BAB III

ANALISA DAN PERANCANGAN SISTEM

Tahap analisis dan perancangan adalah suatu tahap penguraian dari suatu sistem kedalam bagian-bagian atau komponen-komponennya untuk mengidentifikasi dan mengevaluasi permasalahan yang ditemukan dan dibuatkan suatu usulan penyelesaian masalah dan pengembangan sistem yang sesuai.

Pada bab ini akan diuraikan kebutuhan-kebutuhan dari "Aplikasi Penyimpanan *File Online* Dengan Enkripsi Menggunakan Algoritma *Vertical Bit Rotation* (VBR)", baik kebutuhan perangkat lunak maupun kebutuhan perangkat keras. Dan mengadakan sebuah analisis mengenai algoritma kriptografi VBR sehingga ditemukan kekurangan-kekurangannya dan mencari solusi pemecahannya.

3.1 Analisis Masalah

Terdapat masalah-masalah seperti telah dijelaskan sebelumnya pada bab I maka perlu dianalisa satu per satu agar didapatkan solusi terbaik untuk pemecahan masalah-masalah tersebut.

Pertama dalam mengatasi permasalahan keamanan dalam menyimpan *file online* akan digunakan teknik kriptografi. Dengan menggunakan algoritma kriptografi, *file* yang disimpan akan disandikan dengan sebuah kunci yang hanya diketahui oleh orang yang menyimpannya. Kunci tidak disimpan dalam *database* sehingga kerahasiaannya terjamin.

Kedua, akan dibangun sebuah aplikasi penyimpanan *file online* berbasis *web*. *Web* dipilih karena memiliki kelebihan yaitu membuat aplikasi ini dapat berjalan pada semua *platform* sistem operasi. Aplikasi ini dapat berjalan pada beberapa macam *internet browser* yang merupakan aplikasi standard pada sebuah sistem operasi seperti Internet Explorer, Mozilla Firefox, Opera, Google Chrome dan Safari. Dengan menggunakan aplikasi berbasis *web*, pengguna tidak perlu lagi meng-*install* aplikasi tertentu untuk mengenkripsi *file* yang akan disimpan secara *online*.

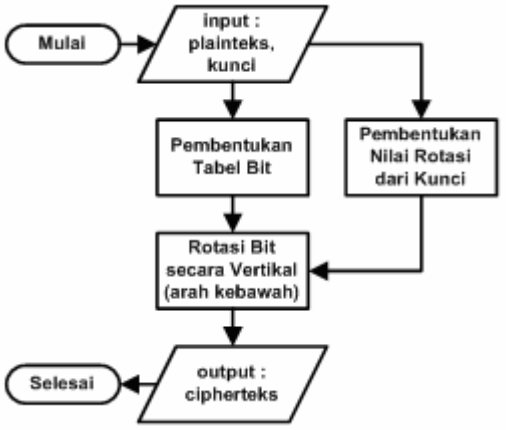
Ketiga, akan diambil data nilai *Avalanche Effect* dari ekperimen-eksperimen, sebagai indikator tingkat keamanan dari penyandian yang dilakukan. Dari nilai-nilai ini nantinya akan dapat ditarik kesimpulan sebagai tingkat keamanan algoritma kriptografi VBR.

3.2 Analisis Algoritma Kriptografi VBR

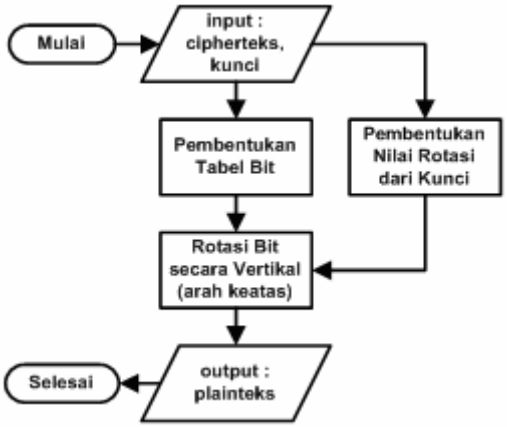
3.2.1 Prosedural Enkripsi Dan Dekripsi Algoritma Kriptografi VBR

Algoritma kriptografi VBR memiliki proses-proses yang sederhana dengan yaitu pembentukan tabel bit, pembentukan nilai variabel rotasi dan rotasi bit secara vertikal. Proses pembentukan nilai variabel rotasi dilakukan pada kunci, sedangkan pembentukan tabel bit dilakukan pada plainteks, sehingga dua proses ini dapat dilakukan secara paralel. Setelah dua proses itu selesai, barulah dapat dilakukan proses selanjutnya yaitu perputaran bit secara vertikal. Dalam algoritma kriptografi VBR tidak terdapat perulangan proses, masing-masing proses berjalan satu kali. Untuk lebih jelasnya dapat dilihat pada prosedural enkripsi dan dekripsi

yang digambarkan dalam bentuk *flowchart* seperti yang terlihat pada gambar 3.1 untuk proses enkripsi dan pada gambar 3.2 untuk proses dekripsi.



Gambar 3.1 Prosedural proses enkripsi algoritma kriptografi VBR

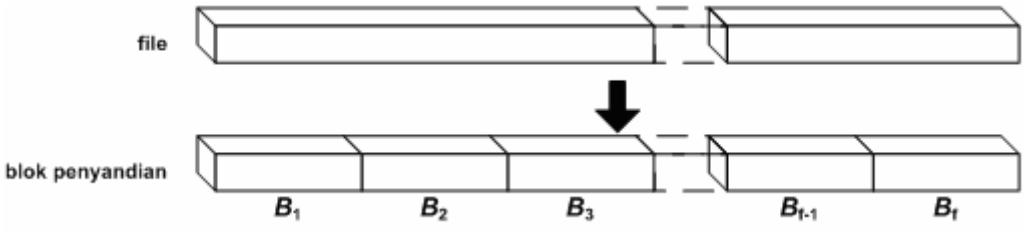


Gambar 3.2 Prosedural proses dekripsi algoritma kriptografi VBR

3.2.2 Analisis Blok Penyandian Pada Algoritma Kriptografi VBR

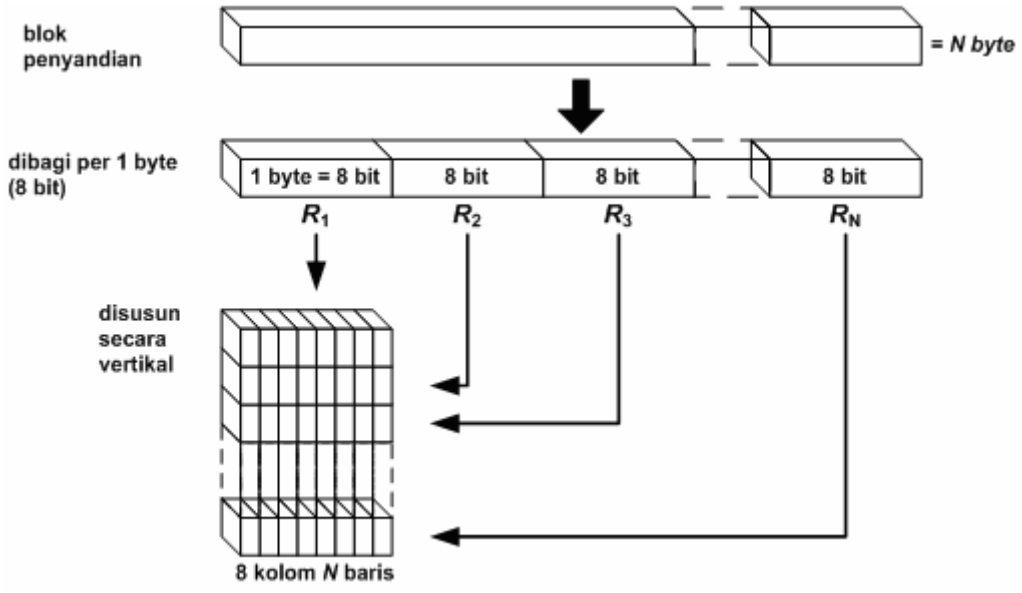
Algoritma Kriptografi VBR merupakan algoritma kriptografi simetri yang termasuk dalam kategori *block cipher*. Dalam algoritma ini dilakukan enkripsi atau dekripsi per blok penyandian (*B*) dengan ukuran blok penyandian maksimal 256 byte.

Apabila ukuran blok penyandian yang ditetapkan adalah *N byte*, maka akan dilakukan enkripsi maupun dekripsi setiap *N* byte dari *file* hingga pada *byte* terakhir dari *file*. Bila *f* adalah jumlah blok penyandian dari sebuah *file*, maka akan terjadi *f* kali iterasi proses enkrisi atau dekripsi. Ilustrasi pembentukan blok penyandian dapat dilihat pada gambar 3.3.



Gambar 3.3 Pembagian *file* kedalam blok penyandian

Dari satu blok penyandian, akan dibentuk sebuah tabel bit yang terdiri dari 8 bit kolom dan *N* bit baris. Caranya dengan memecah blok penyandian kedalam *N* buah bagian, dimana satu bagiannya memiliki besar 1 byte atau sama dengan 8 bit. Lalu bagian bagian tersebut disusun secara vertikal.



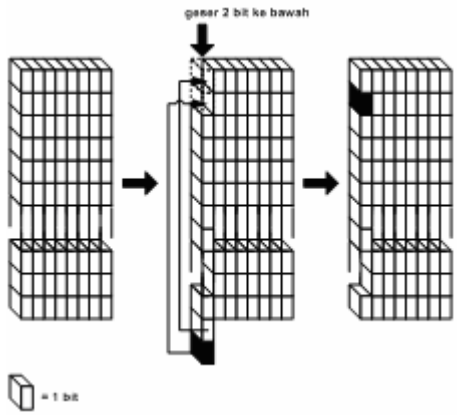
Gambar 3.4 Pembentukan tabel 8 kolom *N* baris

Untuk blok penyandian terakhir (B_f) tidak selalu berukuran *N* byte karena ukuran sebuah *file* tidak selalu merupakan kelipatan dari ukuran blok penyandian. Sehingga ada kemungkinan ukuran dari blok penyandian terakhir (B_f) kurang dari *N* byte.

Pada algoritma kriptografi VBR tidak ada proses *padding*, sehingga apabila ukuran blok yang akan disandikan kurang dari ukuran blok penyandian yang telah ditentukan, tidak akan dilakukan penambahan byte-byte *dummy* untuk melengkapi. Hal ini membuat algoritma kriptografi VBR tidak stabil. Misalnya blok penyandian hanya terdiri dari 1 byte, tabel bit hanya memiliki 1 baris bit, sehingga proses rotasi bit tidak dapat dilakukan.

3.2.3 Analisis Rotasi Bit pada Algoritma Kriptografi VBR

Proses enkripsi dan dekripsi pada algoritma kriptografi VBR adalah memutar (*rotate*) sejauh *r* bit secara vertikal bit-bit yang ada pada tabel 8 kolom *N* baris pada setiap blok penyandian. Memutar bit adalah menggeser rangkaian bit sehingga pada satu ujungnya terdapat bit yang berada diluar area rangkaian dan di ujung lainnya mengalami kekosongan. Lalu meletakkan bit yang berada diluar area rangkaian pada tempat yang kosong pada ujung lain dari rangkaian tersebut.



Gambar 3.5 Rotasi 2 bit ke bawah pada kolom pertama

Besar rotasi pada setiap kolom tidak selalu sama dan tidak selalu berbeda. Sehingga untuk melakukan enkripsi ataupun dekripsi diperlukan variabel rotasi untuk masing-masing kolom. Untuk mendapatkan variabel-variabel rotasi (*r*) ini diolah dari kunci penyandian.

Namun dalam kenyataannya, pergeseran bit sangat dipengaruhi oleh sistem operasi yang digunakan. Untuk sistem operasi 32 bit, pergeseran maksimal yang dapat dilakukan adalah sebesar 32 bit. Sedangkan pergeseran yang diharapkan terjadi pada proses rotasi vertikal maksimal adalah 256 bit. Maka ada ketidak-mampuan dalam melakukan proses bila algoritma kriptografi VBR dipaksakan untuk diterapkan pada mesin dengan sistem operasi 32 bit.

3.2.4 Analisis Kunci (*K*) dan Variabel Rotasi (*r*)

Proses rotasi pada setiap kolom besarnya tergantung pada variabel rotasi (*r*). Proses ini membutuhkan variabel-variabel rotasi untuk merotasi kolom bit pada tabel bit sesuai dengan jumlah kolom pada tabel bit. Ukuran maksimal dari blok penyandian (*N*) adalah 256. Angka ini dapat diwakili oleh karakter ASCII. Sehingga untuk mendapatkan variabel rotasi, bisa didapat dari nilai karakter ASCII yang dimasukkan melalui kunci. Dari pemikiran ini maka dapat disimpulkan bahwa dibutuhkan karakter-karakter ASCII yang diambil nilai desimalnya untuk menjadi nilai dari variabel-variabel rotasi yang dibutuhkan.

Jumlah kolom pada tabel bit adalah 8 kolom bit. Sehingga bila satu buah kolom membutuhkan satu buah nilai variabel rotasi, maka dibutuhkan 8 buah nilai variabel rotasi. Bila nilai variabel rotasi diambil dari nilai desimal masing-masing karakter pada kunci, maka dibutuhkan sebuah kunci dengan panjang 8 karakter.

Bila membandingkan dengan AES-128 yang menjadi standard penyandian saat ini, panjang kunci 8 karakter (64 bit) dapat dinilai tidak memenuhi standard, karena AES-128 menggunakan kunci dengan panjang 128 bit.

Sedangkan variabel rotasi (*r*) yang mengambil nilai desimal dari karakter ASCII kunci, tidak memiliki batasan tertentu. Sehingga *r* dapat bernilai nol (0) ataupun sama dengan ukuran blok penyandian. Apabila kondisi ini terjadi, maka tidak akan terjadi rotasi bit atau rotasi terjadi namun bit kembali pada tempat semula, dan blok penyandian tidak terenkripsi.

3.2.5 Tidak Ada Perubahan Bit pada Algoritma Kriptografi VBR

Sistem kerja algoritma kriptografi VBR adalah merotasi (menggeser) bit sehingga hanya ada perpindahan bit tanpa ada bit yang berubah. Perbandingan jumlah bit 1 (satu) dan jumlah bit 0 (nol) pada plainteks mapun cipherteks akan sama. Sehingga dapat disimpulkan apabila sebuah plainteks memiliki satu macam bit baik itu bit 1 (satu) maupun bit 0 (nol), bila dienkripsi dengan algoritma kriptografi VBR akan menghasilkan cipherteks yang sama dengan plainteks. Dengan kata lain plainteks tidak terenkripsi.

3.2.6 *Avalanche Effect* pada Algoritma Kriptografi VBR

Tingkat keamanan dari sebuah algoritma kriptografi dapat diukur dengan menggunakan parameter nilai *Avalanche Effect* [4]. Berikut akan dianalisa kemungkinan nilai avalanche effect pada algoritma kriptografi VBR.

1. Perbedaan bit pada kunci

   Dapat dipastikan, perubahan 1 bit pada kunci akan berpengaruh pada satu nilai variabel rotasi. Perubahan 1 buah nilai variabel rotasi

hanya akan berpengaruh pada 1 buah kolom bit dari total kolom bit ada 8 buah. Apabila *N* adalah jumlah baris pada tabel bit, dan maksimal perubahan bit pada satu kolom adalah *N* buah bit, maka nilai *Avalanche_Effect* maksimal (AE_{maks}) dapat dihitung sebagai berikut :

$$Avalanche_Effect_{maksimal}\left(AE_{maks}\right) = \frac{N}{8 * N} * 100\%$$

$$Avalanche_Effect_{maksimal}\left(AE_{maks}\right) = 12{,}5\%$$

2. Perbedaan bit pada plainteks

   Algoritma kriptografi VBR mengandalkan perpindahan bit tanpa merubah bit tersebut. Dapat dipastikan, perubahan 1 bit pada plainteks akan merubah 1 bit pada cipherteks. Apabila diambil ukuran blok terkecil, yaitu 1 byte (8 bit), maka nilai *Avalanche_Effect* maksimal (AE_{maks}) dapat dihitung sebagai berikut:

$$Avalanche_Effect_{maksimal}\left(AE_{maks}\right) = \frac{1}{8} * 100\%$$

$$Avalanche_Effect_{maksimal}\left(AE_{maks}\right) = 12{,}5\%$$

Dari analisa perhitungan *avalanche effect* baik dengan merubah bit pada kunci maupun merubah bit pada plainteks, didapatkan nilai *avalanche effect* maksimal adalah 12,5%. Dengan demikian dapat disimpulkan bahwa algoritma kriptografi VBR belum masuk dalam kategori algoritma kriptografi yang baik.

3.3 Modifikasi pada Algoritma Kriptografi VBR

Dari hasil analisa pada algoritma kriptografi VBR didapatkan beberapa kelemahan sebagai berikut :

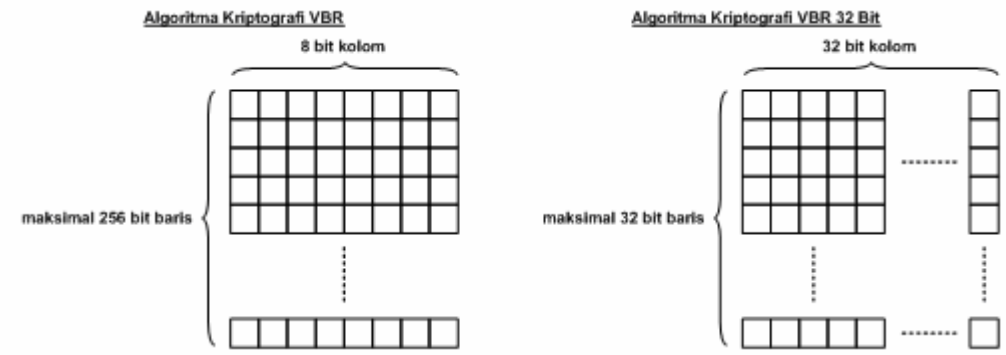
1. Tidak adanya *padding* menyebabkan blok penyandian dengan ukuran terkecil yaitu 1 byte (8 bit) tidak dapat dienkripsi.
2. Maksimal pergeseran bit, yaitu pergeseran 256 bit, tidak dapat dilakukan pada mesin dengan sistem operasi 32 bit.
3. Panjang kunci (8 byte) belum memenuhi standard.
4. Untuk semua nilai variabel rotasi 0 (nol) atau sama dengan ukuran blok penyandian menyebabkan plainteks tidak terenkripsi.
5. Algoritma kriptografi VBR tidak dapat diterapkan pada blok penyandian yang memiliki bit yang sama.
6. Maksimal nilai *avalanche effect* algoritma kriptografi VBR belum masuk dalam kategori baik.

Disebabkan adanya kelemahan-kelemahan pada algoritma kriptografi VBR maka akan dirancang sebuah modifikasi pada algoritma kriptografi VBR yang diberi nama Algoritma Kriptografi VBR 32 Bit.

3.3.1 Modifikasi Ukuran Tabel Bit

Tabel bit dibentuk dengan memiliki baris berjumlah maksimal 32 bit. Ukuran 32 bit adalah ukuran yang sangat sesuai untuk diterapkan, karena pada umumnya komputer masih menggunakan sistem operasi 32 bit. Disamping itu, dengan menerapkan 32 bit sebagai maksimal pergeseran bit, proses ini masih dapat bekerja pada sistem operasi 64 bit.

Modifikasi dalam pembuatan blok pada algoritma kriptografi VBR 32 bit dapat dilihat pada gambar 3.6.



Gambar 3.6 Perubahan metode pembentukan blok

Karena terjadi pengurangan jumlah baris pada tabel bit menjadi 32 bit baris, maka ukuran blok akan menjadi kecil bila mempertahankan lebar kolom yang hanya 8 bit. Solusinya adalah menambah ukuran dari kolom bit menjadi 32 bit seperti yang terlihat pada gambar 3.6. Perubahan ukuran tabel bit menyebabkan maksimal dari ukuran blok penyandian menjadi 128 byte.

Pemilihan sistem operasi tidak akan mempengaruhi penambahan ukuran kolom karena pergeseran bit dilakukan secara vertikal. Namun perubahan ini berpengaruh pada kebutuhan variabel rotasi yang yang juga bertambah sesuai dengan pertambahan jumlah kolom.

3.3.2 Penambahan *SubBlock* dan Proses *Padding*

Pada algoritma kriptografi VBR 32 bit ditambahkan istilah *SubBlock*. *SubBlock* merupakan bagian dari blok penyandian, yang besarnya 16 byte. Ukuran maksimal blok penyandian adalah 128 byte sehingga terdiri dari 8 *SubBlock*. Digunakan *SubBlock* bertujuan untuk menghindari penambahan byte *dummy* yang terlalu besar pada proses *padding*.

Sebelum melakukan proses enkripsi, perlu dipastikan bahwa ukuran blok penyandiaan merupakan kelipatan 16 byte (1 *SubBlock*). Teknik untuk memastikan ukuran blok digunakan operator modulo dan didapatkan sisa hasil bagi. Bila *N* adalah ukuran blok penyandian dengan satuan byte, maka nilai sisa hasil bagi dapat diperoleh dengan rumus :

$$S = N \text{ MOD } 16 \tag{3.1}$$

dimana : S = nilai sisa hasil bagi;

N = ukuran blok penyandian dalam satuan byte ($N > 0$).

Bila nilai *S* yang didapat sama dengan nol(0), maka tidak perlu dilakukan *padding*. Bila S lebih dari nol(0), maka akan dicari jumlah karakter *NULL* (N_{NULL}) yang akan ditambahkan serta ukuran baru dari blok penyandian dengan rumus :

$$N_{\text{NULL}} = 16 - S \tag{3.2}$$

$$N_{\text{pad}} = N + N_{\text{NULL}} \tag{3.3}$$

dimana : N_{NULL} = jumlah karakter *NULL* (jumlah *padding*);

S = nilai sisa hasil bagi;

N_{pad} = ukuran blok penyandian setelah proses padding (byte);

N = ukuran blok penyandian dalam satuan byte ($N > 0$).

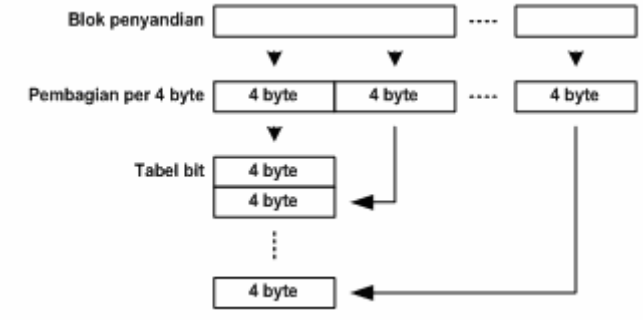
Jumlah baris (n_{B}) pada tabel bit setelah dilakukan proses padding dapat dihitung dengan menggunakan rumus :

$$n_{\text{B}} = N_{\text{pad}} / 4 \tag{3.4}$$

dimana : n_{B} = jumlah baris pada tabel bit;

N_{pad} = ukuran blok penyandian setelah proses padding (byte);

Susunan pembentukkan tabel bit pada algoritma kriptografi VBR 32 bit memiliki 32 kolom bit (4 byte) dan baris yang merupakan kelipatan 4 bit. Rangkaian blok penyandian akan dibagi kedalam per 32 bit, lalu disusun vertikal menjadi baris pertama, kedua dan seterusnya. Penggambaran pembentukan tabel bit dapat dilihat pada gambar 3.7.



Gambar 3.7 Pembentukan Tabel Bit

3.3.3 Aturan Tambahan Variabel Rotasi (*r*)

Untuk menghindari nilai variabel rotasi (*r*) adalah 0 (nol) atau sama dengan jumlah baris (n_B) pada tabel bit maka dibuat sebuah aturan tambahan untuk variabel rotasi. Bila $r = 0$ atau $r = n_B$, maka $r = 1$. Sehingga nilai *r* memiliki *range* antara 1 hingga n_B-1 ($0 < r < n_B$).

3.3.4 Proses Modulo Variabel Rotasi

Maksimal jumlah baris pada tabel bit adalah 32 baris. Proses *padding* tidak serta merta menjadikan jumlah baris menjadi 32 baris, hanya menjadikan kelipatan 4 baris, sehingga memiliki kemungkinan bahwa jumlah baris kurang dari 32. Hal ini dapat menyebabkan nilai variabel rotasi lebih besar daripada jumlah baris pada tabel bit. Agar kondisi ini tidak terjadi maka dilakukan proses Modulo Variabel Rotasi, yaitu mencari nilai modulus (sisa hasil bagi) dari

variabel rotasi (r) dengan jumlah baris (n_B) pada tabel bit sebagai angka pembagi, seperti pada rumus berikut :

$$r_n' = r_n \text{ MOD } n_B \quad (3.5)$$

dimana : r_n' = nilai variabel rotasi setelah proses Modulo;

r_n = nilai variabel rotasi sebelum proses Modulo;

n_B = jumlah baris pada tabel bit.

Sesuai dengan aturan tambahan untuk variabel rotasi, apabila r' bernilai 0 (nol) maka $r' = 1$.

3.3.5 Proses *Initial Rotation* dan Proses *Final Rotation*

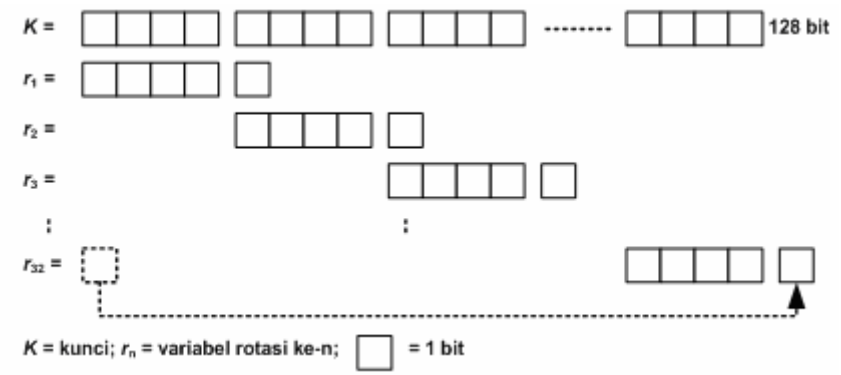
Proses Initial Rotation dan Final Rotation adalah proses rotasi secara vertikal masing-masing kolom pada tabel bit. Besar dari rotasi tergantung pada nilai variabel rotasi. Proses Initial Rotation terdapat pada proses enkripsi yang memiliki arah rotasi ke bawah, sedangkan proses Final Rotation terdapat pada proses dekripsi yang memiliki arah rotasi ke atas.

3.3.6 Modifikasi Kunci (*K*)

Pada algoritma kriptografi VBR 32 bit, besar maksimal pergeseran bit pada setiap kolom sebesar 31 bit. Namun karena jumlah kolom bit menjadi 32 kolom, maka dibutuhkan 32 nilai variabel rotasi. Bila mengambil nilai desimal dari karakter ASCII, akan menjadi terlalu besar, karena nilai maksimal karakter ASCII adalah 255. Untuk mendapatkan nilai maksimal 31, dibutuhkan angka biner dengan ukuran 5 bit. Setiap nilai variabel rotasi terdiri dari 5 bit, maka dibutuhkan 160 bit untuk mendapatkan 32 nilai variabel rotasi. 160 bit sama dengan 20 byte, atau bisa diwakili oleh 20 karakter ASCII 8 bit. Sehingga dapat dipakai kunci

berupa string dengan ukuran 20 karakter untuk mendapatkan 32 variabel rotasi. Atau bila dirasa akan ada kesulitan menggunakan kunci dengan panjang 20 byte, dapat pula digunakan 2 buah kunci dengan panjang masing-masing 10 byte. Hal ini hanya untuk mempermudah pengguna dalam menggunakan kunci.

Teknik kunci 20 byte ataupun 2 buah kunci masing-masing 10 tidak dipakai pada penelitian ini. Penelitian ini memilih sebuah alternatif untuk mendapatkan 32 variabel rotasi dari kunci sepanjang 128 bit (16 byte = 16 karakter). Bila 16 byte dibagi menjadi 32 bagian, maka masing-masing bagian akan terdiri dari 4 bit. Variabel rotasi terdiri dari 5 buah bit, maka 1 bit terakhir (bit ke-5) didapatkan dari bit awal bagian berikutnya. Pada variabel rotasi terakhir mengambil bit awal dari bagian pertama sebagai bit ke-5. Penggambaran untuk mendapatkan nilai variabel rotasi dapat dilihat pada gambar 3.14.



Gambar 3.8 Pembentukan variabel rotasi

3.3.7 Ekspansi Kunci (*K*), Proses *Add Key*, dan Proses *Add SubKey*

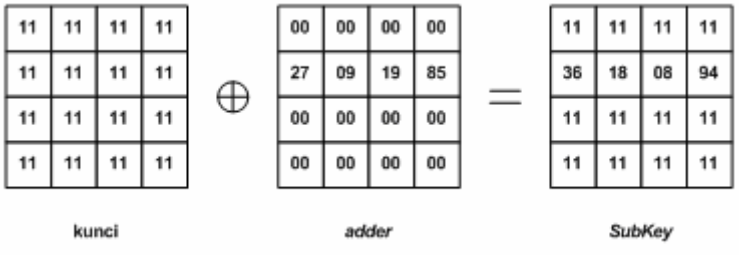
Ekspansi kunci dilakukan untuk membangkitkan 4 buah *SubKey* (*SK*), yang digunakan pada proses *Add SubKey*. Ekspansi kunci dilakukan dengan menambahkan angka hexadesimal ’27 09 19 85’$_H$ yang digeser kekiri (*bitwise*) sesuai kebutuhan pada kunci. Untuk mendapatkan *SubKey* ke *n* (SK_n), kunci

ditambah ’27 09 19 85’H yang digeser sejauh 4 – *n* byte, atau digeser (4-*n*) * 8 bit, kearah kiri.

$$SK_n = Key \oplus \left[27091985_H << \{(4-n)\times 8\}\right] \quad (3.6)$$

dimana : SK_n = SubKey ke n.

Misal kunci terdiri dari 16 byte ‘11_H’, akan membentuk *SubKey* kedua (SK_2), dapat dilihat seperti pada gambar 3.15.

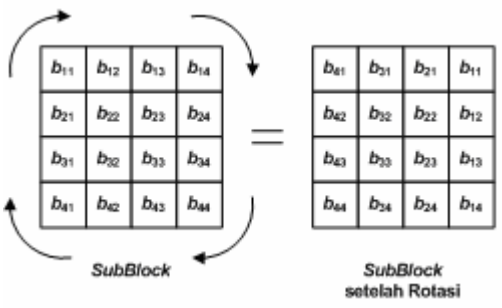


Gambar 3.9 Pembangkitan *SubKey*

Proses *Add Key* adalah penambahan *Key* (kunci) pada setiap *SubBlock* pada blok penyandian, dan proses *Add SubKey* adalah penambahan *SubKey* pada setiap *SubBlock*. *SubKey* dibangkitkan dari proses ekspansi kunci. Penambahan *SubKey* dilakukan urut mulai dari yang pertama hingga keempat, lalu berulang dari awal. Setiap *SubKey* ditambahkan pada setiap *SubBlock*. Untuk putaran berikutnya, penambahan *SubKey* dimulai dari *SubKey* berikutnya pula.

3.3.8 Proses *SubBlock Rotation*

Pada proses *SubBlock* Rotation, bila sebuah *SubBlock* digambarkan sebagai kotak byte 4x4 (*Square*), maka kotak tersebut diputar sebesar 90^o.



Gambar 3.10 Rotasi *SubBlock* searah jarum jam

Pada proses enkripsi pemutaran dilakukan searah jarum jam dan pada proses dekripsi dilakukan sebaliknya.

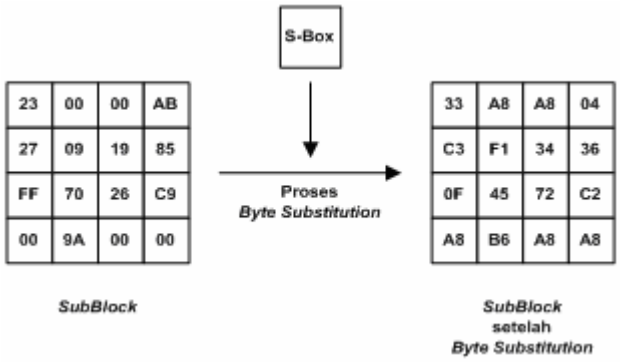
3.3.9 Proses *Byte Substitution*

Byte Substitution dilakukan dengan cara mengganti setiap byte pada *SubBlock* berdasarkan *S-Box* yang tersedia. *S-Box* biasa digunakan pada algoritma *blockcipher*, dan sangat berpengaruh pada kehandalan algoritma kriptografi. Pada algoritma kriptografi VBR 32 bit digunakan sebuah *S-Box* yang dibangkitkan secara involusi acak yang dapat dilihat pada tabel 3.1.

Tabel 3.1 *Substitution Box* (*S-Box*)

		y															
		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
x	0	A8	CD	57	40	AB	F9	ED	9D	AE	1D	E6	A9	17	B4	5F	FF
	1	35	3C	67	BA	F2	68	F4	0C	7A	34	58	8A	61	09	46	DC
	2	5A	2F	66	33	7E	64	72	C3	CB	B9	CE	83	4B	2E	2D	21
	3	5C	3B	69	23	19	10	85	A2	5D	D2	8B	31	11	F8	F3	AA
	4	03	76	EE	59	D0	70	1E	B3	81	75	88	2C	A0	98	FE	F5
	5	EF	B7	96	D7	E9	C0	A1	02	1A	43	20	E3	30	38	EA	0E
	6	BD	1C	B0	93	25	E2	22	12	15	32	B8	FB	8F	92	A5	A6
	7	45	9E	26	CC	E0	49	41	89	BB	D8	18	A3	80	D9	24	D4
	8	7C	48	D3	2B	E5	36	FC	DF	4A	77	1B	3A	BC	E7	9F	6C
	9	F1	C4	6D	63	E4	C5	52	DD	4D	C1	B6	F0	CA	07	71	8E
	A	4C	56	37	7B	B5	6E	6F	DE	00	0B	3F	04	AC	F7	08	D1
	B	62	D6	E1	47	0D	A4	9A	51	6A	29	13	78	8C	60	BE	E8
	C	55	99	C9	27	91	95	FA	CF	FD	C2	9C	28	73	01	2A	C7
	D	44	AF	39	82	7F	D5	B1	53	79	7D	DA	DB	1F	97	A7	87
	E	74	B2	65	5B	94	84	0A	8D	BF	54	5E	EC	EB	06	42	50
	F	9B	90	14	3E	16	4F	F6	AD	3D	05	C6	6B	86	C8	4E	0F

Bila *SubBlock* seperti pada gambar 3.11, maka dilakukan penggantian setiap byte pada *SubBlock* berdasarkan nilai byte tersebut. Anggap saja byte merupakan angka hexadesimal ‘xy’, maka akan dicari persamaannya pada *S-Box* dengan menelusuri nilai ‘x’ dan nilai ‘y’, dari byte pertama hingga byte terakhir.



Gambar 3.11 Proses *Byte Substitution*

Suatu misal, seperti pada gambar 3.11, byte pertama adalah “23_H”. Pada kondisi ini, $x = 2$ dan $y = 3$, kemudian dapat dilihat pasangan *xy* tersebut pada *S-Box* seperti yang terlihat pada tabel 3.1. Bila dilihat pada S-Box, maka “23_H” akan digantikan dengan “33_H”. Begitu seterusnya hingga pada byte terakhir.

3.3.10 Proses *Binary Rotation* dan Proses *Inverse Binary Rotation*

Proses *Binary Rotation* sama dengan proses *Initial Rotation*, yaitu merotasi secara vertikal kolom yang ada pada tabel bit. Pada proses *Binary Rotation*, besar rotasi sesuai *Binary Rotaion Variable* yang didapat dari jumlah bit ‘1’ pada kolom tersebut, memiliki arah rotasi ke bawah, dan digunakan dalam proses enkripsi.

Proses *Inverse Binary Rotation* memiliki kesamaan dengan proses *Binary Rotation*, hanya saja memiliki arah rotasi ke atas dan digunakan dalam proses dekripsi.

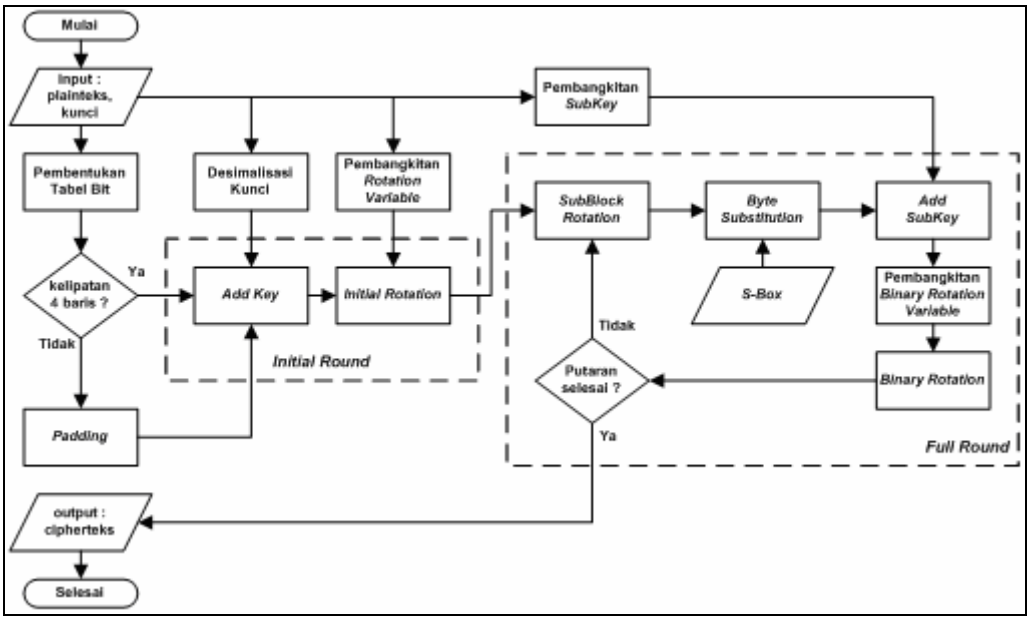
3.3.11 Putaran (*Round*) pada Algoritma Kriptografi VBR 32 Bit

Untuk menambah tingkat keamanan, didesain sebuah iterasi proses berupa putaran-putaran yang memiliki perulangan dinamik. Untuk proses enkripsi, diawali dengan *Initial Round* lalu diikuti *Full Round* dengan perulangan yang dapat ditentukan kemudian. Sedangkan pada proses dekripsi terdiri dari *Full Round* yang berulang dengan jumlah perulangan harus sama dengan jumlah perulangan *Full Round* pada proses enkripsi. Lalu proses dekripsi dilanjutkan dengan *Final Round* yang merupakan kebalikan dari proses *Initial Round* pada proses enkripsi.

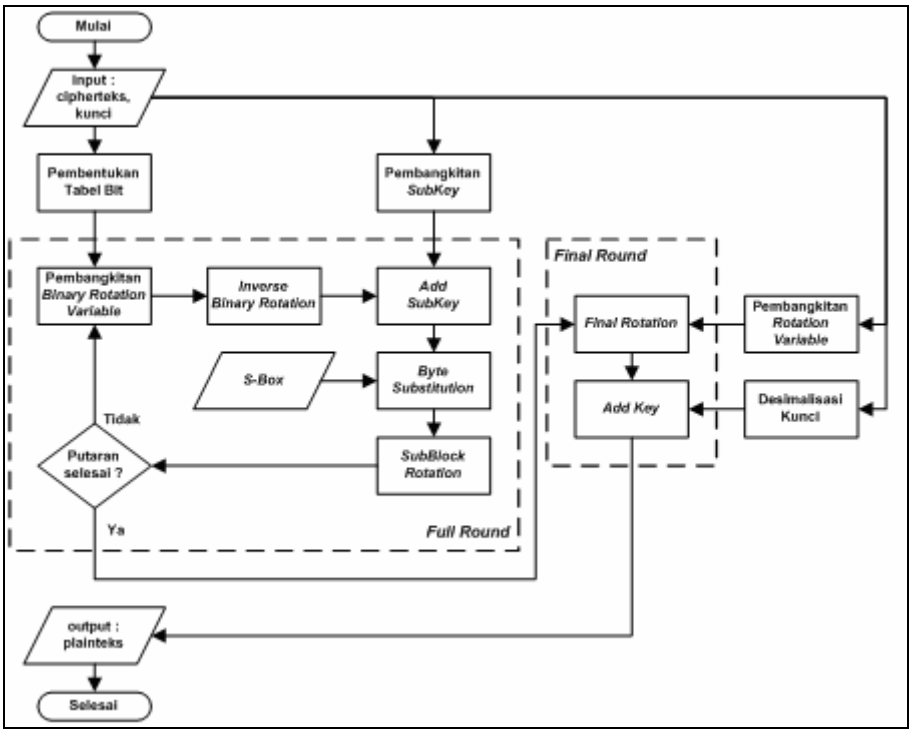
Untuk proses enkripsi, *Initial Round* terdiri dari proses *Add Key* dan proses *Initial Rotation*, sedangkan *Full Round* terdiri dari proses *SubBlock Rotation*, proses *Byte Substitution*, proses *Add SubKey* dan proses *Binary Rotation*. Sedangkan pada proses dekripsi, *Full Round* terdiri dari proses *Inverse Binary Rotation*, proses *Add SubKey*, proses *SubBlock Rotation*, proses *Byte Substitution*, dan *Final Round* terdiri dari proses *Final Rotation* dan proses *Add Key*.

3.3.12 Prosedural Enkripsi dan Dekripsi Algoritma Kriptografi VBR 32 Bit

Proses enkripsi dan proses dekripsi memiliki urutan yang berbeda. Dengan *flowchart*, proses enkripsi dan dekripsi algoritma kriptografi VBR 32 bit dapat tergambarkan proses demi proses sehingga dapat diketahui proses apa saja yang ada baik pada proses enkripsi maupun proses enkripsi. Berikut *flowchart* dari proses enkripsi yang dapat dilihat pada gambar 3.12. Sedangkan untuk *flowchart* dari proses dekripsi dapat dilihat pada gambar 3.13.



Gambar 3.12 Prosedural Proses Enkripsi

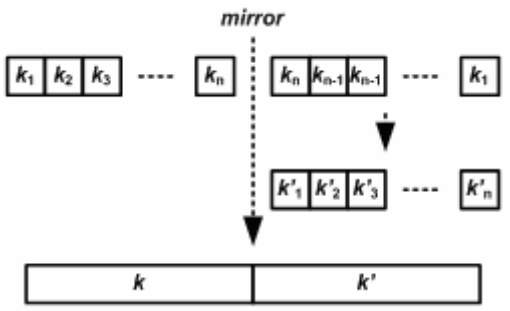


Gambar 3.13 Prosedural Proses Dekripsi

3.3.13 *Mirrored Key Generation* (MKG)

Mirrored Key Generation (MKG) adalah sebuah sistem perubahan kunci penyandian sehingga kunci memiliki ukuran yang sesuai dengan yang diinginkan.

Cara kerja MKG adalah menduplikasi secara terbalik kunci penyandian lalu menggabungkannya. Dengan MKG ukuran kunci akan menjadi dua kali lipat dari ukuran asal. Syarat dari MKG hanya boleh menduplikasi sebanyak satu kali. Hal ini untuk menjaga kekuatan kunci penyandian. Sehingga ukuran awal kunci penyandian minimal setengah dari ukuran kunci yang dibutuhkan. Dalam penelitian kali ini dibutuhkan kunci penyandian dengan ukuran 16 byte. Jadi minimal ukuran kunci yang harus dimasukkan menjadi 8 byte.



Gambar 3.14 Proses MKG

Inovasi MKG pada penelitian ini diterapkan untuk masukkan kata kunci pada Aplikasi Penyimpanan *File Online*, bukan penambahan proses pada Algoritma Kriptografi VBR. Inovasi MKG ini semata-mata untuk mendapatkan panjang kunci yang cocok dengan kebutuhan.

3.4 Analisis Pengguna

Karakteristik *user* dalam sistem berbasis web ini yang diutamakan adalah pengguna yang telah paham tentang dasar penggunaan komputer dan mengetahui tentang bagaimana menggunakan *internet browser*. Karena sistem ini masih menggunakan satu bahasa, yaitu Bahasa Indonesia, sehingga pengguna wajib dapat berbahasa Indonesia.

Dalam sistem ini ada dua macam *user*, yaitu *user* sebagai *administrator* dan *user* sebagai pengguna (penyimpan file online). Penjelasan dari kedua user tersebut adalah sebagai berikut :

1. *User* sebagai *Administrator*

   User Administrator memiliki wewenang untuk mengelola distribusi file. *Administrator* dapat menambah, mengubah, dan menghapus lokasi-lokasi pendistribusian *file online* yang disimpan oleh *user* pengguna.

2. *User* sebagai pengguna

   Pengguna yang ingin menyimpan *file*-nya dalam penyimpanan *file online*. *User* harus mendaftar terlebih dahulu sebagai anggota sehingga memiliki akun, agar dapat melakukan *login* sebagai syarat utama dalam proses menyimpan ataupun mengambil *file online*.

3.5 Analisis Perangkat Lunak

Analisis perangkat lunak disini adalah pembahasan tentang kerja sistem yang akan dibuat kemudian difokuskan pada perangkat lunak. Perangkat lunak yang digunakan dalam pembuatan sistem yaitu sebagai berikut :

1. Sistem Operasi Microsoft Windows Vista Basic
2. Notepad++ v5.5.1 (UNICODE) sebagai *script editor*
3. XAMPP 7.1 sebagai *web server* (Apache *server* dan MySQL *server*)
4. MDaemon 4.0.2 sebagai *Mail server*
5. Cerberus FTP Server 1.71 sebagai FTP *server*
6. Paint.net sebagai *tools* manipulasi gambar.

7. Flock, Google Chrome, Internet Explorer, Mozilla Firefox, Opera, Safari sebagai *internet browser*.

3.6 Analisis Perangkat Keras

Analisis perangkat keras disini adalah pembahasan tentang kerja sistem yang akan dibuat kemudian difokuskan pada perangkat keras. Spesifikasi minimum yang digunakan dalam menjalankan aplikasi ini adalah :

1. Seperangkat *Personal Computer* (PC) dengan spesifikasi sebagai berikut :
   a. *Processor* 1,8Ghz atau lebih baik
   b. *Harddisk* 40 GB
   c. *Memory Card* (RAM) 256 MB
   d. Monitor
   e. *Keyboard* dan *Mouse*
2. Modem atau *ethernet card* dipergunakan sebagai koneksi internet

3.7 Gambaran Umum Sistem

Sistem yang akan dibangun ini digunakan sebagai aplikasi untuk menyimpan dan mengambil *file* secara *online*, dan dapat mengenkripsi *file* pada saat penyimpanan *file* serta mendekripsi *file* pada saat *file* tersebut diambil dari penyimpanan *file online*. Proses enkripsi ataupun dekripsi terjadi di *server*, sehingga *user* tidak perlu meng-*install software* untuk melakukan enkripsi ataupun dekripsi *file*. *File* tersimpan dalam penyimpanan *file online* dalam keadaan terenkripsi, dan kunci penyandian tidak disimpan pada database.

3.8 Perancangan Sistem

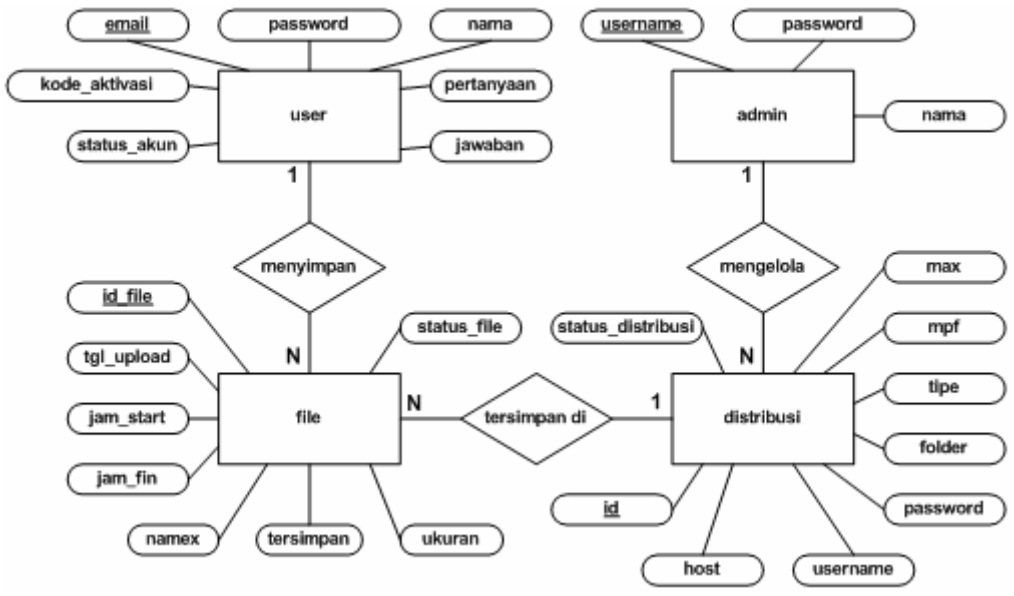
3.8.1 Perancangan Model Sistem

3.8.1.1 *Entity Relationship Diagram*

Pada sistem yang akan dibangun memiliki 4 entitas yang berada dalam sistem. Entitas tersebut adalah :

1. *User* (*user* sebagai pengguna)
2. Admin (*user* sebagai *administrator*)
3. *File* (*file* yang akan disimpan secara *online*)
4. Distribusi (data lokasi penyimpanan *file*)

Masing-masing entitas memiliki properti dan memiliki hubungan antar entitas. Properti dan hubungan antar entitas dapat digambarkan sebagai *entity relationship diagram* dari sistem ini, dapat dilihat pada gambar 3.21.

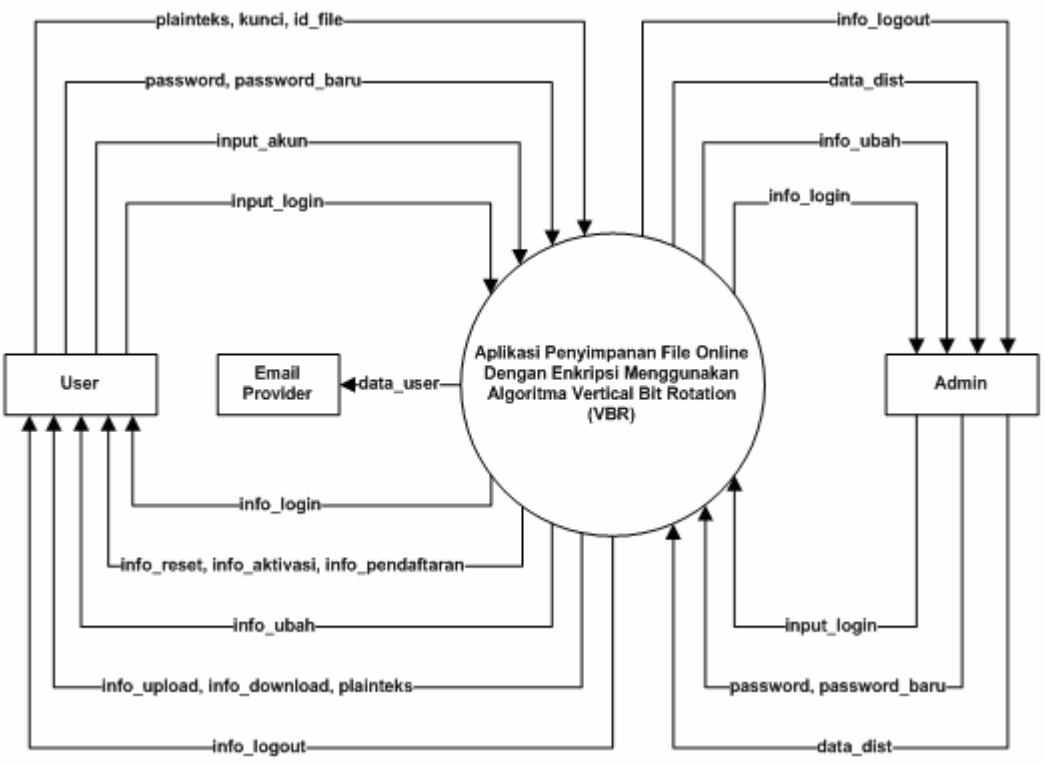


Gambar 3.15 *Entity Relationship Diagram*

3.8.1.2 *Data Flow Diagram*

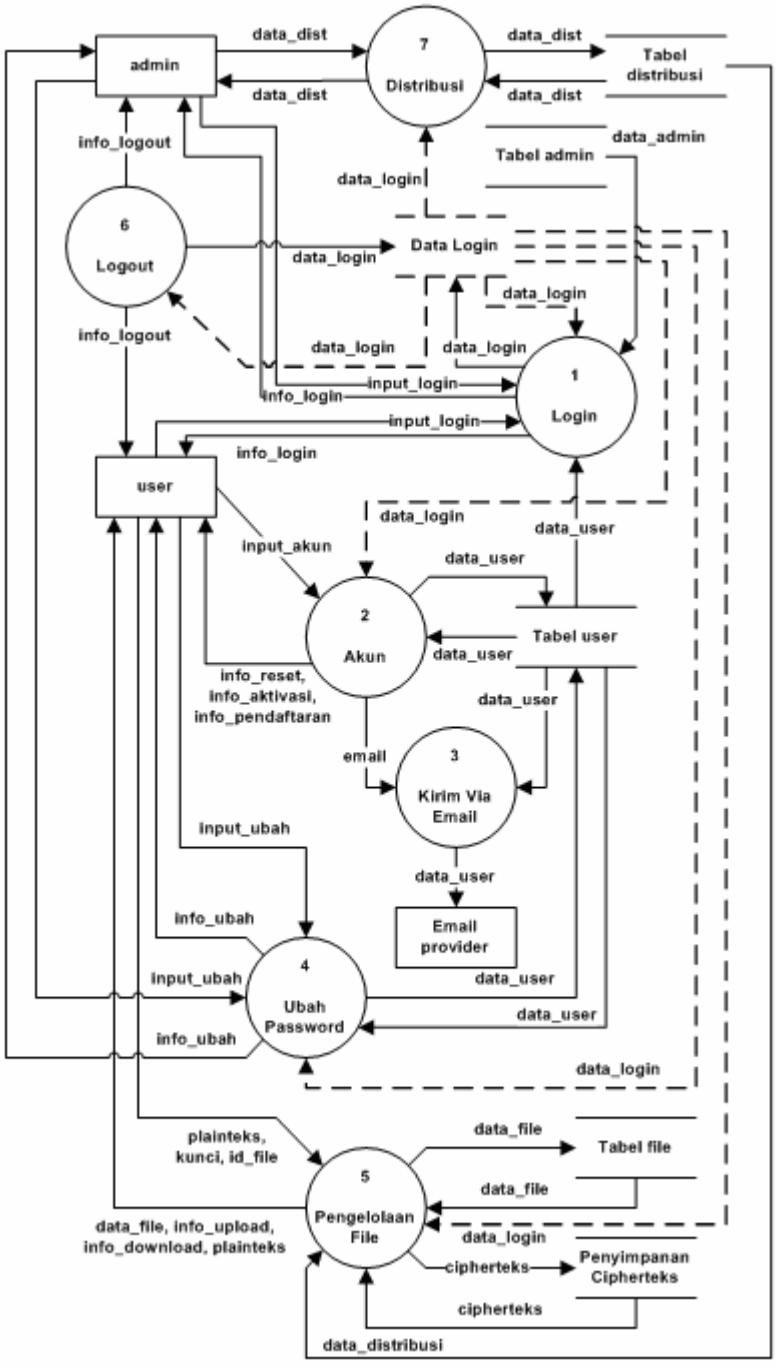
Dalam pembangunan aplikasi dibutuhkan sebuah pemodelan sistem. Salah satu pemodelan sistem yang banyak digunakan adalah *Data Flow Diagram* [8]. *Data Flow Diagram* adalah representasi grafis yang menggambarkan aliran data dan proses perubahan data dari input menjadi output [6].

1. Diagram Konteks / DFD Level 0



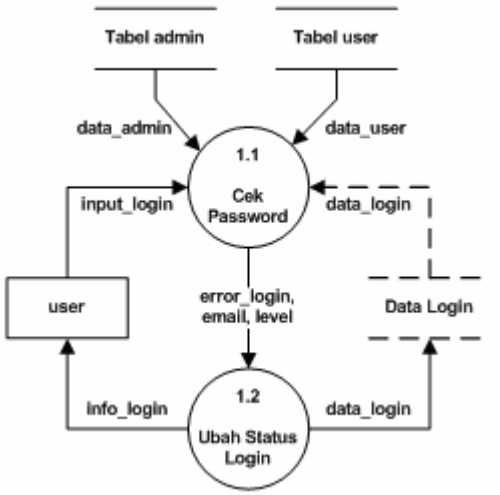
Gambar 3.16 Diagram Konteks / DFD Level 0

2. DFD Level 1



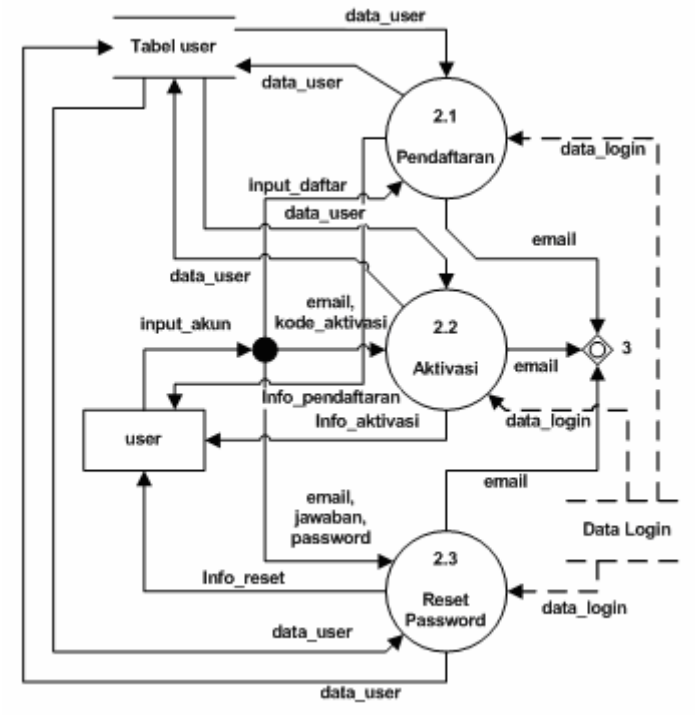
Gambar 3.17 DFD Level 1

3. DFD Level 2 Proses 1 Login



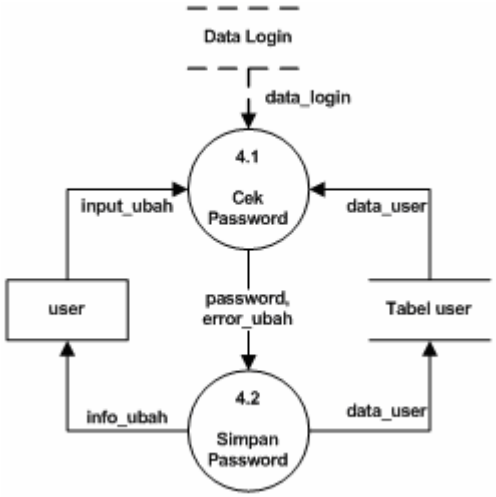
Gambar 3.18 DFD Level 2 Proses 1 Login

4. DFD Level 2 Proses 2 Akun



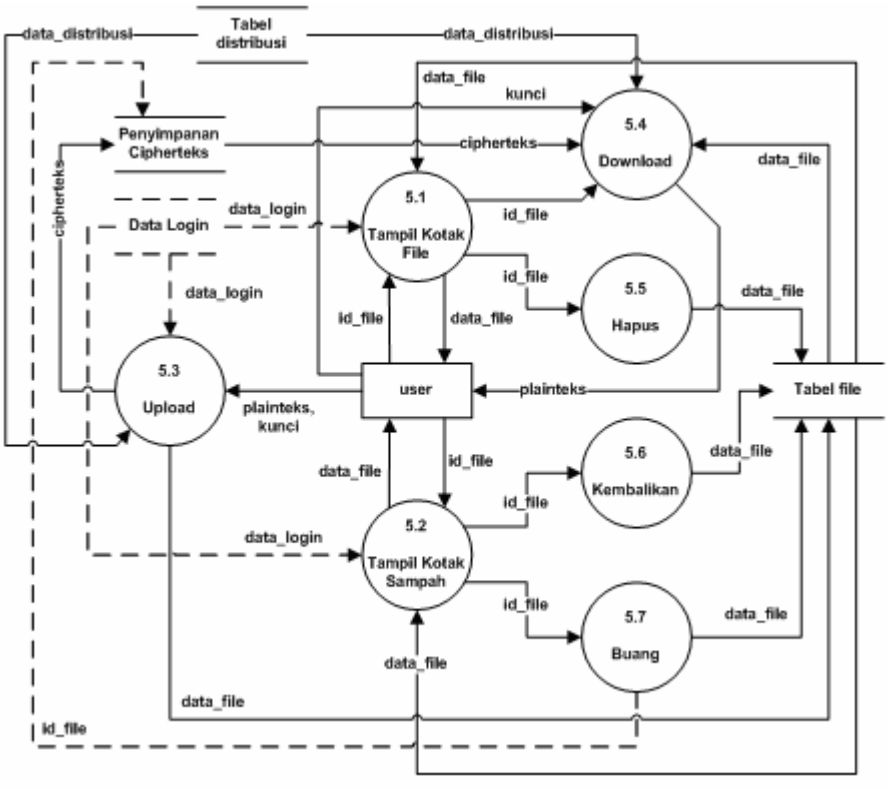
Gambar 3.19 DFD Level 2 Proses 2 Akun

5. DFD Level 2 Proses 4 Perubahan Password



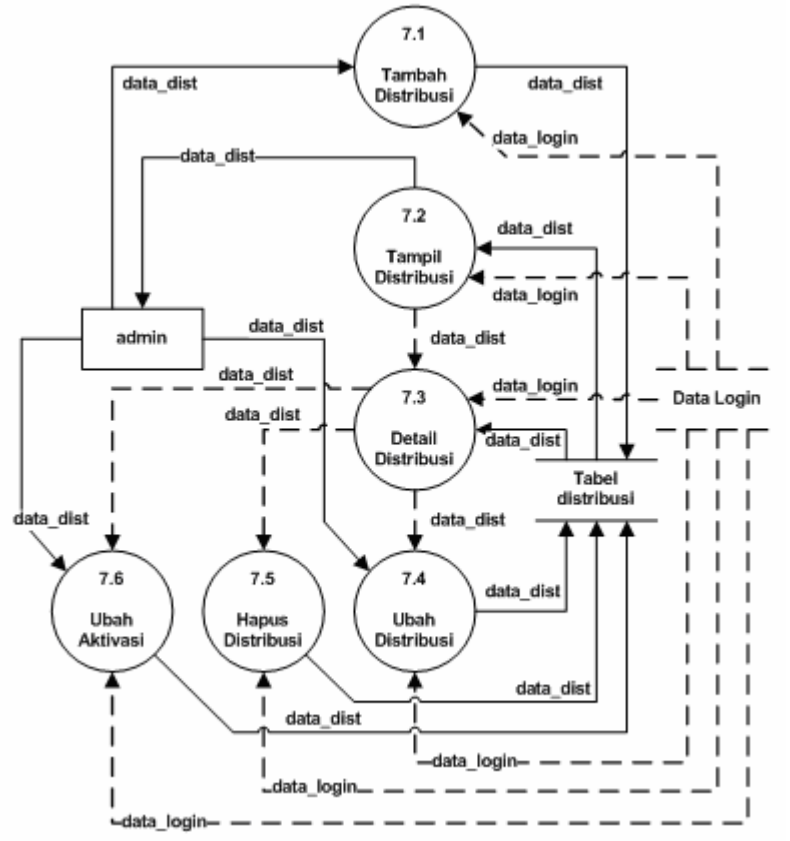
Gambar 3.20 DFD Level 2 Proses 4 Perubahan Password

6. DFD Level 2 Proses 5 Pengelolaan File



Gambar 3.21 DFD Level 2 Proses 5 Pengelolaan *File*

7. DFD Level 2 Proses 7 Distribusi



Gambar 3.22 DFD Level 2 Proses 7 Distribusi

3.8.1.3 Kamus Data

Tabel 3.2 Kamus data

Nama	**input_login**
Penggunaan	- Level 1 Proses 1 Login (*input*) - Level 2 Proses 1.1 Cek Password (*input*)
Deskripsi	Berisi data yang dibutuhkan untuk melakukan proses Login
input_login email password	= email + password = [A...Z | a...z | @ | . | _] = { *any characters* }
Nama	**data_login**
Penggunaan	- Level 1 Proses 1 Login (*input*) - Level 2 Proses 1.1 Cek Password (*input*) - Level 1 Proses 2 Akun (*input*) - Level 2 Proses 2.1 Pendaftaran (*input*) - Level 2 Proses 2.2 Aktivasi (*input*)

	- Level 2 Proses 2.3 Reset Password (*input*) - Level 1 Proses 4 Perubahan Password (*input*) - Level 2 Proses 4.1 Cek Password (*input*) - Level 1 Proses 5 Pengelolaan File (*input*) - Level 2 Proses 5.1 Tampil Kotak File (*input*) - Level 2 Proses 5.2 Tampil Kotak Sampah (*input*) - Level 2 Proses 5.3 Upload (*input*) - Level 1 Proses 7 Distribusi (*input*) - Level 2 Proses 7.1 Tambah Distribusi (*input*) - Level 2 Proses 7.2 Tampil Distribusi (*input*) - Level 2 Proses 7.3 Detail Distribusi (*input*) - Level 2 Proses 7.4 Ubah Distribusi (*input*) - Level 2 Proses 7.5 HapusDistribusi (*input*) - Level 2 Proses 7.6 Ubah Aktivasi (*input*) - Level 1 Proses 6 Logout (*input*) - Level 1 Proses 1 Login (*output*) - Level 2 Proses 1.2 Ubah Status Login (*output*) - Level 1 Proses 6 Logout (*output*)
Deskripsi	Berisi data status *login*, telah *login* atau belum, identitas *user* yang telah login beserta nama, serta level yang menunjukkan apakah *user* tersebut sebagai *user* pengguna atau sebagai *user administrator*.
data_login status email nama level	= status + email + nama = [*Null* | 'OK'] = [*Null* | A...Z | a...z | @ | .] = [*Null* | A..Z | a..z | *space*] = [*Null* | 0..1]
Nama	**info_login**
Penggunaan	- Level 1 Proses 1 Login (*output*) - Level 2 Proses 1.2 Ubah Status Login (*output*)
Deskripsi	Informasi keberhasilan melakukan *login* : 0 = sukses; 1 = akun belum aktif; 2 = email/username belum terdaftar
info_login	= [0..2]
Nama	**data_admin**
Penggunaan	- Level 1 Proses 1 Login (*input*) - Level 2 Proses 1.1 Cek Password (*input*)
Deskripsi	Berisi data sesuai dengan tabel admin.
data_admin username password nama	= username + password + nama = [A...Z | a...z] = {*any charactes*} = [A..Z | a..z | *space*]

Nama	**data_user**
Penggunaan	- Level 1 Proses 1 Login (*input*) - Level 2 Proses 1.1 Cek Password (*input*) - Level 1 Proses 2 Akun (*input*) - Level 2 Proses 2.1 Pendaftaran (*input*) - Level 2 Proses 2.2 Aktivasi (*input*) - Level 2 Proses 2.3 Reset Password (*input*) - Level 1 Proses 3 Kirim Via Email (*input*) - Level 1 Proses 4 Ubah Password (*input*) - Level 1 Proses 4.1 Cek Password (*input*) - Level 1 Proses 2 Akun (*output*) - Level 2 Proses 2.1 Pendaftaran (*output*) - Level 2 Proses 2.2 Aktivasi (*output*) - Level 2 Proses 2.3 Reset Password (*output*) - Level 1 Proses 4 Ubah Password (*output*) - Level 1 Proses 4.2 Simpan Password (*input*)
Deskripsi	Berisi data sesuai dengan tabel user.
data_user	= email + password + nama + kode_aktivasi + pertanyaan + jawaban + status_akun
email	= [A...Z | a...z | @ | .]
password	= {*any charactes*}
nama	= [A..Z | a..z | *space*]
kode_aktivasi	= [0..9]
pertanyaan	= { *legal charactes* }
jawaban	= { *legal charactes* }
status_akun	= [0..1]
Nama	**input_akun**
Penggunaan	- Level 1 Proses 2 Akun (*input*) - Level 2 Proses 2.1 Pendaftaran (*input*) - Level 2 Proses 2.2 Aktivasi (*input*) - Level 2 Proses 2.3 Reset Password (*input*)
Deskripsi	Berisi data user untuk keperluan pengelolaan akun mulai dari pendaftaran awal, aktivasi akun, reset password hingga ubah password.
input_akun	= input_daftar + email + kode_aktivasi + jawaban + password
input_daftar	= email + password + nama + pertanyaan + jawaban
email	= [A...Z | a...z | @ | .]
password	= {*any charactes*}
nama	= [A..Z | a..z | *space*]
kode_aktivasi	= [0..9]
pertanyaan	= { *legal charactes* }
jawaban	= { *legal charactes* }

Nama	**info_pendaftaran**
Penggunaan	- Level 1 Proses 2 Akun (*output*) - Level 2 Proses 2.1 Pendaftaran (*output*)
Deskripsi	Informasi keberhasilan melakukan pendaftaran : 0 = sukses; 1 = email terdaftar, akun belum aktif; 2 = akun aktif;
info_pendaftaran	= [0..2]
Nama	**info_aktivasi**
Penggunaan	- Level 1 Proses 2 Akun (*output*) - Level 2 Proses 2.2 Aktivasi (*output*)
Deskripsi	Informasi keberhasilan melakukan aktivasi : 0 = sukses; 1 = email belum terdaftar; 2 = akun sudah aktif; 3 = kode aktivasi salah;
info_aktivasi	= [0..3]
Nama	**info_reset**
Penggunaan	- Level 1 Proses 2 Akun (*output*) - Level 2 Proses 2.3 Reset Password (*output*)
Deskripsi	Informasi keberhasilan reset password : 0 = sukses; 1 = email belum terdaftar; 2 = akun belum aktif; 3 = jawaban salah;
info_reset	= [0..3]
Nama	**input_ubah**
Penggunaan	- Level 1 Proses 4 Ubah Password (*input*) - Level 2 Proses 4.1 Cek Password (*input*)
Deskripsi	Berisi password lama untuk klarifikasi dan password baru yang akan disimpan.
input_ubah password password_baru	= password + password_baru = {*any charactes*} = {*any charactes*}
Nama	**info_ubah**
Penggunaan	- Level 1 Proses 4 Ubah Password (*output*) - Level 2 Proses 4.2 Simpan Password (*output*)
Deskripsi	Informasi keberhasilan perubahan password : 0 = sukses; 1 = salah password;
info_ubah	= [0..1]
Nama	**data_file**
Penggunaan	- Level 1 Proses 5 Pengelolaan File (*input*) - Level 2 Proses 5.1 Tampil Kotak File (*input*)
	- Level 2 Proses 5.2 Tampil Kotak Sampah (*input*) - Level 2 Proses 5.3 Upload (*input*)

	- Level 2 Proses 5.4 Download (*input*) - Level 1 Proses 5 Pengelolaan File (*output*) - Level 2 Proses 5.3 Upload (*output*) - Level 2 Proses 5.5 Hapus (*output*) - Level 2 Proses 5.6 Kembalikan (*output*) - Level 2 Proses 5.7 Buang (*output*)
Deskripsi	Berisi data sesuai dengan tabel file.
data_file id_file tgl_upload jam_start jam_fin email id_distribusi namex ukuran tersimpan status_file	= id_file + tgl_upload + email + id_distribusi + namex + ukuran + tersimpan + status_file = [0..9] = { *date format* (yyyy-mm-dd) } = { *time format* (hh:mm:ss) } = { *time format* (hh:mm:ss) } = [A...Z | a...z | @ | .] = [0..9] = { *legal filename* } = [0..9] = [0..9] = [0..2]
Nama	**plainteks**
Penggunaan	- Level 1 Proses 5 Pengelolaan File (*input*) - Level 2 Proses 5.3 Upload (*input*) - Level 1 Proses 5 Pengelolaan File (*output*) - Level 2 Proses 5.4 Download (*output*)
Deskripsi	File yang akan masuk dalam proses enkripsi dan disimpan pada penyimpanan *on-line*, dan hasil proses dekripsi yang diambil dari penyimpanan.
plainteks	= { *file* }
Nama	**cipherteks**
Penggunaan	- Level 1 Proses 5 Pengelolaan File (*input*)
	- Level 2 Proses 5.4 Download (*input*) - Level 1 Proses 5 Pengelolaan File (*output*) - Level 2 Proses 5.3 Upload (*output*)
Deskripsi	File yang telah dienkripsi dan disimpan pada penyimpanan *on-line*.
cipherteks	= { *file* }
Nama	**kunci**
Penggunaan	- Level 1 Proses 5 Pengelolaan File (*input*) - Level 2 Proses 5.3 Upload (*input*) - Level 2 Proses 5.4 Download (*input*)
Deskripsi	String sebagai salah satu komponen dalam proses enkripsi dan dekripsi.
kunci	= { *any characters* }

Nama	**info_logout**
Penggunaan	- Level 1 Proses 6 Logout (*output*)
Deskripsi	Informasi keberhasilan *logout*.
info_ubah	= [logout_ok]
Nama	**data_dist**
Penggunaan	- Level 1 Proses 5 Pengelolaan File (*input*) - Level 2 Proses 5.3 Upload (*input*) - Level 2 Proses 5.4 Download (*input*) - Level 1 Proses 7 Distribusi (*input*) - Level 2 Proses 7.1 Tambah Distribusi (*input*) - Level 2 Proses 7.2 Tampil Distribusi (*input*) - Level 2 Proses 7.3 Detail Distribusi (*input*) - Level 2 Proses 7.4 Ubah Distribusi (*input*) - Level 2 Proses 7.5 Hapus Distribusi (*input*) - Level 2 Proses 7.6 Ubah Aktivasi (*input*) - Level 1 Proses 7 Distribusi (*output*) - Level 2 Proses 7.1 Tambah Distribusi (*output*) - Level 2 Proses 7.2 Tampil Distribusi (*output*) - Level 2 Proses 7.3 Detail Distribusi (*output*) - Level 2 Proses 7.4 Ubah Distribusi (*output*) - Level 2 Proses 7.5 Hapus Distribusi (*output*) - Level 2 Proses 7.6 Ubah Aktivasi (*output*)
Deskripsi	Berisi data sesuai dengan tabel distribusi.
data_file	= id + host + username + password + folder + tipe + mpf + max + status_distribusi + admin
id	= [0..9]
host	= [*Null* | A...Z | a...z]
username	= [*Null* | A...Z | a...z]
password	= [*Null* | {*any characters*}]
folder	= [A...Z | a...z]
tipe	= [*Null* | ’HTTP’ | ’FTP’]
mpf	= [0..9]
max	= [0..9]
status_distribusi	= [0..1]
admin	= [A...Z | a...z]

3.8.1.4 Spesifikasi Proses

Tabel 3.3 Proses 1.1 Cek Password

No. Proses	:	1.1
Nama	:	Cek Password
Input	:	data_login, input_login, data_user
Output	:	error_login, email, level

Deskripsi	:	`if (data_login[status] != ”OK”) {`
		`  username = input_login[email]`
		`  if (data_admin[username] != NULL) {`
		`  if (input_login[password] ==`
		`  data_admin[password]) {`
		`  email = username;`
		`  level = 1;`
		`  error_login = 0;`
		`  } else {`
		`  error_login = 3;`
		`  }`
		`  } elseif (data_user[input_login[email]] != NULL) {`
		`  if (data_user[status_akun] == 2) {`
		`  if (input_login[password] ==`
		`  data_user[password]) {`
		`  email = username;`
		`  level = 0;`
		`  error_login = 0;`
		`  } else {`
		`  error_login = 3;`
		`  }`
		`  } else {`
		`  error_login = 1;`
		`  }`
		`  } else {`
		`  error_login = 2;`
		`  }`
		`}`

Tabel 3.4 Proses 1.2 Ubah Status Login

No. Proses	:	1.2
Nama	:	Ubah Status Login
Input	:	error_login, email, level
Output	:	data_login, info_login

Deskripsi	:	`if (error_login == 0) {`
		`  data_login[status] = ”OK”;`
		`  data_login[email] = email;`
		`  data_login[level] = level;`
		`} else {`
		`  data_login[status] = ”NOK”;`
		`  data_login[email] = ””;`
		`}`
		`info_login = error_login;`

Tabel 3.5 Proses 2.1 Pendaftaran

No. Proses	:	2.1
Nama	:	Pandaftaran
Input	:	data_login, input_daftar, data_user

Output	:	data_user, email, info_pendaftaran

Deskripsi	:	`if (data_login[status] != ”OK”) {`
		`  email = input_daftar[email];`
		`  if (data_user[email] == NULL) {`
		`  SQL(INSERT INTO user VALUES input_daftar);`
		`  info_pendaftaran = 0;`
		`  } else {`
		`  if (data_user[status_akun] == 1) {`
		`  info_pendaftaran = 2;`
		`  // email terdaftar, aktif`
		`  } else {`
		`  info_pendaftaran = 1;`
		`  // email terdaftar, belum aktif`
		`  }`
		`  }`
		`}`

Tabel 3.6 Proses 2.2 Aktivasi

No. Proses	:	2.2
Nama	:	Aktivasi
Input	:	data_login, email, kode_aktivasi, data_user
Output	:	data_user, email, info_aktivasi

Deskripsi	:	`if (data_login[status] != ”OK”) {`
		`  if (data_user[email] == NULL) {`
		`  info_aktivasi = 1; // email belum terdaftar`
		`  } else {`
		`  if (data_user[status_akun] == 0) {`
		`  if (kode_aktivasi==data_user[kode_aktivasi]) {`
		`  SQL(UPDATE user SET kode_aktivasi=’’,`
		`  status_akun=2);`
		`  info_aktivasi = 0; // sukses`
		`  } else {`
		`  info_aktivasi = 3; // kode salah`
		`  }`
		`  } else {`
		`  info_aktivasi = 2; // akun sudah aktif`
		`  }`
		`  }`
		`}`

Tabel 3.7 Proses 2.3 Reset Password

No. Proses	:	2.3
Nama	:	Reset Password
Input	:	data_login, email, jawaban, password, data_user,
Output	:	data_user, info_reset, email

Deskripsi	:	`if (data_login[status] != ”OK”) {`
		`  if (data_user[email] == NULL) {`
		`  info_reset = 1; // email belum terdaftar`
		`  } else {`
		`  if (data_user[status_akun] == 2) {`
		`  if (jawaban ==`
		`  data_user[jawaban]) {`
		`  SQL(UPDATE user SET password);`
		`  info_reset = 0; // sukses`
		`  } else {`
		`  info_reset = 3;`
		`  // jawaban salah`
		`  }`

	`} else {`
	`info_reset = 2;`
	`// akun belum aktif`
	`}`
	`}`
	`}`

Tabel 3.8 Proses 3 Kirim Via Email

No. Proses	:	3
Nama	:	Kirim Via Email
Input	:	email, data_user
Output	:	data_user (ke email user)
Deskripsi	:	`mail(to:email,message:data_user);`

Tabel 3.9 Proses 4.1 Cek Password

No. Proses	:	4.1
Nama	:	Cek Password
Input	:	data_login, password, password_baru
Output	:	error_ubah, password

Deskripsi	:	`if (data_login[status] == ”OK”) {`
		`email = data_login[email];`
		`if (data_user[password] == password) {`
		`password = password_baru;`
		`error_ubah = 0;`
		`} else {`
		`error_ubah = 1; // password salah`
		`}`
		`}`

Tabel 3.10 Proses 4.2 Simpan Password

No. Proses	:	4.2
Nama	:	Simpan Password
Input	:	error_ubah, password
Output	:	data_user, info_ubah

Deskripsi	:	`if (error_ubah == 0) {`
		`SQL(UPDATE user SET password)`
		`info_ubah = 0; // sukses`
		`} else {`
		`info_ubah = 1; // salah password`
		`}`

Tabel 3.11 Proses 5.1 Tampil Kotak File

No. Proses	:	5.1
Nama	:	Tampil Kotak File
Input	:	data_login, id_file, kunci, data_file
Output	:	data_file, id_file

Deskripsi	:	`if (data_login[status] == ”OK”) {`
		`SQL(SELECT * FROM file`
		`WHERE email=data_login[email]`
		`AND status_file=1;)`
		`}`

Tabel 3.12 Proses 5.2 Tampil Kotak Sampah

No. Proses	:	5.2
Nama	:	Tampil Kotak Sampah
Input	:	data_login, id_file, data_file
Output	:	data_file, id_file

Deskripsi	:	`if (data_login[status] == ”OK”) {`
		`  SQL(SELECT * FROM file`
		`  WHERE email=data_login[email]`
		`  AND status_file=2);`
		`}`

Tabel 3.13 Proses 5.3 Upload

No. Proses	:	5.3
Nama	:	Upload
Input	:	data_login, plainteks, kunci
Output	:	cipherteks, data_file

Deskripsi	:	`if (data_login[status] == ”OK”) {`
		`  handle = fopen(plainteks,”rb”);`
		`  handle2 = fopen(data_dist + cipherteks,”wb”);`
		`  blok = enkrip(fread(handle),kunci);`
		`  fwrite(handle2,blok);`
		`  SQL(INSERT INTO file VALUES data_file);`
		`}`

Tabel 3.14 Proses 5.4 Download

No. Proses	:	5.4
Nama	:	Download
Input	:	id_file, kunci, data_file, cipherteks
Output	:	plainteks

Deskripsi	:	`SQL(SELECT data_file FROM file WHERE id_file);`
		`handle = fopen(data_dist + cipherteks,”rb”);`
		`handle2 = fopen(plainteks,”wb”);`
		`blok = dekrip(fread(handle),kunci);`
		`fwrite(handle2,blok);`

Tabel 3.15 Proses 5.5 Hapus

No. Proses	:	5.5
Nama	:	Hapus
Input	:	id_file
Output	:	data_file
Deskripsi	:	`SQL(UPDATE file SET data_file[status_file]=2` `  WHERE id_file);`

Tabel 3.16 Proses 5.6 Kembalikan

No. Proses	:	5.6
Nama	:	Kembalikan
Input	:	id_file
Output	:	data_file

Deskripsi	:	SQL(UPDATE file SET data_file[status_file]=1 WHERE id_file);

Tabel 3.17 Proses 5.7 Buang

No. Proses	:	5.7
Nama	:	Buang
Input	:	id_file
Output	:	data_file, id_file
Deskripsi	:	SQL(DELETE FROM file WHERE id_file); DELETE (data_dist + cipherteks[id_file]);

Tabel 3.18 Proses 6 Logout

No. Proses	:	6
Nama	:	Logout
Input	:	data_login
Output	:	data_login, info_logout

Deskripsi	:	if (data_login[status] == ”OK”) {
		data_login[status] = ”NOK”;
		data_login[email] = ””;
		info_logout = 0;
		}

Tabel 3.19 Proses 7.1 Tambah Distribusi

No. Proses	:	7.1
Nama	:	Tambah Distribusi
Input	:	data_login, data_dist
Output	:	data_dist

Deskripsi	:	if ((data_login[status] == ”OK”) &&
		(data_login[level] == 1)) {
		SQL(INSERT INTO distribusi VALUE (data_dist));
		}

Tabel 3.20 Proses 7.2 Tampil Distribusi

No. Proses	:	7.2
Nama	:	Tampil Distribusi
Input	:	data_login, data_dist
Output	:	data_dist

Deskripsi	:	if ((data_login[status] == ”OK”) &&
		(data_login[level] == 1)) {
		SQL(SELECT * FROM distribusi
		WHERE admin=data_login[email])
		}

Tabel 3.21 Proses 7.3 Detail Distribusi

No. Proses	:	7.3
Nama	:	Detail Distribusi
Input	:	data_login, data_dist
Output	:	data_dist
Deskripsi	:	if ((data_login[status] == ”OK”) && (data_login[level] == 1)) {

	SQL(SELECT * FROM distribusi WHERE id=data_dist[id]) }

Tabel 3.22 Proses 7.4 Ubah Distribusi

No. Proses	:	7.4
Nama	:	Ubah Distribusi
Input	:	data_login, data_dist
Output	:	data_dist

Deskripsi	:	if ((data_login[status] == ”OK”) &&
		(data_login[level] == 1)) {
		SQL(UPDATE distribusi SET data_dist
		WHERE id=data_dist[id])
		}

Tabel 3.23 Proses 7.5 Hapus Distribusi

No. Proses	:	7.5
Nama	:	Hapus Distribusi
Input	:	data_login, data_dist
Output	:	data_dist

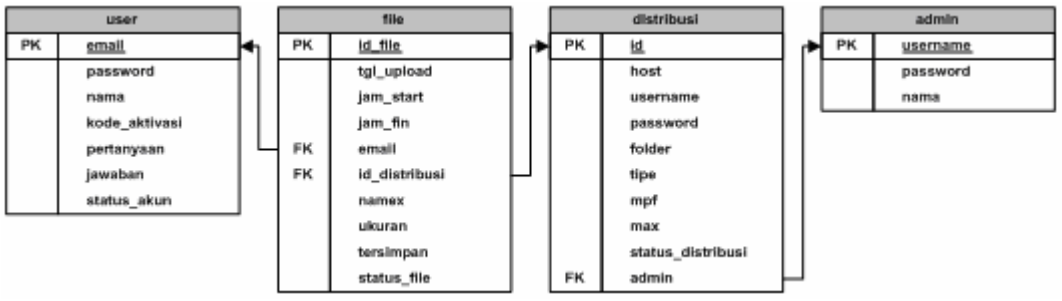
Deskripsi	:	if ((data_login[status] == ”OK”) &&
		(data_login[level] == 1)) {
		SQL(DELETE FROM distribusi
		WHERE id=data_dist[id])
		}

Tabel 3.24 Proses 7.6 Ubah Aktivasi

No. Proses	:	7.6
Nama	:	Ubah Aktivasi
Input	:	data_login, data_dist
Output	:	data_dist

Deskripsi	:	if ((data_login[status] == ”OK”) &&
		(data_login[level] == 1)) {
		SQL(UPDATE distribusi
		SET status_distribusi NOT status_distribusi
		WHERE id=data_dist[id])
		}

3.8.1.5 Skema Relasi



Gambar 3.23 Skema Relasi

3.8.1.6 Struktur Data

1. Tabel user

Tabel 3.25 Tabel user

Tabel	: user		
Primary Key	: email		
Nama Field	Tipe Data	Ukuran	Keterangan
email	varchar	50	PK
password	varchar	100	MD5()
nama	varchar	25	
kode_aktivasi	varchar	50	
pertanyaan	varchar	30	
jawaban	varchar	100	MD5()
status_akun	boolean		

2. Tabel file (data *file* yang tersimpan)

Tabel 3.26 Tabel file

Tabel	: file		
Primary Key	: id_file		
Nama Field	Tipe Data	Ukuran	Keterangan
id_file	int	11	PK
tgl_upload	date		curdate()
jam_start	time		curtime()
jam_fin	time		curtime()
email	varchar	50	FK pada tabel user
id_distribusi	int	11	FK pada tabel distribusi
namex	varchar	100	
ukuran	int	11	
avalanche	int	11	
jumlah_bit	int	11	
status_file	int	1	

3. Tabel distribusi (data lokasi penyimpanan *file online*)

Tabel 3.27 Tabel distribusi

Tabel	: distribusi		
Primary Key	: id		
Nama Field	Tipe Data	Ukuran	Keterangan
id	int	11	PK
host	varchar	100	
username	varchar	20	
password	varchar	16	
folder	varchar	20	
tipe	enum		‘http’,’ftp’
mpf	int	20	
max	int	20	
status_distribusi	int	11	
admin	varchar	30	FK pada tabel admin

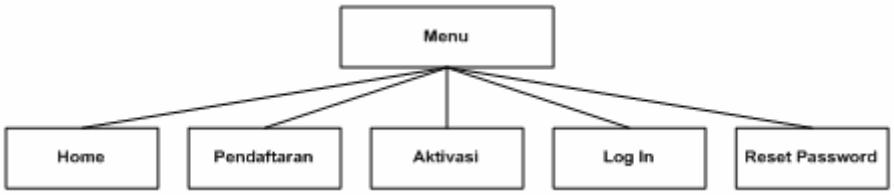
4. Tabel admin

Tabel 3.28 Tabel admin

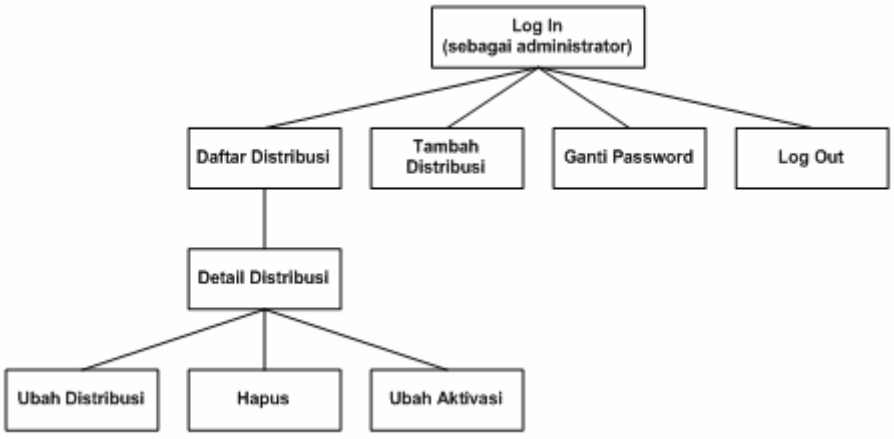
Tabel	: admin		
Primary Key	: username		
Nama Field	Tipe Data	Ukuran	Keterangan
username	varchar	30	PK
password	varchar	100	MD5()
nama	varchar	25	

3.8.2 Perancangan Menu

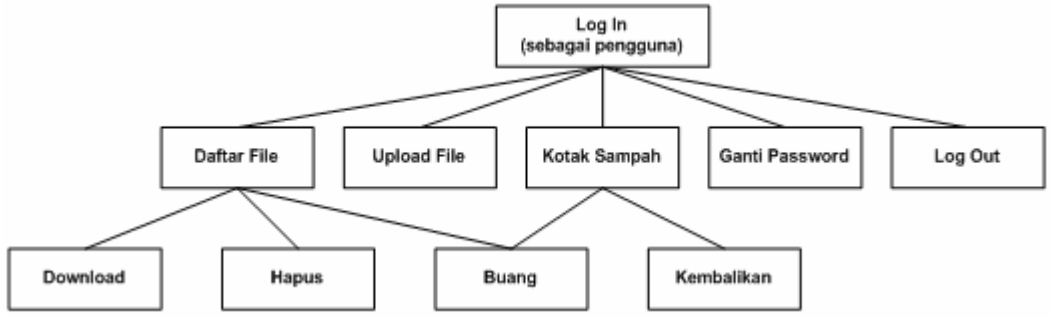
Untuk mempermudah dalam menggunakan aplikasi penyimpanan *on-line* ini, maka akan dirancang suatu menu dengan struktur menu *network*. Struktur ini sangat cocok digunakan pada aplikasi-aplikasi berbasis web. Ada tiga buah perancangan menu dimana masing-masing adalah ketika sebelum *login*, setelah *login* sebagai *user administrator*, dan setelah *login* sebagai *user* pengguna.



Gambar 3.24 Rancangan menu sebelum *login*



Gambar 3.25 Rancangan menu setelah *login* sebagai *user administrator*

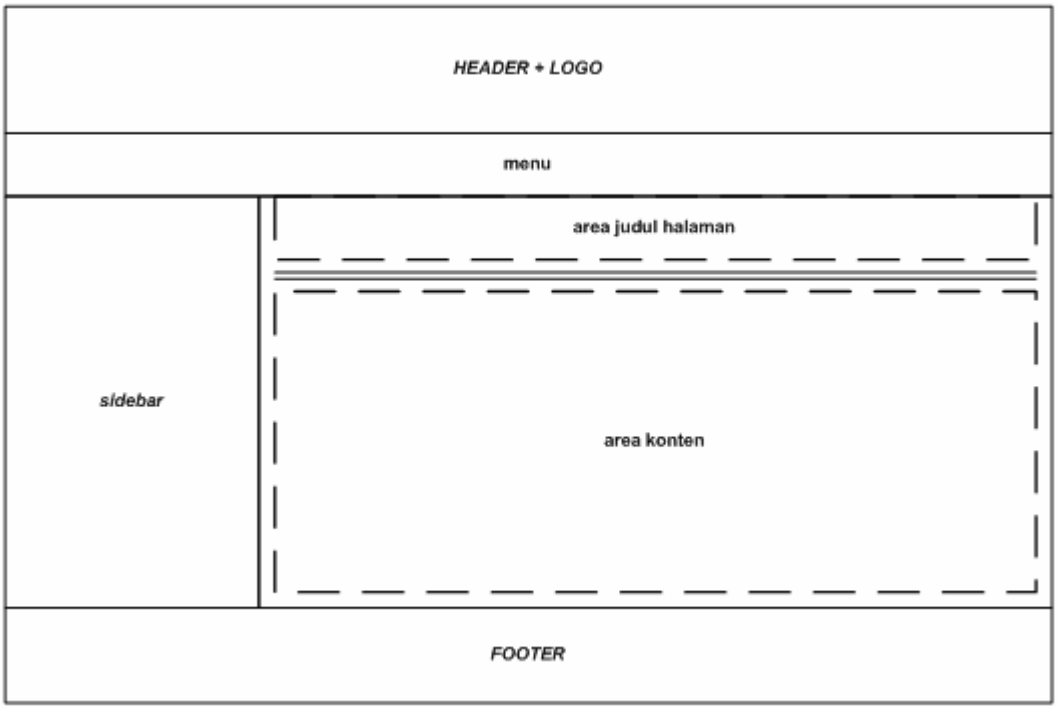


Gambar 3.26 Rancangan menu setelah *login* sebagai *user* pengguna

3.8.3 Perancangan Antarmuka (*Interface*)

Agar aplikasi penyimpanan *file on-line* ini dapat memenuhi kebutuhan pengguna, baik secara fungsional maupun non fungsional, perlu dirancang sebuah antarmuka. Antarmuka ini lebih diutamakan pada konsistensinya dan pada aspek *user friendly*.

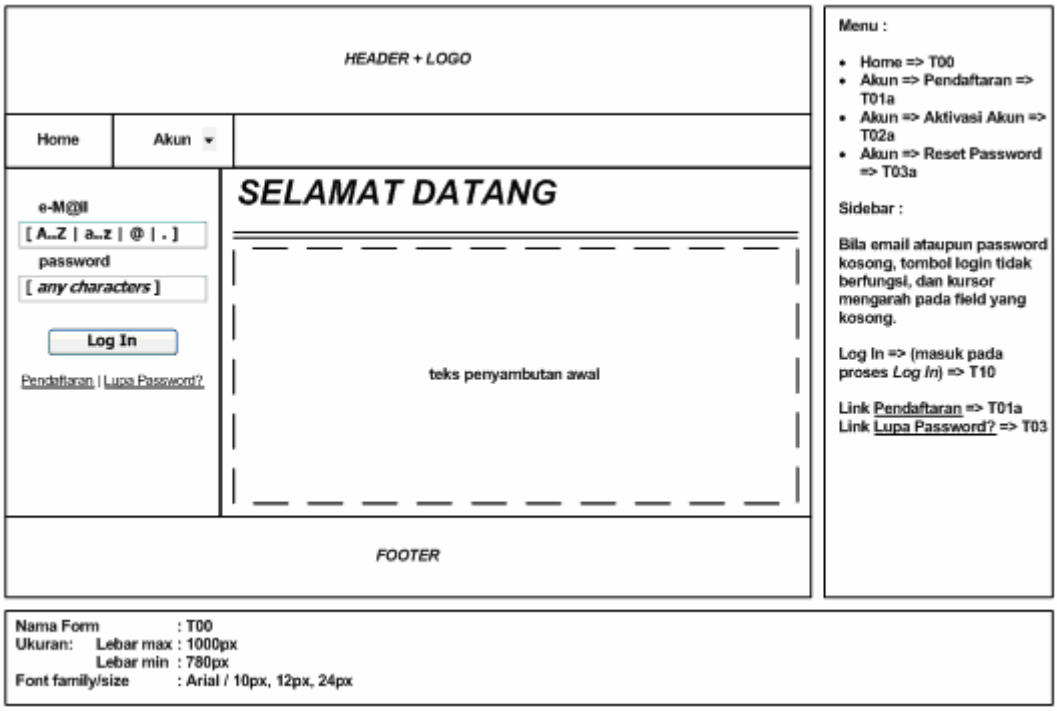
Adapun rancangan tata letak secara umum dapat dilihat pada gambar 3.27.



Gambar 3.27 Rancangan tata letak

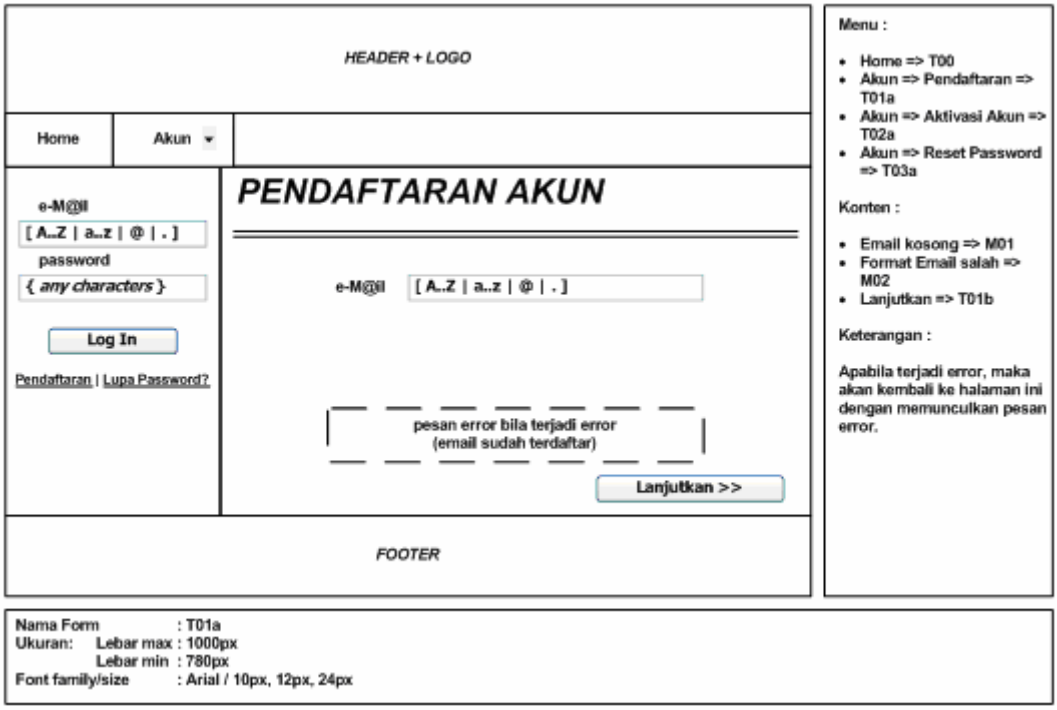
Dapat dilihat pada gambar 3.28 hingga gambar 3.53, rancangan antarmuka pada setiap halamannya dan dilengkapi dengan penjelasan untuk setiap fungsi dari form, tombol, mapun link yang terdapat pada halaman tersebut. Antara halaman yang satu dengan halaman yang lainnya memiliki hubungan sebab akibat. Gambar 3.28 hingga gambar 3.39 adalah halaman sebelum melakukan *Log In*. Gambar 3.40 hingga gambar 3.46 adalah halaman setelah melakukan *Log In* sebagai *user* pengguna dan gambar 3.47 hingga gambar 3.53 adalah halaman setelah melakukan *Log In* sebagai *user administrator*. Untuk tampilan pesan dapat dilihat pada gambar 3.54 hingga gambar 3.77.

1. *Home* (sebelum *login*)



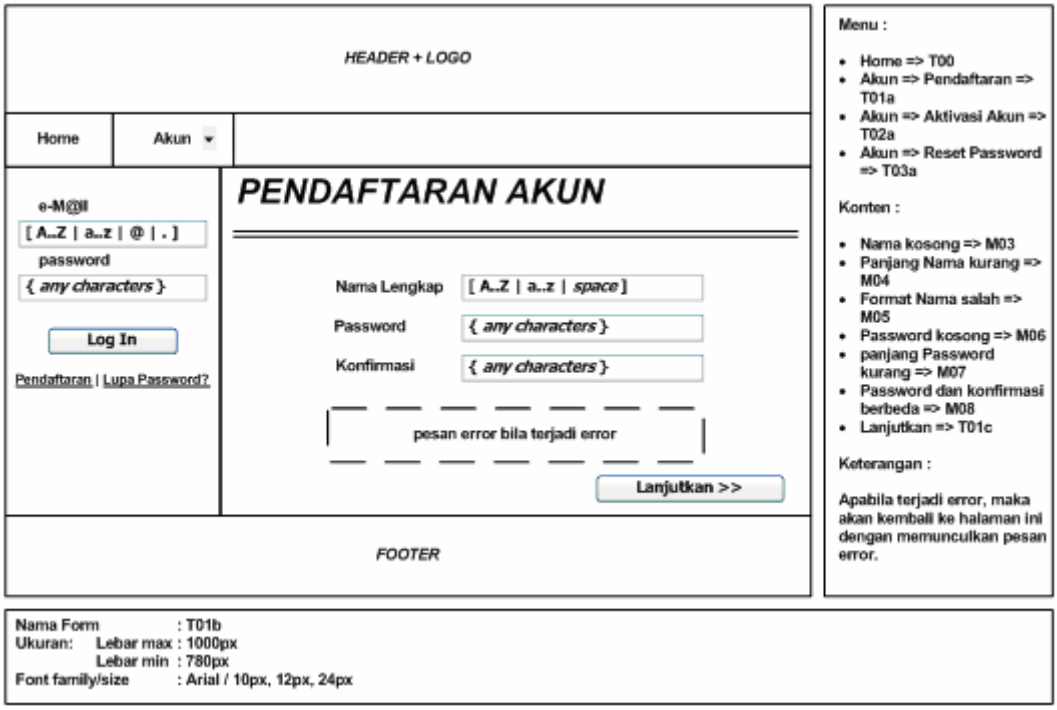
Gambar 3.28 Halaman *Home* sebelum *Log In* (T00)

2. Pendaftaran Akun (memasukkan alamat email)



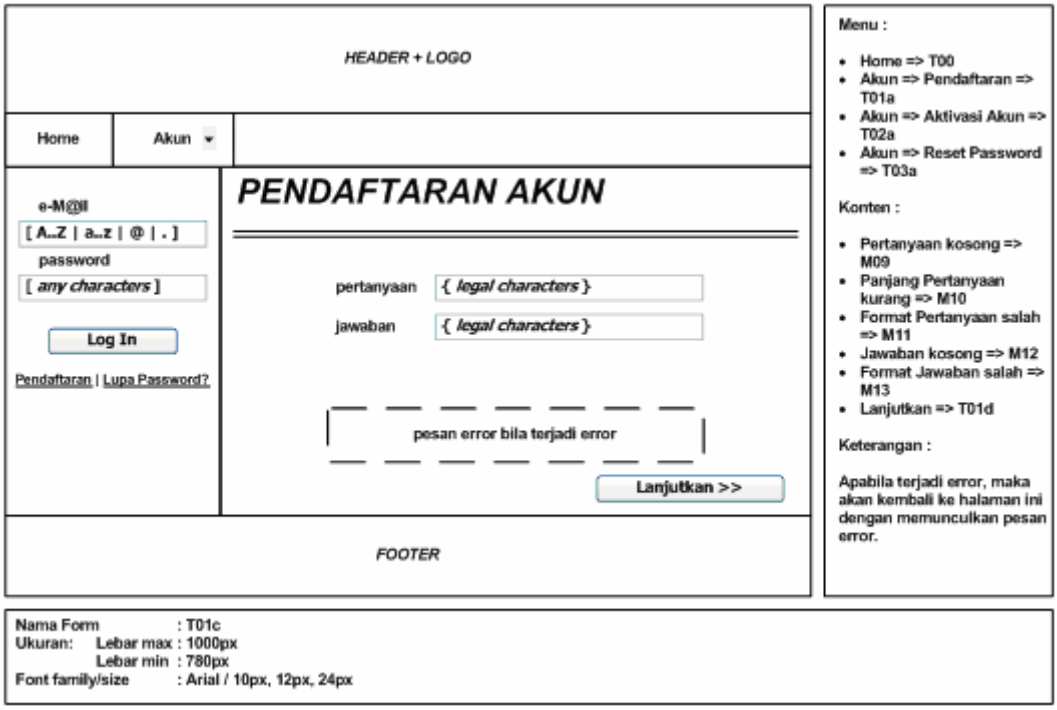
Gambar 3.29 Halaman Pendaftaran akun (T01a)

3. Pendaftaran Akun (memasukkan profil pengguna)



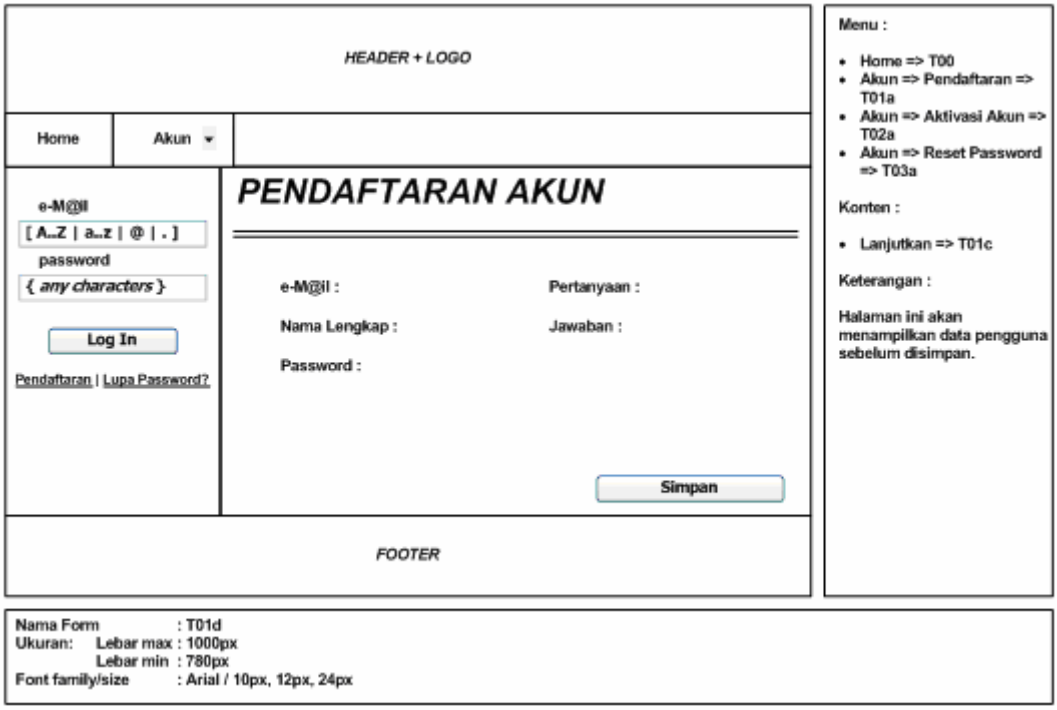
Gambar 3.30 Halaman Pendaftaran akun (T01b)

4. Pendaftaran Akun (memasukkan pertanyaan dan jawaban)



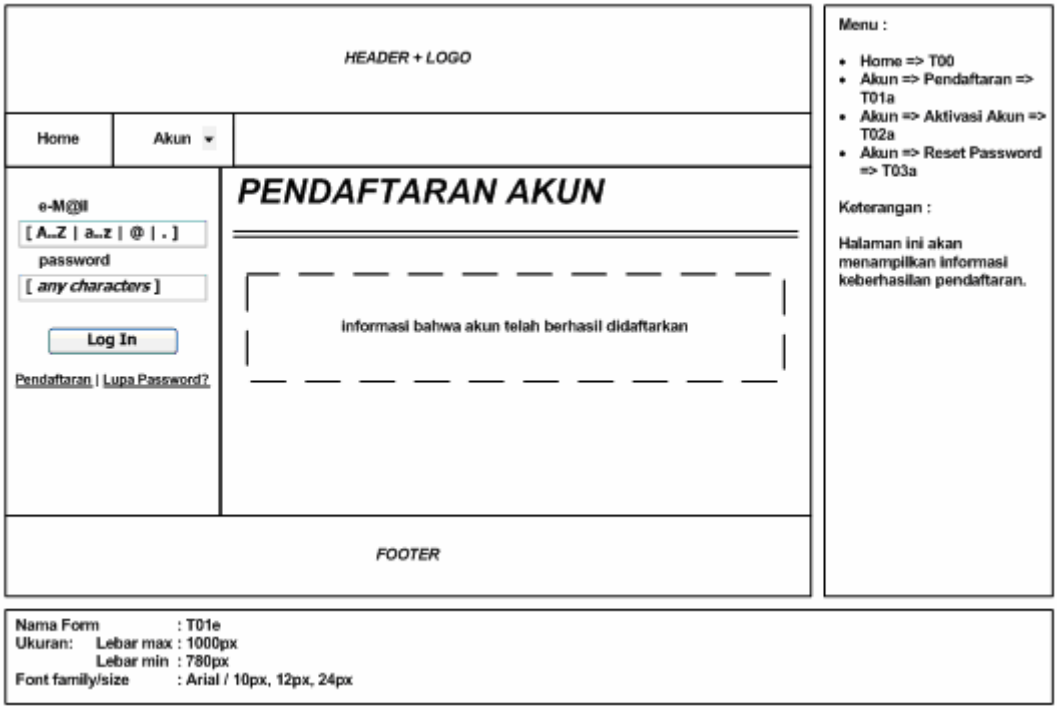
Gambar 3.31 Halaman Pendaftaran akun (T01c)

5. Pendaftaran Akun (menampilkan data pengguna yang akan disimpan)



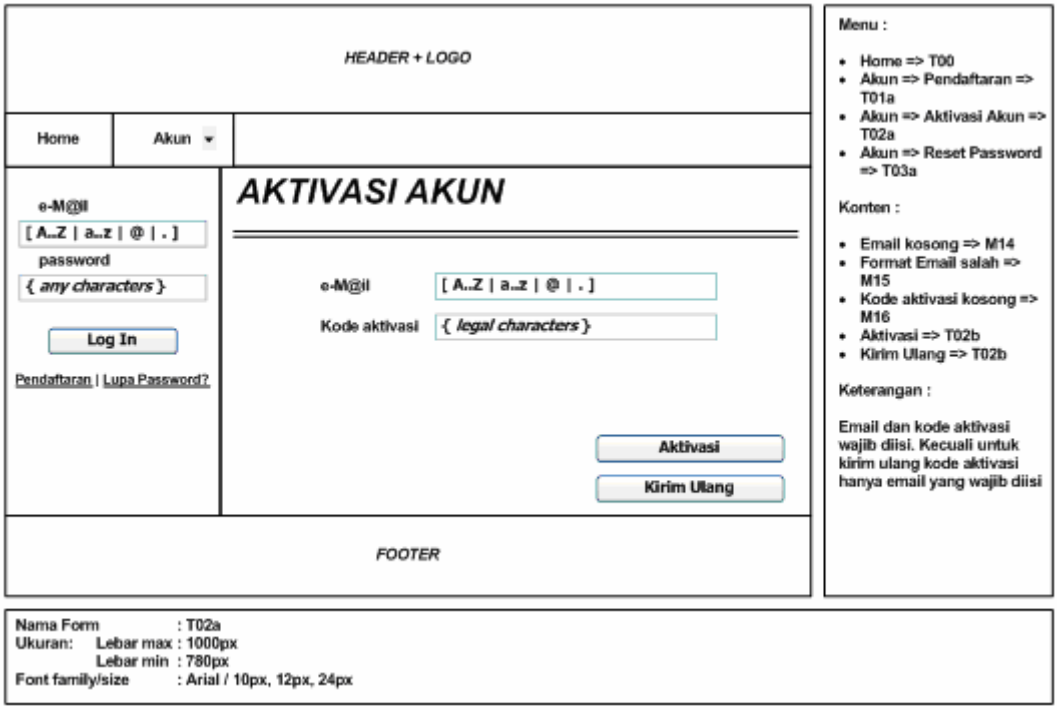
Gambar 3.32 Halaman Pendaftaran akun (T01d)

6. Pendaftaran Akun (informasi keberhasilan)



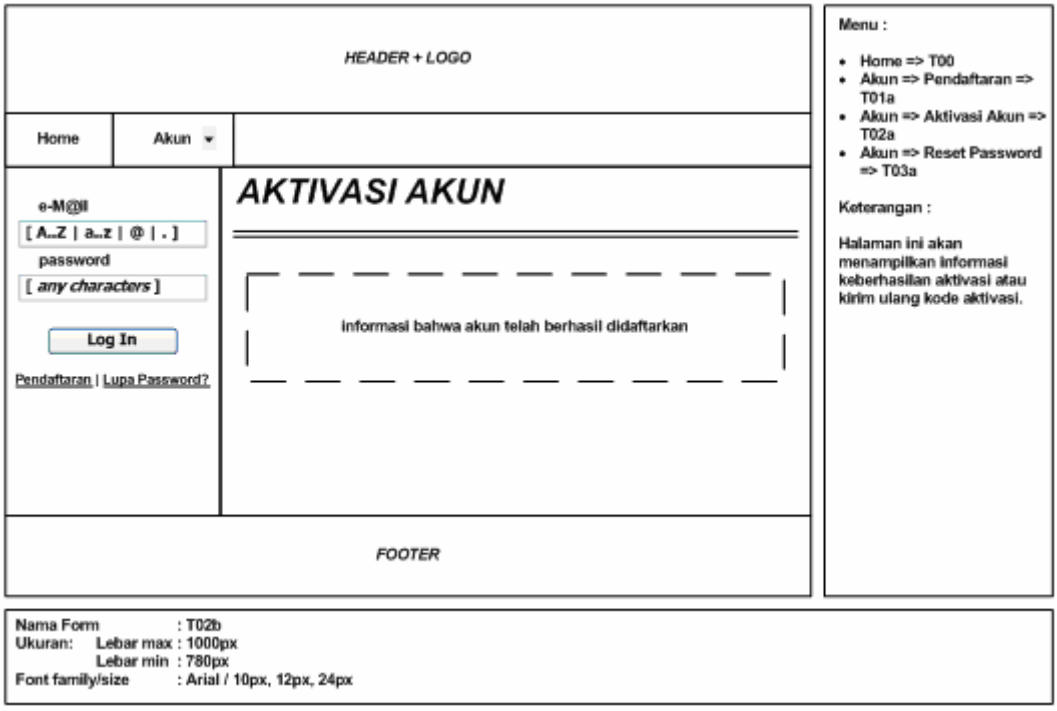
Gambar 3.33 Halaman Pendaftaran akun (T01e)

7. Aktivasi Akun (memasukkan alamat email dan kode aktivasi)



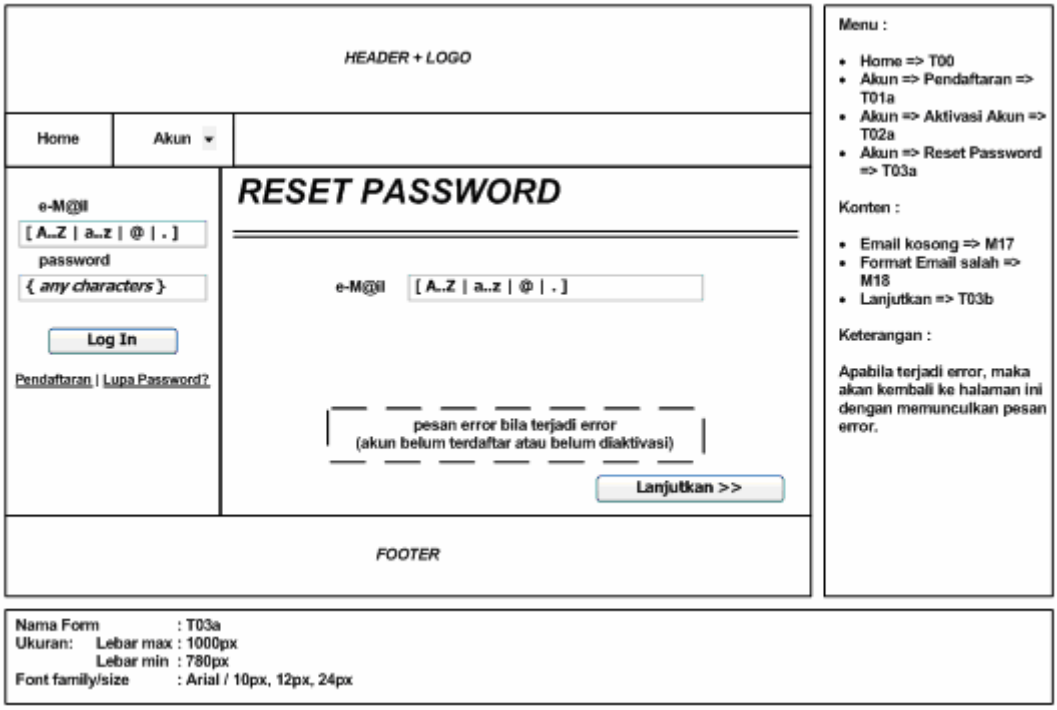
Gambar 3.34 Halaman aktivasi akun (T02a)

8. Aktivasi Akun (informasi keberhasilan aktivasi akun)



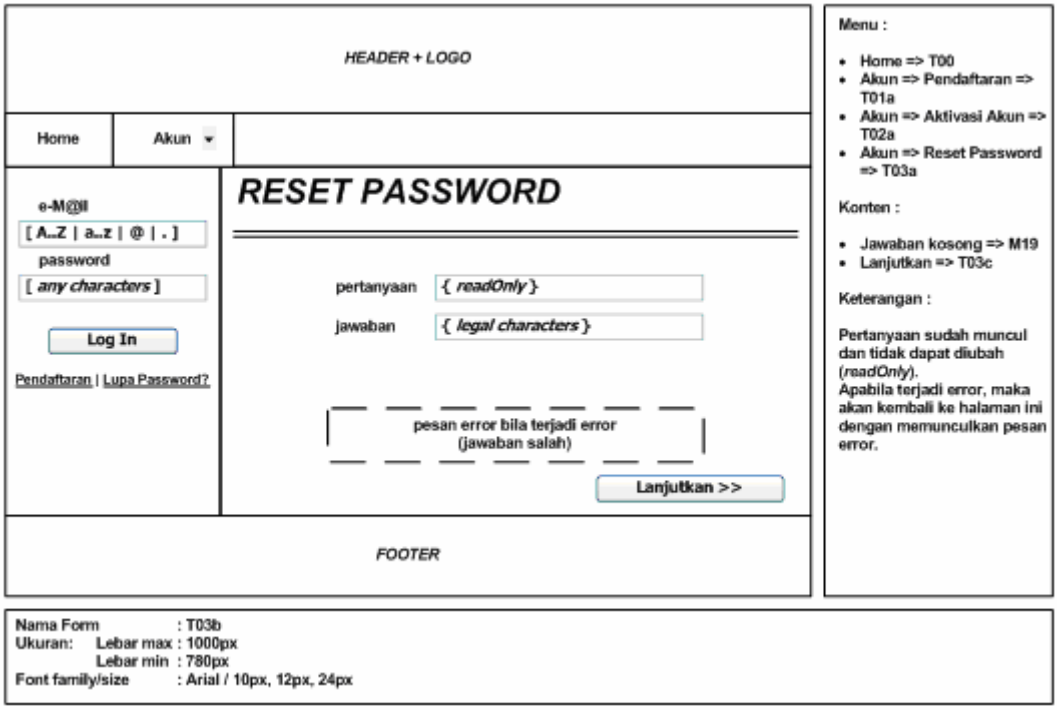
Gambar 3.35 Halaman aktivasi akun (T02b)

9. Reset Password (memasukkan alamat email)



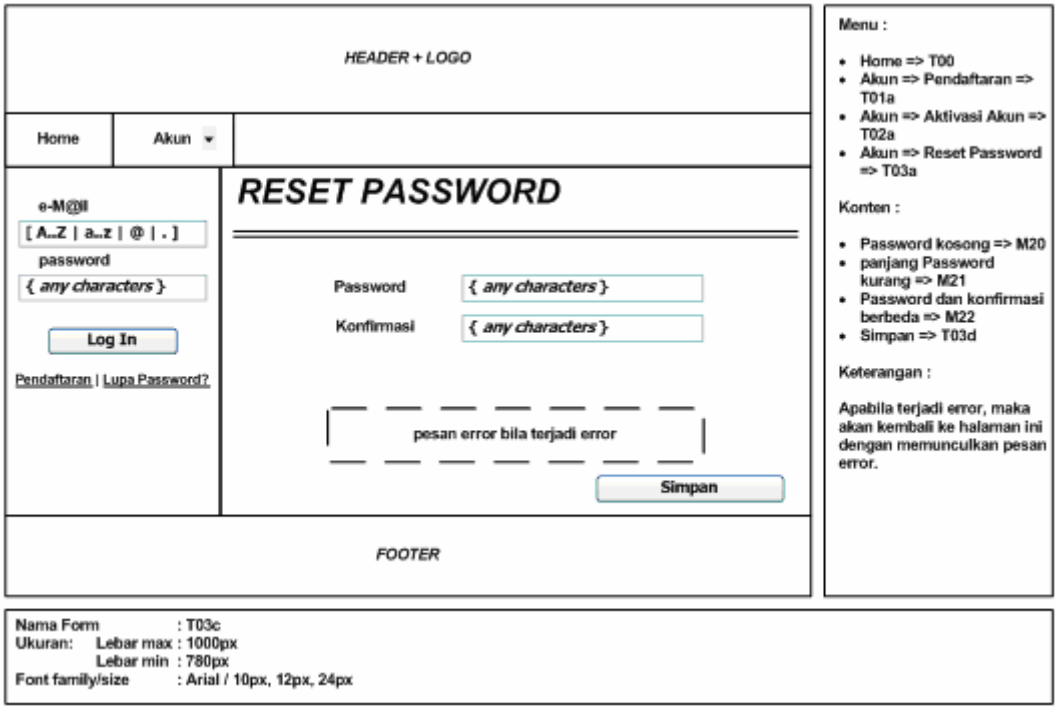
Gambar 3.36 Halaman reset password (T03a)

10. Reset Password (menjawab pertanyaan)



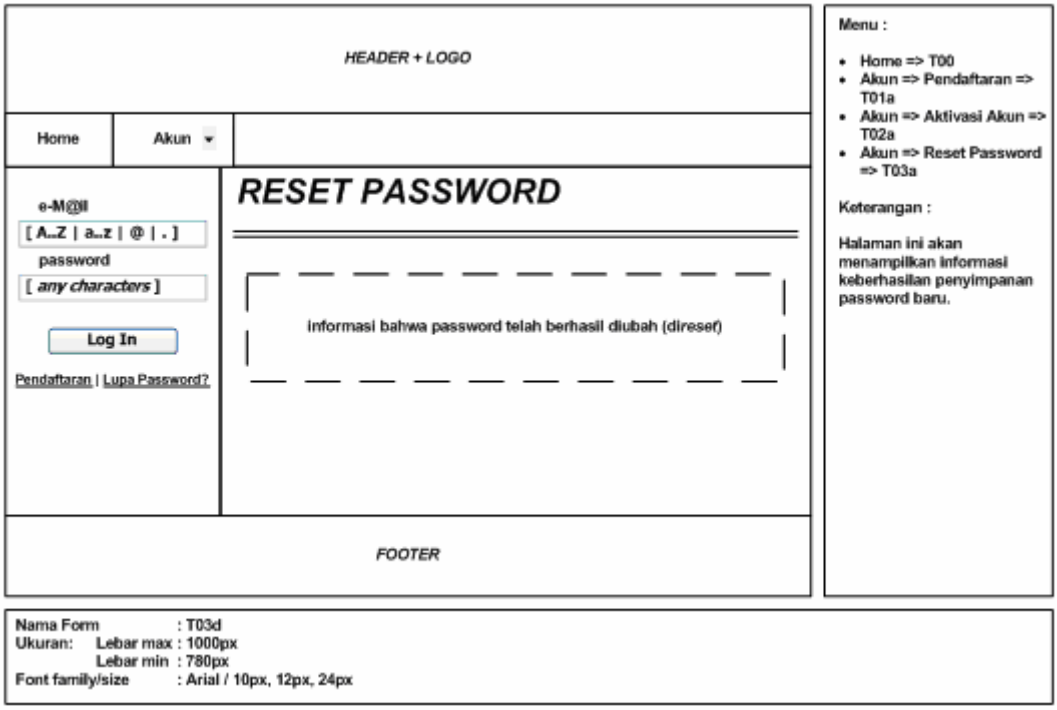
Gambar 3.37 Halaman reset password (T03b)

11. Reset Password (memasukkan password baru)



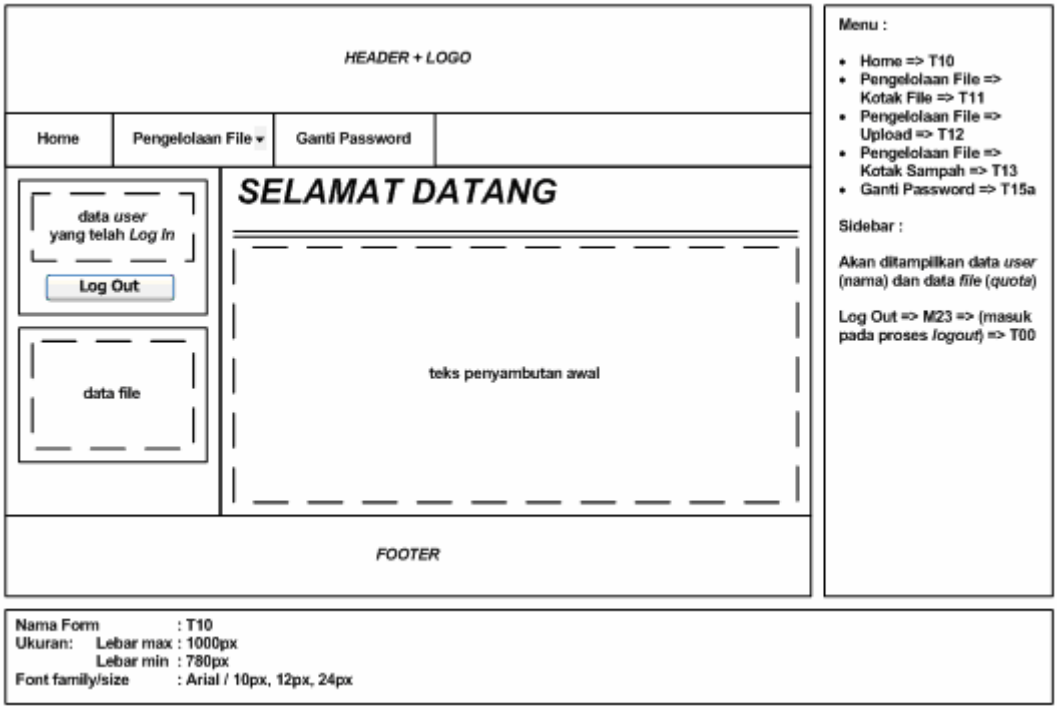
Gambar 3.38 Halaman reset password (T03c)

12. Reset Password (informasi keberhasilan menyimpan password)



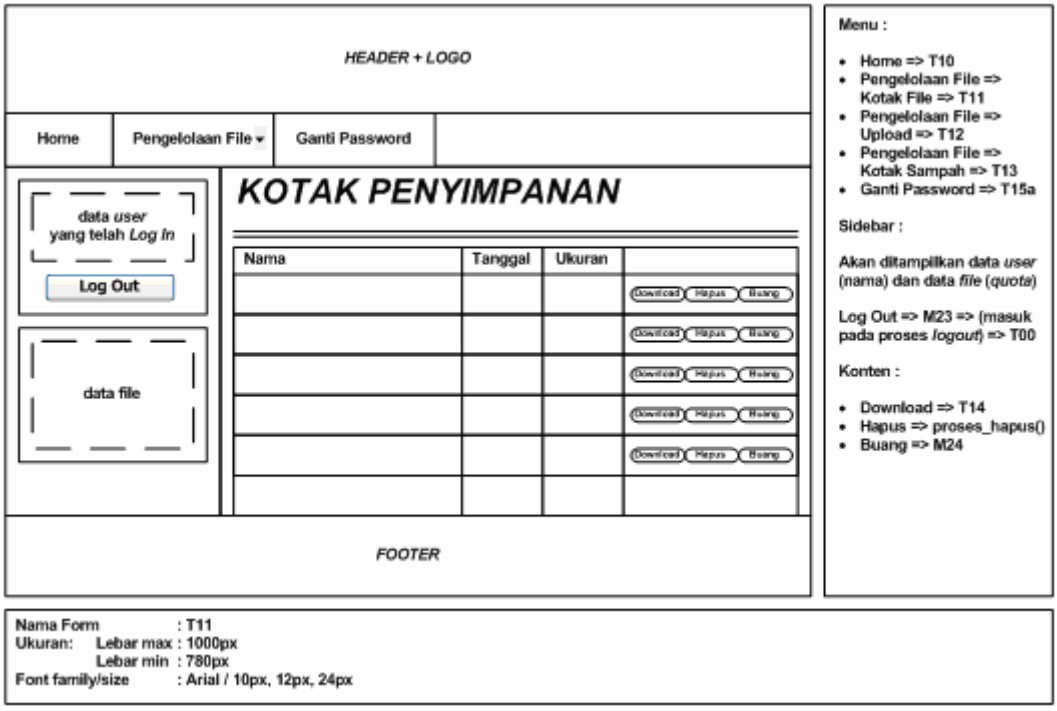
Gambar 3.39 Halaman reset password (T03d)

13. *Home* (setelah *Log In* sebagai *user* pengguna)



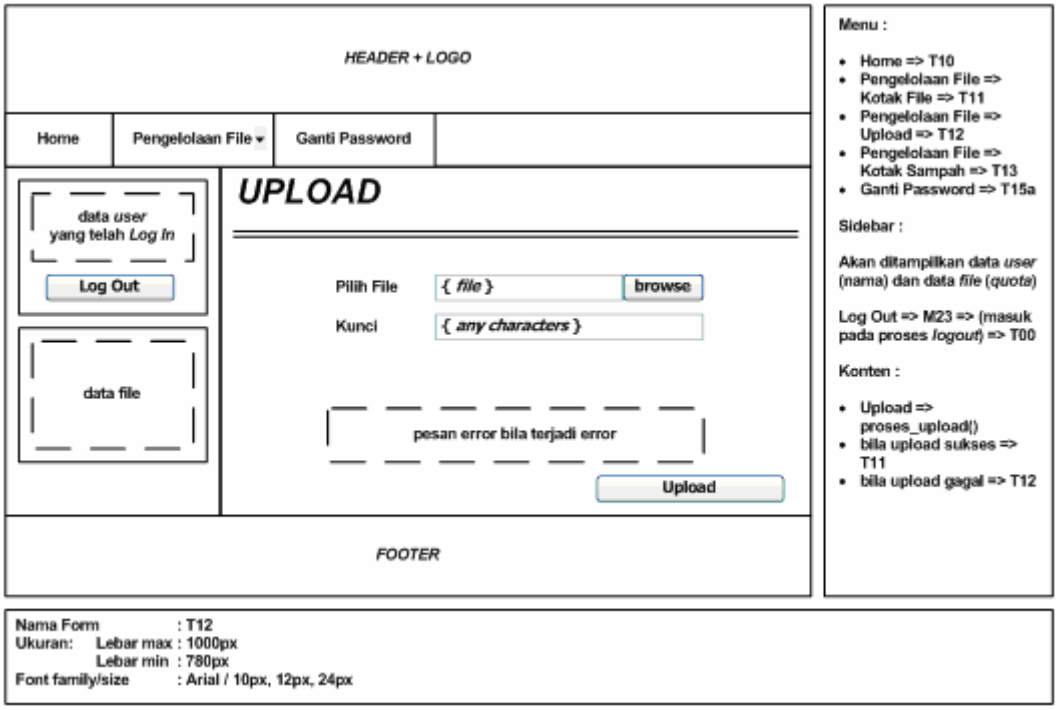
Gambar 3.40 Halaman *Home* setelah *Log In* (T10)

14. Tampil Daftar File pada Kotak Penyimpanan



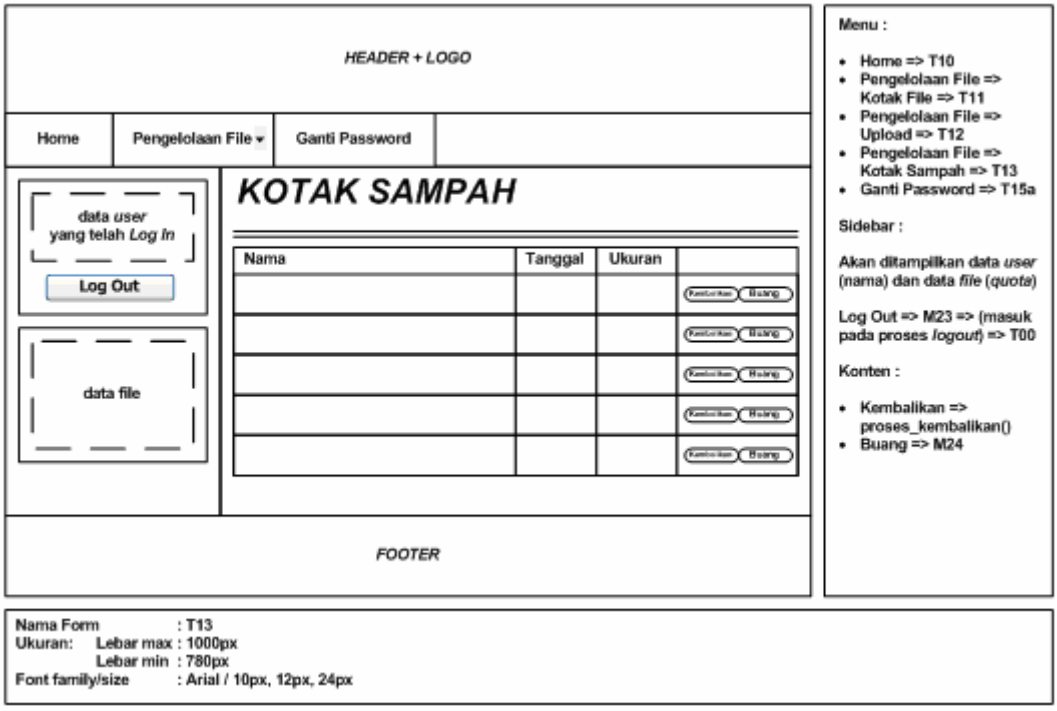
Gambar 3.41 Kotak Penyimpanan (T11)

15. *Upload*



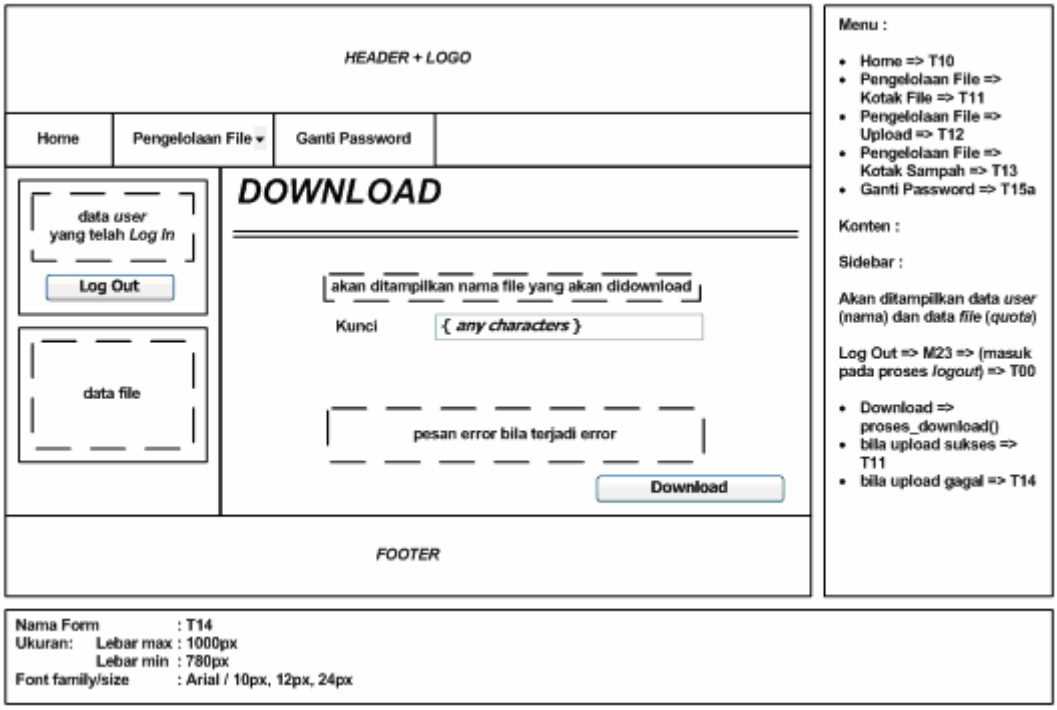
Gambar 3.42 Upload (T12)

16. Tampil Daftar File pada Kotak Sampah



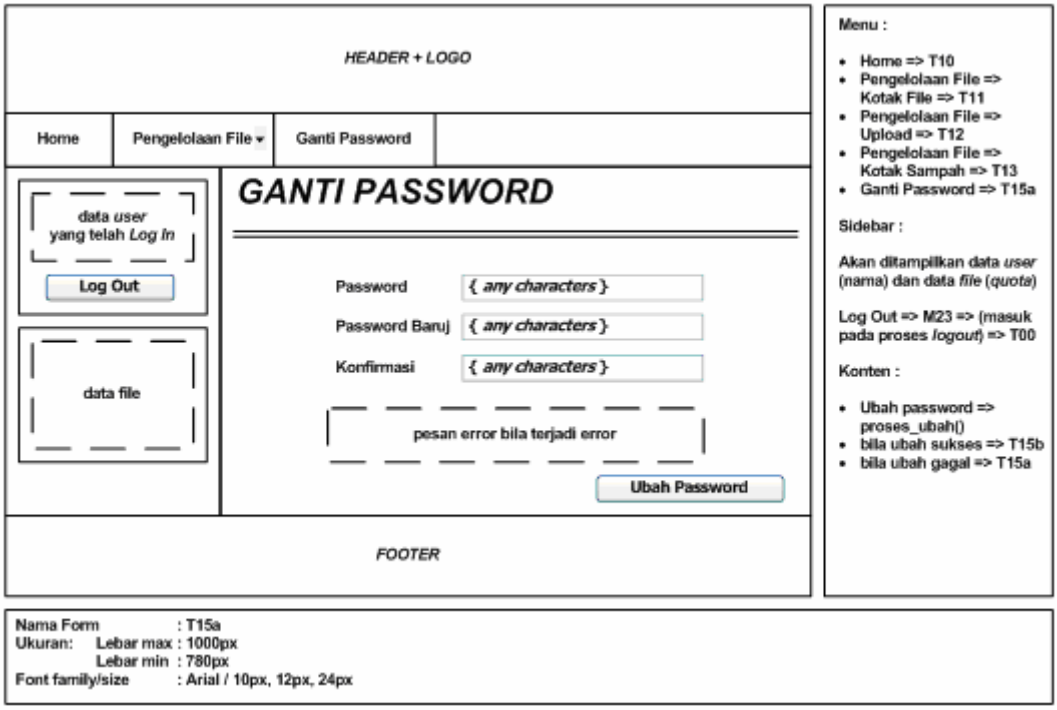
Gambar 3.43 Kotak Sampah (T13)

17. *Download*



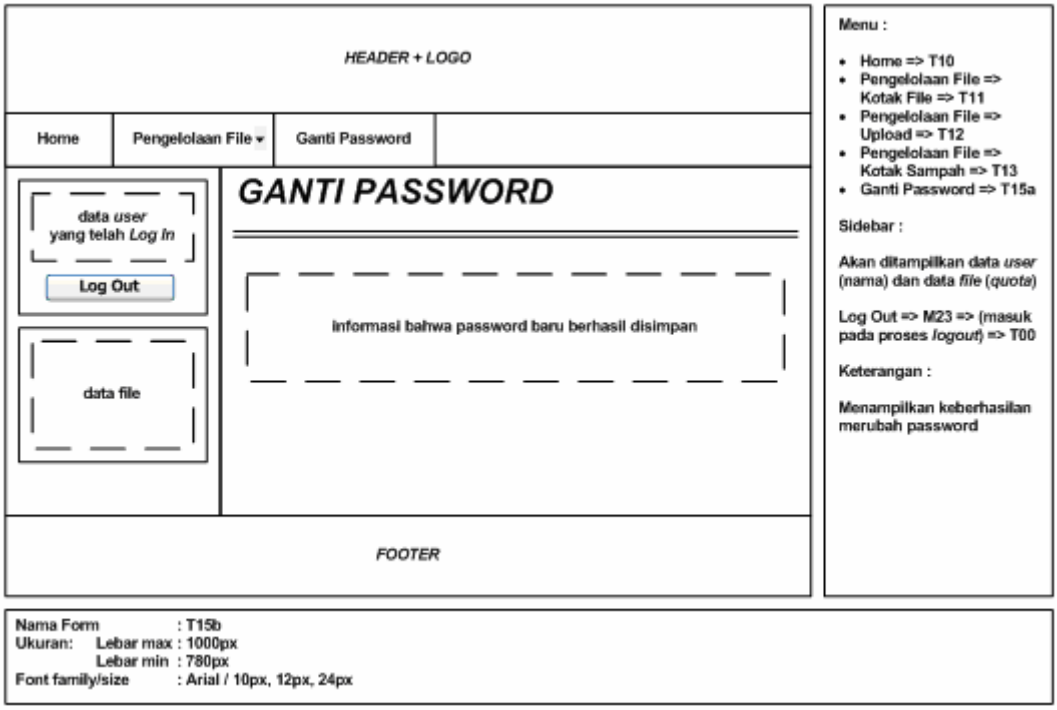
Gambar 3.44 Download (T14)

18. Ganti Pasword (memasukkan password baru)



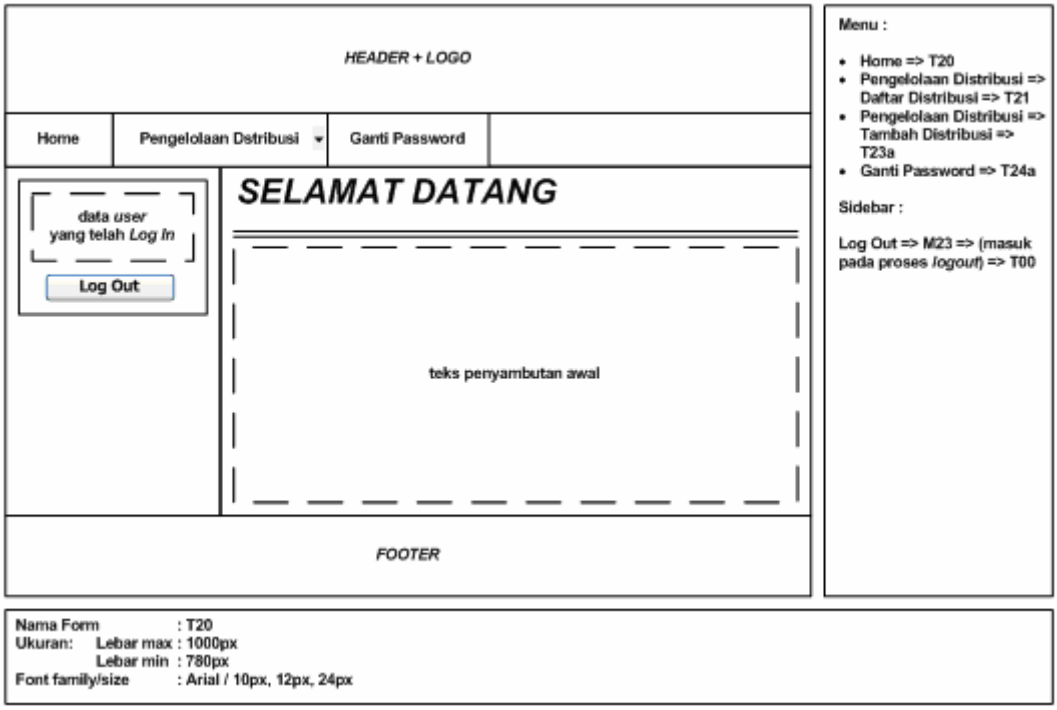
Gambar 3.45 Ganti Password (T15a)

19. Ganti Password (sukses)



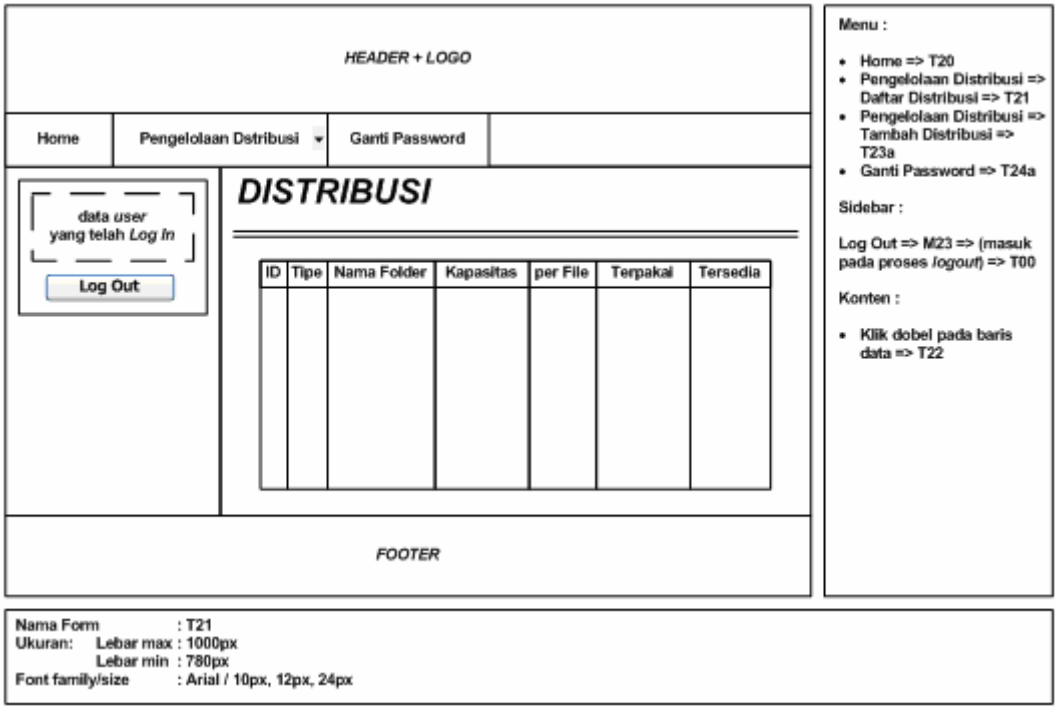
Gambar 3.46 Informasi keberhasilan ganti password (T15b)

20. *Home* (setelah *Log In* sebagai *user administrator*)



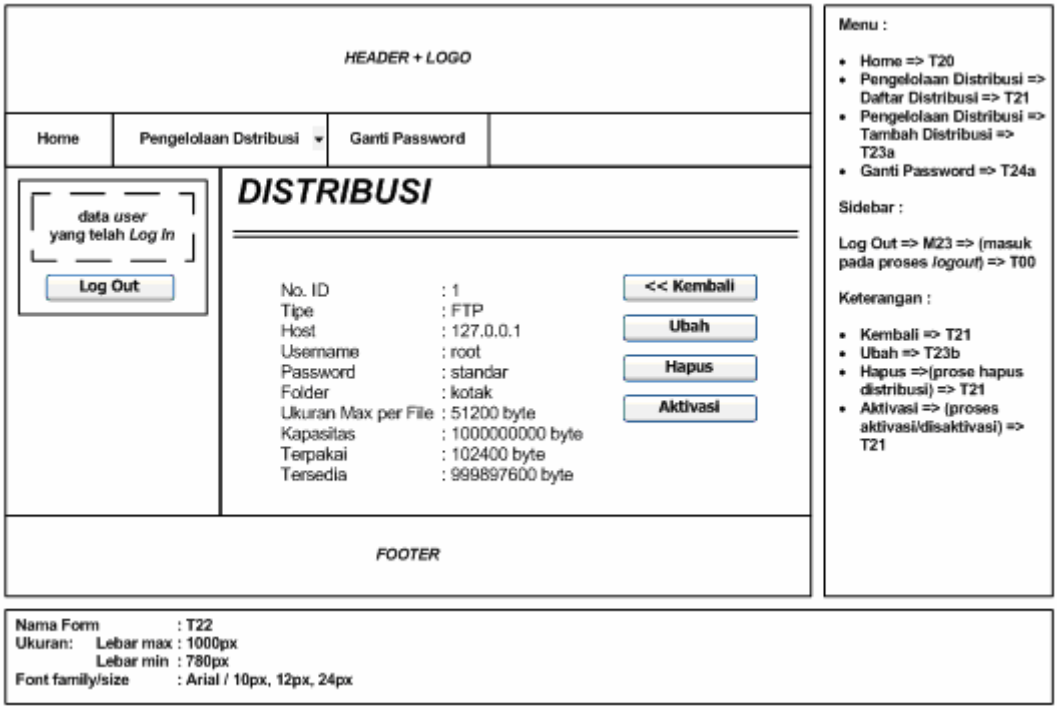
Gambar 3.47 Halaman *Home* setelah *Log In* (T20)

21. Daftar Distribusi



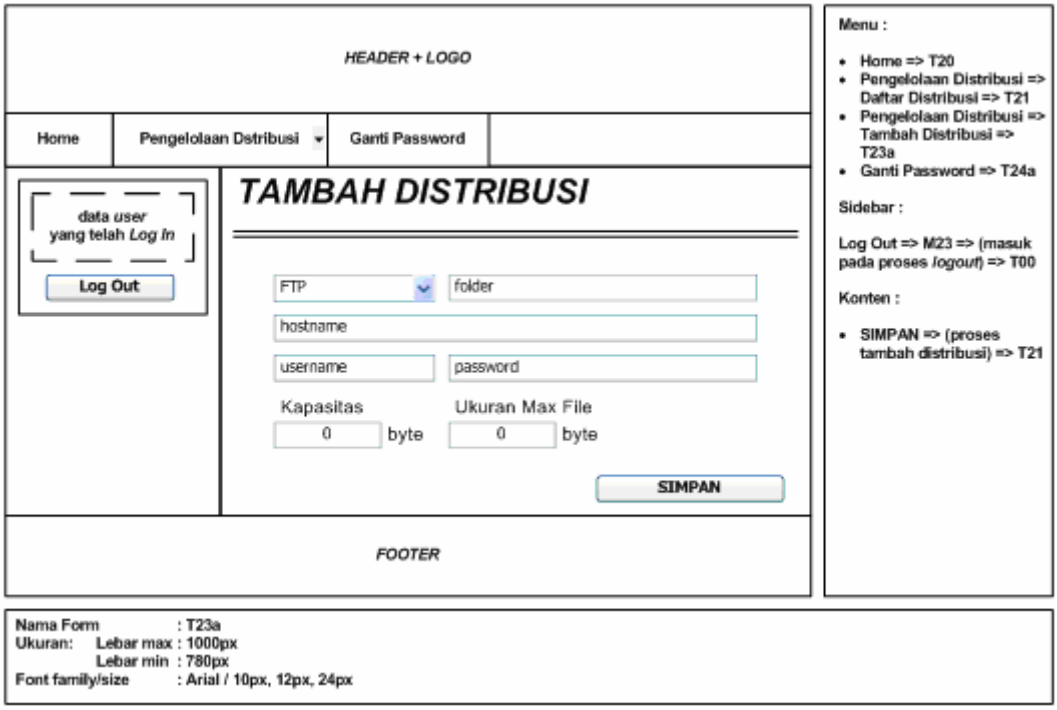
Gambar 3.48 Daftar Distribusi (T21)

22. Detail Distribusi



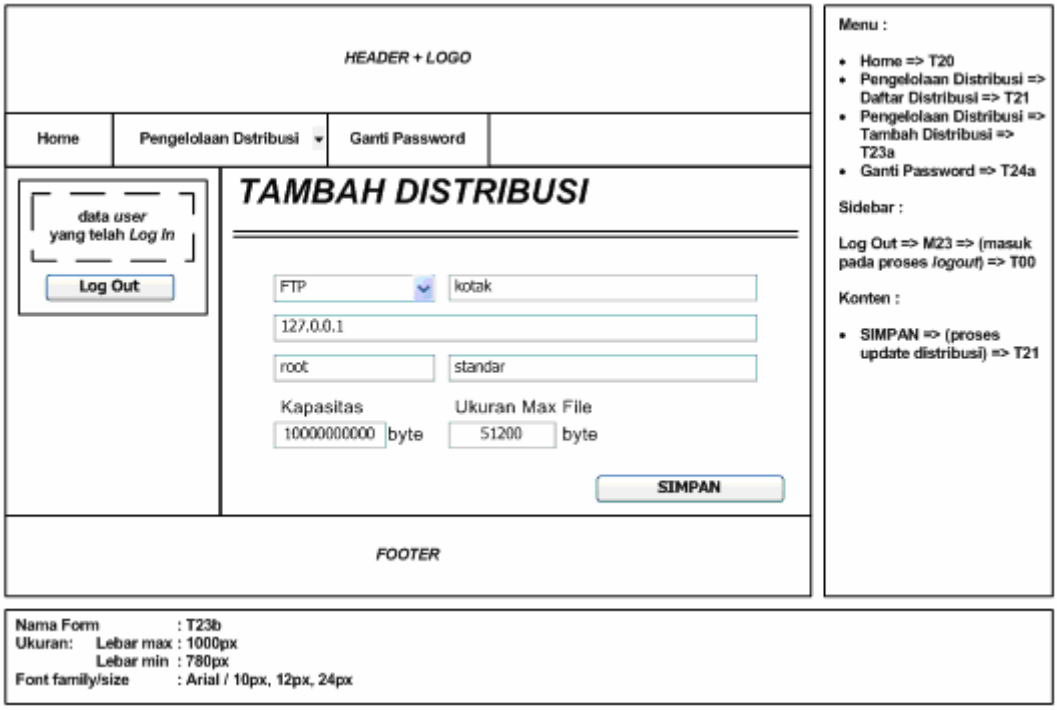
Gambar 3.49 Detail Distribusi (T22)

23. Tambah Distribusi



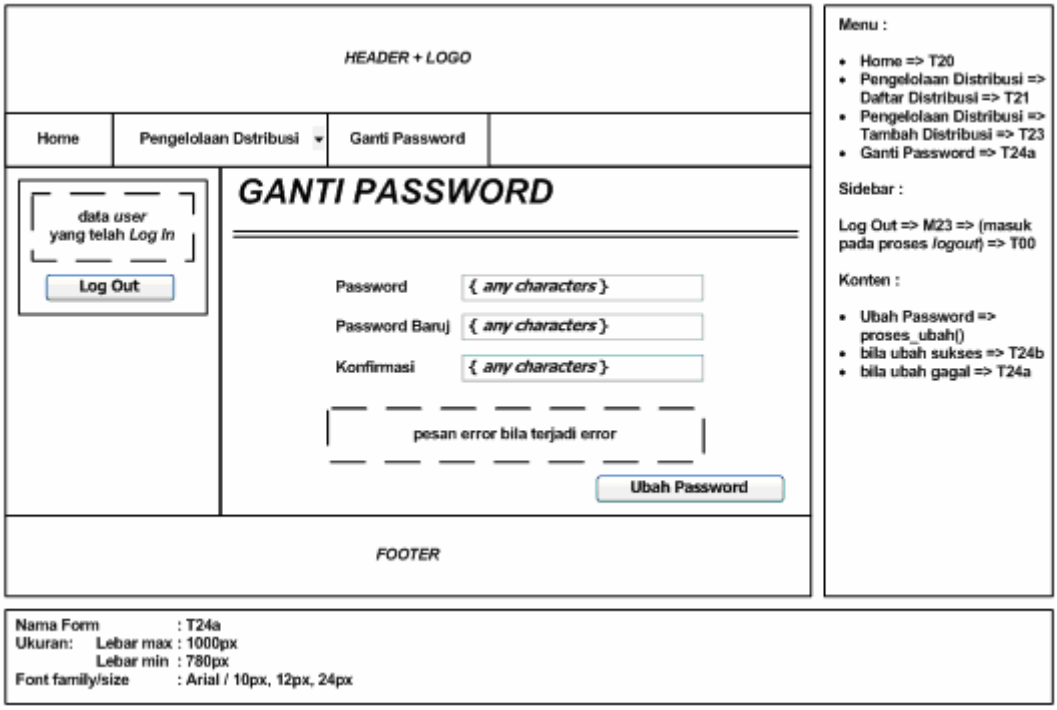
Gambar 3.50 Tambah Distribusi (T23a)

24. Ubah Distribusi



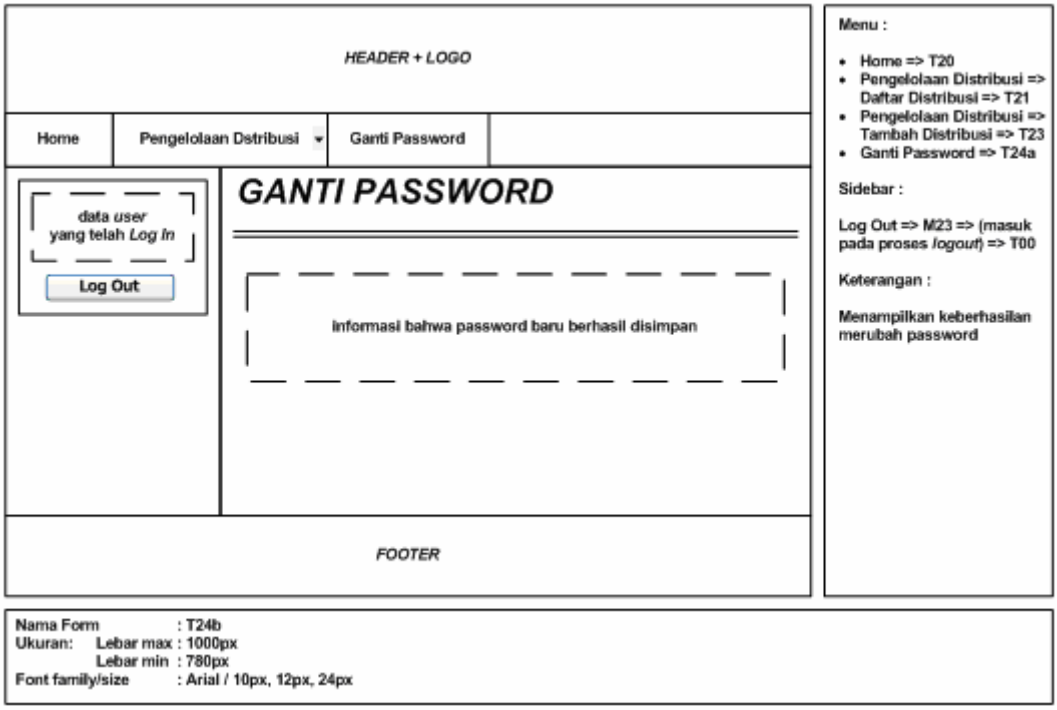
Gambar 3.51 Ubah Distribusi (T23b)

25. Ganti Password (memasukkan password baru)



Gambar 3.52 Ganti password (T24a)

26. Ganti Password (sukses)



Gambar 3.53 Informasi keberhasilan ganti password (T24b)

27. Pesan yang muncul pada saat terjadi salah masukan

Rancangan tampilan untuk pesan dapat dilihat mulai gambar 3.54.



Gambar 3.54 Pesan M01



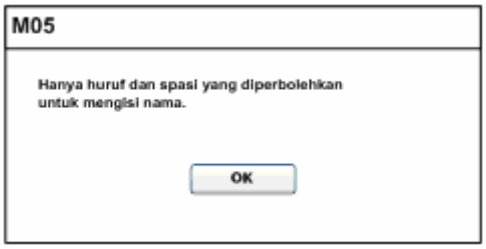
Gambar 3.55 Pesan M02



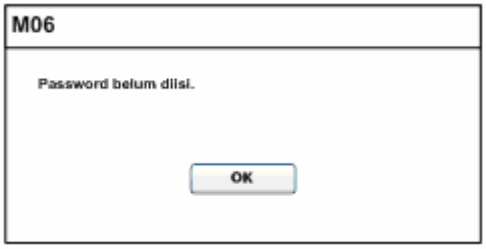
Gambar 3.56 Pesan M03



Gambar 3.57 Pesan M04



Gambar 3.58 Pesan M05



Gambar 3.59 Pesan M06



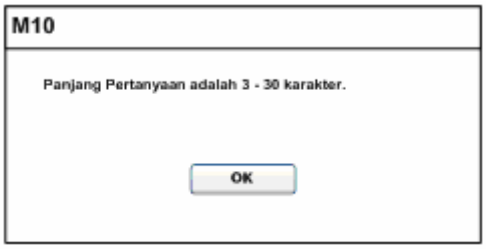
Gambar 3.60 Pesan M07



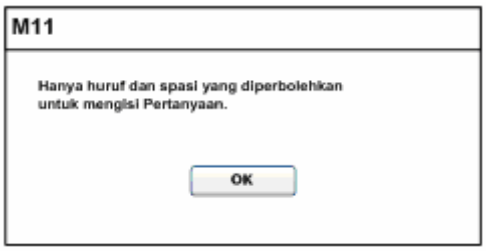
Gambar 3.61 Pesan M08



Gambar 3.62 Pesan M09



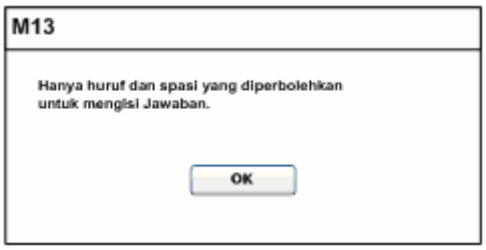
Gambar 3.63 Pesan M10



Gambar 3.64 Pesan M11



Gambar 3.65 Pesan M12



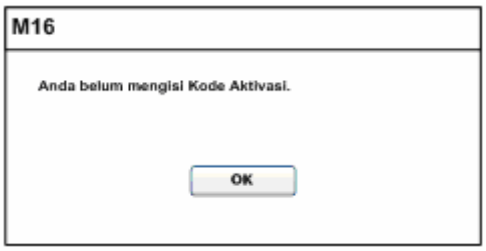
Gambar 3.66 Pesan M13



Gambar 3.67 Pesan M14



Gambar 3.68 Pesan M15



Gambar 3.69 Pesan M16



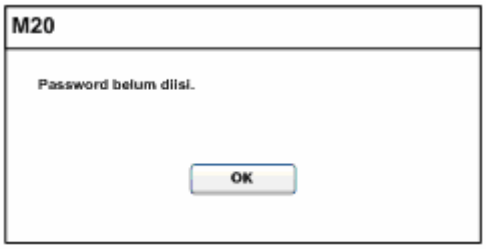
Gambar 3.70 Pesan M17



Gambar 3.71 Pesan M18



Gambar 3.72 Pesan M19



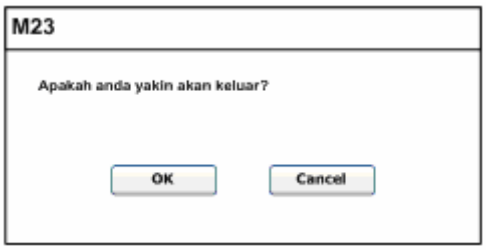
Gambar 3.73 Pesan M20



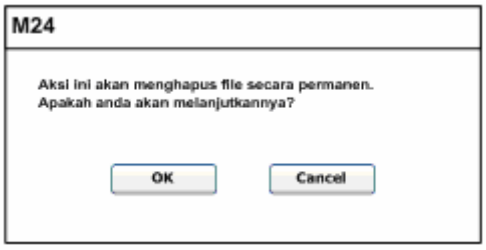
Gambar 3.74 Pesan M21



Gambar 3.75 Pesan M22



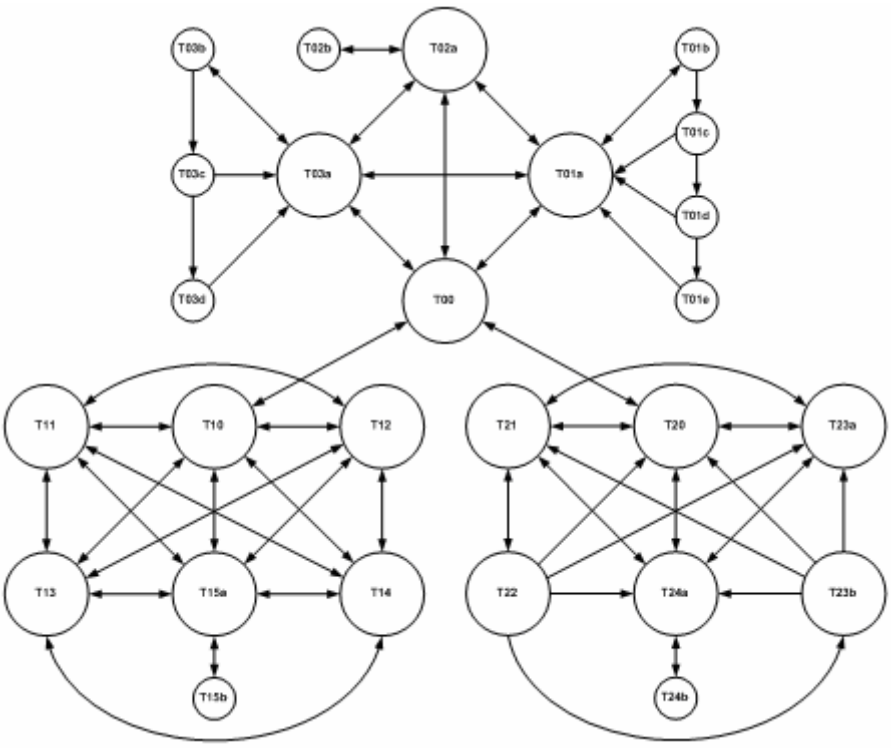
Gambar 3.76 Pesan M23



Gambar 3.77 Pesan M24

28. Jaringan Simantik

Jaringan simantik merupakan hubungan antara halaman-halaman tampilan merupakan urutan pengaksesan.



Gambar 3.78 Jaringan Simantik