BAB II
LANDASAN TEORI

2.1 *Smartphone*

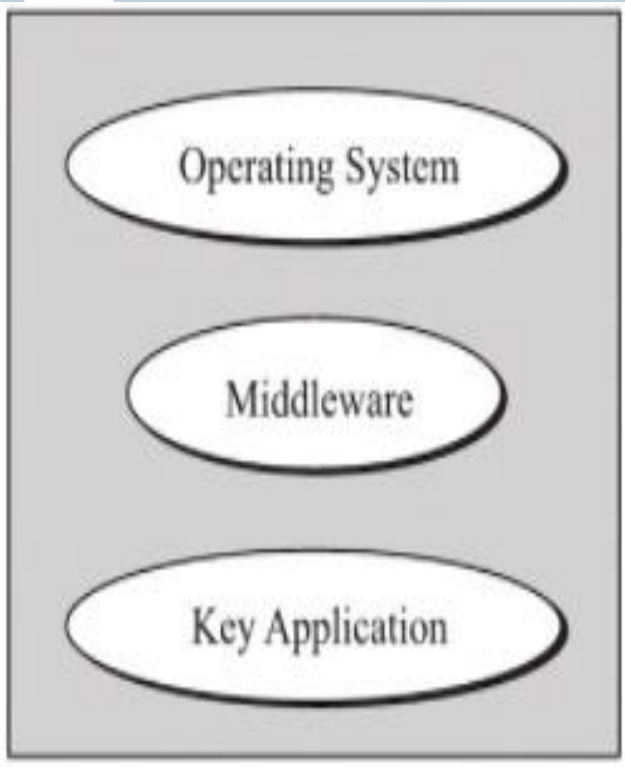
Dikutip dari Oxford Dictionary, smartphone merupakan telpon genggam yang memiliki fungsi seperti computer dan interface yang memiliki fitur layar sentuh, akses internet, dan sistem operasi yang memungkinkan untuk menjalankan aplikasi yang sudah didownload (Oxford, 2018).

Smartphone merupakan perangkat telepon mobile yang menggabungkan antara fungsi telepon genggam dengan fitur – fitur tambahan yang memiliki kemampuan layaknya komputer pribadi. Fitur – fitur yang dimaksud diantaranya voice communication, pemutar audio, pemutar video, aplikasi penjelajah internet, pengirim e-mail, fitur pendownload aplikasi, fitur gaming dan fitur – fitur lain yang meningkatkan kemampuan dari suatu telepon genggam (Carroll & Heiser, 2010).

Smartphone merupakan telepon genggam yang didalamnya telah dilengkapi dengan mikroprosesor, penyimpanan (memori), layar dan modem yang telah ada dalam satu telepon genggam. Smartphone menggabungkan fungsionalitas dari komputer pribadi dan telepon genggam, dimana didalamnya terdapat fitur – fitur seperti pemutar musik, internet, game, dan fitur – fitur pendukung lainnya (williams & Sawyer, 2015).

2.2 Android

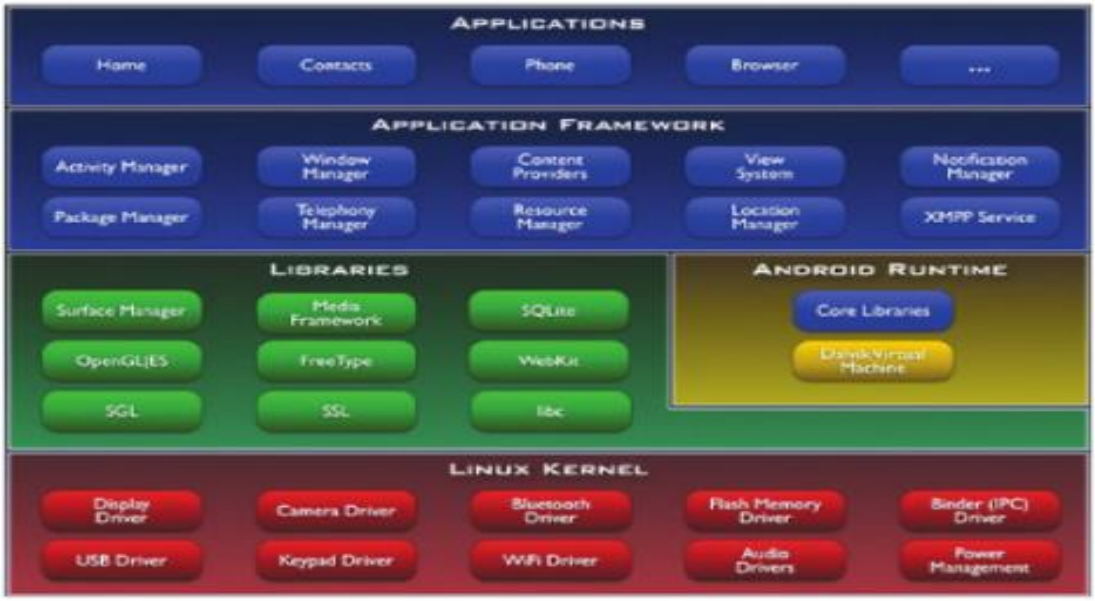
Android merupakan sebuah teknologi *open source* yang memungkinkan sebuah *software* untuk dimodifikasi dan didistribusi oleh para *device manufacturers*,dan *developer software*. Android merupakan sebuah sistem operasi berbasis Linux yang didesain untuk *touchscreen* pada perangkat *mobile* seperti *smartphone*, dan komputer tablet (Dixit, 2014). Pada gambar 2.1 terdapat 3 bagian software dalam android yang terdiri dari *operating sistem, middleware,* dan *key application.*



Gambar 2. 1 bagian utama sistem Android

Sumber: (Dixit, 2014)

pada Gambar 2.2 dalam Android terdapat 5 bagian lapisan arsitekturnya, yang meliputi, Aplikasi, *Framework* Aplikasi, *Libraries*, *Android Runtime*, dan *Linux Kernel* (Dixit, 2014).



Gambar 2. 2 Susunan Arsitektur Sistem Andorid

Sumber: (Dixit, 2014)

2.3 Aplikasi Mobile

Aplikasi *mobile* adalah *software* yang diciptakan untuk para pengguna perangkat genggam seperti *smartphone* dan *Personal Digital Assistans* (PDA). Jenis – jenis aplikasi mobile diantaranya adalah media sosial, aplikasi permainan, aplikasi peta, aplikasi berita, aplikasi bisnis, pemantau cuaca, dan pemutar lagu. (International Telecommunications Union, 2009).

2.4 *User interface*

User interface merupakan sebuah media visual sebagai tempat terjadinya interaksi antara *user* dengan komputer. Tujuan dari adanya *User interface* ini adalah membuat interaksi manusia dengan mesin menjadi lebih mudah, efisien, dan menyenangkan (*user friendly*). Desain *user interface* harus baik karena *user* akan lebih nyaman dalam mengoperasikan sistem komputer apabila komputernya lebih mudah digunakan, mudah untuk dimengerti, dan dapat menapai tujuan yang diharapkan oleh *user* dengan tingkat kekecewaan *user* yang minimal (Dictionary, 2018).

User interface Design merupakan proses membuat tampilan dalam *software* dengan berfokus pada tampilan dan *style*. Desainer membuat *user interface* yang akan membuat *user* lebih mudah dalam menggunakan *software*-nya (Foundation, 2018).

Dasar pembuatan *user interface* berfokus pada *user* agar sebuah sistem yang dipakai mudah untuk digunakan. UI membawa berbagai konsep dari *interaction design*, *visual design*, dan arsitektur informasi (usability.gov, 2018)

2.5 *8 Golden Rules*

Ben Shcneiderman menjelaskan metode 8 *golden rules* sebagai panduan untuk merancang *prototype* sebuah website sehingga memiliki tingkat *usability* yang baik. Diantaranya (Scheinderman & Plaisant, 2018):

1. Berupaya Untuk Selalu Konsisten.

Konsistensi dibutuhkan antar halaman yang masih berhubungan. Tujuannya adalah supaya *user*, terutama *user* yang baru memakainya, tetap dapat mengenali halaman. Dengan demikian akan membuat *user* nyaman dalam mengeksplorasi website.

2. Memungkinkan Penggunaan *Shortcut*.

Dalam merancang *user interface*, seorang *interface designer* harus memperhitungkan penggunaan fitur *shortcut* (icon, singkatan). Fitur ini berguna memudahkan *user* yang sudah ahli dalam menggunakan website tersebut.

3. Memberikan *Feedback* yang informatif bagi *user*.

Dalam membangun sebuah website sebaiknya setiap interaksi yang dilakukan disertai dengan *feedback* kepada *user*-nya Tujuan dari *Feedback* ini adalah agar *user* mengetahui aksi apa yang sedang berjalan.

4. Merancang Dialog Penutup.

Merancang sebuah dialog penutup bagi *user* ketika sudah menyelesaikan sebuah proses yang dijalankan oleh *user*. Tujuannya agar *user* tidak perlu menunggu apakah masih akan ada tahapan lain setelah menyelesaikan suatu proses.

5. Memberikan Penanganan Kesalahan Sederhana.

Dalam merancang sebuah *user interface*, *Interface Designer* harus membuat tampilan website yang sedemikian rupa sehingga *user* dapat terhindar dari kesalahan dan tidak menyebabkan kesalahan pada saat menjalankan proses. Jika terjadi kesalahan, sistem dapat langsung memberikan mekanisme penanganan kesalahan sederhana.

6. Mudah Kembali ke Aksi Sebelumnya.

Poin ini merupakan salah satu poin yang cukup penting dalam merancang sebuah *UI* dan menunjang *UX*. Contohnya adalah adanya tombol *back* (kembali), tombol *delete* (hapus), dan tombol *cancel* (batal). Hal ini bertujuan mengurangi kekuatiran dari *user* ketika terjadi kesalahan.

7. Mendukung internal *locus of control*

Menjadikan *user* sebagai pemegang kendali. Dalam merancang sebuah *UI* haru mengedepankan *user*. Maksudnya *user* harus menjadi inisiator bukan responden sehingga *user* dapat dengan bebas bernavigasi dan mengubah informasi yang dia miliki. Hal ini dapat meningkatkan kepuasan *user* terhadap *UI* website yang sangat mempengaruhi *UX* terhadap website tersebut.

8. Mengurangi Beban Ingatan Jangka Pendek

Dalam merancang *UI* harus memperhatikan tampilan halaman. Tampilan halaman harus sederhana, menarik, dan dapat dimengerti oleh *user*. Dengan tampilan yang baik dan menarik akan mengurangi beban ingatan

jangka pendek *user* dan menghindari *user* dari kebingunan menggunakan website-nya.

2.6 *E-Business/E-Commerce*

Pengertian *E-business* merujuk kepada integrase antara perusahaan dengan teknologi komunikasi. Tujuannya untuk menunjang fungsionalitas agar dapat menciptakan nilai lebih bagi bisnisnya, klien, dan rekan perusahaannya (Owilson, 2018).

Pengertian *E-commerce* merujuk kepada jangkauan luas dalam aktivitas bisnis secara online terhadap produk dan jasa. *E-commerce* biasanya berhubungan dengan aktivitas jual/beli termasuk engan pertukaran kepemilikan, dan pembayaran melalui jaringan internet. *E-commerce* memiliki beberapa tipe diantaranya (Andam, 2013):

- business-to-business (B2B),
- Business-to-*customer* (B2C),
- Business-to-Government (B2G),
- Dan Consumer-to-Consumer (C2C).

2.7 *Usability Testing*

Usability Testing merupakan metode yang digunakan untuk melakukan evaluasi suatu sistem kepada *user* dengan memberikan pertanyaan atau *task*. Tujuannya adalah mengetahui dimana *user* merasa kebingungan dan sulit dalam menggunakan

sistem agar menjadi bahan evaluasi bagi *researcher* dalam memperbaiki sistemnya (experienceux.co.uk, 2018).

Usability Testing merujuk kepada proses pengujian sebuah produk atau jasa kepada *user*. Tujuannya untuk menemukan masalah *usability* dari produk atau jasanya dan menentukan kepuasan partisipan terhadap produk atau jasa yang diuji (Usability.Gov, 2018).

Metode *usability* memiliki 5 aspek menurut (Nielsen, 2012) diantaranya:

1. *Learnability* (Kemampuan Belajar)

menjelaskan tingkat kemudahan pengguna untuk memenuhi task-task dasar ketika pertama kali mereka melihat/menggunakan hasil perancangan.

2. *Efficiency* (Tingkat Efisiensi)

menjelaskan tingkat kecepatan pengguna dalam menyelesaikan task-task setelah mereka mempelajari hasil perancangan.

3. *Memorability* (Kemampuan untuk Diingat)

menjelaskan tingkat kemudahan pengguna dalam menggunakan rancangan. Setelah tidak digunakan selama beberapa waktu.

4. *Errors* (Kemungkinan Kesalahan)

menjelaskan kemungkinan terjadinya error yang dilakukan oleh pengguna, tingkat kejengkelan terhadap error dan cara memperbaiki error.

5. *Satisfaction* (Tingkat Kepuasan)

menjelaskan tingkat kepuasan pengguna dalam menggunakan rancangan.

2.8 *System Usability Scale* (SUS)

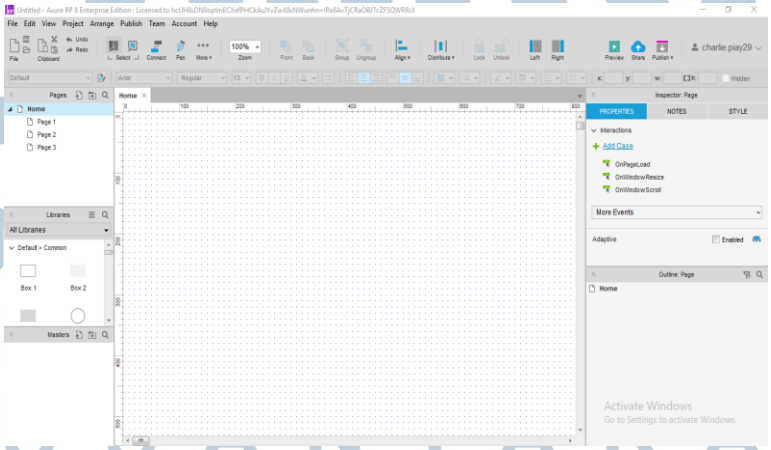
System Usability Scale merupakan metode penilainan tingkat *usability* dengan menggunakan skala *likert*. SUS terdiri dari 10 kuisioner yang pada tiap pertanyaan diberikan 5 skala jawaban mulai dari skala "sangat tidak setuju" sampai ke skala "sangat setuju". Metode SUS diciptakan oleh John Brooke pada tahun 1986. Metode ini memungkinkan digunakan untuk melakukan evaluasi secara menyeluruh termasuk *hardware, software,* aplikasi, dan *website.* SUS sangat mudah untuk dilakukan oleh partisipan. Selain itu SUS juga dapat digunakan pada sampel yang sedikit dengan hasil yang *reliable.*

Dalam menghitung skor SUS, penghitungan dilakukan sebagai berikut. Pertanyaan dengan nomor ganjil memiliki bobot pertanyaan yang bernada positif seperti "saya akan sering menggunakan aplikasi ini" dan "Saya merasa percaya diri menggunakan aplikasi ini". Kemudian partisipan menjawab dengan memilih dari skala yang ada, kemudian skala yang dipilih oleh partisipan dikurangi dengan 1 untuk mendapatkan nilai pertanyaannya. Untuk pertanyaan bernomor genap memiliki pertanyaan yang berbobot negative seperti "saya menemukan sistem tidak praktis" dan "saya berpikir bahwa sistem tidak konsisten". Partisipan kemudian memilih skala yang ada. Skala yang dipilih partisipan pada pertanyaan bernomor genap dijadikan pengurang dari 5 untuk mendapatkan nilai pertanyaannya. Setelah 10 pertanyaan

kuisioner terjawab maka untuk mendapatkan nilai akhir SUS, setiap nilai pertanyaan dijumlah dan hasil ppenjumlahan dikalikan dengan 2,5 untuk mendapatkan nilai SUS-nya. Dikalikan 2,5 karena jika tiap pertanyaan mendapat bobot maksimal, kemudian dijumlahkan, dan di kalikan dengan 2,5 maka hasil akhirnya akan 100 sesuai dengan skala tertinggi SUS. Skala SUS 0 – 100 (Brooke, 2013).

2.9 AXURE Rp

Axure RP adalah sebuah software untuk merancang *prototype*, rangka *prototype*, dokumentasi dan spesifikasi software yang ditujukan untuk aplikasi *mobile* dan *website*. Axure Rp merupakan *software tools* yang diciptakan untuk memungkinkan *non – programmers* untuk membuat *prototype* yang interaktif dengan dokumentasi dengan fitur *drag and drop*. Axure Rp memungkinkan pemakainya untuk membuat *prototype* secara interaktif, melakukan testing, dan medapatkan *feedback* dari testing yang dilakukan dan melakukan perbaikan terhadap protipe hingga mencapai solusi yang tepat (Axure, 2018). Gambar 2.3 merupakan tampilan home *axure*.



Gambar 2. 3 Tampilan Aplikasi Axure Rp

Dalam merancang *prototype* dengan software axure terdapat beberapa fitur yaitu (Axure, 2018)

1. *Dynamic Content*

   Konten dinamis adalah konten yang dapat berinteraksi dengan penggunanya. Contohnya adalah ketika pengguna melakukan *hover* pada suatu konten, maka akan muncul *pop up* informasi konten.

2. *Conditional Flow*

   Conditional flow adalah sistem yang ada pada *Axure RP* dimana sistem tersebut menentukan sebuah kondisi itu benar atau salah. Contohnya pengecekan *username* dan *password* pada saat melakukan *login*.

3. *Animations*

   Dalam *Axure RP*, terdapat fitur animasi yang memungkinkan adanya pergerakan yang halus disetiap obyeknya. Contohnya adalah *slideshow* gambar yang membuat gambar berganti secara *sliding* secara otomatis

4. *Data Driven*

   Data Driven dalam *Axure RP* memberikan fitur bagi *user* untuk mengatur penampilan data. Contohnya adalah filter huruf dari A sampai ke Z secara berurutan

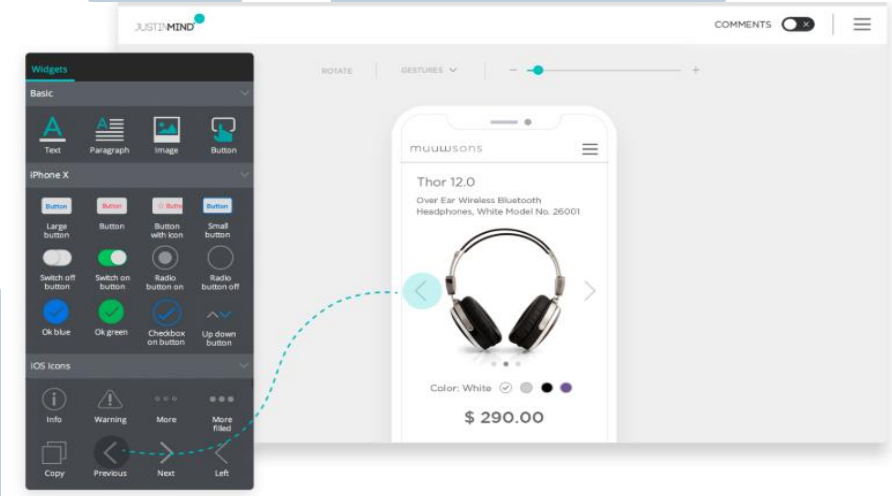
5. *Adaptive Views*

Adapative Views memungkinkan pengguna mendesain *prototype* dengan berbagai macam ukuran. Contohnya adalah ukuran pada *PC, smartphone,* dan *tablet.*

6. *Math Functions*

Axure RP memiliki fitur fungsi matematika untuk menghitung nilai dari variabel-variabel yang dimasukkan oleh pengguna. Contohnya harga di *shop*ping cart yang akan bertambah.

2.10 *Just in Mind Prototyping*



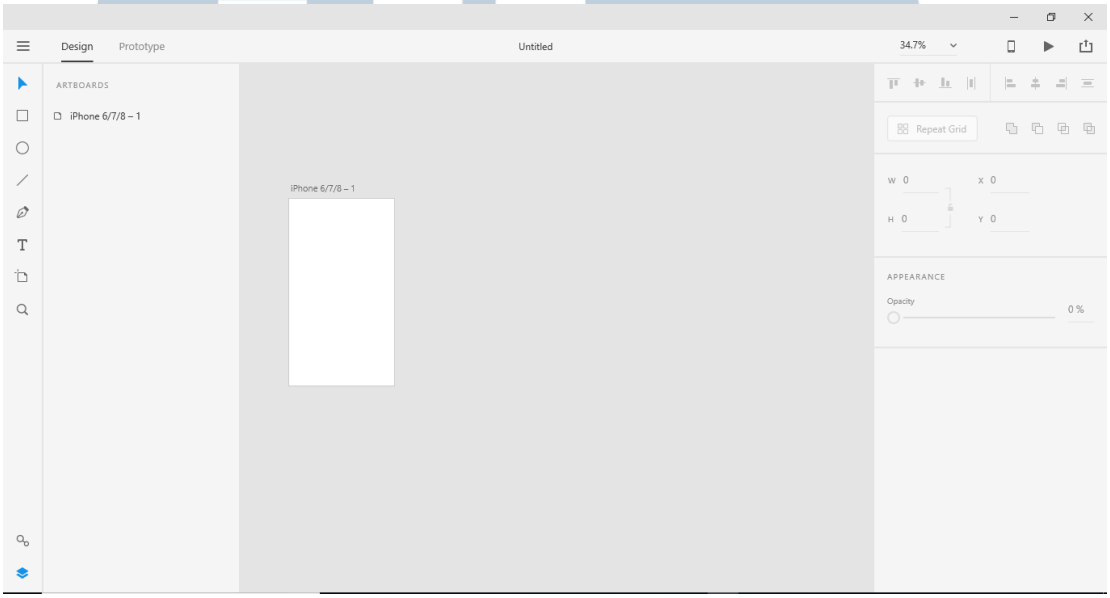
Gambar 2. 4 *just in mind prototyping*

Sumber: (mind, 2018)

Just in Mind merupakan sebuah *software prototyping* yang dapat menghasilkan tampilan aplikasi bukan hanya tampilannya namun juga fungsi yang ada pada tampilan tersebut seperti interaksi antar halaman, animasi, *gestures.* Hasil dari *prototype* juga dapat di *test* secara langsung.

Just in Mind memungkinkan *prototype* yang dibuat dapat di *share* ke rekan kerja dan dilakukan evaluasi secara langsung (mind, 2018) gambar 2.4 merupakan tampilan *just in mind*.

2.11 *Adobe XD*



Gambar 2. 5 tampilan Adobe XD

Adobe XD adalah *software* berbasis vektor yang dikembangkan dan diterbitkan oleh Adobe Inc untuk merancang dan membuat *prototype* untuk aplikasi web dan seluler. Perangkat Lunak tersedia untuk macOS dan Windows (Techcrunch, 2019).

Adobe XD ter intergrasi dengan seluruh *software* adobe inc. yang lain sehingga penggunanya dapat dengan mudah berinteraksi dengan *software* adobe yang lain (Adobe inc., 2019). Gambar 2.5 merupakan tampilan dari *adobe XD.*

2.12 Metode Pengembangan *Prototyping*

Software Prototyping merujuk pada pembangunan purwarupa aplikasi yang masih memiliki keterbatasan fungsi. Metode *prototyping* digunakan untuk memungkinkan user untuk melakukan evaluasi sistem yang dibangun sebelum diimplementasikan sesungguhnya. Berikut merupakan langkah – langkah melakukan *prototyping* menurut Roger S. Pressman yaitu Komunikasi, Perencanaan Secara Cepat, Permodelan Perancangan Secara Cepat, Pembentukan *Prototype*, dan Penyebaran & Pengiriman *Feedback* (Pressman, 2015)

2.13 Metode Pengembangan *Rapid Application Development (RAD)*

Rapid Application Development (RAD) merupakan metode pengembangan software yang di *planning* secara cepat. Pada metode ini, proses *development* fungsi aplikasi dilakukan secara bersamaan dengan proses pembangunan aplikasi. Tujuannya untuk mempersingkat waktu dari *development* aplikasi dan mempercepat *delivery* aplikasi. Tahapan dari *RAD* diantaranya *Bussiness Modeling, Data Modeling, Process Modeling, Application Generation,* dan *Testing and Turnover* (Tutorialspoint, 2018).

2.14 Metode Pengembangan *Waterfall*

Metode pengembangan *Waterfall* merupakan metode pengembangan yang palng sering digunakan dalam proses pembangunan *software*. Metode *waterfall* terdiri dari beberapa proses yaitu *Requierment Analysis, System Design, Implementation, Integration and Testing, Deployment of System,* dan *maintenance.* Setiap proses harus

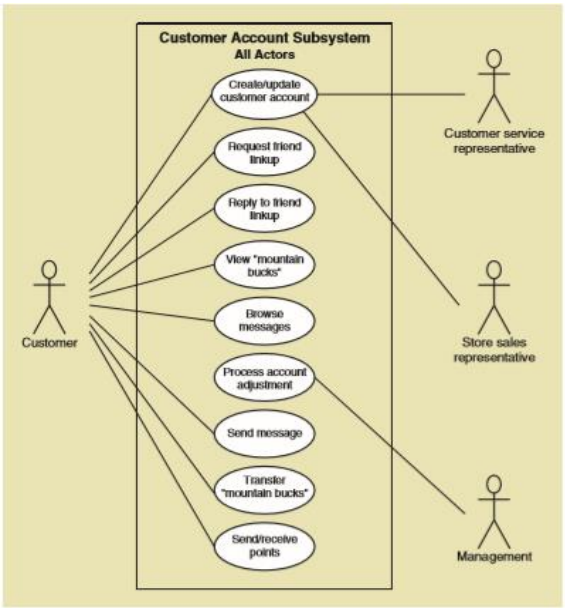
memiliki *goals* yang harus dicapai sebelum berpindah ke tahap selanjutnya (Tutorialspoint, 2018).

2.15 *Unified Modeling Language*

Unified Modeling Language (UML) merupakan sebuah model standar dalam melakukan perancangan pada sebuah sistem. UML digunakan sebagai "Bahasa" untuk melakukan visualisasi dan dokumentasi dari sistem (Dennis, Wixom, & Roth, 2015).

Penggunaan UML bertujuan untuk mempermudah komunikasi antara divisi *analyst* dan d*evelopment* dalam membangun sistem dengan memberikan "Bahasa" standar yaitu berupa diagram dan notasi dimana ditampilkan dengan interaksi/hubungan antara satu objek dengan obek lain (Nyisztor, 2018).

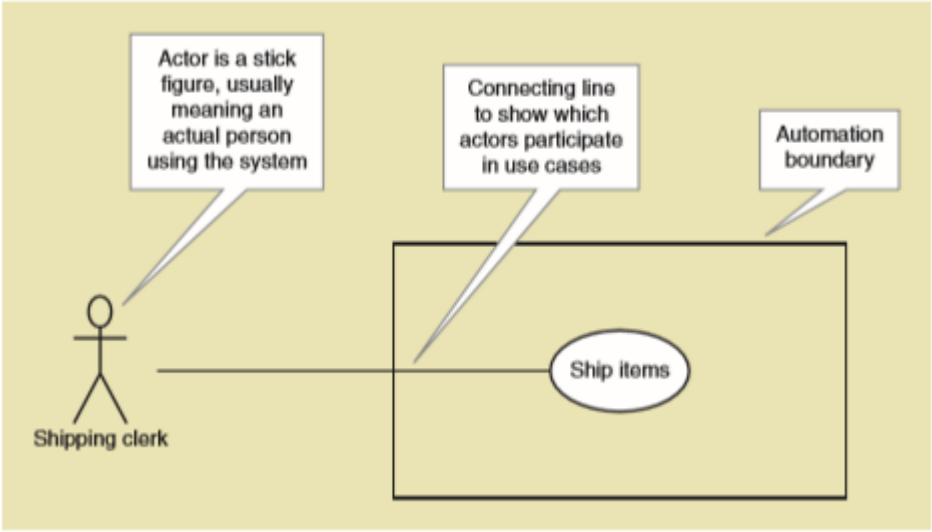
2.15.1 *Use Case Diagram*



Gambar 2. 6 Contoh *Use Case*

Sumber: (Satzinger, Jackson, & Burd, 2012)

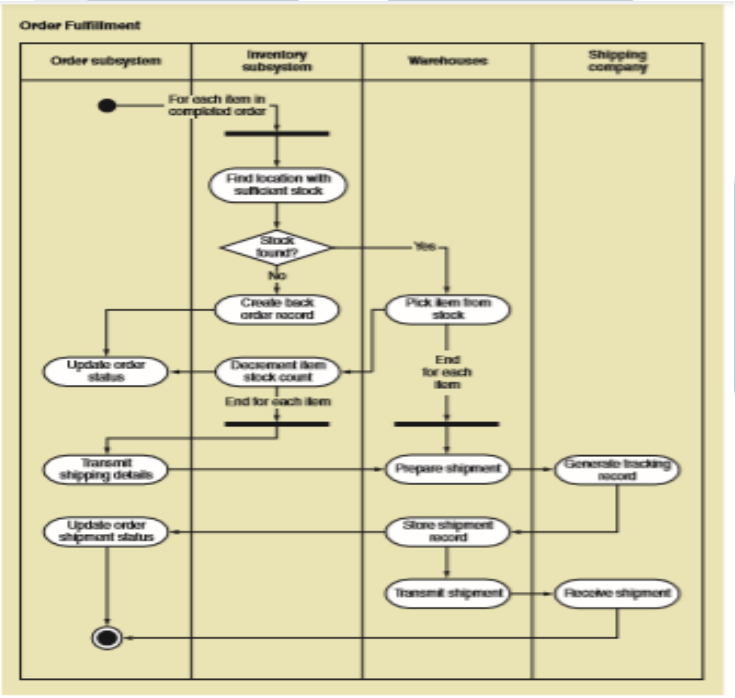
Use Case diagram merupakan model dari UML yang berguna untuk menampilkan *Use Case* dan relasinya terhadap *user*. *Use Case* terdiri dari Actor yang merupakan role dari aktornya (cth. *Customer*, *admin*) yang dilambangkan dengan *stickyman*, *Use Case* yang dilambangkan dengan bentuk *oval* dengan nama *case* didalamnya (cth. *shop*ping), garis penghubung actor dengan *Use Case* yang menampilkan interaksi antara *actor* dan *Use Case*, dan *automation boundary* sebagai pembatas antara *actor* dengan sistem yang dilambangkan dengan kotak mengitari *Use Case* Gambar 2.5 merupakan symbol yang digunakan dalam *use case diagram*.. Gambar 2.4 merupakan contoh dari *use case diagram.* (Satzinger, Jackson, & Burd, 2012).



Gambar 2. 7 simbol dalam *Use Case*

Sumber: (Satzinger, Jackson, & Burd, 2012)

2.15.2 *Activity Diagram*

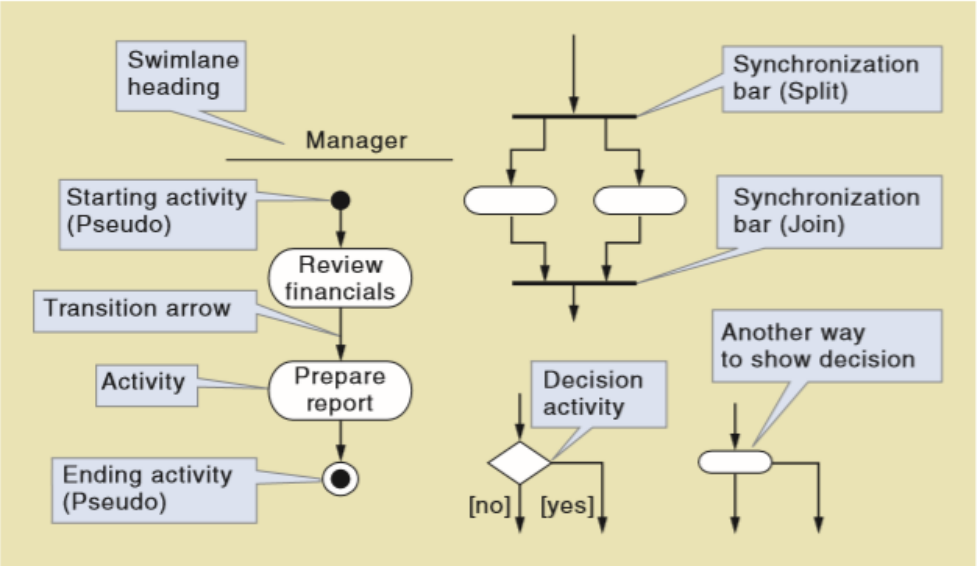


Gambar 2. 8 Contoh *Activity Diagram*

Sumber: (Satzinger, Jackson, & Burd, 2012)

Pada gambar 2.6 merupakan contoh *Activity Diagram*. *Activity Diagram* menjelaskan aktivitas dari berbagai *user* atau sistem, orang yang melakukan aktivitas, dan proses aktivitas yang berjalan. *Activity Diagram* menggambarkan alur aktivitas dalam sistem yang sedang dirancang mulai dari *start*, munculnya *decision*, hingga akhir. Aktivitas individu digambarkan dengan simbol *oval*. Panah pengubung merepresentasikan urutan antar aktivitas. Lingkaran hitam merepresentasikan awal dan

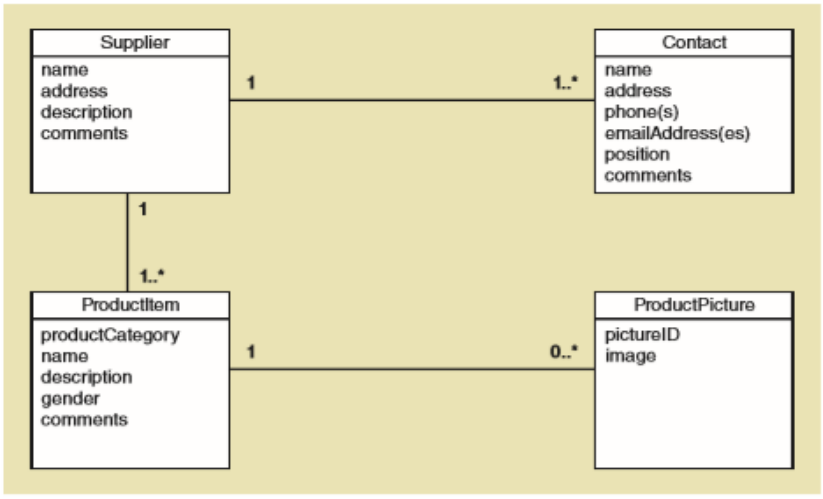
akhir dari *workflow*-nya. Berikut adala simbol lengkap dari *Activity Diagram* pada gambar 2.7 (Satzinger, Jackson, & Burd, 2012).



Gambar 2. 9 simbol *Activity Diagram*

Sumber: (Satzinger, Jackson, & Burd, 2012)

2.15.3 *Class Diagram*



Gambar 2. 10 Contoh *Class Diagram*

Sumber: (Satzinger, Jackson, & Burd, 2012)

Pada gambar 2.8 merupakan contoh *class diagram*. *Class diagram* merupakan diagram yang menggambarkan struktur dan deskripsi dari sebuah *class* dan hubungan mereka satu sama lain (Satzinger, Jackson, & Burd, 2012). Terdapat beberapa hubungan antar kelas diantaranya (Nyisztor, 2018):

a) Asosiasi, menggambarkan *class* yang memiliki atribut berupa kelas lain.

b) Agregasi, hubungan yang menyatakan bagian ("terdiri atas").

c) Pewarisan, hubungan hirarki antar *class*. Memiliki sub *class*.

d) Dinamis, hubungan rangkaian pesan yang dikirimkan antar kelas

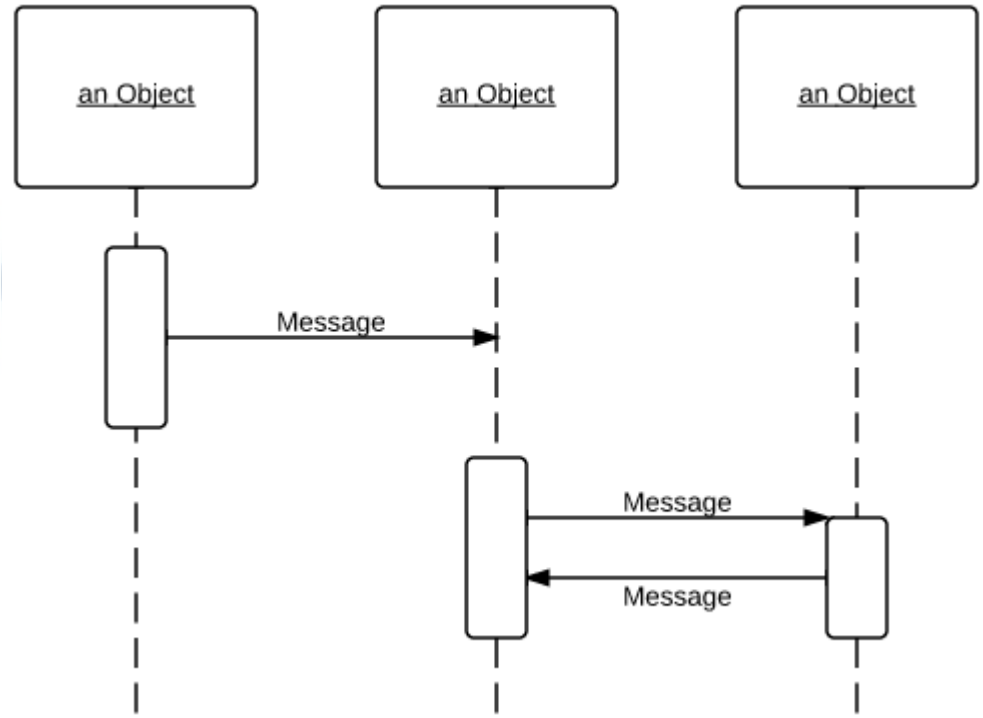
Class memiliki 3 area pokok yaitu Nama, Atribut, dan metode. Atribut dan metode memiliki berbagai macam sifat yaitu (Dennis, Wixom, & Roth, 2015):

a) *Private*, tidak dapat dipanggil dari luar *class* yang bersangkutan.

b) *Protected*, hanya dapat dipanggil oleh *class* yang bersangkutan dan anak – anak kelasnya.

c) *Public*, dapat dipanggil oleh siapa saja.

2.15.4 *Sequence diagram*



Gambar 2. 11 contoh *sequence diagram*.

Sumber: (Dennis, Wixom, & Roth, 2015)

Sequence diagram menggambarkan interaksi antar objek berupa *message* yang diambarkan terhadap waktu. *Message* yang dikirimkan akan mendapatkan return value. *Sequence diagram* terdiri dari dimensi vertikal (waktu) dan objek (horizontal) (Dennis, Wixom, & Roth, 2015).

Message digambarkan dengan garis panah ke objek tujuan. Dan balasan message akan dikirimkan kembali ke aktor dengan simbol garis panah putus – putus. Selain itu lama waktu interaksi ditunjukan dengan *activation bar*. Gambar 2.9 merupakan contoh *Sequence diagram* (Satzinger, Jackson, & Burd, 2012).

2.16 Penelitian Terdahulu

Tabel 2. 1 Tabel penelitian terdahulu

No	Nama	Judul	Metode	Obek Penelitian	Hasil penelitian
1	Egia Rosi Subhiyakto, Danang Wahyu Utomo Prosiding seminar nasional disiplin ilmu & *call for papers* UNISBANK ke -3 2017 Halaman 57-62	Analisis dan Perancangan Aplikasi Pemodelan Kebutuhan Perangkat Lunak dengan Metode *Prototyping*	Menggunakan metode Prototyping Dalam merancang tampilan antarmuka aplikasi pemodelan kebutuhan software. -kali	Aplikasi Pemodelan kebutuhan perangkat lunak.	Hasil analisa dan perancanga n aplikasi pemodelan kebutuhan perangkat lunak berupa *Use Case*, *class* diagram, dan rancangan tampilan aplikasi.2
2	R.D. Banimahendra, H.B. Santoso. *Journal of Physics: Conference Series* ***978*** *(2018) 012024 Page 1- 7*	Implementation and evaluation of LMS mobile application: scele mobile based on *user*-centered design	evaluasi *prototype* kepada 14 partisipan yang diminta untuk menjalankan 9 task dan menilai aplikasinya melalui SUS.	Aplikasi SCeLE yang merupakan learning management system (LMS) berbasis moodle.	Dari 14 partisipan yang diuji, total keseluruhan skor rata – rata SUS adalah 71,25 yang dapat dikategorikan baik dari segi *usability*

Tabel 2.1 merupakan penelitian terdahulu. Terdapat 2 macam penelitian dengan penjelasan sebagai berikut:

Penelitian (Subhiyakto & Utomo, 2017) pada Tabel 2.1 menjelaskan tentang analisis dan perancangan aplikasi permodelan kebutuhan perangkat lunak. Analisa dan perancangan tersebut menggunakan metode *prototyping*. Dari hasil penelitian yang didapat adalah *Use Case, class diagram,* dan rancangan tampilan aplikasi. dari penelitian ini metode *prototyping* juga akan digunakan dalam melakukan analisa dan perancangan aplikasi *mobile* Vinandotech, dikarenakan dalam melakukan perancangan aplikasi dibutuhkan komunikasi yang secara terus menerus antara pengguna dan pengembang.

Penelitian (Banimahendra & Santoso, 2017) pada Tabel 2.1 menjelaskan tentang evaluasi *prototype* terhadap aplikasi SCeLE yang merupakan *learning management system* berbasis *moodle*. Metode evaluasi yang digunakan adalah SUS (*System Usability Scale).* Dari hasil evaluasi terhadap 14 partisipan, didapatkan total skor SUS-nya adalah 71,25. Pada penelitian juga akan digunakan SUS sebagai metode untuk melakukan evaluasi terhadap tampilan awal aplikasi Vinandotech.