

LAPORAN PRAKTIKUM



Kelas IF232 - BL

PROGRAM STUDI TEKNIK INFORMATIKA

UNIVERSITAS MULTIMEDIA NUSANTARA

2023

Disusun oleh :

00000077138 - Jonathan Prasetyo

00000078383 - Hexsel Archiles Virgio Manik

00000077041 - Jonathan Sutandar

00000077134 - Edric Hugo

KAJIAN TEORI

A. Array

Struktur data yang digunakan untuk menyimpan sekumpulan elemen dengan tipe data yang sama dalam satu variabel. Setiap elemen dalam array diidentifikasi oleh indeksinya, yang dimulai dari 0 hingga $(n-1)$, dimana 'n' adalah ukuran array. Array memungkinkan penyimpanan data secara terstruktur dan memungkinkan akses efisien ke setiap elemennya. Dengan menggunakan array, kita dapat mengelompokkan data terkait dan mengaksesnya dengan mudah menggunakan indeks.

B. Linked List

Struktur data linier yang terdiri dari serangkaian simpul atau node yang saling terhubung. Setiap simpul menyimpan data dan sebuah pointer yang menunjuk ke simpul berikutnya dalam urutan. Linked list tidak memerlukan alokasi memori kontinu seperti array, sehingga memungkinkan penambahan dan penghapusan elemen dengan mudah. Terdapat beberapa jenis linked list, termasuk single linked list, double linked list, dan circular linked list, yang masing-masing memiliki karakteristik dan penggunaan yang berbeda.

C. Stack

Struktur data linier yang mengikuti prinsip Last In First Out (LIFO). Ini berarti elemen yang terakhir dimasukkan ke dalam stack akan menjadi yang pertama dikeluarkan. Stack dapat diimplementasikan menggunakan array atau linked list. Operasi dasar pada stack meliputi push (menambahkan elemen ke atas stack), pop (menghapus elemen dari atas stack), dan peek (mengakses elemen teratas tanpa menghapusnya). Stack umumnya digunakan dalam pemrograman untuk mengelola pemanggilan fungsi (call stack), evaluasi ekspresi aritmatika, dan dalam algoritma seperti penelusuran grafik (depth-first search).

D. Queue

Struktur data FIFO yang mirip dengan antrian kehidupan nyata. Ini digunakan untuk penjadwalan tugas dan pengelolaan aliran data. Operasi dasar termasuk Enqueue untuk menambahkan, Dequeue untuk menghapus, dan Front untuk mendapatkan elemen pertama tanpa menghapusnya. Queue dapat diimplementasikan menggunakan array atau linked list, dengan linked list memungkinkan penambahan dan penghapusan yang lebih efisien. Kelebihannya termasuk mempertahankan urutan elemen, sementara kekurangannya adalah kompleksitas waktu terkait dengan operasi enqueue dan dequeue. Pemahaman tentang queue dan implementasinya dalam bahasa C memungkinkan pengembang untuk membuat program yang lebih efisien dan terstruktur.

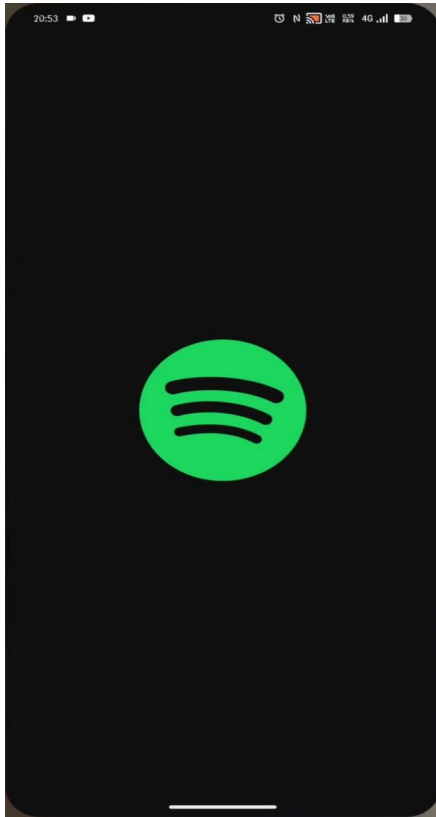
STUDI KASUS

A. Judul

Seorang pengguna sedang mendengarkan lagu di sebuah aplikasi bernama *Spotify*. Sebelum masuk ke dalam sebuah aplikasi *Spotify* pengguna diperlukan untuk **“login”** atau **“register”** kalau belum memiliki akun. Karena sebelumnya ia sudah pernah menggunakan *spotify* dan sudah memiliki akun, ia hanya perlu login saja. Ia ingin **“membuat sebuah playlist”** yang isinya lagu lagu yang ia sukai. lalu ia **“menambahkan lagu lagu ke dalam playlist yang sudah di buat”** lagu yang dimasukan ke dalam playlist adalah lagu barat dengan genre bebas. Pada suatu ketika ia memasukan lagu dengan bahasa Jawa, yang tidak sama/sesuai dengan lagu lagu lain, lalu ia memutuskan untuk **“menghapus lagu yang ada di dalam playlist tersebut.”** sehingga isi dari playlist menjadi sesuai kembali. Ia merasa ingin mengeksplor lagi lagu lagu yang ia belum ketahui sebelumnya, ia mencoba **“memainkan lagu yang ada di beranda”** karena lagunya tidak sesuai dengan selera nya ia **“memencet tombol next”** untuk mencari lagu yang sesuai dengannya. Setelah menemukan lagu yang ia sukai ia **“memasukan lagu tersebut ke dalam playlist”** tetapi ia merasa selama ia memutar lagu di beranda ia merasa melewatkan lagu yang ternyata ia sukai juga, lalu ia **“memencet tombol previous”**. Setelah menemukan lagu yang ia baru temukan dan ternyata ia sukai ia memutuskan untuk **“memasukan lagu tersebut ke dalam playlist”**. setelah ia merasa bahwa playlistnya sudah cukup banyak lagu dan isinya lagu lagu yang ia sukai semua. Ia memutuskan untuk **“memainkan lagu yang ada di dalam playlistnya”**. Karena pada beberapa waktu yang lalu telah diselenggarakan konser Taylor Swift di indonesia, sebelumnya ia tidak tahu apa itu Taylor Swift karena *hype* yang terlalu besar di indonesia ia **“mencari lagu Taylor Swift di *Spotify*”**. Karena pada sebelumnya ia pernah membuat beberapa playlist lain yang berisi lagu lagu yang bergenre pop. ia **“memilih playlist tersebut”** untuk di tambahkan dengan lagu yang baru ia dengar. Setelah menambahkan lagu ia memainkan playlist tersebut dengan cara **“shuffle/acak”**.

FITUR SPOTIFY

1. Logo



Menampilkan Logo Spotify

2. Sign Up/Create Acc

A screenshot of the Spotify sign-up screen. The background is black. At the top left is the Spotify logo. The main heading is "Sign up to start listening" in white. Below it is a form with a label "Email address" and a text input field containing "name@domain.com". A link "Use phone number instead" is below the input field. A green "Next" button is below the link. Below the button is a horizontal line with the word "or" in the center. Under the line are three buttons: "Sign up with Google" (with the Google logo), "Sign up with Facebook" (with the Facebook logo), and "Sign up with Apple" (with the Apple logo). At the bottom, there is a link "Already have an account? Log in here".

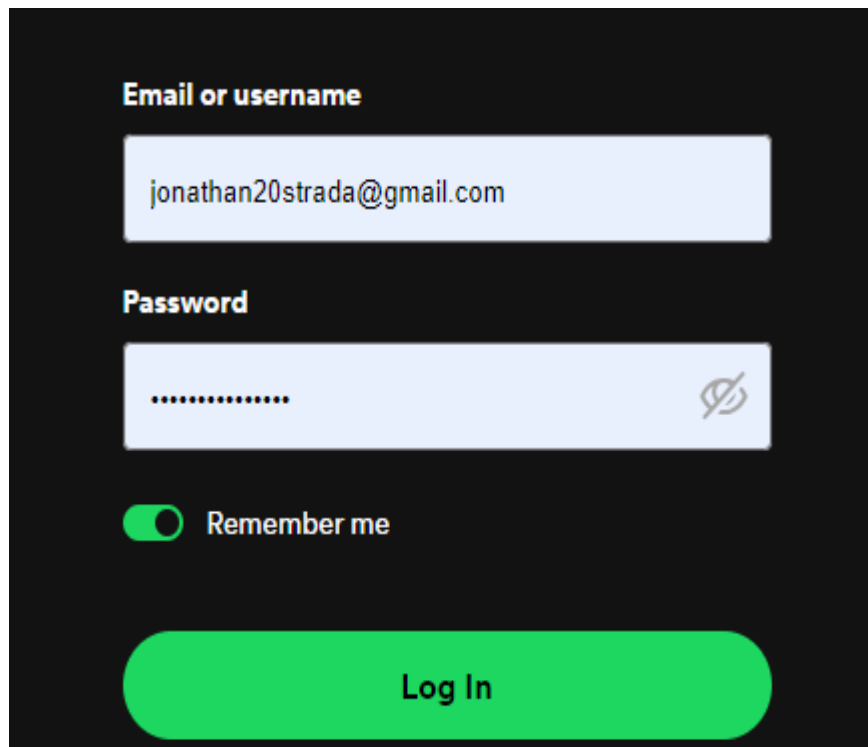
Fitur ini memungkinkan pengguna yang belum memiliki akun untuk membuat akun di aplikasi spotify

Tipe Data:

Email Address (String).

Password (String).

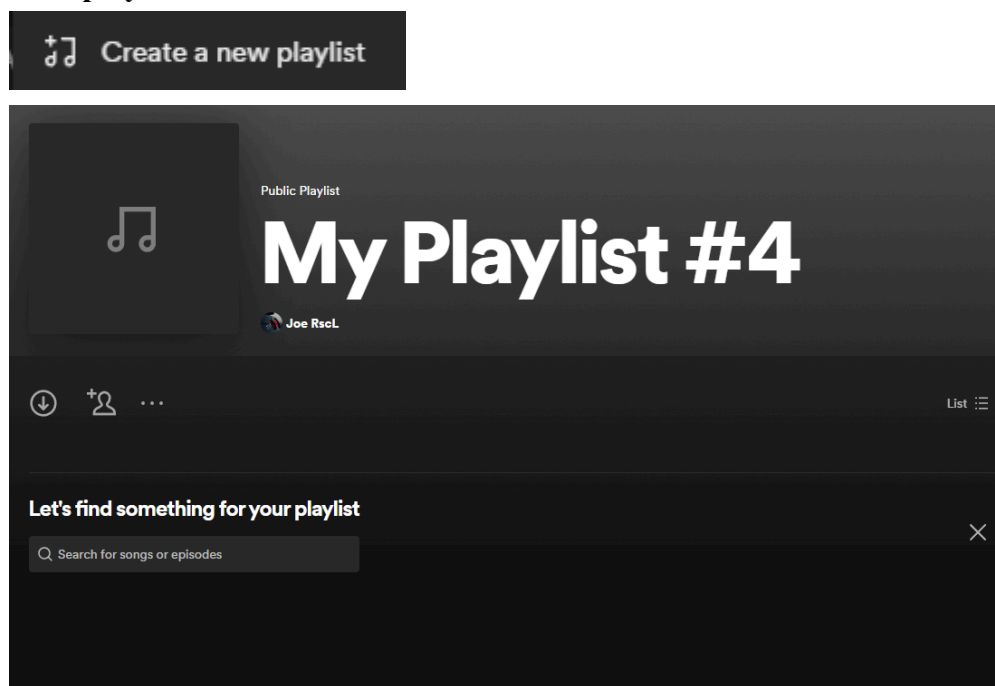
3.Login



The login form is displayed on a dark background. It features two light blue input fields. The first field is labeled "Email or username" and contains the text "jonathan20strada@gmail.com". The second field is labeled "Password" and contains a series of dots, with a toggle icon on the right. Below the password field is a "Remember me" toggle switch, which is currently turned on. At the bottom of the form is a large, rounded, orange button labeled "Log In".

fitur ini meminta user melakukan input berupa username atau email, dan memasukkan password, dan jika password yang di input salah maka tidak akan dapat memasuki program. Tipe data : Username (string), Password (string).

4. Membuat playlist

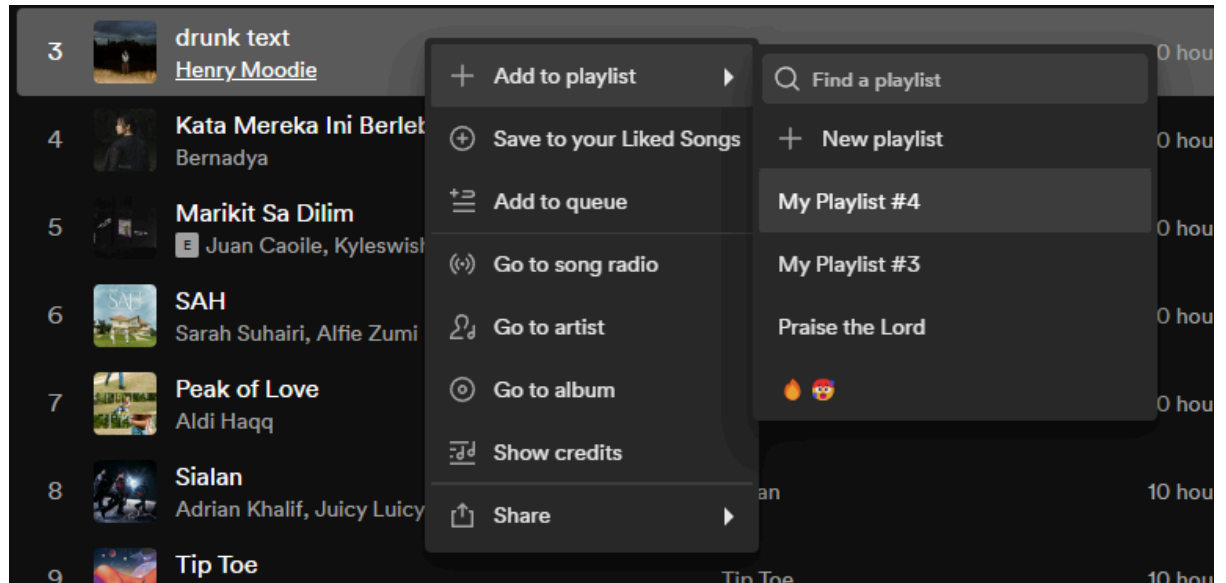


The interface for creating a new playlist is shown. At the top, there is a dark blue button with a plus icon and the text "Create a new playlist". Below this, the main area has a dark background. On the left, there is a square placeholder for a playlist cover with a music note icon. To the right of the cover, the text "Public Playlist" is visible above the large title "My Playlist #4". Below the title, the name "Joe RscL" is displayed. At the bottom of the main area, there is a search bar with the placeholder text "Search for songs or episodes". Above the search bar, there is a prompt "Let's find something for your playlist". To the right of the search bar, there is a close icon (X). On the far right, there is a "List" button with a menu icon.

Fitur ini memungkinkan user untuk membuat playlist baru

Tipe Data : Judul Playlist(string)

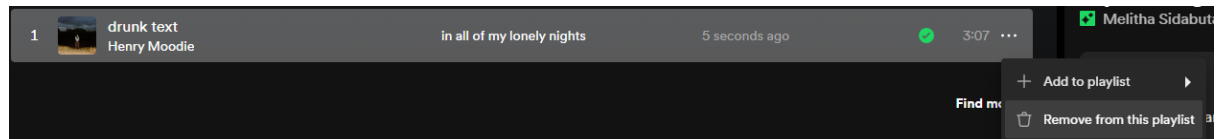
5.Tambah lagu ke playlist



Fitur ini memungkinkan pengguna atau user untuk menambahkan lagu ke dalam playlist yang sudah ada.

Tipe data : judul lagu (string), nama penyanyi (string), name playlist (string).

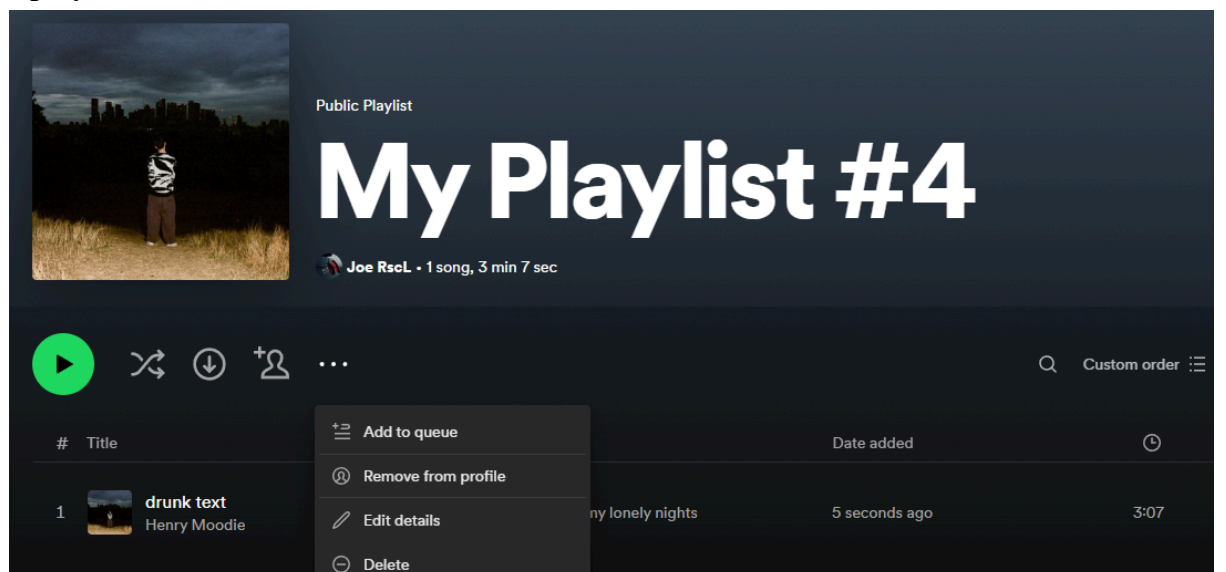
6.Menghapus lagu yang ada di playlist



Fitur ini memungkinkan pengguna untuk menghapus lagu dari playlist yang sudah ada.

Tipe data : judul lagu (string), nama penyanyi (string), name playlist (string).

7.Hapus playlist



Fitur ini berfungsi untuk menghapus playlist yang sudah ada
Tipe data: Nama Playlist (string).

8. Memutar lagu selanjutnya



Berfungsi untuk memutar atau memainkan lagu selanjutnya berdasarkan urutan dalam playlist yang ada.

Tipe Data: Judul lagu (string), Nama Penyanyi (string).

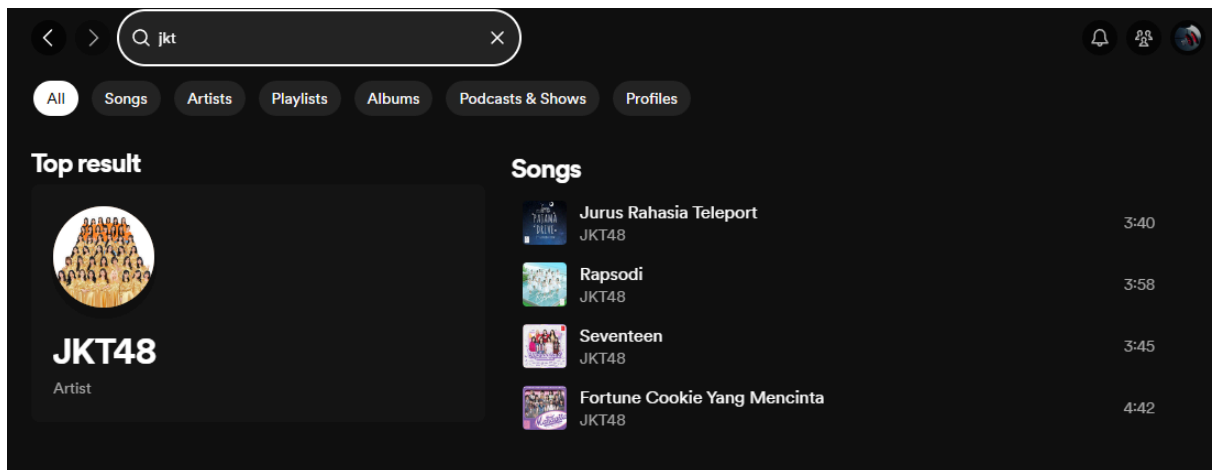
9. Memutar lagu sebelumnya



Berfungsi untuk memutar atau memainkan lagu sebelumnya berdasarkan urutan dalam playlist yang ada.

Tipe Data: Judul lagu (string), Nama Penyanyi (string).

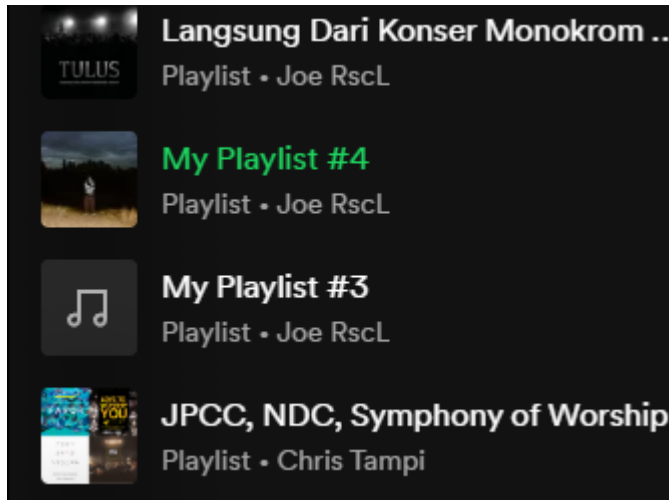
10. Cari lagu



Berfungsi untuk mencari lagu berdasarkan keyword, dapat berupa nama album, nama penyanyi, maupun judul lagu

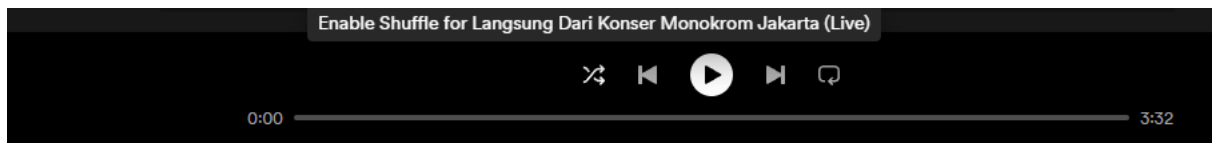
Tipe Data : Judul lagu (string), Nama Album (string), Nama Penyanyi (string)

11. Pilih playlist



Berguna untuk menampilkan serta membuka playlist yang user miliki, jika telah memilih playlist maka akan menampilkan isi dari playlistnya
Tipe Data : Nama Playlist (string), Judul Lagu (string).

12. Shuffle lagu



Berfungsi untuk mengacak urutan lagu saat diputar.
Tipe Data : Judul Lagu (string), nama penyanyi (string).

LINKED LIST

1. Logo

```
file = fopen(logs, "r");
createData(file, baris);
printf("\n");
fclose(file);
```

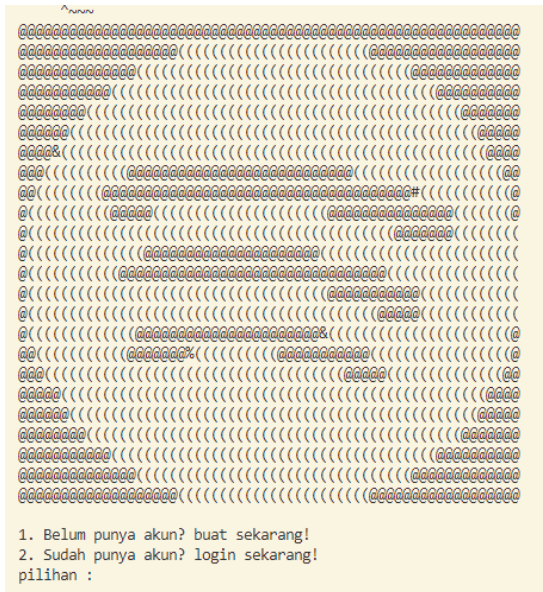
```
void createData(FILE *file, char baris[1000]) {
    char karakter;
    while (fgetc(baris, sizeof(baris), file) != NULL) {
        printf("%s", baris);
    }
}
```

Menampilkan logo spotify dengan cara fopen zlogo.txt
File processing menggunakan access mode "r".

isi zlogo.txt :

[illegible]

Output:



2.Create Account

```
void addAcc(struct akun **head, const char *username, const char *password) {
    struct akun *node = (struct akun*)malloc(sizeof(struct akun));
    strcpy(node->username, username);
    strcpy(node->password, password);
    node->next = *head;
    *head = node;

    FILE *file = fopen("zdatabase.txt", "a");
    if (file == NULL) {
        printf("Error membuka file database.\n");
        return;
    }
    fprintf(file, "%s %s\n", username, password);
    fclose(file);
}
```

```
struct akun {
    char username[30];
    char password[30];
    struct akun *next;
};
```

Kode ini memiliki cara kerja dengan cara melakukan file processing dengan access “a” yang berfungsi memasukan username dan password ke dalam file database.txt, pada proses ini

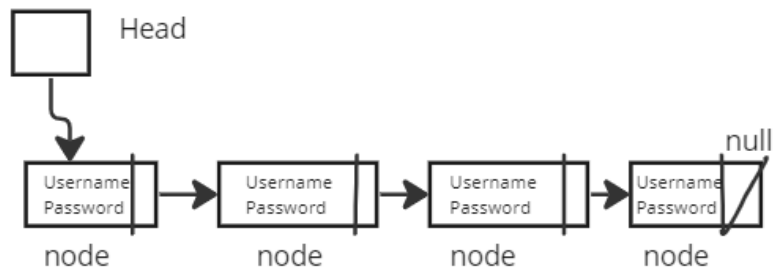
Output :

```
1. Belum punya akun? buat sekarang!  
2. Sudah punya akun? login sekarang!  
pilihan :1  
=====Register=====  
Username : aa  
  
Password : aa
```

isi txt :

```
zdatabase.txt  
1   Joko qwer  
2   Joko 123  
3   joko joko  
4   1 1  
5   jingu jingu  
6   aa aa  
7
```

Linked List :



3.Login

```
login(int kondisi, char username[30],char password[30]) {  
    struct akun *akunHead = NULL;  
    readDatabase(&akunHead);  
    printf("1. Belum punya akun? buat sekarang!\n");  
    printf("2. Sudah punya akun? login sekarang!\n");  
    printf("pilihan :"); scanf("%d", &kondisi);  
  
    if (kondisi == 1) {  
        printf("=====Register=====\\n");  
        printf("Username : ");  
        scanf("%s", username);  
        printf("\\nPassword : ");  
        scanf("%s", password);  
        addAcc(&akunHead, username, password);  
    } else if (kondisi == 2) {  
        printf("=====Login=====\\n");  
        printf("Username : ");  
        scanf("%s", username);  
        printf("\\nPassword : ");  
        scanf("%s", password);  
        if (!cekAkun(akunHead, username, password)) {  
            printf("Username atau password salah.\\n");  
            return main();  
        }  
    }  
}
```

```
bool cekAkun(struct akun *head, const char *username, const char *password) {  
    struct akun *current = head;  
    while (current != NULL) {  
        if (strcmp(current->username, username) == 0 && strcmp(current->password, password) == 0) {  
            return true;  
        }  
        current = current->next;  
    }  
    return false;  
}
```

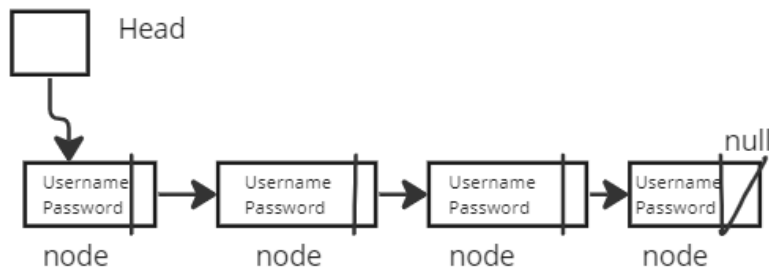
Kode ini berfungsi untuk melakukan pengecekan dari username dan password yang ada pada database.txt., jika salah maka akan meminta user untuk menginput ulang username dan password nya

Tipe Data : Username (string), Password (string).

Output :

```
1. Belum punya akun? buat sekarang!  
2. Sudah punya akun? login sekarang!  
pilihan :2  
=====Login=====  
Username : 1  
Password : 1
```

Linked List:



4. Membuat playlist

```
void createPlaylist() {
    int songLength = 100;
    char titleLength = 50;
    char artistLength = 50;
    char albumLength = 50;
    int numSongs;
    FILE *file;

    struct playlist *head = NULL;
    struct playlist *tail = NULL;

    printf("1. Buat playlist baru\n");
    printf("2. Tambahkan lagu ke playlist yang ada\n");
    printf("Pilihan: ");
    int choice;
    scanf("%d", &choice);

    if (choice == 1) {
        createCustomPlaylist();
    }

    printf("Masukkan jumlah lagu yang ingin ditambahkan ke playlist: ");
    scanf("%d", &numSongs);
    getchar();

    file = fopen("playlist.txt", "a");

    if (file == NULL) {
        printf("Gagal membuka file.\n");
        return;
    }

    for (int i = 0; i < numSongs; i++) {
        struct playlist *node = (struct playlist*)malloc(sizeof(struct playlist));

        printf("Masukkan judul lagu ke-%d: ", i + 1);
        fgets(node->judul, titleLength, stdin);
        strtok(node->judul, "\n");

        printf("Masukkan nama penyanyi untuk lagu ke-%d: ", i + 1);
        fgets(node->penyanyi, artistLength, stdin);
        strtok(node->penyanyi, "\n");

        printf("Masukkan nama penyanyi untuk lagu ke-%d: ", i + 1);
        fgets(node->album, albumLength, stdin);
        strtok(node->album, "\n");

        printf("Masukkan tahun rilis untuk lagu ke-%d: ", i + 1);
        scanf("%d", &node->tahun);
        getchar();

        node->next = NULL;
        node->prev = tail;

        if (head == NULL) {
            head = node;
            tail = node;
        } else {
            tail->next = node;
            tail = node;
        }

        fprintf(file, "%s%s%d\n", node->judul, node->penyanyi, node->album, node->tahun);
    }

    printf("Playlist berhasil ditambahkan ke file.\n");

    fclose(file);
}
```

```

void createCustomPlaylist() {
    char playlistName[50];
    printf("Masukkan nama playlist baru: ");
    scanf("%s", playlistName);

    FILE *file = fopen(strcat(playlistName, ".txt"), "a");
    if (file == NULL) {
        printf("Gagal membuat playlist.\n");
        return;
    }

    printf("Playlist '%s' berhasil dibuat.\n", playlistName);
    fclose(file);
}

```

Void CreatePlaylist berfungsi untuk memungkinkan user membuat sebuah playlist baru, cara kerja kode ini secara singkat adalah kode ini akan membuat file dan membukanya dengan kode access “a”, kemudian program akan meminta user untuk menginput nama playlist, jumlah lagu yang ingin ditambahkan, nama penyanyi, judul lagu, nama album, dan tahun rilis, kemudian akan dimasukan ke dalam file playlist

Output:

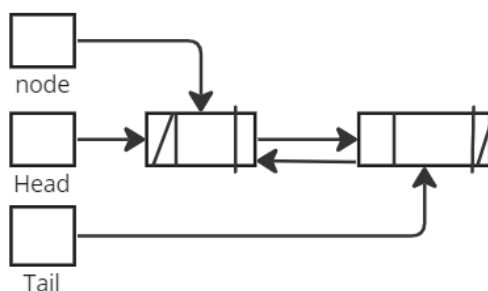
```

=====Home=====
Hello, 1
1. Search by Keyword
2. Choose Playlist
3. Display All Song
4. Create playlist
5. Play Song
6. Logout
Pilihan : 4
1. Buat playlist baru
2. Tambahkan lagu ke playlist yang ada
Pilihan: 1
Masukkan nama playlist baru: testplaylist
Playlist 'testplaylist' berhasil dibuat.
Masukkan jumlah lagu yang ingin ditambahkan ke playlist: 1
Masukkan judul lagu ke-1: test
Masukkan nama penyanyi untuk lagu ke-1: penyanyi
Masukkan tahun rilis untuk lagu ke-1: 2001
Playlist berhasil ditambahkan ke file.

```

Linked List:

Insert dari belakang



5. Tambah lagu ke playlist

```
void addSongToPlaylist(const char *playlistFilename) {  
    FILE *file = fopen(playlistFilename, "a");  
    if (file == NULL) {  
        printf("Gagal membuka playlist.\n");  
        return;  
    }  
  
    struct playlist newSong;  
  
    printf("Masukkan judul lagu: ");  
    fgets(newSong.judul, sizeof(newSong.judul), stdin);  
    strtok(newSong.judul, "\n");  
  
    printf("Masukkan nama penyanyi: ");  
    fgets(newSong.penyanyi, sizeof(newSong.penyanyi), stdin);  
    strtok(newSong.penyanyi, "\n");  
  
    printf("Masukkan album: ");  
    fgets(newSong.album, sizeof(newSong.album), stdin);  
    strtok(newSong.album, "\n");  
  
    printf("Masukkan tahun rilis: ");  
    scanf("%d", &newSong.tahun);  
    getchar();  
  
    fprintf(file, "%s#%s#%s#(%d)\n", newSong.judul, newSong.penyanyi, newSong.album, newSong.tahun);  
  
    fclose(file);  
  
    printf("Lagu berhasil ditambahkan ke playlist.\n");  
}
```

Kode ini berfungsi untuk menambahkan lagu ke dalam playlist yang dipilih menggunakan file processing dengan kode access “a”.

```
=====
```

No	Judul	Penyanyi	Album	Tahun
0	abede	test	test	2123

```
=====
```

Pilih opsi:
1. Putar Lagu
2. Tambahkan Lagu ke Playlist
3. Hapus Lagu dari Playlist
4. Kembali ke Home
Pilihan: 2
Masukkan judul lagu: test
Masukkan nama penyanyi: test
Masukkan album: test
Masukkan tahun rilis: 2122
Lagu berhasil ditambahkan ke playlist.

```
playlist.txt  
1  abede#test#test#(2123)  
2  test#test#test#(2122)  
3
```

6. Menghapus lagu yang ada di playlist

```
void removeSongFromPlaylist(const char *playlistFilename, int songNumber) {
    FILE *file = fopen(playlistFilename, "r");
    if (file == NULL) {
        printf("Gagal membuka playlist.\n");
        return;
    }

    char tempFilename[] = "temp.txt";
    FILE *tempFile = fopen(tempFilename, "w");
    if (tempFile == NULL) {
        fclose(file);
        printf("Gagal membuat file sementara.\n");
        return;
    }

    char buffer[1000];
    int count = 0;

    while (fgets(buffer, sizeof(buffer), file) != NULL) {
        count++;
        if (count != songNumber) {
            fputs(buffer, tempFile);
        }
    }

    fclose(file);
    fclose(tempFile);

    if (remove(playlistFilename) == 0) {
        if (rename(tempFilename, playlistFilename) == 0) {
            printf("Lagu berhasil dihapus dari playlist.\n");
        } else {
            printf("Gagal mengubah nama file.\n");
        }
    } else {
        printf("Gagal menghapus playlist.\n");
    }
}
```

Kode ini berfungsi untuk menghapus lagu yang dipilih di dalam playlist, cara kerja kode ini adalah yang pertama kode ini akan membaca file playlist yang dipilih, kemudian kode ini akan menulis ulang isi dari filenya ke dalam file baru, dan yang terakhir file baru yang dibuat akan diubah namanya sesuai nama file playlist yang dipilih, kode ini menggunakan 2 access mode yaitu "w" dan "r".

Output:

```
-----Pilih Playlist-----
Daftar Playlist:
1. listlagu
2. playlist1
3. playlist2
0. Hapus Playlist
Pilihan : 0
Masukkan nama file playlist yang ingin dihapus: playlist2.txt
Playlist 'playlist2.txt' berhasil dihapus.
Pilih opsi:
1. Putar Lagu
2. Tambahkan Lagu ke Playlist
3. Hapus Lagu dari Playlist
4. Kembali ke Home
Pilihan: █
```

```
▼ UTS_ALGO
> .vscode
≡ listlagu.txt
≡ playlist1.txt
C tempCodeRunnerFile.c
C utsalgo_Array.c
C utsalgo_Array1.c
C utsalgo_Linkedlist.c
≡ utsalgo_Linkedlist.exe
C utsalgo_Linkedlist1.c
≡ utsalgo_Linkedli... M
≡ utsalgo_Linkedlist1.o
≡ utsalgo.exe
≡ utsalgo.o
≡ zdatabase.txt
≡ zlogo.txt
```

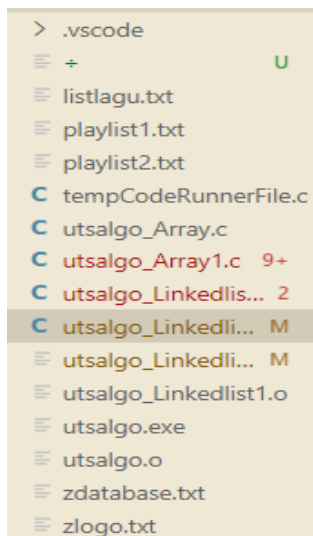
7. Menghapus playlist

```
void deletePlaylistFromFile(const char *filename) {
    if (strcmp(filename, "zdatabase.txt") == 0 || strcmp(filename, "zlogo.txt") == 0) {
        printf("Tidak ada playlist dengan nama '%s' .\n", filename);
    } else {
        if (remove(filename) == 0) {
            printf("Playlist '%s' berhasil dihapus.\n", filename);
        } else {
            printf("Gagal menghapus playlist.\n");
        }
    }
}
```

Kode ini berfungsi untuk menghapus playlist yang ada, cara kerja kode ini adalah menghapus file txt yang dipilih berdasarkan namanya kecuali database.txt dan logo.txt, penghapusan file dilakukan dengan function “remove”.

Output :

```
void deletePlaylistFromFile(const char *filename) {
    if (strcmp(filename, "zdatabase.txt") == 0 || strcmp(filename, "zlogo.txt") == 0) {
        printf("Tidak ada playlist dengan nama '%s' .\n", filename);
    } else {
        if (remove(filename) == 0) {
            printf("Playlist '%s' berhasil dihapus.\n", filename);
        } else {
            printf("Gagal menghapus playlist.\n");
        }
    }
}
```



```
> .vscode
+
U
listlagu.txt
playlist1.txt
playlist2.txt
tempCodeRunnerFile.c
utsalgo_Array.c
utsalgo_Array1.c 9+
utsalgo_Linkedlis... 2
utsalgo_Linkedli... M
utsalgo_Linkedli... M
utsalgo_Linkedlist1.o
utsalgo.exe
utsalgo.o
zdatabase.txt
zlogo.txt
```

8.Next Song

```
switch (playlistOption) {
    case 1: {
        int songNumber;
        printf("Masukkan nomor lagu yang ingin diputar: ");
        scanf("%d", &songNumber);

        FILE *file = fopen(playlistNames[*pilihPlaylist - 1], "r");
        if (file != NULL) {
            char buffer[1000];
            int count = 0;
            struct playlist *current = head;
            while (current != NULL) {
                count++;
                if (count == songNumber) {
                    printf("\nSedang Diputar: %s\n", current->judul);
                    printf("Artis: %s\n", current->penyanyi);
                    printf("Album: %s\n", current->album);
                    playSong(current);
                    sleep(1);
                    break;
                }
                current = current->next;
            }
        }

        while (current != NULL) {
            printf("\n[Menu Player]\n");
            printf("1. Next Song\n");
            printf("2. Previous Song\n");
            printf("3. Stop\n");
            printf("4. Shuffle\n");
            printf("Pilihan: ");
            int kontrol;
            scanf("%d", &kontrol);

            if (kontrol == 1) {
                current = current->next;
                if (current == NULL) {
                    current = head;
                }
                playSong(current);
                sleep(1);
            } else if (kontrol == 2) {
                current = current->prev;
                if (current == NULL) {
                    current = tail;
                }
                playSong(current);
                sleep(1);
            } else if (kontrol == 3) {
                printf("Song playback stopped.\n");
                break;
            }
        }
    }
}
```

Output:

Pilihan : 5

Now Playing: Despacito

Artist: Luis Fonsi ft. Daddy Yankee

Album: Vida

[Menu Player]

1. Next Song

2. Previous Song

3. Stop

4. Shuffle

Pilihan: 1

Now Playing: Closer

Artist: The Chainsmokers ft. Halsey

Album: Collage

[Menu Player]

1. Next Song

2. Previous Song

3. Stop

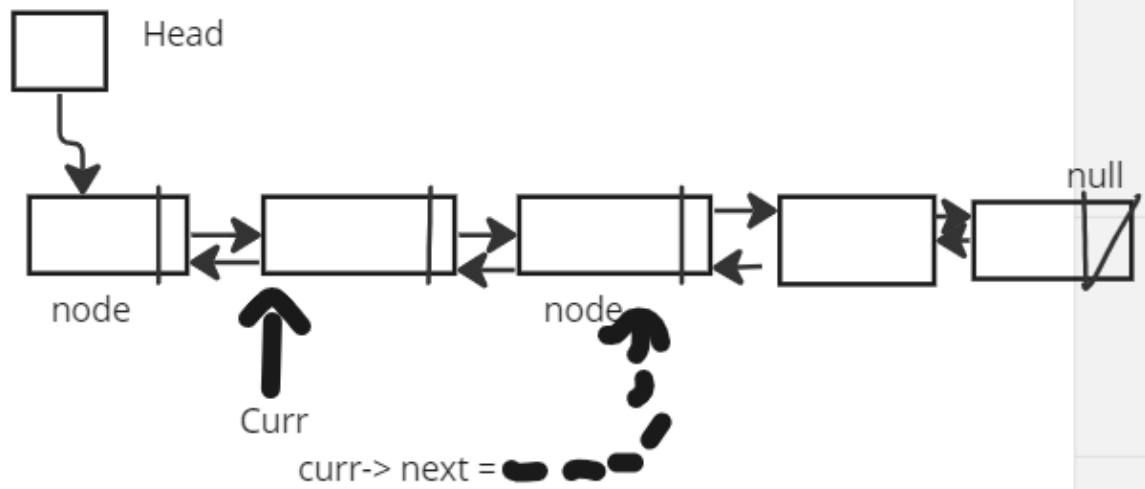
4. Shuffle

Pilihan: 2

listlagu.txt

```
1 Despacito#Luis Fonsi ft. Daddy Yankee#Vida#2019
2 Closer#The Chainsmokers ft. Halsey#Collage#2016
3 Thinking Out Loud#Ed Sheeran#x (Multiply)#2014
4 All of Me#John Legend#Love in the Future#2013
5 Can't Stop the Feeling!#Justin Timberlake#Trolls: Original Motion Picture Soundtrack#2016
6 Havana#Camila Cabello ft. Young Thug#Camila#2017
7 Love Yourself#Justin Bieber#Purpose#2015
8 Shallow#Lady Gaga, Bradley Cooper#A Star Is Born Soundtrack#2018
9 Roar#Katy Perry#PRISM#2013
10 Wrecking Ball#Miley Cyrus#Bangerz#2013
11 Watermelon Sugar#Harry Styles#Fine Line#2019
12 Dynamite#BTS#Dynamite - Single#2020
13 Don't Start Now#Dua Lipa#Future Nostalgia#2019
14 Pumped Up Kicks#Foster the People#Torches#2011
15 We Found Love#Rihanna ft. Calvin Harris#Talk That Talk#2011
16 Memories#Maroon 5#Memories - Single#2019
17 Shape of You#Ed Sheeran#÷ (Divide)#2017
18 Blinding Lights#The Weeknd#After Hours#2020
19 Someone Like You#Adele#21#2011
20 Rolling in the Deep#Adele#21#2010
21 Uptown Funk#Mark Ronson ft. Bruno Mars#Uptown Special#2014
22 Bad Guy#Billie Eilish#WHEN WE ALL FALL ASLEEP, WHERE DO WE GO?#2019
23 Bohemian Rhapsody#Queen#A Night at the Opera#1975
24 Happy#Pharrell Williams#G I R L#2013
25 Closer#The Chainsmokers ft. Halsey#Collage#2016
26 Love Yourself#Justin Bieber#Purpose#2015
27 Can't Stop the Feeling!#Justin Timberlake#Trolls: Original Motion Picture Soundtrack#2016
28 Despacito#Luis Fonsi ft. Daddy Yankee#Vida#2019
29 Shallow#Lady Gaga, Bradley Cooper#A Star Is Born Soundtrack#2018
30 Thinking Out Loud#Ed Sheeran#x (Multiply)#2014
```

Linked List :



9.Prev Song

```
switch (playlistOption) {
    case 1: {
        int songNumber;
        printf("Masukkan nomor lagu yang ingin diputar: ");
        scanf("%d", &songNumber);

        FILE *file = fopen(playlistNames[*pilihPlaylist - 1], "r");
        if (file != NULL) {
            char buffer[1000];
            int count = 0;
            struct playlist *current = head;
            while (current != NULL) {
                count++;
                if (count == songNumber) {
                    printf("\nSedang Diputar: %s\n", current->judul);
                    printf("Artis: %s\n", current->penyanyi);
                    printf("Album: %s\n", current->album);
                    playSong(current);
                    sleep(1);
                    break;
                }
                current = current->next;
            }
        }

        while (current != NULL) {
            printf("\n[Menu Player]\n");
            printf("1. Next Song\n");
            printf("2. Previous Song\n");
            printf("3. Stop\n");
            printf("4. Shuffle\n");
            printf("Pilihan: ");
            int kontrol;
            scanf("%d", &kontrol);

            if (kontrol == 1) {
                current = current->next;
                if (current == NULL) {
                    current = head;
                }
                playSong(current);
                sleep(1);
            } else if (kontrol == 2) {
                current = current->prev;
                if (current == NULL) {
                    current = tail;
                }
                playSong(current);
                sleep(1);
            } else if (kontrol == 3) {
                printf("Song playback stopped.\n");
                break;
            }
        }
    }
}
```

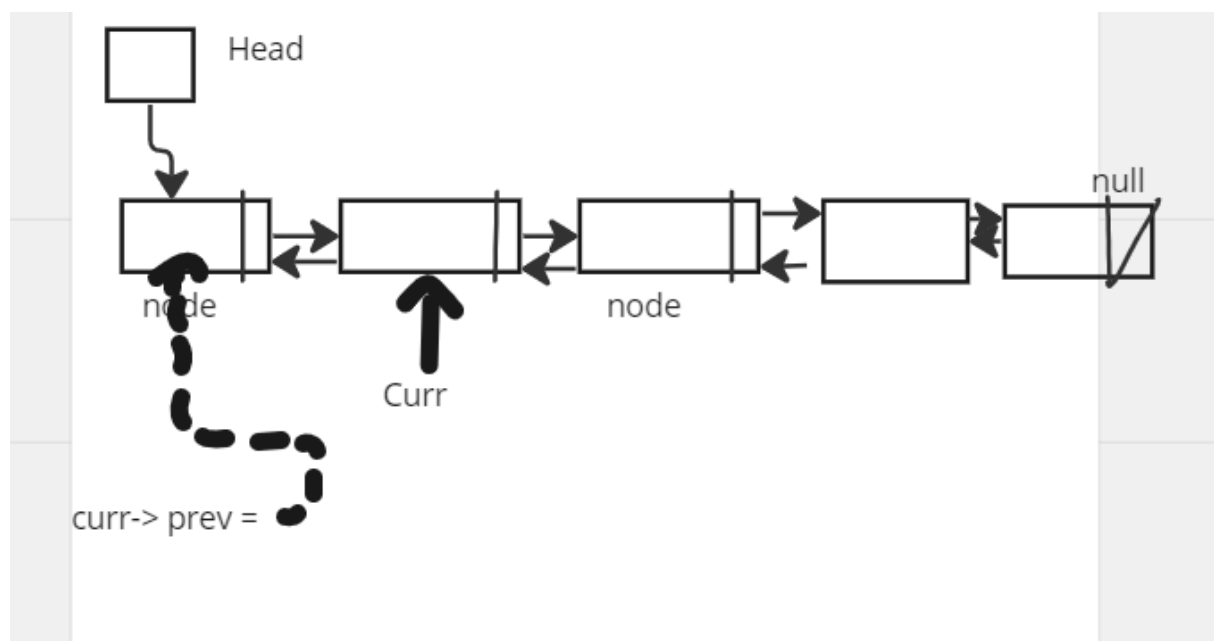
Output:

Now Playing: Closer
Artist: The Chainsmokers ft. Halsey
Album: Collage

[Menu Player]
1. Next Song
2. Previous Song
3. Stop
4. Shuffle
Pilihan: 2

Now Playing: Despacito
Artist: Luis Fonsi ft. Daddy Yankee
Album: Vida

[Menu Player]
1. Next Song
2. Previous Song
3. Stop
4. Shuffle
Pilihan:



10.Cari lagu

```
void searchSong(struct playlist *head, char keyword[50]) {
    printf("Enter keyword to search: ");
    scanf("%s", keyword);

    bool found = false;
    struct playlist *current = head;

    printf("=====\n");
    printf("      Search Results for \"%s\"      \n", keyword);
    printf("=====\n");

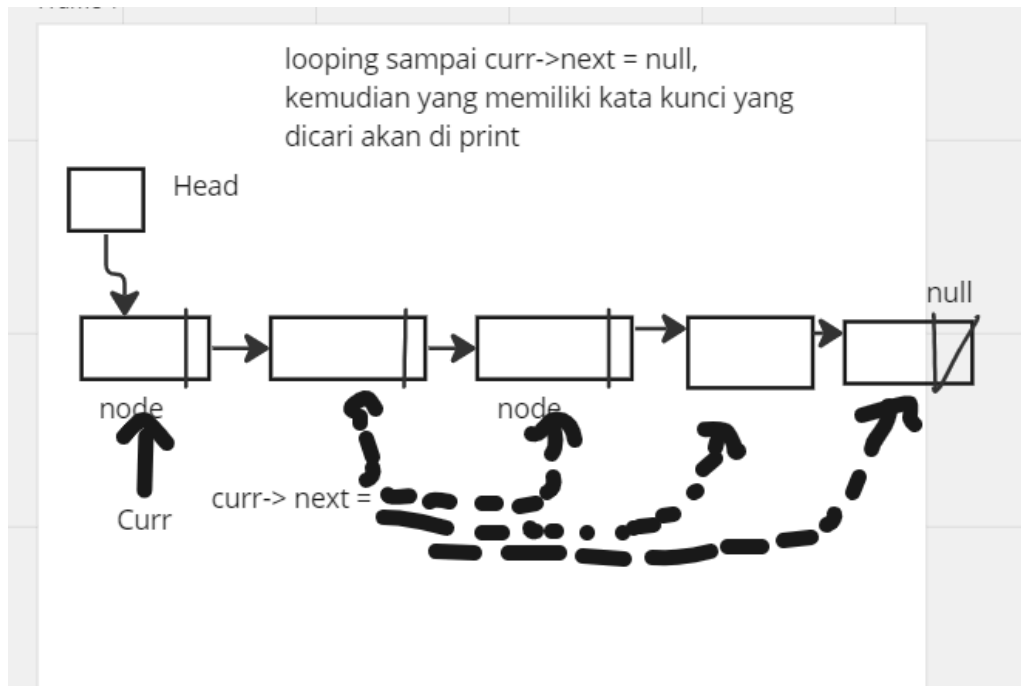
    while (current != NULL) {
        if (strstr(current->judul, keyword) != NULL) {
            printf("Title: %s\n", current->judul);
            printf("Artist: %s\n", current->penyanyi);
            printf("Album: %s\n", current->album);
            printf("Year: %d\n", current->tahun);
            printf("-----\n");
            found = true;
        }
        current = current->next;
    }
    if (!found) {
        printf("No songs found with \"%s\" in the judul.\n", keyword);
    }
}
```

Kode ini berfungsi untuk mencari lagu dengan cara membuka file "listlagu.txt", kemudian akan melakukan pengecekan sampai menemukan lagu yang memiliki keyword tersebut

Output :

```
2. Choose Playlist
3. Display All Song
4. Create playlist
5. Play Song
6. Logout
Pilihan : 1
Enter keyword to search: Love
=====
                Search Results for "Love"
=====
Title: Love Yourself
Artist: Justin Bieber
Album: Purpose
Year: 2015
-----
Title: We Found Love
Artist: Rihanna ft. Calvin Harris
Album: Talk That Talk
Year: 2011
-----
Title: Love Yourself
Artist: Justin Bieber
Album: Purpose
Year: 2015
-----
```

Linked List :



11. Pilih playlist

```
void playlist(int pilihhome, int *pilihPlaylist, struct playlist *head, struct playlist *tail) {
    char playlistNames[50][50];
    if (pilihhome == 2) {
        system("cls");
        printf("-----Pilih Playlist-----\n");
        displayExistingPlaylists();
        printf("0. Hapus Playlist \n");
        printf("Pilihan : ");
        scanf("%d", pilihPlaylist);

        if (*pilihPlaylist == 0) {
            char filename[50];
            printf("Masukkan nama file playlist yang ingin dihapus: ");
            scanf("%49s", filename);
            deletePlaylistFromFile(filename);
        } else {
            DIR *dir;
            struct dirent *direct;
            int count = 0;
            if ((dir = opendir(".")) != NULL) {
                while ((direct = readdir(dir)) != NULL) {
                    if (strstr(direct->d_name, ".txt") != NULL) {
                        strcpy(playlistNames[count], direct->d_name);
                        count++;
                    }
                }
                closedir(dir);
            } else {
                perror("Direktori tidak tersedia");
                return;
            }

            if (*pilihPlaylist > 0 && *pilihPlaylist <= count) {
                displayPlaylist(playlistNames[*pilihPlaylist - 1]);
            } else {
                printf("Pilihan tidak valid.\n");
                return;
            }
        }
    }

    int playlistOption;
    printf("Pilih opsi:\n");
    printf("1. Putar Lagu\n");
    printf("2. Tambahkan Lagu ke Playlist\n");
    printf("3. Hapus Lagu dari Playlist\n");
    printf("4. Kembali ke Home\n");
}
```

```

void displayExistingPlaylists() {
    char playlists[100][50];
    int count = 0;

    system("dir /b > files.txt");
    FILE *file = fopen("files.txt", "r");
    if (file != NULL) {
        printf("Daftar Playlist:\n");

        char line[256];
        while (fgets(line, sizeof(line), file)) {
            strtok(line, "\n");

            if (endsWithTxt(line) && strcmp(line, "zdatabase.txt") != 0 && strcmp(line, "zlogo.txt") != 0 && strcmp(line, "files.txt") != 0) {
                strncpy(playlists[count], line, strlen(line) - 4);
                playlists[count][strlen(line) - 4] = '\0';
                count++;
            }
        }
        fclose(file);
        remove("files.txt");

        for (int i = 0; i < count; i++) {
            printf("%d. %s\n", i + 1, playlists[i]);
        }
    } else {
        perror("Direktori tidak tersedia");
    }
}

```

Kode ini memiliki cara kerja yaitu membuka Directory pada komputer untuk menampilkan seluruh playlist yang ada menggunakan fungsi “Open DIR”, “Read Dir”.

Output:

```

-----Pilih Playlist-----
Daftar Playlist:
1. listlagu
2. playlist1
3. playlist2
0. Hapus Playlist
Pilihan : 1

```

No	Judul	Penyanyi	Album	Tahun
1	Despacito	Luis Fonsi ft. Daddy Yankee	Vida	2019
2	Closer	The Chainsmokers ft. Halsey	Collage	2016
3	Thinking Out Loud	Ed Sheeran	x (Multiply)	2014
4	All of Me	John Legend	Love in the Future	2013
5	Can't Stop the Feeling!	Justin Timberlake	Trolls: Original Motion Picture Soundtrack	2016
6	Havana	Camila Cabello ft. Young Thug	Camila	2017
7	Love Yourself	Justin Bieber	Purpose	2015
8	Shallow	Lady Gaga, Bradley Cooper	A Star Is Born Soundtrack	2018
9	Roar	Katy Perry	PRISM	2013
10	Wrecking Ball	Miley Cyrus	Bangerz	2013
11	Watermelon Sugar	Harry Styles	Fine Line	2019
12	Dynamite	BTS	Dynamite - Single	2020
13	Don't Start Now	Dua Lipa	Future Nostalgia	2019
14	Pumped Up Kicks	Foster the People	Torches	2011
15	We Found Love	Rihanna ft. Calvin Harris	Talk That Talk	2011
16	Memories	Maroon 5	Memories - Single	2019

ARRAY

1. Menampilkan playlist yang tersedia

```
1 void displayPlaylist(const char *playlistName) {
2     FILE *display = fopen(playlistName, "r");
3     if (display != NULL) {
4         char buffer[100];
5         printf("=====\n");
6         printf("| %-30s | %-30s | %-35s | %-5s |\n", "Judul", "Penyanyi", "Album", "Tahun");
7         printf("=====\n");
8
9
10        char judul[50][100];
11        char penyanyi[50][100];
12        char album[50][100];
13        int tahun[50];
14        int index = 0;
15
16        while (fgets(buffer, sizeof(buffer), display) != NULL) {
17            if (strstr(buffer, "zdatabase.txt") != NULL || strstr(buffer, "zlogo.txt") != NULL) {
18                continue;
19            }
20            // Use sscanf to parse the buffer
21            sscanf(buffer, "%[^#]#[^#]#[^#]#(%d)", judul[index], penyanyi[index], album[index], &tahun[index]);
22            printf("| %-30s | %-30s | %-35s | %-5s |\n", judul[index], penyanyi[index], album[index], tahun[index]);
23            index++;
24        }
25
26        printf("=====\n");
27        fclose(display);
28    } else {
29        printf("Gagal membuka playlist.\n");
30    }
31 }
```

Kode di atas berfungsi untuk menampilkan isi dari sebuah playlist yang sudah disimpan dalam file txt. Fungsi ini menerima parameter 'playlistName' yang merupakan nama file playlist yang ingin ditampilkan.

2. Menampilkan playlist yang ada

```
1 void displayExistingPlaylistsArray(char *existingPlaylists[], int *playlistCount) {
2     DIR *dir;
3     struct dirent *direct;
4     if ((dir = opendir(".")) != NULL) {
5         *playlistCount = 0;
6         while ((direct = readdir(dir)) != NULL) {
7             if (strstr(direct->d_name, ".txt") != NULL && strcmp(direct->d_name, "database.txt") != 0 && strcmp(direct->d_name, "logo.txt") != 0) {
8                 // Store the playlist name in the array
9                 existingPlaylists[*playlistCount] = direct->d_name;
10                (*playlistCount)++;
11            }
12        }
13        closedir(dir);
14    } else {
15        perror("Direktori tidak tersedia");
16    }
17 }
```

Kode di atas berfungsi untuk menampilkan daftar playlist yang ada dalam direktori saat ini ke dalam array 'existingPlaylists'. Fungsi ini juga menghitung jumlah playlist yang ada dan menyimpannya dalam variabel 'playlistCount'.

Fungsi ini menggunakan library '<dirent.h>' untuk mengakses direktori. Setiap file yang memiliki =====

3.Mengecek Akun

```
1 bool cekAkunArray(struct akun *accounts[], int numAccounts, const char *username, const char *password) {
2     for (int i = 0; i < numAccounts; i++) {
3         if (strcmp(accounts[i]->username, username) == 0 && strcmp(accounts[i]->password, password) == 0) {
4             return true;
5         }
6     }
7     return false;
8 }
```

4.Membuat Playlist

```
void createCustomPlaylist(char playlistNames[][50], int numPlaylists) {
    for (int i = 0; i < numPlaylists; i++) {
        FILE *file = fopen(strcat(playlistNames[i], ".txt"), "a");
        if (file == NULL) {
            printf("Gagal membuat playlist '%s'.\n", playlistNames[i]);
        } else {
            printf("Playlist '%s' berhasil dibuat.\n", playlistNames[i]);
            fclose(file);
        }
    }
}
```

```
1 void readDatabase(struct akun **head) {
2     FILE *file = fopen("database.txt", "r");
3     if (file == NULL) {
4         printf("Error membuka file database.\n");
5         return;
6     }
7
8     int numAccounts = 0;
9     while (!feof(file)) {
10         struct akun *node = (struct akun*)malloc(sizeof(struct akun));
11         if (fscanf(file, "%s %s\n", node->username, node->password) != 2) {
12             free(node);
13             break;
14         }
15
16         node->next = *head;
17         *head = node;
18         numAccounts++;
19     }
20     fclose(file);
21
22     struct akun *current = *head;
23     struct akun **array = (struct akun **)malloc(numAccounts * sizeof(struct akun *));
24     for (int i = 0; i < numAccounts; i++) {
25         array[i] = current;
26         current = current->next;
27     }
28
29     while (*head != NULL) {
30         struct akun *temp = *head;
31         *head = (*head)->next;
32         free(temp);
33     }
34
35     free(array);
36 }
37 }
```

```

void createPlaylist() {
    int songLength = 100;
    char titleLength = 50;
    char artistLength = 50;
    char playlistNames[50][50];
    int numPlaylists;
    int numSongs;
    FILE *file;

    struct playlist playlist[songLength];

    printf("1. Buat playlist baru\n");
    printf("2. Tambahkan lagu ke playlist yang ada\n");
    printf("Pilihan: ");
    int choice;
    scanf("%d", &choice);

    if (choice == 1) {
        createCustomPlaylist(playlistNames, numPlaylists);
    }

    printf("Masukkan jumlah lagu yang ingin ditambahkan ke playlist: ");
    scanf("%d", &numSongs);
    getchar();

    file = fopen("playlist.txt", "a");

    file = fopen("playlist.txt", "a");

    if (file == NULL) {
        printf("Gagal membuka file.\n");
        return;
    }

    for (int i = 0; i < numSongs; i++) {
        printf("Masukkan judul lagu ke-%d: ", i + 1);
        fgets(playlist[i].judul, titleLength, stdin);
        strtok(playlist[i].judul, "\n");

        printf("Masukkan nama penyanyi untuk lagu ke-%d: ", i + 1);
        fgets(playlist[i].penyanyi, artistLength, stdin);
        strtok(playlist[i].penyanyi, "\n");

        printf("Masukkan tahun rilis untuk lagu ke-%d: ", i + 1);
        scanf("%d", &playlist[i].tahun);
        getchar();

        fprintf(file, "%s#%s#(%d)\n", playlist[i].judul, playlist[i].penyanyi, playlist[i].tahun);
    }

    printf("Playlist berhasil ditambahkan ke file.\n");

    fclose(file);
}

```


6. Membuat Akun

```
void addAcc(struct akun **head, const char *username, const char *password) {
    struct akun *node = (struct akun*)malloc(sizeof(struct akun));
    strcpy(node->username, username);
    strcpy(node->password, password);
    node->next = *head;
    *head = node;

    FILE *file = fopen("database.txt", "a");
    if (file == NULL) {
        printf("Error membuka file database.\n");
        return;
    }
    fprintf(file, "%s %s\n", username, password);
    fclose(file);
}
```

7. Menampilkan logo

```
fclose(file);
char logo[] = "logo.txt";
char baris[1000];

file = fopen(logo, "r");
createData(file, baris);
printf("\n");
fclose(file);

login(kondisi, username, password);
```

8. Menghapus playlist menggunakan array

```
void deletePlaylistFromFile(const char *filename) {
    if (strcmp(filename, "database.txt") == 0 || strcmp(filename, "logo.txt") == 0) {
        printf("Tidak ada playlist dengan nama '%s' .\n", filename);
    } else {
        if (remove(filename) == 0) {
            printf("Playlist '%s' berhasil dihapus.\n", filename);
        } else {
            printf("Gagal menghapus playlist.\n");
        }
    }
}
```

Tugas

Nama Anggota	Pembagian Tugas
Jonathan Prasetyo	Create playlist serta create custom playlist, login & sign up, add & remove song, add & delete playlist, menampilkan logo ascii spotify,
Hexsel Archieles Virgio Manik	Play Song beserta void playSong, Play next play prev, Search, membuat gambar linked list dalam laporan
Jonathan Sutandar	Membuat studi kasus, kajian teori, membuat void displayExistingPlaylistsArray menggunakan array, membuat void display playlist menggunakan array.
Edric Hugo	Membuat display menu, dan juga table lagu, membantu dalam pembuatan array.

Evaluasi

Nama	Evaluasi
Jonathan Prasetyo	<p>Menurut pendapat saya pribadi saya dan kelompok saya sudah mengerahkan <i>effort</i> dan komitmen yang besar bagi proyek kali ini tetapi dari segi pengetahuan dan pengalaman kami yang kurang membuat terjadi banyak tabrakan antar anggota kelompok yang membuat proses pengerjaan <i>project</i> kali ini agak sedikit terhambat karenanya, dan juga ada kesalahan dalam menetapkan prioritas tugas, dimana masalah ini membuat kami tidak bisa memfokuskan diri karena adanya UTS lain yang datang bersamaan dengan adanya <i>project</i> ini. Dengan semua masalah yang kami hadapi terdapat beberapa fitur yang tidak dapat kami realisasikan yaitu fitur repeat song yang berfungsi untuk melakukan looping kembali saat lagu telah selesai di putar, juga pembagian akun yang akan memisahkan playlist akun 1 dengan akun yang satunya lagi, dan ada juga fitur <i>shuffle</i> playlist yang berguna untuk memutar lagu dalam playlist secara acak, dengan semua hal yang harusnya dapat kami kembangkan dan realisasikan ini, kami harap dapat menjadi sarana untuk kami dapat lebih mengembangkan diri ke depannya.</p>
Hexsel Archieles Virgio Manik	<p>Menurut saya kami sekelompok harus bekerja secara lebih efisien dikarenakan progres yang kami alami cukup lambat dan terhambat akibat kurangnya komunikasi dan kurangnya efisiensi dalam menjalani tugas masing masing. Namun kerja sama pada kelompok kami terus meningkat seiring berjalannya waktu dari hari pertama pengerjaan tugas, sampai hari terakhir pengerjaan tugas. Kami juga memiliki beberapa fitur yang tidak dapat direalisasikan seperti "Repeat Song" yaitu melakukan looping lgi terhadap curr yang sedang di play/ lagu yang sedang di play, serta "shuffle play" yang memiliki cara kerja mengacak urutan lagu yang di play "mengcopy terlebih dahulu isi file playlist dengan fopen tipe access "r", dan lakukan strcpy ke dalam array, lalu menggunakan srand untuk mengacak lagunya, kemudian implementasikan double linked list ke lagu yang sudah diacak".</p>
Jonathan Sutandar	<p>Menurut saya, proyek ini memerlukan time management yang sangat baik, agar semua pekerjaan</p>

	<p>dapat diselesaikan dengan baik. kami mengalami beberapa kesulitan. tetapi kesulitan tersebut dapat terselesaikan dengan baik. progress yang cukup signifikan terjadi setiap harinya, walaupun diimbangi dengan UTS teori tetapi kami sekelompok masih bisa memberikan yang terbaik untuk tugas ini. walaupun dapat berjalan dengan baik tetapi ada fitur yang belum bisa direalisasikan oleh kelompok kami seperti fitur repeat song.</p>
Edric Hugo	<p>Dalam mengevaluasi proyek ini, saya merasa bahwa saya mengalami kesulitan dalam menyelesaikan UTS ini. Salah satunya adalah sulitnya membagi waktu untuk mengerjakan karena adanya banyak UTS lainnya dalam waktu yang bersamaan dan harus diselesaikan dalam waktu yang singkat. Selain itu, beberapa fitur juga tidak dapat terwujud karena kurangnya komunikasi dan waktu sehingga membuat progres kurang optimal. Saya merasa bahwa saya masih kurang dalam segi pengalaman dan pengetahuan, sehingga saya menyadari bahwa saya perlu terus mengembangkan diri agar dapat menjadi lebih baik lagi. Saya berharap agar kami semua dapat bekerja lebih maksimal dan memiliki time management yang lebih baik untuk fokus pada pengerjaan UTS sehingga dapat mengerjakan dengan lancar dan dapat terselesaikan dengan baik.</p>