BAB III
ANALISIS DAN DESAIN SISTEM

III.1. Analisis Masalah

Pada bab ini akan dijelaskan tentang tampilan hasil dari aplikasi yang telah dibuat, yang digunakan untuk memperjelas tentang tampilan-tampilan yang ada pada penerapan algoritma *blowfish* untuk database pendaftaran siswa baru (studi kasus: SMK Penerbangan PBD Medan). yaitu mengetahui, sehingga hasil implementasinya dapat dilihat sesuai dengan hasil program yang telah dibuat. Adapun tiap-tiap tampilan yang ada pada program adalah sebagai berikut:

III.2. Penerapan Metode *Blowfish*

Untuk lebih memahami proses enkripsi pada algoritma *blowfish*, maka penulis membuat contoh, perhitungan manual yang terjadi pada proses enkripsi. Dalam hal ini, penulis menggunakan parameter sebagai berikut (Hendri maradona, 2017) :

Plaintext = UTAMA_PU

Password = 2017

Langkah perhitungan manual yang penulis lakukan, yaitu sebagai berikut :

1. Inisialisasi *P-Array* (P_0, P_1, ..., P_{17}) masing-masing 32 bit, seperti Table III.1.

Tabel III.1 ***P-Array Konversi ke Biner***

P-array	Hexa	Konversi Biner (32 bit)
P0	243F6A88	00100100 00111111 01101010 10001000
P1	85A308D3	10000101 10100011 00001000 11010011
P2	13198A2E	00010011 00011001 10001010 00101110
P3	3707344	00000011 01110000 01110011 01000100
P4	A4093822	10100100 00001001 00111000 00100010
P5	299F31D0	00101001 10011111 00110001 11010000
P6	82EFA98	00001000 00101110 11111010 10011000
P7	EC4E6C89	11101100 01001110 01101100 10001001
P8	452821E6	01000101 00101000 00100001 11100110
P9	38D01377	00111000 11010000 00010011 01110111
P10	BE5466CF	10111110 01010100 01100110 11001111
P11	34E90C6C	00110100 11101001 00001100 01101100
P12	C0AC29B7	11000000 10101100 00101001 10110111
P13	C97C50DD	11001001 01111100 01010000 11011101
P14	3F84D5B5	00111111 10000100 11010101 10110101
P15	B5470917	10110101 01000111 00001001 00010111

P16	9216D5D9	10010010 00010110 11010101 11011001
P17	8979FB1B	10001001 01111001 11111011 00011011

1. Inisialisasi *S-Array* yang berjumlah masing-masing 255 dalam bentuk *hexadecimal* yang kemudian dikonversi ke biner, seperti Tabel III.2.

Tabel III.2. Konversi *S-Array* ke Biner

S-Array	Hexa	Konversi Biner
S1,0 ... S1,255	D1310BA6 6E85076A	11010001 00110001 00001011 10100110 01101110 10000101 00000111 01101010
S2,0 ... S2,255	4B7A70E9 DB83ADF7	01001011 01111010 01110000 11101001 11011011 10000011 10101101 11110111
S3,0 ... S3,255	E93D5A68 406000E0	11101001 00111101 01011010 01101000 01000000 01100000 00000000 11100000
S4,0 ... S4,255	3A39CE37 3AC372E6	00111010 00111001 11001110 00110111 00111010 11000011 01110010 11100110

2. *Plaintext* = UTAMA_PU

Tabel III.3. Konversi *Plaintext* Ke Biner

Karakter	ASCII (Hexa)	Biner
U	55	01010101
T	54	01010100
A	41	01000001
M	4D	01001101
A	41	01000001
_	5F	01011111
P	50	01010000
U	55	01010101

3. Kemudian *Plaintext* dibagi menjadi 2 bagian *XL* dan *XR* menjadi :

XL = 01010101 01001001 01001110 01010011

XR = 01010101 01011111 01010000 01010101

Pembangkitan Sub Kunci :

Kunci : 2017

Tabel III.4. Konversi Kunci Ke Biner

Karakter	ASCII (Hexa)	Biner
2	32	00110010
0	30	00110000
1	31	00110001
7	37	00110111

Biner : 00110010 00110000 00110001 00110111

a. SubKunci Untuk Iterasi Pertama :

P_0 = P_0 XOR Kunci

P_0 = 00100100 00111111 01101010 10001000 XOR

00110010 00110000 00110001 00110111

P_0 = 00010110 00000110 01011010 10111111

b. SubKunci untuk iterasi kedua :

P_1 =P_1 XOR P_0

P_1 = 10000101 10100011 00001000 11010011 XOR

00010110 00000110 01011010 10111111

P_1 = 10010011 10100101 01010010 01101100

4. Dalam hal ini penulis, hanya melakukan satu iterasi, dikarenakan total iterasi proses enkripsi adalah 16 putaran.

a. Untuk iterasi pertama i = 0 yaitu :

XL =XL XOR P_0

XL = 01010101 01010000 01001001 00100000 XOR

00010110 00000110 01011010 10111111

XL = 01000011 01010110 00010011 10011111

Fungsi F didapat dari :

XL dibagi menjadi 4 (a, b, c, d) masing-masing 8 bit = a

= 01000011

b = 01010110

c = 00010011

d = 10011111

Fungsi F :$F(XL)$ = $(((S_0.a + S_1.b \bmod 2^{32}) \text{ XOR } S_{2,} c) + S_{3,}d \bmod 2^{32})$

= (11010001 00110001 00001011 10100110 . 01000011) + (01001011 01111010 01110000 11101001. 01010110) mod 2^{32}

=(11011010111111110101100000110001110010 + 11001010110110010000111101110010001 10) mod 2^{32}

= 00011010 11110111 11111010 10111000

XOR S_2.c = 00011010 11110111 11111010 10111000 XOR (11101001 00111101 01011010 01101000. 11100100)

= 00011010 11110111 1111101010111000 XOR 11001111101110101010010010000100101000 00

= 11001111 10100000 01010011 01111110 00011000 + S3.d mod 2^{32}

= (11001111 10100000 01010011 01111110 00011000 + (00111010 00111001 11001110 00110111. 10011111)) mod 2^{32}

= (11001111 10100000 01010011 01111110 00011000 + 10010000101001111001110001010000101001) mod 2^{32}

11001010 00111010 10010010 01000001

F(XL) = 11001010 00111010 10010010 01000001

XR = *F(XL)* XOR *XR*

XR =11001010 00111010 10010010 01000001 XOR 10010001 01111100 00111001 00111100

XR =01011011 01000110 10101011 01111101

Menukar Nilai *XL* dan *XR* :

XL = XR ;*XR = XL*

XL = 01011011 01000110 10101011 01111101;

XR = 01000011 01010110 00010011 10011111

7. Setelah melakukan 16 iterasi, maka akan menghasilkan nilai baru *XL* dan *XR* masingmasing 32 bit. Tukar kembali *XL* dan *XR*. Setelah itu XOR-kan nilai *XL* dan *XR* : *XR = XR* XOR P_{16} dan *XL = XL* XOR P_{17}
8. Kemudian *XL* dan *XR* digabungkan sehingga menjadi 64 bit.
9. Nilai biner tersebut di konversikan ke dalam kode ASCII sehingga menghasilkan *ciphertext* yaitu : 1yCUUzcfO4k=

Untuk lebih memahami proses dekripsi pada algoritma *blowfish*, maka penulis membuat contoh, perhitungan manual yang terjadi pada proses enkripsi. Dalam hal ini, penulis menggunakan parameter sebagai berikut :

Ciphertext = 1yCUUzcfO4k=

Password = 2017

Langkah perhitungan manual yang penulis lakukan, yaitu sebagai berikut:

1. Inisialisasi *P-Array* (P_0, P_1, ..., P_{17}) masing-masing 32 bit, seperti Tabel III.5.

Tabel III.5. Konversi *P-Array* Ke Biner

Parr ay	Hexa	Konversi Biner (32 bit)
P0	243F6A88	00100100 00111111 01101010 10001000

P1	85A3 08D3	10000101 10100011 00001000 11010011
P2	13198A2E	00010011 00011001 10001010 00101110
P3	3707344	00000011 01110000 01110011 01000100
P4	A409 3822	10100100 00001001 00111000 00100010
P5	299F31D0	00101001 10011111 00110001 11010000
P6	82EFA98	00001000 00101110 11111010 10011000
P7	EC4E6C89	11101100 01001110 01101100 10001001
P8	452821E6	01000101 00101000 00100001 11100110
P9	38D0 1377	00111000 11010000 00010011 01110111
P10	BE54 66CF	10111110 01010100 01100110 11001111
P11	34E90C6C	00110100 11101001 00001100 01101100
P12	C0AC29B7	11000000 10101100 00101001 10110111
P13	C97C50DD	11001001 01111100 01010000 11011101
P14	3F84D5B5	00111111 10000100 11010101 10110101
P15	B547 0917	10110101 01000111 00001001 00010111
P16	9216D5D9	10010010 00010110 11010101 11011001
P17	8979FB1B	10001001 01111001 11111011 00011011

2. Inisialisasi *S-Array* yang berjumlah masing-masing 255 dalam bentuk hexadecimal yang kemudian dikonversi ke biner, seperti Tabel III.6.

Tabel III.6. Konversi *S-Array* ke Biner

S-Array	Hexa	Konversi Biner
S1,0 ... S1,255	D1310BA6 6E85076A	11010001 00110001 00001011 10100110 01101110 10000101 00000111 01101010
S2,0 ... S2,255	4B7A70E9 DB83ADF7	01001011 01111010 01110000 11101001 11011011 10000011 10101101 11110111
S3,0 ... S3,255	E93D5A68 406000E0	11101001 00111101 01011010 01101000 01000000 01100000 00000000 11100000
S4,0 ... S4,255	3A39CE37 3AC372E6	00111010 00111001 11001110 00110111 00111010 11000011 01110010 11100110

3. *Ciphertext* = 1yCUUzcfO4k=

Tabel III.7. Konversi *Ciphertext* ke Biner

Karakter	ASCII(*Decimal*)	Biner
1	49	00110001
Y	89	01011001
C	67	01000011

U	85	01010101
U	85	01010101
Z	122	01111010
C	99	01100011
F	102	01100110
O	79	01001111
4	52	00110100
K	107	01101011
=	61	00111101

4. Kemudian dibagi menjadi 2 bagian *XL* dan *XR* menjadi :

XL = 00110001 01011001 01000011 01010101 01010101 01111010

XR = 01100011 01100110 01001111 00110100 01101011 00111101

5. Pembangkitan Sub Kunci :

Kunci : 2017

Tabel III.8. Konversi Kunci Ke Biner

Karakter	ASCII (Hexa)	Biner
2	32	00110010
0	30	00110000
1	31	00110001
7	37	00110111

Biner : 00110010 00110000 00110001 00110111

a. SubKunci Untuk Iterasi Pertama :

P_0 = P_0 XOR Kunci

P_0 = 00100100 00111111 01101010 10001000 XOR

00110010 00110000 00110001 00110111

P_0 = 00010110 00000110 01011010 10111111

b. SubKunci untuk iterasi kedua :

P_1 =P_1 XOR P_0

P_1 = 10000101 10100011 00001000 11010011 XOR

00010110 00000110 01011010 10111111

P_1 = 10010011 10100101 01010010 01101100

6. Dalam hal ini penulis, hanya melakukan dua iterasi, dikarenakan total iterasi proses dekripsi adalah 16 putaran. Untuk iterasi pertama i = 0 yaitu :

   XL =XL XOR P_{17}

   XL = 00110001 01011001 01000011 01010101 01010101 01111010 XOR

   10111011 01000000 11001011 00101110

   XL = 00110001 01011001 01000011 01010101 01010101 01111010

   Fungsi F didapat dari :

   XL dibagi menjadi 4 (a, b, c, d) masing-masing 8 bit =

   a = 00100001

   b = 01101101

   c = 11100100

   d = 00000100

   Fungsi F :$F(XL)$ = $(((S_0.a + S_1.b \bmod 2^{32}) \text{ XOR } S_{2,}\, c) + S_{3,}d \bmod 2^{32})\, S_0.a +$ $S_1.b \bmod 2^{32}$ = (11010001 00110001 00001011 10100110 . 00100001) + (01001011 01111010 01110000 11101001. 01101101) mod 2^{32}

   =(00011010 11110111 01010010 10000000 01100110 +00100000 00100011 00100010 00010011 00110101) mod 2^{32}

   = 00011010 01110100 10010011 10011011

XOR S_2.c = 00011010 01110100 10010011 10011011 XOR (11101001 00111101 01011010 01101000. 11100100)

= 00011010 01110100 10010011 10011011 XOR 11001111 10111010 10100100 10000100 10100000

= 11001111 10100000 11010000 00010111 00111011 + S3.d mod 2^{32} =

(11001111 10100000 11010000 00010111 00111011 +

00111010 00111001 11001110 00110111 . 00000100) mod 2^{32}

= (11001111 10100000 11010000 00010111 00111011 +

11101000111001110011100011011100) mod 2^{32}

= 10001001 10110111 01010000 00010111

F(XL) = 10001001 10110111 01010000 00010111

XR = *F(XL)* XOR *XR*

XR= 10001001 10110111 01010000 00010111 XOR

10010001 01111100 00111001 00111100

XR= 00011000 11001011 01101001 00101011

Menukar Nilai *XL* dan *XR* :

XL = *XR* ;*XR* = *XL*

XL = 00011000 11001011 01101001 00101011;

XR = 00100001 01101101 11100100 00000100

7. Setelah melakukan 16 iterasi, maka akan menghasilkan nilai baru *XL* dan *XR* masingmasing 32 bit. Tukar kembali *XL* dan *XR*. Setelah itu XOR-kan nilai *XL* dan *XR* : *XR* = *XR* XOR P_1dan *XL* = *XL* XOR P_0

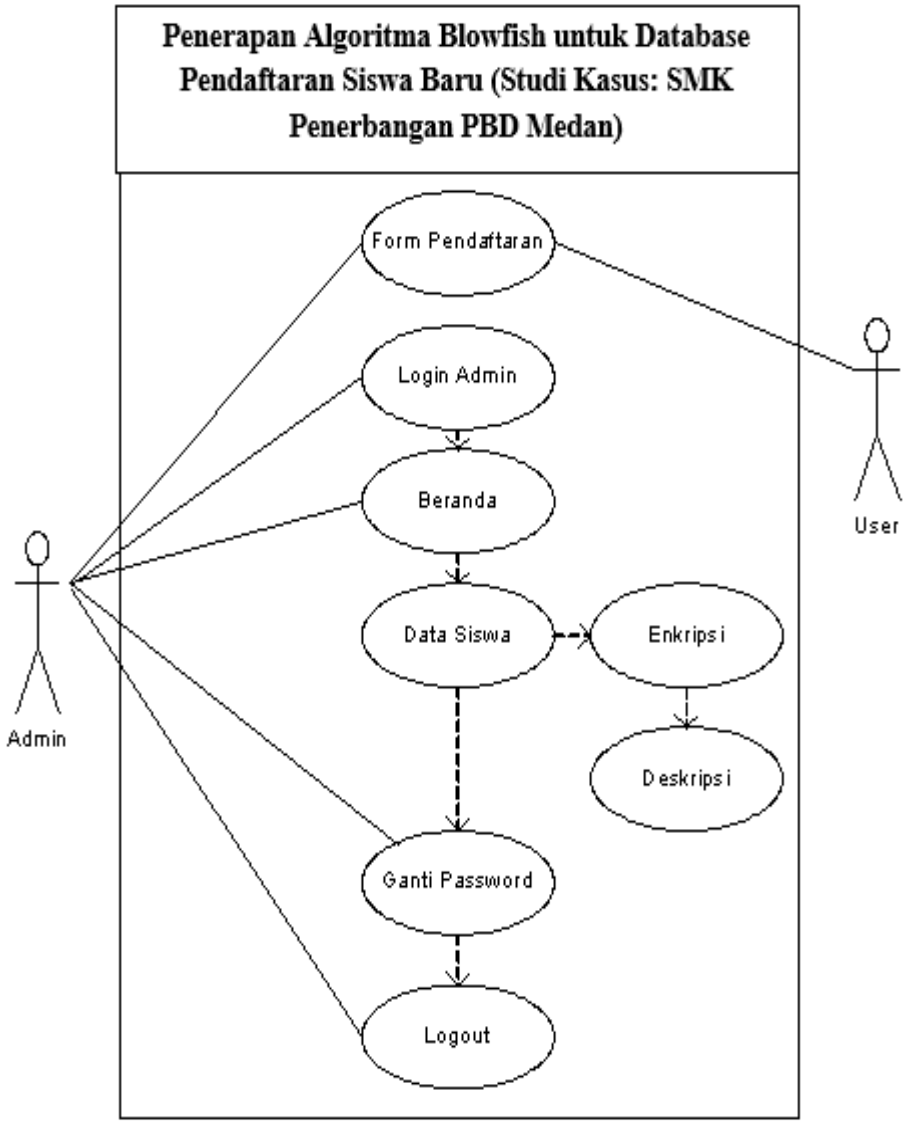
8. Kemudian *XL* dan *XR* digabungkan sehingga menjadi 64 bit
9. Nilai biner tersebut di konversikan ke dalam kode ASCII sehingga menghasilkan plaintext yaitu : UTAMA_PU

III.3. Desain Sistem

Desain sistem yang peneliti gunakan adalah pemodelan *Unified Modeling Language* (UML). Berikut ini adalah beberapa pemodelan *Unified Modeling Language* (UML) yang peneliti gunakan:

III.3.1. *Use Case Diagram*

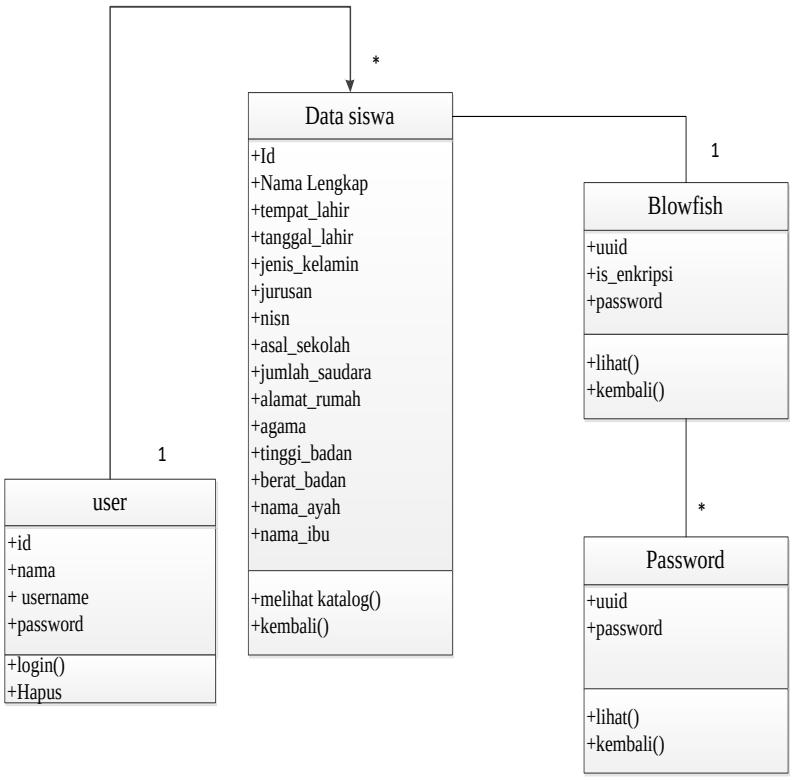
Dalam penyusunan suatu program diperlukan suatu model data yang berbentuk diagram yang dapat menjelaskan suatu alur proses sistem yang akan di bangun. Dalam penulisan skripsi ini penulis menggunakan metode UML yang dalam metode itu penulis menerapkan diagram *Use Case*. Maka digambarlah suatu bentuk diagram *Use Case* yang dapat dilihat pada Gambar III.1:



Gambar III.1. *Use Case Diagram* Penerapan Algoritma *Blowfish* untuk *Database* Pendaftaran Siswa Baru (Studi Kasus: SMK Penerbangan PBD Medan)

III.3.2. *Class Diagram*

Rancangan kelas-kelas yang akan digunakan pada sistem yang akan dirancang dapat dilihat pada Gambar III.2 :



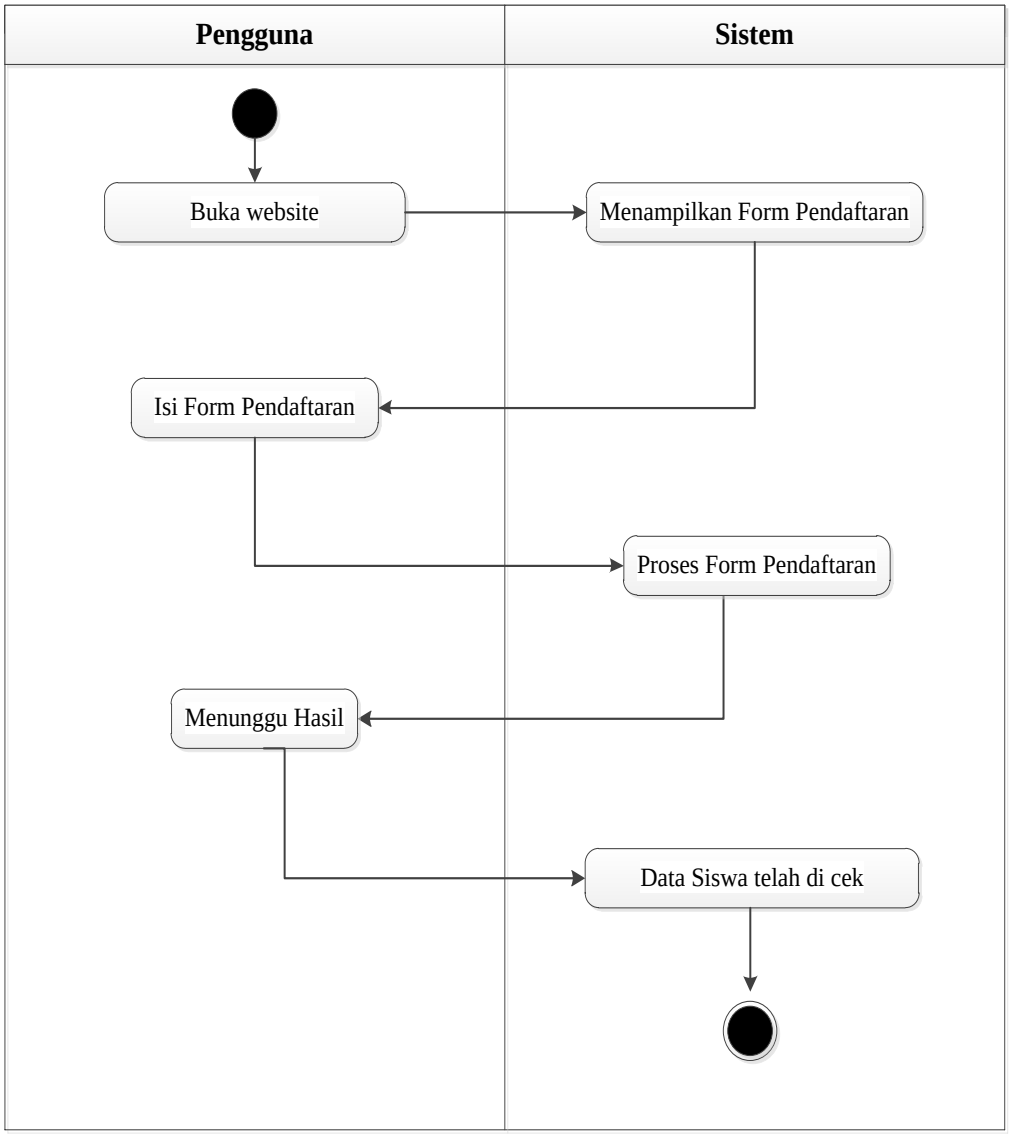
Gambar III.2. *Class Diagram* Penerapan Algoritma *Blowfish* untuk *Database* Pendaftaran Siswa Baru (Studi Kasus: SMK Penerbangan PBD Medan)

III.3.3. *Activity Diagram*

Bisnis proses yang telah digambarkan pada *usecase diagram* dijabarkan dengan *Activity diagram..Activity diagram* merupakan suatu diagram yang dapat menampilkan secara detail urutan.

III.3.3.1. *Activity Diagram Form* Pendaftaran

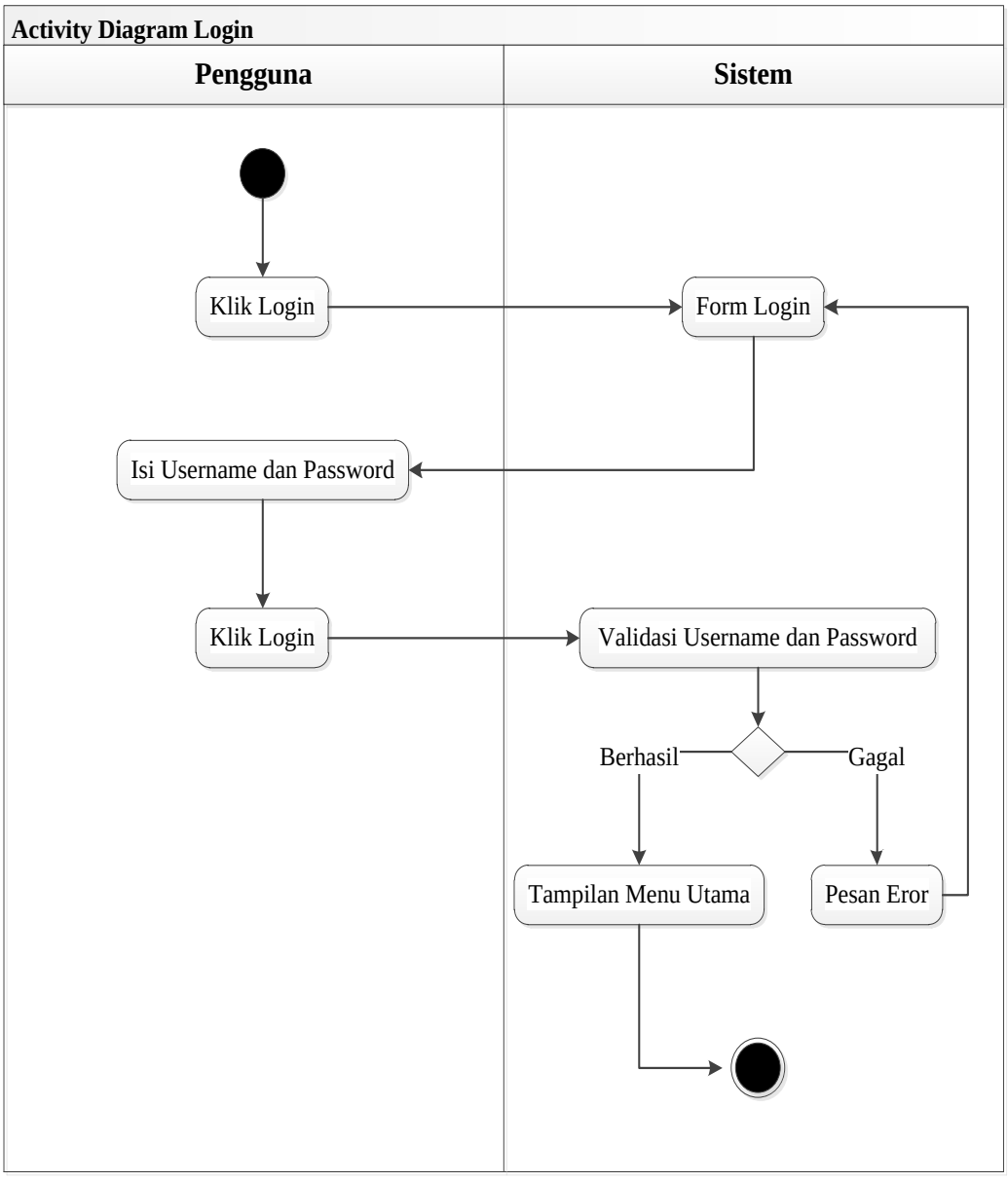
Activity Diagram form pendaftaran yaitu form yang harus di isi oleh siswa yang mendaftar dilakukan oleh user dapat diterangkan dengan langkah-langkah *state* berikut yang ditunjukkan pada Gambar III.3 :



Gambar III.3. *Activity Diagram Form* Pendaftaran

III.3.3.2. *Activity Diagram Login*

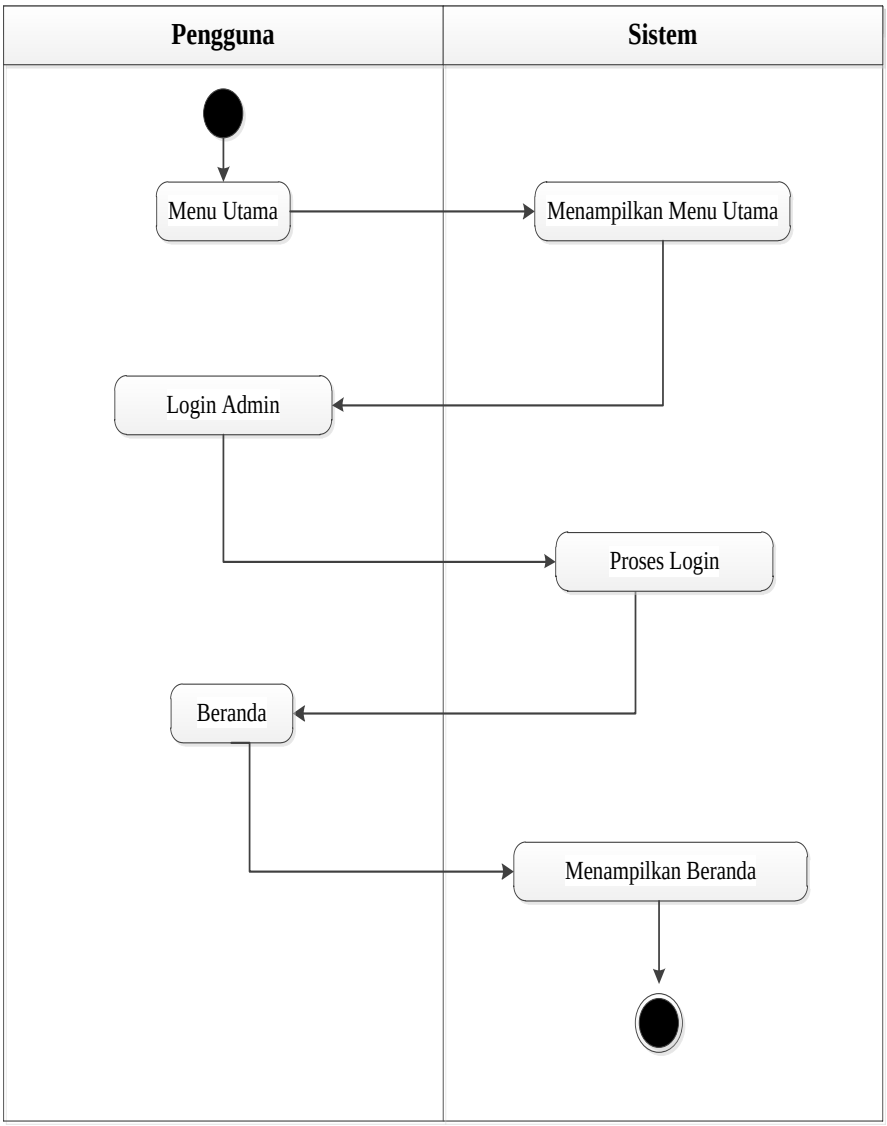
Activity Diagram login yang dilakukan oleh user dapat diterangkan dengan langkah-langkah *state* berikut yang ditunjukkan pada Gambar III.4 :



Gambar III.4. *Activity Diagram Login*

III.3.3.3. *Activity Diagram* Beranda

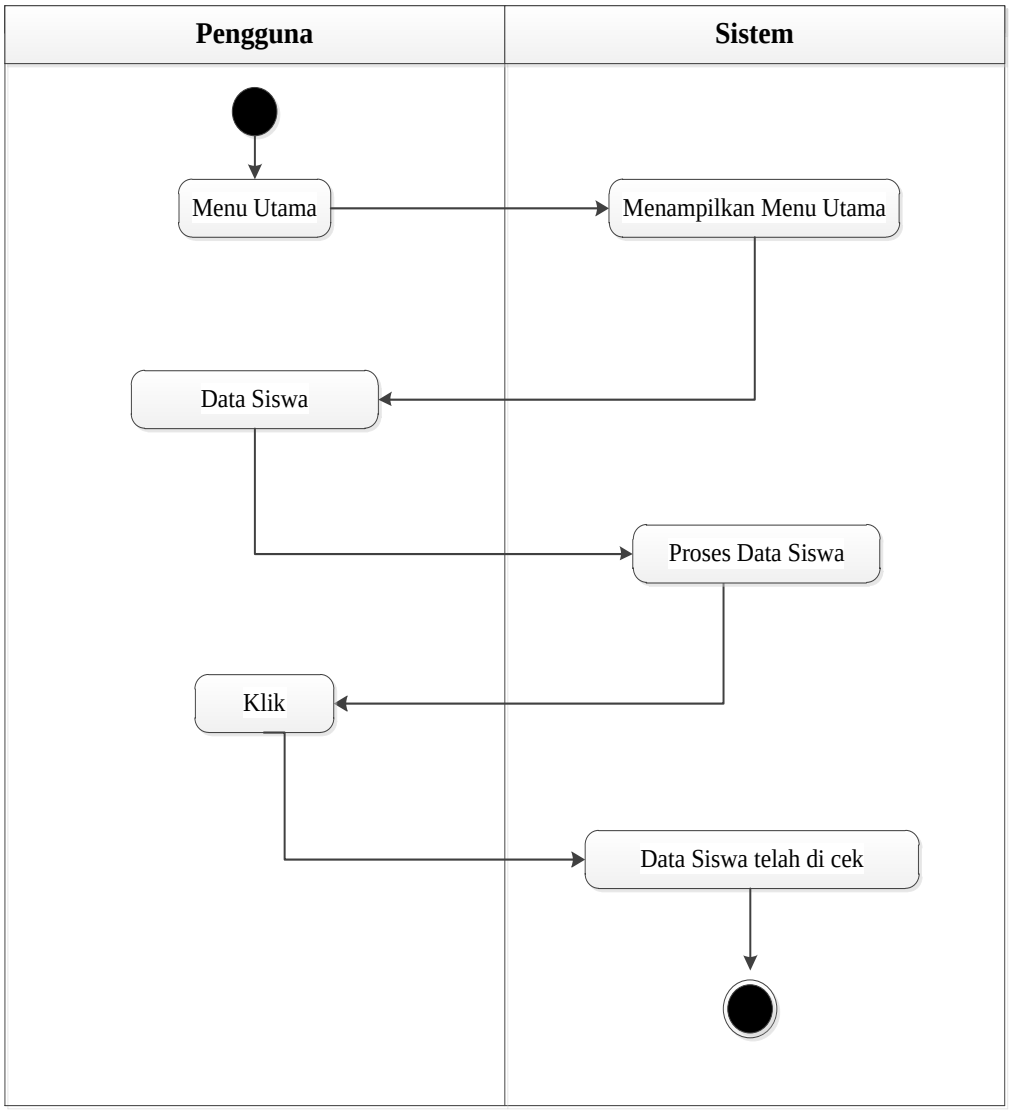
Activity Diagram beranda adalah tampilan awal setelah admin login dapat diterangkan dengan langkah-langkah *state* berikut yang ditunjukkan pada Gambar III.5:



Gambar III.5. *Activity Diagram* Beranda

III.3.3.4. *Activity Diagram* Data Siswa

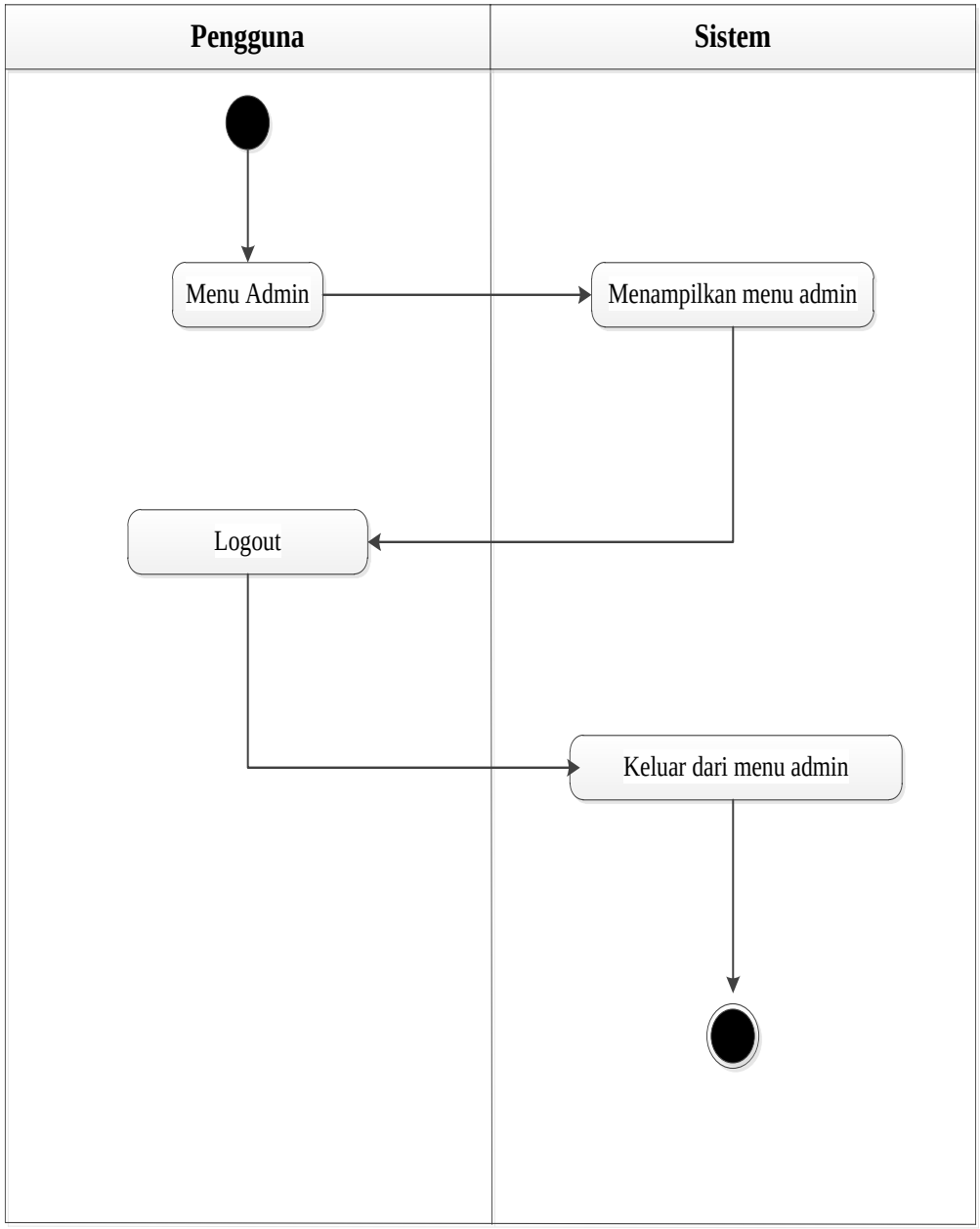
Activity Diagram data siswa yang akan dicek oleh admin apakah siswa itu bisa masuk atau tidak disekolah SMK Penerbangan PBD Medan dapat diterangkan dengan langkah-langkah *state* berikut yang ditunjukkan pada Gambar III.6:



Gambar III.6. *Activity Diagram* Data Siswa

III.3.3.6. *Activity Diagram Logout*

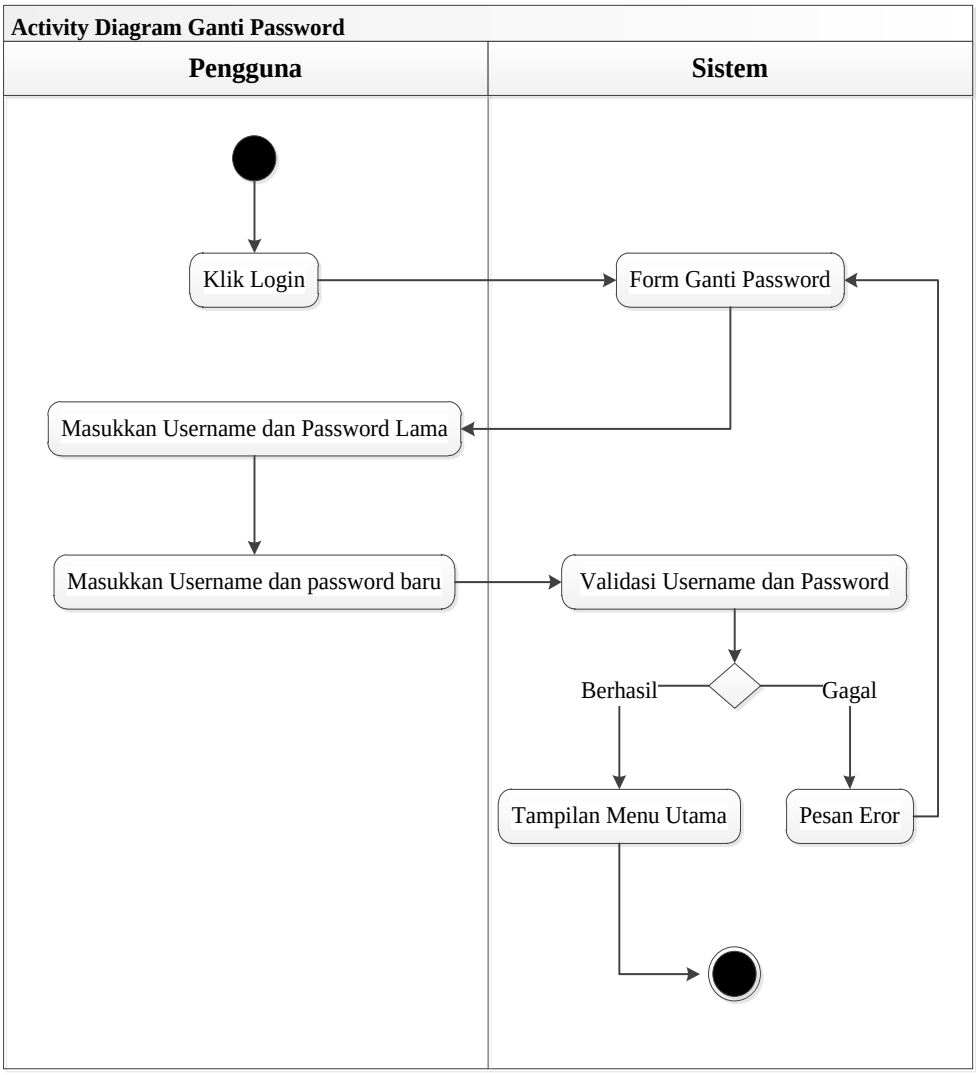
Activity Diagram Logout untuk keluar dari menu *admin* yang dilakukan oleh *admin* dapat diterangkan dengan langkah-langkah *state* berikut yang ditunjukkan pada Gambar III.8:



Gambar III.8. ***Activity Diagram Logout***

III.3.3.5. *Activity Diagram* Ganti *Password*

Activity Diagram ganti *password* yang dilakukan oleh admin dapat diterangkan dengan langkah-langkah *state* berikut yang ditunjukkan pada Gambar III.7:



Gambar III.7. *Activity Diagram* Ganti *Password*

III.3.4. *Sequence Diagram*

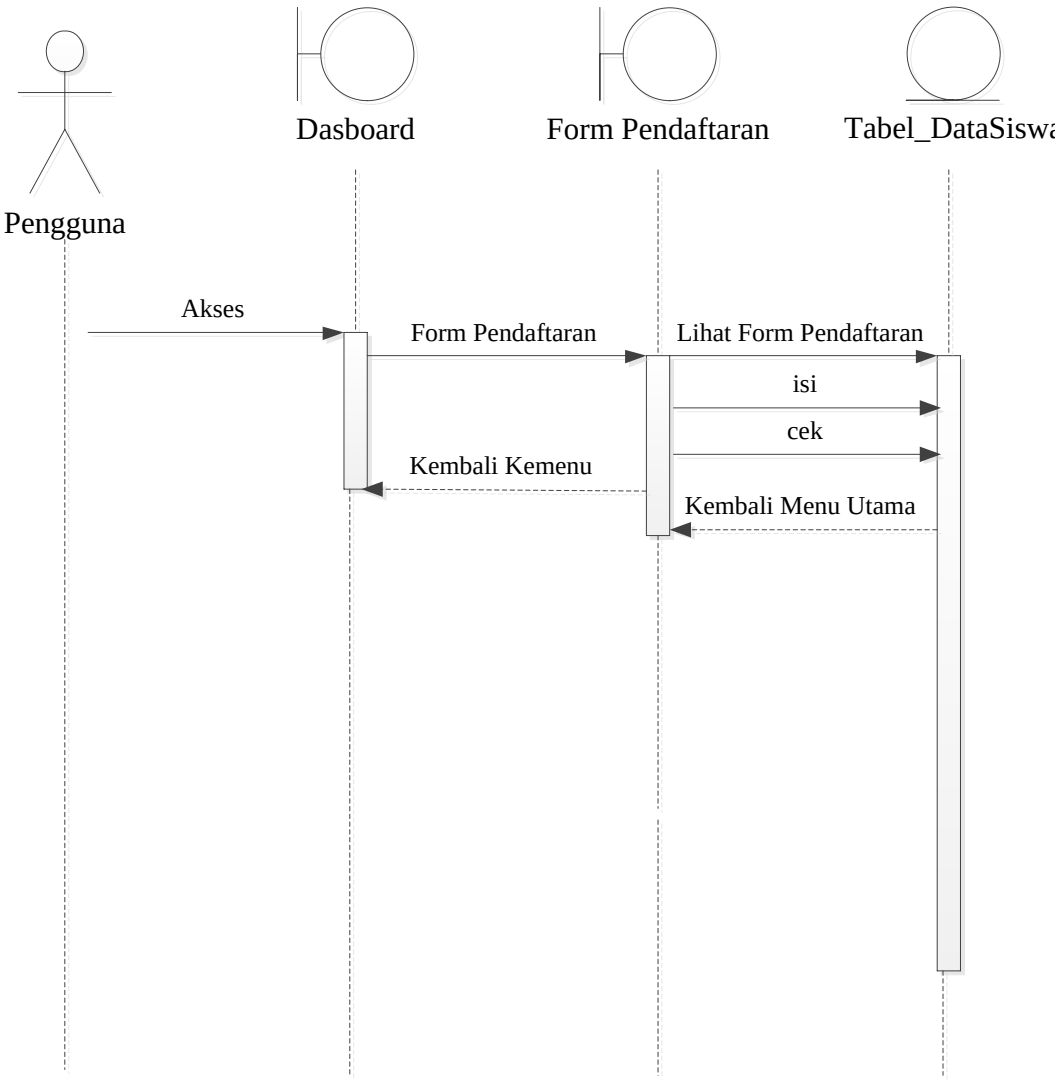
Sequence Diagram (diagram urutan) adalah suatu diagram yang memperlihatkan atau menampilkan interaksi-interaksi antar objek di dalam sistem yang disusun pada sebuah urutan atau rangkaian waktu. Interaksi antar objek tersebut termasuk pengguna, *display*, dan sebagainya berupa pesan/*message*.

Sequence Diagram digunakan untuk menggambarkan skenario atau rangkaian langkah – langkah yang dilakukan sebagai sebuah respon dari suatu kejadian/*event* untuk menghasilkan *output* tertentu. *Sequence Diagram* diawali dari apa yang me-*trigger* aktivitas tersebut, proses dan perubahan apa saja yang terjadi secara *internal* dan *output* apa yang dihasilkan.

Diagram ini secara khusus berasosiasi dengan *use case diagram*.*Sequence* diagram juga memperlihatkan tahap demi tahap apa yang seharusnya terjadi untuk menghasilkan sesuatu di dalam *use case*. *Sequence* Diagram juga dapat merubah atribut atau *method* pada *class* yang telah dibentuk oleh *class diagram*, bahkan menciptakan sebuah *class* baru. *Sequence* Diagram memodelkan aliran logika dalam sebuah sistem dalam cara yang visual.

III.3.4.1. *Sequence Diagram Form* Pendaftaran

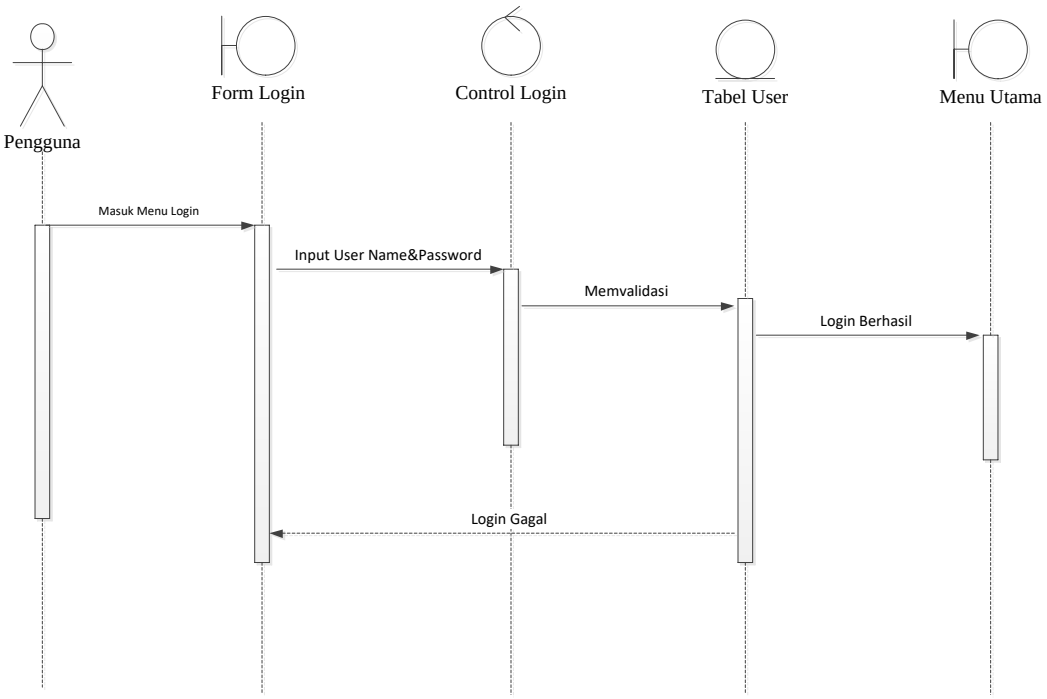
Serangkaian kegiatan *form pendaftaran* yang dilakukan oleh *user* dapat diterangkan dengan langkah-langkah *state* berikut,yang ditunjukkan pada Gambar III.9 :



Gambar III.9. *Sequence Diagram* Pendaftaran

III.3.4.2. *Sequence Diagram Login*

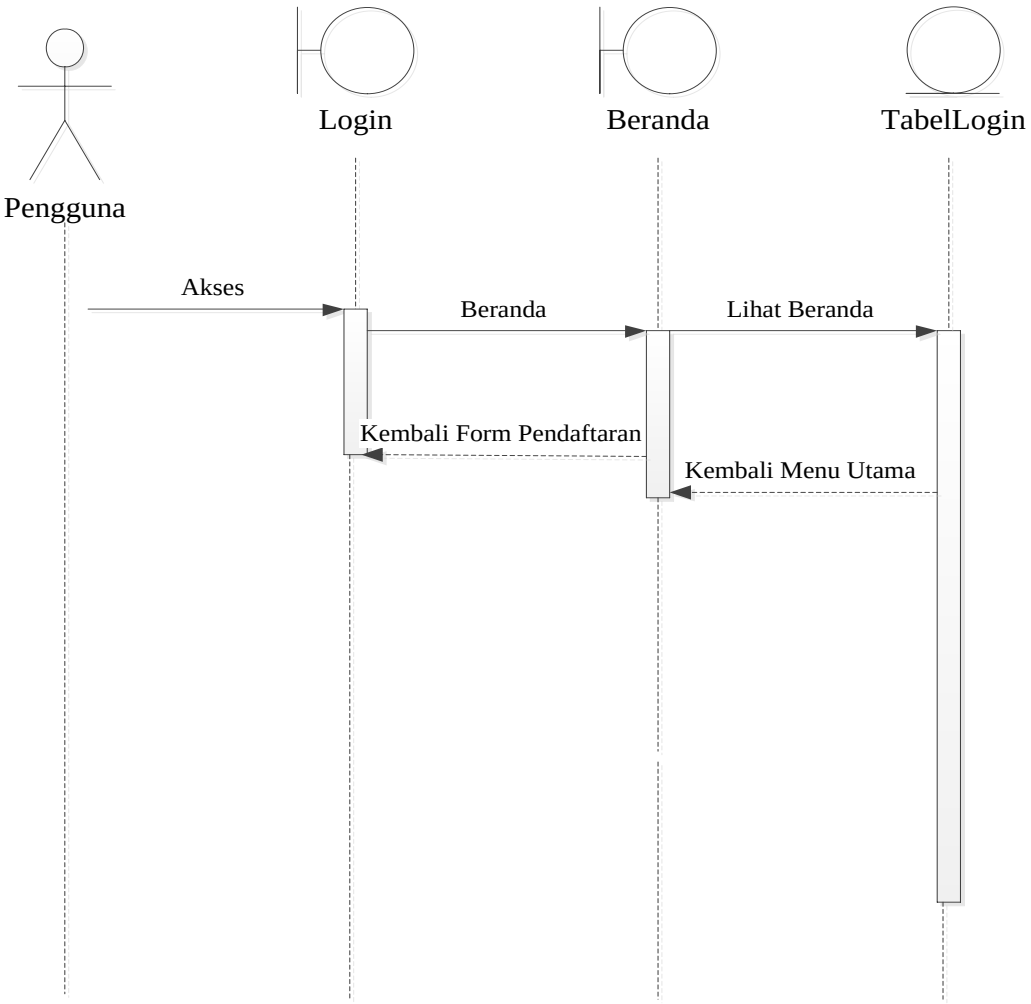
Serangkaian kegiatan *login* yang dilakukan oleh admin dapat diterangkan dengan langkah-langkah *state* berikut,yang ditunjukkan pada gambar III.10 :



Gambar III.10. *Sequence Diagram Login*

III.3.4.3. *Sequence Diagram* Beranda

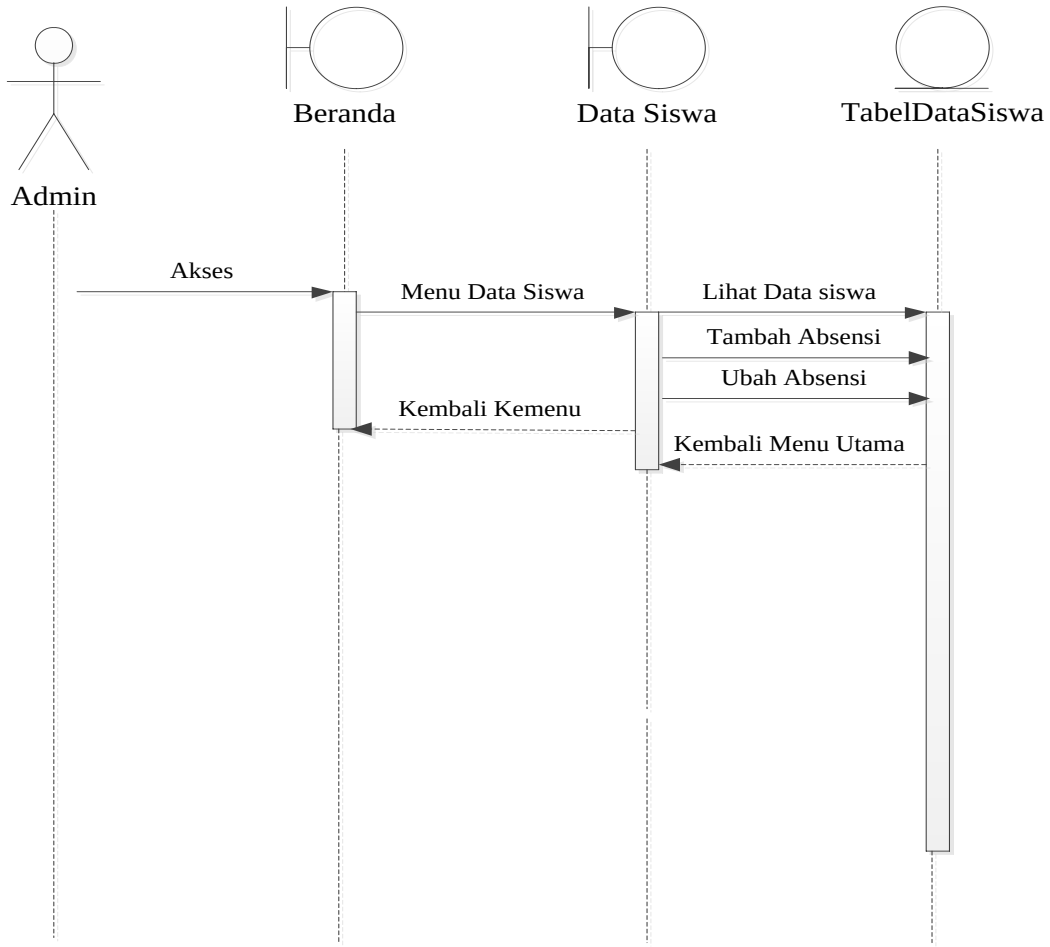
Serangkaian kegiatan pada tampilan beranda yang dilakukan oleh *admin* setelah login dapat diterangkan dengan langkah-langkah *state* berikut,yang ditunjukkan pada Gambar III.11 :



Gambar III.11. *Sequence Diagram* Beranda

III.3.4.4. *Sequence Diagram* Data Siswa

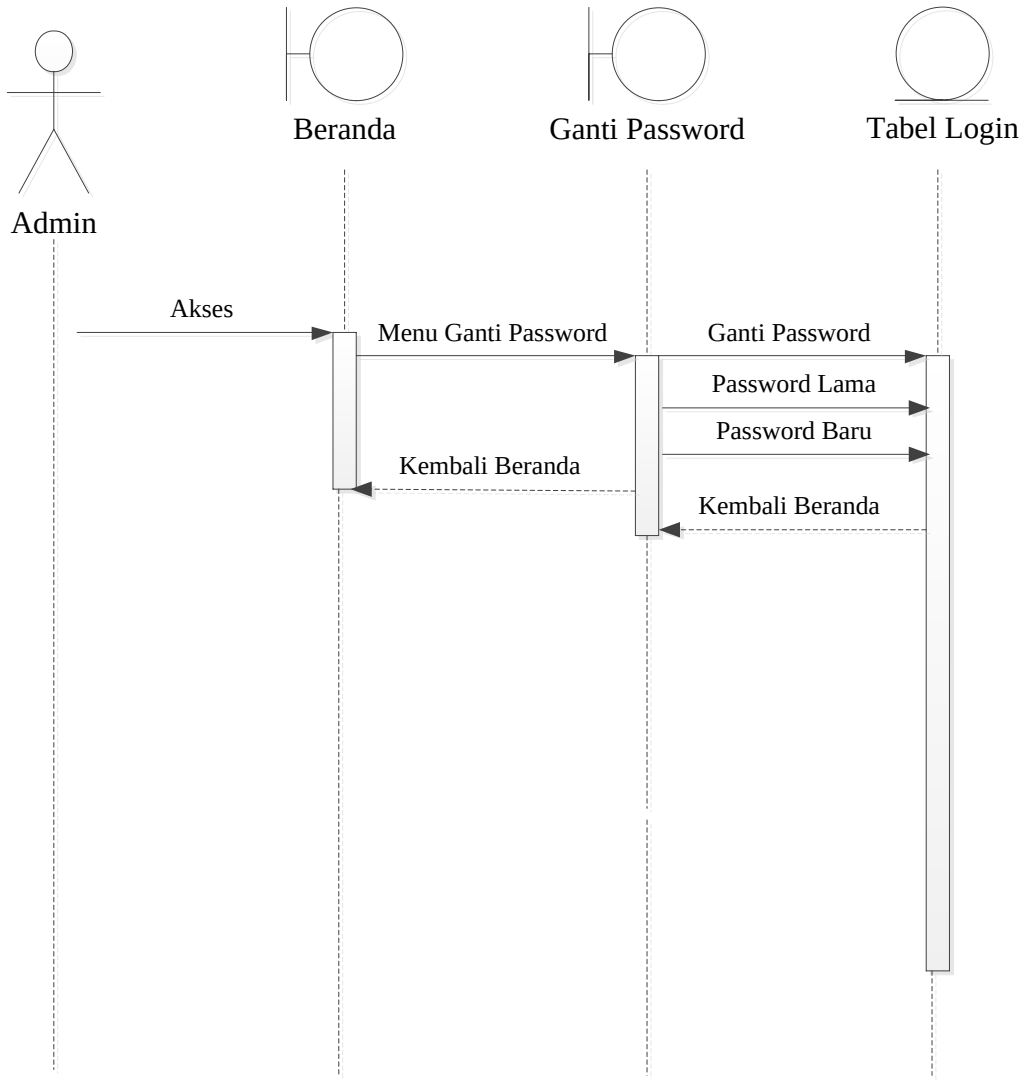
Serangkaian kegiatan data siswa yang dilakukan oleh *admin* dapat diterangkan dengan langkah-langkah *state* berikut, yang ditunjukkan pada Gambar III.12:



Gambar III.12. *Sequence Diagram* Data Siswa

III.3.4.5. *Sequence Diagram* Ganti *Password*

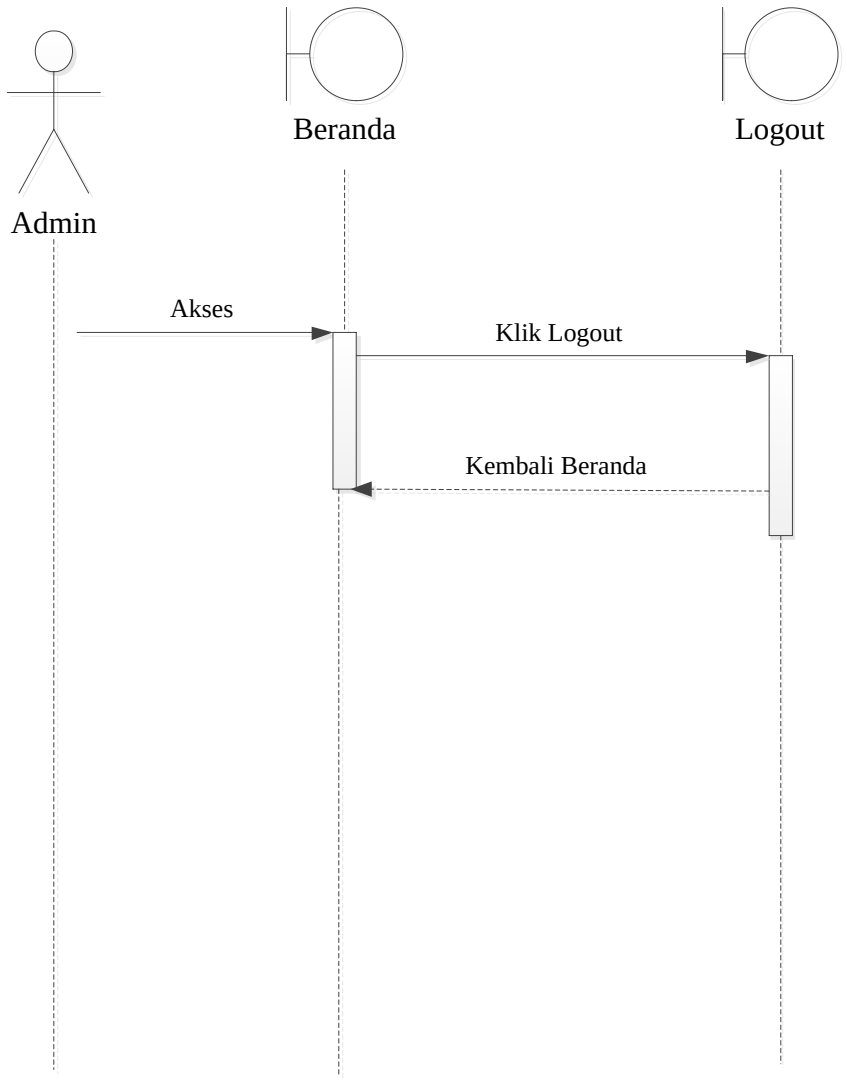
Serangkaian kegiatan ganti password yang dilakukan oleh *admin* dapat diterangkan dengan langkah-langkah *state* berikut, yang ditunjukkan pada Gambar III.13:



Gambar III.13. *Sequence Diagram* Ganti *Password*

III.3.4.6. *Sequence Diagram Logout*

Serangkaian kegiatan *Logout* yang dilakukan oleh *admin* dapat diterangkan dengan langkah-langkah *state* berikut, yang ditunjukkan pada Gambar III.14:



Gambar III.14. *Sequence Diagram Logout*

III.4. Desain *Database*

Desain tabel-tabel dari *database* yang terdapat pada Penerapan Algoritma *Blowfish* untuk *Database* Pendaftaran Siswa Baru (Studi Kasus: SMK Penerbangan PBD Medan) dimulai dari normalisasi kemudian isi desain *database*.

III.4.1. Normalisasi

Normalisasi digunakan untuk menghindari duplikasi terhadap tabel dalam basis data dan yang masih memiliki beberapa ketidakwajaran sehingga menghasilkan tabel yang lebih sederhana.

III.4.2. Desain Tabel

Berikut ini adalah desain tabel dari Penerapan Algoritma *Blowfish* untuk Database Pendaftaran Siswa Baru:

1. Perancangan Tabel Data Siswa

Tabel data siswa merupakan media untuk penyimpanan data siswa, struktur tabel data siswa dapat dilihat pada Tabel berikut;

Tabel III.9 Tabel Perancangan Data Siswa

No	Nama	*Data Type*	*Size*	Keterangan
1.	+Id	*Int*	11	Id
2.	+Nama Lengkap	*Varchar*	50	Nama Lengkap
3.	+tempat_lahir	*Varchar*	50	tempat_lahir
4.	+tanggal_lahir	*Varchar*	50	tanggal_lahir
5.	+jenis_kelamin	*Varchar*	50	jenis_kelamin
6.	+jurusan	*Varchar*	50	Jurusan
7.	+nisn	*Varchar*	50	Nisn
8.	+asal_sekolah	*Varchar*	50	asal_sekolah
9.	+jumlah_saudara	*Varchar*	50	jumlah_saudara
10.	+alamat_rumah	*Varchar*	50	alamat_rumah
11.	+agama	*Varchar*	50	Agama
12.	+tinggi_badan	*Varchar*	50	tinggi_badan
13	+berat_badan	*Varchar*	50	berat_badan
14	+nama_ayah	*Varchar*	50	nama_ayah
15	+nama_ibu	*Varchar*	50	nama_ibu

2. Perancangan Tabel *Users*

Tabel data *Username* merupakan media untuk menampilkan hasil kriptografi, struktur dapat dilihat pada Tabel berikut;

Tabel III.10 Tabel Perancangan *user*

No	Nama	*Data Type*	*Size*	Keterangan
1.	Id	*Int*	11	Enkripsi/Deskripsi
2.	nama	*Varchar*	50	Enkripsi/Deskripsi
3.	username	*Varchar*	50	Enkripsi/Deskripsi
4.	password	*Varchar*	50	Enkripsi/Deskripsi

3. Perancangan Tabel *Blowfish*

Tabel *Blowfish* merupakan media untuk menampilkan hasil kriptografi, struktur dapat dilihat pada Tabel berikut;

Tabel III.11 Tabel *Blowfish*

No	Nama	*Data Type*	*Size*	Keterangan
1.	uuid	*Int*	11	Enkripsi/Deskripsi
2.	Is_enkripsi	*Varchar*	50	Enkripsi/Deskripsi
3.	password	*Varchar*	50	Enkripsi/Deskripsi

4. Perancangan Tabel *Password*

Tabel *Password* merupakan untuk mengambil *password* untuk enkripsi yang dilakukan admin dari hasil kriptografi, struktur dapat dilihat pada Tabel berikut:

Tabel III.12 Tabel *Password*

No	Nama	*Data Type*	*Size*	Keterangan
1.	uuid	*Int*	11	Enkripsi/Deskripsi
2.	password	*Varchar*	50	Enkripsi/Deskripsi

III.5. Desain *User Interface*

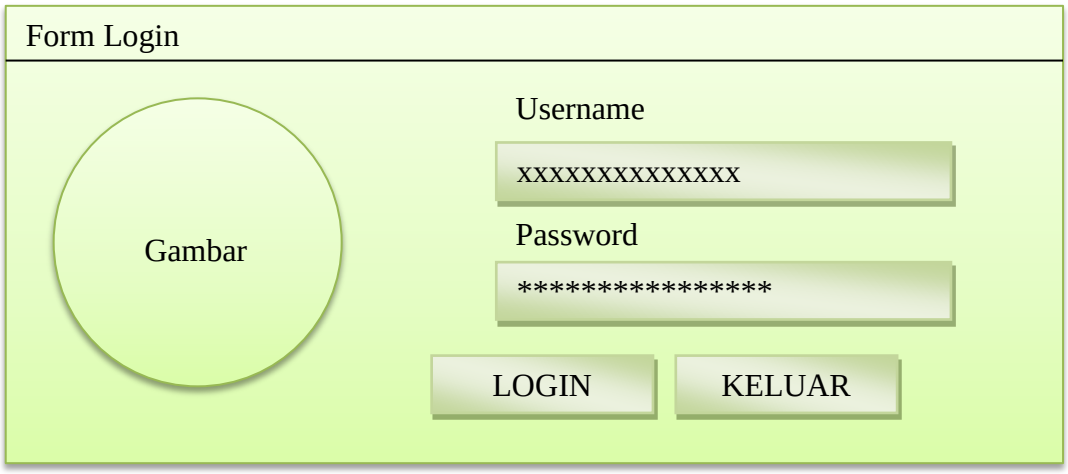
User Interface merupakan tampilan masukan yang penulis rancang guna lebih memudahkan dalam *entry* data. *Entry* data yang dirancang akan lebih mudah dan cepat dan meminimalisir kesalahan penulisan dan memudahkan perubahan. Perancangan tampilan yang dirancang adalah sebagai berikut :

III.5.1. Desain *User Interface* Bagian Administrasi

User Interface bagian administrasi pada Penerapan Algoritma *Blowfish* untuk Database Pendaftaran Siswa Baru adalah sebagai berikut :

1. Rancangan Halaman *Login*

Rancangan *Form Login* Untuk melakukan pengisian data awal *user*, rancangan terlihat pada Gambar di bawah ini:



Gambar III.15 Rancangan *Form Login* Admin

2. Rancangan Halaman *Dasboard*

Rancangan *Form* Admin sebagai penggabung *Form* dan Siswa, *Form* dan ada beberapa menu lainnya. Rancangan *Form* Data siswa ini dapat dilihat pada Gambar dibawah ini:

Gambar III.16 Rancangan *Form Dasboard*

3. Rancangan *Form* Data Siswa

Form Data Siswa adalah *Form* yang berguna untuk meng-*Input* data anggaran yaitu kode Siswa, nama siswa yang sesuai dengan yang dimiliki. Rancangan dapat dilihat pada Gambar di bawah ini:

Gambar III.17 Perancangan *Form* Data Siswa

4. Perancangan *Form* Proses *Blowfish*

Form ini berfungsi untuk meng *Input* data Siswa yang di enkrisipkan dan di deskripsikan. Dan mengenkripsikan data dengan menggunakan metode *Blowfish.* Rancangan dapat dilihat pada Gambar di bawah ini

Gambar III.18 Perancangan *Form* Proses *Blowfish*