

LAPORAN TUGAS BESAR
IF2111 Algoritma dan Struktur Data STI

WayangWave


Dipersiapkan oleh:

Kelompok 1

Bihurin Salsabila Firdaus	18222015
David Dewanto	18222027
Muhammad Rafi Dhiyaulhaq	18222069
Alvin Fadhillah Akmal	18222079
Timotius Vivaldi Gunawan	18222091

Sekolah Teknik Elektro dan Informatika - Institut Teknologi Bandung

Jl. Ganesha 10, Bandung 40132

	Sekolah Teknik Elektro dan Informatika ITB	Nomor Dokumen		Halaman
		<i>IF2111-TB-K-01-3</i>		<i><jml hlm></i>
		<i>Revisi</i>	<i><no revisi></i>	<i><Tgl release></i>

Daftar Isi

1 Ringkasan	5
2 Penjelasan Tambahan Spesifikasi Tugas	6
2.1 Enhance	6
2.2 Multiuser	6
2.3 Radio (Custom Feature)	7
3 Struktur Data (ADT)	7
3.1 Struktur Data Linked List	7
3.2 Struktur Data List	8
3.3 Struktur Data Map	9
3.4 Struktur Data Mesin Karakter	9
3.5 Struktur Data Mesin Kata	9
3.6 Struktur Data Queue	10
3.7 Struktur Data Set	10
3.8 Struktur Data Stack	11
3.10 Struktur Data NowPlaying	12
3.11 Struktur Data UserSession	13
4 Program Utama	13
5 Algoritma-Algoritma Menarik	14
5.1 Tausworthe Random Number Generator	14
6 Data Test	15
6.1 Data Test START	15
6.2 Data Test LOAD	16
6.3 Data Test LIST DEFAULT	16
6.4 Data Test LIST PLAYLIST	17
6.5 Data Test PLAY SONG	18
6.6 Data Test PLAY PLAYLIST	18
6.7 Data Test QUEUE SONG	19
6.8 Data Test QUEUE PLAYLIST	20
6.9 Data Test QUEUE SWAP	21
6.10 Data Test QUEUE REMOVE	21
6.11 Data Test QUEUE CLEAR	21
6.12 Data Test SONG NEXT	21
6.13 Data Test SONG PREVIOUS	21
6.14 Data Test PLAYLIST CREATE	22
6.15 Data Test PLAYLIST ADD	22
6.16 Data Test PLAYLIST SWAP	23
6.17 Data Test PLAYLIST REMOVE	24
6.18 Data Test PLAYLIST DELETE	24

6.19 Data Test STATUS	25
6.20 Data Test SAVE	25
6.21 Data Test QUIT	25
6.22 Data Test HELP	26
6.23 Data Test INVALID COMMAND	26
7 Test Script	27
8 Pembagian Kerja dalam Kelompok	32
9 Lampiran	33
9.1 Deskripsi Tugas Besar	33
9.2 Notulen Rapat	37
9.3 Log Activity Anggota Kelompok	41

1 Ringkasan

WayangWave adalah sebuah aplikasi pemutar musik layaknya *walkman*. WayangWave memiliki fitur utama yang dapat dilakukan oleh pengguna yaitu memutar lagu, menampilkan daftar lagu, membuat dan menghapus *playlist*, mengatur urutan dimainkannya lagu, menampilkan status dari aplikasi. Semua perubahan yang kita lakukan terhadap program pada saat memutar musik seperti menambahkan dan juga menghapus *playlist* dapat disimpan pada sebuah *file*. Berikut adalah penjelasan mengenai cara kerja dari program WayangWave ini.

Pada awal memulai program WayangWave, program akan menampilkan sebuah *main menu* yang berisi *welcome page*. Setelah itu, *main menu* akan menerima masukan berupa *command* START, LOAD, dan HELP. Pada fase tersebut, *command* START, LOAD, dan HELP adalah *command* pertama dari *command* yang harus diberikan oleh pengguna. Ketika pengguna memberikan *command* lain selain START, LOAD, dan HELP sebagai perintah pertama, program akan menampilkan bahwa masukan dari pengguna salah, akan diminta untuk memberikan perintah kembali sampai perintah yang diberikan oleh pengguna adalah diantara ketiga tersebut. Pada *command* START, setelah menekan Enter, akan dibaca *file* konfigurasi *default* yang berisi daftar penyanyi serta album yang dimiliki. *Command* LOAD memiliki satu argumen yaitu *filename* yang merepresentasikan suatu *save file* yang ingin dibuka. HELP *command* digunakan untuk menampilkan daftar *command* untuk dieksekusi dengan deskripsinya.

Saat pengguna telah memilih cara dalam memulai menjalankan program, pengguna dapat menjalankan fitur-fitur yang terdapat pada program WayangWave melalui pemanggilan *command*. Daftar *command* yang dapat diberikan adalah antara lain 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, PLAYLIST SWAP, PLAYLIST REMOVE, PLAYLIST DELETE, STATUS, SAVE, dan QUIT. Apabila pengguna memberikan *command* selain *command* yang telah disebutkan, maka masukan dari pengguna tersebut akan dianggap tidak *valid* dan tidak akan diproses oleh program. Penjelasan singkat mengenai *command* fitur pada program WayangWave adalah sebagai berikut.

Pertama, *Command* LIST adalah *command* untuk menampilkan *list playlist* yang ada. LIST DEFAULT adalah perintah yang digunakan untuk menampilkan *list* penyanyi yang ada yang kemudian dapat melihat album dari penyanyi yang dipilih. *Command* LIST PLAYLIST digunakan untuk menampilkan *playlist* yang ada pada pengguna. PLAY adalah perintah untuk memutar lagu atau *playlist* yang dipilih. PLAY SONG adalah perintah untuk memainkan lagu berdasarkan masukan nama penyanyi, nama album, dan id lagu. PLAY PLAYLIST adalah perintah untuk memainkan lagu berdasarkan ID Playlist.

Perintah selanjutnya adalah QUEUE. QUEUE adalah *command* untuk memanipulasi *queue* lagu. QUEUE SONG sebagai penambah lagu ke dalam *queue*. *Command* QUEUE PLAYLIST sebagai penambahan lagu yang ada dalam *playlist* ke dalam *queue*. QUEUE SWAP sebagai penukar lagu. QUEUE REMOVE sebagai penghapus lagu dari *queue* dan QUEUE CLEAR digunakan untuk mengosongkan *queue*.

Perintah SONG adalah sebagai menavigasi lagu yang ada pada *queue* lagu. Navigasi lagu tersebut dibagi menjadi 2 tipe navigasi yaitu SONG NEXT dan SONG PREVIOUS. Perintah SONG NEXT sebagai pemutar lagu yang berada di dalam *queue*. Sedangkan SONG PREVIOUS untuk memutar lagu yang terakhir kali diputar. Perintah PLAYLIST sebagai *basic command* untuk *playlist*. PLAYLIST CREATE untuk membuat *playlist* baru dan ditambahkan pada daftar *playlist* pengguna. PLAYLIST ADD digunakan untuk menambahkan lagu pada suatu *playlist* yang telah ada sebelumnya pada daftar *playlist* pengguna. *Command* PLAYLIST SWAP digunakan untuk menukar lagu dan *Command* PLAYLIST REMOVE digunakan untuk menghapus lagu sesuai urutan. Perintah PLAYLIST yang terakhir adalah PLAYLIST DELETE yang digunakan untuk melakukan penghapusan suatu *playlist*.

Perintah-perintah terakhir adalah STATUS, SAVE, QUIT, dan INVALID COMMAND. STATUS adalah perintah untuk menampilkan status lagu yang sedang dimainkan, SAVE untuk menyimpan *state* aplikasi terbaru ke dalam suatu *file*, QUIT untuk keluar dari aplikasi, dan INVALID COMMAND untuk menyatakan akan tidak *valid* suatu *command* dan hanya akan mengeluarkan teks error.

Program WayangWave ini menggunakan bahasa pemrograman C dan akan menerapkan beberapa struktur data yang sesuai. ADT yang digunakan pada program ini adalah ADT Linked-List, ADT List, ADT Map, ADT Mesin Karakter, ADT Mesin Kata, ADT Queue, ADT Set, dan ADT STACK. Hal lain mengenai ADT yang digunakan pada program ini akan dijelaskan dengan lebih *detail* pada poin nomor tiga. Pada program WayangWave yang kami buat juga terdapat beberapa fitur tambahan yang akan dijelaskan pada poin nomor dua. Selain itu, terdapat program utama atau *main* yang digunakan sebagai wadah sinkronisasi antar program *command* sehingga program WayangWave dapat dijalankan secara utuh dan selaras. Hal mengenai program utama akan dijelaskan secara lebih lanjut pada poin nomor empat, diikuti dengan *Data Test* dan juga *Test Script* mengenai program WayangWave secara keseluruhan yang akan dijabarkan pada poin nomor enam dan tujuh.

2 Penjelasan Tambahan Spesifikasi Tugas

2.1 Enhance

Fungsi *enhance* akan menerima sebuah input ID PLAYLIST yang dimiliki *user* dan akan diberikan beberapa rekomendasi lagu sehingga dapat diisi ke *playlist*. Untuk fungsi *random* kami implementasikan dengan algoritma Tausworthe RNG. Setelah didapatkan beberapa rekomendasi, user dapat memilih ID REKOMENDASI yang diinginkan dan akan di masukan ke *playlist* tersebut.

2.2 Multiuser

Pada tugas besar ini, kami menambahkan fungsi *multi user* dalam satu *file* dimana setiap *user* akan memiliki *session* masing-masing yang berisi *current song*, antrian, riwayat, dan daftar *playlist*. Pada fungsi ini, terdapat dua fungsi baru yaitu LOGIN dan LOGOUT fungsinya akan menerima masukan sesi tiap *user* dan mengembalikannya kembali ke penyimpanan awalnya. Hanya 10 *user* yang dapat didaftarkan pada program ini.

2.3 Radio (*Custom Feature*)

Sebagai user setia WayangWave pasti akan mudah bosan bila lagu yang sama diulang-ulang terus menerus. Untuk itu, Spotify memberikan fitur Radio yang akan memutar lagu yang mirip sedemikian rupa dengan lagu favorit dari user tersebut. Namun, implementasi *machine learning* pada tugas besar ini masih cukup rumit, alternatif solusi kami adalah membuat algoritma berdasarkan panjang lagu tersebut dan dimudahkan dengan implementasi ADT *undirected graph* yang mengindikasikan bahwa semua lagu dalam radio tersebut terkoneksi pada lagu utama. User akan diperlihatkan seluruh lagu dalam sesi dan akan diberikan pilihan. Setelah diperlihatkan radio dari lagu yang dipilih, user akan diberikan opsi untuk memutar radio lagu tersebut.

```
>> RADIO;
DAFTAR SELURUH LAGU DALAM SESI INI :
BORN PINK :
1. Pink Venom
2. Shut Down
3. Typa Girl
4. Ready For Love
THE ALBUM :
5. How You Like That
6. Ice Cream (with Selena Gomez)
7. Bet You Wanna (Feat. Cardi B)
Favourite Worst Nightmare :
8. Brianstorm
9. Teddy Picker
10. Fluorescent Adolescent
11. Old Yellow Bricks
12. 505
Humbus :
13. My Propeller
14. Crying Lightning
15. Secret Door
16. Pretty Visitors
Silahkan masukan ID lagu yang ingin ditampilkan sebagai radio : █
```

Gambar 2.3.1 : Tampilan utama RADIO

3 Struktur Data (ADT)

Pada program yang kami buat, kami menggunakan beberapa struktur data (ADT) untuk menyelesaikan permasalahan yang ada dan membuat program WayangWave menjadi program yang dapat berjalan dengan baik dan tepat. Beberapa struktur data (ADT) yang kami gunakan adalah ADT Linked List, ADT List, ADT Map, ADT Mesin Karakter, ADT Mesin Kata, ADT Queue, ADT Set, ADT Stack, ADT Graph, ADT NowPlaying, dan ADT UserSession.

3.1 Struktur Data Linked List

```
typedef struct tElmtlist *addressnode;
typedef struct tElmtlist
{
    int idpenyanyi;
    int idalbum;
    int idlagu;
    addressnode next;
} Elmtlist;
```

```
typedef struct
{
    addressnode First;
    Word namaplaylist;
    int idplaylist;
}playlist;
```

Dalam pengembangan program WayangWave, kami memiliki sebuah *linked-list* tunggal. Setiap elemen dalam *linked-list* tersebut direpresentasikan sebagai *node* dengan tiga atribut, yaitu idpenyanyi, idalbum, dan idlagu. Selain itu, terdapat *pointer next* yang menunjukkan alamat dari *node* selanjutnya dalam *linked-list*. Struktur data ini diimplementasikan sebagai ADT dalam bentuk sebuah *file header* linked_list.h yang berisi definisi tipe data dan fungsi yang dapat digunakan oleh program utama. ADT *linked-list* memberikan operasi dasar seperti alokasi, dealokasi, penambahan elemen, penghapusan elemen, dan pencarian elemen.

3.2 Struktur Data List

```
typedef struct
{
    Word namapenyanyi;
    ElType jumlahalbum;
    ElType IdAlbumPertama;
} Penyanyi;

typedef struct
{
    Penyanyi penyanyi[IdxMax - IdxMin + 1];
    int Neff;
} TabInt;

typedef struct
{
    playlist *A;
    int Capacity;
    int Neff;
} ArrayDin;
```

Dalam pengembangan program WayangWave, kami juga menggunakan struktur data *list* statis untuk menyimpan informasi tentang penyanyi, album, dan lagu. *List* statis ini diimplementasikan sebagai array, di mana setiap elemen menyimpan data lengkap mengenai seorang penyanyi, termasuk nama penyanyi (namapenyanyi), jumlah album (jumlahalbum), dan ID album pertama (IdAlbumPertama). Selain menggunakan struktur data *array* statis, kami juga memanfaatkan struktur data *list* dinamis untuk menyimpan *playlist*. Setiap elemen dari *list* dinamis ini menyimpan informasi tentang *playlist*, seperti nama *playlist* (namaplaylist), ID *playlist* (idplaylist), dan lagu-lagu yang termasuk dalam *playlist* tersebut.

3.3 Struktur Data Map

```
typedef struct
{
    keytype Key;
    valuetype Value;           ...
    Word AlbumName;
} Album;

typedef struct
{
    Album Elements[MaxEl];
    address Count;
} MapAlbum;
```

Sketsa Struktur data Map merupakan kumpulan pasangan *Key* yang bertipe *keytype* dengan *Value* yang bertipe *valuetype* Map. *Key* dari Map memiliki nilai yang unik. Struktur data Map memiliki tipe data Map yang terdiri dari Elements[MaxEl] sebagai memori tempat penyimpanan elemen dan sebuah *Count* bertipe integer untuk menyatakan jumlah elemen yang ada pada Map. Persoalan yang diselesaikan dengan menggunakan ADT ini adalah persoalan mengenai menyimpan dan menampilkan judul album untuk setiap penyanyi dan juga judul lagu untuk setiap album. Alasan kami memilih ADT ini adalah karena mengikuti spesifikasi tugas besar yang diberikan serta untuk memberi kemudahan dalam memetakan hubungan antara judul lagu dengan judul albumnya dan juga judul album dengan nama penyanyinya. Diimplementasikan sebagai ADT Map dengan nama *file header* “map.h”.

3.4 Struktur Data Mesin Karakter

Sketsa Struktur data Mesin Karakter memiliki sebuah *variable extern* dengan nama *currentChar* dan *boolean* dengan nama *EOP*. Adapun yang ada dalam ADT ini yaitu *START()* dan *ADV()*. Persoalan yang diselesaikan dengan menggunakan ADT ini adalah persoalan mengenai melakukan *parsing command* program dan pembacaan *file* konfigurasi ke dalam aplikasi. Alasan kami memilih ADT ini adalah karena ADT Mesin Kata membutuhkan ADT Mesin Karakter dalam fungsi *prototype*. Diimplementasikan sebagai ADT mesin karakter dengan nama *file header* “mesinkarakter.h”.

3.5 Struktur Data Mesin Kata

```
typedef struct
{
    char TabWord[NMax];
    int Length;
} Word;
```

Sebagai pelengkap atau penyempurna dari struktur data Mesin Karakter, kelompok kami juga mengaplikasikan struktur data Mesin Kata. Struktur ini berperan dalam menyempurnakan peran Mesin Karakter dalam pembacaan sebuah masukan baik dari pengguna secara langsung, maupun dari *file txt*.

Pada proses pembacaan *file* atau dari *input* pengguna, kumpulan *character* yang dibacakan secara bergantian menduduki variabel *currentWord* yang memiliki tipe *Word*. Melalui variabel tersebut, kata dapat diproses dengan lebih lanjut. Struktur data Mesin Kata ini tersimpan pada *file* `source/ADT/Mesinkata/mesinkata.c` dan pada `source/ADT/Mesinkata/mesinkata.h` untuk *header* program. Penerapan struktur data Mesin Kata ini sama dengan penerapan struktur data Mesin Karakter karena keduanya saling melengkapi dan berbagi peran.

3.6 Struktur Data Queue

```
typedef struct
{
    IdxType penyanyi;
    KEY album;
    KEY lagu;
} isiqueue;

typedef struct
{
    isiqueue Tab[IDX_MAX + 1];
    IdxType idxHead;
    IdxType idxTail;
} Queue;
```

Program *WayangWave* yang kami buat juga memanfaatkan ADT *queue* khususnya dengan tipe data *word*. Pada struktur data ini, definisi ADT *queue* direpresentasikan dengan array secara eksplisit dan alokasi dinamik. Hal ini berarti ada proses alokasi dan dealokasi yang digunakan. Tipe data *Queue* akan menyimpan empat jenis data yaitu tabel penyimpan elemen, alamat penghapusan, alamat penambahan, dan elemen maksimum *queue*. Tipe data dari tempat penyimpanan elemen adalah sebuah tipe *word* sedangkan tipe data dari alamat penghapusan dan penambahan adalah *address*. Tipe data *address* ini adalah tipe data *integer* yang menyimpan *indeks* tabel. Struktur data *queue* ini tersimpan pada *file* `source/ADT/queue/circular_queue.c` dan `source/ADT/queue/circular_queue.h` untuk *header*.

3.7 Struktur Data Set

```
typedef int address;

typedef struct
{
    Word JudulLagu;
    int IdAlbum;
} Lagu;

typedef struct
{
    Lagu AlbumLagu[MaxEl];
    address Count;
```

```

        int idSet;
    } Set;

```

Dalam pengembangan program WayangWave, kami menggunakan struktur data Set untuk mengelola kumpulan lagu yang dimainkan dalam aplikasi. Struktur data Set diimplementasikan dengan representasi *array* eksplisit-statik, dan tipe data ini disimpan dalam *file* source/ADT/Set/set.c dengan *header* tersimpan pada source/ADT/Set/set.h. Setiap elemen dalam *set* direpresentasikan sebagai suatu lagu yang memiliki judul dan identifikasi album. Selain itu, terdapat atribut *Count* untuk menyimpan jumlah elemen dalam *set* dan idSet untuk identifikasi *set* tersebut.

3.8 Struktur Data Stack

```

typedef int idx;

typedef struct
{
    idx penyanyi;
    idx album;
    idx lagu;
} isistack;

typedef struct
{
    isistack T[MaxEl];
    idx TOP;
} Stack;

```

Pada pengerjaan program WayangWave ini, kami menggunakan struktur data *Stack*. Struktur data yang kami gunakan merupakan struktur data *stack* dengan representasi struktur berkait. Struktur data ini juga direpresentasikan secara eksplisit dan alokasi dinamik. Tipe data *stack* akan menyimpan *address* TOP dan *count* dengan *type integer*. *Element* dari *stack* akan menyimpan Info dan yang disimpan merupakan sebuah *integer*. Struktur data *stack* ini tersimpan pada *file* source/ADT/Stack/stack.c dan pada source/ADT/Stack/stack.h untuk *header* ADT.

3.9 Struktur Data Graph

```

typedef int ElType;

typedef struct BMatrix
{
    boolean Mem[barismax][kolommax];
    int NBrseff;
    int NKoleff;
} BMatrix;

typedef struct Graph

```

```

{
    int numEdges;
    int numVertices;
    BMatrix adjacencyMatrix;
} Graph;

```

Pada program WayangWave ini kami mengimplementasikan *undirected graph* dengan struktur data yang memuat integer numEdges, integer numVertices serta adjacencyMatrix yang berupa boolean matrix. Boolean Matrix didefinisikan sebagai BMatrix yang membuat sebuah matrix berisi boolean dan integer NBarisEff serta integer NKolomEff. Konstruktor yang digunakan pada kedua ADT ini adalah MakeMatrix dan CreateGraph dan diisi dengan fungsi untuk memenuhi kebutuhan Radio. ADT ini diperlukan untuk membuat bonus Radio untuk membuat klasifikasi lagu. Fungsi - fungsi ini diimplementasikan pada Radio (*Radio* (src/COMMANDS/radio/))

3.10 Struktur Data NowPlaying

```

typedef int currentid;

typedef struct
{
    IdxType penyanyi;
    IdxType album;
    IdxType lagu;
} NowPlaying;

typedef struct
{
    NowPlaying current;
    Queue antrian;
    Stack riwayat;
    ArrayDin playlists;
    Word namauser;
} UserSession;

```

NowPlaying merupakan struktur data yang berisi idLagu, idAlbum, dan idPenyanyi. Konstruktor yang digunakan pada struktur data ini adalah void NotPlaying() dan fungsi untuk mengecek ada tidaknya lagu yang dimainkan adalah boolean isNotPlaying(). Struktur data ini dibuat untuk memudahkan pengerjaan beberapa fungsi yang membutuhkan data lagu yang sedang dimainkan, dan fungsi-fungsi yang mengimplementasikan struktur data ini di antaranya *Play* (src/COMMANDS/play/) , *Song* (src/COMMANDS/song/) , *status* (src/COMMANDS/status/), dan *Multiuser* (src/COMMANDS/startload/)

3.11 Struktur Data UserSession

```

typedef struct
{
    IdxType penyanyi;
    IdxType album;
    IdxType lagu;
} NowPlaying;

typedef struct
{
    NowPlaying current;
    Queue antrian;
    Stack riwayat;
    ArrayDin playlists;
    Word namauser;
} UserSession;

```

UserSession merupakan struktur data yang berisi lagu yang sedang dimainkan, antrian, riwayat, playlists, dan nama user. Beberapa primitif dari struktur data ini adalah void loginFunction() dan void login Function() untuk mengubah sesi sesuai user serta int SearchUser(UserSession u[], Word Nama) untuk mencari ID dari salah satu user. ADT ini diperlukan karena untuk fungsi bonus multi user, diperlukan penyimpanan sesi dari tiap user. Implementasi struktur data ini digunakan pada fungsi multi user (src/COMMANDS/startload/)

4 Program Utama

Pada program buatan kami, alur program dimulai dari main.c. kemudian kami membuat program START, LOAD, dan HELP. Apabila menggunakan input START maka akan menjalankan program dengan menginput perintah yang diinginkan. Apabila melakukan input LOAD maka harus menginput file yang akan dijalankan. Hal ini akan melanjutkan permainan pada *file* yang sudah tersimpan terlebih dahulu. Apabila melakukan *input* HELP maka secara otomatis akan menampilkan bantuan yang terdiri dari *command* yang akan digunakan dalam program. Pada saat menginput START maka akan menampilkan *command* apa saja yang bisa di input oleh pengguna. *Command* yang dapat diinput antara lain :

- LIST : Menampilkan daftar lagu, *playlist*
- PLAY : Memulai suatu lagu
- QUEUE : Memanipulasi *queue* lagu
- SONG : Menavigasi lagu yang ada pada *queue* lagu saat ini
- PLAYLIST : Membuat *playlist* baru dan ditambahkan pada daftar *playlist* pengguna
- STATUS : Menampilkan lagu yang sedang dimainkan beserta *queue song* yang ada dari playlist mana lagu itu diputar
- SAVE : Menyimpan *state* aplikasi terbaru ke dalam suatu *file*
- QUIT : Keluar dari sesi

Pada saat melakukan QUEUE SONG maka akan menginput lagu yang ingin dimainkan maka pada saat PLAY SONG akan secara otomatis memainkan lagu pada urutan pertama. Lagu akan dimainkan sesuai dengan urutan pada QUEUE SONG yang di input oleh pengguna. *Playlist* juga dapat dibuat sendiri oleh pengguna dan dapat dihapus juga oleh pengguna. Setelah pengguna selesai menggunakan program tentu pengguna akan melakukan *command* QUIT untuk keluar dari program. Pada saat menginput *command* QUIT, maka akan ada dua pilihan yaitu untuk menyimpan *file* yang sudah diproses saat menggunakan program atau tidak menyimpan *file*.

5 Algoritma-Algoritma Menarik

Pada tugas besar ini, kami menemukan algoritma yang menarik untuk digunakan serta algoritma yang kami anggap efisien sehingga dapat mengurangi efisiensi program.

5.1 Tausworthe Random Number Generator

Tausworthe Number Generator adalah sebuah algoritma *random number generator* yang dapat digunakan untuk menghasilkan deret bilangan yang acak. Algoritma ini memuat inisialisasi *seed* (pada tugas besar ini menggunakan waktu *microsecond realtime*) dan dilakukan iterasi *shifting bit operator* serta XOR juga. Algoritma ini kami definisikan menarik karena menggunakan *logical bit operators* untuk mendapatkan angkanya. Algoritma ini digunakan pada bonus ENHANCE sebagai *random number generator*.

```
int tauswortheNumberGenerator(int min, int max){
    struct timeval currentTime;
    gettimeofday(&currentTime, NULL);
    unsigned int s1 = 13, s2 = 19, s3 = 21;
    unsigned int mask = 0xffffffffUL;

    int output, seed = currentTime.tv_usec;
    seed = (seed & mask) << s1 ^ ((seed << s2) & mask);
    seed = (seed & mask) ^ ((seed >> s3) & mask);
    output = (seed % (max-min)) + min;

    return output;
}
```

Gambar 5.1.1 Implementasi Tausworthe Number Generator

5.2 Radio Song Classification

Dalam melakukan fungsi radio, Spotify melakukan inisiatif untuk membuat algoritma yang unik agar radio setiap lagu terdistribusi dengan baik. Dalam melakukan klasifikasi kami menggunakan panjang dari lagu tersebut (length of word) kemudian dijadikan sebuah vertices pada ADT Graph. Panjang tersebut melakukan determinasi seberapa banyak lagu yang akan dimasukan. Algoritma ini digunakan pada fungsi RADIO menggunakan ADT Graph.

```

while(a < lengthlagu+1 && a < MAX){
    if ((lengthlagu + a) % MAX == rsong.El[0])
    {
        a++;
    }
    else
    {
        rsong.El[a] = (lengthlagu + a) % MAX;
        rsong.RNeff++;
        a++;
    }
}

CreateGraph(&g);
MakeMatrix(rsong.RNeff, rsong.RNeff, &g.adjacencyMatrix);
printf("\nRadio dari lagu \"%s\" :\n", temp[rsong.El[0]].TabWord);
for(int i = 0; i < ADJMaxtrix(g).NBrsEff; i++){
    addVerticesToAll(&g, i);
    if (isDirectlyConnected(g, 0, i))
    {
        printf("%d. %s\n", i, temp[rsong.El[i]].TabWord);
    }
}

```

Gambar 6.2.1 : Implementasi pemilihan lagu pada RADIO

6 Data Test

Program WayangWave ini terdiri dari beberapa fitur atau *command* yang dapat diberikan oleh pengguna. Oleh karena itu, diperlukan *testing* pada beberapa kondisi untuk memastikan apakah tiap fitur tersebut dapat berjalan dengan tepat dan benar. Berikut adalah *testing* yang kami lakukan terhadap *command* yang terdapat pada program WayangWave ini, beserta dengan penjelasan mengenai hasil atau *output* yang akan diberikan oleh program pada tiap kondisinya.

6.1 Data Test START

Tes ini dilakukan untuk mengecek keberjalanan program utama dan kondisi awal program

```

>> START;

File konfigurasi aplikasi berhasil dibaca. WayangWave berhasil dijalankan.

```

Gambar 6.1.1 Tampilan ketika berhasil memulai start

```

>> STATUS;

Now Playing:
Belum pernah ada lagu yang dimainkan. Gunakan fitur PLAY dan SONG untuk memainkan lagu.

Queue:
Queue kosong.

```

Gambar 6.1.2 Kondisi awal program

6.2 Data Test LOAD

Tes ini dilakukan untuk mengecek keberjalanan program utama dan kondisi awal program

```
>> LOAD new.txt;

new berhasil dibaca. WayangWave berhasil dijalankan.
```

Gambar 6.2.1 Tampilan ketika berhasil load file yang telah tersimpan

```
>> STATUS;

Now Playing:
Kodaline - Head Held High - Politics of Living.

Queue:
1. Kodaline - Head Held High - Politics of Living
2. Taylor Swift - Willow - Evermore
```

Gambar 6.2.2 Kondisi setelah melakukan load sesi yang sudah ada

6.3 Data Test LIST DEFAULT

Tes ini dilakukan untuk mengecek daftar lagu di suatu album oleh salah satu artis

```
>> LIST DEFAULT;

Daftar Penyanyi :
1. BLACKPINK
2. Arctic Monkeys

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

Gagal. Masukan bukan Y/N.
>> 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 Tampilan ketika list default berhasil dijalankan dan ingin melihat album

```
>> LIST DEFAULT;

Daftar Penyanyi :
1. BLACKPINK
2. Arctic Monkeys

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

Gambar 6.3.2 Tampilan berhasil dijalankan tetapi tidak ingin melihat album

6.4 Data Test LIST PLAYLIST

Tes ini dilakukan untuk mengecek daftar *playlist* yang ada

```
>> LIST PLAYLIST;

Daftar playlist yang kamu miliki :
1. BLACKPINK My Top Three
2. Arctic Monkeys My Top 5
3. Mixed Playlist
```

Gambar 6.4.1 Tampilan ketika list playlist berhasil dijalankan dan memiliki playlist

```
>> LIST PLAYLIST;

Daftar playlist yang kamu miliki :

Kamu tidak memiliki playlist.
```

Gambar 6.4.2 Tampilan ketika tidak ada playlist yang tersimpan

6.5 Data Test PLAY SONG

Tes ini dilakukan untuk mengecek keberjalanan program untuk memutar salah satu lagu

```
>> PLAY SONG;

Daftar Penyanyi :
1. Taylor Swift
2. Arctic Monkeys
3. Kodakline

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 di AM :
1. Do I Wanna Know?
2. R U Mine?
3. Arabella

Masukkan ID Lagu yang dipilih : 1;
Memutar lagu "Do I Wanna Know?" oleh "Arctic Monkeys".
```

Gambar 6.5.1 Tampilan ketika play song dijalankan

```
>> STATUS;

Now Playing:
Arctic Monkeys - Do I Wanna Know? - AM.

Queue:
Queue kosong.
```

Gambar 6.5.2 Status program ketika play song dijalankan

6.6 Data Test PLAY PLAYLIST

Tes ini dilakukan untuk mengecek keberjalanan program untuk memainkan salah satu *playlist* yang tersedia

```
>> LIST PLAYLIST;

Daftar playlist yang kamu miliki :
1. Playlist Favorit

>> PLAY PLAYLIST;

Masukkan ID Playlist: 1;

Memutar playlist "Playlist Favorit".
```

Gambar 6.6.1 Tampilan ketika play playlist berhasil dijalankan

```
>> STATUS;
Current Playlist: Playlist Favorit.

Now Playing:
Kodaline - Head Held High - Politics of Living.

Queue:
1. Arctic Monkeys - R U Mine? - AM
2. Arctic Monkeys - My Propeller - Humbug
3. Arctic Monkeys - Crying Lightning - Humbug
4. Arctic Monkeys - Secret Door - Humbug
5. Arctic Monkeys - Pretty Visitors - Humbug
```

Gambar 6.6.2 Status program setelah dilakukan play playlist

```
Daftar playlist yang kamu miliki :
1. BLACKPINK My Top Three
2. Arctic Monkeys My Top 5
3. Mixed Playlist

>> PLAY PLAYLIST;
Masukkan ID Playlist: 4;
ID Playlist tidak ditemukan!
>> █
```

Gambar 6.6.3 Tampilan ketika id playlist invalid

6.7 Data Test QUEUE SONG

Tes ini dilakukan untuk mengecek keberjalanan fungsi *Queue Song* dan kondisi program sebelum dan setelah *queue*

```
>> STATUS;  
  
Now Playing:  
Taylor Swift - ME! - Lover.  
  
Queue:  
Queue kosong.
```

Gambar 6.7.1 Kondisi awal program

```
>> QUEUE SONG;  
  
Daftar Penyanyi :  
1. Taylor Swift  
2. Arctic Monkeys  
3. Kodaline  
  
Masukkan Nama Penyanyi : Kodaline;  
  
Daftar Album oleh Kodaline :  
1. In a Perfect World  
2. Politics of Living  
  
Masukkan Nama Album yang dipilih : Politics of Living;  
Daftar Lagu di Politics of Living :  
1. Head Held High  
2. Born Again  
  
Masukkan ID Lagu yang dipilih : 2;
```

Gambar 6.7.2 Tampilan ketika dilakukan queue song

```
>> STATUS;  
  
Now Playing:  
Taylor Swift - ME! - Lover.  
  
Queue:  
1. Kodaline - Born Again - Politics of Living
```

Gambar 6.7.3 Kondisi program setelah dilakukan queue song

6.8 Data Test QUEUE PLAYLIST

Tes ini dilakukan untuk mengecek keberjalanan fungsi *Queue Playlist*

```
>> QUEUE PLAYLIST;  
Masukan ID Playlist: 2;  
Berhasil menambahkan playlist Mixed Playlist ke queue.
```

Gambar 6.8.1 Tampilan ketika queue playlist berhasil dijalankan

6.9 Data Test QUEUE SWAP

Tes ini dilakukan untuk mengecek keberjalanan fungsi *Queue Swap*

```
>> QUEUE SWAP 1 2;  
Lagu Typa Girl berhasil ditukar dengan Pink Venom
```

Gambar 6.9.1 Tampilan ketika queue swap berhasil dijalankan

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

Gambar 6.9.2 Tampilan ketika queue swap lagu tidak sesuai

6.10 Data Test QUEUE REMOVE

Tes ini dilakukan untuk mengecek keberjalanan fungsi *Queue Remove*

```
>> QUEUE REMOVE 2;  
Lagu Do I Wanna Know? oleh Arctic Monkeys telah dihapus dari queue!
```

Gambar 6.10.1 Tampilan ketika queue remove berhasil dijalankan

```
>> QUEUE REMOVE 40;  
Lagu dengan urutan ke 40 tidak ada.
```

Gambar 6.10.2 Tampilan ketika queue remove memiliki lagu yang tidak sesuai

6.11 Data Test QUEUE CLEAR

Tes ini dilakukan untuk mengecek keberjalanan fungsi *Queue Clear*

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

Gambar 6.11.1 Tampilan ketika queue clear berhasil dijalankan

6.12 Data Test SONG NEXT

Tes ini dilakukan untuk mengecek keberjalanan fungsi *Song Next*

```
>> SONG NEXT;  
Memutar lagu selanjutnya  
"Pink Venom" oleh "BLACKPINK"
```

Gambar 6.12.1 Tampilan ketika song next berhasil dijalankan

```
>> SONG NEXT;  
Queue kosong, memutar kembali lagu  
"R U Mine?" oleh "Arctic Monkeys"
```

Gambar 6.12.1 Tampilan song next ketika queue kosong

6.13 Data Test SONG PREVIOUS

Tes ini dilakukan untuk mengecek keberjalanan fungsi *Song Previous*

```
>> SONG PREVIOUS;  
Riwayat lagu kosong, memutar kembali lagu  
"Pink Venom" oleh "BLACKPINK"
```

Gambar 6.13.1 Tampilan ketika song previous berhasil dijalankan

```
>> SONG PREVIOUS;  
Memutar lagu sebelumnya  
"505" oleh "Arctic Monkeys"
```

Gambar 6.13.2 Tampilan song previous ketika riwayat lagu kosong

6.14 Data Test PLAYLIST CREATE

Tes ini dilakukan untuk mengecek keberjalanan program dalam membuat playlist baru menggunakan fungsi *Playlist Create*

```
>> 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 6.14.1 Tampilan ketika playlist create berhasil dijalankan

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

Gambar 6.14.2 Tampilan ketika penamaan tidak sesuai

6.15 Data Test PLAYLIST ADD

Tes ini dilakukan untuk mengecek keberjalanan fungsi *Playlist Add* untuk menambahkan suatu lagu/album ke *playlist*

```
>> PLAYLIST ADD 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 Judul Album yang dipilih : BORN PINK;  
  
Daftar Lagu di BORN PINK :  
1. Pink Venom  
2. Shut Down  
3. Typa Girl  
4. Ready For Love  
  
Masukkan ID Lagu yang dipilih : 1;  
1. BLACKPINK My Top Three  
2. Arctic Monkeys My Top 5  
3. Mixed Playlist  
4. K-pop Enjoyer  
  
Masukkan ID Playlist yang dipilih : 1;  
  
Lagu dengan judul "Pink Venom" pada album "BORN PINK" oleh penyanyi "BLACKPINK" berhasil ditambahkan ke dalam playlist "BLACKPINK My Top Three".
```

Gambar 6.15.1 Tampilan ketika playlist add song berhasil dijalankan

```
>> 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 Judul Album yang dipilih : BORN PINK;
1. BLACKPINK My Top Three
2. Arctic Monkeys My Top 5
3. Mixed Playlist
4. K-pop Enjoyer

Masukkan ID Playlist yang dipilih : 4;

Album dengan judul "BORN PINK" berhasil ditambahkan ke dalam playlist pengguna "K-pop Enjoyer".
```

Gambar 6.15.2 Tampilan ketika playlist add album berhasil dijalankan

```
>> PLAYLIST ADD SONG;

Daftar Penyanyi :
1. BLACKPINK
2. Arctic Monkeys

Masukkan Nama Penyanyi yang dipilih : BLACKKPIN;
Penyanyi BLACKKPIN tidak ada dalam daftar. Silakan coba lagi.
```

Gambar 6.15.3 Tampilan ketika playlist add song salah input penyanyi

```
>> 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 Judul Album yang dipilih : BORN WHIT;
Album BORN WHIT tidak ada dalam daftar. Silakan coba lagi.
```

Gambar 6.15.4 Tampilan ketika playlist add album salah input album

6.16 Data Test PLAYLIST SWAP

Tes ini dilakukan untuk mengecek keberjalanan fungsi *Playlist Swap*

```
>> PLAYLIST SWAP 1 1 2;

Berhasil menukar lagu dengan nama "How You Like That" dengan "Typa Girl" di playlist "BLACKPINK My Top Three".
```

Gambar 6.16.1 Tampilan ketika playlist swap berhasil dijalankan

```
>> PLAYLIST SWAP 500 1 2;

Tidak ada playlist dengan playlist ID 500
```

Gambar 6.16.2 Tampilan ketika playlist swap tidak memiliki playlist ID

```
>> PLAYLIST SWAP 1 900 2;  
Tidak ada lagu dengan urutan 900 di playlist "BLACKPINK My Top Three"!
```

Gambar 6.16.3 Tampilan ketika playlist swap tidak memiliki lagu pada urutan

6.17 Data Test PLAYLIST REMOVE

Tes ini dilakukan untuk mengecek keberjalanan fungsi *Playlist Remove* untuk menghapus lagu dalam playlist

```
>> PLAYLIST REMOVE 2 4;  
Lagu "505" oleh "Arctic Monkeys" telah dihapus dari playlist "Arctic Monkeys My Top 5"!
```

Gambar 6.17.1 Tampilan ketika playlist remove berhasil dijalankan

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

Gambar 6.17.2 Tampilan ketika playlist remove tidak memiliki playlist ID

```
>> PLAYLIST REMOVE 1 100;  
Tidak ada lagu dengan urutan 100 di playlist "BLACKPINK My Top Three"!
```

Gambar 6.17.3 Tampilan ketika playlist remove tidak memiliki lagu pada urutan

6.18 Data Test PLAYLIST DELETE

Tes ini dilakukan untuk mengecek keberjalanan fungsi *Playlist Delete*

```
>> PLAYLIST DELETE;  
  
Daftar Playlist Pengguna :  
1. BLACKPINK My Top Three  
2. Arctic Monkeys My Top 5  
3. Mixed Playlist  
4. K-pop Enjoyer  
  
Masukkan ID Playlist yang dipilih : 2;  
  
Playlist ID 2 dengan judul "Arctic Monkeys My Top 5" berhasil dihapus.
```

Gambar 6.18.1 Tampilan ketika playlist delete berhasil dihapus

```
>> PLAYLIST DELETE;  
  
Daftar Playlist Pengguna :  
1. BLACKPINK My Top Three  
2. Arctic Monkeys My Top 5  
3. Mixed Playlist  
  
Masukkan ID Playlist yang dipilih : 10;  
  
Tidak ada playlist dengan playlist ID 10
```

Gambar 6.18.2 Tampilan ketika playlist remove memiliki ID yang tidak valid

6.19 Data Test STATUS

Tes ini dilakukan untuk mengecek keberjalanan fungsi *Status*

```
>> STATUS;
Current Playlist: Mixed Playlist.

Now Playing:
BLACKPINK – THE ALBUM – How You Like That.

Queue:
Your queue is empty.
```

Gambar 6.19.1 Tampilan ketika tidak ada queue dalam sesi

```
>> STATUS;

Now Playing:
Arctic Monkeys – AM – R U Mine?.

Queue:
1. BLACKPINK – Typa Girl – BORN PINK
2. BLACKPINK – How You Like That – THE ALBUM
3. Arctic Monkeys – Do I Wanna Know? – AM
4. BLACKPINK – Shut Down – BORN PINK
5. Arctic Monkeys – Arabella – AM
```

Gambar 6.19.2 Tampilan ketika terdapat queue ; tidak memainkan playlist

6.20 Data Test SAVE

Tes ini dilakukan untuk mengecek keberjalanan fungsi *Save*

```
>> SAVE new.txt;

new berhasil disimpan.
```

Gambar 6.20.1 Tampilan ketika SAVE berhasil dijalankan

6.21 Data Test QUIT

Tes ini dilakukan untuk mengecek keberjalanan fungsi *quit* dan *save* sebelum *quit*

```
>> QUIT;
Apakah kamu ingin menyimpan data sesi sekarang?
Y;
Masukkan nama file!
* Perhatikan kesalahan penulisan dapat mengakibatkan file tidak tersimpan, program akan otomatis tidak menyimpan file apabila salah menulis nama file!
** Contoh penulisan yang benar : "test.txt;"

Proses save gagal. Masukkan format file yang benar! (nama file harus diakhir .txt)
```

Gambar 6.21.1 Tampilan ketika QUIT gagal karena format salah

```
>> QUIT;
Apakah kamu ingin menyimpan data sesi sekarang?
N;
Kamu keluar dari WayangWave.
Dadah ^_^/
```

Gambar 6.21.2 Tampilan ketika QUIT berhasil dijalankan dan tidak disimpan

```
>> QUIT;
Apakah kamu ingin menyimpan data sesi sekarang?
Y;
Masukkan nama file!
* Perhatikan kesalahan penulisan dapat mengakibatkan file tidak tersimpan, program akan otomatis tidak menyimpan file apabila salah menulis nama file!
** Contoh penulisan yang benar : "test.txt;"
new.txt;
new berhasil disimpan.
```

Gambar 6.21.3 Tampilan ketika QUIT berhasil dijalankan dan disimpan

6.22 Data Test HELP

Tes ini dilakukan untuk mengecek fungsi *Help* sebelum dan setelah *start* atau *load*

```
>> HELP;
===== [ Menu Help WayangWave ] =====
1. START -> Untuk masuk sesi baru
2. LOAD -> Untuk memulai sesi berdasarkan file konfigurasi
```

Gambar 6.22.1 Tampilan ketika HELP berhasil dijalankan sebelum aplikasi dijalankan

```
>> HELP;
===== [ Menu Help WayangWave ] =====
1. LIST -> Untuk menampilkan daftar lagu, playlist
2. PLAY -> Untuk memulai suatu lagu
3. QUEUE -> Untuk memanipulasi queue lagu
4. SONG -> Untuk menavigasi lagu yang ada pada queue lagu saat ini
5. PLAYLIST -> Untuk membuat playlist baru dan ditambahkan pada daftar playlist pengguna
6. STATUS -> Untuk menampilkan lagu yang sedang dimainkan beserta Queue song yang ada dan dari playlist mana lagu itu diputar
7. SAVE -> Untuk menyimpan state aplikasi terbaru ke dalam suatu file
8. QUIT -> Untuk keluar dari sesi
```

Gambar 6.22.2 Tampilan ketika HELP berhasil dijalankan saat aplikasi dijalankan

6.23 Data Test INVALID COMMAND

Tes ini dilakukan untuk mengecek program ketika input tidak valid atau kondisi tidak terpenuhi

```
>> LIST DEFAULT;
Command tidak bisa dieksekusi!
```

Gambar 6.23.1 Tampilan ketika LIST tidak valid dan memberikan INVALID COMMAND

```
>> TUBES ALSTRUKDAT START;
Command tidak diketahui!
```

Gambar 6.23.2 Tampilan ketika START tidak valid dan memberikan INVALID COMMAND

```
>> QUIT;
Apakah kamu ingin menyimpan data sesi sekarang?
n;
Command tidak diketahui!
```

Gambar 6.23.3 Tampilan ketika QUIT tidak valid dan memberikan INVALID COMMAND

6.24 Data Test ENHANCE

Tes ini dilakukan untuk mengecek fungsi *ENHANCE* menggunakan *LOAD* dengan konfigurasi


```

>> ENHANCE;
ID | Nama Playlist
0 | Playlist Favorit
Masukan ID playlist yang ingin di-enhance : 3
| Playlist Favorit
PENYANYI | ALBUM | LAGU
Kodaline | Politics of Living | Head Held High
Rekomendasi Lagu yang dapat ditambahkan:
1. Taylor Swift - Lover - I Forgot That You Existed
2. Arctic Monkeys - Humbug - Pretty Visitors
3. Arctic Monkeys - AM - Arabella
4. Taylor Swift - Lover - Daylight
Masukkan ID rekomendasi yang diinginkan : 2;
Lagu berhasil ditambahkan!
| Playlist Favorit
PENYANYI | ALBUM | LAGU
Kodaline | Politics of Living | Head Held High
Arctic Monkeys | Humbug | Pretty Visitors

```

Gambar 6.24.1 Tampilan ketika ENHANCE berhasil dijalankan dengan menggunakan LOAD konfigurasi lalu LOGIN

6.25 Data Test MULTIUSER

Test ini dilakukan untuk mengecek fungsi MULTIUSER

```

>> LOGIN;
Masukkan username user WayangWave : daviddewanto;
Berhasil Login dengan user daviddewanto!

```

Gambar 6.25.1 Kasus 1: Tampilan LOGIN

```

>> LIST DEFAULT;
Silakan login terlebih dahulu.

```

Gambar 6.25.2 Kasus 2: Tampilan apabila belum LOGIN namun ingin mencoba LIST DEFAULT

```

>> SAVE;
Command tidak diketahui!

```

Gambar 6.25.3 Kasus 3: Tampilan apabila sudah LOGIN tetapi tidak bisa save (harus logout terlebih dahulu)

```

>> QUIT;
Silahkan logout terlebih dahulu!

```

Gambar 6.25.4 Kasus 4: Tampilan apabila sudah LOGIN tetapi ingin QUIT (harus LOGOUT dahulu)

6.26 Data Test RADIO

6.27 Test ini dilakukan untuk mengecek fungsi RADIO

```

>> RADIO;
DAFTAR SELURUH LAGU DALAM SESI INI :
Lover :
  1. ME!
  2. I Forgot That You Existed
  3. The Man
  4. Daylight
Red :
  5. The Last Time
  6. Starlight
Evermore :
  7. Evermore
  8. Coney Island
  9. Willow
Favourite Worst Nightmare :
 10. Brianstorm
 11. Teddy Picker
 12. Fluorescent Adolescent
 13. Old Yellow Bricks
 14. 505
Humbug :
 15. My Propeller
 16. Crying Lightning
 17. Secret Door
 18. Pretty Visitors
AM :
 19. Do I Wanna Know?
 20. R U Mine?
 21. Arabella
In a Perfect World :
 22. Big Bad World
 23. Talk
 24. Way Back When
Silahkan masukan ID lagu yang ingin ditampilkan sebagai radio :1;

Radio dari lagu "ME!" :
1. The Last Time
2. Starlight
3. Evermore

Apakah anda ingin play radio ini? (Y/N) Y;
Berhasil memutar radio!

```

Gambar 6.26.1 Tampilan apabila fungsi RADIO berhasil dijalankan

7 Test Script

No	Fitur yang dites	Tujuan Testing	Langkah-langkah testing	Input Data Test	Hasil yang diharapkan	Hasil yang keluar
1	START	Memeriksa apakah <i>file</i> konfigurasi berhasil dibaca dan menampilkan tampilan ketika <i>file</i> konfigurasi berhasil dibuka	Memberikan <i>command</i> START	START;	Gambar 6.1.1 Gambar 6.1.2	Sesuai dengan hasil yang diharapkan
2	LOAD	Memeriksa apakah <i>file</i> berhasil dibaca dan menampilkan tampilan ketika <i>file</i>	Memberikan <i>command</i> LOAD diikuti dengan nama	LOAD <filename>;	Gambar 6.2.1 Gambar 6.2.2	Sesuai dengan hasil yang diharapkan

		berhasil dibuka	<i>file yang valid</i>			
3	LIST DEFAUL T	Menampilkan <i>List</i> Penyanyi, memilih Album, dan melihat lagu berdasarkan <i>list</i> yang dipilih	Memberikan <i>command</i> LIST DEFAULT diikuti dengan memilih List Penyanyi, Album, dan Lagu	LIST DEFAUL T;	Gambar 6.3.1 Gambar 6.3.2	Sesuai dengan hasil yang diharapkan
4	LIST PLAYLIS T	Menampilkan <i>Playlist</i> Penyanyi yang ada. Kemudian pengguna dapat melihat <i>Playlist</i> yang dimiliki	Memberikan <i>command</i> LIST PLAYLIST	LIST PLAYLIS T;	Gambar 6.4.1 Gambar 6.4.2	Sesuai dengan hasil yang diharapkan
5	PLAY SONG	Memutar lagu yang dipilih berdasarkan masukan Nama Penyanyi, Nama Album, dan ID Lagu. Apabila <i>command</i> ini berhasil dieksekusi, <i>queue</i> dan riwayat lagu akan menjadi kosong	Memberikan <i>command</i> PLAY SONG diikuti dengan memilih daftar Penyanyi, Album, dan Lagu pada album tersebut.	PLAY SONG;	Gambar 6.5.1 Gambar 6.5.2	Sesuai dengan hasil yang diharapkan
6	PLAY PLAYLIS T	Memutar lagu berdasarkan ID Playlist yang dipilih. Apabila <i>command</i> ini berhasil dieksekusi, <i>current song</i> akan menjadi lagu pada urutan pertama dan <i>queue</i> berisi semua lagu yang ada dalam <i>playlist</i> yang akan dimainkan dan isi riwayat lagu sama dengan <i>queue</i> .	Memberikan <i>command</i> PLAY PLAYLIST kemudian memasukkan ID Playlist	PLAY PLAYLIS T;	Gambar 6.6.1 Gambar 6.6.2 Gambar 6.6.3	Sesuai dengan hasil yang diharapkan

		Namun dengan urutan yang dibalik (<i>reverse</i>)				
7	QUEUE SONG	Menambah lagu ke dalam <i>queue</i> . QUEUE SONG menerima input lagu berdasarkan Nama Penyanyi, Nama Album, dan ID lagu yang ingin dimasukkan ke dalam <i>queue</i>	Memberikan <i>command</i> QUEUE SONG diikuti dengan Penyanyi, Album, dan Lagu yang diinginkan	QUEUE SONG;	Gambar 6.7.1 Gambar 6.7.2 Gambar 6.7.3	Sesuai dengan hasil yang diharapkan
8	QUEUE PLAYLIST	Menambah lagu yang ada dalam <i>playlist</i> ke dalam <i>queue</i> . QUEUE Playlist menerima input dari ID Playlist yang ingin dimasukkan ke dalam <i>queue</i> saat dijalankan	Memberikan <i>command</i> QUEUE PLAYLIST dan memasukkan ID Playlist	QUEUE PLAYLIST;	Gambar 6.8.1	Sesuai dengan hasil yang diharapkan
9	QUEUE SWAP	Menukar lagu (pada urutan ke-x dan juga urutan ke-y)	Memberikan <i>command</i> QUEUE SWAP dengan urutan x dan y	QUEUE SWAP <x> <y>;	Gambar 6.9.1 Gambar 6.9.2	Sesuai dengan hasil yang diharapkan
10	QUEUE REMOVE	Menghapus lagu dari QUEUE. QUEUE REMOVE menerima input berupa urutan lagu ID yang ingin dihapus dari <i>queue</i>	Memberikan <i>command</i> QUEUE REMOVE diikuti dengan input urutan lagu (id)	QUEUE REMOVE <id>;	Gambar 6.10.1 Gambar 6.10.2	Sesuai dengan hasil yang diharapkan
11	QUEUE CLEAR	Mengosongkan <i>queue</i>	Memberikan <i>command</i> QUEUE CLEAR	QUEUE CLEAR;	Gambar 6.11.1	Sesuai dengan hasil yang diharapkan
12	SONG NEXT	Memutar lagu yang berada di dalam	Memberikan <i>command</i>	SONG NEXT;	Gambar 6.12.1 Gambar 6.12.2	Sesuai dengan

		<i>queue</i> . Lagu yang sedang diputar kemudian ditambah ke dalam daftar riwayat putar lagu. Jika <i>queue</i> kosong, yang diputar adalah lagu yang sedang diputar	SONG NEXT			hasil yang diharapkan
13	SONG PREVIOUS	Memutar lagu yang terakhir kali diputar. Lagu yang sedang diputar kemudian ditambah ke dalam <i>queue</i> dengan urutan pertama. Jika daftar riwayat lagu kosong, yang diputar adalah lagu yang sedang diputar	Memberikan <i>command</i> SONG PREVIOUS	SONG PREVIOUS;	Gambar 6.13.1 Gambar 6.13.2	Sesuai dengan hasil yang diharapkan
14	PLAYLIST CREATE	Membuat <i>playlist</i> baru dan ditambahkan pada daftar <i>playlist</i> pengguna. Keadaan awal <i>playlist</i> adalah kosong. Nama <i>playlist</i> dapat sama dengan <i>playlist</i> yang sudah ada	Memberikan <i>command</i> PLAYLIST CREATE kemudian diikuti dengan memasukkan nama <i>playlist</i> yang diinginkan	PLAYLIST CREATE;	Gambar 6.14.1 Gambar 6.14.2	Sesuai dengan hasil yang diharapkan
15	PLAYLIST ADD	Menambahkan lagu pada suatu <i>playlist</i> yang telah ada sebelumnya pada daftar <i>playlist</i> pengguna. Defaultnya, <i>command</i> PLAYLIST ADD dapat menambahkan satu spesifik lagu atau semua lagu yang ada pada album	Memberikan <i>command</i> PLAYLIST ADD (apabila PLAYLIST ADD SONG diikuti dengan memasukkan Penyanyi, Album, Judul Album, ID Lagu dan ID Playlist.	PLAYLIST ADD;	Gambar 6.15.1 Gambar 6.15.2 Gambar 6.15.3 Gambar 6.15.4	Sesuai dengan hasil yang diharapkan

		<p>kepada suatu <i>existing playlist</i>. Apabila lagu atau lagu - lagu di dalam album ada yang ingin ditambahkan sudah ada di dalam suatu <i>playlist</i> pengguna maka lagu - lagu yang ditambahkan adalah yang belum ada di <i>playlist</i> pengguna. Pesan error ditampilkan apabila masukkan pengguna tidak <i>valid</i> pada setiap permintaan masukkan</p>	<p>Sedangkan PLAYLIST ADD ALBUM diikuti dengan Penyanyi, Judul Album, dan ID Playlist)</p>			
16	PLAYLIST SWAP	<p>Menukar lagu (urutan ke-x dan urutan ke-y di <i>playlist</i> dengan urutan ke ID)</p>	<p>Memberikan <i>command</i> PLAYLIST SWAP dengan urutan key x dan y serta id</p>	<p>PLAYLIST SWAP <id> <x> <y> ;</p>	<p>Gambar 6.16.1 Gambar 6.16.2 Gambar 6.16.3</p>	<p>Sesuai dengan hasil yang diharapkan</p>
17	PLAYLIST REMOVE	<p>Menghapus lagu (dengan urutan pada <i>playlist</i> dengan <i>index</i> ID)</p>	<p>Memberikan <i>command</i> PLAYLIST REMOVE diikuti dengan urutan (n) dan Index (id)</p>	<p>PLAYLIST REMOVE <id> <n> ;</p>	<p>Gambar 6.17.1 Gambar 6.17.2 Gambar 6.17.3</p>	<p>Sesuai dengan hasil yang diharapkan</p>
18	PLAYLIST DELETE	<p>Melakukan penghapusan suatu <i>existing playlist</i> dalam daftar <i>playlist</i> pengguna. Pesan error akan ditampilkan apabila masukkan pengguna tidak <i>valid</i> pada setiap permintaan masukkan</p>	<p>Memberikan <i>command</i> PLAYLIST DELETE diikuti dengan memasukkan ID Playlist yang ingin dihapus</p>	<p>PLAYLIST DELETE;</p>	<p>Gambar 6.18.1 Gambar 6.18.2</p>	<p>Sesuai dengan hasil yang diharapkan</p>

19	STATUS	Menampilkan lagu yang sedang dimainkan beserta <i>queue song</i> yang ada dan dari <i>playlist</i> mana lagu itu diputar.	Memberikan <i>command</i> STATUS	STATUS;	Gambar 6.19.1 Gambar 6.19.2	Sesuai dengan hasil yang diharapkan
20	SAVE	Menyimpan state aplikasi terbaru ke dalam suatu <i>file</i> . SAVE memiliki satu argumen yang merepresentasikan nama <i>file</i> yang akan disimpan dan disimpan pada <i>folder</i> tertentu.	Memberikan <i>command</i> SAVE diikuti dengan <i>filename</i>	SAVE <filename>;	Gambar 6.20.1	Sesuai dengan hasil yang diharapkan
21	QUIT	Keluar dari aplikasi	Memberikan <i>command</i> QUIT diikuti dengan apakah tempat <i>filename</i> jika ingin disimpan	QUIT;	Gambar 6.21.1 Gambar 6.21.2 Gambar 6.21.3	Sesuai dengan hasil yang diharapkan
22	HELP	Menampilkan daftar <i>command</i> yang mungkin untuk dieksekusi dengan deskripsinya.	Memberikan <i>command</i> HELP	HELP;	Gambar 6.22.1 Gambar 6.22.2	Sesuai dengan hasil yang diharapkan
23	INVALID COMMANDS	Apabila tidak <i>valid</i> dan hanya akan mengeluarkan teks error	Akan mengeluarkan Invalid Command apabila <i>input</i> salah	<INVALID COMMANDS>;	Gambar 6.23.1 Gambar 6.23.2 Gambar 6.23.3	Sesuai dengan hasil yang diharapkan
24	ENHANCE	Membuktikan apakah ID Playlist dapat di- <i>enhance</i> (serta lagu yang ingin ditambahkan)	LOAD lalu LOGIN kemudian diikuti <i>command</i> ENHANCE	ENHANCE;	Gambar 6.24.1	Sesuai dengan hasil yang diharapkan

25	MULTIUSER	Membuktikan apakah program bisa berjalan dengan baik	LOAD lalu bisa diikuti dengan LIST DEFAULT; atau SAVE; atau LOGOUT;	LOGIN; //sesi LOGOUT; //sesi berakhir	Gambar 6.25.1 Gambar 6.25.2 Gambar 6.25.3 Gambar 6.25.4	Sesuai dengan hasil yang diharapkan
26	RADIO;	Membuktikan fitur RADIO dapat berjalan dengan baik	Memberikan <i>command</i> RADIO dengan lagu yang ingin diputar	RADIO;	Gambar 6.26.1	Sesuai dengan hasil yang diharapkan

8 Pembagian Kerja dalam Kelompok

No	Nama Anggota - NIM	Deskripsi Kontribusi
1	Bihurin Salsabila Firdaus - 18222015	<ul style="list-style-type: none"> - Membuat fitur QUIT - Membuat fitur HELP - Membuat fitur INVALID COMMAND - Membuat fitur STATUS - Mencetuskan nama SPOTIFILE - Membuat laporan bagian Algoritma Utama - Membuat laporan bagian Daftar Isi - Membuat laporan bagian ADT Map, ADT Mesin Karakter - Membuat Makefile
2	David Dewanto - 18222027	<ul style="list-style-type: none"> - Membuat fitur PLAYLIST - Membuat fitur LOAD - Membuat fitur SAVE - Membuat Driver ADT Linked-List, Queue, Stack - Membuat ADT Linked-List, Queue, Stack - Membuat searchprint - Membantu merapikan laporan - Membuat Makefile - Membantu proses debugging - Membantu handling Command dan ID Playlist Error - Membuat file konfigurasi
3	Muhammad Rafi Dhiyaulhaq - 18222069	<ul style="list-style-type: none"> - Membuat fitur LIST DEFAULT - Membuat fitur LIST PLAYLIST - Membuat laporan bagian Daftar Isi

		<ul style="list-style-type: none"> - Membuat laporan bagian Ringkasan - Membuat laporan bagian ADT (Linked-List, List, Mesin Kata, Queue, Set, dan Stack) - Membuat laporan bagian Program Utama - Membuat laporan bagian Data Test (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, PLAYLIST REMOVE, PLAYLIST DELETE, STATUS, SAVE, QUIT, HELP, INVALID COMMAND) - Membuat laporan bagian Test-Script - Membuat laporan bagian Lampiran (Deskripsi Tugas Besar dan Notulensi Rapat) - Menjadwalkan asistensi dan mencatat Notulensi Rapat - Membuat formating laporan, mengecek KBBI, serta merapihkan laporan - Membuat fitur README - Membuat laporan bagian Data Test ENHANCE - Membuat laporan bagian Data Test MULTIUSER - Membuat laporan bagian Data Test RADIO
4	Alvin Fadhilah Akmal - 18222079	<ul style="list-style-type: none"> - Membuat fitur PLAY - Membuat fitur SONG - Melakukan debug pada fungsi Play, Song, Queue, dan main program - Melakukan tes program menggunakan powershell - Melakukan tes program menggunakan WSL-Ubuntu - Membuat struktur data NowPlaying - Merapikan dan mengisi laporan bagian Data Test dan Test Script - Membuat laporan bagian Struktur data NowPlaying dan UserSession
5	Timotius Vivaldi Gunawan - 18222091	<ul style="list-style-type: none"> - Membuat repository github - Membagi pembagian tugas fungsi - fungsi - Membuat dan merapikan driver ADT Set, Map, Array, Array dinamik, Graph, Mesin kata, Mesin karakter - Membuat fitur START / LOAD - Membuat fitur LIST - Membuat fitur QUEUE - Membuat fitur WelcomeMenu - Membuat main program dan handling setiap inputan dari user - Membuat fitur bonus ENHANCE - Membuat fitur bonus MULTIUSER - Merapikan SAVE, LOAD, START untuk MULTIUSER

		<ul style="list-style-type: none"> - Membuat fitur bonus RADIO - Merapikan makefile untuk setiap fungsi - Melakukan debug pada fungsi - fungsi yang salah - Melengkapi selektor Search-Print - Melakukan tes program menggunakan WSL-Ubuntu - Menambahkan Laporan bagian algoritma menarik dan fitur - fitur tambahan - Merapikan struktur laporan - Merapikan branches dan merging commits pada github
--	--	---

9 Lampiran

9.1 Deskripsi Tugas Besar

WayangWave adalah sebuah aplikasi pemutar musik layaknya *walkman*. WayangWave adalah aplikasi yang dapat meluluhkan hati Roro. WayangWave dapat dikatakan aplikasi yang dapat menggantikan perangkat lunak di *walkman* Roro

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 Mechanics

1. About the System

WayangWave merupakan sebuah program di dalam 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

Pemain dapat memasukkan *command* berikut:

- **START** : Salah satu *command* yang dimasukkan pertama kali oleh pemain ke WayangWave. Setelah menekan Enter, dibaca *file* konfigurasi default yang berisi list game yang dapat dimainkan.

- **LOAD <filename>** : Salah satu *command* yang dimasukkan pertama kali oleh pemain ke WayangWave. Memiliki satu argumen yaitu filename yang merepresentasikan suatu *save file* yang ingin dibuka. Setelah menekan Enter, akan dibaca *save file <filename>* yang berisi list game yang dapat dimainkan, histori dan scoreboard game.
- **LIST DEFAULT**: *Command* 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.
- **LIST PLAYLIST**: *Command* LIST PLAYLIST digunakan untuk menampilkan *playlist* yang ada pada pengguna.
- **PLAY SONG**: *Command* PLAY SONG digunakan untuk memainkan lagu berdasarkan masukan nama penyanyi, nama album, dan id lagu. Ketika *command* ini berhasil dieksekusi, queue dan riwayat lagu akan menjadi kosong.
- **PLAY PLAYLIST**: *Command* PLAY PLAYLIST digunakan 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*.
- **QUEUE SONG**: *Command* QUEUE SONG digunakan 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
- **QUEUE PLAYLIST**: *Command* QUEUE PLAYLIST digunakan untuk menambahkan lagu yang ada dalam playlist ke dalam queue. *Command* ini menerima input dari id playlist yang ingin dimasukkan ke dalam queue.
- **QUEUE SWAP**: *Command* QUEUE SWAP digunakan untuk menukar lagu pada urutan ke *x* dan juga urutan ke *y*.
- **QUEUE REMOVE <id>**: *Command* QUEUE REMOVE digunakan untuk menghapus lagu dari queue. *Command* ini menerima input berupa urutan lagu (**id**) yang ingin dihapus dari queue.
- **QUEUE CLEAR**: *Command* QUEUE CLEAR digunakan untuk mengosongkan queue.
- **SONG NEXT**: *Command* SONG NEXT digunakan 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.
- **SONG PREVIOUS**: *Command* SONG PREVIOUS digunakan 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.
- **PLAYLIST CREATE**: *Command* PLAYLIST CREATE digunakan untuk membuat playlist baru dan ditambahkan pada daftar playlist pengguna. Keadaan

awal playlist adalah kosong. Nama playlist dapat sama dengan playlist yang sudah ada.

- **PLAYLIST ADD:** *Command* PLAYLIST ADD digunakan 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.
- **PLAYLIST SWAP <id> <x> <y>:** *Command* PLAYLIST SWAP digunakan untuk menukar lagu pada urutan ke *x* dan juga urutan ke *y* di playlist dengan urutan ke **id**.
- **PLAYLIST REMOVE <id> <n>:** *Command* PLAYLIST REMOVE digunakan untuk menghapus lagu dengan urutan **n** pada playlist dengan index **id**.
- **PLAYLIST DELETE:** *Command* PLAYLIST DELETE digunakan untuk melakukan penghapusan suatu *existing* playlist dalam daftar playlist pengguna. Tampilkan pesan *error* apabila masukkan pengguna tidak valid pada setiap permintaan masukkan.
- **STATUS:** STATUS merupakan *command* yang digunakan untuk menampilkan lagu yang sedang dimainkan beserta Queue song yang ada dan dari playlist mana lagu itu diputar.
- **SAVE <filename>:** SAVE merupakan *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.
- **QUIT:** QUIT merupakan *command* yang digunakan untuk keluar dari aplikasi WayangWave.
- **HELP:** HELP merupakan *command* yang digunakan menampilkan daftar *command* yang mungkin untuk dieksekusi dengan deskripsinya. Penjelasan dari deskripsi dibebaskan selama masih mendeskripsikan *command* sesuai dengan spek.
- **<INVALID COMMAND>:** *Command* selain yang disebutkan di atas dinyatakan akan tidak valid dan hanya akan mengeluarkan teks error.

Konfigurasi Aplikasi:

Dalam memulai setiap session, akan dibaca sebuah file konfigurasi oleh aplikasi yang akan berlaku di session tersebut. File konfigurasi yang harus dapat dibaca itu sendiri terdapat dua jenis yang berbeda. Masing-masing mewakili sebuah state yang berbeda, yaitu **(A)** state default ketika

suatu session baru benar-benar dimulai; dan **(B)** state ketika suatu session sudah pernah berjalan dan state-nya di-save ke dalam file kemudian ingin di-load kembali.

Daftar ADT:

Berikut daftar ADT yang wajib digunakan:

- ADT LIST: ADT ini digunakan dengan implementasi array yang terbagi berdasarkan tipe pengalokasiannya, yaitu statis dan dinamis.
- ADT MesinKarakter dan MesinKata: ADT ini digunakan untuk melakukan *parsing command* dalam aplikasi dan membaca ataupun membuat file konfigurasi dari aplikasi.
- ADT QUEUE: ADT ini digunakan untuk merepresentasikan urutan lagu yang akan dimainkan dalam aplikasi.
- ADT STACK: ADT ini digunakan untuk merepresentasikan urutan lagu yang telah dimainkan dalam aplikasi, dimana TOP merupakan lagu yang telah dimainkan.
- ADT Set & Map:: ADT Set digunakan untuk menyimpan lagu dari suatu album. Dengan begitu, lagu yang ada dipastikan tidak memiliki duplikat. ADT Map digunakan untuk menyimpan album untuk tiap penyanyi dan juga menyimpan lagu untuk tiap album.
- ADT List dengan Struktur Berkait: ADT ini digunakan untuk merepresentasikan sebuah *playlist* dimana *playlist* merupakan suatu kumpulan lagu yang kontigu.

9.2 Notulen Rapat

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

No. Kelompok /Kelas : 01
Nama Kelompok : Kelompok Spotifile
Anggota Kelompok (Nama/NIM) :

- | | |
|------------------------------|-----------|
| 1. Bihurin Salsabila Firdaus | /18222015 |
| 2. David Dewanto | /18222027 |
| 3. Muhammad Rafi Dhiyaulhaq | /18222069 |
| 4. Alvin Fadhilah Akmal | /18222079 |
| 5. Timotius Vivaldi Gunawan | /18222091 |


Asisten Pembimbing : Kak M Syahrul Surya Putra
Asistensi 1

Tanggal : 1 November 2023	Catatan Asistensi: 1. Value inti berada di Album.
Tempat : Zoom	
Kehadiran Anggota Kelompok: No	

NIM
Tanda tangan
1
18222015



2
18222027



DAVID DEWANTO

3
18222069




4
18222079



5
18222091



2. SAVE akan membuat file baru atau *overwrite*, semua data disimpan di file konfigurasi
3. Mesin kata dan mesin karakter harus dicek kembali karena apabila tidak jalan maka program akan error.
4. Untuk PLAY, QUEUE yang ada dikosongkan lalu diisi dengan PLAYLIST kecuali lagu yang paling awal karena lagu yang paling awal sebagai PLAY NOW (seperti mensimulasikan Spotify).
5. PLAY NOW boleh disimpan di suatu variabel di MAIN.
6. Kalau PLAYLIST di-back maka memutar lagu PLAYLIST paling akhir, jika pakai double linked-list masih bisa (alternatif).
7. PLAY PLAYLIST yang dijadikan PLAY NOW yang paling pertama, yang dimasukkan QUEUE adalah 2 sampai ujung. Yang dijadikan STACK dan di-reverse adalah dari 2 sampai *end* kemudian di-reverse(dimasukkan ke *stack*). Asumsinya seperti lagunya sudah dimainkan, namun kalau kosong dibuat agar kalau dimainkan PLAYLIST diisi QUEUE dan di-reverse.
8. PLAY SONG setelahnya kosong, QUEUE dan STACK juga sama. PLAY PREVIOUS juga kosong.
9. makefile tidak mesti di setiap ADT, simpan makefile di main folder dan kalian buat fungsi run atau yang semacamnya dan yang menerima input ini deliver ke folder untuk di-compile jadi tidak perlu makefile di masing-masing folder. makefile better satu saja di parent folder

	(betternya di parent folder untuk meng compile folder dan driver). Sehingga tidak perlu pusing navigasi folder.
	Tanda Tangan Asisten: 

Asistensi II

Tanggal : 14 November 2023	Catatan Asistensi:
Tempat : Zoom	<ol style="list-style-type: none"> 1. Secara keseluruhan coding sudah aman dan valid secara format. 2. Untuk LIST DEFAULT, boleh input Album yang tidak ada pada spesifikasi. 3. Bonus ENHANCE harus ada isinya, playlist tertentu saja. 4. Selama file konfigurasi sama, maka aman-aman saja. 5. Untuk lagu, jika setelahnya kosong, maka lagu yang akan dimainkan selanjutnya adalah lagu yang sekarang dan riwayatnya tetap sama. 6. QUEUE dibebaskan dan tidak dibatasi. 7. Untuk BONUS LOGIN secara program tidak bisa dibuat yang baru, jumlah user yang mutlak dan tidak bisa dirubah. 8. Secara jumlah yang kalian buat di konfigurasi itu tidak apa-apa tapi perlu diingat kalau file confignya berbeda apakah bisa dijalankan atau tidak.

Kehadiran Anggota Kelompok:

No

NIM

Tanda tangan

1

18222015



2

18222027



DAVID DEWANTO

3

18222069



4


18222079



5

18222091



	Tanda Tangan Asisten: 
--	--

9.3 Log Activity Anggota Kelompok

No	Tanggal	NIM	Nama	Aktivitas
	25 Oktober 2023	18222069	Muhammad Rafi Dhiyaulhaq	- Menjadwakan asistensi pertama
	28 Oktober 2023	18222091	Timotius Vivaldi Gunawan	-membuat template awal tugas besar (struktur program dan ADT)
	28 Oktober 2023	18222027	David Dewanto	- Membuat ADT Linked-List, Queue, Stack
	30 Oktober 2023	18222015	Bihurin Salsabila Firdaus	- Membuat fitur QUIT - Membuat fitur HELP - Membuat fitur INVALID COMMAND - Mencetuskan nama SPOTIFILE
	1 November	18222069	Muhammad Rafi Dhiyaulhaq	-Melakukan Asistensi pertama dan mencatat hasil Asistensi
	7 November 2023	18222069	Muhammad Rafi Dhiyaulhaq	-Membuat fitur LIST DEFAULT - Membuat fitur LIST PLAYLIST - Menjadwakan asistensi kedua

	12 November 2023	18222079	Alvin Fadhilah Akmal	- Membuat algoritma Play Song dan Play Playlist
	13 November 2023	18222015	Bihurin Salsabila Firdaus	- Membuat fitur STATUS
	13 November 2023	18222027	David Dewanto	- Membuat fitur STARTLOAD, PLAYLIST, searchprint
	13 November 2023	18222091	Timotius Vivaldi Gunawan	-Membuat Queue dan dan merapihkan LIST
	13 November 2023	18222079	Alvin Fadhilah Akmal	- Membuat ADT NowPlaying - Update algoritma pada Play
	14 November 2023	18222069	Muhammad Rafi Dhiyaulhaq	-Melakukan Asistensi kedua dan mencatat hasil Asistensi
	14 November 2023	18222091	Timotius Vivaldi Gunawan	-Membuat fitur ENHANCE
	14 November 2023	18222079	Alvin Fadhilah Akmal	- Membuat algoritma Song Next dan Song Previous - Update pada Play dan ADT NowPlaying
	16 November 2023	18222079	Alvin Fadhilah Akmal	- Update pada Play
	18 November 2023	18222015	Bihurin Salsabila Firdaus	- Membuat laporan bagian Algoritma Utama
	18 November 2023	18222079	Alvin Fadhilah Akmal	- Update pada Song

	18 November 2023	18222069	Muhammad Rafi Dhiyaulhaq	<ul style="list-style-type: none"> - Membuat formatting laporan -Membuat laporan bagian Daftar Isi -Membuat laporan bagian Ringkasan -Membuat laporan bagian ADT (Linked-List, List, Mesin Kata, Queue, Set, dan Stack) -Membuat laporan bagian Program Utama -Membuat laporan bagian Data Test -Membuat laporan bagian Test-Script -Membuat laporan bagian Lampiran (Deskripsi Tugas Besar dan Notulensi Rapat)
	18 November 2023	18222027	David Dewanto	Membuat fitur SAVE
	21 November 2023	18222015	Bihurin Salsabila Firdaus	<ul style="list-style-type: none"> - Membuat laporan bagian Daftar Isi - Membuat laporan bagian ADT Map, ADT Mesin Karakter
	21 November 2023	18222091	Timotius Vivaldi Gunawan	-Merapihkan semua ADT + Membuat fitur MULTIUSER dan RADIO
	21 November 2023	18222079	Alvin Fadhilah Akmal	<ul style="list-style-type: none"> - Debug pada fungsi Queue - Update pada main
	24 November 2023	18222069	Muhamamd Rafi	- Membuat Data

			Dhiyaulhaq	Test ENHANCE - Membuat laporan bagian Data Test RADIO - Membuat Data Test MULTIUSER - Mengecek KBBI laporan dan merapihkan laporan
	24 November	18222027	David Dewanto	Melakukan debugging dan membenaran kode program utama
	25 November	Semua Anggota	Semua Anggota	Finalisasi laporan