BAB 3
ANALISIS DAN PERANCANGAN SISTEM

3.1.Analisis Sistem

Analisis sistem merupakan proses penguraian suatu sistem informasi ke dalam komponen-komponen yang lebih sederhana yang bertujuan untuk mengidentifikasi serta mengevaluasi masalah-masalah dan hambatan-hambatan yang mungkin terjadi serta kebutuhan-kebutuhan yang diharapkan menjadi acuan yang dapat mengarah kepada solusi yang terbaik. Pada analisis sistem akan dijelaskan beberapa macam analisis yang berhubungan dengan simulasi yang akan dibangun seperti analisis masalah, analisis penggunaan, analisis algoritma yang digunakan, analisis kebutuhan fungsional dan non fungsional.

3.1.1. Analisis Masalah

Berdasarkan dari identifikasi masalah yang sudah didapat, maka pada sub bab ini akan dijabarkan secara lebih terperinci tentang masalah yang ditemukan dalam penelitian ini. Berikut masalah yang ditemukan:

1. Dalam dunia grafologi banyak cara untuk mengenali karakter orang melalui goresan-goresan tangan seseorang, seperti dari pola tanda tangan, jarak antar huruf, kemiringan huruf dan sebagainya. Salah satu fitur yang akan digunakan dalam penelitian adalah pengenalan tulisan tangan berdasarkan tanda tangan.
2. Penelitian pengenalan pola tanda tangan akan menggunakan jaringan saraf tiruan *Learning Vector Quantization*. Dalam penelitian ini akan menggunakan proses pembelajaran sebagai acuan dan sebagai perbandingan dengan hasil dari pengenalan pola tanda tangan yang dilakukan pada tanda tangan seseorang.

3.1.2. Analisis Simulasi

Simulasi pengenalan pola tanda tangan adalah simulasi pengenalan pola dari tanda tangan yang menghasilkan karakter seseorang. Simulasi ini bekerja dengan cara menganalisis citra dari hasil *scanning* tanda tangan.

Simulasi ini mengimplementasikan algoritma *Learning Vector Quantization* untuk mendapatkan data latihan yang nantinya akan dibandingkan dengan hasil dari analisis simulasi.

3.2.Analisis Penyelesaian Masalah

Dalam menyelesaikan permasalah ini, maka simulasi ini dibangun. Simulasi ini bekerja dengan cara meng*scanning* kertas A4 yang berisi tanda tangan dan menghasilkan citra dengan format JPG atau JPEG atau PNG yang akan diolah dan menghasilkan karakter orang berdasarkan dari analisis pola tanda tangan yang terdapat dalam simulasi. Karakter orang berdasarkan ciri pola tanda tangan yang dimilikinya dapat dilihat pada tabel 3.1

Tabel 3.1 Deskripsi dua fitur dari pola tanda tangan *[1]*

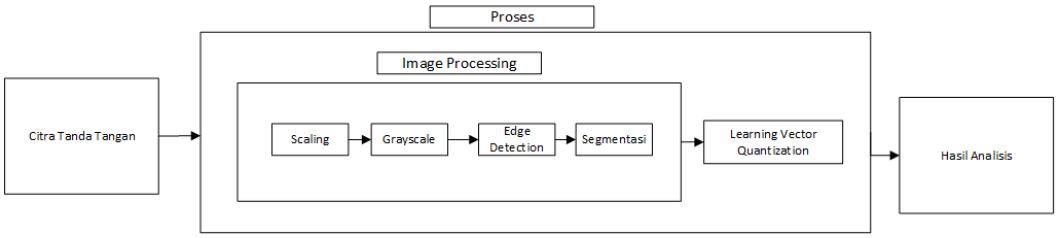
No	Fitur	Ciri	Kepribadian
1	Lengkung	Lengkung Mundur	Nyaman akan masa lalu
		Lengkung Tajam	Mampu memformulasi pikiran secara tajam
		Lengkung Lembut	Hati-hati, ramah, diplomasi
2	Coretan Akhir	Menaik	Terbuka, pandangan ke depan, keinginan maju, percaya diri
		Menurun	Kurang semangat, berfikir realistis, kurang percaya diri, mudah putus asa

3.2.1. Analisis Data Masukan

Data masukan yang dibutuhkan untuk simulasi ini adalah sebuah citra. Citra yang didapat berasal dari hasil scanning tanda tangan. Citra yang digunakan untuk simulasi ini berformat '.jpg' atau '.jpeg' atau juga '.png'.

3.2.2. Analisis Proses

Secara umum simulasi ini memiliki beberapa proses dalam melakukan analisis terhadap citra yang diinputkan kedalam simulasi. Setiap proses memiliki peran masing-masing untuk dapat mengenali citra yang diinputkan. Proses-proses tersebut dapat dilihat pada gambar 3.1 :



Gambar 3.1 Proses-proses yang terjadi pada simulasi

3.2.2.1.Scaling

Scaling adalah proses mengubah ukuran citra agar seluruh citra yang digunakan memiliki ukuran yang sama. Pada proses ini ukuran citra akan diubah menjari ukuran 50 x 100 pixel. Pada matlab fungsi ini diimplementasikan dengan sintak

```
x=imread(filename);
y1=imresize(x,[50 100]);
```

3.2.2.2.Grayscale

Proses *grayscale* adalah mengolah citra normal (RGB) menjadi citra abu-abu. Untuk menjalankan proses perubahan citra dapat menggunakan persamaan sebagai berikut:

$$\boldsymbol{Grayscale} = \frac{\boldsymbol{R} + \boldsymbol{G} + \boldsymbol{B}}{\boldsymbol{3}} \qquad \textbf{3.1}$$

Cara kerja dari proses ini yaitu:

1. Pada citra akan diambil nilai *Red, Green* dan *Blue* untuk dimasukan dalam persamaan diatas.
2. Setelah didapat dan dimasukan dalam persamaan diatas maka akan diperoleh nilai *grayscale.*
3. Nilai *grayscale* yang didapat akan menggantikan nilai RGB pada setiap pixel.

Hasil dari proses diatas dapat dilihat pada gambar 3.2.



Gambar 3.2 Gambar seterlah proses grayscale

Pada matlab fungsi ini diimplementasikan dengan sintak `gray=rgb2gray(y1);`

3.2.2.3.Proses Edge Detection

Pada proses ini citra dari proses *grayscale* akan dideteksi tepinya dan akan menghasilkan nilai biner. Metode *edge detection* yang digunakan adalah metode *Canny*. Alur kerja dari proses ini yaitu:

a. Citra *grayscale* diambil untuk proses deteksi
b. Citra yang dianggap tepi dirubah nilainya menjadi nilai biner
c. Setelah semua tepi selesai dideteksi, setiap pixel disusun kembali sehingga menjadi citra semula.

Hasil dari proses diatas dapat dilihat pada gambar 3.3.



Gambar 3.3 Gambar setelah proses Edge Detection

Pada matlab fungsi ini diimplementasikan dengan sintak

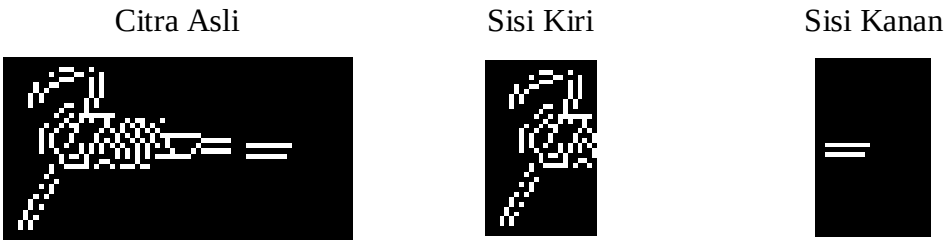
```
ed=edge(gray,'canny',0.06);
```

3.2.2.4.Segmentation

Pada proses ini citra dari proses *edge detection* akan dipotong menjadi dua area yaitu area kanan dan kiri. Proses segmentasi ini bertujuan untuk memperoleh citra dengan fitur Lengkung pada sisi kiri dan citra dengan fitur coretan akhir pada sisi kana. Alur kerja dari proses ini yaitu:

a. Citra hasil *grayscale* dibaca nilai *width*.
b. Pemotongan citra dilakukan dengan kordinat (0 0 50 20) untuk sisi kiri dan (0 81 50 100) untuk sisi kanan.

Hasil dari proses diatas dapat dilihat pada gambar 3.4.



Gambar 3.4 Citra Asli, Sisi Kiri, Sisi Kanan

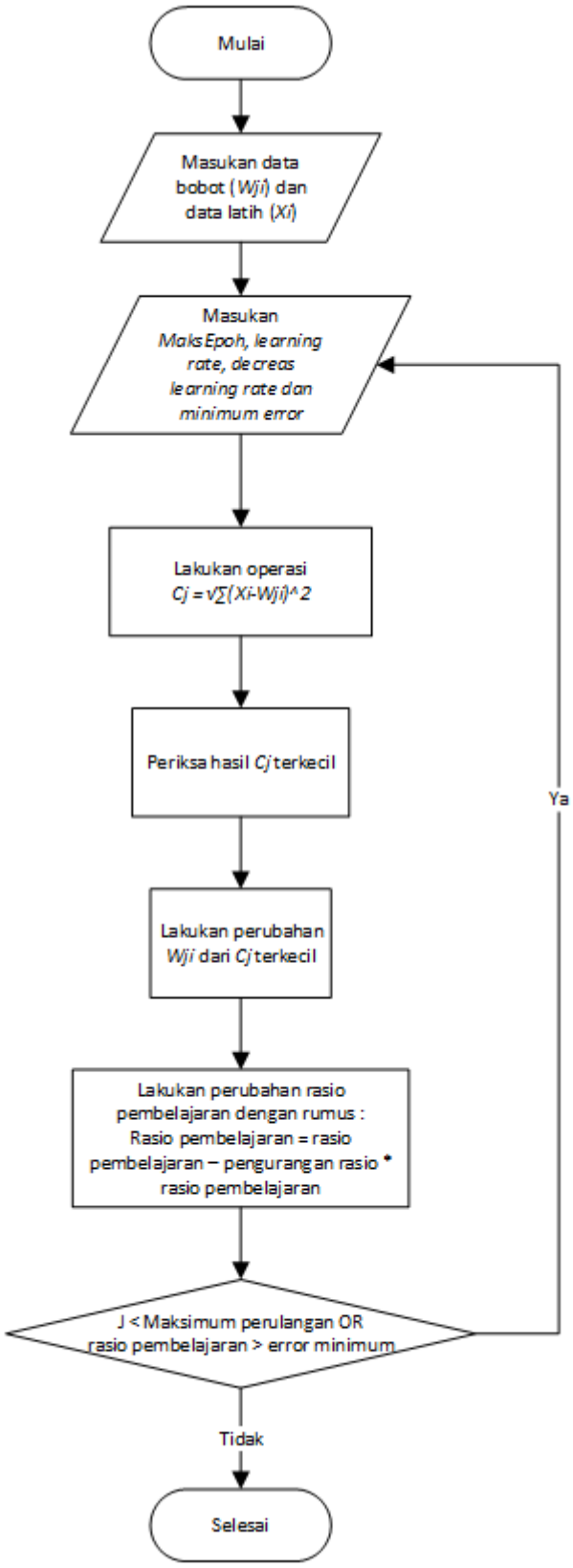
Pada matlab fungsi ini diimplementasikan dengan sintak

```
crp1 = imcrop(x1,[0 0 20 50]);
crp2 = imcrop(x1,[81 0 100 50]);
```

crp1 untuk segmentasi bagian kiri dan crp2 untuk segmentasi bagian kanan.

3.2.2.5.Proses Pembelajaran Algoritma *Learning Vector Quantization*

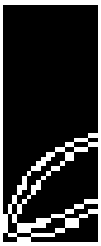
Analisis algoritma yang dilakukan dalam penelitian ini adalah meneliti cara kerja dari algoritma *Learning Vector Quantization* pada simulasi yang akan dibangun. Pada gambar 3.5 berikut ini adalah ilustrasi dari proses pelatihan algoritma *Learning Vector Quantization*:

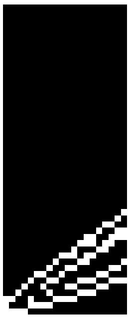


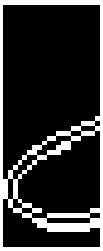
Gambar 3.5 *Flowchart* Proses Pembelajaran LVQ

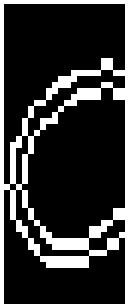
a. Setiap kelas (kelas 1 untuk lengkung mundur, kelas 2 untuk lengkung tajam dan kelas 3 untuk lengkung lembut) diambil salah satu array vektornya dan dijadikan sebagai inisialisasi bobot. Sedangkan array lainnya menjadi data masukan. Penjabaran tersebut dapat dilihat pada tabel 3.2.

Tabel 3.2 Inisialisasi Bobot dan Data Masukan Proses Pembelajaran LVQ

Citra	Kelas	Matrik	
	1	00000000000000000000 00000000000000000011 00000000000000011100 00000000000001100011 00000000000110011100 00000000011001100000 00000000110001100000 00000001100010000000 00000011001100000000 00000100010000000000 00001000100000000000 00001001000000000000 00011011000000000000 00010010000000000000 00100100000000000000 00101000000000000111 01010000000000011000 01010000000001100011	Bobot (W_1)

		10100000001110001100 10100001110001110000 10011111100001110000 10011100000110000000 01100011111000000000 00011100000000000000	
	2	00000000000000000000 00000000000000000001 00000000000000000010 00000000000000001100 00000000000000010011 00000000000001100111 00000000000011101100 00000000000100011000 00000000011101100000 00000000100011000001 00000001001100000110 00000110010000011000 00001001100011100111 00010010001100011000 00100001000011100000 11001000111100000000	Bobot (W_2)

		10001111000000000000 11110000000000000000	
	3	00000000000000000000 00000000000000000000 00000000000000000000 00000000000000000000 00000000000000000000 00000000000000000000 00000000000000000000 00000000000000000000 00000000000000000000 00000000000000000000 00000000000000000000 00000000000000000000 00000000000000000000 00000000000000000000 00000000000000000000 00000000000000000000 00000000000000000000 00000000000000000000 00000000000000000000 00000000000000000001 00000000000000001110 00000000000000110001 00000000000111001110 00000000011000110000 00000000100011000000 00000011000111000000 00000110011100000000 00001001100000000000 00010010000000000000 00100100000000000000 01001000000000000000 01010000000000000000 10100000000000000000 10100000000000000000 10100000000000000000 10100000000000000000 01010000000000000000 01001100000000000000 00110011000000000011 00001100011111111100 00000011100000000011 00000001111111111111 00000000011111111100 00000000000000000000 00000000000000000000 00000000000000000000	Bobot (W_3)

	3	00000000000000000000 00000000000000000000 00000000000000000000 00000000000000000000 00000000000000000000 00000000000000000000 00000000000000000000 00000000000000000000 00000000000000000000 00000000000000001100 00000000000000001100 00000000000111110011 00000000011100000000 00000000111000001100 00000011100011110011 00000010001111110011 00000110001100000000 00001000010000000000 00001000100000000000 00010011100000000000 00010010000000000000 00010100000000000000 00101000000000000000 01101000000000000000 01001000000000000000 01010000000000000000 01010000000000000000 01010000000000000000 01010000000000000000 01010000000000000000 10100000000000000000 01010000000000000000 01010000000000000000 01010000000000000000 01001000000000000011 01001000000000000011 00100100000000001100 00110010000000111000 00010011000000110001 00001001111111000110 00001100111111000110 00000100000000111000 00000011111111000000 00000011111111000000 00000000000000000000 00000000000000000000 00000000000000000000 00000000000000000000 00000000000000000000 00000000000000000000	Masukan (X_1)

b. Tentukan nilai maksimum perulangan, eror minimum, rasio pembelajaran dan pengurangan rasio. Pada contoh ini digunakan batasan sebagai berikut [19]:

   Maksimum perulangan = 100

   Eror minimum = 0,01

   Rasio pembelajaran = 0,05

   Pengurangan rasio = 0,1

c. Perhitungan d sampai f dilakukan selama perulangan masih lebih kecil dari maksimum perulangan atau rasio pembelajaran masih lebih besar dari eror minimum.

d. Untuk setiap data masukan (X) dihitung terhadap setiap data bobot (W) dengan menggunakan persamaan (2.11) dengan variable:

   j = [1,2,3]

   $n = 828$

   Sehingga persamaan (2.11) menjadi

   $$C1 = \sqrt{\sum_{i=1}^{828}(Xi - W1i)^2} = \sqrt{((0-0)^2 + (0-0)^2 + \cdots + (0-0)^2)} = 13{,}6748$$

   Lakukan hal yang sama hingga j = 3. Hasil dari perhitungan diatas dapat dilihat pada tabel 3.3 berikut ini :

Tabel 3.3 Tabel hasil perhitungan vektor

Vektor	Nilai
$C1$	13,6748
$C2$	12,8452
$C3$	13,7113

   Dari hasil perhitungan, didapat bahwa bobot terkecil dimiliki oleh *C2*

e. Dilihat jarak (C) terkecil yang diperoleh pada perhitungan terhadap X_1. Kemudian diperiksa apakah target X_1 sesuai dengan kelas bobot dengan jarak terkecil. Hasil dari perhitungan diatas menunjukan bahwa jarak terkecil ada pada bobot ke-2. Target X_1 adalah kelas 3 dan kelas bobot ke-2 juga kelas 2, hal ini berarti target tidak sesuai dengan hasil perhitungan.

f. Nilai bobot terkecil akan dilakukan perubahan nilai bobot dengan persamaan (2.12) jika kelas target sesuai dengan kelas bobot atau persamaan (2.13) jika

kelas target tidak sesuai kelas bobot. Dalam contoh perhitungan ini, hasil dari perhitungan kelas target tidak sesuai dengan kelas bobot dengan jarak terkecil, sehingga perhitungannya adalah sebagai berikut:

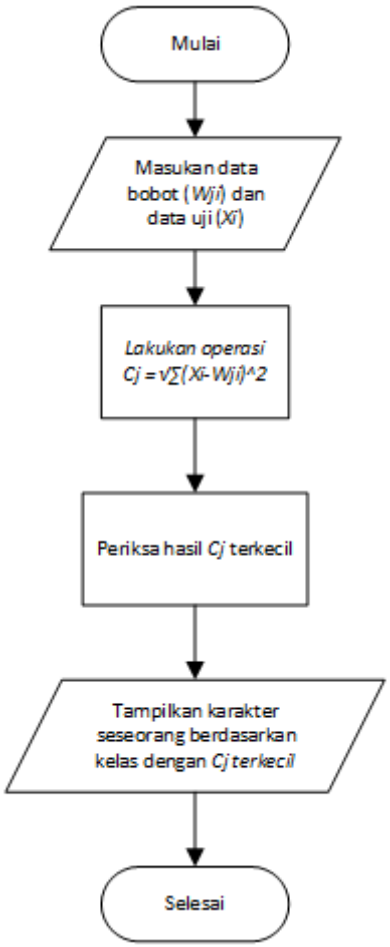
$W_{1.1}$ = 0 - 0,05 * (0 – 0) = 0
$W_{1.2}$ = 0 - 0,05 * (0 – 0) = 0
$W_{1.3}$ = 0 - 0,05 * (0 – 0) = 0
...
$W_{1.828}$ = 0 + 0,05 * (0 – 0) = 0

g. Langkah d sampai f dikerjakan terhadap semua masukan (X), setelah selesai dilakukan perhitungan perubahan nilai rasio dengan persamaan (2.14).
*rasio = 0,05 – 0,1 * 0,05 = 0,045*

h. Perhitungan diulang hingga perulangan berhenti dan mendapatkan matriks nilai bobot terakhir. Pada contoh ini perulangan berhenti pada perulangan ke-15 dengan hasil akhir bobot yang dapat dilihat pada lampiran D.

3.2.2.6.Proses Pengenalan Algoritma *Learning Vector Quantization*

Proses pengenalan dengan menggunakan algoritma *Learning Vector Quantization* menggunakan persamaan yang sama dengan perhitungan bobot pada proses pembelajaran *Learning Vector Quantization*. Gambar 3.6 berikut menunjukan langkah-langkah pengenalannya:



Gambar 3.6 *Flowchart* Proses Pengenalan LVQ

1. Data test dihitung terhadap setiap data bobot (W) dengan dengan menggunakan persamaan (2.11)

   j = [1,2,3]

   n = *828*

   Sehingga persamaan (2.11) menjadi

   $$C1 = \sqrt{\sum_{i=1}^{828}(Xi - W1i)^2} = \sqrt{((0-0)^2 + (0-0)^2 + \cdots + (0-0)^2)} = 13{,}6748$$

   Lakukan hal yang sama hingga j = 3. Hasil dari perhitungan diatas dapat dilihat pada tabel 3.5 berikut ini :

Tabel 3.4 Tabel hasil perhitungan vektor

Vektor	Nilai
$C1$	13,6748
$C2$	13,4360
$C3$	12.3177

Dari hasil perhitungan, didapat bahwa bobot terkecit dimiliki oleh *C3*

2. Dilihat jarak (C) terkecil yang diperoleh pada perhitungan terhadap test. Kelas bobot yang memiliki jarak terkecil menjadi hasil dari proses pengenalan. Dalam perhitungan di atas menghasilkan jarak terkecil ada pada bobot ke-3 (W_3) dan W_3 merupakan kelas 3, sehingga data test memiliki fitur lengkung lembut. Jadi data test sudah cocok dari data pembelajaran.

3.3.Analisis Kebutuhan Non Fungsional

Analisis kebutuhan non fungsional dilakukan untuk menentukan spesifikasi kebutuhan dari simulasi yang akan dibangun. Spesifikasi tersebut akan berisi hal-hal yang harus ada pada simulasi yang akan diimplementasikan.

3.3.1. Analisis Kebutuhan Perangkat Lunak

Analisis perangkat lunak dalam pembuatan simulasi ini dilakukan agar dapat mengoptimalkan implementasi dari simulasi yang akan dibuat. Berdasarkan hasil perkiraan, spesifikasi perangkat lunak yang direkomendasikan untuk mengoperasikan simulasi ini adalah sebagai berikut :

1. Sistem operasi Windows 7 *Professional* 64bit.
2. MATLAB R2013b (sudah terinstall *image processing toolbox & neural network toolbox*).

3.3.2. Analisis Kebutuhan Perangkat Keras

Analisis kebutuhan perangkat keras dilakukan untuk mempermudah proses perancangan dan implementasi. Berdasarkan hasil perkiraan, spesifikasi perangkat

keras yang direkomendasikan untuk mengoprasikan simulasi ini adalah sebagai berikut :

1. *Processor* Intel Core I3 2.3GHz.
2. *Memory* 2GB DDR3.
3. *Input keyboard* dan *Mouse.*
4. Monitor 14.0” dengan resolusi 1366 x 768.

3.3.3. Analisis Pengguna

Untuk menjalankan simulasi yang dibangun dibutuhkan spesifikasi dari pengguna agar dapat menjalankan fungsi-fungsi yang terdapat pada simulasi. Hanya terdapat satu jenis pengguna yang dibutuhkan yaitu penguji. Adapun kemampuan pengguna yang dibutuhkan untuk menjalankan simulasi adalah sebagai berikut :

1. Dapat mengoperasikan komputer.
2. Dapat mengoperasikan Matlab.

3.4.Analisis Kebutuhan Fungsional

Analisis kebutuhan fungsional dilakukan untuk mengetahui spesifikasi kebutuhan fungsional. Spesifikasi kebutuhan fungsional adalah spesifikasi yang rinci terhadap hal-hal yang akan dilakukan pada saat implementasi sistem. Berikut ini adalah kebutuhan fungsional sistem :

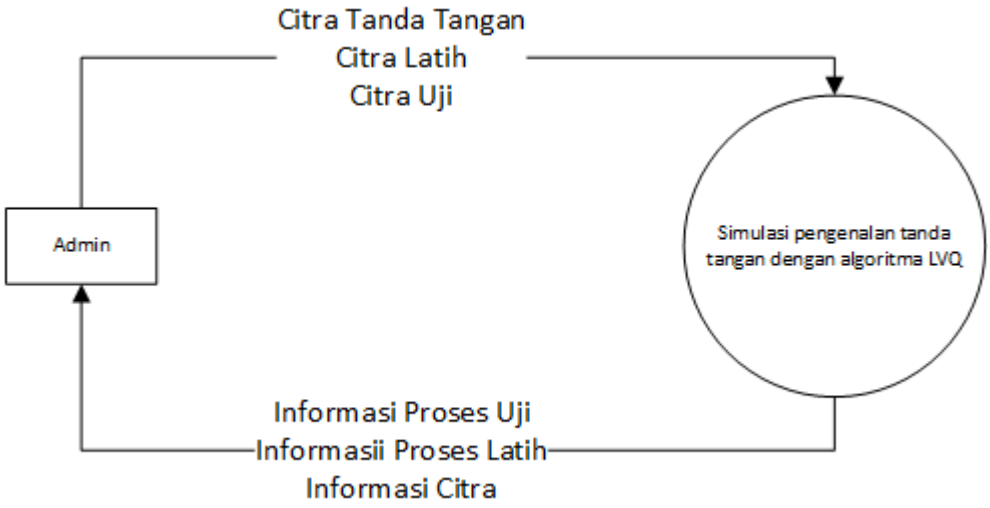
1. *Context Diagram*
2. *Data Flow Diagram* (DFD)
3. *Process Spesification* (PSPEC)

3.4.1. Context Diagram

Kebutuhan-kebutuhan pemakai yang ada akan diubah kedalam model-model yang terstruktur yang dapat dimengerti dan menjadi suatu referensi untuk tahapan pengembangan selanjutnya. Salah satu model tersebut adalah Diagram Konteks (*Context Diagram*) yang menggambarkan hubungan suatu sistem dengan lingkungan luar atau pengguna. Diagram konteks menggambarkan suatu sistem secara global, termasuk aliran data masukan (*input*) ke proses kegiatan (*process*),

dari proses ke proses lain dan dari proses ke luaran (*output*) menjadi sebuah informasi yang terpadu.

Untuk memudahkan dalam penggembangan sistem maka terlebih dahulu dibuat diagram konteks terhadap sistem yang akan dibangun. Pada gambar 3.7 berikut ini adalah ilustrasi diagram konteks dari simulasi yang dibangun.



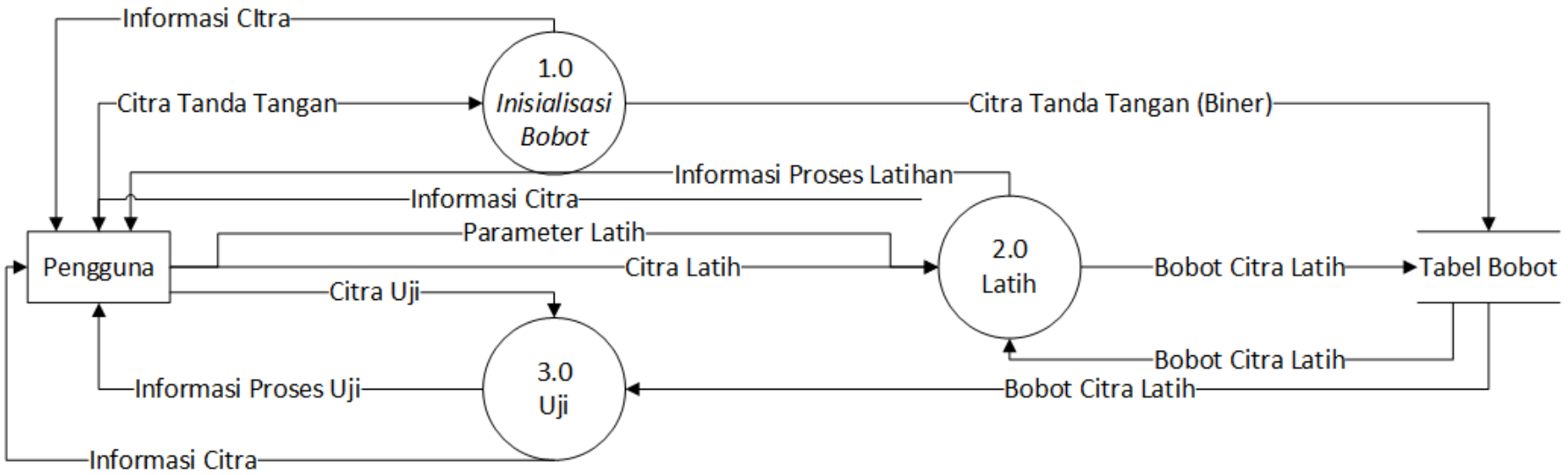
Gambar 3.7 Diagram Konteks

3.4.2. Data Flow Diagram (DFD)

Data Flow Diagram (DFD) adalah suatu model logika data atau proses yang dibuat untuk menggambarkan dari mana asal data dan kemana tujuan data yang keluar dari sistem, dimana data disimpan, proses apa saja yang menghasilkan data tersebut dan interaksi antar data yang tersimpan dan proses yang dikenakan pada data tersebut.

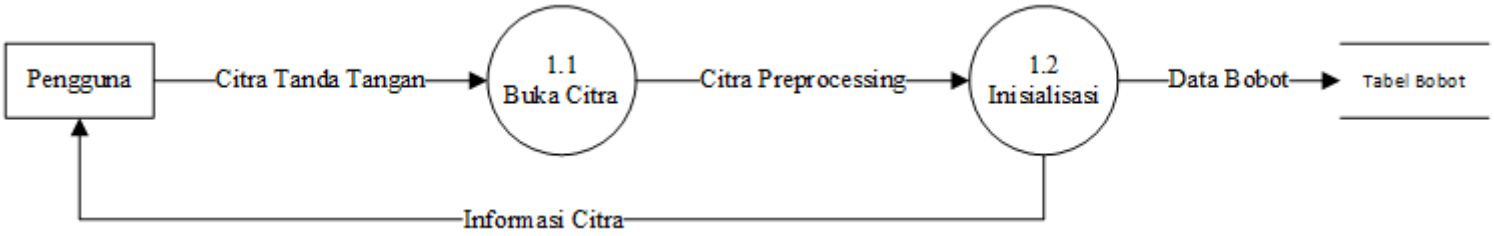
Data Flow Diagram (DFD) sering digunakan untuk menggambarkan suatu sistem yang telah ada atau sistem baru yang akan dikembangkan secara logika tanpa mempertimbangkan lingkungan fisik dimana data tersebut mengalir atau dimana data tersebut akan disimpan. Berikut adalah *Data Flow Diagram* (DFD) dari sistem yang akan dibangun digambarkan pada gambar 3.8 :

3.4.2.1.Data Flow Diagram Level 1

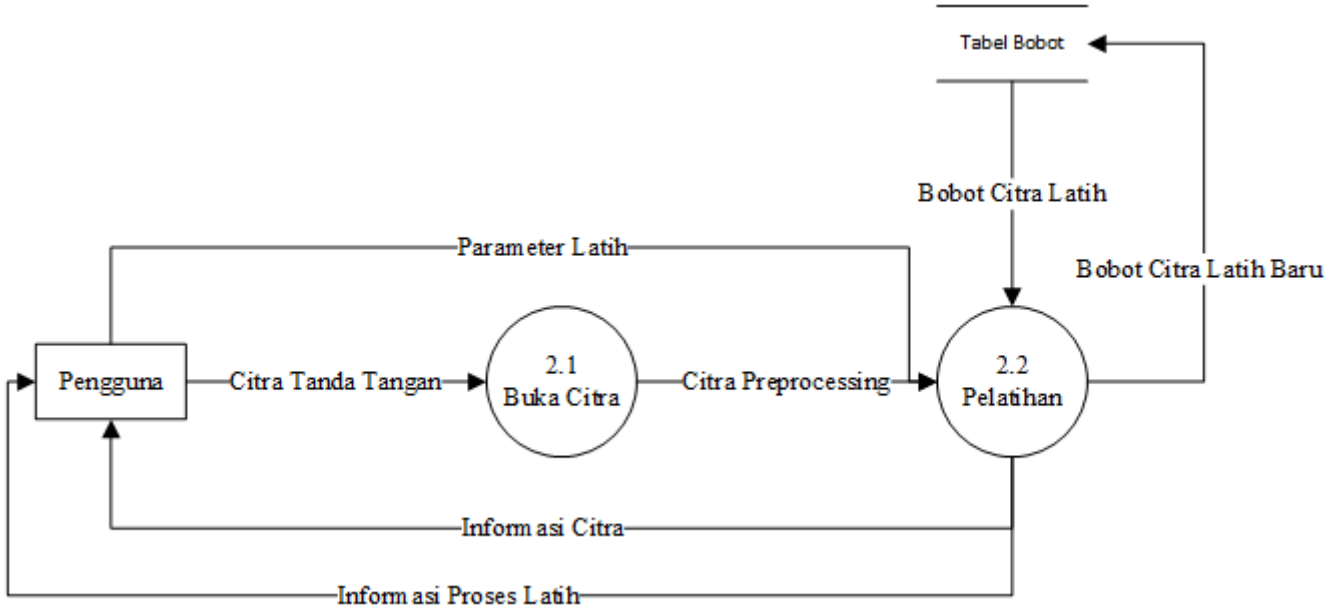


Gambar 3.8 Data Flow Diagram Level 1

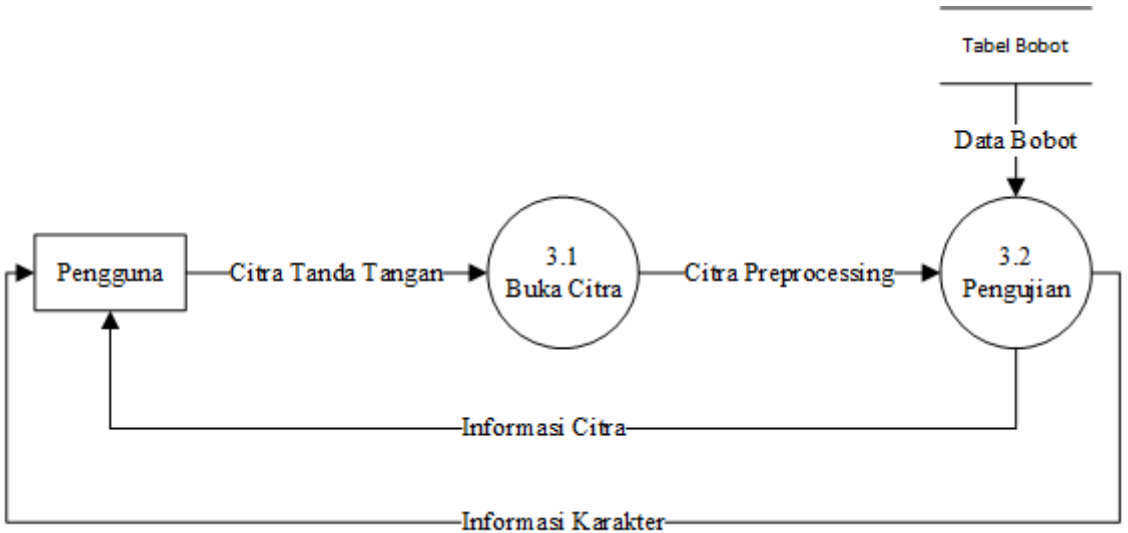
3.4.2.2.Data Flow Diagram Level 2



Gambar 3.9 DFD Level 2 Inisialisasi Bobot



Gambar 3.10 DFD Level 2 Latih



Gambar 3.11 DFD Level 2 Uji

3.4.3. Process Spesification (PSPEC)

Pada tabel 3.6 berikut ini dijabarkan proses-proses pada simulasi yang dibangun.

Tabel 3.5 Proses *Spesification*

No	Proses	Keterangan
1	No. Proses	1.0
	Nama Proses	Inisialisasi Bobot
	Source	Pengguna
	Input	Citra tanda tangan
	Output	Informasi citra, citra tanda tangan(biner)
	Logika Proses	1. Pengguna memasukan citra tanda tangan. 2. Sistem merubah data citra menjadi data biner untuk dimasukan pada nilai bobot 3. Sistem akan menampilkan bagian citra yang dijadikan data bobot.
2	No. Proses	2.0
	Nama Proses	Latih
	Source	Pengguna
	Input	Citra latih, bobot citra latih
	Output	Informasi citra, informasi proses latih, bobot citra latih
	Logika Proses	1. Pengguna memasukan citra tanda tangan. 2. Pengguna memasukan nilai rasio pembelajaran, pengurangan rasio pembelajaran, minimum error, maksimum perulangan dan kelas data masukan. 3. Bobot citra latih menjadi bobot pembanding pada proses latih. 4. Sistem akan menghitung vektor tiap kelas. 5. Periksa vektor bobot terkecil apakah memiliki kelas yang sama dengan data masukan? Jika sama dekatkan nilai data bobot latih dengan nilai data input, jika tidak jauhkan nilai data bobot latih dengan nilai data input. 6. Kurangi nilai rasio pembelajaran.

		7. Apakah sudah mencapai batas perulangan? Jika sudah berhenti, jika tidak kembali ke nomer 4. 8. Sistem akan menyimpan data bobot latih yang baru.
3	No. Proses	3.0
	Nama Proses	Uji
	Source	Pengguna
	Input	Citra uji, bobot citra latih
	Output	Informasi citra, informasi proses uji
	Logika Proses	1. Pengguna memasukan citra tanda tangan. 2. Bobot latih dijadikan bobot pembanding. 3. Sitem akan mencari vektor terkecil dari setiap data bobot pembanding 4. Sistem akan menampilkan info karakter kepribadian.
4	No. Proses	1.1
	Nama Proses	Buka Cittra
	Source	Pengguna
	Input	Citra tanda tangan
	Output	Citra *preprocessing*
	Logika Proses	1. Pengguna memasukan citra tanda tangan 2. Sistem akan melakukan proses *scaling*, *grayscale*, *edge detection* dan *segmentasi* 3. Sistem akan mengirim data citra *preprocessing* ke proses inisialisasi.
5	No. Proses	1.2
	Nama Proses	Inisialisasi
	Source	Proses 1.1 Buka CItra
	Input	Citra *preprocessing*
	Output	Informasi Citra, Data Bobot
	Logika Proses	1. Nilai biner pada citra *preprocessing* disimpan pada matriks.

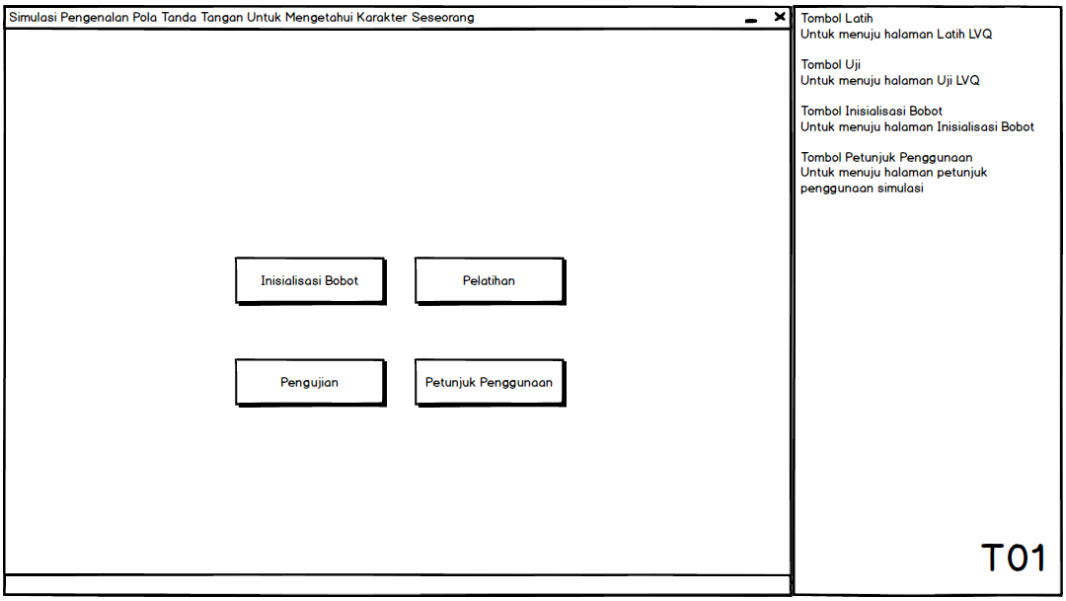
		2. Matriks yang ada disimpan untuk menjadi data bobot.
6	No. Proses	2.1
	Nama Proses	Buka Cittra
	Source	Pengguna
	Input	Citra tanda tangan
	Output	Citra *preprocessing*
	Logika Proses	1. Pengguna memasukan citra tanda tangan 2. Sistem akan melakukan proses *scaling, grayscale, edge detection* dan *segmentasi* 3. Sistem akan mengirim data citra *preprocessing* ke proses pelatihan.
7	No. Proses	2.2
	Nama Proses	Pelatihan
	Source	Proses 2.1 Buka Cira, Pengguna
	Input	Citra *preprocessing,* Bobot Citra Latih, Parameter Latih
	Output	Informasi Citra, Informasi Proses Latih, Bobot Citra Latih Baru
	Logika Proses	1. Nilai biner pada citra *preprocessing* disimpan pada matriks. 2. Sistem menampilkan informasi citra. 3. Pengguna memasukan nilai parameter latih. 4. Sistem membaca data bobot citra latih. 5. Matriks yang ada disimpan untuk menjadi data latih 6. Sistem menghitung vektor dari nilai data bobot latih dengan data latih. 7. Sistem mengubah bobot citra latih. 8. Sistem mengecek kondisi perulangan, jika selesai sistem akan menampilkan informasi citra pelatihan dan emnyimapn bobot citra latih baru, jika belum ulangi langkah ke 6.

8	No. Proses	3.1
	Nama Proses	Buka Cittra
	Source	Pengguna
	Input	Citra tanda tangan
	Output	Citra *preprocessing*
	Logika Proses	1. Pengguna memasukan citra tanda tangan 2. Sistem akan melakukan proses *scaling, grayscale, edge detection* dan *segmentasi* 3. Sistem akan mengirim data citra *preprocessing* ke proses pengujian.
9	No. Proses	3.3
	Nama Proses	Pengujian
	Source	Proses 3.1 Buka Citra
	Input	Citra *preprocessing*
	Output	Informasi Citra, Informasi Karakter
	Logika Proses	1. Nilai biner pada citra *preprocessing* disimpan pada matriks. 2. Matriks yang ada disimpan untuk menjadi data uji 3. Sistem menampilkan informasi citra. 4. Sistem membaca data bobot citra latih 5. .Sistem menghitung vektor dari nilai data bobot latih dengan data uji. 6. Sistem menampilkan informasi kepribadian

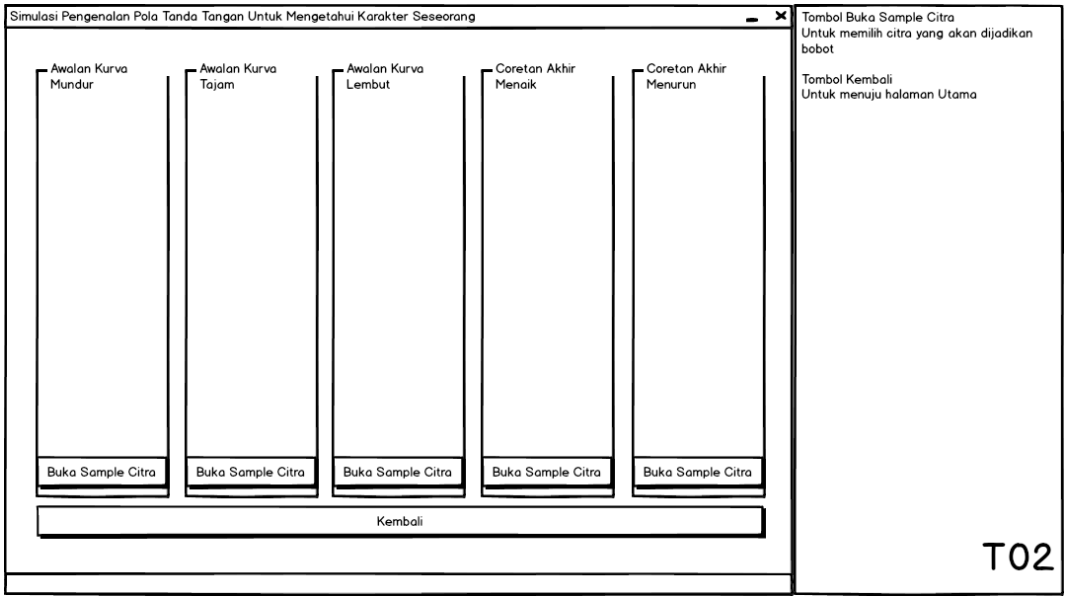
3.5.Perancangan Antarmuka

Perancangan antar muka menjelaskan tampilan kasar dari program yang akan dibangun. Ilustrasi perancangan antar muka dapat dilihat pada gambar 3.9 sampai gambar 3.13.

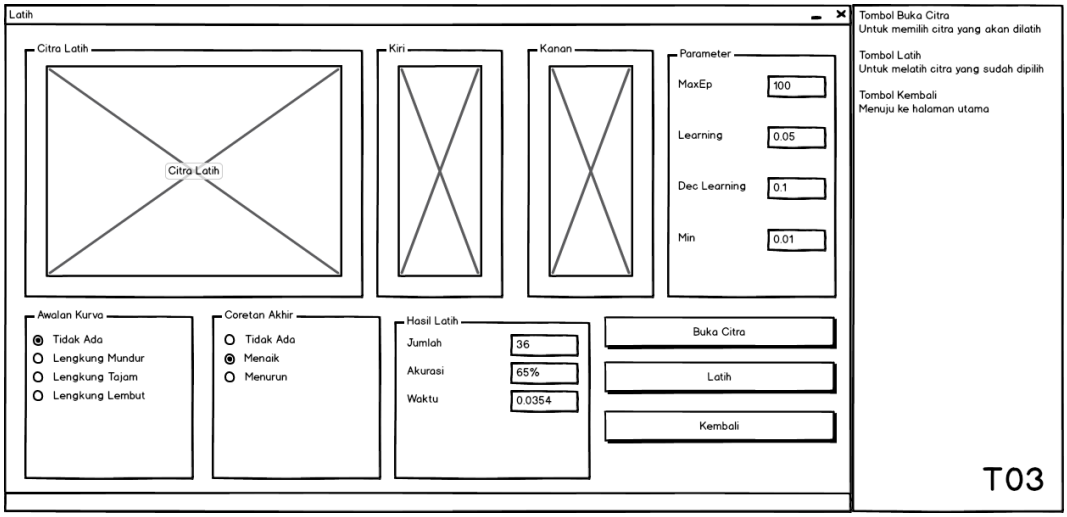
3.5.1. Perancangan Antarmuka Simulasi



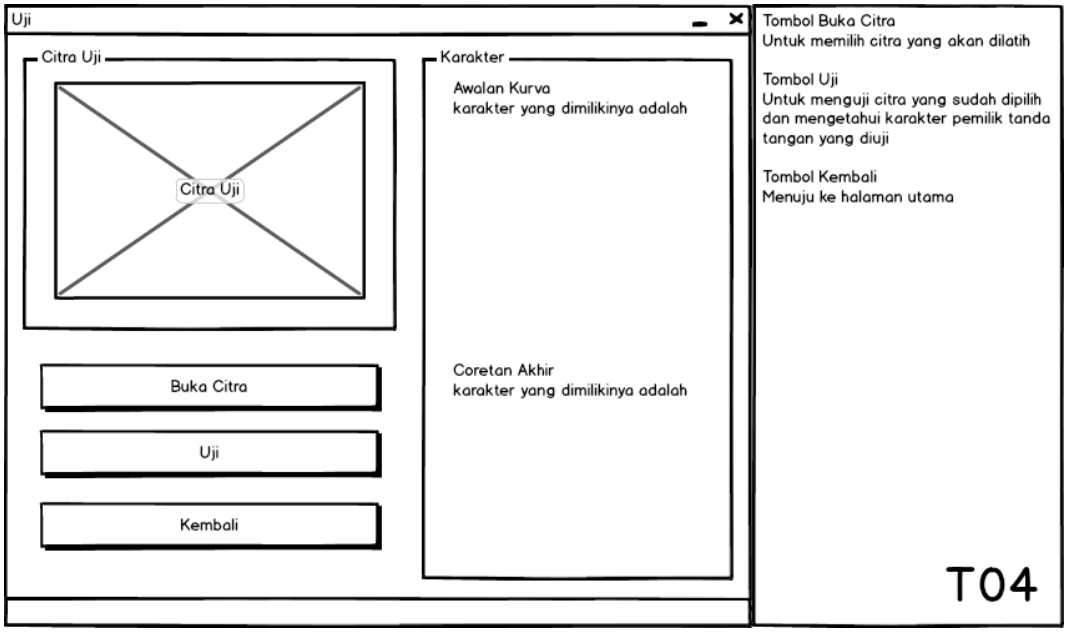
Gambar 3.12 Rancangan Antar Muka Halaman Utama



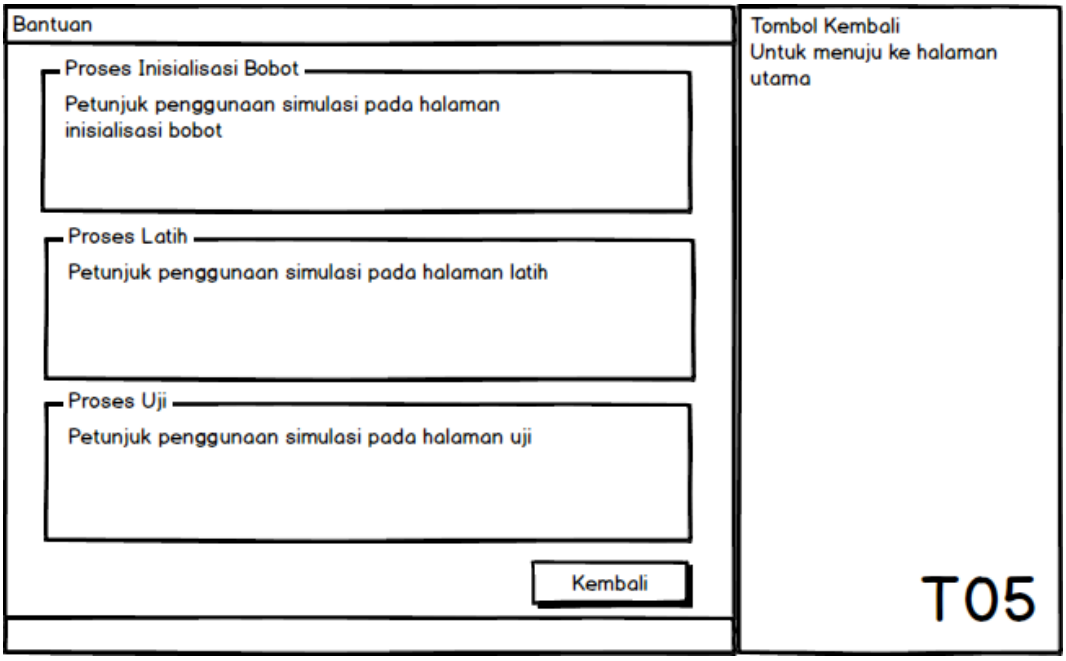
Gambar 3.13 Rancangan Antar Muka Inisialisasi Bobot



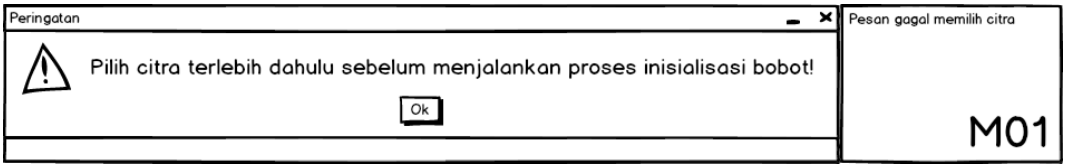
Gambar 3.14 Rancangan Antar Muka Halaman Latih



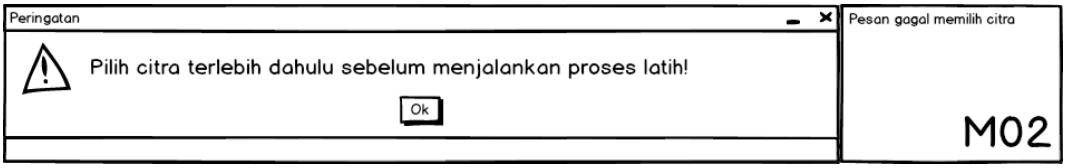
Gambar 3.15 Rancangan Antar Muka Halaman Uji



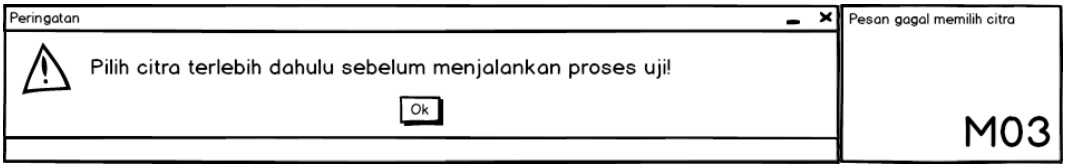
Gambar 3.16 Rancangan Antar Muka Halaman Bantuan



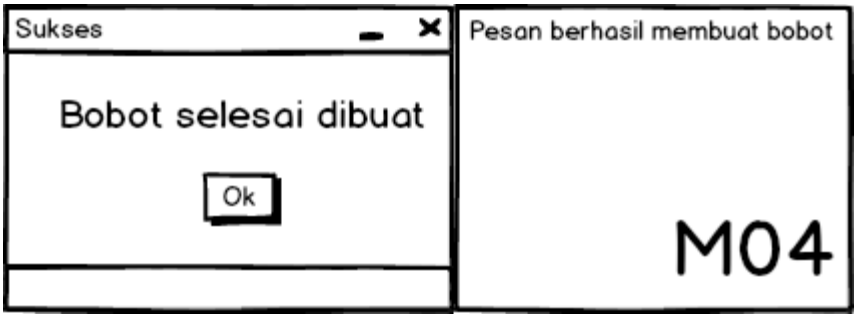
Gambar 3.17 Rancangan Antar Muka Pesan Gagal Pilih Citra Di Halaman Inisialisasi Bobot



Gambar 3.18 Rancangan Antar Muka Pesan Gagal Pilih Citra Di Halaman Latih



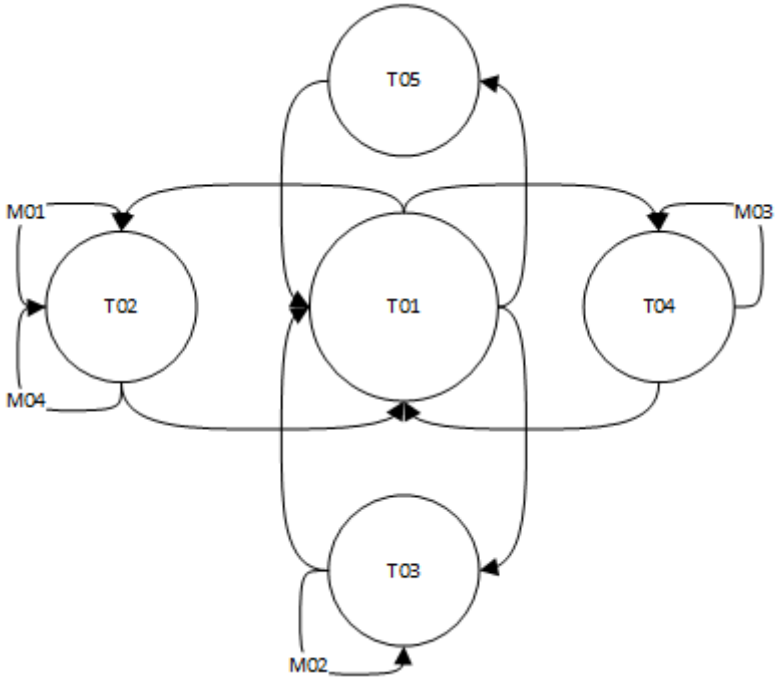
Gambar 3.19 Rancangan Antar Muka Pesan Gagal Pilih Citra Di Halaman Uji



Gambar 3.20 Rancangan Antar Muka Pesan Bobot Selesai Dibuat

3.5.2. Jaringan Semantik

Jaringan semantik merupakan alat efektif untuk merepresentasikan pemetaan data yang bertujuan mencegah terjadinya duplikasi data. Ilustrasinya dapat dilihat pada gambar 3.14.



Gambar 3.21 Jaringan Semantik

Keterangan

T01 : Antar muka halaman utama

T02 : Antar muka halaman inisialisasi bobot

T03 : Antar muka halaman latih

T04 : Antar muka halaman uji

T05 : Antar muka halaman bantuan

M01 : Pesan citra belum dipilih di halaman inisialisasi bobot

M02 : Pesan citra belum dipilih di halaman latih

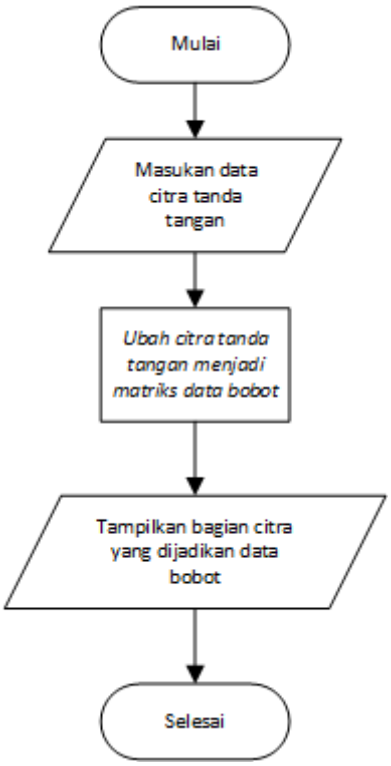
M03 : Pesan citra belum dipilih di halaman uji

M04 : Bobot selesai dibuat

3.5.3. Perancangan Prosedural

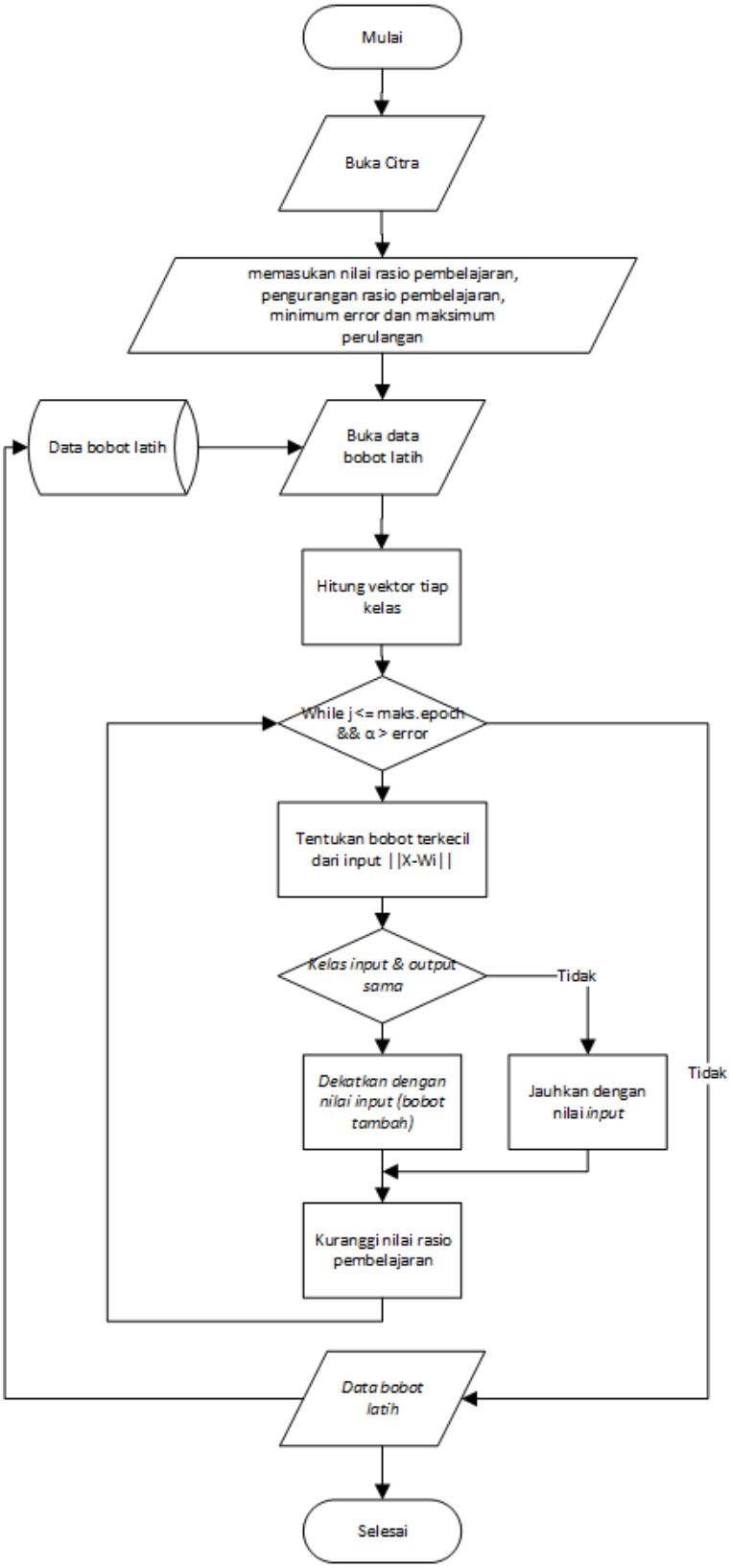
Perancangan prosedural digambarkan dalam bentuk *flowchart*. *Flowchart* merupakan suatu cara untuk menunjukan alir (*flow*) didalam program atau prosedur sistem secara logika. Ilustrasi *flowchart* prosedur-prosedur yang terdapat pada simulasi dapat dilihat pada gambar 3.19 sampai gambar 3.21.

1. Prosedur Inisialisasi Bobot



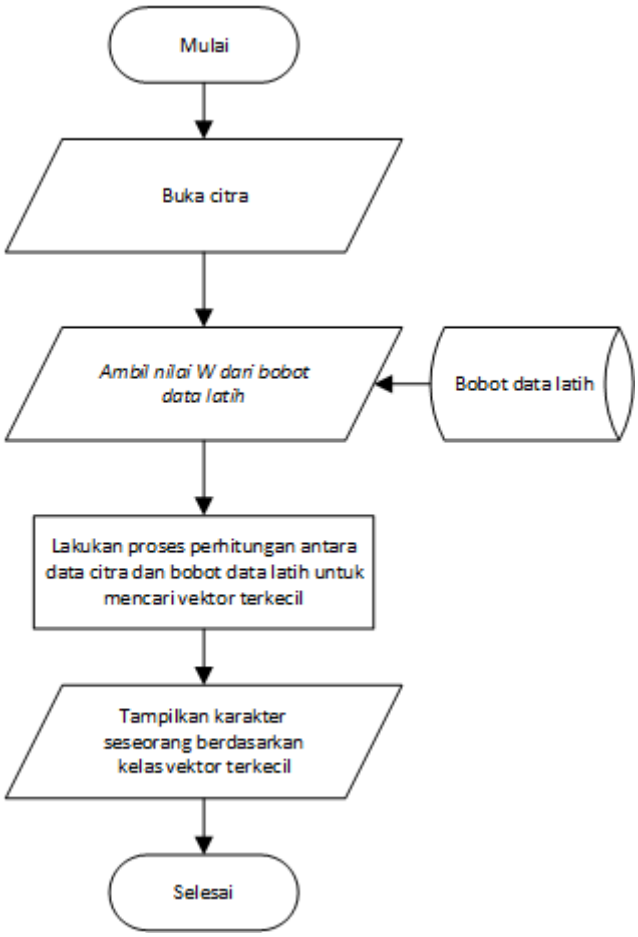
Gambar 3.22 Prosedur Inisialisasi Bobot

2. Prosedur Pelatihan LVQ



Gambar 3.23 Prosedur Pelatihan LVQ

3. Prosedur Pengujian LVQ



Gambar 3.24 Prosedur Pengujian LVQ