SW ENGINEERING CSC648/848

Project Title

Section 1 Team 3
Cameron Paczek - Team Lead
Dimitri Rodriguez - Back End Lead
Stephen Kelvin Justo - Front End Lead
Simon Wu - Git Master
Nathaniel Ray Duya - Scrum Master
Anthony Joshua Hizon - Engineer

Milestone 2
10/21/21

Milestone/Version	Date
M1V1	September 30, 2021
M2V1	October 21, 2021

1. Data Definitions V2

Name	**Definition**	**Usage**
Registered_User	A user who has created a unique account via spotify, including spotifyID and email. Only Registered_Users have access to app functionalities.	User data is stored for login, spotifyID will be displayed
Admin_User	Admin account for use by developers	For developers to make various changes and test functionality
User Preferences	The user's specific preferences for song related values such as tempo, key, danceability, etc. changes as more song data is input and can be edited by Registered_User.	Will be used in determining users preferred type of songs to suggest
Song	Specific song with values such as tempo, key, danceability, etc.	Each song has a song id along with various values that will be used to recommend songs for the user to interact with

Data Definitions Continued

Data	**Sub-Data**
Registered_User	● _id ● spotifyId ● userEmail ● userAvatar ● userDescription(255) ● userPreferences
User Preferences	● preferredDanceability ● prefferedKey ● preferredGenre ● preferredTempo ● preferredLoudness ● preferredLiveness ● preferredEnergy
Song	● _id ● danceability ● key ● genre ● tempo ● loudness ● liveness ● energy

2. Functional Requirements V2

User Log-In Functions

1) A user shall login using a Spotify account via Spotify Authorization (1)
2) A registered user shall logout from an account (1)

User Main In-App Functions

3) A registered user shall be presented with a set of optional questions to determine their musical interest (1)
    a) On first creation of an account, a registered user should be given a quick survey.
    b) The survey will be used in an algorithm to help identify their music interests.
4) A registered user shall be presented with a list of song suggestions based on their musical interest, or an arbitrary list of song suggestions if their musical interest is unknown (1)
    a) An algorithm will be used to give users a variety of songs that are catered to them.
5) A registered user's musical interest shall change based on their chosen songs (1)
    a) An algorithm will be used to change the type of music that will be suggested to users.
6) A registered user shall be able to apply filters to suggested songs such as genre, tempo, danceability, etc. (2)
    a) Users can help tune the algorithm to better cater to what they want.
7) A registered user shall be able to play a song (1)
    a) Users can hear the song that is suggested when presented with a suggestion.
8) A registered user shall be able to add a song to a playlist (2)
    a) Users can add a suggestion to a playlist if they choose to.
9) A registered user shall be able to skip to the next song (2)
    a) Users should skip a song suggestion if they choose to.
10) A registered user shall be able to return to the previous song (2)
    a) Users can go to previous songs they skipped if they did so accidentally.
11) A registered user shall be able to pause and play a song (2)
    a) Users can stop a suggestion from playing.
12) A registered user shall be able to share a song (2)
    a) Users can send links to a song that is suggested or in a playlist.
13) A registered user shall be able to replay a song (2)

a) Users can go back to any point of a song.

User Profile Customizability

14) A registered user shall be able to display songs and playlists on their profile (3)

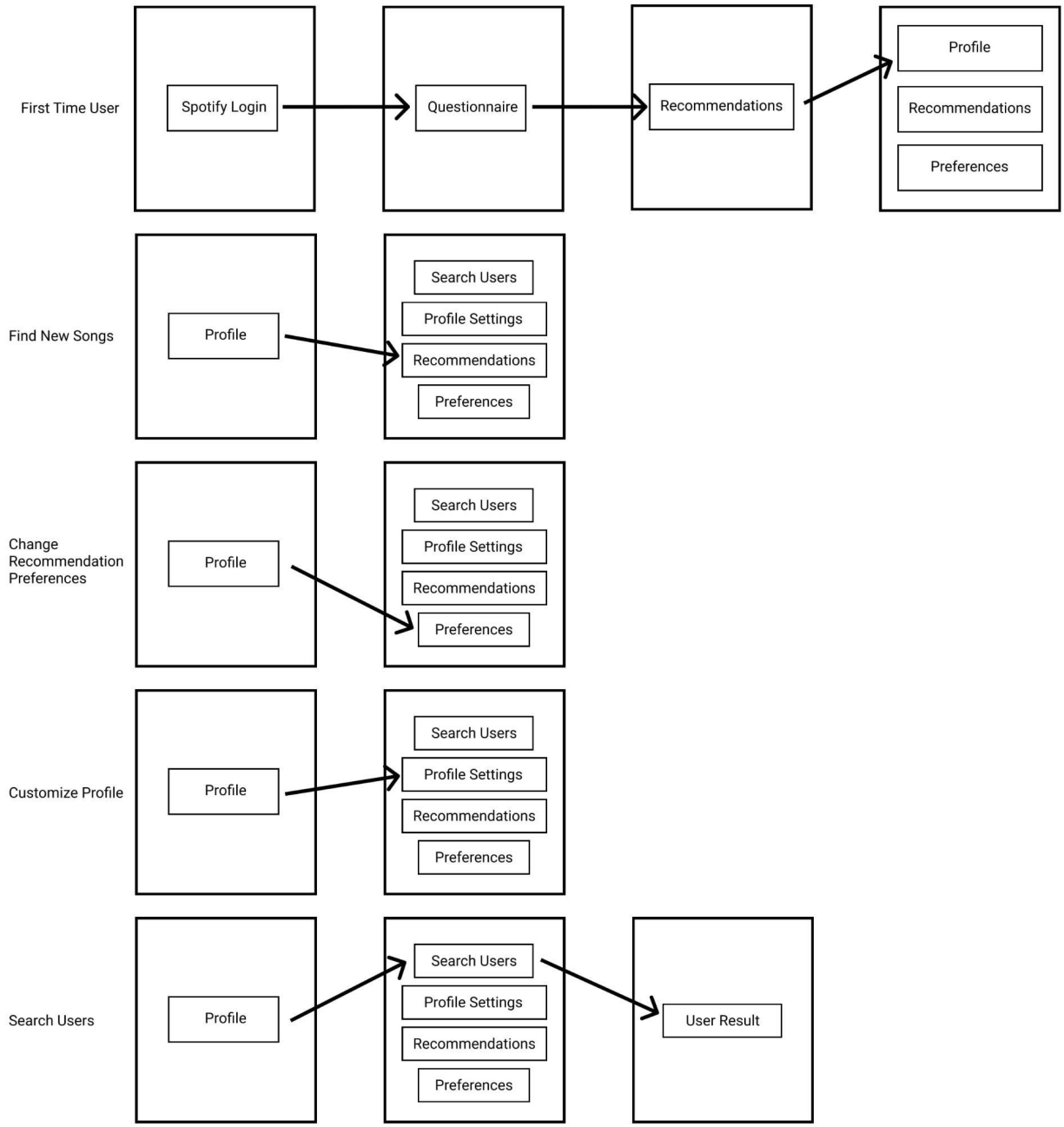
a) Users should be able to see other user's playlists and songs.

b) Users should be able to define who can see playlists and songs.

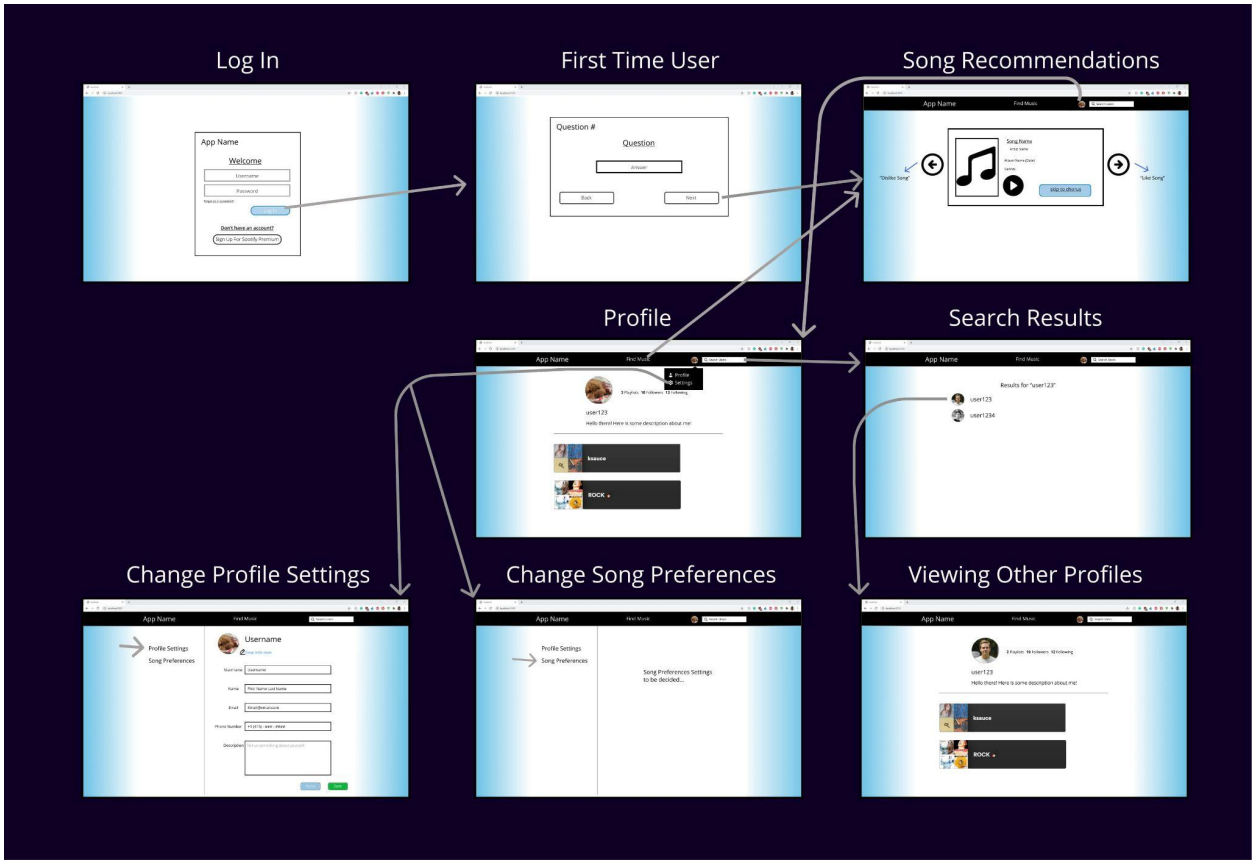
15) A registered user shall be able to create different customizable sections on their profile (2)

a) Users should be able to have multiple music "profiles" for the different music tastes.

3. UI Mockups and Storyboards (high level only)



Picture 1: Gray and White Storyboard



Picture 2: GUI Prototype

UX Principles Summary

Useful:
- Our application will help users find new songs.

Usable:
- Our application recommends new songs to users.

Desirable:
- Our application will be desirable because it allows users to show off their favorite songs on their profiles.
- Our application will be desirable because users can have multiple profiles for their music tastes which allow them to find all types of new music each time they use our application.

Findable:
- Our application will be relatively small so navigating it will be simple.

Credible:
- Our application will be using the Spotify API to generate song recommendations.
- Our application will require users to have Spotify Premium Accounts.

Accessible:
- Our application will not be accessible to people with auditory and vision disabilities.

4. High level Architecture, Database Organization

Users

User
_id
spotifyId
userEmail
userAvatar
userDescription
userPreferences[]

UserPreferences
preferredDanceability
preferredKey
preferredGenre
preferredTempo
preferredLoudness
preferredLiveness
preferredEnergy

Songs

Songs
_id
danceability
key
genre
tempo
loudness
liveness
energy

Add/Delete/Search architecture	*Functional Requirement*
Add/Delete/Search for Users	When a User registers
Search/Display for Users	When Users search for others Users via their SpotifyID or email
Add/Search UserPreferences	When a User likes a song change userPreferences accordingly
Add/Delete/Display UserPreferences	When a User takes new user questionnaire or otherwise configures preferences

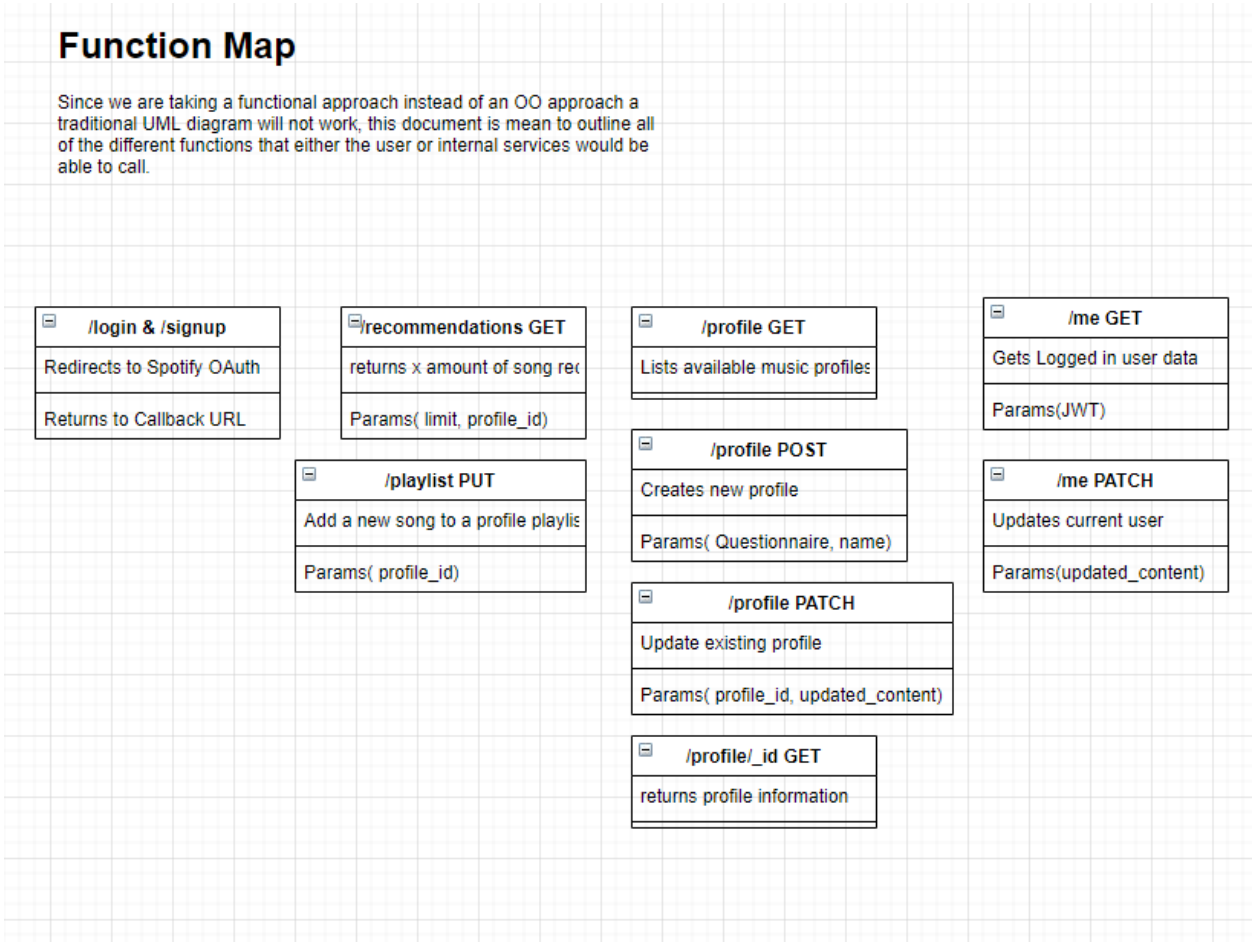
Technical Feasibility

Database operations are built to run smoothly and easily, because we aren't storing much data on our end and are getting a bulk of our data from the third party spotify API there isn't too much concern as to the feasibility of our various database functions. For example, searching for another registered user we simply find the user with the unique spotifyID put into the search field. There isn't much by way of complications.

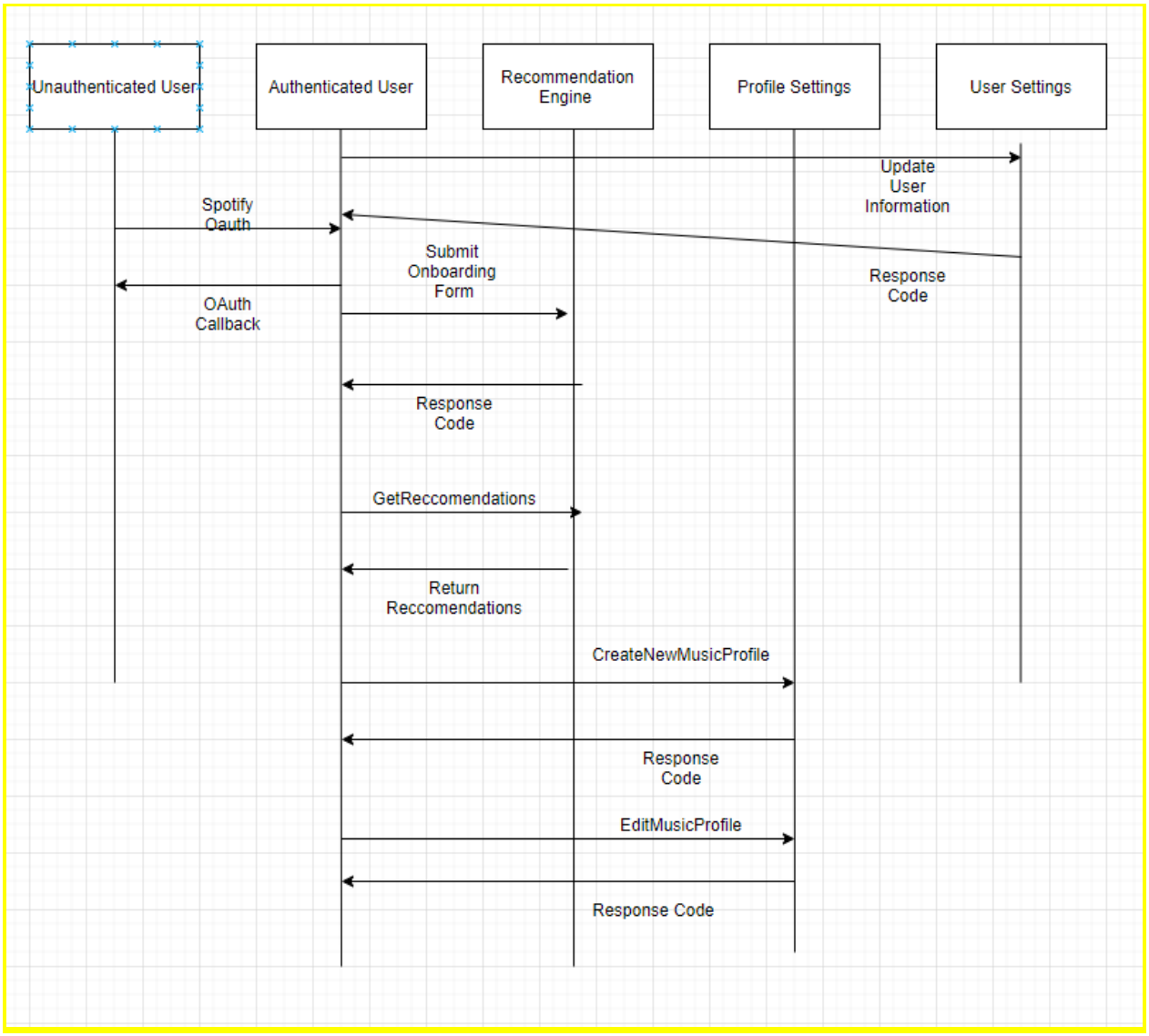
API

We are using Spotify's API, it will allow us to make get requests for search parameters like songs, artists, albums and more. We will be able to make post requests including things like playlist creation, adding to existing playlists, adding songs to users spotify libraries, etc. It will also allow us to make get requests for song values we need for creating our UserPreference like a given song's genre, tempo, danceability and more. The Spotify API is versatile and grants us access to a swath of user information as well as a variety of different functions that are integral to our applications functionality.

5. High Level UML Diagrams



Sequence Diagram



6 .Identify ***actual*** key risks for your project at this time

- **Skill risks and mitigation plan**
    - Our team changed our frontend technology from Reactjs and Nextjs to Vuejs and Nuxtjs. Most team members are unfamiliar with Vuejs so those team members will need to learn it.
        - Since we have not officially begun programming, applying this change will not be difficult nor will it negatively impact any work that we will do in the future.
        - Vuejs is similar to Reactjs so learning it will not be difficult.
            - One of our team members is very experienced with Vuejs so other team members may be able to refer to him for help.
- **Schedule risks**
    - Every team member has a busy schedule so it is very difficult for everyone to be present during team meetings.
        - Our plan to mitigate this risk is to write all the important meeting notes and questions we have into our Discord Server so that absentees are able to catch up on our meetings.
        - We also use Github's kanban board in order to keep track of everyone's work as well as future features that we will need to implement.
- **Teamwork risks**
    - To prevent team members from working on the same functionality and introducing merging conflicts:
        - Everyone will be committing work to their own branches.
            - Two team members will be in charge of pull requests and merging branches into the main branch.
        - Everyone will be assigned a feature via Github's kanban board.

7. Project management

Our back-end and front-end leads chose two team members to help them complete their Milestone 2 tasks. This means that half of our team members were in charge of creating the UI Storyboards and mockups while the other half were in charge of completing the UML diagrams and Database Organization.

We held our outside class meetings every Tuesday to discuss our progress on Milestone 2 and the front-end team shared screenshots of their Storyboard/UI designs on our Discord server.