DATASTRUCTURES PROJECT ABSTRACT

TEAM - 2

K. LAKSHMI YASASVI - 23WH1A05F5 – CSE C

A. LAHARI - 23WH1A05F6 – CSE C

G. HANSINI - 23WH1A05J1 – CSE C

ADIFA SANIYA - 23WH1A05G3 – CSE C

P. AKSHAYA - 23WH1A05E9 - CSE C

Project Title: MUSIC PLAYLIST MANAGER

Project Description: This project is a simple Music Playlist Manager implemented in C using a **circular doubly linked list** to manage songs. Users can add, remove, and display songs in their playlist, as well as mark songs as favorites. The application allows for playback controls, including playing the next or previous song, and features a shuffle play option to randomly select songs. It offers a user-friendly menu for interacting with the playlist, making it an engaging way to manage music collections.

Practical Implementation of Our Code in Leading Music Streaming Platforms

1.Spotify

Spotify is a leading music streaming service offering millions of songs. It allows users to create personalized playlists and discover new music through recommendations.
Application of Our Project: Our playlist manager can serve as the backend logic for Spotify's functionality, enabling users to add or remove songs and mark favorites with ease. **The efficiency of our circular doubly linked list ensures smooth navigation and quick updates to playlists, which is critical for user satisfaction.**

2. Apple Music

Apple Music features an extensive catalog of songs and exclusive content. It offers seamless integration with Apple devices, allowing users to manage playlists and listen offline.
Application of Our Project: The playlist management capabilities in our project can enhance Apple Music by providing users with an intuitive way to curate and personalize their music collections. **Our logic allows for efficient organization, enabling users to find and manage their favorite tracks effortlessly.**

3. YouTube Music

YouTube Music combines music videos and audio tracks, enabling users to create playlists based on their favorite artists and genres.
Application of Our Project: The features of adding and removing songs from playlists in our manager can be integrated into YouTube Music, allowing users to enjoy dynamic playback options like shuffling. **The implementation of shuffling and quick additions ensures a lively user experience, keeping users engaged.**

4. Pandora

Pandora allows users to create personalized radio stations based on their song preferences, using algorithms to generate playlists**.**
Application of Our Project: Our project's playlist management can help manage user-created playlists, enabling users to add, remove, and favorite songs, thereby enhancing their personalized music experience**. Our logic supports dynamic playlist adjustments, allowing for a more tailored listening experience that adapts to user preferences.**

5. SoundCloud

SoundCloud provides a platform for independent artists to share their music and allows users to create playlists of their favorite tracks.
Application of Our Project: The functionality in our Music Playlist Manager can be directly applied to SoundCloud, enabling users to curate their listening experience effortlessly. **The efficient management of user playlists ensures artists' work is highlighted and easily accessible to fans, fostering a stronger connection.**

Our Music Playlist Manager provides a solid foundation for playlist management that can be integrated into various music streaming applications. By implementing essential features for adding, removing, and organizing songs, our project enhances user experience and enables efficient curation of music collections. The principles of circular doubly linked lists and simple playlist management can be

expanded upon in larger systems, demonstrating the relevance of our project in the evolving landscape of digital music streaming.

Source Code:

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <time.h> // For shuffle play

// Define the structure for a song
typedef struct Song {
    char title[50];
    int isFavorite;  // 0 for normal, 1 for favorite
    struct Song* next;
    struct Song* prev;
} Song;

// Function to create a new song
Song* createSong(char* title) {
    Song* newSong = (Song*)malloc(sizeof(Song));
    strcpy(newSong->title, title);
    newSong->isFavorite = 0;  // Default is not favorite
    newSong->next = newSong->prev = newSong; // Circular linked list
    return newSong;
```

```
}

// Function to add a song to the playlist
Song* addSong(Song* head, char* title) {
    Song* newSong = createSong(title);
    if (head == NULL) {
        return newSong;  // First song in the playlist becomes the head
    } else {
        Song* last = head->prev;
        newSong->next = head;
        newSong->prev = last;
        last->next = newSong;
        head->prev = newSong;
        return head;
    }
}

// Function to display the playlist
void displayPlaylist(Song* head) {
    if (head == NULL) {
        printf("The playlist is empty.\n");
        return;
    }
```

```
    Song* temp = head;
    do {
        printf("Song: %s\n", temp->title);
        temp = temp->next;
    } while (temp != head);
}

// Function to remove a song from the playlist
Song* removeSong(Song* head, char* title) {
    if (head == NULL) {
        printf("Playlist is empty.\n");
        return NULL;
    }

    Song* current = head;
    Song* previous = NULL;

    // Traverse the circular linked list to find the song
    do {
        if (strcmp(current->title, title) == 0) {
            if (current->next == current) {  // Only one song in the playlist
                free(current);
                return NULL;
            } else {
```

```
            // If removing head song, update head
            if (current == head) {
                head = current->next;
            }

            previous = current->prev;
            previous->next = current->next;
            current->next->prev = previous;
            free(current);
            return head;
        }
    }
    current = current->next;
} while (current != head);

printf("Song not found in playlist.\n");
return head;
}

// Function to play the next song
void playNext(Song** currentSong) {
    if (*currentSong == NULL) {
        printf("Playlist is empty.\n");
    } else {
```

```
        *currentSong = (*currentSong)->next;
        printf("Now playing: %s\n", (*currentSong)->title);
    }
}

// Function to play the previous song
void playPrevious(Song** currentSong) {
    if (*currentSong == NULL) {
        printf("Playlist is empty.\n");
    } else {
        *currentSong = (*currentSong)->prev;
        printf("Now playing: %s\n", (*currentSong)->title);
    }
}

// Function to shuffle play a random song
void shufflePlay(Song** currentSong, Song* head) {
    if (head == NULL) {
        printf("Playlist is empty.\n");
        return;
    }

    srand(time(0)); // Seed for random number generation
    int totalSongs = 0;
```

```
    Song* temp = head;

    // Count total number of songs
    do {
        totalSongs++;
        temp = temp->next;
    } while (temp != head);

    // Select a random song
    int randomIndex = rand() % totalSongs;
    for (int i = 0; i < randomIndex; i++) {
        *currentSong = (*currentSong)->next;
    }

    printf("Shuffle playing: %s\n", (*currentSong)->title);
}

// Function to mark a song as a favorite
void addToFavorites(Song* song) {
    song->isFavorite = 1;
    printf("Added %s to favorites.\n", song->title);
}

// Function to display only favorite songs
```

```
void displayFavorites(Song* head) {
    if (head == NULL) {
        printf("No songs in the playlist.\n");
        return;
    }

    Song* temp = head;
    int found = 0;
    do {
        if (temp->isFavorite) {
            printf("Favorite Song: %s\n", temp->title);
            found = 1;
        }
        temp = temp->next;
    } while (temp != head);

    if (!found) {
        printf("No favorite songs in the playlist.\n");
    }
}

// Main function
int main() {
    Song* playlist = NULL;
```

```
Song* currentSong = NULL;

int choice;
char title[50];

while (1) {
    printf("\n--- Music Playlist Manager---\n");
    printf("1. Add Song\n");
    printf("2. Remove Song\n");
    printf("3. Display Playlist\n");
    printf("4. Play Next Song\n");
    printf("5. Play Previous Song\n");
    printf("6. Shuffle Play\n");
    printf("7. Add to Favorites\n");
    printf("8. Display Favorite Songs\n");
    printf("9. Exit\n");
    printf("Enter your choice: ");
    scanf("%d", &choice);

    switch (choice) {
        case 1: {
            printf("Enter song title: ");
            scanf("%s", title);
            playlist = addSong(playlist, title);
```

```
            if (currentSong == NULL) {
                currentSong = playlist;  // Set the first song as current song
            }
            break;
        }
        case 2: {
            printf("Enter song title to remove: ");
            scanf("%s", title);
            playlist = removeSong(playlist, title);
            if (playlist == NULL) {
                currentSong = NULL;
            }
            break;
        }
        case 3: {
            displayPlaylist(playlist);
            break;
        }
        case 4: {
            playNext(&currentSong);
            break;
        }
        case 5: {
            playPrevious(&currentSong);
```

```
        break;
    }
    case 6: {
        shufflePlay(&currentSong, playlist);
        break;
    }
    case 7: {
        printf("Enter song title to add to favorites: ");
        scanf("%s", title);
        Song* temp = playlist;
        int found = 0;
        do {
            if (strcmp(temp->title, title) == 0) {
                addToFavorites(temp);
                found = 1;
                break;
            }
            temp = temp->next;
        } while (temp != playlist);
        if (!found) {
            printf("Song not found in playlist.\n");
        }
        break;
    }
```

```
            case 8: {
                displayFavorites(playlist);
                break;
            }
            case 9:
                printf("Exiting...\n");
                return 0;
            default:
                printf("Invalid choice, please try again.\n");
        }
    }

    return 0;
}
```

Output:

```
--- Music Playlist Manager ---
1. Add Song
2. Remove Song
3. Display Playlist
4. Play Next Song
5. Play Previous Song
6. Shuffle Play
7. Add to Favorites
8. Display Favorite Songs
9. Exit
Enter your choice: 1
Enter song title: sadanannu
```

```
--- Music Playlist Manager ---
1. Add Song
2. Remove Song
3. Display Playlist
4. Play Next Song
5. Play Previous Song
6. Shuffle Play
7. Add to Favorites
8. Display Favorite Songs
9. Exit
Enter your choice: 1
Enter song title: mellaga
```

```
--- Music Playlist Manager ---
1. Add Song
2. Remove Song
3. Display Playlist
4. Play Next Song
5. Play Previous Song
6. Shuffle Play
7. Add to Favorites
8. Display Favorite Songs
9. Exit
Enter your choice: 1
Enter song title: birdsofafeather
```

```
--- Music Playlist Manager ---
1. Add Song
2. Remove Song
3. Display Playlist
4. Play Next Song
5. Play Previous Song
6. Shuffle Play
7. Add to Favorites
8. Display Favorite Songs
9. Exit
Enter your choice: 3
Song: sadanannu
Song: mellaga
Song: birdsofafeather
```

```
--- Music Playlist Manager ---
1. Add Song
2. Remove Song
3. Display Playlist
4. Play Next Song
5. Play Previous Song
6. Shuffle Play
7. Add to Favorites
8. Display Favorite Songs
9. Exit
Enter your choice: 6
Shuffle playing: birdsofafeather
```

```
--- Music Playlist Manager ---
1. Add Song
2. Remove Song
3. Display Playlist
4. Play Next Song
5. Play Previous Song
6. Shuffle Play
7. Add to Favorites
8. Display Favorite Songs
9. Exit
Enter your choice: 7
Enter song title to add to favorites: mellaga
Added mellaga to favorites.
```

```
--- Music Playlist Manager ---
1. Add Song
2. Remove Song
3. Display Playlist
4. Play Next Song
5. Play Previous Song
6. Shuffle Play
7. Add to Favorites
8. Display Favorite Songs
9. Exit
Enter your choice: 8
Favorite Song: mellaga
```

```
--- Music Playlist Manager ---
1. Add Song
2. Remove Song
3. Display Playlist
4. Play Next Song
5. Play Previous Song
6. Shuffle Play
7. Add to Favorites
8. Display Favorite Songs
9. Exit
Enter your choice: 5
Now playing: mellaga
```

```
--- Music Playlist Manager ---
1. Add Song
2. Remove Song
3. Display Playlist
4. Play Next Song
5. Play Previous Song
6. Shuffle Play
7. Add to Favorites
8. Display Favorite Songs
9. Exit
Enter your choice: 9
Exiting...


=== Code Execution Successful ===
```