BAB 3
ANALISIS DAN PERANCANGAN

3.1 Analisis Sistem

Analisis sistem adalah tahapan dimana seorang peneliti atau pengembang menganalisis sebuah sistem yang bertujuan untuk mengidentifikasi permasalahan-permasalahan yang terdapat pada sebuah sistem yang sedang berjalan dan menentukan kebutuhan-kebutuhan dari sistem yang akan dibangun. Pada tahap analisis sistem ini terdapat beberapa analisis di dalamnya meliputi analisis masalah, analisis sistem yang akan dibuat, analisis arsitektur sistem, analisis teknologi, analisis kebutuhan non fungsional, dan analisis kebutuhan fungsional.

3.1.1 Analisis Masalah

Analisis masalah adalah penjabaran masalah-masalah yang ditemui dalam penelitian yang dilakukan. Analisis masalah perlu dilakukan sebelum melakukan perancangan sistem dengan tujuan untuk melakukan evaluasi pada prosedur yang telah ada, merumuskan tujuan yang ingin dicapai dengan sistem yang baru serta menyususun rencana pembangunan sistem. Dan nantinya hasil dari analisis masalah ini dapat dijadikan acuan dalam melakukan perancangan sistem yang akan dibangun.

Pada penelitian yang dilakukan yaitu berkaitan dengan *pointing antenna* dan instalasi antena, terdapat beberapa masalah yang umum terjadi dan dialami oleh teknisi pemasang antena. Permasalahan pertama adalah sulitnya melakukan *pointing antenna* dan perhitungan *fresnel zone*. Selain diperlukan perangkat pendukung seperti GPS, rumus yang digunakan dalam melakukan perhitungan *fresnel zone* juga cukup rumit. Teknisi pemasang antena juga harus mengetahui jarak dan elevasi permukaan bumi atau kontur tanah antara dua titik *pointing antenna* tersebut untuk memastikan *pointing antenna* yang dilakukan tidak terhalang oleh elevasi bumi seperti bukit, pegunungan, ataupun objek lainnya. Dari

permasalah ini tidak sedikit membuat para teknisi enggan untuk menghitung *fresnel zone* dan melakukan *pointing antenna* secara asal-asalan. Tentunya hal ini akan berdampak buruk pada komunikasi radio dari dua titik antena tersebut. Sinyal yang dikirimkan oleh satu antena bisa saja tidak diterima dengan baik oleh antena penerima akibat dari adanya objek yang menghalangi atau arah pointing antena yang tidak tepat atau *Line Of Sight (LOS)*. Dengan demikan dari uraian permasalahan yang pertama ini, pada aplikasi yang dibangun nantinya harus mampu mempermudah teknisi dalam melakukan *pointing antenna* dan perhitungan *fresnel zone*. Pada fitur ini juga harus mencakup bagaimana elevasi permukaan bumi atau kontur tanah dari dua titik lokasi antena agar teknisi pemasang antena dapat mengetahui terhalang atau tidaknya *pointing antenna* yang dilakukan.

Permasalahan kedua yang muncul ketika teknisi akan melakukan pointing antenna adalah sulitnya menentukan lokasi tower terdekat untuk pemasangan antena. Tidak seperti mencari ATM terdekat, minimarket terdekat ataupun tempat-tempat umum lainnya yang sudah banyak tersedia pada aplikasi *smartphone* baik itu menggunakan google maps ataupun aplikasi lainnya, mencari tower terdekat untuk memasang antena sangatlah rumit. Jika teknisi tidak memiliki data tower, maka teknisi akan melakukan survey dan mencari langsung ke lapangan untuk menentukan tower instalasi antenna. Tentunya dari permasalahan ini akan menyita cukup banyak waktu dan tidak efisien. Dengan demikian pada aplikasi yang dibangun nantinya harus memiliki fitur menampilkan lokasi tower terdekat dari lokasi pengguna dengan memanfaatkan GPS dan data tower dari OpenCelliD API.

Permasalahan yang terakhir ketika melakukan pointing antenna adalah ketersediaan perangkat pendukung seperti kompas dan GPS yang terbatas. Kompas diperlukan untuk mengarahkan posisi antena ke tower atau antena lawan sesuai dengan perhitungan sudut azimuth. Sedangkan GPS digunakan untuk membantu mendapatkan kordinat dan elevasi permukaan bumi dari lokasi pengguna. Akan tetapi karena ketersediaannya yang terbatas membuat proses *pointing antenna* menjadi terhambat. Untuk itu aplikasi yang dibangun nantinya harus dapat menggantikan keberadaan perangkat tersebut dan tetap memenuhi kebutuhan yang ada.

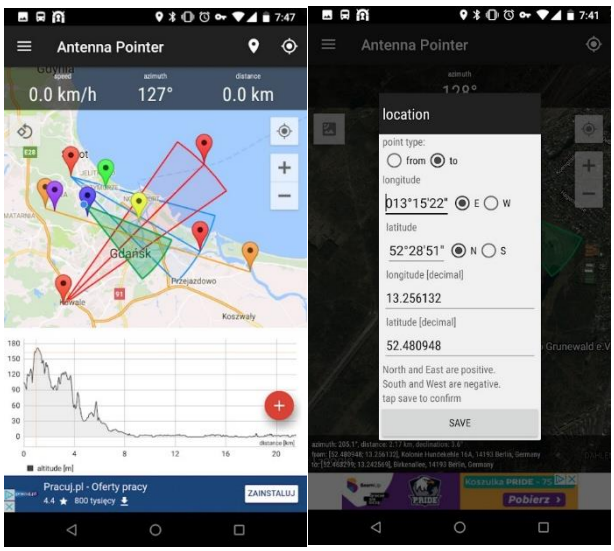
3.1.2 Analisis Aplikasi Sejenis

Analisis aplikasi sejenis bertujuan untuk menganalisis fungsionalitas dan alur dari aplikasi yang telah ada. Hal ini bertujuan untuk memilah fungsionalitas dan alur yang akan diadopsi pada sistem yang diteliti berdasarkan manfaat yang sudah dihasilkan dari aplikasi sejenis dan mencari nilai tambah dari perangkat lunak yang akan dibangun pada penelitian ini. Dengan demikian aplikasi yang dibangun dapat menjadi aplikasi yang lebih baik dan sesuai kebutuhan yang ada.

Pada penelitian ini terdapat dua aplikasi sejenis yang akan diteliti diantaranya adalah Antenna Pointer *by* Maciej Grzegorczyk dan Dish Install, Pointer & Align *by* InfoSofty. Pertimbangan pemilihan kedua aplikasi tersebut dikarenakan kedua aplikasi tersebut merupakan aplikasi *platform mobile Android* yang fitur-fiturnya mendekati kebutuhan dari penelitian dan memiliki jumlah unduhan diatas 100.000. Dan untuk perbandingannya, paramater pembanding yang digunakan mengacu kepada komponen-komponen sistem informasi.

1. Antenna Pointer *by* Maciej Grzegorczyk

   Aplikasi ini merupakan aplikasi untuk *pointing antenna* yang mampu melakukan *multiple pointing* sekaligus dalam satu *maps*. Pada aplikasi ini proses yang dilakukan hanya menghitung sudut azimuth, jarak dan elevasi.



Gambar 3.1 Aplikasi Antenna Pointer by Maciej Grzegorczyk

2. Dish Install, Pointer & Align *by* InfoSofty

   Sama halnya dengan aplikasi Antenna Pointer *by* Maciej Grzegorczyk, aplikasi ini hanya dapat melakukan proses perhitungan sudut azimuth untuk pointing antenna. Yang berbeda dari aplikasi ini adalah memiliki fitur kompas digital.



Gambar 3.2 Aplikasi Dish Install, Pointer & Align by InfoSofty

Berdasarkan fungsi pada masing-masing aplikasi tersebut maka dapat dilakukan analisis perbandingan dari kedua aplikasi diatas dengan menggunakan parameter komponen sistem informasi. Berikut adalah tabel analisis perbandingan aplikasi sejenis.

Tabel 3.1 Analisis Perbandingan Aplikasi Sejenis

Komponen Pembanding	Antenna Pointer *by* Maciej Grzegorczyk	Dish Install, Pointer & Align *by* InfoSofty
Input	Masukan berupa data lokasi *pointing* (kordinat)	Masukan berupa data lokasi *pointing* (kordinat)
Proses	Data masukkan diproses menjadi beberapa fungsionalitas seperti sudut azimuth, jarak, dan elevasi permukaan bumi	Data masukan diproses menjadi beberapa fungsionalitas seperti sudut azimuth, jarak, dan kompas digital

Komponen Pembanding	Antenna Pointer *by* Maciej Grzegorczyk	Dish Install, Pointer & Align *by* InfoSofty
Output	Data sudut azimuth, jarak, visualiasi arah antena, dan grafik elevasi	Data sudut azimuth, jarak, dan visualisasi kompas digital
Teknologi	Teknologi GPS dan Magnetic Field	Teknologi GPS dan Magnetic Field
Software	Aplikasi ini menggunakan sistem operasi android dengan minimal version 4.4 Kitkat atau lebih tinggi	Aplikasi ini menggunakan sistem operasi android dengan minimal version 4.1 Jelly Bean atau lebih tinggi
Hardware	Aplikasi ini menggunakan beberapa perangkat keras yang sudah terpasang default disetiap perangkat android diantaranya: GPS dan sensor magnet	Aplikasi ini menggunakan beberapa perangkat keras yang sudah terpasang default disetiap perangkat android diantaranya: GPS dan sensor magnet
Data	Menggunakan penyimpanan internal pada perangkat mobile	Menggunakan penyimpanan internal pada perangkat mobile
Komponen Kendali	Aplikasi ini akan menampilkan warning untuk melakukan update jika pengguna yang menggunakan aplikasi ini tidak melakukan update secara berkala	Aplikasi ini akan menampilkan warning untuk melakukan update jika pengguna yang menggunakan aplikasi ini tidak melakukan update secara berkala
Kelebihan	Aplikasi ini mendukung menampilkan sudut azimuth dan elevasi permukaan bumi	Aplikasi ini mendukung menampilkan sudut azimuth dan kompas digital
Kekurangan	Aplikasi ini tidak mendukung perhitungan *fresnel zone*, tidak menampilkan tower terdekat, tidak ada kompas, dan *history* instalasi antena	Aplikasi ini tidak mendukung perhitungan *fresnel zone*, tidak menampilkan tower terdekat, tidak ada perhitungan elevasi, dan *history* instalasi antena

Dari hasil analisis kedua aplikasi sejenis, aplikasi yang dibangun akan mengadopsi dan memiliki beberapa fungsionalitas diantaranya:

1. Menampilkan elevasi permukaan bumi antara dua titik pointing antenna
2. Manampilkan kompas sebagai dari implementasi sudut azimuth

Berdasarkan analisis tersebut juga dapat disimpulkan keunikan atau nilai tambah dari perangkat lunak yang dibangun dibandingkan dengan aplikasi sejenis adalah sebagai berikut:

1. Menghitung *fresnel zone* dan mengintegrasikannya dengan grafik elevasi permukaan bumi agar mengetahui objek yang menghalangi
2. Manampilkan tower terdekat pada maps sesuai lokasi pengguna
3. *History* instalasi antena untuk menyimpan data hasil instalasi antena

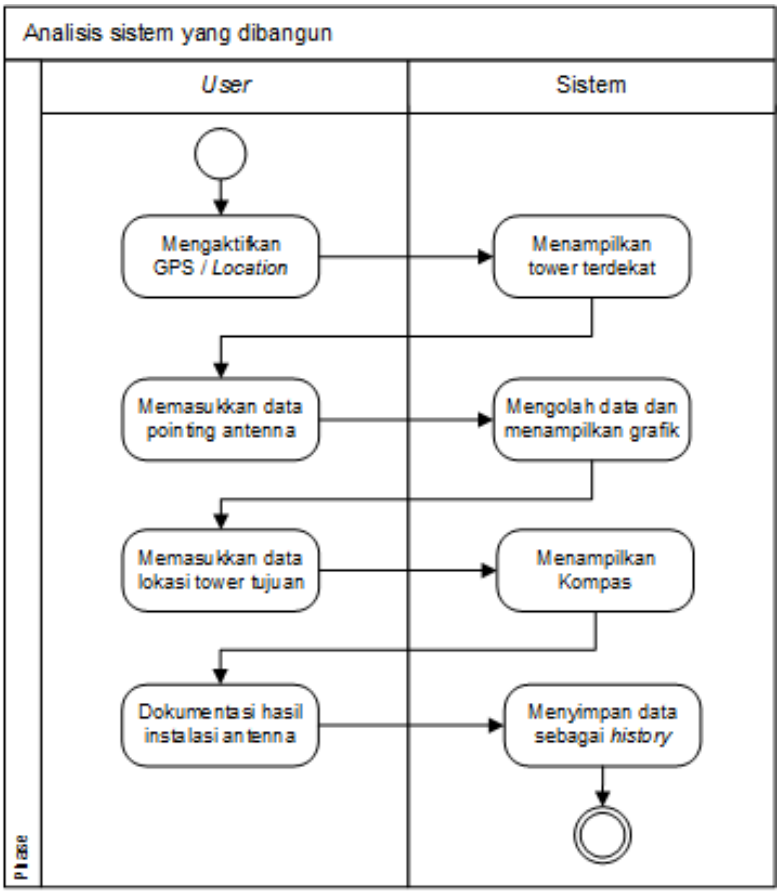
3.1.3 Analisis Sistem yang Dibangun

Analisis sistem yang dibangun merupakan gambaran lengkap sistem yang akan di bangun. Analisis sistem yang dibangun ini juga didasarkan pada analisis masalah dan analisis aplikasi sejenis yang sudah dilakukan sebelumnya. Sehingga dari kedua analisis tersebut akan didapatkan gambaran aplikasi yang akan dibangun berikut dengan prosedur dari aplikasi yang dibangun.

Pada penelitian ini aplikasi yang dibangun harus memenuhi kebutuhan dan solusi dari masalah yang ada yaitu sulitnya melakukan *pointing antenna* dan perhitungan *fresnel zone*, sulitnya mencari tower terdekat dan keberadaan perangkat pendukung (kompas dan GPS) yang terbatas. Dari ketiga masalah utama tersebut dan berdasarkan analisis aplikasi sejenis maka gambaran aplikasi yang dibangun akan memiliki fitur *pointing antenna* untuk menghitung *fresnel zone*, fitur tower terdekat dan fitur kompas digital. Dan untuk tambahannya penulis bermaksud untuk menambahkan fitur *history* instalasi antenna agar pengguna dapat menyimpan dan melakukan dokumentasi dari instalasi antena yang sudah dilakukan. Selain itu agar lebih aman fitur *history* instalasi antenna ini juga akan dilengkapi dengan fitur *login* dan daftar yang sudah terintegrasi.

Adapun prosedur dari analisis sistem yang di bangun adalah sebagai berikut yang merupakan gambaran umum dari fitur-fitur sistem yang dibangun:

1. *User* membuka aplikasi dan mengaktifkan GPS atau *location*
2. Sistem menampilkan lokasi tower terdekat
3. *User* memasukan data pointing antenna
4. Sistem mengolah data pointing dan menampilkan dalam bentuk grafik
5. *User* memasukan data lokasi tower tujuan untuk menggunakan kompas
6. Sistem menampilkan kompas yang mengarah ke lokasi tower tujuan
7. *User* mengambil foto hasil instalasi antenna
8. Sistem menyimpannya sebagai data histori instalasi antena



Gambar 3.3 Prosedur Sistem Yang Dibangun

3.1.3.1 Analisis Perhitungan *Fresnel Zone*

Pada aplikasi yang dibangun terdapat proses untuk mengolah *fresnel zone* dari pointing antenna yang dilakukan dan menampilkannya dalam bentuk grafik. Untuk analisis perhitungan dari *fresnel zone* ini dapat menggunakan rumus standar perhitungan *fresnel zone* yang sudah ada, sebagai contoh bagaimanakah *fresnel zone* dari pointing antenna dengan data-data pointing sebagai berikut:

Jarak antara titik A ke titik B = 8 Km
Frekuensi dari antena = 2.4 GHz
Ketinggian tower titik A = 70 m
Ketinggian tower titik B = 100 m

Dari contoh soal diatas, hal pertama yang perlu dilakukan adalah menghitung *wavelength* (panjang gelombang) terlebih dahulu, dengan rumus sebagai berikut:

$$\lambda = \frac{c}{f}$$

c adalah kecepatan cahaya dalam satuan meter per second yaitu $299.792.10^6$ atau lebih tepatnya 299792458. Sedangkan f adalah nilai frekuensi dalam satuan Hz, jika f = 2.4 GHz maka dalam satuan Hz adalah 2400000000. Jadi nilai λ adalah:

λ = 299792458 / 2400000000
λ = 0.12491352416666666
λ = 0.125 (dibulatkan)

Sebagaimana kita ketahui untuk menghitung radius *fresnel zone* menggunakan rumus:

$$R = \sqrt{\frac{\lambda d_1 d_2}{d_1 + d_2}}$$

Tapi pada kasus ini akan dipersempit dengan bantuan *samples* untuk memecah jarak ke beberapa bagian. Misalnya *samples* yang digunakan adalah 10, sehingga akan terjadi 10 kali perhitungan radius *fresnel zone*.

Sebagaimana sudah dijelaskan pada bab sebelumnya mengenai perhitungan LOS clearance, $C_{LOS,}$ dapat dihitung dengan rumus:

$$C_{LOS} = h_1 + \frac{h2 - h1}{d} . d_1 - h_{er} - ho$$

Karena dalam penelitian ini menggunakan *samples* maka h_{er} atau ketinggian *effective radius* tidak digunakan.

$$C_{LOS} = h1 + \frac{h2 - h1}{d} . d_1 - ho$$

Kemudian untuk menghitung ketinggian LOS di suatu titik samples adalah penjumlahan dari C_{LOS} dengan ho:

$$H_{LOS} = C_{LOS} + ho$$

Sehingga jika didefinisikan dengan detail rumusnya menjadi:

$$H_{LOS} = h1 + \frac{h2 - h1}{d} . d_1$$

Rumus tersebut disubstitusikan kembali dengan *samples*:

$$H_{LOS} = h_1 + \frac{h2 - h1}{s} . s_i$$

Keterangan:

H_{LOS} = ketinggian di suatu titik LOS
h1 = ketinggian antena di titik tower A (m)
h2 = ketinggian antena di titik tower B (m)
s = banyaknya sampel yang digunakan
s_i = sampel ke-i

Untuk lebih jelasnya berikut adalah perhitungan radius *fresnel zone* dan LOS (*Line Of Sight)* di setiap titik sampel yang sebanyak 10 kali sampel:

Sampel ke-1:

Jarak = 800 m, sehingga nilai d1 = 800 m dan nilai d2 = 7200 m

R = √ (0.125 x 800 x 7200) / (800 + 7200)

R = 9.487

LOS = 70 + (100 – 70) / 10 x 1

LOS = 73

Sampel ke-2:

Jarak = 1600 m, sehingga nilai d1 = 1600 m dan nilai d2 = 6400 m

R = √ (0.125 x 1600 x 6400) / (1600 + 6400)

R = 12.649

LOS = 70 + (100 – 70) / 10 x 2

LOS = 76

Sampel ke-3:

Jarak = 2400 m, sehingga nilai d1 = 2400 m dan nilai d2 = 5600 m

R = √ (0.125 x 2400 x 5600) / (2400 + 5600)

R = 14.491

LOS = 70 + (100 – 70) / 10 x 3

LOS = 79

Sampel ke-4:

Jarak = 3200 m, sehingga nilai d1 = 3200 m dan nilai d2 = 4800 m

R = √ (0.125 x 3200 x 4800) / (3200 + 4800)

R = 15.492

LOS = 70 + (100 – 70) / 10 x 4

LOS = 82

Sampel ke-5:

Jarak = 4000 m, sehingga nilai d1 = 4000 m dan nilai d2 = 4000 m

R = √ (0.125 x 4000 x 4000) / (4000 + 4000)

R = 15.811

LOS = 70 + (100 – 70) / 10 x 5

LOS = 85

Sampel ke-6:

Jarak = 4800 m, sehingga nilai d1 = 4800 m dan nilai d2 = 3200 m

R = √ (0.125 x 4800 x 3200) / (4800 + 3200)

R = 15.492

LOS = 70 + (100 – 70) / 10 x 6

LOS = 88

Sampel ke-7:

Jarak = 5600 m, sehingga nilai d1 = 5600 m dan nilai d2 = 2400 m

R = √ (0.125 x 5600 x 2400) / (5600 + 2400)

R = 14.491

LOS = 70 + (100 – 70) / 10 x 7

LOS = 91

Sampel ke-8:

Jarak = 6400 m, sehingga nilai d1 = 6400 m dan nilai d2 = 1600 m

R = √ (0.125 x 6400 x 1600) / (6400 + 1600)

R = 12.649

LOS = 70 + (100 – 70) / 10 x 8

LOS = 94

Sampel ke-9:

Jarak = 7200 m, sehingga nilai d1 = 7200 m dan nilai d2 = 800 m

R = √ (0.125 x 7200 x 800) / (7200 + 800)

R = 9.487

LOS = 70 + (100 – 70) / 10 x 9

LOS = 97

Sampel ke-10:

Jarak = 8000 m, sehingga nilai d1 = 8000 m dan nilai d2 = 0 m

R = √ (0.125 x 8000 x 0) / (8000 + 0)

R = 0

LOS = 70 + (100 – 70) / 10 x 10

LOS = 100

Dari hasil perhitungan radius *fresnel zone* tersebut kemudian diolah kembali untuk menghitung batas atas dan batas bawah dari *fresnel zone*. Rumus dari perhitungan batas atas *fresnel zone* atau *top fresnel zone* adalah sebagai berikut:

$$Top\ Fresnel\ Zone = LOS\ +\ R$$

Sedangkan untuk menghitung batas bawah *fresnel zone* atau *bottom fresnel zone* adalah sebagai berikut:

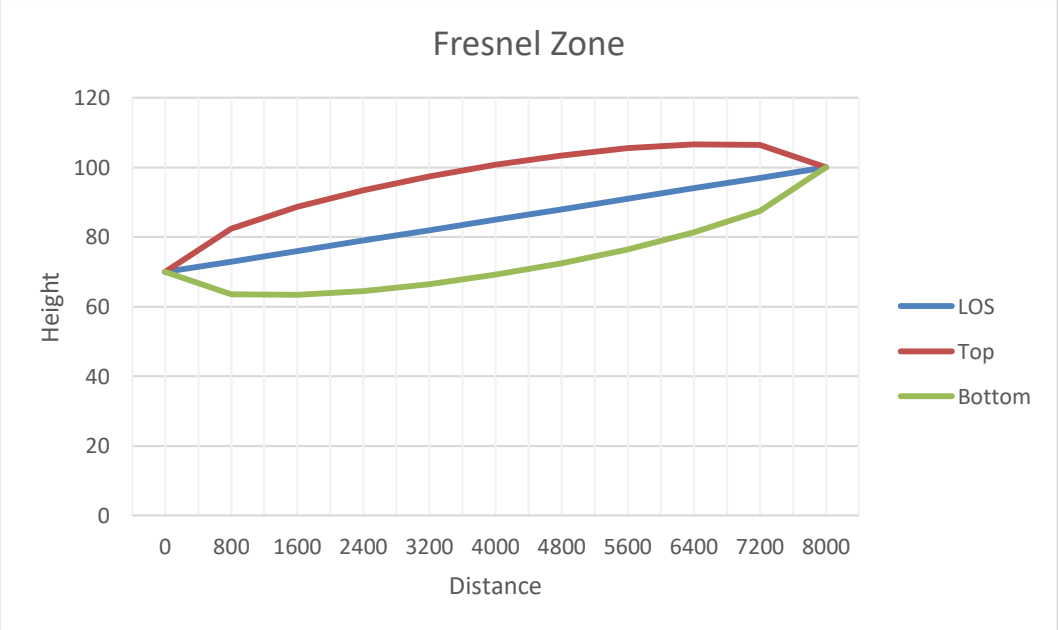
$$Bottom\ Fresnel\ Zone = LOS -\ R$$

Dari kedua rumus diatas tersebut untuk lebih memudahkan perhitungan batas atas (*Top Fresnel Zone*) dan batas bawah (*Bottom Fresnel Zone*) sebaiknya dibuat tabel yang berkolerasi dengan LOS yang sudah dihitung juga. Berikut adalah tabel perhitungan *top fresnel zone* dan *bottom fresnel zone* sesuai sampel:

Tabel 3.2 Contoh Perhitungan Fresnel Zone

Sampel	Jarak (m)	Radius (R)	Direct (LOS)	Top Fresnel Zone (LOS + R)	Bottom Fresenel Zone (LOS – R)
1	800	9.487	73	82.487	63.513
2	1600	12.649	76	88.649	63.351
3	2400	14.491	79	93.491	64.509
4	3200	15.491	82	97.491	66.509
5	4000	15.811	85	100.811	69.189
6	4800	15.491	88	103.491	72.509
7	5600	14.491	91	105.491	76.509
8	6400	12.649	94	106.649	81.351
9	7200	9.487	97	106.487	87.513
10	8000	0	100	100	100

Jika direpresentasikan dalam bentuk grafik maka tampilannya adalah seperti Gambar 3.4. Pada gambar tersebut area *fresnel zone* terlihat sedikit kaku karena sampel yang digunakan hanya sedikit, yaitu 10. Semakin banyak sampel yang digunakan pada perhitungan *fresnel zone*, maka grafik dan area dari *fresnel zone* akan semakin akurat.



Gambar 3.4 Grafik Perhitungan Fresnel Zone

3.1.4 Analisis Teknologi

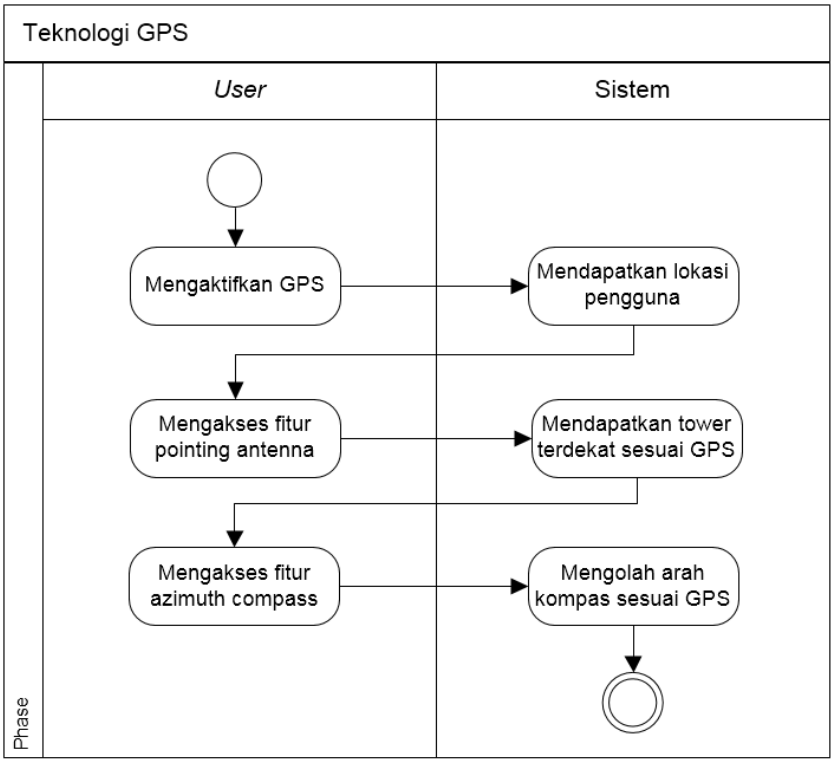
Analisis teknologi bertujuan untuk mengetahui teknologi apa saja yang akan digunakan, cara kerjanya, dan bagaimana penggunaannya pada sistem yang dibangun. Adapun analisis teknologi yang digunakan pada sistem yang dibangun antara lain:

3.1.4.1 GPS

Pada penelitian ini untuk memenuhi kebutuhan fungsional aplikasi dalam menentukan lokasi tower terdekat dan mengarahkan antena ke titik *pointing* tower lawan yang dituju diperlukan penggunaan teknologi GPS. Dengan penggunaan teknologi GPS dapat membantu sistem yang dibangun dalam mendapatkan *current location* atau koordinat dari lokasi pengguna. Dari koordinat yang sudah didapat tersebut kemudian akan diolah untuk menjalankan fungsi tower terdekat dan kompas pada aplikasi yang dibangun.

Cara kerja atau penggunaan GPS pada aplikasi yang dibangun ini pada hakikatnya sama dengan penggunaan GPS pada umumnya di aplikasi lain yang sudah ada seperti google maps. Berikut adalah cara kerja dari GPS pada aplikasi yang dibangun:

1. Pengguna mengaktifkan GPS terlebih dahulu sebelum menjalankan aplikasi *antenna pointer*
2. Ketika mengakses menu *pointing antenna* atau menu kompas, aplikasi mulai memanfaatkan GPS untuk mendapatkan lokasi dari pengguna
3. Aplikasi kemudian menampilkan lokasi pengguna dari GPS tersebut pada maps dalam menu *pointing antenna* atau kompas
4. Pada menu *pointing antenna*, lokasi pengguna dari GPS juga akan dimanfaatkan untuk memproses data tower terdekat dari pengguna
5. Pada menu kompas, lokasi pengguna dari GPS akan dimanfaatkan sebagai titik sumber dari arah kompas dan menghitung jarak lokasi pengguna dengan titik tujuan



Gambar 3.5 Skema Penggunaan Teknologi GPS

Penggunaan teknologi GPS pada sistem yang dibangun ini sangatlah penting karena jika GPS pada perangkat *smartphone* pengguna tidak aktif maka beberapa fungsi atau fitur pada sistem yang dibangun tidak dapat berjalan dengan baik. Diantaranya adalah fitur tower terdekat dan kompas digital yang sangat bergantung pada teknologi tersebut. Dengan demikian diharapkan perangkat *smartphone* yang digunakan oleh pengguna sudah memiliki dukungan terhadap teknologi GPS dan sudah dalam keadaan aktif ketika menjalankan aplikasi yang dibangun.

3.1.4.2 Google Maps API

Pada penelitian ini Google Maps API akan digunakan di dalam sistem yang dibangun. Pemilihan dan penggunaan Google Maps API ini berdasarkan pada fungsionalitas sistem yang akan dibangun untuk menyediakan kemudahan proses pointing antenna dan perhitungan *fresnel zone*. Dengan penggunaan Google Maps

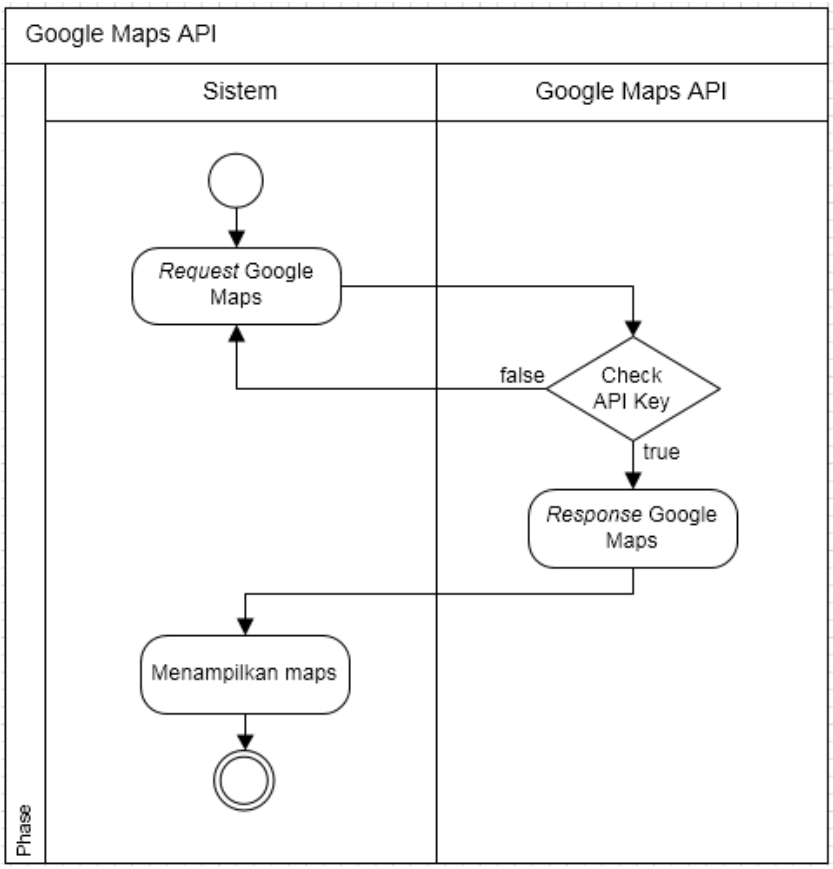
API ini pengguna dapat melihat langsung lokasi tower terdekat pada peta dan menentukan titik pointing antenna dengan cara *drop pin marker* pada peta.

Cara kerja dan penggunaan dari Google Maps API ini cukup mudah, karena IDE yang digunakan oleh peneliti adalah *Android Studio* maka *activity* atau *template* dari google maps ini sudah tersedia dan dapat langsung digunakan. Untuk mengintegrasikannya dengan sistem yang dibangun cukup menyematkan API Key yang sudah disediakan oleh Google. Adapun untuk lebih jelasnya berikut adalah tahapan penggunaan dari Google Maps API:

1. Dapatkan API Key terlebih dahulu. Key ini berfungsi sebagai izin akses terhadap penggunaan layanan Google Maps API. Untuk mendapatkan Google Maps API Key ini dapat langsung mengakses URL https://developers.google.com/maps/documentation/android-api/signup pada browser, kemudian ikuti setiap langkah yang tertera.
2. Menyiapkan *activity* atau *template* google maps. Secara default Android Studio sudah memiliki *template* untuk *activity* Google Maps dan dapat ditambahkan langsung pada project antenna pointer yang dibangun.
3. Menyematkan Google Maps API Key yang sudah didapat pada file google_maps_api.xml dalam project. Jika key tidak sesuai atau tidak disematkan maka maps tidak akan muncul pada aplikasi. Berikut adalah contoh code pada file google_maps_api.xml

```
<string name="google_maps_key" templateMergeStrategy
="preserve" translatable="false"> API_KEY </string>
```

4. Test *run* aplikasi untuk memastikan bahwa default setting dan tampilan peta dari google maps berhasil dimuat pada aplikasi



Gambar 3.6 Skema Penggunaan Teknologi Google Maps

Penggunaan Google Maps API pada sistem yang dibangun ini sangatlah penting karena tanpa adanya Google Maps beberapa fungsi atau fitur pada sistem yang dibangun tidak dapat berjalan dengan baik. Diantaranya adalah fitur tower terdekat, pointing antenna, dan kompas digital yang sangat bergantung pada teknologi tersebut. Dan untuk penggunaan Google Maps ini memerlukan koneksi *internet* untuk memuat gambar-gambar peta digital berikut degan berbagai *object maps* lainnya. Dengan demikian diharapkan pengguna dari aplikasi yang dibangun memiliki dukungan koneksi *internet* agar layanan Google Maps API ini dapat berjalan baik.

3.1.4.3 Google Elevation API

Slain Google Maps API, pada penelitian ini Google Elevation API juga akan digunakan di dalam sistem yang dibangun. Dan seperti halnya Google Maps API, pemilihan dan penggunaan Google Elevation API ini berdasarkan pada fungsionalitas sistem yang akan dibangun untuk menyediakan kemudahan proses pointing antenna dan perhitungan *fresnel zone*. Dengan penggunaan Google Elevation API ini pengguna dapat melihat elevasi permukaan bumi atau kontur tanah dalam bentuk grafik ketika melakukan perhitungan *fresnel zone*. Grafik tersebut merupakan *output* dari sistem yang dibangun yang merupakan representasi dari data-data perhitungan *fresnel zone* dan elevasi permukaan bumi.

Cara kerja dari Google Elevation API ini cukup mudah, aplikasi yang dibangun yaitu aplikasi *antenna pointer* cukup memberikan data koordinat dari dua titik tower *pointing antenna* ke URL Google Elevation API. Data kedua koordinat tersebut didapatkan dari *input* pengguna dengan cara *drop pin* pada maps. Kemudian Google Elevation API akan memberikan *response* data-data elevasi permukaan bumi sepanjang jalur dua titik koordinat tersebut dalam bentuk JSON. Data yang sudah didapatkan diolah lebih lanjut oleh aplikasi *antenna pointer* untuk ditampilkan dalam grafik agar pengguna dapat melihat langsung hasilnya. Sedangkan penggunaannya didalam aplikasi adalah sebagai berikut:

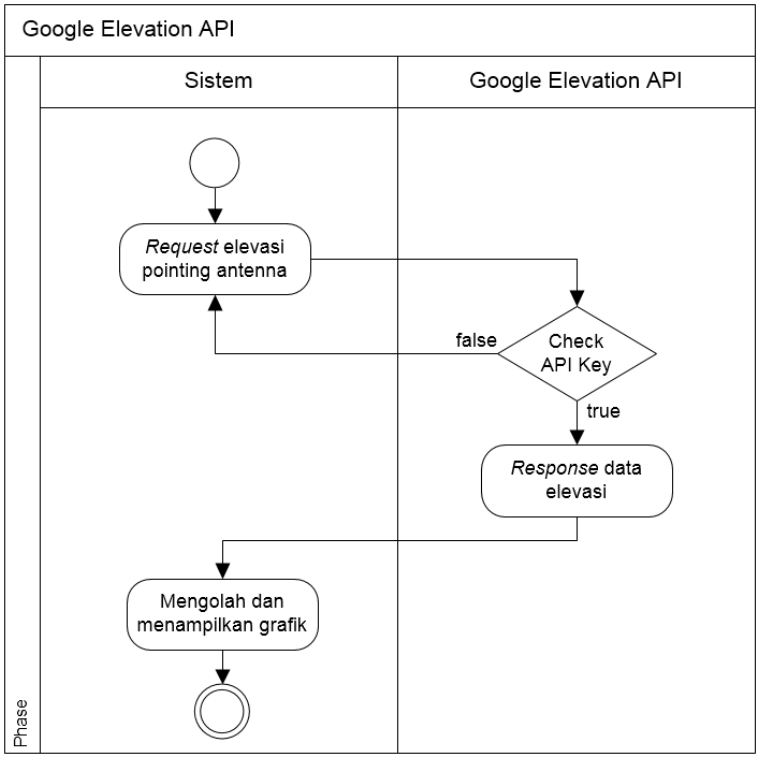
1. Dapatkan API Key terlebih dahulu. Jika sebelumnya sudah mendapatkan Key pada Google Maps API maka Key tersebut masih dapat digunakan.
2. Akses Google Elevation API dengan contoh URL berikut

```
https://maps.googleapis.com/maps/api/elevation/json?pat
h=36.5785,-118.2919|36.23998,-116.8317&samples=5&key=
YOUR_API_KEY
```

3. Berikut adalah contoh response dari Google Elevation API

```
{ "results": [{
         "elevation": 4411.94189453125,
         "location": {
            "lat": 36.578581,
            "lng": -118.291994},
         "resolution": 19.08790397644043}, {
         "elevation": -84.51690673828125,
         "location": {
            "lat": 36.23998,
            "lng": -116.83171},
         "resolution": 9.543951988220215}],
   "status": "OK" }
```


Penggunaan Google Elevation API pada sistem yang dibangun ini sangatlah penting karena tanpa adanya API ini fungsi atau fitur pointing antenna tidak dapat berjalan dengan baik. Dan untuk penggunaan dari Google Elevation API ini memerlukan koneksi *internet* untuk memuat data elevasi permukaan bumi. Dengan demikian diharapkan pengguna dari aplikasi antenna pointer ini memiliki dukungan koneksi *internet* agar layanan Google Elevation ini dapat berjalan baik.



Gambar 3.7 Skema Penggunaan Teknologi Google Elevation

3.1.4.4 OpenCelliD API

Pada penelitian ini OpenCelliD API akan digunakan di dalam sistem yang dibangun. Pemilihan dan penggunaan OpenCelliD API ini berdasarkan pada fungsionalitas sistem yang akan dibangun untuk menyediakan informasi tower terdekat pada saat pointing antenna. Dengan penggunaan OpenCelliD API ini aplikasi yang dibangun dapat menyediakan data tower-tower terdekat yang kemudian ditampilkan dalam bentuk marker di google maps.

Cara kerja dan penggunaan dari OpenCelliD API ini sebetulnya berkaitan dengan *platform web* yang difungsikan sebagai API pada sistem yang dibangun, berikut adalah detail penggunaannya:

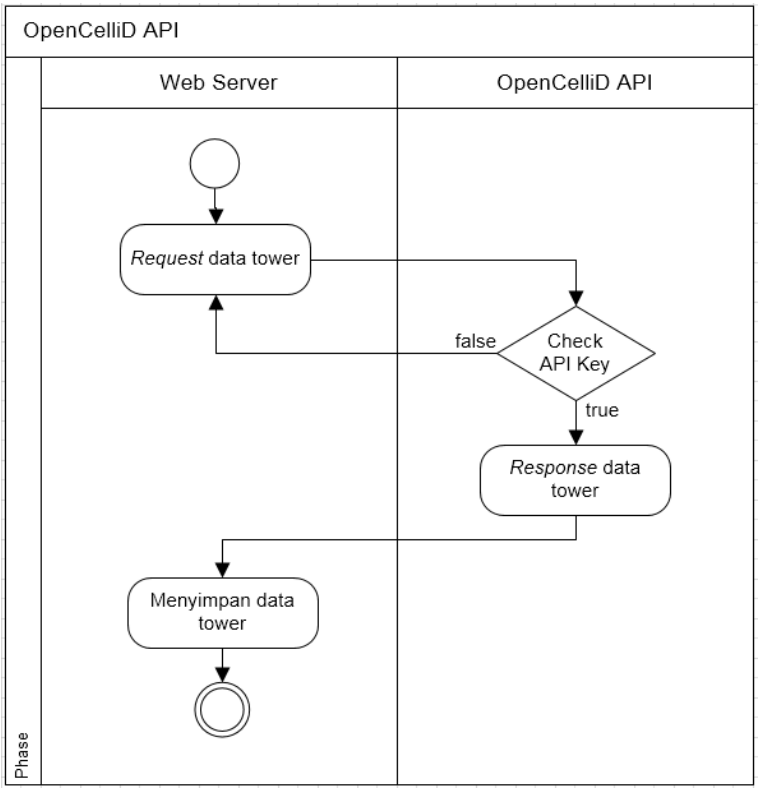
1. Dapatkan API Key pada website https://www.opencellid.org untuk izin akses API.
2. Buat post request untuk meminta data lokasi tower

```
$curl = curl_init();
curl_setopt_array($curl, array(
  CURLOPT_URL => "https://us1.unwiredlabs.com/v2/process.php",
  CURLOPT_RETURNTRANSFER => true,
  CURLOPT_ENCODING => "",
  CURLOPT_MAXREDIRS => 10,
  CURLOPT_TIMEOUT => 30,
  CURLOPT_HTTP_VERSION => CURL_HTTP_VERSION_1_1,
  CURLOPT_CUSTOMREQUEST => "POST",
  CURLOPT_POSTFIELDS => "{\"token\":
\"your_API_token\",\"radio\": \"gsm\",\"mcc\": 310,\"mnc\":
410,\"cells\": [{\"lac\": 7033,\"cid\": 17811}],\"wifi\":
[{\"bssid\": \"00:17:c5:cd:ca:aa\",\"channel\":
11,\"frequency\": 2412,\"signal\": -51}, {\"bssid\":
\"d8:97:ba:c2:f0:5a\"}],\"address\": 1}",
));
$response = curl_exec($curl);
$err = curl_error($curl);
curl_close($curl);
```

3. Berikut adalah contoh respon dari OpenCelliD API

```
{"status": "ok",
 "balance": 100,
 "lat": 39.573726,
 "lon": -105.005387,
 "accuracy": 730,
 "address": "1289-1425 West Mineral Avenue, Littleton,
CO 80120, USA"}
```

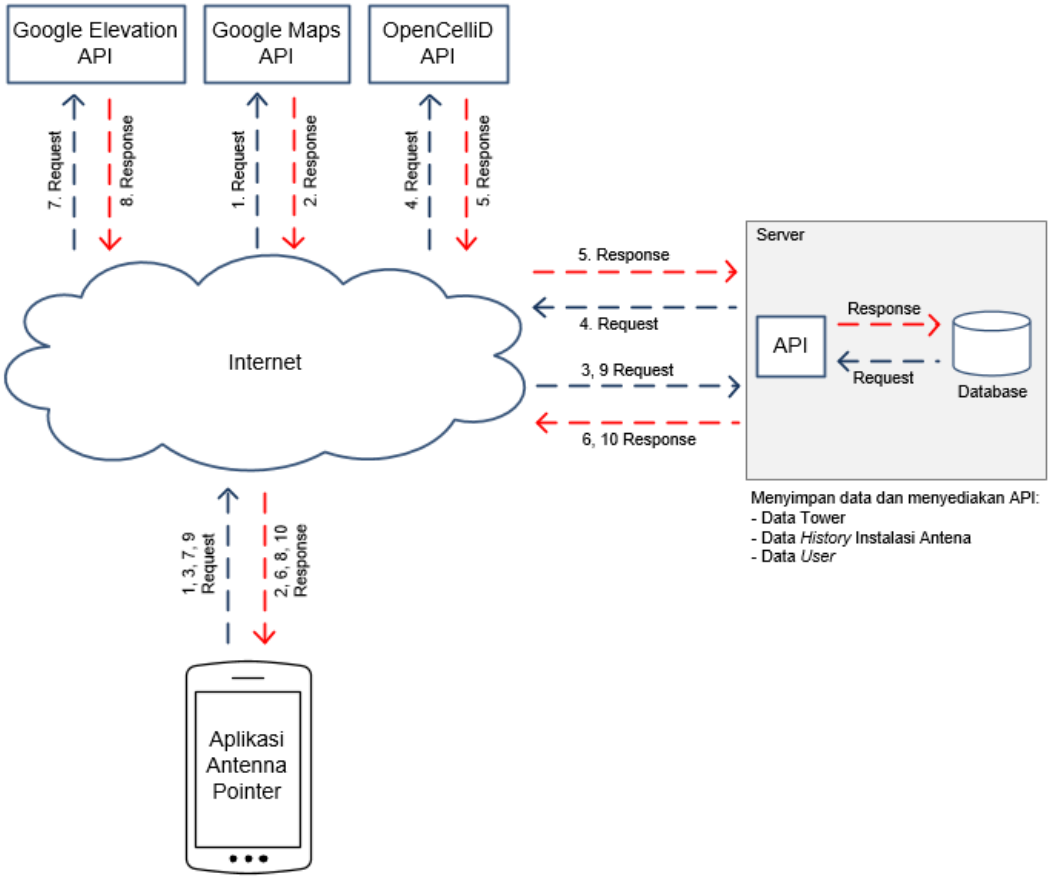
Proses *request* untuk OpenCelliD API ini dilakukan oleh *web server* karena OpenCelliD hanya mendukung satu lokasi tower per *request*. Untuk itu data lokasi tower yang didapatkan dari OpenCelliD disimpan terlebih dahulu di *web server*. Kemudian ketika *platform mobile* melakukan request data tower terdekat maka *web server* yang sebagai API meberikan *response* beberapa data lokasi tower sekaligus.



Gambar 3.8 Skema Penggunaan Teknologi OpenCelliD API

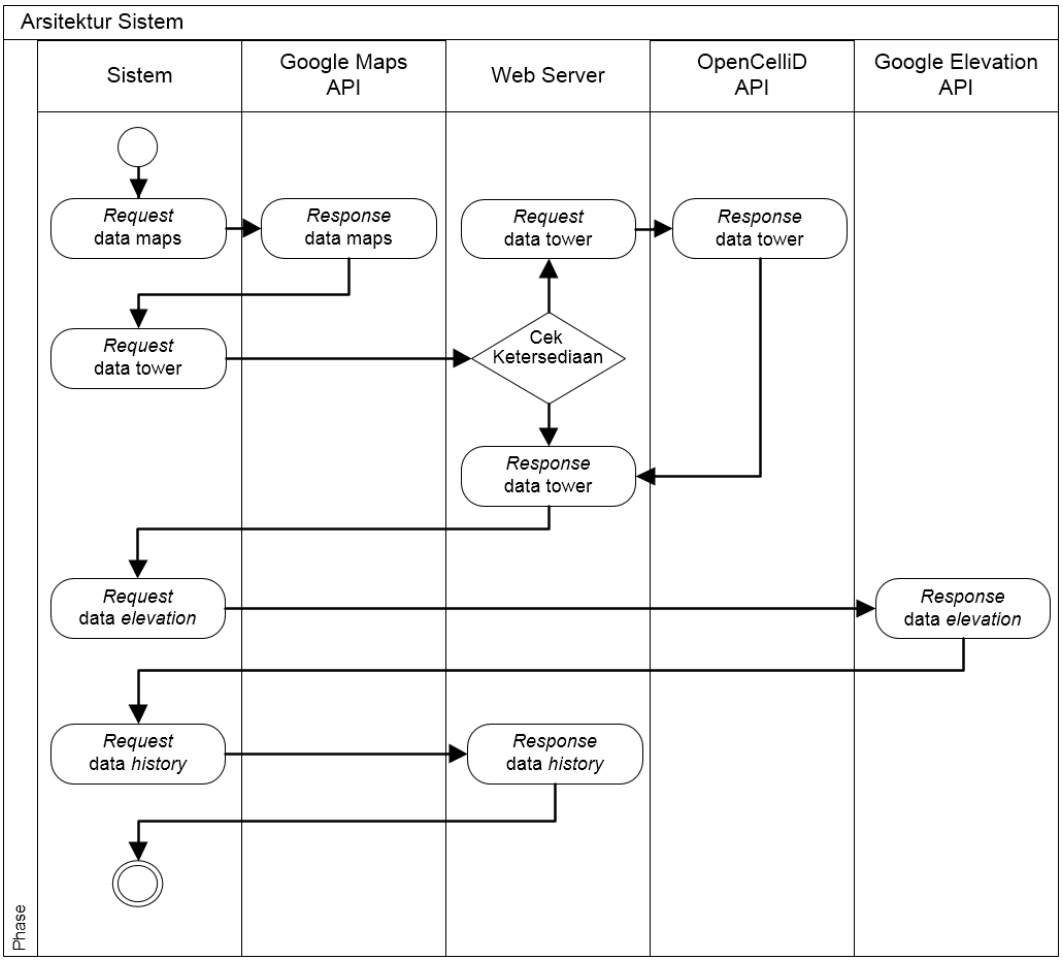
3.1.5 Analisis Arsitektur Sistem

Analisis arsitektur sistem bertujuan untuk mengidentifikasi arsitektur sistem yang dibangun. Pada penelitian ini arsitektur sistem yang dibangun menggunakan *platform mobile* dan *web* yang difungsikan sebagai API. *Platform mobile* merupakan sistem utama dimana aplikasi antenna pointer akan dipasang. Sedangkan untuk *platform web* peneliti menyediakan sebuah *web server* yang berfungsi untuk menyimpan data dan menyediakan API untuk kebutuhan data *history* instalasi antena, data *user* dan data tower yang didapat dari API OpenCelliD. Untuk lebih jelasnya gambar berikut adalah arsitektur sistem secara keseluruhan:



Gambar 3.9 Arsitektur Sistem yang Dibangun

Dari gambar arsitektur sistem yang dibangun tersebut agar lebih mudah dipahami dapat dilihat skema berikut:



Gambar 3.10 Skema Arsitektur Sistem yang Dibangun

Berikut adalah deskripsi dari arsitektur sistem yang dibangun:

1. Perangkat *mobile* melakukan *request* ke Google Maps API untuk mendapatkan *maps*
2. Google Maps API memberikan *response* kepada perangkat *mobile* yang kemudian perangkat mobile menampilkan *maps* tersebut
3. Perangkat *mobile* melakukan *request* ke *web server* untuk mendapatkan data tower terdekat

4. *Web server* melakukan pengecekan ketersediaan tower terdekat dari lokasi pengguna. Jika tidak ditemukan tower terdekat *web server* akan melakukan *update* data tower ke OpenCelliD API.
5. OpenCelliD API memberikan *reponse* balikan kepada *web server* berupa data tower terbaru
6. *Web server* memberikan *response* data tower kepada perangkat *mobile* yang kemudian perangkat *mobile* menampilkan tower terdekat pada maps
7. Perangkat *mobile* melakukan *request* ke Google Elevation API untuk mendapatkan elevasi permukaan bumi antara dua titik lokasi tower pointing
8. Google Elevation API memberikan *response* data elevasi permukaan bumi yang kemudian akan diolah dan ditampilkan oleh perangkat *mobile*
9. Perangkat *mobile* melakukan *request* ke *web server* untuk mendapatkan data *history* instalasi antenna
10. *Web server* memberikan *response* data *history* kepada perangkat *mobile* untuk ditampilkan kepada pengguna.

Berdasarkan pemaparan arsitektur sistem yang dibangun baik dalam bentuk gambar ataupun tahapan-tahapan dalam skema dapat diketahui bahwa sistem yang dibangun ini bergantung pada penggunaan API. Terdapat tiga jenis API yang sudah tersedia dan bersifat *public* yang digunakan dalam aplikasi ini yaitu Google Maps API, Google Elevation API, dan OpenCelliD API. Selain itu terdapat satu API tambahan dalam *web server* yang peneliti buat untuk menyimpan dan menyediakan API data *history* instalasi antena, data user dan data tower. Data tower tidak diakses langsung dari perangkat mobile ke OpenCelliD API melainkan disimpan dalam *web server* terlebih dahulu dikarenakan *response* dari OpenCelliD API tidak mendukung *multiple* data tower. Sehingga dalam kaitannya agar aplikasi dapat memenuhi kebutuhan fungsi tower terdekat dari lokasi pengguna diperlukan *web server* khusus yang mengolah data dari OpenCelliD API tersebut.

3.1.6 Analisis Spesifikasi Kebutuhan Perangkat Lunak

Spesifikasi kebutuhan perangkat lunak dari sistem yang dibangun terbagi menjadi dua yaitu spesifikasi kebutuhan fungsional dan spesifikasi kebutuhan non fungsional. Spesifikasi kebutuhan fungsional akan menjabarkan layanan sistem yang harus disediakan. Berikut adalah spesifikasi kebutuhan fungsional:

Tabel 3.3 Spesifikasi Kebutuhan Fungsional

Kode SKPL	Spesifikasi Kebutuhan Perangkat Lunak
SKPL-F-001	Sistem *mobile* menyediakan fasilitas bagi *user* untuk melihat lokasi tower terdekat
SKPL-F-002	Sistem *mobile* menyediakan fasilitas bagi *user* untuk melakukan *pointing antenna* dan mendapatkan hasil perhitungan *fresnel zone* dalam bentuk grafik
SKPL-F-003	Sistem *mobile* menyediakan fasilitas bagi *user* untuk melihat kompas sesuai dengan arah lokasi tower tujuan
SKPL-F-004	Sistem *mobile* menyediakan fasilitas bagi *user* untuk daftar atau registrasi user baru
SKPL-F-005	Sistem *mobile* menyediakan fasilitas bagi *user* untuk *login* kedalam sistem
SKPL-F-006	Sistem *mobile* menyediakan fasilitas bagi *user* untuk *reset password* jika *user* lupa password login
SKPL-F-007	Sistem *mobile* menyediakan fasilitas bagi *user* untuk melihat *history* instalasi antena
SKPL-F-008	Sistem *mobile* menyediakan fasilitas bagi *user* untuk menambah *history* instalasi antena
SKPL-F-009	Sistem *mobile* menyediakan fasilitas bagi *user* untuk menghapus *history* instalasi antena

Selain spesifikasi kebutuhan perangkat lunak, terdapat spesifikasi kebutuhan non fungsional. Spesifikasi kebutuhan non fungsional akan memberikan batasan layanan yang ditawarkan sistem seperti batasan waktu, batasan pengembangan proses standarisasi dan lain-lain. Adapun spesifikasi kebutuhan non fungsional dapat dilihat pada tabel berikut:

Tabel 3.4 Spesifikasi Kebutuhan Non Fungsional

Kode SKPL	Spesifikasi Kebutuhan Perangkat Lunak
SKPL-NF-001	Sistem yang dibangun memiliki dua subsistem yaitu *mobile* dan *web*
SKPL-NF-002	Sistem yang dibangun minimal menggunakan sistem operasi android versi 5.0 Lolipop
SKPL-NF-003	Sistem yang dibangun memanfaatkan teknologi GPS
SKPL-NF-004	Sistem yang dibangun memanfaatkan teknologi *magnetic field*
SKPL-NF-005	Sistem yang dibangun memanfaatkan koneksi *internet* pada *smartphone*
SKPL-NF-006	Sistem dibangun dengan spesifikasi *hardware* yang memenuhi standar minimum kebutuhan
SKPL-NF-007	Perangkat lunak yang dibangun harus menghasilkan format file standard (.json) yang bisa digunakan oleh pihak luar yang berkepentingan.

3.1.6 Analisis Kebutuhan Non Fungsional

Analisis non-fungsional merupakan analisis yang dibutuhkan untuk menentukan spesifikasi kebutuhan sistem. Spesifikasi ini juga meliputi elemen atau komponen-komponen apa saja yang dibutuhkan untuk sistem yang akan dibangun sampai dengan sistem tersebut diimplementasikan. Analisis kebutuhan ini juga menentukan spesifikasi masukan yang diperlukan, sistem keluaran yang akan dihasilkan sistem dan proses yang dibutuhkan untuk mengolah masukan sehingga menghasilkan suatu keluaran yang diinginkan. Analisis kebutuhan non fungsional pada sistem ini meliputi analisis perangkat lunak, analisis perangkat keras dan perangkat pikir.

1. Persyaratan Produk

   Berikut adalah kebutuhan non fungsional berdasarkan klasifikasi *product requirement* atau persyaratan produk.

   a. SKPL-NF-001 Sistem yang dibangun memiliki dua subsistem yaitu *mobile* dan *web*

b. SKPL-NF-002 Sistem yang dibangun minimal menggunakan sistem operasi andorid versi 5.0 Lolipop dan dibangun dengan dengan kebutuhan perangkat lunak minimum:

Tabel 3.5 Kebutuhan Perangkat Lunak

No	Kebutuhan Perangkat Lunak
1	Sistem Operasi Windows 7 atau diatasnya
2	Web server XAMPP versi 1.1.7
3	Kode editor berupa sublime text 3
4	Integrated development environment (IDE) Andorid Studio
5	Software Development Kit (SDK) version 27.0.1

c. SKPL-NF-003 Sistem yang dibangun memanfaatkan teknologi GPS
d. SKPL-NF-004 Sistem yang dibangun memanfaatkan teknologi *magnetic field*
e. SKPL-NF-005 Sistem yang dibangun memanfaatkan koneksi *internet* pada *smartphone*
f. SKPL-NF-006 Sistem dibangun dengan spesifikasi *hardware* yang memenuhi standar minimum kebutuhan seperti

Tabel 3.6 Kebutuhan Perangkat Keras Subsistem Mobile

No	Kebutuhan Perangkat Keras Subsistem *Mobile*
1	Sistem Operasi Android versi 5.0 Lolipop
2	Minimum Processor Quad Core 1.8 GHz
3	Minimum RAM 1 GB
4	Memiliki sensor GPS
5	Memiliki sensor *magnetic field*
6	Memiliki koneksi internet 3G/4G/Wifi

Tabel 3.7 Kebutuhan Perangkat Keras Subsistem Web

No	Kebutuhan Perangkat Keras Subsistem *Web*
1	Minimum CPU @2.0 GHz
2	Minimum RAM 1 GB
3	Minimum Harddisk 20GB
4	Koneksi Internet

2. Persyaratan Eksternal

   Kebutuhan non fungsional berdasarkan klasifikasi *external requirement* atau persyaratan eksternal adalah SKPL-NF-007, perangkat lunak yang dibangun harus menghasilkan format file standar (.json) yang bisa digunakan oleh pihak luar yang berkepentingan.

3. Analisis Kebutuhan Perangkat Pikir

   Analisis kebutuhan perangkat pikir adalah batasan-batasan dari layanan dan fungsi-fungsi yang dibutuhkan oleh sistem dari segi perangkat pikir. Pada penelitian ini analisis kebutuhan perangkat pikir dikelompokkan menjadi tiga kategori yaitu pengetahuan dan pengalaman, pekerjaan dan karakteristik tugas, dan karakteristik fisik.

   a. Pengetahuan dan pengalaman

      Pengetahuan dan pengalaman yang diharapkan dari pengguna perangkat lunak pada penelitian ini adalah sebagai berikut:

Tabel 3.8 Pengetahuan dan Pengalaman Pengguna

Pengetahuan dan Pengalaman	Tingkatan
Pengalaman penggunaan sistem	Rendah sampai tinggi
Pengalaman penggunaan aplikasi	Rendah sampai tinggi
Pengalaman tugas yang berkaitan	Rendah sampai tinggi
Terbiasa menggunakan sistem lain	Cukup
Tingkat Pendidikan	SMA / SMK keatas
Kemampuan membaca	Sedang sampai tinggi
Kemampuan mengetik	Sedang sampai tinggi
Kemampuan bahasa	English / Indonesia

   b. Karakteristik tugas

      Karakteristik tugas yang diharapkan dari pengguna perangkat lunak pada penelitian ini adalah sebagi berikut:

Tabel 3.9 Karakteristik Tugas Pengguna

Karakteristik Tugas	Tingkatan
Struktur Kerja	Sedang
Interaksi sosial	Perlu
Pelatihan	Tidak perlu
Kategori pekerjaan	Teknisi
Frekuensi penggunaan	Sedang
Tugas atau kebutuhan yang penting	Sedang

c. Karakteristik fisik

Karakteristik fisik yang diharapkan dari pengguna perangkat lunak pada penelitian ini adalah sebagai berikut:

Tabel 3.10 Karakteristik Fisik Pengguna

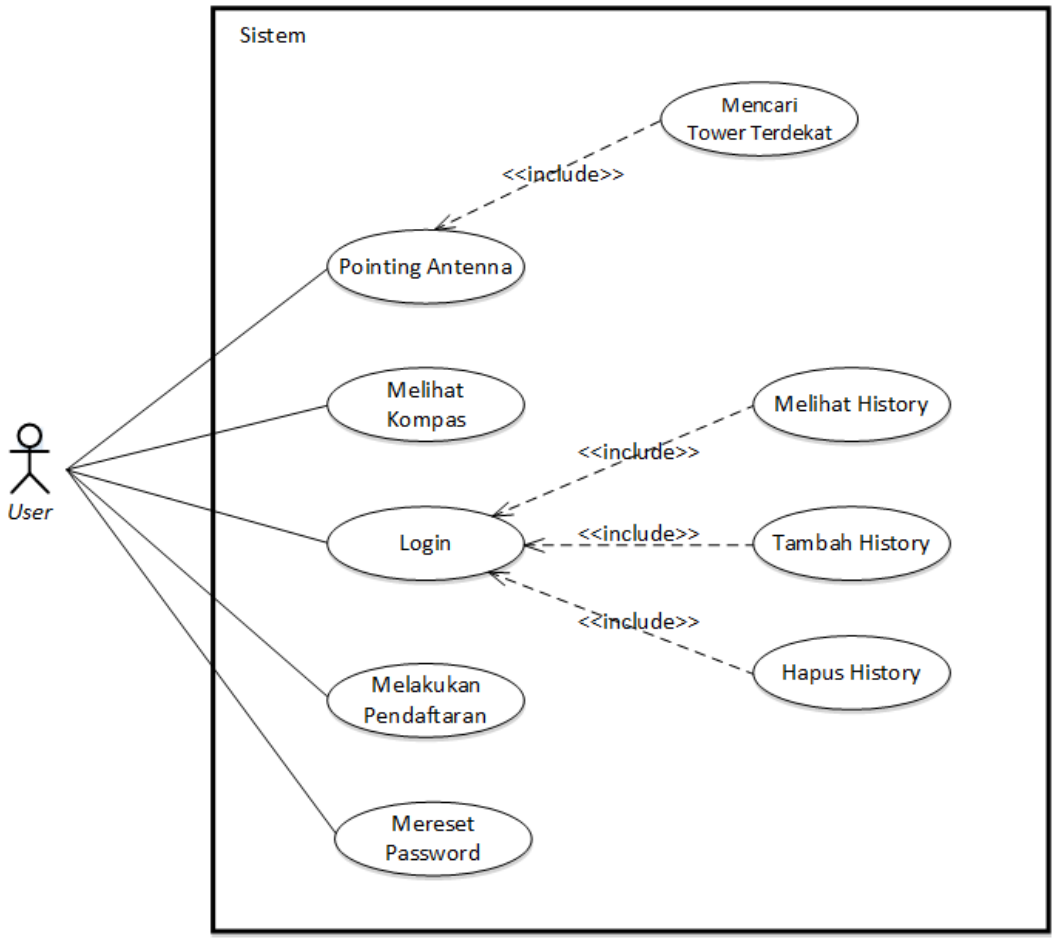
Karakteristik Fisik	Level
Umur	16 tahun sampai dengan 40 tahun
Jenis kelamin	Laki-laki
Penggunaan tangan	Kedua tangan (kiri dan kanan)
Cacat Fisik	Tidak cacat fisik

3.1.7 Analisis Kebutuhan Fungsional

Analisis kebutuhan fungsional menggambarkan proses kegiatan yang akan diterapkan dalam sistem dan menjelaskan kebutuhan yang diperlukan agar sistem dapat berjalan dengan baik serta sesuai dengan kebutuhan. Analisis yang dilakukan dimodelkan dengan UML (*Unifield Modeling Language*). Tahap-tahap pemodelan UML meliputi *use case*, *activity diagram*, *class diagram* dan *diagram sequence*.

3.1.7.1 *Use Case Diagram*

Use Case Diagram memperlihatkan hubungan diantara aktor dan *use case*. Aktor merepresentasikan seorang user atau subsistem lain yang akan berinteraksi dengan sistem. Sedangkan *use case* merupakan kejadian yang menggambarkan interaksi antara *user* dengan sistem. Fungsionalitas sistem didefinisikan ke dalam *use case* dari sudut eksternal sistem yang berguna untuk uji kelayakan sistem.



Gambar 3.11 *Use Case Diagram* Aplikasi *Antenna Pointer*

Definisi aktor dan deskripsi peran untuk aktor di dalam sistem dapat dilihat table sebagai berikut:

Tabel 3.11 Definisi Aktor

No	Aktor	Deskripsi
1	*User*	Memiliki wewenang untuk menggunakan fungsionalitas-nya pada level user interface

Identifikasi *use case* yang terdapat dalam sistem dapat dilihat pada tabel sebagai berikut:

Tabel 3.12 Definisi *Use Case*

No	Aktor	Deskripsi
1	Mencari Tower Terdekat	Sistem menyediakan fasilitas bagi *user* untuk melihat lokasi tower terdekat
2	*Pointing Antenna*	Sistem menyediakan fasilitas bagi *user* untuk melakukan pointing antenna dan mendapatkan hasil perhitungan *fresnel zone* dalam bentuk grafik
3	Melihat Kompas	Sistem menyediakan fasilitas bagi *user* untuk melihat kompas sesuai dengan arah lokasi tower tujuan
4	Melakukan Pendaftaran	Sistem menyediakan fasilitas bagi *user* untuk daftar atau registrasi user baru
5	Login	Sistem menyediakan fasilitas bagi *user* untuk *login* kedalam sistem
6	Mereset *Password*	Sistem menyediakan fasilitas bagi *user* untuk *reset password* jika *user* lupa password login
7	Melihat *History*	Sistem menyediakan fasilitas bagi *user* untuk melihat *history* instalasi antena
8	Tambah *History*	Sistem menyediakan fasilitas bagi *user* untuk menambah *history* instalasi antena
9	Hapus *History*	Sistem menyediakan fasilitas bagi *user* untuk menghapus *history* instalasi antena

3.1.7.2 *Use Case Scenario*

Use case scenario menjelaskan bagaimana skenario dari setiap proses yang ada pada setiap *use case*. Pada skenario *use case* ini dapat dijabarkan bagaimana alur dari setiap proses pada setiap *use case* berdasarkan interaksi antara aktor dan sistem. Adapun *use case scenario* untuk setiap *use* case *sebagai* berikut:

1. *Use case* Mencari Tower Terdekat

   Berikut adalah tabel yang menjelaskan *use case scenario* atau langkah - langkah dari *use case* Tower Terdekat:

Tabel 3.13 *Use Case Scenario* Mencari Tower Terdekat

Use Case Name	Mencari Tower Terdekat	
Related Requirements	-	
Goal Context	*User* mendapatkan informasi tower terdekat	
Preconditions	Mengakses halaman pointing antenna	
Successful End Condition	Tower terdekat berhasil ditampilkan pada maps	
Failed End Condition	Tower tidak tampil pada maps	
Primary Actor	*User*	
Trigger	*User* memilih menu pointing antenna	
Main Flow	*Step*	*Action*
	1	User memilih menu Pointing Antenna
	2	Sistem menampilkan maps
	3	Sistem menampilkan tower terdekat di maps

2. *Use case Pointing Antenna*

   Berikut adalah tabel yang menjelaskan *use case scenario* atau langkah - langkah dari *use case* Pointing Antenna:

Tabel 3.14 *Use Case Scenario Pointing Antenna*

Use Case Name	Pointing Antenna	
Related Requirements	-	
Goal Context	*User* dapat melakukan pointing antenna	
Preconditions	Mengakses halaman pointing antenna	
Successful End Condition	Pointing antenna berhasil menghasilkan grafik	
Failed End Condition	Gagal memproses pointing antenna pada grafik	
Primary Actor	*User*	
Trigger	*User* memilih menu pointing antenna	
Main Flow	*Step*	*Action*
	1	*User* memilih menu Pointing Antenna
	2	Sistem menampilkan maps berikut dengan form isian pointing
	3	*User* input data pointing
	4	Sistem mengolah data pointing antenna
Extension	*Step*	*Action*
	4.1	Sistem berhasil menampilkan grafik
	4.2	Sistem gagal menampilkan grafik

3. *Use case* Melihat Kompas

   Berikut adalah tabel yang menjelaskan *use case scenario* atau langkah - langkah dari *use case* Kompas:

Tabel 3.15 *Use Case Scenario* Melihat Kompas

Use Case Name	Melihat Kompas	
Related Requirements	-	
Goal Context	*User* dapat melihat kompas mengarah tower tujuan	
Preconditions	Mengakses halaman kompas	
Successful End Condition	Kompas secara akurat mengarah ke lokasi tower tujuan	
Failed End Condition	Arah kompas tidak sesuai	
Primary Actor	*User*	
Trigger	*User* memilih menu kompas	
Main Flow	*Step*	*Action*
	1	*User* mengakses menu kompas
	2	Sistem menampilkan maps agar *user* dapat memilih lokasi tower tujuan
	3	*User drop pin* pada maps sebagai lokasi tower tujuan
	4	Sistem mengolah kordinat lokasi dan menampilkan kompas

4. *Use case* Melakukan Pendaftaran

   Berikut adalah tabel yang menjelaskan *use case scenario* atau langkah - langkah dari *use case* Daftar:

Tabel 3.16 *Use Case Scenario* Melakukan Pendaftaran

Use Case Name	Melakukan Pendaftaran	
Related Requirements	-	
Goal Context	*User* dapat melakukan pendaftaran *account* baru	
Preconditions	*User* belum login dan belum memiliki *account*	
Successful End Condition	*User* berhasil melakukan pendaftaran atau membuat *account* baru	
Failed End Condition	*User* gagal melakukan pendaftaran *account* baru	
Primary Actor	*User*	
Trigger	*User* memilih menu daftar	
Main Flow	*Step*	*Action*
	1	*User* mengakses halaman daftar
	2	Sistem menampilkan *form* pendaftaran

	3	*User* mengisi data pendaftaran
	4	Sistem melakukan validasi data
	5	Menampilkan pesan pendaftaran berhasil
Extension	*Step*	*Action*
	4.1	Sistem memproses dan menyimpan data pendaftaran
	4.2	Sistem melakukan *redirect* ke form pendaftaran dan tampil pesan gagal

5. *Use case* Login

   Berikut adalah tabel yang menjelaskan *use case scenario* atau langkah - langkah dari *use case Login*:

Tabel 3.17 *Use Case Scenario Login*

Use Case Name	*Login*	
Related Requirements	-	
Goal Context	*User* dapat *login* kedalam sistem	
Preconditions	*User* belum memiliki *session login*	
Successful End Condition	*User* berhasil login dan masuk kedalam sistem	
Failed End Condition	*User* gagal melakukan *login*	
Primary Actor	*User*	
Trigger	*User* memilih menu login	
Main Flow	*Step*	*Action*
	1	*User* mengakses halaman *login*
	2	Sistem menampilkan form *login*
	3	*User* mengisi data *login*
	4	Sistem melakukan validasi data
	5	Sistem melakukan *redirect* ke halaman melihat *history*
Extension	*Step*	*Action*
	4.1	Sistem membuat session login
	4.2	Sistem melakukan *redirect* ke form *login* dan tampil pesan gagal

6. *Use case* Mereset *Password*

   Berikut adalah tabel yang menjelaskan *use case scenario* atau langkah - langkah dari *use case* Lupa *Password:*

Tabel 3.18 *Use Case Scenario* Mereset *Password*

Use Case Name	Mereset *Password*	
Related Requirements	-	
Goal Context	*User* dapat me-*reset password* login	
Preconditions	*User* memiliki akun *login* atau sudah terdaftar	
Successful End Condition	*User* berhasil me-*reset password*	
Failed End Condition	*User* gagal me-*reset password*	
Primary Actor	*User*	
Trigger	*User* memilih tombol lupa *password*	
Main Flow	*Step*	*Action*
	1	*User* mengakses halaman lupa *password*
	2	Sistem menampilkan form lupa *password*
	3	*User* mengisi data form
	4	Sistem melakukan validasi data
	5	Sistem menampilkan pesan *reset password* sudah terkirim melalui *email*
Extension	*Step*	*Action*
	4.1	Sistem mengirim email *reset password*
	4.2	Sistem melakukan *redirect* ke form lupa *password* dan tampil pesan gagal

7. *Use case* Melihat *History*

   Berikut adalah tabel yang menjelaskan *use case scenario* atau langkah - langkah dari *use case* Melihat *History:*

Tabel 3.19 *Use Case* Scenario Melihat *History*

Use Case Name	Melihat *History*	
Related Requirements	-	
Goal Context	*User* dapat melihat *history* instalasi antena	
Preconditions	*User* sudah *login* terlebih dahulu	
Successful End Condition	Sistem berhasil menampilkan list *history* instalasi antena	
Failed End Condition	Sistem gagal menampilkan *history* instalasi antena	
Primary Actor	*User*	
Trigger	*User* mengakses menu *history*	
Main Flow	*Step*	*Action*
	1	*User* mengakses halaman *history*
	2	Sistem menampilkan *list history*
	3	*User* memilih *history* untuk melihat detail
	4	Sistem menampilkan detail dari *history*

8. *Use case* Tambah *History*

   Berikut adalah tabel yang menjelaskan *use case scenario* atau langkah - langkah dari *use case* Tambah *History:*

Tabel 3.20 *Use Case Scenario* Tambah *History*

Use Case Name	Tambah *History*	
Related Requirements	-	
Goal Context	*User* dapat menambahkan *history* instalasi antena	
Preconditions	*User* sudah *login* dan berada di halaman *history*	
Successful End Condition	Sistem berhasil menambahkan *history* instalasi antena	
Failed End Condition	Sistem gagal menambahkan *history* instalasi antena	
Primary Actor	*User*	
Trigger	Halaman melihat *history* berhasil ditampilkan	
Main Flow	*Step*	*Action*
	1	*User* memilih tombol tambah *history*
	2	Sistem menampilkan form tambah *history*
	3	*User* mengisi data *history*
	4	Sistem melakukan validasi data
Extension	*Step*	*Action*
	4.1	Sistem berhasil menyimpan data
	4.2	Data gagal tersimpan dan kembali ke form

9. *Use case* Hapus *History*

   Berikut adalah tabel yang menjelaskan *use case scenario* atau langkah - langkah dari *use case* Hapus *History:*

Tabel 3.21 *Use Case Scenario* Hapus *History*

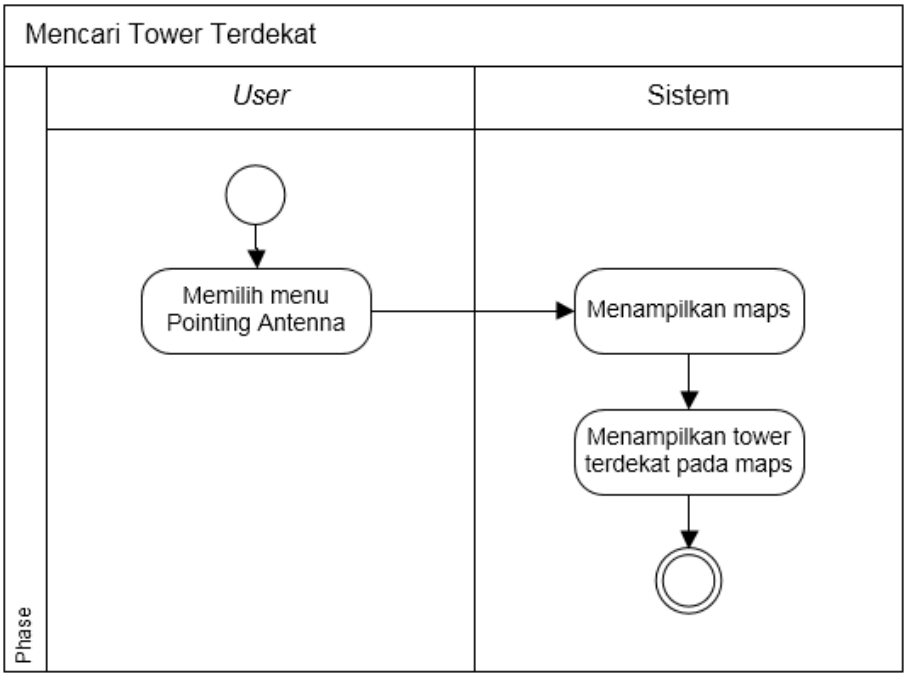
Use Case Name	Hapus *History*	
Related Requirements	-	
Goal Context	*User* dapat menghapus *history* instalasi antena	
Preconditions	*User* sudah *login* dan berada di halaman *history*	
Successful End Condition	Sistem berhasil menghapus *history* instalasi antena	
Failed End Condition	Sistem gagal menghapus *history* instalasi antena	
Primary Actor	*User*	
Trigger	Halaman melihat *history* berhasil ditampilkan	
Main Flow	*Step*	*Action*
	1	*User* memilih *history* yang akan dihapus
	2	Sistem menampilkan pesan konfirmasi

	3	Sistem memuat ulang list history
Extension	*Step*	*Action*
	2.1	Sistem menghapus data *history*
	2.2	Sistem batal menghapus data *history*

3.1.7.4 *Activity Diagram*

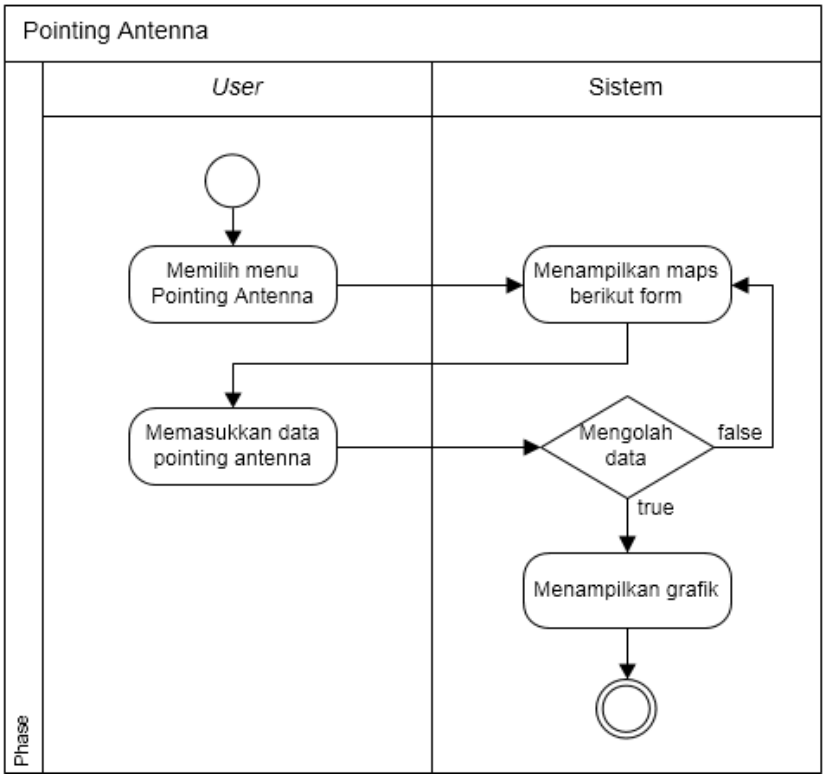
Activity diagram menggambarkan proses pelayanan dan urutan aktivitas dalam sebuah proses atau *use case* untuk memperlihatkan urutan aktivitas proses bisnis. *Activity diagram* sangat bermanfaat untuk memahami proses dari sistem secara keseluruhan. Activity diagram dibuat berdasarkan sebuah atau beberapa *use case* pada *use case diagram* dan mengacu pada *use case scenario* yang sudah dibuat. Adapun beberapa *acitvity diagram* pada setiap *use case* sebagai berikut:

1. *Activity Diagram* Mencari Tower Terdekat



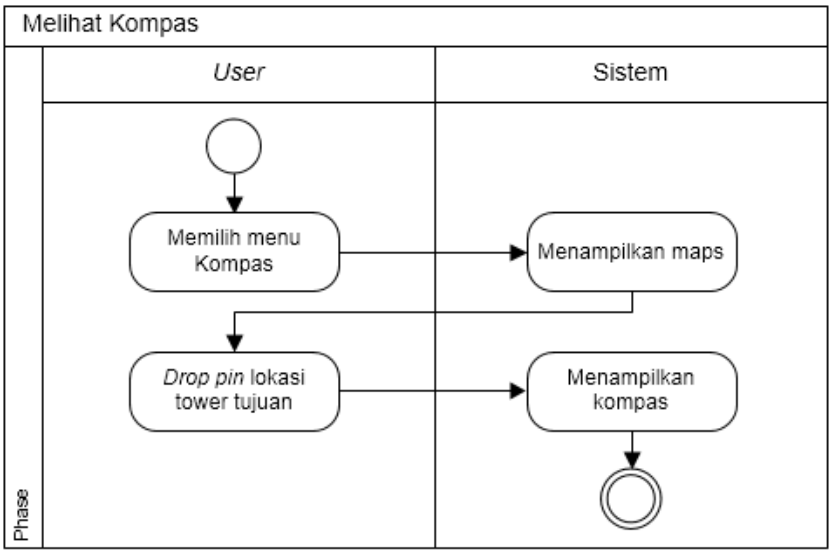
Gambar 3.12 *Activity Diagram* Mencari Tower Terdekat

2. *Activity Diagram Pointing Antenna*



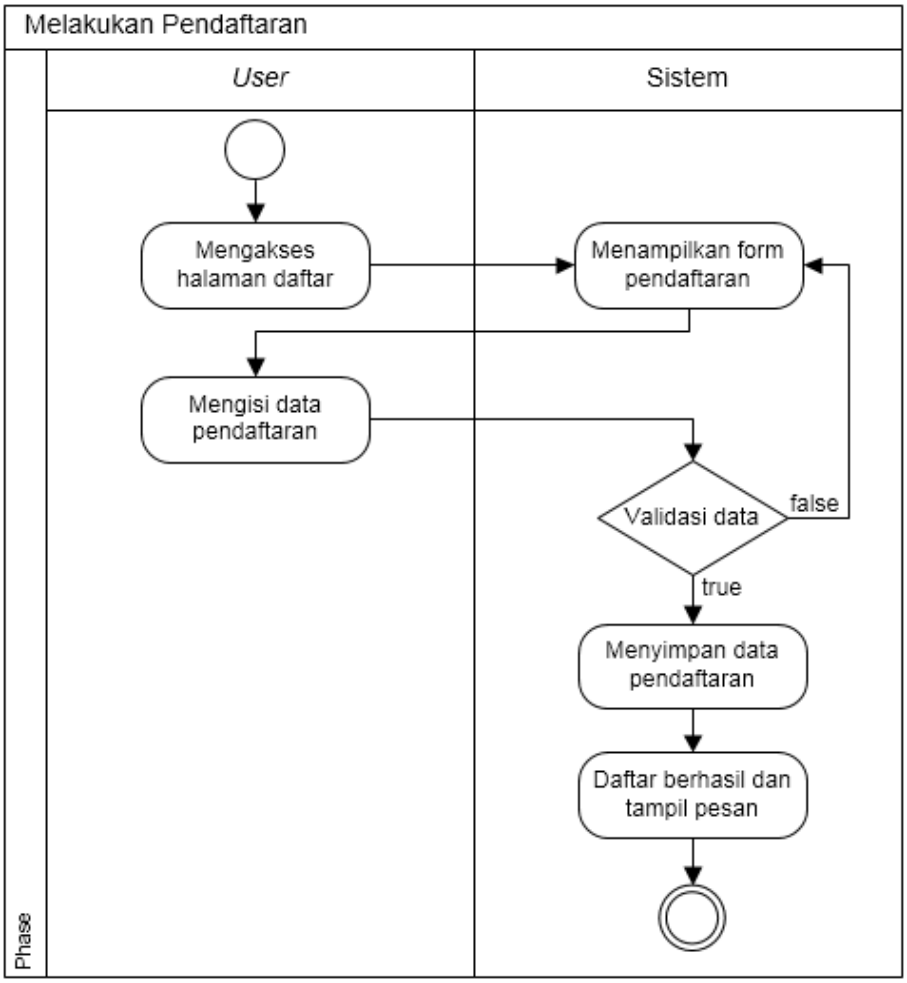
Gambar 3.13 ***Activity Diagram Pointing Antenna***

3. *Activity Diagram* Melihat Kompas



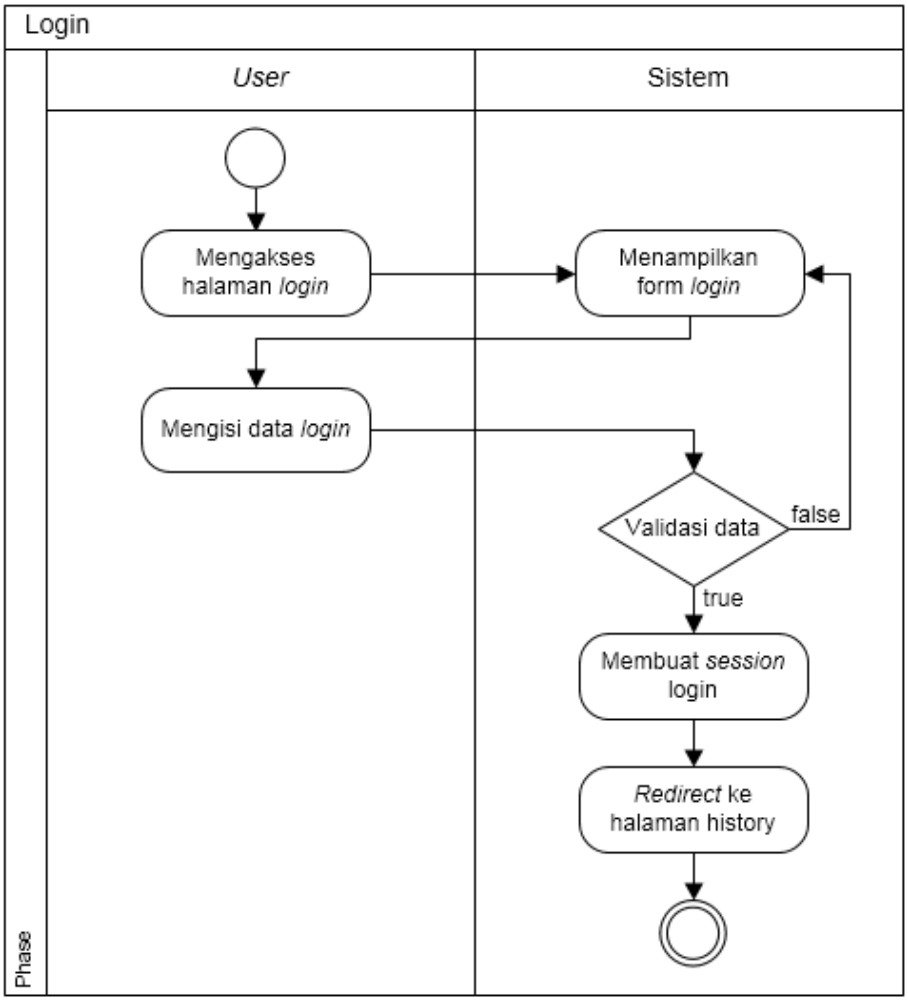
Gambar 3.14 ***Activity Diagram*** **Melihat Kompas**

4. *Activity Diagram* Melakukan Pendaftaran



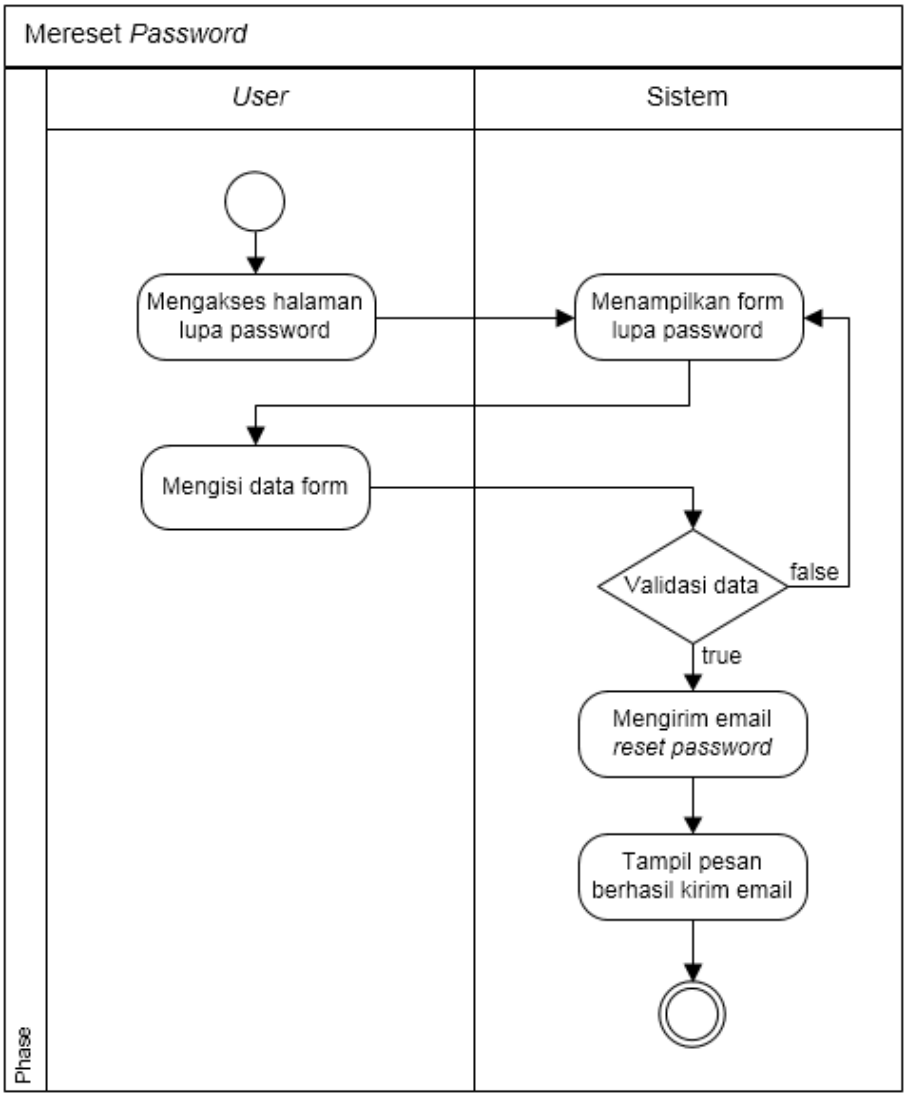
Gambar 3.15 *Activity Diagram* Melakukan Pendaftaran

5. *Activity Diagram Login*



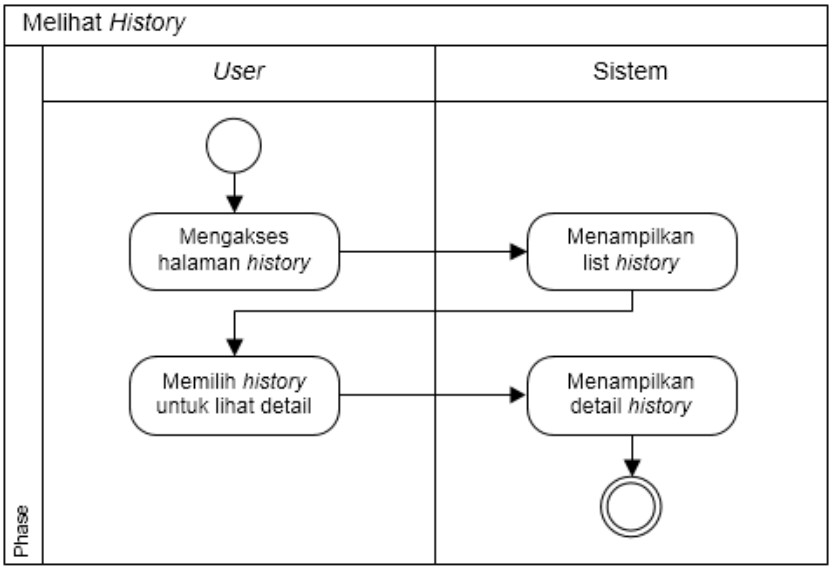
Gambar 3.16 ***Activity Diagram Login***

6. *Activity Diagram* Mereset *Password*



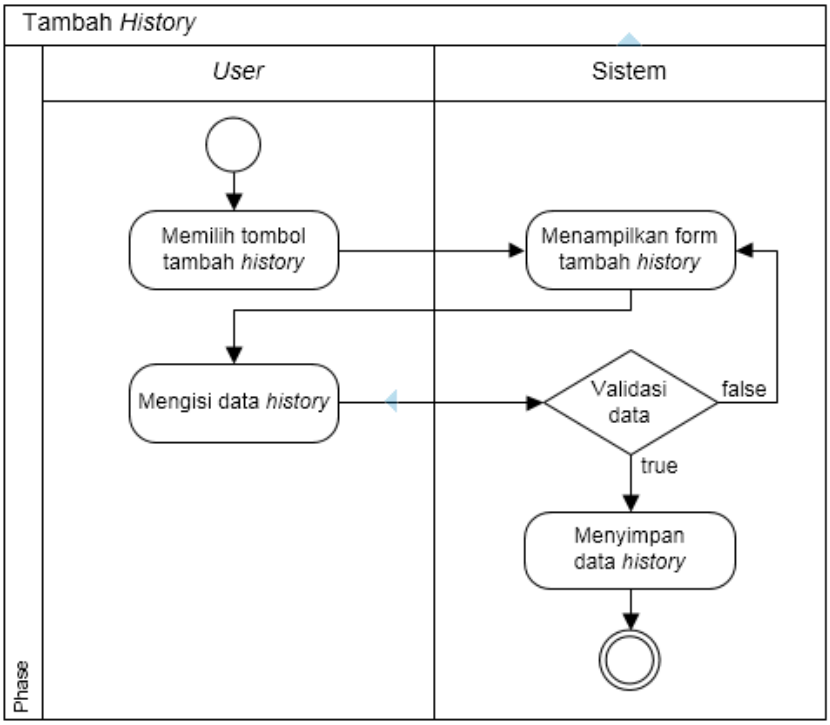
Gambar 3.17 *Activity Diagram* Mereset *Password*

7. *Activity Diagram* Melihat *History*



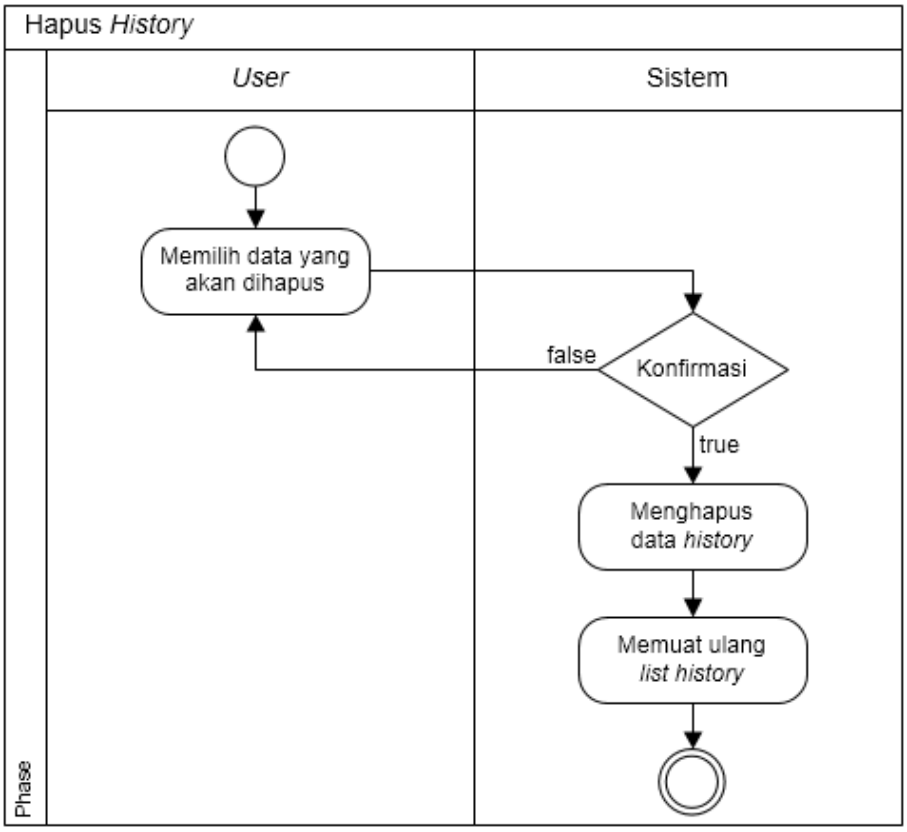
Gambar 3.18 Activity Diagram* Melihat *History

8. *Activity Diagram* Tambah *History*



Gambar 3.19 *Activity Diagram* Tambah *History*

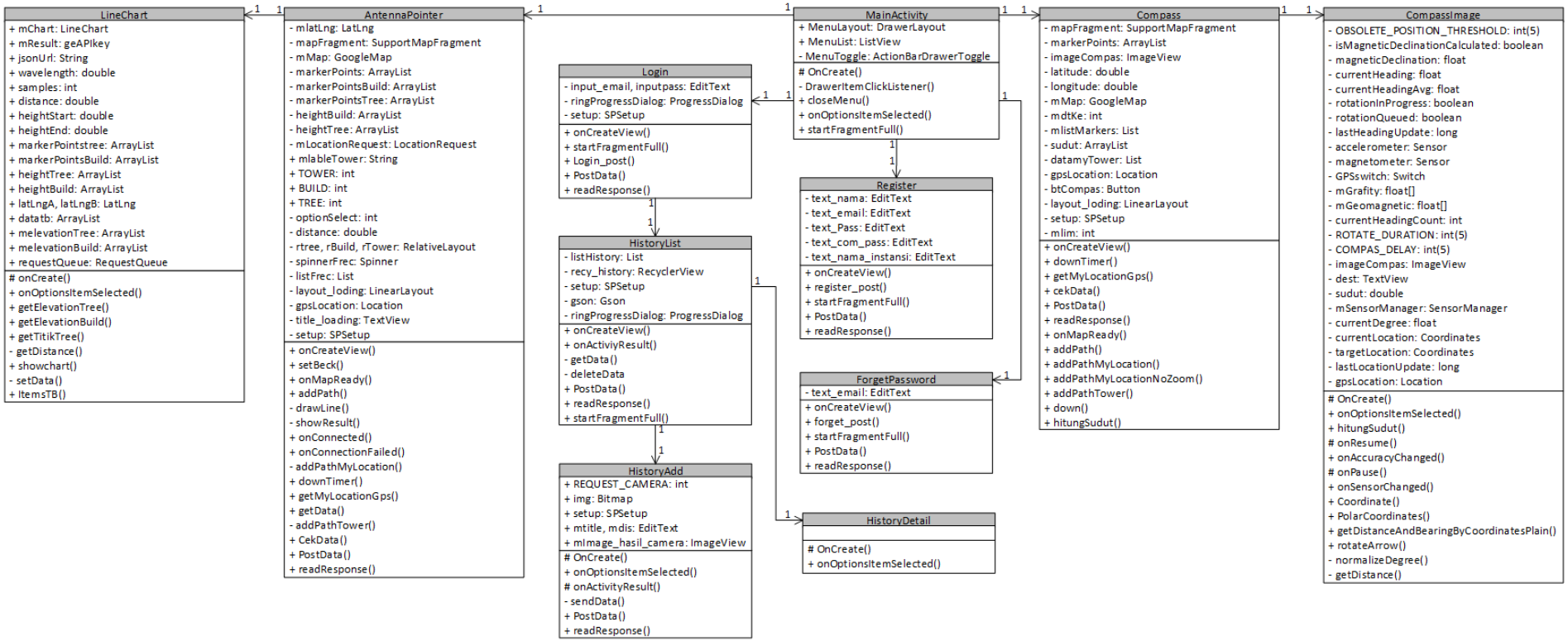
9. *Activity Diagram* Hapus *History*



Gambar 3.20 *Activity Diagram* Hapus *History*

3.1.7.5 *Class Diagram*

Class Diagram digunakan untuk menggambarkan *class-class* yang terlibat dalam analisis sebuah sistem yang akan dibangun. Dengan *class diagram* struktur dan deskripsi *class* serta hubungan antar *class* akan terlihat dengan jelas beserta *method-method* setiap *class*-nya. Adapun class diagram dari sistem yang dibangun dapat dilihat pada gambar berikut:

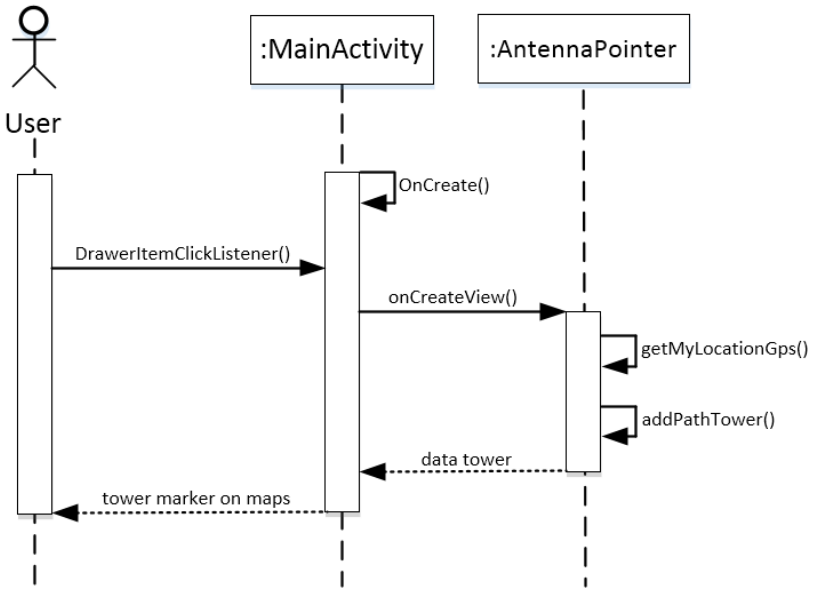


Gambar 3.21 *Class Diagram* Aplikasi *Antenna Pointer*

3.1.7.6 *Sequence Diagram*

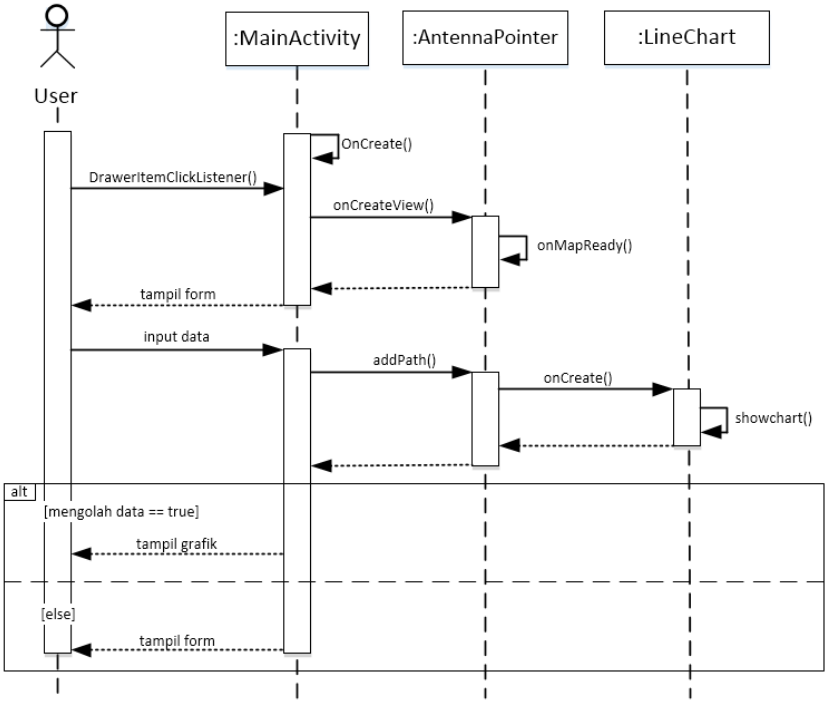
Sequence Diagram digunakan untuk menggambarkan interaksi antar objek dan mengindikasikan komunikasi diantara objek-objek tersebut. Diagram ini akan menunjukkan pesan-pesan yang bertukar antar objek yang melakukan suatu tugas atau *method* pada objek tersebut. Terdapat beberapa *sequence diagram* pada sistem yang dibangun sebagai berikut:

1. *Sequence Diagram* Mencari Tower Terdekat



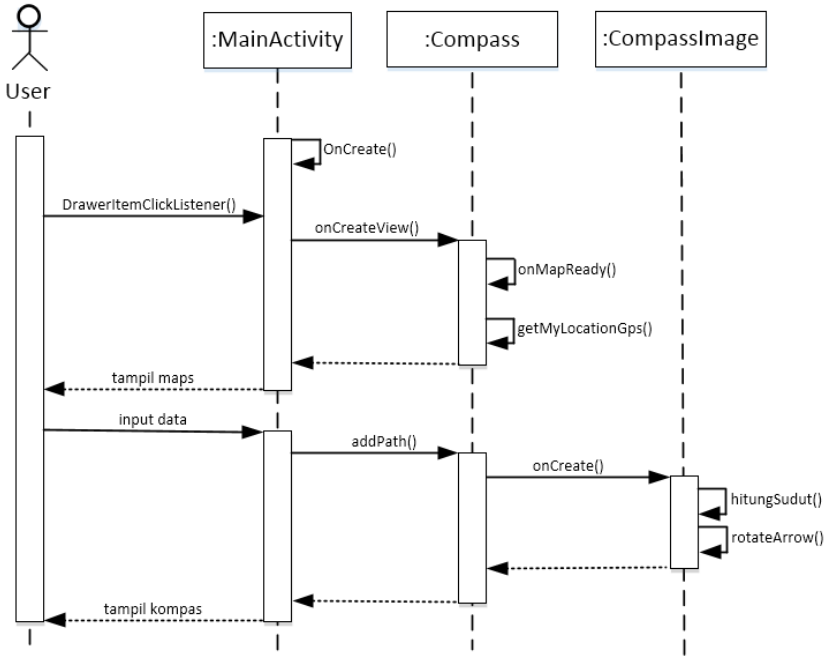
Gambar 3.22 *Sequence Diagram* Mencari Tower Terdekat

2. *Sequence Diagram Pointing Antenna*



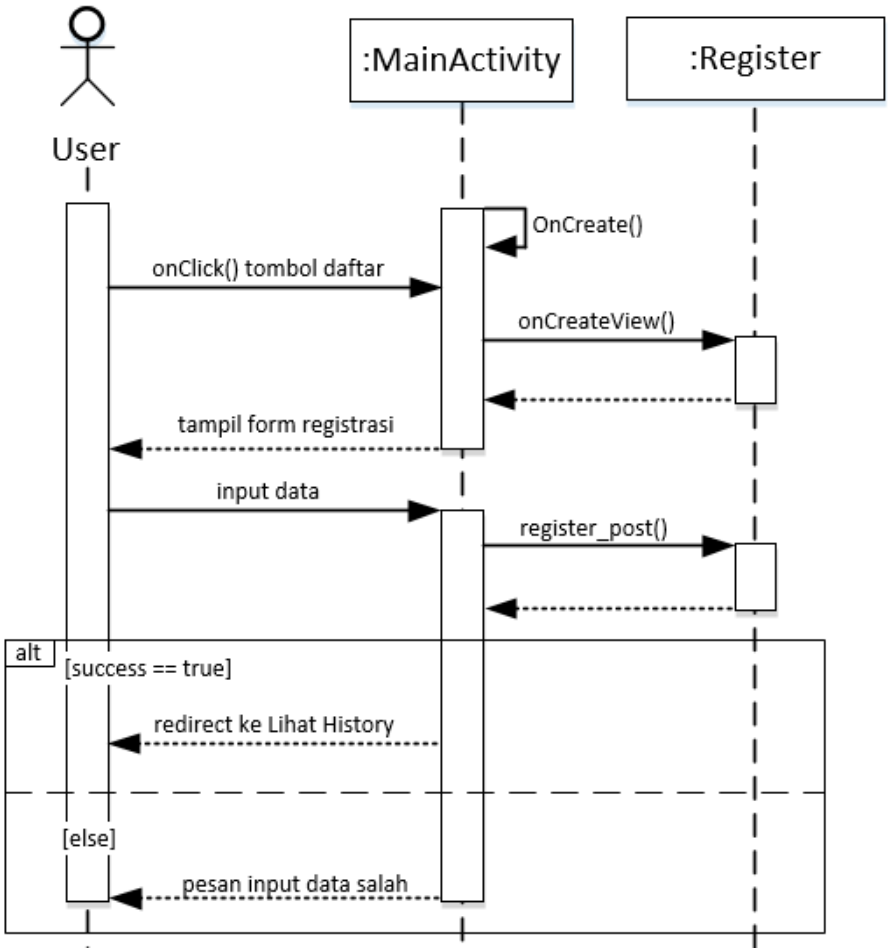
Gambar 3.23 *Sequence Diagram Pointing Antenna*

3. *Sequence Diagram* Melihat Kompas



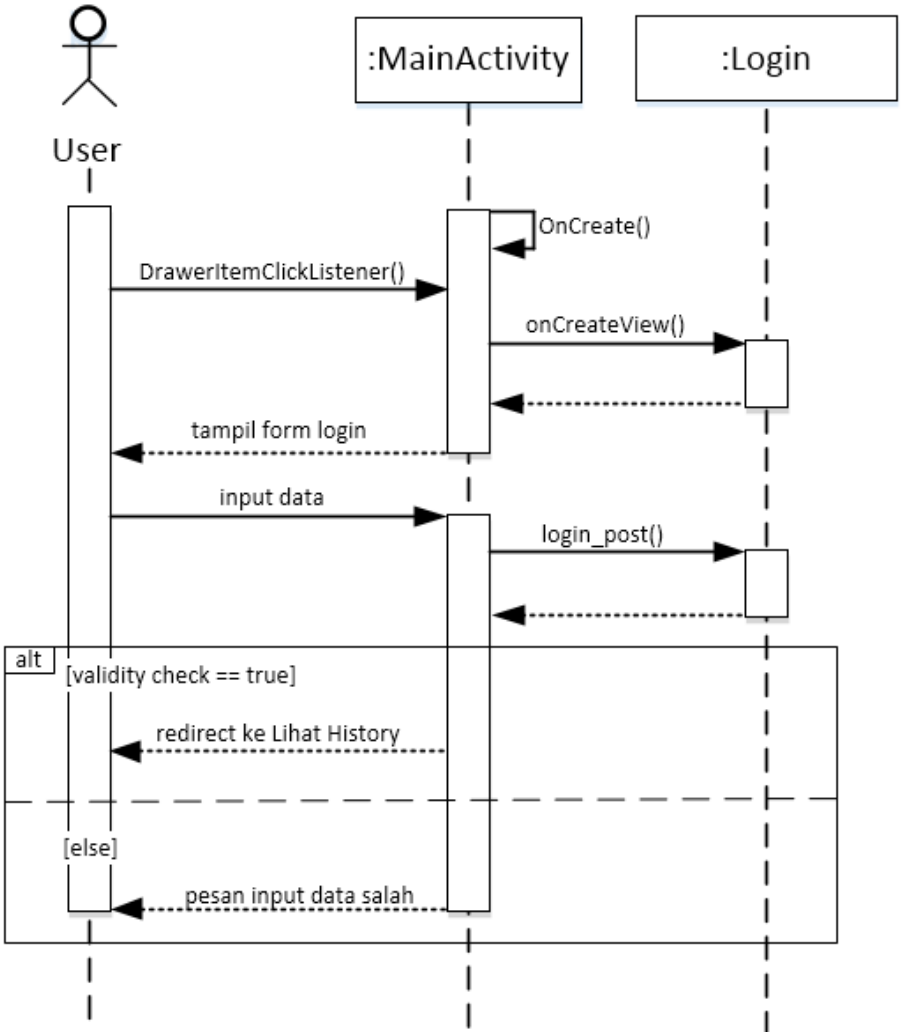
Gambar 3.24 *Sequence Diagram* Melihat Kompas

4. *Sequence Diagram* Melakukan Pendaftaran



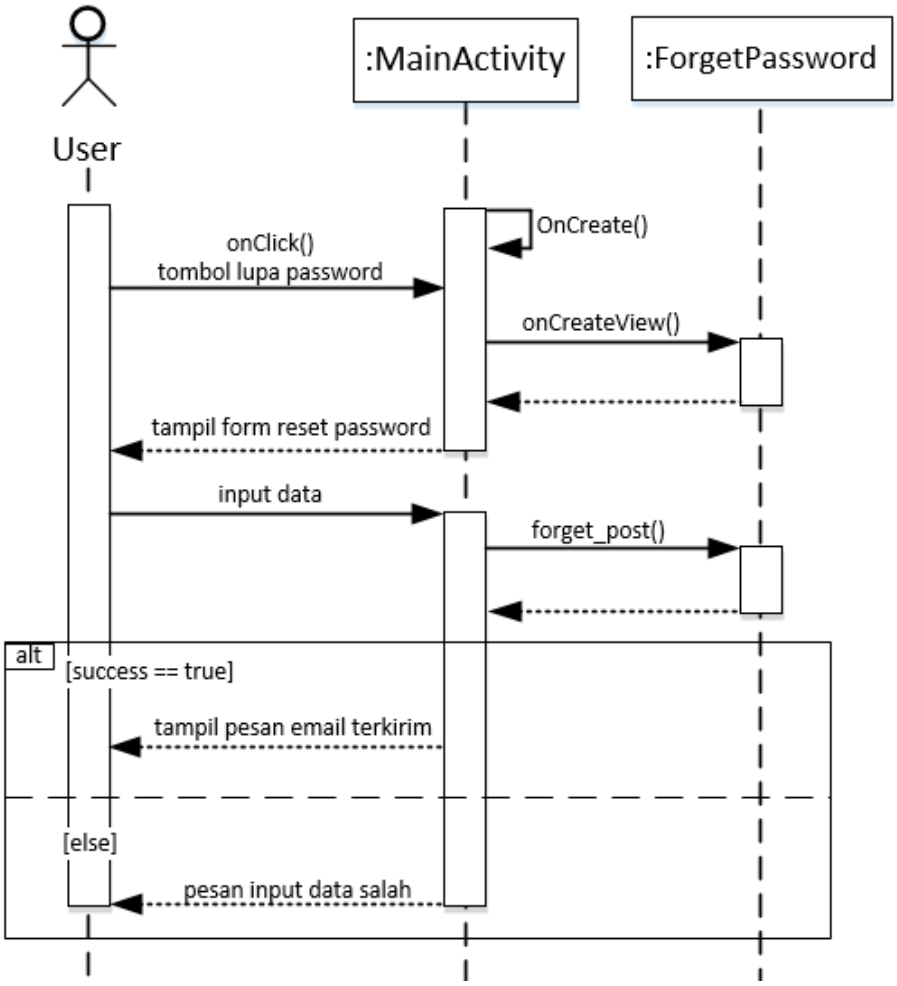
Gambar 3.25 *Sequence Diagram* Melakukan Pendaftaran

5. *Sequence Diagram Login*



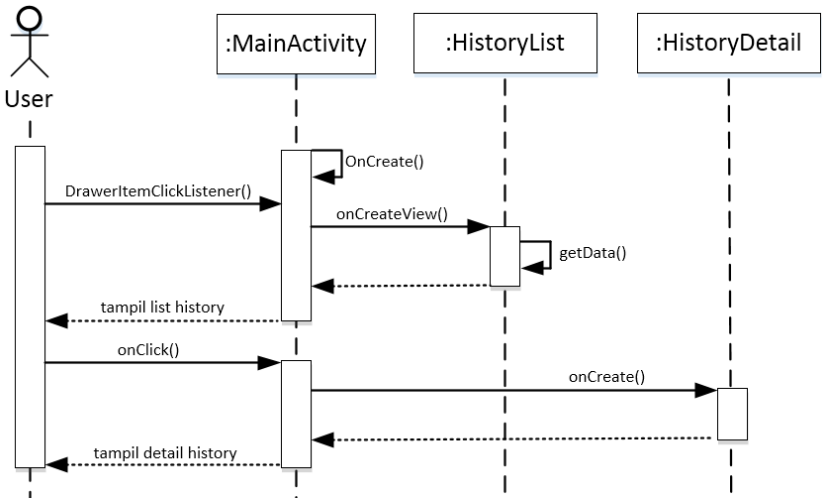
Gambar 3.26 *Sequence Diagram Login*

6. *Sequence Diagram* Mereset *Password*



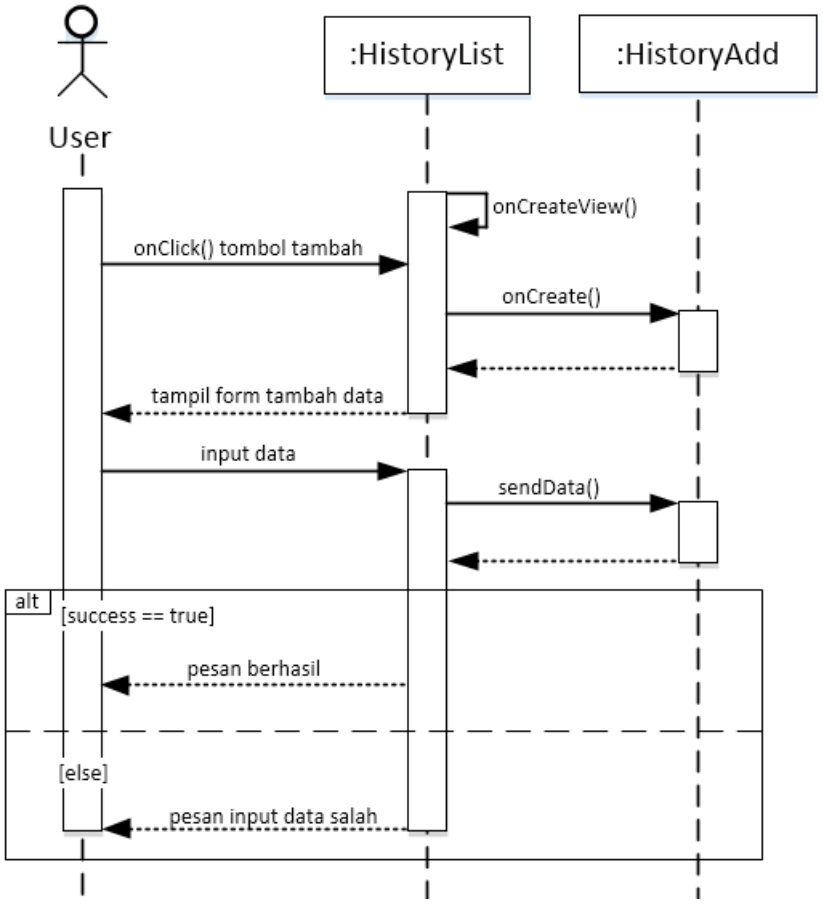
Gambar 3.27 *Sequence Diagram* Mereset *Password*

7. *Sequence Diagram* Melihat *History*



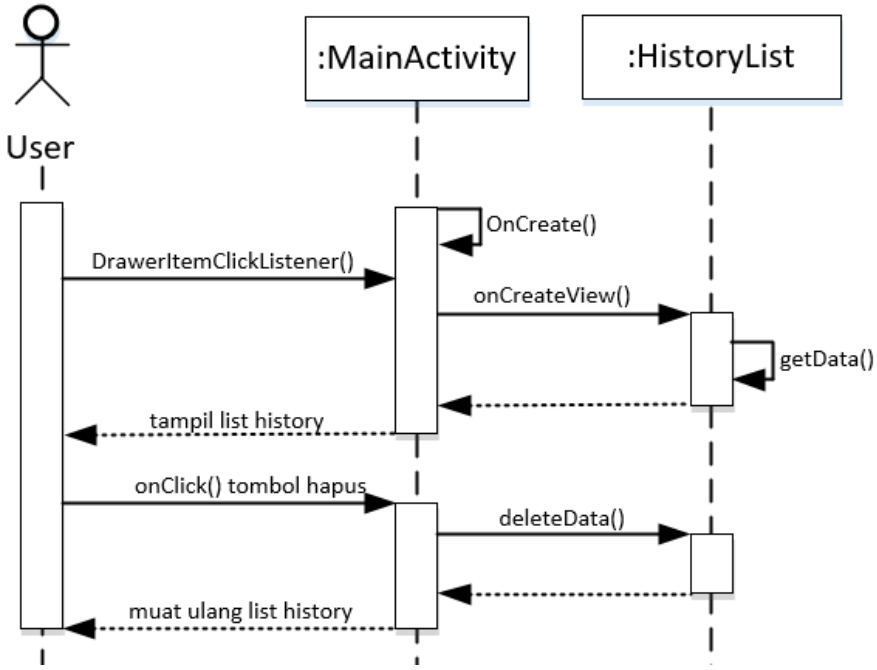
Gambar 3.28 *Sequence Diagram* Melihat *History*

8. *Sequence Diagram* Tambah *History*



Gambar 3.29 *Sequence Diagram* Tambah *History*

9. *Sequence Diagram* Hapus *History*



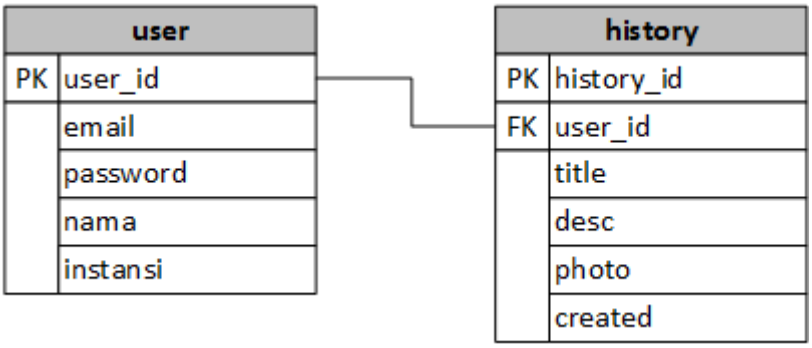
Gambar 3.30 *Sequence Diagram* Hapus *History*

3.2 Perancangan Sistem

Perancangan akan dimulai setelah tahap analisis terhadap sistem selesai dilakukan. Perancangan dapat didefinisikan sebagai teknik dan prinsip bagi tujuan pendefinisian suatu perangkat, suatu proses atau system dalam detail yang memadai untuk memungkinkan realisasi fisiknya. Perancangan digambarkan sebagai proses multi-langkah dimana representasi perancangan data, skema relasi, struktur file, perancangan antar muka, dan perancangan prosedural dari persyaratan informasi.

3.2.1 Skema Relasi

Skema relasi merupakan rangkaian hubungan antara dua tabel atau lebih pada sistem basis data. Pada gambar dibawah ini, merupakan gambaran rangkaian basis data pada pembangunan aplikasi antenna pointer.



Gambar 3.31 Skema Relasi

3.2.2 Struktur Tabel

Perancangan struktur tabel bertujuan untuk mengidentifikasi atribut-atribut yang ada pada suatu tabel. Perancangan struktur tabel didasarkan kepada perancangan basis data yang telah dirancang sebelumnya. Adapun perancangan struktur tabel pada sistem yang dibangun adalah sebagai berikut:

1. Tabel *user*

Tabel 3.22 Struktur Tabel *user*

Field	**Tipe Data**	**Ukuran**	**Keterangan**
user_id	int	11	*Primary Key*
email	varchar	100	
password	varchar	50	
nama	varchar	100	
instansi	varchar	50	

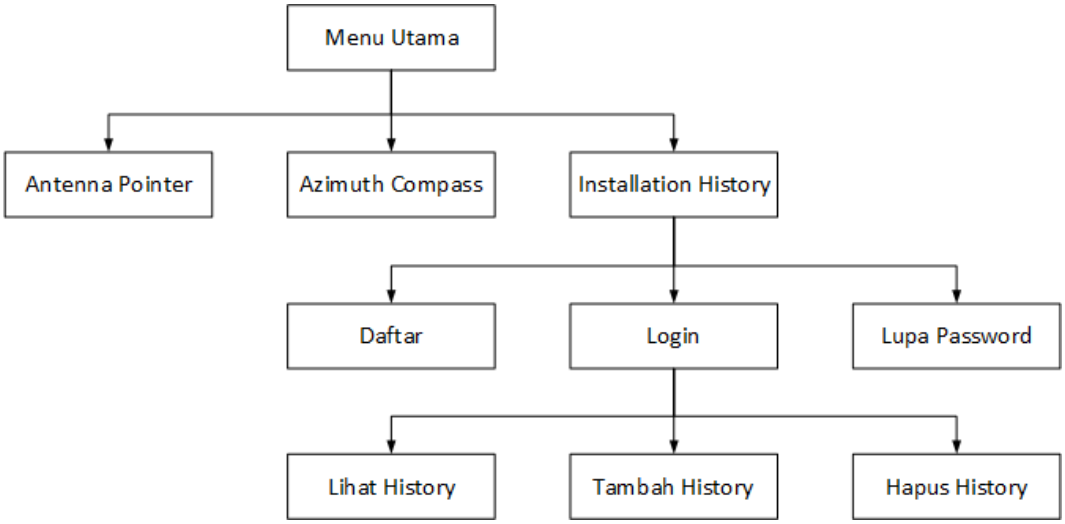
2. Tabel *history*

Tabel 3.23 Struktur Tabel *history*

Field	Tipe Data	Ukuran	Keterangan
history_id	int	11	*Primary Key*
user_id	int	11	*Foreign Key*
title	varchar	100	
desc	text		
photo	varchar	100	
created	datetime		

3.2.3 Perancangan Struktur Menu

Perancangan struktur menu merupakan gambaran jalur pemakaian aplikasi sehingga aplikasi yang mudah dipahami dan mudah digunakan. Perancangan struktur menu dari pembangunan aplikasi antenna pointer adalah sebagai berikut:

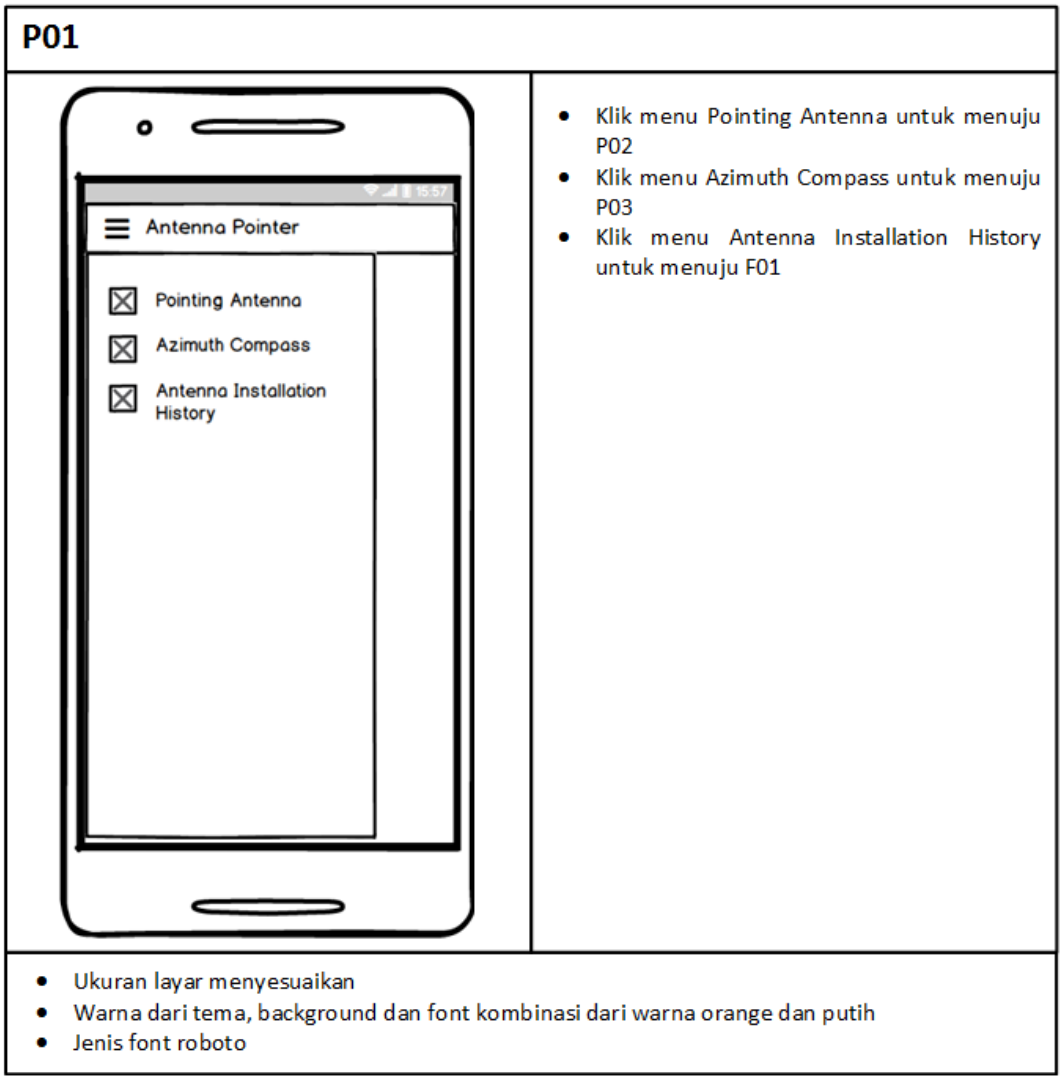


Gambar 3.32 Struktur Menu Aplikasi *Antenna Pointer*

3.2.4 Perancangan Antarmuka

Perancangan antarmuka mendeskripsikan rencana tampilan dari setiap tampilan dan form yang akan digunakan pada pembangunan aplikasi antenna pointer. Perancangan antarmuka pada aplikasi antenna pointer ini dapat dilihat pada gambar-gambar berikut:

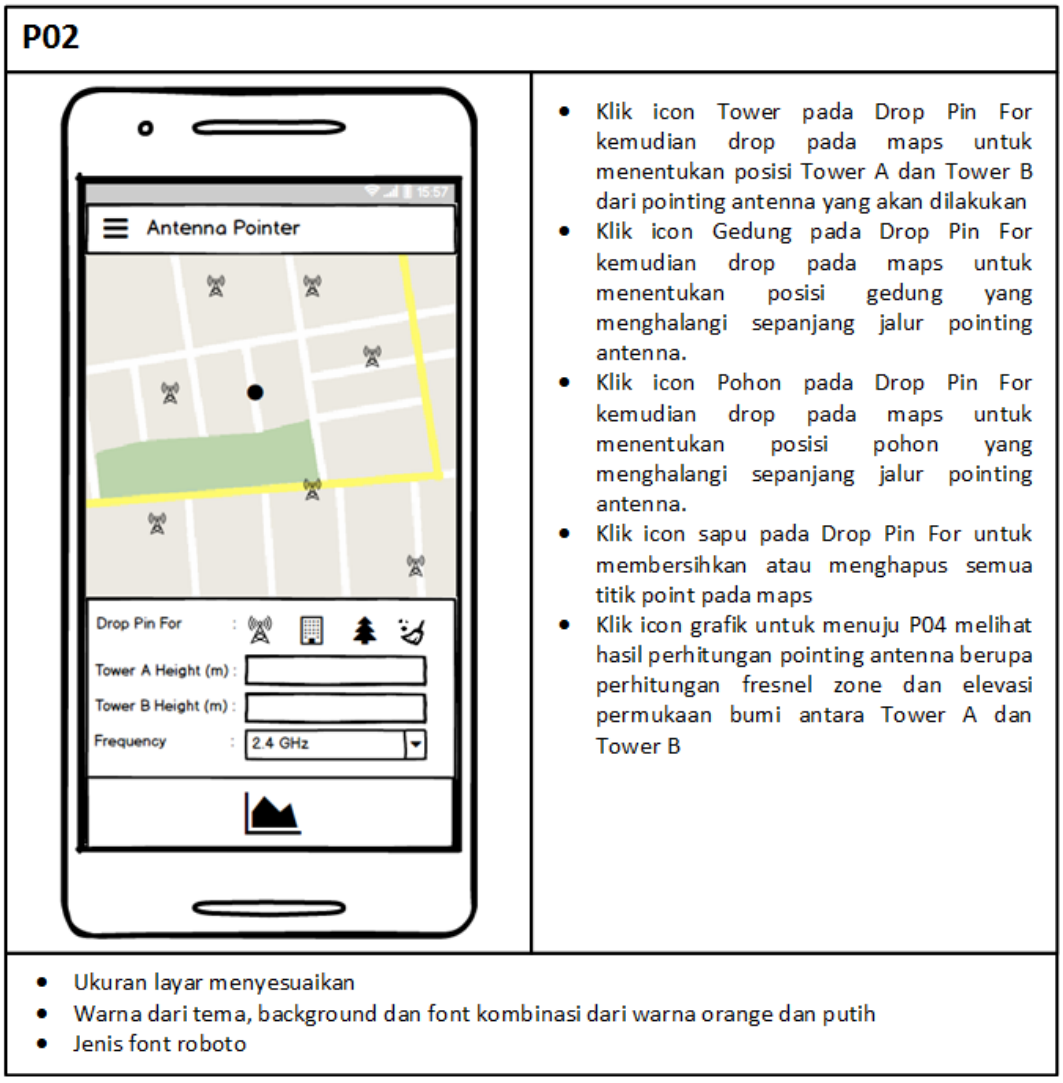
1. Menu Utama



Gambar 3.33 Antarmuka Menu Utama

2. Halaman *Pointing Antenna*

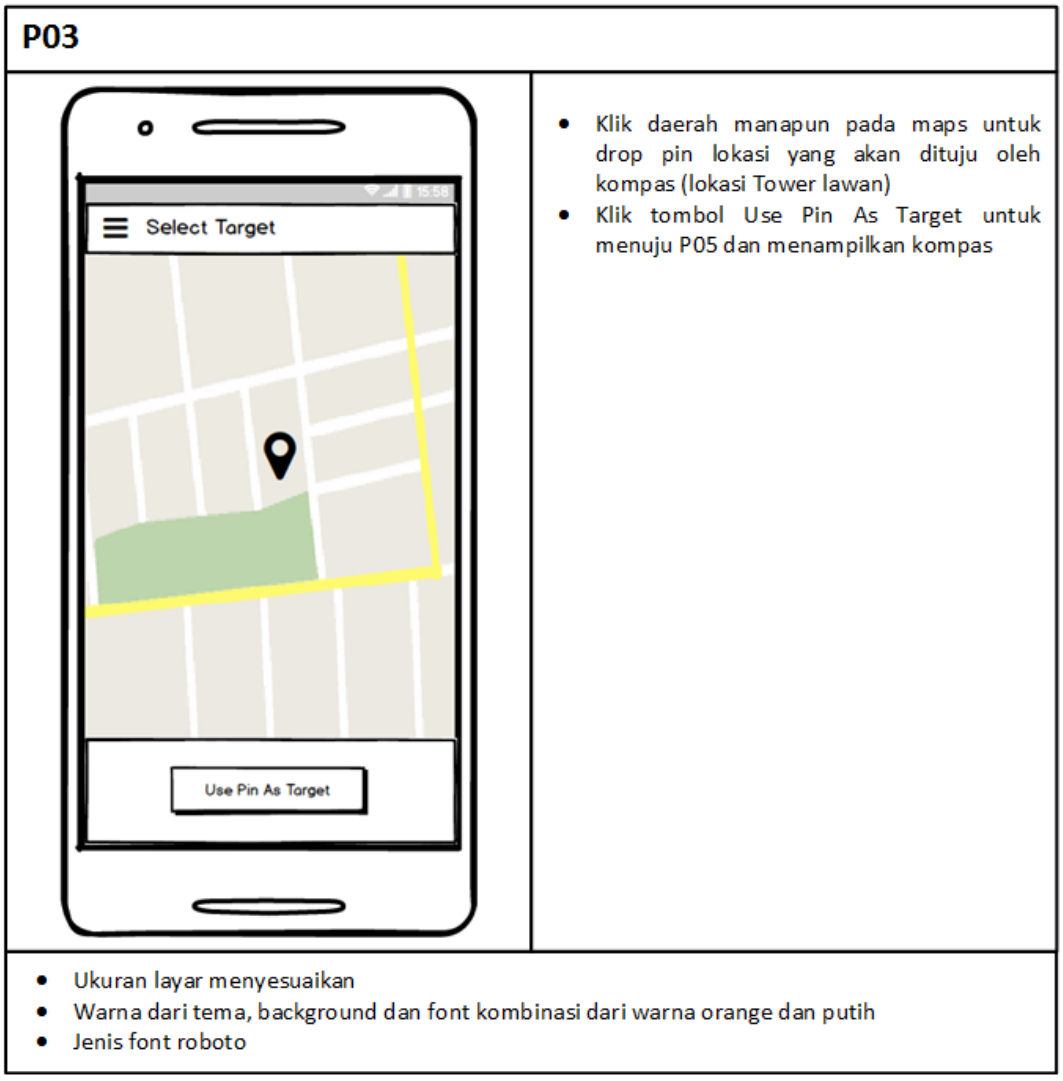
   Pada antarmuka ini digambarkan bagaimana tampilan dari halaman pointing antenna pada sub sistem mobile yaitu pada aplikasi android yang akan digunakan pengguna. Adapun antarmuka login dapat dilihat pada Gambar 3.26 berikut:



Gambar 3.34 Antarmuka Halaman *Pointing Antenna*

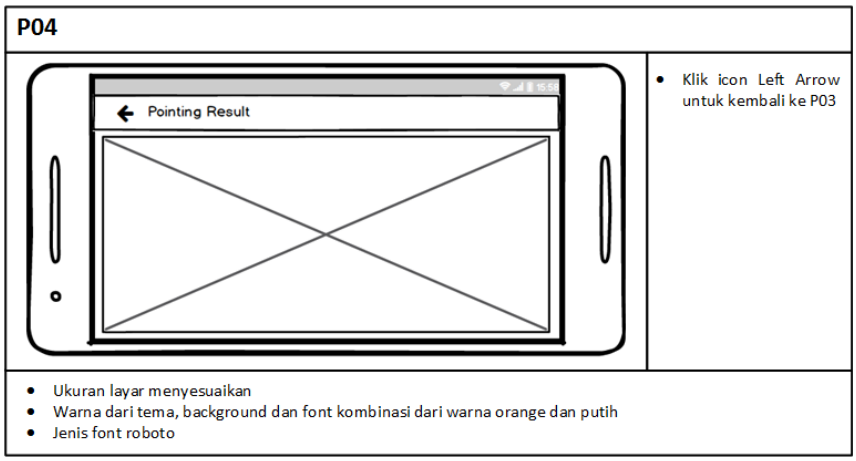
3. Halaman *Azimuth Compass*

   Pada antarmuka ini digambarkan bagaimana tampilan dari halaman *azimuth compass* pada sub sistem mobile yaitu pada aplikasi android yang akan digunakan pengguna. Adapun antarmuka login dapat dilihat pada Gambar 3.27 berikut:



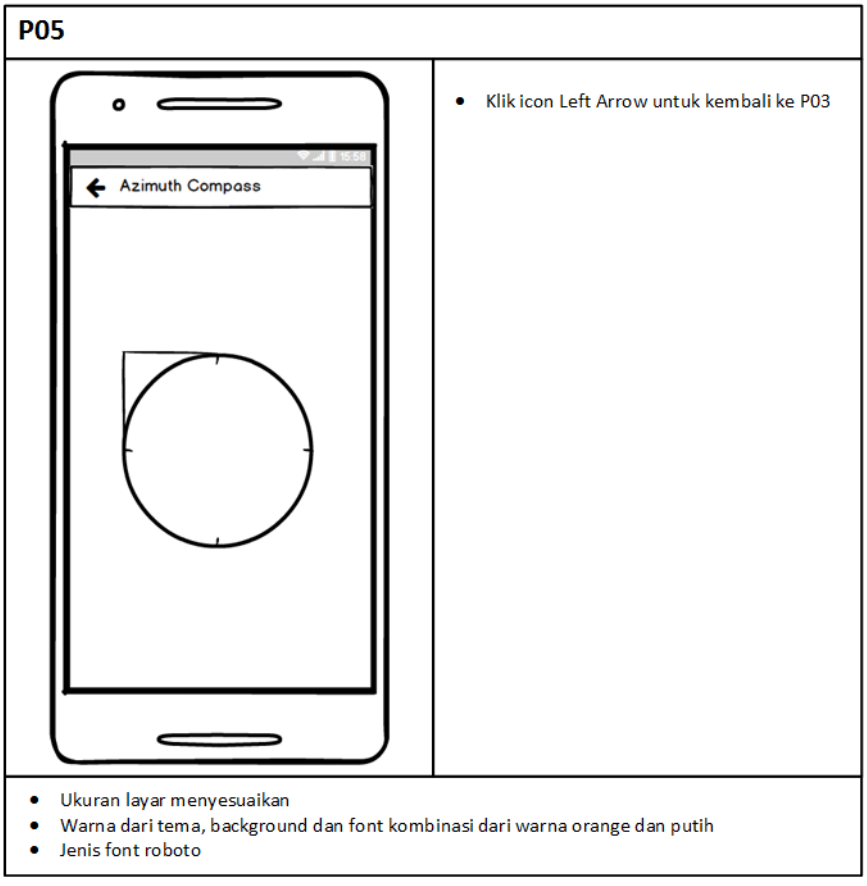
Gambar 3.35 Antarmuka Halaman *Azimuth Compass*

4. Halaman Hasil Pointing Antenna



Gambar 3.36 Antarmuka Halaman Hasil *Pointing Antenna*

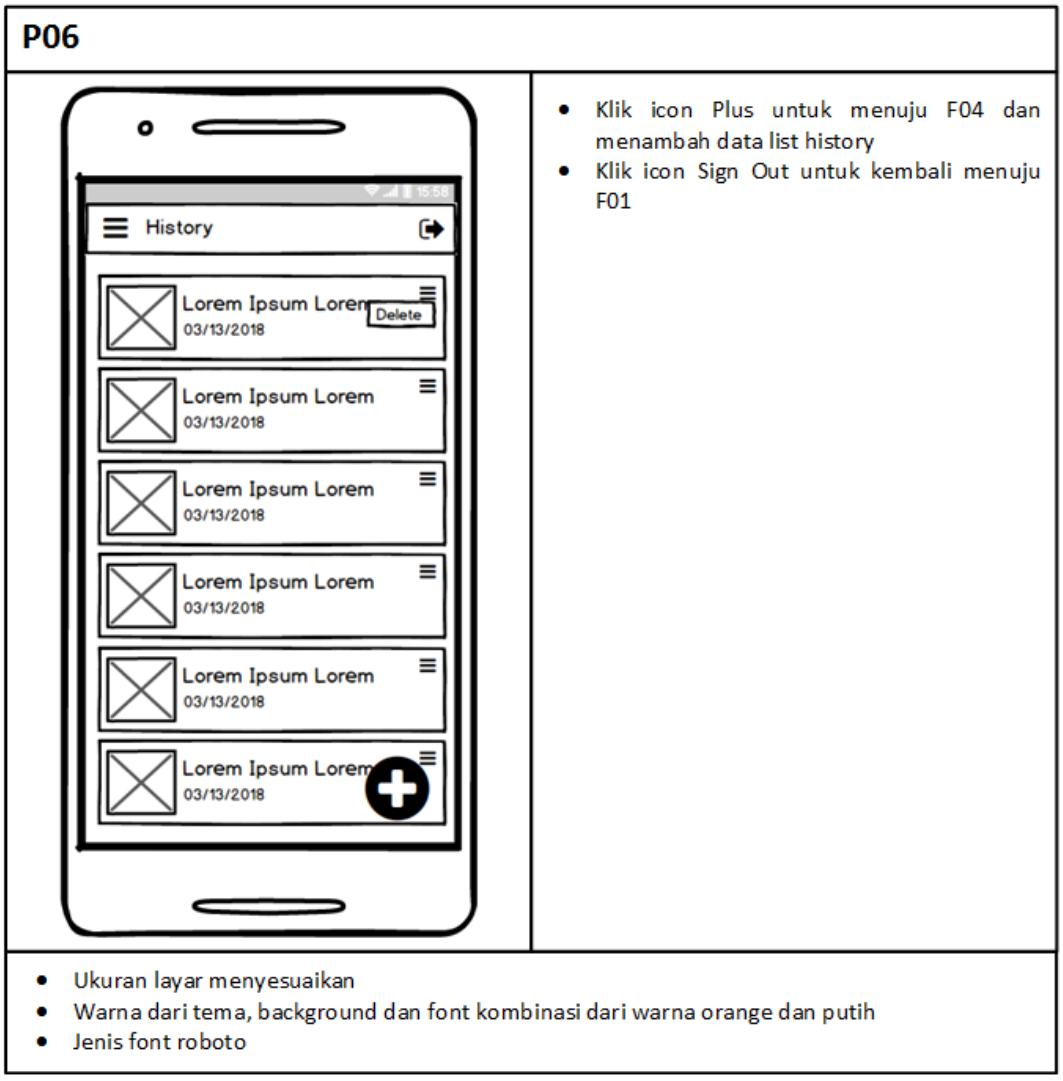
5. Halaman Gambar Kompas



Gambar 3.37 Antarmuka Halaman Gambar Kompas

6. Halaman Lihat *History* Instalasasi Antena

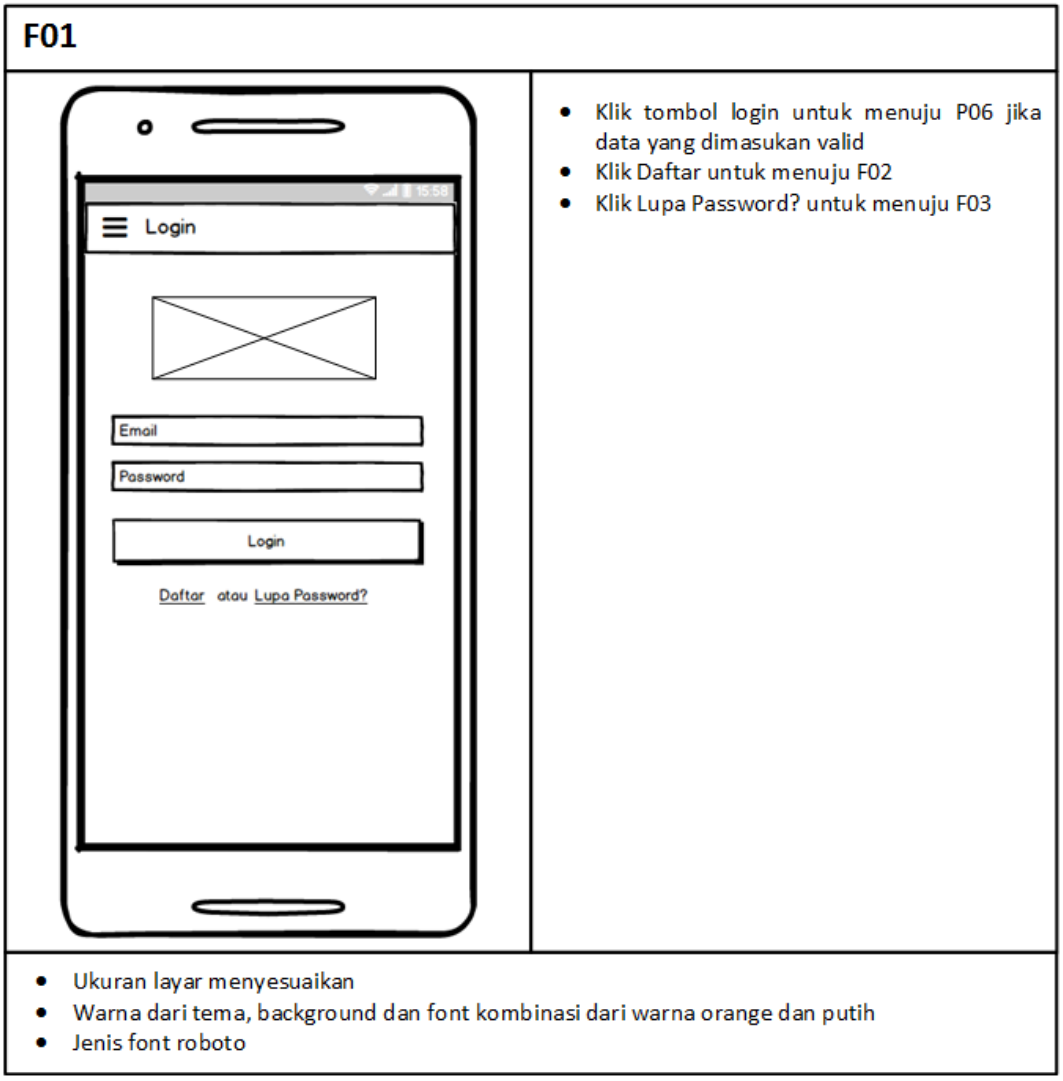
   Pada antarmuka ini digambarkan bagaimana tampilan dari halaman lihat *history* instalasi antenna pada sub sistem mobile yaitu pada aplikasi android yang akan digunakan pengguna. Adapun antarmuka login dapat dilihat pada Gambar 3.30 berikut:



Gambar 3.38 Antarmuka Halaman Lihat *History* Instalasi Antenna

7. Form *Login*

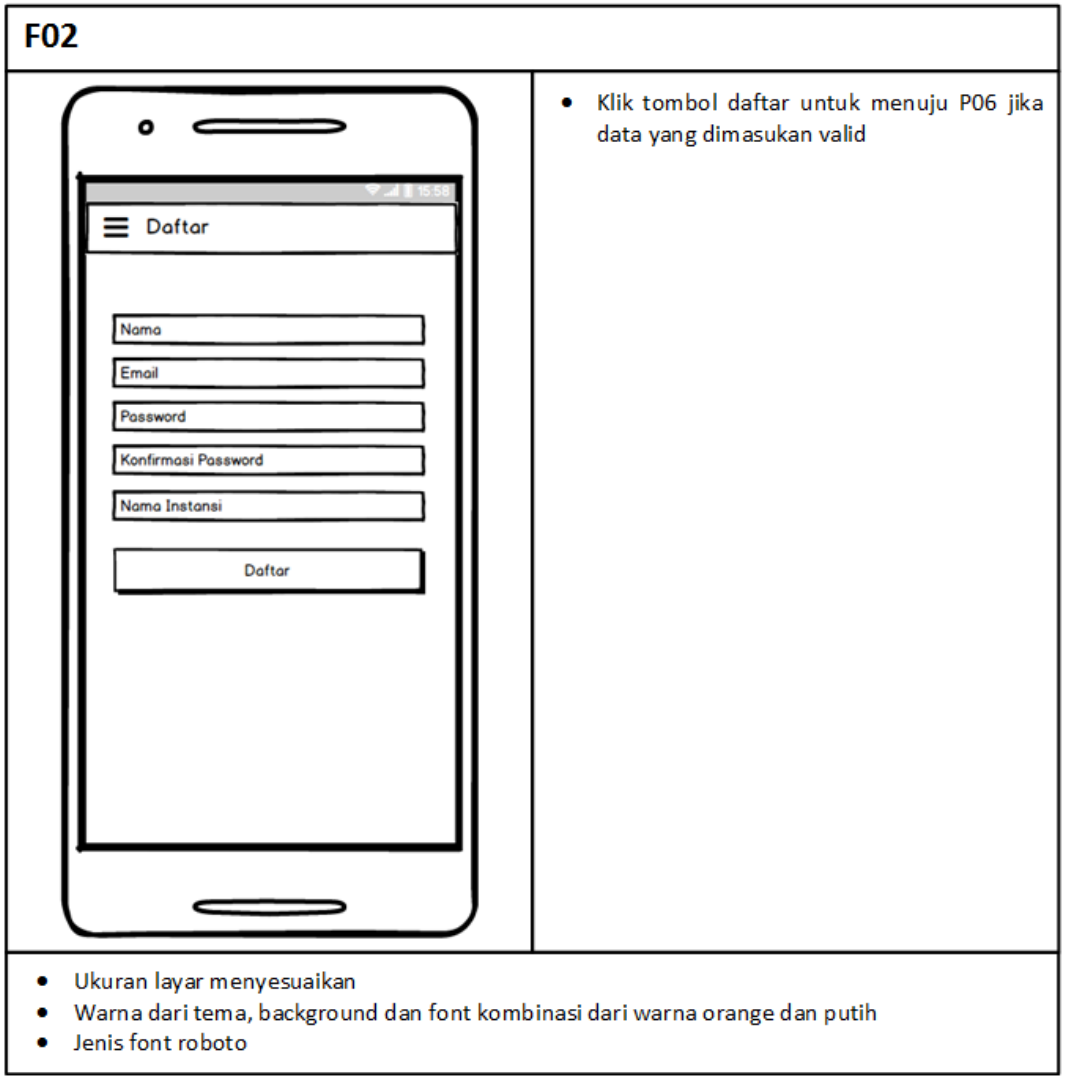
   Pada antarmuka ini digambarkan bagaimana tampilan dari form *login* pada sub sistem mobile yaitu pada aplikasi android yang akan digunakan pengguna. Adapun antarmuka login dapat dilihat pada Gambar 3.31 berikut:



Gambar 3.39 Antarmuka Form *Login*

8. Form Daftar

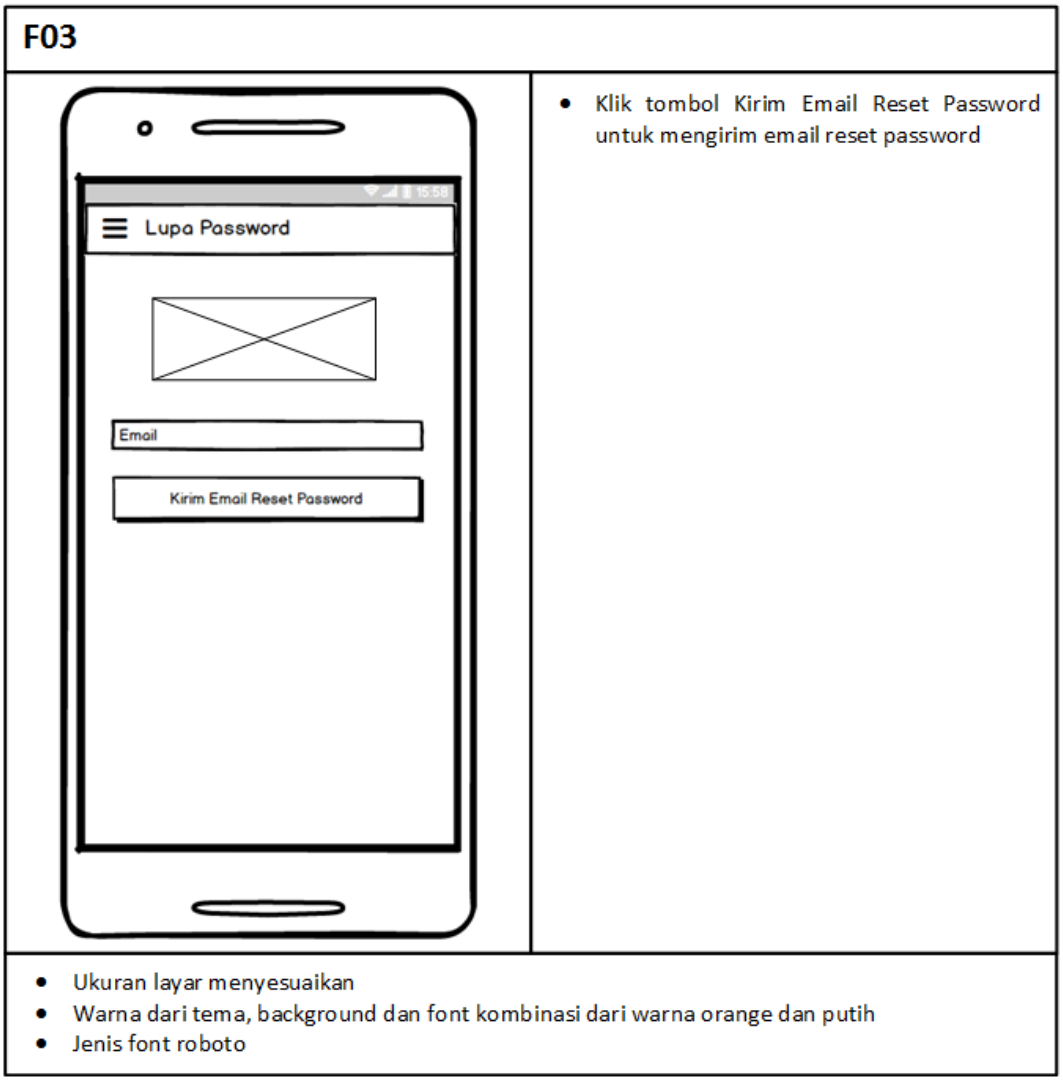
   Pada antarmuka ini digambarkan bagaimana tampilan dari form daftar pada sub sistem mobile yaitu pada aplikasi android yang akan digunakan pengguna. Adapun antarmuka login dapat dilihat pada Gambar 3.31 berikut:



Gambar 3.40 Antarmuka Form Daftar

9. Form Lupa *Password*

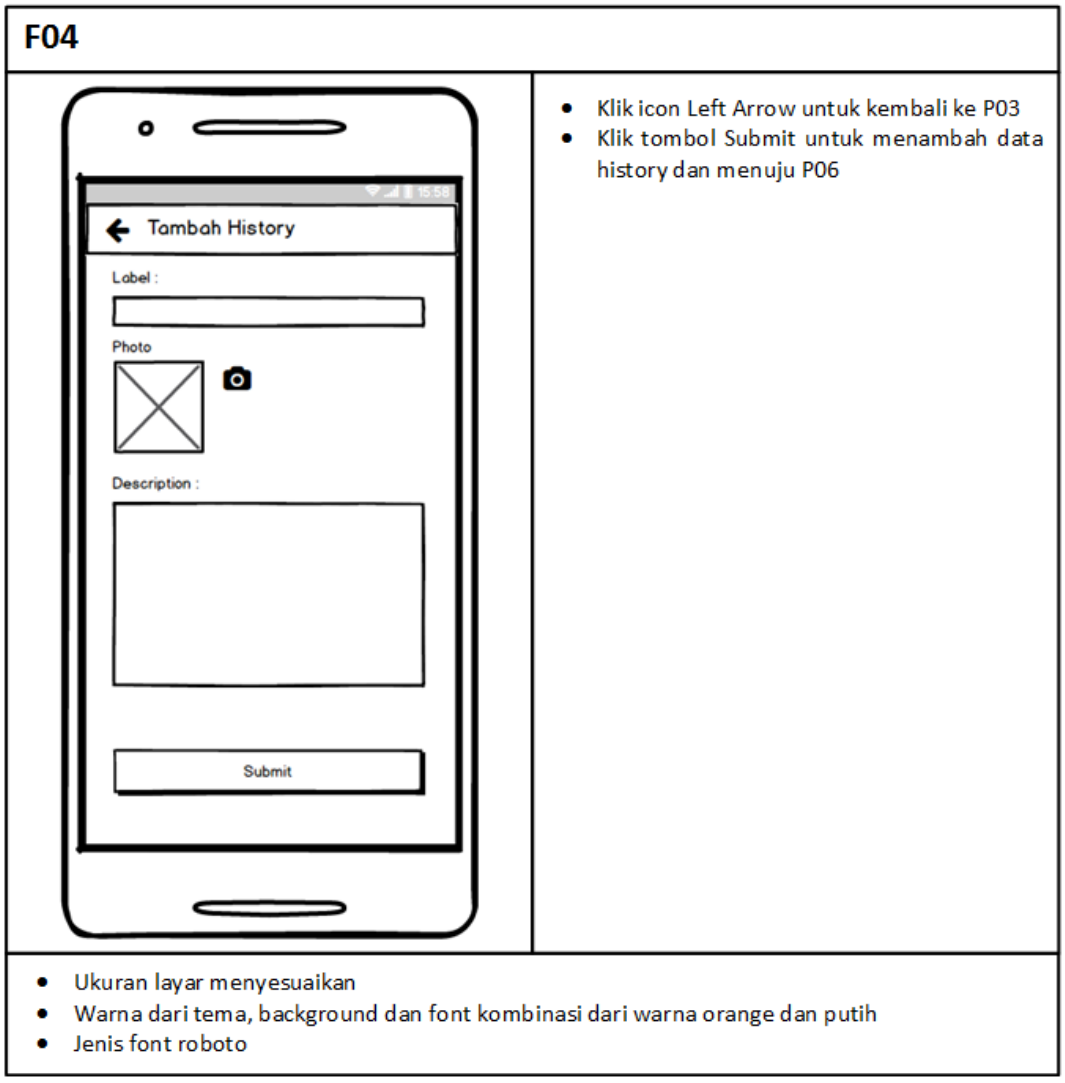
   Pada antarmuka ini digambarkan bagaimana tampilan dari form lupa *password* pada sub sistem mobile yaitu pada aplikasi android yang akan digunakan pengguna. Adapun antarmuka login dapat dilihat pada Gambar 3.31 berikut:



Gambar 3.41 Antarmuka Form Lupa *Password*

10. Form Tambah *History*

Pada antarmuka ini digambarkan bagaimana tampilan dari form tambah *history* pada sub sistem mobile yaitu pada aplikasi android yang akan digunakan pengguna. Adapun antarmuka login dapat dilihat pada Gambar 3.31 berikut:



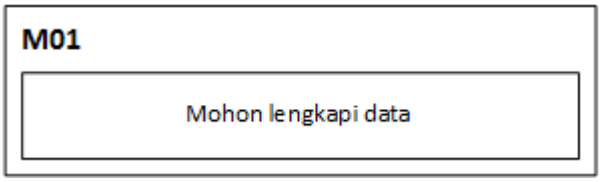
Gambar 3.42 Antarmuka Form Tambah *History*

3.2.5 Perancangan Pesan

Perancangan pesan pada sistem bertujuan untuk memberikan rancangan pesan penanganan kesalahan atas kesalahan yang mungkin dilakukan oleh pengguna. Terdapat beberapa perancangan pesan pada aplikasi antenna pointer sebagai berikut:

1. Pesan data masukan tidak boleh kosong

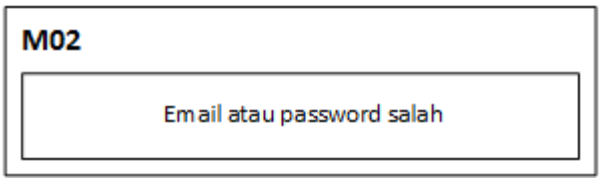
   Pesan ini akan muncul jika pengguna tidak mengisi *field input* yang seharusnya diisi atau terdapat data *input* yang kosong pada form daftar ataupun form tambah data *history*.



Gambar 3.43 Pesan Data Masukan Tidak Boleh Kosong

2. Pesan kesalahan pada *login*

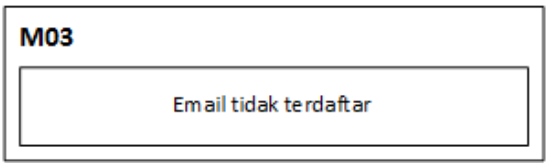
   Pesan ini akan muncul di halaman login jika pengguna salah memasukan data *login* baik itu *username* ataupun *password* ketika akan melakukan login kedalam aplikasi antenna pointer.



Gambar 3.44 Pesan Kesalahan Pada Login

3. Pesan kesalahan pada form lupa *password*

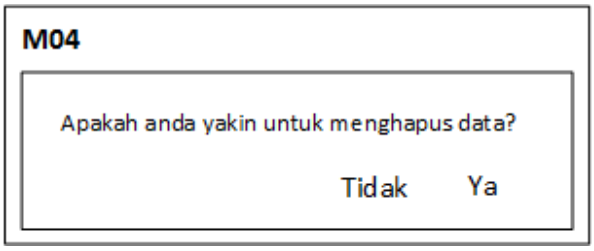
   Pesan ini akan muncul di halaman lupa *password* jika pengguna salah memasukan *email* atau *email* yang di input tidak *valid* dan tidak terdaftar pada aplikasi antenna pointer.



Gambar 3.45 Pesan Kesalahan Pada Form Lupa *Password*

4. Pesan konfirmasi hapus data *history*

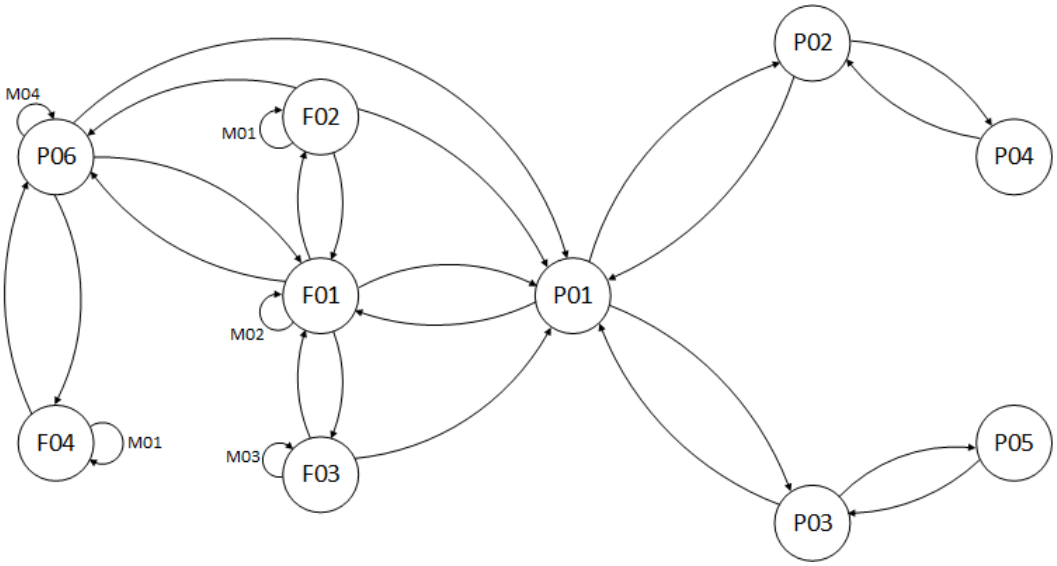
   Pesan ini akan muncul di halaman *history* instalasi antenna jika pengguna hendak menghapus data *history*. Pesan yang muncul berupa konfirmasi untuk melanjutkan proses menghapus data *history* atau membatalkannya.



Gambar 3.46 Pesan Konfirmasi Menghapus Data

3.2.6 Jaringan Semantik

Jaringan semantik aplikasi yang dibangun dapat dilihat pada gambar dibawah ini:



Gambar 3.47 Jaringan Semantik Aplikasi *Antenna Pointer*