RANCANG BANGUN *PUSH NOTIFICATION* BERBASIS MOODLE PADA PERANGKAT ANDROID

SKRIPSI

LABORATORIUM REKAYASA PERANGKAT LUNAK

Untuk memenuhi sebagian persyaratan mencapai gelar Sarjana Komputer

Disusun Oleh:

DWINTA AYU PRAMITA

NIM. 0910683036

KEMENTERIAN RISET TEKNOLOGI DAN PENDIDIKAN TINGGI

UNIVERSITAS BRAWIJAYA

PROGRAM TEKNOLOGI INFORMASI DAN ILMU KOMPUTER

PROGRAM STUDI INFORMATIKA/ILMU KOMPUTER

MALANG

2015

LEMBAR PERSETUJUAN

RANCANG BANGUN *PUSH NOTIFICATION* BERBASIS MOODLE PADA PERANGKAT ANDROID

SKRIPSI

LABORATORIUM REKAYASA PERANGKAT LUNAK

Untuk memenuhi sebagian persyaratan mencapai gelar Sarjana Komputer

Disusun Oleh:

DWINTA AYU PRAMITA

NIM. 0910683036

Telah diperiksa dan disetujui oleh:

Pembimbing I	Pembimbing II,
Ismiarta Aknuranda, S.T., M.Sc, Ph.D.	**Denny Sagita R., S.Kom., M.Kom.**
NIK. 740719 06 1 1 0079	**NIK. 851124 06 1 1 0250**

LEMBAR PENGESAHAN

RANCANG BANGUN *PUSH NOTIFICATION* BERBASIS MOODLE PADA PERANGKAT ANDROID

SKRIPSI

LABORATORIUM REKAYASA PERANGKAT LUNAK

Untuk memenuhi sebagian persyaratan mencapai gelar Sarjana Komputer

Disusun Oleh:
DWINTA AYU PRAMITA
NIM. 0910683036

Setelah dipertahankan di depan Majelis Penguji
pada tanggal 8 Juli 2015
dan dinyatakan memenuhi syarat untuk memperoleh
gelar Sarjana dalam bidang Ilmu Komputer

Penguji I	**Penguji II,**
Agi Putra Kharisma, S.T., M.T.	**Aswin Suharsono, S.T., M.T.**
NIK. 201304 860430 1 001	**NIK. 840919 06 1 1 0251**

Penguji III,

Eriq Muhammad Adams J., S.T., M.Kom.
NIK. 19850410 201212 1 001

Mengetahui,
Ketua Program Studi Informatika / Ilmu Komputer

Drs. Marji, M.T.
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, 8 Juli 2015

Mahasiswa,

Dwinta Ayu Pramita

NIM.0910683036

KATA PENGANTAR

Dengan nama Allah SWT Yang Maha Pengasih dan Penyayang. Segala puji bagi Allah SWT karena atas rahmat dan hidayahNya-lah penulis dapat menyelesaikan proposal skripsi ini dengan baik meskipun tidak tepat dengan waktu yang dijadwalkan. Penyusunan skripsi ini adalah dengan maksud untuk memenuhi salah satu persyaratan dalam mengikuti ujian akhir Sarjana Program Studi Informatika/ Ilmu Komputer pada Program Teknologi Informasi dan Ilmu Komputer Universitas Brawijaya, dengan judul **"Rancang Bangun *Push Notification* Berbasis MOODLE pada Perangkat Android"**

Banyak hambatan yang telah dihadapi penulis dalam penyelesaian skripsi ini, namun berkat bantuan dan dukungan dari berbagai pihak hambatan-hambatan tersebut dapat terlewati. 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 kepada :

1. Ismiarta Aknuranda, ST., M.Sc, Ph.D., selaku dosen pembimbing I yang telah memberikan ilmu dan saran untuk proposal skripsi ini.
2. Denny Sagita R., S.Kom., M.Kom., selaku dosen pembimbing II yang telah memberikan ilmu dan saran untuk proposal skripsi ini.
3. Arief Andy Soebroto, ST., M.Kom., selaku dosen pembimbing akademik yang telah memberikan pengarahan selama penulis menempuh pendidikan di Program Teknologi Informasi dan Ilmu Komputer Universitas Brawijaya.
4. Drs. Marji, MT., selaku Ketua Program Studi Informatika/Ilmu Komputer Universitas Brawijaya.
5. Ir. Sutrisno, MT. selaku Ketua Program Teknologi Informasi dan Ilmu Komputer (PTIIK)
6. Segenap bapak dan ibu dosen yang telah mendidik dan mengajarkan

ilmunya kepada penulis selama menempuh pendidikan di Program Teknologi Informasi dan Ilmu Komputer Universitas Brawijaya.

7. Orang tua saya, Agus Supradana dan Sri Wahyuningsih serta saudara kandung saya Arief Budi Prastetyo dan Ramadhan Bakti Pradama yang telah memberikan dukungan moral dan material.
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. Seluruh mahasiswa Program Teknologi Informasi dan Ilmu Komputer, khususnya teman – teman yang telah membantu terealisasinya skripsi ini.
10. Teman - teman Bontang dan teman – teman kost, terimakasih atas segala dukungan dan semangatnya sampai terselesaikannya skripsi ini.
11. Semua pihak yang telah membantu terselesaikannya penyusunan skripsi ini yang tidak dapat disebutkan satu per satu.

Hanya doa dan rasa terima kasih yang dapat penulis berikan semoga Tuhan membalas seluruh kebaikan dan berkat yang melimpah. Penulis menyadari bahwa skripsi ini jauh dari sempurna dan banyak kekurangan. Untuk itu, saran dan kritik yang membangun sangat penulis harapkan. Akhir kata, penulis berharap semoga skripsi ini dapat memberikan sesuatu yang bermanfaat bagi semua pihak yang membacanya

Malang, 8 Juli 2015

Penulis

ABSTRAK

Dwinta Ayu Pramita. 2015. : Rancang Bangun *Push Notification* Berbasis Moodle pada Perangkat Android. Skripsi Program Studi Informatika/Ilmu Komputer, Program Teknologi Informasi dan Ilmu Komputer, Universitas Brawijaya. Dosen Pembimbing : Ismiarta Aknuranda, S.T., M.Sc., Ph.D. dan Denny Sagita R., S.Kom., M.Kom.

Kemudahan akses media pembelajaran secara online mendorong pembelajar memiliki tuntutan untuk selalu memantau secara periodik perubahan yang terjadi di e-learning. Dalam rangka meningkatkan efektifitas dan fleksibilitas e-learning diperlukan suatu sistem *push notification* yang dapat memantau atau memonitoring perubahan yang terjadi pada e-learning dan diterima pembelajar secara *real time* melalui piranti *mobile*. Bantuan teknologi *mobile learning* memudahkan pengaksesan e-learning yang dapat dilakukan dimanapun dan kapanpun sehingga tidak membatasi mobilitas pembelajar. LMS adalah aplikasi e-learning berbasis *web* yang berfungsi mengatur tata laksana penyelenggaraan pembelajaran di dalam e-learning. Salah satu LMS yang bersifat *open source* dan banyak digunakan adalah Moodle. Aplikasi ini dikembangkan menggunakan metode *reuse oriented*, metode ini mencangkup kegiatan spesifikasi kebutuhan, analisis komponen, modifikasi kebutuhan, desain sistem dengan *reuse*, pengembangan dan integrasi serta pengujian. Sistem ini mampu memberikan notifikasi berupa pembaharuan materi, quiz, tugas, dan seluruh layanan yang telah disediakan oleh Moodle. Aplikasi *push notification* berjalan pada sistem operasi Android yang telah diuji cobakan kepada 30 pembelajar. Seluruh notifikasi sampai kepada pembelajar dan menampilkan isi berupa mata kuliah, tipe yang berubah, dan batas waktu sesuai dengan perubahan yang terjadi pada e-learning Moodle.

Kata Kunci : *push notification*, LMS, Moodle, *mobile learning*

ABSTRACT

Dwinta Ayu Pramita. 2015. : Design Push Notification Moodle Based on Android Devices. Department of Computer Science and Information Technology, University of Brawijaya. Advisors : Ismiarta Aknuranda, S.T., M.Sc., Ph.D. and Denny Sagita R., S.Kom., M.Kom.

Ease access of media online learning encourages learners to constantly monitor the changes that occur periodically in e-learning. In order to improve the effectiveness and flexibility of e-learning requires a push notification system that can monitor changes in e-learning and learners can received it in real time via mobile devices. Mobile learning technology facilitate learners to access e-learning that can be open anywhere and anytime so it does not restrict the mobility of learners. LMS is an application of e-learning web-based that can controls the operation of learning in e-learning web-based. One of LMS which is open source and widely used is Moodle. This application was developed using oriented reuse method, this method covers the activities of specification requirements, component analysis, modification requirements, system design with reuse, development and integration and testing. This system is capable of providing notification in the form of renewal materials, quizzes, assignments, and all services that have been provided by Moodle. Push notification application runs on the Android operating system that has been tested to 30 learners. Learners will get the notification and display the contents that contains subject, the type has changed, and the deadline in accordance with changes in the Moodle e-learning.

Keywords : push notification, LMS, Moodle, mobile learning

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

BAB I
PENDAHULUAN

1.1 Latar Belakang

Penggunaan teknologi informasi dan komunikasi dalam dunia pendidikan terus berkembang dengan berbagai strategi dan pola. Salah satu hasil dari perkembangan dunia pendidikan ialah sistem e-learning. Sistem tersebut merupakan suatu bentuk pembelajaran yang memanfaatkan perangkat elektronik, dan media digital.

LMS (*Learning Management Sistem*) adalah aplikasi e-learning (*Electronic Learning*) berbasis *web* yang biasa digunakan oleh lembaga pendidikan dan perusahaan yang dapat berfungsi untuk mengatur tata laksana penyelenggaraan pembelajaran di dalam e-learning. Salah satu LMS yang bersifat *open source* dan banyak digunakan adalah Moodle (*Modular Object-Oriented Dynamic Learning Environment*) [KAY-10:13].

Moodle memberikan banyak kemudahan dalam pengelolaan pembelajaran. Moodle mengatasi batasan ruang dan waktu bagi pembelajar dan instruktur. Di sisi lain, penggunaan Moodle secara intensif mengakibatkan pembelajar memiliki tuntutan untuk selalu memantau secara periodik perubahan yang terjadi di Moodle. Selain itu pembelajar seringkali keberatan untuk membuka Moodle menggunakan perangkat selularnya karena membutuhkan *bandwidth* besar untuk mengaksesnya. Sebagian besar pembelajar membuka Moodle hanya ketika memegang komputer *desktop*. Hal tersebut sulit untuk dilakukan dimanapun dan kapanpun serta menjadi pembatas mobilitas seorang pembelajar.

Efektifitas dan fleksibilitas Moodle dapat ditingkatkan dengan bantuan teknologi *mobile learning* (m-learning). M-learning merupakan penggabungan teknologi *mobile communication* dengan e-learning. Salah satu kegunaannya adalah sebagai wadah diskusi dan penyebaran materi pembelajaran dengan menggunakan teknologi berbasis internet pada piranti *mobile* [MDM-06].

Peneliti bermaksud memanfaatkan keberadaan Moodle dan dukungan kemampuan teknologi seluler yang telah ada untuk mengembangkan layanan *push notification* pada piranti *mobile*. Aplikasi tersebut nantinya akan menjadi aplikasi m-learning yang dapat mengingatkan seorang pembelajar mengenai perubahan bahan ajar dan berita-berita yang terkait dengan mata kuliah yang ditempuh.

Aplikasi *push notification* merupakan suatu alternatif pembantu untuk menghubungkan Moodle dengan *mobile device*. *Push notification* merupakan sebuah teknologi penyampaian informasi dari aplikasi perangkat lunak untuk perangkat komputasi tanpa permintaan khusus dari *client* [SEA-14].

Dalam usulan peneliti, aplikasi *push notification* diterapkan pada *smartphone* dengan sistem operasi Android karena sistem operasi ini paling banyak digunakan untuk perangkat bergerak di dunia, yang mencapai 44.95% dari seluruh sistem operasi yang ada [STA-13].

Berdasarkan paparan informasi tersebut, penulis membangun suatu aplikasi yang memiliki kemampuan integrasi *mobile* OS Android dengan Moodle berdasarkan model pengembangan berorientasi pada penggunaan ulang (*reuse oriented*). Penulis mengambil judul "**Rancang Bangun Push Notification Berbasis Moodle Pada Perangkat Android"** dengan harapan aplikasi ini dapat membantu memudahkan pembelajar dalam kegiatan pemantauan berita terkini mengenai perubahan yang terjadi pada Moodle.

1.2 Rumusan Masalah

1. Bagaimana melakukan analisis kebutuhan pembuatan aplikasi *Push Notification* berbasis Moodle di dalam sistem operasi Android ?
2. Bagaimana memodelkan aplikasi *Push Notification* ke dalam model desain sebuah perangkat lunak berorientasi objek yang memanfaatkan pola-pola perancangan (*Design Patterns*) ?
3. Bagaimana melakukan pengujian pada aplikasi *Push Notification* di dalam sistem operasi Android sebagai aplikasi pendukung pembelajaran?

1.3 Tujuan

Mengembangkan sesuai dengan latar belakang dan rumusan masalah, tujuan dari penyusunan skripsi ini adalah merancang, mengimplementasikan, dan menguji aplikasi *Push Notification* berbasis Moodle pada perangkat Android sehingga dapat membantu memudahan mahasiswa dalam pemantauan kegiatan perkuliahan.

1.4 Batasan Masalah

Dari rumusan masalah yang telah disampaikan maka penelitian ini difokuskan dengan batasan sebagai berikut:

1. Aplikasi yang dibangun merupakan aplikasi *client* berbasis *mobile* Android.
2. Aplikasi *push notification* akan sampai ketika pembelajar selalu dalam keadaan *login* dan terhubung dengan jaringan internet.
3. Aplikasi *push notification* hanya dapat melihat mata kuliah yang mengalami perubahan dan jenis perubahan berupa penambahan materi, chat, database, forum, quiz, wiki, survei, url, serta aktivitas lain yang terdapat pada Moodle 2.6. Aplikasi tidak dapat melakukan pengunduhan materi atapun kegiatan lainnya.
4. Pengujian yang dilakukan ditekankan kepada fungsionalitas dari *Push Notification* di dalam sistem operasi android dan performa dari aplikasi *push notification*.

1.5 Manfaat

Penulisan skripsi ini diharapkan mempunyai manfaat yang baik dan berguna bagi pembaca dan penulis. Adapun manfaat yang diharapkan adalah sebagai berikut:

- Bagi Penulis
  1. Mendapatkan pengetahuan dan pemahaman mengenai Moodle sehingga dapat mengimplementasikannya sesuai dengan kebutuhan.

2. Mendapatkan pengetahuan dan wawasan terkait analisa dan perancangan aplikasi *Push Notification* android berbasis Moodle.

- Bagi pembaca/pengguna

1. Mendapatkan wawasan mengenai proses implementasi berbasis Moodle pada perangkat bergerak sistem operasi Android.
2. Sebagai sarana mempermudah perkuliahan mahasiswa untuk mengetahui berita terkini dalam e-learning dengan muculnya *push notification* pada perangkat bergerak sistem operasi Android.

1.6 Sistematika Penulisan

Sistematika penulisan dalam skripsi ini sebagai berikut:

BAB I Pendahuluan

Pada bab ini akan dijelaskan mengenai latar belakang pembuatan skripsi, rumusan masalah, batasan masalah, tujuan, manfaat, dan sistematika penulisan.

BAB II Landasan Kepustakaan

Landasan kepustakaan terbagi menjadi dua yaitu kajian pustaka yang munguraikan mengenai penelitian sebelumnya dan dasar teori yang mendasari pembuatan aplikasi *push notification* pada sistem operasi Android.

BAB III Metodologi Penelitian

Dalam bab metode penelitian terdapat metode dan langkah kerja dalam merancancang aplikasi. Langkah-langkah yang dijelaskan dalam bab ini meliputi studi literatur, perancangan, implementasi, pengujian, dan analisis serta pengambilan kesimpulan dan saran.

BAB IV Analisis Persyaratan dan Perancangan

Bab ini membahas analisis persyaratan dan perancangan aplikasi *push notification* pada sistem operasi Android.

BAB V Implementasi

Bab Implementasi membahas implementasi dari aplikasi *push notification* pada sistem operasi Android sesuai dengan perancangan sistem yang telah dibuat.

BAB VI Pengujian dan Analisis

Bab pengujian dan analisis ini memuat hasil pengujian dan analisis terhadap perangkat lunak yang telah direalisasikan.

BAB VII Penutup

Pada bab ini terdapat kesimpulan yang diambil berdasarkan analisis hal-hal penting dari pembuatan dan pengujian perangkat lunak yang dikembangkan dalam skripsi ini serta saran – saran untuk pengembangan dari aplikasi lebih lanjut.

BAB II
KAJIAN PUSTAKA DAN DASAR TEORI

Pada Bab ini akan berisi kajian pustaka dan dasar teori yang diperlukan untuk penelitian. Kajian pustaka adalah membahas penelitian yang telah ada dan yang diusulkan dan dijelaskan mengenai konsep teori yang dijadikan landasan dalam pengembangan aplikasi ini . Teori yang dibahas adalah Rekayasa Perangkat Lunak, *Software Process Model*, *Unified Modelling Language*, Pengujian Perangkat Lunak, Pola-pola perancangan, Sistem Operasi Android, e-larning, m-learning, *Google Cloud Messanger* (GCM), *Modular Object Oriented Dynamic Learning Enviroment* (Moodle), Teknologi *Push*.

2.1 Kajian Pustaka

Kajian pustaka pada penelitian ini adalah membahas penelitian sebelumnya yang berjudul "*Moodlbile: Extending Moodle To The Mobile On/Offline Scenario*". Penelitian tersebut membahas mengenai pembelajaran berbasis *web* yang terintegrasi dengan *mobile learning*. Parameter keberhasilan dari penelitian yang dilakukan adalah dapat memenuhi kebutuhan pembelajaran Moodle seperti: forum, kalender, wiki, glossary dan internal mail yang dapat bekerja secara *on/offline* dan menggunakan teknologi *web service* [FOR-08].

Berbeda dengan yang Moodlbile, pada penelitian ini aplikasi berjalan harus dengan menggunakan koneksi internet. Aplikasi yang akan dibuat bukan menampilkan data yang telah disimpan saat *online*, tetapi menampilkan data berita terbaru mengenai merubahan yang terjadi di e-learning moodle. Aplikasi yang akan dibangun adalah sebuah aplikasi *push notification* yang berjalan *real time* pada perangkat bergerak di dalam sistem operasi Android untuk memudahkan pemantauan perubahan yang terjadi pada e-learning Moodle. Aplikasi *push notification* akan menampilkan berita seluruh aktivitas yang diinputkan oleh pengajar. Pembuatan aplikasi pada penelitian ini menggunakan teknologi GCM sebagai pertukaran data dari *server* ke Android *application*.

2.2 Rekayasa Perangkat Lunak

Rekayasa perangkat lunak adalah disiplin ilmu yang membahas semua aspek produksi perangkat lunak, mulai dari tahap awal spesifikasi sistem sampai pemeliharaan sistem setelah digunakan [SOM-11].

Rekayasa perangkat lunak lebih fokus pada bagaimana membuat perangkat lunak yang memenuhi kriteria berikut [SOM-11] :

- Dapat terus dipelihara setelah perangkat lunak selesai dibuat seiring berkembangnya teknologi dan lingkungan (*maintainability*)
- Dapat diandalkan dengan proses bisnis yang dijalankan dan perubahan yang terjadi (*dependability and roboust*)
- Efisien dari segi sumber daya dan penggunaan
- Kemampuan untuk dipakai sesuai dengan kebutuhan (*usability*)

Dari kriteria di atas maka perangkat lunak yang baik adalah perangkat lunak yang dapat memenuhi kebutuhan pelanggan (*customer*) atau pemakai perangkat lunak (*user*) atau berorientasi pada pelanggan atau pemakai perangkat lunak, bukan berorientasi pada pembuat atau pengembang perangkat lunak.

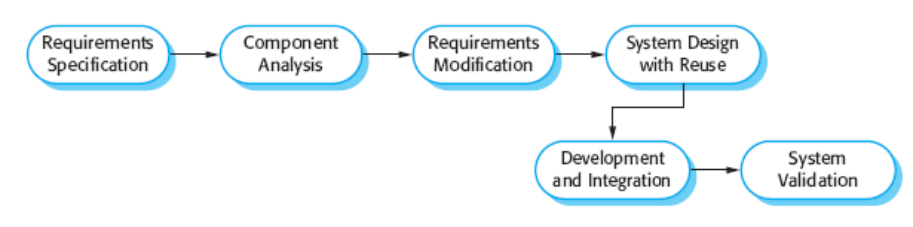
Secara umum, perekayasa perangkat lunak memakai pendekatan yang sistematis dan terorganisir terhadap pekerjaan mereka karena cara ini seringkali paling efektif untuk menghasilkan perangkat lunak berkualitas tinggi. Namun, rekayasa ini sebenarnya mencakup masalah pemilihan metode yang paling sesuai untuk suatu keadaan, pendekatan yang lebih kreatif, dan pengembangan yang mungkin efektif pada beberapa keadaan [SOM-11:7-10].

2.3 Software Process Model

Model proses perangkat lunak adalah representasi yang sederhana dari proses perangkat lunak. Setiap model proses merupakan proses dari perspektif tertentu, dan dengan demikian hanya menyediakan informasi parsial tentang proses tersebut. Sebuah model proses untuk rekayasa perangkat lunak dipilih berdasarkan sifat proyek dan aplikasi yang akan dibuat.

Reuse-oriented software engineering adalah penggunaan kembali perangkat lunak yang telah ada. Hal ini sering terjadi ketika orang-orang yang bekerja pada proyek mengetahui desain atau kode yang mirip dengan apa yang dibutuhkan. Mereka mencari dan memodifikasi sesuai kebutuhan lalu menggabungkan ke dalam sistem. Pendekatan *reuse-oriented software engineering* mengandalakan komponen perangkat lunak yang dapat digunakan kembali dan mengintegrasikan kerangka untuk komposisi komponen sistem daripada mengembangkan sistem dari awal [SOM-11:29-35].

Tahapan utama dari *reuse-oriented software engineering* secara langsung mencerminkan dasar pembangunan kegiatan. Model proses secara umum dari pengembangan berbasis *reuse* dapat dilihat pada gambar 2.1.



Gambar 2.1 Reuse Oriented Model

Sumber : [SOM-11:35]

Walaupun inisialisasi spesifikasi kebutuhuan dan validasi pada proses ini sama dengan proses pengembangan perangkat lunak yang lain, namun terdapat perbedaan pada pertengahan proses, yaitu dengan adanya proses *reuse-oriented*. Menurut *Sommerville* [SOM-11] proses ini antara lain:

1. Analisis komponen

   Dari spesifikasi kebutuhan yang telah ditentukan, dicari komponen yang dapat mengimplementasi spesifikasi tersebut. Biasanya, tidak ada komponen yang benar-benar mampu memenuhi spesifikasi dan mungkin hanya beberapa komponen yang memenuhi persyaratan fungsional dari spesifikasi kebutuhan.

2. Modifikasi kebutuhan

   Selama tahap ini kebutuhan dianalisis menggunakan informasi tentang komponen yang telah ditentukan. Kemudian kebutuhan dimodifikasi untuk mencerminkan komponen yang tersedia. Jika dalam satu kondisi tidak memungkinkan untuk memodifikasi kebutuhan, kegiatan analisis komponen dapat dilakukan kembali untuk mencari solusi alternatif.

3. Desain sistem dengan *reuse*

   Selama fase ini dilakukan desain *framework* pada sistem atau *framework* yang tersedia akan digunakan kembali. Para desainer memperhitungkan komponen yang digunakan kembali dan mengatur *framework* untuk mengembangkannya.

4. Pengembangan dan integrasi perangkat lunak

   Perangkat lunak yang tidak dapat diperoleh secara eksternal maka akan dikembangkan dan komponen diintegrasikan untuk menciptakan sistem baru.

Sommerville [SOM-11] juga menyebutkan tiga tipe komponen perangkat lunak yang dapat digunakan dalam proses *reuse-oriented*, yaitu :

1. *Web service* yang dikembangkan berdasarkan *service standard* dan tersedia untuk digunakan secara *remote.*
2. Koleksi *object* yang dikembangkan sebagai *package* untuk diintegrasikan dengan sebuah *framework* komponen seperti .NET dan J2EE.
3. Sistem perangkat lunak yang berdiri sendri yang dapat dikonfigurasi untuk digunakan dalam lingkungan tertentu.

Rekayasa perangkat lunak *reuse-oriented* memiliki keuntungan jelas dalam mengurangi jumlah perangkat lunak yang dikembangkan dan mengurangi biaya dan resiko.

Cakupan *reuse-oriented* yang digunakan pada aplikasi nantinya adalah *program library* dan *design pattern*. *Program library* digunakan untuk proses penampilan *notification*, sedangkan *design patern* digunakan untuk merancang arsitektur aplikasi agar dapat digunakan kembali pada tahap pengembangan. *Program libray* yang akan digunakan pada aplikasi adalah *Google Cloud Messaging* dan *design pattern* yang digunakan adalah *factory method Pattern* dan *singleton pattern.*

2.4 Pola – pola Perancangan

Seorang pemrogram atau pengembang perangkat lunak bertugas menuntaskan suatu permasalahan dalam membuat sebuah perangkat lunak dengan melakukan perubahan dari model, abstraksi desain ke dalam baris kode bahasa pemrograman. Tidak jarang masalah yang dihadapi seorang *programmer* bersifat rekursif atau dapat dikatakan berulang, karena dari sudut pandang pengalaman *programmer* tersebut mungkin dia sudah membuat solusi atas masalah pemrograman tersebut dimasa lalu tapi entah kenapa ketika dihadapkan pada masalah yang jika ditarik prinsip utamanya adalah sama, *programmer* seolah kembali berupaya menemukan solusi pemrograman itu.

Permasalahan tersebut adalah hal yang biasa ditemui dalam dunia pengembangan perangkat lunak dan dapat dijawab dengan menggunakan *Design Patterns* atau pola-pola perancangan. *Design Patterns* merupakan sekumpulan solusi inti yang terbukti mampu diterapkan pada beberapa permasalahan pemrograman berorientasi objek yang terulang dalam situasi dan kondisi tertentu. *Design Patterns* mulai populer didalam pengembangan perangkat lunak ketika Erich Gamma beserta rekan yang kemudian disebut sebagai *Gang Of Four* (GoF) membuat buku *Design Patterns*: *Elements of Reusable object-oriented Software* [GAM-01] yang didalamnya terdapat 23 jenis patterns standar walaupun hanya sebagian saja yang lebih banyak ditemui (diimplementasikan) pada dunia pengembangan perangkat lunak [HOL-05].

23 Pattern tersebut dikategorikan menjadi tiga jenis yang berbeda berdasarkan penggunaanya :

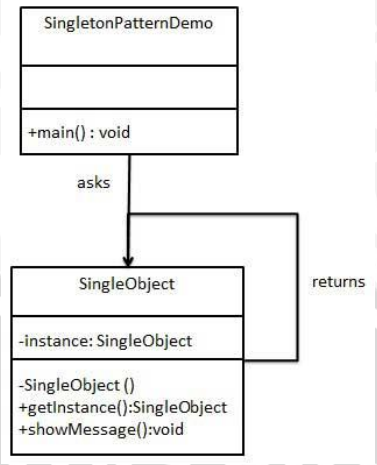
a. *Creational Patterns*: digunakan untuk membuat konstruksi sebuah objek sehingga proses pembuatan objek (instansiasi) terpisah dari sistem yang hendak mengimplementasikanya.
b. *Structural Patterns*: digunakan untuk membuat struktur objek yang besar yang terdiri dari banyak objek-objek lain yang terpisah.
c. *Behavioural Patterns*: digunakan mengatur algoritma, relasi dan tanggung jawab antar objek dalam sebuah sistem.

Pada proses skripsi ini, pola-pola perancangan yang digunakan dalam perancangan aplikasi adalah *singleton pattern* dan *factory method pattern.*

2.4.1 Singletone Pattern

Singleton pattern adalah salah satu *desain pattern* sederhana pada pemrograman java. Jenis *desain pattern* berada di bawah *creational pattern* yang memberikan salah satu cara terbaik untuk membuat objek.

Ada class yang hanya perlu diinstansiasi satu kali saja. Alasan : Dalam beberapa kasus hanya dibutuhkan satu object saja. Memastikan suatu *class* hanya dapat "menghasilkan" satu object saja. Konsepnya mirip global variable dengan perbedaan, pada *singleton,* object hanya dibuat saat dibutuhkan, sedangkan global variable dibuat saat awal program dijalankan.



Gambar 2.2 Struktur Singletone Pattern

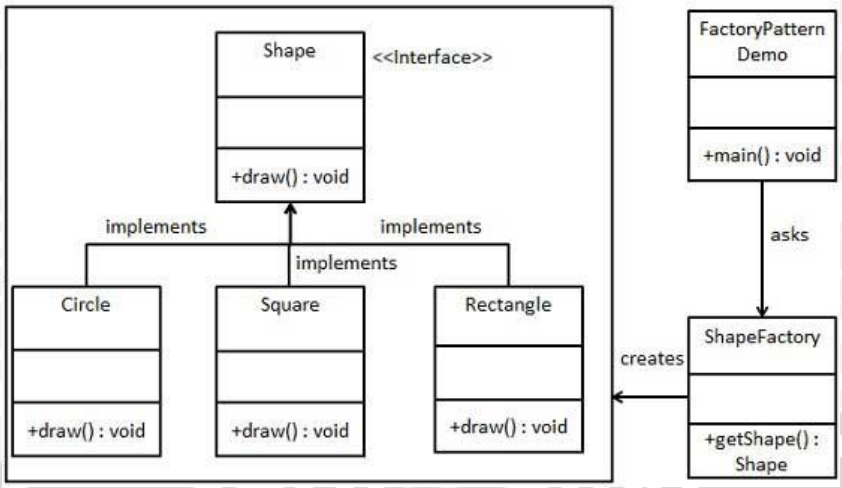
Sumber : [TUT-14]

2.4.2 Factory Pattern

Factory Pattern merupakan *pattern* yang *creational* yang biasa digunakan untuk memisahkan (*decouple*) proses pembuatan atau instasiasi sebuah objek (produk) dari objek lain (*client*) yang menggunakannya.

Tujuan dari adanya factory pattern supaya apabila terjadi perubahan pada product *class* tidak menyebabkan perubahan kode pada *client*. Akibat dari perubahan dapat diminimalisir, serta factory dapat digunakan oleh banyak *class*.

Factory method adalalah method yang memiliki fungsi untuk melakukan konstruksi *class* menjadi object dan mengembalikan referensi object tersebut. *Factory* adalah objek yang berfungsi membuat objek lain (*produk*). *Class* ini menyembunyikan proses pembuatan produk dari *klien* sehingga *klien* tidak perlu tahu proses pembuatannya, bahkan klien juga tidak perlu tahu nama *class* dari produk yang dia minta.



Gambar 2.3 Struktur Factory Pattern

Sumber : [TUT-14]

2.5 Unified Modelling Language (UML)

UML adalah sebuah bahasa yang telah menjadi standar dalam industri untuk visualisasi, merancang dan mendokumentasikan sistem piranti lunak. UML menawarkan sebuah standar untuk merancang model sebuah sistem. UML mendefinisikan notasi dan *syntax*/semantik. Notasi UML merupakan sekumpulan bentuk khusus untuk menggambarkan berbagai diagram piranti lunak. Setiap bentuk memiliki makna tertentu, dan UML *syntax* 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*) [DHA-03].

Adapun sejarah UML dimulai dari banyaknya metodologi pemodelan berorientasi objek yang telah bermunculan di dunia di era tahun 1990. Diantaranya metodologi booch, metodologi coad, metodologi OOSE, metodologi OMT, metodologi shlaer-mellor, metodologi wirfs-brock, dan sebagainya. Dikarenakan metodologi yang ada membawa notasi sendiri-sendiri yang mengakibatkan timbul masalah baru apabila ada sebuah kerjasama dengan *group*/perusahaan lain yang menggunakan metodologi yang berlainan sehingga pada tahun 1996 UML dijadikan standar bahasa pemodelan untuk aplikasi berorientasi objek [DHA-03].

Pada UML versi 2.3 terdiri dari 13 macam diagram yang dikelompokkan dalam 3 kategori [DHA-03]. Berikut ini penjelasan singkat dari pembagian kategori tersebut.

- *Structure diagrams* yaitu kumpulan diagram yang digunakan untuk menggambarkan suatu struktur statis dari sistem yang dimodelkan.
- *Behavior diagrams* yaitu kumpulan diagram yang digunakan untuk menggambarkan kelakuan sistem atau rangkaian perubahan yang terjadi pada sebuah sistem.
- *Interaction diagrams* yaitu kumpulan diagram yang digunakan untuk menggambarkan interaksi sistem dengan sistem lain maupun interaksi antar subsistem pada suatu sistem.

2.5.1 Use Case Diagram

Diagram *use case* merupakan pemodelan untuk kelakuan (*behavior*) sistem informasi yang akan dibuat. *Use case* mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem informasi yang akan dibuat. Diagram *use case* digunakan untuk mengetahui fungsi apa saja yang ada di dalam sebuah sistem informasi dan siapa saja yang berhak menggunakan fungsi-fungsi itu.

Syarat penamaan pada diagram *use case* adalah nama didefinisikan sesimpel mungkin dan dapat dipahami. Ada dua hal utama pada *use case* yaitu pendefinisian apa yang disebut aktor dan *use case*.

1. Aktor merupakan orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat di luar sistem informasi yang akan dibuat itu sendiri. Simbol aktor adalah gambar orang, namun aktor belum tentu merupakan orang.
2. *Use case* merupakan fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antarunit atau aktor.

2.5.2 Class Diagram

Class diagram adalah suatu diagram yang memperlihatkan atau menampilkan struktur dari sebuah sistem,sistem tersebut akan menampilkan system kelas,atribut dan hubungan antara kelas ketika suatu sistem telah selesai membuat diagram. Objek diagram adalah suatu diagram yang berfungsi untuk mengatur atribut, objek dan hubungan antara *class*. Objek diagram juga dapat menampilkan struktur model sistem dalam waktu tertentu. *Class* dapat merupakan implementasi dari sebuah *interface,* yaitu *class* abstrak yang hanya memiliki metoda. *Interface* tidak dapat langsung diinstansiasikan, tetapi harus diimplementasikan dahulu menjadi sebuah *class.* Dengan demikian *interface* mendukung resolusi metoda pada saat *run-time* [DHA-03].

Class memiliki tiga area pokok :

1. Nama (dan *stereotype*)
2. Atribut merupakan variabel-variabel yang dimiliki oleh suatu kelas
3. Metoda atau operasi adalah fungsi-fungsi yang dimiliki oleh suatu kelas

Atribut dan metoda dapat memiliki salah satu sifat berikut :

1. *Private,* tidak dapat dipanggil dari luar *class* yang bersangkutan
2. *Protected,* hanya dapat dipanggil oleh *class* yang bersangkutan dan anak - anak yang mewarisinya
3. *Public,* dapat dipanggil oleh siapa saja

Hubungan antara *class* dibagi menjadi empat bagian antara lain.

1. Asosiasi, yaitu hubungan statis antar *class*. Umumnya menggambarkan *class* yang memiliki atribut berupa *class* lain, atau

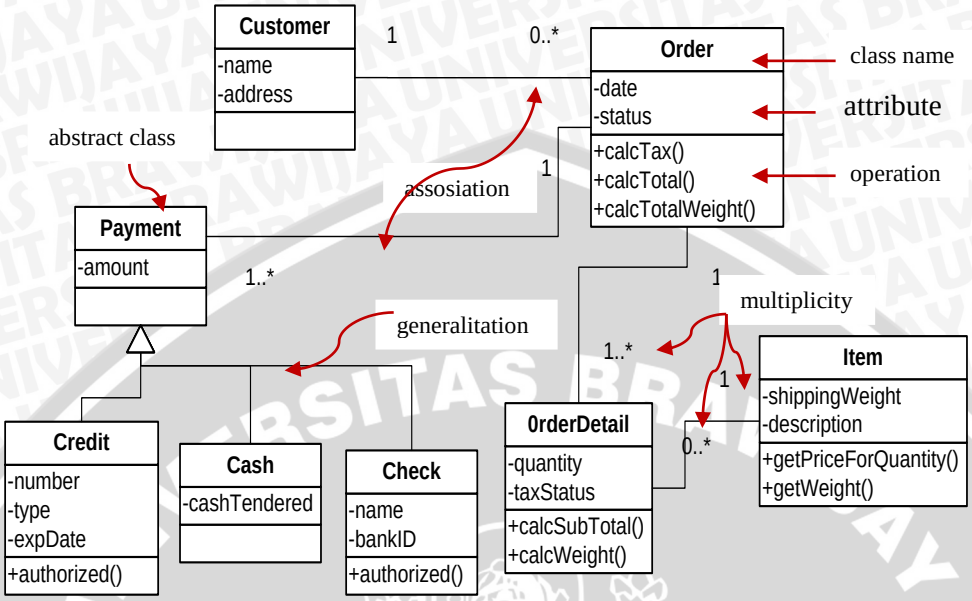
class yang harus mengetahui eksistensi *class* lain. Panah *navigability* menunjukkan arah *query* antar *class*.

2. Agregasi, yaitu hubungan yang menyatakan bagian ("terdiri atas").
3. Pewarisan, yaitu hubungan hirarkis antar *class*. *Class* dapat diturunkan dari *class* lain, mewarisi semua atribut dan metoda *class* asalnya, menambahkan fungsionalitas baru.
4. Hubungan dinamis, yaitu rangkaian pesan (*message*) yang dikirim dari satu *class* kepada *class* lain. Hubungan dinamis dapat digambarkan dengan menggunakan *sequence diagram.*

Kelas-kelas yang ada pada struktur sistem harus dapat melakukan fungsi-fungsi sesuai dengan kebutuhan sistem. Susunan struktur kelas yang baik pada diagram kelas sebaiknya memiliki jenis-jenis kelas berikut:

- Kelas main
  Kelas yang memiliki fungsi awal dieksekusi ketika sistem dijalankan
- Kelas yang menangani tampilan sistem
  Kelas yang mendefinisikan dan mengatur tampilan ke pemakai
- Kelas yang diambil dari pendefisian use case
  Kelas yang menangani fungsi-fungsi yang harus ada diambil dari pendefisian use case
- Kelas yang diambil dari pendefisian data
  Kelas yang digunakan untuk memegang atau membungkus data menjadi sebuah kesatuan yang diambil maupun akan disimpan ke basis data.

Jenis-jenis kelas di atas juga dapat digabungkan satu sama lain sesuai dengan pertimbangan yang dianggap baik asalkan fungsi-fungsi yang sebaiknya ada pada struktur kelas tetap ada. Gambar 2.4 merupakan sebuah *class* diagram yang memiliki beberapa hubungan antar *class*



Gambar 2.4 Contoh Class Diagram

Sumber : [DHA-03]

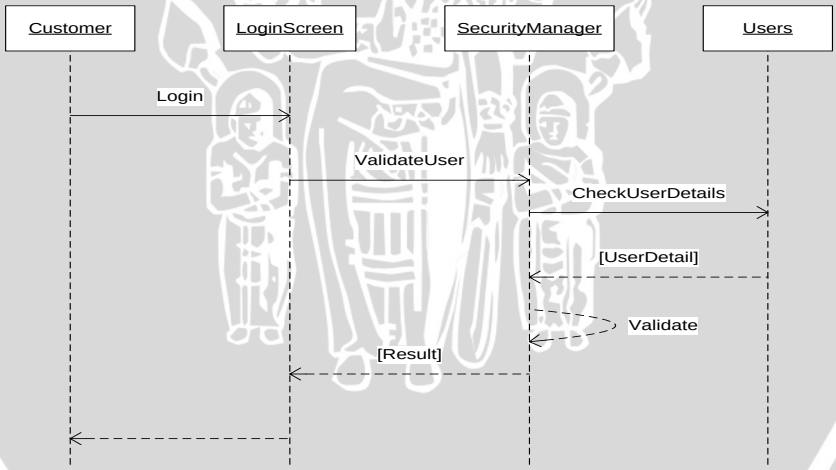
2.5.3 Sequence Diagram

Sequence diagram adalah suatu diagram yang menggambarkan interaksi antar obyek dan mengindikasikan komunikasi diantara obyek-obyek tersebut. Diagram ini juga menunjukkan serangkaian pesan yang dipertukarkan oleh obyek-obyek yang melakukan suatu tugas atau aksi tertentu. *Sequence diagram* sering diibaratkan dengan penggambaran kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup objek dan message yang dikirmkan dan diterima antarobjek

Obyek-obyek tersebut kemudian diurutkan dari kiri ke kanan, aktor yang menginisiasi interaksi biasanya diletak di paling kiri dari diagram. Pada diagram ini, dimensi vertikal merepresentasikan waktu [DHA-03].

Bagian paling atas dari diagram menjadi titik awal dan waktu berjalan ke bawah sampai dengan bagian dasar dari diagram. Garis vertical pada *sequence* diagram disebut *lifeline*, dilekatkan pada setiap obyek atau aktor. Kemudian, *lifeline* tersebut digambarkan menjadi kotak ketika obyek melakukan suatu operasi, kotak tersebut disebut *activation box*. Obyek dikatakan mempunyai *live activation* pada saat tersebut. Pesan yang dipertukarkan antar obyek digambarkan sebagai sebuah anak panah antara *activation box* pengirim dan penerima. Kemudian diatasnya diberikan label pesan [DHA-03].

Banyaknya *sequence diagram* yang harus digambar adalah sebanyak pendefisian use case yang memiliki proses sendiri atau yang penting semua use case yang telah didefinisikan interaksi jalannya pesan sudah dicakup pada *sequence diagram* sehingga semakin banyak use case yang didefinisikan maka *sequence diagram* yang harus dibuat juga semakin banyak. Gambar 2.5 merupakan contoh *sequence* diagram.



Gambar 2.5 Contoh Sequence Diagram

Sumber : [DHA-03]

2.6 Pengujian Perangkat Lunak

Untuk mengetahui apakah suatu produk hasil rekayasa, khususnya perangkat lunak sudah dapat berjalan dengan baik sesuai dengan kebutuhan, dapat dilakukan uji pada setiap fungsi yang ada sekaligus mencari kesalahan yang ada pada sistem. Salah satu metode untuk melakukan uji fungsionalitas adalah metode *black-box*. Uji *black-box* dibutuhkan untuk menguji perilaku yang fokus pada persyaratan fungsional perangkat lunak. Metode ini memungkinkan pengembang perangkat lunak untuk menentukan *input* untuk melaksanakan semua persyaratan fungsional program. Uji *black-box* digunakan untuk mengunkap kesalahn seperti berikut [PRE-10:495]:

1. Fungsi yang hilang atau tidak benar.
2. Kesalahan antar-muka.
3. Kesalahan pada struktur data atau basis data eksternal.
4. Kesalahan perilaku dan performa.
5. Kesalahan inisialisasi dan terminasi.

Uji *black-box* ini akan dilakukan pada aplikasi *push notification* pada perangkat Android. Pada aplikasi *push notification* pada perangkat Android juga akan dilengkapi dengan pengujian performa.

Pengujian performa merupakan bagian dari pengujian sistem. Pengujian performa dirancang untuk menguji kinerja *run-time software* dalam sistem yang terintegrasi. Pengujian performa terjadi sepanjang proses pengujian. Pengujian performa sering digabungkan dengan *stress testing* dan biasanya memerlukan *hardware* dan *software instrumentasi*. *Performance testing* biasanya dilakukan sebagai salah satu bagian terpenting dan mandatori dalam pengujian aplikasi non-fungsionalitas. Aspek-aspek seperti kecepatan (baik dari sisi waktu respon aplikasi, data rendering dan pengaksesan), kapasitas, stabilitas dan skalabilitas [PRE-01].

2.7 Sistem Operasi Android

Sistem operasi Android adalah *platform* terbuka dari Google yang dirancang untuk perangkat mobile. Tujuan dari *platform* terbuka tersebut adalah untuk mempercepat inovasi dalam konsumen perangkat bergerak dan menawarkan pengalaman dalam perangkat bergerak yang lebih kaya, lebih murah dan lebih baik [SAF-12].

Android adalah *platform open source.* Mulai dari *lob-level* Linux *modul*, semua *native library*, sampai *application framework* untuk semua aplikasi bersifat terbuka. Android dilisensikan di bawah lisensi *Business-friendly licences* (Apache/MIT) sehingga orang lain dapat memperpanjang dan menggunakan untuk berbagai keperluan secara bebas. Pengembang memiliki akses untuk sumber kode pada seluruh *platform*. Sedangkan produsen dapat dengan mudah menanamkan sistem operasi Android pada perangkat keras tertentu.

Tujuan utama diciptakannya Android adalah untuk perangkat *mobile.* Ketika merancang Android terdapat kendala di mana perangkat *Mobile* kemungkinan tidak akan ada perubahan yang signifikan di masa mendatang. Faktor yang pertama, sumber tenaga dari perangkat *mobile* adalah baterai dan performa baterai kemungkinan tidak akan berkembang dengan pesat dalam waktu dekat. Faktor yang kedua, perangkat *mobile* berukuran kecil, berarti bahwa perangkat *mobile* akan selalu memiliki banyak keterbatasan dalam hal memori dan kecepatan. Android dirancang untuk berjalan pada segala macam jenis perangkat keras. Android tidak membuat asumsi tentang ukuran layar perangkat, resolusi, *chipset,* dan sebagainya.

Platform Android adalah sebuah software stack produksi Google untuk perangkat mobile yang terdiri atas sistem operasi, middleware, dan key applications. Android Standard Development Kit (Android SDK) adalah tools API (Application Programming Interface) yang diperlukan untuk mulai mengembangkan aplikasi pada platform android dengan menggunakan bahasa pemrograman Java. Android berjalan didalam Dalvik Virtual Machine, sebuah virtual machine yang khusus dirancang dan dikustomisasi untuk memastikan feature – feature berjalan lebih efisien pada perangkat mobile.

2.7.1 Fitur – fitur Android

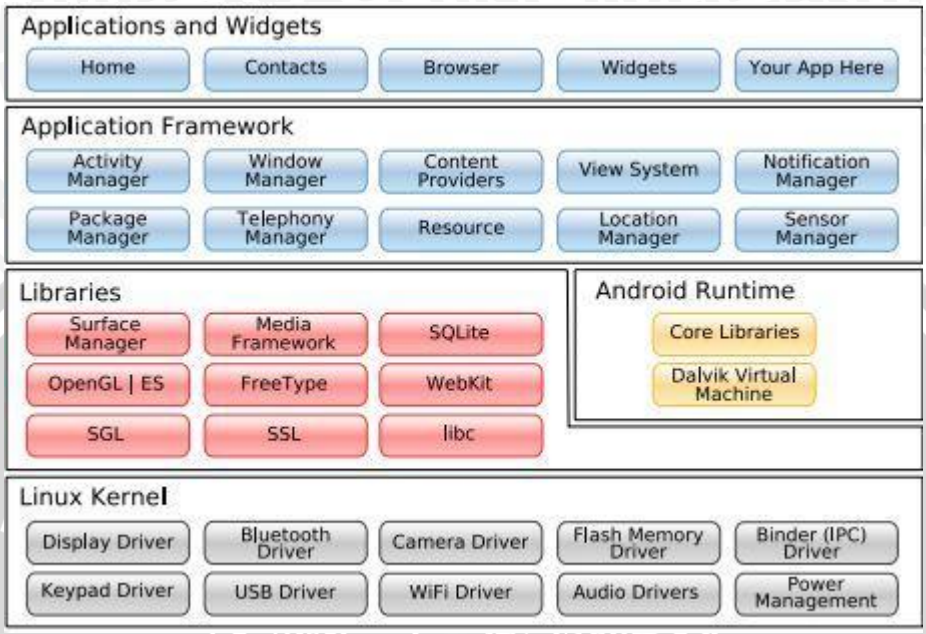
Android menyediakan berbagai fitur dan API (Application Programming Interface) yang memungkinkan akses ke berbagai fitur yang dimiliki perangkat mobile. Adapun fitur – fitur yang ditawarkan Android adalah [SAF-12] :

1. Framework Application, yang memudahkan konsep reuse komponen. Programmer dapat menggunakan beberapa fungsi yang telah disediakan.
2. Dalvik virtual machine, yaitu virtual machine yang khusus dioptimasi untuk perangkat mobile. Tiap aplikasi dalam android memiliki instance virtual machine yang dapat bekerja secara efisien dalam lingkungan memori yang terbatas.
3. Integrated browser, browser yang terintegrasi. Web browser berbasis WebKit engine terdapat pada browser default Android atau pun dapat diintegrasikan dengan aplikasi lain.
4. Optimized graphics, grafis yang dioptimalkan dan didukung oleh libraries grafis 2D dan 3D berdasarkan spesifikasi OpenGL ES 1.0 yang mendukung akselerasi hardware.
5. SQLite, sebagai tempat penyimpanan data secara terstruktur. Basis data relational yang ringan namun sangat powerfull.
6. Media Support, dukungan media untuk berbagai format audio, video, dan gambar yang umum di pasaran (MPEG4, H.264, MP3, AAC, AMR, JPG, PNG, GIF), GSM Telephony (mendukung fungsi komunikasi GSM, tergantung hardware)
7. Bluetooth, EDGE, 3G, dan WiFi. Mendukung komunikasi pada jaringan (tergantung hardware).
8. Kamera, GPS, kompas, dan accelerometer, mendukung berbagai fitur yang disediakan oleh hardware (tergantung hardware).

Kakas pengembangan yang lengkap, termasuk device emulator, tools untuk debugging, profiling memori dan performa, dan plugin untuk Eclipse Integrated Development Enviroment (IDE).

2.7.2 Arsitektur Android

Secara garis besar arsitektur android dapat digambarkan dan dijelaskan sebagai berikut :



Gambar 2.6 Arsitektur Android

Sumber : [SAF-12:9]

Pada gambar 2.6 ditunjukkan bahwa arsitektur android terbagi menjadi lima bagian yaitu *Application, Application Framework, Libraries, Android Runtime,* dan *Linux Kernel.*

2.7.2.1 *Application dan Widget*

Application dan widget merupakan program yang langsung berhubungan dengan user. Baik program yang merupakan bawaan dari android sendiri maupun program yang dibuat oleh developer menggunakan bahasa perograman java. Contoh program bawaan dari platform android sendiri adalah email client, program SMS, calendar, maps, web browser, contact, dan sebagainya.

2.7.2.2 *Application Framework*

Lapisan ini berisi sekumpulan API yang dapat digunakan oleh programmer maupun core application dari android. Lapisan ini dirancang untuk memudahkan penggunaan komponen dari android sendiri (reuse). Aplikasi manapun dalam android dapat berbagi fungsi sehingga aplikasi lain dapat memanfaatkannya. Aplikasi pada android disusun atas beberapa komponen :

- Sekumpulan views. Digunakan untuk mengatur tampilan pada aplikasi. Contohnya adalah lists, grids, text box, button, bahkan embeddable web browser.
- Content providers. Komponen yang mengatur agar aplikasi dapat mengakses resources dari aplikasi lain (seperti contacts), atau berbagi data dengan aplikasi lain.
- Resource manager. Menyediakan akses kepada resource non-code seperti localized string, grafik dan file layout.
- Notification manager. Memungkinkan agar suatu aplikasi dapat menampilkan peringatan yang dapat di kostumasi pada status bar.
- Activity manager. Mengatur siklus aplikasi dan navigasi antar aplikasi yang sedang berjalan.

2.7.2.3 *Libraries*

Android mendukung beberapa library C/C++ yang digunakan pada berbagai komponen android. Kemampuan ini dapat diakses oleh developer melalui android application framework. Beberapa library diantaranya adalah :

- System C library. Implementasi library C standart (libc).
- Media libraries. Mendukung berbagai format multimedia (termasuk MPEG4, H.264, MP3, AAC, AMR, JPG, PNG).
- Surface Manager. Mengatur akses ke subsistem display.
- LibWebCore. Engine web browser modern.
- SGL. Engine grafis 2D.
- 3D Library. Implementasi OpenGL ES 1.0 yang mendukung akselerasi hardware.

- FreeType. Rendering untuk bitmap dan vector font.
- SQLite. Basis data relasional yang kecil namun sangat ampuh.

2.7.2.4 *Android Runtime*

Pada setiap aplikasi pada android memiliki proses-nya masing-masing. Setiap aplikasi tersebut memiliki instans dari Dalvik Virtual Machine (VM). Dalvik Virtual Machine dirancang agar suatu device dapat menjalankan beberapa VM secara efisien. Dalvik VM mengeksekusi file dengan format Dalvik Executable format (.dex) yang dirancang untuk meminimalkan memory footprint.

Dalvik VM berbasis register, dan dapat menjalankan kelas-kelas yang dikompilasi dangan bahasa pemrograman java dan ditrasformasikan menjadi format .dex. dalvik VM sendiri bergantung pada kernel linux untuk fungsi dasarnya, seperti threading, dan manajemen memori secara low-level.

2.7.2.5 *Linux Kernel*

Android menggunakan Kernel Linux versi 2.6 sebagai sistem utama. Fungsi kernel yang digunakan antara lain untuk keamanan, manajemen memori, manajemen proses, manajemen jaringan dan driver model. Kernel juga berfungsi sebagai layer abstrak antara hardware dan lapisan lainnya pada software stack.

2.8 Pembelajaran Elektonik (e-learning)

E-learning mengandung pengertian yang luas, sehingga banyak pakar yang menguraikan tentang definisi e-learning dari berbagai sudut pandang. Salah satu definisi yang cukup dapat diterima banyak pihak misalnya dari Darin E. Hartley yang menyatakan: *e-learning merupakan suatu jenis belajar mengajar yang memungkinkan tersampaikannya bahan ajar ke siswa dengan menggunakan media internet, Intranet atau media jaringan komputer lain.*

LearnFrame.Com dalam Glossary of *eLearning Terms* meatakan suatu definisi yang lebih luas bahwa: e-learning adalah sistem pendidikan yang menggunakan aplikasi elektronik untuk mendukung belajar mengajar dengan media

Ada 3 (tiga) fungsi e-learning terhadap kegiatan pembelajaran di dalam kelas, yaitu sebagai tambahan (suplemen), pelengkap (komplemen), atau pengganti (substitusi). Dikatakan berfungsi sebagai suplemen (tambahan), apabila peserta didik mempunyai kebebasan memilih, apakah akan memanfaatkan materi e-learning atau tidak. Dalam hal ini, tidak ada kewajiban/keharusan bagi peserta didik untuk mengakses materi e-learning. Sebagai komplemen berarti materi e-learning diprogramkan untuk menjadi materi *reinforcement* (pengayaan) atau remedial bagi peserta didik di dalam mengikuti kegiatan pembelajaran konvensional. Materi e-learning juga dapat berfungsi sebagai *enrichment*, apabila kepada peserta didik yang dapat dengan cepat menguasai atau memahami materi pelajaran yang disampaikan guru secara tatap muka (*fast learners*) diberikan kesempatan untuk mengakses materi e-learning yang memang secara khusus dikembangkan untuk mereka.

Selain itu, e-learning dapat menjangkau peserta didik dalam cakupan yang luas. Dengan fleksibilitas waktu dan tempat, maka jumlah peserta didik yang dapat dijangkau melalui kegiatan pembelajaran elektronik semakin lebih banyak. Ruang dan tempat serta waktu tidak lagi menjadi hambatan. Siapa saja, di mana saja, dan kapan saja, seseorang dapat belajar. Interaksi dengan sumber belajar dilakukan melalui internet [TUR-09].

2.8.1 *Learning Management System (LMS)*

Proses pembelajaran e-learning memerlukan sistem yang mampu mengelola pembelajaran secara *online*. Salah satu aplikasi pendukung e-learning adalah *Learning Management System* (LMS), yang dapat berfungsi untuk mengatur tata laksana penyelenggaraan pembelajaran di dalam model e-learning. [ADR-08]

LMS merupakan sebuah perangkat lunak atau *software* untuk keperluan administrasi, dokumentasi, pencarian materi, laporan sebuah kegiatan, pemberian materi materi pelatihan kegiatan belajar mengajar secara *online* yang terhubung ke internet [ELL-09].

Pada umumnya LMS dibangun berbasis *web*, yang berjalan pada sebuah *web server* dan dapat diakses oleh pesertanya melalui *web browser (web client). Server* ditempatkan di suatu tempat yang dapat diakses darimanapun oleh pesertanya dengan memanfaatkan koneksi internet. LMS menyediakan akses ke layanan e-learning bagi siswa, pengajar, dan administrator. Layanan ini termasuk akses control, penyediaan konten pembelajaran, alat komunikasi, dan administrasi kelompok pengguna [COB-06]. Pada *tool* akses *control instruktur, educator,* dan admin mengatur sebagaimana mestinya sehingga hanya peserta yang terdaftar yang dapat mengakses dan melihatnya. Selain menyediakan pengontrolan, LMS juga menyediakan barbagai *tool*s yang sangat berguna bagi keberlangsungan belajar dan mengajar.

2.8.2 *Modular Object Oriented Dynamic Learning Enviroment (MOODLE)*

Moodle (*Modular Object-Oriented Dynamic Learning Environment*) adalah salah satu alternatif LMS yang bersifat *open source*. Moodle memiliki anggota komunitas lebih dari 350.000. Moodle tersedia dalam 75 bahasa dan digunakan oleh lebih dari 15 juta siswa di seluruh dunia [FAC:13]. Moodle pertama kali diperkenalkan oleh Martin Daugiamas, seorang doctor computer scientist dan educator yang mengembangkan *Learning Management System* di Curtin University of Technology di Kota Perth, Australia. Moodle terus dikembangkan dan akhirnya dirilis Moodle versi 1.0 pada tanggal 20 Agustus 2002 [ARA-11].

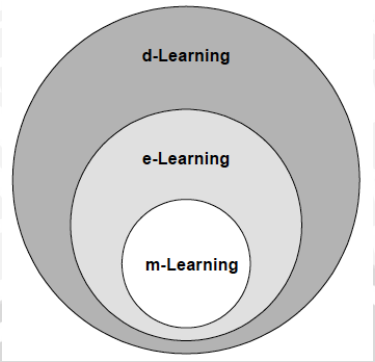
Moodle merupakan salah satu aplikasi dari konsep dan mekanisme belajar mengajar yang memanfaatkan teknologi informasi, yang dikenal dengan konsep pembelajaran elektronik atau e-learning. Moodle dapat digunakan secara bebas sebagai produk sumber terbuka (*open source*) di bawah lisensi GNU. Moodle dapat diinstal di komputer dan sistem operasi apapun yang bisa menjalankan PHP dan mendukung database SQL .[MOD-13]

Moodle adalah sebuah nama untuk sebuah program aplikasi yang dapat mengubah sebuah media pembelajaran ke dalam bentuk *web*. Moodle juga merupakan sebuah *Learning Management System* yang digunakan untuk membuat sebuah proses belajar (*learning*) bisa dilakukan secara online, powerful dan fleksibel. Moodle terus dikembangkan dan diteliti untuk meningkatkan fungsionalitas fasilitas pendidikan jarak jauh yang memanfaatkan internet.

2.9 Mobile Learning

Perubahan terkini dan signifikan dalam lingkungan belajar adalah permintaan mobilitas. Perkembangan pengguna ponsel diseluruh dunia dari tahun ke tahun semakin meningkat. Ponsel menjadi populer di kalangan mahasiswa dan sebagian besar masyarakat yang mampu membayar biaya operasional ponsel. *Mobile Learning* (m-Learning) adalah generasi berikutnya e-learning dan berdasarkan pada perangkat *mobile*. Satu keuntungan adalah ketersediaan tinggi dari perangkat tersebut: penetrasi pasar ponsel di Austria saat ini pada tingkat 81% dan jumlahnya terus bertambah. Hal ini dapat ditekankan bahwa mayoritas penduduk memiliki ponsel yang mereka miliki di tangan sebagian besar waktu. Akibatnya, m-learning akan menjadi instrumen penting untuk belajar sepanjang masa [HOL-05]. M-learning memfokuskan pada fungsi program yang lebih spesifik sehingga lebih mudah diakses.

M-learning merupakan bagian dari e-learning sehingga, dengan sendirinya juga merupakan bagian dari d-learning *(distance learning)*. Mobile learning mengacu pada penggunaan perangkat IT genggam dan bergerak, seperti PDA (Personal Digital Assistant), telepon genggam, laptop, dan tablet PC dalam pengajaran dan pembelajaran. Mobile learning memunginkan pembelajar dapat memobilisasi perangkat pembelajaran yang digunakan sehingga dapat dilakukan kapan saja dan dimana saja tanpa terbatas ruang fan waktu. Mobile learning menyajikan informasi pendidikan dan pertukaran informasi antara pembelajar dan pengajar [GEO-O4].



Gambar 2.7 Posisi m-learning bagian dari e-learning dan d-learning

Sumber : [GEO-04]

2.10 Google Cloud Messaging (GCM)

Google Cloud Messaging (GCM) pada Android adalah layanan untuk mengirim data dari *server* ke perangkat Android. Layanan GCM merupakan layanan gratis yang diberikan oleh google dan tidak mempedulikan seberapa besar kuota aplikasi. *Server* akan memberikan pesan notifikasi ke semua perangkat yang memiliki regid (ID perangkat yang terdaftar). Regid yaitu ID yang mewakili suatu perangkat yang menggunakan aplikasi tertentu, id tersebut nantinya kan dijadikan notification_key sehingga dapat menerima pesan notifikasi yang dikirim oleh server. Layanan GCDM ini didesain bukan untuk mengirimkan *konten*, tetapi hanya seperti kiriman yang ditujukan untuk pemberitahuan aplikasi anda sehingga aplikasi anda tersebut tahu kapan harus melakukan *request* ke *server* anda [DEV-13].

2.11 Teknologi *Push*

Push notification adalah sebuah teknologi penyampaian informasi dari aplikasi perangkat lunak untuk perangkat komputasi tanpa permintaan khusus dari *client*. Distribusi data pada teknologi *push* di mana data yang dipilih secara otomatis akan dikirimkan ke komputer pengguna atau perangkat *mobile (client)* secara *real time* atau pada interval yang telah ditentukan. Teknologi push digunakan untuk memperbaharui suatu berita. Pemberitahuan *push notification* dapat berupa suara dan ditampilkan dengan berbagai cara sesuai dengan model *smartphone* atau perangkat komputasi lainnya. *Push notification* dapat dicontrol sesuai dengan keinginan pengguna.

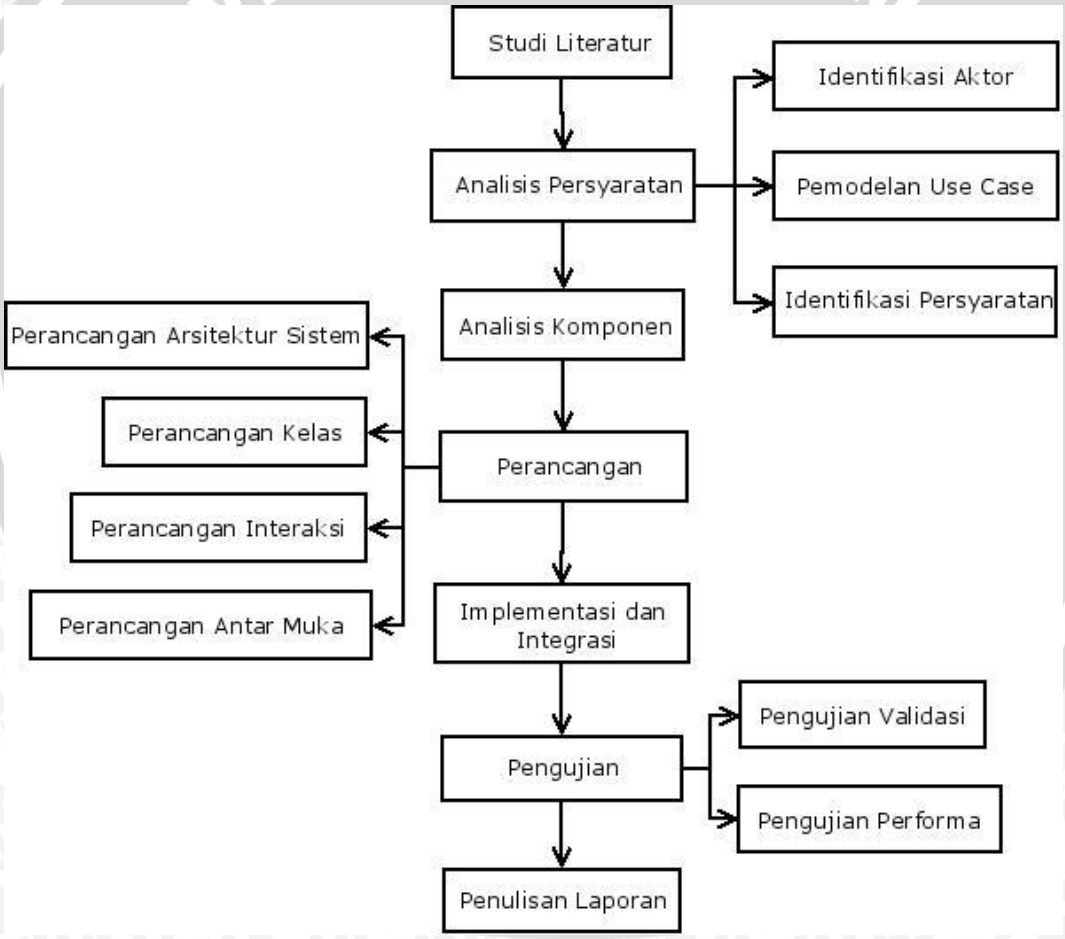
Keuntungan penting dari *push notification* dalam komputasi *mobile* adalah bahwa teknologi tidak memerlukan aplikasi khusus pada perangkat *mobile* untuk membuka pesan yang akan diterima . Hal ini memungkinkan *smartphone* untuk menerima dan menampilkan berita ketika layar perangkat terkunci dan aplikasi yang memberi *push notification* ditutup.

Perangkat dan jasa dalam teknologi *push* berbeda bergantung pada metode yang memberikan teknologi *push*. Pengembang Apple misalnya, dapat menggunakan Apple *Push Notification Service* sebagai sarana pengembangan teknologi *push* ke perangkat iOS . Tidak seperti "teknologi *pull*", dimana *client* harus melakukan permintaan data dari *server*. *Browsing web* merupakan salah satu contoh dari teknologi *pull*.

BAB III
METODOLOGI PENELITIAN

3.1 Metode Penelitian

Pada bab ini dijelaskan mengenai prosedur dan kegiatan yang akan dilakukan dalam pengerjaan skripsi, yaitu studi literatur, analisis persyaratan, analisis komponen, perancangan, implementasi dan pengujian dalam pengembangan perangkat lunak yang akan dibuat.



Gambar 3.1 Alur Proses Penelitian

Adaptasi dari [SOM-11:35]

Metodologi penelitian yang digunakan pada gambar 3.1 adalah adaptasi dari pengembangan berbasis *reuse* seperti yang dijelaskan pada bab sebelumnya Subbab 2.3 *Reuse-oriented Software Engineering*. Adaptasi yang dilakukan adalah menggunakan tahap-tahap yang terdapat pada gambar 2.1 namun menghilangkan tahap modifikasi kebutuhan. Tahap modifikasi kebutuhan bertujuan untuk menyesuaikan atau memodifikasi setiap kebutuhan pada aplikasi agar sesuai dengn komponen-komponen yang telah tersedia. Alasan mengapa tahap modifikasi kebutuhan tidak digunakan adalah komponen yang akan digunakan sudah mampu untuk memenuhi kebutuhan pada aplikasi.

3.1.1 Studi Literatur

Studi literatur merupakan penelusuran literatur yang bertujuan dalam menyusun dasar teori yang digunakan untuk menunjang skripsi. Penelusuran literatur dapat bersumber dari buku, media, pakar maupun hasil penelitian orang lain. Teori-teori pendukung tersebut meliputi:

1. Rekayasa Perangkat Lunak
2. *Software Proses Model*
    - *Reuse-Oriented Model*
3. Pola - Pola Perancangan
4. *Unified Modelling Language*
    - *Use Case* Diagram
    - *Class* Diagram
    - *Sequence* Diagram
5. Pengujian Perangkat Lunak
6. Pembelajaran Elektronik (e-learning)
    - Learning Management System (LMS)
    - *Modular Object Oriented Dynamic Learning Enviroment* (MOODLE)

7. *Mobile Learning*
8. Sistem Operasi Android
9. *Google Cloud Messaging* (GCM)
10. Teknologi *Push*

3.1.2 Analisis Persyaratan

Tujuan dari tahap analisis persyaratan adalah memahami persyaratan yang dibutuhkan oleh sistem. Analisis persyaratan merupakan langkah awal untuk menentukan aplikasi seperti apa yang akan dihasilkan. Sistem yang baik dan sesuai dengan kebutuhan pengguna sangat bergantung kepada keberhasilan dalam proses ini. Identifikasi aktor digunakan untuk menjelaskan aktor pengguna yang akan menggunakan sistem sesuai dengan peran dan kebutuhanya. Setelah itu dilakukan pemodelan dengan use case.

Untuk mempermudah dalam menganalisis sistem dibutuhkan dua jenis kebutuhan. Dua jenis kebutuhan tersebut adalah kebutuhan fungsional dan kebutuhan nonfungsional. Metode analisis yang digunakan adalah dengan pendekatan berorientasi objek yang memakai bahasa pemodelan UML (*Unified Modelling Language*). Kebutuhan fungsional akan dijelaskan dalam daftar kebutuhan yang kemudian dimodelkan dengan *use case diagram* dan dijelaskan pada table spesifikasi *use case*. *Use case diagram* digunakan untuk menggambarkan tujuan yang ingin dicapai oleh aktor di dalam sistem. Kebutuhan non-fungsional akan dijabarkan dalam bentuk tabel yang nantinya akan diuji setelah aplikasi berhasil dibuat.

3.1.3 Analisis Komponen

Tujuan dari tahap analisis persyaratan adalah memahami persyaratan yang dibutuhkan oleh sistem. Analisis persyaratan merupakan langkah awal untuk menentukan aplikasi seperti apa yang akan dihasilkan. Sistem yang baik dan sesuai dengan kebutuhan pengguna sangat bergantung kepada keberhasilan dalam proses ini. Identifikasi aktor digunakan untuk menjelaskan aktor pengguna yang akan menggunakan sistem sesuai dengan peran dan kebutuhanya. Setelah itu dilakukan pemodelan dengan use case.

Untuk mempermudah dalam menganalisis sistem dibutuhkan dua jenis kebutuhan. Dua jenis kebutuhan tersebut adalah kebutuhan fungsional dan kebutuhan nonfungsional. Metode analisis yang digunakan adalah dengan pendekatan berorientasi objek yang memakai bahasa pemodelan UML (*Unified Modelling Language*). Kebutuhan fungsional akan dijelaskan dalam daftar kebutuhan yang kemudian dimodelkan dengan *use case diagram* dan dijelaskan pada table spesifikasi *use case*. *Use case diagram* digunakan untuk menggambarkan tujuan yang ingin dicapai oleh aktor di dalam sistem. Kebutuhan non-fungsional akan dijabarkan dalam bentuk tabel yang nantinya akan diuji setelah aplikasi berhasil dibuat.

3.1.4 Perancangan

Hasil dari analisis persyaratan menjadi acuan dalam perancangan perangkat lunak. Perancangan perangkat lunak menggunakan pendekatan berorientasi objek yang kemudian dimodelkan ke dalam bentuk arsitektur sistem, diagram kelas, diagram interaksi dan perancangan antar muka. Pada perancangan sistem nantinya akan menggunakan beberapa *Design Patterns* atau pola-pola perancangan khususnya pada pembuatan diagram kelas walaupun sebenarnya secara keseluruhan juga mempengaruhi proses mekanisme didalam sistem itu sendiri. Pertimbangan penggunaan ulang (*reuse*) terhadap beberapa komponen perangkat lunak yang akan digunakan juga dibahas pada bagian ini.

3.1.5 Implementasi

Implementasi Pengembangan Aplikasi *push* notification pada perangkat Android dilakukan sesuai dengan hasil analisis persyaratan dan perancangan sistem. Selama tahap ini komponen-komponen yang telah ditentukan akan dilakukan penambahan atau perubahan ke dalam sistem. Implementasi meliputi :

- Pengembangan aplikasi sesuai hasil perancangan.
- Pembuatan basis data sesuai dengan kebutuhan sistem.
- Penggunaan komponen *Google Cloud Messaging* akan membantu menhubungkan antara pengguna *mobile device* dengan Moodle
- Pembuatan antarmuka sistem dibangun sesuai dengan perancangan antarmuka.

3.1.6 Pengujian

Pengujian sistem pada penelitian ini dilakukan agar dapat menunjukan bahwa sistem telah mampu bekerja sesuai dengan spesifikasi dari kebutuhan sistem dan menemukan celah kesalahan (*error*) yang mungkin secara tidak sengaja terdapat dalam sistem aplikasi. Pengujian dilakukan dengan teknik *blackbox testing* dimana menguji fungsional aplikasi apakah sesuai dengan spesifikasi dan juga tingkat keakuratan serta pewaktuan yang dirasakan oleh pengguna menggunakan pengujian performa.

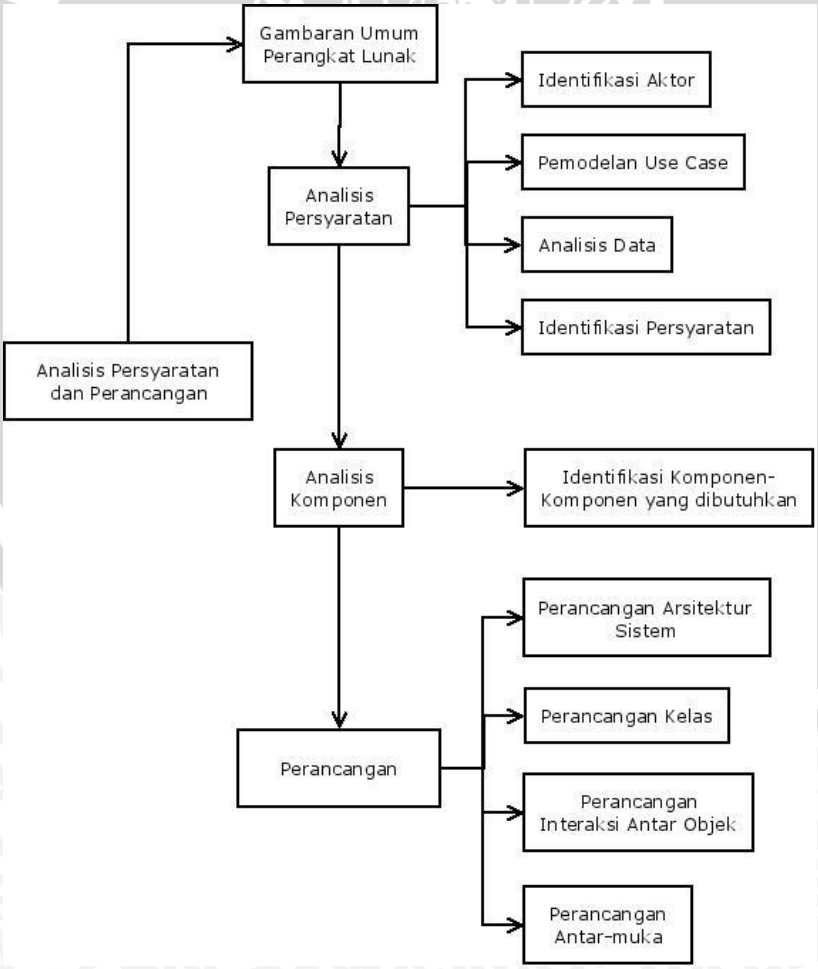
Pengujian performa dilakukan untuk mengetahui apakah kebutuhan non-fungsional pada sistem telah terpenuhi. Pengujian performa dilakukan dengan cara pengujian aplikasi oleh 30 pengguna yang telah terdaftar pada e-learning Moodle dan menentukan keakuratan isi seperti mata kuliah, tipe perubahan dan batas waktu serta waktu sampainya notifikasi kepada pengguna.

3.1.7 Penulisan Laporan

Laporan penelitian ditulis setelah semua proses pengerjaan skripsi dilalui. Laporan berisi dokumentasi perancangan sistem beserta saran untuk memberikan pertimbangan dan pengembangan aplikasi selanjutnya.

BAB IV
ANALISIS PERSYARATAN DAN PERANANGAN

Pada bab ini akan dibahas mengenai dua hal yaitu identifikasi kebutuhan dan perancangan perangkat lunak *push notification* berbasis Moodle pada perangkat Android. Analisis persyaratan berguna untuk mengetahui kebutuhan dari pengguna sedangkan perancangan digunakan untuk mendesain sistem sesuai dengan persyaratan yang telah didefinisikan. Proses modifikasi kebutuhan tidak dilakukan pada perancangan aplikasi berbasis *reuse* ini. Hal tersebut dikarenakan seluruh kebutuhan sudah terpenuhi dengan komponen yang ada. Diagram blok identifikasi kebutuhan dan perancangan digambarkan pada gambar 4.1.



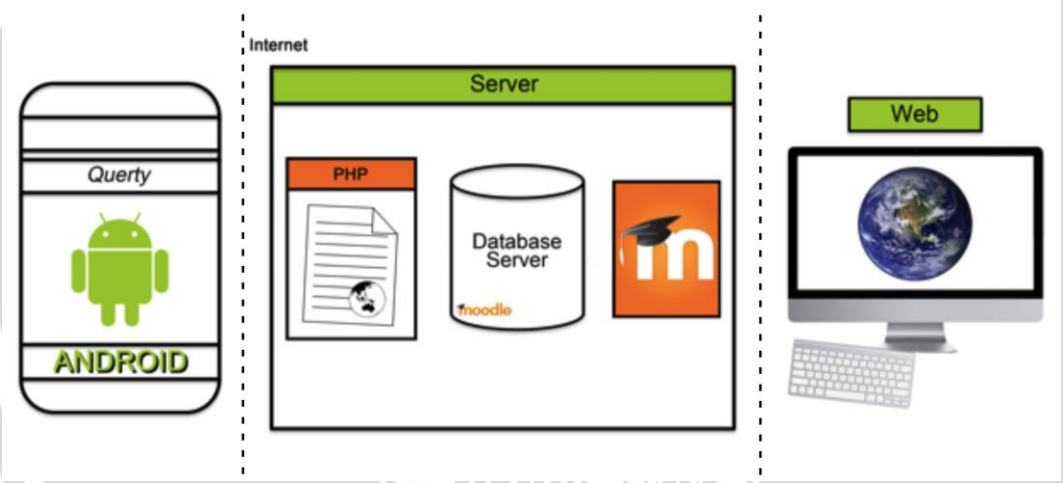
Gambar 4.1 Alur Analisis Persyaratan dan Perancangan

4.1 Analisis Persyaratan

Tahap analisis kebutuhan terdiri dari tiga langkah yaitu analisa kebutuhan meliputi identifikasi aktor, pemodelan use case dan kebutuhan fungsional dan non-fungsional.

4.1.1 Gambaran Umum Perangkat Lunak

Untuk lebih memahami bagaimana interaksi antara aplikasi yang dibuat dengan pembelajar, pada sub bab ini disertakan ilustrasi interaksi pembelajar dengan e-learning Moodle beserta komponen penunjangnya. Berikut adalah cara kerja antara pembelajar dengan e-learning Moodle.



Gambar 4.2 Cara Kerja Aplikasi

Ilustrasi pada gambar 4.2 merupakan gambaran secara umum dari sudut pandang pembelajar dengan e-learning Moodle. Pada gambar 4.2 terlihat hubungan antara e-learning Moodle dengan pembelajar yang diasumsikan dengan sebuah *device* Android. Dengan menggunakan komputasi seperti *computer dekstop* yang terhubung dengan koneksi internet maka sebuah e-learning Moodle dapat dikendalikan oleh pengajar. *File* bertipe PHP merupakan bahasa pemrograman penghubung Moodle agar dapat diakses oleh *device* Android.

Perangkat lunak yang akan dikembangkan merupakan sebuah aplikasi *push notification* berbasis Moodle di dalam sistem operasi Android. Aplikasi ini merupakan proses pengembangan dari e-learning Moodle berbasis *web* menjadi aplikasi m-learning. Aplikasi *push notification* akan memudahkan pembelajar dalam memonitor kegiatan yang terjadi pada e-learning Moodle. Pembelajar akan mendapatkan *push notification* setiap kali ada perubahan dalam e-learning Moodle. Perubahan yang dapat menampilkan *push notifikasi* yaitu penambahan materi, quiz, online text, dan lain sebagainya.

Aplikasi *push notification* akan berjalan apabila seorang pembelajar telah terdaftar pada e-learning Moodle. Dalam keadaan *login* aplikasi *push notification* akan secara *real time* mengirimkan notifikasi, dan apabila pembelajar menghendaki untuk membuka halaman e-learniing Moodle maka akan terhubung langsung dengan e-learning Moodle. Aplikasi dapat berjalan dengan bantuan koneksi internet. Aplikasi *push notification* akan menampilkan *push notification* dan list dari keseluruhan notifikasi. Aplikasi *push notification* akan menampilkan sesuai dengan mata kuliah yang diambil seorang pembelajar.

4.1.2 Identifikasi Aktor

Pada tabel 4.1 dibawah ini dijelaskan jenis-jenis aktor yang terdapat ada aplikasi *push notificaion* dan deskripsi dari perilaku aktor-aktor tersebut dalam sistem.

Tabel 4.1 Identifikasi Aktor

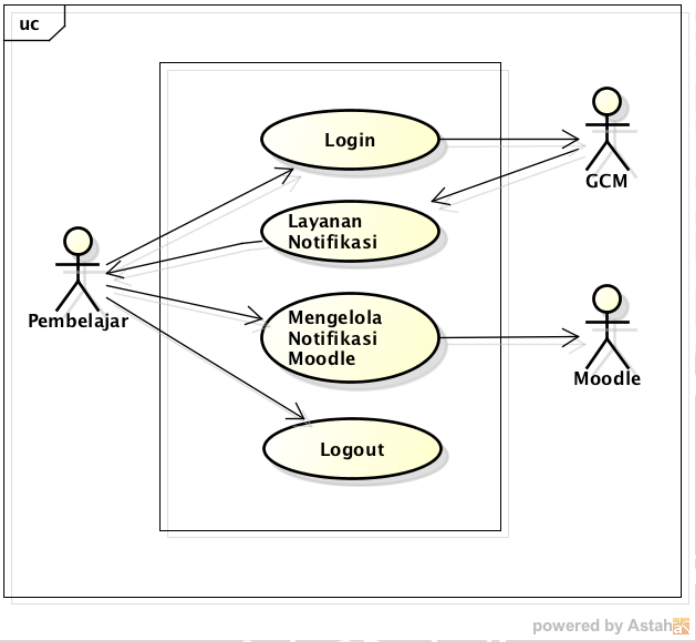
Aktor	Deskripsi
Pembelajar	• Pembelajar merupakan aktor utama yang menjadi fokus pelayanan dalam notifikasi Moodle. • Pembelajar dapat memanfaatkan fitur dari aplikasi mulai dari menerima berita terkini mengenai perubahan yang terjadi di e-learning, melihat list dari keseluruhan notifikasi yang muncul, mengunjungi e-learning Moodle sesuai dengan notifikasi yang diterima, serta dapat menghapus notification.

Aktor	Deskripsi
Pengajar	• Aktor yang bertanggung jawab atas pengelolaan mata kuliah pada e-lerning Moodle.
GCM	• Sub sistem di luar dari sistem yang dibuat
Moodle	• Sub sistem endukung dari sistem yang dibuat.
Admin	• Aktor yang bertanggung jawab atas pengelolaan Moodle.

Pengajar dan admin merupakan aktor pendukung yang bertugas untuk mengelola e-learning Moodle bukan sebagai pengguna aplikasi *push notification*.

4.1.3 Pemodelan Use Case

Perancangan use case akan menggunakan RUP (Rational Unified Process) style dimana terdiri dari beberapa bagian yaitu Brief Description, Actor, Basic Flow, Alternate Flow, Key Scenario, Pre-Condition, dan Post-Condition. Brief Description menjelaskan secara singkat siapa yang berinteraksi dengan Use Case dan tujuan apa yang dicapai oleh Use Case. Basic Flow menjelaskan langkah – langkah interaksi antara aktor dengan sistem yang pada umumnya akan terjadi. Alternate flow menjelaskan langkah-langkah pada beberapa kasus interaksi yang berbeda dari basic flow seperti varian operasional, kasus unik (odd cases) maupun exception error. Key Scenarios menyebutkan beberapa skenario yang terdiri dari basic flow dan alternate flow yang mungkin akan berjalan pada sistem. Pre-Condition menjelaskan kondisi awal yang mungkin perlu dipenuhi oleh use case sebelum menjalankan skenario. Post-Condition menjelaskan kondisi setelah menjalankan skenario use case.



Gambar 4.3 Diagram Use Case Sistem

Penjelasan diagram *use case* pada gambar 4.3 layanan notifikasi Moodle dijelaskan sebagai berikut :

a. *Use case login* untuk menangani proses autentifikasi *user* yang akan masuk ke dalam sistem. Use case login melibatkan dua actor yaitu pembelajar dan GCM. Pemebelajar melakukan login dan akan mengirimkan data untuk mendaftarkan ke dalam GCM.
b. Menerima notifikasi merupakan layanan yang akan didapat oleh *user* apabila terjadi perubahan pada moodle yang dilakukan oleh pengajar. Sistem akan memengirimkan pemberitahuan terkini dalam bentuk notifikasi sesuai dengan pelajaran yang diambil. GCM akan mengirimkan notifikasi kepada pembelajar.
c. Mengelola notifikasi Moodle digunakan untuk menangani proses pengelolaan notifikasi oleh *user*. Pengelolaan ini mencangkup melihat kumpulan notifiksi dalam bentuk list dan meliputi kenjungan ke dalam Moodle serta menghapus notifikasi. Pembelajar dapat melakukan *action* pada notifikasi berupa penghapusan dan mengunjungi e-learning yang langsung terhubung dan sesuai dengan Moodle.
d. *Logout* digunakan apabila seorang pembelajar akan keluar dari sistem.

Tabel 4.2 Spesifikasi *Use Case - Login*

Name	*Login*
Brief Description	*Use case login* untuk menangani proses autentifikasi *user* yang akan masuk ke dalam sistem.
Actor	Pembelajar dan GCM
Basic Flow	1. Menampilkan halaman *login* *Use case* dimulai ketika pembelajar akan masuk ke dalam sistem. Sistem akan menampilkan form *login* yang berisi kolom *username* dan *password*. 2. Masukkan *username* dan *password* Pembelajar memasukkan *username* dan *password* sesuai dengan yang terdaftar pada Moodle. 3. Mengecek *username* dan *password* Sistem memeriksa apakah *username* dan *password* yang dimasukkan valid atau tidak. 4. Pengiriman data ke GCM Data yang telah valid akan dikirimkan ke GCM untuk mendapatkan layanan selanjutnya 5. Menampilkan halaman awal Sistem akan menampilkan halaman awal pembelajar dari layanan notifikasi Moodle. *Use case* selesai
Alternative Flow	1. Gagal *login* AT BF mengecek *username* dan *password*, jika *username* dan *password* yang dimasukkan tidak sesuai maka kembali ke proses BF menampilkan halaman *login* dan sistem akan menampilkan pesan bahwa proses *login* gagal. 2. Field kosong AT BF *field* kosong maka sistem akan menampilkan pesan untuk mengisi field *username* atau password yang masih kosong. 3. Tidak ada jaringan internet AT BF tidak ada jaringan internet akan mengakibatkan pembelajar tidak dapat melakukan *login*.

Key Scenarios	1. *Basic Flow* 2. *Basic Flow* + AF Gagal *Login* 3. *Basic Flow* + AF field kosong 4. *Basic Flow* + AF tidak ada jaringan internet
Pre-Condition	Pembelajar telah terdaftar dalam Moodle
Post Condition	Menampilkan halaman utama pembelajar

Tabel 4.3 Spesifikasi *Use Case* – Layanan Notifikasi

Name	Layanan Notifikasi
Brief Description	*Use case* layanan notifikasi merupakan layanan yang akan didapat oleh user apabila terjadi perubahan pada Moodle yang dilakukan oleh pengajar.
Actor	Pembelajar dan GCM
Basic Flow	1. Membiarkan aplikasi dalam kondisi *login* dan selalu terhubung dengan koneksi internet *Use case* dimulai ketika pembelajar telah masuk pada halaman utama dan tersambung dengan koneksi internet. 2. Pengajar melakukan tindakan pada Moodle Sistem akan mengirimkan perubahan yang terjadi ke pembelajar sesuai dengan pelajar yang dipilih pembelajar. 3. Pengiriman data dari GCM Perubahan yang terjadi pada Moodle akan dikirimkan oleh GCM sesuai dengan data pada saat awal *login.* 4. Menerima notifikasi Pembelajar akan menerima notifikasi secara *real time* sesuai dengan perubahan yang terjadi. *Use case* selesai
Alternative Flow	1. Tidak dalam keadaan *login* Ketika terjadi perubahan pada Moodle maka pembelajar tidak akan mendapatkan notifikasi.

Key Scenarios	1. *Basic Flow* 2. *Basic Flow* + AF tidak dalam keadaan *login*
Pre-Condition	Pembelajar telah login dan memiliki akses jaringan internet
Post Condition	Menampilkan notifikasi secara *real time*

Tabel 4.4 Spesifikasi *Use Case* – Mengelola Notifikasi

Name	Mengelola Notifikasi Moodle
Brief Description	*Use case* mengelola notifikasi Moodle digunakan untuk menjelaskan bagaimana pembelajar dapat melakukan pemantauan kembali mengenai notifikasi yang telah diterima.
Actor	Pembelajar dan Moodle
Basic Flow	1. Menampilkan kumpulan notifikasi *Use case* dimulai ketika pembelajar ingin melihat seluruh notifikasi yang telah diterima. Pembelajar masuk ke dalam sistem dan sistem menampilkan kumpulan notifikasi yang telah diterima. 2. Mengunjungi Moodle Pembelajar memilih dan mengklik tepat pada notifikasi yang diinginkan untuk dikunjungi dan sistem akan langsung terhubung dengan Moodle sesuai dengan notifikasi yang telah diterima. 3. Menghapus notifikasi Sistem menampilkan fungsi hapus pada setiap notifikasi dan dapat menghapus notifikasi yang diinginkan. Sistem juga menyediakan layanan menghapus notifikasi secara keseluruhan. *Use case* selesai
Alternative Flow	-
Key Scenarios	*1. Basic Flow*
Pre-Condition	Telah mendapatkan notifikasi, dalam keadaan *login* dan terkoneksi dengan internet
Post Condition	Notifikasi pada *list* notifikasi akan tersisa sesuai tindakan penghapusan.

Tabel 4.5 Spesifikasi *Use Case – Logout*

Name	Logout
Brief Description	*Use case* logout merupakan layanan yang berfungsi untuk mengakhiri layanan notifikasi atau mengakhiri sistem dengan pembelajar.
Actor	Pembelajar
Basic Flow	*Use case* dimulai ketika pembelajar ingin mengakhiri penggunaan sistem. Pembelajar memilih fungsi *logout* pada sistem, sistem akan menyajikan pertanyaan untuk meyakinkan ingin *logout* atau tidak. Apabila setuju ingin *logout* maka pembelajar akan keluar dari sistem. *Use case* selesai
Alternative Flow	-
Key Scenarios	*1. Basic Flow*
Pre-Condition	Pembelajar telah *login* dan memiliki akses jaringan internet
Post Condition	Pembelajar keluar dari sistem.

4.1.4 Analisis Data

Analisis data bertujuan untuk mendapatkan struktur data dan penyimpanan data yang dibutuhkan perangkat lunak. Pada pembuatan aplikasi *push notification* berbasis Moodle akan menggunakan *database* yang telah tersedia pada Moodle. Namun untuk melengkapi kebutuhan fungsional pada aplikasi *push notification* ditambahkan tabel yang yang berfungsi untuk mengkonfigurasikan Android dengan Moodle. Pada analisis data ini menuliskan kebutuhan tempat penyimpanan data yang belum tersedia pada e-learning Moodle dan basis data untuk kebutuhan aplikasi pada Android. Berdasarkan analisis kebutuhan, aplikasi push notification ini membutuhkan :

Tabel 4.6 Analisis Data Moodle

No	Kebutuhan	Keterangan
1	Identitas kode unik penyimpanan data identitas setiap pembelajar yang *login*	Berisi identitas setiap pembelajar yang login. Dijadikan kunci unik per data yang nantinya akan terhubung dengan data pada Android.
2	Penyimpanan data pembelajar yang *login*	Kolom untuk menyimpan data *username* pembelajar dan akan disesuaikan dengan tabel data pembelajar yang telah terdaftar pada e-learning Moodle.
3	Penyimpanan registasi ID yang di dapat dari GCM setelah melakukan *login*	Untuk menyimpan registrasi id yang didapat dari GCM.

Tabel 4.7 Analisis Data Android

No	Kebutuhan	Keterangan
1	Identitas kode unik penyimpanan data identitas setiap pembelajar yang *login*	Berisi identitas setiap pembelajar yang login. Dijadikan kunci unik per data yang nantinya akan terhubung dengan data pada e-learning Moodle.
2	Penyimpanan data pembelajar yang *login*	Kolom untuk menyimpan data *username* pembelajar dan akan disesuaikan dengan tabel data pembelajar yang telah terdaftar pada e-learning Moodle.
3	Penyimpanan password	Berisi password yang sesuai dengan e-learning Moodle.

4	Menyimpan data notifikasi yang diterima dari e-learning Moodle	Berisi notifikasi yang sesuai dengan *database* pada e-learning Moodle.
5	Menyimpan alamat notifikasi agar dapat terhubung dengan e-learning Moodle.	Berisi URL notifikasi yang diterima

Analisis data pada aplikasi *push notification* pada perangkat Android disesuaikan dengan data e-learning Moodle yang telah tersedia.

4.1.5 Persyaratan Fungsional dan Non-fungsional Sistem

Daftar kebutuhan ini disebut dengan *Software Requirement Specification* (SRS). Pada daftar kebutuhan dijelaskan spesifikasi kebutuhan fungsional Pengguna. Pada setiap spesifikasi kebutuhan terdapat aktor yang bertindak terhadap spesifikasi tersebut. Detail daftar kebutuhan fungsional pengguna ditunjukkan pada tabel 4.8

Tabel 4.8 Persyaratan Fungsional Sistem

Kode SRS	Kebutuhan	Aktor
SRS_PN_001	Sistem dapat melayani pembelajar dalam melakukan login bagi yang telah terdaftar pada Moodle.	Pembelajar
SRS_PN_002	Sistem mampu mengirimkan data atau informasi tentang berita terkini mengenai perubahan yang terjadi di moodle.	Pembelajar
SRS_PN_003	Sistem mampu menyimpan notifikasi yang telah diterima dan menampilkan dalam bentuk kumpulan notifikasi.	Pembelajar
SRS_PN_004	Sistem mampu menghubungkan dan menyesuaikan notifikasi yang dikirimkan dengan e-learning moodle.	Pembelajar
SRS_PN_005	Sistem mampu menghapus notifikasi yang telah dikirimkan.	Pembelajar

SRS_PN_006	Sistem dapat melayani pembelajar untuk keluar dari sistem.	Pembelajar

Kebutuhan non-fungsional juga harus disediakan oleh sistem. Kebutuhan non-fungsional merupakan batasan layanan atau fungsi yang ditawarkan sistem seperti batasan waktu, batasan pengembangan proses, dan standarisasi. Daftar kebutuhan non-fungsional dapat dilihat pada Tabel 4.9

Tabel 4.9 Persyaratan Non-Fungsional Sistem

Parameter	Deskripsi Kebutuhan
Performance	Sistem harus dapat memiliki akurasi isi, dan pewaktuan (*timing*) yang sesuai dengan perubahan yang terjadi pada e-learning Moodle: • pewaktuan (*timing*) terlihat pada kecepatan waktu sampainya notifikasi pada *device* saat terjadi perubahan pada Moodle. • Akurasi terlihat pada kesesuaian antara isi dari notifikasi dengan isi pada Moodle. Isi pada notifikasi berupa mata kuliah, tipe perubahan, dan batas waktu.

4.2 Analisis Komponen

Proses analisis komponen diawali dengan mencari semua komponen yang berhubungan dengan pokok bahasan skripsi. Langkah selanjutnya adalah memilih beberapa komponen yang sesuai dengan kebutuhan yang diperlukan dalam aplikasi. Komponen yang dapat memenuhi kebutuhan akan langsung dipakai dalam aplikasi, sedangkan komponen yang tidak sesuai akan mengalami tahap modifikasi kebutuhan. Daftar komponen yang digunakan pada aplikasi disajikan dalam bentuk tabel 4.10

Tabel 4.10 Daftar Komponen Aplikasi

Nama Library	Nama Komponen	Deskripsi Komponen
GCM	GCM	Komponen GCM merupakan komponen untuk memberikan notifikasi secara real time kepada user yang sedang *login*. Komponen ini akan diletakkan pada *package* pushnotification. Komponen ini terdiri dari method register untuk mendaftarkan akun ke *web services* google, dan method message handler untuk menerima notifikasi terbaru. Komponen GCM dapat digunakan apabila telah mendaftar secara gratis melalui https://cloud.google.com/ untuk mendapatkan API Key dan sender ID
SQLite	SQLite	SQLite merupakan library database yang dimiliki sistem operasi Android yang digunakan untuk mengelola data pada aplikasi.

Perancangan dengan komponen bertujuan untuk merancang komponen-komponen yang telah diperoleh sesuai dengan spesifikasi kebutuhan. Perancangan tersebut mencakup penambahan maupun perubahan pada komponen-komponen yang telah ditentukan. Perancangan dengan komponen terdiri dari empat tahap yaitu antara lain perancangan analisa *use case*, perancangan model, perancangan interaksi dan perancangan antarmuka.

4.3 Perancangan

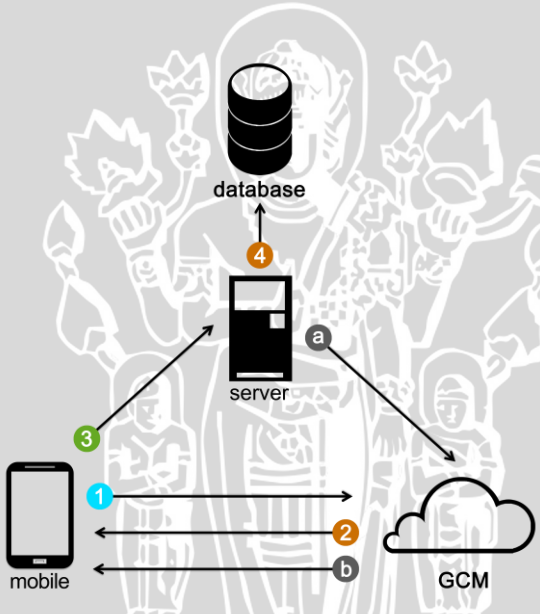
Perancangan perangkat lunak dengan menggunakan UML (Unified Modelling Language) akan dibagi dalam beberapa jenis yaitu : perancangan Arsitektur sistem, perancangan kelas (static Class diagram), perancangan interaksi (Sequence diagram) dan perancangan antarmuka.

4.3.1 Perancangan Arsitektur Sistem

Pada perancangan perangkat lunak *push notification* menyertakan perancangan *factory method pattern dan singleton pattern*. *Singleton pattern* akan digunakan pada kelas alertdialogmanager. *Singleton pattern* berfungsi untuk mencegah instanisasi berulang dalam suatu *resource* atau *class*. Dalam satu aplikasi/method *calling/object calling* hanya satu kali akses koneksi ke *database* saja, jadi tidak setiap pembuatan object baru atau eksekusi fungsi yang sama koneksinya diakses berulang. Seluruh *alert* yang ada akan tersimpan dalam *database* dan apabila pembelajar *login* memanggil instan maka dia akan mendapatkan notifikasi tanpa harus mengakses *database* berkali kali walaupun beda pengguna.

Factory method pattern berupa *class* yang membuat *object* tanpa perlu menggunakan *keyword "new"* untuk membuat *object* tersebut. Ide pembuatan *class factory* ini adalah untuk mempermudah proses pembuatan object dengan parameter tertentu. Salah satu keuntungannya adalah ketika program yang cukup besar, yang sebagian besar kodenya bergantung pada beberapa *class* utama. Dalam hal ini ada *class* Notif yang membutuhkan parameter akses ke sebuah file. Seluruh program menginisiasi *class* tersebut dalam object, kemudian satu-persatu menentukan parameter tersebut. Ketika ingin mengubah agar *class* Notif menggunakan akses database, maka Anda harus mengedit seluruh baris yang berkaitan dengan itu. Tetapi jika menggunakan class Factory untuk *class User* tersebut, maka cukup mengubahnya di *class* Factory, tanpa perlu melihat seluruh kode program

Aplikasi *push notification* dapat berjalan apabila perangkat bergerak selalu dalam keadaan *login* dan terkoneksi dengan jaringan internet. Notifikasi tidak akan diterima apabila pembelajar tidak dalam keadaan *login* dan terkoneksi dengan internet. Apabila suatu saat seorang pembelajar melakukan *logout* ke dalam aplikasi maka tetap tidak akan mendapatkan notifikasi sewaktu dalam keadaan tidak *login.* Hal tersebut dikarenakan sistem berkerja berdasarkan *registrasi id* yang dimiliki setiap perangkat bergerak dari GCM. Setiap perangkat bergerak yang telah keluar dari aplikasi maka notifikasi akan terhapus secara keseluruhan. Sistem tersebut dirancang agar satu perangkat bergerak dapat digunakan oleh banyak pembelajar dan tidak menumpuk data-data pembelajar sebelumnya.



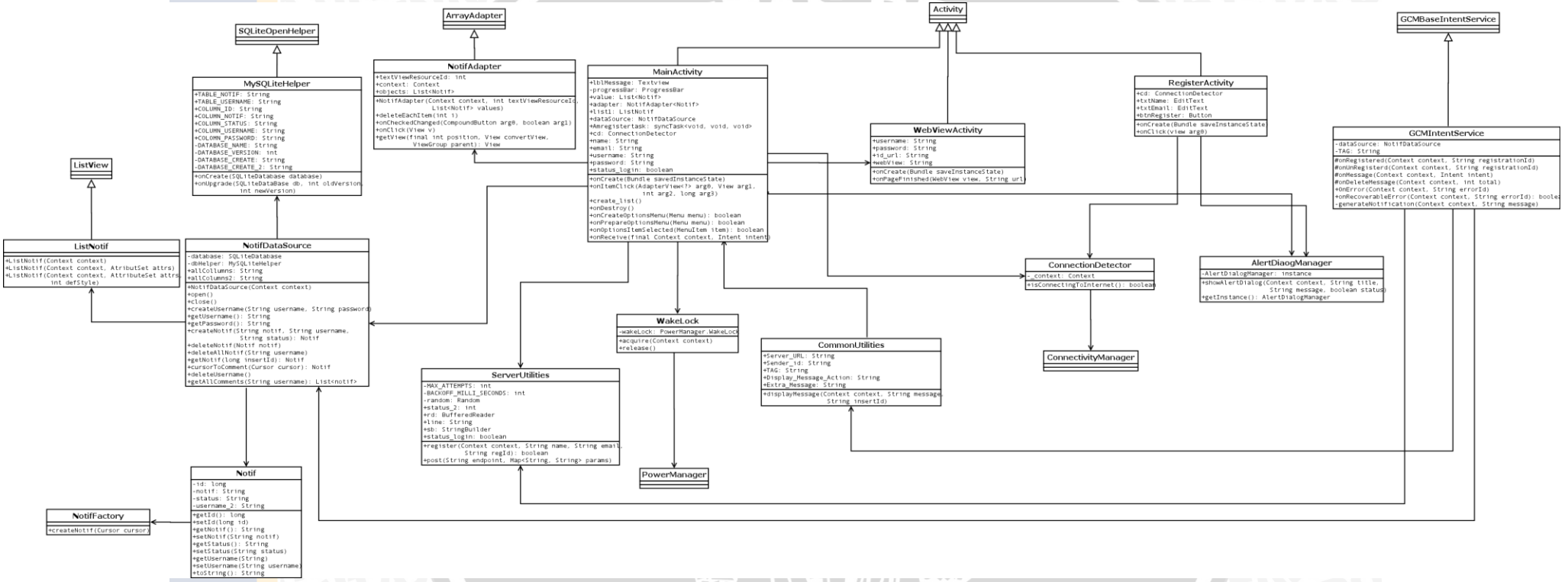
Gambar 4.4 Perancangan Aplikasi *Push Notification*

Pada Gambar 4.4 dapat dilihat bahwa terdapat tiga bagian inti dalam sistem, yaitu perangkat Android, GCM, dan *server*. Google *Cloud Messaging* (GCM) merupakan layanan yang membantu *developer* untuk mengirimkan data melalui *server* kepada perangkat Android yang dimiliki.

1. Device Android mengirimkan *sender ID, Application ID* ke GCM *server* untuk pendaftaran.

2. Apabila telah sukses mendaftar, GCM akan mengirimkan *registration ID* ke *device* Android.
3. *Registration ID* akan dikirimkan oleh *device* Android ke *server*, dalam hal ini adalah Moodle.
4. Server Moodle akan menyimpan *registration ID* ke dalam *database* untuk penggunaan selanjutnya.
5. Setiap kali *push notification* diperlukan, *server* akan mengirim pesan ke GCM, yang dikirimkan berupa *registration ID* (*registration ID* pada *server* yang sebelumnya telah disimpan pada *database*)
6. GCM akan mengirimkan pesan dan pengguna akan mendapatkan layanan selanjutnya.

Perancangan Class Diagram



Gambar 4.5 Diagram Kelas pada Android

Diagram kelas memberikan gambaran (diagram statis) tentang sistem dan relasi-relasi yang terdapat di dalamnya. Kelas-kelas yang telah teridentifikasi dapat memiliki hubungan antar kelas. Kelas-kelas juga dapat memiliki pewarisan dan ketergantungan terhadap komponen-komponen yang telah ditentukan.

Penggunaan *Design Patterns* dalam merancang sebuah sistem juga dianjurkan karena dengan menggunakanya mampu meningkatkan kualitas sebuah perangkat lunak dalam hal *reusability*, *maintainability*, *extensibility* serta bisa mengurangi waktu yang dibutuhkan dalam pengembangan perangkat lunak. Pada perancangan kelas ini menggunakan *Singleton pattern* dan *Factory Method pattern*.

4.3.2.1 Kelas pada Android

a. Kelas RegisterActivity

   Kelas yang berfungsi untuk menampilkan halaman login dan akan menuju ke class MainActivity.

b. Kelas ConnectionDetector

   Kelas ConnectionDetector berfungsi untuk mengetahui apakah aplikasi terhubung dengan koneksi internet atau tidak.

c. Kelas CommonUtilities

   Kelas CommonUtilities berfungsi untuk menampung pengaturan seperti Sever Url dan Sender ID dan juga class ini menjadi penghubung dalam menampilkan pesan notif antara Class GCMIntentService ke kelas MainActivity.

d. Kelas WakeLock

   Kelas Wakelocker berfungsi agar tetap melakukan pengecekan jika sewaktu-waktu ada notifikasi masuk. Berfungsi untuk menghidupkan handphone ketika handphone tersebut dalam keadaan *sleep*.

e. Kelas AlertDialogManager

   Kelas AlertDialogManager berfungsi untuk menampilkan peringatan berupa pop up window di aplikasi. Berisi method yang berfungsi sebagai *singelton pattern.*

f. Kelas GCMIntentService

Kelas GCMInternetService berisi logika penyusunan data notifikasi, pengecekan registrasi dengan google push notification, pendaftaran google push notification, dan pengecekan bila ada error di notifikasi, atau service dalam penerimaan notifikasi.

g. Kelas ServerUtilities

Kelas ServerUtilities berfungsi sebagai penghubung ke server. Dalam kelas ServerUtilities terdapat method yang berfungsi mengirimkan username, email, dan registrationId ke server.

h. Kelas NotifDataSource

Kelas ini berfungsi sebagai controller dari database, penengah yang akan mengkonversi input sebagai perintah untuk model atau view.

i. Kelas Notif

Kelas ini berfungsi sebagai model dari database. Berisi data aplikasi dan fungsi-fungsi yang berhubungan dengan database. Kelas Notif merupakan *product* dari *factory method pattern.*

j. Kelas ListNotif

Class yang berfungsi sebagai view dari notification. Menampilkan data yang diambil dari model dan diolah lewat controller.

k. Kelas MySQLiteHelper

Kelas ini berfungsi untuk membuat database baru. Menyimpan notifikasi ke *database* mengunakan SQLite. Pada kelas MySQLiteHelper berisi deklarasi seperti nama tabel, nama *database*, nama kolom, dan sebagainya.

l. Kelas WebViewActivity

Kelas ini berfungsi sebagai penghung antara notifikasi dengan e-learning moodle

m. Kelas MainActivity

Kelas MainActivity merupakan kelas utama dari aplikasi *push notification.* Pada kelas MainActivity terdapat parameter intent *username* dan *email* yang akan didaftarkan ke dalam google push notification. Dalam MainActivity juga terdapat pengaturan menu option. Kelas MainActivity berisi implementasi dari singeltone pattern yang berfungsi agar hanya satu object alert saja yang terbentuk dan tersimpan dalam *database* agar tidak berulang kali terbentuk.

n. Kelas NotifAdapter

Kelas yang berfungsi sebagai jembatan pengaturan tampilan Android dengan SQLite.

o. Kelas NotifFactory

Kelas NotifFactory merupakan kelas yang menjadi *congcreate creator* pada factory *method pattern.* Pada aplikasi push notification mengadaptasi factory method pattern, karena hanya ada satu *object* yang akan dibuat yaitu notif. Pengadaptasian pattern berfungsi apabila ingin menambahkan *object* lain pada aplikasi, sehingga hanya butuh menambahkan kelas yang bersifat *abstak* sebagai kelas *creator* dan menambahkan object yang akan dibuat pada kelas *congcreate creator*.

4.3.2.2 Kelas pada Moodle

a. Kelas android_config

Kelas android_config berfungsi sebagai configurasi Moodle dengan Android. Dalam kelas tersebut berisi pendefinisian google api key, dan koneksi *database*.

b. Kelas android_functions

Kelas android_functions berisi fungsi fungsi yang dibutuhkan oleh Android seperti *sent* notif serta koneksi ke dalam *database* untuk mengambil google *key*.

c. Kelas android_gcm

Kelas android_gcm berfungsi sebagai kelas yang menghubungkan dengan layanan google cloud messager. Dalam kelas tersebut disertakan juga pengaturan waktu.

d. Kelas *login*

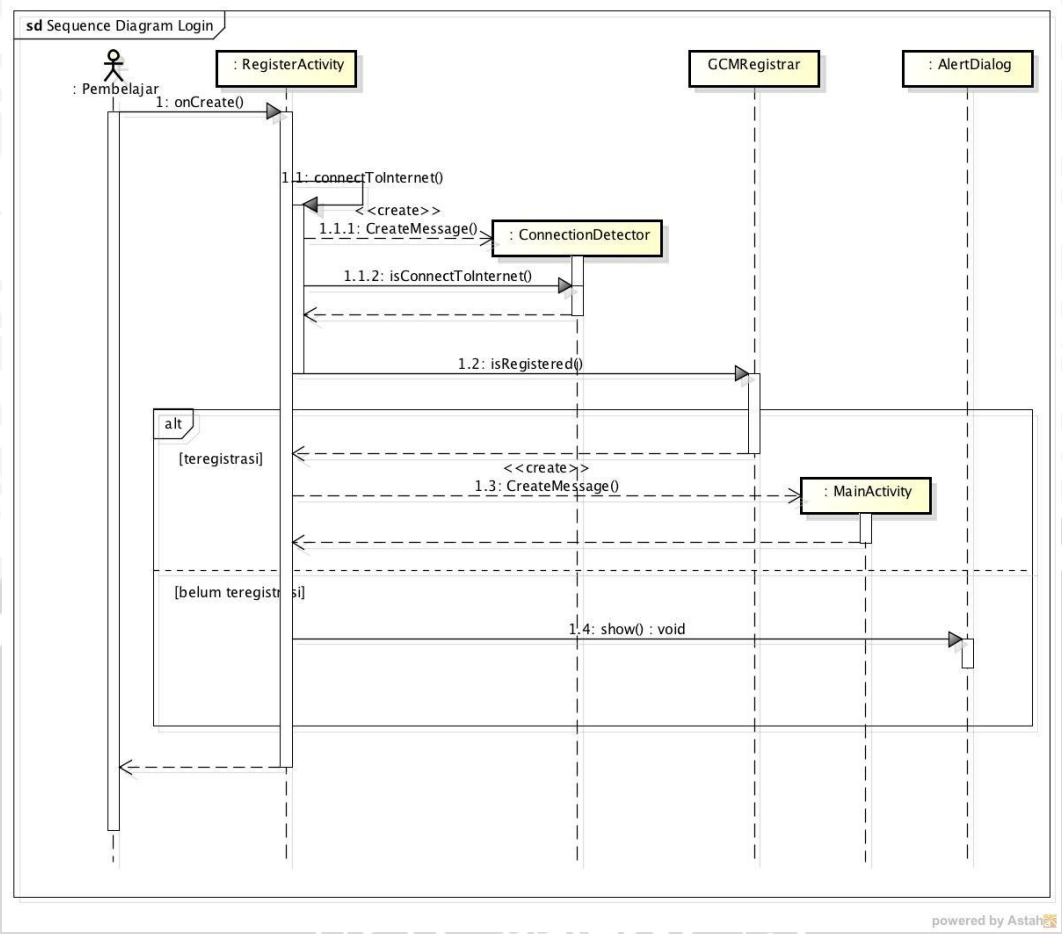
Kelas *login* berisi pengecekan kondisi login *username* dan *password* untuk *device* sesuai dengan Moodle.

e. Kelas *logout*

Kelas *logout* berfungsi sebagai koneksi ke dalam *database* sebagai pengakhiran pembelajar.

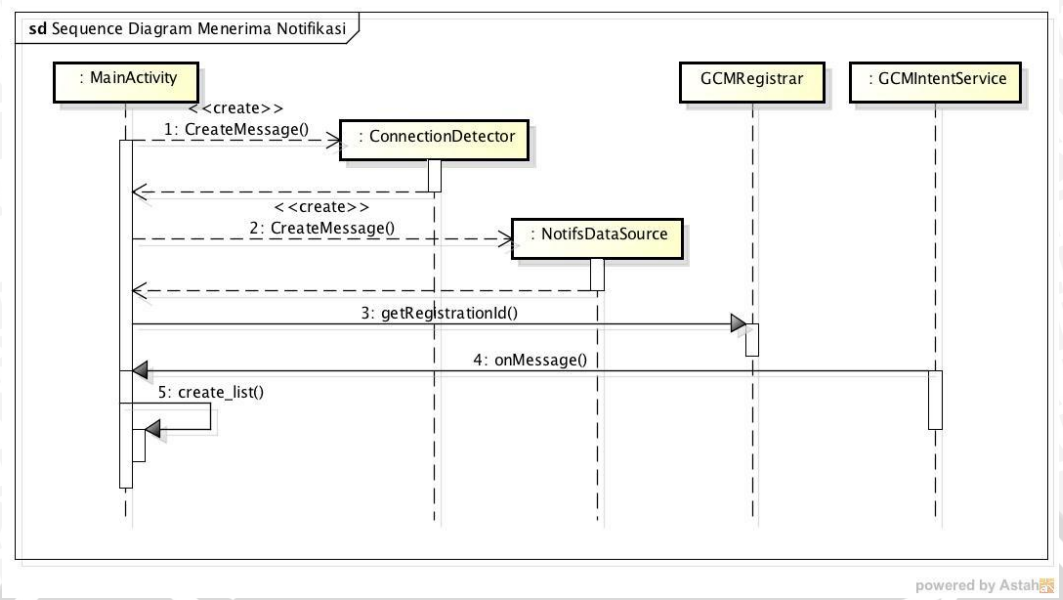
4.3.2 Perancangan Sequence Diagram

Sequence diagram digunakan untuk menggambarkan interaksi antar objek yang disusun dalam urutan waktu. *Sequence diagram* berupa skenario atau rangkaian langkah-langkah yang dilakukan dari sebuah kejadian untuk menghasilkan keluaran tertentu. Skenario pada *sequence diagram* ini menjelaskan interaksi antar kelas dalam sistem yang mana telah diuraikan berdasar *Use Case.*



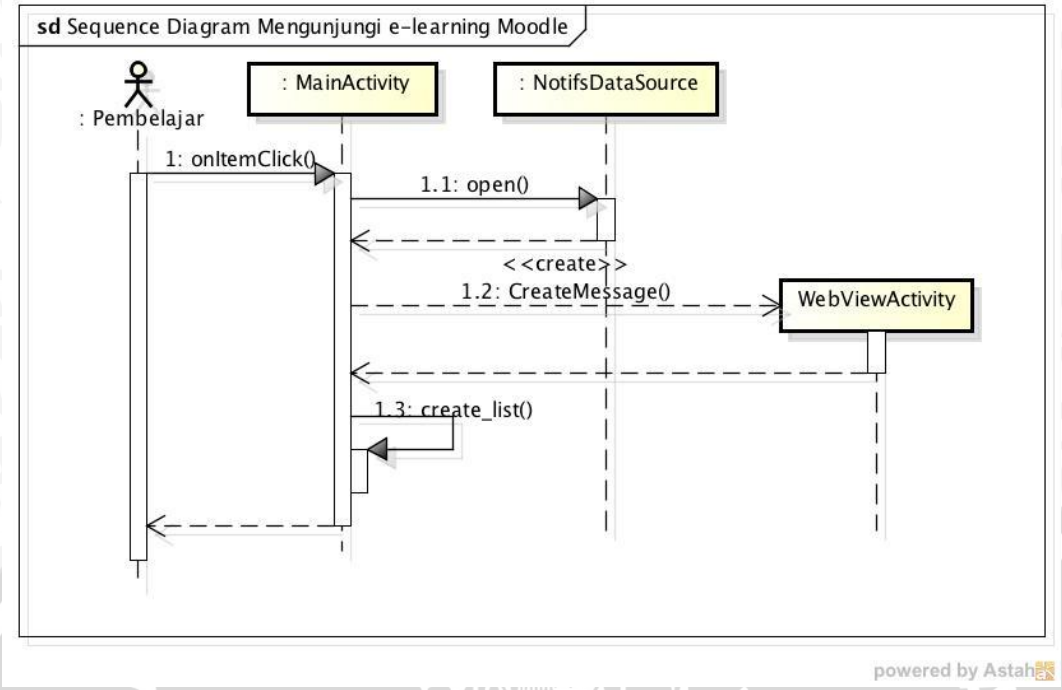
Gambar 4.6 Diagram Interaksi Login

Pada diagram gambar 4.6 menjelaskan kelas yang berinteraksi pada aktifitas*login*. Proses awal yang akan ketika pembelajar pertama kali membuka aplikasi adalah kelas RegisterActivity. Kelas RegisterActivity memfasilitasi proses login sekaligus pendaftaran ke dalam GCM. Ketika telah berhasil login maka akan masuk ke dalam kelas MainActivity.



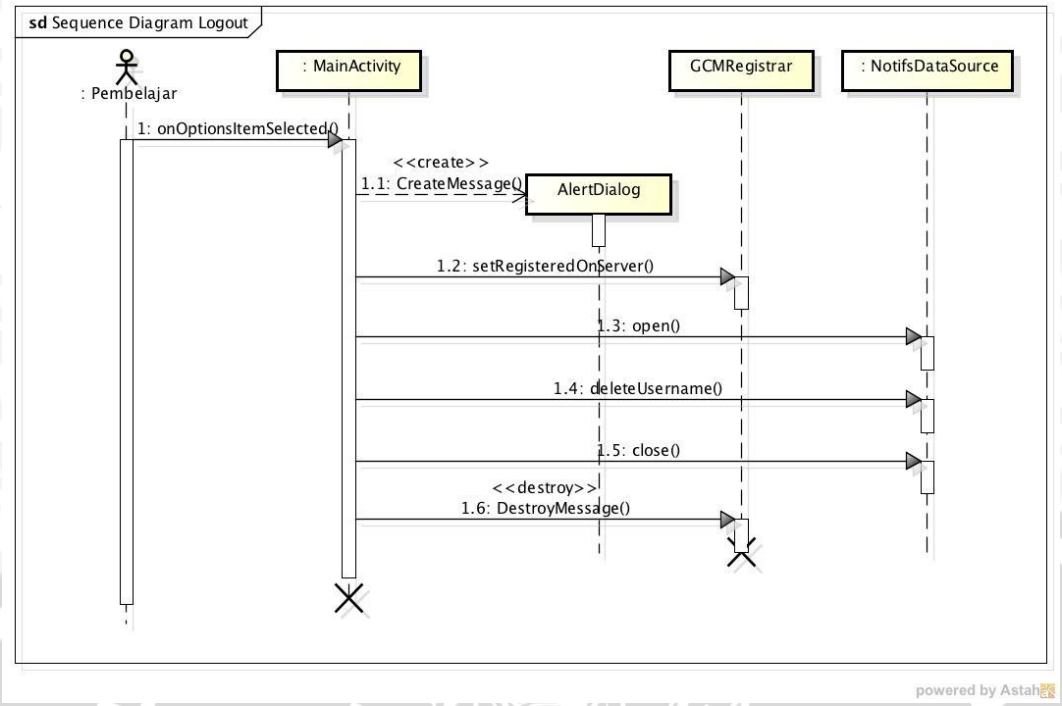
Gambar 4.7 Diagram Interaksi Menerima Notifikasi

Pada diagram gambar 4.7 menjelaskan kelas yang berinteraksi pada aktifitas menerima notifikasi. Untuk dapat menerima notifikasi harus tetap terkoneksi dengan internet. Apabila telah dilakukan pengecekan koneksi selajutnya adalah pengambilan data pembelajar yang telah login dan teregistrasi dari kelas NotifDataSource.Ketika pembelajar telah login dan terdaftar maka akan mendapatkan registrasiID. Akan dilakukan pengecekan registrationID untuk mendapatkan notifikasi. Apabila telah ada dan sesuai maka secra langsung method onMessage() dipanggil dr service dan akan ditampilkan menggunakan method create_list().



Gambar 4.8 Diagram Interaksi Mengunjungi E-learning Moodle

Pada diagram gambar 4.8 menjelaskan kelas yang berinteraksi pada aktifitas mengunjungi e-learning Moodle. Ketika pembelajar menekan tepat pada notifikasi yang diikan untuk dikunjungi maka akan menjalakan method onItemClick(). Setelah itu proses open() database SQLite untuk mengambil alamat URL pada Moodle. Ketika telah didapatkan data maka akan langsung menuju webVieActivity lalu akan menampilkannya dengan memanggil method create_list(). Pembelajar akan melihat tampilan Moodle sesuai notikasi yang di dapat.



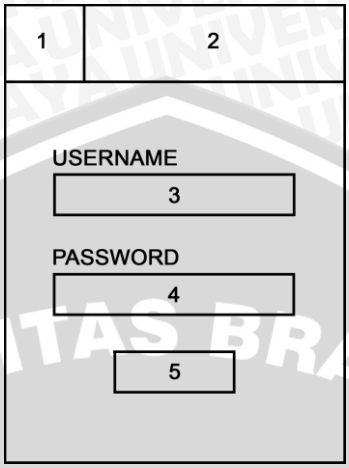
Gambar 4.9 Diagram Interaksi logout

Pada diagram gambar 4.9 menjelaskan kelas yang berinteraksi pada aktifitas *logout*. Pembelajar akan menjalakan method onOptionItemSelected() yaitu method yang mengarah pada pemilihan menu option. Setelah pemilihan menu logout maka akan keluar alert. Ketika dilanjutkan memilih "YA" maka sistem akan menjalankan method setRegisteredOnServer untuk melakukan pengecekan registrationID. Proses selajutnya adalah mebuka database dan melakukan penghapusan sesuai dengan *username*. Ketika logou sistem akan di destroy dan selesai.

4.3.3 Perancangan Antarmuka

Perancangan antarmuka mempunyai tujuan untuk membuat sebuah media komunikasi yang efektif anta manusia dengan sebuah sistem.Tujuan dari perancangan antarmuka adalah untuk membuat sebuah tampilan dari aplikasi yang sederhana agar memudahkan pengguna dalam mengoperasikan aplikasi. Perancangan antarmuka akan menjadi acuan dalam membuat antar-muka untuk sistem.

1. Antarmuka Login



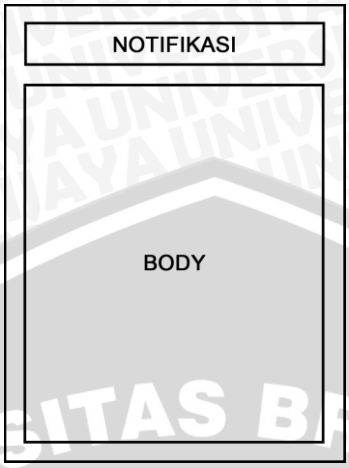
Gambar 4.10 Antarmuka *Login* Aplikasi *Push Notification*

Antarmuka *login* merupakan antarmuka yang pertama diakses oleh user saat pertama kali masuk aplikasi. Gambar 4.10 menunjukan perancangan antarmuka *login*. Pada antarmuka *login* terdapat 2 kolom EditText yang berfungsi sebagai tempat user memasukan data *username* dan *password.* Pada antarmuka ini juga terdapat tombol login yang berisi logika jika user ingin masuk ke dalam aplikasi.

Tabel 4.11 Keterangan Antarmuka Login

No	Nama	Keterangan
1	Logo	Merupakan icon dari aplikasi yang sedang dijalankan
2	*Header* Judul	merupakan judul aplikasi yang sedang dijalankan
3	Kolom Pengisian *Username*	Digunakan untuk menerima masukan *username* pengguna.
4	Kolom Pengisian *Password*	Digunakan untuk menerima masukan *password* pengguna.
5	Tombol Navigasi	Digunakan untuk memproses validitas pengguna

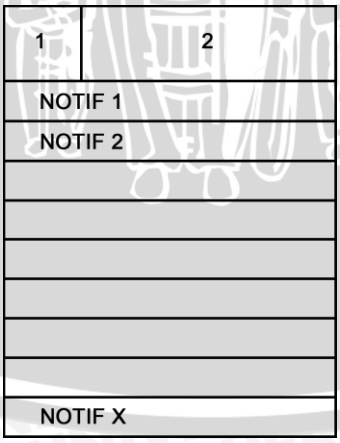
2. Antarmuka Menerima Notifikasi



Gambar 4.11 Antarmuka Menerima Notifikasi

Gambar 4.11 menunjukan perancangan antarmuka Notifikasi. Antarmuka Notifikasi berisi alert notifikasi terbaru. Notifikasi ditampilkan pada header android sehingga apabila user tidak membuka aplikasi, maka notifikasi dapat langsung diketahui pada header android. Hal tersebut dapat berjalan apabila selalu dalam keadaan login dan terhubung dengan internet.

3. Antarmuka Melihat List Notifikasi



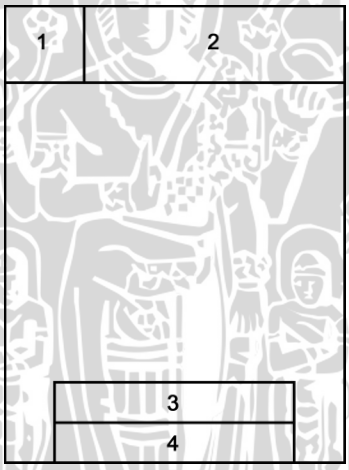
Gambar 4.12 Antarmuka Melihat Notifikasi

Apabila aplikasi dibuka maka akan muncul daftar *list* seluruh notifikasi yang masuk. Jumlah tampilan list notifikasi bergantung dengan layar perangkat bergerak yang digunakan.

Tabel 4.12 Keterangan Antarmuka Melihat Notifikasi

No	Nama	Keterangan
1	Logo	Merupakan icon dari aplikasi yang sedang dijalankan
2	*Header* Judul	merupakan judul aplikasi yang sedang dijalankan
3	Notif x	*List* dari detail notif yang muncul pada *header* android.

4. Antarmuka Option



Gambar 4.13 Antarmuka Menu Option

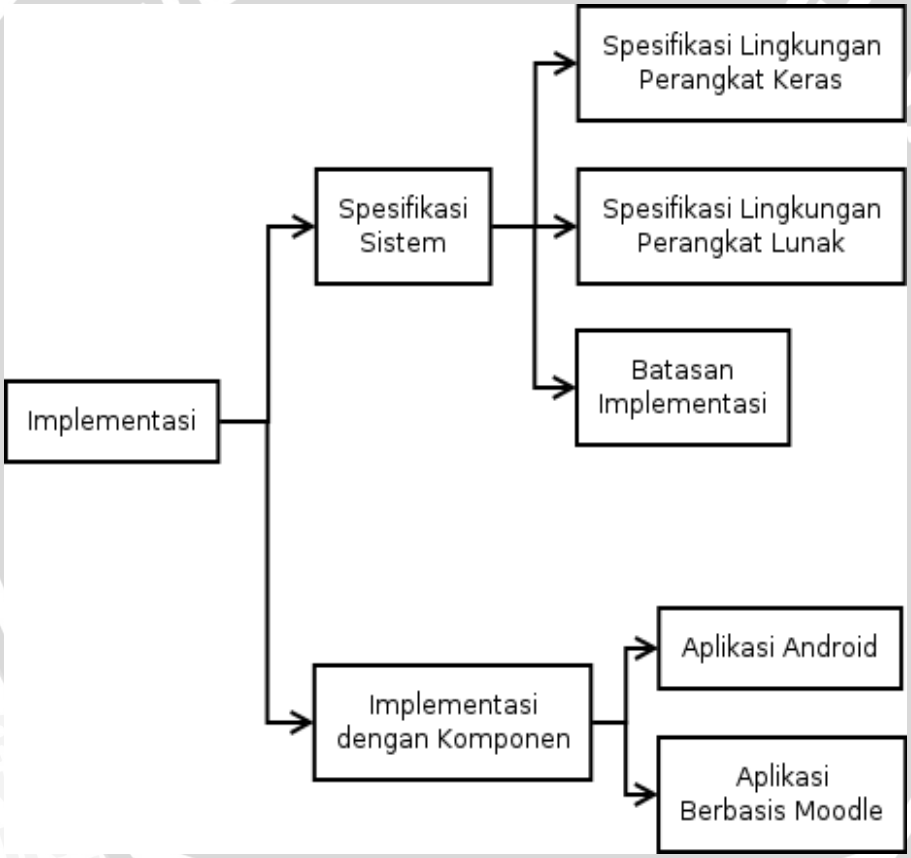
Antarmuka option merupakan tampilan navigasi untuk melakukan tindakan yang akan diberikan kepada notifikasi oleh pengguna yaitu hapus atau keluar dari aplikasi.

Tabel 4.12 Keterangan Antarmuka Menu Option

No	Nama	Keterangan
1	Logo	Merupakan icon dari aplikasi yang sedang dijalankan
2	*Header* Judul	merupakan judul aplikasi yang sedang dijalankan
3	Tombol Navigasi	Tombol ini digunakan setelah memilih notif yang akan dihapus untuk melakukan proses hapus notif.
4	Tombol Navigasi	Digunakan untuk memproses keluar dari aplikasi dan menuju ke antarmuka *login*.

BAB V
IMPLEMENTASI

Bab ini membahas mengenai tahapan implementasi perangkat lunak *push notification* berdasarkan hasil yang telah didapatkan dari analisis kebutuhan dan proses perancangan perangkat lunak yang dibuat. Pembahasan terdiri atas penjelasan tentang spesifikasi sistem, batasan – batasan dalam implementasi, implementasi tiap *class* pada *file* program dan implementasi antarmuka.



Gambar 5.1 Tahap – tahap Implementasi

5.1 Spesifikasi Lingkungan Sistem

Hasil analisis persyaratan perangkat lunak yang telah dilakukan pada bagian bab 4 dijadikan sebagai panduan dalam proses pengimplementasian perangkat lunak *push notification* berbasis Moodle pada perangkat Android. Spesifikasi lingkungan sistem yang digunakan dalam proses implementasi baik itu perangkat keras (*hardware*) maupun perangkat lunak (*software*) akan dijelaskan pada bagian tabel 5.1 dan tabel 5.2.

Tabel 5.1 Spesifikasi Lingkungan Perangkat Keras

MacBook	
Processor	2,13 GHz, Intel Core 2 Duo
Memory (RAM)	2 GB 800 MHz DDR2 SDRAM
Harddisk	159,18 GB
Graphic Card	NVIDIA GeForce 9400M 256 MB

Tabel 5.2 Spesifikasi Lingkungan Perangkat Lunak

MacBook	
Operating System	OS X 10.8.5 32-bit
Programming Language	Java dan PHP Moodle Versi 2.6
Java Development Kit (JDK)	JDK Version 7 Update 10
Java Runtime Environment (JRE)	JRE Version 7 Update 10
Integrated Development Environment	Android *Developer ToolsBundle* (*Eclipse* + ADT *Plugin*, Android SDK *Tools*, Android *Platform Tools*)

5.2 Batasan Implementasi

Beberapa batasan dalam mengimplementasikan sistem perangkat lunak untuk aplikasi *push notification* pada *smartphone* Android adalah sebagai berikut:

a. Pembuatan aplikasi Push Notification berbasis perangkat bergerak ini dikerjakan dengan bahasa Pemrograman Java di lingkungan sistem operasi Android.
b. Fitur yang dibahas pada implementasi perangkat lunak ini antara lain :
    1. Fitur login
    2. Fitur menerima notifikasi
    3. Fitur melihat list notifikasi
    4. Fitur mengunjungi e-learning moodle
    5. Fitur menghapus notifikasi
    6. Fitur logout
c. Implementasi hanya dilakukan berdasarkan *usecase* aplikasi *push notification*
d. Tingkat kecepatan penerimaan data dari *server* tergantung kecepatan akses internet telepon cerdas pengguna

5.3 Implementasi dengan Komponen

Aplikasi *push notification*berbasis Moodle memiliki beberapa fitur. Proses implementasi aplikasi juga menyertakan komponen-komponen yang dianggap dapat membantu pengembangan sehingga dapat memenuhi kebutuhan sistem dan juga meringkas waktu yang diperlukan. Komponen adalah elemen perangkat lunak yang mempunyai standar sesuai dengan model komponen, bisa secara independen digunakan, dan mempunyai *interfaces* (antarmuka) yang menyediakan akses fungsi tanpa harus mengekspos internal detail komponen tersebut [SOM-11].

Berikut ini merupakan tabel yang menjelaskan tugas dari masing-masing komponen yang digunakan dalam membangun aplikasi :

Tabel 5.3 Komponen Perangkat Lunak

No	Nama Komponen	Fungsi
1	GCM	Menyediakan layanan notifikasi.
2	SQLite	Menjadipengolahan basis data pada program Java Android karena sistem SQLite sebagai management basis data.

5.3.1 Implementasi Program Android

Implementasi halaman login berdasarkan perancangan kelas diagram pada bab perancangan. Proses login dilakukan ketika pembelajar pertama kali masuk ke dalam aplikasi push notification.

```
1  publicvoid onCreate(Bundle savedInstanceState) {
2  super.onCreate(savedInstanceState);
3     setContentView(R.layout.activity_register);
4     Intent intent_main = getIntent();
5     Bundle extras = intent_main.getExtras();
6  if(extras!=null){
7       String status_login = intent_main
8       .getStringExtra("status_login");
9  if(status_login.equals("0")){
10      alert.showAlertDialog(RegisterActivity.this,
11          "Error",
12          "Username atau Password Anda Salah", false);
13    }
14 if(status_login.equals("1")){
15      alert.showAlertDialog(RegisterActivity.this,
16          "Error",
17          "Tidak dapat menyambung ke Server", false);
18    }}
19  cd = new ConnectionDetector(getApplicationContext());
20 if(!cd.isConnectingToInternet()) {
21     alert.showAlertDialog(RegisterActivity.this,
22        "Internet Connection Error",
23        "Please connect to working Internet connection",
24 false);
25 return;}
26 if(SERVER_URL == null || SENDER_ID == null ||
27 SERVER_URL.length() == 0 || SENDER_ID.length() == 0) {
28     alert.showAlertDialog(RegisterActivity.this,
29        "Configuration Error!",
30        "Please set your Server URL and GCM Sender ID",
31 false);
32 return;}
33 if (GCMRegistrar.isRegisteredOnServer
34        (getApplicationContext())) {
35     Intent i = new  Intent(getApplicationContext(),
```

```
36          MainActivity.class);
37      i.putExtra("name", "");
38      i.putExtra("email", "");
39      startActivity(i);
40      finish();}
41   txtName = (EditText) findViewById(R.id.txtName);
42   txtEmail = (EditText) findViewById(R.id.txtEmail);
43   btnRegister = (Button) findViewById(R.id.btnRegister);
44   btnRegister.setOnClickListener(new View.OnClickListener(){
45
46  publicvoid onClick(View arg0) {
47    String name = txtName.getText().toString();
48    String email = txtEmail.getText().toString();
49  if (name.trim().length()>0&&email.trim().length() > 0)
50       Intent i = new Intent(getApplicationContext(),
51                MainActivity.class);
52       i.putExtra("name", name);
53       i.putExtra("email", email);
54       startActivity(i);
55       finish();
56    } else {
57       alert.showAlertDialog(RegisterActivity.this,
58           "Registration Error!", "Please enter your details",
59       false);
     }}});
    }}
```

Gambar 5.2 Implementasi Halaman Login

Gambar 5.2 merupakan cuplikan *code* dari *class* RegisterActivityyang berfungsi sebagai syarat atau kondisi sebelum dapat masuk ke dalam aplikasi. Adapun penjelasan dari gambar 5.2 sebagai berikut:

1. Baris 6-42 merupakan kondisiyang digunakan untuk *check* validasi login, konfigurasi ke moodle dan GCM serta koneksi internet dalam device.
2. Baris 43-44 digunakan untuk mengambil nama dan email pengguna yang nantinya dilanjutkan ke MainActivity.
3. Baris 48-62 merupakan method yang berjalan ketika button registerasi di klik. Dalam method tersebut akan mengambil username serta password yang digunakan dan akan dicocokkan dengan data yang telah ada pada moodle.

Implementasi GCM

```
1  publicvoid onCreate(Bundle savedInstanceState) {
2   super.onCreate(savedInstanceState);
3   setContentView(R.layout.activity_main);
4   progressBar = (ProgressBar) findViewById(R.id.progressBar1);
5   cd = new ConnectionDetector(getApplicationContext());
```

```
 context = getApplicationContext();
if (!cd.isConnectingToInternet()) {
  alert.showAlertDialog(MainActivity.this,
   "Internet Connection Error",
   "Please connect to working Internet connection", false);
return;
 }
 Intent i = getIntent();
 name = i.getStringExtra("name");
 email = i.getStringExtra("email");

 GCMRegistrar.checkDevice(this);
 GCMRegistrar.checkManifest(this);
 datasource = new NotifsDataSource(this);
 lblMessage = (TextView) findViewById(R.id.lblMessage);
 list1 = (ListNotif) findViewById(R.id.list1);
 list1.setOnItemClickListener(new android.widget.
       AdapterView.OnItemClickListener() {
 registerReceiver(mHandleMessageReceiver, new IntentFilter(
      DISPLAY_MESSAGE_ACTION));
final String regId = GCMRegistrar.getRegistrationId(this);
if (regId.equals("")) {
  GCMRegistrar.register(this, SENDER_ID);
 } else {
if (GCMRegistrar.isRegisteredOnServer(this)) {
    Toast.makeText(getApplicationContext(),
      "Already registered with GCM", Toast.LENGTH_LONG)
      .show();
    progressBar.setVisibility(View.INVISIBLE);
    lblMessage.setVisibility(View.INVISIBLE);
    datasource.open();
    username = datasource.getUsername();
password = datasource.getPassword();
    datasource.close();
    status_login=true;
    create_list();
 } else {
final Context context = this;
  mRegisterTask = new AsyncTask<Void, Void, Void>() {

protected Void doInBackground(Void... params) {
   ServerUtilities.register(context, name, email, regId);
returnnull;
 }

protectedvoid onPostExecute(Void result) {
   mRegisterTask = null;
  }
};
 mRegisterTask.execute(null, null, null);
  }
 }
}
privatefinal BroadcastReceiver mHandleMessageReceiver = new
     BroadcastReceiver() {

publicvoid onReceive(final Context context, Intent intent) {
```

```
63   String
64  newMessage=intent.getExtras().getString(EXTRA_MESSAGE);
65   WakeLocker.acquire(getApplicationContext());
66   lblMessage.setText(newMessage + "\n");
67   Toast.makeText(getApplicationContext(),"New Message: " +
68        newMessage, Toast.LENGTH_LONG).show();
69   WakeLocker.release();
70   }
    };
```

Gambar 5.3 Implementasi GCM

Gambar 5.3 merupakan cuplikan *code* dari *class* MainActivityyang berfungsi untuk menerima push notifikasi yang dikirimkan oleh server secara real time. Adapun penjelasan dari gambar 5.3 sebagai berikut:

1. Baris 3-12 merupakan sebuah *method* yang digunakan untuk *check* koneksi internet dalam device.
2. Baris 13-15 digunakan untuk mengambil data nama dan email.
3. Baris 17-18 digunakan untuk *check manifest* dan *device* apakah sudah support dengan google push notification atau tidak.
4. Baris 28 digunakan untuk mendaftarkan device ke google push notification.
5. Baris 62-70 merupakan *method* yang digunakan untuk menerima notifikasi dari server. Baris 63 berfungsi untuk menerima pesan dan menyimpanya di *variable* message. Baris 64 berfungsi untuk mengaktifkan daya jika *device* dalam posisi *standby*. Baris 65-66 berfungsi menampilkan notifikasi ke interface.

Implementasi Web View

```
1   publicvoid onCreate(Bundle savedInstanceState) {
2   super.onCreate(savedInstanceState);
3   final WebView webview = new WebView(this);
4    setContentView(webview);
5    Intent i = getIntent();
6    username = i.getStringExtra("username");
7    password = i.getStringExtra("password");
8    id_url = i.getStringExtra("id_url");
9    WebSettings webSettings = webview.getSettings();
10   webSettings.setJavaScriptEnabled(true);
11   webview.setWebViewClient(new WebViewClient());
12   Log.i("URL",id_url);
13   String                          postData                    =
14  "username="+username+"&password="+password;
```

```
15   webview.postUrl("http://learning123.esy.es/login/index.php",
16           EncodingUtils.getBytes(postData, "utf-8"));
17   webview.setWebViewClient(new WebViewClient() {
18
19     publicvoid onPageFinished(WebView view, String url) {
20  if(url.equals("http://learning123.esy.es/")){
21       view.loadUrl(id_url);
22     }
23       Log.i("URLL", webview.getUrl());
24    }
25   });
     }
```

Gambar 5.4 Implementasi Web View

Gambar 5.4 merupakan cuplikan *code* dari *class* WebViewActivityyang berfungsi untuk menampilkan url dari notification yang telah didapat. Adapun penjelasan dari gambar 5.4 sebagai berikut:

1. Baris 6-8 digunakan untuk mengambil data username dan password dari pengguna serta url sesuai dengan notifikasi yang didapat.
2. Baris 13-14 digunakan untuk mengirimkan data kepada moodle.
3. Baris 18-22 merupakan sebuah method apabila ketika telah sesuai semua dengan data pada moodle maka akan menampilkan url sesuai dengan notifikasi.
4. Web view berjalan ketika method onItemClick pada class mainActivity berjalan.
5. Method onItemClick dipanggil dimana apabila seorang pengguna melakukan tindakan klik pada notifikasi yang dikehendaki dibuka maka method berjalan dan akan memulai aktifitas mengunjungi url (WebView) sesuai dengan notifikasi.

Implementasi Pengaturan Notifikasi

```
protectedvoid onMessage(Context context, Intent intent) {
 Log.i(TAG, "Received message");
 String username = intent.getExtras().getString("username");
 String message0 = intent.getExtras().getString("course");
 String message = intent.getExtras().getString("name");
 String message2 = intent.getExtras().getString("descript");
 String message3 = intent.getExtras().getString("fromdate");
 String message4 = intent.getExtras().getString("todate");
 String id_cource = intent.getExtras().getString("id");

 message = message0+"\n\n"+message+"\n\n"+message2+"\n\nDari
        Tanggal : "+message3+"\nSampai Tanggal : "+message4;
if(message!=null)
 {
   datasource = new NotifsDataSource(context);
   datasource.open();
   String current_username = datasource.getUsername();
if(username.equals(current_username)){
      Notif notif=datasource
            .createNotif(message,username,id_cource);
      displayMessage(context,message,String
            .valueOf(notif.getId()));
      generateNotification(context, message);
   }
   datasource.close();
 }
else
displayMessage(context, message,"");
}

privatestaticvoid    generateNotification(Context    context,
String  message) {
int icon = R.drawable.ic_launcher;
long when = System.currentTimeMillis();
 NotificationManager notificationManager =
   (NotificationManager)context
   .getSystemService(Context.NOTIFICATION_SERVICE);
 Notification notification = new Notification(icon, message,
   when);
 String title = context.getString(R.string.app_name);
 Intent notificationIntent = new Intent
   (context,MainActivity.class);
 notificationIntent.setFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP
   | Intent.FLAG_ACTIVITY_SINGLE_TOP);
 PendingIntent intent = PendingIntent.getActivity(context, 0,
   notificationIntent, 0);
 notification.setLatestEventInfo(context, title, message,
   intent);
 notification.flags |= Notification.FLAG_AUTO_CANCEL;

 notification.defaults |= Notification.DEFAULT_SOUND;
 notification.defaults |= Notification.DEFAULT_VIBRATE;
 notificationManager.notify(0, notification);
}
```

Gambar 5.5 Implementasi Pengaturan Notifikasi

Gambar 5.5 merupakan cuplikan *code* yang berfungsi untuk menampilkan url pengaturan isi dari notification. Adapun penjelasan dari gambar 5.5 sebagai berikut:

1. Baris 1-12 merupakan method onMessage yang dipanggil ketika terdapat notification dari Moodle. Dalam method tersebut mengambil sebagian informasi dari moodle sesuai dengan username pengguna yaitu mata kuliah yang diambil, jenis tugas, tanggal diberikannya tugas dan tanggal berakhirnya tugas.
2. Baris 13-29 digunakan untuk suatu kondisi dimana pada baris 15 melakuakna inisialisasi data base, dan pada baris 16 database akan di buka. Apabila seluruh data sesuai dengan database maka akan ditampilkan message sesuai dengan informasi yang telah ditentukan sebelumnya. Pada baris 23 akan memanggil generateNotification
3. Baris 31 merupakan method generateNotification yang digunakan untuk membuat notifikasi pada device.
4. Baris 52-54 merupakan pengaturan sound serta vibrate dari notification.

Implementasi Configurasi Android dengan Moodle

```
1   staticfinal String SERVER_URL =
2        "http://learning123.esy.es/android/login.php";
3
4   staticfinal String SENDER_ID = "889654159728";
5
6   staticfinal String TAG = "AndroidHive GCM";
7
8   staticfinal String DISPLAY_MESSAGE_ACTION =
9        "com.androidhive.pushnotifications.DISPLAY_MESSAGE";
10
11  staticfinal String EXTRA_MESSAGE = "message";
12
13  staticvoid     displayMessage(Context     context,     String
14  message,String insertId) {
15                Intent           intent           =           new
16  Intent(DISPLAY_MESSAGE_ACTION);
17                intent.putExtra(EXTRA_MESSAGE, message);
18                intent.putExtra("insertId", insertId);
19                context.sendBroadcast(intent);
    }}
```

Gambar 5.6 Implementasi Configurasi Android dengan Moodle

Gambar 5.6 merupakan cuplikan *code* dari *class* CommonUtilitiesyang berfungsi untuk menampung pengaturan seperti server url dan sender id serta sebagai class penghubung dalam menampilkan pesan notif antara class GCMIntentService dengan Main Activity . Adapun penjelasan dari gambar 5.6 sebagai berikut:

1. Baris 1 digunakan sebagai pengaturan url dari moodle.
2. Baris 4 digunakan sebagai pengaturan sender id yang telah didapatkan dari pendaftaran pada google cloud messaging.
3. Baris 13-19 merupakan method displayMessage, digunakan untuk menampung nilai atau data notication dari Moodle dan akan menujukannya ke class MainActivity.

5.3.2 Implemetasi Program Moodle

Program *push notification* dapat berjalan setelah terconfigurasi dengan baik dengan Moodle.

```
1  <?php
2  define("DB_HOST", "mysql.main-hosting.com");
3  define("DB_USER", "u297685914_yjesa");
4  define("DB_PASSWORD", "asumyJuRaZ");
5  define("DB_DATABASE", "u297685914_upynu");
6  define("GOOGLE_API_KEY",
7  "AIzaSyACn4Vx1_hqWR2rg9u_9G9OuqbyEA8HugY");
8  class DB_Connect {
9        function __construct() {}
10       function __destruct() {}
11       public function connect() {
12 $con = mysql_connect(DB_HOST, DB_USER, DB_PASSWORD);
13             mysql_select_db(DB_DATABASE);
14             return $con;
15       }
16       public function close($con) {
17             mysql_close($con);} }
18 ?>
```

Gambar 5.7 Implementasi Configurasi Moodle

Gambar 5.7 merupakan cuplikan *code* dari *class* android_config.phpuntuk mengkonfigurasi android dengan moodle. Kode php ini berfungsi untuk mengoneksikan database dan mengirim notifikasi dan akan diterima oleh android. Adapun penjelasan dari gambar 5.7 sebagai berikut:

1. Baris 2 – 7 merupakan variabel konfigurasi string koneksi database dan variabel konstan lainnya.
2. Baris 11 merupakan nama kelas yang digunakan untuk koneksi ke *database*.
3. Baris 12 merupakan sintak *query* yang digunakan untuk koneksi ke *database mysql*.
4. Baris 13 merupakan fungsi untuk *select database*.
5. Baris 16-17 merupakan fungsi untuk menutup koneksi *database*.

Implementasi GCM pada Moodle

```
1  <?php
2  class GCM {
3  function __construct() {}
4  public function send_notification($registatoin_ids,$message) {
5  require_once ("android_config.php");
6
7  $url = 'https://android.googleapis.com/gcm/send';
8  $fields = array(
9          'registration_ids' => $registatoin_ids,
10         'data' => $message,
11         'waktu'=> date(),
12 );
13
14     echo "<pre>";
15     print_r($fields);
16     die();
17
18 $headers = array(
19         'Authorization: key=' . GOOGLE_API_KEY,
20 'Content-Type: application/json'
21 );
22
23 $ch = curl_init();
24 curl_setopt($ch, CURLOPT_URL, $url);
25     curl_setopt($ch, CURLOPT_POST, true);
26     curl_setopt($ch, CURLOPT_HTTPHEADER, $headers);
27     curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);
28 curl_setopt($ch, CURLOPT_SSL_VERIFYPEER, false);
29     curl_setopt($ch, CURLOPT_POSTFIELDS, json_encode($fields));
30     $result = curl_exec($ch);
31
32 if ($result === FALSE) {
```

```
33  die('Curl failed: ' . curl_error($ch));
34      }
35  curl_close($ch);
36  }
37  } ?>
```

Gambar 5.8 Implementasi GCM pada Moodle

Gambar 5.8 merupakan cuplikan *code* dari *class* android_gcm.phpyang berfungsi untuk mengkonfigurasi android dengan moodle. Kode php ini berfungsi untuk mengoneksikan database dan mengirim notifikasi dan akan diterima oleh android. Adapun penjelasan dari gambar 5.8 sebagai berikut:

1. Baris 4 merupakan method yang berfungsi untuk mengirimkan notifikasi.
2. Baris 23 berfungsi sebagai awal penghubung ke url yang dimana urlnya telah dituliskan sebelumnya
3. Baris 24 – 29 merupakan fungsi untuk memanggil opsi curl atau paramater parameter yang bernilai konstan
4. Baris 30 berfungsi untuk menampilkan atau mencetak halaman url
5. Baris 32 – 34 merupakan kondisi dari hasil url apabila salah akan error
6. Baris 35 berfungsi untuk menutup resource url

Implementasi Fungsi Android pada Moodle

```
1   public function setNotif($fromform, $url){
2     $this->sendNotif($fromform, $url);
3   }
4
5   private function sendNotif($fromform, $url) {
6     require_once ("android_gcm.php");
7     $gcm = new GCM();
8     $con = $this->connect();
9
10    //GET COURSE NAME FROM COURSE ID
11    $cID = $fromform->course;
12    $cName = mysql_query("SELECT c.fullname, cc.name
13                          FROM mdl_course AS c
14                          JOIN mdl_course_categories AS cc
15                          ON c.category = cc.id
16                          WHERE c.id = $cID", $con);
17    $cName = mysql_fetch_row($cName);
18    $cName = $cName[1]." ".$cName[0];
19
20    $actName = $fromform->name;
21    $actDesc = "Not Available";
22    $actFromdate = "Not Available";
23    $actTodate = "Not Available";
24
```

```
25    if(isset($fromform->introeditor)){
26      foreach($fromform->introeditor as $row){
27          $actDesc = str_replace(array("<p>", "</p>"), '', $row);
28          break;
29      }
30    }
31    if(isset($fromform->timeavailable)){
32      $actFromdate = date('D, d M Y h:i:s',
33      $fromform->timeavailable);
34    }
35    if(isset($fromform->timedue)){
36      $actTodate = date('D, d M Y h:i:s', $fromform->timedue);
37    }
38
39    //MESSAGE CONTENT
40    $message = array("id" => $url,
41                     "course" => $cName,
42                     "name" => $actName,
43                     "descript" => $actDesc,
44                     "fromdate" => $actFromdate,
45                     "todate" => $actTodate);
46
47    //GET USERNAME FROM COURSE ID
48    $cUser = mysql_query("SELECT u.username
49                          FROM mdl_role_assignments ra, mdl_user u,
50                               mdl_course c, mdl_context cxt
51                          WHERE ra.userid = u.id AND
52                               ra.contextid = cxt.id AND
53                               cxt.contextlevel = 50 AND
54                               cxt.instanceid = c.id AND
55                               c.id = $cID AND roleid = 5", $con);
56
57    while ($row = mysql_fetch_row($cUser)) {
58         //GET REGID FROM USERNAME
59          $regid = mysql_query("SELECT regid
60                               FROM mdl_android
61                               WHERE username = '".$row[0]."'", $con);
62          if(mysql_num_rows($regid) > 0){
63             regid = mysql_fetch_row($regid);
64             $message["username"] = $row[0];
65             $gcm->send_notification(array($regid[0]), $message);
66          }
67    }
68    $this->db->close($con);
69  }
70  private $db;
71  private function connect(){
72     require_once ("android_config.php");
73     $this->db = new DB_Connect();
74     return $this->db->connect();
75  }
```

Gambar 5.9 Implementasi Fungsi Android pada Moodle

Gambar 5.9 merupakan cuplikan *code* dari *class* android_functions.phpmerupakan method yang berfungsi android. Adapun penjelasan dari gambar 5.9 sebagai berikut:

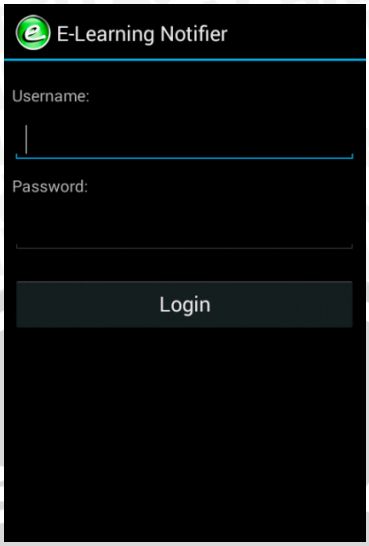
1. Baris 5 - 8 merupakan method yang berfungsi untuk mengirimkan notifikasi.
2. Baris 10 – 23 merupakan query untuk mendapatkan data dari database serta proses pengambilan variable apa saja yang akan ditampilkan
3. Baris 25 – 37 mejadi suatu kondisi pengecekan waktu dan course
4. Baris 39 – 45 mengambil isi untuk dijadikan isi dari pesan notifikasi
5. Baris 47 – 55 merupakan query untuk menentukan siapa saja user dalam suatu mata kuliah
6. Baris 58 – 69 mendapatkan registrasi id dari username yang telah login
7. Baris 70 – 74 merupakan fungsi untuk mengoneksikan database

5.4 Implementasi Antarmuka

Antarmuka aplikasi *Push Notification* pada perangkat bergerak sistem operasi Android digunakan oleh pengguna untuk berinteraksi dengan Moodle. Implementasi antarmuka berdasarkan perancangan antarmuka pada bab perancangan sub bab 4.4.1.4. Antarmuka perangkat lunak ini dibagi menjadi 5, yaitu antarmuka login, notifikasi, hapus notifikasi, mengunjungi Moodle, logout.

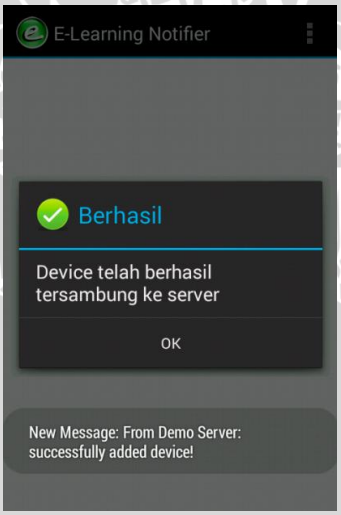
a. Implementasi Antarmuka Halaman Login

Implementasi antarmuka login berdasarkan perancangan antarmuka pada bab perancangan merupakan halaman yang pertama diakses oleh user ketika masuk ke dalam aplikasi *push notification.* Halaman ini memiliki kolom username dan password untuk diisikan oleh user. Pada halaman ini juga terlihat tombol login yang harus ditekan oleh user jika ingin masuk ke dalam aplikasi *push notification.*Gambar 5.10 menunjukkan antarmuka login.



Gambar 5.10 Antarmuka *Login*

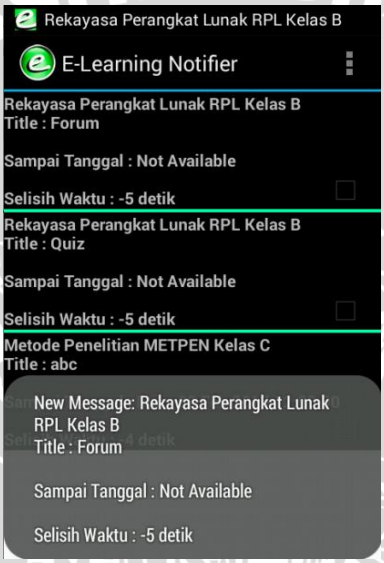
Login berhasil ketika pembelajar telah mendapatkan pesan bahwa device yang digunakan telah berhasil terhubung ke server. Untuk masuk ke dalam halaman home aplikasi *push notification*pembelajar menekan tombol ok pada pesan tersebut.



Gambar 5.11 Antarmuka Device terhubung dengan GCM

b. Implementasi Antarmuka Notification

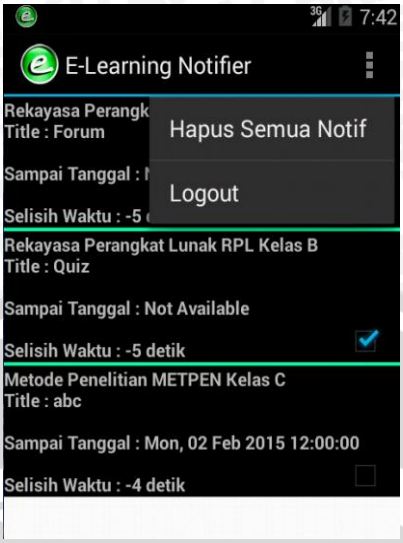
Implementasi antarmuka notifikasi berdasarkan perancangan antarmuka pada bab perancangan merupakan halaman yang digunakan pengguna untuk melihat pemberitahuan terbaru segala aktivitas di dalam elearning yang berkaitan dengan dirinya secara realtime. Antarmuka ini memuat kelas, mata kuliah, jenis aktivitas, dan tanggal berakhir. Data notifikasi yang didapat dari database ditampilkan ke interface menggunakan tipe list view. Gambar 5.12 menunjukan antarmuka notifikasi



Gambar 5.12 Antarmuka Notifikasi

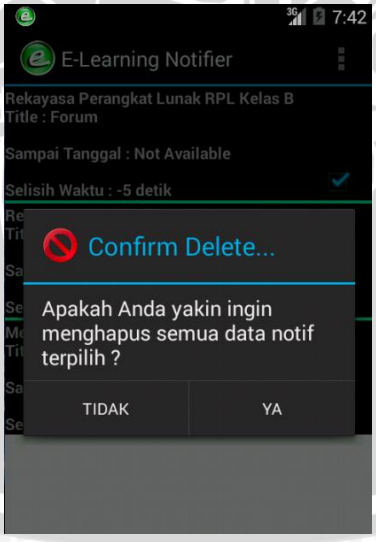
c. Implementasi Antarmuka Hapus Notification

Implementasi antarmuka hapus notifikasi berdasarkan perancangan antarmuka pada bab perancanganmerupakan halaman yang digunakan untuk menghapus notifikasi yang diinginkan. Antarmuka ini membantu pengguna memonitor notifikasi mana yang sudah lampau atau yang terbaru. Gambar 5.13 menunjukkan antarmuka hapus notifikasi.



Gambar 5.13 Antarmuka Hapus Notifikasi

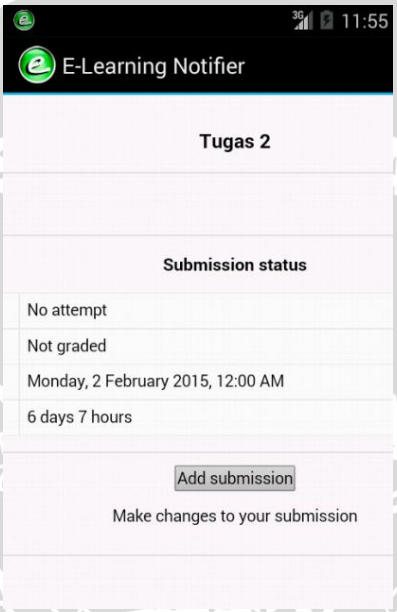
Untuk dapat menghapus notifikasi pembelajar diminta untuk memilih dan memberi check list pada notifikasi yang akan dihapus. Fasilitas hapus terdapat pada menu option. Sebelum terhapus sistem akan mengkonfirmasi, apabila yakin untuk dihapus maka pembelajar memilih tombol ya dan notifikasi yang diinginkan akan terhapus.



Gambar 5.14 Antarmuka Konfirmasi Hapus Notifikasi

d. Implementasi Antarmuka Mengunjungi Moodle

Implementasi antarmuka mengunjungi Moodle berdasarkan perancangan antarmuka pada bab perancangan dapat diakses dengan cara mengklik notifikasi yang diinginkan. Antarmuka website digunakan untuk melihat secara detail isi dari notifikasi yang didapat dan untuk melakukan kegiatan selanjutnya. Gambar 5.15 menunjukkan antarmuka Moodle.



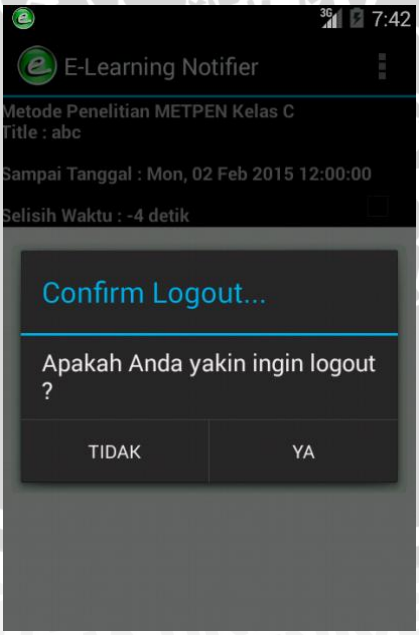
Gambar 5.15 Antarmuka Mengunjungi Moodle

e. Implementasi Antarmuka Logout

Implementasi antarmuka logout berdasarkan perancangan antarmuka pada bab perancangan merupakan halaman yang digunakan oleh user ketika akan mengakhiri aplikasi push notification.Gambar 5.16 menunjukkan antarmuka logout.

Gambar 5.16 Antarmuka *Logout*

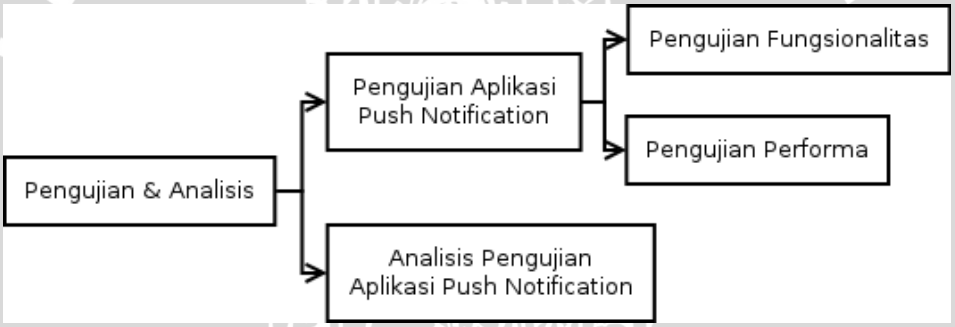
Sebelum benar-benar keluar dari aplikasi, pembelajar dimintai konfirmasi apakah yakin ingin keluar dari aplikasi atau tidak. Jika ingin mengakhiri aplikasi maka pembelajar memilih tombol ya dan sistem akan berakhir.



Gambar 5.17 Antarmuka Konfirmasi Logout

BAB VI
PENGUJIAN DAN ANALISIS

Bab ini membahas mengenai tahapan pengujian dan analisis Aplikasi *Push Notification* berbasis Moodle pada perangkat Android yang telah dibangun. Proses pengujian dilakukan melalui dua tahapan pengujian fungsionalitas dan pengujian performa. Pada pengujian fungsionalitas dan performa akan digunakan teknik pengujian *Black-BoxTesting.*Pengujian dan analisis sistem perlu dilakukan untuk mengetahui apakah sistem yang telah dibuat telah memenuhi kebutuhan dan sesuai dengan tujuan.



Gambar 6.1 Tahap – tahap Pengujian

6.1 Pengujian Fungsionalitas Aplikasi *Push Notification*

Pengujian aplikasi *push notification* pada perangkat Android dilakukan pada setiap SRS yang telah didefinisikan pada tabel 4.2 daftar kebutuhan fungsional aplikasi *push notification*. Setiap SRS akan diuji fumgsionalitasnya dan ditetapkan statusnya apakah valid atau tidak. Status valid atau tidaknya suatu SRS adalah berdasarkan kriteria seperti berikut ini:

- Fungsionalitas dapat berjalan dengan benar seperti yang diinginkan pada analisis kebutuhan.

6.1.1 Kasus Uji Pengujian

Kasus uji memuat nama kasus uji, obyek uji, tujuan pengujian, prosedur uji dan hasil yang diharapkan. Terdapat enam kasus uji dalam pengujian fungsionalitas diantaranya adalah kasus uji *login*, kasus uji menerima notifikasi, kasus uji melihat *list* notifikasi, kasus uji mengunjungi e-learning moodle, kasus uji menghapus notifikasi, dan kasus uji logout.

a. Kasus Uji *Login*

Kasus pengujian fungsi *login*dilakukan sebagai awal untuk mengetahui apakah proses *login*telah berjalan dengan baik sehingga data *username* dan *password* pengguna bila sesuai dengan Moodle akan mengijinkan pengguna masuk ke halaman utama aplikasi dan jika terjadi ketidak sesuaian antara *username* dan*password* pengguna maka pengguna akan gagal masuk ke halaman utama aplikasi.

Tabel 6.1 Kasus Uji untuk Pengujian Fungsional *Login*

Nama Kasus Uji	Kasus Uji *Login*
Objek Uji	SRS_PN_001
Tujuan Pengujian	Pengujian dilakukan untuk memastikan bahwa aplikasi dapat memenuhi kebutuhan fungsional bagi pembelajar untuk dapat masuk ke dalam aplikasi *push notification*
Prosedur Uji	1. Pembelajar membuka aplikasi 2. Tampilan *login* keluar 3. Pembelajar memasukkan *username* *4.* Pembelajar memasukkan *pasword* 5. Pembelajar menekan tombol *login* 6. Telah terdaftar ke dalam Moodle
Hasil yang diharapkan	Pembelajar dapat masuk ke dalam halaman aplikasi *push notification* dan terdaftar ke GCM

b. Kasus Uji Menerima Notifikasi

Kasus pengujian fungsi menerima notifikasi disini dilakukan sebagai langkah untuk mengetahui apakah fungsi menerima notifikasi telah berjalan dengan baik sehingga pembelajar dapat menerima notifikasi secara *real time* dan sesuai dengan e-learning Moodle.

Tabel 6.2 Kasus Uji untuk Pengujian Fungsional Menerima Notifikasi

Nama Kasus Uji	Kasus Uji menerima notifikasi
Objek Uji	SRS_PN_002
Tujuan Pengujian	Pengujian dilakukan untuk memastikan bahwa aplikasi dapat memenuhi kebutuhan fungsional bagi Pembelajar untuk dapat menerima notifikasi secara *real time* dan sesuai dengan e-learningMoodle
Prosedur Uji	1. Aplikasi telah berjalan dan pengguna telah terdaftar ke GCM serta Moodle 2. Pembelajar harus selalu dalam kondisi *login* 3. Pembelajar harus selalu terhubung dengan internet
Hasil yang diharapkan	Aplikasi dapat menampilkan notifikasi secara *real time* sesuai dengan Moodle

c. Kasus Uji Melihat *List* Notifikasi

Kasus pengujian fungsi melihat *list* notifikasi disini dilakukan sebagai langkah untuk mengetahui apakah fungsi tersebut telah berjalan dengan baik sehingga seluruh notifikasi yang diterima dapat dilihat.

Tabel 6.3 Kasus Uji untuk Pengujian Fungsional Melihat *List* Notifikasi

Nama Kasus Uji	Kasus Uji melihat *list* notifikasi
Objek Uji	SRS_PN_003
Tujuan Pengujian	Pengujian dilakukan untuk memastikan bahwa aplikasi dapat memenuhi kebutuhan fungsional bagi pengguna untuk dapat melihat *list* notification

Prosedur Uji	1. Pembelajar telah mendapatkan notifikasi dari Moodle 2. Pembelajar selalu dalam keadaan *login* 3. Pembelajar terhubung dengan koneksi internet 4. Permbelajar masuk ke dalam aplikasi *push notification*
Hasil yang diharapkan	Pembelajar dapat melihat *list* notifikasi dari notifikasi yang telah datang secara keseluruhan selama pembelajar dalam keadaan *login*

d. Kasus Uji Menghapus Notifikasi

Kasus pengujian fungsi menghapus notifikasi disini dilakukan sebagai langkah untuk mengetahui apakah fungsi menghapus notifikasi disini telah berjalan dengan baik, sehingga jika pembelajar ingin menghapus notifikasi dengan menekan tombol *option* lalu memilih hapus maka notifikasi terhapus.

Tabel 6.4 Kasus Uji untuk Pengujian Fungsional Menghapus Notifikasi

Nama Kasus Uji	Kasus Uji menghapus notifikasi
Objek Uji	SRS_PN_004
Tujuan Pengujian	Pengujian dilakukan untuk memastikan bahwa aplikasi dapat memenuhi kebutuhan fungsional bagi pembelajar untuk dapat menghapus notifikasi pada *mobile*
Prosedur Uji	1. Aplikasi telah berjalan dan pengguna telah menerima notifikasi 2. Pembelajar memilih notifikasi yang akan dihapus 3. Pembelajar menekan tombol *option* dan memilih hapus 4. Memastikan apakah akan yakain untuk dihapus 5. Pengguna menekan tombol “Ya”
Hasil yang diharapkan	Aplikasi berhasil menghapus notifikasi yang diinginkan pembelajar untuk dihapus

e. Kasus Uji Mengunjungi e-learning Moodle

Kasus pengujian fungsi mengunjungi e-learning Moodle disini dilakukan sebagai langkah untuk mengetahui apakah notifikasi yang didapat sesuai dan terhubung dengan e-learning Moodle.

Tabel 6.5 Kasus Uji untuk Pengujian Fungsional Mengunjungi e-learning Moodle

Nama Kasus Uji	Kasus Uji mengunjungi e-learning Moodle
Objek Uji	SRS_PN_005
Tujuan Pengujian	Pengujian dilakukan untuk memastikan bahwa aplikasi dapat memenuhi kebutuhan fungsional bagi pembelajar untuk dapat langsung mengunjungi *website* sesuai dengan notifikasi yang muncul
Prosedur Uji	1. Aplikasi telah berjalan dan pembelajar telah menerima notifikasi 2. Pembelajar memilih notifikasi 3. Pembelajar menekan tepat di notifikasi yang diinginkan untuk dibuka
Hasil yang diharapkan	Aplikasi menampilkan halaman *web* sesuai dengan notifikasi yang didapat.

f. Kasus Uji *Log Out*

Kasus pengujian fungsi *logout* disini dilakukan sebagai langkah untuk mengetahui apakah fungsi *logout* sudah berjalan dengan baik sehingga pengguna dapat keluar dari aplikasi dengan menekan *button logout* jika pengguna merasa sudah cukup dalam menggunakan aplikasi ini.

Tabel 6.6 Kasus Uji untuk Pengujian Fungsional *Logout*

Nama Kasus Uji	Kasus Uji logout
Objek Uji	SRS_PN_006
Tujuan Pengujian	Pengujian dilakukan untuk memastikan bahwa aplikasi dapat memenuhi kebutuhan fungsional bagi pembelajar

	untuk mengakhiri aplikasi
Prosedur Uji	1. Pembelajar telah *login* ke aplikasi 2. Pembelajar menekan tombol option dan memilih tombol logout
Hasil yang diharapkan	Aplikasi berakhir dan muncul halaman *login* untuk masuk lagi ke dalam aplikasi.

6.1.2 Hasi Pengujian Fungsionalitas

Hasil pengujian pada setiap kasus uji akan dijabarkan pada tabel 5.8 untuk mengetahui apakah setiap fungsi pada aplikasi telah bernilai valid untuk memenuhi kebutuhan fungsional aplikasi *push notifikasi*pada *smartphone*Android.

Tabel 6.7 Hasil Pengujian Fungsionalitas

No	Kasus Uji	Kondisi yang diharapkan	Kondisi yang didapat	Hasil
1.	*Login*	• Ditampilkan halaman *login* yang berisi kolom input *username* dan *password.* • Jika *username* dan *password* benar, maka *user* akan masuk ke halaman utama aplikasi • Hanya pengguna yang telah terdaftar dalam Moodle saja yang dapat masuk ke dalam aplikasi • *User* akan terdaftar ke GCM	• Ditampilkan halaman *login* yang berisi kolom input *username* dan *password.* • Jika *username* dan *password* benar, maka *user* akan masuk ke halaman utama aplikasi • Hanya pengguna yang telah terdaftar dalam Moodle saja yang dapat masuk ke dalam aplikasi • *User* akan terdaftar ke GCM	Valid

No	Kasus Uji	Kondisi yang diharapkan	Kondisi yang didapat	Hasil
2	Menerima notifikasi	• Aplikasi dapat menerima notifikasi secara *real time* sesuai dengan Moodle melalui GCM • Notifikasi yang diterima sesuai dengan kelas dan mata kuliah yang diambil pengguna	• Aplikasi dapat menerima notifikasi secara *real time* sesuai dengan Moodle melalui GCM • Notifikasi yang diterima sesuai dengan kelas dan mata kuliah yang diambil pengguna	Valid
3	Melihat list notifikasi	• Aplikasi dapat menampilkan *list* notifikasi secara keseluruhan selama pembelajar dalam keadaan *login*	• Aplikasi dapat menampilkan *list* notifikasi secara keseluruhan selama pembelajar dalam keadaan *login*	Valid
4	Menghapus notifikasi	• Aplikasi dapat menghapus notifikasi yang diinginkan untuk dihapus	• Aplikasi dapat menghapus notifikasi yang diinginkan untuk dihapus	Valid
5	Mengunjungi e-learning Moodle	• Pengguna dapat terhubung langsung ke halaman e-learning Moodle sesuai dengan notifikasi yang didapat	• Pengguna dapat terhubung langsung ke halaman e-learning Moodle sesuai dengan notifikasi yang didapat	Valid
6	*Logout*	• Pengguna berhasi keluar dari aplikasi *push notification* dan akan kembali ke halaman *login*	• Pengguna berhasi keluar dari aplikasi *push notification* dan akan kembali ke halaman *login*	Valid

6.1.3 Analisis Hasil Pengujian Fungsionalitas

Proses analisis terhadap hasil pengujian fungsionalitas dilakukan dengan melihat konformitas antara hasil kinerja sistem dengan daftar kebutuhan. Hasil pengujian fungsionalitas pada kasus uji a sampai dengan f dengan metode *black-box testing* adalah 100%, sehingga dapat disimpulkan bahwa implementasi dan fungsionalitas aplikasi *push notification*dapat berjalan sesuai dengan spesifikasi kebutuhan yang telah dijabarkan pada tahap analisis kebutuhan.

6.2 Pengujian Performa

Pengujian performa dilakukan untuk memenuhi kebutuhan non fungsional. Pengujian dilakukan dengan cara menguji keluaran yang sesuai dengan perubahan yang terjadi pada e-learning Moodle dan waktu sampainya notifikasi pada perangkat *mobile*. Pengujian performa pada skripsi ditujukan untuk menghitung waktu eksekusi aplikasi dan kesesuaian isi dari notifikasi dengan e-learning Moodle.

6.2.1 Hasil Pengujian Performa

Pengujian performa dilakukan kepada seluruh mahasiswa yang telah terdaftar pada e-learning Moodle. Dalam hal ini pengujian dilakukan tidak dalam satu lokasi. Mahasiswa yang terdaftar dalam e-learning Moodle tersebar pada beberapa tempat. Mahasiswa berada di kost, rumah, kampus dan lain sebagainya. Dengan syarat selalu terhubung dengan internet dan selalu dalam keadaan login maka mahasiswa yang terdaftar dalam e-learning Moodle akan mendapatkan notifikasi sesuai dengan mata kuliah yang sedang ditempuh.

Notifikasi yang diterima pembelajar menampilkan informasi sesuai dengan analisis kebutuhan yang telah ditetapkan. Setiap notifikasi menampilan informasi mata kuliah yang ditempuh, tipe perubahan berupa materi, quiz, tugas, dan lain sebagainya serta batas waktu yang telah ditetapkan sesuai dengan Moodle.

Tabel 6.8 Hasil Pengujian Performa

NO	Nama	NIM	Mata Kuliah											
			METPEN			RPL			PPP		APS			BASDAT	
			A	B	C	A	B	C	A	B	A	B	C	A	B
1	Aggy K.	0910683008			25		26		27			27		25	
2	Anom Harya	0910680009			25		26		27		27				25
3	Antharangga B	0910681003	25			27			27		27				25
4	Ardhian Dharma	0910680006	976				904			987			990	999	
5	Arga Wirya	0910681008	821					806		832			855	844	
6	Arief Budi	0910680008			30			30		30		30		30	
7	Bayu Adhi	0910683013	25				26			27	27				25
8	Bintang Karunia	0910680020	32				31		32			32			31
9	Bagus Ibnu	0910683039		30				27		32		30		27	
10	Cantika Nur P	0910680011			27			27	27		27			27	
11	Delis Sukmawati	0910683030	43					43		44			42		44
12	Diah Ayu	0910680073		26		27			27				27		25
13	Erwin Aprilianto	0910683022		26		24			25		27			27	
14	Febri Angelina	0910681101		26		25				27		27		25	
15	Gerry Wisudawan	0910683042		43			43			44			42		44
16	Hafidz Rahmat	0910680046	32					31		32			30	27	
17	Hanas Subakti	0910680033	110			78				99			117		89
18	Ikrar Muhammad	0910683034		25			27		26		27				25
19	Ilham Ubaidillah	0910680017			43		43		42		42			43	
20	Imam Wahyudi	0910681023			1009			987		1013		1036			1023
21	Moh Halimi	0910680041			1648			1674	1666		1626			1664	
22	M. Riski Gus	0910680028	32			31			31		32				33
23	Nizam Ghonim	0910683023			26	25			26			26		26	
24	Rakanta Rifky	0910681079		26		26				26			26		26
25	Ridwan Aditama	0910681004			389			410		385	362			373	

26	Ramadhan Bakti	0910683036			32			32	31			32		31	
27	Rayu Sili	0910683033	25					27	26				26		25
28	Reny Khairina	0910683019	32					31		32			31		33
29	Rizky Kurnia	0910681068		27		25				26		26		26	
30	Silvi Alviolita	0910681083		27		25			26				27		27

Rata – rata waktu sampainya setiap notifikasi $= \frac{jumlah\ dari\ seluruh\ waktu\ notifikasi\ yang\ datang}{jumlah\ notifikasi\ yang\ datang}$

$= \frac{28.326\ detik}{150}$

= 188.84 detik/notifikasi

Tabel 6.9 Keterangan Lokasi dan Provider

No	Nama	NIM	Lokasi	Provider
1	Aggy K.	0910683008	Malang	IM3
2	Anom Harya	0910680009	Malang	Wifi Speedy
3	Antharangga B	0910681003	Malang	Simpati
4	Ardhian Dharma	0910680006	Jakarta	XL
5	Arga Wirya	0910681008	Kepanjen	IM3
6	Arief Budi	0910680008	Malang	Simpati
7	Bayu Adhi	0910683013	Malang	Simpati
8	Bintang Karunia	0910680020	Malang	Simpati
9	Bagus Ibnu	0910683039	Malang	XL
10	Cantika Nur P	0910680011	Malang	XL
11	Delis Sukmawati	0910683030	Madiun	Simpati
12	Diah Ayu	0910680073	Bojonegoro	Simpati
13	Erwin Aprilianto	0910683022	Malang	Simpati
14	Febri Angelina	0910681101	Bojonegoro	Simpati
15	Gerry Wisudawan	0910683042	Sidoarjo	3
16	Hafidz Rahmat	0910680046	Jakarta	Simpati
17	Hanas Subakti	0910680033	Malang, Kampus	Wifi
18	Ikrar Muhammad	0910683034	Bali	XL
19	Ilham Ubaidillah	0910680017	Malang	3
20	Imam Wahyudi	0910681023	Mojokerto	IM3
21	Moh Halimi	0910680041	Bontang	3
22	M. Riski Gus	0910680028	Madiun	Simpati
23	Nizam Ghonim	0910683023	Malang, Kampus	IM3
24	Rakanta Rifky	0910681079	Malang	Simpati

25	Ridwan Aditama	0910681004	Balikpapan	Simpati
26	Ramadhan Bakti	0910683036	Ponorogo	IM3
27	Rayu Sili	0910683033	Malang	XL
28	Reny Khairina	0910683019	Bojonegoro	XL
29	Rizky Kurnia	0910681068	Yogyakarta	Simpati
30	Silvi Alviolita	0910681083	Situbondo	IM3

6.2.2 Analisis Hasil Pengujian Performa

Dengan menggunakan perhitungan selisih antara waktu device dengan waktu server akan didapatkan *respon time* dari notifikasi. Tabel 6.8 merupakan hasil dari pengujian performa yang dilakukan ketika seluruh pembelajar yang terdaftar login dalam waktu bersamaan.

Proses analisis terhadap hasil pengujian performa dilakukan dengan melihat waktu sampainya notifikasi ke perangkat android serta kesesuaian keluaran notifikasi yang diterima dengan e-learning Moodle. Pada waktu yang bersamaan 30 pembelajar yang terdaftar dalam Moodle menerima seluruh notifikasi sesuai dengan mata kuliah yang ditempuh dan notifikasi sesuai dengan perubahan yang terjadi pada e-learning Moodle.

Waktu sampainya notifikasi setiap pembelajar berbeda beda, namun pada setiap pembelajar memiliki pola kecenderungan yang sama pada setiap notifikasi yang didapat antara notifikasi satu dengan lainnya. Adanya perbedaan waktu sampainya notifikasi setiap pembelajar menurut peneliti dimungkinkan oleh lokasi pembelajar saat menerima notifikasi serta *provider* yang digunakan. Setiap lokasi memiliki signal yang berbeda-beda tergantung provider yang digunakan, sehingga penerimaan notifikasi setiap pembelajar berbeda pula.

Sebagai contoh pembelajar atas nama Moh.Halimi, pola waktu yang terbentuk antara 1626-1674 dan pembelajar atas nama Aggy K. memiliki pola waktu antara 25-27. Pada setiap pembelajar telihat bahwa sampainya setiap notifikasi setiap pembelajar memiliki pola *range* tersendiri. Dari data yang di dapat waktu tercepat sampainya notifikasi satu dengan lainnya setiap pembelajar hanya berjarak satu detik, sedangkan waktu terlama sampainya notifikasi satu dengan lainnya setiap pembelajar adalah 50 detik.

Dari hasil pengujian perfoma didapatkan juga waktu rata-rata sampainya setiap notifikasi dari seluruh pembelajar. Hasil rata-rata waktu dari 150 notifikasi yang masuk yaitu 188,84 detik. Nilai rata-rata yang mencapai 188,84 tersebut dikarenakan adanya pola waktu yang terbentuk mecapai nilai ribuan.

BAB VII
PENUTUP

7.1 Kesimpulan

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

1. Perancangan aplikasi *Push Notification* berbasis Moodle pada sistem operasi Androidtelah dibuat sesuai spesifikasi kebutuhan yang telah dianalisa.
2. Model pengembangan perangkat lunak menggunakan model *reuse-oriented software engineering*. Model tersebut memudahkan untuk mengembangkan perangkat lunak dengan menggunakan komponen-komponen yang sudah ada sebelumnya.
3. Aplikasi *Push Notification* berbasis moodle pada sistem operasi Android telah dibuat sesuai perancangan dan diimplementasikan atau diadaptasikan dari komponen-komponen yang telah ditentukan, antara lain: *library* GCM, SQLite.
4. Berdasarkan dari hasil pengujian fungsionalitas, implementasi dan fungsionalitas aplikasi *Push Notification* berbasis Moodle pada sistem operasi Androiddapat berjalan dengan baik. Hal ini dibuktikan dengan hasil pengujian *Black-Box* yang memberikan nilai prosentase sebesar 100%.
5. Berdasarkan dari hasil pengujian performa, aplikasi*push notification* dapat menerima seluruh notifikasi sesuai dengan mata kuliah yang ditempuh dan notifikasi sesuai dengan perubahan yang terjadi pada e-learning Moodle. Notifikasi yang di dapat setiap pembelajar berisi mata kuliah yang ditempuh, tipe perubahan, dan batas waktu. Isi dari notifasi telah sesuai dengan kebutuhan yang telah direncanakan sebelumnya. Pada setiap pembelajar telihat bahwa sampainya setiap notifikasi setiap pembelajar memiliki pola *range* tersendiri.Pengujian performa juga

menghasilkan waktu rata-rata sampainya setiap notifikasi yaitu 188,84 detik. Dari rata-rata yang di dapat terlihat bahwa waktu sampainya notifikasi mencapai 188,84 detik dikarenakan adanya pola waktu yang terbentuk dari setiap pembelajar yang mencapai hingga nilai ribuan.

7.2 Saran

Penelitian ini merupakan sebagian kecil dari pengembangan pembelajaran berbasis Moodle pada system operasi Android. Terdapat banyak sekali hal yang dapat dilakukan untuk mengembangkan penelitian ini sehingga dapat tercipta aplikasi yang dapat memenuhi dan seluruh kebutuhan pembelajar. Saran yang diberikan untuk pengembangan penelitian selanjutnya antara lain sebagai berikut :

1. Memperkaya fitur-fitur yang disediakan, sehingga pembelajar lebih dimudahkan dalam kegiatan perkuliahan. Sebagai contoh, terdapat fitur untuk mengunduh materi dan mengunggah tugas.
2. Melakukan pengujian usability pada penggunaan aplikasi ***push notification*** untuk mengetahui tingkat kenyamanan pembelajar dalam menggunakannya.
3. Perlu adanya pengembangan aplikasi ke dalam *platform* atau lingkungan sistem operasi yang berbeda agar seluruh pembelajar dapat menggunakan.
4. Melakukan penelitian terkait aplikasi, misalnya penelitian untuk mengetahui faktor pasti yang menghambat sampainya notifikasi ke pembelajar.

DAFTAR PUSTAKA

[ADR-08] Adri, M., 2008. *Konsep Dasar eLearning dengan Moodle*. Kuliah Umum IlmuKomputer.com

[ARA-11] Aranda, A.D., 2011. *Moodle for Distance Education*. Volume 8, Issue 2.

[COB-06] Cobcroft, R., Towers, S., Smith, J., & Bruns, A. 2006. "*Literature Review Into Mobile Learning in The University Context*." Queensland: Queensland University of Technology Creative Industries Faculty.

[DEV-14] *Creative Commons Attribution 3.0 License. Google Cloud Messaging for Android*. [online] Tersedia di : <https://developer.android.com/google/gcm/index.html> [Diakses 28 November 2013]

[DHA-03] Dharwiyanti, S., dan Wahono, R. S., 2003. *Pengantar Unified Modeling Language (UML).* Kuliah Umum IlmuKomputer.Com.

[ELL-09] Ellis, R.K., 2009. "*Field Guide to Learning Management Systems*". American Society for Training & Development, ASTD Inc.

[FOR-08] Forment, M.A., Guerrero, J. C., 2008. "*Moodlbile: Extending Moodle To The Mobile On/Offline Scenario*". Inmaculada Arnedillo: Barcelona, Spain.

[GAM-01] Gamma, E., 1995.*Design Pattern Elements of Reusable Object Oriented Software.*Addison-Wesley Pub Co.New York

[GAR-14] Gargeta, M., Nakamura, N., 2014, "*Learning Android*", O'Reilly Media, Inc. Sebastopol.

[GEO-O4] Geordiev, T., Georgieva, E., & Smrikarov, A., 2004. "*M-Learning – a New Stage of E-Learning."* International Conference on Computer Systems and Technologies, CompSysTech2004.

[HOL-05] Holzinger, A., Nischelwitzer, A., & Meisenberger, M., 2005. "*Mobile Phones as a Challenge for m-Learning: Examples for Mobile Interactive Learning Objects (MILOs)."* PerCom 2005 Workshops : IEEE

[COB-06] Cobcroft, R., Towers, S., Smith, J., & Bruns, A., 2006. "*Literature Review Into Mobile Learning in The University Context."* Queensland: Queensland University of Technology Creative Industries Faculty.

[KAT-10] Kats, Y., 2010. "*Learning Management System Technologies and Software Solutions or Online Teaching: Tools and Applications*". Information Science Reference: New York.

[MET-06] Metcalf , D.S., DeMarco, J.M., 2006. "*mLearning: Mobile Learning and Performance in the Palm of Your Hand*". HRD Press, Inc: U.S and Canada.

[MOD-13] Moodle. 2013. *About Moodle*. [online] Tersedia di : <http://docs.moodle.org/25/en/About_Moodle> [Diakses 4 September 2013.]

[PRE-01] Pressman, R. 2001. *Software Engineering: A Practitioner's Approach, 5th Edition*. Mc Graw-Hill.

[PRE-10] Pressman, R. 2010. *Software Engineering: A Practitioner's Approach, 7th Edition*. Mc Graw-Hill.

[RUM-99] Rumbaugh, J. Jacobson, I., & Booch, G., 1999. "*The Unified Modeling Language Reference Manual*". Addison Wesley : Canada.

[SAF-12] Safaat, N., 2012. "*Android:Pemrograman Aplikasi Mobile Smartphone dan Tablet PC Berbasis Android*" Penerbit Informatika. Bandung

[SEA-14] Rouse M., 2014. *Push Notification*. [online] Tersedia di <http://searchconsumerization.techtarget.com/definition/push-notification> [Diakses 4 November 2014]

[SOM-11] Sommerville, I., 2011. *Software Engineering 9th Edition*. Pearson

[STA-13] Statcounter. 2013. *StatCounter Global Stats*. [online] Tersedia di : <http://gs.statcounter.com> [Diakses 3 Oktober 2013].

[TUR-09] Turino, Purwanto, Yuliman, Soeleman, Arief. 2009. "E-Learning Bahasa Inggris Berbasis Web". Jurnal Teknologi Informasi, Volume 5 Nomor 2,

[TUT-14] ISSN 1414-9999. Tutorials Point. *Design Patterns in Java Tutorial*. [online] Tersedia di : <http://www.tutorialspoint.com/design_pattern> [Diakses 30 Oktober 2014]