BAB II

LANDASAN TEORI

2.1 Informasi

(TM. Zaini dan Layanan Destasari, 2010) Informasi adalah data yang telah diolah, atau data yang telah memiliki arti, sedangkan data adalah terdiri dari sejumlah fakta dan angka-angka yang *relative* tidak berarti bagi pemakainya. Perubahan data menjadi informasi dilakukan oleh pengolah informasi (*information processor*). Pengolahan informasi adalah salah satu elemen kunci dalam sistem informasi.

2.2 Notifikasi

(Kamus Besar Bahasa Indonesia, 2010 : 168) menguraikan Pengertian notifikasi berarti pemberitahuan atau kabar. Notifikasi dalam penelitian ini dijabarkan untuk mengetahui jika adanya pemesanan dan pembayaran.

2.3 Aplikasi Mobile

(Mobile Marketing Association, 2015) Aplikasi *Mobile* adalah perangkat lunak yang berjalan pada perangkat *mobile* seperti *smartphone* atau tablet PC. Aplikasi *Mobile* juga dikenal sebagai aplikasi yang dapat diunduh dan memiliki fungsi tertentu sehingga menambah fungsionalitas dari perangkat *mobile* itu sendiri. Untuk mendapatkan *mobile application* yang diinginkan, *user* dapat mengunduhnya melalui situs tertentu sesuai dengan sistem operasi yang dimiliki. *Google Play* dan *iTunes* merupakan beberapa contoh dari situs yang menyediakan beragam aplikasi bagi pengguna Android dan iOS untuk mengunduh aplikasi yang diinginkan.

2.4 Android

(Gunawan, H., & Sari, Y. P. 2017, November) Android adalah sebuah sistem operasi untuk *smartphone* dan tablet. Sistem operasi dapat diilustrasikan sebagai ’jembatan’ antara piranti (*device*) dan penggunanya, sehingga pengguna bisa berinteraksi dengan *device*-nya dan menjalankan aplikasi-aplikasi yang tersedia pada *device*.

Android merupakan sistem operasi untuk perangkat *mobile* yang berbasis *Linux* dan bersifat terbuka atau *open source* dengan lisensi GNU yang dimiliki *Google*.

1. **Versi Android**

   Adapun versi-versi android yang pernah dirilis adalah sebagai berikut :

   1. Android Versi 1.0 Alpha - Dirilis pada 23 September 2008.
   2. Android Versi 1.1 Bender (Beta) - Dirilis pada 9 Februari 2009.
   3. Android Versi 1.5 CupCake - Dirilis pada 27 April 2009.
   4. Android Versi 1.6 Donut - Dirilis pada 15 September 2009.
   5. Android Versi 2.0 - 2.1 Eclair - Dirilis pada 26 Oktober 2009.
   6. Android Versi 2.2 Frozen Yoghurt - Dirilis pada 10 Mei 2010.
   7. Android Versi 2.3 GingerBread - Dirilis pada 6 Desember 2010.
   8. Android Versi 3.0-3.2 HoneyComb - Dirilis pada 22 Pebruari 2011.
   9. Android Versi 4.0 Ice Cream Sandwich - Dirilis pada 19 Oktober 2011.
   10. Android Versi 4.1 - 4.3 Jeally Bean - Dirilis pada 27 Juni 2012.
   11. Android Versi 4.4 KitKat - Dirilis pada 31 Oktober 2013.
   12. Android Versi 5.0 Lollipop - Dirilis pada 5 Juni 2014.
   13. Android Versi 6.0 Marshmallow - Dirilis pada 17 Agustus 2015.
   14. Android Versi 7.0 Nougat - Dirilis pada 18 Juli 2016.
   15. Android Versi 8.0 Oreo - Dirilis pada 21 Agustus 2017.

2. **Statistik Distribusi OS Android**

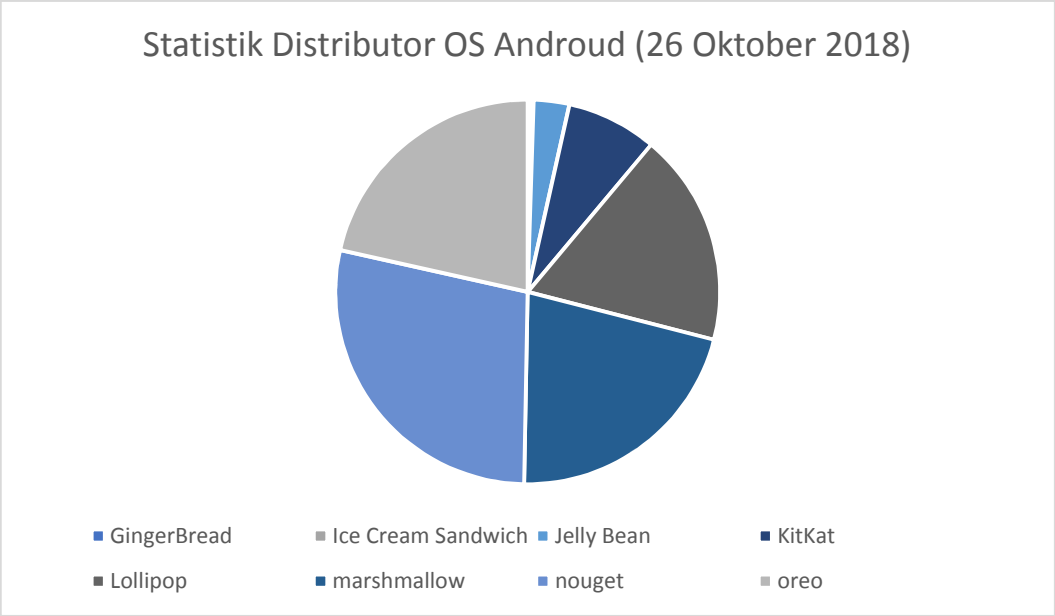
   (https://developer.android.com/about/dashboards/) Google baru-baru ini merilis angka statistik distribusi sistem operasi Android di berbagai perangkat mobile di seluruh dunia pada 26 Oktober 2018. Dalam statistik tersebut terlihat Android 6.0 *Marshmallow* menjadi yang tertinggi dengan persentase total 21.3%. Di urutan kedua berhasil dikuasai oleh Android 7.0

Nougat yang memiliki *persentase* mencapai 18.1%. Kedua sistem operasi ini memang layak menduduki angka tertinggi karena saat ini smartphone Android memang masih banyak yang menggunakan Android *Marshmallow* dan Nougat mengingat vendor-vendor di seluruh dunia masih mempercayakan kedua sistem operasi ini pada smartphone buatannya. Sementara pada urutan ketiga diduduki Android5.1 *Lollipop* dengan *persentase* 14.4% yang dususul oleh Android 8.0 *Oreo* dengan presentase 14.0 %. kedua sistem operasi ini masih banyak digunakan pada smartphone *entry-level* di berbagai negara. Selanjutnya, Android 7.1 *Nougat* dengan persentase total 10.1%. Selanjutnya, Andorid 4.4 *Kitkat* dengan presentase 7.6%. Selanjtnya Android 8.1 *Oreo* dengan *presentase* 7.5% dan android versi 5.0 *Lollipop* sudah mulai ditinggalkan pengguna Android karena hanya memiliki persentase 3.5% dam umtuk urutan di bawah nya yaitu oleh android dengan versi 4.1.x, 4.2.x dan 4.3 *Jelly Bean* dengan total 3.0% berikutnya android versi 4.0.3 dan 4.0.4 *Ice cream sandwich* dengan *presentase* 0.3 % di urutan terakhir yaitu android versi 2.3.3-2.3.7 *Ginger bread* dengan *presentase* 0.2% Hal ini tentu wajar karena *Google* sendiri sudah menghentikan dukungan untuk sistem operasi *Ginger bread* ini pada Q1 2017. Sementara versi *Ice Cream Sandwich* yang meski terus melempem tetapi masih akan didukung dalam waktu lebih lama.

Berikut data statistik pengguna sistem operasi android yang dikumpulkan selama periode 7 hari yang berakhir pada 26 Oktober 2018, Setiap versi dengan distribusi kurang dari 0,1% tidak ditampilkan.

Version	Kode Name	Distribution
2.3.3- 2.3.7	GingerBread	0,2%
4.0.3- 4.0.4	Ice Cream Sandwich	0,3%
4.1.x	Jelly Bean	1,1%
4.2.x		1,5%

4.3		0,4%
4.4	KitKat	7,6%
5.0	Lollipop	3,5%
5.1		14,4%
6.0	Marshmallow	21,3%
7.0	Nougat	18,1%
7.1		10,1%
8.0	Oreo	14.0%
8.1		7.5%



Gambar 2.1 Statistik distribusi OS Android

2.5 Elemen Android

1. *Dalvik Virtual Machine* (DVM)

Salah satu element kunci dari Android adalah *Dalvik Virtual Machine* (DVM). Android berjalan di *Dalvik Virtual Machine* (DVM) bukan di *Java Virtual Machine* (JVM), sebenarnya banyak persamaannya dengan *Java Virtual Machine* (JVM) seperti *Java Mobile Edition* (JME), tetapi Android

menggunakan *virtual machine* sendiri yang dirancang untuk memastikan beberapa fitur-fitur berjalan lebih efisien pada perangkat *mobile*.

2 Android SDK (*Software Development Kit*)

Android SDK adalah tools API (*Application Programing Interface*) yang diperlukan untuk mulai mengembangkan aplikasi pada *platform* Android menggunakan bahasa pemrograman *Java*. Android merupakan subset perangkat lunak untuk ponsel yang meliputi sistem operasi *middleware* dan aplikasi kunci yang di-*release* oleh *Google*. Saat ini disediakan Android SDK (*Software Development Kit*) sebagai alat bantu dan API untuk mulai mengembangkan aplikasi pada *platform* Android menggunakan bahasa pemrograman *Java*. Sebagai *platform* aplikasi-netral, Android memberi kesempatan untuk membuat aplikasi yang dibutuhkan.

3 ADT (*Android Development Tools*)

Android development tools adalah *plugin* yang di desain untuk IDE Android Studio yang memberikan kemudahan dalam mengembangkan aplikasi Android dengan menggunakan IDE Android Studio. Dengan menggunakan ADT untuk Android Studio akan memudahkan dalam membuat aplikasi *project* Android, membuat GUI aplikasi, dan menambahkan komponen-komponen yang lainnya.

2.6 Firebase

Firebase adalah layanan pihak ketiga, *Firebase* bisa dikatakan sebagai layanan DbaaS (*Database as a Service*) dengan konsep *realtime*. Tidak hanya berfungsi sebagai tempat penyimpannan data, tapi juga di sediakan API untuk implementasi *web socket*. *Firebase* menyediakan *library* untuk berbagai *client*

platform. *Browser* menggunakan *Javascript* API dan *mobile* menggunakan OBJ-C atau Android API.

Berikut penjelasan beberapa fitur yang tersedia pada firebase :

1. ***Firebase Cloud Messaging* (FCM)**

   Firebase Cloud Messaging untuk Android (FCM) adalah layanan yang membantu pengembang mengirim data dari server untuk aplikasi mereka Android pada perangkat Android. Ini bisa menjadi pesan ringan memberitahu aplikasi Android bahwa ada data baru yang akan diambil dari server (misalnya, film diunggah oleh seorang teman), atau bisa juga pesan yang berisi sampai dengan 4KB data *payload* (sehingga aplikasi seperti *instant messaging* dapat mengkonsumsi pesan langsung). Layanan FCM menangani semua aspek antrian pesan dan pengiriman ke aplikasi target Android berjalan pada perangkat target.

2. ***Firebase Authentication***

   Firebase Authentication menyediakan layanan *backend*, SDK yang mudah digunakan, dan *library* UI yang siap pakai untuk mengautentikasi pengguna ke aplikasi Anda. *Firebase Authentication* mendukung autentikasi menggunakan sandi, nomor telepon, penyedia identitas gabungan yang populer, seperti *Google, Facebook, dan Twitter*, dan lain-lain.

3. ***Firebase Storage***

   Cloud Storage untuk *Firebase* adalah layanan penyimpanan objek yang andal, sederhana, dan hemat biaya yang dibuat untuk skala *Google*. *Firebase* SDK untuk *Cloud Storage* menambahkan keamanan *Google* pada *upload* dan *download file* untuk aplikasi *Firebase* Anda, bagaimana pun kualitas jaringannya. Anda dapat menggunakan SDK kami untuk menyimpan gambar, audio, video, atau konten buatan pengguna lainnya. Di server, Anda dapat menggunakan *Google Cloud Storage* untuk mengakses *file* yang sama.

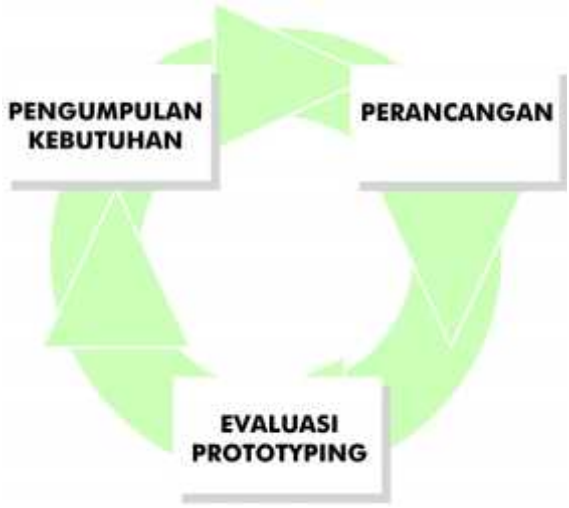
4. ***Firebase Realtime Database***

Firebase Realtime Database adalah *database* yang di-*host* di *cloud*. Data disimpan sebagai JSON dan disinkronkan secara *realtime* ke setiap klien yang terhubung. Ketika Anda membuat aplikasi lintas-*platform* dengan SDK Android, iOS, dan *JavaScript*, semua klien akan berbagi sebuah *instance Realtime Database* dan menerima *update* data terbaru secara otomatis.

2.7 Metode Pengembangan Perangkat Lunak

2.7.1 Metode Prototype

(Yuni Puspita Sari, 2016) *Prototyping* pardigma dimulai dengan pengumpulan kebutuhan, pengembang bertemu dengan pengguna dan mengidentifikasikan objektif keseluruhan dari perangkat lunak, selanjutnya mengidentifikasi segala kebutuhan yang diketahui secara garis besar di mana definisi-definisi lebih jauh merupakan keharusan, kemudian dilakukan perancangan kilat, lalu diakhiri dengan evaluasi *prototyping*, yang dapat dilihat pada gambar berikut ini :



Gambar 2.2 Model *prototype*

Model yang digunakan dalam pengembangan sistem pada penelitian ini adalah dengan menerap kan model *prototype* untuk merancang aplikasi **Perangkat Lunak Informasi Servis Berkala Menggunakan Push Notification Pada Astra Honda Authorized Service Station (AHASS) Tunas Dwipa Matra Berbasis**

Android. Tahap-tahap yang dilakukan dalam pengembangan sistem ini adalah sebagai berikut :

1) Pengumpulan kebutuhan

   Developer dan klien bertemu untuk menentukan tujuan umum, kebutuhan yang diketahui dan gambaran bagian-bagian yang akan dibutuhkan berikutnya Detail kebutuhan mungkin tidak dibicarakan disini, pada awal pengumpulan kebutuhan. Selanjutnya peneliti akan melakukan analisis terhadap data apa saja yang dibutuhkan, seperti analisis terhadap sistem yang berjalan, analisis kebutuhan perangkat lunak, analisis kebutuhan perangkat keras, dan analisis kebutuhan notofikasi imunisasi.

2) Perancangan

   Perancangan dilakukan dengan cepat dan rancangan mewakili semua aspek *software* yang diketahui, dan rancangan ini menjadi dasar pembuatan *prototype*. Dalam tahap ini peneliti akan membangun sebuah versi *prototype* yang dirancang kembali dimana masalah-masalah tersebut diselesaikan.

3) Evaluasi *prototype*

   Pada tahap ini, calon pengguna mengevaluasi *prototype* yang dibuat dan digunakan untuk memperjelas kebutuhan *software*. *Software* yang sudah jadi dijalankan dan akan dilakukan perbaikan apabila kurang memuaskan. Perbaikan termasuk dalam memperbaiki kesalahan/ kerusakan yang tidak ditemukan pada langkah sebelumnya.

Kelebihan dari *Prototype Model* adalah sebagai berikut :

1) *End user* dapat berpartisipasi aktif.
2) Penentuan kebutuhan lebih mudah diwujudkan.
3) Mempersingkat waktu pengembangan *software*.

Kekurangan dari *Prototype Model* adalah sebagai berikut:

1) Proses analisis dan perancangan terlalu singkat.
2) Mengesampingkan alternatif pemecahan masalah.

3) Biasanya kurang fleksibel dalam menghadapi perubahan.
4) *Prototype* yang dihasilkan tidak selamanya mudah dirubah.
5) *Prototype* terlalu cepat selesai.

2.8 Perancangan

Perancangan merupakan tahapan yang dilakukan untuk memulai pembangunan sistem dimana disesuaikan dengan identifikasi pengumpulan kebutuhan yang telah dilakuakan peneliti. Proses perancangan dimulai dari perancangan sistem yang telah diusulkan kemudian dilanjutkan dengan pembuatan perangkat lunak dimana berupa *Unified Modeling Language* (UML), Struktur *Database,* dan perancangan Antarmuka (*Interface*) sistem.

2.8.1 *Unified Modeling Language* (UML)

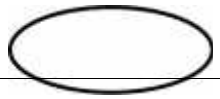
(Sri karnila, 2015) *Unified Modelling Language* (UML) adalah proses penggambaran informasi-informasi dengan notasi-notasi baku yang telah disepakati sebelumnya .

Diagram UML terdapat 7 diagram yaitu :

1. ***Use Case Diagram***

Bersifat statis. Diagram ini memperlihatkan himpunan *use case* dan aktor-aktor (suatu jenis khusus dari kelas). Diagram ini terutama sangat penting untuk mengorganisasi dan memodelkan perilaku dari suatu sistem yang dibutuhkan.

Tabel 2.1. Simbol *Use Case Diagram.*

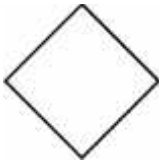
Simbol	Keterangan
	*Actor* : Seseorang atau sesuatu yang berinteraksi dengan sistem yang sedang dikembangkan.
	*Use case* : perungkat tertinggi dari fungsionalitas

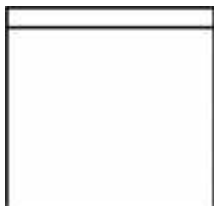
	yang dimiliki sistem.
	*Association :*adalah relasi antara actor dan *use case.*
	*Generalisasi:*untuk memperlihatkan struktur pewaris yang terjadi.

2. ***Activity Diagram***

Bersifat dinamis. Diagram aktivitas ini adalah tipe khusus dari diagram state yang memperlihatkan aliran dari suatu aktifitas ke aktifitas lainnya dalam suatu sistem. Diagram ini terutama penting dalam pemodelan fungsifungsi dalam suatu sistem dan memberi tekanan pada kendali antar objek

Tabel 2.2. Simbol *Activity Diagram.*

Simbol	Keterangan
	*Activity* : Memperlihatkan bagaimana masing-masing kelas antarmuka saling berinteraksi satu sama lain.
	*Initial Node* : Bagaimana objek dibentuk atau diawali
	*Actifity Final Node* : Bagaimana objek dibentuk dan diakhiri.
	*Decision* : Asosiasi percabangan dimana jika ada pilihan aktifitas lebih dari satu.

	*Swimlane* : Memisalkan organisasi bisnis yang bertanggung jawab terhadap aktifitas yang terjadi.
	*Join* : Digunakan untuk menunjukkan kegiatan yang digabungkan.
	*Fork* : Digunakan untuk menunjukkan kegiatan yang dilakukan secara paralel

3. ***Sequence Diagram***

Bersifat dinamis. Diagram urutan adalah interaksi yang menekankan pada pengiriman pesan (*message*) dalam suatu waktu tertentu

4. ***Class Diagram***

Diagram kelas bersifat statis. Diagram ini memperlihatkan himpunan kelas-kelas, antarmuka, kolaborasi-kolaborasi, serta relasirelasi. Diagram ini umum dijumpai pada pemodelan sistem berorientasi objek

5. ***Object Diagram***

Bersifat statis. Diagram ini mempelihatkan objek-objek serta relasi-relasi antar objek. Diagma objek memperlihatkan instansiasi statis dari segala sesuatu yang dijumpai pada diagram kelas.

6. ***Component Diagram***

Bersifat statis. Diagram komponen ini memperlihatkan organisasi serta kebergantungan sistem/perangkat lunak pada komponen-komponen yang telah

ada sebelumnya. Diagram ini berhubungan dengan diagram kelas dimana komponen secara tipikal dipetakan ke dalam suatu satu atau lebih kelaskelas, antarmuka-antarmuka (*interface*) serta kolaborasi-kolaborasinya.

7. ***Deployment Diagram***

Bersifat statis. Diagram ini memperlihatkan konfigurasi saat aplikasi dijalankan saat run time. Diagram ini membuat simpul-simpul (*node*) beserta komponen-komponen yang ada di dalamnya. *Deployment* diagram berhubungan erat dengan diagram komponen dimana *deployment* diagram memuat satu atau lebih komponen-komponen. Diagram ini sangat berguna saat aplikasi kita berlaku sebagai aplikasi yang dijalankan pada banyak mesin (*distributed computing*).

2.9 Pengujian *Black Box Testing*

(Pressman, 2012) *Black-Box Testing* atau Pengujian Kotak Hitam atau juga disebut *Behavioral Testing*, berfokus pada persyaratan fungsional dari perangkat lunak. Artinya, teknik *Black-Box Testing* memungkinkan untuk mendapatkan *set* kondisi masukan yang sepenuhnya akan melaksanakan semua persyaratan fungsional untuk suatu program.

Black-Box Testing bukan merupakan *alternatif* dari pengujian *White-BoxTesting*. Sebaliknya, *Black-Box Testing* adalah pendekatan komplementer yang mungkin untuk mengungkap kelas yang berbeda dari kesalahan daripada metode *White-Box Testing*.

Black-BoxTesting mencoba untuk menemukan kesalahan dalam kategori berikut :

a. Fungsi tidak benar atau hilang.
b. Kesalahan *interface* atau antarmuka.
c. Kesalahan dalam struktur data atau akses *database* eksternal.
d. Kesalahan kinerja atau perilaku.
e. Kesalahan inisialisasi dan terminasi.

2.10 Penelitian Terkait

Tabel 2.3 berikut ini adalah beberapa penelitian terdahulu yang berkaitan dengan media pembelajaran.

Tabel 2.3 Penelitian Terkait

No.	Nama	Judul	Terbit	Uraian
1.	Febrian Prayoga, S.Kom	Perancangan Prototype Aplikasi Pengumuman Kelas Menggunakan Teknologi Firebase Cloud Message pada Android.	Universitas kristen satya wacana, jawa tengah September 2016	Aplikasi ini dibangun dengan menggunakan metodologi waterfall. Aplikasi berbasis android ini menggunakan fitur FCM, sebagai media untuk pengiriman pesan push notification.
2.	Ali Mintarko, Muhammad Lutfi, Eka Puji Widiyanto, S.T., M.Kom	Rancang Bangun Aplikasi Chatroom Berbasis Android dengan Pemanfaatan Google Cloud Messaging sebagai sarana pengiriman pesan.	STMIK GI MDP, Palembang, Maret 2015	pemanfaatan cloud computing sebagai tempat penyimpanan, pengguna dapat menambah kapasitas penyimpanan data tanpa harus membeli peralatan tambahan seperti harddisk.
3.	Atep Aulia Rahman., S.T., M.KOM. dan Nadya Mahardika Citra Pertiwi.,	Sistem Informasi Prakerin Dengan Metode Push Notification Pada Bagian Hubungan Industri (Hubin) Di Smk Mathla'ul	STMIK LPKIA BANDUNG, BANDUNG September 2017	Dibuatnya Prototype aplikasi web berbasis YII untuk membantu siswa mendapatkan

	S.Si., MOS., MTCNA.	Anwar Kopo Bandung		informasi dan konfirmasi secara cepat dan langsung dengan dibantu fiture web notification (Push Notification). Prototype aplikasi web ini menyediakan laporan hasil akhir rekapitulasi per-periode Prakerin