BAB 3
ANALISIS DAN PERANCANGAN SISTEM

1.1 Analisis Masalah

Pada umumnya fitur pencarian di kebanyakan sistem operasi pada komputer berbasis teks. Fitur tersebut mungkin efektif jika pengguna ingat nama file yang ingin dicari. Namun untuk pencarian citra pada khususnya fitur pencarian berbasis teks dinilai kurang efektif pada saat merepresentasikan citra yang dicari di dalam sebuah database, sehingga seringkali diperoleh pencarian citra yang tidak sesuai dengan yang diinginkan. Hal tersebut bisa terjadi dikarenakan keterbatasan *user* dalam mengingat kata kunci atau nama *file* citra yang diinginkan. Oleh karena itu dibuat aplikasi *image retrieval* untuk melakukan proses pencarian citra. Tujuan utamanya untuk mempermudah *user* dalam melakukan pencarian citra.

Metode yang digunakan untuk melakukan pencarian citra ini adalah *Histogram Euclidean Distance*, yaitu metode yang dapat melakukan pencarian berdasarkan fitur warna pada suatu citra. Cara kerja metode ini adalah dengan membuat histogram warna dari citra masukan dan citra yang terdapat pada gallery komputer lalu membandingkan jarak antar citra tersebut. dimana semakin kecil nilai komputasinya, maka semakin mirip citra tersebut.

1.2 Analisis Sistem

Analisis sistem merupakan suatu cara yang digunakan dalam mengidentifikasi dan mengevaluasi suatu permasalahan, hambatan yang dihadapi dan kebutuhan yang diharapkan dalam sebuah sistem. Analisa 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. Analisis sistem yang dilakukan untuk membangun aplikasi ini sebagai berikut :

1.2.1 Analisis Prosedur yang Sedang Berjalan

Berdasarkan hasil pengamatan, maka dapat diambil kesimpulan mengenai prosedur bagaimana seseorang melakukan proses pencarian citra. Penjelasannya sebagai berikut :



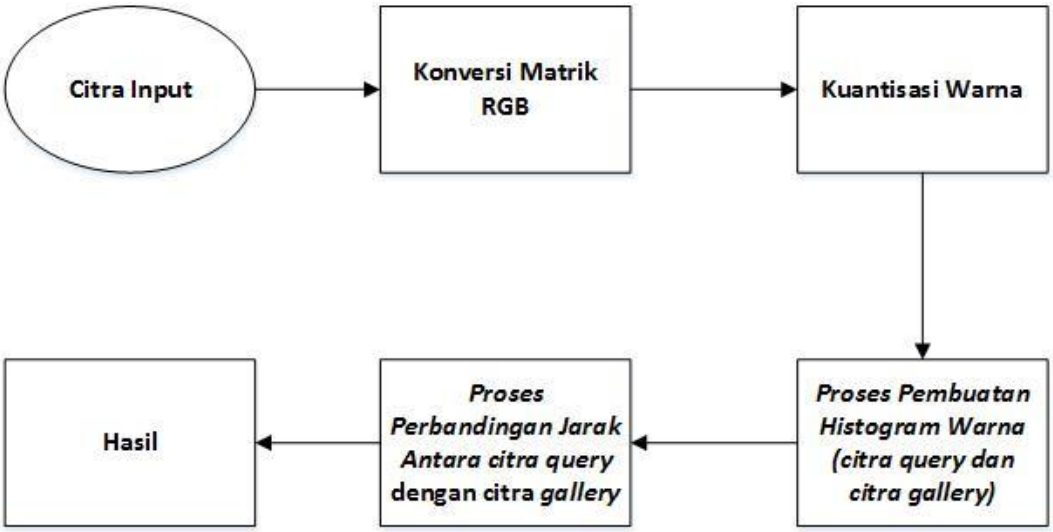
Gambar 3.1 Analisis Prosedur yang sedang berjalan

Adapun penjelasan dari gambar diatas sebagai berikut :

1. Pengguna melakukan pencarian citra dengan cara menggunakan fitur pencarian berbasis teks yang tersedia di komputer pengguna atau dengan cara manual menelusuri tiap folder yang terdapat di komputernya.
2. Pengguna dapat melihat hasil pencarian gambar menggunakan cara yang dia pilih tersebut.

1.2.2 Analisis Sistem yang Akan Dibangun

Sistem yang akan di bangun adalah sistem pencarian citra berdasarkan konten warna pada suatu citra. Sistem ini bisa membandingkan tingkat kemiripan dari suatu citra. Adapun gambaran umum tentang sistem yang akan dibangun dapat dilihat pada gambar 3.5 dibawah ini.



Gambar 3.2 Alur Proses Kerja Sistem yang Akan Dibangun

Berikut penjelasan alur proses sistem yang akan dibangun :

1. Pengguna memasukan citra/gambar yang akan dicari.
2. Citra diubah kedalam bentuk matrik-matrik RGB.
3. Kuantisasi warna merupakan proses untuk melakukan pengurangan jumlah warna dengan mengambil beberapa perwakilan warna yang dapat membedakan satu bagian dengan bagian yang lain pada citra.
4. *Histogram warna*, pada tahap ini sistem membuat *array of RGB* pada tiap channelnya dari citra *query* dan citra *gallery*.
5. setelah membuat histogram warna dari suatu citra maka selanjutnya adalah menghitung jarak antar histogram warna dari citra masukan dan citra-citra yang dicari dengan menggunakan *Euclidean Distance*.
6. Outputnya citra yang sama atau mirip dengan citra masukan.

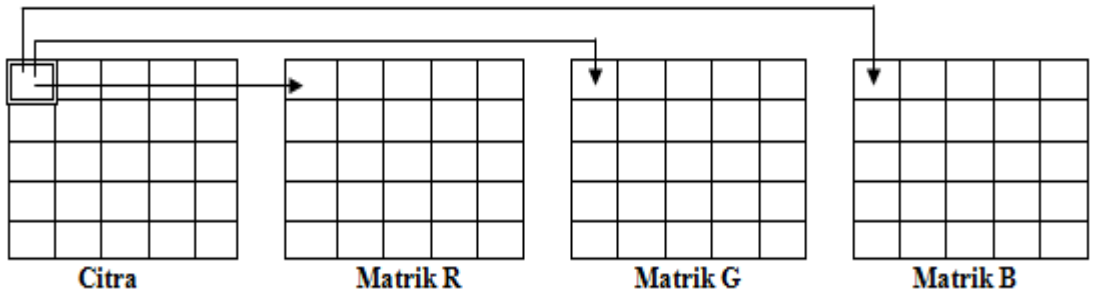
1.3 Analisis Metode Histogram *Euclidean Distance* pada Aplikasi *Image Retrieval*

Analisis metode digunakan untuk mengetahui alur proses dari sebuah metode yang digunakan dapat diterapkan ke dalam aplikasi yang dibangun. Pembangunan aplikasi *image retrieval* ini menggunakan metode *Histogram Euclidean Distance* membandingkan warna pada citra dan menghitung jarak antar histogram warnanya. Sehingga mendapatkan kondisi dimana jarak antar kedua citra tersebut bernilai 0 atau kecil yang menyatakan tingkat kemiripan antar citra tersebut.

1.3.1 Analisis Matrik RGB (Red, Green, Blue)

Pada tahap ini suatu citra dikonversi menjadi matrik atau citra dalam bentuk piksel-piksel RGB. Setelah citra menjadi piksel-piksel matrik RGB, maka dipisah lagi matrik RGB tadi menjadi 3 *channel* warna menjadi Matrik R, Matrik G, dan matrik B. Penjelasannya adalah sebagai berikut :

Contoh apabila citra berukuran 5 x 5



Gambar 3.3 Konversi Matrik

Nilai tiap piksel yang ada dicitra akan di konversi ke nilai piksel ke matrik R, matrik G, dan matrik B. Jumlah kolom dan baris harus sama dan apabila pada posisi (0,0) pada image maka pada matrik R, G, dan B pun harus pada posisi (0,0).

1.3.2 Analisis Histogram Warna

Histogram warna merupakan distribusi warna dalam sebuah gambar yang didapatkan dengan menghitung jumlah pixel dari setiap bagian range warna, secara tipikal dalam dua atau tiga dimensi. Di dalam histogram ini terdapat *bins* atau intensitas kemunculan nilai warna pada suatu gambar.

Dalam pembuatan histogram, nilai RGB yang punya range dari 0-255 akan memiliki kemungkinan kombinasi warna sebesar 16.777.216 (didapat dari 256 x 256 x 256). Pada proses komputasi, tentu saja proses ini menghabiskan banyak waktu.

Masalah tersebut dapat diatasi dengan kuantisasi warna, yaitu suatu prosedur untuk mengurangi kemungkinan jumlah warna. Dengan cara ini, jumlah warna yang besar tadi bisa dkurangi, sehingga proses yang dibutuhkan akan semakin mudah. Pada penelitian ini digunakan kuantisasi pada nilai-nilai warna seperti yang terlihat pada tabel dibawah ini.

Tabel 3.1 Tabel Kuantisasi

Nilai RGB	Nilai Kuantisasi
0 - 31	0
32 – 63	32
64 - 95	64
96 - 127	96
128 - 159	128
160 – 191	160
192 - 223	192
224 – 255	224

Misalnya terdapat suatu citra berukuran 4x4 piksel yang telah dibentuk menjadi Matrik R, matrik G, dan matrik B seperti terlihat pada gambar 3.7 ini.

(129, 23, 230)	(23, 160, 70)	(50, 250, 189)	(140, 180, 40)
(140, 250, 50)	(23, 170, 99)	(80, 20, 178)	(128, 45, 130)
(25, 230, 68)	(240, 123, 67)	(123, 123, 189)	(25, 180, 189)
(80, 45, 50)	(140, 250, 189)	(129, 45, 99)	(23, 250, 40)

R

129	23	50	140
140	23	80	128
25	240	123	25
80	140	129	23

G

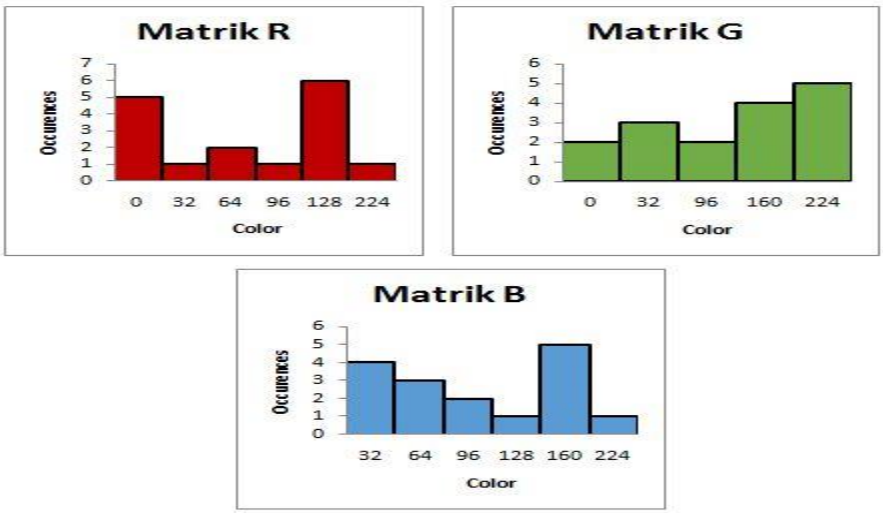
23	160	250	180
250	170	20	45
230	123	123	180
45	250	45	250

B

230	70	189	40
50	99	178	130
68	67	189	189
50	189	99	40

Gambar 3.4 Matrik R,G,B

Tahap selanjutnya adalah membuat histogram warna dari masing-masing *channel* matrik diatas dengan terlebih dahulu di kuantisasi, sehingga menjadi seperti berikut:



Gambar 3.5 Histogram Warna RGB

Pembuatan histogram warna dilakukan pada citra input maupun citra yang berada dalam gallery, sehingga dapat dilakukan penghitungan jarak antar histogram citra input dan citra yang berada dalam *gallery*.

1.3.3 Analisis Euclidean Distance

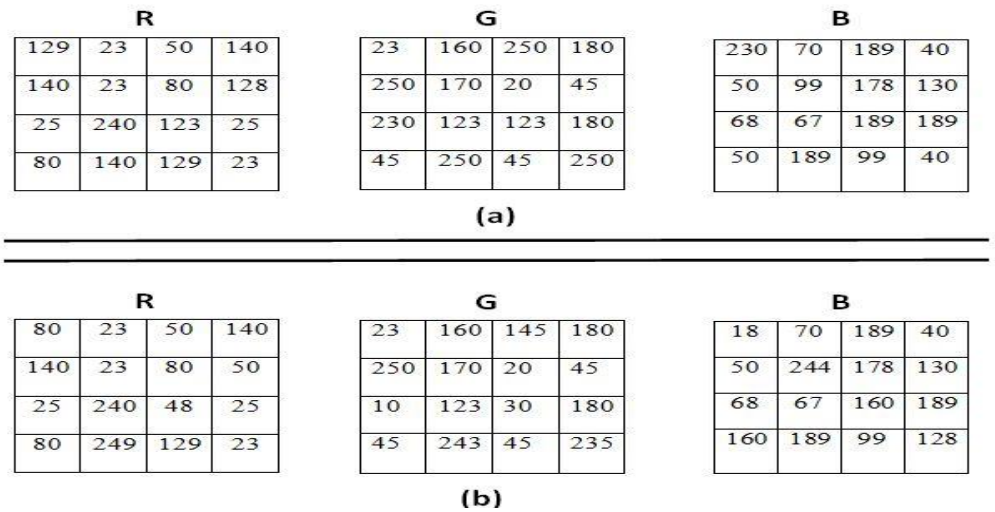
Pada tahapan ini *Euclidean distance* digunakan untuk menghitung jarak antar histogram warna. Dimana tahapan ini dapat menentukan tingkat kemiripan dari citra-citra tersebut.

Secara matematis *Histogram Euclidean Distance* dituliskan sebagai berikut:

$$d^2(h,g) = \sum_A \sum_B \sum_C (h(a,b,c) - g(a,b,c))^2$$

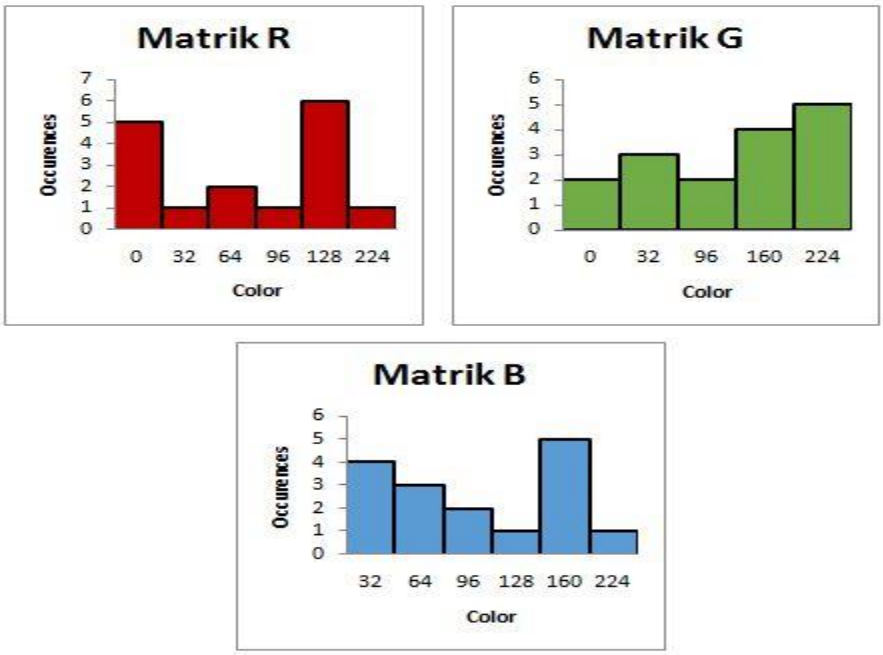
Dimana h dan g mewakili 2 histogram warna, dan a, b, c adalah *channel* warnanya.

Sebelum menghitung jarak antar histogramnya, terlebih dahulu dilakukan normalisasi histogram untuk mengatasi lebarnya histogram yang terbentuk akibat jumlah piksel yang besar dan juga untuk perhitungan citra-citra yang memiliki perbedaan ukuran piksel. misal terdapat histogram warna dari suatu citra input dan citra *gallery* berukuran 4x4 piksel, penjelasannya adalah sebagai berikut:



Gambar 3.6 (a) Matrik RGB Citra Masukan, (b) Matrik RGB Citra gallery

Terlihat pada gambar 3.9 Adalah matrik-matrik RGB dari citra masukan dan juga citra *gallery*, maka tahap selanjutnya adalah buatlah histogram warna dari tiap-tiap matrik tadi dengan terlebih dahulu dikuantisasi sehingga histogramnya terlihat seperti gambar berikut:



Gambar 3.7 Histogram Warna Citra Masukan

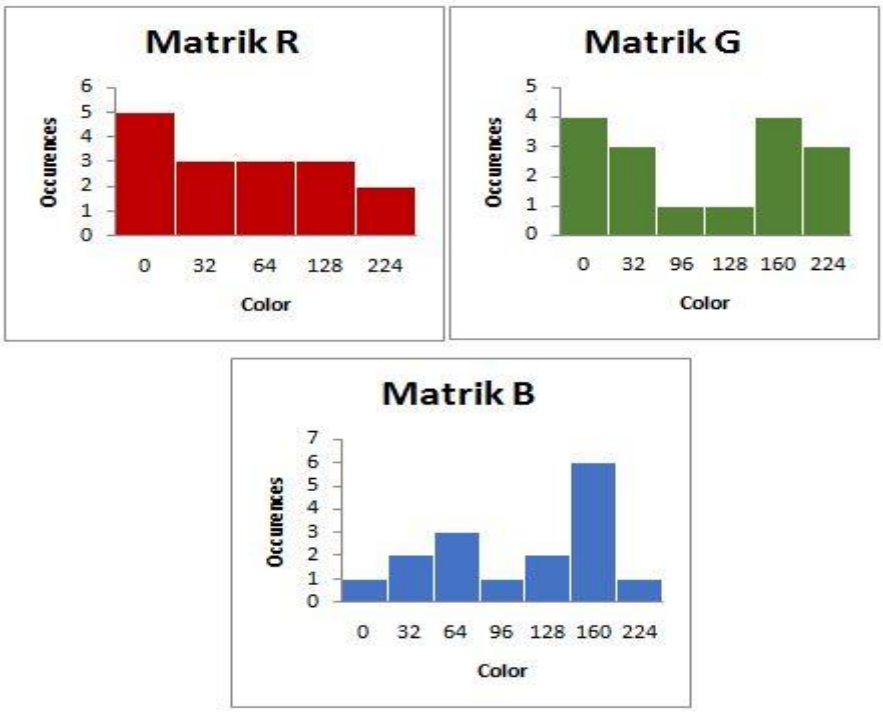
Setelah itu tahap selanjutnya adalah normalisasi histogram seperti terlihat pada tabel berikut ini :

Tabel 3.2 Normalisasi Histogram Citra Masukan

Normalisasi Histogram Matrik R						
Color	0	32	64	96	128	224
Occurences (y)	5	1	2	1	6	1
N	5	6	8	9	15	16
$\frac{y}{N}$	0,3125	0,0625	0.125	0,0625	0,375	0,0625
Normalisasi Histogram Matrik G						
Color	0	32	96	160	224	

Occurences (y)	2	3	2	4	5	
N	2	5	7	11	16	
$\frac{y}{N}$	0,125	0,1875	0,125	0,25	0,3125	
Normalisasi Histogram Matrik B						
Color	32	64	96	128	160	224
Occurences (y)	4	3	2	1	5	1
N	4	7	9	10	15	16
$\frac{y}{N}$	0,25	0,1875	0,125	0,0625	0,3125	0,0625

Pada citra *gallery* juga dilakukan tahap yang sama yaitu terlebih dahulu dibuat histogram warna dari tiap-tiap matriknya dengan dikuantisasi terlebih dahulu sehingga hasilnya tampak seperti gambar berikut ini :



Gambar 3.8 Histogram Warna Citra Gallery

Setelah itu tahap selanjutnya adalah normalisasi histogram seperti terlihat pada tabel 3.3 berikut ini ;

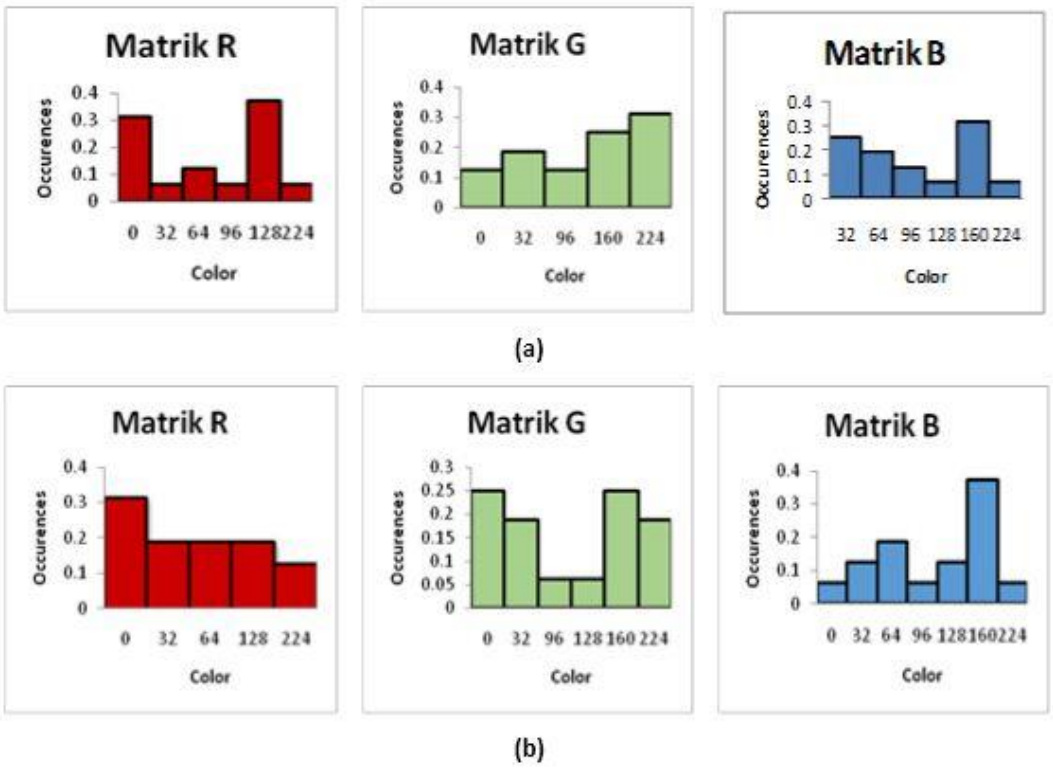
Tabel 3.3 Normalisasi Histogram Citra *Gallery*

Normalisasi Histogram Matrik R					
Color	0	32	64	128	224
Occurences (y)	5	3	3	3	2
N	5	8	11	14	16
$\frac{y}{N}$	0,3125	0,1875	0.1875	0,1875	0,125

Normalisasi Histogram Matrik G						
Color	0	32	96	128	160	224
Occurences (y)	4	3	1	1	4	3
N	4	7	8	9	13	16
$\frac{y}{N}$	0,25	0,1875	0,0625	0,0625	0,25	0,1875

Normalisasi Histogram Matrik B							
Color	0	32	64	96	128	160	224
Occurences (y)	1	2	3	1	2	6	1
N	1	3	6	7	9	15	16
$\frac{y}{N}$	0,0625	0,125	0,1875	0,0625	0,125	0,375	0,0625

Setelah masing-masing histogram dari citra tersebut dinormalisasi maka histogram warnanya masing-masing akan menjadi seperti gambar berikut:



Gambar 3.9 Hasil Normalisasi (a) Histogram Citra Masukan, (b) Histogram Citra gallery

Tahap selanjutnya adalah menghitung jarak antar histogram untuk mengetahui seberapa mirip citra masukan dengan citra *gallery* berdasarkan jarak antar histogramnya. Pada tiap *occurrences* nilai citra input maupun citra gallery dilakukan pengurangan. Seperti yang dijelaskan pada tabel-tabel dibawah ini:

Tabel 3.4 Perhitungan jarak matrik R

Citra *Input*		Citra *Gallery*		Hasil
Nilai	***Occurences***	**Nilai**	***Occurences***	
0	0,3125	0	0.3125	0
32	0,0625	32	0.1875	0,125
64	0,125	64	0.1875	0,0625
96	0,0625	96	0	0,0625
128	0,375	128	0.1875	0,1875
224	0,0625	224	0.125	0,0625
Total				**0,5**

Tabel 3.5 Perhitungan jarak matrik G

Citra *Input*		Citra *Gallery*		Hasil
Nilai	***Occurences***	**Nilai**	***Occurences***	
0	0.125	0	0.25	0,125
32	0.1875	32	0.1875	0
96	0.125	96	0.0625	0,0625
128	0	128	0.0625	0,0625
160	0.25	160	0.25	0
224	0.3125	224	0.1875	0,125
Total				**0,375**

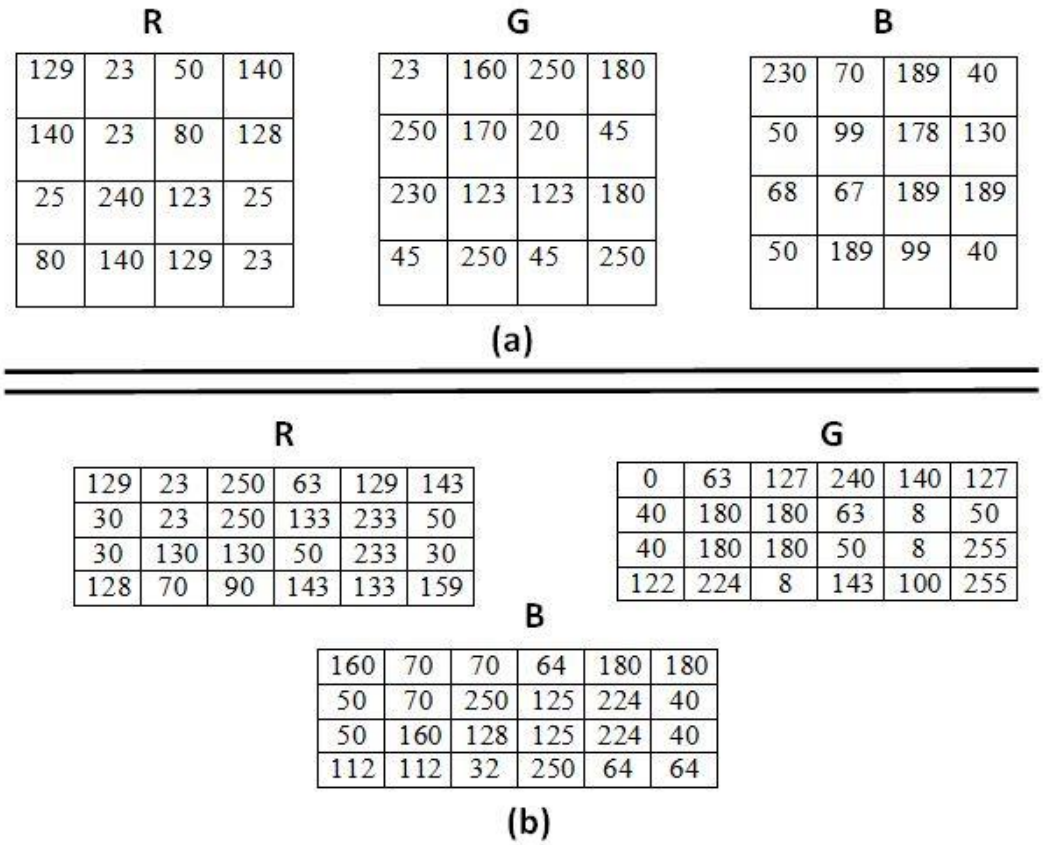
Tabel 3.6 Perhitungan jarak matrik B

Citra *Input*		Citra *Gallery*		Hasil
Nilai	***Occurences***	**Nilai**	***Occurences***	
0	0	0	0.0625	0
32	0.25	32	0.125	0,125
64	0.1875	64	0.1875	0
96	0.125	96	0.0625	0,0625
128	0.0625	128	0.125	0,0625
160	0.3125	160	0.375	0,0625
224	0.0625	224	0.0625	0
Total				**0,3125**

Tabel-tabel diatas adalah hasil perhitungan jarak antara histogram warna citra input dan citra gallery pada channel warna R, G, dan B dimana hasil haruslah bernilai *absolute*.

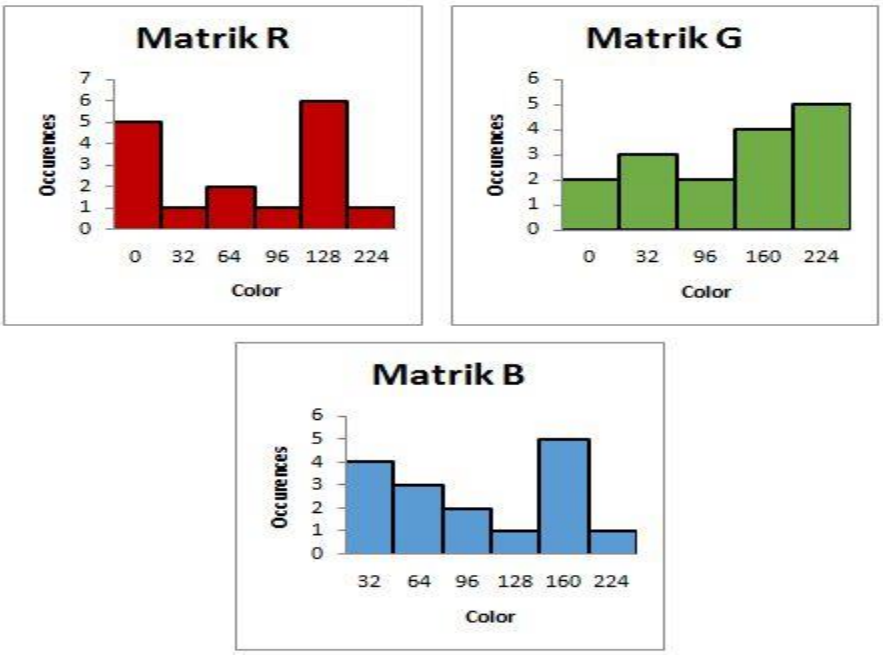
Dari tabel-tabel diatas, diketahui channel R bernilai 0,5, channel G bernilai 0,37 5 dan channel B bernilai 0,3125, maka total jarak *Euclidean Distance* RGB nya adalah 0,5 + 0,375 + 0,3125 = 1,1875. Dimana Semakin kecil jaraknya, maka semakin tinggi pula tingkat kemiripannya.

Sedangkan untuk citra yang berbeda ukuran pikselnya, misal akan membandingkan citra masukan sebelumnya yang berukuran 4x4 piksel dengan citra *gallery* berukuran 6x4 piksel. dijelaskan sebagai berikut :



Gambar 3.10 (a) Matrik RGB Citra Masukan, (b) Matrik RGB citra *gallery*

Pada gambar 3.14 Adalah matrik-matrik RGB dari citra masukan dan juga citra *gallery*, maka tahap selanjutnya adalah buatlah histogram warna dari tiap-tiap matrik tadi dengan terlebih dahulu dikuantisasi sehingga histogramnya terlihat seperti gambar berikut:



Gambar 3.11 Histogram Warna Citra Masukan

Setelah itu tahap selanjutnya adalah normalisasi histogram seperti terlihat pada tabel 3.7 berikut ini :

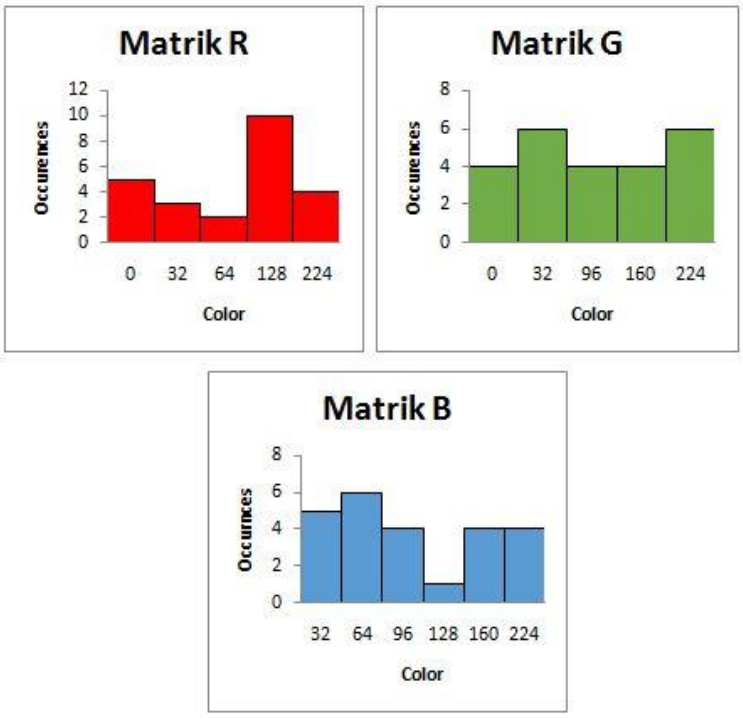
Tabel 3.7 Normalisasi Histogram Citra masukan

Normalisasi Histogram Matrik R						
Color	0	32	64	96	128	224
Occurences (y)	5	1	2	1	6	1
N	5	6	8	9	15	16
$\frac{y}{N}$	0,3125	0,0625	0.125	0,0625	0,375	0,0625
Normalisasi Histogram Matrik G						
Color	0	32	96	160	224	

Occurences (y)	2	3	2	4	5
N	2	5	7	11	16
$\frac{y}{N}$	0,125	0,1875	0,125	0,25	0,3125

Normalisasi Histogram Matrik B						
Color	32	64	96	128	160	224
Occurences (y)	4	3	2	1	5	1
N	4	7	9	10	15	16
$\frac{y}{N}$	0,25	0,1875	0,125	0,0625	0,3125	0,0625

Pada citra *gallery* juga dilakukan tahap yang sama yaitu terlebih dahulu dibuat histogram warna dari tiap-tiap matriknya dengan dikuantisasi terlebih dahulu sehingga hasilnya tampak seperti gambar berikut ini :



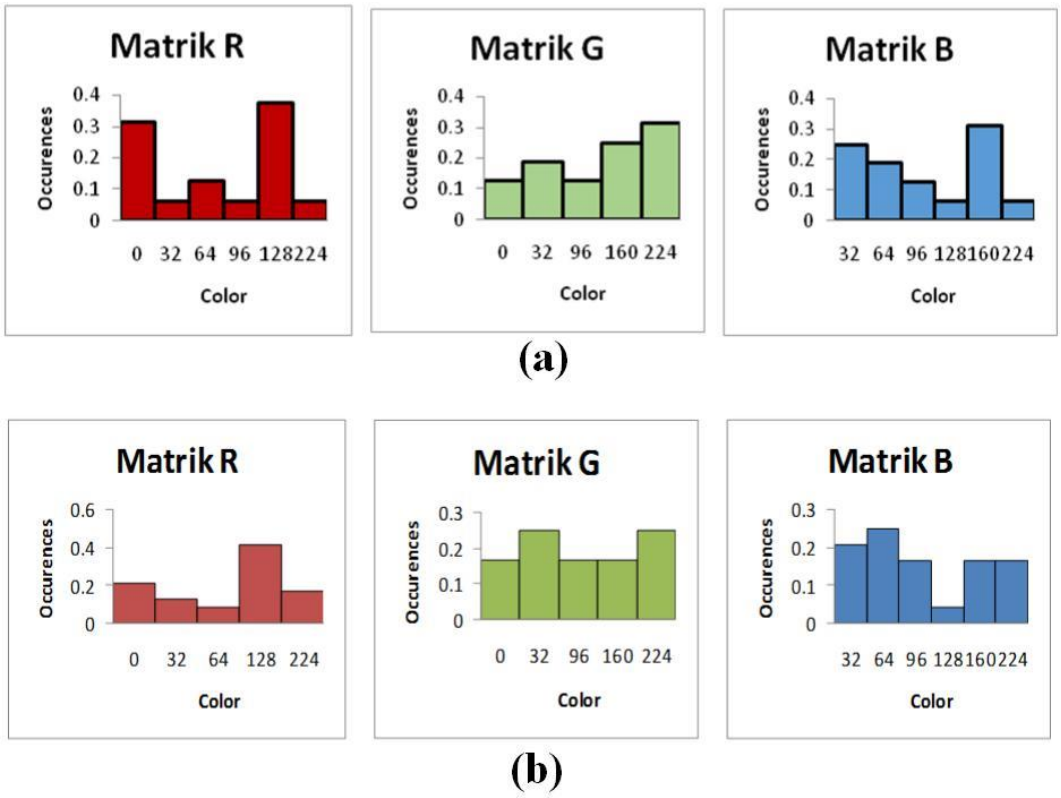
Gambar 3.12 Histogram Warna Citra *Gallery*

Setelah itu tahap selanjutnya adalah normalisasi histogram warna dari citra *gallery* seperti terlihat pada tabel 3.8 berikut ini :

Tabel 3.8 Normalisasi Histogram Warna Citra *Gallery*

Normalisasi Histogram Matrik R					
Color	0	32	64	128	224
Occurences (y)	5	3	2	10	4
N	5	8	10	20	24
$\frac{y}{N}$	0,2083	0,125	0.0833	0,4166	0,1666
Normalisasi Histogram Matrik G					
Color	0	32	96	160	224
Occurences (y)	4	6	4	4	6
N	4	10	14	18	24
$\frac{y}{N}$	0,1666	0,25	0,1666	0,1666	0,25
Normalisasi Histogram Matrik B					
Color	32	64	96	128	160	224
Occurences (y)	5	6	4	1	4	4
N	5	11	15	16	20	24
$\frac{y}{N}$	0,2083	0,25	0,1666	0,0416	0,1666	0,1666

Setelah masing-masing histogram dari citra tersebut dinormalisasi maka histogram warnanya masing-masing akan menjadi seperti gambar berikut:



Gambar 3.13 Hasil Normalisasi (a) Histogram Citra Masukan, (b) Histogram Citra *gallery*

Tahap selanjutnya adalah menghitung jarak antar histogram untuk mengetahui seberapa mirip citra masukan dengan citra *gallery* berdasarkan jarak antar histogramnya. Pada tiap *occurrences* nilai citra input maupun citra gallery dilakukan pengurangan. Seperti yang dijelaskan pada tabel-tabel dibawah ini:

Tabel 3.9 Perhitungan Jarak Matrik R

Citra *Input*		Citra *Gallery*		Hasil
Nilai	***Occurences***	**Nilai**	***Occurences***	
0	0,3125	0	0,2083	0,1042
32	0,0625	32	0,125	0,0625
64	0,125	64	0.0833	0,0417
96	0,0625	96	0	0,0625
128	0,375	128	0,4166	0,0416
224	0,0625	224	0,1666	0,1041
Total				**0,4166**

Tabel 3.10 Perhitungan Jarak Matrik G

Citra *Input*		Citra *Gallery*		Hasil
Nilai	***Occurences***	**Nilai**	***Occurences***	
0	0.125	0	0,1666	0,0416
32	0.1875	32	0.25	0,0625
96	0.125	96	0,1666	0,0416
160	0.25	160	0,1666	0,0834
224	0.3125	224	0.25	0,0625
Total				**0,2916**

Tabel 3.11 Perhitungan Jarak Matrik B

Citra *Input*		Citra *Gallery*		Hasil
Nilai	***Occurences***	**Nilai**	***Occurences***	
32	0.25	32	0,2083	0,0417
64	0.1875	64	0.25	0,0625
96	0.125	96	0,1666	0,0416
128	0.0625	128	0,0416	0,0209
160	0.3125	160	0,1666	0,1459
224	0.0625	224	0,1666	0,1041
Total				**0,4167**

Tabel-tabel diatas adalah hasil perhitungan jarak antara histogram warna citra input dan citra gallery yang berbeda ukuran pada channel warna R, G, dan B dimana hasil haruslah bernilai *absolute*.

Dari tabel-tabel diatas, diketahui channel R bernilai 0,4166, channel G bernilai 0,2916 dan channel B bernilai 0,4167, maka total jarak *Euclidean Distance* RGB nya adalah 0,4166 + 0,2916 + 0,4167 = 1,1249. Hasil ini lebih kecil dibandingkan dengan hasil perbandingan pertama yang bernilai 1.1875. maka pada citra *gallery* kedua ini dianggap lebih mirip dengan citra masukannya.

1.3.4 Analisis Kebutuhan Non-Fungsional

Analisis kebutuhan non fungsional merupakan analisis yang dibutuhkan untuk menentukan spesifikasi kebutuhan yang di perlukan oleh Aplikasi. Kebutuhan non fungsional terdiri dari analisis perangkat keras, analisis perangkat lunak dan analisis pengguna (*user*). Analisis non fungsional bertujuan agar aplikasi yang dibangun dapat digunakan sesuai dengan kebutuhan pengguna aplikasi dalam mencari inforamsi yang dibutuhkan.

1.3.4.1 Analisis Kebutuhan Perangkat Lunak

Perangkat lunak digunakan dalam sebuah sistem merupakan perintah-perintah yang diberikan kepada perangkat keras agar bisa dapat berinteraksi diantara keduanya. Adapun perangkat lunak yang dibutuhkan dalam membangun aplikasi *image retrieval* berbasis *Desktop* ini adalah sebagai berikut :

1. Sistem Operasi Windows 8.1
2. IDE Netbeans 8
3. JDK 8

Perangkat lunak yang digunakan oleh *user* untuk menjalankan aplikasi *image retrieval* berbasis desktop adalah sistem operasi windows (Windows XP, Vista, 7,8).

1.3.4.2 Analisis Kebutuhan Perangkat Keras

Komputer terdiri dari perangkat keras dan perangkat lunak yang saling berinteraksi. Perangkat lunak memberikan instruksi-instruksi kepada perangkat keras untuk melakukan suatu tugas tertentu, sehingga dapat menjalankan suatu sistem di dalamnya.

Pada aplikasi *image retrieval* Berbasis *Desktop* ini, perangkat yang digunakan adalah sebagai berikut :

1. Spesifikasi *minimum* perangkat keras yang dibutuhkan oleh pengembang.
   a. Prosesor dengan kecepatan 1.8 Ghz.
   b. RAM 2 Gb.

c. Hardisk 20 Gb.

d. VGA *Card* 256 Mb.

e. Monitor.

f. *Mouse* dan *Keyboard*.

2. Spesifikasi *minimum* perangkat keras yang dibutuhkan oleh pengguna.

a. Prosesor dengan kecepatan 1.8 Ghz ke atas.

b. RAM 2 Gb.

c. Hardisk 500 Mb.

d. VGA *Card* 256 Mb.

e. Monitor.

f. *Mouse* dan *Keyboard*.

Maka dengan spesifikasi tersebut aplikasi dapat dijalankan dengan baik dan tidak akan terjadi masalah yang dapat menghambat berjalannya aplikasi.

1.3.4.3 Analisis Pengguna Aplikasi (*User*)

Analisis pengguna aplikasi ditujukan untuk seluruh masyarakat atau seluruh orang yang ingin mendapatkan kemudahan dalam melakukan pencarian gambar pada *gallery* mereka. Karakteristik *user* untuk menggunakan aplikasi ini adalah:

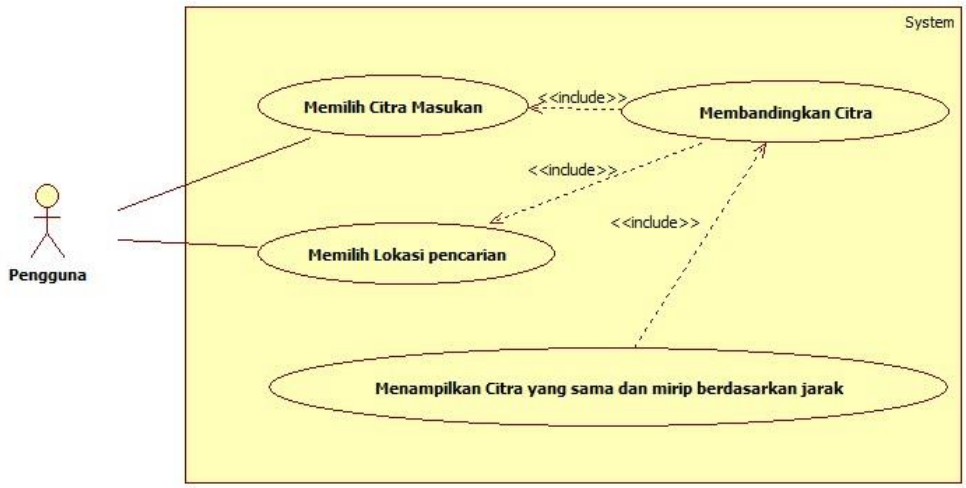
1. Minimal dapat menjalankan aplikasi (mengoperasikan komputer)
2. Usia 12 tahun keatas.

1.3.5 Analisis Kebutuhan Fungsional

Analisis kebutuhan fungsional menggambarkan proses kegiatan yang akan diterapkan dalam sebuah sistem dan menjelaskan kebutuhan yang diperlukan sistem agar sistem dapat berjalan dengan baik sesuai kebutuhan. Pada bagian ini akan dibahas tentang *use case diagram*, *activity diagram*, *class diagram*, dan *sequence diagram*.

1.3.5.1 Use case diagram

Pada sub bab berikut ini dijelaskan bagaimana *UseCase* yang terjadi dalam pembangunan aplikasi ini yang dimodelkan dalam sekumpulan *UseCase* dan *Actor* dan bagaimana hubungan-hubungannya yang akan dipetakan dalam diagram *UseCase*, dilengkapi dengan skenario untuk menjelaskan dari gambaran *UseCase* yang ada dengan menggambarkan skenario per *UseCase*. *UseCase diagram* dapat dilihat pada Gambar 3.17 berikut ini.



Gambar 3.14 *UseCase* aplikasi *Image Retrieval*

1. ***Actor Definition***

   Actor Definition berfungsi untuk menjelaskan *Actor* yang terdapat pada *Use case diagram*. *Actor Definition* diterangkan pada Tabel 3.4 di bawah ini.

Tabel 3.12 Actor Definition

No	*Actor*	Deskripsi
1	Pengguna	Orang yang menggunakan Aplikasi

2. *Use Case Definition*

Use Case Definition berfungsi untuk menjelaskan fungsi *Use Case* yang terdapat pada *Use case diagram. Use Case Definition* diterangkan pada tabel 3.13 di bawah ini.

Tabel 3.13 Use Case Definition

No	*Use Case*	Deskripsi
1	Memilih citra masukan	Fungsionalitas untuk memasukkan citra sebagai citra masukan
2	Memilih Lokasi Pencarian	Fungsionalitas untuk memilih lokasi pencarian citra pada *gallery*
3	Membandingkan Citra	Proses membandingkan citra masukan dan citra dalam *gallery*
4	Menampilkan citra yang sama dan mirip berdasarkan jarak	Fungsionalitas untuk melihat hasil perbandingan citra.

3. *Use Case* Skenario

Use Case Skenario mendeskripsikan urutan langkah-langkah dalam proses bisnis baik yang dilakukan aktor terhadap sistem maupun yang dilikakukan oleh sistem terhadap aktor. Berdasarkan *use case diagram* pada Gambar 3.17 diatas maka *use case Skenario* untuk aplikasi yang dibangun akan dijelaskan sebagai berikut.

1. ***Use Case* Skenario memilih citra masukan**

   Use Case Skenario dari *Use Case* memilih citra masukan dijelaskan pada tabel di bawah ini.

Tabel 3.14 Use Case Skenario memilih citra masukan

Identifikasi	
Nomor	1
Nama	Memilih citra masukan
Tujuan	Proses untuk memasukkan citra sebagai citra *query*
Aktor	Pengguna
Utama	
Kondisi Awal	Pengguna berada di menu utama
Aksi Aktor	**Reaksi Sistem**
1. Memilih tombol *input* citra	
	2. Sistem menampilkan halaman *gallery* penyimpanan citra
3. Memilih 1 citra sebagai citra masukan	
Kondisi akhir	Sistem menampilkan citra yang dipilih pengguna

2. *Use Case* Skenario Memilih lokasi pencarian

Use Case Skenario dari *Use Case* Memilih lokasi pencarian dijelaskan pada tabel di bawah ini.

Tabel 3.15 Use Case Memilih lokasi pencarian

Identifikasi	
Nomor	2
Nama	Memilih Lokasi Pencarian
Tujuan	Menetapkan Lokasi Pencarian pada *Explorer*
Aktor	Pengguna
Utama	
Kondisi Awal	Pengguna berada di halaman utama
Aksi Aktor	**Reaksi Sistem**
1. Memilih tombol *browse*	
	2. sistem menampilkan halaman *explorer*
3. menampilkan lokasi pencarian	
	3. sistem melakukan proses perbandingan

***3. Use Case* Skenario Membandingkan Citra**

Use Case Skenario dari *Use Case* Membandingkan Citra dijelaskan pada tabel di bawah ini.

Tabel 3.16 Use Case Membandingkan Citra

Identifikasi	
Nomor	2
Nama	Membandingkan Citra
Tujuan	Proses membandingkan citra masukan dengan citra dalam *gallery*

Aktor	Pengguna
Utama	
Kondisi Awal	Menu utama dengan kondisi citra masukan dan lokasi pencarian sudah dipilih
Aksi Aktor	**Reaksi Sistem**
	1. sistem melakukan proses perbandingan citra masukan dan citra yang terdapat pada *gallery*

4. *Use Case* Skenario Menampilkan citra yang sama dan mirip berdasarkan jarak

Use Case Skenario dari *Use Case* Menampilkan citra yang sama dan mirip berdasarkan jarak dijelaskan pada tabel di bawah ini.

Tabel 3.17 Use Case Menampilkan Citra yang sama dan mirip berdasarkan jarak

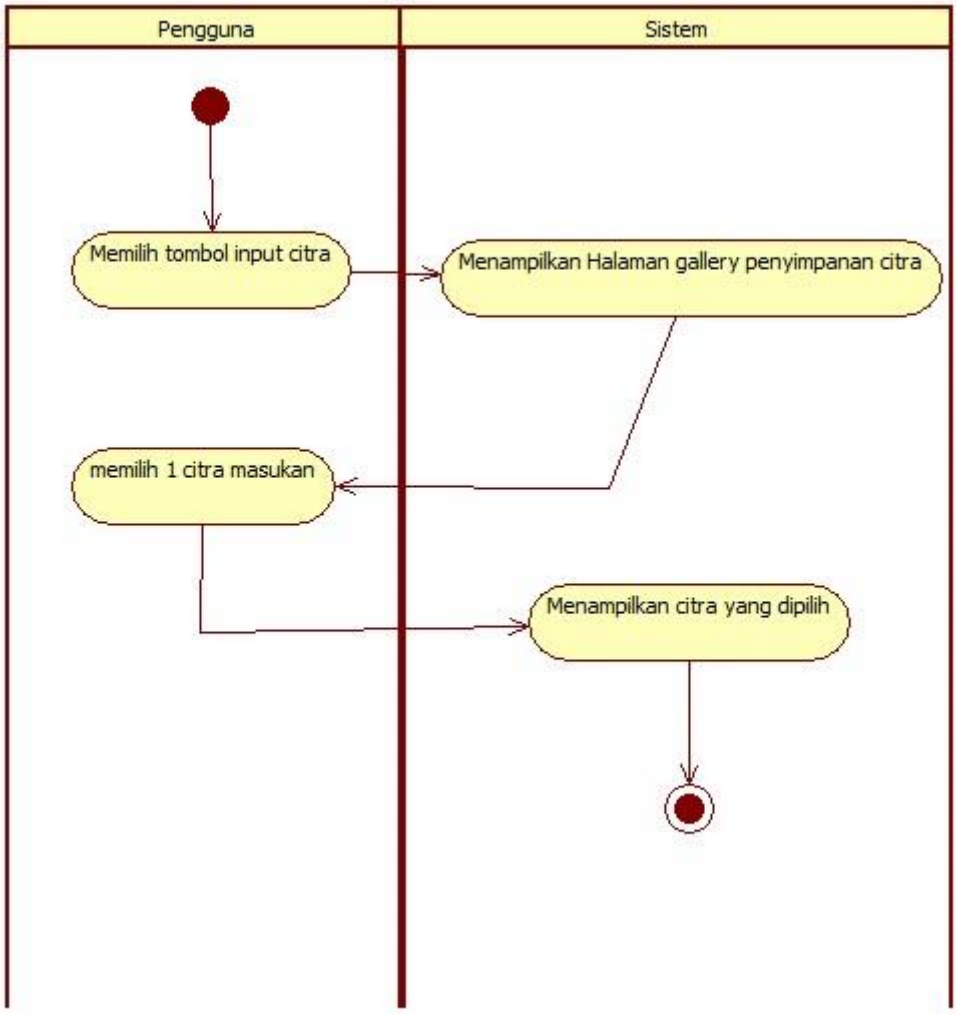
Identifikasi	
Nomor	3
Nama	Menampilkan citra yang sama dan mirip berdasarkan jarak
Tujuan	Melihat hasil pencarian citra
Aktor	Pengguna
Utama	
Kondisi Awal	Pengguna berada di halaman utama
Aksi Aktor	**Reaksi Sistem**
1. Memilih tombol browse	2. sistem menampilkan citra-citra yang sama dan dianggap mirip

1.3.5.2 Activity Diagram

Activity diagram menggambarkan berbagai alir aktivitas dalam sistem yang sedang dirancang, bagaimana masing-masing alir berawal, *decision* yang mungkin terjadi, dan bagaimana mereka berakhir. *Activity diagram* juga dapat menggambarkan proses paralel yang mungkin terjadi pada beberapa eksekusi. Berikut ini *activity diagram* pada aplikasi ini.

1. *Activity Diagram* Memilih Citra Masukan.

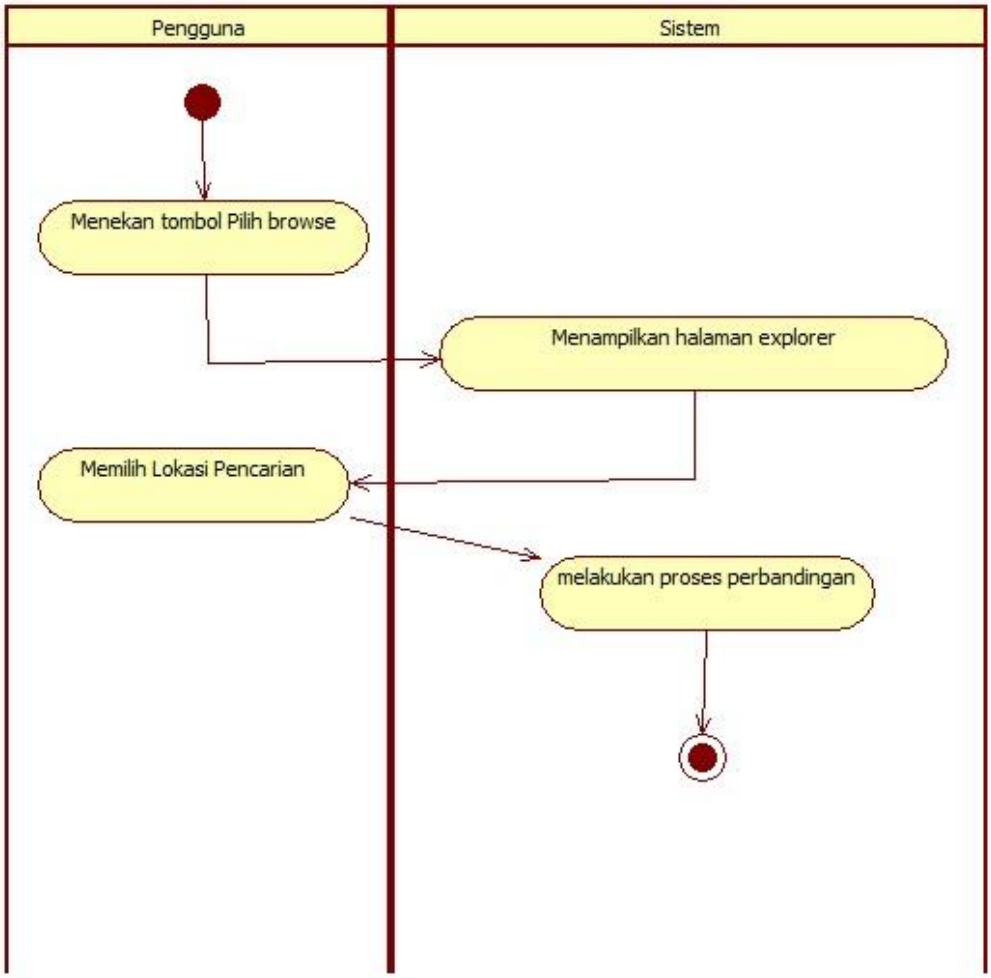
Berikut ini adalah diagram yang menunjukan alur aksi pada aktivitas memilih citra masukan yang dapat dilihat pada gambar 3.18 berikut ini.



Gambar 3.15 *Activity Diagram* Memilih citra masukan

2. *Activity Diagram* Memilih Lokasi Pencarian

Berikut ini adalah diagram yang menunjukan alur aksi pada aktivitas Memilih Lokasi Pencarian yang dapat dilihat pada gambar 3.19 berikut ini.



Gambar 3.16 Activity Diagram Membandingkan Citra

3. *Activity Diagram* Membandingkan Citra

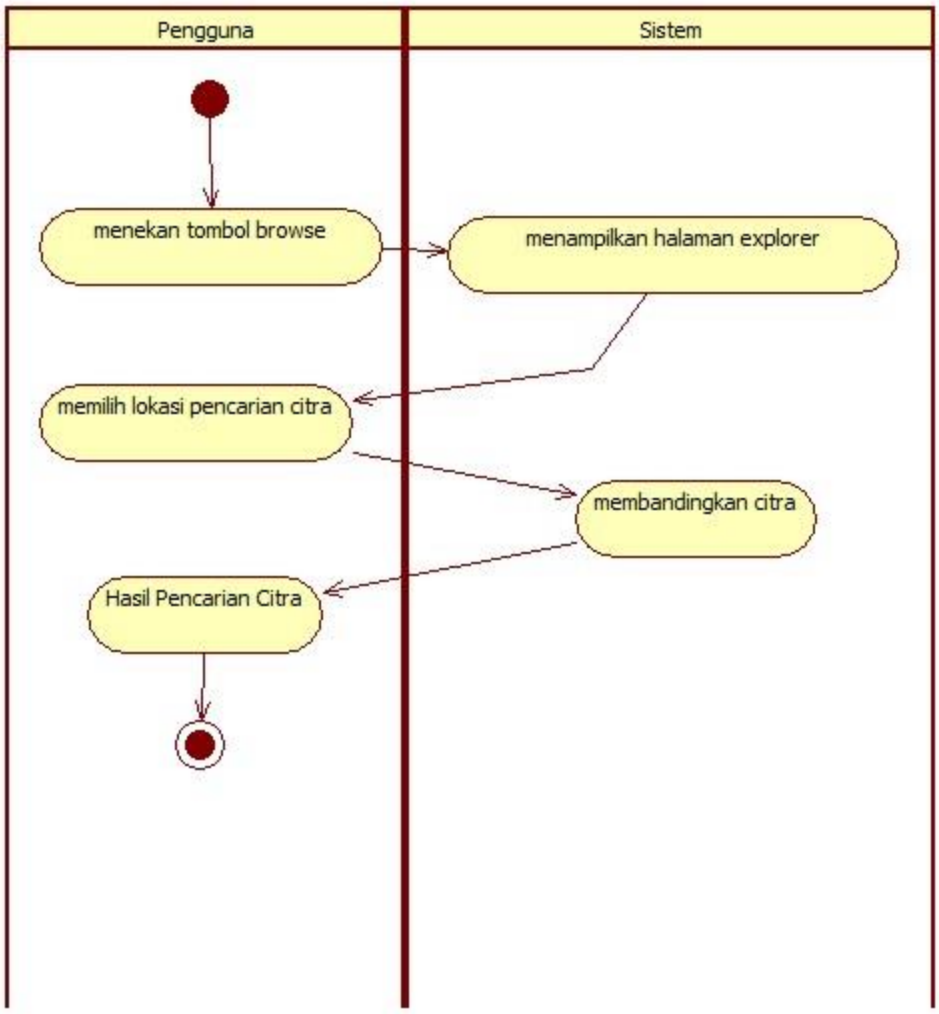
Berikut ini adalah diagram yang menunjukan alur aksi pada aktivitas Membandingkan Citra yang dapat dilihat pada gambar 3.20 berikut ini.



Gambar 3.17 Activity Diagram Melihat citra yang sama dan menyerupai

4. *Activity Diagram* Menampilkan citra yang sama dan mirip berdasarkan jarak

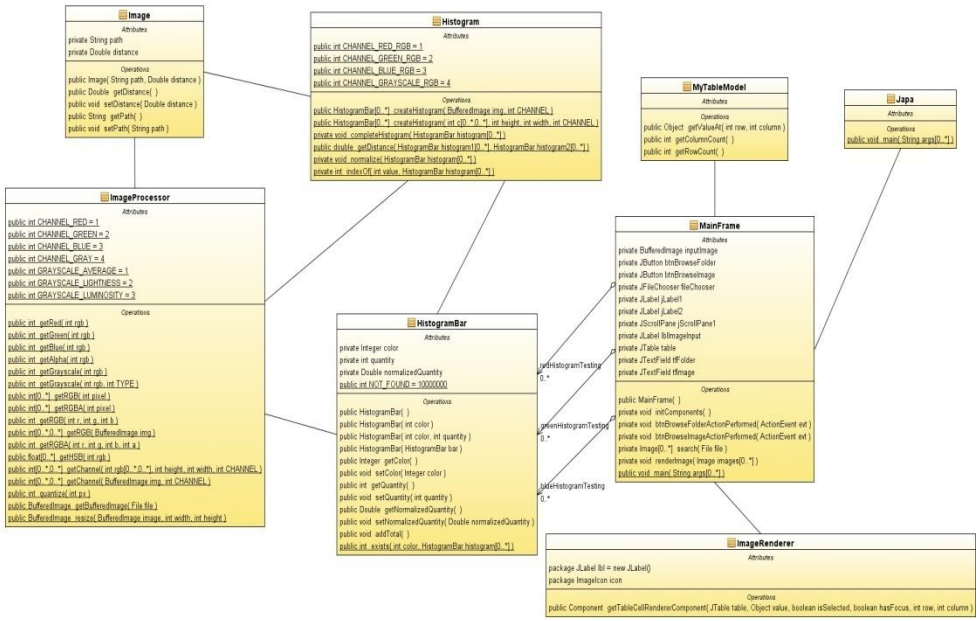
Berikut ini adalah diagram yang menunjukan alur aksi pada aktivitas Menampilkan citra yang sama dan mirip berdasarkan jarak yang dapat dilihat pada gambar 3.21 berikut ini.



Gambar 3.18 *Activity Diagram* Memilih lokasi pencarian

1.3.5.3 Class Diagram

Class diagram menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang dibuat untuk membangun sistem. Perancangan struktur sistem yang terdapat pada aplikasi ini dapat dilihat pada Gambar 3.22.



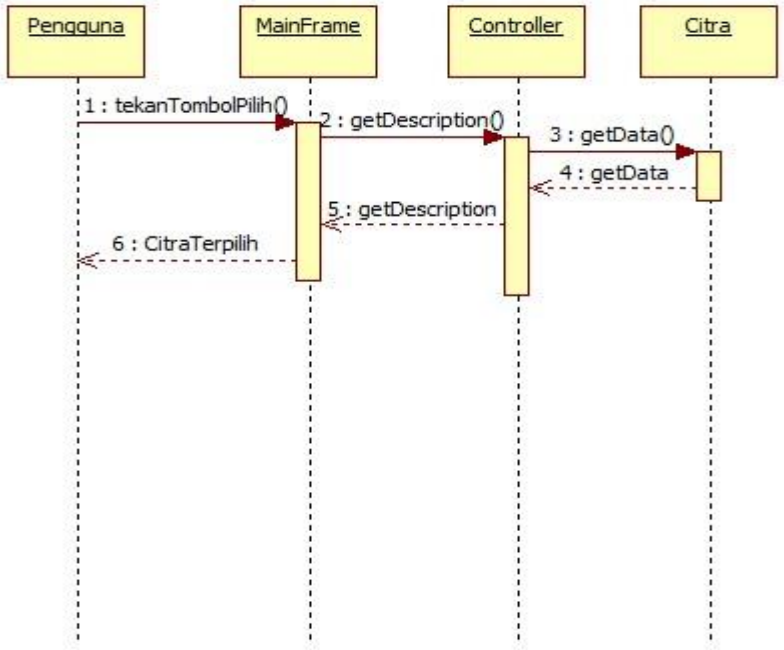
Gambar 3.19 *class diagram*

1.3.5.4 Sequence Diagram

Sequence Diagram menggambarkan interaksi antar masing-masing objek pada setiap use case dalam urutan waktu. Interaksi ini berupa pengiriman serangkaian data antar objek-objek yang saling berinteraksi. Sequence Diagram pada aplikasi ini terdiri dari *sequence diagram* memilih citra masukan , *sequence diagram* memilih lokasi pencarian, *sequence diagram* membandingkan citra, dan *sequence diagram* Menampilkan citra yang sama dan mirip berdasarkan jarak. Berikut ini penjelasan dari masing-masing *sequence diagram*:

1. *Sequence Diagram* Memilih citra masukan

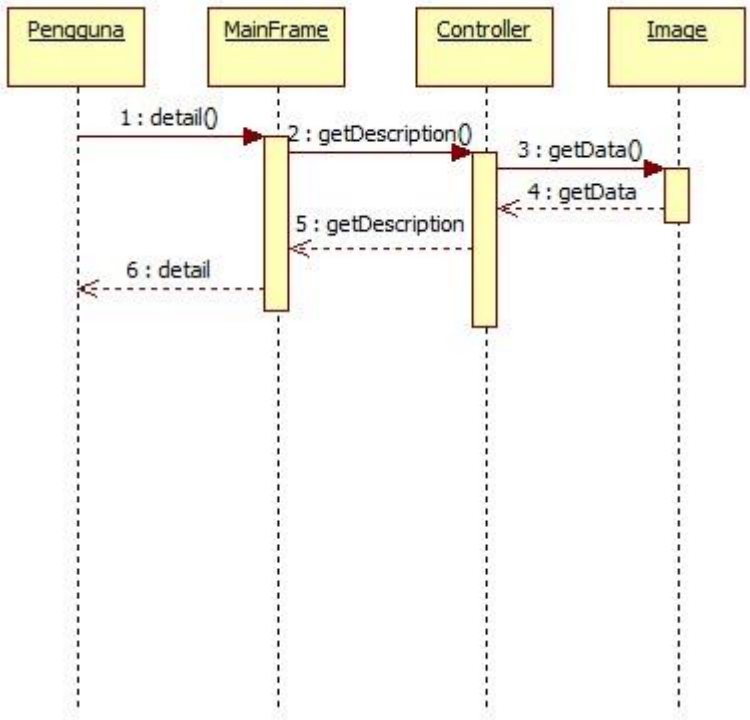
Sequence diagram Menampilkan *level* 1 pada *game* Perjuangan si Pitung ini dapat dilihat pada Gambar 3.23.



Gambar 3.20 Sequence Diagram Memilih citra masukan

2. *sequence diagram* memilih lokasi pencarian

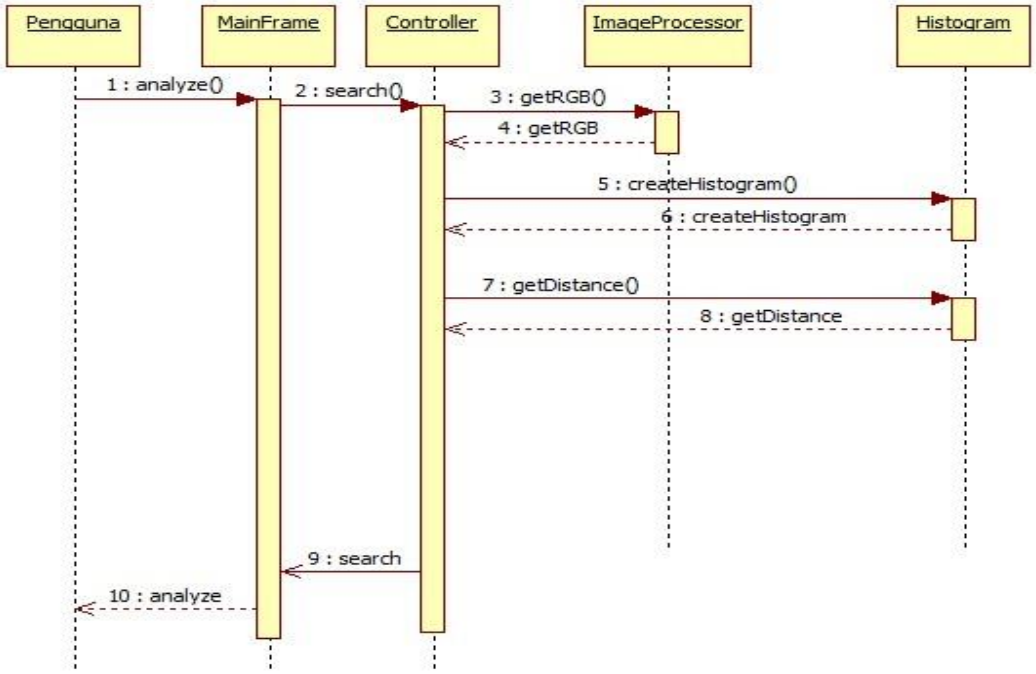
Sequence diagram memilih lokasi pencarian pada aplikasi ini dapat dilihat pada Gambar 3.24.



Gambar 3.21 sequence diagram memilih lokasi pencarian

3. Sequence Diagram Membandingkan Citra

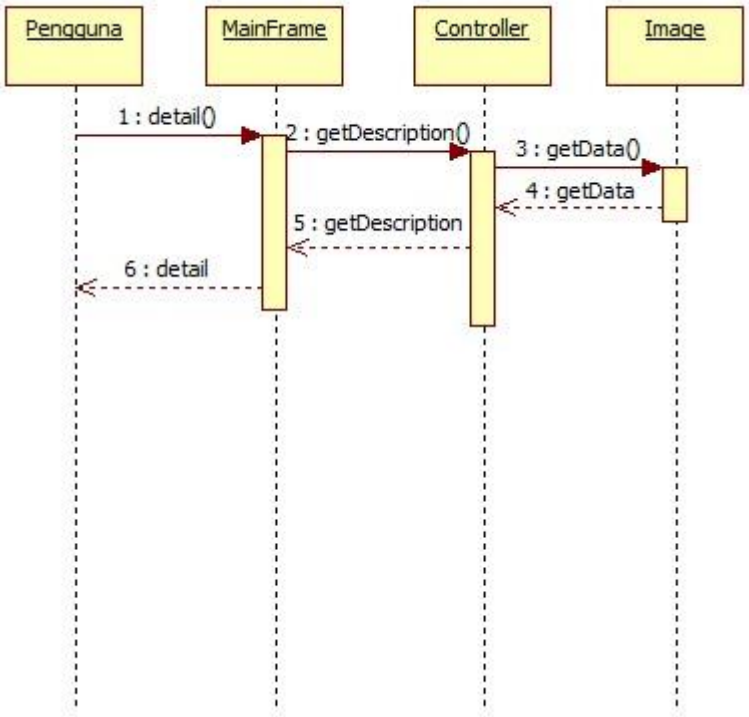
Sequence diagram membandingkan citra pada aplikasi ini dapat dilihat pada Gambar 3.25.



Gambar 3.22 Sequence Diagram Membandingkan Citra

4. *sequence diagram* menampilkan citra yang sama dan mirip berdasarkan jarak

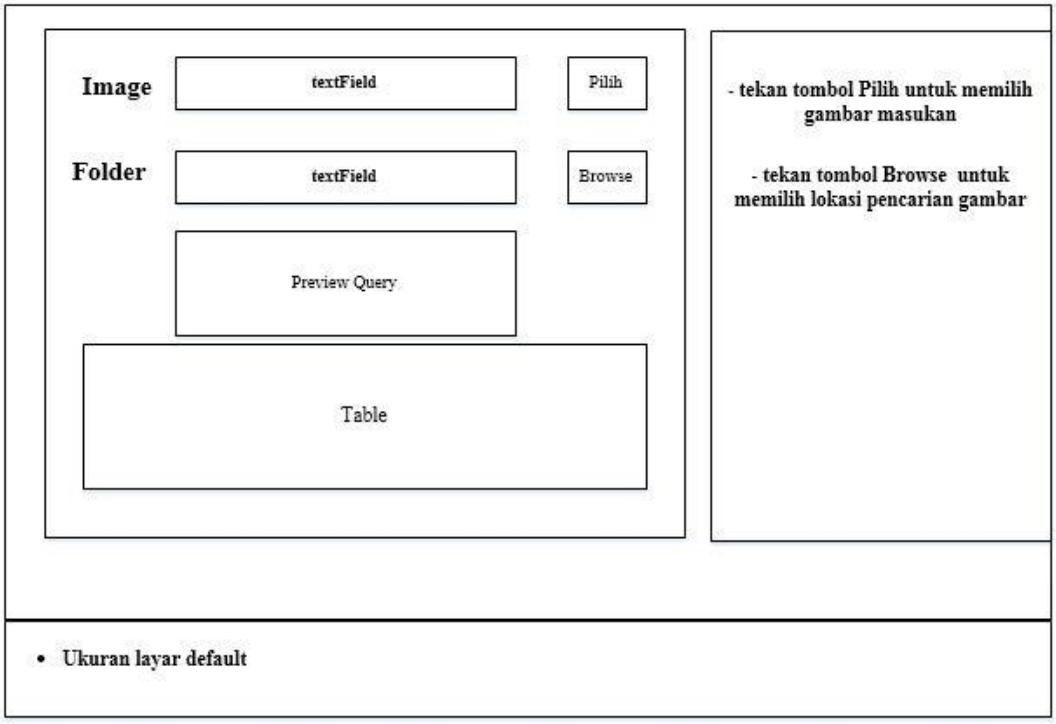
Sequence diagram menampilkan citra yang sama dan mirip berdasarkan jarak pada aplikasi ini dapat dilihat pada Gambar 3.26.



Gambar 3.23 sequence diagram menampilkan citra yang sama dan menyerupai

1.3.6 Perancangan Antarmuka

Perancangan antarmuka bertujuan untuk memberikan gambaran tentang aplikasi yang akan dibangun, sehingga akan mempermudah dalam mengimplementasikan aplikasi serta akan memudahkan pembuatan aplikasi. Adapun perancangan antarmuka perangkat lunak *image retrieval* ini adalah sebagai berikut :



Gambar 3.24 Antarmuka Aplikasi