



INSTITUT TEKNOLOGI DEL
USER TRACKING WITH WI-FI ACCESS POINT

TUGAS AKHIR

13317008	Evi Yolenta Silalahi
13316009	Sunarti Barasa
13316013	Royda Venny Sitorus

FAKULTAS INFORMATIKA DAN TEKNIK ELEKTRO
PROGRAM STUDI TEKNOLOGI KOMPUTER

LAGUBOTI
AGUSTUS 2019



INSTITUT TEKNOLOGI DEL

USER TRACKING WITH WI-FI ACCESS POINT

TUGAS AKHIR

**Diajukan sebagai salah satu syarat untuk memperoleh
gelar Diploma Teknik**

13317008	Evi Yolenta Silalahi
13316009	Sunarti Barasa
13316013	Royda Venny Sitorus

**FAKULTAS INFORMATIKA DAN TEKNIK ELEKTRO
PROGRAM STUDI TEKNOLOGI KOMPUTER**

**LAGUBOTI
AGUSTUS 2019**

HALAMAN PERNYATAAN ORISINALITAS

Tugas Akhir ini adalah hasil karya saya sendiri, dan semua sumber baik yang dikutip maupun dirujuk telah saya nyatakan dengan benar.

Nama : Evi Yolenta Silalahi

NIM : 13317008

Tanda Tangan : 

Tanggal : Agustus 2020

Nama : Sunarti Barasa

NIM : 13317009

Tanda Tangan : 

Tanggal : Agustus 2020

Nama : Royda V Sitorus

NIM : 13317013

Tanda Tangan : 

Tanggal : Agustus 2020

HALAMAN PENGESAHAN

Tugas akhir ini diajukan oleh :
Nama : Evi Yolenta Silalahi
NIM : 13317008
Program studi : Diploma III Teknologi Komputer

Nama : Sunarti Barasa
NIM : 13317009
Program studi : Diploma III Teknologi Komputer

Nama : Royda V Sitorus
NIM : 13317013
Program studi : Diploma III Teknologi Komputer
Judul Tugas Akhir : User Tracking with Wi-Fi Access Point.

Telah berhasil dipertahankan dihadapan dewan penguji dan diterima sebagai bagian persyaratan yang diperlukan untuk memperoleh gelar Amd.kom program studi Diploma 3 Teknologi Informasi Fakultas Informatika dan Teknik Elektro Institut Teknologi Del.

DEWAN PENGUJI

Pembimbing : Pandapotan Siagian, ST, M.Eng (.)

Pembimbing : Ahmad Zatnika Purwalaksana, S.Si., M.Si (.)

Penguji : Istas Manalu, S.Si., M.Sc (.)

Penguji : Sari Muthia Silalahi, S.ST.,M.Sc (.)

Ditetapkan : Laguboti

Tanggal : Agustus 2020

KATA PENGANTAR

Segala puji dan syukur kepada Tuhan Yang Maha Esa atas berkat dan rahmat yang diberikan kepada penulis sehingga pengerjaan Tugas Akhir dengan judul **“User tracking with Wi-Fi Access Point ”** dapat diselesaikan dengan baik dan tepat waktu. Dokumen Tugas Akhir ini sebagai syarat untuk kelulusan Diploma III Teknologi Komputer, Fakultas Informatika dan Teknik Elektro, Institut Teknologi Del.

Pada masa pengerjaan Tugas Akhir, sangat banyak pihak – pihak yang berperan dalam membantu penulis menyelesaikan Tugas Akhir ini, oleh karena itu pada kesempatan ini penulis mengucapkan terima kasih yang sebesar – besarnya kepada:

1. Orangtua dan keluarga dari penulis Tugas Akhir ini, atas segala dukungan dan doa selama masa perkuliahan dan pengerjaan Tugas Akhir ini. Bapak Pandapotan Siagian, ST, M.Eng dan Ahmad Zatnika Purwalaksana, S.Si., M.Si sebagai dosen pembimbing yang telah menuntun dan memberi masukan selama penulisan Tugas Akhir ini. Hingga pengerjaan Tugas Akhir ini dapat berjalan dengan baik dan lancar.
2. Bapak Istas Manalu, S.Si., M.Sc dan Ibu Sari Muthia Silalahi, S.ST.,M.Sc sebagai dosen penguji yang telah memberikan pandangan kritikan dan saran yang sangat berguna dalam membangun demi keberhasilan pengerjaan Tugas Akhir ini.
3. Teman-teman yang telah berperan dalam memberi dukungan dan masukan dalam mengerjakan Tugas Akhir ini.

Dalam penulisan Tugas Akhir ini, penulis mengharapkan agar laporan Tugas Akhir ini dapat memberikan manfaat bagi semua pihak yang membacanya terutama pihak-pihak yang bergerak dalam bidang teknologi. Penulis menyadari dalam Tugas Akhir terdapat banyak kesalahan dan kekurangan, penulis memohon maaf atas segala kekurangan dan menerima saran dan kritik yang membangun kepada Penulis demi pengembangan Tugas Akhir ini agar lebih baik lagi. Akhir kata, semoga Tugas Akhir ini berguna dan bermanfaat bagi pembaca.

Laguboti, Agustus 2020

Evi Yolenta Silalahi

Sunarti Barasa

Royda V Sitorus

HALAMAN PERNYATAAN PERSETUJUAN PUBLIKASI DOKUMEN TUGAS AKHIR UNTUK KEPENTINGAN AKADEMIS

Sebagai sivitas akademik Institut Teknologi Del, penulis yang bertanda tangan di bawah ini:

Nama : Evi Yolenta Silalahi
NIM : 13317008
Program Studi : Diploma III Teknologi Komputer

Nama : Sunarti Barasa
NIM : 13317009
Program Studi : Diploma III Teknologi Komputer

Nama : Royda Venny Sitorus
NIM : 13317013
Program Studi : Diploma III Teknologi
Komputer Fakultas : Informatika dan Teknik
Elektro Jenis Karya : Tugas Akhir

Demi pengembangan ilmu pengetahuan, menyetujui untuk memberikan kepada Institut Teknologi Del **Hak Bebas Royalti Noneksklusif (*Non-exclusive Royalty-Fee Right*)** atas karya ilmiah penulis yang berjudul:

User tracking with Wi-Fi Access Point

beserta perangkat yang ada (jika diperlukan). Dengan Hak Bebas Royalti Noneksklusif ini Institut Teknologi Del berhak menyimpan, mengalih/media- format dalam bentuk pangkalan data (*database*), merawat, dan memublikasikan tugas akhir penulis selama tetap mencantumkan nama penulis sebagai penulis/pencipta dan sebagai pemilik Hak cipta.

Demikian pernyataan ini penulis buat dengan sebenarnya.

Dibuat di :
Laguboti Pada tanggal :
Agustus 2020

Yang menyatakan,



(Evi Yolenta Silalahi)



(Sunarti Barasa)



(Royda Venny Sitorus)

ABSTRAK

Nama : Evi Yolenta Silalahi
Program Studi : Diploma 3 Teknologi Komputer
Judul : *User Tracking with Wi-Fi Access Point*
Nama : Sunarti Barasa
Program Studi : Diploma 3 Teknologi Komputer
Judul : *User Tracking with Wi-Fi Access Point*
Nama : Royda V Sitorus
Program Studi : Diploma 3 Teknologi Informasi
Judul : *User Tracking with Wi-Fi Access Point*

Tugas Akhir yang berjudul “*User Tracking with Wi-Fi Access Point*” merupakan sebuah sistem *tracking* yang bertujuan untuk membaca *access point* yang terkoneksi, menampilkan IP serta MAC Address *access point* yang terkoneksi. Sistem ini akan menggunakan *access point* sebagai pemancar jaringan yang akan terhubung dengan pengguna dan menggunakan database yang akan digunakan sebagai tempat penyimpanan data dari pengguna yang terkoneksi dan akan di tampilkan pada aplikasi Android dan web server.

User Tracking with Wi-Fi Access Point ini merupakan sistem yang dapat bekerja dengan baik jika memiliki koneksi jaringan yang baik pula. Sistem ini juga memerlukan banyak *access point* agar dapat memperoleh banyak data-data pengguna.

Kelebihan dari sistem ini adalah sistem menggunakan aplikasi android untuk mempermudah pengguna dan sistem pada web server yang dapat menampilkan nama *user*, *IP address*, *MAC Address* dan menampilkan grafik per *access point*. Sedangkan kekurangan sistem *User Tracking with Wi-Fi Access Point* tidak dapat melakukan *disable/enable* dan dalam pengerjaan Tugas Akhir membutuhkan *access point* 3 buah agar dapat melakukan perbandingan grafik *access point*. Namun karena kekurangan alat hanya 1 *access point* yang digunakan dalam pengerjaan tugas akhir ini

Kata Kunci – *Tracking, Access Point, Wi-Fi, Users*

ABSTRACT

Name : Evi Yolenta Silalahi
Study program : Diploma 3 in Computer Technology
Title : User Tracking with Wi-Fi Access Points

Name : Sunarti Barasa
Study program : Diploma 3 in Computer Technology
Title : User Tracking with Wi-Fi Access Points

Name : Royda V Sitorus
Study program : Diploma 3 in Information Technology
Title : User Tracking with Wi-Fi Access Points

The Final Project, entitled "User Tracking with Wi-Fi Access Points" is a tracking system intended to read the connected access point, display the IP and MAC address of the connected access point. This system will use an access point as a network transmitter that will connect with users and use a database that will be used as a storage place for data from connected users and will be displayed on Android applications and web servers.

User Tracking with Wi-Fi Access Points is a system that can be used well if it has a good network connection too. This system also buys many access points so that it can get a lot of user data.

The advantage of this system is a system that uses the Android application to facilitate users and systems on a web server that can display user names, IP addresses, MAC addresses and display graphs per access point. While the lack of a system of User Tracking with Wi-Fi Access Points cannot deactivate / activate and in the process of Final Project requires 3 access points in order to be able to do a graph access point. However, due to lack of tools, only 1 access point is used in this final project

Keywords - Tracking, Access Points, Wi-Fi, Users

DAFTAR ISI

HALAMAN PERNYATAAN ORISINALITAS.....	ii
HALAMAN PENGESAHAN.....	iii
KATA PENGANTAR	iv
HALAMAN PERNYATAAN PERSETUJUAN PUBLIKASI DOKUMEN TUGAS AKHIR UNTUK KEPENTINGAN AKADEMIS	v
ABSTRAK.....	vi
ABSTRACT.....	vii
DAFTAR ISI.....	viii
DAFTAR TABEL.....	xi
DAFTAR GAMBAR	xii
BAB I.....	1
PENDAHULUAN	1
1.1 Latar Belakang	1
1.2 Rumusan Masalah.....	2
1.3 Tujuan	2
1.4 Lingkup	2
1.5 Batasan.....	2
1.6 Pendekatan	2
1.7 Sistematika Penyajian	3
1.8 Defenisi, Akronim dan Singkatan	4
BAB II.....	6
TINJAUAN PUSTAKA	6
2.1 Landasan Teori.....	6
2.1.1 Wi-Fi (Wireless Fidelity)	6
2.1.2 Access Point.....	7
2.1.3 Android studio	8
2.1.4 MAC Address (Media Access Control Address)	9
2.1.5 IP Address (Internet Protocol Address)	10
2.1.6 Switch	10
2.1.7 Database	12
2.2 Related Work	12
2.2.1 Indoor Localization and Tracking using Wi-Fi Access Points [4]	13
2.2.2 A user application-based access point selection algorithm for dense WLANs [2].....	13
2.2.3 Kontribusi Tugas Akhir	14
BAB III	17
ANALISIS DAN DESAIN	17
3.1 Analisis	17
3.1.1 Analisis Masalah.....	17

3.1.2	Analisis Pemecahan Masalah.....	19
3.1.3	Analisis Kebutuhan Sistem.....	19
3.2	Perancangan System	20
3.2.1	Design Perangkat Keras	21
3.2.2	Design Perangkat Lunak	21
3.3.3	Flowchat Android	22
3.3.4	<i>Flowchart</i> web server.....	23
3.3.5	Flowchart System.....	24
3.3.6	Use Case Diagram Web Server.....	25
3.3.7	Use Case Diagram Android	25
3.3.8	Use Case Tracking Skenario	26
3.2	Design	27
3.2.1	Desain Tampilan Android.....	27
3.2.3	Desain Tampilan Web.....	29
BAB IV		32
IMPLEMENTASI DAN PENGUJIAN		32
4.1	Implemetasi Sistem Jaringan	32
4.1.1	Konfigurasi Server ke switch.....	32
4.1.2	Konfigurasi Server dengan <i>Access Point</i>	33
4.2.1	Implementasi Fungsi Menu Login	34
4.2.2	Implementasi Fungsi Menu Tambah Access Point	35
4.2.3	Implementasi Fungsi Menu Update	35
4.2.4	Implementasi Fungsi Menu Delete	36
4.2.5	Implementasi Fungsi Menu Logout	37
4.3	Implentasi Android	37
4.3.1	Implementasi Navigation menu	38
4.3.2	Implementasi Layout.....	39
4.3.3	Implementasi Gradle	40
4.3.4	Implementasi Sqlite.....	41
4.3.5	Implementasi Menu pada Aplikasi	42
4.4	Implementasi Menghubungkan database MySQL dengan SQLite	44
4.5	Pengujian.....	44
4.5.1	Pengujian Sistem Jaringan	44
4.5.2	Pengujian Web server	46
4.5.3	Pengujian Android	48
BAB V		32
KESIMPULAN DAN SARAN.....		32
5.1	Kesimpulan	32
5.2	Saran	32

DAFTAR PUSTAKA	xii
Lampiran	xiv

DAFTAR TABEL

Tabel 1 Daftar Definisi	4
Tabel 2 Daftar Akronim dan Singkatan	5
Tabel 3. Kebutuhan Perangkat Keras.....	19
Tabel 4 Kebutuhan Perangkat Lunak.....	20

DAFTAR GAMBAR

Gambar 1. Cara Kerja <i>Wi-Fi</i>	6
Gambar 2. Topologi Peer-to-Peer	7
Gambar 3. Topologi Acces Point	8
Gambar 4. Tampilan Android Studio	9
Gambar 5. MAC Address	9
Gambar 6. IP Address	10
Gambar 7. Topologi penggunaan switch	11
Gambar 8 <i>System</i> yang sudah ada	17
Gambar 9 Tampilan User yang terhubung pada Unifi	18
Gambar 10 Tampilan grafik pada Unifi	18
Gambar 11. Perancangan <i>System</i>	21
Gambar 12 <i>Flowchart</i> Android	22
Gambar 13 <i>Flowchart</i> Web Server	23
Gambar 14 <i>Use Case Diagram</i> Web Server	25
Gambar 15 Use Case Diagram User Tracking with Wi-Fi Access Point	25
Gambar 16 Halaman Dashboard	27
Gambar 17 Halaman Channel Graph	28
Gambar 18 Halaman Time Graph	28
Gambar 19 Halaman Login web	29
Gambar 20 Halaman dashboard web	29
Gambar 21 Halaman Tambah Access Point web	30
Gambar 22 Halaman Update Access Point web	30
Gambar 23 Halaman Delete web	31
Gambar 24 Halaman Logout web	31
Gambar 25 Implementasi Sistem Jaringan	32
Gambar 26 Topologi Server ke switch	33
Gambar 27 Konfigurasi Switch di Server	33
Gambar 28 Konfigurasi Access Point	34
Gambar 29 Implementasi Fungsi Menu Login	35
Gambar 30 Implementasi Fungsi Menu Tambah Access Point	35
Gambar 31 Implementasi Fungsi Menu Update	36
Gambar 32 Implementasi Fungsi Menu Delete	36
Gambar 33 Code Membaca Access Point	37
Gambar 34 Implementasi Fungsi Menu Logout	37
Gambar 35 Navigation Menu	38
Gambar 36 Peletakan Layout dan Gambar Layout	39
Gambar 37 Desain dan Code Layout	39
Gambar 38 Langkah Membuka Folder Tempat Gradle	40
Gambar 39 Penyesuaian Tempat Folder Gradle	41
Gambar 40 Code Gradle	41
Gambar 41 Pemanggilan library sqllite helper	42
Gambar 42 Code membaca access point	43
Gambar 43 Code Menampilkan Grafik Access Point	43
Gambar 44 Code Menampilkan Grafik Access Point Perwaktu	43
Gambar 45 Code menghubungkan database MySQL dengan SQLite	44
Gambar 46 Pengujian IP Access Point pada Server	45
Gambar 47 Pengujian Access Point Terkoneksi	45
Gambar 48 Menu Login	46
Gambar 49 Menu Tambah Access Point	46
Gambar 50 Menu Update	47
Gambar 51 Menu Delete	47
Gambar 52 Menu Logout	48
Gambar 53 Menu Access Point	48
Gambar 54 Menu Channel Graph	49

Gambar 55 Menu Time Graph.....	50
--------------------------------	----

BAB I

PENDAHULUAN

Pada bab ini berisi penjelasan mengenai latar belakang, tujuan, ruang lingkup, pendekatan, dan sistematika penyajian Tugas Akhir

1.1 Latar Belakang

Pada era globalisasi ilmu pengetahuan dan teknologi berkembang secara cepat dan pesat. Hal tersebut dikarenakan masyarakat modern selalu membutuhkan teknologi dalam kehidupan sehari-hari terutama pada teknologi berupa *Wi-Fi*. Beberapa teknologi yang salah satunya merupakan teknologi yang digunakan untuk menentukan posisi dalam gedung berdasarkan sinyal nirkabel. Teknologi berupa *Wi-Fi* telah dikembangkan dengan sistem *positioning* dengan menentukan posisi suatu tempat di dalam gedung yang pada umumnya menggunakan *Global Positioning System* (GPS) tetapi ada kekurangan karena adanya masalah gangguan pada gedung^[1]. Beberapa peneliti telah mengembangkan konsep *indoor positioning system* dengan berbagai variasi cara dan metode, seperti penggunaan *multilayer perceptron* untuk penentuan^[2].

Di Institut Teknologi Del memiliki sistem aplikasi *Unifi* yang berguna untuk admin menghitung jumlah pengguna *access point* dan admin mengetahui *Wi-Fi* yang digunakan oleh pengguna (*user*) di gedung Institut Teknologi Del. Sistem yang telah ada masih memiliki banyak kekurangan yaitu sistem belum menggunakan aplikasi android dan tampilan grafik pada sistem belum terstruktur dengan baik pada sistem di Institut Teknologi Del.

Dalam tugas akhir ini, *Developer* akan membangun sebuah sistem yang berjudul *User Tracking with Wi-Fi Access Point*. Dalam pengerjaan tugas akhir ini, jaringan sangat dibutuhkan untuk melakukan pembacaan data dari *access point*. Data *access point* seperti *IP address*, *MAC address*, nama *device* dan membaca grafik *access point* sehingga *developer* perlu melakukan konfigurasi sistem jaringan antara server ke switch kemudian switch terhubung *access point*. Model lokasi yang digunakan sebagai tempat implementasi adalah Perumahan Desa Manduamas. Dalam implelementasi *User Tracking with Wi-Fi Access Point* ini akan di uji coba menggunakan aplikasi android untuk mampu membaca *access point* (*IP address*, *MAC address*, nama perangkat, vendor, dan grafik per *access point*) dan membaca data *access point* pada web server.

12 Rumusan Masalah

Rumusan masalah yang menjadi acuan pengerjaan tugas akhir ini adalah sebagai berikut:

1. Bagaimana mengkonfigurasi *access point* agar dapat di baca web server?
2. Bagaimana membangun web server untuk dapat menampung data *access point*?
3. Bagaimana membaca *access point* dan grafik *access point* pada android?

13 Tujuan

Berdasarkan latar belakang diatas, tujuan dari tugas akhir adalah

1. Membangun sebuah sistem jaringan dan mengkonfigurasi *access point* agar dapat di baca web server.
2. Web server juga dapat menampung data *access point*
3. Pada android dapat membaca *access point* dan menampilkan grafik *access point*.

14 Lingkup

Lingkup pembahasan dalam kajian ini bertujuan agar masalah yang akan diteliti dapat lebih terarah dan terfokus sesuai dengan apa yang direncanakan yaitu:

1. Menggunakan 1 buah *access point* sebagai sumber signal WiFi.
2. Menampilkan grafik *access point* pada android
3. Membaca *access point* pada android dan menambahkan *access point* pada server

15 Batasan

Batasan masalah dalam pengerjaan tugas akhir ini adalah :

- a. Dalam pengerjaan Tugas Akhir membutuhkan *access point* 3 buah agar dapat melakukan perbandingan grafik *access point*. Namun karena kekurangan alat hanya 1 *access point* yang digunakan dalam pengerjaan tugas akhir ini.
- b. Pada Analisis permasalahan sistem *Unifi* terdapat fungsi yang belum dapat diselesaikan yaitu *disable/enable*.

16 Pendekatan

Pendekatan yang dilakukan pada Tugas Akhir ini terdiri dari:

1. Perumusan Masalah

Pada tahap ini penulis mencari suatu permasalahan yang dapat diselesaikan dari lingkungan sekitar.

2. Melakukan Studi Literatur

Pada tahap ini penulis mencari referensi yang berkaitan dengan sistem yang akan dibuat dengan membaca dan memahami jurnal. Penulis juga melakukan perbandingan terhadap metode yang telah ada sebelumnya, yaitu tentang ruang lingkup dari metode yang dipakai dalam penelitian.

3. Membuat Gambaran *System Overview*

Pada tahap ini penulis menggambarkan bagaimana sistem *User tracking with Wi-Fi Access Point*.

4. *Design*

Pada tahap ini penulis merancang perangkat dan aplikasi *web* berdasarkan gambaran *system overview* yang telah dibuat.

5. Implementasi

Tahap ini merupakan tahapan untuk pembuatan sistem yang telah ditetapkan dari *design*. Selain itu, tahap ini juga merupakan jawaban dari pertanyaan yang ada pada rumusan masalah sehingga pada tahap terakhir penulis akan menarik kesimpulan dari hasil penelitian yang telah dilakukan.

6. *Testing and Analysis*

Pada tahap ini penulis melakukan pengujian terhadap implementasi yang telah dilakukan dan juga melakukan analisis terhadap hasil implementasi.

7. Dokumentasi

Penulis menarik kesimpulan dari implementasi dan *Testing and Analysis* dan menuliskan dalam laporan pengerjaan Tugas Akhir.

1.7 Sistematika Penyajian

Secara garis besar, dokumen ini disajikan dalam tiga bab, yaitu.

1. Bab I Pendahuluan

Pada bab ini berisi mengenai penjelasan mengenai latar belakang, tujuan pelaksanaan, ruang lingkup, pendekatan yang dilakukan dan sistematika penyajian Tugas Akhir.

2. Bab II Tinjauan Pustaka

Pada bab ini berisikan teori-teori yang digunakan untuk mengerjakan Tugas Akhir.

3. Bab III Analisis dan Perancangan Sistem

Pada bab ini dijelaskan mengenai perancangan sistem yang dilakukan saat pengimplementasian.

4. Bab IV Implementasi dan Pengujian

Pada bab ini berisi tentang implementasi dan pengujian sistem yang telah dibuat.

5. Bab V Kesimpulan dan Saran

Pada bab ini dituliskan mengenai kesimpulan dari pengerjaan Tugas Akhir.

18 Defenisi, Akronim dan Singkatan

Berikut adalah daftar defenisi, akronim dan singkatan yang digunakan dalam dokumen. Definisi yang digunakan pada dokumen ini dapat dilihat pada tabel.

Tabel 1 Daftar Definisi

No	Istilah	Definisi
1.	<i>Network</i>	sebuah sistem operasi yang terdiri atas sejumlah komputer dan perangkat jaringan lainnya yang bekerja bersama-sama untuk mencapai suatu tujuam yang sama atau suatu jaringan kerja yang terdiri dari titik-titik (<i>nodes</i>) yang terhubung satu sama lain, dengan atau tanpa kabel
2.	<i>System Overview</i>	Sistem yang berjalan atau yang ada saat ini
3.	<i>Designer</i>	Orang yang berperan merancang gambaran produk yang akan dibuat
4.	<i>Software</i>	suatu bagian dari sistem komputer yang tidak memiliki wujud fisik dan tidak terlihat karena merupakan sekumpulan data elektronik yang disimpan dan diatur oleh komputer berupa program yang dapat menjalankan suatu perintah.
5.	<i>Hardware</i>	sebuah komponen fisik pada komputer yang digunakan oleh sistem untuk menjalankan perintah yang telah diprogramkan atau dalam arti singkatnya sebuah komponen pada komputer yang bisa disentuh, dilihat dan diraba
6.	<i>Wi-Fi</i>	sebuah teknologi yang memanfaatkan peralatan elektronik untuk bertukar data secara nirkabel (menggunakan gelombang radio) melalui sebuah jaringan komputer, termasuk koneksi internet berkecepatan tinggi.
6.	<i>Tracking</i>	suatu proses pelacakan, yangbisa digunakan untuk pelacakan suatu objek, warna, garis , dan lain-lain.
7.	<i>Mapping</i>	Suatu sistem untuk memetakan keberadaan

Tabel 2 Daftar Akronim dan Singkatan

No.	Istilah	Deskripsi
1.	AP	<i>Access Point</i>
2.	GPS	<i>Global Positioning System</i>
3.	WAP	<i>Wireless Access Point</i>
4.	Wi-Fi	<i>Wireless Fidelity</i>
5.	WLAN	<i>Wireless Local Area Network</i>
6.	KNN	<i>K-Nearest Neighbors</i>
7.	SVM	<i>Support Vector Machine</i>
8.	IP	<i>Internet Protokol</i>
9.	LAN	<i>Local Area Network</i>
10.	SSID	<i>Service Set Identifier</i>
11.	SDK	<i>Software Development Kit</i>

BAB II

TINJAUAN PUSTAKA

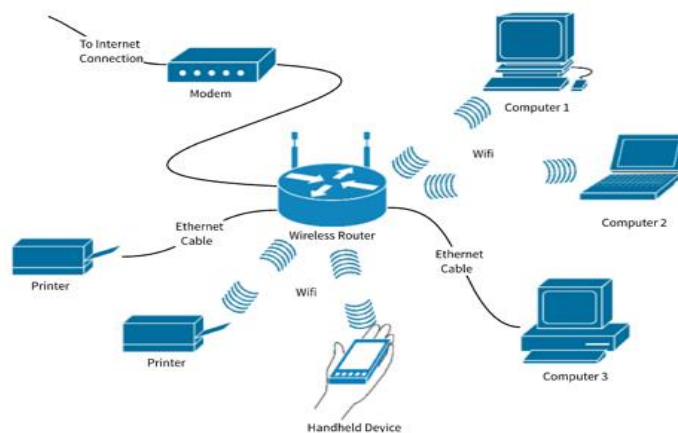
Pada bab ini dijelaskan tinjauan pustaka yang akan digunakan sebagai dasar teori dalam pengerjaan Tugas Akhir.

2.1 Landasan Teori

Pada bagian ini diuraikan secara ringkas mengenai komponen-komponen utama yang digunakan dalam *User Tracking with Wi-Fi Access Point*.

2.1.1 Wi-Fi (Wireless Fidelity)

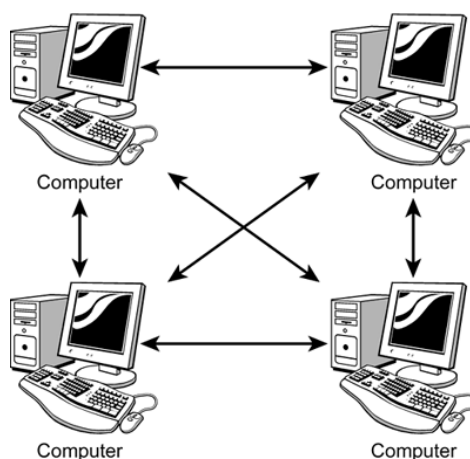
Wi-Fi (Wireless Fidelity) merupakan salah satu varian teknologi komunikasi dan informasi yang bekerja pada jaringan dan perangkat *Wireless Local Area Network (WLAN)*^[5]. Bagaimana masa depan Wi-Fi? Levkowitz berpendapat bahwa pertumbuhan hari ini sangat luas dalam dunia telekomunikasi, banyak produk baru, perangkat baru, baik perangkat lunak maupun perangkat keras sedang di kembangkan, Insinyur komputer bekerja sepanjang waktu untuk endatangkan ide-ide baru yang mungkin di capai^[6]. Sesuai dengan namanya, perangkat yang dibutuhkan untuk mengakses internet dengan layanan ini juga nirkabel. Jika dibandingkan dengan internet lainnya, *Wi-Fi* lebih mudah instalasinya. Namun, pastinya harus ada perangkat utama seperti *wireless* atau *access point* dan jaringan internet. Kecepatan akses internet *wireless* tergolong tinggi dan bisa mencapai 54 Mbps. Selain itu koneksi cenderung stabil sehingga proses pengaksesan layanan internet bisa dilakukan dengan lancar. *Wi-Fi* memiliki kelebihan dan kekurangan. Berikut gambar cara kerja dari bagaimana *Wi-Fi* terhubung.



Gambar 1. Cara Kerja Wi-Fi

<https://www.nesabamedia.com/pengertian-wifi-beserta-fungsi-dan-cara-kerja-wifi/>

Pada saat melakukan koneksi pengaksesan *Wi-Fi* Ada dua mode akses koneksi *Wi-Fi*, yaitu diantaranya yaitu pertama *Ad-Hoc* sebagai mode koneksi ini adalah mode dimana beberapa komputer terhubung secara langsung, atau lebih dikenal dengan istilah *Peer-to-Peer* yaitu jaringan komputer dimana setiap host dapat menjadi server dan juga menjadi client secara bersamaan seperti gambar dibawah ini.



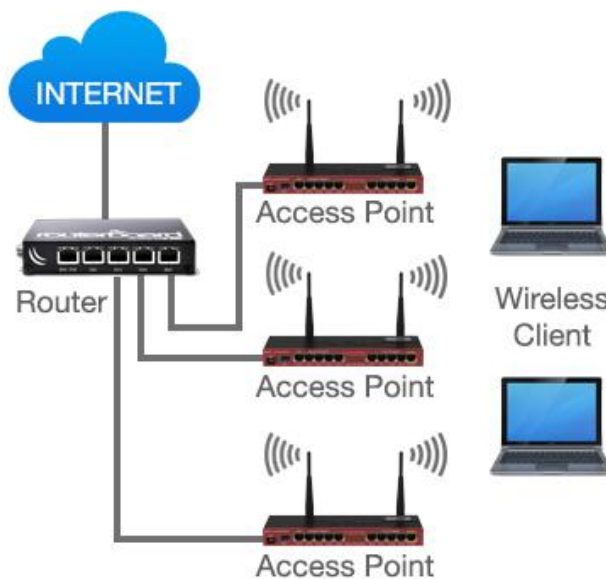
Gambar 2. Topologi Peer-to-Peer

(<https://www.dictio.id/t/apa-yang-dimaksud-dengan-topologi-peer-to-peer/13740/2>)

Keuntungannya, lebih murah dan praktis bila yang terkoneksi hanya 2 atau 3 komputer, tanpa harus membeli access point dan kedua Infrastruktur Menggunakan *Access point* yang berfungsi sebagai pengatur lalu lintas data, sehingga memungkinkan banyak *Client* dapat saling terhubung melalui jaringan (*Network*).

2.1.2 Access Point

Access point adalah sebuah perangkat jaringan yang berisi sebuah *transceiver* dan antena untuk transmisi dan menerima sinyal ke dan dari *client remote*. *Wireless Access Point* (WAP) adalah alat yang di gunakan untuk menghubungkan alat-alat dalam suatu jaringan dari dan ke jaringan *wireless*. *Access Point* adalah hub bagi jaringan *wireless* baik itu di ruangan maupun di jaringan dalam kota. Fungsi dari Access Point adalah mengirim dan menerima data, sebagai buffer data antara WLAN dengan Wired LAN, mengkonversi sinyal frekuensi radio (RF) menjadi sinyal digital yang akan disalurkan melalui kabel atau disalurkan ke perangkat WLAN yang lain dengan dikonversi ulang menjadi sinyal frekuensi radio^[7].



Gambar 3. Topologi Acces Point

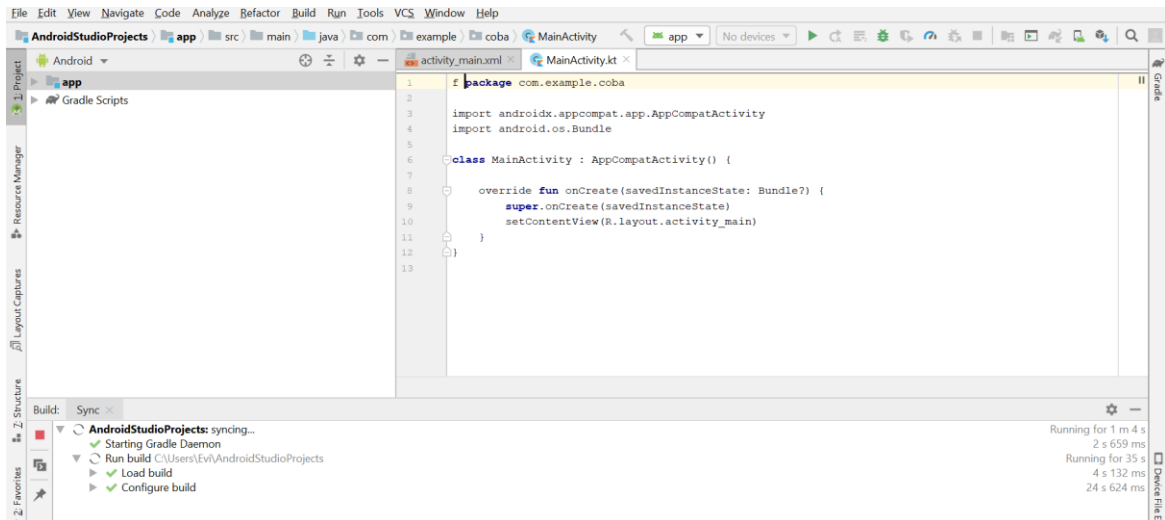
(http://www.mikrotik.co.id/artikel_lihat.php?id=167)

Access point bekerja dengan menyediakan koneksi antara jalur data sinyal RF yang dibentuk oleh *Wi-Fi* dengan jalur data elektrik yang dibentuk oleh kabel *ethernet*. Selain itu, *access point* juga melakukan pengontrolan akses, enkripsi data, toleransi kesalahan, serta manajemen jaringan.

2.1.3 Android studio

Android Studio merupakan Integrated Development Environment (IDE) resmi untuk pengembangan aplikasi Android yang diumumkan pada Konferensi Google I/O, Android Studio tersedia secara bebas dibawah Lisensi Apache^[8]. *Android* berubah menjadi *platform* yang begitu cepat dalam melakukan inovasi. Hal ini tidak lepas dari pengembangan utama dibelakangnya, yaitu Google. Google yang mengakuisisi *Android* dan kemudian membuatkan sebuah *platform*.

Platform android terdiri dari Sistem Operasi berbasis *Linux*, sebuah GUI (*Graphic User Interface*), sebuah *web browser* dan Aplikasi Studio *End-User* yang dapat di download dan juga para pengembang bisa dengan leluasa berkarya serta menciptakan aplikasi yang terbaik dan terbuka untuk digunakan oleh berbagai macam perangkat.



Gambar 4. Tampilan Android Studio

aplikasi *Android Studio* juga telah memberi akses ke *Android Software Development Kit* (SDK). SDK ini bisa disebut sebagai ekstensi dari kode Java yang memperbolehkannya untuk berjalan dengan mulus pada perangkat atau *device Android*. Jadi, kalau Java dibutuhkan untuk menulis programnya, *Android SDK* diperlukan untuk menjalankan programnya di *Android*.

2.1.4 MAC Address (Media Access Control Address)

Dalam sebuah komputer, MAC address adalah nomor khusus yang berada di setiap network card, switch, router, Access Point dan apasaja yang dapat terhubung dengan perangkat lain dalam, mekanisme jaringan ^[9]. *MAC Address* mengizinkan perangkat-perangkat dalam jaringan agar dapat berkomunikasi antara satu dengan yang lainnya. Sebagai contoh, dalam sebuah jaringan berbasis teknologi *ethernet*, setiap header dalam frame Ethernet mengandung informasi mengenai *MAC address* dari komputer sumber (*source*) dan *MAC address* dari komputer tujuan (*destination*).

Network Connection Details	
Network Connection Details:	
Property	Value
Connection-specific DNS ...	
Description	Intel(R) Dual Band Wireless-AC 3165
Physical Address	7C-67-A2-BA-B6-07
DHCP Enabled	Yes
IPv4 Address	192.168.43.67
IPv4 Subnet Mask	255.255.255.0
Lease Obtained	02 January 2020 12:51:15
Lease Expires	02 January 2020 14:21:15
IPv4 Default Gateway	192.168.43.1
IPv4 DHCP Server	192.168.43.1
IPv4 DNS Server	192.168.43.1
IPv4 WINS Server	
NetBIOS over Tcpip Enab...	Yes
Link-local IPv6 Address	fe80::202e:6c9b:860e:4864%21
IPv6 Default Gateway	
IPv6 DNS Server	

Gambar 5. MAC Address

Beberapa perangkat, seperti halnya *bridge* dan *switch Layer-2* akan melihat pada informasi *MAC address* dari komputer sumber dari setiap *frame* yang ia terima dan menggunakan informasi *MAC address* ini untuk membuat “tabel *routing*” internal secara dinamis. Perangkat-perangkat tersebut pun kemudian menggunakan tabel yang baru dibuat itu untuk meneruskan *frame* yang ia terima ke sebuah *port* atau segmen jaringan tertentu di mana komputer atau *node* yang memiliki *MAC address* tujuan berada.

2.1.5 IP Address (Internet Protocol Address)

IP address adalah metode pengalamatan pada jaringan komputer dengan memberikan sederet angka pada komputer (*host*), router atau peralatan jaringan lainnya^[10]. *IP address* sebenarnya bukan diberikan kepada komputer (*host*) atau *router*, melainkan pada *interface* jaringan dari *host* / *router* tersebut. IP (*Internet protocol*) sendiri di desain untuk interkoneksi sistem komunikasi komputer pada jaringan paket *switched*. Pada jaringan TCP/IP, sebuah komputer diidentifikasi dengan alamat IP. Tiap-tiap komputer memiliki alamat IP yang unik, masing-masing berbeda satu sama lainnya. Hal ini dilakukan untuk mencegah kesalahan pada *transfer* data. Terakhir, protokol data akses berhubungan langsung dengan media fisik. Berikut contoh *IP address* yang biasa digunakan.

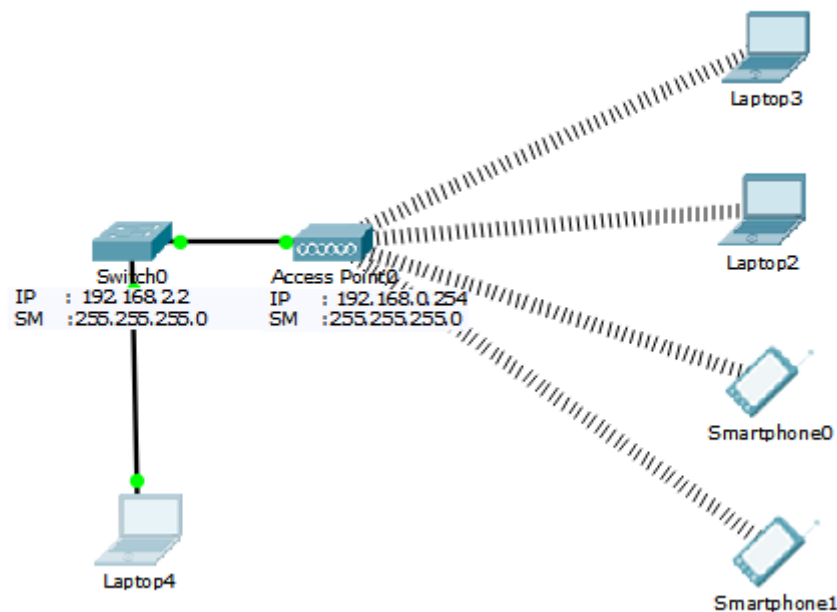
```
Connection-specific DNS Suffix . : 
Link-local IPv6 Address . . . . . : fe80::cdb1:f0f3:dc3d:22b1%6
IPv4 Address. . . . . : 192.168.118.1
Subnet Mask . . . . . : 255.255.255.0
Default Gateway . . . . . :
```

Gambar 6. IP Address

Secara umum protokol ini bertugas untuk menangani pendeteksian kesalahan pada saat *transfer* data, namun untuk komunikasi datanya, IP mengimplementasikan dua fungsi dasar yaitu *addressing* dan *fragmentasi*.

2.1.6 Switch

Switch dapat berfungsi sebagai penghubung antara beberapa perangkat yang terdapat di jaringan komputer. Misalnya saja perangkat seperti komputer, router, modem dan juga perangkat yang lainnya. Switch menerima pesan yang telah dihubungkan dengan nya dan kemudian akan meneruskan atau mengirimkan pesan tersebut ke beberapa perangkat.



Gambar 7. Topologi penggunaan switch

(<https://blog.dimensidata.com/pengertian-jenis-dan-fungsi-switch-pada-jaringan-komputer/>)

Switch dalam sebuah jaringan pada dasarnya dapat dibedakan menjadi beberapa jenis, yakni:

a. *Fast Forward / Cut through*

Jenis switch yang pertama ini hanya melakukan pengecekan alamat tujuan yang terletak pada *header frame*. Kemudian *frame* ini akan dilanjutkan kepada *host* tujuan. Kondisi yang terjadi inipun dapat membuat *latency time*. Meskipun begitu, switch jenis ini merupakan yang tercepat di jenisnya.

b. *Store and Forward*

Switch dengan jenis ini biasanya akan menyimpan *frame* untuk rentang waktu tertentu yang kemudian akan di cek terlebih dahulu oleh sistem CRC (*Cyclic Redudancy Check*) yang kemudian akan diteruskan menuju host yang menjadi tujuannya. Jika ditemukan adanya *frame* yang *error*, maka akan dibuang. Switch ini merupakan switch yang paling dipercaya di antara yang lainnya.

c. *Modified Cut through atau Fragment free Switch*

Switch jenis ini akan melakukan pemeriksaan pada 64 byte pertama dari *frame*. Jika ada *frame* yang mengalami kesalahan dikarenakan tabrakan, maka *frame* tersebut biasanya tidak akan diteruskan. Hal ini akan selalu menjamin frame untuk sampai pada tujuan yang dimaksud. Jumlah 64 *byte* ini dipilih karena merupakan jumlah minimum yang

dianggap krusial dan penting untuk melakukan pengecekan apakah sebuah frame baik-baik saja atau *error*.

d. Adaptive Switching Switch

Ini dibuat untuk dioperasikan pada *cut through* dengan model normal. Namun jika ditemukan kealahan yang dianggap terlalu tinggi, maka switch biasanya akan melakukan konfigurasi kembali secara otomatis yang kemudian akan dijalankan pada *mode store and forward*.

2.17 Database

Database adalah pengolahan data dalam pengingat eksternal (misal, hardisk) yang memungkinkan seseorang dengan mudah menyimpan data dan sekaligus menggunakan ketika memerlukannya^[12]. Pada pengerjaan proyek ini, kami akan menggunakan dua buah database yaitu Mysqli dan SQLite. MySQLi merupakan salah satu ekstensi PHP untuk mengakses fungsional yang disediakan MySQL 4.1 ke atas. Jika pada tulisan sebelumnya mengakses MySQL dengan menggunakan MySQL Extension, MySQL Improved Extension ditujukan agar dapat menggunakan fitur MySQL versi 4.1.3 ke atas, sedangkan ekstensi MySQL lama diperuntukkan untuk versi MySQL sebelumnya^[13]. SQLite menggunakan kelas manajemen basis data untuk menyimpan basis datanya sendiri^[14]. Untuk menghubungkan kedua database ini harus dilakukan synchron agar data yang terdapat pada database MySQL dapat dilihat dari database SQLite. Ada terdapat beberapa cara untuk melakukan pensynchronan database Yang pertama adalah dengan menggunakan syncadapter. Tetapi ini hanya akan menarik data dari server dan Anda tidak akan memiliki sinkronisasi segera setelah pembaruan dilakukan. Yang kedua adalah dengan menggunakan Pemberitahuan *push* . Dengan FCM, *firebase cloud messaging*, atau GCM, *google cloud messaging*, Anda dapat memberi tahu server Anda untuk mengirim pesan ke setiap telepon yang terhubung. Ponsel Anda kemudian dapat menangani pesan dan menyinkronkan dirinya dengan basis data Anda. *Obvisouly* dalam solusi mana pun Anda tidak akan memiliki akses langsung ke database Anda karena membawa masalah keamanan besar sehingga Anda harus melewati server web dengan API untuk menghubungkan aplikasi Anda ke database Anda.

2.2 Related Work

Pada bagian ini diuraikan secara ringkas mengenai penelitian yang berkaitan dengan pengerjaan Tugas Akhir.

2.2.1 Indoor Localization and Tracking using Wi-Fi Access Points [4]

Sistem ini memperkenalkan sistem pelacakan baru yang mengandalkan *Wi-Fi* dan perangkat seluler. Tujuan utama tidak hanya untuk membuat pemanfaatan terbaik dari infrastruktur yang ada tersedia dalam suatu organisasi tetapi membuat penyebaran sistem yang paling komersial dengan menggunakan teknologi yang sudah tersedia untuk konsumen. Di masa depan, sistem ini dapat diintegrasikan dengan pelacakan luar dan pemosisian untuk membentuk sistem lengkap yang akan membantu pengguna untuk memungkinkan pelacakan untuk lokasi *indoor* dan *outdoor*. Sistem dalam ruangan untuk pelacakan pengguna dan perangkat karena alasan keamanan juga dapat menjadi ruang lingkup sistem di masa depan. Temuan lokasi dengan menggunakan teknologi nirkabel adalah salah satu teknologi jaringan sensor nirkabel yang muncul dan penting. GPS dapat digunakan untuk area luar saja, tidak dapat digunakan untuk melacak pengguna di dalam gedung. Lokasi dalam ruangan termasuk bangunan seperti bandara, mal besar, supermarket, universitas, dan infrastruktur besar. Masalah signifikan yang dipecahkan sistem ini adalah melacak pengguna di dalam gedung. Lokasi dalam ruangan yang akurat dapat ditemukan dengan menggunakan Indikasi Kekuatan Sinyal yang Diterima (RSSI). Perangkat keras tambahan tidak diperlukan untuk RSSI, dan terlebih lagi, mudah dimengerti. Nilai RSS (Kekuatan Sinyal yang Diterima) dihitung dengan bantuan *Access Point* Wi-Fi dan perangkat seluler. Sistem harus menyediakan lokasi yang tepat dari pengguna dan juga melacak pengguna. Makalah ini menyajikan sistem yang membantu dalam mencari tahu lokasi yang tepat dan pelacakan perangkat seluler di lingkungan dalam ruangan. Ini juga dapat digunakan untuk menavigasi pengguna ke tujuan yang diperlukan menggunakan fungsi navigasi.

2.2.2 A user application-based access point selection algorithm for dense WLANs [2]

Sistem ini memiliki skema pemilihan *access point* (AP) saat ini sebagian besar didasarkan pada kekuatan sinyal yang diterima, tetapi berkinerja buruk dalam banyak situasi. Untuk mengatasi masalah ini, sejumlah skema alternatif mengumpulkan dan menganalisis beban aktual dari setiap *access point* (AP). Namun, skema ini dapat menimbulkan latensi yang signifikan dan memberi sinyal *overhead* dalam jaringan area lokal nirkabel (WLAN) yang padat. Berbasis aplikasi pengguna yang mempertimbangkan informasi historis tentang kinerja *access point* (AP). Tanpa mendorong aktivitas pensinyalan apa pun, skema memantau jumlah lalu lintas jaringan yang digunakan oleh aplikasi dan memperkirakan throughput *access point*

(AP) yang dapat dicapai. Skema kami menggunakan karakteristik lalu lintas aplikasi dengan tujuan untuk memprediksi kinerja *access point* (AP) secara akurat. Dengan menggunakan studi pengukuran di lingkungan WLAN yang padat menunjukkan bahwa skema mencapai throughput yang lebih tinggi dan latensi asosiasi yang lebih rendah daripada skema yang ada di tempat-tempat yang sangat mudah diakses oleh pengguna.

2.2.3 Kontribusi Tugas Akhir

Berdasarkan jurnal yang berkaitan di atas maka, penulis membuat *User Tracking with Wi-Fi Access Point* untuk pengerjaan Tugas Akhir. Berikut merupakan kontribusi yang dilakukan penulis untuk membuat *User Tracking with Wi-Fi Access Point* sebagai pembeda dari sistem yang sudah ada.

1. Sistem *User Tracking with Wi-Fi Access Point* menggunakan sistem dimana pengguna dapat melakukan monitoring jumlah *users*, *ip address*, *MAC address* dan. Pada Sistem ini admin juga dapat melihat grafik *access point* secara terpisah dan data-data hasil monitoring tersimpan pada database.
2. Data-data yang tersimpan pada server dapat melakukan *remote* menggunakan aplikasi *android*.

BAB III

ANALISIS DAN DESAIN

Pada Bab ini menjelaskan mengenai hasil analisis terhadap system yang dibangun.

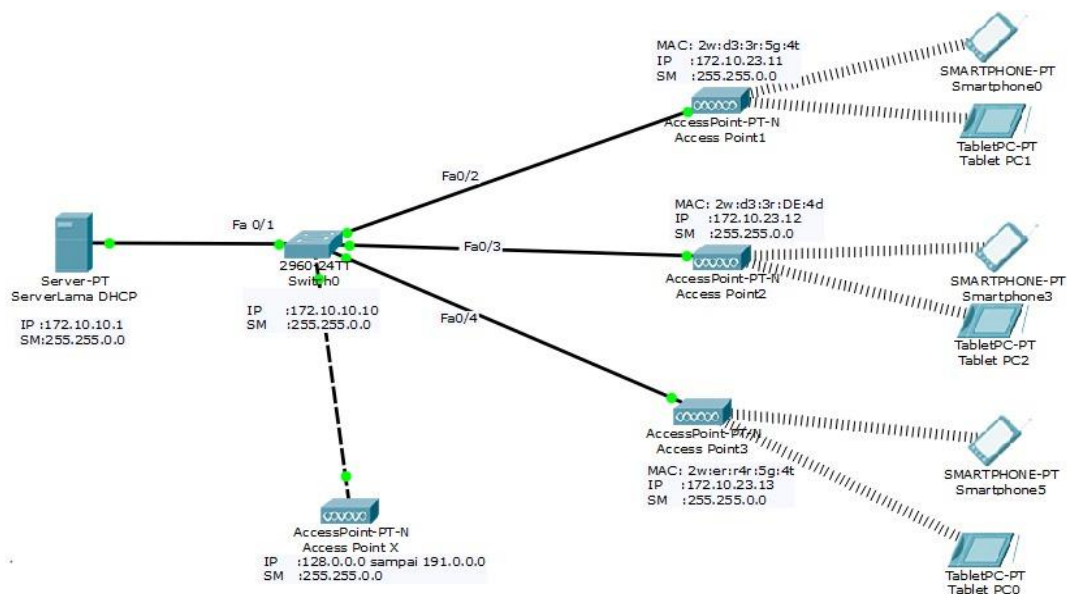
3.1 Analisis

Pada subbab ini akan dibahas mengenai analisis masalah, pemecahan masalah dan kebutuhan system untuk menyelesaikan permasalahan yang akan dihadapi selama penyelesaian Proyek Akhir.

3.1.1 Analisis Masalah

Masalah yang menjadi latar belakang dibuatnya proyek ini adalah:

1. Admin tidak dapat mengakses *Unifi* melalui aplikasi android
2. Grafik yang ditampilkan di *Unifi* bukan per *access point* namun keseluruhan *access point* yang ada di IT Del.
3. Nama *device* yang terhubung ke *access point* yang di tampilkan di *dashboard Unifi* bukan nama *user* tapi nama mesin *device* yang terhubung.



Gambar 8 System yang sudah ada

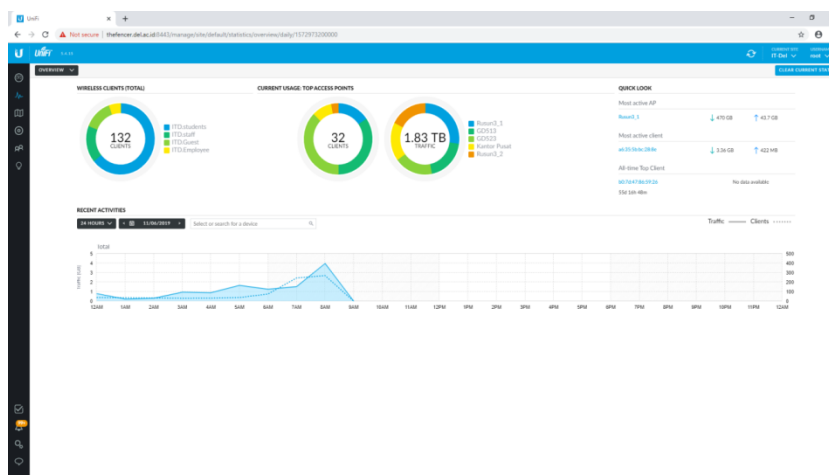
Pada gambar 8 diatas dapat dilihat tampilan topologi di sistem yang sudah ada. Perangkat yang digunakan adalah server, switch, dan *access point*. Server pada topologi berguna untuk menyimpan data pengguna yang terhubung pada *access point*. Switch berfungsi sebagai penghubung antara server dengan *access point* yang ada di IT Del. *Access point* berfungsi sebagai pemancar sinyal yang akan digunakan oleh *user*. Ketiga perangkat tersebut saling terhubung sehingga dapat digunakan oleh *user*.

Kelancaran penggunaan *access point* oleh *user* dapat di monitor dengan sistem *Unifi*. Namun, sistem *Unifi* memiliki beberapa kekurangan diantaranya yaitu tidak dapat mengakses *Unifi* melalui aplikasi android, grafik yang ditampilkan di *Unifi* bukan per-*access point*, dan nama *device user* yang ditampilkan nama mesin *device* yang terhubung. Gambaran sistem *Unifi* tersebut ada pada gambar 9 dibawah ini.

DEVICE NAME	IP ADDRESS	STATUS	MODEL	VERSION	UPTIME	ACTIONS
R-Down OUTDOOR	172.23.40.12	CONNECTED	Unifi AP-Outdoor	3.12.2920	21d 25m 13s	LOCATE RESTART UPGRADE
GD521	172.20.9.32	CONNECTED	Unifi AP-Pro	3.7.58.6385	65d 20h 27m 44s	LOCATE RESTART UPGRADE
Switch FTI 1	172.20.39.2	CONNECTED	Unifi Switch 8 POE-150W	3.12.3861	21d 25m 40s	LOCATE RESTART UPGRADE
Kantor Pusat	172.20.9.18	CONNECTED	Unifi AP	3.7.58.6385	4d 18h 10m 51s	LOCATE RESTART UPGRADE
Switch FTI 3	172.20.39.4	CONNECTED	Unifi Switch 8 POE-150W	3.12.3861	21d 27m	LOCATE RESTART UPGRADE
GD523	172.20.9.43	CONNECTED	Unifi AP-Pro	3.7.58.6385	4d 18h 32m 49s	LOCATE RESTART UPGRADE
GD512	172.20.9.33	CONNECTED	Unifi AP	3.12.2920	78d 10h 47m 34s	LOCATE RESTART UPGRADE
GD712	172.20.9.13	CONNECTED	Unifi AP-Outdoor	3.12.2920	8d 15m 8s	LOCATE RESTART UPGRADE
Town House	172.23.40.47	CONNECTED	Unifi AP-Outdoor	3.12.2920	14d 18h 58m 38s	LOCATE RESTART UPGRADE
FTI LL3_1	172.20.9.45	CONNECTED	Unifi AP-Pro	3.7.58.6385	1d 19h 34m 14s	LOCATE RESTART UPGRADE
GD 722	172.20.9.12	CONNECTED	Unifi AP	3.7.58.6385	15d 18h 6m 44s	LOCATE RESTART UPGRADE
RMK	172.20.8.10	CONNECTED	Unifi AP-Pro	3.7.58.6385	27d 19h 3m 12s	LOCATE RESTART UPGRADE
Auditorium	172.20.9.26	CONNECTED	Unifi AP-Pro	3.7.58.6385	21d 25m 44s	LOCATE RESTART UPGRADE
GD723	172.20.1.28	CONNECTED	Unifi AP-Pro	3.7.58.6385	6d 12m 53s	LOCATE RESTART UPGRADE
Rusun3_2	172.16.17.70	CONNECTED	Unifi AP-Pro	3.7.58.6385	21d 26m 36s	LOCATE RESTART UPGRADE
FTI LL4_1	172.20.5.15	CONNECTED	Unifi AP-Pro	3.7.58.6385	1d 18h 15m 49s	LOCATE RESTART UPGRADE
GD513	172.20.9.17	CONNECTED	Unifi AP-Pro	3.12.2920	4d 21h 25m 12s	LOCATE RESTART UPGRADE
Library	172.20.10.9	CONNECTED	Unifi AP-Pro	3.7.58.6385	5d 1h 15m 29s	LOCATE RESTART UPGRADE
Rusun3_1	172.16.17.108	CONNECTED	Unifi AP-Pro	3.7.58.6385	21d 26m 40s	LOCATE RESTART UPGRADE
GD522	172.20.9.31	CONNECTED	Unifi AP-Pro	3.7.58.6385	4d 21h 25m 16s	LOCATE RESTART UPGRADE
GD524	172.22.9.16	CONNECTED	Unifi AP-Pro	3.7.58.6385	4d 18h 35m 8s	LOCATE RESTART UPGRADE
FTI LL3_2	172.20.7.10	CONNECTED	Unifi AP-Pro	3.7.58.6385	21d 26m 34s	LOCATE RESTART UPGRADE
FTI LL1_1	172.20.40.100	CONNECTED	Unifi AP-Pro	3.7.58.6385	1d 12h 35m 17s	LOCATE RESTART UPGRADE
Yayasan Cabang Baru	172.16.17.60	CONNECTED	Unifi AP-Pro	3.7.58.6385	11d 17h 46m 45s	LOCATE RESTART UPGRADE
GD 825	172.20.9.21	CONNECTED	Unifi AP-Pro	3.7.58.6385	1h 42m 9s	LOCATE RESTART UPGRADE
GD514-OUTDOOR	172.20.9.24	CONNECTED	Unifi AP-Outdoor	3.7.58.6385	4d 18h 10m 41s	LOCATE RESTART UPGRADE
FTI LL2_2	172.20.1.119	CONNECTED	Unifi AP-Pro	3.7.58.6385	1d 18h 17m 36s	LOCATE RESTART UPGRADE

Gambar 9 Tampilan User yang terhubung pada Unifi

Pada gambar 10 dapat dilihat bahwa *system* yang sudah ada sebelumnya menampilkan daftar Access Point, IP, Status, Model, Versi, serta lama Access point sudah aktif.



Gambar 10 Tampilan grafik pada Unifi

Pada Gambar 10 dapat dilihat system yang sudah ada menampilkan grafik Access Point, Jumlah user yang terkoneksi dalam satu grafik.

3.1.2 Analisis Pemecahan Masalah

Berdasarkan permasalahan yang sudah di jelaskan pada subbab sebelumnya, maka penulis berusaha untuk memecahkan masalah yang ada dengan cara sebagai berikut:

1. *Unifi* tidak dapat di otak-atik untuk menghubungkan ke aplikasi android sehingga developer membangun sistem baru
2. Field- field di *Unifi* hanya menampilkan (nama mesin *user*, *IP address*, *MAC address*, dan nama *access point* yang digunakan) dan sistem *Unifi* tampilan grafik bukan per-Access point tapi keseluruhan sehingga developer membangun sistem pada web server yang dapat menampilkan nama *user*, *IP address*, *MAC Address* dan menampilkan grafik per *access point*.
3. Developer membangun android yang terhubung dan akses ke web server.

3.1.3 Analisis Kebutuhan Sistem

Pada subbab ini akan dijelaskan mengenai kebutuhan yang diperlukan selama proses pengerjaan *User Tracking with Wi-Fi Access Point*. Kebutuhan tersebut mencakup perangkat keras serta perangkat lunak yang akan digunakan.

3.1.3.1 Kebutuhan Perangkat Keras

Perangkat keras yang dibutuhkan selama pengerjaan Tugas Akhir ini dapat dilihat pada Tabel.

Tabel 3. Kebutuhan Perangkat Keras

Perangkat Keras	Spesifikasi	Keterangan
Laptop 1	Lenovo ideapad 110-14ISK <i>Operating System</i> : Windows 10 Pro 64-bit <i>Memory</i> : RAM 4GB DDR4.	Membuat dokumen Tugas Akhir, Sebagai <i>host</i> yang terkoneksi dengan jaringan
Laptop 2	Merk : Lenovo G40 Processor : Intel Corei5 RAM : 8.00 GB Harddisk : 1TB	Sebagai Server yang mengatur konfigurasi jaringan
Device(<i>Handphone</i> , Laptop)	<i>Handphone</i> dan laptop yang dapat terkoneksi dengan Jaringan <i>Wi-Fi</i>	Sebagai <i>host</i> yang terkoneksi dengan jaringan <i>Wi-Fi</i>

<i>Access Point</i>	Merk :D-link Model :DGS-1024C	Memancarkan sinyal untuk ke <i>device</i> lain seperti Laptop dan Ponsel
<i>Switch</i>	Merk :D-link Model :DGS-1024C	Sebagai penghubung server dan <i>access point</i> agar host dapat terhubung ke <i>access point</i> dan server dapat membaca ID <i>host</i> .

3.1.3.2 Kebutuhan Perangkat Lunak

Perangkat lunak yang dibutuhkan selama pengerjaan Tugas Akhir ini dapat dilihat pada Tabel.

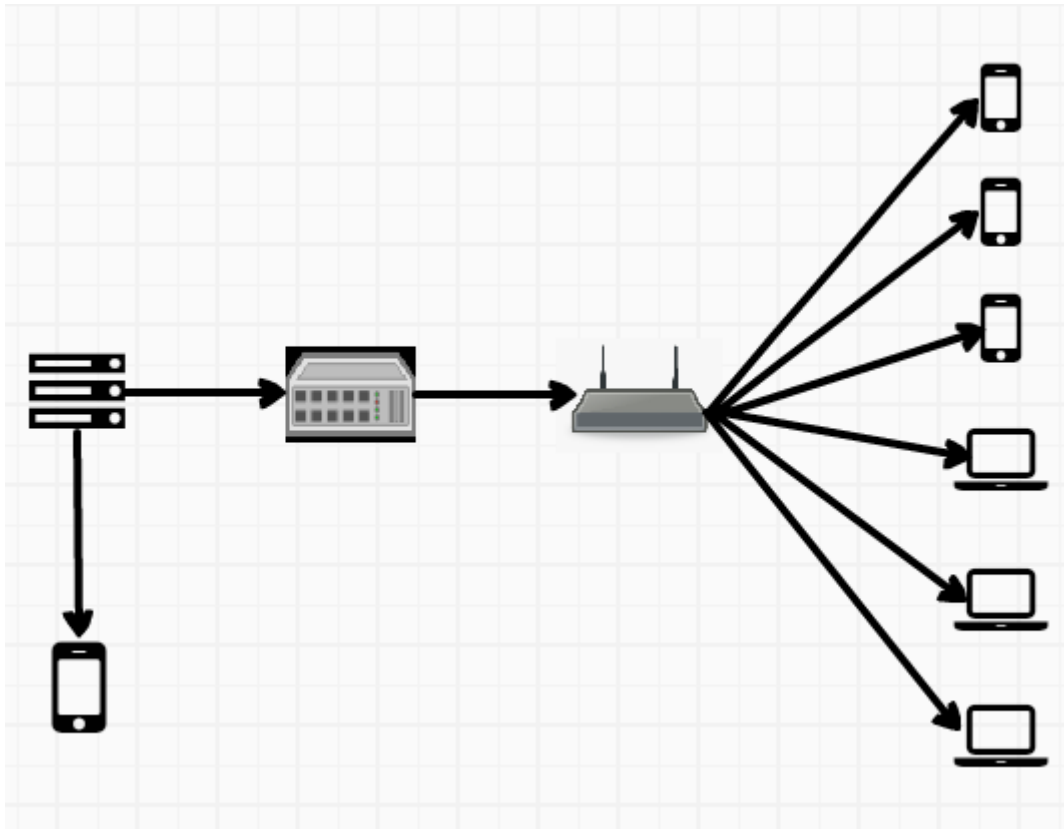
Tabel 4 Kebutuhan Perangkat Lunak

No.	Perangkat lunak	<i>Specification</i>
1.	<i>Operating System</i>	Windows 10 Pro 64-bit
2.	<i>Development Tools</i>	Android Studio
3.	<i>Programming Language</i>	Java
4.	<i>Database Tools</i>	Firebase
5.	<i>Design Tools</i>	Cisco Packet Tracer, Bizagi
6.	<i>OFFICE</i>	Ms. Office 2016

3.2 Perancangan System

Perancangan merupakan tahap awal sebelum membangun atau mengerjakan tugas/sistem, dalam pengerjaan Proyek *User Tracking with Wi-Fi Access Point* akan menggunakan *software* dan *hardware*. Oleh karena itu dengan tahap perancangan ini, segala yang dapat menghambat proses pengerjaan *system* ini dapat diminimalisir.

Perancangan system *User Tracking with Wi-Fi Access Point* dapat dilihat pada gambar



Gambar 11. Perancangan System

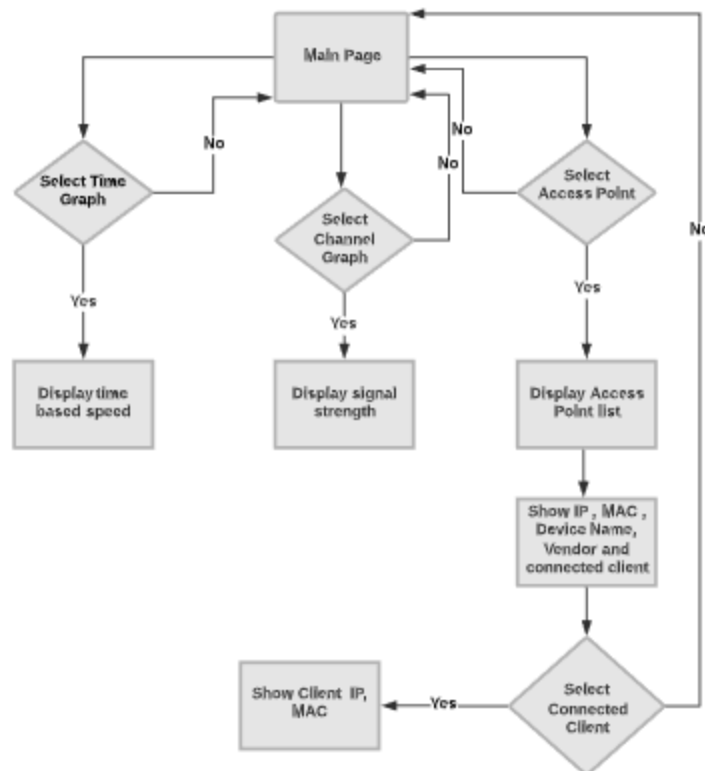
3.2.1 Design Perangkat Keras

Pada bagian ini dijelaskan bagaimana gambaran rangkaian sistem yang akan dikembangkan dari sistem yang sudah ada sebelumnya.

3.2.2 Design Perangkat Lunak

Perancangan Perangkat lunak juga termasuk kedalam bagian perancangan sistem. Pada pengerjaan *User Tracking with Wi-Fi Access Point* terdapat *Flowchart*. *Flowchart* merupakan bagan yang menggambarkan urutan proses dalam suatu program. Berikut gambar *flowchart* perancangan perangkat lunak .

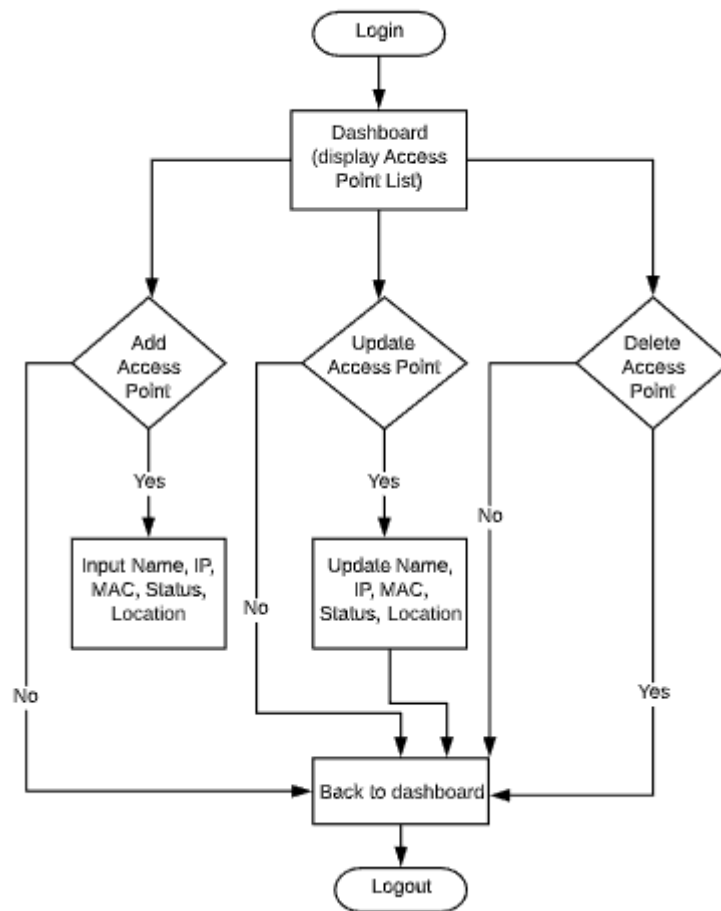
3.3.3 Flowchat Android



Gambar 12 Flowchart Android

Gambar 12 Menjelaskan proses menjalankan *android* yang akan langsung menampilkan halaman utama yang berisi pilihan *access point* mana yang ingin dilihat oleh admin, saat admin mengklik *access point* maka admin akan dialihkan ke tampilan tabel yang berisi nama *user* yang terkoneksi, IP Address dan MAC address *user*. Admin juga dapat melihat data Access Point seperti IP Address, MAC Address, Serta Vendor dari Access Point.

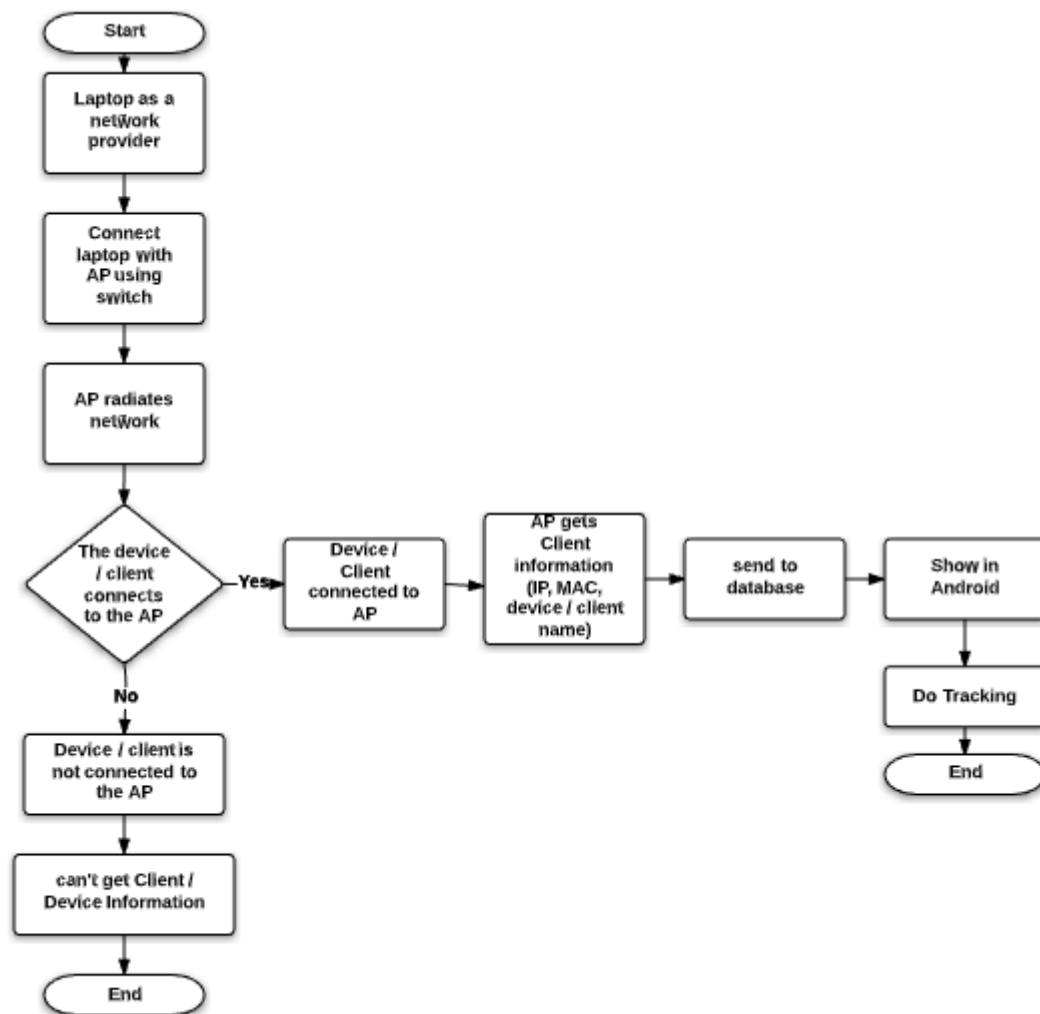
3.3.4 Flowchart web server



Gambar 13 Flowchart Web Server

Gambar 12. Menjelaskan proses menjalankan web server yang diawali dengan *login* dapat menggunakan *username* dan *password* yang sudah didaftarkan terlebih dahulu, lalu melakukan *login*, *user* yang memiliki hak akses pada website akan dialihkan ketampilan *dashboard* yang menampilkan *table* daftar *access point* yang sudah ditambahkan di database. Pada web server ini terdapat fitur menambahkan *access point* yang dapat digunakan untuk menambahkan daftar *access point* yang akan di koneksikan. Untuk fitur selanjutnya adalah fitur *update*, fitur ini dapat digunakan untuk meng-*update* informasi tentang *access point* semisalnya ada perubahan seperti IP atau Lokasi. Selanjutnya ada menu *delete* yang dapat digunakan jika saja ada *access point* yang tidak akan digunakan lagi. Fitur terakhir yang ada pada web server ini adalah *logout*, *user* yang memiliki akses pada web server ini, dapat melakukan *logout* jika sudah menyelesaikan urusan dengan web server ini.

3.3.5 Flowchart System

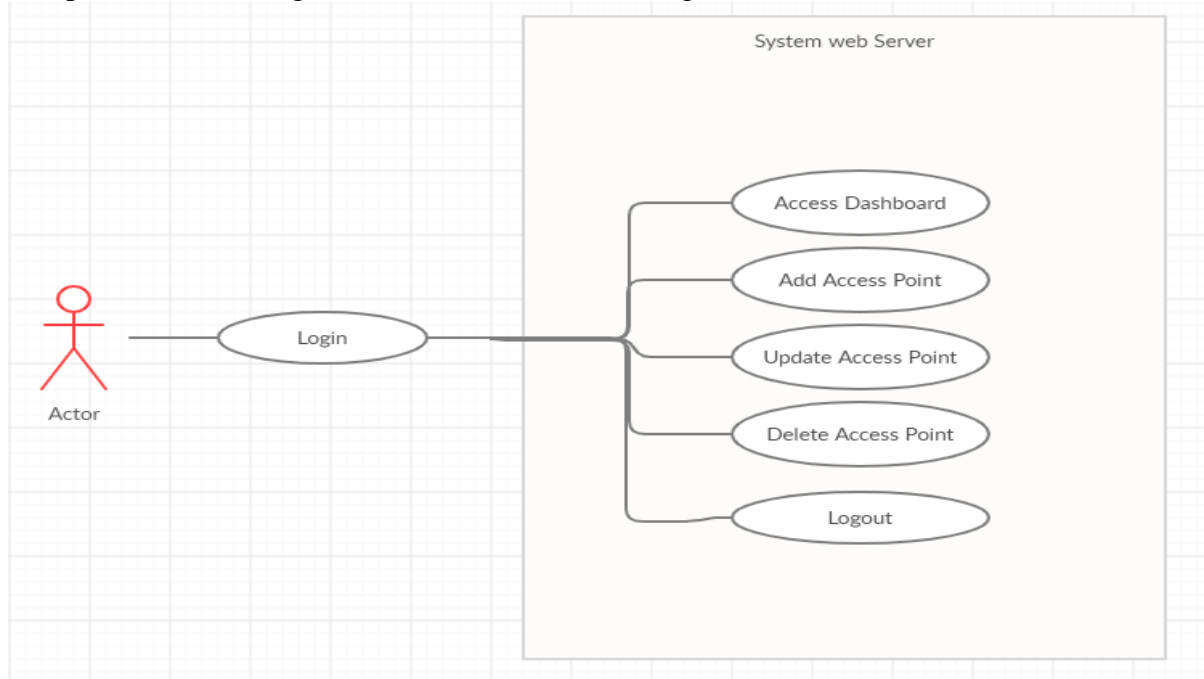


Gambar 13. Flowchart System

Gambar 13 Menjelaskan proses mengaktifkan sistem yang diawali dengan mengaktifkan laptop yang digunakan sebagai penyokong jaringan. Switch digunakan untuk menghubungkan laptop dengan *access point* (*Access Point* berfungsi sebagai pemancar jaringan). Selanjutnya, *device/client* akan melakukan koneksi dengan AP, setelah terkoneksi maka informasi seperti IP, MAC address, dan *device /client* name akan didapat oleh access point lalu dikirimkan ke database sebagai tempat penyimpanan data tersebut. Lalu android akan mengambil data tersebut dan menampilkannya di *dashborad* Android.

3.3.6 Use Case Diagram Web Server

Tampilan use case diagram web server User Tracking with Wi-Fi Access Point

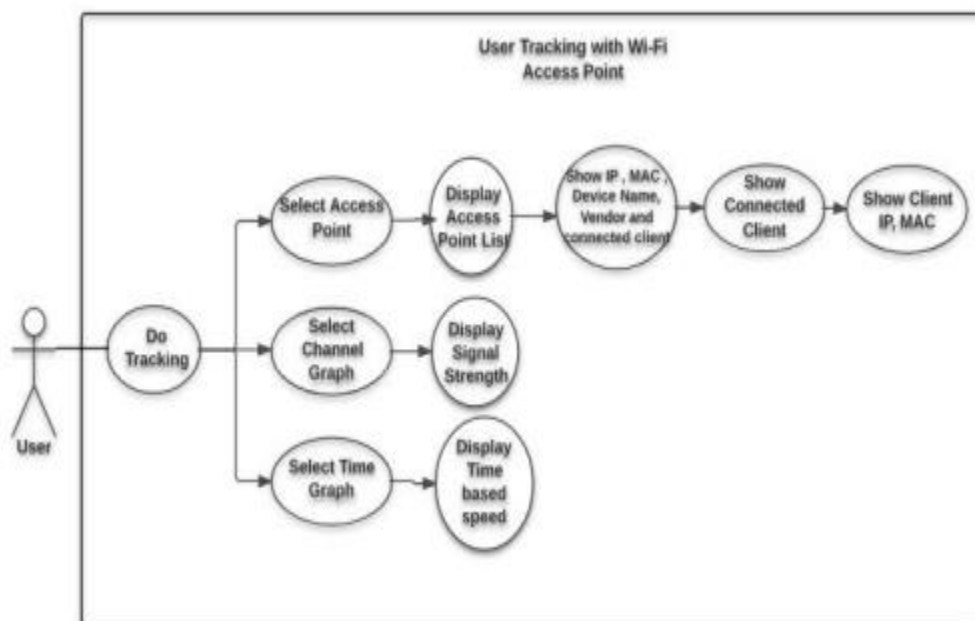


Gambar 14 Use Case Diagram Web Server

Dari gambar diatas dapat dilihat bahwa user yang memiliki hak akses pada web server akan dapat mengakses semua menu yang terdapat pada web server seperti menu menambahkan Access Point, melakukan update saat ada data yang akan diubah pada access point, menghapus data access point yang sudah tidak digunakan lagi dan logout saat user sudah selesai menggunakan web server.

3.3.7 Use Case Diagram Android

Tampilan use case diagram Android User Tracking with Wi-Fi Access Point



Gambar 15 Use Case Diagram User Tracking with Wi-Fi Access Point

Dari gambar diatas dapat dilihat bahwa user yang akan memiliki hak akses penuh pada Android dapat menggunakan Fungsi – Fungsi yang disediakan pada Android , seperti:

1. *User* dapat memilih *access point* yang terdaftar pada *database*
2. *User* dapat menampilkan *list Access Point*
3. *User* dapat melihat IP, MAC Address, Nama *Device*, Vendor dari *Access Point*
4. *User* dapat melihat perangkat-perangkat yang terkoneksi
5. *User* dapat melihat IP, MAC Address dari perangkat-perangkat yang terkoneksi
6. *User* dapat melihat melihat grafik dan menampilkan kekuatan dari signal *Access Point*

3.3.8 Use Case Tracking Skenario

Use Case Id	UCD1	
Use Case Name	<i>Tracking</i>	
Brief Description	<i>Use case ini menjelaskan aktivitas tracking</i>	
Actor	<i>Admin</i>	
Precondition	<i>Admin</i>	
Included Use Case	-	
Basic Flow of Events	<i>Admin Action</i>	<i>System response</i>
		1. Sistem menampilkan informasi (ip, mac, device name) client yang terkoneksi dengan <i>Access Point</i>
	2.Memilih menu yang ingin diakses user	
		3. Menampilkan menu yang dipilih user
Error Flow	<i>Admin Action</i>	<i>System Response</i>
		1. Sistem tidak menampilkan informasi (ip, mac, device name) client yang terkoneksi dengan <i>Access Point</i>
	2.Memeriksa apakah access point berjalan dengan baik,	

		3. Sistem menampilkan informasi (ip, mac, device name) client yang terkoneksi dengan <i>Access Point</i>
Post Condition	Kembali kehalaman dashboard	

3.2 Design

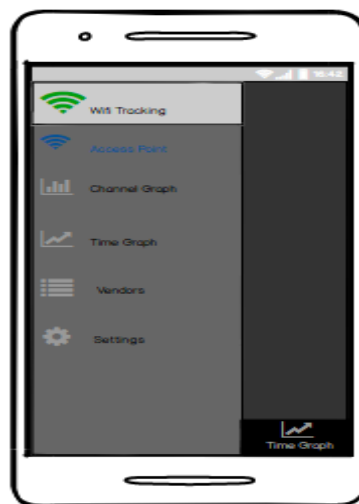
Pada subbab ini akan digambarkan rancangan antarmuka pengguna yang akan di bangun dalam bentuk tampilan web dan Android pada *User Tracking with Wi-Fi Access Point*.

3.2.1 Desain Tampilan Android

Pada bagian ini akan dijelaskan bentuk tampilan dari android untuk sistem yang dibuat

3.2.1.1 Halaman Dashboard

Seperti yang dapat dilihat pada gambar dibawah, bagian ini akan menampilkan menu seperti list *Access Point*, *channel graph*, *time graph*, *vendors*, dan *setting*. *User* yang mengelola Android nantinya dapat menggunakan semua menu yang terdapat pada android



Gambar 16 Halaman Dashboard

3.2.2.2 Halaman Channel Graph

Seperti yang dapat dilihat pada gambar 15, pada menu ini *user* dapat melihat grafik yang menampilkan grafik kecepatan *access point*, sehingga *user* yang mengelolah android ini nantinya dapat mengetahui seberapa cepat atau seberapa lambat kecepatan pada tiap *access point*.



Gambar 17 Halaman Channel Graph

3.2.2.3 Halaman Time Graph

Seperti yang dapat dilihat pada tampilan ini *user* yang mengelolah android nantinya dapat melihat *graph* yang menampilkan grafik seberapa cepat atau seberapa lambat kecepatan *access point* berdasarkan waktu



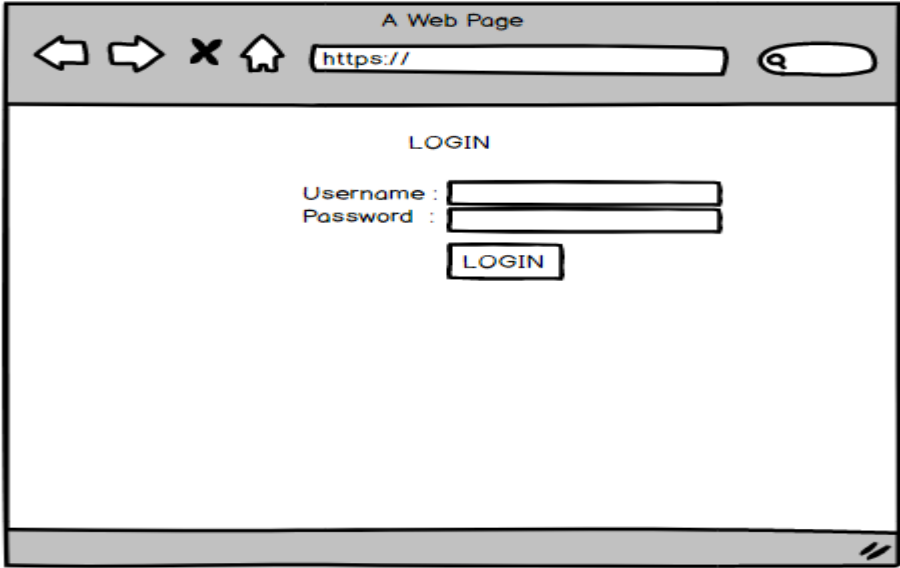
Gambar 18 Halaman Time Graph

3.2.3 Desain Tampilan Web

Pada bagian ini akan dijelaskan bentuk tampilan dari android untuk sistem yang dibuat

3.2.2.1 Halaman Login

Halaman *Login* digunakan untuk menjaga keamanan informasi pada web. Aplikasi ini dibuat menggunakan *login* untuk menjaga keamanan data-data *access point* yang terdaftar dan hanya *user* yang memiliki *username* dan *password* yang terdaftar yang dapat melakukan *login* pada web dan melakukan penambahan *access point*.



A Web Page

https://

LOGIN

Username :

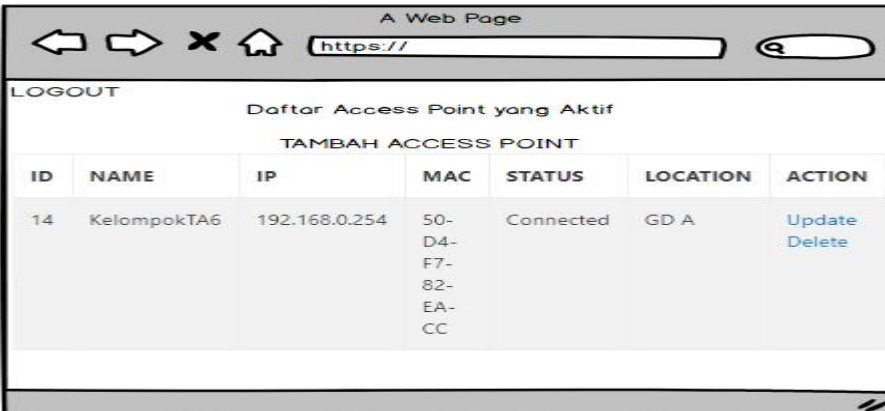
Password :

LOGIN

Gambar 19 Halaman Login web

3.2.2.2 Halaman Dashboard

Seperti yang dapat dilihat pada gambar, pada bagian ini akan menampilkan daftar *Access Point* yang ada. *User* dapat melakukan penambahan *Access Point*, mengedit dan menghapus *Access Point*.



A Web Page

https://

LOGOUT

Daftar Access Point yang Aktif

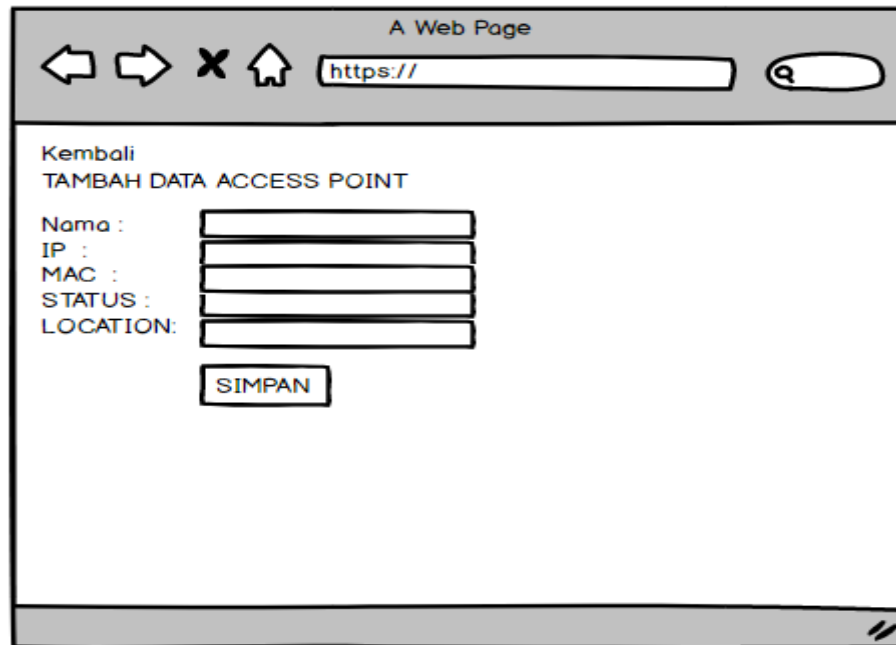
TAMBAH ACCESS POINT

ID	NAME	IP	MAC	STATUS	LOCATION	ACTION
14	KelompokTA6	192.168.0.254	50-D4-F7-82-EA-CC	Connected	GD A	Update Delete

Gambar 20 Halaman dashboard web

3.2.2.3 Tampilan Tambah *Access Point*

Seperti yang dapat dilihat pada bagian ini *user* dapat menambahkan *Access Point* dengan memasukkan data-data *Access Point* yang ingin ditambahkan seperti Nama, IP Address, MAC Address, status serta lokasi dari *Access Point*.

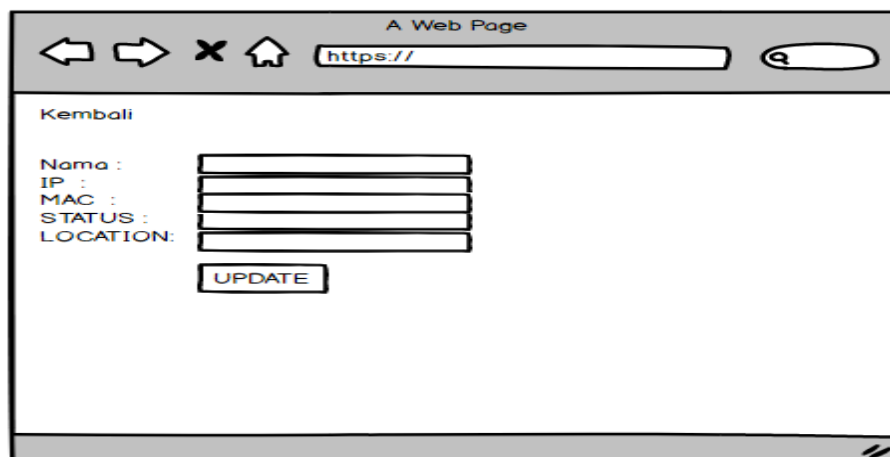


The screenshot shows a web browser window titled "A Web Page". The address bar contains "https://". The main content area displays a form titled "Kembali TAMBAH DATA ACCESS POINT". The form includes five input fields labeled "Nama :", "IP :", "MAC :", "STATUS :", and "LOCATION:". Below these fields is a button labeled "SIMPAN".

Gambar 21 Halaman Tambah Access Point web

3.2.2.4 Tampilan Update

Pada bagian ini *user* dapat melakukan update atau pengeditan pada data *Access Point* seperti mengubah nama access point, IP Address, MAC Address, dan lokasi *Access Point* jika semisalnya ada data-data dari *Access Point* yang perlu diubah.

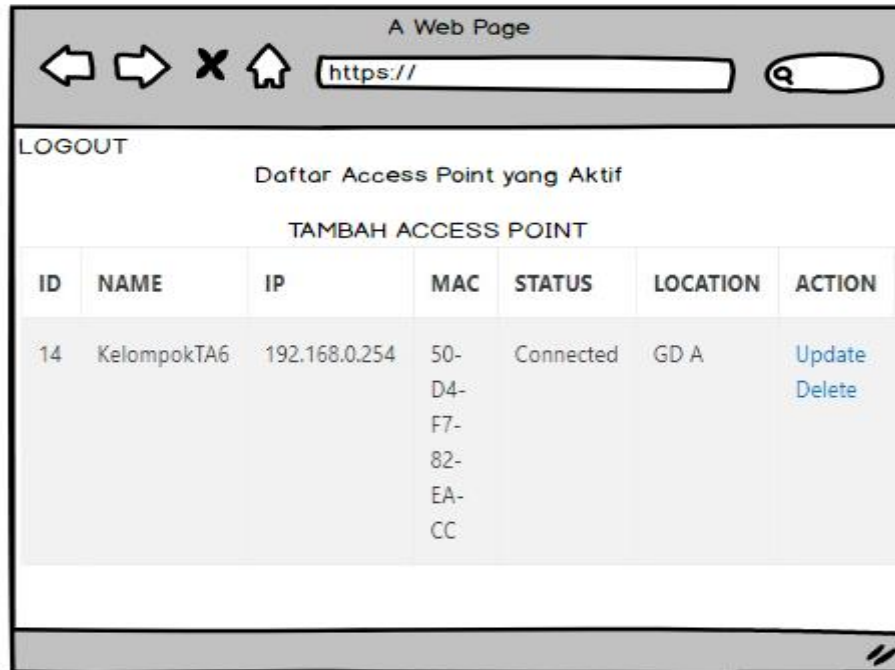


The screenshot shows a web browser window titled "A Web Page". The address bar contains "https://". The main content area displays a form titled "Kembali". The form includes five input fields labeled "Nama :", "IP :", "MAC :", "STATUS :", and "LOCATION:". Below these fields is a button labeled "UPDATE".

Gambar 22 Halaman Update Access Point web

3.2.2.5 Delete

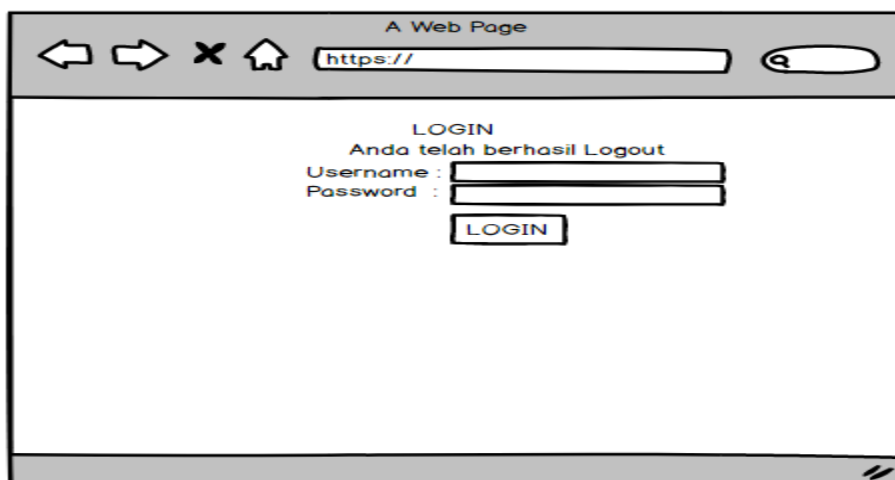
Seperti yang dapat di lihat pada gambar, pada fitur ini user yang mengontrol web server dapat menghapus data *Access Point* yang ada jika semisalnya *Access Point* tersebut tidak digunakan lagi.



Gambar 23 Halaman Delete web

3.2.2.6 Logout

Seperti yang terlihat pada gambar, pada fitur ini user dapat melakukan *logout* saat *user* yang mengontrol web server selesai memonitor *Access Point* melalui web, agar keamanan data-data yang ada pada web server tetap terjaga keamanannya.



Gambar 24 Halaman Logout web

BAB IV

IMPLEMENTASI DAN PENGUJIAN

Pada bab ini menjelaskan mengenai deskripsi umum terkait aplikasi, yang meliputi kebutuhan implementasi, batasan implementasi, implementasi aplikasi dan implementasi fungsi.

4.1 Implementasi Sistem Jaringan

Pada subbab ini dijelaskan mengenai implementasi jaringan yang kami bangun sesuai gambar 11 yang dilakukan dalam pembuatan sistem *User Tracking with Wi-Fi Access Point*. Implementasi dapat dilihat pada gambar berikut.



Gambar 25 Implementasi Sistem Jaringan

Pada gambar diatas terdapat topologi jaringan yang saling terhubung. Dimana terdapat perangkat switch, *access point* dan server. Switch yang digunakan untuk membangun sistem ini adalah switch D-Link DGS-1024C Untuk menghubungkan switch ke access point dan perangkat lainnya menggunakan kabel *Straight*. *Access point* yang digunakan adalah *access point* TL-WA801ND. Dan server yang digunakan pada sistem ini adalah laptop yang berfungsi sebagai penyimpanan data dan konfigurasi antara switch dan *access point*.

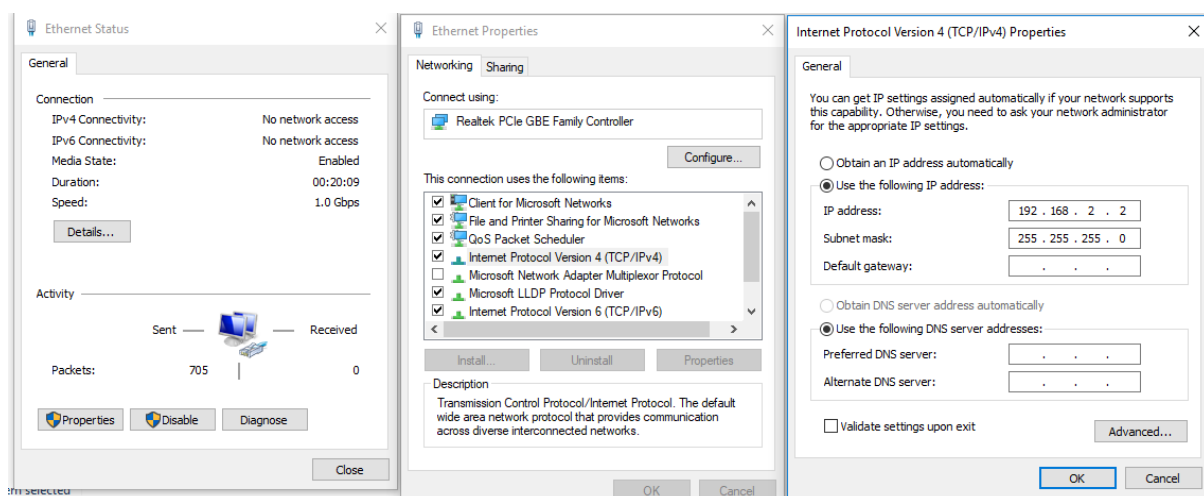
4.1.1 Konfigurasi Server ke switch

Pada sub bab ini dijelaskan mengenai konfigurasi yang dilakukan saat proses pengerjaan sistem Tugas Akhir. Berikut tampilan topologi jaringan yang menghubungkan switch dengan ke server yang di hubungan menggunakan Kabel UTP (*Straight*) dalam jaringan.



Gambar 26 Topologi Server ke switch

Agar kedua perangkat saling terhubung akan dilakukan konfigurasi antara server dan switch. Untuk mengkonfigurasi perangkat tersebut dilakukan di server. Berikut adalah konfigurasi switch di server. Pada gambar IP untuk switch adalah 192.168.2.2 dan Subnet Mask 255.255.255.0



Gambar 27 Konfigurasi Switch di Server

4.1.2 Konfigurasi Server dengan Access Point

Pada subbab ini akan di jelaskan mengenai konfigurasi Access Point ke server yang digunakan pada sistem. IP yang digunakan *developer* di Access point dengan IP Address 192.168.2.3 dengan Subnet Mask 255.255.255.0

4.1.2.1 Konfigurasi Access Point

Pada subbab ini akan dijelaskan mengenai konfigurasi Access point ke server menggunakan nama domain di google chrome 192.168.2.3. Saat konfigurasi Access point developer menggunakan setup static dan memberi security dan Bandwith 2.4 GHz di Access point

The screenshot displays the web interface of a TP-Link Wireless N Access Point WA801ND. The browser address bar shows the IP address 192.168.2.3. The interface has a teal header with the TP-Link logo and the device name. A left sidebar contains navigation links: Status, Quick Setup, Operation Mode, Network, Wireless, DHCP, System Tools, and Logout. The main content area is divided into sections: Status (showing Firmware and Hardware versions), LAN (showing MAC, IP, and Subnet), and Wireless (showing Operation Mode, Radio status, SSID, Mode, Channel, and Width). Below these are 'Wireless Settings' and 'LAN Settings' sections. At the bottom, there are 'Back' and 'Finish' buttons.

Section	Parameter	Value
Status	Firmware Version	0.9.1.3.16 v0001.0 Build 170905 Rel.59579n
	Hardware Version	TL-WA801ND v5 00000005
LAN	MAC Address	50:D4:F7:82:EA:CC
	IP Address	192.168.2.3
	Subnet Mask	255.255.255.0
Wireless	Operation Mode	Access Point
	Wireless Radio	Enabled
	Name(SSID)	TAKelompok6
	Mode	11bgn mixed
	Channel	Auto(Channel 1)
	Channel Width	Auto
	MAC Address	50:D4:F7:82:EA:CC
Wireless Settings		
	Operation Mode	Access Point
	Wireless Channel	Auto
	Wireless Network Name(SSID)	TAKelompok6
	Wireless Security Mode	WPA2-PSK
	Wireless Password	Kelompok6
LAN Settings		
	Default Access	http://tplinkap.net
	LAN Type	Static IP
	IP Address	192.168.2.3

Buttons: Back, Finish

Gambar 28 Konfigurasi Access Point

4. Implementasi Web Server

Pada subbab ini dijelaskan mengenai implementasi yang dilakukan dalam pembuatan web server. Implementasi dapat dilihat pada gambar berikut.

4.2.1 Implementasi Fungsi Menu Login

Berikut adalah *code* yang digunakan untuk membuat menu *login* pada web server. Seperti

yang terlihat pada gambar ada dikatakan *username* dan *password*, *username* dan *password* tersebut diambil dari database yang sudah didaftarkan terlebih dahulu.

```

1 <?php
2 // mengaktifkan session php
3 session_start();
4
5 // menghubungkan dengan koneksi
6 include 'layout/koneksi.php';
7
8 // menangkap data yang dikirim dari form
9 $username = $_POST['username'];
10 $password = $_POST['password'];
11
12 // menyeleksi data admin dengan username dan password yang sesuai
13 $data = mysqli_query($koneksi,"SELECT * from login where username='$username' and password='$password'");
14
15 // menghitung jumlah data yang ditemukan
16 $cek = mysqli_num_rows($data);
17
18 if($cek > 0){
19     $_SESSION['username'] = $username;
20     $_SESSION['status'] = "login";
21     header("location:layout/index.php");
22 }else{
23     header("location:index.php?pesan=gagal");
24 }
25 ?>

```

Gambar 29 Implementasi Fungsi Menu Login

4.2.2 Implementasi Fungsi Menu Tambah Access Point

Berikut adalah *code* yang digunakan untuk menambahkan *access point* ke database, setiap *access point* yang ditambahkan akan langsung masuk dan tersimpan ke database yang sudah dibuat.

```

<?php
// koneksi database
include 'koneksi.php';

// menangkap data yang di kirim dari form
$name = $_POST['nama'];
$ip = $_POST['ip'];
$mac = $_POST['mac'];
$status = $_POST['status'];
$location = $_POST['location'];

// menginput data ke database
mysqli_query($koneksi,"insert into accesspoint values('','$name','$ip','$mac','$status','$location')");

// mengalihkan halaman kembali ke index.php
header("location:index.php");

?>

```

Gambar 30 Implementasi Fungsi Menu Tambah Access Point

4.2.3 Implementasi Fungsi Menu Update

Berikut adalah *code* untuk mengupdate data-data *access point*, dan setelah melakukan *update* data –data tersebut juga akan langsung tersimpan ke dalam database.

```

<?php
// include database connection file
include_once("koneksi.php");

// Check if form is submitted for user update, then redirect to homepage after update
if(isset($_POST['update']))
{
    $id = $_POST['id'];
    $nama = $_POST['nama'];
    $ip = $_POST['ip'];
    $mac = $_POST['mac'];
    $status = $_POST['status'];
    $location = $_POST['location'];

    // update user data
    $result = mysqli_query($koneksi, "UPDATE accesspoint SET nama='$nama',ip='$ip',mac='$mac',status='$status', location='$location' WHERE id=$id");

    // Redirect to homepage to display updated user in list
    header("Location: index.php");
}
}
<?php
// Display selected user data based on id
// Getting id from url
$id = $_GET['id'];

// Fetch user data based on id
$result = mysqli_query($koneksi, "SELECT * FROM accesspoint WHERE id=$id");

while($accesspoint_data = mysqli_fetch_array($result))
{
    $nama = $accesspoint_data['name'];
    $ip = $accesspoint_data['ip'];
    $mac = $accesspoint_data['mac'];
    $status = $accesspoint_data['status'];
    $location = $accesspoint_data['location'];
}
}
<html>
<head>
<title>Update Access Point</title>
</head>

```

Gambar 31 Implementasi Fungsi Menu Update

4.2.4 Implementasi Fungsi Menu Delete

Berikut adalah *code* yang digunakan untuk membuat fungsi *delete* seperti *access point* tidak digunakan lagi maka *user* dapat menghapusnya dan saat sudah dihapus, data *access point* juga akan langsung terhapus pada database.

```

<?php
// include database connection file
include_once("koneksi.php");

// Get id from URL to delete that user
$id = $_GET['id'];

// Delete user row from table based on given id
$result = mysqli_query($koneksi, "DELETE FROM accesspoint WHERE id=$id");

// After delete redirect to Home, so that latest user list will be displayed.
header("Location:index.php");
?>

```

Gambar 32 Implementasi Fungsi Menu Delete

Berikut adalah *code* untuk database, dapat dilihat pada *code* ini bahwa terdapat *code* yang menunjukkan *access point* yang ditambahkan sudah terdapat didalam *code*.


```

CREATE DATABASE /*!32312 IF NOT EXISTS*/`tracking` /*!40100 DEFAULT CHARACTER SET utf8mb4 */;

USE `tracking`;

/*Table structure for table `accesspoint` */

DROP TABLE IF EXISTS `accesspoint`;

CREATE TABLE `accesspoint` (
  `id` int(50) NOT NULL AUTO_INCREMENT,
  `name` varchar(255) NOT NULL,
  `ip` varchar(255) NOT NULL,
  `mac` varchar(255) NOT NULL,
  `status` varchar(255) NOT NULL,
  `location` varchar(255) NOT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=9 DEFAULT CHARSET=latin1;

/*Data for the table `accesspoint` */

insert into `accesspoint`(`id`,`name`,`ip`,`mac`,`status`,`location`) values
(7,'KelompokTA06','192.168.2.3','50:D4:F7:82:EA:CC','Connected','Rumah A'),
(8,'KelompokTA6','192.168.2.4','50:D4:F7:82:EA:CC','Connected','Rumah B');

/*Table structure for table `login` */

DROP TABLE IF EXISTS `login`;

CREATE TABLE `login` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `username` varchar(255) NOT NULL,
  `password` varchar(255) NOT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=2 DEFAULT CHARSET=latin1;

/*Data for the table `login` */

insert into `login`(`id`,`username`,`password`) values
(1,'Admin','Admin');

```

Gambar 33 Code Membaca Access Point

4.2.5 Implementasi Fungsi Menu Logout

Berikut adalah *code* yang digunakan untuk membuat fungsi *logout*, yang akan digunakan saat *user* yang memiliki akses selesai menggunakan web sever.

```

k?php
// mengaktifkan session
session_start();

// menghapus semua session
session_destroy();

// mengalihkan halaman sambil mengirim pesan logout
header("location:../index.php?pesan=logout");
?>

```

Gambar 34 Implementasi Fungsi Menu Logout

4.3 Implentasi Android

Pada bagian ini dijelaskan mengenai implementasi aplikasi *Wi-Fi Tracking* dalam

pengerjaan Tugas Akhir. Aplikasi *Wi-Fi Tracking* ini merupakan aplikasi yang dikembangkan menggunakan android studio dengan bahasa pemrograman java dan penyimpanan data menggunakan SQLite. Aplikasi *Wifi Tracking* menggunakan *gradle* 5.2.1 pada android studio, *gradle* ini salah satu cara yang fleksibel untuk mengompilasi, *mem-build*, dan mengemas aplikasi atau *library Android*. Aplikasi *Wi-Fi Tracking* akan terhubung ke *access point* yang sebelumnya telah dikonfigurasi pada sub bab 4.1 Implementasi Jaringan. Aplikasi *Wi-Fi Tracking* dapat membaca *IP Address*, *MAC Address*, nama *device*, membaca frekuensi *access point*, membaca kecepatan *access point*, menampilkan grafik *access point*, menampilkan kecepatan grafik *access point* perwaktu, dan menghitung jumlah *user*. Aplikasi *Wi-Fi tracking* akan menyimpan data dengan database SQLite yang disinkronkan juga dengan database *MySQL* yang terdapat pada web server subbab 4.2 Implementasi *Web Server*.

4.3.1 Implementasi Navigation menu

Pada bagian navigation menu terdapat tiga menu utama dan satu tambahan yaitu *Access point*, *Channel Graph*, *Time Graph*, dan *vendors*. Pada menu *access point* berfungsi untuk membaca *IP address*, *MAC address*, nama *device*, frekuensi *access point*, dan Kecepatan *access point*. Pada menu *Channel Graph* berfungsi untuk menampilkan grafik kecepatan *access point*. Pada menu *Time Graph* berfungsi untuk menampilkan grafik sesuai kecepatan *access point* per waktu. Pada menu *vendors* hanya sebagai menu pendukung untuk membaca *vendors technology* yang digunakan *device* yang terhubung pada *access point*. Gambaran *code* yang digunakan pada *navigation* menu dapat dilihat pada gambar 32.

```
public enum NavigationMenu {
    ACCESS_POINTS(R.drawable.ic_network_wifi, "Access Points", NavigationItemFactory.ACCESS_POINTS, NavigationOptionFactory.AP),
    CHANNEL_GRAPH(R.drawable.ic_insert_chart, "Channel Graph", NavigationItemFactory.CHANNEL_GRAPH, NavigationOptionFactory.OTHER),
    TIME_GRAPH(R.drawable.ic_show_chart, "Time Graph", NavigationItemFactory.TIME_GRAPH, NavigationOptionFactory.OTHER),
    VENDORS(R.drawable.ic_list_grey, R.string.action_vendors, NavigationItemFactory.VENDORS),
    SETTINGS(R.drawable.ic_settings, "Settings", NavigationItemFactory.SETTINGS);

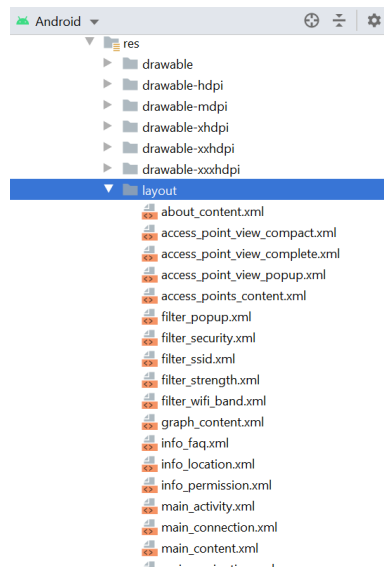
    private final int icon;
    private final int title;
    private final List<NavigationOption> navigationOptions;
    private final NavigationItem navigationItem;

    NavigationMenu(int icon, int title, @NonNull NavigationItem navigationItem, @NonNull List<NavigationOption> navigationOptions) {
        this.icon = icon;
        this.title = title;
        this.navigationItem = navigationItem;
        this.navigationOptions = navigationOptions;
    }
}
```

Gambar 35 Navigation Menu

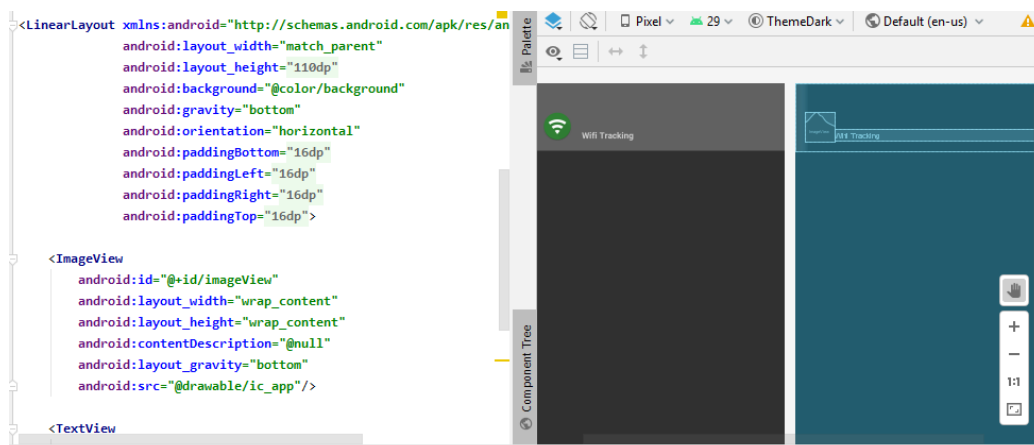
4.3.2 Implementasi Layout

Pada bagian *layout* merupakan penataan letak desain aplikasi *Wi-Fi tracking*. Letak desain yang dimaksud seperti yang terdapat pada menu *access point* dan menu grafik, penataan letak ini akan memadukan unsur-unsur komunikasi grafis seperti warna, teks, dan gambar tertata dengan baik. Penataan layout dan salah satu gambaran layout pada Aplikasi *Wi-Fi Tracking* dapat dilihat pada gambar 33.



Gambar 36 Peletakan Layout dan Gambar Layout

Gambaran *layout* dibawah ini merupakan salah satu *layout* beserta *code* yang digunakan untuk desain aplikasi *Wi-Fi tracking*. Pada *layout* dibawah ini terdapat dua cara untuk mendesain yaitu pertama *drag* langsung sesuai yang ingin dirancang dan kedua dapat langsung *code* pada *file*.

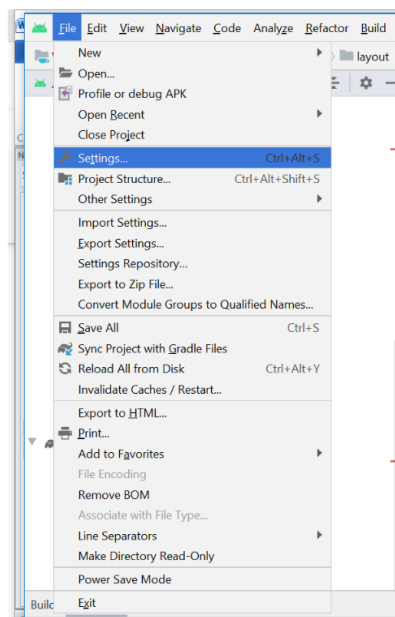


Gambar 37 Desain dan Code Layout

4.3.3 Implementasi Gradle

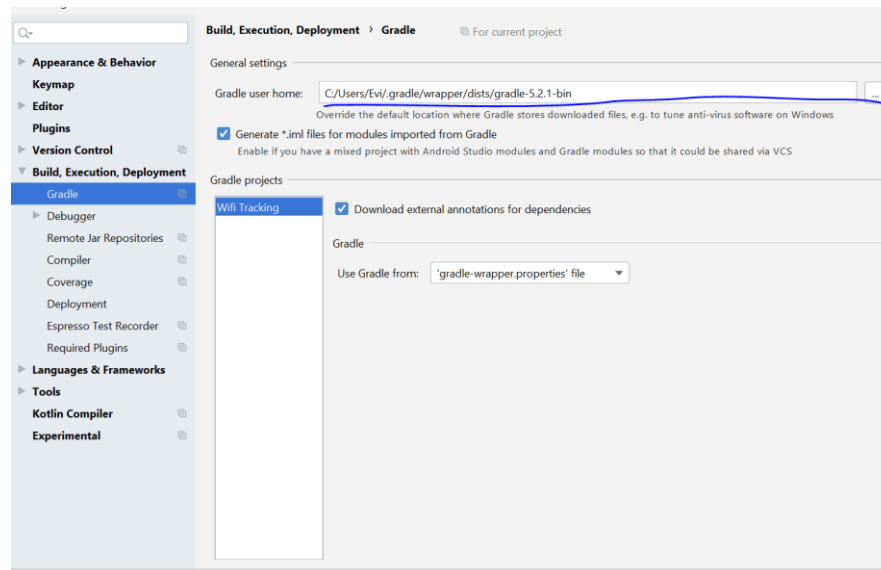
Pada bagian *gradle*, peletakan folder *gradle* dan *code gradle* sangat berpengaruh saat menjalankan aplikasi dan harus disesuaikan dengan aplikasi yang dikembangkan. *Gradle* berfungsi untuk mengompilasi, mem-*build*, dan mengemas aplikasi atau *library* Android sehingga dapat mengoptimalkan Kecepatan Build aplikasi *Wi-Fi tracking*. Langkah-langkah peletakan folder *gradle* adalah sebagai berikut.

1. Pada *toolbar* android arahkan kursor pada menu *file*, lalu klik *setting*. Seperti gambar 35 berikut.



Gambar 38 Langkah Membuka Folder Tempat Gradle

2. Lalu, akan muncul letak folder seperti gambar dibawah ini. Sesuaikan letak folder *gradle* dan versi *gradle*, versi *gradle* yang digunakan aplikasi *Wi-Fi tracking* adalah *gradle* 5.2.1.



Gambar 39 Penyesuaian Tempat Folder Gradle

Untuk *code gradle* terdapat versi android yang dapat menggunakan aplikasi *Wi-Fi tracking*, properti modul *project* untuk versi aplikasi, dan melakukan *plugin* android. Gambaran *code* dapat dilihat pada gambar 37.

```
android {
    compileSdkVersion 29
    buildToolsVersion "29.0.2"

    defaultConfig {
        applicationId "com.vrem.wifianalyzer"
        minSdkVersion 19
        targetSdkVersion 29
        versionCode
        versionName
        testInstrumentationRunner "androidx.test.runner.AndroidJUnitRunner"
    }

    buildTypes {
        release {
            minifyEnabled true
            shrinkResources true
            proguardFiles getDefaultProguardFile("proguard-android.txt")
            signingConfig
        }
        debug {
```

Gambar 40 Code Gradle

4.3.4 Implementasi Sqlite

Pada bagian ini menjelaskan penyimpanan data yang digunakan aplikasi yaitu database SQLite, database SQLite menyimpan data seperti *IP address*, *MAC address*, dan nama *device* dari *access point*. Pada database SQLite hal yang dilakukan perlu memanggil *library* SQLite *helper* untuk menerapkan metode yang akan membuat serta mengelola database dan tabel. Nama database dan *field* tabel yang akan digunakan dapat disesuaikan sesuai kebutuhan Gambaran pemanggilan *library* SQLite dan nama database dapat dilihat pada

gambar 38 dibawah ini.

```
import android.content.Context;
import android.content.DialogInterface;
import android.content.DialogInterface.OnClickListener;
import android.content.Intent;
import android.database.Cursor;
import android.database.SQLException;
import android.database.sqlite.SQLiteDatabase;
import android.database.sqlite.SQLiteException;
import android.database.sqlite.SQLiteOpenHelper;
import android.net.DhcpInfo;
import android.net.wifi.ScanResult;
import android.net.wifi.WifiInfo;
import android.net.wifi.WifiManager;
import android.os.Build.VERSION;
import android.os.Bundle;
import android.text.Html;

public class MainActivity extends Activity {
    static DBHelper BD = null;
    static DBHelper BD2 = null;
    static DBHelper BD3 = null;
    static String DB_NAME = "bd";
    static String DB_NAME2 = "bdaccesspoint";
    static String DB_NAME3 = "bdvendor";
    static String FirstNumberIP = "";
    static int IdDispositivoEscogido = 0;
    static final String KEY_COL1 = "field1";
    static final String KEY_COL2 = "field2";
    static final String KEY_ID = "_id";
    static boolean Killed;
    static String LastNumberIP = "";
    static String LinkSpeed = "";
```

Gambar 41 Pemanggilan library sqllite helper

4.3.5 Implementasi Menu pada Aplikasi

Pada aplikasi ada tiga menu utama yang akan dikembangkan yaitu membaca *access point*, menampilkan grafik *access point* dan menampilkan grafik kecepatan *access point* perwaktu. Ketiga fungsi ini memiliki tiga *code* utama tersendiri. Pada menu membaca Access point dapat membaca *ip address*, *MAC address*, nama *device*, frekuensi *access point*, kecepatan, dan *vendor technology*. Pada saat melihat detail *access point* pada menu *access point* maka dapat melihat *user* yang terhubung. Pada menu menampilkan grafik *access point* dapat memperlihatkan grafik *access point*. Pada menu menampilkan grafik kecepatan *access point* perwaktu dapat berubah grafik sesuai kecepatan *access point* perwaktu . Gambaran setiap *code* untuk membaca *access point*, menampilkan grafik *access point*, dan menampilkan grafik kecepatan *access point* perwaktu dapat dilihat pada gambar 38, gambar 39, dan gambar 40 dibawah ini .

```

37 public WiFiConnection(@NonNull String SSID, @NonNull String BSSID, @NonNull String ipAddress, int linkSpeed) {
38     this.SSID = SSID;
39     this.BSSID = BSSID;
40     this.ipAddress = ipAddress;
41     this.linkSpeed = linkSpeed;
42 }
43
44 @NonNull
45 public String getSSID() { return SSID; }
46
47 @NonNull
48 public String getBSSID() { return BSSID; }
49
50 @NonNull
51 public String getIpAddress() { return ipAddress; }
52
53 public int getLinkSpeed() { return linkSpeed; }
54
55 public boolean isConnected() { return !EMPTY.equals(this); }
56

```

Gambar 42 Code membaca access point

```

47 @NonNull
48 DataPoint[] getDataPoints(@NonNull WiFiDetail wiFiDetail, int levelMax) {
49     WiFiSignal wiFiSignal = wiFiDetail.getWiFiSignal();
50     int frequencyStart = wiFiSignal.getFrequencyStart();
51     int frequencyEnd = wiFiSignal.getFrequencyEnd();
52     int level = Math.min(wiFiSignal.getLevel(), levelMax);
53     return new DataPoint[]{
54         new DataPoint(frequencyStart, GraphConstants.MIN_Y),
55         new DataPoint(frequencyStart + WiFiChannels.FREQUENCY_SPREAD, level),
56         new DataPoint(wiFiSignal.getCenterFrequency(), level),
57         new DataPoint(frequencyEnd - WiFiChannels.FREQUENCY_SPREAD, level),
58         new DataPoint(frequencyEnd, GraphConstants.MIN_Y)
59     };
60 }
61
62 void addSeriesData(@NonNull GraphViewWrapper graphViewWrapper, @NonNull Set<WiFiDetail> wiFiDetails, int levelMax) {
63     IterableUtils.forEach(wiFiDetails, new SeriesClosure(graphViewWrapper, levelMax));
64 }

```

Gambar 43 Code Menampilkan Grafik Access Point

```

43 this.scanCount = 0;
44 this.xValue = 0;
45 this.timeGraphCache = new TimeGraphCache();
46 }
47
48 @NonNull
49 Set<WiFiDetail> addSeriesData(@NonNull GraphViewWrapper graphViewWrapper, @NonNull List<WiFiDetail> wiFiDetails) {
50     Set<WiFiDetail> inOrder = new TreeSet<>(wiFiDetails);
51     IterableUtils.forEach(inOrder, new AddDataClosure(graphViewWrapper, levelMax));
52     adjustData(graphViewWrapper, inOrder);
53     Set<WiFiDetail> result = getNewSeries(inOrder);
54     xValue++;
55     if (scanCount < GraphConstants.MAX_SCAN_COUNT) {
56         scanCount++;
57     }
58     if (scanCount == 2) {
59         graphViewWrapper.setHorizontalLabelsVisible(true);
60     }
61     return result;
62 }
63
64 void adjustData(@NonNull GraphViewWrapper graphViewWrapper, @NonNull Set<WiFiDetail> wiFiDetails) {

```

Gambar 44 Code Menampilkan Grafik Access Point Perwaktu

4.4 Implementasi Menghubungkan database MySQL dengan SQLite

Pada subbab ini akan menjelaskan pengimplementasian menghubungkan database MySQL dengan database SQLite. Adapun data yang dapat saling terhubung pada database masing-masing adalah seperti *IP address access point*, *MAC Address*, nama *device*. Gambaran *code* untuk menghubungkan database MySQL dengan database SQLite terdapat pada gambar

```

154         final String name = editTextName.getText().toString().trim();
155
156         StringRequest stringRequest = new StringRequest(Request.Method.POST, URL_SAVE_NAME,
157             new Response.Listener<String>() {
158                 @Override
159                 public void onResponse(String response) {
160                     progressDialog.dismiss();
161                     try {
162                         JSONObject obj = new JSONObject(response);
163                         if (!obj.getBoolean("error")) {
164                             //if there is a success
165                             //storing the name to sqlite with status syncd
166                             saveNameToLocalStorage(name, NAME_SYNCED_WITH_SERVER);
167                         } else {
168                             //if there is some error
169                             //saving the name to sqlite with status unsyncd
170                             saveNameToLocalStorage(name, NAME_NOT_SYNCED_WITH_SERVER);
171                         }
172                     } catch (JSONException e) {
173                         e.printStackTrace();
174                     }
175                 }
176             },

```

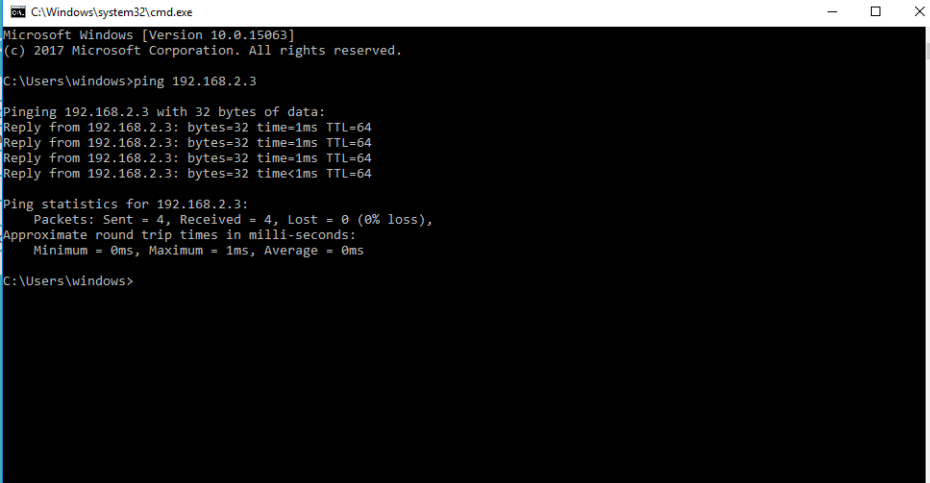
Gambar 45 Code menghubungkan database MySQL dengan SQLite

4.5 Pengujian

Pada subbab ini akan dijelaskan tentang pengujian terhadap implementasi yang telah dilakukan untuk melihat dan memastikan bahwa perangkat yang dibangun dapat berjalan dengan baik. Pengujian yang dilakukan yaitu skenario pengujian terhadap Sistem jaringan , web server dan android.

4.5.1 Pengujian Sistem Jaringan

Pada subbab ini akan dijelaskan tentang pengujian konfigurasi jaringan yang telah terkoneksi. Untuk pengujian *access point* maka dilakukan ping di server berikut adalah tampilan pengujian *IP address access point* A yang telah terkoneksi ke perangkat lainnya dengan perintah ping 192.168.2.3.



```

C:\Windows\system32\cmd.exe
Microsoft Windows [Version 10.0.15063]
(c) 2017 Microsoft Corporation. All rights reserved.

C:\Users\windows>ping 192.168.2.3

Pinging 192.168.2.3 with 32 bytes of data:
Reply from 192.168.2.3: bytes=32 time=1ms TTL=64
Reply from 192.168.2.3: bytes=32 time=1ms TTL=64
Reply from 192.168.2.3: bytes=32 time=1ms TTL=64
Reply from 192.168.2.3: bytes=32 time<1ms TTL=64

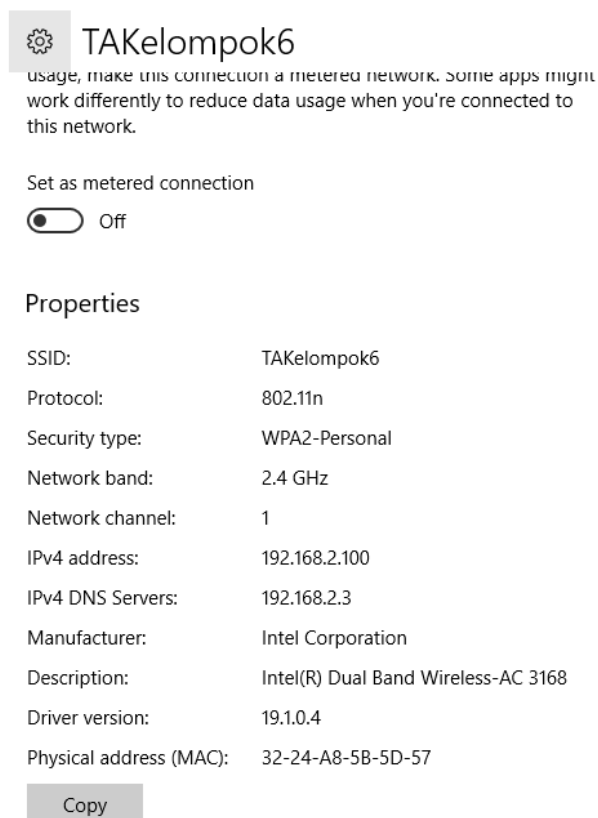
Ping statistics for 192.168.2.3:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 0ms, Maximum = 1ms, Average = 0ms

C:\Users\windows>

```

Gambar 46 Pengujian IP Access Point pada Server

Pada pengujian tersebut kita dapat melihat status jaringan terkoneksi dengan perangkat Server. Berikut ini tampilan pengujian *access Point* yang telah terkoneksi.



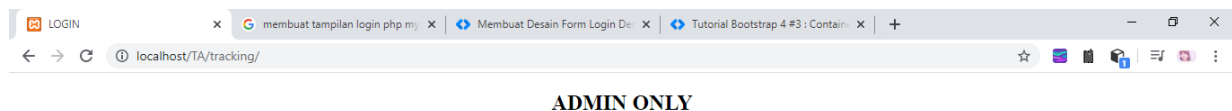
Gambar 47 Pengujian Access Point Terkoneksi

4.5.2 Pengujian Web server

Pada bagian subbab ini akan dijelaskan mengenai pengujian dalam fungsi-fungsi yang terdapat dalam web server yang telah dibuat.

4.5.2.1 Pengujian Fungsi Menu Login

Berikut merupakan tampilan *login* web server yang dibuat untuk *Wi-Fi Tracking*. Pada bagian ini *user* yang bertanggung jawab untuk mengelolah web server harus melakukan *login* terlebih dahulu untuk dapat mengakses data-data yang akan ditampilkan web server.



Gambar 48 Menu Login

4.5.2.2 Pengujian Fungsi Menu Tambah Access Point

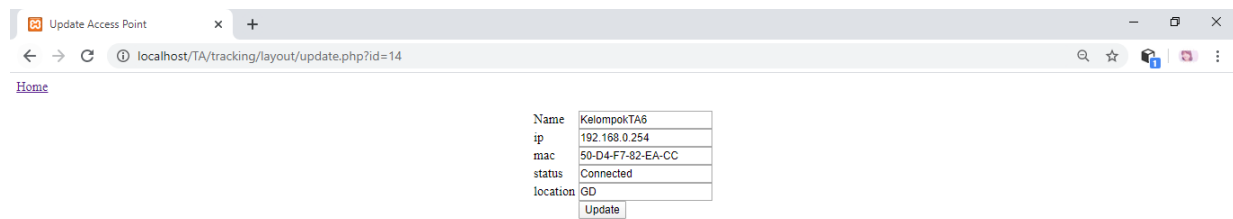
Berikut merupakan fungsi tambah Access Point pada web server yang dibuat untuk *Wi-Fi Tracking*. User yang memiliki hak akses pada web server dapat menambahkan access point yang akan dikoneksikan, untuk menambahkan *access point* tersebut *user* terlebih dahulu harus memasukkan data-data *access point* seperti nama *access point*, *IP address* dan *MAC address*, status *access point* dan lokasi *access point*.



Gambar 49 Menu Tambah Access Point

4.5.2.3 Pengujian Fungsi Menu Update

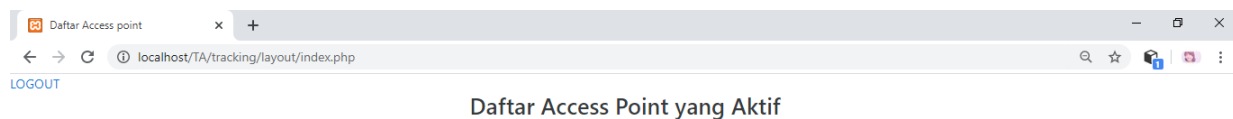
Berikut merupakan fungsi *update Access Point* pada web server yang dibuat untuk *Wi-Fi Tracking*. *User* yang memiliki hak akses pada web server dapat mengubah data-data *access point* jika ada perubahan pada data-data *access point* tersebut melalui fungsi *update* ini.



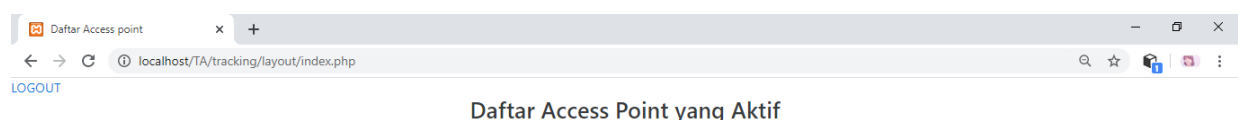
Gambar 50 Menu Update

4.5.2.4 Pengujian Fungsi Menu Delete

Berikut merupakan fungsi *delete Access Point* pada web server yang dibuat untuk *Wi-Fi Tracking*. User yang memiliki hak akses pada web server ini dapat menghapus *access point* dari daftar *access point* jika saja *access point* tersebut tidak digunakan lagi melalui fungsi delete ini.



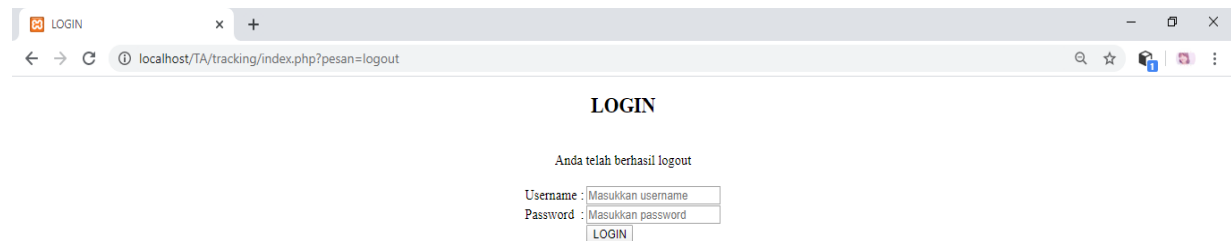
Begini adalah tampilan *dashboard* setelah melakukan *delete* pada *access point* yang sudah tidak digunakan lagi.



Gambar 51 Menu Delete

4.5.2.1 Pengujian Fungsi Menu Logout

Berikut merupakan fungsi *logout* pada web server yang dibuat untuk *Wi-Fi Tracking*, *User* yang sudah selesai mengakses web server dapat melakukan *logout* agar semua data-data yang ada di web server tetap terjaga keamanannya.



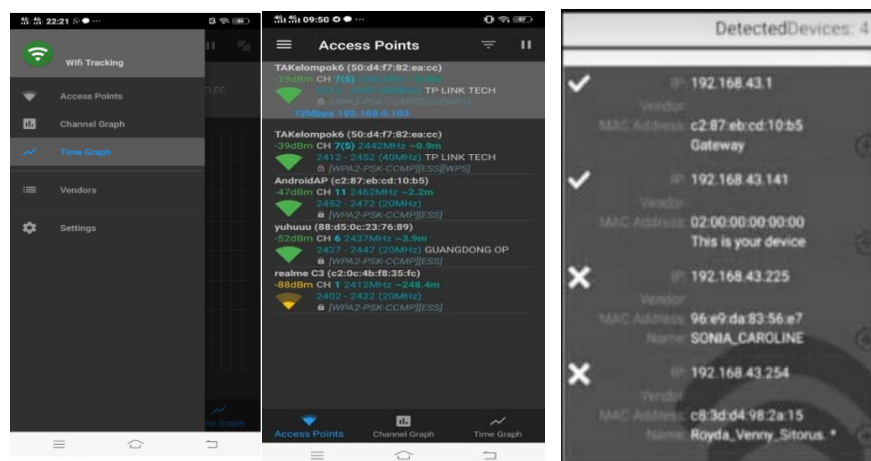
Gambar 52 Menu Logout

4.5.3 Pengujian Android

Pada bagian subbab ini akan dijelaskan mengenai pengujian aplikasi, pengujian akan dilakukan pada menu *access point*, menu *channel graph*, menu *time graph* yang terdapat pada aplikasi *wifi tracking*

4.5.3.1 Pengujian Menu access point

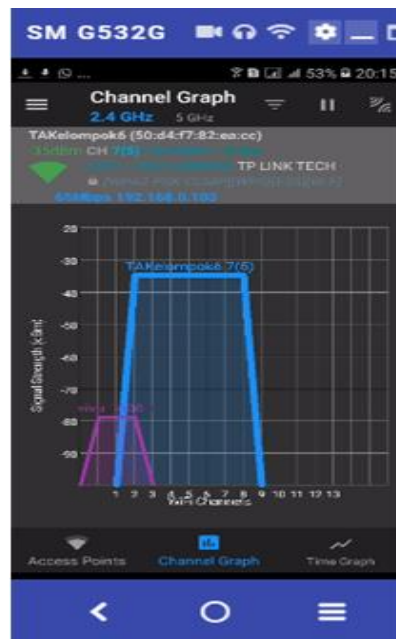
Pada menu *access point* dapat membaca *IP address access point* 192.168.0.103, *MAC address* 50:d4:f7:82:ea:cc, nama *device* TAKelompok6, *vendor technology* yang digunakan *TP LINK TECH*, frekuensi 40MHz, dan kecepatan 72Mbps. Ketika melihat detail *access point* kita dapat melihat *user* yang terhubung pada *access point* yaitu Nama *device* Sonia_Caroline, Royda_Venny_Sitorus, *s address* 192.168.43.225, 192.168.43.254, *MAC Address* 96:e9:da:83:56:e7, c8:3d:d4:98:2a:15. Gambaran menu *access point* dapat kita lihat pada gambar 48 dibawah ini.



Gambar 53 Menu Access Point

4.5.3.2 Pengujian Menu Channel Graph

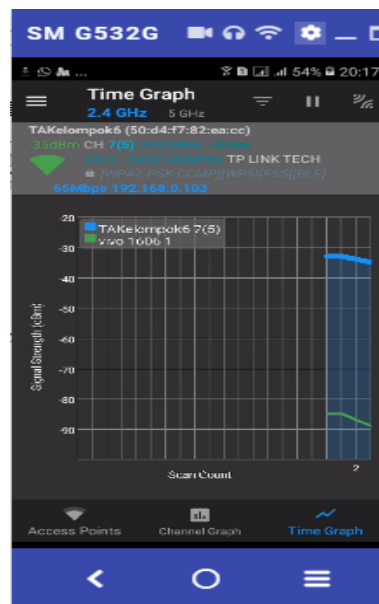
Pada menu *channel graph* akan menampilkan grafik *access point*, grafik *access point* pada menu aplikasi *Wi-Fi tracking* berbentuk grafik batang. Pada menu ini terbaca juga *IP address access point* 192.168.0.103, *MAC address* 50:d4:f7:82:ea:cc, nama *device* TAKelompok6, *vendor technology* yang digunakan *TP LINK TECH*, frekuensi 40MHz, dan kecepatan 72Mbps. Pada grafik terlihat *Bandwidth* yang digunakan *access point* antara 30 dengan 40 dBm. Gambaran menu *channel graph* dapat dilihat pada gambar 49 dibawah ini.



Gambar 54 Menu Channel Graph

4.5.3.3 Pengujian Menu Time Graph

Pada menu *time graph* dapat menampilkan grafik kecepatan *access point* perwaktu, grafik *access point* pada menu aplikasi *Wi-Fi tracking* berbentuk grafik kurva. Pada menu ini terbaca juga *IP address access point* 192.168.0.103, *MAC address* 50:d4:f7:82:ea:cc, nama *device* TAKelompok6, *vendor technology* yang digunakan *TP LINK TECH*, frekuensi 40MHz, dan kecepatan 72Mbps. Pada kurva terlihat *throughput* yang digunakan *access point* antara 30 dengan 40 dBm, setelah dilakukan *scan* jumlah pengguna pada *access point*. Gambaran menu *time graph* dapat dilihat pada gambar 50 dibawah ini.



Gambar 55 Menu Time Graph

BAB V

KESIMPULAN DAN SARAN

5.1 Kesimpulan

Setelah melakukan pengerjaan pada sistem *User Tracking with Wi-Fi Access Point* dapat diperoleh kesimpulan yaitu:

1. Sistem jaringan yang dibangun menggunakan perangkat server, Switch dan *Access Point* yang saling terhubung sehingga pengguna dapat mengakses dan menggunakan jaringan Wi-Fi dan sistem yang dibangun akan mendapatkan data pengguna yang memakai jaringan Wi-Fi yang telah dibangun.
2. Perangkat sistem *tracking* yang bertujuan untuk menampilkan jumlah *client* yang terkoneksi dengan *access point*, menampilkan IP address serta MAC address *client* yang terkoneksi. Sistem ini akan menggunakan *access point* sebagai pemancar jaringan yang akan terhubung dengan *client* dan menggunakan *database* yang akan digunakan sebagai tempat penyimpanan data dari *client* yang terkoneksi dan akan di tampilkan pada aplikasi Android.
3. Sistem menghubungkan database MySQL web server dengan database SQLite agar dapat menampilkan dan membaca data *user* yang telah terhubung ke sistem.

5.2 Saran

Adapun saran yang diberikan penulis setelah melakukan pengerjaan Tugas Akhir ini, yaitu:

1. *Access point* perlu melakukan *setting repeater* sewaktu mengkonfigurasi *access point* agar *access point* dapat menggunakan *hotspot* pribadi sebagai sumber pancaran *signal access point* dan akan menghemat biaya karena jika menggunakan *wifi access point* secara langsung perlu melakukan pembayaran perbulan sesuai penggunaan wifi kepada pihak telkom.
2. Aplikasi *Wifi Tracking* dalam pengerjaan tugas akhir ini memiliki banyak kekurangan, untuk pengembangan aplikasi selanjutnya diharapkan dapat menambahkan fungsi yang lebih bermanfaat dan membantu pengguna.
3. Pada saat menghubungkan database MySQL web server dengan database SQLite aplikasi, tidak semua data tersinkronisasi pada masing- masing database sehingga data yang terdapat pada database hanya memiliki sebagian data yang sama.

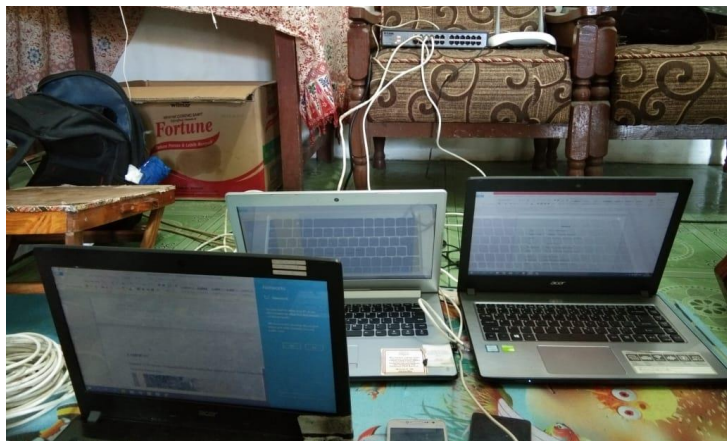
DAFTAR PUSTAKA

- [1] Hu, Bin. 2013. *Wi-Fi Based Indoor Positioning System Using Smartphones*.
- [2] Lukito, Yuan. 2017. *Perceptron Multilayer Model for Indoor Positioning Systems Wi-Fi based*.
- [3] S. Darekar, A. Chikane, R. Diwate, A. Deshmukh, and A. Shinde, "Tracking System using GPS and GSM: Practical Approach," *Int. J. Sci. Eng. Res.*, vol. 3, no. 5, pp. 3– 6, 2012.
- [4] A. Juansyah, "Pembangunan Aplikasi Child Tracker Berbasis Assisted – Global Positioning System (A-GPS) Dengan Platform Android," *J. Ilm. Komput. dan Inform.*, vol. 1, no. 1, pp. 1–8, 2015.
- [5] Sari, N.H. (2018). *Wi-Fi (Wireless Fidelity)*.
- [6] Al-Alawi, Adel Ismail. 2016. *Wi-Fi Technology: Future Market Challenges and Opportunities*.
- [7] Mulyana, H. (September 2013). Perancangan Aplikasi Pemeriksaan IP Address Aktif pada Jaringan Komputer dengan Metode Penguji *Black Box*. Cawang, Jakarta Timur. Vol. X No.1.
- [8] Riska., Ginta, Prama Wira., Patrick. 2017. Analisis and Implementation Wireless Extension Point with SSID.
- [9] Sigh, Akshay., Sharma, Sakshi., Sigh, Sashwat. 2016. *Android Application Development using Android Studio and PHP Framework*.
- [10] Sari, R.D., Siahaan, A.P.U., Supiyandi, S., Muttaqin, M. 2017. *A review of IP and MAC Address Filtering in Wireless Network Security*.
- [11] Wardoyo, S., Ryadi, T., Fhrizal, R. (2 September 2014). Analisis Performa *File Transport Protocol* pada perbandingan Metode IPv4 Murni, IPv6 Murni dan *Tunneling 6t04* Berbasis *Router Mikrotik*. Cilegon, Indonesia. Vol:3 No.2.
- [12] Duskarnaen, M. Ficky., Nurfalah, Febri. 2017. *Wireless network analysis, designing, and implementation point to point between campus a and campus b*.
- [13] Djumadi. *Website Development with PHP, Mysql, CSS and HTML*.
- [14] Harison., Syarif, Ahmad. 2016. Geographic information systems in means west pasaman district
- [15] Chowdhury, Md. Abu Hena. Network Based Location Tracking Using MAC Address of Smart Devices.
- [16] M. K. Hoq, "Mobile Tracking System using Web Application and

- Android Apps,” vol. 6, no. 02, pp. 257–262, 2017.
- [17] H. P. Yosephat Suryo Susilo, “Sistem Pelacakan Dan Pengamanan Kendaraan Berbasis GPS Dengan Menggunakan Komunikasi GPRS,” vol. 13, no. 1, pp. 21–32, 2014.
- [18] R. Ramadi, *Making travel history travel GPS TRACKER Web based on mobile phone using J2ME. Informatics Engineering. University Of Islam State Syarif Hidayatullah Jakarta*. 2011.
- [19] A. Sonita and R. F. Fardianitama, “Aplikasi E-Order Menggunakan Firebase dan Algoritme Knuth Morris Pratt Berbasis Android,” *Pseudocode*, vol. 5, no. 2, pp. 38– 45, 2018.

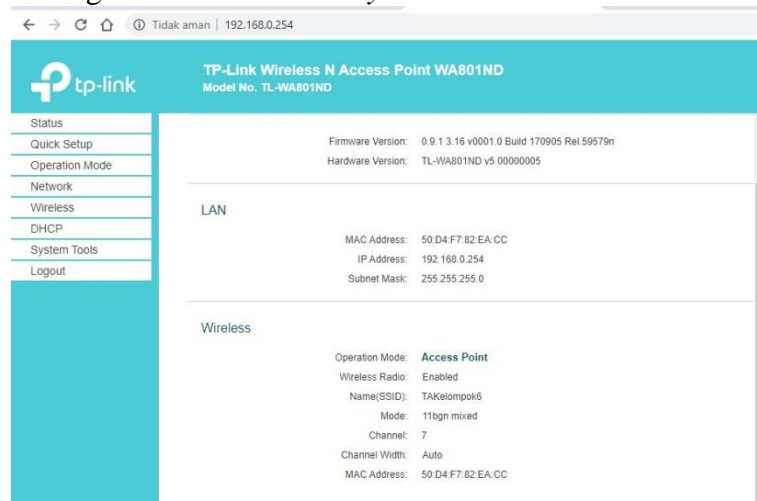
Lampiran

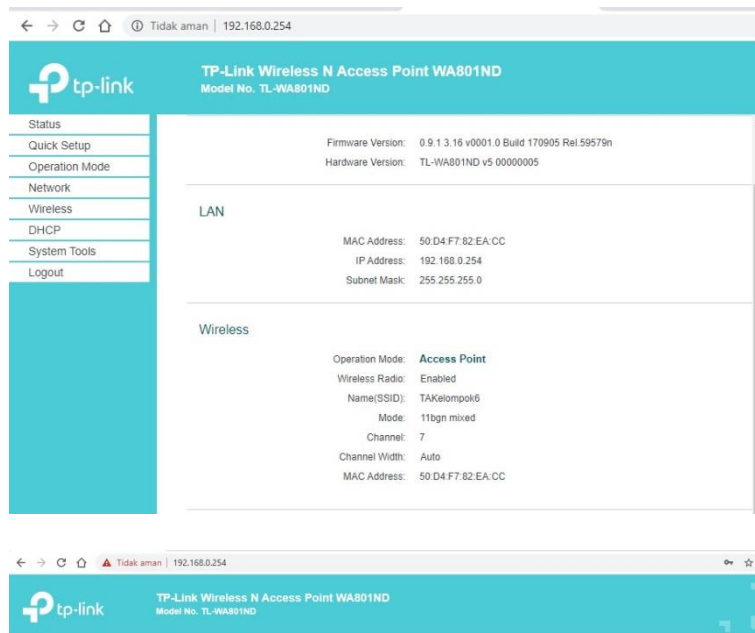
A. Gambaran Topologi Jaringan Pada Sistem



B. Konfigurasi Access Point

- Konfigurasi Access Point Dynamic





C. Code yang digunakan untuk android Code untuk database

```
package lksystems.wifiintruder;

import android.app.Activity;
import android.app.AlertDialog;
import android.content.ContentValues;
import android.content.Context;
import android.content.DialogInterface;
import android.content.DialogInterface.OnClickListener;
import android.content.Intent;
import android.database.Cursor;
import android.database.SQLException;
import android.database.sqlite.SQLiteDatabase;
import android.database.sqlite.SQLiteException;
import android.database.sqlite.SQLiteOpenHelper;
import android.net.DhcpInfo;
import android.net.wifi.ScanResult;
import android.net.wifi.WifiInfo;
import android.net.wifi.WifiManager;
import android.os.Build.VERSION;
import android.os.Bundle;
import android.text.Html;
import android.view.KeyEvent;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.widget.ListView;
import android.widget.RelativeLayout;
import android.widget.SimpleAdapter;
import android.widget.TextView;
import android.widget.Toast;
import com.google.android.gms.ads.AdListener;
import com.google.android.gms.ads.AdRequest;
import com.google.android.gms.ads.AdRequest.Builder;
import com.google.android.gms.ads.AdSize;
import com.google.android.gms.ads.AdView;
import com.google.android.gms.ads.InterstitialAd;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.InputStream;
import java.io.OutputStream;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;
import java.util.Timer;
import java.util.TimerTask;

public class MainActivity extends Activity {
    static DBHelper BD = null;
    static DBHelper BD2 = null;
    static DBHelper BD3 = null;
    static String DB_NAME = "bd";
    static String DB_NAME2 = "bdsettings";
    static String DB_NAME3 = "bdvendor";
```

```

static String FirstNumberIP = "";
static int IdDispositivoEscogido = 0;
static final String KEY_COL1 = "field1";
static final String KEY_COL2 = "field2";
static final String KEY_ID = "_id";
static boolean Killed;
static String LastNumberIP = "";
static String LinkSpeed = "";
static String MyIP = "";
static String MyMac = "";
static String MySSID = "";
static String MyVendor = "";
static boolean NeedScan = false;
static int NumberOnList = 0;
static String PartialIP = "";
static boolean PrimeraVezFirstScreen;
static String SecondNumberIP = "";
static String ThirdNumberIP = "";
static int ToastHelper = 0;
static int VersionApp = 0;
static int VersionDB = 0;
static InterstitialAd interstitial;
static InterstitialAd interstitial2;
static SQLiteDatabase myDataBase;
static SQLiteDatabase myDataBase2;
int BotonBack = 0;
List<ScanResult> Intensidad;
ListView Lista;
Context MyContext = this;
TextView Temp1;
TextView Temp2;
TextView Temp3;
TextView Temp4;
TextView Temp5;
TextView Temp6;
int TempIntensidad = 0;
SimpleAdapter adapter;
ArrayList<HashMap<String, String>> list = new ArrayList<>();
int size = 0;
Timer timer = new Timer();
Toast toast;
WifiManager wifi;

public class DBHelper extends SQLiteOpenHelper {
    final Context myContext;

    public DBHelper(Context context) {
        super(MainActivity.this.MyContext, MainActivity.DB_NAME,
null, 1);
        this.myContext = context;
    }

    public boolean removeSettings() {
        return MainActivity.myDataBase.delete("settings", null, null)
> 0;
    }

    public long insertSettings(String Version, int Votar) {
        ContentValues newValues = new ContentValues();

```

```

        newValues.put("version", Version);
        newValues.put("votar", Integer.valueOf(Votar));
        return MainActivity.myDataBase.insert("settings", null,
newValues);
    }

    public long insertDispositivo(String mac, String Name) {
        ContentValues newValues = new ContentValues();
        newValues.put("mac", mac);
        newValues.put("name", Name);
        return MainActivity.myDataBase.insert("friend_list", null,
newValues);
    }

    public boolean removeDispositivo(String mac) {
        if (mac.equals("todo")) {
            if (MainActivity.myDataBase.delete("friend_list", null,
null) > 0) {
                return true;
            }
            return false;
        } else if (MainActivity.myDataBase.delete("friend_list",
"mac='" + mac.toString() + "'", null) <= 0) {
            return false;
        } else {
            return true;
        }
    }

    public boolean updateAlarma(Integer _rowIndex, String categoria,
String nombreplato) {
        ContentValues newValues = new ContentValues();
        newValues.put(MainActivity.KEY_COL1, categoria);
        newValues.put(MainActivity.KEY_COL2, nombreplato);
        return MainActivity.myDataBase.update("alarma", newValues,
new StringBuilder().append("_id=").append(_rowIndex).toString(), null) >
0;
    }

    public boolean updateCat(Integer _rowIndex, String categoria) {
        ContentValues newValues = new ContentValues();
        newValues.put(MainActivity.KEY_COL1, categoria);
        return MainActivity.myDataBase.update("alarma", newValues,
new StringBuilder().append("_id=").append(_rowIndex).toString(), null) >
0;
    }

    public void createDataBase() throws IOException {
        if (!checkDataBase()) {
            getReadableDatabase();
            try {
                copyDataBase();
            } catch (IOException e) {
                throw new Error("Error copiando Base de Datos");
            }
        }
    }

    public boolean checkDataBase() {

```

```

        SQLiteDatabase checkDB = null;
        try {
            checkDB =
        SQLiteDatabase.openDatabase(MainActivity.this.getFilesDir().getPath() +
        MainActivity.DB_NAME, null, 0);
        } catch (SQLException e) {
        }
        if (checkDB != null) {
            checkDB.close();
        }
        if (checkDB != null) {
            return true;
        }
        return false;
    }

    public void copyDataBase() throws IOException {
        String obj = this.myContext.getAssets().toString();
        InputStream myInput =
        this.myContext.getAssets().open(MainActivity.DB_NAME);
        OutputStream myOutput = new
        FileOutputStream(MainActivity.this.getFilesDir().getPath() +
        MainActivity.DB_NAME);
        byte[] buffer = new byte[1024];
        while (true) {
            int length = myInput.read(buffer);
            if (length > 0) {
                myOutput.write(buffer, 0, length);
            } else {
                myOutput.flush();
                myOutput.close();
                myInput.close();
                return;
            }
        }
    }

    public void open() throws SQLException {
        try {
            createDataBase();
            String myPath = MainActivity.this.getFilesDir().getPath()
+ MainActivity.DB_NAME;
            if (MainActivity.DB_NAME == "bdvendor") {
                MainActivity.myDataBase2 =
                SQLiteDatabase.openDatabase(myPath, null, 0);
                try {
                    MainActivity.myDataBase.execSQL("delete from
        vendor");
                } catch (SQLException e) {
                }
            } else {
                MainActivity.myDataBase =
                SQLiteDatabase.openDatabase(myPath, null, 0);
            }
        } catch (IOException e2) {
            throw new Error("This is Error");
        }
    }
}

```

```

        public synchronized void close() {
            if (MainActivity.DB_NAME == "bdvendor") {
                if (MainActivity.myDataBase2 != null) {
                    MainActivity.myDataBase2.close();
                }
            } else if (MainActivity.myDataBase != null) {
                MainActivity.myDataBase.close();
            }
            super.close();
        }

        public void onCreate(SQLiteDatabase db) {
        }

        public void onUpgrade(SQLiteDatabase db, int oldVersion, int
newVersion) {
            Toast.makeText(MainActivity.this.MyContext, "update",
0).show();
        }
    }

    /* access modifiers changed from: protected */
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        interstitial = new InterstitialAd(this);
        interstitial2 = new InterstitialAd(this);
        interstitial.setAdUnitId("ca-app-pub-
2885006023541960/3057467337");
        interstitial2.setAdUnitId("ca-app-pub-
2885006023541960/6333472135");
        AdRequest adRequest2 = new Builder().build();
        if (FirstScreen.TempPubli) {
            interstitial2.loadAd(adRequest2);
        }
        interstitial2.setAdListener(new AdListener() {
            public void onAdClosed() {
                MainActivity.this.wifi = (WifiManager)
MainActivity.this.getSystemService("wifi");
                WifiInfo connectionInfo =
MainActivity.this.wifi.getConnectionInfo();
                if (!MainActivity.this.wifi.isWifiEnabled() ||
MainActivity.MyIP.equals("0.0.0.0")) {
                    Toast.makeText(MainActivity.this.MyContext,
MainActivity.this.getResources().getString(R.string.RedNoLista),
0).show();

                    MainActivity.this.wifi.setWifiEnabled(true);
                    return;
                }
                MainActivity.this.timer.cancel();
                MainActivity.NeedScan = true;
                Intent intent = new Intent(MainActivity.this.MyContext,
Red.class);
                intent.addFlags(131072);
                MainActivity.this.startActivity(intent);
            }
        });
        interstitial.setAdListener(new AdListener() {
            public void onAdClosed() {
                MainActivity.this.timer.cancel();
            }
        });
    }

```



```

        Intent intent = new Intent(MainActivity.this.MyContext,
Exit.class);
        intent.addFlags(131072);
        MainActivity.this.startActivity(intent);
        MainActivity.this.finish();
    }
});
Killed = true;
DB_NAME = DB_NAME2;
BD2 = new DBHelper(this.MyContext);
if (!PrimeraVezFirstScreen) {
    Intent intent = new Intent(getApplicationContext(),
FirstScreen.class);
    intent.addFlags(131072);
    startActivity(intent);
    finish();
    return;
}
setContentView(R.layout.activity_main);
this.Temp1 = (TextView) findViewById(R.id.textViewTitulo1);
this.Temp2 = (TextView) findViewById(R.id.textViewTitulo2);
this.Temp3 = (TextView) findViewById(R.id.textViewRed);
this.Temp5 = (TextView) findViewById(R.id.textViewRed2);
this.Temp1.setText(getResources().getString(R.string.Titulo1));
this.Temp2.setText(getResources().getString(R.string.Titulo2));

this.Temp3.setText(getResources().getString(R.string.ConectadoRed) + "
");
    DB_NAME = "bd";
    BD = new DBHelper(this.MyContext);
    BD.open();
    DB_NAME = "bdvendor";
    BD3 = new DBHelper(this.MyContext);
    BD3.open();
    this.wifi = (WifiManager) getSystemService("wifi");
    if (!this.wifi.isWifiEnabled()) {
        Toast.makeText(getApplicationContext(),
getResources().getString(R.string.ActivandoWifi), 1).show();
        this.wifi.setWifiEnabled(true);
    }
    this.list.clear();
    this.Lista = (ListView) findViewById(R.id.List);
    this.Lista.setDividerHeight(0);
    this.Lista.setDivider(null);
    this.adapter = new SimpleAdapter(this.MyContext, this.list,
R.layout.main_item_two_line_rows_main, new String[]{"line1", "line2"},
new int[]{R.id.text1, R.id.text2});
    this.Lista.setAdapter(this.adapter);
    this.Lista.setSelector(17170445);
    this.Lista.setCacheColorHint(0);
}

```

Code untuk membaca Access Point

```

import org.apache.commons.lang3.StringUtils;
import org.junit.After;
import org.junit.Before;
import org.junit.Test;
import org.junit.runner.RunWith;
import org.robolectric.annotation.Config;

```

```

import org.robolectric.annotation.LooperMode;

import java.util.Collections;

import androidx.annotation.NonNull;
import androidx.test.ext.junit.runners.AndroidJUnit4;

import static org.junit.Assert.assertEquals;
import static org.mockito.Mockito.mock;
import static org.mockito.Mockito.never;
import static org.mockito.Mockito.verify;
import static org.mockito.Mockito.when;
import static org.robolectric.annotation.LooperMode.Mode.PAUSED;

@RunWith(AndroidJUnit4.class)
@Config(sdk = Build.VERSION_CODES.P)
@LooperMode(PAUSED)
public class ConnectionViewTest {
    private static final String SSID = "SSID";
    private static final String BSSID = "BSSID";
    private static final String IP_ADDRESS = "IPADDRESS";

    private MainActivity mainActivity;
    private ConnectionView fixture;

    private Settings settings;
    private WiFiData wiFiData;
    private AccessPointDetail accessPointDetail;
    private AccessPointPopup accessPointPopup;

    @Before
    public void setUp() {
        mainActivity = RobolectricUtil.INSTANCE.getActivity();

        accessPointDetail = mock(AccessPointDetail.class);
        accessPointPopup = mock(AccessPointPopup.class);

        wiFiData = mock(WiFiData.class);
        settings = MainContextHelper.INSTANCE.getSettings();

        fixture = new ConnectionView(mainActivity);
        fixture.setAccessPointDetail(accessPointDetail);
        fixture.setAccessPointPopup(accessPointPopup);
    }

    @After
    public void tearDown() {
        MainContextHelper.INSTANCE.restore();

        mainActivity.setCurrentNavigationMenu(NavigationMenu.ACCESS_POINTS);
    }

    @Test
    public void testConnectionGoneWithNoConnectionInformation() {
        // setup

        when(settings.getConnectionViewType()).thenReturn(ConnectionViewType
            .COMPLETE);
    }

```

```

withConnectionInformation(withConnection(WiFiAdditional.EMPTY));
    // execute
    fixture.update(wifiData);
    // validate
    assertEquals(View.GONE,
mainActivity.findViewById(R.id.connection).getVisibility());
    verifyConnectionInformation();
}

@Test
public void
testConnectionGoneWithConnectionInformationAndHideType() {
    // setup
    WiFiDetail connection =
withConnection(withWiFiAdditional());

    when(settings.getConnectionViewType()).thenReturn(ConnectionViewType
.HIDE);
        withConnectionInformation(connection);
        withAccessPointDetailView(connection,
ConnectionViewType.COMPLETE.getAccessPointViewType());
        // execute
        fixture.update(wifiData);
        // validate
        assertEquals(View.GONE,
mainActivity.findViewById(R.id.connection).getVisibility());
        verifyConnectionInformation();
    }

@Test
public void testConnectionVisibleWithConnectionInformation() {
    // setup
    WiFiDetail connection =
withConnection(withWiFiAdditional());

    when(settings.getConnectionViewType()).thenReturn(ConnectionViewType
.COMPLETE);
        withConnectionInformation(connection);
        withAccessPointDetailView(connection,
ConnectionViewType.COMPLETE.getAccessPointViewType());
        // execute
        fixture.update(wifiData);
        // validate
        assertEquals(View.VISIBLE,
mainActivity.findViewById(R.id.connection).getVisibility());
        verifyConnectionInformation();
    }

@Test
public void testConnectionWithConnectionInformation() {
    // setup
    WiFiAdditional wifiAdditional = withWiFiAdditional();
    WiFiDetail connection = withConnection(wifiAdditional);

    when(settings.getConnectionViewType()).thenReturn(ConnectionViewType
.COMPLETE);
        withConnectionInformation(connection);
        withAccessPointDetailView(connection,
ConnectionViewType.COMPLETE.getAccessPointViewType());

```

```

        // execute
        fixture.update(wiFiData);
        // validate
        WiFiConnection wiFiConnection =
        wiFiAdditional.getWiFiConnection();
        View view = mainActivity.findViewById(R.id.connection);
        TextView ipAddressView = view.findViewById(R.id.ipAddress);
        assertEquals(wiFiConnection.getIpAddress(),
        ipAddressView.getText().toString());
        TextView linkSpeedView = view.findViewById(R.id.linkSpeed);
        assertEquals(View.VISIBLE, linkSpeedView.getVisibility());
        assertEquals(wiFiConnection.getLinkSpeed() +
        WifiInfo.LINK_SPEED_UNITS, linkSpeedView.getText().toString());
    }

    @Test
    public void testConnectionWithInvalidLinkSpeed() {
        // setup
        WiFiConnection wiFiConnection = new WiFiConnection(SSID,
        BSSID, IP_ADDRESS, WiFiConnection.LINK_SPEED_INVALID);
        WiFiDetail connection = withConnection(new
        WiFiAdditional(StringUtils.EMPTY, wiFiConnection));

        when(settings.getConnectionViewType()).thenReturn(ConnectionViewType
        .COMPLETE);
        withConnectionInformation(connection);
        withAccessPointDetailView(connection,
        ConnectionViewType.COMPLETE.getAccessPointViewType());
        // execute
        fixture.update(wiFiData);
        // validate
        View view = mainActivity.findViewById(R.id.connection);
        TextView linkSpeedView = view.findViewById(R.id.linkSpeed);
        assertEquals(View.GONE, linkSpeedView.getVisibility());
    }

    @Test
    public void testNoDataIsVisibleWithNoWiFiDetails() {
        // setup

        when(settings.getConnectionViewType()).thenReturn(ConnectionViewType
        .COMPLETE);

        when(wiFiData.getConnection()).thenReturn(withConnection(WiFiAdditio
        nal.EMPTY));
        // execute
        fixture.update(wiFiData);
        // validate
        assertEquals(View.VISIBLE,
        mainActivity.findViewById(R.id.no_data).getVisibility());
        assertEquals(View.VISIBLE,
        mainActivity.findViewById(R.id.no_location).getVisibility());
        verify(wiFiData).getWiFiDetails();
    }

    @Test
    public void testNoDataIsGoneWithWiFiDetails() {
        // setup
        WiFiDetail wiFiDetail =

```

```

withConnection(WiFiAdditional.EMPTY);

when(settings.getConnectionViewType()).thenReturn(ConnectionViewType
.COMPLETE);
    when(wiFiData.getConnection()).thenReturn(wiFiDetail);

when(wiFiData.getWiFiDetails()).thenReturn(Collections.singletonList
(wiFiDetail));
    // execute
    fixture.update(wiFiData);
    // validate
    assertEquals(View.GONE,
mainActivity.findViewById(R.id.no_data).getVisibility());
    verify(wiFiData).getWiFiDetails();
}

@Test
public void
testNoDataIsGoneWithNavigationMenuThatDoesNotHaveOptionsMenu() {
    // setup

mainActivity.setCurrentNavigationMenu(NavigationMenu.VENDORS);

when(settings.getConnectionViewType()).thenReturn(ConnectionViewType
.COMPLETE);

when(wiFiData.getConnection()).thenReturn(withConnection(WiFiAdditio
nal.EMPTY));
    // execute
    fixture.update(wiFiData);
    // validate
    assertEquals(View.GONE,
mainActivity.findViewById(R.id.no_data).getVisibility());
    verify(wiFiData, never()).getWiFiDetails();
}

@Test
public void testScanningIsVisibleWithNoWiFiDetails() {
    // setup

when(settings.getConnectionViewType()).thenReturn(ConnectionViewType
.COMPLETE);

when(wiFiData.getConnection()).thenReturn(withConnection(WiFiAdditio
nal.EMPTY));
    // execute
    fixture.update(wiFiData);
    // validate
    assertEquals(View.VISIBLE,
mainActivity.findViewById(R.id.scanning).getVisibility());
    verify(wiFiData).getWiFiDetails();
}

@Test
public void testScanningIsGoneWithWiFiDetails() {
    // setup
    WiFiDetail wiFiDetail =
withConnection(WiFiAdditional.EMPTY);

```

```

when(settings.getConnectionViewType()).thenReturn(ConnectionViewType
.COMPLETE);
    when(wiFiData.getConnection()).thenReturn(wiFiDetail);

when(wiFiData.getWiFiDetails()).thenReturn(Collections.singletonList
(wiFiDetail));
    // execute
    fixture.update(wiFiData);
    // validate
    assertEquals(View.GONE,
mainActivity.findViewById(R.id.scanning).getVisibility());
    verify(wiFiData).getWiFiDetails();
}

@Test
public void
testScanningIsGoneWithNavigationMenuThatDoesNotHaveOptionsMenu() {
    // setup

mainActivity.setCurrentNavigationMenu(NavigationMenu.VENDORS);

when(settings.getConnectionViewType()).thenReturn(ConnectionViewType
.COMPLETE);

when(wiFiData.getConnection()).thenReturn(withConnection(WiFiAdditio
nal.EMPTY));
    // execute
    fixture.update(wiFiData);
    // validate
    assertEquals(View.GONE,
mainActivity.findViewById(R.id.scanning).getVisibility());
    verify(wiFiData, never()).getWiFiDetails();
}

@Test
public void testViewCompactAddsPopup() {
    // setup
    WiFiDetail connection =
withConnection(withWiFiAdditional());

when(settings.getConnectionViewType()).thenReturn(ConnectionViewType
.COMPACT);
    withConnectionInformation(connection);
    View view = withAccessPointDetailView(connection,
ConnectionViewType.COMPACT.getAccessPointViewType());
    // execute
    fixture.update(wiFiData);
    // validate

verify(accessPointPopup).attach(view.findViewById(R.id.attachPopup),
connection);

verify(accessPointPopup).attach(view.findViewById(R.id.ssid),
connection);
}

private WiFiDetail withConnection(@NonNull WiFiAdditional
WiFiAdditional) {
    return new WiFiDetail(SSID, BSSID, StringUtils.EMPTY,

```

```

        new WiFiSignal(2435, 2435, WiFiWidth.MHZ_20, -55, true),
        wifiAdditional);
    }

    private WiFiAdditional withWiFiAdditional() {
        WiFiConnection wifiConnection = new WiFiConnection(SSID,
BSSID, IP_ADDRESS, 11);
        return new WiFiAdditional(StringUtils.EMPTY,
        wifiConnection);
    }

    private View withAccessPointDetailView(@NonNull WiFiDetail
        connection, @NonNull AccessPointViewType accessPointViewType) {
        ViewGroup parent =
        mainActivity.findViewById(R.id.connection).findViewById(R.id.connect
        ionDetail);
        View view =
        mainActivity.getLayoutInflater().inflate(accessPointViewType.getLayo
        ut()), parent, false);
        when(accessPointDetail.makeView(null, parent, connection,
        false, accessPointViewType)).thenReturn(view);
        when(accessPointDetail.makeView(parent.getChildAt(0),
        parent, connection, false, accessPointViewType)).thenReturn(view);
        return view;
    }

    private void withConnectionInformation(@NonNull WiFiDetail
        connection) {
        when(wifiData.getConnection()).thenReturn(connection);
    }

    private void verifyConnectionInformation() {
        verify(wifiData).getConnection();
    }
}

```

Code untuk membaca frekuensi Access Point

a. Frekuensi 5 GHZ

```

public class WiFiChannelsGHZ5Test {
    private WiFiChannelsGHZ5 fixture;

    @Before
    public void setUp() {
        fixture = new WiFiChannelsGHZ5();
    }

    @Test
    public void testGetWiFiChannelByFrequency() {
        validateFrequencyToChannel(5180, 5320, 10, 36, 2);
        validateFrequencyToChannel(5500, 5720, 10, 100, 2);
        validateFrequencyToChannel(5745, 5825, 10, 149, 2);
    }

    private void validateFrequencyToChannel(int frequencyStart, int
        frequencyEnd, int frequencyIncrement, int channelStart, int
        channelIncrement) {
    }
}

```

```

        int channel = channelStart;
        for (int frequency = frequencyStart; frequency <=
frequencyEnd; frequency += frequencyIncrement) {
            assertEquals(channel,
fixture.getWiFiChannelByFrequency(frequency).getChannel());
            channel += channelIncrement;
        }
    }

    @Test
    public void testGetWiFiChannelByFrequencyFail() {
        assertEquals(WiFiChannel.UNKNOWN,
fixture.getWiFiChannelByFrequency(5167));
        assertEquals(WiFiChannel.UNKNOWN,
fixture.getWiFiChannelByFrequency(5828));
    }

    @Test
    public void testGetWiFiChannelFirst() {
        assertEquals(36,
fixture.getWiFiChannelFirst().getChannel());
    }

    @Test
    public void testGetWiFiChannelLast() {
        assertEquals(165,
fixture.getWiFiChannelLast().getChannel());
    }

    @Test
    public void testGetWiFiChannelPair() {
        Pair<WiFiChannel, WiFiChannel> wiFiChannelPair =
fixture.getWiFiChannelPairFirst(Locale.JAPAN.getCountry());
        validatePair(36, 64, wiFiChannelPair);
    }

    @Test
    public void testGetWiFiChannelPairWithInvalidCountry() {
        Pair<WiFiChannel, WiFiChannel> wiFiChannelPair =
fixture.getWiFiChannelPairFirst(null);
        validatePair(36, 64, wiFiChannelPair);
    }

    @Test
    public void testGetWiFiChannelPairs() {
        List<Pair<WiFiChannel, WiFiChannel>> wiFiChannelPairs =
fixture.getWiFiChannelPairs();
        assertEquals(3, wiFiChannelPairs.size());
        validatePair(36, 64, wiFiChannelPairs.get(0));
        validatePair(100, 144, wiFiChannelPairs.get(1));
        validatePair(149, 165, wiFiChannelPairs.get(2));
    }

    private void validatePair(int expectedFirst, int expectedSecond,
Pair<WiFiChannel, WiFiChannel> pair) {
        assertEquals(expectedFirst, pair.first.getChannel());
        assertEquals(expectedSecond, pair.second.getChannel());
    }

```



```

@Test
public void testGetWiFiChannelByFrequency5GHZ() {
    // setup
    Pair<WiFiChannel, WiFiChannel> wifiChannelPair =
fixture.getWiFiChannelPairs().get(1);
    // execute
    WiFiChannel actual = fixture.getWiFiChannelByFrequency(2000,
wifiChannelPair);
    // validate
    assertEquals(WiFiChannel.UNKNOWN, actual);
}

@Test
public void testGetWiFiChannelByFrequency5GHZInRange() {
    // setup
    Pair<WiFiChannel, WiFiChannel> wifiChannelPair =
fixture.getWiFiChannelPairs().get(1);
    // execute
    WiFiChannel actual =
fixture.getWiFiChannelByFrequency(wifiChannelPair.first.getFrequency
()), wifiChannelPair);
    // validate
    assertEquals(wifiChannelPair.first, actual);
}

```

b. Frekuensi 2 GHZ

```

public class WiFiChannelsGHZ2Test {

    private WiFiChannelsGHZ2 fixture;

    @Before
    public void setUp() {
        fixture = new WiFiChannelsGHZ2();
    }

    @Test
    public void testIsInRange() {
        assertTrue(fixture.isInRange(2400));
        assertTrue(fixture.isInRange(2499));
    }

    @Test
    public void testIsNotInRange() {
        assertFalse(fixture.isInRange(2399));
        assertFalse(fixture.isInRange(2500));
    }

    @Test
    public void testGetWiFiChannelByFrequency() {
        assertEquals(1,
fixture.getWiFiChannelByFrequency(2410).getChannel());
        assertEquals(1,
fixture.getWiFiChannelByFrequency(2412).getChannel());
        assertEquals(1,
fixture.getWiFiChannelByFrequency(2414).getChannel());

        assertEquals(6,
fixture.getWiFiChannelByFrequency(2437).getChannel());
        assertEquals(7,

```

```

fixture.getWifiChannelByFrequency(2442).getChannel();

        assertEquals(13,
fixture.getWifiChannelByFrequency(2470).getChannel());
        assertEquals(13,
fixture.getWifiChannelByFrequency(2472).getChannel());
        assertEquals(13,
fixture.getWifiChannelByFrequency(2474).getChannel());

        assertEquals(14,
fixture.getWifiChannelByFrequency(2482).getChannel());
        assertEquals(14,
fixture.getWifiChannelByFrequency(2484).getChannel());
        assertEquals(14,
fixture.getWifiChannelByFrequency(2486).getChannel());
    }

    @Test
    public void testGetWifiChannelByFrequencyNotFound() {
        assertEquals(WifiChannel.UNKNOWN,
fixture.getWifiChannelByFrequency(2399));
        assertEquals(WifiChannel.UNKNOWN,
fixture.getWifiChannelByFrequency(2409));
        assertEquals(WifiChannel.UNKNOWN,
fixture.getWifiChannelByFrequency(2481));
        assertEquals(WifiChannel.UNKNOWN,
fixture.getWifiChannelByFrequency(2481));
        assertEquals(WifiChannel.UNKNOWN,
fixture.getWifiChannelByFrequency(2487));
        assertEquals(WifiChannel.UNKNOWN,
fixture.getWifiChannelByFrequency(2500));
    }

    @Test
    public void testGetWifiChannelByChannel() {
        assertEquals(2412,
fixture.getWifiChannelByChannel(1).getFrequency());
        assertEquals(2437,
fixture.getWifiChannelByChannel(6).getFrequency());
        assertEquals(2442,
fixture.getWifiChannelByChannel(7).getFrequency());
        assertEquals(2472,
fixture.getWifiChannelByChannel(13).getFrequency());
        assertEquals(2484,
fixture.getWifiChannelByChannel(14).getFrequency());
    }

    @Test
    public void testGetWifiChannelByChannelNotFound() {
        assertEquals(WifiChannel.UNKNOWN,
fixture.getWifiChannelByChannel(0));
        assertEquals(WifiChannel.UNKNOWN,
fixture.getWifiChannelByChannel(15));
    }

    @Test
    public void testGetWifiChannelFirst() {
        assertEquals(1, fixture.getWifiChannelFirst().getChannel());
    }

```

```

@Test
public void testGetWiFiChannelLast() {
    assertEquals(14, fixture.getWiFiChannelLast().getChannel());
}

@Test
public void testGetWiFiChannelPairs() {
    List<Pair<WiFiChannel, WiFiChannel>> pair =
fixture.getWiFiChannelPairs();
    assertEquals(1, pair.size());
    validatePair(1, 14, pair.get(0));
}

@Test
public void testGetWiFiChannelPair() {
    validatePair(1, 14,
fixture.getWiFiChannelPairFirst(Locale.US.getCountry()));
    validatePair(1, 14, fixture.getWiFiChannelPairFirst(null));
}

private void validatePair(int expectedFirst, int expectedSecond,
Pair<WiFiChannel, WiFiChannel> pair) {
    assertEquals(expectedFirst, pair.first.getChannel());
    assertEquals(expectedSecond, pair.second.getChannel());
}

@Test
public void testGetAvailableChannels() {
    assertEquals(11,
fixture.getAvailableChannels(Locale.US.getCountry()).size());
    assertEquals(13,
fixture.getAvailableChannels(Locale.UK.getCountry()).size());
}

@Test
public void testGetWiFiChannelByFrequency2GHZ() {
    // setup
    Pair<WiFiChannel, WiFiChannel> wiFiChannelPair =
fixture.getWiFiChannelPairs().get(0);
    // execute
    WiFiChannel actual = fixture.getWiFiChannelByFrequency(2000,
wiFiChannelPair);
    // validate
    assertEquals(WiFiChannel.UNKNOWN, actual);
}

@Test
public void testGetWiFiChannelByFrequency2GHZInRange() {
    // setup
    Pair<WiFiChannel, WiFiChannel> wiFiChannelPair =
fixture.getWiFiChannelPairs().get(0);
    // execute
    WiFiChannel actual =
fixture.getWiFiChannelByFrequency(wiFiChannelPair.first.getFrequency(),
wiFiChannelPair);
    // validate
    assertEquals(wiFiChannelPair.first, actual);
}

```

```
}
```

Code untuk tampilan grafik kecepatan *Access Point*

```
public class DataManagerTest {
    private static final int LEVEL = -40;

    private DataManager fixture;

    @Before
    public void setUp() {
        RobolectricUtil.INSTANCE.getActivity();
        fixture = new DataManager();
    }

    @Test
    public void testGetNewSeries() {
        // setup
        Pair<WiFiChannel, WiFiChannel> wifiChannelPair =
        WiFiBand.GHZ2.getWiFiChannels().getWiFiChannelPairs().get(0);
        List<WiFiDetail> expected =
        makeWiFiDetails(wifiChannelPair.first.getFrequency());
        // execute
        Set<WiFiDetail> actual = fixture.getNewSeries(expected,
        wifiChannelPair);
        // validate
        assertEquals(expected.size() - 1, actual.size());
        assertTrue(actual.contains(expected.get(0)));
        assertFalse(actual.contains(expected.get(1)));
        assertTrue(actual.contains(expected.get(2)));
    }

    @Test
    public void testGetDataPoints() {
        // setup
        WiFiDetail expected = makeWiFiDetail("SSID", 2455);
        // execute
        DataPoint[] actual = fixture.getDataPoints(expected,
        GraphConstants.MAX_Y);
        // validate
        assertEquals(5, actual.length);
        assertEquals(new DataPoint(2445, -100).toString(),
        actual[0].toString());
        assertEquals(new DataPoint(2450, LEVEL).toString(),
        actual[1].toString());
        assertEquals(new DataPoint(2455, LEVEL).toString(),
        actual[2].toString());
        assertEquals(new DataPoint(2460, LEVEL).toString(),
        actual[3].toString());
        assertEquals(new DataPoint(2465, -100).toString(),
        actual[4].toString());
    }

    @Test
    public void testGetDataPointsExpectLevelToEqualLevelMax() {
```

```

        // setup
        int expectedLevel = LEVEL - 10;
        WiFiDetail expected = makeWiFiDetail("SSID", 2455);
        // execute
        DataPoint[] actual = fixture.getDataPoints(expected,
expectedLevel);
        // validate
        assertEquals(5, actual.length);
        assertEquals(new DataPoint(2445, -100).toString(),
actual[0].toString());
        assertEquals(new DataPoint(2450, expectedLevel).toString(),
actual[1].toString());
        assertEquals(new DataPoint(2455, expectedLevel).toString(),
actual[2].toString());
        assertEquals(new DataPoint(2460, expectedLevel).toString(),
actual[3].toString());
        assertEquals(new DataPoint(2465, -100).toString(),
actual[4].toString());
    }

    @Test
    public void testAddSeriesDataWithExistingWiFiDetails() {
        // setup
        GraphViewWrapper graphViewWrapper = mock(GraphViewWrapper.class);
        WiFiDetail wiFiDetail = makeWiFiDetail("SSID", 2455);
        Set<WiFiDetail> wiFiDetails = Collections.singleton(wiFiDetail);
        DataPoint[] dataPoints = fixture.getDataPoints(wiFiDetail,
GraphConstants.MAX_Y);
        when(graphViewWrapper.isNewSeries(wiFiDetail)).thenReturn(false);
        // execute
        fixture.addSeriesData(graphViewWrapper, wiFiDetails,
GraphConstants.MAX_Y);
        // validate
        verify(graphViewWrapper).isNewSeries(wiFiDetail);
        verify(graphViewWrapper).updateSeries(
            eq(wiFiDetail),
            argThat(new DataPointsEquals(dataPoints)),
            eq(Boolean.TRUE));
    }

    @Test
    public void testAddSeriesDataNewWiFiDetails() {
        // setup
        GraphViewWrapper graphViewWrapper = mock(GraphViewWrapper.class);
        WiFiDetail wiFiDetail = makeWiFiDetail("SSID", 2455);
        Set<WiFiDetail> wiFiDetails = Collections.singleton(wiFiDetail);
        when(graphViewWrapper.isNewSeries(wiFiDetail)).thenReturn(true);
        // execute
        fixture.addSeriesData(graphViewWrapper, wiFiDetails,
GraphConstants.MAX_Y);
        // validate
        verify(graphViewWrapper).isNewSeries(wiFiDetail);
        verify(graphViewWrapper).addSeries(
            eq(wiFiDetail),
            ArgumentMatchers.<TitleLineGraphSeries<DataPoint>>any(),
            eq(Boolean.TRUE));
    }

    private WiFiDetail makeWiFiDetail(@NonNull String SSID, int frequency)

```

```

{
    WiFiSignal wiFiSignal = new WiFiSignal(frequency, frequency,
WiFiWidth.MHZ_20, LEVEL, true);
    return new WiFiDetail(SSID, "BSSID", StringUtils.EMPTY,
wiFiSignal, WiFiAdditional.EMPTY);
}

private List<WiFiDetail> makeWiFiDetails(int frequency) {
    return Arrays.asList(makeWiFiDetail("SSID1", frequency),
makeWiFiDetail("SSID2", -frequency), makeWiFiDetail("SSID3", frequency));
}
}

```

Code untuk menampilkan grafik perwaktu

```

public class DataManagerTest {
    private static final String BSSID = "BSSID";
    private static final int LEVEL = -40;
    private GraphViewWrapper graphViewWrapper;
    private TimeGraphCache timeGraphCache;
    private DataManager fixture;

    @Before
    public void setUp() {
        RobolectricUtil.INSTANCE.getActivity();

        timeGraphCache = mock(TimeGraphCache.class);
        graphViewWrapper = mock(GraphViewWrapper.class);

        fixture = new DataManager();
        fixture.setTimeGraphCache(timeGraphCache);
    }

    @Test
    public void testAddSeriesDataIncreaseXValue() {
        // setup
        assertEquals(0, fixture.getXValue());
        // execute
        fixture.addSeriesData(graphViewWrapper, Collections.emptyList(),
GraphConstants.MAX_Y);
        // validate
        assertEquals(1, fixture.getXValue());
    }

    @Test
    public void testAddSeriesDataIncreaseCounts() {
        // setup
        assertEquals(0, fixture.getScanCount());
        // execute
        fixture.addSeriesData(graphViewWrapper, Collections.emptyList(),
GraphConstants.MAX_Y);
        // validate
        assertEquals(1, fixture.getScanCount());
    }

    @Test
    public void testAddSeriesDoesNotIncreasesScanCountWhenLimitIsReached()

```

```

{
    // setup
    fixture.setScanCount(GraphConstants.MAX_SCAN_COUNT);
    // execute
    fixture.addSeriesData(graphViewWrapper, Collections.emptyList(),
GraphConstants.MAX_Y);
    // validate
    assertEquals(GraphConstants.MAX_SCAN_COUNT,
fixture.getScanCount());
}

@Test
public void testAddSeriesSetHorizontalLabelsVisible() {
    // setup
    fixture.setScanCount(1);
    // execute
    fixture.addSeriesData(graphViewWrapper, Collections.emptyList(),
GraphConstants.MAX_Y);
    // validate
    assertEquals(2, fixture.getScanCount());
    verify(graphViewWrapper).setHorizontalLabelsVisible(true);
}

@Test
public void testAddSeriesDoesNotSetHorizontalLabelsVisible() {
    // execute
    fixture.addSeriesData(graphViewWrapper, Collections.emptyList(),
GraphConstants.MAX_Y);
    // validate
    verify(graphViewWrapper,
never()).setHorizontalLabelsVisible(true);
}

@Test
public void testAdjustDataAppendsData() {
    // setup
    Set<WiFiDetail> wiFiDetails = Collections.emptySet();
    List<WiFiDetail> difference = makeWiFiDetails();
    int xValue = fixture.getXValue();
    Integer scanCount = fixture.getScanCount();
    DataPoint dataPoint = new DataPoint(xValue, GraphConstants.MIN_Y +
GraphConstants.MIN_Y_OFFSET);

    when(graphViewWrapper.differenceSeries(wiFiDetails)).thenReturn(difference
);
    // execute
    fixture.adjustData(graphViewWrapper, wiFiDetails);
    // validate
    IterableUtils.forEach(difference, new WiFiDetailClosure(dataPoint,
scanCount));
    verify(timeGraphCache).clear();
}

@Test
public void testGetNewSeries() {
    // setup
    Set<WiFiDetail> wiFiDetails = new TreeSet<>(makeWiFiDetails());
    Set<WiFiDetail> moreWiFiDetails = new
TreeSet<>(makeMoreWiFiDetails());

```

```

        when(timeGraphCache.active()).thenReturn(moreWiFiDetails);
        // execute
        Set<WiFiDetail> actual = fixture.getNewSeries(wiFiDetails);
        // validate
        assertTrue(actual.containsAll(wiFiDetails));
        assertTrue(actual.containsAll(moreWiFiDetails));
        verify(timeGraphCache).active();
    }

    @Test
    public void testAddDataToExistingSeries() {
        // setup
        Integer scanCount = fixture.getScanCount();
        int xValue = fixture.getXValue();
        WiFiDetail wiFiDetail = makeWiFiDetail("SSID");
        DataPoint dataPoint = new DataPoint(xValue, LEVEL);
        when(graphViewWrapper.isNewSeries(wiFiDetail)).thenReturn(false);
        // execute
        fixture.addData(graphViewWrapper, wiFiDetail,
            GraphConstants.MAX_Y);
        // validate
        verify(graphViewWrapper).isNewSeries(wiFiDetail);
        verify(graphViewWrapper).appendToSeries(
            eq(wiFiDetail),
            argThat(new DataPointEquals(dataPoint)),
            eq(scanCount),

eq(wiFiDetail.getWiFiAdditional().getWiFiConnection().isConnected()));
        verify(timeGraphCache).reset(wiFiDetail);
    }

    @Test
    public void testAddDataToExistingSeriesExpectLevelToEqualToLevelMax()
    {
        // setup
        int expectedLevel = LEVEL - 10;
        Integer scanCount = fixture.getScanCount();
        int xValue = fixture.getXValue();
        WiFiDetail wiFiDetail = makeWiFiDetail("SSID");
        DataPoint dataPoint = new DataPoint(xValue, expectedLevel);
        when(graphViewWrapper.isNewSeries(wiFiDetail)).thenReturn(false);
        // execute
        fixture.addData(graphViewWrapper, wiFiDetail, expectedLevel);
        // validate
        verify(graphViewWrapper).appendToSeries(
            eq(wiFiDetail),
            argThat(new DataPointEquals(dataPoint)),
            eq(scanCount),

eq(wiFiDetail.getWiFiAdditional().getWiFiConnection().isConnected()));
    }

    @Test
    public void testAddDataNewSeries() {
        // setup
        WiFiDetail wiFiDetail = makeWiFiDetailConnected("SSID");
        when(graphViewWrapper.isNewSeries(wiFiDetail)).thenReturn(true);
        // execute

```



```

        fixture.addData(graphViewWrapper, wiFiDetail,
GraphConstants.MAX_Y);
        // validate
        verify(graphViewWrapper).isNewSeries(wiFiDetail);
        verify(timeGraphCache).reset(wiFiDetail);
        verify(graphViewWrapper).addSeries(
            eq(wiFiDetail),
            ArgumentMatchers.<LineGraphSeries<DataPoint>>any(),
eq(wiFiDetail.getWi-FiAdditional().getWi-FiConnection().isConnected()));
    }

    private Wi-FiDetail makeWi-FiDetailConnected(@NonNull String SSID) {
        Wi-FiConnection wi-FiConnection = new Wi-FiConnection(SSID, BSSID,
"IPADDRESS", 11);
        Wi-FiAdditional wi-FiAdditional = new Wi-FiAdditional("VendorName",
wi-FiConnection);
        return new Wi-FiDetail(SSID, BSSID, StringUtils.EMPTY,
makeWi-FiSignal(), wi-FiAdditional);
    }

    private Wi-FiSignal makeWi-FiSignal() {
        return new Wi-FiSignal(2455, 2455, Wi-FiWidth.MHZ_20, LEVEL, true);
    }

    private Wi-FiDetail makeWi-FiDetail(@NonNull String SSID) {
        return new Wi-FiDetail(SSID, BSSID, StringUtils.EMPTY,
makeWi-FiSignal(), Wi-FiAdditional.EMPTY);
    }

    private List<Wi-FiDetail> makeWi-FiDetails() {
        return Arrays.asList(makeWi-FiDetailConnected("SSID1"),
makeWi-FiDetail("SSID2"), makeWi-FiDetail("SSID3"));
    }

    private List<Wi-FiDetail> makeMoreWi-FiDetails() {
        return Arrays.asList(makeWi-FiDetail("SSID4"),
makeWi-FiDetail("SSID5"));
    }

    private class Wi-FiDetailClosure implements Closure<Wi-FiDetail> {
        private final DataPoint dataPoint;
        private final Integer scanCount;

        private Wi-FiDetailClosure(@NonNull DataPoint dataPoint, @NonNull
Integer scanCount) {
            this.dataPoint = dataPoint;
            this.scanCount = scanCount;
        }

        @Override
        public void execute(Wi-FiDetail wi-FiDetail) {
            verify(graphViewWrapper).appendToSeries(
                eq(wi-FiDetail),
                argThat(new DataPointEquals(dataPoint)),
                eq(scanCount),
eq(wi-FiDetail.getWi-FiAdditional().getWi-FiConnection().isConnected()));
            verify(timeGraphCache).add(wi-FiDetail);

```

```

    }
}
}

```

Code untuk membaca *technology vendor*

```

public class VendorAdapterTest {
    private static final String VENDOR_NAME1 = "N1";
    private static final String VENDOR_NAME2 = "N2";
    private static final String VENDOR_NAME3 = "N3";

    private MainActivity mainActivity;
    private VendorService vendorService;
    private List<String> vendors;
    private VendorAdapter fixture;

    @Before
    public void setUp() {
        mainActivity = RobolectricUtil.INSTANCE.getActivity();
        vendorService = MainContextHelper.INSTANCE.getVendorService();

        vendors = Arrays.asList(VENDOR_NAME1, VENDOR_NAME2, VENDOR_NAME3);
        when(vendorService.findVendors()).thenReturn(vendors);

        fixture = new VendorAdapter(mainActivity, vendorService);
    }

    @After
    public void tearDown() {
        MainContextHelper.INSTANCE.restore();
    }

    @Test
    public void testConstructor() {
        assertEquals(vendors.size(), fixture.getCount());
        assertEquals(vendors.get(0), fixture.getItem(0));
        assertEquals(vendors.get(1), fixture.getItem(1));
        assertEquals(vendors.get(2), fixture.getItem(2));
        verify(vendorService).findVendors();
    }

    @Test
    public void testGetView() {
        // setup

        when(vendorService.findMacAddresses(VENDOR_NAME2)).thenReturn(Arrays.asList("VALUE1", "VALUE2", "VALUE3"));
        String expected = "VALUE1, VALUE2, VALUE3";
        ViewGroup viewGroup =
mainActivity.findViewById(android.R.id.content);
        // execute
        View actual = fixture.getView(1, null, viewGroup);
        // validate
        assertNotNull(actual);

        assertEquals(VENDOR_NAME2,
actual.<TextView>findViewById(R.id.vendor_name).getText().toString());
    }
}

```

```

        assertEquals(expected,
actual.<TextView>findViewById(R.id.vendor_macs).getText().toString());

        verify(vendorService).findMacAddresses(VENDOR_NAME2);

        verify(vendorService, never()).findVendorName(VENDOR_NAME1);
        verify(vendorService, never()).findVendorName(VENDOR_NAME3);
    }

```

Code untuk membaca user

```

public class Dispositivo extends Activity {
    Button Boton1;
    ListView Lista;
    Context MyContext = this;
    EditText Name;
    int QueEs = 0;
    TextView Temp1;
    TextView Temp2;
    TextView Temp3;
    TextView Temp4;
    TextView Temp5;
    TextView Temp6;
    int TempContador = 0;
    int TempI;
    String TempIP;
    String TempMac = "";
    String TempName;
    String TempVendor;
    SimpleAdapter adapter;
    ArrayList<HashMap<String, String>> list = new ArrayList<>();

    /* access modifiers changed from: protected */
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.dispositivo);
        this.Temp1 = (TextView) findViewById(R.id.textViewTitulo1);
        this.Temp2 = (TextView) findViewById(R.id.textViewTitulo2);
        this.Temp3 = (TextView) findViewById(R.id.textViewRed);
        this.Temp5 = (TextView) findViewById(R.id.textViewRed2);
        this.Temp6 = (TextView) findViewById(R.id.textview1);
        this.Temp1.setText(getResources().getString(R.string.Titulo7));
        this.Temp2.setText(getResources().getString(R.string.Titulo8));
        this.list.clear();
        this.Lista = (ListView) findViewById(R.id.List);
        this.Lista.setDividerHeight(0);
        this.Lista.setDivider(null);
        this.adapter = new SimpleAdapter(this.MyContext, this.list,
R.layout.main_item_two_line_rows_dispositivo, new String[]{"line1",
"line2"}, new int[]{R.id.text1, R.id.text2});
        this.Lista.setAdapter(this.adapter);
        this.Lista.setSelector(17170445);
        this.Lista.setCacheColorHint(0);
        this.Boton1 = (Button) findViewById(R.id.button);
        this.Name = (EditText) findViewById(R.id.editText);
        this.Name.setHint(getResources().getString(R.string.PonerNombre));
    }

```

```

    }

    /* access modifiers changed from: protected */
    public void onStart() {
        super.onStart();
    }

    /* access modifiers changed from: protected */
    public void onResume() {
        this.TempContador = 0;
        if (MainActivity.MyIP.equals("0.0.0.0") ||
!MainActivity.PrimeravezFirstScreen) {
            Intent intent = new Intent(getApplicationContext(),
MainActivity.class);
            intent.addFlags(536870912);
            startActivity(intent);
            super.finish();
        }
        int i = 1;
        while (i <= 254) {
            if (MainActivity.Killed &&
Red.InDispositivos[0].InDispositivo[i].IP != null) {
                if (Red.InDispositivos[0].InDispositivo[i].Name == null) {
                    Red.InDispositivos[0].InDispositivo[i].Name = "";
                }
                if (Red.InDispositivos[0].InDispositivo[i].Vendor == null)
{
                    Red.InDispositivos[0].InDispositivo[i].Vendor = "";
                }
                if (this.TempContador ==
MainActivity.IdDispositivoEscogido) {
                    this.TempI = i;
                    this.TempMac =
Red.InDispositivos[0].InDispositivo[i].Mac;
                    if
(Red.InDispositivos[0].InDispositivo[i].IP.equals(MainActivity.MyIP)) {
                        this.QueEs = 0;
                        this.Name.setVisibility(4);
                        try {
                            this.list.clear();
                            new HashMap();
                            HashMap<String, String> item = new
HashMap<>();
                            item.put("line1", "IP: ");
                            item.put("line2", MainActivity.MyIP);
                            this.list.add(item);
                            HashMap<String, String> item2 = new
HashMap<>();
                            item2.put("line1",
getResources().getString(R.string.Fabricante) + ": ");
                            item2.put("line2", MainActivity.MyVendor);
                            this.list.add(item2);
                            HashMap<String, String> item3 = new
HashMap<>();
                            item3.put("line1",
getResources().getString(R.string.MACAdress) + " ");
                            item3.put("line2", MainActivity.MyMac);
                            item3.put("line3", "-----");
                            this.list.add(item3);

```

```

runOnUiThread(new Runnable() {
    public void run() {

Dispositivo.this.adapter.notifyDataSetChanged();
    }
});
} catch (Exception e) {
}

this.Temp3.setText(getResources().getString(R.string.EsteEsTuDispositivo))
;
    this.Temp5.setText("");
    this.Temp6.setText("");
} else if
(Red.InDispositivos[0].InDispositivo[i].IP.equals(MainActivity.MyGateWay))
{
    this.QueEs = 1;
    this.Name.setVisibility(4);
    this.list.clear();
    new HashMap();
    HashMap<String, String> item4 = new HashMap<>();
    item4.put("line1", "IP: ");
    item4.put("line2",
Red.InDispositivos[0].InDispositivo[i].IP);
    this.list.add(item4);
    HashMap<String, String> item5 = new HashMap<>();
    item5.put("line1",
getResources().getString(R.string.Fabricante) + ": ");
    item5.put("line2",
Red.InDispositivos[0].InDispositivo[i].Vendor);
    this.list.add(item5);
    String TempConocido =
Red.InDispositivos[0].InDispositivo[i].Name;
    Cursor cursor =
MainActivity.myDataBase.rawQuery("Select name from friend_list where mac =
'" + Red.InDispositivos[0].InDispositivo[i].Mac + "'", null);
    if (cursor.moveToFirst() && cursor.getString(0) !=
"") {
        TempConocido = cursor.getString(0);
    }
    cursor.close();
    HashMap<String, String> item6 = new HashMap<>();
    item6.put("line1",
getResources().getString(R.string.NetBios) + ": ");
    if
(Red.InDispositivos[0].InDispositivo[i].Name.equals("")) {
        item6.put("line2", " - ");
    } else {
        item6.put("line2", TempConocido);
    }
    this.list.add(item6);
    HashMap<String, String> item7 = new HashMap<>();
    item7.put("line1", "MacAddress: ");
    item7.put("line2",
Red.InDispositivos[0].InDispositivo[i].Mac);
    this.list.add(item7);
    runOnUiThread(new Runnable() {
        public void run() {

```

```

Dispositivo.this.adapter.notifyDataSetChanged();
        }
    });

    this.Temp3.setText(getResources().getString(R.string.GateWay2));
    this.Temp5.setText("");

    this.Temp6.setText(getResources().getString(R.string.Router));
    } else {
        boolean TempConocido2 = false;
        Cursor cursor2 =
MainActivity.myDataBase.rawQuery("Select mac from friend_list where mac =
'" + Red.InDispositivos[0].InDispositivo[i].Mac + "'", null);
        if (cursor2.moveToFirst()) {
            for (int i2 = 1; cursor2.getCount() >= i2;
i2++) {
                TempConocido2 = true;
                cursor2.moveToNext();
            }
        }
        cursor2.close();
        if (TempConocido2) {
            this.QueEs = 2;
            this.Name.setVisibility(0);
            this.list.clear();
            new HashMap();
            HashMap<String, String> item8 = new
HashMap<>();
            item8.put("line1", "IP: ");
            item8.put("line2",
Red.InDispositivos[0].InDispositivo[i].IP);
            this.list.add(item8);
            this.TempIP =
Red.InDispositivos[0].InDispositivo[i].IP;
            HashMap<String, String> item9 = new
HashMap<>();
            item9.put("line1",
getResources().getString(R.string.Fabricante) + ": ");
            item9.put("line2",
Red.InDispositivos[0].InDispositivo[i].Vendor);
            this.list.add(item9);
            this.TempVendor =
Red.InDispositivos[0].InDispositivo[i].Vendor;
            HashMap<String, String> item10 = new
HashMap<>();
            item10.put("line1",
getResources().getString(R.string.NetBios) + ": ");
            if
(Red.InDispositivos[0].InDispositivo[i].Name.equals("")) {
                item10.put("line2", " - ");
            } else {
                item10.put("line2",
Red.InDispositivos[0].InDispositivo[i].Name);
            }
            this.list.add(item10);
            this.TempName =
Red.InDispositivos[0].InDispositivo[i].Name;
            String str =
Red.InDispositivos[0].InDispositivo[i].Name;

```

```

        Cursor cursor3 =
MainActivity.myDataBase.rawQuery("Select name from friend_list where mac =
'" + Red.InDispositivos[0].InDispositivo[i].Mac + "'", null);
        if (cursor3.moveToFirst() &&
cursor3.getString(0) != "") {
            String TempConocido22 =
cursor3.getString(0);
        }
        cursor3.close();

this.Name.setHint(getResources().getString(R.string.CambiarNombre));
        HashMap<String, String> item11 = new
HashMap<>();
        item11.put("line1", "MacAddress: ");
        item11.put("line2",
Red.InDispositivos[0].InDispositivo[i].Mac);
        this.list.add(item11);
        this.TempMac =
Red.InDispositivos[0].InDispositivo[i].Mac;
        runOnUiThread(new Runnable() {
            public void run() {

Dispositivo.this.adapter.notifyDataSetChanged();
            }
        });

this.Temp3.setText(getResources().getString(R.string.Conocido));
this.Temp5.setText(getResources().getString(R.string.Conocido2));
this.Temp6.setText(getResources().getString(R.string.DispConocido2));
        } else {
            this.QueEs = 3;
            this.Name.setVisibility(0);
            this.list.clear();
            new HashMap();
            HashMap<String, String> item12 = new
HashMap<>();
            item12.put("line1", "IP: ");
            item12.put("line2",
Red.InDispositivos[0].InDispositivo[i].IP);
            this.list.add(item12);
            HashMap<String, String> item13 = new
HashMap<>();
            item13.put("line1",
getResources().getString(R.string.Fabricante) + ": ");
            item13.put("line2",
Red.InDispositivos[0].InDispositivo[i].Vendor);
            this.list.add(item13);
            String TempConocido23 =
Red.InDispositivos[0].InDispositivo[i].Name;
            Cursor cursor4 =
MainActivity.myDataBase.rawQuery("Select name from friend_list where mac =
'" + Red.InDispositivos[0].InDispositivo[i].Mac + "'", null);
            if (cursor4.moveToFirst() &&
cursor4.getString(0) != "") {
                TempConocido23 = cursor4.getString(0);
            }
            cursor4.close();

```

```

HashMap<>();
HashMap<String, String> item14 = new
    item14.put("line1",
getResources().getString(R.string.NetBios) + ": ");
    if
(Red.InDispositivos[0].InDispositivo[i].Name.equals("")) {
        item14.put("line2", " - ");
    } else {
        item14.put("line2", TempConocido23);
    }
    this.list.add(item14);
    HashMap<String, String> item15 = new
HashMap<>();
    item15.put("line1", "MacAddress: ");
    item15.put("line2",
Red.InDispositivos[0].InDispositivo[i].Mac);
    this.list.add(item15);
    runOnUiThread(new Runnable() {
        public void run() {

Dispositivo.this.adapter.notifyDataSetChanged();
        }
    });

this.Temp3.setText(getResources().getString(R.string.Desconocido));
this.Temp5.setText(getResources().getString(R.string.Desconocido2));
this.Temp6.setText(getResources().getString(R.string.DispDesconocido2));
    }
    }
    i = 255;
    }
    this.TempContador++;
    }
    i++;
    }
    if (this.QueEs == 0 || this.QueEs == 1) {
this.Boton1.setText(getResources().getString(R.string.Retroceder));
    } else if (this.QueEs == 2) {

this.Boton1.setText(getResources().getString(R.string.DelDevice));
    } else {

this.Boton1.setText(getResources().getString(R.string.AddDevice));
    }
    AdView adView = new AdView(this);
    adView.setAdUnitId("ca-app-pub-2885006023541960/1382257735");
    adView.setAdSize(AdSize.SMART_BANNER);
    AdView adView2 = (AdView) findViewById(R.id.adView);
    adView2.loadAd(new Builder().build());
    final AdView finalAdView = adView2;
    adView2.setAdListener(new AdListener() {
        public void onAdLoaded() {
            int height = finalAdView.getHeight();
            new RelativeLayout(Dispositivo.this.MyContext);
            ((RelativeLayout)
Dispositivo.this.findViewById(R.id.TODO2)).setPadding(0, 0, 0, height +

```



```

1);
    }
    });
    super.onResume();
}

/* access modifiers changed from: protected */
public void onPause() {
    super.onPause();
}

/* access modifiers changed from: protected */
public void onStop() {
    super.onStop();
}

/* access modifiers changed from: protected */
public void onRestart() {
    super.onRestart();
}

/* access modifiers changed from: protected */
public void onDestroy() {
    super.onDestroy();
}

public void OnClickButton(View v) {
    if (this.QueEs == 3) {
        MainActivity.NumberOnList++;
        this.Name.toString().trim().equals("");
        if (this.Name.length() >= 1) {
            Red.InDispositivos[0].InDispositivo[this.TempI].Name =
this.Name.getText().toString();
            MainActivity.BD.insertDispositivo(this.TempMac,
this.Name.getText().toString());
        } else {
            MainActivity.BD.insertDispositivo(this.TempMac, "");
        }
        MainActivity.ToastHelper = 2;
        if (MainActivity.MyIP.equals("0.0.0.0") ||
!MainActivity.PrimeravezFirstScreen) {
            Intent intent = new Intent(getApplicationContext(),
FirstScreen.class);
            intent.addFlags(536870912);
            startActivity(intent);
            finish();
            return;
        }
        Intent intent2 = new Intent(this.MyContext, Red.class);
        intent2.addFlags(131072);
        startActivity(intent2);
        finish();
    } else if (this.QueEs == 2) {
        this.TempName =
Red.InDispositivos[0].InDispositivo[this.TempI].Name;
        String TempConocido2 =
Red.InDispositivos[0].InDispositivo[this.TempI].Name;
        Cursor cursor = MainActivity.myDataBase.rawQuery("Select name
from friend_list where mac = '" +

```

```

Red.InDispositivos[0].InDispositivo[this.TempI].Mac + "'", null);
    if (!(!cursor.moveToFirst() || cursor.getString(0) == "" ||
cursor.getString(0) == null)) {
        TempConocido2 = cursor.getString(0);
    }
    cursor.close();
    if
(TempConocido2.equals(Red.InDispositivos[0].InDispositivo[this.TempI].Name
)) {
        Red.InDispositivos[0].InDispositivo[this.TempI].Name = "(
" + this.TempName + " )";
    }
    MainActivity.NumberOnList--;
    MainActivity.BD.removeDispositivo(this.TempMac);
    MainActivity.ToastHelper = 3;
    if (MainActivity.MyIP.equals("0.0.0.0") ||
!MainActivity.PrimeravezFirstScreen) {
        Intent intent3 = new Intent(getApplicationContext(),
FirstScreen.class);
        intent3.addFlags(536870912);
        startActivity(intent3);
        finish();
        return;
    }
    Intent intent4 = new Intent(this.MyContext, Red.class);
    intent4.addFlags(131072);
    startActivity(intent4);
    finish();
} else if (MainActivity.MyIP.equals("0.0.0.0") ||
!MainActivity.PrimeravezFirstScreen) {
    Intent intent5 = new Intent(getApplicationContext(),
FirstScreen.class);
    intent5.addFlags(536870912);
    startActivity(intent5);
    finish();
} else {
    Intent intent6 = new Intent(this.MyContext, Red.class);
    intent6.addFlags(131072);
    startActivity(intent6);
    finish();
}
}

public void onBackPressed() {
    if (MainActivity.MyIP.equals("0.0.0.0") ||
!MainActivity.PrimeravezFirstScreen) {
        Intent intent = new Intent(getApplicationContext(),
MainActivity.class);
        intent.addFlags(536870912);
        startActivity(intent);
        finish();
        return;
    }
    this.Name.toString().trim().equals("");
    if (this.Name.length() >= 1 && this.QueEs == 2) {
        MainActivity.BD.removeDispositivo(this.TempMac);
        MainActivity.BD.insertDispositivo(this.TempMac,
this.Name.getText().toString());
        Red.InDispositivos[0].InDispositivo[this.TempI].Name =

```

```

this.Name.getText().toString());
        MainActivity.ToastHelper = 1;
    }
    Intent intent2 = new Intent(getApplicationContext(), Red.class);
    intent2.addFlags(131072);
    startActivity(intent2);
    finish();
}
}

```

Code untuk membaca kecepatan *Access Point*

```

public class WiFiConnection {
    public static final int LINK_SPEED_INVALID = -1;
    public static final WiFiConnection EMPTY = new
    WiFiConnection(StringUtils.EMPTY, StringUtils.EMPTY, StringUtils.EMPTY,
    LINK_SPEED_INVALID);

    private final String SSID;
    private final String BSSID;
    private final String ipAddress;
    private final int linkSpeed;

    public WiFiConnection(@NonNull String SSID, @NonNull String BSSID,
    @NonNull String ipAddress, int linkSpeed) {
        this.SSID = SSID;
        this.BSSID = BSSID;
        this.ipAddress = ipAddress;
        this.linkSpeed = linkSpeed;
    }

    @NonNull
    public String getSSID() {
        return SSID;
    }

    @NonNull
    public String getBSSID() {
        return BSSID;
    }

    @NonNull
    public String getIpAddress() {
        return ipAddress;
    }

    public int getLinkSpeed() {
        return linkSpeed;
    }

    public boolean isConnected() {
        return !EMPTY.equals(this);
    }

    @Override
    public boolean equals(Object o) {
        if (this == o) return true;
    }
}

```

```

        if (o == null || getClass() != o.getClass()) return false;

        WiFiConnection that = (WiFiConnection) o;

        return new EqualsBuilder()
            .append(getSSID(), that.getSSID())
            .append(getBSSID(), that.getBSSID())
            .isEquals();
    }

    @Override
    public int hashCode() {
        return new HashCodeBuilder(17, 37)
            .append(getSSID())
            .append(getBSSID())
            .toHashCode();
    }

    @Override
    public String toString() {
        return ToStringBuilder.reflectionToString(this);
    }
}

```

Code untuk Navigation Menu

```

public enum NavigationMenu {
    ACCESS_POINTS(R.drawable.ic_network_wifi,
        R.string.action_access_points, NavigationItemFactory.ACCESS_POINTS,
        NavigationOptionFactory.AP),
    CHANNEL_GRAPH(R.drawable.ic_insert_chart,
        R.string.action_channel_graph, NavigationItemFactory.CHANNEL_GRAPH,
        NavigationOptionFactory.OTHER),
    TIME_GRAPH(R.drawable.ic_show_chart, R.string.action_time_graph,
        NavigationItemFactory.TIME_GRAPH, NavigationOptionFactory.OTHER),
    VENDORS(R.drawable.ic_list_grey, R.string.action_vendors,
        NavigationItemFactory.VENDORS),
    SETTINGS(R.drawable.ic_settings, R.string.action_settings,
        NavigationItemFactory.SETTINGS);

    private final int icon;
    private final int title;
    private final List<NavigationOption> navigationOptions;
    private final NavigationItem navigationItem;

    NavigationMenu(int icon, int title, @NonNull NavigationItem
        navigationItem, @NonNull List<NavigationOption> navigationOptions) {
        this.icon = icon;
        this.title = title;
        this.navigationItem = navigationItem;
        this.navigationOptions = navigationOptions;
    }

    NavigationMenu(int icon, int title, @NonNull NavigationItem
        navigationItem) {

```

```

        this(icon, title, navigationItem, NavigationOptionFactory.OFF);
    }

    public int getTitle() {
        return title;
    }

    public void activateNavigationMenu(@NonNull MainActivity mainActivity,
@NonNull MenuItem menuItem) {
        navigationItem.activate(mainActivity, menuItem, this);
    }

    public void activateOptions(@NonNull MainActivity mainActivity) {
        IterableUtils.forEach(navigationOptions, new
ActivateClosure(mainActivity));
    }

    public boolean isWiFiBandSwitchable() {
        return navigationOptions.contains(NavigationOptionFactory.WIFI_SWITCH_ON);
    }

    public boolean isRegistered() {
        return navigationItem.isRegistered();
    }

    int getIcon() {
        return icon;
    }

    @NonNull
    NavigationItem getNavigationItem() {
        return navigationItem;
    }

    @NonNull
    List<NavigationOption> getNavigationOptions() {
        return navigationOptions;
    }

    private class ActivateClosure implements Closure<NavigationOption> {
        private final MainActivity mainActivity;

        private ActivateClosure(@NonNull MainActivity mainActivity) {
            this.mainActivity = mainActivity;
        }

        @Override
        public void execute(NavigationOption input) {
            input.apply(mainActivity);
        }
    }
}

```