

## **BAB 2**

### **TINJAUAN PUSTAKA DAN DASAR TEORI**

#### **2.1 Tinjauan Pustaka**

Penelitian terkait rancangan Aplikasi Pendaftaran Antrian Online Untuk Pelanggan IndiHome di Plasa Telkom Berbasis Android terdapat pada 3 jurnal yang memiliki ruang lingkup yang sama antara lain :

1. Teknik Telekomunikasi, Universitas Telkom, Jurnal yang ditulis Sukma Bahrul Aziz, Tengku A Riza, Rohmat Tulloh, pada tahun 2016 dengan judul “Perancangan dan Implementasi Aplikasi Sistem Antrian Untuk Pasien Pada Dokter Umum Berbasis Android dan SMS Gateway”. Rancangan antrian online menggunakan android dan SMS Gateway untuk pelanggan yang tidak mempunyai android. Penelitian yang dilakukan hanya dapat digunakan oleh pasien, dan fokus hanya pada fitur pemesanan antrian. Pada dashboard hanya menampilkan nomor antrian pasien, nomor yang sedang dilayani dan tidak memiliki pilihan jenis antrian.
2. Magister Sistem Informasi Vol.3, No.2. Agustus 2017, Universitas Kristen Satya Wacana Salatiga, jurnal yang ditulis Abdi Samuel dan Danny Manongga dengan judul “Sistem Antrian Online PT. Bank Negara Indonesia (Persero) Tbk Kantor Cabang Parigi”. Hasil pesanan nomor urut yang dipesan oleh nasabah secara online, mencakup nomor antrian dan waktu yang dibutuhkan dalam antrian . Namun mekanisme aplikasi antrian yang dibangun hanya dapat diakses melalui web.
3. Teknik Informatika Vol.5, No.3. Agustus 2018, Universitas Brawijaya, jurnal yang ditulis Rizal Arif Zulfikar, Ahmad Afif Supianto, dengan judul “Rancang Bangun Aplikasi Antrian Poliklinik Berbasis Mobile” aplikasi antrian online berbasis mobile digunakan untuk pelanggan , dan website untuk admin. Dilengkapi dengan fitur pendukung, aplikasi juga menyediakan fitur untuk memesan ruangan rawat inap. Namun peneliti tidak memberikan estimasi waktu yang dibutuhkan pelanggan untuk dapat mengetahui berapa lama waktu yang dibutuhkan untuk melakukan antrian dengan nomor yang di dapat.

#### **2.2 Landasan Teori**

Pada sub bab ini akan membahas mengenai teori-teori yang menunjang penelitian. Akan dijelaskan tentang teori yang digunakan penulis untuk perancangan, pembuatan hingga pada tahap pengujian aplikasi.

### **2.2.1 Aplikasi**

Belakangan ini istilah kata aplikasi banyak sekali digunakan. Menurut Jogiyanto HM (dalam suhartini (2017), “Aplikasi merupakan penerapan, menyimpan sesuatu hal, data, permasalahan, pekerjaan ke dalam suatu sarana atau media yang dapat digunakan untuk diterapkan menjadi sebuah bentuk yang baru.”

Menurut KBBI (Kamus Besar Bahasa Indonesia) “Aplikasi adalah penerapan dari rancang sistem untuk mengolah data yang menggunakan aturan atau ketentuan bahasa pemrograman tertentu. Aplikasi adalah suatu program komputer yang dibuat untuk mengerjakan dan melaksanakan tugas khusus dari user (pengguna).”

Dengan demikian dapat disimpulkan bahwa pengertian aplikasi adalah sebuah rancangan dari sebuah sistem yang dapat difungsikan secara khusus untuk dapat mengerjakan suatu tugas atau perintah tertentu yang telah diatur sesuai dengan yang membangun rancangan tersebut.

### **2.2.2 Antrian**

Antrian dengan bahasa inggrisnya disebut sebagai *queueing atau waiting line* adalah fenomena yang sering dijumpai pada kehidupan sehari-hari. Pada umumnya, kebanyakan dari semua orang pernah merasakan menunggu sebuah antrian, biasanya antrian ini terjadi dalam suatu garis tunggu di sebuah fasilitas pelayanan umum atau tempat lainnya sebelum mendapatkan sebuah produk, jasa maupun layanan. Antrian dapat terjadi karena berbagai sebab, penyebab secara umum terjadinya antrian dikarenakan pelanggan yang datang melebihi jumlah perkiraan dari fasilitas yang tersedia untuk melayani seorang pelanggan, sehingga mengakibatkan terjadinya penumpukan pelanggan dan pelanggan harus menunggu untuk beberapa waktu agar bisa dilayani. Banyak hal yang dilakukan untuk mengurai sebuah antrian, mulai menambah fasilitas pelayanan hingga dengan menggunakan aplikasi pendaftaran antrian secara online.

#### **2.2.2.1 Teori Antrian**

Menurut Iqbal (2011:95) “Antrian terdapat pada kondisi apabila obyek-obyek menuju suatu area untuk dilayani, namun kemudian menghadapi keterlambatan disebabkan oleh mekanisme pelayanan mengalami kesibukan. Antrian timbul karena adanya ketidakseimbangan antara yang dilayani dengan pelayanannya.”Sedangkan menurut (Heizer dan Render, 2005) dalam (Sunu & Broto,2017) “Antrian merupakan hal penting dalam manajemen operasi. Sistem antrian bisa ditemukan pada sektor industri maupun sektor jasa. Antrian bisa

barisan orang atau barang yang menunggu untuk dilayani dan meninggalkan barisan setelah dilayani.”

Dengan demikian dapat disimpulkan bahwa pengertian dari sebuah teori antrian adalah terjadinya penumpukan yang dilakukan oleh sekelompok orang karena terbatasnya sebuah pelayanan. Antrian dilakukan untuk tujuan tertentu atau guna mendapatkan produk, jasa maupun pelayanan yang diinginkan

### **2.2.2.2 Sistem Antrian**

Menurut (Bambang , 2016) “Sistem antrian adalah hasil pengembangan teori antrian, yang mengatur pelayanan sesuai kedatangan untuk mencapai kinerja yang efektif dan efisien, sebagai solusi dari masalah antrian.”

Sedangkan menurut (Bronson, 1991) dalam (Nia, Sugito & Budi, 2016) “Sistem antrian adalah himpunan pelanggan, pelayan, dan aturan yang mengatur kedatangan dan proses pelayanan.”

Dengan demikian dapat disimpulkan bahwa pengertian dari sistem antrian adalah datangnya beberapa pelanggan untuk mendapatkan produk, jasa atau pelayanan telah diatur oleh sebuah sistem, sehingga proses pelayanan dapat bekerja lebih efektif dan efisien.

### **2.2.2.3 Disiplin Antrian**

Menurut (Anitha,2018) “Disiplin antrian akan sangat berkaitan dengan urutan pelayanan yang akan dilakukan pelanggan saat melakukan antrian guna memasuki fasilitas pelayanan. Disiplin antrian merupakan sebuah aturan antrian yang mengacu pada peraturan pelanggan yang ada dalam barisan untuk menerima pelayanan yang terdiri dari :

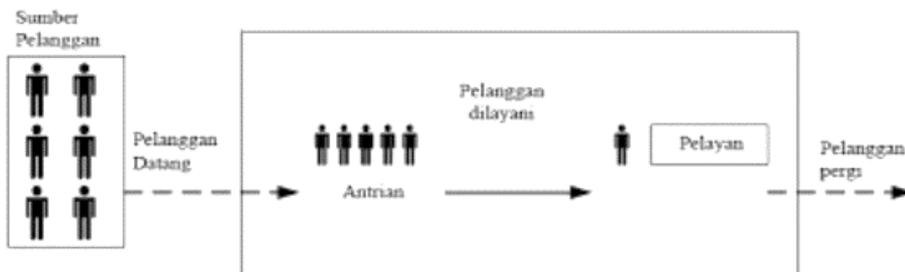
1. *First-come First-Served* atau FCFS  
Artinya, pelanggan lebih dulu datang (sampai), lebih dulu dilayani (keluar). Misalnya, antrian pada loket pembelian tiket bioskop.
2. *Last-come First-served* atau LCFS  
Artinya, pelanggan yang tiba terakhir yang lebih dulu keluar. Misalnya, sistem antrian dalam elevator untuk lantai yang sama.
3. *Priority Service* atau Disiplin Prioritas  
Artinya, prioritas pelayanan diberikan kepada pelanggan yang mempunyai prioritas lebih tinggi dibandingkan dengan pelanggan yang mempunyai prioritas lebih rendah, meskipun yang terakhir ini kemungkinan sudah lebih dahulu tiba dalam garis tunggu.
4. *Service In Random Order (SIRO)* atau Disiplin Acak  
Artinya, panggilan didasarkan pada peluang secara random, tidak soal siapa yang lebih dulu tiba.”

### 2.2.2.4 Model-Model Antrian

Menurut (Christie,2018) “Ada empat model struktur antrian dasar yang umum terjadi dalam seluruh sistem antrian:

1. Satu Jalur, Satu Tahap (*Single Channel, Single Phase*)

Sistem antrian satu jalur satu tahap berarti bahwa dalam sistem tersebut hanya terdapat satu pemberi layanan serta satu jenis layanan yang diberikan, sehingga yang telah menerima pelayanan dapat langsung keluar dari sistem antrian.

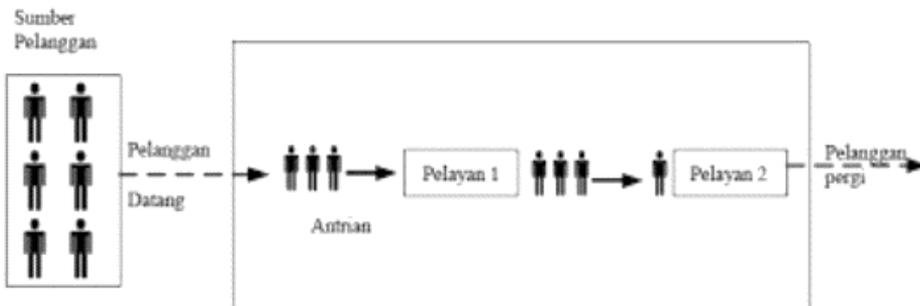


Gambar 2. 1 Satu Jalur, Satu Tahap

Gambar 2.1 menjelaskan alur kedatangan pelanggan terdiri satu jalur dan akan dilayani oleh satu fasilitas pelayanan, setelah pelayanan selesai pelanggan dapat meninggalkan fasilitas layanan.

2. Satu Jalur, Banyak Tahap (*Single Channel, Multi Phase*)

Sistem antrian satu saluran banyak tahap berarti dalam sistem antrian tersebut terdapat lebih dari satu jenis layanan yang diberikan, tetapi dalam setiap jenis layanan hanya terdapat satu pemberi layanan.

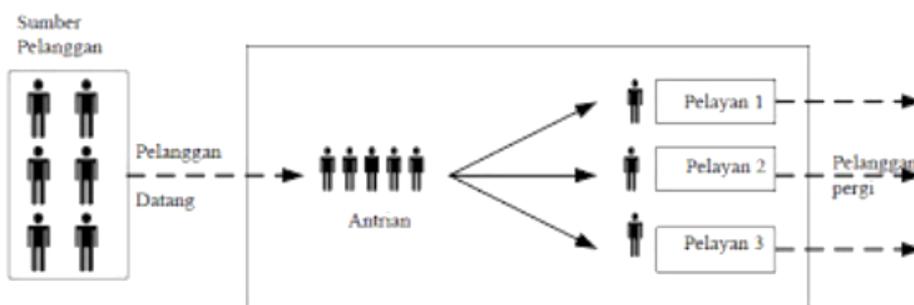


Gambar 2. 2 Satu Jalur Banyak tahap

Gambar 2.2 menjelaskan alur kedatangan pelanggan terdiri satu jalur dan akan dilayani oleh beberapa satu pelayanan dan terjadi beberapa tahap , pelanggan harus melewati setiap fasilitas pelayanan yang tersedia. Setelah melewati tahapan pertama fasilitas pelayanan, maka pelanggan akan diarahkan untuk ke fasilitas pelayanan selanjutnya dan begitu seterusnya hingga fasilitas pelayanan berakhir. Setelah tahapan fasilitas pelayanan dilalui oleh pelanggan, maka pelanggan diperbolehkan untuk meninggalkan fasilitas pelayanan.

3. Banyak Jalur, Satu Tahap (*Multi Channel, Single Phase*)

Sistem antrian banyak saluran satu tahap adalah sistem antrian dimana terdapat satu jenis layanan dalam sistem antrian tersebut, namun terdapat lebih dari satu pemberi layanan.

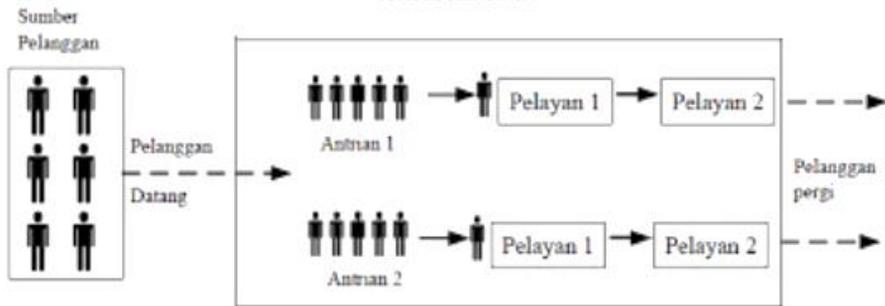


Gambar 2. 3 Banyak Jalur, Satu Tahap

Gambar 2.3 menjelaskan alur kedatangan pelanggan terdiri satu jalur dan akan disediakan beberapa fasilitas pelayanan namun hanya satu tahap, pelanggan hanya melewati satu tahapan fasilitas pelayanan, namun jika fasilitas pelayanan yang lain kosong atau sedang tidak dalam melayani pelanggan, maka fasilitas lainnya dapat memanggil pelanggan selanjutnya untuk dapat dilayani. Sehingga hal ini akan meminimalisir terjadinya sebuah antrian.

4. Banyak Jalur, Banyak Tahap (*Multi Channel, Multi Phase*)

Sistem antrian banyak saluran banyak tahap adalah sistem antrian dimana terdapat lebih dari satu jenis layanan dan terdapat lebih dari satu pemberi layanan dalam setiap jenis layanan.



Gambar 2. 4 Banyak Jalur, Banyak tahap

Gambar 2.4 menjelaskan alur kedatangan pelanggan terdiri satu jalur yang disediakan beberapa fasilitas pelayanan dan akan melalui beberapa tahap, pelanggan akan disediakan beberapa fasilitas pelayanan, jika terdapat fasilitas pelayanan yang kosong atau sedang tidak dalam melayani pelanggan, maka fasilitas lainnya dapat memanggil pelanggan selanjutnya untuk dapat dilayani, kemudian jika pelayanan pada fasilitas pelayanan pertama telah selesai maka akan diarahkan ke tahap fasilitas pelayanan yang sesuai dengan kebutuhan pelanggan.”

### 2.2.3 Android

Menurut Hermawan (2011 : 1) dalam (Siipung,2018) “Android merupakan OS (*Operating System*) Mobile yang tumbuh ditengah OS lainnya yang berkembang dewasa ini. OS lainnya seperti *Windows Mobile*, *i-Phone* atau OS, *Symbian*, dan masih banyak lagi. Akan tetapi, OS yang ada ini berjalan dengan memprioritaskan aplikasi inti yang dibangun sendiri tanpa melihat potensi yang cukup besar dari aplikasi pihak ketiga. “

Android menurut Nazaruddin (2012:1) dalam (Siipung,2018) “Merupakan sistem operasi untuk telepon seluler yang berbasis Linux. Android menyediakan platform terbuka bagi para pengembang untuk menciptakan aplikasi mereka sendiri untuk digunakan oleh bermacam peranti bergerak. Android umum digunakan di *smartphone* dan juga *tablet PC*. Fungsinya sama seperti sistem operasi *Symbian* di *Nokia*, *iOS* di *Apple* dan *BlackBerry OS*.”

Dengan demikian dapat disimpulkan bahwa pengertian dari android adalah sebuah sistem operasi yang digunakan untuk telepon seluler berbasis linux dimana didalamnya terdapat berbagai aplikasi untuk membantu pengguna sekaligus disediakan sebuah platform untuk dapat mengembangkan atau menciptakan sebuah aplikasi sesuai dengan kebutuhan pengguna.

### 2.2.3.1 Sejarah Android

Menurut (Hendra, Alicia & Arie, 2015) “Perjalanan Android dimulai sejak Oktober 2003 ketika 4 orang pakar IT, Andi Rubin, Rich Miner, Nick Sears dan Chris White mendirikan Android.Inc, di California US. Visi Android untuk mewujudkan *mobile device* yang lebih peka dan mengerti pemiliknya, kemudian menarik raksasa dunia maya *Google*. *Google* kemudian mengakuisisi Android pada Agustus 2005. OS Android dibangun berbasis *platform* Linux yang bersifat *open source*, senada dengan Linux, Android juga bersifat *Open Source*. Dengan nama besar *Google* dan konsep *open source* pada OS Android, tidak membutuhkan waktu lama bagi android untuk bersaing dan menyisihkan *Mobile OS* lainnya seperti *Symbian*, *Windos Mobile*, *Blackberry* dan *iOS*. Kini siapa yang tak kenal Android yang telah menjelma menjadi penguasa *Operating System* bagi *Smartphone*.”

Menurut (Andi, 2015) “Android adalah sebuah sistem operasi perangkat mobile berbasis linux yang mencakup sistem operasi, middleware dan aplikasi. Android menyediakan platform terbuka bagi para pengembang untuk menciptakan aplikasi mereka. Awalnya, Google Inc. membeli Android Inc. yang merupakan pendatang baru yang membuat peranti lunak untuk ponsel atau *smartphone*. Kemudian untuk mengembangkan Android, dibentuklah *Open Handset Alliance*, konsorsium dari 34 perusahaan peranti keras, peranti lunak dan telekomunikasi, termasuk *Google*, *HTC*, *Intel*, *Motorola*, *Qualcomm*, *T-Mobile*, dan *Nvidia*. Pada saat perilis perdana Android, 5 November 2007, Android bersama *Open Handset Alliance* menyatakan mendukung pengembangan *open source* pada perangkat mobile. Di lain pihak, *Google* merilis kode-kode Android di bawah lisensi *Apache*, sebuah lisensi perangkat lunak dan *open platform* perangkat seluler”

Dapat disimpulkan dari penjelasan tentang perjalanan sejarah android yang berawal di tahun Oktober 2003 dengan visinya yang akan mewujudkan sebuah *mobile device*. Kemudian pada tahun 2005, android diakuisisi oleh *Google*. Kemudian android diputuskan dibuat dengan menggunakan *Linux* sebagai dasar untuk *Android OS*, hal ini dilakukan guna tercapainya sebuah strategi untuk memasuki ruang *mobile*. Adapun sejarah singkat versi android adalah sebagai berikut :

Tabel 2. 1 Sejarah singkat versi android

Versi Android	Tanggal Rilis	Nama Versi
1.1	9 Februari 2019	

Versi Android	Tanggal Rilis	Nama Versi
1.5	30 April 2009	Cupcake
1.6	15 September 2009	Donut
2.0 / 2.1	26 Oktober 2009	Éclair
2.2	10 Mei 2010	Froyo
2.3	6 Desember 2010	Gingerbread
3.0 / 3.1 / 3.2	22 Februari 2011	Honeycomb
4.0	19 Oktober 2011	Ice Cream Sandwich
4.1	9 Juli 2012	Jelly Bean
4.4	31 Oktober 2013	Kitkat
5.0	12 November 2014	Lollipop
6.0	28 Mei 2015	Marshmallow
7.0	22 Agustus 2016	Nougat
8.0	21 Agustus 2017	Oreo
9.0	6 Agustus 2018	Pie
10.0	7 Agustus 2019	Q

Sumber: DiMarzio (2012:2-3)

### 2.2.3.2 Java

(Bambang Haryanto, 2011:2, Esensi-esensi Bahasa Pemrograman *Java*. Yogyakarta: Andi) dalam (Materipasti, 2016) “*Java* merupakan bahasa berorientasi objek untuk pengembangan aplikasi mandiri, aplikasi berbasis internet, aplikasi untuk perangkat cerdas yang dapat berkomunikasi lewat internet/ jaringan komunikasi. Melalui teknologi *java*, dimungkinkan perangkat *audio streo* dirumah terhubung jaringan komputer. *Java* tidak lagi hanya untuk membuat *applet* yang memerintah halaman web tapi *java* telah menjadi bahasa untuk pengembangan aplikasi skala *interprise* berbasis jaringan besar.”

Menurut Abdul Kadir (2011:3) dalam (Ali Subhan, 2017) “*Java* merupakan pemrograman yang bersifat lintas platform. Artinya, bahasa ini dapat dipakai untuk menyusun program pada berbagai sistem operasi (Linux, Windows, UNIX).”

Dapat disimpulkan dari penjelasan tentang *java* adalah sebuah bahasa pemrograman yang dapat digunakan di berbagai platform. Bahasa pemrograman *java* menjadi bahasa untuk pengembangan berbasis jaringan besar, karena dengan menggunakan *java* seorang pengembang dapat dengan mudah menjalankan program di semua komputer yang support dengan *java*.

### **2.2.3.3 Android SDK (*Software Development Kit*)**

Menurut (Nazruddin Safaat, 2011) dalam (Ari Pambudi, 2013) “Android SDK adalah tools API (Application Programming Interface) yang diperlukan untuk mulai mengembangkan aplikasi pada platform Android menggunakan bahasa pemrograman Java. Saat ini disediakan Android SDK sebagai alat bantu dan API untuk mulai mengembangkan aplikasi pada platform Android menggunakan bahasa pemrograman Java. SDK mencakup perangkat tools pengembangan yang komprehensif. Android SDK terdiri dari debugger, libraries, handset emulator, dokumentasi, contoh kode program dan tutorial.”

Sedangkan menurut (Ir. Yuniar Supardi, 2015) “Android SDK merupakan alat atau tool yang digunakan untuk membuat aplikasi platform Android menggunakan bahasa pemrograman java. Android SDK merupakan tools bagi programmer yang ingin mengembangkan aplikasi berbasis google android. Android SDK mencakup seperangkat alat pengembangan yang komprehensif. Android SDK terdiri dari debugger, libraries, handset emulator, dokumentasi, contoh kode dan tutorial.”

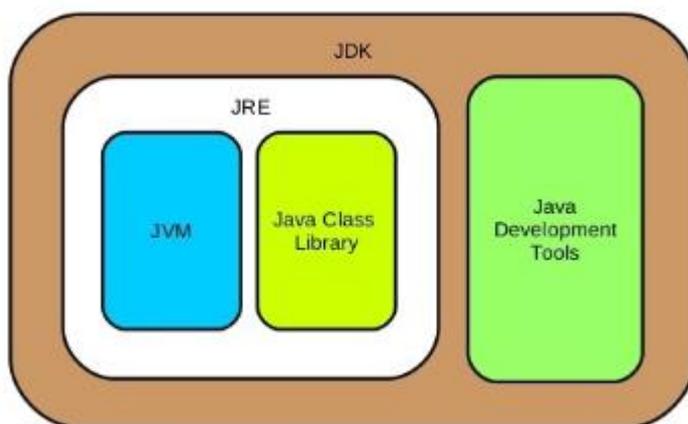
Dengan demikian dapat disimpulkan bahwa android SDK adalah sebuah alat dari kode java yang dapat secara lancar dan diperbolehkan untuk bisa berjalan di perangkat android. Pemrograman java sendiri dibutuhkan untuk menulis program, sedangkan android SDK dibutuhkan untuk menjalankan program di Android. Maka dengan menggabungkan keduanya dibutuhkanlah aplikasi Android Studio. Sehingga seorang programmer dapat menemukan bug pada aplikasi yang sedang dikembangkannya.

### **2.2.3.4 Android JDK (*Java Development Kit*)**

Menurut Khannedy (2011:1) dalam (Lilip, 2016) “*Java Development Kit* merupakan perangkat lunak yang digunakan untuk melakukan proses kompilasi dari kode Java menjadi bytecode yang dapat dimengerti dan dapat dijalankan oleh Java Runtime Environment (JRE)”. Java Development Kit wajib terinstall pada komputer yang akan melakukan proses pembuatan aplikasi berbasis Java. Namun Java Development Kit tidak wajib terinstal di komputer yang akan menjalankan aplikasi yang dibangun menggunakan Java.

Sedangkan Menurut (DeCoster, 2012), “*Java* adalah sebuah teknologi yang diperkenalkan oleh *Sun Microsystems* pada pertengahan tahun 1990. Menurut definisi Sun, *Java* adalah nama untuk sekumpulan teknologi untuk membuat dan menjalankan perangkat lunak pada *computer standalone* ataupun pada lingkungan jaringan. Untuk membuat program *Java* dibutuhkan *kompiler* dan *interpreter* untuk program *Java* berbentuk *Java Development Kit* (JDK) yang diproduksi oleh *Sun Microsystems*. Sebelum memulai instalasi *Android SDK*, terlebih dahulu kita harus melakukan *instalasi* JDK di komputer.”

Dengan demikian dapat disimpulkan bahwa *android JDK* adalah sebuah aplikasi yang digunakan untuk mengkompilasi dari kode *java* menjadi *bytecode*. *JDK* yang kami gunakan untuk dapat mengompilasi aplikasi *Android* yang kami rancang ini adalah *Java SE Development Kit 8*.



Gambar 2. 5 Isi dari JDK (Duniaikom, 2019).

Gambar 2.5 menjelaskan bahwa pada sebuah *JDK* sebuah paket yang berisi berisi *JVM* (*Java Virtual Machine*) yang berfungsi untuk mengubah bahasa program *java* menjadi bahasa mesin, hal ini dibutuhkan karena bahasa *java* merupakan bahasa pemrograman tingkat tinggi atau mendekati bahasa manusia maka dibutuhkan pengkonversi ke bahasa mesin., *JRE* (*Java Runtime Environment*) yang berfungsi menjalankan aplikasi yang dibangun menggunakan *java* dan terdapat aplikasi yang dibutuhkan untuk proses pembuatan yang menggunakan kode pemrograman *java*.

### 2.2.3.5 Android Studio

Menurut (Efmi,2018) “*Android Studio* merupakan sebuah *IDE* (*Integrated Development Environment*) untuk pengembangan aplikasi *android*, aplikasi ini dipublikasikan oleh *Google* pada tanggal 16 mei 2013 dan tersedia

secara gratis dibawah lisensi *Apache 2.0*, Android studio ini menggantikan software pengembangan android sebelumnya yaitu *Eclipse*.”

Sedangkan menurut (Android, 2020) “Android Studio adalah Lingkungan Pengembangan Terpadu (*Integrated Development Environment/IDE*) resmi untuk pengembangan aplikasi Android, yang didasarkan pada IntelliJ IDEA. Selain sebagai editor kode dan fitur developer IntelliJ yang andal, Android Studio menawarkan banyak fitur yang meningkatkan produktivitas Anda dalam membuat aplikasi Android, seperti:

1. Sistem build berbasis Gradle yang fleksibel
2. Emulator yang cepat dan kaya fitur
3. Lingkungan terpadu tempat Anda bisa mengembangkan aplikasi untuk semua perangkat Android
4. Terapkan Perubahan untuk melakukan push pada perubahan kode dan *resource* ke aplikasi yang sedang berjalan tanpa memulai ulang aplikasi
5. Template kode dan integrasi GitHub untuk membantu Anda membuat fitur aplikasi umum dan mengimpor kode sampel
6. Framework dan fitur pengujian yang lengkap
7. Fitur lint untuk merekam performa, kegunaan, kompatibilitas versi, dan masalah lainnya
8. Dukungan C++ dan NDK
9. Dukungan bawaan untuk *Google Cloud Platform*, yang memudahkan integrasi *Google Cloud Messaging* dan *App Engine*”

Dengan demikian dapat disimpulkan bahwa android studio adalah sebuah *Integrated Development Environment (IDE)* yang digunakan untuk membuat dan mengembangkan sebuah aplikasi yang dapat dijalankan pada platform android. Android studio berbasis IntelliJ IDEA, dimana IDE ini digunakan untuk bahasa pemrograman Java, untuk layout atau tampilannya menggunakan bahasa XML. Untuk *deploy* ke perangkat android, android studio telah terintegrasi dengan *Android Software Development Kit (SDK)*.

#### **2.2.4 Unified Modeling Language**

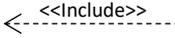
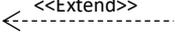
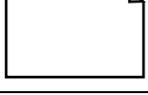
Menurut Sukamto dan Shalahuddin (2014:137) menjelaskan bahwa “UML (Unified Modeling Language) adalah standarisasi bahasa pemodelan untuk pembangunan perangkat lunak yang dibangun menggunakan teknik pemograman berorientasi objek. UML (Unified Modeling Language) muncul karena adanya kebutuhan pemodelan visual untuk menspesifikasi, menggambarkan, membangun, dan dokumentasi dari sistem perangkat lunak.”

1. Use Case Diagram

Menurut Satzinger (2011 : 20) “Use Case Diagram merupakan rangkaian tindakan yang dilakukan oleh sistem, aktor mewakili user atau sistem lain yang

berinteraksi dengan sistem yang dimodelkan.” Berikut simbol-simbol use case diagram disertai dengan penjelasan kegunaan dari simbolnya.

Tabel 2. 2 Simbol Use Case Diagram

Gambar	Nama	Keterangan
	<i>Actor</i>	Menspesifikasikan himpunan peran yang pengguna mainkan ketika berinteraksi dengan use case.
	<i>Association</i>	Apa yang menghubungkan antara objek satu dengan objek lainnya.
	<i>System</i>	Menspesifikasikan paket yang menampilkan sistem secara terbatas.
	<i>Generalization</i>	Hubungan dimana objek anak (descendent) berbagi perilaku dan struktur data dari objek yang ada di atasnya objek induk (ancestor).
	<i>Include</i>	Menspesifikasikan bahwa use case sumber secara <i>eksplisit</i> .
	<i>Extend</i>	Menspesifikasikan bahwa use case target memperluas perilaku dari use case sumber pada suatu titik yang diberikan.
	<i>Use Case</i>	Deskripsi dari urutan aksi-aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang terukur bagi suatu aktor.
	<i>Collaboration</i>	Interaksi aturan-aturan dan elemen lain yang bekerja sama untuk menyediakan perilaku yang lebih besar dari jumlah dan elemen-elemennya (sinergi).
	<i>Note</i>	Elemen fisik yang eksis saat aplikasi dijalankan dan mencerminkan suatu sumber daya komputasi.

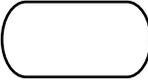
Tabel 2. 2 Simbol Use Case Diagram menjelaskan simbol-simbol yang digunakan untuk membuat use case diagram beserta nama dari setiap simbol dan dilengkapi dengan penjelasan kegunaan dari setiap simbol.

Dengan demikian dapat disimpulkan bahwa use case diagram digunakan untuk mengetahui alur atau tindakan yang akan terjadi pada sebuah aplikasi yang akan kita bangun. Use case diagram menggambarkan bagaimana user atau aktor akan berinteraksi dengan aplikasi yang dibangun.

## 2. Activity Diagram

Menurut Sukanto dan Shalahuddin (2014:161) “diagram aktivitas atau activity diagram adalah menggambarkan aliran kerja atau aktifitas dari sebuah sistem atau proses bisnis atau menu yang ada pada perangkat lunak. Diagram aktifitas menggambarkan aktifitas sistem bukan apa yang dilakukan oleh aktor.”

Tabel 2. 3 Simbol Activity Diagram

Gambar	Nama	Keterangan
	<i>Activity</i>	Memperlihatkan bagaimana masing-masing kelas antarmuka saling berinteraksi satu sama lain.
	<i>Action</i>	State dari sistem yang mencerminkan eksekusi dari suatu aksi.
	<i>Initial Node</i>	Bagaimana objek dibentuk atau diawali.
	<i>Activity Final Node</i>	Bagaimana objek dibentuk dan dihancurkan
	<i>Fork Node</i>	Satu aliran yang pada tahap tertentu berubah menjadi beberapa aliran

Tabel 2. 3 Simbol Activity Diagram menjelaskan simbol-simbol yang digunakan untuk membuat activity diagram beserta nama dari setiap simbol dan dilengkapi dengan penjelasan kegunaan dari setiap simbol.

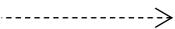
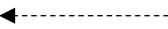
Dengan demikian dapat disimpulkan bahwa activity diagram adalah penjelasan tentang alur kerja dari setiap menu yang ada pada program. Menggambarkan bagaimana sebuah menu akan bereaksi. Pada Diagram Activity

lebih menekankan pada penjelasan alur atau aktifitas menu dari aplikasi bukan aktifitas yang dilakukan oleh user atau aktor.

### 3. Class Diagram

Menurut Rosa dan M. Shalahudin (2014:141) “diagram kelas atau class diagram menggambarkan struktur program dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun aplikasi.”

Tabel 2. 4 Simbol Class Diagram

Gambar	Nama	Keterangan
	<i>Generalization</i>	Hubungan dimana objek anak ( <i>descendent</i> ) berbagi perilaku dan struktur data dari objek yang ada di atasnya objek induk ( <i>ancestor</i> ).
	<i>Nary Association</i>	Upaya untuk menghindari asosiasi dengan lebih dari 2 objek.
	<i>Collaboration</i>	Deskripsi dari urutan aksi-aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang terukur bagi suatu aktor.
	<i>Dependency</i>	Hubungan dimana perubahan yang terjadi pada suatu elemen mandiri ( <i>independent</i> ) akan mempegaruhi elemen yang bergantung padanya elemen yang tidak mandiri.
	<i>Realization</i>	Operasi yang benar-benar dilakukan oleh suatu objek.
	<i>Class</i>	Himpunan dari objek-objek yang berbagi atribut serta operasi yang sama.
	<i>Association</i>	Apa yang menghubungkan antara objek satu dengan objek lainnya

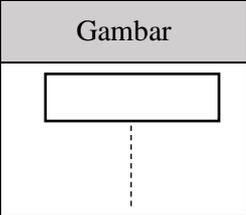
Tabel 2. 5 Simbol Class Diagram menjelaskan simbol-simbol yang digunakan untuk membuat class diagram beserta nama dari setiap simbol dan dilengkapi dengan penjelasan kegunaan dari setiap simbol.

Dengan demikian dapat disimpulkan bahwa class diagram adalah menjelaskan tentang penggambaran struktur dari program aplikasi yang akan dibangun.

#### 4. Sequence Diagram

Sequence adalah satu dari dua interaksi diagram yang mengilustrasikan objek-objek yang berhubungan dengan use case dan message atau pesan-pesannya. Menurut Pratama (2014:48) “sequence diagram menggambarkan aliran pengiriman pesan yang terjadi di aplikasi, sebagai bentuk interaksi dengan pengguna (user).”

Tabel 2. 5 Simbol Sequence Diagram

Gambar	Nama	Keterangan
	<i>LifeLine</i>	Objek entity, antarmuka yang saling berinteraksi.
	<i>Message</i>	Spesifikasi dari komunikasi antar objek yang memuat informasi-informasi tentang aktifitas yang terjadi
	<i>Initial Node</i>	Spesifikasi dari komunikasi antar objek yang memuat informasi-informasi tentang aktifitas yang terjadi

Tabel 2. 4 Simbol Sequence Diagram menjelaskan simbol-simbol yang digunakan untuk membuat sequence diagram beserta nama dari setiap simbol dan dilengkapi dengan penjelasan kegunaan dari setiap simbol.

Dengan demikian dapat disimpulkan bahwa sequence diagram adalah interaksi diagram yang masih berhubungan dengan use case diagram yang mana menggambarkan bagaimana interaksi pesan yang dikirim dari sebuah program, dengan pesan yang disampaikan oleh program terjadi sebuah interaksi yang akan dilakukan oleh user atau aktor.

Berikut mengenai penjelasan UML (*Unified Modelling Language*) dilengkapi dengan jenis-jeninya. Dengan demikian dapat disimpulkan bahwa UML (*Unified Modelling Language*) adalah sebuah penggambaran visual dari gambaran perancangan sebuah program yang akan dibuat oleh seorang

pemrogram. Dengan menggunakan UML (*Unified Modelling Language*) akan mempermudah pengembangan aplikasi yang berkelanjutan, karena perancangan dari sebuah program telah terdokumentasi.

### **2.2.5 Website**

Menurut Nugroho dalam Ade (2016) menjelaskan bahwa “Website atau situs dapat diartikan sebagai kumpulan halaman-halaman yang berasal dari file-file berisi bahasa pemrograman yang saling berhubungan digunakan untuk menampilkan informasi, gambar bergerak dan tidak bergerak, suara dan atau gabungan dari semuanya itu baik yang bersifat statis maupun dinamis.”

Menurut Arief (2011:7a) dalam (Muhammad Soni, 2016) “Web adalah salah satu aplikasi yang berisikan dokumen-dokumen multimedia (teks, gambar, suara, animasi, video) di dalamnya yang menggunakan protokol HTTP (Hypertext Transfer Protocol) dan untuk mengaksesnya menggunakan perangkat lunak yang disebut browser.”

Dengan demikian dapat disimpulkan bahwa *Website* adalah sebuah aplikasi yang berisikan sekumpulan dokumen-dokumen yang dapat menampilkan berbagai macam informasi multimedia seperti teks, data, animasi, video maupun gabungan dari semuanya sehingga akan menjadi satu rangkaian program yang saling berhubungan satu sama lain, untuk menghubungkan semuanya maka dibutuhkanlah sebuah jaringan halaman (*hiperlink*).

### **2.2.6 CodeIgniter**

Menurut (Betha Sidik, 2012) dalam (Mara & Qadhli, 2017) "CodeIgniter adalah Sebuah framework php yang bersifat open source dan menggunakan metode MVC (Model, View, Controller) untuk memudahkan developer atau programmer dalam membangun sebuah aplikasi berbasis web tanpa harus membuatnya dari awal."

Menurut Winarno dkk (2011:16) dalam (Herva, 2015) "Codeigniter dapat mempercepat proses pengembangan web, karena web framework memiliki library yang sudah didefinisikan sehingga programmer hanya tinggal memanggil library dan fungsi-fungsi tersebut, tidak perlu lagi membuatnya dari nol serta menghemat waktu dalam penggunaannya"

Dengan demikian dapat disimpulkan bahwa codeigniter adalah sebuah framework php atau kerangka kerja. Codeigniter memiliki library yang sudah didefinisikan sehingga mempermudah pemrogram untuk memanggil library, fungsi dan class-class tanpa harus membuatnya dari awal.

### 2.2.7 Konsep MVC

Menurut Rosa dan Shalahuddin (2011 : 192) dalam (Imam, 2014) “Konsep *Model View Controller* (MVC) bertujuan agar sebuah aplikasi dapat mudah dipelihara oleh orang-orang di dalam tim pengembangan yang berbeda spesifikasi pekerjaan, misalnya *database* administrator (DBA) untuk mengurus masalah basis data, blok *controller* untuk programmer, dan blok *view* untuk desainer antarmuka (*interface designer*).”

Sedangkan menurut (Badiyanto, 2013) “MVC adalah sebuah metode untuk membuat sebuah aplikasi dengan memisahkan data atau *query* (*Model*) dari tampilan atau user *interface* (*View*) dan cara pemrosesannya (*Controller*). Pembuatan program yang menggunakan MVC ini biasanya dikemas dalam kerangka kerja (*framework*), sehingga pengembang aplikasi tinggal menggunakan kerangka kerja yang sudah disediakan.

1. *Model*, merupakan kelas yang mendasari logika proses dalam aplikasi perangkat lunak dan kelas yang terkait dengannya. Model adalah suatu objek yang tidak mengandung informasi tentang *user interface*.
2. *View*, merupakan kumpulan dari kelas yang mewakili unsur-unsur dalam *user interface* (semua hal user dapat melihat dan merespon pada layar, seperti tombol, tampilan kotak, dan sebagainya).
3. *Controller*, merupakan kelas yang menghubungkan *model* dan *view*, dan digunakan untuk berkomunikasi antara kelas dalam *model* dan *view*.

Dengan demikian dapat disimpulkan bahwa konsep MVC adalah sebuah metode untuk pembangunan aplikasi berbasis web. Dimana terdapat 3 komponen yang dipisahkan dan memiliki peran serta tanggung jawab masing-masing yaitu model, view dan Controller guna mempermudah tim pengembangan.

### 2.2.8 Database

Pengertian database menurut Anhar (2010:45), “Database adalah sekumpulan tabel-tabel yang berisi data dan merupakan kumpulan dari field atau kolom. Struktur file yang menyusun sebuah database adalah data Record dan Field”.

Sedangkan menurut (Nugroho, Yuliandri Priyo, 2012) dalam (Ade, 2016) “Database juga merupakan kumpulan data yang umumnya menggambarkan aktifitas-aktifitas dan pelakunya dalam suatu organisasi. Sistem database merupakan sistem komputer yang digunakan untuk menyimpan dan mengelola data tersebut.”

Dengan demikian dapat disimpulkan bahwa database adalah sekumpulan data yang tersusun pada sebuah tabel berisikan field dan data record yang dapat diolah guna menghasilkan sebuah informasi yang dibutuhkan.

*Halaman ini sengaja dikosongkan*