Object Oriented Programming 2 – Java

Name: Amir Hosein Khanmohammadi

Student ID: 991646689

Instructor: Hosein Marzi

Course: PROG24178

Project Name: Music Catalog System (Beat Browser)

Date: March 03, 2023

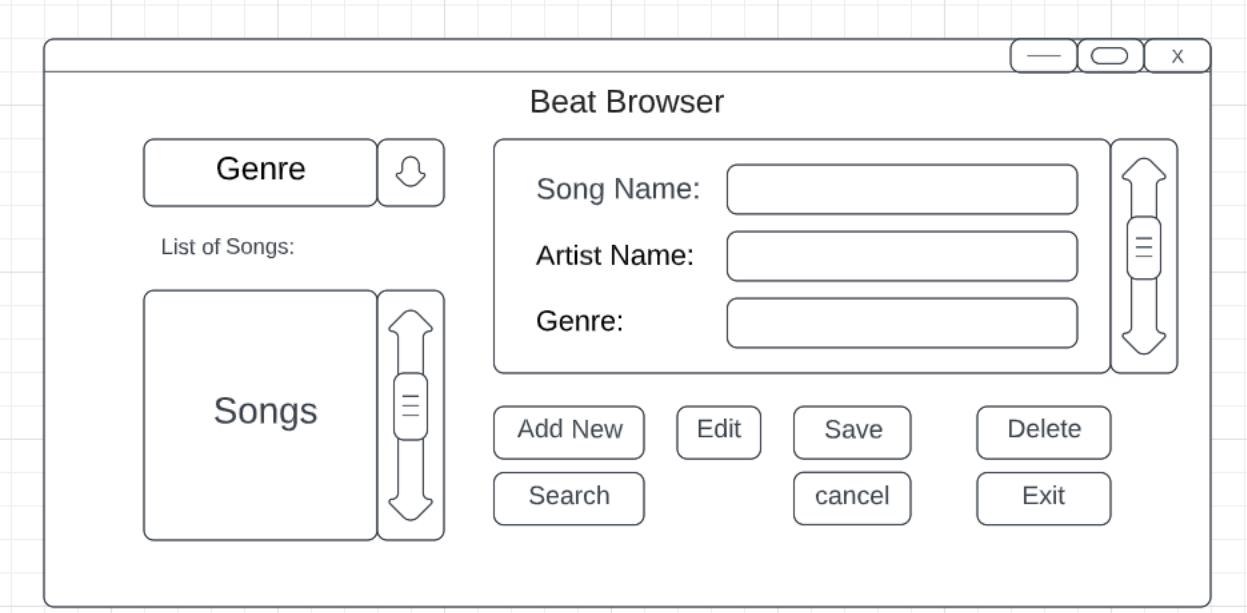
Proposal

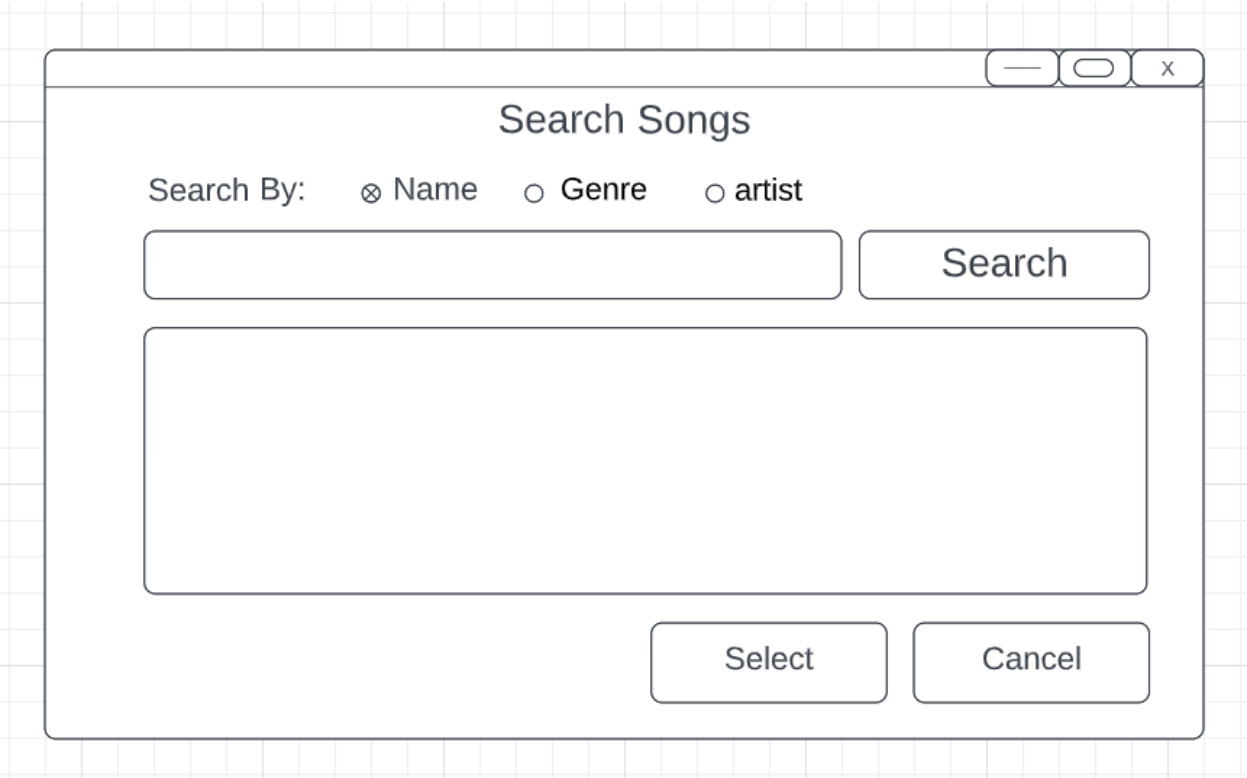
The proposed Music Catalog System is a professional and efficient application that enables music enthusiasts to manage their music collections with ease. The system stores and tracks detailed information of songs across six categories (The categories are Pop, Rock, Jazz, Hip-Hop, Country, and Classical), including title, artist, album, release date, genre, length, writer(s), producer(s), and awards and nominations. This comprehensive database ensures that users can keep their music collections organized and up-to-date. The application is designed to cater to the needs of music enthusiasts who want to maintain a well-organized music collection, and it provides a user-friendly interface that displays the list of songs by category. Users can easily browse, search, edit, and delete songs from their collection, as well as add new songs to it. The search function enables users to find specific songs by title, artist, or category quickly. Editing and deleting songs from the collection is straightforward, with a prompt for confirmation before deleting to prevent accidental deletion. The system also allows users to add new songs to their collection with minimal effort. Overall, the Music Catalog System is an essential tool for music enthusiasts who want to manage their music collections efficiently and accurately.

The file description for the Music Catalog System in a chart format:

Field	Description	Data Type
Title	The name of the song	String
Artist	The name of the performer or band	String
Album	The album on which the song appears	String
Release date	The date the song was officially released	Date
Genre	The musical style of the song	String
Length	The duration of the song in minutes and seconds	String
Writer(s)	The person or people who wrote the lyrics and/or music for the song	String
Producer(s)	The person or people who produced the recording of the song	String
Awards/Nominations	Any awards or nominations the song has received, such as Grammy Awards or Academy Awards	String
ID	Every song got a unique ID whenever added to the database.	int

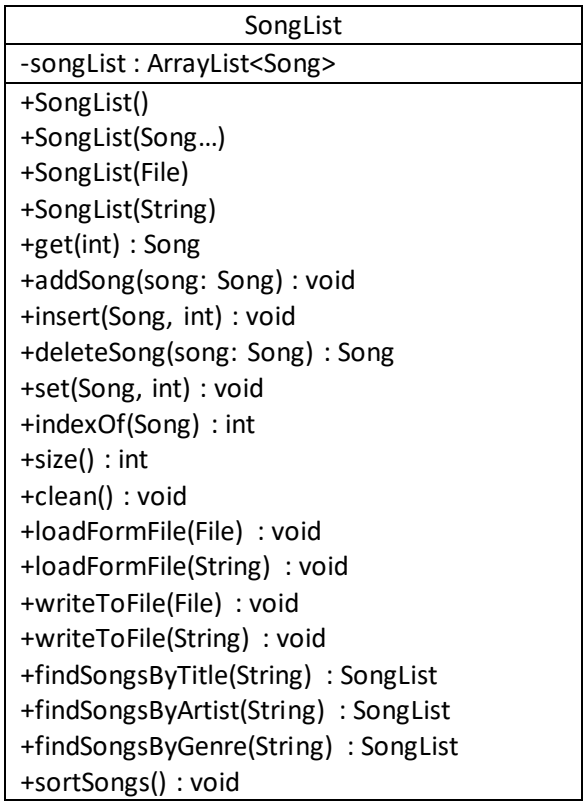
Screen designs or sketches

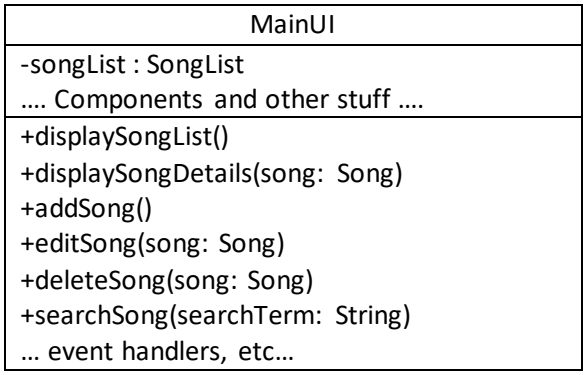


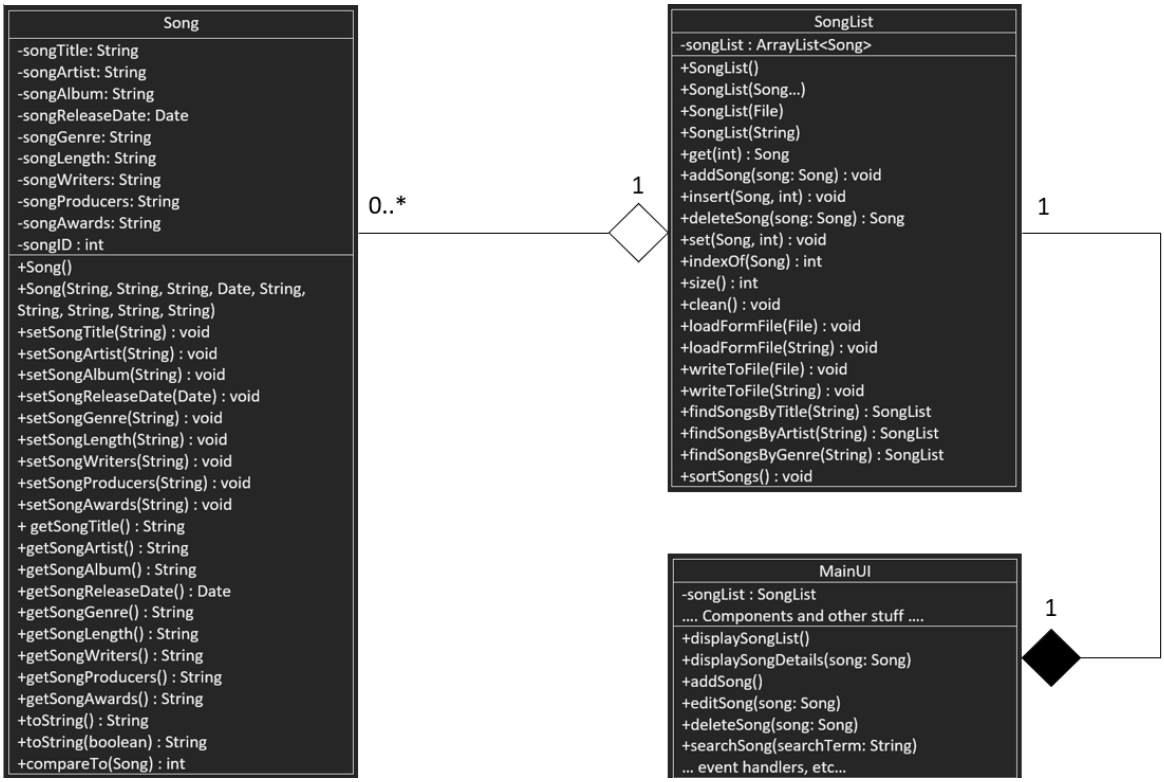


UML Diagrams

Song
-songTitle: String -songArtist: String -songAlbum: String -songReleaseDate: Date -songGenre: String -songLength: String -songWriters: String -songProducers: String -songAwards: String -songID : int
+Song() +Song(String, String, String, Date, String, String, String, String, String) +setSongTitle(String) : void +setSongArtist(String) : void +setSongAlbum(String) : void +setSongReleaseDate(Date) : void +setSongGenre(String) : void +setSongLength(String) : void +setSongWriters(String) : void +setSongProducers(String) : void +setSongAwards(String) : void + getSongTitle() : String +getSongArtist() : String +getSongAlbum() : String +getSongReleaseDate() : Date +getSongGenre() : String +getSongLength() : String +getSongWriters() : String +getSongProducers() : String +getSongAwards() : String +toString() : String +toString(boolean) : String +compareTo(Song) : int


SongList
-songList : ArrayList<Song>
+SongList()
+SongList(Song…)
+SongList(File)
+SongList(String)
+get(int) : Song
+addSong(song: Song) : void
+insert(Song, int) : void
+deleteSong(song: Song) : Song
+set(Song, int) : void
+indexOf(Song) : int
+size() : int
+clean() : void
+loadFormFile(File) : void
+loadFormFile(String) : void
+writeToFile(File) : void
+writeToFile(String) : void
+findSongsByTitle(String) : SongList
+findSongsByArtist(String) : SongList
+findSongsByGenre(String) : SongList
+sortSongs() : void


MainUI
-songList : SongList
…. Components and other stuff ….
+displaySongList()
+displaySongDetails(song: Song)
+addSong()
+editSong(song: Song)
+deleteSong(song: Song)
+searchSong(searchTerm: String)
… event handlers, etc…


Song
-songTitle: String
-songArtist: String
-songAlbum: String
-songReleaseDate: Date
-songGenre: String
-songLength: String
-songWriters: String
-songProducers: String
-songAwards: String
-songID : int
+Song()
+Song(String, String, String, Date, String, String, String, String, String)
+setSongTitle(String) : void
+setSongArtist(String) : void
+setSongAlbum(String) : void
+setSongReleaseDate(Date) : void
+setSongGenre(String) : void
+setSongLength(String) : void
+setSongWriters(String) : void
+setSongProducers(String) : void
+setSongAwards(String) : void
+ getSongTitle() : String
+getSongArtist() : String
+getSongAlbum() : String
+getSongReleaseDate() : Date
+getSongGenre() : String
+getSongLength() : String
+getSongWriters() : String
+getSongProducers() : String
+getSongAwards() : String
+toString() : String
+toString(boolean) : String
+compareTo(Song) : int
0..*
1
SongList
-songList : ArrayList<Song>
+SongList()
+SongList(Song...)
+SongList(File)
+SongList(String)
+get(int) : Song
+addSong(song: Song) : void
+insert(Song, int) : void
+deleteSong(song: Song) : Song
+set(Song, int) : void
+indexOf(Song) : int
+size() : int
+clean() : void
+loadFormFile(File) : void
+loadFormFile(String) : void
+writeToFile(File) : void
+writeToFile(String) : void
+findSongsByTitle(String) : SongList
+findSongsByArtist(String) : SongList
+findSongsByGenre(String) : SongList
+sortSongs() : void
1
1
MainUI
-songList : SongList
.... Components and other stuff ....
+displaySongList()
+displaySongDetails(song: Song)
+addSong()
+editSong(song: Song)
+deleteSong(song: Song)
+searchSong(searchTerm: String)
... event handlers, etc...

To develop a high-quality Music Catalog System, there are several classes/packages and concepts that the developer will need to research and understand thoroughly. These include:

- **File Input/Output**: The program must be able to read and write data to a file in order to store the list of songs and their details. The developer will need to research file input/output in Java, including best practices for working with files, error handling, and security considerations.

- **GUI Development**: Developing a user-friendly and visually appealing graphical user interface (GUI) is essential for any application. The developer will need to learn how to use Java's Swing or JavaFX libraries to create windows, buttons, text boxes, and other GUI components. They must also research best practices for UI design, including layout, color schemes, and typography.

- **Collections Framework**: The Java Collections Framework is a powerful set of classes and interfaces that provide a way to store and manipulate groups of objects. The developer must learn how to use ArrayLists and HashMaps to store and manage the list of songs and their details, and research best practices for working with collections.

- **Regular Expressions**: When searching for songs in the catalog, regular expressions can be used to match search terms to song titles or artist names. The developer must research regular expressions in Java, including syntax, functions, and performance considerations.

- **Data Validation**: To ensure that the user enters valid data when adding or editing a song, the program must perform data validation. The developer must research best practices for data validation in Java, including input sanitization, error handling, and security considerations.

Overall, developing a Music Catalog System is a complex and challenging task that requires in-depth knowledge of several core Java concepts, as well as some more specialized topics related to file I/O, GUI development, and data validation. The developer must be committed to thorough research and testing in order to ensure that the final product is of high quality and meets the needs of the target user.