# Project Report

# for Music Playlist Manager

### Version 1.0

**Submitted to Ms. Tania Khatun Assistant Professor**

## Department of Computer Science and Engineering

### Prepared by

### Md. Mehedi Hasan Shoib: 221-15-5511

**Md. Sabbir Ahamed: 221-15-5553**

**Adnan Rahman Sayeem: 221-15-5505**

**Md. Jahid Hassan Maruf: 221-15-5248**

**Shahnewaz Saeem: 221-15-5442**

**Daffodil International University**

**Date: 08-11-2023**

***Project Report for Data Structure Lab***

### Table of Contents

[Table of Contents ii](#_TOC_250005)

1. [Introduction 3](#_TOC_250004)
   1. Purpose 3
   2. Objective 3
2. [Overall Description 3](#_TOC_250003)
   1. Product Perspective 3
   2. Motivation 3
   3. User Classes and Characteristics 3
3. [Project features and Outcomes 4](#_TOC_250002)
   1. List of Feature 4
   2. Database 4
   3. Outcomes of the Project 4
4. [External Interface Requirements 4](#_TOC_250001)
   1. User Interfaces 4
   2. Input-Output Demo 5-9
5. [Future Scope and Limitation 10](#_TOC_250000)
   1. Limitation 10
   2. Future Scope 10
   3. Conclusion 10

***Project Report for Data Structure Lab***

#### 

#### 1. Introduction

#### 1.1 Purpose: The purpose of the "Music Player" project is to create a versatile and user-friendly music playback application that allows users to manage their music library efficiently. It addresses the need for a feature-rich music player that can be easily extended and customized.

#### 1.2 Objective: The primary objective of this project is to implement a music player using linked lists, queues, and stacks in C. Additionally, our objectives include:

#### To provide users with the ability to add new songs to their library.

#### To enable users to search for specific songs quickly.

#### To create a queuing system for managing the playback order.

#### To delete specific songs by searching them.

#### To allow users to print out recently played songs and queued songs as text files.

#### 1.3 References/Related Works: We referred to existing music player software such as Winamp, VLC Media Player, and related open-source projects to understand design principles and functionality. Additionally, online C programming resources were used for the development process.

#### 2. Overall Description

#### 2.1 Product Perspective: The "Music Player" project is designed as a standalone application that operates independently. It provides users with a comprehensive music playback solution while offering customization options and seamless integration with their music libraries.

#### 2.2 Motivation: The motivation behind this project was to create a music player that combines the challenge of implementing fundamental data structures like linked lists, queues, and stacks with a practical and useful application. We were inspired by the idea of building a music player from scratch.

#### 2.3 User Classes and Characteristics: The primary user classes include music enthusiasts, casual listeners, and anyone who wants a straightforward yet feature-rich music player. Users are expected to have basic computer literacy and a passion for music.

#### 2.4 Operating Environment: The "Music Player" project is designed to run on standard desktop or laptop computers. It is platform-independent and works on Windows, Linux, and macOS. Users need a computer with a compatible operating system and sufficient hardware resources to run the application smoothly.

#### 

#### 3. Project Features and Outcomes:

#### 3.1 List of Features: The features of the "Music Player" project include:

#### Adding new songs to the library

#### Play the next song from playlist

#### Searching for songs by title

#### Printing a list of recently played songs and also store as a text file

#### Printing playlist and exporting the playlist as a text file

#### Delete specific song from playlist

#### 3.2 Database: The project uses a linked list data structure to manage the music library. Each node of the linked list stores information about a song, including title, artist and duration. The queue and stack data structures are used for managing the playlist order and storing recently played songs.

#### 3.3 Outcomes of the Project: The project has successfully implemented all planned features, and the "Music Player" is fully functional. Users can add songs, search for them, create and manage playlists including delete a specific song, and export playlists and play history as text files.

#### External Interface Requirements

#### User Interfaces: The user interface features a clean and intuitive design. Users can easily navigate through their music library, add songs, and manage the playback queue.

#### 

#### Fig-1: User Interface

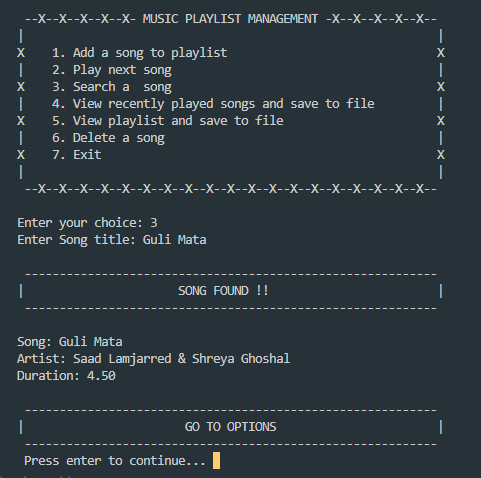
#### 4.2 Input-Output Demo: Here, you can provide step-by-step demonstrations of how users can perform key tasks, including adding a new song, searching for a song, creating a playlist, and exporting playlists and play history.

#### 

#### Fig-2 Add New Song

#### 

#### Fig-3 Play Song

****

**Fig-4 Search Specific Song**



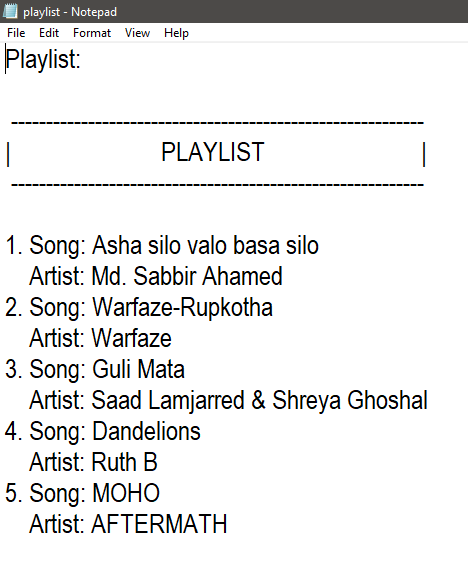
**Fig-5 View Recently Played Songs**

****

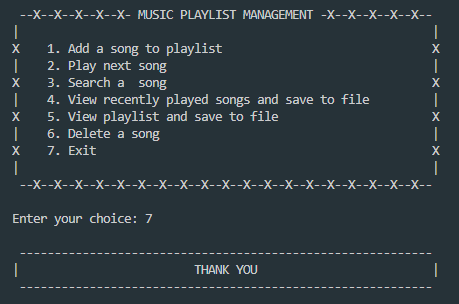
**Fig-6 Recently Played Songs Saved in Text File**

****

**Fig-7 View Play List**

****

**Fig-8 View Play List Saved in Text File**

****

**Fig-9 Thank You**

#### Future Scope and Limitation

#### 5.1 Limitation: The project currently has limitations related to the size of the music library it can handle efficiently and a lack of support for advanced features like equalizer settings or streaming capabilities.

#### 5.2 Future Scope: Future enhancements could include improving performance, expanding supported audio formats, implementing equalizer settings, and adding network capabilities for streaming music. Additionally, user interface enhancements and support for mobile platforms could be explored.

#### 5.3 Conclusion: In conclusion, the "Music Player" project has successfully achieved its objectives of implementing a music player using linked lists, queues, and stacks in C. It provides a functional and extensible solution for music playback and management, and there are ample opportunities for future improvements and feature enhancements.