BAB III

ANALISIS DAN PERANCANGAN SISTEM

Bab ini memberikan penjelasan mengenai analisis dan perancangan dari aplikasi yang akan dibangun. Analisis dan perancangan merupakan dua tahap awal dalam membangun suatu perangkat lunak. Dua tahap ini merupakan landasan untuk berlanjut ke tahap implementasi.

3.1 Analisis Masalah

Di zaman modern yang serba instan dan cepat ini, semakin mudah pertukaran informasi dewasa ini tidak hanya membawa dampak positif bagi kemajuan teknologi, tetapi juga membawa dampak negatif yang hampir tidak dapat dihindari yaitu plagiarisme. Praktek plagiarisme ini sangat sering dilakukan terutama di kalangan akademisi baik sekolah maupun di perguruan tinggi. Tindakan plagiarisme ini dapat berdampak pada penurunan kreatifitas siswa maupun mahasiswa. Teknik Plagiarisme yang paling sering dilakukan dengan melakukan *copy paste* suatu dokumen. Cukup dengan menggunakan teknik *copy paste* tugas milik teman, tugas-tugas sekolah atau kuliah dapat terselesaikan.

Permasalahannya, untuk mengatasi praktek plagiarisme terbukti cara *persuasif* disini maksudnya pendekatan langsung kepada mahasiswa yang bersangkutan bahwa tindakan plagiarisme tersebut tidak baik dilakukan sudah terbukti tidak efektif. Sistem pengukuran tingkat similaritas dokumen merupakan solusi tepat yang sebaiknya dilakukan agar tindakan curang tersebut dapat

diminimalisasi. Oleh karena itu untuk meminimalisasi praktik plagiarisme tersebut diperlukan sistem pengukuran tingkat similaritas dokumen terhadap karya tulis seseorang.

Berdasarkan permasalahan tersebut maka akan dirancang dan dibangun suatu "Sistem Pengukuran Tingkat Similaritas Dokumen". Teknik yang digunakan dalam perancangan aplikasi yaitu dengan membandingkan kemiripan dokumen asli dengan dokumen yang ingin diuji. Sebelum adanya aplikasi pengukuran tingkat similaritas dokumen ini, memeriksa dokumen membutuhkan waktu yang cukup lama dengan harus membaca *hard copy*. Dengan adanya aplikasi ini, sekumpulan *file* dokumen dapat diuji apakah antar dokumen dan yang lainnya memiliki kesamaan atau tidak, diharapkan dengan adanya aplikasi ini praktek plagiarisme dapat dicegah.

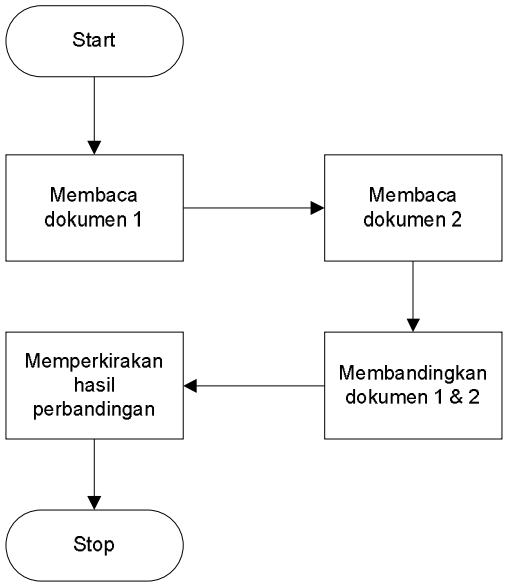
3.1.1 Analisis Metode

Analisis fungsional sistem merupakan penjelasan bagaimana kerja sistem pengukuran tingkat similaritas dokumen ini, pada sub bab ini akan dijelaskan bagaimana kerja sistem yang masih manual, dan bagaimana kerja sistem dengan kemudahan yang ditawarkan setelah adanya aplikasi pengukuran tingkat similaritas dokumen.

a. Sistem Manual

   Pada gambar dibawah dapat dilihat alur kerja sistem yang masih manual yang akan dilakukan oleh user, dan dapat diperkirakan akan memakan waktu yang lama untuk membandingkan dokumen yang satu dengan

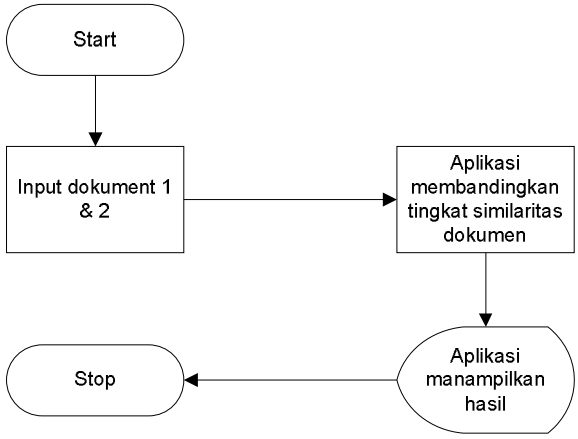
yang lainnya. Selain itu hasil yang diperoleh belum tentu akurat, karena hasil perbandingan yang didapat hanya perkiraan saja dan tanpa menggunakan metode tertentu, dapat dilihat pada gambar 3.1 berikut.



Gambar 3.1 Sistem Manual

b. Aplikasi Similaritas Dokumen

Dokumen merupakan data yang diuji dalam sistem ini adalah berupa dokumen teks dengan membandingkan hasil similaritas. Dengan adanya aplikasi ini pengguna tidak perlu membaca dan berusaha membandingkan dokumen, karena semua itu akan dilakukan oleh aplikasi ini, pengguna cukup memasukkan dokumen yang akan dibandingkan dan langsung dapat melihat hasil perbandingan, dapat dilihat pada gambar 3.2.



Gambar 3.2 Aplikasi Similaritas Dokumen

3.1.2 Analisis Algoritma

Aplikasi yang dibuat adalah berupa aplikasi pengukuran tingkat similaritas dokumen. Inputan pada aplikasi ini berupa dokumen teks yang mempunyai ekstensi .doc, .docx, .txt, dan .pdf. User akan menginputkan 2 dokumen, yaitu dokumen asli dan dokumen yang ingin diuji. Setelah itu, sistem akan memproses kedua dokumen tersebut dan mengevaluasi berapa persen nilai similaritas antara dokumen tersebut. Proses yang pertama kali dilakukan oleh sistem adalah membaca *file* teks yang diinputkan oleh user. Setelah dokumen diinputkan oleh user tadi, sistem akan melakukan pengecekan terhadap dokumen tersebut sehingga akan didapatkan informasi berupa jumlah kata, jumlah kalimat, jumlah paragraf dan ukuran dokumen tersebut.

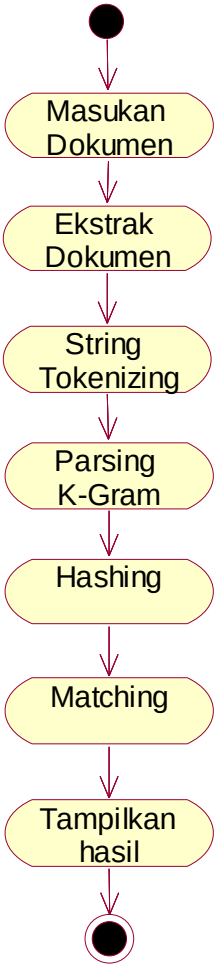
Setelah sistem mendapatkan informasi dari dokumen yang telah diinputkan, selanjutnya sistem akan masuk ke tahap *string tokenizing.* Pada tahap ini akan dilakukan beberapa proses, yaitu pengambilan info dokumen dan *tokenizing.* Proses *tokenizing* adalah proses penghilangan kata-kata dan tanda baca

yang kurang penting, seperti kata spasi, titik, koma, dan sebagainya dan mengubah hurup menjadi *lower case* semua. Contoh berikut adalah dokumen yang sebelum dilakukan *string tokenizing* : Di zaman modern yang serba instan dan cepat ini, Semakin mudahnya pertukaran informasi dewasa ini tidak hanya membawa dampak positif bagi kemajuan teknologi, tetapi juga membawa dampak negatif yang hampir tidak dapat dihindari yaitu plagiarisme. Setelah dilakukan *string tokenizing* maka dokumennya akan berubah menjadi seperti ini : dizamanmodernyangserbainstandancepatinisemakinmudahnyapertukaraninformas idewasainitidakhanyamembawadampakpositifbagikemajuanteknologitetapijugame mbawadampaknegatifyanghampirtidakdapatdihindariyaituplagiarisme.

Setelah melakukan *string tokenizing* langkah selanjutnya adalah *parsing k-gram*, yaitu memecah kata menjadi potongan-potongan dimana setiap potongan mengandung karakter sebanyak *k*.

Setelah melakukan *parsing k-gram* langkah berikutnya adalah menghitung nilai hashing. Melakukan proses *Hashing* terhadap seluruh pecahan string tadi yang telah dibagi menjadi k bagian.

Setelah menghitung nilai hashing langkah berikutnya adalah melakukan pencocokan nilai hash atau *hash matching*.



Gambar 3.3 *Activity* Menghitung Nilai Similaritas

Untuk lebih jelas mengenai proses menghitung nilai similaritas maka akan diperlihatkan contoh kasus dan cara menghitungnya dengan cara manual sebagai berikut:

Contoh kasus

Teks asli: Saya Dennis

Teks uji: Saya Ahmad

1. Tokenizing

   Setelah dilakukan tokenizing kalimatnya akan menjadi seperti ini.

   Teks asli: sayadennis

   Teks uji: sayaahmad

2. Parsing K-Gram

   Pembagian kalimat berdasarkan K-Gram.

   K-Gram = 4

   Maka kalimat akan menjadi potongan seperti dibawah ini:

   Tabel perbandingan teks asli dan teks uji.

Teks Asli	
no	**substring**
1	saya
2	ayad
3	yade
4	aden
5	denn
6	enni
7	nnis

Teks Uji	
no	**substring**
1	saya
2	ayaa
3	yaah
4	aahm
5	ahma
6	hmad

3. Hashing

Merupakan proses pengubahan karakter ke bilangan hash. Untuk melakukan pengubahan tersebut menggunakan rumus (II.1) sebagai berikut.

$$H = C_1 * a^{(k-1)} + C_2 * a^{(k-2)} + C_3 * a^{(k-3)} \ldots + C_k * a^0$$

Pola : “saya”

Bill_hash dari say $= s * 10^3 + a * 10^2 + y * 10^1 + a * 10^0$

= 115*1000 + 97*100 + 121*10 + 97*1

= 115000 + 9700 + 1210 + 97

= 126007

Pola : “ayad”

Bill_hash dari say $= a * 10^3 + y * 10^2 + a * 10^1 + d * 10^0$

= 97*1000 + 121*100 + 97*10 + 100*1

= 97000+ 12100 + 970 + 100

= 110170

Pola : “yade”

Bill_hash dari say $= y * 10^3 + a * 10^2 + d * 10^1 + e * 10^0$

= 121*1000 + 97*100 + 100*10 + 101*1

= 121000 + 9700 + 1000 + 101

= 131801

Dengan cara yang sama dilakukan pencarian nilai hash terhadap semua pola karakter satu demi satu.

Dengan melakukan perhitungan yang sama terhadap kalimat yang telah di K-Gram maka didapat nilai hasing sebagai berikut:

Tabel perbandingan teks asli dan teks uji setelah nilai hash didapatkan.

Teks Asli		
no	**substring**	**Hash**
1	saya	126007
2	ayad	110170
3	yade	131801
4	aden	108120
5	denn	111310
6	enni	113205
7	nnis	122165

Teks Uji		
no	**substring**	**Hash**
1	saya	126007
2	ayaa	110167
3	yaah	131774
4	aahm	107849
5	ahma	108587
6	hmad	115970

Keterangan : nilai hash yang sama.

Setelah nilai hash didapat semua, kemudian dicari nilai hash yang sama dari kedua dokumen. Seperti nilai hash yang ditandai warna hijau pada tabel di atas.

4. Pencocokan Nilai Hashing

Setelah itu dihitung ada berapa jumlah hash yang sama. Dengan cara

mencocokan satu demi satu.

Tabel pencocokan nilai hash.

126007	=	126007
126007	≠	110167
126007	≠	131774
126007	≠	107849
126007	≠	108587
126007	≠	115970

110170	≠	126007
110170	≠	110167
110170	≠	131774
110170	≠	107849
110170	≠	108587
110170	≠	115970

Dengan cara yang sama dilakukan pencocokan terhadap semua nilai hash satu demi satu.

5. Pengukuran Nilai Similaritas

$$S = \frac{N_t}{N_x + N_y}$$

$$= \frac{2}{7+6}$$

$$= 0{,}153846$$

Untuk mendapatkan persentase similaritas, kalikan dengan persen, jadi tingkat kesamaan dari teks yang di uji = 15.3846 %.

3.1.3 Analisis Kebutuhan Non Fungsional

Analisis kebutuhan non-fungsional merupakan proses indentifikasi dan evaluasi mengenai kebutuhan sistem maupun pengguna, sehingga diharapkan aplikasi yang dibangun berjalan dengan baik.

3.1.3.1 Analisis Perangkat Lunak (*Software*)

Kebutuhan *software* merupakan kebutuhan akan perangkat lunak yang digunakan untuk membangun program aplikasi pengukuran tingkat similaritas dokumen ini. Diantaranya kebutuhan tersebut adalah :

1. Sistem operasi berbasis *windows*, yang akan digunakan adalah sistem operasi *Windows 7 Ultimate.*
2. *Tools*/bahasa pemrograman, yang akan digunakan adalah *Java Programming*.

3.1.3.2 Analisis Perangkat Keras (*Hardware*)

Dalam perancangan aplikasi tingkat similaritas dokumen ini menggunakan komputer (PC) dengan spesifikasi:

1. Prosesor Core 2 Duo CPU E4500 @2.20 GHz
2. Memory 3GB, Harddisk 500GB

3. Monitor 17”, Motherboard dan keyboard.

3.1.3.3 Analisis Pengguna

Terdapat 2 pengguna di dalam sistem yang akan di buat ini yaitu :

1. Visitor
2. Admin

Pengguna inilah yang akan menjadi aktor dalam analisis ataupun perancangan yang memiliki tugasnya masing-masing, Visitor merupakan pengguna yang hanya menggunakan aplikasi ini hanya sebagai user. Visitor dapat memilih jenis Scan yang akan digunakan dan kemudian memilih atau memasukkan dokumen yang akan dibandingkan dan kemudian melihat hasilnya yaitu berapa persen tingkat kesamaan dokumen yang dibandingkan. Sedangkan seorang admin berperan sebagai pengolahan berbagai kegiatan yang ada pada aplikasi seperti mengelola admin dan mengelola dokumen.

3.1.4 Analisis Kebutuhan Fungsional

Analisis kebutuhan fungsional akan membahas perancangan sistem di dalam aplikasi. Perancangan sistem adalah penggambaran, perancangan dan pembuatan sketsa. Pada tahap ini, *tools*/alat bantu yang digunakan untuk menganalisis sistem adalah UML (*Unified Modelling Language*).

3.1.4.1 *Use Case Diagram*

Use Case Diagram meliputi definisi aktor, use case dan skenario. Aktor

disini ada 2 yaitu admin dan visitor, dimana admin bisa mengelola admin dan mengelola dokumen sedangkan visitor bisa melakukan *quick scan* dan *normal scan* dokumen.

3.1.4.2 Definisi Aktor

Kebutuhan fungsional Merupakan kebutuhan yang harus dimiliki oleh perangkat lunak yang akan dibangun. Kebutuhan fungsional dalam aplikasi di bagi menjadi 2 bagian. Yaitu bagian pengguna (visitor) dan admin. Kebutuhan fungsional tersebut akan dideskripsikan dalam bentuk Tabel, sebagai berikut:

1. Kebutuhan Fungsional Admin

Admin	
Kode	Kebutuhan Fungsional
ADM-01	Login sebagai Admin
ADM-02	Penyajian Profil
ADM-03	Pengelolaan Admin
ADM-04	Pengelolaan Dokumen
ADM-05	Pengelolaan Subdokumen

Tabel 3.1 Kebutuhan Fungsional Admin

2. Kebutuhan Fungsional Visitor

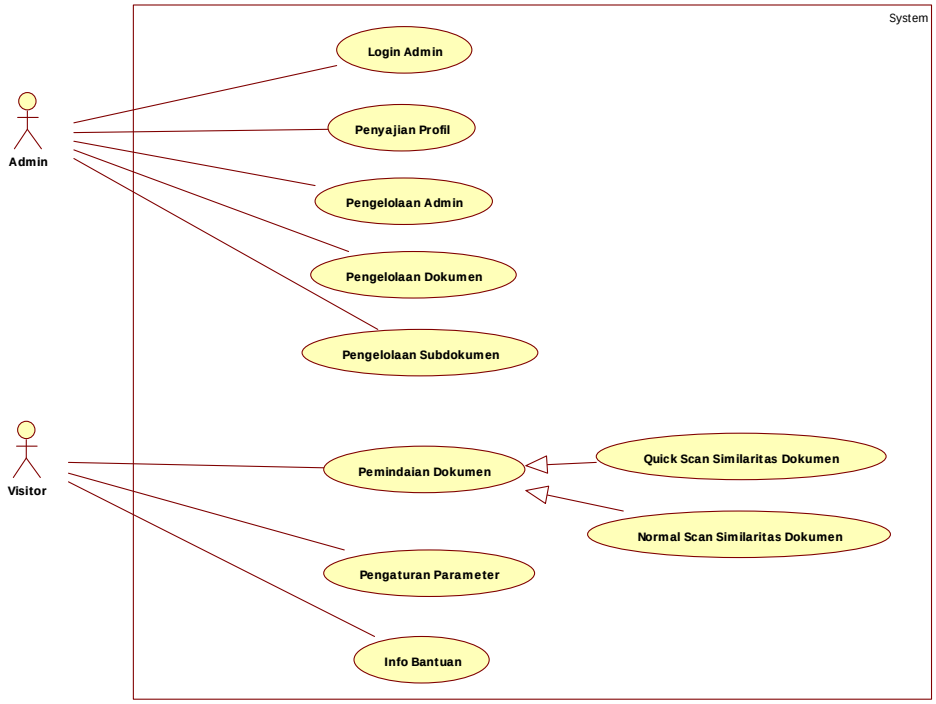
Visitor	
Kode	**Kebutuhan Fungsional**
VST-01	Quick scan similaritas dokumen

VST-02	Normal scan similaritas dokumen
VST-03	Pengaturan parameter
VST-04	Info bantuan

Tabel 3.2 Kebutuhan Fungsional Visitor

3.1.4.3 Definisi *Use Case*

Diagram *Use Case* digunakan untuk memodelkan bisnis proses berdasarkan perspektif pengguna sistem. Use case diagram terdiri atas diagram untuk use case dan *actor*. *Actor* merepresentasikan orang yang akan mengoperasikan atau orang yang berinteraksi dengan sistem aplikasi, sesuai analisis aktor, maka use case diagramnya memiliki 2 aktor yaitu visitor dan admin.



Gambar 3.4 *Use Case* Diagram

3.1.4.4 Definisi Skenario

Skenario merupakan penjelasan mengenai jalan cerita / alur dari masing-masing *use case*.

1. Admin

a. Login

Identifikasi			
Nomor	ADM-01		
Nama	Login		
Tujuan	Memasuki aplikasi sebagai admin.		
Deskripsi	Proses memasuki aplikasi sebagai admin.		
Aktor	Admin		
Skenario Utama			
Kondisi Awal	Form login telah aktif. Admin belum melakukan login.		
No	**Aksi Aktor**	**No.**	**Reaksi Sistem**
1.	Masukkan *username* dan *password*, klik "Login".		
		2.	Sistem validasi *username* dan *password* dengan data yang ada di *database*.
		3.	Sistem menampilkan menu "Manage Document".

Kondisi Akhir	Admin telah berhasil login dan menu “Manage Document” aktif.
Eksepsi	Jika *username* dan *password* tidak sesuai, muncul pesan bahwa login gagal.

Tabel 3.3 Skenario Login

b. Lihat Profil

Identifikasi			
Nomor	ADM-02		
Nama	Penyajian Profil		
Tujuan	Menampilkan profil admin yang login		
Deskripsi	Proses menampilkan profil admin yang sedang login.		
Aktor	Admin		
Skenario Utama			
Kondisi Awal	Admin telah berhasil login.		
No	**Aksi Aktor**	**No.**	**Reaksi Sistem**
1.	Klik “*Home*”.		
		2.	Sistem memuat (*load*) profil admin dari *database*, dan menyajikannya.
Kondisi Akhir	Profil admin tampil di halaman home admin.		
Eksepsi	-		

Tabel 3.4 Skenario Lihat profil

c. Kelola Admin

Identifikasi	
Nomor	ADM-03

Nama	Pengelolaan Admin		
Tujuan	Mengelola data admin.		
Deskripsi	Proses menampilkan list admin untuk kemudian mengelolanya.		
Aktor	Admin		
Skenario Utama			
Kondisi Awal	Menu “Manage Admin” telah aktif.		
No	**Aksi Aktor**	**No.**	**Reaksi Sistem**
1.	Klik “*Manage Admin*”.		
		2.	Sistem memuat (*load*) admin dari *database*, dan menyajikannya.
Kondisi Akhir	Data admin telah tampil dalam bentuk list		
Eksepsi	-		

Tabel 3.5 Skenario Kelola Admin

d. Input Admin

Identifikasi	
Nomor	ADM-03-1
Nama	Input Admin
Tujuan	Menambah data admin.
Deskripsi	Proses menambahkan data admin yang baru ke *database*.
Aktor	Admin
Skenario Utama	
Kondisi Awal	Halaman kelola admin telah aktif.

No	Aksi Aktor	No.	Reaksi Sistem
1.	Klik "*Add Admin*".		
		2.	Sistem menyajikan form input admin.
3.	Input admin baru.		
		4.	Admin yang baru telah disimpan di *database*.
Kondisi Akhir	Admin telah diinputkan di *database*.		
Eksepsi	Jika pengisian field pada form tidak sesuai aturan, muncul pesan bahwa dokumen tidak berhasil diinputkan.		

Tabel 3.6 Skenario Input Admin

e. Edit Admin

Identifikasi	
Nomor	ADM-03-2
Nama	Edit Admin
Tujuan	Mengedit data admin tertentu.
Deskripsi	Proses mengedit data admin tertentu.
Aktor	Admin
Skenario Utama	
Kondisi Awal	Halaman kelola admin telah aktif. List dokumen telah ditampilkan.

No	Aksi Aktor	No.	Reaksi Sistem
1.	Klik "*Edit*".		
		2.	Sistem menyajikan form edit admin berikut data admin yang akan diedit.

3.	Edit data admin sesuai kebutuhan, klik “Update”.		
		4.	Sistem update admin di *database*, munculkan pesan proses edit berhasil.
Kondisi Akhir	Data admin tertentu telah diupdate di *database*.		
Eksepsi	Jika pengisian field pada form tidak sesuai aturan, muncul pesan proses edit gagal.		

Tabel 3.7 Skenario Edit Admin

f. Hapus Admin

Identifikasi			
Nomor	ADM-03-3		
Nama	Hapus Admin		
Tujuan	Menghapus data admin tertentu.		
Deskripsi	Proses menghapus data admin tertentu dari *database*.		
Aktor	Admin		
Skenario Utama			
Kondisi Awal	Halaman kelola admin telah aktif. List dokumen telah ditampilkan.		
No	**Aksi Aktor**	**No.**	**Reaksi Sistem**
1.	Klik “Hapus”.		
		2.	Sistem menyajikan form konfirmasi hapus data admin.

3.	Konfirmasi hapus data admin.		
		4.	Sistem menghapus data admin tertentu dari *database*.
Kondisi Akhir	Data admin tertentu telah dihapus dari *database*.		
Eksepsi	Jika data admin sedang digunakan, muncul pesan bahwa data admin masih digunakan.		

Tabel 3.8 Skenario Hapus Admin

g. Kelola Dokumen

Identifikasi			
Nomor	ADM-04		
Nama	Pengelolaan Dokumen		
Tujuan	Mengelola data dokumen.		
Deskripsi	Proses menampilkan list dokumen untuk kemudian mengelolanya.		
Aktor	Admin		
Skenario Utama			
Kondisi Awal	Menu “Manage Document” telah aktif.		
No	**Aksi Aktor**	**No.**	**Reaksi Sistem**
1.	Klik “*Manage Document*”.		
		2.	Sistem memuat (*load*) data dokumen dari *database*, dan menyajikannya.
Kondisi Akhir	Data dokumen telah tampil dalam bentuk list		

Eksepsi	-

Tabel 3.9 Skenario Kelola Dokumen

h. Input Dokumen

Identifikasi			
Nomor	ADM-04-1		
Nama	Input Dokumen		
Tujuan	Menambah data dokumen.		
Deskripsi	Proses menambahkan data dokumen yang baru ke *database*.		
Aktor	Admin		
Skenario Utama			
Kondisi Awal	Halaman kelola dokumen telah aktif.		
No	**Aksi Aktor**	**No.**	**Reaksi Sistem**
1.	Klik "*Add Document*".		
		2.	Sistem menyajikan form input dokumen.
3.	Input data dokumen baru.		
		4.	Data dokumen yang baru telah disimpan di *database*.
Kondisi Akhir	Data dokumen telah diinputkan di *database*.		
Eksepsi	Jika pengisian field pada form tidak sesuai aturan, muncul pesan bahwa data dokumen tidak berhasil diinputkan.		

Tabel 3.10 Skenario Input Dokumen

i. Edit Dokumen

Identifikasi			
Nomor	ADM-04-2		
Nama	Edit Dokumen		
Tujuan	Mengedit data dokumen tertentu.		
Deskripsi	Proses mengedit data dokumen tertentu.		
Aktor	Admin		
Skenario Utama			
Kondisi Awal	Halaman kelola dokumen telah aktif. List dokumen telah ditampilkan.		
No	**Aksi Aktor**	**No.**	**Reaksi Sistem**
1.	Klik *"Edit"*.		
		2.	Sistem menyajikan form edit dokumen berikut data dokumen yang akan diedit.
3.	Edit data dokumen sesuai kebutuhan, klik "Update".		
		4.	Sistem update data dokumen di *database*, munculkan pesan proses edit berhasil.
Kondisi Akhir	Data dokumen tertentu telah diupdate di *database*.		
Eksepsi	Jika pengisian field pada form tidak sesuai aturan, muncul pesan proses edit gagal.		

Tabel 3.11 Skenario Edit Dokumen

j. Hapus Dokumen

Identifikasi			
Nomor	ADM-04-3		
Nama	Hapus Dokumen		
Tujuan	Menghapus data dokumen tertentu.		
Deskripsi	Proses menghapus data dokumen tertentu dari *database*.		
Aktor	Admin		
Skenario Utama			
Kondisi Awal	Halaman kelola dokumen telah aktif. List dokumen telah ditampilkan.		
No	**Aksi Aktor**	**No.**	**Reaksi Sistem**
1.	Klik "Hapus".		
		2.	Sistem menyajikan form konfirmasi hapus data dokumen.
3.	Konfirmasi hapus data dokumen.		
		4.	Sistem menghapus data dokumen tertentu dari *database*.
Kondisi Akhir	Data dokumen tertentu telah dihapus dari *database*.		
Eksepsi	Jika data dokumen masih digunakan, muncul pesan bahwa data dokumen masih digunakan.		

Tabel 3.12 Skenario Hapus Dokumen

k. Kelola Subdokumen

Identifikasi			
Nomor	ADM-05		
Nama	Pengelolaan Subdokumen		
Tujuan	Mengelola data subdokumen.		
Deskripsi	Proses menampilkan list subdokumen untuk mengelolanya.		
Aktor	Admin		
Skenario Utama			
Kondisi Awal	Halaman kelola subdokumen telah aktif.		
No	**Aksi Aktor**	**No.**	**Reaksi Sistem**
1.	Klik "*Subdocument*".		
		2.	Sistem memuat (*load*) data subdokumen dari *database*, dan menyajikannya.
Kondisi Akhir	Data subdokumen telah ditampilkan dalam bentuk list.		
Eksepsi	-		

Tabel 3.13 Skenario Kelola Subdokumen

l. Input Subdokumen

Identifikasi	
Nomor	ADM-05-1
Nama	Input Subdokumen
Tujuan	Menambah data subdokumen.
Deskripsi	Proses menambah data subdokumen yang baru ke database.

Aktor		Admin	
Skenario Utama			
Kondisi Awal		Halam kelola subdokumen telah aktif.	
No	**Aksi Aktor**	**No.**	**Reaksi Sistem**
1.	Klik button "*Add Subdocument*.		
		2.	Sistem menyajikan form input subdokumen.
3.	Input data subdokumen yang baru.		
		4.	Data subdokumen yang baru telah disimpan di *database*.
Kondisi Akhir		Data subdokumen telah diinputkan di *database*.	
Eksepsi		Jika pengisian field pada form tidak sesuai aturan, muncul pesan bahwa subdokumen tidak berhasil diinputkan.	

Tabel 3.14 Skenario Input Subdokumen

m. Lihat Konten

Identifikasi	
Nomor	ADM-05-2
Nama	Lihat Konten
Tujuan	Menampilkan konten dari subdokumen.
Deskripsi	Proses menampilkan konten dari subdokumen tertentu.
Aktor	Admin

Skenario Utama			
Kondisi Awal	Halaman kelola subdokumen telah aktif. List subdokumen telah ditampilkan.		
No	**Aksi Aktor**	**No.**	**Reaksi Sistem**
1.	Klik *"Content"*.		
		2.	Sistem menampilkan content dari subdokumen.
Kondisi Akhir	Konten subdokumen tertentu telah ditampilkan		
Eksepsi	-		

Tabel 3.15 Skenario Lihat Konten

n. Edit Subdokumen

Identifikasi	
Nomor	ADM-05-3
Nama	Edit Subdokumen
Tujuan	Mengedit data subdokumen tertentu.
Deskripsi	Proses data mengedit subdokumen tertentu.
Aktor	Admin

Skenario Utama			
Kondisi Awal	Halaman kelola subdokumen telah aktif. List subdokumen telah ditampilkan.		
No	**Aksi Aktor**	**No.**	**Reaksi Sistem**
1.	Klik *"Edit"*.		
		2.	Sistem menyajikan form edit subdokumen berikut data subdokumen yang akan diedit.

3.	Edit data subdokumen sesuai kebutuhan, klik "Update"		
		4.	Sistem update data subdokumen di *database*, munculkan pesan proses edit berhasil.
Kondisi Akhir	Data subdokumen tertentu telah diupdate di *database*.		
Eksepsi	Jika pengisian field pada form tidak sesuai aturan, muncul pesan proses edit gagal.		

Tabel 3.16 Skenario Edit Subdokumen

o. Hapus Subdokumen

Identifikasi			
Nomor	ADM-05-4		
Nama	Hapus Subdokumen		
Tujuan	Menghapus data subdokumen tertentu.		
Deskripsi	Proses menghapus data subdokumen tertentu dari *database*.		
Aktor	Admin		
Skenario Utama			
Kondisi Awal	Halaman kelola subdokumen telah aktif. List subdokumen telah ditampilkan.		
No	**Aksi Aktor**	**No.**	**Reaksi Sistem**
1.	Klik "Hapus".		

		2.	Sistem menyajikan form konfirmasi hapus data subdokumen.
3.	Konfirmsi hapus data subdokumen.		
		4.	Sistem menghapus data subdokumen tertentu dari *database*.
Kondisi Akhir	Data subdokumen tertentu telah dihapus dari *database*.		
Eksepsi	-		

Tabel 3.17 Skenario Hapus Subdokumen

2. **Visitor**

a. Quick Scan Similaritas Dokumen

Identifikasi			
Nomor	VST-01		
Nama	Quick Scan Similaritas Dokumen		
Tujuan	Mengetahui tingkat similaritas dua dokumen yang diinput.		
Deskripsi	Proses membandingkan tingkat similaritas antar dua dokumen yang dinputkan secara langsung oleh visitor.		
Aktor	Visitor		
Skenario Utama			
Kondisi Awal	Halaman visitor telah aktif.		
No	**Aksi Aktor**	**No.**	**Reaksi Sistem**

1.	Klik "*Quick Scan Document*.		
		2.	Sistem menyajikan form quick scan dokumen.
3.	Input dokumen 1 dan dokumen 2.		
		4.	Sistem membandingkan tingkat similaritas dua dokumen tersebut dengan cepat.
Kondisi Akhir	Tampil hasil yang berupa persentase similaritas dokumen.		
Eksepsi	Jika pengisian field pada form tidak sesuai aturan akan dimunculkan pesan.		

Tabel 3.18 Skenario Quick Scan Similaritas Dokumen

b. Normal Scan Similaritas Dokumen

Identifikasi			
Nomor	VST-02		
Nama	Normal Scan Similaritas Dokumen		
Tujuan	Mengetahui tingkat similaritas dua dokumen.		
Deskripsi	Proses membandingkan tingkat similaritas antar dokumen yang dinput dengan dokumen di database.		
Aktor	Visitor		
Skenario Utama			
Kondisi Awal	Halaman visitor telah aktif.		
No	**Aksi Aktor**	**No.**	**Reaksi Sistem**

1.	Klik *"Normal Scan Document"*.		
		2.	Sistem menyajikan form normal scan dokumen.
3.	Input dokumen 1 dan pilih dokumen di database.		
		4.	Sistem membandingkan tingkat similaritas dua dokumen tersebut dengan cepat.
Kondisi Akhir	Tampil hasil yang berupa persentase similaritas dokumen.		
Eksepsi	Jika pengisian field pada form tidak sesuai aturan akan dimunculkan pesan.		

Tabel 3.19 Skenario Normal Scan Similaritas Dokumen

c. Setting Parameter

Identifikasi			
Nomor	VST-03		
Nama	Pengaturan Parameter		
Tujuan	Merubah nilai parameter algoritma Rabin-Karp.		
Deskripsi	Proses merubah nilai parameter algoritma Rabin-Karp.		
Aktor	Visitor		
Skenario Utama			
Kondisi Awal	Halaman utama aplikasi telah aktif.		
No	**Aksi Aktor**	**No.**	**Reaksi Sistem**

1.	Klik *"Normal Scan Document"*.		
		2.	Sistem menyajikan form setting parameter.
3.	Edit nilai parameter sesuai kebutuhan, klik "Update".		
		4.	Sistem mengupdate nilai parameter algoritma Rabin-Karp.
Kondisi Akhir	Nilai parameter berhasil diupdate.		
Eksepsi	Jika pengisian field pada form tidak sesuai aturan akan dimunculkan pesan.		

Tabel 3.20 Skenario Setting Parameter

d. Lihat Bantuan

Identifikasi			
Nomor	VST-05		
Nama	Info Bantuan		
Tujuan	Melihat bantuan atau informasi aplikasi.		
Deskripsi	Proses melihat bantuan atau informasi aplikasi.		
Aktor	Visitor		
Skenario Utama			
Kondisi Awal	Halaman utama aplikasi telah aktif.		
No	**Aksi Aktor**	**No.**	**Reaksi Sistem**
1	Klik "About".		

		2	Sistem menyajikan form informasi aplikasi.
Kondisi Akhir	Informasi terkait aplikasi telah ditampilkan.		
Eksepsi	-		

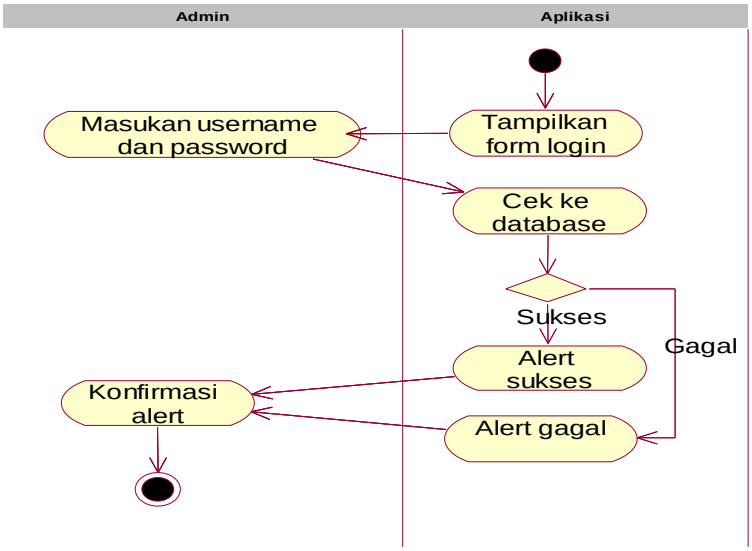
Tabel 3.21 Skenario Admin Lihat Bantuan

3.1.4.5 *Activity Diagram*

Activity diagram menggambarkan aliran aktivitas dalam sistem, bagaimana masing-masing aliran dimulai, keputusan yang mungkin terjadi, dan bagaimana aktivitas berakhir. *Activity* diagram juga dapat menggambarkan proses pararel yang mungkin terjadi pada beberapa eksekusi. *Activity* diagram dapat dibagi menjadi beberapa *swimlane object* untuk menggambarkan object mana yang bertanggung jawab untuk aktivitas tertentu.

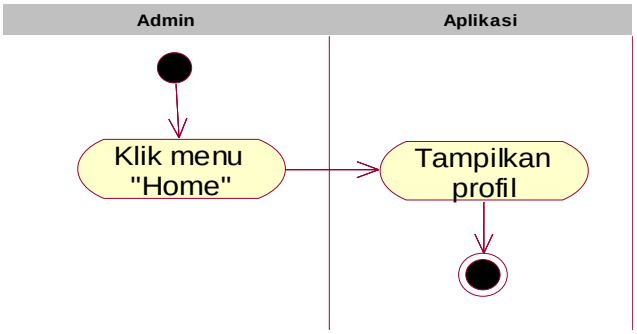
a. Activity Diagram Admin

1) Login



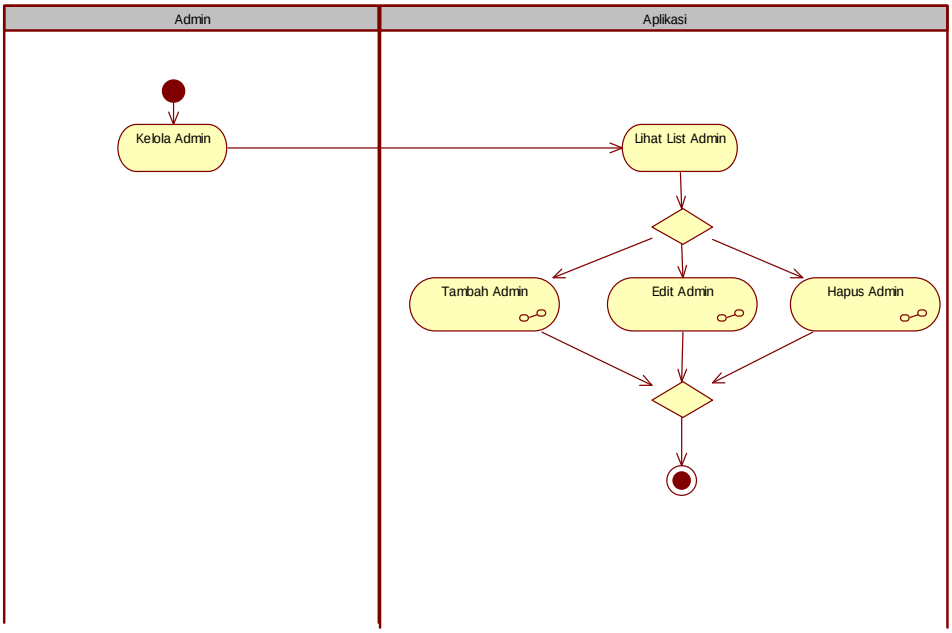
Gambar 3.5 *Activity Diagram Login*

2) Lihat Profil



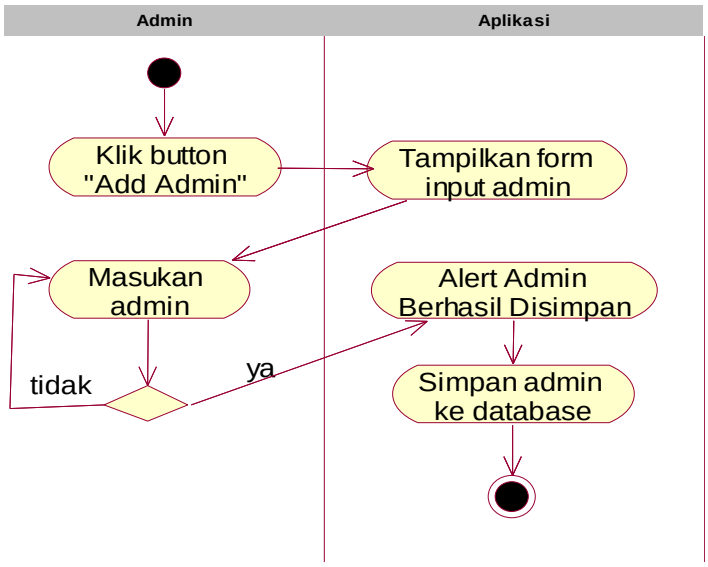
Gambar 3.6 *Activity Diagram Lihat Profil*

3) Kelola Admin



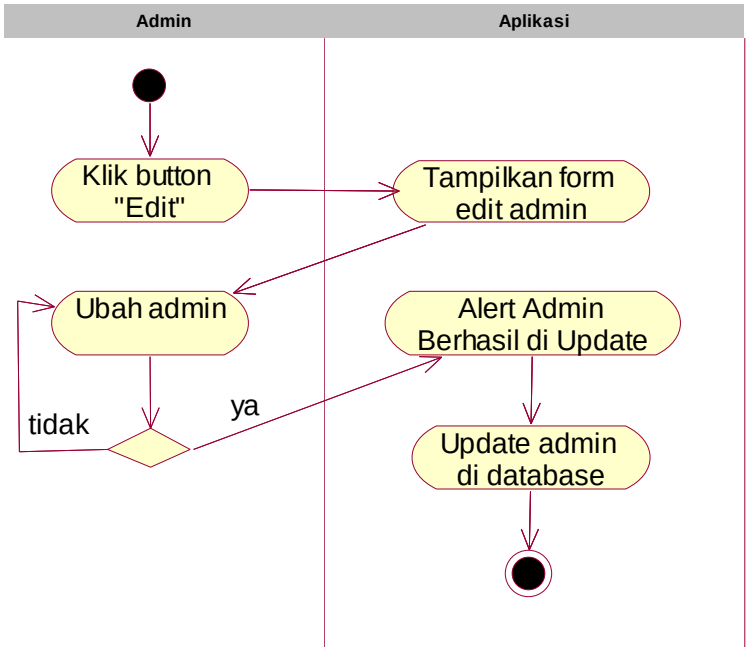
Gambar 3.7 *Activity Diagram* Kelola Admin

4) Tambah Admin



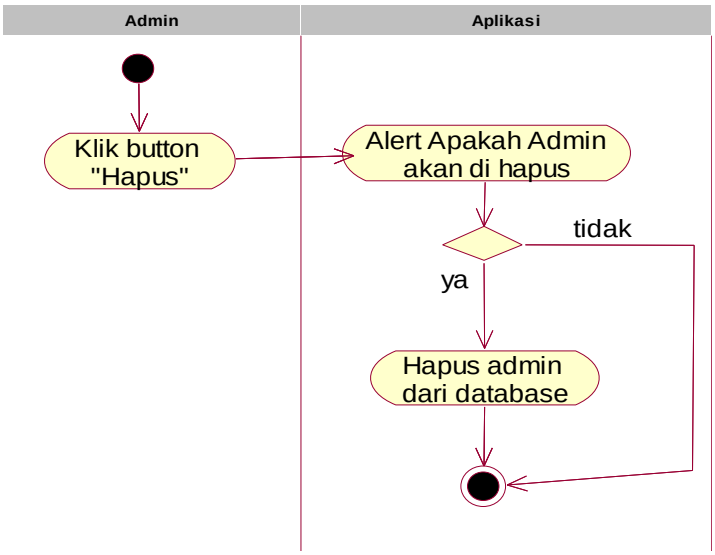
Gambar 3.8 *Activity Diagram* Tambah Admin

5) Edit Admin



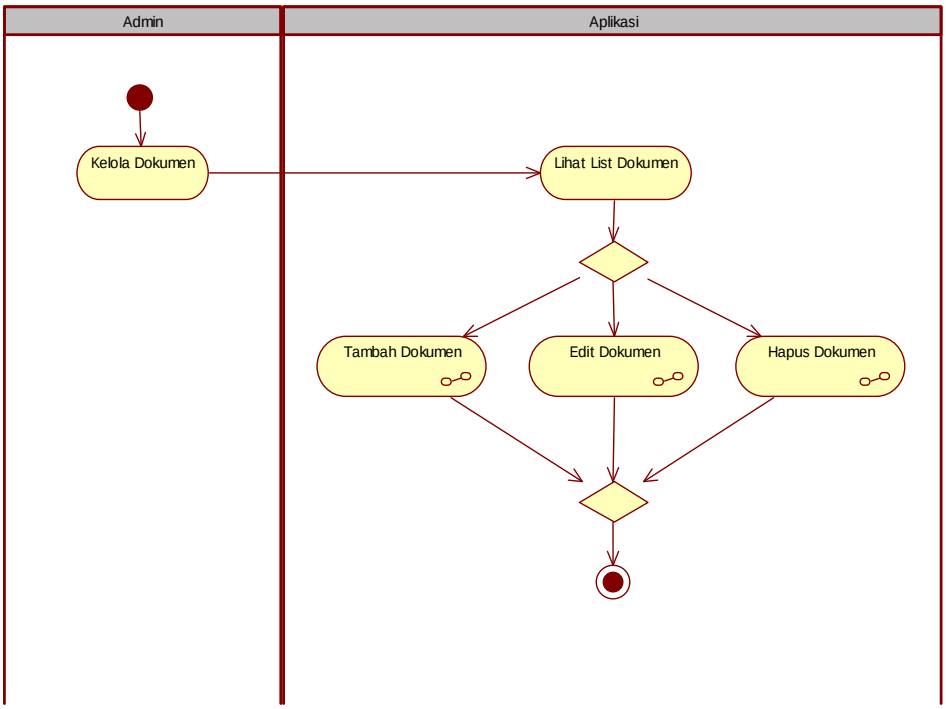
Gambar 3.9 *Activity Diagram* Edit Admin

6) Hapus Admin



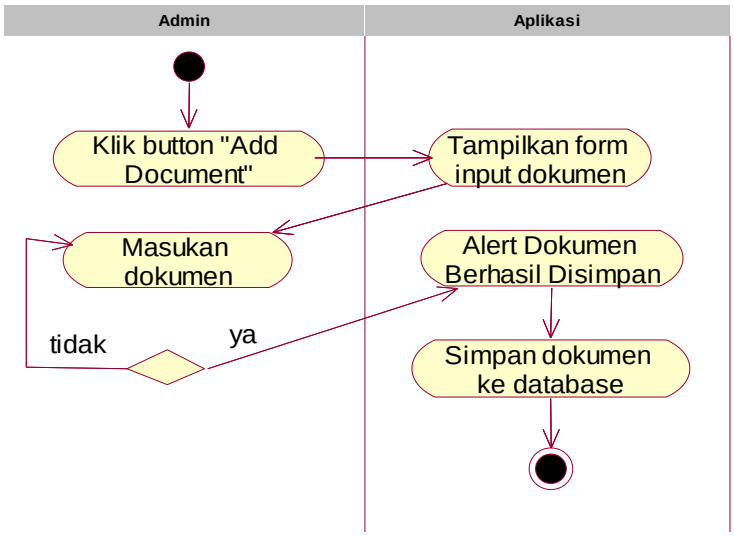
Gambar 3.10 *Activity Diagram* Hapus Admin

7) Kelola Dokumen



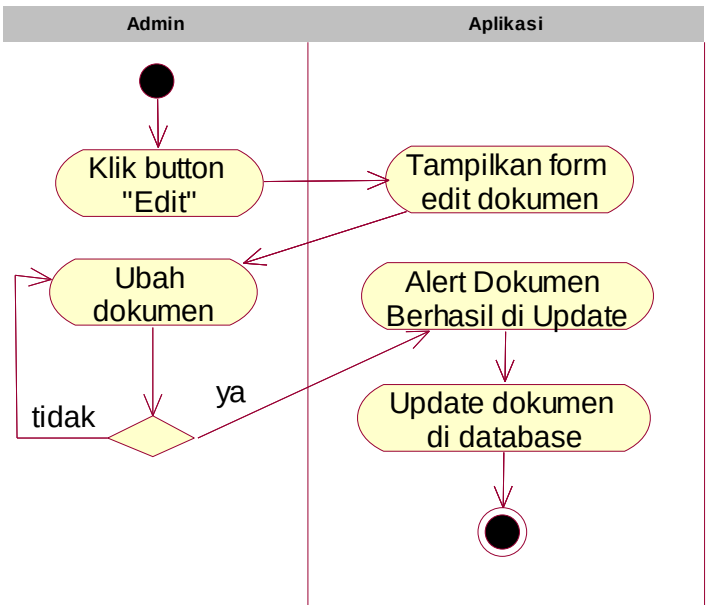
Gambar 3.11 *Activity Diagram* Kelola Dokumen

8) Tambah Dokumen



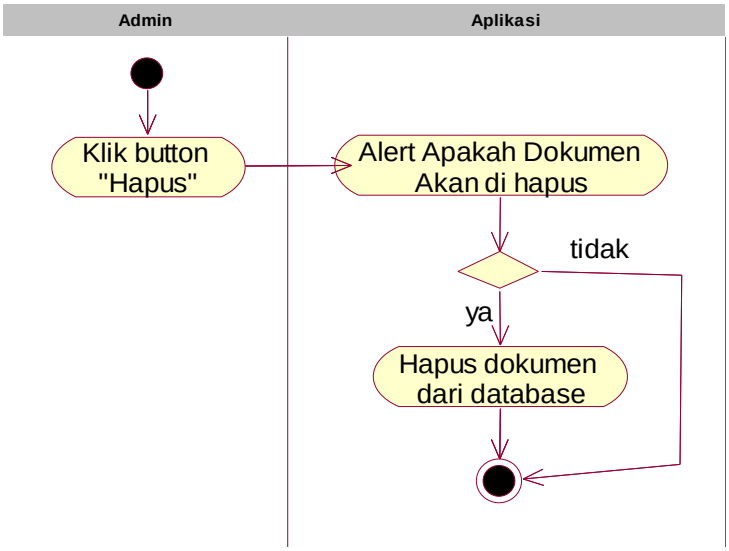
Gambar 3.12 *Activity Diagram* Tambah Dokumen

9) Edit Dokumen



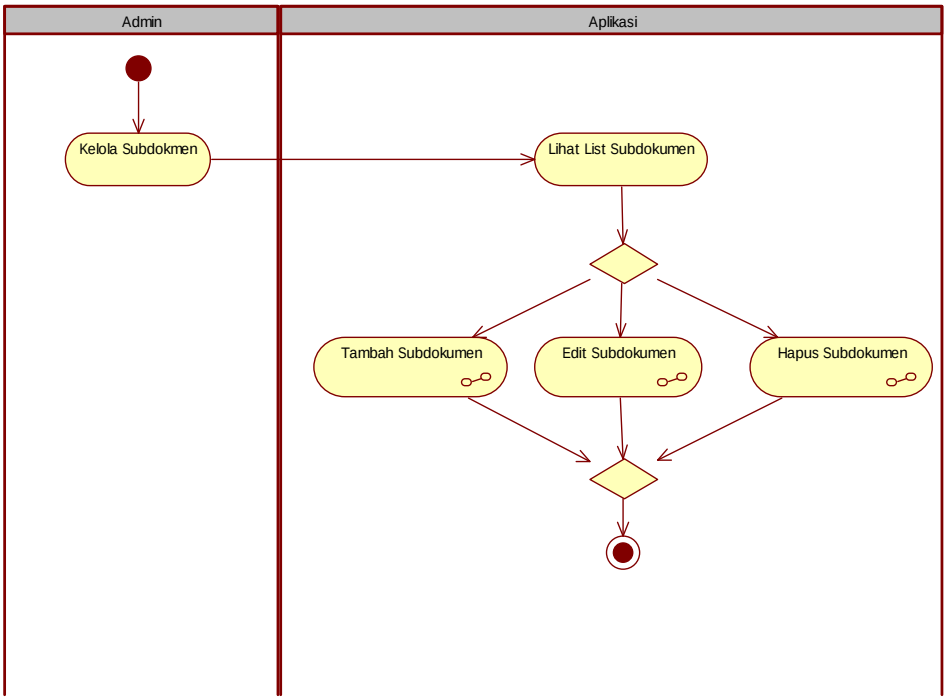
Gambar 3.13 *Activity Diagram* Edit Dokumen

10) Hapus Dokumen



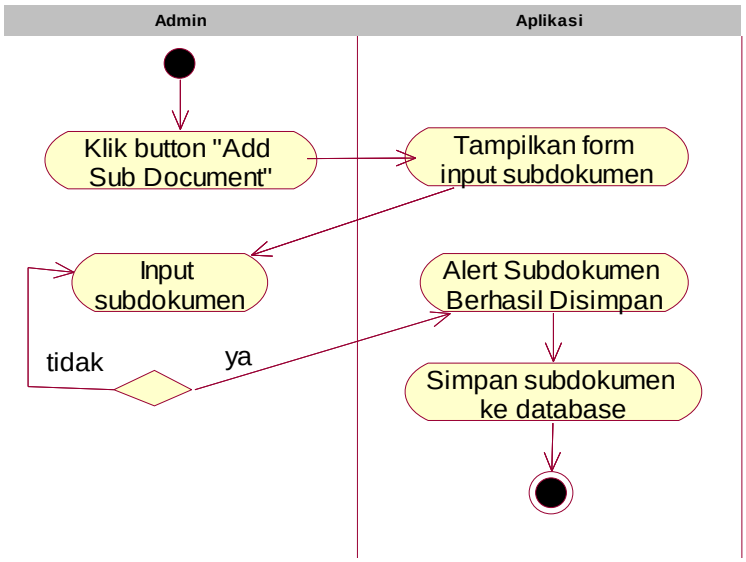
Gambar 3.14 *Activity Diagram* Hapus Dokumen

11) Kelola Subdokumen



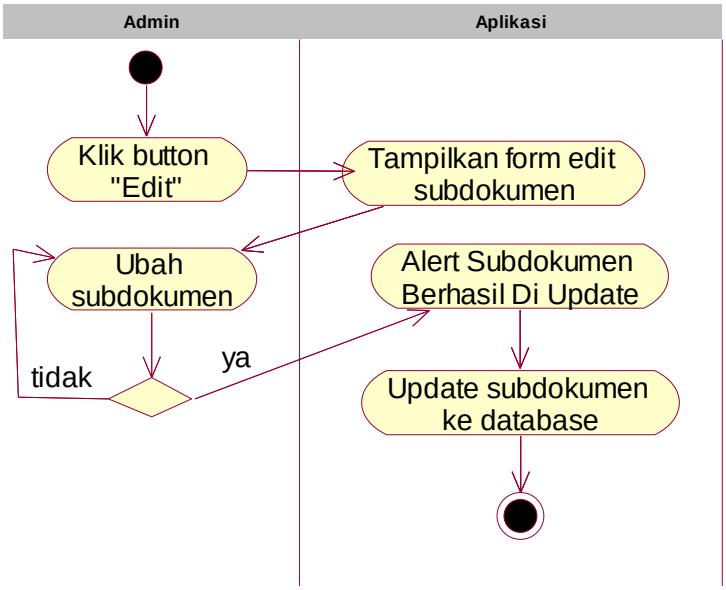
Gambar 3.15 *Activity Diagram* Kelola Subdokumen

12) Tambah Subdokumen



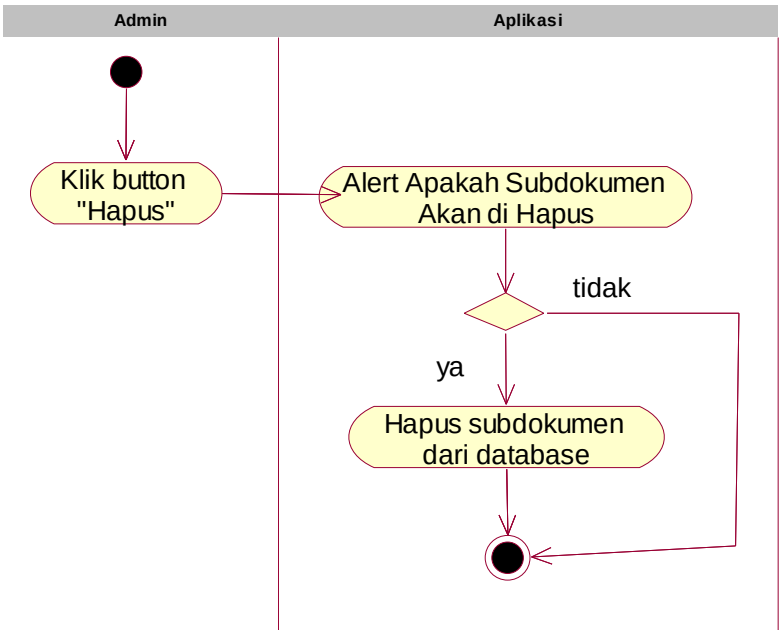
Gambar 3.16 *Activity Diagram* Tambah Subdokumen

13) Edit Subdokumen



Gambar 3.17 *Activity Diagram* Edit Subdokumen

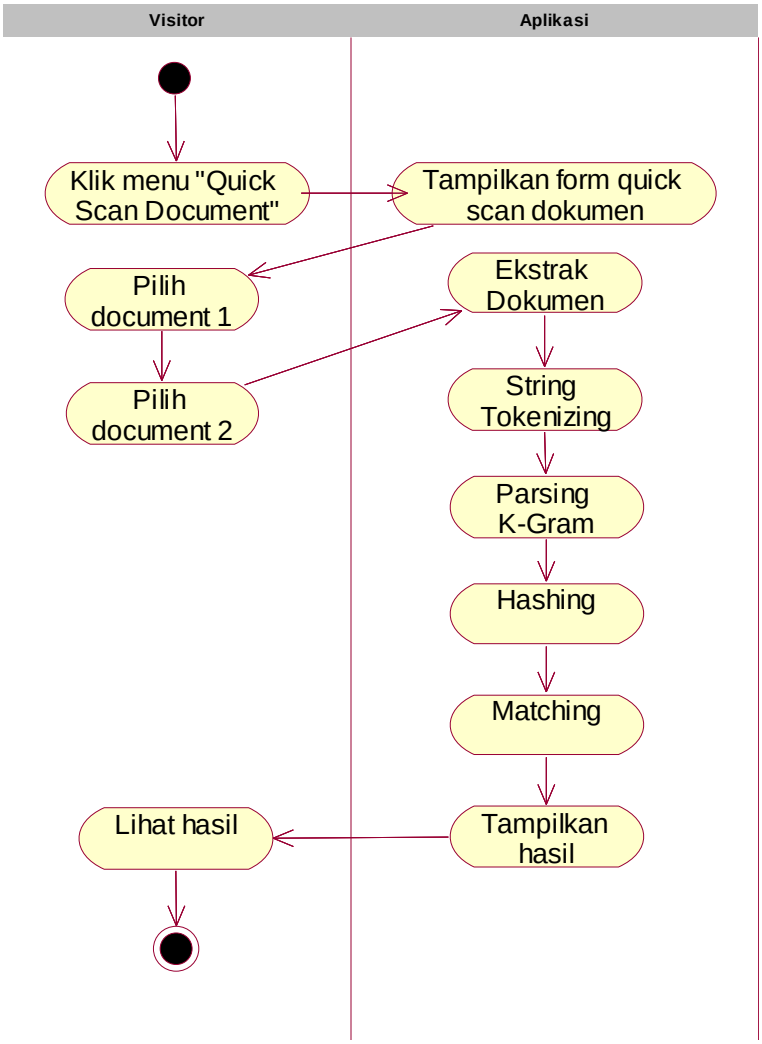
14) Hapus Subdokumen



Gambar 3.18 *Activity Diagram* Hapus Subdokumen

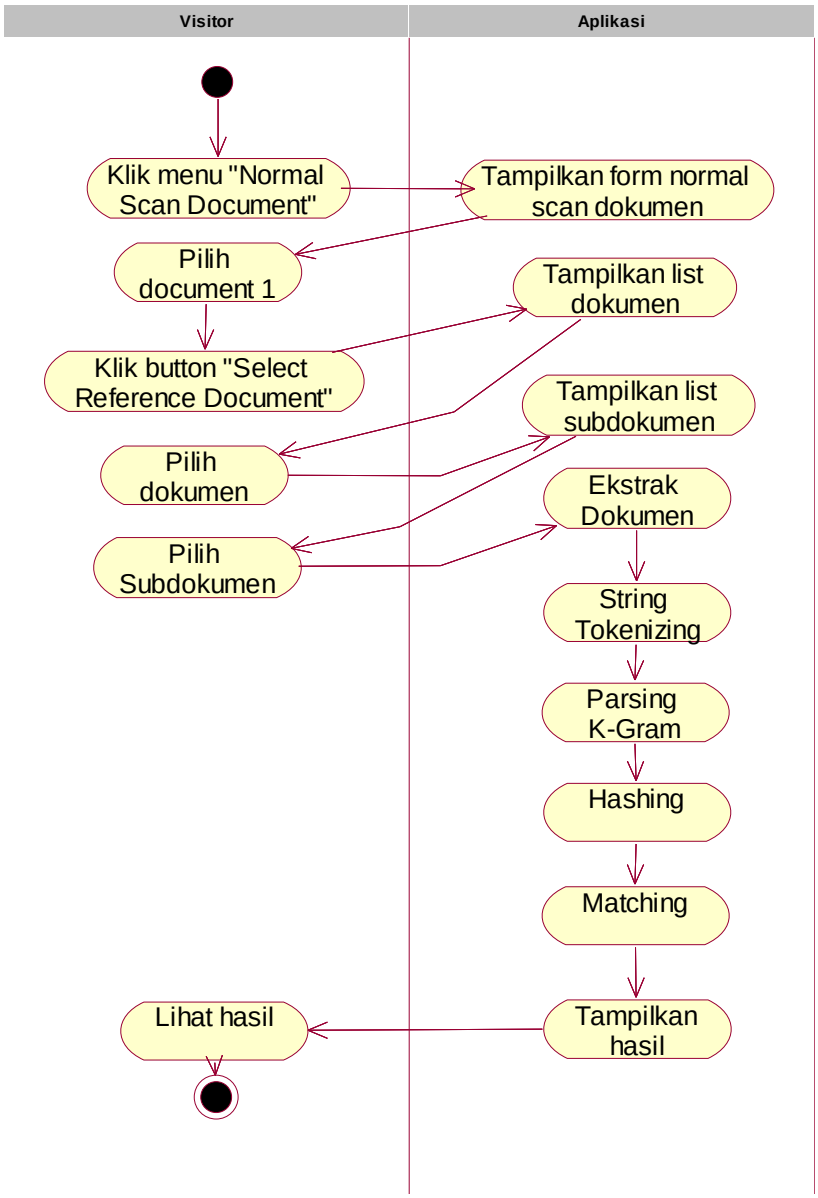
b. Activity Diagram Visitor

1) Quick Scan Similaritas Dokumen



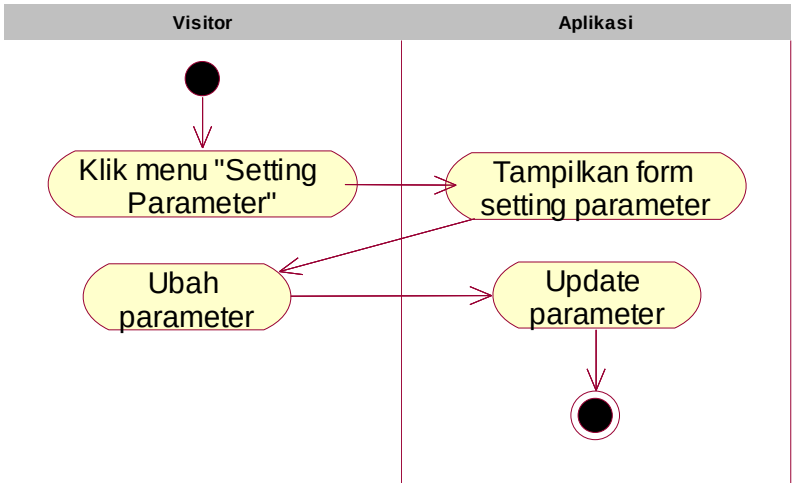
Gambar 3.19 *Activity Diagram Quick Scan Similaritas Dokumen*

2) Normal Scan Similaritas Dokumen



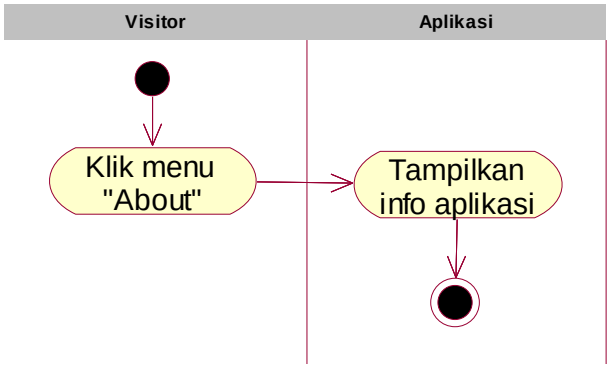
Gambar 3.20 *Activity Diagram Normal Scan Similaritas Dokumen*

3) Pengaturan Parameter



Gambar 3.21 *Activity Diagram Setting Parameter*

4) Info Bantuan



Gambar 3.22 *Activity Diagram Lihat Bantuan*

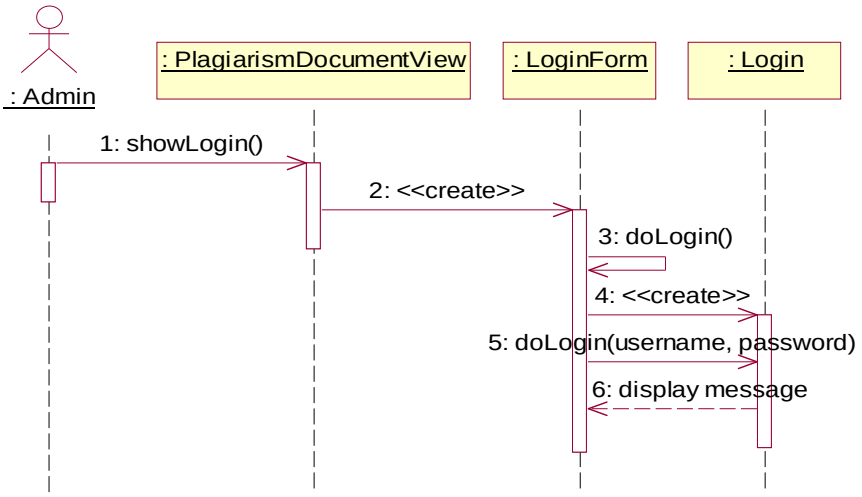
3.1.4.6 Sequence Diagram

Menggambarkan interaksi antar objek di dalam dan di sekitar sistem (termasuk pengguna, display dan sebagainya) berupa *message* yang digambarkan terhadap waktu. Dalam tahap perancangan, diagram ini akan digambarkan sebagai berikut:

a. *Sequence Diagram* Admin

1) Login

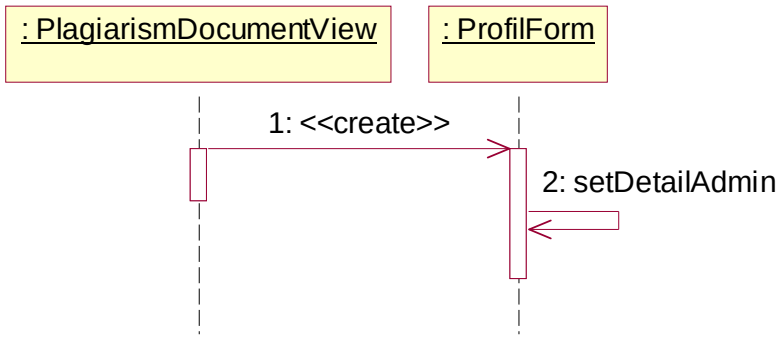
Halaman home aplikasi telak aktif. Ketika di klik menu "Login", aplikasi akan memanggil fungsi showLogin() pada kelas PlagiarismDocumentView. Fungsi ini mengeksekusi konstruktor LoginForm() untuk mengaktifkan kelas LoginForm sehingga halaman login admin ditampilkan. Setelah username dan password diisi dan diklik button "Login", aplikasi akan menjalankan fungsi doLogin() pada kelas LoginForm. Di dalam fungsi ini didapatkan nilai username dan password, inisialisasi kelas Login yang terhubung ke database, pemanggilan fungsi doLogin(username, dan password) pada kelas login untuk pengecekan apakah username dan password admin ada di database atau tidak. Jika ada, maka akan dimunculkan alert bahwa login berhasil begitu sebaliknya.



Gambar 3.23 *Sequence Diagram Login*

2) Lihat Profil

Admin telah berhasil login. Ketika diklik menu “Home”, aplikasi akan memanggil fungsi showHome() yang didalamnya terdapat proses inisialisasi kelas ProfilForm dengan mengeksekusi konstruktor ProfilForm untuk menampilkan profil admin.

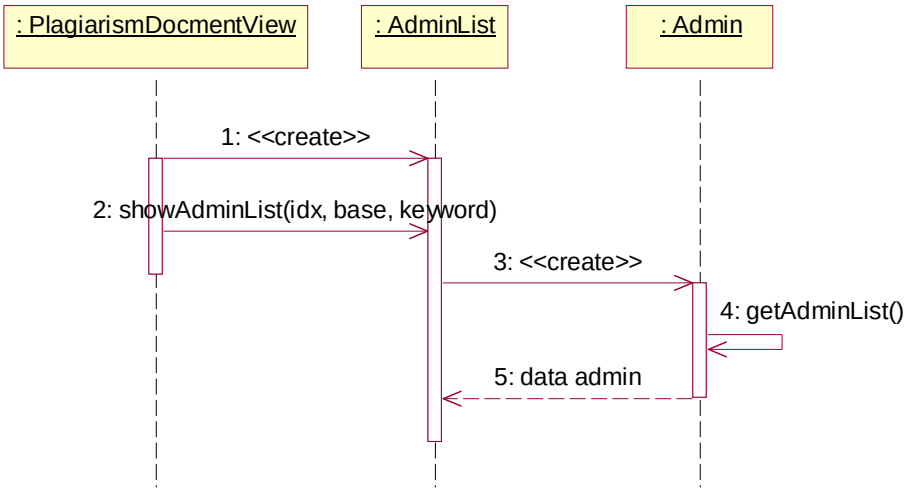


Gambar 3.24 *Sequence Diagram Lihat Profil*

3) Kelola Admin

Halaman home admin aplikasi telak aktif. Ketika di klik menu “Manage Admin”, aplikasi akan memanggil fungsi showAdminList() pada kelas PlagiarismDocumentView. Fungsi ini mengeksekusi konstruktor AdminList() untuk mengaktifkan kelas AdminList. Di dalam konstruktor itu, dieksekusi fungsi showAdminList(idx, base, keyword). Di dalam fungsi inilah dilakukan inisialisasi kelas Admin yang terhubung ke database dimana kelas ini memiliki fungsi getAdminList() untuk mengambil data admin yang ada di database untuk ditampilkan di halaman

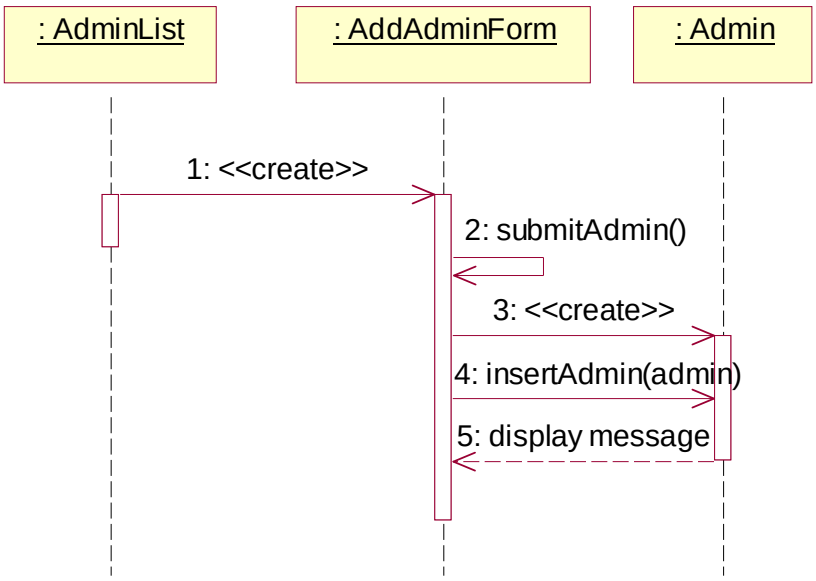
kelola admin.



Gambar 3.25 *Sequence Diagram Kelola Admin*

4) Input Admin

Halaman kelola admin telah aktif. Ketika diklik button “Add Admin”, aplikasi memanggil fungsi addAdmin() pada kelas AdminList. Pada fungsi ini terjadi inisialisasi kelas AddAdminForm dengan mengeksekusi konstruktor AddAdminForm() untuk menampilkan form input admin. Setelah field pada form telah diisi dan diklik button “Submit”, fungsi submitAdmin() dieksekusi untuk menerima input data admin, menginisialisasi kelas Admin yang terhubung ke database, dan memanggil fungsi insertAdmin(admin) untuk menginsert data admin baru ke database. Setelah itu ditampilkan alert sebagai informasi sukses tidaknya proses input data admin.

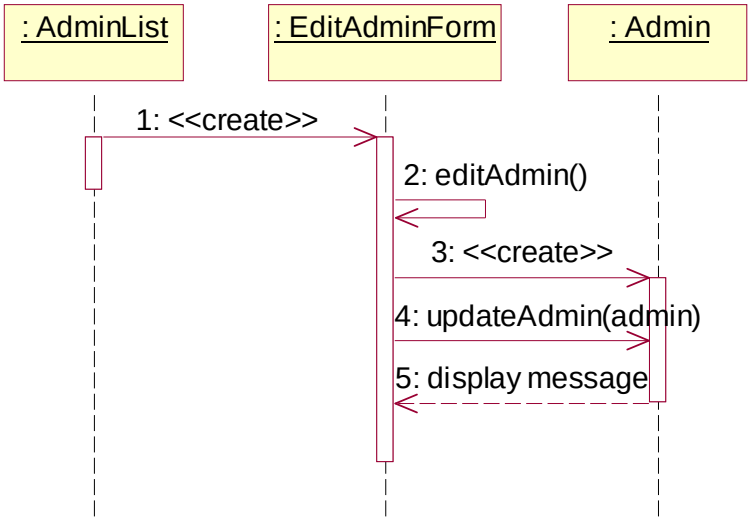


Gambar 3.26 *Sequence Diagram Input Admin*

5) Edit Admin

Halaman kelola admin telah aktif. Ketika diklik button “Edit”, aplikasi akan memanggil fungsi editAdminForm(vectorAdmin). Pada fungsi ini dilakukan pemanggilan konstruktor EditAdminForm(vectorAdmin) untuk mengaktifkan kelas EditAdminForm sehingga ditampilkan form edit data admin. Setelah field pada form diisi dan diklik button “Update”, fungsi editAdmin() dieksekusi untuk menerima input, mengaktifkan kelas Admin yang terhubung ke database, dan mengeksekusi fungsi updateAdmin(admin) untuk merubah data admin tertentu pada database. Setelah itu, alert ditampilkan untuk menunjukkan berhasil

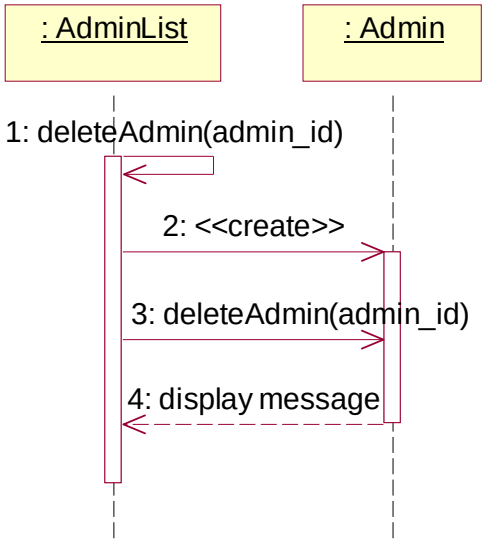
atau tidaknya proses edit data admin.



Gambar 3.27 *Sequence Diagram Edit Admin*

6) Hapus Admin

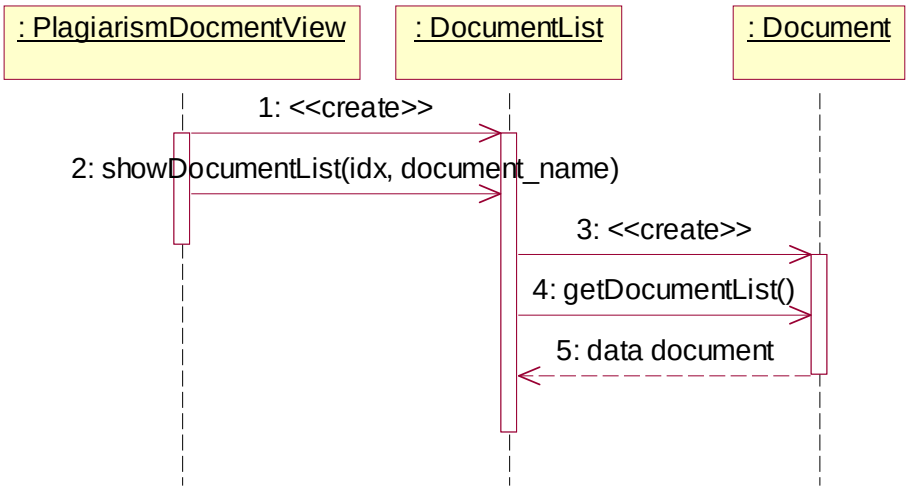
Halaman kelola admin telah aktif. Ketika diklik button “Hapus”, aplikasi akan memanggil fungsi deleteAdmin(admin_id) pada kelas AdminList. Pada fungsi ini dilakukan inisialisasi kelas Admin yang terhubung ke database dan dieksekusi fungsi deleteAdmin(admin_id) untuk menghapus data admin di database. Setelah itu, ditampilkan alert untuk menunjukkan berhasil tidaknya proses hapus data admin.



Gambar 3.28 *Sequence Diagram Hapus Admin*

7) Kelola Dokumen

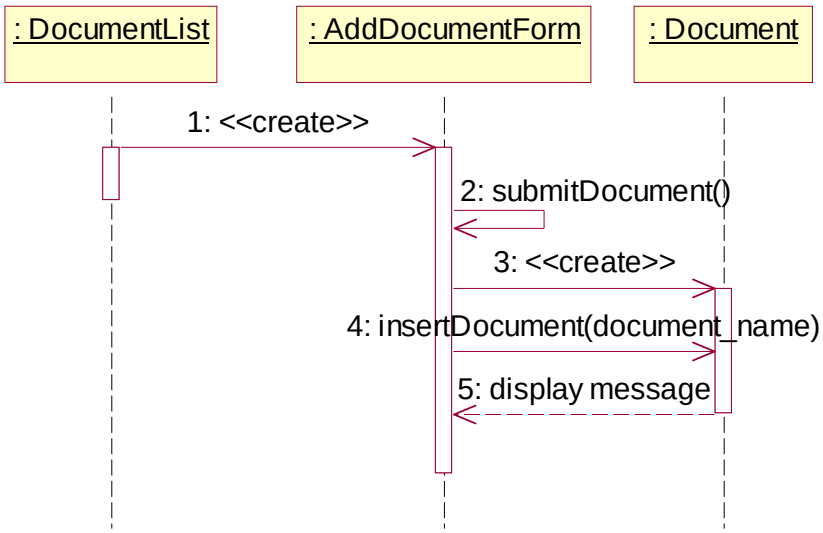
Halaman home aplikasi telak aktif. Ketika di klik menu "Manage Document", aplikasi akan memanggil fungsi showDocumentList() pada kelas PlagiarismDocumentView. Fungsi ini mengeksekusi konstruktor DocumentList() untuk mengaktifkan kelas DocumentList. Di dalam konstruktor itu, dieksekusi fungsi showDocumentList(idx, document_name). Di dalam fungsi inilah dilakukan inisialisasi kelas Document yang terhubung ke database dimana kelas ini memiliki fungsi getDocumentList() untuk mengambil data dokumen yang ada di database untuk ditampilkan di halaman kelola dokumen.



Gambar 3.29 *Sequence Diagram Kelola Dokumen*

8) Input Dokumen

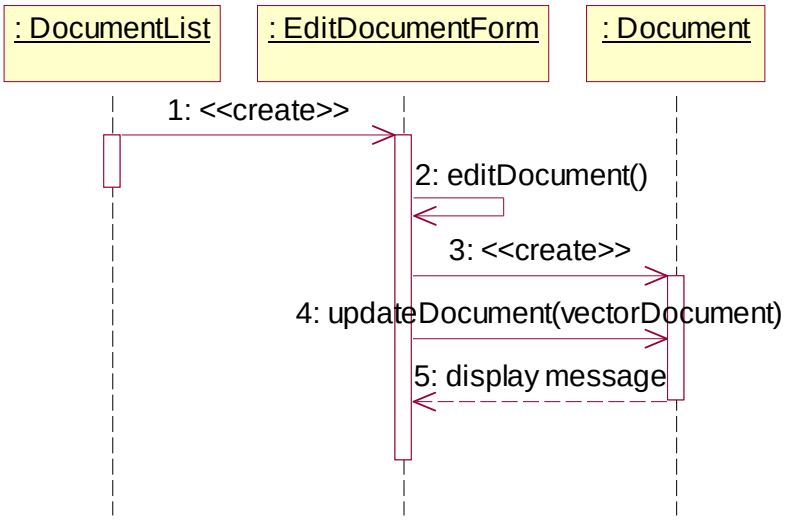
Halaman kelola dokumen telah aktif. Ketika diklik button “Add Document”, aplikasi memanggil fungsi addDocument() pada kelas DocumentList. Pada fungsi ini terjadi inisialisasi kelas AddDocumentForm dengan mengeksekusi konstruktor AddDocumentForm() untuk menampilkan form input dokumen. Setelah field pada form telah diisi dan diklik button “Submit”, fungsi submitDocument() dieksekusi untuk menerima input nama dokumen, menginisialisasi kelas Document yang terhubung ke database, dan memanggil fungsi insertDocument(document_name) untuk menginsert dokumen baru ke database. Setelah itu ditampilkan alert sebagai informasi sukses tidaknya proses input dokumen.



Gambar 3.30 *Sequence Diagram Input Dokumen*

9) Edit Dokumen

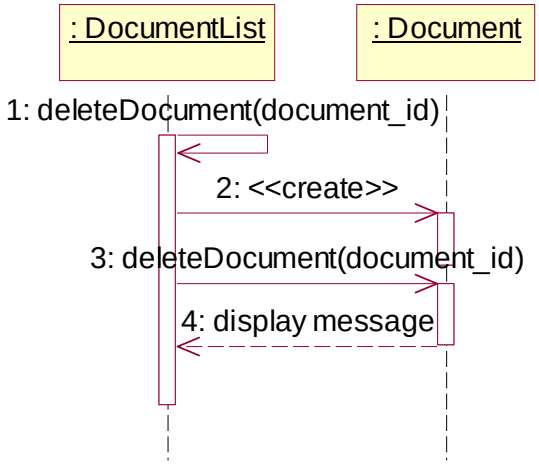
Halaman kelola dokumen telah aktif. Ketika diklik button "Edit", aplikasi akan memanggil fungsi editDocument(vecDoc). Pada fungsi ini dilakukan pemanggilan konstruktor EditDocumentForm(vectorDocument) untuk mengaktifkan kelas EditDocumentForm sehingga ditampilkan form edit dokumen. Setelah field pada form diisi dan diklik button "Update", fungsi editDocument() dieksekusi untuk menerima input, mengaktifkan kelas Document yang terhubung ke database, dan mengeksekusi fungsi updateDocument(vectorDocument) untuk merubah data dokumen tertentu pada database. Setelah itu, alert ditampilkan untuk menunjukkan berhasil atau tidaknya proses edit.



Gambar 3.31 *Sequence Diagram Edit Dokumen*

10) Hapus Dokumen

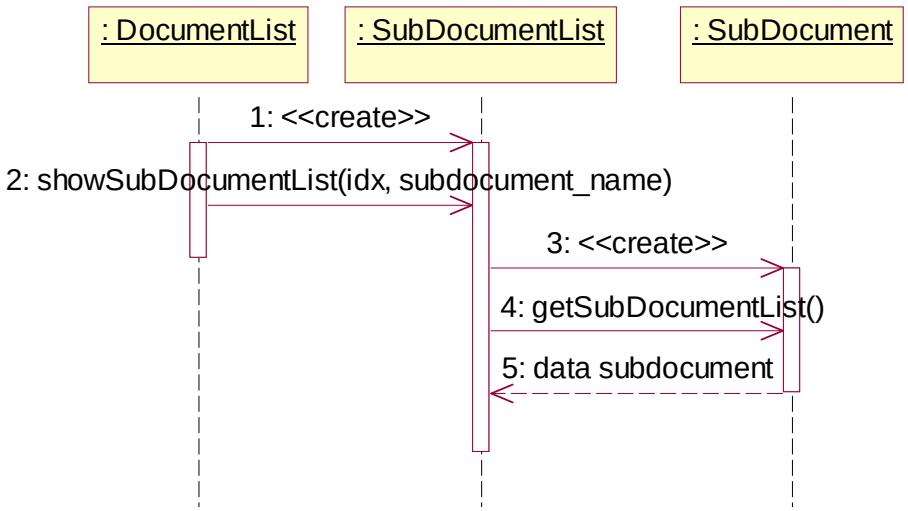
Halaman kelola dokumen telah aktif. Ketika diklik button "Hapus", aplikasi akan memanggil fungsi deleteDocument(document_id) pada kelas DocumentList. Pada fungsi ini dilakukan inisialisasi kelas Document yang terhubung ke database dan dieksekusi fungsi deleteDocument(document_id) untuk menghapus dokumen di database. Setelah itu, ditampilkan alert untuk menunjukkan berhasil tidaknya proses hapus.



Gambar 3.32 *Sequence Diagram Hapus Dokumen*

11) Kelola Subdokumen

Halaman kelola dokumen telak aktif. Ketika diklik menu “Subdocument”, aplikasi akan memanggil fungsi showSubDocumentList(document) pada kelas DocumentList. Di dalam fungsi ini dilakukan eksekusi konstruktor SubDocumentList(document) untuk mengaktifkan kelas SubDocumentList. Di dalam konstruktor itu, dieksekusi juga fungsi showSubDocumentList(idx, subdocument_name). Di dalam fungsi inilah terjadi inisialisasi kelas SubDocument yang terhubung ke database dimana kelas ini memiliki fungsi getSubDocumentList() untuk mengambil data subdokumen yang ada di database untuk ditampilkan di halaman kelola subdokumen.

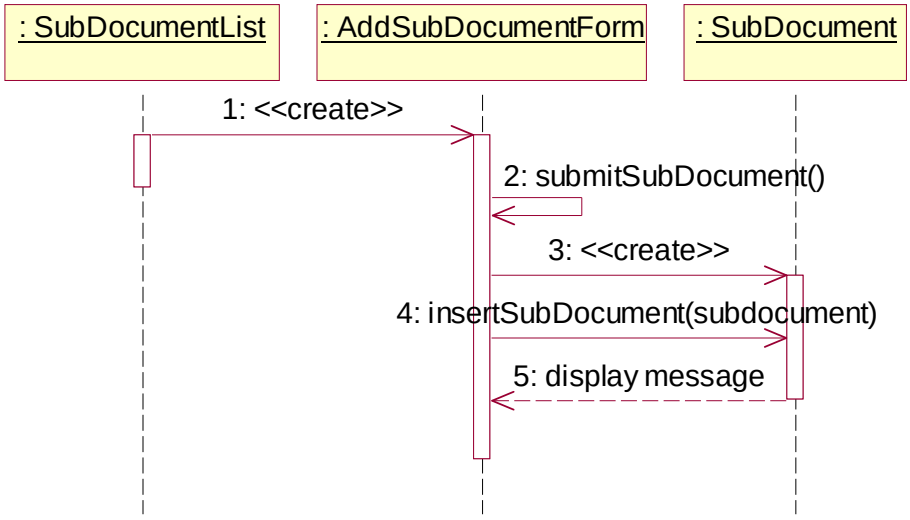


Gambar 3.33 *Sequence Diagram Kelola Subdokumen*

12) Input Subdokumen

Halaman kelola subdokumen telah aktif. Ketika diklik button "Add Sub Document", aplikasi memanggil fungsi addSubDocument() pada kelas SubDocumentList. Di dalam fungsi ini dilakukan inisialisasi kelas AddSubDocumentForm dengan melakukan eksekusi konstruktor AddSubDocumentForm() untuk menampilkan form input subdokumen. Setelah field pada form telah diisi dan diklik button "Submit", fungsi submitSubDocument() dieksekusi untuk menerima input nama subdokumen, menginisialisasi kelas SubDocument yang terhubung ke database, dan memanggil fungsi insertSubDocument(subdocument) untuk menginsert subdokumen baru ke database. Setelah itu ditampilkan alert sebagai informasi

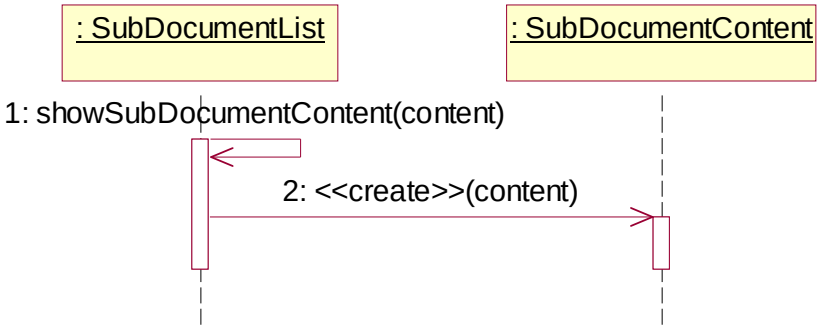
sukses tidaknya proses input subdokumen.



Gambar 3.34 *Sequence Diagram Admin Input Subdokumen*

13) Lihat Konten

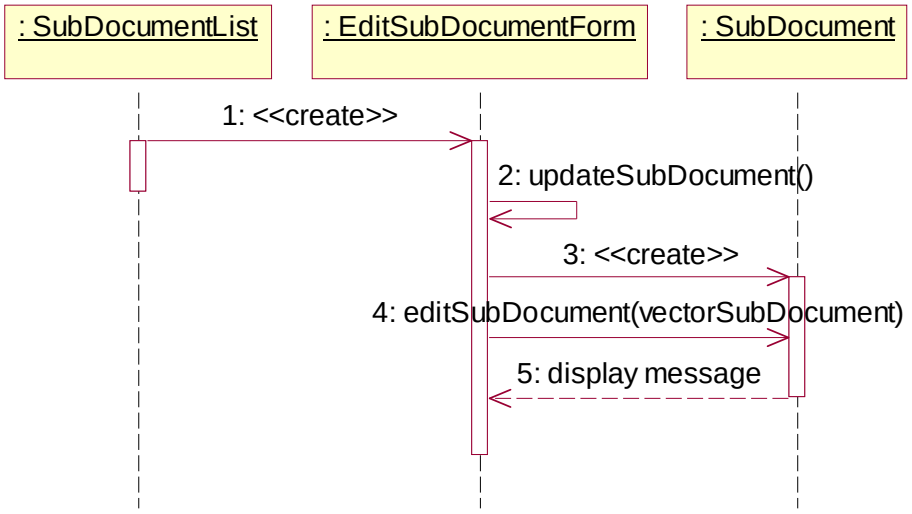
Halaman kelola subdokumen telah aktif. Ketika diklik button "Content", aplikasi memanggil fungsi showSubDocumentContent(content) pada kelas SubDocumentList. Di dalam fungsi ini dilakukan inisialisasi kelas SubDocumentContent dengan cara melakukan eksekusi konstruktor SubDocumentContent(content) untuk menampilkan form yang berisi konten dari subdokumen.



Gambar 3.35 *Sequence Diagram Lihat Konten*

14) Edit Subdokumen

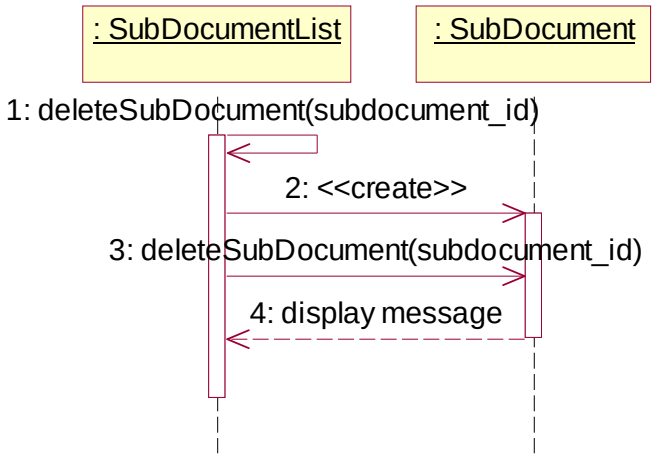
Halaman kelola subdokumen telah aktif. Ketika diklik button "Edit", aplikasi akan memanggil fungsi editSubDocument(vecsubdoc). Pada fungsi ini dilakukan pemanggilan konstruktor EditSubDocument(vectorDocument, vectorSubDocuments) untuk mengaktifkan kelas EditSubDocument sehingga ditampilkan form edit subdokumen. Setelah field pada form diisi dan diklik button "Update", fungsi updateSubDocument() dieksekusi untuk menerima input, mengaktifkan kelas SubDocument yang terhubung ke database, dan mengeksekusi fungsi editSubDocument(vectorSubDocument) untuk merubah data subdokumen tertentu pada database. Setelah itu, alert ditampilkan untuk menunjukkan berhasil atau tidaknya proses edit.



Gambar 3.36 *Sequence Diagram Edit Subdokumen*

15) Hapus Subdokumen

Halaman kelola subdokumen telah aktif. Ketika diklik button "Hapus", aplikasi akan memanggil fungsi deleteSubDocument(subdocument_id) pada kelas SubDocumentList. Pada fungsi ini dilakukan inisialisasi kelas SubDocument yang terhubung ke database dan dieksekusi fungsi deleteSubDocument(subdocument_id) untuk menghapus subdokumen di database. Setelah itu, ditampilkan alert untuk menunjukkan berhasil atau tidaknya proses hapus subdokumen.



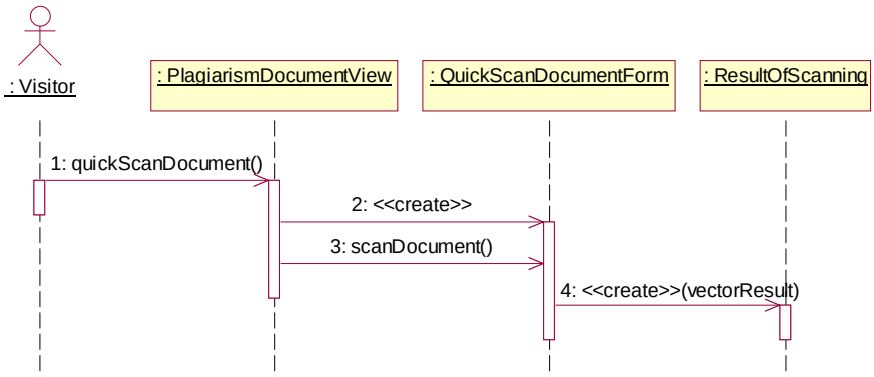
Gambar 3.37 *Sequence Diagram Hapus Subdokumen*

b. *Sequence Diagram* Visitor

1) Quick Scan Similaritas Dokumen

Halaman home aplikasi telak aktif. Ketika di klik menu “Quick Scan Document”, aplikasi akan memanggil fungsi quickScanDocument() pada kelas PlagiarismDocumentView. Di dalam fungsi ini dieksekusi konstruktor QuickScanDocument() untuk mengaktifkan kelas QuickScanDocument sehingga halaman quick scan dokumen ditampilkan. Setelah form diisi, dan diklik button “Process”, aplikasi akan menjalankan fungsi scanDocument(). Di dalam fungsi ini dilakukan ekstraksi konten dokumen, hashing, dan perhitungan similaritas antar dokumen. Hasil akan ditampilkan pada kelas ResultOfScanning dengan melakukan eksekusi terhadap konstruktor

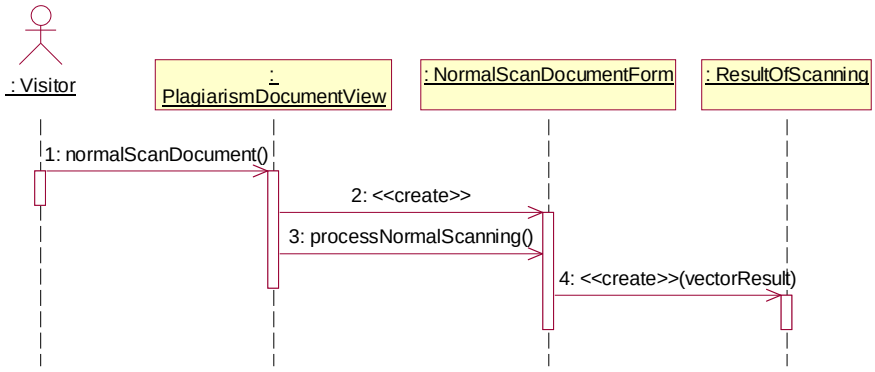
ResultOfScanning(vectorResult) .



Gambar 3.38 *Sequence Diagram Quick Scan Similaritas Dokumen*

2) Normal Scan Similaritas Dokumen

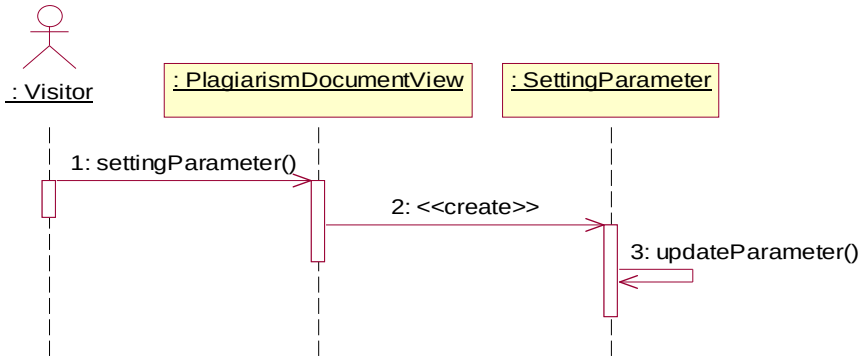
Halaman home aplikasi telak aktif. Ketika di klik menu “Normal Scan Document”, aplikasi akan memanggil fungsi normalScanDocument() pada kelas PlagiarismDocumentView. Di dalam fungsi ini dieksekusi konstruktor NormalScanDocument() untuk mengaktifkan kelas NormalScanDocument sehingga halaman normal scan dokumen ditampilkan. Setelah form diisi, dan diklik button “Process”, aplikasi akan melakukan eksekusi fungsi processNormalScanning(). Di dalam fungsi ini dilakukan ekstraksi konten dokumen, hashing, dan perhitungan similaritas antar dokumen. Hasil akan ditampilkan pada kelas ResultOfScanning dengan melakukan eksekusi terhadap konstruktor ResultOfScanning(vectorResult) .



Gambar 3.39 *Sequence Diagram Normal Scan Similaritas Dokumen*

3) Setting Parameter

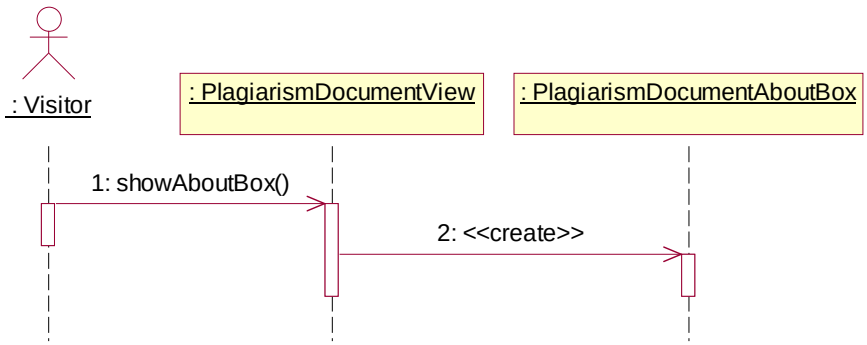
Halaman home aplikasi telah aktif. Ketika diklik menu "Setting Parameter", aplikasi akan menjalankan fungsi settingParameter. Fungsi ini mengaktifkan kelas PlagiarismDocumentView dengan mengeksekusi konstruktor SettingParameter() untuk menampilkan halaman setting parameter. Setelah form diisi dan diklik button "Update", fungsi updateParameter() dipanggil untuk merubah nilai parameter default yang telah didefinisikan sebelumnya.



Gambar 3.40 *Sequence Diagram Setting Parameter*

4) Lihat Bantuan

Halaman home aplikasi telah aktif. Ketika diklik menu “About”, aplikasi akan memanggil fungsi showAboutBox(). Fungsi ini mengaktifkan kelas PlagiarismDocumentAboutBox dengan melakukan eksekusi konstruktor PlagiarismDocumentAboutBox() untuk menampilkan informasi aplikasi.



Gambar 3.41 *Sequence Diagram Lihat Bantuan*

3.1.4.7 Identifikasi Kelas

Secara garis besar, dalam pemodelan rational unified process kelas dibagi menjadi tiga, yaitu *Boundary Class, Control Class dan Entity Class*. Berikut ini adalah identifikasi kelas berdasarkan proses (*use case*) yang dimiliki:

a. Identifikasi Kelas *Use Case* Admin

No.	***Use Case***	***Boundary***	***Control***	***Entity***
1.	Login	LoginForm	PlagiarismDocumentView	Login
2.	Lihat Profil	ProfilForm	PlagiarismDocumentView	-
3.	Kelola Admin	AdminList	PlagiarismDocumentView	Admin
4.	Input Admin	AddAdminForm	AdminList	Admin

5.	Edit Admin	EditAdminForm	AdminList	Admin
6.	Hapus Admin	AdminList	AdminList	Admin
7.	Kelola Dokumen	DocumentList	PlagiarismDocumentView	Document
8.	Input Dokumen	AddDocumentForm	DocumentList	Document
9.	Edit Dokumen	EditDocumentForm	DocumentList	Document
10.	Hapus Dokumen	DocumentList	DocumentList	Document
11.	Kelola Subdokumen	SubDocumentList	PlagiarismDocumentView	SubDocument
12.	Input Subdokumen	AddSubDocumentForm	SubDocumentList	SubDocument
13.	Lihat Konten	SubDocumentContent	SubDocumentList	-
14.	Edit Subdokumen	EditSubDocumentForm	SubDocumentList	SubDocument
15	Hapus Subdokumen	SubDocumentList	SubDocumentList	SubDocument

Tabel 3.4 Indentifikasi Kelas Use Case Admin

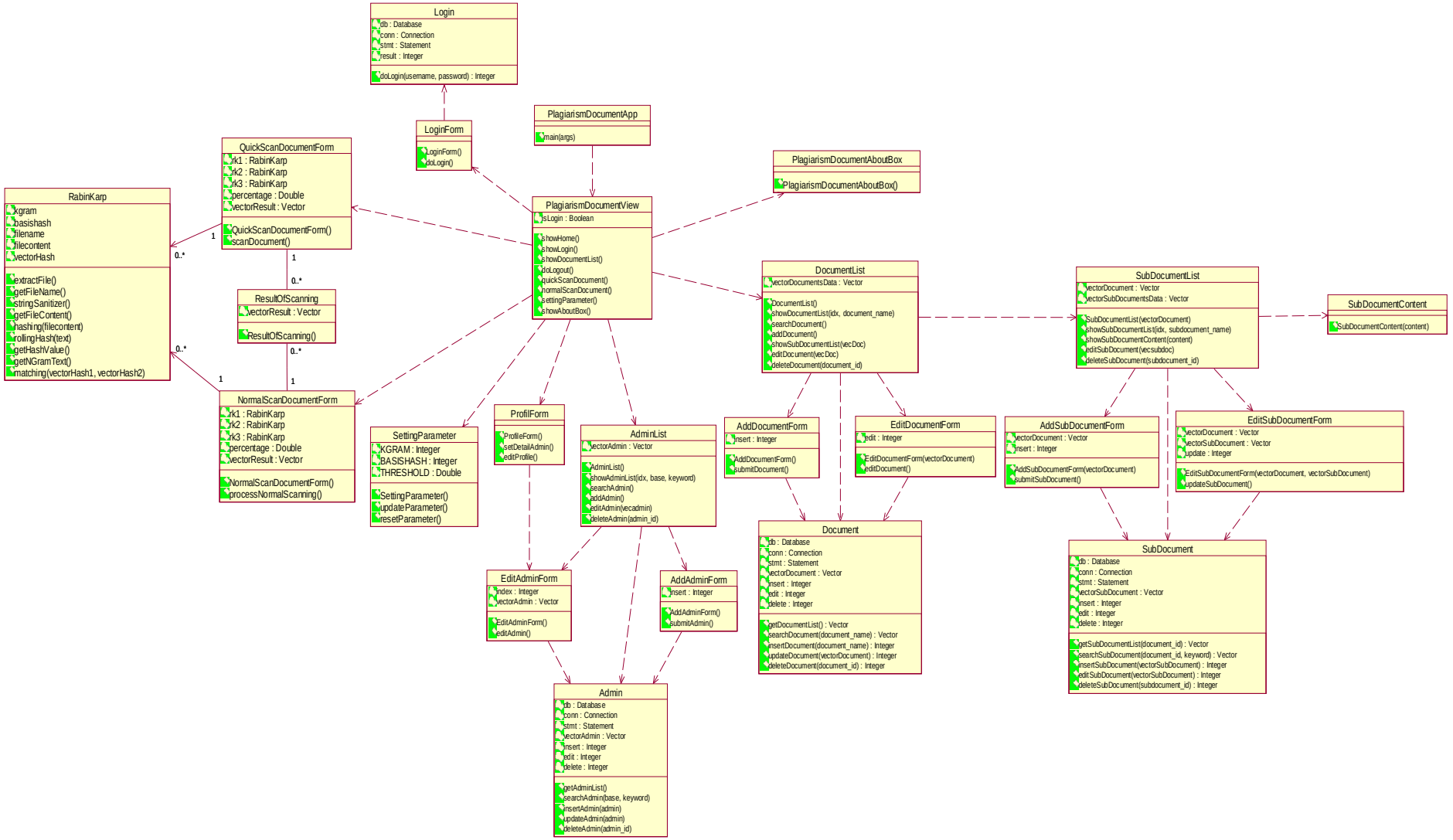
b. Identifikasi Kelas *Use Case* Visitor

No	***Use Case***	***Boundary***	***Control***	***Entity***
1	Quick Scan Similaritas Dokumen	QuickScanDocumentForm, ResultOfScanning	PlagiarismDocumentView	-
2	Normal Scan Similaritas Dokumen	NormalScanDocumentForm, ResultOfScanning	PlagiarismDocumentView	Document, SubDocument
3	Setting Parameter	SettingParameter	PlagiarismDocumentView	-
4	Lihat Bantuan	PlagiarismDocumentAboutBox	PlagiarismDocumentView	-

Tabel 3.5 Indentifikasi Kelas Use Case Visitor

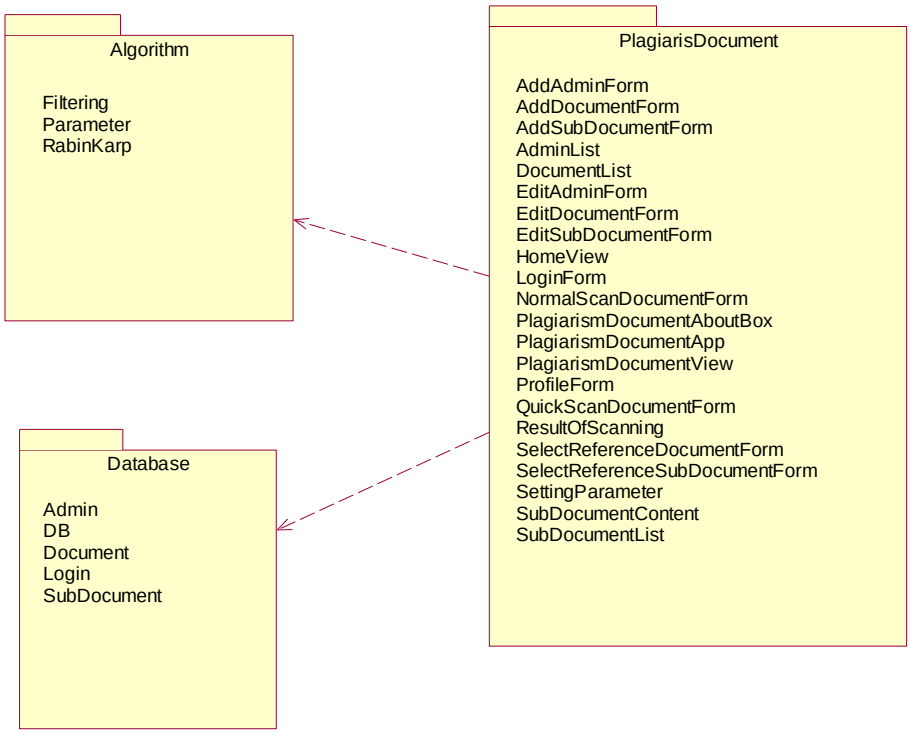
3.1.4.8 *Class Diagram*

Menggambarkan kelas dari suatu sistem, keterkaitan antar kelas, atribut dan operasi/fungsi dari sebuah kelas.



Gambar 3.42 *Class Diagram*

3.1.4.9 *Package Diagram*

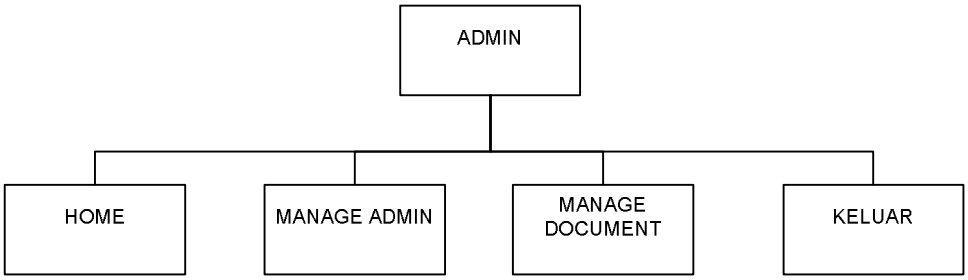


Gambar 3.43 Package Diagram

3.2 Desain Struktur Menu

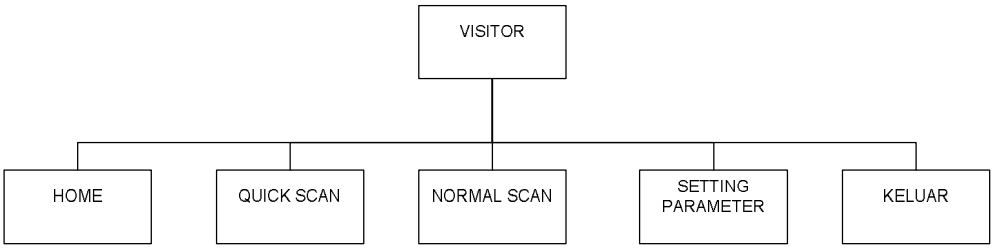
Perancangan struktur menu berisikan menu dan submenu yang berfungsi memudahkan *user* didalam menggunakan sistem. Beriktu adalah desain struktur menu yang akan diterapkan pada perangkat lunak.

1. **Struktur Menu Admin**



Gambar 3.44 Struktur Menu Admin

2. Struktur Menu Visitor

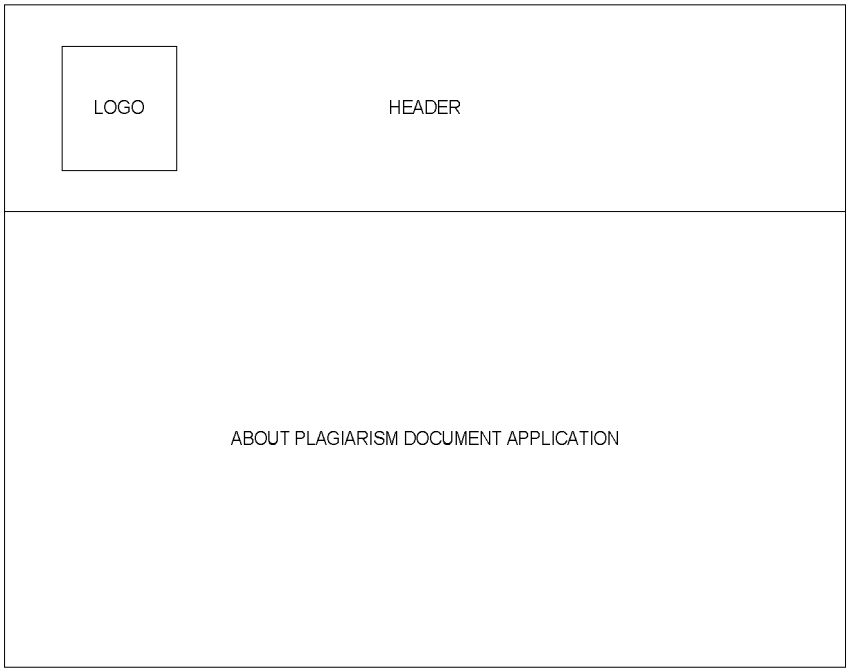


Gambar 3.45 Struktur Menu Visitor

3.3 Desain *Interface* (*User Interface*)

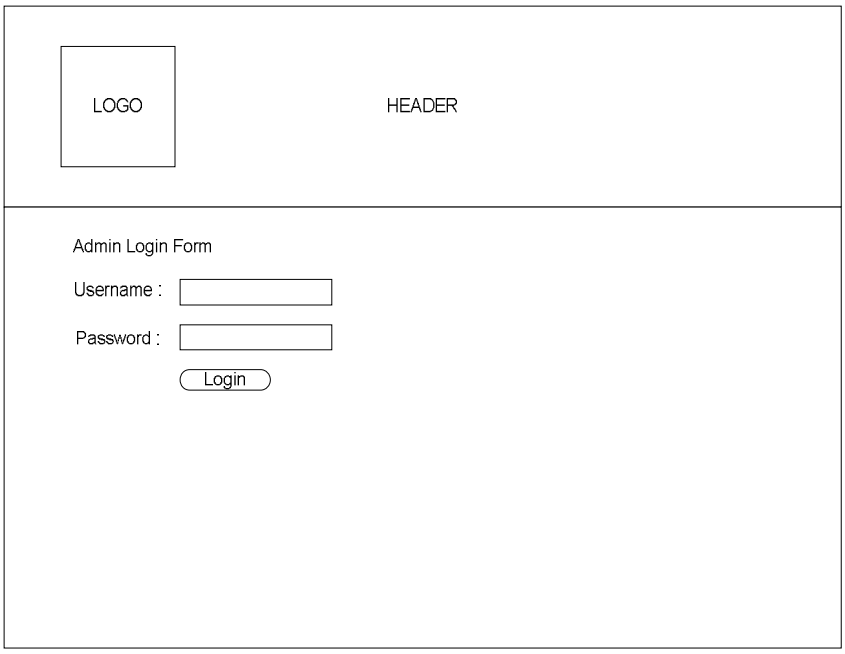
Rancangan tampilan (*user interface)* merupakan draft dari antarmuka yang akan diterapkan pada perangkat lunak.

a. User Interface Home



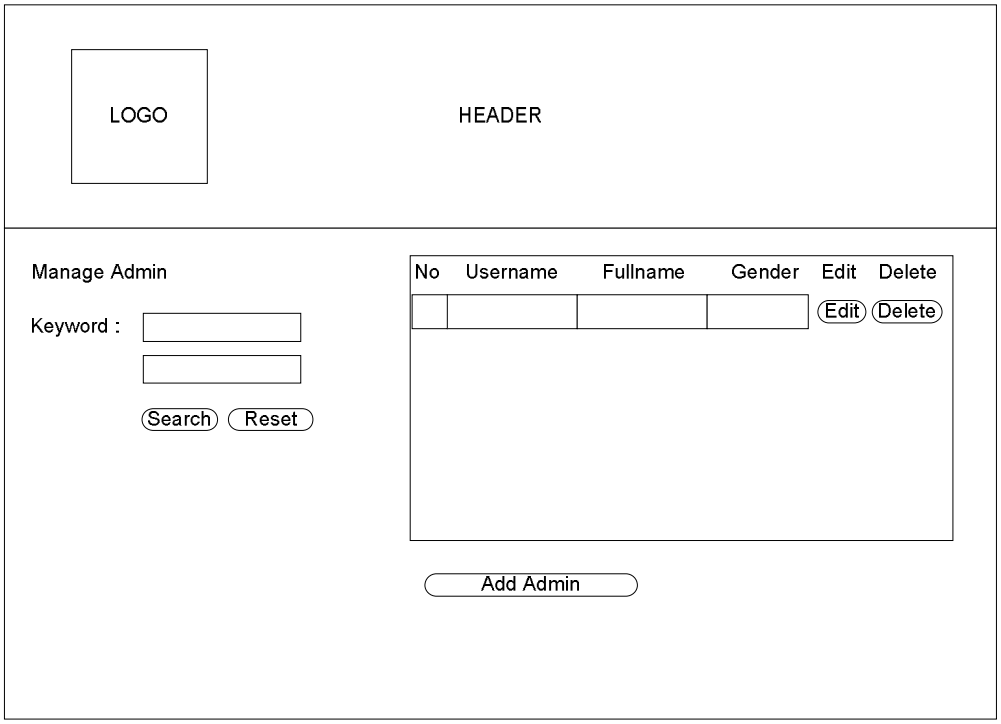
Gambar 3.46 *User Interface Home*

b. User Interface Login



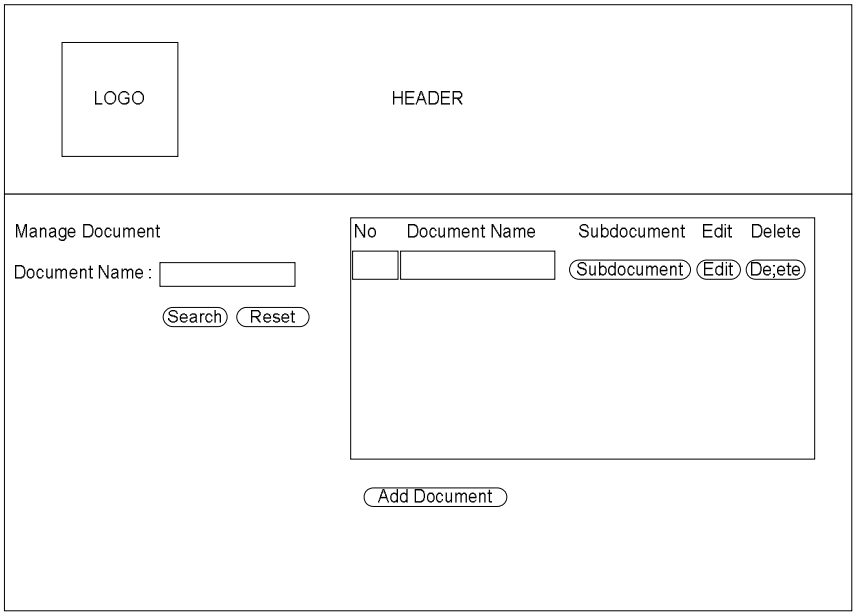
Gambar 3.47 *User Interface Login*

c. User Interface Kelola Admin



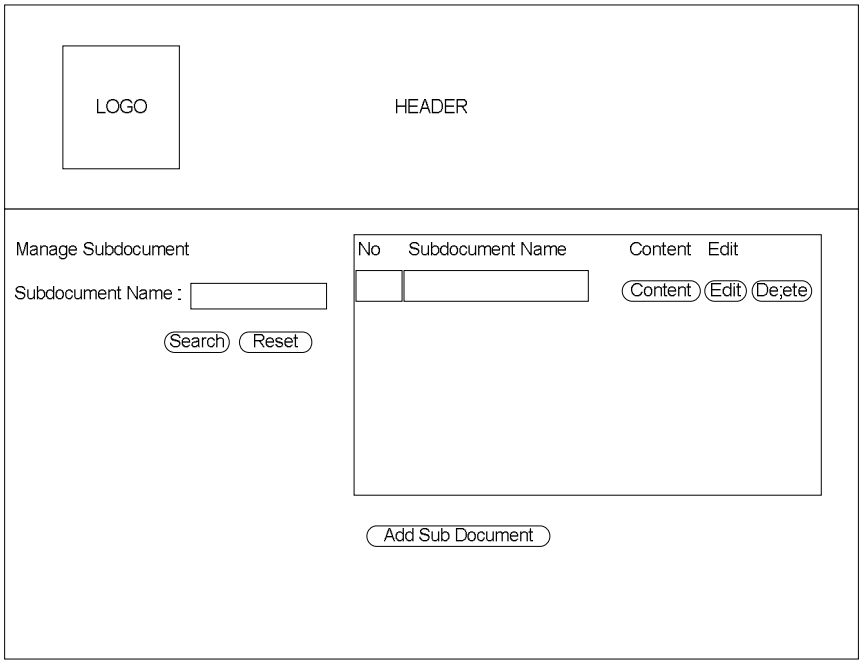
Gambar 3.48 *User Interface Kelola Admin*

d. User Interface Kelola Dokumen



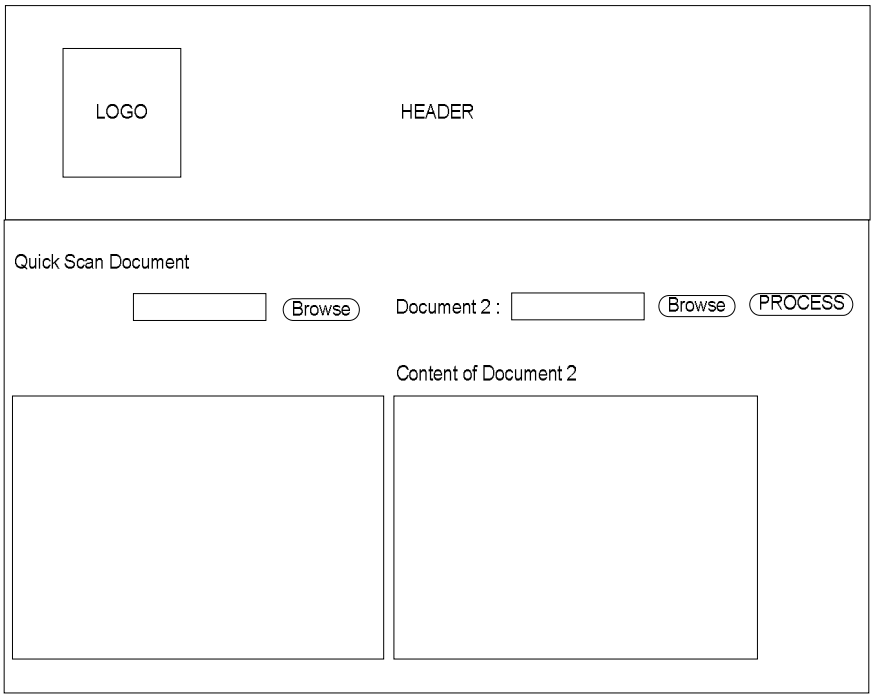
Gambar 3.49 *User Interface Kelola Dokumen*

e. User Interface Kelola Subdokumen



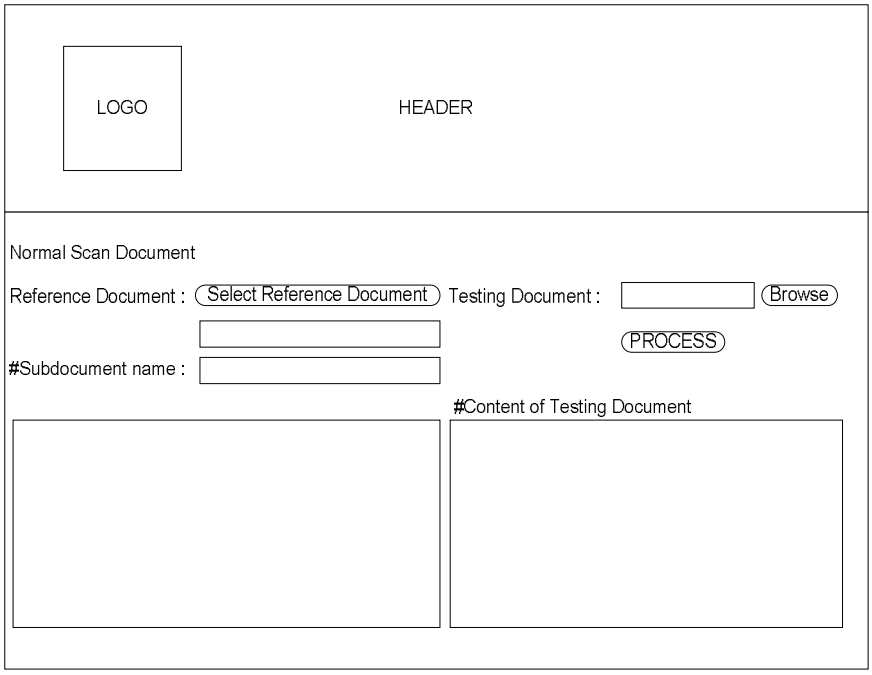
Gambar 3.50 *User Interface Kelola Subdokumen*

f. User Interface Quick Scan Dokumen



Gambar 3.51 *User Interface Quick Scan Dokumen*

g. User Interface Normal Scan Dokumen



Gambar 3.52 *User Interface Normal Scan Dokumen*