BAB II

LANDASAN TEORI

2.1 Geotagging

Geotagging adalah sebuah proses untuk menambahkan konten identifikasi geografis. *Geotagging* juga bisa disebut sebagai sebuah bentuk dari *geospatial metadata* (Harvey, 2014). Proses *geotagging* berasal dari GPS yaitu berdasarkan dari garis lintang dan garis bujur atau *latitude* dan *longitude*. *Geotagging* mengijinkan proses pengindeksan pada suatu lokasi. Sebuah aplikasi yang memanfaatkan fitur dari *geotagging* biasanya menambahkan konten-konten media berupa koordinat *latitude*, *longitude*, jarak dan nama lokasi. Data yang ditambahkan terdiri dari fitur berupa teks dan visual. Berikut adalah beberapa teknik implementasi antara lain sebagai berikut:

1. Berbagi dengan teman tentang informasi lokasi tertentu.
2. Merekomendasikan tempat perdagangan atau informasi yang bersifat kebudayaan berdasarkan lokasi lokal.
3. Melacak toko-toko yang memiliki kupon diskon pada daerah lokasi sektiar pengguna.
4. Pengumpulan data untuk lokasi tertentu.
5. Menyediakan informasi untuk lokasi tertentu berdasarkan waktu yang sebenarnya.

2.2 *Agile Methods*

Metode pengembangan Agile (Agile Methods) merupakan salah satu metode pengembangan perangkat lunak yang terbaru dan memiliki langkah yang berbeda dengan metode pengembangan perangkat lunak yang lainnya. Perbedaan tersebut meliputi cara kerja dan langkah-langkah yang ada pada Agile Methods (Ependi, 2012:A-1). Agile Methods mulai berkembang pada era tahun 2000-an dan merupakan metodologi baru yang sangat fleksibel. Agile Methods dikembangkan karena pada metodologi pengembangan perangkat lunak tradisional memiliki

banyak hal yang membuat proses pengembangan tidak dapat berjalan dengan baik sesuai yang diharapkan user (Widodo, 2008:1).

Widodo (2006:E-95) menyebutkan bahwa agile bisa berarti tangkas, cepat, atau ringan. Agility merupakan metode yang ringan dan cepat dalam pengembangan perangkat lunak. Cara berpikir agile tentang pengembangan perangkat lunak secara resmi dideskripsikan pada Agile Manifesto (Shore, 2008:9). Manifesto Pengembangan Perangkat Lunak Agile adalah sebagai berikut (http://agilemanifesto.org/iso/id/) :

"Manifesto Pengembangan Perangkat Lunak Agile. Kami menemukan cara yang lebih baik untuk mengembangkan perangkat lunak dengan melakukan dan membantu sesama untuk menggunakannya. Melalui usaha ini kami telah dapat menghargai:

Individu dan interaksi lebih dari proses dan sarana perangkat lunak Perangkat lunak yang bekerja lebih dari dokumentasi yang menyeluruh Kolaborasi dengan klien lebih dari negosiasi kontrak. Tanggap terhadap perubahan lebih dari mengikuti rencana demikian, walaupun kami menghargai hal di sisi kanan, kami lebih menghargai hal di sisi kiri."

Agile Alliance (http://agilemanifesto.org/iso/id/principles.html) mendefinisikan 12 prinsip untuk mencapai proses yang termasuk dalam agility, yaitu :

a. Prioritas utama adalah memuaskan klien dengan menghasilkan perangkat lunak yang bernilai secara dini dan rutin.
b. Menyambut perubahan kebutuhan walaupun terlambat dalam pengembangan perangkat lunak. Proses Agile memanfaatkan perubahan untuk keuntungan kompetitif klien.
c. Menghasilkan perangkat lunak yang bekerja secara rutin dari jangka waktu beberapa minggu sampai beberapa bulan dengan preferensi kepada jangka waktu yang lebih pendek.
d. Rekan bisnis dan pengembang perangkat lunak harus bekerja sama setiap hari dan sepanjang proyek.
e. Mengembangkan proyek di sekitar individual yang termotivasi. Memberi lingkungan dan dukungan yang dibutuhkan dan mempercayai untuk menyelesaikan pekerjaan dengan baik.

f. Metode yang paling efisien dan untuk menyampaikan informasi dari dan dalam tim pengembangan perangkat lunak adalah dengan komunikasi secara langsung.
g. Perangkat lunak yang bekerja adalah ukuran utama kemajuan.
h. Proses Agile menggalakkan pengembangan berkelanjutan, sponsor, pengembang, dan pengguna akan dapat mempertahankan kecepatan tetap secara berkelanjutan.
i. Perhatian yang berkesinambungan terhadap keunggulan Teknis dan rancangan yang baik meningkatkan agility.
j. Kesederhanaan (seni memaksimalkan jumlah pekerjaan yang belum dilakukan) adalah hal yang amat penting.
k. Arsitektur, kebutuhan, dan rancangan perangkat lunak terbaik muncul dari tim yang dapat mengorganisir diri sendiri.
l. Secara berkala, tim pengembangan berefleksi tentang bagaimana untuk menjadi lebih efektif, kemudian menyesuaikan dan menyelaraskan kebiasaan bekerjanya.

Agile Methods memiliki kelebihan dibandingkan dengan metode-metode lainnya. Menurut Ependi (2012:A-2), kelebihan Agile Methods antara lain adalah meningkatkan rasio kepuasan klien. Klien dapat melakukan review lebih awal terhadap perangkat lunak yang dikembangkan pada saat proses pengembangan. Agile Methods mengurangi resiko kegagalan implementasi perangkat lunak dari segi nonteknis. Jika terjadi kegagalan dalam proses pengembangan, maka nilai kerugian (material maupun immaterial) tidak terlalu besar. Agile Methods adalah metode pengembangan yang bersifat fleksibel dalam menanggapi perubahan-perubahan kebutuhan dan persyaratan dengan tetap mengedepankan komunikasi, kepuasan klien, perangkat lunak yang bekerja, dan kolaborasi.

Saat ini Agile Methods sudah cukup banyak berkembang, di antaranya adalah (Ependi, 2012:A-2) Extreme Programming (XP), Scrum Methodology, Crystal Family, Dynamic System Development Method (DSDM), Adaptive Software Development (ASD), Feature Driven Development (FDD). Agile Methods paling

popular yang digunakan untuk pengembangan aplikasi Web adalah Extreme Programming (XP) (Kappel, et al, 2006:211).

2.3 Metode Pengembangan Sistem

Penelitian ini menggunakan pendekatan *extreme programming* (XP) dalam mendesain sistem aplikasi layanan berbasis lokasi yang terdiri dari 4 (empat) tahapan yaitu:

1. Planning

   Pada tahap planning perlu dilakukan kegiatan identifikasi permasalahan terkait, mengetahui dan melakukan analisa alur sistem yang existing dan new system untuk kebutuhan dan penetapan pelaksanaan pembuatan sistem.

2. Design

   Pada tahap design ini perlu dilakukan perancangan sistem untuk aplikasi menggunakan diagram Unified Modelling Language (UML) untuk menampilkan pemodelan sistem, dan pemodelan arsitektur. Sedangkan untuk perancangan database menggunakan Entity Relationship Diagram (ERD).

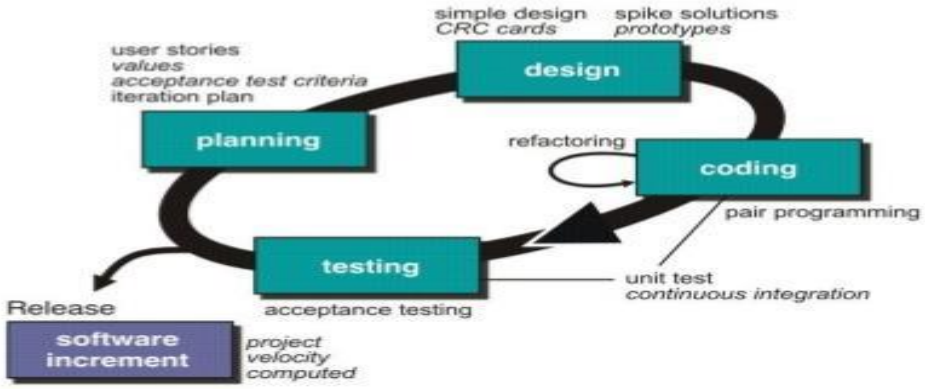
3. Coding

   Pada tahap coding merupakan langkah dalam pembuatan sistem dengan memperhatikan tahapan desain yang sudah dilakukan. Bahasa yang digunakan dalam membuat aplikasi yaitu PHP dengan framework codeigniter, sementara database menggunakan MySQL.

4. Testing

   Tahap ini merupakan langkah terakhir untuk mengetahui apakah desain sistem aplikasi dapat digunakan sesuai dengan kebutuhan pengguna. Pengujian dilakukan dengan menggunakan metode blackbox testing.

Gambar 2.1 menunjukan tahapan yang dilakukan dalam menggunakan pendekatan extreme programming.



Gambar 2.1. Tahapan Proses extreme programming

2.4 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 (Yulianti, 2016).

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* Open GL.
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.

2.5 Basis Data

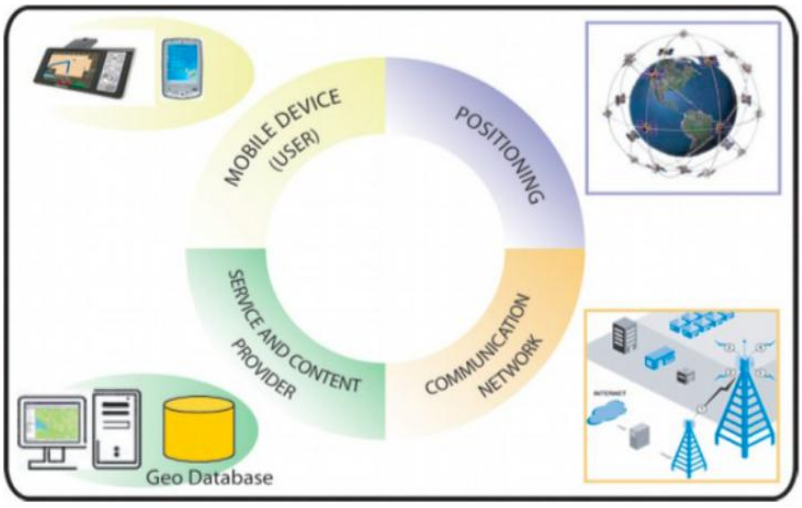
Basis data atau database merupakan kumpulan data satu dengan data lainnya yang tersimpan dalam satu tempat penyimpanan luar dan membutuhkan suatu perangkat lunak untuk menjalankannya (Yulianti, 2016). Elemen-elemen basis data antara lain :

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.

2.6 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. 2 **Komponen *Location Based Service* (LBS)**
(Sumber : Susanty, 2019**)**

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.

2.7 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).

2.8 Perangkat Lunak Pendukung

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

2.8.1 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, 2016).

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

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

2.8.2 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, 2016).

2.8.3 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 (Setianni & Syahputri 2019).

2.9 *Blackbox Testing*

Black Box Testing adalah metode pengujian perangkat lunak yang tes fungsionalitasnya dari aplikasi yang bertentangan dengan struktur internal atau kerja. Pengetahuan khusus dari kode aplikasi / struktur internal dan pengetahuan pemrograman pada umumnya tidak diperlukan. Uji kasus dibangun di sekitar spesifikasi dan persyaratan, yakni, aplikasi apa yang seharusnya dilakukan.Menggunakan deskripsi eksternal perangkat lunak, termasuk spesifikasi, persyaratan, dan desain untuk menurunkan uji kasus. Tes ini dapat menjadi fungsional atau non-fungsional, meskipun biasanya fungsional. Perancang uji memilih input yang valid dan tidak valid dan menentukan output yang benar. Tidak ada pengetahuan tentang struktur internal benda uji itu.

Metode ujicoba blackbox memfokuskan pada keperluan fungsional dari software. Karna itu ujicoba blackbox memungkinkan pengembang software untuk membuat himpunan kondisi input yang akan melatih seluruh syarat-syarat fungsional suatu program. Ujicoba blackbox bukan merupakan alternatif dari ujicoba whitebox, tetapi merupakan pendekatan yang melengkapi untuk menemukan kesalahan lainnya, selain menggunakan metode whitebox. Ujicoba blackbox berusaha untuk menemukan kesalahan dalam beberapa kategori, diantaranya :

1. Fungsi-fungsi yang salah atau hilang
2. Kesalahan interface
3. Kesalahan dalam struktur data atau akses database eksternal
4. Kesalahan performa
5. kesalahan inisialisasi dan terminasi

Dekomposisi kebutuhan untuk dites secara sistematis :

a. Untuk dapat membuat test cases yang efektif, harus dilakukan dekomposisi dari tugas-tugas testing suatu sistem ke aktivitas-aktivitas yang lebih kecil dan dapat dimanajemeni, hingga tercapai test case individual.

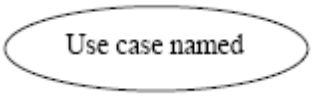
b. Tentunya, dalam disain test case juga digunakan mekanisme untuk memastikan bahwa test case yang ada telah cukup mencakup semua aspek dari sistem.
c. Pendisainan test case dilakukan secara manual, tidak ada alat bantu otomasi guna menentukan test cases yang dibutuhkan oleh sistem, karena tiap sistem berbeda, dan alat bantu tes tak dapat mengetahui aturan benar-salah dari suatu operasi.
d. Desain tes membutuhkan pengalaman, penalaran dan intuisi dari seorang tester.

2.10 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. 1 Simbol-Simbol *UseCase.*
(Sumber : Rosa, 2015**)**

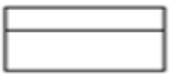
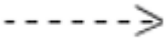
SIMBOL	NAMA	KETERANGAN
	**Actor**	Actor adalah pengguna sistem. Actor tidak terbatas hanya manusia saja, jika sebuah sistem berkomunikasi dengan aplikasi lain dan membutuhkan input atau memberikan *output*, maka aplikasi tersebut juga bisa dianggap sebagai actor.
	***Use Case***	*Use case* digambarkan 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. 2 Simbol *Class Diagram*
(Sumber : Rosa, 2015**)**

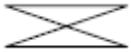
GAMBAR	NAMA	KETERANGAN

	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. 3 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*)

2.11 Penelitian Terdahulu

Berikut merupakan jurnal terkait dengan penelitian terdahulu:

No	Nama	Judul	Terbit/Tahun	Keterangan
1.	Al-Ikhsan, Safaruddin Hidayat; Eosina, Puspa ;Pratama, Ilham Surya	APLIKASI MONITORING JALAN RUSAK KOTA BOGOR BERBASIS ANDROID MENGUNAKAN GEOTAGGING (STUDI KASUS: DINAS BINAMARGA KOTA BOGOR)	Universitas IBN Khaldun Bogor/2015	Penelitian ini membahas mengenai perancangan aplikasi monitoring jalan rusak untuk mempermudah pihak pelanan binamarga untuk mengetahui lokasi kerusakan jalan di kota Bogor.
2.	Purnomo, Anggi Mahadika; Priyambadha, Bayu; Kharisma, Adi Putra	Pengembangan Aplikasi Mobile Pelaporan Keluhan Pelanggan PDAM Menggunakan Fitur Geotagging Berbasis Android (Studi Kasus: PDAM Tirta Tuah Benua Kutai Timur)	JPTIIK/2019	Penelitian ini membahas mengenai pengembangan aplikasi mobile pelaporan keluhan pelanggan PDAM untuk mempermudah masyarakat dalam pelaporan kerusakan dan mempermudah pihak PDAM dalam mencari alamat apabila lokasi berada di pelosok.