BAB III

ANALISIS DAN PERANCANGAN

3.1 Analsis Sistem

Analisis adalah tahap aktifitas kreatif dimana analis berusaha memahami permasalahan secara mendalam. Ini adalah proses interative yang terus berjalan hingga permasalahan dapat dipahami dengan benar. Analisis bertujuan untuk mendapatkan pemahaman secara keseluruhan tentang sistem yang akan dibuat berdasarkan masukan dari pihak-pihak dan juga pengalaman analis yang berkepentingan dengan sistem tersebut.

3.1.1 Analisis Masalah

Permasalahan yang dibuat dalam tugas akhir ini adalah membuat simulasi algoritma GZip untuk mengompresi data, kemudian mengimplementasikannya pada proses pengiriman data. Hasil simulasi algoritma GZip yang telah dibuat ini kemudian dilakukan uji coba, sehingga kita bisa melihat hasil kompresi dan simulasi yang telah dibuat. Data dari hasil penelitian dengan algoritma GZip ini beserta pengimplementasiannya akan dibandingkan dengan *file* aslinya.

Dari perbandingan ini akan diperoleh kelebihan dan kekurangan dari algoritma ini, dan selanjutnya dari perbandingan tersebut akan diperoleh kesimpulan.

3.1.2 Analisis dan Kebutuhan Fungsional

3.1.2.1 Actor Identification

Tahap pertama yang dilakukan dalam melakukan analisis berorientasi objek menggunakan UML adalah menentukan actor atau pengguna sistem. Kata aktor dalam konteks UML, menampilkan peran (*roles*) yang pengguna (atau sesuatu di luar sistem yang dikembangkan yang dapat berupa perangkat keras, *end user*, sistem yang lain, dan sebagainya).

3.1.3 Analisis dan Kebutuhan Non-Fungsional

Analisis dan kebutuhan non-fungsional meliputi analisis dan kebutuhan pengguna, analisis dan kebutuhan perangkat keras, serta analisis dan kebutuhan perangkat lunak.

3.1.3.1 Analisis dan Kebutuhan Pengguna (*user*)

Pengguna diartikan sebagai orang yang mengakses dan menggunakan perangkat lunak aplikasi kompresi algoritma GZip, dalam hal ini pengguna harus memiliki kemampuan dasar untuk mengoperasikan komputer dan memiliki kapabilitas dalam menggunakan aplikasi komputer.

3.1.3.2 Analisis dan Kebutuhan Perangkat Keras

Perangkat keras adalah seluruh komponen atau unsur peralatan yang digunakan untuk menunjang pembangunan suatu aplikasi, dalam kasus ini adalah aplikasi kompresi menggunakan algoritma GZip.

Tabel 3.1 Tabel spesifikasi perangkat keras

Processor	***Memory (RAM)***	***Hard-disk***	***Monitor (Resolution)***
Intel Pentium 4	Kapasitas memory mulai dari 512 MB sampai 1 GByte	Kapasitas hard-disk mulai dari 40 GByte	Pada umumnya menggunakan monitor dengan resolusi 1024 x 768 (32 bit) 60 Hz

3.1.3.3 Analisis dan Kebutuhan Perangkat Lunak

1. Sistem operasi yang digunakan adalah Windows 7.
2. Perangkat lunak yang digunakan untuk membangun aplikasi ini adalah Eclipse 3.5.1.

3.1.4 Analisi Tehadap Sebuah *File* Teks

Analisis dilakukan dengan menggunakan *file* teks dengan isi ‘AAAAAAAAAAABCABCAAAAA’. Proses kompresi mula-mula dilakukan dengan metode LZ77 dimana pada metode ini isi *file* debaca sebagai sebuah *string* yang ditampung dalam variabel baru yaitu *Siliding Windows* (SW) dan *Read Ahead* (RA). SW merupakan *array* yang dapat menampung 10 *byte* dan RA dapat menampung 11 *byte*. SW kemudian diisi dengan 10 karakter pertama dari frase dan RA diisi dengan 11 karakter dimulai dari *index* terakhir SW ditambah 1. Karena SW dan RA merupakan *array* dengan tipe data *byte*, maka dibutuhkan *function* ‘ord’ untuk mengkonversikan nilai *char* ke bentuk *byte*. Terdapat sebuah variabel ‘hasil’ yang akan menyimpan hasil dari kompresi yang telah dilakukan. Pertama-tama, variabel hasil diisi dengan SW.

Frase ‘AAAAAAAAAAABCABCAAAAA’

Hasil ‘AAAAAAAAAA’

Setelah literasi awal dilakukan, kemudian dibandingkan apakah isi dari RA sama atau merupakan bagian dari SW, jika tidak maka *index* akhir sari RA akan dikurangi 1. Proses ini terus dilakukan jika isi dari RA sama dengan 2. Jika RA sama dengan 2 maka variabel hasil akan ditambahkan dengan isi dari RA dan *index* awal SW akan bergeser sebanyak 2 ke kanan.

Frase ‘AAAAAAAAAAABCABCAAAAA’

Hasil ‘AAAAAAAAAAAB’

Jika isi dari RA sama atau terdapat pada SW maka pada variabel hasil akan dicatat 2 buah kode yang terdiri dari *offset* dan *length*. Setelah itu *index* awal SW akan bergeser sebanyak isi dari RA.

Frase ‘AAAAAAAAAAABCABCAAAAA’

Hasil ‘AAAAAAAAAAABCA22’

Pada proses kompresi akan berhenti apanila *index* akhir dari SW sudah sama dengna panjang dari frase. Proses kemudian dilanjutkan dengan menambah *flag* pada setiap 8 *byte* pada variabel hasil. Hasil akhir dari proses kompresi dengan metode LZ77 ini ditampung dalam variabel ‘hasil2’.

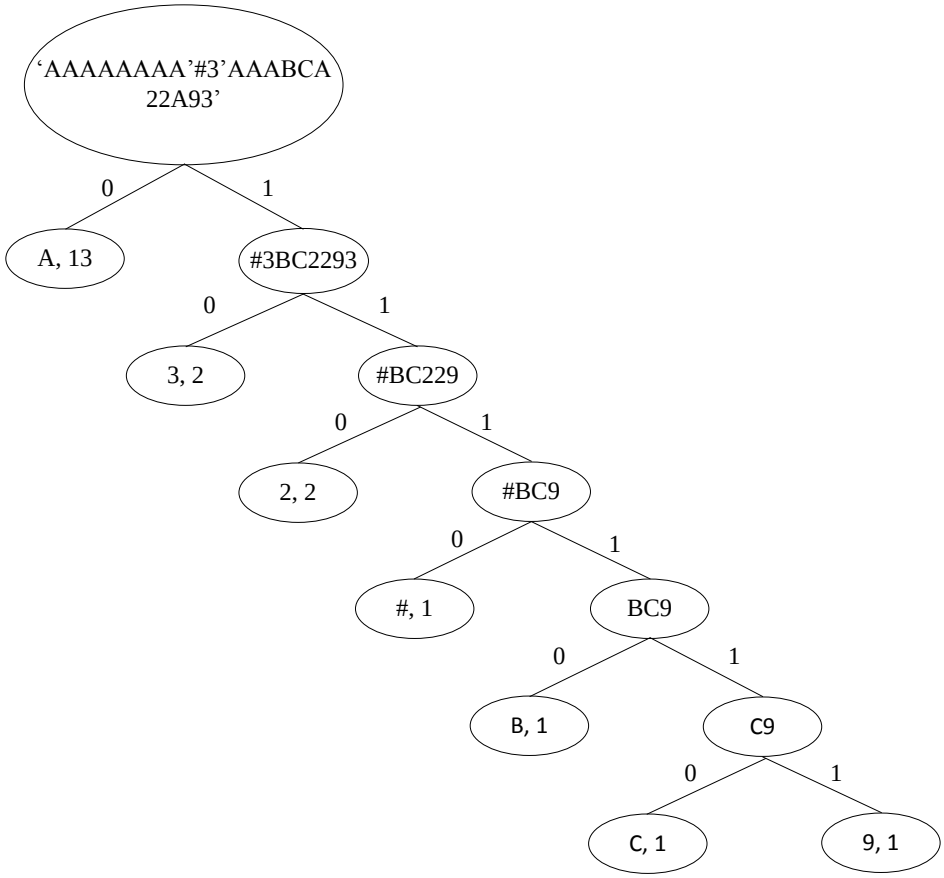
Frase ‘AAAAAAAAAAABCABCAAAAA’

Hasil ‘AAAAAAAAAAABCA2293’

Hasil2 ‘AAAAAAAA’#3’AAABCA22A93’

Isi dari variabel hasil2 tersebut kemusian dikompresi lagi dengan mentode Huffman. Langkah pertama dari proses kompresi dengna metode Huffman adalah

dengan membentuk sebuah *tree* yang berasal dari kumpulan *node-node*. Setiap *node* memiliki variabel data dan *value* dan juga memiliki 1 anak yaitu *left* dan *right*. Nilai dari variabel hasil2 akan dikonversikan menjadi bilangan ASCII(0-225). Nilai dari variabel *value* diperoleh dari jumlah kemunculan karakter tersebut.



Gambar 3.1 Pohon Huffman Proses Pertama

Adapun pada Huffman *tree* setiap *leaf* adalah karakter yang terdapat pada sebuah *file* yang akan dikompresi. Setiap mencapai suatu *leaf*, proses akan mencatat alur yang terjadi ke dalam senuah variabel ‘path’ dan kemudian akan disimpan ke dalam variabel *dictionary* dapat dilihat pada tabel 3.2

Tabel 3.2 Kode dictionary Huffman

Karakter	Frekuensi	Peluang	Kode Huffman
A	13	13/21	0
3	2	2/21	10
2	2	2/21	110
#	1	1/21	1110
B	1	1/21	11110
C	1	1/21	111110
9	1	1/21	1111110

Setelah dikompres dengan menggunakan Kode Huffman, string tersebut dapat direpresentasikan menjadi rangkaian bit:

010110111011110111101111110

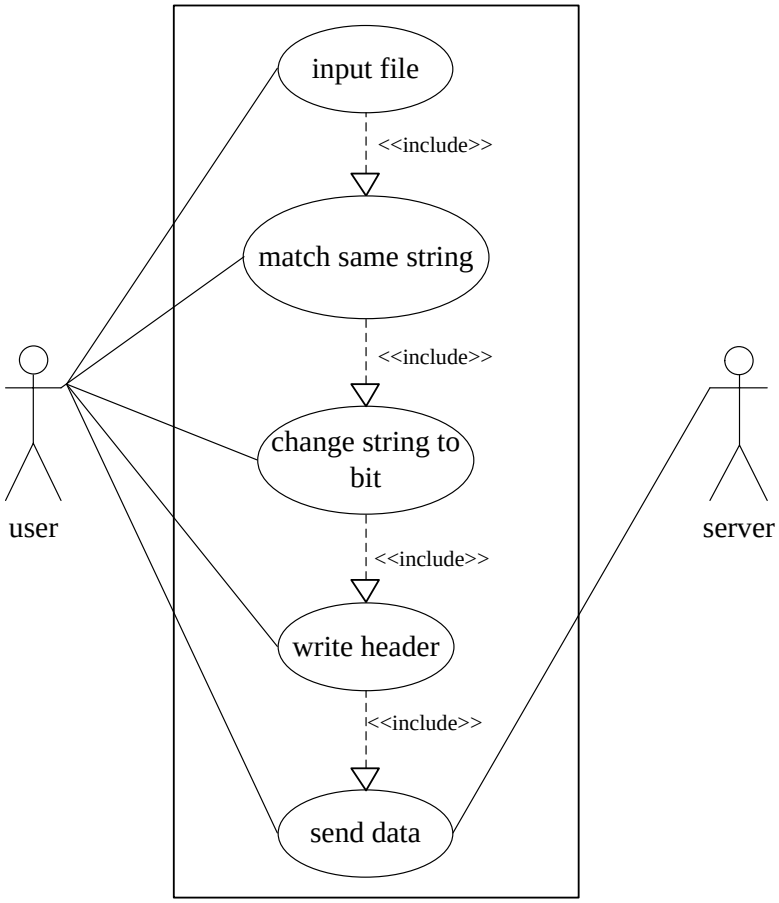
Setelah *dictionary* telah diperoleh, maka langkah terakhir yaitu menyalin *dictionary* ke *file output* kemudian mengkonveri setiap karakter yang terdapat pada *file input* sesuai dengan *dictionary* karakter tersebut.

3.2 Use Case Diagram

Use case diagram digunakan untuk menggambarkan fungsionalitas yang diharapkan dari sebuah sistem. Sebuah use case merepresentasikan sebuah interaksi antara aktor dengan sistem. Use case diagram pada Gambar 3.2 yang menggambarkan bagaimana proses yang terjadi pada aplikasi kompresi transfer data.

3.2.1 Use Case Diagram Kompresi

Use case diagram proses kompres dapat dilihat pada gambar 3.2



Gambar 3.2 Use case diagram proses kompresi file

Pada use case proses kompresi terjadi beberapa proses yaitu :

1. Mengambil *file* yang akan dikompres.
2. Membaca *file*.
3. Sistem mencari *string* yang sama pada *file*.
4. *String* yang sama akan dirubah kedalam bentuk *bit*.
5. Menulis hasil *bit* pada *header*.
6. Mengirimkan data ke *server*.

3.2.2 Skenario Use Case Kompresi

Skenario setiap bagian pada use case menunjukkan proses apa yang terjadi pada setiap bagian didalam use case tersebut, dimana user memberikan perintah pada setiap bagian dan respon apa yang diberikan oleh sistem kepada user setelah user memberikan perintah pada setiap bagian – bagian use case.

3.2.1.1 Server

Identifikasi		
Nomor	1	
Nama	Server	
Tujuan	Memastikan sebuah komputer menjadi server	
Deskripsi	Menerima *file* dari *client*/*user*	
Tipe		
Aktor	-	
Skenario Utama		
Kondisi awal	Aplikasi kompresi dalam keadaan baru dibuka	
Aksi Aktor		**Reaksi Sistem**
-		Server menerima dan meyimpan *file* yang telah dikirimkan oleh c*lient*
-		Server mengirim file yang diminta oleh *client*
Skenario Alternatif – Autentikasi Gagal		
Aksi Aktor		**Reaksi Sistem**
-		-
Kondisi akhir	Sistem menampilan status proses	

3.2.1.2 User

Identifikasi	
Nomor	2
Nama	*User/client*
Tujuan	Mengirimkan file kepada Server
Deskripsi	Pada saat aplikasi kompresi dibuka *user* dapat mengirimkan *file* kepada server.
Tipe	

Aktor	User	
Skenario Utama		
Kondisi awal	Server Aktif	
Aksi Aktor		**Reaksi Sistem**
Memasukkan *file* yang akan dikirimkan kepada server.		-
		Memberikan informasi status pengiriman pada *user*.
Skenario Alternatif – Autentikasi Gagal		
Aksi Aktor		**Reaksi Sistem**
Memasukkan file yang akan dikirimkan kepada server jika server sedang tidak aktif..		Memberikan informasi bahwa pengiriman gagal.
Kondisi akhir	1. File yang dikirimkan akan berhasil jika server aktif. 2. File yang dikirmkan akan gagal jika server tidak aktif.	

3.2.1.3 Input File

Identifikasi		
Nomor	3	
Nama	Input File	
Tujuan	User menentukan *file* yang akan dikompresi	
Deskripsi	Sistem menerima *file* yang dipilih user dan melakukan kompresi	
Tipe		
Aktor	User	
Skenario Utama		
Kondisi awal	Aplikasi kompresi dalam keadaan baru dibuka	
Aksi Aktor		**Reaksi Sistem**
Memilih dan menginputkan *file* yang akan dikompresi.		-
Skenario Alternatif – Autentikasi Gagal		
Aksi Aktor		**Reaksi Sistem**
Menginputkan *file* yang akan dikompresi		-
Kondisi akhir	User telah memilih file yang akan dikompresi	

3.2.1.4 Match Same String

Identifikasi		
Nomor	4	
Nama	Match Same String	
Tujuan	Mencari s*tring* yang sama pada *file.*	
Deskripsi	Sistem telah menerima *file* yang dipilih user	
Tipe		
Aktor		
Skenario Utama		
Kondisi awal	1. Aplikasi kompresi dalam keadaan dibuka 2. Terdapat file yang akan dikompresi	
Aksi Aktor		**Reaksi Sistem**
		Mencari *string* yang sama pada *file* yang akan dikompresi
Skenario Alternatif – Autentikasi Gagal		
Aksi Aktor		**Reaksi Sistem**
Kondisi akhir	Seluruh *string* yang sama pada *file* telah ditemukan	

3.2.1.5 Change String to Bit

Identifikasi		
Nomor	5	
Nama	Change String to Bit	
Tujuan	Merubah s*tring* yang sama pada *file* kedalam bit	
Deskripsi	Sistem telah mencari *string* yang sama	
Tipe		
Aktor		
Skenario Utama		
Kondisi awal	Aplikasi kompresi dalam keadaan dibuka Terdapat file yang akan dikompresi	
Aksi Aktor		**Reaksi Sistem**
		Merubah *string* yang sama kedalam bit, dengan membentuk *tree*.
Skenario Alternatif – Autentikasi Gagal		
Aksi Aktor		**Reaksi Sistem**
Kondisi akhir	Seluruh *string* telah di *encode* kedalam bit	

3.2.1.6 Write Header

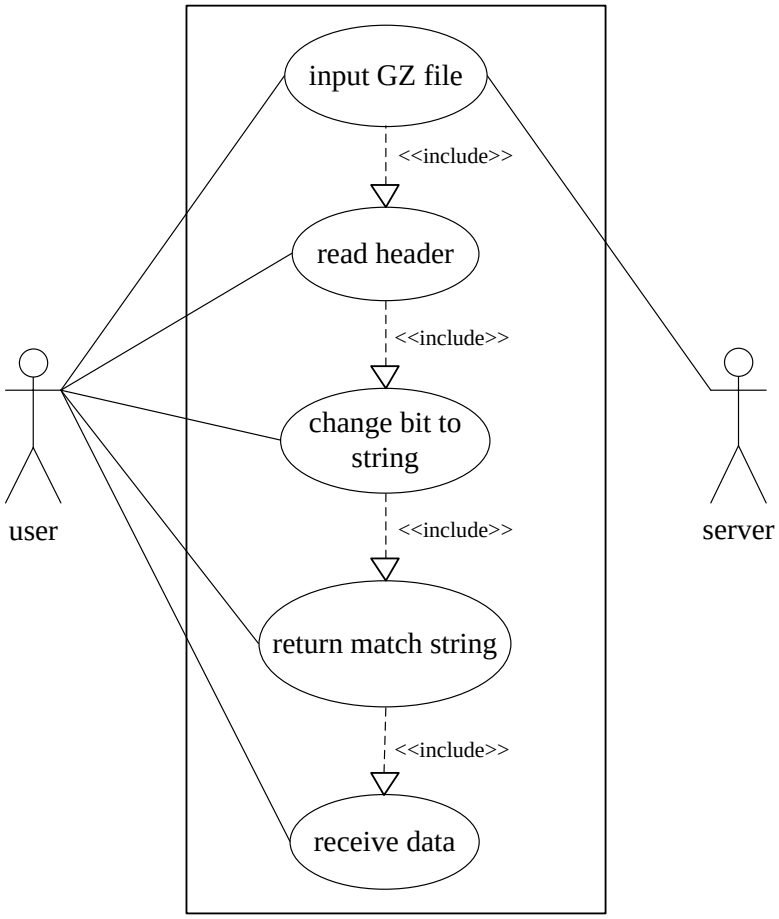
Identifikasi		
Nomor	6	
Nama	Write Header	
Tujuan	Membentuk *dictionary*	
Deskripsi	Sistem akan memberikan *header file*	
Tipe		
Aktor		
Skenario Utama		
Kondisi awal	Aplikasi kompresi dalam keadaan dibuka Terdapat file yang akan dikompresi	
Aksi Aktor		**Reaksi Sistem**
		Merubah *string* yang sama kedalam bentuk biner, dengan membentuk *tree*.
Skenario Alternatif – Autentikasi Gagal		
Aksi Aktor		**Reaksi Sistem**
Kondisi akhir	Membentuk *dictionary file* pada *header*	

3.2.1.7 Send File

Identifikasi		
Nomor	7	
Nama	Send file	
Tujuan	Mengirim *file* yang telah dikompres	
Deskripsi	Sistem akan mengirim *file* yang telah dikompres	
Tipe		
Aktor		
Skenario Utama		
Kondisi awal		
Aksi Aktor		**Reaksi Sistem**
		Sistem mengirimkan *file* yang telah dikompres kepada server
Skenario Alternatif – Autentikasi Gagal		
Aksi Aktor		**Reaksi Sistem**
Kondisi akhir	File asli telah terkompres dan tersimpan pada server.	

3.2.3 Use Case Diagram Dekompresi

Use case diagram proses kompres dapat dilihat pada gambar 3.3



Gambar 3.3 Use case diagram proses dekompresi file

Pada use case proses dekompresi terjadi beberapa proses yaitu :

1. Mengambil *file* yang akan didekompres.
2. Membaca *header file*.
3. *Bit* dirubah kedalam bentuk *string.*
4. Sistem mengembalikan posisi *string* yang sama pada *file*.
5. Menerima data dari *server*.

3.2.4 Skenario Use Case Dekompresi

3.2.4.1 Server

Identifikasi		
Nomor	1	
Nama	Server	
Tujuan	Memastikan sebuah komputer menjadi server	
Deskripsi	Menerima *file* dari *client*/*user*	
Tipe		
Aktor	-	
Skenario Utama		
Kondisi awal	Aplikasi kompresi dalam keadaan baru dibuka	
Aksi Aktor		**Reaksi Sistem**
-		Server menerima dan meyimpan *file* yang telah dikirimkan oleh c*lient*
-		Server mengirim file yang diminta oleh *client*
Skenario Alternatif – Autentikasi Gagal		
Aksi Aktor		**Reaksi Sistem**
-		-
Kondisi akhir	Sistem menampilan status proses	

3.2.4.2 User

Identifikasi		
Nomor	2	
Nama	*User/client*	
Tujuan	Menerima file dari Server	
Deskripsi	Pada saat aplikasi dekompresi dibuka *user* dapat menerima file dari server.	
Tipe		
Aktor	User	
Skenario Utama		
Kondisi awal	Server Aktif	
Aksi Aktor		**Reaksi Sistem**
Memilih *file* yang akan didekompres dari server.		-

	Memberikan informasi status pengiriman pada *user*.
Skenario Alternatif – Autentikasi Gagal	
Aksi Aktor	**Reaksi Sistem**
Kondisi akhir	*File* telah dipilih jika server dalam keadaan aktif

3.2.4.3 Input GZ File

Identifikasi	
Nomor	3
Nama	Input GZ File
Tujuan	User menentukan *file* GZ yang akan didekompresi
Deskripsi	Sistem menerima *file GZ* yang dipilih user dan melakukan dekompresi
Tipe	
Aktor	User
Skenario Utama	
Kondisi awal	Aplikasi kompresi dalam keadaan baru dibuka
Aksi Aktor	**Reaksi Sistem**
Memilih *file* yang akan didekompresi dari server.	-
Skenario Alternatif – Autentikasi Gagal	
Aksi Aktor	**Reaksi Sistem**
Memilih *file* yang akan didekompresi	-
Kondisi akhir	User telah memilih file yang akan didekompresi

3.2.4.4 Read Header

Identifikasi	
Nomor	4
Nama	Read Header
Tujuan	Membaca *dictionary*
Deskripsi	Sistem akan membaca *header file*
Tipe	
Aktor	
Skenario Utama	
Kondisi awal	Aplikasi kompresi dalam keadaan dibuka Terdapat file yang akan didekompresi

Aksi Aktor	Reaksi Sistem
	Membaca alur *tree*
Skenario Alternatif – Autentikasi Gagal	
Aksi Aktor	**Reaksi Sistem**
Kondisi akhir	Membaca *dictionary file* pada *header*

3.2.4.5 Change Bit to String

Identifikasi	
Nomor	5
Nama	Change Bit to String
Tujuan	Merubah *bit* menjadi *string*
Deskripsi	Sistem telah membaca *dictionary* dan menelusuri *tree*
Tipe	
Aktor	
Skenario Utama	
Kondisi awal	Aplikasi kompresi dalam keadaan dibuka Terdapat file yang akan dikompresi
Aksi Aktor	**Reaksi Sistem**
	Menelsuri *tree* dan merubah *bit* kedalam *string*.
Skenario Alternatif – Autentikasi Gagal	
Aksi Aktor	**Reaksi Sistem**
Kondisi akhir	Seluruh *bit* telah di *dencode* kedalam *string*

3.2.4.6 Return Match Same String

Identifikasi	
Nomor	6
Nama	Return Match String
Tujuan	Mengembalikan posisi *string* yang sama
Deskripsi	Sistem telah menerima hasil *dencoding*
Tipe	
Aktor	
Skenario Utama	
Kondisi awal	1. Aplikasi kompresi dalam keadaan dibuka 2. Terdapat file yang akan dikompresi

Aksi Aktor	Reaksi Sistem
	Mengembalikan posisi *string* yang sama pada *file*
Skenario Alternatif – Autentikasi Gagal	
Aksi Aktor	**Reaksi Sistem**
Kondisi akhir	Seluruh *string* yang sama pada *file* telah kembali pada posisi semula.

3.2.4.7 Receive File

Identifikasi	
Nomor	7
Nama	Receive file
Tujuan	Menerima *file* yang telah didekompres
Deskripsi	Sistem akan mengirim *file* yang telah didekompres
Tipe	
Aktor	
Skenario Utama	
Kondisi awal	
Aksi Aktor	**Reaksi Sistem**
	Sistem mengirimkan *file* yang telah didekompres kepada *user*
Skenario Alternatif – Autentikasi Gagal	
Aksi Aktor	**Reaksi Sistem**
Kondisi akhir	File asli telah terdekompres dan tersimpan pada *user*.

3.3 Sequence Diagram

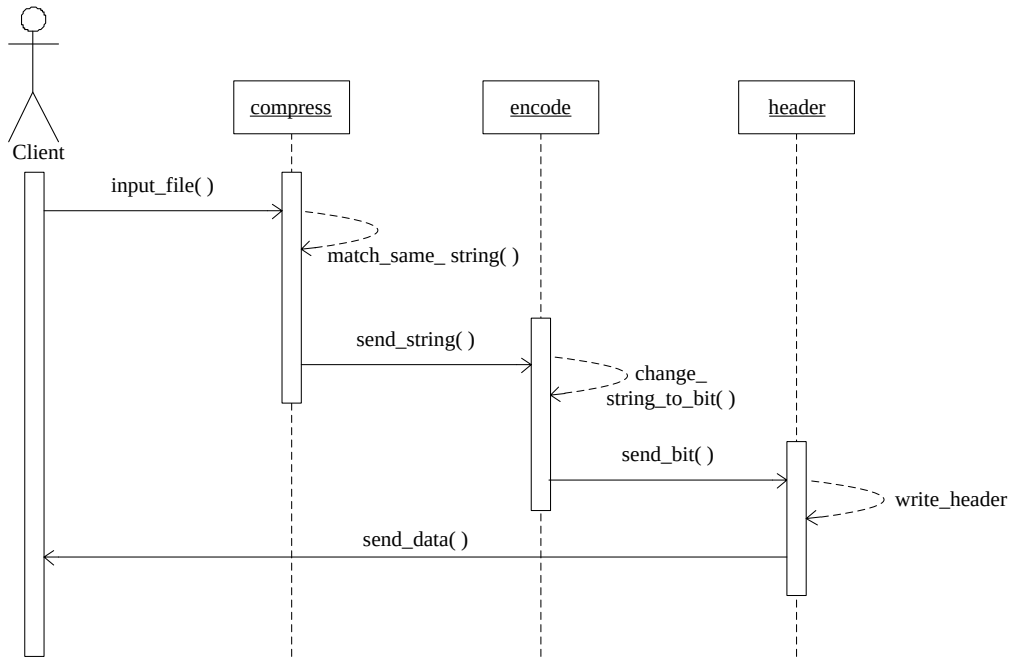
Data Sequence diagram menggambarkan interaksi antar objek di dalam dan di sekitar sistem (termasuk pengguna, *display*, dan sebagainya) berupa *message* yang digambarkan terhadap waktu. *Sequence diagram* terdiri atar dimensi vertikal (waktu) dan dimensi horizontal (objek-objek yang terkait).

Sequence diagram biasa digunakan untuk menggambarkan skenario atau rangkaian langkah-langkah yang dilakukan sebagai respons dari sebuah *event* untuk menghasilkan *output* tertentu. Diawali dariapa yang men-*trigger* aktivitas tersebut, proses dan perubahan apa saja yang terjadi secara internal dan *output* apa yang dihasilkan.

Sequence Diagram yang digambarkan dalam perancangan sistem dapat dilihat pada gambar 3.4 dan gambar 3.5.

3.3.1 Sequence Diagram Kompresi

Sequence diagram proses kompres dapat dilihat pada gambar 3.4



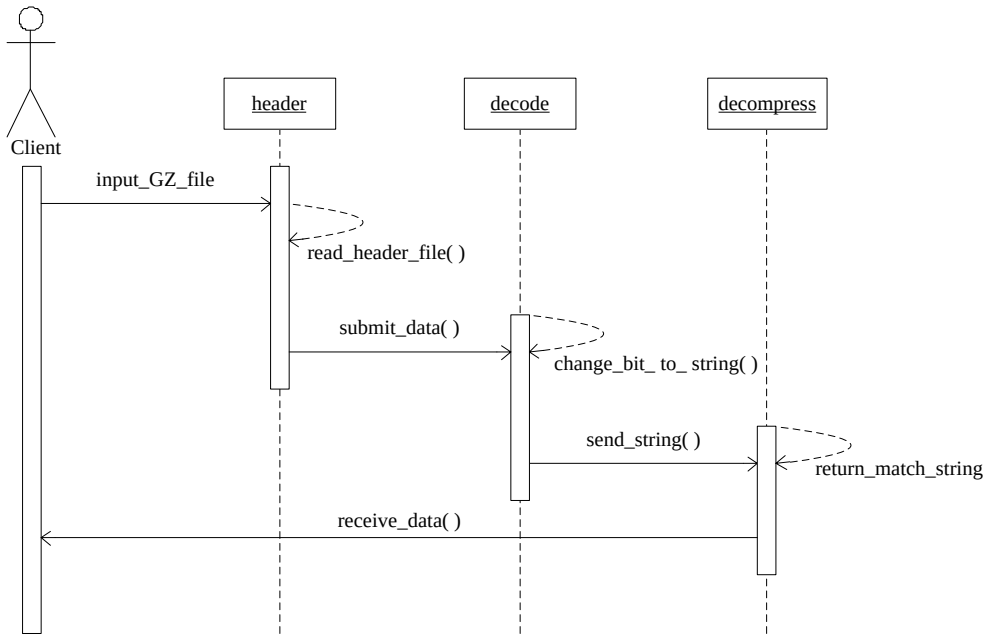
Gambar 3.4 Sequence diagram proses kompresi

Pada sequence diagram di atas terjadi beberapa proses yaitu :

1. *User* mengambil *file* yang akan di kompres.
2. Sistem membaca *file*.
3. Sistem mencari *string* yang sama.
4. Sistem mengirim *string* untuk di *encode*.
5. Sistem melakukan perubahan dari *string* ke *bit*.
6. Sistem mengirim data *bit*.
7. Sistem menulis *header*.
8. Sistem mengirim data.

3.3.2 Sequence Diagram Dekompresi

Sequence diagram proses kompres dapat dilihat pada gambar 3.5



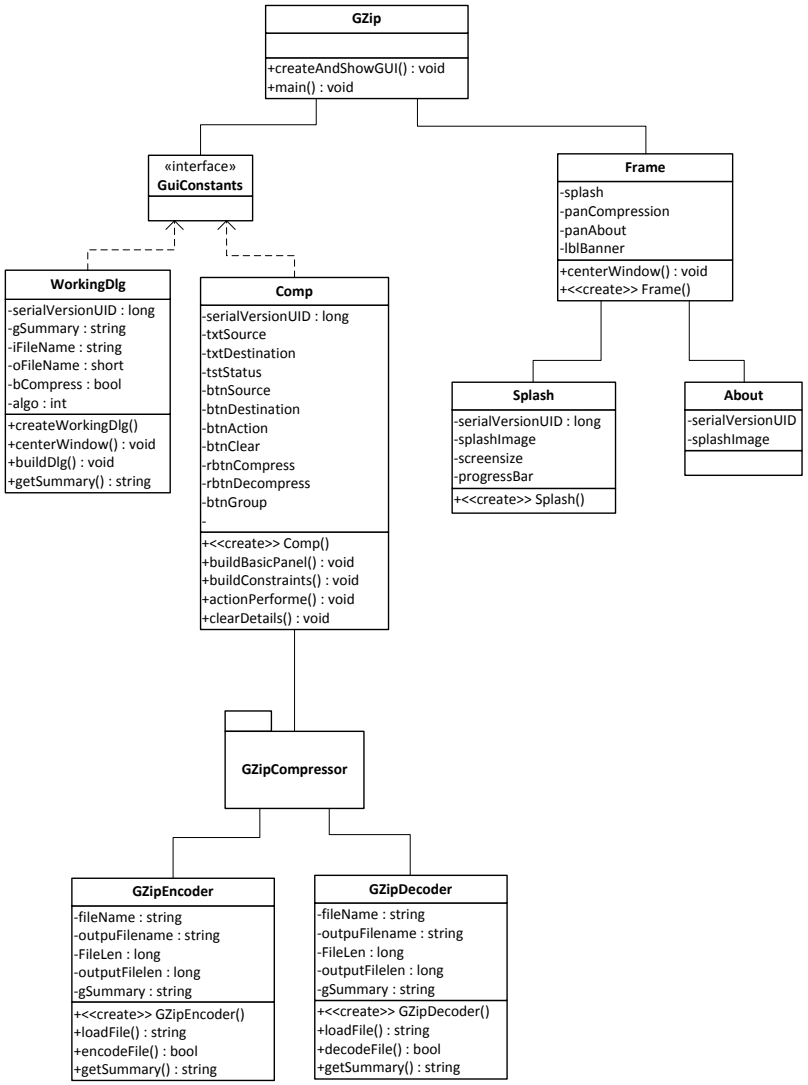
Gambar 3.5 Sequence diagram proses dekompresi

Pada sequence diagram di atas terjadi beberapa proses yaitu :

1. *User* mengambil *file* yang akan di dekompres.
2. Sistem membaca *header file*.
3. Sistem melakukan perubahan dari *bit* ke *string*.
4. Sistem mengirim *string* untuk di dekompres.
5. Sistem mengembalikan posisi *string* yang sama.
6. Sistem mengirim data *string*.
7. Sistem mengirim data.

3.4 Class Diagram

Class Diagram menggambarkan keadaan suatu sistem (*atribut*), dan memberikan pelayanan untuk menyelesaikan keadaan tersebut (metoda). Class diagram dapat dilihat pada gambar 3.6.



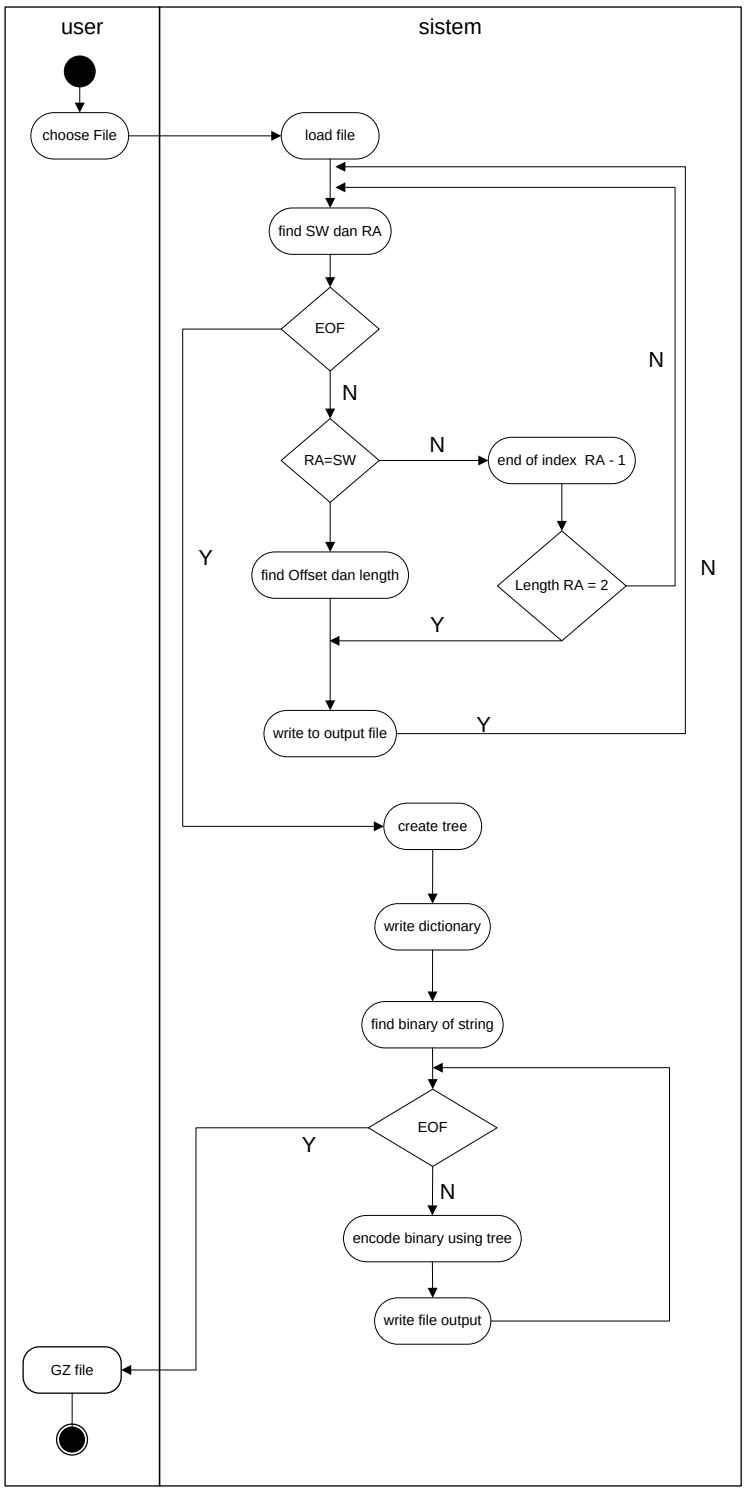
Gambar 3.6 Class diagram

3.5 Activity Diagram

Activity diagram menggambarkan berbagai alir aktivitas dalam sistem yang sedang dirancang, bagaimana masing-masing alir berawal, *decision* yang mungkin terjadi, dan bagaimana mereka berakhir. *Activity diagram* juga dapat menggambarkan proses paralel yang mungkin terjadi pada beberapa eksekusi. *Activity diagram* merupakan *state diagram* khusus, di mana sebagian besar *state* adalah *action* dan sebagian besar transisi di-*trigger* oleh selesainya *state* sebelumnya (*internal processing*). Oleh karena itu *activity diagram* tidak menggambarkan behaviour internal sebuah sistem (dan interaksi antar subsistem) secara eksak, tetapi lebih menggambarkan proses-proses dan jalur-jalur aktivitas dari level atas secara umum. Sebuah aktivitas dapat direalisasikan oleh satu *use case* atau lebih. Aktivitas menggambarkan proses yang berjalan, sementara *use case* menggambarkan bagaimana aktor menggunakan sistem untuk melakukan aktivitas. Gambar 3.7 dan gambar 3.8 menunjukan activity diagram yang dilakukan aplikasi.

3.5.1 Activity Diagram Kompresi

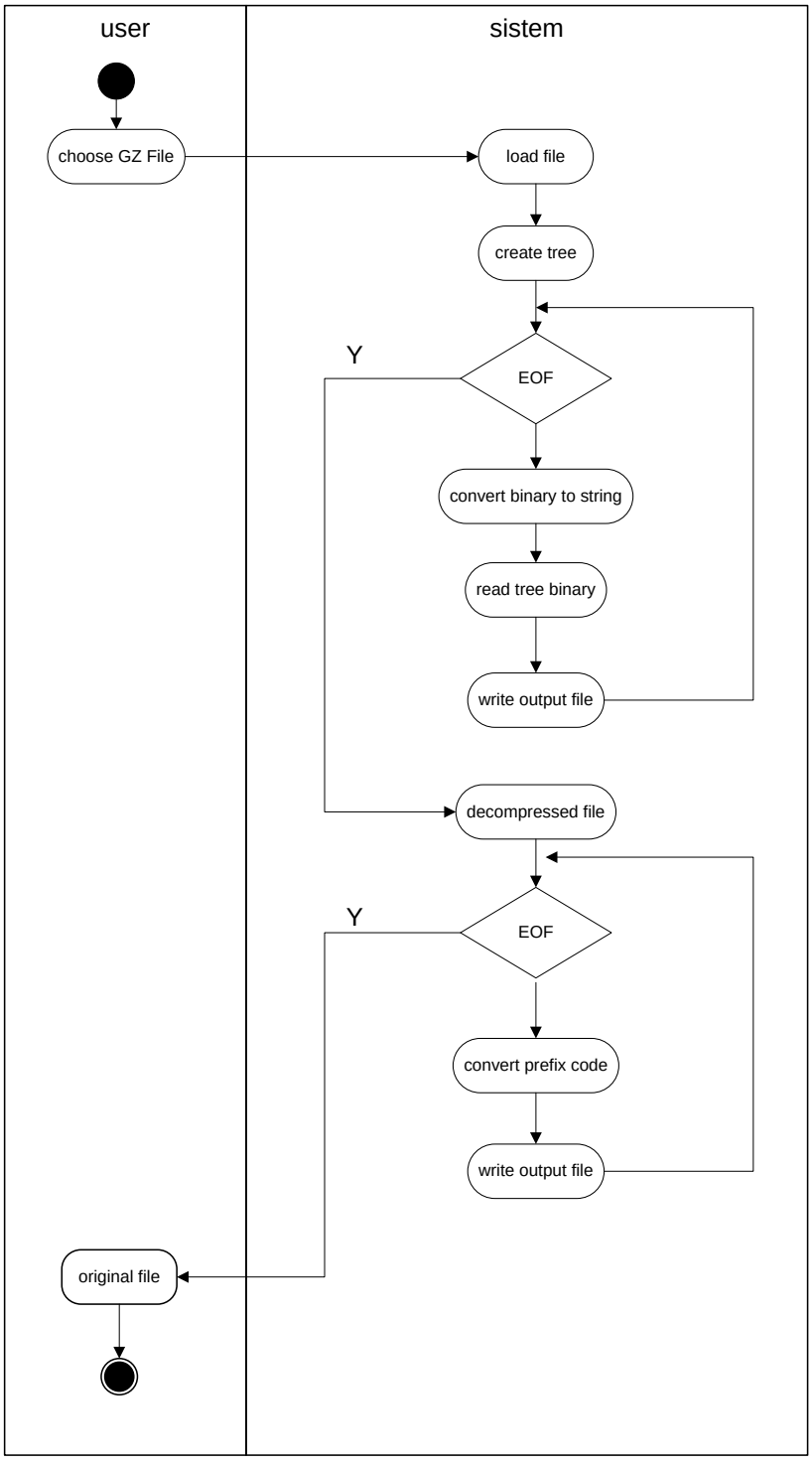
Activity diagram proses kompres dapat dilihat pada gambar 3.7



Gambar 3.7 Activity diagram proses kompresi

3.5.2 Activity Diagram Dekompresi

Activity diagram proses kompres dapat dilihat pada gambar 3.8



Gambar 3.8 Activity diagram proses dekompresi

3.6 Perancangan Sistem

Perancangan sistem adalah suatu proses yang menggambarkan bagaimana suatu sistem dibangun untuk memenuhi kebutuhan pada fase analisis. Adapun tahapan yang dilakukan dalam perancangan sistem ini membahas mengenai tujuan perancangan sistem, dan perancangan antar muka.

3.6.1 Tujuan Perancangan Sistem

Perancangan sistem merupakan tindak lanjut dari tahap analisa. Perancangan sistem bertujuan untuk memberikan gambaran sistem yang akan dibuat. Dengan kata lain perancanagn sistem didefinisikan sebagai penggambaran atau pembuatan sketsa dari beberapa elemen yang terpisah kedalam satu kesatuan yang utuh dan berfungsi. Selain itu juga perancangan bertujuan untuk lebih mengarahkan sistem yang terinci, yaitu pembuatan perancangan yang jelas dan lengkap yang nantinya akan digunakan untuk pembuatan simulasi. Aplikasi kompresi ini dibuat dengan sederhana, sehingga diharapkan user dapat dengan mudah menggunakan aplikasi kompresi ini.

3.6.2 Perancangan Antar Muka

Perancangan antarmuka dibutuhkan untuk mewakili keadaan sebenarnya dari aplikasi yang akan dibangun, berikut akan disajikan perancangan antarmuka dari aplikasi yang akan dibangun:

3.6.2.1 Antar Muka Menu Compress

Antar muka menu compress digunakan untuk melakukan proses mengkompres file. Gambar 3.9 menampilkan rancangan menu compress.



Gambar 3.9 Rancangan antar muka menu Compress

Tabel 3.3. Keterangan tampilan menu compress

No Objek	Jenis Objek	Fungsi Objek
1	Compressor/Decompressor	Untuk masuk kehalaman kompres / dekompres
2	About	Untuk menampilkan halaman tentang *software* dan pembuat
3	Input	Untuk mencantumkan tulisan Input File
4	Output	Untuk mencantumkan tulisan Output File
5	Procedure	Untuk memilih proses yang akan dilakukan (kompresi/dekompresi)
6	Status bar	Untuk mencantumkan informmasi proses yang telah dilakuakan
7.	Compress	Untuk melakukan proses kompresi file
8	Clear	Untuk menghapus informasi yang tercamtum pada status bar

3.6.2.1 Antar Muka Menu Decompress

Antar muka menu decompress digunakan untuk melakukan proses dekompresi file. Gambar 3.10 menampilkan rancangan menu decompress.



Gambar 3.10 Rancangan antar muka menu Decompress

Tabel 3.4. Keterangan tampilan menu Decompress

No Objek	Jenis Objek	Fungsi Objek
1	Compressor/Decompressor	Untuk masuk kehalaman kompres / dekompres
2	About	Untuk menampilkan halaman tentang *software* dan pembuat
3	Input	Untuk mencantumkan tulisan Input File
4	Output	Untuk mencantumkan tulisan Output File
5	Procedure	Untuk memilih proses yang akan dilakukan (kompresi/dekompresi)
6	Status bar	Untuk mencantumkan informmasi proses yang telah dilakuakan

7.	Decompres	Untuk melakukan proses kompresi file
8	Clear	Untuk menghapus informasi yang tercamtum pada status bar

3.6.2.3 Antar Muka Menu About

Antar muka menu about digunakan untuk menampilkan informasi mengenai *software* dan juga pembangun *software*. Gambar 3.11 menampilkan rancangan menu about.



Gambar 3.11 Rancangan antar muka menu About