BAB 3
ANALISIS DAN PERANCANGAN SISTEM

3.1 Analisis Sistem

Analisis sistem merupakan suatu cara yang digunakan dalam mengidentifikasi dan mengevaluasi suatu permasalah, hambatan yang dihadapi dan kebutuhan yang diharapkan dalam sebuah sistem. Analisis ini diperlukan sebagai dasar untuk tahapan perancangan sistem. Hasil dari proses analisis sistem ini akan menghasilkan berbagai kesimpulan dan saran yang dapat digunakan untuk dijadikan dasar perancangan aplikasi yang akan dibangun.

3.1.1 Analisis Masalah

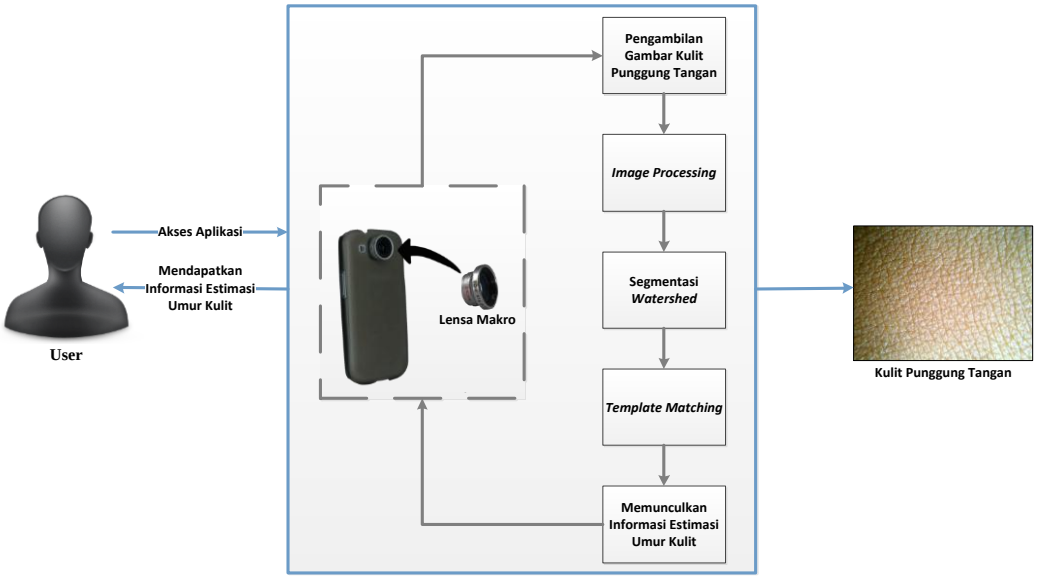
Analisis masalah yang dimaksud disini adalah setiap usia muda tidak menjamin kulit yang harus serta terlihat muda, terkadang ada juga orang berusia muda yang kulitnya telah berkerut. Proses penuaan dini pun sudah dapat terdektesi dari tangan yang tidak terawat, tanda-tanda penuaan dapat terlihat pada kulit punggung tangan seseorang. Sedangkan kita tidak menyadari penuaan yang terjadi pada kulit datang lebih dini.

Teknologi yang berkembang sekarang ini, khususnya teknologi *mobile* yang unggul dengan sisi ruang geraknya (*mobilitas*), *user friendly*, serta praktis sehingga dapat menutupi kekurangan yang terdapat pada media saat ini. Merunjuk terhadap kekurangan yang ada maka dibangun aplikasi *mobile* dengan menggunakan teknologi pengolahan citra digital (*Digital Image Processing*) dengan memanfaatkan fasilitas kamera pada *mobile.* Dengan demikian aplikasi ini bertujuan untuk mengenali pigmentasi kulit, dan kerutan-kerutan halus yang dapat menjadi acuan dalam mendeteksi penuaan dini pada kulit.

3.1.2 Analisis Sistem yang Akan Dibangun

Sistem yang akan dibangun adalah suatu aplikasi *mobile* menggunakan *platform* android dengan teknologi pengolahan citra digital (*Digital Image Processing*). Tujuan yang ingin dicapai dari perancangan aplikasi ini adalah dapat

mengenali pigmentasi kulit dan kerutan-kerutan halus yang dapat menjadi acuan dalam mendeteksi penuaan dini pada kulit. Adapun gambaran umum tentang sistem yang akan dibangun dapat dilihat pada Gambar 3.1.



Gambar 3.1 Proses Kerja Sistem yang Akan Dibangun

Gambar 3.1 menggambarkan bahwa data yang dipilih berupa gambar kulit punggung tangan manusia. Terdapat proses untuk mengenali pigmentasi kulit dan kerutan-kerutan pada kulit, yaitu *image processing*, *watershed* dan *template matching*. Hasil dari proses ditampilkan pada layar *mobile*.

3.1.3 Analisis Dataset

Penelitian ini menggunakan dataset gambar punggung tangan kulit manusia yang merupakan basisdata citra dengan format JPG dengan ukuran rata-rata per citra adalah 500x500. Contoh citra yang digunakan dalam peneitian ini dapat dilihat pada gambar 3.2.

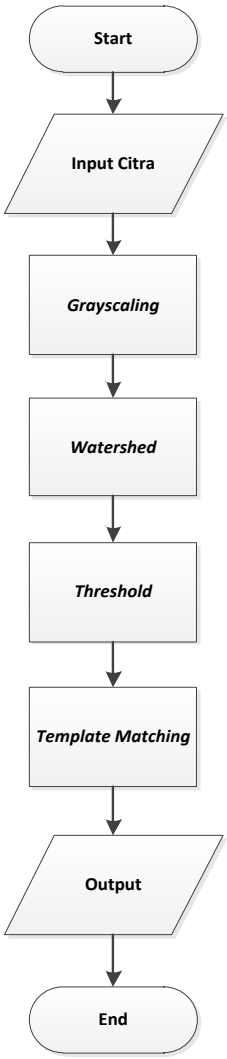


Gambar 3.2 Contoh Citra yang digunakan

Citra kulit punggung tangan pada gambar 3.2 adalah citra yang dipilih untuk mengklasifikasikan penuaan dini pada kulit dengan 21 kategori berbeda, citra kulit punggung tangan yang dijadikan dataset diambil berdasarkan usia dari 20 sampai 40 Tahun. Karena tanda penuaan kulit untuk setiap tahapan usia tidaklah sama, memasukin rentang usia 25-35 tahun merupakan awal penuaan, munculnya garis-garis halus dan keriput.

3.2 Analisis Metode

Analisis metode digunakan untuk mengetahui alur proses dari sebuah metode yang digunakan dapat diterapkan ke dalam aplikasi yang dibangun. Pembangunan aplikasi ini menggunakan metode *watershed* yang digunakan untuk proses segmentasi citra dan *template matching* untuk mencocokan tiap-tiap bagian dari suatu citra. Adapun penerapan metode pada alur kerja sistem dapat dilihat pada gambar 3.3.



Gambar 3.3 Penerapan Metode Pada Alur Kerja Sistem

Berikut penerapan metode pada alur kerja sistem yang akan dibangun adalah:

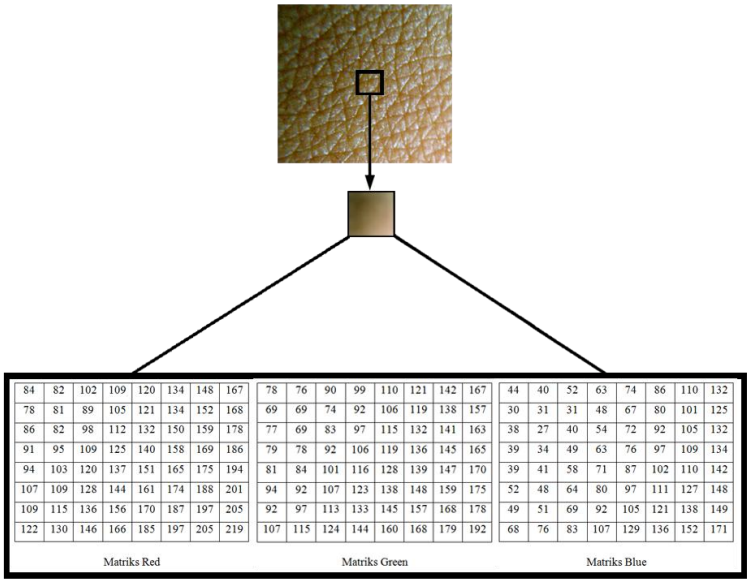
1. Input Citra, Pengguna mengambil gambar atau memilih citra/gambar kulit manusia yang akan diproses oleh sistem.
2. *Grayscale,* Pada tahap ini mengubah citra warna (RGB) untuk memperoleh informasi intensitas dari gambar. Sehingga dapat disortir mulai dari hitam untuk intensitas yang paling lemah sampai dengan putih yang intensitas yang paling kuat.

3. *Watershed*, pada tahap ini dilakukan untuk memisahkan satu objek dengan objek lain. Pada tahap ini metode watershed membagi skala keabuan dalam *region* yang berbeda dengan merepresentasikan citra sebagai *relief* topografi.
4. *Threshold* berfungsi untuk mengubah citra berderajat keabuan menjadi citra biner atau hitam putih sehingga dapat diketahui daerah mana yang termasuk obyek dan *background* dari citra secara jelas, citra hasil *threshold* ini digunakan lebih lanjut untuk proses pengenalan obyek.
5. Proses *template matching*, dalam tahapan ini berfungsi untuk mencocokkan tiap-tiap bagian dari citra uji dengan citra yang menjadi acuan (*template*) yang sudah tersimpan didataset.
6. Output, yaitu proses menampilkan usia kulit untuk menentukan penuaan dini pada kulit.

Untuk lebih jelas mengenai proses-proses yang dilakukan pada sistem ini, penjelasannya sebagai berikut:

3.2.1 Analisis Tahap Grayscaling

Tahap ini dilakukan untuk mengubah citra warna menjadi citra *greyscale*, hal ini digunakan untuk menyederhanakan model citra. Citra warna terdiri dari 3 layer matrik yaitu R-layer, G-layer, B-layer. Sebagai contoh, misalkan citra berukuran 8x8 *pixel* dengan nilai-nilai R, G, dan B sebagai berikut:



Gambar 3.4 Matrik (R, G dan B)

Jadi dalam proses ini akan mengubah 3 layer matrik citra berwarna menjadi 1 layer matrik *grayscale* dan akan menghasilkan citra *grayscale*. Dalam citra ini tidak ada lagi warna, yang ada adalah derajat keabuan. Pada dasarnya proses *grayscaling* dilakukan dengan meratakan nilai *pixel* dari 3 nilai R, G, dan B dengan menggunakan persamaan berikut:

Grayscale = 0.299*R + 0.587*G + 0.114*B .....(3-1)

Contoh untuk menentukan nilai *grayscale* diilustrasikan dengan nilai-nilai R, G, dan B pada gambar 3.3, untuk konversi warna menjadi *grayscale* digunakan persamaan 3-1. Misalkan untuk *pixel* di koordinat (0,0) perhitungannya sebagai berikut:

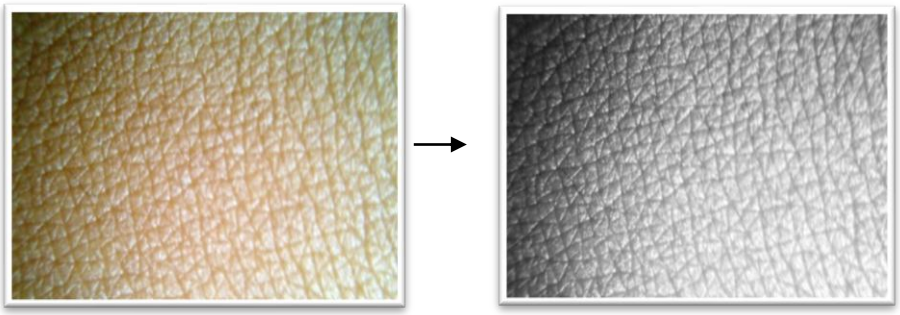
Grayscale = (0,299*84) + (0,587*78) + (0,114*44)
= 25,12 + 45,79 + 5,02
= 75,93

Didapatkan hasil perhitungan untuk koordinat (0,0) = 75,93. Setelah perhitungan tersebut dilakukan untuk semua koordinat, maka dari hasil perhitungan *grayscale* dilakukan *rounding* (pembulatan) nilai ke paling dekat. Jadi hasil pembulatan dari koordinat untuk (0,0) = 76. Sehingga didapatkan citra *grayscale* sebagai berikut:

(i,j)	0	1	2	3	4	5	6	7
0	76	74	93	97	108	120	140	163
1	73	72	82	68	105	118	136	155
2	75	69	67	96	114	132	141	162
3	78	78	92	106	119	136	146	165
4	80	84	101	116	129	140	149	171
5	93	89	107	123	140	149	171	177
6	92	96	114	134	146	156	170	180
7	106	114	124	144	161	170	181	194

Gambar 3.5 Matriks Grayscale

Citra yang berwarna akan digantikan dengan citra *grayscale* sesuai dengan nilai yang sudah didapatkan melalui perhitungan di atas. Berikut adalah citra *grayscale* yang dihasilkan.



Gambar 3.6 Hasil Konversi Citra RGB Menjadi Citra *Grayscale*

3.2.2 Analisis Tahap Watershed

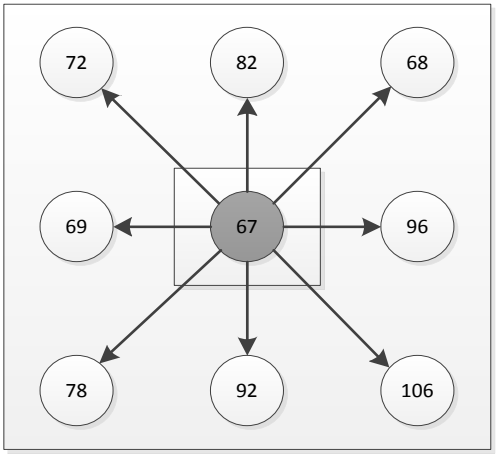
Setelah nilai citra *greyscale* diproleh, dilakukan proses untuk mencari garis *watershed* yang mempresentasikan objek yang akan disegmentasikan. Langkah pertama dalam mengoprasikan *watershed* untuk sebuah citra adalah mengindentifikasi bagian intensitas minimum. Titik dari nilai minimum ini merupakan dasar dari *watershed*. Proses *grayscale* digunakan untuk menghasilkan nilai citra yang rata, untuk mengeliminasi dataran tinggi pada citra dan menyederhanakan proses indentifikasi maksimum dan minimum. Untuk membedakan titik maksimum dan minimum, setiap titik akan dibandingkan

dengan delapan titik tetangga terdekatnya. Jika delapan titik tetangganya lebih besar daripada titik tengah, maka titik diindentifikasi sebagai intensitas minimum. Sebaliknya, jika delapan titik tetangganya memiliki intensitas yang lebih kecil dari titik tengah maka titik tersebut memiliki intensitas maksimum.

Contoh untuk menentukan titik minimum pada nilai citra diilustrasikan pada matrik *grayscale* yang ada dibawah ini, dimana pusat *pixel* dan setiap titik akan dibandingkan dengan delapan titik tetangga terdekatnya.

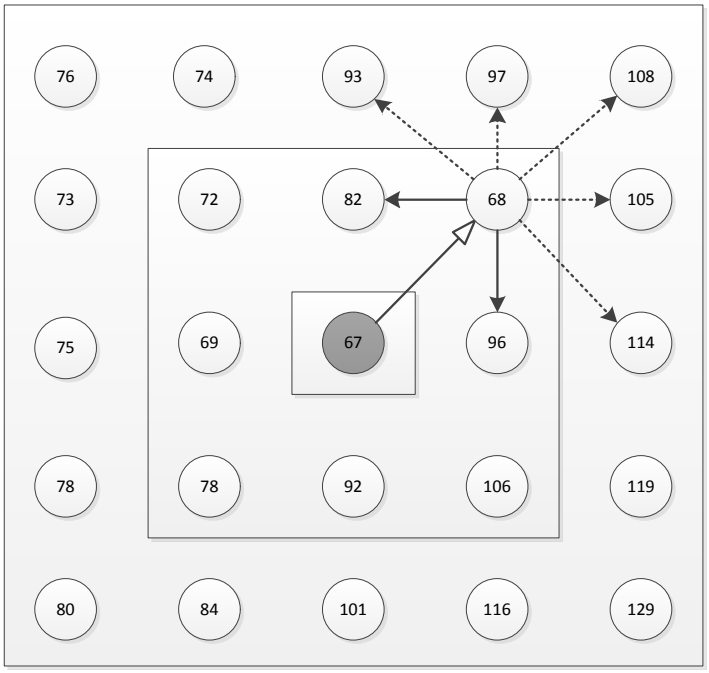
(i,j)	0	1	2	3	4	5	6	7
0	76	74	93	97	108	120	140	163
1	73	**72**	**82**	**68**	105	118	136	155
2	75	**69**	**67**	**96**	114	132	141	162
3	78	**78**	**92**	**106**	119	136	146	165
4	80	84	101	116	129	140	149	171
5	93	89	107	123	140	149	171	177
6	92	96	114	134	146	156	170	180
7	106	114	124	144	161	170	181	194

Pada langkah pertama mengindentifikasi nilai minimum dari matrik *greyscale* diatas, setelah itu dilakukan perbadingan nilai dengan delapan tetangganya dengan menggunakan struktur elemen 3x3. Dimana pusat *pixel* adalah nilai 67, karena nilai 67 adalah nilai minimum dari semua nilai yang lain.



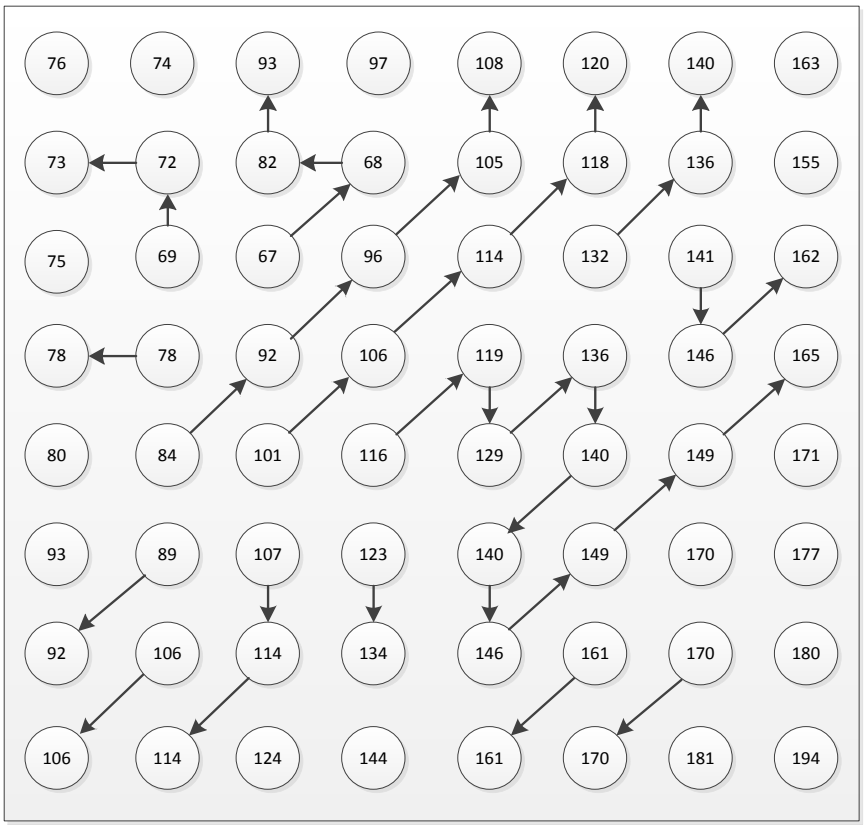
Langkah selanjutnya pada pusat *pixel* memutuskan mencari nilai minimum dari delapan tetangganya, misalnya untuk mencari nilai minimum dari delapan

tetangganya yang terhubung akan dibandingkan dengan pusat *pixel*, jika kondisi nilai pusat *pixel* lebih rendah dari nilai citra yang lain, maka melakukan pencarian nilai citra yang hampir mendekati nilai pusat *pixel*, untuk menentukan nilai minimum dan garis watershed.



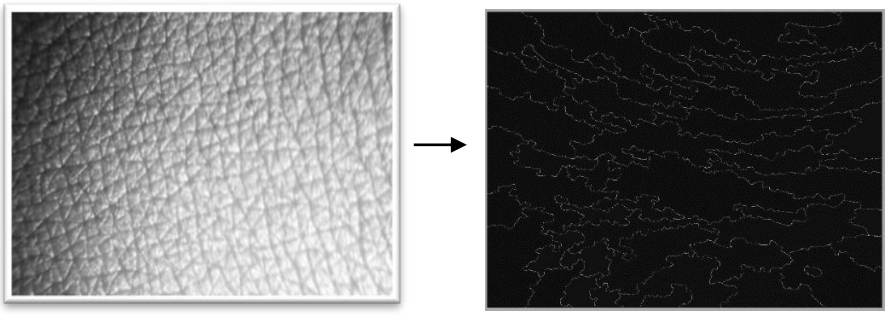
Setelah ditemukan nilai minimunnya, maka akan terjadi garis dari nilai 67 ke nilai 68. Selanjutnya dilakukan lagi perbandingan dengan delapan tetangganya dengan struktur 3x3 dimana sekarang titik pusat *pixel* adalah nilai 68, dan ulangi langkah perbandingan dengan delapan tetangganya sehingga seluruh *pixel* terhubung. Konsep yang terjadi pada watershed ini yaitu:

1. Tentukan daerah (*region*) R1, R2,.., Rn, dengan merepresentasikan objek yang memiliki nilai minimum.
2. Dilakukan perbadingan titik pusat *pixel* dengan delapan tetangganya dengan menggunakan struktur elemen 3x3.
3. Bentuklah dam (garis) pada posisi dimana dua daerah terhubung.
4. Ulangi langkah tiga sehingga semua daerah terhubung, dan didapatkan hasilnya seperti gambar dibawah ini.



Gambar 3.7 Hasil Pembentukan dam (garis) Watershed

Proses segmentasi *watershed* dilakukan untuk mengenali pigmentasi kulit dan kerutan-kerutan halus pada kulit, maka diperoleh citra hasil segmentasi *watershed* seperti terlihat pada gambar dibawah ini.



Gambar 3.8 Hasil Konversi Citra *Grayscale* Menjadi Citra Hasil Segmentasi *Watershed*

3.2.3 Analisis Tahap Threshold

Tahap ini berfungsi untuk mengubah citra menjadi citra biner yaitu 0 dan 1, dan bertujuan untuk memisahkan dan membedakan antara objek dan background. Proses *threshold* atau binarisasi pada prinsipnya adalah melakukan pengubahan nilai intensitas warna piksel menjadi 2 nilai yaitu 0 atau warna hitam dan 1 atau warna putih, berarti nilai pixel dikelompkan menjadi dua kelompok seperti ditunjukan pada rumus berikut:

$$f_0(x,y) = \begin{cases} 0, & f_1(x,y) < 128 \\ 1, & f_1(x,y) \geq 128 \end{cases} \quad \text{.....(3-2)}$$

Menentukan hasil threshold dilakukan pembandingan, jika nilai citra lebih dari 128 maka akan diubah menjadi warna putih dengan nilai 1. Jika nilai kurang dari 128 maka akan diubah menjadi warna hitam dengan nilai 0. Citra hasil *threshold* ini digunakan lebih lanjut untuk proses pengenalan.

3.2.4 Template Matching

Setelah proses *threshold* maka sistem sudah mendapatkan deretan angka biner dari citra sumber, langkah selanjutnya adalah melakukan proses pengenalan pola dengan setiap deretan angka biner dari citra target yang sudah dipersiapkan didalam *dataset*.

Pengenalan pola dengan menggunakan metode *template matching* dilakukan dengan cara membandingkan citra masukan dengan citra *template*. Citra masukan dihitung berdasarkan banyaknya titik yang sesuai dengan citra *template*.

Pixel citra biner ditelusuri mulai dari kiri atas hingga ke kanan bawah. Citra biner dengan *pixel* berwarna hitam akan direpsentasikan dengan nilai 0. Sedangkan *pixel* citra yang berwarna putih akan direpsentasikan dengan nilai 1. Deretan angka biner pada citra masukan akan dihitung dengan deretan angka biner pada citra *template*. *Template* dengan nilai eror terkecil merupakan *template* citra yang paling sesuai dengan citra masukan.

Contoh berikut ini adalah penerapan pengenalan pigmentasi kulit, dan kerutan-kerutan halus yang dapat menjadi acuan dalam penentuan penuaan dini

pada kulit dengan menggunakan metode *template matching* yang menggunakan persamaan berikut:

$$\min e = \sum_{(x,y)\epsilon W} (I_{x,y} - T_{x,y})^2 \text{ .....(3-2)}$$

I adalah pola pixel citra masukan yang akan dibandingkan. T adala pola pixel citra template. Tabel 3.1 adalah tabel contoh deretan angka biner citra *template* yang sudah tersimpan didataset.

Tabel 3.1 Contoh Deretan Angkat Biner Citra *Template* 8x8 *pixel*

Dataset	Deretan Angka Biner Citra *Template*
Citra Kulit 20 Tahun.jpg	0 0 0 1 0 0 0 0 0 0 0 0 1 1 0 1 0 0 0 0 0 0 1 0 0 1 0 0 0 0 0 0 1 1 1 0 1 0 0 1 1 0 1 1 0 1 1 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0
Citra Kulit 21 Tahun.jpg	0 1 0 0 0 0 0 0 0 1 1 0 0 0 0 0 0 0 1 1 0 0 0 0 0 0 0 1 1 0 1 0 0 0 1 0 0 1 0 1 0 0 1 1 1 0 1 0 0 1 0 1 0 0 0 0 0 0 1 0 1 0 0 0
Citra Kulit 22 Tahun.jpg	0 0 0 0 0 1 0 0 1 1 0 0 0 1 0 0 0 0 1 0 0 0 1 0 1 0 1 0 0 1 0 0 0 1 0 1 0 0 1 1 0 1 1 1 0 0 1 0 0 1 0 0 0 1 0 1 0 0 0 0 0 0 1 0
Citra Kulit 23 Tahun.jpg	0 0 0 0 0 1 0 0 0 0 0 0 1 1 0 0 0 0 0 0 0 1 0 0 0 0 0 0 1 1 0 0 1 0 0 0 1 0 0 1 0 1 1 0 0 1 1 0 0 0 1 0 1 0 1 0 0 0 0 1 1 0 1 0

Citra Kulit 24 Tahun.jpg	0 0 1 0 1 0 1 0 0 1 0 0 1 1 0 0 0 1 0 1 0 1 0 0 0 1 0 1 1 0 0 0 1 0 1 0 1 0 0 0 0 1 0 1 0 0 0 1 0 0 1 0 0 0 1 0 0 0 1 0 0 1 0 0
Citra Kulit 25 Tahun.jpg	0 0 0 0 0 1 0 0 0 1 0 1 1 0 0 0 1 0 1 0 1 0 0 0 1 0 1 0 0 0 0 0 0 0 0 0 0 0 1 1 0 1 1 0 1 1 0 0 0 0 1 0 1 1 0 0 0 0 0 0 1 0 0 0
Citra Kulit 26 Tahun.jpg	0 1 0 1 0 0 0 0 1 0 0 0 1 0 0 0 0 0 0 1 0 0 0 1 0 0 1 0 0 1 0 1 0 0 0 1 1 0 1 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 1 0 0 0
Citra Kulit 27 Tahun.jpg	0 0 1 0 1 0 0 1 0 0 1 0 0 1 0 1 0 0 1 0 1 0 0 1 0 0 1 0 1 0 0 0 0 1 0 0 1 0 0 0 0 1 0 0 0 1 0 0 0 1 0 0 0 1 0 0 1 0 0 0 1 0 0 0
Citra Kulit 28 Tahun.jpg	1 0 1 0 0 0 0 1 0 1 1 0 0 1 0 1 1 0 1 0 1 0 0 1 0 0 1 0 1 0 0 0 0 1 0 0 1 0 0 0 0 1 0 0 0 1 0 0 0 1 0 0 0 1 0 0 1 0 0 0 1 0 0 0

Citra Kulit 29 Tahun.jpg	0 0 0 0 1 0 0 1 0 0 1 0 0 1 1 1 0 1 1 0 1 0 0 1 1 0 1 0 1 0 0 0 0 1 0 0 1 0 0 0 0 1 0 0 0 1 0 0 0 1 1 0 0 1 0 0 1 0 0 1 1 0 0 0
Citra Kulit 30 Tahun.jpg	0 1 0 0 0 1 0 1 1 0 1 0 0 1 0 0 0 0 0 1 0 0 1 0 0 0 0 0 1 0 0 1 0 1 0 0 1 0 0 1 0 1 0 0 0 0 1 0 1 0 0 0 0 1 0 0 0 1 0 0 1 0 0 0
Citra Kulit 31 Tahun.jpg	0 0 0 1 0 1 0 1 1 0 1 0 1 0 1 0 0 1 0 1 0 0 1 0 0 0 1 0 0 0 0 1 0 1 0 0 1 0 0 1 0 1 0 1 0 0 1 0 1 0 1 0 0 1 0 0 0 1 0 0 1 0 0 0
Citra Kulit 32 Tahun.jpg	1 0 1 0 1 0 0 1 0 1 0 0 0 1 0 1 1 0 1 0 1 0 1 1 0 0 0 0 0 1 0 0 0 1 0 0 1 0 0 0 1 0 0 1 0 1 0 0 0 1 1 0 1 0 0 0 1 0 0 1 0 0 0 0
Citra Kulit 33 Tahun.jpg	0 0 1 0 1 0 0 1 1 0 0 1 0 1 0 1 0 1 1 0 0 0 1 1 1 0 0 0 0 1 0 0 0 1 0 0 1 0 0 0 1 0 0 1 0 1 0 0 0 1 0 0 1 0 1 0 1 0 0 1 0 0 0 1

Citra Kulit 34 Tahun.jpg	0 0 0 0 1 0 0 1 1 0 1 0 0 1 0 0 0 1 0 0 1 0 1 0 0 0 1 0 1 0 0 1 0 1 0 1 0 0 1 0 0 0 1 0 0 1 0 0 0 0 0 0 0 1 0 0 1 0 0 0 1 0 0 0
Citra Kulit 35 Tahun.jpg	0 0 0 0 0 1 0 0 0 0 0 1 1 0 0 0 1 0 1 0 1 0 0 0 0 1 0 1 0 1 0 0 0 0 1 0 1 0 1 1 0 1 0 0 0 1 0 0 1 0 1 0 1 0 0 0 0 1 0 0 1 0 0 0
Citra Kulit 36 Tahun.jpg	1 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 0 1 0 0 0 1 0 0 1 0 0 0 1 0 0 1 0 1 0 0 1 1 0 0 1 0 0 1 0 1 0 1 0 0 1 0 1 0 1 0 0 0
Citra Kulit 37 Tahun.jpg	0 0 1 0 0 0 1 0 0 1 0 1 0 1 0 0 0 0 1 0 1 0 1 0 1 0 1 0 0 1 0 0 0 1 0 1 0 0 0 1 1 0 1 0 0 0 1 0 0 1 0 0 1 0 0 1 1 0 0 0 0 1 0 0
Citra Kulit 38 Tahun.jpg	0 0 1 0 0 1 0 0 0 1 0 0 1 0 0 0 1 0 1 0 0 0 0 0 0 1 0 1 0 1 0 0 1 0 0 0 1 0 1 1 0 1 0 0 0 1 1 0 0 0 1 0 1 0 1 0 0 0 0 1 1 0 1 0

Citra Kulit 39 Tahun.jpg	0 1 0 0 1 0 0 1 1 0 1 0 0 1 0 1 0 1 0 1 0 0 1 0 1 0 1 0 1 0 0 1 0 1 0 1 0 0 1 0 0 0 1 0 0 1 0 0 0 0 0 1 0 0 1 0 0 0 0 0 1 0 0 1
Citra Kulit 40 Tahun.jpg	1 0 0 1 0 1 0 0 0 0 1 0 1 0 0 1 0 1 1 0 1 0 1 0 0 1 0 1 0 1 0 0 1 0 1 0 1 0 1 0 0 1 0 1 0 1 0 0 1 0 1 0 1 0 0 0 0 1 0 0 1 0 0 0

Diilustrasikan deretan angka biner berikut ini adalah citra masukan kulit 8x8 *pixel*

0 0 0 1 0 1 0 1 0 1 0 0 1 1 0 0 1 0 1 1 1 0 0 0 0 1 0 1 1 0 1 0

0 0 0 0 1 0 0 0 0 0 1 1 1 1 0 1 0 1 0 1 0 1 0 0 1 0 0 0 0 1 1 1

Langkah selanjutnya melakukan pencocokan citra masukan dan citra *template.* Tabel 3.2 adalah perhitungan nilai eror dengan menggunakan persamaan 3-2, yaitu untuk menentukan kecocokan pada citra masukan, contoh perhitungannya seperti dibawah ini:

Tabel 3.2 Perhitungan Nilai Eror

Dataset	Perhitungan Nilai Eror
Citra Kulit 20 Tahun.jpg	$(0-0)^2 + (0-0)^2 + (0-0)^2 + (1-1)^2 + (0-0)^2 + (1-0)^2 + (0-0)^2 + (1-0)^2 +$ $(0-0)^2 + (1-0)^2 + (0-0)^2 + (0-0)^2 + (1-1)^2 + (1-1)^2 + (0-0)^2 + (0-1)^2 +$ $(1-0)^2 + (0-0)^2 + (1-0)^2 + (1-0)^2 + (1-1)^2 + (0-0)^2 + (0-1)^2 + (0-1)^2 +$ $(0-1)^2 + (1-1)^2 + (0-0)^2 + (1-0)^2 + (1-0)^2 + (0-0)^2 + (1-0)^2 + (0-0)^2 +$ $(0-1)^2 + (0-1)^2 + (0-1)^2 + (0-0)^2 + (1-1)^2 + (0-0)^2 + (0-0)^2 + (0-1)^2 +$ $(1-0)^2 + (0-0)^2 + (1-1)^2 + (1-1)^2 + (1-0)^2 + (1-1)^2 + (0-1)^2 + (1-0)^2 +$ $(0-0)^2 + (1-1)^2 + (0-0)^2 + (1-0)^2 + (0-0)^2 + (1-0)^2 + (0-0)^2 + (0-0)^2 +$ $(1-0)^2 + (0-0)^2 + (0-0)^2 + (0-0)^2 + (0-0)^2 + (1-0)^2 + (1-0)^2 + (1-0)^2$ $= 27$
Citra Kulit 21 Tahun.jpg	$(0-0)^2 + (0-1)^2 + (0-0)^2 + (1-0)^2 + (0-0)^2 + (1-0)^2 + (0-0)^2 + (1-0)^2 +$ $(0-0)^2 + (1-1)^2 + (0-1)^2 + (0-0)^2 + (1-0)^2 + (1-0)^2 + (0-0)^2 + (0-0)^2 +$ $(1-0)^2 + (0-0)^2 + (1-1)^2 + (1-1)^2 + (1-0)^2 + (0-0)^2 + (0-0)^2 + (0-0)^2 +$ $(0-0)^2 + (1-0)^2 + (0-0)^2 + (1-1)^2 + (1-1)^2 + (0-0)^2 + (1-1)^2 + (0-0)^2 +$ $(0-0)^2 + (0-0)^2 + (0-1)^2 + (0-0)^2 + (1-0)^2 + (0-1)^2 + (0-0)^2 + (0-1)^2 +$ $(0-0)^2 + (0-0)^2 + (1-1)^2 + (1-1)^2 + (1-1)^2 + (1-0)^2 + (0-1)^2 + (1-0)^2 +$

	$(0-0)^2 + (1-1)^2 + (0-0)^2 + (1-1)^2 + (0-0)^2 + (1-0)^2 + (0-0)^2 + (0-0)^2 +$ $(1-0)^2 + (0-0)^2 + (0-1)^2 + (0-0)^2 + (0-1)^2 + (1-0)^2 + (1-0)^2 + (1-0)^2$ $= 24$
Citra Kulit 22 Tahun.jpg	$(0-0)^2 + (0-0)^2 + (0-0)^2 + (1-0)^2 + (0-0)^2 + (1-1)^2 + (0-0)^2 + (1-0)^2 +$ $(0-1)^2 + (1-1)^2 + (0-0)^2 + (0-0)^2 + (1-0)^2 + (1-1)^2 + (0-0)^2 + (0-0)^2 +$ $(1-0)^2 + (0-0)^2 + (1-1)^2 + (1-0)^2 + (1-0)^2 + (0-0)^2 + (0-1)^2 + (0-0)^2 +$ $(0-1)^2 + (1-0)^2 + (0-1)^2 + (1-0)^2 + (1-0)^2 + (0-1)^2 + (1-0)^2 + (0-0)^2 +$ $(0-0)^2 + (0-1)^2 + (0-0)^2 + (0-1)^2 + (1-0)^2 + (0-0)^2 + (0-1)^2 + (0-1)^2 +$ $(0-0)^2 + (0-1)^2 + (1-1)^2 + (1-1)^2 + (1-0)^2 + (1-0)^2 + (0-1)^2 + (1-0)^2 +$ $(0-0)^2 + (1-1)^2 + (0-0)^2 + (1-0)^2 + (0-0)^2 + (1-1)^2 + (0-0)^2 + (0-1)^2 +$ $(1-0)^2 + (0-0)^2 + (0-0)^2 + (0-0)^2 + (0-0)^2 + (1-0)^2 + (1-1)^2 + (1-0)^2$ $= 30$
Citra Kulit 23 Tahun.jpg	$(0-0)^2 + (0-0)^2 + (0-0)^2 + (1-0)^2 + (0-0)^2 + (1-1)^2 + (0-0)^2 + (1-0)^2 +$ $(0-0)^2 + (1-0)^2 + (0-0)^2 + (0-0)^2 + (1-1)^2 + (1-1)^2 + (0-0)^2 + (0-0)^2 +$ $(1-0)^2 + (0-0)^2 + (1-0)^2 + (1-0)^2 + (1-0)^2 + (0-1)^2 + (0-0)^2 + (0-0)^2 +$ $(0-0)^2 + (1-0)^2 + (0-0)^2 + (1-0)^2 + (1-1)^2 + (0-1)^2 + (1-0)^2 + (0-0)^2 +$ $(0-1)^2 + (0-0)^2 + (0-0)^2 + (0-0)^2 + (1-1)^2 + (0-0)^2 + (0-0)^2 + (0-1)^2 +$ $(0-0)^2 + (0-1)^2 + (1-1)^2 + (1-0)^2 + (1-0)^2 + (1-1)^2 + (0-1)^2 + (1-0)^2 +$ $(0-0)^2 + (1-0)^2 + (0-1)^2 + (1-0)^2 + (0-1)^2 + (1-0)^2 + (0-1)^2 + (0-0)^2 +$ $(1-0)^2 + (0-0)^2 + (0-0)^2 + (0-1)^2 + (0-1)^2 + (1-0)^2 + (1-1)^2 + (1-0)^2$ $= 30$
Citra Kulit 24 Tahun.jpg	$(0-0)^2 + (0-0)^2 + (0-1)^2 + (1-0)^2 + (0-1)^2 + (1-0)^2 + (0-1)^2 + (1-0)^2 +$ $(0-0)^2 + (1-1)^2 + (0-0)^2 + (0-0)^2 + (1-1)^2 + (1-1)^2 + (0-0)^2 + (0-0)^2 +$ $(1-0)^2 + (0-1)^2 + (1-0)^2 + (1-1)^2 + (1-0)^2 + (0-1)^2 + (0-0)^2 + (0-0)^2 +$ $(0-0)^2 + (1-1)^2 + (0-0)^2 + (1-1)^2 + (1-1)^2 + (0-0)^2 + (1-0)^2 + (0-0)^2 +$ $(0-1)^2 + (0-0)^2 + (0-1)^2 + (0-0)^2 + (1-1)^2 + (0-0)^2 + (0-0)^2 + (0-0)^2 +$ $(0-0)^2 + (0-1)^2 + (1-0)^2 + (1-1)^2 + (1-0)^2 + (1-0)^2 + (0-0)^2 + (1-1)^2 +$ $(0-0)^2 + (1-0)^2 + (0-1)^2 + (1-0)^2 + (0-0)^2 + (1-0)^2 + (0-1)^2 + (0-0)^2 +$ $(1-0)^2 + (0-0)^2 + (0-1)^2 + (0-0)^2 + (0-0)^2 + (1-1)^2 + (1-0)^2 + (1-0)^2$ $= 28$
Citra Kulit 25 Tahun.jpg	$(0-0)^2 + (0-0)^2 + (0-0)^2 + (1-0)^2 + (0-0)^2 + (1-1)^2 + (0-0)^2 + (1-0)^2 +$ $(0-0)^2 + (1-1)^2 + (0-0)^2 + (0-1)^2 + (1-1)^2 + (1-0)^2 + (0-0)^2 + (0-0)^2 +$ $(1-1)^2 + (0-0)^2 + (1-1)^2 + (1-0)^2 + (1-1)^2 + (0-0)^2 + (0-0)^2 + (0-0)^2 +$ $(0-1)^2 + (1-0)^2 + (0-1)^2 + (1-0)^2 + (1-0)^2 + (0-0)^2 + (1-0)^2 + (0-0)^2 +$ $(0-0)^2 + (0-0)^2 + (0-0)^2 + (0-0)^2 + (1-0)^2 + (0-0)^2 + (0-1)^2 + (0-1)^2 +$ $(0-0)^2 + (0-1)^2 + (1-1)^2 + (1-0)^2 + (1-1)^2 + (1-1)^2 + (0-0)^2 + (1-0)^2 +$ $(0-0)^2 + (1-0)^2 + (0-1)^2 + (1-0)^2 + (0-1)^2 + (1-1)^2 + (0-0)^2 + (0-0)^2 +$ $(1-0)^2 + (0-0)^2 + (0-0)^2 + (0-0)^2 + (0-1)^2 + (1-0)^2 + (1-0)^2 + (1-0)^2$ $= 25$
Citra Kulit 26 Tahun.jpg	$(0-0)^2 + (0-1)^2 + (0-0)^2 + (1-1)^2 + (0-0)^2 + (1-0)^2 + (0-0)^2 + (1-0)^2 +$ $(0-1)^2 + (1-0)^2 + (0-0)^2 + (0-0)^2 + (1-1)^2 + (1-0)^2 + (0-0)^2 + (0-0)^2 +$ $(1-0)^2 + (0-0)^2 + (1-0)^2 + (1-1)^2 + (1-0)^2 + (0-0)^2 + (0-0)^2 + (0-1)^2 +$ $(0-0)^2 + (1-0)^2 + (0-1)^2 + (1-0)^2 + (1-0)^2 + (0-1)^2 + (1-0)^2 + (0-1)^2 +$ $(0-0)^2 + (0-0)^2 + (0-0)^2 + (0-1)^2 + (1-1)^2 + (0-0)^2 + (0-1)^2 + (0-0)^2 +$ $(0-0)^2 + (0-0)^2 + (1-0)^2 + (1-0)^2 + (1-1)^2 + (1-0)^2 + (0-0)^2 + (1-0)^2 +$ $(0-0)^2 + (1-0)^2 + (0-0)^2 + (1-0)^2 + (0-0)^2 + (1-1)^2 + (0-0)^2 + (0-0)^2 +$ $(1-0)^2 + (0-0)^2 + (0-0)^2 + (0-0)^2 + (0-1)^2 + (1-0)^2 + (1-0)^2 + (1-0)^2$ $= 30$
Citra Kulit 27 Tahun.jpg	$(0-0)^2 + (0-0)^2 + (0-1)^2 + (1-0)^2 + (0-1)^2 + (1-0)^2 + (0-0)^2 + (1-1)^2 +$ $(0-0)^2 + (1-0)^2 + (0-1)^2 + (0-0)^2 + (1-0)^2 + (1-1)^2 + (0-0)^2 + (0-1)^2 +$ $(1-0)^2 + (0-0)^2 + (1-1)^2 + (1-0)^2 + (1-1)^2 + (0-0)^2 + (0-0)^2 + (0-1)^2 +$ $(0-0)^2 + (1-0)^2 + (0-1)^2 + (1-0)^2 + (1-1)^2 + (0-0)^2 + (1-0)^2 + (0-0)^2 +$ $(0-0)^2 + (0-1)^2 + (0-0)^2 + (0-0)^2 + (1-1)^2 + (0-0)^2 + (0-0)^2 + (0-0)^2 +$ $(0-0)^2 + (0-1)^2 + (1-0)^2 + (1-0)^2 + (1-0)^2 + (1-1)^2 + (0-0)^2 + (1-0)^2 +$ $(0-0)^2 + (1-1)^2 + (0-0)^2 + (1-0)^2 + (0-0)^2 + (1-1)^2 + (0-0)^2 + (0-0)^2 +$

	$(1-1)^2 + (0-0)^2 + (0-0)^2 + (0-0)^2 + (0-1)^2 + (1-0)^2 + (1-0)^2 + (1-0)^2$ = 26
Citra Kulit 28 Tahun.jpg	$(0-1)^2 + (0-0)^2 + (0-1)^2 + (1-0)^2 + (0-0)^2 + (1-0)^2 + (0-0)^2 + (1-1)^2 +$ $(0-0)^2 + (1-1)^2 + (0-1)^2 + (0-0)^2 + (1-0)^2 + (1-1)^2 + (0-0)^2 + (0-1)^2 +$ $(1-1)^2 + (0-0)^2 + (1-1)^2 + (1-1)^2 + (1-0)^2 + (0-1)^2 + (0-0)^2 + (0-1)^2 +$ $(0-0)^2 + (1-0)^2 + (0-1)^2 + (1-0)^2 + (1-1)^2 + (0-0)^2 + (1-0)^2 + (0-0)^2 +$ $(0-0)^2 + (0-1)^2 + (0-0)^2 + (0-0)^2 + (1-1)^2 + (0-0)^2 + (0-0)^2 + (0-0)^2 +$ $(0-0)^2 + (0-1)^2 + (1-0)^2 + (1-0)^2 + (1-0)^2 + (1-1)^2 + (0-0)^2 + (1-0)^2 +$ $(0-0)^2 + (1-1)^2 + (0-0)^2 + (1-0)^2 + (0-0)^2 + (1-1)^2 + (0-0)^2 + (0-0)^2 +$ $(1-1)^2 + (0-0)^2 + (0-0)^2 + (0-0)^2 + (0-1)^2 + (1-0)^2 + (1-0)^2 + (1-0)^2$ = 21
Citra Kulit 29 Tahun.jpg	$(0-0)^2 + (0-0)^2 + (0-0)^2 + (1-0)^2 + (0-1)^2 + (1-0)^2 + (0-0)^2 + (1-1)^2 +$ $(0-0)^2 + (1-0)^2 + (0-1)^2 + (0-0)^2 + (1-0)^2 + (1-1)^2 + (0-1)^2 + (0-1)^2 +$ $(1-0)^2 + (0-1)^2 + (1-1)^2 + (1-0)^2 + (1-1)^2 + (0-0)^2 + (0-0)^2 + (0-1)^2 +$ $(0-1)^2 + (1-0)^2 + (0-1)^2 + (1-0)^2 + (1-1)^2 + (0-0)^2 + (1-0)^2 + (0-0)^2 +$ $(0-0)^2 + (0-1)^2 + (0-0)^2 + (0-0)^2 + (1-1)^2 + (0-0)^2 + (0-0)^2 + (0-0)^2 +$ $(0-0)^2 + (0-1)^2 + (1-0)^2 + (1-0)^2 + (1-0)^2 + (1-1)^2 + (0-0)^2 + (1-0)^2 +$ $(0-0)^2 + (1-1)^2 + (0-1)^2 + (1-0)^2 + (0-0)^2 + (1-1)^2 + (0-0)^2 + (0-0)^2 +$ $(1-1)^2 + (0-0)^2 + (0-0)^2 + (0-1)^2 + (0-1)^2 + (1-0)^2 + (1-0)^2 + (1-0)^2$ = 29
Citra Kulit 30 Tahun.jpg	$(0-0)^2 + (0-1)^2 + (0-0)^2 + (1-0)^2 + (0-0)^2 + (1-1)^2 + (0-0)^2 + (1-1)^2 +$ $(0-1)^2 + (1-0)^2 + (0-1)^2 + (0-0)^2 + (1-0)^2 + (1-1)^2 + (0-0)^2 + (0-0)^2 +$ $(1-0)^2 + (0-0)^2 + (1-0)^2 + (1-1)^2 + (1-0)^2 + (0-0)^2 + (0-1)^2 + (0-0)^2 +$ $(0-0)^2 + (1-0)^2 + (0-0)^2 + (1-0)^2 + (1-1)^2 + (0-0)^2 + (1-0)^2 + (0-1)^2 +$ $(0-0)^2 + (0-1)^2 + (0-0)^2 + (0-0)^2 + (1-1)^2 + (0-0)^2 + (0-0)^2 + (0-1)^2 +$ $(0-0)^2 + (0-1)^2 + (1-0)^2 + (1-0)^2 + (1-0)^2 + (1-0)^2 + (0-1)^2 + (1-0)^2 +$ $(1-0)^2 + (1-0)^2 + (0-0)^2 + (1-0)^2 + (0-0)^2 + (1-1)^2 + (0-0)^2 + (0-0)^2 +$ $(1-0)^2 + (0-1)^2 + (0-0)^2 + (0-0)^2 + (0-1)^2 + (1-0)^2 + (1-0)^2 + (1-0)^2$ = 32
Citra Kulit 31 Tahun.jpg	$(0-0)^2 + (0-0)^2 + (0-0)^2 + (1-1)^2 + (0-0)^2 + (1-1)^2 + (0-0)^2 + (1-1)^2 +$ $(0-1)^2 + (1-0)^2 + (0-1)^2 + (0-0)^2 + (1-1)^2 + (1-0)^2 + (0-1)^2 + (0-0)^2 +$ $(1-0)^2 + (0-1)^2 + (1-0)^2 + (1-1)^2 + (1-0)^2 + (0-0)^2 + (0-1)^2 + (0-0)^2 +$ $(0-0)^2 + (1-0)^2 + (0-1)^2 + (1-0)^2 + (1-0)^2 + (0-0)^2 + (1-0)^2 + (0-1)^2 +$ $(0-0)^2 + (0-1)^2 + (0-0)^2 + (0-0)^2 + (1-1)^2 + (0-0)^2 + (0-0)^2 + (0-1)^2 +$ $(0-0)^2 + (0-1)^2 + (1-0)^2 + (1-1)^2 + (1-0)^2 + (1-0)^2 + (0-1)^2 + (1-0)^2 +$ $(1-0)^2 + (1-0)^2 + (0-1)^2 + (1-0)^2 + (0-0)^2 + (1-1)^2 + (0-0)^2 + (0-0)^2 +$ $(1-0)^2 + (0-1)^2 + (0-0)^2 + (0-0)^2 + (0-1)^2 + (1-0)^2 + (1-0)^2 + (1-0)^2$ = 34
Citra Kulit 32 Tahun.jpg	$(0-1)^2 + (0-0)^2 + (0-1)^2 + (1-0)^2 + (0-1)^2 + (1-0)^2 + (0-0)^2 + (1-1)^2 +$ $(0-0)^2 + (1-1)^2 + (0-0)^2 + (0-0)^2 + (1-0)^2 + (1-1)^2 + (0-0)^2 + (0-1)^2 +$ $(1-1)^2 + (0-0)^2 + (1-1)^2 + (1-0)^2 + (1-1)^2 + (0-0)^2 + (0-1)^2 + (0-1)^2 +$ $(0-0)^2 + (1-0)^2 + (0-0)^2 + (1-0)^2 + (1-0)^2 + (0-1)^2 + (1-0)^2 + (0-1)^2 +$ $(0-0)^2 + (0-1)^2 + (0-0)^2 + (0-0)^2 + (1-1)^2 + (0-0)^2 + (0-0)^2 + (0-0)^2 +$ $(0-1)^2 + (0-0)^2 + (1-0)^2 + (1-1)^2 + (1-0)^2 + (1-1)^2 + (0-0)^2 + (1-0)^2 +$ $(1-0)^2 + (1-1)^2 + (0-1)^2 + (1-0)^2 + (0-1)^2 + (1-0)^2 + (0-0)^2 + (0-0)^2 +$ $(1-1)^2 + (0-0)^2 + (0-0)^2 + (0-1)^2 + (0-0)^2 + (1-0)^2 + (1-0)^2 + (1-0)^2$ = 30
Citra Kulit 33 Tahun.jpg	$(0-0)^2 + (0-0)^2 + (0-1)^2 + (1-0)^2 + (0-1)^2 + (1-0)^2 + (0-0)^2 + (1-1)^2 +$ $(0-1)^2 + (1-0)^2 + (0-0)^2 + (0-1)^2 + (1-0)^2 + (1-1)^2 + (0-0)^2 + (0-1)^2 +$ $(1-0)^2 + (0-1)^2 + (1-1)^2 + (1-0)^2 + (1-0)^2 + (0-0)^2 + (0-1)^2 + (0-1)^2 +$ $(0-1)^2 + (1-0)^2 + (0-0)^2 + (1-0)^2 + (1-0)^2 + (0-1)^2 + (1-0)^2 + (0-0)^2 +$ $(0-0)^2 + (0-1)^2 + (0-0)^2 + (0-0)^2 + (1-1)^2 + (0-0)^2 + (0-0)^2 + (0-0)^2 +$ $(0-1)^2 + (0-0)^2 + (1-0)^2 + (1-1)^2 + (1-0)^2 + (1-1)^2 + (0-0)^2 + (1-0)^2 +$ $(1-0)^2 + (1-1)^2 + (0-0)^2 + (1-0)^2 + (0-1)^2 + (1-0)^2 + (0-1)^2 + (0-0)^2 +$ $(1-1)^2 + (0-0)^2 + (0-0)^2 + (0-1)^2 + (0-0)^2 + (1-0)^2 + (1-0)^2 + (1-1)^2$

	= 34
Citra Kulit 34 Tahun.jpg	$(0-0)^2 + (0-0)^2 + (0-0)^2 + (1-0)^2 + (0-1)^2 + (1-0)^2 + (0-0)^2 + (1-1)^2 +$ $(0-1)^2 + (1-0)^2 + (0-1)^2 + (0-0)^2 + (1-0)^2 + (1-1)^2 + (0-0)^2 + (0-0)^2 +$ $(1-0)^2 + (0-1)^2 + (1-0)^2 + (1-0)^2 + (1-1)^2 + (0-0)^2 + (0-1)^2 + (0-0)^2 +$ $(0-0)^2 + (1-0)^2 + (0-1)^2 + (1-0)^2 + (1-1)^2 + (0-0)^2 + (1-0)^2 + (0-1)^2 +$ $(0-0)^2 + (0-1)^2 + (0-0)^2 + (0-1)^2 + (1-0)^2 + (0-0)^2 + (0-1)^2 + (0-0)^2 +$ $(0-0)^2 + (0-0)^2 + (1-1)^2 + (1-0)^2 + (1-0)^2 + (1-1)^2 + (0-0)^2 + (1-0)^2 +$ $(1-0)^2 + (1-0)^2 + (0-0)^2 + (1-0)^2 + (0-0)^2 + (1-1)^2 + (0-0)^2 + (0-0)^2 +$ $(1-1)^2 + (0-0)^2 + (0-0)^2 + (0-0)^2 + (0-1)^2 + (1-0)^2 + (1-0)^2 + (1-0)^2$ $= 31$
Citra Kulit 35 Tahun.jpg	$(0-0)^2 + (0-0)^2 + (0-0)^2 + (1-0)^2 + (0-0)^2 + (1-1)^2 + (0-0)^2 + (1-0)^2 +$ $(0-0)^2 + (1-0)^2 + (0-0)^2 + (0-1)^2 + (1-1)^2 + (1-0)^2 + (0-0)^2 + (0-0)^2 +$ $(1-0)^2 + (0-0)^2 + (1-1)^2 + (1-0)^2 + (1-1)^2 + (0-0)^2 + (0-0)^2 + (0-0)^2 +$ $(0-0)^2 + (1-1)^2 + (0-0)^2 + (1-1)^2 + (1-0)^2 + (0-1)^2 + (1-0)^2 + (0-0)^2 +$ $(0-0)^2 + (0-0)^2 + (0-1)^2 + (0-0)^2 + (1-1)^2 + (0-0)^2 + (0-1)^2 + (0-1)^2 +$ $(0-0)^2 + (0-1)^2 + (1-0)^2 + (1-0)^2 + (1-0)^2 + (1-1)^2 + (0-0)^2 + (1-0)^2 +$ $(1-1)^2 + (1-0)^2 + (0-1)^2 + (1-0)^2 + (0-1)^2 + (1-0)^2 + (0-0)^2 + (0-0)^2 +$ $(1-0)^2 + (0-1)^2 + (0-0)^2 + (0-0)^2 + (0-1)^2 + (1-0)^2 + (1-0)^2 + (1-0)^2$ $= 29$
Citra Kulit 36 Tahun.jpg	$(0-1)^2 + (0-0)^2 + (0-0)^2 + (1-0)^2 + (0-0)^2 + (1-0)^2 + (0-0)^2 + (1-0)^2 +$ $(0-0)^2 + (1-1)^2 + (0-0)^2 + (0-0)^2 + (1-0)^2 + (1-0)^2 + (0-0)^2 + (0-0)^2 +$ $(1-0)^2 + (0-0)^2 + (1-1)^2 + (1-0)^2 + (1-0)^2 + (0-0)^2 + (0-0)^2 + (0-1)^2 +$ $(0-0)^2 + (1-0)^2 + (0-0)^2 + (1-1)^2 + (1-0)^2 + (0-0)^2 + (1-1)^2 + (0-0)^2 +$ $(0-0)^2 + (0-0)^2 + (0-1)^2 + (0-0)^2 + (1-0)^2 + (0-1)^2 + (0-0)^2 + (0-1)^2 +$ $(0-0)^2 + (0-0)^2 + (1-1)^2 + (1-1)^2 + (1-0)^2 + (1-0)^2 + (0-1)^2 + (1-0)^2 +$ $(1-0)^2 + (1-1)^2 + (0-0)^2 + (1-1)^2 + (0-0)^2 + (1-1)^2 + (0-0)^2 + (0-0)^2 +$ $(1-1)^2 + (0-0)^2 + (0-1)^2 + (0-0)^2 + (0-1)^2 + (1-0)^2 + (1-0)^2 + (1-0)^2$ $= 26$
Citra Kulit 37 Tahun.jpg	$(0-1)^2 + (0-0)^2 + (0-0)^2 + (1-0)^2 + (0-0)^2 + (1-0)^2 + (0-0)^2 + (1-0)^2 +$ $(0-0)^2 + (1-1)^2 + (0-0)^2 + (0-0)^2 + (1-0)^2 + (1-0)^2 + (0-0)^2 + (0-0)^2 +$ $(1-0)^2 + (0-0)^2 + (1-1)^2 + (1-0)^2 + (1-0)^2 + (0-0)^2 + (0-0)^2 + (0-1)^2 +$ $(0-0)^2 + (1-0)^2 + (0-0)^2 + (1-1)^2 + (1-0)^2 + (0-0)^2 + (1-1)^2 + (0-0)^2 +$ $(0-0)^2 + (0-0)^2 + (0-1)^2 + (0-0)^2 + (1-0)^2 + (0-1)^2 + (0-0)^2 + (0-1)^2 +$ $(0-0)^2 + (0-0)^2 + (1-1)^2 + (1-1)^2 + (1-0)^2 + (1-0)^2 + (0-1)^2 + (1-0)^2 +$ $(1-0)^2 + (1-1)^2 + (0-0)^2 + (1-1)^2 + (0-0)^2 + (1-1)^2 + (0-0)^2 + (0-0)^2 +$ $(1-1)^2 + (0-0)^2 + (0-1)^2 + (0-0)^2 + (0-1)^2 + (1-0)^2 + (1-0)^2 + (1-0)^2$ $= 26$
Citra Kulit 38 Tahun.jpg	$(0-0)^2 + (0-0)^2 + (0-1)^2 + (1-0)^2 + (0-0)^2 + (1-1)^2 + (0-0)^2 + (1-0)^2 +$ $(0-0)^2 + (1-1)^2 + (0-0)^2 + (0-0)^2 + (1-1)^2 + (1-0)^2 + (0-0)^2 + (0-0)^2 +$ $(1-1)^2 + (0-0)^2 + (1-1)^2 + (1-0)^2 + (1-0)^2 + (0-0)^2 + (0-0)^2 + (0-0)^2 +$ $(0-0)^2 + (1-1)^2 + (0-0)^2 + (1-1)^2 + (1-0)^2 + (0-1)^2 + (1-0)^2 + (0-0)^2 +$ $(0-1)^2 + (0-0)^2 + (0-0)^2 + (0-0)^2 + (1-1)^2 + (0-0)^2 + (0-1)^2 + (0-1)^2 +$ $(0-0)^2 + (0-1)^2 + (1-0)^2 + (1-0)^2 + (1-0)^2 + (1-1)^2 + (0-1)^2 + (1-0)^2 +$ $(1-0)^2 + (1-0)^2 + (0-1)^2 + (1-0)^2 + (0-1)^2 + (1-0)^2 + (0-1)^2 + (0-0)^2 +$ $(1-0)^2 + (0-0)^2 + (0-0)^2 + (0-1)^2 + (0-1)^2 + (1-0)^2 + (1-1)^2 + (1-0)^2$ $= 29$
Citra Kulit 39 Tahun.jpg	$(0-0)^2 + (0-1)^2 + (0-0)^2 + (1-0)^2 + (0-1)^2 + (1-0)^2 + (0-0)^2 + (1-1)^2 +$ $(0-1)^2 + (1-0)^2 + (0-1)^2 + (0-0)^2 + (1-0)^2 + (1-1)^2 + (0-0)^2 + (0-1)^2 +$ $(1-0)^2 + (0-1)^2 + (1-0)^2 + (1-1)^2 + (1-0)^2 + (0-0)^2 + (0-1)^2 + (0-0)^2 +$ $(0-1)^2 + (1-0)^2 + (0-1)^2 + (1-0)^2 + (1-1)^2 + (0-0)^2 + (1-0)^2 + (0-1)^2 +$ $(0-0)^2 + (0-1)^2 + (0-0)^2 + (0-1)^2 + (1-0)^2 + (0-0)^2 + (0-1)^2 + (0-0)^2 +$ $(0-0)^2 + (0-0)^2 + (1-1)^2 + (1-0)^2 + (1-0)^2 + (1-1)^2 + (0-0)^2 + (1-0)^2 +$ $(1-0)^2 + (1-0)^2 + (0-0)^2 + (1-1)^2 + (0-0)^2 + (1-0)^2 + (0-1)^2 + (0-0)^2 +$ $(1-0)^2 + (0-0)^2 + (0-0)^2 + (0-0)^2 + (0-1)^2 + (1-0)^2 + (1-0)^2 + (1-1)^2$ $= 35$

Citra Kulit 40 Tahun.jpg	$(0-1)^2 + (0-0)^2 + (0-0)^2 + (1-1)^2 + (0-0)^2 + (1-1)^2 + (0-0)^2 + (1-0)^2 +$ $(0-0)^2 + (1-0)^2 + (0-1)^2 + (0-0)^2 + (1-1)^2 + (1-0)^2 + (0-0)^2 + (0-1)^2 +$ $(1-0)^2 + (0-1)^2 + (1-1)^2 + (1-0)^2 + (1-1)^2 + (0-0)^2 + (0-1)^2 + (0-0)^2 +$ $(0-0)^2 + (1-1)^2 + (0-0)^2 + (1-1)^2 + (1-0)^2 + (0-1)^2 + (1-0)^2 + (0-0)^2 +$ $(0-1)^2 + (0-0)^2 + (0-1)^2 + (0-0)^2 + (1-1)^2 + (0-0)^2 + (0-1)^2 + (0-0)^2 +$ $(0-0)^2 + (0-1)^2 + (1-0)^2 + (1-1)^2 + (1-0)^2 + (1-1)^2 + (0-0)^2 + (1-0)^2 +$ $(1-1)^2 + (1-0)^2 + (0-1)^2 + (1-0)^2 + (0-1)^2 + (1-0)^2 + (0-0)^2 + (0-0)^2 +$ $(1-0)^2 + (0-1)^2 + (0-0)^2 + (0-0)^2 + (0-1)^2 + (1-0)^2 + (1-0)^2 + (1-0)^2$ $= 31$

Jika nilai eror masing-masing *template* sudah diketahui, maka cari nilai eror terkecil. Nilai eror terkecil adalah *template* yang paling sesuai dengan citra masukan. Nilai terkecil dari hasil perhitungan pada tabel 3.2 adalah 21. Maka citra *template* yang paling sesuai dengan citra masukan adalah citra kulit 20 Tahun.jpg.

3.3 Analisis Kebutuhan Non-Fungsional

Adapun kebutuhan non-fungsional untuk menjalankan aplikasi ini meliputi kebutuhan perangkat keras, kebutuhan perangkat lunak dan pengguna sistem yang akan memakai aplikasi. Analisis kebutuhan non-fungsional bertujuan agar aplikasi yang dibangun dapat digunakan sesuai dengan kebutuhan pengguna aplikasi dalam mencari informasi yang dibutuhkan.

3.3.1 Analisis Kebutuhan Perangkat Keras

Kebutuhan perangkat keras untuk menjalankan aplikasi ini adalah smartphone Android yang memiliki beberapa spesifikasi utama sebagai berikut:

1. Sistem operasi Android v4.0 (*Ice Cream Sandwich*).
2. Kamera 5 MP, untuk keperluan pengambilan gambar yang dilakukan secara langsung.
3. *Random Access Memory* (RAM) 1 GB
4. LCD *capacitive touchscreen*
5. Media penyimpanan eksternal, untuk keperluan menyimpan gambar yang dihasilkan oleh program aplikasi.

3.3.2 Analisis Kebutuhan Perangkat Lunak

Perangkat lunak yang dipakai dalam pengembangan aplikasi, diantaranya adalah sebagai berikut:

1. Sistem Operasi Windows 7 Home Edition

2. Java Development Kit versi 1.8.0, untuk keperluan kompilasi kode-kode program.
3. Eclipse IDE 3.7, digunakan sebagai perangkat kompilasi kode-kode program.
4. SQLite untuk keperluan menyimpan data didalam database android.
5. Android SDK, untuk pengambilan aplikasi android.
6. Android Development Tool, sebagai plugin untuk mengintegrasikan android SDK pada eclipse.

3.3.3 Analisis Kebutuhan User

Sfesifikasi pengguna sistem dimaksudkan untuk mengetahui siapa saja *actor* yang terlibat dalam menjalankan sistem:

Pengguna aplikasi yaitu para pengguna *smartphone* dengan *platform* android versi 4.0 atau lebih tinggi.

Tabel 3.3 Karakteristik Pengguna

Pengguna	Tugas
User	Mulai menjalankan aplikasi
	Mengambil gambar melalui kamera *default*
	Mengklik button *watershed* dan *template matching* untuk mengubah gambar kulit menjadi sebuah informasi usia kulit
	Keluar dari aplikasi

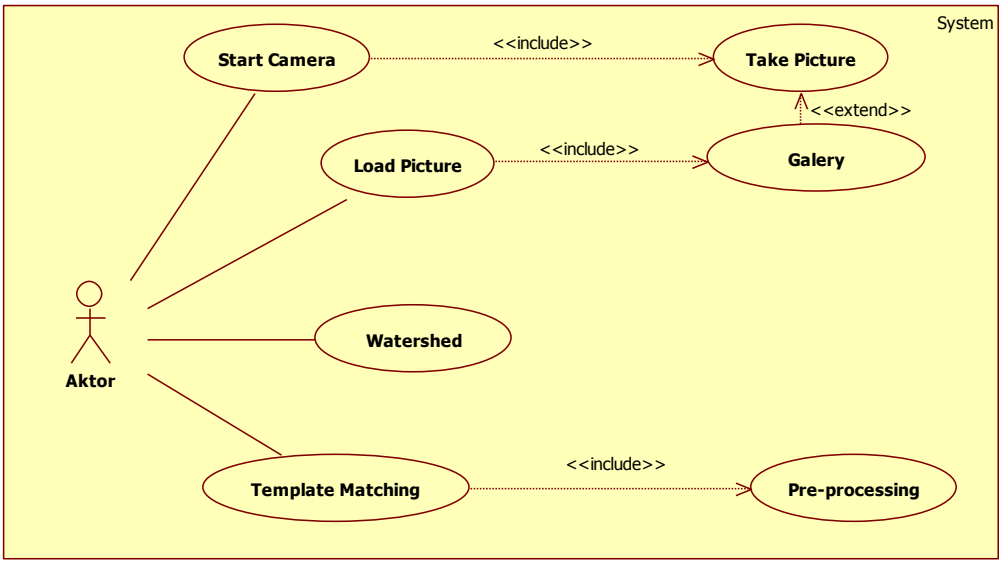
3.4 Analisis Kebutuhan Fungsional

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

Analisis yang dilakukan dimodelkan dengan UML (*Unfield Modeling Language*), Tahap-tahap pemodelan dalam analisis tersebut antaranya diagram *use case*, diagram *activity*, diagram *class* dan diagram *sequence*. Diagram tersebut dipergunakan sebagai desain rancangan untuk mempermudah proses pengembangan aplikasi khususnya untuk aplikasi yang dikembangkan menggunakan pemrograman berorientasi objek.

3.4.1 Use Case Diagram

Use case diagram digunakan untuk mengetahui apa saja yang dapat dilakukan oleh pengguna/aktor terhadap fungsionalitas yang terdapat pada aplikasi yang dirancang.



Gambar 3.9 Diagram Use Case

3.4.1.1. Identifikasi Actor

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

Tabel 3.4 Aktor Use Case

No	Aktor	Deskripsi
A-01	User	Merupakan aktor dari aplikasi yang dibangun atau pengguna aplikasi.

Penggambaran *use case* diagram sistem yaitu *use case* menunjukkan bagaimana interaksi antara aktor dan sistem. Aktor user yaitu yang menjalankan aplikasi.

3.4.1.2. Identifikasi Use Case

Berikut tabel Identifikasi *Use Case* yang terdapat pada aplikasi:

Tabel 3.5 Use Case Aktor

Use Case Aplikasi User		
No	**Use Case**	**Deskripsi**
UC-01	Take Picture	Fungsional untuk memulai mengambil gambar menggunakan kamera
UC-02	Load Picture	Fungsional untuk memungkinkan pengguna memilih foto yang sudah tersimpan di galery
UC-03	Watershed	Fungsional untuk memproses citra yang sudah di greyscale lalu melakukan segmentasi watershed pada citra
UC-04	Template Matching	Fungsional untuk mencocokkan tiap-tiap bagian dari citra uji dengan citra yang menjadi acuan (*template*) yang sudah tersimpan didataset dan memunculkan informasi berupa usia kulit untuk menentukan penuaan dini pada kulit

3.4.2 Skenario Use Case

Skenario *use case* mendeskripsikan urutan langkah-langkah dalam proses bisnis, baik dilakukan aktor terhadap sistem maupun yang dilakukan oleh sistem terhadap aktor.

3.4.2.1. Skenario Use Case Take Picture

Skenario *use case take picture* gambar dapat dilihat pada tabel 3.6.

Tabel 3.6 Skenario Use Case Take Picture

Identifikasi	
Nama Use Case User	Take Picture
Aktor	*User*
Tujuan	Digunakan untuk mengambil gambar dengan menyorotkan kamera pada objek yang akan diambil
Keadaan Awal	Sistem menampilkan kamera
Skenario Utama	
Aksi Aktor	**Reaksi Sistem**
	1. Menampilkan Menu Utama
2. Menyorotkan kamera ponsel ke objek yang akan diambil	
3. Pilih tombol kamera untuk pengambilan gambar	
	4. Menyimpan gambar

3.4.2.2. Skenario Use Case Load Picture

Skenario *use case load picture* gambar dapat dilihat pada tabel 3.7.

Tabel 3.7 Skenario Use Case Load Picture

Identifikasi	
Nama Use Case User	Load Picture
Aktor	*User*
Tujuan	Proses untuk memilih foto melalui galery
Keadaan Awal	Sistem menampilkan galery
Skenario Utama	
Aksi Aktor	**Reaksi Sistem**
	1. Menampilkan galery

3.4.2.3. Skenario Use Case Watershed

Skenario *use case watershed* gambar dapat dilihat pada tabel 3.8.

Tabel 3.8 Skenario Use Case Watershed

Identifikasi	
Nama Use Case User	Watershed
Aktor	*User*
Tujuan	Digunakan untuk melihat proses watershed pada gambar
Skenario Utama	
Aksi Aktor	**Reaksi Sistem**
1. Memilih button watershed	
	2. Menampilkan proses watershed pada gambar

3.4.2.4. Skenario Use Case Pre-processing

Skenario *use case* pre-processing dapat dilihat pada tabel 3.9.

Tabel 3.9 Skenario Use Case Pre-processing

Identifikasi	
Nama Use Case User	Pre-processing
Aktor	*Sistem*
Tujuan	Menampilkan citra pada saat proses watershed
Skenario Utama	
Aksi Aktor	**Reaksi Sistem**
	1. Menampilkan pre-processing

3.4.2.5. Skenario Use Case Template Matching

Skenario *use case* template matching dapat dilihat pada tabel 3.10.

Tabel 3.10 Skenario Use Case Template Matching

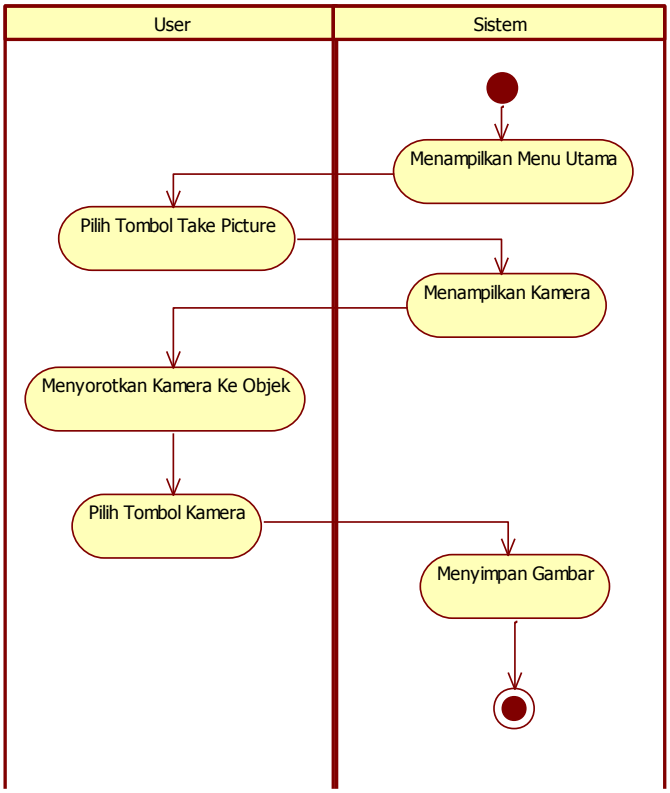
Identifikasi	
Nama Use Case User	Template Matching
Aktor	*Sistem*
Tujuan	Menampilkan hasil pencocokan
Skenario Utama	
Aksi Aktor	**Reaksi Sistem**
	1. Menampilkan informasi estimasi umur kulit

3.4.3 Activity Diagram

Activity diagram digunakan untuk menggambarkan alur rangkaian proses yang terjadi di dalam sistem aplikasi serta menjelaskan bagaimana sebuah *use case* terjadi atau dipicu dari *use case* yang lain.

3.4.3.1. Activity Diagram Take Picture

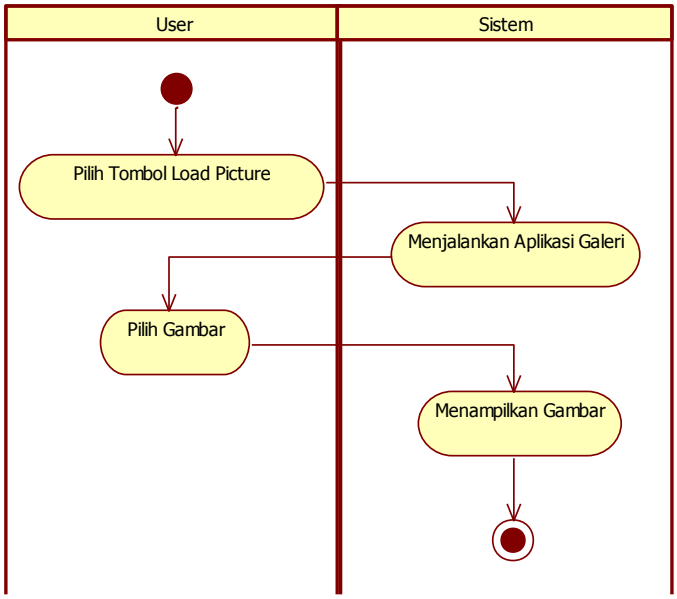
Activity diagram *take picture* menggambarkan alur aktifitas untuk memulai aplikasi antara *user* dan sistem seperti terlihat pada gambar 3.10.



Gambar 3.10 Activity Diagram *Take Picture*

3.4.3.2. Activity Diagram Load Picture

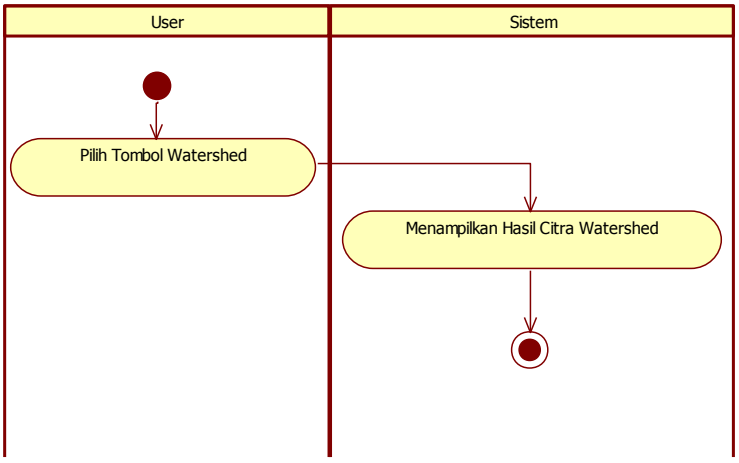
Activity diagram *load picture* menggambarkan alur aktifitas proses *load picture* antara *user* dan sistem seperti terlihat pada gambar 3.11.



Gambar 3.11 Activity Diagram *Load Picture*

3.4.3.3. Activity Diagram Watershed

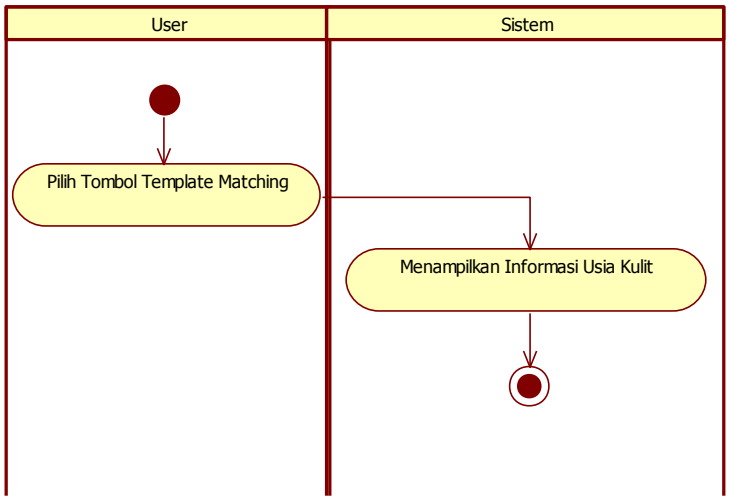
Activity diagram proses segmentasi *watershed* menggambarkan alur aktifitas proses segmentasi *watershed* antara *user* dan sistem seperti terlihat pada gambar 3.12.



Gambar 3.12 Activity Diagram Proses Segementasi *Watershed*

3.4.3.4. Activity Diagram Template Matching

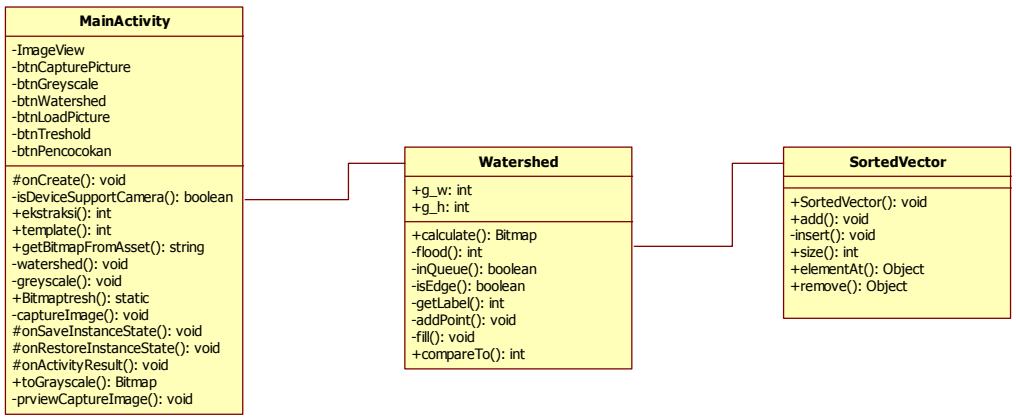
Activity diagram *template matching* menggambarkan alur aktifitas proses template matching dari sistem seperti terlihat pada gambar 3.13.



Gambar 3.13 Activity Diagram Proses *Template Matching*

3.4.4 Class Diagram

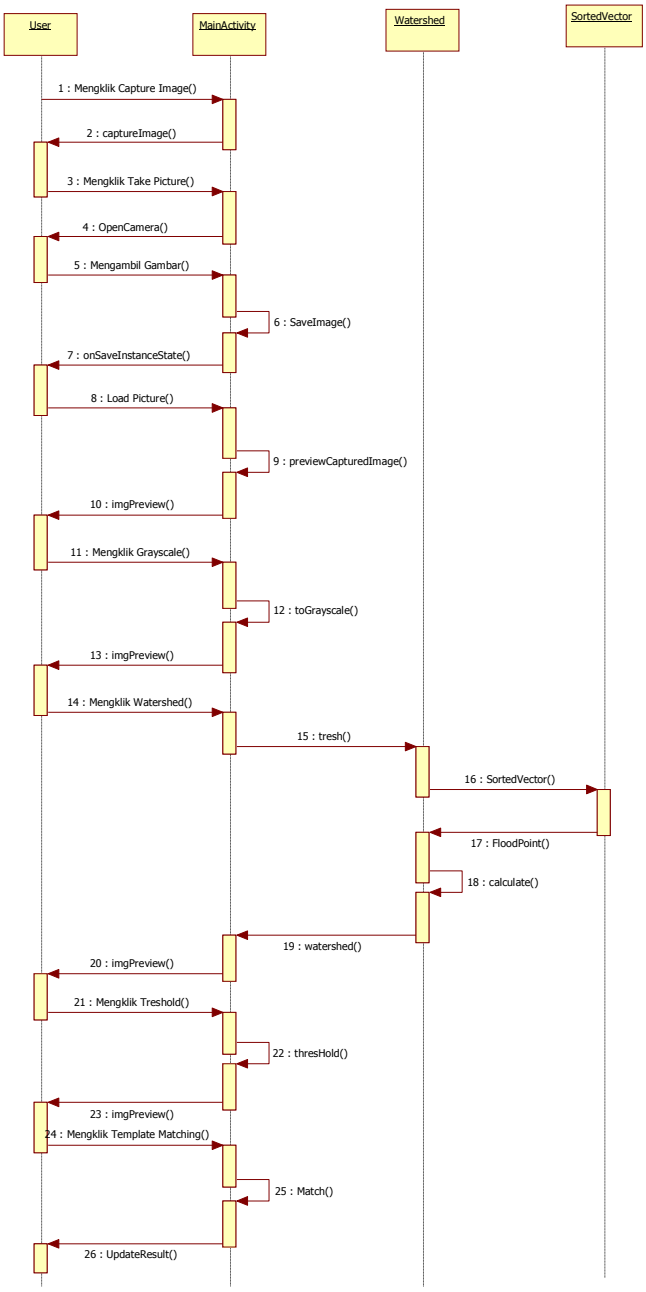
Class Diagram menggambarkan struktur dan hubungan antar objek- objek yang ada pada sistem. Struktur itu meliputi atribut-atribut dan metode metode yang ada pada masing-masing class. Hubungannya dari masing-masing class yang ada tersebut digambarkan seperti pada gambar dibawah.



Gambar 3.14 Class Diagram

3.4.5 Sequence Diagram

Diagram *sequence* menggambarkan urutan aktifitas yang terjadi di dalam sistem. Diagram ini menunjukkan pengiriman pesan-pesan yang melewati objek yang terlibat di dalam sistem serta tahapan langkah yang harus dilalui untuk mencapai *use case* tertentu.



Gambar 3.15 Sequence Diagram

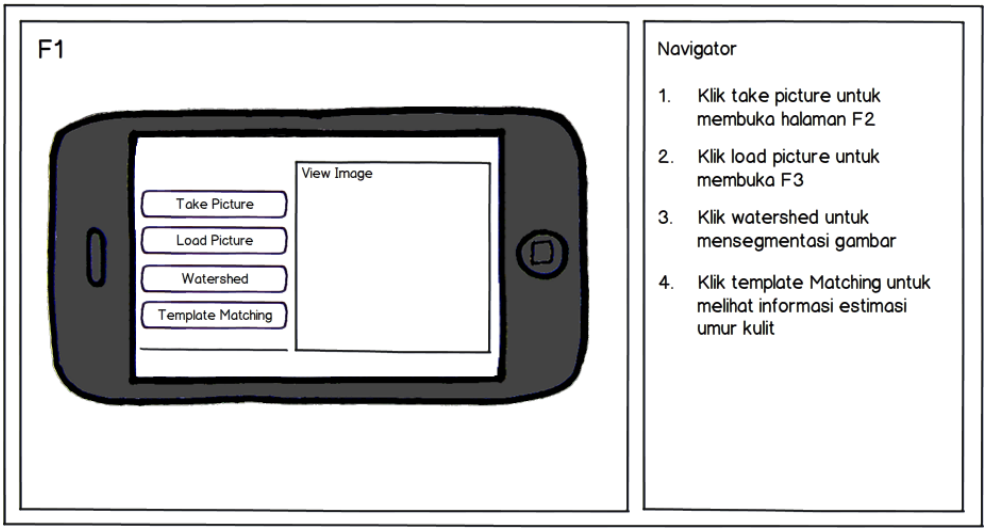
3.5 Perancangan Sistem

Adapun tahapan yang dilakukan dalam perancangan sistem ini membahas mengenai tujuan perancangan antarmuka, perancangan jaringan simantik, dan perancangan prosedural.

3.5.1 Perancangan Antarmuka Sistem

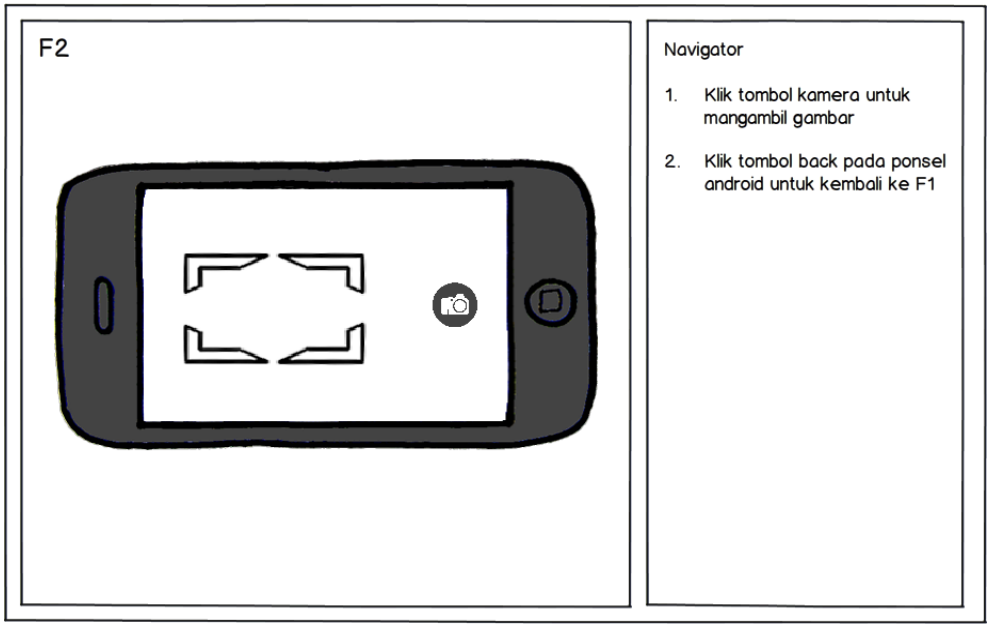
Tahap perancangan antarmuka bertujuan untuk memberikan gambaran tentang aplikasi yang akan dibangun, sehingga akan mempermudah dalam mengimplementasikan aplikasi serta akan memudahkan pembuatan aplikasi. Berikut ini gambaran perancangan antarmuka yang ada dari aplikasi yang akan dibangun yaitu:

3.5.1.1. Perancangan Antar Muka Menu Utama



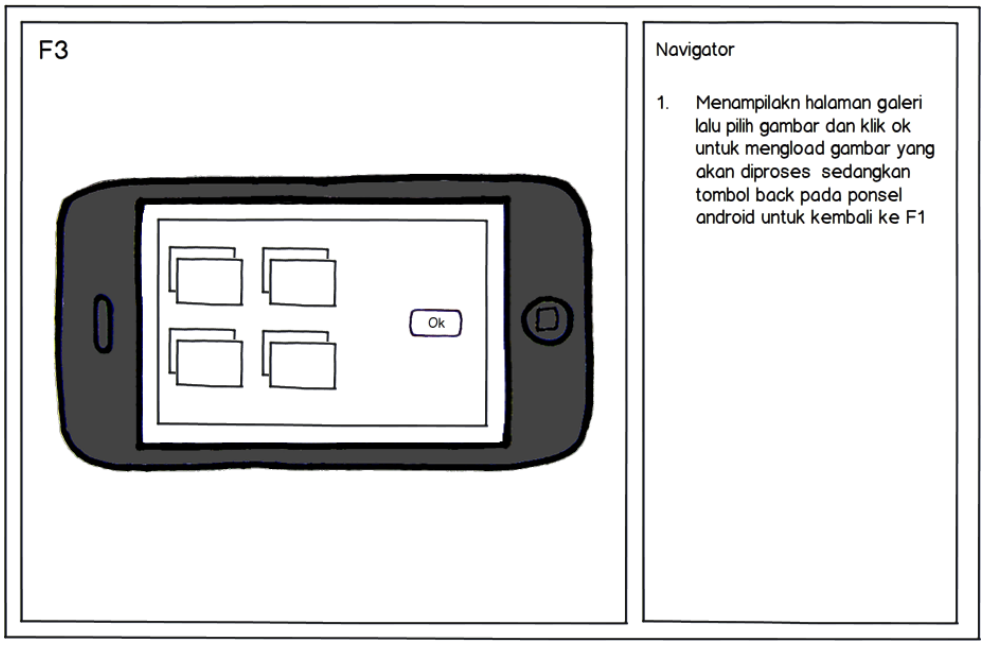
Gambar 3.16 Perancangan Antar Muka Menu Utama

3.5.1.2. Perancangan Antar Muka Take Picture



Gambar 3.17 Perancangan Antar *Take Picture*

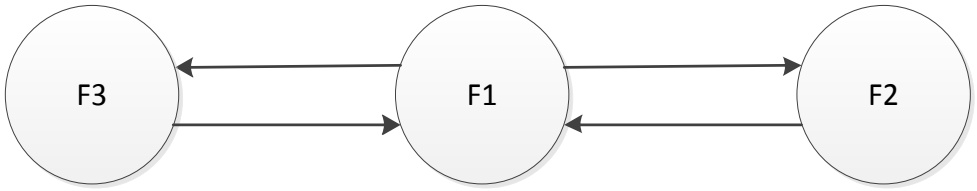
3.5.1.3. Perancangan Antar Muka Load Picture



Gambar 3.18 Perancangan Antar Muka *Load Picture*

3.5.2 Perancangan Jaringan Semantik

Jaringan semantik merupakan gambaran grafis yang menunjukkan antar objek (lingkaran) dan garis yang menggambarkan formasi antar objek tersebut. Adapun jaringan semantik untuk aplikasi mendeteksi penuaan dini pad kulit dapat dilihat pada gambar dibawah:



Gambar 3.19 Jaringan Semantik

Keterangan:

F1 : Tampilan Menu Utama

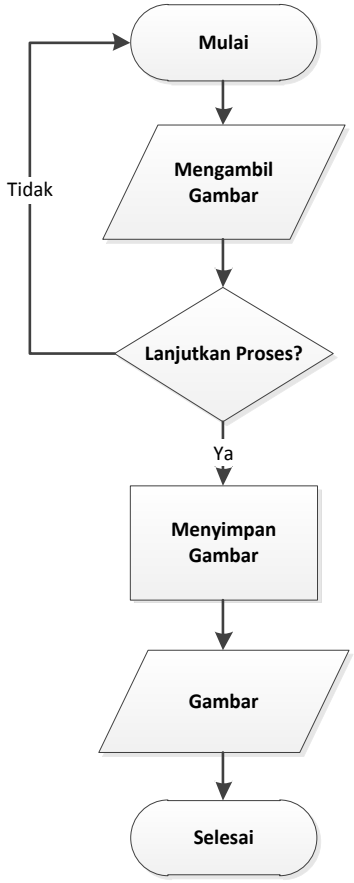
F2 : Tampilan Take Picture

F3 : Tampilan Load Picture

3.5.3 Perancangan Prosedural

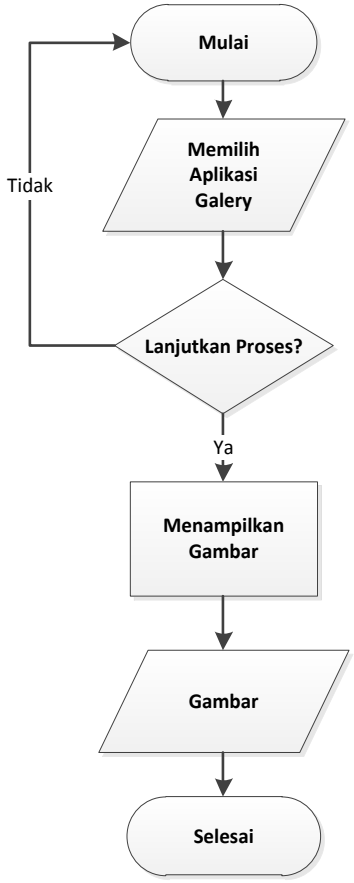
Perancangan prosedural digunakan untuk menggambarkan langkah-langkah atau alur dari setiap proses yang ada pada sistem. Berikut adalah rancangan prosedural dari aplikasi yang akan dibangun.

3.5.3.1. Perancangan Prosedural Take Picture



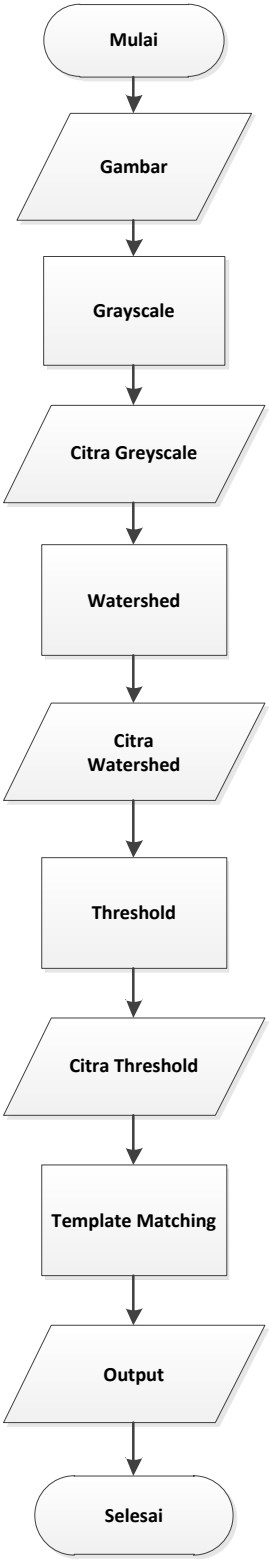
Gambar 3.20 Prosedural Take Picture

3.5.3.2. Perancangan Prosedural Load



Gambar 3.21 Prosedural Load Picture

3.5.3.3. Perancangan Prosedural Pre-processing



Gambar 3.22 Prosedural Pre-processing