

BAB II

LANDASAN TEORI

2.1.1 *Android studio*

Firly (2018:13) menjelaskan bahwa *Android studio* merupakan *integrated development environment (IDE)* atau dalam artian lain adalah sebuah lingkungan pengembangan terintegrasi resmi yang memang merancang khusus untuk pengembangan system operasi *Google Android*. Aplikasi ini dibangun di atas sebuah perangkat lunak yang dinamakan *IntelliJ IDEA* milik *JetBrains*. Bisa juga dibilang bahwa *android studio* merupakan pengganti dari *Eclipse android development tool* atau *ADT* sebagai *IDE* utama dalam pengembangan aplikasi *android* yang asli.

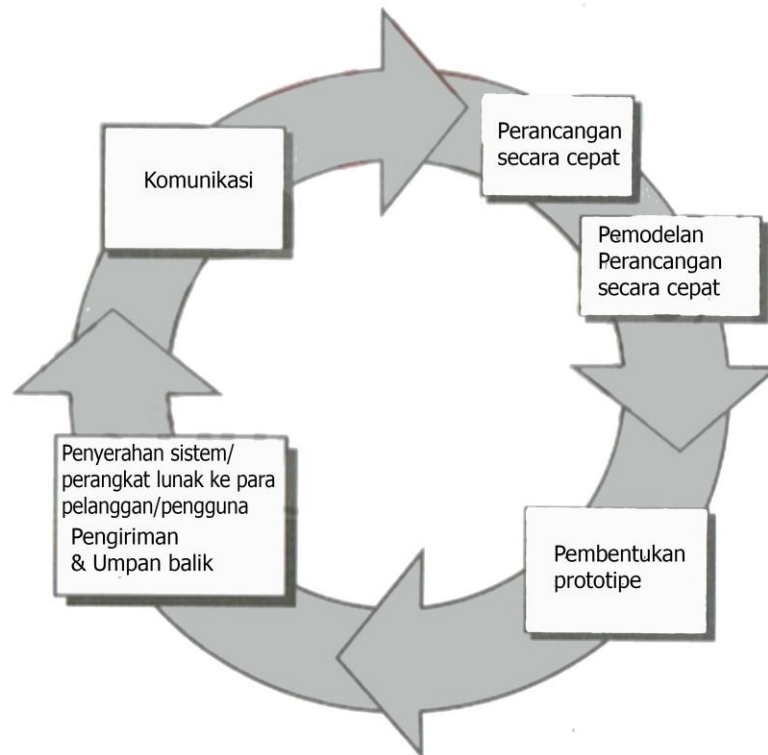
Android studio diluncurkan pada tanggal 16 mei 2013 dalam *konferensi google I/O* yang pada saat itu masih dalam tahap pratinjau akses *versi 0.1* sebagai perintis. Hingga pada akhirnya *versi* stabil *3.0* yang *liris* pada pertengahan bulan oktober 2017 dan menjadi *software* terlaris dikalangan *developer* muda. Aplikasi ini dapat digunakan diberbagai sistem operasi yaitu *windows,linux* dan *macOS*.

Aplikasi ini menawarkan berbagai fitur canggih yang akan meningkatkan kemampuan *produktivitas* dalam proses pengembangan aplikasi. Berikut ini adalah beberapa hal yang akhirnya banyak mengundang *developer* untuk melirik *android studio* sebagai *software* pengembang :

- a. Dukungan dari C++, *NDK* dan sekarang *kotlin*
- b. Perkembangan yang *up to date*
- c. Sistem berbasis *Gradle* yang dinilai fleksibel
- d. Lingkungan yang mencakup seluruh perangkat *android*
- e. *Emulator* yang cepat dan kaya akan fitur
- f. Alat pengujian dan kerangka yang juga *ekstensif*
- g. *Instant Run*
- h. Dukungan *google cloud platform*

2.2 Metode Pengembangan Perangkat Lunak

Pressman (2010:51) menjelaskan meskipun pembuatan *prototype* dapat di gunakan sebagai model proses yang berdiri sendiri, pembuatan protoipe lebih umum digunakan sebagai teknik yang dapat diimplementasikan di dalam konteks setiap model proses perangkat lunak, dan paradigma pembuatan *prototype* seringkali membantu tim pengembang perangkat lunak dan para *stakeholder* untuk memahami lebih baik apa yang akan dikembangkan saat spesifikasi kebutuhan belum jelas.



Gambar 2.1 Tahapan Pengembangan Aplikasi

Berikut ini adalah penjelasan dari gambar 2.1

1. Komunikasi

Tahap komunikasi ini adalah tahapan komunikasi antara *developer* dan pelanggan mengenai tujuan pembuatan *software*, mengidentifikasi apakah kebutuhan diketahui.

2. Perancangan secara cepat

Tahap perancangan secara cepat ini adalah tahapan perancangan cepat setelah terjalin komunikasi.

3. Pemodelan perancangan secara cepat

Tahap pemodelan perancangan secara cepat ini adalah tahapan segera membuat model, dan pemodelan cepat fokus pada gambaran dari segi *software* apakah *visible* menurut pelanggan.

4. Pembentukan prototipe

Tahap pembentukan prototipe ini adalah tahapan pemodelan cepat menuntun pada pembuatan dari prototipe.

5. Penyerahan sistem / perangkat lunak ke para pelanggan / pengguna pengiriman dan umpan balik

Tahap penyerahan sistem / perangkat lunak ke para pelanggan / pengguna pengiriman dan umpan balik tahapan ini adalah prototipe yang dikirimkan kemudian dievaluasi oleh pelanggan, umpan balik digunakan untuk menyaring kebutuhan untuk program.

2.3 Pengujian Kotak Hitam

Pressman (2010:597) menjelaskan bahwa pengujian kotak hitam, juga disebut pengujian perilaku, yang berfokus pada persyaratan fungsional perangkat lunak. Pengujian kotak hitam memungkinkan untuk membuat beberapa kumpulan kondisi masukan yang sepenuhnya akan melakukan semua kebutuhan fungsional untuk program. Pengujian kotak hitam berupaya untuk menemukan kesalahan dalam kategori berikut : (1) fungsi yang salah atau hilang, (2) kesalahan antarmuka, (3) kesalahan dalam struktur data atau akses basis data eksternal, (4) kesalahan perilaku atau kinerja, dan (5) kesalahan inisialisasi dan penghentian.

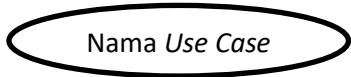
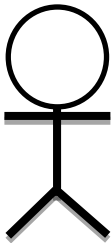
2.4 Unified Modeling Language (UML)




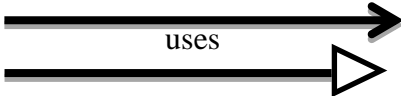
Menurut Mulyani (2016) *Unified Modeling Language (UML)* adalah sebuah Teknik pengembangan sistem yang menggunakan Bahasa garfish sebagai alat untuk pendokumentasian dan melakukan spesifikasi pada sistem. Tujuan Penggunaan UML yaitu untuk memodelkan suatu sistem yang menggunakan konsep berorientasi objek dan menciptakan bahasa pemodelan yang dapat digunakan baik oleh manusia maupun mesin. Menurut Mulyani (2016) tipe-tipe diagram UML adalah sebagai berikut :

2.4.1 Use Case Diagram

Rosa & Shalahuddin (2018:155) menguraikan bahwa *Use Case Diagram* merupakan deskripsi peringkat tinggi bagaimana perangkat lunak (aplikasi) akan digunakan oleh penggunanya. Selanjutnya use case tidak hanya penting pada tahap analisis, tetapi juga sangat penting untuk perancangan, untuk mencari kelas-kelas yang terlibat dalam aplikasi, serta untuk melakukan pengujian. Diagram use case bersifat statis, diagram ini memperlihatkan himpunan use-case dan aktor-aktor (suatu jenis khusus dari kelas). Simbol – simbol *Use Case* dapat dilihat pada tabel 2.1

Tabel 2.1 Komponen *Use Case Diagram*



Simbol	Keterangan
<p><i>Use Case</i></p> 	Fungsionalitas yang disediakan sistem sebagai unit – unit yang saling bertukar pesan antar unit dan aktor.
<p>Aktor / Actor</p> 	Orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat diluar sistem informasi.


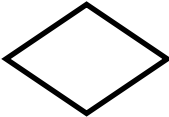

<p>Asosiasi / <i>Association</i></p> 	<p>Komunikasi antar aktor dan <i>Use Case</i> yang berpartisipasi.</p>
<p>Ekstensi / <i>extend</i></p> <p><<extend>></p> 	<p>Relasi <i>Use Case</i> tambahan ke sebuah <i>Use Case</i> dimana <i>Use Case</i> yang ditambah dapat berdiri sendiri walau tanpa <i>Use Case</i> tambahan.</p>
<p>Generalisasi / <i>generalization</i></p> 	<p>Hubungan generalisasi dan spesialisasi antara dua buah <i>Use Case</i> yang mana fungsi yang satu lebih umum dari yang lainnya.</p>
<p>Include / <i>Use Case</i></p> <p><<include>></p> <p>uses</p> 	<p>Relasi <i>Use Case</i> tambahan ke sebuah <i>Use Case</i> dimana <i>Use Case</i> yang ditambahkan memerlukan <i>Use Case</i> ini untuk menjalankan fungsinya.</p>

2.4.2 *Activity Diagram*

Rosa & Shalahuddin (2018:161) menguraikan bahwa *Activity diagram* menggambarkan workflow (alir kerja) atau aktivitas dari sebuah system atau proses bisnis atau menu yang ada pada perangkat lunak. Tahap perancangan *activity diagram* menjabarkan masing – masing *activity* pada perancangan *use case*. Simbol – simbol *Activity Diagram* dapat dilihat pada tabel 2.2.

Tabel 2.2 Komponen *Activity Diagram*

Simbol	Keterangan
<p>Aktivitas</p> 	<p>Aktivitas yang dilakukan sistem, aktivitas biasanya diawali dengan kata kerja.</p>
<p>Status awal</p> 	<p>Bagaimana objek dibentuk atau diawali.</p>


<p>Status akhir</p> 	<p>Bagaimana objek dibentuk dan diakhiri.</p>
<p>Percabangan / join</p> 	<p>Asosiasi percabangan dimana jika ada pilihan aktifitas lebih dari satu.</p>
<p>Penggabungan / join</p> 	<p>Asosiasi penggabungan dimana lebih dari satu aktivitas digabungkan menjadi satu</p>

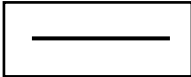


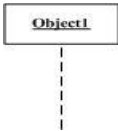
2.4.3 *Sequence Diagram*

Rosa (2018:135) menjelaskan bahwa *Sequence diagram* di UML terutama digunakan untuk memodelkan interaksi antara aktor dan objek dalam sistem dan interaksi antara obyek itu sendiri. UML memiliki sintaks yang kaya untuk sequence diagram, yang memungkinkan berbagai jenis interaksi yang dimodelkan. Sesuai namanya, sequence diagram menunjukkan urutan interaksi yang terjadi antara use case. Sequence diagram memiliki dua buah karakteristik yaitu :

1. Setiap objek memiliki lifeline yang digambarkan dengan garis putus-putus vertikal dan garis ini menunjukkan daur hidup dari sebuah objek.
2. Terdapat fokus kontrol yang digambarkan dengan sebuah persegi panjang yang tipis dan tinggi. Fokus kontrol ini menunjukkan periode waktu selama sebuah objek melakukan sebuah event.

Tabel 2.3 *Komponen Sequence Diagram*

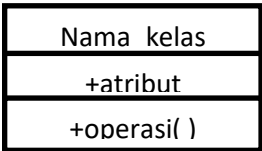
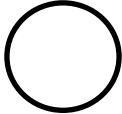

Simbol	Nama Elemen	Keterangan
	<i>Life Line</i>	Objek entitas, antar muka yang saling berinteraksi



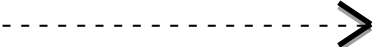

	<i>Activation</i>	Menggambarkan hubungan antar objek dengan <i>message</i>
	<i>Message (call)</i>	Menggambarkan alur message yang merupakan kejadian objek pengirim <i>life line</i> ke objek penerima <i>life line</i>
	<i>Message (return)</i>	Menggambarkan alur pengambilan <i>message</i> ke objek pemanggil dan tanda bahwa objek penerima telah menyelesaikan prosesnya.
	<i>Object</i>	Object adalah instance dari sebuah class yang dituliskan tersusun secara horizontal diikuti lifeline

2.4.4 Class Diagram

Gunawan & Sari (2017:319) menguraikan bahwa *Class diagram* menggambarkan atribut / poperti suatu system, sekaligus menawarkan layanan untuk memanipulasi keadaan metode atau fungsi. *Class diagram* menggambarkan struktur dan deskripsi class, package, dan objek beserta hubungan satu sama lain. Simbol – simbol yang ada pada *Class Diagram* ditunjukkan oleh Tabel 2.3.

Tabel 2.5 Komponen *Class Diagram*

Simbol	Deskripsi
<p>Kelas</p> 	Kelas pada struktur system.
<p>Antarmuka / <i>interface</i></p>  <p>Nama <i>interface</i></p>	Sama dengan konsep <i>interface</i> dalam pemograman berorientasi objek.
<p>Asosisasi / <i>association</i></p> 	Relasi antarkelas dengan makna umum, asosia biasanya disertai dengan <i>multiplicity</i> .

<p>Asosiasi berarah / <i>directed association</i></p> 	<p>Relasi antarkelas dengan makna kelas yang satu digunakan oleh kelas yang lain, asosiasi biasanya juga disertai dengan <i>multiplicity</i>.</p>
<p>generalisasi</p> 	<p>Relasi antarkelas dengan makna Generalisasi – spesialisasi (umum khusus).</p>
<p>Kebergantungan / <i>dependency</i></p> 	<p>Relasi antarkelas dengan makna kebergantungan antarkelas.</p>
<p>Agregasi / <i>aggregation</i></p> 	<p>Relasi antar kelas dengan makna semua – bagian</p>

2.5 Penelitian Terdahulu

Berikut merupakan jurnal terkait dengan penelitian terdahulu :

No	Nama	Judul	Terbit / Tahun	Keterangan
1	Whildan syaihul fikri, Eka whidi Y dan Yogie indra K	Aplikasi Pemesanan Cetak Foto Online (Studi Kasus: Studio FotoXYZ)	Konferensi Nasional ICT-M Politeknik Telkom (KNIP) / 2012	Penelitian ini membahas mengenai Aplikasi pemesanan cetak foto online yang berada di studio foto XYZ, sehingga mampu membuat proses pemesanan cetak foto lebih cepat tanpa pelanggan harus datang ke studio foto dan dapat membantu perusahaan dalam pengolahan data pemesanan cetak foto.
2	Asilah salma, Irfan darmawan & Faisal mufied	Perancangan Aplikasi Callme Berbasis Android menggunakan metode Prototyping (Modul administrasi	e-Proceeding of Engineering / 2017	Penelitian ini membahas mengenai perancangan aplikasi callme berbasis android untuk mempermudah dalam pemesanan makanan dan menghemat biaya costumer karena tidak melalui pihak ketiga. Metode yang digunakan dalam perancangan aplikasi ini adalah metode Prototyping.

		Customer dan Admin)		
3	Surawijaya & Eko budi	Aplikasi Mobile Driver Online Berbasis Android Untuk Perusahaan Rental Kendaraan	Ultima InfoSys / 2017	Penelitian ini membahas mengenai Aplikasi mobile Driver online berbasis android untuk membantu perusahaan rental kendaraan dalam membagi jadwal driver, dapat memantau driver apabila keluar jalur, dan memfasilitasi bagi para pelanggan dalam melakukan pemesanan.
4	Raharjo, Hidayat dan Yudato	Aplikas Penyedia Layanan Jasa Tukang Bangunan	JTET(Jurnal Teknik Elektro Terapan) Vol.6 No.2 Agustus 2017	Penelitian ini membahas mengenai perlunya disediakan aplikasi mobile yang mampu menghubungkan tukang-tukang bangunan dan masyarakat melalui perangkat <i>smartphone</i> , karena sulitnya mencari tenaga tukang bangunan yang berpengalaman dan bisa dipercaya
5	Nuzula	System Pelayanan dan Pemesanan Online Pada Toko Bangunan Sumarno Jaya Depok	Jurnal String Vol.2 No.3 April 2018	Penelitian ini membahas mengenai media internet diharapkan mampu memberikan informasi kepada orang banyak untuk mengetahui keberadaan Toko bangunan sumarno jaya, tujuannya untuk mempermudah para konsumen untuk melakukan pelayanan dan pemesanan secara tidak langsung ke toko untuk berbelanja produk bangunan.