

# LAPORAN TUGAS BESAR

## IF2111 Algoritma dan Struktur Data STI


### WayangWave

Dipersiapkan oleh:

Kelompok 10

Muhammad Rifa Ansyari	(18222004)
Irfan Musthofa	(18222056)
Taufiq Ramadhan Ahmad	(18222060)
Kerlyn Deslia Andeskar	(18222090)
Farah Aulia	(18222096)

Sekolah Teknik Elektro dan Informatika - Institut Teknologi Bandung  
Jl. Ganesha 10, Bandung 40132

	Sekolah Teknik Elektro dan Informatika ITB	Nomor Dokumen		Halaman
		IF2111-TB-02-10		<jml hlm>
		Revisi	0	25 Oktober 2023

# Daftar Isi

1	Ringkasan	4
2	Penjelasan Tambahan Spesifikasi Tugas	5
2.1	Enhance (Bonus)	5
2.2	Fitur Interface	5
3	Struktur Data (ADT)	6
3.1	List	6
3.1.1	Sketsa Struktur Data	6
3.1.2	Persoalan yang Dapat Diselesaikan	7
3.1.3	Alasan Pemilihan ADT	7
3.1.4	Implementasi Sebagai ADT	7
3.2	Playlist (Array Dinamis dan Linked List)	7
3.2.1	Sketsa Struktur Data	7
3.2.2	Persoalan yang Dapat Diselesaikan	9
3.2.3	Alasan Pemilihan ADT	9
3.2.4	Implementasi Sebagai ADT	9
3.3	Mesin Kalimat (Mesin Kata dan Mesin Karakter)	9
3.3.1	Mesin Kata	9
3.3.1.1	Sketsa Struktur Data	9
3.3.1.2	Persoalan yang Dapat Diselesaikan	10
3.3.1.3	Alasan Pemilihan ADT	10
3.3.1.4	Implementasi Sebagai ADT	10
3.3.2	Mesin Karakter	10
3.3.2.1	Sketsa Struktur Data	10
3.3.2.2	Persoalan yang Dapat Diselesaikan	11
3.3.2.3	Alasan Pemilihan ADT	11
3.3.2.4	Implementasi Sebagai ADT	11
3.4	Queue	11
3.4.1	Sketsa Struktur Data	11
3.4.2	Persoalan yang Dapat Diselesaikan	11
3.4.3	Alasan Pemilihan ADT	11
3.4.4	Implementasi Sebagai ADT	11
3.5	Stack	12
3.5.1	Sketsa Struktur Data	12
3.5.2	Persoalan yang Dapat Diselesaikan	12

3.5.3 Alasan Pemilihan ADT	12
3.5.4 Implementasi Sebagai ADT	12
3.6 Set	12
3.6.1 Sketsa Struktur Data	12
3.6.2 Persoalan yang Dapat Diselesaikan	13
3.6.3 Alasan Pemilihan ADT	13
3.6.4 Implementasi Sebagai ADT	13
3.7 Map	13
3.7.1 Sketsa Struktur Data	13
3.7.2 Persoalan yang Dapat Diselesaikan	14
3.7.3 Alasan Pemilihan ADT	14
3.7.4 Implementasi Sebagai ADT	14
4 Program Utama	14
5 Algoritma-Algoritma Menarik	21
5.1 Random Angka di Enhance	21
6 Data Test	21
6.1 START	21
6.2 LOAD <filename>	22
6.3 LIST	23
6.3.1 LIST DEFAULT	23
6.3.2 LIST PLAYLIST	24
6.4 PLAY	24
6.4.1 PLAY SONG	24
6.4.2 PLAY PLAYLIST	25
6.5 QUEUE	26
6.5.1 QUEUE SONG	26
6.5.2 QUEUE PLAYLIST	27
6.5.3 QUEUE SWAP <x> <y>	27
6.5.4 QUEUE REMOVE <id>	28
6.5.5 QUEUE CLEAR	28
6.6 SONG	28
6.6.1 SONG NEXT	28
6.6.2 SONG PREVIOUS	29
6.7 PLAYLIST	29
6.7.1 PLAYLIST CREATE	29
6.7.2 PLAYLIST ADD	30
6.7.2.1 PLAYLIST ADD SONG	30

6.7.2.2 PLAYLIST ADD ALBUM	31
6.7.3 PLAYLIST SWAP <id> <x> <y>	32
6.7.4 PLAYLIST REMOVE <id> <n>	32
6.7.5 PLAYLIST DELETE	32
6.8 STATUS	33
6.9 SAVE <filename>	34
6.10 QUIT	34
6.11 HELP	35
6.12 INVALID COMMAND	36
6.13 ENHANCE	36
7 Test Script	37
8 Pembagian Kerja dalam Kelompok	41
9 Lampiran	42
9.1 Deskripsi Tugas Besar	42
9.2 Notulen Rapat	44
9.3 Log Activity Anggota Kelompok	48

# 1 Ringkasan

Aplikasi WayangWave dibuat untuk membantu Bondowoso agar dapat meluluhkan hati Roro yang hobi mendengarkan lagu hip-hop terbaru dari *walkman*-nya, kami selaku kumpulan makhluk kasat mata pun merancang program WayangWave tersebut. WayangWave merupakan sebuah program aplikasi berbasis *command-line interface* yang dibuat dengan menggunakan bahasa pemrograman C, yang memungkinkan pengguna untuk merasakan pengalaman layanan pemutaran musik. Program ini memiliki lima fungsi atau fitur utama, seperti memutar lagu, menampilkan daftar lagu, membuat dan menghapus *playlist*, mengatur penyusunan urutan pemutaran lagu, dan menayangkan status dari aplikasi. Selain itu, terdapat pula tambahan fitur berupa menampilkan daftar *playlist* dengan tambahan lagu rekomendasi *random* dari penyanyi *random* dan album *random*.

Laporan ini berisi penjelasan mengenai program WayangWave yang telah dibuat, yaitu spesifikasi program, mencakup tipe data bentukan atau struktur data (ADT) yang digunakan dalam pembuatan program ini, algoritma program utama, algoritma-algoritma menarik yang ditemukan, data tes program, dan *test script*. Pada program ini tentunya terdapat program utama atau *main program* yang digunakan sebagai wadah sinkronisasi antar program *command* sehingga program WayangWave ini dapat dijalankan dengan baik. Dalam laporan ini juga terlampir pembagian kerja setiap anggota kelompok, notulen rapat, deskripsi tugas besar, dan *log activity* anggota kelompok.

Dalam program WayangWave yang kami buat ini, tersedia *command* START, LOAD, LIST DEFAULT, LIST PLAYLIST, PLAY SONG, PLAY PLAYLIST, QUEUE SONG, QUEUE PLAYLIST, QUEUE SWAP, QUEUE REMOVE, QUEUE CLEAR, SONG NEXT, SONG PREVIOUS, PLAYLIST CREATE, PLAYLIST ADD SONG, PLAYLIST ADD ALBUM, PLAYLIST SWAP, PLAYLIST REMOVE, PLAYLIST DELETE, STATUS, SAVE, QUIT, HELP, <INVALID COMMAND>, dan ENHANCE. Masing-masing penjelasan dan hasil dari pemanggilan *command* tersebut akan dijelaskan lebih lanjut dalam laporan ini.

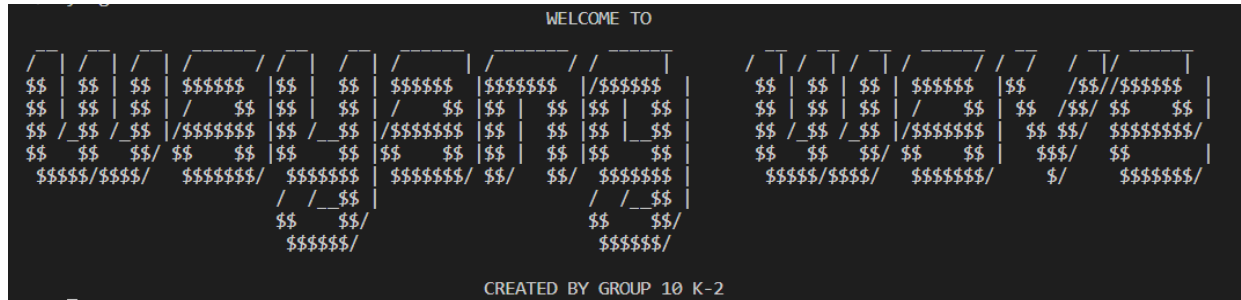
## 2 Penjelasan Tambahan Spesifikasi Tugas

### 2.1 Enhance (Bonus)

Sebagai tambahan, terdapat fitur baru yang diperkenalkan dengan *command* ENHANCE. Fitur ENHANCE meminta pengguna untuk memilih *playlist* yang ingin ditambahkan dengan lagu-lagu *random*. Setelah itu, akan ditampilkan lagu-lagu *random* yang telah dimasukkan ke dalam *playlist*. Lagu yang ditambahkan maksimal sebanyak lagu di dalam *playlist* sehingga apabila *playlist* kosong tidak ada lagu yang direkomendasikan atau ditambahkan. Lagu-lagu yang ditambahkan belum ada di *playlist*.

### 2.2 Fitur Interface

Fitur Interface merupakan fitur tambahan yang kami buat dengan tujuan untuk memperindah tampilan dari program WayangWave sehingga lebih menarik untuk digunakan oleh pengguna. Berikut merupakan bentuk *interface* awal program WayangWave yang kami buat:



Gambar 2.2.1

## 3 Struktur Data (ADT)

### 3.1 List

#### 3.1.1 Sketsa Struktur Data

```

/* Kamus Umum */
#define MaxEl 20
#define NMAX 20

/* Definisi elemen dan koleksi objek */
#define IdxType int
#define ElType char*

typedef struct {
    ElType A[MaxEl];
    int Neff;
} ListStatic;

typedef struct{
    Eltype A[NMAX];
    int neff;
} ListPenyanyi;

```

Pada ADT ini, terdapat dua tipe list yaitu list penyanyi yang digunakan untuk menyimpan penyanyi dari *file* config dan *list* statis biasa. Keduanya bertipe *list*. Primitif hanya digunakan pada *list* statis. Berikut primitif yang digunakan dalam ADT ini:

1. Konstruktor:
  - a. ListStatic MakeListStatis();
2. Prototipe:
  - a. boolean IsEmptyListStatis(ListStatic L);
  - b. int LengthListStatis(ListStatic L);
3. Operasi-Operasi:
  - a. void InsertLastListStatis(ListStatic \*L, ElType X) ;
4. Selektor:
  - a. ElType GetListStatis(ListStatic L, IdxType i) ;

### 3.1.2 Persoalan yang Dapat Diselesaikan

ADT ini digunakan untuk menyimpan nama penyanyi yang ada pada *file* config untuk ADT *list* penyanyi. Kemudian untuk *list* statis digunakan untuk menyimpan lagu-lagu yang ditampilkan pada suatu album.

### 3.1.3 Alasan Pemilihan ADT

ADT ini digunakan karena diperlukan penyimpanan untuk daftar penyanyi yang ada di *file* konfigurasi dan daftar lagu yang ditampilkan pada suatu album. Sehingga memudahkan dalam pencarian ketika dibutuhkan hanya dengan mengakses indeksinya.

### 3.1.4 Implementasi Sebagai ADT

*File-file* yang menggunakan ADT List adalah main.c, list.h, list.c, driverl.c, driverlistC.c, enhance.h, enhance.c, Clist.h, Clist.c, load.h, load.c, save.h, save.c, pick.h, dan pick.c.

## 3.2 Playlist (Array Dinamis dan Linked List)

### 3.2.1 Sketsa Struktur Data

```
typedef char* valuetype;
typedef struct tElmtlist *address;

// ===== STRUCT AFTER DEFAULT =====

// Struct Song
typedef struct{
    valuetype namaP;
    valuetype namaA;
    valuetype namaS;
} Song;

// playlist
typedef struct tElmtlist {
    Song info;
    address next;
} ElmtList;

typedef struct{
    address First;
    valuetype namaPlaylist;
} List;

typedef struct
{
    List *playlist;
    int Capacity;
    int Neff;
} ArrayDinPlaylist;
```

Pada ADT *playlist* terdapat dua tipe ADT yaitu *array* dinamis dan *linked list*. *Array* dinamis digunakan untuk menyimpan *playlist* sedangkan lagu-lagu dari *playlist* tersebut disimpan dalam *linked list*.

Berikut primitif yang digunakan untuk ADT *array* dinamis:

1. Konstruktor:
  - a. ArrayDinPlaylist MakeArrayDin();

2. Prototype:
  - a. `boolean IsEmptyArrayDin(ArrayDinPlaylist array);`
  - b. `int Length(ArrayDinPlaylist array);`
3. Destruktor:
  - a. `void DeallocateArrayDin(ArrayDinPlaylist *array);`
4. Selektor:
  - a. `address Get(ArrayDinPlaylist array, int i);`
  - b. `int GetCapacity(ArrayDinPlaylist array);`
5. Operasi-Operasi:
  - a. `void InsertAt(ArrayDinPlaylist *array, address P, int i);`
  - b. `void InsertLast(ArrayDinPlaylist *array, address el);`
  - c. `void InsertFirst(ArrayDinPlaylist *array, address el);`
  - d. `void DeleteAt(ArrayDinPlaylist *array, int i);`
  - e. `void DeleteLast(ArrayDinPlaylist *array);`
  - f. `void DeleteFirst(ArrayDinPlaylist *array);`

Berikut primitif yang digunakan untuk ADT *linked list*:

1. Konstruktor:
  - a. `void CreateEmpty (List *L);`
  - b. `address Alokasi (Song X);`
2. Destruktor:
  - a. `void Dealokasi (address *P);`
3. Prototype:
  - a. `boolean IsEmptyListLinier (List L);`
  - b. `boolean SearchLaguPlaylist(List L, char* namalagu);`
  - c. `address Search (List L, Song X);`
  - d. `int NbElmt (List L)`
  - e.
4. Operasi-Operasi:
  - a. `void InsVFirst (List *L, Song X);`
  - b. `void InsVLast (List *L, Song X);`
  - c. `void DelVFirst (List *L, Song *X);`
  - d. `void DelVLast (List *L, Song *X);`
5. Display:
  - a. `void PrintInfo (List L);`

Primitif lain untuk membantu dalam pengimplementasian *command playlist*:

1. Konstruktor:
  - a. `void CreatePlaylist (ArrayDinPlaylist *array, valuetype namaplaylist);`
  - b. `Song CreateSong(valuetype namapenyanyi, valuetype namaalbum, valuetype namalagu);`
2. Operasi-Operasi:
  - a. `void PlaylistAddSong (ArrayDinPlaylist *array, int i, Song lagu);`
  - b. `void tukarplaylist(ArrayDinPlaylist *array, int idP, int x, int y);`
  - c. `void DeleteLaguPlaylist (ArrayDinPlaylist *array, int idP, int x, Song *lagu);`
  - d. `void DeletePlaylist (ArrayDinPlaylist *array, int idP);`



### 3.2.2 Persoalan yang Dapat Diselesaikan

ADT ini digunakan untuk menyimpan *playlist* dan lagu-lagu yang ada dalam *playlist* sehingga berguna untuk pengimplementasian *command playlist* atau *command* lain yang berhubungan dengan *playlist*.

### 3.2.3 Alasan Pemilihan ADT

Alasan memilih ADT ini karena *array* dinamis memiliki ukuran yang fleksibel dan linked list agar tiap *list* bisa saling terhubung secara kontigu. Dibutuhkan pula penyimpanan yang fleksibel untuk menyimpan lagu-lagu pada *playlist*. Disini kami menyimpan elemen dengan tipe baru yaitu tipe *song* agar penyanyi, album, dan lagu dapat langsung disimpan sehingga bisa diakses dengan mudah.

### 3.2.4 Implementasi Sebagai ADT

*File-file* yang menggunakan ADT Playlist adalah *main.c*, *playlist.h*, *playlist.c*, *Cplaylist.h*, dan *Cplaylist.c*, *enhance.h*, *enhance.c*, *Clist.h*, *Clist.c*, *load.h*, *load.c*, *save.h*, *save.c*.

## 3.3 Mesin Kalimat (Mesin Kata dan Mesin Karakter)

### 3.3.1 Mesin Kata

#### 3.3.1.1 Sketsa Struktur Data

```
#define NMax 50
#define BLANK '\n'

typedef struct
{
    char TabWord[NMax]; /* container penyimpan kata, indeks yang dipakai [0..NMax-1] */
    int Length;
} Word;

/* State Mesin Kata */
extern boolean EndWord;
extern Word currentWord;
```

1. Operasi-Operasi:
  - a. void IgnoreBlanks();
  - b. void STARTWORD(char \* FILE);
  - c. void STARTINPUTKATA();
  - d. void ADVWORD();
  - e. void CopyWord();
  - f. char \* ConcatChar(char \* path, char \* filename);
2. Konverter:
  - a. int WordToInt(Word word);
  - b. char \* WordToString (Word word);
  - c. Word StringtoWord (char\*string);

- d. char \*commWordToString(Word word);
- 3. Selektor:
  - a. Word takeword(Word command, int ke);
  - b. Word takekata(Word w);
  - c. Word takenonspace(Word w);
  - d. void hapustikom(Word \*w);
  - e. Word takewordsemicolon(Word command, int ke);
- 4. Tes Kebenaran:
  - a. boolean IsEqual(Word w, char \*c);
  - b. boolean IsInWord(char\* dicari, Word sumber);
  - c. boolean IsEqualString(char\* c1, char \*c2);

### 3.3.1.2 Persoalan yang Dapat Diselesaikan

ADT ini digunakan untuk melakukan *parsing command* program dan pembacaan *file* konfigurasi ke dalam aplikasi. Selain itu, dipakai untuk mengkonversi tipe data tertentu agar sesuai dengan spesifikasi.

### 3.3.1.3 Alasan Pemilihan ADT

ADT ini digunakan karena diperlukannya suatu fungsi untuk membaca file konfigurasi dan file penyimpanan, serta membaca masukan dari pengguna tanpa menggunakan fungsi *scanf* bawaan C yang tidak diperbolehkan digunakan dalam tugas besar ini. Karena hasil pembacaan yang didapat dari fungsi ini berbentuk tipe data *Word*, maka ADT ini juga diperlukan untuk mengkonversi dan melakukan operasi-operasi tertentu terhadap hasil pembacaan tersebut

### 3.3.1.4 Implementasi Sebagai ADT

*File-file* yang menggunakan ADT Mesin Kata adalah *mesinkata.h*, *mesinkata.c*, *main.c*, *ADT.h*, *ADT.c*, *enhance.h*, *enhance.c*, *Clist.h*, *Clist.c*, *load.h*, *load.c*, *pick.h*, *pick.c*, *save.h*, *save.c*, dan *driverkata.c*.

## 3.3.2 Mesin Karakter

### 3.3.2.1 Sketsa Struktur Data

```
#define MARK '\0'
/* State Mesin */
extern char currentChar;
extern boolean EOP;
```

Primitif yang digunakan dalam ADT ini:

1. void START(char \*FILE);
2. void STARTINPUT();
3. void ADV();
4. char GetCC();
5. boolean IsEOP();

### 3.3.2.2 Persoalan yang Dapat Diselesaikan

ADT ini digunakan untuk mengakses char dari sebuah *command* yang berasal dari *user* ataupun dari *file* yang nantinya akan diakses di ADT mesin kata.

### 3.3.2.3 Alasan Pemilihan ADT

ADT ini diperlukan karena kita membutuhkan START dan ADV untuk mengakses char dari sebuah *input* dari *user* ataupun dari *file*.

### 3.3.2.4 Implementasi Sebagai ADT

*File-file* yang menggunakan ADT Mesin Kata adalah mesinkarakter.h, mesinkarakter.c, mesinkata.h, mesinkata.c, main.c, ADT.h, ADT.c, enhance.h, dan enhance.c, Clist.h, Clist.c, load.h, load.c, pick.h, pick.c, save.h, save.c, dan driverkata.c.

## 3.4 Queue

### 3.4.1 Sketsa Struktur Data

```
#define NMAX 20
#define EMPTY ""
#define IDX_UNDEF -999

// ===== STRUCT AFTER DEFAULT =====
// queue buat queue
typedef struct
{
    Song buffer [NMAX*3];
    int idxTail;
} queue;
```

### 3.4.2 Persoalan yang Dapat Diselesaikan

ADT ini digunakan untuk menyimpan sebuah *queue* dari kumpulan lagu-lagu, ADT ini dapat menerima tambahan lagu secara individu, maupun secara berkelompok dalam wujud suatu *playlist*. ADT ini digunakan dalam implementasi dari *command queue*.

### 3.4.3 Alasan Pemilihan ADT

ADT ini digunakan dalam *command queue* yang menerapkan konsep FIFO (*First In First Out*). Elemen yang terdapat dalam ADT ini pun sudah dimodifikasi dengan tipe *song*, sehingga bisa lebih mempermudah.

### 3.4.4 Implementasi Sebagai ADT

*File-file* yang menggunakan ADT Queue adalah main.c, queue.h, queue.c, driver.c, load.h, load.c, play.h, play.c, Cqueue.h, Cqueue.c, save.h, dan save.c.

## 3.5 Stack

### 3.5.1 Sketsa Struktur Data

```
#define NMAX 20
#define EMPTY ""
#define IDX_UNDEF -999

// =====** STRUCT AFTER DEFAULT **=====
// stack buat history
typedef struct
{
    Song T[NMAX*3];
    int TOP;
} Stack;
```

### 3.5.2 Persoalan yang Dapat Diselesaikan

ADT ini digunakan untuk menyimpan elemen lagu yang telah dimainkan, seperti riwayat lagu yang telah dimainkan. ADT ini menerima tambahan elemen ketika lagu telah selesai dimainkan, ataupun telah di lanjutkan.

### 3.5.3 Alasan Pemilihan ADT

ADT ini digunakan dalam *command play* dan juga *song*, dimana ADT ini dimanfaatkan sebagai riwayat lagu yang telah dimainkan, sehingga ADT ini dapat diakses menggunakan *command song*. ADT ini digunakan dikarenakan untuk mengimplementasikan konsep riwayat lagu, terdapat kesamaan dengan konsep stack, yaitu LIFO (*Last In First Out*), sehingga saat pemanggilan *command song previous*, yang terakhir kali masuk ke dalam ADT ini, akan yang pertama kali kembali dipanggil.

### 3.5.4 Implementasi Sebagai ADT

*File-file* yang menggunakan ADT Stack adalah main.c, stack.h, stack.c, driverstack.c, load.h, load.c, play.h, play.c, save.h, save.c.

## 3.6 Set

### 3.6.1 Sketsa Struktur Data

```
#define NMAX 20
#define EMPTY ""
#define IDX_UNDEF -999

typedef char* valuetype;
typedef int IDs;
```

```
// ***** STRUCT DEFAULT *****
//struct pada setsong
typedef struct{
    IDs Idalbum;
    valuetype namalagu;
} elmtsong;

typedef struct{
    elmtsong Elements[NMAX*5];
    int count;
} SetSong;
```

### 3.6.2 Persoalan yang Dapat Diselesaikan

Digunakan untuk menyimpan lagu dari suatu album sehingga tidak ada yang duplikat.

### 3.6.3 Alasan Pemilihan ADT

ADT ini digunakan karena diperlukan penyimpanan untuk lagu dari suatu album agar tidak duplikat dari file konfigurasi.

### 3.6.4 Implementasi Sebagai ADT

File-file yang menggunakan ADT Set adalah main.c, set.h, set.c, driverset.c,

## 3.7 Map

### 3.7.1 Sketsa Struktur Data

```
#define NMAX 20
#define EMPTY ""
#define IDX_UNDEF -999

typedef int keytype;
typedef char* valuetype;
typedef int IDs;

// ***** STRUCT DEFAULT *****
//struct pada album
typedef struct{
    keytype keyAlbum;
    IDs IdPenyanyi;
    valuetype valueAlbum;
} Album;

typedef struct{
    Album Elements[NMAX*3];
```

```
int count;  
} MapAlbum;
```

### 3.7.2 Persoalan yang Dapat Diselesaikan

ADT Map digunakan untuk menyimpan album tiap penyanyi dan juga menyimpan album untuk tiap lagu. Kita bisa menentukan kepemilikan album dari suatu penyanyi dari IDs penyanyi yang disimpan di Album.

### 3.7.3 Alasan Pemilihan ADT

Karena ADT map dapat memetakan beberapa nilai dari suatu ADT sehingga album dari suatu penyanyi dan lagu-lagunya dapat disimpan.

### 3.7.4 Implementasi Sebagai ADT

*File-file* yang menggunakan ADT Set adalah main.c, map.h, map.c, drivermap.c

## 4 Program Utama

Program utama dimulai dengan pendefinisian *header* yang boleh digunakan untuk spesifikasi program, yakni <stdio.h>, <stdlib.h>, dan seluruh *header* dari semua fungsi/prosedur dan ADT yang telah dibuat. Terdapat sembilan belas *header* yang telah dibuat yaitu ADT.h, list.h, mesinkarakter.h, mesinkata.h, playlist.h, queue.h, stack.h, help.h, load.h, pick.h, Cplaylist.h, invalidcommand.h, song.h, Clist.h, play.h, Cqueue.h, status.h, enhance.h, dan save.h.

Setelah pendefinisian keseluruhan *header* yang dibutuhkan, dilanjutkan dengan pembuatan fungsi mainafter(ListPenyanyi inpenyanyi, MapAlbum inalbum, SetSong insong, queue inqueue, Stack instack, ArrayDinPlaylist inplaylist, CurrentSong incursong, valuetype innamaplaylist, boolean inputisplayplaylist) yang berfungsi sebagai *main* setelah *command* START dan LOAD dijalankan, serta fungsi displaywayangwave() yang berfungsi sebagai *interface* awal program WayangWave. Kemudian, masuk ke bagian utama program yaitu int main() yang berisikan penginisiasian semua variabel, pengosongan variabel, pemanggilan fungsi displaywayangwave(), dan apabila memenuhi syarat looping maka akan terdapat perintah pemanggilan fungsi/prosedur tertentu yang telah dibuat sebelumnya, penginputan dan pencetakan *command* untuk menjalankan program utama, pemanggilan fungsi/prosedur mainafter dan <INVALID COMMAND> apabila *command* yang dimasukkan tidak sesuai, tidak diketahui, maupun tidak bisa dieksekusi.

Jika *command* tidak diketahui dan tidak sesuai maka pengguna akan diarahkan untuk memanggil atau memasukkan *command* HELP. Jika pengguna memasukkan *command* START, pengguna dapat langsung mengakses semua *command* yang tersedia dan yang ada di program aplikasi WayangWave. Dari masing-masing *command* yang berhasil dibaca, maka masing-masing fungsi yang ada di dalamnya akan langsung dipanggil.

Berikut ini adalah algoritma dari program utama yang kami buat dengan *file* yang diberi nama “main.c”:

```

#include "ADT.h"
#include <stdio.h>
#include <stdlib.h>

#include "../ADT/List/list.h"
#include "../ADT/MesinKalimat/mesinkarakter.h"
#include "../ADT/MesinKalimat/mesinkata.h"
#include "../ADT/Playlist/playlist.h"
#include "../ADT/Queue/queue.h"
#include "../ADT/Stack/stack.h"

#include "../command/Help/help.h"
#include "../command/Load/load.h"
#include "../command/Pick/pick.h"
#include "../command/Playlist/Cplaylist.h"
#include "../command/invalidcommand/invalidcommand.h"
#include "../command/Song/song.h"
#include "../command/ListC/Clist.h"
#include "../command/Play/play.h"
#include "../command/Queue/Cqueue.h"
#include "../command/Status/status.h"
#include "../command/Enhance/enhance.h"
#include "../command/Save/save.h"

void mainafter(ListPenyanyi inpenyanyi, MapAlbum inalbum, SetSong insong, queue inqueue, Stack instack,
ArrayDinPlaylist inplaylist, CurrentSong incursong, valuetype innamaplaylist, boolean inputisplayplaylist){
    // input command
    printf(">>> ");
    STARTINPUTKATA();
    Word input = takeword(currentWord,1);

    while(true){
        //quit
        if (IsEqual(input,"QUIT;")){
            break;
        }

        //invalid comm 2
        else if (IsEqual(input,"START;") || IsEqual(input,"LOAD;")){
            invalid_command2();
        }

        //list
        else if (IsEqual(input,"LIST;")){
            Word nextinput = takeword(currentWord,2);
            if (IsEqual(nextinput,"DEFAULT;")){
                listdefault(inpenyanyi,inalbum,insong);
            }
            else if (IsEqual(nextinput,"PLAYLIST;")){
                Dlistplaylist(inplaylist);
            }
            else{
                invalid_command4();
            }
        }
    }
}

```

```

//play
else if (IsEqual(input,"PLAY")){
    Word nextinput = takeword(currentWord,2);
    if (IsEqual(nextinput,"SONG;")){
        playsong(&incursong,inpenyanyi,inalbum,insong,&inqueue,&instack,&inputisplayplaylist);
    }
    else if (IsEqual(nextinput,"PLAYLIST;")){
        playplaylist(&incursong,inplaylist,&inqueue,&instack,&innamaplaylist,&inputisplayplaylist);
    }
    else{
        invalid_command4();
    }
}

//queue
else if (IsEqual(input,"QUEUE")){
    Word nextinput = takeword(currentWord,2);
    if (IsEqual(nextinput,"SONG;")){
        queuelagu(&inqueue,inpenyanyi,inalbum,insong,&inputisplayplaylist,innamaplaylist,inplaylist);
    }
    else if (IsEqual(nextinput,"PLAYLIST;")){
        queueplaylist(inplaylist,&inqueue,&innamaplaylist,&inputisplayplaylist,incursong);
    }
    else if (IsEqual(nextinput,"SWAP")){
        int validasi = currentWord.Length-11;
        hapustikom(&currentWord);
        int id1 = -99, id2 = -99;
        if (validasi>=4){
            id1 = WordToInt(takeword(currentWord,3));
            id2 = WordToInt(takeword(currentWord,4));
            swapqueue(id1,id2,&inqueue);
        }
        else{
            invalid_command4();
        }
    }
    else if (IsEqual(nextinput,"REMOVE")){
        int validasi = currentWord.Length-13;
        hapustikom(&currentWord);
        if (validasi>=2){
            int id = WordToInt(takeword(currentWord,3));
            removequeue(id,&inqueue);
        }
    }
    else if (IsEqual(nextinput,"CLEAR;")){
        clearqueue(&inqueue);
    }
    else{invalid_command4();}
}

//song
else if (IsEqual(input,"SONG")){
    Word nextinput = takeword(currentWord,2);
    if (IsEqual(nextinput,"NEXT;")){
        songnext(&incursong,&inqueue,&instack);
    }
}

```



```

    }
    else if (IsEqual(nextinput,"PREVIOUS;")){
        songprevious(&incursong,&inqueue,&instack);
    }
    else{
        invalid_command4();
    }
}

//playlist
else if (IsEqual(input,"PLAYLIST")){
    Word nextinput = takeword(currentWord,2);
    if (IsEqual(nextinput,"CREATE;")){
        playlistcreate(&inplaylist);
    }
    else if (IsEqual(nextinput,"ADD;")){
        Word inputadd = takeword(currentWord,3);
        if (IsEqual(inputadd,"SONG;")){
            tambahlaguplaylist(inpenyanyi,inalbum,insong,&inplaylist);
        }
        else if (IsEqual(inputadd,"ALBUM;")){
            tambahalbumplaylist(inpenyanyi,inalbum,insong,&inplaylist);
        }
        else{
            invalid_command4();
        }
    }
    else if (IsEqual(nextinput,"SWAP;")){
        int validasi = currentWord.Length-14;
        hapustikom(&currentWord);
        int IDplaylist = -99, x = -99,y=-99;
        if(validasi >= 6){
            IDplaylist = WordToInt(takeword(currentWord,3));
            x = WordToInt(takeword(currentWord,4));
            y = WordToInt(takeword(currentWord,5));
            swapplaylist(&inplaylist,IDplaylist,x,y);
        }
        else{
            invalid_command4();
        }
    }
    else if (IsEqual(nextinput,"REMOVE;")){
        int validasi = currentWord.Length-16;
        int IDplaylist = -99, x = -99;
        if (validasi >=4){
            IDplaylist = WordToInt(takeword(currentWord,3));
            x = WordToInt(takeword(currentWord,4));
            removeplaylist(&inplaylist,IDplaylist,x);
        }
        else{
            invalid_command4();
        }
    }
    else if (IsEqual(nextinput,"DELETE;")){
        hapusplaylist(&inplaylist);
    }
}

```

```

        else{
            invalid_command4();
        }
    }
    //enhance
    else if (IsEqual(input, "ENHANCE;")){
        enhanceplaylist(&inplaylist, inpenyanyi, inalbum, insong);
    }
    //status
    else if (IsEqual(input, "STATUS;")){
        status(incursong, inqueue, inputisplayplaylist, innamaplaylist);
    }

    //save
    else if (IsEqual(input, "SAVE;")){
        hapustikom(&currentWord);
        char *filename = WordToString(takeword(currentWord, 2));
        save(filename, inpenyanyi, inalbum, insong, inqueue, instack, inplaylist, incursong);
    }

    //help
    else if (IsEqual(input, "HELP;")){
        help_after();
    }

    //invalid comm 1
    else{
        invalid_command5();
    }
    printf(">>> ");
    STARTINPUTKATA();
    input = takeword(currentWord, 1);
}

//quit
printf("\nApakah kamu ingin menyimpan data sesi sekarang (Y/N)? ");
STARTINPUTKATA();
hapustikom(&currentWord);
Word YN = takeword(currentWord, 1);
while (!(IsEqual(YN, "Y") || IsEqual(YN, "N")))
{
    printf("Input tidak valid. Silakan memasukkan 'Y' jika ingin menyimpan data sesi sekarang dan 'N' jika tidak ingin menyimpan data sesi sekarang (Y/N): ");
    STARTINPUTKATA();
    hapustikom(&currentWord);
    YN = takeword(currentWord, 1);
}
if (IsEqual(YN, "Y"))
{
    char *filename;
    printf("Masukkan nama file penyimpanan: ");
    STARTINPUTKATA();
    hapustikom(&currentWord);
    filename = WordToString(currentWord);
}

```

[illegible]

```

CreateQueue(&antrian);
LaguSkrng.namaP = "-";
displaywayangwave();

// input command
printf(">>> ");
STARTINPUTKATA();
Word input = takeword(currentWord,1);

while (true){
    // start
    if (IsEqual(input,"START;")){
        printf("starting game..\n");
        char *file = "./src/config/config.txt";
        STARTCOM(&Penyanyi,&Album,&Song, file);

        printf("Berhasil masuk kedalam aplikasi, selamat menikmati!\n");
        mainafter(Penyanyi,Album,Song,antrian,history,playlist,LaguSkrng,namaplaylist,isplayplaylist);
        break;
    }

    // help
    else if (IsEqual(input,"HELP;")){
        help_before();
    }

    // load
    else if (IsEqual(input,"LOAD")){
        hapustikom(&currentWord);
        char *input2 = WordToString(takeword(currentWord,2));
        char *filename = ConcateChar("src/config/",input2);

        if (Checkload(filename)){
            printf("Load file %s..\n",input2);
            Load(&Penyanyi,&Album,&Song,filename,&antrian,&history,&playlist,&LaguSkrng);
            printf("Berhasil masuk kedalam aplikasi, selamat menikmati!\n");
            mainafter(Penyanyi,Album,Song,antrian,history,playlist,LaguSkrng,namaplaylist,isplayplaylist);
            break;
        }
        else{
            invalid_command3();
        }
    }

    if (IsEqual(input,"LIST") || IsEqual(input,"PLAY") || IsEqual(input,"QUEUE") || IsEqual(input,"SONG") ||
    IsEqual(input,"PLAYLIST") || IsEqual(input,"STATUS;") || IsEqual(input,"SAVE") || IsEqual(input,"QUIT;")){
        invalid_command3();
    }

    //invalid command
    else{
        invalid_command5();
    }

    // input command
    printf(">>> ");
    STARTINPUTKATA();

```

```

        input = takeword(currentWord,1);
    }
}

```

## 5 Algoritma-Algoritma Menarik

Sebenarnya yang paling menarik adalah kami. Tetapi selain itu, ada beberapa algoritma yang menurut kami cukup menarik yaitu:

### 5.1 Random Angka di Enhance

```

#include <time.h>

srand(time(NULL));
int nilaiAcak = rand() % (nilaimax - nilaimin + 1) + nilaimin;

```

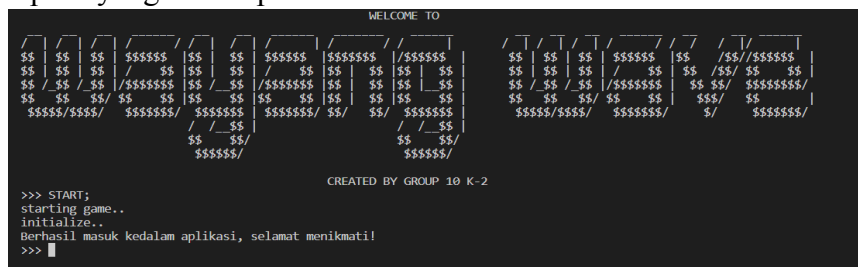
Algoritma ini digunakan pada fitur bonus ENHANCE. Algoritma ini memanfaatkan fungsi yang ada di *library time*. Srand(time(NULL)) berguna untuk menginisialisasi *seed* sehingga *seed* akan berbeda untuk tiap kali program dijalankan. Fungsi rand akan me-random angka acak dari rentang nilai *min* sampai dengan nilai *max*.

## 6 Data Test

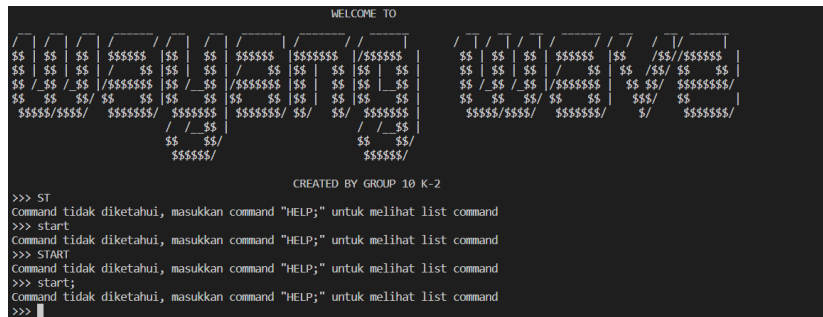
Berikut ini *data test* yang terdapat pada program WayangWave:

### 6.1 START

*Command START* akan membaca *file* konfigurasi *default* yang berisikan daftar penyanyi serta album yang dimiliki. Fitur yang dites dengan data test ini ialah *output* dan tampilan yang muncul ketika pengguna meng-input *command START* dengan benar dan salah. Tujuannya untuk menguji apakah *command START* ini dapat menjalankan keseluruhan program WayangWave. Jika *command START* berhasil maka akan muncul tampilan seperti Gambar 6.1.1. *Command START* berhasil dibaca jika pengguna menginput “START;”. *Command START* gagal terbaca apabila pengguna salah meng-input, yaitu apabila pengguna meng-input selain di luar “START;”, seperti yang tertera pada Gambar 6.1.2.

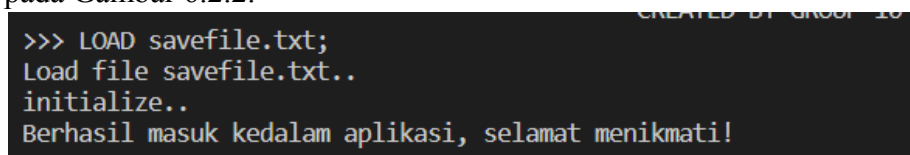


Gambar 6.1.1

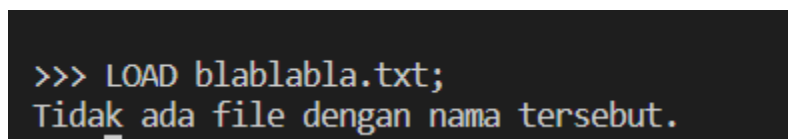


## 6.2 LOAD <filename>

*Command* LOAD <filename> akan membuka *filename* yang merepresentasikan suatu *save file* yang ingin dibuka. *File* diperoleh dari *folder* tertentu, contohnya *save*, lalu *command* LOAD <filename> tersebut akan membaca *save file* <filename> yang berisikan *list* penyanyi, album, dan lagu yang dapat diputar. Fitur yang dites dengan *data test* ini ialah tampilan yang muncul ketika *command* LOAD dipanggil saat sebelum menyimpan *save file* dan saat setelah *save file* disimpan. Tujuannya untuk menguji apakah *command* LOAD dapat membaca *file* dan berfungsi dengan baik. Jika *command* LOAD berhasil dijalankan dan terdapat *save file* yang disimpan maka akan muncul tampilan seperti Gambar 6.2.1. Sebaliknya, jika *command* LOAD berhasil dijalankan tetapi tidak ada *save file* yang disimpan, maka tampilan yang muncul ialah seperti pada Gambar 6.2.2.



Gambar 6.2.1



Gambar 6.2.2

### 6.3 LIST

### 6.3.1 LIST DEFAULT

*Command* LIST DEFAULT akan menampilkan *list* penyanyi yang ada lalu dapat memilih untuk melihat album dari penyanyi yang dipilih serta melihat daftar lagu yang ada dari album yang dipilih. Fitur yang dites dengan data test ini ialah tampilan dan output yang muncul setelah *command* LIST DEFAULT dipanggil, yang mana apabila *input* yang dimasukkan sesuai maka akan muncul tampilan seperti pada Gambar 6.3.1.1 dan Gambar 6.3.1.2. Tujuannya untuk memastikan *command* LIST DEFAULT dapat menampilkan *list* penyanyi, album, dan lagu yang tersedia jika *input* yang dimasukkan benar dan sesuai. Tampilan yang keluar akan seperti di Gambar 6.3.1.1 jika memasukkan konfirmasi “Y;” untuk melihat album ataupun lagu. Jika

STEI- ITB	IF2111-TB-02-10	Halaman 22 dari 48 halaman
<p>Template dokumen ini dan informasi yang dimilikinya adalah milik Sekolah Teknik Elektro dan Informatika ITB dan bersifat rahasia. Dilarang me-reproduksi dokumen ini tanpa diketahui oleh Sekolah Teknik Elektro dan Informatika ITB</p>		

pengguna memasukkan “N;” pada konfirmasi apakah ingin melihat album yang ada, maka tampilan yang keluar adalah seperti di Gambar 6.3.1.2. Apabila pengguna tidak memasukkan “;” (tanda titik koma), maka *output* yang dihasilkan ialah berupa pengingat agar jangan lupa untuk memakai titik koma dan diarahkan untuk mencoba lagi sebagaimana yang tertera pada Gambar 6.3.1.3.

```
>>> LIST DEFAULT;
Daftar Penyanyi :
1. BLACKPINK
2. Arctic Monkeys

Ingin melihat album yang ada? (Y/N) : Y;
Pilih penyanyi untuk melihat album mereka: BLACKPINK;

Daftar album oleh BLACKPINK :
1. BORN PINK
2. THE ALBUM

Ingin melihat lagu yang ada? (Y/N): Y;
Pilih album untuk melihat lagu yang ada di album : BORN PINK;

Daftar Lagu di BORN PINK:
1. Pink Venom
2. Shut Down
3. Typa Girl
4. Ready For Love
>>> █
```

Gambar 6.3.1.1

```
>>> LIST DEFAULT;

Daftar Penyanyi :
1. BLACKPINK
2. Arctic Monkeys

Ingin melihat album yang ada? (Y/N) : N;
```

Gambar 6.3.1.2

```
>>> LIST DEFAULT;

Daftar Penyanyi :
1. BLACKPINK
2. Arctic Monkeys

Ingin melihat album yang ada? (Y/N) : Y

Jangan lupa pakai titik koma. Silakan coba lagi.

Ingin melihat album yang ada? (Y/N) : N

Jangan lupa pakai titik koma. Silakan coba lagi.

Ingin melihat album yang ada? (Y/N) : N;

>>> █
```

Gambar 6.3.1.3

### 6.3.2 LIST PLAYLIST

*Command* LIST PLAYLIST akan menampilkan daftar *playlist* yang dimiliki oleh pengguna. Fitur yang dites dengan *data test* ini ialah tampilan dan *output* yang muncul setelah *command* LIST PLAYLIST dipanggil. Tujuannya untuk memastikan *command* LIST PLAYLIST dapat menampilkan *list playlist* yang dimiliki. *Output* yang disampaikan akan seperti di Gambar 6.3.2.1 jika ternyata pengguna tidak mempunyai daftar *playlist*. Apabila pengguna memiliki *playlist* sebelumnya, maka akan ditampilkan *output* berupa daftar nama-nama *playlist* yang pengguna miliki seperti yang tertera pada Gambar 6.3.2.2.

```
>>> LIST PLAYLIST;

Daftar playlist yang kamu miliki:
Kamu tidak memiliki playlist.
```

Gambar 6.3.2.1

```
>>> LIST PLAYLIST;

Daftar playlist yang kamu miliki:
1. K-POP ENJOYER
2. Ijo Marmut
```

Gambar 6.3.2.2

## 6.4 PLAY

### 6.4.1 PLAY SONG

*Command* PLAY SONG akan memainkan lagu berdasarkan nama penyanyi, nama album, dan id lagu yang di-*input* pengguna yang kemudian ketika *command* ini berhasil dieksekusi maka *queue* dan riwayat lagu akan menjadi kosong. Fitur yang dites dengan *data test* ini ialah tampilan atau *output* yang muncul setelah *command* PLAY SONG dipanggil. Tujuannya untuk memastikan *command* PLAY SONG dapat memainkan atau memutar lagu berdasarkan *input* yang sesuai, serta tidak dapat memainkan lagu jika *input* tidak sesuai. *Output* yang dikeluarkan akan seperti di Gambar 6.4.1.1 jika pengguna memasukkan *input* yang sesuai dan tepat yaitu berupa keterangan informasi judul/nama lagu beserta penyanyinya dari lagu yang sedang diputar/dimainkan. Sebaliknya *output* yang diberikan akan seperti 6.4.1.2 jika *input* yang dimasukkan pengguna tidak sesuai dan tidak tepat sehingga akan menyuruh pengguna untuk mencoba lagi sampai *input* tepat dan sesuai.

```
>>> PLAY SONG;

Daftar Penyanyi :
1. BLACKPINK
2. Arctic Monkeys

Masukkan Nama Penyanyi yang dipilih : BLACKPINK;

Daftar album oleh BLACKPINK :
1. BORN PINK
2. THE ALBUM

Masukkan Nama Album yang dipilih : BORN PINK;

Daftar lagu Album BORN PINK oleh BLACKPINK:
1. Pink Venom
2. Shut Down
3. Typa Girl
4. Ready For Love

Masukkan ID lagu yang dipilih : 3;

Memutar lagu 'Typa Girl' oleh 'BLACKPINK'.
```

Gambar 6.4.1.1

```
>>> PLAY SONG;

Daftar Penyanyi :
1. BLACKPINK
2. Arctic Monkeys

Masukkan Nama Penyanyi yang dipilih : BLAXKPINK
Jangan lupa pakai titik koma. Silakan coba lagi.

Masukkan Nama Penyanyi yang dipilih : BLAXPNK;

Penyanyi BLAXPNK tidak ada dalam daftar, silakan coba lagi

Masukkan Nama Penyanyi yang dipilih :
```



Gambar 6.4.1.2

## 6.4.2 PLAY PLAYLIST

*Command* PLAY PLAYLIST akan memainkan lagu berdasarkan *id playlist* yang di-*input* oleh pengguna yang kemudian setelah berhasil dieksekusi maka *current song* akan menjadi lagu pada urutan pertama *playlist* dan *queue* akan berisi semua lagu yang ada dalam *playlist* yang akan dimainkan, dan isi riwayat lagu sama dengan *queue* tetapi dengan urutan yang di-*reverse*. Fitur yang dites dengan *data test* ini ialah hasil *output* yang muncul ketika *command* PLAY PLAYLIST dipanggil. Tujuannya untuk memastikan *command* PLAY PLAYLIST dapat memainkan atau memutar *playlist* yang tersedia/dimiliki berdasarkan *input* yang sesuai, serta tidak dapat memainkan *playlist* jika *input* tidak sesuai dan jika tidak memiliki *playlist* sama sekali. *Output* yang dikeluarkan akan seperti di Gambar 6.4.2.1 jika pengguna memanggil *command* PLAY PLAYLIST tetapi tidak memiliki *playlist* sama sekali. *Output* yang dikeluarkan akan seperti di Gambar 6.4.2.2 jika pengguna memanggil *command* PLAY PLAYLIST tetapi tidak ada *playlist* dengan *id* yang diinput. Serta *output* yang dikeluarkan akan seperti di Gambar 6.4.2.3 jika pengguna memanggil *command* PLAY PLAYLIST dan memiliki *playlist* dengan *id* yang di-*input*.

```
>>> PLAY PLAYLIST;  
Playlist kosong, tidak ada yang bisa dimainkan.
```

Gambar 6.4.2.1

```
>>> PLAY PLAYLIST;  
  
Masukkan ID Playlist yang dipilih : 3;  
  
ID 3 tidak ada dalam daftar, silakan coba lagi
```

Gambar 6.4.2.2

```
>>> PLAY PLAYLIST;  
  
Masukkan ID Playlist yang dipilih : 1;  
  
Memutar playlist 'K-POP ENJOYER'.
```

Gambar 6.4.2.3

## 6.5 QUEUE

### 6.5.1 QUEUE SONG

*Command* QUEUE SONG akan menambahkan lagu ke dalam *queue* yakni akan menerima *input* lagu berdasarkan nama penyanyi, nama album, dan *id* dari lagu yang ingin dimasukkan ke dalam *queue*. Fitur yang dites dengan *data test* ini ialah hasil *output* dan tampilan yang muncul ketika *command* QUEUE SONG dipanggil. Tujuannya untuk memastikan *command* QUEUE SONG dapat menambahkan lagu ke dalam antrian serta memastikan *command* QUEUE SONG tidak dapat berjalan jika *input* tidak sesuai. *Output* yang dikeluarkan akan seperti di Gambar 6.5.1.2 jika pengguna memanggil *command* QUEUE SONG dengan *input* yang sesuai dan tepat. *Output* yang dikeluarkan akan seperti di Gambar 6.5.1.1 jika pengguna memanggil *command* QUEUE

SONG dengan *input* yang tidak benar dan tidak tepat sehingga program akan menyuruh pengguna untuk terus mencoba menginput dengan benar dan sesuai agar dapat menambahkan lagu ke *queue*/antrian.

```
>>> QUEUE SONG;

Daftar Penyanyi :
1. BLACKPINK
2. Arctic Monkeys

Masukkan Nama Penyanyi yang dipilih : BLACKPINK;

Daftar album oleh BLACKPINK :
1. BORN PINK
2. THE ALBUM

Masukkan Nama Album yang dipilih : THE ALBUM;

Daftar lagu Album THE ALBUM oleh BLACKPINK:
1. How You Like That
2. Ice Cream (with Selena Gomez)
3. Bet You Wanna (Feat. Cardi B)

Masukkan ID lagu yang dipilih : 1
Jangan lupa pakai titik koma. Silakan coba lagi.

Masukkan ID lagu yang dipilih : 1;

Berhasil menambahkan lagu 'How You Like That' oleh 'THE ALBUM' ke queue.
```

Gambar 6.5.1.1

```
>>> QUEUE SONG;

Daftar Penyanyi :
1. BLACKPINK
2. Arctic Monkeys

Masukkan Nama Penyanyi yang dipilih : Arctic Monkeys
Jangan lupa pakai titik koma. Silakan coba lagi.

Masukkan Nama Penyanyi yang dipilih : Arctic Monkeys;

Daftar album oleh Arctic Monkeys :
1. Favourite Worst Nightmare
2. Humbug
3. AM

Masukkan Nama Album yang dipilih : AM;

Daftar lagu Album AM oleh Arctic Monkeys:
1. Do I Wanna Know ?
2. R U Mine?
3. Arabella

Masukkan ID lagu yang dipilih : 2;

Berhasil menambahkan lagu 'R U Mine?' oleh 'AM' ke queue.
```

Gambar 6.5.1.2

## 6.5.2 QUEUE PLAYLIST

*Command* QUEUE PLAYLIST akan menambahkan lagu yang ada dalam *playlist* ke dalam *queue* yakni akan menerima *input* dari *id playlist* yang ingin dimasukkan ke dalam *queue*. Fitur yang dites dengan *data test* ini ialah hasil output dan tampilan yang muncul ketika *command* QUEUE PLAYLIST dipanggil. Tujuannya untuk memastikan *command* QUEUE PLAYLIST dapat menambahkan lagu yang ada dalam *playlist* ke dalam antrian serta memastikan *command* QUEUE PLAYLIST tidak dapat berjalan jika *input id playlist* tidak sesuai. *Output* yang dikeluarkan akan seperti di Gambar 6.5.2.1 jika pengguna memanggil *command* QUEUE PLAYLIST tetapi tidak memiliki *playlist* sehingga tidak ada *playlist* yang bisa di-*queue*. *Output* yang dikeluarkan akan seperti di Gambar 6.5.2.2 jika pengguna memanggil *command* QUEUE PLAYLIST dengan *input id playlist* yang sesuai dan benar. *Output* yang dikeluarkan akan seperti di Gambar 6.5.2.3 jika pengguna memanggil *command* QUEUE PLAYLIST dengan *input* yang tidak benar dan tidak tepat (yang dalam hal ini tidak ada *playlist* dengan *id* yang diinput)

sehingga program akan menyuruh pengguna untuk terus mencoba menginput dengan benar dan sesuai agar dapat menambahkan lagu yang ada dalam *playlist* ke dalam *queue*/antrian.

```
>>> QUEUE PLAYLIST;  
Tidak ada playlist yang bisa diqueue.
```

Gambar 6.5.2.1

```
>>> QUEUE PLAYLIST;  
  
Masukkan ID Playlist yang dipilih : 2;  
  
Berhasil menambahkan playlist 'K-POP ENJOYER' ke queue.
```

Gambar 6.5.2.2

```
>>> QUEUE PLAYLIST;  
  
Masukkan ID Playlist yang dipilih : 6;  
  
ID 6 tidak ada dalam daftar, silakan coba lagi  
  
Masukkan ID Playlist yang dipilih : █
```

Gambar 6.5.2.3

### 6.5.3 QUEUE SWAP <x> <y>

*Command* QUEUE SWAP <x> <y> akan menukar lagu pada urutan ke-x dan juga urutan ke-y. Fitur yang dites dengan *data test* ini ialah hasil *output* ketika *command* QUEUE SWAP berhasil dipanggil. Tujuannya untuk memastikan *command* QUEUE SWAP <x> <y> dapat menukar lagu pada urutan ke-x dan urutan ke-y serta berfungsi dengan baik. *Output* yang dikeluarkan akan seperti di Gambar 6.5.3.1 jika pengguna berhasil memanggil *command* QUEUE SWAP <x> <y> dengan *input* yang benar dan tepat serta sesuai. *Output* yang dikeluarkan akan seperti di Gambar 6.5.3.2 jika pengguna memanggil *command* QUEUE SWAP <x> <y> dengan *input* yang tidak sesuai dan tidak ada lagu dengan urutan nomor berikut.

```
>>> QUEUE SWAP 1 2;  
Lagu 'R U Mine?' berhasil ditukar dengan 'How You Like That'
```

Gambar 6.5.3.1

```
>>> QUEUE SWAP 80 2;  
Lagu dengan urutan ke 80 tidak terdapat dalam queue!
```

Gambar 6.5.3.2

### 6.5.4 QUEUE REMOVE <id>

*Command* QUEUE REMOVE <id> akan menghapus lagu dari *queue* yakni dengan menerima *input* berupa urutan lagu (*id*) yang ingin dihapus dari *queue*. Fitur yang dites dengan *data test* ini ialah hasil *output* ketika *command* QUEUE REMOVE dipanggil. Tujuannya untuk memastikan *command* QUEUE REMOVE <id> dapat menghapus lagu dari antrian sesuai *input* urutan lagu

(id). *Output* yang dikeluarkan akan seperti di Gambar 6.5.4.1 jika pengguna berhasil memanggil *command* QUEUE REMOVE <id> dengan *input* yang benar dan tepat serta sesuai. *Output* yang dikeluarkan akan seperti di Gambar 6.5.4.2 jika pengguna memanggil *command* ini dengan *input* yang kurang sesuai dan tidak ada lagu dengan urutan nomor berikut.

```
>>> QUEUE REMOVE 2;
Lagu 'How You Like That' oleh 'BLACKPINK' telah dihapus dari queue!
```

Gambar 6.5.4.1

```
>>> QUEUE REMOVE 10;
Lagu dengan urutan ke 10 tidak ada.
```

Gambar 6.5.4.2

## 6.5.5 QUEUE CLEAR

*Command* QUEUE CLEAR akan mengosongkan *queue*. Fitur yang dites dengan *data test* ini ialah hasil *output* ketika *command* QUEUE CLEAR dapat dipanggil. Tujuannya untuk memastikan *command* QUEUE CLEAR dapat berhasil mengosongkan antrian seperti di Gambar 6.5.5.1.

```
>>> QUEUE CLEAR;
Queue berhasil dikosongkan.
```

Gambar 6.5.5.1

## 6.6 SONG

### 6.6.1 SONG NEXT

*Command* SONG NEXT akan memutar lagu yang berada di dalam *queue* yang kemudian lagu yang sedang diputar ini akan ditambah ke dalam daftar riwayat putar lagu, jika *queue* kosong maka yang diputar adalah lagu yang sedang diputar. Fitur yang dites dengan *data test* ini ialah hasil *output* dan tampilan yang muncul ketika *command* SONG NEXT dipanggil. Tujuannya untuk memastikan *command* SONG NEXT dapat memutar lagu yang sedang diputar saat ini jika antrian kosong. *Output* yang dikeluarkan akan seperti di Gambar 6.6.1.1 jika pengguna memanggil *command* SONG NEXT tetapi tidak memiliki lagu dalam *queue*. *Output* yang dikeluarkan akan seperti di Gambar 6.6.1.2 jika pengguna memanggil *command* SONG NEXT tetapi *queue* kosong sehingga akan memutar lagu yang saat ini sedang diputar. *Output* yang dikeluarkan akan seperti di Gambar 6.5.1.3 jika pengguna memanggil *command* SONG NEXT dan memiliki lagu dalam *queue* dan sedang memutar lagu yang berbeda dengan lagu dalam *queue*.

```
>>> SONG NEXT;
Command tidak bisa dieksekusi!
```

Gambar 6.6.1.1

```
Memutar lagu 'Pink Venom' oleh 'BLACKPINK'.
>>> SONG NEXT;
Queue kosong, memutar kembali lagu
'Pink Venom' oleh 'BLACKPINK'
```

Gambar 6.6.1.2

```
>>> SONG NEXT;  
Memutar lagu selanjutnya  
'Bet You Wanna (Feat. Cardi B)' oleh 'BLACKPINK'
```

Gambar 6.6.1.3

## 6.6.2 SONG PREVIOUS

*Command* SONG PREVIOUS akan memutar lagu yang terakhir kali diputar yang mana lagu yang sedang diputar kemudian akan ditambah ke dalam *queue* dengan urutan pertama. Jika daftar riwayat lagu kosong, maka yang diputar ialah lagu yang sedang diputar. Fitur yang dites dengan *data test* ini ialah hasil *output* dan tampilan yang muncul ketika *command* ini dipanggil. Tujuannya untuk memastikan *command* SONG PREVIOUS dapat memutar lagu yang terakhir kali diputar serta dapat memutar lagu yang sedang diputar saat ini jika riwayat lagu kosong. *Output* yang dikeluarkan akan seperti di Gambar 6.6.2.1 jika pengguna memanggil *command* SONG PREVIOUS tetapi daftar riwayat lagu kosong sehingga akan dipanggil lagu yang sedang diputar untuk dimainkan lagi. *Output* yang dikeluarkan akan seperti di Gambar 6.6.2.2 jika pengguna memanggil *command* SONG PREVIOUS dan memiliki daftar lagu yang pernah diputar sehingga akan dipanggil lagu yang terakhir kali diputar untuk dimainkan lagi.

```
>>> SONG PREVIOUS;  
Riwayat lagu kosong, memutar kembali lagu  
'Ice Cream (with Selena Gomez)' oleh "BLACKPINK"
```

Gambar 6.6.2.1

```
>>> SONG PREVIOUS;  
Memutar lagu sebelumnya  
'Typa Girl' oleh 'BLACKPINK'
```

Gambar 6.6.2.2

## 6.7 PLAYLIST

### 6.7.1 PLAYLIST CREATE

*Command* PLAYLIST CREATE akan membuat *playlist* baru dan ditambahkan pada daftar *playlist* pengguna yang mana keadaan awal *playlist* adalah kosong dan nama *playlist* dapat sama dengan *playlist* yang sudah ada. Fitur yang dites dengan *data test* ini ialah hasil *output* dan tampilan yang muncul ketika *command* ini dipanggil. Tujuannya untuk memastikan *command* PLAYLIST CREATE dapat membuat *playlist* baru dan ditambahkan pada daftar *playlist* pengguna serta memastikan *command* ini tidak dapat berjalan dengan baik jika *input* tidak valid. *Output* yang dikeluarkan akan seperti di Gambar 6.7.1.1 dan 6.7.1.2 jika pengguna memanggil *command* PLAYLIST CREATE dengan *input* yang sesuai dan tepat aturan. *Output* yang dikeluarkan akan seperti di Gambar 6.7.1.3 dan 6.7.1.4 jika pengguna memanggil *command* ini dengan *input* yang tidak sesuai aturannya sehingga program akan menyuruh pengguna untuk terus mencoba lagi dengan meng-*input* dengan benar dan sesuai agar *playlist* dapat dibuat. Aturannya dalam penamaan *playlist* adalah minimal terdapat 3 karakter dan menggunakan tanda “,” (titik koma) di akhir penulisan nama *playlist*.

```
>>> PLAYLIST CREATE;
Masukkan nama playlist yang ingin dibuat : K-POP ENJOYER;
Playlist K-POP ENJOYER berhasil dibuat!
Silakan masukkan lagu - lagu artis terkini kesayangan Anda!
```

Gambar 6.7.1.1

```
>>> PLAYLIST CREATE;
Masukkan nama playlist yang ingin dibuat : YES;
Playlist YES berhasil dibuat!
Silakan masukkan lagu - lagu artis terkini kesayangan Anda!
```

Gambar 6.7.1.2

```
>>> PLAYLIST CREATE;
Masukkan nama playlist yang ingin dibuat : A;
Minimal terdapat 3 karakter. Silakan coba lagi.
```

Gambar 6.7.1.3

```
Masukkan nama playlist yang ingin dibuat :   ;
Minimal terdapat 3 karakter selain white space dalam nama playlist. Silakan coba lagi.
```

Gambar 6.7.1.4

## 6.7.2 PLAYLIST ADD

### 6.7.2.1 PLAYLIST ADD SONG

*Command* PLAYLIST ADD SONG berarti menambahkan lagu ke dalam *playlist* yang dimiliki. Fitur yang dites dengan *data test* ini ialah hasil *output* dan tampilan yang muncul ketika *command* ini dipanggil. Tujuannya untuk memastikan *command* PLAYLIST ADD SONG dapat menambahkan lagu ke dalam *playlist* serta memastikan tidak dapat berjalan dengan baik jika *input* tidak sesuai. *Output* yang dikeluarkan akan seperti di Gambar 6.7.2.1.1 jika *input* benar atau valid, dan akan seperti Gambar 6.7.2.1.2 jika pengguna memanggil *command* ini dengan *input* yang tidak valid, sehingga program akan menyuruh pengguna untuk mencoba lagi.

```
>>> PLAYLIST ADD SONG;
Daftar Penyanyi :
1. BLACKPINK
2. Arctic Monkeys
Masukkan Nama Penyanyi yang dipilih : Arctic Monkeys;
Daftar album oleh Arctic Monkeys :
1. Favourite Worst Nightmare
2. Humbug
3. AM
Masukkan Nama Album yang dipilih : Favourite Worst Nightmare;
Daftar lagu Album Favourite Worst Nightmare oleh Arctic Monkeys:
1. Brannan
2. Teddy Picker
3. Fluorescent Adolescent
4. Old Yellow Bricks
5. 505
Masukkan ID lagu yang dipilih : 5;
Daftar Playlist :
1. K-POP ENJOYER
2. ASMR 205
3. On Repeat Daily
Masukkan ID Playlist yang dipilih : 3;
Lagu dengan judul "505" pada album Favourite Worst Nightmare oleh penyanyi Arctic Monkeys berhasil ditambahkan ke dalam playlist On Repeat Daily.
```

Gambar 6.7.2.1.1

```
>>> PLAYLIST ADD SONG;
Daftar Penyanyi :
1. BLACKPINK
2. Arctic Monkeys
Masukkan Nama Penyanyi yang dipilih : Arctix Monkey;
Penyanyi Arctix Monkey tidak ada dalam daftar, silakan coba lagi
```

Gambar 6.7.2.1.2

### 6.7.2.2 PLAYLIST ADD ALBUM

Command `PLAYLIST ADD ALBUM` berarti menambahkan album ke dalam *playlist* yang dimiliki. Fitur yang dites dengan *data test* ini ialah hasil *output* dan tampilan yang muncul ketika *command* ini dipanggil. Tujuannya untuk memastikan *command* `PLAYLIST ADD ALBUM` dapat menambahkan album ke dalam *playlist* serta memastikan tidak dapat berjalan dengan baik jika *input* tidak sesuai. *Output* yang dikeluarkan akan seperti di Gambar 6.7.2.2.1 jika *input* benar dan valid dan akan seperti Gambar 6.7.2.2.2 jika pengguna memanggil *command* ini dengan *input* yang tidak tepat dan tidak valid, sehingga program akan menyuruh pengguna untuk mencoba lagi.

```
>>> PLAYLIST ADD ALBUM;
Daftar Penyanyi :
1. BLACKPINK
2. Arctic Monkeys
Masukkan Nama Penyanyi yang dipilih : BLACKPINK;
Daftar album oleh BLACKPINK :
1. BORN PINK
2. THE ALBUM
Masukkan Nama Album yang dipilih : BORN PINK;
Daftar Playlist :
1. K-POP ENJOYER
2. ASIK JOS
3. On Repeat Daily
Masukkan ID Playlist yang dipilih : 1;
Album dengan judul 'BORN PINK' telah ditambahkan ke dalam playlist pengguna 'K-POP ENJOYER'.
```

Gambar 6.7.2.2.1

```
>>> PLAYLIST ADD ALBUM;
Daftar Penyanyi :
1. BLACKPINK
2. Arctic Monkeys
Masukkan Nama Penyanyi yang dipilih : BLACKPINK;
Daftar album oleh BLACKPINK :
1. BORN PINK
2. THE ALBUM
Masukkan Nama Album yang dipilih : BORN PI;
Album BORN PI tidak ada dalam daftar, silakan coba lagi
```

Gambar 6.7.2.2.2

### 6.7.3 PLAYLIST SWAP <id> <x> <y>

Command `PLAYLIST SWAP <id> <x> <y>` akan menukar lagu pada urutan ke-x dan juga urutan ke-y di *playlist* dengan urutan ke *id*. Fitur yang dites dengan *data test* ini ialah hasil *output* ketika *command* ini berhasil dipanggil. Tujuannya untuk memastikan *command* ini dapat menukar lagu pada urutan ke-x dan urutan ke-y di *playlist* dengan urutan ke-*id* serta berfungsi dengan baik. *Output* yang dikeluarkan akan seperti di Gambar 6.7.3.1 jika pengguna berhasil memanggil *command* ini dengan *input* *id*, *x*, dan *y* yang sesuai. *Output* yang dikeluarkan akan seperti di Gambar 6.7.3.2 jika pengguna memanggil *command* ini dengan *input* *id* yang kurang tepat. *Output* yang dikeluarkan akan seperti di Gambar 6.7.3.3 jika pengguna memanggil *command* ini dengan *input* urutan yang kurang tepat, yakni tidak ada lagu dengan urutan nomor berikut di *playlist* yang dipilih dan dimiliki.

```
>>> PLAYLIST SWAP 1 1 2;
Berhasil menukar lagu dengan nama "Pink Venom" dengan "Shut Down" di playlist "K-POP ENJOYER"
```

Gambar 6.7.3.1

```
>>> PLAYLIST SWAP 500 1 2;
Tidak ada playlist dengan playlist ID 500
```

Gambar 6.7.3.2



```
>>> PLAYLIST SWAP 1 900 2;
Tidak ada lagu dengan urutan 900 di playlist 'K-POP ENJOYER'
```

Gambar 6.7.3.3

#### 6.7.4 PLAYLIST REMOVE <id> <n>

Command `PLAYLIST REMOVE <id> <n>` akan menghapus lagu dengan urutan *n* pada *playlist* dengan *index id*. Fitur yang dites dengan *data test* ini ialah hasil *output* ketika *command* ini dipanggil. Tujuannya untuk memastikan *command* ini dapat menghapus lagu dengan urutan *n* pada *playlist* dengan *index id* serta dapat berfungsi dengan baik. *Output* yang dikeluarkan akan seperti di Gambar 6.7.4.1 jika pengguna berhasil memanggil *command* ini dengan *input* yang benar dan tepat serta sesuai. *Output* yang dikeluarkan akan seperti di Gambar 6.7.4.2 jika pengguna memanggil *command* ini dengan *input* yang kurang sesuai yaitu tidak ada lagu dengan urutan nomor berikut dari *id playlist* yang dipilih. *Output* yang dikeluarkan akan seperti di Gambar 6.7.4.3 jika tidak ada *playlist* dengan *id* yang di-*input* berikut.

```
>>> PLAYLIST REMOVE 1 2;
Lagu 'Shut Down' oleh 'BLACKPINK' telah dihapus dari playlist 'K-POP ENJOYER'
```

Gambar 6.7.4.1

```
>>> PLAYLIST REMOVE 2 4;
Tidak ada lagu dengan urutan 4 di playlist 'ASIK JOS'
```

Gambar 6.7.4.2

```
>>> PLAYLIST REMOVE 100 2;
Tidak ada playlist dengan playlist ID 100
```

Gambar 6.7.4.3

#### 6.7.5 PLAYLIST DELETE

Command `PLAYLIST DELETE` akan menghapus *existing playlist* dalam daftar *playlist* pengguna. Fitur yang dites dengan *data test* ini ialah hasil *output* ketika *command* ini dapat dipanggil. Tujuannya untuk memastikan *command* ini dapat berhasil menghapus *existing playlist* dalam daftar *playlist* serta berfungsi dengan baik seperti di Gambar 6.7.5.1. *Output* akan seperti Gambar 6.7.5.2 jika *input id playlist* tidak sesuai.

```
>>> PLAYLIST DELETE;
Daftar Playlist :
1. K-POP ENJOYER
2. ASIK JOS
3. On Repeat Daily
Masukkan ID Playlist yang dipilih : 3;
Playlist ID 3 dengan judul 'On Repeat Daily' berhasil dihapus.
```

Gambar 6.7.5.1

```
Playlist ID 3 dengan judul 'On Repeat Daily' berhasil dihapus.
>>> PLAYLIST DELETE;
Daftar Playlist :
1. K-POP ENJOYER
2. ASIK JOS
Masukkan ID Playlist yang dipilih : 5;
ID 5 tidak ada dalam daftar, silakan coba lagi
```

Gambar 6.7.5.2



## 6.8 STATUS

*Command* STATUS akan menampilkan lagu yang sedang dimainkan beserta *queue song* yang ada dan dari *playlist* mana lagunya diputar. Fitur yang dites dengan *data test* ini ialah hasil *output* dan tampilan ketika *command* STATUS dapat dipanggil. Tujuannya untuk memastikan *command* ini dapat berjalan dengan baik. Output akan seperti Gambar 6.8.1 jika tidak ada lagu yang diputar dan queue kosong. Output akan seperti Gambar 6.8.2 jika terdapat lagu tetapi queue kosong. Output akan seperti 6.8.3 jika ada lagu dan ada queue. Output akan seperti Gambar 6.8.4 jika ada lagu, ada queue, dan playlist.

```
>>> STATUS;

Now Playing:
No songs have been played yet. Please search for a song to begin playback.

Queue:
Your queue is empty.
```

Gambar 6.8.1

```
>>> STATUS;

Now Playing:
BLACKPINK - BORN PINK - Shut Down

Queue:
Your queue is empty.
```

Gambar 6.8.2

```
>>> STATUS;

Now Playing:
BLACKPINK - BORN PINK - Ready For Love

Queue:
1. BLACKPINK - BORN PINK - Typa Girl
2. Arctic Monkeys - Humbug - Secret Door
3. Arctic Monkeys - AM - Do I Wanna Know ?
4. BLACKPINK - THE ALBUM - How You Like That
```

Gambar 6.8.3

```
>>> STATUS;

Current Playlist: BLACKPINK My Top Three

Now Playing:
BLACKPINK - THE ALBUM - How You Like That

Queue:
1. BLACKPINK - BORN PINK - Typa Girl
2. BLACKPINK - BORN PINK - Pink Venom
```

Gambar 6.8.4

## 6.9 SAVE <filename>

*Command* SAVE <filename> akan menyimpan *state* aplikasi terbaru ke dalam suatu *file* data dan memiliki satu argumen yang merepresentasikan nama *file* yang akan disimpan di *folder* config. Fitur yang dites dengan *data test* ini ialah hasil *output* dan tampilan ketika *command* SAVE dapat dipanggil. Tujuannya untuk menguji apakah dapat menyimpan *file* dan datanya. Output akan seperti Gambar 6.9.1 jika *command* SAVE berhasil dipanggil dan file pun akan tersimpan.

```
>>> SAVE savefile.txt;
Save file: savefile.txt berhasil disimpan!
```

Gambar 6.9.1

## 6.10 QUIT

*Command* QUIT akan mengeluarkan pengguna dari aplikasi. Fitur yang dites dengan *data test* ini ialah hasil *output* dan tampilan ketika *command* QUIT berhasil dipanggil. Tujuannya untuk menguji apakah dapat mengeluarkan program dan dapat menyimpan atau tidak menyimpan data sesi sekarang serta menguji apakah *command* ini dapat berfungsi jika input tidak valid. Output akan seperti Gambar 6.10.1 jika *command* QUIT berhasil dipanggil dan memilih untuk menyimpan data sesi sehingga file pun akan tersimpan. Output akan seperti Gambar 6.10.2 jika *command* QUIT berhasil dipanggil dan pengguna memilih untuk tidak menyimpan data sesi sehingga akan langsung keluar dari program. Output akan seperti Gambar 6.10.3 jika *command* QUIT berhasil dipanggil tetapi pengguna memasukkan input yang tidak valid sehingga program akan meminta pengguna memasukkan input sampai valid untuk diproses.

```
>>> QUIT;

Apakah kamu ingin menyimpan data sesi sekarang (Y/N)? Y;
Masukkan nama file penyimpanan: save.txt;
Save file: save.txt berhasil disimpan!

Data sesi sekarang milik kamu telah berhasil disimpan.
Thank you for using WayangWave :D
```

Gambar 6.10.1

```
>>> QUIT;

Apakah kamu ingin menyimpan data sesi sekarang (Y/N)? N;

Kamu keluar dari WayangWave.
Dadah ^_^/
```

Gambar 6.10.2

```
>>> QUIT;

Apakah kamu ingin menyimpan data sesi sekarang (Y/N)? U;
Input tidak valid. Silakan memasukkan 'Y' jika ingin menyimpan data sesi sekarang dan 'N' jika tidak ingin menyimpan data sesi sekarang (Y/N): A;
Input tidak valid. Silakan memasukkan 'Y' jika ingin menyimpan data sesi sekarang dan 'N' jika tidak ingin menyimpan data sesi sekarang (Y/N):
```

Gambar 6.10.3

## 6.11 HELP

*Command* HELP akan menampilkan *output* pada layar *user* yang berisi daftar *command* yang dapat dilakukan berdasarkan kondisinya. Fitur yang dites dengan *data test* ini ialah hasil *output* ketika *command* HELP berhasil dipanggil. Tujuannya untuk membantu *user* dalam memberi *input command* yang tersedia. Output akan seperti Gambar 6.11.1 jika *command* HELP berhasil dipanggil dengan kondisi belum me-load *save file* apapun maupun belum memanggil *command* START. Sedangkan *Output* seperti Gambar 6.11.2 ditampilkan saat *user* sudah me-load *save file* atau memanggil *command* START.



```
>>> START;
starting game..
initialize..
Berhasil masuk kedalam aplikasi, selamat menikmati!
>>> TUBES ALSTRUKDAT;
Command tidak diketahui, masukkan command "HELP;" untuk melihat list command
```

Gambar 6.12.2

```
>>> START;
starting game..
initialize..
Berhasil masuk kedalam aplikasi, selamat menikmati!
>>> LOAD filekonfigurasi.txt;
Command tidak bisa dieksekusi!
```

Gambar 6.12.3

## 6.13 ENHANCE

Command ENHANCE menerima masukan nama playlist yang akan di-enhance lalu menampilkan daftar playlist dan tambahan lagu rekomendasi random dari penyanyi random dan album random dan ditampilkan proses pemilihan lagu random yang ingin dimasukkan ke dalam playlist. Fitur yang dites dengan *data test* ini ialah hasil *output command* ini ketika berhasil dipanggil. Tujuannya untuk memastikan command ini dapat berjalan dengan baik. Output akan seperti Gambar 6.13.1 jika pengguna tidak ada playlist dan akan seperti Gambar 6.13.2 jika pengguna memiliki playlist dan memasukkan input id playlist yang sesuai sehingga lagu dapat ditambahkan/di-enhance.

```
>>> ENHANCE;

Playlist kosong, tidak ada yang bisa dienhance
```

Gambar 6.13.1

```
>>> ENHANCE;

Silakan pilih playlist yang akan dienhance.
Daftar Playlist :
1. AIUEO
2. DANCING~~

Masukkan ID Playlist yang dipilih : 1;

Lagu yang ditambahkan ke playlist 'AIUEO' adalah
1. BLACKPINK - BORN PINK - Shut Down
2. Arctic Monkeys - Favourite Worst Nightmare - Old Yellow Bricks
```

Gambar 6.13.2

## 7 Test Script

No.	Fitur yang Dites	Tujuan Testing	Langkah-Langkah Testing	Input Data Test	Hasil yang Diharapkan	Hasil yang Keluar
1	START	Menguji apakah command START dapat menjalankan keseluruhan program WayangWave.	Memasukkan command START ketika program berhasil di-compile.	Gambar 6.1	Gambar 6.1	Sesuai dengan hasil yang diharapkan.

2	START	Menguji apakah command START gagal menjalankan keseluruhan program WayangWave jika input salah.	Memasukkan command START yang salah ketika program berhasil di-compile.	Gambar 6.1	Gambar 6.1	Sesuai dengan hasil yang diharapkan.
3	LOAD <filename>	Menguji apakah command LOAD dapat membaca file dan berfungsi dengan baik.	Memasukkan command LOAD diikuti nama file ketika program berhasil di-compile.	Gambar 6.2	Gambar 6.2	Sesuai dengan hasil yang diharapkan.
4	LIST DEFAULT	Memastikan command LIST DEFAULT dapat menampilkan list penyanyi, album, dan lagu yang tersedia.	Memasukkan command LIST DEFAULT ketika program dijalankan.	Gambar 6.3.1	Gambar 6.3.1	Sesuai dengan hasil yang diharapkan.
5	LIST PLAYLIST	Memastikan command LIST PLAYLIST dapat menampilkan list playlist yang dimiliki.	Memasukkan command LIST PLAYLIST ketika program dijalankan.	Gambar 6.3.2	Gambar 6.3.2	Sesuai dengan hasil yang diharapkan.
6	PLAY SONG	Memastikan command PLAY SONG dapat memainkan lagu berdasarkan input nama penyanyi, nama album, dan id lagu yang sesuai.	Memasukkan command PLAY SONG ketika program dijalankan.	Gambar 6.4.1.1	Gambar 6.4.1.1	Sesuai dengan hasil yang diharapkan.
7	PLAY SONG	Memastikan command PLAY SONG tidak dapat memainkan lagu berdasarkan input nama penyanyi, nama album, dan id lagu yang tidak sesuai.	Memasukkan command PLAY SONG dan input yang tidak sesuai.	Gambar 6.4.1.2	Gambar 6.4.1.2	Sesuai dengan hasil yang diharapkan.
8	PLAY PLAYLIST	Memastikan command PLAY PLAYLIST dapat memainkan lagu berdasarkan id playlist yang diinput.	Memasukkan command PLAY PLAYLIST ketika program dijalankan.	Gambar 6.4.2.3	Gambar 6.4.2.3	Sesuai dengan hasil yang diharapkan.
9	PLAY PLAYLIST	Memastikan command PLAY PLAYLIST tidak dapat memainkan lagu jika playlist masih kosong.	Memasukkan command PLAY PLAYLIST ketika program dijalankan.	Gambar 6.4.2.1	Gambar 6.4.2.1	Sesuai dengan hasil yang diharapkan.
8	QUEUE SONG	Memastikan command QUEUE SONG dapat menambahkan lagu ke dalam antrian.	Memasukkan command QUEUE SONG ketika program dijalankan.	Gambar 6.5.1.2	Gambar 6.5.1.2	Sesuai dengan hasil yang diharapkan.

9	QUEUE SONG	Memastikan command QUEUE SONG tidak dapat berjalan jika input tidak sesuai.	Memasukkan command QUEUE SONG dan input yang tidak sesuai.	Gambar 6.5.1.1	Gambar 6.5.1.1	Sesuai dengan hasil yang diharapkan.
10	QUEUE PLAYLIST	Memastikan command QUEUE PLAYLIST dapat menambahkan lagu yang ada dalam playlist ke dalam antrian.	Memasukkan command QUEUE PLAYLIST ketika program dijalankan	Gambar 6.5.2.2	Gambar 6.5.2.2	Sesuai dengan hasil yang diharapkan.
10	QUEUE PLAYLIST	Memastikan command QUEUE PLAYLIST tidak dapat berjalan jika input id playlist tidak sesuai.	Memasukkan command QUEUE PLAYLIST dan input yang tidak sesuai.	Gambar 6.5.2.3	Gambar 6.5.2.3	Sesuai dengan hasil yang diharapkan.
11	QUEUE SWAP <x> <y>	Memastikan command QUEUE SWAP dapat menukar lagu pada urutan ke x dan juga urutan ke y dan dapat berfungsi dengan baik.	Memasukkan command QUEUE SWAP ketika program dijalankan.	Gambar 6.5.3	Gambar 6.5.3	Sesuai dengan hasil yang diharapkan.
11	QUEUE REMOVE <id>	Memastikan command QUEUE REMOVE dapat menghapus lagu dari antrian sesuai dengan input urutan lagu (id).	Memasukkan command QUEUE REMOVE ketika program dijalankan.	Gambar 6.5.4	Gambar 6.5.4	Sesuai dengan hasil yang diharapkan.
12	QUEUE CLEAR	Memastikan command QUEUE CLEAR dapat mengosongkan antrian.	Memasukkan command QUEUE CLEAR ketika program dijalankan.	Gambar 6.5.5.1	Gambar 6.5.5.1	Sesuai dengan hasil yang diharapkan.
13	SONG NEXT	Memastikan command SONG NEXT dapat memutar lagu yang berada di dalam antrian.	Memasukkan command SONG NEXT ketika program dijalankan.	Gambar 6.6.1.3	Gambar 6.6.1.3	Sesuai dengan hasil yang diharapkan.
14	SONG NEXT	Memastikan command SONG NEXT dapat memutar lagu yang sedang diputar saat ini jika antrian kosong.	Memasukkan command SONG NEXT ketika program dijalankan dan dalam keadaan queue (antrian) kosong.	Gambar 6.6.1.2	Gambar 6.6.1.2	Sesuai dengan hasil yang diharapkan.
15	SONG PREVIOUS	Memastikan command SONG PREVIOUS dapat memutar lagu yang terakhir kali diputar.	Memasukkan command SONG PREVIOUS ketika program dijalankan.	Gambar 6.6.2.2	Gambar 6.6.2.2	Sesuai dengan hasil yang diharapkan.

16	SONG PREVIOUS	Memastikan command SONG PREVIOUS dapat memutar lagu yang saat ini sedang diputar ketika riwayat lagu kosong.	Memasukkan command SONG PREVIOUS ketika program dijalankan dan dalam keadaan riwayat lagu kosong.	Gambar 6.6.2.1	Gambar 6.6.2.1	Sesuai dengan hasil yang diharapkan.
17	PLAYLIST CREATE	Memastikan command PLAYLIST CREATE dapat membuat playlist baru dan ditambahkan pada daftar playlist pengguna.	Memasukkan command PLAYLIST CREATE ketika program dijalankan.	Gambar 6.7.1.1 & Gambar 6.7.1.2	Gambar 6.7.1.1 & Gambar 6.7.1.2	Sesuai dengan hasil yang diharapkan.
18	PLAYLIST CREATE	Memastikan command PLAYLIST CREATE tidak dapat berjalan dengan baik jika input tidak valid.	Memasukkan command PLAYLIST CREATE dan input yang tidak valid.	Gambar 6.7.1.3 & Gambar 6.7.1.4	Gambar 6.7.1.3 & Gambar 6.7.1.4	Sesuai dengan hasil yang diharapkan.
19	PLAYLIST ADD SONG	Memastikan command PLAYLIST ADD SONG dapat menambahkan lagu ke dalam playlist.	Memasukkan command PLAYLIST ADD SONG ketika program dijalankan.	Gambar 6.7.2.1.1	Gambar 6.7.2.1.1	Sesuai dengan hasil yang diharapkan.
20	PLAYLIST ADD SONG	Memastikan command PLAYLIST ADD SONG tidak dapat berjalan dengan baik jika input tidak sesuai.	Memasukkan command PLAYLIST ADD SONG dan input yang tidak sesuai.	Gambar 6.7.2.1.2	Gambar 6.7.2.1.2	Sesuai dengan hasil yang diharapkan.
21	PLAYLIST ADD ALBUM	Memastikan command PLAYLIST ADD ALBUM dapat menambahkan album ke dalam playlist.	Memasukkan command PLAYLIST ADD ALBUM ketika program dijalankan.	Gambar 6.7.2.2.1	Gambar 6.7.2.2.1	Sesuai dengan hasil yang diharapkan.
22	PLAYLIST ADD ALBUM	Memastikan command PLAYLIST ADD ALBUM tidak dapat berjalan dengan baik jika input tidak sesuai.	Memasukkan command PLAYLIST ADD ALBUM dan input yang tidak sesuai.	Gambar 6.7.2.2.2	Gambar 6.7.2.2.2	Sesuai dengan hasil yang diharapkan.
23	PLAYLIST SWAP <id> <x> <y>	Memastikan command PLAYLIST SWAP dapat menukar lagu pada urutan ke x dan juga urutan ke y di playlist dengan urutan ke id dan dapat berfungsi dengan baik.	Memasukkan command PLAYLIST SWAP ketika program dijalankan.	Gambar 6.7.3	Gambar 6.7.3	Sesuai dengan hasil yang diharapkan.
24	PLAYLIST REMOVE <id> <n>	Memastikan command PLAYLIST REMOVE dapat menghapus lagu	Memasukkan command PLAYLIST	Gambar 6.7.4	Gambar 6.7.4	Sesuai dengan

		dengan urutan n pada playlist dengan index id dan dapat berfungsi dengan baik.	REMOVE ketika program dijalankan.			hasil yang diharapkan.
25	PLAYLIST DELETE	Memastikan command PLAYLIST DELETE dapat menghapus suatu <i>existing</i> playlist dalam daftar playlist dan dapat berfungsi dengan baik.	Memasukkan command PLAYLIST DELETE ketika program dijalankan.	Gambar 6.7.5	Gambar 6.7.5	Sesuai dengan hasil yang diharapkan.
26	STATUS	Memastikan command STATUS dapat berjalan dengan baik.	Memasukkan command STATUS ketika program dijalankan.	Gambar 6.8	Gambar 6.8	Sesuai dengan hasil yang diharapkan.
27	SAVE <filename>	Menguji apakah command SAVE dapat menyimpan state aplikasi terbaru ke dalam suatu file.	Memasukkan command SAVE ketika program dijalankan.	Gambar 6.9.1	Gambar 6.9.1	Sesuai dengan hasil yang diharapkan.
28	QUIT	Menguji apakah command QUIT dapat mengeluarkan program dan menyimpan data sesi sekarang.	Memasukkan command QUIT lalu memilih untuk menyimpan data sesi sekarang.	Gambar 6.10.1	Gambar 6.10.1	Sesuai dengan hasil yang diharapkan.
29	QUIT	Menguji apakah command QUIT dapat mengeluarkan program tanpa menyimpan data sesi sekarang.	Memasukkan command QUIT lalu memilih untuk tidak menyimpan data sesi sekarang.	Gambar 6.10.2	Gambar 6.10.2	Sesuai dengan hasil yang diharapkan.
30	QUIT	Menguji apakah command QUIT tidak dapat mengeluarkan program jika input tidak sesuai.	Memasukkan command QUIT dan input yang tidak sesuai.	Gambar 6.10.3	Gambar 6.10.3	Sesuai dengan hasil yang diharapkan.
31	HELP	Memastikan command HELP dapat berjalan dengan baik ketika dipanggil sebelum memasuki sesi.	Memasukkan command HELP saat sebelum memasuki sesi program.	Gambar 6.11.1	Gambar 6.11.1	Sesuai dengan hasil yang diharapkan.
32	HELP	Memastikan command HELP dapat berjalan dengan baik ketika dipanggil setelah memasuki sesi.	Memasukkan command HELP saat setelah memasuki sesi program.	Gambar 6.11.2	Gambar 6.11.2	Sesuai dengan hasil yang diharapkan.
33	<INVALID COMMAND>	Memastikan <INVALID COMMAND> dapat berfungsi dengan baik dan sesuai.	Memasukkan command yang salah dan tidak tepat ketika program dijalankan.	Gambar 6.12.1, Gambar 6.12.2, Gambar 6.12.3	Gambar 6.12.1, Gambar 6.12.2, Gambar 6.12.3	Sesuai dengan hasil yang diharapkan.



34	ENHANCE	Memastikan command ENHANCE dapat menampilkan daftar playlist dengan tambahan lagu secara <i>random</i> dari penyanyi dan album <i>random</i> .	Memasukkan command ENHANCE ketika program dijalankan.	Gambar 6.13.1, Gambar 6.13.2	Gambar 6.13.1, Gambar 6.13.2	Sesuai dengan hasil yang diharapkan.
----	---------	--	---	------------------------------	------------------------------	--------------------------------------

## 8 Pembagian Kerja dalam Kelompok

Berikut pembagian kerja setiap anggota dalam kelompok kami.

Nama lengkap / NIM	Deskripsi tugas
Muhammad Rifa Ansyari / 18222004	Struktur data, Command Start, Main, Membantu Debugging.
Irfan Musthofa / 18222056	Load, Save, Modify ADT Mesin Kata, Melengkapi Laporan, Membantu Debugging.
Taufiq Ramadhan Ahmad / 18222060	ADT Stack, ADT Queue, Command Queue, Command Play, Command Song.
Kerlyn Deslia Andeskar / 18222090	Quit, Status, List Default, List Playlist, Invalid Command, Help, ReadMe, ADT Set, ADT Map, Membuat Laporan (Ringkasan, Sketsa Struktur Data ADT, Program Utama, Data Test, Test Script, Lampiran Deskripsi Tubes & Notulen Rapat).
Farah Aulia / 18222096	ADT Mesin Kata, ADT Playlist, ADT List, Command Playlist, Bonus Enhance, Command Pick, Melengkapi Laporan, Membantu Debugging.

## 9 Lampiran

### 9.1 Deskripsi Tugas Besar

#### Spesifikasi Umum

Buatlah sebuah aplikasi simulasi berbasis CLI (command-line interface). Sistem ini dibuat dalam **bahasa C** dengan menggunakan **struktur data yang sudah kalian pelajari** di mata kuliah ini. Kalian boleh menggunakan (atau memodifikasi) struktur data yang sudah kalian buat untuk praktikum pada tugas besar ini. Daftar ADT yang wajib digunakan dapat dilihat

pada bagian [Daftar ADT](#). Library yang boleh digunakan hanya **stdio.h**, **stdlib.h**, **time.h**, dan **math.h**

## System Mechanic

### 1. About the System

WayangWave merupakan sebuah aplikasi yang bisa mensimulasikan *service* pemutaran musik. WayangWave ini memiliki beberapa fitur utama, yaitu:

1. Memutar lagu
2. Menampilkan daftar lagu
3. Membuat dan menghapus *playlist*
4. Mengatur urutan dimainkannya lagu
5. Menampilkan status dari aplikasi

### 2. Main Menu

Ketika program pertama kali dijalankan, WayangWave akan memperlihatkan main menu yang berisi *welcome page* dan beberapa command yaitu **START**, **LOAD**, dan juga **HELP**. Setelah itu, *main menu* akan menerima masukan berupa command yang akan dijelaskan pada bagian berikutnya.

### 3. Command

- a. **START**: Salah satu command yang dimasukkan pertama kali dalam WayangWave. Setelah menekan Enter, dibaca file konfigurasi default yang berisi daftar penyanyi serta album yang dimiliki.
- b. **LOAD <filename>**: Salah satu command yang dimasukkan pertama kali dalam WayangWave. Command ini memiliki satu argumen yaitu filename yang merepresentasikan suatu *save file* yang ingin dibuka. *File* didapatkan dari folder tertentu, contohnya *save*. Setelah menekan Enter, akan dibaca *save file <filename>* yang berisi list penyanyi, album, dan lagu yang bisa diputar. Lebih detailnya bisa dilihat pada [Konfigurasi Aplikasi](#).
- c. **LIST**: Command yang digunakan untuk menampilkan list playlist yang ada, list penyanyi, list album dari penyanyi, dan list lagu yang ada di album. Terdapat dua jenis list, **DEFAULT** dan **PLAYLIST**.
  1. **LIST DEFAULT**: Untuk melihat list penyanyi yang ada. Selanjutnya dapat memilih untuk melihat album dari penyanyi yang dipilih. Kemudian melihat lagu yang ada dari album yang dipilih. Terdapat konfirmasi apakah ingin melihat album/lagu.
  2. **LIST PLAYLIST**: Untuk menampilkan playlist yang ada pada pengguna.
- d. **PLAY**: Command yang digunakan untuk memutar lagu atau playlist yang dipilih. Ketika command **PLAY** dieksekusi, queue yang ada dihapus ketika memainkan lagu atau digantikan oleh lagu dalam playlist ketika memainkan playlist. Terdapat dua jenis play, **SONG** dan **PLAYLIST**.
  1. **PLAY SONG**: Untuk memainkan lagu berdasarkan masukan nama penyanyi, nama album, dan id lagu. Ketika command ini berhasil dieksekusi, queue dan riwayat lagu akan menjadi kosong.
  2. **PLAY PLAYLIST**: Untuk memainkan lagu berdasarkan id playlist. Ketika command ini berhasil dieksekusi, current song akan menjadi lagu

pada urutan pertama playlist dan queue akan berisi semua lagu yang ada dalam playlist yang akan dimainkan dan isi riwayat lagu sama dengan queue, tetapi dengan urutan yang di-*reverse*.

- e. **QUEUE**: command yang digunakan untuk memanipulasi queue lagu. Command ini memiliki 5 tipe, yaitu **SONG**, **PLAYLIST**, **SWAP**, **REMOVE**, dan **CLEAR**.
  - 1. **QUEUE SONG**: Untuk menambahkan lagu ke dalam queue. Command ini menerima input lagu berdasarkan nama penyanyi, nama album, dan id dari lagu yang ingin dimasukkan ke dalam queue.
  - 2. **QUEUE PLAYLIST**: Untuk menambahkan lagu yang ada dalam playlist ke dalam queue. Command ini menerima input dari id playlist yang ingin dimasukkan ke dalam queue.
  - 3. **QUEUE SWAP <x> <y>**: Untuk menukar lagu pada urutan ke **x** dan juga urutan ke **y**.
  - 4. **QUEUE REMOVE <id>**: Untuk menghapus lagu dari queue. Command ini menerima input berupa urutan lagu (**id**) yang ingin dihapus dari queue.
  - 5. **QUEUE CLEAR**: Untuk mengosongkan queue.
- f. **SONG**: Command yang digunakan untuk navigasi lagu yang ada pada queue lagu saat ini. Terdapat 2 tipe navigasi yaitu **NEXT** dan **PREVIOUS**.
  - 1. **SONG NEXT**: Untuk memutar lagu yang berada di dalam queue. Lagu yang sedang diputar kemudian ditambah ke dalam daftar riwayat putar lagu. Jika queue kosong, yang diputar adalah lagu yang sedang diputar.
  - 2. **SONG PREVIOUS**: Untuk memutar lagu yang terakhir kali diputar. Lagu yang sedang diputar kemudian ditambah ke dalam queue dengan urutan pertama. Jika daftar riwayat lagu kosong, yang diputar adalah lagu yang sedang diputar.
- g. **PLAYLIST**: untuk melakukan basic command untuk playlist yaitu **CREATE**, **ADD**, **SWAP**, **REMOVE** dan **DELETE**.
  - 1. **PLAYLIST CREATE**: Untuk membuat playlist baru dan ditambahkan pada daftar playlist pengguna. Keadaan awal playlist adalah kosong. Nama playlist dapat sama dengan playlist yang sudah ada.
  - 2. **PLAYLIST ADD**: Untuk menambahkan lagu pada suatu playlist yang telah ada sebelumnya pada daftar playlist pengguna. Pada defaultnya, command ini hanya dapat menambahkan satu spesifik lagu atau semua lagu yang ada pada album kepada suatu *existing* playlist. Apabila lagu atau lagu - lagu di dalam album yang ingin ditambahkan sudah ada di dalam suatu playlist pengguna maka lagu - lagu yang ditambahkan adalah yang belum ada di playlist pengguna. Tampilkan pesan *error* apabila masukkan pengguna tidak valid pada setiap permintaan masukkan.
  - 3. **PLAYLIST SWAP <id> <x> <y>**: Untuk menukar lagu pada urutan ke **x** dan juga urutan ke **y** di playlist dengan urutan ke **id**.

4. **PLAYLIST DELETE:** Untuk melakukan penghapusan suatu *existing* playlist dalam daftar playlist pengguna. Tampilkan pesan *error* apabila masukkan pengguna tidak valid pada setiap permintaan masukkan.
- h. **STATUS:** Command yang digunakan untuk menampilkan lagu yang sedang dimainkan beserta Queue song yang ada dan dari playlist mana lagu itu diputar.
- i. **SAVE <filename>:** Command yang digunakan untuk menyimpan state aplikasi terbaru ke dalam suatu file. Command **SAVE** memiliki satu argumen yang merepresentasikan nama file yang akan disimpan. Penyimpanan dilakukan pada folder tertentu, misal folder save.
- j. **QUIT:** Command yang digunakan untuk keluar dari aplikasi WayangWave.
- k. **HELP:** Menampilkan daftar command yang mungkin untuk dieksekusi dengan deskripsinya. Penjelasan dari deskripsi dibebaskan selama masih mendeskripsikan command sesuai dengan spek.
- l. **<INVALID COMMAND>:** Command-command selain yang disebutkan di atas dinyatakan akan tidak valid dan hanya akan mengeluarkan teks error.

#### **BONUS**

1. **ENHANCE:** Fitur tambahan berupa command **ENHANCE** yang akan menerima masukkan nama playlist yang akan di-enhance. Lalu, ditampilkan daftar playlist dengan tambahan lagu rekomendasi random dari penyanyi random dan album random. Lalu, ditampilkan proses pemilihan lagu random yang ingin dimasukkan ke dalam playlist. Format command untuk menerima masukan dan menghasilkan keluaran dibebaskan selama memenuhi ketentuan di atas. Algoritma random diimplementasikan terlebih dahulu sebelum membuat fitur keseluruhan.
2. **Multi User:** Fitur tambahan multi user mengharuskan Anda melakukan **LOGIN** saat aplikasi dimulai. Setiap user memiliki queue, history, dan playlist yang berbeda sehingga saat melakukan save Anda harus menyimpan state sesuai user yang login. Selain itu, ada command **LOGOUT** untuk keluar dari akun user dan login ke akun user yang lain.
3. **Custom Fitur:** Membuat Custom Fitur sesuai dengan kreatifitas Anda dengan menggunakan **ADT Tree** atau **Graph**.

## **9.2 Notulen Rapat**

**Form Asistensi Tugas Besar  
IF2111/Algoritma dan Struktur Data STI  
Sem. 1 2023/2024**

No. Kelompok/Kelas : 10/K02  
 Nama Kelompok : A Magic Teapot  
 Anggota Kelompok (Nama/NIM) :  
     1. Muhammad Rifa Ansyari/18222004  
     2. Qady Zakka Raymaula/18222038  
     3. Irfan Musthofa/18222056






<b>STEI- ITB</b>	<b>IF2111-TB-02-10</b>	<b>Halaman 44 dari 48 halaman</b>
Template dokumen ini dan informasi yang dimilikinya adalah milik Sekolah Teknik Elektro dan Informatika ITB dan bersifat rahasia. Dilarang me-reproduksi dokumen ini tanpa diketahui oleh Sekolah Teknik Elektro dan Informatika ITB.		


4. Taufiq Ramadhan Ahmad/18222060
5. Kerlyn Deslia Andeskar/18222090
6. Farah Aulia/18222096

Asisten Pembimbing

: Muhammad Davin Dzimar







#### Asistensi I

<b>Tanggal : 30 Oktober 2023</b>	<b>Catatan Asistensi:</b>  - Dijelaskan spesifikasi tugas besar yang akan dikerjakan - <i>List playlist</i> sebelum <i>load</i> = kosong - Kalau lagi <i>play song</i> , terus <i>play song</i> lagi, semua <i>history &amp; queue</i> jadi kosong
<b>Tempat : ITB</b>	
<b>Kehadiran Anggota Kelompok:</b>  <div style="text-align: center;"> 1 18222004  </div> <div style="text-align: center;"> 2 18222038   </div> <div style="text-align: center;"> 3 18222056  </div> <div style="text-align: center;"> 4 18222060  </div> <div style="text-align: center;"> 5 18222090  </div> <div style="text-align: center;"> 6 18222096  </div>	

	<b>Tanda Tangan Asisten:</b> 
--	--

## Asistensi II

<b>Tanggal :</b> 14 November 2023 <b>Tempat :</b> Google Meet	<b>Catatan Asistensi:</b> <ol style="list-style-type: none"> <li>1. <i>Struct</i> sudah ok dan mencerminkan perwujudan yang di-<i>expect</i>. Tapi balik lagi, kalau menemukan komplikasi ke depannya diperbaiki lagi dari sekarang biar mempermudah pekerjaan juga.</li> <li>2. ADT kalau bisa lebih optimal lagi implementasinya agar bisa memenuhi program keseluruhan. Tapi kalau misalnya menurut kalian ini sudah ok tidak apa, yang penting kedepannya tidak ada masalah. Perhatiin dan pintar-pintar mengaturnya jangan sampai menemukan masalah untuk nantinya.</li> <li>3. Karena <i>start</i> sudah, mungkin boleh langsung ke <i>load</i>.</li> <li>4. CurrentSong bukan masuk <i>history</i> dan antrian, dia menjabarkan apa yang sedang dimainkan. <i>History</i> berarti yang sudah selesai dimainkan. Antrian berarti yang akan dimainkan selanjutnya.</li> <li>5. Analisis apakah implementasi sekarang akan bermasalah nantinya karena akan lebih baik diperbaiki dari sekarang.</li> <li>6. Jangan terlalu fokus ke soal bonus itu sudah bagus Yang penting lengkapi fitur-fitur dasar dahulu.</li> <li>7. Progres sudah ok dan untuk masalah konfigurasi membaca <i>file</i> setidaknya sudah paham.</li> </ol>
--	--

<p><b>Kehadiran Anggota Kelompok:</b></p> <p>1 18222004 </p> <p>2 18222038</p> <p>3 18222056 </p> <p>4 18222060 </p> <p>5 18222090 </p> <p>6 18222096 </p>	
	<p><b>Tanda Tangan Asisten:</b></p> 

### 9.3 Log Activity Anggota Kelompok

Log Activities	Anggota kelompok yang terlibat	Tanggal
Meet offline 1 / diskusi konsep dan pembagian tugas	Semua anggota	1-November-2023
Push mesin kata	Farah Aulia / 18222096	10-November-2023
Buat struktur ADT, Push start	Muhammad Rifa A./ 18222004	12-November-2023
Push ADT Stack, queue	Taufiq Ramadhan / 18222060	10-November-2023
Meet offline 2 / membuat ADT dan diskusi struct	Semua anggota	11-November-2023
Push ADT list dan pick	Farah Aulia / 18222096	15-November-2023
Push playlist	Farah Aulia / 18222096	16-November-2023
Push help, status, L	Kerlyn Deslia A. / 18222090	16-November-2023
Push command play, song	Taufiq Ramadhan / 18222060	16-November-2023
Meet offline 3 / menyelesaikan seluruh command, fitur, dan debugging	Semua anggota	18-November-2023
Push queue	Taufiq Ramadhan / 18222060	18-November-2023
Push load	Irfan Musthofa / 18222056	19-November-2023
Push save, revisi, debugging	Muhammad Rifa Ansyari dan Irfan Musthofa	20-November-2023
Push quit, push ReadMe	Kerlyn Deslia A. / 18222090	20-November-2023, 22-November-2023
Push main	Muhammad Rifa A./ 18222004	21-November-2023
Push enhance (bonus)	Farah Aulia	21-November-2023
Debugging dan membuat laporan	Semua anggota	21-24 November 2023