# **Software Requirements**

Version 1.0, last updated by Jingze Yang at 2020-08-27

# **Software Requirements Specification (SRS)**

Revision History:

|  |  |  |
| --- | --- | --- |
| Date | Author | Description |
| 2020-08-25 | Jingze Yang | RA\_1.0 |
| 2020-08-27 | Jingze Yang | Add Using Cases |
| 2020-09-08 | Jingze Yang | Revise |

**Catalogues**

[Software Requirements 1](#_Toc7595)

[Software Requirements Specification (SRS) 1](#_Toc12859)

[1. Introduction 3](#_Toc28770)

[1.1    Intended Audience and Purpose 3](#_Toc1432)

[1.2    How to use the document 3](#_Toc19813)

[2. Concept of Operations 4](#_Toc15457)

[2.1 System Context 4](#_Toc17974)

[2.2 System capabilities 5](#_Toc26248)

[3. Use Cases 6](#_Toc17811)

[3.1 An overview of the use cases 6](#_Toc15389)

[1. User perspective 6](#_Toc25195)

[2. Admin perspective 7](#_Toc17137)

[3.2 Detailed description of 7 use cases 7](#_Toc4989)

[Case1: Users register/log in/log out/exit 7](#_Toc17964)

[Case2 : Users Choose Labels of Music 10](#_Toc4374)

[Case3 : Users Search More About Song 11](#_Toc22438)

[Case4 : Users Add Song to Personal Center 12](#_Toc23936)

[Case5: The administrator uploads the song information 13](#_Toc4360)

[Case6：Recommend Music after Registration 16](#_Toc29213)

[Case7：Recommend Music after Using for A Period 17](#_Toc15106)

[4. Internal Requirements 18](#_Toc15954)

[4.1 Recommendation algorithm module 18](#_Toc29462)

[4.2 Service module 19](#_Toc20125)

[4.3 Crawler algorithm module 19](#_Toc27844)

[4.4 GUI module 19](#_Toc25998)

[4.5 Database Module: Database by Minimum Debug  20](#_Toc4385)

[5. Quality Requirements 20](#_Toc23944)

[6. Expected Subsets 21](#_Toc6173)

[7. Expected Changes 21](#_Toc8556)

[8. Appendices 21](#_Toc20743)

## **Introduction**

### 1.1    Intended Audience and Purpose

This document is intended to provided information guiding the installation and development process, ensuring that all system requirements are met. The following entities may find the document useful:

**Primary Customer** - This page will detail all of the application requirements as understood by the production team. The customer should be able to determine that their requirements will be correctly reflected in the final product through the information found on this page.

**User** - A prospective user will be able to use this document to identify the main functionailty included in the application. Furthermore, the application will have a set of system requirements before the application can be run. Details regarding these requirements can be found here.

**Development Team** - Details of specific requirements that the final software build must include will be located here. Developers can use this document to ensure the software addresses each of these requirements.

**QA Team** - By developing testing procedures founded in the system requirements, the QA Team can create a comprehensive testing regimen that will guarantee requirements are met.

### 1.2    How to use the document

Table of Contents:  
  
1. Introduction  
2. Concept of Operations - broad description of the purpose of the application

2.1 System Context - details any specific system requirements the application will require to run

2.2 System Capabilities - description in prose of all capabilities available to the user in the address book  
3. Use cases - A detailed look at each functional requirement, describing the application context both before and after an action is taken  
4. Internal Requirements - The requirements of each module and what a module need from other modules  
5. Quality Requirements - Requirements not pertaining to the function of the application will be listed here  
6. Expected Subsets - Expected levels of functionality at checkpoints during development  
7. Expected Changes - Future features and directions the project is expected to take  
8. Appendices - Details aiding the understanding of this document

8.1 Definitions and acronyms - Any technical terms or abbreviations will be spelled out here for ease of use of the document

8.1.1 Definitions - Definitions of technical or unusual terminology

8.1.2 Acronyms and Abreviations - Any abreviated terms will be expanded here

8.2 References - any external references necessary or helpful to understanding this document will be listed here

## **Concept of Operations**

Music recommendation System aims to track user’s favorite labels of music and songs added to user’s personal center to provide user’s with the most suitable songs for them. It has a UI to allow users to experience these functions.For more details on the usage and capabilities of the application read the section,[System Capabilities](https://uocis.assembla.com/spaces/cis422w18-team2/wiki/Software_Requirements).

### **System Context**

**System Requirements:**

Requires a system with a GUI display because all of the operations are performed through a GUI. All the functions for users are achieved through a Web, thus users have to surf on this web.

Windows:

Windows 10 (8u51 and above)

Windows 8.x (Desktop)

Windows 7 SP1

Mac OS X:

Mac OS X 10.8.3+, 10.9+

Mobile Devices:

Apple

Huawei

Galaxy

Xiaomi

Web Console Browser:

Chrome

Firefox

Safari

### **System capabilities**

Music Recommendation System is a software project to track user’s favorite labels of music and songs added to user’s personal center to provide user’s with the most suitable songs for them. Its abbreviation is MRS. It has a UI to allow users to experience these functions. The system will run on a web. MRS has a UI to register for a new account. When a user wants to use the system, he must first register on the website.

After the user complete the registration, the system will guide user to choose some labels of music which he(or she) like. And then, MRS will provide user with a list of song he(or she) may like indicated by his(or her) choice. In the meantime, MRS will provide user with Hit Songs List together with the recommended list.Later, user can click the song to search more about it. And if the user like the song, he(or she) can click the button “ADD” to add the song to his(or her) personal center.

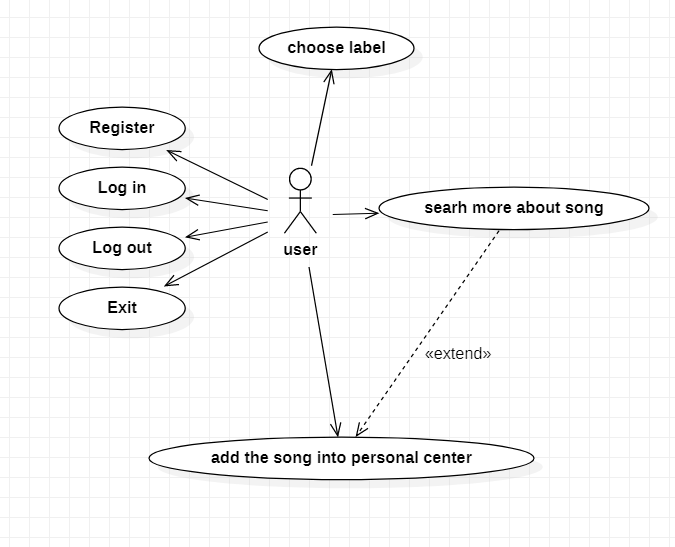
After using MRS for a period, when user logs in MRS, this system will provide user with the most suitable songs according to the songs which was added into personal center.

All functions are guaranteed to run correctly by the system administrator. The system administrator will monitor the data information of various functions in the background. The whole system will be perfect, it has reliable algorithm and excellent background staff. Even if there are errors, they will be solved in the shortest time.

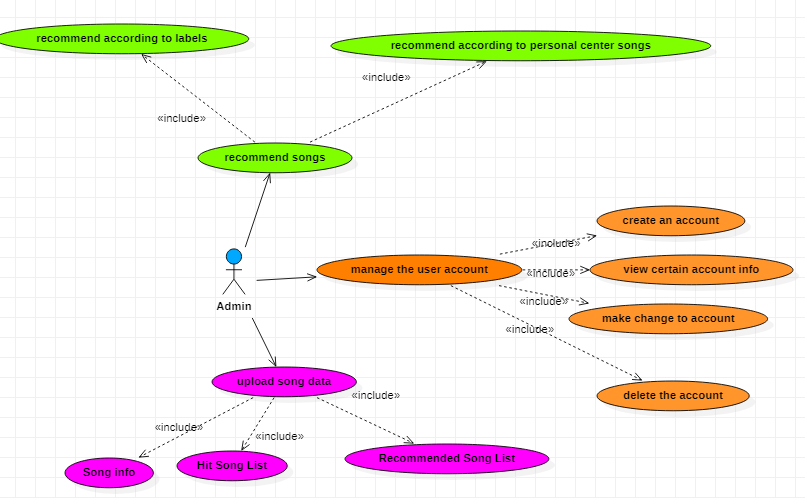
## **Use Cases**

### **An overview of the use cases**

1. **User perspective**



1. **Admin perspective**



### **Detailed description of 7 use cases**

### **Case1: Users register/log in/log out/exit**

|  |  |  |  |
| --- | --- | --- | --- |
| Use Case | Case1: Users register/log in/log out/exit | | |
| Version | 1.0 | Created (date): | 2020/08/27 |
| Author | Jingze Yang | | |
| Source | Users information | | |
| Goals | Users can register/log in /log out/exit MRS . | | |
| Summary | Users can register an account after entering legitimate user information. Log in /log out / exit MRS | | |
| Actors | Users | | |
| Trigger | Users enter the registration interface, fill in the information, click the registration button./Users click log in / log out / exit button. | | |
| Precondition | The user successfully registered an account after entering legitimate user information. | | |
| Frequency | Seldom | | |
| Post conditions | User will register his device to MRS successfully, and MRS saves user’s relevant information if everything goes on well. However, if there’s some other influence stopping the progress, nothing will change./If the user exits the MRS, his information will be deleted from MRS. | | |
| Diagram |  | | |

|  |  |  |
| --- | --- | --- |
| **Basic Flow** | *Actor* | *System* |
| 1. | User clicks on the menu:”REGISTER”. |  |
| 2. | User import relevant information to register her account to MRS. |  |
| 3. |  | MRS affirm there’s no same account name. |
| 4. | User read the agreement message and click “I agree”. |  |
| 5. |  | Display "successfully registered" |
| 1. | User clicks on the menu:”EXIT”. |  |
| 2. |  | Dispaly "Are you sure you want to exit? Registered info will be deleted." |
| 3. | User clicks on the menu:”SURE”. |  |
| 4. |  | Display "Successfully exit" and delete user’s information from MRS. |

|  |  |  |
| --- | --- | --- |
| **Alternative Flow** | *Actor* | *System* |
| 1. | User put in a repetitive account name. | MRS shows “please change a name.” |
| 2. | User disagrees with the agreement message and click ”disagree”. | MRS shows”Failed to register” |
| 3. | User clicks on the menu:”YES”. | The MRS returns to the home page |
| 4. | User clicks on the menu:”NO”. | The MRS keep wait |
| 5. | User quit the MRS compulsorily( eg. turn off the computer or battery off). | Nothing is saved. |

### 

### **Case2 : Users Choose Labels of Music**

|  |  |  |  |
| --- | --- | --- | --- |
| Use Case | Case2:Users are guided to choose music labels according to their performance. | | |
| Version | 1.0 | Created (date): | 2020/08/27 |
| Author | Jingze Yang | | |
| Source | Labels on the website | | |
| Goals | Choosing music labels. | | |
| Summary | Users are guided to choose labels of music once users successfully registered. These choices will be saved by MRS. | | |
| Actors | User | | |
| Trigger | User clicks buttons of labels which they want to choose. | | |
| Precondition | The website is running.  User is logged in.  Song information was imported. | | |
| Frequency | Seldom(once user successfully register) | | |
| Post conditions | Users who successfully registered choose music labels they are interested in. And this result is saved by MRS. | | |
| Diagram |  | | |

|  |  |  |
| --- | --- | --- |
| **Basic Flow** | *Actor* | *System* |
| 1. | User clicks on the buttons of music labels he wants to choose. |  |
| 2 . | User clicks button “Submit”. | MRS save this choice. |

|  |  |  |
| --- | --- | --- |
| **Alternative Flow** | *Actor* | *System* |
| 1. | User choose less than 5 labels. | MSR shows “Please choose more labels.” |
| 2. | User quit the MRS  compulsorily ( eg. turn off the computer or battery off). | Nothing is saved. |

### **Case3 : Users Search More About Song**

|  |  |  |  |
| --- | --- | --- | --- |
| Use Case | Case3:Users search for more information about a song. | | |
| Version | 1.0 | Created (date): | 2020/08/27 |
| Author | Jingze Yang | | |
| Source | Song info in MSR. | | |
| Goals | Search more about a song | | |
| Summary | Users search on the website and click the song to get more information about it. | | |
| Actors | User | | |
| Trigger | User search to learn more about a song. | | |
| Precondition | The website is running.  User is logged in.  Songs’ information was imported. | | |
| Frequency | Often | | |
| Post conditions | Users get to know more information about the song. | | |
| Diagram |  | | |

|  |  |  |
| --- | --- | --- |
| **Basic Flow** | *Actor* | *System* |
| 1. | User searches on the website of the song. |  |
| 2 . | User clicks the song. | The website shows detail information of the song. |

|  |  |  |
| --- | --- | --- |
| **Alternative Flow** | *Actor* | *System* |
| 1. | User quit the MRS  compulsorily ( eg. turn off the computer or battery off). | Nothing is saved. |

### **Case4 : Users Add Song to Personal Center**

|  |  |  |  |
| --- | --- | --- | --- |
| Use Case | Case4:Users add song to personal center. | | |
| Version | 1.0 | Created (date): | 2020/08/27 |
| Author | Jingze Yang | | |
| Source | Song info in MSR. | | |
| Goals | Add song to personal center | | |
| Summary | Users click button “ADD” and add song to personal center. | | |
| Actors | User | | |
| Trigger | User add song to personal center. | | |
| Precondition | The website is running.  User is logged in.  Songs’ information was imported. | | |
| Frequency | Often | | |
| Post conditions | Users add songs into personal center successfully. | | |
| Diagram |  | | |

|  |  |  |
| --- | --- | --- |
| **Basic Flow** | *Actor* | *System* |
| 1. | User click the button “ADD”. |  |
| 2 . |  | MSR add the song to personal center. |

|  |  |  |
| --- | --- | --- |
| **Alternative Flow** | *Actor* | *System* |
| 1. | User quit the MRS  compulsorily ( eg. turn off the computer or battery off). | Nothing is saved. |

### **Case5: The administrator uploads the song information**

|  |  |  |  |
| --- | --- | --- | --- |
| Use Case | Case5: The administrator uploads the song information | | |
| Version | 1.0 | Created (date): | 2020/08/27 |
| Author | Jingze Yang | | |
| Source | The origin of the data in the case. | | |
| Goals | Admin uploads songs’ information from database. | | |
| Summary | The administrator will upload the existing data according to the standard specification to the server of the program for other users to call. | | |
| Actors | Administrator | | |
| Trigger | The administrator upload or update the songs’ information. | | |
| Precondition | The administrator has prepared the information he wants to upload. | | |
| Frequency | Often | | |
| Post conditions | Administrators can download songs’ information data in the database after the completion of uploading by the administrator, and the administrator can update or delete the uploaded data at any time.  If the administrator did not complete the upload before closing the application last time, the upload continues when the application is opened again. | | |
| Diagram |  | | |

|  |  |  |
| --- | --- | --- |
| Basic Flow | Actor | System |
| 1. | The administrator enters the UI interface for uploading data |  |
| 2. | Select the songs’ data to upload locally. |  |
| 3. |  | The songs’ data is saved into the database |
| 4. | Get the response |  |

|  |  |  |
| --- | --- | --- |
| Alternative Flow | Actor | System |
| 1. | The administrator enters the UI interface for uploading data |  |
| 2. | Select the map data to upload locally |  |
| 3. |  | The songs’ data is being saved into the database |
| 4. | Quit the application |  |
| 5. |  | Generate a warning information for the admin |
| 6. | Continue/Stop uploading |  |
| 7. |  | The songs’ data is (not) saved into the database |
| 8. | Get response |  |

|  |  |  |
| --- | --- | --- |
| Alternative Flow | Actor | System |
| 1. | The administrator enters the UI interface for uploading data |  |
| 2. | Select the map data to upload locally |  |
| 3. |  | The upload process fails |
| 4. | Re-upload |  |

### 

### **Case6：Recommend Music after Registration**

|  |  |  |  |
| --- | --- | --- | --- |
| Use Case | Case6：Recommend music after registration | | |
| Version | 1.0 | Created (date): | 2020/08/27 |
| Author | Jingze Yang | | |
| Source | Database | | |
| Goals | MSR recommend music for users after their registration | | |
| Summary | MSR recommend music for users after their registration according to labels users choose. | | |
| Actors | Admin | | |
| Trigger | MSR recommend music | | |
| Precondition | Users has registered and chose their favorite labels. | | |
| Frequency | Often | | |
| Post conditions | MSR provide users with a list of recommended songs according to the labels they have chosen. | | |
| Diagram |  | | |

|  |  |  |
| --- | --- | --- |
| **Basic Flow** | *Actor* | *System* |
| 1. | Choose music labels successfully. |  |
| 2. |  | Access the data from database |
| 3. |  | Show Recommended Song List. |

|  |  |  |
| --- | --- | --- |
| **Alternative Flow** | *Actor* | *System* |
| 1. |  | Fail accessing the data from database |

### **Case7：Recommend Music after Using for A Period**

|  |  |  |  |
| --- | --- | --- | --- |
| Use Case | Case7：Recommend music after using for a period | | |
| Version | 1.0 | Created (date): | 2020/08/27 |
| Author | Jingze Yang | | |
| Source | Database | | |
| Goals | MSR recommend music for users after using for a period | | |
| Summary | MSR recommend music for users after using for a period according to songs added to personal center. | | |
| Actors | Admin | | |
| Trigger | MSR recommend music | | |
| Precondition | Users has used MSR for a time and added songs to their personal center. | | |
| Frequency | Often | | |
| Post conditions | MSR provide users with a list of recommended songs according to the songs added to personal center. | | |
| Diagram |  | | |

|  |  |  |
| --- | --- | --- |
| **Basic Flow** | *Actor* | *System* |
| 1. | Use MSR for a time. |  |
| 2. | Add songs to personal center. | Access the data from database and analyse it. |
| 3. |  | Show Recommended Song List. |

|  |  |  |
| --- | --- | --- |
| **Alternative Flow** | *Actor* | *System* |
| 1. |  | Fail accessing the data from database |

## **Internal Requirements**

### Recommendation algorithm module

The main responsibility of the recommendation algorithm module is to offer users with the music that users may interested in. It can be divided into two parts.

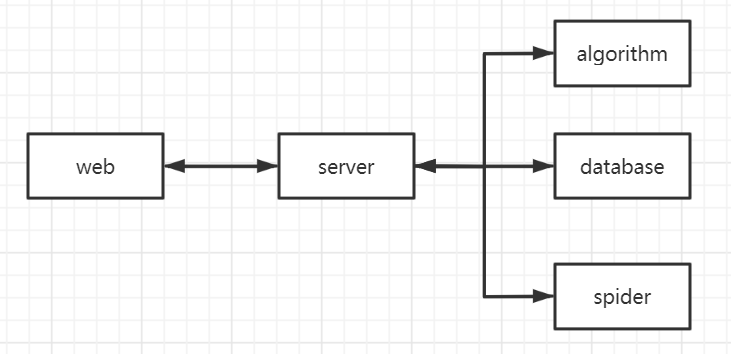
1. To tackle the cold-start problems, the RS will push the music according to user-selected-tags initially. Hence, we need music database and the tags for the period.
2. A CF-based methodology is applied to make recommendation, which needs music logs and music database as input.

As it stated above, the main communication of the module is with the database.

### Service module

The role of service module is to process request from client and return the result to client.

Aim of this module is to provide service on the Linux server except Algorithm and database and return results to the front user.



### Crawler algorithm module

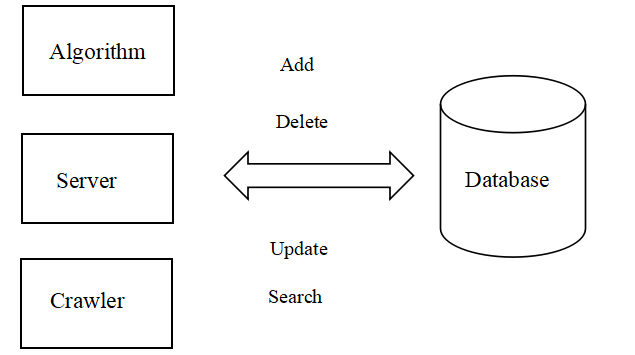
This module aims at scrawling information of music lists and music information in an certain format. This module need database module to provide functions of saving music lists information and music information.

### GUI module

This module is the foundation of Service module. It is the gist of the design of Service module.

### Database Module: Database by Minimum Debug

The key responsibility of the database module is to establish, organize and maintain the data. For other modules, they can add, update, delete and search for the data if necessary. For instance, we will obtain a series of data from the *Crawler Module*, and then we will make the transfer job for the future use of the *Algorithm Module*. The following figure shows the basic relationship between *Database Module* and other modules.



## **Quality Requirements**

1. The number of concurrent operations supported by the system: 100 simultaneous requests.And be sure that connections are closed as fast as possible.
2. System response time: 1 second.
3. Disk space: The server has 500GByte disk space. But no more than 1 Gigabytes are expected for application and bootstrap data.
4. Positioning accuracy: 5 meters

## **Expected Subsets**

1. MSR enable users to register and log in on the website.
2. MSR allow users to search for more information about songs.
3. MSR allow users to add songs to their personal center.
4. MSR provide users with Hit Song List.
5. MSR recommends users the most suitable songs for them according to labels which user has chosen after registration.
6. MSR recommends users the most suitable songs for them according to the songs which have been added into personal center by users.

## **Expected Changes**

1. GUI will be more friendly to users and add more functions.
2. Allow system to translate song lists from other music app to MSR.

## **Appendices**

**MRS: Music Recommendation System**