Apex Prime

Audex
Music Player

Prepared By:

Mohammad Arham Usman (24k-1016)
Khizer Sheikh (24k-0520)
Usman Hasan (24k-0759)

Github: Audex

Apex Prime

Goal

This project will be submitted as partial requirement for the course CS-2001 in fall 2025. It aims to develop a high-performance, feature-rich desktop music player, serving as a practical application to design, implement, and analyze the efficiency of various data structures and algorithms covered in the course.

Description

The Audex Music Player is a C++ application that allows users to import, manage, and play their local music library. Our proposed system will offer a clean, modern graphical user interface for users to easily browse, search, sort, and organize their music collection. The core of the project focuses on using an optimal combination of data structures to ensure the application is both memory-efficient and highly responsive, even with very large music libraries.

Apex Prime

Users

A single primary user who interacts with the application to manage and listen to their music.

Tools

- Language: C++ (Standard: C++17)
- IDE: Microsoft Visual Studio
- Core Libraries:
  - SFML: For window creation, event handling, and audio playback.
  - TGUI: For building the graphical user interface (buttons, lists, sliders).
  - TagLib: For reading metadata (artist, title, album art) from .mp3 files.
  - pfd (Portable File Dialogs):: For opening the native folder selection dialog.
  - Standard C++ Libraries: <filesystem>, <vector>, <string>, etc.

Apex Prime

Features

1. Music Library Management and Playback:
    - Data Structure: Doubly Linked List
    - Algorithm/Use: Serves as the primary container for the music library. Its structure is ideal for sequential playback, enabling efficient O(1) "Next" and "Previous" song operations.
2. Recursive Folder Scanning for Music Import:
    - Algorithm: Recursion
    - Use: To traverse the user's selected directory and all its subdirectories to find and import all .mp3 files into the library.
3. Instant Search by Song Title/Artist:
    - Data Structure: Hash Table
    - Algorithm/Use: Provides near-instant O(1) average-case lookup for songs. The hash table will map song titles and artists to pointers to the nodes in the main Doubly Linked List, powering a real-time search feature.
4. Persistent Sorted Library View:
    - Data Structure: AVL Tree
    - Algorithm/Use: Maintains a perpetually balanced BT of all songs, sorted by artist, with efficient, always-sorted library view and fast O(log n) lookup by artist.

Features

5.Advanced Library Sorting:
- Algorithm: Quick Sort / Merge Sort
- Use: To allow the user to dynamically sort the main library view by various criteria such as title, artist, album, or duration with an efficient O(n log n) sorting algorithm.

6.Playback History Feature:
- Data Structure: Stack (LIFO)
- Use: To manage the history of played songs. The last song played is pushed onto the stack, enabling a logical "Back" functionality with an O(1) pop operation.

7."Play Next" / Play Queue Feature:
- Data Structure: Queue (FIFO)
- Use: To manage a temporary list of songs the user wants to play next. Songs are enqueued and played in the order they were added, providing a fair, temporary override of the main playlist with O(1) operations.

Apex Prime

Schedule

To be submitted one week before the final exam of the fall 2025 semester.

Status

- [ ] Accepted
- [ ] Rejected

Teacher

Course Teacher: Sir Basit Ali

Signature: _____