Milestone 4 - Application Design & Architecture, Database Design

Team Information

Team Number: 024_2

Team Name: PlaySyncers

Team Members:

- Gaukas Wang
- Logan O’Brien
- Kaitlin McConnell
- Erica Lowrey

List of Features (Revised, Ordered by developing priority)

- Landing Page (Completed)
    - Basic HTML

- User Management System (Completed)
    - User Sign-up/Registration (HTML, JS, Flask, PSQL)
    - User Log-in (HTML, JS, Flask, PSQL)
    - User Cookie verification (Flask, PSQL)

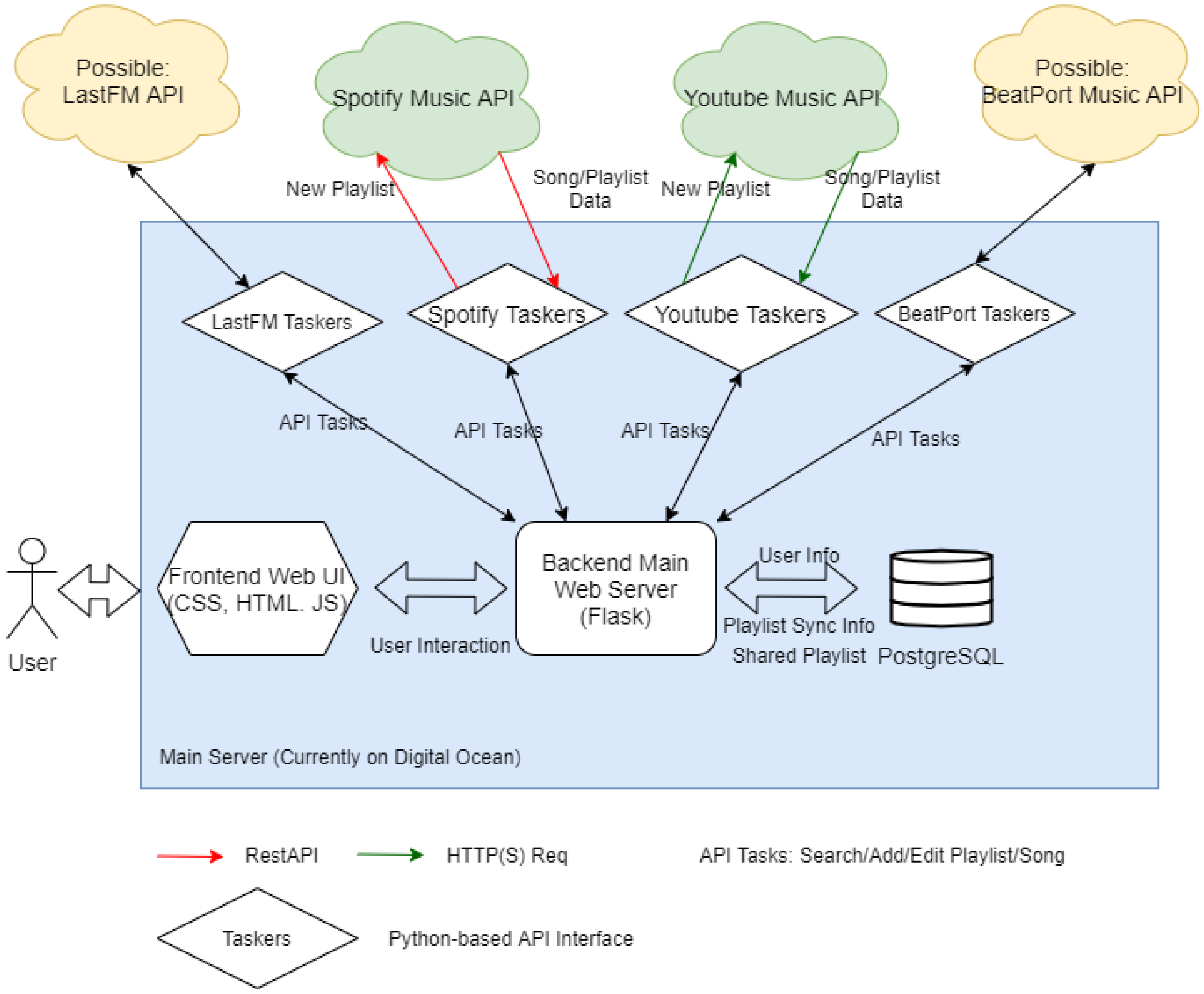
- Authorized platform manager (In Progress)
    - User voluntary 3rd party platform connection, including YouTube Music, Spotify, etc (HTML, JS, Flask, PSQL)
    - Authorization Interface for backend taskers (Flask, PSQL)

- Playlist Scanner (In Progress)
    - Playlist Scanning Taskers (Flask, Python)
    - User interaction for selected playlists (HTML, JS, Flask)

- Playlist Constructor (In Progress)
    - Playlist Constructing Taskers (Flask, Python)
    - User interaction for target playlists (HTML, JS, Flask)

- Playlist Monitor/Synchronizer
    - The software will monitor the user’s playlists (either continuously or at specified intervals). Any changes made to one playlist will be reflected on another (each playlist will be specified by the user).

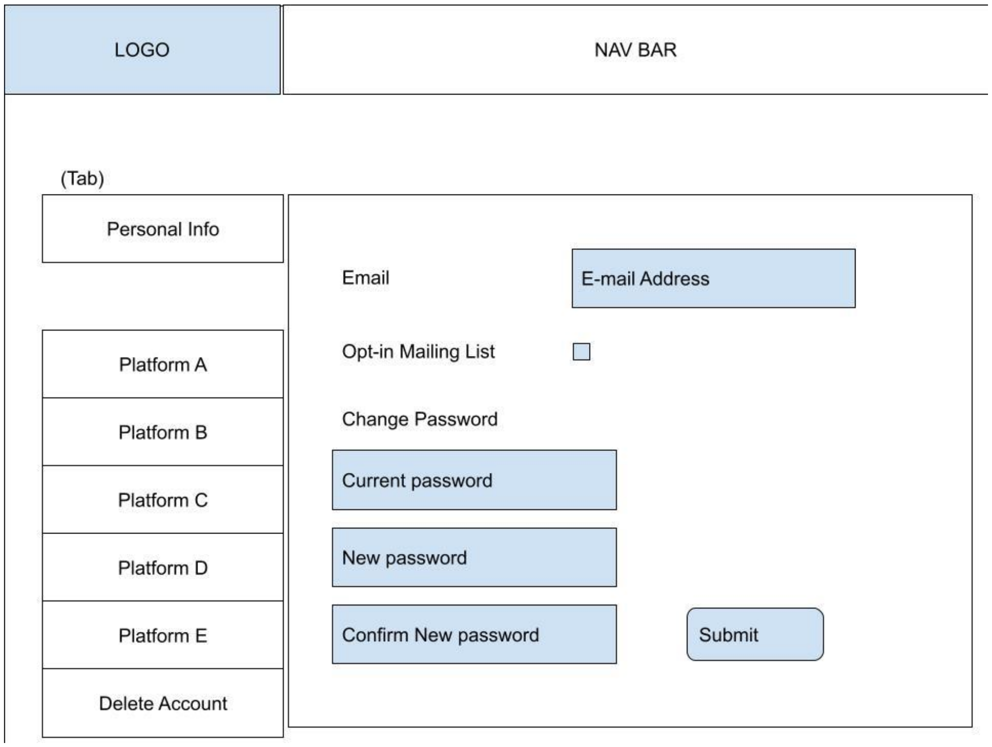
Architecture Diagram



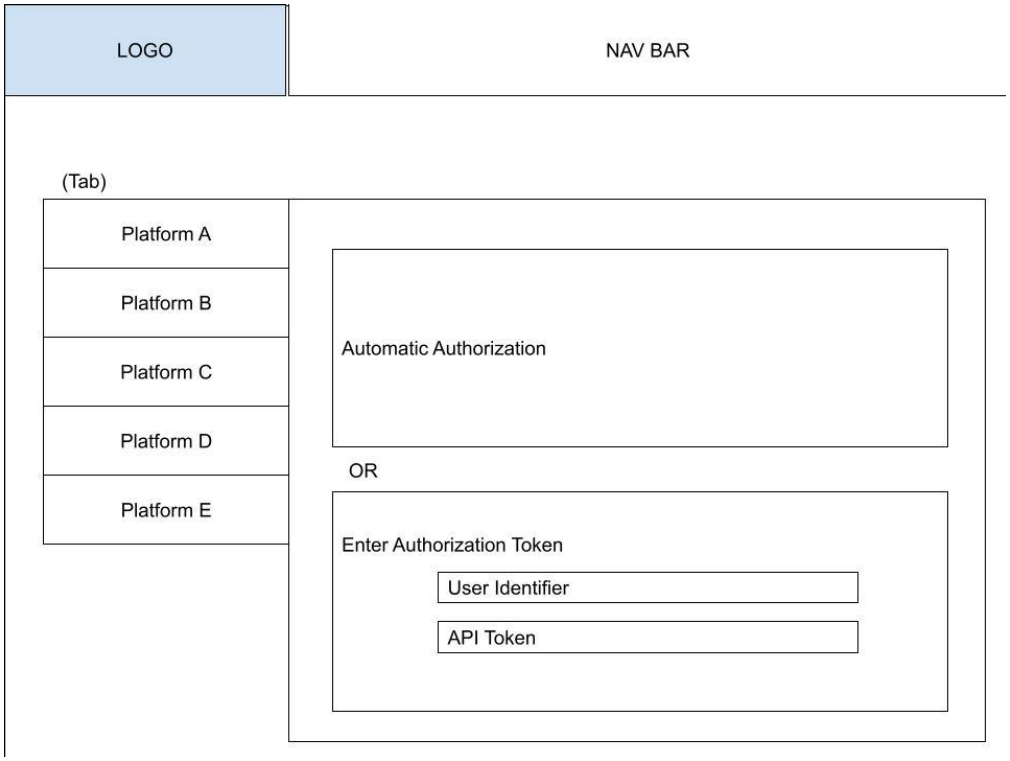
Front End Design

- Landing Page: https://playsync.me/
- About Page: https://playsync.me/about
- Login Page: https://playsync.me/login
- Sign-Up Page: https://playsync.me/signup

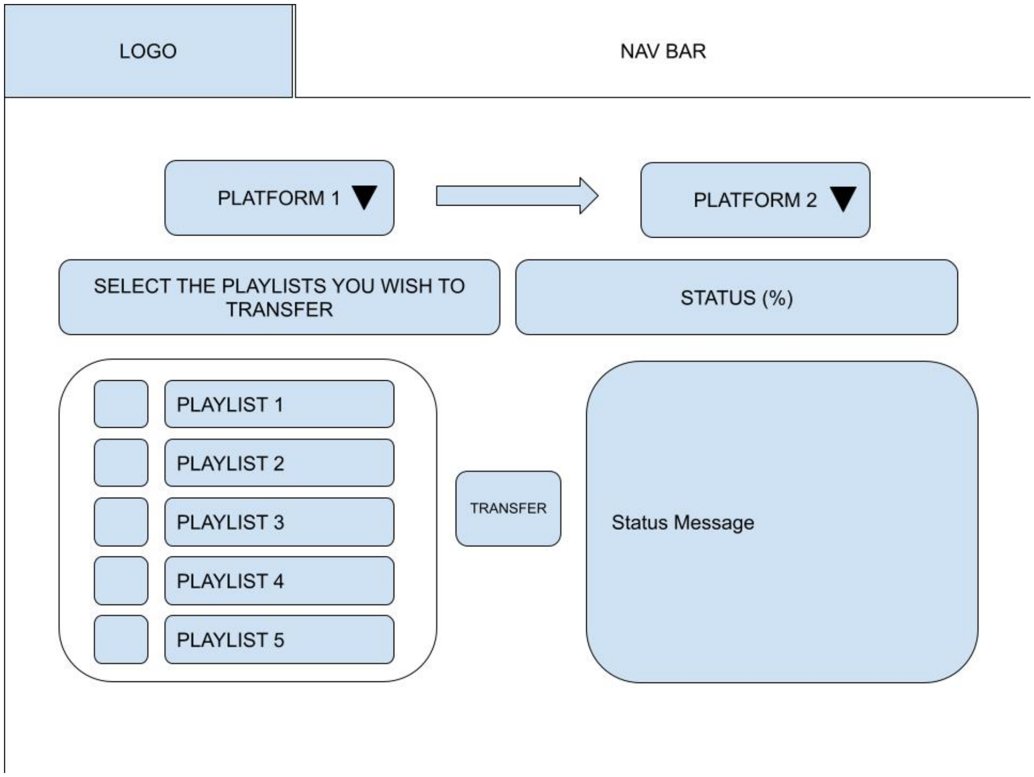
- Profile Page (In Progress): https://freesync.studio/profile_page.html



- Auth Page (integration with profile page/In progress)



- Transfer Page (In Progress):



Web Service Design

- User Login: Despite the local database supported traditional user login, we would like to implement OAuth-based 3rd-party login to support major music streaming platforms which will require us to use multiple external User Login APIs.

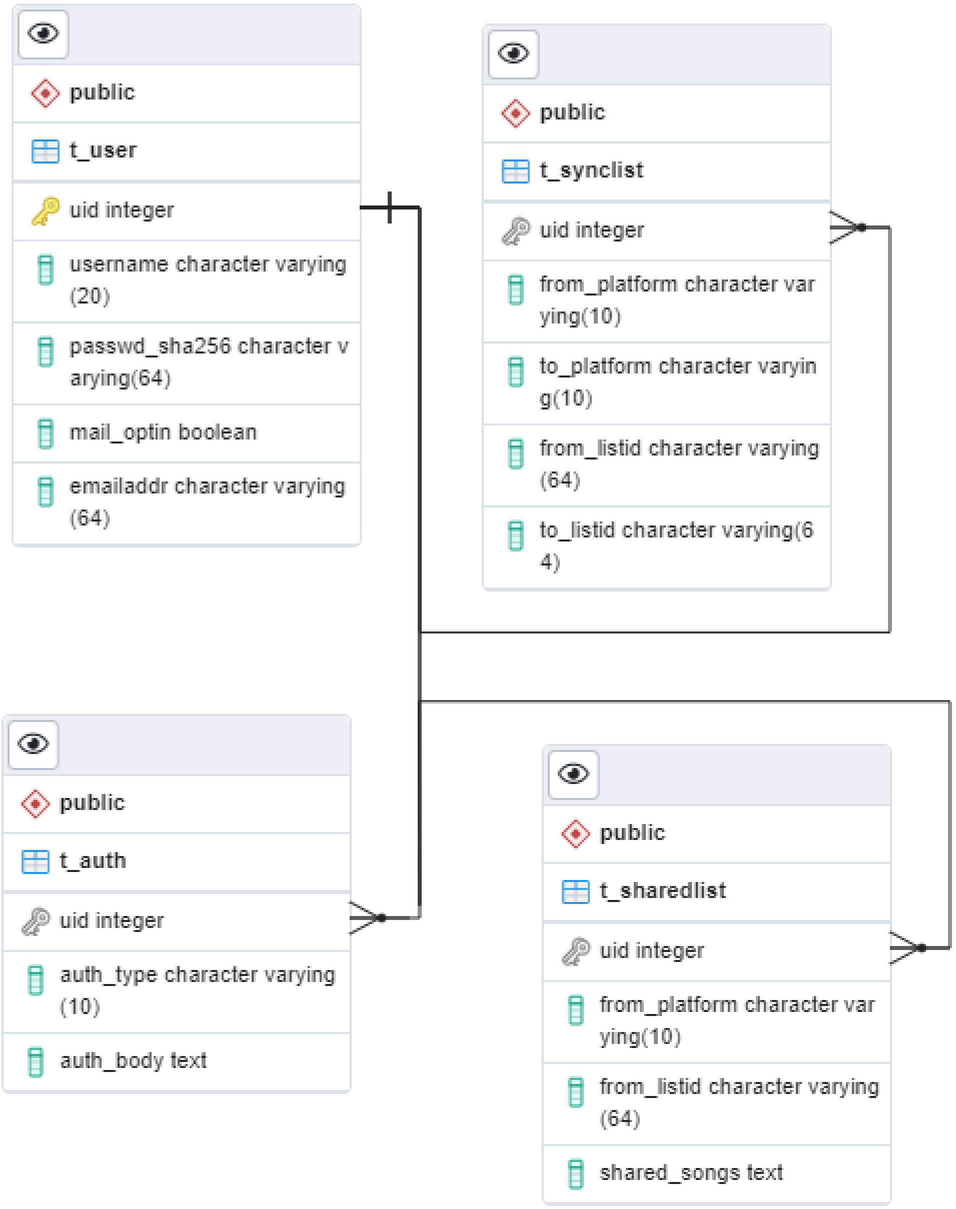
- Youtube Music API: This API is used to read/update/create user's playlists from their YouTube music library. Due to no Official API is provided by Google/YouTube, we are using an unofficial WebRequest-based implementation under python (https://ytmusicapi.readthedocs.io/en/latest/).

- Spotify API: The Spotify API is used to get user's playlist data from their account in order to scan, and potentially import playlist information. Right now, we are using a Spotify developer account (https://developer.spotify.com/), with a created app in order to use the provided API. We pass the user's authorized account information to the API, and the API passes back any fetched data.

Database design

We are using PostgreSQL as DBMS for our project. In the database, we will store the user identification/authorization data in `t_user` and `t_auth`, playlists' synchronization info in `t_synclist,` and user-shared (usually to a friend on another platform) playlist in `t_sharedlist`

Below is the table structure for our current database design.



public
t_user
uid integer
username character varying (20)
passwd_sha256 character varying(64)
mail_optin boolean
emailaddr character varying (64)

public
t_synclist
uid integer
from_platform character varying(10)
to_platform character varying(10)
from_listid character varying (64)
to_listid character varying(64)

public
t_auth
uid integer
auth_type character varying (10)
auth_body text

public
t_sharedlist
uid integer
from_platform character varying(10)
from_listid character varying (64)
shared_songs text

Individual Contributions

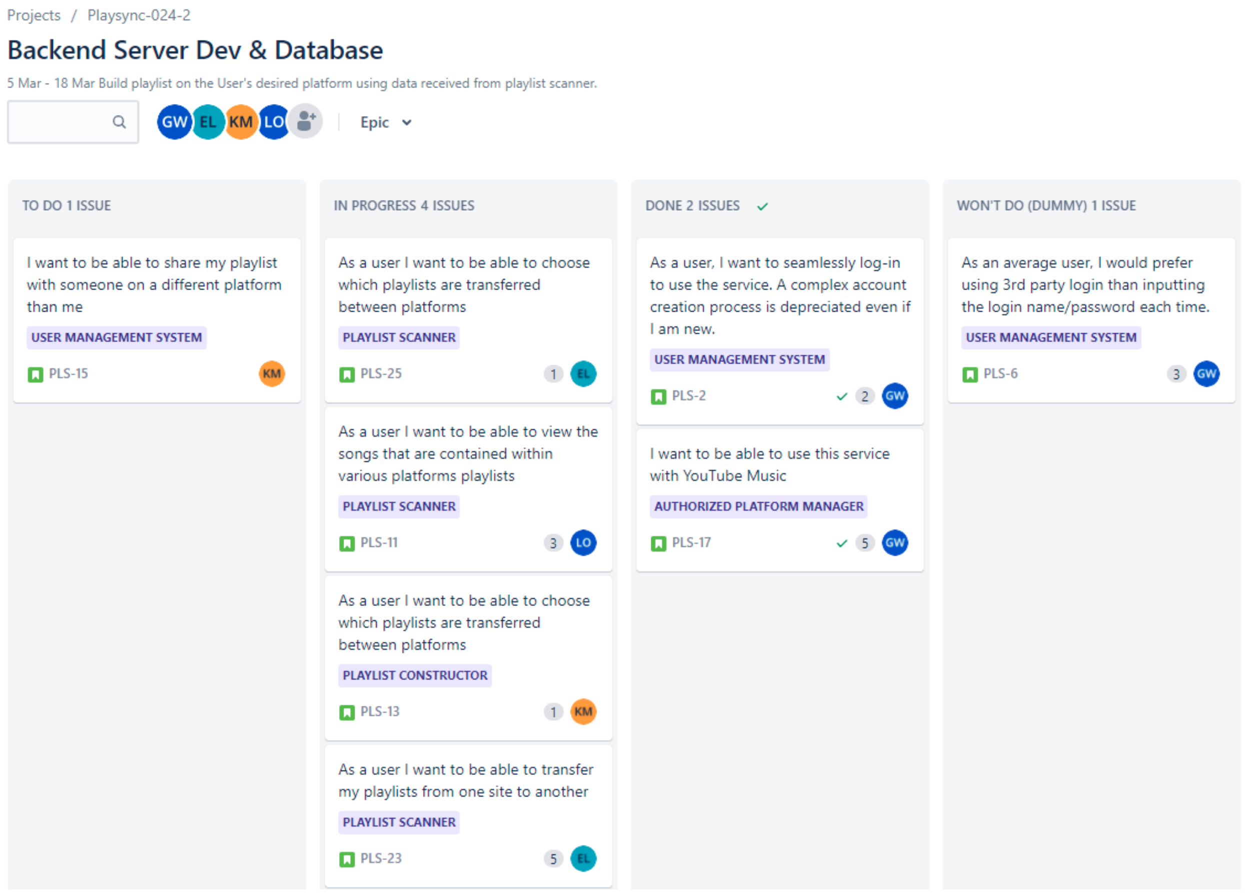
Latest **non-trivial**[1] commit from everyone:

- Gaukas:
    - Mainly working on backend/database design part. Currently developing/integrating Youtube Music API with the backend server. Has already finished user management system's backend/database part.

[1] Excluding: typo fixes, merging other's PR, meeting notes

    - Latest commit: YouTube API Taskers https://github.com/CSCI-3308-CU-Boulder/3308SP21_024_2/commit/82d45e30b3a06e3becc654eee108f4a9c7d04fbc
- Logan:
    - Latest git commit includes updates to the styles.css file along with html for the current profile page where users will be able to change personal settings(e-mail, password) along with music platform auth settings(link new platforms/view current) and delete account. I would like to finish all html and begin focusing solely on API taskers. Awaiting response for approval/denial of apple developer account access from OIT for apple music API access.
    - https://github.com/CSCI-3308-CU-Boulder/3308SP21_024_2/commit/db1b1d0e2f5545beea8e33b688d4e018225745d0
- Kaitlin:
    - Latest git commit includes a folder to a pre-existing Spotify Python library for the Spotify web API as well as a python script that searches Spotify for a given song, artist, or user. So far, the script only searches for public information based on user input and does not edit any data (i.e. create playlist, etc.). The next step is to implement user authentication in order to access the users private data (private playlists, saved songs, etc.)
    - https://github.com/CSCI-3308-CU-Boulder/3308SP21_024_2/commit/e9bcd01056bdd6c9ca7d2bf1a35f3000ac76bde5
- Erica:
    - Worked on various HTML pages for the website, including some JavaScript. Latest commit includes the WIP of the transfer page which will be used to select platforms to transfer to and from, playlists to be transferred, and a display of the content of each song.
    - https://github.com/CSCI-3308-CU-Boulder/3308SP21_024_2/commit/b58509dc471450f7ce18fdb522d0b556b28dd0e0

Current Project Management Board



Projects / Playsync-024-2

Backend Server Dev & Database

5 Mar - 18 Mar Build playlist on the User's desired platform using data received from playlist scanner.

TO DO 1 ISSUE

I want to be able to share my playlist with someone on a different platform than me
USER MANAGEMENT SYSTEM
PLS-15 KM

IN PROGRESS 4 ISSUES

As a user I want to be able to choose which playlists are transferred between platforms
PLAYLIST SCANNER
PLS-25 1 EL

As a user I want to be able to view the songs that are contained within various platforms playlists
PLAYLIST SCANNER
PLS-11 3 LO

As a user I want to be able to choose which playlists are transferred between platforms
PLAYLIST CONSTRUCTOR
PLS-13 1 KM

As a user I want to be able to transfer my playlists from one site to another
PLAYLIST SCANNER
PLS-23 5 EL

DONE 2 ISSUES ✓

As a user, I want to seamlessly log-in to use the service. A complex account creation process is depreciated even if I am new.
USER MANAGEMENT SYSTEM
PLS-2 ✓ 2 GW

I want to be able to use this service with YouTube Music
AUTHORIZED PLATFORM MANAGER
PLS-17 ✓ 5 GW

WON'T DO (DUMMY) 1 ISSUE

As an average user, I would prefer using 3rd party login than inputting the login name/password each time.
USER MANAGEMENT SYSTEM
PLS-6 3 GW

Challenges

- Apple Music API requires a developer account to access features, we are attempting to gain access through CU's OIT department, however, our project does not rely on successful integration of Apple Music API as there are several other platforms with free API's that we are looking into/working with.
- Identifying a common procedure for our core functions that works with all music platforms will be a challenge. As we become more familiar with the various API's and their requirements we will be able to begin working on a solution.
- Handling specific edge cases such as when a user is transferring from platform A to platform B in which platform B does not have one or more songs that are within platform A. We need to identify a common set of rules to follow when we encounter instances such as these (e.g. we simply don't transfer said song(s) and leave a notification for the user, we search for the most similar result, etc..).