BAB 3

ANALISIS DAN PERANCANGAN SISTEM

3.1 Analisis Sistem

Analisis sistem merupakan penguraian dari suatu sistem informasi yang utuh ke dalam bagian-bagian komponennya dengan maksud untuk mengindentifikasi dan mengevaluasi permasalahan-permasalahan, kesempatan-kesempatan, hambatan-hambatan yang terjadi dan kebutuhan-kebutuhan yang diharapkan sehingga dapat diusulkan perbaikan-perbaikan. Analisis merupakan tahapan yang paling penting, karena kesalahan dalam tahap ini akan menyebabkan kesalahan di tahap selanjutnya.

3.1.1 Analisis Masalah

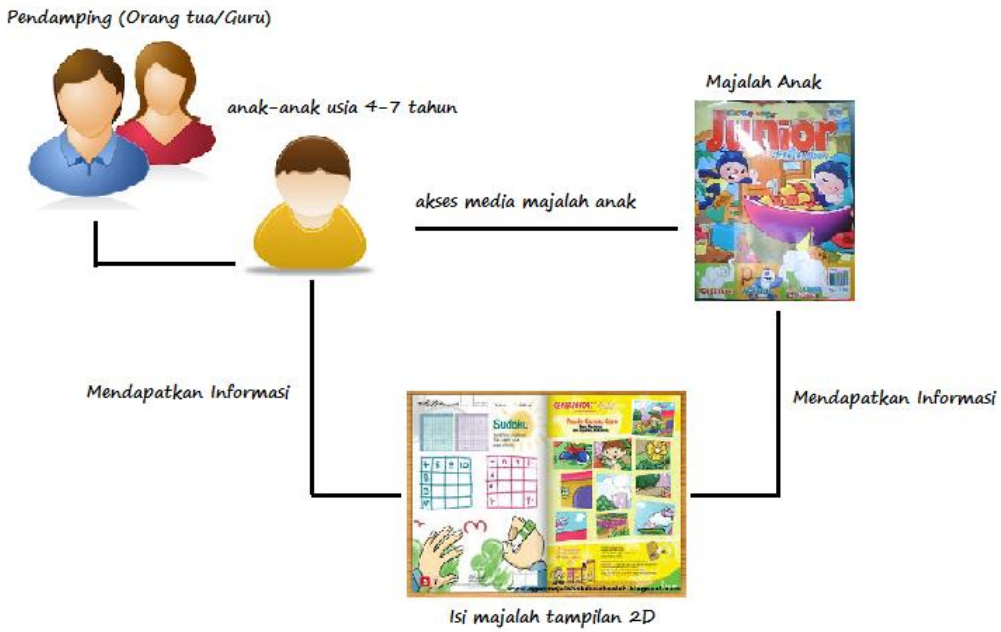
Berdasarkan penelitian dasar dari Nurchasanah (Dosen Jurusan Sastra Indonesia, Fakultas Sastra Universitas Negeri Malang) dengan judul *Model Performansi Majalah Anak-anak Untuk Meningkatkan Kreatifitas Berbahasa Lisan Anak Usia Pra-Sekolah* bahwa majalah anak selama ini difungsikan di TK memuat berbagai kompetensi dasar yang harus dikuasi anak diantaranya kompetensi berbahasa, daya pikir dan keterampilan. Ketiga kompetensi ini disajikan dengan berbagai variasi strategi pembelajaran. Namun demikian, sampai saat ini belum diketahui secara pasti model yang dapat difungsikan oleh guru untuk meningkatkan kompetensi berbahasa lisan anak dan diminati anak. Pada umumnya, guru hanya menyajikan majalah itu sebagai media pembelajaran tanpa mempertimbangkan kesesuaian dengan anak. Akibatnya anak-anak sering merasa jenuh belajar, karena model

pembelajarannya, isi dan latihan-latihannya selalu berulang dari edisi yang satu ke edisi berikutnya.

Keunggulan dari teknologi *augmented reality* dalam segi menampilkan suatu informasi dapat diterapkan pada majalah anak untuk dapat memberikan solusi dari permasalahan yang ada. Dengan demikian, tujuan dari pembangunan aplikasi yaitu meningkatkan minat belajar kepada anak-anak dengan memberikan fasilitas tambahan teknologi *augmented reality* berupa penerapan metode *markerless* pada gambar di majalah dan membuat aplikasi berbasis *website* untuk menjalankan aplikasi tersebut. Dengan menerapkan teknologi ini, diharapkan penyampaian informasi yang didapatkan lebih menarik dan interaktif yaitu berupa animasi 3D.

3.1.2 Analisis Sistem yang Berjalan

Analisis Sistem atau analisis proses adalah tahapan yang memberi gambaran tentang sistem yang sedang berjalan sekarang. Analisis ini bertujuan untuk memberi gambaran yang lebih detail bagaimana cara kerja dari sistem sekarang adalah pada gambar dibawah ini :



Gambar 3.1 Alur Sistem yang sedang berjalan

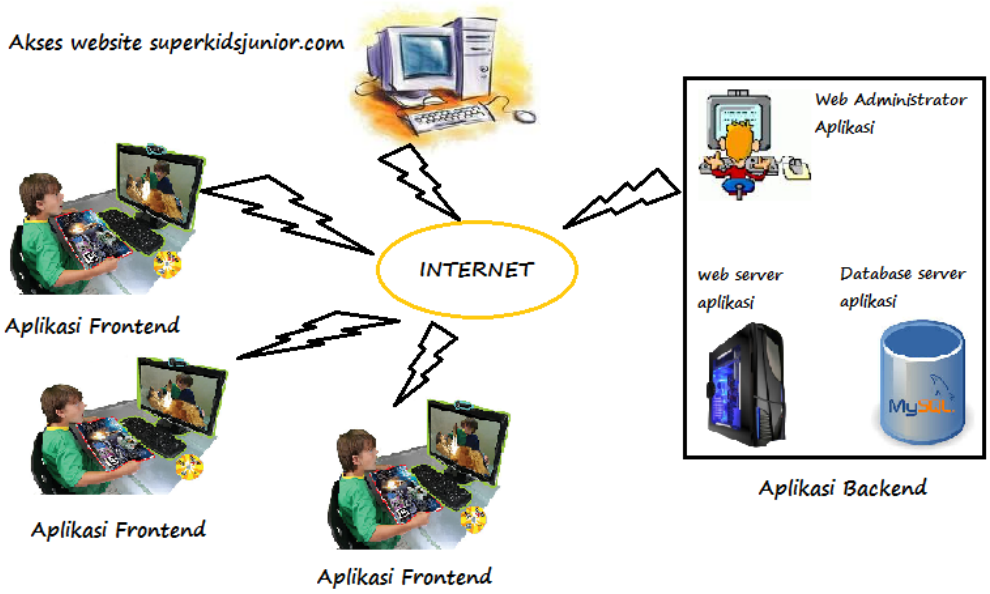
1. *User* disini adalah anak-anak yang berusia dari 4 – 7 tahun yang membaca majalah anak untuk mendapatkan informasi dan ilmu. Dan Pendamping adalah orang tua atau guru untuk mengarahkan anak-anak dalam proses belajar.
2. Majalah Anak adalah salah satu media penyampaian informasi dan sarana pembelajaran dalam pengenalan huruf, angka dan bentuk. Informasi di majalah berupa teks dan gambar 2D.

Dari gambaran prosedur media yang didapat untuk mengakses informasi pada majalah anak, teknologi *augmented reality* bisa dijadikan sebagai media alternatif sehingga konten informasi yang ingin disampaikan menjadi lebih menarik dan interaktif.

3.1.3 Analisis Arsitektur Sistem

Arsitektur fisik sistem yang dibangun menggunakan model *tree tier*. Pada arsitektur ini terdiri dari komponen aplikasi ARMagazine (*front-end Component*), *web admin* aplikasi, *web*

service dan *database server* aplikasi (*back-end Component*). Arsitektur sistem dalam pembangunan aplikasi ini dapat dilihat pada gambar 3.2.



Gambar 3.2 Arsitektur Aplikasi

Front-end adalah aplikasi majalah anak AR yang dijalankan melalui komputer atau *notebook* dengan fasilitas *webcam*. Aplikasi ini berfungsi sebagai antarmuka antara pembaca atau *user front-end* dengan sistem.

Back-end adalah aplikasi pendukung yang berfungsi sebagai penyuplai atau sumber data pada aplikasi *front-end*. Aplikasi ini terdiri dari tiga sub bagian utama yaitu *web administrator* aplikasi, *web service* aplikasi dan *database* server aplikasi. *Web administrator* aplikasi merupakan web yang berfungsi sebagai pengolah sumber basis data yang terdapat pada *database server* yang dikelola oleh *user back-end*. *Web service* aplikasi merupakan web yang berfungsi sebagai media pertukaran data antar database server dan aplikasi. *Database server* aplikasi merupakan aplikasi yang berfungsi sebagai sumber data (*data resource*) aplikasi.

3.1.4 Analisis Kebutuhan AR Toolkit/Library

Toolkit dan *library* untuk AR telah banyak dikembangkan. Perbandingan beberapa *toolkit* dan *library* untuk AR dapat dilihat pada Tabel 3.1

Tabel 3.1 Perbandingan *toolkit* dan *library* untuk AR

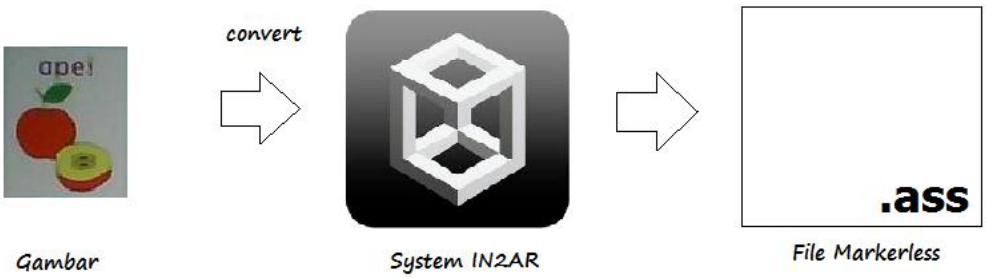
Parameter		**AR Toolkit/Library**			
		AR Toolkit	**Flartoolkit**	**D'Fusion Studio**	**IN2AR**
1.	Lisensi	Gratis	Gratis	Gratis	Gratis
2.	Dukungan bhs pemograman	C	Action script	LUA scripting	Action script
3.	Dukungan marker	Berbingkai hitam	Berbingkai hitam	Hampir semua jenis	Hampir semua jenis
4.	Authoring/non autohring	Non Authoring	Non Authoring	Authoring	Non Authoring
5.	Format model 3 dimensi yang dapat ditampilkan	wrl	.dae, .obj, .3ds	scenetta	.dae, .obj, .3ds
6.	Dukungan platform	Multiplatform	Multiplatform	Multiplatform	Multiplatform
7.	Distribusi kepada pengguna	Harus install ARtoolkit di computer pengguna	.swf	Menggunakan D'Fusion @HomeDekstopPlayer	.swf

Kelebihan IN2AR jika dibandingkan dengan toolkit atau library lain adalah :

1. Selain menyediakan lisensi berbayar (2212,50 EURO per proyek), juga terdapat lisensi gratis. Perbedaan kedua lisensi tersebut adalah bahwa aplikasi AR yang dibuat dengan lisensi gratis harus menyertakan logo IN2AR didalam aplikasi.
2. Dukungan pengembang di forum IN2AR sangat kuat. Pengembangan aktif menjawab pertanyaan seputar teknis IN2AR daru *user frontend*.
3. IN2AR dapat mengenali hampir semua jenis gambar sebagai marker. Berbeda dengan ARToolkit dan FLARToolkit yang hanya mengenali gambar dengan bingkai hitam sebagai marker.
4. Distribusi aplikasi yang dibuat dengan IN2AR cukup mudah. Aplikasi didistribusikan menggunakan file .swf yang dapat ditampilkan di halaman web. User frontend yang memiliki web browser dengan flash player dan computer webcam serta terkoneksi internet akan dapat mengaskses aplikasi AR tersebut.

Selain memiliki beberapa kelebihan, IN2AR juga memiliki kelemahan yaitu :

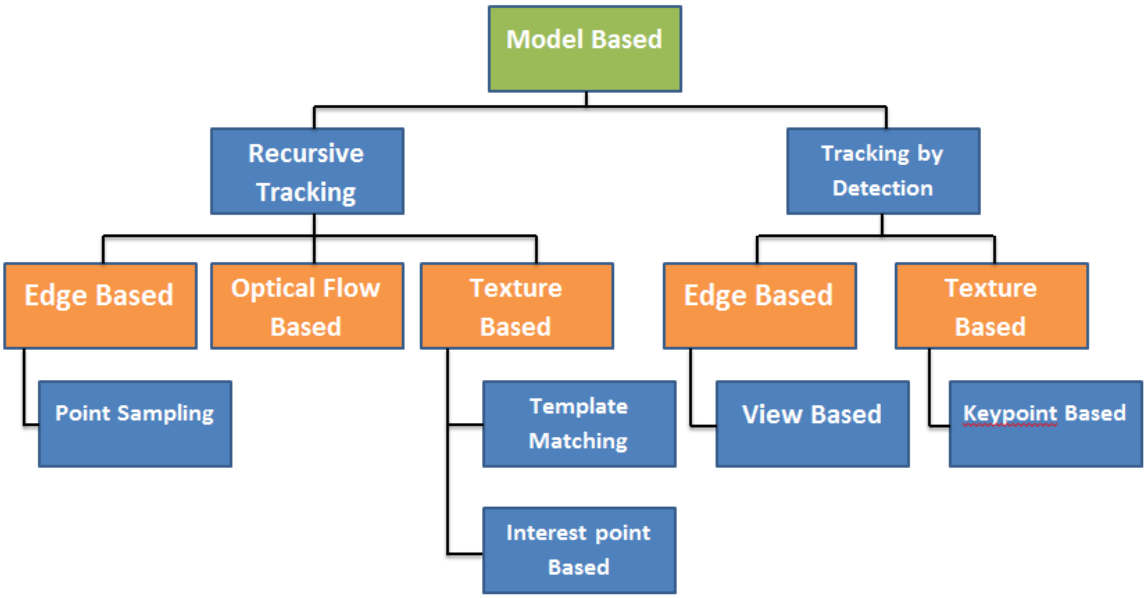
1. Pengguna harus mengenal bahasa pemograman, terutama bahasa pemograman Action Script 3. Hal ini karena segala macam perubahan dalam penggunaan IN2AR adalah dengan cara mengetikkan kode dalam bahasa pemograman Action Script 3. IN2AR tidak memiliki dokumentasi yang lengkap dan sistematis. Pengguna IN2AR harus melakukan eksplorasi sendiri untuk dapat menggunakan IN2AR sesuai kebutuhan. Aktivitas di forum IN2AR akan sangat membantu jika pengguna mengalami kesulitan.
2. Pembuatan marker untuk lisensi gratis harus melalui tim IN2AR. Gambar yang akan dijadikan marker harus dikirim terlebih dahulu ke tim IN2AR untuk dibuatkan file .ass.



Gambar 3.3 Proses Pembentukan *Marker*

3.1.5 Analisis *Markerless* AR *Tracking System*

Mengingat sifat pelacakannya, teknik berbasis model (*Model Based Techniques*) dapat diklasifikasikan dalam 2 bagian (Gambar 3.4) :



Gambar 3.4 *Model Based Techniques*

1. Pelacakan Rekursif (*Recursive Tracking*), dimana pada saat sebelum objek dimunculkan terlebih dahulu dilakukan perkiraan untuk menghitung pelacakan yang terjadi.
2. Pelacakan dengan pendeteksian (*Tracking by detection*), pada klasifikasi ini dilakukan perhitungan pelacakan tanpa perkiraan

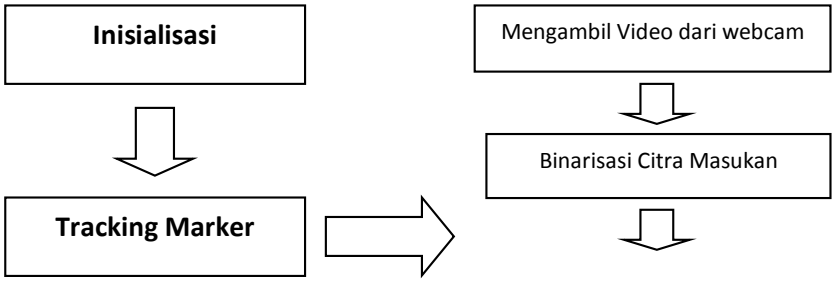
sebelumnya, yang memungkinkan otomatis inisialisasi dan pemulihan dari kegagalan.

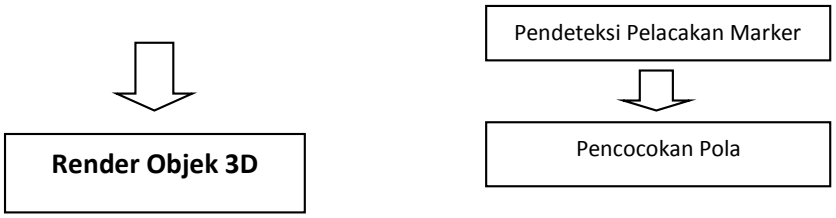
Pelacakan rekursif menjadi teknik model yang digunakan oleh pihak IN2AR untuk melakukan teknik *markerless AR tracking system*.

3.1.6 Analisis Metode Terhadap Kasus

Analisis metode terhadap kasus merupakan analisis yang mendeskripsikan bagaimana proses augmented reality IN2AR dari awal inisialisasi, tracking marker, sampai dengan proses rendering objek 3D dengan metode markerless yang diterapkan oleh IN2AR.

Dalam perancangan aplikasi dengan teknologi AR, seolah-olah menggabungkan objek virtual dengan objek nyata, dalam hal ini objek virtual berupa 2D atau 3D dan objek nyatanya berupa gambar dengan pola tertentu (*Markerless*). Sistem Augmented Reality Display yang digunakan adalah teknik spatial display (pembahasan pada subbab 2.6.3.3) dengan *screen display* (bisa menggunakan monitor ataupun proyektor).





Gambar 3.5 Diagram Alur Sistem *Augmented Reality*

Secara keseluruhan, proses sistem *augmented reality* dapat digambarkan dengan diagram alir pada gambar 3.5. aplikasi melakukan inisialisasi terlebih dahulu sebelum melakukan *tracking marker*, *marker* dideteksi dari masukan video *webcam*, jika *marker* terdeteksi maka objek 3D di-*render*.

Secara garis besarnya, dalam perancangan aplikasi ini ada tiga bagian utama yaitu sebagai berikut :

1. Inisialisasi
2. *Tracking Marker*
3. *Rendering Objek* 3D

3.1.6.1 Inisialisasi

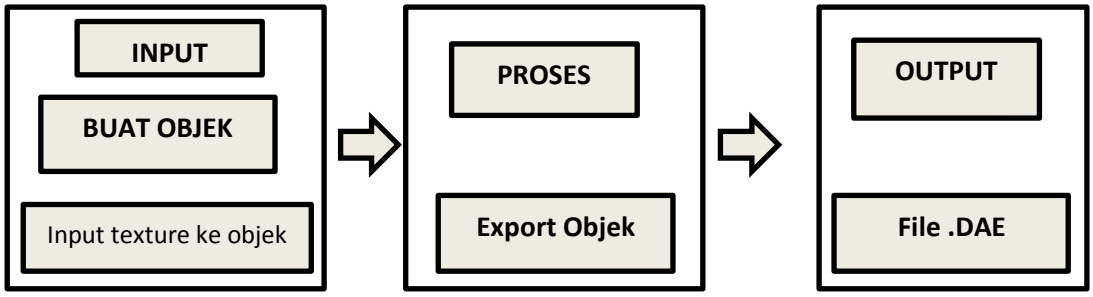
Pada tahap ini ditentukan *marker* yang akan digunakan, sumber input video-nya dan objek 3D yang akan digunakan. Pada bagian inisialisasi ini, objek 3D diinisialisasi terlebih dahulu karena loading objek 3D memerlukan waktu yang cukup lama.

3.1.6.1.1 Inisialisasi Model 3D

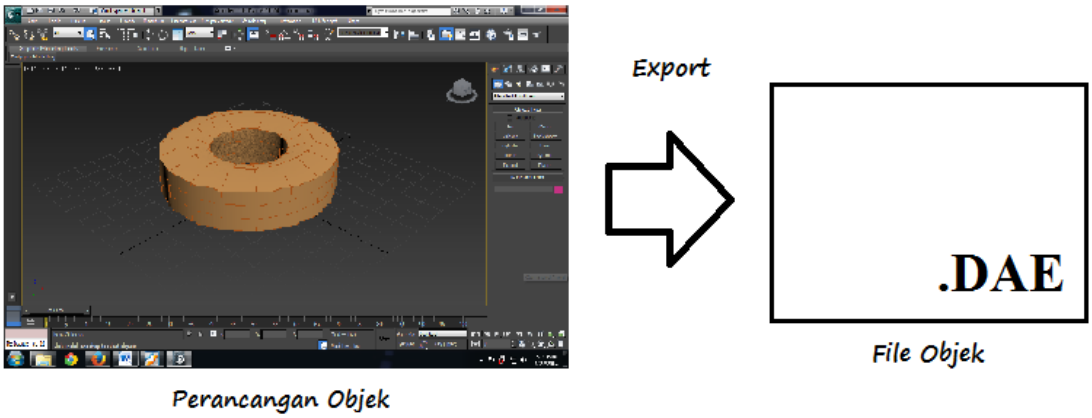
Model 3D yang akan ditampilkan di-*load* terlebih dahulu, agar aplikasi dapat menampilkan objek 3D tertentu tanpa merubah atau membangun

ulang aplikasi, diperlukan sebuah file konfigurasi untuk menentukan objek 3D yang akan di-*load* sesuai dengan pola *marker* yang dideteksi.

1. **Proses Pembentukan Objek**



Gambar 3.6 Proses Pembentukan Data Objek 3D



Gambar 3.7 Proses export objek

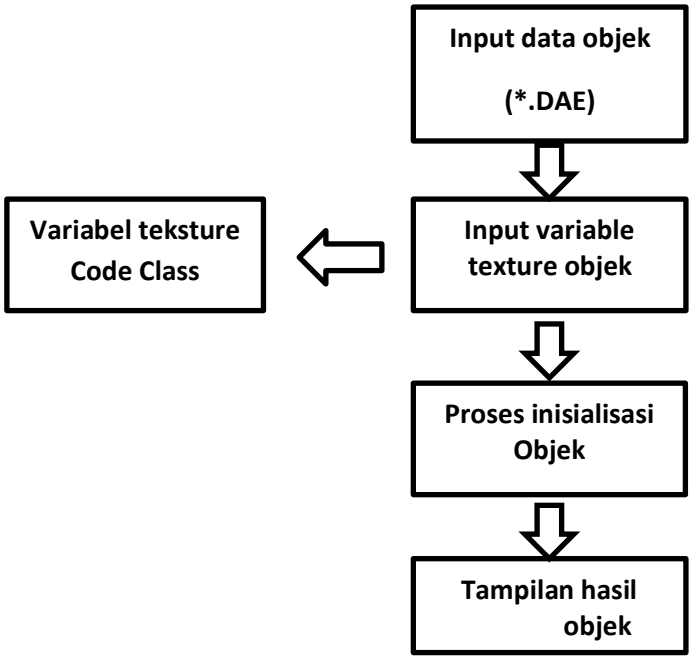
Dalam proses pemodelan terdiri dari 3 langkah, yaitu :

1. Menyesuaikan objek 3D dengan animasi atau bentuk yang akan dibuat.
2. Memasukan teksture sesuai dengan objek 3D.

3. Mengexport objek yang sudah dirancang dan dibuat ke dalam format collada (*.DAE).

Pada tahapan ketiga yaitu mengexport objek ke dalam format collada (*.DAE), tidak bisa dilakukan secara manual dengan menggunakan export bawaan dari aplikasi 3D pembuat objek. Export objek harus terlebih dahulu meng-install aplikasi OpenCOLLADA agar objek dapat ditampilkan sesuai dengan yang dibuat.

2. Proses Input Data Objek



Gambar 3.8 Proses Input Data Objek

a. Input data objek

Proses pertama yaitu menambahkan data objek .DAE ke dalam *library* IN2AR. File .DAE ini berfungsi untuk memanggil bentuk objek yang sudah diexport.

b. Input variable tekstur objek

Proses ini berfungsi untuk menambahkan tekstur pada objek, tekstur pada objek tidak bisa digunakan atau muncul sebelum ada penambahan variable dan pengaturan kode yang ditambahkan pada function *init3Dmodel* agar tekstur tersebut sesuai dengan model yang telah dibuat.

1. Penambahan variabel *teksture*

   Penambahan variabel *teksture* disesuaikan dengan jumlah dan nama material id pada file .DAE yang digunakan pada objek.

2. Penambahan *code class* variabel *teksture*

   Gambar atau *teksture* yang ditambah disesuaikan dengan variabel *teksture*.

c. Proses Inisialisasi objek

   Proses ini dilakukan pada saat objek di-compile menjadi aplikasi dan data objek beserta teksturnya objek akan di inisialisasi

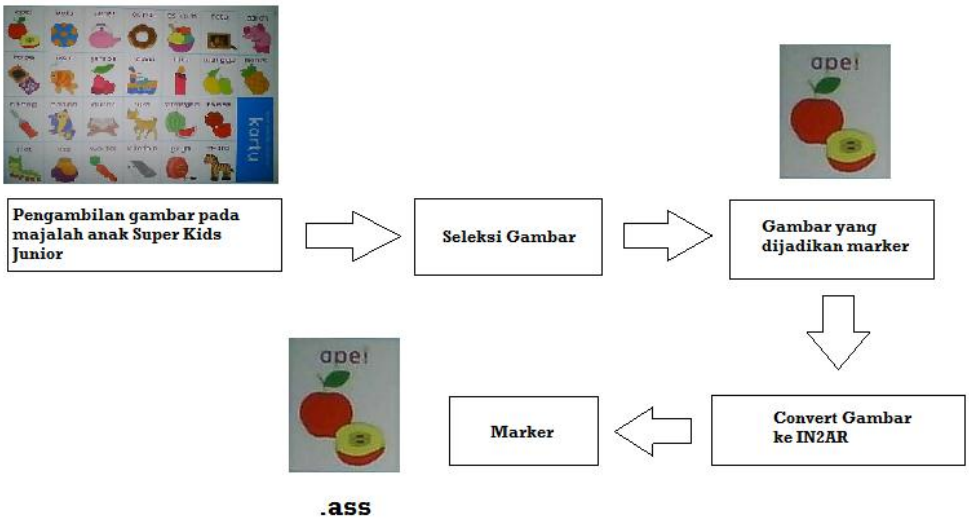
d. Tampilan hasil objek

   Proses yang terakhir adalah menampilkan hasil objek yang telah melalui proses inisialisasi, dimana pada saat objek tampil disesuaikan dengan marker yang ada.

3.1.6.1.2 Marker Yang Digunakan

Marker yang digunakan pada analisis ini didapatkan dari hasil gambar pada majalah anak yang sudah ada, ini disebabkan karena aplikasi ini masih bersifat sementara belum diresmikan.

Adapun proses pengambilan *marker* adalah sebagai berikut :



Gambar 3.9 Proses Pembentukan *Marker*

Proses pembuatan gambar menjadi *marker* dilakukan oleh pihak IN2AR dengan cara mengirimkan gambar yang akan di-*convert*, melalui *marker engine* yang disediakan oleh pihak IN2AR. Setelah gambar di-*convert* menghasilkan file dengan format .ass. file tersebut kemudian dijadikan masukan pada *coding* untuk mendeteksi gambar yang dijadikan *marker*.

Spesifikasi pola *marker* :

1. Pola *marker* minimum harus memiliki lebar 550 pixels
2. Format gambar yang dikirimkan .jpg



Gambar 3.10 *Marker* yang digunakan

3.1.6.2 *Tracking Marker*

IN2AR memiliki kemampuan untuk mendeteksi gambar dan menghitung posisi gambar tersebut menggunakan *webcam* standar. Informasi posisi yang didapatkan akan dipergunakan untuk menempatkan objek atau model tiga dimensi atau video ke dalam posisi gambar atau *markerless*. Ada 5 langkah dalam proses kerja *Tracking Marker* IN2AR.

1. Menjalankan IN2AR dengan membuat *instance* dari *class* ASFEAT.

```
asfeat = new ASFEAT();

        asfeat.addEventListener(Event.INIT, setup);
```

2. Lakukan pengaturan berbagai parameter dari IN2AR, diantaranya : jumlah titik *region of interest* di tiap frame gambar webcam yang dapat dideteksi, jumlah marker yang digunakan, pengaturan memori dan lain-lain.

```
private function initIN2AR():void
                {
                        var maxPoints:int = 300;
//banyaknya titik ROI di tiap frame gambar webcam
yang dapat dideteksi
                        var maxMarkers:int = 1;
//jumlah marker yang kita gunakan, untuk saat ini
single marker dulu yha :)
                        var maxTransformError:Number
= 10 * 10;

                        var memory:ByteArray = new
ByteArray; //pengaturan memori yang diperlukan oleh
IN2AR
                        memory.endian =
Endian.LITTLE_ENDIAN;
```

```
memory.length =
asfeat.lib.calcRequiredChunkSize(WIDTH, HEIGHT,
maxPoints, maxMarkers);
                        memory.position = 0;

        applicationDomain.domainMemory = memory;

asfeat.lib.addReferenceObject(ByteArray(new
MarkerASS)); //menambahkan marker yang digunakan

        asfeat.lib.setSingleReferenceMode(true);
//single marker

        asfeat.lib.addListener(ASFEATDetectionEvent
.DETECTED, onMarkerDetected);
                }
```

Kemudian aktifkan *webcam* dan menampilkannya.

```
private function initWebcam():void
                {
                        bitmapData = new
BitmapData(WIDTH, HEIGHT);
                        var camera:Camera =
Camera.getCamera(); //mengaktifkan webcam
                        camera.setMode(WIDTH,
HEIGHT, FPS);
                        video = new Video(WIDTH,
HEIGHT);
                        video.attachCamera(camera);
                        addChild(video);
                }
```

3. Inisialisasi untuk penyesuaian parameter Camera 3D di Papervision3D

```
package arsupport
{
        import
ru.inspirit.asfeat.calibration.IntrinsicParameters;

        import org.papervision3d.cameras.Camera3D;
        import org.papervision3d.core.math.Matrix3D;

        public class ARPV3DCamera extends Camera3D
        {
                protected var intrinsic:IntrinsicParameters;
                public function
ARPV3DCamera(intrinsic:IntrinsicParameters, width:int,
height:int)
                {
                        super();
                        this.z = 0;
                        this.intrinsic = intrinsic;
                        setupCamera(width, height);
                }
                public function setupCamera(width:int,
height:int):void
                {
                        useProjectionMatrix = true;
                        _projection = new
Matrix3D(getPV3DProjectionMatrix(width, height, 1, 10000));
                }

                override public function
transformView(transform:Matrix3D = null):void
                {
                        eye.calculateInverse(this.transform);
                        eye.calculateMultiply4x4(_projection,
eye);
                }
                public final function
getPV3DProjectionMatrix(width:int, height:int,
nearClip:Number = 10, farClip:Number = 10000):Array
                {

var PV3D_PROJECTION_MATRIX:Array = new Array(16);

                        var i:int, j:int;
                        var trans:Array = [[], [], []];
                        var icpara:Array = [[], [], []];
                        var p:Array = [[], [], []];
                        var q:Array = [[], [], [], []];
```

```
                        decomposeMat(intrinsic.fx, -
intrinsic.fy, intrinsic.cx, (height - 1) * 1.0 -
intrinsic.cy, icpara, trans);
                        for(i = 0; i < 3; ++i)
                        {
                                for(j = 0; j < 3; ++j)
                                {
                                        p[i][j] = icpara[i][j]
/ icpara[2][2];
                                }
                        }

        q[0][0] = (2.0 * p[0][0] / (width - 1));
        q[0][1] = (2.0 * p[0][1] / (width - 1));
        q[0][2] = -((2.0 * p[0][2] / (width - 1))  - 1.0);
        q[0][3] = 0.0;

        q[1][0] = 0.0;
        q[1][1] = -(2.0 * p[1][1] / (height - 1));
        q[1][2] = -((2.0 * p[1][2] / (height - 1)) - 1.0);
        q[1][3] = 0.0;

        q[2][0] = 0.0;
        q[2][1] = 0.0;
        q[2][2] = -(farClip + nearClip) / (nearClip -
farClip);
        q[2][3] = 2.0 * farClip * nearClip / (nearClip -
farClip);

        q[3][0] = 0.0;
        q[3][1] = 0.0;
        q[3][2] = 1.0;
        q[3][3] = 0.0;
        for (i = 0; i < 4; i++) { // Row.
// First 3 columns of the current row.
        for (j = 0; j < 3; j++) { // Column.

PV3D_PROJECTION_MATRIX[i*4 + j] =
        q[i][0] * trans[0][j] +
        q[i][1] * trans[1][j] +
        q[i][2] * trans[2][j];
                                }
// Fourth column of the current row.
PV3D_PROJECTION_MATRIX[i*4 + 3]=
        q[i][0] * trans[0][3] +
        q[i][1] * trans[1][3] +
        q[i][2] * trans[2][3] +
        q[i][3];
                        }

                        return PV3D_PROJECTION_MATRIX;
                }
```

```
public final function decomposeMat(fx:Number, fy:Number,
cx:Number, cy:Number, cpara:Array, trans:Array):void
                {
                        var rem1:Number, rem2:Number,
rem3:Number;
                        var c00:Number, c01:Number,
c02:Number, c03:Number, c10:Number, c11:Number, c12:Number,
c13:Number, c20:Number, c21:Number, c22:Number, c23:Number;

                        c00 = fx;
                        c01 = 0.0;
                        c02 = cx;
                        c03 = 0.0;
                        c10 = 0.0;
                        c11 = fy;
                        c12 = cy;
                        c13 = 0.0;
                        c20 = 0.0;
                        c21 = 0.0;
                        c22 = 1.0;
                        c23 = 0.0;

                        var r:int;
                        var c:int;
                        for (r = 0; r < 3; ++r)
                        {
                                for (c = 0; c < 4; ++c)
                                {
                                        cpara[r][c] = 0.0;
                                }
                        }
                        cpara[2][2] = Math.sqrt(c20*c20 +
c21*c21 + c22*c22);
                        trans[2][0] = c20 / cpara[2][2];
                        trans[2][1] = c21 / cpara[2][2];
                        trans[2][2] = c22 / cpara[2][2];
                        trans[2][3] = c23 / cpara[2][2];

                        cpara[1][2] = trans[2][0]*c10 +
trans[2][1]*c11 + trans[2][2]*c12;
                        rem1 = c10 - cpara[1][2] *
trans[2][0];
                        rem2 = c11 - cpara[1][2] *
trans[2][1];
                        rem3 = c12 - cpara[1][2] *
trans[2][2];
```

```
cpara[1][1] = Math.sqrt(rem1*rem1 + rem2*rem2 + rem3*rem3);
                        trans[1][0] = rem1 / cpara[1][1];
                        trans[1][1] = rem2 / cpara[1][1];
                        trans[1][2] = rem3 / cpara[1][1];

cpara[0][2] = trans[2][0]*c00 + trans[2][1]*c01 +
trans[2][2]*c02;
cpara[0][1] = trans[1][0]*c00 + trans[1][1]*c01 +
trans[1][2]*c02;
rem1 = c00 - cpara[0][1] * trans[1][0] - cpara[0][2] *
trans[2][0];
rem2 = c01 - cpara[0][1] * trans[1][1] - cpara[0][2] *
trans[2][1];
rem3 = c02 - cpara[0][1] * trans[1][2] - cpara[0][2] *
trans[2][2];
cpara[0][0] = Math.sqrt(rem1*rem1 + rem2*rem2 + rem3*rem3);

trans[0][0] = rem1 / cpara[0][0];
trans[0][1] = rem2 / cpara[0][0];
trans[0][2] = rem3 / cpara[0][0];

trans[1][3] = (c13 - cpara[1][2] * trans[2][3]) /
cpara[1][1];
trans[0][3] = (c03 - cpara[0][1] * trans[1][3] - cpara[0][2]
* trans[2][3]) / cpara[0][0];

                        for (r = 0; r < 3; ++r)
                        {
                                for (c = 0; c < 3; ++c)
                                {
                                        cpara[r][c] /=
cpara[2][2];
                                }
                        }
                }
        }
}
```

4. Proses pendeteksian ada atau tidaknya marker dilakukan pada tiap frame gambar yang diperoleh dari webcam.

```
private function onEnterFrame(e:Event):void
                {
                        markerFound = false;
                        bitmapData.draw(video);
                        asfeat.lib.detect(bitmapData);

                        if (!markerFound)
                        {
                                onMarkerLost(0);
                        }
                }
```

5. Jika ada marker yang terdeteksi, maka model 3D akan ditampilkan di atas marker tersebut berdasarkan matriks transformasi dari marker IN2AR yang telah dipetakan ke matriks transformasi Papervision3D. Bila marker tidak terdeteksi, maka model 3D akan dihilangkan.

```
private function
onMarkerDetected(e:ASFEATDetectionEvent):void
                {
        var markerInfoList:Vector. = e.detectedReferences;
        var markerInfo:ASFEATReference;
        var n:int = markerInfoList.length; //jumlah marker
yang terdeteksi

                for (var i:int = 0; i < n; i++) //berguna
ketika kode ini nanti dimodifikasi menjadi multiple marker
                        {
        markerInfo = markerInfoList[i];
        var rot:Vector. = markerInfo.rotationMatrix;
        var trans:Vector. = markerInfo.translationVector;

//mapping matriks transformasi dari IN2AR ke PV3D
        var PV3D_MATRIX:Matrix3D = new Matrix3D()
        PV3D_MATRIX.n11 = rot[1]; //Scale & Yaw & Roll
        PV3D_MATRIX.n12 = rot[0]; //Roll
        PV3D_MATRIX.n13 = rot[2]; //Yaw
        PV3D_MATRIX.n14 = trans[0]; //Trans
        PV3D_MATRIX.n21 = rot[4]; //Roll
        PV3D_MATRIX.n22 = rot[3]; //Scale & Pitch & Rol
        PV3D_MATRIX.n23 = rot[5]; //Pitch
        PV3D_MATRIX.n24 = trans[1]; //Trans
        PV3D_MATRIX.n31 = rot[7]; //Yaw
        PV3D_MATRIX.n32 = rot[6]; //Yaw & Pitch
        PV3D_MATRIX.n33 = rot[8]; //Scale & Pitch
        PV3D_MATRIX.n34 = trans[2]; //Trans

if (arrayModelContainer[markerInfo.id].visible == false)
                                {
arrayModelContainer[markerInfo.id].visible = true;
                                }
arrayModelContainer[markerInfo.id].transform =
PV3D_MATRIX;
markerFound = true;
renderEngine.render();
                        }
                }
```

```
private function onMarkerLost(i:int):void
                {
                        arrayModelContainer[i].visible =
false;
                        renderEngine.render();
                }
```

3.1.6.3 Rendering Objek 3D

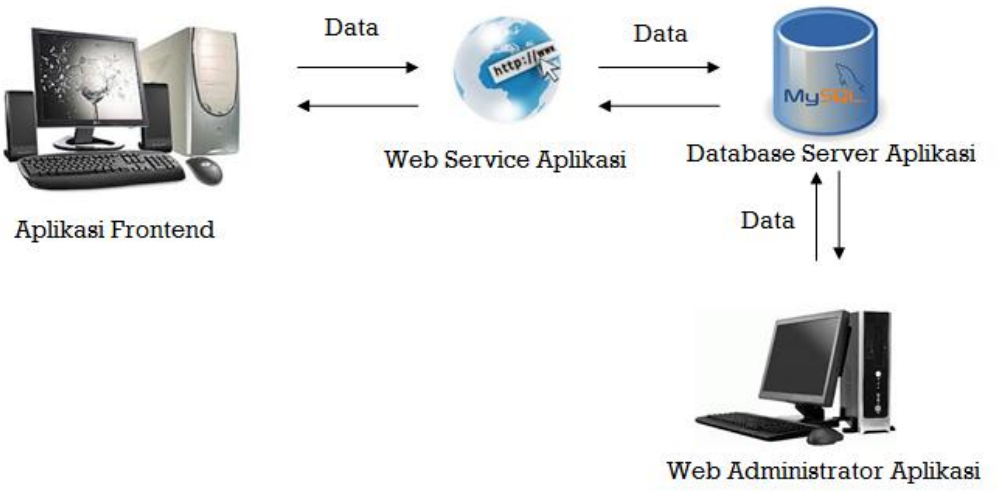
Transformasi matriks yang dikalkulasikan di step sebelumnya yang digunakan IN2AR dan menampilkan objek yang sesuai dengan sebuah *library* 3D. IN2AR menyertakan kelas pendukung yang mengkonversikan transformasi matriks IN2AR ke setiap kelas matriks internal *library* 3D tersebut.



Gambar 3.11 Render Objek 3D

3.1.7 Analisis Alur Data Sistem

Analisis ini berfungsi untuk menggambarkan secara rinci bagaimana sistem bekerja, meliputi :



Gambar 3.12 Alur Data Sistem

1. Aplikasi *Front-end*

   Aplikasi dalam hal ini adalah analisis fungsionalitas aplikasi untuk mengakses data dari *database server* melalui *website* Super Kids Junior.

2. *Web Service* Aplikasi

   Web Service aplikasi disini menampilkan tampilan aplikasi sesuai data yang ada pada *database server* aplikasi yang nantinya akan ditampilkan pada situs Super Kids Junior.

3. *Database Server* Aplikasi

   Pada analisis ini *database server* aplikasi menampung data aplikasi yang akan dibuat oleh *web administrator* aplikasi yang kemudian akan digunakan sesuai dengan keperluan pada *web service* Super Kids Junior aplikasi.

4. *Web Administrator* Aplikasi

   Web administrator aplikasi disini bertugas untuk membuat objek yang akan ditampilkan pada aplikasi berupa animasi atau video sesuai dengan informasi yang ada pada majalah.

3.1.8 Analisis Kebutuhan Non Fungsional

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

Pada analisis kebutuhan sistem non-fungsional ini dijelaskan analisis kebutuhan perangkat lunak, analisis kebutuhan perangkat keras, dan analisis pengguna.

3.1.8.1 Analisis Kebutuhan Perangkat Lunak

Perangkat lunak yang digunakan untuk membangun aplikasi buku AR ini adalah sebagai berikut :

1. Sistem Operasi Microsoft 7 Ultimate
2. Adobe Flash CS5
3. FlashDevelop Ver 4.0 RC3
4. Adobe Dreamweaver CS6
5. Autodesk 3DS Max 2013
6. *Papervision3D*

3.1.8.2 Analisis Kebutuhan Perangkat Keras

Berdasarkan studi literature terhadap proyek-proyek yang mengembangkan teknologi *Augmented Reality*, maka diperoleh spesifikasi minimum perangkat keras untuk teknologi *augmented reality* seperti berikut ini :

1. Processor minimal 1,8 GHz atau lebih
2. Random Access Memory (RAM) 1GB
3. VGA card minimal 128 MB
4. *Mouse* dan *Keyboard*
5. *Webcam 1,3 Megapixel*, dipakai untuk menangkap citra yang kemudian diproses oleh *tracker library*. Hasil dari proses ini akan menghasilkan matriks transformasi *marker relative* terhadap *webcam*.
6. Majalah Super Kids Junior sebagai *Marker*, dibutuhkan oleh *tracking library* untuk menempatkan model virtual di dunia nyata dengan cara menentukan koordinat *marker relative* kamera.

3.1.8.3 Analisis Pengguna Sistem (*User*)

Analisis pengguna dimaksudkan untuk mengetahui siapa saja pengguna yang terlibat dalam proses penggunaan aplikasi majalah AR sehingga dapat diketahui hak akses masing-masing pengguna dapat dilihat table 3.1.

Pengguna sistem dibagi menjadi dua bagian, yaitu :

1. Pengguna aplikasi majalah AR (*user front-end*) yaitu para pembaca majalah Super Kids Junior, dalam menggunakan sistem ini pengguna diharuskan terhubung dengan koneksi jaringan internet dan memiliki *webcam* untuk bisa menggunakan aplikasi ini.

2. Admin (*user back-end*) sebagai pengelola konten mempunyai kapabilitas dalam mengelola database seperti menambah, mengubah dan menghapus melalui *web administrator*.

Tabel 3.2 Tabel Karakteristik Pengguna

Pengguna	Hak Akses	Hak Akses
User Front-end	Mengakses data informasi majalah AR melalui aplikasi *Front-end*	Melihat informasi
User Back-end	Mengakses mengelola aplikasi *back-end*	Mengubah, menambah, menghapus rubrik data dan menampilkan data

3.1.9 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.

3.1.9.1 Analisis Kebutuhan Sistem

Dalam pembangunan aplikasi perancangan sistem, 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 berita animasi pada *display user front-end*
2. Informasi berita animasi atau video yang diinformasikan jelas.
3. Memberikan informasi visual mengenai berita yang terkait dengan bagian atau rubrik majalah.
4. Informasi mengenai berita animasi yang sudah diterapkan teknologi *augmented reali*ty segera ditampilkan

3.1.9.2 Analisis Kebutuhan Konten

Analisis ini bertujuan untuk menganalisis konten yang diperlukan agar memudahkan perancangan informasi setelah sistem yang dibutuhkan telah diketahui. Adapun analisis kebutuhan konten dengan kriteria berupa konten Animasi, yaitu menampilkan animasi berupa objek 3D.

3.1.9.3 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 ke dalam bentuk diagram.

Analisis kebutuhan fungsional ini dimodelkan dengan menggunakan UML (*Unifed Modeling Language*). Dimana tahap-tahap perancangan yang dilakukan dalam membangun aplikasi buku AR antara lain *Use Case Diagram, Class Diagram, Sequence Diagram* dan *Activity Diagram.*

1. Use Case Diagram

Diagram *Use Case* merupakan pemodelan untuk kelakuan (*behavior*) sistem informasi yang akan dibuat. *Use Case* mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem informasi yang akan dibuat. *Use case* terdiri dari tiga bagian yaitu identifikasi *actor*, identifikasi *use case* dan scenario *use case*.

Berikut ini adalah perancangan proses-proses yang terdapat pada aplikasi buku AR, yang digambarkan dengan *Use Case Diagram* yang dapat dilihat berikut ini :

1.1 Identifikasi Aktor

Aktor adalah abstraksi dari orang dan sistem yang lain yang mengkatifkan fungsi dari target sistem. Berikut adalah aktor yang berperan dalam menjalankan sistem yang dibangun.

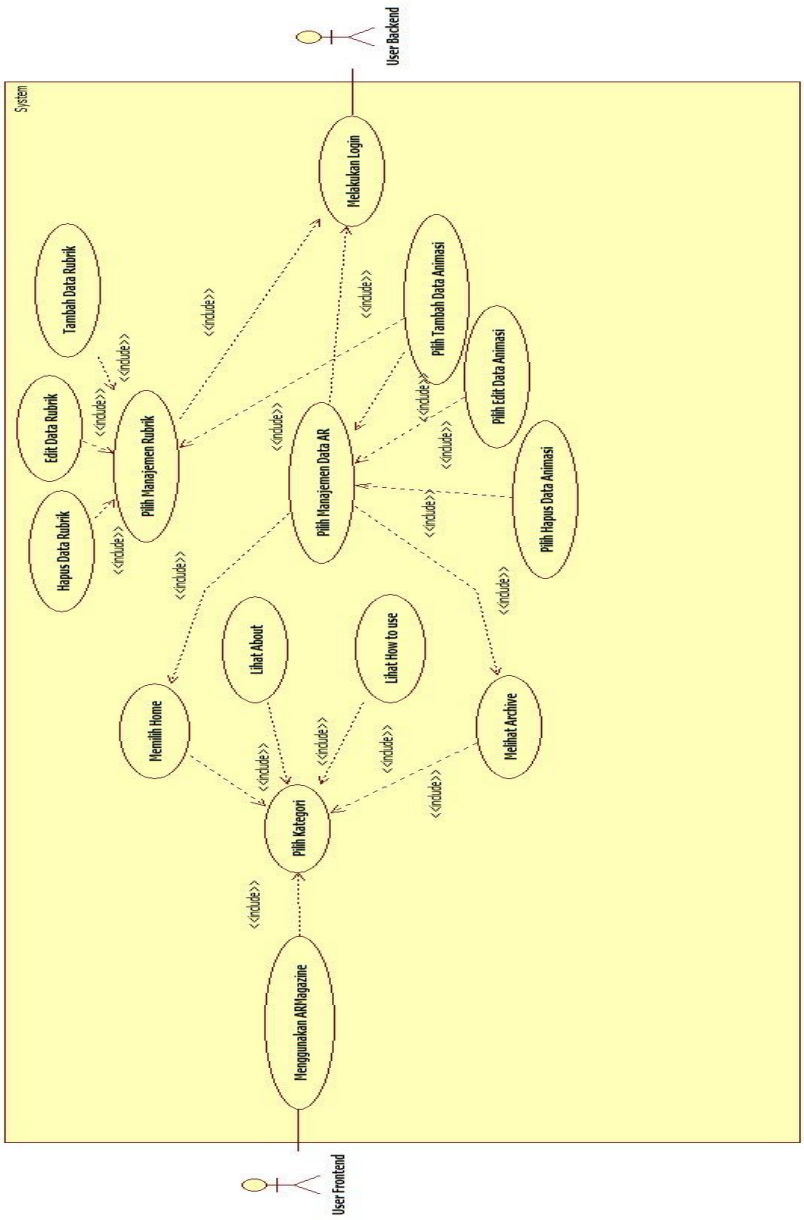
Tabel 3.3 Tabel *Actor Use Case*

No	Aktor	Deskripsi
A-01	*User Frontend*	Merupakan *actor* atau pengguna akhir dari aplikasi yang dibangun, aplikasi *frontend* (AR*Magazine*).
A-02	*User Backend*	Meupakan *actor* yang berperan mengelola data informasi dalam *database server* melalui *web administrator*.

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 *user frontend* yaitu yang menjalankan aplikasi *frontend*, sedangkan aktor *user backend* berperan mengelola aplikasi *backend* pada *web administrator*.

1.2 *Use Case Diagram*


User Backend
System
Melakukan Login
Tambah Data Rubrik
Edit Data Rubrik
Hapus Data Rubrik
Pilih Manajemen Rubrik
Pilih Tambah Data Animasi
Pilih Edit Data Animasi
Pilih Hapus Data Animasi
Pilih Manajemen Data AR
Memilih Home
Lihat About
Lihat How to use
Melihat Archive
Pilih Kategori
Menggunakan ARMagazine
User Frontend
<<include>>

Gambar 3.13 *Use Case Diagram*

1.3 Identifikasi *Use Case Diagram*

Berikut table identifikasi *use case* yang terdapat dalam sistem :

Tabel 3.4 Identifikasi *Use Case Diagram*

Use Case* Aplikasi *Frontend		
No	***Use Case***	**Deskripsi**
UCF-01	Menggunakan AR Magazine	Fungsionalitas untuk langsung menjalankan dan menampilkan aplikasi AR*Magazine* dengan menggunakan *webcam* pc/laptop.
UCF-02	Pilih Rubrik	Fungsionalitas memungkinkan *user frontend* memilih rubrik yang ada pada Majalah Super Kids Junior *Online*.
UCF-03	Melihat *Home*	Fungsionalitas untuk menampilkan halaman *home*
UCF-04	Lihat *About*	Fungsionalitas untuk menampilkan *About* tentang aplikasi dan sedikit penjelasan tentang teknologi *augmented reality*.
UCF-05	Lihat *How to use*	Fungsinalitas untuk menampilkan cara penggunaan aplikasi.
UCF-06	Memilih *Archive*	Fungsionalitas untuk menampilkan dan memilih *archive*/data sesuai dengan tanggal penerbitan majalah.
Use Case* Aplikasi *Backend		
No	***Use Case***	**Deskripsi**

UCB-1	Melakukan Login	Fungsionalitas untuk proses autentifikasi hak akses kepada *user backend*.
UCB-2	Pilih Manajemen Rubrik	Fungsionalitas yang memungkinkan *user backend* dapat mengelola data rubrik.
UCB-3	Tambah Rubrik	Fungsionalitas untuk menambah data rubrik.
UCB-4	Edit Rubrik	Fungsionalitas untuk mengubah atau *update* data rubrik.
UCB-5	Hapus Rubrik	Fungsionalitas untuk menghapus data rubrik.
UCB-6	Pilih Manajemen Data AR	Fungsionalitas yang memungkinkan *user backend* dapat mengelola data AR.
UCB-7	Pilih Tambah Data Animasi	Fungsionalitas untuk menambahkan data AR Animasi
UCB-8	*Edit* Data Animasi	Fungsionalitas untuk mengubah atau *update* data AR Animasi.
UCB-9	Hapus Data Animasi	Fungsionalitas untuk menghapus data AR Animasi.
UCB-10	Pilih Tambah data video	Fungsionalitas untuk menambahkan data AR.
UCB-11	*Edit* Data Video	Fungsionalitas untuk mengubah atau *update* data AR.
UCB-12	Hapus Data Video	Fungsinalitas untuk menghapus data AR.

1.4 Skenario *Use Case*

1.4.1 Skenario *Use Case* Aplikasi *Frontend*

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

Tabel 3.5 Skenario *Use Case* Menggunakan AR*Magazine*

Identifikasi	
Nama Use Case	Menggunakan ARMagazine
Aktor	User Frontend
Tujuan	Menggunakan Aplikasi ARMagazine
Keadaan Awal	Sistem Menampilkan menu Rubrik
Skenario Utama	
Aksi Aktor	**Reaksi Sistem**
	1. Menampilkan halaman menu aplikasi
2. Menyiapkan *webcam* dan majalah yang akan di AR-kan dan menekan *allow* yang ditampilkan sistem	
	3. Setelah menekan *allow* maka sistem akan menampilkan informasi pada *display* sesuai dengan marker dan tanggal majalah diterbitkan
Skenario Alternatif – Validasi Gagal	
Aksi Aktor	**Reaksi Sistem**

	1. "*Flash Player not detected, Click Below to download the latest Adobe Flash Player*".
2. Menekan tombol "*Download*"	
	2. Sistem akan terminate
Kondisi Akhir	Menampilkan informasi sesuai yang dipilih

Tabel 3.6 Skenario Pilih Rubrik

Identifikasi	
Nama Use Case	Pilih Rubrik
Aktor	User *Frontend*
Tujuan	Melihat konten berdasarkan rubrik
Keadaan Awal	Sistem menampilkan menu rubrik
Skenario Utama	
Aksi Aktor	**Reaksi Sistem**
	1. Menampilkan jenis-jenis rubrik (Home, About, How to Use, Archive)
2. Memilih jenis rubrik yang ditampilkan sistem.	
	3. Jika memilih salah satu dari jenis rubrik, maka informasi yang ditampilkan adalah halaman web rubrik yang dipilih.
Skenario Alternatif – Validasi Gagal	
Aksi Aktor	**Reaksi Sistem**
	1. Menampilkan pesan "*Sorry, a connection Error occurred! Please*

	check if you are connected to the internet"
2. Menekan tombol "*Download*"	
	3. Sistem akan terminate.
Kondisi Akhir	Menampilkan informasi sesuai yang dipilih

Tabel 3.7 Skenario *Use Case* lihat *Home*

Identifikasi	
Nama Use Case	Melihat *Home*
Aktor	*User Frontend*
Tujuan	Menampilkan tampilan awal/*home* aplikasi
Keadaan Awal	Sistem menampilkan konten rubrik
Skenario Utama	
Aksi Aktor	**Reaksi Sistem**
1. Menekan tombol "*Home*"	
	2. Menampilkan konten menu yang dapat dipilih untuk ditampilkan.
Kondisi Akhir	Sistem menampilkan menu aplikasi.

Tabel 3.8 Skenario *Use Case* lihat *About*

Identifikasi	
Nama Use Case	Lihat *About*
Aktor	*User Frontend*

Tujuan	Menampilkan informasi tentang aplikasi dan teknologi *augmented reality*
Keadaan Awal	Sistem menampilkan konten rubrik
Skenario Utama	
Aksi Aktor	**Reaksi Sistem**
1. Menekan tombol "*About*"	
	2. Menampilkan informasi *About*
Kondisi Akhir	Sistem menampilkan menu *About*

Tabel 3.9 Skenario *Use Case* lihat *How to Use*

Identifikasi	
Nama Use Case	Melihat *How to Use*
Aktor	*User Frontend*
Tujuan	Menampilkan informasi tentang penggunaan aplikasi
Keadaan Awal	Sistem menampilkan konten rubrik
Skenario Utama	
Aksi Aktor	**Reaksi Sistem**
1. Menekan tombol "*How to Use*"	
	2. Menampilkan informasi bagaimana penggunaan aplikasi yang baik dan benar.
Kondisi Akhir	Sistem menampilkan menu *How to use*

Tabel 3.10 Skenario *Use Case* memilih *Archive*

Identifikasi	
Nama Use Case	Memilih Archive
Aktor	User Frontend
Tujuan	Melihat detail konten arsip yang dipilih
Keadaan Awal	Sistem menampilkan konten rubrik
Skenario Utama	
Aksi Aktor	**Reaksi Sistem**
1. Menekan tombol "*Archive*"	
	2. Menampilkan oprional *Archive* yang dapat dipilih untuk ditampilkan
3. Memilih salah satu konten *archive*	
Skenario Alternatif – Validasi Gagal	
Aksi Aktor	**Reaksi Sistem**
	1. Muncul pesan "There is no application information available. Plaese try again"
2. Memilih konten yang tersedia	
	2. Sistem akan terminate
Kondisi Akhir	Aplikasi menampilkan informasi yang dipilih

1.4.2 Skenario *Use Case* Aplikasi *Backend*

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

Tabel 3.11 Skenario *Use Case* Melakukan *Login*

Identifikasi	
Nama Use Case	Melakukan *Login*
Aktor	*User Backend*
Tujuan	Masuk ke dalam 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 diinputkan
	3. Bila sesuai maka menampilkan halaman web menu utama.
Skenario Alternatif – Validasi Gagal	
Aksi Aktor	**Reaksi Sistem**
	1. Muncul pesan "Ada kesalahan pada *username* dan *password*"
2. Melakukan input *username* dan *password* pada halaman web login admin	
Kondisi Akhir	Halaman menu utama ditampilkan

Tabel 3.12 Skenario *Use Case* Pilih Manajemen Rubrik

Identifikasi	
Nama Use Case	Pilih Manajemen Rubrik
Aktor	*User Backend*

Tujuan	Menampilkan data rubrik secara keseluruhan serta fungsional pengolahan data.
Keadaan Awal	Menampilkan halaman manajemn rubrik
Skenario Utama	
Aksi Aktor	**Reaksi Sistem**
1. Memilih manajemen rubric pada menu navigator	
	2. Menampilkan halaman manajemen rubrik
Kondisi Akhir	Menampilkan data rubrik

Table 3.13 Skenario *Use Case* Tambah Rubrik

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

Skenario Alternatif – Validasi Gagal	
Aksi Aktor	**Reaksi Sistem**
	1. Menampilkan pesan kesalahan sesuai dengan field-field pada pengisian tambah rubrik
2. Melengkapi field-field pada pengisian tambah rubrik	
Kondisi Akhir	Data rubric bertambah sesuai dengan input data

Table 3.14 Skenario *Use Case* Edit Rubrik

Identifikasi	
Nama Use Case	Edit Rubrik
Aktor	User *Backend*
Tujuan	Mengubah data rubrik
Keadaan Awal	Sistem menampilkan halaman pengolahan rubrik
Skenario Utama	
Aksi Aktor	**Reaksi Sistem**
1. Mengisi field-field pada halaman ubah rubrik	
	2. Melakukan proses ubah data pada basis data dan menyimpannya.
Skenario Alternatif – Validasi Gagal	
Aksi Aktor	**Reaksi Sistem**
	1. Menampilkan pesan kesalahan sesuai dengan field-field pada pengisian edit rubrik

2. Melengkapi field-field pada pengisian edit rubrik	
Kondisi Akhir	Perubahan data rubrik sesuai dengan ubah data

Table 3.15 Skenario *Use Case* Hapus Rubrik

Identifikasi	
Nama Use Case	Hapus Rubrik
Aktor	User *Backend*
Tujuan	Menghapus data rubrik
Keadaan Awal	Sistem menampilkan halaman pengolahan rubrik
Skenario Utama	
Aksi Aktor	**Reaksi Sistem**
1. Memilih hapus pada kolom aksi table Data	
	2. Menampilkan pesan konfirmasi "Apakah Anda yakin menghapus Rubrik dengan id <id_rubrik>?
3. Menyetujui penghapusan dengan memilih atau menekan tombol "OK"	
	4. Menghapus data rubrik
	5. Menyimpan hasil perubahan data rubrik
Kondisi Akhir	Perubahan data rubrik sesuai dengan data yang dihapus.
Skenario Alternatif – Validasi Gagal	
Aksi Aktor	**Reaksi Sistem**

	1. Menampilkan pesan kesalahan "Data Tidak bisa dihapus"
2. Menyetujui pesan kesalahan dengan memilih atau menekan tombol "OK"	
Kondisi Akhir	Kembali ke halaman pengolahan rubrik

Table 3.16 Skenario *Use Case* Pilih Manajemen Data AR

Identifikasi	
Nama Use Case	Pilih Manajemen Data AR
Aktor	*User Backend*
Tujuan	Menampilkan table data AR
Keadaan Awal	Menampilkan halaman data AR
Skenario Utama	
Aksi Aktor	**Reaksi Sistem**
1. Memilih manajemen data pada menu navigator	
	2. Menampilkan halaman manajemen data
Kondisi Akhir	Menampilkan halaman manajemen data

Tabel 3.17 Skenario *Use Case* Pilih Tambah Data Animasi

Identifikasi	
Nama Use Case	Pilih Tambah Data Animasi
Aktor	User *Backend*
Tujuan	Menambah data AR animasi
Keadaan Awal	Sistem menampilkan halaman pengolahan data animasi
Skenario Utama	
Aksi Aktor	**Reaksi Sistem**
1. Memilih "Tambah" pada pilihan halaman data	
	2. Menampilkan halaman pengisian tambah data
3. Mengisi field-field pada pengisian tambah data	
	4. Melakukan proses tambah data pada basis data
	5. Menyimpan hasil tambah data
Skenario Alternatif – Validasi Gagal	
Aksi Aktor	**Reaksi Sistem**
	1. Menampilkan pesan kesalahan sesuai dengan field-field pada pengisian tambah data animasi
2. Melengkapi field-field pada pengisian tambah data	
Kondisi Akhir	Data AR bertambah sesuai dengan input data

Table 3.18 Skenario *Use Case* Edit Data Animasi

Identifikasi

Nama Use Case	Edit Data Animasi
Aktor	User *Backend*
Tujuan	Mengubah data AR Animasi
Keadaan Awal	Sistem menampilkan halaman pengolahan data
Skenario Utama	
Aksi Aktor	**Reaksi Sistem**
1. Memilih "Edit" pada kolom aksi table Data	
	2. Menampilkan field-field data
3. Mengisi field-field pada halaman ubah data AR	
	4. Melakukan proses ubah data pada basis data dan menyimpannya.
	5. Menyimpan hasil perubahan data rubrik
Skenario Alternatif – Validasi Gagal	
Aksi Aktor	**Reaksi Sistem**
	1. Menampilkan pesan kesalahan sesuai dengan field-field pada pengisian edit data animasi
2. Melengkapi field-field pada pengisian edit data	
Kondisi Akhir	Perubahan data AR sesuai dengan ubah data

Table 3.19 Skenario *Use Case* Hapus Data Animasi

Identifikasi	
Nama Use Case	Hapus Data Animasi

Aktor	User *Backend*
Tujuan	Menghapus data AR Animasi
Keadaan Awal	Sistem menampilkan halaman pengolahan data
Skenario Utama	
Aksi Aktor	**Reaksi Sistem**
1. Memilih hapus pada kolom aksi table Data	
	2. Menampilkan pesan konfirmasi "Apakah Anda yakin menghapus data <type_ar> dengan id <id_data>?
3. Menyetujui penghapusan dengan memilih atau menekan tombol "OK"	
	4. Menghapus data ar
	5. Menyimpan hasil perubahan data ar
Kondisi Akhir	Perubahan data ar sesuai dengan data yang dihapus.

2. Class Diagram

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

Tabel 3.20 Deskripsi *Class Diagram ARMagazine*

Backend Application		
Class	**Jenis Kelas**	**Deskripsi**
Manajemen_rubrik	*Interface*	Class yang berisikan tampilan data rubrik
Manajemen_data	*Interface*	Class yang berisikan tampilan data AR
Admin	*Interface*	Class yang berisikan method-method untuk fungsionalitas tampilan utama aplikasi yaitu menampilkan *Home*.
Login	*Interface*	Class yang berisikan tampilan *login web*
Cadmin	*Control*	Class yang berisikan *method-method* untuk mengontrol *class interface*
M_admin	*Control*	Class yang berisikan *method* untuk mengontrol *class control* Cadmin
M_rubrik	*Control*	Class yang berisikan *method* untuk mengontrol *class control* Cadmin
M_data	*Control*	Class yang berisikan *method* untuk mengontrol *class control* Cadmin
Main	*Control*	Class yang berisikan *menthod-method* untuk mengontrol *class* paket fileAR
ARPV3DCamera	*Control*	Class yang mengaktifkan fungsionalitas kamera pada *webcam* untuk ditampilkan

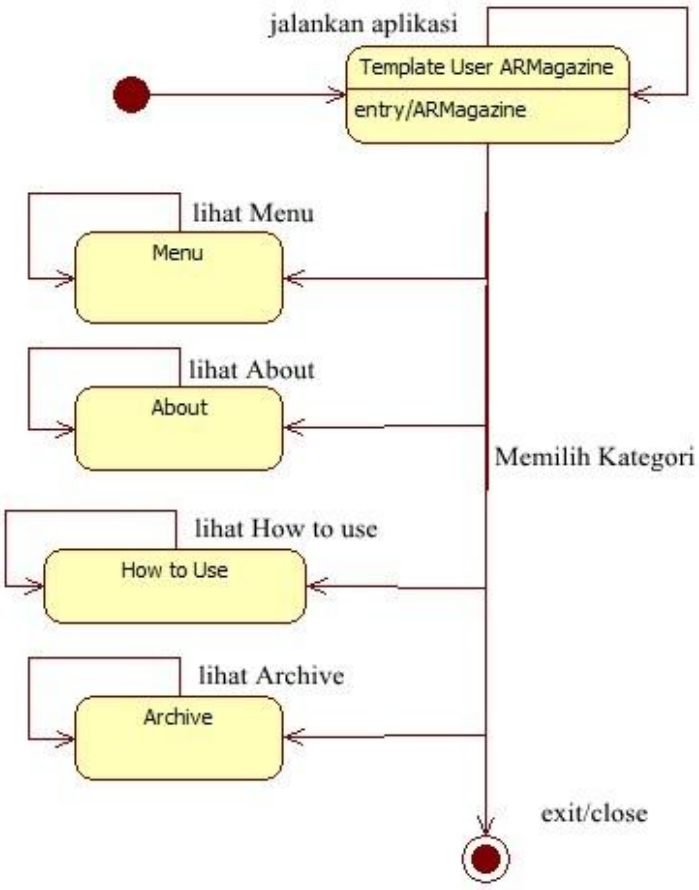
MarkerASS	*Control*	Class yang menginisialisasikan fungsionalitas *marker* untuk ditampilkan
ModelDAE	*Control*	Class yang menginisialisasikan fungsionalitas model bentuk untuk ditampilkan
bs_Admin	*Entity*	Class yang berisikan data admin
bs_AR	*Entity*	Class yang berisikan data AR
bs_Rubrik	*Entity*	Class yang berisikan data rubrik
Frontend Application		
Class	**Jenis Kelas**	**Deskripsi**
Template	*Interface*	Class yang berisikan method-method untuk fungsionalitas tampilan utama aplikasi yaitu menampilkan *Home*
Vabout	*Interface*	Class yang berisikan method untuk fungsionalitas tampilan halaman *about*
Vhowtouse	*Interface*	Class yang berisikan method untuk fungsionalitas tampilan halaman *how to use*
Varchive	*Interface*	Class yang berisikan method untuk fungsionalitas tampilan halaman *archive*
Ctemplate	*Control*	Class yang berisikan method-method untuk mengontrol *class interface template*
About	*Control*	Class yang berisikan method-method untuk mengontrol *class interface about*

Howtouse	*Control*	Class yang berisikan method-method untuk mengontrol *class interface howtouse*
Carchive	*Control*	Class yang berisikan method-method untuk mengontrol *class interface* varchive
M_archive	*Control*	Class yang berisikan method-method untuk mengontrol *class control* Archive

3. State machine Diagram

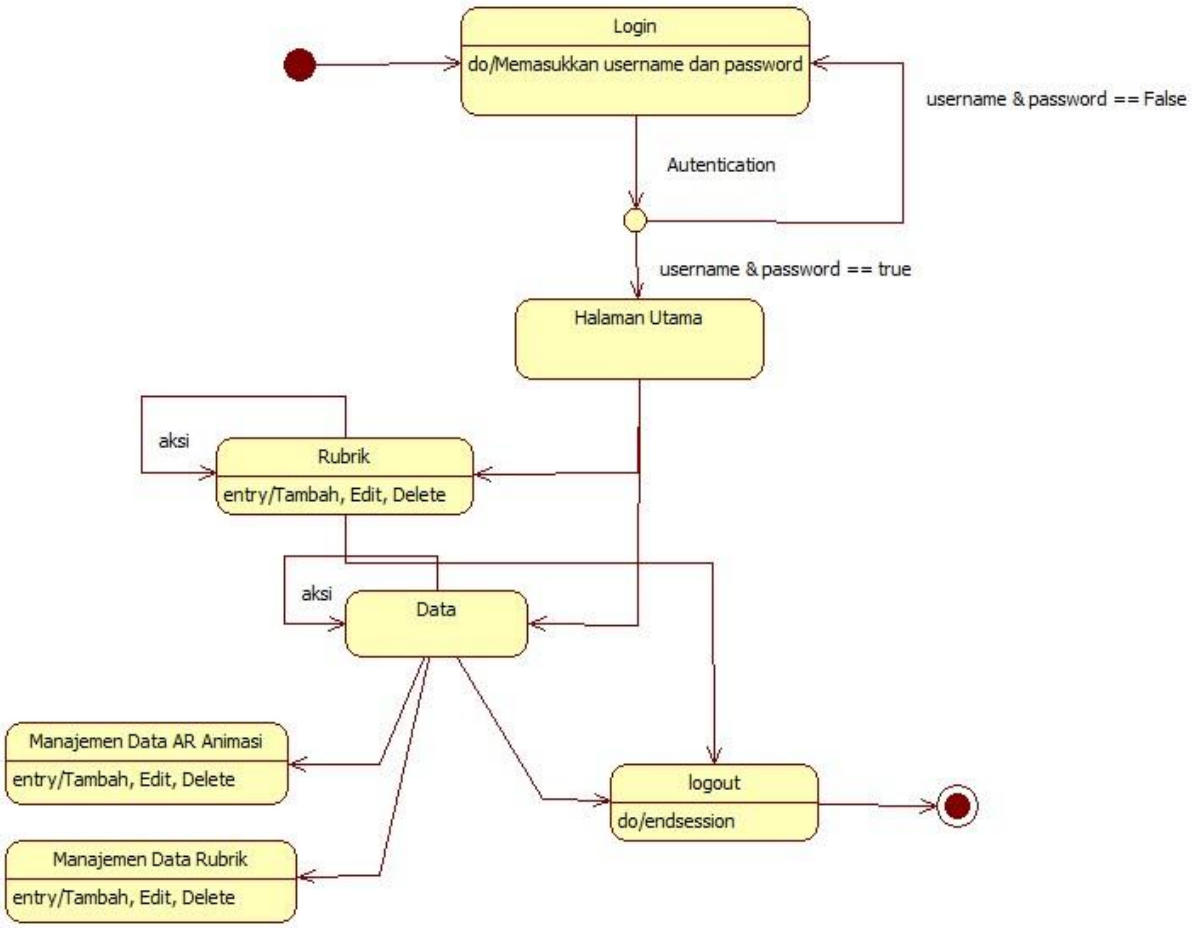
State machine Diagram mengekspresikan perilaku sebagi sebuah progresif (perkembangan) melalui sekumpulan *state, triggered by events,* dan *action* yang mungkin terjadi. *State machine diagram* juga biasa dikenal sebagai *behavioural state diagram*. *State machine diagram* biasanya digunakan untuk mendekripsikan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

3.1 *State Machine Diagram* Aplikasi *Frontend*



Gambar 3.14 *State Machine Diagram* Aplikasi *Frontend*

3.2 *State Machine Diagram* Aplikasi *Backend*

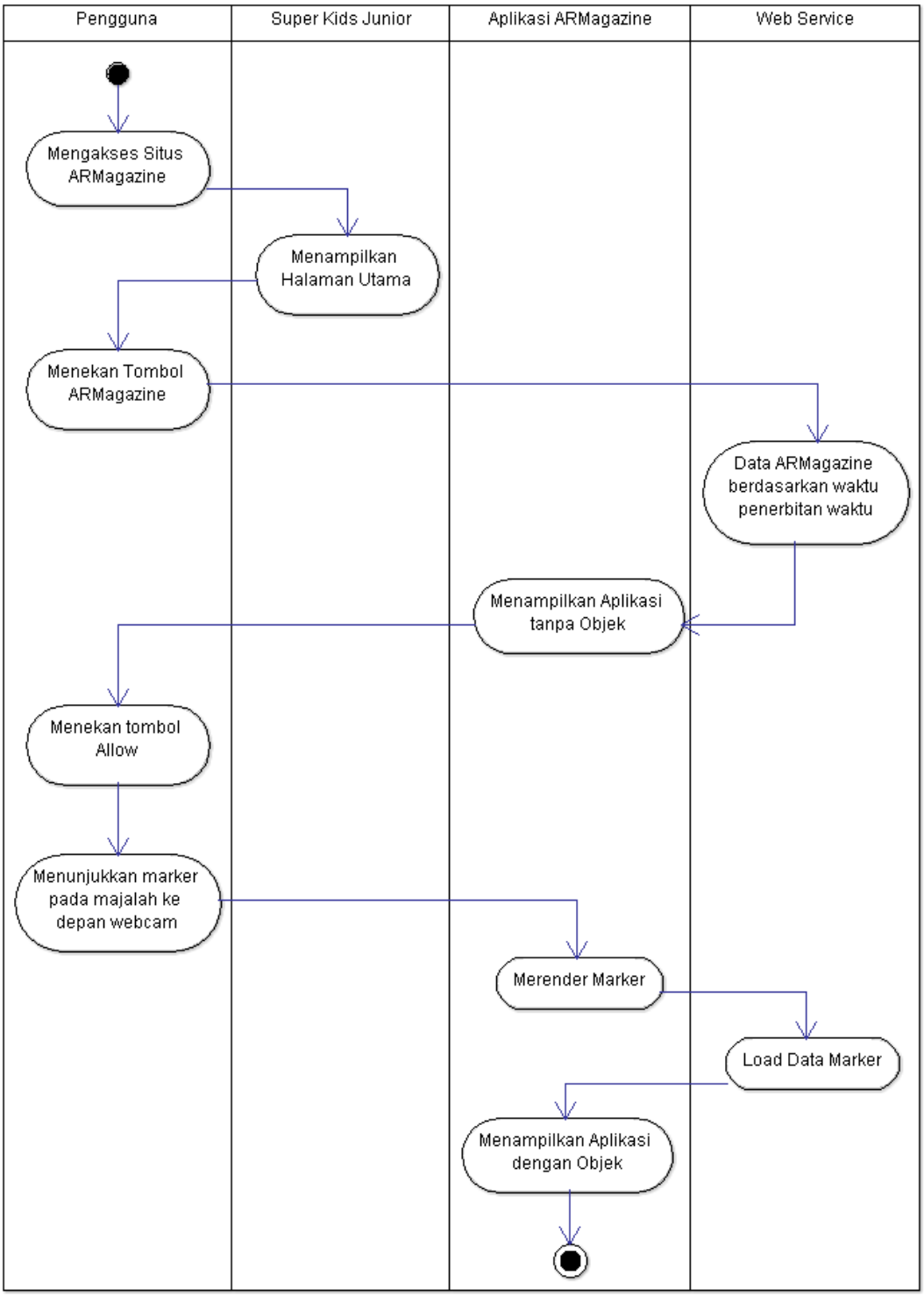


Gambar 3.15 *State Machine Diagram* Aplikasi *Backend*

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.

4.1 *Activity Diagram* Menggunakan *ARMagazine*

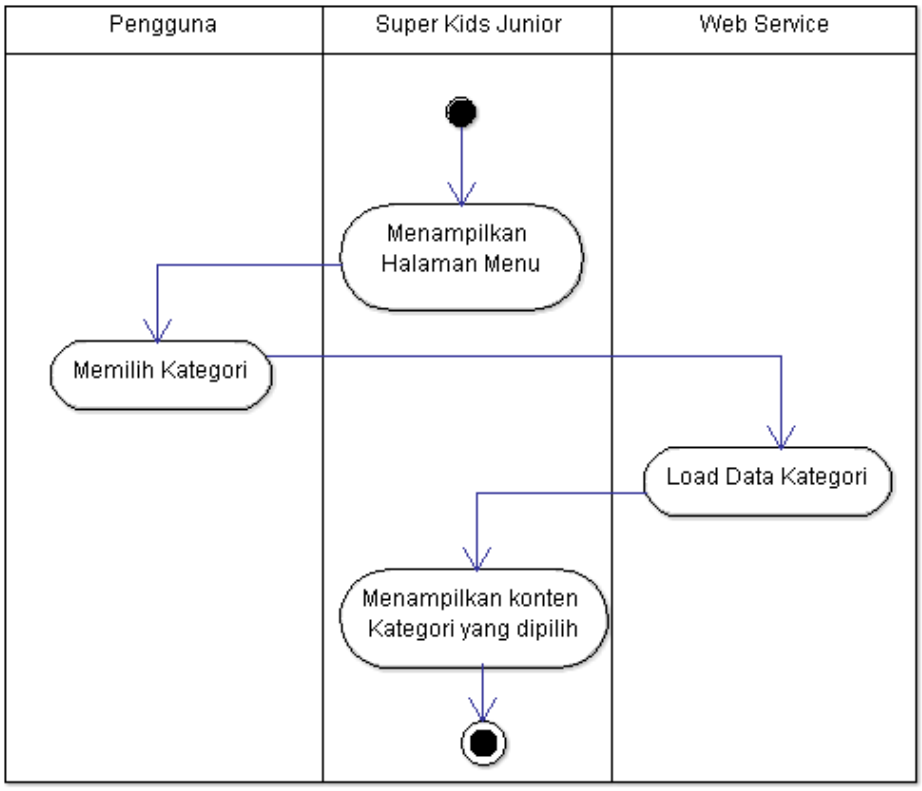


Gambar 3.16 *Activity Diagram* Menggunakan *ARMagazine*

Gambar 3.16 menggambarkan aktivitas yang terjadi pada aplikasi saat *user frontend* pertama kali menggunakan aplikasi. Dimana pada

saatpenggunaan menjalankan aplikasi *user frontend* harus menunjukkan gambar Koran yang sudah diimplementasi metode makerless ke depan webcam agar objek yang ada pada gambar tersebut muncul.

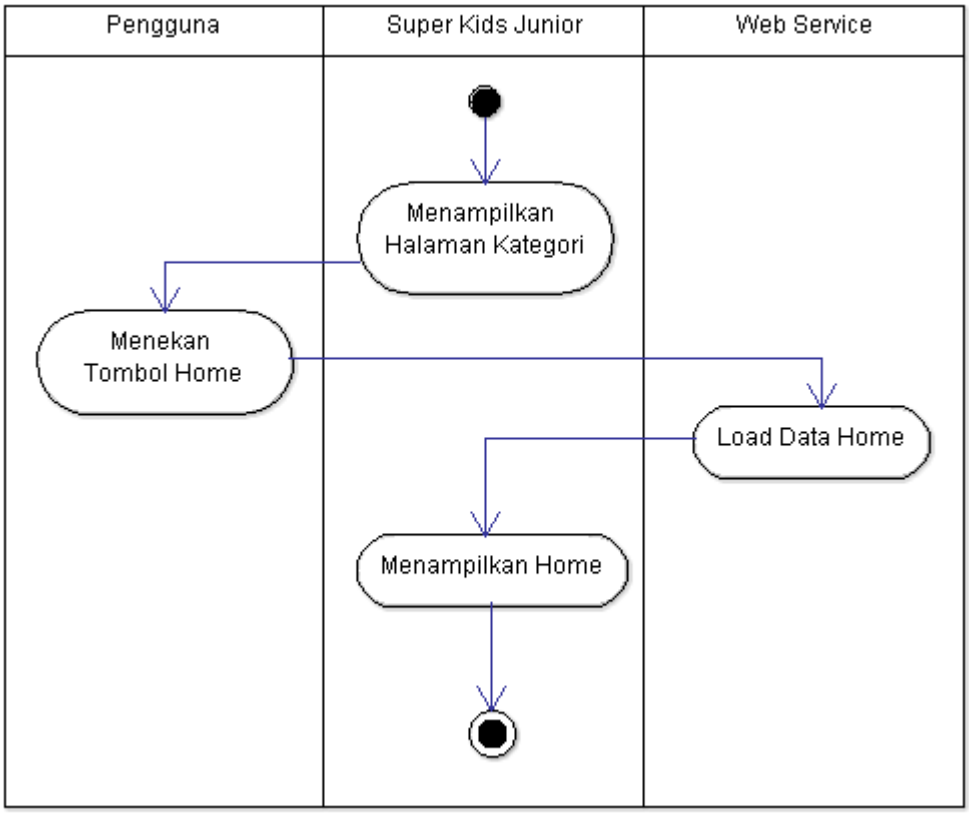
4.2 *Activity Diagram* Pilih Kategori



Gambar 3.17 *Activity Diagram* Pilih Rubrik

Gambar 3.17 menggambarkan aktivitas ketika user frontend memilih rubrik. *User frontend* memilih salah satu rubrik, selanjutnya sistem akan me-load data pada server (*web server*) untuk menampilkan konten yang sesuai dengan jenis rubrik yang dipilih.

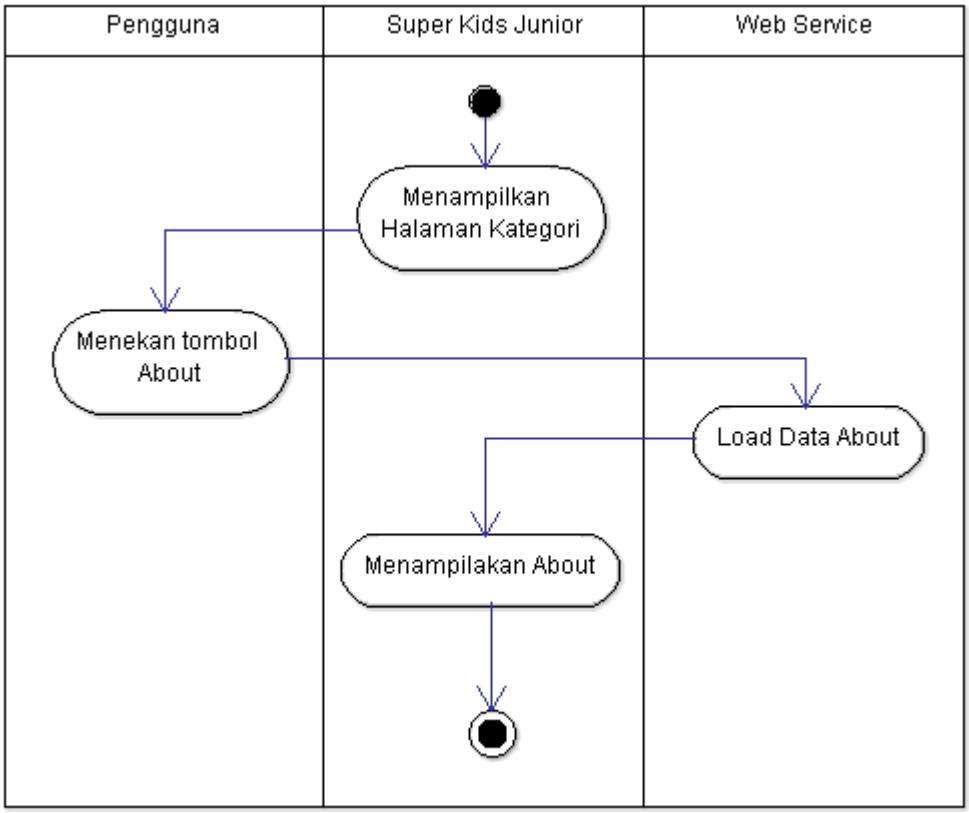
4.3 *Activity Diagram Home*



Gambar 3.18 *Activity Diagram Home*

Gambar 3.18 menggambarkan aktivitas ketika *user frontend* memilih *home*. *User frontend* menekan tombol *home*, selanjutnya sistem akan me-*load* data *home* pada server (*web server*) untuk menampilkan rubrik *home*. Pada rubrik *home* ini menjadi halaman utama pada saat membuka situs iml.com. *Home* ini berisi tentang aplikasi ARMagazine yang bisa langsung digunakan sesuai dengan hari pada saat situs dibuka.

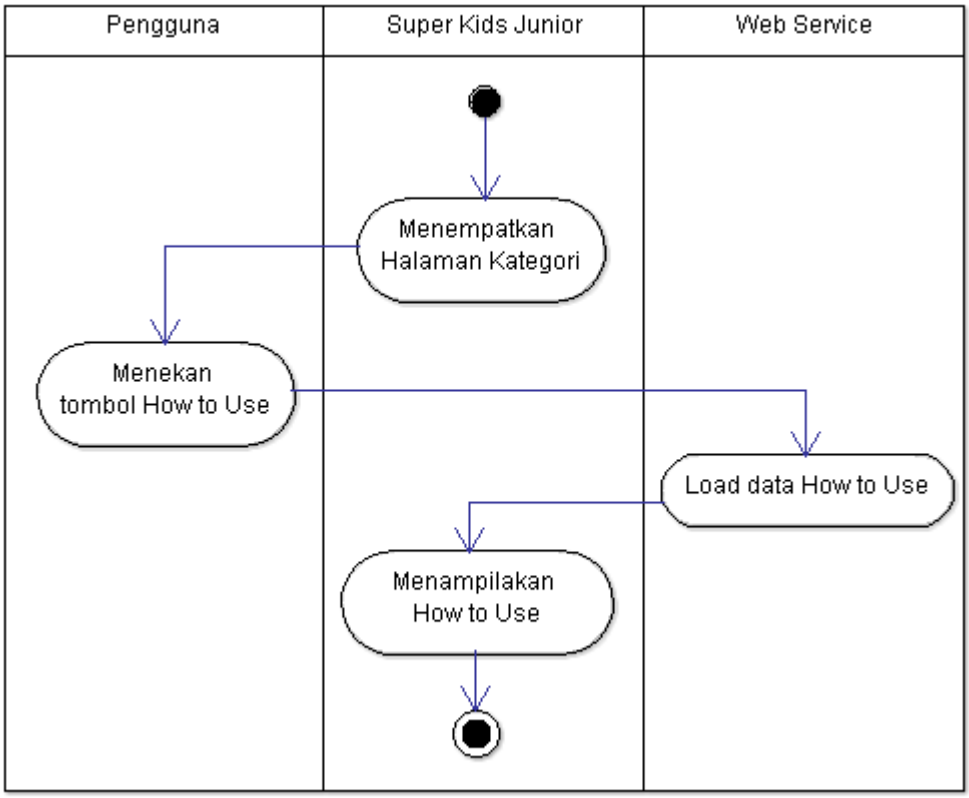
4.4 *Activity Diagram* Lihat *About*



Gambar 3.19 *Activity Diagram* Lihat *About*

Gambar 3.19 menggambarkan aktivitas ketika *user frontend* memilih *about. User frontend* menekan tombol *about*, selanjutnya sistem akan me-*load* data *about* pada server (*web server*) untuk menampilkan rubrik *about*. Pada rubrik *about* berisikan halaman informasi ringkas tentang aplikasi dan teknologi *augmented reality* yang digunakan.

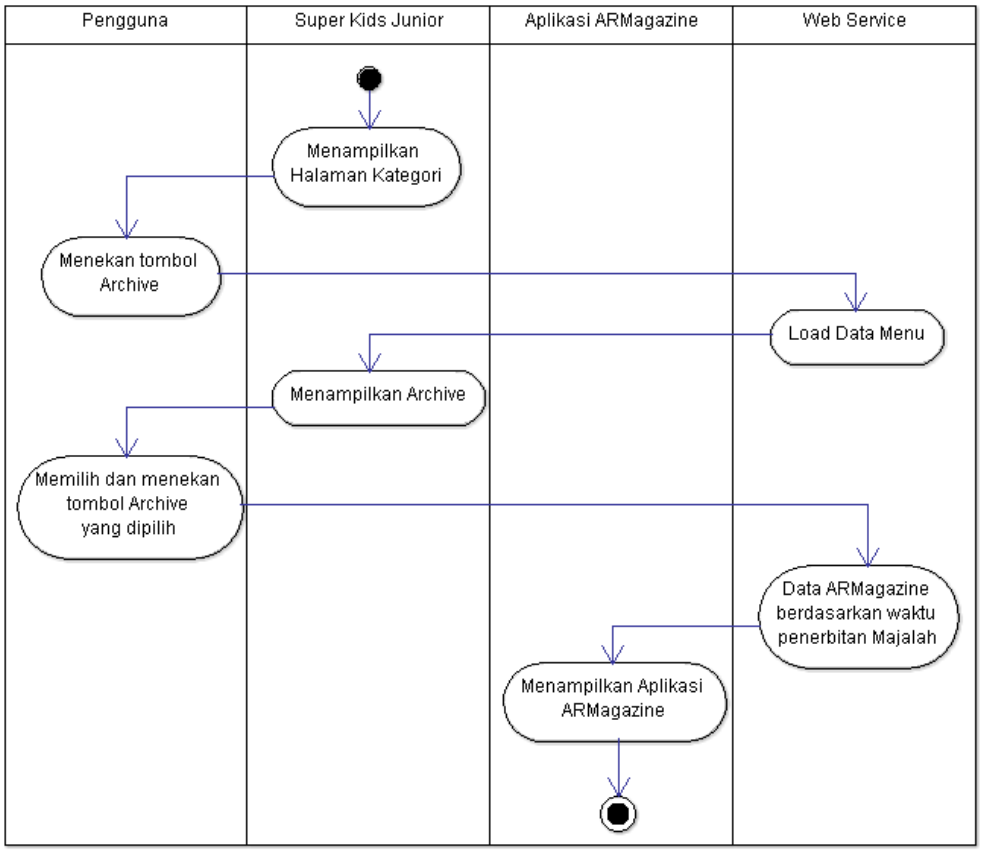
4.5 *Activity Diagram* Lihat *How to use*



Gambar 3.20 *Activity Diagram* Lihat *How to use*

Gambar 3.20 menggambarkan aktivitas ketika *user frontend* memilih *how to use*. *User frontend* menekan tombol *How to use*, selanjutnya sistem akan me-*load* data *How to use* pada server (*web server*) untuk menampilkan rubrik *how to use*. Pada rubrik *How to use* menjelaskan tentang cara penggunaan aplikasi secara baik dan benar.

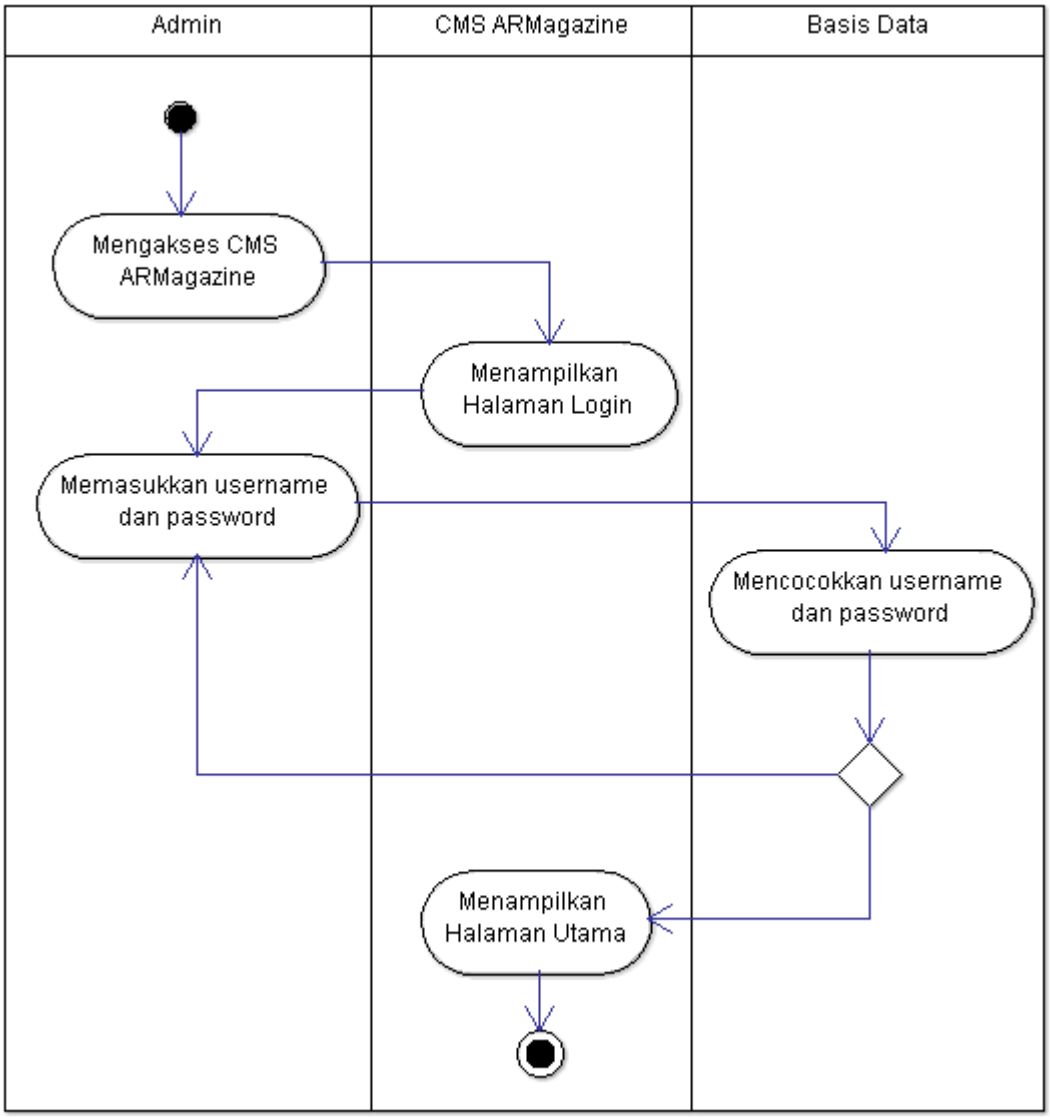
4.6 *Activity Diagram* Memilih *Archive*



Gambar 3.21 *Activity Diagram* memilih *Archive*

Gambar 3.21 menggambarkan aktivitas ketika *user frontend* memilih *archive. User frontend* menekan tombol *archive*, selanjutnya sistem akan me-*load* data *archive* pada server (*web server*) untuk menampilkan rubrik *archive*, kemudian *user frontend* dapat memilih *archive* sesuai dengan tanggal terbitnya majalah yang dimiliki. Setelah proses pemilihan *archive* dengan menekan tombol *archive* yang dipilih kemudian sistem akan me-*load* data untuk ditampilkannya aplikasi *ARMagazine*.

4.7 *Activity Diagram Login* CMS

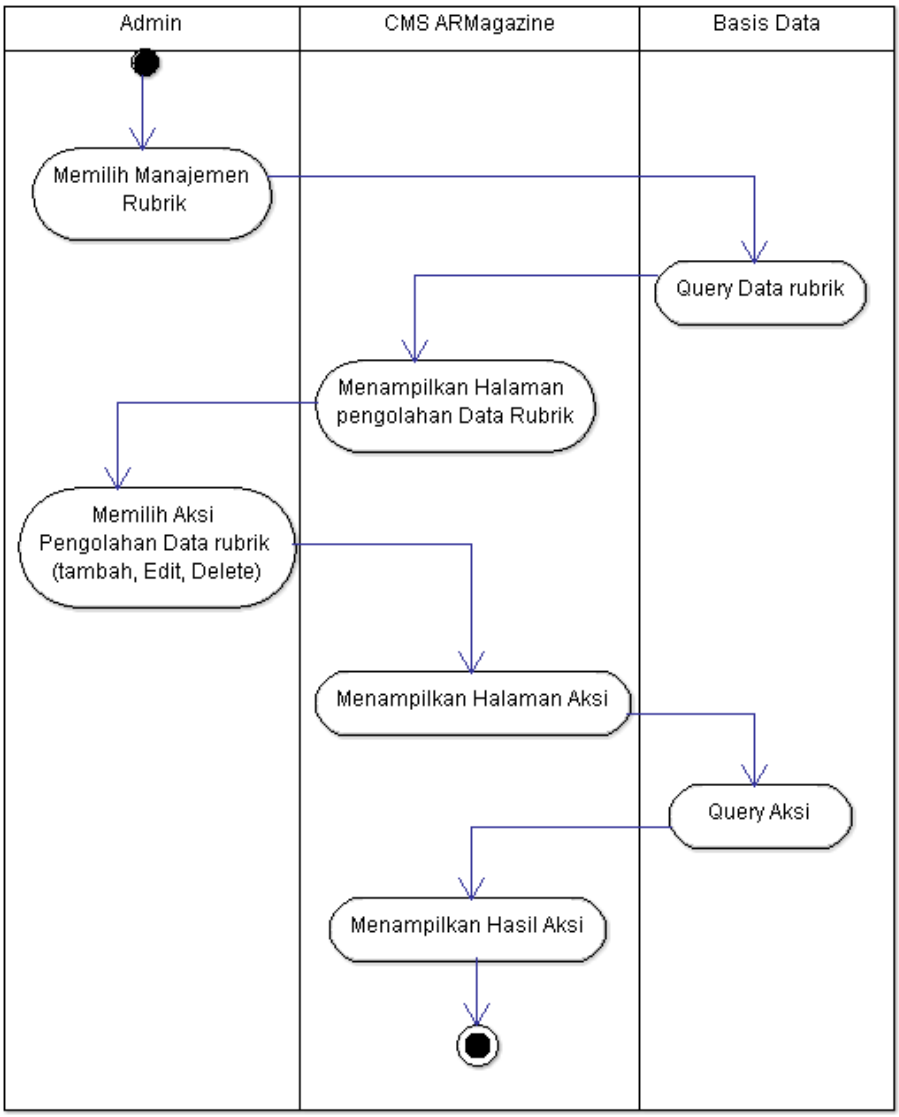


Gambar 3.22 *Activity Diagram Login* CSM

Gambar 3.22 menggambarkan aktivitas *login* untuk masuk ke halaman utama CSM. *User Backend* mengakses halaman CSM, kemudian CSM akan menampilkan halaman *login*, kemudian *user backend* melakukan input *username* dan *password*. *Server* akan melakukan verifikasi, jika *username* dan *password* tidak sesuai maka *user backend* akan diarahkan kembali ke halaman *login* untuk melakukan

input *username* dan *password* yang valid. Jika *username* dan *password* yang input valid, maka akan ditampilkan halaman utama web CSM.

4.8 *Activity Diagram* CMS Rubrik

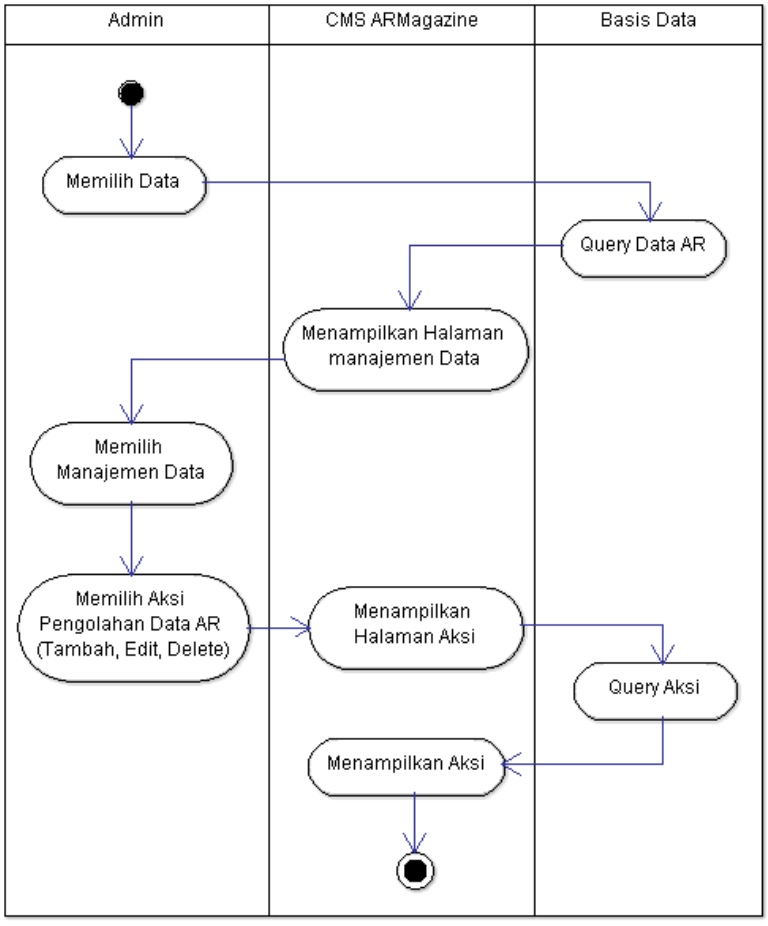


Gambar 3.23 *Activity Diagram* CSM Rubrik

Gambar 3.23 menggambarkan aktivitas pengolahan data rubrik. Proses ini berjalan setelah *user backend* berhasil melakukan *login*. *User*

backend memilih menu rubrik, selanjutnya CSM akan menampilkan halaman rubrik. *User backend* dapat mengelola data rubrik dengan menggunakan aksi Tambah, Ubah atau hapus. CMS akan menampilkan halaman berdasarkan aksi yang dilakukan *user backend*, saat bersamaan *query* akan dipanggil dan dieksekusi pada *database server* kemudian hasilnya ditampilkan pada web CMS.

4.9 *Activity Diagram* CMS Data



Gambar 3.24 *Activity Diagram* CMS Data

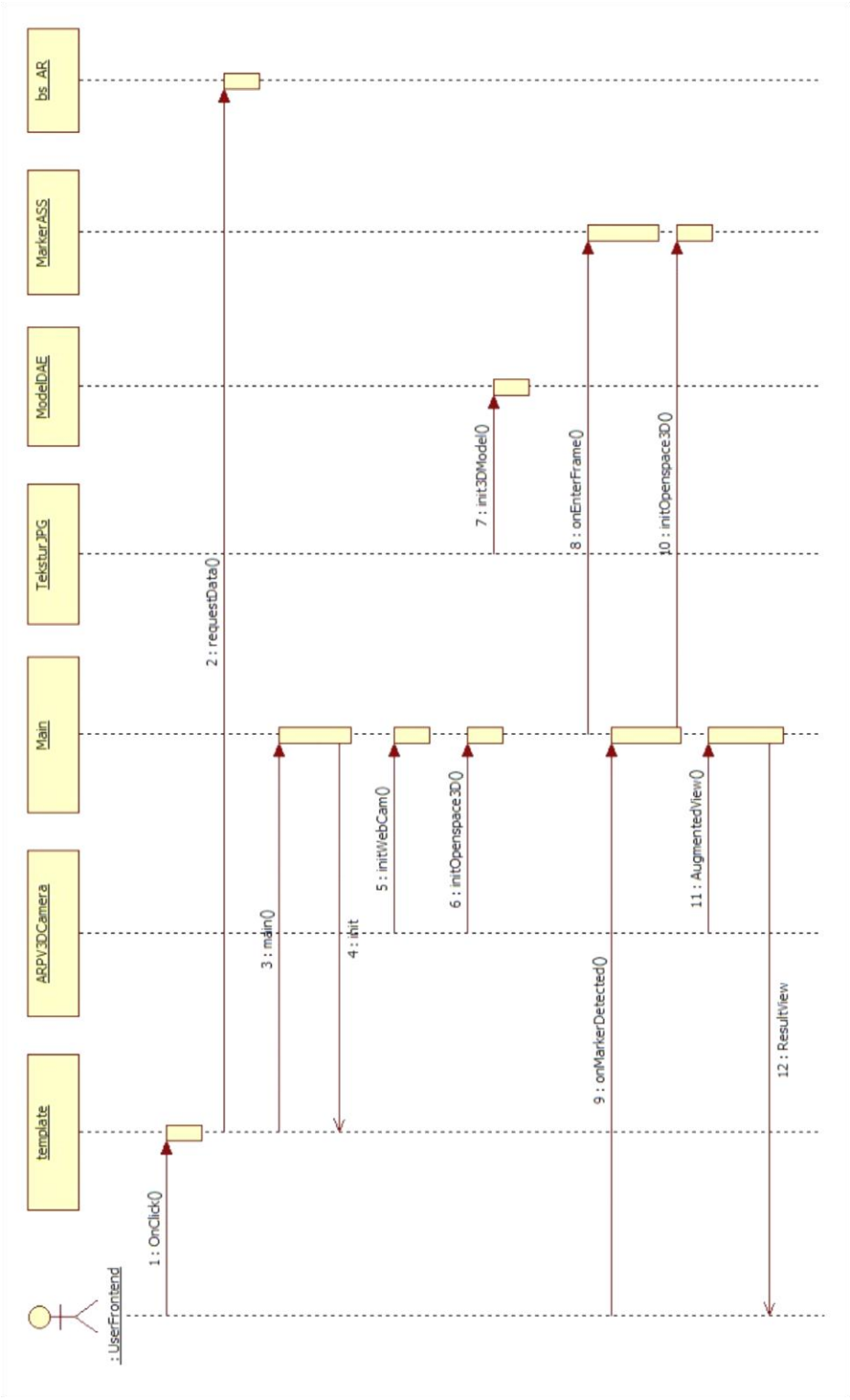
Gambar 3.24 mengambarkan aktivitas pengolahan data ar. Proses ini berjalan setelah *user backend* berhasil melakukan login. *User backend* memilih menu data, selanjutnya CMS akan menampilkan halaman data

ar manajemen yang ada. Kemudian *User backend* memilih manajemen data dan *user backend* dapat mengelola data ar dengan menggunakan aksi Tambah, Ubah atau hapus. CMS akan menampilkan halaman berdasarkan aksi yang dilakukan *user backend*, saat bersamaan *query* akan dipanggil dan dieksekusi pada database server kemudian hasilnya ditampilkan pada web CMS.

5. Sequence Diagram

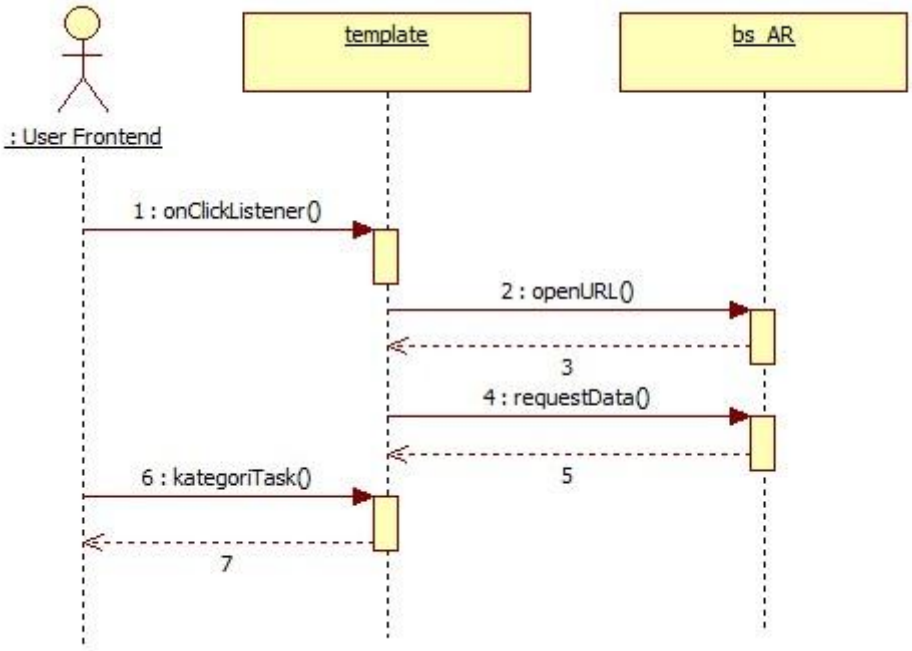
Sequence diagram menggambarkan perilakuk pada sebuah *scenario*. Diagram ini menunjukkan sejumlah contoh objek dan *message* (pesan) yang diletakkan diantara objek-objek ini di dalam *use case*. Komponen utama *sequence diagram* terdiri atas objek yang dituliskan dengan kotak segiempat. *Message* diwakili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

5.1 *Sequence Diagram* menggunakan *ARMagazine*



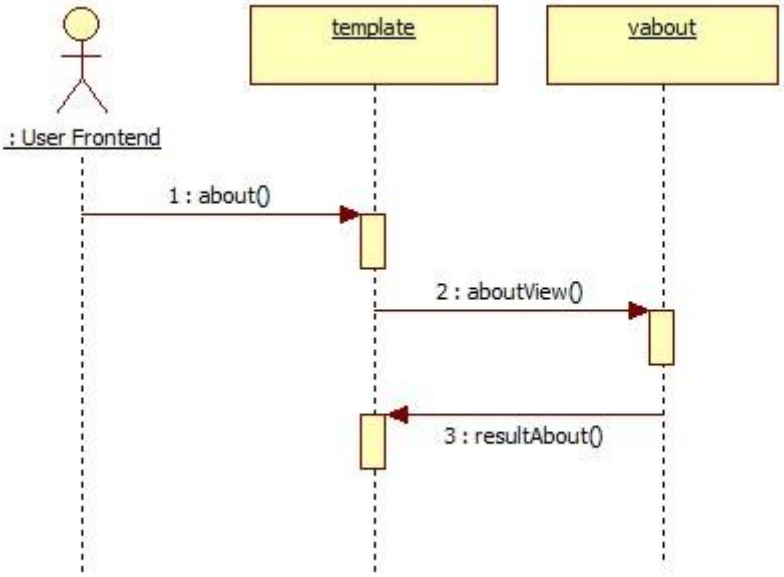
Gambar 3.25 *Sequence Diagram* Menggunakan *ARMagazine*

5.2 *Sequence Diagram* Pilih Kategori



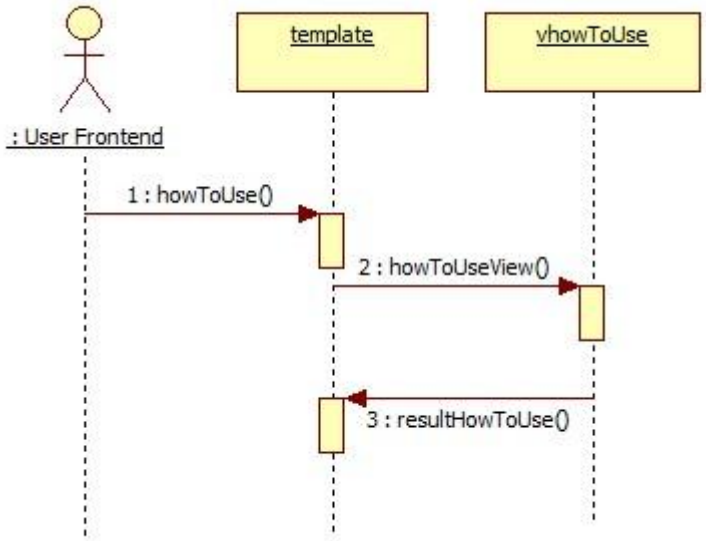
Gambar 3.26 *Sequence Diagram* Pilih Kategori

5.3 *Sequence Diagram* Lihat About



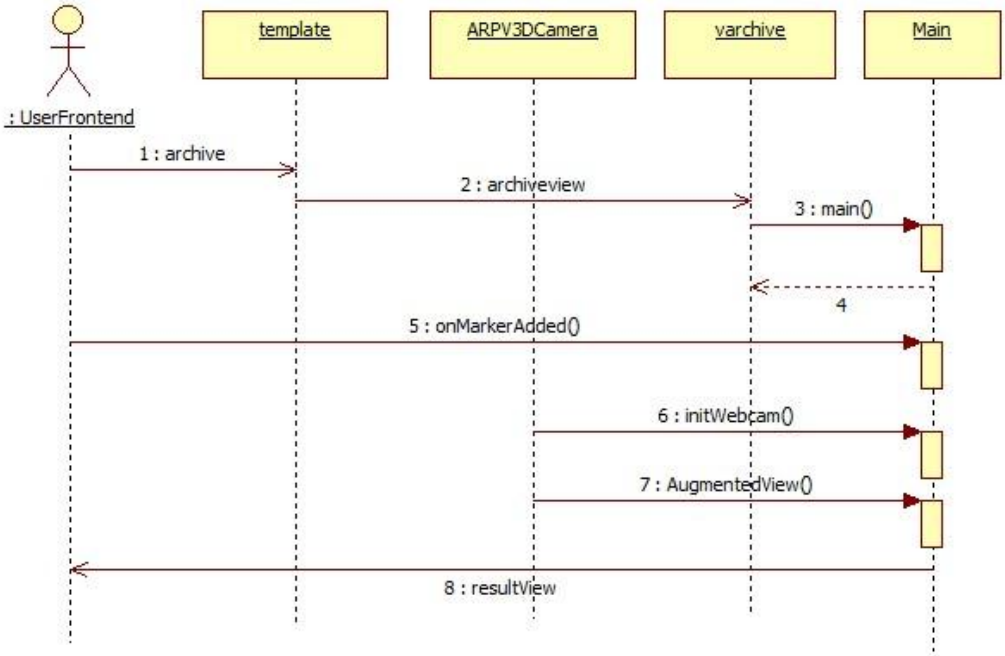
Gambar 3.27 *Sequence Diagram* Pilih About

5.4 *Sequence Diagram* Lihat *How to use*



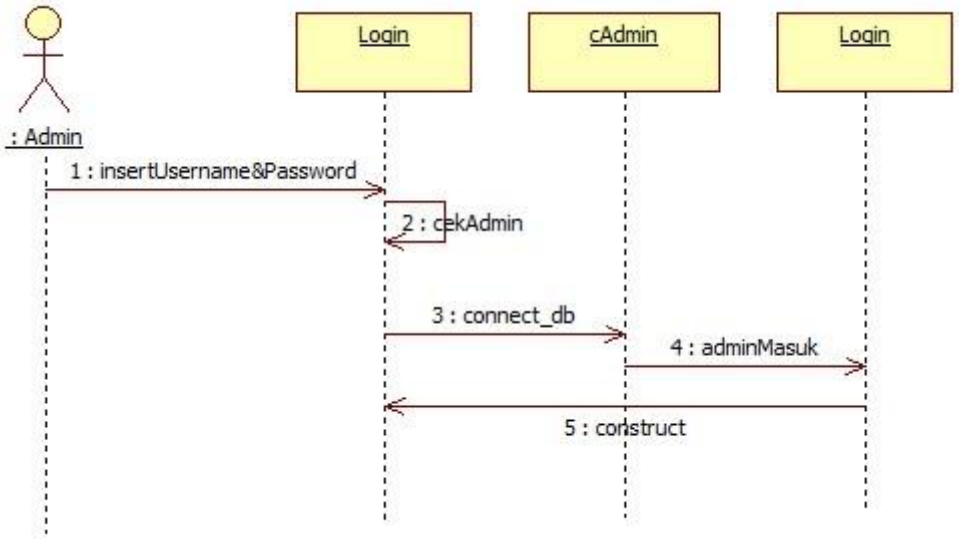
Gambar 3.28 *Sequence Diagram* lihat *How to use*

5.5 *Sequence Diagram* Memilih *Archive*



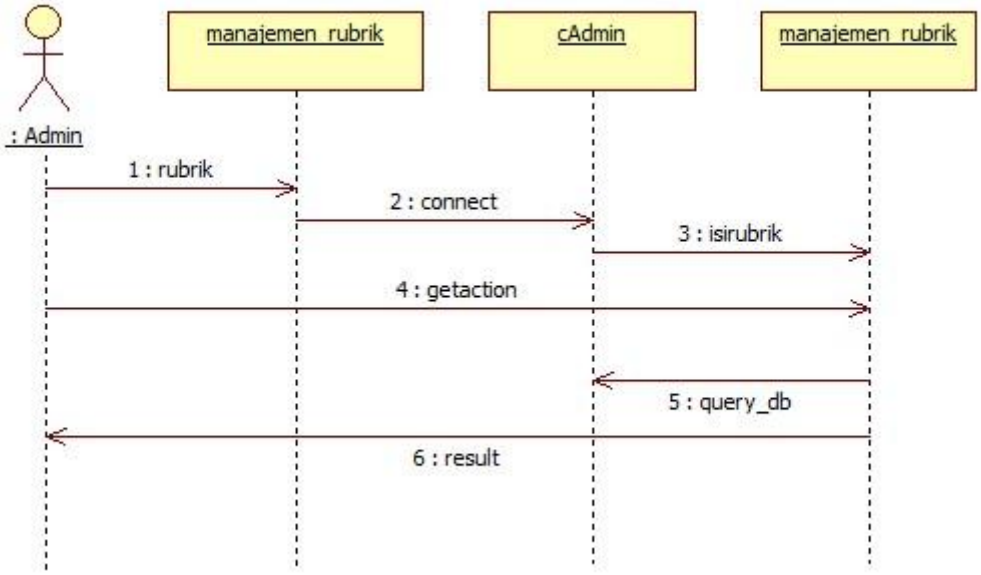
Gambar 3.29 *Sequence Diagram* Pilih *Archive*

5.6 *Sequence Diagram Login CMS*



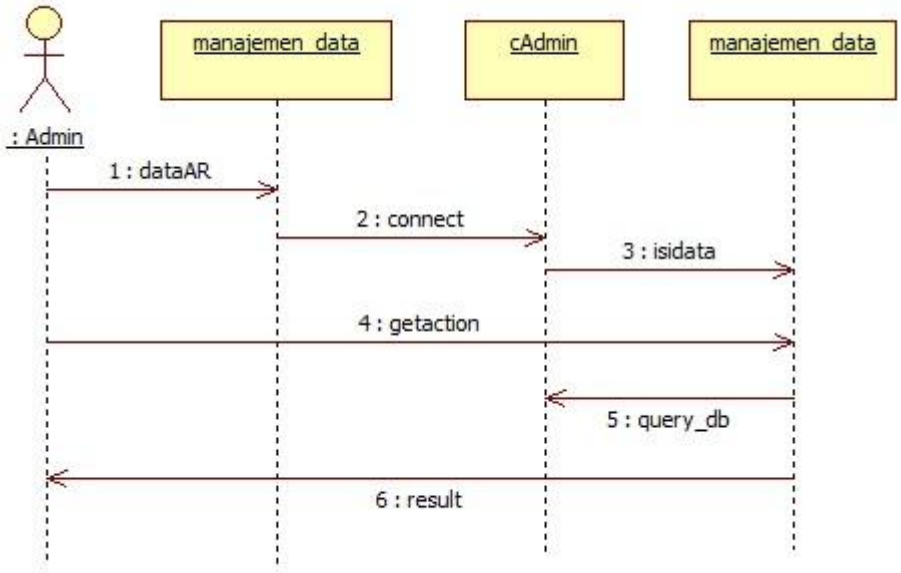
Gambar 3.30 *Sequence Diagram Login CMS*

5.7 *Sequence Diagram* Rubrik



Gambar 3.31 *Sequence Diagram* Rubrik

5.8 *Sequence Diagram* Data

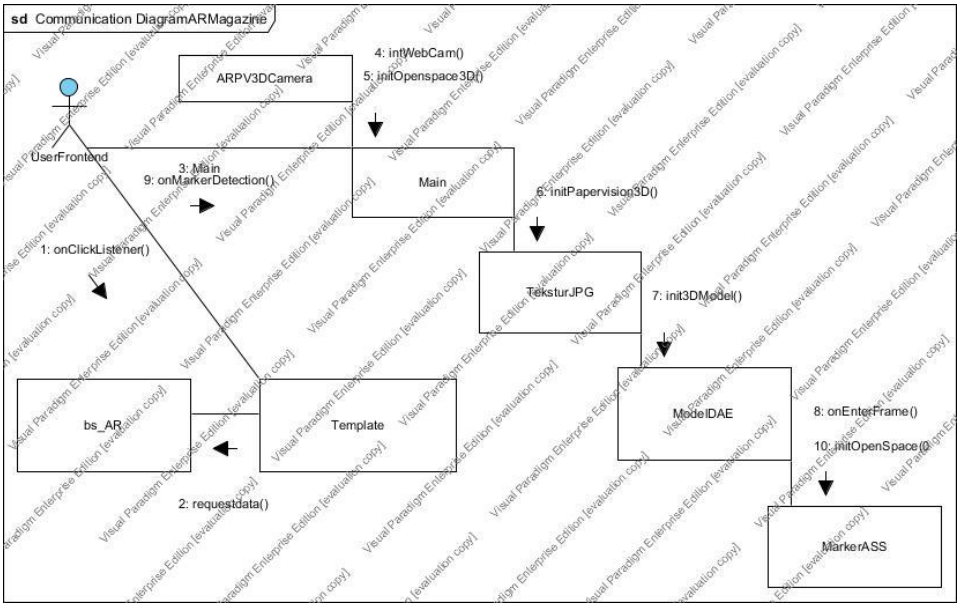


Gambar 3.32 *Sequence Diagram* Data

6. Communication Diagram

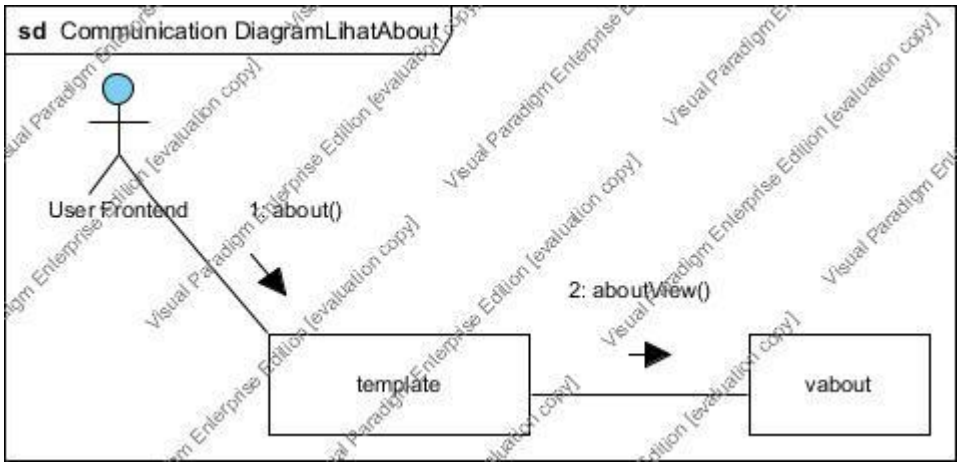
Diagram komunikasi juga digunakan untuk memodelkan perilaku dinamis dari *use case*. Bila dibandingkan dengan *Diagram Sequence*, Diagram komunikasi lebih terfokus pada menampilkan kolaborasi dari objek bukan urutan waktu. *Communication Diagram* merupakan pengembangan dari *collaboration diagram* pada UML versi sebelumnya.

6.1 *Communication Diagram* menggunakan *ARMagazine*



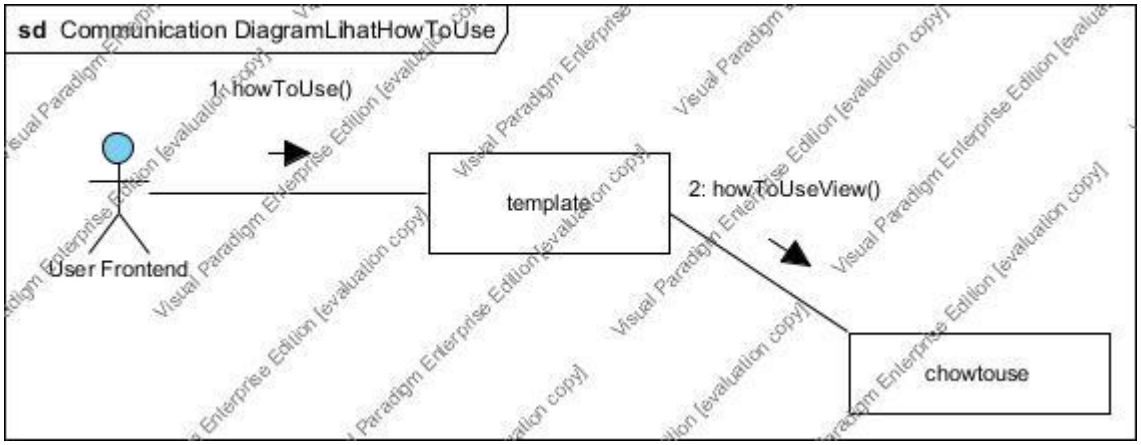
Gambar 3.33 *Communication Diagram ARMagazine*

6.2 *Communication Diagram* Lihat About



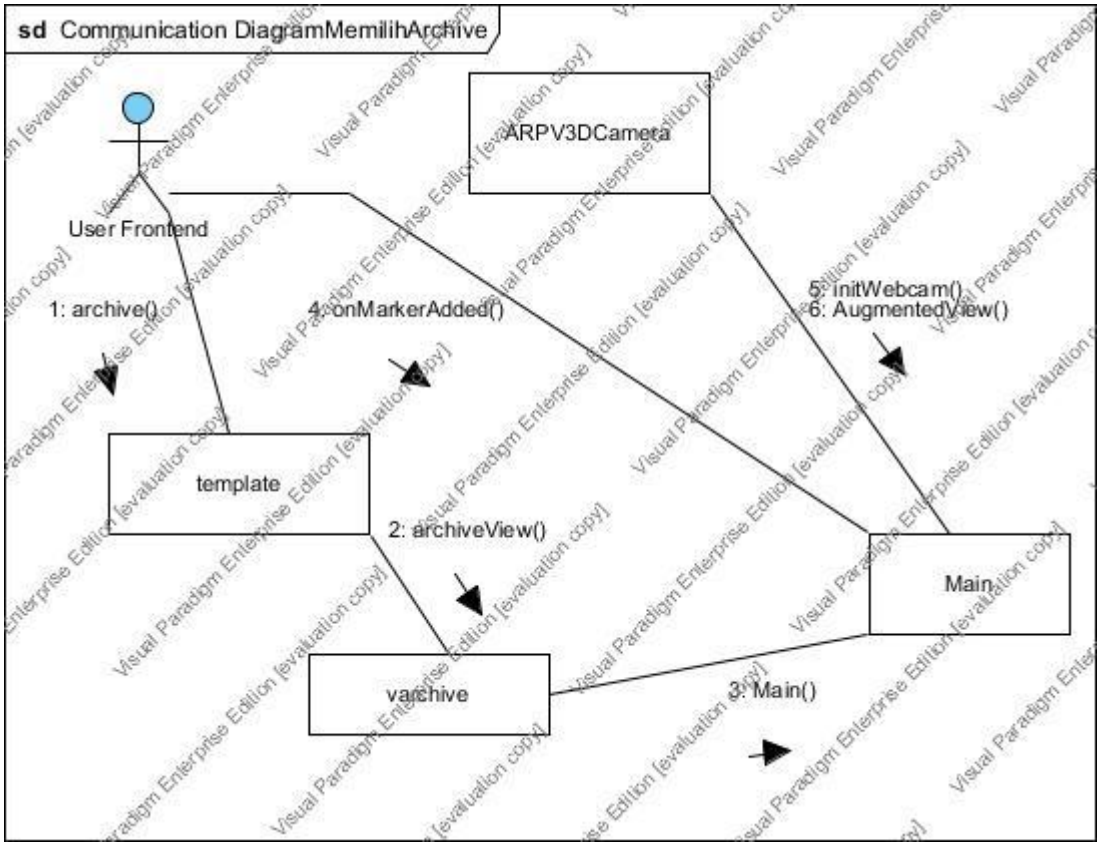
Gambar 3.34 *Communication Diagram* Lihat *About*

6.3 *Communication Diagram* Lihat *How to use*



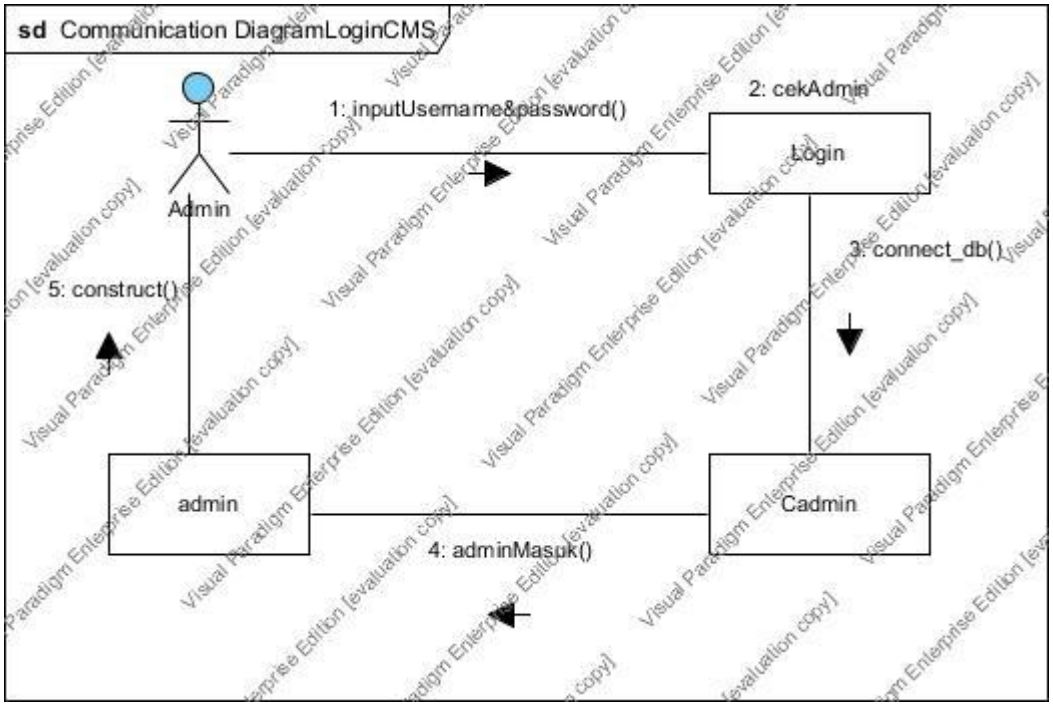
Gambar 3.35 *Communication Diagram* Lihat *How to Use*

6.4 *Communication Diagram* Memilih *Archive*



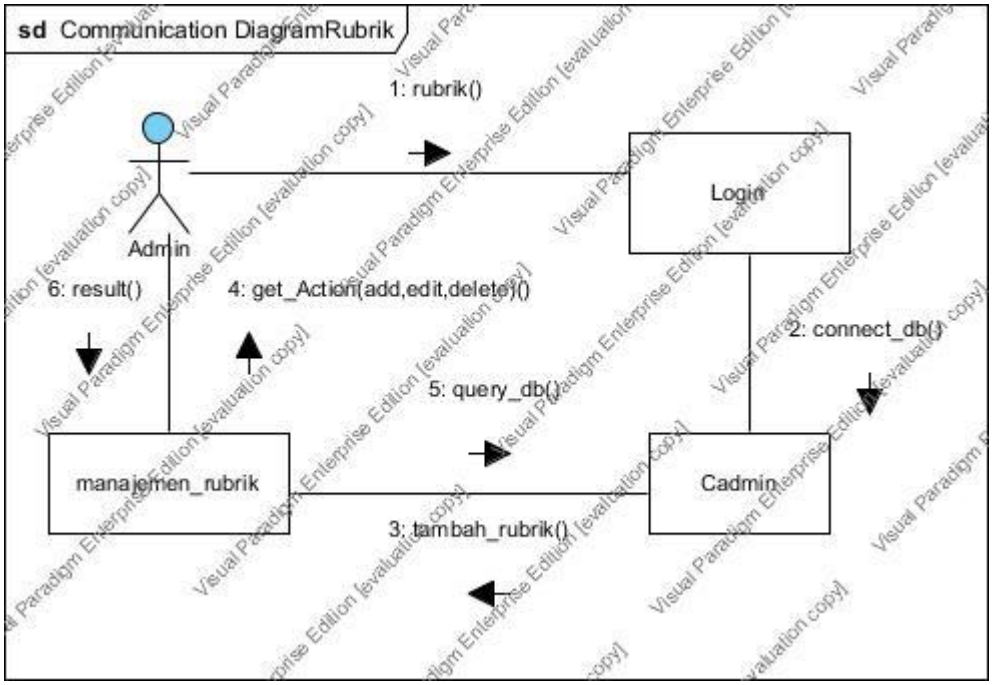
Gambar 3.36 *Communication Diagram* Lihat *Archive*

6.5 *Communication Diagram Login CMS*



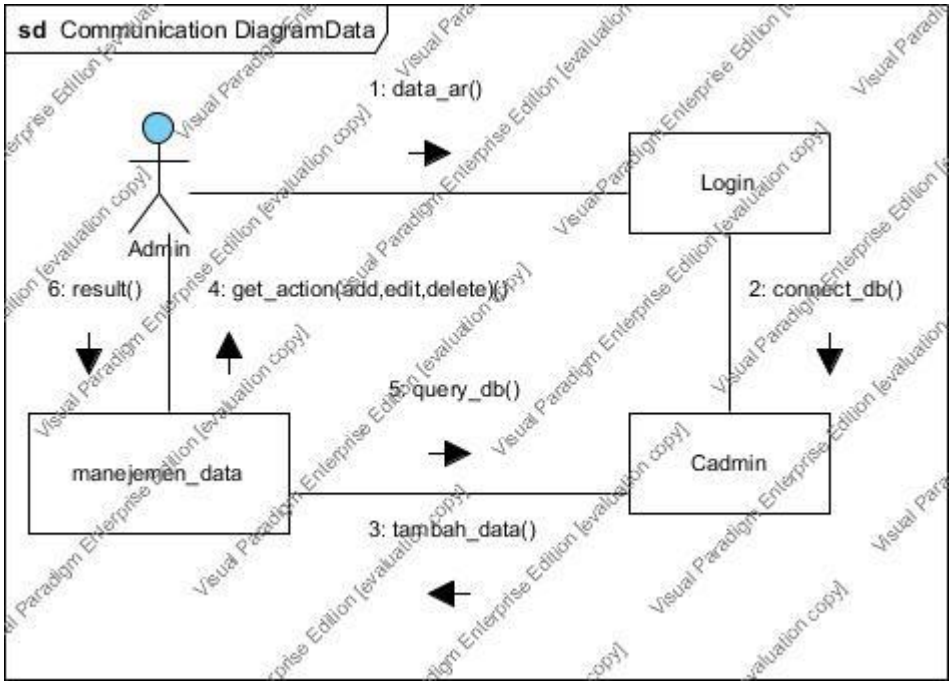
Gambar 3.37 *Communication Diagram Login CMS*

6.6 *Communication Diagram* Rubrik



Gambar 3.38 *Communication Diagram* Rubrik

6.7 *Comminication Diagram* Data



Gambar 3.39 *Communication Diagram* Data

3.2 Perancangan Sistem

Perancangan akan dimulai setelah tahap analisi terhadap sistem selesai dilakukan. Perancangan dapat didefinisikan sebagai penggambaran, perencanaan dan pembuatan sketsa atau pengaturan dari beberapa elemen yang terpisah ke dalam satu kesatuan yang utuh dan berfungsi.

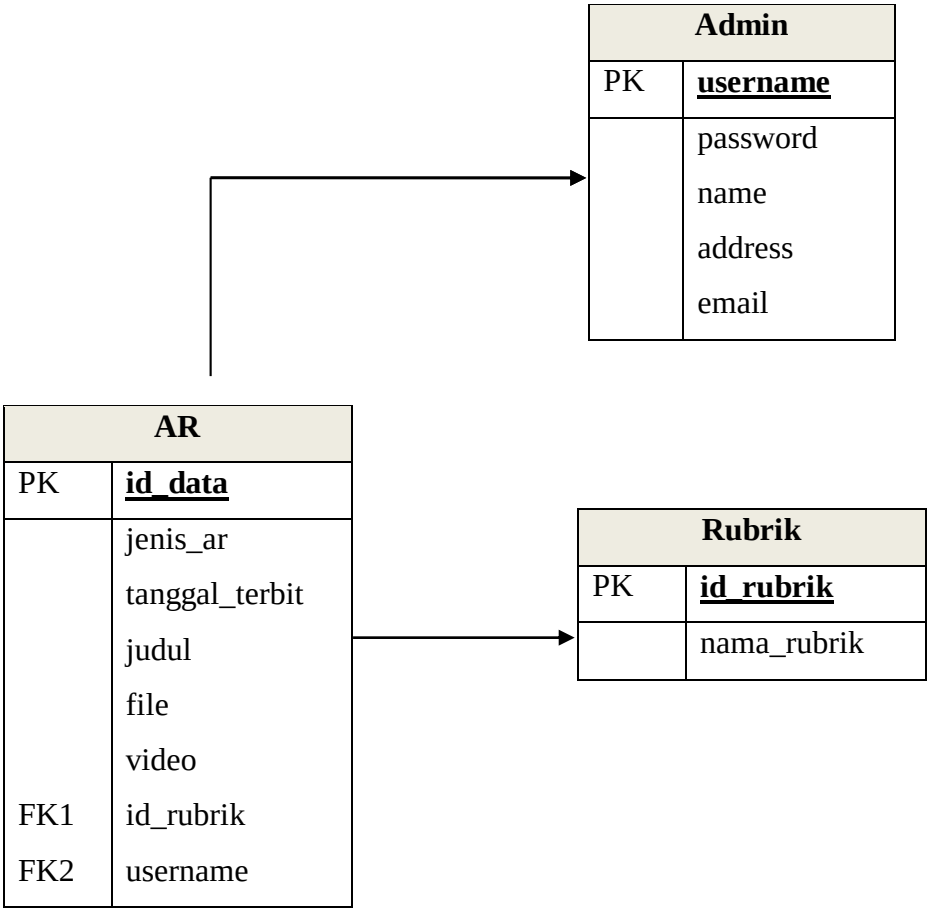
3.2.1 Perancangan Basis Data

Perancangan basis data merupakan tahapan untuk memetakan model konseptual ke model basis data yang akan dipakai. Perancangan basis data terbagi mendjadi dua yaitu skema relasi dan perancangan sktruktur data.

3.2.1.1 Skema Relasi

Skema relasi atau relasi antar table 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.



Gambar 3.40 Skema Relasi

3.2.1.2 Struktur Tabel

1. Tabel Admin

Tabel 3.21 Struktur Tabel Admin

Nama Field	Tipe Data	Panjang	Keterangan
Username	Varchar	20	Primary Key
Password	Varchar	50	
Name	Varchar	50	
Address	Varchar	50	
Email	Varchar	40	

2. Tabel Rubrik

Tabel 3.22 Struktur Tabel Rubrik

Nama Field	Tipe Data	Panjang	Keterangan
id_rubrik	Integer	Auto increment	Primary Key
nama_rubrik	Varchar	50	

3. Tabel AR

Tabel 3.23 Struktur Tabel AR

Nama Field	Tipe Data	Panjang	Keterangan
id_data	Varchar	30	Primary Key
jenis_ar	Varchar	50	
tanggal_terbit	Date		
judul	Varchar	50	
file	Text		
video	Text		
id_rubrik	Integer	Auto increment	Foreign Key, reference (T.rubrik (id_rubrik))
username	Varchar	20	Foreign Key, reference (T.admin(username))

3.2.2 Perancangan Grafis dan Antarmuka

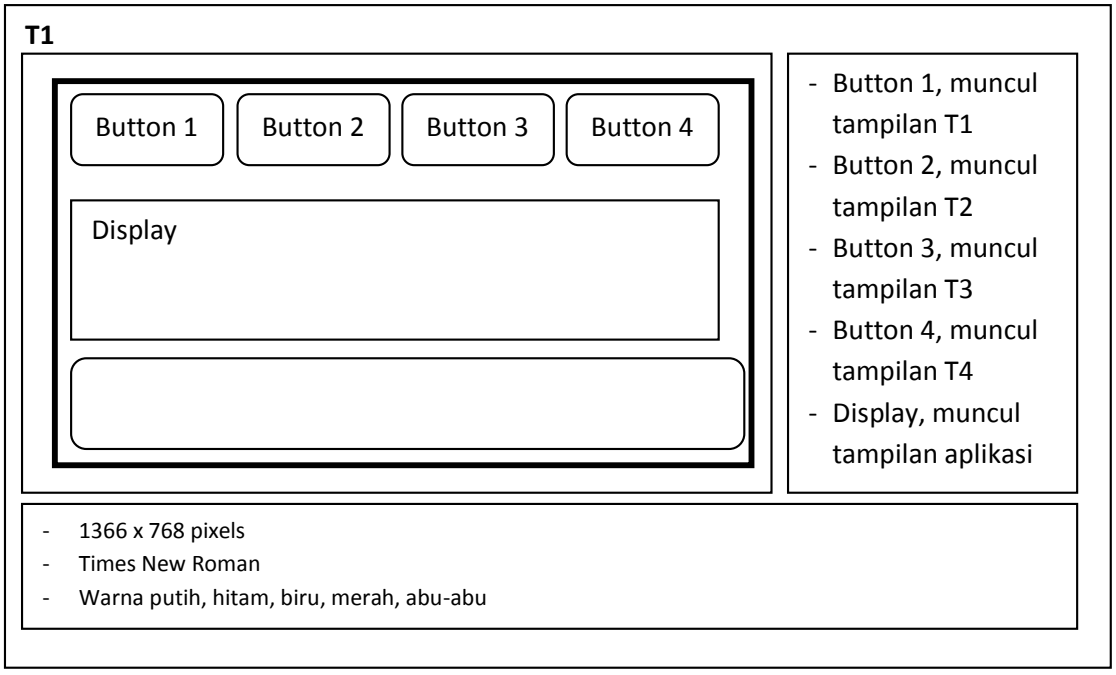
Tahap perancangan desain bertujuan untuk mencari bentuk yang optimal dari aplikasi yang akan dibangun dengan pertimbangan factor-faktor permasalahan dan kebutuhan yang ada pada sistem seperti yang telah ditetapkan pada tahap analisis. Dalam

tahap ini upaya yang dilakukan yaitu dengan cara mengkobinasikan penggunaan teknologi perangkat keras dan perangkat lunak yang tepat sehingga diperoleh desain yang optimal dan mudah diimplementasikan.

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

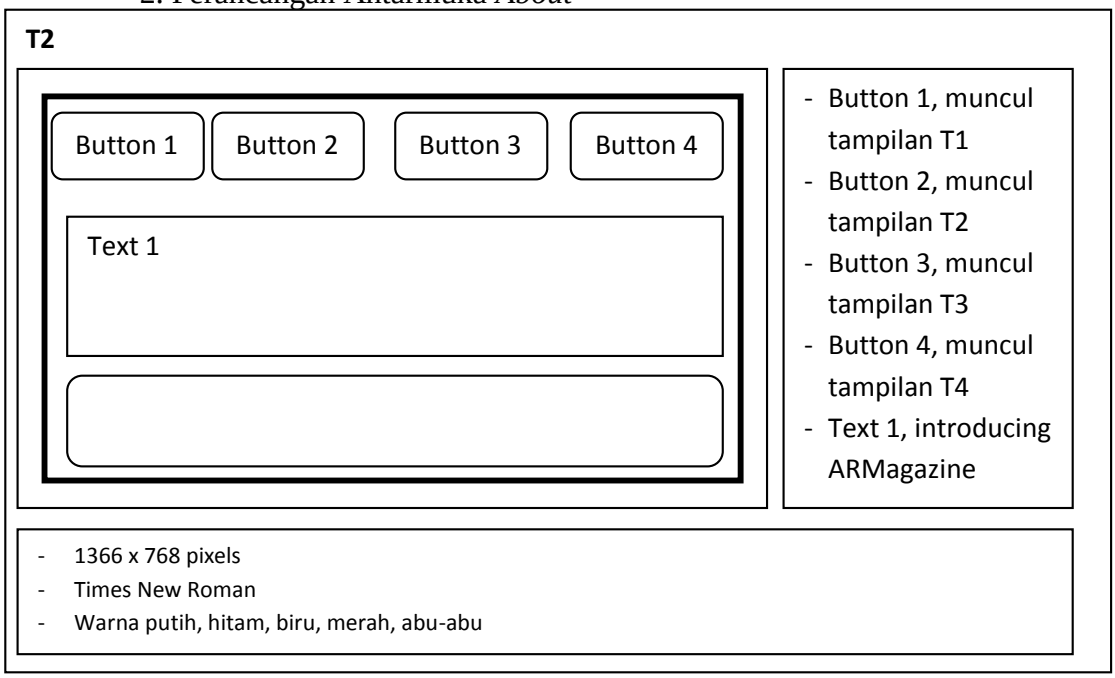
3.2.2.1 Perancangan Antarmuka Aplikasi *Frontend*

1. Perancangan Antarmuka Menu



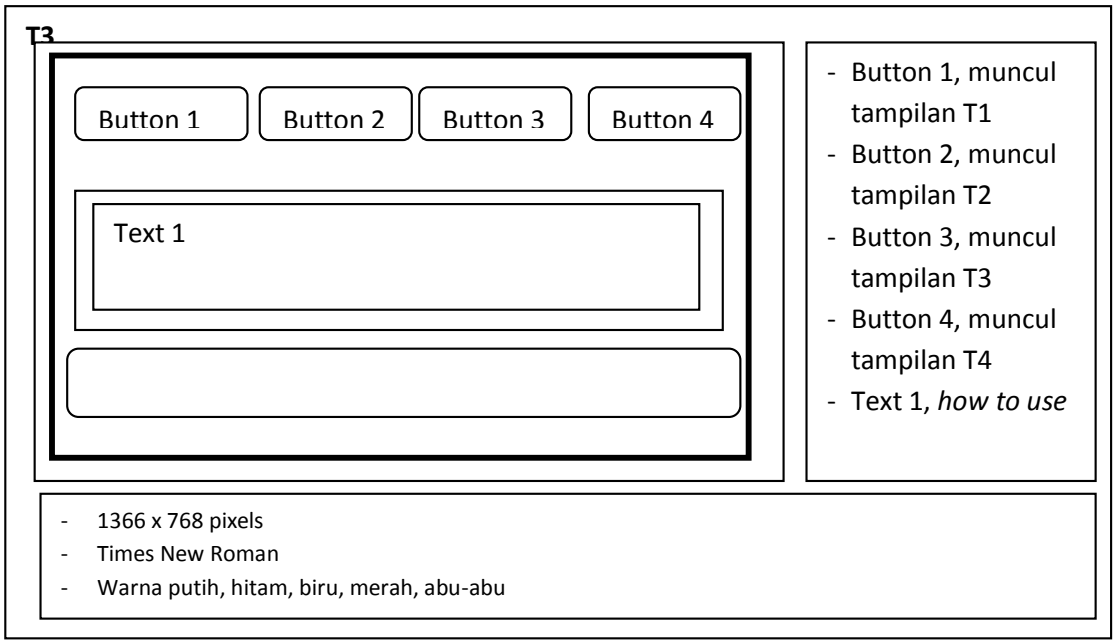
Gambar 3.41 Perancangan Antarmuka Menu

2. Perancangan Antarmuka *About*



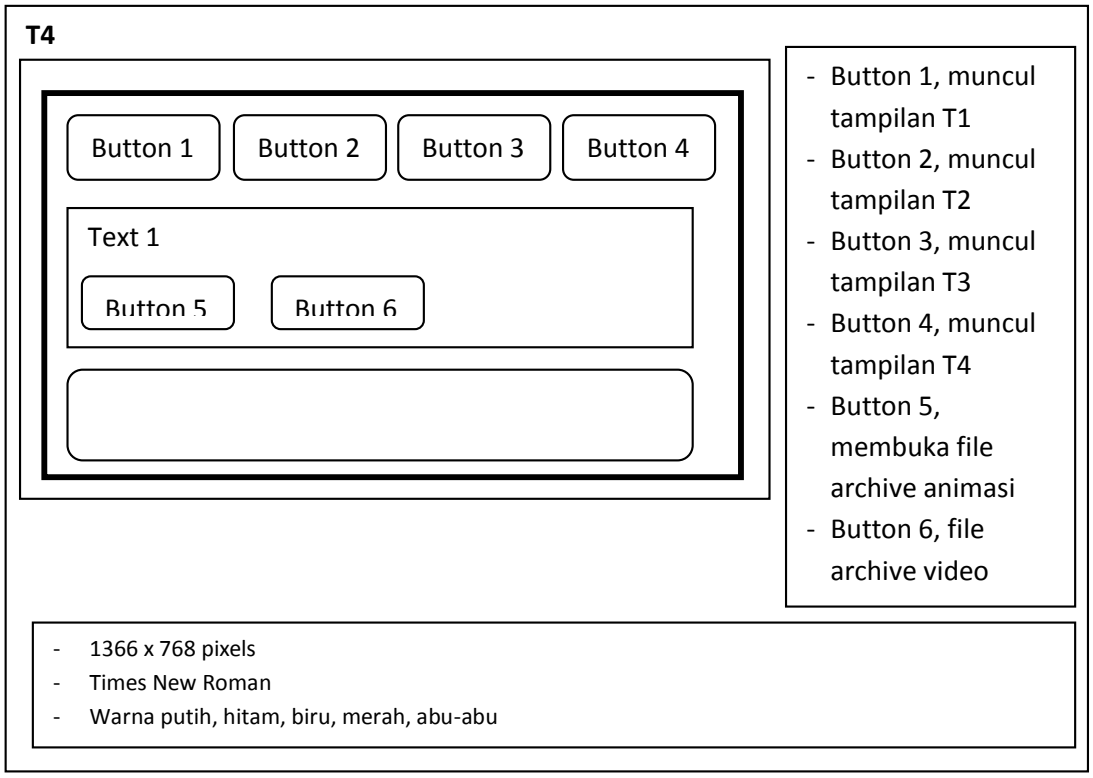
Gambar 3.42 Perancangan Antarmuka *About*

3. Perancangan Antarmuka *How to use*



Gambar 3.43 Perancangan Antarmuka *How to Use*

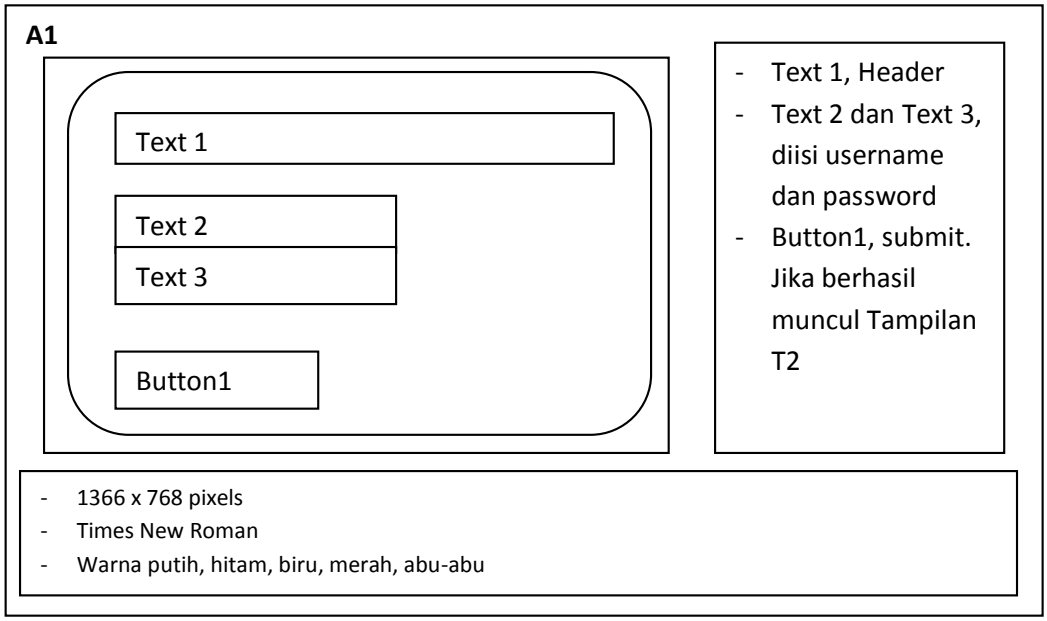
4. Perancangan Antarmuka *Archive*



Gambar 3.44 Perancangan Antarmuka *Archive*

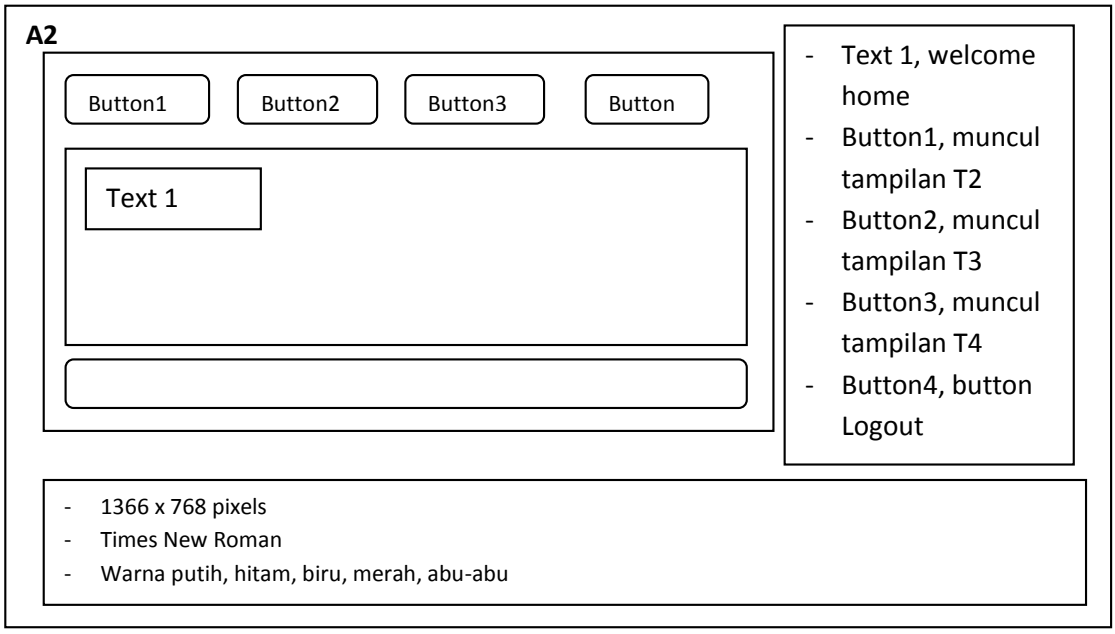
3.2.2.2 Perancangan Antarmuka Aplikasi *Backend*

1. Perancangan Antarmuka *Login CMS ARMagazine*



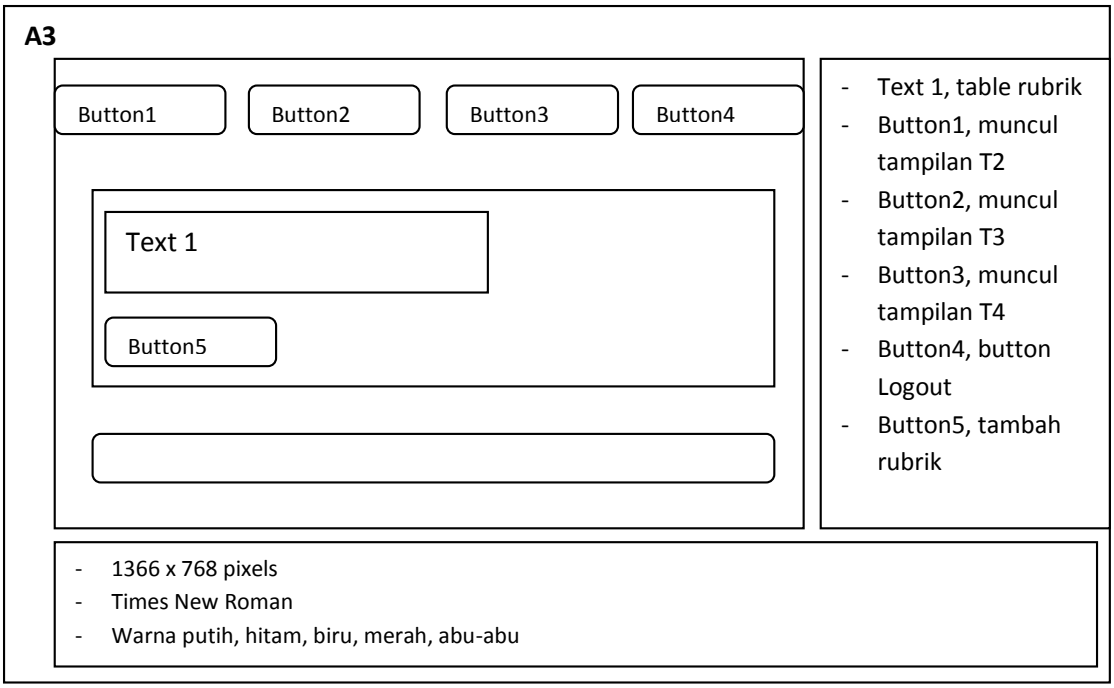
Gambar 3.45 Perancangan Antarmuka *Login CMS*

2. Perancangan Antarmuka Halaman Utama CMS



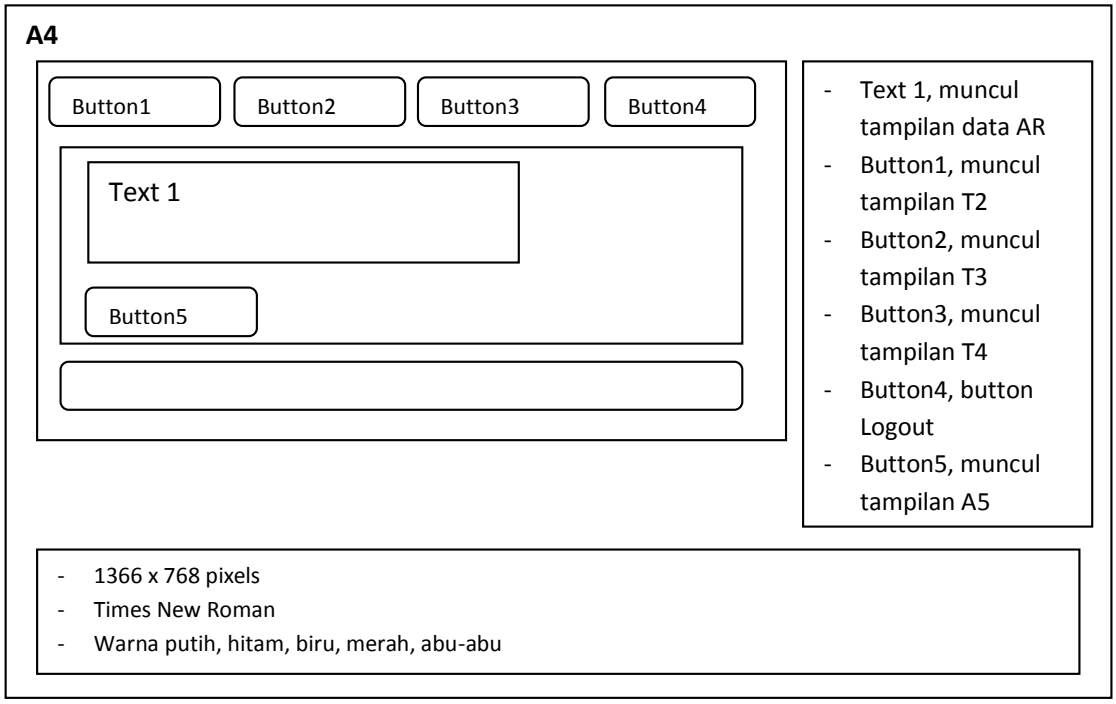
Gambar 3.46 Perancangan Antarmuka Halaman Utama CMS

3. Perancangan Antarmuka Rubrik



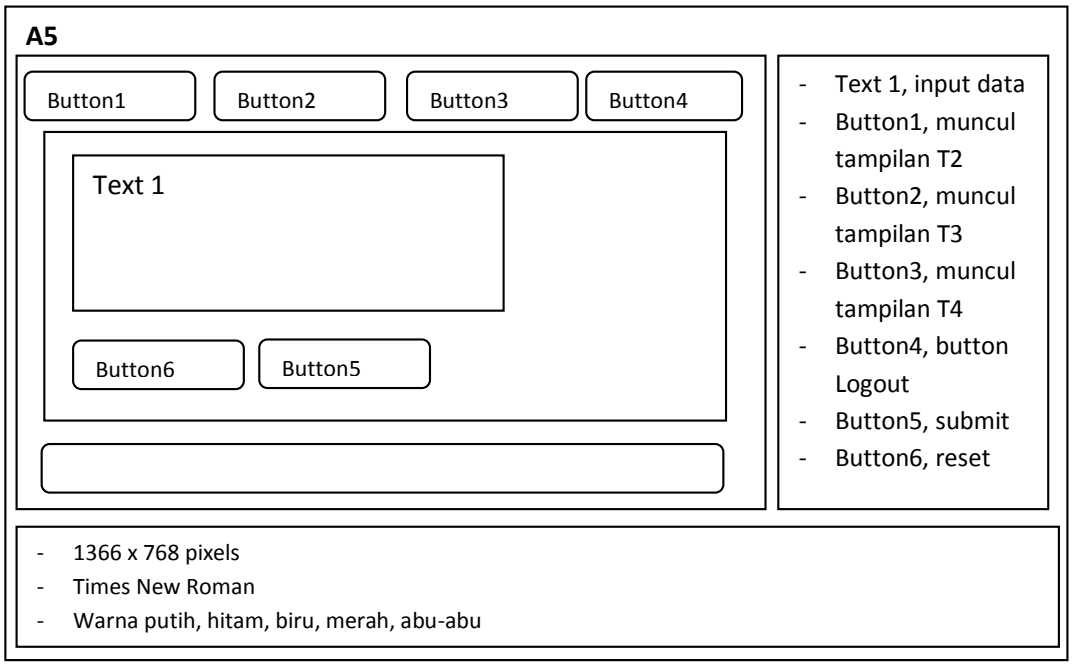
Gambar 3.47 Perancangan Antarmuka Rubrik

4. Perancangan Antarmuka Data Objek AR



Gambar 3.48 Perancangan Antarmuka Data Objek AR

5. Perancangan Antarmuka Manajemen Tambah Data AR



Gambar 3.49 Perancangan Antarmuka Manajemen tambah Data AR Animasi

3.2.2.3 Perancangan Pesan

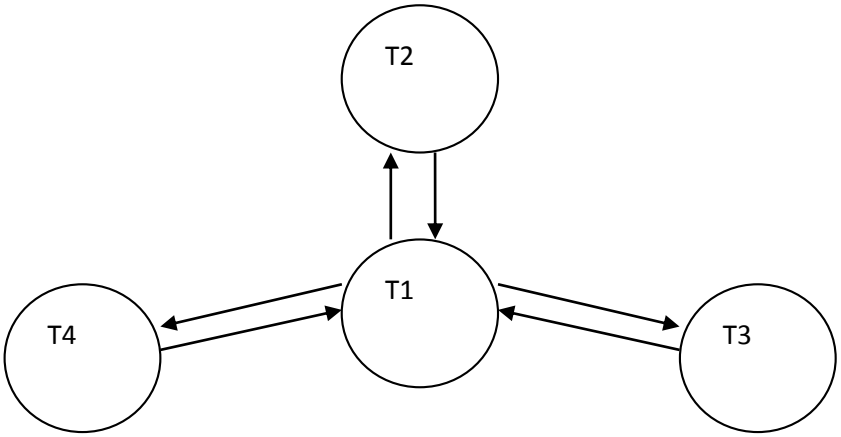
Pada gambar dibawah ini merupakan perancangan pesan yang terdapat pada aplikasi *ARMagazine*, adapun tampilan pesan yang ada adalah sebagai berikut :



3.2.2.4 Jaringan Semantik

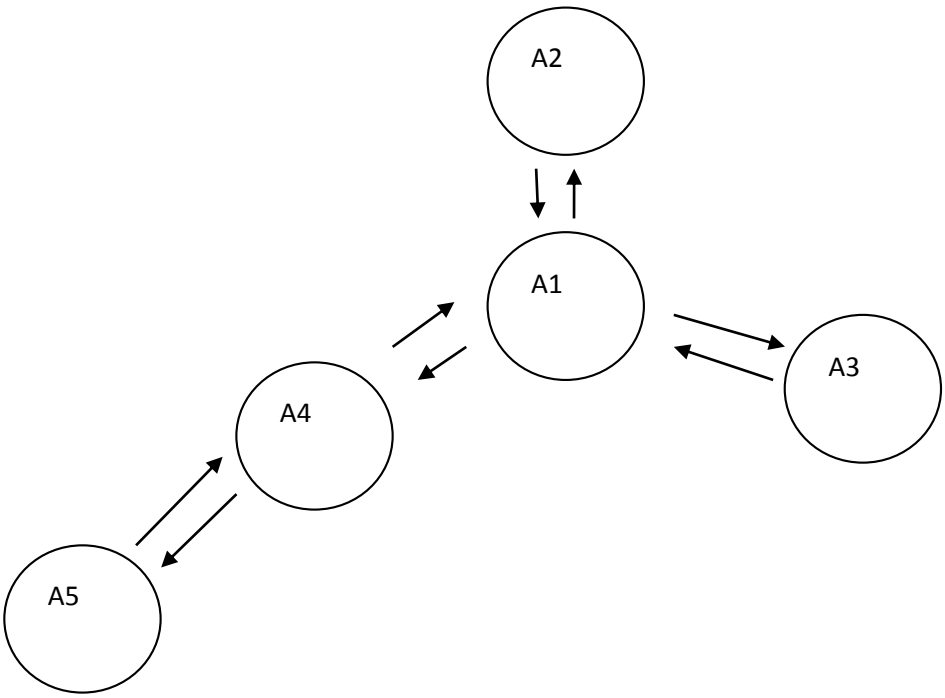
Jaringan semantik sistem yang akan dibangun dibedakan sesuai dengan jenis penggunaannyayaitu *user frontend*, *user backend*. Jaringan semantiknya dapat dilihat pada gambar dibawah ini :

1. Jaringan Semantik *User Frontend*



Gambar 3.50 Jaringan Semantik *User Frontend*

2. Jaringan Semantik *User Backend*



Gambar 3.51 Jaringan Semantik Admin