BAB II

TINJAUAN PUSTAKA

2.1. Usia Lanjut

Usia Lanjut merupakan anugerah. Menjadi tua dengan segenap keterbatasannya pasti akan dialami oleh seseorang bila ia panjang umur. Di Indonesia, istilah untuk kelompok lanjut usia ini belum baku, orang memiliki sebutan yang berbeda-beda. Ada yang menggunakan istilah lanjut usia ada pula usia lanjut atau jompo. Di Indonesia telah disetujui bahwa penduduk lanjut usia adalah mereka yang berumur 60 tahun keatas. Sesuai Undang-undang nomor 13 tahun 1998 pasal 1 ada di muatkan mengenai pengertian lanjut usia yaitu seseorang yang telah mencapai usia 60 tahun keatas (Hidir & Aisyah, 2014).

2.2. Android

Android merupakan *Operating System (OS) Mobile* yang berkembang dewasa ini.OS lainnya seperti Windows Mobile, Symbian OS, iOS, dan masih banyak lagi juga menawarkan keoptimalan berjalan di atas *hardware* yang ada. Akan tetapi OS yang ini berjalan dengan memprioritaskan aplikasi inti yang dibangun sendiri tanpa melihat potensi yang cukup besar dari aplikasi pihak ketiga (Nazaruddin, 2014).

Berikut fitur-fitur yang ada pada Android adalah :

1. *Framework* Aplikasi

   Memungkinkan penggunaan dan pemindahan dari komponen yang tersedia.
2. Dalvik *Virtual Machine*

   Virtual Machine untuk pengoptimalan perangkat *mobile*.
3. Grafik

   Grafik 2D dan 3D yang menggunakan *library* OpenGL.
4. SQLite

Untuk penyimpanan data.

5. Mendukung Media

   Audio, video, dan berbagai format gambar (MPEG4, , MP3, H.264, AAC, AMR, JPG, GIF, PNG).

6. GSM, Bluetooth, EDGE, 3G dan Wi-Fi (tergantung *hardware).*
7. Kamera, *Global Positioning System* (GPS), kompas dan *accelerometer* (tergantung *hardware*).Lingkungan pengembangan yang kaya seperti emulator, *debugging*, dan *plugin eclipse* IDE.

1.3. Basis Data

Basis data atau database merupakan kumpulan data satu dengan data lainnya yang tersimpan dalam satu tempat penyimpanan luar dan membutuhkan suatu perangkat lunak untuk menjalankannya (Edhy, 2012). Elemen-elemen basis data antara lain :

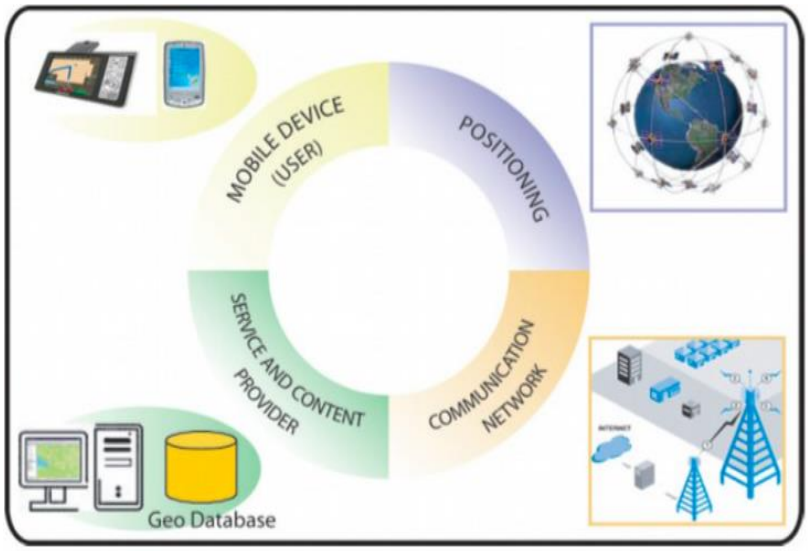
1. Entitas kumpulan dari beberapa objek yang memiliki karakter sama namun bisa di bedakan satu dengan lain nya. Contoh objek nya berupa orang, barang atau tempat.
2. Atribut unit terkecil dalam data yang mempunyai relasi dan makna bagi pengguna atau *user.*
3. Data *value* data atau informasi yang tersimpan pada tipe data, atribut, dan elemen. Contoh nilai data yaitu atribut nama siswa.
4. *Field*/tabel kumpulan karakter yang di bentuk dalam satu arti. Jika field itu terdapat seperti nama barang atau nomor barang, maka *field* yang terdapat harus berhubungan dengan atau berkaitan dengan nama dan nomor barang tersebut.
5. *Record* ialah kumpulan dari beberapa *field* yang lengkap, dan di hitung dalam bentuk satuan baris.

1.4. *Location Based Service* (LBS)

Location based service adalah layanan informasi yang di akses menggunakan piranti *mobile* melalui jaringan internet dan seluler serta memanfaatkan kemampuan penunjuk lokasi pada piranti *mobile*. Konsep

dari metode *Location Based Service* ini sendiri menggunakan database informasi geografis yang digabungkan dengan teknologi *Global Positioning System* (GPS) yang tertanam di *smartphone* pengguna untuk melacak suatu pergerakan *device* pengguna dan mengirimkan informasi yang dibutuhkan oleh *device* pengguna. (Susanty, Astari, & Thamrin, 2019).

1. *Mobile Device* yaitu sebuah alat yang digunakan oleh pengguna untuk meminta informasi yang dibutuhkan. Perangkat memungkinkan yaitu perangkat yang memiliki fasilitas navigasi seperti PDA, *mobile phone*, laptop dan lainnya.
2. *Communication Network* adalah jaringan selular yang mengirimkan data pengguna dan permintaan layanan.
3. *Positioning Component* biasanya posisi pengguna harus ditentukan untuk pengolahan layanan. Posisi pengguna dapat diperoleh menggunakan jaringan komunikasi atau dengan menggunakan *Global Positioning System* (GPS).
4. *Service and Content Provider* yaitu penyedia layanan informasi data yang dapat di minta oleh pengguna. Komponen LBS dapat ditunjukan pada gambar berikut:



Gambar 2. 1 Komponen Location Based Service

a. Unsur Utama pada *Location Based Service* (LBS)

Location Based Service (LBS) memiliki unsur utama yaitu :

1. *Location* (API *Maps*) menyediakan perangkat bagi sumber atau *source* untuk *location based service* (LBS), *Application Programming Interface* (API) *maps* menyediakan fasilitas untuk menampilkan dan memanipulasi peta.
2. *Location Provider* (API *Location*) menyediakan teknologi pencarian lokasi yang digunakan oleh perangkat. API *Location* berhubungan dengan data GPS (*Global Positioning System*) dan data lokasi *real-time*. API *Location* berada pada data android yaitu data paket internet yang digunakan oleh perangkat.

b. Cara akses layanan *Location Based Service* (LBS)

Pada *platform* ada dua cara yang berbeda untuk mengakses layanan LBS:

1. Inisiatif dari *platform*:

   Pengguna mengirimkan permintaan (teks) untuk informasi tentang layanan di daerah dekat sekitarnya.

2. Inisiatif dari pengguna

   Pengguna register terlebih dahulu untuk menerima tertentu informasi setiap kali dekat dengan tempat pengguna. Pengguna menerima diminta informasi pada item baru apabila di dekat tempat sekitar tersebut.

1.5. *Global Positioning System* (GPS)

Global Positioning System (GPS) merupakan suatu kumpulan satelit dan sistem kontrol yang memungkinkan sebuah penerima GPS untuk mendapatkan lokasinya di permukaan bumi 24 jam sehari. Sistem ini menggunakan sejumlah satelit yang berada di orbit bumi, yang memancarkan sinyal ke bumi dan di tangkap oleh sebuah alat penerima. *Global Positioning System* (GPS) adalah sistem untuk menentukan posisi di permukaan bumi dengan bantuan singkronisasi sinyal satelit. Sistem ini menggunakan minimal 4 satelit yang mengirimkan gelombang mikro ke bumi. Sinyal ini di terima oleh alat penerima di permukaan dan di gunakan

untuk menentukan posisi, kecepatan, arah dan waktu (Susanty, Astari, & Thamrin, 2019).

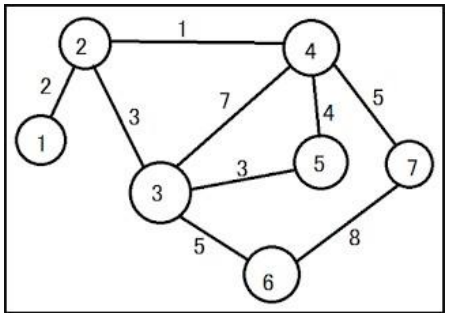
1.6. Algoritma Dijkstra

Algoritma Dijkstra merupakan algoritma yang lebih efisien dibandingkan algoritma Warshall untuk mencari lintasan terpendek, meskipun implementasinya juga lebih sukar.

Algoritma Dijkstra dinamai menurut penemunya, seorang ilmuwan komputer, Edsger Dijkstra adalah sebuah algoritma rakus (*greedy algorithm*) yang dipakai dalam memecahkan permasalahan jarak terpendek (*shortest path problem*) untuk sebuah graf berarah (*directed graph*) dengan bobot-bobot sisi (*edge weights*) yang bernilai tak-negatif(Wikipedia).

Misalnya, bila verteks dari sebuah graf melambangkan kota-kota dan bobot sisi (*edge weights*) melambangkan jarak antara kota-kota tersebut, maka algoritma Dijkstra dapat digunakan untuk menemukan jarak terpendek antara dua kota.

Berikut contoh pencarian jalur terpendek, dengan node 1 sebagai titik awal :



Gambar 2. 2 Contoh Jalur Pencarian Dengan Algoritma Dijkstra

Contoh penerapan dibuat dalam bentuk tabel agar mudah dalam melihat hasil jalur terpendek. Notasi dalam tabel algoritma Dijkstra memiliki format (s-j,D), dimana s-j menunjukkan rute dari node s menuju node j, sementara D menunjukkan jarak total antara kedua node tersebut.

Baris pertama masih berupa inisialisasi, yaitu Dj akan memiliki nilai jika tersambung langsung dan tidak memiliki nilai jika tidak tersambung

langsung. Karena node 1 hanya memiliki 1 tetangga yaitu node 2, maka i = 2 dimasukkan pada himpunan N.

Node 2 sudah berperan sebagai perpanjangan node sumber (node 1), sehingga sekarang node-node yang terhubung dengan node 2 sudah bisa dijangkau oleh node 1 via node 2. Diketahui node 3 dan node 4 terhubung langsung dengan node 2, sehingga rutenya ditulis (1-2-3) dan (1-2-4).

Untuk langkah selanjutnya dipilih node i yang telah tersambung dengan node s namun belum masuk dalam himpunan N. Diketahui yaitu node 3 dan node 4. Node yang dipilih adalah yang memiliki jumlah jarak yang paling minimum, yaitu node 4.

Seperti langkah yang sebelumnya, sekarang node 4 ikut berperan sebagai perpanjangan dari node 1 sehingga node 5 dan node 7 yang terhubung langsung dengan node 4 sudah bisa dijangkau oleh node 1 dengan rute yang tertampil.

Sebenarnya disini mulai terjadi perbandingan nilai jarak. Dengan dimasukkannya node 4 dalam himpunan N, maka node 4 dapat melakukan perhitungan minimum menuju node tertentu meskipun node tersebut telah diketahui rute dan jaraknya.

Dalam kasus ini dapat diambil node 3. Node 3 sudah diketahui rute dan jaraknya pada baris ke-2. Dengan masuknya node 4 pada himpunan N, maka node 4 akan menghitung jarak minimum antara *entry* D3 yang terdahulu dengan yang baru. Berikut perbandingan via node 2 dengan via node 4:

Via node 2 --> D2 + C23 = 2 + 3 = 5

Via node 4 --> D4 + C43 = 3 + 7 = 10

Maka *entry* yang dipertahankan adalah via node 2 dengan jarak 5. Sebelumnya telah dipilih node 4 sebagai anggota N karena bertetangga dengan node 2. Sekarang dipilih tetangga node 2 yang lainya yaitu node 3. Perbandingan yang terjadi:

Pada D4

Via node 2 --> D2 + C24 = 2 + 1 = 3

Via node 3 --> D3 + C34 = 5 + 7 = 12

Maka *entry* yang dipertahankan adalah via node 2 dengan jarak 3.

Pada D5

Via node 4 --> D4 + C45 = 3 + 4 = 7

Via node 3 --> D3 + C35 = 5 + 3 = 8

Maka entry yang dipertahankan adalah via node 4 dengan jarak 7.

Sehingga table hasil akhirnya sebagai berikut:

Tabel 2. 1 Tabel Contoh Pencarian Rute Terpendek

N	D2	D3	D4	D5	D6	D7
{1}	(1-2, 2)	∞	∞	∞	∞	∞
{1,2}	(1-2, 2)	(1-2-3, 5)	(1-2-4, 3)	∞	∞	∞
{1,2,4}	(1-2, 2)	(1-2-3, 5)	(1-2-4, 3)	(1-2-4-5, 7)	∞	(1-2-4-7, 8)
{1,2,3,4}	(1-2, 2)	(1-2-3, 5)	(1-2-4, 3)	(1-2-4-5, 7)	(1-2-3-6, 10)	(1-2-4-7, 8)
{1,2,3,4,5}	(1-2, 2)	(1-2-3, 5)	(1-2-4, 3)	(1-2-4-5, 7)	(1-2-3-6, 10)	(1-2-4-7, 8)
{1,2,3,4,5,7}	(1-2, 2)	(1-2-3, 5)	(1-2-4, 3)	(1-2-4-5, 7)	(1-2-3-6, 10)	(1-2-4-7, 8)
{1,2,3,4,5,6,7}	(1-2, 2)	(1-2-3, 5)	(1-2-4, 3)	(1-2-4-5, 7)	(1-2-3-6, 10)	(1-2-4-7, 8)

Dalam penerapannya terhadap bumi, formula ini harus dikalikan dengan jarijari dari lingkaran bumi yang nilainya 6371 km. Untuk nilai latitude dan longitude yang berbentuk derajat desimal maka harus di udah menjadi radians dengan cara mengkalikan nilai latitude dan longitude dengan 1 derajat atau 0.01745329251994 rad.

$r = 6371\ km$

Jarak Terdekat = 6371 * ACOS(SIN(RADIANS(lati)) *

SIN(RADIANS($lat)) + COS(RADIANS(longi - $lng))*

COS(RADIANS(lati)) * COS(RADIANS($lat))

1.7. Perangkat Lunak Pendukung

Berikut adalah perangkat lunak pendukung dalam penunjang pembangunan aplikasi yang akan di bangun.

2.7.1. Android Studio

Android studio adalah IDE (*Integrated Development Environment*) resmi untuk pengembangan aplikasi *Androi*d dan bersifat *open source* atau gratis. Peluncuran *Android Studio* ini diumumkan oleh *Google* pada 16 mei 2013 pada *event Google* I/O *Co*nference untuk tahun 2013. Sejak saat itu, *Android Studio* mengantikan *Eclipse* sebagai IDE resmi untuk mengembangkan aplikasi *Android* (Juansyah, 2015).

Android studio sendiri dikembangkan berdasarkan *IntelliJ* IDEA yang mirip dengan *Eclipse* disertai dengan ADT *plugin* (*Android Development Tools*). *Android studio* memiliki fitur :

a. Projek berbasis pada *Gradle Build*
b. *Refactory* dan pembenahan bug yang cepat
c. *Tools* baru yang bernama "*Lint*" dikalim dapat memonitor kecepatan, kegunaan, serta kompetibelitas aplikasi dengan cepat.
d. Mendukung *Proguard And App-signing* untuk keamanan.
e. Memiliki GUI aplikasi android lebih mudah
f. Didukung oleh *Google Cloud Platfrom* setiap aplikasi yang dikembangkan.

2.7.2. *Java Development Kit* (JDK)

Java Development Kit (JDK) adalah sekumpulan perangkat lunak yang dapat kamu gunakan untuk mengembangkan perangkat lunak yang berbasis Java, sedangkan JRE adalah sebuah implementasi dari *Java Virtual Machine* yang benar-benar digunakan untuk menjalankan program java. Baisanya, setiap JDK berisi satu atau lebih JRE dan berbagai alat pengembangan lain seperti sumber *compiler java, bundling, debuggers, development libraries* dan lain sebagainya (Juansyah, 2015).

2.7.3. *Google Maps* API

Google Maps API adalah layanan berbasis web yang menyediakan informasi rinci suatu wilayah geografis dan situs di seluruh dunia. Selain peta jalan konvensional, *Google Maps* memperlihatkan di udara dengan satelit dari banyak tempat. Di beberapa kota, *Google Maps* menawarkan pemandangan jalan yang terdiri foto yang diambil dari kendaraan (Ariyanti, Khairil, & Kanedi, 2015).

1.8. Metode Pengumpulan Data

Teknik pengumpulan data merupakan faktor penting demi keberhasilan penelitian. Hal ini berkaitan dengan bagaimana cara mengumpulkan data, siapa sumbernya, dan apa alat yang digunakan (Hendryadi, 2014).

1. Observasi

Obrservasi merupakan salah satu teknik pengumpulan data yang tidak hanya mengukur sikap dari responden (wawancara) namun juga dapat digunakan untuk merekam berbagai fenomena yang terjadi (situasi, kondisi). Teknik ini digunakan bila penelitian ditujukan untuk mempelajari perilaku manusia, proses kerja, gejala-gejala alam dandilakukan pada responden yang tidak terlalu besar.

2. Wawancara

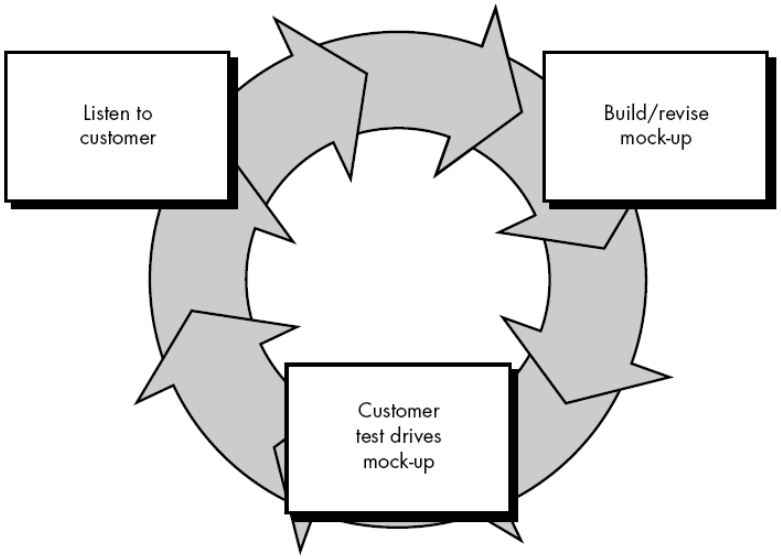
Wawancara merupakan teknik pengumpulan data yang dilakukan melalui tatap muka dantanya jawab langsung antara pengumpul data maupun peneliti terhadap nara sumber atausumber data.

3. Kuesioner

Kuesioner adalah daftar pertanyaan tertulis yang telah disusun sebelumnya. Pertanyaanpertanyaan yang terdapat dalam kuesioner, atau daftar pertanyaan tersebut cukup terperincidan lengkap dan biasanya sudah menyediakan pilihan jawaban (kuesioner tertutup) atau memberikan kesempatan responden menjawab secara bebas (kuesioner terbuka).

1.9. Metode Pengembangan Sistem

Prototype merupakan metodologi pengembangan *software* yang menitik pada pendekatan aspek desain, fungsi dan *user-interface*. *Developer* dan *user* fokus pada *user-interface* dan bersama-sama mendefisikan spesifikasi, fungsi, desain dan bagaimana *software* bekerja. *Developer* dan *user* bertemu melakukan komunikasi dan menentukan tujuan umum, kebutuhan yang diketahui dan gambaran bagian-bagian yang akan dibutuhkan. *Developer* mengumpulkan detail dari kebutuhan dan memberikan suatu gambaran dengan cetak biru (*prototype*). Dari proses tersebut akan diketahui detail-detail yang harus dikembangkan atau ditambahkan oleh *developer* terhadap cetak biru, atau menghapus detail-detail yang tidak diperlukan oleh *user*. Proses akan terjadi terus menerus sehingga produk sesuai dengan keinginan dari *user*.



Gambar 2. 3 Metode Pengembangan Sistem Prototype

(Sumber : Rosa, 2015**)**

Pengembangan dari perancangan sistem ini dalam pelaksanaannya menggunakan tiga tahap siklus pengembangan model *prototype* yaitu:

1. Mendengarkan Pelanggan *(Listen to Customer)* merupakan tahap pertama dalam merancang sebuah sistem. Pada tahap ini akan menentukan informasi-informasi yang dibutuhkan oleh pelanggan agar tercipta sebuah aplikasi sehingga mengarah pada tujuan dibuatnya aplikasi tersebut.
2. Membangun dan Memperbaiki prototipe *(Build/revice mockup)* dalam tahap ini dilakukan perancangan dan pengkodean untuk sistem yang diusulkan yang mana tahapannya meliputi perancangan proses-proses yang akan terjadi dalam sistem, perancangan diagram UML yang akan digunakan, perancangan antarmuka keluaran serta dilakukan tahap pengkodean terhadap rancangan-rancangan yang telah didefinisikan, kelengkapan *software* dan *hardware*.
3. Pengujian prototipe pada tahapan ini dilakukan pengujian terhadap sistem yang telah disusun dan melakukan pengenalan terhadap sistem yang telah diujikan serta evaluasi apakah sistem yang sudah jadi sudah sesuai dengan yang diharapkan.

1.10. Teori *Blackbox Testing*

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

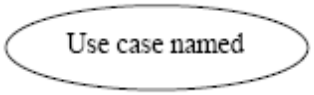
1.11. UML (*Unified Modeling Language*)

Unified Modeling Language (UML) adalah bahasa spesifikasi standar yang dipergunakan untuk mendokumentasikan, menspesifikasikan dan membangun perangkat lunak. UML merupakan metodologi dalam mengembangkan sistem berorientasi objek dan juga merupakan alat untuk mendukung pengembangan sistem. *Unified Modeling Language* (UML) adalah sebuah bahasa yang berdasarkan grafik atau gambar untuk memvisualisasi, menspesifikasikan, membangun, dan pendokumentasian dari sebuah sistem pengembangan software berbasis OO (*Object-Oriented*). UML sendiri juga memberikan standar penulisan sebuah sistem *blue print*, yang meliputi konsep bisnis proses, penulisan kelas-kelas dalam bahasa program yang spesifik, skema database, dan komponen - komponen yang diperlukan dalam sistem *software*. Diagram *Unified Modelling Language* (UML) (Suendri, 2018) antara lain sebagai berikut:

1. *Use Case* Diagram, *Use case* menggambarkan *external view* dari sistem yang akan kita buat modelnya (Rosa, 2015) Model *use case* dapat dijabarkan dalam diagram *use case*, tetapi perlu diingat, diagram tidak indetik dengan model karena model lebih luas dari diagram. *Use case* harus mampu mengambarkan urutan aktor yang menghasilkan nilai terukur (Rosa, 2015).

Tabel 2. 2 Simbol-Simbol UseCase

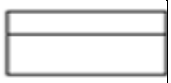
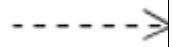
(Sumber : Rosa, 2015**)**

SIMBOL	NAMA	KETERANGAN
	**Actor**	Actor adalah pengguna sistem. Actor tidak terbatas hanya manusia saja, jika sebuah sistem berkomunikasi dengan aplikasi lain dan membutuhkan input atau memberikan *output*, maka aplikasi tersebut juga bisa dianggap sebagai actor.
	***Use Case***	*Use case* digambarkan sebagai lingkaran elips dengan nama use case dituliskan didalam elips tersebut.
	Association	Asosiasi digunakan untuk menghubungkan actor dengan *use case.* Asosiasi digambarkan dengan sebuah garis yang menghubungkan antara Actor dengan *Use Case.*

2. *Class Diagram,* Kelas sebagai suatu set objek yang memiliki atribut dan perilaku yang sama, kelas kadang disebut kelas objek (Rosa, 2015). *Class* memiliki tiga area pokok yaitu :
   1. Nama, kelas harus mempunyai sebuah nama.
   2. Atribut, adalah kelengkapan yang melekat pada kelas. Nilai dari suatu kelas hanya bisa diproses sebatas atribut yang dimiliki.
   3. Operasi, adalah proses yang dapat dilakukan oleh sebuah kelas, baik pada kelas itu sendiri ataupun kepada kelas lainnya.

Tabel 2. 3 Simbol Class Diagram

(Sumber : Rosa, 2015**)**

GAMBAR	NAMA	KETERANGAN
	***Generalization***	Hubungan dimana objek anak (*descendent*) berbagi perilaku dan struktur data dari objek yang ada di atasnya objek induk (*ancestor*).
	***N-Ary Association***	Upaya untuk menghindari asosiasi dengan lebih dari 2 objek.
	***Class***	Himpunan dari objek-objek yang berbagi atribut serta operasi yang sama.
	***Collaboration***	Deskripsi dari urutan aksi-aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang terukur bagi suatu actor.
	***Realization***	Operasi yang benar-benar dilakukan oleh suatu objek.
	***Dependency***	Hubungan dimana perubahan yang terjadi pada suatu elemen mandiri (*independent*) akan mempengaruhi elemen yang bergantung padanya elemen yang tidak mandiri.
	Association	Apa yang menghubungkan antara objek satu dengan objek lainnya.

3. *Activity Diagram*, Diagram actifitas menunjukkan aktifitas sistem dalam bentuk kumpulan aksi-aksi, bagaimana masing-masing aksi tersebut dimulai, keputusan yang mungkin terjadi hingga berakhirnya aksi. *Activity diagram* juga dapat menggambarkan proses lebih dari satu aksi salam waktu bersamaan. "Diagram *activity* adalah aktifitas-aktifitas, objek, *state*, transisi *state* dan *event*. Dengan kata lain kegiatan diagram alur kerja menggambarkan perilaku sistem untuk aktifitas" (Rosa, 2015).

Tabel 2. 4 Simbol Activity Diagram

(Sumber : Rosa, 2015**)**

SIMBOL	KETERANGAN
	Titik Awal
	Titik Akhir
	*Activity*
	Pilihan Untuk mengambil Keputusan
	*Fork*; Digunakan untuk menunjukkan kegiatan yang dilakukan secara parallel atau untuk menggabungkan dua kegiatan peralel menjadi satu.
	*Rake*; Menunjukkan adanya dekomposisi
	Tanda Waktu
	Tanda pengiriman
	Tanda penerimaan
	Aliran akhir (*Flow Final*)

1.12. Penelitian Terkait

Pada (Fadhoelor Rohman, Agung Budi Cahyono 2013) mengembangkan mobile fasilitas untuk pengguna jalan berbasis android haversine formula yang menggunakan aplikasi SIG berbasis mobile phone tentang fasilitas umum untuk pengguna jalan seperti SPBU, Kantor Polisi dan Puskesmas/Rumah Sakit di Pulau Madura dengan menggunakan sistem operasi Android 2.3. Aplikasi SIG fasilitas umum Dapat didesain dan dikembangkan dengan pemograman bahasa Java. Aplikasi mobile phone yang dibuat dapat dapat diakses pada telepon genggam dengan sistem operasi Android minimal 2.3. Aplikasi dapat menampilkan fasilitas umum yaitu SPBU sebanyak 35, Kantor Polisi sebanyak 11, dan Puskesmas/Rumah Sakit sebanyak 9.

Selanjutnya pada (Uswah Hasanah, Novi Safriadi, Tursina 2015) menghasilkan sistem Rancang Bangun Aplikasi *Location Based Service* Lokasi Masjid Pontianak Menggunakan Metode Dijkstra Berbasis Android akan mempermudah pencarian masjid di Pontianak. Kemudahan yang dirasakan pengguna dalam mengakses informasi lokasi masjid dapat digunakan sebagai upaya pengenalan teknologi berbasis sistem informasi geografis kepada masyarakat yang ingin mengetahui masjid terdekat hanya dengan menggunakan handphone android.

Selanjutnya pada (Muh Udka, R. Rizal Isnanto, Rinta Kridalukmana 2015) menghasilkan *Location Based Serv*ice Panduan Pencarian Rumah Sakit Dengan Platforn Android Di Kota Semarang. Pemanfaatan teknologi Location Based Service dalam pengembangan aplikasi profil kampus universitas mulawarman memadukan teknologi *Geographic Information System, Internet Service, dan Mobile Devices* ini membantu memberikan informasi profil dan lokasi fakultas serta fasilitas-fasilitas di Universitas Mulawarman secara mudah, cepat dan akurat.