**Music Library Management System Project Report**

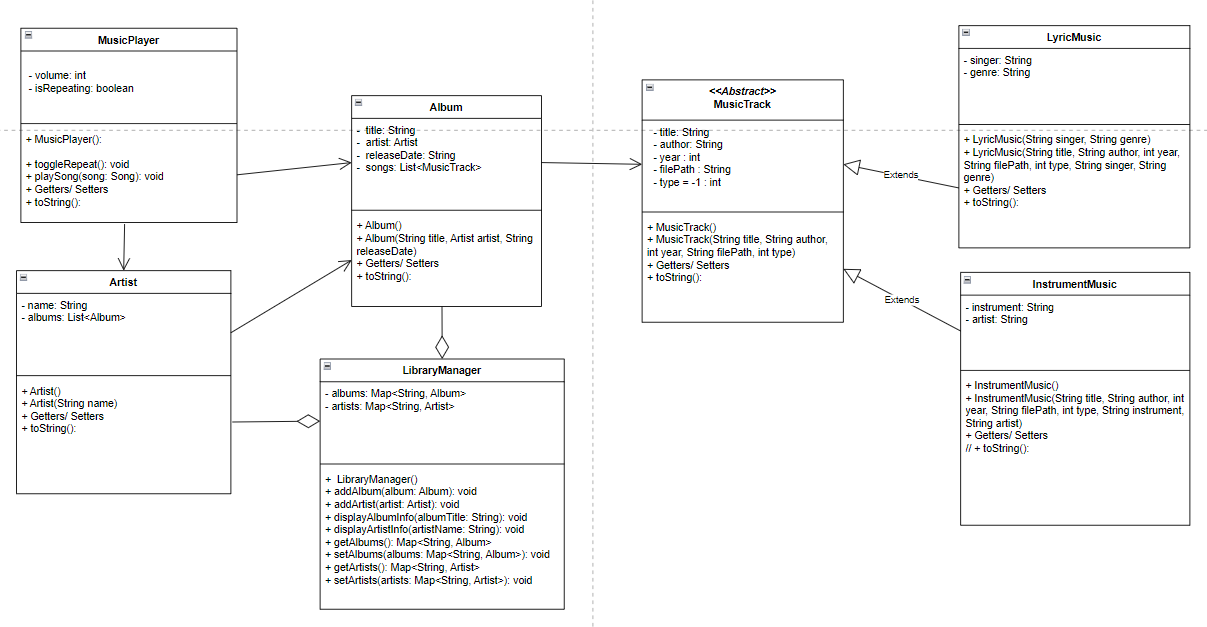
# **1. Project Overview**

* **Project Title:** Music Library Management System
* **Team Members:**
  + Phạm Nguyễn Bảo Trang
  + Trần Thị Ngọc Thủy
* **Submission Date:** 20/08/2024
* **Project Description:** This project is a Music Library Management System developed as a console application. It enables the admin to manage and play songs, organize playlists, search by keyword, and sort by types of music.

# **2. System Design**

**2.1 UML Diagrams**

* **Class Diagram:**

**

**2.2 Class Design**

| **Class Name** | **Description** | **Responsibilities** | **Attributes** | **Methods** |
| --- | --- | --- | --- | --- |
| **MusicPlayer** | Core class of the system | Manages playlists, playback, volume, and repeat mode. | albums, artists, volume, isRepeating, currentSongIndex | addAlbum(), addArtist(), setCurrentPlaylist(), setVolume(), toggleRepeat(), playSong() |
| **LibraryManager** | Manages music library | Handles the storage and retrieval of albums and artists | albums, artists | addAlbum(), addArtist(), getAlbums(), getArtists(), setAlbums(), setArtists() |
| **MusicTrack** | Represents a generic song | Defines common attributes and methods for all song types. | title, author, year, filePath, type | getTitle(), getAuthor(), getYear(), getFilePath(), getType() |
| **LyricMusic** | Song with lyrics | Holds information specific to songs with lyrics. | singer, genre (inherits from MusicTrack) | getSinger(), setSinger(), getGenre(), setGenre(), getDescription() |
| **InstrumentMusic** | Instrumental song | Holds information specific to instrumental songs. | instrument, artist (inherits from MusicTrack) | getInstrument(), setInstrument(), getArtist(), setArtist(), getDescription() |
| **Album** | Represents a music album | Stores information about an album and its songs. | title, artist, releaseDate, songs | addSong(), getSongs(), getArtist(), getTitle() |
| **Artist** | Represents a music artist | Holds information about an artist and their albums. | name, albums | getName(), setName(), getAlbums(), setAlbums() |

# **3. SOLID Principles Implementation**

**3.1 Single Responsibility Principle (SRP)**

Each class in the code adheres to SRP as each has a single responsibility:

· MusicTrack handles general track information.

· LyricMusic and InstrumentMusic extend from MusicTrack with specific functionalities.

· Artist represents an artist's information and associated albums.

· Album handles album-related functionalities.

· MusicPlayer manages the music library and playback features.

**3.2 Open/Closed Principle (OCP)**

* MusicTrack is open for extension (subclasses like and ), but not for modification: LyricMusic and InstrumentMusic
* Adding new track types would not require modifying existing classes.
* LibraryManager

**3.3 Liskov Substitution Principle (LSP)**

* LyricMusic and InstrumentMusic can be used interchangeably wherever a MusicTrack object is expected, ensuring consistent behavior.

**3.4 Interface Segregation Principle (ISP)**

**3.5 Dependency Inversion Principle (DIP)**

* The MusicPlayer class depends on the abstract MusicTrack class rather than concrete subclasses like LyricMusic or InstrumentMusic, allowing for flexibility and easier extension.

# **4. Console Application**

* **Description:** The console application provides a text-based interface for users to interact with the music library. Users can play songs, manage playlists, view information about artists and albums, and control playback settings.
* **Sample Input/Output:**
  + **Input:** 4 (Choose option 4: Play a song)
  + **Input:** Summer Vibes (Enter the title of the song)
  + **Output:** Now playing: LyricMusic [singer=Bob Williams, genre=Pop, title=Summer Vibes, artist=Pop Artist] from file: /path/to/summer.mp3 at volume: 50%
  + **Input:** 5 (Choose option 5: Toggle repeat mode)
  + **Output:** Repeat mode: ON

**5. Swing GUI Application**

* **Description:**

# **6. Challenges and Solutions**

* **Challenges:**
  + Implementing robust music player functionality within the console application.
  + Organizing and managing the data for songs, artists, and albums effectively.
* **Solutions:**
  + Utilized appropriate data structures like ArrayList and HashMap to store and manage data.
  + Implemented the MusicPlayer class with core functionalities for playback, volume control, and repeat mode.

# **7. Conclusion**

This project successfully implemented a Music Library Management System with functionalities for playing songs, managing playlists, and viewing information about artists and albums. The console application provides a user-friendly interface for interacting with the library. While a GUI is not currently implemented, the project demonstrates the core functionalities of a music library system.

# **8. Appendix**

* **Code:** *(Include snippets of key code sections here. Optionally, attach the full source code as a separate file.)*
* **References:** *(List any references used during the project.)*