PLAYLIST MANAGEMENT SYSTEM

NAME: SARTHAK SINGH

SECTION: B2

UNIVERSITY ROLL NO: 2319524

INTRODUCTION

- Managing playlists is a common requirement for music enthusiasts and professionals.
- The given project implements a Playlist Management System in C++ that allows users to create, manage, and modify playlists effectively.
- The problem addressed here is organizing songs into playlists with capabilities to add, update, sort, save, and remove songs while ensuring data consistency through unique song IDs.

OBJECTIVE

The primary objective of this code is to provide a robust and user-friendly **Playlist Management System** that allows users to organize and manage their music collections efficiently. The code is designed to demonstrate key programming concepts, such as object-oriented design, data management, file handling, and user interaction, while fulfilling practical requirements for playlist management.

FEATURES

Playlist Management Features

- Create Playlists:
  - Users can create multiple playlists with unique names.
  - Duplicate playlist names are not allowed.
- Display Existing Playlists:
  - Lists all available playlists to help users manage them.

Song Management Features

- Add Songs to Playlists:
  - Songs can be added to a specific playlist using a unique song ID.
  - Input validation ensures only valid data is added (e.g., positive durations and non-duplicate IDs).
- Update Song Details:
  - Users can update existing song details (ID, artist, title, and duration).
  - Prevents assigning duplicate IDs within the same playlist.
- Remove Songs:
  - Allows the removal of a specific song by its ID.
- Sort Songs:
  - Sorts songs within a playlist by their duration in ascending order.

File Management Features

- Save Playlists to Files:
  - Playlists can be saved to a text file for persistence.
  - Each song is stored in a standardized text format.

Validation and Error Handling

- Unique Song IDs:
  - Ensures that each song within a playlist has a unique ID.
  - Prevents accidental overwriting or duplication.
- Input Validation:
  - Checks for valid playlist names, numeric song durations, and non-empty inputs.

Interactive User Interface

- **Menu-Driven Program:**
  - Provides users with a menu to choose actions such as creating playlists, adding songs, and more.
  - Includes options to handle invalid user input gracefully.

LIMITATIONS AND FUTURE ENHANCEMENTS

While the project achieves its goals, some limitations exist:

- **Loading Playlists from Files**: The feature is partially implemented (commented out in the code) and not functional in its current state.
- **Advanced Search Options**: Users cannot search for songs by title, artist, or other criteria.
- **Improved Feedback**: More intuitive error handling and feedback could enhance user experience.

By addressing these limitations, the project can further improve its functionality and user engagement.

CONCLUSION

The Playlist Management System successfully fulfills its primary objectives , providing a functional and user-friendly platform for organizing and managing playlists. This Playlist Management System provides a comprehensive and interactive way to manage multiple music playlists. With the use of object-oriented programming principles, the system allows users to create playlists, add songs, update song details, remove songs, and display playlists in a user-friendly interface. The system supports sorting playlists by song duration and provides functionality to save playlists to a file for later use.

THANK YOU