System Requirements Specification

Analysis Documentation

Music Space Project

Team members:

Student number	Student name
14301776	Kevin Corcoran
14538597	Lauren O'Neill
14360281	Bob Tysall
14316846	Michael Coll

Version: v1.0

Date: 15/03/2017

Table of Contents

1. Introduction

1.1. Purpose of the system:

The purpose of the MusicSpace system is to allow registered users to access the music catalogue and stream the content at a premium defined by the company hosting the system.

1.2. Definitions/Glossary:

Playlist: A collection of songs arranged by a user in a particular order, shared with other users, and which can be rated by the users.
Database: A large server, on which is stored all user account information, as well as the entirety of the music library
Daily-Tally: A counter associated with each account responsible for keeping track of the number of songs listened to within a specified 24hr time period.

Top-Up: A system function allowing users to top up their account using a 3rd party api in order to gain access to further library privileges.

1.3. Technology to be used:

3rd Party API for payment system.
Desktop interface and/or web page-based interface.
Database servers, containing user information and a catalogue of music offered by the system.
Web server, from which data is streamed by user and the 3rd Party API is accessed.
API for interfacing with 3rd Party API, database, and web server.

2. Specific Requirements

2.1. Function aspects:

2.1.1. **Main Functions**

Users are entitled to 50 free songs a day before a premium is invoked for each song/album accessed by the user (this requires topping up a balance kept by the system).
Users are required to register to the service before they are entitled to begin accessing music from the database. Upon accessing music, users are entitled to give a rating of all content accessed, which allows the synthesis of popular/unpopular track statistics & analytics.

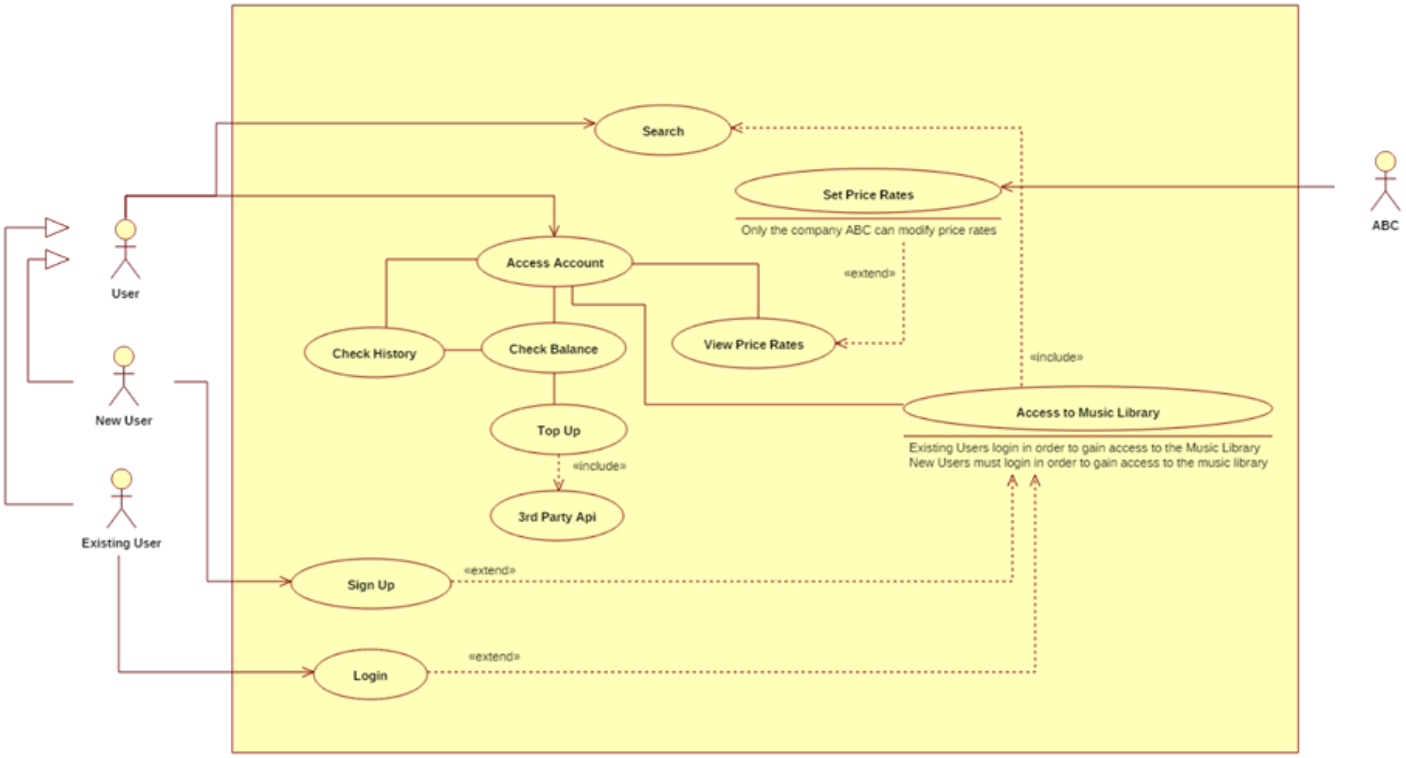
All registered users, in addition to accessing content, are entitled to form and access playlists of their own design, involving music drawn across potentially several artists and albums.

The premium paid for access to the songs & albums by users is determined by the company owning the system (ABC), who reserve the right to adjust the prices as they see fit at any given, with notice given to users of price adjustment upon requested access.

All users are required to complete transactions through a 3rd Party API, wherein users top up their accounts. All transactions made through this API are recorded and stored in the user account information in the database for subsequent queries and data analysis.

A tallying system will be implemented which tracks each user's frequency of access of songs and albums, both to track an individual's content access history and to generate larger-scope access related statistics (e.g. most popular song of week, most accessed download of month).

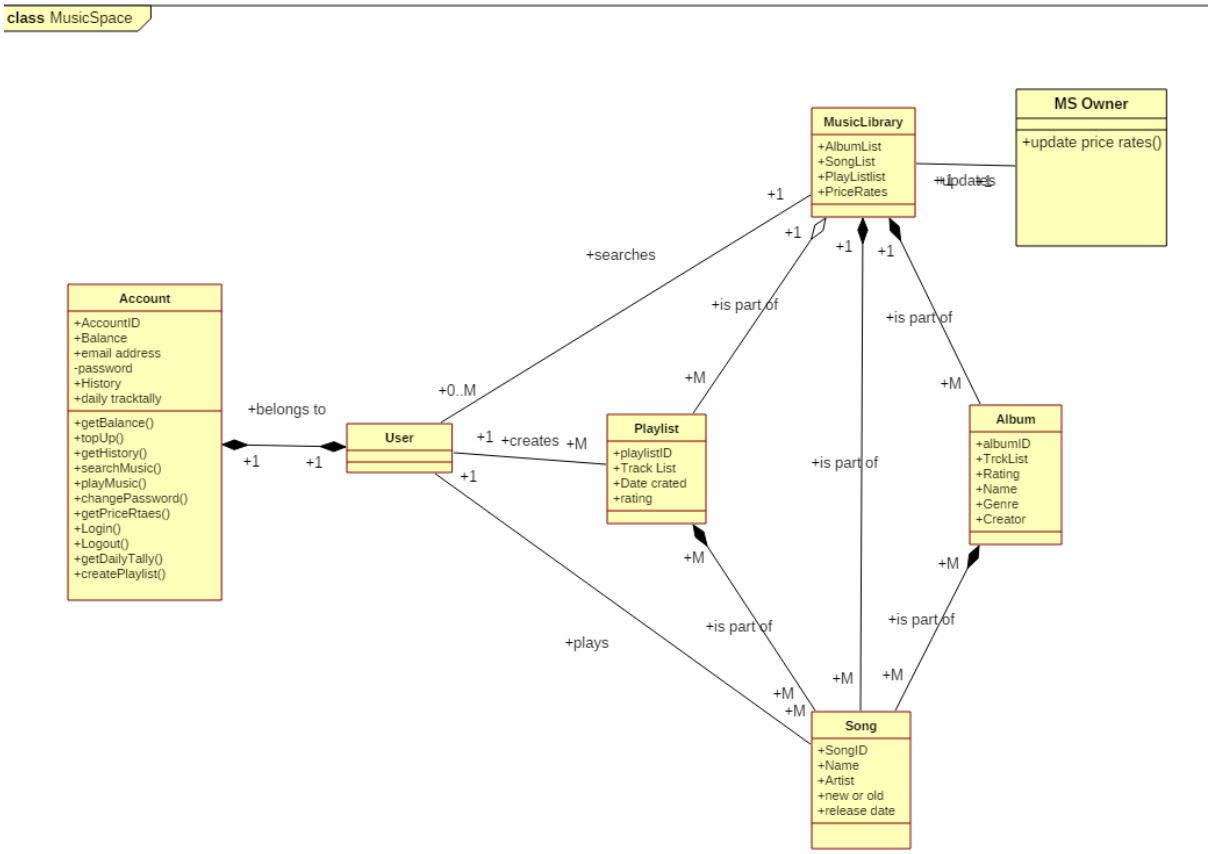
2.1.2. **Use Case Diagram**



The above use case diagram outlines the overall functionality of the system. The registering of new users and logging in of existing users is present. Users can access all account functionality, and ABC can solely set the price rates.

2.2. Informational Aspect

2.2.1. Class Diagram



The above class diagram outlines the informational aspect of the Music Space project. Each user is associated with an account, which holds the users functional information (ID, Balance, login details, transactional history, and daily tally of access) as well as the various functions a user can invoke (for example, getting their balance, topping up, logging in and out, searching the music library).

The user class is also associated with the playlist, music library, and song classes, to ensure that the user can (through the interface) create and access playlists, generally utilise the music library functionality, and listen to the music itself.

Each song is associated with the user (as explained) and the playlist, music library, and album classes (as a part of). The song class ensures that each song has a unique identifier, as well as various necessary metadata.

The playlist class also ensures that each instance has a unique identifier. Playlists are created by users, and have a rating based on aggregate scores from other users. They are comprised of songs, arranged in order by the users creating them. They can (optionally) have a description attached.

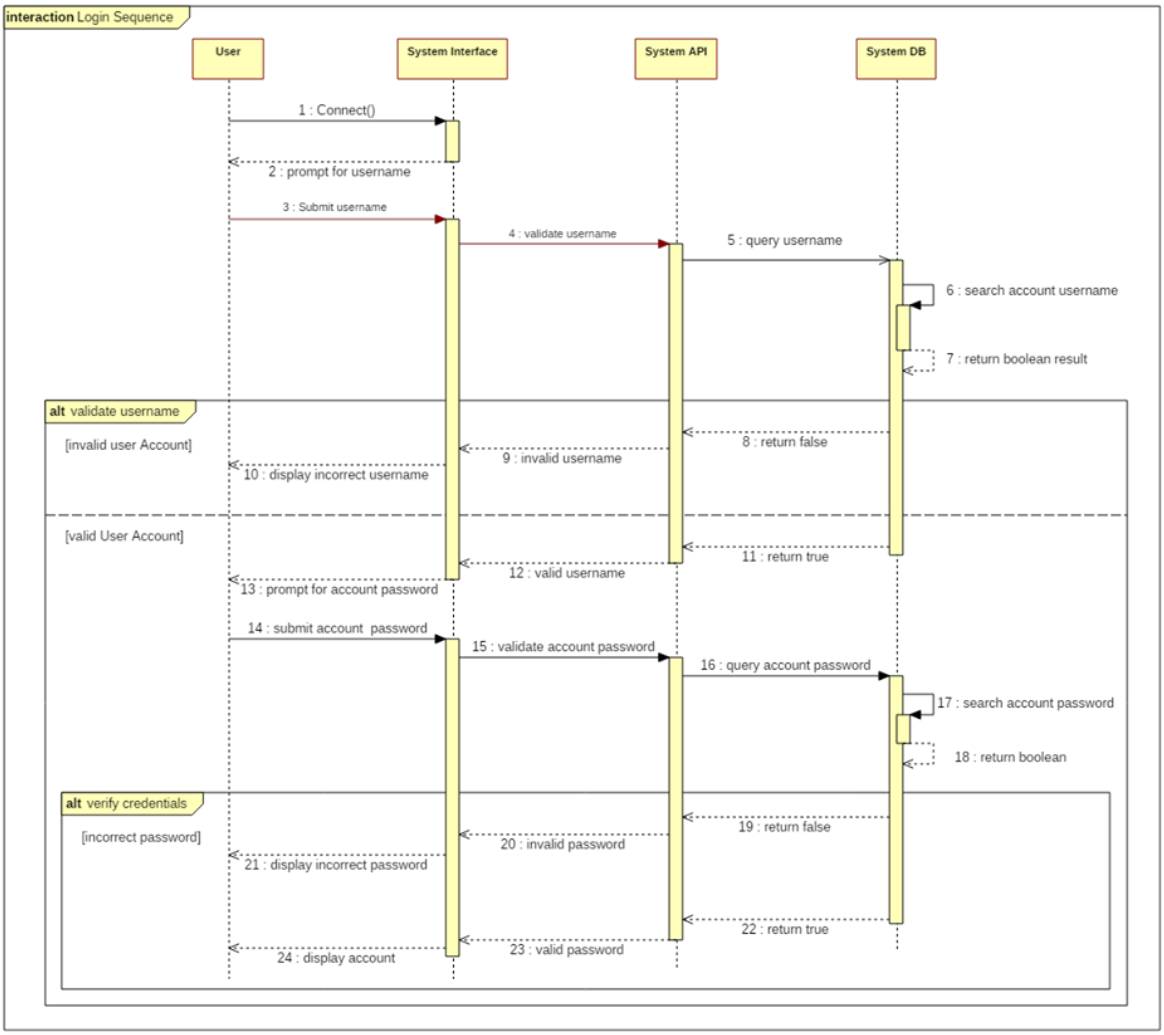
Albums include a unique identifier, and are comprised of songs arranged in specific order as dictated by the artist/publisher providing the albums. A variety of metadata is attached to each album, as is the case with each song.

The Music Library class contains within it references to all songs, albums, and playlists within the system, for access by the User. A list of all songs, albums, and playlists is kept within the Library, and updated with each new addition to the library.

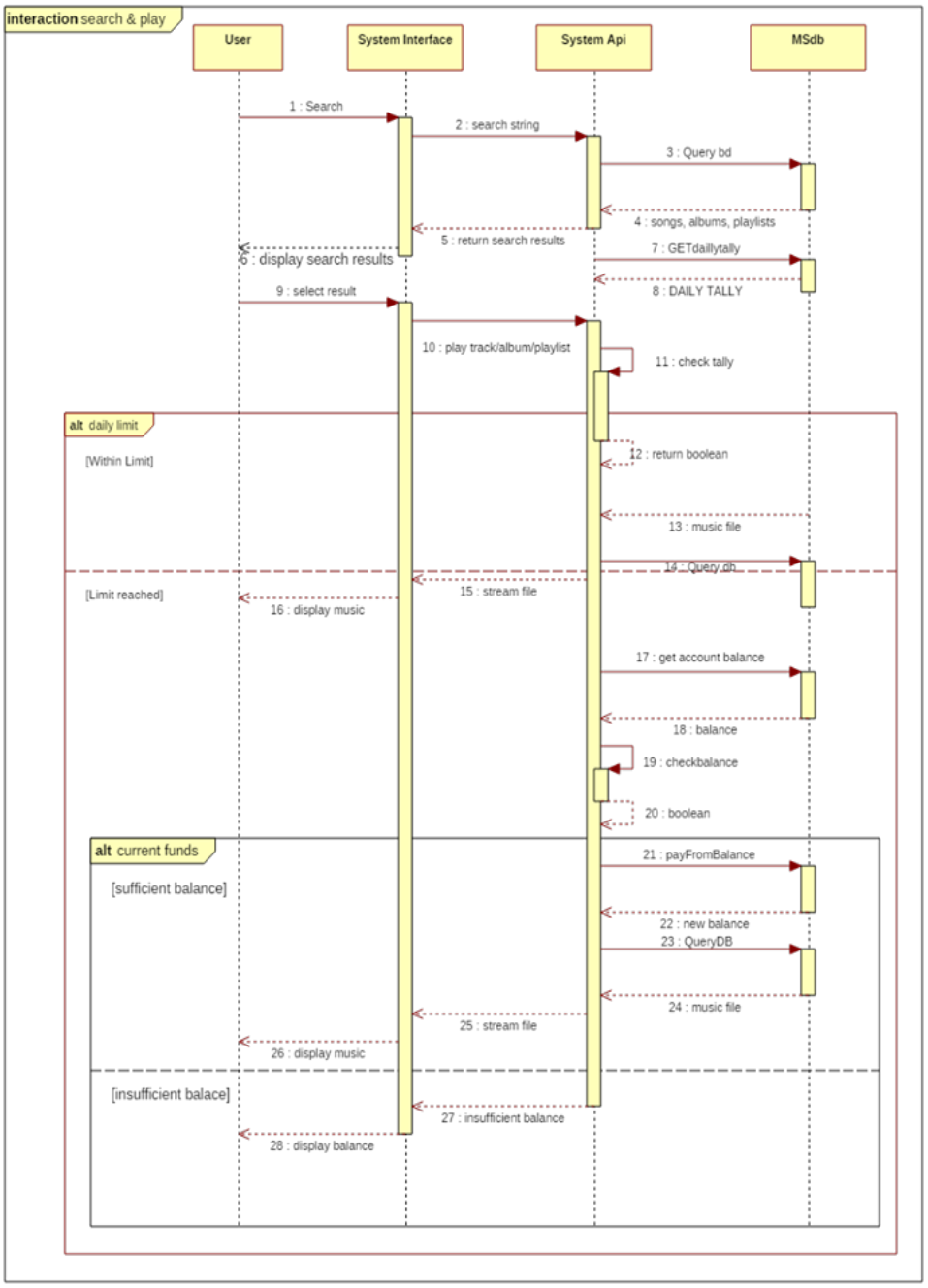
The MS Owner class refers to the owner of the Music Space system, who is entitled to complete access to the Music Library (for administrative purposes), primarily for the purposes of editing the prices outlined by the system (as is their right, at any given time with notice given to users).

2.3. Behavioural Aspect

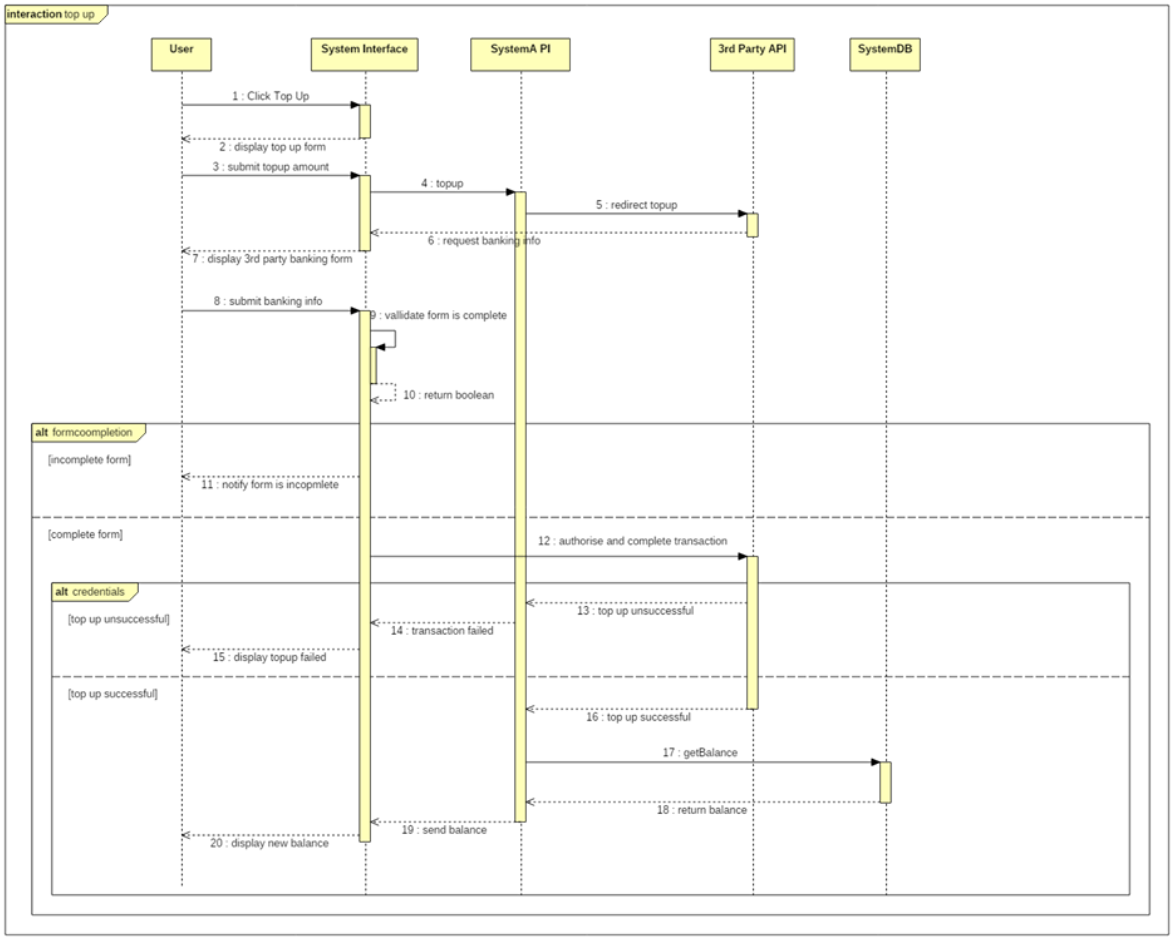
2.3.1. Sequence Diagram



The above sequence diagram outlines the login sequence of the system. It accounts for interaction between the user, system interface, system API, and system database. It brings the user through initial connection, submission of username, validation of username, and subsequent submission of password. The sequence diagram accounts for the situations in which the username is invalid, or the username is valid but an incorrect password is submitted.



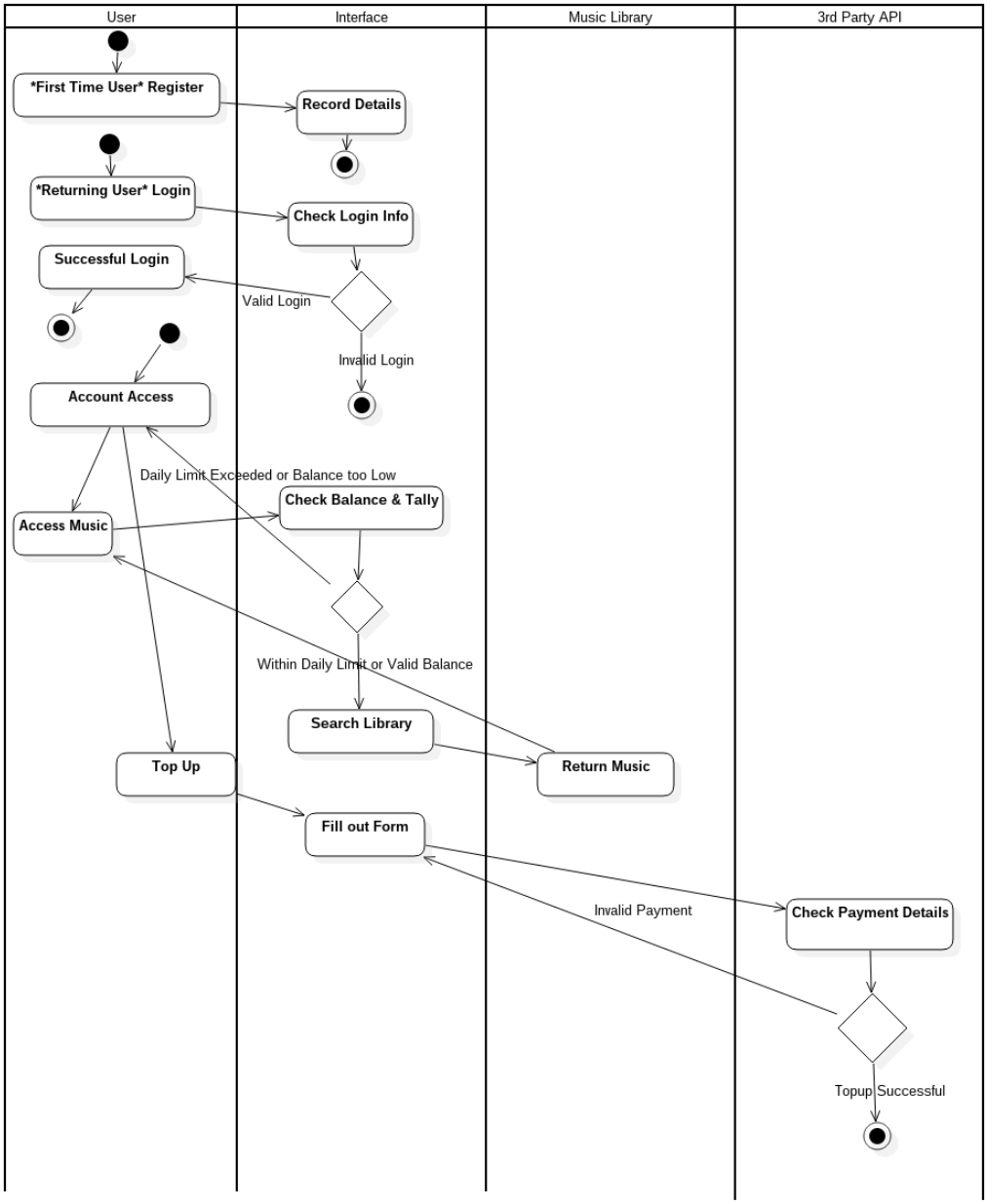
The above sequence diagram outlines the system's handling of searching for and subsequently selecting and playing music. Using the same agents as the login diagram, it steps through the querying of the database by the user, the selection of music by the user, and the system's checking whether or not the user is entitled to access that music; this is achieved by first checking if the user's daily usage tally is below 50, and if so allowing them to access, and if not checking their balance- if their balance is sufficient, access is permitted, otherwise the access request is rejected and the accounts (subpar) balance is displayed.



Our final sequence diagram outlines the top-up procedure. There is an additional agent here, the 3rd party API, responsible for handling monetary transactions. This procedure involves running the user through the filling out of the top-up declaration form, the filling out of the 3rd party banking form (which will be processed by the 3rd party api), and the submission and validation of this banking information. If the forms are filled out incorrectly or only partially completed, they are rejected. If the topup is successful, the user is alerted to this and a new account balance is displayed.

2.3.2. Activity Diagram

In way of an Activity diagram, our team opted to design a Swimlane Diagram:



This diagram outlines which actions within the system are handled and executed by which actors (for example, that initial registration is handled by a relationship between the user and the interface). This diagram outlines all of the basic activity of the system and their actors, including registration (user and interface), login(user and interface), music access (user, interface, and music library/database), and topup (user, interface, and 3rd party API).