BAB 3

ANALISIS DAN PERANCANGAN SISTEM

3.1 Analisis Sistem

Analisis sistem merupakan penguraian dari suatu sistem informasi yang utuh ke dalam bagian-bagian komponennya untuk mengidentifikasikan masalah-masalah dan hambatan-hambatan, sehingga dapat diusulkan kebutuhan-kebutuhan untuk memperbaikinya. Tahap analisis sistem adalah tahapan dasar sebelum perancangan untuk memperbaiki sistem yang lama. Dari hasil analisis tersebut dapat dirancang atau diperbaiki menjadi sebuah sistem yang lebih efektif dan efisien.

3.1.1 Analisis Masalah

Setelah melakukan penelitian dan wawancara dengan Bapak Soffi selaku pengurus distro terhadap sistem yang sedang berjalan pada Distro Heaven Skateboards, terdapat beberapa masalah diantaranya :

1. Adanya tindak kejahatan yang sering terjadi pada bagian aksesories. Pelanggan yang memasuki distro tidak ketahuan mencuri topi, kacamata, dompet maupun berbagai macam aksesories yang ada pada distro ini.
2. Aksesories yang tidak jadi dibeli menjadi rusak maupun kotor, yang mengakibatkan turunnya minat pelanggan yang lain untuk mencoba memakai aksesories tersebut.
3. Belum adanya fasilitas mencoba aksesories secara *online.*

Dari permasalahan di atas, maka dibutuhkan Pembangunan *Virtual Accessories* Room pada e-commerce di Distro Heaven Skateboards Bandung yang akan dikelola oleh admin pada distro ini. Aplikasi yang dibuat ini menggunakan Haar Cascade Classifier sebagai algoritmanya.

3.1.2 Analisis Algoritma Haar Cascade Classifier

Aplikasi virtual aksesories ini menggunakan *library Marilena.* Pada *library* ini menggunakan algoritma, yaitu algoritma Haar Cascade Classifier . Tahap awal yang harus ditentukan adalah *face detector* atau deteksi wajah yang akan digunakan, sumber input videonya, dan objek visual apa saja yang akan diload. Semua hal tersebut tergambar di proses inisialisasi yang merupakan proses penentuan awal dari semua hal yang diperlukan untuk menjalankan proses selanjutnya yaitu *face detector*, objek visual, dan kamera.

Pada bagian inisialisasi ini, objek visual juga diinisialisasi terlebih dahulu karena loading objek visual memerlukan waktu yang cukup lama. Objek visual yang akan ditampilkan diload terlebih dahulu. Agar perangkat lunak dapat menampilkan objek visual tertentu tanpa mengubah atau membangun ulang aplikasi, diperlukan sebuah konfigurasi untuk menentukan objek visual yang akan diload. Contoh konfigurasi yang diperlukan dalam apliaksi virtual aksesories itu berisi informasi direktori model aksesories yang digunakan, nama file gambarnya, dan skala yang cocok sehingga penampilan objek bisa lebih proporsional. Barulah setelah itu objek dapat diload.

Kamera dan *face detector* juga merupakan bagian penting pada perangkat lunak ini. *Face detector* akan mengatur hal-hal yang berkaitan dengan

pendeteksian. Dalam sistem ini yang berperan sebagai *face detector* adalah *library* Marilena. Sedangkan kamera mulai menginisialisai video masukan dari kamera sampai pada tahap pemprosesan gambar. Pada perangkat lunak ini hal - hal yang perlu di atur dari *face detector* adalah skala yang ingin digunakan dan koordinat posisi untuk objek visual. Dan untuk kamera akan diatur *frame per rate*, lebar, dan tinggi *stage*.

Pada penelitian ini digunakan metode Haar yang menggunakan algoritma Haar Cascade Classifier . Dengan menggunakan metode ini proses pelacakan untuk berbagai posisi wajah yang berbeda dari sampel yang digunakan tingkat keberhasilannya mencapai 70%. Hal ini terjadi karena sistem pelacakan wajah pada Haar dikhususkan untuk pelacakan wajah dengan posisi lurus ke depan terhadap kamera *(frontal face),* sehingga tidak dapat melacak wajah yang bukan pada posisi tersebut. Jarak objek dari kamera mempengaruhi ukuran citra hasil pelacakan semakin jauh dari kamera semakin kecil citra hasil pelacakan. Adapun tiga tahapan pada proses deteksi wajah yang dijelaskan pada halaman berikutnya.

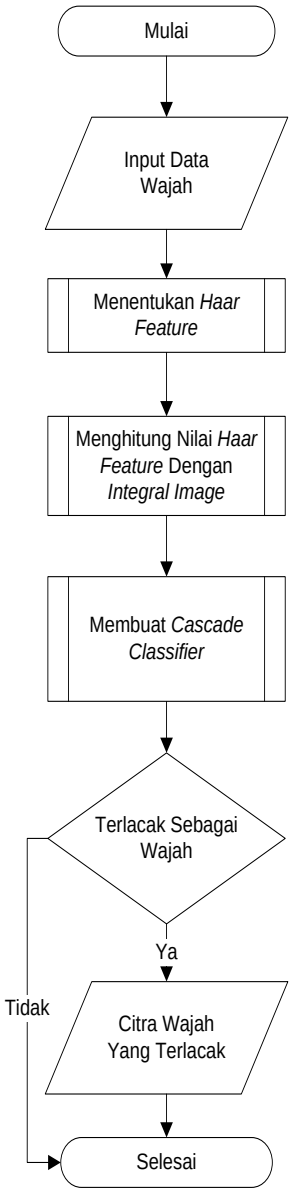
1. Penskalaan ukuran gambar

   Scaling atau penskalaan merupakan proses mengubah ukuran gambar digital menjadi lebih besar ataupun lebih kecil. Untuk sistem akan memanfaatkan *package display object* yang merupakan *package class* milik Flash. *Scaling* dapat dilakukan dengan memanipulasi ukuran tampilan objek dalam dua cara, menggunakan salah satu properti (*width and height*) atau properti method skala yaitu *scalex* dan *scaley*. Properti skala mewakili ukuran relatif dari tampilan objek dibandingkan dengan ukuran aslinya.

Pada awalnya harus ditentukan dahulu faktor skala untuk membuat layar menampilkan dan memproses gambar dari kamera sesuai dengan ukuran yang diinginkan. Tahap *Scaling* ini membutuhkan skala yang tepat, karena tahap scaling dilakukan pada beberapa proses dan semua nilai skala harus konsisten. Skala yang tepat dan umum digunakan adalah empat karena menyesuaikan dengan panjang dan lebar layar yang umum digunakan yaitu 500x450, akan tetapi bisa bersifat dinamis dengan mengubah nilai dari variabel faktor skala. Jika faktor skala terlalu besar tidak hanya mempengaruhi terhadap tampilan tetapi juga proses akan menjadi lebih lama. Jika faktor skala terlalu kecil maka penampilan *stage* di flash akan tidak jelas.

2. Cara kerja Haar Cascade Classifier

   Alur cara kerja algoritma Haar Cascade Classifier untuk mendeteksi wajah dapat dilihat pada gambar di halaman berikutnya.



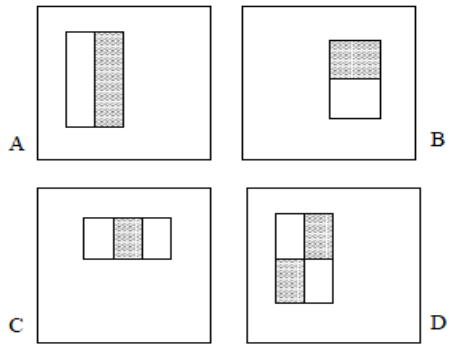
Gambar 3. 1 Alur Cara Kerja Haar Cascade Classifier

Berikut ini tahapan-tahapan cara kerja algoritma Haar Cascade Classifier diantaranya, yaitu :

a. Menentukan Haar *Feature*

Setelah melakukan penskalaan dan mengubah gambar menjadi *grayscale* sebelumnya kemudian dilakukan pencarian posisi wajah atau

mata yaitu dengan cara mencari fitur-fitur yang memiliki tingkat pembeda yang tinggi. Hal ini dilakukan dengan mengevaluasi setiap fitur terhadap data latih dengan menggunakan nilai dari fitur tersebut. Fitur yang memiliki batas terbesar antara wajah dan bukan wajah dianggap sebagai fitur terbaik.



Gambar 3. 2 Haar *Features*

Pada sistem ini fitur yang digunakanhanya dua jenis fitur yaitu fitur (b) dan fitur (c). Setelah melakukan penskalaan sebelumnya kemudian dilakukan pencarian posisi wajah yaitu dengan cara mencari fitur-fitur yang memiliki tingkat pembeda yang tinggi. Hal ini dilakukan dengan mengevaluasi setiap fitur terhadap data latih dengan menggunakan nilai dari fitur tersebut. Fitur yang memiliki batas terbesar antara wajah dan bukan wajah dianggap sebagai fitur terbaik.



Gambar 3. 3 Pencarian dengan Haar *Features*

b. Menghitung Nilai Haar *Feature* Dengan *Integral Image*

Pada gambar 3.4 dapat dilihat bahwa fitur a dan b terdiri dari dua persegi panjang, sedangkan fitur c terdiri dari tiga persegi panjang dan fitur d empat persegi panjang. Cara menghitung nilai dari fitur ini adalah mengurangkan nilai piksel pada area hitam dengan piksel pada area putih. Untuk mempermudah proses penghitungan nilai fitur, metode ini menggunakan sebuah media berupa citra integral.

Citra integral adalah sebuah citra yang nilai tiap pikselnya merupakan akumulasi dari nilai piksel atas dan kirinya. Sebagai contoh, piksel (a,b) memiliki nilai akumulatif untuk semua piksel (x,y) dimana $x \leq a$ dan $y \leq b$. Salah satu contoh dalam perangkat lunak ini selain untuk perhitungan area mata juga digunakan untuk perhitungan bagian tepi wajah, seperti perhitungan pada halaman berikutnya.



Gambar 3. 4 Contoh Proses Deteksi

Sebagai contoh perhitungan *integral image* dengan angka-angka piksel sampel dapat dilihat pada perhitungan di bawah ini.

1	2	3
2	3	4
4	5	1

1	1+2	1+2+3
1+2	1+2+2+3	1+2+3+2+3+4
1+2+4	1+2+2+3+4+5	1+2+3+2+3+4+4+5+1

Maka hasil perhitungan citra integralnya adalah :

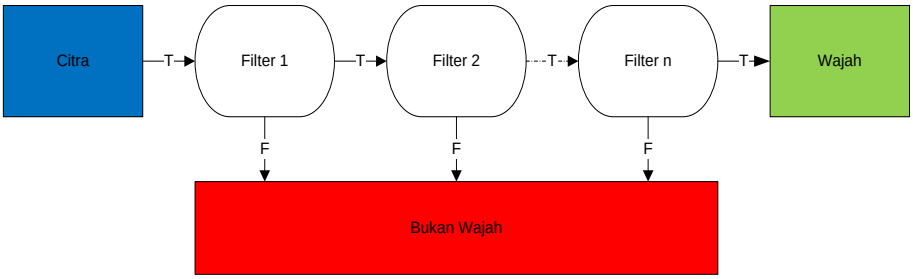
1	3	6
3	8	15
7	17	25

Dari perhitungan integral image diatas, maka diperolah nilai integral fitur putih= { 3,8,15 }dan nilai integral fitur hitam = {7,17,25}. Maka nilai Haar *feature* tersebut adalah :

$$\mathbf{f(x) = (7 + 17 + 25) - (3 + 8 + 15) = 23}$$

c. Membuat *Cascade Classifier*

Pada proses ini menggunakan klasifikasi bertingkat. Filter pada masing-masing level mengklasifikasikan gambar yang sebelumnya telah difilter. Ketika filter berhasil melewatkan *image region*, *image region* kemudian masuk pada filter yang selanjutnya. *Image region* yang telah melalui semua filter akan dianggap sebagai "Wajah" dan jika filter tersebut gagal, maka daerah tersebut dianggap sebagai "Bukan Wajah".



Gambar 3. 5 Cascade Classifier

Untuk dapat menentukan wajah atau bukan wajah, Haar menggunakan sebuah pelatihan. Proses pelatihan ini dikenal dengan algoritma *haartraining* yang pada akhir pelatihan akan menghasilkan parameter model statistik.

Training ini memiliki isi yang berbeda baik dari segi jumlah *stage*, jumlah *tree*, model segi empat dari fitur (*rectangle*) maupun nilai *treshold* nya. *Stage* melambangkan banyaknya tingkatan dalam *cascade of classifier*, dalam *training* ini digunakan 22 tingkatan (stage 0 sampai stage 21). Tingkatan ini digunakan untuk mengurangi jumlah *sub window* citra yang perlu diperiksa. Di tingkatan pertama dilakukan

pengklasifikasian terhadap seluruh *sub window* citra, lalu di tingkatan kedua dilakukan pengklasifikasian terhadap *sub window* yang berasal dari hasil pengklarifikasian tingkatan pertama. Maka semakin tinggi tingkatanya semakin sedikiit jumlah *sub window* yang harus diperiksa. Di tiap tingkatan terdiri dari beberapa *tree*, biasanya semakin tinggi tingkatan maka *tree* yang terdapat di dalamnya pun semakin banyak. Pada *stage* 0 terdapat 2 *tree* dan pada *stage* 21 terdapat 212 *tree*. Setelah proses pendeteksian wajah berhasil dilakukan, maka sistem akan menetukan wajah dan tidak.

Pseudo Code Algoritma Haar Cascade Classifier :

```
1. function run (r: Rectangle, tree: FeatureTree) ←
   integer
2.  (I.S : nilai rectangle dari strong classifier dan
    tree telah terdefinisi)
3.  (F.S : menghasilkan classifier terbaik)
4.
5.  Kamus :
6.  x,y,w,h : integer
7.  mean : real
8.  variance_norm_factor, inv_window_area : real
9.  features : array of real
10. val, sum, st_th : real
11. i,j : integer
12. feature : FeatureBase
13. tree : FeatureTree
14.
15. Algoritma :
16.
17. x ← r↑.x
18. y ← r↑.y
19. w ← r↑.width
20. h ← r↑.height
21. val ← 0
22. sum ← 0
23. i ← 0
24. j ← 0
25. st_th ← 0
26.
27. mean ← targetImage.getSum(x,y,w,h) *
    inv_window_area
28. variance_norm_factor ← targetImage.getSum2(x,y,w,h)*
```

```
   inv_window_area - mean*mean
29.
30. if(variance_norm_factor >= 0) then
31.    variance_norm_factor ←
   Math.sqrt(variance_norm_factor)
32. else
33.    variance_norm_factor ← 1
34. endif
35.
36. while (tree ≠ nil) do
37.    feature ← tree.firstFeature
38.    val ← 0
39.    st_th ← tree.stage_threshold
40.
41.    while (feature  ≠ nil) do
42.          sum ← feature.getSum(targetImage, x, y)
43.
44.          if (sum < feature.threshold *
   variance_norm_factor) then
45.                val = val + feature.left_val
46.          else
47.                val = val + feature.right_val
48.          endif
49.
50.          if (val > st_th) then
51.                break
52.          endif
53.          feature ← feature.next
54.    endwhile
55.
56.    if (val < st_th) then
57.          return 0
58.    endif
59.    tree ← tree.next
60. endwhile
61. return 1
62. endfunction
```

Kompleksitas potongan *pseudocode* di atas dapat dianalisis dengan menggunakan metode Big O. Pada halaman berikutnya dapat dilihat hasil dari analisis Big O.

Pseudocode	**Nilai Big O**
1. x ← r.x	O(1)
2. y ← r.y	O(1)

3. w ← r↑.width	O(1)
4. h ← r↑.height	O(1)
5. mean ← targetImage.getSum(x,y,w,h) * inv_window_area	O(1)
6. variance_norm_factor ← targetImage.getSum2(x,y,w,h)* inv_window_area - mean*mean	O(1)
7. if(variance_norm_factor >= 0) then	O(1)
8. variance_norm_factor ← Math.sqrt(variance_norm_factor)	O(1)
9. else	
10. variance_norm_factor ← 1	O(1)
11. endif	
12. while (tree ≠ nil) do	O(n)
13. feature ← tree.firstFeature	O(1)
14. val ← 0	O(1)
15. st_th ← tree.stage_threshold	O(1)
16. while (feature ≠ nil) do	O(n)
17. sum ← feature.getSum(targetImage, x, y)	O(1)
18. if (sum < feature.threshold * variance_norm_factor)	O(1)

then	
19. val = val + feature.left_val	O(1)
20. else	
21. val = val + feature.right_val	O(1)
22. endif	
23. if (val > st_th) then	O(1)
24. break	O(1)
25. endif	
26. feature ← feature.next	O(1)
27. endwhile	
28. if (val < st_th) then	O(1)
29. return 0	O(1)
30. endif	
31. tree ← tree.next	O(1)
32. endwhile	
33. return 1	O(1)
34. endfunction	
Jumlah	**$O(n^2)$**

Dapat dilihat bahwa jumlah nilai kompleksitas Big O dari Algoritma Haar Cascade Classifier adalah **O(n^2).**

Setelah melalui beberapa proses penyaringan akhirnya didapatkan posisi wajah yang sudah pasti seperti diperlihatkan pada gambar di bawah ini.



Gambar 3. 6 Posisi Wajah Terdeteksi

3.1.3 Analisis Spesifikasi Kebutuhan Perangkat Lunak

Pada tahap analisis ini, terdapat dua analisis spesifikasi kebutuhan perangkat lunak yaitu analisis SKPL-F (Spesifikasi kebutuhan perangkat lunak fungsional) dan analisis SKPL-NF (Spesifikasi kebutuhan perangkat lunak non-fungsional). Kedua analisis tersebut berisi deskripsi dari kebutuhan perangkat lunak yang akan dibangun.

3.1.3.1 Analisis Spesifikasi Kebutuhan Fungsional

Analisis spesifikasi kebutuhan fungsional berisi pernyataan dari layanan sistem yang harus disediakan, bagaimana sistem harus bereaksi terhadap input tertentu dan bagaimana sistem harus berperilaku dalam situasi tertentu. Intinya yaitu menjelaskan fungsionalitas dari sistem yang akan dibangun secara detail. Di bawah ini merupakan spesifikasi kebutuhan fungsional.

Tabel 3. 1 Analisis Spesifikasi Kebutuhan Fungsional

Nomor	Spesifikasi Kebutuhan Fungsional
SRS-F-001	Sistem hanya dapat dikelola oleh admin.
SRS-F-002	Sistem harus bisa menyediakan layanan pengelolaan model aksesories yang berformat (*.png) oleh admin
SRS-F-003	Sistem harus dapat meng-*capture* wajah terhadap kamera.
SRS-F-004	Sistem dapat mengambil foto *user* setelah menggunakan virtual aksesories.
SRS-F-005	Sistem menyediakan fitur *order* untuk membeli aksesories yang sedang digunakan.

3.1.3.2 Analisis Spesifikasi Kebutuhan Non-Fungsional

Analisis spesifikasi kebutuhan non-fungsional merupakan deskripsi mengenai batasan dari layanan dan fungsi-fungsi dari sistem yang akan dibangun. Di bawah ini merupakan spesifikasi kebutuhan non-fungsional perangkat lunak yang akan dibangun :

Tabel 3. 2 Analisis Spesifikasi Kebutuhan Non-Fungsional

Nomor	Spesifikasi Kebutuhan Non-Fungsional
SRS-NF-001	Sistem dibangun dalam bentuk *website*.
SRS-NF-002	Sistem dapat mendeteksi wajah dengan baik apabila menggunakan *webcam* minimal 1.3 *Megapixel*.
SRS-NF-003	Posisi wajah harus diluruskan *(Frontal Face)* terhadap *webcam*.

SRS-NF-004	Sistem memerlukan cahaya yang berada di depan kamera dan di belakang kamera agar wajah dapat terdeteksi dengan baik.
SRS-NF-005	Objek yang dideteksi oleh sistem harus memiliki fitur wajah yang lengkap seperti dua mata, hidung, dan mulut.

3.1.4 Analisis Kebutuhan Non-Fungsional

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

3.1.4.1 Analisis Kebutuhan Perangkat Keras

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

1. Processor AMD Athlon II X2 2.8 GHz.
2. RAM 4 GB.
3. Harddisk 250 MB.
4. Monitor LED 16’.
5. VGA ATI Radeon HD 4600 Series.
6. *Webcam* 1.3 *Megapixel.*

Rekomendasi spesifikasi minimum perangkat keras untuk mengakses perangkat lunak *virtual accessories* room berbasis web di Distro Heaven Skateboards Bandung ini adalah sebagai berikut :

1. Processor minimal 1.8 GHz.
2. RAM 1 GB atau lebih.
3. Harddisk 60 GB.
4. Monitor CRT 14’.
5. VGA NVidia GeForce.
6. *Webcam* 2 *Megapixel.*

3.1.4.2 Analisis Kebutuhan Perangkat Lunak

Perangkat lunak yang dibutuhkan untuk menjalankan aplikasi *virtual accessories* room berbasis web di Distro Heaven Skateboards Bandung ini adalah sebagai berikut :

1. Sistem operasi Windows 7.
2. *Library* Marilena.
3. Adobe Flash Player 11.1.
4. Adobe Flash Professional CS5.
5. Adobe Photoshop CS4.
6. Google Chrome.

Berdasarkan analisis perangkat lunak di atas, rekomendasi minimum perangkat lunak untuk mengakses perangkat *virtual accessories room* berbasis web di Distro Heaven Skateboards Bandung adalah sebagai berikut :

1. Adobe Flash Player 10.

2. Google Chrome, Internet Explorer, atau Mozilla Firefox.

3.1.4.3 Analisis Kebutuhan *User*

Pengunjung yang mengunjungi Distro Heaven Skateboards ini berasal dari berbagai kalangan, terutama untuk anak-anak *skateboards* yang hobi membeli aksesories. Jadi, terdapat karakteristik *user web accessories room* ini adalah sebagai berikut :

1. *User* harus memiliki kemampuan menggunakan internet dan *web browser*.
2. *User* tidak memiliki buta warna.
3. *User* tidak memiliki cacat pada wajah, seperti tidak memiliki mata, hidung, mulut, dll.
4. *User* harus mengerti bahasa Inggris dengan level minimal *low*.

3.1.5 Analisis File

Analisis file yang digunakan pada sistem ini yaitu *file* XML yang akan digunakan untuk menyimpan data-data dari setiap objek. Di bawah ini merupakan struktur secara umum :

```
<?xml version=”1.0” encoding=”ISO-8859-1”?>
<images>
      <image>
          <id>          </id>
          <file>        </file>
          <thumb>       </thumb>
      </image>
</images>
```

Berikut ini merupakan deskripsi dari setiap *tag* xml yang digunakan :

Tabel 3. 3 Deskripsi *Tag* XML

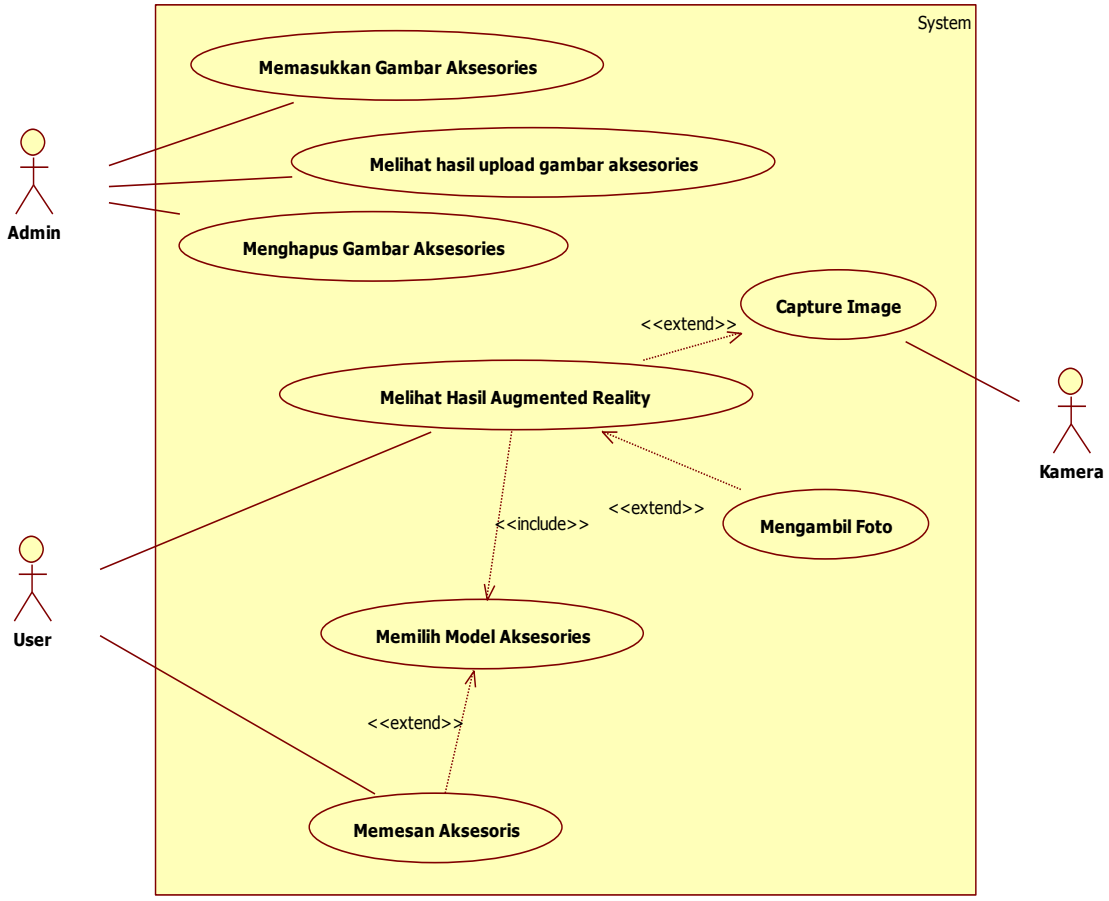
***Tag* XML**	**Fungsi**
<images> </images>	Merupakan root dari xml
<image> </image>	Alamat gambar
<id></id>	Menyimpan ID aksesories
<file> </file>	Menyimpan gambar virtual aksesories
<thumb> </thumb>	Menyimpan gambar aksesories

3.1.6 Analisis Kebutuhan Fungsional

Analisis kebutuhan fungsional merupakan analisis yang diperlukan untuk memodelkan sistem. Pemodelan yang digunakan untuk memodelkan perangkat lunak virtual aksesories ini adalah pemodelan berorientasi objek yang dimodelkan dengan menggunakan *tools* StarUML (*Star Unified Modeling Language*).

3.1.6.1 *Use Case Diagram*

Use Case Diagram digunakan untuk menggambarkan konteks sistem yang akan dibangun dan fungsionalitas yang disediakan oleh sistem itu. Use Case Diagram juga menggambarkan siapa (atau apa) berinteraksi dengan sistem. Di dalam sistem hanya terdapat satu *user*. *User* ini yang akan menggunakan sistem yang terpasang pada *website* Heaven Skateboards. Tugas *user* yaitu sebagai aktor yang dapat terlihat pada gambar 3.9 pada halaman berikutnya.



Gambar 3. 7 *Use Case Diagram*

Terdapat beberapa *use case* scenario untuk menjalankan aplikasi ini, diantaranya yaitu :

1. *Use Case* Skenario Memasukkan Gambar Aksesories

Tabel 3. 4 *Use Case* Skenario Memasukkan Gambar Aksesories

Nama	Memasukkan gambar aksesories
Aktor	Admin
Trigger	Admin membuka form masukkan gambar aksesories
Skenario Utama	

Kondisi Awal	Admin telah melakukan login ke *website*
Aksi Aktor	**Reaksi Sistem**
1. Admin mengupload model aksesories (*.png)	
	2. Sistem mengecek gambar
	3. Sistem menyalin gambar aksesories ke server.
4. Admin mengupload gambar *thumbnails* (*.png)	
	5. Sistem mencek size gambar
	6. Sistem menyalin gambar thumbnails ke server
7. Admin menekan tombol simpan	
	8. Sistem menyimpan data alamat gambar aksesories dan thumbnails aksesories ke xml.
Kondisi Akhir	Sistem dapat menyimpan data baru ke xml

2. Use Case Skenario Melihat Hasil *Upload* Gambar Aksesories

Tabel 3. 5 Use Case Skenario Melihat Hasil Upload Gambar Aksesories

Nama	Melihat hasil *upload* gambar aksesories	
Aktor	Admin	
Trigger	Admin membuka form *view* gambar aksesories	
Skenario Utama		
Kondisi Awal	Admin telah melakukan login ke website	
Aksi Aktor		**Reaksi Sistem**
1. Admin memilih model aksesories		
		2. Sistem menampilkan thumbnails aksesories yang dipilih
Kondisi Akhir	Sistem berhasil menampilkan gambar aksesories	

3. *Use Case* Skenario Menghapus Gambar Aksesories

Tabel 3. 6 *Use Case* Skenario Menghapus Gambar Aksesories

Nama	Menghapus gambar aksesories
Aktor	Admin
Trigger	Admin membuka form hapus gambar aksesories
Skenario Utama	
Kondisi Awal	Admin telah melakukan login ke website

Aksi Aktor	Reaksi Sistem
3. Admin memilih thumbnails aksesories	
	4. Sistem menampilkan thumbnails aksesories yang dipilih
5. Admin menekan tombol hapus	
	6. Sistem menghapus gambar XML yang dipih
Kondisi Akhir	Sistem berhasil menghapus data di XML

4. *Use Case* Skenario *Capture Image*

Tabel 3. 7 *Use Case* Skenario *Capture Image*

Nama	*Capture image*
Aktor	Kamera
Trigger	*User* mengaktifkan kamera
Skenario Utama	
Kondisi Awal	Sistem telah terhubung dengan kamera

Aksi Aktor	Reaksi Sistem
1. Menempatkan wajah dengan kondisi ruangan yang terang di depan kamera	2. Melakukan pengolahan citra
	3. Melakukan deteksi posisi

	wajah
	4. Menangkap koordinat wajah
Kondisi Akhir	Koordinat mata atau wajah yang diperoleh menjadi input proses selanjutnya
Skenario Alternatif	
Aksi Aktor	**Reaksi Sistem**
1. Menempatkan wajah dengan kondisi ruangan yang gelap di depan kamera	2. Melakukan pengolahan citra
	3. Melakukan deteksi posisi wajah
	4. Proses deteksi gagal

5. *Use Case Skenario* Memilih Model Aksesories

Tabel 3. 8 *Use Case Skenario* Memilih Model Aksesories

Nama	Memilih model aksesories
Aktor	*User*
Trigger	*User* memilih gambar aksesories ada di menu
Skenario Utama	
Kondisi Awal	Sistem telah menerima input koordinat posisi wajah
Aksi Aktor	**Reaksi Sistem**
1. Memilih objek pada menu topi atau kacamata	

	2. Memanggil spesifikasi model dari file xml
	3. Menampilkan spesifikasi model
	4. Merender model aksesories yang dipilih
	5. Menempatkan aksesories pada koordinat yang telah didapat.
Kondisi Akhir	Sistem dapat menempatkan aksesories virtual pada posisi kepala *user*

6. Use Case Skenario Melihat Hasil *Augmented Reality*

Tabel 3. 9 Use Case Skenario Melihat Hasil Augmented Reality

Nama	Melihat hasil *augmented reality*	
Aktor	*User*	
Trigger	*User* menghadap tegak lurus ke kamera	
Skenario Utama		
Kondisi Awal	Model aksesories yang ditampilkan telah dipilih oleh *user*	
Aksi Aktor		**Reaksi Sistem**
1. *User* menghadap tegak lurus terhadap kamera		

	2. Menampilkan model aksesories ke layar
3. Melakukan pergerakan mata atau wajah yang dapat tertangkap kamera	
	4. Model aksesories mengikuti pergerakan wajah
Kondisi Akhir	User dapat melihat model aksesories yang telah digunakan
Skenario Alternatif	
Aksi Aktor	**Reaksi Sistem**
1. Posisi mata atau wajah tidak menghadap kamera	
	2. Gagal menampilkan model aksesories

7. *Use Case* Skenario Mengambil Foto

Tabel 3. 10 *Use Case* Skenario Mengambil Foto

Nama	Mengambil Foto
Aktor	*User*
Trigger	*User* memilih menu foto
Skenario Utama	

Kondisi Awal	Sistem telah menerima input berupa video *real time* yang telah ada virtual aksesories
Aksi Aktor	**Reaksi Sistem**
1. Menekan tombol *capture* pada menu	
	2. Mengkopi piksel di layar
	3. Menampung hasil sementara
Kondisi Akhir	Sistem dapat mengambil gambar user secara *real time*

8. *Use Case* Skenario Memesan Aksesories

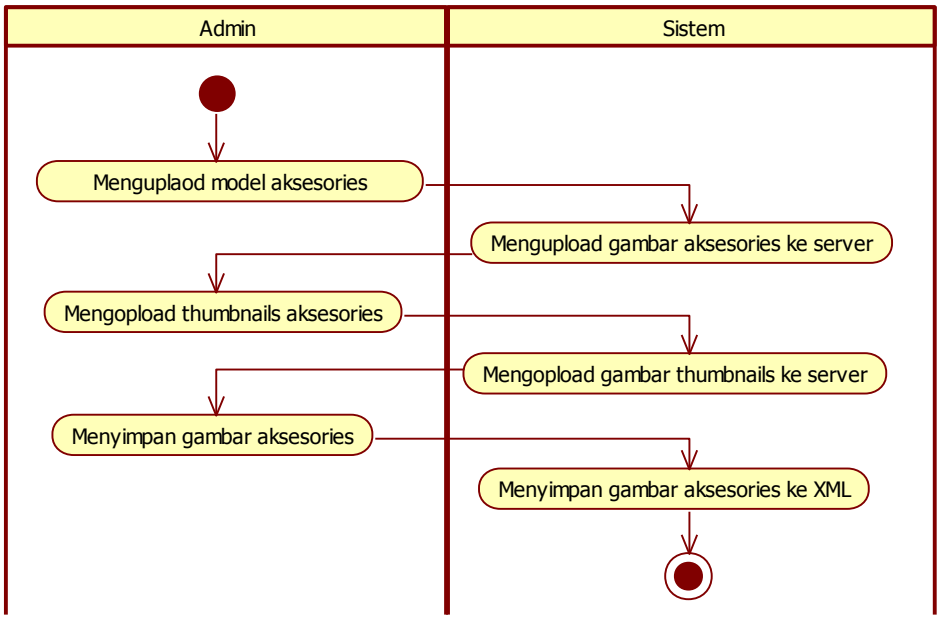
Tabel 3. 11 *Use Case* Skenario Memesan Aksesories

Nama	Memesan aksesories
Aktor	*User*
Trigger	*User* memilih menu *order*
Skenario Utama	
Kondisi Awal	User telah memilih aksesories yang telah digunakan
Aksi Aktor	**Reaksi Sistem**
1. Memilih menu *order*	
	2. Mengambil gambar aksesories yang akan dikirim ke keranjang belanja
Kondisi Akhir	Keranjang belanja telah terisi item aksesories yang telah dicoba oleh *user*

3.1.6.2 *Activity Diagram*

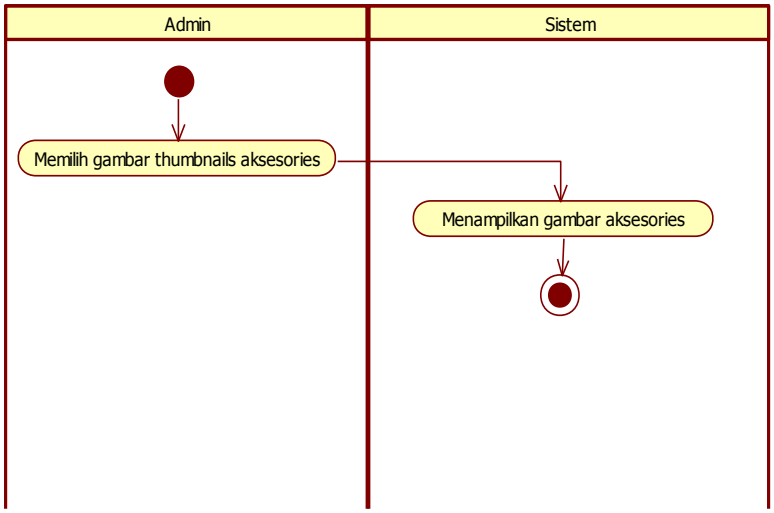
Activity diagram merupakan cara menggambarkan proses bisnis dan urutan aktivitas dalam sebuah proses yang dipakai pada *business modeling* untuk memperlihatkan urutan aktifitas proses bisnis. Struktur diagram ini mirip *flowchart* atau *Data Flow Diagram* (DFD) pada perancangan terstruktur. *Activity diagram* sangat bermanfaat apabila kita membuat diagram ini terlebih dahulu dalam memodelkan sebuah proses untuk membantu memahami proses secara keseluruhan. *Activity diagram* dibuat berdasarkan sebuah atau beberapa *use case* pada *use case diagram* yang dapat dilihat pada halaman berikutnya.

1. *Activity Diagram* Memasukkan Gambar Aksesories



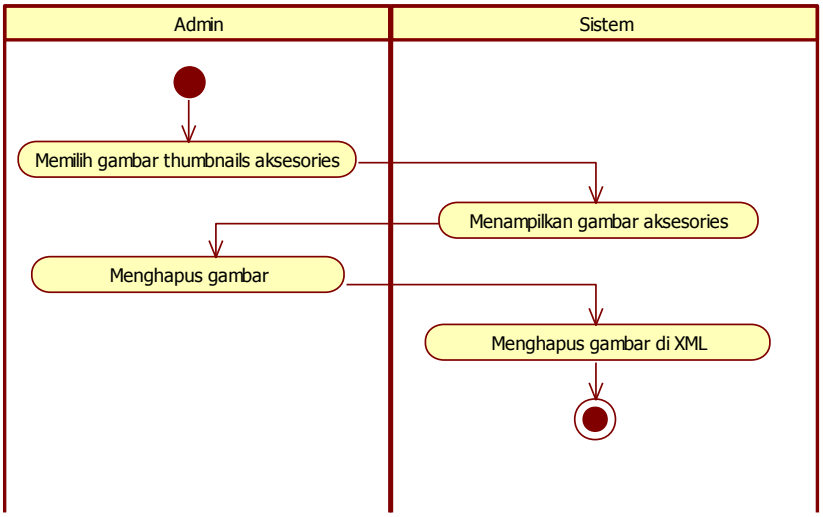
Gambar 3. 8 *Activity Diagram* Memasukkan Gambar Aksesories

2. *Activity Diagram* Melihat Hasil *Upload* Gambar Aksesories



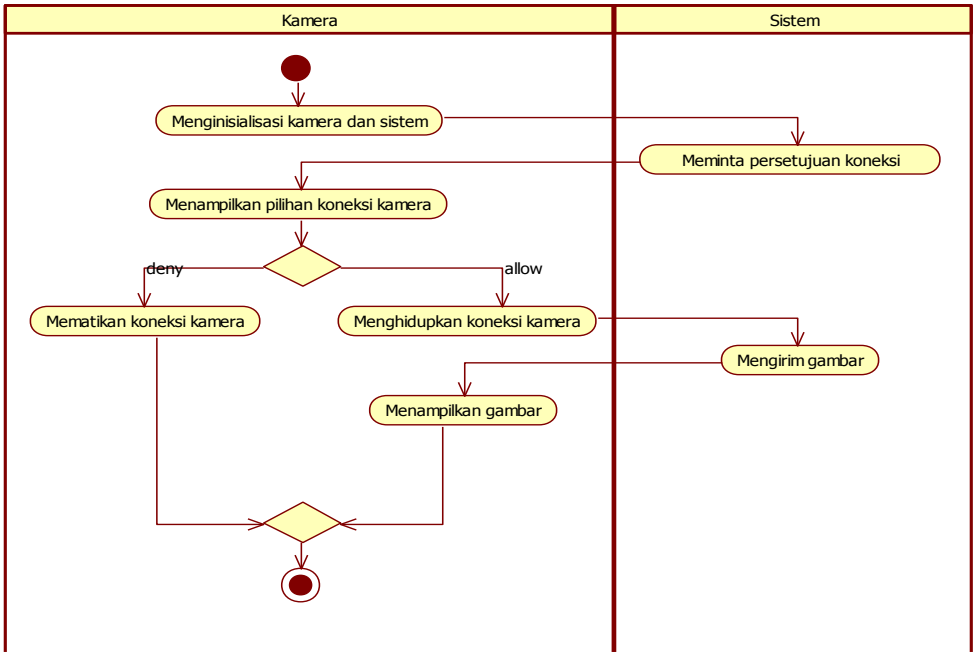
Gambar 3. 9 Activity Diagram Melihat Hasil Upload Gambar Aksesories

3. *Activity Diagram* Menghapus Gambar Aksesories



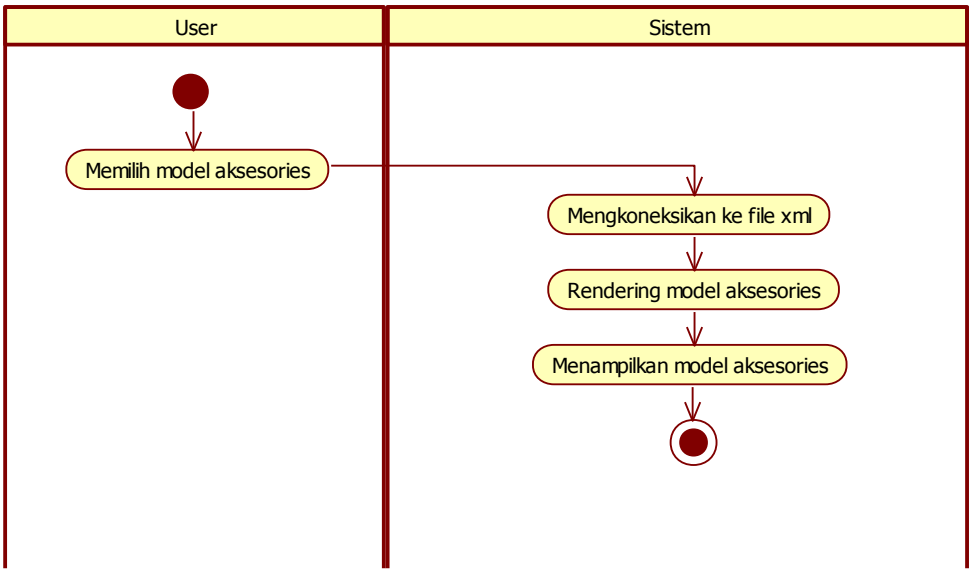
Gambar 3. 10 *Activity Diagram* Menghapus Gambar Aksesories

4. *Activity Diagram Capture Image*



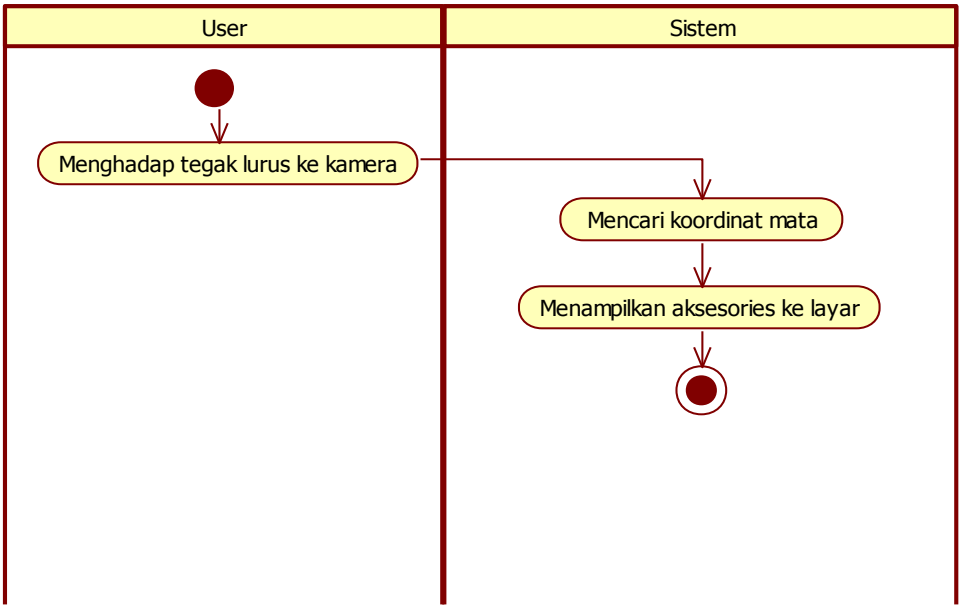
Gambar 3. 11 *Activity Diagram Capture Image*

5. *Activity Diagram* Memilih Model Aksesories



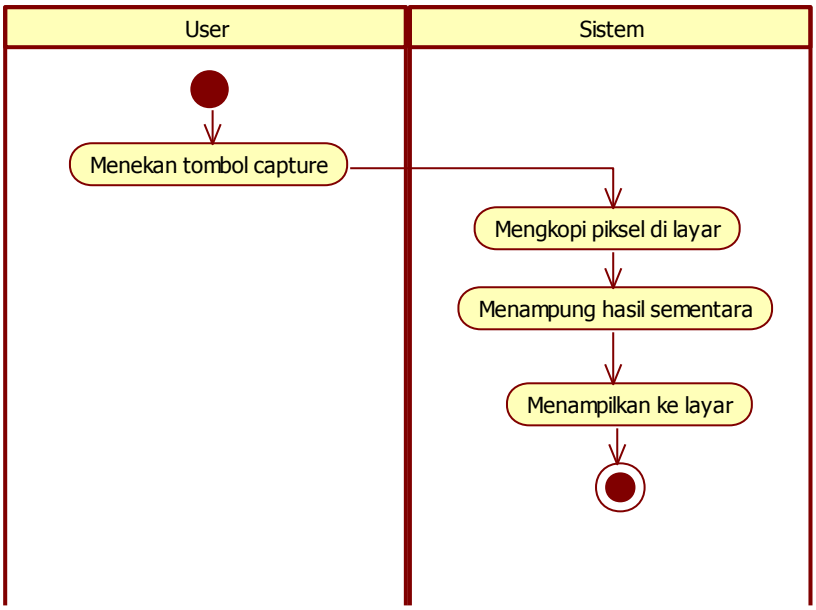
Gambar 3. 12 *Activity Diagram* Memilih Model Aksesories

6. *Activity Diagram* Melihat Hasil *Augmented Reality*



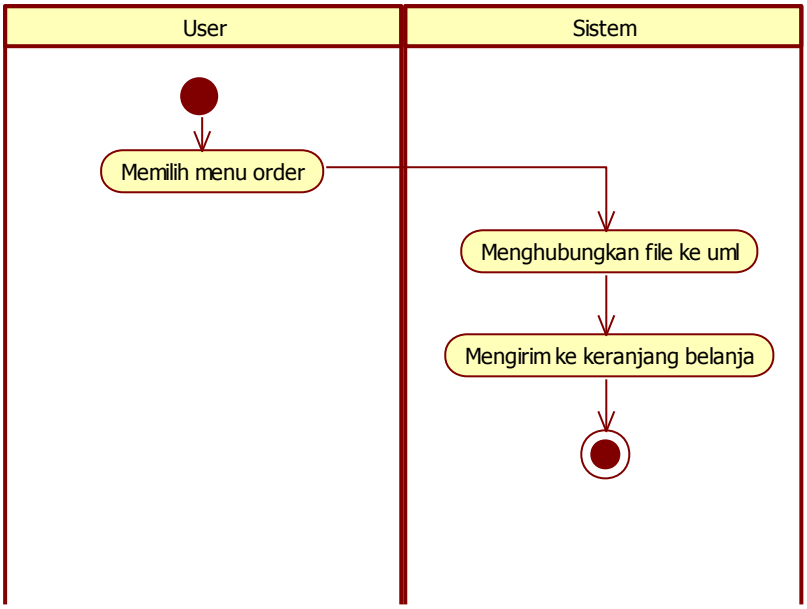
Gambar 3. 13 *Activity Diagram* Melihat Hasil *Augmented Reality*

7. *Activity Diagram* Mengambil Foto



Gambar 3. 14 *Activity Diagram* Mengambil Foto

8. *Activity Diagram* Memesan Aksesories

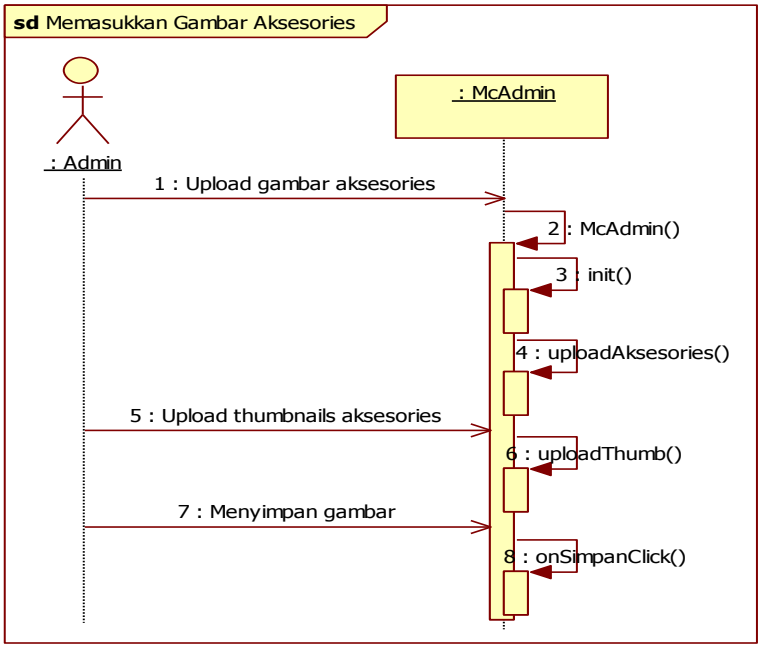


Gambar 3. 15 *Activity Diagram* Memesan Aksesories

3.1.6.3 *Sequence Diagram*

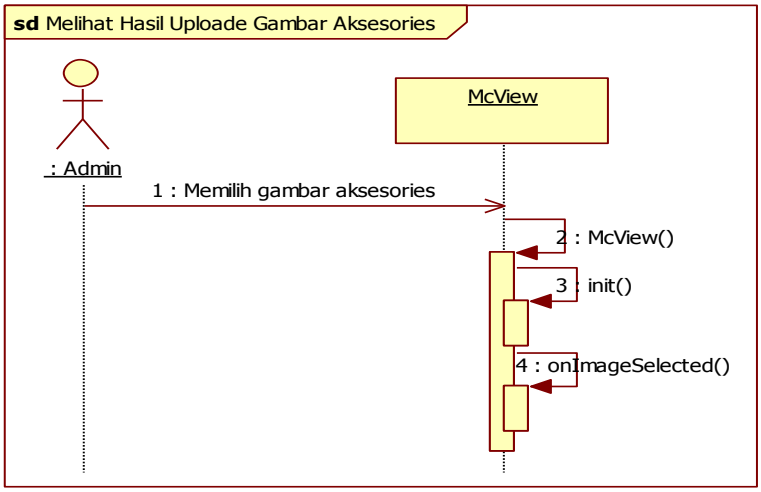
Sequence diagram merupakan suatu diagram yang menggambarkan interaksi antar objek dan mengindikasikan komunikasi diantara objek-objek tersebut. Diagram ini juga menunjukkan serangkaian pesan yang dipertukarkan oleh objek-objek yang melakukan suatu tugas atau aksi tertentu. Objek-objek tersebut kemudian diurutkan dari kiri ke kanan, aktor yang menginisialisasi interaksi biasanya disimpan di paling kiri dari diagram. Pada halaman berikutnya dapat dilihat *sequence diagram* pada aplikasi virtual aksesories.

1. *Sequence Diagram* Memasukkan Gambar Aksesories



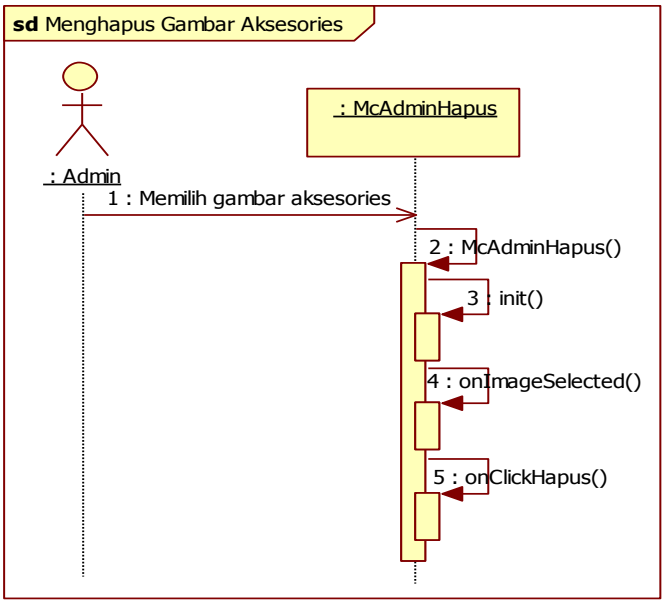
Gambar 3. 16 *Sequence Diagram* Memasukkan Gambar Aksesories

2. *Sequence Diagram* Melihat Hasil *Uplad* Gambar Aksesories



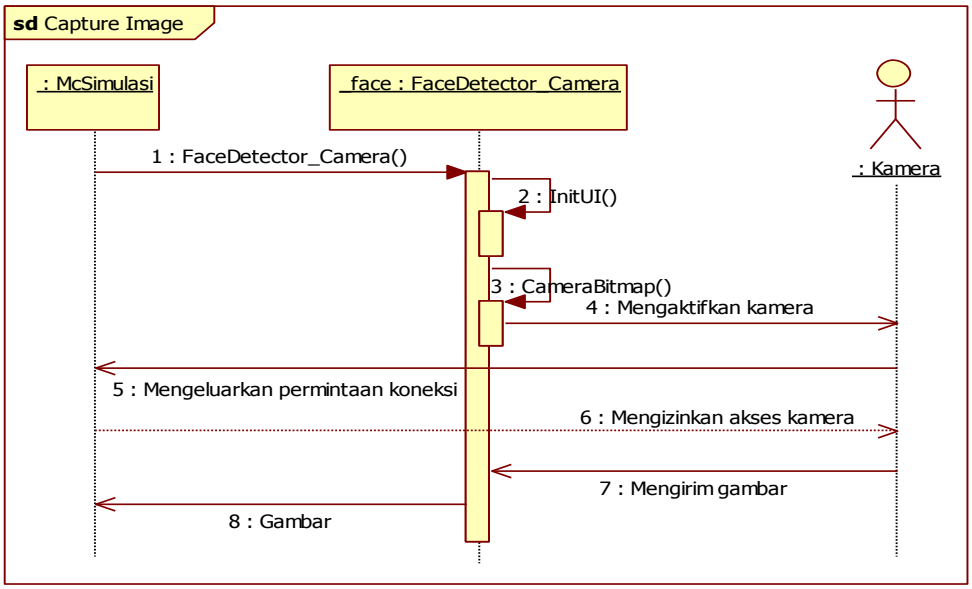
Gambar 3. 17 Sequence Diagram Melihat Hasil Upload Gambar Aksesories

3. *Sequence Diagram* Menghapus Gambar Aksesories



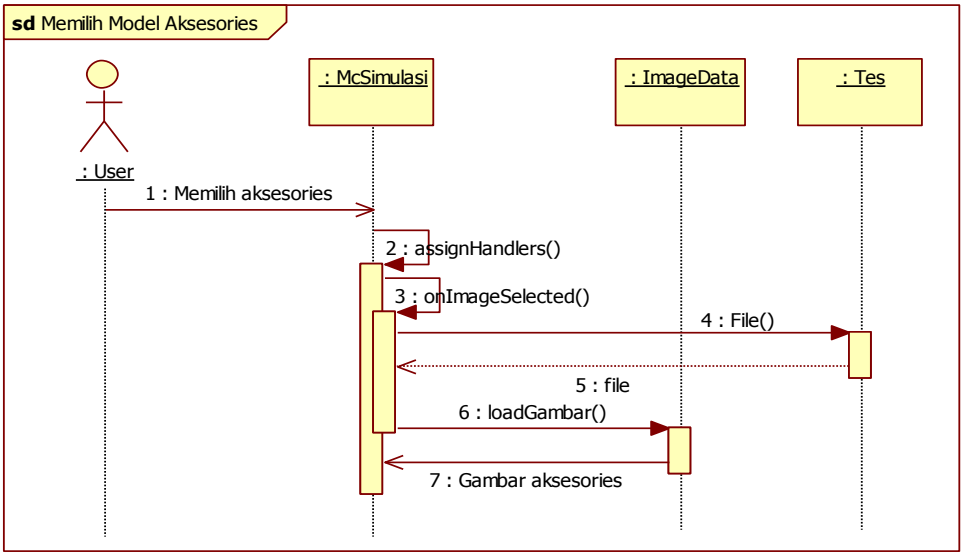
Gambar 3. 18 *Sequence Diagram* Menghapus Gambar Aksesories

4. *Sequence Diagram Capture Image*



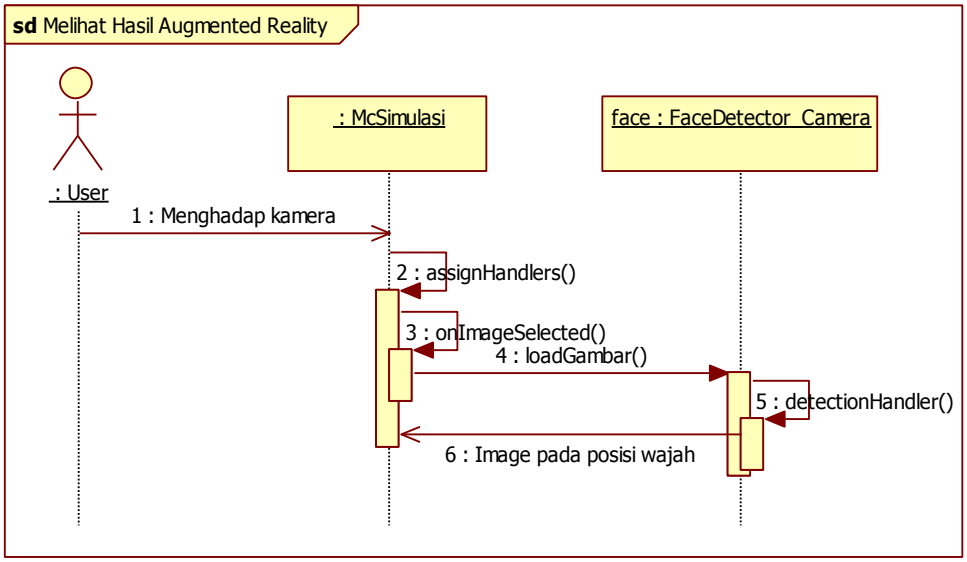
Gambar 3. 19 *Sequence Diagram Capture Image*

5. *Sequence Diagram* Memilih Model Aksesories



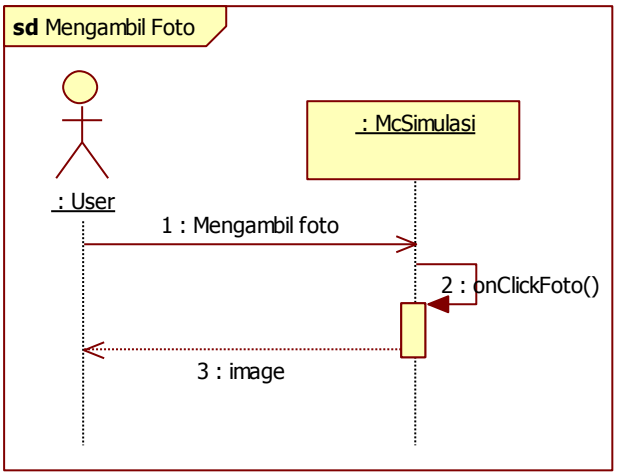
Gambar 3. 20 *Sequence Diagram* Memilih Model Aksesories

6. *Sequence Diagram* Melihat Hasil *Augmented Reality*



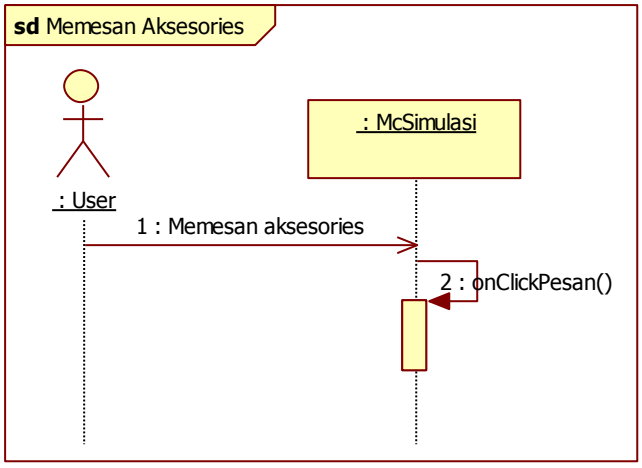
Gambar 3. 21 *Sequence Diagram* Melihat Hasil *Augmented Reality*

7. *Sequence Diagram* Mengambil Foto



Gambar 3. 22 *Sequence Diagram* Mengambil Foto

8. *Sequence Diagram* Memesan Aksesories

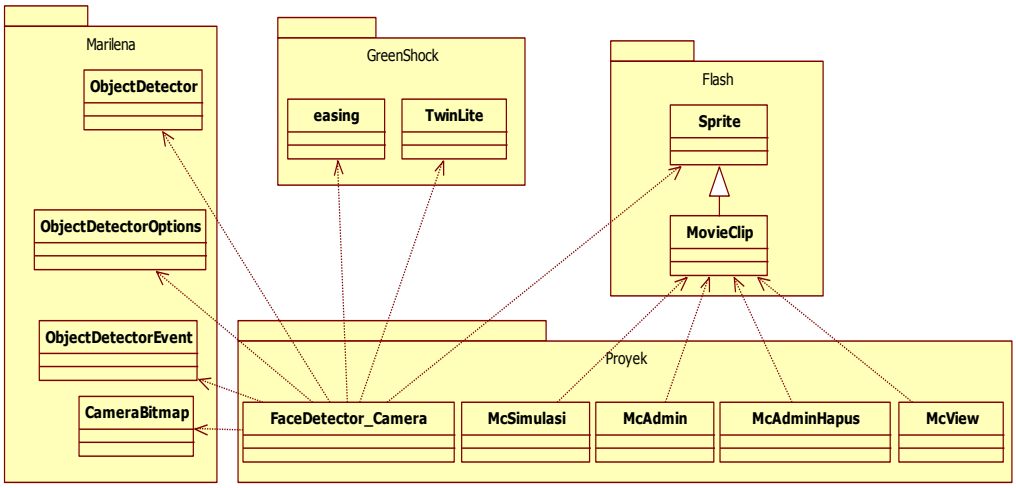


Gambar 3. 23 *Sequence Diagram* Memesan Aksesories

3.1.6.4 *Package Diagram*

Package diagram digunakan untuk menjelaskan hubungan modularitas, enkapsulasi, dan abstraksi pada pengembangan sistem yang kompleks. *Package*

diagram juga digunakan untuk menjelaskan hubungan *dependency* dan *visibility* antar *class* yang memiliki asosiai secara modular dengan *class* lain dalam sistem. *Package diagram* memungkinkan perancangan sistem bisa di mengerti dengan tingkat pemahaman yang lebih tinggi pada kolaborasi antar elemen model. Dengan menggunakan *package diagram* perancangan *class diagram* bisa di modularisasi secara lebih terstruktur. *Package diagram* sangat dibutuhkan ketika *class diagram* yang dibangun sudah semakin kompleks, sehingga penggambaran *class diagram* bisa dibangun secara terpisah dengan menggunakan *Package diagram* untuk mewakili hubungan yang antar class pada *package* yang berbeda. Di bawah ini merupakan *package diagram* dari *virtual accessories room*.

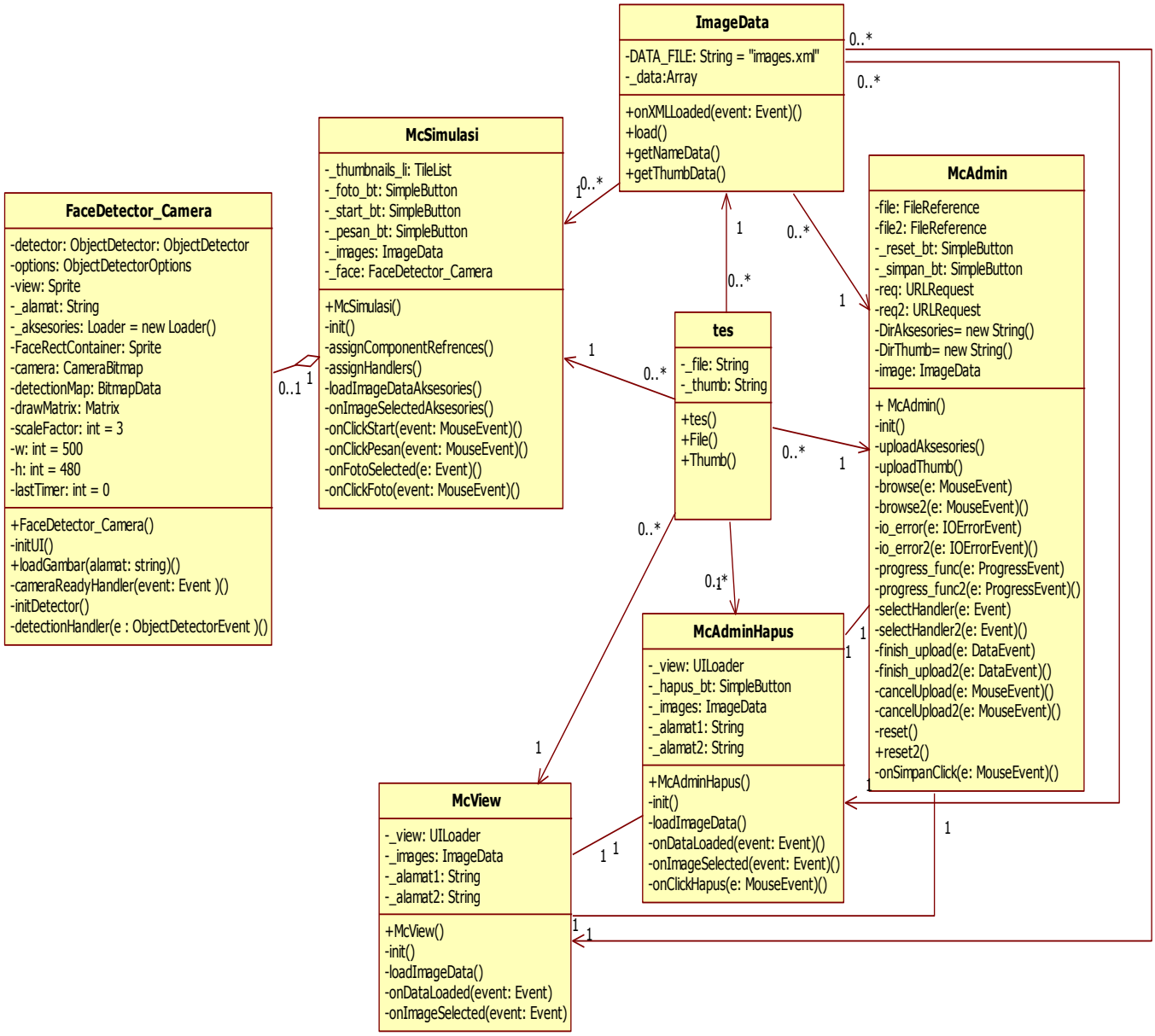


Gambar 3. 24 *Package Diagram*

3.1.6.5 *Class Diagram*

Class diagram digunakan untuk menunjukan hubungan dan eksistensi sebuah class dari sudut pandang logis dalam sistem. Sebuah Class Diagram merepresentasikan struktur class yang membentuk arsitektur dan perilaku sistem

dalam proses desain dan analisis. *Class diagram* memberikan gambaran secara menyeluruh tentang object dan class serta relasinya pada arsitektur sistem yang dibangun. *Class diagram* menyediakan fasilitas berbagi model sehingga model yang telah dibangun bisa digunakan kembali (reusable) untuk pemodelan yang lebih rinci pada perilaku dinamis. Notasi pada *class diagram* bisa digunakan untuk menjelaskan *Object Oriented Programming,* sehingga semua konsep *Object Oriented Programming* bisa di jelaskan menggunakan notasi *Class diagram*. Pada halaman selanjutnya ini merupakan *class diagram* dari *virtual accessories room*.



Gambar 3. 25 *Class Diagram*

Definisi *Class Diagram* :

Tabel 3. 12 *Class* FaceDetector_Camera

Class	FaceDetector_Camera		
Fungsi	Untuk menangkap kamera		
Attribut	Nama	*Visibility*	Keterangan
	detector: ObjectDetector: ObjectDetector	*Private*	Objek dari kelas ObjectDetector
	options: ObjectDetectorOptions	*Private*	Objek dari kelas ObjectDetectorOptions
	view: Sprite	*Private*	Atribut bertipe data sprite untuk menampilkan kamera
	_alamat: String	*Private*	Atribut bertipe String untuk menampung alamat model virtual
	_aksesories: Loader = new Loader()	*Private*	Atribut untuk menampung model virtual
	FaceRectContainer: Sprite	*Private*	Atribut bertipe data sprite untuk menampilkan kamera
	camera: CameraBitmap	*Private*	Objek dari kelas CameraBitmap
	detectionMap:	*Private*	Atribut untuk mendeteksi

	BitmapData		kotak hitam pada deteksi wajah
	drawMatrix: Matrix	*Private*	Atribut untuk menampuk matriks wajah
	scaleFactor: int = 3	*Private*	Skala untuk menampikan kamera
	w: int = 500	*Private*	Lebar dari layar kamera
	h: int = 480	*Private*	Tinggi dari layar kamera
	lastTimer: int = 0	*Private*	Waktu buat memulai kamera
Method	FaceDetector_Camera()	*Public*	*Method Constructor*
	initUI()	*Private*	Inisialisasi antarmuka
	loadGambar(alamat: string)()	*Public*	Buat load model gambar
	cameraReadyHandler(event: Event)()	*Private*	Event ketika kamera telah menyala
	initDetector()	*Private*	Inisialisasi deteksi
	detectionHandler(e : ObjectDetectorEvent)()	*Private*	Event ketika wajah telah terdeteksi

Tabel 3. 13 *Class* McSimulasi

Class	McSimulasi		
Fungsi	Antarmuka untuk user		
Attribut	Nama	*Visibility*	Keterangan

	_thumbnails_li: TileList	*Private*	Atribut untuk menyimpan daftar gambar
	_foto_bt: SimpleButton	*Private*	Atribut untuk foto gambar
	_start_bt: SimpleButton	*Private*	Atribut untuk memulai kamera
	_pesan_bt: SimpleButton	*Private*	Atribut untuk memesan produk
	_images: ImageData	*Private*	Objek dari kelas ImageData
	_face: FaceDetector_Camera	*Private*	Objek dari kelas FaceDetector_Camera
Method	McSimulasi()	*Public*	*Method Constructor*
	init()	*Private*	Inisialisasi Antarmuka
	assignComponentRefrences()	*Private*	Inisialisasi komponen yang ada di form
	assignHandlers()	*Private*	Untuk menghindari event yang terjadi
	loadImageDataAksesories()	*Private*	Untuk load gambar *thumbnails*
	onImageSelectedAksesories()	*Private*	Untuk klik gambar *thumbnails*
	onClickStart(event: MouseEvent)()	*Private*	Untuk memulai kamera
	onFotoSelected(e:	*Private*	Untuk menampung foto

	Event)()		
	onClickFoto(event: MouseEvent)()	*Private*	Untuk klik foto
	onClickPesan(event: MouseEvent)()	*Private*	Untuk klik pesan

Tabel 3. 14 *Class* ImageData

Class	ImageData		
Fungsi	Untuk control ke xml		
Attribut	Nama	*Visibility*	Keterangan
	DATA_FILE: String = "images.xml"	*Private*	Untuk menampung file yang ada di xml
	_data:Array	*Private*	Untuk menampung *array* di xml
Method	onXMLLoaded(event: Event)()	*Public*	Untuk load isi xml
	load()	*Public*	Event ketika load terjadi
	getNameData()	*Public*	Untuk memanggil gambar virtual
	getThumbData()	*Public*	Untuk memanggil *thumbnails* gambar

Tabel 3. 15 *Class* Tes

Class	Tes		
Fungsi	Untuk menampung gambar		
Attribut	Nama	*Visibility*	Keterangan
	_file: String	*Private*	Gambar virtual yang ada dalam xml
	_thumb: String	*Private*	Gambar *thumbnails* yang ada dalam xml
Method	tes()	*Public*	*Method Constructor*
	File()	*Public*	Untuk memanggil gambar virtual
	Thumb()	*Public*	Untuk memanggil gambar *thumbnails*

Tabel 3. 16 *Class* McAdmin

Class	McAdmin		
Fungsi	Form isi gambar		
Attribut	Nama	*Visibility*	Keterangan
	file: FileReference	*Private*	Untuk menampung gambar virtual ke PHP
	File2: FileReference	*Private*	Untuk menampung gambar *thumbnails* ke PHP
	_reset_bt: SimpleButton	*Private*	Untuk mengulang *upload*

	_simpan_bt: SimpleButton	*Private*	Untuk menyimpan hasil *upload*
	req: URLRequest	*Private*	Untuk manggil gambar virtual ke PHP
	Req2: URLRequest	*Private*	Untuk manggil gambar *thumbnails* ke PHP
	DirAksesories= new String()	*Private*	Tempan gambar virtual ditampung
	DirThumb= new String()	*Private*	Tempan gambar *thumbnails* ditampung
	image: ImageData	*Private*	Objek dari kelas ImageData
Method	McAdmin()	*Public*	Method Constructor
	init()	*Private*	Inisialisasi antarmuka
	uploadAksesories()	*Private*	Untuk upload gambar virtual
	uploadThumb()	*Private*	Untuk upload gambar thumbnails
	browse(e: MouseEvent) ()	*Private*	Untuk mencari gambar virtual yang akan diupload
	browse2(e: MouseEvent) ()	*Private*	Untuk mencari gambar *thumbnails* yang akan diupload
	io_error(e:	*Private*	Gagal upload gambar virtual

	IOErrorEvent) ()		
	io_error2(e: IOErrorEvent) ()	*Private*	Gagal upload gambar *thumbnails*
	progress_func(e: ProgressEvent) ()	*Private*	Progress selama upload gambar virtual
	progress_func2(e: ProgressEvent) ()	*Private*	Progress selama upload gambar *thumbails*
	selectHandler(e: Event) ()	*Private*	Event ketika memilih gambar virtual yang akan diupload
	selectHandler2(e: Event) ()	*Private*	Event ketika memilih gambar *thumbnails* yang akan diupload
	finish_upload(e: DataEvent) ()	*Private*	Upload gambar virtual berhasil
	finish_upload2(e: DataEvent) ()	*Private*	Upload gambar *thumbnails* berhasil
	cancelUpload(e: MouseEvent)()	*Private*	Upload gamabr virtual dibatalkan
	cancelUpload2(e: MouseEvent)()	*Private*	Upload gamabr *thumbnails* dibatalkan
	reset()	*Private*	Reset *upload* gambar virtual
	reset2()	*Public*	Reset *upload* gambar

			thumbnails
	onSimpanClick()	*Private*	Untuk menyimpan gambar yang telah di *upload* ke dalam xml

Tabel 3. 17 *Class* McAdminHapus

Class	McAdminHapus		
Fungsi	Form hapus gambar		
Attribut	Nama	*Visibility*	Keterangan
	_view: UILoader	*Private*	Untuk load gambar thumbnails
	_hapus_bt: SimpleButton	*Private*	Untuk menghapus gambar
	_images: ImageData	*Private*	Objek dari kelas ImageData
	_alamat1: String	*Private*	Untuk menampung alamat gambar virtual yang akan dihapus
	_alamat2: String	*Private*	Untuk menampung alamat gambar *thumbnails* yang akan dihapus
Method	McAdminHapus()	*Public*	Method Constructor
	init()	*Private*	Inisialisasi antarmuka

	loadImageData()	*Private*	Untuk memanggil gambar dari kelas ImageData
	onDataLoaded(event: Event)()	*Private*	Untuk meload gambar yang akan dihapus
	onImageSelected(event: Event)()	*Private*	Untuk memilih gambar yang akan dihapus
	onClickHapus(e: MouseEvent)()	*Private*	Untuk menghapus gambar

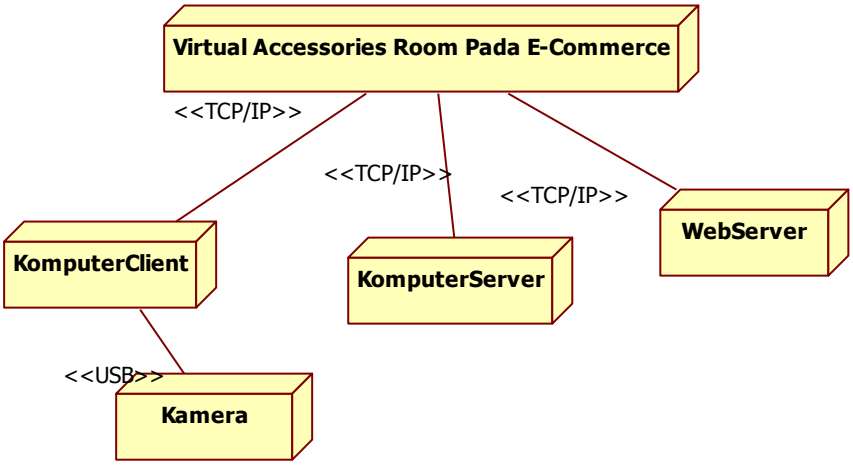
Tabel 3. 18 *Class* McView

Class	McView		
Fungsi	Untuk melihat list gambar		
Attribut	Nama	*Visibility*	Keterangan
	_view: UILoader	*Private*	Untuk load gambar thumbnails
	_images: ImageData	*Private*	Objek dari kelas ImageData
	_alamat1: String	*Private*	Untuk menampung alamat gambar virtual yang akan dihapus
	_alamat2: String	*Private*	Untuk menampung alamat gambar *thumbnails* yang akan dihapus
Method	McView()	*Public*	Method Constructor

	init()	*Private*	Inisialisasi antarmuka
	loadImageData()	*Private*	Untuk memanggil gambar dari kelas ImageData
	onDataLoaded(event: Event)	*Private*	Untuk meload gambar yang akan dilihar
	onImageSelected(event: Event)	*Private*	Untuk memilih gambar yang akan dilihat

3.1.6.6 *Deployment Diagram*

Deployment diagram digunakan untuk menunjukan alokasi artefak pada node dalam desain fisik sebuah sistem. Sebuah *deployment diagram* mewakili sebuah gambaran ke dalam struktur artefak suatu sistem. *Deployment diagram* lebih berfokus pada aspek fisik dari *object-oriented system*. Dengan menggunakan *deployment diagram*, gambaran mengenai distribusi komponen dan relasinya dengan sistem maupun *platform* tempat sistem berjalan bisa digambarkan dengan lebih jelas, sehingga dengan menggunakan *deployment diagram*, gambaran akan interaksi antara *software* dan *hardware* pada sistem akan terlihat lebih jelas. Gambar di bawah ini merupakan gambar *deployment diagram* dari *virtual accessories room*.



Gambar 3. 26 *Deployment Diagram*

3.2 Perancangan Sistem

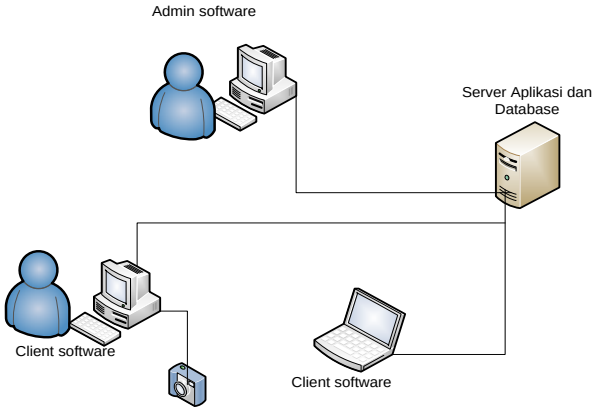
Perancangan merupakan bagian dari metodologi pembangunan suatu perangkat lunak yang harus dilakukan setelah melalui tahapan analisis. Pada bagian ini akan dijelaskan perancangan sistem yang dimaksudkan untuk menggambarkan perbedaan antara sistem yang sedang berjalan dengan sistem yang diusulkan.

Langkah-langkah yang dilakukan dalam tahapan perancangan ini adalah sebagai berikut :

1. Perancangan Arsitektural Perangkat Lunak
2. Perancangan Struktur Menu
3. Perancangan Antarmuka Perangkat Lunak
4. Perancangan Pesan
5. Perancangan Jaringan Semantik
6. Perancangan Prosedural

3.2.1 Perancangan Arsitektur Perangkat Lunak

Perancangan arsitektur merupakan perancangan yang dibuat sebelum perangkat lunak dibangun. Dalam perangkat lunak virtual aksesories dibangun atas tiga komponen yakni komponen basisdata, komponen perangkat tengah dan komponen *web browser*. Berikut ini gambar dari perancangan arsitektur perangkat lunak yang dapat dilihat pada gambar 3.28 di bawah ini.

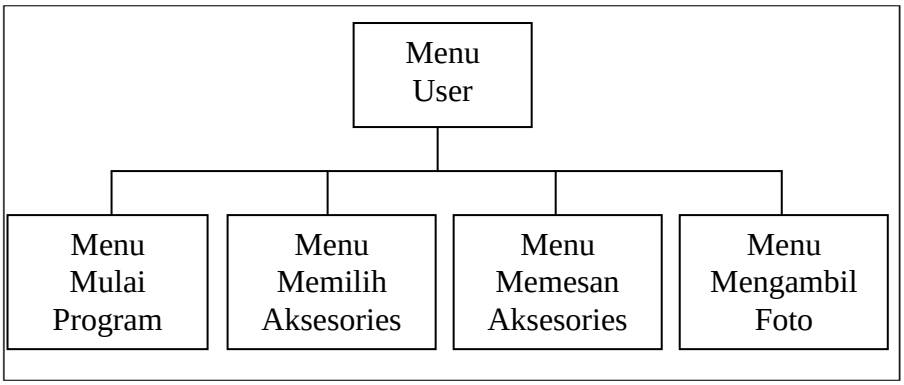


Gambar 3. 27 Arsitektural Perangkat Lunak

3.2.2 Perancangan Struktur Menu

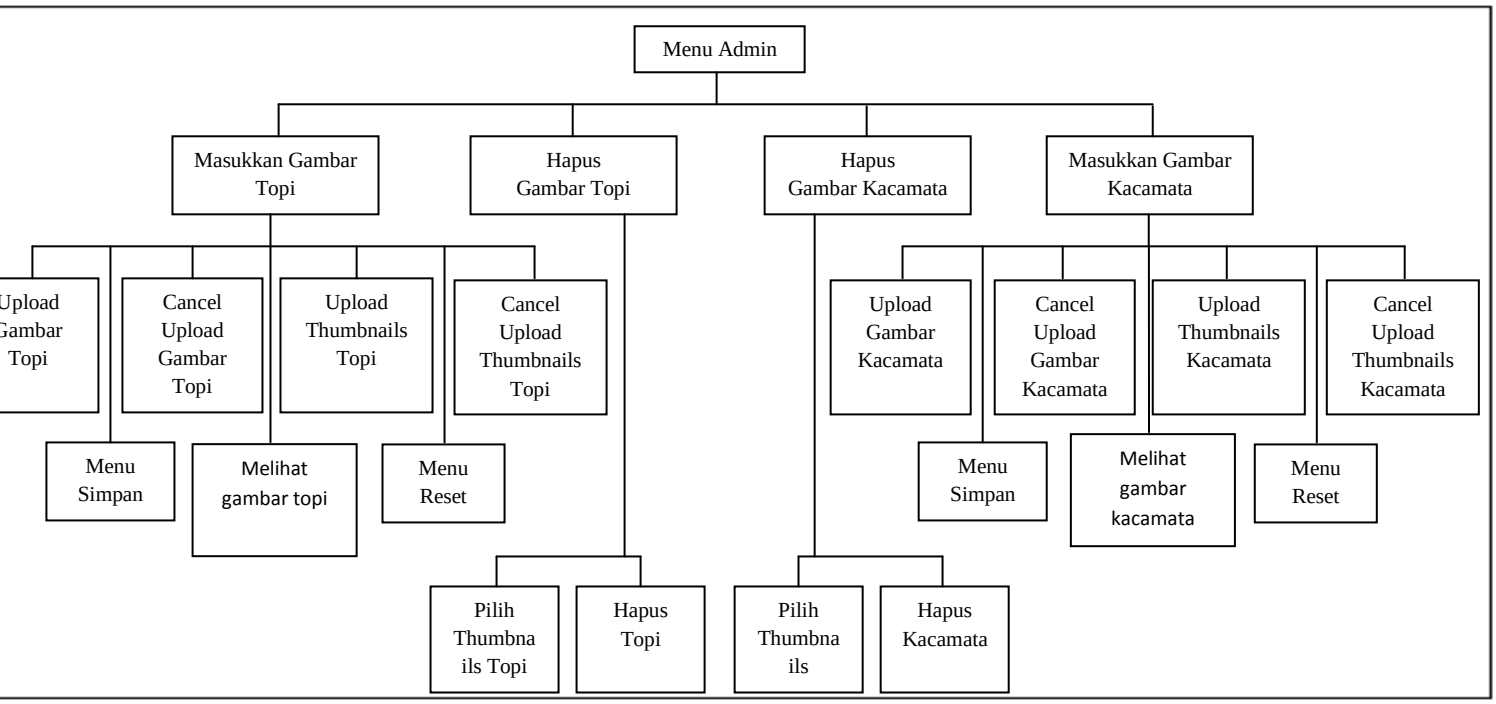
Struktur menu dirancang sesuai dengan level pengguna sistem. Terdapat dua pengguna perangkat ini yaitu *user* dan admin. Adapun struktur menunya dapat dilihat pada gambar di halaman berikutnya.

1. Struktur Menu *User*



Gambar 3. 28 Struktur Menu User

2. Struktur Menu Admin



Gambar 3. 29 Struktur Menu Admin

3.2.3 Perancangan Antarmuka Perangkat Lunak

Interface atau antar muka merupakan tampilan dari suatu perangkat lunak yang berperan sebagai media komunikasi yang digunakan sebagai sarana

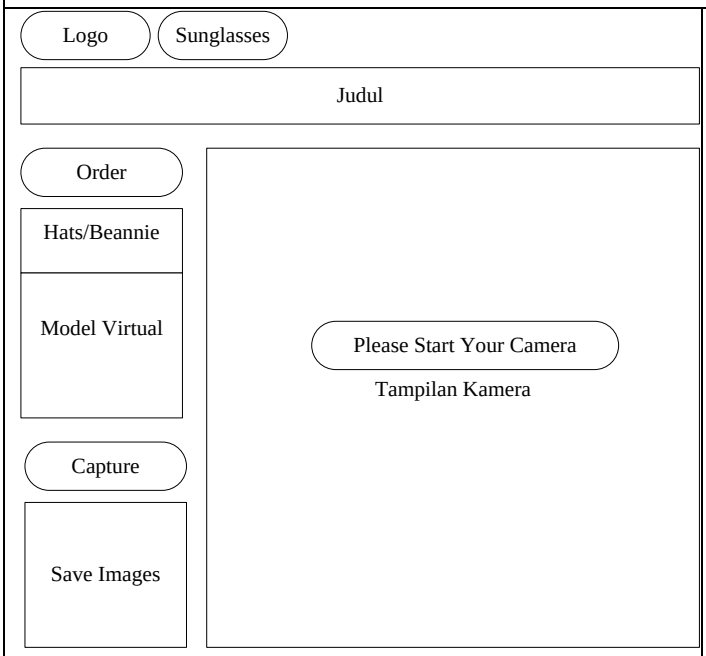
berdialog antara perangkat lunak dengan *user*. Sistem yang akan dibangun diharapkan menyediakan *interface* yang mudah dipahami dan digunakan oleh *user*. Perancangan antarmuka perangkat lunak pada virtual aksesories ini meliputi dua perancangan antarmuka yaitu, perancangan antarmuka *user* dan perancangan antarmuka admin.

3.2.3.1 Perancangan Antarmuka *User*

Perancangan antarmuka *user* adalah tampilan untuk user mencoba virtual aksesories yang terdiri dari dua layar yaitu untuk topi dan kacamata yang dapat dilihat pada gambar di bawah ini.

3.2.3.1.1 Antarmuka Topi

T01



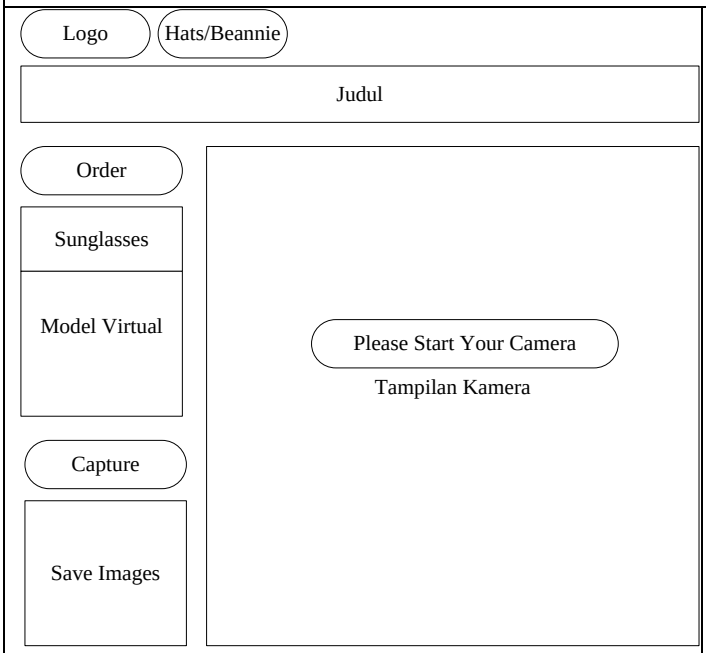
- Klik Logo untuk ke http://www.heavenskateboards.com
- Klik *Sunglasses* untuk melihat jenis kacamata
- Klik *Please Start Your Camera* untuk memulai aplikasi
- Klik *Order* untuk memesan topi
- Klik Model Virtual untuk

	melihat topi virtual - Klik *Capture* untuk memotret gambar dan menyimpannya pada *Save Images*
1. Ukuran 1024x768 pixel 2. Font : Calibri, 20 3. Warna : #FFFFFF, #000000, #0066FF 4. Gambar : bg.jpg, iconlogo,png, bdg_heavenskateboards.jpg, h1.jpg, h2.png, h22.png, h3.jpg, Untitlaed.jpg	

Gambar 3. 30 Antarmuka Topi

3.2.3.1.2 Antarmuka Kacamata

T01



- Klik Logo untuk ke http://www.heavenskateboards.com
- Klik *Hats/beannie* untuk melihat jenis topi yang dijual
- Klik *Please Start Your Camera* untuk memulai aplikasi
- Klik *Order* untuk memesan topi
- Klik Model Virtual untuk melihat topi virtual
- Klik *Capture* untuk memotret gambar dan menyimpannya pada *Save Images*

1. Ukuran 1024x768 pixel
2. Font : Calibri, 20
3. Warna : #FFFFFF, #000000, #0066FF
4. Gambar : bg.jpg, iconlogo,png, bdg_heavenskateboards.jpg, h1.jpg,

h2.png, h22.png, h3.jpg, Untitlaed.jpg

Gambar 3. 31 Antarmuka Kacamata

3.2.3.2 Perancangan Antarmuka Admin

Perancangan antarmuka admin adalah tampilan untuk admin mengelola model aksesories topi dan kacamata. Antarmuka admin memiliki enam antarmuka yaitu antarmuka login admin, antarmuka halaman utama, antarmuka memasukkan gambar topi, menghapus gambar topi, memasukkan gambar kacamata, dan menghapus gambar kacamata.

3.2.3.2.1 Perancangan Antarmuka Login Admin

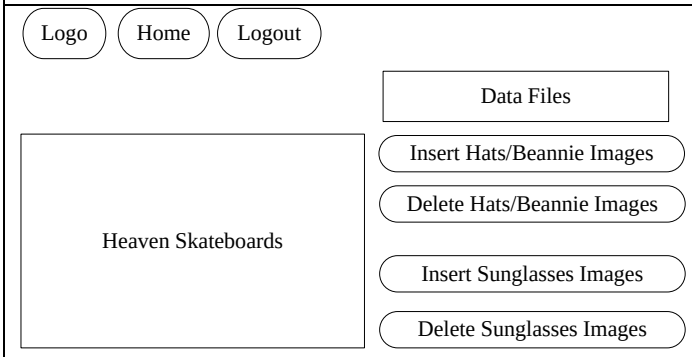
Perancangan antarmuka memasukkan gambar aksesories adalah tampilan pada admin untuk menambah gambar aksesories di perangkat lunak virtual aksesories yang dapat dilihat pada gambar 3.33 di bawah ini.

<table>
<tr><td colspan="2">A01</td></tr>
<tr><td>Admin Heaven Skateboards Login
Username
Password
Login</td><td>- Masukkan username dan password, lalu klik login untuk masuk ke A02</td></tr>
<tr><td colspan="2">1. Ukuran 1024x768 pixel
2. Font : Calibri, 20
3. Warna : #FFFFFF, #000000, #0066FF
4. Gambar : bg.jpg, iconlogo.png, h2.png</td></tr>
</table>

Gambar 3. 32 Perancangan Antarmuka Login Admin

3.2.3.2.2 Perancangan Antarmuka Halaman Utama

Perancangan antarmuka memasukkan gambar aksesories adalah tampilan pada admin untuk menambah gambar aksesories di perangkat lunak virtual aksesories yang dapat dilihat pada gambar 3.34 di bawah ini.

<table>
<tr><td colspan="2">A02</td></tr>
<tr><td>
</td><td>- Klik Logo untuk ke http://www.heavenskateboards.com
- Klik Home untuk masuk A02
- Klik Logout untuk kembali ke A01
- Klik Insert Hats/Beannie Images untuk masuk ke A03
- Klik Delete Hats/Beannie Images untuk masuk ke A04
- Klik Insert Sunglasses Images untuk masuk ke A05
- Klik Delete Sunglasses Images untuk masuk ke A06</td></tr>
</table>

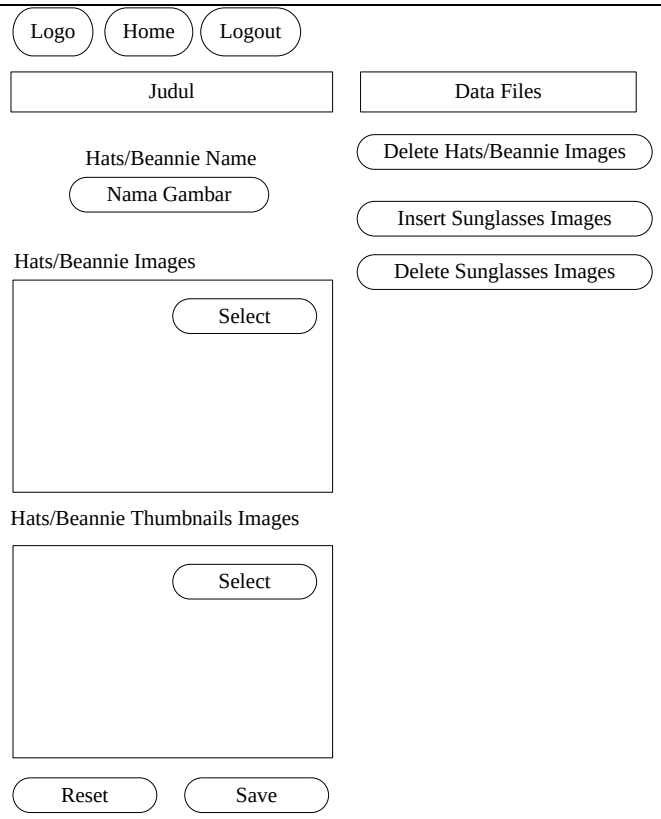
5. Ukuran 1024x768 pixel
6. Font : Calibri, 20
7. Warna : #FFFFFF, #000000, #0066FF
8. Gambar : bg.jpg, iconlogo.png, h2.png

Gambar 3. 33 Perancangan Antarmuka Halaman Utama

3.2.3.2.3 Perancangan Antarmuka Memasukkan Gambar Aksesories

Perancangan antarmuka memasukkan gambar aksesories adalah tampilan pada admin untuk menambah gambar aksesories yang terdiri dari memasukkan gambar topi dan memasukkan gambar kacamata pada perangkat lunak virtual aksesories yang dapat dilihat pada gambar 3.35 di halaman berikutnya.

A03



- Klik Logo untuk ke http://www.heavenskateboards.com
- Klik Home untuk masuk A02
- Klik *Logout* untuk kembali ke A01
- Klik *Delete Hats/Beannie Images* untuk masuk ke A04
- Klik *Insert Sunglasses Images* untuk masuk ke A05
- Klik *Delete*

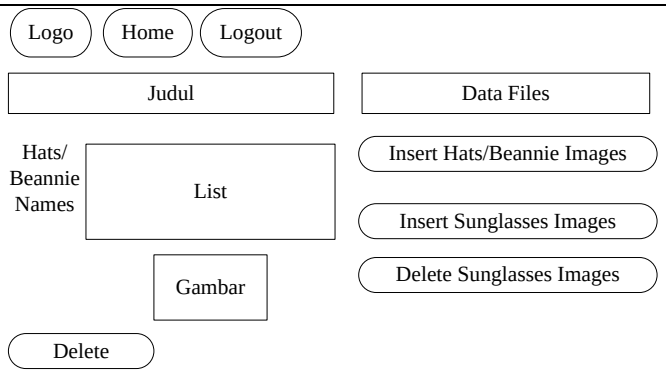
	Sunglasses Images untuk masuk ke A06 - Ketik pada nama gambar untuk memasukkan nama gambar - Klik select pada *Hats/Beannie Images* untuk upload gambar topi - Klik select pada *Hats/Beannie Thumbnails Images* untuk upload gambar Thumbnails topi - Klik *Reset* untuk mengulang - Klik *Save* untuk menyimpan gambar
9. Ukuran 1024x768 pixel 10. Font : Calibri, 20 11. Warna : #FFFFFF, #000000, #0066FF 12. Gambar : bg.jpg, iconlogo.png, h2.png	

Gambar 3. 34 Perancangan Antarmuka Memasukkan Gambar Topi

3.2.3.2.4 Perancangan Antarmuka Menghapus Gambar Topi

Perancangan antarmuka memasukkan gambar aksesories adalah tampilan pada admin untuk menambah gambar aksesories di perangkat lunak virtual aksesories yang dapat dilihat pada gambar 3.36 di bawah ini.

A04



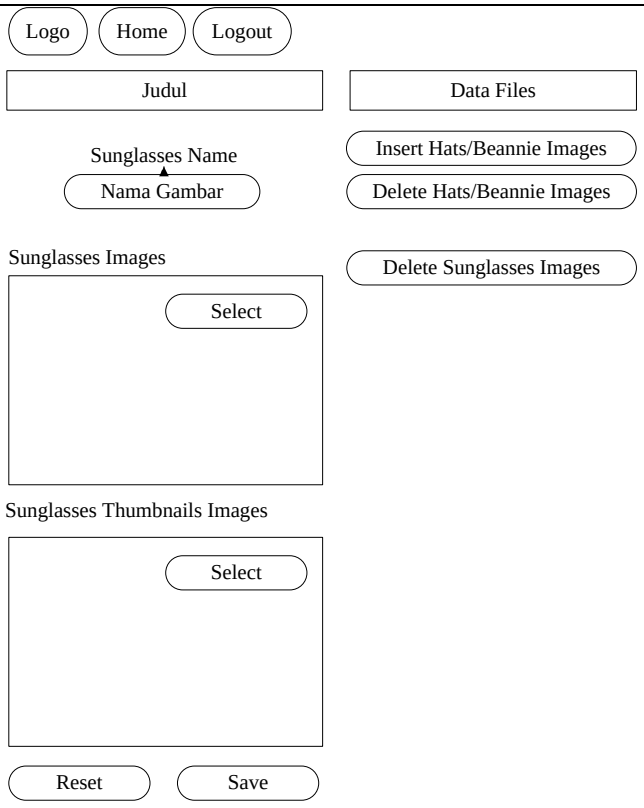
- Klik Logo untuk ke http://www.heavenskateboards.com
- Klik Home untuk masuk A02
- Klik *Logout* untuk kembali ke A01
- Klik *Insert Hats/Beannie Images* untuk masuk ke A03
- Klik *Insert Sunglasses Images* untuk masuk ke A05
- Klik *Delete Sunglasses Images* untuk masuk ke A06
- Klik *List* untuk memilih gambar yang akan dihapus
- Klik *Delete* unutk

	menghapus gambar
13. Ukuran 1024x768 pixel 14. Font : Calibri, 20 15. Warna : #FFFFFF, #000000, #0066FF 16. Gambar : bg.jpg, iconlogo.png, h2.png	

Gambar 3. 35 Perancangan Antarmuka Menghapus Gambar Topi

3.2.3.2.5 Perancangan Antarmuka Memasukkan Gambar Kacamata

Perancangan antarmuka memasukkan gambar aksesories adalah tampilan pada admin untuk menambah gambar aksesories di perangkat lunak virtual aksesories yang dapat dilihat pada gambar 3.37 pada halaman berikutnya.

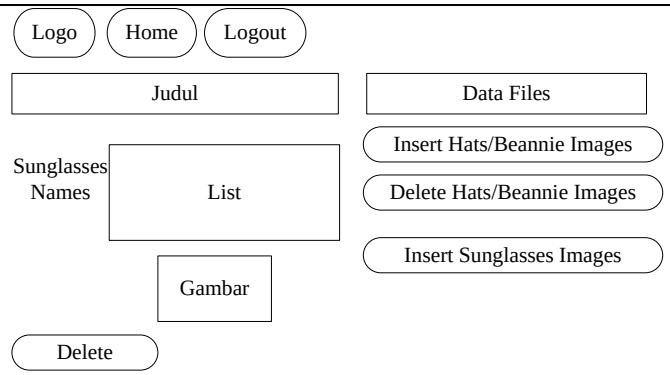
A05	
	- Klik Logo untuk ke http://www.heavenskateboards.com - Klik Home untuk masuk A02 - Klik *Logout* untuk kembali ke A01 - Klik *Insert Hats/Beannie Images* untuk masuk ke A03 - Klik *Delete Hats/Beannie Images* untuk

	masuk ke A04 - Klik *Delete Sunglasses Images* untuk masuk ke A06 - Ketik pada nama gambar untuk memasukkan nama gambar - Klik select pada *Sunglasses Images* untuk upload gambar kacamata - Klik select pada *Sunglasses Thumbnails Images* untuk upload gambar Thumbnails kacamata - Klik *Reset* untuk mengulang - Klik *Save* untuk menyimpan gambar
17. Ukuran 1024x768 pixel 18. Font : Calibri, 20 19. Warna : #FFFFFF, #000000, #0066FF 20. Gambar : bg.jpg, iconlogo.png, h2.png	

Gambar 3. 36 Perancangan Antarmuka Memasukkan Gambar Kacamata

3.2.3.2.6 Perancangan Antarmuka Menghapus Gambar Kacamata

Perancangan antarmuka memasukkan gambar aksesories adalah tampilan pada admin untuk menambah gambar aksesories di perangkat lunak virtual aksesories yang dapat dilihat pada gambar 3.38 di halaman berikutnya.

A06	
	- Klik Logo untuk ke http://www.heavenskateboards.com - Klik Home untuk masuk A02 - Klik *Logout* untuk kembali ke A01 - Klik *Insert Hats/Beannie Images* untuk masuk ke A03 - Klik *Delete Hats/Beannie Images* untuk masuk ke A04 - Klik *Insert Sunglasses Images* untuk masuk ke A05 - Klik *List* untuk memilih gambar yang akan dihapus - Klik *Delete* unutk

	menghapus gambar
21. Ukuran 1024x768 pixel 22. Font : Calibri, 20 23. Warna : #FFFFFF, #000000, #0066FF 24. Gambar : bg.jpg, iconlogo.png, h2.png	

Gambar 3. 37 Perancangan Antarmuka Menghapus Gambar Kacamata

3.1.1 Perancangan Pesan

Tabel 3.19 Pada halaman berikutnya merupakan tabel perancangan pesan yang terdapat pada apliaksi virtual aksesories.

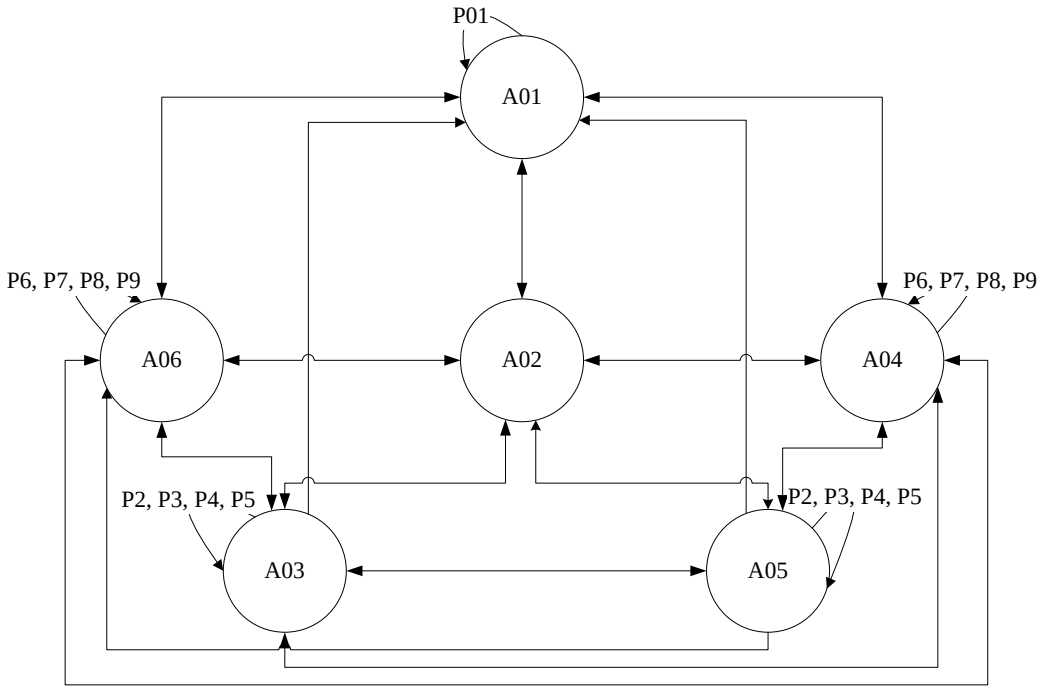
Tabel 3. 19 Perancangan Pesan

No	Text Pesan	Keterangan
P01	Salah memasukkan *username* atau *password*	Pesan kesalahan di A01
P02	Isi data terlebih dahulu !	Pesan kesalahan di A03 dan A05
P03	File tidak dapat diupload !	Pesan kesalahan di A03 dan A05
P04	Upload berhasil	Pesan berhasil di A03 dan A05
P05	Proses simpan berhasil	Pesan berhasil di A03 dan A05
P06	Proses simpan gagal !	Pesan kesalahan di A04 dan A06

P07	Pilih nama terlebih dahulu !	Pesan kesalahan di A04 dan A06
P08	Proses hapus berhasil	Pesan berhasil di A04 dan A06
P09	Proses hapus gagal !	Pesan kesalahan di A04 dan A06

3.1.2 Perancangan Jaringan Semantik

Jaringan semantik merupakan gambaran hubungan antara form dan pesan. Jaringan semantik yang dijelaskan pada gambar 3.40 pada halaman berikutnya adalah jaringan semantik untuk admin.



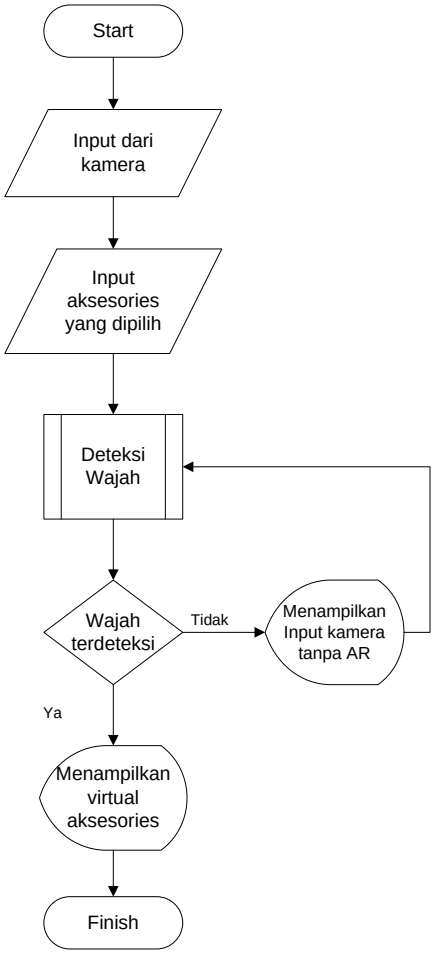
Gambar 3. 38 Jaringan Semantik

3.1.3 Perancangan Prosedural

Perancangan prosedural ini akan digunakan sebagai algoritma dasar dalam mengkodekan prosedur yang ada. Adapun perancangan prosedural untuk

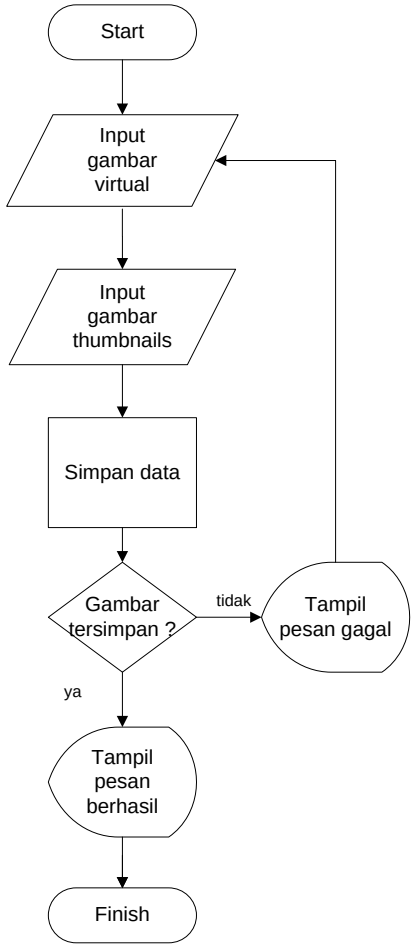
pembangunan aplikasi virtual aksesories ini yang dapat dilihat pada gambar di bawah ini.

1. Prosedural Penggunaan Virtual Aksesories



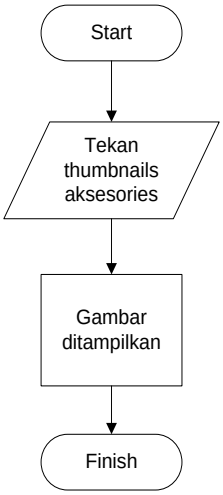
Gambar 3. 39 Proseduran Penggunaan Virtual Aksesories

2. Prosedural Memasukkan Gambar Aksesories



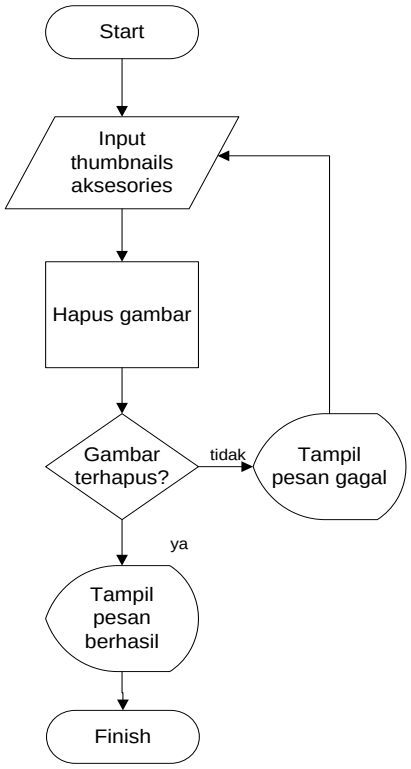
Gambar 3. 40 Prosedural Memasukkan Gambar Aksesories

3. Prosedural Melihat Hasil Upload Gambar Aksesories



Gambar 3. 41 Prosedural Melihat Hasil Upload Gambar Aksesories

4. Prosedural Menghapus Gambar Aksesories



Gambar 3. 42 Prosedural Menghapus Gambar Aksesories