Spotifoo PLAYER

Table of Contents

[1. About 2](#_Toc109519141)

[2. Project Overview 2](#_Toc109519142)

[2.1 Play Song 2](#_Toc109519143)

[2.2 Filter By Artist 2](#_Toc109519144)

[2.3 Filter By Album 2](#_Toc109519145)

[2.4 Filter By Genre 2](#_Toc109519146)

[2.5 Search 2](#_Toc109519147)

[2.6 Sample data 3](#_Toc109519148)

[3. Implementation and Design 3](#_Toc109519149)

[3.1 Technologies 3](#_Toc109519150)

[3.2 Design and Flow diagram 3](#_Toc109519151)

[3.2.1 Architecture and design 3](#_Toc109519152)

[3.2.2 Class diagram 4](#_Toc109519153)

[3.2.3 Sequence diagram 4](#_Toc109519154)

[3.2.4 Model diagram 5](#_Toc109519155)

[3.2.5 Menus flow charts 6](#_Toc109519156)

[3.2.6 Screens 6](#_Toc109519157)

# About

Spotifoo is command line interface application, that allows you to stream music instead of downloading it from an unknown source. The user can listen to the songs according to his choice and he can able to choose the song based on his interest. We can only navigate by typing a number from an option list and then press enter to execute the command. Typing 0 and pressing enter in any part of the application will send you back to the main menu. This application is providing features to users like display all the songs, by category (album, artist and genre) and searching the songs by name.

# Project Overview

The application is a command line interface (terminal) application that does not have a graphical user interface. The user only navigates by typing a number from an option list and then press enter to execute the command. Typing 0 and pressing enter in any part of the application will send you back to the main menu.

## Play Song

When pressing option #1 on the main menu, the app must show a list of songs by reading the text file located inside the asset folder. When the user chooses a specific song, the music file should open in the default music player installed in your computer. In addition, it should also open the corresponding album art. Or show a placeholder image if the album art is not found.

**Note:** Only open the album art, if the app can play the song.

## Filter By Artist

When pressing option #2 on the main menu, the app will replace the main menu options with a list of artists available based on the .txt file. Typing the number corresponding to an artist will in turn, show a list of songs that belong to this artist.

## Filter By Album

When pressing option #3 on the main menu, the app will replace the main menu options with a list of albums available based on the .txt file.

## Filter By Genre

When pressing option #4 on the main menu, the app should show a list of genres. The only four genres available are:

1. Metal
2. Pop
3. Reggaeton
4. Rock

## Search

When pressing option #5 on the main menu, the app should allow me to perform a search based solely on the name of the songs. The app must prompt the user to input some text. Then the app must display a list of songs that matches the whole text or part of the text inside the name of a song.   
Example: The assets folder mentioned in the section “what we provide” has these songs:

* Welcome to the Jungle (Guns and Roses)
* Welcome to the family (Avenged Sevenfold)

If the user types “welcome” it should return both results and any other that matches the text. But if the user types “welcome to the jungle” it should only return the song from Guns and Roses.

## Sample data

Spotifoo has permission from top artists around the world, to share their songs with us. Thus, we created a folder called **assets** with 2 sub folders. One for the music files with a bunch of .mp3 files and another for the album art in .png files.

We also provide a file called data.txt with the information of the songs with information about the song name, artist, album, genre, file name, and album image. Your app must read this file to display the list of songs and other features requested in the mandatory requirements listed below.

**Note:** The folder and text file, contain some invalid data on purpose. Do not try to edit the text inside data.txt or add missing files. Instead, write code to validate the errors.

# Implementation and Design

## Technologies

The following technologies are used in designing and developing the Spotifoo application.

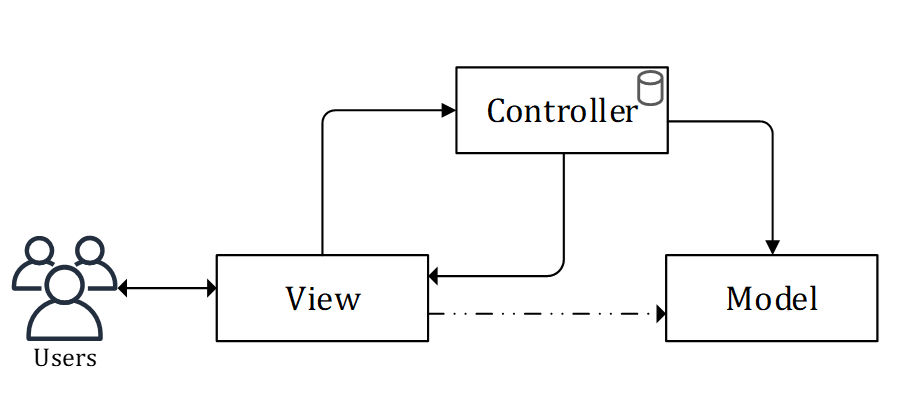
1. Java – Coding language for implementing the Spotifoo application
2. Eclipse - IDE Purpose
3. Visual Paradigm - UML Diagram tool
4. Microsoft word – Document preparation

## Design and Flow diagram

### Architecture and design

Simple MVC architecture shall be followed to designing the Spotifoo application see below. Here Song, Artis, Album and Genre shall be acted as module, the controller shall be followed *Singleton* design patter, it reads the songs data from the file which is provided by the customer and prepare the model object. This functionality shall be done only once and maintained model objects in the form of database cache using collections API.

Next, the view module is mainly responsible for preparing the menus by using controller cache data based on the *User* choice.



### Class diagram

This section shall be explained about the relationship between different classes.

**Model**: All the model classes shall be followed java bean’s structure and implemented Comparableinterface in order maintain the order while caching the data into collections API.

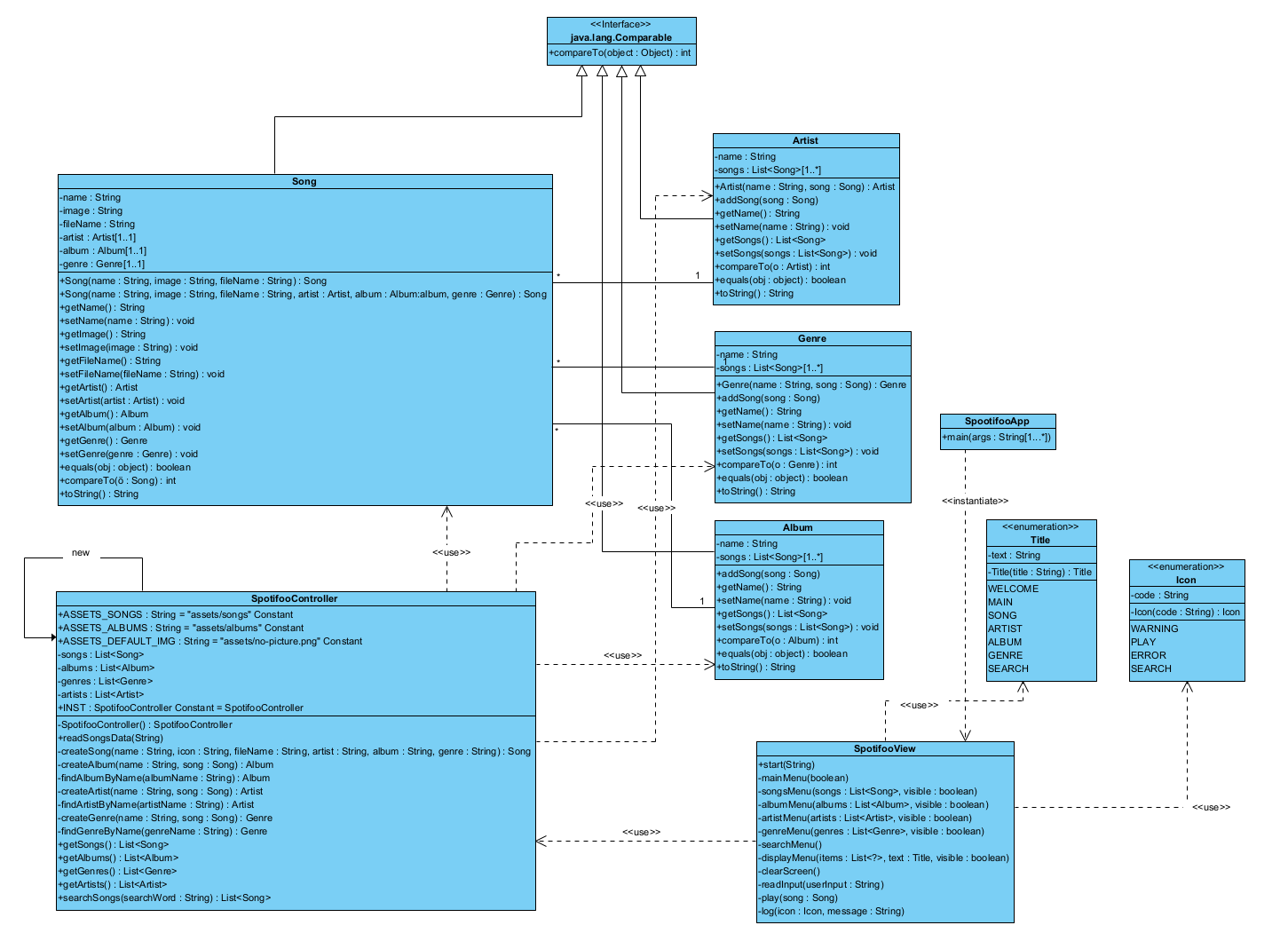
**Controller**: The *SpotifooController.java* is controller class which follows the singleton design patter to maintain the single instance, it reads the customer provided songs data file in fixed format (i.e., comma separated values, see below) and caches the songs data in-memory in structured format for lateral usage.

***Example***: <song\_name>,<artist\_name>,<album\_name>,<genre\_name>,<file\_name>,<image>

*”Andy you are a star,The Killers,Hot fuss,rock,andy-you-are-a-star.mp3,hot-fuss.png”*

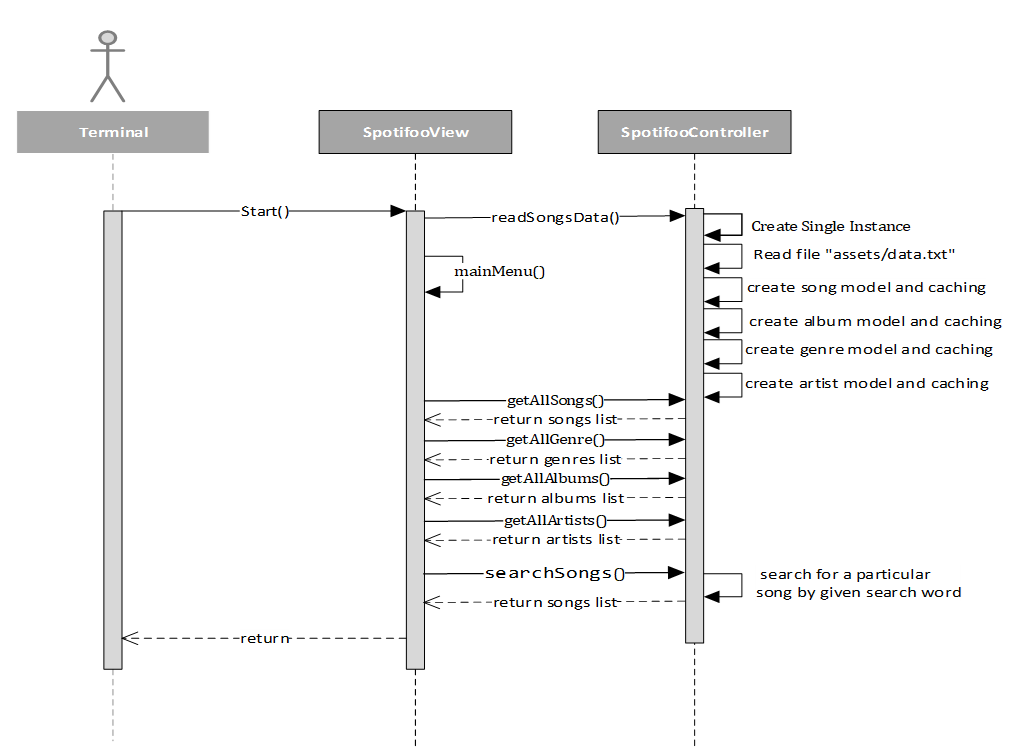
**View:** The *SpotifooView.java* is responsible for building the user interface and displayed in the terminal/console based on the user inputs and performs the validation checks.

1. Title enumeration shall be used for constants purpose and holds the menus title.
2. Icon enumeration shall be holding the *Unicode characters* for icons which shall be shown along with the warning/errors message in the terminal.
3. *SpotifooApp.java* is a main class for the application and it invokes the start method of *SpotifooView* class.



### Sequence diagram

The sequence diagram explains about the logic interaction between the objections in the system and execution flow.



### Model diagram

The below picture explains about the module relationship and the below four models are introduced in Spotifoo application.

1. Songs.java – Responsible for holding the songs metadata like.

|  |  |
| --- | --- |
| **Field Name** | **Description** |
| Name | Name of the song. |
| File name | Song file name with mp3 extension |
| Image | Image name of the song |
| Artist | Artist name of the song |
| Genre | Genre name of the song |
| Album | Album name of the song |

1. Album.java – Maintains the Album metadata like,

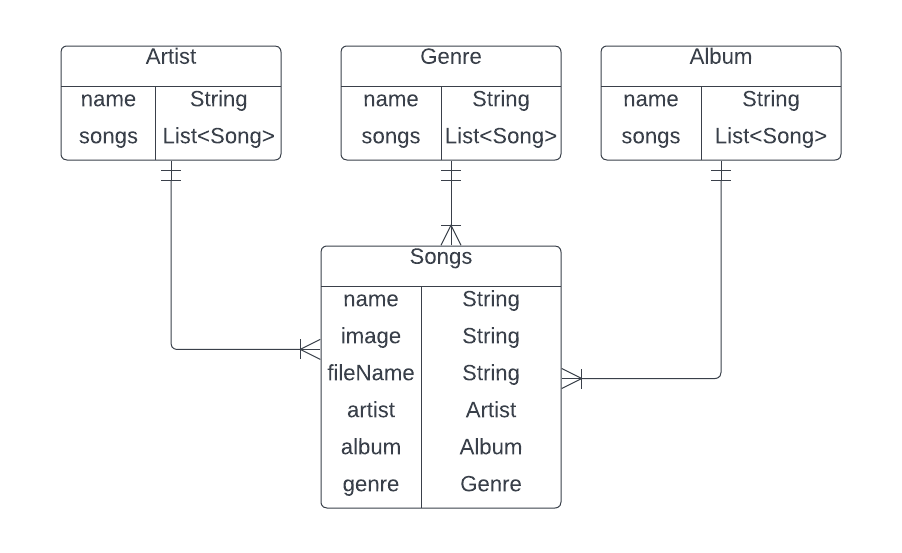
|  |  |
| --- | --- |
| **Field Name** | **Description** |
| Name | Name of the album |
| Songs | List of songs which are belongs to this album |

1. Artist.java – Responsible for holding the song artist metadata.

|  |  |
| --- | --- |
| **Field Name** | **Description** |
| Name | Name of the artist |
| Songs | List of songs which are belongs to this Artist |

1. Genre.java - Responsible for holding the song Genre metadata.

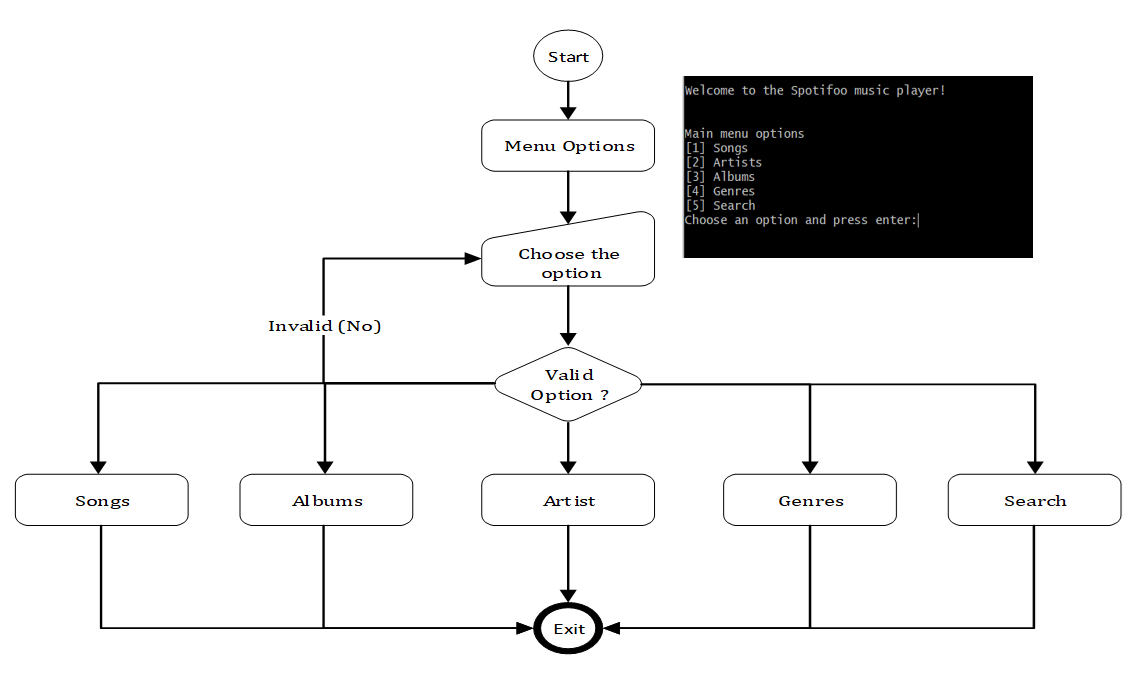
|  |  |
| --- | --- |
| **Field Name** | **Description** |
| Name | Name of the genre |
| Songs | List of songs which are belongs to this Genre |



### Menus flow charts

This section helps to identify the different menus mandatory steps and explains the execution flow.

#### Main

The main menu shall be shown to the user initially, it will provide few options as described in below picture.

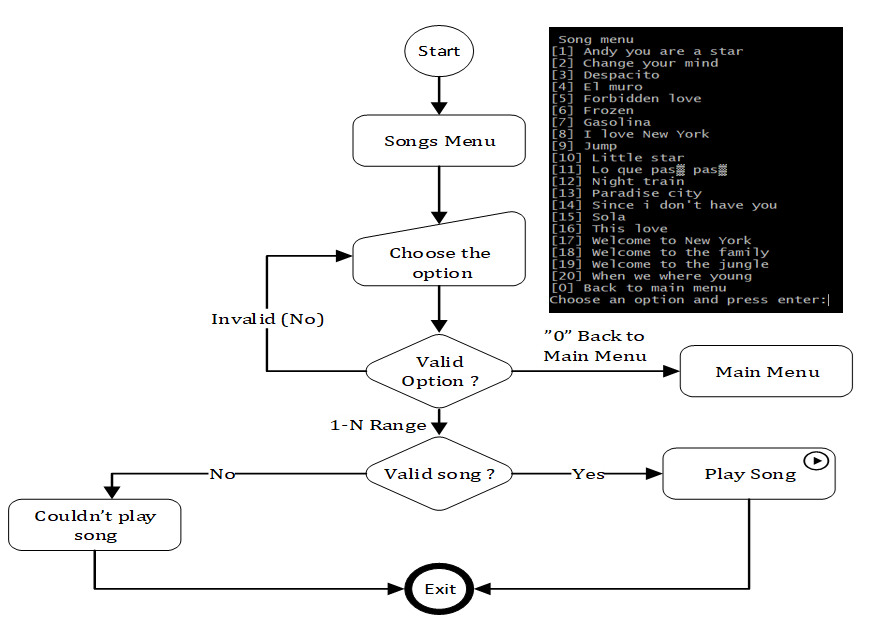
Here user has an option to choose the numbers between 1-5, once user enters the value in specified range that specific functionality shall be executed and moved into the next menu.

**Error**: Any input that entered the user other than 1-5 that shall be treated as a Invalid case and wait for the further user input.

#### Songs

The songs menu shall be shown to the user in different scenarios, i.e., from *Main Menu, Artists, Albums, Genre* and *Search* see below picture.

For example, If user has chosen the option “1” from *Main Menu* then all the songs shall be displayed into songs menu.

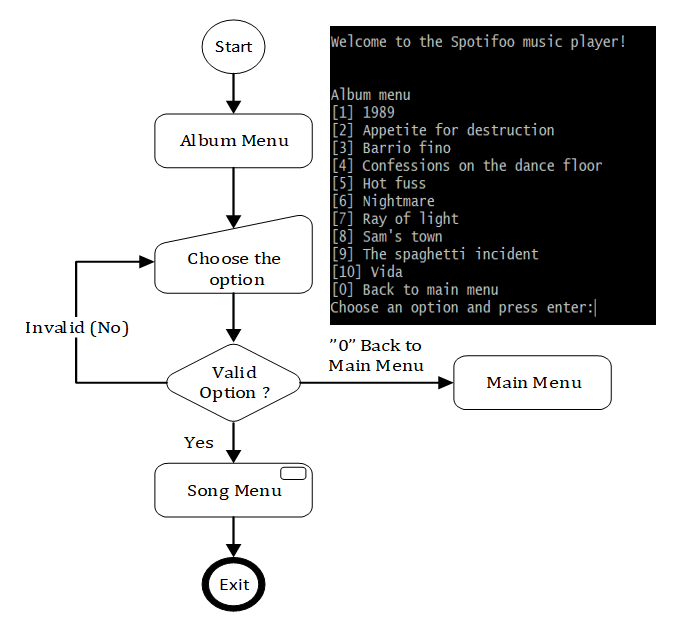


Here user has an option to choose any of the song between [1-N], assume that there is more than 1 song and Option “0” take back to main menu.

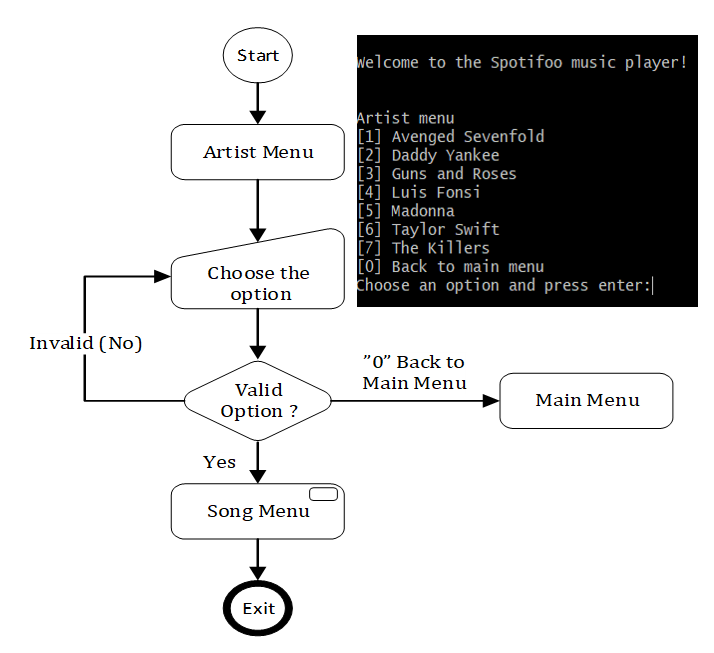
Now, the user selected song shall be played in system default media player and song image is also opened in system default image viewer. Note that if the song is valid and it doesn’t contain any image then default image is opened.

**Error**: Any input that entered the user other than “0” and songs range, that shall be treated as a Invalid case and wait for the further user input.

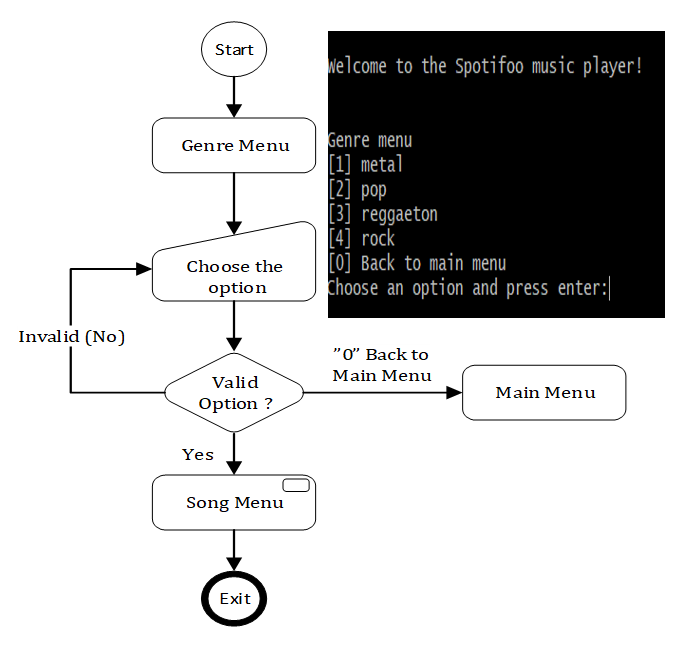
#### Albums



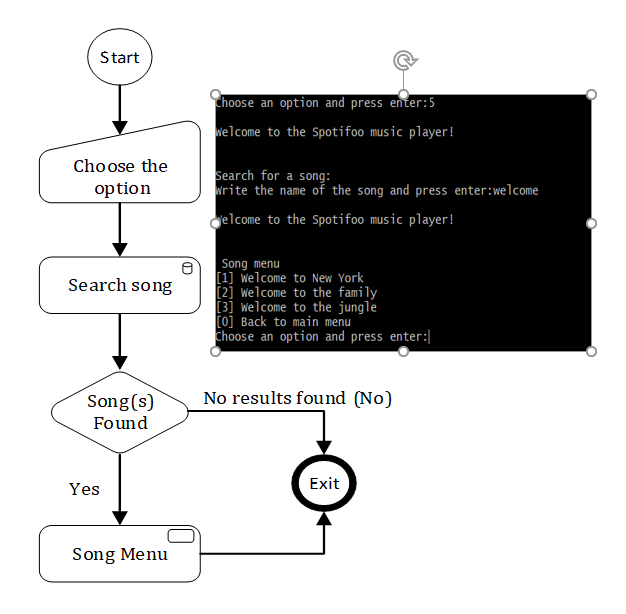
#### Artist



#### Genre

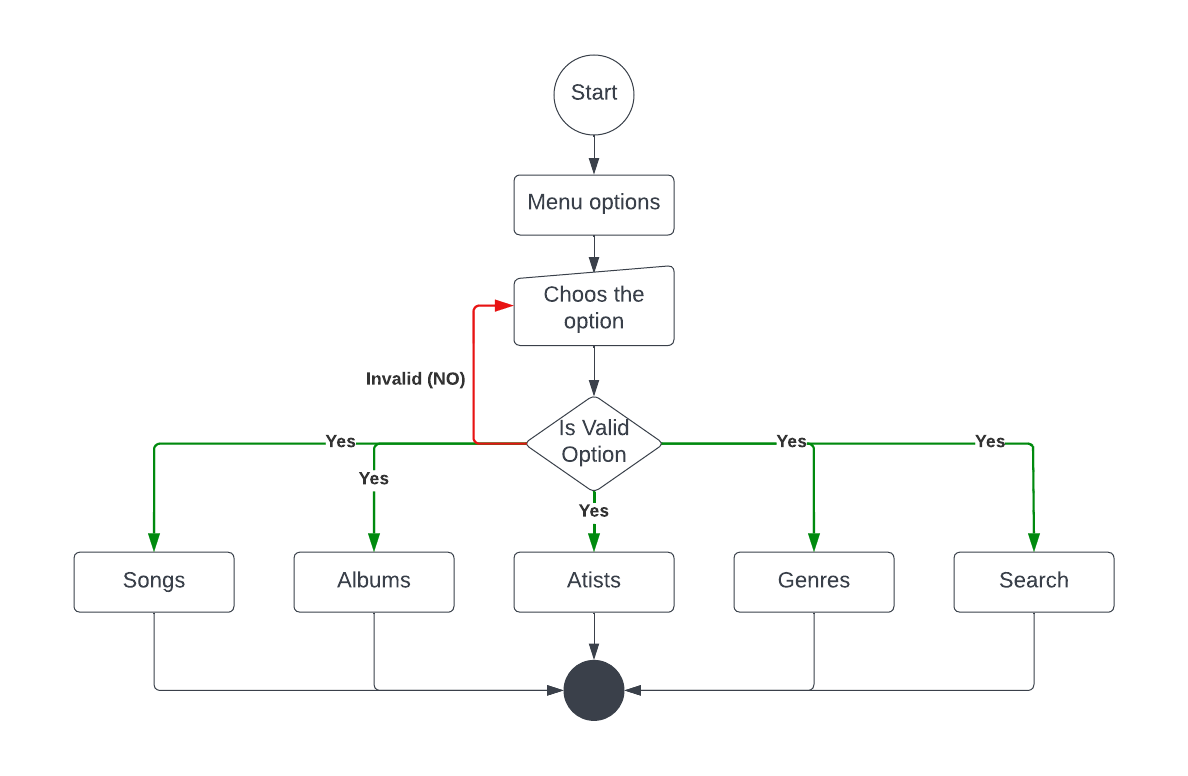


#### Search



### Screens

Navigation:



Models:

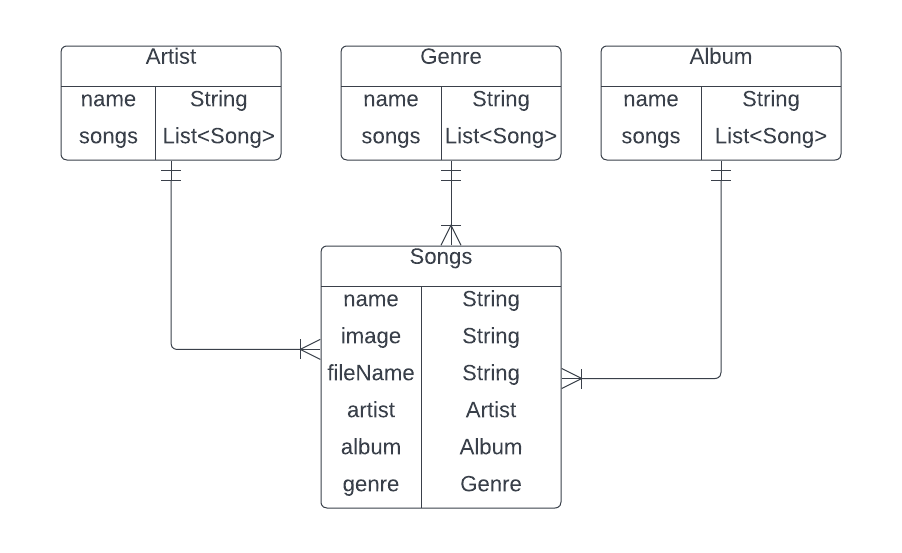
In our Spotifoo App, we have 4 models. They are

Song.java

Artists.java

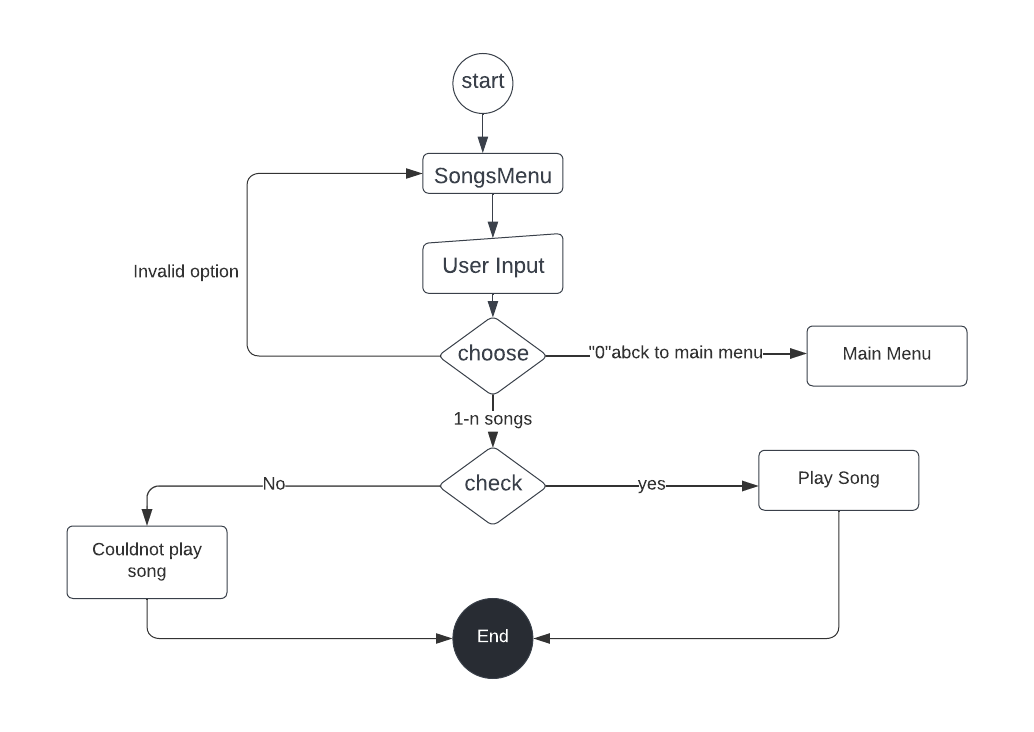
Albums.java

Genres.java



In this I have created models for each of the items in the menu. The navigation picture of each model is shown below.

Song.java

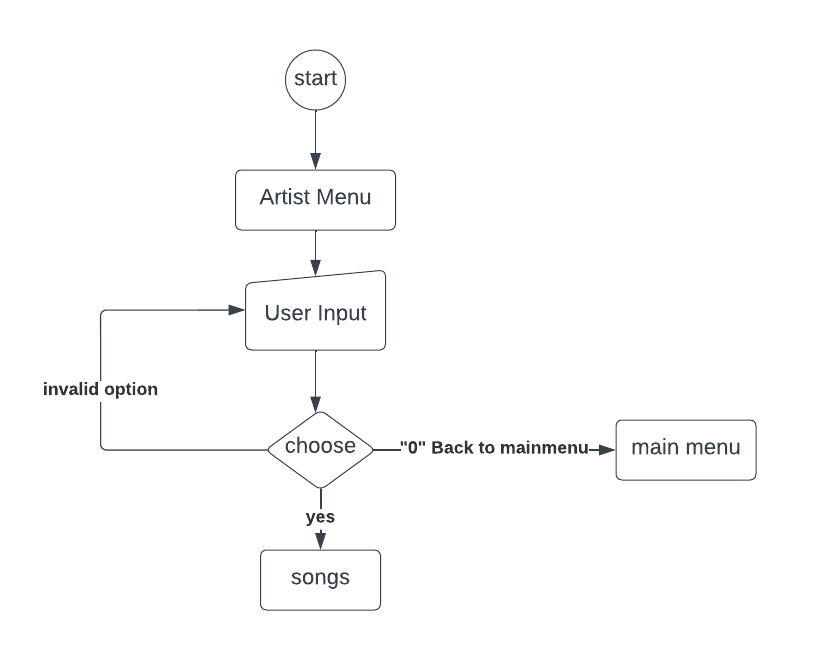


Above is the UML diagram of the model song. Here the user can select any song to his choice. He

enters the option of the song and the song plays. After that, it exits out of the application. Whenever

user clicks invalid option, exception occurs and he is asked to choose another option.

Artists

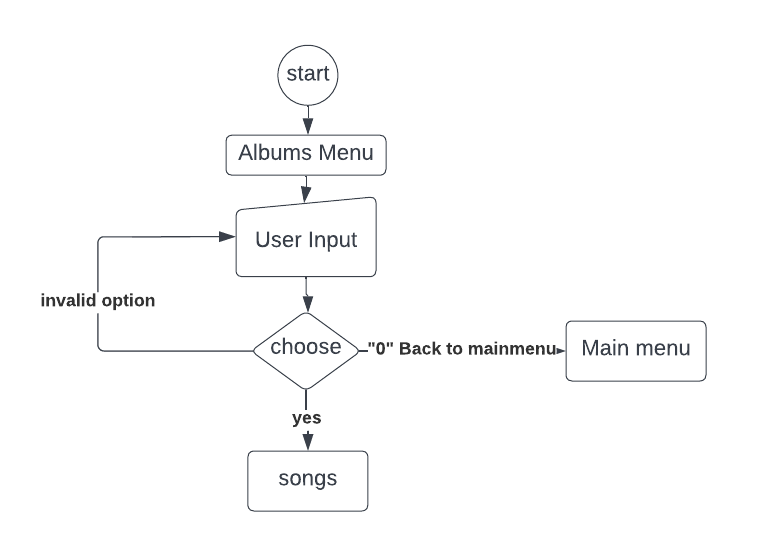


Above is the UML diagram of Artist model. Here the user enters the option and looks at the

Artists that are available. He then selects the required artist and plays the song. If the user

enters invalid number, he gets an exception.

Albums

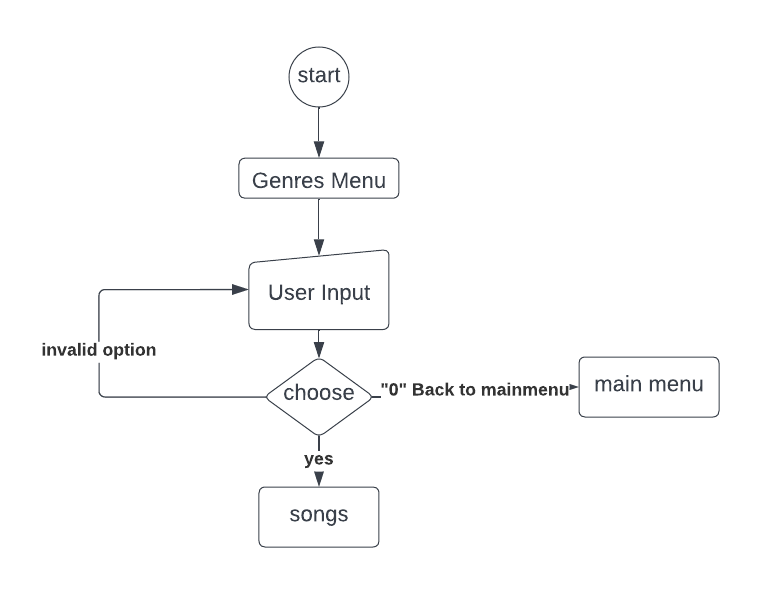


Above is the UML diagram of Albums model. Here the user enters the input and gets all the

Albums that are available in the application. He selects the required option and listens to the songs.

However, he gets an invalid exception, if he enters wrong input.

Genres

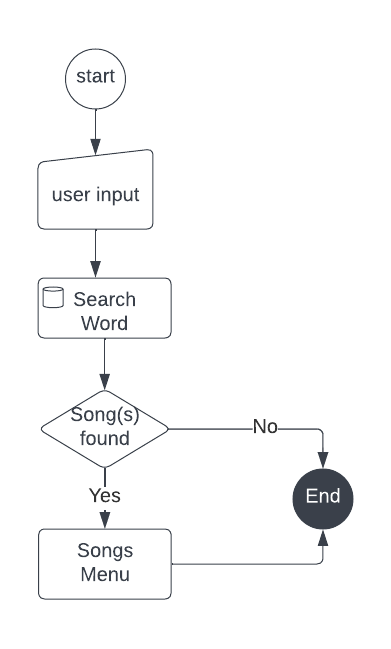


This is the model diagram of the Genres model. Here, the User enters the input and gets the Genres

that are available in the application. He then, selects the requires option and listen to the songs. He

gets an exception, when he enters the wrong option.

Search



Above is the UML diagram for the search option. Here the User can search for a song in the

application. However, if he enters wrong name of the song, he gets exit from the application. When

he enters name of the song, the name of the song that matches the search will be displayed from

albums, artists and genres.