# CA 3: Experiential Learning

## Group Members:

| Sr. No. | PRN | Name of Student | Mail id |
|---|---|---|---|
| 1 | 22070122085 | Jahnvi Nair | jahnvi.nair.btech2022@sitpune.edu.in |
| 2 | 22070122112 | Manya Khullar | manya.khullar.btech2022@sitpune.edu.in |
| 3 | 22070122167 | Ruhani Rai Dhamija | ruhani.dhamija.btech2022@sitpune.edu.in |

## Problem Statement:

Creating a C++ program on the topic of a music player, which includes the concepts of polymorphism and inheritance.

## Explanation:

The program represents one for a music player, which has a variety of functions it can perform. It is a basic music management system with playlist management features. It allows you to create and manage playlists, play and navigate between songs, toggle shuffle and repeat modes, and even delete said playlists.

Song navigation itself is a very simplistic idea: playing a song, and moving between multiple songs. From a broad perspective, when downloading music, it is difficult to keep track of all the tracks. Playlist management helps here with being able to group different songs for easier classification. These groups can be selected between, and deleted according to preference. If you are not in control of the device at hand and want to listen to a randomized selection of songs, toggling shuffle and repeat options help to play an unordered set of songs, also repeating previously played songs.

This program defines three classes: ‘Media’, ‘Song’, and ‘Playlist’. ‘Media’ is the base class, including all media items, which in this regard means the saved songs. ‘Song’ is a derived class from ‘Media’, representing a song with the artist attribute as well. The ‘Playlist’ class is for the main playlist management and functionality, including adding, playing, shuffling, and repeating tracks, as well as for creating, deleting, and selecting playlists.

On running the code, the user can see a main menu, which is repeated as a loop until an exit option is selected. The following functions are possible in the said menu:

1. Playing the current track (if a playlist is selected)
2. Moving to the next track (if a playlist is selected)
3. Moving to the previous track (if a playlist is selected)
4. Toggling shuffle mode (if a playlist is selected)
5. Toggling repeat mode (if a playlist is selected)

6. Creating a new playlist
7. Managing a new playlist
8. Deleting the currently selected playlist (if any)
9. Exiting the program

On selecting the managing playlist option (option 7 above), the option to select a playlist out of a given number of inserted playlists is provided. This option is possible only if at least one playlist has been created by the user. In fact, all other options only provide results if at least one playlist has been created by the user.

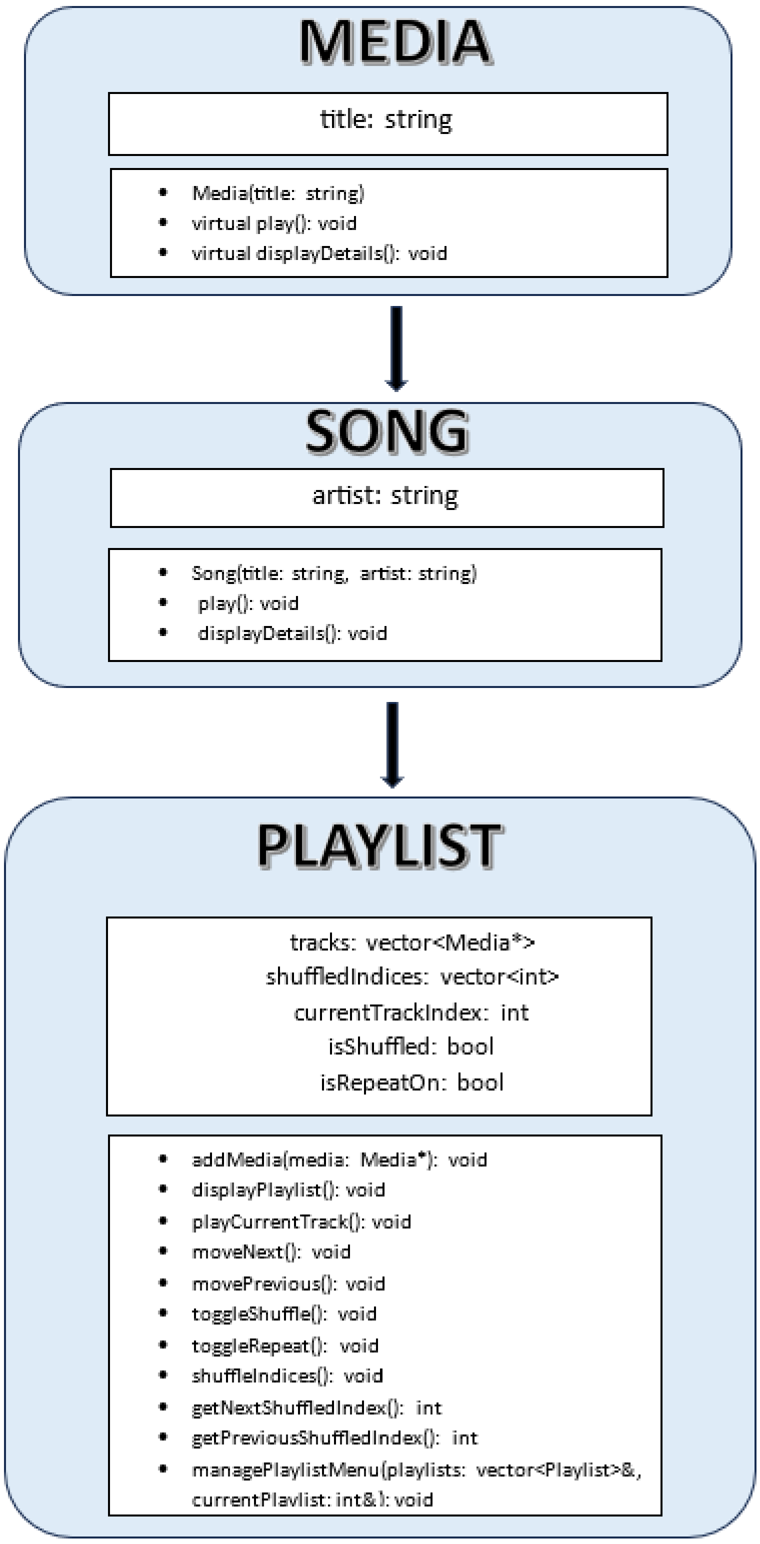
Error handling has also been implemented in the code, to make the user's experience more easy-going. If a non-numeric value is inserted by the user in the select option, an "Invalid Input" message is displayed, after which the code loops again to go back to the main menu. Other similar safety nets have been provided to avoid an infinite loop, which can prove hectic for a user. The program also uses simplistic language for an easier understanding and a user-friendly experience.

Inheritance of classes includes a new class inheriting properties and behaviours from a pre-existing class (the base class). In the instance of this program, the base class is 'Media', and 'Song' is the derived class, which implies the latter inherits properties from the former. The 'Media' class has a 'title' attribute, and functions 'play()' and 'displayDetails()', which are inherited by the 'Song' class. The 'Song' class further adds an 'artist' attribute, thereby making it a sub-class of 'Media' with additional attributes. This class further modifies the inherited functions, to make them more specialized towards songs.

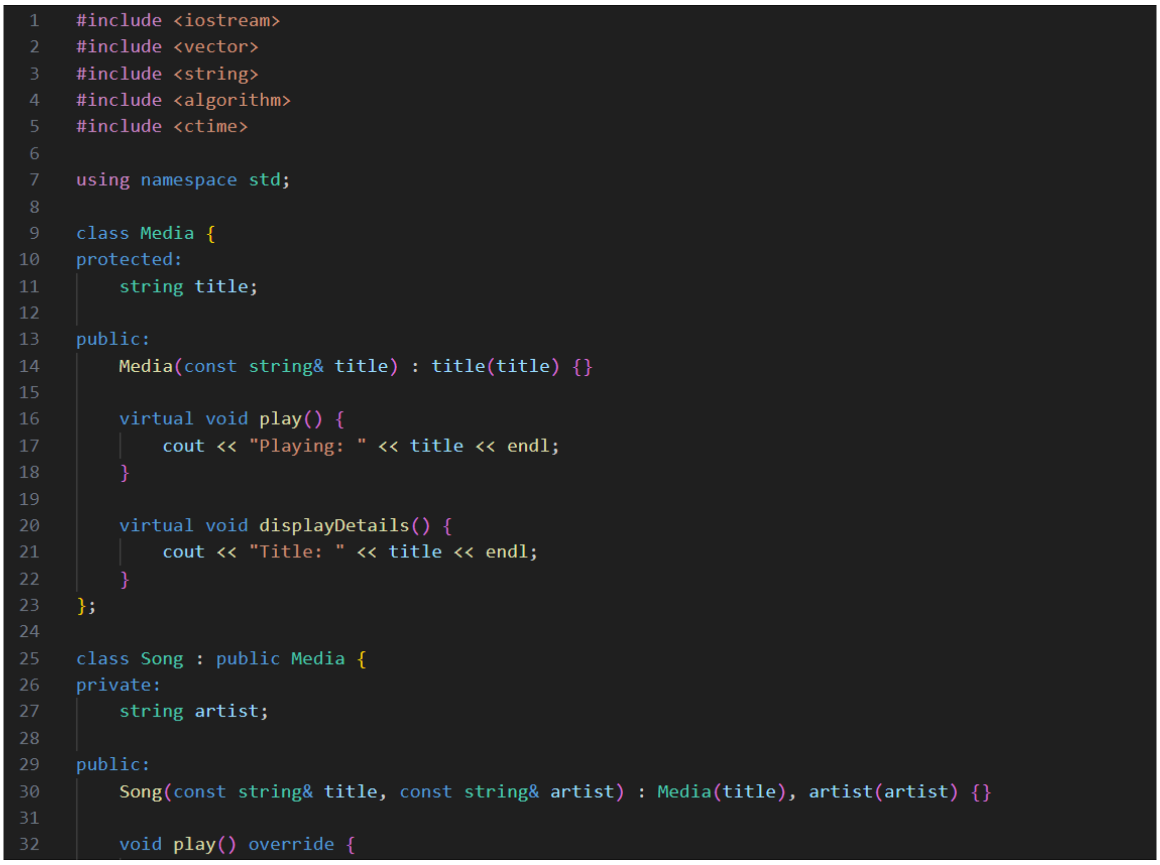
Polymorphism is another significant concept, allowing objects of different classes to be treated as objects of a common base class. In this instance, the class 'Media' defines two virtual functions: 'play()' and 'displayDetails()', virtual implying they are intended to be overridden by derived classes. As explained earlier, the 'Song' class overrides the functions with its own implementations.

In the 'Playlist' class, pointers to the 'Media' class are used to represent tracks in the playlist. For this, the 'Playlist' class invokes the above virtual functions on the media items in the playlist. This allows the playlist to hold a mix of media items, including both basic media and songs since all of them inherit from the common base class 'Media'. The polymorphism feature ensures that the appropriate versions of said objects are used. That is, either the main version defined in the base class or the modified version in the derived class.

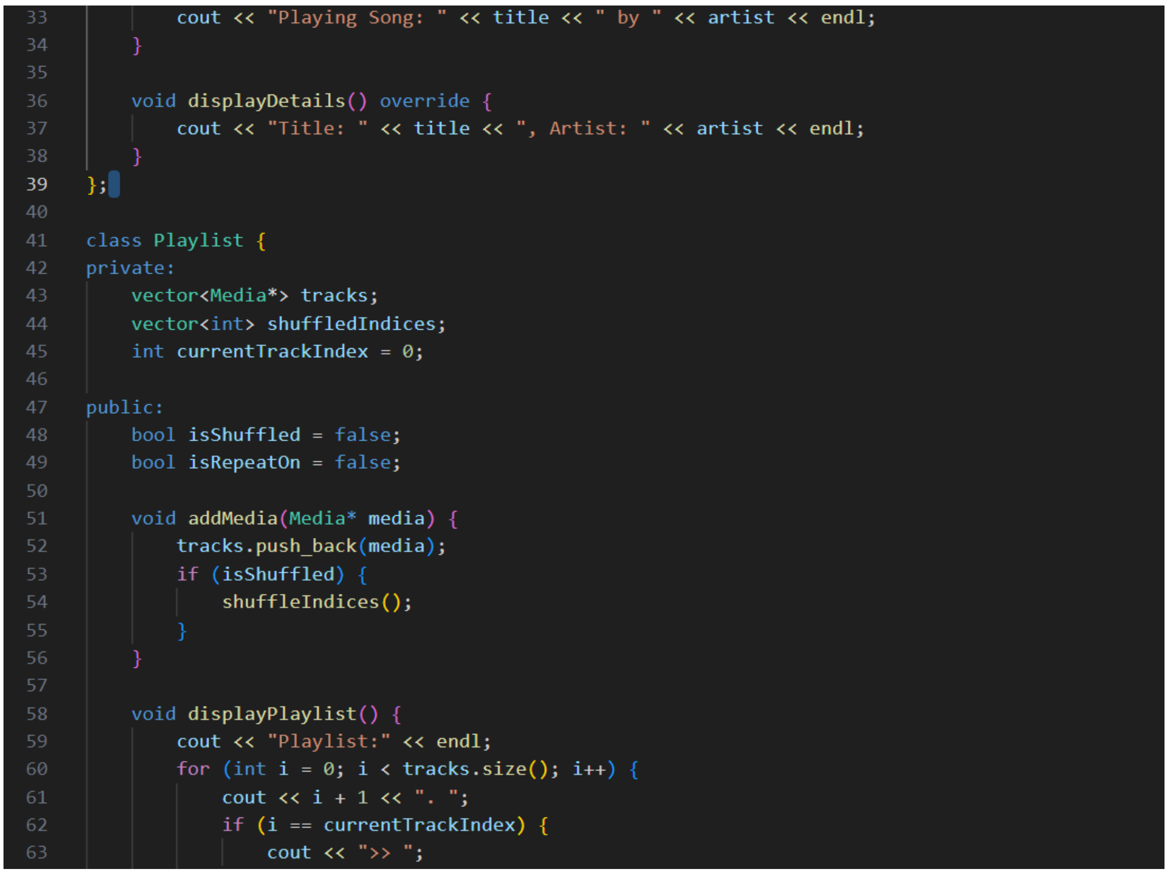
**Class Diagram:**


MEDIA
title: string
• Media(title: string)
• virtual play(): void
• virtual displayDetails(): void
SONG
artist: string
• Song(title: string, artist: string)
• play(): void
• displayDetails(): void
PLAYLIST
tracks: vector<Media*>
shuffledIndices: vector<int>
currentTrackIndex: int
isShuffled: bool
isRepeatOn: bool
• addMedia(media: Media*): void
• displayPlaylist(): void
• playCurrentTrack(): void
• moveNext(): void
• movePrevious(): void
• toggleShuffle(): void
• toggleRepeat(): void
• shuffleIndices(): void
• getNextShuffledIndex(): int
• getPreviousShuffledIndex(): int
• managePlaylistMenu(playlists: vector<Playlist>&, currentPlaylist: int&): void

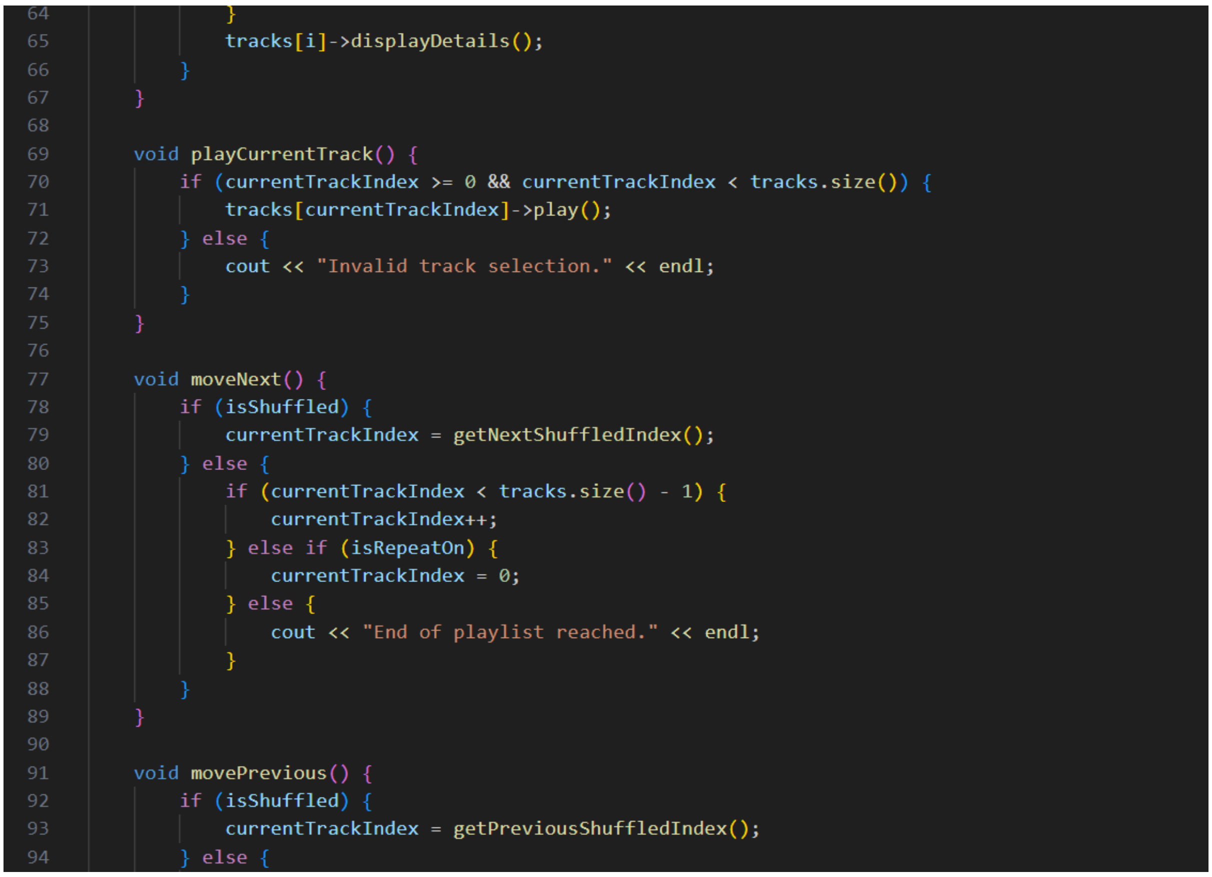
## Code snippets:



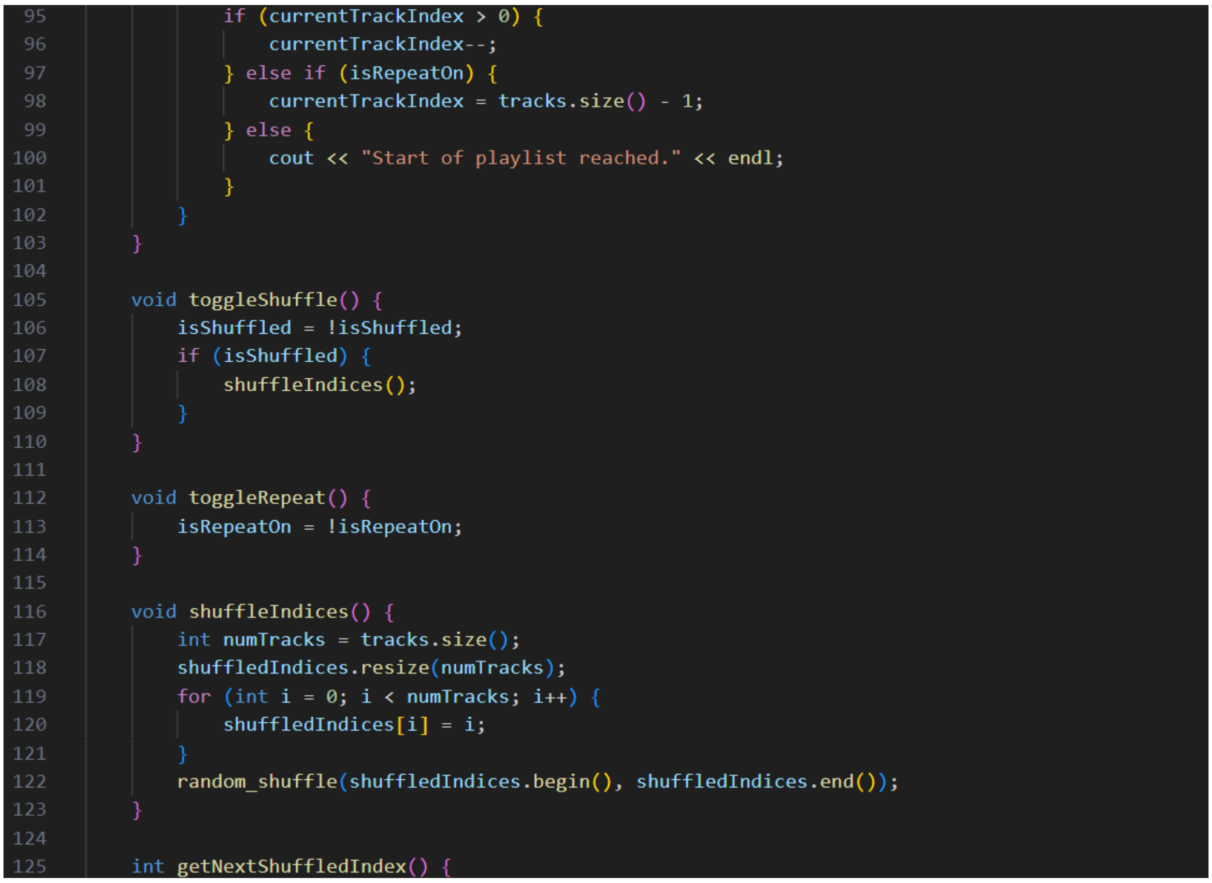
```cpp
#include <iostream>
#include <vector>
#include <string>
#include <algorithm>
#include <ctime>

using namespace std;

class Media {
protected:
    string title;

public:
    Media(const string& title) : title(title) {}

    virtual void play() {
        cout << "Playing: " << title << endl;
    }

    virtual void displayDetails() {
        cout << "Title: " << title << endl;
    }
};

class Song : public Media {
private:
    string artist;

public:
    Song(const string& title, const string& artist) : Media(title), artist(artist) {}

    void play() override {
```



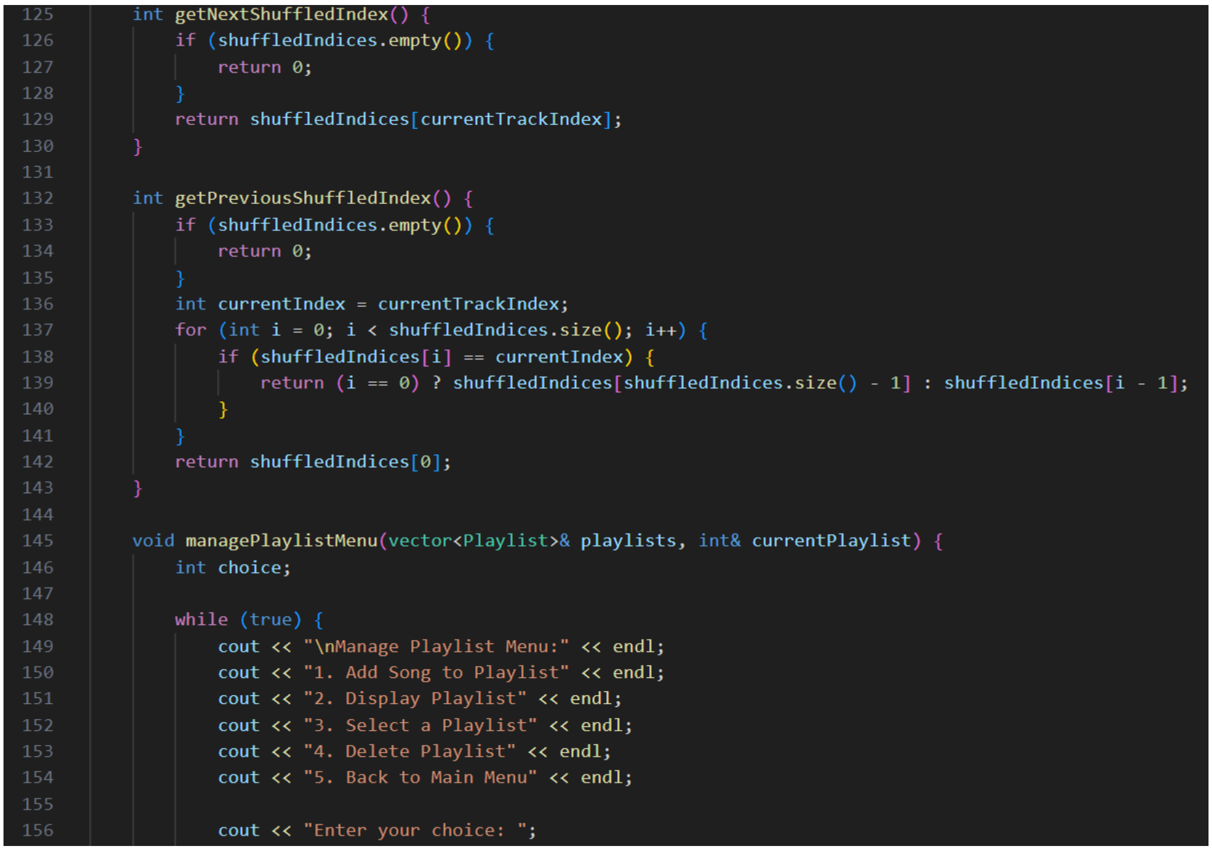
```cpp
        cout << "Playing Song: " << title << " by " << artist << endl;
    }

    void displayDetails() override {
        cout << "Title: " << title << ", Artist: " << artist << endl;
    }
};

class Playlist {
private:
    vector<Media*> tracks;
    vector<int> shuffledIndices;
    int currentTrackIndex = 0;

public:
    bool isShuffled = false;
    bool isRepeatOn = false;

    void addMedia(Media* media) {
        tracks.push_back(media);
        if (isShuffled) {
            shuffleIndices();
        }
    }

    void displayPlaylist() {
        cout << "Playlist:" << endl;
        for (int i = 0; i < tracks.size(); i++) {
            cout << i + 1 << ". ";
            if (i == currentTrackIndex) {
                cout << ">> ";
```



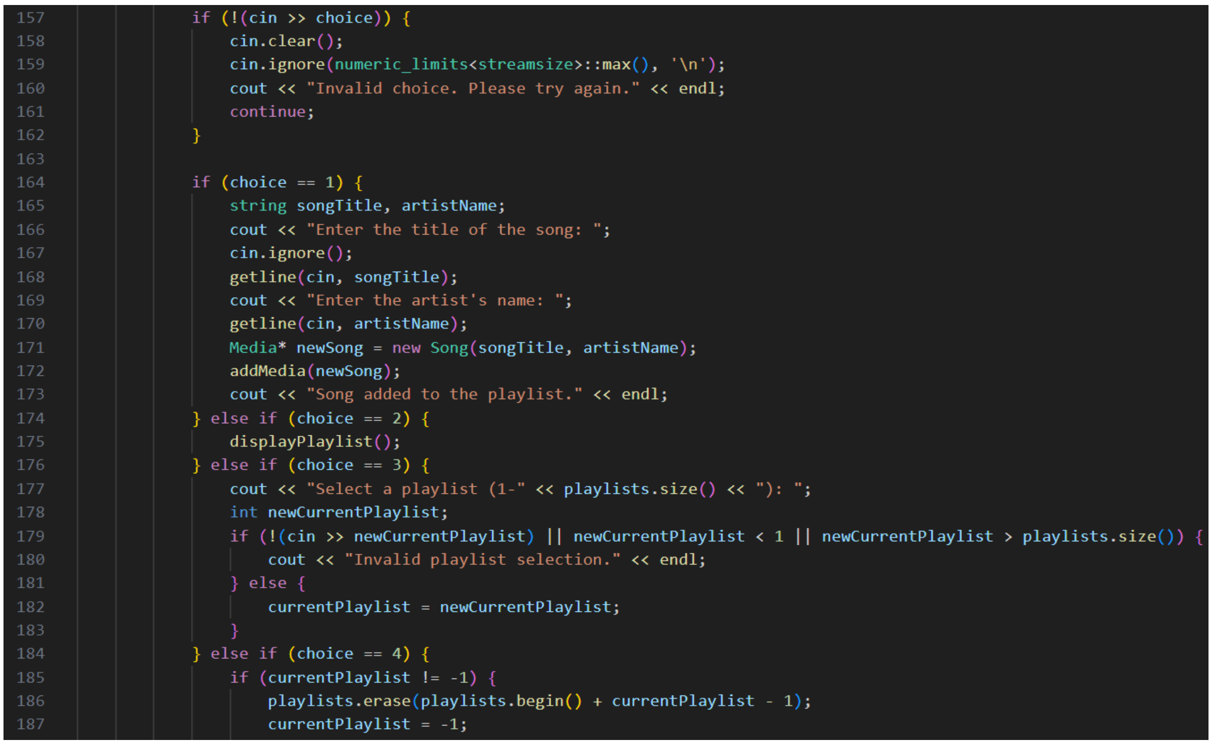
```cpp
            }
            tracks[i]->displayDetails();
        }
    }

    void playCurrentTrack() {
        if (currentTrackIndex >= 0 && currentTrackIndex < tracks.size()) {
            tracks[currentTrackIndex]->play();
        } else {
            cout << "Invalid track selection." << endl;
        }
    }

    void moveNext() {
        if (isShuffled) {
            currentTrackIndex = getNextShuffledIndex();
        } else {
            if (currentTrackIndex < tracks.size() - 1) {
                currentTrackIndex++;
            } else if (isRepeatOn) {
                currentTrackIndex = 0;
            } else {
                cout << "End of playlist reached." << endl;
            }
        }
    }

    void movePrevious() {
        if (isShuffled) {
            currentTrackIndex = getPreviousShuffledIndex();
        } else {
```



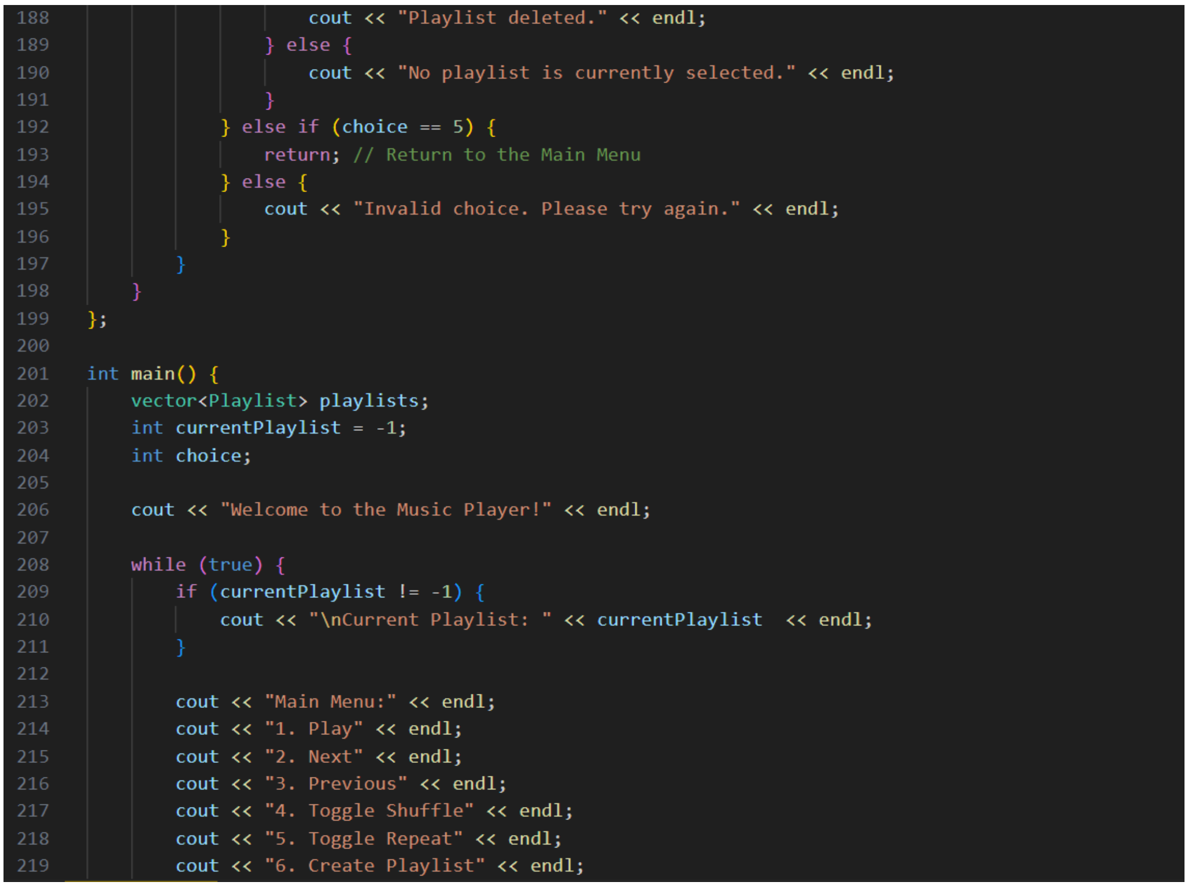
```cpp
            if (currentTrackIndex > 0) {
                currentTrackIndex--;
            } else if (isRepeatOn) {
                currentTrackIndex = tracks.size() - 1;
            } else {
                cout << "Start of playlist reached." << endl;
            }
        }
    }

    void toggleShuffle() {
        isShuffled = !isShuffled;
        if (isShuffled) {
            shuffleIndices();
        }
    }

    void toggleRepeat() {
        isRepeatOn = !isRepeatOn;
    }

    void shuffleIndices() {
        int numTracks = tracks.size();
        shuffledIndices.resize(numTracks);
        for (int i = 0; i < numTracks; i++) {
            shuffledIndices[i] = i;
        }
        random_shuffle(shuffledIndices.begin(), shuffledIndices.end());
    }

    int getNextShuffledIndex() {
```



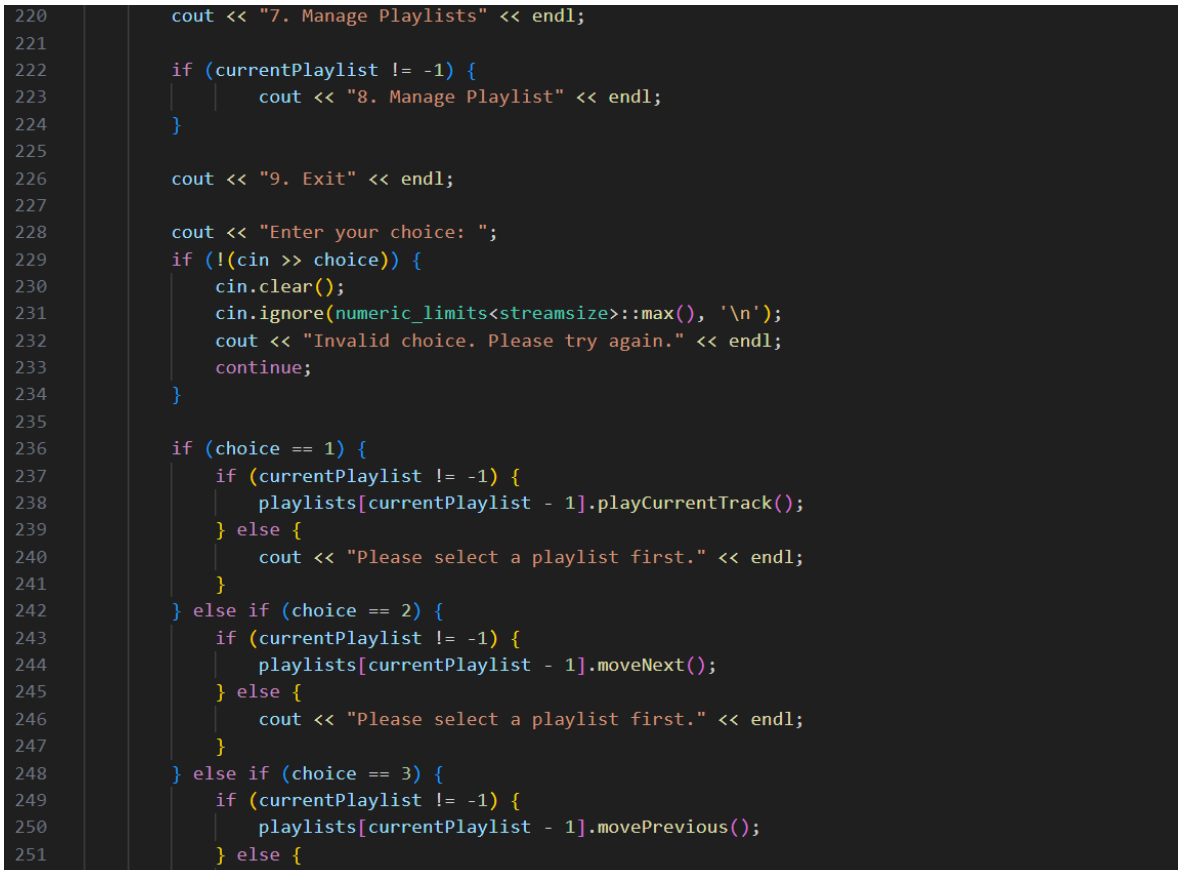
```cpp
    int getNextShuffledIndex() {
        if (shuffledIndices.empty()) {
            return 0;
        }
        return shuffledIndices[currentTrackIndex];
    }

    int getPreviousShuffledIndex() {
        if (shuffledIndices.empty()) {
            return 0;
        }
        int currentIndex = currentTrackIndex;
        for (int i = 0; i < shuffledIndices.size(); i++) {
            if (shuffledIndices[i] == currentIndex) {
                return (i == 0) ? shuffledIndices[shuffledIndices.size() - 1] : shuffledIndices[i - 1];
            }
        }
        return shuffledIndices[0];
    }

    void managePlaylistMenu(vector<Playlist>& playlists, int& currentPlaylist) {
        int choice;

        while (true) {
            cout << "\nManage Playlist Menu:" << endl;
            cout << "1. Add Song to Playlist" << endl;
            cout << "2. Display Playlist" << endl;
            cout << "3. Select a Playlist" << endl;
            cout << "4. Delete Playlist" << endl;
            cout << "5. Back to Main Menu" << endl;

            cout << "Enter your choice: ";
```



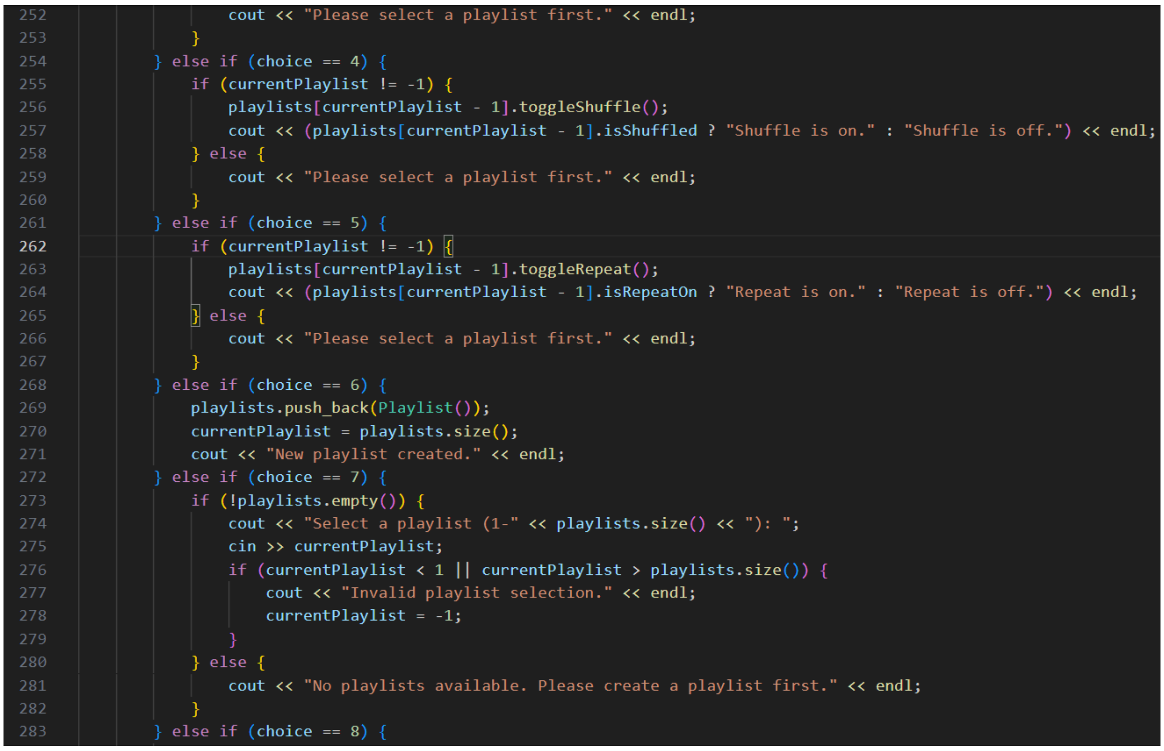
```cpp
            if (!(cin >> choice)) {
                cin.clear();
                cin.ignore(numeric_limits<streamsize>::max(), '\n');
                cout << "Invalid choice. Please try again." << endl;
                continue;
            }

            if (choice == 1) {
                string songTitle, artistName;
                cout << "Enter the title of the song: ";
                cin.ignore();
                getline(cin, songTitle);
                cout << "Enter the artist's name: ";
                getline(cin, artistName);
                Media* newSong = new Song(songTitle, artistName);
                addMedia(newSong);
                cout << "Song added to the playlist." << endl;
            } else if (choice == 2) {
                displayPlaylist();
            } else if (choice == 3) {
                cout << "Select a playlist (1-" << playlists.size() << "): ";
                int newCurrentPlaylist;
                if (!(cin >> newCurrentPlaylist) || newCurrentPlaylist < 1 || newCurrentPlaylist > playlists.size()) {
                    cout << "Invalid playlist selection." << endl;
                } else {
                    currentPlaylist = newCurrentPlaylist;
                }
            } else if (choice == 4) {
                if (currentPlaylist != -1) {
                    playlists.erase(playlists.begin() + currentPlaylist - 1);
                    currentPlaylist = -1;
```



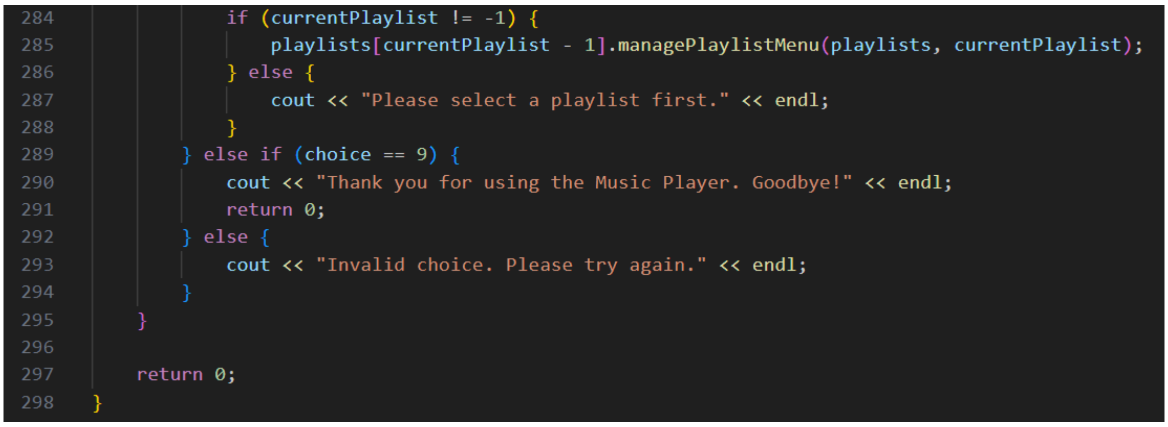
```cpp
                    cout << "Playlist deleted." << endl;
                } else {
                    cout << "No playlist is currently selected." << endl;
                }
            } else if (choice == 5) {
                return; // Return to the Main Menu
            } else {
                cout << "Invalid choice. Please try again." << endl;
            }
        }
    }
};

int main() {
    vector<Playlist> playlists;
    int currentPlaylist = -1;
    int choice;

    cout << "Welcome to the Music Player!" << endl;

    while (true) {
        if (currentPlaylist != -1) {
            cout << "\nCurrent Playlist: " << currentPlaylist  << endl;
        }

        cout << "Main Menu:" << endl;
        cout << "1. Play" << endl;
        cout << "2. Next" << endl;
        cout << "3. Previous" << endl;
        cout << "4. Toggle Shuffle" << endl;
        cout << "5. Toggle Repeat" << endl;
        cout << "6. Create Playlist" << endl;
```



```cpp
        cout << "7. Manage Playlists" << endl;

        if (currentPlaylist != -1) {
                cout << "8. Manage Playlist" << endl;
        }

        cout << "9. Exit" << endl;

        cout << "Enter your choice: ";
        if (!(cin >> choice)) {
            cin.clear();
            cin.ignore(numeric_limits<streamsize>::max(), '\n');
            cout << "Invalid choice. Please try again." << endl;
            continue;
        }

        if (choice == 1) {
            if (currentPlaylist != -1) {
                playlists[currentPlaylist - 1].playCurrentTrack();
            } else {
                cout << "Please select a playlist first." << endl;
            }
        } else if (choice == 2) {
            if (currentPlaylist != -1) {
                playlists[currentPlaylist - 1].moveNext();
            } else {
                cout << "Please select a playlist first." << endl;
            }
        } else if (choice == 3) {
            if (currentPlaylist != -1) {
                playlists[currentPlaylist - 1].movePrevious();
            } else {
```



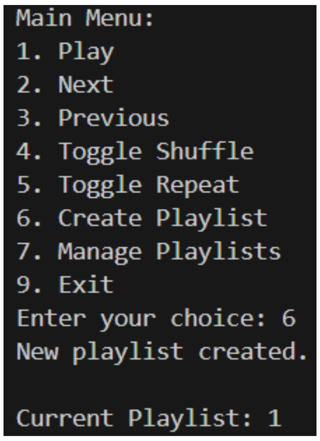
```cpp
                cout << "Please select a playlist first." << endl;
            }
        } else if (choice == 4) {
            if (currentPlaylist != -1) {
                playlists[currentPlaylist - 1].toggleShuffle();
                cout << (playlists[currentPlaylist - 1].isShuffled ? "Shuffle is on." : "Shuffle is off.") << endl;
            } else {
                cout << "Please select a playlist first." << endl;
            }
        } else if (choice == 5) {
            if (currentPlaylist != -1) {
                playlists[currentPlaylist - 1].toggleRepeat();
                cout << (playlists[currentPlaylist - 1].isRepeatOn ? "Repeat is on." : "Repeat is off.") << endl;
            } else {
                cout << "Please select a playlist first." << endl;
            }
        } else if (choice == 6) {
            playlists.push_back(Playlist());
            currentPlaylist = playlists.size();
            cout << "New playlist created." << endl;
        } else if (choice == 7) {
            if (!playlists.empty()) {
                cout << "Select a playlist (1-" << playlists.size() << "): ";
                cin >> currentPlaylist;
                if (currentPlaylist < 1 || currentPlaylist > playlists.size()) {
                    cout << "Invalid playlist selection." << endl;
                    currentPlaylist = -1;
                }
            } else {
                cout << "No playlists available. Please create a playlist first." << endl;
            }
        } else if (choice == 8) {
```



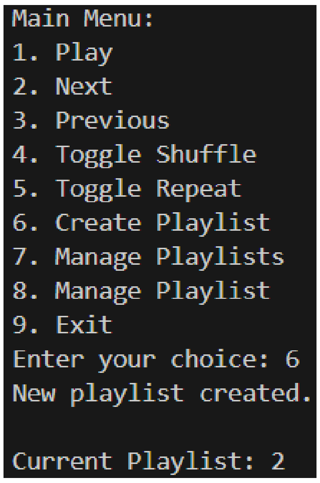
```cpp
            if (currentPlaylist != -1) {
                playlists[currentPlaylist - 1].managePlaylistMenu(playlists, currentPlaylist);
            } else {
                cout << "Please select a playlist first." << endl;
            }
        } else if (choice == 9) {
            cout << "Thank you for using the Music Player. Goodbye!" << endl;
            return 0;
        } else {
            cout << "Invalid choice. Please try again." << endl;
        }
    }

    return 0;
}
```

## Input/Output:

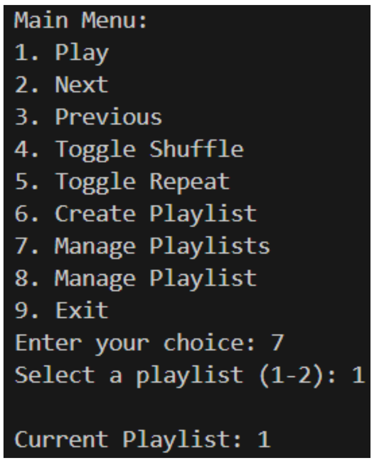
### Creating playlists



```
Main Menu:
1. Play
2. Next
3. Previous
4. Toggle Shuffle
5. Toggle Repeat
6. Create Playlist
7. Manage Playlists
9. Exit
Enter your choice: 6
New playlist created.

Current Playlist: 1
```

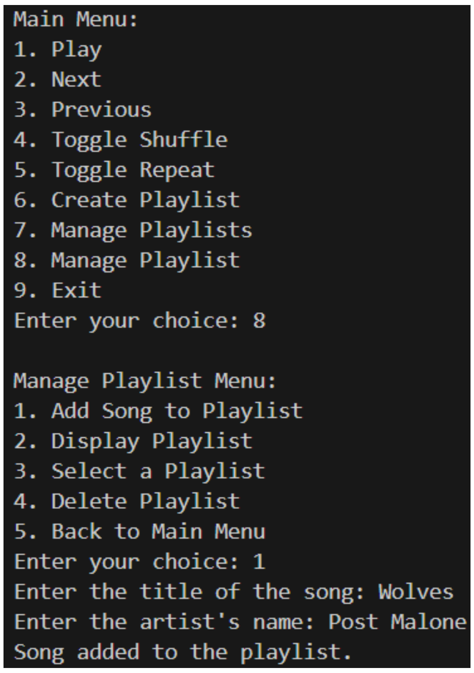

```
Main Menu:
1. Play
2. Next
3. Previous
4. Toggle Shuffle
5. Toggle Repeat
6. Create Playlist
7. Manage Playlists
8. Manage Playlist
9. Exit
Enter your choice: 6
New playlist created.

Current Playlist: 2
```

**<u>Selecting playlists</u>**

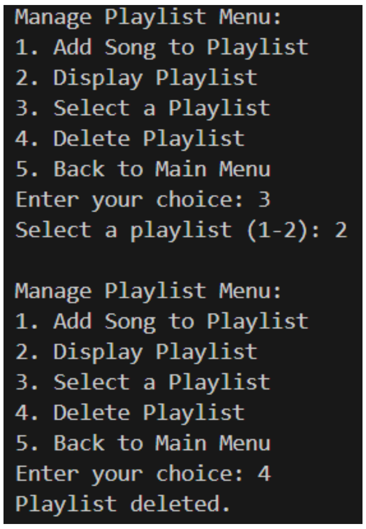

```
Main Menu:
1. Play
2. Next
3. Previous
4. Toggle Shuffle
5. Toggle Repeat
6. Create Playlist
7. Manage Playlists
8. Manage Playlist
9. Exit
Enter your choice: 7
Select a playlist (1-2): 1

Current Playlist: 1
```

**<u>Managing playlist</u>**

1. **<u>Adding song</u>**

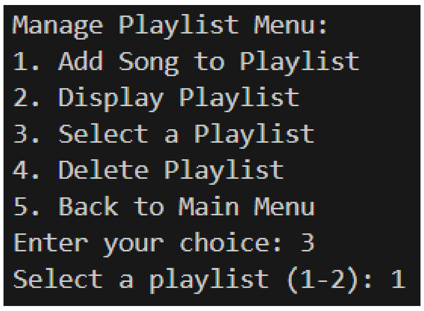

```
Main Menu:
1. Play
2. Next
3. Previous
4. Toggle Shuffle
5. Toggle Repeat
6. Create Playlist
7. Manage Playlists
8. Manage Playlist
9. Exit
Enter your choice: 8

Manage Playlist Menu:
1. Add Song to Playlist
2. Display Playlist
3. Select a Playlist
4. Delete Playlist
5. Back to Main Menu
Enter your choice: 1
Enter the title of the song: Wolves
Enter the artist's name: Post Malone
Song added to the playlist.
```

2. **Deleting selected playlist**

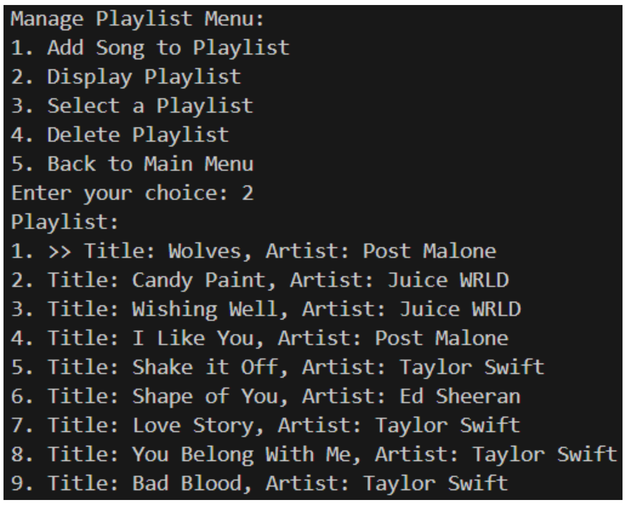


```
Manage Playlist Menu:
1. Add Song to Playlist
2. Display Playlist
3. Select a Playlist
4. Delete Playlist
5. Back to Main Menu
Enter your choice: 3
Select a playlist (1-2): 2

Manage Playlist Menu:
1. Add Song to Playlist
2. Display Playlist
3. Select a Playlist
4. Delete Playlist
5. Back to Main Menu
Enter your choice: 4
Playlist deleted.
```

3. **Displaying selected playlist**

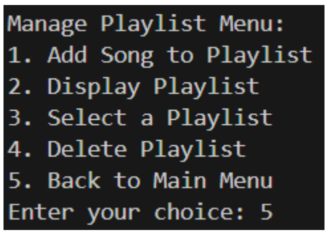


```
Manage Playlist Menu:
1. Add Song to Playlist
2. Display Playlist
3. Select a Playlist
4. Delete Playlist
5. Back to Main Menu
Enter your choice: 3
Select a playlist (1-2): 1
```



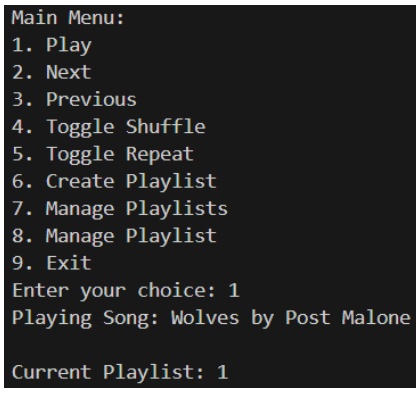
```
Manage Playlist Menu:
1. Add Song to Playlist
2. Display Playlist
3. Select a Playlist
4. Delete Playlist
5. Back to Main Menu
Enter your choice: 2
Playlist:
1. >> Title: Wolves, Artist: Post Malone
2. Title: Candy Paint, Artist: Juice WRLD
3. Title: Wishing Well, Artist: Juice WRLD
4. Title: I Like You, Artist: Post Malone
5. Title: Shake it Off, Artist: Taylor Swift
6. Title: Shape of You, Artist: Ed Sheeran
7. Title: Love Story, Artist: Taylor Swift
8. Title: You Belong With Me, Artist: Taylor Swift
9. Title: Bad Blood, Artist: Taylor Swift
```

4. **Returning to main menu**

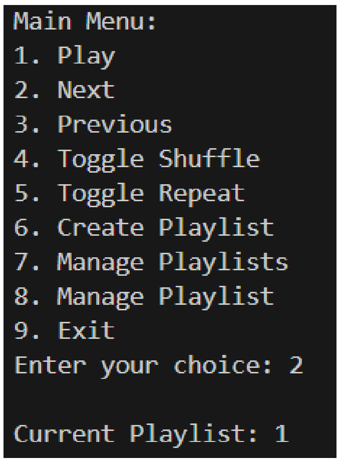


```
Manage Playlist Menu:
1. Add Song to Playlist
2. Display Playlist
3. Select a Playlist
4. Delete Playlist
5. Back to Main Menu
Enter your choice: 5
```

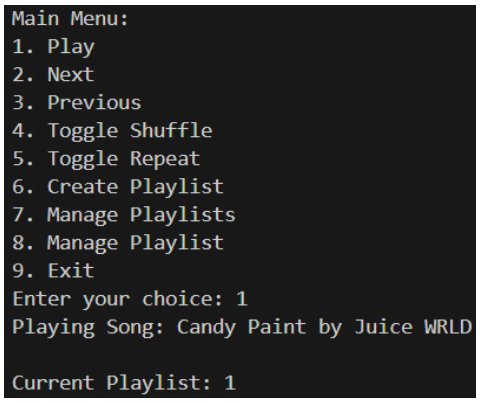
**Playing and Navigating Songs**



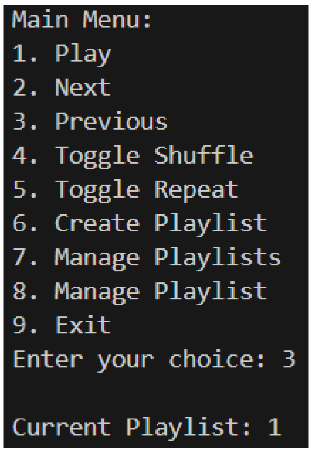
```
Main Menu:
1. Play
2. Next
3. Previous
4. Toggle Shuffle
5. Toggle Repeat
6. Create Playlist
7. Manage Playlists
8. Manage Playlist
9. Exit
Enter your choice: 1
Playing Song: Wolves by Post Malone

Current Playlist: 1
```



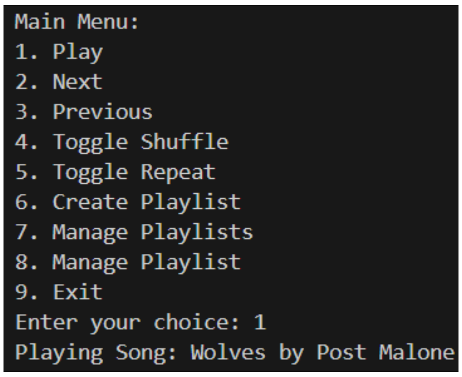
```
Main Menu:
1. Play
2. Next
3. Previous
4. Toggle Shuffle
5. Toggle Repeat
6. Create Playlist
7. Manage Playlists
8. Manage Playlist
9. Exit
Enter your choice: 2

Current Playlist: 1
```



```
Main Menu:
1. Play
2. Next
3. Previous
4. Toggle Shuffle
5. Toggle Repeat
6. Create Playlist
7. Manage Playlists
8. Manage Playlist
9. Exit
Enter your choice: 1
Playing Song: Candy Paint by Juice WRLD

Current Playlist: 1
```

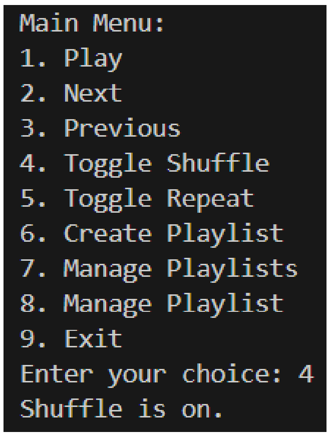


```
Main Menu:
1. Play
2. Next
3. Previous
4. Toggle Shuffle
5. Toggle Repeat
6. Create Playlist
7. Manage Playlists
8. Manage Playlist
9. Exit
Enter your choice: 3

Current Playlist: 1
```

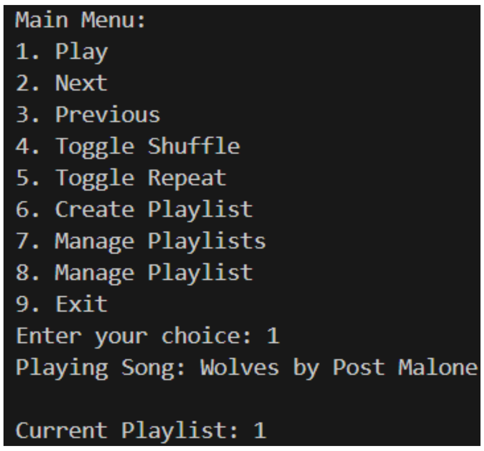


```
Main Menu:
1. Play
2. Next
3. Previous
4. Toggle Shuffle
5. Toggle Repeat
6. Create Playlist
7. Manage Playlists
8. Manage Playlist
9. Exit
Enter your choice: 1
Playing Song: Wolves by Post Malone
```

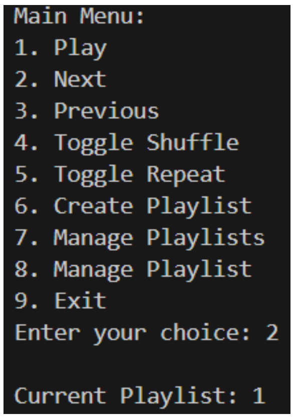
**Toggling shuffle**



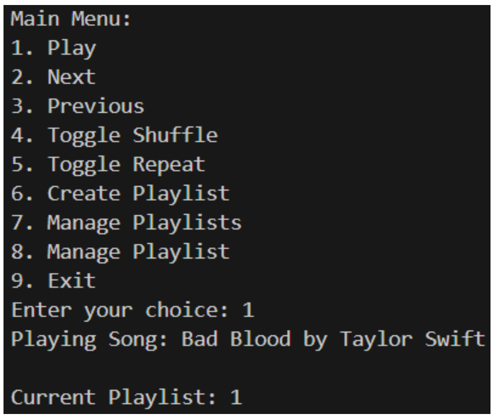
```
Main Menu:
1. Play
2. Next
3. Previous
4. Toggle Shuffle
5. Toggle Repeat
6. Create Playlist
7. Manage Playlists
8. Manage Playlist
9. Exit
Enter your choice: 4
Shuffle is on.
```



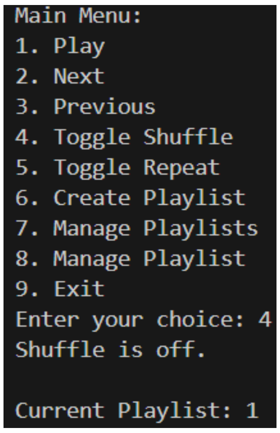
```
Main Menu:
1. Play
2. Next
3. Previous
4. Toggle Shuffle
5. Toggle Repeat
6. Create Playlist
7. Manage Playlists
8. Manage Playlist
9. Exit
Enter your choice: 1
Playing Song: Wolves by Post Malone

Current Playlist: 1
```



```
Main Menu:
1. Play
2. Next
3. Previous
4. Toggle Shuffle
5. Toggle Repeat
6. Create Playlist
7. Manage Playlists
8. Manage Playlist
9. Exit
Enter your choice: 2

Current Playlist: 1
```

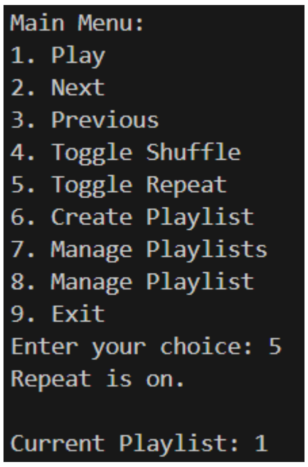


```
Main Menu:
1. Play
2. Next
3. Previous
4. Toggle Shuffle
5. Toggle Repeat
6. Create Playlist
7. Manage Playlists
8. Manage Playlist
9. Exit
Enter your choice: 1
Playing Song: Bad Blood by Taylor Swift

Current Playlist: 1
```

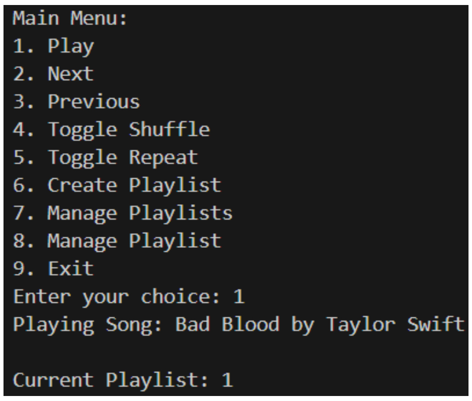


```
Main Menu:
1. Play
2. Next
3. Previous
4. Toggle Shuffle
5. Toggle Repeat
6. Create Playlist
7. Manage Playlists
8. Manage Playlist
9. Exit
Enter your choice: 4
Shuffle is off.

Current Playlist: 1
```

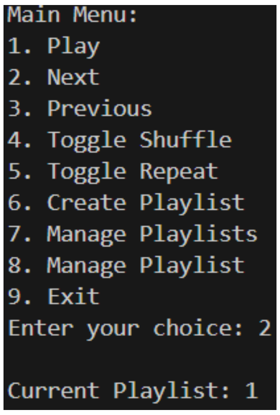
**<u>Toggling repeat</u>**



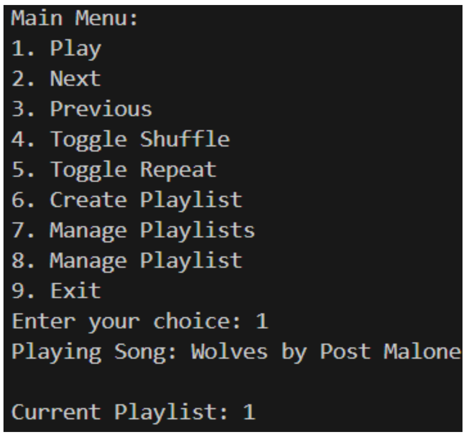
```
Main Menu:
1. Play
2. Next
3. Previous
4. Toggle Shuffle
5. Toggle Repeat
6. Create Playlist
7. Manage Playlists
8. Manage Playlist
9. Exit
Enter your choice: 5
Repeat is on.

Current Playlist: 1
```



```
Main Menu:
1. Play
2. Next
3. Previous
4. Toggle Shuffle
5. Toggle Repeat
6. Create Playlist
7. Manage Playlists
8. Manage Playlist
9. Exit
Enter your choice: 1
Playing Song: Bad Blood by Taylor Swift

Current Playlist: 1
```

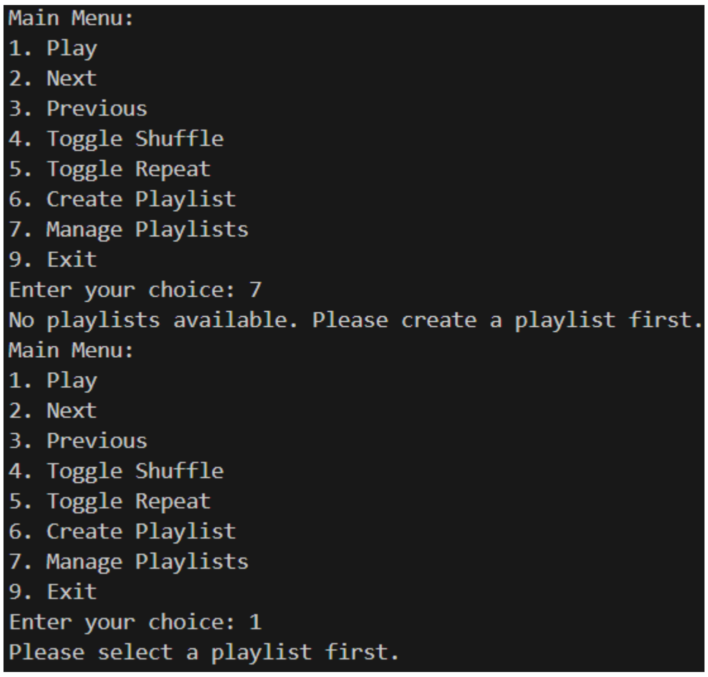


```
Main Menu:
1. Play
2. Next
3. Previous
4. Toggle Shuffle
5. Toggle Repeat
6. Create Playlist
7. Manage Playlists
8. Manage Playlist
9. Exit
Enter your choice: 2

Current Playlist: 1
```

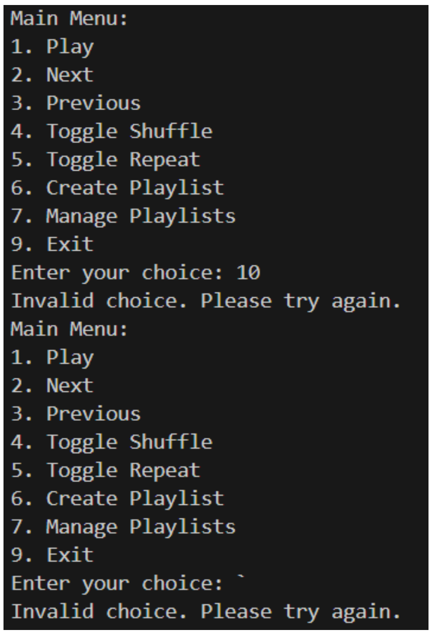


```
Main Menu:
1. Play
2. Next
3. Previous
4. Toggle Shuffle
5. Toggle Repeat
6. Create Playlist
7. Manage Playlists
8. Manage Playlist
9. Exit
Enter your choice: 1
Playing Song: Wolves by Post Malone

Current Playlist: 1
```

**Error Handling**

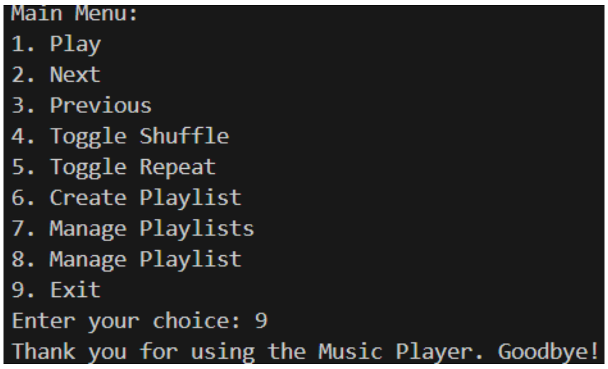


```
Main Menu:
1. Play
2. Next
3. Previous
4. Toggle Shuffle
5. Toggle Repeat
6. Create Playlist
7. Manage Playlists
9. Exit
Enter your choice: 7
No playlists available. Please create a playlist first.
Main Menu:
1. Play
2. Next
3. Previous
4. Toggle Shuffle
5. Toggle Repeat
6. Create Playlist
7. Manage Playlists
9. Exit
Enter your choice: 1
Please select a playlist first.
```



```
Main Menu:
1. Play
2. Next
3. Previous
4. Toggle Shuffle
5. Toggle Repeat
6. Create Playlist
7. Manage Playlists
9. Exit
Enter your choice: 10
Invalid choice. Please try again.
Main Menu:
1. Play
2. Next
3. Previous
4. Toggle Shuffle
5. Toggle Repeat
6. Create Playlist
7. Manage Playlists
9. Exit
Enter your choice: `
Invalid choice. Please try again.
```

**<u>Exiting the program</u>**



```
Main Menu:
1. Play
2. Next
3. Previous
4. Toggle Shuffle
5. Toggle Repeat
6. Create Playlist
7. Manage Playlists
8. Manage Playlist
9. Exit
Enter your choice: 9
Thank you for using the Music Player. Goodbye!
```

## Github repository link:

https://github.com/jahnvinair/PPMusicPlayer