# BAB II
# LANDASAN TEORI

## 2.1 Geofencing

Geofencing merupakan teknologi yang digunakan untuk memantau objek bergerak seperti *smartphone,*kendaraan dan lain-lain dengan menggunakan jaringan satelit *Global Positioning Sistem (GPS)* (Beny, Budiman, & Nugroho, 2017). Geofencing menggambarkan sebuah area (*geofence)* yang memiliki batas-batas geografis dari suatu peta. Geofencing pada umumnya dapat dimanfaatkan untuk membantu melacak pengiriman barang yang dibawa oleh suatu kendaraan, memantau posisi seseorang, menjalankan bisnis komersial tertentu, dan absensi otomatis di suatu perusahaan atau universitas. Ukuran wilayah dari geofencing yaitu berkisar dari beberapa meter sampai beberapa kilometer. Bentuk area sebuah *geofence* yaitu berbentuk sebuah lingkaran (*circle)* sedangkan mekanisme menentukan area ditentukan oleh *latitude, longitude*, dan radius dari titik yang ditentukan.

Di dalam sistem operasi Android *geofencing* dibungkus dalam sebuah *library*. Fitur utama *geofencing* yang ada di sistem operasi Android adalah sistem dapat memberikan peringatan berupa notifikasi saat target masuk, tinggal, dan keluar dari area yang sudah ditentukan sebelumnya. Oleh karena itu, *geofencing* juga bisa digunakan untuk meningkatkan kesiapsiagaan terhadap suatu bencana.

## 2.2 *Location Based Service (LBS)*

Location Based Service (LBS) adalah layanan yang dapat diakses menggunakan perangkat *mobile* untuk mengetahui keberadaan lokasi dari pengguna perangkat dan memberikan informasi layanan yang tersedia berdasarkan lokasi tersebut (Fauzi, 2015). Teknologi yang digunakan dalam *Location Based Service* adalah *Global Positioning Sistem (GPS)* dan *cell-bassed location dari* Google. Menurut (Steiniger & Neun, 2008) terdapat lima komponen dalam dalam *Location Based Service* yaitu:

1. *Mobile Device (User)*

   *Mobile Device* adalah perangkat keras seperti *smartphone* yang digunakan untuk meminta layanan dan menerima suatu informasi.

2. *Positioning*

   *Positioning* adalah penentuan posisi *Mobile Device* yang dapat memberikan informasi berupa latitude dan longitude berdasarkan *Global Positioning Service (GPS).*

3. *Communication Network*

   *Communication Network* berfungsi menghubungkan dari penyedia layanan kepada pengguna layanan atau sebaliknya melalui sebuah *gateway*.

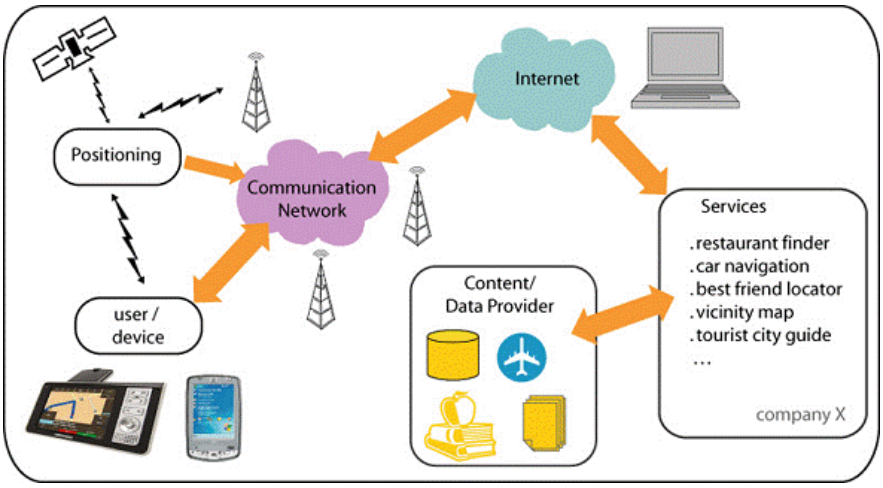
4. *Service and Application Provider*

   *Service and Application Provider* yaitu penyedia layanan menyediakan informasi yang dibutuhkan pengguna seperti lokasi pengguna saat ini, rute ke lokasi tertentu dan jarak antara lokasi satu dengan lokasi lainya.

5. *Data and Content Provider*

   *Data and Content Provider* adalah penyedia data dari suatu layanan tertentu yang disajikan kepada pengguna.

Komponen-komponen tersebut saling berhubungan satu sama lain. Hubungan tersebut dapat dilihat pada Gambar 2.1.




Gambar 2.1 Hubungan Komponen *Location Based Services*

## 2.3 REST (*Represetational State Transfer*)

*Representational State* Transfer (REST) adalah standar arsitektur komunikasi pada *web service*. Arsitektur REST terdapat sebuah REST server yang menyediakan sumber daya dan REST *client* untuk menggunakan sumber daya tersebut. Setiap sumber daya diindentifikasi oleh URIs (*Universal Resources Identifiers).* Sumber daya tersebut dapat berupa teks, JSON (*Javascript Object Notation*) atau XML (*Extensible Markup Language)* (Feridi, 2019). REST yang digunakan untuk sebuah *web service* dikenal sebagai RESTful. RESTful menggunakan

metode HTTP (*Hypertext Transfer Protocol*) yang digunakan untuk menerapkan konsep arsitektur REST, metode HTTP yang disebut *verb*. Metode HTTP tersebut antara lain:

1. GET digunakan untuk membaca sumber daya yang sepesifik yang ada di dalam *database* server.
2. PUT digunakan untuk mengirim data ke *database* server .
3. POST digunakan untuk mengirim data ke *database* server atau memperbarui data dengan suatu pengenal tertentu.
4. DELETE berguna untuk menghapus data dari *database* server.

## 2.4 *Waterfall*

Metode Waterfall adalah pengembangan perangkat lunak yang termasuk dalam model SDLC (*Software Development Life Cycle*) (Bassil, 2012). Terdapat lima tahapan dalam mengembangkan perangkat lunak menggunakan metode *waterfall* yaitu:

a. Analisis

   Tahapan analisis yaitu tahapan untuk mencari informasi yang diperlukan dalam pengembangan perangkat lunak. Tahapan analisis bertujuan agar pengguna dapat memahami perangkat lunak tersebut yang terdiri dari fungsionalitas perangkat lunak dan batasan perangkat lunak. Informasi dapat diperoleh dengan studi literatur dan observasi.

b. Desain

   Tahapan desain adalah tahapan untuk melakukan perancangan perangkat lunak. Metode perancangan memanfaatkan informasi yang didapatkan dari metode analisis. Dalam parancangan menggunakan UML (*Unified Modelling Language).* UML merupakan teknik untuk menggambarkan fungsionalitas sistem, jadi sistem dapat dipahami tidak hanya oleh *developer* saja.

c. Implementasi

   Tahapan implementasi merupakan realisasi dari proses bisnis dan desain menjadi sebuah kode program yang akan dikompilasi menjadi aplikasi operasional.

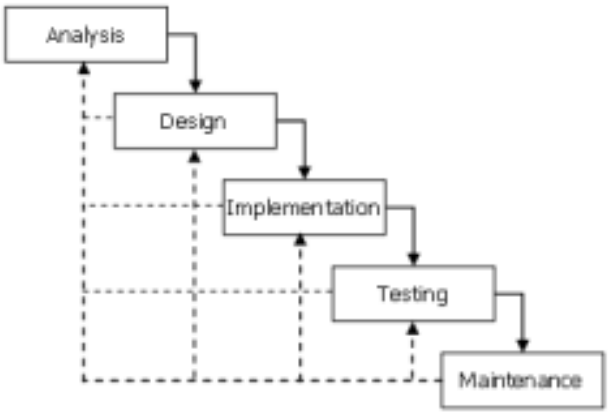
d. Pengujian

   Tahapan pengujian merupakan proses untuk melakukan pemeriksaan terhadap suatu perangkat lunak agar perangkat lunak tersebut memenuhi persyaratan dan spesifikasi yang diharapkan.

e. Pemeliharaan

Tahapan pemeliharaan merupakan proses untuk memodifikasi kode program apabila terjadi kesalahan atau munculnya suatu *bug* yang terdapat pada tahapan pengujian.

Tahapan dari metodologi *waterfall* dapat dilihat pada Gambar 2.2 Tahapan Metodologi Waterfall.




Gambar 2.2 Tahapan Metodologi Waterfall

## 2.5 Google Maps

Google Maps adalah layanan peta *online*, yang meliputi *landmarks*, peta topografi, peta vektor, peta satelit dan lain-lain (Yang & Hsu, 2016). Google maps menawarkan layanan seperti pencarian lokasi, wilayah atau jalan, penentuan arah atau navigasi, pengukuran jarak tempuh, dan pemantuan kondisi lalu lintas (Utari & Wibowo, 2013). Google mengembangkan Google Maps secara *open source* melalui teknologi *Application Programming Interface* (API). Oleh karena itu Google menawarkan kepada para pengembang untuk mengembangkan suatu aplikasi yang membutuhkan tampilan peta dengan fitur-fitur tertentu dengan memanfaatkan API tersebut. Dibutuhkan sebuah kunci berupa kode unik yang didapatkan setelah mendaftar di halaman resmi Google untuk bisa menggunakan fitur dari Google API. Terdapat dua lisensi dari Google Maps API yaitu standar dan bisnis. Google Maps API dibuat menggunakan bahasa pemrograman Javascript, maka pengembang diharapkan sudah mengetahui dasar-dasar pemrograman tersebut dan mengetahui Pemrograman Berorientasi *Object*. Adapun gambar perbedaan fitur Google Maps API dapat ditunjukan pada Gambar 2.3.

| Features | Maps API | Maps API for Business |
|---|---|---|
| Street View | ✓ | ✓ |
| Geocoding Web Service | 2500 requests per 24 hour period | 100 000 requests per 24 hour period |
| Directions Web Service | 2500 requests per 24 hour period with 10 waypoints per request | 100 000 requests per 24 hour period with 23 waypoints per request |
| Distance Matrix Web Service | 100 elements per query<br>100 elements per 10 seconds<br>2500 elements per 24 hour period | 625 elements per query<br>1000 elements per 10 seconds<br>100 000 elements per 24 hour period |
| Elevation Web Service | 2500 requests per 24 hour period with 25 000 samples per 24 hour period | 100 000 requests per 24 hour period with 1 000 000 samples per 24 hour period |
| Static Maps API maximum resolution | 640 x 640 | 2048 x 2048 |
| Static Maps API maximum scale | 2X | 4X |
| Street View Image API maximum resolution | 640 x 640 | 2048 x 2048 |
| Analytics | | ✓ |

Gambar 2.3 Perbedaan fitur lisensi standar dan bisnis

## 2.6 SQLite

SQLite merupakan suatu *library* penyimpanan data yang bekerja secara tunggal yaitu tidak banyak membutuhkan *library* eksternal, mengakses basis data secara *Serverless* yaitu dapat melakukan *read* dan *write* secara langsung dari file basis data tanpa membutuhkan server yang terpusat (Setiyadi & Harihayati, n.d.). Beberapa karakteristik dari SQLite adalah:

1. SQLite menyimpan basis data secara *single cross-platform* artinya SQLite dapat digunakan untuk menyimpan suatu data yang membutuhkan penyimpanan secara lokal.
2. SQLlite berukuran kecil dan ringan, membutuhkan kurang lebih antara 200 *kilobyte* sampai dengan 400 *kilobyte.*
3. SQLite mendukung sebagian besar bahasa pemrograman yang terdapat di standar SQL92.
4. SQLite dibangun menggunakan bahasa ANSI-C dan tersedia di berbagai sistem operasi seperti UNIX dan Windows.

SQLite mendukung perintah standar basis data *relational* yang sama seperti basis data MySQL, perintah-perintah tersebut terdiri dari *Data Definition Language* (DDL), *Data Manipulation Language* (DML), dan *Data Query Language* (DQL) yang memiliki perbedaan sebagai berikut :

a. *Data Definition Language* (DDL)

- *Create* berfungsi untuk membuat tabel baru

- *Drop* berfungsi menghapus semua tabel
- *Alter* berfungsi memodifikasi tabel

b. *Data Manipulation Language* (DML)

- *Insert* berfungsi menambahkan data baru ke dalam basis data
- *Update* berfungsi mengubah data di dalam basis data
- *Delete* berfungsi menghapus data dari basis data

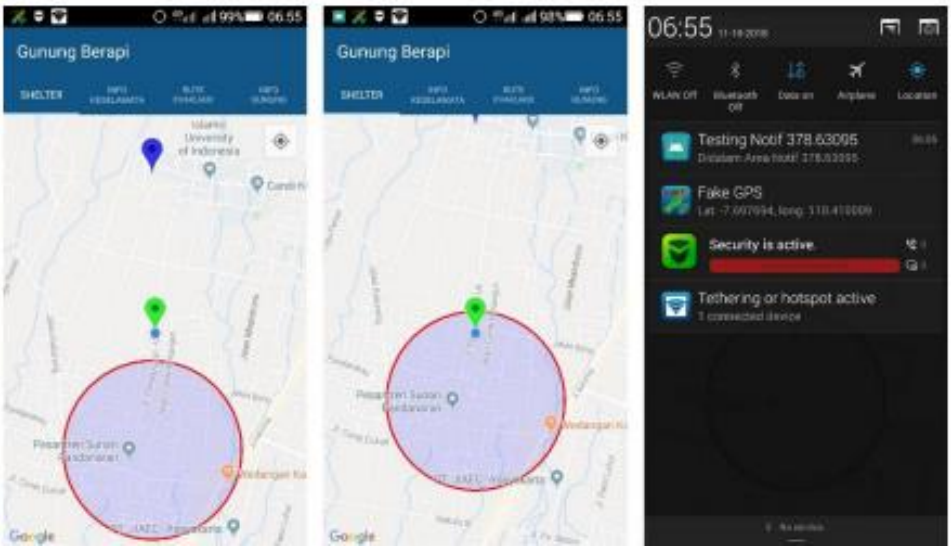
c. *Data Query Language* (DQL)

- *Select* berfungsi membuat permintaan memperoleh data data basis data

## 2.7 **Penelitian Terdahulu**

Pada penelitian ini, dilakukan proses analisis terhadap penelitian terdahulu sebagai sumber referensi penelitian sistem peringatan kawasan rawan bencana memanfaatkan teknik *geofencing* pada platform android: studi kasus Gunung Merapi. Penelitian tersebut adalah :

a. Penelitian dengan judul Aplikasi Peringatan Kesiapsiagaan Terhadap Bencana Alam Erupsi Gunung Berapi Berbasis Android oleh Tedy Suwandi. Dalam penelitian ini dijelaskan implementasi sistem peringatan bencana Gunung Berapi berbasis Android. Daerah implementasi sistem ini ditujukan untuk sekitar daerah wisata Gunung Merapi Daerah Istimewa Yogyakarta. Aplikasi ini memberikan informasi seputar erupsi gunung Berapi dan memberikan informasi jika pengguna memasuki area rawan bencana. Teknologi yang digunakan yaitu Google Maps API yang bisa memberikan tampilan peta *online* yang di dalamnya terdapat lokasi pengguna yang memanfaatkan GPS *(Global Positioning Sistem)*, area rawan bencana, zona evakuasi, dan rute menuju zona evakuasi. Setiap informasi yang terdapat pada peta *online* digambarkan dengan sebuah *marker*. Dalam sistem tersebut hanya terdapat satu pengguna, area rawan bencana, zona evakuasi, dan rute menuju zona evakuasi. Rute menuju zona evakuasi dibuat secara statis dan tidak ada manajemen pengolahan data yang dilakukan oleh pengelola sistem. Adapun informasi penelitian tersebut dapat dilihat pada Gambar 2.4 Visualisasi area dan notifikasi.




Gambar 2.4 Visualisasi area dan notifikasi

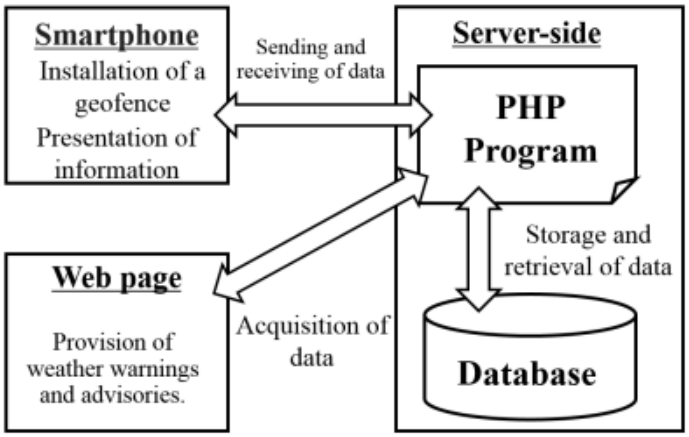
pada penelitian tersebut juga terdapat tampilan berupa detail zona evakuasi berserta rute menuju zona evakuasi tersebut. Adapun tampilan zona evakuasi dapat dilihat pada Gambar 2.5.


Gambar 2.5 Zona Evakuasi

*b.* *Using* Geofencing *for a Disaster Information Sistem* dilakukan oleh Akira Suyama. Dalam penelitian tersebut dijelaskan implementasi *geofencing* sebagai bentuk kewaspadaan terhadap bencana dengan cara mendeteksi pergerakan pengguna dan memberikan

informasi risiko kepada pengguna. Arsitektur dari sistem tersebut adalah *client*-server. Kerja server yaitu untuk menampilkan informasi berupa area *geofence* dan memonitor pergerakan *client*. Dengan demikian sistem dapat mengirimkan peringatan dengan tepat waktu untuk pengguna jika terjadi bahaya. Adapun informasi area dapat dilihat pada Gambar 2.6 dan struktur sistem dapat dilihat pada Gambar 2.7.


Gambar 2.6 Area Geofencing




Gambar 2.7 Arsitektur Sistem

Perbandingan penelitian sistem peringatan kawasan rawan bencana memanfaatkan teknik *geofencing* pada platform android : studi kasus Gunung Merapi dengan kedua penelitian sejenis yang telah dijelaskan diatas dapat dilihat pada Tabel 2.1.

Tabel 2.1 Perbandingan Penelitian Terdahulu

| | Aplikasi Peningkatan Kesiapsiagaan Terhadap Bencana Alam Erupsi Gunung Berapi Berbasis Android | *Using* Geofencing *for a Disaster Information Sistem* | Sistem Peringatan Kawasan Rawan Bencana Memanfaatkan Teknik Geofencing Pada Platform Android: Studi Kasus Gunung Merapi |
|---|---|---|---|
| Platform | Android | iOS | Android |
| Tujuan | Memberikan informasi peringatan bencana alam erupsi gunung Berapi berbasis Android | Menggunakan teknik *geofencing* untuk memberikan informasi becana berbasis iOS | Menggunakan teknik *geofencing* untuk menginformasikan bencana Gunung Merapi berbasis Android |
| Teknologi | - Google Maps API | - Arsitektur *client*-server untuk mendeteksi keberadaan pengguna<br>- bahasa Pemrograman Swift | - RESTful API sebagai data area rawan bencana dan zona evakuasi<br>- Bahasa pemrograman Kotlin<br>- Google Maps API |
| Mekanisme | Terdapat sebuah area rawan bencana berbentuk lingkaran dan *shelter* atau zona evakuasi yang dibuat secara statis di dalam sistem yang dapat memicu notifikasi jika lokasi | Terdapat area rawan bencana berbentuk lingkaran dan mendapatkan notifikasi jika pengguna berada di area tersebut | Terdapat area rawan bencana berbentuk lingkaran dan mendapatkan notifikasi jika pengguna berada di area tersebut dengan dilengkapi rute |

| | | | |
|---|---|---|---|
| | pengguna berada di dalam area rawan bencana | | menuju zona evakuasi terdekat |
| Pengolahan Data | - Tidak ada | - MariaDB | - MySql<br>- SqlLite<br>- Firebase |