BAB 2
TINJAUAN PUSATAKA

1.1. Landasan Teori

Landasaran teori merupakan dasar teori dan definisi serta konsep dari apa-apa saja yang akan digunakan dalam sebuah penelitian. Landasaran teori-teori yang peneliti gunakan pada penelitian ini adalah sebagai berikut.

1.1.1. Tinjauan Objek Penelitian

Berikut ini adalah landasan teori dari objek penelitian yang diriset oleh peneliti:

1.1.1.1.Desa Cibodas

Desa Cibodas adalah sebuah desa di kecamatan Pacet Kabupaten Cianjur Jawa Barat. Luas wilayah desa Cibodas adalah 509 Ha. Desa cibodas secara geografis dikelilingi oleh dua gugusan bukit, yaitu sebelah selatan bukit cikanyere dan sebelah barat dikelilingi oleh bukit Gunung Kasur dan Gunung Gedogan. Dan didalamnya dialiri dengan aliran sungai yang setiap musim tidak pernah kering, yaitu sungai Cipendawa, sungai cibodas, yang membelah wilayah Desa Cibodas persis membagi dua wilayah Desa Cibodas dan juga dilalui dengan aliran sungai Cipendawa. Akses Jalan umum ke desa ini hanya ada satu yaitu jalan Hanjawar-pacet. Selain jalan tersebut hanya ada jalan setapak dan jalan-jalan bukit yang. Peta wilayah desa Kecamatan Pacet diperlihatkan oleh gambar 2.1 berikut :

Sumber gambar : http://kotakita.com

Gambar 2.1 Peta wilayah Kecamatan Pacet

1.1.2. Transportasi

Transportasi atau *transportare*, dimana trans berarti seberang dan portate berarti mengangkut atau membawa (sesuatu) ke sebelah lainnya dari suatu tempat ke tempat lainnya. Berarti transportasi adalah suatu jasa yang diberikan, guna memudahkan manusia atau barang untuk dibawa dari suatu tempat ke tempat lainnya. Dengan demikian transportasi dapat didefinisikan sebagai usaha dan kegiatan mengangkut atau membawa dan atau penumpang dari suatu tempat ke tempat lainnya[11].

Transportasi dapat diklasisfikasikan dari berbagai segi[11]. Dari segi barang yang diangkut, transportasi dapat diklasifikasikan sebagai berikut :

a. Angkutan penumpang
b. Angkutan barang
c. Angkutan pos (*mail*)

Dari segi geografis, angkutan umum dapat dibagi sebagai berikut :

a. Angkutan antar benua
b. Angkutan antar kontinental
c. Angkutan antar pulau
d. Angkutan antar kota
e. Angkutan antar daerah
f. Angkutan dalam kota

Dari segi Teknik dan alat pengangkutnya, transportasi dapat dibagi sebagai berikut:

a. Angkutan jalan raya
b. Angkutan rel
c. Angkutan melalui air di pedalaman (seperti di sungai, kanal, danau, dan sebagainya)
d. Angkutan pipa (seperti pengangkut minyak, bensin dan sebagainya)

1.1.2.1. Angkutan Umum

Angkutan umum penumpang yaitu angkutan yang dilakukan dengan sewa atau pembayaran. Pengangkutan umum dibedakan dalam tiga kategori utama yaitu

angkutan antar kota, angkutan perkotaan dan angkutan pedesaan. Peranan utama angkutan umum adalah melayani kepentingan mobilitas masyarakat dalam melakukan kegiatannya, baik kegiatan sehari-hari yang berjarak pendek atau menengah, maupun kegiatan sewaktu-waktu antar provinsi[12]. Aspek lain pelayanan angkutan umum adalah peranannya dalam pengendalian lalu lintas penghematan energi, dan pengembangan wilayah.

1.1.3. Angkutan Umum Desa Cibodas Parigi-Cipanas

Angkutan umum desa Cibodas Parigi-Cipanas adalah salah satu angkutan umum desa yang telah terdaftar pada Dinas Perhubungan Kabupaten Cianjur. Angkutan ini adalah salah satu dari Sembilan angkutan umum yang beroperasi di terminal atau kota Cipanas. Perbedaan kombinasi warna kendaraan angkutan menjadi ciri dari masing-masing trayek. Tabel 2.1 berikut ini adalah detail dari kesembilan trayek yang beroperasi di kota Cipanas :

Tabel 2.1 Rute angkutan umum terminal Cipanas

No	Nama Trayek	Rute Perjalanan	Warna angkot	Keterangan warna
1	Cipanas – Cianjur	Cianjur (Ramayana) - Cugenang - Ciherang - Pacet – Cipanas		Biru tua
2	Cipanas – Ciherang	Cipanas - Pacet – Ciherang		Kuning
3	Cipanas - Beunying	Cipanas - Pacet – Beunying		Kuning - Putih
4	Cipanas - Pasir Kampung	Cipanas - Pasir Kampung		Kuning - Hijau
5	Cipanas - Puncak	Cipanas - Cimacan - Loji – Puncak		Kuning - Merah muda
6	Cipanas - Cibodas	Cipanas - Rarahan - Cibodas		Kuning - Coklat Muda
7	Cipanas - Loji Sukanagalih	Cipanas - Cimacan - Loji - Sukanagalih – GSP		Kuning - Hitam
8	Cipanas - Simpang – Mariwati	Cipanas - GSP - Simpang - Sukaresmi – Mariwati		Kuning - Ungu
9	Cipanas - Cibodas (Selanjambu)	Cipanas - GSP - Cibodas Selajambu		Kuning - Biru

Angkutan yang peneliti jadikan objek penelitian adalah trayek Cipanas – Cibodas (Selajambu) yang terdaftar dengan nama trayek Cipanas-Selajambu. Angkutan umum ini beroperasi dimulai dari pukul 04:00 WIB sampai dengan 18:00 WIB. Bertrayek atau memiliki rute dari desa Cibodas Parigi menuju kota Cipanas dengan estimasi waktu sekitar 30 menit untuk setiap perjalanan. Angkutan umum desa Cibodas Parigi-Cipanas memiliki jarak tempuh 8km dengan biaya ongkos maksimal Rp.7000 untuk setiap perjalanan per orang.

Tabel 2.2 berikut ini memperlihatkan detail biaya angkutan umum Cipanas-Cibodas(selajambu) :

Tabel 2.2 Detail harga

No	Asal – Tujuan	Pelajar	Umum	Tambahan barang
1	Cipanas - Cibodas	Rp. 5.000	Rp. 7.000	Sesuai jumlah / berat barang
2	Cipanas – GSP	Rp. 3.000	Rp. 4.000	
3	GSP – Cibodas	Rp. 3.000	Rp. 4.000	
4	Jarak dekat	Rp. 2000	Rp. 2000	

Adapun sistem kerja atau alur kerja angkutan umum desa ini setiap hari adalah dengan di awali mencari penumpang di desa Cibodas Parigi (selanjambu) dengan cara berkeliling desa. Hal ini untuk memaksimalkan didapatkannya warga yang ingin menggunakan jasa angkutan umum ini. Setelah berkeliling, selanjutnya pengemudi memberhentikan kendaraan dan menunggu penumpang(mengetem) di depan kantor desa Cibodas. Setelah isi kendaraan dirasa cukup penumpang, angkutan melaju menuju tujuan yaitu kota Cipanas. Sistem pembayaran menggunakan uang tunai rupiah yang dibayarkan penumpang pada saat penumpang turun atau sudah sampai tujuan.

Perjalanan kembali ke desa Cibodas Parigi dimulai dengan pengemudi menunggu para penumpang di terminal angkutan umum yang terletak di Pasar Induk Cipanas. Setelah dirasa cukup penumpang, pengemudi kembali mengemudi dengan tujuan desa Cibodas, dan mengantarkan penumpang ke jalan/depan rumah penumpang sekaligus berkeliling dan mencari penumpang untuk diantarkan

kembali dengan tujuan kota Cipanas. Gambar 2.2 berikut ini adalah gambar dari angkutan umum desa Cipanas – Cibodas (Selajambu) yang dijadikan objek penelitian :

Gambar 2.2 Angkutan umum Cipanas – Cibodas (Selajambu)

Manajemen angkutan umum desa ini dilakukan oleh masing-masing pemilik kendaraan. Jika sebelumnya (awal pembelian kendaraan) manajemen keuangan dan perawatan kendaraan dilakukan oleh koperasi, pada saat ini para pemilik kendaraan memilih untuk mengatur sendiri keuangan dan perawatan serta kerja dari angkutan umum yang dimilikinya.

Pemilik kendaraan memilih mengatur sendiri manajemen kerjanya dengan alasan sudah terlunasinya kendaraan yang sebelumnya di kreditkan oleh koperasi. Selain itu pengemudi lebih suka untuk melakukan perawatan kendaraan secara pribadi daripada terorganisir oleh koperasi. Pengemudi yang bekerja diharuskan menyetor kepada pemilik kendaraan sebesar Rp.60.000 per hari.

1.1.4. Aplikasi *Mobile*

Aplikasi *mobile* pada dasarnya adalah segala jenis aplikasi yang dibuat/dirancang untuk dipergunakan dalam perangkat *mobile*. Perangkat mobile yang dimaksud adalah Android/Ios/Windows. Sebuah perangkat lunak *mobile* pengalaman pengguna proyek dapat dibagi ke dalam dua kategori utama: yang

pertama adalah Konteks, yaitu elemen yang harus dipahami tetapi tidak bisa diubah atau dikendalikan. Ini termasuk perangkat keras biaya, kemampuan *platform* dan Konvensi UI, dan lingkungan di mana aplikasi Anda digunakan. Yang kedua Implementasi, yaitu elemen yang dapat dikontrol dalam suatu aplikasi, seperti kinerja, desain, dan integrasi dengan fitur *platform* seperti sebagai data accelerometer atau notifikasi [13].

1.1.5. Informasi

Secara umum informasi adalah data yang telah diolah menjadi bentuk yang bermanfaat. Menurut Jogiyanto, Informasi adalah data yang diolah menjadi bentuk yang lebih berguna dan lebih berarti bagi yang menerimanya. Dapat dikatakan bahwa data merupakan bahan mentah, sedangkan informasi adalah bahan jadi atau bahan yang telah siap digunakan. Jadi, sumber dari informasi adalah data. Data adalah kenyataan yang menggambarkan suatu kejadian dan kesatuan nyata. Indikator yang dapat digunakan untuk mengukur kualitas informasi adalah *relevance, accurate, completeness, timeliness,* dan *understandability* dari informasi yang dihasilkan.[14]

1.1.6. Internet

Internet adalah sebuah system informasi berskala global yang penggunanya terhubung satu sama lain secara logika oleh alamat (*address*) yang unik secara global yang berbasis pada *Internet Protocol* (IP). Mendukung komunikasi menggunakan TCP/IP, menyediakan, menggunakan, dan membuatnya bisa diakses baik secara umum maupun khusus. Penggunaan berbagai media pun telah bergeser, seperti pendokumentasian, tayangan dan lainnya kini beralih ke internet [15].
Fungsi internet yang bisa digunakan antara lain:

a. Sarana pendukung kegiatan ekonomi
b. Sarana pendukung kegiatan Pendidikan
c. Sarana *social media*
d. Sarana hiburan
e. Dan lain-lain

1.1.7. Android

Android adalah sebuah sistem operasi untuk smartphone dan tablet. Sistem operasi dapat diilustrasikan sebagai 'jembatan' antara peranti (*device*) dan penggunanya, sehingga pengguna dapat berinteraksi dengan *device*-nya dan menjalankan aplikasi-aplikasi yang tersedia pada device. Di dunia *personal computer*, sistem operasi yang banyak di pakai adalah Windows, Mac, dan Linux [16].

Di dunia *mobile device* (*smartphone* dan tablet), sistem operasi yang menguasai pasar saat ini adalah Android. Menurut data *market share* dari Gartner, Inc, pada tahun 2017, Android memegang 85.9% *market share smartphone* di seluruh dunia. iOS yang merupakan sistem operasi dari iPhone menduduki peringkat kedua dengan 14%, lalu disusul dengan system operasi lain dengan 0.1% market share. Detail angka informasi diatas bisa di lihat pada tabel 2.3 berikut :

Tabel 2.3 *Market Share Smartphone* pada tahun 2018

Operating System	**2017 Unit**	**2017 *Market Share*(%)**	**2016 *Unit***	**2016 *Market Share*(%)**
Android	1,320,118.10	85.90%	1,268,562.70	84.80%
IOS	214,924.40	14.00%	216,064.00	14.40%
Other OS	1,493.00	0.10%	11,332.20	0.80%
Total	**1,536,535.50**	**100%**	**1,495,958.90**	**100%**

Sumber : https://www.gartner.com/newsroom/id/3859963

Sejak pertama diperkenalkan pada tahun 2007, Android telah mengalami perubahan dan perkembangan yang pesat. Tabel 2.4 menunjukan versi-versi android beserta tanggal rilisnya :

Tabel 2.4 Versi-versi Android

Nama	Versi	Peluncuran
Cupcake	1.5	27-Apr-09
Donut	1.6	15-Sep-09
Éclair	2.0 – 2.1	26-Oct-09
Froyo	2.2 – 2.2.3	20-May-10
Gingerbread	2.3 – 2.3.7	6-Dec-10
Honeycomb	3.0–3.2.6	22-Feb-11
Ice Cream Sandwich	4.0 – 4.0.4	18-Oct-11
Jelly Bean	4.1 – 4.3.1	9-Jul-12
KitKat	4.4 – 4.4.4	31-Oct-13
Lollipop	5.0 – 5.1.1	12-Nov-14
Marshmallow	6.0 – 6.0.1	5-Oct-15
Nougat	7	Agustus / September 2016
Oreo	8	Agustus 2017
Pie	9	Agustus 2018

Sumber: https://mainthebest.com/smartphones/tingkatan-versi-android/

Adapun Adapun versi-versi android yang pernah diliris sebagai berikut :

1. **Android versi 1.1 (API level 1)**

Pada 9 Maret 2009, Google merilis Android versi 1.1. Android versi ini dilengkapi dengan pembaruan estetis pada aplikasi, jam alarm, *voice search* (pencarian suara), pengiriman pesan dengan Gmail, dan pemberitahuan email[17].

2. **Android versi 1.5 cupcake (API level 3)**

Pada pertengahan Mei 2009, Google kembali merilis telepon seluler dengan menggunakan Android dan SDK (*Software Development Kit*) dengan versi 1.5 (*Cupcake*). Terdapat beberapa pembaruan termasuk juga penambahan beberapa fitur dalam seluler versi ini yakni kemampuan merekam dan menonton video dengan modus kamera, mengunggah video ke Youtube dan gambar ke Picasa langsung dari telepon, dukungan *Bluetooth* A2DP, kemampuan terhubung secara otomatis ke *headset Bluetooth*, animasi layar, dan *keyboard* pada layar yang dapat disesuaikan dengan system[17].

3. Android versi 1.6 Donut (API level 4)

Donut (versi 1.6) dirilis pada September dengan menampilkan proses pencarian yang lebih baik dibanding sebelumnya, penggunaan baterai indikator dan kontrol applet VPN. Fitur lainnya adalah galeri yang memungkinkan pengguna untuk memilih foto yang akan dihapus kamera, camcorder dan galeri yang dintegrasikan CDMA / EVDO, 802.1x, VPN, Gestures, dan *Text-to-speech engine*; kemampuan dial 10 kontak teknologi *text to change speech*[17].

4. Android versi 2.0 - 2.1 Eclair (API level 5- 7)

Pada 3 Desember 2009 kembali diluncurkan ponsel Android dengan versi 2.0/2.1 (*Eclair*), perubahan yang dilakukan adalah pengoptimalan *hardware*, peningkatan *Google Maps* 3.1.2, perubahan UI dengan *browser* baru dan dukungan HTML5, daftar kontak yang baru, dukungan *flash* untuk kamera 3,2 MP, digital *Zoom*, dan *Bluetooth* 2.1[17].

5. Android versi 2.2 – 2.2.3 Froyo :Frozen Yoghurt (API level 8)

Pada 20 Mei 2010, Android versi 2.2 (Froyo) diluncurkan. Perubahan-perubahan umumnya terhadap versi-versi sebelumnya antara lain 11 dukungan *Adobe Flash 10.1*, kecepatan kinerja dan aplikasi 2 sampai 5 kali lebih cepat, *intergrasi V8*[17].

6. Android versi 2.3 – 2.3.7 Gingerbread (API level 9 - 10)

Android *Gingerbread* di rilis pada 6 Desember 2010. Perubahan-perubahan umum yang didapat dari Android versi ini antara lain peningkatan kemampuan permainan (*gaming*), peningkatan fungsi *copy paste*, layar antar muka (*User Interface*) didesain ulang, dukungan format video VP8 dan WebM, efek audio baru (*reverb, equalization, headphone virtualization*, dan *bass boost*), dukungan kemampuan *Near Field Communication* (NFC), dan dukungan jumlah kamera yang lebih dari satu[17].

7. Android versi 3.0 – 3.2 Honeycomb (API level 11 - 13)

Android *Honeycomb* di rilis pada awal 2012. Merupakan versi Android yang dirancang khusus untuk device dengan layar besar seperti Tablet PC. Fitur baru yang ada pada Android *Honeycomb* antara lain yaitu dukungan terhadap *prosessormulticore*dangrafisdengan*hardwareacceleration. UserInterface* pada

Honeycomb juga berbeda karena sudah didesain untuk tablet. Tablet pertama yang memakai *Honeycomb* adalah tablet Motorola Xoom yang dirilis bulan Februari 2011. Selain itu sebuah perangkat keras produksi Asus bernama *Eee Pad Transformer* juga menggunakan OS Android *Honeycomb* dan diharapkan akan masuk ke pasaran Indonesia pada Mei 2011[17].

8. Android versi 4.0 – 4.0.4 Ice cream sandwich (API level 14 -15)

Android *Ice Cream Sandwich* diumumkan secara resmi pada 10 Mei 2011 di ajang *Google I/O Developer Conference* (San Francisco), pihak Google mengklaim Android *Ice Cream Sandwich* akan dapat digunakan baik di *smartphone* ataupun tablet. Android *Ice Cream Sandwich* membawa fitur *Honeycomb* untuk *smartphone* serta ada penambahan fitur baru seperti membuka kunci dengan pengenalan wajah, jaringan data pemantauan penggunaan dan *control*, terpadu kontak jaringan sosial, perangkat tambahan fotografi, mencari email secara offline[17].

9. Android versi 4.1 – 4.3 Jelly bean (API level 16 - 18)

Android *Jelly Bean* juga diluncurkan pada acara *Google* I/O 10 Mei 2011 yang lalu. Android versi ini membawa sejumlah keunggulan dan fitur baru, diantaranya peningkatkan *input keyboard*, desain baru fitur pencarian, UI yang baru dan pencarian melalui *Voice Search* yang lebih cepat. Versi ini juga dilengkapi *Google Now* yang dapat memberikan informasi yang tepat pada waktu yang tepat pula. Salah satu kemampuannya adalah dapat mengetahui informasi cuaca, lalu-lintas, ataupun hasil pertandingan olahraga. Sistem operasi Android *Jelly Bean* 4.1 pertama kali digunakan dalam produk tablet Asus, yakni *Google* Nexus7[17].

10. Android versi 4.4 kitkat (API level 19)

Beberapa waktu yang lalu *Google* secara resmi memperkenalkan sistem operasi terbarunya, yaitu update Sistem Operasi Android versi 4.4 yang diberi nama KitKat. Kabar tersebut cukup mengagetkan banyak pihak terlebih pengambilan nama untuk versi terbaru Android tersebut ternyata diluar perkiraan. Sebelumnya banyak kabar yang berhembus bahwa update sistem operasi Android terbaru yang akan diusung oleh *Google* akan dinamai Android *Key Lime Pie*, namun ternyata *Google* menepis semua rumor tersebut dengan memperkenalkan Android KitKat.

11. Android versi 5.0 Lollipop (API level 21)

Android 5.0 pertama kali diperkenalkan di bawah codename "Android L" pada 25 Juni 2014 selama presentasi keynote pada konferensi pengembang Google I / O. Di samping Lollipop, presentasi difokuskan pada sejumlah *platform* Android yang berorientasi dan teknologi baru, termasuk Android TV, pada *platform* Android Auto, dapat dipakai pada platform komputasi Android Wear, dan *platform* pelacakan kesehatan Google Fit.

12. Android versi 6.0 Marshmallow (API level 23)

Android Marshmallow memberikan dukungan asli untuk pengenalan sidik jari, memungkinkan penggunaan sidik jari untuk membuka perangkat dan otentikasi Play Store dan pembelian Android Pay; API standar juga tersedia untuk melaksanakan otentikasi berbasis sidik jari dalam aplikasi lain. Android Marshmallow mendukung USB Type-C, termasuk kemampuan untuk menginstruksikan perangkat untuk mengisi daya perangkat lain melalui USB. Marshmallow juga memperkenalkan "pranala yang diverifikasi" yang dapat dikonfigurasi untuk membuka langsung dalam aplikasi tertentu mereka tanpa petunjuk pengguna lanjut.

13. Android versi 7.0 – 7.1 Nougat (API level 24 – 25)

Android "Nougat" rilis 7.0 besar dari sistem operasi Android. Ini pertama kali dirilis sebagai pratinjau pengembang pada tanggal 9 Maret 2016, dengan gambar pabrik untuk perangkat Nexus saat ini, serta dengan "Program Beta" baru yang memungkinkan perangkat yang didukung ditingkatkan versinya ke versi Android Nougat melalui over-the-air update. Rilis terakhir adalah pada tanggal 22 Agustus 2016.

Pratinjau akhir pembuatannya dirilis pada tanggal 18 Juli 2016, dengan nomorNPD90G.Padatanggal 19 Oktober 2016, Google merilis Android 7.1 sebagai pratinjau pengembang untuk Nexus 5X, Nexus 6P dan Pixel C. Pratinjau kedua mulai tersedia pada 22 November 2016, sebelum versi final diluncurkan ke publik pada bulan Desember. 5, 2016.

14. Android versi 8.0 Oreo (API level 26)

Android Oreo adalah rilis utama ke 8 dari sistem operasi Android. Ini pertama kali dirilis sebagai preview pengembang pada tanggal 21 Maret 2017 untuk perangkat Nexus dan Pixel saat ini. Pratinjau pengembang terakhir dirilis pada tanggal 24 Juli 2017, dengan rilis stabil yang diharapkan pada bulan Agustus atau September 2017.

Arsitekur sistem operasi android disusun dalam beberapa lapisan perangkat lunak atau *software*. Terdapat empat lapisan dalam arsitektur android yang dimulai dari *Linux Kernel* sebagai lapisan terbawah, *Libraries* dan *Android Runtime*, lalu *Application Framework*, hingga lapisan paling atas yaitu *Applications*. Berikut penjelasan dari setiap lapisan yang digambarkan pada Gambar 2.3.

Sumber : https://developer.android.com/guide/platform/?hl=id

Gambar 2.3 Arsitektur Android

1. Linux Kernel

Linux Kernel yang berada pada bagian bawah lapisan perangkat lunak Android menyediakan suatu perantara bagi perangkat keras (*hardware*) dengan lapisan atas perangkat lunak Android. Pada Linux versi 2.6, Linux Kernel menyediakan *preemptive multitasking* layanan level dasar inti sistem seperti *memory*, *process*, dan *power management*. Kernel menyediakan juga suatu tingkatan jaringan dan *device drivers* untuk perangkat keras seperti monitor, Wi-Fi, dan audio.

2. Android Runtime – ART

Ketika suatu aplikasi Android dibangun dengan Android Studio, aplikasi dikompilasikan menjadi suatu *intermediate bytecode* yang kerap dirujuk sebagai DEX format. Saat aplikasi dijalankan pada suatu perangkat, Android Runtime (ART) menggunakna suatu proses yang disebut sebagai *Ahead-of-Time* (AOT) *compilation* untuk menerjemahkan *bytecode* tersebut menjadi perintah - perintah dasar yang dapat dieksekusi oleh *processor* perangkat.

3. Android *Libraries*

Android libraries diperuntukan khusus bagi pengembangan Android. *Application framework libraries* dan *libraries* lain yang mendukung pembuatan elemen antarmuka, gambar grafis, dan akses *database* merupakan contoh *libraries* yang masukk dalam kategori ini.

4. *Application Framework*

Application Framework merupakan sekumpulan layanan yang bersama – sama membentuk suatu lingkungan dimana aplikasi - aplikasi Android berjalan dan diatur. *Framework* ini menerapkan konsep saling berbagi dan saling melengkapi yang mana suatu aplikasi mampu membagikan kemampuan beserta dengan data yang terkait didalamnya untuk dapat digunakan oleh aplikasi lainnya.

5. *Applications*

Applications terdapat pada lapisan teratas dari susunan arsitektur Android. Pada lapisan ini yaitu seluruh aplikasi bawaan (*native applications*) yang telah dilengkapi dengan fitur khusus seperti aplikasi web browser dan email, serta apliaksi pihak ketiga (*third party applications*) yang diinstal oleh pengguna sendiri setelah membeli perangkat tersebut.

1.1.8. Android Studio

Android Studio adalah sebuah IDE untuk *Android Development* yang dikenalkan pihak *google* pada acara *Google I/O* di tahun 2013. *Android Studio* dibuat berdasarkan IDE Java populer, yaitu *IntelliJ IDEA. Android Studio* merupakan IDE resmi untuk pengembangan aplikasi Android. *Android Studio* dibangun dengan tujuan mempercepat proses pembuatan maupun pengembangan aplikasi Android yang berkualitas tinggi untuk setiap *device* Android[18].

Android studio menangani *Android Activity Lifecycle* yang di control oleh 7 method class Activity pada android[18]. 7 Method tersebut antara lain :

a. *OnCreate* : Terpanggil saat pertama kali aplikasi dijalankan.
b. *OnStart* : Terpanggil saat *activity* menjadi terlihat kepada *user*.
c. *OnResume* : Terpanggil saat *activity* akan mulai berinteraksi dengan *user*.
d. *OnPause* : Terpanggil saat *activity* menjadi tidak terlihat oleh *user*.
e. *OnStop* : Terpanggil saat *activity* tidak lagi berinteraksi dengan *user*.
f. *OnDestroy* : Terpanggil saat sebelum aplikasi ditutup.

Berikut ini adalah elemen dan *tools* yang tersedia untuk memudahkan pengembang untuk membuat sebuah aplikasi menggunakan Android Studio[19] :

1. *Menu Bar*

Menu Bar terletak pada bagian paling atas dari jendela kerja Android Studio yaitu menampilkan sekumpulan menu yang dapat dipilih, seperti *File, Edit, View* dan seterusnya. Dibawah menu bar terdapat *toolbar* yang dapat menampilkan *icon – icon shortcut* untuk melakukan aksi tertentu seperti *Open, Save, Run* dan sebaginya. Kemudian di bawah *toolbar* terdapat *Navigation bar* yang memperlihatkan lokasi *file* yang sedang dibuka dalam bentuk navigasi *folder* dan *file.*

2. *Document Tabs*

Android Studio mampu membuka sejumlah *file* secara bersamaan untuk diedit oleh pengembang. Saat membuka *file* baru, *document tab* yang baru akan dibentuk yang disertai dengan nama *file* tersebut.

3. *Project Window*

Project Window berguna untuk memperlihatkan susunan *folder* dan *file* yang digunakan dalam proyek dalam bentuk *tree* yang mudah ditelusuri.

4. *Bottom Sidebar*

Bottom Sidebar merupakan *sidebar* yang dapat dibuka untuk menampilkan informasi khusus yang dibutuhkan seperti jendela *Terminal, Messages, Event* Log dan sebagainya. Berfungsi untuk membantu dalam proses debugging aplikasi.

5. *Editor Area*

Editor Area merupakan area utama dimana kode program ditampilkan, diitulis dan diubah oleh pengembang. Secara *default, editor* akan menampilkan satu *panel* tunggal yang menyajikan isi dari *file* yang sedang dibuka.

6. *Editor Gutter Area*

Gutter area digunakan oleh editor untuk menampilkan infromasi *icons* dan *controls*. Beberapa informasi yang ditampilkan adalah *debugging breakpoint markers, boomarks,* dan *line numbers*.

7. *Validation and Marker Sidebar*

Pada bagian atas *validation sidebar* terdapat *icon* kecil yang dapat berubah warna dari hijau yaitu tidak ada peringatan atau kesalahan yang terdeteksi dalam file, kemudian kuning yaitu terdapat peringatan yang terdeteksi, dan merah yaitu terdapat kesalahan yang harus diperbaiki.

8. *Status Bar*

Status Bar yaitu terdapat informasi tentang status sesi *editing* yang sedang aktif. Informasi tersebut dapat berupa posisi *cursor* saat ini dan format *encoding file* yang dibuka.

1.1.9. Java

Java adalah salah satu bahasa pemrograman yang paling populer. Java dapat digunakan untuk berbagai hal, termasuk pengembangan perangkat lunak aplikasi *mobile*, dan pengembangan sistem yang besar. Inilah yang membuat bahasa pemrograman *Java* sangat terkenal di lingkungan pengembang perangkat lunak.

Seperti bahasa pemrograman lain, bahasa Java memiliki struktur sendiri, aturan sintaks, dan paradigma pemrograman. paradigma pemrograman bahasa Java didasarkan pada konsep OOP. Bahasa Java merupakan turunan bahasa C , sehingga aturan sintaks yang terlihat akan seperti bahasa C. Misalnya, blok kode yang modular dalam metode dan dibatasi oleh karakter '({' dan '})', dan variabel dideklarasikan sebelum digunakan. Secara struktural, bahasa Java diatur dengan package. Di Dalam *package* ada *class*, dan dalam *class* ada *method*, variabel, konstanta, dan banyak lagi.

1.1.10. OOP (*Object Oriented Programming*)

Object Oriented Programming adalah sebuah teknik pengembangan aplikasi yang berbasis pada objek-objek. Semua data dan fungsi pada OOP dibungkus dalam *class-class* atau objek-objek[20].OOP diciptakan untuk mengatasi keterbatasan pada Bahasa pemrogramman tradisional. Konsep dari OOP sendiri adalah semua pemecahan masalah dibagi kedalam objek. Semua aksi pada OOP berasal dari pengiriman pesan antar objek[21].

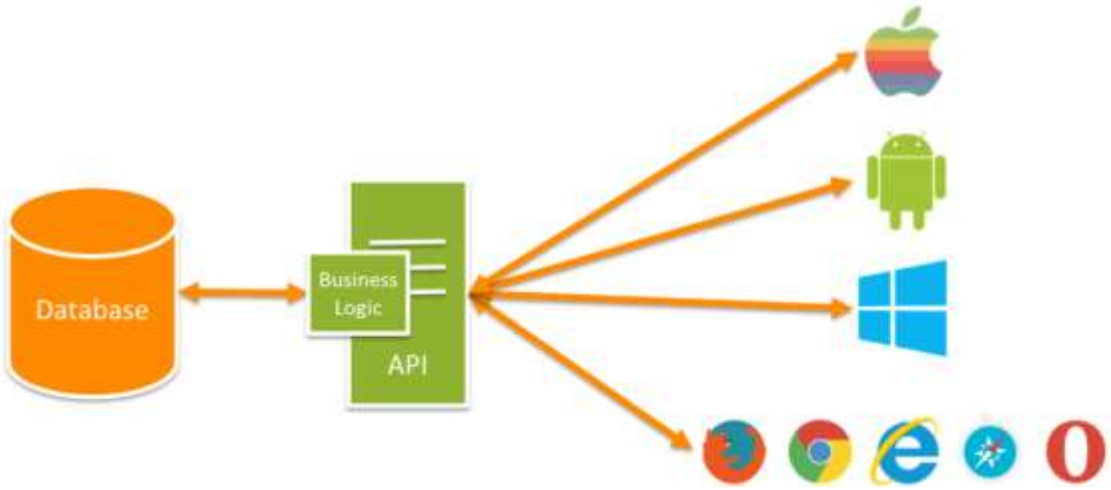
Sederhananya objek adalah kumpulan variabel dan fungsi yang dibungkus dalam satu entitas. Objek diciptakan melalui sebuah class atau dengan istilah *instance of class.* Elemen utama dari objek adalah *Attribute* dan *Method. Attribute* adalah nilai-nilai yang tersimpan pada objek tersebut. Sedangkan *method* adalah aksi yang bisa dilakukan oleh sebuah objek.

Class adalah struktur data atau *blueprint* suatu objek. Lebih jelasnya adalah sebuah bentuk dasar yang mendefinisikan variabel, method umum pada semua *object*. Kode-kode di dalam sebuah class erbagi menjadi dua kelompok, yaitu property dan *method*. *Property* adalah suatu wadah penyimpanan di dalam *class* yang bisa menampung informasi. Sederhananya property itu bisa disebut variabel didalam *class*. Sedangkan method adalah fungsi atau prosedur yang ada pada sebuah *class*.

1.1.11. API (*Application Programming Interface*)

API atau singkatan dari *Application Programming Interface* adalah sekumpulan fungsi, perintah dan protokol yang dapat menghubungkan satu aplikasi

dengan aplikasi lainnya agar dapat berinteraksi[22]. Seiring dengan perkembangan internet, API dapat diimplementasikan pada sisi server dan dapat digunakan oleh beberapa aplikasi yang terhubung ke server walaupun berbeda basis melalui protocol tertentu. Pada protocol HTTP, API umumnya disebut dengan *Web Application Programming Interface* atau *Web Service*[22]. Adapun cara kerja API secara sederhana ditunjukan oleh gambar 2.4 berikut :



Sumber gambar : https://www.codepolitan.com/mengenal-apa-itu-web-api-5a0c2855799c8

Gambar 2.4 Model kerja API

1.1.12. Google Maps API

Google Maps API adalah sebuah API yang disediakan oleh Google kepada para pengembang aplikasi untuk memanfaatkan fitur map pada Google Map dalam mengembangkan aplikasi. Google Maps API menyediakan beberapa fitur seperti : untuk memanipulasi peta, mencari lokasi dan menyimpan data lokasi / data peta pada aplikasi. Pengembang (*developer)* dapat memanfaatkan layanan-layanan yang ditawarkan oleh Google Maps setelah melakukan registrasi dan mendapatkan Google Maps API *Key*. Google menyediakan API ini secara gratis.

Google Maps API memungkinkan pengguna API ini untuk menambahkan beberapa grafik ke dalam Map. Grafik-grafik tersebut ialah :

a. Ikon pada posisi spesifik *(marker*).
b. Segmen set-set garis (*Polylines*).
c. *Enclosed Segment* (*Polygons*).
d. Grafis bitmap ditambatkan ke posisi tertentu di peta (*Ground Overlay*).
e. Set gambar yang ditampilkan di atas peta dasar (*Tile Overlay*).

Seperti yang tercatat oleh *Svennerberg, Google Maps API* adalah API yang paling populer di internet. Pencatatan yang dilakukan pada bulan Mei tahun 2010 ini menyatakan bahwa 43% mashup (aplikasi dan situs web yang menggabungkan dua atau lebih sumber data) menggunakan *Google Maps API*. Beberapa tujuan dari penggunaan *Google Maps API* adalah untuk melihat lokasi, mencari alamat, mendapatkan petunjuk mengemudi dan lain sebagainya [23].

1.1.13. Geofence

API geofencing memungkinkan pengembang untuk menentukan batas, juga disebut sebagai geofences, yang mengelilingi area yang diminati. Aplikasi mendapat pemberitahuan saat perangkat melintasi geofence, yang memungkinkan pengguna memberikan pengalaman yang bermanfaat saat pengguna berada di sekitarnya. Teknologi Geofence API ini secara otomatis menggunakan sensor-sensor pada *device* pengguna untuk mendeteksi lokasi keberadaan *device*.[23]

Sebagai contoh, aplikasi maskapai penerbangan dapat menentukan geofence di sekitar bandara ketika reservasi penerbangan mendekati waktu naik pesawat. Saat perangkat melintasi geofence, aplikasi dapat mengirim pemberitahuan yang membawa pengguna ke aktivitas yang memungkinkan mereka mendapatkan boarding pass pengguna.[23]

Geofence adalah sebauh API yang masih bagian dari API Location google. Termasuk di dalamnya yaitu Geofence, GeofenceRequest, GeofenceApi, GeofenceEvent, GeofenceCodes. Geofence merupakan antarmuka yang mewakili area yang dipantau.

Geofence dibuat dengan menggunakan *Geofence.Builder*. Selama pembangunan area geofence, pengembang harus menetapkan *latitude, longitude*, tanggal kadaluarsa area, sifat *responsive*, pengenal, dan jenis transisi yang digunakan.

Berikut ini adalah cara penerapan *Geofence.Builder*:

```
Geofence geofence = new Geofence.Builder()
    .setRequestId(GEOFENCE_REQ_ID) // Geofence ID
    .setCircularRegion( LATITUDE, LONGITUDE, RADIUS)
    .setExpirationDuration( DURANTION )
    .setTransitionTypes(
Geofence.GEOFENCE_TRANSITION_ENTER |
Geofence.GEOFENCE_TRANSITION_EXIT )
    .build();
```

Untuk menjaga konsumsi daya seminimal mungkin, disarankan untuk menggunakan geofence dengan radius minimal 100 meter untuk kebanyakan situasi. Jika geofences berada di pedesaan, Anda harus meningkatkan radius menjadi 500 meter atau lebih tinggi untuk memastikan geofences efektif.

- *Geofence_Transition_dwell* menunjukkan bahwa pengguna memasuki area tersebut dan menghabiskan beberapa waktu di sana. Hal ini berguna untuk menghindari banyak peringatan saat pengguna masuk dan keluar dari area terlalu cepat. Anda dapat mengkonfigurasi waktu tinggal dengan menggunakan parameter *setLoiteringDelay*.
- *Geofence_transition_enter* menunjukkan kapan pengguna memasuki wilayah yang dipantau.
- *Geofence_transsition_exit* menunjukkan kapan pengguna keluar dari wilayah ini.

1.1.14. Basis Data

Secara garis besar, Basis data atau *database* adalah kumpulan file yang berelasi satu sama lain. Relasi tersebut biasa ditunjukan dengan kunci (*key)* pada tiap file. Suatu basis data menunjukan kumpulan data dan file yang dipakai dalam satu lingkup informasi.

Dalam satu file terdapat field-field yang berisi data-data. Suatu system manajemen basis data berisi koleksi data yang saling berelasi dan satu set program untuk mengakses data tersebut. Jadi sistem manajemen basis data dan set program

pengelola untuk menambah data, menghapus data, mengambil data dan membaca data. [24]

1.1.15. DBMS

Database Management System (DBMS) telah menjadi alat mendasar untuk mengelola informasi, prinsip-prinsip dan praktik sistem database[25]. Pada intinya DBMS adalah sebuah perangkat lunak yang diarancang khusus untuk mengelola database serta menjalankan operasi terhadap data yang diminta oleh pengguna database.

Tujuan utama DBMS adalah untuk menghindarkan pengguna database dalam kekacauan dalam hal pengelolaan data dengan jumlah yang besar. Selain itu DBMS menjadi perantara antar pengguna dan database. Ada dua jenis bahasa komputer yang dapat digunakan dalam berinteraksi dengan DBMS, yaitu:

a. Data Definition Language (DDL) ; DDL digunakan untuk menggambarkan desain dari basis data secara keseluruhan, mulai dari membuat tabel baru, memuat indeks, maupun mengubah tabel.
b. Data Manipulation Language (DML) ; DML digunakan untuk memanipulasi dan mengambil data dari database, menghapus data dari database, dan mengubah data pada suatu database.

1.1.16. Firebase

Firebase adalah penyedia layanan *realtime database* dan *backend* sebagai layanan. Produk utama dari Firebase adalah suatu database yang menyediakan *API* untuk memungkinkan pengembang menyimpan, mensingkronkan data melalui *multiple client* yang disimpan di *cloud*-nya[26]. Firebase membuat sejumlah produk layanan untuk pengembangan aplikasi mobile maupun web seperti *authentication*, *storage*, *realtime database* dan lain-lain.

Berikut ini adalah fitur-fitur yang ada pada Firebase guna mendukung pembangunan aplikasi :

a. *Cloud* FireStore
b. ML (*Machine Learning) Kit*
c. Cloud Function

d. Authentication

e. Hosting

f. Cloud Storage

g. *Realtime Database*

Firebase memiliki banyak *library* yang memungkinkan untuk mengintegrasikan layanan ini dengan Android, Ios, Javacript, Java, Objective-C dan Node.JS[26]. Database Firebase pun bisa diakses melalui REST API. Pengembang menggunakan REST API untuk *post* data yang selanjutnya Firebase *client* yang sudah diterapkan pada aplikasi mengambil data secara realtime[26].

Layanan Firebase yang digunakan pada pembangunan sistem aplikasi ini adalah antar lain Real-time database, Authentication dan FCM (Firebase cloud messaging). Firebase real-time database peneliti pilih karena dinilai handal. Handal yang dimaksud adalah Firebase *real-time database* sudah dilengkapi dengan *event handler*, dimana setiap perubahan yang terjadi dalam database bisa segera dideteksi secara langsung dan ditangani, sehingga sangat cocok untuk menyimpan data yang kolaboratif seperti pada penelitian ini. Fitur lain yang digunakan pada penelitian ini adalah Authentication. Untuk masuk pengguna ke aplikasi, pengembang aplikasi pertama-tama mendapatkan kredensial otentikasi dari pengguna. Kredensial ini dapat berupa alamat email dan kata sandi pengguna, atau token OAuth dari penyedia identitas gabungan. Lalu, meneruskan kredensial ini ke Firebase Authentication SDK. Layanan backend Firebase kemudian akan memverifikasi kredensial tersebut dan mengembalikan respons kepada klien.

Setelah berhasil masuk, pengguna dapat mengakses informasi profil dasar pengguna, dan dapat mengontrol akses pengguna ke data yang disimpan dalam produk Firebase lainnya. Pengguna juga dapat menggunakan token otentikasi yang disediakan untuk memverifikasi identitas pengguna di layanan *backend* Firebase sendiri.

Firebase *realtime database* dilengkapi dengan fitur *online* dan *offline support.* Saat melakukan perubahan data saat koneksi ke Firebase tidak tersedia, maka secara otomatis perubahan tersebut disimpan secara *offline* pada

penyimpanan lokal. Perubahan pada server dilakukan saat pengguna kembali *online*. Semua itu dilakukan secara otomatis menggunakan *library* pada Firebase.

1.1.17. JSON

JSON adalah format penukaran data yang sederhana, bagi programer format ini mudah dibaca dan ditulis, sedangkan bagi mesin, format ini mudah untuk proses *parse* dan *generate*. JSON merupakan bagian dari JavaScript *Programming Language, Standard* ECMA-262 3rd *Edition - December* 1999. JSON merupakan format teks bahasa pemrograman yang berdiri sendiri namun menggunakan konvensi standar yang biasa digunakan oleh para programer bahasa pemrograman C, C++, C#, Java, JavaScript, Perl, Python, and many others. Hal ini menjadikan JSON sebagai bahasa penukaran data yang ideal [26]. JSON terbuat dari dua struktur:

1. Kumpulan pasangan nama/nilai. Pada beberapa bahasa, hal ini dinyatakan sebagai objek (*object*), rekaman (*record)*, struktur (*struct*), kamus (*dictionary*), tabel hash (*hash table*), daftar berkunci *(keyed list*), atau *associative array*.
2. Daftar nilai terurutkan (*an ordered list of values*). Pada kebanyakan bahasa, hal ini dinyatakan sebagai larik (*array*), vektor (*vector*), daftar (*list*), atau urutan (*sequence*).

Gambar 2.5 berikut ini adalah salah satu contoh bentuk Json :

```
{
    "glossary": {
        "title": "example glossary",
		"GlossDiv": {
            "title": "S",
			"GlossList": {
                "GlossEntry": {
                    "ID": "SGML",
                    "SortAs": "SGML",
                    "GlossTerm": "Standard Generalized Markup Language",
                    "Acronym": "SGML",
                    "Abbrev": "ISO 8879:1986",
                    "GlossDef": {
                        "para": "A meta-markup language, used to create markup languages such as DocBook.",
                        "GlossSeeAlso": ["GML", "XML"]
                    },
                    "GlossSee": "markup"
                }
            }
        }
    }
}
```

Sumber Gambar : http://json.org/example.html

Gambar 2.5 Contoh data JSON

1.1.18. Geo-Fire

Geo-Fire adalah sebuah *library open source* yang diperuntukan untuk Java/Android. *Library* ini memungkinkan dan mengizinkan pengguna untuk menyimpan dan meng-*query* set kunci-kunci berdasarkan dari lokasi geografik[28].

Pada dasarnya, *library* Geo-fire memudahkan untuk menyimpan lokasi-lokasi (*latitude, longitude*) menggunakan String *key* atau kunci. Keuntungan utama penggunaan *library* ini adalah memungkinkan mengolah *key* pada lokasi-lokasi spesifik secara real time[28]. Geo-Fire menyimpan data pengguna dalam formatnya sendiri dan lokasinya sendiri di dalam basis data Firebase. Ini memungkinkan format data dan aturan keamanan pengguna aplikasi tidak berubah serta mempermudah permintaan geografik tanpa mengubah data yang ada.

Objek Geo-Fire digunakan untuk membaca dan menulis data lokasi geo ke basis data dan untuk membentuk queri. Untuk membuat *instance* GeoFire baru, pengguna harus memasukannya ke dalam referensi basis dara Firebase. Berikut ini adalah sebuah contoh pembuatan *instance* GeoFire[28] :

```
DatabaseReference ref = FirebaseDatabase .  getInstance () .
getReference ( " path / to / geofire " );
 GeoFire geoFire = GeoFire new (ref);
```

Untuk pengaturan lokasi (*set location*) pada GeoFire, cukup dengan memanggil *method* ‘setLocation()’. *Method* ini meneruskan kunci sebagai String dan lokasi sebagai objeknya yang berisi koordinat lintang (*longitude*) dan bujur (*latitude*). Adapun untuk menghapus lokasi dan menghapusnya dari basis data, pengguna cukup menggunakan *method* ‘removeLocation()’ kunci lokasi ke removelocation. Berikut ini adalah contoh dari kedua method tersebut :

```
geoFire .  setLocation ( " firebase-hq " , GeoLocation new ( 37.7853889
, - 122.4056973 ));

geoFire .  removeLocation ( " firebase-hq " );
```

GeoFire memungkinkan pengguna untuk melakukan queri semua kunci dalam area geografis menggunakan objek GeoQuery. Saat lokasi untuk kunci berubah, kueri diperbarui dalam waktu *real-time* dan memunculkan *event* yang memberi tahu pengguna jika ada kunci yang relevan telah

dipindahkan. Parameter GeoQuery dapat diperbarui untuk mengubah ukuran dan pusat area yang diminta[28]. Berikut ini adalah contoh dari geoQuery :

```
GeoQuery geoQuery = geoFire .  queryAtLocation (GeoLocation new (
37.7832 , - 122.4056 ), 0.6 );
```

Berikut ini adalah 5 *Key Events* untuk menerima *event* untuk Geo Query[28].

1. ***Key Entered***: Lokasi kunci saat cocok dengan kriteria query.
2. ***Key Exited***: Lokasi kunci saat sudah tidak cocok dengan kriteria query (keluar dari kriteria query).
3. ***Key Moved***: Lokasi kunci berubah, namun lokasi masih di dalam atau masih cocok dengan kriteria query.
4. ***Query Ready***: Semua data saat ini telah diambil dari server.
5. ***Query Error***: Adanya suatu kesalahan dalam melakukan query lokasi. Contohnya adanya pelanggaran keamanan yang terjadi.

Selain hal-hal diatas, pada GeoFire pun menyediakan *event* data. Maksudnya, jika pengguna menyimpan data model dan data geo di lokasi basis data yang sama, Pengguna mungkin ingin mengakses *DataSnapshot* sebagai bagian dari *event* geo. Dalam hal ini, bisa menggunakan GeoQueryDataEventListener daripada *listener* kunci.

Listener "data event" ini memiliki semua *event* yang sama dengan *listener* kunci dengan satu jenis acara tambahan. DataSnapshot mendasarinya data telah berubah. Setiap *event* "data dipindahkan" diikuti oleh *event* yang diubah data tetapi pengguna juga bisa mendapatkan perubahan *event* tanpa bergerak jika data yang diubah tidak memengaruhi lokasi.Menambahkan *event listener* data mirip dengan menambahkan *listener event* utama. Berikut ini adalah caranya:

```
  geoQuery .  addGeoQueryDataEventListener (GeoQueryDataEventListener
baru () {
}
```

1.1.19. GPS

Global Positioning System (GPS) adalah system satelit navigasi dan penentuan posisi yang dimiliki dan dikelola oleh Amerika Serikat[18]. Sistem ini didesain untuk memberikan posisi dan kecepatan tiga-dimensi serta informasi mengenai waktu, secara terus menerus di seluruh dunia tanpa bergantung waktu dan cuaca, bagi banyak orang secara simultan.

Dalam menentukan lokasi dari perangkat Android, ada beberapa cara yang digunakan dalam memperoleh data tersebut. Cara yang digunakan sebagai berikut:

a. GPS Provider

   Dalam menentukan posisi pengguna applikasi atau pengguna ponsel, perangkat harus terhubung dengan satelit.

b. Network Provider

   Saat pengguna tidak terhubung dengan satellite, maka secara otomatis perangkat akan mencari posisi *base Transceiver Station* dari Network provider perangkat tersebut.

Pada pembangunan aplikasi pada penelitian ini sangat dibutuhkannya teknologi GPS. *Global Positioning System* adalah sistem yang memungkinkan pengguna untuk menentukan lokasi akurat menggunakan satelit. Secara umum, semua satelit navigasi dinamakan GNSS (*Global Navigation Satelite System*).

Ponsel pintar pengguna adalah sebuah *receiver*, atau alat penerima sinyal satelit-satelit GNSS. Secara umum *receiver* ini disebut GPS *receiver,* karena GPS adalah rangkaian satelit navigasi yang paling umum dikenal dan paling pertama dipopulerkan.

1.1.20. Kamus Data

Kamus data adalah suatu Teknik untuk menjelaskan atau mendeskripsikan kolom-kolom pada masing-masing tabel pada *database* yang dibuat. Dengan menggunakan kamus data, analisis system dapat mendefinisikan data yang mengalir berisi informasi tentang struktur *database* tersebut.

1.1.21. Alat Pemodelan Sistem

Alat-alat permodelan sistem informasi sangat dibutuhkan dalam proses

analisis dan perancangan sistem. Pemodelan system antara lain sebagai berikut :

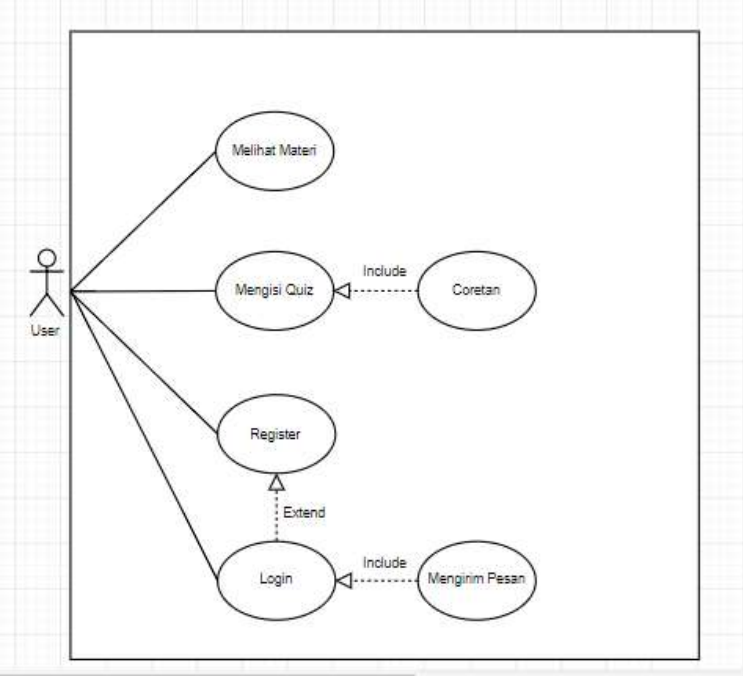
1.1.22. UML (*Unified Modelling Language*)

Unified Modelling Language (UML) adalah sebuah tool atau alat yang telah menjadi standar dalam industri untuk visualisasi, merancang dan mendokumentasikan sistem piranti lunak. UML memberikan sebuah standar untuk merancang model sebuah sistem. Dengan menggunakan UML dapat dibuat model untuk semua jenis aplikasi piranti lunak, dimana aplikasi tersebut dapat berjalan pada piranti keras, sistem operasi dan jaringan apapun, serta ditulis dalam bahasa pemrograman apapun. Tetapi karena UML juga menggunakan *class* dan *operation* dalam konsep dasarnya, maka lebih cocok untuk penulisan piranti lunak dalam bahasa berorientasi objek seperti C++, Java, atau VB. NET [29]. Unified Modeling Language (UML) bisa juga berarti notasi grafis untuk menggambar diagram konsep perangkat lunak. Satu dapat menggunakannya untuk menggambar diagram dari domain masalah, desain perangkat lunak yang diusulkan, atau implementasi perangkat lunak yang sudah selesai.

1.1.23. Use Case

Use case Digaram menjelaskan system digunakan dan merupakan titik awal dari pemodelan UML. Use case merupakan suatu pemodelan untuk menggambarkan aktifitas-aktifitas sistem yang akan dibuat. Selain itu use case mendeskripsikan apa yang sistem lakukan tanpa mendeskripsikan bagaimana sistem menyelesaikannya.

Usecase diagram merupakan bagian tertinggi dari fungsionalitas yang dimiliki sistem yang menggambarkan bagaimana seseorang atau aktor dalam menggunakan dan memanfaatkan sistem. Use case terdiri dari tiga bagian yaitu identifikasi aktor, identifikasi use case, dan skenario use case [29]. Contoh bagian-bagian Use case dapat dilihat pada gambar 2.6 berikut :

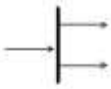
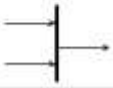


Gambar 2.6 Contoh Use Case

1.1.24. Activity Diagram

Activity diagram memodelkan alur kerja (*workflow*) sebuah proses bisnis dan urutan aktivitas dalam suatu proses. Mempresentasikan alur dari aktifitas yang terjadi pada suatu usecase. Aktifitas yang dimaksud adalah kumpulan dari aksi-aksi. Diagram aktifitas terdiri dari kumpulan aksi, sub aktifitas, dan transisis. Satu diagram hanya mempunyai 1 inisial state dan 1 atau lebih end state.

Activity *diagram* juga sangat berguna ketika ingin menggambarkan perilaku paralel atau menjelaskan bagaimana perilaku dalam berbagai use case berinteraksi. Dapat digunakan statechart diagram untuk memodelkan perilaku dinamis satu kelas atau objek. Statechart diagram memperlihatkan urutan keadaan sesaat (*state*) yang dilalui sebuah objek, kejadian yang menyebabkan sebuah transisi dari satu state atau aktivitas ke state atau aktivitas lainnya, dan aksi yang menyebabkan perubahan satu state lainnya, dan aksi yang menyebabkan perubahan satu state atau aktivitas. Diagram aktivitas paling cocok digunakan untuk memodelkan urutan aktivitas dalam suatu proses [29]. Gambar 2.7 berikut adalah symbol dan penjelasan dari symbol diagram *activity* :

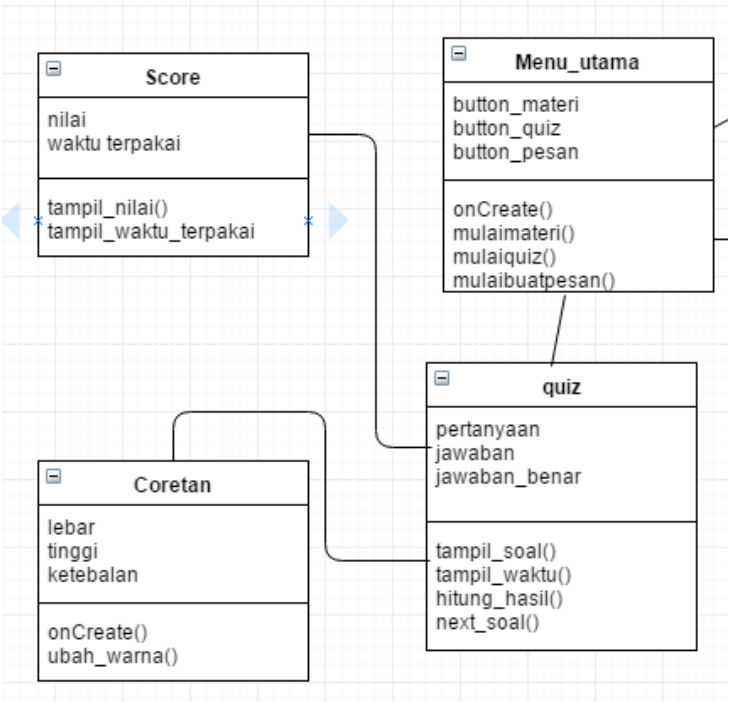
Simbol	Keterangan
	Start Point
	End Point
	Activities
	Fork (Percabangan)
	Join (Penggabungan)
	Decision/Split Merge
Swimlane	Sebuah cara untuk mengelompokkan activity berdasarkan Actor (mengelompokkan activity dalam sebuah urutan yang sama)

Sumber gambar: https://correctmy.com

Gambar 2.7 Simbol *Activity Diagram*

1.1.25. Class Diagram

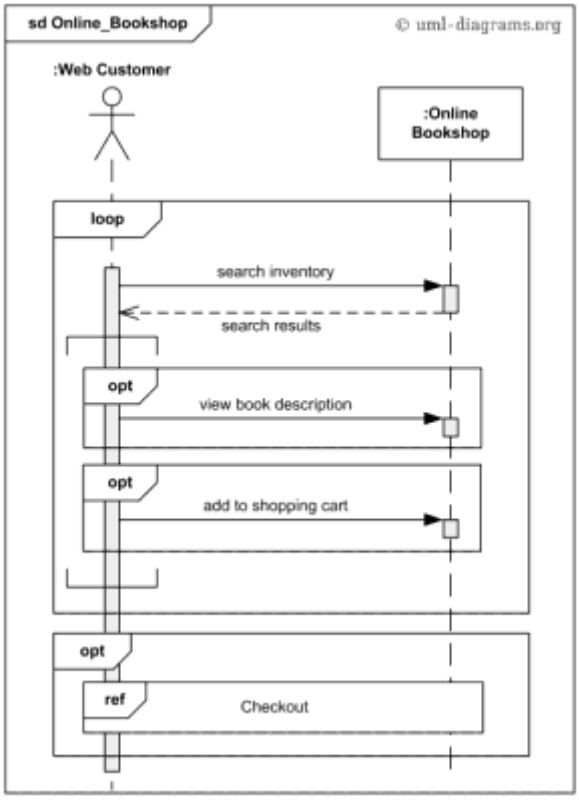
Class diagram membantu dalam visualisasi struktur kelas – kelas dari suatu sistem dan merupakan tipe diagram yang paling banyak. *Class diagram* memperlihatkan hubungan antar kelas dan penjelasan detail tiap – tiap kelas di dalam model desain (dalam *logical view*) dari suatu sistem. Selama proses analisi, *class diagram* memperlihatkan aturan – aturan dan tanggung jawab entitas yang menentukan perilaku sistem. Selama proses analisis, *class diagram* memperlihatkan aturan – aturan dan tanggung jawab entitas yang menentukan perilaku sistem. Selama tahap decían, *class diagram* berperan dalam menangkap struktur dari semua kelas yang membentuk arsitektur sistem yang dibuat [29]. Gambar 2.8 berikut ini adalah contoh kecil *class diagram* :



Gambar 2.8 Contoh *Class Diagram*

1.1.26. Sequence Diagram

Diagram sequence menjelaskan interaksi objek yang disusun dalam suatu urutan waktu. Diagram ini secara khusus berasosiasi dengan *usecase*. *Sequence diagram* memperlihatkan tahap demi tahap apa yang seharusnya terjadi untuk menghasilkan sesuatu di dalam use case. Diagram sequen sebaiknya digunakan diawal tahap desain atau analisis karena kesederhanaannya dan mudah untuk dimengerti [29]. Gambar 2.9 berikut ini adalah contoh dari sequence diagram :



Sumber gambar : https:// www.uml-diagrams.org/online-shopping-uml-sequence-diagram-example.html?context=seq-examples

Gambar 2.9 Contoh *Sequence Diagram*

1.1.27. Pengujian Sistem

Pengujian adalah suatu proses pelaksanaan suatu program dengan tujuan menemukan suatu kesalahan. Suatu kasus test yang baik adalah apabila test tersebut mempunyai kemungkinan menemukan sebuah kesalahan yang tidak terungkap. Suatu test yang sukses adalah bila test tersebut membongkar suatu kesalahan yang awalnya tidak ditemukan. [31]

1.1.28. Pengujian Alpha

Pengujian ini termasuk didalamnya adalah *Black Box* dan *White Box* . *Black Box Testing* berfokus pada spesifikasi fungsional dari perangkat lunak. Tester dapat mendefinisikan kumpulan kondisi input dan melakukan pengetesan pada spesifikasi fungsional program[31]. Black Box Testing bukanlah solusi alternatif dari *White*

Box Testing tapi lebih merupakan pelengkap untuk menguji hal-hal yang tidak dicakup oleh *White Box Testing*. *Black Box* Testing cenderung untuk menemukan hal-hal berikut:

1. Fungsi yang tidak benar atau tidak ada.
2. Kesalahan antarmuka (interface errors).
3. Kesalahan pada struktur data dan akses basis data.
4. Kesalahan performansi (performance errors).
5. Kesalahan inisialisasi dan terminasi.

1.1.29. Pengujian Beta

Pengujian Beta dilakukan dengan menerima langsung respon dari pengguna mengenai kepuasan, keluhan, bahkan saran atas aplikasi yang telah pengguna gunakana. Metode pengujian bisa menggunakan kuisioner kepada para pengguna ataupun wawancara dan rating dan komentar pada aplikasi yang telah dibangun.

Hasil pengujian beta ini dihitung menggunakan skala *Likert*. Skala *Likert* kerap digunakan sebagai skala penilaian karena memberi nilai terhadap sesuatu. Skala *likert* adalah suatu skala psikometrik yang umum digunakan dalam kuesioner, dan skala yang paling banyak digunakan dalam riset berupa survey. [32] Skala *likert* mempunyai empat atau lebih butir-butir pertanyaan yang dikombinasikan sehingga membentuk sebuah skor/ nilai yang merepresentasikan sifat individu, misalkan pengetahuan, sikap, dan perilaku [32] .Untuk keperluan analisis kuantitatif [32], skala jawaban pada skala *likert* dapat diberi skor misalnya:

1. Sangat setuju (SS) diberi skor 5
2. Setuju (S) deiberik skor 4
3. Ragu-ragu (RG) diberi skor 3
4. Tidak setuju (TS) diberi skor 2
5. Sangat tidak setuju (STS) diberi skor 1