# 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 untuk mengidentifikasikan masalah masalah dan hambatan-hambatan, sehingga dapat diusulkan kebutuhan-kebutuhan untuk memperbaikinya. Tahap analisis sistem adalah tahapan dasar sebelum perancangan untuk memperbaiki sistem yang lama. Dari hasil analisis tersebut dapat dirancang atau diperbaiki menjadi sebuah sistem yang lebih efektif dan efisien.

### 3.1.1 Analisis Masalah

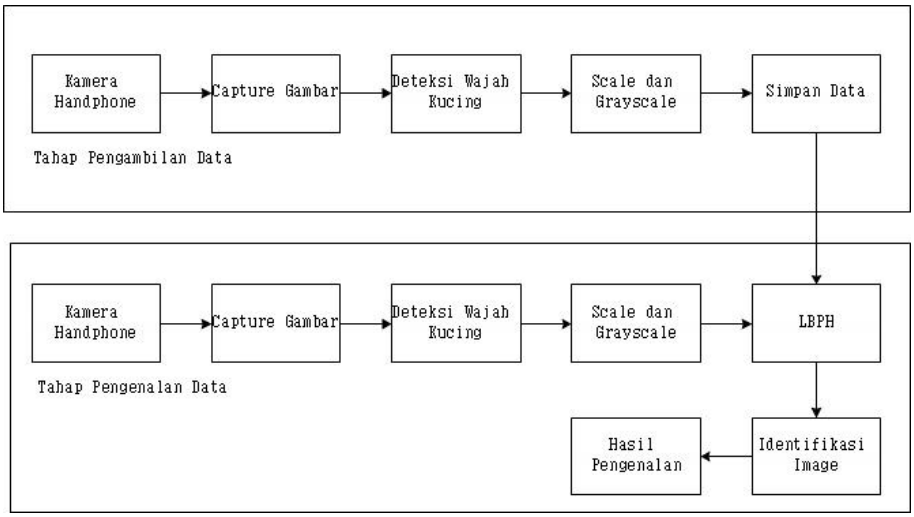
Berdasarkan hasil analisis terhadap sistem yang ada, maka diidentifikasikan masalah-masalah yang ada sebagai berikut :

1. Membantu penguna dalam pengidentifikasian ras kucing.
2. Memberikan informasi standat dari jenis ras tersebut.
3. Mengimplementasikan deteksi dan pengenalan ras kucing menggunakan android dengan metode Viola-Jones dan Local Binari Pattern Histogram (LBPH).
4. Menguji dan menganalisa hasil dari perangkat lunak dalam melakukan sistem deteksi (persentase keberhasilan deteksi) berdasarkan metode Viola-Jones dan Local Binari Pattern Histogram (LBPH).

### 3.1.2 Analisis Algoritma

Analisis algoritma dilakukan untuk dapat mengetahui alur proses dari algoritma yang digunakan dan dapat diterapkan kedalam sistem perangkat lunak. Dalam proses pendeteksian ini menggunakan library dalam OpenCV. Pada pendeteksian wajah (face detetection) kucing pada perangkat lunak ini digunakan

metode yaitu viola-jones. Serta dibantu oleh algoritma Local Binari Pattern Histograms (LBPH) untuk pelabelan gambar (face recognition). Berikut ini adalah tahap-tahap yang dilakukan untuk mendeteksi ras kucing, yaitu:




**Gambar 3.1 Blok Diagram System**

Gambar 3.1 merupakan blok diagram software untuk melakukan pengenalan dari sistem yang digunakan, dimana jalannya sistem dibagi menjadi 2, yaitu blok penyimpanan data dan blok pengenalan wajah. Jalannya sistem blok diagram adalah sebagai berikut :

1. Pengambilan gambar secara real-time menggunakan kamera pada handphone.
2. Proses deteksi kucing menggunakan algoritma Viola-Jones dengan haar cascade.
3. Proses penskalaan (scaling) dan grayscale.
4. Proses penyimpanan data gambar kucing berdasarkan ras.
5. Proses pengenalan dengan Local Binary Pattern Histogram (LBPH).

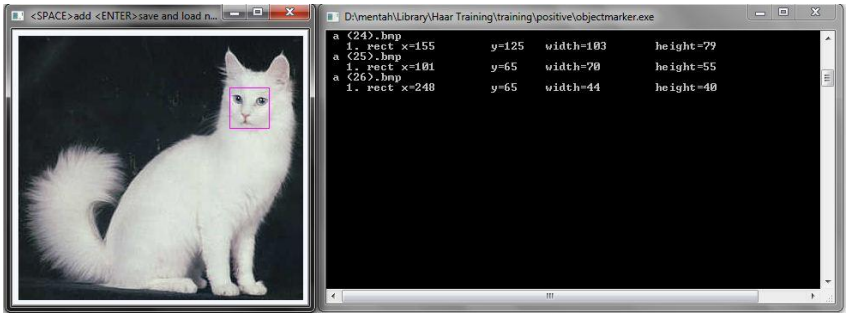
### 3.1.2.1 Proses Deteksi Kucing

Proses deteksi kucing menggunakan algoritma Viola-Jones yaitu menggunakan haar cascade. Untuk mendeteksi diperlukan data training setelah itu pengimplementasian algoritma haar cascade pada program.

### 3.1.2.2 Training Data

Pada proses haar cascade ini membutuhkan data training berupa gambar positif dan gambar negatif. Gambar positif adalah gambar dari objek yang akan dideteksi dalam penelitian ini yaitu gambar wajah kucing sedangkan gambar negatif adalah gambar selain objek yang akan dideteksi seperti background, gunung, lemari, pohon, dan lain sebagainya. Proses training dilakukan menggunakan tool yang berada dalam OpenCV yaitu [21]:

1. Objectmarker.exe, program ini digunakan untuk manandai object secara manual.
2. Createsamples.exe, program ini digunakan untuk menciptakan sample dataset positif.
3. Haartraining.bat, program ini digunakan untuk proses training.
4. Convert.bat, program ini digunakan untuk mengubah cascade classifier ke bentuk file xml, untuk kemudian digunakan pada program utama.




**Gambar 3.2 Proses Menandai Objek saat Training Data**

Setelah didapatkan nilai training yang telah disimpan dalam file xml, selanjutnya data akan dibandingkan dengan gambar yang didapat dari kamera untuk mendeteksi wajah kucing. Kemudian setiap frame mendeteksi apakah objek gambar yang masuk wajah atau bukan.

### 3.1.2.3 Algoritma Haar Cascade

Setelah training data telah dilakukan, selanjurnya memanggil algoritma Haar Cascade dalam program. Pelacakan wajah pada penelitian ini dikhususkan untuk pelacakan wajah dengan posisi lurus ke depan terhadap kamera *(frontal face)*. Dan berikut ini proses-proses yang terjadi dalam metode haar cascade.

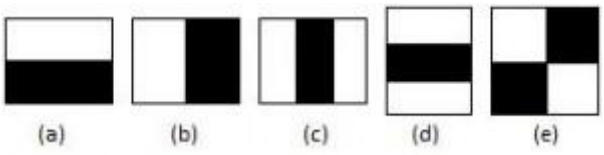
1 Proses Menentukan Haar

Nilai Haar feateure pada openCV yang dikenal dengan Haarcascade. Proses ini dilakukan untuk mendapatkan sampel dari gambar yang didapat dari kamera dan akan dibandingkan dengan nilai yang ada pada OpenCV. Kemudian nilai frame akan digunakan untuk mendeteksi apakah objek gambar yang masuk wajah atau bukan.

Keberadaan ada atau tidaknya fitur wajah ditentukan dengan mengurangi nilai pixel di wilayah gelap dengan nilai pixel di wilayah terang. Jadi Setiap gambar dirubah kedalam warna hitam dan putih. Jika nilai dari hasil perbedaanya di atas dari ambang batas selama masa pembelajaran citra maka fitur tersebut dapat dikatakan ada.

Fitur-fitur ini merupakan gambaran dari wajah yang dikelompokkan berdasarkan sisi yang terang dan sisi yang gelap. Contohnya daerah mata memiliki sisi yang lebih gelap daripada bagian di antara dua mata.

Terdapat 3 jenis fitur berdasarkan jumlah persegi yang terdapat di dalamnya, seperti yang dapat dilihat pada gambar di bawah ini :




**Gambar 3.3 Point Fitur Metode Viola Jones**

Pada gambar di atas dapat dilihat bahwa fitur (a) dan (b) terdiri dari dua persegi panjang, sedangkan fitur (c) dan (d) terdiri dari tiga persegi panjang dan

fitur (e) empat persegi panjang. Cara menghitung nilai dari fitur ini adalah mengurangkan nilai piksel pada area hitam dengan piksel pada area putih. Untuk mempermudah proses penghitungan nilai fitur, algoritma Viola-Jones menggunakan sebuah media berupa citra integral.

2 Proses Menghitung Citra Integral

Untuk menghitung nilai dari setiap fitur Haar maka digunakan citra integral. Setiap fitur mempunyai nilai yang berbeda sehingga dibutuhkan cara untuk mendapatkan fitur yang memiliki nilai yang paling baik

Cara menghitung nilai dari fitur ini adalah mengurangkan nilai piksel pada area hitam dengan piksel pada area putih. Untuk mempermudah proses penghitungan nilai fitur, metode ini menggunakan sebuah media berupa citra integral yaitu adalah sebuah citra yang nilai tiap pikselnya merupakan akumulasi dari nilai piksel atas dan kirinya.

3 Proses Membuat Cascade Classifier

Hasil deteksi dari Haar ini belum terlalu bagus karena hasil deteksinya masih sangat sedikit lebih baik dari asal tebak. Jika ingin mendapatkan hasil yang lebih akurat maka harus dilakukan proses Haar secara *massal* (banyak), semakin banyak proses Haar yang dilakukan maka akan semakin akurat hasil dicapai, namun waktu yang dibutuhkan untuk memproses gambar tersebut akan menjadi lebih lama. Pemrosesan Haar-Like feature yang banyak tersebut diorganisir atau diatur di dalam *classifier cascade*.

Setelah objek terdeteksi dibuatlah kotak pendeteksian menggunakan :

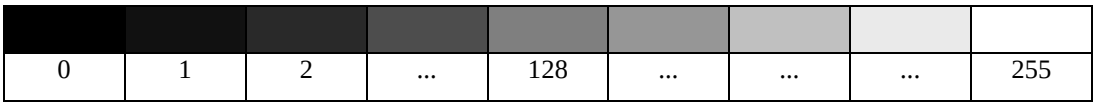
```
Rect[] facesArray = faces.toArray();
for (int i = 0; i < facesArray.length; i++)
   Core.rectangle(mRgba, facesArray[i].tl(),
facesArray[i].br(), FACE_RECT_COLOR, 3);
```

Keterangan

1. Rect adalah template class untuk membuat kotak 2 dimensi.
2. facesArray adalah nama objek dari template class Rect.
3. faces.toArray adalah mengubah setiap frame gambar ke dalam array.
4. Core.rectangle adalah fungsi untuk menggambar kotak.
5. mRgba adalah gambar.
6. facesArray[i].tl() adalah point vertex kotak.
7. facesArray[i].br() adalah point vertex kotak lawan.
8. FACE_RECT_COLOR adalah warna kotak yang telah dideklarasi sebelumnya.
9. 3 adalah ketebalan kotak.

### 3.1.2.4 Tahap Grayscale

Grayscale adalah warna-warna piksel yang berada dalam rentang gradasi warna hitam dan putih. Tahap ini adalah tahap setelah melalui proses scaling kemudian diubah menjadi gambar dua warna dengan proses grayscaling. Proses grayscaling dilakukan oleh sistem perangkat lunak. Pada umumnya warna yang dipakai warna hitam sebagai warna minimal (0) dan warna putih (255) sebagai warna maksimalnya, sehingga warna antaranya adalah abu-abu, seperti ditunjukkan pada Gambar 3.4.




**Gambar 3.4 Pallete skala grayscale**

Derajat keabuan sendiri sebenarnya memiliki beberapa nilai, tidak hanya skala 0 sampai 255. Hal ini tergantung pada nilai kedalaman pixel yang dimiliki oleh citra.

Misalnya dalam gambar kucing yang telah dicapture mempunyai nilai warna Red, Green dan Blue seperti pada Tabel 3.1.

**Tabel 3.1 Nilai Warna Gambar Capture**

| x/y | | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | R | 210 | 215 | 209 | 220 | 220 | 207 | 203 | 192 | 208 | 209 |
| | G | 200 | 190 | 175 | 185 | 191 | 161 | 155 | 129 | 173 | 163 |
| | B | 175 | 136 | 104 | 121 | 135 | 84 | 79 | 52 | 115 | 88 |
| 1 | R | 212 | 221 | 228 | 222 | 216 | 214 | 207 | 209 | 193 | 193 |
| | G | 185 | 192 | 202 | 196 | 189 | 173 | 171 | 173 | 130 | 143 |
| | B | 132 | 136 | 145 | 145 | 142 | 107 | 113 | 123 | 61 | 72 |
| . . . | | | | | | | | | | | |
| 9 | R | 212 | 221 | 228 | 222 | 216 | 214 | 207 | 209 | 193 | 193 |
| | G | 185 | 192 | 202 | 196 | 189 | 173 | 171 | 173 | 130 | 143 |
| | B | 132 | 136 | 145 | 145 | 142 | 107 | 113 | 123 | 61 | 72 |

Menghasilkan nilai keabuan dengan rumus

$$s = \frac{r + g + b}{3}$$

Menjadi

**Tabel 3.2 Nilai Grayscale Hasil Perhitungan**

| x/y | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 195 | 180 | 163 | 175 | 182 | 151 | 146 | 124 | 165 | 153 |
| 1 | 176 | 183 | 192 | 188 | 182 | 165 | 164 | 168 | 128 | 136 |
| .<br>.<br>. | | | | | | | | | | |
| 9 | 176 | 183 | 192 | 188 | 182 | 165 | 164 | 168 | 128 | 136 |

Pada program untuk mengubah kamera menjadi grayscale menggunakan

```
inputFrame.gray();
```

Keterangan

- inputFrame adalah  image dari camera view.
- gray adalah metode untuk mengubah frame pada kamera menjadi grayscale.

### 3.1.2.5 Tahap Scaling

Tahap scaling adalah tahap untuk teknik yang berguna untuk merubah ukuran *gambar* dalam hal ini memperbesar dan memperkecil *gambar*, biasanya ukuran untuk proses pendeksian wajah yang efektif menggunakan pixel 20x20. Misalkan gambar yang telah dicapture mempunyai nilai value :

**Tabel 3.3 Nilai Data Gambar**

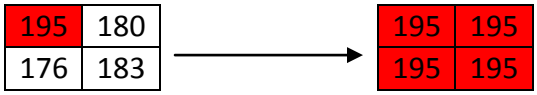
| x/y | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 195 | 180 | 163 | 175 | 182 | 151 | 146 | 124 | 165 | 153 |
| 1 | 176 | 183 | 192 | 188 | 182 | 165 | 164 | 168 | 128 | 136 |
| .<br>.<br>. | | | | | | | | | | |
| 9 | 176 | 183 | 192 | 188 | 182 | 165 | 164 | 168 | 128 | 136 |

Dikarenakan gambar berukuran 10x10 maka gambar harus diskala sehingga menjadi 20x20. Karena rumus untuk penskalaan atau scaling adalah

$$x' = s_x . x$$

$$y' = s_y . y$$

Dimana $\mathbf{s_x}$ dan $\mathbf{s_y}$ adalah faktor skala masing-masing dalam arah **x** dan arah **y.** Maka Sx =2 dan Sy =2.



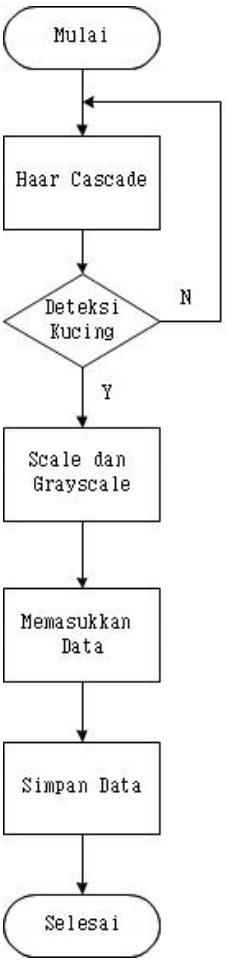

Karena ukuran gambar diperbesar maka nilai dua petak tetanggannya mempunyai nilai yang sama sehingga menghasilkan :

**Tabel 3.4 Nilai Scale Hasil Perhitungan**

| x/y | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 195 | 195 | 180 | 180 | 163 | 163 | 175 | 175 | 182 | 182 | 151 | 151 | 146 | 146 | 124 | 124 | 165 | 165 | 153 | 153 |
| 1 | 195 | 195 | 180 | 180 | 163 | 163 | 175 | 175 | 182 | 182 | 151 | 151 | 146 | 146 | 124 | 124 | 165 | 165 | 153 | 153 |
| 2 | 176 | 176 | 183 | 183 | 192 | 192 | 188 | 188 | 182 | 182 | 165 | 165 | 164 | 164 | 168 | 168 | 128 | 128 | 136 | 136 |
| 3 | 176 | 176 | 183 | 183 | 192 | 192 | 188 | 188 | 182 | 182 | 165 | 165 | 164 | 164 | 168 | 168 | 128 | 128 | 136 | 136 |

### 3.1.2.6 Proses Pengenalan Kucing

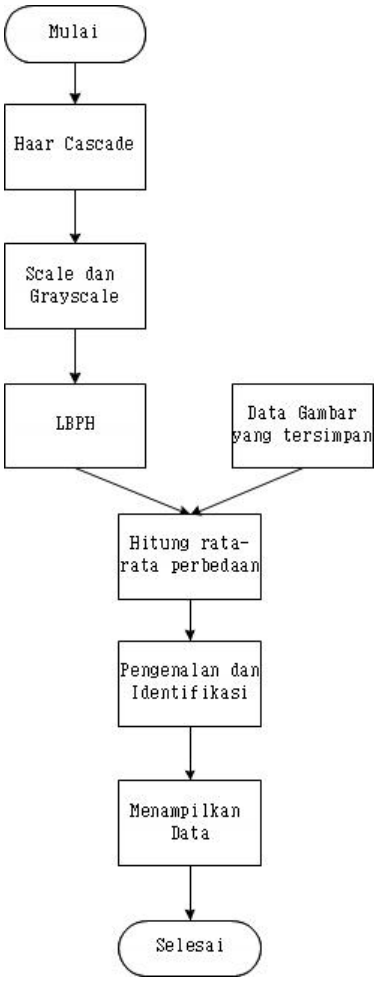
Sebelum gambar dapar diberi label, terlebih dahulu gambar dan data kucing harus disimpan dalam database. Pada proses pengambilan data dan gambar terdapat tahap-tahap seperti pada Gambar 3.5.




**Gambar 3.5 Flowchart Penyimpanan Data Gambar**

### 3.1.2.7 Proses Pengidentifikasian Ras Kucing

Dalam proses pengidentifikasian, kamera diarahkan pada kucing yang akan dideteksi, lalu jika dikenali maka akan ditampilkan data-data kucing. Pada proses pendeteksian memiliki beberapa tahapan seperti pada Gambar 3.6.



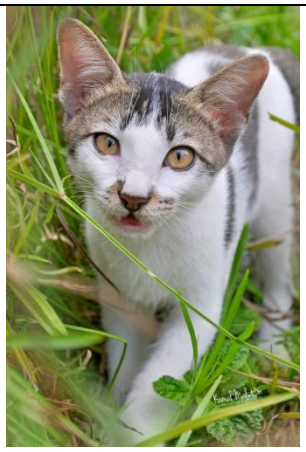

**Gambar 3.6 Flowchart Pengenalan Ras Kucing**

Untuk pendeteksian ras kucing dalam penelitian ini menggukan Local Binary Pattern Histogram (LBPH). Sedangkan untuk gambar data ras kucing memiliki ciri-ciri sebagai berikut:

**Tabel 3.5 Tabel Ciri-Ciri dari Gambar Kucing**

| No | Gambar | Ciri-ciri |
|---|---|---|
| 1 |  | Persia<br>1. Bentuk kepala : bulat dan lebar.<br>2. Telinga : kecil dan bulat<br>3. Mata : besar, bulat<br>4. Dagu : pas ke wajah<br>5. Hidung : pesek<br>6. Leher : pendek, tebal, berotot<br>7. Badan : pedek dan bulat<br>8. Kaki : besar dan bulat<br>9. Ekor : pendek dan lurus<br>10. Bulu : panjang<br>11. Warna : beragam |
| 2 |  | Exotic<br>1. Bentuk kepala : bulat dan lebar.<br>2. Telinga : kecil dan bulat<br>3. Mata : besar, bulat<br>4. Dagu : pas ke wajah<br>5. Hidung : pesek<br>6. Leher : pendek, tebal, berotot<br>7. Badan : pedek dan bulat<br>8. Kaki : besar dan bulat<br>9. Ekor : pendek dan lurus<br>10. Bulu : pendek<br>11. Warna : beragam |

| | | |
|---|---|---|
| 3 |  | Anggora<br>1. Bentuk kepala : kecil, lembut, dan datar<br>2. Telinga : besar, lebar, runcing<br>3. Mata : besar, seperti walnut dengan warna mata biru atau kuning<br>4. Dagu : tegak lurus dengan hidung<br>5. Hidung : mancung<br>6. Leher : ramping dan anggun<br>7. Badan : panjang dan ramping<br>8. Kaki : panjang dan bulat<br>9. Ekor : panjang dengan ujung menyempit<br>10. Bulu : panjang<br>11. Warna : Putih bersih |
| 4 |  | Himalayan<br>1. Bentuk kepala : bulat dan lebar.<br>2. Telinga : kecil dan bulat<br>3. Mata : besar, bulat berwarna biru<br>4. Dagu : pas ke wajah<br>5. Hidung : pesek<br>6. Leher : pendek, tebal, berotot<br>7. Badan : pedek dan bulat<br>8. Kaki : besar dan bulat<br>9. Ekor : pendek dan lurus<br>10. Bulu : panjang<br>11. Warna : bersih dan harus ada kontras pada telinga, kaki, kaki, ekor |

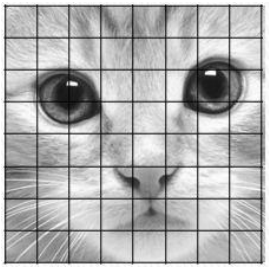
| | | |
|---|---|---|
| 5 |  | Domestic<br>1. Bentuk kepala : kecil dan lancip.<br>2. Telinga : besar<br>3. Mata : besar, berwarna hijau atau kuning<br>4. Dagu : lancip<br>5. Hidung : mancung<br>6. Leher : pendek, tebal, berotot<br>7. Badan : pedek dan bulat<br>8. Kaki : kecil dan panjang<br>9. Ekor : pendek, panjang<br>10. Bulu : pendek<br>11. Warna : beragam |

### 3.1.2.8 Local Binari Pattern Histograms (LBPH)

Setelah melakukan pendeteksian wajah dengan menggunakan haar feature, selanjutnya adalah melakukan pengenalan wajah atau recognition. Pada face recognition setelah melakukan proses pendeteksian kemudian dilakukan poses pengenalan wajah. Pada pengenalan wajah ini digunakan algoritma Local Binari Pattern Histograms (LBPH).

Local Binary Pattern Histogram (LBPH) adalah teknik baru dari metode LBP untuk mengubah performa hasil pengenalan wajah. LBPH adalah metode yang paling cocok untuk dilakukan pengenalan citra wajah untuk diimplementasikan pada perangkat bergerak android karena menggunakan penghitungan yang sederhana.

Mekanismenya adalah pertama-tama membagi daerah gambar menjadi 8x8.



Setiap blok diubah menjadi thresholding dengan ukuran 3x3 untuk mendapatkan nilai histogram. Lalu mengambil satu pixel tengah untuk menjadi pembanding.

- Jika nilai tetangga >= nilai pembanding =1
- Jika nilai tetangga < nilai pembanding = 0

Misalkan gambar yang telah dicapture mempunyai nilai :

| | | |
|---|---|---|
| 195 | 195 | 180 |
| 195 | 195 | 180 |
| 176 | 176 | 183 |

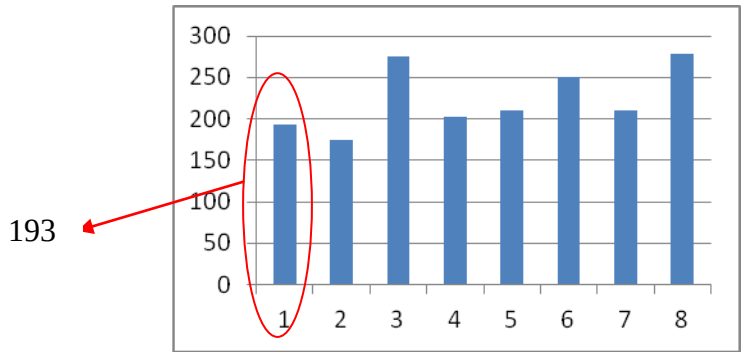
Treshold →

| | | |
|---|---|---|
| 1 | 1 | 0 |
| 1 | | 0 |
| 0 | 0 | 0 |

Binary : 11000001

Desimal : 193

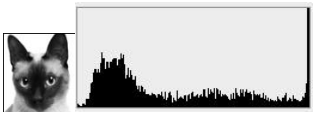
Dibentuk histogram dengan menambahkan nilai setiap blok sesuai dengan pola biner. Misalkan blok yang tadi menghasilkan nilai 193 maka di buat grafik batang seperti pada Gambar 3.7.




**Gambar 3.7 Histogram Hasil**

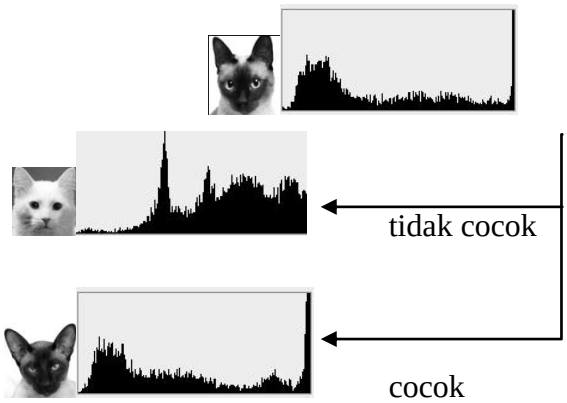
Setelah nilai histogram didapatkan, lalu hasil histogtam akan dibandingkan dengan gambar yang telah di simpan sebelumnya. Sehigga dapat diidentifikasikan apakah gambar tersebut memiliki ciri yang sama dengan hasil deteksi atau tidak.

Misalkan suatu gambar yang telah dicapture memiliki histogram



**Gambar 3.8 Pencapturan dan Histogram Gambar**

Lalu sistem akan mendeteksi setiap gambar, apakah memiliki kecocokan dengan gambar yang di capture. Jika memiliki kecocokan sistem akan menampilkan data-data dari gambar tersebut.




**Gambar 3.9 Pencocokan Gambar**

Untuk menmbandingkan dua histogram yang berbeda dilakukan menggunakan rumus Chi Square yaitu

$$X^2(I,M) = \sum_{i=1}^{n} \frac{(I_i - M_i)^2}{(I_i + M_i)} \quad \text{.....(1)}$$

Dengan I adalah histogram pertama dam M adalah histogram kedua. Dan n adalah jumlah elemen dalam histogram. Pengukuran Chi Squate efektif jika terdapat kesamaan antara sepasang histogram.

Dan Jika gambar yang di deteksi cocok akan menghasilkan data-data tentang kucing tersebut. Sebagai informasi kepada user tentang ras kucing tersebut.

### 3.1.3 Analisis Kebutuhan Non Fungsional

Analisis kebutuhan non fungsional adalah sebuah langkah untuk menganalisis sumber daya manusia yang akan menggunakan perangkat lunak yang dibangun, perangkat keras dan perangkat lunak yang dimiliki sesuai dengan kebutuhan atau permintaan, sehingga dapat ditentukan kompabilitas perangkat lunak yang dibangun terhadap sumber daya yang ada. Analisis kebutuhan non fungsional harus bisa mencakup kebutuhan dan fakta yang ada.

#### 3.1.3.1 Analisis Kebutuhan Perangkat Keras

Adapun kebutuhan perangkat keras yang diperlukan untuk membangun sistem ini adalah sebagai berikut :

1 Processor Intel(R) Atom(TM)  1,7 GHz.
2 RAM 2 GB.
3 Harddisk 250 MB.
4 VGA ATI Radeon HD 4600 Series.

Rekomendasi spesifikasi minimum handphone dalam mengakses aplikasi ini adalah :

1 Sistem Operasi Android versi 4.2
2 Kamera 1,3 Megapixel

#### 3.1.3.2 Analisis Kebutuhan Perangkat Lunak

Perangkat lunak yang dibutuhkan untuk membangun aplikasi ini adalah sebagai berikut :

1 Sistem Operasi Windows 7
2 Eclips Helios
3 OpenCV library
4 Adobe Photoshop CS3

### 3.1.3.3 Analisis Pengguna

Karakteristik pengguna perangkat lunak ini adalah sebagai berikut :

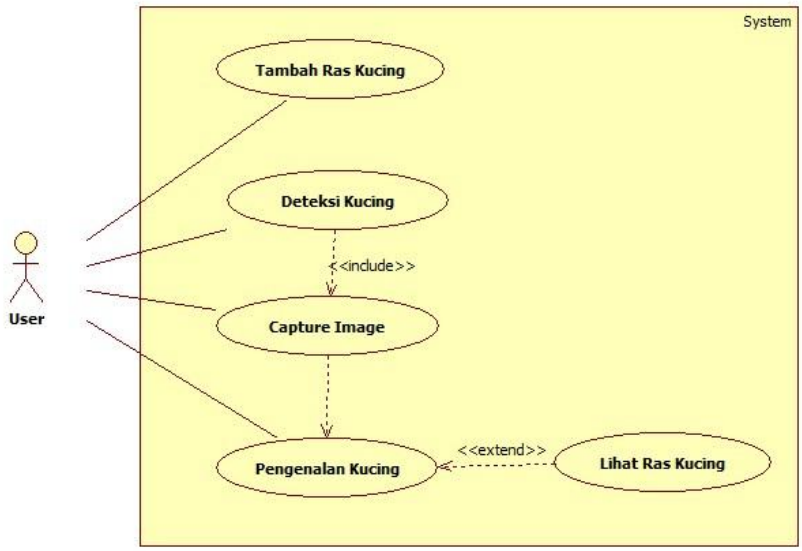
1 *User* harus memiliki kemampuan menggunakan handphone bersistem operasi Android.

### 3.1.4 Analisis Kebutuhan Fungsional

Analisis kebutuhan fungsional merupakan alur dan pekerjaan dari perangkat lunak yang akan dibangun. Dalam perangkat lunak pengenalan wajah ini digunakan metode pendekatan berorientasi objek.

### 3.1.4.1 Use Case Diagram

Use Case Diagram digunakan untuk menggambarkan konteks sistem yang akan dibangun dan fungsionalitas yang disediakan oleh sistem itu. Use Case Diagram juga menggambarkan siapa atau apa berinteraksi dengan sistem. Di dalam sistem hanya terdapat satu user. Tugas user yaitu sebagai aktor yang dapat terlihat pada Gambar 3.10 berikut ini:




**Gambar 3.10 Use Case Diagram**

### 3.1.4.2 Definisi Aktor

Aktor dalam system ini adalah *user*. *User* merupakan orang dari berbagai kalangan dan umur yang mampu mengoperasikan handphone bersistem operasi android.

### 3.1.4.3 Definisi Use case

Definisi dari diagram *use case* pada Gambar 3.10, dideskripsikan secara singkat dalam Tabel 3.6.

**Tabel 3.6 Definisi Use Case**

| No | Use ase | Deskripsi |
|---|---|---|
| 1 | Tambah Ras Kucing | Menambah data ras kucing untuk disimpan dalam database |
| 2 | Browse Camera | Memasuki kamera view |
| 3 | Capture Image | Mengambil sample citra dari kamera |
| 4 | Deteksi Kucing | Mengolah citra wajah hasil capture dari kamera untuk dilakukan deteksi citra wajah kucing |
| 5 | Pengenalan Kucing | Mengolah citra wajah hasil capture dari kamera untuk dilakukan pengenalan dengan citra wajah yang sudah tersimpan di dalam database. |
| 6 | Lihat Ras Kucing | Menampilkan data ras kucing setelah gambar kucing dikenali |

### 3.1.4.4 Use Case Skenario

Skenario proses-proses yang terjadi pada aplikasi, dipaparkan dengan tujuan untuk lebih memudahkan dalam pembangunan aplikasi, agar lebih konsisten.

1 Use Case skenario Tambah Data Ras Kucing

Use case scenario tambah data ras kucing menjelaskan interaksi antara aktor user dengan use case tambah data ras kucing. Adapun use case scenario tambah data ras kucing dapat dilihat pada Tabel 3.7.

**Tabel 3.7 Use Case Skenario Tambah Data Ras Kucing**

| | | |
|---|---|---|
| **Nama Usecase** | Menambah Data RasKucing | |
| **Tujuan** | Menambah data ras kucing | |
| **Prasyarat** | Telah masuk ke dalam aplikasi | |
| **Kondisi Sukses** | Data kucing tersimpan | |
| **Kondisi Gagal** | Data kucing tidak tersimpan | |
| **Aktor** | User | |
| **Alur Utama** | **Tahap** | **Aksi** |
| | 1 | User menekan tombol tambah ras kucing |
| | 2 | User mengisi data-data kucing standar kucing ras berupa nama ras, bentuk kepala, telinga, mata, dagu, hidung, leher, badan, kaki, ekor, bulu, warna, sifat, harga |
| | 3 | User menekan tombol Save |
| | 4 | System menyimpan data dalam database |
| **Alur Alternatif** | **Tahap** | **Aksi** |
| | 2.1 | User salah memasukan data Kucing |
| | 4.1 | System gagal menyimpan data kucing |

2 Use Case skenario Capture Image

Use case scenario capture image menjelaskan interaksi antara aktor user dengan use case capture image. Adapun use case scenario capture image dapat dilihat pada Tabel 3.7.

**Tabel 3.8 Use Case Skenario Capture Image**

| | | |
|---|---|---|
| **Nama Usecase** | Capture Image | |
| **Tujuan** | Mengambil sample citra dari kamera | |
| **Prasyarat** | Berada pada kamera view | |
| **Kondisi Sukses** | Citra hasil capture ditampilkan | |
| **Kondisi Gagal** | Citra hasil capture tidak dapat ditampilkan | |
| **Aktor** | User | |
| **Alur Utama** | **Tahap** | **Aksi** |
| | 1 | User mencapture citra wajah |
| | 2 | System mendeteksi wajah kucing |
| | 3 | System mencapture dan menyimpan hasil capture |
| **Alur Alternatif** | **Tahap** | **Aksi** |
| | 2.1 | System tidak dapat menampilkan hasil capture |

3 Use Case Skenario Deteksi Kucing

Use case scenario deteksi kucing menjelaskan interaksi antara aktor user dengan use case deteksi kucing. Adapun use case scenario deteksi kucing dapat dilihat pada Tabel 3.9.

**Tabel 3.9 Use Case Skenario Deteksi Kucing**

| | | |
|---|---|---|
| **Nama Usecase** | Deteksi Kucing | |
| **Tujuan** | Mendeteksi kucing | |
| **Prasyarat** | Berada pada kamera view | |
| **Kondisi Sukses** | Kucing terdeteksi | |
| **Kondisi Gagal** | Kucing tidak terdeteksi | |
| **Aktor** | User | |
| **Alur Utama** | **Tahap** | **Aksi** |
| | 1 | User menekan tombol train |

| | | |
|---|---|---|
| | 2 | User mengarahkan kamera handphone pada kucing |
| | 3 | System mendeteksi kucing |
| | 4 | Sistem menampilkan kotak deteksi kucing |
| **Alur Alternatif** | **Tahap** | **Aksi** |
| | 3.1 | System tidak mengenali kucing |

4 Use Case Skenario Pengenalan Kucing

Use case scenario pengenalan kucing menjelaskan interaksi antara aktor user dengan use case pengenalan kucing. Adapun use case scenario pengenalan kucing dapat dilihat pada Tabel 3.10.

**Tabel 3.10 Use Case Skenario Pengenalan Kucing**

| **Nama Usecase** | Pengenalan Kucing | |
|---|---|---|
| **Tujuan** | Mengenali kucing yang terdeteksi | |
| **Prasyarat** | Berada pada kamera view | |
| **Kondisi Sukses** | Ras kucing yang dikenali ditampilkan | |
| **Kondisi Gagal** | Ras kucing tidak ditampilkan | |
| **Aktor** | User | |
| **Alur Utama** | **Tahap** | **Aksi** |
| | 1 | User menekan tombol cari dan mengarahkan kamera pada kucing |
| | 2 | System mendeteksi kucing |
| | 3 | System mencocokan kucing dengan database |
| | 4 | System menemukan data kucing |
| | 5 | System menampilkan data kucing berupa jenis ras kucing |
| **Alur Alternatif** | **Tahap** | **Aksi** |
| | 4.1 | System tidak menemukan data kucing |

| | | |
|---|---|---|
| | 5.1 | System tidak menampilkan data kucing |

5 U

se Case skenario Lihat Data Ras Kucing

Use case scenario lihat data ras kucing menjelaskan interaksi antara aktor user dengan use lihat data ras kucing. Adapun use case scenario lihat data ras kucing dapat dilihat pada Tabel 3.7.

**Tabel 3.11 Use Case Skenario Lihat Data Ras Kucing**

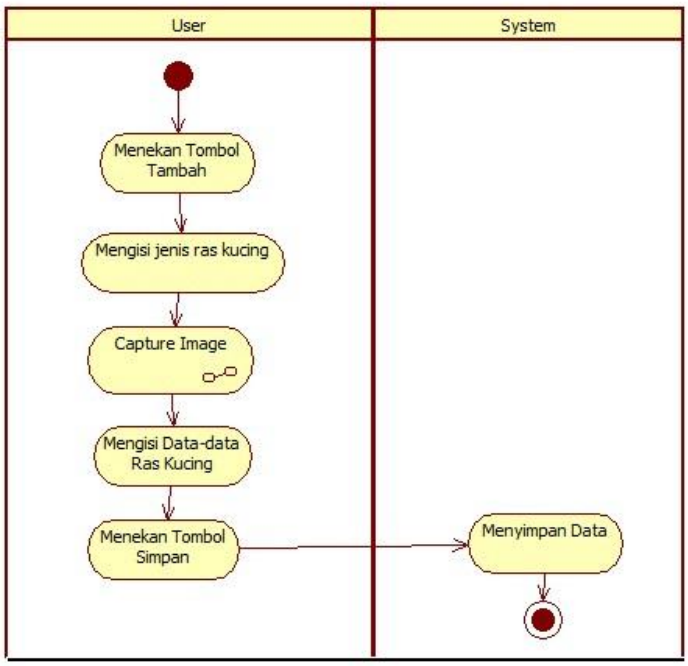
| | | |
|---|---|---|
| **Nama Usecase** | Lihat Data Ras Kucing | |
| **Tujuan** | Melihat data ras kucing | |
| **Prasyarat** | Telah melakukan deteksi | |
| **Kondisi Sukses** | Data kucing ditampilkan | |
| **Kondisi Gagal** | Data kucing tidak dapat ditampilkan | |
| **Aktor** | User | |
| **Alur Utama** | **Tahap** | **Aksi** |
| | 1 | System telah mengenali ras kucing |
| | 2 | User menekan tombol lihat data |
| | 3 | System menampilkan data-data kucing berupa nama ras, bentuk kepala, telinga, mata, dagu, hidung, leher, badan, kaki, ekor, bulu, warna, sifat, harga |
| **Alur Alternatif** | **Tahap** | **Aksi** |
| | 3.1 | System tidak dapat menampilkan data-data kucing |

### 3.1.4.5 Activity Diagram

Activity diagram merupakan cara menggambarkan proses bisnis dan urutan aktivitas dalam sebuah proses yang dipakai pada business modeling untuk memperlihatkan urutan aktifitas proses bisnis. Activity diagram dibuat berdasarkan sebuah atau beberapa use case pada use case diagram .

1 Activity Diagram Tambah Data Ras Kucing

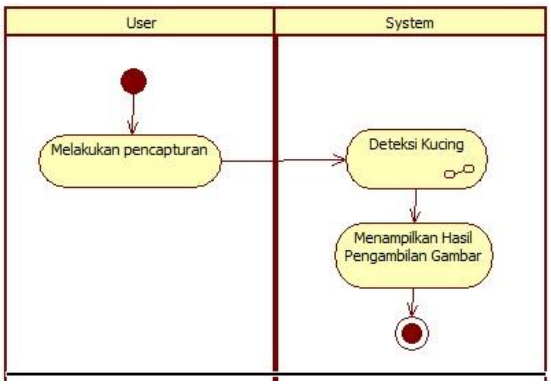
Activity diagram tambah data ras kucing menjelaskan aliran kerja aktor user pada saat tambah data ras kucing. Activity diagram tambah data ras kucing dapat dilihat pada Gambar 3.11.




**Gambar 3.11 Activity Diagram Tambah Data Ras Kucing**

2 Activity Diagram Capture Image

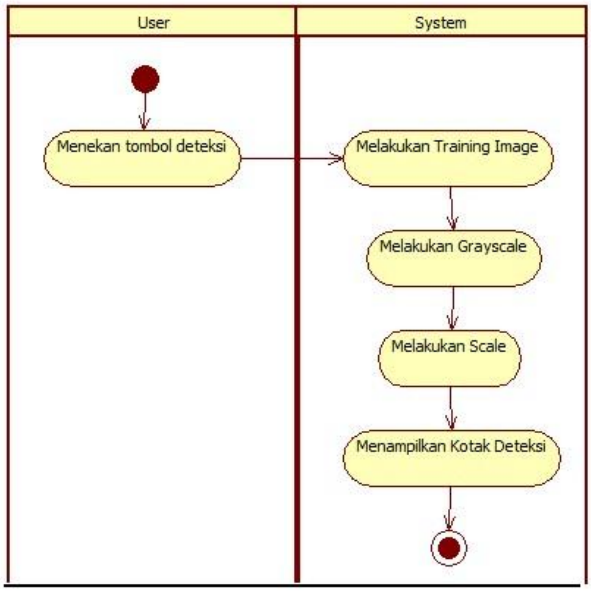
Activity diagram capture image menjelaskan aliran kerja aktor user pada saat capture image. Activity diagram capture image dapat dilihat pada Gambar 3.12.




**Gambar 3.12 Activity Diagram Lihat Data Ras Kucing**

3 Activity Diagram Deteksi Kucing

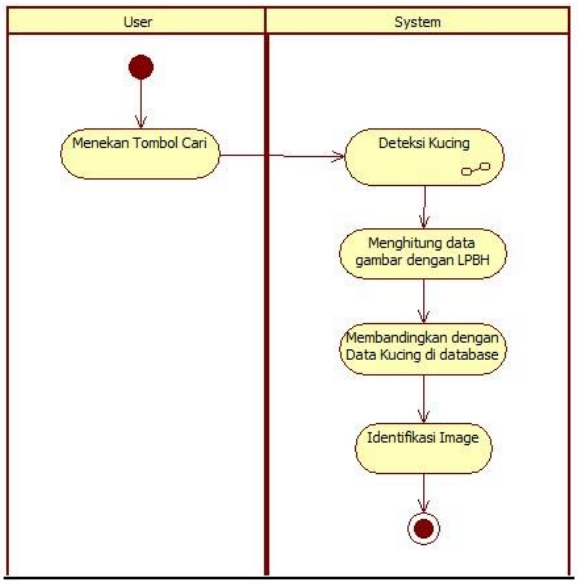
Activity diagram deteksi kucing menjelaskan aliran kerja aktor user pada saat deteksi kucing. Activity diagram deteksi kucing dapat dilihat pada Gambar 3.13




**Gambar 3.13 Activity Diagram Mendeteksi Kucing**

4 Activity Diagram Pengenalan Kucing

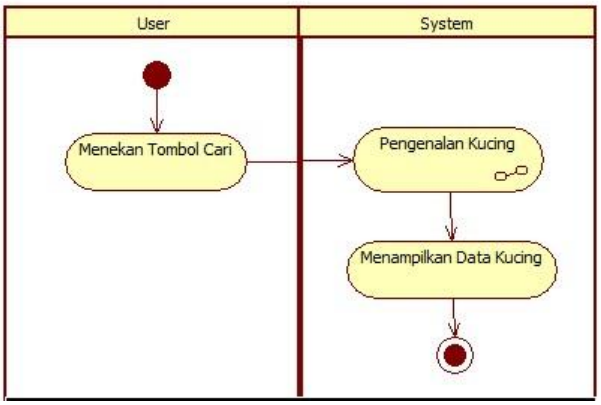
Activity diagram deteksi kucing menjelaskan aliran kerja aktor user pada saat deteksi kucing. Activity diagram deteksi kucing dapat dilihat pada Gambar 3.14.




**Gambar 3.14 Activity Diagram Pengenalan Kucing**

5 Activity Diagram Lihat Data Ras Kucing

Activity diagram lihat data ras kucing menjelaskan aliran kerja aktor user pada saat tambah lihat data ras kucing. Activity diagram lihat data ras kucing dapat dilihat pada Gambar 3.15.



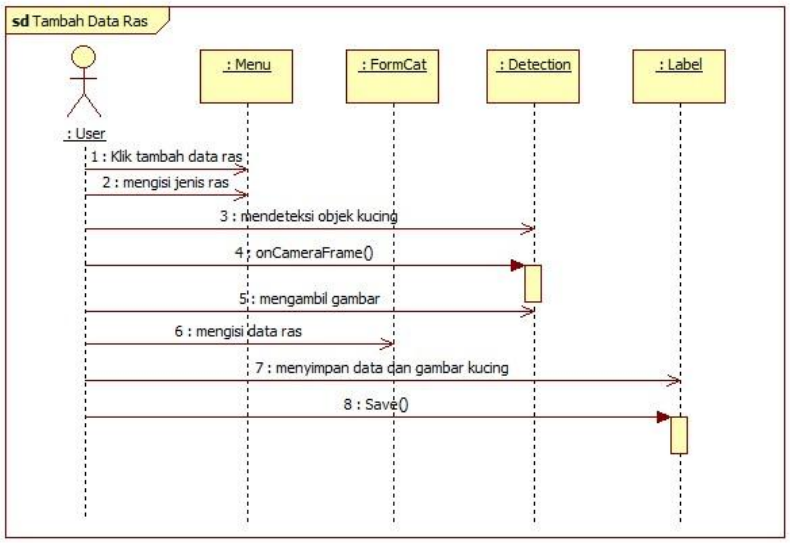

**Gambar 3.15 Activity Diagram Lihat Data Ras Kucing**

### 3.1.4.6 Sequence Diagram

Squence diagram merupakan gambaran interaksi antar masing-masing objek pada setiap use case dalam urutan waktu. Interaksi ini berupa pengiriman serangkaian data antar objek-objek yang saling berinteraksi.

1. Sequence Diagram Tambah Data Ras Kucing

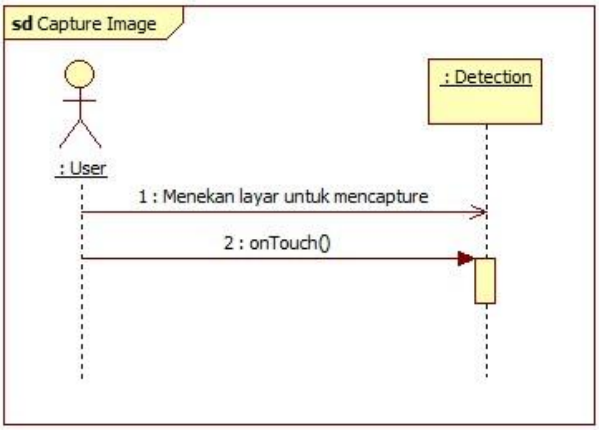
   Squence diagram tambah data ras kucing menunjukkan rangkaian pesan yang dikirim antara objek pengguna dengan objek lainnya dalam proses tambah data ras kucing. Squence diagram tambah data ras kucing dapat dilihat pada **Gambar 3.16**.




**Gambar 3.16 Sequence Diagram Tambah Data Ras Kucing**

2. Sequence Diagram Capture Image

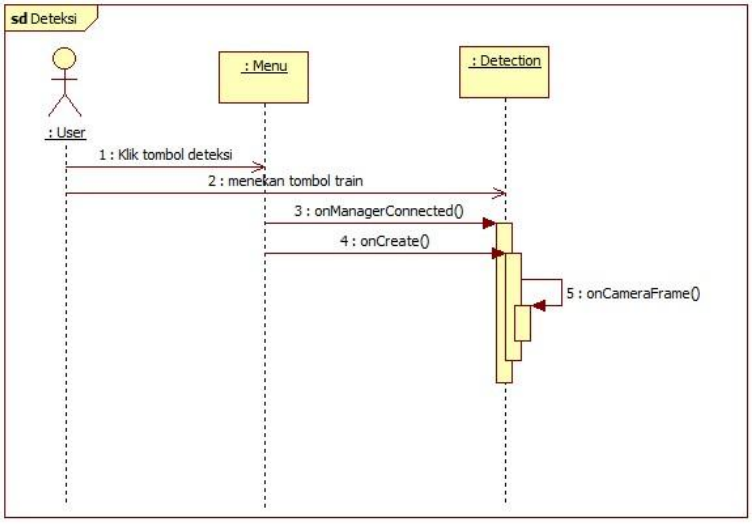
   Squence diagram capture image menunjukkan rangkaian pesan yang dikirim antara objek pengguna dengan objek lainnya dalam proses capture image . Squence diagram capture image dapat dilihat pada **Gambar 3.17**.




**Gambar 3.17 Sequence Diagram Capture Image**

3. Sequence Diagram Deteksi

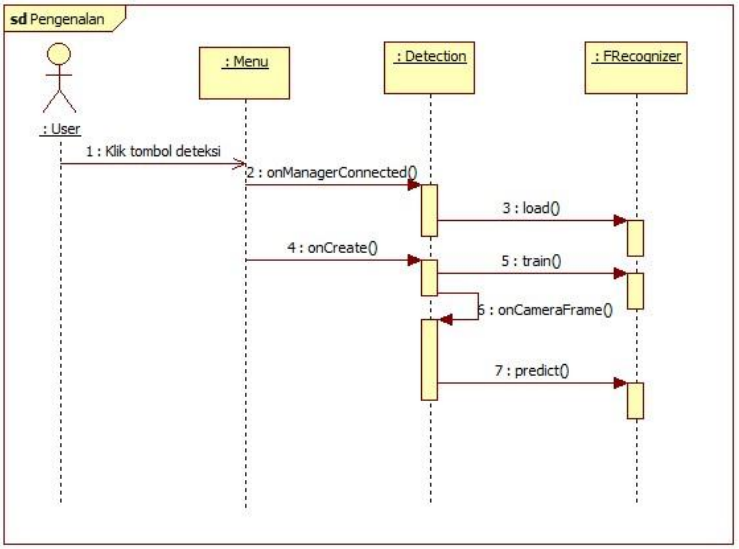
Squence diagram deteksi menunjukkan rangkaian pesan yang dikirim antara objek pengguna dengan objek lainnya dalam proses deteksi. Squence diagram deteksi dapat dilihat pada **Gambar 3.18**.




**Gambar 3.18 Sequence Diagram Deteksi**

4. Sequence Diagram Pengenalan Kucing

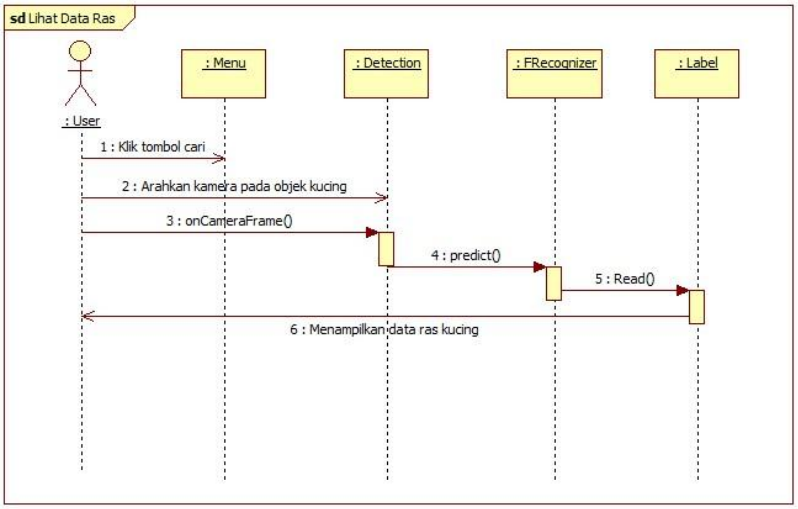
Squence diagram deteksi menunjukkan rangkaian pesan yang dikirim antara objek pengguna dengan objek lainnya dalam proses deteksi. Squence diagram deteksi dapat dilihat pada Gambar 3.19.




**Gambar 3.19 Sequence Diagram Pengenalan Kucing**

5. Sequence Diagram Lihat Data Ras Kucing

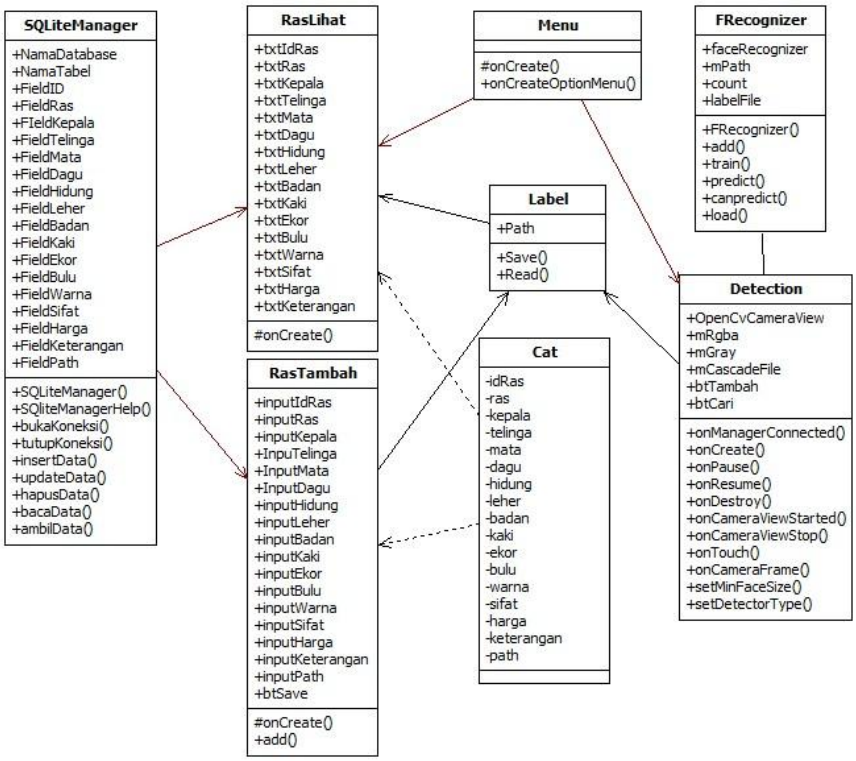
Squence diagram lihat data ras kucing menunjukkan rangkaian pesan yang dikirim antara objek pengguna dengan objek lainnya dalam proses lihat data ras kucing. Squence diagram lihat data ras kucing dapat dilihat pada **Gambar 3.20**.




**Gambar 3.20 Sequence Diagram Lihat Data Ras Kucing**

### 3.1.4.7 Class Diagram

Class diagram digunakan untuk menunjukan hubungan sebuah class dari dalam suatu sistem. Class diagram memberikan gambaran secara menyeluruh tentang object dan class serta relasinya pada arsitektur sistem yang dibangun. Adapun gambaran *class diagram* yang dibangun dapat dilihat pada Gambar 3.21.




**Gambar 3.21 Class Diagram**

### 3.1.5 Analisis Basis Data

Analisis basis data yang digunakan pada sistem ini yaitu *SQLite* yang akan digunakan untuk menyimpan data-data dari setiap objek. Berikut ini merupakan penjelasannya.

#### 3.1.5.1 Perancangan Tabel

Berikut adalah gambaran dari tabel kucing yang digunakan untuk menyimpan-data data kucing yang akan disimpan pada database SQLite seperti pada Gambar 3.22.

| Kucing | | |
|---|---|---|
| idRas | int | <pk> |
| ras | varchar(20) | |
| bulu | varchar(20) | |
| bentuk kepala | varchar(20) | |
| telinga | varchar(20) | |
| mata | varchar(20) | |
| dagu | varchar(20) | |
| hidung | varchar(20) | |
| leher | varchar(20) | |
| badan | varchar(20) | |
| kaki | varchar(20) | |
| ekor | varchar(20) | |
| warna | varchar(20) | |
| sifat | varchar(20) | |
| harga | varchar(20) | |
| keterangan | varchar(50) | |
| path | varchar(50) | |

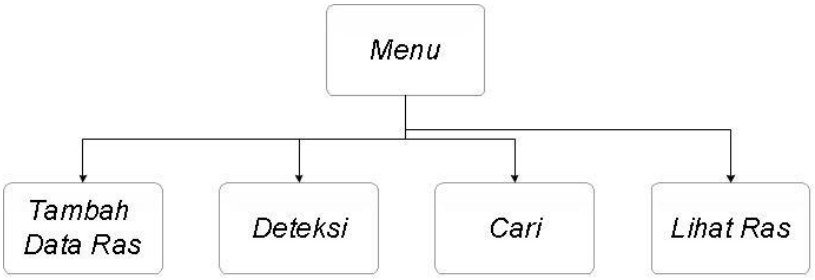
**Gambar 3.22 Perancangan Tabel**

## 3.2 Perancangan Sistem

Perancangan merupakan bagian dari metodologi pembangunan suatu perangkat lunak yang harus dilakukan setelah melalui tahapan analisis. Pada bagian ini akan dijelaskan perancangan sistem yang dimaksudkan untuk menggambarkan perbedaan antara sistem yang sedang berjalan dengan sistem yang diusulkan.

### 3.2.1 Perancangan Struktur Menu

Struktur menu dirancang sesuai dengan level pengguna sistem. Pengguna perangkat lunak ini adalah *user*. Adapun struktur menunya dapat dilihat pada Gambar 3.23.



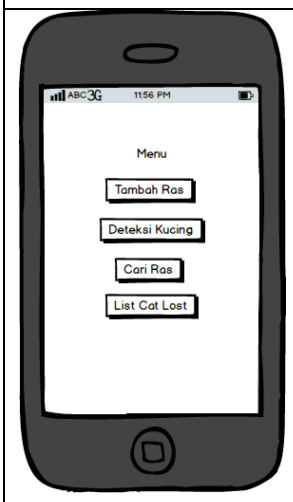

**Gambar 3.23 Perancangan Struktur Menu**

### 3.2.2 Perancangan Antarmuka Perangkat Lunak

*Interface* atau antar muka merupakan tampilan dari suatu perangkat lunak yang berperan sebagai media komunikasi yang digunakan sebagai sarana berdialog antara perangkat lunak dengan *user*. Sistem yang akan dibangun diharapkan menyediakan *interface* yang mudah dipahami dan digunakan oleh *user*.

**1. Perancangan Form Menu**

Gambaran umum dari perancangan antarmuka form menu dapat dilihat pada **Gambar 3.24**.

| T01 – Menu | | |
|---|---|---|
|  | - Klik Tambah ras maka akan menuju T03<br>- Klik Detection maka akan menuju T03<br>- Klik Cari maka akan menuju T03<br>- Klik List maka akan menuju T04 | - Ukuran layar menyesuaikan<br>- Font family sans, ukuran dan warna menyesuaikan<br>- Warna background #ffffff |

**Gambar 3.24 Perancangan Form Menu**

**2. Perancangan Form Tambah Data Ras Kucing**

Gambaran umum dari perancangan antarmuka form tambah data ras kucing dapat dilihat pada **Gambar 3.25**.

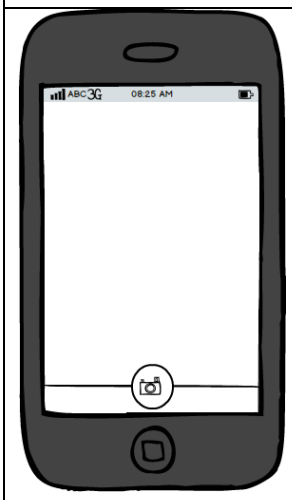
| T02 – Tambah Data Ras Kucing |
|---|

| | | |
|---|---|---|
|  | - Klik save maka akan muncul M02 | - Ukuran layar menyesuaikan<br>- Font family sans, ukuran dan warna menyesuaikan<br>- Warna background #ffffff |

**Gambar 3.25 Perancangan Form Tambah Data Ras Kucing**

### 3. Perancangan Form Deteksi

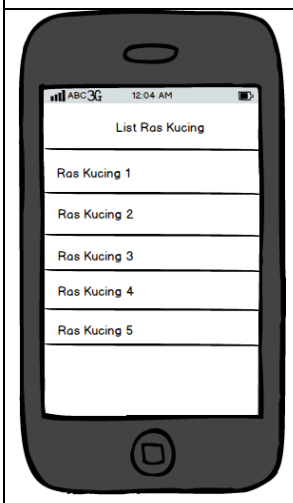
Gambaran umum dari perancangan antarmuka form deteksi dapat dilihat pada **Gambar 3.26**.

| T03 – Deteksi | | |
|---|---|---|
|  | - Capture kucing maka akan menuju T02<br>- Menemukan ras kucing maka akan muncul M01 dan jika setuju menuju T05 | - Ukuran layar menyesuaikan<br>- Font family sans, ukuran dan warna menyesuaikan<br>- Warna background #ffffff |

**Gambar 3.26 Perancangan Form Deteksi**

### 4. Perancangan Form List Data Ras Kucing

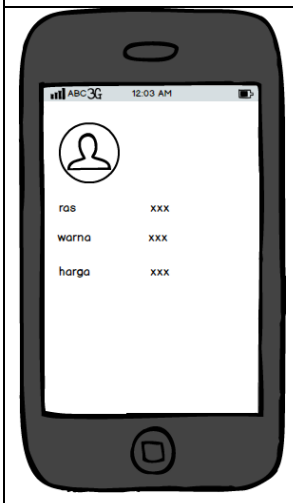
Gambaran umum dari perancangan antarmuka list data ras kucing dapat dilihat pada **Gambar 3.28**.

| T04 – Detail Kucing Hilang | | |
|---|---|---|
|  | - Klik salah satu list maka akan menuju T05 | - Ukuran layar menyesuaikan<br>- Font family sans, ukuran dan warna menyesuaikan<br>- Warna background #ffffff |

**Gambar 3.27 Perancangan Form Lihat Data Ras Kucing**

### 5. Perancangan Form Lihat Data Ras Kucing

Gambaran umum dari perancangan antarmuka lihat data ras kucing dapat dilihat pada **Gambar 3.28**.

| T05 – Detail Kucing Hilang | | |
|---|---|---|
|  | - Klik Kembali maka akan menuju T04 | - Ukuran layar menyesuaikan<br>- Font family sans, ukuran dan warna menyesuaikan<br>- Warna background #ffffff |

**Gambar 3.28 Perancangan Form Lihat Data Ras Kucing**

### 3.2.3 Perancangan Pesan

Perancangan pesan mendeskripsikan rencana tampilan dari setiap pesan yang akan digunakan. Berikut ini gambaran dari perancangan pesan yang terlihat pada **Gambar 3.29** sampai dengan **Gambar 3.30**.

**1. Perancangan Pesan Berhasil Mendeteksi**

Pesan ini muncul pada saat sistem berhasil melakukan mengidentifikasi ras. Adapun gambaran umum dari perancangan pesan berhasil mendeteksi yang dapat dilihat pada **Gambar 3.30**.

M01 – Pesan berhasil mendeteksi




**Gambar 3.29 Perancangan Pesan Berhasil Mendeteksi**

**2. Perancangan Pesan Simpan Data**

Pesan ini muncul pada saat sistem menyimpan data. Adapun gambaran umum dari perancangan pesan simpan data yang dapat dilihat pada **Gambar 3.30**

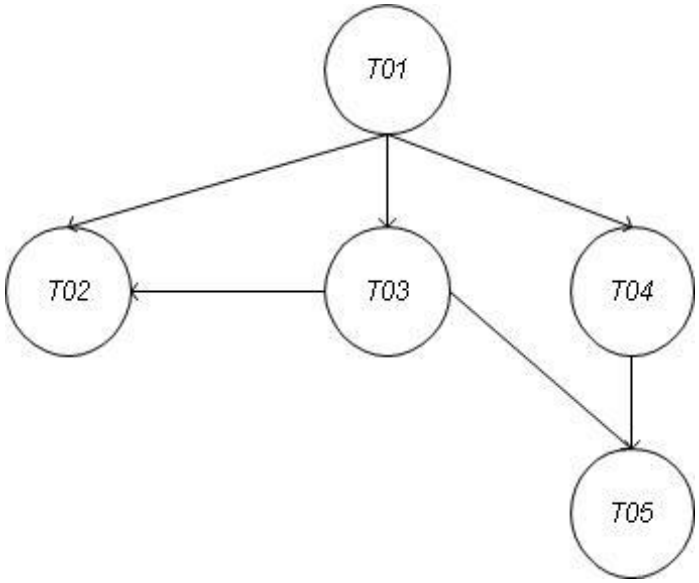
M02 – Pesan Simpan Data




**Gambar 3.30 Perancangan Pesan Simpan Data**

### 3.2.4 Jaringan Semantik

Jaringan semantik merupakan gambaran hubungan antara form dan pesan. Jaringan semantik yang dijelaskan pada **Gambar 3.31**.




**Gambar 3.31 Jaringan Semantik**