BAB II

TINJAUAN PUSTAKA

2.1. Bimbingan

Bimbingan merupakan salah satu bidang dan program dari pendidikan, dan program ini ditujukan untuk membantu mengoptimalkan perkembangan siswa. Menurut Tolbert, bim-bingan adalah seluruh program atau semua kegiatan dan layanan dalam lembaga pendidikan yang diarahkan pada membantu individu agar mereka dapat menyusun dan melaksanakan rencana serta melakukan penyesuaian diri dalam semua aspek kehidupannya sehari-hari. Bimbingan merupakan layanan khusus yang berbeda dengan bidang pendidikan lainnya.(Fenti Hikmawati., 2016).

2.2 Aplikasi

Aplikasi merupakan program yang dijalankan disuatu pemroses. Aplikasi adalah penerapan, pengimplementasian suatu hal, data, permasalahan, pekerjaan ke dalam suatu sarana atau media yang dapat digunakan untuk menerapkan atau mengimplementasikan hal atau permasalahan nilai-nilai dasar dari hal, data, permasalahan atau pekerjaan (Utari, Mesran, & Silalahi, 2016).

Aplikasi adalah *software* yang dibuat oleh suatu perusahaan komputer untuk mengerjakan tugas-tugas tertentu, misalnya *Ms.World*, *Ms.Excel* (Yuni Arkhiansyah & Rasikun, 2018).

Aplikasi adalah program siap pakai untuk melayani kebutuhan pengguna dalam berbagai aktifitas untuk pengolahan data (Nurhayati, Josi, & Hutagalung, 2017).
Maka dapat disimpulkan aplikasi adalah suatu program yang berisikan perintah-perintah untuk melayani pengguna melakukan aktifitas pengolahan data.

2.3 *Android*

Android merupakan sebuah sistem operasi telepon seluler dan komputer tablet layar sentuh (*touchscreen*) yang berbasis *Linux.* Namun seiring perkembangannya Android berubah menjadi platform yang begitu cepat dalam melakukan inovasi. Hal ini tidak lepas dari pengembang utama dibelakangnya yaitu Google. Google lah yang mengakusisi *Android*, kemudian membuatkan sebuah *platform*. *Platform Android* terdiri dari sistem operasi berbasis *Linux*, sebuah GUI (*Graphic User Interface*), sebuah *web browser* dan aplikasi *end-user* yang dapat di *download* dan juga para pengembang bisa dengan leluasa berkarkaya serta menciptakan aplikasi yang terbaik dan terbuka untuk digunakan oleh berbagai macam perangkat (Kasman, 2015).

2.3.1 Versi *Android*

Android akan terus berusaha memperbarui sistem operasinya agar terus memuaskan kebutuhan pasar global. Kemajuan teknologi saat ini tentunya tidak lepas dari perkembangan teknologi yang semakin hari semakin terbaharui (Nadia Firly, 2019).

Hal tersebut terlihat dari adanya versi demi versi yang terus diluncurkan oleh *Android*. Berikut adalah tabel yang menunjukkan berbagai versi *Android* yang telah dirilis :

Tabel 2.1 Tabel Versi Android

Versi	Nama	Tanggal Rilis
1.0 (API level 1)	-	23 September 2008
1.1 (API level 2)	-	9 Februari 2009
1.5 (API level 3)	*Cupcake*	27 April 2009
1.6 (API level 4)	*Donut*	15 September 2009
2.0 (API level 5)	*Eclair*	26 Oktober 2009
2.0.1 (API level 6)	*Eclair*	3 Desember 2009
2.1 (API level 7)	*Eclair*	12 Januari 2010
2.2-2.2.3 (API level 8)	*Froyo*	20 Mei 2010
2.3-2.3.2 (API level 9)	*Gingerbread*	6 Desember 2010
2.3-3.2.3.7 (API level 10)	*Gingerbread*	9 Februari 2011
3.0 (API level 11)	*Honeycomb*	22 Februari 2011
3.1 (API level 12)	*Honeycomb*	10 Mei 2011

3.2 (API level 13)	*Honeycomb*	15 Juli 2011
4.0-4.0.2 (API level 14)	*Ice Cream Senwich*	19 Oktober 2011
4.0.3-4.0.4 (API level 15)	*Ice Cream Senwich*	16 Desember 2011
4.1 (API level 16)	*Jelly Bean*	27 Juni 2012
4.2 (API level 17)	*Jelly Bean*	29 Oktober 2012
4.3 (API level 18)	*Jelly Bean*	24 Juli 2013
4.4 (API level 19)	*Kit-Kat*	31 Oktober 2013
5.0 (API level 21)	*Lolipop*	12 November 2014
6.0 (API level 22)	*MarshMallow*	5 Oktober 2015
7.0 (API level 23)	*Nougat*	9 Maret 2016
7.1 (API level 24)	*Nougat*	19 Oktober 2016
8.0 (API level 26)	*Oreo*	21 Maret 2017

2.4 Java

Java diciptakan oleh suatu tim yang dipimpin oleh Patrick Naughton dan James Gosling dalam suatu proyek dari *sun microsystem* yang memiliki kode *green* dengan tujuan untuk menghasilkan bahasa komputer sederhana yang dapat dijalankan di peralatan sederhana dengan tidak terikat pada arsitekture tertentu, mulanya disebut oak, tetapi karena oak sendiri merupakan nama dari bahasa pemrograman komputer yang sudah ada, maka *sun* mengubahnya menjadi *java.sun* kemudian meluncurkan *browser* dari *java* yang disebut *hot java* yang mampu menjalankan *applet.* Setelah itu teknologi *java* diadopsi oleh *Netscape* yang memungkinkan program *java* dijalankan di *browser netscape* yang kemudian diikuti *Internet Explore*.Karena keunikan dan kelebihannya, teknologi *java* mulai menarik banyak *vendor* seperti *IBM, Symantec, Inprise*, dll. Sun merilis versi awal *java* secara resmi pada awal tahun (Suyanto, 2015), 1996 yang kemudian terus berkembang hingga muncul *JDK* 1.1 kemudian *JDK* 1.2 yang mulai disebut sebagai versi *java2* karena banyak mengandung peningkatan dan perbaikan. Perubahan utama adalah swing yang merupakan teknologi *GUI (Graphical UserInterface)* yang mampu menghasilkan *windows* yang *portable*. Dan pada tahun 1998-1999 lahirlah teknologi *J2EE (Java 2 Enterprise Edition).*

2.4.1 JDK (*Java Development Kit*)

Java Development Kit atau JDK merupakan sebuah perangkat lunak yang berfungsi dalam proses memanajemenisasi aplikasi java. Hal tersebut dikarenakan Anda akan menggunkan bahasa pemrograman *Java* dalam membuat aplikasi di *Android Studio* (Nadia Firly, 2019).

2.4.2 SDK (*Software Development Kit*)

Software Development Kit atau SDK, merupakan sebuah kit yang berfungsi untuk mengembangkan berbagai aplikasi berbasis *Android* oleh para developer. Di dalam SDK telah terdapat berbagai *tools* yang bertujuan untuk proses pengembangan aplikasi seperti proses *debugger, software libraries, emulator*, dokumentasi, *sample cod* (Nadia Firly, 2019).

2.5 XML (*Extensible Markup Language*)

XML (Extensible Markup Language) di manfaatkan dalam mendefinisikan dokumen dengan format standar dimana yang dapat dibaca dan di dukung oleh aplikasi-aplikasi xml yang kompatibel. Bahasa format XML bisa digunakan dengan halaman html, akan tetapi XML itu sendiri bukan bahasa *markup*. Sebaliknya, XML itu merupakan “metabahasa” yang dapat di pakai dalam membuat bahasa markup untuk aplikasi khusus.Sebagai contoh nya itu, dapat menggambarkan item yang bisa diakses di saat membutuhkan halaman web dimana pada dasarnya, XML ini dapat memungkinkan Anda untuk membuat database informasi tanpa memiliki *database* yang sebenarnya. Meskipun secara default hanya digunakan dalam aplikasi web, banyak program lainnya juga yang dapat menggunakan dokumen XML, misalnya kode sumber aplikasi *Android*. Mungkin jelasnya pengertian dari XML *(Extensible Markup Language)* adalah bahasa markup untuk keperluan umum yang telah disarankan oleh w3c dalam hal membuat dokumen markup untuk kepentingan pertukaran data antar sistem yang beraneka ragam.Tepatnya xml yaitu kelanjutan dari *HTML (HyperText Markup*

Language) dimana yang merupakan bahasa standar untuk melacak Internet. Jadi, untuk membaca bahasa markup ada kesinambungannya. Baca dan cari tahulah selengkap-lengkapnya XML justru di desain untuk mempu menyimpan data secara lengkap, ringkas serta mudah dalam mengatur. Kata kunci utama dari XML ini adalah data (jamak dari datum) apabila jika diolah bisa memberikan informasi. XML juga menyediakan suatu cara terstandarisasi namun dapat dimodifikasi untuk menggambarkan isi dari dokumen. Dengan sendirinya, XML dapat digunakan dalam menggambarkan sembarang *view database*, akan tetapi hanya dengan suatu cara yang standar (Wardhani, 2016).

2.6 Android Studio

Android studio merupakan *Integrated Development Environment* (IDE) atau dalam artian lain adalah sebuah lingkungan pengembang terintegrasi resmi yang memang dirancang khusus untuk pengembangan sistem operasi *Google Android.* Aplikasi yang satu ini, dibangun di atas sebuah perangkat lunak yang dinamakan *IntelliJ IDEA* milik *JetBrains.* Bisa dibilang juga bahwa *Android Studio* merupakan pengganti dari *Eclipse Android Development Tools* atau ADT sebagai IDE utama dalam pengembangan aplikasi *Android* yang asli. Tentunya *Android* merupakan *platfrom open source* yang bebas untuk dikembangkan oleh siapapun (Nadia Firly, 2019).

2.7 *Smartphone*

Gadget adalah perangkat alat elektronik kecil yang memiliki banyak fungsi. *Gadget* (*smartphone*) memiliki banyak fungsi bagi penggunanya sehingga dinilai lebih memudahkan. *Gadget* (*smartphone*) atau dengan kata sederhana telphone gengam yang saat ini telah memiliki beragam fiture dan fungsi yang semakin kompleks guna memudahkan pemakainya merupakan trobosan baru dari telephone gengam sebelumnya. (Garini dalam Rohman, 2017).

2.8 Firebase

Firebase adalah *Cloud Service Provider* dan *Backend as a Service* yang dimiliki oleh Google. *Firebase* merupakan solusi yang ditawarkan oleh Google untuk mempermudah dalam pengembangan aplikasi *mobile* maupun web dan bersifat *Realtime Database. Firebase* memiliki banyak SDK yang memungkinkan untuk mengintergrasikan layanan ini dengan *Android, IOS, Javascript, C++* hingga *Unity*. Pada penggunaan *Firebase* diperlukan akses internet dalam menjalankan aplikasi tersebut. Dikarenakan data yang tersimpan pada tempat penyimpanan berbasis *Online* (Nadia Firly, 2019).

2.8.1 Fitur Firebase

Setelah API *Firebase* dimasukkan ke Dalam aplikasi *Android* atau iOS, pengembang bisa menggunakan fitur *Firebase* dengan coding yang simpel. Beberapa fitur yang disediakan *Firebase*, yaitu :

1. *Analytics*

   Fitur ini membuat para pengembang aplikasi dapat mengerti bagaimana para pengguna menggunakan aplikasi mereka. SDK *analiytics* menangkap sendiri data yang dibutuhkan oleh para *developer*. Dashboard juga menyediakan detil seperti pengguna paling aktif atau fitur apa saja yang paling digunakan dalam aplikasi tersebut. *Analytic*s juga menyediakan data yang telah dirangkum.

2. *Authentication*

   Fitur ini membuat para pengembang dapat mengijinkan pengguna untuk mengakses aplikasi. *Firebase* menyediakan fitur login melalui Gmail, Github, Twitter, Facebook dan juga autentikasi buatan sendiri.

3. *Messaging*

   FCM (*Firebase Cloud Messaging*) membuat para pengguna dapat mengirimkan pesan ke berbagai platform tanpa biaya tambahan. *Messaging* juga dapat digunakan untuk kebutuhan notifikasi.

4. *Real-time Database*

   Database di *Firebase* adalah database berbasis *cloud* dan tidak membutuhkan SQL untuk mengambil dan menyimpan data atau bisa disebut juga NoSQL. Database ini sangatlah cepat dan dapat diandalkan yang artinya data dapat dibaharui dan disinkronisasikan dengan cepat. Data juga dijaga meskipun pengguna kehilangan koneksi internetnya.

5. *Storage*

   Firebase juga menyediakan fasilitas penyimpanan. *Firebase* dapat menyimpan dan mengambil konten seperti gambar, video, dan audio langsung dari SDK. Meng-*upload* dan men-*download* juga dilakukan di *background*. Data yang disimpan akan aman dan hanya pengguna yang diijinkan yang dapat mengaksesnya.

6. *Hosting*

   Firebase juga menyediakan fitur *hosting*. *Firebase* mengirimkan konten web secara cepat dan konten selalu dikirim dengan aman.

7. *Crash reporting*

   Fitur *crash reporting* di *Firebase* membuat pengembang dapat mengetahui kesalahan ketika terjadi *crash*.

2.9 *Blackbox Testing*

Black box testing adalah *tipe testing* yang memerlukan perangkat lunak yang tidak diketahui kinerja internalnya. Sehingga para tester memandang perangkat lunak seperti layaknya sebuah “ kotak hitam” yang tidak penting dilihat isinya, tapi dikenal proses testing dibagian luar. Metode ujicoba *blackbox* memfokuskan pada keperluan fungsional dari software. Karena itu ujicoba *blackbox* memungkinkan pengembang *software* untuk membuat himpunan kondisi input yang akan melatih seluruh syarat-syarat fungsional suatu program (Sari, 2016).

2.9 UML (*Unifed Modeling Language*)

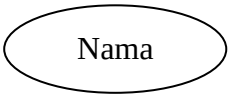
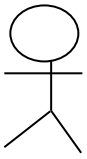
UML (*Unified Modeling Language*) adalah salah standar bahasa yang banyak digunakan di dunia industri untuk mendefinisikan *requirement*, membuat analisis & desain, serta menggambarkan arsitektur dalam pemrograman berorientasi objek. Dapat didefinisikan UML adalah standart bahasa untuk mendefinisikan dari *requirement*, membuat analisis & desain dan menggambarkan arsitektur dalam pemrograman yang berorientasi pada objek (Ariansyah, Fajriyah & Febby Satryadi Prasetyo, 2017).

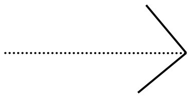
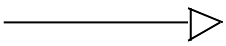
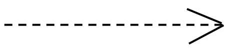
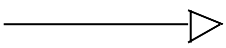
2.9.1 *Use Case* Diagram

Use case Diagram menggambarkan *external view* dari sistem yang akan *user* buat modelnya. Model *use case* dapat dijabarkan dalam diagram *use case*, tetapi perlu diingat, diagram tidak indetik dengan model karena model lebih luasdari diagram. *Use case* harus mampu mengambarkan urutan aktor yang menghasilkan nilai terukur (Prabowo Pudjo Widodo, dalam Suendri, 2018).

Use Case Diagram adalah sesuatu atau proses merepresentasikan hal-hal yang dapat dilakukan oleh aktor dalam menyelesaikan sebuah pekerjaan.” (Mamed Rofendy Manalu, 2015). Berikut adalah simbol-simbol yang ada pada *use case* diagram :

Tabel 2.2 Tabel Simbol-simbol *Use case Diagram*

Nama Simbol	Simbol	Deskripsi
Use Case		Fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar unit atau aktor, biasanya dinyatakan dengan menggunakan kata kerja di awal *frase* nama *use case*.
Aktor / *actor* 	Nama *actor*	Orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat diluar sistem informasi yang akan

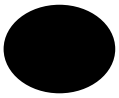
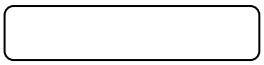
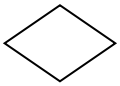
		dibuat itu sendiri, jadi walau pun simbol dari aktor adalah gambar orang, tapi aktor belum tentu merupakan orang, biasanya dinyatakan. menggunakan kata benda diawali *frase* nama actor.
Asosiasi / *association*		Komunikasi antar aktor dan *use case* yang berpartisipasi pada *use case* atau *use case* memiliki interaksi dengan aktor.
Ekstensi / *extend*	<<extend>> 	*Case* tambahan memiliki nama depan yangsama dengan *use case* yang ditambahkan. Arah panah mengarah pada *use case* yang ditambahkan.
Generalisasi/ *generalization*		Hubungan generalisasi dengan spesialisasi antara dua buah *use case* dimana fungsi yang satu adalah fungsi yang lebih umum dari yang lainnya. Arah panah mengarah pada *use case* yang menjadi generalisasinya (umum)
Menggunakan *include / uses*	<<include>>  <<uses>> 	Fungsinya atau sebagai syarat dijalankan *use case* ini Ada dua sudut pandang yang cukup besar mengenai include di *use case* : • *Include* berarti *use case* yang ditambahkan akan selalu dipanggil saat *use case* tambahan dijalankan • *Include* berarti *use case* yang tambahan akan selalu melakukan pengecekan apakah *use case* yang

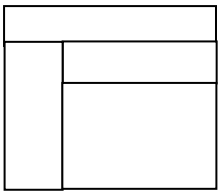
		ditambahkan telah dijalankan sebelum *use case* tambahan dijalankan Kedua interpretasi di atas dapat dianut salah satu atau keduanya tergantung pada pertimbangan dan interpretasi yang dibutuhkan

2.9.2 *Activity Diagram*

Diagram *activity* menunjukkan aktivitas sistem dalam bentuk kumpulan aksi-aksi, bagaimana masing-masing aksi tersebut dimulai, keputusanyang mungkin terjadi hingga berakhirnya aksi. *Activity* diagram juga dapat menggambarkan proses lebih dari satu aksi salam waktu bersamaan.“ Diagram *Activity* adalah aktifitas-aktifitas, objek, *state*, transisi *state* dan *event*. Dengan kata lain kegiatan diagram alur kerja menggambarkan perilaku sistem untuk aktivitas” (Haviluddin dalam Suendri, 2018).

Tabel 2.3 Tabel Simbol-Simbol Diagram *Activity*

Nama	Simbol	Deskripsi
Status awal		Status awal aktivitas sistem, sebuah diagram aktivitas memiliki sebuah status awal.
Aktivitas		Aktivitas yang dilakukan sistem, aktivitas biasanya diawali dengan kata kerja.
Percabangan/ *Decision*		Asosiasi percabangan dimana jika ada pilihan aktivitas lebih dari satu.
Penggabungan/ *Join*		Asosiasi penggabungan dimana lebih dari satu aktivitas digabungkan menjadi satu.
Status Akhir		Status akhir yang dilakukan oleh sistem, sebuah diagram

		aktivitas memiliki sebuah status akhir.
		Memisahkan organisasi bisnis yang bertanggunga jawab terhadap aktivitas yang terjadi.