersebut, spesifikasi server, dan hal-hal lain yang bersifat fisikal. Sebuah *node* adalah server, *workstation*, atau piranti keras lain yang digunakan untuk men-*deploy* komponen dalam lingkungan sebenarnya. Hubungan antar *node* (misalnya TCP/IP) dan *requirement* dapat juga didefinisikan dalam diagram ini.Gambar 2.10 merupakan contoh dari class diagram.



Gambar 2.10 Contoh*Deployment Diagram*
Sumber:[WES-05]

2.6.4 *Sequence Diagram*

Diagram *sequence* adalah diagram interaksi yang menekankan waktu pemesanan pesan.Secara grafis, diagram *sequence* adalah tabel yang menunjukkan benda diatur sepanjang sumbu X dan pesan, memerintahkan dalam waktu yang meningkat, sepanjang sumbu Y [WES-06]. Gambar 2.11 merupakan contoh dari sequence diagram.



Gambar 2.11 Contoh *Sequence Diagram*
Sumber:[WES-05]

2.7 GPS

GPS (Global Positioning System) adalah sistem navigasi yang berbasiskan satelit yang saling berhubungan yang berada di orbitnya. Satelit-satelit itu milik Departemen Pertahanan (Departemen of Defense) Amerika Serikat yang pertama kali diperkenalkan mulai tahun 1978 dan pada tahun 1994 sudah memakai 24 satelit. Untuk dapat mengetahui posisi seseorang maka diperlukan alat yang diberinama GPS reciever yang berfungsi untuk menerima sinyal yang dikirim dari satelit GPS [PRA-05]. Posisi di ubah menjadi titik yang dikenal dengan nama Way-point nantinya akan berupa titik-titik koordinat lintang dan bujur dari posisi seseorang atau suatu lokasi kemudian di layar pada peta elektronik.

2.7.1 Fungsi GPS

Sistem ini didesain untuk memberikan posisi dan kecepatan tiga-dimensi serta informasi mengenai waktu, secara kontinyu di seluruh dunia tanpa bergantung waktu dan cuaca secara secara cepat, akurat, murah.

2.7.2 Ketelitian GPS

Ketelitian dari GPS dapat mencapai beberapa mm untuk ketelitian posisinya, beberapa cm/s untuk ketelitian kecepatannya dan beberapa nanodetik untuk ketelitian waktunya. Ketelitian posisi yang diperoleh akan tergantung pada beberapa faktor yaitu metode penentuan posisi, geometri satelit, tingkat ketelitian data, dan metode pengolahan datanya.

2.7.3 Cara Kerja GPS

Setiap daerah di atas permukaan bumi ini minimal terjangkau oleh 3-4 satelit.
Pada prakteknya, setiap GPS terbaru bisa menerima sampai dengan 12 chanel satelit sekaligus. Kondisi langit yang cerah dan bebas dari halangan membuat GPS dapat dengan mudah menangkap sinyal yang dikirimkan oleh satelit. Semakin banyak satelit yang diterima oleh GPS, maka akurasi yang diberikan juga akan semakin tinggi. Cara kerja GPS secara logik ada 5 langkah:

1. Memakai perhitungan “triangulation” dari satelit.
2. Untuk perhitungan “triangulation”, GPS mengukur jarak menggunakan travel time sinyal radio.

3. Untuk mengukur travel time, GPS memerlukan memerlukan akurasi waktu yangtinggi.
4. Untuk perhitungan jarak, kita harus tahu dengan pasti posisi satelit dan ketingianpada orbitnya.
5. Terakhir harus menggoreksi delay sinyal waktu perjalanan di atmosfer sampaiditerima reciever.

Satelit GPS berputar mengelilingi bumi selama 12 jam di dalam orbit yang akurat dia dan mengirimkan sinyal informasi ke bumi. GPS reciever mengambil informasi itu dandengan menggunakan perhitungan "triangulation" menghitung lokasi user dengan tepat.GPS reciever membandingkan waktu sinyal di kiirim dengan waktu sinyal tersebut di terima.Dari informasi itu didapat diketahui berapa jarak satelit. Dengan perhitungan jarak jarakGPS reciever dapatmelakukan perhitungan dan menentukan posisi user dan menampilkan dalam peta elektronik.Sebuah GPS reciever harus mengunci sinyal minimal tiga satelit untuk memenghitung posisi 2D (latitude dan longitude) dan track pergerakan. Jika GPS recieverdapat menerima empat atau lebih satelit, maka dapat menghitung posisi 3D (latitude,longitude dan altitude). Jika sudah dapat menentukan posisi user, selanjutnya GPS dapatmenghitung informasi lain, seperti kecepatan, arah yang dituju, jalur, tujuan perjalanan,jarak tujuan, matahari terbit dan matahari terbenam dan masih banyak lagi. Satelit GPS dalam mengirim informasi waktu sangat presesi karena Satekittersebut memakai jam atom. Jam atom yang ada pada satelit jalam dengan partikel atomyang di isolasi, sehingga dapatmenghasilkan jam yang akurat dibandingkan dengan jam biasa.Perhitungan waktu yang akurat sangat menentukan akurasi perhitungan untukmenentukan informasi lokasi kita. Selain itu semakin banyak sinyal satelit yang dapatditerima maka akan semakin presesi data yang diterima karena ketiga satelit mengirimpseudo-random code dan waktu yang sama. Ketinggian itumenimbulkan keuntungan dalam mendukung proses kerja GPS, bagi kita karena semakin tinggi maka semakin bersih atmosfer, sehingga gangguansemakin sedikit dan orbit yang cocok dan perhitungan matematika yang cocok. Satelit harus tetap pada posisi yang tepat sehingga stasiun di bumi harus terus memonitor setiappergerakan satelit, dengan bantuan radar yang presesi salalu di cek tentang altitude,posision dan kecepatannya.

2.7.4 *Asissted-Global Positioning System* (A-GPS)

Frank Van Diggelen[DIG-09:1] mengatakan bahwaA-GPS merupakan penyempurnaan dari GPS sebagai satelit penentu posisi di belahan bumi.Satelit GPS yang dimiliki bumi mempunyai konstelasi 24 satelit dalam enam orbit yang mendekati lingkaran, setiap orbit ditempati oleh empat buah satelit dengan interval antara yang tidak sama. Orbit satelit GPS berinklinasi 550° terhadap bidang equator dengan ketinggian rata-rata dari permukaan bumi sekitar 20.200 km [ELR-06:1].

Metode *Advanced Positioning* yang terdapat pada A-GPS merupakan metode penentuan posisi yang paling tinggi akurasinya dibandingkan metode deteksi posisi lainnya seperti *Time Difference Of Arrival* (TDOA) maupun *Enhanced Observed Time Difference* (E-OTD). A-GPS jauh lebih efisien dan efektif dalam mengakses informasi dari satelit karena tidak perlu mencari data satu persatu dari ke-24 satelit yang ada, namun A-GPS telah mengetahui sasaran (satelit) mana yang dibutuhkan atau dituju[ELR-06:125].

Pada sistim A-GPS, telepon genggam akan menangkap sinyal satelit yang lalu dikirimkan ke*server* penyedia layanan telepon, hasil perhitungan lokasi yang dilakukan oleh *server*dikirimkan kembali ke telepon genggam. Peta juga dapat dikirimkan oleh *server* tersebut atau sudah disimpan pada telepon genggam. Sistem ini hanya berfungsi bilajaringan telepon genggam mampu disediakan oleh pengguna. Proses perhitungan data dilakukan oleh *server* penyedia jaringan telepon, maka telepon genggam tidakmemerlukan *prosesor* yang canggih[ELR-06:125-126].

2.8. Pengujian Perangkat Lunak

Pengujian (*testing*)arsitektur dari perangkat lunak berorientasi objek menghasilkan sekumpulan *layered subsystems* yang mengenkapsulasi kelas-kelas yang berkolaborasi. Setiap elemen sistem (subsistem dan *class*) melakukan fungsi yang membantu untuk mencapai kebutuhan sistem. Hal ini sangat penting untuk menguji sebuah *object-orientedsystem* pada berbagai macam level yang berbeda dalam sebuah usaha untuk menemukan kesalahan-kesalahan yang mungkin terjadi

dari kolaborasi kelas-kelas dan komunikasi subsistem melewati *architetural layer* [PRE-01:631].

2.8.1 Teknik Pengujian

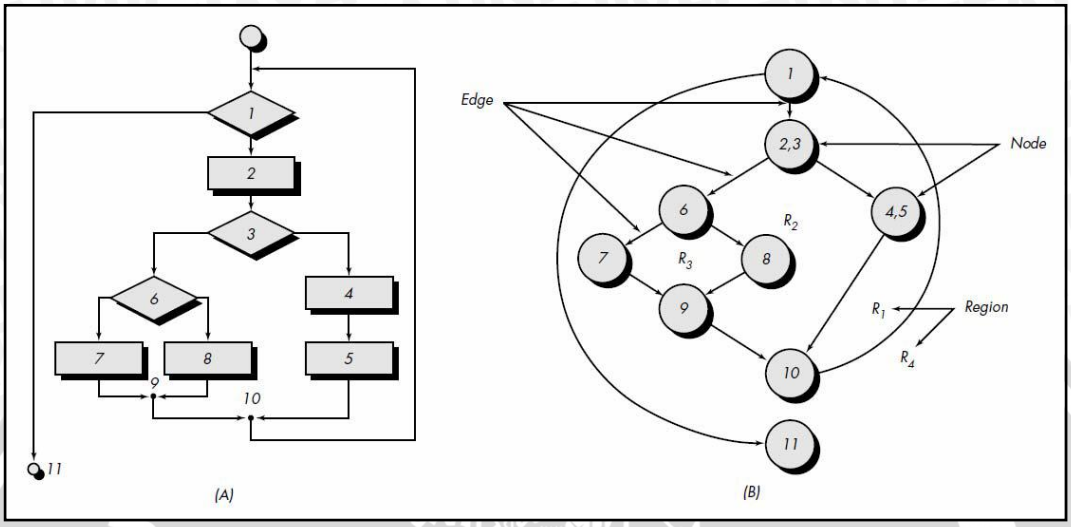
Pengujian perangkat lunak memerlukan perancangan kasus uji (*test case*) agar dapat menemukan kesalahan dalam waktu singkat dan usaha minimum. Berbagai macam metode perancangan kasus uji telah berevolusi. Metode-metode ini menyediakan *developer* pendekatan sistematis untuk pengujian. Terlebih lagi metode-metode ini menyediakan mekanisme yang dapat membantu memastikan kelengkapan dari pengujian dan menyediakan kemungkinan tertinggi untuk menemukan kesalahan-kesalahan dalam perangkat lunak [PRE-01:443]. Teknik atau metode perancangan kasus uji yang digunakan adalah *black-box testing* dan *white-box testing*.

1. *White-Box Testing*

White-box testing atau *glass-box testing* merupakan sebuah metode perancangan kasus uji yang menggunakan struktur kontrol dari perancangan prosedural untuk memperoleh kasus uji [PRE-01:444]. Ada dua jenis pengujian yang termasuk *white-box testing* yaitu *basis path testing* dan *control structure testing.*

Pada skripsi ini menggunakan *basis path testing* yang diusulkan pertama kali oleh Tom McCabe [PRE-01:445]. *Basis path testing* ini memungkinkan perancang kasus uji memperoleh ukuran kompleksitas logis dari sebuah perancangan prosedural dan menggunakan pengukuran ini sebagai pedoman untuk mendefinisikan *basis set* dari jalur eksekusi (*execution path*). *Test case* yang dilakukan untuk menggunakan *basis set* tersebut dijamin untuk menggunakan setiap *statement* di dalam program paling tidak sekali selama pengujian. Sebelum metode *basis path* dapat diperkenalkan, notasi sederhana untuk representasi aliran kontrol yang disebut diagram alir (*flow graph*) harus diperkenalkan. Setiap representasi desain prosedural yang berupa *flow chart* dapat diterjemahkan ke dalam *flow graph*. Gambar 2.12 menunjukkan transformasi *flow chart* ke *flow*

graph. Setelah *flow graph* didefinisikan maka harus ditentukan ukuran kompleksitas (*cyclomatic complexity*).



Gambar 2.12 Transformasi *flow chart* ke *flow graph*

Sumber : [PRE-01:447]

Cyclomatic complexity adalah metriks perangkat lunak yang memberikan pengukuran kuantitatif terhadap kompleksitas logis suatu program. Bila metriks ini digunakan dalam konteks metode pengujian *basis path*, maka nilai yang terhitung untuk *cyclomatic complexity* menentukan jumlah jalur independen (*independent path*) dalam *basis set* suatu program dan memberi batas atas bagi jumlah pengujian yang harus dilakukan untuk memastikan bahwa semua *statement* telah dieksekusi sedikitnya satu kali.

Jalur independen adalah jalur yang melalui program yang mengenalkan sedikitnya satu rangkaian *statement* proses baru atau suatu kondisi baru. Untuk menentukan *cyclomatic complexity* bisa dilakukan dengan beberapa cara, diantaranya [PRE-01:448]:

1. Jumlah *region* pada *flow graph* sesuai dengan *cyclomatic complexity*.
2. *Cyclomatic complexity* V(G), untuk grafik G adalah V(G) = E – N + 2, dimana E adalah jumlah *edge*, dan N adalah jumlah *node.*
3. V(G) = P + 1, dimana P adalah jumlah *predicate node* yaitu *node* yang merupakan kondisi (ada 2 atau lebih *edge* akan keluar *node* ini).

2. *Black-Box Testing*

Black-box testing atau *behavioral testing* berfokus pada persyaratan fungsional perangkat lunak [PRE-01:459]. Dengan demikian, pengujian *black-box* memungkinkan perekayasa perangkat lunak mendapatkan serangkaian kondisi *input* yang sepenuhnya menggunakan semua persyaratan fungsional untuk semua program. Pengujian *black-box* bukan merupakan alternatif dari teknik *white-box*, tetapi merupakan pendekatan komplementer yang kemungkinan besar mampu mengungkap kelas kesalahan daripada metode *white-box*.

Pengujian *black-box* berusaha menemukan kesalahan dalam kategori berikut :

1. Fungsi-fungsi yang tidak benar atau hilang
2. Kesalahan *interface*
3. Kesalahan dalam struktur data atau akses *database* eksternal
4. Kesalahan kinerja
5. Inisialisasi dan kesalahan terminasi.

Tidak seperti pengujian *white-box*, yang dilakukan pada saat awal proses pengujian, pengujian *black-box* cenderung diaplikasikan selama tahap akhir pengujian. Karena pengujian *black-box* memperhatikan struktur kontrol, maka perhatian berfokus pada domain informasi. Pengujian didesain untuk menjawab pertanyaan-pertanyaan berikut :

- Bagaimana validitas fungsional diuji ?
- Kelas input apa yang akan membuat *test case* menjadi baik ?
- Apakah sistem sangat sensitif terhadap harga input tertentu ?
- Bagaimana batasan dari suatu data diisolasi ?
- Kecepatan dan volume data apa yang dapat ditolerir oleh sistem ?
- Apa pengaruh kombinasi tertentu dari data terhadap operasi sistem ?

3. Strategi Pengujian

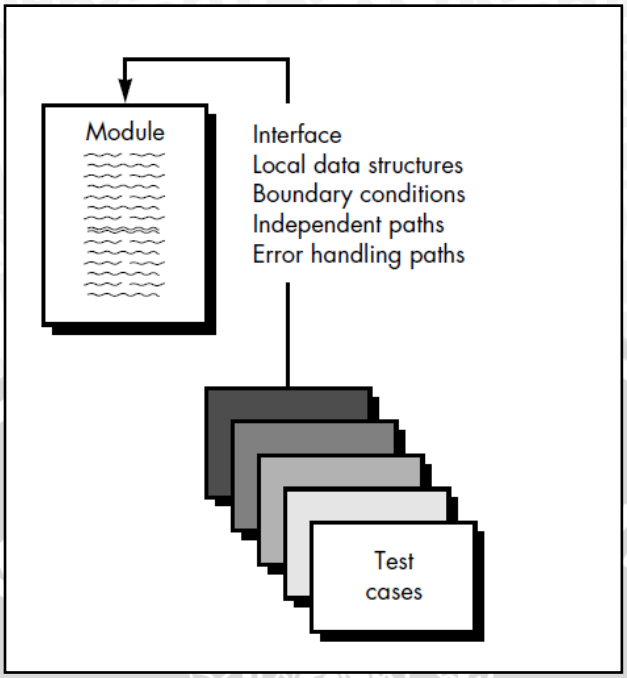
Strategi untuk pengujian perangkat lunak mengintegrasikan metode desain *test case* perangkat lunak ke dalam sederetan langkah yang direncanakan dengan

baik, dan hasilnya adalah konstruksi perangkat lunak yang berhasil [PRE-01:477]. Sejumlah strategi pengujian perangkat lunak telah diusulkan di dalam literatur. Strategi pengujian harus mengakomodasi pengujian tingkat rendah yang diperlukan untuk membuktikan bahwa segmen kode sumber yang kecil telah diimplementasikan dengan tepat, demikian juga pengujian tingkat tinggi yang memvalidasi fungsi-fungsi sistem mayor yang berlawanan dengan kebutuhan pelanggan. Proses pengujian dimulai dengan pengujian yang berfokus pada setiap modul secara individual (*unit testing*), dilanjutkan dengan pengujian integrasi (*integration testing*) dan berakhir pada pengujian validasi (*validation testing*) [PRE-01:481].

a. Pengujian Unit

Pengujian unit berfokus pada usaha verifikasi pada inti terkecil dari desain perangkat lunak, yakni modul. Dengan menggunakan gambaran desain prosedural sebagai panduan, jalur kontrol yang penting diuji untuk mengungkap kesalahan di dalam batas modul tersebut. Kompleksitas relatif dari pengujian dan kesalahan yang diungkap dibatasi oleh ruang lingkup batasan yag dibangun untuk pengujian unit. Pengujian unit biasanya berorientasi pada *white-box*, dan langkahnya dapat dilakukan secara paralel untuk model bertingkat [PRE-01:485].

Pengujian yang terjadi sebagai bagian dari inti digambarkan secara skematis pada Gambar 2.13*Interface* modul diuji untuk memastikan bahwa informasi secara tepat mengalir masuk dan keluar dari inti program yang diuji. Struktur data lokal diuji untuk memastikan bahwa data yang tersimpan secara temporal dapat tetap menjaga integritasnya selama semua langkah di dalam suatu algoritma dieksekusi. Kondisi batas diuji untuk memastikan bahwa modul beroperasi dengan tepat pada batas yang ditentukan untuk membatasi pemrosesan. Semua jalur independen (jalur dasar) yang melalui struktur kontrol dipakai sedikitnya satu kali. Dan akhirnya, penanganan kesalahan uji [PRE-01:485].



Gambar 2.13 Pengujian unit

Sumber : [PRE-01:487]

b. Pengujian Validasi

Pada kulminasi pengujian terintegrasi, perangkat lunak secara lengkap dirakit sebagai suatu paket; kesalahan *interfacing* telah diungkap dan dikoreksi, dan seri akhir dari pengujian perangkat lunak, yaitu pengujian validasi dapat dimulai. Validasi dapat ditentukan dengan berbagai cara, tetapi definisi yang sederhana adalah bahwa validasi berhasil bila perangkat lunak berfungsi dengan cara yang dapat diharapkan secara bertanggung jawab oleh pelanggan. Validasi perangkat lunak dicapai melalui sederetan pengujian *black-box* yang memperlihatkan konformitas dengan persyaratan. Rencana pengujian menguraikan kelas-kelas pengujian yang akan dilakukan, dan prosedur pengujian menentukan *test case* spesifik yang akan digunakan untuk mengungkap kesalahan dalam konformitas dengan persyaratan. Baik rencana dan prosedur didesain untuk memastikan apakah semua persyaratan fungsional dipenuhi; semua persyaratan kinerja dicapai; dokumentasi benar dan direkayasa oleh manusia: dan persyaratan lainnya dipenuhi (transportabilitas, kompatibilitas, pembetulan kesalahan, maintanabilitas) [PRE-01:495].

c. Pengujian Performa

Setelah semua langkah pengujian perangkat lunak secara terstruktur dilakukan, maka perlu dilakukan pengujian sistem di lingkungan dimana dia bekerja untuk mengetahui performa dari perangkat lunak tersebut. Pengujian sistem dirancang untuk menguji kinerja *run-time*dari perangkat lunak dalam konteks sistem terintegrasi. Pengujian performa melibatkan *monitoring* pemanfaatan sumber daya dari perangkat lunak yang diuji seperti perangkat lunak pendukung dan perangkat keras. Pengujian performa dilakukan secara spesifik sesuai dengan tipe perangkat lunak yang diuji. Pengujian performa bertujuan untuk mengungkap situasi yang menyebabkan degradasi dan kemungkinan kegagalan sistem [PRE-01:498].

2.9 *Web Service*

Menurut W3C [W3C-11] *web service* adalah suatu sistem perangkat lunak yang didisain untuk mendukung interaksi mesin ke mesin pada suatu jaringan. *Web service* mempunyai suatu interface yang diuraikan dalam suatu format *machine-processible* seperti WSDL. Sistem lain yang berinteraksi dengan *web service* dilakukan melalui *interface* atau antar muka menggunakan pesan seperti pada SOAP. Pada umumnya pesan ini melalui HTTP dan XML yang merupakan salah satu standard *web*. Perangkat lunak aplikasi yang ditulis dalam berbagai bahasa pemrograman dan berjalan pada berbagai *platform* dapat menggunakan *web service* untuk pertukaran data pada jaringan komputer seperti *Internet*. Sebagai contoh adalah antara Java dan Python atau Microsoft Windows dan aplikasi Linux.

Menurut Michael C. Daconta [DAC-05], w*eb service* adalah aplikasi perangkat lunak yang dapat ditemukan, diuraikan, dan diakses berdasarkan pada XML atau biasa disebut *Application Programming Interface*(API) dan protokol *standard web* pada *intranet*, *extranet*, dan *Internet*. Pada skripsi ini, metode pertukaran data yang dipakai adalah JSON (*JavaScript Object Notation*).

2.9.1 JSON (*JavaScript Object Notation*)

JSON (*JavaScript Object Notation*) adalah format pertukaran data yang

ringan, mudah dibaca, mudah diterjemahkan dan dibuat (*generate*) oleh komputer. Format ini dibuat berdasarkan bagian dari bahasa pemprograman JavaScript. JSON merupakan format teks yang tidak bergantung pada bahasa pemprograman apapun. JSONditulis menggunakan gaya bahasa yang umum digunakan oleh programmer keluarga C termasuk C, C++, C#, Java, JavaScript, Perl dan Python. JSON sangat ideal sebagai bahasa pertukaran-data [JSO-11].

JSON terbuat dari dua struktur [JSO-11] :

1. Kumpulan pasangan nama atau nilai. Pada beberapa bahasa, hal ini dinyatakan sebagai objek (*object*), rekaman (*record*), struktur (*struct*), kamus (*dictionary*), tabel hash (*hash table*), daftar berkunci (*keyed list*), atau *associative array*.
2. Daftar nilai terurutkan (*an ordered list of values*). Pada kebanyakan bahasa, hal ini dinyatakan sebagai larik (*array*), vektor (*vector*), daftar (*list*), atau urutan (*sequence*).

Penggunaan JSON secara umum terdiri dari fungsi *encode* dan *decode*. Fungsi *encode* untuk mengubah data menjadi JSON *array* dan fungsi *decode* untuk mengubah JSON *array* menjadi suatu nilai kembalian. Contoh penggunaan sintaks fungsi *encode* pada metode JSON dapat dilihat pada Gambar 2.14.

```
<?php
$arr =array('a' => 1, 'b' => 2, 'c' => 3, 'd' => 4, 'e' => 5);
echo json_encode($arr);
?>
```

Gambar 2.14 Sintaks JSON *encode*

Sumber : [PHP-11]

Method`json_encode($arr)` akan mengubah nilai dari variabel `$arr` menjadi suatu format yang dapat dibaca sebagai JSON *array*. Hasil keluaran dari sintaks tersebut dapat dilihat pada Gambar 2.15.

```
{"a":1,"b":2,"c":3,"d":4,"e":5}
```

Gambar 2.15 Hasil JSON *encode*

Sumber : [PHP-11]

JSON *decode* adalah proses memperoleh nilai dari suatu variable dengan format JSON *array*. Sintaks JSON pada proses *decode* dapat dilihat pada Gambar 2.16.

```
<?php
$json = '{"foo-bar": 12345}';
$obj = json_decode($json);
print $obj->{'foo-bar'};
```

Gambar 2.16 Sintaks JSON *decode*

Sumber : [PHP-11]

Pada sintaks di atas, *method*`json_decode($json)` akan mengekstrak nilai dari variabel JSON *array*`$json` menjadi suatu nilai *output*. Hasil keluaran dari proses *decode*tersebut dapat dilihat pada Gambar 2.17.

```
12345
```

Gambar 2.17 Hasil JSON *decode*

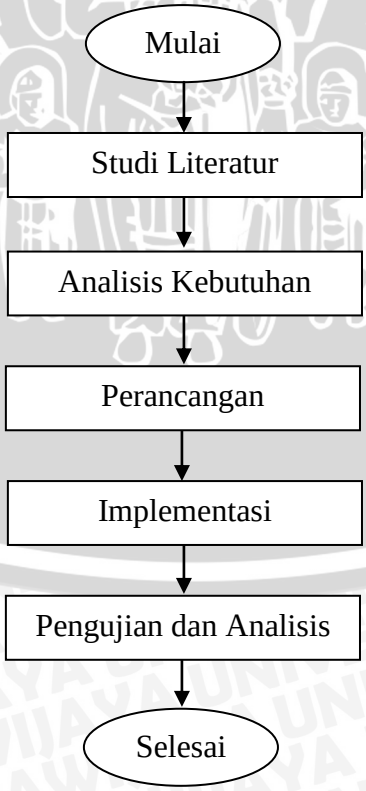
Sumber : [PHP-11]

BAB III
METODOLOGI PENELITIAN

Pada bab ini dijelaskan langkah-langkah yang akan dilakukan dalam perancangan, implementasi dan pengujian dari aplikasi rancang bangun aplikasi informasi gedung Universitas Brawijaya pada smartphone Android.Metodologi penelitian yang dilakukan melibatkan beberapa langkah berikut:

1. Studi Literatur
2. Analisis Kebutuhan
3. Perancanganan
4. Implementasi
5. Pengujian dan analisis

Keterkaitan antar langkah-langkah tersebut dapat dilihat pada gambar 3.1. Setiap langkah ini akan dijelaskan pada sub bab berikutnya setelah deskripsi singkat tentang penggunaan aplikasi informasi gedung Universitas Brawijaya pada smartphone Android.



Gambar 3.1 Diagram Alir Metodologi Penelitian.

3.1 Deskripsi Aplikasi

Aplikasi *augmented reality* informasi gedungUniversitas Brawijaya merupakan aplikasi yang mengolah informasi yang sebelumnya belum digunakan secara maksimal menjadi informasi yang memiliki nilai lebih. Informasi utama yang diolah dalam aplikasi ini adalah letak gedung berupa kordinat. Aplikasi ini menunjukkan letak gedung-gedung yang ada di Universitas Brawijaya melalui *mobile device (smartphone* android). Aplikasi ini memanfaatkan fungsi kamera untuk mendeteksi gedung-gedung dan menggunakan fungsi gps untuk mendeteksi koordinat dari gedung-gedung yang ada di Universitas Brawijaya. Informasi ditampilkan menggunakan teknologi *Augmented Reality (*AR*).*

Aplikasi ini dibuat untuk dapat menghasilkan informasi yang mudah, akurat dan cepat. Kemudahan didapatkan dari *device*yang digunakan yaitu *mobile device*dalam hal ini adalah *smartphone* Android. Banyak orang membawa perangkat ini,dimanapun dan kapanpun. Kecepatan informasi didapatkan dari penggunaan AR sebagai teknologi untuk menampilkan informasi. AR menampilakn informasi secara *realtime.*AR yang digunakan adalah*markerless*AR yang dalam bahasa indonesia diartikan menjadi AR tanpa marker. Hal ini bukan berarti AR tidak memakai *marker*namun menggunakan *marker* yang tak terlihat yang yaitu koordinat lokasi. Pembacaan koordinat lokasi dilakukan saat aplikasi dijalankan dan menghasilkan informasi yang bersifat realtime. Keakuratan data yang ditampilkan bergantung pada proses pengumpulan data. Data yang dikumpulkan akan disimpan di *database* yang diletakkan di *server*.

3.2 Studi Literatur

Studi literatur menjelaskan dasar teori yang digunakan untuk menunjang penulisan skripsi. Teori-teori pendukung tersebut meliputi:

a. *Augmented reality* (AR)

- *AR Engine* (Mixare)

  Mempelajari *mixare* sebagai *engine* untuk menampilkan *markerless* AR pada sistem operasi Android.

b. *Component-Based Software Engineering*

1. *Domain Engineering :*
   - i. *Domain Analysis*
   - ii. *Domain Model*
   - iii. *Software Architecture Development*

iv. *Structural Model*
v. *Reusable Component Development*
vi. *Repository Reusable Artifacts / Components*

2. *Component-Based Development*
   i. *Analysis*
   ii. *Architectural Design*
   iii. *Component Qualification*
   iv. *Component Adaptation*
   v. *Component Engineering*
   vi. *Component Composition*
   vii. *Testing*
   viii. *Application Software*
   ix. *Component Update*

c. Sistem Operasi Android

- Fitur Android

  Bagian ini menjelaskan fitur-fitur yang ada pada sistem operasi Android sehinga bisa dijadikan referensi untuk membangun aplikasi.

- Arsitektur Android

  Bagian ini menjelaskan arsitektur dari sistem operasi Android sehingga bisa dijadikan referensi untuk membangun aplikasi.

d. UML

- *Use Case*

  Bagian ini digunakan sebagai acuan untuk membuat diagram yang menggambarkan interaksi aktor dengan sistem.

- *Class Diagram*

  Bagian ini digunakan sebagai acuan untuk membuat diagram yang menggambarkan struktur dan deskripsi *class, package, dan* objek beserta hubungan satu sama lain.

- *Deployment Diagram*

  Bagian ini digunakan sebagai acuan untuk membuat diagram yang menggambarkan detail bagaimana komponen di-*deploy*.

- *Sequence Diagram*

  Bagian ini digunakan sebagai acuan untuk membuat diagram yang menggambarkan interaksi antar objek di dalam dan sekitar sistem berupa *message* yang digambarkan terhadap waktu.

e. Pengujian Perangkat Lunak

1. Teknik Pengujian
   - *i. White-Box Testing*
   - *ii. Black-Box Testing*
2. Strategi Pengujian
   - i. Pengujian Unit
   - ii. Pengujian Integrasi
   - iii. Pengujian Validasi
   - iv. Pengujian Performa

3.3 Analisis Kebutuhan

Analisis kebutuhan bertujuan untuk mendapatkan semua kebutuhan yang diperlukan dari aplikasi yang akan dibangun. Aplikasi ini dalam pelaksanaan implementasinya memerlukan Kebutuhan. Analisis diperlukan untuk dapat membangun aplikasi yang terstruktur.

Adapun kebutuhan aplikasi informasi gedung Universitas Brawijaya berbasis *smartphone* Android dibagi dalam beberapa bagian, yaitu:

1. Deskripsi Umum Perangkat Lunak

2. Kebutuhan fungsional

3. Kebutuhan Non Fungsional

4. Diagram *Use Case*

3.4 Perancangan Sistem

Perancangan aplikasi dilakukan setelah semua kebutuhan sistem didapatkan melalui tahap analisis kebutuhan. Perancangan aplikasi berdasarkan

Obeject OrientedAnalysis dan *Object Oriented Design* menggunakan pemodelan UML (*Unified Modeling Language*). Perancangan subsistemaplikasidilakukan dengan mengidentifikasi kelas-kelas dan *interface-interface* yang dibutuhkan yang dimodelkan dalam *class diagram.* Hubungan interaksi antar elemen (objek) yang telah diidentifikasi dimodelkan dalam *Sequence Diagram* yang menggambarkan interaksi antar objek yang disusun dalam urutan waktu.

3.5 Implementasi

Implementasi aplikasi dilakukan dengan mengacu kepada perancangan aplikasi. Implementasi perangkat lunak dilakukan dengan menggunakan bahasa pemrograman objek yaitu menggunakan bahasa pemrograman Java dengan menggunakan library Android dan library untuk *AR.*

3.6 Pengujian dan Analisis

Pengujian perangkat lunak pada skripsi ini dilakukan agar dapat menunjukkan bahwa perangkat lunak telah mampu bekerja sesuai dengan spesifikasi dari kebutuhan yang melandasinya. Pengujian dilakukan menggunakan metode *whitebox testing* dan *blackbox testing.*

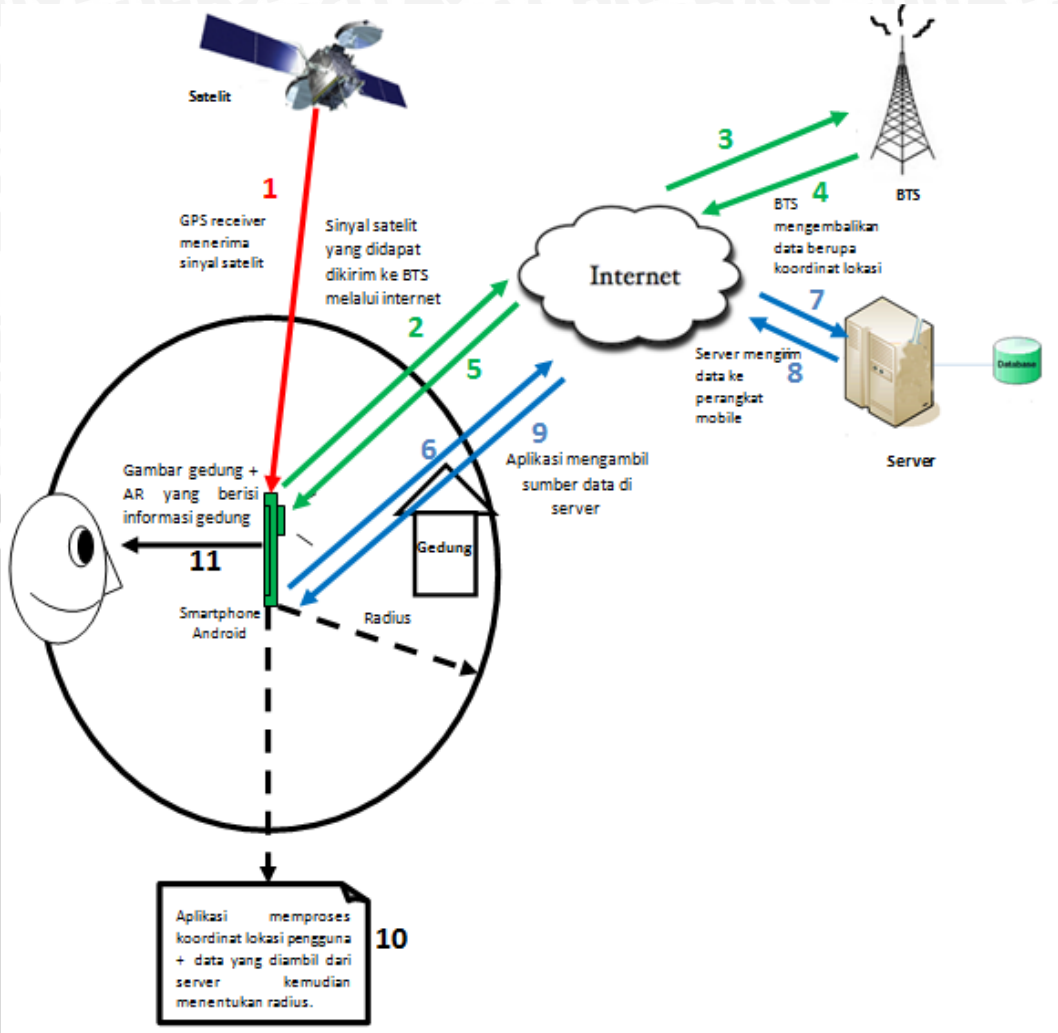
Proses pengujian perangkat lunak pada skripsi ini terdapat tiga tahap (strategi) yaitu pengujian unit, pengujian validasi, dan pengujian performa. Pada pengujian unit digunakan teknik *White-Box Testing* dengan teknik *Basis Path Testing*. Pada teknik *Basis Path Testing,* proses pengujian dilakukan dengan memodelkan algoritma pada suatu *flow graph,* menentukan jumlah kompleksitas siklomatis *(cyclomatic complexity),* menentukan sebuah basis set dari jalur independen dan memberikan kasus uji *(test case)* pada setiap basis set yang telah ditentukan. Pengujian validasi menggunakan metode pengujian *Black Box,* karena tidak diperlukan konsentrasi terhadap alur jalannya algoritma program dan lebih ditekankan untuk menemukan konformitas antara fungsi sistem dengan daftar kebutuhan. Pada pengujian performa dilakukan pengujian menggunakan *tool* yang disebut apache *benchmark* untuk mengetahui performa *server*.

BAB IV
PERANCANGAN

Pada bab ini dibahas mengenai perancangan aplikasiinformasi gedungUniversitas Brawijaya (UB) pada *smartphone* android. Perancangan dilakukan dengan tiga tahap. Pada tahap pertama dilakukan perancangan umum sistem, tahap dua dilakukan analisis kebutuhan dan pada tahap keketiga dilakukan perancangan perangkat lunak. Pada tahap perancangan umum sistem dijelaskan gambaran umum tentang aplikasi informasi gedung UB pada *smartphone* android. Pada tahap analisis kebutuhan terdiri dari lima langkah yaitu kebutuhan deskripsi umum perangkat lunak, deskripsi kebutuhan fungsional, kebutuhan non fungsional, diagram *use case*. Tahap perancangan perangkat lunak memiliki lima langkah, yaitu perancangan arsitektural, perancangan basis data, pemodelan diagram *class* untuk menggambarkan perancangan struktur *class – class* yang menyusun perangkat lunak Informasi Gedung Universitas Brawijaya pada *Smartphone* Android, pemodelan diagram *sequence* untuk menggambarkan interaksi antar objek atau *class* di dalam perangkat lunak Informasi Gedung UB pada *Smartphone* Android, dan perancangan antarmuka pengguna dari perangkat lunak Informasi Gedung UB pada *Smartphone* Android.

4.1. Perancangan Sistem

Perancangan sistem merupakan tahap awal dari perancangan perangkat lunak. Perancangan sistem dilakukan untuk merepresentasikan arsitektur sistem yang akan dibuat secara umum. Arsitektur sistem yang akan dirancangan ditunjukkan dalam Gambar 4.1.



Gambar 4.1 Perancangan sistem

Sumber : [Perancangan]

Keterangan Gambar 4.1 :

1. *Smartphone* menerima sinyal dari satelit memanfaakan fungsi dari GPS receiver yang ada pada smartphone. Untuk GPS, dibutuhkan 3-4 satelit untuk mendapatkan koordinat pengguna.
2. Smartphone memanfaatkan fungsi A-GPS, setelah menerima sinyal dari 1 satelit maka akan diteruskan ke BTS melalui media *internet.*
3. BTS menerima sinyal yang telah dikirim A-GPS.
4. BTS mengirimkan data ke *smartphone* berupa data posisi.
5. *Smartphone* menerima data posisi psengguna.
6. Aplikasi meminta *request* ke *server* yang di-*hosting* pada IP *public* melalui media *internet.*

7. *Request* diteruskan ke *server*.
8. *Server* memberikan respon dengan mengirimkan data dengan memanfaatkan *web service* (JSON)
9. Smartphone menerima data informasi gedung dari *server*.
10. Aplikasi memproses koordinat pengguna dan menggunakannya sebagai titik tengah radius wilayah jangkauan aplikasi.
11. Data koordinat yang telah diterima dari *server* dan masuk dalam radius yang telah dibuat pada nomor 10, akan ditampilkan ke layar *smartphone* dengan media *augmented reality*.

4.2. Analisis Kebutuhan

Analisis kebutuhan mengacu pada gambaran perangkat lunak Informasi Gedung UB dan identifikasi kebutuhan – kebutuhan yang ingin didapatkan oleh pengguna. Pada tahap ini terdiri dari empat langkah yaitu deskripsi umum perangkat lunak, Kebutuhan fungsional, kebutuhan non fungsional, diagram *usecase*.

4.2.1. Deskripsi Umum Perangkat Lunak

1. Deskripsi Umum Sistem

Aplikasiinformasi gedungUB merupakan aplikasi yang mengolah informasi yang sebelumnya belum digunakan secara maksimal menjadi informasi yang memiliki nilai lebih. Informasi utama yang diolah dalam aplikasi ini adalah letak gedung berupa kordinat. Aplikasi ini menunjukkan letak gedung-gedung yang ada di UB melalui *mobile device (smartphone* android). Aplikasi ini memanfaatkan fungsi kamera untuk mendeteksi gedung-gedung dan menggunakan fungsi gps untuk mendeteksi koordinat dari gedung-gedung yang ada di UB. Informasi ditampilkan menggunakan teknologi *Augmented Reality (*AR*)*.

Aplikasi ini dibuat untuk dapat menghasilkan informasi yang mudah, akurat dan cepat. Kemudahan didapatkan dari *device* yang digunakan yaitu *mobile device* dalam hal ini adalah *smartphone* Android. Banyak orang membawa perangkat ini, dimanapun dan kapanpun. Kecepatan informasi didapatkan dari penggunaan AR sebagai teknologi untuk menampilkan informasi. AR

menampilakn informasi secara *realtime.* AR yang digunakan adalah *markerless* AR yang dalam bahasa indonesia diartikan menjadi AR tanpa marker. Hal ini bukan berarti AR tidak memakai *marker* namun menggunakan *marker* yang tak terlihat yang yaitu koordinat lokasi. Pembacaan koordinat lokasi dilakukan saat aplikasi dijalankan dan menghasilkan informasi yang bersifat realtime. Keakuratan data yang ditampilkan bergantung pada proses pengumpulan data. Data yang dikumpulkan akan disimpan di *database* yang diletakkan di *server*.

2. Fungsi Produk

Aplikasi Informasi Gedung Universitas Brawijaya memiliki fungsi utama yaitu :

a. Melihat informasi gedung UB.
b. Mencari lokasi gedung yang ada di UB.
c. Melihat daftar gedung yang ada di UB.

3. Identifikasi Aktor

Pada perangcangan Rancang Bangunaplikasi *augmented reality* informasi gedunguniversitas brawijaya pada *smartphone* Android, akan terdapat dua tingkatan pengguna aplikasi, yaitu:

Tabel 4.1. Identifikasi Aktor.

Aktor	Deskripsi
Pengguna	Pengguna adalah subyek yang akan menggunakan aplikasi informasi gedunguniversitas brawijaya pada *smartphone* android untuk mengetahui informasi gedung di Universitas Brawijaya.
Admin	Admin adalah subyek yang melakukan penambahan, penghapusan, dan perubahan basis data yang berada di server yang mengolah informasi yang ditampilkan oleh aplikasi informasi gedungUniversitas Brawijaya pada *smartphone* Android.

Sumber : Perancangan

4. Lingkungan Operasi

Aplikasi Informasi Gedung UB berjalan pada *mobile device* dengan sistem operasi Android.

4.2.2. Kebutuhan Fungsional

Tabel 4.2. Spesifikasi kebutuhan Fungsional Pengguna.

SRS No	Kebutuhan	*Use Case*
SRS-C001	Perangkat lunak harus dapat menampilkan informasi gedung dengan AR sebagai tampilan dalam *mobile device.*	Melihat Informasi Gedung
SRS-C002	Perangkat Lunak harus dapat mencari informasi gedung dengan AR sebagai tampilan dalam *mobile device.*	Mencari Informasi Gedung
SRC-C003	Perangkat lunak harus dapat menampilkan informasi detail tentang gedung yang dipilih.	Melihat Detail Informasi Gedung
SRC-C004	Perangkat lunak harus dapat menampilkan halaman *about*	Melihat *About*

Sumber : Perancangan

Tabel 4.3. Spesifikasi Kebutuhan Fungsional Admin.

SRS No	Kebutuhan	*Use Case*
SRS-A001	Perangkat lunak dapat melakukan seleksi pengguna.	*Login*
SRS-A002	Perangkat lunak harus dapat menambahkan informasi yang di *server*.	Menambah Informasi di *database*
SRS-A003	Perangkat lunak harus dapat menghapus informasi yang di *server*.	Menghapus Informasi di *database*
SRS-A004	Perangkat lunak harus dapat mengubah informasi yang di *server*.	Mengubah Informasi di *database*
SRS-A005	Perangkat lunak dapat menghapus *session* dari pengguna.	*Logout*

Sumber : Perancangan

4.2.3. Kebutuhan Non Fungsional

Tabel 4.4. Spesifikasi Kebutuhan Non Fungsional.

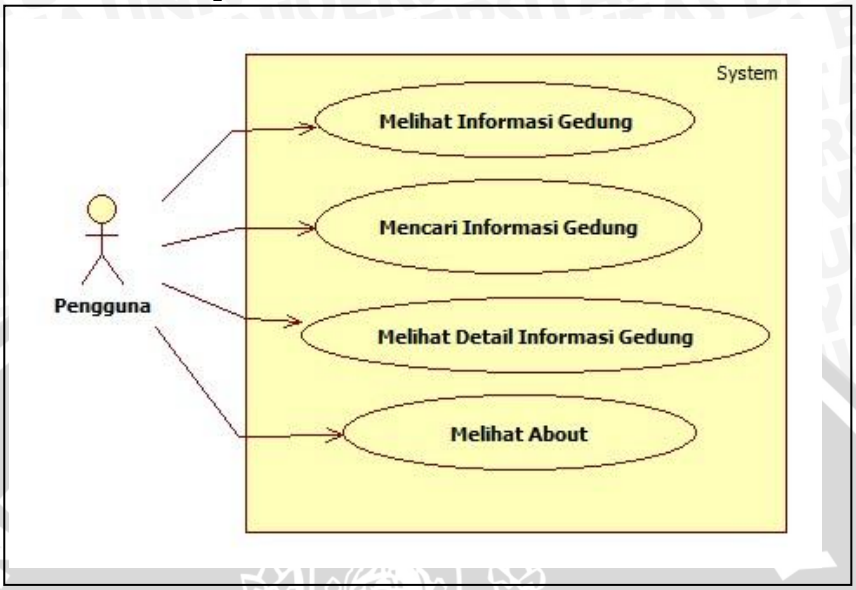
Parameter	Deskripsi kebutuhan
Availability	Perangkat lunak harus dapat beroperasi terus - menerus selama waktu yang diinginkan (24 jam atau bahkan melebihi jam
Response time	Perangkat lunak harus dapat berjalan dengan waktu proses tidak lebih dari 10 detik dengan *bandwith* minimal 64 *kbps*.
Control	Membatasi akses penggunaan perangkat lunak dengan meyediakan 2 sisi aplikasi, yaitu *client* dan *administrator*.
Memory	Aplikasi ini harus ringan dan tidak membutuhkan memory tinggi sehingga aplikasi ini dapat dijalankan pada smartphone android dengan spesifikasi rendah minimal RAM 128 MB.

Sumber : Perancangan

4.2.4. Diagram Use Case

Use case merupakan suatu fungsionalitas yang menggambarkan apa yang diharapkan dari sebuah sistem. *Use case* mempresentasikan sebuah interaksi antara aktor dengan sistem. *Use case* merupakan sebuah pekerjaan yang akan dilakukan oleh aktor terhadap sistem tersebut. *Use case* sangat membantu ketika sedang menyusun *requirement* sebuah sistem, mengkomunikasikan rancangan denganklien, dan merancang *test case* untuk semua *feature* yang ada pada sistem.

1. **Diagram *Use Case* Aplikasi *Client***



Gambar 4.2 Diagram *use case* pengguna.

Sumber : Perancangan

a. Skenario *Use Case* Mencari Informasi Gedung

Tabel 4.5. *Use Case* Mencari Informasi.

Skenario Kasus Pada Sistem	
Nomor *Use Case*	SRS C-002
Nama	Mencari Informasi Gedung
Tujuan	Untuk mencari informasi gedung UB
Deskripsi	*Use Case* menjelaskan bagaimana pengguna mencari informasi tentang gedung UB.
Aktor	Pengguna
Skenario Utama	
Kondisi Awal	Perangkat lunak informasi gedung UB sudah berjalan dan pengguna memilih menu “cari” kemudian perangkat lunak mengakses fungsi kamera dari *device*.
Aksi Aktor	Reaksi Sistem
Pengguna mengarahkan *device* ke	1. Perangkat lunak mengunduh

gedung yang akan diinginkan.	datasource di *server*. 2. Data yang diunduh dari *server* digunakan sebagai masukan yang selanjutnya diproses oleh AR *engine*
Skenario Alternatif 1 : Data gedung yang diinginkan tidak ada.	
	3. AR engine tidak dapat menampilkan AR yang berisi informasi gedung.
Kondisi Akhir	Objek AR akan muncul dalam layar yang beisi informasi gedung tersebut

Sumber : Perancangan

b. Skenario *Use Case* Melihat Informasi Gedung

Tabel 4.6. *Use Case* Melihat Informasi.

Skenario Kasus Pada Sistem	
Nomor *Use Case*	SRS C-001
Nama	Melihat Informasi Gedung
Tujuan	Untuk mendapatkan informasi gedung UB
Deskripsi	*Use Case* menjelaskan bagaimana pengguna mendapatkan informasi tentang gedung UB.
Aktor	Pengguna
Skenario Utama	
Kondisi Awal	Perangkat lunak informasi gedung UB sudah berjalan dan mengakses fungsi kamera dari *device*.
Aksi Aktor	Reaksi Sistem
Pengguna mengarahkan *device* ke gedung yang akan diinginkan.	1. Perangkat lunak mengunduh datasource di *server*. 2. Data yang diunduh dari *server* digunakan sebagai masukan yang

	selanjutnya diproses oleh AR *engine*
Skenario Alternatif 1 : Data gedung yang diinginkan tidak ada.	
	1. AR engine tidak dapat menampilkan AR yang berisi informasi gedung.
Kondisi Akhir	Objek AR akan muncul dalam layar yang beisi informasi gedung tersebut

Sumber : Perancangan

c. Skenario *Use Case* Melihat Detail Informasi Gedung

Tabel 4.7. *Use Case* Melihat Detail Informasi.

Skenario Kasus Pada Sistem	
Nomor *Use Case*	SRS C-003
Nama	Melihat Detail Informasi Gedung
Tujuan	Untuk mendapatkan informasi gedung UB secara detail
Deskripsi	*Use Case* menjelaskan bagaimana pengguna mendapatkan informasi tentang gedung UB secara detail.
Aktor	Pengguna
Skenario Utama	
Kondisi Awal	Perangkat lunak informasi gedung UB sudah berjalan dan mengakses fungsi kamera dari *device* dan AR muncul pada layar *device.*
Aksi Aktor	Reaksi Sistem
1. Pengguna menekan tombol menu pada *device* dan memilih menu detail. 2. Jika *list* tempat telah ditampilkan, pengguna menekan salah satu *list*yang ingin diketahui informasi	3. Perangkat lunak mengunduh datasource di *server*danmenampilakan dalam bentuk *list.* 4. Data yang diunduh dari *server* digunakan sebagai masukan yang

detailnya.	selanjutnya diproses oleh perangkat lunak kemudian ditampilkan dengan *message box*.
Skenario Alternatif 1 : Detail Informasi yang diinginkan tidak tersedia.	
	1. *Message box* tidak muncul.
Kondisi Akhir	*Message box* yang berisi detail info gedung akan muncul paa layar.

Sumber : Perancangan

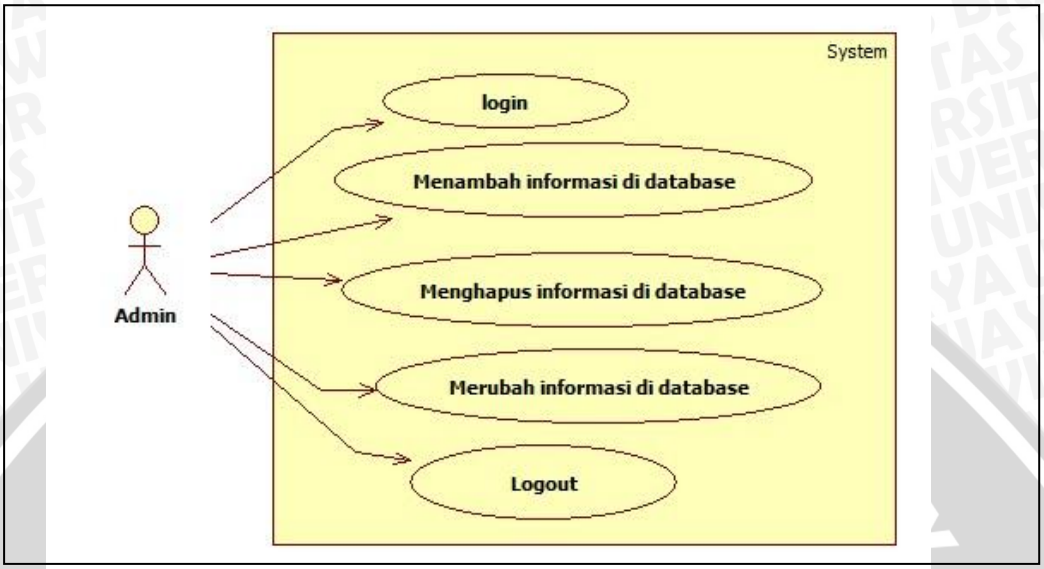
d. Skenario *Use Case* Melihat *About*

Tabel 4.8. *Use Case* Melihat Halaman *About*.

Skenario Kasus Pada Sistem	
Nomor *Use Case*	SRS C-004
Nama	Melihat Halaman About
Tujuan	Untuk mendapatkan informasi tentang aplikasi informasi gedung UB.
Deskripsi	*Use Case* menjelaskan bagaimana pengguna mendapatkan informasi tentang aplikasi informasi gedung UB.
Aktor	Pengguna
Skenario Utama	
Kondisi Awal	Perangkat lunak informasi gedung UB sudah berjalan dan menampilkan halaman menu.
Aksi Aktor	Reaksi Sistem
Pengguna memilih menu “about”.	1. Perangkat lunak memunculkan halaman about.
Kondisi Akhir	Halaman *about* berhasil ditampilkan

Sumber : Perancangan

2. Diagram *Use Case* Aplikasi *Administrator*



Gambar 4.3 Diagram *use case* admin.

Sumber : Perancangan

a. Skenario *Use Case Login*

Tabel 4.9. *Use CaseLogin*.

Skenario Kasus Pada Sistem	
Nomor *Use Case*	SRS A-001
Nama	*Login*
Tujuan	Untuk menyeleksi pengguna yang sah.
Deskripsi	*Use case* menjelaskan bagaimana admin melakukan proses *login* sehingga dapat mengakses halaman utama.
Aktor	Admin
Skenario Utama	
Kondisi Awal	Sistem memunculkan halaman *login.*
Aksi Aktor	Reaksi Sistem
Admin memasukkan data yang dibutuhkan untuk login (*username* dan	1. Sistem akan menerima masukan data *login* kemudian sistem

password), kemudian menekan tombol *login*.	melakukan pengecekan terhadap data yang telah dimasukkan oleh admin. Jika data sesuai maka sistem akan menampilkan halaman utama admin.
Skenario Alternatif 1 : Jika *username* atau *password* kosong	
	2. Sistem akan menampilkan pesan peringatan bahwa *username* atau *password.*
Skenario Alternatif 2 : Jika *username* dan *password* salah	
	3. Sistem akan menampilkan pesan peringatan bahwa *username* dan *password* salah.
Kondisi Akhir	Sistem akan menampilkan halaman utama admin.

Sumber : Perancangan

b. Skenario *Use Case* Menambah Informasi di *Database*

Tabel 4.10. *Use Case* Menambah Informasi di *Database*.

Skenario Kasus Pada Sistem	
Nomor *Use Case*	SRS A-002
Nama	Menambah informasi di *database.*
Tujuan	Untuk menambakan data informasi gedung UB.
Deskripsi	*Use Case* ini menjelaskan bagaimana admin menambahkan data informasi tentang gedung UB pada *server*.
Aktor	Admin
Skenario Utama	
Kondisi Awal	Admin berhasil melakukan *login* dan masuk halaman utama admin. Admin menekan tombol tambah dan berhasil masuk halaman tambah data.

Aksi Aktor	Reaksi Sistem
Admin memasukkan data informasi gedung UB (*Latitude ,longitude ,elevation, distance ,tittle, has detail page ,web*) dan menekan tombol save	1. Sitem menerima data informasi gedung UB yang akan ditambahkan ke *server* kemudian sistem melakukan proses penambahan data
Skenario Alternatif 1 : Jika Admin menekan tombol cancel	
	2. Sistem kembali ke halaman utama.
Kondisi Akhir	Data informasi gedung UB berhasil ditambakan ke *server*.

Sumber : Perancangan

c. Skenario *Use Case* MenghapusInformasi di *Database*

Tabel 4.11. *Use Case* Menghapus Informasi di *Database*.

Skenario Kasus Pada Sistem	
Nomor *Use Case*	SRS A-003
Nama	Menghapus informasi di *database.*
Tujuan	Untuk menghapus data informasi gedung UB.
Deskripsi	*Use Case* ini menjelaskan bagaimana admin menghapus data informasi tentang gedung UB pada *server*.
Aktor	Admin
Skenario Utama	
Kondisi Awal	Admin berhasil melakukan *login* dan masuk halaman utama admin.
Aksi Aktor	**Reaksi Sistem**
Admin menekan *link Delete* pada data yang akan dihapus.	1. Sistem menerima data berupa *id* informasi kemudian melakukan proses penghapusan data dengan *id* informasi tersebut.
Kondisi Akhir	Data informasi gedung UB berhasil dithapus dari *server*.

Sumber : Perancangan

d. Skenario *Use Case* Merubah Informasi di *Database*

Tabel 4.12. *Use Case* Merubah Informasi di *Database*.

Skenario Kasus Pada Sistem	
Nomor *Use Case*	SRS A-004
Nama	Merubah Informasi di *database*
Tujuan	Untuk merubah data informasi gedung UB
Deskripsi	*Use Case* ini menjelaskan bagaimana admin merubah data informasi tentang gedung UB pada *server*.
Aktor	Admin
Skenario Utama	
Kondisi Awal	Admin berhasil melakukan *login* dan masuk halaman utama admin. Admin menekan *link* "Edit" kemudian masuk halaman ubah data.
Aksi Aktor	Reaksi Sistem
Admin melakukan perubahan data dengan merubah data informasi gedung UB (*Latitude ,longitude ,elevation, distance ,tittle, has detail page ,web*) sesuai kebtuhan kemudian menekan tombol "Edit"	1. Sistem menerima masukan berupa id Informasi kemudian menampilkan data informasi gedung sesuai dengan masukan id informasi dan melakukan proses perubahan data pada *server*.
Skenario Alternatif 1 : Jika Admin menekan tombol cancel	
	2. Sistem Kembali kehalaman utama
Kondisi Akhir	Data informasi gedung UB berhasil dirubah.

Sumber : Perancangan

e. Skenario *Use Case Logout*

Tabel 4.13. *Use CaseLogout*.

Skenario Kasus Pada Sistem	
Nomor *Use Case*	SRS A-005

Nama	*Logout*
Tujuan	Untuk melakukan *logout*
Deskripsi	*Use Case* menjelaskan bagaimana admin melakukan proses *logout*
Aktor	Admin
Skenario Utama	
Kondisi Awal	Admin berhasil melakukan *login* dan masuk halaman utama admin.
Aksi Aktor	Reaksi Sistem
Admin menekan tombol "Logout"	1. Sistem melakukan penghapusan *session* admin dan masuk halaman login.
Kondisi Akhir	Admin berhasil logout dan tidak dapat kembali kehalaman utama tanpa melakukan *usecase login.*

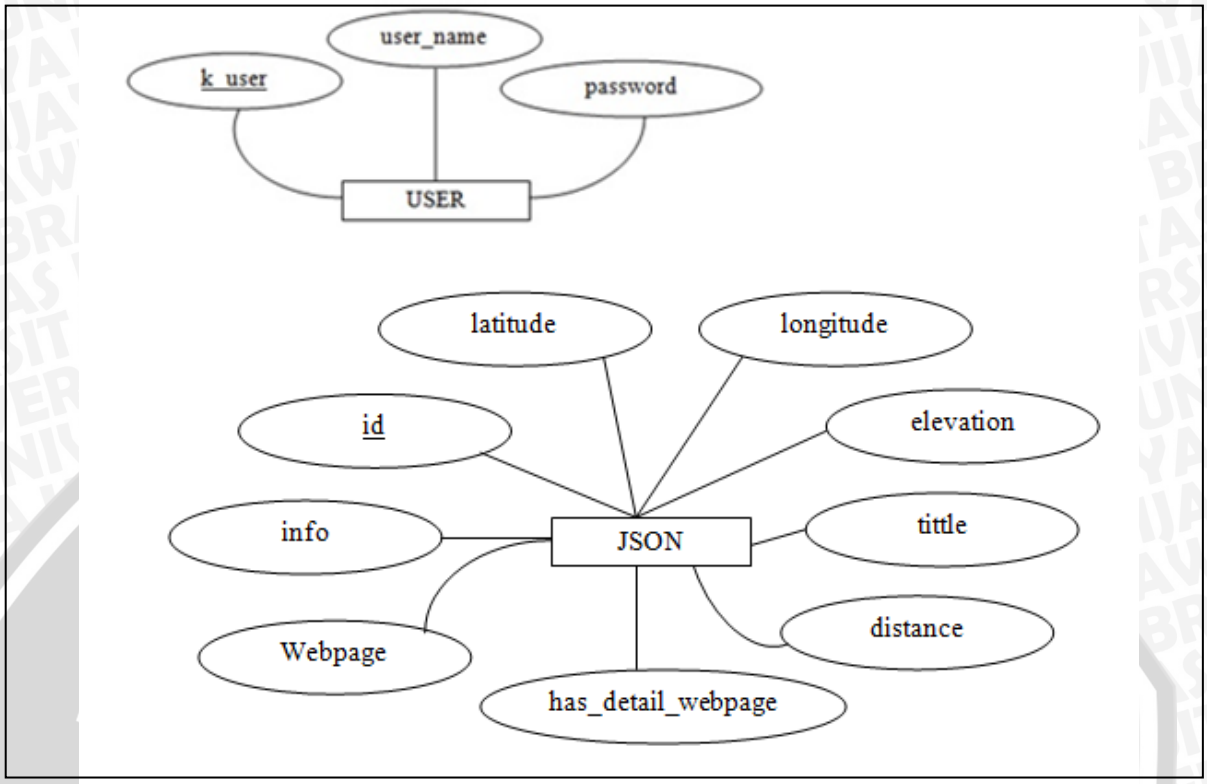
Sumber : Perancangan

4.3. Perancangan Perangkat Lunak

Perancangan perangkat lunak dilakukan dalam empat tahap, yaitu perancangan basis data, diagram *class* untuk menggambarkan perancangan struktur *class – class* yang menyusun Aplikasi Informasi Gedung UB pada *smartphone* Android, diagram *sequence* untuk menggambarkan interaksi antar objek atau *class* di dalam perangkat lunak Aplikasi Informasi Gedung UB pada *smartphone* Android, dan perancangan antarmuka pengguna dari Aplikasi Informasi Gedung UB pada *smartphone* Android. Perancangan perangkat lunak pada skripsi ini menggunakan pendekatan desain berorientasi objek yang direpresentasikan dengan menggunakan UML (*Unified Modelling Language*).

4.3.1. Perancangan Database

Basis data berfungsi sebagai tempat menyimpan data. Perancangan basis data digunakan untuk merancang basis data yang akan dibuat agar masukan dan keluaran program sesuai dengan apa yang diharapkan. Perancangan basis data mengambil acuan dari proses analisis data yang dilakukan pada tahap analisis kebutuhan. Arsitektur basis data yang akan dirancang dijelaskan pada Gambar 4.3



Gambar 4.4 ER Diagram Perangkat Lunak Informasi Gedung UB.

Sumber : Perancangan

a. Tabel Info_Gedung

Nama Tabel : Info_Gedung

Jumlah *field* : 8

Fungsi : Menampung semua informasi tentang gedung

Tabel 4.14. Struktur Tabel Info_Gedung.

Sumber : Perancangan

No.	Nama *Field*	Tipe	Lebar	Keterangan
1	Id	Varchar	20	Id gedung
2	Longitude	Varchar	20	Garis bujur gedung
3	Latitude	Varchar	20	Garis lintang gedung
4	Elevation	Varchar	20	Ketinggian gedung
5	Tittle	Varchar	50	Informasi gedung
6	Distance	Varchar	20	Jarak gedung
7	has_detail_page	Integer	1	Indikator adanya web

8	web_page	Varchar	50	Alamat web gedung

b. Tabel user

Nama Tabel : user

Jumlah *field* : 3

Fungsi : Menampung akun dari admin.

Tabel 4.15. Struktur Tabel user.

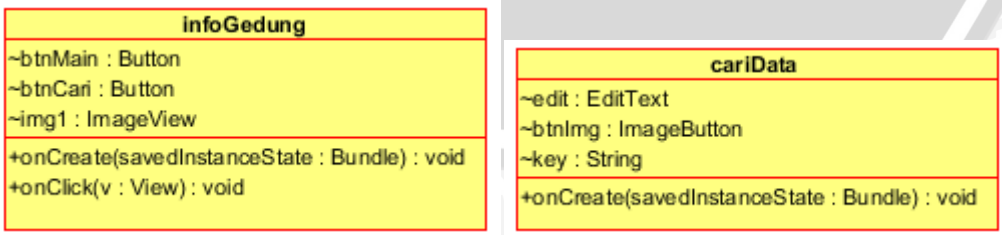
Sumber : Perancangan

No.	Nama *Field*	Tipe	Lebar	Keterangan
1	k_username	Integer	10	Id *username* admin
2	Username	Varchar	45	*Username* admin
3	Password	Varchar	45	*Password* admin

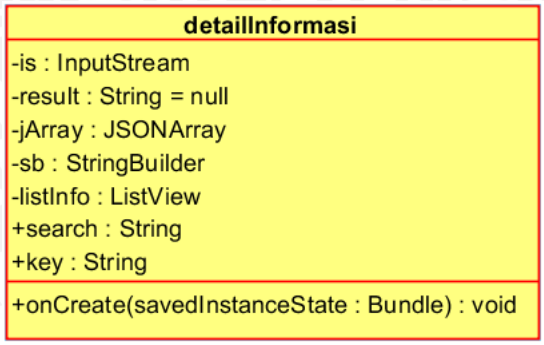
4.3.2. Diagram Class

Class diagram menggambarkan struktur dan deskripsi *class, package* dan objek beserta hubungan satu sama lain seperti *containment*, pewarisan, asosiasi, dan lain-lain yang membentuk sebuah perangkat lunak. *Class* bisa didapatkan dengan menganalisis secara detail terhadap *use case* yang dimodelkan. Gambar 4.4 menunjukkan diagram *class* dari perangkat lunak Informasi Gedung UB. Pada perancangan proses ini terdapat empat elemen *class* pada sisi aplikasi *client* yang dibahasmeliputi *class* infoGedung, *class* cariData, *class* MixView, *class* about. Gambar 4.5 menunjukkan diagram *class* pada *server*. Ada 2 class pada aplikasi server yang akan dibahas meliputi *class* login, *class* operasi.

a. Class Diagram Aplikasi Client




MixView
-camScreen : CameraSurface
-augScreen : AugmentedView
-isInited : boolean
-mixContext : MixContext
~dWindow : PaintScreen
~dataView : DataView
-downloadThread : Thread
-RTmp : float[] = new float[9]
-Rot : float[] = new float[9]
-I : float[] = new float[9]
-grav : float[] = new float[3]
-mag : float[] = new float[3]
-sensorMgr : SensorManager
-sensors : List<Sensor>
-sensorGrav : Sensor
-sensorMag : Sensor
-rHistIdx : int = 0
-tempR : Matrix = new Matrix()
-finalR : Matrix = new Matrix()
-smoothR : Matrix = new Matrix()
-histR : Matrix[] = new Matrix[60]
-m1 : Matrix = new Matrix()
-m2 : Matrix = new Matrix()
-m3 : Matrix = new Matrix()
-m4 : Matrix = new Matrix()
-myZoomBar : SeekBar
-mWakeLock : WakeLock
-fError : boolean
-compassErrorDisplayed : int = 0
-zoomLevel : String
-zoomProgress : int
-searchNotificationTxt : TextView
~search : String
~key : String
+TAG : String = "Mixare"
+PREFS_NAME : String = "MyPrefsFileForMenuItems"
-myZoomBarOnSeekBarChangeListener : OnSeekBarChangeListener = new SeekBar.OnSeekBarChangeListener() {
Toast t;
public void onProgressChanged(SeekBar seekBar, int progress, boolean fromUser) {
float myout = calcZoomLevel();
zoomLevel = String.valueOf(myout);
zoomProgress = myZoomBar.getProgress();
t.setText("Radius: " + String.valueOf(myout));
t.show();
}
public void onStartTrackingTouch(SeekBar seekBar) {
Context ctx = seekBar.getContext();
t = Toast.makeText(ctx, "Radius: ", Toast.LENGTH_LONG);
//zoomChanging= true;
}
public void onStopTrackingTouch(SeekBar seekBar) {
SharedPreferences settings = getSharedPreferences(PREFS_NAME, 0);
SharedPreferences.Editor editor = settings.edit();
/*store the zoom range of the zoom bar selected by the user*/
editor.putInt("zoomLevel", myZoomBar.getProgress());
editor.commit();
myZoomBar.setVisibility(View.INVISIBLE);
//zoomChanging= false;
myZoomBar.getProgress();
t.cancel();
setZoomLevel();
}
}
+isZoombarVisible() : boolean
+doError(ex1 : Exception) : void
+killOnError() : void
+repaint() : void
+setErrorDialog() : void
+onCreate(savedInstanceState : Bundle) : void
-handleIntent(intent : Intent) : void
#onNewIntent(intent : Intent) : void
-doMixSearch(query : String) : void
#onPause() : void
#onResume() : void
+onCreateOptionsMenu(menu : Menu) : boolean
+onOptionsItemSelected(item : MenuItem) : boolean
+calcZoomLevel() : float
-setZoomLevel() : void
+onSensorChanged(evt : SensorEvent) : void
+onTouchEvent(me : MotionEvent) : boolean
+onKeyDown(keyCode : int, event : KeyEvent) : boolean
+onAccuracyChanged(sensor : Sensor, accuracy : int) : void
+onTouch(v : View, event : MotionEvent) : boolean



Gambar 4.5 *Class Diagram* Aplikasi Informasi Gedung UB

Sumber : Perancangan

1. **Class infoGedung**

Tabel 4.16. Deskripsi *class* infogedung.

Sumber : Perancangan

Nama class : infoGedung
Deskripsi : *Class* ini adalah *class* yang pertama kali dieksekusi. *Class* infoGedung berfungsi sebagai awalan dari aplikasi yang merepresentasikan menu utama dari aplikasi Informasi Gedung UB. Class infoGedung adalah turunan dari class activity.
Nama *Method* : onCreate (Bundle savedInstanceStated) **Fungsi:** *Method* ini adalah *method* yang pertama kali dijalankan saat *class* infoGedung dijalankan. *Method* ini berisi deklarasi dari komponen-komponen yang menyusun *layout* menu utama.
Nama *Method* : onClick (View v) **Fungsi:** *Method* ini digunakan untuk memberi *event* pada saat menekan komponen-komponen yang menyusun *layout* menu utama.

2. **Class cariData**

Tabel 4.17. Deskripsi *class* cariData.

Sumber : Perancangan

Nama class : cariData
Deskripsi :

Class cariData adalah class yang merepresentasikan menu pencarian dari aplikasi Informasi Gedung UB. Class cariData adalah turunan dari class activity.
Nama *Method* : onCreate (Bundle savedInstanceStated) **Fungsi:** *Method* ini adalah *method* yang pertama kali dijalankan saat *class* cariData dijalankan. *Method* ini berisi deklarasi dari komponen-komponen yang menyusun *layout* menu pencarian.
Nama *Method* : onClick (View v) **Fungsi:** *Method* ini digunakan untuk memberi *event* pada saat menekan komponen-komponen yang menyusun *layout* menu pencarian.

3. Class detailInformation

Tabel 4.18. Deskripsi *class* detailInfo.

Sumber : Perancangan

Nama class : detailInfo
Deskripsi : *Class* detailInfo adalah class yang menampung fungsi untuk menampilkan detail informasi dari aplikasi Informasi Gedung UB. Class infoGedung adalah turunan dari class activity.
Nama *Method* : onCreate (Bundle savedInstanceStated) **Fungsi:** *Method* ini adalah *method* yang pertama kali dijalankan saat *class* infoGedung dijalankan. *Method* ini berisi deklarasi dari komponen-komponen yang menyusun *layout* menu detailInformasi.

4. Class MixView

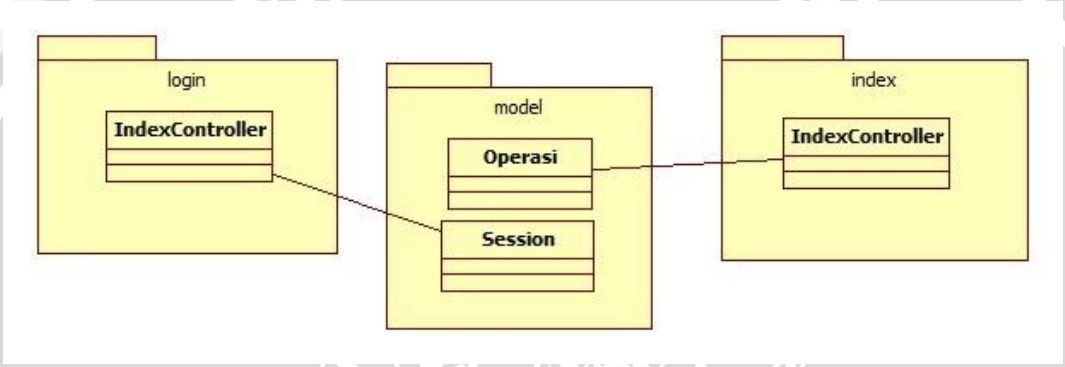
Tabel 4.19. Deskripsi *class* mixView.

Sumber : Perancangan

Nama class : mixView
Deskripsi : *Class* mixView adalah class yang merepresentasikan fungsi-fungsi utama dari aplikasi Informasi Gedung UB. Class mixView adalah turunan dari class activity.

Class MixView adalah *class* utama dari Mixare.
Nama *Method* : onCreate (Bundle savedInstanceStated) **Fungsi:** *Method* ini adalah *method* yang pertama kali dijalankan saat *class* cariData dijalankan. *Method* ini berisi deklarasi dari komponen-komponen yang menyusun *layout* fungsi utama dari aplikasi.

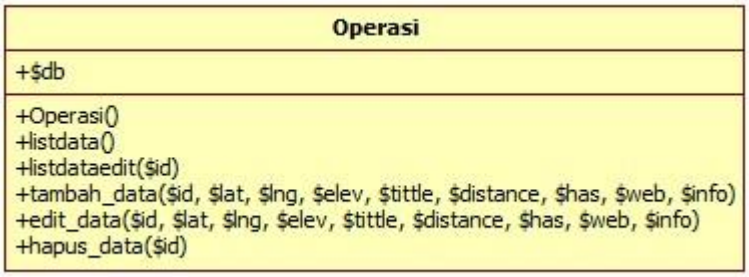
b. *Class Diagram* Aplikasi *Server*



Gambar 4.6 *Class Diagram Administrator*Informasi Gedung UB

Sumber : Perancangan

1. *Class* Operasi



Gambar 4.7 *Class* diagram untuk *class* Operasi

Sumber : Perancangan

Nama class : Operasi
Deskripsi :

Class Operasi merupakan *class* yang merupakan bagian dalam modul model.
Nama *Method* : Operasi() **Fungsi:** *Method* ini adalah *method* yang pertama kali dijalankan saat *class* Operasi dijalankan.
Nama *Method* : listdata() **Fungsi:** *Method* ini digunakan untuk menampilkan semua data yang ada di *database.*
Nama *Method* : listeditdata($id) **Fungsi:** *Method* ini digunakan untuk menampilkan data yang ada di *database*dengan id sesuai parameter.
Nama *Method* : tambah_data ($id,$lat,$lng,$elev,$tittle,$distance,$has,$web,$info) **Fungsi:** *Method* ini digunakan untuk menambahkan data kedalam *database*
Nama *Method* : edit_data ($id,$lat,$lng,$elev,$tittle,$distance,$has,$web,$info) **Fungsi:** *Method* ini digunakan untuk mengubah data di *database*
Nama *Method* : hapus_data ($id) **Fungsi:** *Method* ini digunakan untuk menghapus data di *database*

Tabel 4.20. Deskripsi *class* operasi.
Sumber : Perancangan

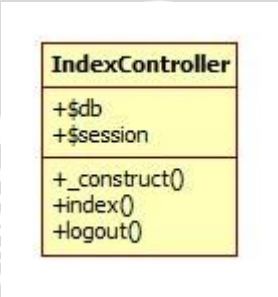
2. *Class* Session

Gambar 4.8 *Class* diagram untuk *class* Session
Sumber : Perancangan

Nama class : Session
Deskripsi : *Class* Session merupakan *class* yang merupakan bagian dalam modul model.
Nama *Method* : login($username, $password) **Fungsi:** *Method* ini adalah *method* yang digunakan untuk menyeleksi pengguna.

Tabel 4.21. Deskripsi *class* Session.
Sumber : Perancangan

3. *Class* IndexController



Gambar 4.9 *Class* diagram untuk *class* IndexController
Sumber : Perancangan

Nama class : IndexController
Deskripsi : *Class* Session merupakan *class* yang merupakan bagian dalam modul model.
Nama *Method* : _construct() **Fungsi:** *Method* ini adalah *method* digunakan untuk mebuat *constructor*.
Nama *Method* : index() **Fungsi:** *Method* ini adalah *method* utama dari *class* Session.
Nama *Method* : logout() **Fungsi:** *Method* ini adalah *method* digunakan untuk menghapus *session*.

Tabel 4.22. Deskripsi *class* indexController.
Sumber : Perancangan

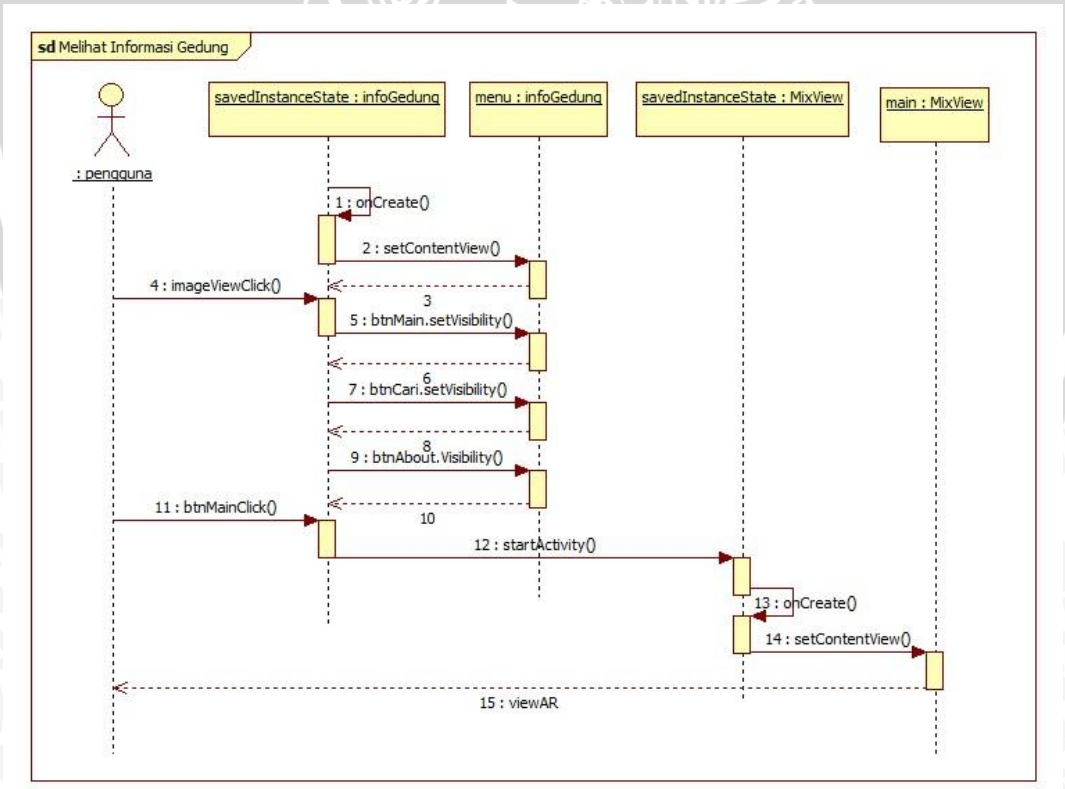
4.3.3. Diagram Sequence

Sequence diagram menggambarkan interaksi antar objek di dalam dan di sekitar sistem (termasuk pengguna, *display*, dan sebagainya) berupa *message* yang digambarkan terhadap waktu. *Sequence diagram* terdiri atas dimensi vertikal (waktu) dan dimensi horisontal (objek-objek yang terkait). *Sequece diagram* bisa digunakan untuk menggambarkan scenario atau rangkaian langkah-langkah yang dilakukan sebagai respons dari sebuah *event* untuk menghasilkan *output* tertentu. Diawali dari apa yang men-*trigger* aktivias tersebut, proses dan perubahan apa saja yang terjadi secara internal dan *output* apa saja yang dihasilkan.

1. Diagram *Sequence* Aplikasi *Client*

a. Diagram *Sequence* Melihat Informasi Gedung

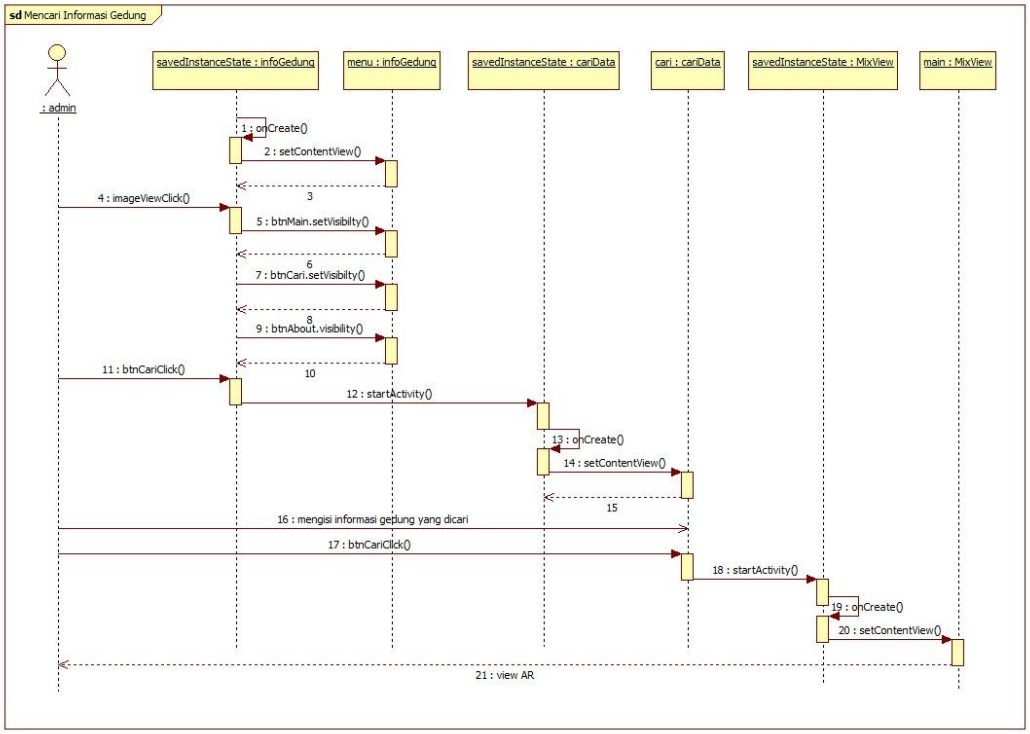
Gambar 4.10 merupakan diagram sekuensial Melihat Informasi Gedung. Diagram sekuensial ini menggambarkan interaksi ketika pengguna melihat informasi gedung dengan menggunakan perangkat lunak informasi gedung UB.



Gambar 4.10 Diagram *sequence* Melihat Informasi Gedung
Sumber : Perancangan

b. Diagram *Sequence* Mencari Informasi Gedung

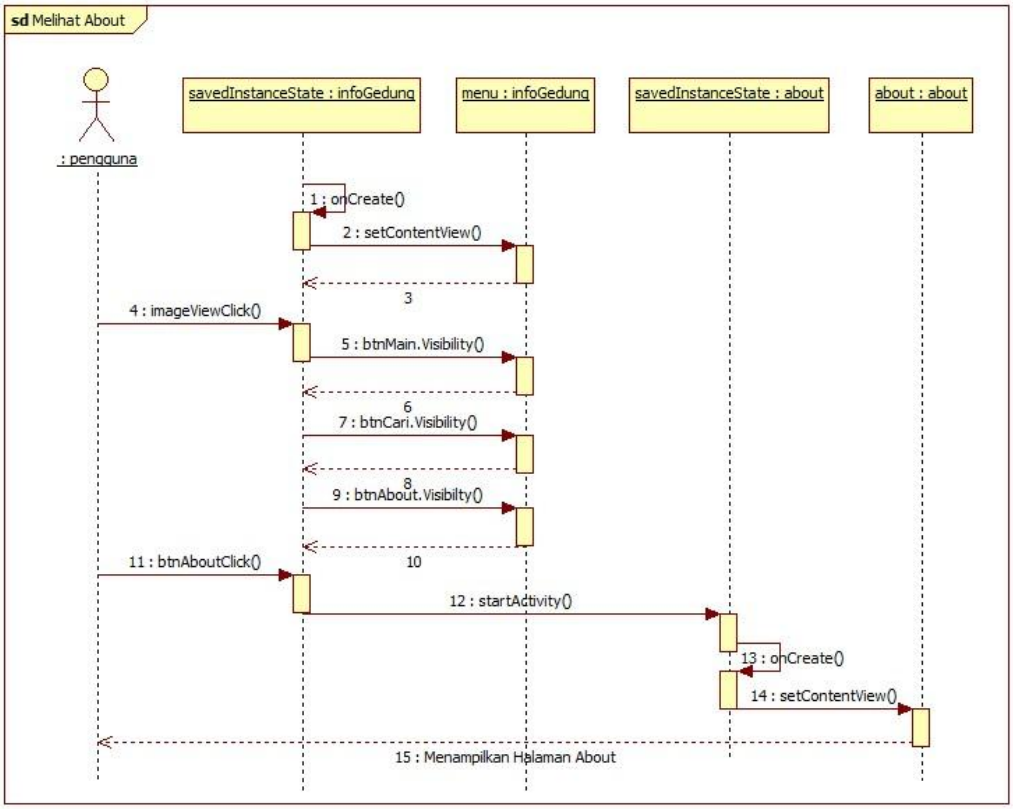
Gambar 4.11 merupakan diagram sekuensial Mencari Informasi Gedung. Diagram sekuensial ini menggambarkan interaksi ketika pengguna mencari informasi gedung dengan menggunakan perangkat lunak informasi gedung UB.



Gambar 4.11 Diagram *sequence* Mencari Informasi Gedung
Sumber : Perancangan

c. Diagram *Sequence* Melihat About

Gambar 4.12 merupakan diagram sekuensial Melihat About. Diagram sekuensial ini menggambarkan interaksi ketika pengguna melihat halaman aboutaplikasi gedung dengan menggunakan perangkat lunak informasi gedung UB.

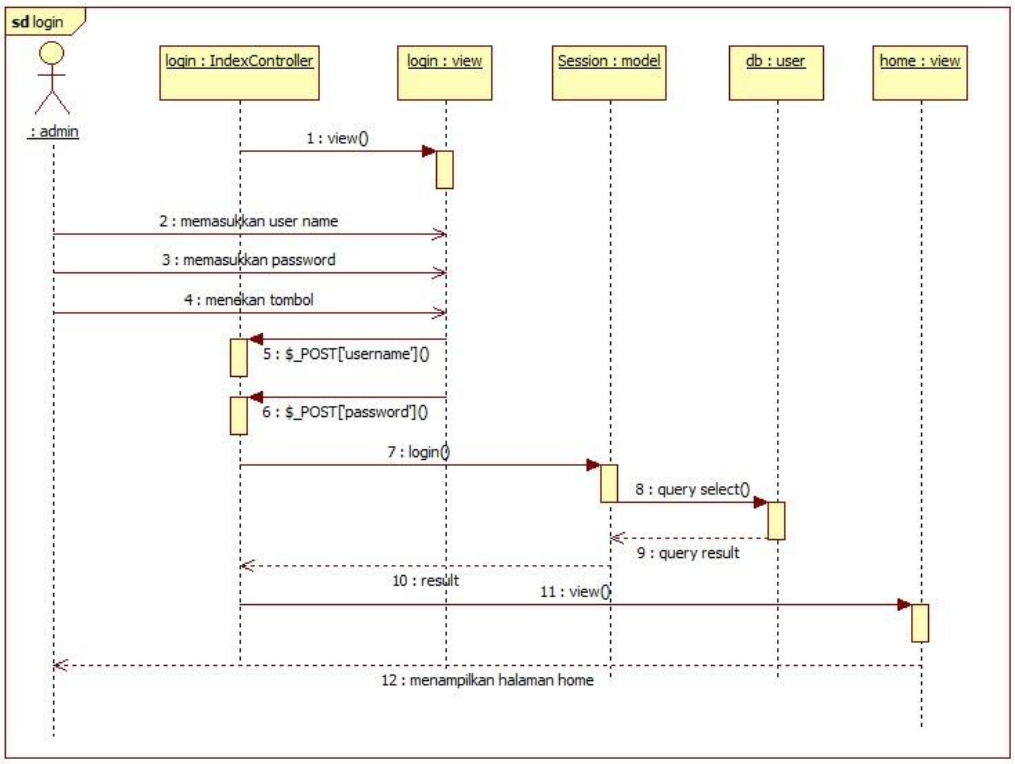


Gambar 4.12 Diagram *sequence* Melihat About
Sumber : Perancangan

2. Diagram *Sequence* Aplikasi Admin

a. Diagram *Sequence* Login

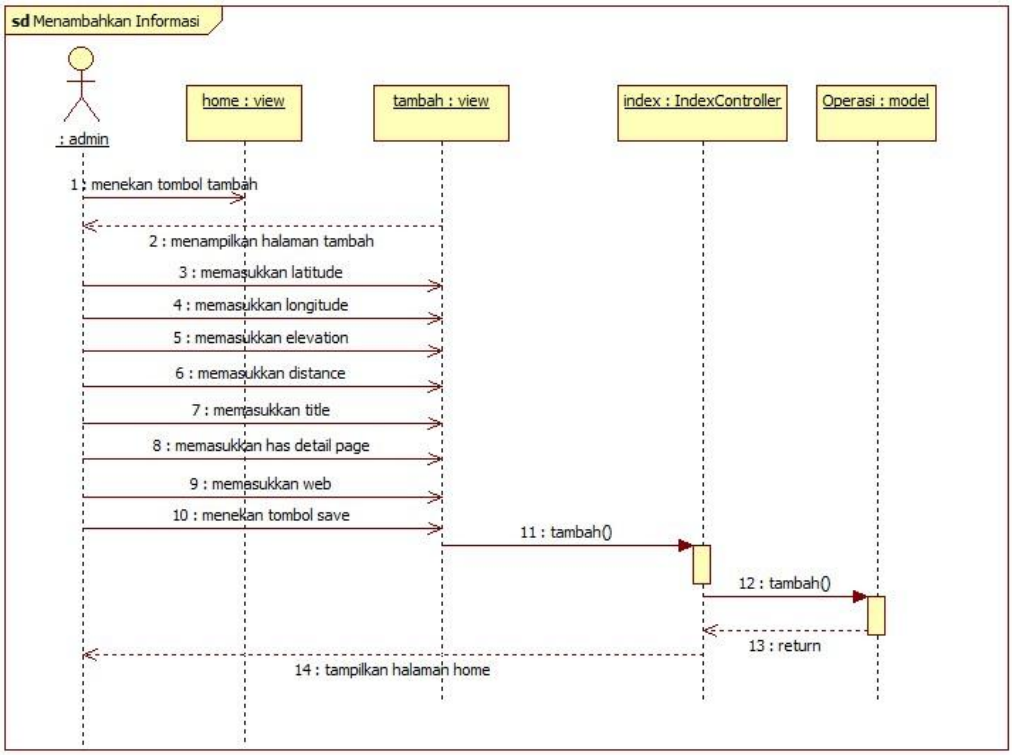
Gambar 4.13 merupakan diagram sekuensial Login. Diagram sekuensial ini menggambarkan interaksi ketika admin melakukan proses login.



Gambar 4.13 Diagran *Sequence* Login
Sumber : Perancangan

b. Diagram *Sequence* Menambah Informasi di *Database*

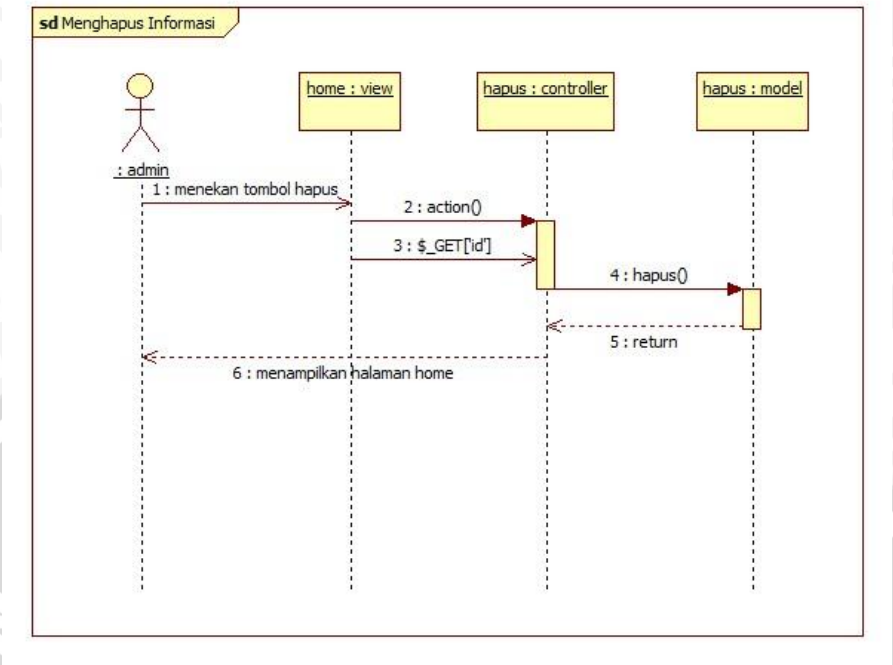
Gambar 4.14 merupakan diagram sekuensial Menambah Informasi di *database*. Diagram sekuensial ini menggambarkan interaksi ketika admin melakukan proses penambahan data di *database*.



Gambar 4.14 Diagram *sequence* Menambahkan Informasi
Sumber : Perancangan

c. Diagram *Sequence* Menghapus Informasi di *Database*

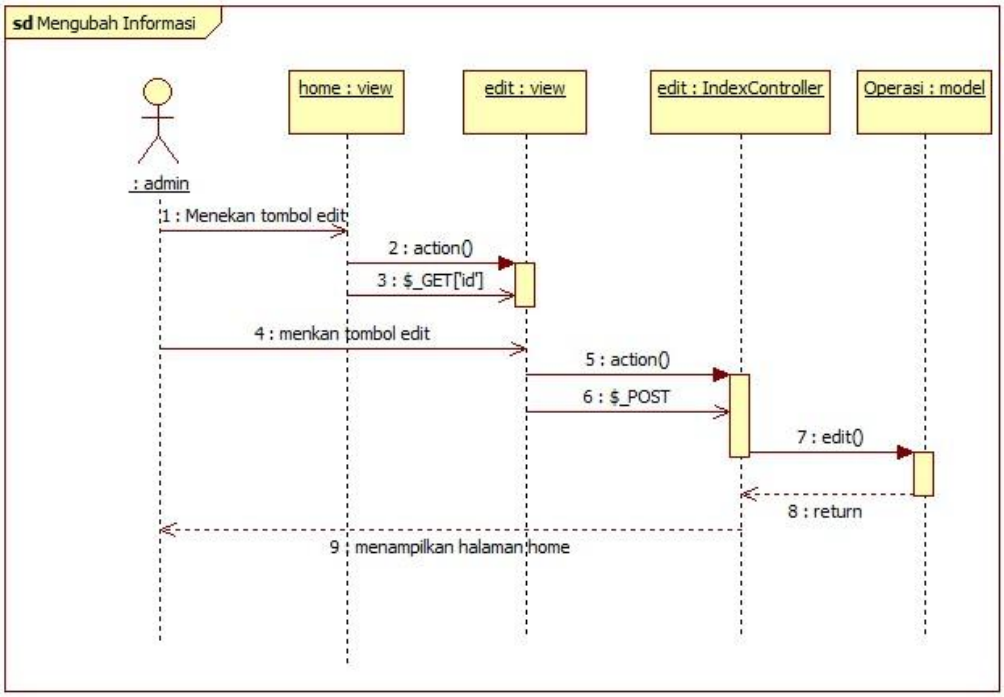
Gambar 4.15 merupakan diagram sekuensial Menghapus Informasi di *database*. Diagram sekuensial ini menggambarkan interaksi ketika admin melakukan proses penghapusan data di *database*.



Gambar 4.15 Diagram *sequence* Menghaspus Informasi
Sumber : Perancangan

d. Diagram *Sequence* Mengubah Informasi di *Database*

Gambar 4.16 merupakan diagram sekuensial Mengubah Informasi di *database*. Diagram sekuensial ini menggambarkan interaksi ketika admin melakukan proses pengubahan data di *database*.



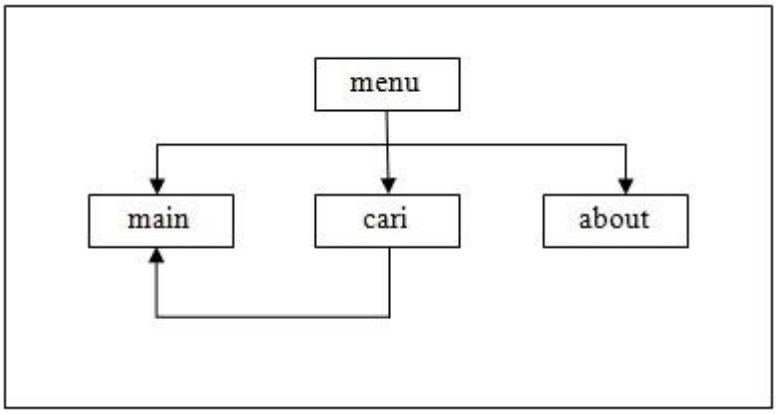
Gambar 4.16 Diagram *Sequence* Mengubah Informasi
Sumber : Perancangan

4.3.4. Perancangan Antarmuka

Pada bagian ini dijelaskan mengenai perancangan antarmuka dari perangkat lunak Informasi Gedung UB. Perancangan antarmuka sangat diperlukan untuk mempermudah pengguna dalam menggunakan perangkat lunak Informasi Gedung UB. Perancangan antarmuka dibagi menjadi dua yaitu perancangan antarmuka pada sisi pengguna dan pada sisi *administrator*.

1. Perancangan Antarmuka Pada Aplikasi Pengguna

Antarmuka pada aplikasi pengguna berupa *layout* Android yang memudahkan pengguna untuk menggunakan aplikasi Informasi Gedung UB.



Gambar 4.17 *Sitemap* Antarmuka Aplikasi Informasi Gedung UB
Sumber : Perancangan

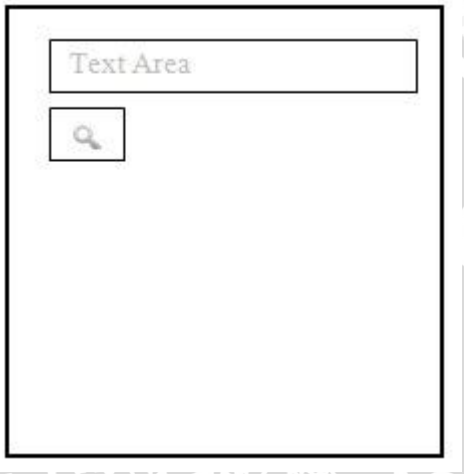
a. Halaman menu

Halaman menu merupakan halaman yang berisi menu-menu yang dapat dipilih oleh pengguna. Pada halaman ini terdapat tiga tombol yang mewakili aktivitas yang dapat dilakukan oleh pengguna yaitu tombol Info Gedung, tombol cari dan tombol about. Tombol info gedung berfungsi untuk melihat Informasi Gedung UB yang terdapat dalam radius 100m dari posisi pengguna. Tombol cari berfungsi untuk menampilkan halaman cari. Tombol about berfungsi untuk menampilkan halaman about. Pada gambar 4.18 menunjukkan rancangan antarmuka halaman menu.

Gambar 4.18 Rancangan Antarmuka Halaman Menu
Sumber : Perancangan

b. Halaman Cari

Halaman cari merupakan halaman yang berfungsi untuk melakukan pencarian gedung UB berdasarkan nama Fakultas atau Jurusan. Terdapat *text area*untuk menuliskan gedung yang dicari dan tombol yang untuk melakukan proses pencarian. Gambar 4.19 menunjukkan rancangan antarmuka halaman cari.



Gambar 4.19 Rancangan Antarmuka Halaman Cari
Sumber : Perancangan

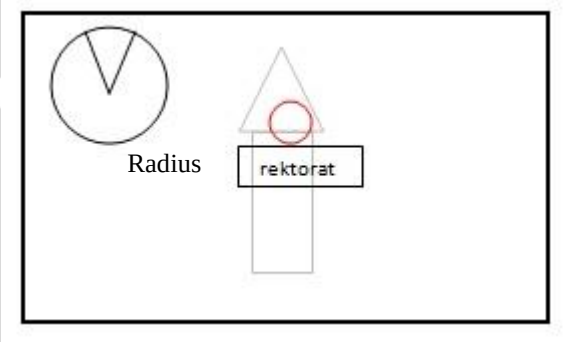
c. Halaman about

Halaman about berisi informasi tentang aplikasi Informasi Gedung UB. Di halaman ini, pengguna mengetahui sekilas tentang aplikasi dan penulis.Gambar 4.20 menunjukkan rancangan antarmuka halaman about.

Gambar 4.20 Rancangan Antarmuka Halaman About
Sumber : Perancangan

d. Halaman Utama

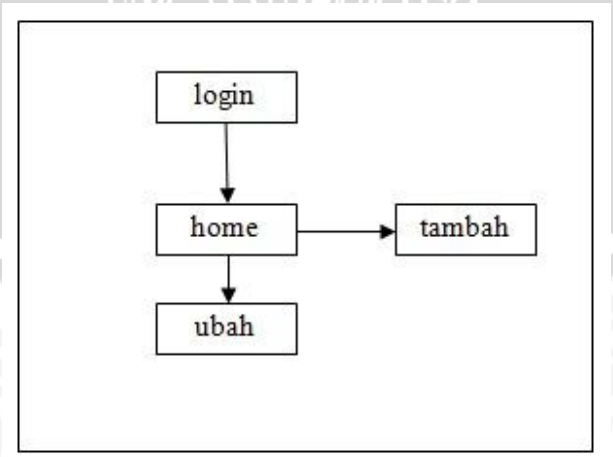
Halaman utama merupakan halaman utama dari aplikasi Informasi Gedung UB. Halaman ini merupakan halaman yang menampilkan informasi Gedung UB berupa AR dan message box sebagai media. Halaman ini mengakses fungsi kamera perangkat sehingga hasil dari tangkapan kamera digunakan sebagai background. Pada halaman ini terdapat lingkaran radius yang berfungsi mendeteksi koordinat dari gedung dalam radius tertentu. Gambar 4.21 menunjukkan rancangan antarmuka halaman main.



Gambar 4.21 Rancangan Antarmuka Halaman utama
Sumber : Perancangan

2. Perancangan Antarmuka Pada Aplikasi *Administrator*

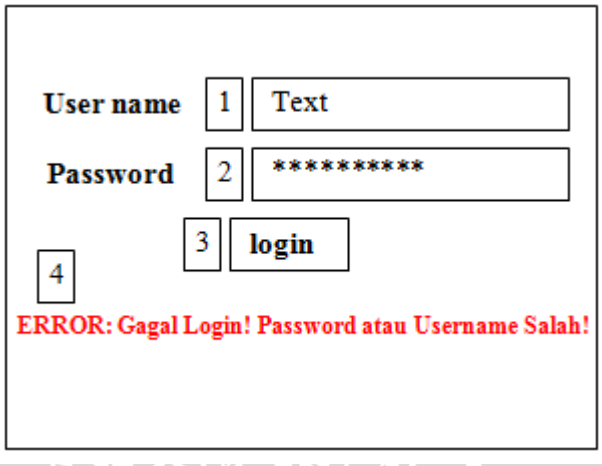
Antarmuka pada sisi *admisnitrator*berupa halaman *web.* Perancangan antarmuka pada sisi *administrator* bertujuan untuk memudahkan admin untuk melakukan penambahan, penghapusan dan pengubahan data di *server*. Gambar 4.22 menunjukkan *sitemap*dari aplikasi *administrator*.



Gambar 4.22 *Sitemap* Antarmuka Aplikasi *Administrator*
Sumber : Perancangan

a. Halaman login

Halaman login merupakan salah satu antarmuka pengguna untuk aplikasi *administrator* yang berfungsi untuk mempermudah *administrator* dalam melakukan login. Gambar 4.23 akan menunjukkan perancangan tampilan antarmuka dari halaman login.



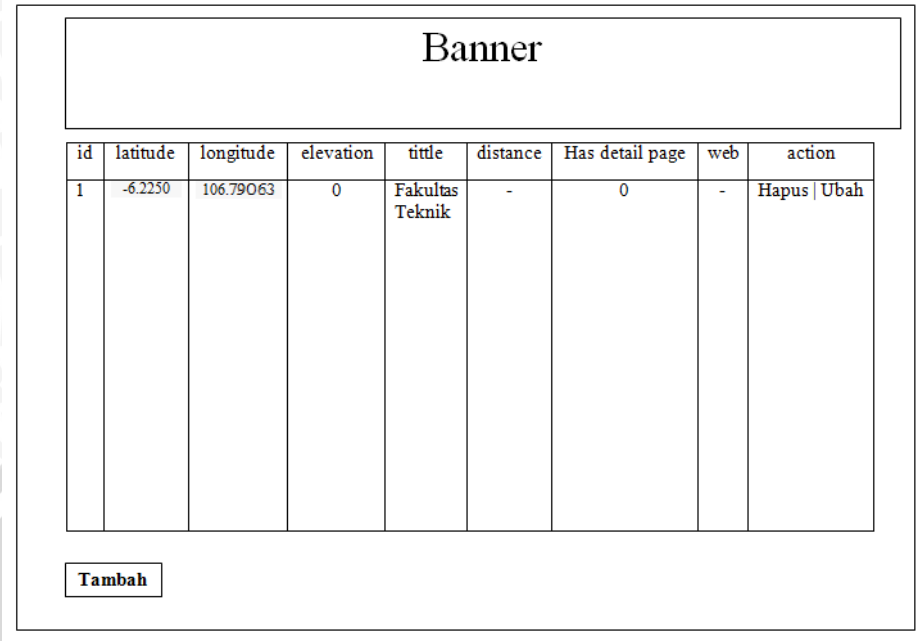
Gambar 4.23 Rancangan Antarmuka Halaman Login
Sumber : Perancangan

Keterangan Gambar 4.23 :

1. *Text field* untuk mengisi *username*.
2. *Text field* untuk mengisi *password*.
3. Tombol loginuntuk melakukan proses login.
4. Pesan eror jika tidak berhasil melakukan login.

b. Halaman home

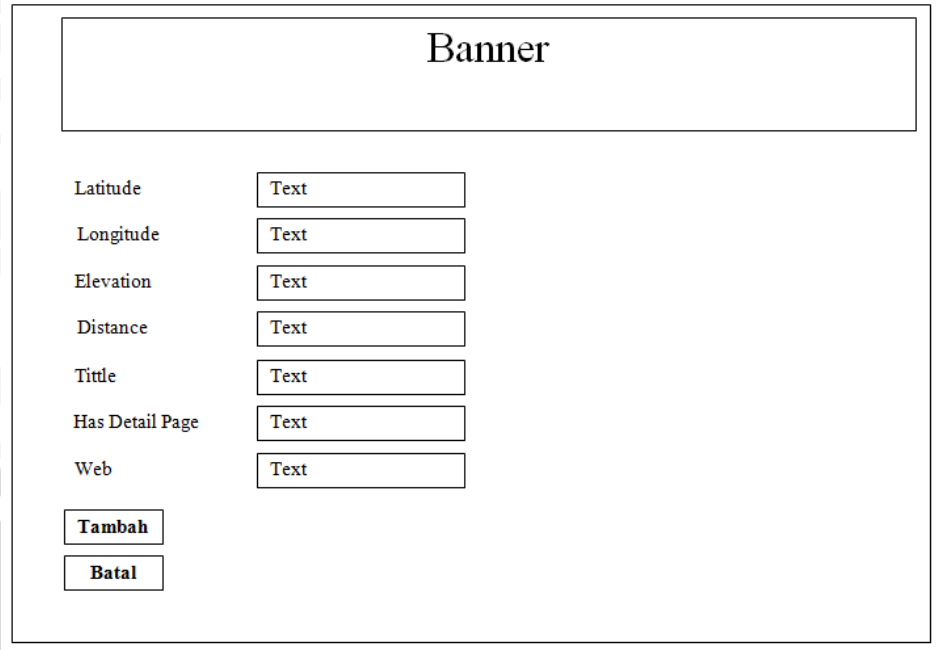
Halaman home merupakan halaman utama yang menampilkan data-data yang disimpan di basis data. Informasi-informasi tentang gedung UB dapat dilihat di halaman ini. Halaman ini berisi tabel yang menampung informasi tentang gedung – gedung UB. Halaman ini juga berisi link untuk mengubah atau menghapus data dan berisi tombol tambah untuk menuju halaman tambah. Gambar 4.24 menunjukkan rancangan antarmuka halaman home.



Gambar 4.24 Rancangan Antarmuka Halaman home.
Sumber : Perancangan

c. Halaman tambah

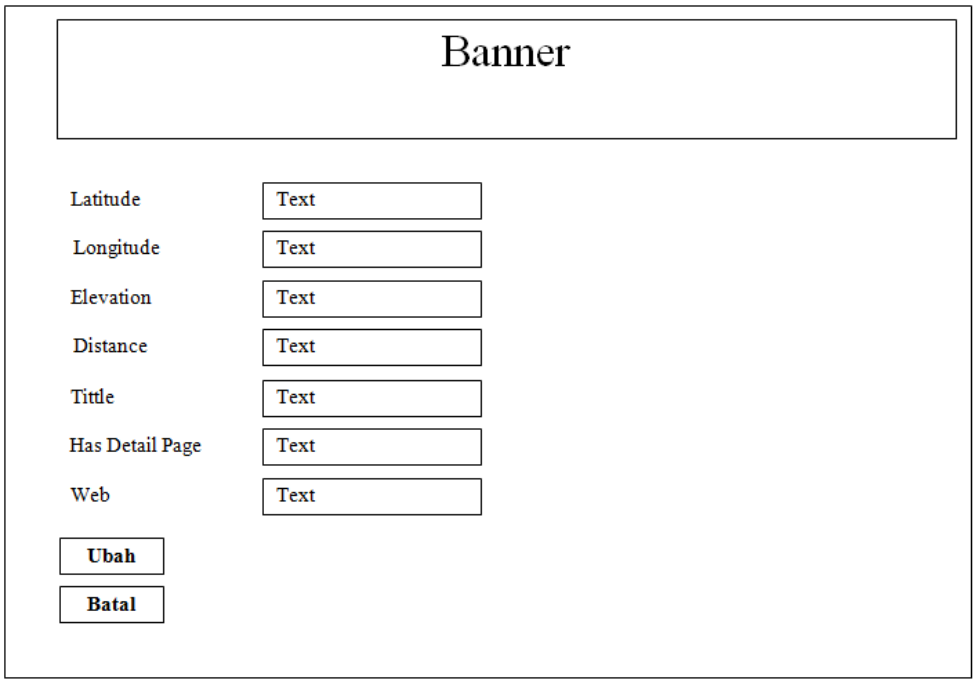
Halaman tambah berfungsi untuk memudahkan admin untuk menambahkan data pada basis data. Halaman ini berisi *text field*untuk mengisi data berupa data *latitude, longitude, elevation, tittle, distance, has detailpage, web.* Halaman ini juga berisi tombol untuk melakukan proses penambahan data pada basis data. Gambar 4.25 menunjukkan perancangan antarmuka halaman tambah.



Gambar 4.25 Rancangan Antarmuka Halaman Tambah
Sumber : Perancangan

d. Halaman ubah

Halaman ubah berfungsi untuk memudahkan admin untuk mengubah data pada basis data. Halaman ini berisi *text field* untuk mengubah data berupa data *latitude, longitude, elevation, tittle, distance, has detailpage, web.* Halaman ini juga berisi tombol untuk melakukan proses perubahan data pada basis data. Gambar 4.26 menunjukkan perancangan antarmuka halaman ubah.



Gambar 4.26 Rancangan Antarmuka Halaman ubah
Sumber : Perancangan

BAB V
IMPLEMENTASI

Bab ini membahas mengenai tahapan implementasi Aplikasi Informasi Gedung UBberdasarkan hasil yang telah didapatkan dari analisis kebutuhan dan proses perancangan perangkat lunak. Pembahasan terdiri atas penjelasan tentang spesifikasi sistem, batasan – batasan dalam implementasi, implementasi basis data, implementasi tiap *class* pada *file* program, implementasi algoritma, dan implementasi antarmuka.

5.1. Spesifikasi Sistem

Perangkat lunak Informasi Gedung UB Book dikembangkan dalam lingkungan implementasi yang terdiri dari perangkat keras dan perangkat lunak.

5.1.1. Spesifikasi Perangkat Keras

Spesifikasi perangkat keras yang dipakai dalam proses pengembangan dijelaskan pada Tabel 5.1.

Personal Computer	
Processor	Intel(R) Core(TM) i3-2100 CPU @ 3.10GHz
Memory (RAM)	4096 MB
Harddisk	WD 500 GB
Motherboard	GIGABYTE H61M-DS2
Graphic Card	Intel(R) HD Graphics Family 1GB (on board)
Monitor	LG Flatron ez T530s
Webcam	-

Tabel 5.1 Tabel Spesifikasi Perangkat Keras Komputer
Sumber: Implementasi

Spesifikasi perangkat keras yang dipakai dalam implementasi aplikasi Informasi Gedung UB dijelaskan pada Tabel 5.2

Samsung Galaxy Gio (GT-S5660)	
Processor	800 MHz
Memory (RAM)	278 MB

Storage	158 MB (Internal)
Display size	320 x 480 pixels, 3.2 inches (~180 ppi pixel density)
Sensor	Accelerometer, proximity, compass
Camera	3.15 MP, geo-tagging

Tabel 5.2 Spesifikasi Perangkat Keras Perangkat *Mobile*
Sumber: Implementasi

5.1.2. Spesifikasi Perangkat Lunak

Spesifikasi perangkat lunak yang dipakai dalam proses pengembangan perangkat lunak Informasi Gedung UB dijelaskan pada Tabel 5.3.

Personal Computer	
Operating System	Windows 7 Ultimate 64-bit (6.1, Build 7601) Service Pack 1
Programming Language	Java (Aplikasi Client), PHP (Aplikasi Administrator)
Software Development Kid	Java Development Kit 6 Update 23
Programming Environment	Java Runtime Environment 6
Augmented Reality Engine	Mixare (Mix Reality Engine)
Database Management System	My SQL
Integrated Development Environment	Eclipse HELIOS

Tabel 5.3 Spesifikasi Perangkat Lunak Komputer.
Sumber: Implementasi

Spesifikasi perangkat lunak yang dipakai dalam implementasi aplikasi Informasi Gedung UB dijelaskan pada Tabel 5.4.

Samsung Galaxy Gio (GT-S5660)	
OS	Android 2.3.4 (Gingerbread)

Tabel 5.4 Spesifikasi Perangkat Keras Perangkat *Mobile*
Sumber: Implementasi

5.2.Batasan – Batasan Implementasi

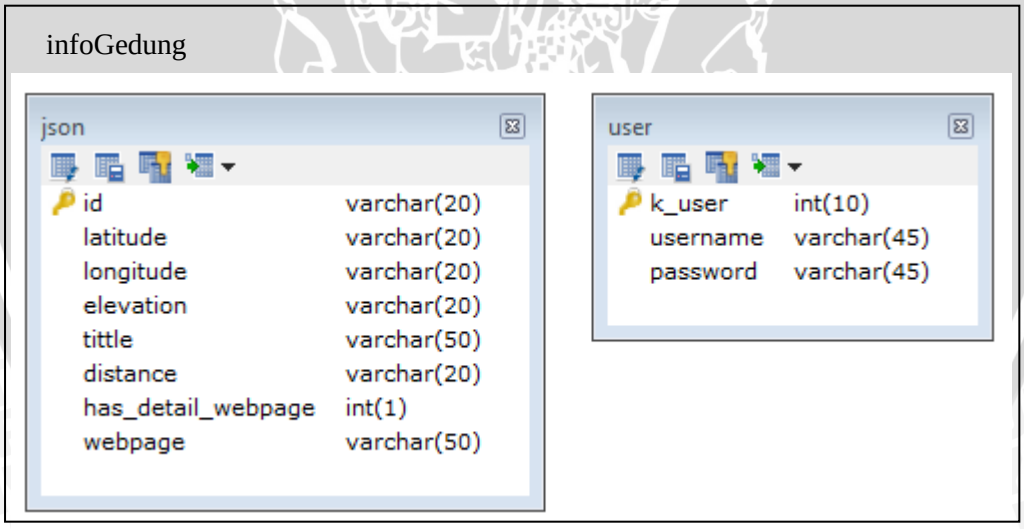
Beberapa batasan dalam mengimplementasikan perangkat lunak Informasi Gedung UBadalah sebagai berikut :

1. Perangkat lunak Informasi Gedung UBdirancang dan dijalankan dengan menggunakan *Eclipse Helios*.

2. Aplikasi untuk administrator menggunakan bahasa pemrograman PHP.
3. Perangkat lunak ini dibangun dengan menggunakan teknologi *augmented reality* dengan *Augmented RealityEngineMixare (Mix Augmented Reality Engine).*
4. *Database Management System* yang digunakan adalah MySQL.
5. *Datasource* yang digunakan adalah *datasource* yang dibangun sendiri.
6. Pengambilan data dari server menggunakan teknologi JSON.

5.3.Implementasi Basis Data

Implementasi penyimpanan data dilakukan dengan *database management system* MySQL. Hasil implementasi penyimpanan data ini berupa *script – script* SQL. Hasil implementasi SQL pada *database* ini dimodelkan dalam diagram konseptual *entity relationship.* Gambar 5.1 menggambarkan diagram konseptual *entitiy relationship* dari Aplikasi Informasi Gedung UB.



Gambar 5.1 Diagram ER konseptual dari Aplikasi *Administrator*
Sumber: Implementasi

5.4.Implementasi *Class* dan *Interface* Pada *File* Program

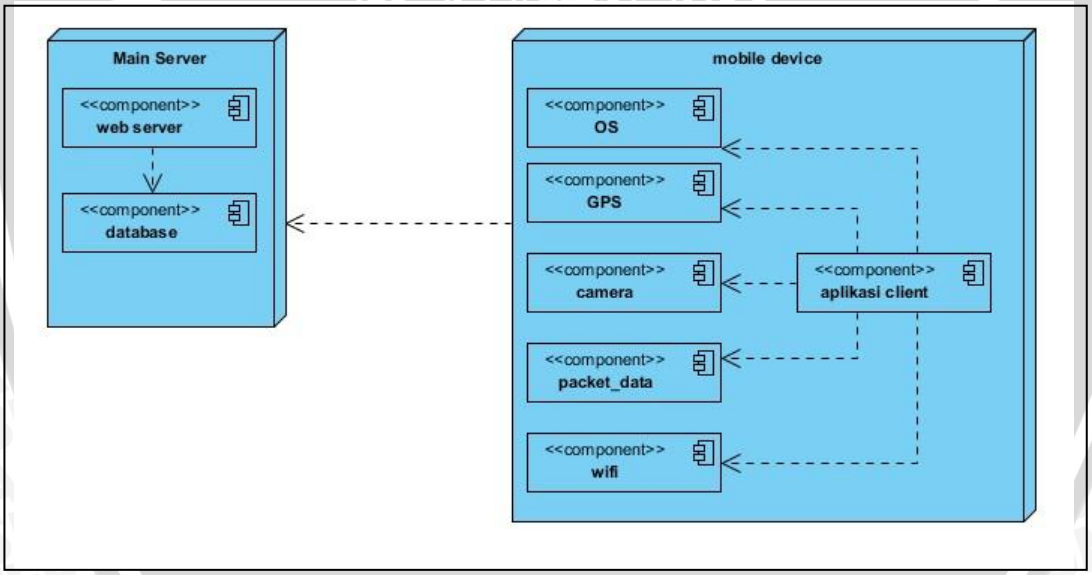
No.	*Package*	Nama *Class*	Nama *File* Program
1.	org.mixare	infoGedung	infoGedung.java

2.	org.mixare	cariData	cariData.java
3.	org.mixare	About	about.java
4.	org.mixare	MixView	MixView.java
5.	org.mixare	detailInformasi	detailInformasi.java

Tabel 5.5 Implementasi *class* pada kode program *.java
Sumber: Implementasi

5.5.*Deployment Diagram*

Deployment/physical diagram menggambarkan detail bagaimana komponen di-*deploy* dalam infrastruktur sistem, di mana komponen akan terletak (pada mesin, server atau piranti keras apa), bagaimana kemampuan jaringan pada lokasi tersebut, spesifikasi server, dan hal-hal lain yang bersifat fisikal. Sebuah *node* adalah server, *workstation*, atau piranti keras lain yang digunakan untuk men-*deploy* komponen dalam lingkungan sebenarnya. Gambar 5.2 menunjukkan *deployment diagram* dari perangkat lunak Informasi Gedung UB.



Gambar 5.2 *Deployment Diagram*Informasi Gedung UB.
Sumber: Implementasi

Gambar 5.2 menggambarkan komponen-komponen yang menyusun perangkat lunak Informasi gedung UB dan dimana komponen-komponen tersebut di-*deploy*. Perangkat lunak Informasi Gedung UB menggunakan *web hosting* sebagai *main server*. Komponen-komponen yang menyusun *main server* adalah *webserver* dan *database*. Webserver digunakan sebagai *web service* untuk mengirim data yang dibutuhkan. *Web service* yang digunakan adalah JSON.

Database digunakan sebagai media untuk penyimpanan data. Database yang digunakan adalah *mysql*.*Mobile device* yang digunakan perangkat lunak Informasi Gedung UB adalah Samsung Galaxy Gio dengan *OS* Android. *Mobile device* terdiri dari beberapa komponen yang dibutuhkan oleh perangkat lunak Informasi Gedung UB, yaitu OS, kamera, GPS, *mobile internet*, dan aplikasi *client*. OS berfungsi sebagai *environment* tempat berjalannya aplikasi *client*. Aplikasi *client* mengakses fungsi kamera untuk menampilkan gedung-gedung yang akan di deteksi. GPS digunakan untuk menentukan posisi pengguna dan mengenali *marker* yang berupa titik koordinat dari gedung. Mobile internet digunakan untuk mengakses data yang ada di *main server*.

5.6. Implementasi Algoritma

Perangkat lunak Informasi Gedung UB memiliki beberapa proses atau *method* yang terdapat pada beberapa *class*. Beberapa *method* yang akan dicantumkan dalam penulisan makalah skripsi ini hanya untuk algoritma dari beberapa proses (operasi) saja sehingga tidak semua *method* akan dicantumkan. Implementasi algoritma ini akan direpresentasikan dalam bentuk *pseudocode* dari algoritma – algoritma tersebut.

5.6.1 Implementasi Algoritma Aplikasi Administrator

1. Login

Proses *login* dilakukan setelah *user* menekan tombol login pada halaman login. Ketika user menekan tombol login maka akan mengirimkan nilai dari username dan password menuju method index() dan pada controller class LoginController. Pada method index() ini akanada variabel data untuk menampung data kiriman. Kemudian memanggil method login(username,password) pada model class Session yang memiliki nilai kembalian dan akan ditampung dalam variabel login. *Method* login(username, password) pada *class* Session berfungsi untuk mengeksekusi *query* logindengan parameter username dan password. Hasil dari eksekusi query akan dimasukkan dalam session dan dikembalikan dalam variabel result. Nilai kembalian pada controller yang ditampung pada variabel login akan diseleksi apakah login sukses

atau tidak, jika sukses akan menuju halaman home , jika gagal akan kembali ke halaman login. Implementasi algoritma untuk proses *login*digambarkan pada Gambar 5.3.

```
NAMA ALGORITMA Login

DEKLARASI
$db;
$session;
$data = $_POST;
$error = false;
DESKRIPSI
Masukan :
   1. $username
   2. $password
Proses :
   1. call index() in controller class LoginController and send
      $username, $password.
      a. Memberi nilai variable data dengan nilai dari pengiriman
         dengan metode post username dan password dari halaman
         login.
      b. $login = $this->session->login($data['username'],
         $data['password']):
          i.   $pass= md5.
         ii.   $statement = query login.
        iii.   $query= eksekusi query.
         iv.   If($query) then $_SESSION['appLogin']['username']=
               $result['username'] andreturn $result.
          v.   If not, return false
      c. If(!$login) then:
           i.  $error = 'no user'.
      d. else:view home page.
Keluaran :
   1. view main page.
```

Gambar 5.3 Implementasi Algoritma *Login.*
Sumber: Implementasi.

Pada algoritma login terdapat beberapa variabel, antara lain $db, $session, $data , $error. Variabel $data berisi variabel $_POST yang berarti variabel yang

menampung hasil dari *method* POST. Terdapat dua variabel masukan yaitu variabel username dan variabel password yang ditunjukkan pada pada baris 1 dan 2 pada bagian masukan dari gambar 5.3.

Proses jalannya algoritma login adalah sebagai berikut :

Baris 1 berarti memanggil method index() pada controller class LoginController dan mengirimkan nilai username dan password. Method index berisi :

a. Memberi nilai variable data dengan nilai dari pengiriman dengan metode postusername dan password dari halaman login.
b. Variable login menampung pemanggilan method login(username,password) yang berisi:
   i. Variable pass menampung hasil enkripsi password dengan md5.
   ii. Variable statement menampung*query* login.
   iii. Variable query menampunghasil eksekusi *query*.
   iv. Seleksi kondisi jika kembalian query ada maka data user id dimasukkan dalam *session* dan variable result 1.
   v. Jika kembalian query tidak ada maka result = 0.
   vi. Mengembalikan nilai result.
c. Jika variable login bernilai 0 maka dilakukan langkah berikut:
   i. Kembali ke halaman login dengan menampilkan pesan error.
d. Jika variable login tidak bernilai 0 maka masuk ke halaman home.

2. Menambahkan Data

Proses menambahkan data dilakukan setelah *user* memasukkan data-data pada *field* yang ada pada halaman tambah, kemudian menekan tombol tambah. Pada saat menekan tombol tambah, *method* save() pada *class* IndexController dipanggil. Gambar 5.4 merupakan algoritma menambahkan data.

```
NAMA ALGORITMA Menambahkan data

DEKLARASI
$db;
$user;
```

```
$data = $_POST;
$error = false;
DESKRIPSI
Masukan :
   1. $id
   2. $lat
   3. $long
   4. $elev
   5. $title
   6. $has
   7. $web
   8. $info
Proses :
   1. call save ():
         a. $data = $_POST.
         b. $tambah = $this->operasi->tambah_data():
               i.  $statement = query insert data
              ii.  $query = eksekusi query.
             iii.  If($query) then result $query.
              iv.  If not result result.
         c. If not succes :
               i.  Variabel $error -> 'gagal'.
         d. view home page.
Keluaran :
            1. view home page.
```

Gambar 5.4 Implementasi Algoritma Menambahkan Data di Server.
Sumber: Implementasi.

Pada algoritma menambahkan data terdapat beberapa variabel, antara lain $db, $user, $data, $error.Variabel $data berisi variabel $_POST yang berarti variabel yang menampung hasil dari *method* POST. Terdapat beberapa variabel masukan yaitu $id, $lat, $long, $elev, $title, $has, $web, $info yang ditunjukkan pada pada baris 1 sampai8 pada bagian masukan dari gambar 5.4.

Proses jalannya algoritma menambah data adalah sebagai berikut :

Baris 1 menunjukkan pemanggilan method save () pada view halaman tambah yang berisi:

a. Variabel $data menampung variabel $_POST

b. Variabel $tambah berisipemanggilan*method* tambah_data() pada model operasi yang berisi :
   i. Variabel $statement menampung query untuk menmbahkan data.
   ii. Variable query menampung hasil eksekusi query.
   iii. Seleksi kondisi jika kembalian query variable result = query.
   iv. Jika kembalian query tidak ada maka result = 0.
   v. Mengembalikan nilai result.
c. Jika tidak berhasil menambahkan data maka :
   i. Variabel $error = ‘gagal’.
d. Menuju halaman home.

5.6.2 Implementasi Algoritma Aplikasi *Client*

1. Mencari Informasi Gedung

Fungsi ini untuk melakukan pencarian informasi gedung UB. Setelah masuk menu search, isikan lokasi yang dicari pada *text box* kemudian tekan tombol cari. Setelah itu proses akan berjalan. Gambar 5.5 merupakan algoritma untuk mencari informasi gedung UB

```
NAMA ALGORITMA Mencari informasi gedung

DEKLARASI
edit : EditText
btnImg : ImageButton
url, key, search : String
i : Intent
DESKRIPSI
Proses :
  1. run btnImg.setOnClickListener(new onClickListener ())
  2. url = edit.getText().toString();
  3. Intent     i     =     new     Intent(getApplicationContext(),
     MixView.class);
  4. i.putExtra(key,url).
  5. run starActivityForResult(i,0).
  6. Class mixview run.
  7. If (extras not null) then
        a. search = extras.
```

```
   8. else :
         b. search = null.
   9. Request                server              "http://antonio-
      shoes.com/infogedungub/json.php?cari="+search+""
   10. Get JSON data.
Keluaran :
   1. View data from server.
```

Gambar 5.5 Implementasi algoritma mencari informasi gedung UB
Sumber: Implementasi

Pada algoritma mencari informasi gedung terdapat beberapa variabel antara lain edit dengan tipe data EditText, btnImg dengan tipe data ImageButton, url, key, search dengan tipe data String dan i dengan tipe data intent.

Proses jalannya algoritma mencari informasi UB adalah sebagai berikut :

Pada baris 1 : tombol cari di tekan, method btnImg.setOnClickListener(new onClickListener ()) dijalankan.

Pada baris 2 : Kemudian variabel url diisi dengan nilai dari variabl edit.

Pada baris 3 : Membuat objek Intent i yang berisi kelas untuk menampilkan informasi.

Pada baris 4 : Memasukkan variabel url ke objek i dengan method i.putExtra(key,url), dimana key adalah nama variabel yang akan ditangkap oleh kelas untuk menampilka informasi, sedangkan url adalah nilai dari variabel key.

Pada baris 5 : Menjalankan method starActivityForResult(i,0) untuk memulai activity kelas untuk menampilkan informasi.

Pada baris 6 : Activity class untuk menampilkan informasi berjalan.

Pada baris 7 : Jika variabel key tidak kosong maka :

a. nilai dari variabel key ditampung di variabel search.

Pada baris 8 : Namun jika variabel key kosong maka :

a. variabel search tetap kosong.

Pada baris 9: Request ke server dengan alamat "http://antonio-shoes.com /infogedungub/ json.php?cari="+search+""

Pada Baris 10 : Server mengirim data berupa JSON sesuai data yang dicari.

2. Melihat Detail Informasi Gedung

NAMA ALGORITMA melihat detail informasi gedung

DEKLARASI

is : InputStream
result : String
jArray : JSONArray
sb : StringBuilder
listInfo : ListView
judul, info : String
map : HashMap<String, String>

DESKRIPSI

Proses :

1. class detailInformasi run.
2. nameValuePairs = new arrayList.
3. try :
   a. create http connection.
4. catcherror message.
5. Try :
   a. Create buffer object.
6. catch error message.
7. Try :
   a. Create json array and json object.
   b. For(i=0;i<jArray.length;i++) :
      i. json_data = jArray.getJSONObject(i).
      ii. judul = json_data.getString("tittle").
      iii. info = json_data.getString("info").
      iv. map = new HashMap<String, String>().
      v. map.put("Tittle",judul).
      i. map.put("Informasi","::"+info+"::").
      ii. daftarLokasi.add(map).
8. catch error message.
9. SimpleAdapter infoAdapter = new SimpleAdapter(this, daftarLokasi, R.layout.*buffertext*, new String[] {"Tittle", "Informasi"}, newint[] {R.id.*Judul*, R.id.*Info*});
10. listInfo.setAdapter(videoAdapter);
11. Give event onItemClickListener

```
Keluaran :
   1. view message box.
```

Gambar 5.6 Implementasi algoritma melihat detail informasi gedung UB
Sumber: Implementasi

Pada algoritma melihat detail informasi gedung terdapat beberapa variabel antara lain is dengan tipe data InputStream, result tipe data String, jArray tipe data JSONArray, sb tipe data StringBuilder, listInfo tipe data ListView, judul, info tipe data String, map tipe data HashMap<String, String>

Proses jalannya algoritma melihat detail informasi gedung yang ditunjukkan gambar 5.6 adalah sebagai berikut :

Pada baris 1 : Ketika memilih menu detail informasi dijalankan, kelas detailInformasi dijalankan

Pada baris2 : Membuat objek arrayList dengan nama nameValuePairs

Pada baris3 : Melakukan beberapa hal, antara lain :

a. Membuat objek HttpClient untu membuat koneksi dengan server dengan nama httpclient.
b. Membuat objek HttpPost untuk mengirimkan alamat server dengan nama httppost.
c. Memasang entity untuk objek httpost untuk meng-encode alamat server.
d. Membuat object HttpRespone dengan nama respone untuk menampung hasil eksekusi httpclient dengan parameter httpost.
e. Membuat objek HttpEntity dengan nama entity untuk menampung data yang ada pada server.
f. Mengisi variabel is dengan hasil method getEntity pada objek entity

Pada baris 4 : Jika terjadi kesalahan, maka akan muncul peringatan kegagalan

Pada baris 5 : Melakukan beberapa hal, antara lain :

a. Membuat objek BufferReader dengan nama reader untuk menampung data yang dibaca. Data diambil dari variabel is.
b. Membuat objek StringBuilder.
c. Membuat variabel line dengan nilai 0.

d. Ketika variabel line yang diberi nilai dari hasil method readLine() dari objek reader bernilai tidak sama dengan null maka variabel sb akan ditambahkan dengan nilai variabel line.
e. Mengakses method close() dari object is.
f. Mengisi variabel result dengan object sb yang di konversi menjadi tipe String.

Pada baris 6 : Jika terjadi kesalahan, maka akan muncul peringatan kegagalan

Pada baris 7 : Melakukan beberapa hal, antara lain :

a. Membuat object JSONArray dengan nama jArray untuk menampung data yang diambil dari variabel result yang berbentuk JSON.
b. Membuat objek JSONObject dengan nama json_data.
c. Untuk variabel i=0 sampai kurang dari banyaknya data pada object jArray maka dilakukan :
    i. Mengisi objek json_data dengan hasil dari method getJSONObject(i) dari object jArray.
    ii. Mengisi variabel judul dengan method getString(“title”) dari objek json_data.
    iii. Mengisi variabel info dengan hasil dari method getString(“info”) dari objek json_data.
    iv. Membuat objek HasMap dengan nama map untuk mengumpulkan data.
    v. Mengakses method put dari objek map untuk memasukkan data ke objek map.
    vi. Memasukkan data objek map pada objek daftarLokasi.

Pada baris 8 : Jika terjadi kesalahan, maka akan muncul peringatan kegagalan.

Pada baris 9 : Membuat objek SimpleAdapter dengan nama infoAdapter.

Pada baris 10: Mengakses method setAdapter(infoAdapter) untuk mengisi object listInfo.

Pada baris 11 : Memberikan event onItemClickListener pada object listInfo yang melakukan :

a. Memberi nilai Array url dengan nilai dari listInfo.

b. Memberi nilai Array tittle dengan nilai dari url[0] dengan substring 8 karakter dan displit dengan tanda koma (,).
c. Menampilkan dialog box dengan judul yang berisi nilai dari object tittle dan isi yang berisi nilai dari object url.

5.7. Implementasi *Web Service*

Web service digunakan untuk melakukan komunikasi data antara aplikasi dengan web server. Pada skripsi ini metode pertukaran data menggunakan JSON (Java Script Object Notation). Fungsi JSON yang digunakan adalah JSON encode. Data yang di notasikan ke dalam JSON adalah data-data informasi gedung. Data didapatkan dari tabel json yang ada pada *database*. Data yang ditampilkan ditunjukkan dalam tabel :

No	Nama	Tipe Data	Keterangan
1	id	Integer	Identitas data, *auto increment*.
2	Latitude	String	Berisi data garis lintang lokasi
3	Longitude	String	Berisi data garis bujur
4	Elevation	String	Menyimpan data tinggi posisi
5	Tittle	String	Menampung data judul artikel
6	Distance	String	Menampung data jarak
7	Has_detail_webpage	Boolean	Menampung data indikator adanya *webpage*
8	Web_page	String url	Menampung data alamat *web*
9	info	String	Menampung data informasi gedung UB

Tabel 5.6. Data informasi gedung UB.

Hasil dari fungsi encode dapat dilihat pada gambar 5.7 :

```
[{"id":"3","latitude":"                -7.953773","longitude":"112.614681",
"elevation":"0",      "tittle":            "PTIIK",      "distance":"0",
"has_detail_webpage":"1",  "webpage":"informatika.ub.ac.id"  ,"info":"Gedung
yang digunakan untuk kegiatan perkuliah oleh 4 program studi yaitu Teknik
Informatika, Ilmu Komputer, Sistem Informasi dan Teknik Komputer."}]
```

Gambar 5.7 Implementasi JSON *encode.*

Sumber: Implementasi

5.8.Implementasi Antarmuka

Implementasi antarmuka aplikasi Informasi Gedung UB terdiri dari 2 bagian, yaitu implementasi antarmuka aplikasi *client* dan implementasi antarmuka aplikasi *adminisrator*. Implementasi antarmuka aplikasi *client*menggunakan *xml android* sedangkan implementasi antarmuka aplikasi *administrator* menggunakan html.

5.8.1. Implementasi Antarmuka Aplikasi *Client*

Antarmuka aplikasi *client* dengan memperhitungkan faktor *user friendly* sehingga pengguna dengan mudah menggunakan aplikasi Informasi Gedung UB. Tema yang digunakan dalam aplikasi ini mencerminkan UB. Motif yang digunakan diambil dari *website*resmi UB dan penggunaan gambar dan logo yang menggambarkan lingkungan UB.

a. Halaman Menu

Halaman menu berisi tombol – tombol yang mengimplementaikan menu – menu yang ada pada aplikasi Informasi Gedung UB. Pada halaman ini terdapat logo aplikasi, logo UB, logo PTIIK dan logo prodi TIF. Hal ini dimaksudkan untuk mencerminkan tema UB pada aplikasi ini. Gambar 5.8 menunjukkan implementasi antarmuka dari halaman Menu.

Gambar 5.8 Implementasi Antarmuka Halaman Menu
Sumber: Implementasi

b. Halaman Cari

Halaman cari berisi *textarea* untuk mengisikan tempat yang dicari dan tombol yang digunakan untuk melakukan proses pencarian. Gambar latar yang digunakan

adalah gedung rektorat UB dengan kaca pembesar untuk memperkuat kesan halaman pencarian dengan nuansa UB. Gambar 5.9 menunjukkan implementasi antarmuka halaman cari.

Gambar 5.9 Implementasi Antarmuka Halaman Cari.
Sumber: Implementasi

c. Halaman About

Halaman about berisi profil dari aplikasi Informasi Gedung UB. Terdapat logo dari aplikasi ini. Gambar 5.10 menunjukkan implementasi antarmuka dari halaman about.

Gambar 5.10 Implementasi Antarmuka Halaman About.
Sumber: Implementasi

d. Halaman Utama

Halaman utama merupakan halaman untuk menampilkan AR. Halaman ini mengakses fungsi kamera yang digunakan sebagai *background layout.* Pada halaman ini terdapat lingkaran radius yang dijadikan ukuran radius koordinat yang di deteksi. Gambar 5.11 menunjukkan implementasi antarmuka dari halaman utama.

Gambar 5.11 Implementasi Antarmuka Halaman Utama.
Sumber: Implementasi

e. Halaman Detail

Halaman detail berisi daftar gedung yang ada di UB. Halaman ini berfungsi untuk mendapatkan informasi gedung secara detail. Dengan menekan nama Fakultas/Jurusan yang ada pada daftar, pengguna akan mendapatkan informasi gedung secara detail melalui *message box.* Gambar 5.12 menunjukkan implementasi antarmuka dari halaman detail.

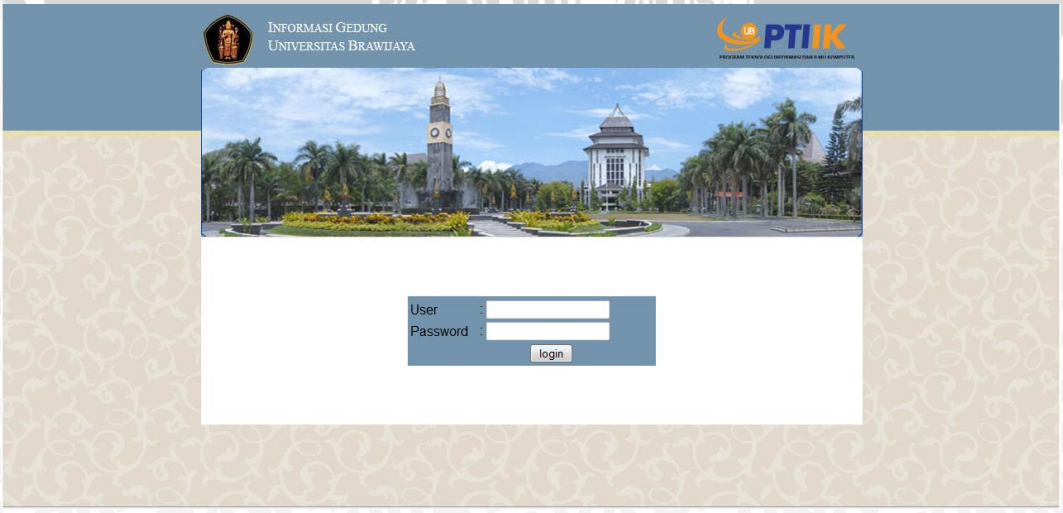
Gambar 5.12 Implementasi Antarmuka Halaman Detail.
Sumber: Implementasi

5.8.2. Implementasi Antarmuka Aplikasi *Administrator*

Antarmuka aplikasi *adminsitrator* digunakan untuk memudahkan admin untuk mengelola data informasi gedung UB. Apliksi administrator dibuat dalam bentuk web menggunakan html dan php.

a. Halaman Login

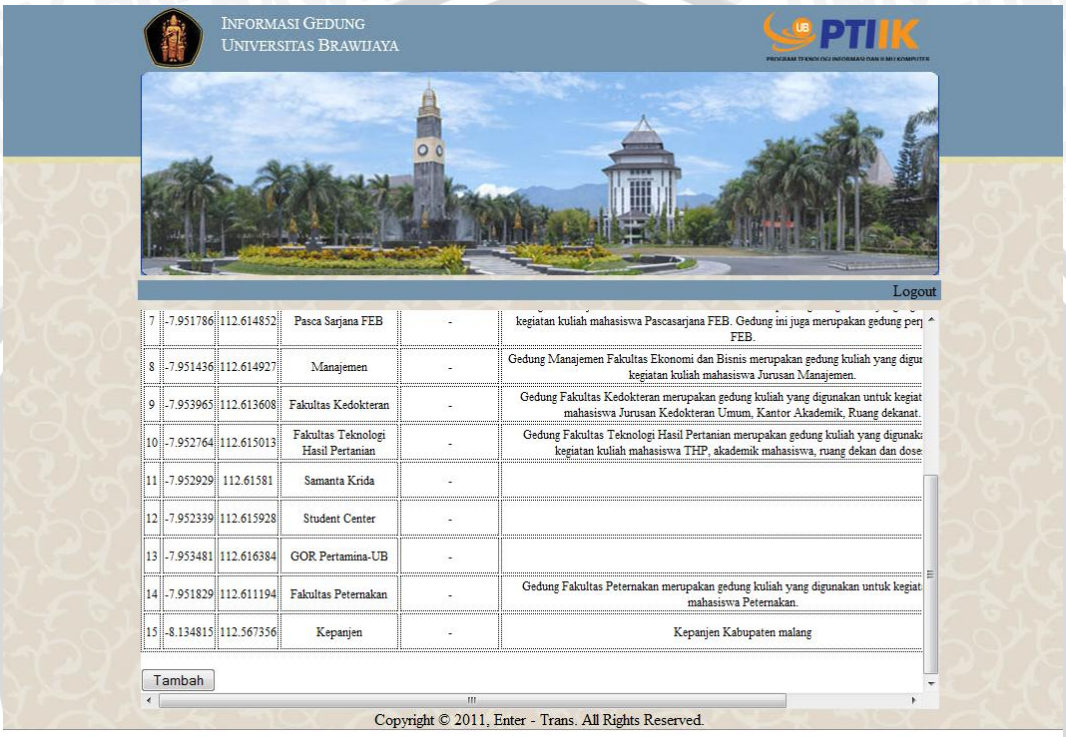
Halaman login merupakan halaman yang berfungsi untuk mempermudah *administrator* dalam melakukan login. Gambar 5.13 akan menunjukkan implementasi antarmuka dari halaman login.



Gambar 5.13 Implementasi Antarmuka Halaman Login.
Sumber: Implementasi

b. Halaman Utama

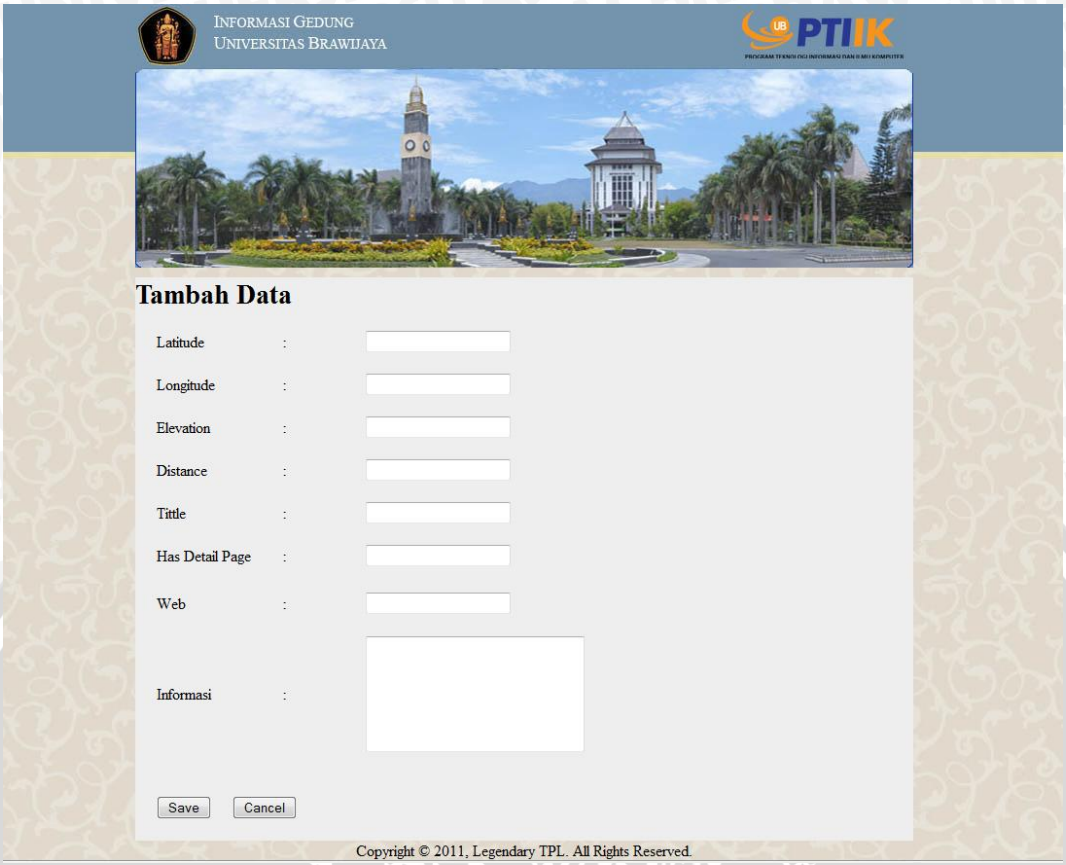
Halaman utama merupakan halaman yang pertama kali diakses setelah sukses melakukan login. Halaman ini berisi semua data informasi gedung yang ada di *database*. Gambar 5.14 menunjukkan implementasi antarmuka dari halaman utama.



Gambar 5.14 Implementasi Antarmuka Halaman Utama.
Sumber: Implementasi

c. Halaman Tambah Data

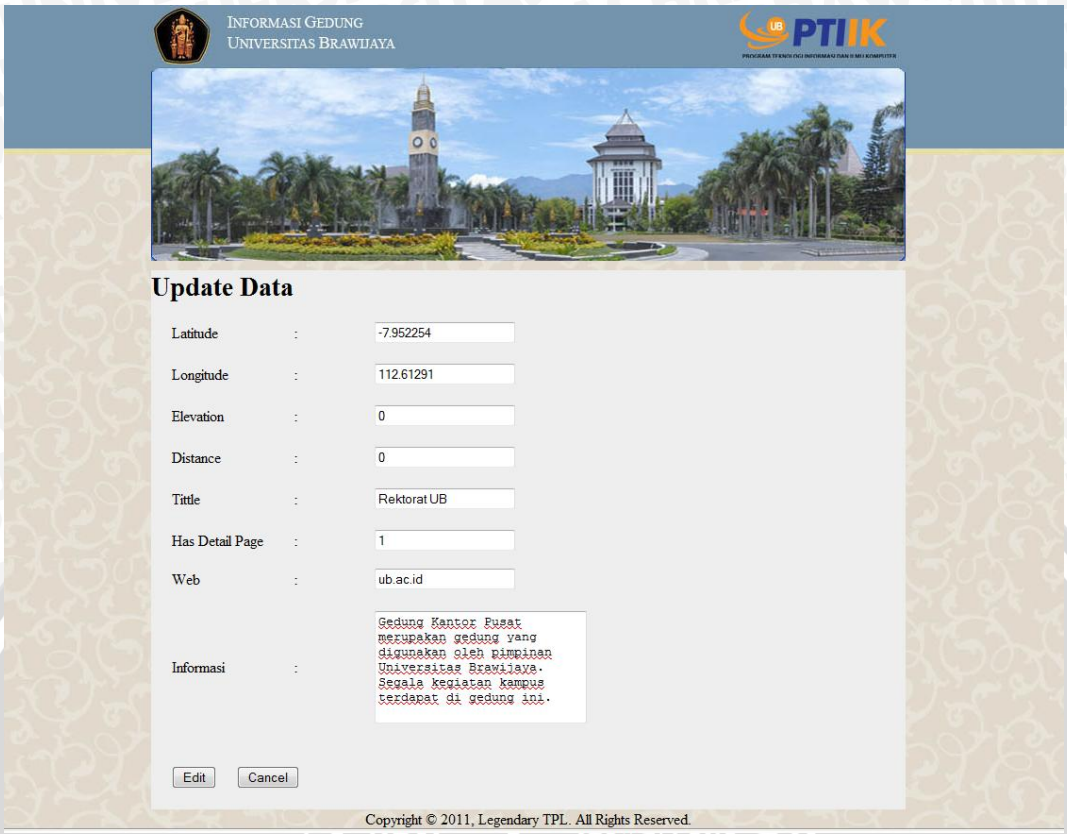
Halaman tambah data merupakan halaman yang digunakan untuk menambahkan data ke database. Halaman ini berisi *textbox* yang digunakan untuk mengisikan data, tombol "save" yang digunakan untuk melakukan proses memasukkan data dan tombol "cancel" untuk kembali ke halaman utama. Gambar 5.15 menunjukkan implementasi antarmuka dari halaman tambah data.



Gambar 5.15 Implementasi Antarmuka Halaman Tambah Data
Sumber: Implementasi

d. Halaman Ubah Data

Halaman ubah data merupakan halaman yang digunakan untuk mengubah data yang ada pada*database*. Halaman ini berisi *textbox* yang digunakan untuk mengubah data, tombol "Edit" yang digunakan untuk melakukan proses memasukkan data dan tombol "cancel" untuk kembali ke halaman utama. Gambar 5.16 menunjukkan implementasi antarmuka dari halaman ubah data.



Gambar 5.16 Implementasi Antarmuka Halaman Ubah Data.
Sumber: Implementasi

BAB VI
PENGUJIAN DAN ANALISIS

Bab ini membahas mengenai tahapan pengujian dan analisis perangkat lunak Informasi Gedung UByang telah dikembangkan. Proses pengujian dilakukan melalui tiga tahapan (strategi) yaitu pengujian unit, pengujian validasi, dan pengujian performa. Pada pengujian unit akan digunakan teknik pengujian *White-Box* (*White-Box Testing*). Pada pengujian validasi dan performa akan digunakan teknik pengujian *Black-Box* (*Black-Box Testing*).

6.1. Pengujian

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

6.1.1. Pengujian Unit

Perangkat lunak yang dikembangkan dengan paradigma *object-oriented programming* menerapkan pengujian unit untuk suatu *method* (operasi) dari suatu *class*. Pada pengujian unit perangkat lunak Informasi Gedung UBdigunakan teknik *White-Box Testing*dengan teknik *Basis Path Testing*. Pada teknik *Basis Path Testing*, proses pengujian dilakukan dengan memodelkan algoritma pada suatu *flow graph*, menentukan jumlah kompleksitas siklomatis *(cyclomatic complexity)*, menentukan sebuah basis set dari jalur independen dan memberikan kasus uji *(test case)* pada setiap basis set yang telah ditentukan.

1. Pengujian Unit Algoritma Mencari Informasi Gedung

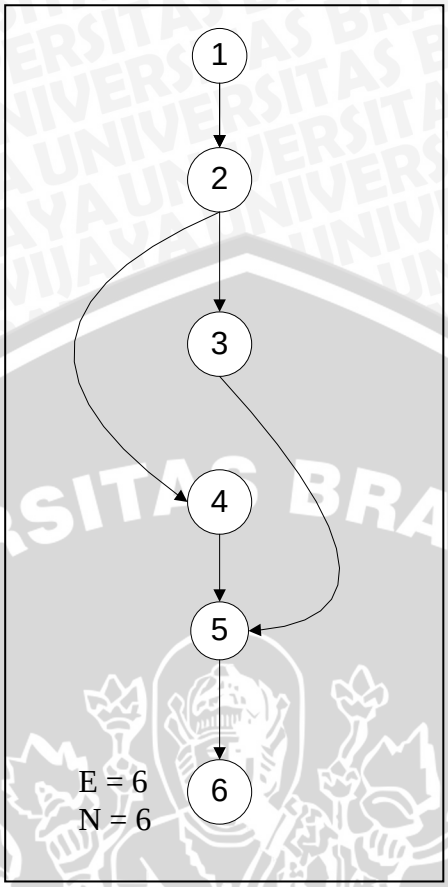
Fungsi ini untuk melakukan pencarian informasi gedung UB. Algoritma mencari informasi gedung terdapat pada kelas cariData. Gambar 6.1 menggambarkan proses pengujian unit dari algoritma mencari informasi gedung dan gambar 6.2 menunjukkan pemodelan *flow graph* pada algoritma mencari informasi gedung.

```
NAMA ALGORITMA Mencari informasi gedung

DEKLARASI
edit : EditText
btnImg : ImageButton
url, key, search : String
i : Intent
DESKRIPSI
Proses :
  1. run btnImg.setOnClickListener(new onClickListener ())
  2. url = edit.getText().toString();
  3. Intent   i   =   new   Intent(getApplicationContext(),
     MixView.class);
  4. i.putExtra(key,url).
  5. run starActivityForResult(i,0).
  6. Class mixview run.
  7. If (extras not null) then
      a. search = extras.
  8. else :
      a. search = null.
  9. Request            server          "http://antonio-
     shoes.com/infogedungub/json.php?cari="+search+""
  10. Get JSON data.
Keluaran :
  1. View data from server.
```

1
2
3
4
5
6

Gambar 6.1 Pengujian Unit algoritma mencari informasi gedung.
Sumber : Pengujian dan Analisis



Gambar 6.2*Flow graph* algoritma mencari informasi gedung
Sumber : Pengujian dan Analisis

Pemodelan ke dalam *flow graph* yang telah dilakukan terhadap algoritma mencari informasi gedung menghasilkan jumlah kompleksitas siklomatis *(cyclomatic complexity)* melalui persamaan V(G) = E – N + 2, dimana V(G) merupakan jumlah kompleksitas siklomatis, E merupakan sisi atau *edge* (garis penghubung antar *node*) dan N merupakan jumlah simpul *(node).*

V(G) = E – N + 2
= 6 – 6 + 2
= 2

Berdasarkan dari nilai *cyclomatic complexity* yang telah didapatkan dari perhitungan maka ditentukan dua buah basis set dari jalur *independent*, yaitu :

Jalur 1 : 1 – 2 – 3 – 5 – 6

Jalur 2 : 1 – 2 – 3 –4 – 5 – 6

Jalur	Kasus Uji	Hasil yang diharapkan	Hasil yang didapatkan
1	Variabel search tidak kosong	Data akan ditampilkan sesuai dengan variabel search.	Data akan ditampilkan sesuai dengan variabel search.
2	Variabel search kosong	Tidak ada yang ditampilkan	Tidak ada yang ditampilkan

Tabel 6.1 Kasus uji untuk pengujian unit algoritma mencari informasi gedung
Sumber : Pengujian dan Analisis

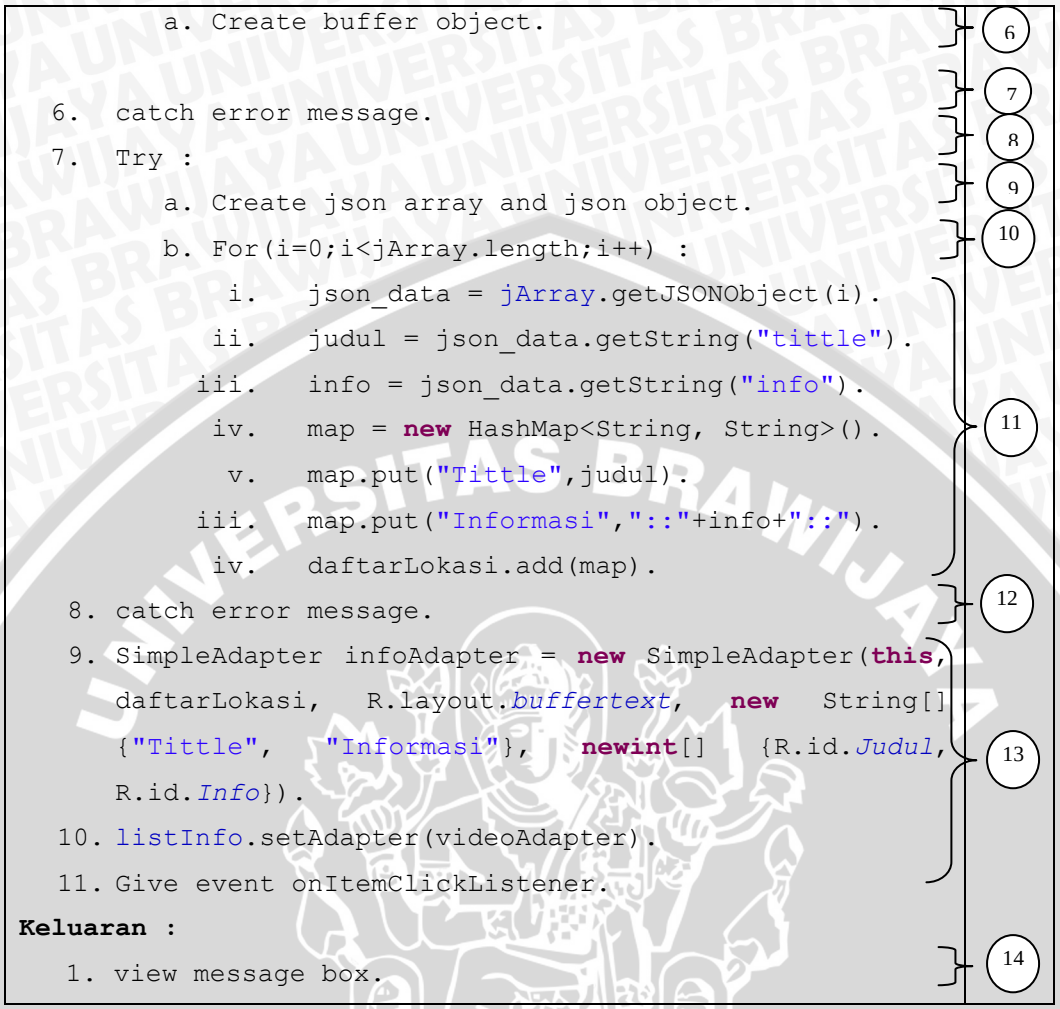
2. Pengujian Unit Algoritma Melihat Detail Informasi Gedung

Fungsi ini untuk melihat detail informasi gedung UB. Algoritma melihat detail informasi gedung terdapat pada kelas detailInformasi. Gambar 6.3 menggambarkan proses pengujian unit dari algoritma melihat detail informasi gedung dan gambar 6.4 menunjukkan pemodelan *flow graph* pada algoritma melihat detail informasi gedung.

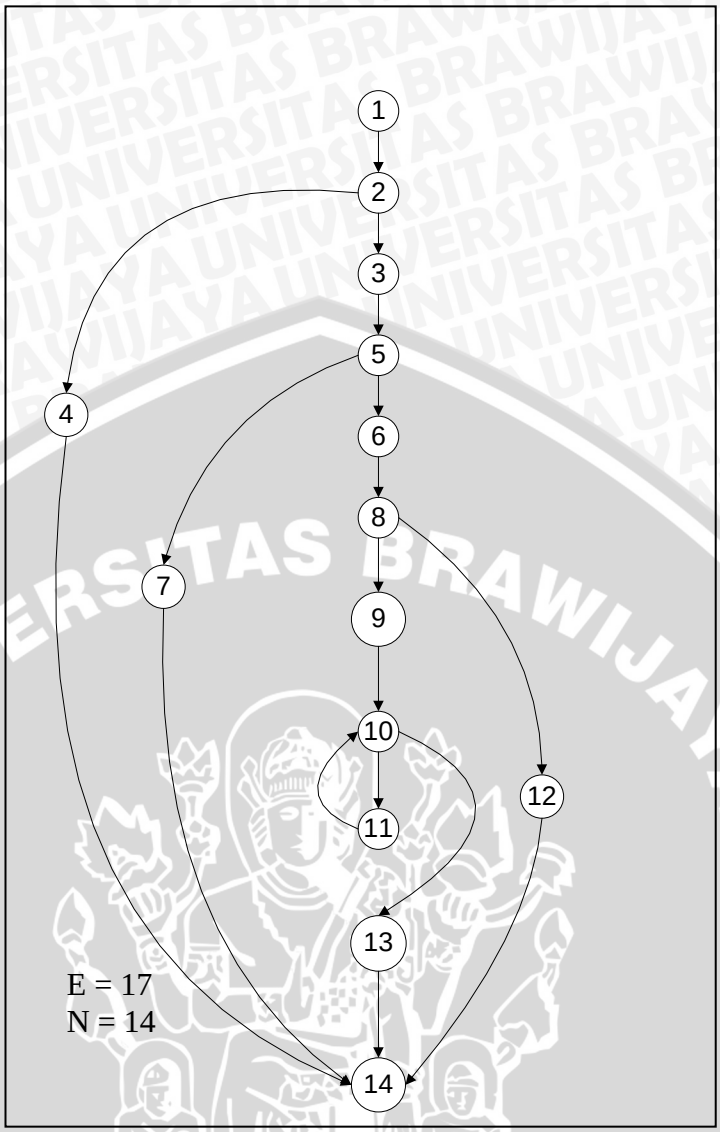
```
NAMA ALGORITMA melihat detail informasi gedung

DEKLARASI
is : InputStream
result : String
jArray : JSONArray
sb : StringBuilder
listInfo : ListView
judul, info : String
map : HashMap<String, String>
DESKRIPSI
Proses :
  1. class detailInformasi run.
  2. nameValuePairs = new arrayList.
  3. try :
       b. create http connection.
  4. catcherror message.
  5. Try :
```

1
2
3
4
5



Gambar 6.3 Pengujian Unit algoritma melihat detail informasi gedung.
Sumber : Pengujian dan Analisis



Gambar 6.4*Flow graph* algoritma melihat detail informasi gedung
Sumber : Pengujian dan Analisis

Pemodelan ke dalam *flow graph* yang telah dilakukan terhadap algoritma melihat detail informasi gedung menghasilkan jumlah kompleksitas siklomatis *(cyclomatic complexity)* melalui persamaan V(G) = E – N + 2, dimana V(G) merupakan jumlah kompleksitas siklomatis, E merupakan sisi atau *edge* (garis penghubung antar *node*) dan N merupakan jumlah simpul *(node).*

V(G) = E – N + 2
= 17 – 14 + 2
= 5

Berdasarkan dari nilai *cyclomatic complexity* yang telah didapatkan dari perhitungan maka ditentukan lima buah basis set dari jalur *independent*, yaitu :

Jalur 1 : 1 – 2 – 3 – 5 – 6 – 8 – 9 – 10 – 13 – 14

Jalur 2 : 1 – 2 – 3 – 5 – 6 – 8 – 9 – 10 – 11 – 10 – 13 – 14

Jalur 3 : 1 – 2 – 4 – 14

Jalur 4 : 1 – 2 – 3 – 5 – 7 – 14

Jalur 5 : 1 – 2 – 3 – 5 – 6 – 8 – 12 – 14

Jalur	Kasus Uji	Hasil yang diharapkan	Hasil yang didapatkan
1	Tidak memenuhi syarat perulangan	Data tidak muncul	Data tidak muncul
2	Memenuhi syarat perulangan	Data yang muncul sesuai dengan data yang ada pada *database* yang ada di *server*	Data yang muncul sesuai dengan data yang ada pada *database* yang ada di *server*
3	Terjadi kesalahan sistem pada saat membuat koneksi	Muncul pesan eror	Mincul pesan eror
4	Terjadi kesalahan sistem pada saat penampungan data.	Muncul pesan eror	Muncul pesan eror
5	Terjadi kesalahan sistem pada saat membuat object *Array*	Muncul pesan eror	Muncul pesan eror

Tabel 6.2 Kasus uji pengujian unit algoritma melihat informasi gedung
Sumber : Pengujian dan Analisis

6.1.2. Pengujian Validasi

Pengujian validasi digunakan untuk mengetahui apakah sistem yang dibangun sudah benar sesuai dengan yang dibutuhkan. *Item - item* yang telah dirumuskan dalam daftar kebutuhan 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 konformitas antara kinerja sistem dengan daftar kebutuhan. Pada skripsi ini dilakukan pengujian validasi terhadap perangkat lunak Informasi Gedung UB.

1. Kasus Uji Validasi Aplikasi Client

a. Kasus Uji untuk Menampilkan Informasi Gedung dengan AR sebagai Tampilan dalam *Mobile Device.*

Nama Kasus Uji	Kasus Uji Menampilkan Informasi Gedung dengan *Augmented Reality* sebagai Tampilan dalam *Mobile Device*
Objek Uji	SRS-C001
Tujuan Pengujian	Pengujian dilakukan untuk memastikan bahwa aplikasi dapat memenuhi kebutuhan fungsional untuk menampilkan objek *augmented reality* pada layar.
Prosedur Uji	1. Aplikasi dijalankan 2. Memilih menu "aplikasi" 3. Mengarahkan *device* ke gedung yang ada di lingkungan UB
Hasil yang Diharapakan	Objek AR akan muncul dalam layar yang beisi informasi gedung tersebut

Tabel 6.3 Kasus uji untuk pengujian validasi menampilkan informasi gedung dengan *augmented reality* sebagai tampilan dalam *mobile device*
Sumber : Pengujian dan Analisis

b. Kasus Uji untukMencari Informasi Gedung dengan *Augmented Reality* sebagai Tampilan dalam *Mobile Device.*

Nama Kasus Uji	Kasus Uji Mencari Informasi Gedung dengan *Augmented Reality* sebagai Tampilan dalam *Mobile Device*
Objek Uji	SRS-C002
Tujuan Pengujian	Pengujian dilakukan untuk memastikan bahwa aplikasi dapat memenuhi kebutuhan fungsional untuk

	menampilkan objek *augmented reality* pada layar sesuai gedung yang dicari.
Prosedur Uji	1. Aplikasi dijalankan 2. Memilih menu "cari" 3. Memasukkan nama gedung yang dicari
Hasil yang Diharapakan	Objek AR akan muncul dalam layar sesuai dengan nama gedung yang dicari

Tabel 6.4 Kasus uji untuk pengujian validasi mencari informasi gedung dengan *augmented reality* sebagai tampilan dalam *mobile device*
Sumber : Pengujian dan Analisis

c. Kasus Uji untuk Menampilkan Informasi DetailGedung yang Dipilih.

Nama Kasus Uji	Kasus Uji Menampilkan Informasi Detail tentang Gedung yang Dipilih
Objek Uji	SRS-C003
Tujuan Pengujian	Pengujian dilakukan untuk memastikan bahwa aplikasi dapat memenuhi kebutuhan fungsional untuk menampilkan detail informasi dari gedung
Prosedur Uji	1. Aplikasi dijalankan 2. Memilih menu "cari" atau aplikasi 3. Memilih menu "detail" 4. Memilih salah satu gedung yang akan dilihat informasinya secara detail.
Hasil yang Diharapakan	Muncul *message box* yang berisi informasi gedung secara detail

Tabel 6.5 Kasus uji untuk pengujian validasi menampilkan informasi detail gedung yang dipilih
Sumber : Pengujian dan Analisis

d. Kasus Uji untuk Menampilkan Halaman *About.*

Nama Kasus Uji	Kasus Uji Menampilkan Halaman *About*
Objek Uji	SRS-C004

Tujuan Pengujian	Pengujian dilakukan untuk memastikan bahwa aplikasi dapat memenuhi kebutuhan fungsional untuk menampilkan halaman a*bout*
Prosedur Uji	1. Aplikasi dijalankan 2. Memilih menu "about"
Hasil yang Diharapakan	Menampilkan halaman *about*

Tabel 6.6 Kasus uji untuk pengujian validasi menampilkan halaman *about*

Sumber : Pengujian dan Analisis

2. Kasus Uji Validasi Aplikasi *Administrator*

a. Kasus Uji untuk *Login* Sukses

Nama Kasus Uji	Kasus Uji *Login* Sukses
Objek Uji	SRS-A001
Tujuan Pengujian	Pengujian dilakukan untuk memastikan bahwa aplikasi dapat memenuhi kebutuhan fungsional menyeleksi pengguna yang sukses melakukan *login*
Prosedur Uji	1. Akses *web* 2. Memasukkan *username* dan *password*yang sesuai dengan data yang ada di *database* 3. Tekan tombol "login"
Hasil yang Diharapakan	*User*berhasil *login* dan masuk halaman utama

Tabel 6.7 Kasus uji untuk pengujian validasi untuk *login* sukses

Sumber : Pengujian dan Analisis

b. Kasus Uji untuk Login Gagal

Nama Kasus Uji	Kasus Uji *Login* Gagal
Objek Uji	SRS-A001
Tujuan Pengujian	Pengujian dilakukan untuk memastikan bahwa aplikasi dapat memenuhi kebutuhan fungsional menyeleksi pengguna yang gagal melakukan *login*
Prosedur Uji	1. Akses *web*

	2. Memasukkan *username* dan *password*yang tidak sesuai dengan data yang ada di *database* 3. Tekan tombol "login"
Hasil yang Diharapakan	*User*gagal *login* dan muncul peringatan

Tabel 6.8 Kasus uji untuk pengujian validasi*login* gagal
Sumber : Pengujian dan Analsisi

c. Kasus Uji untuk Menambah Data di *Database*

Nama Kasus Uji	Kasus Uji Menambah Data di *Database*
Objek Uji	SRS-A002
Tujuan Pengujian	Pengujian dilakukan untuk memastikan bahwa aplikasi dapat memenuhi kebutuhan fungsional menambahkan informasi di *database*
Prosedur Uji	1. Akses *web* 2. Masuk halaman utama 3. Tekan tombol "tambah" 4. Masukkan data pada halaman tambah 5. Tekan tombol "tambah"
Hasil yang Diharapakan	Data berhasil ditambahkan pada *database*

Tabel 6.9 Kasus uji untuk pengujian validasi menambah data di *database*
Sumber : Pengujian dan Analisis

d. Kasus Uji untuk Menghapus Data di *Database*

Nama Kasus Uji	Kasus Uji Menghapus Data di *Database*
Objek Uji	SRS-A003
Tujuan Pengujian	Pengujian dilakukan untuk memastikan bahwa aplikasi dapat memenuhi kebutuhan fungsional menghapus informasi di *database*
Prosedur Uji	1. Akses *web* 2. Masuk halaman utama 3. Pilih "hapus" pada tabel

Hasil yang Diharapakan	Data berhasil dihapus pada *database*

Tabel 6.10 Kasus uji untuk pengujian validasi menghapus data di *database*
Sumber : Pengujian dan Analisis

e. Kasus Uji untuk Mengubah Data di *Database*

Nama Kasus Uji	Kasus Uji Mengubah Data di *Database*
Objek Uji	SRS-A004
Tujuan Pengujian	Pengujian dilakukan untuk memastikan bahwa aplikasi dapat memenuhi kebutuhan fungsional mengubah informasi di *database*
Prosedur Uji	1. Akses *web* 2. Masuk halaman utama 3. Pilih "edit" pada tabel 4. Ubah data yang dirubah 5. Tekan tombol "ubah"
Hasil yang Diharapakan	Data berhasil dirubah pada *database*

Tabel 6.11 Kasus uji untuk pengujian validasi mengubah data di *database*
Sumber : Pengujian dan Analisis

3. Hasil Uji Validasi Aplikasi *Client*

No	Nama Kasus Uji	Hasil Yang Diharapkan	Hasil Yang Didapatkan	Status Valid
1.	Kasus Uji Menampilkan Informasi Gedung dengan *Augmented Reality* sebagai Tampilan dalam *Mobile Device*	Objek AR akan muncul dalam layar yang beisi informasi gedung tersebut	Objek AR muncul dalam layar yang beisi informasi gedung tersebut	Valid
2.	Kasus Uji MencariInformasi	Objek AR akan muncul dalam	Objek AR muncul dalam	Valid

	Gedung dengan *Augmented Reality* sebagai Tampilan dalam *Mobile Device*	layar sesuai dengan nama gedung yang dicari	layar sesuai dengan nama gedung yang dicari	
3.	Kasus Uji Menampilkan Informasi Detail tentang Gedung yang Dipilih	Muncul *message box* yang berisi informasi gedung secara detail	Muncul *message box* yang berisi informasi gedung secara detail	Valid
4.	Kasus Uji Menampilkan Halaman *About*	Menampilkan halaman *about*	Menampilkan halaman *about*	Valid

Tabel 6.12 Hasil uji validasi aplikasi *client*
Sumber : Pengujian dan Analisis

4. Hasil Uji Validasi Aplikasi *Administrator*

No	Nama Kasus Uji	Hasil Yang Diharapkan	Hasil Yang Didapatkan	Status Valid
1.	Kasus Uji *Login* Sukses	*User*berhasil *login* dan masuk halaman utama	*User*berhasil *login* dan masuk halaman utama	Valid
2.	Kasus Uji *Login* Gagal	*User*gagal *login* dan muncul peringatan	*User*gagal *login* dan muncul peringatan	Valid
3.	Kasus Uji Menambah Data di *Database*	Data berhasil ditambahkan pada *database*	Data berhasil ditambahkan pada *database*	Valid
4.	Kasus Uji Menghapus Data di *Database*	Data berhasil dihapus pada *database*	Data berhasil dihapus pada *database*	Valid
5.	Kasus Uji	Data berhasil	Data berhasil	Valid

	Mengubah Data di *Database*	dirubah pada *database*	dirubah pada *database*	

Tabel 6.13 Hasil uji validasi aplikasi admin
Sumber : Pengujian dan Analisis

6.1.3 Pengujian Performa

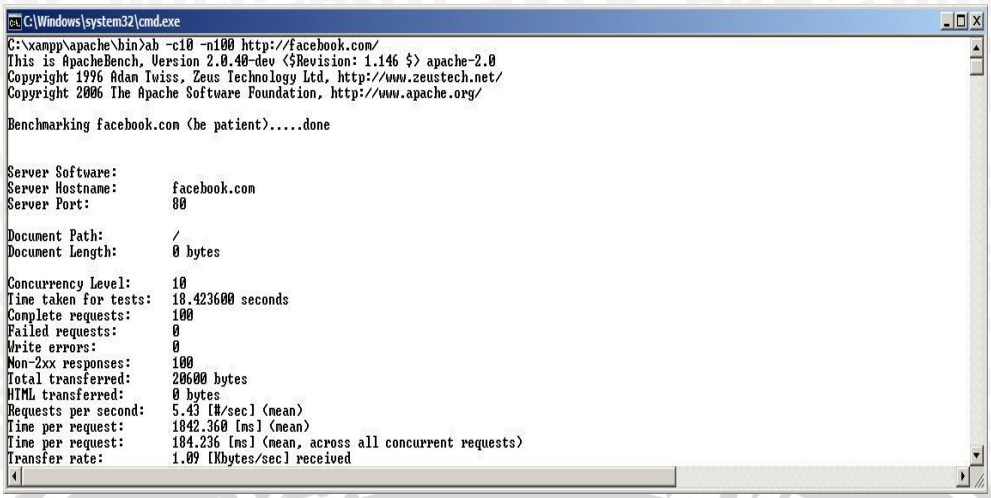
Pengujian performa dilakukan untuk mengetahui bagaimana performa *server* dari aplikasi dalam mengatasi banyaknya *request* dalam satuan waktu yang di lakukan secara bersamaan dan dalam jumlah tertentu. Pengujian performa dilakukan menggunakan *tool* yang disebut apache *benchmark.*

Apache *benchmark* adalah *tool* untuk mengukur performa dari apache*server* dalam melayani *request* dari *client*.Adapun sintak yang digunakan dalam apache *benchmark*ditunjukan pada Gambar 6.5.

```
ab -c 10 -n 100[url]
```

Gambar 6.5 Sintak apache *benchmark*
Sumber : [APA-12]

Sintak pada Gambar 6.5 memiliki beberapa parameter antara lain `ab`, `-c`, `-n`, dan `url`. Parameter `ab`merupakan singkatan dari *Apache Benchmark*. Parameter `-c`10 adalah jumlah *concurrent connection*, sebanyak 10 koneksi secara bersamaan yang dibuat dalam satu waktu. Parameter `-n`100 adalah jumlah *request* yang dibuat ke server tujuan, sebanyak 100 *request.* Parameter terakhir adalah `url`, alamat halaman yang akan diproses oleh apache*benchmark*. Gambar 6.6 menunjukkan antarmuka apache *benchmark* saat dijalankan.



Gambar 6.6 Tampilan apache *benchmark*saat dijalankan

Sumber : [Pengujian]

Pengujian dilakukan pada jaringan yang dimiliki oleh Teknik Informatika Universitas Brawijaya melalui dua tahap, yaitu pengujian performa *server* yang berada pada jaringan *intranet* dan pada *server* yang di-*hosting* secara *online*. Dalam pengujian ini obyek yang digunakan adalah *fileweb service* json.php yang ada pada *server* dengan alamat IP 172.21.3.200 dan pada *server online* dengan alamat infogedungub.webatu.com, sehingga nanti akan diketahui *throughput* dari *request* ke *web service* tersebut. Sintak yang digunakan dalam pengujian performasi *server* pada jaringan internet menggunakan apache *benchmark* adalah :

```
ab  -c 10 -n  100 http://infogedungub.webatu.com/json.com
```

Gambar 6.7 Sintak apache *benchmark*pada jaringan *internet*

Sumber : Pengujian dan Analisis

Sintak yang digunakan dalam pengujian performasi *server* pada jaringan intranet menggunakan apache *benchmark* adalah :

```
ab  -c 10 -n  100 http://172.21.3.200/infogedungub/json.com
```

Gambar 6.8 Sintak apache *benchmark*pada jaringan *intranet*

Sumber : Pengujian dan Analisis

Percobaan dilakukan dengan menggunakan jumlah *request* yang tetap

namun dengan jumlah *concurrency* yang berbeda.Spesifikasi *hadware* komputer yang digunakan sebagai *server*dalam jaringan *intranet* dalam pengujian ini dijelaskan pada Tabel 6.14.

Tabel 6.14 Spesifikasi perangkat keras *server* dalam percobaan

Nama Komponen	Spesifikasi
Prosesor	Intel® Core™ 2 Duo CPU P8400 @ 2.26*GHz*
Memory (RAM)	4 *GB*
Hardisk	Toshiba MK1652GSX, kapasitas 160 *GB*, 5400 *rpm*
Mother Board	Toshiba Satellite Pro U400
Kartu Grafis	Mobile Intel® GMA 4500MHD

Sumber: Pengujian dan Analisis

Pada pengujian yang pertama, percobaan dilakukan dengan menggunakan 100 *request* dengan jumlah *concurrency* yang berbeda yaitu mulai dari 5 hingga 80 dengan interval 2n pada lingkungan jaringan Teknik Informatika Universitas Brawijaya. Tabel 6.15 menunjukkan hasil pengujian apache *benchmark* dengan 100 *request*pada jaringan *intranet*.

Tabel 6.15 Hasil percobaan apache *benchmark* dengan 100 *request pada intranet*

Request	*Concurrency*	*Request / Second*	*Transfer Rate*
100	5	42,45	137,93 KBps
100	10	78,17	253,99 KBps
100	20	25,14	81,67 KBps
100	40	72,03	234,01 KBps
100	80	57,23	185,96 KBps
Rata-rata		55,004	178,712 KBps

Sumber: Pengujian dan Analisis

Berdasarkan kasus uji pertama pada Tabel 6.15didapatkan hasil sebagai berikut :

a. Nilai *transfer rate* minimum sebesar 81,67,04KBps dengan jumlah *request persecond*yang dapat dilayani *server* sebesar25,14 *request*.

b. Nilai *transfer rate* maksimum sebesar 253,99KBps dengan jumlah *request persecond*yang dapat dilayani *server* sebesar78,17 *request*.
c. Dari kelima percobaan yang dilakukan didapatkan nilai *transfer rate* rata – rata adalah sebesar 178,712KBps dengan dengan *request per second* rata-rata sebesar 55,004*request*.

Pengujian dilanjutkan dengan mengukur performa *server* yang ada pada *hosting online*, percobaan dilakukan dengan menggunakan 100 *request* dengan jumlah *concurrency* yang berbeda-beda. Tabel 6.16 menunjukkan hasil pengujian apache *benchmark* dengan 100 *request*pada jaringan *Internet*.

Tabel 6.16 Hasil percobaan apache *benchmark* dengan 100 *request pada Internet*

Request	*Concurrency*	*Request / Second*	*Transfer Rate*
100	5	2,06	6,26 KBps
100	10	2,19	6,65 KBps
100	20	2,03	6,15 KBps
100	40	2,05	6,21 KBps
100	80	1,35	4,11 KBps
Rata-rata		1,94	5,89 KBps

Sumber: Pengujian dan Analisis

Berdasarkan kasus uji pada Tabel 6.16didapatkan hasil sebagai berikut :

a. Nilai *transfer rate* minimum sebesar 4,11KBps dengan jumlah *request persecond*yang dapat dilayani *server* sebesar1,35 *request*.
b. Nilai *transfer rate* maksimum sebesar 6,65KBps dengan jumlah *request persecond*yang dapat dilayani *server* sebesar2,19 *request*.
c. Dari kelima percobaan yang dilakukan didapatkan nilai *transfer rate* rata – rata adalah sebesar 5,89KBps dengan dengan *request per second* rata-rata sebesar 1,94*request*.

Kasus uji selanjutnya adalah dengan menggunakan jumlah *request* yang

lebih banyak yaitu 200 *request* dengan jumlah *concurrency* yang berbeda yaitu mulai dari 5 hingga 80 dengan interval 2n. Tabel 6.17 menunjukkan hasil pengujian *apache benchmark* dengan 200*request*pada jaringan *intranet*.

Tabel 6.17 Hasil percobaan apache *benchmark* dengan 200 *request*pada *intranet*

Request	*Concurrency*	*Request / Second*	*Transfer Rate*
200	5	85,47	277,69 KBps
200	10	117,62	382,15 KBps
200	20	62,24	202,2 KBps
200	40	86,62	281,45 KBps
200	80	119,82	389,29 KBps
Rata-rata		94,354	306,556 KBps

Sumber: Pengujian dan Analisis

Berdasarkan kasus uji keduapada Tabel 6.17didapatkan hasil sebagai berikut :

a. Nilai *transfer rate* minimum sebesar 202,2KBps dengan jumlah *request persecond*yang dapat dilayani *server* sebesar62,24 *request*.
b. Nilai *transfer rate* maksimum sebesar 389,29KBps dengan jumlah *request persecond*yang dapat dilayani *server* sebesar119,82 *request*.
c. Dari kelima percobaan yang dilakukan, didapatkan nilai *transfer rate* rata – rata adalah sebesar 306,556KBps dengan dengan *request per second* rata-rata sebesar 94,354*request*.

Pengujian dilanjutkan dengan mengukur performa *server* yang ada pada *hosting online*, percobaan dilakukan dengan menggunakan 200 *request* dengan jumlah *concurrency* yang berbeda-beda. Tabel 6.18 menunjukkan hasil pengujian apache *benchmark* dengan 200 *request*pada jaringan *Internet*.

Tabel 6.18 Hasil percobaan apache *benchmark* dengan 200 *request*pada *Internet*

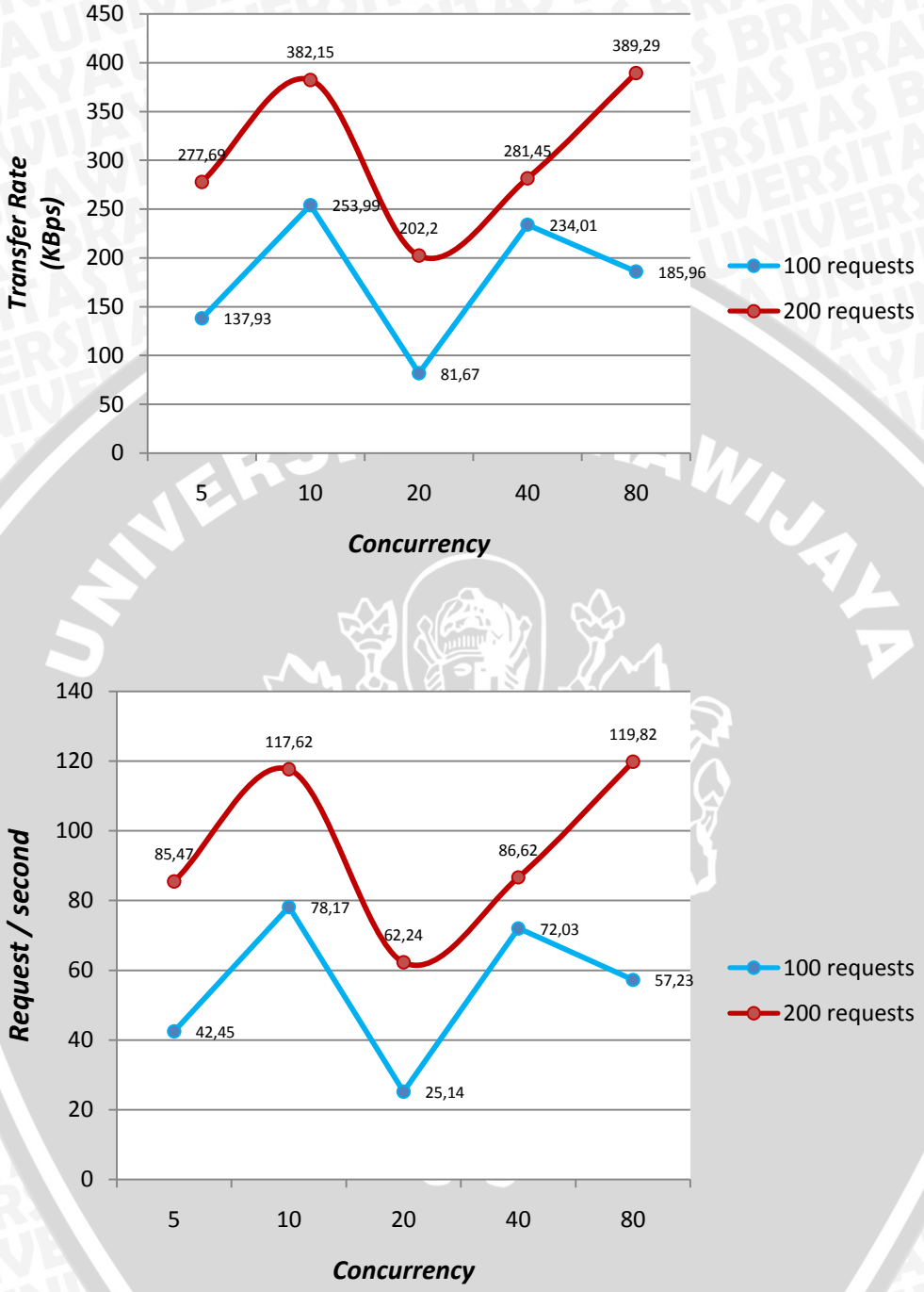
Request	*Concurrency*	*Request / Second*	*Transfer Rate*
200	5	2,26	7,97 KBps
200	10	3,11	9,39 KBps
200	20	3,27	9,93 KBps
200	40	3,46	10,44 KBps
200	80	0	0 KBps
Rata-rata		2,492	7,546 KBps

Sumber: Pengujian dan Analisis

Berdasarkan kasus uji keduapada Tabel 6.18didapatkan hasil sebagai berikut :

a. Nilai *transfer rate* minimum sebesar 0KBps dengan jumlah *request persecond*yang dapat dilayani *server* sebesar0 *request*.
b. Nilai *transfer rate* maksimum sebesar 10,44KBps dengan jumlah *request persecond*yang dapat dilayani *server* sebesar3,46 *request*.
c. Dari kelima percobaan yang dilakukan, didapatkan nilai *transfer rate* rata – rata adalah sebesar 7,546KBps dengan dengan *request per second* rata-rata sebesar 2,492*request*.

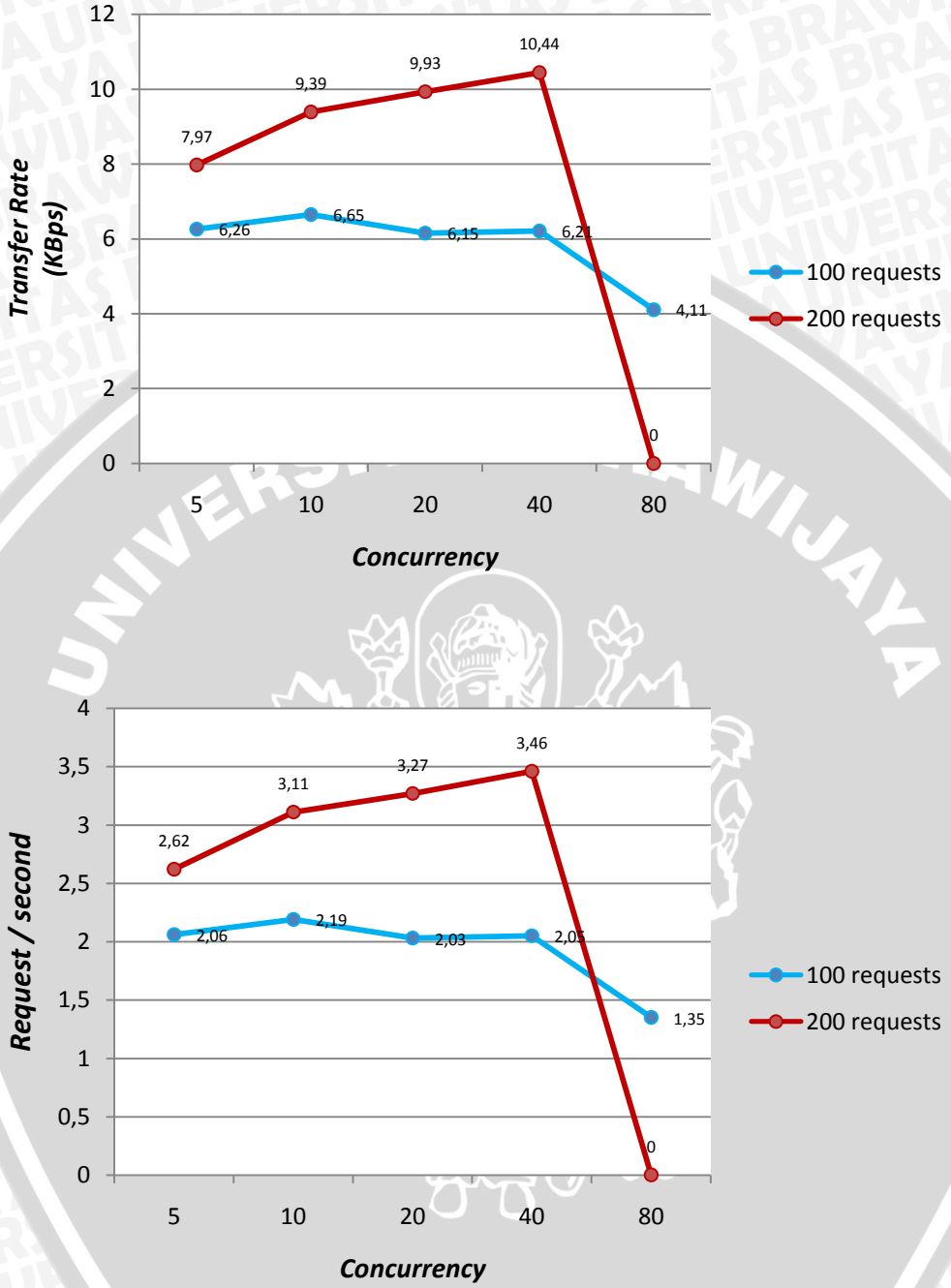
Dari percobaan yang dilakukan dalam lingkungan jaringan Teknik Informatika Universitas Brawijaya, hasil yang didapatkan dapat dideskripsikan ke dalam grafik yang menunjukkan hubungan antara *concurrency* dengan *transfer rate* dan *concurrency* dengan *request per second*. Gambar 6.9 menunjukkan grafik performa *server* pada jaringan *intranet*.



Gambar 6.9 Grafik hasil pengujian pada jaringan *intranet*

Sumber: Pengujian dan Analisis

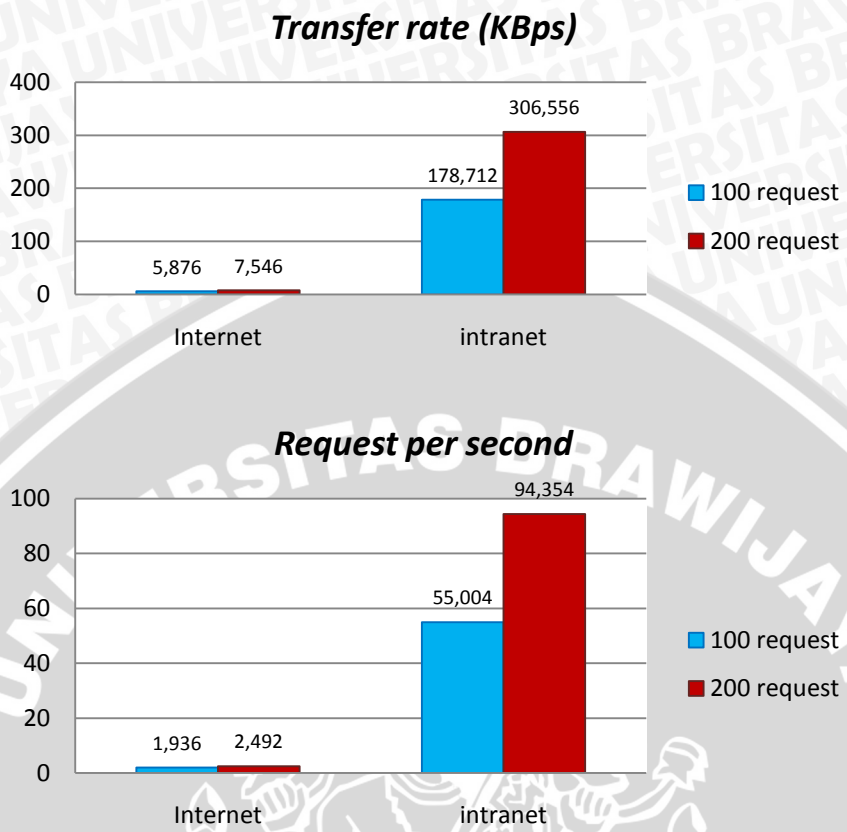
Grafik dari hasil pengujian yang dilakukan pada *server* yang di *hosting* secara *online* dan diakses dari jaringan *Internet* Teknik Informatika Universitas Brawijaya dapat dilihat pada Gambar 6.10.



Gambar 6.10 Grafik hasil pengujian pada jaringan *Internet*

Sumber: Pengujian dan Analisis

Diagram dari rata-rata performa*server* pada jaringan *intranet* dan *Internet* dapat dilihat pada Gambar 6.10.



Gambar 6.11 Diagram rata-rata hasil pengujian

Sumber: Pengujian dan Analisis

6.2. Analisis

Proses analisis bertujuan untuk mendapatkan kesimpulan dari hasil pengujian perangkat lunak Informasi Gedung UByang telah dilakukan. Proses analisis mengacu pada dasar teori sesuai dengan hasil pengujian yang didapatkan. Analisis dilakukan terhadap hasil pengujian di setiap tahap pengujian. Proses analisis yang dilakukan meliputi analisis hasil pengujian unit, analisis hasil pengujian validasi, dan analisis hasil pengujian performa.

6.2.1. Analisis Hasil Pengujian Unit

Proses analisis terhadap hasil pengujian unit dilakukan dengan melihat kesesuaian fungsi dari implementasi unit modul yang diuji dengan hasil perancangan perangkat lunak yang telah dirancang sebelumnya. Berdasarkan hal

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

6.2.2. Analisis Hasil Pengujian Validasi

Proses analisis terhadap hasil pengujian validasi dilakukan dengan melihat kecocokan antara hasil kinerja sistem dengan daftar kebutuhan. Berdasarkan hasil pengujian validasi dapat disimpulkan bahwa implementasi dan fungsionalitas perangkat lunak Informasi Gedung UBtelah memenuh kebutuhan yang telah dijabarkan pada tahap analisis kebutuhan.

6.2.3. Analisis Hasil Pengujian Performa

Dari hasil pengujian dapat ditarik kesimpulan bahwa nilai *transfer rate*dan jumlah *request per second* tidak linier terhadap banyaknya koneksi (*concurrency*) karena tidak memenuhi persamaan linier y=ax+b. Ketidaklinieran tersebut disebabkan oleh tidak adanya parameter waktu yang membatasi pada saat pengujian tiap n *concurrency* dan n *request*. Semakin tinggi *transfer rate*maka *request* yang dapat dilayani oleh *server* dalam satu detik juga akan semakin banyak. Nilai rata-rata dari data hasil pengujian performa dapat disimpulkan bahwa *webserver* pada jaringan*intranet* memiliki performa yang jauh lebih baik daripada performa *webserver* pada jaringan*internet*.

Pada pengujian performa terdapat parameter waktu yang dapat diuji pada performa aplikasi. Namun tidak dilakukan pada penelitian kali ini karena pada penelitian sebelumnya telah dilakukan dan didapatkan hasil bahwa proses pengambilan data tidak dipengaruhi oleh waktu tetapi lebih dipengaruhi olehkeadaansekitaryangada[AND-2012].

BAB VII
PENUTUP

7.1. Kesimpulan

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

1. Aplikasi Informasi Gedung UB pada *smartphone* Android dapat digunakan untuk mengenali koordinat gedung – gedung UB yang telah didaftarkan di *datasource* sebelumnya dan menampilkan informasi gedung dengan media *augmented reality* berbasis GPS sesuai dengan perancangan yang telah dibuat.
2. Koneksi data antara *web server* dan aplikasi pada *smartphone* android telah berhasil diimplementasikan dengan metode pengiriman data dari *web server* ke aplikasi *mobile* di android menggunakan metode JSON.
3. Hasil pengujian unit menunjukkan bahwa unit modul dari program sudah memenuhi kebutuhan fungsional yang telah dirancang pada tahap perancangan.
4. Berdasarkan hasil pengujian validasi dapat disimpulkan bahwa implementasi dan fungsionalitas perangkat lunak Informasi Gedung UBtelah memenuh kebutuhan yang telah dijabarkan pada tahap analisis kebutuhan.
5. Hasil pengujian nilai *throughput* data pada *server* yang dilakukan menggunakan *apache benchmark* dengan 2 jumlah *request* yang berbeda serta jumlah *concurrency* yang berbeda secara liniermenunjukkan bahwa nilai *transfer rate* dan jumlah *request per second* tidak linier terhadap banyaknya koneksi (*concurrency*). Ketidaklinieran tersebut disebabkan oleh tidak adanya parameter waktu yang membatasi pada saat pengujian tiap n *concurrency* dan n *request*.
6. Dari data *request per second* dan *transfer rate* yang didapat dari hasil pengujian dengan parameter *request* dan *concurrency* pada pengujian *webserver* pada jaringan *internet* dan *intranet* dapat disimpulkan bahwa

webserver pada *intranet* memiliki performa yang jauh lebih baik daripada performa *webserver* pada *internet.*

7.2. Saran

Saran yang dapat diberikan untuk pengembangan aplikasi Informasi Gedung UB pada *smartphone* android ini antara lain adalah :

1. Berdasarkan permasalahan yang terjadi dalam pengaksesan *datasource* dalam bentuk JSON, sebaiknya digunakan *web server*yang menyediakan hak akses sebagai *root.*
2. Untuk pengembangan lebih lanjut dari aplikasi ini, antarmuka untuk menampilkan informasi dibuat lebih *user friendly*.
3. Aplikasi ini dapat dikembangkan lebih lanjut dengan memberikan hak akses kepada *administrator* untuk menambahkan data gedung (*latitude*, *longitude*, nama gedung, *website*, detail informasi gedung) melalui aplikasi *client.*
4. Aplikasi ini dapat dikembangkan lebih lanjut dengan penambahan *data source* untuk informasi gedung di luar UB seperti kuliner malang, tempat rekreasi dan lain-lain, sehingga dapat digunakan masyarakat luas.

DAFTAR PUSTAKA

[AND-12] Ventausa, Andhika. 2012. *"Rancang Bangun Aplikasi Informasi Gedung Di Universitas Brawijaya Malang pada Posisi Pengguna dengan Sistem Operasi Android"*. Skripsi. Universitas Brawijaya. Malang.

[AZU-97] Azuma, Ronald T. (1997), *A Survey of Augmented Reality Presence: Teleoperators and Virtual Environment*, http://www.cs.unc.edu/~azuma/ ARpresence.pdf, Tanggal Akses 6 Januari 2012.

[DIG-09] Diggelen, Frank van. 2009. *A-GPS: Assisted GPS, GNSS, and SBAS*. Artech House.

[ELR-06] El-Rabbany, Ahmed. 2006.*Introduction to GPS: The Global PositioningSystem, second edition*. Norwood: Artech House.

[HER-11] Hermawan, Stephanus. 2011. *Mudah membuat aplikasi android*, Yogyakarta : Pemerbit ANDI.

[JSO-11] Json.org. 2011. *Pengenalan JSON*.<http://json.org> diakses tanggal : 25 Juli 2011.

[MIX-11] http://www.mixare.org. Tanggal Akses 10 November 2011.

[PHP-11] PHP team. 2011. *JSON Book Manual*. <http://php.net/manual> diakses tanggal : 27 Juli 2011.

[PRA-05] Prahasta, Eddy. 2005. Konsep – Konsep Dasar Sistem Informasi Geografis, Bandung : Penerbit Informatika.

[PRE-01] Pressman, Roger S. 2001. S*oftware Engineering : A Practitioner's Approach, Fifth Edition*. McGraw Hill.

[PRE-10] Pressman, Roger S. 2010. S*oftware Engineering : A Practitioner's Approach, seventh Edition*. McGraw Hill.

[SAF-11] Safaat H, Nazruddin. 2011. *Pemrograman Aplikasi Mobile Smartphone dan Tablet PC BerbasisAndroid*, Bandung :Penerbit INFORMATIKA.

[UBM-12] http://www.ub.ac.id/id/tentang_ub/profil.html. Tanggal Akses 01 Januari 2012.

[W3C-11] W3C. 2011. *Web Services Tutorial*. <http://www.w3schools.com/webservices/default.asp> diakses tanggal : 11 Juli 2011.

[WES-05] Wesley, Addison. 2005. *The Unified Modeling Language User Guide SECOND EDITION*.