TUGAS AKHIR - KI141502

RANCANG BANGUN SISTEM *REAL TIME TRACKING INDOOR POSITION* UNTUK STUDI KASUS PADA GEDUNG TEKNIK INFORMATIKA ITS

DINAR WINIA MAHANDHIRA
NRP 5112100002

Dosen Pembimbing I
Dr. Tech. Ir. R. V. Hari Ginardi, M.Sc.

Dosen Pembimbing II
Dini Adni Navastara, S.Kom., M.Sc.

JURUSAN TEKNIK INFORMATIKA
Fakultas Teknologi Informasi
Institut Teknologi Sepuluh Nopember
Surabaya 2016

[Halaman ini sengaja dikosongkan]

TUGAS AKHIR - KI141502

RANCANG BANGUN SISTEM *REAL TIME TRACKING INDOOR POSITION* UNTUK STUDI KASUS PADA GEDUNG TEKNIK INFORMATIKA ITS

DINAR WINIA MAHANDHIRA
NRP 5112100002

Dosen Pembimbing I
Dr. Tech. Ir. R. V. Hari Ginardi, M.Sc.

Dosen Pembimbing II
Dini Adni Navastara, S.Kom., M.Sc.

JURUSAN TEKNIK INFORMATIKA
Fakultas Teknologi Informasi
Institut Teknologi Sepuluh Nopember
Surabaya 2016

[Halaman ini sengaja dikosongkan]

FINAL PROJECT - KI141502

DESIGN SYSTEM FOR REAL TIME TRACKING INDOOR POSITION FOR CASE STUDY ON DEPARTMENT OF INFORMATICS ITS

DINAR WINIA MAHANDHIRA
NRP 5112100002

Supervisor I
Dr. Tech. Ir. R. V. Hari Ginardi, M.Sc.

Supervisor II
Dini Adni Navastara, S.Kom., M.Sc.

DEPARTMENT OF INFORMATICS
FACULTY OF INFORMATION TECHNOLOGY
INSTITUT TEKNOLOGI SEPULUH NOPEMBER
SURABAYA 2016

[Halaman ini sengaja dikosongkan]

LEMBAR PENGESAHAN

RANCANG BANGUN SISTEM *REAL TIME TRACKING INDOOR POSITION* UNTUK STUDI KASUS PADA GEDUNG TEKNIK INFORMATIKA ITS

TUGAS AKHIR

Diajukan Untuk Memenuhi Salah Satu Syarat
Memperoleh Gelar Sarjana Komputer
pada
Bidang Studi Algoritma dan Pemrograman
Program Studi S-1 Jurusan Teknik Informatika
Fakultas Teknologi Informasi
Institut Teknologi Sepuluh Nopember

Oleh
DINAR WINIA MAHANDHIRA
NRP : 5112100002

Disetujui oleh Pembimbing Tugas Akhir:

1. Dr. Tech. Ir. R. V. Hari Ginardi, M.Sc. .....
   NIP: 19650518 199203 1 003 (Pembimbing 1)

2. Dini Adni Navastara, S.Kom., M.Sc. .....
   NIP: 19851017 201504 2 001 (Pembimbing 2)

SURABAYA
JUNI, 2016

[Halaman ini sengaja dikosongkan]

RANCANG BANGUN SISTEM REAL TIME TRACKING INDOOR POSITION UNTUK STUDI KASUS PADA GEDUNG TEKNIK INFORMATIKA ITS

Nama Mahasiswa	**: DINAR WINIA MAHANDHIRA**
NRP	**: 5112100002**
Jurusan	**: Teknik Informatika FTIF-ITS**
Dosen Pembimbing 1	**: Dr. Tech. Ir. R. V. Hari Ginardi, M.Sc.**
Dosen Pembimbing 2	**: Dini Adni Navastara, S.Kom, M.Sc.**

Abstrak

Teknologi pelacakan keberadaan sudah banyak digunakan akhir-akhir ini, seperti navigasi menuju suatu tempat, pengambilan gambar menggunakan geotagging, pelacakan keberadaan seseorang, pemetaan berbagai gedung, dan lain sebagainya. Semua aktivitas tersebut menggunakan teknologi Global Positioning System (GPS). Namun, disamping pemanfaatannya yang cukup besar, GPS masih memiliki kelemahan yaitu rendahnya tingkat akurasi saat digunakan di dalam gedung. Padahal, hampir kebanyakan orang menghabiskan waktunya untuk beraktivitas di dalam gedung.

Teknologi yang ada saat ini untuk mengatasi hal tersebut adalah memanfaatkan sinyal WiFi yang ada di dalam gedung atau disebut WiFi Positioning System (WPS) untuk melacak keberadaan seseorang atau suatu barang di dalam gedung. Sistem ini memanfaatkan sinyal WiFi yang telah melalui proses klasifikasi sehingga didapatkan suatu koordinat tertentu. Koordinat tersebut juga dapat digunakan untuk menghasilkan informasi lain, seperti mengetahui nama ruangan atau daerah. Namun, pelacakan keberadaan yang dilakukan secara real time tidak bisa hanya mengandalkan

sinyal WiFi. Selain karena proses klasifikasi yang dilakukan cukup membutuhkan waktu, klasifikasi WiFi tidak bisa selalu dilakukan terutama saat suatu daerah memiliki WiFi atau Access Point yang sedikit. Oleh karena itu, sistem ini perlu dikembangkan dengan memanfaatkan sensor yang ada di smartphone untuk mendeteksi pergerakan seseorang secara real-time, seperti accelerometer, magnetometer, dan gyroscope.

Uji coba dilakukan di lantai tiga gedung Teknik Informatika ITS dengan tiga macam pengujian, yaitu pengujian akurasi penentuan koordinat keberadaan seseorang, pendeteksi langkah seseorang, dan penentuan arah hadap seseorang. Pada pengujian yang dilakukan, sistem memberikan performa cukup baik dengan rata-rata akurasi sebesar 2,5 meter untuk penentuan posisi awal pengguna. Sedangkan rata-rata persentase akurasi untuk pendeteksian langkah dan arah hadap pengguna adalah sebesar 94,8% dan 94,48%.

Kata kunci: Indoor Localization, Location Based Service, Sensor Android

DESIGN SYSTEM FOR REAL TIME TRACKING INDOOR POSITION FOR CASE STUDY ON DEPARTMENT OF INFORMATICS ITS

Student's Name	**: DINAR WINIA MAHANDHIRA**
Student's ID	**: 5112100002**
Department	**: Teknik Informatika FTIF-ITS**
First Advisor	**: Dr. Tech. Ir. R. V. Hari Ginardi, M.Sc.**
Second Advisor	**: Dini Adni Navastara, S.Kom, M.Sc.**

Abstract

Tracking system has been widely used nowadays, such as navigating to a place, taking pictures using geotagging, tracking position, mapping various buildings, etc. All these activities using Global Positioning System (GPS). However, Namun, disamping pemanfaatannya yang cukup besar, GPS still has a weakness such as low level accuracy when it used in the building. In fact, most of people doing their activities in the building.

Current technology to solve this problem is using radio frequency (RF) such as WiFi signal inside the building instead of GPS. It is called WiFi Positioning System (WPS) and it used for tracking position of person inside the building. This system use WiFi signal that processed by classification so it gets certain coordinates or position. This coordinate can be used to get other information such as room information. But real time tracking position can not just rely on WiFi signal. Besides it takes time for classification process, this classification can not be done continuously especially in area that has bit access point. Therefore, these systems need to be developed by utilizing existing motion sensors in smartphone to predict user movement in real time such as accelerometer, magnetometer, dan gyroscope.

This system is tested on third floor of Department of Informatics ITS by three kinds of testing that is determining the coordinates of starting point, step detection, and heading estimation. On the tests performed, this system gives a pretty good performance with an average accuracy of 2.5 meter for determining coordinates of starting point. While the average ppercentage of accuracy for step detection and heading estimatioin is 94.48% and 94.48%.

Keywords: Android Sensor, Location Based Service, Indoor Localization.

KATA PENGANTAR

Puji syukur penulis panjatkan kepada Tuhan Yang Maha Esa karena atas segala karunia dan rahmat-Nya penulis dapat menyelesaikan Tugas Akhir yang berjudul "Rancang Bangun Sistem Real Time Tracking Indoor Position untuk Studi Kasus Pada Gedung Teknik Informatika ITS".

Pengerjaan Tugas Akhir ini merupakan suatu kesempatan yang sangat baik bagi penulis agar dapat belajar lebih banyak untuk memperdalam dan meningkatkan apa yang telah didapatkan selama menempuh perkuliahan di Teknik Informatika ITS. Dengan Tugas Akhir ini penulis juga dapat menghasilkan suatu implementasi dari apa yang telah penulis pelajari.

Selesainya Tugas Akhir ini tidak lepas dari bantuan dan dukungan beberapa pihak. Sehingga pada kesempatan ini penulis mengucapkan terima kasih kepada:

1. Tuhan Yang Maha Esa atas segala karunia dan rahmat-Nya yang telah diberikan selama ini.
2. Orang tua, saudara serta keluarga penulis yang telah memberikan dukungan moral dan material serta do'a yang tak terhingga untuk penulis. Serta selalu memberikan semangat dan motivasi pada penulis dalam mengerjakan Tugas Akhir ini.
3. Bapak Dr. Tech. Ir. R. V.Hari Ginardi, M.Sc. selaku pembimbing I dan Ibu Dini Adni Navastara, S.Kom, M.Sc. yang telah membantu dan membimbing penulis dalam menyelesaikan Tugas Akhir ini.
4. Bapak Dr. Agus Zainal Arifin, S.Kom., M.Kom. selaku dosen wali penulis yang telah memberikan motivasi, nasehat, bimbingan, dan bantuan yang banyak kepada penulis selama berkuliah di kampus.
5. Bapak dan Ibu dosen Teknik Informatika yang banyak memberikan motivasi dan ilmu kepada penulis selama berkuliah di kampus.

6. Angkatan 2012, yang telah membantu penulis memberikan semangat untuk berjuang di dunia perkuliahan Teknik Informatika ITS.
7. Sahabat dekat yang selalu memberikan dukungan dan menemani hari-hari penulis: Eric, Putri dan Ari.
8. Teman seperjuangan penulis di AP Alifa, Riko, Ratih yang terus memotivasi penulis dalam pengerjaan Tugas Akhir ini.
9. Fauzan, Fadrian, Anggara, Rona, dan Agha yang telah bersedia meluangkan waktunya di saat penulis meminta bantuan.
10. Administrator Lab. Pemrograman, Karsten, Christo, Otniel, Adhe, Anggara, Wawan, Udin, Sani, Ine yang selalu menghibur penulis saat bosan.
11. Keluarga besar Kesma BEM ITS Muda Bersahabat dan BEM ITS Kolaborasi, Mas Zaid, Mbak Fahmy, Mas Vilat, Mbak Fitri, Linda, Marta, Yenny, Hafizh, Indra, Umam, dan Ariska yang senantiasa menemani hari-hari penulis di tahun ketiga dan keempat.
12. Serta semua pihak yang tidak dapat disebutkan satu persatu dan telah turut membantu penulis dalam pengerjaan Tugas Akhir ini.

Penulis menyadari bahwa Tugas Akhir ini masih memiliki banyak kekurangan. Sehingga dengan kerendahan hati, penulis mengharapkan kritik dan saran dari pembaca untuk perbaikan ke depan.

Surabaya, Juni 2016

DAFTAR ISI

[Halaman ini sengaja dikosongkan]

DAFTAR GAMBAR

DAFTAR TABEL

BAB I
PENDAHULUAN

1.1 Latar Belakang

Perkembangan teknologi *Location Based Service* (LBS) semakin lama mulai mengalami perkembangan yang beragam. Pada awalnya, sebagian besar orang banyak mengembangkan LBS di lingkungan *outdoor*, dimana *Global Positioning System* (GPS) memiliki peran yang sangat besar di dalamnya. Namun sistem ini memiliki akurasi yang rendah untuk membedakan posisi di dalam suatu ruangan atau gedung, terlebih lagi GPS hanya memiliki konsep *2D Localization* dimana GPS tidak dapat membedakan lokasi ketinggian tempat atau gedung. Menurut survei yang dilakukan oleh *Strategy Analytic* pada tahun 2010, lebih dari 70% orang menghabiskan waktunya dengan beraktivitas di dalam gedung seperti pusat perbelanjaan, kantor, universitas, bandara, dan lain sebagainya. Tentu saja hal ini menjadi peluang besar terhadap pengembangan *Indoor Positioning System* (IPS), salah satunya adalah *tracking indoor position*.

Banyak manfaat yang bisa didapatkan dalam mengembangkan *tracking indoor position*, seperti menunjukkan lokasi keberadaan pengguna saat itu juga, memberi label dan informasi suatu ruangan yang sedang dilewati, bahkan kedepannya bisa juga digunakan untuk sistem navigasi di dalam gedung. Namun tidak seperti saat diluar gedung, *tracking* di dalam gedung memiliki beberapa keterbatasan. Pertama, dibutuhkan penentu lokasi atau posisi yang dapat memberikan informasi lebih akurat selain GPS, mengingat rendahnya akurasi GPS di dalam gedung. Solusi yang telah ada selama ini adalah menggunakan *Radio Frequency* (RF) seperti *WiFi* sebagai *Access Point* (AP). Namun muncul permasalahan baru untuk kasus gedung yang memiliki sedikit *WiFi*, yaitu kurangnya informasi akurat untuk mengetahui detail posisi serta arah dan perpindahan yang dilakukan pengguna [1]. Untuk mengatasi permasalahan kedua, maka perlu digunakan pendekatan lain, misalnya penggunaan

sensor untuk mengetahui pergerakan pengguna secara spesifik, apakah saat itu pengguna sedang melakukan pergerakan atau diam di tempat.

Dengan adanya permasalahan di atas dan teknologi yang sedang berkembang, usulan Tugas Akhir ini adalah membuat suatu aplikasi *mobile* yang dapat melakukan pelacakan posisi seseorang secara *real time* di dalam suatu gedung. Dengan usulan Tugas Akhir ini, pengguna diharapkan dapat mengetahui informasi lokasi dimana dia berada saat itu juga. Studi kasus yang digunakan dalam Tugas Akhir ini adalah gedung Teknik Informatika ITS.

1.2 Rumusan Masalah

Rumusan masalah yang diangkat dalam Tugas Akhir ini adalah sebagai berikut:

1. Bagaimana cara mengembangkan teknologi IPS untuk *real-time tracking* pada *smartphone*?
2. Bagaimana cara memperoleh dan mengolah data untuk IPS?
3. Bagaimana akurasi *tracking indoor position* menggunakan sistem ini?

1.3 Batasan Masalah

Permasalahan yang dibahas dalam Tugas Akhir ini memiliki beberapa batasan, di antaranya sebagai berikut:

1. Aplikasi dibangun pada sistem operasi Android dengan minimum SDK 5.0 dan memiliki sensor *accelerometer* serta *magnetometer*.
2. Studi kasus aplikasi ini dilakukan pada gedung Teknik Informatika ITS lantai tiga.
3. *Smartphone* harus berada dalam posisi *steady hand* saat digunakan.
4. Aplikasi akan menganggap setiap pergerakan pengguna sebagai pergerakan maju ke depan, sehingga saat pengguna berjalan mundur akan tetap dianggap bergerak maju ke arah depan.

5. Aplikasi hanya dapat dijalankan jika tersambung dengan *WiFi* yang sesuai dengan lingkungan gedung Teknik Informatika atau *data sample* sistem.
6. Aplikasi memerlukan koneksi internet untuk transfer data dengan *server*.
7. Terdapat dua aplikasi yang akan dibuat, yaitu aplikasi Android dan *web service*.

1.4 Tujuan

Tujuan dari pembuatan Tugas Akhir ini adalah sebagai berikut:

1. Membangun sistem *real time tracking indoor position* menggunakan sinyal *WiFi* dan sensor pada *smartphone* untuk mengetahui posisi dan perpindahan seseorang secara *real time* pada gedung Teknik Informatika ITS.
2. Mengukur dan mengevaluasi akurasi dari metode yang dibuat.

1.5 Manfaat

Pengerjaan Tugas Akhir ini diharapkan dapat mempermudah seseorang mengetahui posisinya sendiri dan posisi pengguna lain secara *realtime*, serta dapat mengetahui semua aktivitas yang terjadi pada saat berada di lingkungan gedung Teknik Informatika ITS.

1.6 Metodologi

Ada beberapa tahap dalam proses pengerjaan Tugas Akhir ini, mulai dari pencarian literatur, pengimplementasian, uji coba, hingga pembuatan laporan akhir. Berikut adalah tahap-tahap yang dilakukan dalam pembuatan Tugas Akhir ini.

a. Studi Literatur

Pada bagian studi literatur ini nantinya akan dipelajari yaitu Pemrograman Android, Google Maps API, *web service* PHP, dan penggunaan sensor *accelerometer* serta *magnetometer* pada

sistem operasi Android, klasifikasi *Enhanced Weighted k-Nearest Neighbor* (EWKNN), estimasi arah hadap, dan algoritma *step detection*.

b. Perancangan dan Desain Sistem

Pada tahap ini dilakukan perancangan sistem dengan menggunakan studi literatur dan mempelajari konsep aplikasi yang akan dibuat. Dengan berbekal teori, metode, dan informasi yang sudah terkumpul pada tahap sebelumnya diharapkan dapat membantu dalam proses perancangan sistem. Tahap ini merupakan tahap terpenting pada bentuk awal atau *prototype* yang akan diimplementasikan ke depannya.

c. Implementasi

Pada tahap ini dilakukan implementasi rancangan sistem yang telah dibuat. Tahapan ini merealisasikan apa yang terdapat pada tahapan sebelumnya sehingga menjadi sebuah aplikasi dengan simulasi yang digunakan sesuai dengan apa yang telah direncanakan.

d. Uji Coba dan Evaluasi

Pada tahap ini aplikasi yang telah selesai dibuat akan diuji. Pengujian dan evaluasi akan dilakukan dengan melihat kesesuaian dan ketepatan sistem mendeteksi posisi serta pergerakan pengguna saat berjalan atau sedang berhenti.

e. Penyusunan Laporan Tugas Akhir

Pada tahap ini disusun laporan Tugas Akhir sebagai dokumentasi pelaksanaan Tugas Akhir dari awal sampai akhir yang mencakup seluruh konsep, teori, implementasi, dan hasil yang telah dikerjakan.

1.7 Sistematika Penulisan Laporan Tugas Akhir

Buku Tugas Akhir ini disusun dengan sistematika penulisan yang akan dijelaskan sebagai berikut.

BAB I. PENDAHULUAN

Bab yang berisi mengenai latar belakang, tujuan, dan manfaat dari pembuatan Tugas Akhir. Selain itu permasalahan, batasan masalah, metodologi yang digunakan, dan sistematika penulisan juga merupakan bagian dari bab ini.

BAB II. TINJAUAN PUSTAKA

Bab ini berisi penjelasan secara detail mengenai dasar-dasar penunjang dan teori-teori yang digunakan untuk mendukung pembuatan Tugas Akhir ini.

BAB III. ANALISIS DAN PERANCANGAN

Bab ini berisi tentang desain sistem yang akan dikerjakan. Di dalamnya terdapat rancangan kasus penggunaan dan rancangan tampilan.

BAB IV. IMPLEMENTASI

Bab ini membahas implementasi dari desain yang telah dibuat pada bab sebelumnya. Penjelasan berupa *code* yang digunakan untuk proses implementasi.

BAB V. UJI COBA DAN EVALUASI

Bab ini menjelaskan kemampuan perangkat lunak dengan melakukan pengujian kebenaran dari sistem yang telah dibuat.

BAB VI. KESIMPULAN DAN SARAN

Bab ini merupakan bab terakhir yang menyampaikan kesimpulan dari hasil uji coba yang dilakukan dan saran untuk pengembangan perangkat lunak ke depannya.

[Halaman ini sengaja dikosongkan]

BAB II
TINJAUAN PUSTAKA

Bab ini berisi penjelasan teori-teori yang berkaitan dengan algoritma yang diajukan pada pengimplementasian program. Penjelasan ini bertujuan untuk memberikan gambaran secara umum terhadap program yang dibuat dan berguna sebagai penunjang dalam pengembangan sistem.

2.1 Location Based Service (LBS)

Location Based Service (LBS) adalah suatu layanan yang menyediakan informasi dengan memanfaatkan informasi-informasi geografis yang ada pada suatu perangkat bergerak. Dalam menyediakan informasi, LBS biasanya didukung oleh peta yang ditampilkan dalam perangkat bergerak [2]. Layanan ini memungkinkan untuk melakukan perhitungan yang berhubungan dengan lokasi dan jarak seseorang, misalnya rute terdekat, tempat terdekat, dll. LBS terdiri dari beberapa komponen, yaitu [3]:

a. Perangkat *Mobile*

Perangkat *mobile* digunakan untuk meminta informasi yang dibutuhkan pengguna, seperti teks, gambar, dan suara. Contoh perangkat *mobile* adalah laptop, *handphone*, dll.

b. Komponen Pelacakan

Komponen pelacakan dalam hal ini adalah sistem yang digunakan untuk menentukan posisi suatu objek. Komponen pelacakan yang paling banyak digunakan terutama pada *smartphone* adalah *Global Positioning System* (GPS). GPS bekerja dengan menggunakan satelit GPS yang memancarkan sinyal ke bumi lalu diterima oleh *smartphone*. Selain GPS, komponen pelacakan yang lain adalah *Wireless Positioning System* yang menggunakan sinyal *WiFi* untuk menentukan lokasi suatu objek. Pada pengerjaan Tugas Akhir ini, komponen pelacakan yang digunakan adalah *Wireless Positioning System* (WPS).

c. Jaringan Komunikasi

Jaringan komunikasi digunakan untuk mengirimkan data pengguna dan permintaan layanan dari perangkat *mobile* ke penyedia layanan dan meminta informasi kembali untuk pengguna.

d. Penyedia Aplikasi dan Layanan

Penyedia layanan menawarkan sejumlah layanan yang berbeda kepada pengguna dan bertanggung jawab untuk pengolahan permintaan layanan. Layanan tersebut dapat berupa perhitungan posisi dan menemuan rute yang berhubungan dengan posisinya, atau bahkan mencari informasi yang spesifik tentang posisi objek yang lain.

e. Penyedia Konten dan Data

Penyedia layanan biasanya disimpan dan dikelola oleh pihak yang memberikan layanan pengelolaan data.

2.2 Indoor Positioning System (IPS)

Indoor Positioning System adalah sebuah solusi yang didasari oleh *magnetic*, data sensor atau perangkat jaringan yang digunakan untuk menemukan suatu benda atau lokasi seseorang secara nirkabel di dalam bangunan [4].

Sebenarnya, layanan yang menyediakan informasi posisi, navigasi, dan *timing* sudah banyak dikenal oleh masyarakat umum dengan sebutan *Global Positioning System* (GPS). Namun karena GPS memiliki kelemahan pada tingkat akurasi pelacakan di dalam bangunan, maka pemanfaatan IPS sebagai teknologi nirkabel di sebuah bangunan dapat meningkatkan akurasi pelacakan tersebut saat berada di dalam gedung.

2.3 Indoor Localization

Indoor Localization merupakan teknologi yang digunakan untuk menentukan lokasi sebuah objek atau seseorang yang berada di dalam gedung. Teknologi ini hampir sama dengan IPS, namun letak perbedaannya *Indoor Positioning System* menentukan koordinat global dari sebuah lokasi (contoh: garis bujur dan garis

lintang), sedangkan *indoor localization* menentukan koordinat relatif (contoh: Laboratorium AP, Ruang IF-106) [3].

2.4 WiFi

WiFi adalah teknologi jaringan nirkabel yang memungkinkan komputer dan perangkat digital lain untuk berkomunikasi melalui sinyal nirkabel. Ini menggambarkan komponen jaringan yang berbasis pada salah satu standar 802.11 yang dikembangkan oleh IEEE dan diadopsi oleh *WiFi Alliance.* Contoh standar *WiFi* antara lain 802.11a, 802.11b, 802.11g, 802.11n, dan 802.11ac [3].

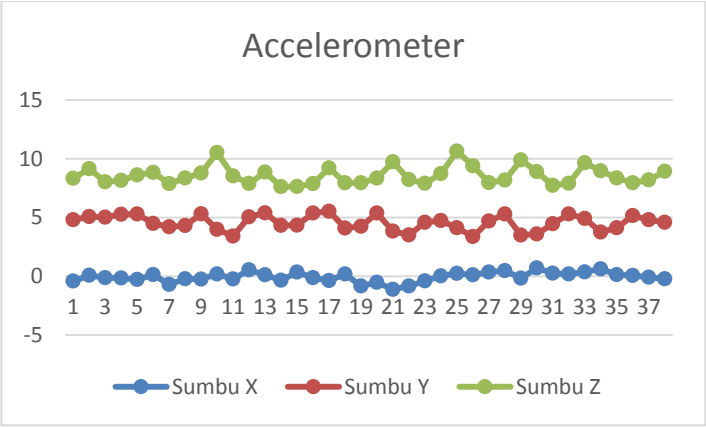
WiFi merupakan cara standar menghubung ke jaringan nirkabel. Hampir semua komputer modern memiliki *built-in chip WiFi* yang memungkinkan pengguna untuk menemukan dan terhubung ke *router* nirkabel. Sebagian besar perangkat *mobile*, *video game,* dan perangkat *standalone* lain juga mendukung *WiFi.*

2.5 Accelerometer

Accelerometer adalah sebuah sensor yang digunakan untuk mengukur percepatan suatu objek, baik statis maupun dinamis. Secara statis, *accelerometer* melakukan pengukuran terhadap gravitasi bumi. Sedangkan secara dinamis, *accelerometer* melakukan pengukuran terhadap percepatan atau perpindahan pada objek yang bergerak [5].

Alat ini bekerja berdasarkan hukum fisika. Dijelaskan bahwa apabila suatu konduktor digerakkan melalui suatu medan magnet, atau jika suatu medan magnet digerakkan melalui suatu konduktor, maka akan timbul suatu tegangan induksi pada kondiktor tersebut.

Accelerometer akan mengukur percepatannya secara langsung ketika bergerak secara horizontal yang dikarenakan oleh pergerakan horizontal. Contoh pergerakan sumbu pada *accelerometer* ditunjukkan pada Gambar 2.1, dimana sumbu X pada grafik merupakan satuan detik, dan sumbu Y merupakan nilai dari ketiga sumbu *accelerometer*.



Gambar 2.1 Grafik Pergerakan Sumbu *Accelerrometer*

Gambar 2.2 Orientasi sumbu *accelerometer* pada perangkat *mobile*

Pada *smartphone* terdapat pula sensor *accelerometer* yang terdiri dari tiga sumbu yaitu x, y, dan z seperti pada Gambar 2.2. Penggunaan *accelerometer* dalam aplikasi ini adalah untuk mendeteksi pergerakan pengguna apakah diam di tempat atau sedang berjalan.

Dalam penentuan posisi menggunakan *smartphone*, data yang dihasilkan *accelerometer* biasanya digunakan untuk *step detection* yang kemudian diolah untuk memperkirakan pergerakan dari pengguna.

2.6 Magnetometer

Magnetometer adalah suatu sistem atau perangkat yang bekerja atas dasar pendeteksian gaya magnet bumi. Biasanya *magnetometer* digunakan untuk menentukan arah mata angin.

Gambar 2.3 Orientasi *magnetometer* pada *smartphone*

Pada perangkat *mobile, magnetometer* memiliki *output* berupa besar medan magnet bumi yang diukur dalam tiga sumbu yang dapat digunakan untuk menentukan sudut arah hadap seperti pada Gambar 2.3. Dalam hal *positioning*, sensor ini digunakan untuk menentukan arah ketika pengguna sedang menuju suatu tempat. Untuk menghindari kesalahan pengukuran pada keadaan sensor miring, biasanya penggunaan sensor *magnetometer* akan digabungkan dengan *output* dari *accelerometer* [1].

2.7 PHP

PHP adalah singkatan dari "PHP: Hypertext Prepocessor", yaitu bahasa pemrograman yang digunakan secara luas untuk penanganan pembuatan dan pengembangan sebuah situs web dan dapat digunakan bersamaan dengan HTML. PHP diciptakan oleh Rasmus Lerdorf pertama kali tahun 1994. Pada awalnya PHP adalah singkatan dari *Personal Home Page Tools*. Selanjutnya diganti menjadi FI. Sejak versi 3.0, nama bahasa ini diubah menjadi "PHP: Hypertext Prepocessor" dengan singkatannya PHP. PHP versi terbaru adalah versi ke-5. Berdasarkan survei oleh Netcraft pada bulan Desember 1999, lebih dari sejuta laman menggunakan PHP, di antaranya adalah NASA, Mitsubishi, dan RedHat. PHP dalam Tugas Akhir ini akan digunakan untuk membuat *web service* yang digunakan untuk menyimpan maupun mengolah data yang dikirim melalui aplikasi Android [6].

2.8 JSON

JSON (*JavaScript Object Notation*) adalah format pertukaran data yang ringan [1]. JSON mudah untuk dibaca dan ditulis oleh manusia dan memudahkan mesin untuk mengurai dan menciptakan. Format ini berdasar dari himpunan bagian bahasa pemrograman JavaScript standar ECMA-262 edisi ke-3, Desember 1999. JSON merupakan format teks yang benar-benar berbeda atau independen namun tetap menggunakan konvensi yang mudah dikenali bagi *programmer* yang sudah terbiasa dengan bahasa C/C++, C#, Java, JavaScript, Perl, Python, dan bahasa lainnya. Sifat JSON yang seperti ini menjadikannya format pertukaran data yang ideal.

JSON digunakan untuk format dari data yang dikirimkan baik dari aplikasi Android ke PHP maupun sebaliknya, karena JSON akan mempermudah mengirim dan menerjemahkan data dari sistem yang menggunakan beberapa bahasa pemrograman yang berbeda .

[1] JSON, accessed Desember 1, 2015. http://www.json.org/.

2.9 MySQL

MySQL adalah sebuah perangkat lunak sistem manajemen basis data SQL (*database management system*) atau DBMS yang *multithread*, multi-pengguna, dengan sekitar 6 juta instalasi di seluruh dunia. MySQL AB membuat MySQL tersedia sebagai perangkat lunak gratis dibawah lisensi GNU *General Public License* (GPL), tetapi mereka juga menjual di bawah lisensi komersial untuk kasus-kasus dimana penggunaannya tidak cocok dengan penggunaan GPL [2].

MySQL sebenarnya merupakan turunan salah satu konsep utama dalam basis data sejak lama, yaitu SQL (*Structured Query Language*). SQL adalah sebuah konsep pengoperasian *database*, terutama untuk pemilihan atau seleksi dan pemasukan data, yang memungkinkan pengoperasian data dikerjakan dengan mudah secara otomatis.

Keuntungan dari MySQL adalah mendukung untuk standar SQL, berjalan di berbagai sistem informasi seperti Windows, Linux, FreeBSD dan Mac OS. Antarmuka MySQL juga tersedia untuk berbagai bahasa pemrograman antara lain C/C++, C#, Java, PHP, Phyton dan Ruby. MySQL berisi tabel-tabel yang digunakan untuk menyimpan data yang akan diproduksi oleh aplikasi dalam Tugas Akhir ini. Data tersebut akan diolah sedemikian rupa hingga memberikan suatu informasi yang berguna dalam aplikasi Tugas Akhir ini.

2.10 Algoritma *Enhanced Weighted k-Nearest Neighbor* (EWKNN)

Algoritma k-NN adalah metode yang menggunakan algoritma *supervised*. *Supervised learning* bertujuan untuk menemukan pola baru dalam data dengan menghubungkan pola data yang sudah ada dengan data yang baru. Tujuan dari algoritma k-NN adalah untuk mengklasifikasi objek baru berdasarkan atribut

---

[2] "MySQL" Wikipedia, accessed December 1, 2015. http://id.wikipedia.org/wiki/MySQL.

dan data *training*. Dimana hasil dari data yang baru diklasifikasikan berdasarkan mayoritas dari kategori pada k-NN. Algoritma k-NN menggunakan klasifikasi ketetanggaan untuk menentukan nilai prediksi dari data yang baru menggunakan nilai *k* sebagai jumlah ketetanggaan yang diambil. Untuk algoritma dari k-NN akan dijelaskan seperti berikut [3].

1. Terdapat sekumpulan *dataset* dalam *database.*
2. Sebuah bilangan bulat positif k dan sampel data baru ditentukan.
3. Pilih k data dalam *database* yang memiliki jarak terdekat dengan sampel baru tersebut.
4. Cari kelas yang paling umum dengan sampel baru tersebut.
5. Kelas yang paling umum merupakan kelas dari sampel baru tersebut.

Sebagai contoh kasus, terdapat 10 data yang terbagi atas dua kategori. Perbedaan data direpresentasikan dengan warna yang berbeda seperti Gambar 2.4.

Gambar 2.4 Contoh *dataset*

Kemudian ditambahkan data baru yang akan diuji seperti Gambar 2.5.

Gambar 2.5 Contoh *dataset* dengan data *testing*

Jika nilai dari *k* adalah 3, maka terdapat dua kelas hijau dan satu kelas biru yang terdekat dengan data *testing*. Sehingga data baru tersebut diklasifikasikan sama dengan kelas yang berwarna hijau seperti digambarkan pada Gambar 2.6.

Gambar 2.6 Hasil k-NN

EWKNN adalah modifikasi dari klasifikasi k-NN, dimana perbedaannya adalah k-NN menggunakan nilai *k* yang statis dan ditentukan dari awal, sedangkan EWKNN menggunakan nilai *k* yang dinamis. Sebagai contoh jika *k* yang ditentukan adalah 5, maka perkiraan posisi akan dihitung dari 5 jarak terdekat dengan data baru. Saat nilai *k* telah ditentukan secara statis, beberapa ketetanggaan terkadang memiliki jarak yang cukup jauh dengan data baru. Jika *data sample* yang memiliki jarak terdekat dengan data baru hanya berjumlah 3 ketetanggaan, tentu saja perkiraan posisi menggunakan *k* sebanyak 3 akan jauh lebih akurat dibandingkan menggunakan *k* sebanyak 5. Sehingga dengan ditentukannya nilai *k* secara dinamis, diharapkan dapat meningkatkan akurasi dari klasifikasi itu sendiri.

Pada EWKNN, jumlah *k* yang dinamis ditentukan melalui pemangkasan *dataset* menggunakan suatu *threshold* dimana jarak yang kurang dari *threshold* akan dipilih sebagai ketetanggaan. *Threshold* ini ditentukan menggunakan Persamaan 2.1, yang merupakan rata-rata dari selisih jarak (S_G) suatu *data sample* terdekat (D_1) dengan *data sample* lainnya (D_G), dengan *G* merupakan jumlah *data sample*.

$$\boldsymbol{E(S)} = \frac{(\boldsymbol{S_2+S_3+\cdots+S_G})}{\boldsymbol{G-1}} \quad (2.1)$$

2.11 Perhitungan Pendeteksi Pergerakan

Pendeteksi pergerakan adalah sebuah teknik yang digunakan untuk mendeteksi dan mengetahui apakah suatu pengguna sedang bergerak atau tidak. Teknik ini digunakan dengan cara menghitung nilai signifikansi dari nilai perubahan dari sumbu x, y, dan z *accelerometer*. Hasil yang didapat dari sensor *accelerometer* kemudian diolah menggunakan *low-pass filter* untuk menghilangkan gaya gravitasi di dalamnya. Jika nilai tersebut melebihi dari *threshold*, maka sistem akan mendeteksi bahwa ada pergerakan. Jika tidak, maka tidak terdapat pergerakan[3].

2.12 Android

Android adalah *software stack* untuk perangkat *mobile* yang mencakup sistem operasi dan *middleware*. Android menyediakan platform terbuka bagi para pengembang untuk menciptakan aplikasi mereka sendiri untuk digunakan oleh bermacam peranti bergerak [4]. SDK Android menyediakan alat dan API yang diperlukan untuk mulai mengembangkan aplikasi pada *platform* Android mengguna kan bahasa pemrograman Java. SDK Android merupakan alat kompilasi kode bersama dengan data dan sumber daya *file* ke dalam paket Android dan *file* arsip dengan akhiran APK. Semua kode dalam *file* APK tunggal dianggap satu aplikasi.

2.13 Komponen Android

Komponen aplikasi adalah bagian penting dari aplikasi Android. Setiap komponen memiliki titik fokus yang berbeda. Setiap komponen tidak selalu bergantung antara yang satu dengan

---

[3] "Sensors Motion" Android Developer, accessed December 1, 2015. https://developer.android.com/guide/topics/sensors/sensors_motion.html

[4] "Android: Sistem Operasi Smartphone" Ubaya, accessed December 1, 2015. http://www.ubaya.ac.id/2014/content/articles_detail/7/Android--Sistem-Operasi-pada-Smartphone.html

yang lainnya, namun masing-masing merupakan entitas sendiri dan memainkan peran khusus, masing-masing membantu mendefinisikan perilaku keseluruhan aplikasi. Terdapat empat komponen aplikasi Android, yaitu *activities, services, content providers*, dan *broadcast receivers* [5] . Masing-masing dari komponen tersebut akan dijelaskan seperti berikut.

a. Activities

Activities merepresentasikan *single screen* dengan antarmuka. Sebagai contoh, aplikasi pesan elektronik memiliki satu *activity* yang menunjukkan *list* dari pesan elektronik baru serta *activity* lainnya berfungsi untuk membaca pesan elektronik tertentu. Meskipun kedua *activities* tersebut dapat bekerja bersamaan pada aplikasi pesan elektronik, setiap *activity* bersifat independen atau berdiri sendiri dan tidak bergantung dengan *activity* lainnya[6].

b. Service

Service adalah komponen yang berjalan di *background* untuk melakukan operasi tertentu atau melakukan pekerjaan untuk proses *remote*. *Service* tidak menyediakan antarmuka pengguna. Sebagai contoh, *service* mungkin memutar musik di latar belakang saat pengguna berada dalam aplikasi yang berbeda, atau mungkin mengambil data melalui jaringan tanpa menghalangi interaksi pengguna dengan suatu *activity.*

c. Broadcast Receivers

Broadcast receivers merupakan komponen yang merespon pengumuman *broadcast* ke seluruh sistem. Terdapat banyak *broadcast* yang berasal dari sistem, misalnya *broadcast* yang mengumumkan bahwa *screen* telah dimatikan, baterai rendah, dan lain-lain. Aplikasi juga dapat memulai sistem *broadcast* misalnya

---

[5] "Application Fundamentals" Android Developer, accessed May 20, 2016. http://developer.android.com/guide/components/fundamentals.html

[6] "Activity" Android Developer, accessed May 20, 2016. http://developer.android.com/guide/components/fundamentals.html.

untuk membiarkan aplikasi lain mengetahui bahwa beberapa data telah diunduh ke perangkat dan tersedia bagi mereka untuk dapat digunakan. Secara umum, *broadcast receivers* merupakan pintu gerbang untuk komponen lain berinteraksi satu sama lain dan dimaksudkan untuk melakukan jumlah kerja yang sangat minimum.

d. Fragment

Fragment merupakan perilaku atau sebagian dari antarmuka pengguna dalam sebuah *activity.* Beberapa *fragment* dapat digabungkan ke dalam satu *activity* untuk membangun *multi-pane interface* dan dapat menggunakannya kembali *fragment* tersebut ke dalam beberapa *activities. Fragment* dapat digunakan sebagai bagian dari modul pada sebuah *activity,* yangmana memiliki siklus hidup sendiri, menerima *input events* sendiri, dan dapat ditambah atau dipindah saat *activity* sedang berjalan (semacam *sub activity* yang dapat digunakan kembali pada *activity* yang berbeda).

e. SensorEventListener

SensorEventListener merupakan *public interface* yang terdapat pada Android yang digunakan untuk menerima notifikasi dari *sensorManager* ketika nilai sensor yang didapatkan berubah. *SensorEventListener* mengambil data sensor dari *hardware* Android. *SensorEventListener* memiliki dua *public method*, yaitu *onAccuracyChanged* dan *onSensorChanged.* Kedua *public method* tersebut memiliki fungsi yang berbeda dan akan dijelaskan sebagai seperti berikut.

- *onAccuracyChanged* digunakan ketika akurasi dari sensor mengalami perubahan.
- *onSensorChanged* digunakan ketika nilai dari sensor mengalami perubahan.

BAB III
ANALISIS DAN PERANCANGAN

Pada bab ini akan dijelaskan hal-hal yang berkaitan dengan perancangan sistem yang akan dibuat dalam Tugas Akhir ini, dimulai dari deskripsi umum mengenai perangkat lunak yang akan dibuat, perancangan proses-proses yang ada, dan arsitektur umum sistem.

3.1 Deskripsi Umum

Dalam Tugas Akhir ini dibangun sebuah perangkat lunak sistem pelacakan pergerakan di dalam gedung Teknik Informatika. Aplikasi ini berjalan di atas sistem operasi Android yang memiliki sensor berupa *accelerometer* dan *magnetometer*. Hal-hal yang akan diukur adalah posisi awal, perpindahan, dan arah pergerakan pengguna. Sensor *accelerometer* dan *magnetometer* digunakan untuk menentukan arah dan perpindahan pengguna, sedangkan posisi awal pengguna ditentukan oleh klasifikasi *WiFi* yang tertangkap oleh sistem saat itu juga.

Tahap awal pada pengerjaan Tugas Akhir ini adalah membagi gedung Teknik Informatika ke dalam beberapa bagian atau titik yang disebut *Reference Points* (RPs). Masing-masing RP akan memuat informasi berupa koordinat dan daftar *Access Points* (APs) yang diterima oleh *smartphone*, yaitu berupa *Received Signal Strength* (RSS) setiap *WiFi* yang terdapat pada titik tersebut. Kemudian dilakukan pengumpulan *data sample* kekuatan sinyal *WiFi* yang ditangkap oleh *smartphone* pada setiap titik yang telah ditentukan dengan ukuran 240 cm x 240 cm. Proses pengumpulan *data sample* akan dilakukan dengan cara melakukan scanning RSS beberapa kali karena mempertimbangkan fluktuasi sinyal *WiFi* yang dipengaruhi oleh kondisi lingkungan sekitar. Data seluruh RSS yang dikumpulkan akan disimpan ke dalam *database*.

RSS yang ditangkap memiliki informasi berupa BSSID (*Basic Service Set Identifier*) dan kekuatan sinyalnya dengan

satuan dBm (*Decibel-miliwatt*). Pada umumnya kekuatan sinyal *WiFi* yang ditangkap akan bernilai negatif, dikarenakan kekuatan sinyal untuk jaringan yang ditransmisikan nilai-nilai dBm positif. BSSID merupakan sebuah alamat unik berupa 12 karakter *alphanumeric* yang mengidentifikasi *Access Point* yang menciptakan jaringan nirkabel. Setiap kali *scanning* akan didapatkan sekumpulan RSS yang kemudian disebut *Reference Point* (RP) dan digunakan untuk menentukan posisi pengguna.

Setelah *data sample* di seluruh titik terkumpul, tahap berikutnya pengguna dapat mengetahui lokasi awalnya dengan mengirimkan RSS baru yang terdeteksi pada posisinya saat itu juga. Kemudian pengguna dapat berjalan dan aplikasi akan melakukan *update* posisi pengguna melalui algortima deteksi pergerakan pengguna. Algoritma deteksi pergerakan pengguna dilakukan dengan mengolah data yang diterima oleh sensor *accelerometer* dan *magnetometer* pada *smartphone*. Jika pengguna terdeteksi melakukan suatu pergerakan dengan arah tertentu, maka aplikasi akan mengubah posisi pengguna ke koordinat yang baru sesuai hasil perhitungan.

3.2 Arsitektur Sistem

Dalam sistem ini setiap komponen memiliki peran masing-masing. Komponen sistem terdiri sebagai berikut:

a. Pengguna

*P*engguna adalah orang yang menggunakan *smartphone* berbasis Android.

b. Device

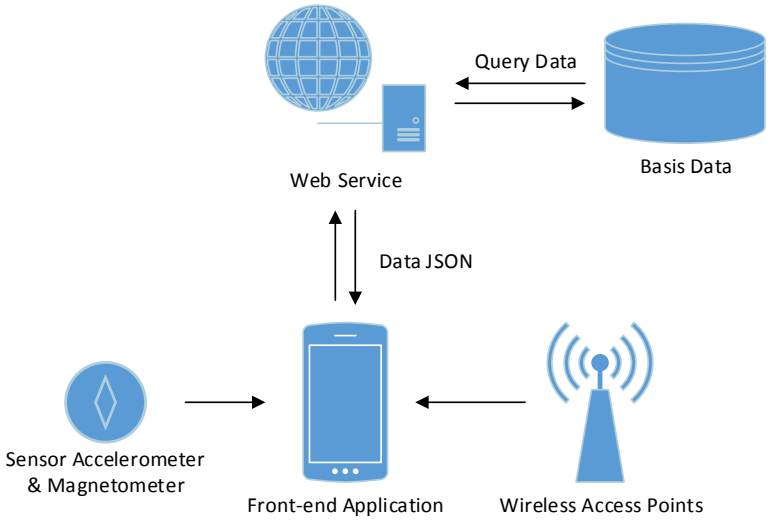
Device yang digunakan adalah *smartphone* berbasis Android minimal versi 5.0 dengan merk Asus tipe Zenfone 2 ZE551ML.

c. Sensor Accelerometer

Sensor *accelerometer* merupakan *hardware* yang melekat pada *smartphone* berbasis Android yang biasa digunakan untuk mengukur percepatan.

d. Sensor Magnetometer

Sensor *magnetometer* adalah *hardware* yang melekat pada *smartphone* yang biasa digunakan untuk menentukan arah.



Gambar 3.1 Arsitektur Sistem

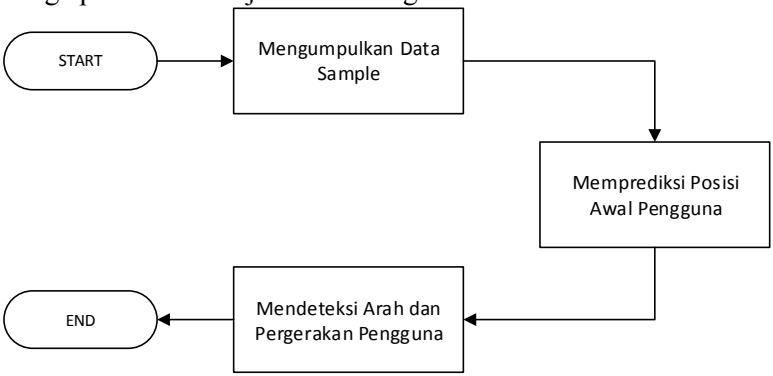
Adapun perancangan arsitektur sistem yang telah dibuat dapat dilihat pada Gambar 3.1. Berdasarkan Gambar 3.1, perancangan alur arsitektur sistem Tugas Akhir ini adalah sebagai berikut.

1. *Smartphone* dipegang oleh pengguna dalam kondisi *steady hand.*
2. Pengguna berdiri di suatu titik lokasi selama beberapa detik, dan aplikasi akan mengumpulkan sinyal *WiFi* pada titik tersebut.
3. Aplikasi akan mendeteksi posisi pengguna dengan melakukan klasifikasi data *WiFi* yang telah terkumpul dengan data *WiFi* yang terdapat dalam basis data sistem.
4. Setelah menentukan posisi awal, aplikasi akan mendeteksi pergerakan yang dilakukan oleh pengguna.

5. Jika pengguna menekan tombol *refresh*, maka aplikasi akan melakukan deteksi ulang terhadap posisi pengguna saat itu juga.

3.3 Perancangan Proses

Pada subbab ini akan dibahas secara mendetail dari rancangan proses sistem *realtime tracking indoor positioning*. Terdapat tiga proses utama dalam pembangunan sistem ini yang akan dijelaskan pada Gambar 3.2. Penjelasan lebih lanjut tentang ketiga proses akan dijelaskan sebagai berikut.



Gambar 3.2 Perancangan Proses Sistem

3.3.1 Proses Pengumpulan *Data Sample*

Proses ini merupakan pengumpulan *data sample* yang akan digunakan untuk melakukan perhitungan estimasi lokasi pengguna nantinya. Data yang disimpan antara lain BSSID, kekuatan sinyal yang didapat, *access point* dari sinyal tersebut dan juga posisinya dalam koordinat kartesius (sumbu x, y, dan z).

Sebelum mengumpulkan *data sample*, dibutuhkan peta yang menggambarkan gedung dalam skala tertentu, dalam studi kasus Tugas Akhir ini adalah gedung Teknik Informatika ITS lantai tiga. Dalam gedung tersebut, terdapat ruangan-ruangan yang akan dibagi lagi menjadi ukuran 240 cm x 240 cm untuk diambil *data sample* RSS berupa BSSID dan kekuatan sinyalnya.

Pengumpulan *data sample* untuk setiap posisi dalam ruangan dilakukan beberapa kali untuk diambil BSSID yang stabil beserta kekuatan sinyal rata-ratanya, dikarenakan sinyal *WiFi* yang diterima bersifat fluktuatif. *Data sample* yang berhasil didapat selanjutnya dimasukkan ke dalam *database* sistem untuk digunakan pada proses memprediksi lokasi pengguna.

Hampir setiap gedung pasti memiliki level ketinggian lebih dari satu lantai, termasuk gedung Teknik Informatika ITS. Dan setidaknya di setiap lantai tersebut pastilah ada beberapa *access point* yang terpasang. Sistem yang akan dibangun pada Tugas Akhir ini tidak dapat membedakan *access point* yang terpasang di setiap lantai. Dengan kata lain, pada saat pengguna melakukan *scanning,* sistem akan memasukkan seluruh RSS *WiFi* yang tertangkap oleh aplikasi, tidak peduli RSS *WiFi* tersebut berasal dari lantai lain.

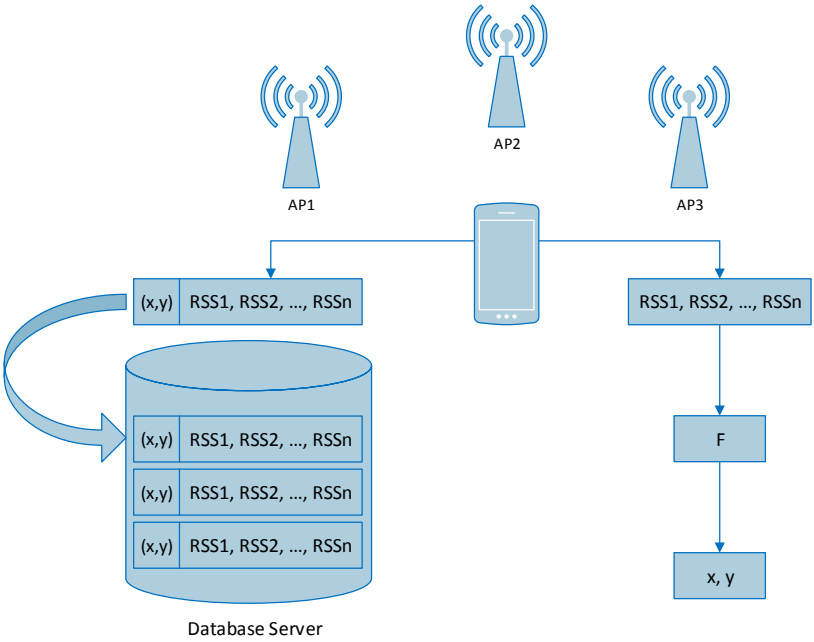
3.3.2 Proses Memprediksi Lokasi Awal Pengguna

Proses ini merupakan proses memprediksi lokasi pengguna berdasarkan kekuatan sinyal yang diterima saat itu terhadap *data sample* yang telah dikumpulkan. Untuk melakukan prediksi lokasi pengguna, dibutuhkan beberapa data RSS yang terteangkap oleh perangkat pengguna yang disebut *Received Signal Strength Vector* atau RSSV seperti pada Gambar 3.3.

Proses mempredikasi lokasi awal pengguna pada Tugas Akhir ini menggunakan metode klasifikasi *Enhanced Weighted k-Nearest Neighbors* (EWKNN). Metode ini merupakan metode kNN yang telah dimodifikasi. Secara garis besar, alur kerjanya dapat dilihat pada Gambar 3.4.

Algoritma dimulai dengan membandingkan selisih RSS *WiFi* baru dengan setiap set RSS *WiFi* yang tersimpan pada basis data sistem seperti pada Persamaan 3.1.

$$\boldsymbol{D_i = \sum_{j=1}^{N} \left| A_j - R_{i,j} \right|, i = 1, 2, 3, \ldots, L} \qquad (3.1)$$



Gambar 3.3 Proses Prediksi Lokasi

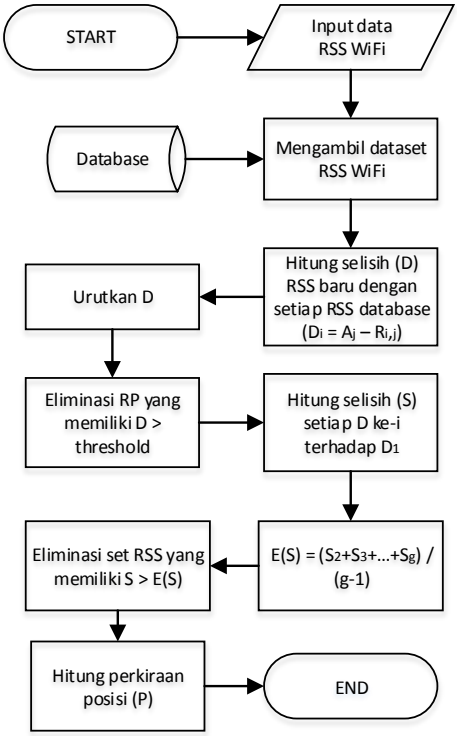
dimana *Aj* merupakan RSS ke-*j* dari AP baru dan *Ri*,*j* merupakan RSS ke-*j* pada AP ke-*i* dalam *database*. Sedangkan *L* adalah jumlah set RSS (RP) yang terdapat pada *database*.

Kemudian D diurutkan dari yang terkecil, dan ditentukan suatu *threshold* sehingga D yang memiliki nilai lebih besar dari *threshold* akan dibuang. Sejumlah RP yang tersisa selanjutnya dihitung rata-rata perbedaan D antar RP menggunakan Persamaan 3.2, dimana *G* merupakan jumlah RP yang tersisa, dan S_G adalah perbedaan antara D_1 dan D_g ($g = 2,\ldots,G$).

$$\boldsymbol{E(S)} = \frac{(\boldsymbol{S_2}+\boldsymbol{S_3}+\cdots+\boldsymbol{S_G})}{\boldsymbol{G-1}} \qquad (3.2)$$

Setelah didapat rata-rata perbedaan *D* tiap RP sebagai E(S), maka RP yang memiliki *S* lebih kecil dari *E(S)* akan dijadikan sebagai kandidat, dan sisanya akan dibuang. Jumlah RP yang tersisa inilah yang dijadikan sebagai *K*. Langkah terakhir adalah menghitung estimasi posisi pengguna menggunakan rata-rata koordinat setiap RP yang menjadi kandidat seperti pada Persamaan 3.3.

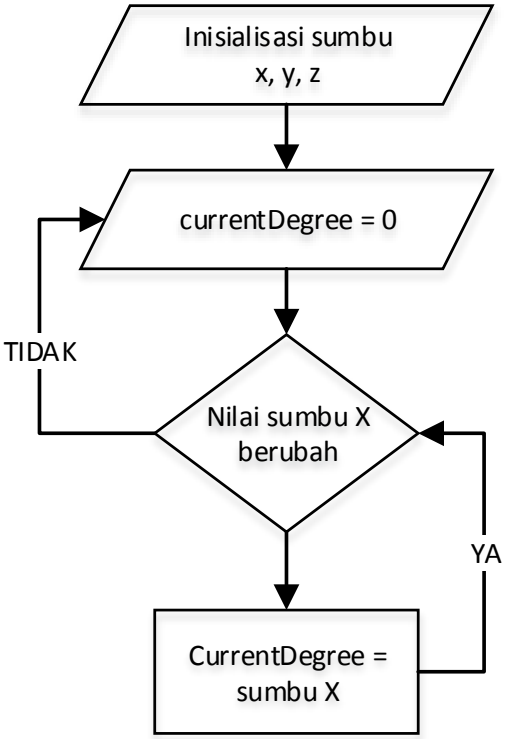
$$\boldsymbol{P} = \frac{\frac{\mathbf{1}}{\boldsymbol{D_1}}\boldsymbol{L(RP_1)} + \frac{\mathbf{1}}{\boldsymbol{D_2}}\boldsymbol{L(RP_2)} + \ldots + \frac{\mathbf{1}}{\boldsymbol{D_K}}\boldsymbol{L(RP_K)}}{\frac{\mathbf{1}}{\boldsymbol{D_1}} + \frac{\mathbf{1}}{\boldsymbol{D_2}} + \cdots + \frac{\mathbf{1}}{\boldsymbol{D_K}}} \qquad (3.3)$$



Gambar 3.4 Diagram Alir Algoritma EWKNN

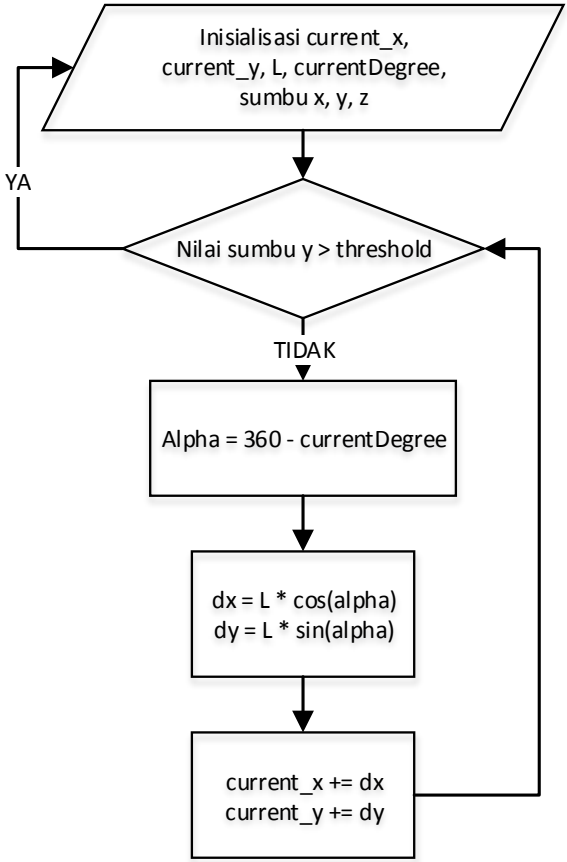
3.3.3 Proses Mendeteksi Arah dan Pergerakan Pengguna

Pendeteksian arah dan pergerakan pengguna didapatkan dari pembacaan sensor *accelerometer* dan *magnetometer*. Sensor *magnetometer* digunakan untuk menetukan pengguna sedang menghadap ke arah mana. Untuk alur kerja dari sensor *magnetometer* dapat dilihat pada Gambar 3.5. Karena sensor *magnetometer* digunakan untuk menentukan arah hadap pengguna secara horizontal, maka orientasi sumbu x saja yang perlu diperhatikan.



Gambar 3.5 Diagram Alir Kerja *Magnetometer*

Sedangkan sensor *accelerometer* pada dasarnya akan mendeteksi percepatan terhadap sumbu x, y, dan z. Karena nilai yang dihasilkan dari sensor *accelerometer* masih dipengaruhi oleh gaya gravitasi, maka perlu dilakukan *preprocessing* terhadap nilai keluaran dari *accelerometer*. Langkah *preprocessing* yang digunakan untuk menghilangkan gaya gravitasi adalah *low pass filter*.



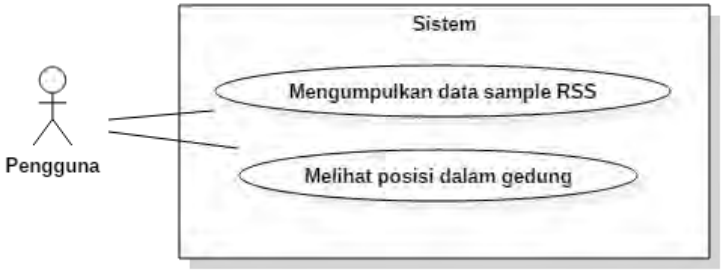
Gambar 3.6 Diagram Alir Kerja *Accelerometer*

Sensor *accelerometer* digunakan untuk mengetahui apakah pengguna sedang bergerak maju atau tidak, maka yang perlu diperhatikan adalah nilai dari sumbu y apakah melebihi suatu batas yang telah ditentukan atau tidak. Jika nilai sumbu y lebih kecil dari batas yang ditentukan, maka pengguna dianggap sedang berjalan. Kemudian *marker* yang menandakan posisi pengguna saat itu akan berpindah dengan jarak tertentu ke arah yang telah ditentukan oleh sensor *magnetometer*. Alur kerja dari *accelerometer* dijelaskan dalam diagram alir Gambar 3.6.

Saat pengguna berjalan menggunakan aplikasi, posisi *smartphone* harus berada dalam kondisi *steady hand.* Kondisi *steady hand* dalam hal ini adalah *smartphone* selalu berada dalam posisi mendatar. Hal ini dikarenakan *accelerometer* akan memprediksi pergerakan maju pengguna, sehingga jika posisi *smartphone* tidak selalu mendatar, akan mempengaruhi nilai sumbu y yang berubah secara tidak normal. Perubahan nilai sumbu y yang tidak normal ini nantinya juga akan mempengaruhi *step detection* pada sistem, dimana jika guncangan atau perubahan nilai pada sumbu y melebihi *threshold* tidak akan dianggap sebagai sebuah langkah. Tentu saja sistem tidak akan menunjukkan performa yang cukup bagus jika kondisi ini tidak terpenuhi.

3.4 Perancangan Kasus Penggunaan

Adapun diagram kasus penggunaan yang telah dibuat dapat dilihat pada Gambar 3.7. Berdasarkan Gambar 3.7, aktor pada aplikasi adalah pengguna.



Gambar 3.7 Diagram Kasus Penggunaan

Pengguna adalah orang yang menggunakan aplikasi ini. Pengguna dapat mengumpulkan data RSS *WiFi* setiap titik kemudian memasukkan ke dalam *database* sistem. Pengguna juga dapat melihat posisinya pada aplikasi yang disajikan dalam bentuk peta gedung Teknik Informatika. Posisi tersebut akan terus berubah mengikuti pergerakan pengguna. Untuk penjelasan lengkap mengenai kasus penggunaan pada aplikasi ini dapat dilihat pada Tabel 3.1.

Tabel 3.1 Deskripsi Kasus Penggunaan

No	Kode	Nama	Keterangan
1.	UC-01	Mengumpulkan data RSS	Pengguna dapat menyimpan data RSS sebagai dataset sistem
2.	UC-02	Menampilkan posisi dalam gedung	Pengguna dapat mengetahui lokasi keberadaannya pada saat itu

3.4.1 Deskripsi Kasus Penggunaan UC-01

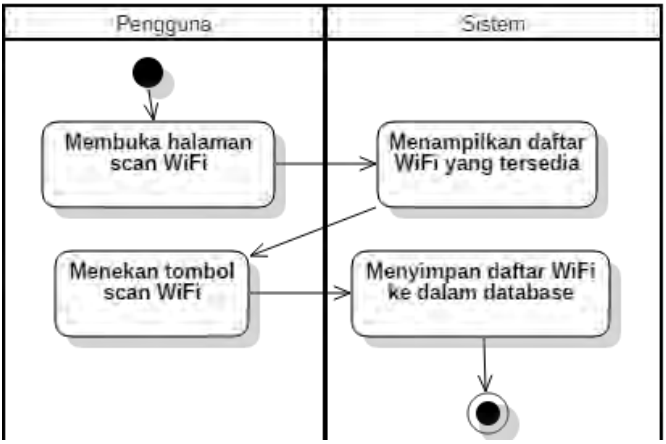
Kasus penggunaan UC-01 merupakan kasus penggunaan mengumpulkan data RSS *WiFi* yang berhasil ditangkap oleh aplikasi. Alur penggunaan UC-01 dapat dilihat pada Tabel 3.2 dan diagram aktivitas kasus seperti pada Gambar 3.8.

Tabel 3.2 Rincian Alur Kasus Penggunaan UC-01

Nama Use Case	**Mengumpulkan data RSS**
Nomor	UC-01
Aktor	Pengguna
Kondisi Awal	Data RSS *WiFi* belum tersimpan
Kondisi Akhir	Data RSS *WiFi* tersimpan dalam *database*

Alur Kejadian Normal	
Pengguna	Sistem
1. Pengguna membuka halaman Scan WiFi 3. Pengguna menekan tombol Scan	2. Sistem menampilkan daftar *WiFi* di sekitar posisi pengguna 4. Sistem menyimpan daftar *WiFi* ke dalam *database* sistem
Alur Kejadian Alternatif	
-	

Rincian alur kasus mengumpulkan data RSS *WiFi* dimulai saat pengguna membuka halaman Scan *WiFi*, kemudian aplikasi akan menampilkan *form* dan daftar RSS *WiFi* yang berhasil ditangkap oleh *smartphone. Form* tersebut berisi nama lokasi dan koordinat yang akan diambil RSS *WiFi* sebagai *data sample*. Pengguna diharuskan mengisi *form* terlebih dahulu sebelum melakukan pengumpulan data. Kemudian aplikasi akan mengirim daftar RSS ke dalam *database* sistem setiap kali pengguna menekan tombol "Scan".



Gambar 3.8 Diagram Aktivitas Kasus Penggunaan UC-01

3.4.2 Deskripsi Kasus Penggunaan UC-02

Kasus penggunaan kode UC-02 merupakan kasus penggunaan melihat posisi pengguna dalam gedung. Alur dimulai saat pengguna membuka halaman peta. Pertama kali halaman peta dibuka, sistem akan mengumpulkan daftar RSS *WiFi* di sekitar lokasi pengguna.

Kemudian sistem melakukan klasifikasi RSS *WiFi* tersebut terhadap *data sample* pada *database.* Setelah didapatkan perkiraan koordinat pengguna, sistem akan menampilkan posisi pengguna pada peta dalam bentuk *marker.*

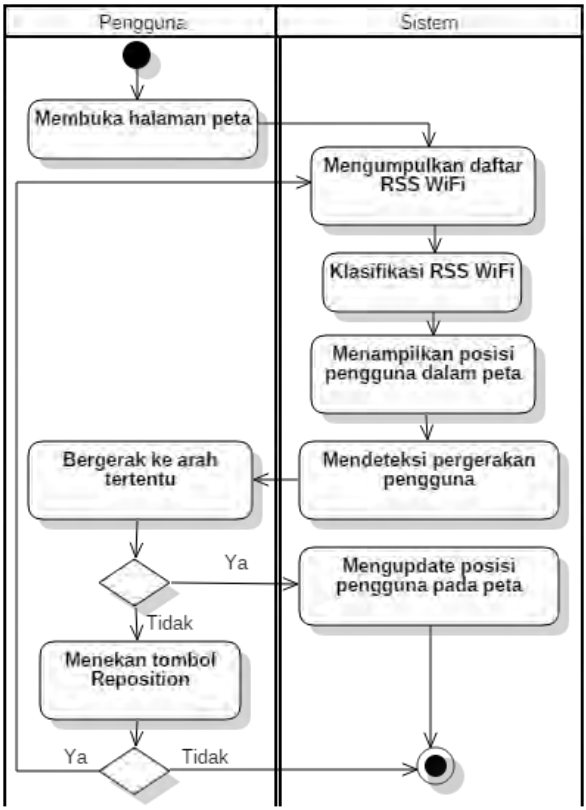
Tabel 3.3 Rincian Alur Kasus Penggunaan UC-02

Nama Use Case	**Menampilkan Posisi Pengguna**
Nomor	UC-02
Aktor	Pengguna
Kondisi Awal	Sistem belum menampilkan posisi pengguna
Kondisi Akhir	Sistem menampilkan posisi pengguna dalam peta
Alur Kejadian Normal	
Pengguna	Sistem
1. Pengguna membuka halaman peta	
	2. Sistem mengambil daftar RSS *WiFi* pada posisi pengguna
	3. Sistem melakukan klasifikasi RSS *WiFi*
	4. Sistem menampilkan posisi awal pengguna pada peta
	5. Sistem mendeteksi pergerakan pengguna menggunakan sensor *accelerometer* dan *magnetometer*
A.6. Pengguna bergerak kearah tertentu	

A.7. Pengguna menekan tombol “Reposition”	
Alur Kejadian Alternatif	
A.6. Sistem mendeteksi pergerakan pengguna	
Pengguna	Sistem
	A.6.1. Sistem melakukan update posisi pengguna pada peta A.6.2. Selesai
A.7. Pengguna menekan tombol “Reposition”	
Pengguna	Sistem
	A.7.1. Sistem mengambil data RSS *WiFi* di sekitar posisi pengguna A.7.2. Sistem melakukan klasifikasi RSS *WiFi* A.7.3. Sistem melakukan update posisi pengguna pada peta A.7.4. Selesai

Kemudian saat posisi awal pengguna telah ditentukan, sistem mulai melakukan pendeteksian pergerakan pengguna menggunakan sensor *accelerometer* dan *magnetometer*. Apabila pengguna melakukan pergerakan ke arah tertentu, maka sistem akan melakukan *update* posisi pengguna pada peta dengan menggerakkan *marker* ke arah tertentu dan jarak tertentu sesuai pergerakan pengguna.

Pengguna juga bisa melakukan *reposition* dengan menekan tombol “Reposition”, dan sistem akan melakukan klasifikasi ulang untuk mendapatkan posisi baru dari pengguna. Rincian alur kasus menampilkan posisi pengguna secara lengkap dapat dillihat pada Tabel 3.3 dan diagram aktivitas kasus penggunaan UC-02 dijelaskan pada Gambar 3.9.



Gambar 3.9 Diagram Aktivitas Kasus Penggunaan UC-02

3.5 Percancangan *Web Service*

Pada subbab ini akan dijelaskan bagaimana rancangan *web service* pada sistem yang dibangun. *Web service* dalam hal ini berguna untuk melakukan *transfer* data antara *client* (*smartphone* pengguna) dengan *server*. *Web service* yang digunakan pada sistem ini adalah RESTful *web service* menggunakan kerangka kerja CodeIgniter. Semua perintah, fungsi, maupun protokol yang digunakan dalam *web service* selalu diakses melalui *host* yang

sama dan semua data yang ditransfer selalu dalam format JSON. Untuk lebih detailnya akan dijelaskan pada bagian-bagian di bawah ini.

3.5.1 Parameter *Request* dan *Response*

Dalam penggunaannya, ada dua macam *method* yang terdapat pada *web service* saat *client* melakukan *request,* yaitu *GET* dan *POST* yang akan dijelaskan secara rinci pada Tabel 3.4. Secara umum, *response* yang dikirim oleh *server* selalu berupa JSON dengan tiga bagian, yaitu *code, message,* dan *data.* Masing-masing bagian memiliki fungsi tertentu yang akan dijelaskan pada Tabel 3.5.

Tabel 3.4 *Method Request* pada *Web Service* Sistem

Method	Deskripsi
GET	Digunakan untuk mengambil *resource* atau data yang pasti.
POST	Digunakan untuk membuat *resource* atau data tertentu, seperti mengambil *resource* dengan kriteria tertentu.

Tabel 3.5 Bagian dari *Response* JSON

Nama *Key*	Keteranan *Value*
code	Digunakan untuk mengetahui apakah operasi yang dilakukan oleh *web service* berhasil atau tidak. Jika gagal, maka *response* ini akan dijadikan sebagai pesan *eror*.
message	Merupakan pesan pada *response*, digunakan jika sistem ingin menampilkan suatu notifikasi tertentu.
data	Merupakan data utama pada *response,* saat sistem perlu memberikan umpan balik berupa data yang diminta oleh *client.*

```
{
   “code”:”777”,
   “message”:”Berhasil menambahkan position”,
   “data”:17
}
```

Gambar 3.10 Contoh *Response* JSON Sukses

```
{
   “code”:”444”,
   “message”:”Gagal menambahkan position”,
   “data”:NULL
}
```

Gambar 3.11 Contoh *Response* JSON Gagal

Status code pada *response* terdiri dari dua macam, yaitu 444 dan 777 yang akan dijelaskan pada Tabel 3.6. Untuk contoh format *response* berupa JSON akan ditampilkan pada Gambar 3.10 dan Gambar 3.11.

Tabel 3.6 Status *Code* pada *Response* JSON

Code	Deskripsi
444	Sistem gagal memproses data, dengan mengembalikan *response* data berupa *NULL*.
777	Sistem berhasil memproses data, dan mengembalikan pesan atau data tertentu pada *client.*

3.5.2 Operasi pada Web Service

Terdapat beberapa operasi atau fungsi yang dijalankan pada sistem, dan semua operasi tersebut dilakukan dengan mengakses suatu *link* tertentu.

Gambar 3.12 Format URL *Web Service* Sistem

Karena *web service* ini dibangun menggunakan *framework* CodeIgniter, maka format *link* untuk mengakses suatu fungsi selalu sama, yaitu ditunjukkan pada Gambar 3.12 [7].

Tabel 3.7 *Controller* pada *Web Service* Sistem

Controller	Deskripsi
AccessPoints	Mengatur segala proses yang berhubungan dengan data *Access Point.*
Positions	Mengatur segala proses yang berhubungan dengan data *Position.*
ReceivedSignalStrengths	Mengatur segala proses yang berhubungan dengan data RSS.
Rooms	Mengatur segala proses yang berhubungan dengan data ruangan.
Classify	Mengatur segala proses yang berhubungan dengan klasifikasi.

Dalam *web service* ini yang sering digunakan adalah *controller* dan *resource*. *Resource* merupakan fungsi atau sekumpulan perintah yang digunakan untuk memproses data, sedangkan *controller* merupakan bagian yang berisi beberapa *resource.* Secara umum, seluruh *controller* dan *resource* yang digunakan dalam *web service* akan dijelaskan secara rinci pada Tabel 3.7 dan Tabel 3.8. Penjelasan detail operasi atau fungsi pada setiap *controller* akan dijabarkan pada subbab di bawah, yaitu menambahkan *access point,* memprediksi koordinat posisi awal pengguna, menambahkan posisi, menambahkan RSS, serta mendapatkan data seluruh ruangan.

[7] "RESTful Service in CodeIgniter" EnvatoTuts+, accessed June 21, 2016. http://code.tutsplus.com/tutorials/working-with-restful-services-in-codeigniter--net-8814

Tabel 3.8 *Resource* pada *Web Service* Sistem

Resource	Deskripsi
add	Fungsi untuk menambahkan data baru pada tabel tertentu. Fungsi ini digunakan di beberapa *controller*, dan fungsi ini memiliki parameter input yang berbeda-beda di setiap *controller* tergantung dari data yang dibutuhkan pada tabel tersebut.
get	Fungsi yang digunakan untuk mengambil data dari tabel tertentu.
mean	Fungsi yang megatur segala proses klasifikasi mulai dari membandingkan *data sample* hingga menghasilkan prediksi koordinat posisi pengguna.

3.5.2.1 Menambahkan *Access Point*

Fungsi ini merupakan fungsi pada *web service* yang digunakan untuk menambahkan setiap *access point* baru yang belum pernah ada dalam *database* sistem. Aplikasi *client* akan mengirimkan data berupa *JSONArray* pada saat pengguna melakukan *sampling* RSS, dan sistem akan menerimanya untuk kemudian ditambahkan ke dalam *database* jika *access point* tersebut belum terdaftar dalam *database server*. *Method* yang digunakan pada fungsi ini adalah HTTP *Post*.

Data *JSONArray* yang dikirim ke *web service* sebagai parameter *request* akan dijelaskan pada Tabel 3.9. Parameter *response* yang dikembalikan oleh *web service* apabila data telah dikirim berupa *JSONArray* dan akan dijelaskan pada Tabel 3.10.

Tabel 3.9 Parameter *Request* Menambahkan *Access Point*

Nama *Key*	Keterangan *Value*
bssid	Kode unik dari *access point.*
ssid	Nama dari *access point.*

Tabel 3.10 Parameter *Response* Menambahkan *Access Point*

Nama *Key*	Keterangan *Value*
code	1. Bernilai "444" jika data yang dikirim gagal ditambahkan pada *database* sistem . 2. Bernilai "777" jika data yang dikirim berhasil ditambahkan ke dalam *database* sistem.
message	1. Bernilai "Gagal menambahkan *Access Point*" jika data yang dikirim gagal ditambahkan. 2. Bernilai "Berhasil menambahkan *Access Point*" jika data yang dikirim berhasil ditambahkan ataupun telah ada dalam *database.*
data	Bernilai *NULL.*

3.5.2.2 Memprediksi Koordinat Posisi Awal

Fungsi ini merupakan fungsi yang digunakan untuk memprediksi posisi awal pengguna. Fungsi ini akan menjalankan kasifikasi EWKNN dengan membandingkan RSS baru terhadap seluruh RSS yang terdapat dalam *database* sistem. Sebagai *input* awal, aplikasi akan menangkap seluruh data RSS *WiFi* di sekitar pengguna saat itu, kemudian mengirimkannya ke *server* dalam bentuk *JSONArray*. *Method* yang digunakan pada fungsi ini adalah HTTP *Post.*

Data *JSONArray* yang dikirim ke *web service* sebagai parameter *request* berupa *JSONArray* dan akan dijelaskan pada Tabel 3.11.

Tabel 3.11 Parameter *Request* Memprediksi Posisi Awal

Name *Key*	Keterangan *Value*
bssid	Sebuah *string* yang berisi sekumpulan *bssid* dari seluruh RSS yang diterima oleh perangkat. Setiap *bssid* akan dipisahkan oleh ";;". Contoh: 90:f6:52:8b:1d:a8;;10:fe:ed:3b:33:e3;;.

Name *Key*	Keterangan *Value*
ss	Sebuah *string* yang memuat sekumpulan *signal strength* dari setiap *bssid* yang tertangkap oleh perangkat. Setiap *signal strength* juga akan dipisahkan oleh ";;". Contoh: -67;;-76;;.

Setelah data dikirim, *web service* akan mengembalikan *response* berupa data *JSONArray* dengan parameter yang akan dijelaskan pada Tabel 3.12.

Tabel 3.12 Parameter *Response* Memprediksi Posisi Awal

Nama *Key*	Keterangan *Value*
code	1. Bernilai "444" jika sistem gagal mendapatkan koordinat posisi awal. 2. Bernilai "777" jika sistem berhasil mendapatkan koordinat posisi awal.
message	1. Bernilai "Gagal mendapatkan posisi, *WiFi* tidak sesuai dengan sistem" jika RSS *WiFi* yang diterima tidak ada yang cocok dengan RSS *WiFi* pada *database* sistem. 2. Bernilai "Gagal mendapatkan posisi" ketika jarak RSS *WiFi* baru terlalu jauh dari setiap *data sample* yang ada. 3. Bernilai "Anda berada di [nama daerah]" ketika sistem berhasil melakukan klasifikasi dan memprediksi koordinat posisi awal.
data	1. Bernilai *NULL* jika sistem gagal mendapatkan koordinat posisi awal. 2. Bernilai koordinat tertentu dalam bentuk *string* dengan format "[koordinat x];;[koordinat y]" saat sistem berhasil mendapatkan koordinat.

3.5.2.3 Menambahkan Posisi

Fungsi ini merupakan fungsi yang digunakan untuk menambahkan data posisi baru. Data *input* merupakan data berupa

JSONArray yang dikirim oleh aplikasi pada saat pengguna melakukan *sampling*, kemudian *web service* akan menambahkannya ke dalam *database* sistem. *Method* yang digunakan pada fungsi ini adalah HTTP *Post*.

Data *JSONArray* yang dikirim ke *web service* sebagai parameter *request* akan dijelaskan pada Tabel 3.13. Parameter *response* yang dikembalikan oleh *web service* setelah data dikirim berupa *JSONArray* dan akan dijelaskan pada Tabel 3.14.

Tabel 3.13 Parameter *Request* Menambahkan Posisi

Nama *Key*	Keterangan *Value*
x_coord	Koordinat x posisi pengguna saat itu.
y_coord	Koordinat y posisi pengguna saat itu.
z_coord	Tingkat atau lantai dimana pengguna berada.
id_room	Kode unik untuk daerah yang dipilih oleh pengguna.

Tabel 3.14 Parameter *Response* Menambahkan Posisi

Nama *Key*	Keterangan *Value*
code	1. Bernilai "444" jika data yang dikirim gagal ditambahkan pada *database* sistem. 2. Bernilai "777" jika data yang dikirim berhasil ditambahkan ke dalam *database* sistem dan berhasil mengembalikan *id_position* dari data posisi yang baru ditambahkan.
message	1. Bernilai "Gagal menambahkan *Position*" jika data yang dikirim gagal ditambahkan. 2. Bernilai "id_position kosong" jika sistem berhasil menambahkan data posisi namun gagal mendapatkan *id_position* dari data baru. 3. Bernilai "Berhasil menambahkan *Position*" jika data yang dikirim berhasil ditambahkan ke dalam *database* dan sistem berhasil mendapatkan *id_position* dari data yang berhasil ditambahkan tersebut.

Nama *Key*	Keterangan *Value*
data	1. Bernilai *id_position* dari data yang baru saja ditambahkan jika sistem berhasil menambahkan data dan mendapatkan *id_position* dari data baru tersebut. 2. Bernilai *NULL* saat sistem gagal menambahkan data posisi ataupun gagal mendapatkan *id_position* dari data yang baru saja ditambahkan.

3.5.2.4 Menambahkan RSS

Fungsi ini merupakan fungsi yang digunakan *web service* untuk menambahkan data RSS *WiFi* baru. *Input* dari fungsi ini merupakan data berupa *JSONArray* yang dikirim oleh aplikasi saat pengguna melakukan *sampling*, kemudian *web service* akan menambahkan data tersebut ke dalam *database* sistem. *Method* yang digunakan pada fungsi ini adalah HTTP *Post*.

Data *JSONArray* yang dikirim ke *web service* sebagai parameter *request* akan dijelaskan pada Tabel 3.15. Parameter *response* yang dikembalikan oleh *web service* setelah aplikasi mengirimkan data berupa *JSONArray* dan akan dijelaskan pada Tabel 3.16.

Tabel 3.15 Parameter *Request* Menambahkan RSS

Nama *Key*	Keterangan *Value*
bssid	Kode unik dari *access point*
signal_strength	Kekuatan siyal dari *access point*
id_position	Kode unik dari posisi pengguna saat itu
frequency	Frekuensi sinyal dari *access point*
date_received	Waktu saat *sampling* dilakukan

Tabel 3.16 Parameter *Response* Menambahkan RSS

Nama *Key*	Keterangan *Value*
code	1. Bernilai “444” jika data yang dikirim gagal ditambahkan pada *database* sistem.

Nama *Key*	Keterangan *Value*
	2. Bernilai "777" jika data yang dikirim berhasil ditambahkan ke dalam *database* sistem.
message	1. Bernilai "Gagal menambahkan RSS" jika data yang dikirim gagal ditambahkan. 2. Bernilai "RSS berhasil ditambahkan" jika data yang dikirim berhasil ditambahkan ke dalam *database*.
data	Bernilai *NULL*.

3.5.2.5 Mendapatkan Ruangan

Fungsi ini digunakan oleh *web service* untuk mendapatkan data seluruh ruangan dan mengirimnya ke aplikasi saat pengguna membuka halaman Scan *WiFi*. Tidak ada input untuk fungsi ini, dan *method* yang digunakan merupakan HTTP *Get*.

Data yang dikembalikan saat *web service* menjalankan fungsi ini adalah *JSONArray* dengan parameter yang akan dijelaskan pada Tabel 3.17. Sedangkan untuk data dari seluruh ruangan dan daerah yang berhasil didapatkan oleh sistem akan disajikan dalam bentuk *JSONArray* seperti contoh pada Gambar 3.13.

```
{
   "code":"777",
   "message":"Berhasil mendapatkan list room",
   "data":[
      {
         "id_room":"1",
         "name":"Lab.RPL"
      },
      {
         "id_room":"2",
         "name":"Lab.NCC"
      }
   ]
}
```

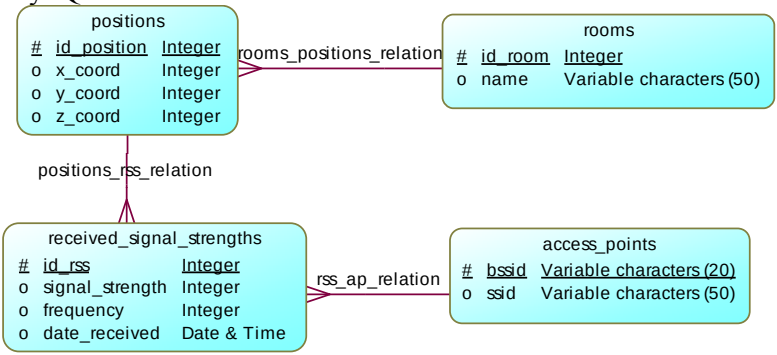
Gambar 3.13 Contoh *Response* JSON dari Data Ruangan

Tabel 3.17 Parameter *Response* Mendapatkan Ruangan

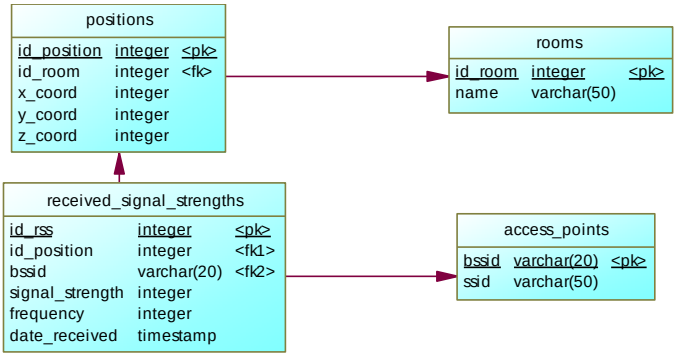
Nama *Key*	Keterangan *Value*
code	1. Bernilai "444" jika sistem gagal mendapatkan data seluruh ruangan atau daerah dari *database* sistem. 2. Bernilai "777" jika sistem berhasil mendapatkan data seluruh ruangan dan daerah dari *database* sistem.
message	1. Bernilai "Daftar *room* kosong" jika sistem gagal mendapatkan data seluruh ruangan. 2. Bernilai "Berhasil mendapatkan *list rom*" jika sistem berhasil mendapatkan data seluruh ruangan dan daerah dari *database* sistem.
data	1. Bernilai *NULL* jika sistem gagal mendapatkan data seluruh ruangan dan daerah. 2. Bernilai data seluruh ruangan dan daerah yang disajikan dalam bentuk *JSONArray*.

3.6 Perancangan *Database* Sistem

Pada subbab ini akan dijelaskan bagaimana rancangan *database* yang digunakan pada sistem aplikasi ini. *Database* pada sistem yang dibangun pada Tugas Akhir ini menggunakan DBMS MySQL.



Gambar 3.14 *Conceptual Data Model* (CDM) *Web Service*



Gambar 3.15 *Physical Data Model* (PDM) *Web Service*

Database digunakan untuk penyimpanan dan pengelolaan seluruh data RSS, *access point*, ruangan dan posisinya. *Conceptual Data Model* (CDM) dan *Physical Data Model* (PDM) dari *database* ini dapat dilihat pada Gambar 3.14 dan Gambar 3.15.

3.6.1 Rancangan Tabel Data Access Points

Tabel *Access Point* digunakan untuk menyimpan data *access point* berupa BSSID (*Basic Service Set Identifier*) dan SSID (*Service Set Identifier*). Detail atribut tabel *Access Point* dijelaskan pada Tabel 3.18.

Tabel 3.18 Atribut Tabel *Access Point*

Nama Kolom	Tipe Data	Keterangan
bssid	*varchar(20)*	*Primary Key* kode unik pada *Access Point*
ssid	*varchar(50)*	Nama jaringan *WiFi* dari *Access Point*

Tabel *Access Point* memiliki relasi ke tabel lainnya yakni sebagai berikut.

1. Tabel Kekuatan Sinyal yang Diterima (*Received Signal Strength*)
   Hubungan dengan tabel *access point* adalah tabel *access point* menyimpan informasi *access point* yang dibutuhkan

pada setiap *record* RSS. RSS juga memiliki data kekuatan sinyal yang diterima dari *access point* tersebut.

3.6.2 Rancangan Tabel Data *Received Signal Strengths*

Tabel kekuatan sinyal yang diterima atau RSS digunakan untuk sampel data *Indoor Localization* yang akan diproses untuk memprediksi lokasi dari pengguna. Detail tabel RSS dijelaskan pada Tabel 3.19.

Tabel RSS memiliki relasi ke tabel lainnya yakni sebagai berikut.

1. Tabel Posisi (*Position*)
   Tabel *position* menyimpan informasi posisi detail lokasi untuk setiap sinyal *WiFi* yang ditangkap pada tabel RSS.
2. Tabel *Access Point*
   Tabel *Access Point* menyimpan informasi detail *access point* untuk setiap sinyal *WiFi* yang ditangkap pada tabel RSS.

Tabel 3.19 Atribut Tabel Kekuatan Sinyal yang Diterima (*Received Signal Strength)*

Nama kolom	Tipe Data	Keterangan
id_rss	*int*	*Primary Key* tabel RSS
bssid	*varchar(20)*	*Foreign Key* posisi RSS terdeteksi
id_position	*varchar(20)*	*Foreign Key Access Point* yang terdeteksi
signal_strength	*int*	Kekuatan sinyal dari *access point* yang terdeteksi

3.6.3 Rancangan Tabel Data Positions

Tabel posisi digunakan untuk menyimpan semua posisi yang terdapat pada lokasi gedung. Setiap lokasi pada suatu ruangan akan dibagi setiap ukuran 2,4 m x 2,4 m. Posisi pada setiap ukuran tersebut akan disimpan di tabel posisi. Detail tabel Posisi dijelaskan pada Tabel 3.20.

Tabel 3.20 Atribut Tabel Posisi (*Position*)

Nama kolom	Tipe Data	Keterangan
id_position	*int*	*Primary Key* tabel *Position*
id_room	*int*	*Foreign Key* ruangan dari posisi
x_coord	*int*	Koordinat X pada posisi
y_coord	*int*	Koordinat Y pada posisi
z_coord	*int*	Koordinat Z pada posisi

Tabel posisi memiliki relasi ke tabel lainnya yakni sebagai berikut.

1. Tabel Ruangan (*Rooms)*
   Hubungan dengan tabel posisi adalah tabel ruangan akan menyediakan informasi lengkap tentang ruangan pada setiap posisi
2. Tabel Kekuatan Sinyal yang Diterima (*Received Signal Strength)*
   Tabel RSS menyimpan atribut *id* dari tabel posisi, sehingga informasi detail posisi dari setiap sinyal yang tertangkap pada tabel RSS dapat diketahui.

3.6.4 Rancangan Tabel Data Rooms

Tabel Ruangan (*Room)* digunakan untuk menyimpan data informasi detail ruangan. Detail tabel Ruangan dijelaskan pada Tabel 3.21. Tabel Ruangan memiliki relasi ke tabel lainnya yaitu sebagai berikut.

1. Tabel Ruangan (*Rooms)*
   Tabel Ruangan menyimpan informasi ruangan lengkap untuk setiap posisi dalam tabel Posisi, sehingga setiap ruangan memiliki beberapa posisi atau bagian.

Tabel 3.21 Tabel Ruangan (*Room)*

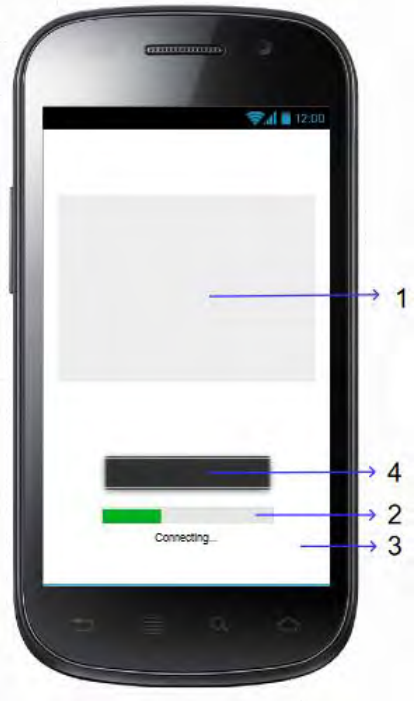
Nama kolom	Tipe Data	Keterangan
id_room	*int*	*Primary Key* tabel *Room*
name	*varchar(50)*	Nama lengkap dari ruangan

3.7 Perancangan Antarmuka Aplikasi

Pada subbab ini akan dibahas secara detail dari rancangan antarmuka aplikasi *Real Time Tracking Indoor Positioning.*

3.7.1 Antarmuka Halaman Splash Screen

Halaman ini merupakan halaman awal saat pengguna membuka aplikasi. Halaman ini berguna untuk melakukan *checking* apakah perangkat telah tersambung internet dan menerima sinyal *WiFi* yang sesuai dengan kriteria *data sample*. Jika telah sesuai, maka halaman ini akan meneruskan ke halaman utama aplikasi. Jika tidak sesuai, maka halaman ini akan menampilkan peringatan. Untuk perancangan antarmuka halaman Splash Screen dapat dilihat pada Gambar 3.16.



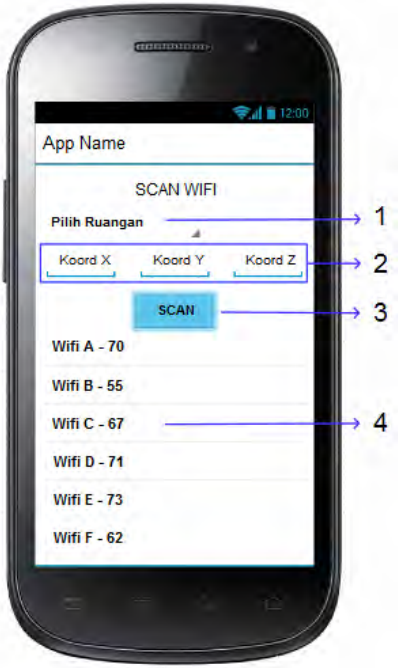
Gambar 3.16 Antarmuka Halaman Splash Screen

Berikut ini adalah penjelasan masing-masing nomor yang tertera pada Gambar 3.16.

1. Berupa gambar logo aplikasi.
2. Berupa *progress bar* yang menunjukkan aplikasi sedang berjalan dan melakukan *checking.*
3. Berupa *textview* yang menampilkan status proses aplikasi.
4. Berupa *toast* yang menampilkan pesan apakah aplikasi dapat berjalan atau tidak.

3.7.2 Antarmuka Halaman Scan WiFi

Gambar rancangan antarmuka halaman Scan WiFi ditunjukkan pada Gambar 3.17. Pada halaman tersebut, pengguna dapat mengumpulkan daftar RSS di suatu *Reference Point* dan menyimpannya ke dalam *database*.



Gambar 3.17 Rancangan Antarmuka Halaman Scan WiFi

Keterangan nomor pada Gambar 3.17 adalah sebagai berikut.

1. Berupa combo box yang digunakan untuk memilih nama ruangan atau daerah.
2. Berupa *textbox* untuk mengisi koordinat yang menjadi *Reference Point.*
3. Berupa tombol yang digunakan untuk melakukan penyimpanan daftar RSS ke dalam *database* sistem.
4. Berupa *list* yang menampilkan daftar informasi seluruh RSS WiFi yang berhasil ditangkap oleh perangkat.

3.7.3 Antarmuka Halaman Peta

Gambar rancangan antarmuka yang menampilkan posisi pengguna saat itu juga dalam bentuk peta gedung Teknik Informatika ditunjukkan pada Gambar 3.18.

Gambar 3.18 Rancangan Antarmuka Halaman Peta

Berikut ini adalah penjelasan masing-masing nomor yang tertera pada Gambar 3.18.

1. Peta gedung Teknik Informatika dengan *marker* sebagai posisi pengguna saat itu juga. *Marker* tersebut bisa bergerak dengan jarak dan arah tertentu sesuai pergerakan pengguna.

2. Berupa tombol *refresh*, digunakan untuk menunjukkan kembali posisi pengguna jika sistem mengalami masalah saat tengah digunakan.

BAB IV
IMPLEMENTASI

Pada bab ini akan dibahas mengenai implementasi yang dilakukan berdasarkan rancangan yang telah dijabarkan pada bab sebelumnya. Implementasi yang dijelaskan meliputi lingkungan pembangunan perangkat lunak *client* dan server, implementasi antarmuka pengguna, dan implementasi proses-proses yang terjadi pada masing-masing kasus penggunaan pada perangkat lunak. Implementasi sistem mengacu pada perancangan yang ditulis pada Bab 3. Namun, tidak menutup kemungkinan adanya perubahan-perubahan dari rancangan tersebut apabila memang diperlukan.

4.1 Lingkungan Implemetasi

Lingkungan implementasi yang akan digunakan untuk melakukan implementasi adalah komputer dan perangkat *smartphone.* Untuk komputer yang digunakan memiliki spesifikasi sebagai berikut.

1. Perangkat keras
   - Prosesor: Intel® Core™ i3-3240 CPU @ 3.40GHz.
   - Memori: 8.00 GB.
2. Perangkat lunak
   - Windows 10 64 bit sebagai sistem operasi.
   - Android Studio 1.5.1 sebagai IDE untuk implementasi aplikasi *smartphone.*
   - SQLyog Ultimate v8.6 sebagai aplikasi manajemen *database* sistem.
   - Sublime Text 3 sebagai *editor* untuk merancang *web service* yang digunakan sistem.
   - Pencil untuk merancang perencanaan antarmuka aplikasi perangkat bergerak pengguna.
   - Power Designer 15.0 untuk merancang *database.*
   - StarUML v2.5.1 untuk mendesain diagram perancangan sistem.

Sedangkan perangkat *smartphone* yang digunakan dalam pengembangan dan pengujian sistem yaitu Asus Zenfone 2 ZE551ML dengan spesifikasi sebagai berikut.

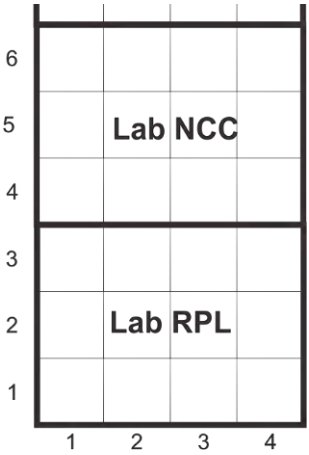
- Android OS, v5.0 (Lollipop) .
- Intel Atom Z3580 CPU Quad-core 2.3 GHz (4GB RAM).
- *WiFi* 802.11 a/b/g/n/ac.
- Memiliki sensor *accelerometer* dan *magnetometer.*

4.2 Implementasi Proses

Pada subbab ini akan dibahas tentang fungsi atau proses yang berjalan pada sistem dan aplikasi *realtime tracking indoor positioning*. Proses yang terjadi mulai dari pengambilan *data sample* hingga memprediksi pergerakan pengguna akan dijelaskan sebagai berikut.

4.2.1 Implementasi Proses Pengumpulan *Data Sample*

Sebelum melakukan prediksi lokasi pengguna, perlu mempersiapkan peta lokasi seluruh ruangan yang ada pada gedung Teknik Informatika ITS, dalam studi kasus ini adalah lantai tiga. Peta tersebut digunakan sebagai panduan ketika melakukan pengumpulan data, proses prediksi lokasi dan uji coba.



Gambar 4.1 Representasi Peta dalam Koordinat Kartesius

Proses pembuatan peta dilakukan dengan embagi setiap ruangan menjadi beberapa bagian dengan ukuran 2,4 m x 2,4 m dan ditentukan dengan koordinat. Posisi pengguna diketahui dari sumbu X dan Y, sedangkan tinggi lantai pengguna diketahui dari sumbu Z. Contoh peta ruangan digambarkan seperti pada Gambar 4.1. Pemetaan lantai tiga gedung Teknik Informatika ITS ke dalam koordinat kartesius dapat dilihat pada lampiran.

Setelah pembuatan peta seluruh ruangan, pengumpulan data sampel dilakukan untuk setiap posisi pada ruangan tersebut. Di setiap koordinat, akan diambil *data sample* beberapa kali. Pada proses pengambilan *data sample,* aplikasi mengumpulkan seluruh data *WiFi* berupa BSSID, SSID dan level sinyalnya, serta mencatat ruangan dan koordinatnya.

4.2.2 Implementasi Algoritma Klasifikasi EWKNN

Setelah *data sample* terkumpul, data tersebut digunakan saat proses prediksi posisi pengguna. Proses ini merupakan proses mengolah data sinyal *WiFi* yang terdeteksi di sekitar posisi pengguna, kemudian data tersebut akan dihitung menggunakan metode *Enhanced Weighted K-Nearest Neighbor* (EWKNN) terhadap *data sample* yang terkumpul hingga posisi pengguna dapat teridentifikasi.

Saat proses prediksi posisi pengguna, aplikasi akan melakukan *scanning* sinyal *WiFi* untuk mendapatkan BSSID, SSID serta kekuatan sinyal yang diterima (RSS) dari beberapa *access point* yang terdeteksi di sekitar posisi pengguna. Selanjutnya *server* akan menerima daftar RSS tersebut dan memprosesnya hingga dapat memprediksi lokasi pengguna.

Klasifikasi EWKNN dijalankan setelah aplikasi mengirimkan data RSS yang diterima di sektiar pengguna ke *server*, kemudian data tersebut akan dibandingkan dengan *data sample* yang ada di dalam *database*. Seperti yang telah dijelaskan pada bab sebelumnya, *data sample* RSS tersebut dikelompokkan dalam setiap *Reference Point* kemudian diurutkan mulai dari jarak terdekat hingga terjauh berdasarkan selisih nilai antar RSS input dengan RSS *data sample.* Setelah diurutkan, dilakukan eliminasi

Reference Point berdasarkan *threshold* yang ditentukan. *Threshold* atau nilai jarak ambang batas yang digunakan adalah rata-rata dari jumlah seluruh jarak. *Reference Point* yang diambil adalah yang memiliki jarak dibawah *threshold*.

```
1  INITIALIZE array_d, array_s
2  //input array list RSS yang berhasil ditangkap
3  INPUT rss_new
4  //mengambil seluruh data posisi, kemudian
   menghitung selisih setiap RSS sampel
5  GET position FROM database
6  FOREACH position DO
7     GET rss_position FROM database
8     array_d[position] = abs (rss_position –
   rss_new)
9  ENDFOREACH
10 SORT_ASCENDING array_d
11 threshold = sum_of_d / count_of_d
12 //filter data selisih menggunakan threshold
13 FOREACH array_d as i DO
14    IF array_d[i] > threshold THEN
15       remove array_d[i]
16    ELSE THEN
17       array_s[i] = array_d[i] – array_d[1]
18 ENDFOREACH
19 //menghitung rata-rata selisih RSS, dan
   menentukan titik yang menjadi “kandidat”
20 e = sum_of_s / count_of_s
21 FOREACH array_d as i THEN
22    IF array_d[i] < e THEN
23       position(x,y) = GET position of
   array_d[i] FROM DATABASE
24 ENDFOREACH
25 //menghitung perkiraan posisi pengguna
26 estimated_position = mean_of_position(x,y)
27 RETURN estimated_position
```

Gambar 4.2 *Pseudocode* Algoritma EWKNN

Dari *Reference Point* yang tersisa, ditentukan suatu nilai E(S) yang merupakan rata-rata dari jumlah setiap selisih jarak *Reference Point* dengan jarak terkecil seperti yang telah dijelaskan di bab sebelumnya. Kemudian dilakukan pemilihan *Reference Point* yang memiliki selisih jarak dibawah E(S), yang digunakan untuk menghitung prediksi posisi pengguna. *Pseudocode* untuk algoritma klasifikasi EWKNN dapat dilihat pada Gambar 4.2.

4.2.3 Implementasi Algoritma Pendeteksi Pergerakan Pengguna

Saat aplikasi selesai melakukan klasifikasi dan menentukan titik awal posisi pengguna, aplikasi akan mendeteksi arah dan pergerakan pengguna menggunakan sensor *magnetometer* dan *accelerometer*.

Magnetometer mendeteksi arah hadap pengguna dengan mengecek perubahan nilai pada sumbu x. Sesuai orientasi sumbu yang telah dijelaskan sebelumnya, nilai yang diambil untuk pengecekan arah hadap pengguna hanya dari sumbu x. Nilai 0 pada sumbu x berarti pengguna atau *smartphone* sedang menghadap ke arah utara. Sistem akan melakukan pengecekan arah secara terus-menerus selama aplikasi berjalan. Jika terdapat perubahan arah, maka sistem akan melakukan *update* arah pengguna yang ditunjukkan oleh rotasi *marker* posisi pengguna. *Pseudocode* untuk algoritma penentuan arah dapat dilihat pada Gambar 4.3.

```
1  INCLUDE magnetometer_module
2  //inisialisasi derajat awal menghadap utara
   dengan nilai 0
3  INITIALIZE current_degree = 0
4  //jika terjadi perubahan derajat, maka marker
   akan diputar sesuai derajat yang baru
5  IF current_degree != x_axis_magnetometer THEN
6     current_degree = x_axis_magnetometer
7     ROTATE marker(current_degree)
```

Gambar 4.3 *Pseudocode* Algoritma Penentuan Arah Hadap Pengguna

Accelerometer digunakan untuk mengecek apakah pengguna sedang bergerak maju atau diam di tempat. Karena nilai yang dihasilkan dari *accelerometer* merupakan percepatan yang masih dipengaruhi oleh gravitasi bumi, maka yang perlu dilakukan adalah menghilangkan gaya gravitasi dari percepatan tersebut, salah satunya menggunakan *low-pass filter*. Dalam pengaplikasian *low-pass filter*, terdapat konstanta *alpha* yang digunakan sebagai *filter* nilai gravitasi. Nilai *alpha* tersebut ditentukan menggunakan Persamaan 4.1.

$$\alpha = \frac{t}{t+dt} \tag{4.1}$$

dimana *dt* merupakan kecepatan pengambilan data per detik, dalam Tugas Akhir ini menggunakan pengambilan data lima kali setiap detik. Sedangkan *t* adalah suatu konstanta waktu.

```
1   INCLUDE accelereometer_module
2   INITIALIZE length, current_degree, threshold
3   //melakukan low-pass filter
4   y_axis = y_axis_accelerometer – gravity
5   //jika y_axis melebihi threshold, dianggap
    bergerak maju
6   IF y_axis > threshold THEN
7      //menggeser marker sejauh length dengan
    sudut current_degree
8      dx = length * cos(current_degree)
9      dy = length * sin(current_degree)
10     SET marker(x+dx, y+dy)
```

Gambar 4.4 *Pseudocode* Algoritma Pendeteksi Pergerakan Pengguna

Sesuai orientasi sumbu *accelerometer* yang telah dijelaskan sebelumnya, nilai yang diambil untuk mengecek apakah pengguna bergerak maju atau tidak hanyalah sumbu Y. Jika nilai dari sumbu Y melebihi nilai batas ambang atau *threshold* yang ditentukan, maka sistem akan mendeteksi bahwa pengguna sedang bergerak maju. Pergerakan maju pengguna direpresentasikan oleh sistem

dengan menggeser *marker* pada peta ke arah yang telah ditunjukkan *magnetometer*. *Pseudocode* untuk algoritma pendeteksi pergerakan pengguna dapat dilihat pada Gambar 4.4.

4.3 Implementasi Kasus Penggunaan

Dalam subbab ini akan diimplementasikan fungsi-fungsi utama dari aplikasi sistem *realtime tracking indoor positioning*. Implementasi yang ada berbentuk *pseudocode*.

4.3.1 Implementasi Pengumpulan Data RSS WiFi

Menambah data *WiFi* berfungsi untuk menambahkan data RSS *WiFi* sebagai *data sample* untuk klasifikasi. Data-data yang dibutuhkan antara lain nama ruangan atau daerah, koordinat *reference point,* dan daftar RSS *WiFi* beserta level kekuatannya. Nama ruangan atau daerah dapat dipilih melalui *dropdown* yang tersedia. Sedangkan untuk koordinat *reference point* dapat diisi di *textbox* yang telah disediakan. Data koordinat tersebut terdiri dari tiga sumbu yaitu sumbu x, y, dan z. Dimana sumbu x dan y merupakan koordinat posisi di suatu lantai, dan sumbu z merupakan level ketinggian gedung, dalam studi kasus ini yaitu lantai tiga. Pengumpulan data RSS *WiFi* dilakukan dengan mengisi tingkat ketinggian gedung secara manual, karena sistem tidak bisa membedakan *WiFi* antar lantai pada gedung. Daftar RSS *WiFi* yang berhasil ditangkap akan otomatis ditunjukkan oleh sistem dalam bentuk *list view.*

```
1  INCLUDE internet
2  INCLUDE wifi_module
3  INPUT idRuangan
4  INPUT x_coord, y_coord, z_coord
5  IF tombolScan CLICKED THEN
6     IF list_of_RSS != NULL THEN
7        INSERT list_of_RSS into database
```

Gambar 4.5 *Pseudocode* Menambahkan data *WiFi*

Tahap awal pengumpulan data RSS adalah pengguna mengisi nama daerah dan koordinat posisi, kemudian pengguna

bisa menekan tombol “Scan” maka sistem akan menyimpan data RSS *WiFi* ke dalam *database* sistem. Implementasi berupa *pesudocode* untuk menambah data RSS *WiFi* dapat dilihat pada Gambar 4.5. Sebagai contoh *data sample* yang berhasil disimpan dalam *database* sistem akan tampak seperti pada Gambar 4.6.

IdRSS	SignalStrength	BSSID	IdPosition
24	-62	d8:24:bd:79:77:3c	2
25	-56	d8:24:bd:79:75:53	2
26	-61	52:f0:2f:57:4f:20	2
27	-68	fc:dd:55:27:09:60	2
28	-72	64:a6:51:f3:20:25	2

Gambar 4.6 Data RSS dalam *Database*

4.3.2 Implementasi Menampilkan Posisi Awal Pengguna

Posisi awal pengguna didapatkan melalui proses klasifikasi menggunakan metode EWKNN. Pada saat aplikasi pertama kali berjalan, sistem akan menangkap RSS *WiFi* yang berada di sekitar posisi pengguna. RSS *WiFi* tersebut akan dikirimkan ke *server* untuk dilakukan klasifikasi. Seluruh RSS *WiFi* yang berhasil ditangkap akan dibandingkan dengan *data sample* yang telah tersimpan di dalam *database* sistem. Proses klasifikasi dilakukan mulai dari perhitungan selisih level kekuatan set RSS baru dengan setiap set RSS *data sample* hingga mencari rata-rata koordinat yang menjadi “kandidat” posisi pengguna.

```
1  INCLUDE internet
2  INCLUDE wifi_module
3  INPUT list_od_RSS
4  IF list_of_RSS != null THEN
5    koordinat = classify()
6    show maps
7    set_marker_position(koordinat.x, koordinat.y)
8  ELSE THEN
9    DISPLAY peringatan
```

Gambar 4.7 *Pseudocode* Mendapatkan Posisi Awal Pengguna

Setelah didapatkan hasil koordinat, maka aplikasi akan menampilkan peta gedung Teknik Informatika dan meletakkan *marker* di posisi yang sesuai dengan koordinat hasil klasifikasi. Implementasi berupa *pseducode* untuk menampilkan posisi awal pengguna dapat dilihat pada Gambar 4.7.

4.4 Implementasi Antarmuka

Pada subbab ini akan dibahas implementasi antarmuka aplikasi berdasarkan rancangan antarmuka yang telah dibahas pada bab 3. Antarmuka yang akan dibahas terdiri dari tiga bagian yang dijelaskan sebagai berikut.

4.4.1 Antarmuka Halaman Splash Screen

Halaman *splash screen* merupakan halaman depan saat pengguna pertama kali membuka aplikasi. Halaman ini terdiri dari gambar logo aplikasi, *progress bar*, serta keterangan proses yang sedang dilakukan aplikasi.

Gambar 4.8 Implementasi Antarmuka *Splash Screen*

Kegunaan halaman *splash screen* adalah untuk mengecek beberapa kondisi mulai dari ketersediaan koneksi internet perangkat, ketersediaan WiFi yang dapat digunakan, serta kesesuaian WiFi yang ditangkap dengan WiFi yang terdapat dalam *database* sistem. Jika telah memenuhi seluruh kondisi yang ditentukan, maka aplikasi akan melakukan klasifikasi untuk menentukan posisi awal pengguna, kemudian menampilkan hasilnya dalam bentuk peta dengan *marker* sebagai posisi awal pengguna. Sebaliknya, jika terdapat kondisi tidak sesuai dengan ketentuan, maka aplikasi akan memunculkan pesan *error* bahwa aplikasi tidak dapat berjalan. Gambar 4.8 menunjukkan implementasi halaman *splash screen.*

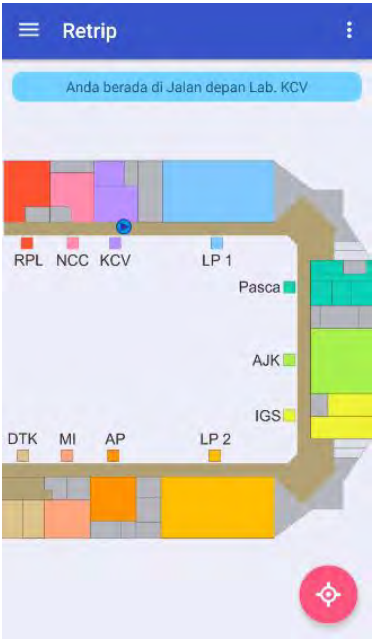
4.4.2 Antarmuka Halaman Scan WiFi

Halaman ini digunakan untuk pengambilan *data sample* RSS *WiFi* di setiap titik. Setelah pengguna memilih nama ruangan dan mengisi koordinat, maka aplikasi akan menyimpan daftar RSS *WiFi* yang diterima ke dalam *database* sistem setiap kali pengguna menekan tombol "Scan". Untuk tampilan implementasi antarmuka halaman Scan WiFi dapat dilihat pada Gambar 4.9.

Gambar 4.9 Implementasi Antarmuka Scan WiFi

4.4.3 Antarmuka Halaman Peta

Halaman ini merupakan halaman utama aplikasi, yang menampilkan peta dan posisi pegguna saat itu juga. Saat pengguna bergerak, maka *marker* yang menandakan posisi pengguna juga akan bergerak sesuai arah pengguna. Terdapat tombol "Refresh" jika pengguna ingin menampilkan kembali posisinya saat itu. Hal ini untuk mengantisipasi saat terjadi kesalahan pada sistem saat melakukan *tracking*. Untuk implementasi antarmuka halaman peta dapat dilihat pada Gambar 4.10.



Gambar 4.10 Implementasi Antarmuka Halaman Peta

4.5 Implementasi Basis Data

Pada subbab ini akan dibahas implementasi dari rancangan *database* sistem yang telah dibahas pada bab 3. Terdapat dua implementasi pada subbab ini yaitu implementasi struktur *database* dan implementasi *query* yang akan dijelaskan sebagai berikut.

4.5.1 Implementasi Struktur Database

Implementasi struktur *database* merupakan implementasi sintaks yang digunakan untuk membangun tabel-tabel yang dibutuhkan untuk menyimpan data oleh aplikasi. Implementasi *database* menggunakan sintaks SQL dan terdiri atas beberapa tabel sebagai berikut.

4.5.1.1 Implementasi Tabel Access Point

Tabel *access point* merupakan tabel yang idgunakan untuk menyimpan data informasi berupa BSSID dan SSID dari *access point* tersebut. Implementasi tabel *access point* ditunjukkan pada Gambar 4.11.

```
1  CREATE TABLE access_point (
2    bssid varchar(20) NOT NULL,
3    ssid varchar(50) DEFAULT NULL,
4    PRIMARY KEY (bssid)
5  )
```

Gambar 4.11 Implementasi Tabel *Access Point*

4.5.1.2 Implementasi Tabel Posisi (Position)

Tabel Posisi merupakan tabel yang digunakan untuk menyimpan data seluruh posisi pada gedung Teknik Informatika ITS. Informasi posisi yang disimpan adalah koordinat kartesius berupa sumbu x, y, dan z serta kode ruangan dari posisi tersebut. Implementasi tabel Posisi ditunjukkan pada Gambar 4.12.

```
1  CREATE TABLE position (
2    id_position int(11) NOT NULL AUTO_INCREMENT,
3    x_coord int(11) DEFAULT NULL,
4    y_coord int(11) DEFAULT NULL,
5    z_coord int(11) DEFAULT NULL,
6    id_room int(11) DEFAULT NULL,
7    PRIMARY KEY (id_position),
8    CONSTRAINT FK_position FOREIGN KEY (id_room)
   REFERENCES room (id_room) ON DELETE CASCADE ON
   UPDATE CASCADE)
```

Gambar 4.12 Implementasi Tabel Posisi

4.5.1.3 Implementasi Tabel Kekuatan Sinyal yang Diterima (Received Signal Strength)

Tabel *Received Signal Strength* (RSS) merupakan tabel yang digunakan untuk menyimpan data semua *WiFi* yang diterima untuk setiap posisi. Data yang disimpan berupa kode BSSID untuk mengetahui *access point* mana yang memancarkan sinyal, kode posisi dari tabel Posisi untuk mengetahui posisi tertangkapnya sinyal dan kekuatan sinyal yang ditangkap. Implementasi tabel RSS ditunjukkan pada Gambar 4.13.

```
1  CREATE TABLE received_signal_strength (
2    id_rss int(11) NOT NULL AUTO_INCREMENT,
3    signal_strength int(11) DEFAULT NULL,
4    bssid varchar(20) DEFAULT NULL,
5    id_position int(11) DEFAULT NULL,
6    PRIMARY KEY (id_rss),
7    CONSTRAINT FK_receivedsignalstrength FOREIGN
   KEY (id_position) REFERENCES position
   (id_position) ON DELETE CASCADE ON UPDATE
   CASCADE,
8    CONSTRAINT FK_receivedsignalstrength2 FOREIGN
   KEY (bssid) REFERENCES access_point (bssid) ON
   DELETE CASCADE ON UPDATE CASCADE
   )
```

Gambar 4.13 Implementasi Tabel *Received Signal Strength*

4.5.1.4 Implementasi Tabel Ruangan (Room)

Tabel Ruangan merupakan tabel yang digunakan untuk menyimpan daftar informasi ruangan yang terdapat pada gedung Teknik Informatika ITS. Informasi ruangan yang disimpan terdiri atas kode ruangan sebagai identitas unik ruangan dan nama ruangan tersebut. Implementasi tabel Ruangan ditunjukkan pada Gambar 4.14.

```
1  CREATE TABLE room (
2    id_room int(11) NOT NULL AUTO_INCREMENT,
3    name varchar(50) DEFAULT NULL,
4    PRIMARY KEY (id_room))
```

Gambar 4.14 Implementasi Tabel *Rooms*

4.5.2 Implementasi *Query*

Query yang digunakan pada aplikasi ini adalah SQL. *Server* untuk mengeksekusi *query* ini berupa *web service* yang dibangun menggunakan *framework* PHP CodeIgniter. Aplikasi akan mengirim data berupa *object* ke *server* dan *output* yang dikembalikan ke aplikasi berupa JSON. Implementasi dari beberapa proses *query* akan dibahas dalam subbab berikut ini.

4.5.2.1 Implementasi *Query* Menambahkan *Data Sample* RSS *WiFi*

Sebelum membuka halaman Scan WiFi, *server* melakukan *query* untuk mendapatkan data seluruh ruangan pada *database*. Implementasi *pseudocode* untuk *query* mendapatkan data ruangan ditampilkan pada Gambar 4.15.

```
1  result = get_all(room)
2  IF result->num_rows > 0 THEN
3     RETURN json_encode(result)
4  ELSE
5     RETURN peringatan
```

Gambar 4.15 Implemetasi *Query* Mengambil Daftar Ruangan

Jika berhasil mendapatkan daftar ruangan, maka hasil *query* akan ditampilkan dalam bentuk *dropdown*. Setelah memilih ruangan dan mengisi koordinat titik, aplikasi akan mengirim data tersebut ke *server* bersama dengan RSS *WiFi* yang tertangkap. *Query* untuk menambahkan *data sample* RSS *WiFi* dieksekusi setelah pengguna menekan tombol "Scan". Sebelum mengeksekusi *query*, *server* perlu memeriksa apakah pengguna telah memilih ruangan dan mengisi koordinat. Dalam menambahkan daftar RSS *WiFi* terdapat tiga sub proses, yaitu:

1. Menambahkan data posisi (*position*), yaitu menambahkan posisi baru sesuai koordinat dan ruangan yang telah pengguna masukkan sebelumnya.
2. Menambahkan data access point, yaitu menambahkan data *WiFi* berupa kode unik BSSID dan nama *WiFi* atau SSID.

Jika BSSID tersebut telah ada dalam *database* sistem, maka proses ini dilewati.

3. Menambahkan data RSS, yaitu menambahkan detail RSS seperti kekuatan sinyal, BSSID, dan posisi dari setiap *WiFi* yang tertangkap oleh aplikasi.

Implementasi *pseudocode* untuk *query* menambahkan data posisi, data *access point* dan data RSS dapat dilihat pada Gambar 4.16, Gambar 4.17 dan Gambar 4.18.

```
1  INPUT x_coord, y_coord, z_coord, id_position
2  result = insert_into_position(x_coord, y_coord,
   z_coord, id_position)
3  IF result IS TRUE THEN
4     RETURN json_encode(get_last_id_position)
5  ELSE
6     RETURN peringatan
```

Gambar 4.16 Implementasi *Query* Menambah Data Posisi

```
1  INPUT bssid, ssid
2  IF bssid IS EXIST THEN
      RETURN sukses
3  ELSE IF bssid NOT EXIST THEN
4     result = insert_into_acess_point(bssid, ssid)
5     IF result IS TRUE THEN
6        RETURN sukses
7     ELSE THEN
8        RETURN peringatan
```

Gambar 4.17 Implementasi *Query* Menambah Data *Access Point*

```
1  INPUT bssid, id_position, signal_strength
2  result = insert_into_rss(bssid, id_position,
   signal_strength)
3  IF result IS TRUE THEN
4     RETURN sukses
5  ELSE
6     RETURN peringatan
```

Gambar 4.18 Implementasi *Query* Menambah Data RSS

4.5.2.2 Implementasi *Query* Penentuan Posisi Awal Pengguna

Sebelum membuka halaman peta, aplikasi terlebih dahulu menangkap RSS *WiFi* yang ada di sekitar pengguna saat itu juga. RSS *WiFi* yang berhasil ditangkap selanjutnya dikirim ke server untuk diklasifikasikan terhadap *data sample* pada *database* sistem. *Query* untuk mendapatkan seluruh *data sample* dapat dilihat pada Gambar 4.19.

```
1  positions = get_all(position)
2  FOREACH positions DO
3     GET rss FROM position
4  ENDFOREACH
```

Gambar 4.19 *Query* Mendapatkan Seluruh *Data Sample*

Setelah melakukan klasifikasi RSS *WiFi* menggunakan *data sample*, hasil yang dikeluarkan oleh sistem merupakan koordinat yang menjadi perkiraan posisi. Koordinat ini kemudian digunakan untuk memindahkan *marker* posisi pengguna pada peta, dan juga untuk mendapatkan informasi ruangan atau daerah yang menjadi keterangan pada aplikasi. *Query* untuk mendapatkan informasi ruangan atau daerah ditunjukkan pada Gambar 4.20.

```
1  //Input berupa koordinat x dan y dari hasil
   klasifikasi
2  INPUT x, y
3  GET room_name FROM room WHERE x_coord = x AND
4  y_coord = y
5  RETURN room_name
```

Gambar 4.20 *Query* Mendapatkan Informasi Ruangan

BAB V
UJI COBA DAN EVALUASI

Pada bab ini akan dibahas mengenai uji coba dari segi fungsionalitas aplikasi. Uji coba fungsionalitas dibagi menjadi beberapa skenario fungsionalitas yang terdapat pada aplikasi.

5.1 Lingkungan Uji Coba

Lingkungan pengujian merupakan perangkat-perangkat yang dilibatkan dalam proses pengujian. Lingkungan pengujian ini menggunakan perangkat keras berupa *smartphone* berbasis Android yang memiliki sensor *accelerometer* dan *magnetometer*, terdapat fasilitas perangkat *WiFi* serta dapat terhubung dengan internet. Adapun spesifikasi *smartphone* yang digunakan dijelaskan pada Tabel 5.1.

Tabel 5.1 Spesifikasi Lingkungan Pengujian

Spesifikasi	Keterangan
Nama *Smartphone*	Asus Zenfone 2 ZE551ML
Sistem Operasi	Android OS, v5.0
Prosesor	Quad-core 2.3 GHz
RAM	4 GB
WLAN	Wi-Fi 802.11 a/b/g/n/ac
Sensor	*Accelerometer, magnetometer*

5.2 Pengujian Fungsionalitas

Subbab ini menjelaskan tentang skenario pengujian fungsionalitas perangkat lunak pada Tugas Akhir ini. Pengujian didokumentasikan secara sistematis sebagai tolok ukur keberhasilan sistem. Validasi pengujian ditentukan dengan mencocokkan informasi yang disimpan dalam *database* sistem dengan informasi yang ditampilkan pada aplikasi.

5.2.1 Pengujian Penambahan *Data Sample WiFi*

Skenario pengujian fungsionalitas ini dapat dilihat pada Tabel 5.2. Saat pengguna menekan menu Scan WiFi, aplikasi akan

menampilkan halaman Scan WiFi. Namun sebelum menampilkan halaman tersebut, aplikasi akan meminta data seluruh daerah yang telah terdaftar dalam *database*. Data tersebut kemudian ditampilkan dalam bentuk *dropdown*. Kemudian aplikasi juga menampilkan *form* isian koordinat titik yang harus diisi oleh pengguna serta daftar RSS *WiFi* di titik tersebut yang teridentifikasi oleh *smartphone*. Jika pengguna menekan tombol "Scan" maka aplikasi akan mengirimkan data tersebut ke sistem untuk dimasukkan ke dalam *database*. Jika berhasil, aplikasi akan menampilkan notifikasi jumlah RSS *WiFi* yang berhasil disimpan dalam *database* sistem seperti pada Gambar 5.1. Hasil RSS WiFi yang berhasil disimpan dalam *database* sistem akan tampak seperti pada Gambar 5.2.

Tabel 5.2 Skenario Pengujian Menambahkan *Data Sample* RSS *WiFi*

Informasi	Keterangan
Nomor	UT-01
Nama Uji Coba	Menambahkan *data sample* RSS *WiFi*
Tujuan	Menambahkan *data sample* ke dalam *database* sistem
Kondisi Awal	Halaman Scan WiFi belum ditampilkan
Skenario	Pengguna membuka Halaman Scan WiFi atau memilih menu Scan WiFi
Masukan	1. Nama ruangan 2. Koordinat ruangan 3. Daftar RSS *WiFi*
Keluaran yang diharapkan	1. Aplikasi menampilkan notifikasi jumlah RSS *WiFi* yang berhasil disimpan 2. RSS *WiFi* dapat ditambahkan ke dalam *database* sistem
Hasil pengujian	Berhasil

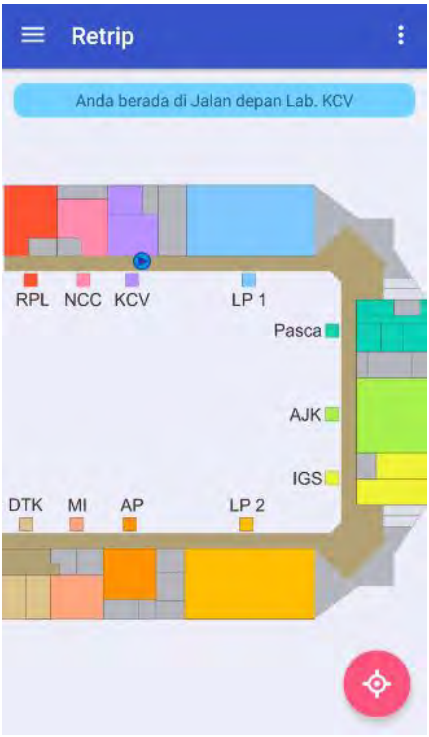
Gambar 5.1 Pengujian Menambahkan *Data Sample* RSS *WiFi*

	id_rss	signal_strength	bssid	id_position
Edit Copy Delete	41308	-60	90:f6:52:8b:1d:a8	2251
Edit Copy Delete	41309	-70	56:fd:52:86:f0:2c	2251
Edit Copy Delete	41310	-70	f0:79:59:24:f2:ef	2251
Edit Copy Delete	41311	-73	bc:20:10:4a:da:22	2251
Edit Copy Delete	41312	-80	0a:18:d6:ad:03:91	2251
Edit Copy Delete	41313	-41	f8:1a:67:37:3d:70	2251
Edit Copy Delete	41314	-45	64:70:02:3f:4e:59	2251
Edit Copy Delete	41315	-48	0a:18:d6:ad:03:8b	2251
Edit Copy Delete	41316	-68	52:f0:2f:57:4f:20	2251
Edit Copy Delete	41317	-83	00:12:5f:0b:5a:7e	2251

Gambar 5.2 Hasil Penyimpanan *Data Sample* RSS *WiFi*

5.2.2 Pengujian Penentuan Posisi Pengguna

Pengujian fungsionalitas untuk penentuan posisi pengguna dimulai saat pengguna membuka aplikasi. Jika RSS *WiFi* yang diterima oleh perangkat sesuai dengan *data sample WiFi* di *database* sistem, maka aplikasi akan menampilkan peta gedung dan posisi pengguna dalam bentuk *marker,* serta tampilan keterangan nama daerah sebagai posisi pengguna saat itu.



Gambar 5.3 Tampilan Posisi dan Keterangan Posisi Pengguna

Tampilan peta dan keterangan daerah posisi pengguna ditunjukkan seperti pada Gambar 5.3. Skenario pengujian fungsionalitas ini akan dijelaskan pada Tabel 5.3.

Tabel 5.3 Skenario Pengujian Penentuan Posisi Pengguna

Nomor	UT-02
Nama Uji Coba	**Menampilkan posisi pengguna**
Tujuan	Mengetahui keberadaan pengguna saat itu
Kondisi Awal	Halaman peta belum ditampilkan
Skenario	Pengguna membuka halaman peta atau membuka aplikasi pertama kali
Masukan	Daftar RSS *WiFi*
Keluaran yang diharapkan	1. Aplikasi menampilkan peta dan *marker* sebagai posisi pengguna 2. Aplikasi menampilkan keterangan daerah posisi pengguna
Hasil pengujian	Berhasil

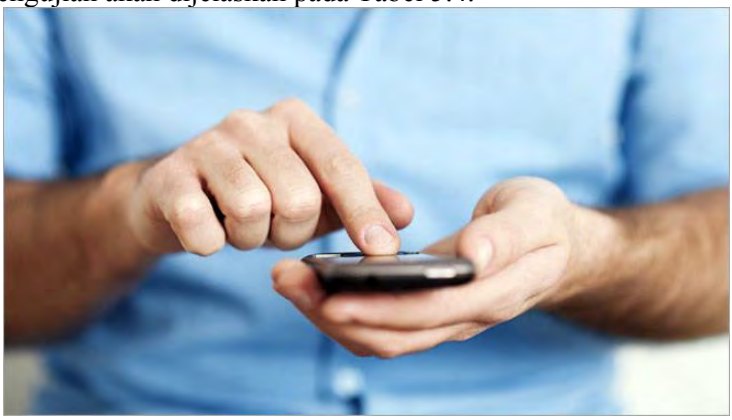
5.3 Pengujian Akurasi

Pengujian akurasi dilakukan dengan membandingkan hasil *output* dari uji coba sistem dengan keadaan yang sebenarnya. Pengujian akurasi ini terdiri dari tiga macam, yaitu akurasi koordinat posisi pengguna, jumlah langkah pengguna, dan arah hadap pengguna. Ketiga pengujian ini dilakukan pada setiap uji coba kemudian diambil rata-rata akurasinya. Untuk skenario dan hasil dari masing-masing pengujian akan dijelaskan lebih detail pada subbab selanjutnya.

5.3.1 Skenario Pengujian Akurasi

Pada bagian ini akan dijelaskan mengenai skenario pengujian keakuratan sistem beserta hasilnya. Uji coba pada aplikasi Tugas Akhir ini dilakukan dengan menentukan empat rute yang masing-masing memiliki koordinat awal dan akhir. Pada setiap rute, dilakukan lima kali percobaan dan pengguna berjalan melalui rute tersebut menggunakan aplikasi yang telah dibuat. Posisi *smartphone* harus dalam kondisi *steady hand* saat digunakan yaitu berada pada posisi datar dan stabil seperti

ditunjukkan pada Gambar 5.4. Untuk keempat rute yang menjadi pengujian akan dijelaskan pada Tabel 5.4.



Gambar 5.4 Posisi *Steady Hand* pada *Smartphone*

Tabel 5.4 Rute Skenario Pengujian Akurasi

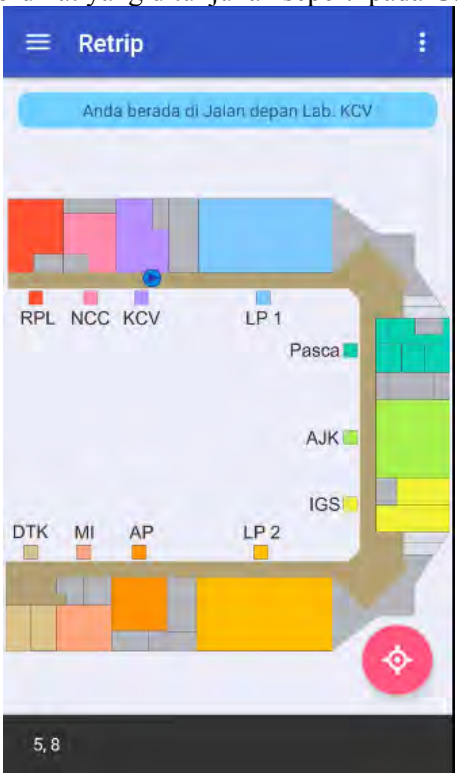
No	Koordinat		Keterangan
	Awal	Akhir	
1.	5, 9	5, 18	Jalan depan Lab. KCV → Plasa Lama 3
2.	5, 15	12, 20	Plasa Lama 3 → Jalan depan Lab. AJK
3.	24, 14	22, 19	Lab. Pemrograman 2 → Plasa Baru 3
4.	14, 23	17, 20	Lab. AJK → Jalan depan Lab. IGS

5.3.2 Hasil Pengujian Akurasi

Pada bagian ini akan dijelaskan beberapa macam pengujian yang akan dilakukan terhadap skenario yang telah ditentukan. Pengujian tersebut meliputi pengujian akurasi posisi awal pengguna, deteksi langkah pengguna, dan estimasi arah hadap pengguna. Untuk lebih jelasnya setiap pengujian akan dijelaskan subbab berikut ini.

5.3.2.1 Pengujian Akurasi Koordinat Posisi Pengguna

Nilai akurasi pendeteksian lokasi pada suatu titik atau *Reference Point* dihitung dengan mencari rata-rata dari seluruh percobaan di setiap rute. Setiap percobaan akan menghasilkan koordinat-koordinat yang ditunjukan seperti pada Gambar 5.5.



Gambar 5.5 Tampilan Koordinat Hasil Uji Coba

Karena sistem *Indoor Positioning* ini menghitung dengan membandingkan langsung RSS yang diterima dengan RSS yang ada pada *database* sistem, maka jika *test area* menghasilkan akurasi yang kurang dapat ditingkatkan dengan menambah *data sample* baru pada daerah sekitar *test area.* Namun perlu diperhatikan bahwa semakin banyak *data sample*, proses

klasifikasi akan berlangsung lebih lama. Hasil pengujian akurasi koordinat di setiap rute akan dijelaskan pada Tabel 5.5.

Secara keseluruhan, sistem yang dibangun pada Tugas Akhir ini menghasilkan rata-rata akurasi sebesar 2,5 meter pada keempat rute yang diuji coba seperti ditunjukkan pada Tabel 5.6. Pada Tabel 5.6 akurasi atau *error rate* dihitung menggunakan rumus Euclidean Distance yang dijelaskan pada Persamaan 5.1. Sedangkan akurasi jarak dihitung dengan mengalikan nilai *error rate* dengan lebar koordinat, yaitu 2,4 meter.

$$\boldsymbol{e = \sqrt{(x_1 - x_0)^2 + (y_1 - y_0)^2}} \qquad (5.1)$$

Keterangan:
x_1 dan y_1 : koordinat hasil klasifikasi.
x_0 dan y_0 : koordinat yang sebenarnya.

Tabel 5.5 Hasil Pengujian Akurasi Koordinat Posisi

No	Rute	Koordinat Awal		Koordinat Akhir	
		x_0, y_0	x_{est}, y_{est}	x_0, y_0	x_{est}, y_{est}
1.	**Rute 1**				
	Percobaan 1	5, 9	5, 9	5, 18	5, 16
	Percobaan 2		5, 7		5, 16
	Percobaan 3		5, 8		5, 17
	Percobaan 4		5, 8		5, 16
	Percobaan 5		5, 8		5, 17
	Rata-rata		5, 8		5, 16.4
2.	**Rute 2**				
	Percobaan 1	5, 15	5, 15	12, 20	11, 20
	Percobaan 2		5, 14		12, 20
	Percobaan 3		5, 14		12, 20
	Percobaan 4		5, 14		14, 20
	Percobaan 5		5, 14		12, 20
	Rata-rata		5, 14.2		12.2, 20

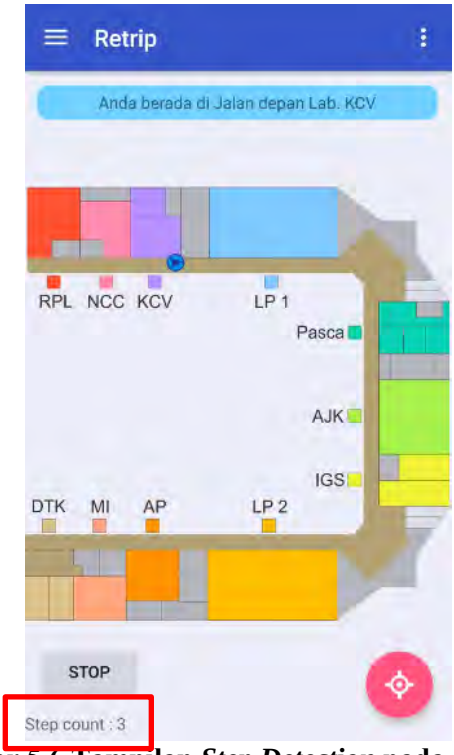
No	Rute	Koordinat Awal		Koordinat Akhir	
		x_0, y_0	x_{est}, y_{est}	x_0, y_0	x_{est}, y_{est}
3.	**Rute 3**				
	Percobaan 1	24, 14	23, 14	22, 19	20, 20
	Percobaan 2		23, 15		20, 20
	Percobaan 3		23, 15		20, 20
	Percobaan 4		24, 13		20, 20
	Percobaan 5		23, 13		20, 21
	Rata-rata		23.2, 14		20, 20.2
4.	**Rute 4**				
	Percobaan 1	14, 23	15, 23	18, 20	16, 20
	Percobaan 2		15, 22		18, 20
	Percobaan 3		15, 23		18, 20
	Percobaan 4		15, 23		18, 20
	Percobaan 5		15, 23		17, 20
	Rata-rata		15, 22.8		17.4, 20

Tabel 5.6 Hasil Akurasi Koordinat Posisi Pengguna

No	Rute	Error rate	Akurasi jarak (m)
1.	Rute 1		
	Koordinat Awal	1	2,4
	Koordinat Akhir	1,6	3,84
2.	Rute 2		
	Koordinat Awal	0,8	1,92
	Korodinat Akhir	0,2	0,48
3.	Rute 3		
	Koordinat Awal	0,8	1,92
	Koordinat Akhir	2,3	5,5
4.	Rute 4		
	Koordinat Awal	1,02	2,44
	Koordinat Akhir	0,6	1,44
Rata – rata			2,5

5.3.2.2 Pengujian Akurasi Langkah Pengguna

Nilai akurasi pendeteksian langkah pengguna dihitung dengan membandingkan jumlah langkah yang terdeteksi pada aplikasi dengan jumlah langkah sebenarnya.



Gambar 5.6 Tampilan *Step Detection* pada Aplikasi

Setiap percobaan menghasilkan jumlah langkah yang kemudian di rata-rata untuk satu rute. Untuk tampilan perhitungan langkah pengguna pada aplikasi ditunjukkan pada Gambar 5.6. Setiap terdeteksi sebuah langkah, jumlah step pada aplikasi akan bertambah. Untuk hasil pengujian akurasi pendeteksi jumlah langkah pengguna ditampilkan pada Tabel 5.7. Berdasarkan hasil pengujian akurasi tersebut, secara keseluruhan rata-rata akurasi

pendeteksian langkah pengguna pada sistem ini sebesar 94,8% seperti ditunjukkan pada Tabel 5.8.

Tabel 5.7 Hasil Pengujian Akurasi Deteksi Jumlah Langkah

No.	Rute	$Step_0$	$Step_{est}$	Akurasi
1.	**Rute 1**			
	Percobaan 1	37	39	94,8%
	Percobaan 2	36	36	100%
	Percobaan 3	34	33	97%
	Percobaan 4	35	34	97,1%
	Percobaan 5	34	34	100%
	Rata-rata			**97,8%**
2.	**Rute 2**			
	Percobaan 1	47	43	91,4%
	Percobaan 2	50	51	98%
	Percobaan 3	48	47	97,9%
	Percobaan 4	48	50	96%
	Percobaan 5	45	44	97,7%
	Rata-rata			**96,2%**
3.	**Rute 3**			
	Percobaan 1	52	55	94,5%
	Percobaan 2	55	60	91,6%
	Percobaan 3	54	58	93,1%
	Percobaan 4	53	58	91,3%
	Percobaan 5	53	57	92,9%
	Rata-rata			**92,7%**
4.	**Rute 4**			
	Percobaan 1	48	52	92,3%
	Percobaan 2	55	61	90,1%
	Percobaan 3	47	52	90,3%
	Percobaan 4	49	55	89,1%
	Percobaan 5	48	48	100%
	Rata-rata			**92,3%**

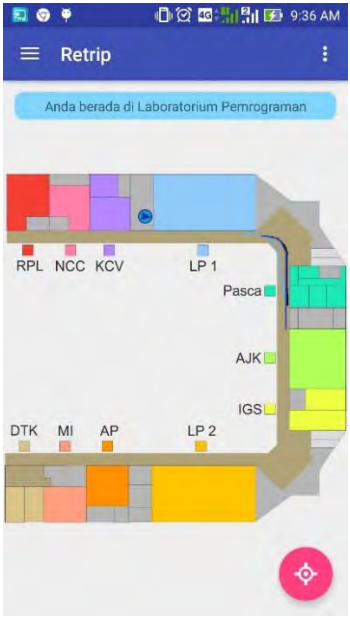
Tabel 5.8 Rata-Rata Pengujian Akurasi *Step Detection*

No	Rute	Akurasi
1.	Rute 1	97,8%
2.	Rute 2	96,2%
3.	Rute 3	92,7%
4.	Rute 4	92,3%
Rata-rata		**94,8%**

5.3.2.3 Pengujian Akurasi Arah Hadap Pengguna

Pengujian arah hadap pengguna dilakukan dengan berjalan mengikuti jalan, sementara aplikasi akan mencatat arah hadap pengguna dalam derajat dalam selang waktu tertentu. Seperti yang telah disampaikan sebelumnya, nilai 0 merupakan arah utara. Setelah uji coba dan didapatkan sekumpulan nilai, nilai tersebut disimpan ke dalam sebuah *file* dan kemudian disajikan dalam bentuk garis seperti pada Gambar 5.7.

```
1  p.lineTo(385+22, 459+22);
2  p.lineTo(394+22, 459+22);
3  p.lineTo(403+22, 459+22);
4  p.lineTo(412+22, 459+22);
5  p.lineTo(421+22, 459+22);
6  p.lineTo(430+22, 459+22);
7  p.lineTo(439+22, 459+22);
8  p.lineTo(448+22, 460+22);
9  p.lineTo(457+22, 460+22);
10 p.lineTo(466+22, 460+22);
11 p.lineTo(475+22, 460+22);
12 p.lineTo(484+22, 460+22);
13 p.lineTo(493+22, 460+22);
14 p.lineTo(502+22, 460+22);
15 p.lineTo(511+22, 460+22);
16 p.lineTo(520+22, 460+22);
17 p.lineTo(529+22, 460+22);
18 p.lineTo(538+22, 460+22);
19 p.lineTo(547+22, 460+22);
20 p.lineTo(556+22, 460+22);
21 p.lineTo(565+22, 460+22);
22 p.lineTo(574+22, 460+22);
23 p.lineTo(583+22, 460+22);
24 p.lineTo(592+22, 460+22);
25 p.lineTo(601+22, 461+22);
```



Gambar 5.7 Hasil Pengujian Arah Hadap Pengguna

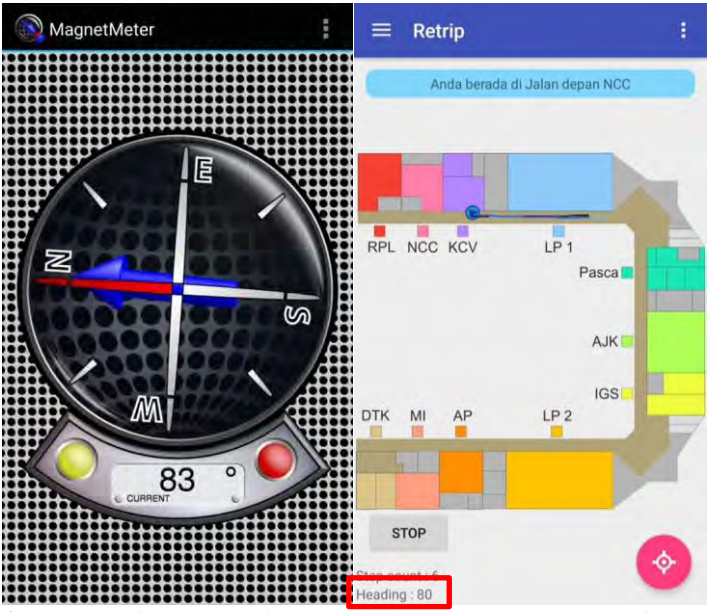
Pengujian akurasi arah hadap pengguna dilakukan dengan membandingkan hasil aplikasi dengan *ground truth*. *Ground truth* yang digunakan dalam pembandingan adalah aplikasi kompas lain yang telah ada sebelumnya, dalam hal ini adalah aplikasi MagnetMeter. Tampilan aplikasi ini tampak seperti pada Gambar 5.8. Uji coba dilakukan dengan menjalankan dua aplikasi pada waktu yang sama di dua *smartphone* yang berbeda namun memiliki tipe yang sama, kemudian pengguna berjalan sesuai dengan skenario yang ditentukan.

Gambar 5.8 Tampilan Aplikasi MagnetMeter

Salah satu tampilan perbandingan arah pada dua aplikasi akan ditampilkan pada Gambar 5.9. Hasil perbandingan nilai kedua aplikasi akan dijelaskan lebih detail pada Tabel 5.10. Pengambilan nilai sebagai perbandingan dilakukan setiap lima detik. Perhitungan akurasi dilakukan dengan menggunakan Persamaan 5.2 dan kategori yang digunakan untuk perbandingan akan dijelaskan pada Tabel 5.9.

$$a = \frac{|d_{est} - d_{gt}|}{90} x\ 100\ \% \qquad (5.2)$$

keterangan:
α : akurasi
d_{est} : nilai arah aplikasi uji coba
d_{gt} : nilai arah aplikasi pembanding



Gambar 5.9 Perbandingan Nilai pada Waktu yang Sama

Tabel 5.9 Kategori Tingkat Akurasi Arah Hadap Pengguna

Perbedaan	Kategori
0^0	Sangat Baik
1^0 - 15^0	Baik
16^0 - 30^0	Cukup Baik
31^0 – 45^0	Buruk
> 45^0	Sangat Buruk

Tabel 5.10 Perbandingan Nilai Arah pada Salah Satu Percobaan

Retrip	MagnetMeter	Selisih	Akurasi
Skenario 1			
76	88	12	86%
82	90	8	91%
86	92	6	93%
91	94	3	96%
86	90	4	95%
Skenario 2			
90	97	7	92%
93	91	2	97%
148	152	4	95%
182	189	7	92%
174	180	6	93%
165	171	6	93%
Skenario 3			
283	271	12	86%
8	15	7	92%
93	87	6	93%
81	87	6	93%
99	101	2	97%
84	83	1	98%
146	152	6	93%
Skenario 4			
0	5	5	94%
4	6	2	98%
271	271	0	100%
228	221	7	92%
173	173	0	100%
182	184	2	98%
168	172	4	95%
185	187	2	98%

Dari seluruh percobaan yang dilakukan, rata-rata akurasi dari estimasi arah hadap pengguna adalah sebesar 94,48%. Perhitungan rata-rata akurasi estimasi arah hadap pengguna ditunjukkan pada Tabel 5.11.

Tabel 5.11 Rata-Rata Estimasi Arah Hadap Pengguna

Nama Rute	Selisih	Akurasi
Rute 1	3,95	95,6%
Rute 2	4,91	94,5%
Rute 3	5,37	94,03%
Rute 4	5,62	93,74%
Rata-rata	**4,96**	**94,48%**

5.4 Evaluasi Pengujian

Berdasarkan hasil pengujian fungsionalitas dan akurasi sistem yang telah disampaikan, pada subbab ini akan dijelaskan mengenai evaluasi dari masing-masing pengujian.

5.4.1 Evaluasi Pengujian Fungsionalitas

Hasil dari pengujian fungsionalitas yang telah dilakukan memberikan hasil yang sesuai dengan skenario yang telah direncanakan. Evaluasi pengujian pada masing-masing fungsionalitas dijelaskan sebagai berikut:

1. Pengujian menambahkan *data sample* RSS *WiFi* telah sesuai dengan *output* yang diharapkan. Kondisi ini ditunjukkan pada pengujian UT-01 yang memberikan informasi bahwa aplikasi berhasil memasukkan sejumlah RSS *WiFi* ke dalam *database* sistem.
2. Pengujian menampilkan posisi pengguna dalam peta gedung telah sesuai dengan *output* yang diharapkan. Kondisi ini ditunjukkan pada pengujian UT-02 yang dapat menampilkan posisi pengguna dalam bentuk *marker* dan juga menampilkan keterangan daerah dimana pengguna sedang berada.

5.4.2 Evaluasi Pengujian Akurasi

Untuk mengevaluasi pengujian akurasi dari sistem ini, akan dibagi ke dalam tiga subbab sesuai dengan tiga jenis pengujian pada subbab sebelumnya.

5.4.2.1 Evaluasi Pengujian Akurasi Koordinat Posisi Pengguna

Berdasarkan hasil pengujian yang ditunjukkan pada Tabel 5.6, akurasi yang dihasilkan oleh sistem ini masih cukup rendah yaitu sekitar 2,5 meter. Dalam hal ini, rendahnya akurasi dikarenakan kekuatan sinyal WiFi yang diterima bersifat fluktuatif. Sinyal WiFi yang tidak konsisten tersebut mengakibatkan perhitungan koordinat posisi menjadi kurang akurat, dikarenakan perbedaan sinyal yang diterima dengan sinyal yang ada pada database sistem cukup berbeda, sebagai contoh seperti pada Tabel 5.12. Contoh tersebut merupakan koordinat yang memiliki error rate paling besar dari seluruh percobaan, sehingga pada Tabel 5.12 akan dianalisa mengapa error rate pada koordinat tersebut cukup besar. Analisa dilakukan dengan membandingkan RSS WiFi pada koordinat sesungguhnya dengan koordinat yang menjadi hasil perhitungan sistem, dengan cara menghitung kedekatan antara RSS WiFi yang diterima oleh aplikasi dengan RSS WiFi pada dua koordinat tersebut. Dengan data sinyal WiFi yang diterima tersebut, perhitungan koordinat posisi pengguna menjadi berbeda dari kenyataan, seperti perhitungan jarak kedekatan yang ditunjukkan pada Tabel 5.13.

Tabel 5.12 Perbandingan Data Sinyal *WiFi* yang Diterima dengan *Data Sample*

BSSID	Rata-rata Kekuatan Sinyal Pada Koordinat (dBm)		
	(x, y)	(20, 20)	(22, 19)
c0:c1:c0:35:72:04	-74	-75,1	-77,2
f8:d1:11:80:b3:7c	-77	-82,2	-80,8
c0:c1:c0:e7:b3:a1	-76	-78,1	-81,9
10:fe:ed:89:1d:8b	-79	-85,5	-84,6

BSSID	Rata-rata Kekuatan Sinyal Pada Koordinat (dBm)		
	(x, y)	(20, 20)	(22, 19)
10:fe:ed:9b:93:ac	-78	-74,1	-79,8
0a:18:d6:0b:8c:d8	-82	-75,3	-83,9
00:26:5a:b3:a7:a0	-71	-66,8	-72,3
10:fe:ed:9b:93:ab	-76	-74,2	-81,7
70:77:81:35:f9:85	-80	-83,4	-85
0a:18:d6:ad:04:52	-79	-76,7	-79,9
e8:94:f6:30:bf:3b	80	-79,9	-86

Tabel 5.13 Perbandingan Jarak Data Uji Coba dengan *Data Sample*

Koordinat	Jarak Kedekatan
(20, 20)	**37,3**
(22, 19)	**41,1**

5.4.2.2 Evaluasi Pengujian Akurasi Pendeteksi Langkah Pengguna

Berdasarkan Tabel 5.8, rata-rata total akurasi pendeteksian langkah pengguna sudah baik. Pada skenario pertama menghasilkan 97,8%, skenario kedua menghasilkan 96,2% dan skenario ketiga menghasilkan 92,7% serta skenario keempat menghasilkan 92,3%. Dari hasil tersebut dapat dilihat bahwa tingkat akurasi secara keseluruhan yang dihasilkan dari skenario pengujian memiliki akurasi yang cukup tinggi yaitu dengan rata-rata 94,8%. Keempat pengujian menunjukkan persentase akurasi lebih dari 80% sehingga dapat disimpulkan pendeteksian langkah pengguna dalam aplikasi ini telah berjalan dengan baik.

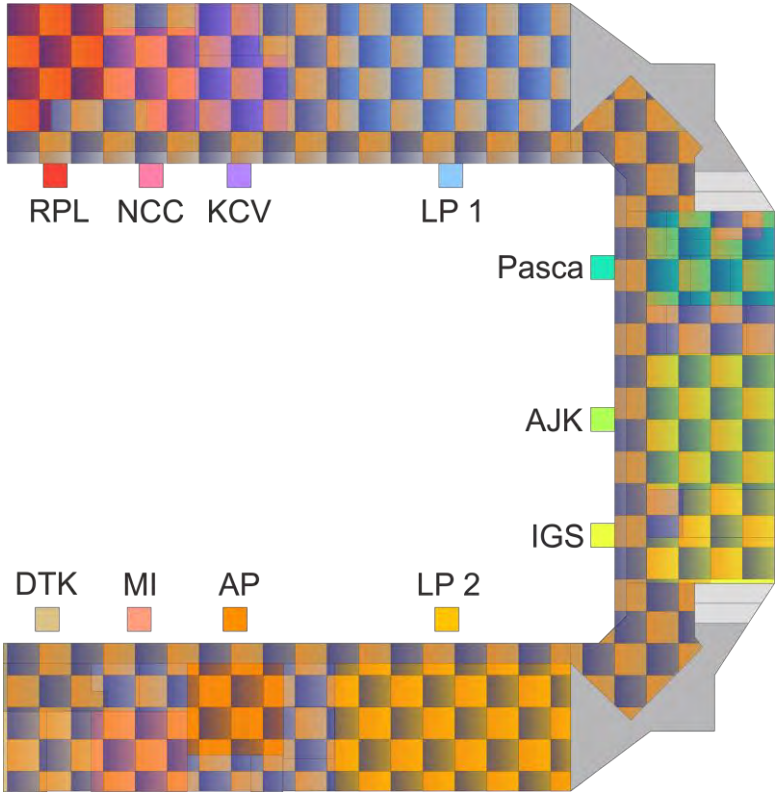
5.4.2.3 Evaluasi Pengujian Akurasi Arah Hadap Pengguna

Berdasarkan hasil pengujian yang ditunjukkan oleh Tabel 5.11, rata-rata total akurasi penentuan arah hadap pengguna sudah

baik. Pada skenario pertama menghasilkan 95,6%, skenario kedua menghasilkan 94,54%, dan skenario ketiga menghasilkan 94,03% serta skenario keempat menghasilkan 93,74%. Dari hasil tersebut dapat dilihat bahwa tingkat akurasi secara keseluruhan memiliki akurasi yang cukup tinggi yaitu 93%. Keempat pengujian menunjukkan persentase akurasi lebih dari 80% sehingga dapat disimpulkan pendeteksian arah hadap pengguna dalam aplikasi ini telah berjalan dengan baik.

[Halaman ini sengaja dikosongkan]

LAMPIRAN



Gambar 8.1 Pemetaan Lantai Tiga Gedung Teknik Informatika ITS

[Halaman ini sengaja dikosongkan]

BAB VI
KESIMPULAN DAN SARAN

Bab ini membahas mengenai kesimpulan yang dapat diambil dari hasil uji coba yang telah dilakukan sebagai jawaban dari rumusan masalah yang dikemukakan. Selain kesimpulan, juga terdapat saran yang ditujukan untuk pengembangan penelitian lebih lanjut.

Sistem *Indoor Positioning* menggunakan perangkat *mobile* seperti *smartphone* masih menjadi masalah yang menantang. Seperti GPS yang tidak bekerja secara akurat di dalam gedung, sistem *Indoor Positioning* juga memiliki kelemahan yaitu sangat bergantung pada infrastruktur gedung yaitu sinyal *WiFi*, sehingga sistem ini terkadang tidak dapat bekerja secara optimal dan *real time* di setiap bagian gedung.

Untuk itulah dikembangkan sistem *Indoor Positioning* yang menggunakan sensor gerak sebagai tambahan untuk mendeteksi langkah dan estimasi arah pengguna saat berjalan. Penggunaan sinyal *WiFi* hanya digunakan sebagai estimasi titik awal pengguna. Hal ini berguna untuk pelacakan keberadaan seseorang secara *real time*, terutama saat persebaran sinyal *WiFi* di suatu gedung tidak tersebar secara merata.

6.1 Kesimpulan

Dalam proses pengerjaan Tugas Akhir dari tahap pendahuluan, kajian pustaka, analisis, perancangan, implementasi dan pengujian sistem *Real Time Tracking Indoor Position* diperoleh kesimpulan sebagai berikut.

1. Proses *sampling* kekuatan sinyal *WiFi* di setiap titik telah berhasil dilakukan dengan membagi wilayah gedung ke dalam *train area* seluas 2,4 m x 2,4 m. Pada setiap titik dilakukan *scanning* sebanyak sepuluh kali untuk merepresentasikan kekuatan sinyal *WiFi* yang bersifat fluktuatif.

2. Proses pendeteksian lokasi yang dilakukan dengan memproses sinyal *WiFi* di sekitar pengguna menjadi estimasi koordinat dengan algoritma EWKNN seperti yang dijelaskan pada bab sebelumnya dapat dijalankan dengan baik.
3. Proses pendeteksian langkah pengguna menggunakan sensor *accelerometer* dan estimasi arah hadap pengguna menggunakan sensor *magnetometer* seperti yang dijelaskan pada bab sebelumnya dapat dijalankan dengan baik.
4. Tugas Akhir ini berhasil menerapkan sistem *Real Time Tracking Indoor Position* dengan rata-rata akurasi 2,5 meter untuk penentuan posisi awal pengguna, dan persentase rata-rata akurasi 94,8% untuk pendeteksian langkah pengguna serta 94,48% untuk estimasi arah hadap pengguna.

6.2 Saran

Berikut ini merupakan beberapa saran mengenai pengembangan lebih lanjut sistem *Real Time Tracking Indoor Position* berdasarkan hasil rancangan, implementasi dan uji coba yang telah dilakukan.

1. Memperbanyak jumlah dan variasi *data sample* sinyal *WiFi* sebagai solusi atas fluktuasi sinyal *WiFi*.
2. Memperbanyak lokasi uji coba pada seluruh gedung, tidak hanya pada satu lantai.
3. Menampilkan informasi lokasi pengguna tidak hanya pada saat penentuan posisi awal pengguna, namun juga pada saat sistem mendeteksi pergerakan pengguna.
4. Menampilkan posisi seluruh pengguna yang sedang menggunakan aplikasi.
5. Dapat digunakan di berbagai perangkat dengan resolusi yang berbeda-beda, karena aplikasi saat ini hanya dapat bekerja dengan baik pada *smartphone* tertentu.

DAFTAR PUSTAKA

[1] W. Waqar, Y. Chen and A. Vardy, "Smartphone positioning in sparse Wi-Fi environments," *Computer Communications,* pp. 1-9, 2015.

[2] S. Kumar, M. A. Qadeer and A. Gupta, "Location Based Services using Android (LBSOID)," 2009.

[3] M. F. Ghanianto, "Implementasi Indoor Localization Menggunakan Sinyal Wi-Fi dan Clustering Filtered K-Nearest Neighbors untuk Pelacakan Keberadaan Seseorang dan Evaluasi Akurasi Pelacakan di Kampus Teknik Informatika ITS," Surabaya, 2015.

[4] K. Curran, E. Furey, T. Lunney, J. Santos, D. Woods and A. Caughey, "An Evaluation of Indoor Location Determination Technologies," *Journal of Location Based Services,* vol. 5, pp. 61-78, 2011.

[5] G. Millette, Android Sensor Programming, Indianapolis: John Wiley & Sons Inc., 2012.

[6] L. J. Mitchell, PHP Web Services, Sebastopol: O Reilly Media, 2013.

[Halaman ini sengaja dikosongkan]

BIODATA PENULIS

Dinar Winia Mahandhira, biasa dipanggil Dinar, dilahirkan di kota Jember pada tanggal 12 Juli 1994. Penulis adalah anak ketiga dari tiga bersaudara dan dibesarkan di kota Jember, Jawa Timur. Penulis menempuh pendidikan di SDN Kepatihan 17 Jember, SMP Negeri 2 Jember dan SMA Negeri 1 Jember. Pada tahun 2012, penulis mengikuti SNMPTN Undangan dan diterima di Teknik Informatika Institut Teknologi Sepuluh Nopember Surabaya yang terdaftar dengan NRP 5112100002. Di jurusan Teknik Informatika ini, penulis mengambil rumpun mata kuliah Algoritma dan Pemrograman. Selama di kuliah, penulis banyak belajar mengenai pemrograman Java, Python, PHP, pemrograman perangkat bergerak. Penulis juga pernah menjadi assiten praktikum di beberapa mata kuliah seperti Sistem Basis Data, Teknologi Pembuatan Web, Data Mining, dll. Selain itu, penulis aktif di beberapa organisasi di antaranya Staff Kesejahteraan Mahasiswa HMTC Bersahabat, Staff Hubungan Luar HIMASA Jember, Staff Ahli Kesejahteraan Mahasiswa HTMC Berkarya, Staff Kementerian Kesejahteraan BEM ITS Muda Bersahabat, dan Asdirjen Kementerian Kesejahteraan BEM ITS Kolaborasi.