BAB II
TINJAUAN PUSTAKA

I.1 Aplikasi Mobile

Aplikasi *mobile* dapat diartikan sebagai sebuah produk dari sistem komputasi *mobile*, yaitu sistem komputasi yang dapat dengan mudah dipindahkan secara fisik dan yang komputasi kemampuan dapat digunakan saat mereka sedang dipindahkan. Contohnya seperti *Personal Digital Assistant (PDA)*, *smartphone* dan ponsel. Dengan menggunakan aplikasi *mobile,* dapat dengan mudah melakukan berbagai macam aktivitas mulai dari hiburan, berjualan, belajar, mengerjakan pekerjaan kantor, *browsing* dan lain sebagainya. Pemanfaatan aplikasi *mobile* untuk hiburan paling banyak digemari oleh hampir 70% pengguna *smartphone*, karena dengan memanfaatkan adanya fitur *game*, *music player*, sampai *video player* membuat kita menjadi semakin mudah menikmati hiburan kapan saja dan dimanapun [4].

Adapun manfaat dari penggunaan aplikasi *mobile*, antaranya sebagai berikut:

1. Meningkatkan Proses Bisnis

   Aplikasi *mobile* juga berguna untuk meningkatkan proses bisnis yang sudah berjalan. Contohnya penggunaan *WhatsApp*.

2. Meningkatkan Kualitas Komunikasi

   Aplikasi *mobile* juga berguna untuk saling berbagi informasi yang berhubungan dengan banyak hal maka komunikasi yang sebelumnya sulit dilakukan karena harus bertemu langsung bisa semakin mudah dengan memanfaatkan aplikasi *mobile* tersebut.

3. Meningkatkan Kualitas Hidup

   Dengan menciptakan aplikasi *mobile* maka memudahkan manusia ketika menjalani keseharian hidup. Contohnya pengguna yang memesan makanan secara *online* dengan aplikasi.

4. Terdepan Dalam Persaingan

   Aplikasi *mobile* dapat digunakan untuk menawarkan kepada pelanggan sebagai peluang dalam persaingan.

5. Meningkatkan Keterlibatan Pelanggan

Meningkatkan dan membangun keterlibatan pelanggan dapat dilakukan dengan aplikasi *mobile*. Contohnya para pelanggan bisa terhubung dengan fitur *help desk* atau fitur pesan sehingga dapat berkomunikasi dengan penjual.

6. Membangun dan Mengenalkan Merek

   Aplikasi bisnis khususnya untuk bisnis tidak hanya menjadi sarana menyebarkan informasi, namun juga membangun dan mengenalkan merek kepada pelanggan maupun calon pelanggan [5].

I.2 Sistem Operasi Android

Android merupakan suatu sistem operasi berbasis *Linux* yang digunakan untuk telepon pintar (*smartphone*) ataupun pada komputer tablet. Android menyediakan *platform* terbuka bagi para pengembang dalam menciptakan aplikasi mereka sendiri untuk digunakan oleh bermacam peranti bergerak. Android dikembangkan oleh Android, *Inc* tahun 2005 lalu dibeli oleh *Google* pada tahun 2007. Kemudian untuk mengembangkan dan mempercanggih sistem operasi Android maka dibentuklah *Open Handset Alliance (OHA)*, konsorsium dari beberapa perusahaan peranti keras, perangkat lunak, dan telekomunikasi seperti *Google*, *INTEL*, *HTC*, *Motorola*, *Qualcomm*, dan *T-Mobile*. *Google* membuat kode-kode android dibawah lisensi *Apache* [6].

Sistem operasi Android berkembang dengan beberapa versi antara lain:

1. Android Versi 1.1
2. Android Versi 1.5 (*Cupcake*)
3. Android Versi 1.6 (*Donut*)
4. Android Versi 2.0/2.1 (*Eclair*)
5. Android Versi 2.2 (*Froyo*)
6. Android Versi 2.3-2.3.2/2.3.3-2.3.7 (*Gingerbread*)
7. Android Versi 3.1/3.2 (*Honeycomb*)
8. Android Versi 4.0.3-4.0.4(*ICS*)
9. Android Versi 4.1/4.2/4.3 (*Jelly Bean*)
10. Android Versi 4.4 (*Kit Kat*)
11. Android Versi 5.0/5.1 (*Lolipop*)
12. Android Versi 6.0 (*Marshmellow*)

13. Android Versi 7.0 (*Nougat*)
14. Android Versi 8.0 (*Oreo*)
15. Android Versi 9.0 (*Pie*)
16. Android Versi 10 (10)
17. Android Versi 11 (11)

I.2.1 Kelebihan Aplikasi Android

Aplikasi android memiliki beberapa kelebihan antara lain:

1. *User Friendly,* sistem android sangat mudah untuk dijalankan. Hanya membutuhkan waktu sebentar saja untuk mempelajari sistem android.
2. Pengguna akan sangat mudah mendapat beragam notifikasi dari *smartphone*. Untuk mendapatkannya, pengguna bisa mengatur beberapa akun yang dimiliki seperti *SMS, Email, Voice Dial,* dan lainnya.
3. Keunggulan lainnya terdapat dari segi tampilan sistem android yang menarik dan tidak kalah baiknya dengan *iOS* (*Apple*). Hal ini dikarenakan dari awal, android mengusung konsep dan teknologi *iOS* hanya saja android merupakan versi murah dari *iOS*.
4. Sistem operasi memiliki konsep *open source* yang mana pengguna dapat bebas mengembangkan sistem android versi miliknya sendiri. Sehingga akan banyak sekali *Custom ROM* yang dapat digunakan.
5. Tersedia beragam pilihan aplikasi menarik. Dari mulai aplikasi gratis hingga aplikasi berbayar [5].

I.2.2 Kekurangan Aplikasi Android

Aplikasi android memiliki beberapa kekurangan antara lain:

1. *Update System* yang kurang efektif. Sistem android sering mengalami peningkatan versi yang ditawarkan kepada pelanggan. Namun untuk meng*update* sistem android bukan hal yang mudah. Pengguna diharuskan untuk menunggu masing-masing *vendor* merilis resmi *update* terbaru dari sistem android tersebut.
2. Baterai yang cepat habis, mungkin hal ini sering dialami oleh pengguna sistem android. Apalagi jika sering menyalakan paket data serta

menggunakan *widget* dan aplikasi yang berjalan terlalu berlebihan sehingga menyebabkan daya baterai berkurang dengan cepat.

3. Sering mengalami lemot atau lag, hal ini biasanya berkaitan dengan spesifikasi dari masing-masing perangkat seluler. Namun jika sistem android memang tidak bersahabat dengan aplikasi-aplikasi yang dimiliki tentu akan berdampak pada leletnya penggunaan *smartphone*. Hal ini dikaitkan dengan *RAM* atau *Chipset* yang kurang memadai [5].

I.3 Konsep Dasar Layanan PetShop

Memelihara hewan bukan hanya sekedar memelihara layaknya hewan ternak, pemilik hewan juga harus memerhatikan yang disebut dengan kesejahteraan hewan (*animal welfare*). Banyak hal yang harus dipenuhi oleh pemilik hewan untuk menjamin hewan peliharaannya sejahtera. Memelihara hewan diikuti dengan adanya tuntutan berupa kewajiban untuk bertanggung jawab terhadap keberlangsungan hidup hewan peliharaannya. Bahkan pemilik hewan peliharaan harus memperlakukan hewannya dengan manusiawi. Tidak hanya memastikan bahwa hewan peliharaannya dapat hidup, tetapi juga pemilik harus mampu memastikan hewan peliharaannya dalam kondisi sehat secara fisik dan mental serta tidak kekurangan suatu apapun, seperti pakan, minuman dan tempat berteduh. Memelihara *companion animal* dapat dijadikan sebagai sarana rekreasi dengan cara bermain bersama hewan peliharaan, mengisi waktu luang dengan memandikan atau mengajak berjalan-jalan hewan peliharaan [7].

Pet Shop adalah tempat penjualan perlengkapan hewan peliharaan, penitipan hewan peliharaan, juga pelayanan kesehatan untuk hewan peliharaan. Di zaman serba modern ini semakin banyak orang yang memiliki kecintaan terhadap hewan peliharaan termasuk kucing dan anjing, bahkan reptil juga ada. Kecintaan orang dengan hewan peliharaannya ini membuat mereka rela mengeluarkan biaya yang cukup banyak demi memenuhi kebutuhan hewan peliharaan mereka seperti: makanan, sampo, perawatan hewan, kandang, dan kebutuhan yang lain.

Jasa hotel hewan yang kebanjiran pelanggan, terutama saat mudik atau hari besar, banyak orang yang memilih untuk liburan. Hewan peliharaan mungkin akan sulit dibawa sehingga banyak yang memilih untuk menitipkan mereka di *pet shop*. Bisnis

pet shop merupakan bisnis yang cukup sederhana dan mudah dikelola dengan menyediakan berbagai macam kebutuhan hewan peliharaan. Untuk membuka bisnis *pet shop* memang harus memiliki modal yang cukup besar dan pengetahuan yang tinggi dalam perawatan hewan [8].

I.3.1 Grooming

Grooming harus diperhatikan dengan baik untuk menjaga kesehatan hewan sekaligus pemilik hewan. *Grooming* adalah tindakan menghilangkan rambut-rambut atau bulu-bulu yang telah mati. Tindakan *grooming* ini dilakukan supaya hewan tidak melakukan *grooming* sendiri dengan cara memijat-mijat bulunya, yang justru akan membahayakan kesehatannya. Selain itu dengan melakukan *grooming* secara teratur, akan meminimalisir jumlah bulu-bulu yang rontok, yang kadang-kadang akan menimbulkan alergi pada sejumlah orang yang tidak tahan dengan bulu hewan. *Grooming* juga dimaksudkan untuk mencari kotoran maupun telur pinjal yang mungkin ada. Selain itu pada waktu *grooming* kita juga bisa mengecek kesehatan kulit, mengetahui adanya bulu yang kusut, kulit yang rusak atau memerah, atau gangguan-gangguan lain pada kulit [9].

I.3.2 Pet Hotel

Penitipan hewan merupakan usaha yang menyediakan layanan penitipan hewan. Pasal 1 angka 21 Undang-Undang Nomor 41 Tahun 2014 tentang Peternakan dan Kesehatan Hewan, tidak mengenal istilah penitipan hewan. Akan tetapi dikenal dengan istilah bidang kesehatan hewan, adalah kegiatan yang menghasilkan produk dan/atau jasa yang menunjang upaya dalam mewujudkan kesehatan hewan. Berbicara mengenai kesehatan hewan, tidak lain memuat segala urusan yang berkaitan dengan perlindungan sumber daya hewan yang meliputi kesehatan hewan, lingkungan hewan, kesejahteraan hewan, dan peningkatan akses pasar untuk mendukung kedaulatan, kemandirian, bahkan ketahanan pangan yang bersumber dari hewan. Penjelasan di atas membutuhkan suatu tindakan secara nyata, salah satunya yaitu tindakan yang berkaitan dengan penangkapan dan penanganan hewan, penempatan dan pengadangan hewan, pemeliharaan dan perawatan hewan, pengangkutan hewan, serta perlakuan dan pengayoman yang wajar terhadap hewan

sebagai makhluk hidup yang mencakup perihal pemeliharaan, pengamanan, perawatan, dan pengayoman hewan dilakukan dengan sebaik mungkin sehingga hewan bebas dari rasa lapar dan haus, rasa sakit, penganiayaan dan penyalahgunaan, serta rasa takut dan tertekan [10].

I.4 Marketplace

Marketplace merupakan situs *online* yang berperan sebagai pihak ketiga yang berperan menyambungkan penjual dan pembeli di dunia maya atau *online*. Untuk memudahkan pemahaman, *marketplace* bisa juga disebut pasar dalam bentuk *online*. Biasanya kerja sama yang dilakukan oleh *marketplace* adalah menyediakan lapak *online* untuk digunakan berjualan secara *online* juga.

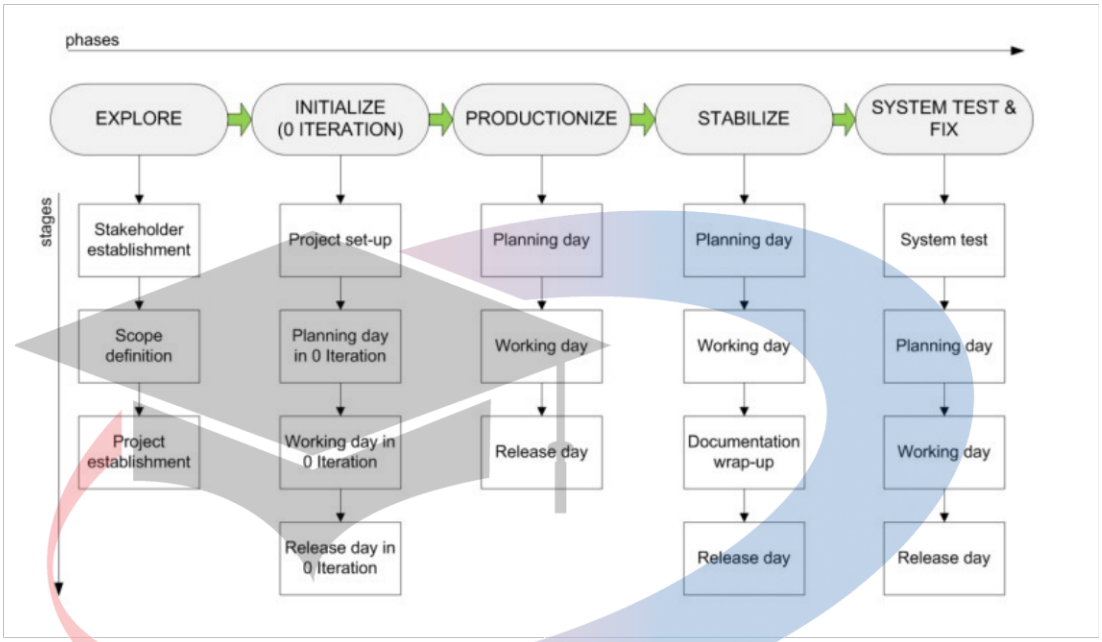
Keunggulan menggunakan *marketplace* adalah:

1. Membangun sebuah toko pada *marketplace* tidak memerlukan biaya sama sekali.
2. Dengan bergabung pada sebuah *marketplace* seseorang penjual tak harus pusing dengan promosi.
3. Karena *website* sudah disediakan dan tidak membangun sendiri, maka saat terjadi masalah atau *error* pada *website*, pemilik *online shop* tidak perlu repot untuk memperbaikinya.
4. Biasanya para pembeli lebih percaya dengan *marketplace* dibandingkan *online shop* [11].

I.5 Mobile-D

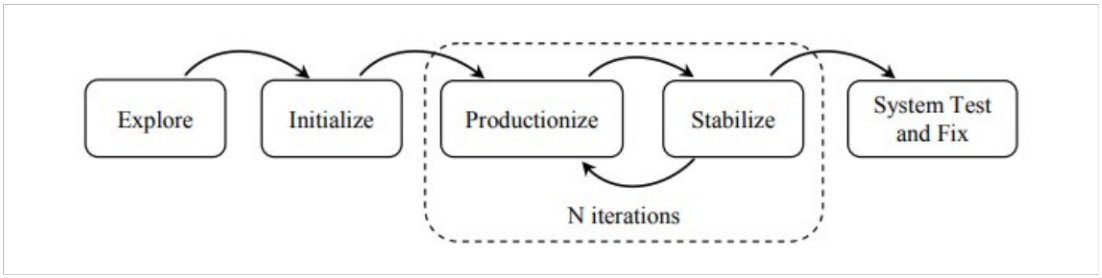
Mobile-D merupakan upaya pertama dari penggabungan metode *agile* untuk pengembangan aplikasi *mobile*. *Mobile-D* pertama dikenalkan oleh Abrahamsson et el pada tahun 2004. metodologi ini merupakan penggabungan dari *Extreme Programing (practices), Crystal Methodologies (scalability), dan Rational Unified Process (coverage)* [12]. *Mobile-D* terdiri dari lima fase yaitu *Explore, Initialize, Productionize, Stabilize, System Test and Fix*. Masing-masing fase ini memiliki sejumlah tahapan, tugas dan praktik terkait. *Mobile-D* telah diterapkan dalam proyek pengembangan, dan beberapa keuntungan telah diamati, seperti peningkatan visibilitas kemajuan, deteksi dan perbaikan lebih awal dalam masalah teknis,

kerapatan cacat rendah dalam produk akhir, dan kemajuan yang konstan dalam pengembangan [13]. Fase tersebut akan digambarkan dalam bagan gambar 2.1 berikut:



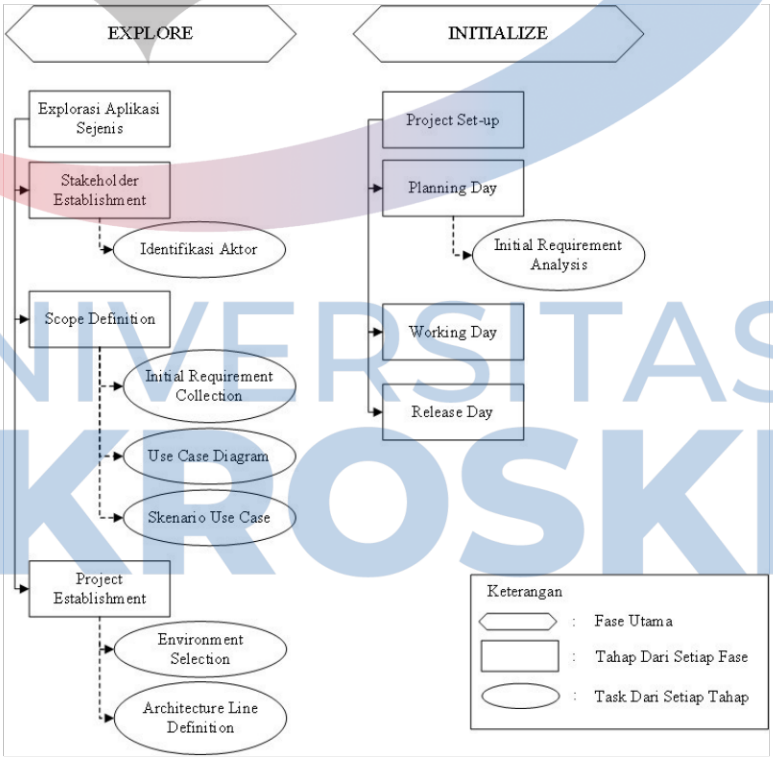
Gambar 2. 1 Fase dan Tahap *Mobile-D*

Proses pada *Mobile-D* mencakup lima tahapan yang akan dijalankan secara urutan inkremental sebagian atau parsial. Tujuan dari fase pertama yang disebut *Explore* adalah untuk mempersiapkan fondasi untuk pengembangan perangkat lunak seperti identifikasi aktor, mendefiniskan cakupan. Fase *Initialize*, dimana pada fase ini harus mendeskripsikan dan mempersiapkan semua komponen aplikasi serta untuk memprediksi semua isu kritis. Kemudian fase *Productionize* dan *Stabilize* dijalankan secara iteratif untuk mengembangkan semua fitur produk atau disebut juga tahap implementasi. Terakhir adalah fase *System Text and Fix* bertujuan untuk mendeteksi apakah sistem yang dihasilkan mengimplementasi fungsionalitas yang ditentukan oleh pelanggan dengan benar. Ini juga memberikan umpan balik kepada tim proyek mengenai fungsionalitas sistem dan informasi cacat untuk pemasangan terakhir dari proses *Mobile-D* [14]. Proses tersebut akan digambarkan pada Gambar 2.2 berikut:



Gambar 2. 2 Proses pada *Mobile-D*

Pada penulisan ini tidak menerapkan semua fase dan tahap serta *task* yang ada pada Metode *Mobile-D*. Penulis hanya mengadopsi beberapa tahap sesuai fase dan tahap serta *task* dari Metode *Mobile-D* sesuai dengan kebutuhan penulisan ini. Adopsi yang penulis pakai dari metode *Mobile-D* akan digambarkan pada Gambar 2.3 berikut:



Gambar 2. 3 Fase, Tahap dan *Task* dari adopsi metodologi *Mobile-D*

I.5.1 Explore

Fase ini harus membuat rencana perancangan terlebih dahulu serta menyusun karakteristik proyek itu sendiri. Pada fase ini proses perancangan terdiri dari empat

tahap, yaitu *stakeholder establishment*, *scope definition*, dan *project establishment*. Tujuan dari tahapan *explore* adalah pada perencanaan dan pembentukan pada proyek baru [13].

1. Eksplorasi Aplikasi Sejenis

   Tahap ini melakukan explorasi untuk mengidentifikasi kebutuhan sistem aplikasi PetShop yang diambil dari aplikasi sejenis sehingga dapat disimpulkan dalam tabel definisi kebutuhan sistem.

2. *Stakeholder Establishment*

   Tahap ini mengidentifikasi semua kelompok pemangku kepentingan serta sumber daya yang relevan dibutuhkan untuk proses perumusan peran. *Task* yang ada pada tahap ini terbagi menjadi dua, yaitu *customer establishment* dan *stakeholder group establishment* [12]. Pada penulisan ini hanya mengadopsi *task customer establishment* untuk mengidentifikasi pemangku kepentingan yang akan di adopsi pada bagian identifikasi aktor.

3. *Scope Definition*

   Tahap dimana tujuan serta ruang lingkup perancangan perangkat lunak dirumuskan dan disepakati bersama. Tujuan dari tahap ini adalah untuk menentukan tujuan proyek yang baru jadi baik mengenai isi maupun *timeline* proyek. *Task* yang ada pada tahap ini terbagi menjadi dua, yaitu *initial requirement collection* dan *initial project planning* [12]. Pada penulisan ini hanya mengadopsi *task initial requirement collection* untuk mengidentifikasi kebutuhan awal sistem dimana penulis melakukan penggalian kebutuhan sistem dan juga menjabarkan spesifikasi kebutuhan yang akan dimodelkan dalam *use case diagram* dan skenario *use case*.

4. *Project Establishment*

   Tahap ini tujuannya untuk menentukan lingkungan teknis perancangan, serta lingkungan kerja yang dibutuhkan dan juga sumber daya manusia dalam proyek perancangan. *Task* terdapat pada tahap ini terbagi menjadi empat, yaitu *environment selection*, *personnel allocation*, *architecture line definition*, dan *procces establishment* [12]. Pada penulisan ini hanya mengadopsi *task environment selection* untuk menentukan lingkungan

perancangan dan *task architecture line definition* untuk memberikan gambaran umum serta deskripsi umum dari sistem yang dirancang.

I.5.2 Initialize

Fase ini mendeskripsikan dan mempersiapkan semua kebutuhan aplikasi yang akan dibangun secara iteratif. Fase ini terdiri dari 4 tahap, yaitu *project set-up*, *planning day*, *working day* dan *release day* [13].

1. *Project Set-up*

   Tahap pertama fase *initialize* terdiri dari tiga *task* yaitu, *environment set-up*, *training*, dan *customer communication establishment* [12]. Pada penulisan ini hanya mengadopsi *task environment set-up* untuk menyiapkan dan melakukan instalasi seluruh arsitektur perangkat lunak perancangan, sehingga saat proses perancangan sistem sudah mempunyai kesiapan sepenuhnya untuk pertumbuhan arsitektural sistem yang sistematis saat menerapkan seluruh kebutuhan selama fase perancangan perangkat lunak.

2. *Planning Day*

   Tahap kedua fase *initialize* terdiri dari dua *task* yaitu *architecture line planning* dan *initial requirement analysis* [12]. Pada penulisan ini hanya mengadopsi *task initial requirement analysis* untuk merumuskan dan menganalisis kebutuhan sistem yang akan dimodelkan dalam *class diagram*.

3. *Working Day*

   Tahap ketiga fase *initialize* tidak terdapat *task* pada tahapan ini, tujuan dari tahapan ini untuk menciptakan fungsionalitas yang mencakup sebagian besar elemen desain arsitektural dan juga menciptakan basis untuk fitur lainnya, *working day* membentuk fase awal untuk pengembangan pada hari yang sebenarnya (*productionize dan stabilize*) [14]. Pada tahap ini penulis melakukan perancangan sistem mulai dari perancangan basis data yang dimodelkan dalam *JSON Schema* serta struktur dan *wireframe* yang akan diberikan pada tahap *release day*.

4. *Release Day*

   Tahap keempat fase *initialize* ini melanjutkan yang ada pada tahap *working day* dimana penulis melakukan perancangan antarmuka akhir [14].

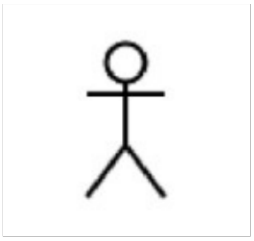
I.6 Bahasa UML

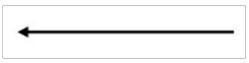
UML (*Unified Modeling Languange*) merupakan produk dari pendekatan berorientasi objek (*object-oriented approach*), yang biasanya dikontraskan dengan pendekatan matematis (*mathematical approach*). UML merupakan bahasa yang sederhana dan grafis. UML memanfaatkan diagram untuk mempresentasikan model. Diagram-diagram yang dipakai dalam UML adalah diagram *use case*, klas, sekuens, transisi *state*, aktivitas, dan komponen [15].

I.6.1 Use Case Diagram

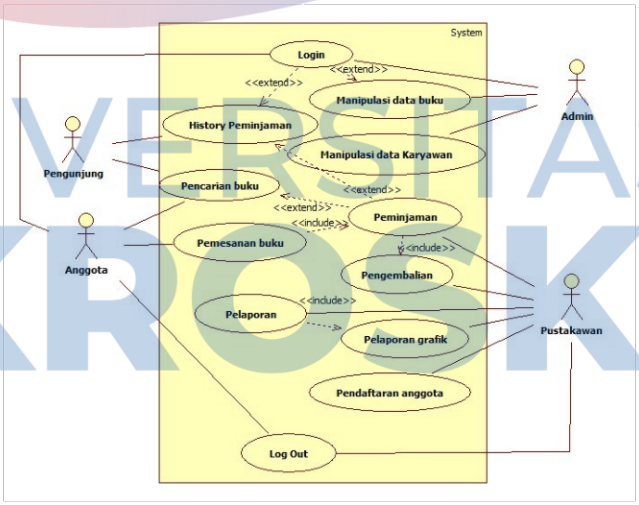
Use Case Diagram merupakan pemodelan untuk kelakuan sistem informasi yang akan dibuat. *Use Case* mendeskripsikan sebuah interaksi antara satu atau lebih *actor* dengan sistem informasi yang akan dibuat. *Use Case* digunakan untuk mengetahui fungsi apa saja yang ada di dalam sebuah sistem informasi dan siapa saja yang berhak menggunakan fungsi-fungsi itu [16].

Tabel 2. 1 Simbol-Simbol *Use Case Diagram*

No	Simbol	Nama	Keterangan
1.		*Use Case*	Fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertuktar pesan antar unit atau *actor*, biasanya dinyatakan dengan menggunakan kata kerja awal fase nama *use case*.
2.		Aktor	Berinteraksi dengan sistem informasi yang akan dibuat diluar sistem informasi yang akan dibuat itu sendiri, walaupun simbol dari aktor adalah gambar orang, tapi aktor belum tentu merupakan orang, biasanya dinyatakan menggunakan kata benda diawali fase nama *actor*.

3.		Asosiasi/ *Association*	Komunikasi antara *actor* dan *use case* yang berpartisipasi pada *use case* atau *use case* memiliki interaksi dengan *actor*.
4.		*Extend*	Relasi *use case* tambahan ke sebuah *use case* dimana *use case* yang ditambahkan dapat berdiri sendiri walaupun tanpa *use case* tambahan itu.
5.		*Include*	Relasi *case* tambahan ke sebuah *use case* dimana *use case* yang ditambahkan memerlukan *use case* ini untuk menjalankan fungsi atau sebagai syarat dijalankan *use case*.

Berikut ini merupakan contoh dari *use case diagram*



Gambar 2. 4 Contoh Penggunaan *Use Case Diagram*

Diagram *use case* yang dibuat harus cukup jelas dan detail, termasuk pemilihan nama *use case* yang pendek dan ringkas. Penggambaran diagram *use case* dan penamaannya yang jelas dan ringkas, sangatlah penting guna menghindari adanya risiko salah interpretasi dari pembaca atau pengguna.

Pedoman umum dalam mendeskripsikan *use case* adalah sekitar 1-2 halaman per *use case*. Pendekatan terstruktur untuk deskripsi *use case* berisi informasi sebagai berikut:

1. Nama.
2. Deskripsi singkat.
3. Prakondisi: prasyarat untuk keberhasilan eksekusi.
4. Pascakondisi: status sistem setelah eksekusi berhasil.
5. Situasi kesalahan: kesalahan yang relevan dengan domain masalah.
6. Status sistem tentang terjadinya kesalahan.
7. Aktor yang berkomunikasi dengan *use case*.
8. Pemicu: peristiwa yang memulai *use case*.
9. Proses standar: langkah individu yang harus diambil.
10. Proses alternatif: penyimpanan dari proses standar.

Berikut contoh deskripsi *use case* tentang pemesanan ruang kuliah dalam sistem administrasi siswa. Deskripsinya dapat disederhanakan tetapi diharapkan cukup memenuhi tujuan. Proses standar dan proses alternatif dapat disempurnakan lebih lanjut atau situasi kesalahan lainnya dan proses alternatif dapat dipertimbangkan [17].

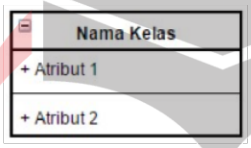
Nama:	Pemesanan ruang kuliah
Deskripsi:	Seorang karyawan memesan ruang kuliah suatu event
Prakondisi:	Karyawan diotorisasi memesan ruang kuliah Karyawan log-in ke sistem
Pascakondisi:	Ruang Kuliah telah dipesan
Kondisi error	Tidak tersedia ruang kuliah yang kosong
Status sistem ketika terjadi error	Karyawan gagal/belum berhasil memesan ruang kuliah
Aktor	Karyawan
Pemicu	Karyawan memerlukan ruang kuliah
Proses standar	1> Karyawan memilih ruang kuliah 2> Karyawan memilih tanggal 3> Sistem mengonfirmasi bahwa ruang kuliah kosong 4> Karyawan konfirmasi pesanan
Proses Alternatif	3'> Ruang kuliah tidak kosong 4'> Sistem mengajukan sebuah alternatif ruang kuliah 5'> Karyawan memilih alternatif ruang kuliah dan konfirmasi pesanan

Gambar 2. 5 Contoh Penggunaan Deskripsi *Use Case*

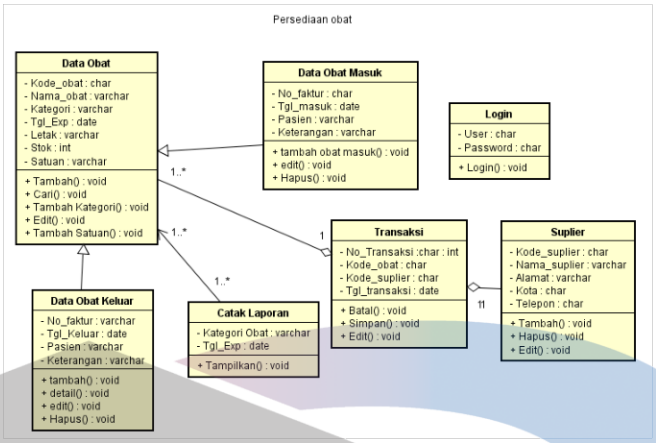
I.6.2 Class Diagram

Class Diagram menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem. *Class diagram* mendeskripsikan jenis-jenis objek dalam sistem dan berbagai hubungan statis yang terdapat diantara mereka [16].

Tabel 2. 2 Simbol-Simbol *Class Diagram*

No	Simbol	Nama	Keterangan
1.		*Package*	*Package* merupakan sebuah bungkusan dari satu atau lebih kelas.
2.		Operasi	Kelas pada struktur sistem.
3.		Antar muka / *interface*	Sama dengan konsep *interface* dalam pemograman berorientasi objek.
4.		Asosiasi	Relasi antara kelas dengan makna umum, asosiasi biasanya juga disertai dengan *multiplicity*.
5.		Asosiasi berarah/*directed association*	Relasi antara kelas dengan makna kelas yang satu digunakan oleh kelas yang lain, asosiasi biasanya juga disertai dengan *multiplicity*.
6.		Generalisasi	Relasi antara kelas dengan makna generalisasi-generalisasi (umum khusus).
7.		Agregasi	Relasi antara kelas dengan makna semuan-bagian (*whole-part*).

Berikut ini merupakan contoh dari *class diagram*



Gambar 2. 6 Contoh Penggunaan *Class Diagram*

I.7 Basis Data

Basis data yang juga dikenal sebagai *database*, terdiri dari kata basis dan data. Data merupakan catatan atas kumpulan fakta yang mewakili suatu objek. Data memiliki ciri bersifat mentah dan tidak memiliki konteks, sedangkan basis atau *base* dapat diartikan sebagai markas, tempat berkumpul dari suatu objek atau representasi objek. Atau dengan kata lain basis data didefinisikan sebagai sekumpulan data yang terintegrasi, yang diorganisasi untuk memenuhi kebutuhan para pemakai di dalam suatu organisasi. Maksud dari terintegrasi adalah, setiap data (yang nantinya kita sebut sebagai tabel) akan memiliki hubungan dengan data yang lainnya (data yang terhubung) [18].

I.8 Firebase

Firebase adalah suatu layanan *BaaS* (*Backend as a Service*) yang ditawarkan oleh *Google* untuk mempercepat pekerjaan *developer*. Dengan menggunakan *firebase*, *apps developer* bisa fokus dalam mengembangkan aplikasi tanpa memberikan *effort* yang besar untuk urusan *backend*.

Singkat cerita mengenai sejarah dari *Firebase* didirikan pertama kali pada tahun 2011 oleh Andrew Lee dan James Tamplin. Produk *Firebase* yang pertama kali adalah *Realtime Database*. *Realtime Database* digunakan *developer* untuk menyimpan data dan *synchronize* ke banyak *user*. Kemudian ia berkembang sebagai

layanan pengembang aplikasi. Pada bulan Oktober 2014, perusahaan tersebut diakuisisi oleh Google.

Mengenai segi layanan, kamu bisa memanfaatkan dan menggunakan layanan *Firebase* secara *free* (gratis). Tentu saja dengan adanya batasan-batasan tertentu. Layanan-layanan yang tersedia dari *Firebase* ada 2 pilihan, di antarannya:

1. *Spark*: kita bisa menggunakan layanan secara gratis.
2. *Blaze*: kita akan dikenakan biaya sesuai dengan pemakaian layanan.

Adapun jenis-jenis atau fitur-fitur dari *Firebase,* antara lain:

1. *Firebase Realtime Database*

   Firebase Realtime Database adalah *database* yang di-*host* melalui *cloud*. Data disimpan dan dieksekusi dalam bentuk JSON dan disinkronkan secara *realtime* ke setiap *user* yang terkoneksi. Hal ini berfungsi memudahkan kamu dalam mengelola suatu *database* dengan skala yang cukup besar. Ketika kamu membuat aplikasi lintas-*platform*/*multiplatform* menggunakan SDK Android, *iOS*, dan juga JS(*JavaScript*), semua pengguna akan berbagi sebuah *instance Realtime Database* dan menerima *update*-an data secara serentak dan otomatis.

2. *Firebase Cloud Firestore*

   Cloud Firestore merupakan *database NoSQL* yang di *hosting* di *cloud* dan dapat diakses melalui *SDK real* oleh aplikasi *iOS*, Android dan web. Seperti halnya *Firebase Realtime Database*, *Cloud Firestore* membuat datamu tetap terkoneksi di aplikasi *user* melalui *listener realtime* dan menawarkan layanan secara *offline* untuk aplikasi seluler dan web. Dengan begitu, kamu dapat membuat aplikasi yang *powerfull*, responsif, dan mampu bekerja tanpa bergantung pada latensi koneksi internet.

3. *Firebase Analytics*

   Firebase Analytics adalah salah satu fitur pada *Firebase* yang digunakan sebagai koleksi data dan *reporting* untuk aplikasi Android maupun *iOS*. Koleksi data pun bervariasi. Sebagai contoh, dalam membuat suatu laporan atau *report* untuk pengguna aplikasi di negara Indonesia saja, atau mungkin negara lain seperti Singapura. Dan dapat juga bisa melihat bagian mana saja dari aplikasi yang paling sering digunakan oleh *user*.

4. *Firebase Cloud Messaging and Notifications*

   FCM (*Firebase Cloud Messaging*) yaitu menyediakan koneksi yang handal dan tentunya hemat baterai antar server maupun antar *device*. Sehingga dapat mengirim dan menerima pesan serta notifikasi di Android, *iOS*, dan web tanpa perlu biaya.

5. *Firebase Authentication*

   Firebase Authentication adalah salah satu layanan *back-end*, fitur Android dan iOS, SDK yang mudah digunakan, dan tampilan *interfaces* yang siap pakai untuk mengautentikasi pengguna ke aplikasi yang kamu buat. *Firebase Authentication* mendukung autentikasi menggunakan nomor telepon, sandi, penyedia identitas popular seperti Google, Facebook, dan sebagainya.

6. *Firebase Hosting*

   Firebase Hosting, suatu layanan *hosting* konten web. Hanya dengan satu instruksi, dapat mengimplementasikan aplikasi web serta menyajikan konten statis maupun dinamis ke *CDN* (jaringan penayangan konten) *global* dengan cepat. *Firebase Hosting* mampu menayangkan konten melalui koneksi yang begitu aman, mengirimkan konten secara cepat, dan mendukung semua jenis konten untuk di *hosting*, mulai dari file *HTML* dan *CSS* status hingga *API* atau layanan *mikro Express.js*. [19]