BAB III

ANALISIS DAN PERANCANGAN SISTEM

Analisis dan perancangan sistem memerlukan tahapan yang sistematis untuk men-dapatkan rancangan sistem yang baik dan sesuai dengan kegunaan dan tujuannya. Tahap awal dari analisis adalah menganalisa kebutuhan-kebutuhan sistem mulai dari kebutuh-an pengguna, kebutuhan non fungsional, dan kebutuhan fungsional. Sedangkan untuk tahap perancangan aplikasi yaitu perancangan database dan peranc angan antarmuka.

3.1 Analisis Sistem

Analisis sistem didefinisikan sebagai penguraian dari suatu sistem informasi yang utuh ke dalam bagian-bagian komponennya dengan maksud untuk meng-identifikasi dan mengevaluasi permasalahan, kesempatan, hambatan yang terjadi dan kebutuhan yang diharapkan sehingga dapat diusulkan perbaikan sesuai dengan kebutuhan.

3.1.1 Analisis Masalah

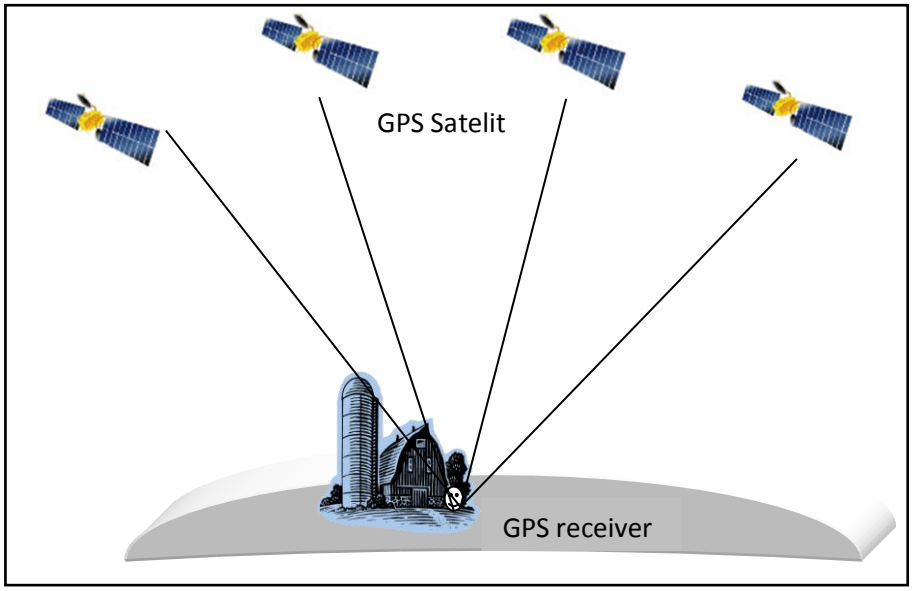
Berdasarkan pada solusi permasalahan dari aplikasi sebelumnya yang dianggap masih memiliki kekurangan dalam menginformasikan fasilitas umum yaitu bagaimana informasi ditampilkan dalam aplikasi kepada pengguna. Maka dilakukan rancangan ulang untuk membangun aplikasi dan memberikan tampilan antarmuka aplikasi yang baru yang mudah dioperasikan sehingga informasi yang diinformasikan mudah untuk dicerna oleh pengguna.

Dalam prinsip-prinsip perancangan antarmuka (*user interface*), sebuah aplikasi dituntut untuk dapat memberikan informasi sesuai dengan kebutuhan pengguna, mudah dimengerti dan dioperasikan sehingga fungsionalitas dari aplikasi dapat digunakan dengan maksimal oleh pengguna.

Keunggulan dari teknologi *augmented reality* dalam segi menampilkan suatu informasi dapat diterapkan kedalam aplikasi untuk dapat memberikan solusi dari permasalahan dari aplikasi sebelumnya dan tujuan yang ingin dicapai dari perancangan ulang aplikasi dapat tercapai. Dengan demikian, tujuan dari pembangunan aplikasi yaitu memberikan kemudahan kepada pengguna untuk mendapatkan atau mengenalkan informasi-informasi mengenai lokasi fasilitas umum seperti kuliner, rumah sakit, toko, ATM, SPBU, dan *factory outlet* yang berada disekiling pengguna dapat tercapai.

3.1.2 Analisis Penentuan Posisi Dengan GPS

Penentuan posisi dengan GPS adalah reseksi (pengikatan kebelakang) dengan jarak, yaitu dengan pengukuran jarak secara simultan ke beberapa satelit GPS yang koordinatnya telah diketahui. Untuk menentukan koordinat suatu titik di bumi, *receiver* setidaknya membutuhkan 4 satelit yang dapat ditangkap sinyalnya dengan baik. Secara *default* posisi atau koordinat yang diperoleh bereferensi ke *global datum* yaitu *World Geodetic System* 1984 atau disingkat WGS’84.



Gambar III.1 Ilustrasi komunikasi GPS receiver dengan satelit GPS

Secara garis besar penentuan lokasi posisi dengan GPS ini dibagi menjadi dua metode yaitu metode absolut dan metode relatif.

1. Metode absolut atau dikenal sebagai *point positioning*, menentukan posisi hanya berdasarkan pada satu alat penerima (*receiver*) saja. Ketelitian posisi dalam beberapa meter (tingkat akurasi rendah) dan umumnya hanya diperuntukkan bagi keperluan navigasi.
2. Metode relatif atau sering disebut *differential positioning*, menentukan posisi dengan menggunakan lebih dari satu *receiver*. Satu GPS dipasang pada lokasi tertentu dimuka bumi dan secara terus menerus menerima sinyal dari satelit dalam jangka waktu tertentu yang dijadikan sebagai referensi bagi yang lainnya. Metode ini menghasilkan posisi dengan tingkat akurasi tinggi (umumnya kurang dari 1 meter) dan diaplikasikan untuk keperluan survei geodesi ataupun pemetaan yang memerlukan tingkat akurasi tinggi.

Untuk keperluan penelitian ini digunakan metode absolut yang menggunakan *single receiver* tipe navigasi karena tujuan penggunaannya hanya untuk pengguna individu serta perangkat yang digunakan merupakan perangkat dengan *single receiver* yaitu smartphone.

1) Ketelitian penentuan posisi dengan menggunakan GPS antara lain :
   1 GPS dapat digunakan setiap saat tanpa bergantung waktu dan cuaca.
   2 GPS dapat digunakan oleh banyak orang pada waktu yang sama dan pemakaiannya tidak bergantung pada batas politik dan alam.
   3 Penggunaan GPS dalam penentuan posisi secara relatif tidak bergantung dengan kondisi topografis daerah survei.
   4 Posisi yang ditentukan dengan GPS mengacu ke datum global yang dinamakan *World Geodetic System 1984* (WGS'84). Dengan kata lain posisi yang diberikan oleh GPS akan selalu mengacu ke datum yang sama.
   5 Pemakaian sistem GPS tidak dikenakan biaya, setidaknya sampat saat ini.
   6 Alat *receiver* GPS untuk penentuan posisi suatu titik relative lebih mudah dan tidak mengeluarkan biaya.
   7 Data pengamatan GPS sukar untuk dimanipulasi.

2) Faktor-faktor yang Mempengaruhi Ketelitian Posisi GPS :

Penentuan posisi dengan GPS dapat mengalami gangguan atau interferensi yang mengakibatkan penentuan posisi menjadi tidak akurat. Berikut beberapa faktor yang mempengaruhi keakurasian penentuan posisi dengan GPS :

1) Ketelitian Data :
   1 Tipe data yang digunakan.

2 Kualitas receiver GPS yang digunakan.

3 Level dari kesalahan dan bias.

2) Geometri Satelit :

1 Jumlah satelit.

2 Lokasi dan distribusi satelit.

3 Lama pengamatan.

3) Metode Penentuan Posisi

1) Absolute dan *differential positioning.*

2) Static, rapid static, pseudo-kinematic, stop-and-go, kinematic.

3) One & multi monitor stations.

4) Strategi Pemrosesan Data :

1 Realtime & positioning processing.

2 Strategi eliminasi dan pengkoreksian kesalahan dan bias.

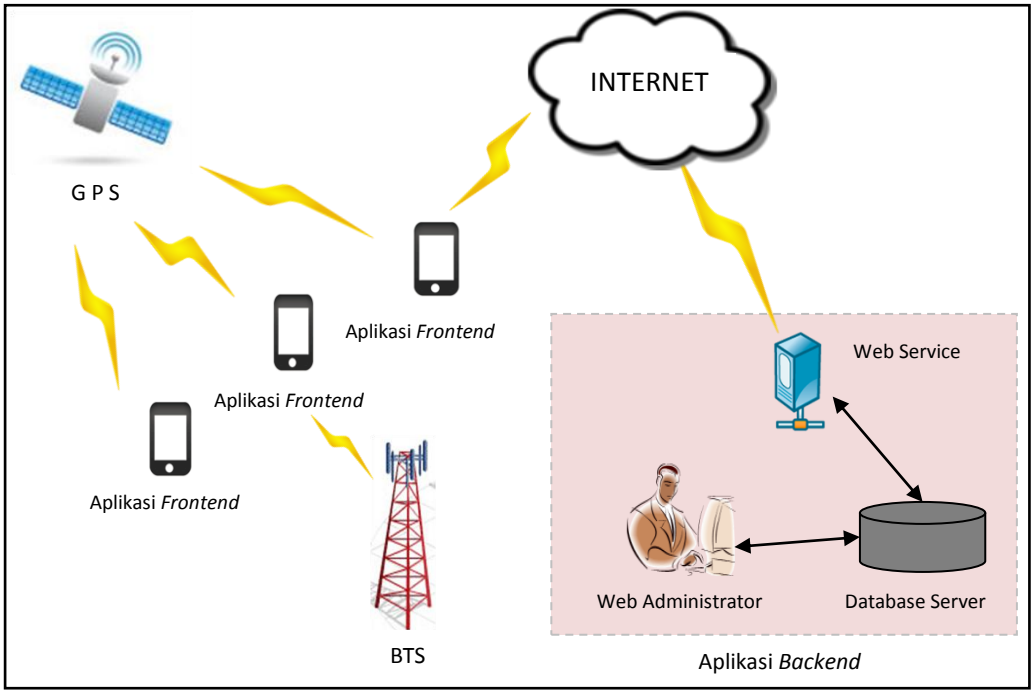
3 Metode estimasi yang digunakan.

4 Pemrosesan *baseline* & perataan jaringan.

5 Kontrol kualitas.

3.1.3 Analisis Arsitektur Sistem

Arsitektur fisik sistem yang dibangun menggunakan model *two tier*. Arsitektur fisik sistem terdiri dari komponen berikut yaitu aplikasi *ARPublicFacilities* (*frontend*), web admin, web service, dan database server (backend). Arsitektur sistem dalam pembangungan aplikasi fasilitas umum ini ditunjukkan pada gambar III.2.



Gambar III.2 Arsitektur Sistem

Frontend adalah aplikasi fasilitas umum yang sudah terinstal dalam perangkat *mobile* yang digunakan (perangkat android). Aplikasi ini berfungsi sebagai antarmuka antara pengguna dengan sistem. Pengembangan antarmuka sistem, dibangun dengan menggunakan IDE Eclipse dan SDK Android dengan bahasa pemrograman java android.

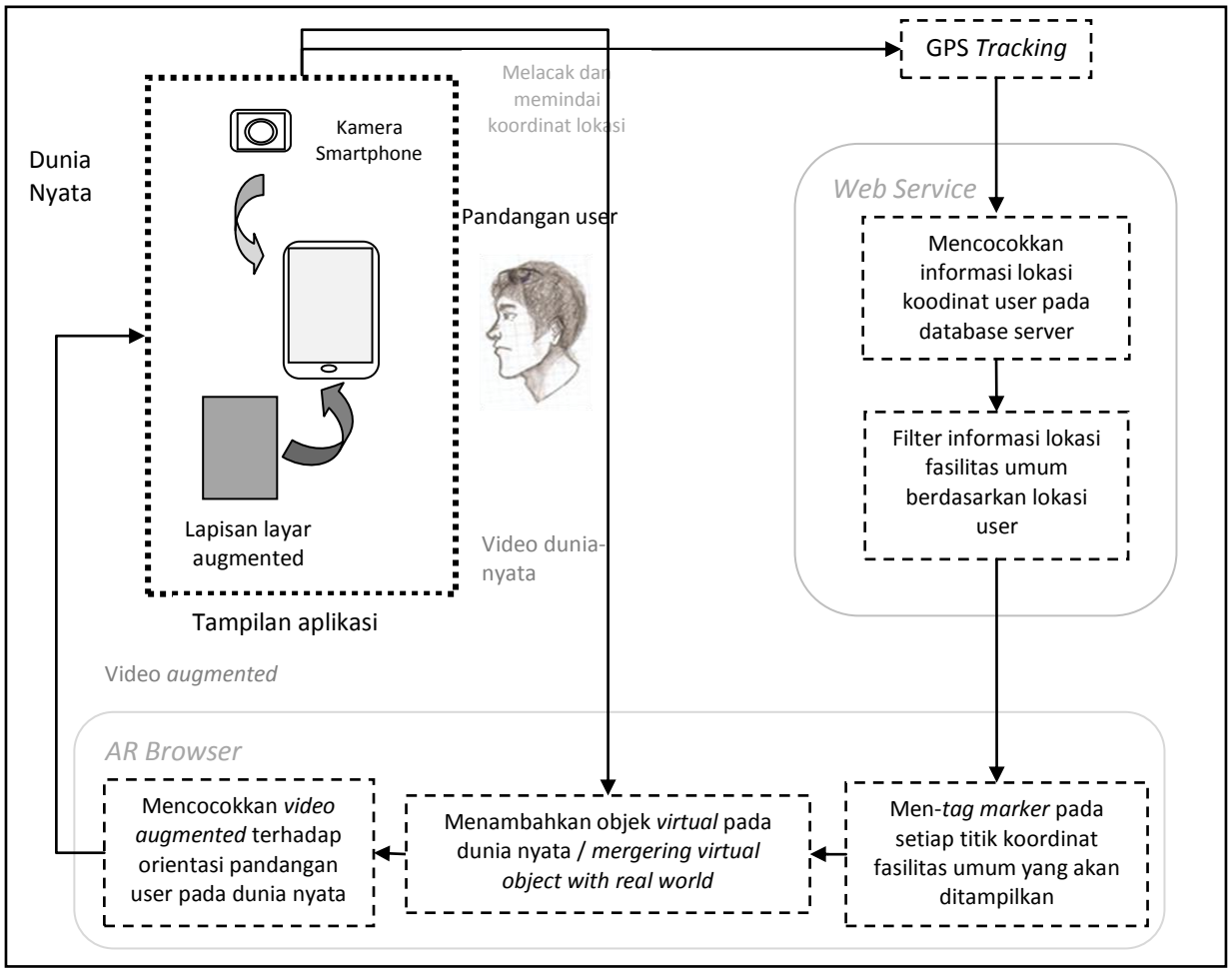
Backend adalah aplikasi pendukung yang berfungsi sebagai penyuplai atau sumber data pada aplikasi *frontend*. Aplikasi ini terdiri dari tiga subbagian utama yaitu web administrator, web service, dan database server. Web administrator merupakan web yang berfungsi sebagai pengolah sumber basis data yang terdapat pada database server yang dikelola oleh admin. Web administrator dibangun dengan menggunakan bahasa pemrograman PHP. Web service merupakan web yang berfungsi sebagai media pertukaran data antar database server dan aplikasi. Web service dalam penelitian ini dibangun dengan menggunakan pemrograman

PHP yang dikombinasikan dengan JSON (*Javascript Object Notation*). Database server merupakan aplikasi yang berfungsi sebagai sumber data (*data resources*) aplikasi. Pada bagian ini pengelolaan database menggunakan menggunakan MySQL. Akses koneksi jaringan yang digunakan dari *frontend* untuk terhubung ke web service melalui perangkat mobile menggunakan jaringan seluler. GPS receiver yang terdapat dalam perangkat *mobile* terhubung ke satelit GPS yang berfungsi untuk menentukan posisi pengguna di bumi, sehingga dari informasi ini, informasi fasilitas umum yang disampaikan relevan terhadap wilayah dimana pengguna berada.

3.1.4 Analisis Sistem *Augmented Reality Display*

Seperti yang dijelaskan dalam bab II, *augmented reality* adalah teknologi yang menggabungkan dunia maya dengan objek virtual dalam waktu nyata (*real-time*), sehingga menghasilkan suatu pandangan baru terhadap lingkungan dunia nyata. Dalam pembangunan aplikasi ini, sistem pelacakan lokasi yang digunakan adalah metode pelacakan GPS (GPS *based tracking*) untuk mencari atau memindai titik koordinat setiap lokasi fasilitas umum pada suatu wilayah yang sebelumnya telah terdaftar dalam database server. Ketika lokasi *user* telah ditentukan, maka aplikasi akan mengirim data lokasi tersebut ke webservice untuk ditentukan *konten* yang terdekat dengan lokasi pengguna berada. Dari setiap koordinat (*latitude* dan *longitude*) lokasi konten tersebut, akan di *tag* sebuah gambar dalam hal ini penanda (*marker*) pada setiap titik koordinat fasilitas umum yang nantinya melalui kamera akan ditampilkan dalam layar *handphone*. Untuk menampilkan titik koordinat fasilitas umum yang telah di *tag* dengan sebuah penanda dalam layar *handphone*, digunakan mixare AR *browser* (*augmented*

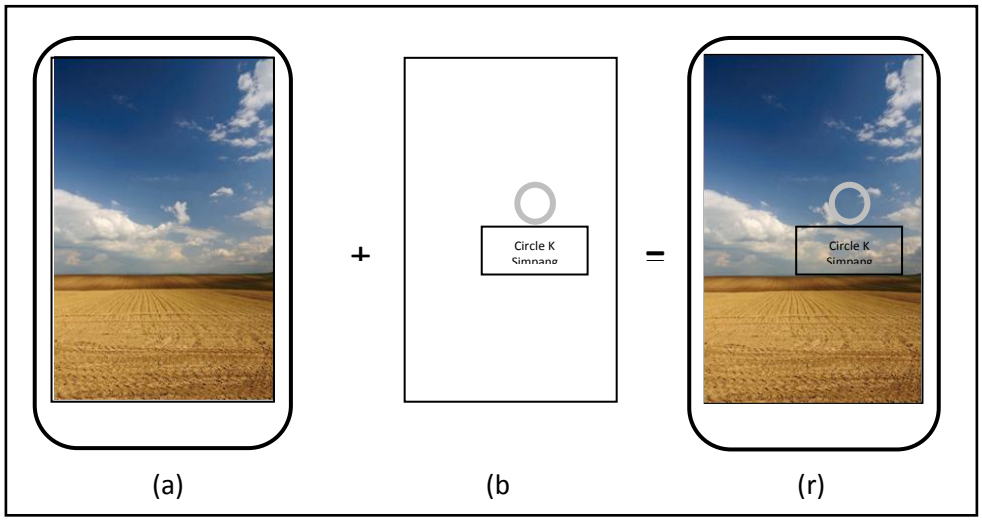
reality engine). AR browser merupakan aplikasi perangkat lunak yang bertujuan untuk mengambil, menampilkan, dan melintaskan sumber informasi melalui antarmuka *augmented reality*. Berikut gambar yang menggambarkan sistem *augmented reality*.



Gambar III.3 Arsitektur Sistem *ARPublicFacilities*

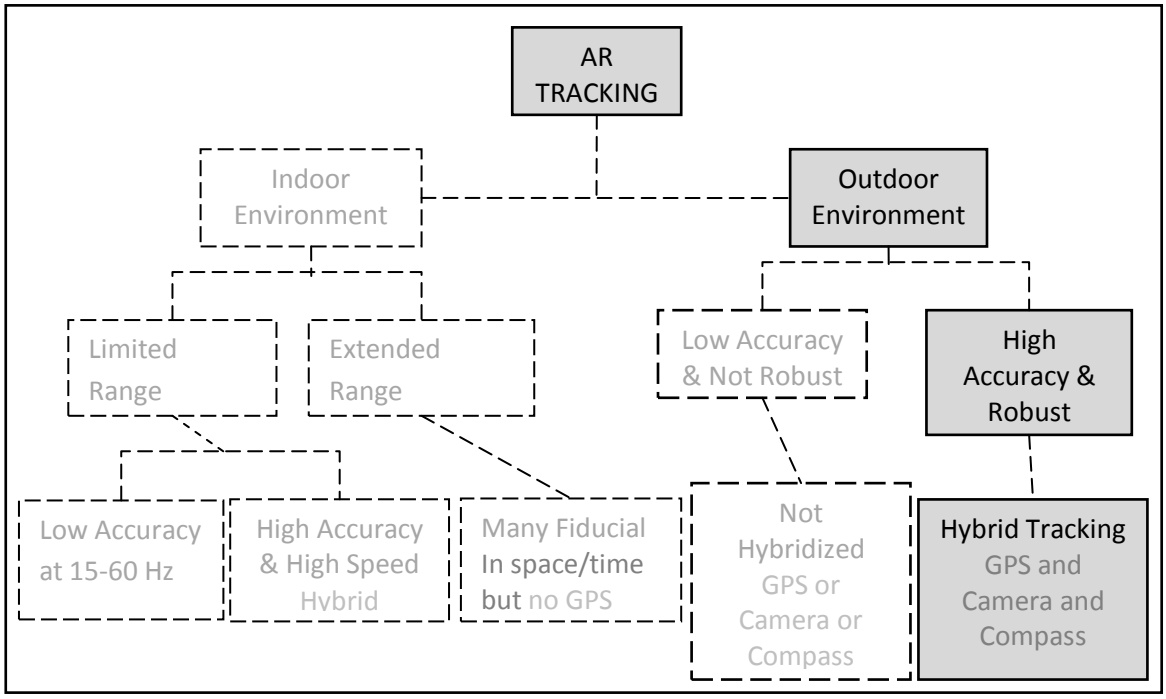
Pada gambar III.3, diasumsikan bahwa posisi keberadaan pengguna di suatu wilayah telah ditentukan. Aplikasi akan memindai atau melacak titik koordinat fasilitas umum yang sebelumnya telah terdaftar dalam database server, kemudian akan mencocok data informasi dengan wilayah dimana pengguna berada, sehingga informasi yang ditampilkan relevan. Kini informasi fasilitas umum yang

akan ditampilkan telah ditentukan, selanjutnya setiap titik koordinat fasilitas umum akan ditempelkan (*tag*) sebuah penanda dimana prosesnya menghasilkan objek maya (*virtual object*). Objek maya kemudian akan dilapiskan atau digabungkan dengan layar video dunia nyata yang ditangkap oleh pengguna melalui kamera pada *handphone*, tahap terakhir dilakukan kalibrasi *compass* (mencocokkan orientasi posisi kamera terhadap lingkungan disekitar pengguna) proses ini menghasilkan video *augmented*. Tahap ini merupakan hasil akhir penggabungan yang akan pengguna lihat dalam layar *handphone* yang digunakan, sebuah tampilan penyisipan informasi terhadap fasilitas umum dalam dunia nyata.



Gambar III.4 (a) Layar dunia nyata yang ditangkap kamera, (b) Layar hasil proses *augmented*, (r) Tampilan *Augmented* : Menggabungkan layar dunia nyata dan layar *augmented*

3.1.5 Analisis *Augmented Reality Tracking System*



Gambar III.5 *AR Tracking Taxonomy*

Gambar III.5, merupakan diagram *taxonomy* AR *tracking* berdasarkan lingkungan atau area pengembangan. Dalam penelitian ini, area pengembangannya adalah pada lingkungan terbuka (*outdoor environment*). Tujuan dari pengembangan ini adalah untuk memberikan informasi mengenai suatu wilayah, dalam hal ini kota Bandung, dan informasi yang diinformasikan adalah fasilitas umum. Pada gambar III.5 telah dipetakan penggunaan AR *tracking* berdasarkan lingkungan pengembangan dan tingkat metode pengembangannya. *Hybrid tracking system* merupakan metode pencarian yang paling sesuai, dalam penelitian ini karena lingkungan atau area penelitian yang sangat luas dan dibutuhkan metode pelacakan untuk mencakup wilayah penelitian maka digunakan pelacakan GPS (*GPS Based Tracking*). Oleh karena itu, dibutuhkan data mengenai wilayah penelitian sebagai sumber data yang akan

digunakan untuk mengidentifikasi wilayah tersebut, data-data tersebut adalah data informasi geografis (berupa titik koordinat dari setiap fasilitas umum yang akan diinformasikan). Data-data ini akan menjadi objek pelacakan (*tracking object*) dari aplikasi dengan menggabungkan beberapa sensor yang terdapat dalam perangkat seperti GPS, *accelerometer*, *digital compass*.

3.1.6 Analisis Implementasi *Augmented Reality Browser* (ARBrowser) Mixare

Analisis implementasi *augmented reality browser* (ARBrowser) Mixare merupakan analisis untuk mendeskripsikan bagaimana proses *augmented virtual object* kedalam layar perangkat yang digunakan (*smartphone*).

3.1.6.1 Virtual Object (Marker)

Virtual object (benda maya) adalah suatu objek yang dihasilkan melalui proses komputerisasi yang hasilnya dapat berupa objek 2 dimensi atau 3 dimensi. Sesuai dengan pengembangan aplikasi yang dibangun menggunakan ARBrowser Mixare, yaitu benda maya yang digunakan berbentuk lingkaran yang disebut *marker*. *Marker* ini berfungsi sebagai penanda letak suatu objek nyata yang diinformasikan terhadap pandangan dunia nyata. Implementasi *marker* berupa lingkaran menandakan keberadaan fasilitas umum di suatu wilayah dimana terjadi tahap perubahan bentuk *marker* berdasarkan jarak radius antara posisi pengguna dan keberadaan fasilitas umum.



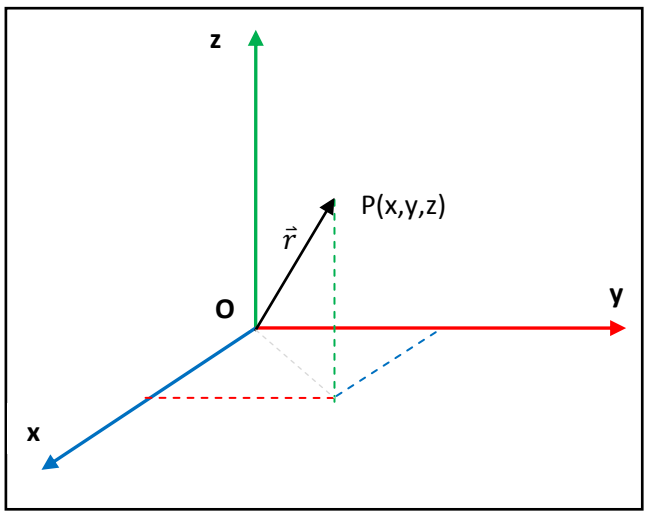
Gambar III.6 (a) Object A dengan jarak 1.0 km, (b) Object B dengan jarak 500m, (c) Object C dengan jarak 70m

3.1.6.2 Layar dan *Virtual Object* (*Embedded Marker Into Screen*)

Penempatan *virtual object* (*marker*) ke dalam layar perangkat yang digunakan terhadap orientasi pandangan pengguna pada lingkungan nyata diimplementasikan dengan menggunakan sistem koordinat 3 dimensi (*three-dimensional coordinate system*) dengan vektor dan matriks.

3.1.6.3 Geometri dalam Ruang, Vektor

Vektor dalam matematika dan fisika adalah obyek geometri yang memiliki besar dan arah. Vektor digunakan untuk mendeskripsikan jarak sebuah sebuah titik dalam ruang (3 dimensi), yang titik-titiknya direpresentasikan dengan sistem koordinat kartesian (*coordinate cartesian system*), P(x,y,z).

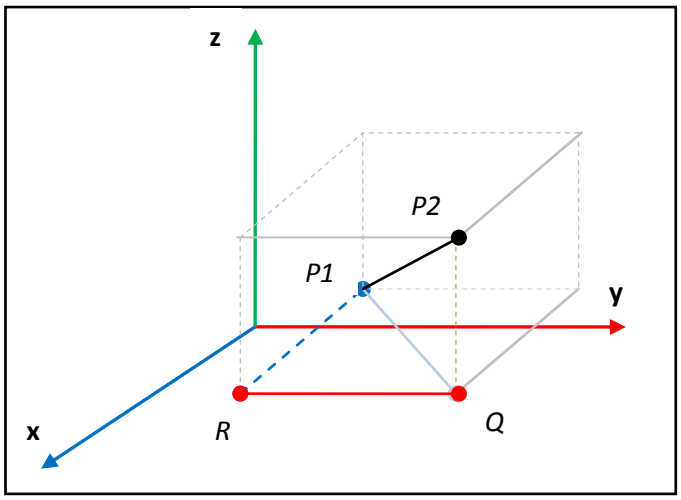


P(x,y,z) merupakan sebuah titik yang terletak dalam ruang (*space*), dimana $\vec{r}$ adalah jarak dari titik asal (*origin*) ke titik P(x,y,z). Untuk menghitung jarak ($\vec{r}$), digunakan rumus :

$$|\vec{r}| = \sqrt{x^2 + y^2 + z^2}$$

Atau

Jika diketahui $\overrightarrow{P1}$ (x_1, y_1, z_1) dan $\overrightarrow{P2}(x_2, y_2, z_2)$:



$$|P1P2| = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2 - (z_2 - z_1)^2}$$

3.1.6.4 Perkalian Titik (*Dot Product*)

Dot product adalah bentuk perkalian antara 2 vektor yang akan menghasil-kan skalar. Dapat didefinisikan dalam rumus :

$$\bar{a} \bullet \bar{b} = |\bar{a}|.|\bar{b}|.\cos\theta$$

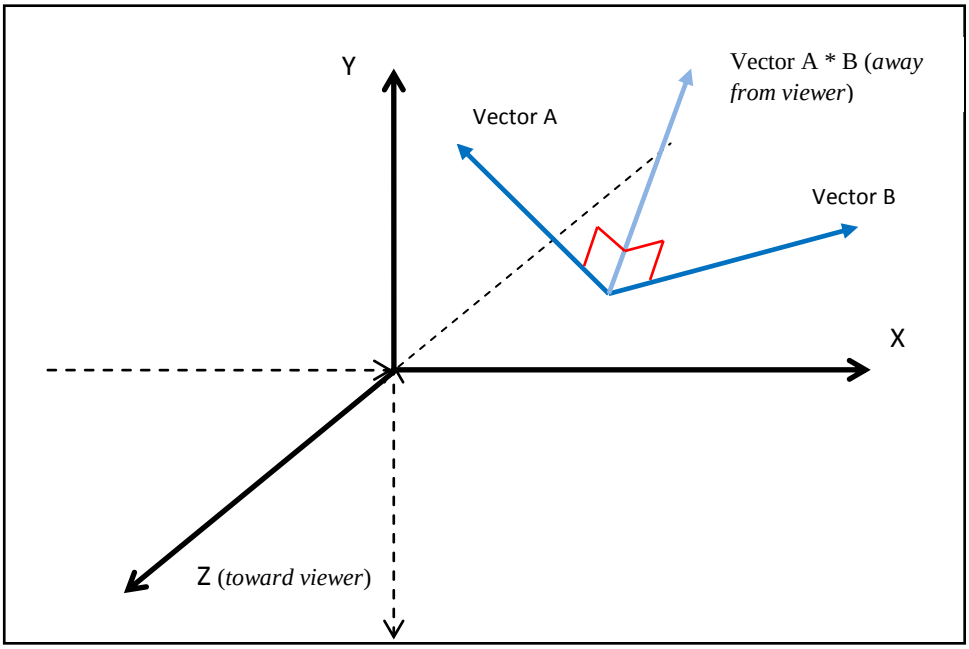
dimana θ adalah sudut yang dibentuk oleh kedua vektor $\bar{a}$ dan $\bar{b}$.

Dalam bentuk komponen vektor ruang (3 dimensi), bila diketahui $\bar{a}$ = $[a_1, a_2, a_3]$ dan $\bar{b}$ = $[b_1, b_2, b_3]$. Dapat didefinisikan dalam rumus :

$$\bar{a} \bullet \bar{b} = a_1 b_1 + a_2 b_2 + a_3 b_3$$

3.1.6.5 Perkalian Silang (*Cross Product*)

Perkalian silang dari dua buah vektor akan menghasilkan sebuah vektor baru, sehingga perkalian silang dua buah vektor juga disebut dengan perkalian vektor. Hasil perkalian silang vektor A dan vektor B (dibaca A *cross* B) menghasilkan vektor C, dimana vektor yang dihasilkan (vektor C) selalu tegak lurus (*perpendicular*) dengan bidang yang dibentuk oleh vektor A dan vektor B.



Hasil dari vector A * B didefinisikan pada rumus berikut :

$$x = A_y * B_z - B_y * A_z$$

$$y = A_z * B_x - B_z * A_x$$

$$z = A_x * B_y - B_x * A_y$$

Perkalian vektor (*cross product*) hanya berlaku pada vektor ruang (3 dimensi) karena dalam vektor 2 dimensi tidak memungkinkan untuk dihasilkan vektor baru yang tegak lurus (*perpendicular*) terhadap hasil perkalian dua vektor.

3.1.6.6 Matriks dan Transformasi Geometri

Matriks adalah sebuah element array berbentuk persegi dimana dioperasikan sebagai satu objek yang tersusun menurut baris dan kolom. Elemen tersebut biasanya berupa angka-angka tetapi dapat juga berupa persamaan matematika lainnya yang dapat ditambah dan dikalikan.

1. Pengaplikasian dari Matriks (*Application of Matrices*)

   Pengaplikasian matriks secara luas digunakan dalam geometri, fisika, dan aplikasi komputer grafis.

   1) Transformasi menggunakan matriks (*Transforms using matrices*)

      1. Perkalian Matriks dan Vektor (*Multiply Matriks by Vector***)**

         Sebuah matriks dapat mengubah sebuah vektor menjadi vektor lain dengan mengakalikannya dengan matriks, seperti ekspresi berikut :

$$\begin{pmatrix} v_{out_1} \\ v_{out_2} \\ v_{out_3} \end{pmatrix} = \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix} \begin{pmatrix} v_{in_1} \\ v_{in_2} \\ v_{in_3} \end{pmatrix}$$

Persamaan diatas mengekspresikan sebagai sebuah transformasi $\begin{pmatrix} v_{in_1} \\ v_{in_2} \\ v_{in_3} \end{pmatrix}$ menjadi $\begin{pmatrix} v_{out_1} \\ v_{out_2} \\ v_{out_3} \end{pmatrix}$. Jika titik tersebut diaplikasikan atau digunakan pada titik dalam bidang ruang (3 dimensi), maka matriks dapat digunakan sebagai transformasi vektor.

2. Penerapan Transformasi (*Applying The Transform*)

Hasil dari perkalian antara matriks dan vektor dapat diibaratkan vektor sebagai n x 1 Matriks, misalnya pengalian sebuah matriks berordo 3x3 dengan vektor $\overline{v_{in}} \begin{pmatrix} v_{in_1} \\ v_{in_2} \\ v_{in_3} \end{pmatrix}$ untuk mengubah vektor v_{in} menjadi v_{out} , diperoleh :

$$\begin{pmatrix} v_{out_1} \\ v_{out_2} \\ v_{out_3} \end{pmatrix} = \begin{pmatrix} a_{11} * v_{in_1} + a_{12} * v_{in_2} + a_{13} * v_{in_3} \\ a_{21} * v_{in_1} + a_{22} * v_{in_2} + a_{23} * v_{in_3} \\ a_{31} * v_{in_1} + a_{32} * v_{in_2} + a_{33} * v_{in_3} \end{pmatrix}$$

3. Skala (*Scalling*)

Penskalaan koordinat dimaksudkan untuk menggandakan setiap komponen yang ada pada objek secara scalar. Penskalaan dapat dilakukan dengan menggunakan matriks.

$$\begin{pmatrix} m_{11} & 0 & 0 \\ 0 & m_{22} & 0 \\ 0 & 0 & m_{33} \end{pmatrix}$$

Dimana $m_{11} = m_{22} = m_{33}$. Atau disebut sebagai matriks identity (matriks diagonal yang elemen-elemen diagonal utamanya semua bernilai 1).

4. Translasi (*Translation*)

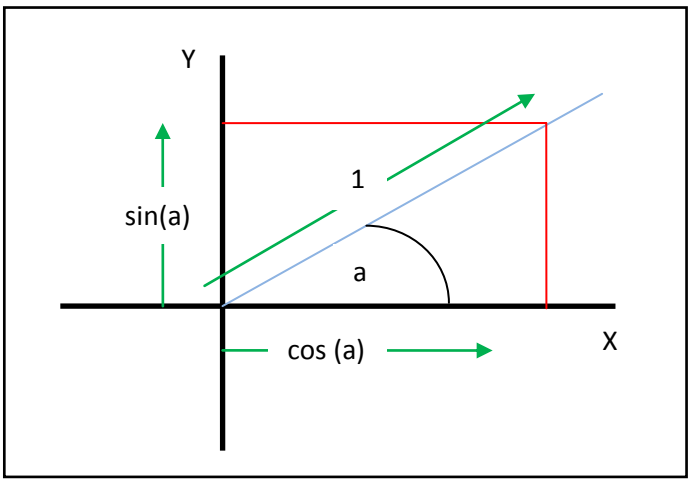
Translasi adalah suatu pergerakan atau perpindahan semua titik dari objek pada suatu jalur lurus (direpresentasikan oleh vektor) sehingga menempati posisi baru.

5. Rotasi (*Rotation*)

   Rotasi adalah mereposisi semua titik dari objek sepanjang jalur lingkaran dengan pusat pada pivot.

2) Matriks Rotasi

Rotasi dapat direpresentasikan dengan matriks *orthogonal* (matriks persegi yang merepresentasikan translasi vektor garis lurus).



Rotasi pada sumbu Z, diamsumsikan sebuah rotasi pada sudut a berlawanan arah jarum jam, dapat direpresentasikan dengan vektor pada arah Z positif.

Jika diketahui titik (x=1, y=0), maka titik rotasinya (x=cos(a), y=sin(a)).

Jika diketahui titik (x=0, y=1), maka titik rotasinya (x=-sin(a), y=cos(a)).

3) Rotasi 3 dimensi (3D Rotations)

1. Rotasi pada sumbu Z

Direpresentasi dengan matriks :

$$R_z = \begin{pmatrix} \cos(a) & -\sin(a) & 0 \\ \sin(a) & \cos(a) & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Contoh, jika sudut rotasi = +90, diperoleh :

$$R_z = \begin{pmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Maka arah rotasi dapat di deskripsikan dengan sistem tangan kanan (right-hand system) , dimana ibu jari adalah arah +Z (mendekati pengamat) dan jari-jari lain menunjukkan arah rotasi positif. Jadi jika digambarkan , gambar A =  di rotasi menjadi gambar B = 

2. Rotasi pada sumbu Y

Direpresentasikan dengan matriks :

$$R_y = \begin{pmatrix} \cos(a) & 0 & \sin(a) \\ 0 & 1 & 0 \\ -\sin(a) & 0 & \cos(a) \end{pmatrix}$$

Contoh, jika sudut rotasi = +90, maka :

$$R_y = \begin{pmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ -1 & 0 & 0 \end{pmatrix}$$

Maka arah rotasi dapat di deskripsikan dengan sistem tangan kanan , dimana ibu jari adalah arah +Y (keatas) dan jari-jari lainnya menunjukkan arah rotasi positif. Jadi jika digambarkan, gambar A = , di rotasi menjadi gambar B = .

3. Rotasi Pada Sumbu X

Di representasikan dengan matriks :

$$R_x = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos(a) & -\sin(a) \\ 0 & \sin(a) & \cos(a) \end{pmatrix}$$

Contoh, jika sudut rotasi = +90, diperoleh :

$$R_x = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{pmatrix}$$

Maka arah rotasi dapat di deskripsikan dengan sistem tangan kanan , dimana ibu jari adalah arah +X (ke kanan) dan jari-jari lainnya menunjukkan arah rotasi positif. Jadi, jika

digambarkan, gambar A = , di rotasi menjadi gambar B = .

4) Hubungan Matriks dengan Disiplin Matematika lainnya

Matriks dapat digunakan dalam berbagai bentuk disiplin matematika, seperti :

1) Sebagai sebuah vektor dengan baris dan kolom.
2) Sebagai sebuah tensor (generalisasi dari scalar dan vektor).
3) Merepresentasikan sekumpulan persamaan garis lurus (*linear equations*).

Matriks dalam vektor ruang, yang memungkinkan digunakan operasi sebagai berikut :

Tabel III.1 Operasi Matriks

Operasi	Notasi	Deskripsi
Penjumlahan (*Addition*)	M[a+b] = M[a] + M[b]	Penjumlahan 2 matriks dilakukan dengan menjumlahkan sesuai elemen dari 2 matriks tersebut.
Perkalian Skalar (*Scalar multiplication*)	M[s*a] = s * M[a]	*Scalar product* dari matriks dapat dilakukan dengan mengalikan *scalar product* dengan setiap elemen pengalinya.

Axioms

Tabel III.2 *Axioms*

Axiom	***Addition***	***Scalar Multiplication***	***Multiplication***
Associativity	(a+b)+c = a+(b+c)	(s1 s2) a = s1 (s2 a)	(a * b) * c = a * (b * c)
Commutativity	a + b = b + a		Tidak selalu komutatif
Distributivity		s*(b +c) = s*b + s*c (s1 + s2)*a = s1*a + s2*a	a*(b+c) = a*b + a*c (a+b)*c = a*c + b*c
Identity	a + 0 = a 0 + a = a	1 * a = a	a * 1 = a 1 * a = a
Inverses	a + (-a) = 0 (-a) + a = 0		a * a^{-1} = 1 $a^{-1} * a = 1$, jika $\det[a] \neq 0$

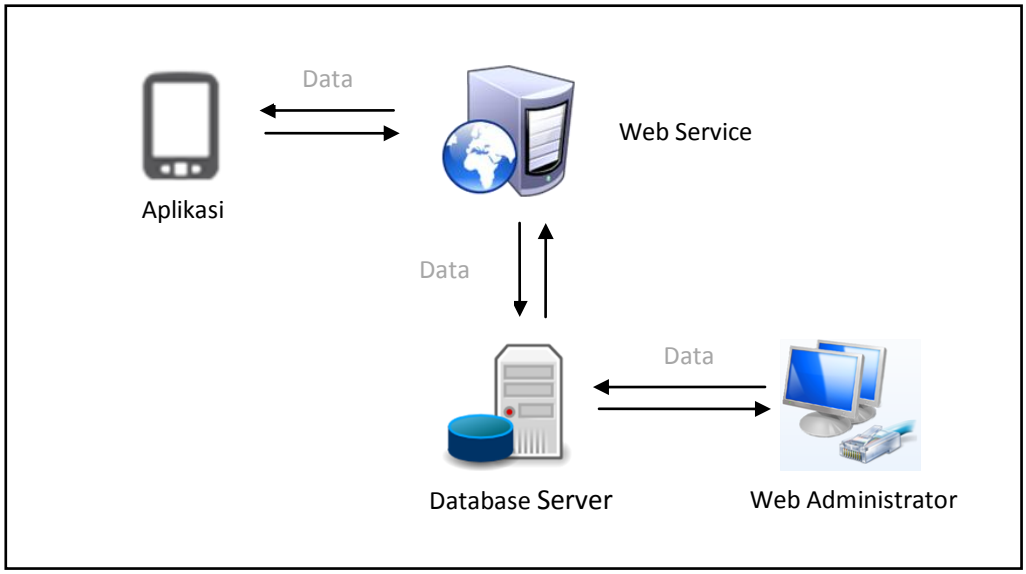
Tabel III.2 mendeskripsikan properti dari operasi matriks (perkalian matriks bukanlah komutatif).

5) Hubungan Matriks dan Vektor

Vektor memiliki kaitan yang erat dengan matriks, keduanya dapat dianggap sebagai satu matriks terarah. Matriks tidak dapat dibentuk dari sebuah vektor karena elemen dari vektor harus memenuhi sktruktur matematika yang dikenal sebagai *field* (Aljabar : sebuah struktur aljabar dengan dua operasi, penjumlahan dan perkalian; Geometri dan Fisika: dalam geometri, sebuah vektor atau bidang scalar dan dalam fisika, sebuah vektor atau *scalar quantity*, dimana nilainya adalah fungsi posisi dalam ruang (*space*)) dan sebuah vektor bukanlah *field* karena vektor tidak memiliki perkalian komutatif dan properti lainnya yang dibutuhkan dalam *field.*

3.1.7 Analisis Alur Data Sistem

Analisis alur data sistem merupakan analisis yang berfungsi untuk menggambarkan secara rinci bagaimana sistem bekerja.



Gambar III.7 Alur Data Sistem

Analisis ini meliputi :

1 Aplikasi *mobile*

Aplikasi *mobile* dalam hal ini adalah analisis fungsionalitas aplikasi untuk mengakses data dari database server melalui web service. Aplikasi ini dibangun pada *platform* android dan komunikasi data antar aplikasi dengan database server menggunakan *parsing* data dari web service yang bertipe JSON.

1. Web *service*

Web service pada penelitian ini berfungsi sebagai *middle application* atau perantara aplikasi dengan database server. Cara kerja web service ini yaitu dengan mengambil data dari database

server dan kemudian mengkonversi data tersebut kedalam format pertukaran data JSON.

2. *Protocol* HTTP

   Protocol HTTP merupakan *protocol* jaringan lapisan aplikasi yang digunakan untuk sistem informasi terdistribusi, kolaboratif, dan menggunakan hypermedia. Pada penelitian ini HTTP digunakan aplikasi *mobile* dan web administrator sebagai protocol yang dapat men-distribusikan data yang bersumber dari database server.

3. AR Browser (*augmented reality browser*)

   AR browser merupakan sebuah aplikasi *user agent* (aplikasi yang berada pada *client* implementasi dari jaringan *protocol* dimana digunakan dalam komunikasi sistem *client-server*) yang berfungsi untuk mengumpulkan, menyajikan dan melintasikan sumber informasi melalui antarmuka *augmented reality*.

3.1.8 Analisis Pertukaran Data *Backend* dan *Frontend*

Agar informasi fasilitas umum yang tersimpan dalam database server dapat tampil dalam aplikasi, maka digunakan JSON, format penulisan data untuk menampilan atau mentransfer data yang tersimpan dalam database server ke aplikasi yang berada pada *client* yaitu pengguna aplikasi. Terdapat dua bagian utama data yang dipertukarkan yaitu data kategori dan data konten.

Berikut adalah contoh format data yang dipertukarkan :

1) Format data kategori

```
{
    "result": [
        {
            "id": "1",
            "kategori": "Rumah sakit",
            "image": "hospital.png"
        }
]
}
```

Format data kategori adalah data yang pertama kali di unduh oleh aplikasi ketika pengguna mengaktifkan aplikasi *ARPublicFacilities*. Data ini menjadi tampilan utama aplikasi atau menu utama dimana terdapat beberapa pilihan kategori fasilitas umum yang dapat dipilih oleh pengguna.

2) Format data konten

```
{
    "status": "OK",
    "num_results": 1,
    "results": [
        {
            "id": "84",
            "lat": "-6.88497258",
            "lng": "107.61355564",
            "elevation": "0",
            "title": "McDonalds Simpang Dago",
            "address": "Jl. Ir. H. Djuanda No.181",
            "phone": "14045",
            "tipe": "Rumah Makan Cepat saji",
            "description":  "Buka  24  jam  dan  melayani
delivery order",
            "has_detail_page": "1",
            "image": "mcdonald.png",
            "webpage": "http://www.mcdelivery24hrs.com/",
            "distance": "0.33"
        }
 ]
}
```

Format data konten adalah data yang berisi keterangan mengenai suatu fasilitas umum berdasarkan kategori yang telah dipilih sebelumnya. Data inti (data untuk *augmented marker*) dalam aplikasi ini adalah data geografis, meliputi *latitude*, *longitude*, dan *elevation* (*elevation* adalah data geografis yang memuat data ketinggian (posisi vertikal) suatu objek dari suatu titik tertentu), data elevation diberi nilai *default* 0 karena informasi mengenai data ini masih sulit untuk ditentukan.

3.1.9 Analisis Kebutuhan Non Fungsional

Analisis kebutuhan non fungsional menggambarkan kebutuhan sistem yang menitik-beratkan pada properti perilaku yang dimiliki oleh sistem, diantaranya kebutuhan perangkat keras, perangkat lunak, serta *user* sebagai bahan analisis kekurangan dan kebutuhan yang harus dipenuhi dalam perancangan sistem yang akan diterapkan.

3.1.9.1 Spesifikasi Perangkat Lunak

Perangkat lunak digunakan dalam sebuah sistem merupakan kumpulan perintah-perintah yang diberikan kepada perangkat keras agar saling berinteraksi untuk melakukan suatu tugas. Perangkat lunak yang dibutuhkan untuk membangun aplikasi ARPublicFacilities adalah sebagai berikut :

1) Sistem operasi Windows Vista Home Basic Service Pack 1
2) JDK (*Java Development Kit*) 1.6 dan SDK (*Standard Developmenet Kit*) untuk Windows
3) ADT (*Android Development Tools*) 0.9.9

4) Eclipse Helios 3.5

5) Mixare AR Browser (*Augmented Reality Browser*)

3.1.9.2 Spesifikasi Perangkat Keras

Komputer dan *smartphone* terdiri dari perangkat keras dan perangkat lunak yang saling berinteraksi. Perangkat lunak memberikan instruksi-instruksi kepada perangkat keras untuk melakukan tugas tertentu. Pembangunan aplikasi ARPublicFacilities, perangkat keras yang digunakan adalah sebagai berikut :

1) *Processor* Intel Core 2 Duo
2) RAM 2GB
3) *Harddisk* 250GB
4) *Mouse* dan *Keyboard*
5) HTC Desire (OS android 2.2)

3.1.9.3 Spesifikasi Pengguna Sistem (*user*)

Spesifikasi pengguna sistem dimaksudkan untuk mengetahui siapa saja *actor* yang terlibat dalam menjalankan sistem. Sistem informasi aplikasi ARPublicFacilities menggunakan platform teknologi *mobile* dan web.

Pengguna sistem dibagi menjadi dua bagian, yaitu :

1 Pengguna aplikasi fasilitas umum (*end user*) yaitu para pengguna *smartphone* dengan *platform* android versi 2.2 atau lebih tinggi. Dalam menggunakan sistem ini, pengguna diharuskan terhubung dengan koneksi jaringan GPRS/EDGE/UMTS, dan sensor dalam *smartphone* aktif

seperti GPS *receiver*, *digital compass, accelerometer*, dan kamera. Untuk bisa menggunakan aplikasi ini.

2 Admin sebagai pengelola konten mempunyai kapabilitas dalam mengelola database seperti menambah, mengubah, dan menghapus data melalui web administrator.

Berikut ini karakteristik pengguna sistem yang ditunjukkan oleh tabel berikut :

Tabel III.3 Tabel Karakteristik Pengguna

Pengguna	Usia	Hak Akses	Tingkat pendidikan	Pengalaman
End User	> 17 tahun	Mengakses data informasi fasilitas umum melalui aplikasi front-end	Min SMA, atau sederajat	Min. Menjalankan aplikasi pada perangkat android
Admin	> 20 tahun	Mengakses, mengelola aplikasi back-end	Min.D3	Min. Mengatur dan memperbaiki data fasilitas umum pada web admin

3.1.10 Analisis Perhitungan Jarak Dengan *Great-circle Distance* Menggunakan Haversine Formula

Berikut ini salah satu rumus yang sering digunakan untuk menentukan jarak dua titik koordinat (*latitude* dan *longitude*) dengan mengabaikan bentuk geografis baik bukit maupun lembah.

```
Jarak = ( ACOS ( Sin(latitude1 * pi/180 ) *
          Sin(latitude2 * pi/180) +
```

```
        Cos(latitude1 *pi/180 ) * Cos(latitude2 *
        pi/180 ) * Cos(abs((longitude2 * pi/180) -
        (longitude1 *pi/180)))) / PI * 180) *
        1.852 * 60.0
```

Dimana,

Latitude1 = Nilai *Latitude* dari lokasi awal (satuan radian).

Longitude1 = Nilai *longitude* dari lokasi awal (satuan radian).

Latitude2 = Nilai *Latitude* dari lokasi tujuan (satuan radian).

Longitude2 = Nilai *longitude* dari lokasi tujuan (satuan radian)..

Contoh :

Misalnya untuk mencari jarak antar Bandara Husein Sastranegara Bandung ke Bandara Soekarna Hatta, Banten. Jika diketahui :

1 Bandara Husein Sastranegara :

Latitude 1 : - $6^0 54' 8.98''$

Longitude 1 : $107^0 34' 48.81''$

2 Bandara Soekarno Hatta :

Latitude 2 : - $6^0 7' 23.94''$

Longitude 2 : - $106^0 39' 41.85''$

Koordinat diatas masih dalam bentuk *degree*, *minute,* dan *second*. Dan akan diubah ke dalam bentuk *decimal degree* :

Misalnya, *Latitude* 1 : - $6^0 54' 8.98''$

1. Menghitung total dari nilai *seconds* : $54' 8.98''$

   ((54 * 60) + 8.98) = 3248,98

2. Hasil perhitungan pertama dibagi 3600

(3248.98 / 3600) = 0.902494

3.Menjumlahkan nilai *degree* dengan hasil pembagian, dengan mengabaikan nilai negatif

- (6 + 0.902494) = -6.902494

Dari hasil perhitungan untuk keempat koordinat dengan menggunakan cara diatas, diperoleh koordinat seperti berikut :

1 Bandara Husein Sastranegara

Latitude 1 : -6.902494.

Longitude 1 : 107.579670

2 Bandara Soekarno Hatta

Latitude 2 : -6.123316

Longitude 2 : 106.661625

Selanjutnya, perhitungan mencari jarak :

```
Jarak = ( ACOS ( Sin(latitude1 * pi/180 ) *
         Sin(latitude2 * pi/180) +
         Cos(latitude1 *pi/180 ) *
         Cos(latitude2 * pi/180 ) *
         Cos(abs((longitude2 * pi/180) -
         (longitude1 *pi/180)))) / PI * 180) *
         1.852 * 60.0

Hasil = 133.300 KM
```

3.1.11 Analisis Kebutuhan Fungsional

Analisis kebutuhan fungsional menggambarkan proses kegiatan yang akan diterapkan dalam sebuah sistem dan menjelaskan kebutuhan yang diperlukan sistem agar sistem dapat berjalan dengan baik serta sesuai dengan kebutuhan.

Analisis kebutuhan fungsional ini meliputi analisis kebutuhan sistem, analisis kebutuhan data, spesifikasi sistem dan pemodelan sistem.

Pemodelan sistem dimodelkan dengan menggunakan UML (*Unified Modeling Language*). Tahap-tahap pemodelan dalam analisis tersebut antara lain *Use Case Diagram, Class Diagram, State Diagram, Activity Diagram, Sequence Diagram, Collaboration Diagram, Component Diagram,* dan *Deployment Diagram.*

3.1.9.1 Analisis Kebutuhan Sistem

Dalam perancangan sistem pada aplikasi yang dibangun, sebelumnya dilakukan analisis kebutuhan sistem agar sistem yang dibangun sesuai dengan yang diharapkan atau dibutuhkan pengguna. Adapun kebutuhan sistem yang diperlukan antara lain :

1 Menampilkan informasi mengenai lokasi keberadaan fasilitas umum, sesuai atau relevan terhadap posisi pengguna.
2 Informasi fasilitas umum yang diinformasikan jelas.
3 Memberikan informasi visual mengenai letak fasilitas umum terhadap posisi dan orientasi pandangan pengguna.
4 Informasi mengenai lokasi fasilitas umum segera ditampilkan tanpa pengguna melakukan pencarian terlebih dahulu.

3.1.9.2 Analisis Kebutuhan Data

Analisis kebutuhan data bertujuan untuk menganalisis kebutuhan data yang diperlukan dan memudahkan perancangan informasi setelah sistem yang dibutuhkan telah diketahui. Adapun analisis kebutuhan data dengan kriteria sebagai berikut :

1 Data ATM (Anjungan Tunai Mandiri)

Menampilkan data ATM meliputi nama bank penyedia ATM, alamat, dan jarak.

2 Data Toko

Menampilkan data Toko meliputi nama Toko, alamat, jarak, dan keterangan.

3 Data Kuliner

Menampilkan data Kuliner meliputi nama penyedia Kuliner, alamat, jarak, dan keterangan.

4 Data Rumah Sakit

Menampilkan data Rumah sakit meliputi nama Rumah Sakit, alamat, jarak, dan keterangan.

5 Data Stasiun Pengisian Bahan Bakar Umum (SPBU)

Menampilkan data SPBU meliputi no registrasi SPBU, alamat, jarak dan keterangan.

6 Data *Factory Outlet*

Menampilkan data *Factory Outlet* meliputi nama *Factory Outlet*, alamat, jarak, dan keterangan.

3.1.9.3 Spesifikasi Sistem

Adapun spesifikasi sistem yang dibutuhkan sebagai berikut :

1 Pengembangan sistem menggunakan model arsitektur *two tier* (2-tier). Membagi proses menjadi dua bagian, yaitu *frontend*, aplikasi yang berjalan

dipihak *client* dan *backend*, berupa *database server* sebagai sumber data, dikembangkan menggunakan CMS (*Content Management System*).

2 Komunikasi antara aplikasi *client* dan *server* melalui koneksi protokol HTTP dengan memanfaatkan jaringan seluler.

3 Sistem yang berjalan pada *client* (*frontend*) sifatnya hanya memberikan informasi mengenai fasilitas umum tanpa keterlibatan pengguna dalam prosesnya (tidak mengijinkan input manual seperti pencarian).

4 Spesifikasi sistem pada sisi *client* (*frontend*), yaitu :

   a. Lokasi saat mengaktifkan atau menjalankan aplikasi *frontend* dianjurkan dalam lingkungan terbuka atau tidak sedang berada dalam bangunan tertutup agar informasi yang ditampilkan akurat.
   b. Perangkat yang digunakan memiliki kemampuan seperti GPS *receiver, accelerometer, digital compass*, dan kamera.
   c. Pengguna berada dalam wilayah cakupan jaringan seluler.

5 Spesifikasi sistem pada sisi *server* (*backend*), yaitu :

   a. Memfasilitasi kebutuhan data pada aplikasi *frontend*.
   b. Dapat dilakukan pengelolaan data seperti tambah, ubah, dan hapus data.

3.1.9.4 Pemodelan Sistem

Pemodelan sistem berfungsi untuk menentukan fungsi-fungsi yang dapat dilakukan oleh sistem pada aplikasi serta menentukan kelas yang dibutuhkan

untuk realisasi fungsi-fungsi sistem yang telah dianalisis sebelumnya dan mendeskripsikannya kedalam bentuk diagram.

1 *Use Case Diagram*

Digunakan untuk memodelkan atau menggambarkan batasan sistem dan fungsi-fungsi utamanya. Mendiskripsikan fungsi dari sebuah sistem dari perspektif pengguna, use case bekerja dengan cara mendeskripsikan tipikal interaksi antara pengguna sebuah sistem dengan sistemnya sendiri melalui sebuah cerita bagaimana sebauh sistem dipakai. *Use case* terdiri dari tiga bagian yaitu identifikasi *actor*, identifikasi *use case*, dan *scenario use case*.

1) Identifikasi *Actor*

Aktor adalah abstraksi dari orang dan sistem yang lain yang mengaktifkan fungsi dari target sistem. Berikut adalah aktor-aktor yang berperan dalam menjalan sistem yang dibangun.

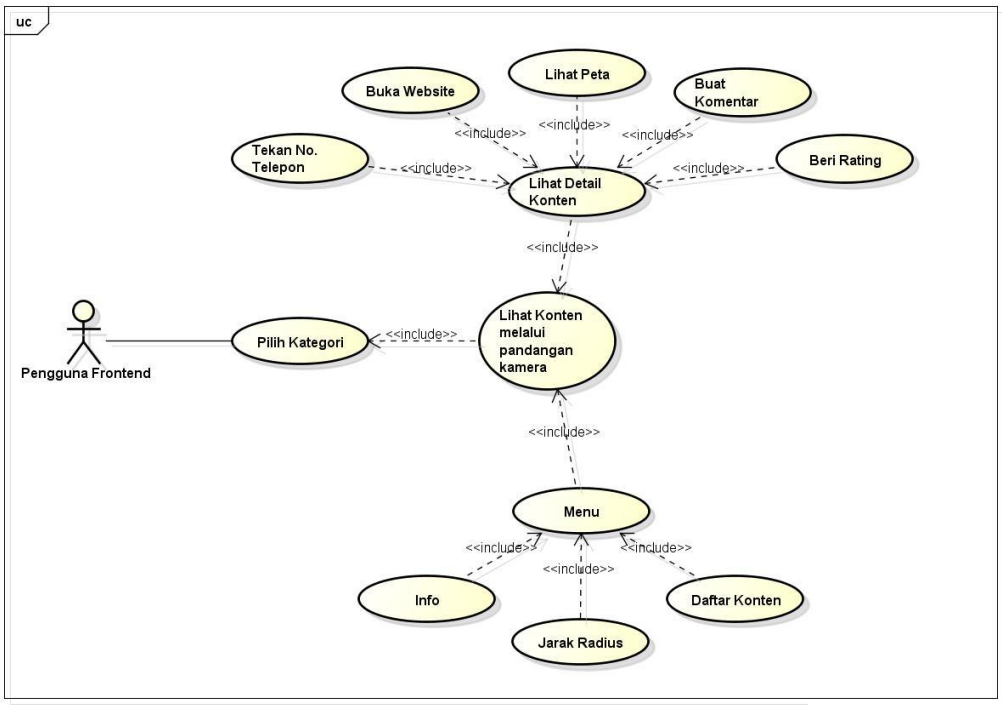
Tabel III.4 Tabel *Actor Use Case*

No	Aktor	Deskripsi
A-01	Pengguna *Frontend*	Merupakan *actor* atau pengguna akhir dari aplikasi yang dibangun, aplikasi *frontend* (*ARPublicFacilities*)
A-02	Pengguna *Backend*	Merupakan *actor* yang berperan mengelola data informasi dalam *database server* melalui CMS berbasis web.

Pada gambar III.8 dan gambar III.9, penggambaran *use case diagram* sistem dibagi menjadi 2 bagian yaitu *use case diagram frontend* dan *use case diagram backend*. Kedua *use case diagram* menunjukkan bagaimana interaksi antara aktor dan sistem, dimana terdapat dua aktor yang berperan dan sistem yang terdiri dari aplikasi *frontend* dan

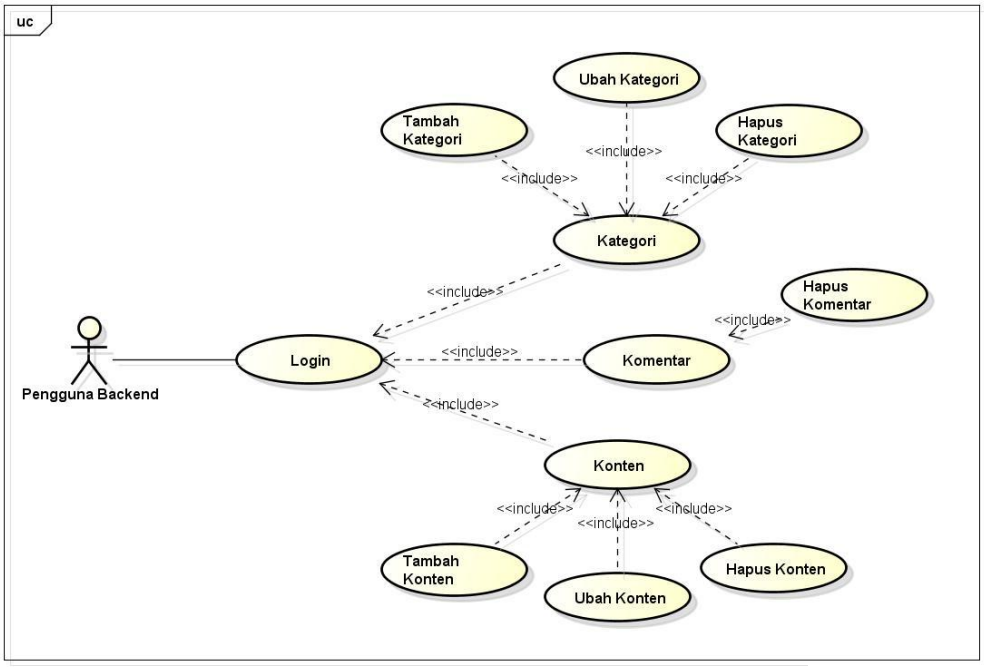
backend. Aktor pengguna *fronted* yaitu yang menjalankan aplikasi *frontend*. Sedangkan aktor pengguna *backend* berperan mengelola aplikasi *backend* pada CMS berbasis web.

1.1 *Use Case Diagram* Aplikasi *Frontend*



Gambar III.8 *Use Case Diagram* Aplikasi *Frontend*

1.2 *Use Case Diagram* Aplikasi *Backend*



Gambar III.9 *Use Case Diagram* Aplikasi *Backend*

2) Identifikasi *Use Case Diagram*

Berikut tabel identifikasi use case yang terdapat dalam sistem :

Tabel III.5 Identifikasi *Use Case Diagram*

Use Case* Aplikasi *Frontend		
No	***Use Case***	**Deskripsi**
UCF-01	Pilih Kategori	Fungsionalitas memungkinkan pengguna memilih kategori fasilitas umum, terdiri dari ATM, Kuliner, Toko, Rumah Sakit, SPBU, *factory outlet*
UCF-02	Lihat Konten melalui Pandangan Kamera	Fungsionalitas untuk menampilan konten berdasarkan kategori aktif (yang dipilih) dengan menggunakan kamera ponsel untuk melihat keberadaan konten.
UCF-03	Lihat Detail Konten	Fungsionalitas untuk menampilkan detail konten
UCF-04	Lihat Peta	Fungsionalitas untuk menampilkan peta rute navigasi konten.
UCF-05	Buka Website	Fungsionalitas untuk menampilkan halaman web konten.
UCF-06	Tekan No.Telepon	Fungsionalitas untuk menampilkan *phone dialer*.

UCF-07	Menu	Fungsionalitas untuk menampilkan menu aplikasi.
UCF-08	Jarak Radius (*Zoom Level*)	Fungsionalitas untuk mengatur jarak konten yang ingin ditampilkan.
UCF-09	Daftar Konten (*List View*)	Fungsionalitas untuk menampilkan daftar konten berdasarkan kategori
UCF-10	Info	Fungsionalitas untuk menampilkan informasi geografis keberadaan pengguna.
UCF-11	Buat Komentar	Fungsionalitas yang memungkinkan pengguna dapat memberi komentar mengenai suatu konten.
UCF-12	Beri *Rating*	Fungsionalitas yang memungkinkan pengguna dapat memberikan peringkat (rating) pada suatu konten.
Use Case* Aplikasi *Backend		
No	***Use Case***	**Deskripsi**
UCB-01	*Login*	Fungsionalitas untuk proses autentifikasi hak akses kepada admin
UCB-02	Konten	Fungsionalitas yang memungkinkan admin dapat mengelola data konten.
UCB-03	Tambah Konten	Fungsionalitas untuk menambahkan data konten.
UCB-04	Ubah Konten	Fungsionalitas untuk mengubah atau *update* data konten.
UCB-05	Hapus Konten	Fungsionalitas untuk menghapus data konten.
UCB-06	Kategori	Fungsionalitas yang memungkinkan admin dapat mengelola data kategori.
UCB-07	Tambah Kategori	Fungsionalitas untuk menambahkan data kategori.
UCB-08	Ubah Kategori	Fungsionalitas untuk mengubah atau *update* data kategori.
UCB-09	Hapus Kategori	Fungsionalitas untuk menghapus data kategori.
UCB-10	Komentar	Fungsionalitas untuk menampilkan komentar konten.
UCB-11	Hapus Komentar	Fungsionalitas untuk menghapus data Komentar.

3) Skenario *Use Case*

3.1 Skenario *Use Case* Aplikasi *Frontend*

Skenario proses-proses yang terdapat dalam *use case diagram* aplikasi *frontend* dapat dilihat pada tabel-tabel berikut :

Tabel III.6 Skenario *Use Case* Pilih Kategori

Identifikasi	
Nama Use Case	Pilih Kategori
Aktor	Pengguna *Frontend*
Tujuan	Memilih jenis (kategori) fasilitas umum.
Keadaan awal	Sistem menampilkan menu *kategori*
Skenario Utama	
Aksi Aktor	**Reaksi Sistem**
	1. Menampilkan jenis-jenis kategori (ATM, Kuliner, Toko, SPBU, Rumah Sakit, *Factory outlet*)
2. Memilih jenis fasilitas umum yang ditampilkan sistem.	
	3. Jika memilih satu dari beberapa jenis informasi yang diinformasikan.Maka informasi yang ditampilkan adalah berdasarkan jenis informasi yang dipilih. Misalnya ATM, maka informasi yang ditampilkan hanya informasi ATM saja.
Skenario Alternatif – Validasi Gagal	
Aksi Aktor	**Reaksi Sistem**
	1.” *Sorry!, ARPublicFacilities has stopped unexpectedly. Please try again*.“
2. Menekan tombol “*Force Close*”	
	3. Sistem akan terminate.
Kondisi Akhir	Menampilkan informasi kategori sesuai yang dipilih.

Tabel III.7 Skenario *Use Case* Lihat Konten melalui Pandangan Kamera

Identifikasi	
Nama *Use Case*	Lihat Konten melalui Pandangan Kamera
Aktor	Pengguna *Frontend*
Tujuan	Melihat konten (fasilitas umum) berdasarkan kategori
Keadaan Awal	Sistem menampilkan menu

	kategori
Skenario Utama	
Aksi Aktor	**Reaksi Sistem**
1. Menemukan konten dengan mengubah orientasi ponsel terhadap posisi pengguna untuk melihat letak konten yang diinformasikan dengan melalui tampilan kamera ponsel.	
	2. Menampilkan letak konten berdasarkan jenis kategori yang dipilih.
Skenario Gagal	
Aksi Aktor	**Reaksi Sistem**
	1." *Sorry, a connection Error occured!. Please check if you are connected to the Internet. "*
2. Menghentikan aplikasi atau keluar dari aplikasi.	
	3. Sistem akan terminate
Keadaan Akhir	Menampilkan tampilan *augmented* konten berdasarkan kategori yang dipilih

Tabel III.8 Skenario *Use Case* Lihat Detail Konten

Identifikasi	
Nama Use Case	Lihat Detail Konten
Aktor	Pengguna *Frontend*
Tujuan	Melihat informasi detail konten yang dipilih.
Keadaan awal	Sistem menampilkan *augmented* konten.
Skenario Utama	
Aksi Aktor	**Reaksi Sistem**
1. Memilih salah satu *augmented marker konten.*	
	2. Menampilkan window baru yang berisi informasi detail dari konten yang dipilih.
Skenario Gagal	
Aksi Aktor	**Reaksi Sistem**
	1. "*There is no location information available within this radius. Please try to search in a wider range*".
2. Mengubah range jarak konten	

pada *zoom level control* yang terdapat dalam fitur menu.	
Keadaan Akhir	Aplikasi menampilkan informasi detail dari konten yang dipilih sebelumnya.

Tabel III.9 Skenario *Use Case* Lihat Peta

Identifikasi	
Nama *Use Case*	Lihat Peta
Aktor	Pengguna *Frontend*
Tujuan	Menampilkan peta rute navigasi berdasarkan konten yang telah dipilih sebelumnya.
Scenario Utama	
Aksi Aktor	**Reaksi Sistem**
1. Memilih tombol Peta pada tampilan detail konten.	
	2. Mengunduh google maps untuk menampilkan peta rute navigasi.
Keadaan Akhir	Aplikasi menampilkan peta rute navigasi berdasarkan konten yang dipilih sebelumnya.

Tabel III.10 Skenario *Use Case* Buka *Website*

Identifikasi	
Nama Use Case	Buka *Website*
Aktor	Pengguna *Frontend*
Tujuan	Menampilkan halaman web konten
Skenario Utama	
Aksi Aktor	**Reaksi Sistem**
1. Memilih tombol Website pada tampilan detail konten	
	2. Membuka halaman browser dan mengakses alamat URL website konten.
Skenario Gagal	
Aksi Aktor	**Reaksi Sistem**
	1. Menampilkan pesan “Maaf, website konten ini belum ditemukan”
2. Menekan tombol OK	
Keadaan Akhir	Sistem menampilkan halaman web konten

Tabel III.11 Skenario *Use Case* Tekan No.Telepon

Identifikasi	
Nama Use Case	Tekan No.Telepon
Aktor	Pengguna *Frontend*
Tujuan	Menampilkan phone dialer
Skenario Utama	

Aksi Aktor	Reaksi Sistem
1. Memilih tombol *Phone* pada tampilan detail konten	
	2. Menampilkan *phone dialer* beserta no telepon konten
Skenario Gagal	
Aksi Aktor	**Reaksi Sistem**
	1. Menampilkan pesan "Maaf, nomor telepon ini belum ditemukan"
2. Menekan tombol OK	
Keadaan Akhir	Sistem menampilkan *dialer phone* beserta no telepon konten

Tabel III.12 Skenario *Use Case* Menu

Identifikasi	
Nama Use Case	Menu
Aktor	Pengguna *Frontend*
Tujuan	Menampilkan menu aplikasi
Keadaan Awal	Sistem menampilkan "Lihat Konten melalui pandangan kamera"
Skenario Utama	
Aksi Aktor	**Reaksi Sistem**
1. Menekan tombol "Menu".	
	2. Menampilkan optional menu yang dapat dipilih untuk ditampilkan.
Keadaan Akhir	Sistem menampilkan menu aplikasi.

Tabel III.13 Skenario *Use Case* Daftar Konten

Identifikasi	
Nama Use Case	Daftar Konten (*List View*)
Aktor	Pengguna *Frontend*
Tujuan	Menampilkan seluruh konten berdasarkan kategori
Keadaan Awal	Sistem menampilkan "Lihat Konten melalui pandangan kamera"
Skenario Utama	
Aksi Aktor	**Reaksi Sistem**
1. Menekan tombol "Menu".	
	2. Menampilkan "Menu" pilihan.
3. Memilih tombol "*List View"*	
	4. Menampilkan daftar konten berdasarkan kategori yang dipilih sebelumnya
Skenario Gagal	

Aksi Aktor	Reaksi sistem
	1.” *Sorry!, ARPublicFacilities has stopped unexpectedly. Please try again*. “
2. Menekan tombol “*Force Close*”	
	3. Sistem terminate
Keadaan Akhir	Menampilkan daftar konten

Tabel III.14 Skenario *Use Case* Jarak Radius (*Zoom Level*)

Identifikasi	
Nama Use Case	Jarak Radius
Aktor	Pengguna *Frontend*
Tujuan	Mengatur jarak radius konten yang ingin ditampilkan
Keadaan Awal	Sistem menampilkan “Lihat Konten melalui pandangan kamera.”
Skenario Utama	
Aksi Aktor	**Reaksi Sistem**
1. Menekan tombol “Menu”.	
	2. Menampilkan “*Menu*” pilihan.
3. Memilih tombol “*Zoom Level”*	
	4. Menampilkan kontrol untuk pengaturan jarak konten yang ingin ditampilkan.
Skenario Gagal – Validasi Gagal	
Aksi Aktor	**Reaksi Sistem**
	1.” *Sorry!, ARPublicFacilities has stopped unexpectedly. Please try again*. “
2. Menekan Tombol “*Force Close*”	
	3. Sistem akan terminate.
Keadaan Akhir	Menampilkan konten berdasarkan pengaturan jarak radius.

Tabel III.15 Skenario *Use Case* Info

Identifikasi	
Nama Use Case	Info
Aktor	Pengguna *Frontend*
Tujuan	Menampilkan informasi geografis posisi keberadaan pengguna.
Keadaan awal	Sistem menampilkan “Lihat Konten melalui pandangan kamera”
Skenario Utama	
Aksi Aktor	**Reaksi Sistem**

1. Menekan tombol "*Menu*".	
	2. Menampilkan "*Menu*" pilihan.
3. Memilih tombol "*Info*"	
	4. Menampilkan informasi geografis posisi keberadaan pengguna.
Skenario Gagal	
Aksi Aktor	**Reaksi Sistem**
	1." *Sorry!, ARPublicFacilities has stopped unexpectedly. Please try again*. "
2. Menekan Tombol "*Force Close*"	
	3. Sistem akan terminate.
Keadaan Akhir	Menampilkan informasi geografis pengguna.

Tabel III.16 Skenario Use Case Buat Komentar

Identifikasi	
Nama Use Case	Buat Komentar
Aktor	Pengguna Frontend
Tujuan	Memungkinkan pengguna untuk memberikan Komentar mengenai suatu konten tertentu.
Keadaan awal	Menampilkan halaman detail konten
Skenario Utama	
Aksi Aktor	**Reaksi Sistem**
1. Memilih tombol "Add Review" yang terdapat pada tampilan detail konten	
	2. Menampilkan jendela baru yang berisi isian teks untuk memasukkan Komentar pengguna.
3. Mengisi seluruh isian teks yang terdapat dalam jendela komentar baru	
	4. Mengirim data komentar baru ke server.
Skenario Gagal	
Aksi Aktor	**Reaksi Sistem**
1. Mengisi kolom isian komentar dan menekan tombol "S*hare*"	
	2. Menampilkan pesan "*Sorry, your review cannot set. Please completely fill all the fields. Thanks*" gagal menyimpan komentar baru.
2. Menekan tombol "OK". Mengisi semua kolom isian komentar.	
	3. Komentar berhasil dikirim ke server dan

	kemudian ditampilkan dalam jendela detail konten
Kondisi Akhir	Komentar baru berhasil dikirim ke server.

Tabel III.17 Skenario Use Case Beri Rating

Identifikasi	
Nama Use Case	Beri Rating
Aktor	Pengguna *Frontend*
Tujuan	Memungkinkan pengguna untuk memberikan nilai peringkat suatu konten.
Keadaan Awal	Menampilkan halaman detail konten
Skenario Utama	
Aksi Aktor	**Reaksi Sistem**
1. Menekan tombol "Add Rate" pada jendela detail konten	
	2. Menampilkan skala nilai peringkat yang dapat dipilih pengguna.
3. Memberikan nilai peringkat konten.	
	4. Nilai peringkat konten dikirim ke server
Kondisi Akhir	Nilai peringkat berhasil dikirim ke server

3.2 Skenario *Use Case* Aplikasi *Backend*

Skenario proses-proses yang terdapat dalam *use case diagram* aplikasi *backend* dapat dilihat pada tabel-tabel berikut :

Tabel III.18 Skenario *Use Case Login*

Identifikasi	
Nama Use Case	*Login*
Aktor	Pengguna *Backend*
Tujuan	Masuk kedalam sistem
Keadaan Awal	Sistem menampilkan halaman login admin
Skenario Utama	
Aksi Aktor	**Reaksi Sistem**
1. Mengisi *username* dan *password* pada halaman web *login* admin	
	2. Sistem melakukan autentikasi *username* dan *password* yang diiunputkan.

	3. Bila sesuai maka menampilan halaman web menu utama.
Skenario Alternatif – Validasi Gagal	
Aksi Aktor	**Reaksi Sistem**
	1. Menampilkan pesan "*Error : Invalid username or password. Please try again*"
2. Melakukan input *username* dan *password* pada halaman web login admin	
Kondisi Akhir	Halaman menu utama ditampilkan

Tabel III.19 Skenario *Use Case* Kategori

Identifikasi	
Nama Use Case	Kategori
Aktor	Pengguna *Backend*
Tujuan	Menampilkan data kategori secara keseluruhan serta fungsionalitas pengolahan data
Keadaan Awal	Menampilkan halaman utama CMS *ARPublicFacilities*
Skenario Utama	
Aksi Aktor	**Reaksi Sistem**
1. Memilih Kategori pada menu navigator	
	2. Menampilkan halaman pengolahan kategori
Kondisi Akhir	Menampilkan data kategori.

Tabel Skenario III.20 *Use Case* Tambah Kategori

Identifikasi	
Nama Use Case	Tambah Kategori
Aktor	Pengguna *Backend*
Tujuan	Menambah data kategori
Keadaan Awal	Sistem menampilkan halaman pengelohan kategori.
Skenario Utama	
Aksi Aktor	**Reaksi Sistem**
1. Memilih *Tambah* pada pilihan halaman Kategori	
	2. Menampilkan halaman pengisian tambah kategori.
3. Mengisi *field-field* pada pengisian tambah *kategori*.	
	4. Melakukan proses tambah data pada basis data.
	5. Menyimpan hasil tambah data.

Skenario Alternatif – Validasi Gagal	
Aksi Aktor	**Reaksi Sistem**
	1. Menampilkan pesan bahwa "*Insert cannot perform*".
2. Melengkapi *field-field* pada pengisian tambah kategori	
Kondisi Akhir	Data kategori bertambah sesuai dengan input data.

Tabel III.21 Skenario *Use Case* Ubah Kategori

Identifikasi	
Nama Use Case	Ubah Kategori
Aktor	Pengguna *Backend*
Tujuan	Mengubah data kategori
Keadaan Awal	Sistem menampilan halaman pengolahan kategori
Skenario Utama	
Aksi Aktor	**Reaksi Sistem**
1. Memilih gambar pensil pada kolom *action* tabel Kategori	
	2. Menampilkan halaman halaman ubah kategori.
3. Mengisi *field-field* pada halaman ubah kategori	
	4. Melakukan proses ubah data pada basis data dan menyimpannya.
Skenario Alternatif – Validasi Gagal	
Aksi Aktor	**Reaksi Sistem**
-	1. Menampilan pesan "*Edit cannot perform*".
Keadaan Akhir	Perubahan data kategori sesuai dengan ubah data.

Tabel III.22 Skenario *Use Case* Hapus Kategori

Identifikasi	
Nama Use Case	Hapus Kategori
Aktor	Pengguna *backend*
Tujuan	Menghapus data kategori
Keadaan Awal	Sistem menampilkan halaman pengolahan kategori
Skenario Utama	
Aksi Aktor	**Reaksi Sistem**
1. Memilih gambar silang pada kolom *action* tabel Kategori	
	2. Menampilkan pesan konfirmasi

	hapus kategori
3. Menyetujui penghapusan dengan memilih atau menekan tombol “OK”	
	4. Menghapus data kategori.
	5. Menyimpan hasil perubahan data
Kondisi Akhir	Perubahan data kategori sesuai dengan data yang dihapus.

Tabel III.23 Skenario *Use Case* Konten

Identifikasi	
Nama Use Case	Konten
Aktor	Pengguna *Backend*
Tujuan	Menampilkan data kategori secara keseluruhan serta fungsionalitas pengolahan data
Keadaan Awal	Menampilkan halaman utama CMS *ARPublicFacilities*
Skenario Utama	
Aksi Aktor	**Reaksi Sistem**
1. Memilih Konten pada pilihan menu *navigator*	
	2. Menampilkan halaman pengolahan *kategori*
Kondisi Akhir	Menampilkan data konten.

Tabel III.24 Skenario *Use Case* Tambah Konten

Identifikasi	
Name Use Case	Tambah Konten
Aktor	Pengguna *Backend*
Tujuan	Menambah data konten
Keadaan Awal	Sistem menampilkan halaman konten
Skenario Utama	
Aksi Aktor	**Reaksi Sistem**
1. Memilih *Tambah* pada pilihan halaman Konten	
	2. Menampilkan halaman isian tambah konten.
3. Mengisi *field-field* pada halaman tambah konten	
	4. Melakukan proses tambah data pada basis data dan menyimpannya..
Skenario Alternatif – Validasi Gagal	
Aksi Aktor	**Reaksi Sistem**
	1. Menampilkan pesan “*Insert cannot perform*”.
2. Melengkapi input data pada	

field-field tambah konten	
Kondisi Akhir	Data konten bertambah sesuai dengan input data.

Tabel III.25Skenario *Use Case* Ubah Konten

Identifikasi	
Nama Use Case	Ubah Konten
Aktor	Admin
Tujuan	Mengubah data konten
Keadaan Awal	Menampilkan halaman pengolahan konten.
Skenario Utama	
Aksi Aktor	**Reaksi Sistem**
1. Memilih gambar pensil pada kolom *action* tabel Konten	
	2. Menampilkan halaman ubah konten
3. Mengubah data pada field-field yang diinginkan	
	4. Melakukan proses ubah data pada basis data
	5. Menyimpan hasil perubahan data.
Skenario Alternatif – Validasi Gagal	
Aksi Aktor	**Reaksi Sistem**
	1. Menampilan pesan "*Ubah cannot perform".*
Kondisi Akhir	Perubahan data konten sesuai dengan ubah konten.

Tabel III.26 Skenario *Use Case* Hapus Konten

Identifikasi	
Nama Use Case	Hapus Konten
Aktor	Admin
Tujuan	Menghapus data konten
Keadaan Awal	Sistem menampilkan halaman pengolahan konten
Skenario Utama	
Aksi Aktor	**Reaksi Sistem**
1. Memilih gambar silang pada kolom *action* tabel Konten	
	2. Menampilkan pesan konfirmasi hapus konten.
3. Menyetujui penghapusan dengan memilih atau menekan tombol "OK"	
	4. Menghapus data konten

	5. Menyimpan hasil perubahan data
Kondisi Akhir	Perubahan data konten sesuai data yang dihapus.

Tabel III.27 Skenario Use Case Komentar

Identifikasi	
Nama Use Case	Komentar
Aktor	Pengguna Backend
Tujuan	Menampilkan semua ulusan mengenai konten.
Keadaan awal	Sistem menampilkan halaman utama CMS
Skenario Utama	
Aksi Aktor	**Reaksi Sistem**
1. Memilih tombol "Review's Here" pada menu navigator	
	2. Menampilkan data komentar yang telah masuk.
Keadaan akhir	Data komentar konten ditampilkan

Tabel III.28 Skenario Use Case Hapus Komentar

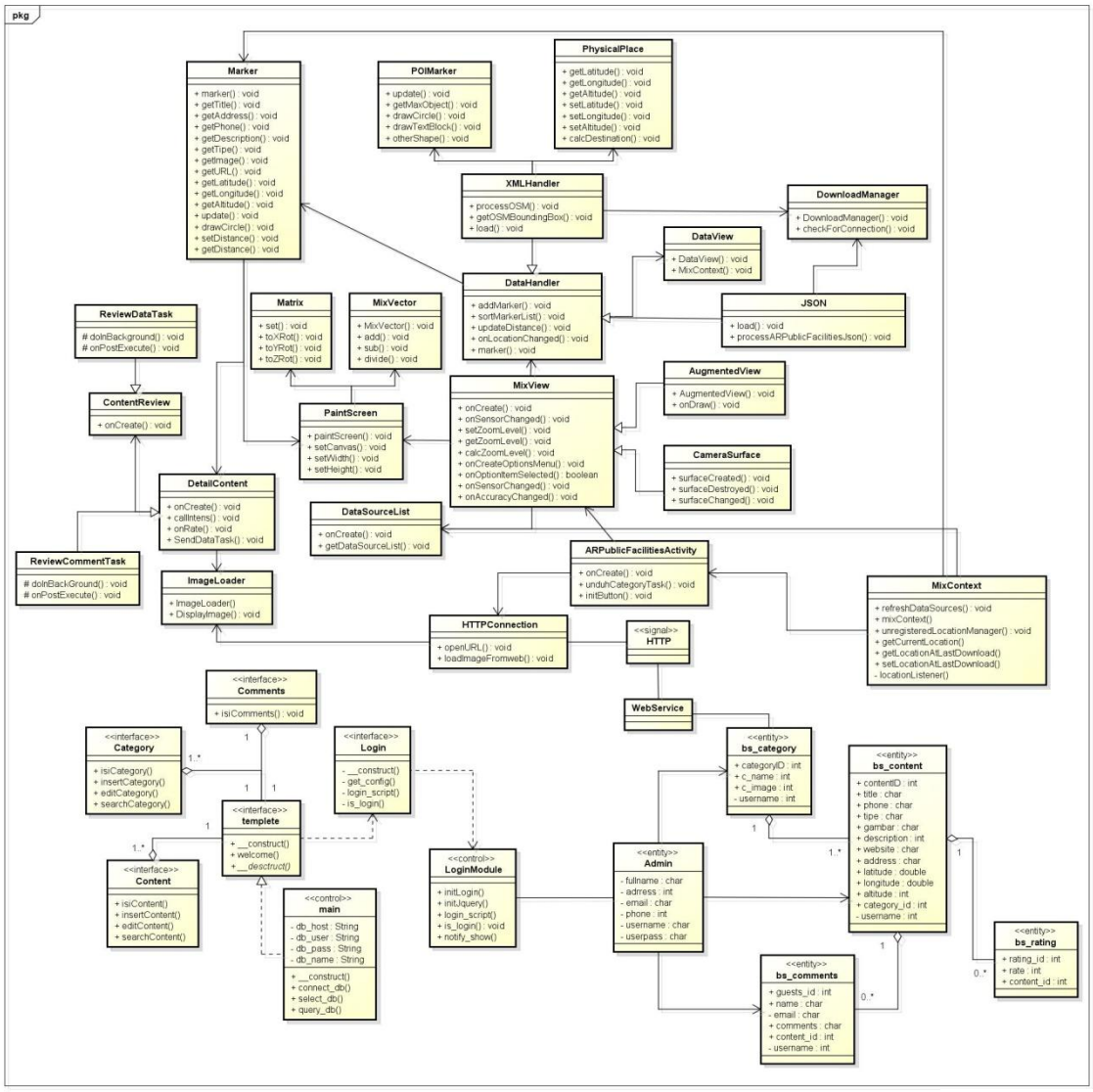
Identifikasi	
Nama Use Case	Hapus Komentar
Aktor	Pengguna Backend
Tujuan	Menghapus data komentar
Keadaan Awal	Menampilkan halamaan utama CMS ARPublicFacilities
Skenario Utama	
Aksi Sistem	**Reaksi Sistem**
1. Memilih tombol "Review's here" pada jendela detail konten	
	2. Menampilkan seluruh data peringkat konten.
3. Mengklik gambar silang pada kolom *action* untuk menghapus data satu baris	
	4. Menghapus data komentar
Keadaan akhir	Data komentar berhasil dihapus.

2 *Class Diagram*

Class diagram adalah sebuah spesifikasi yang jika diinstansiasi akan menghasilkan sebuah objek dan merupakan inti dari pengembangan dan desain berorientasi objek. *Class* menggambarkan keadaan (*attribut* atau *property*) suatu sistem, sekaligus menawarkan layanan untuk memanipulasi keadaan tersebut (metoda atau fungsi).

Berikut adalah *class diagram* dari sistem aplikasi *ARPublicFacilities*.

1) Class Diagram ARPublicFacilities



Gambar III.10 *Class Diagram ARPublicFacilities*

Tabel III.29 Deskripsi *Class Diagram ARPublicFacilities*

Backend Application		
Class	**Jenis Class**	**Deskripsi**
bs_konten	Entity	Class yang berisikan data konten
bs_kategori	Entity	Class yang berisikan data kategori
bs_admin	Entity	Class yang berisikan data admin
bs_comments	Entity	Class yang berisikan data komentar
bs_rating	Entity	Class yang berisikan data rating
Web Administrator		
Class	**Jenis Class**	**Deskripsi**
Category	*Interface*	*Class* yang berisikan tampilan data kategori
Contents	*Interface*	*Class* yang berisikan tampilan data konten
Comments	*Interface*	*Class* yang berisikan tampilan data komentar
Template	*Interface*	*Class* yang berisikan menu navigator web
Login	*Interface*	*Class* yang berisikan tampilan login web
Main	*Control*	*Class* yang berisikan *method-method* untuk mengontrol *class interface*
LoginModule	*Control*	*Class* yang berisikan *method-method* untuk mengontrol *class interface* Login
Frontend Application		
Class	**Jenis Class**	**Deskripsi**
ARPublicFacilities	*Interface*	Class yang berisikan *method-method* untuk fungsionalitas tampilan utama aplikasi yaitu menampilkan kategori
HTTPConnection	*Control*	Class yang berisikan *method-method* untuk menghubungkan aplikasi dengan webservice
DownloadManager	*Control*	Class yang berisikan *method-method* untuk membangun koneksi ke webservice dan men-download data
Marker	*Control*	Class yang berisikan *method-method* untuk menciptakan marker dari setiap informasi konten
MixView	*Control*	Class utama dari aplikasi yang mengontrol *class-class* dengan fungsionalitas yang berbeda.
POIMarker	*Control*	Class yang berisikan *method-method* untuk fungsionalitas penanda.
DataHandler	*Control*	Class yang berisikan *method-method*

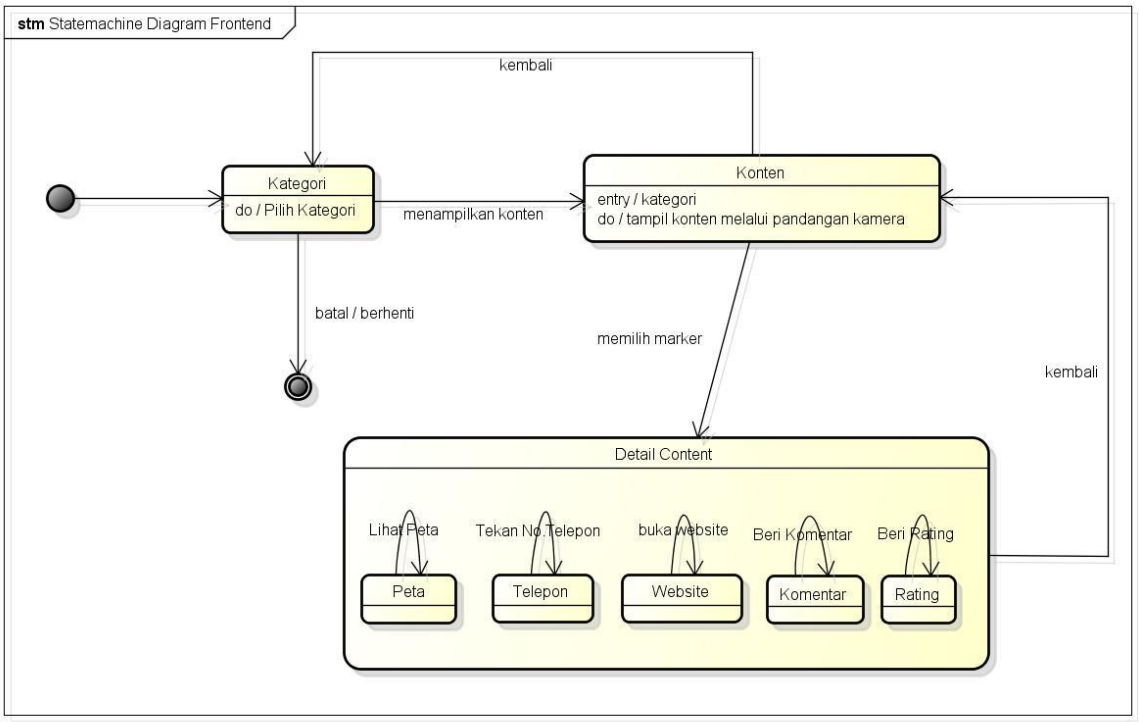
		yang menyediakan objek penanda pada data tertentu.
DataSourceList	*Control*	Class yang berisikan *method-method* untuk menyimpan link (alamat sumber data).
JSON	*Control*	Class yang berisikan *method-method* untuk fungsionalitas menyusun urutan penanda.
XMLHandler	*Control*	Class yang berisikan *method-method* untuk fungsionalitas menempatkan marker pada setiap titik koordinat
PaintScreen	*Interface*	Class yang berisikan *method-method* untuk fungsionalitas tampilan augmented.
PhysicalPlace	*Control*	Class yang berisikan *method-method* untuk fungsionalitas menyimpan informasi geografis dan data tampilan pada peta. Menghitung jarak dengan menggunakan teori "*great-circle-distance*"
Matrix	*Control*	Class yang merepresentasikan matrix 3x3
DetailKonten	*Interface*	Class yang berisikan *method-method* untuk fungsionalitas menampilkan detail konten.
AugmentedView	*Interface*	Class yang memiliki fungsionalitas menampilkan augmented marker dalam layar handphone.
CameraView	*Interface*	Class yang mengaktifkan fungsionalitas kamera pada handhpone untuk ditampilkan augmented view.
DataView	*Control*	Class yang berisikan *method-method* untuk fungsionalitas menampilkan menu kontrol aplikasi.
ImageLoader	*Control*	Class yang berisikan method-method untuk me-*load* gambar dari webservice ke aplikasi
ContentReview	*Control*	Class yang berisikan *method-method* untuk fungsionalitas mengirim data review ke database server melalui webservice.
MixContext	*Control*	*Class* yang berisikan *method-method* untuk fungsionalitas menentukan lokasi pengguna
ReviewCommentsTask	*Control*	*Class* yang berisikan *method-method* untuk fungsionalitas menampilkan komentar
ReviewDataTask	*Control*	*Class* yang berisikan *method-method*

		untuk fungsionalitas mengirim data komentar dari *frontend* ke *backend*.

3 *State Machine Digram*

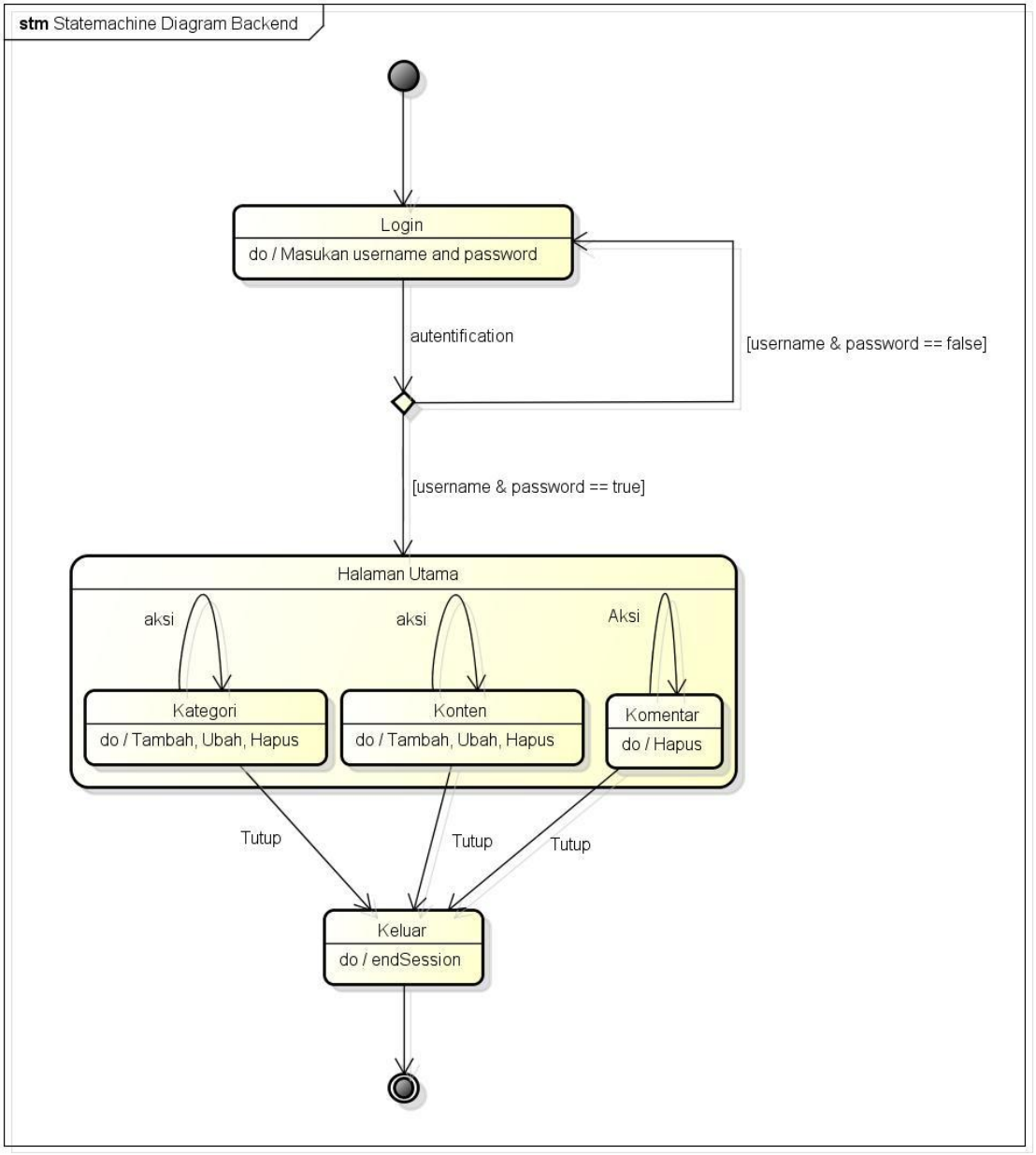
State machine diagram mengekspresikan perilaku sebagai sebuah progresif (perkembangan) melalui sekumpulan *state*, *triggered by events*, dan *action* yang mungkin terjadi. *State machine diagram* juga biasa dikenal sebagai *behavioural state diagram. State machine diagram* biasanya digunakan untuk mendeskripsikan perilaku sebuah *object*. Namun, juga dapat digunakan pada elemen yang besar dari sistem apapun.

1) *State Machine Diagram* Aplikasi *Frontend*



Gambar III.11 *State Machine Diagram* Aplikasi *Frontend*

2) *State Machine Diagram* Aplikasi *Backend*

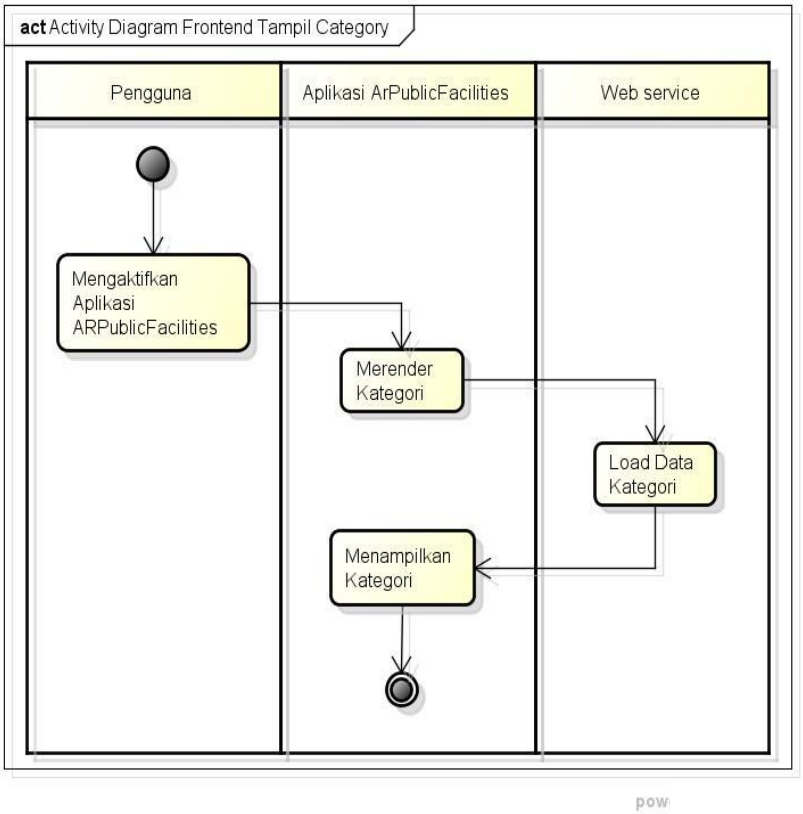


Gambar III.12 *State Diagram* Aplikasi *Backend*

4 *Activity Diagram*

Diagram activity menggambarkan berbagai alir aktivitas dalam sistem yang dirancang, bagaimana masing-masing alir berawal, *decision* yang mungkin terjadi dan bagaimana mereka berakhir. Penggambaran *activity* diagram memiliki kemiripan dengan *flowchart diagram*. *Activity diagram* memodelkan *event-event* yang terjadi pada *Use Case* dan digunakan untuk pemodelan aspek dinamis dari sistem.

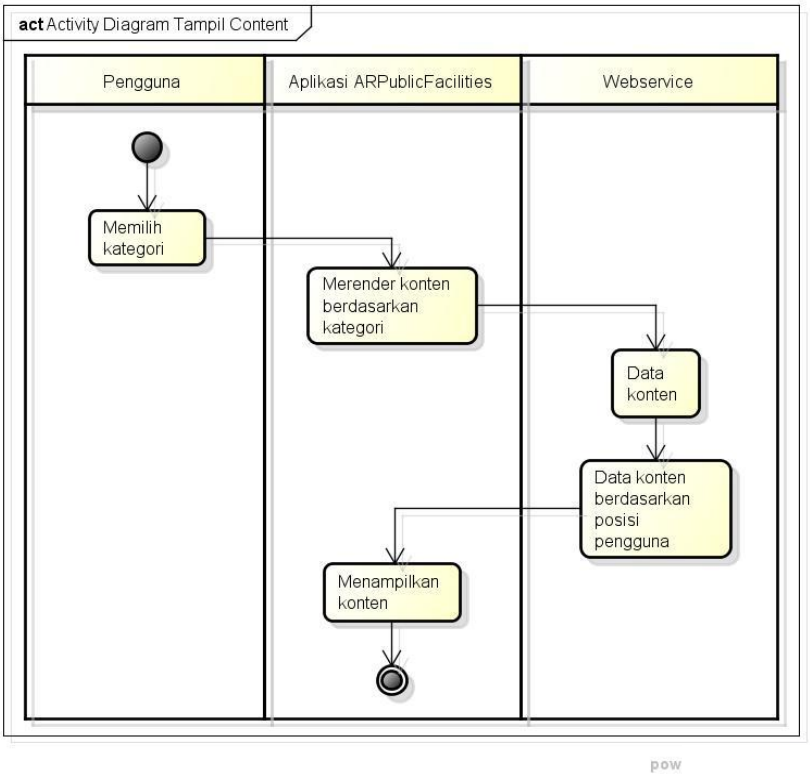
1) *Activity Diagram* Lihat Kategori



Gambar III.13 *Activity Diagram* Lihat Kategori

Gambar III.13, menggambarkan aktivitas yang terjadi pada aplikasi saat pengguna pertama kali mengaktifkan aplikasi. Dimana dalam aplikasi akan menyajikan beberapa jenis kategori yang dapat dipilih pengguna.

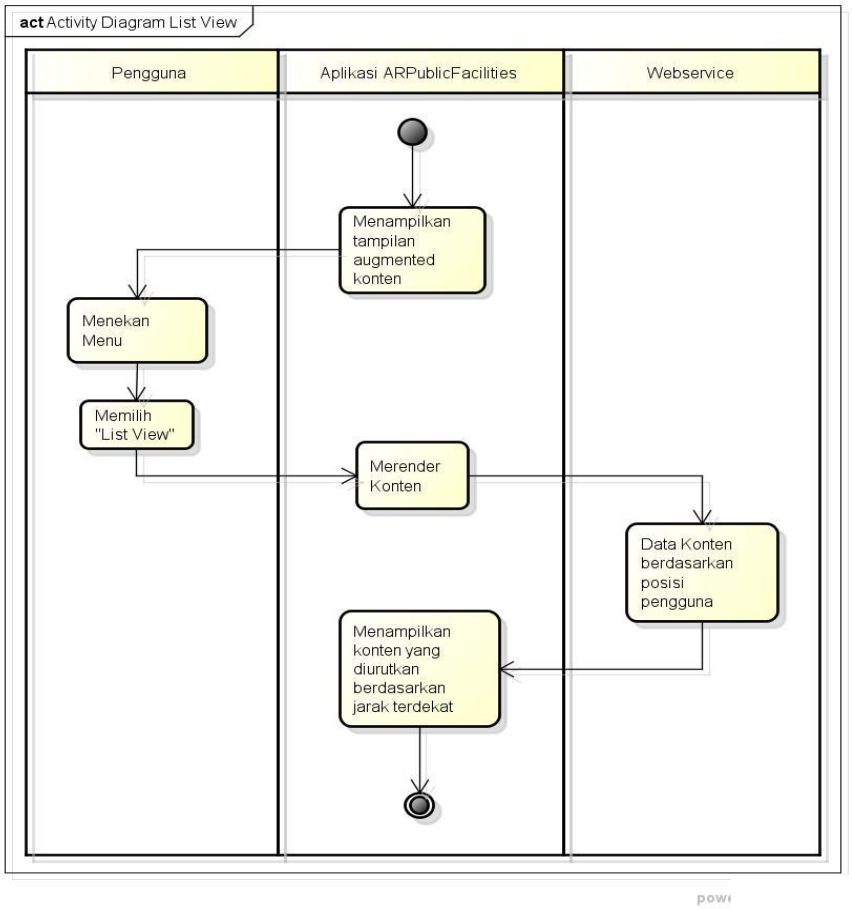
2) *Activity Diagram* Lihat Konten Melalui Pandangan Kamera



Gambar III.14 *Activity Diagram* Lihat Konten Melalui Pandangan Kamera

Gambar III.14, menggambarkan aktivitas ketika pengguna memilih kategori. Pengguna memilih salah satu kategori, selanjutnya aplikasi akan me-*request* data pada server (web service) untuk menampilkan konten yang sesuai dengan jenis kategori yang dipilih. Web service melakukan proses pencarian data konten berdasarkan kategori kemudian melakukan pengurutan data berdasarkan lokasi pengguna, hasilnya dikirim ke aplikasi untuk kemudian ditampilkan dalam layar *handphone*.

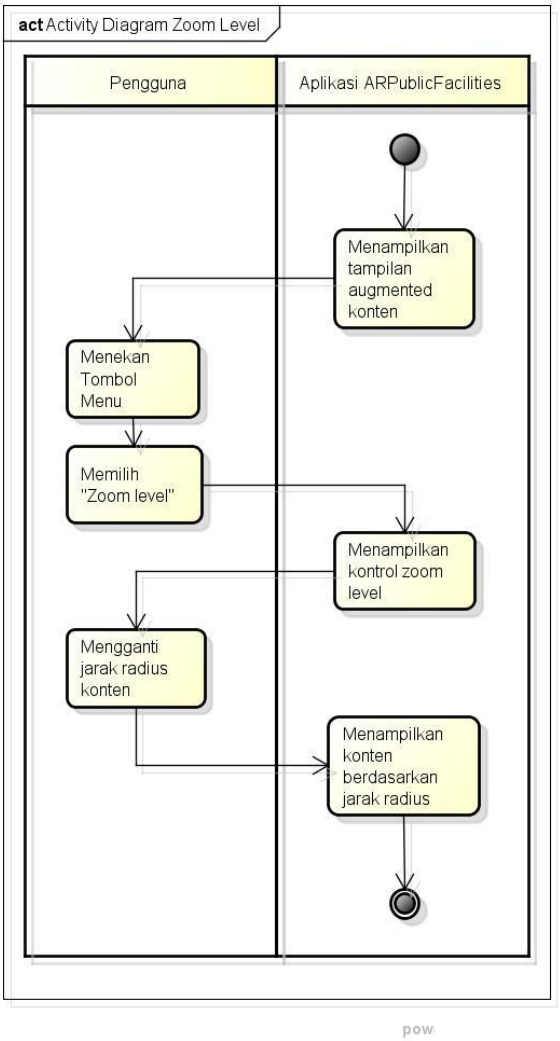
3) *Activity Diagram* Daftar Konten (*List View*)



Gambar III.15 *Activity Diagram* Daftar Konten (*List View*)

Gambar III.15, menggambarkan aktivitas menampilkan daftar semua konten. Aktivitas ini muncul ketika kategori telah dipilih dan aplikasi menampilkan konten berdasarkan kategori telah dipilih sebelumnya. Pengguna dapat menekan tombol "*Menu"* kemudian memilih tombol "*List View"* untuk melihat semua informasi konten yang ada.

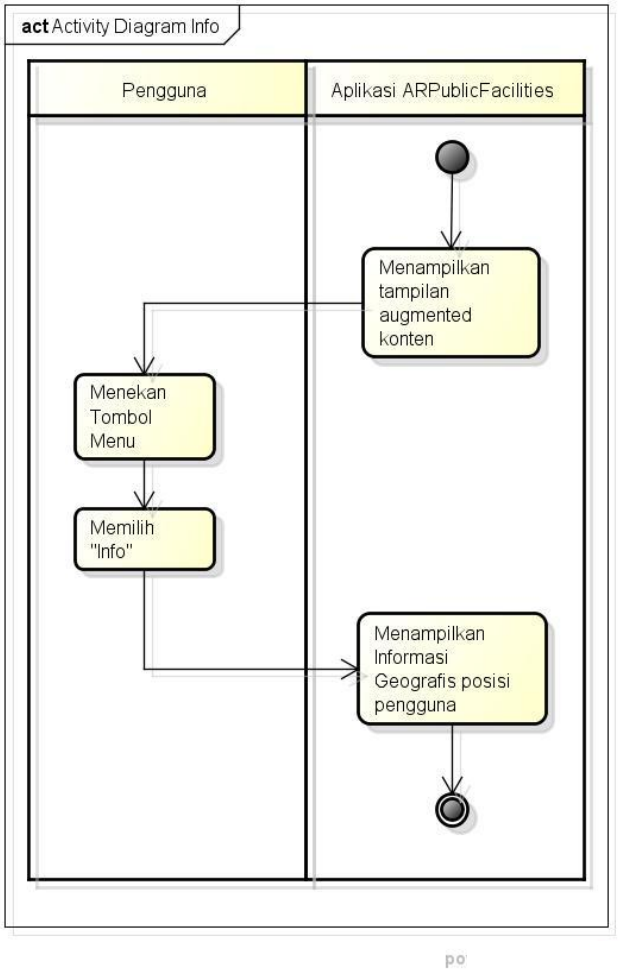
4) *Activity Diagram Zoom Level*



Gambar III.16 *Activity Diagram Zoom Level*

Gambar III.16, menggambarkan aktivitas pengaturan jarak radius konten yang ingin ditampilkan pada tampilan *augmented*. Aktivitas ini muncul ketika kategori telah dipilih dan aplikasi menampilkan konten berdasarkan kategori telah dipilih. Pengguna dapat menekan tombol "*Menu"* kemudian memilih tombol "*Zoom Level"*. Menu ini berfungsi untuk memberikan pilihan kepada pengguna untuk dapat mengatur jarak radius konten yang ingin ditampilkan.

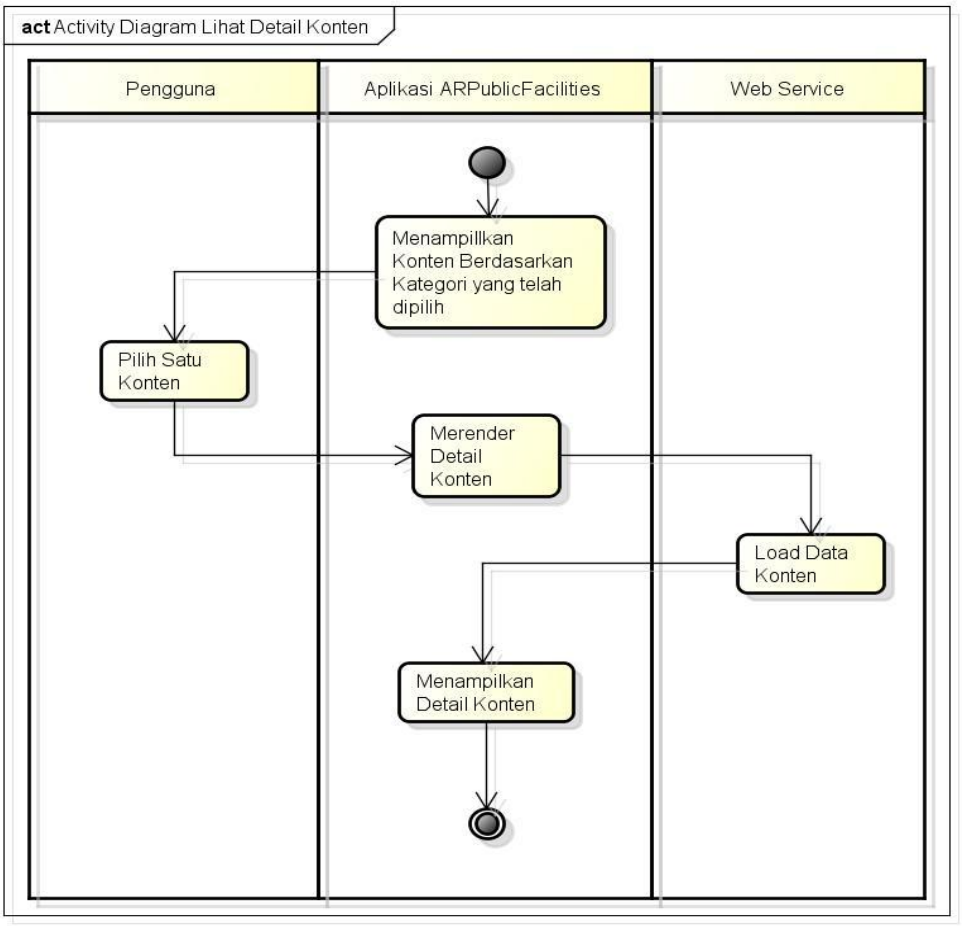
5) *Activity Diagram* Info



Gambar III.17 *Activity Diagram* Info

Gambar III.17 menggambarkan aktivitas menampilkan informasi geografis posisi pengguna terkini. Aktivitas ini muncul ketika kategori telah dipilih dan aplikasi menampilkan konten berdasarkan kategori yang telah dipilih sebelumnya. Pengguna dapat menekan tombol "*Menu"* kemudian memilih tombol "*Info"*.

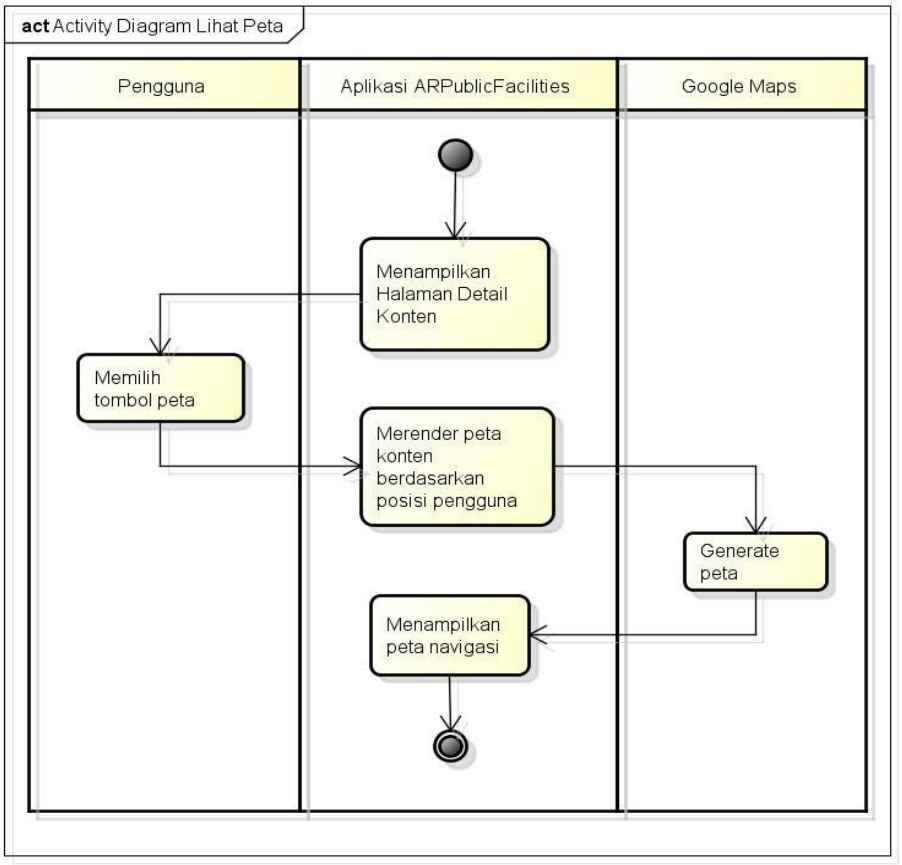
6) *Activity Diagram* Lihat Detail Konten



Gambar III.18 *Activity Diagram* Lihat Detail Konten

Gambar III.18 menggambarkan aktivitas menampilkan detail konten. Aktivitas ini muncul ketika aplikasi menampilkan semua informasi konten setelah pengguna memilih menu "*List View*" atau keadaan dimana aplikasi menampilkan tampil konten dengan kamera. Selanjutnya pengguna dapat memilih salah satu konten, kemudian aplikasi akan menampilkan detail dari konten tersebut.

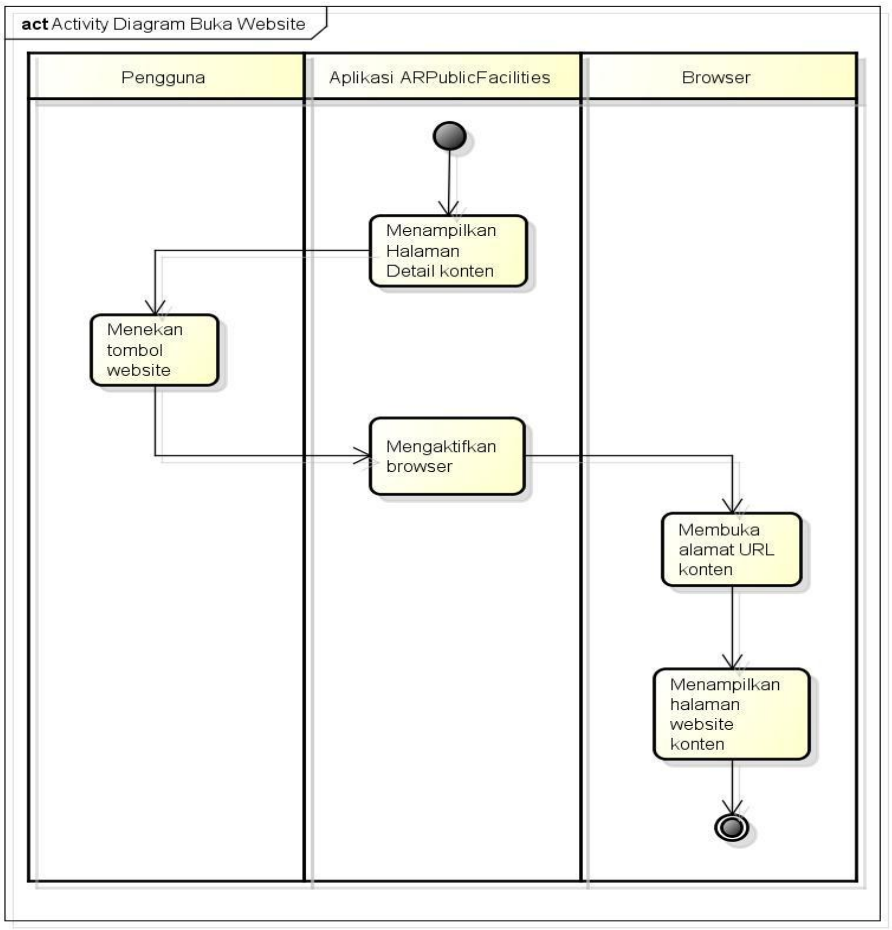
7) *Activity Diagram* Lihat Peta



Gambar III.19 *Activity Diagram* Lihat Peta

Gambar III.19 menggambarkan aktivitas menampilkan peta rute navigasi. Aktivitas ini muncul ketika aplikasi menampikan detail konten, kemudian pengguna menekan atau memilih tombol "Peta". Selanjutnya aplikasi akan mengunduh peta Google Maps untuk ditampilkan dalam layar yaitu peta rute navigasi.

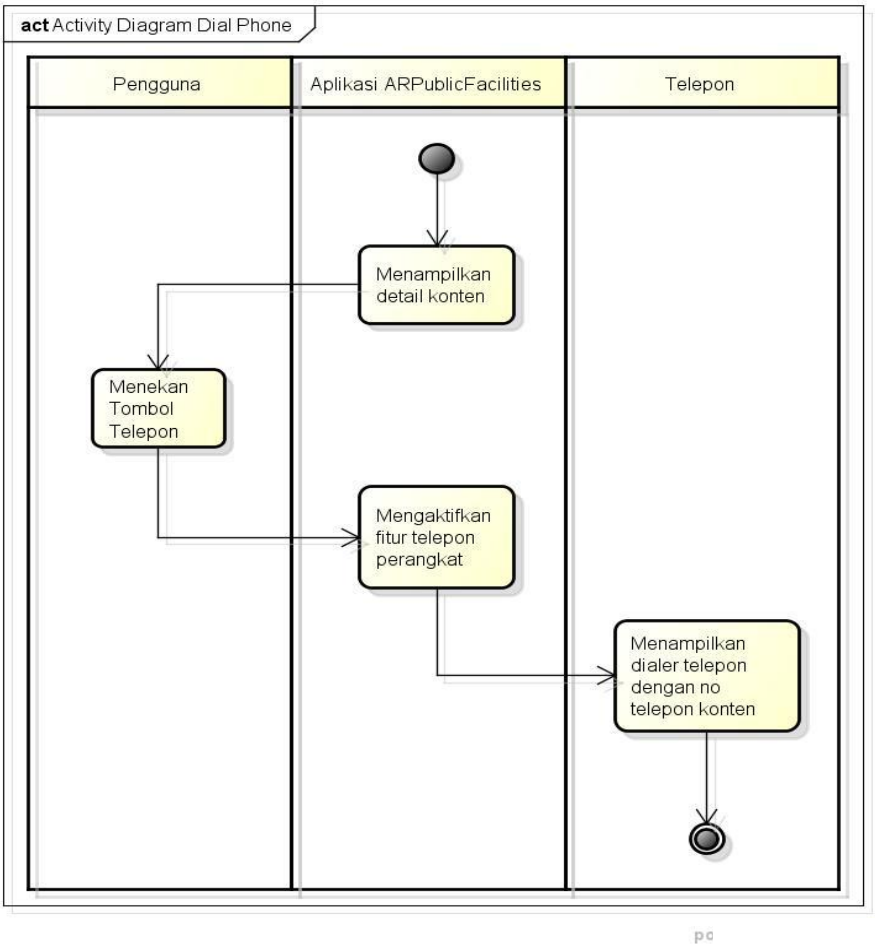
8) *Activity Diagram* Buka Website



Gambar III.20 *Activity Diagram* BukaWebsite

Gambar III.20, menggambarkan aktivitas menampilkan halaman website konten. Aktivitas ini muncul ketika aplikasi menampikan detail konten, kemudian pengguna menekan atau memilih tombol "*website*", selanjutnya aplikasi akan membuka browser untuk membuka alamat website konten tersebut.

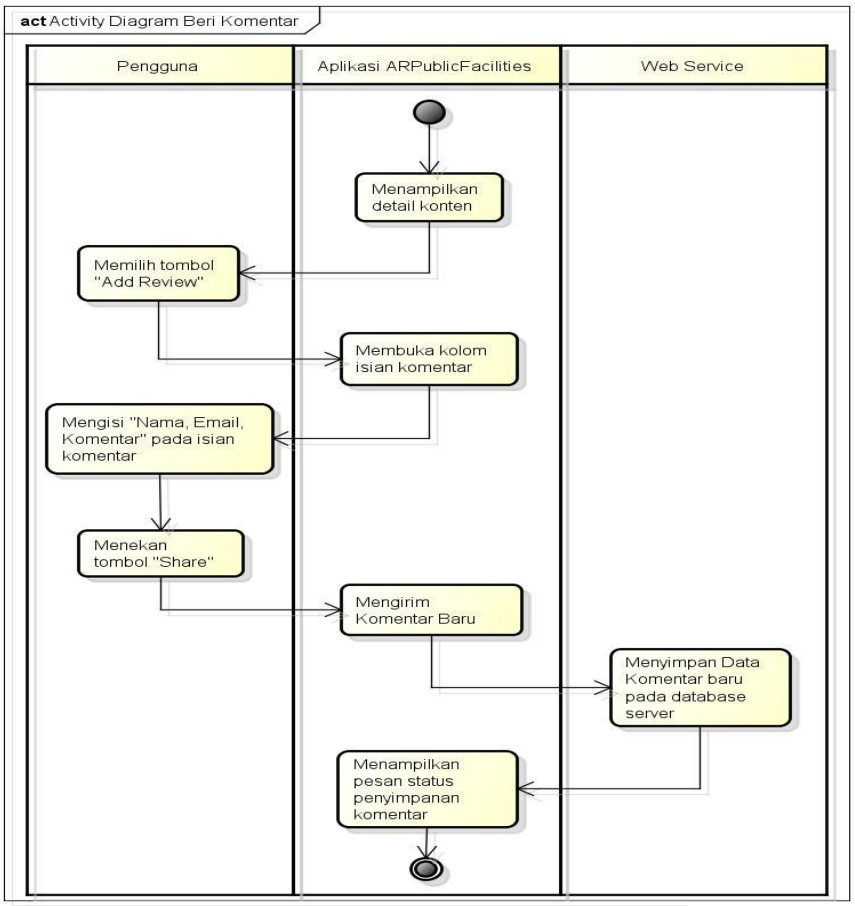
9) *Activity Diagram* Tekan No.Telepon



Gambar III.21 *Activity Diagram* Tekan No.Telepon

Gambar III.21, menggambarkan aktifitas untuk fungsionalitas menekan nomor telepon konten (*Phone Dialer*). Aktivitas ini muncul ketika aplikasi menampikan detail konten, kemudian pengguna menekan atau memilih tombol ‘‘Telepon*’’,* selanjutnya aplikasi akan menampilkan *phone dialer* beserta nomor telepon konten.

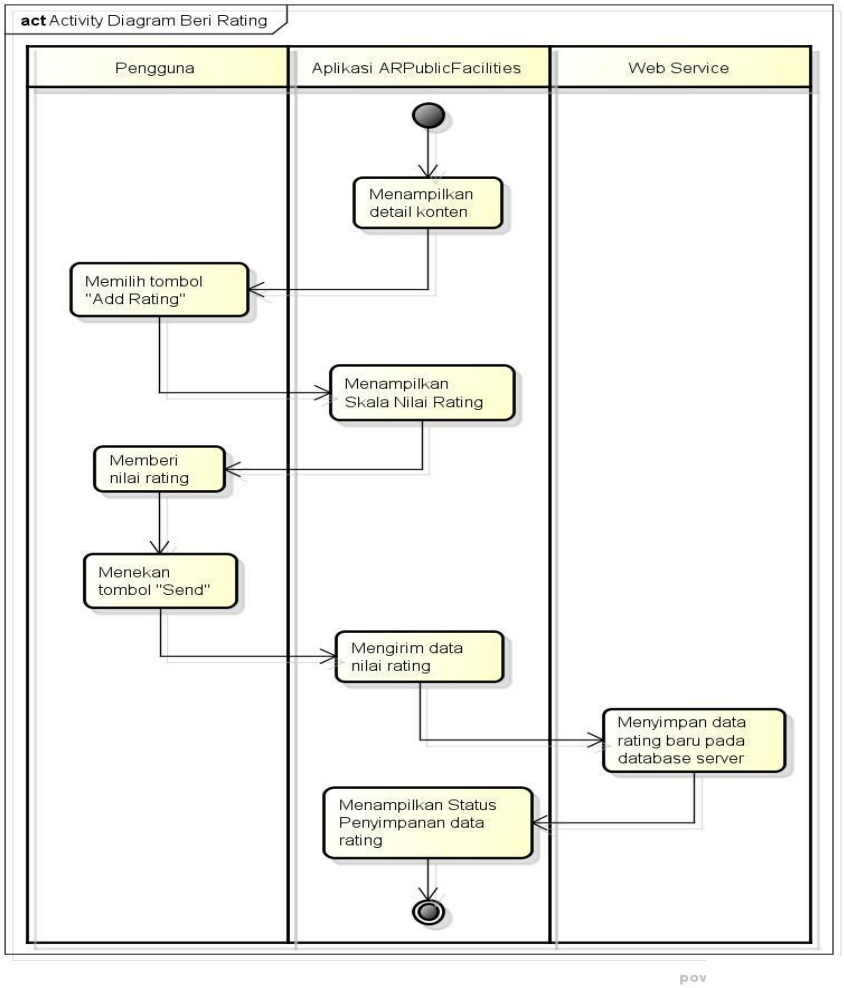
10) *Activity Diagram* Beri Komentar



Gambar III.22 *Activity Diagram* Beri Komentar

Gambar III.22, menggambarkan aktifitas untuk fungsionalitas memberi komentar pada suatu konten. Aktifitas ini muncul ketika aplikasi menampilkan detail konten, kemudian pengguna menekan atau memilih tombol "*Add Review*", selanjutnya aplikasi akan menampilkan isian review yang harus diisi oleh pengguna. Hasil isian komentar baru akan dikirim ke database server melalui web service.

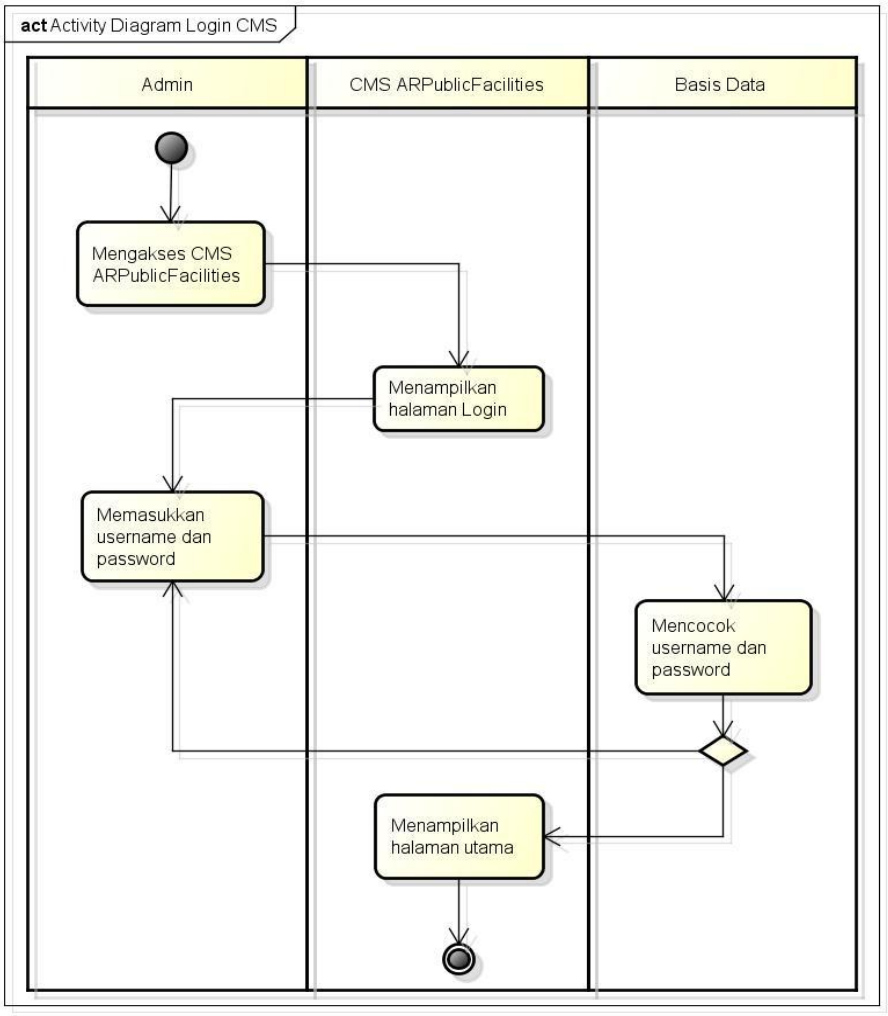
11) *Activity Diagram* Beri *Rating*



Gambar III.23 *Activity Diagram* Beri *Rating*

Gambar III.23, menggambarkan aktifitas untuk fungsionalitas memberi *rating* pada suatu konten. Aktifitas ini muncul ketika aplikasi menampilkan detail konten, kemudian aplikasi akan menampilkan skala nilai *rating* yang dapat dipilih oleh pengguna. Selanjutnya pengguna menekan tombol "*Send*" setelah memberi nilai *rating*, kemudian aplikasi akan mengirim data rating baru untuk kemudian disimpan pada database server melalui web service.

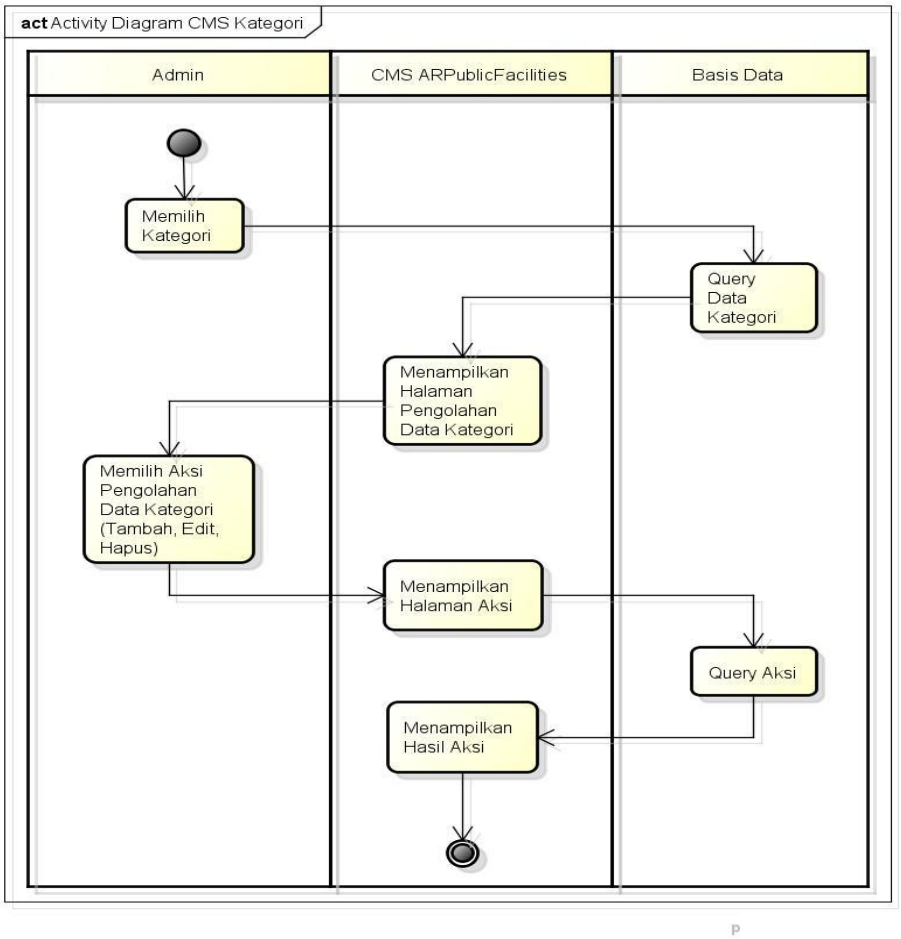
12) *Activity Diagram Login* CMS



Gambar III.24 *Activity Diagram* Login CMS

Gambar III.24, menggambarkan aktivitas login untuk masuk ke halaman utama CMS. Admin mengakses halaman CMS, kemudian CMS akan menampilkan halaman login, kemudian admin melakukan input *username* dan *password*. Server akan melakukan verifikasi, jika *username* dan *password* tidak sesuai maka admin akan diarahkan kembali ke halaman login untuk melakukan input *username* dan *password* yang valid. Jika *username* dan *password* yang di input valid, maka akan ditampilkan halaman utama web CMS.

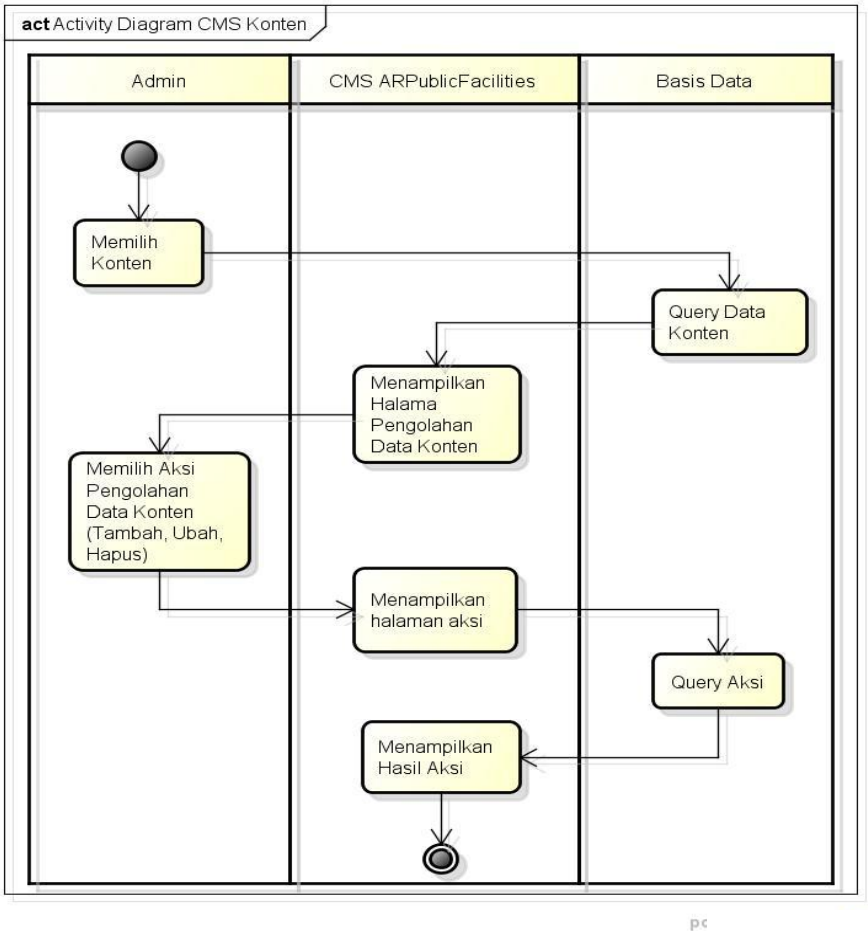
13) *Activity Diagram* CMS Kategori



Gambar III.25 *Activity Diagram* CMS Kategori

Gambar III.25, menggambarkan aktivitas pengolahan data kategori. Proses ini berjalan setelah admin berhasil melakukan login. Admin memilih menu kategori, selanjutnya CMS akan menampilkan halaman kategory. Admin dapat mengelola data kategori dengan menggunakan aksi Tambah, Ubah, atau Hapus. CMS akan menampilkan halaman berdasarkan aksi yang dilakukan admin, saat bersamaan *query* akan dipanggil dan dieksekusi pada database server kemudian hasilnya ditampilkan pada web CMS

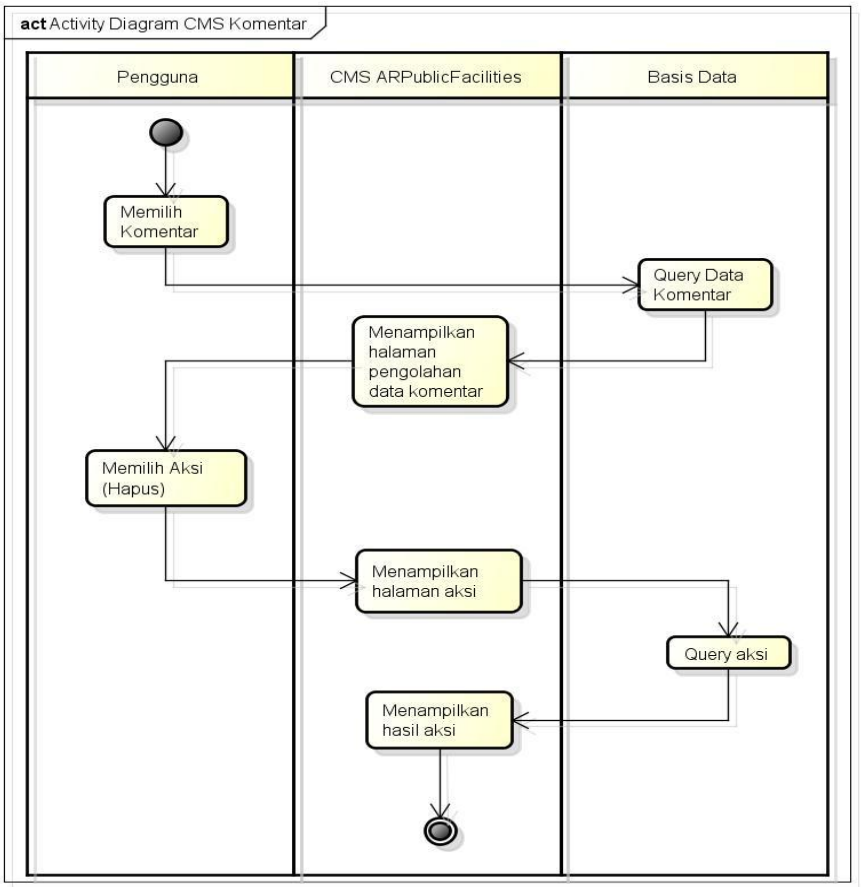
14) *Activity Diagram* CMS Konten



Gambar III.26 *Activity Diagram* CMS Konten

Gambar III.26 menggambarkan aktivitas pengolahan data konten. admin memilih menu konten, selanjutnya CMS akan menampilkan halaman konten. Admin dapat mengelola data konten dengan menggunakan aksi Tambah, Ubah, atau Hapus. CMS akan menampilkan proses berdasarkan aksi yang ingin dilakukan admin, saat bersamaan *query* dieksekusi pada database server kemudian hasilnya ditampilkan pada aplikasi CMS.

15) *Activity Diagram* CMS Komentar



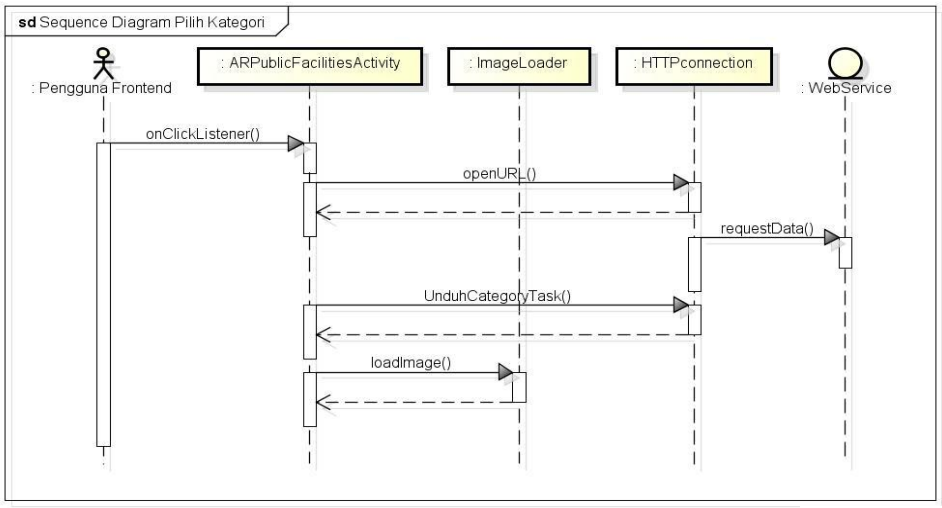
Gambar III.27 *Activity Diagram* CMS Komentar

Gambar III.27, menggambarkan aktifitas pengolahan data komentar. Admin memilih menu komentar, selanjutnya CMS akan menampilkan halaman komentar. Admin hanya dapat mengelola data komentar yaitu hapus komentar, karena fungsionalitas penambahan data komentar dilakukan oleh pada aplikasi frontend. Tujuan dari penghapusan ini yaitu untuk menyaring komentar-komentar yang masuk atau mencegah adanya komentar yang berisi komentar negatif.

5 *Sequence Diagram*

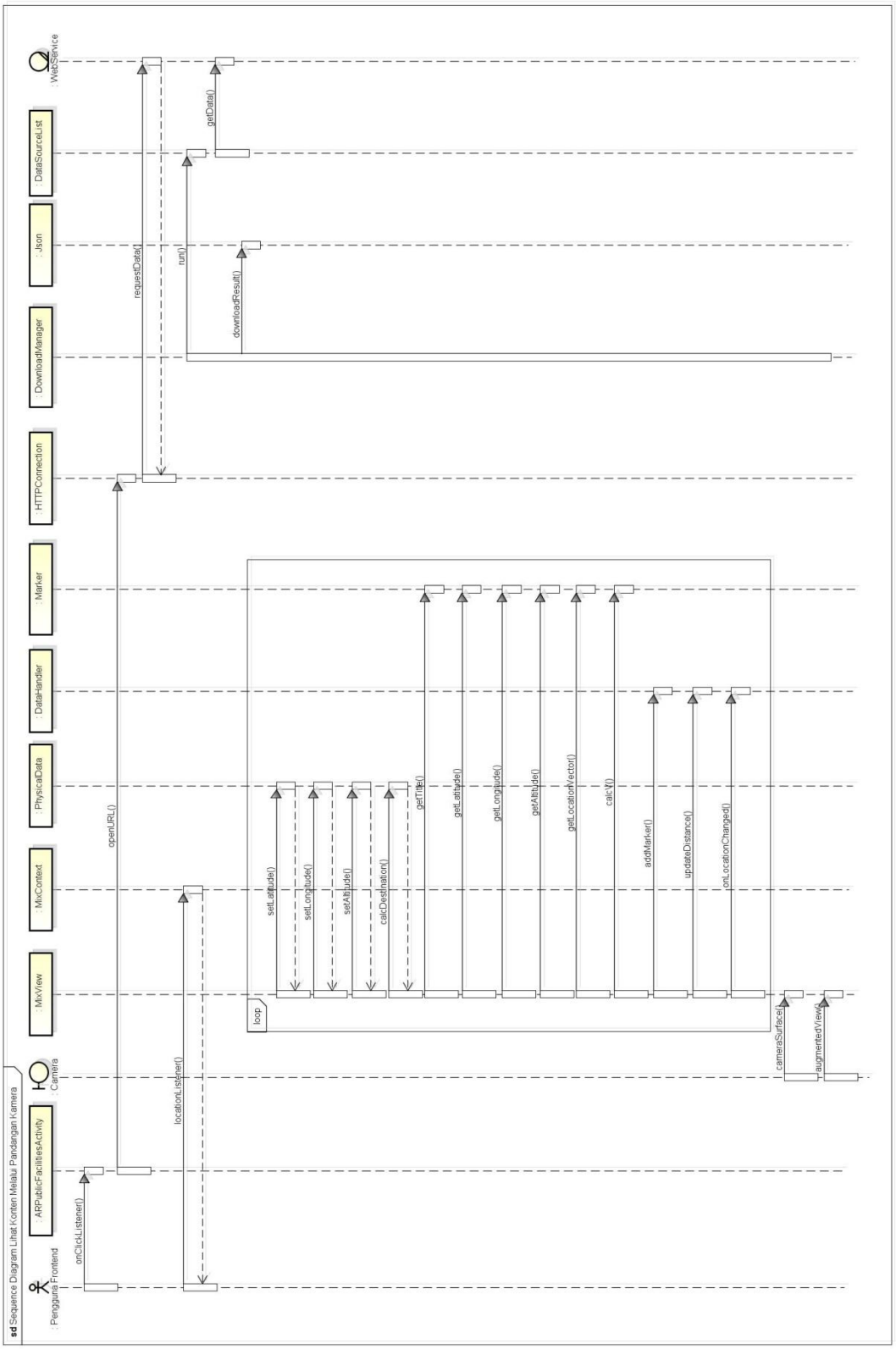
Sequence diagram menggambarkan perilaku pada sebuah *scenario*. Diagram ini menunjukkan sejumlah contoh obyek dan *message* (pesan) yang diletakkan diantara obyek-obyek ini di dalam use case. Komponen utama *sequence diagram* terdiri atas obyek yang dituliskan dengan kotak segiempat. *Message* diwakili oleh garis dengan tanda panah dan waktu yang ditunjukkan dengan progres vertikal.

1) *Sequence Diagram* Pilih Kategori



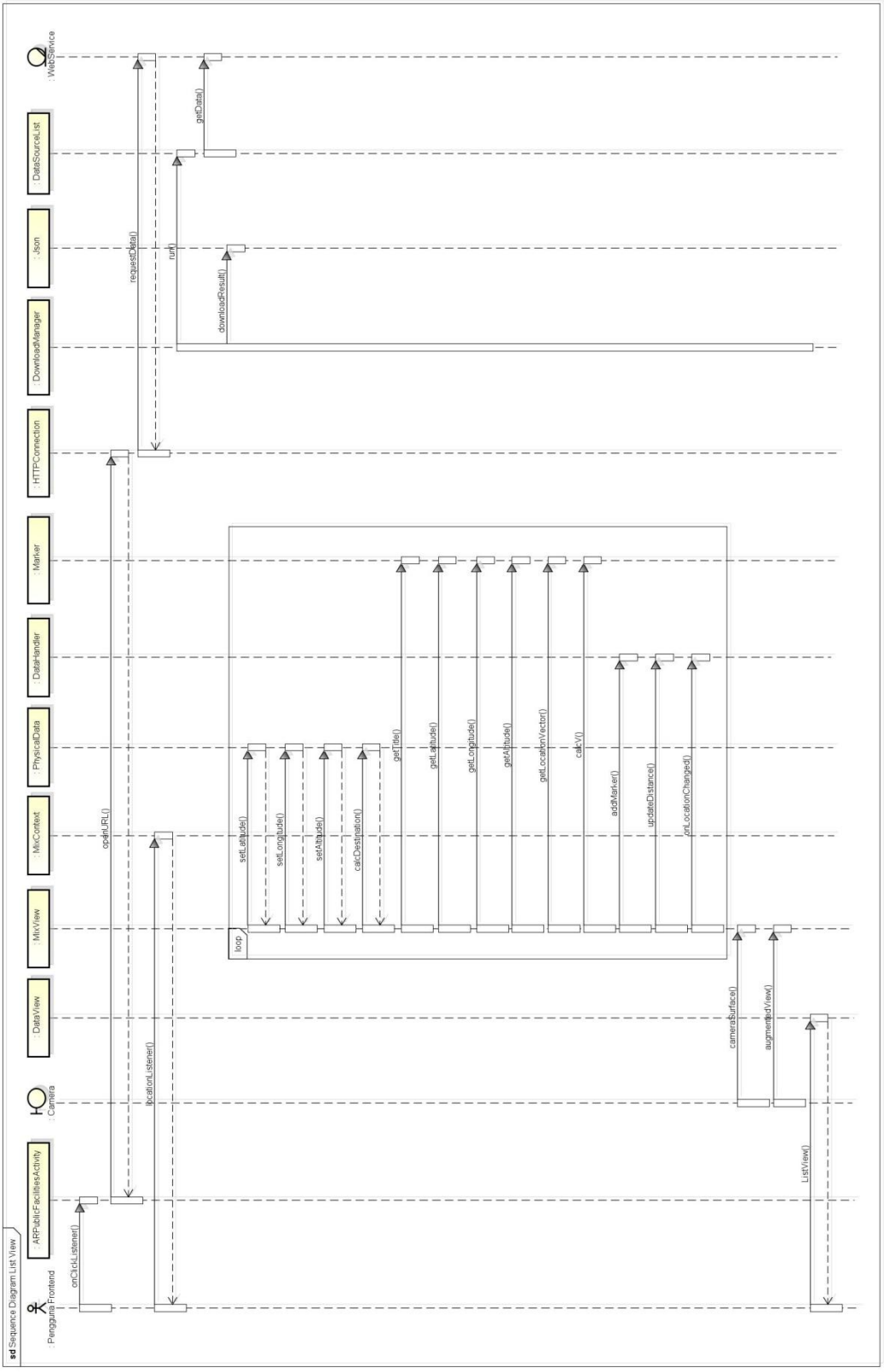
Gambar III.28 *Sequence Diagram* Pilih Kategori

2) *Sequence Diagram* Lihat Konten Melalui Pandangan Kamera



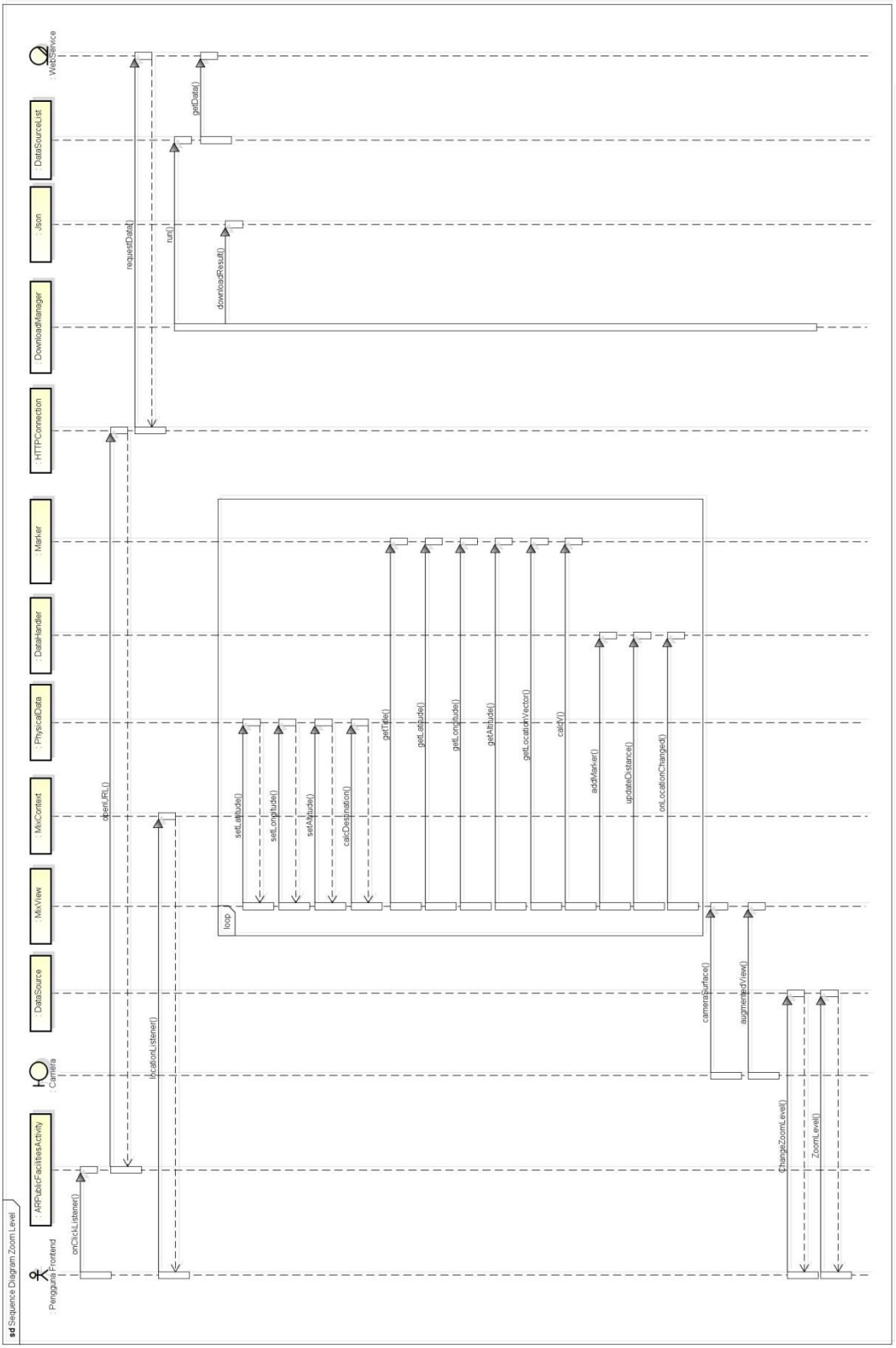
Gambar III.29 *Sequence Diagram* Lihat Konten Melalui Pandangan Kamera

3) *Sequence Diagram* Daftar Konten (*List View*)



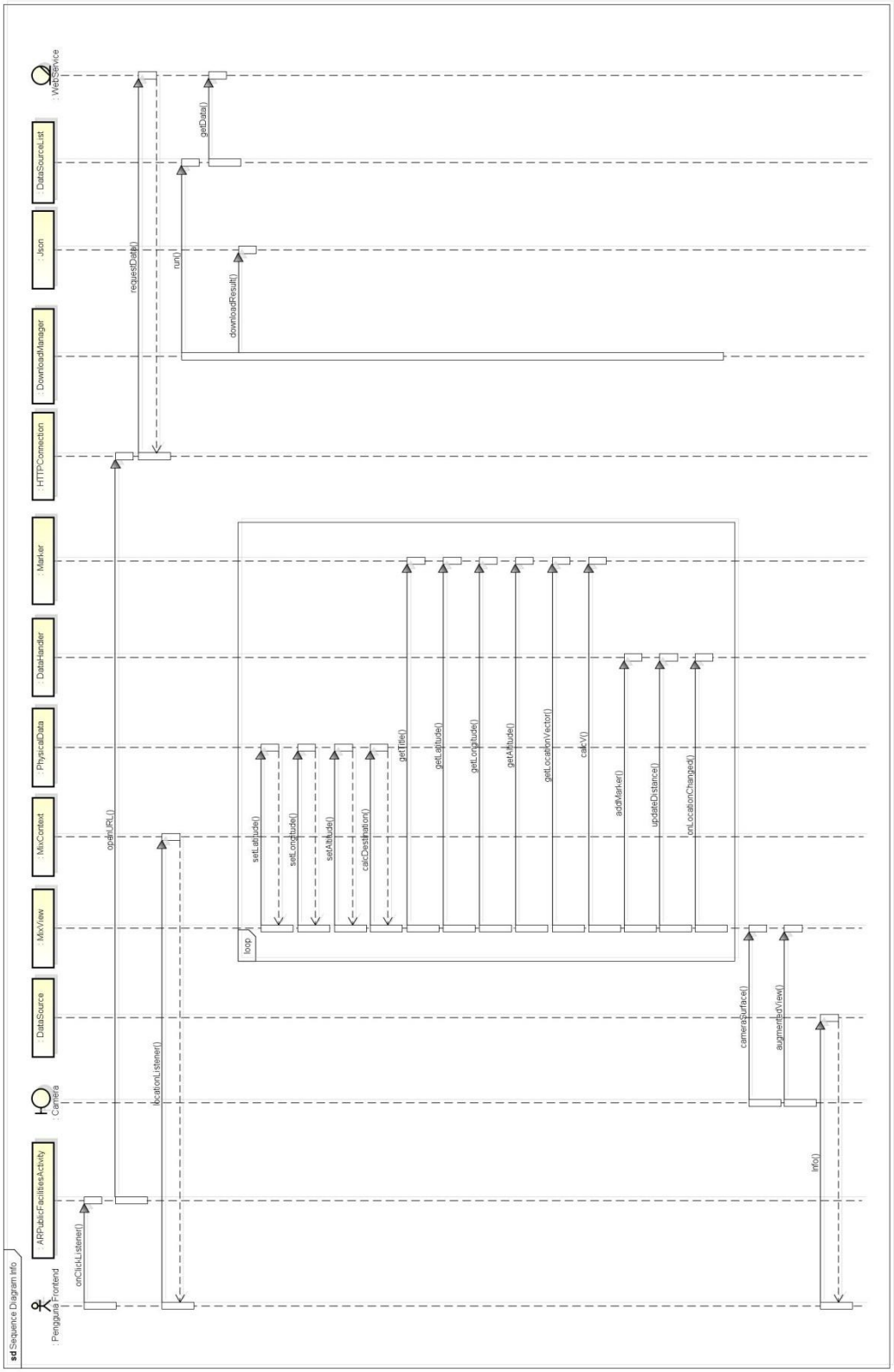
Gambar III.30 *Sequence Diagram* Daftar Konten (*List Vew*)

4) *Sequence Diagram* Jarak Radius (*Zoom Level*)



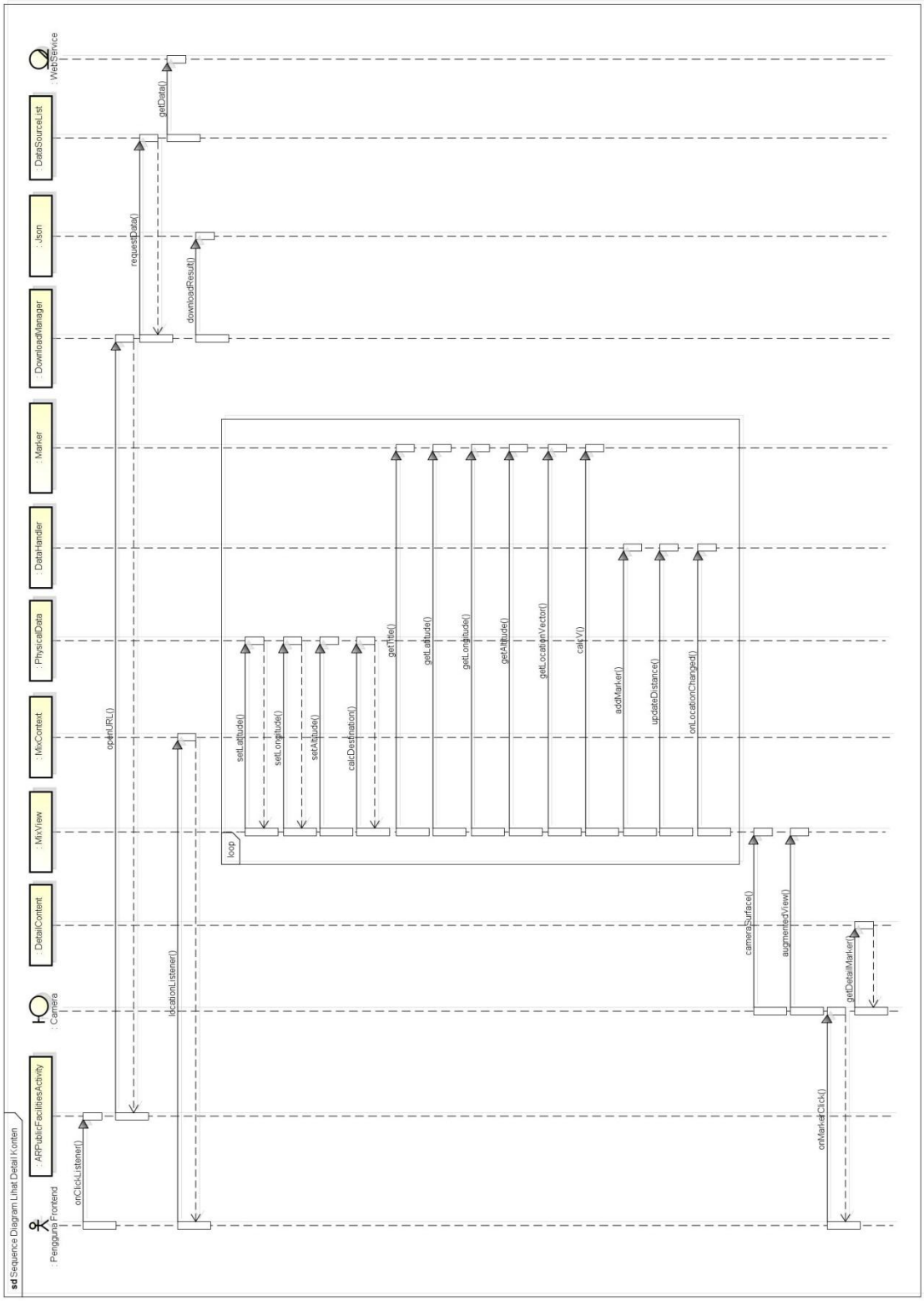
Gambar III.31 *Sequence Diagram* Jarak Radius (*Zoom Level*)

5) *Sequence Diagram* Info



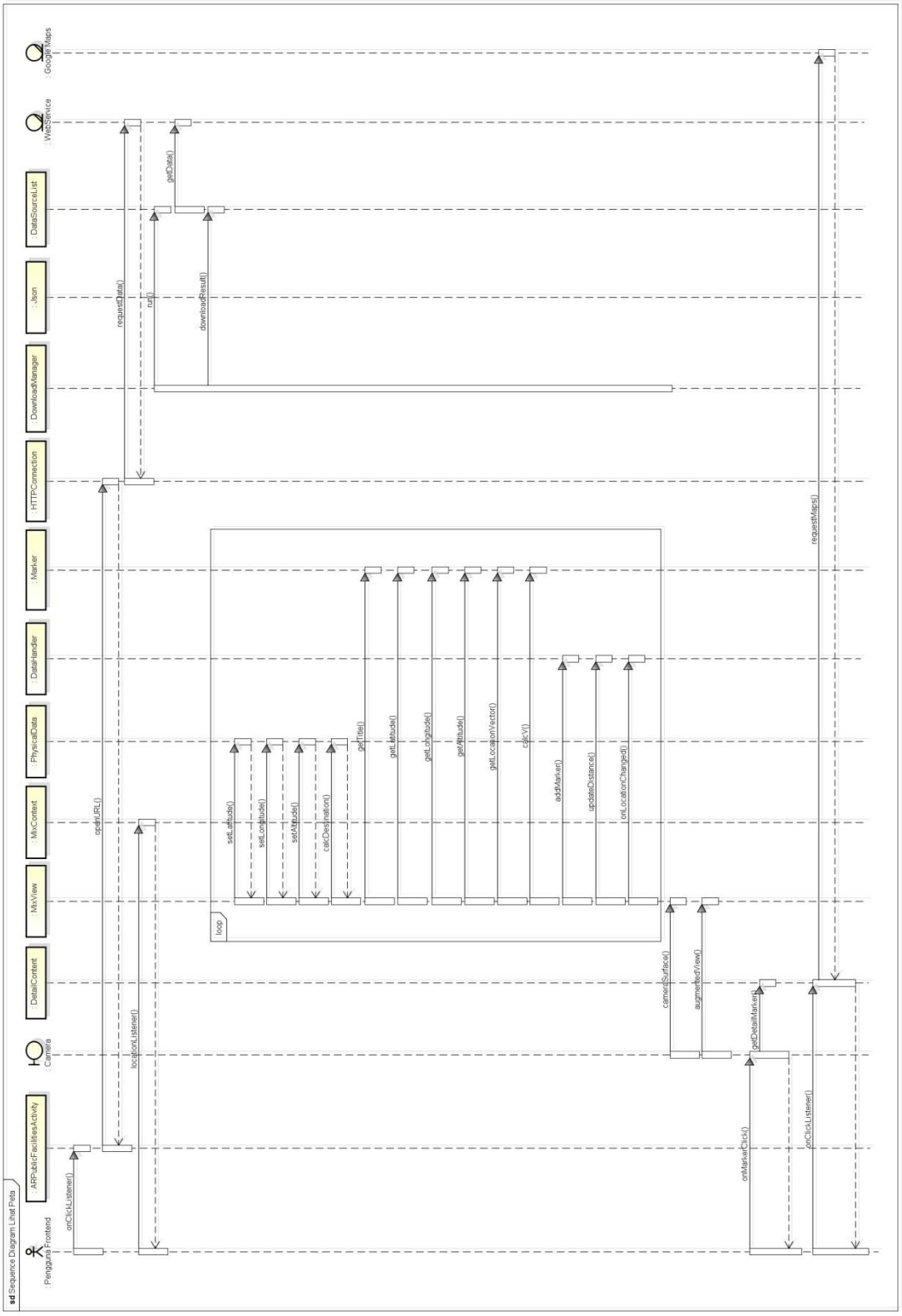
Gambar III.32 *Sequence Diagram* Info

6) *Sequence Diagram* Lihat Detail Konten



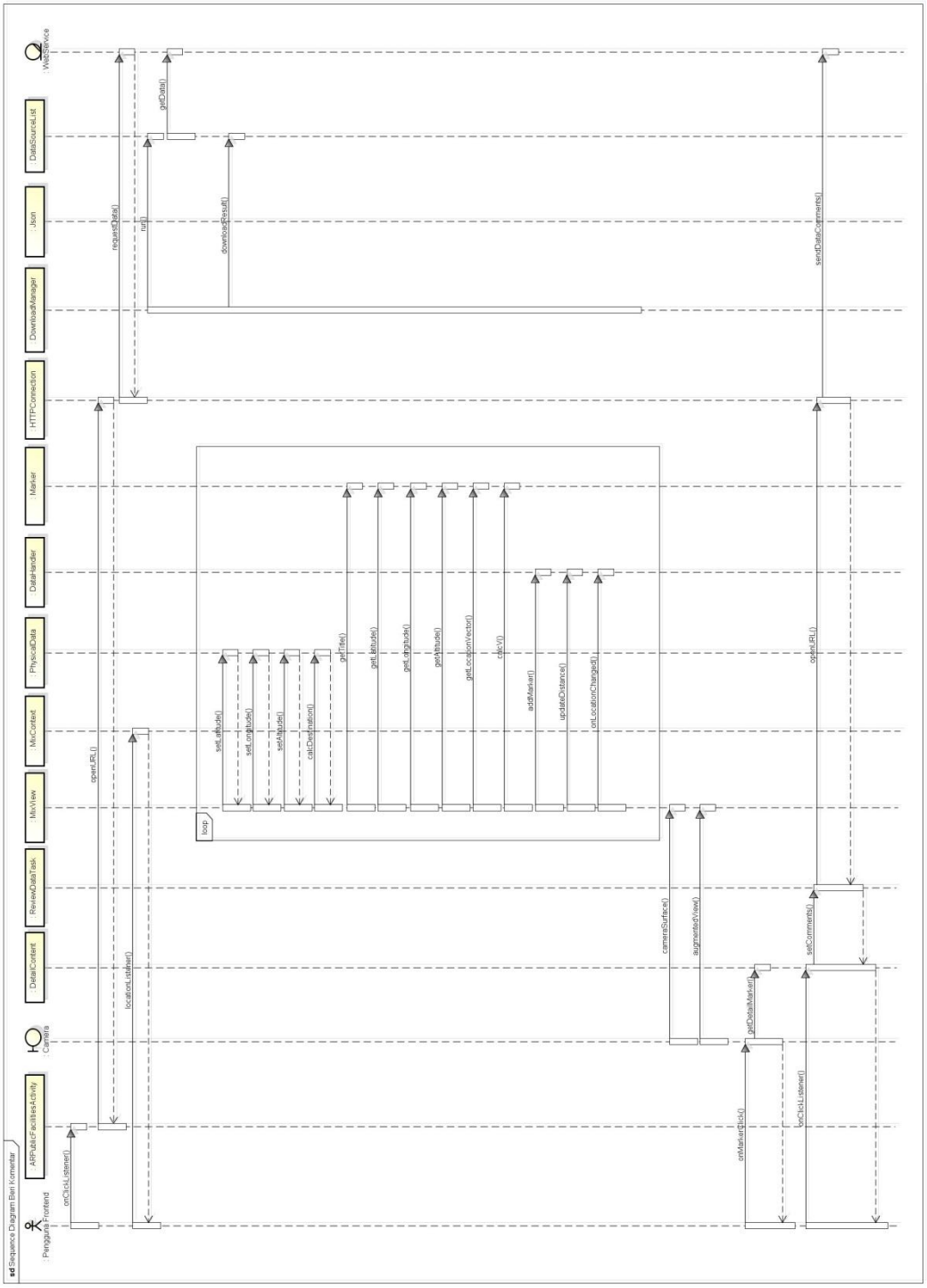
Gambar III.33 *Sequence Diagram* Lihat Detail Konten

7) *Sequence Diagram* Lihat Peta



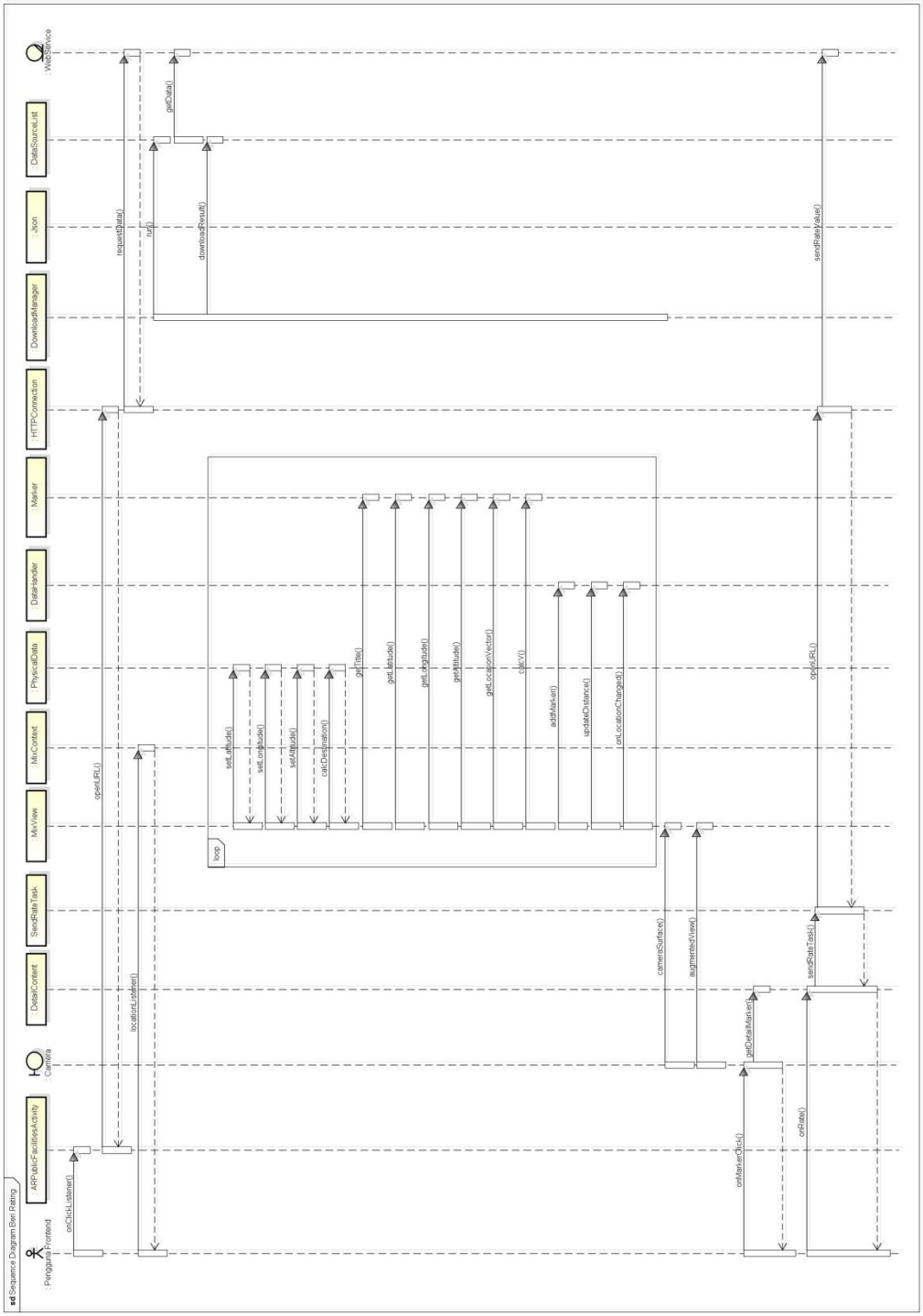
Gambar III.34 *Sequence Diagram* Lihat Peta

8) *Sequence Diagram* Beri Komentar



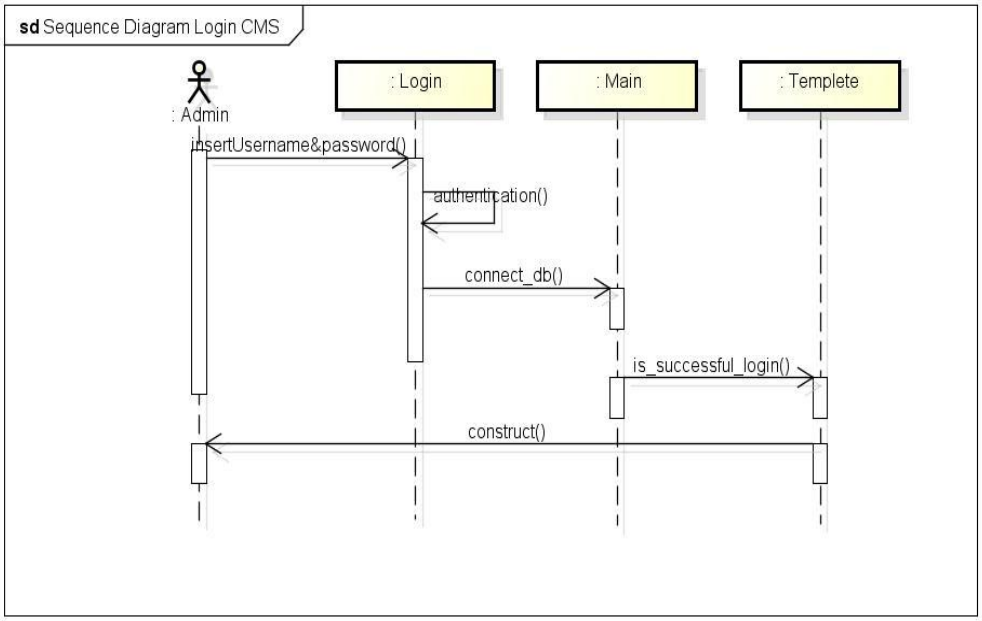
Gambar III.35 *Sequence Diagram* Beri Komentar

9) *Sequence Diagram* Beri *Rating*



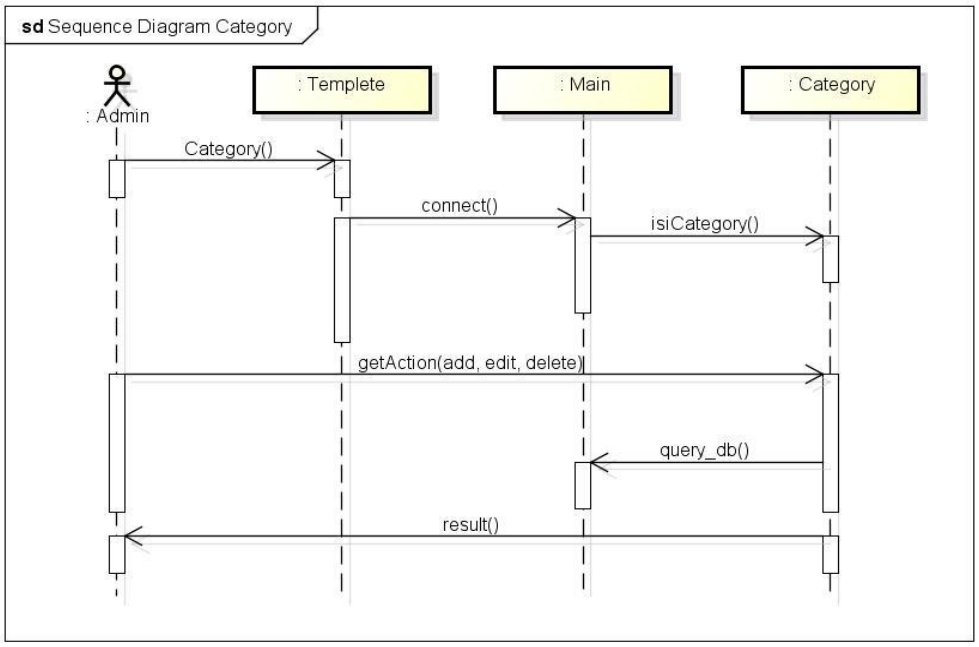
Gambar III.36 *Sequence Diagram* Beri *Rating*

10) *Sequence Diagram Login* CMS



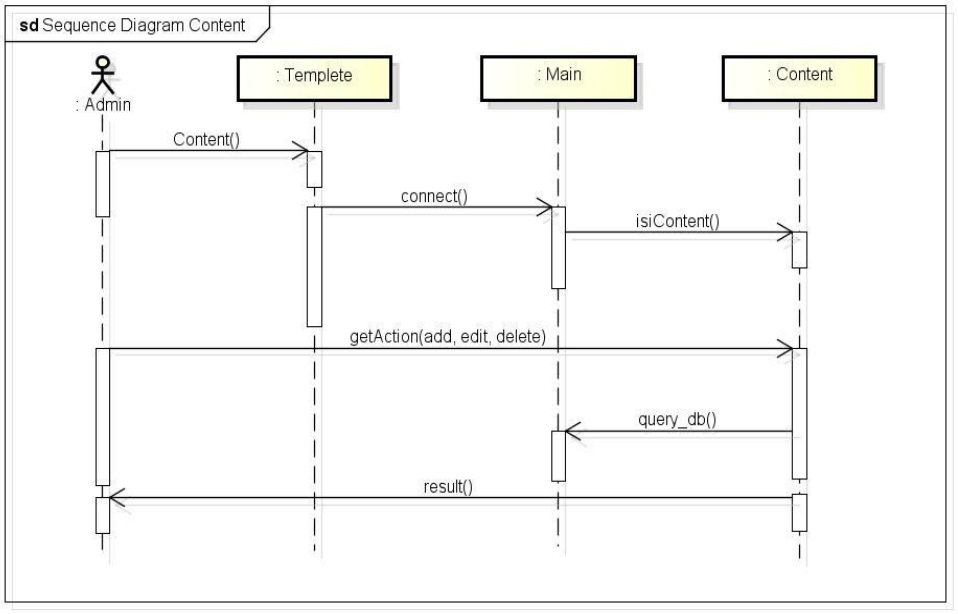
Gambar III.37 *Sequence Diagram Login CMS*

11) *Sequence Diagram* Kategori



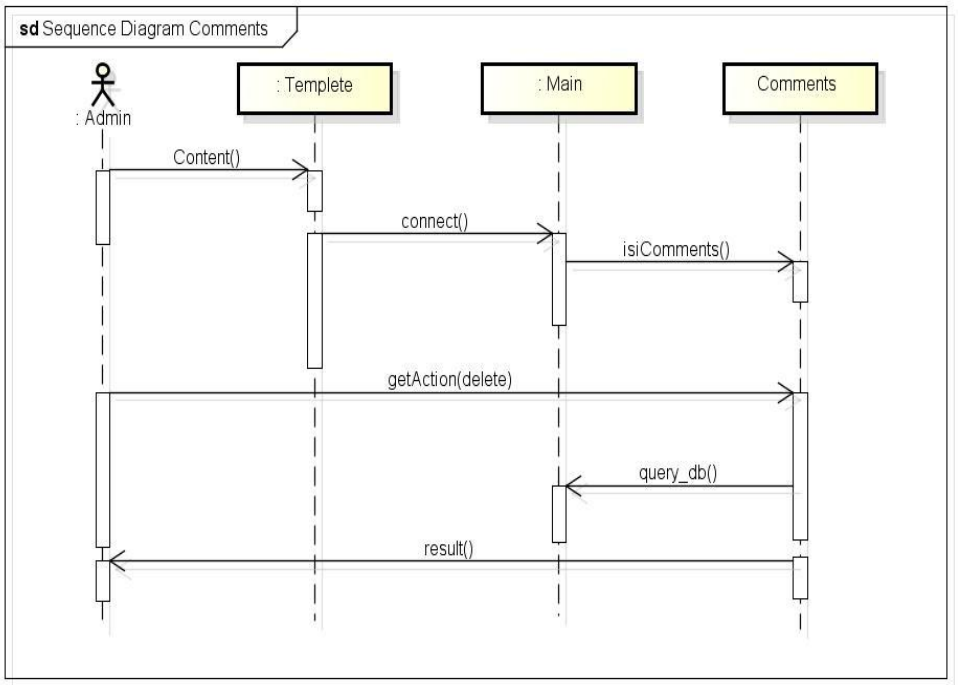
Gambar III.38 *Sequence Diagram* Kategori

12) *Sequence Diagram* Konten



Gambar III.39 *Sequence Diagram* Konten

13) *Sequence Diagram* Komentar

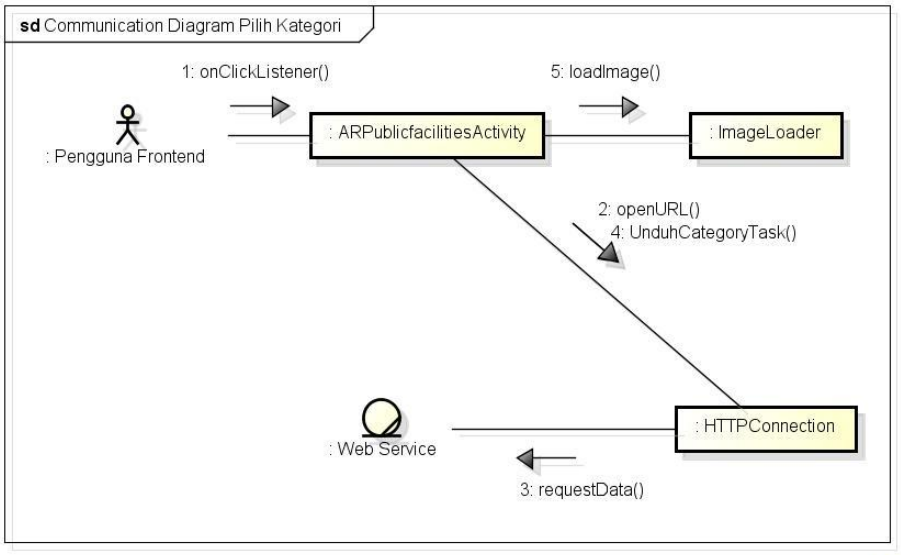


Gambar III.40 *Sequence Diagram* Komentar

6 *Collaboration Diagram*

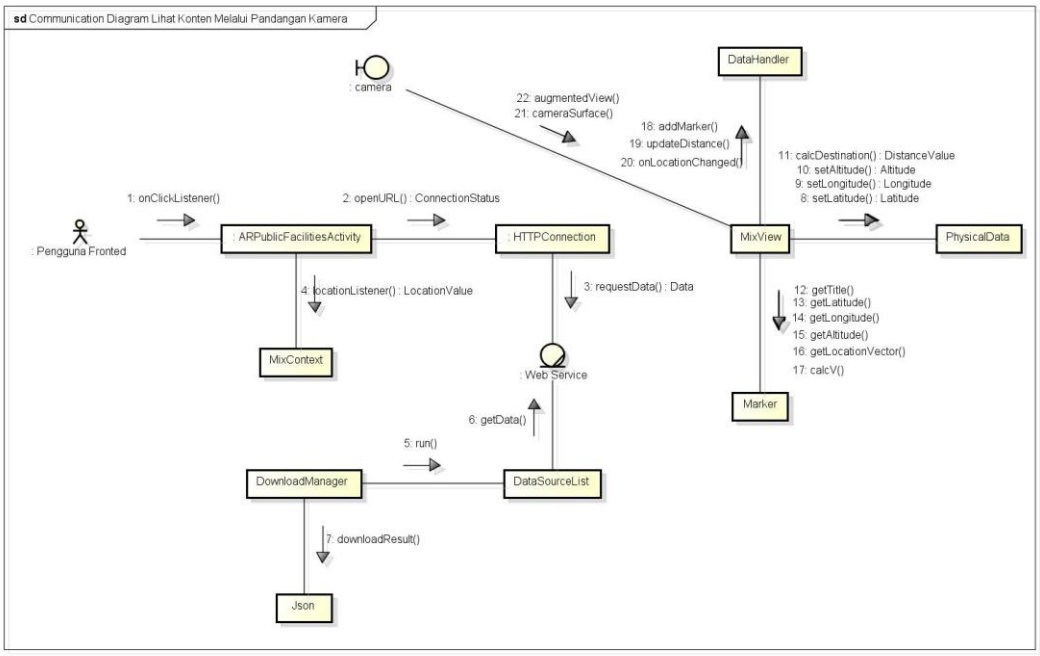
Collaboration diagram adalah diagram yang menggambarkan interaksi antar objek seperti *sequence diagram*, tetapi lebih menekankan pada peran masing-masing objek dan bukan pada waktu penyampaian *message*. Setiap *message* memiliki *sequence number*, dimana message dari level tertinggi memiliki nomor 1. *Message* dari level yang sama memiliki prefiks yang sama.

1) *Collaboration Diagram* Pilih Kategori



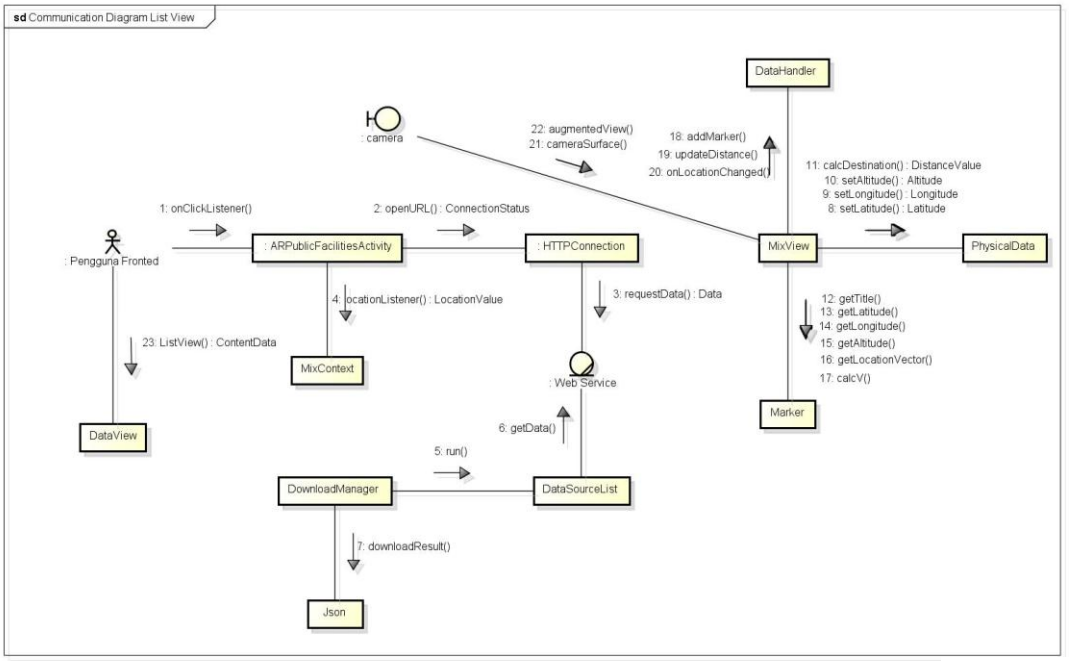
Gambar III.41 *Collaboration Diagram* Pilih Kategori

2) Collaboration Diagram Lihat Konten Melalui Pandangan Kamera



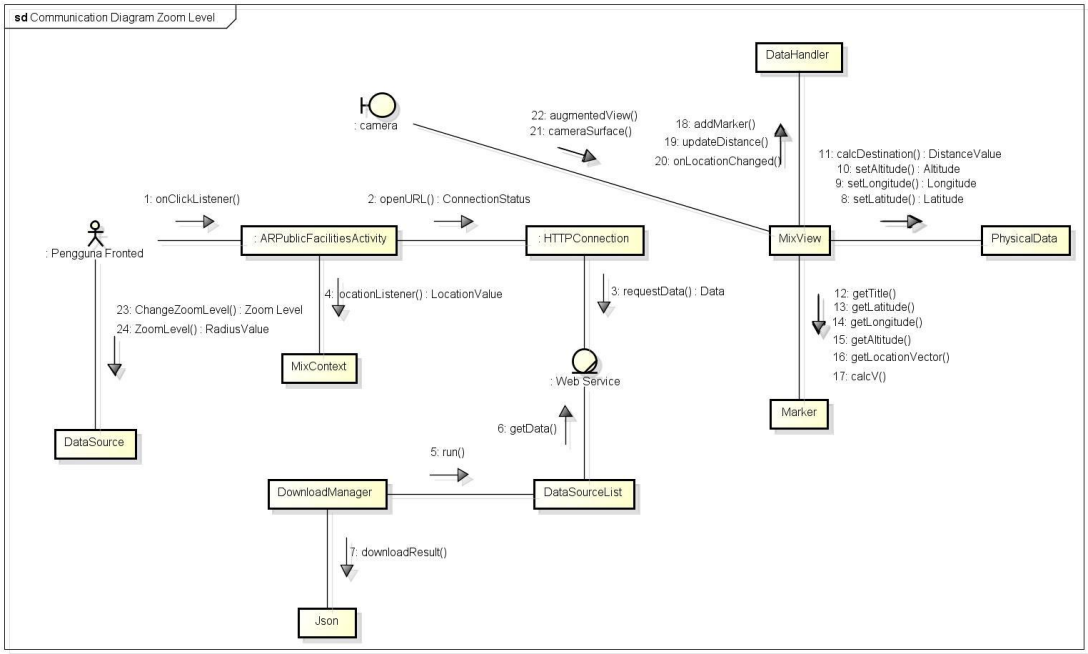
Gambar III.42 *Collaboration Diagram* Lihat Konten Melalui Pandangan Kamera

3) Collaboration Diagram Daftar Konten (*List View*)



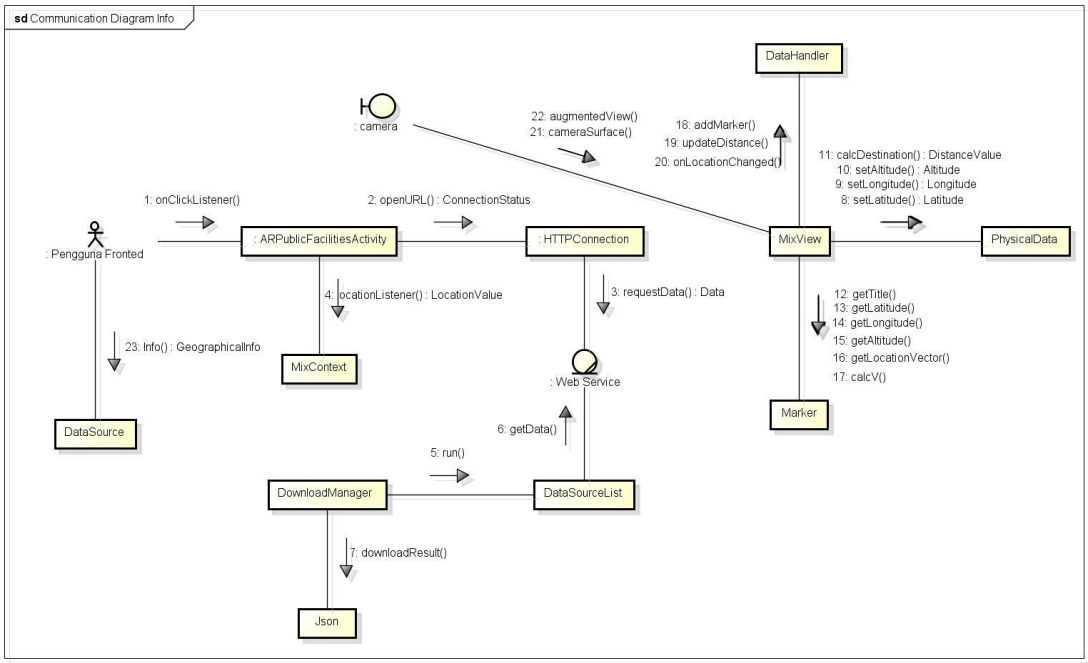
Gambar III.43 *Collaboration Diagram* Daftar Konten (*List View*)

4) Collaboration Zoom Level



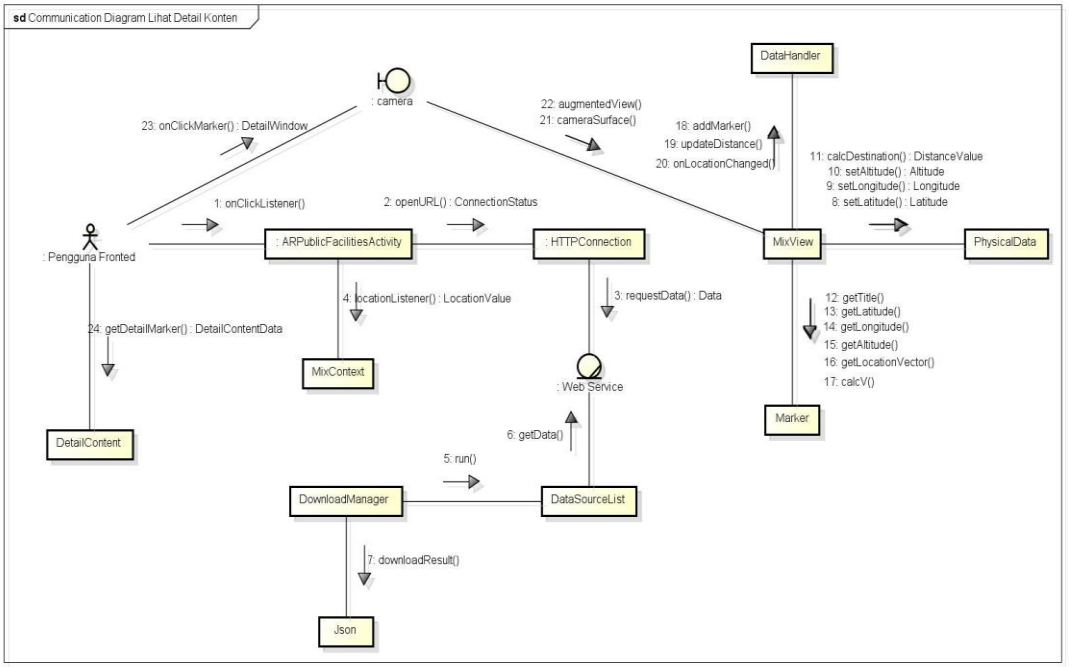
Gambar III.44 *Collaboration Diagram Zoom Level*

5) Collaboration Diagram Info



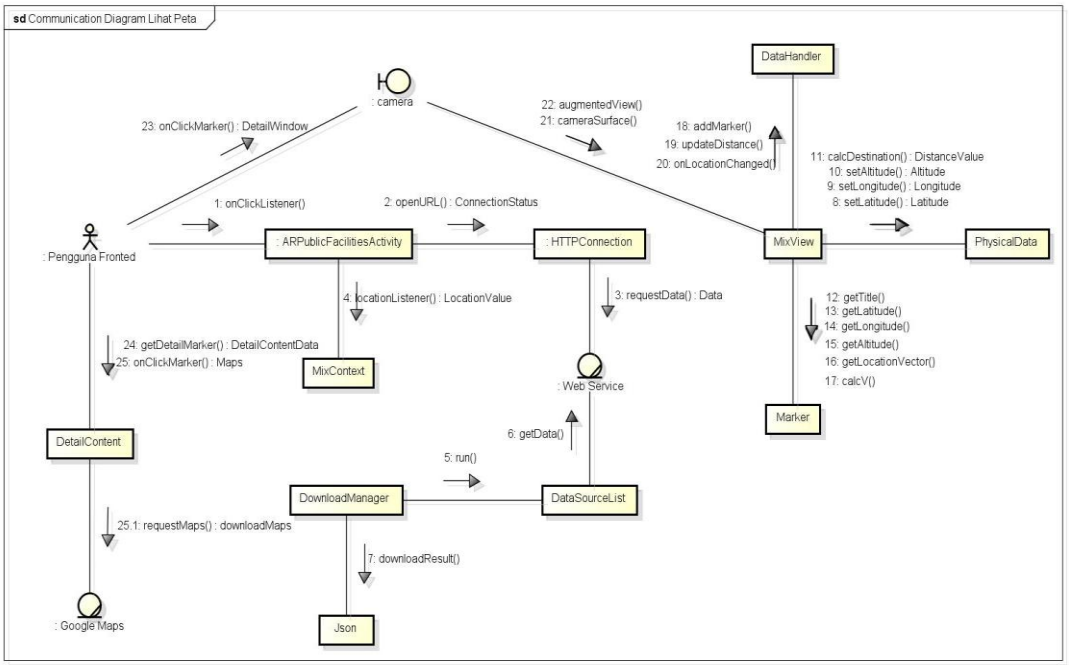
Gambar III.45 *Collaboration Diagram Info*

6) Collaboration Diagram Lihat Detail Konten



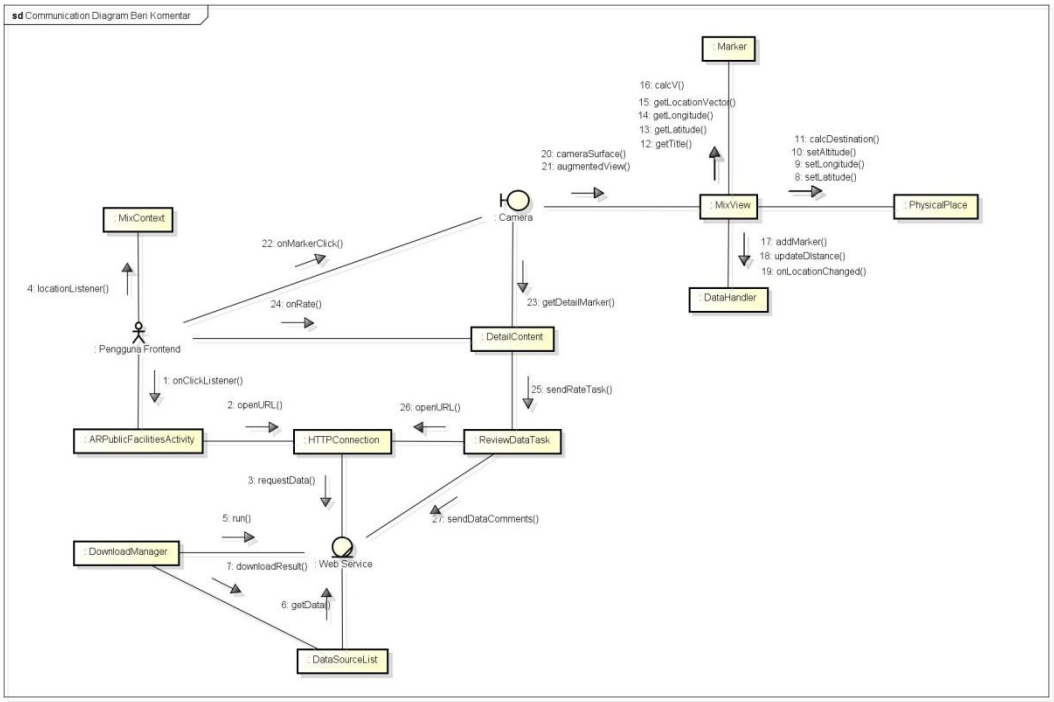
Gambar III.46 *Collaboration Diagram* Lihat Detail Konten

7) Collaboration Diagram Lihat Peta



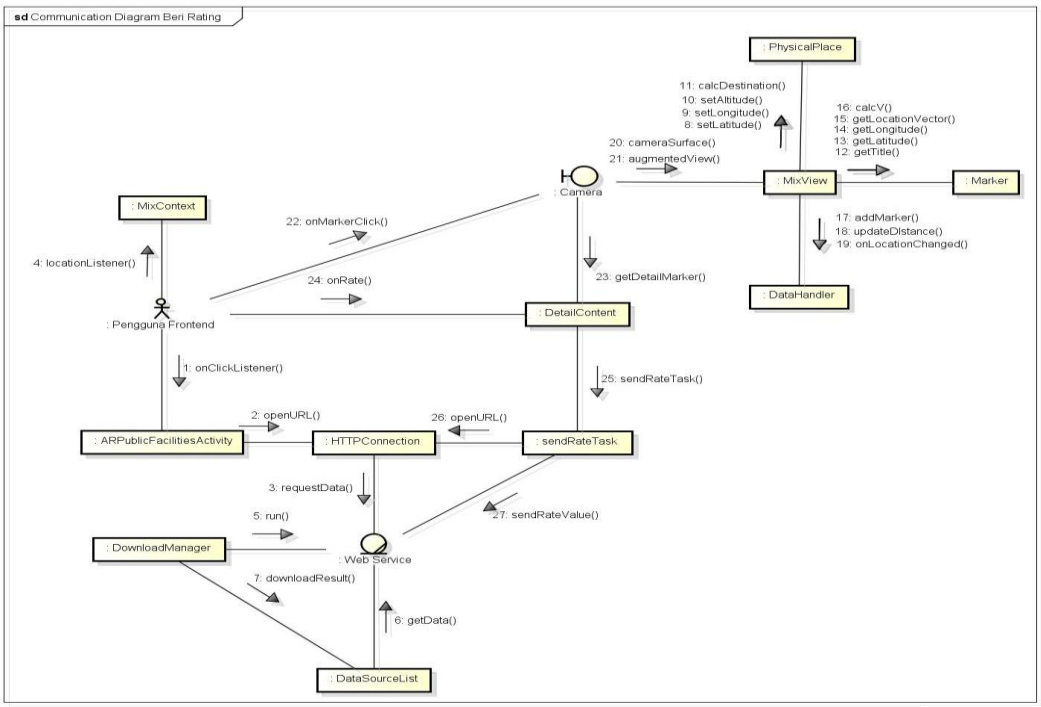
Gambar III.47 *Collaboration Diagram* Lihat Peta

8) *Collaboration Diagram* Beri Komentar



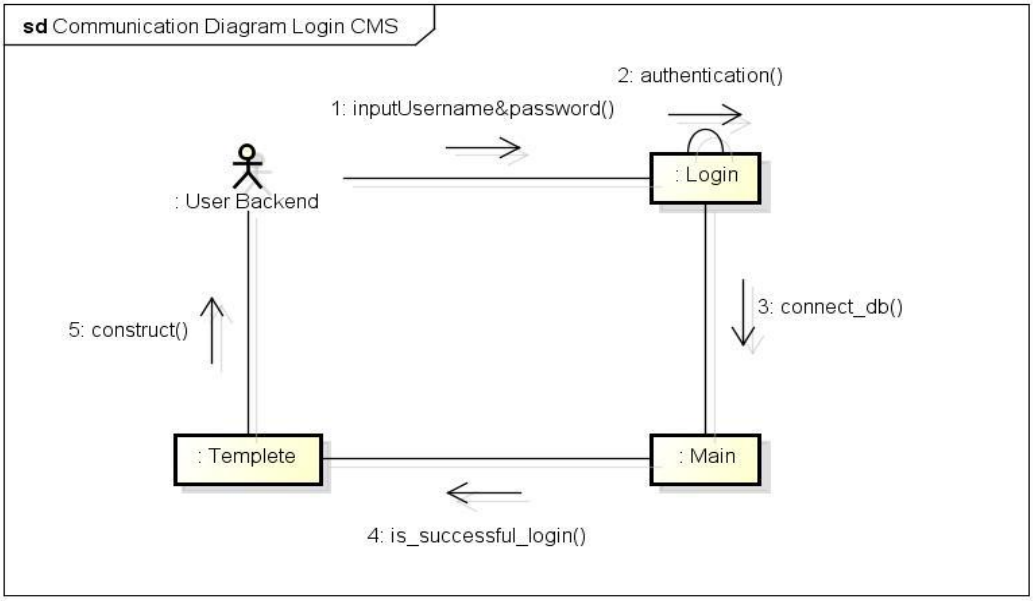
Gambar III.48 *Collaboration Diagram* Beri Komentar

9) *Collaboration Diagram* Beri *Rating*



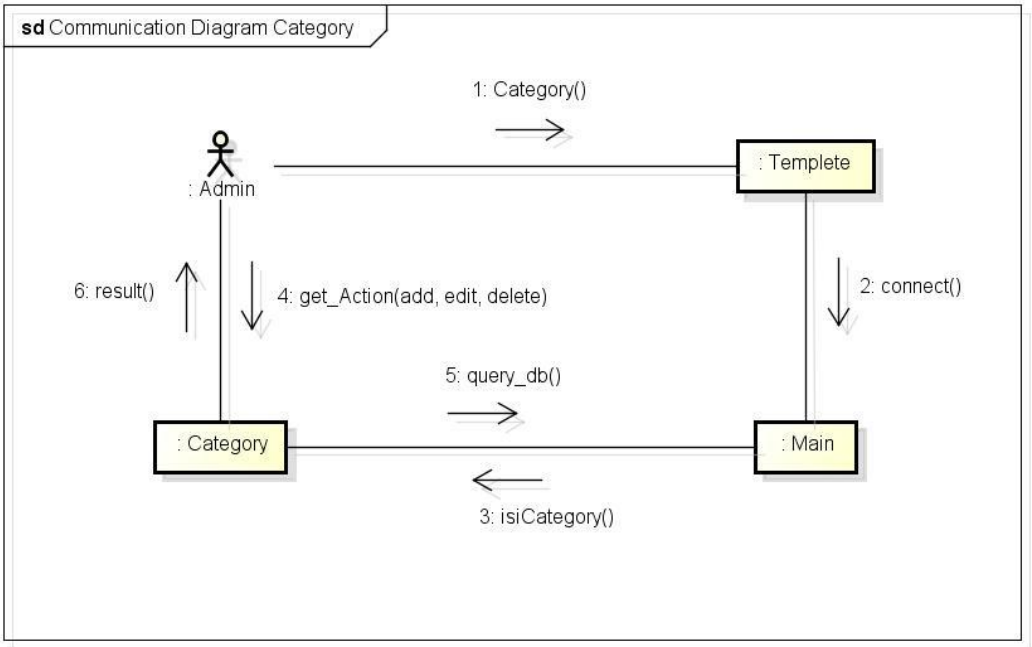
Gambar III. 49 *Collaboration Diagram* Beri *Rating*

10) *Collaboration Diagram* Login CMS



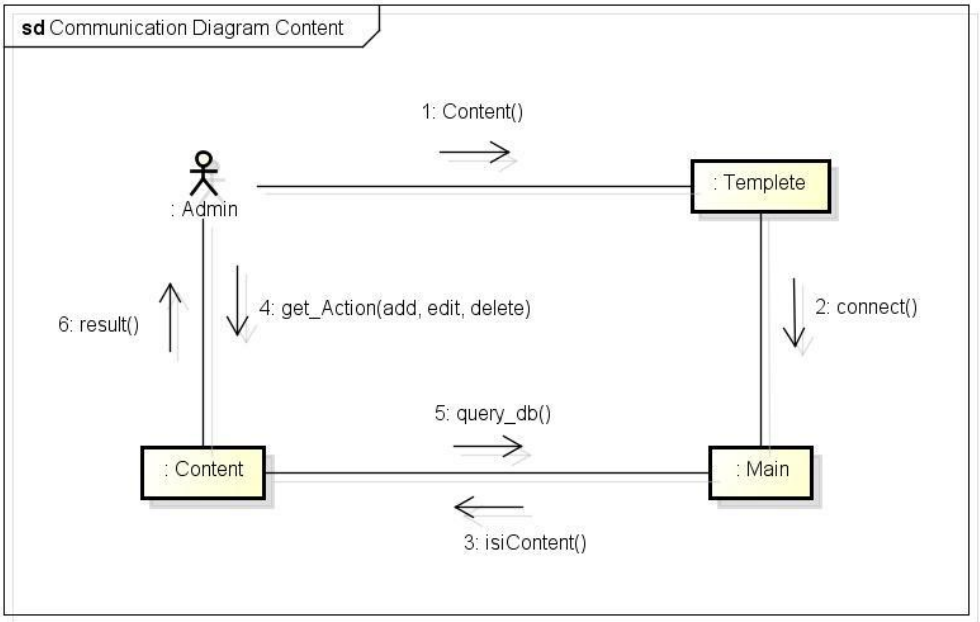
Gambar III.48 *Collaboration Diagram* Login CMS

11) *Collaboration Diagram* Kategori



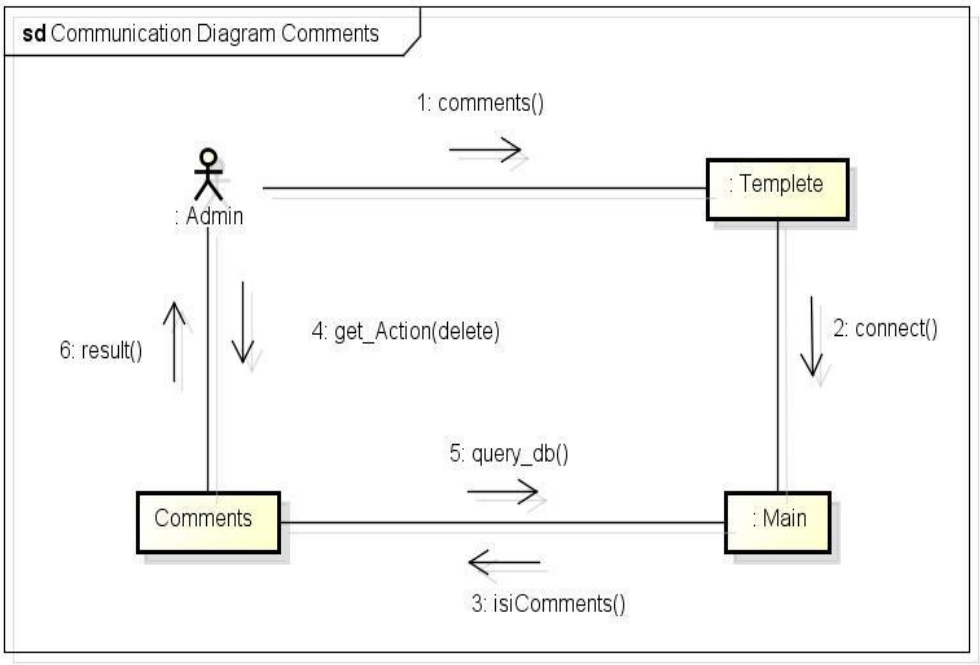
Gambar III.49 *Collaboration Diagram* Kategori

12) *Collaboration Diagram* Konten



Gambar III.50 *Collaboration Diagram* Konten

13) *Collaboration Diagram* Komentar

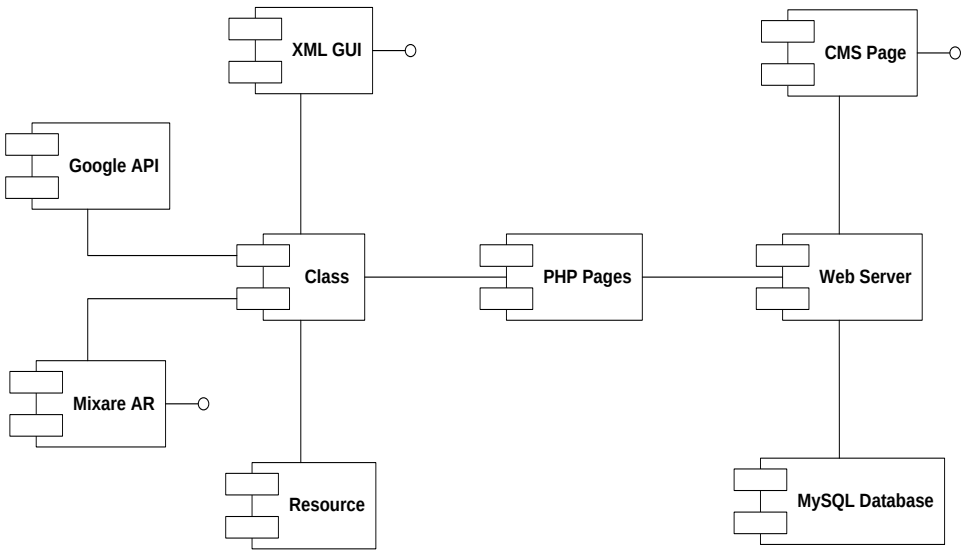


Gambar III.51 *Collaboration Diagram* Komentar

7 *Component Diagram*

Component diagram menggambarkan struktur dan hubungan antar komponen piranti lunak, termasuk ketergantungan (*dependency*) diantaranya. Komponen piranti lunak adalah modul berisi *code*, baik berisi *source code* maupun *binary code*, baik *library* maupun *executable*, baik yang muncul pada *compile time*, *link time*, maupun *run time*. Umumnya komponen terbentuk dari beberapa *class* dan/atau *package*, tapi dapat juga dari komponen-komponen yang lebih kecil.

Komponen dapat juga berupa *interface*, yaitu kompulan layanan yang disediakan sebuah komponen untuk komponen lain.



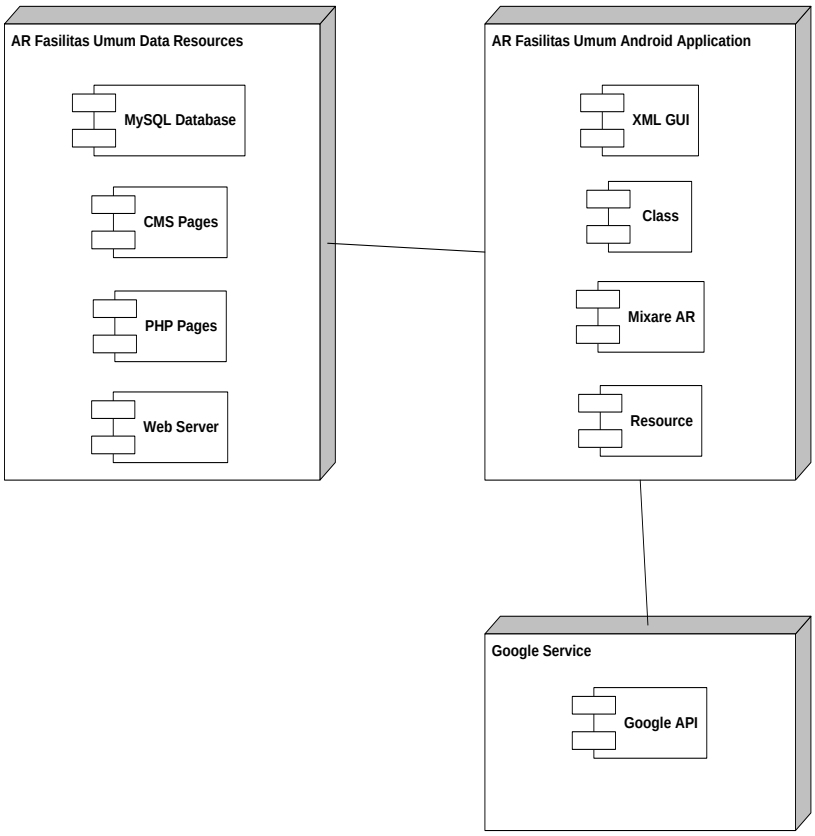
Gambar III.52 *Component Diagram*

8 *Deployment Diagram*

Deployment diagram menggambarkan detail bagaimana komponen di-*deploy* dalam infrastruktur sistem, di mana komponen akan terletak (pada mesin, server atau piranti keras apa), bagaimana kemampuan

jaringan pada lokasi tersebut, spesifikasi server, dan hal-hal lain yang bersifat fisikal.

Sebuah *node* adalah server, *workstation* , atau piranti keras lain yang digunakan untuk men-*deploy* komponen dalam lingkungan sebenarnya. Hubungan antar *node* (misalnya TCP/IP) dan *requirement* dapat juga didefinisikan dalam diagram ini.



Gambar III.53 *Deployment Diagram*

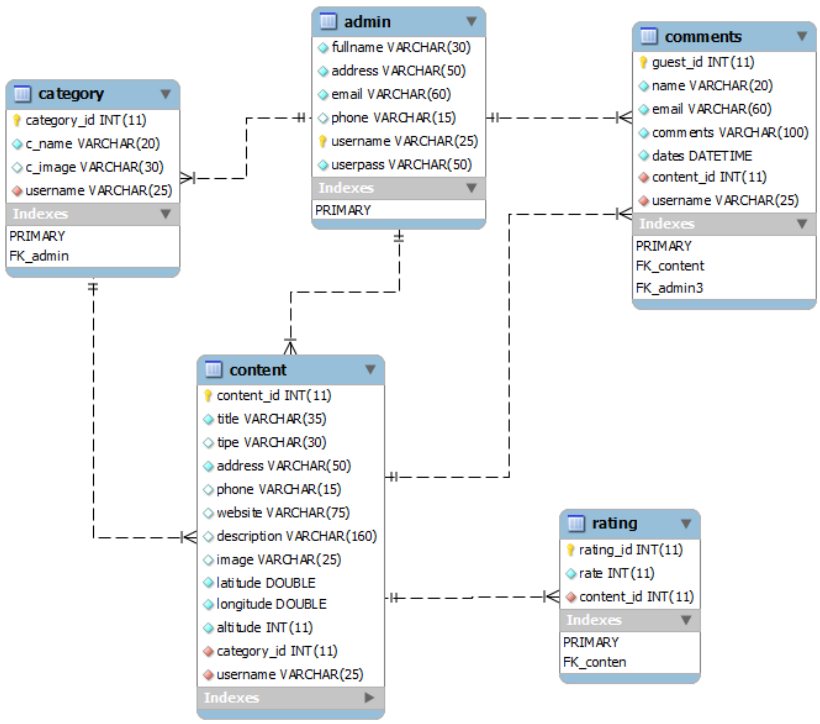
3.2 Perancangan Sistem

Perancangan sistem bertujuan untuk menspesifikasikan aspek-aspek teknik yang menjadi solusi dalam perancangan. Pada tahap ini perancangan akan didefinisikan secara detail untuk mengatasi masalah-masalah yang lebih teknik, berkaitan dengan kegiatan implementasi seperti perancangan arsitektur sistem, perancangan basis data, dan perancangan antarmuka.

3.2.1. Skema Relasi

Skema relasi atau relasi antar tabel merupakan gabungan antar file yang mempunyai kunci utama yang sama, sehingga file-file tersebut menjadi satu kesatuan yang dihubungkan oleh field kunci (*primary key*). Pada proses ini elemen-elemen data dikelompokkan menjadi satu file database beserta entitas dan hubungan.

Berikut adalah gambar skema relasi :



Gambar III.54 Skema Relasi Aplikasi *ARPublicFacilities*

3.2.2. Struktur Tabel

1) Tabel Admin

Tabel III.30 Struktur Tabel Admin

Nama Field	Tipe Data	Panjang	Keterangan
fullname	Varchar	30	
address	Varchar	50	
email	Varchar	60	
phone	Varchar	15	
username	Varchar	25	Primary Key
userpass	Varchar	50	

2) Tabel *Category*

Tabel III.31 Struktur Tabel *Category*

Nama Field	Tipe Data	Panjang	Keterangan
category_id	Integer	11	Primary Key
c_name	Varchar	20	
c_image	Varchar	30	
username	Varchar	25	Foreign Key references Admin(username)

3) Tabel *Content*

Tabel III.32 Struktur Tabel *Content*

Nama Field	Tipe Data	Panjang	Keterangan
content_id	Integer	11	Primary Key
title	Varchar	35	
tipe	Varchar	30	
address	Varchar	50	
phone	Varchar	15	
website	Varchar	75	
description	Varchar	160	
image	Varchar	20	
latitude	Double		
longitude	Double		
altitude	Integer	11	
category_id	Integer	11	Foreign Key references Category(category_id)
username	Varchar	25	Foreign Key references Admin(username)

4) Tabel *Comments*

Tabel III.33 Struktur Tabel *Comments*

Nama Field	Tipe Data	Panjang	Keterangan
guests_id	Integer	11	Primary Key
name	Varchar	20	
email	Varchar	60	
comments	Varchar	100	
dates	Datetime		
content_id	Integer	11	Foreign Key references content(content_id)
username	Varchar	25	Foreign Key References Admin(username)

5) Tabel *Rating*

Tabel III.34 Struktur Tabel *Rating*

Nama Field	Tipe Data	Panjang	Keterangan
rate_id	Integer	11	Primary Key
rate	Integer	11	
content_id	Integer	11	Foreign Key references Content(content_id)

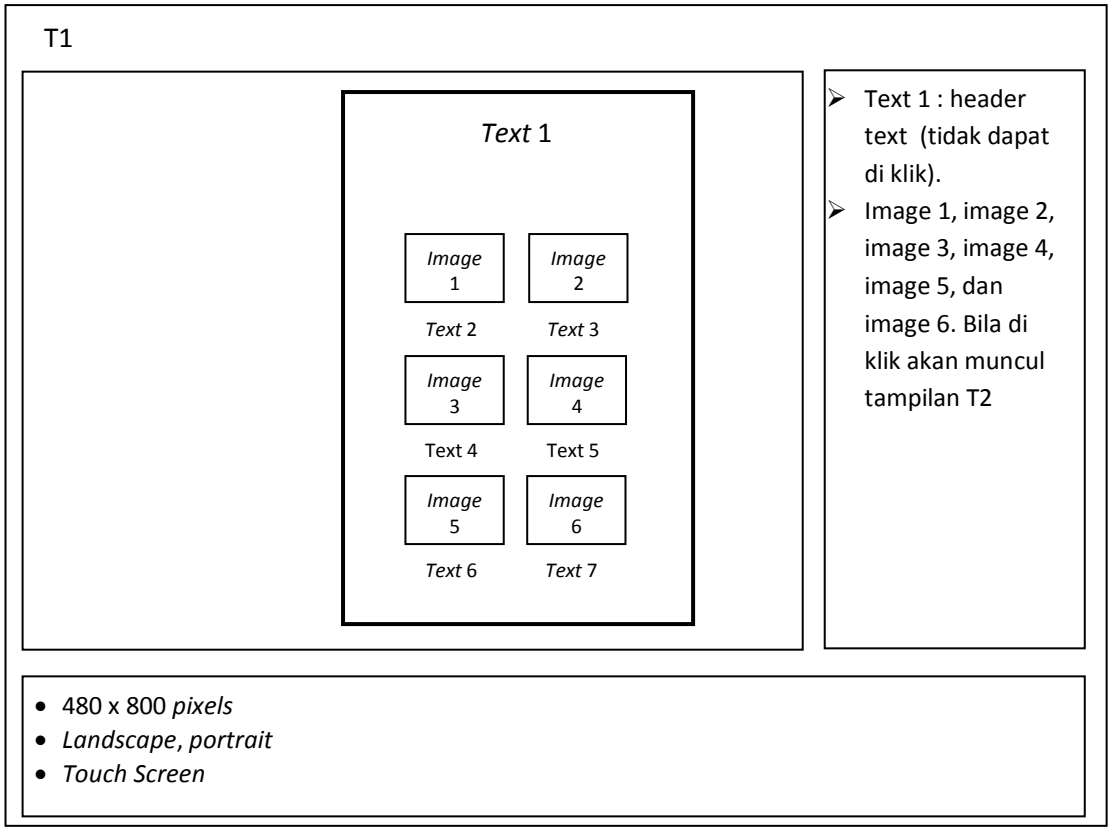
3.2.3. Perancangan Grafis dan Antarmuka

Tahap perancangan desain bertujuan untuk mencari bentuk yang optimal dari aplikasi yang akan dibangun dengan pertimbangan faktor-faktor permasalahan dan kebutuhan yang ada pada sistem seperti yang telah ditetapkan pada tahap analisis. Dalam tahap ini upaya yang dilakukan yaitu dengan cara mengkombinasikan penggunaan teknologi perangkat keras dan perangkat lunak yang tepat sehingga diperoleh desain yang optimal dan mudah di implementasikan.

Perancangan grafis dan antarmuka terdiri dari dua bagian utama yaitu desain grafis dan antarmuka aplikasi *frontend* dan *backend.*

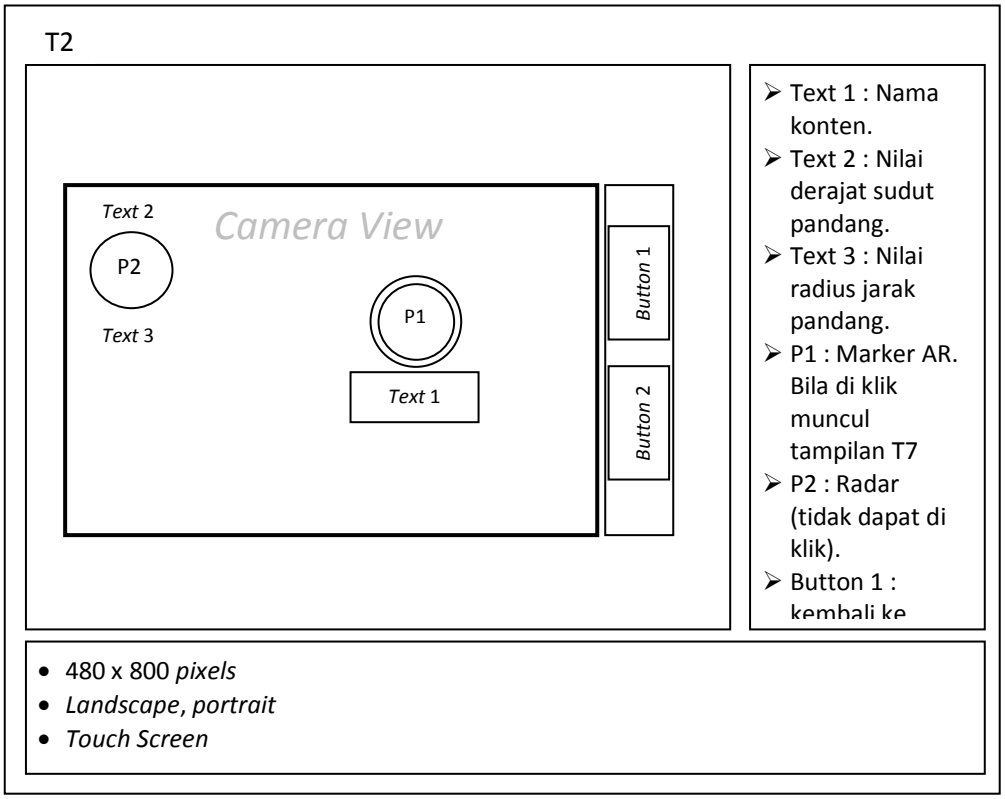
3.2.3.1 Perancangan Antarmuka Aplikasi *Frontend*

1) Perancangan Antarmuka Menu Utama / *Lihat Category*



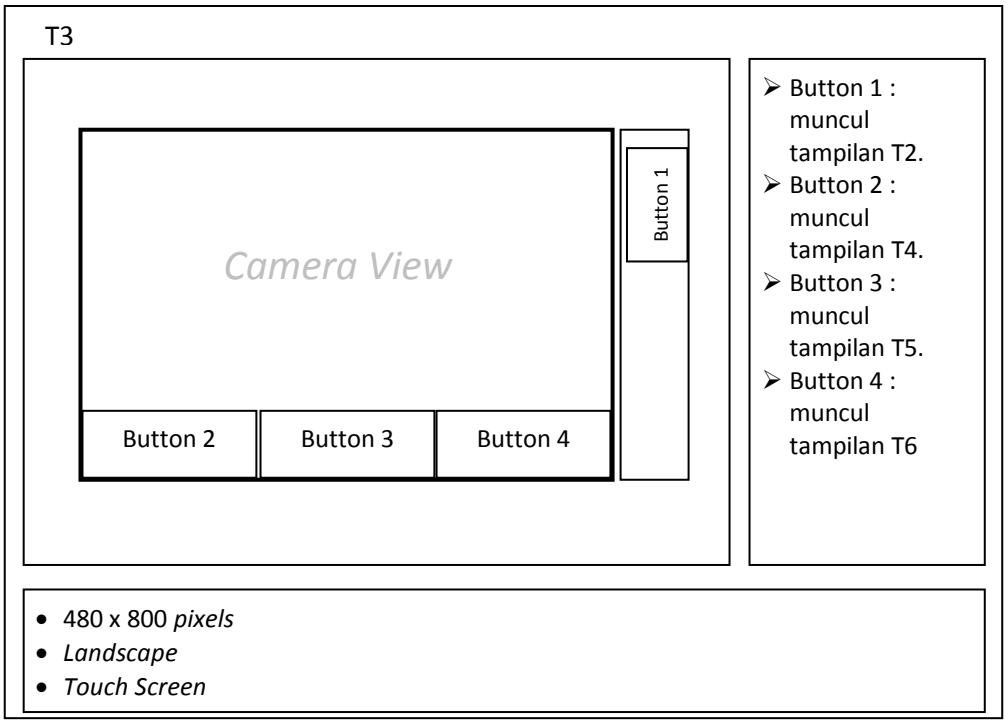
Gambar III.55 Perancangan Antarmuka Menu Utama (*Lihat Kategori*)

2) Perancangan Antarmuka Lihat Konten Melalui Pandangan Kamera



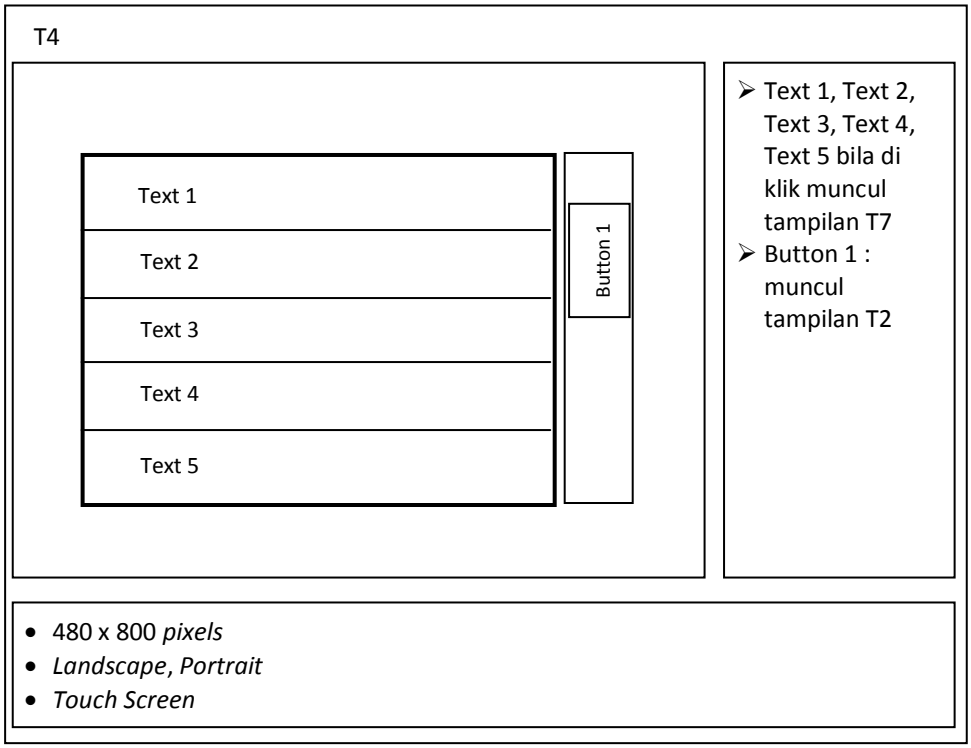
Gambar III.56 Perancangan Antarmuka Lihat Konten Melalui Pandangan Kamera

3) Perancangan Antarmuka Menu Kontrol



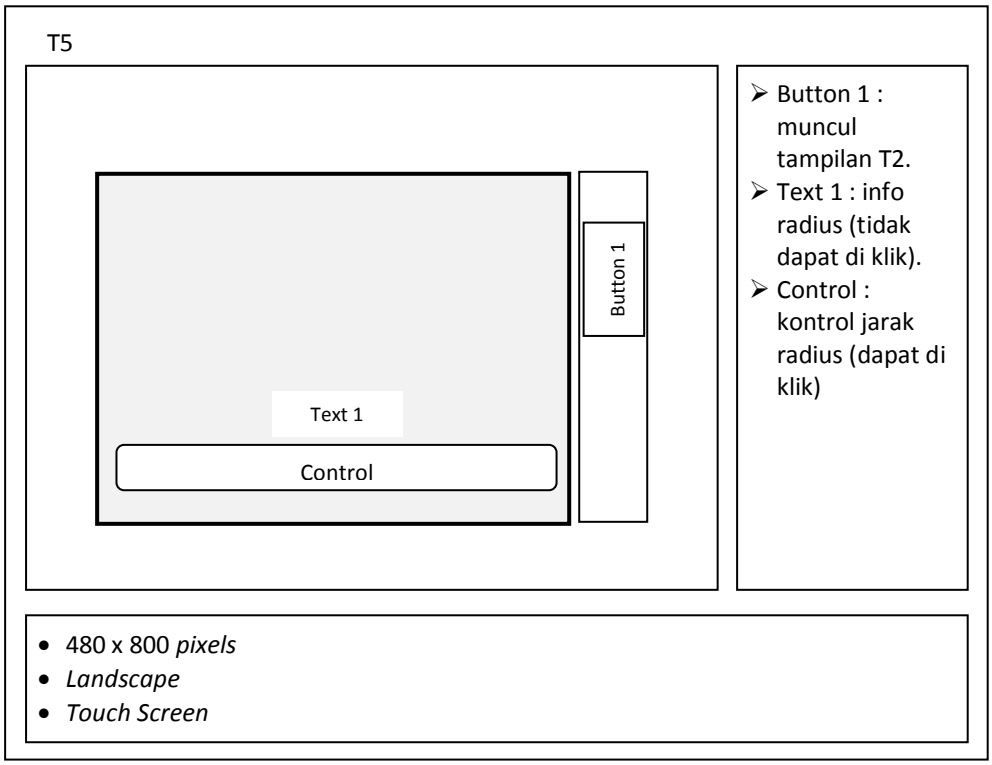
Gambar III.57 Perancangan Antarmuka Menu Kontrol

4) Perancangan Antarmuka Daftar Konten (*List View*)



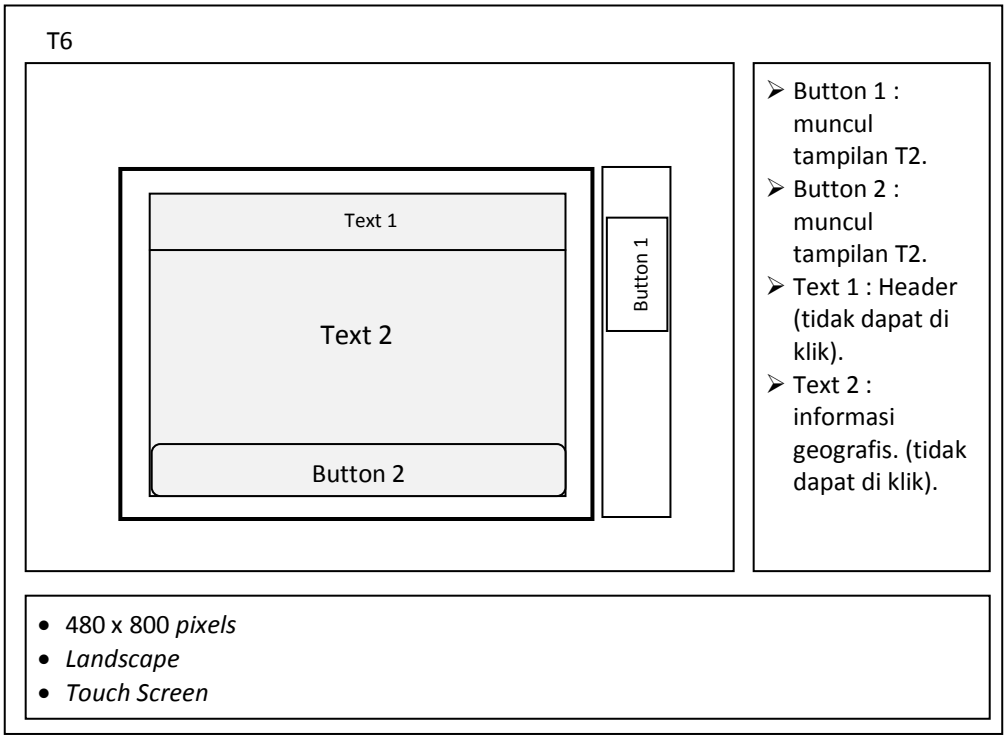
Gambar III.58 Perancangan Antarmuka Daftar Konten (*List View*)

5) Perancangan Antarmuka *Zoom Level*



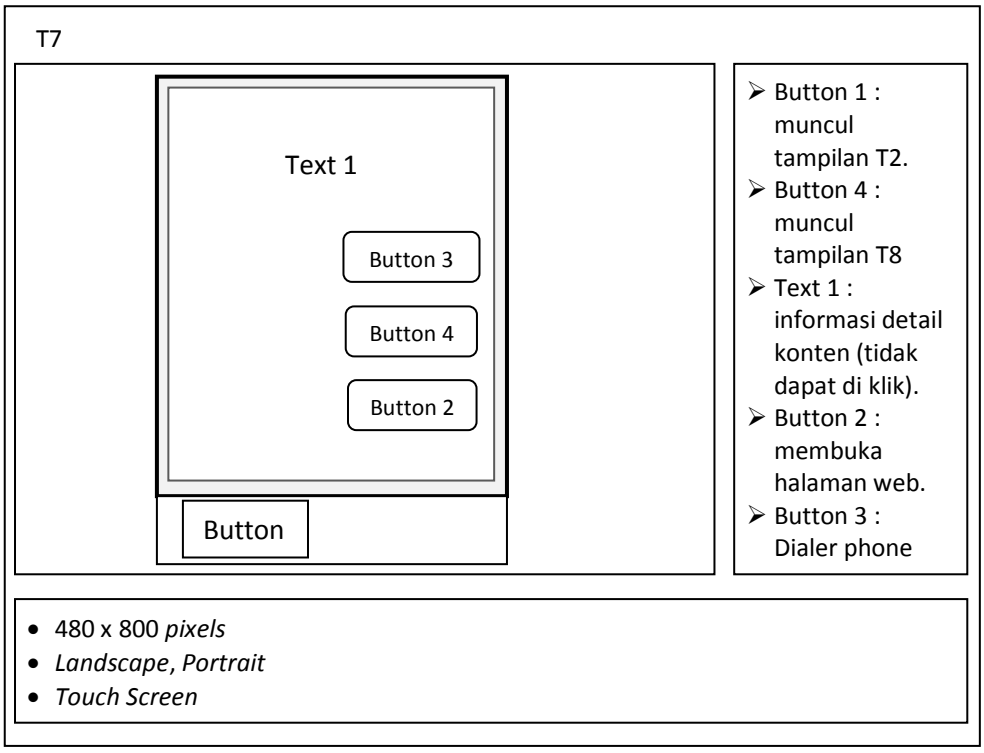
Gambar III.59 Perancangan Antarmuka *Zoom Level*

6) Perancangan Antarmuka Info



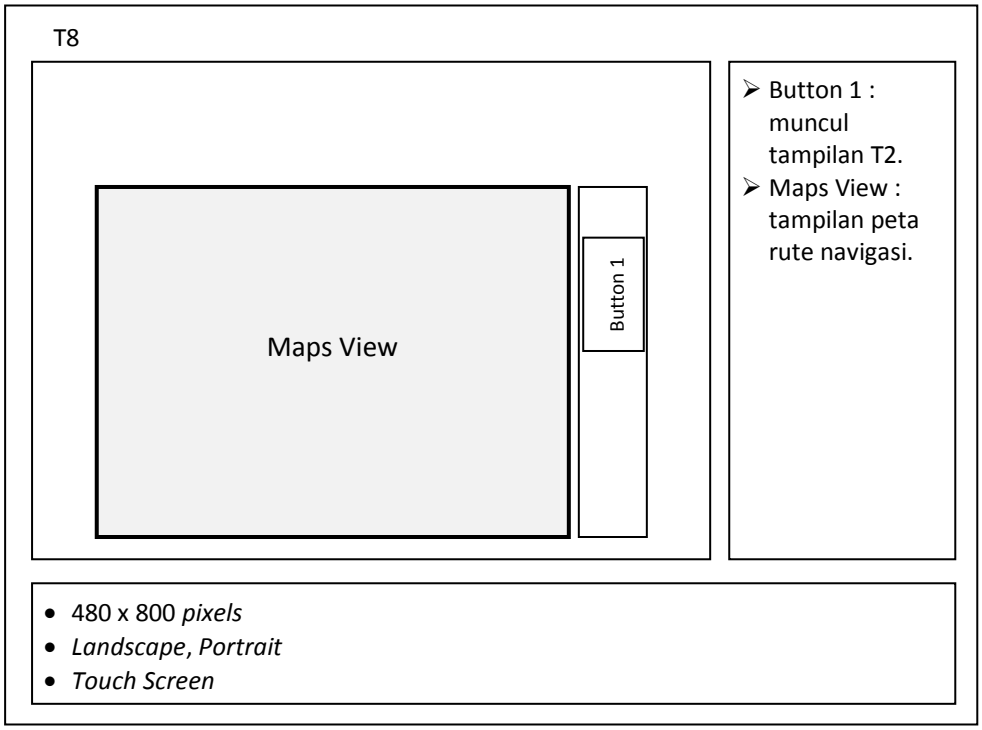
Gambar III.60 Perancangan Antarmuka Info

7) Perancangan Antarmuka Lihat Detail Konten



Gambar III.61 Perancangan Antarmuka Lihat Detail Konten

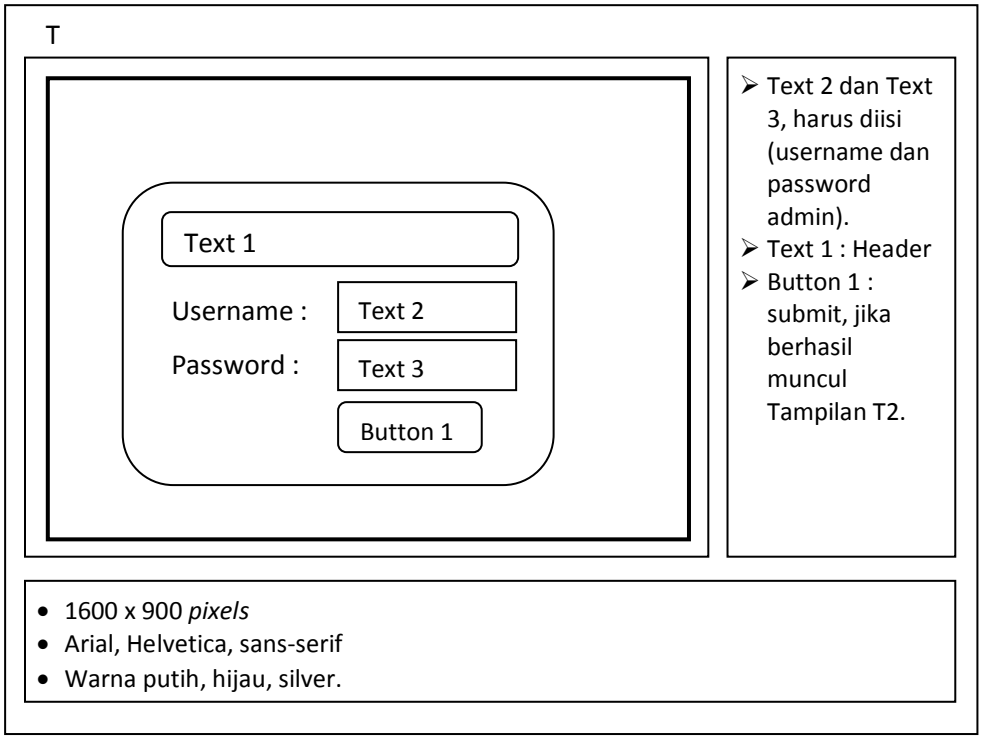
8) Perancangan Antarmuka Lihat Peta



Gambar III.62 Perancangan Antarmuka Lihat Peta

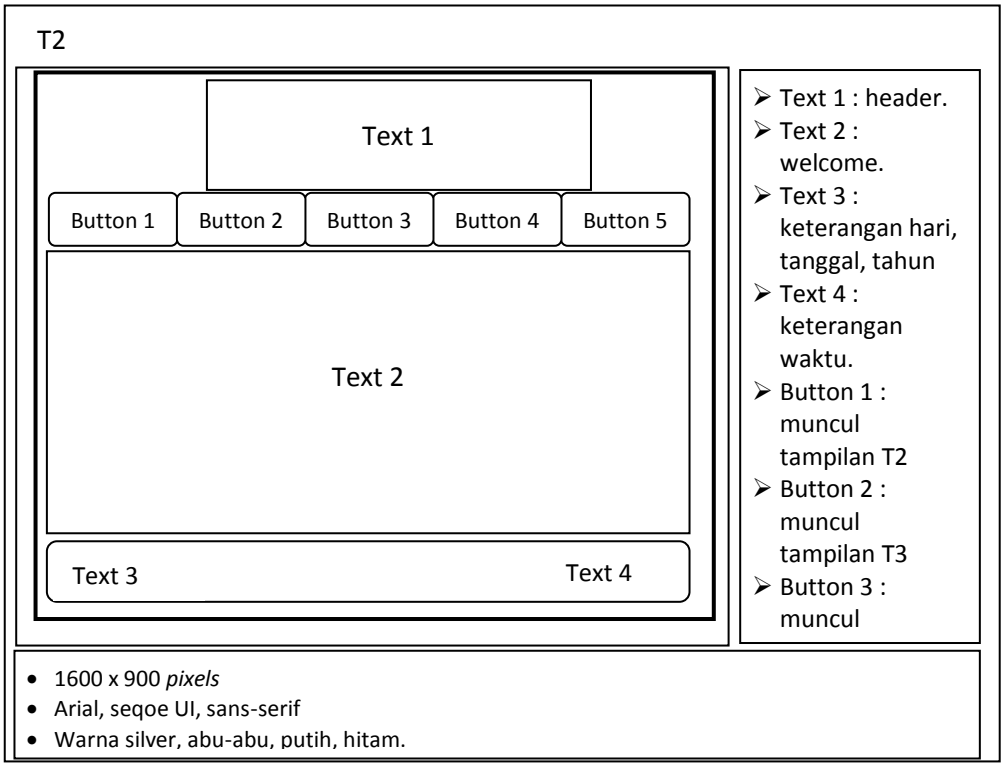
3.2.3.2 Perancangan Antarmuka Aplikasi *Backend*

1) Perancangan Antarmuka *Login* CMS *ARPublicfacilities*



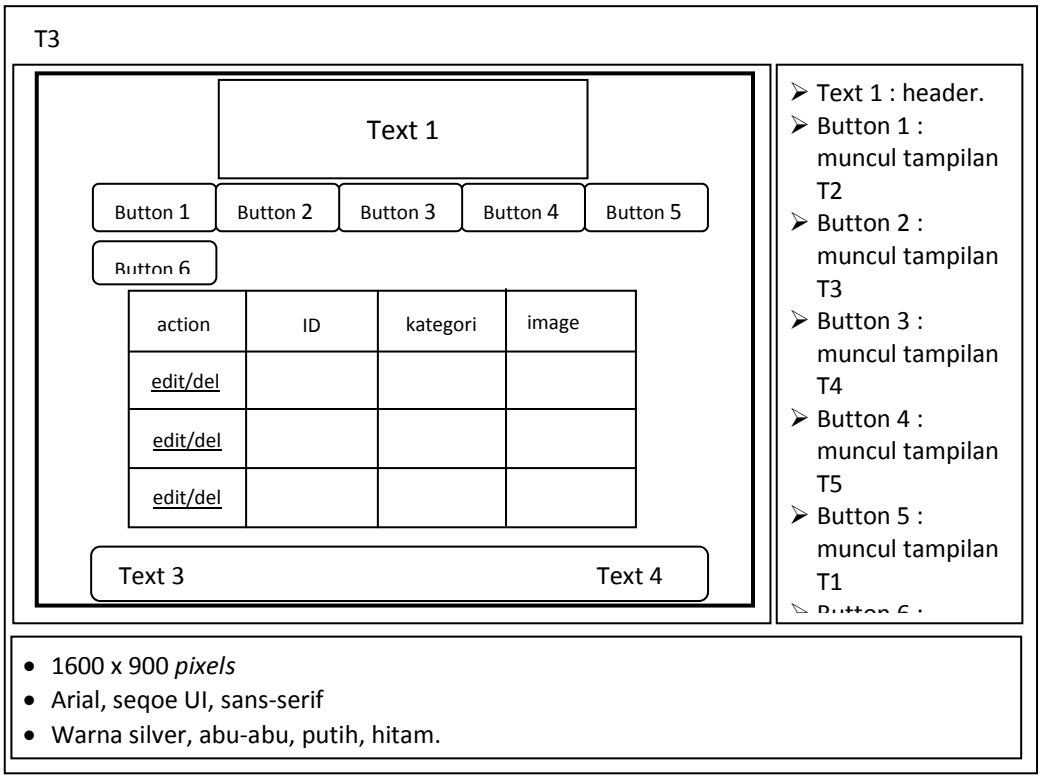
Gambar III.63 Perancangan Antarmuka *Login CMS ARPublicfacilities*

2) Perancangan Antarmuka Halaman Utama CMS



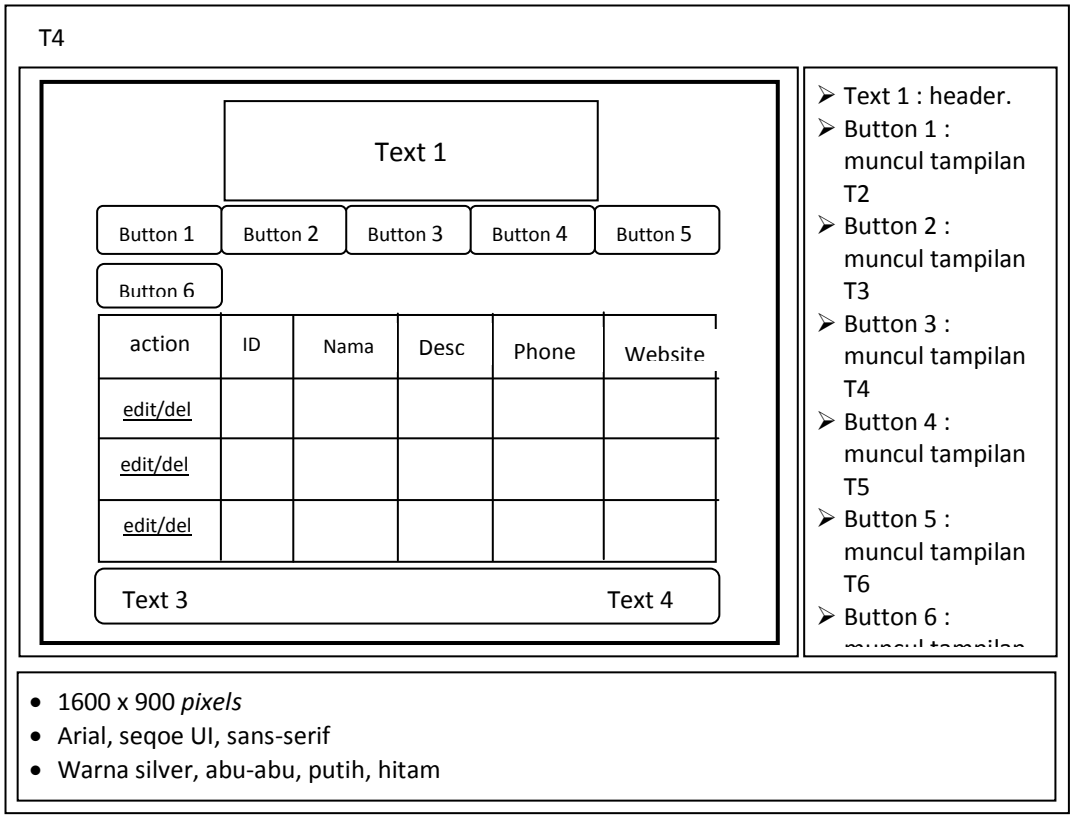
Gambar III.64 Perancangan Antarmuka Halaman Utama

3) Perancangan Antarmuka Kategori



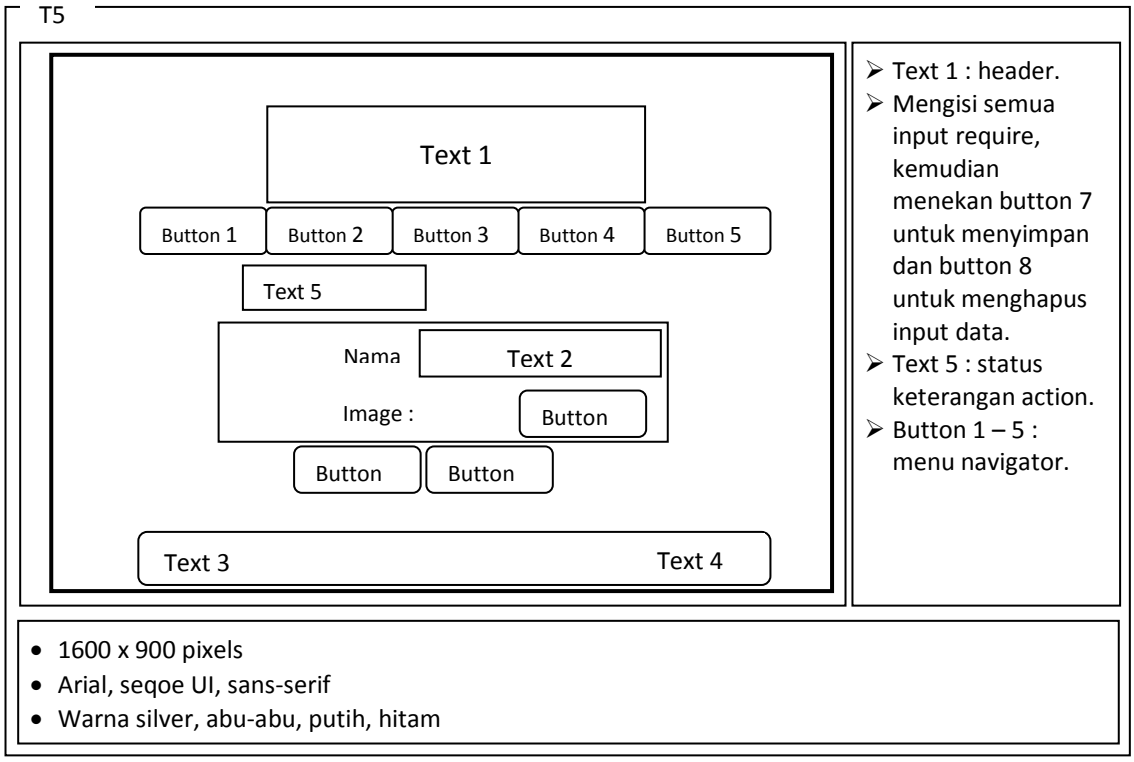
Gambar III.65 Perancangan Antarmuka Kategori

4) Perancangan Antarmuka Konten



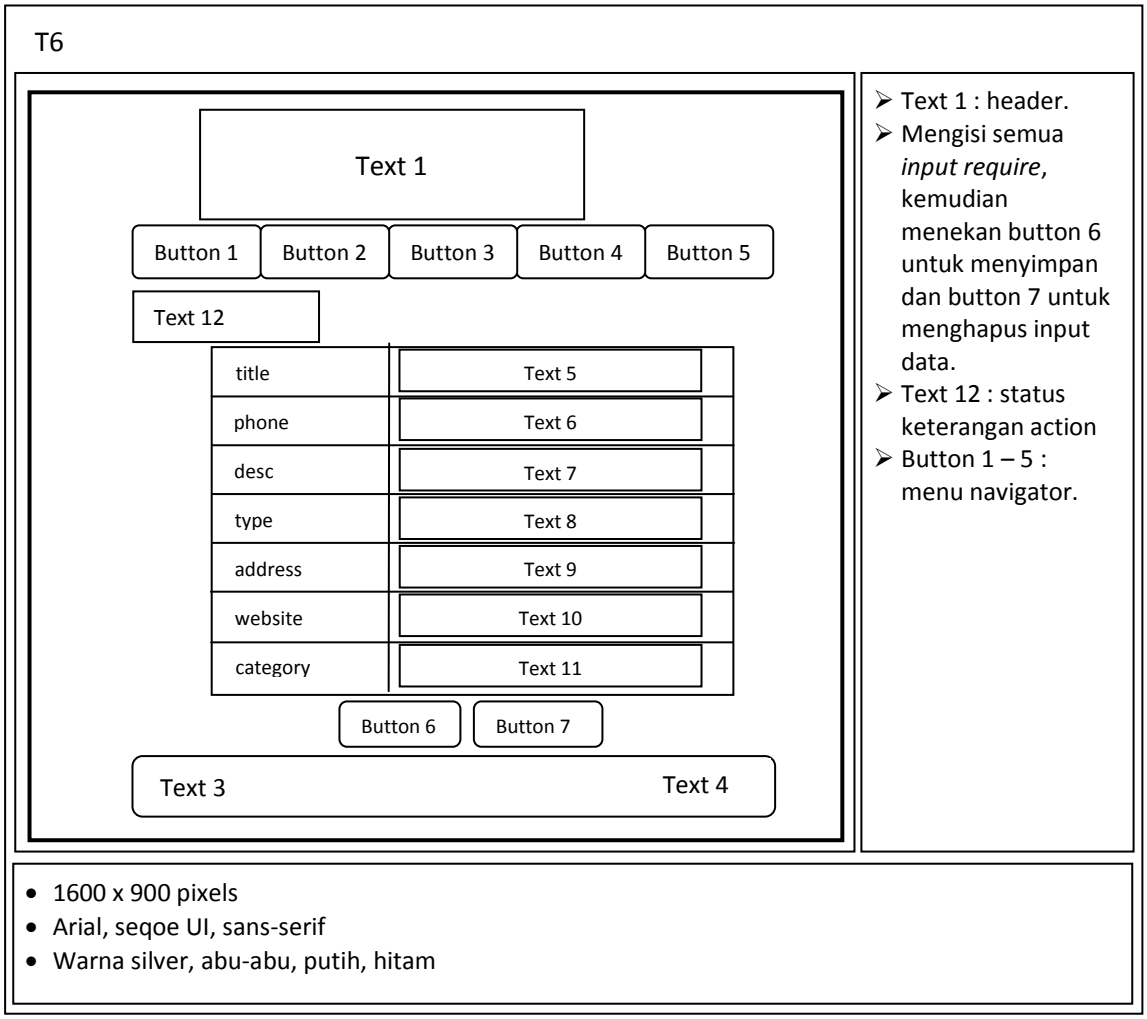
Gambar III.66 Perancangan Antarmuka Konten

5) Perancangan Antarmuka Tambah Kategori



Gambar III.67 Perancangan Antarmuka Tambah Kategori

6) Perancangan Antarmuka Tambah Konten



Gambar III.68 Perancangan Antarmuka Tambah Konten