PES UNIVERSITY

(Established under Karnataka Act No. 16 of 2013)

Object Oriented Analysis and Design using Java (UE20CS352)

Mini Project - Sasta Spotify

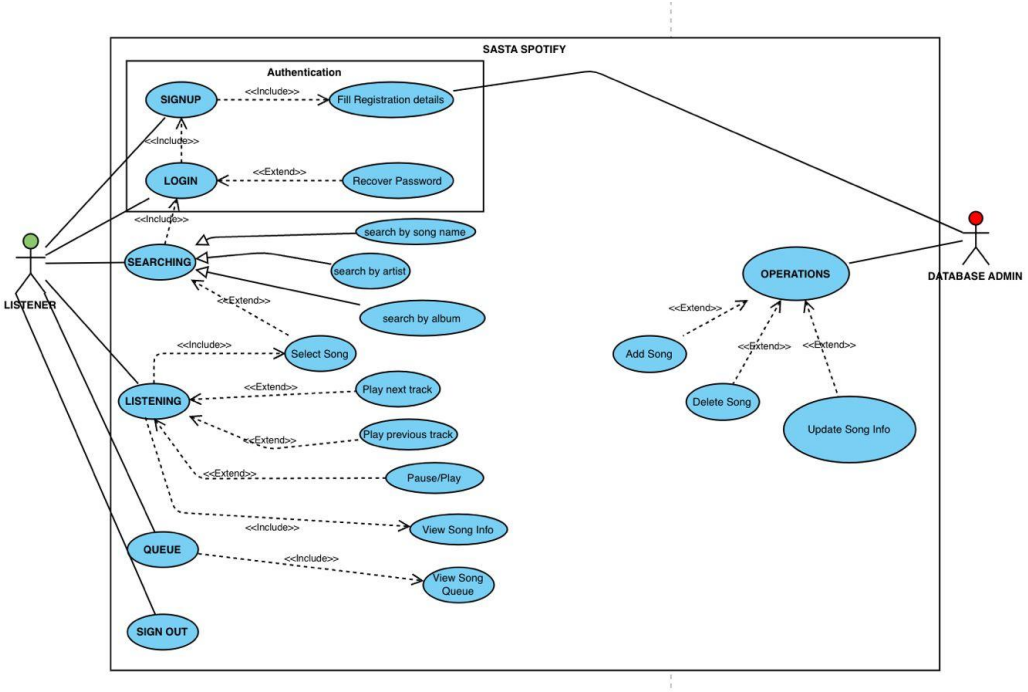
Team Details :

Mihir Santosh Sanjay	PES1UG20CS253
Meghana Anand	PES1UG20CS252
Pooshpal Baheti	PES1UG20CS283
Prateek Rao	PES1UG20CS303

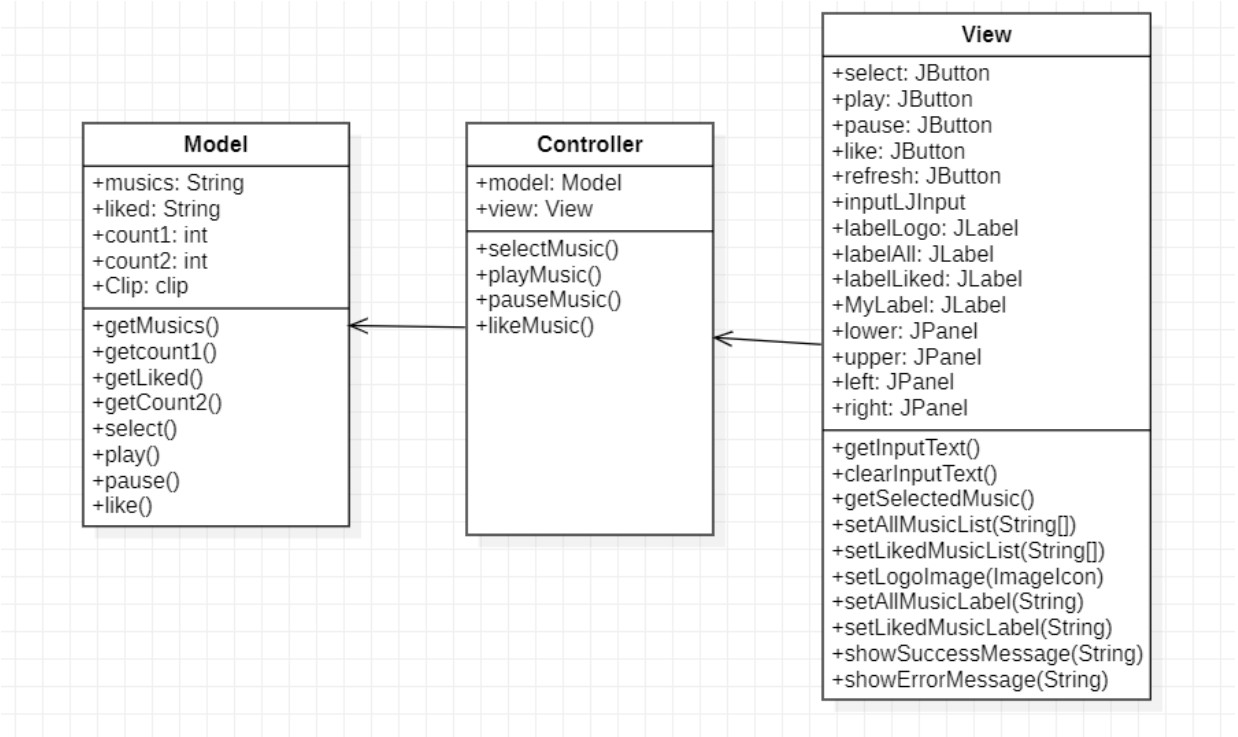
Problem Statement:

Sasta SPotify - The Music Player project is a software application that allows users to play, manage, and like their music files on their desktop computers. The project utilizes Java. Music is an integral part of our lives and people listen to music for various reasons such as relaxation, entertainment, and motivation. With the increasing availability of digital music, there is a need for a reliable and convenient way to manage and play music files. The Music Player project aims to provide a comprehensive and convenient solution to manage and play music files on desktop computers.

Use Case Diagram:



Class Model:



Design Principles Used:

GRASP Principles

1. Controller: The Controller pattern is responsible for receiving user input and updating the model and the view accordingly. In the music player project, the controller would be responsible for receiving user input such as play, pause, and stop, and updating the music player's view and model accordingly.

2. Low Coupling: The Low Coupling pattern suggests that classes should have minimal dependencies on other classes.

3. High Cohesion: The High Cohesion pattern suggests that classes should be designed to have a single, well-defined purpose.

SOLID Principles

1. Single Responsibility Principle (SRP):
The Model class has only one responsibility, which is to handle data and the related logic. It loads data from text files, initializes and stores data in ArrayLists, retrieves data for the View class, and performs some basic operations such as selecting music, playing, pausing, and liking a particular music file. Similarly, the View class is responsible for managing the user interface and providing appropriate feedback to the user.

2. Open-Closed Principle (OCP):
The View class is open for extension but closed for modification, which means that the class's behavior can be extended without altering its source code. It uses polymorphism to display images on the screen, and the implementation of the ImagePanel class can be changed without affecting the View class's functionality.

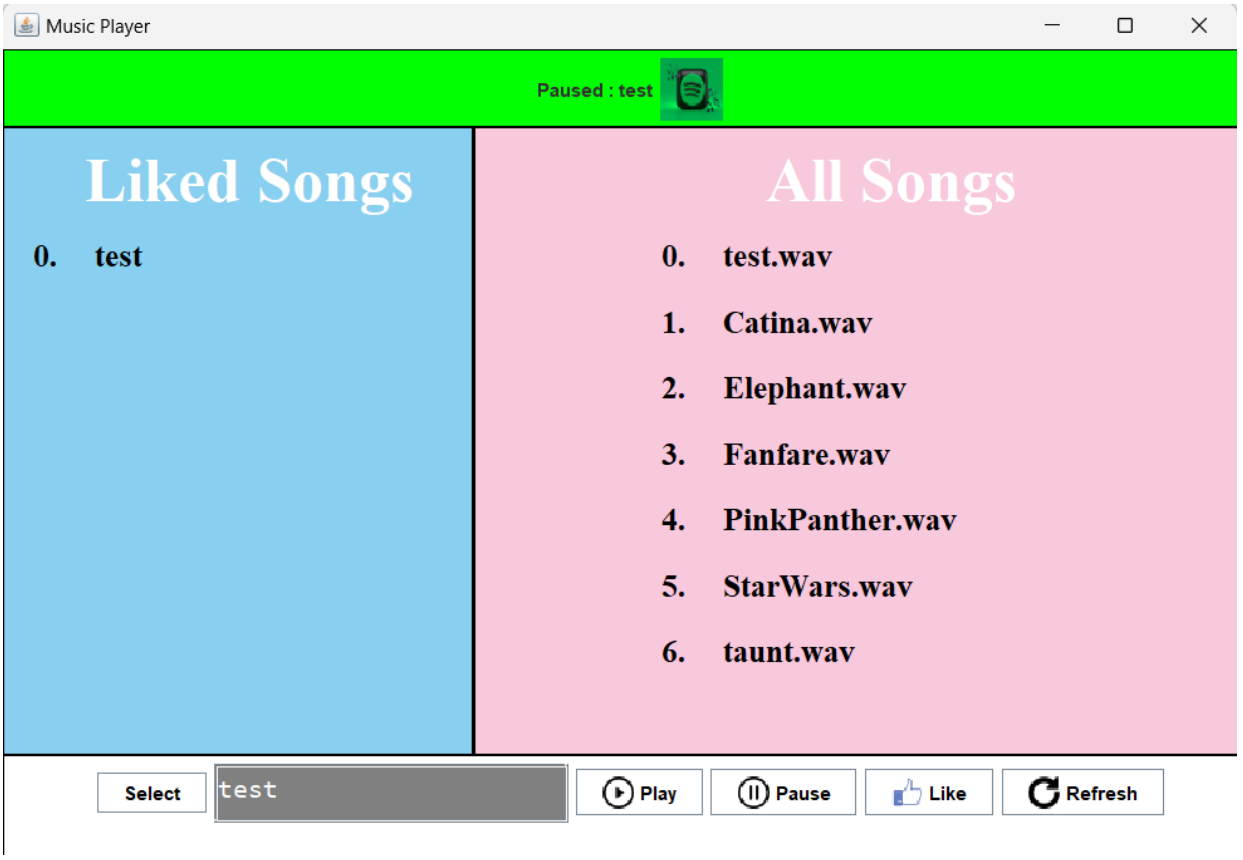
3. Interface Segregation Principle (ISP):
The given code does not explicitly define any interfaces; however, it follows the ISP principle in spirit by creating small and cohesive classes that provide only the functionality that they are intended to provide. For instance, the Model class is responsible for handling data, and the View class is responsible for handling the UI components.

4.Dependency Inversion Principle (DIP):
The Model class has no dependencies on any other class, and it only depends on the data and the logic it needs to implement. The View class's constructor takes no parameters, and it creates and uses objects independently without depending on any external resources. Thus, the given code satisfies the DIP.

Submission Github Link:
https://github.com/Pooshpal/Sasta-Spotify

Individual Contributions:
The base functionality of generating a Framework and MVC architecture was created by Pooshpal. The functionality of the Music Player to play, pause and like songs was built by Prateek Rao. The functionality of Status Bar, Loading the music library files was built by Meghana. Finally the functionality to like songs and refresh the liked song list was worked on by Mihir. The project was a success due to this combined team effort. Important components were built first and then the later functionalities were added on.

Screenshots of Project Demo:



Thank You.