Spotify PULSE

Personalized User-Learned Social Experience

Team 18
Rheaa Sharma
Noah Stern
Dane Shaffer
Alex Polivka
Jason Zheng
Collin Scott

CS 307 **Team 18**
Project Name: Spotify PULSE **Project Coordinator**: Alisa Garcia
Team Members: Rheaa Sharma, Noah Stern, Dane Shaffer, Alex Polivka, Jason Zheng, Collin Scott

Table of Contents

Rubric: https://endor.cs.purdue.edu/~cs307/rubrics/design.pdf
Examples:

- https://endor.cs.purdue.edu/~cs307/sample/designdocument_chefly.pdf
- https://endor.cs.purdue.edu/~cs307/sample/designdocument_virtucards.pdf
- https://endor.cs.purdue.edu/~cs307/sample/spring18_designdocument_letsmeet.pdf
- https://endor.cs.purdue.edu/~cs307/sample/spring16_designdocument_laundrychamp.pdf
- https://endor.cs.purdue.edu/~cs307/sample/fall15_designdocument_famus.pdf

Purpose

Spotify offers a range of in-house tools for tracking statistics and song recommendations. However, there are notable gaps in their functionality, leading to the emergence of third-party Spotify optimization products that offer various services to fill the vacuum left by Spotify. For those that live and breathe music, such services–especially when features are not necessarily consolidated in only one–may not be enough. Enter PULSE, our innovative solution designed to seamlessly integrate these separate offerings into a unified platform. With PULSE, music lovers can effortlessly explore new music, access detailed user statistics, track their listening activity, and manage their account and more—all from one convenient location.

Our primary objective is to transform the Spotify user experience by introducing a plethora of exciting features. These enhancements include quality-of-life improvements, providing users with more comprehensive statistics, enabling users to understand their music tastes better. Through these statistics, users can tailor new song recommendations to expand their music library. We are also introducing cutting-edge features, such as an automatic AI DJ playlist mixer and interactive games that leverage your playlists and favorite songs to deliver a dynamic and engaging experience.

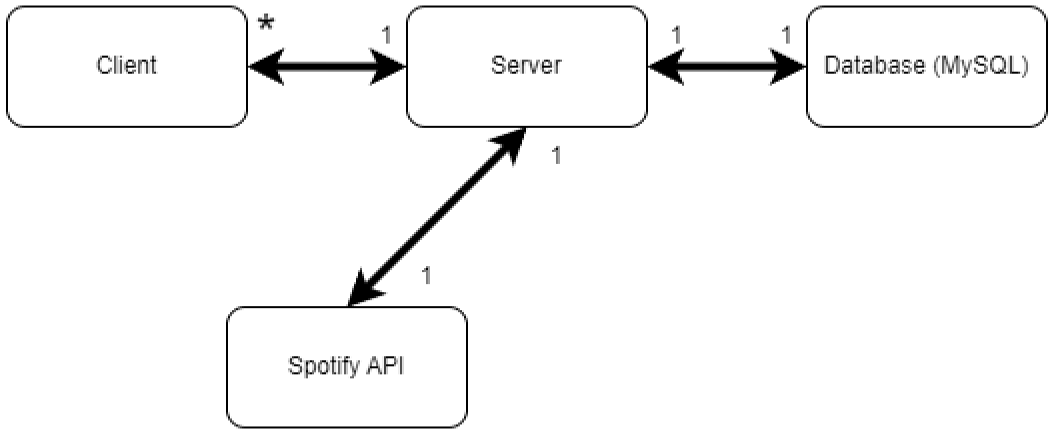
What sets PULSE apart from its main competitors, including volt.fm, last.fm, statsforspotify.com, and Spotify Wrapped, is our commitment to serving as a singular hub for these products' features and more. Our overarching goal is to create a one-of-a-kind, seamless platform that consolidates essential and fun tools, offering a user-friendly, comprehensive solution for all your Spotify needs.

CS 307 **Team 18**
Project Name: Spotify PULSE **Project Coordinator**: Alisa Garcia
Team Members: Rheaa Sharma, Noah Stern, Dane Shaffer, Alex Polivka, Jason Zheng, Collin Scott

Design Outline:

High Level Overview:

Pulse is derived from the many-to-one client-server architecture, with many clients all communicating with the same server. When a user attempts to log in, the login attempt is communicated to the server, and a spotify login page is sent back to the user. When the user logs in with their spotify credentials, Spotify API generates a user token which is grabbed by the server, and used to get the user data from Spotify to be stored in our database. The server will then store, access, and send data as needed based on client requests.



1. **Client**
   a. The client is where the user interacts with the system
   b. The client will have a system flow as described in the diagram below, as well as UI features similar to what is shown in the UI mockup
2. **Server**
   a. The server handles all traffic between the client application and the database.
   b. On login, the server will interact with Spotify's API to authenticate user tokens and transfer data as needed, as well as to send the Spotify login link to the client.
   c. The server will retrieve data from the database after it has been retrieved from Spotify, as well as send said data to the user in the form of statistics when requested
3. **Database**
   a. The database will store relevant data that is retrieved by the server.
   b. The database will store data generated by the user such as game highscores and information about the client's account such as username, friends, and settings
   c. Typical data stored in the database will be favorite/recent songs, artists, and playlists, as well as Spotify generated data about relevant songs including BPM, danceability and energy.

4. **Spotify API**
    a. Spotify API will be used to log users in and generate a unique token which will be used per session by the server to request user data from Spotify.
    b. Spotify API will be used to generate information about users to be stored in the database.

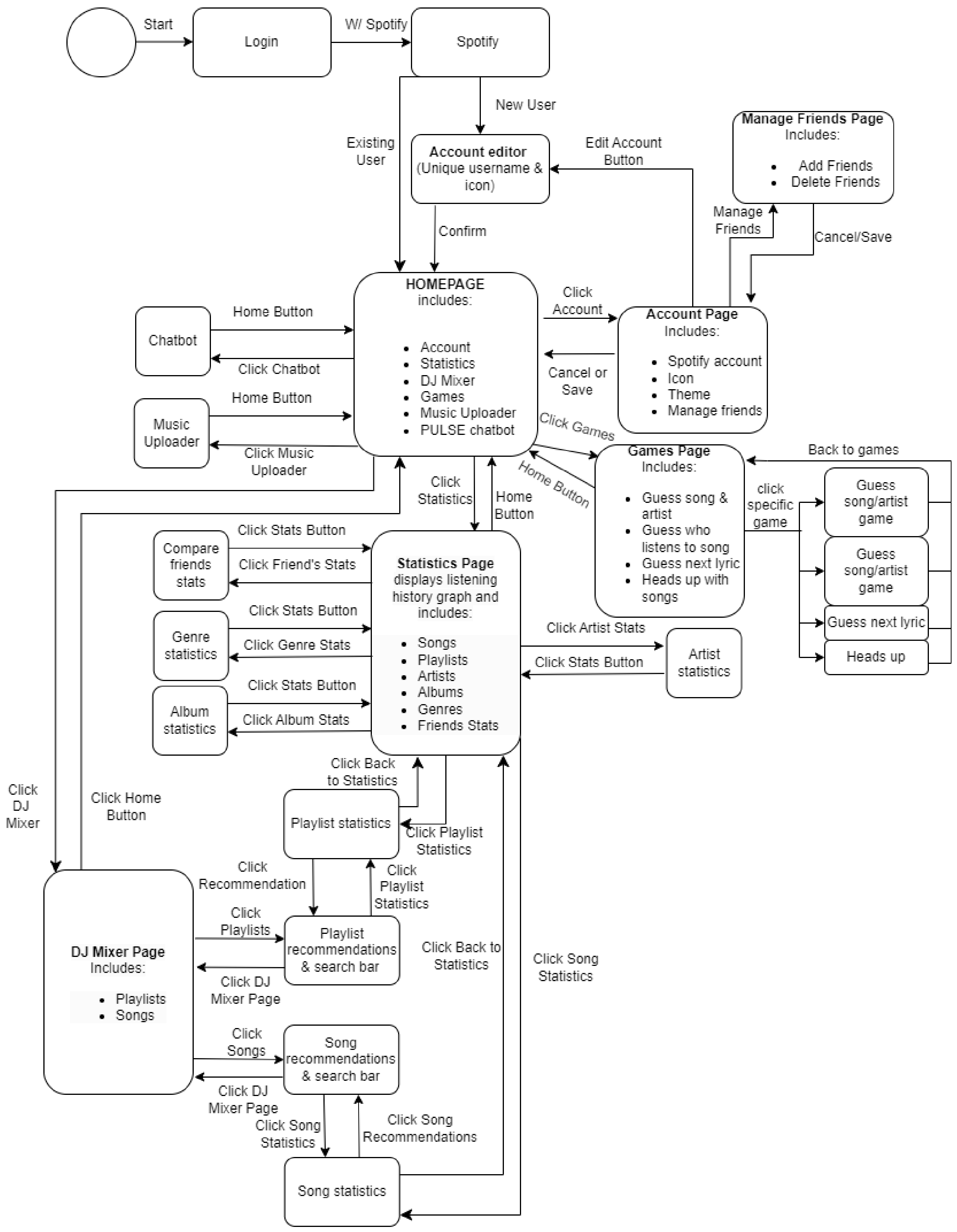
Navigation Flow Map:

Upon entering the PULSE webpage, the user is greeted with a login page in which they are prompted to login with spotify. Continuing to login will take the user to spotify's login page will then take the user back to PULSE upon login. If the user is new to PULSE, they will be prompted to edit their PULSE username and icon before being taken to the homepage

The homepage includes the bulk of PULSE's services including the account page, statistics, DJ mixer, games, music uploader, and PULSE chatbot. Clicking on the chatbot or music uploader tabs will take the user to the respective page, both of which are isolated with only the option to return to the homepage. Similarly a user clicks on the games or account options, they will have the option to choose specific games or account settings which will take them to a new isolated page with only the relevant information and the option to go back.

From the homepage, if the user chooses to look at statistics, they are taken to the statistics page with the options to display various types of statistics, all of which, except for playlist and song statistics, will take them to another isolated statistics page. If the user chooses song or playlist statistics, they are taken to a statistics page which in addition to linking back to the main statistics hub, links to recommendations and search bars too. From this page, the user is able to not only go back to statistics, but go to the DJ mixer hub which links back to the homepage as well.

CS 307 **Team 18**
Project Name: Spotify PULSE **Project Coordinator**: Alisa Garcia
Team Members: Rheaa Sharma, Noah Stern, Dane Shaffer, Alex Polivka, Jason Zheng, Collin Scott



CS 307 **Team 18**
Project Name: Spotify PULSE **Project Coordinator**: Alisa Garcia
Team Members: Rheaa Sharma, Noah Stern, Dane Shaffer, Alex Polivka, Jason Zheng, Collin Scott

Design Issues:

Functional Requirements

1. **Basic Infrastructure**
    a. As a user, I would like to be able to register for a PULSE account using Spotify
    b. As a user, I would like to be able to register for a PULSE account using my email (if time allows)
    c. As a user, I would like to be able to register for a PULSE account using OAuth such as Google (if time allows)
    d. As a user, I would like to be able to login and manage my PULSE account
    e. As a user, I would like to be able to reset my password if I forget it
    f. As a user, I would like to sync my account across devices/platforms
    g. As a user, I would like to access all other pages from a home page
2. **Settings**
    a. As a user, I would like to have dark/light mode
    b. As a user, I would like to be able to change text size
    c. As a user, I would like to have custom backgrounds
    d. As a user, I would like to have preset themes
    e. As a user, I would like to be able to change my Spotify account details in PULSE
    f. As a user, I would like to be able to view my Spotify friends
    g. As a user, I would like to edit my username
    h. As a user, I would like to modify my profile information such as gender, icon picture, and location
3. **Statistics Dump**
    a. As a user, I would like for my top songs and artists to be tracked
    b. As a user, I would like to be able to export spotify data and import it into PULSE to get advanced statistics listed below
    c. As a user, I would like for number of minutes listened to be tracked in my listening activity
    d. As a user, I would like for percentage of song versus total music listened to be tracked in my listening activity
    e. As a user, I would like for times when music was listened to (morning, afternoon, night, etc.) in my listening activity
    f. As a user, I would like for times when songs were listened to fully or skipped or repeated in my listening activity
    g. As a user, I would like to to see songs, artists, playlists, genres and favorite era of music sorted by the metrics above

h. As a user, I would like to be able to specify what time period I want my stats to be from.
i. As a user, I would like to be able to view statistics in a user-friendly fashion
j. As a user, I would like to be able to download my and my friend's statistics in a CSV, pdf, or txt file
k. As a user, I would like to be able to create custom graphs with specified data and formats
l. As a user, I would like to be able to view my friends' statistics as per previously mentioned user tracked statistics
m. As a user, I would like to import my friends stats from a csv/txt file to compare

4. **Playlist Page**
   a. As a user, I would like for recommendations for songs to add on the playlist based on Genre, Artist, Song Writer, Location, Album, or User Emotion (informed by Spotify API)
   b. As a user, I would an AI to DJ for me based on BPM matching
   c. As a user, I would like a playlist activity monitor that displays the amount of times the playlist was played in an user-friendly manner on the playlist page
   d. As a user, I would like to choose two playlists and have a new playlist made from the Spotify-recommended songs of both playlists
   e. As a user, I would like to choose two playlists and have a new playlist to merge them and have them flow into each other based on BPM
5. **Song Page**
   a. As a user, I would like to be able to find similar songs (based on Spotify API parameters such as loudness) to the current song
   b. As a user, I would like to be able to view lyrics to songs
   c. As a user, I would like for songs to be automatically captioned based on lyrics (if time allows)
   d. As a user, I would like for songs to be automatically removed from playlists after a certain threshold of skips set by the user
   e. As a user, I would like for the song to have music videos automatically linked if available on Youtube or other such platforms
   f. As a user, I would like for a song activity monitor that displays the amount of times the song was played in an user-friendly manner on the song page
6. **Game Page**
   a. As a user, I would like to play games such as Guess the song/artist as a quiz format
   b. As a user, I would like to play games such as Guess the song/artist as a heads-up format

c. As a user, I would like to play games such as Guess who listens to this song as a quiz format
d. As a user, I would like to play games such as Guess who listens to this song as a heads-up format
e. As a user, I would like to play games such as Guess the next lyric as a quiz format
f. As a user, I would like to play games such as Guess the next lyric as a heads-up format

7. **Search (if time allows)**
    a. As a user, I would like to be able to search for songs, albums, artists, playlists.
    b. As a user, I would like to be able to filter my search with keywords such as year, genre, new songs and popularity.
8. **Song Player (if time allows)**
    a. As a user, I would like to be able to listen to samples of songs when they are recommended to me
    b. As a user, I would like to be able to listen to spotify through a music player on PULSE
9. **Music Uploader (if time allows)**
    a. As a user, I would like to be able to upload local files and add them to new/existing playlists
    b. As a user, I would like to be able to auto-fill metadata for these local files
10. **TicketMaster API (if time allows)**
    a. As a user, I would like to be able to view music show recommendations when viewing relevant songs/artists/albums
11. **PULSE Chat Bot (if time allows)**
    a. As a user, I would like to interact with a PULSE chatbot through a textbox
    b. As a user, I would like to be able to get song recommendations per 4A-F through PULSE chatbot
    c. As a user, I would like to be able to automatically generate playlists per 4A-F through PULSE chatbot
    d. As a user, I would like to be able to get song information by inputting a short phrase from the song into PULSE chatbot
    e. As a user, I would like to get help with finding/doing a certain action on the website by asking PULSE chatbot
    f. As a user, I would like to be able to give feedback to developers about requested features, bugs or general messages through PULSE chatbot

Nonfunctional Requirements

1. **Quality of Life**
    a. As a user, I would like the website to be responsive with reasonable speed and at all times
        i. Reasonable speed: < 400ms for typical navigation inputs
        ii. Be able to reasonably handle double-inputs

2. **Accessibility**
    a. As a user, I would like the website to be usable in multiple formats
        i. Support for landscape and portrait mode
        ii. Support for desktop and mobile browser
        iii. Support for scaled-up/down windows

3. **Maintainability**
    a. As a developer, I would like to be able to easily maintain features
    b. As a developer, I would like to be able to easily add new features

4. **Security**
    a. Personal information and passwords will be encrypted when transferred using an 18-bit encryption
    b. Encrypt using a more-secure Caesar cipher (if time allows)
    c. User information from Spotify should be stored in a secure manner to protect against common exploits such as cross-site-scripting, buffer overflow, and SQL injection using industry standard practices such as Microsoft's own SQL Server documentation on common security practices. This includes practices like input sanitization, privilege controls, and encryption

5. **Architecture and Performance**
    a. As a developer, I want to run the frontend using ReactJS and the backend using Python
    b. As a developer, I want to use SQL for storing data
    c. As a developer, I want to connect the frontend and backend using Django

Functional Issues

1. What spotify accessibility do you need to create an account with PULSE

   Option 1: No Spotify account
   Option 2: Free Spotify account
   Option 3: Premium Spotify account
   Option 4: Free or Premium Spotify account

   Choice: Option 4
   Justification: We want everyone with a spotify account to be able to access the powerful tools offered by PULSE regardless of if they are a free or premium member. However, Spotify API limits the amount of the data that can be retrieved for free members, making it impossible to offer them all the features of PULSE. With that stated, we are committed to making sure free members have access to all PULSE features that can reasonably be accessed within the limits of the API.

2. Do users need to sign in to use PULSE?

   Option 1: Have the users log in using username/password
   Option 2: Have the users log in using Spotify
   Option 3: Have the users log in using OAuth (Google, Facebook, or Apple)

   Choice: Spotify login w/ PULSE account and OAuth if time allows
   Justification: Users will have to log in using Spotify accounts to link their accounts so that's the default choice. It would be useful if we implement our own account system as well as OAuth. However, since our core functionality are our statistics, song recommendation and games, these login options will be implemented later if we have time to do so.

3. How do we decide what qualifies as a user's favorite song or playlist?

   Option 1: Number of listens
   Option 2: Time listened
   Option 3: Weighted time listened by song length

   Choice: All three.
   Justification: We want to give a lot of power to the user to make the decision for themself for what should count as a 'favorite.' The user will be able to select their choice from a dropdown menu and their favorites will be displayed based on the criteria they selected.

4. How can Spotify account details from the user be updated in PULSE?

Option 1: User will edit their PULSE account manually to reflect their Spotify account details
Option 2: PULSE account details will automatically be updated when the user edits them in Spotify

Choice: Both
Justification: We want the user to be able to customize their icon and username if they like to be different from their Spotify account but certain details such as spotify user name and other details will be the same as Spotify

5. What customization options should be provided for the user
   Option 1: Full customization – everything including background, theme, and text
   Option 2: Limited customization – custom color and limited text options
   Option 3: Preset themes – users can choose from preset themes such as light and dark

   Choice: Option 3 – Preset themes
   Justification: Preset themes allow for the user to customize their PULSE experience, while not allowing them access to modify every specific element which would have the potential to pose stability issues. Users will be able choose a look for PULSE that is meaningful and handcrafted by the developer team. In the future, more complex themes could be added to give the user even more flexibility.

6. How should custom graphs be created and displayed for users?
   Option 1: Full customization - everything can change
   Option 2: Limited customization - choice of graph type, background, text size, data
   Option 3: Fully preset graphs - user only chooses graph type and data

   Choice: Option 3 – Fully preset graphs
   Justification: This allows for the easiest time developing graphs with a good look. By only allowing the choice of graph and data, it decreases the work required to display the data. In the future, more customization could be added but for now, preset graphs would be beneficial for us as we are inexperienced developers.

7. What format should interactive games follow (e.g., Guess the song/artist)?
   Option 1: Single-device format.
   Option 2: Multi-device or competitive format.

Choice: Single-device format
Justification: Single-device format would allow for easier development time and is also the main way users would be envisioned to play these games since these are party/friend games. Moreover, multiplayer games across the internet increases complexity for something that is PULSE's main feature.

8. How should the user interact with the PULSE chat bot?
   Option 1: Preset prompts that the user can select from a dropdown
   Option 2: Use ChatGPT API to train a machine-learned chat bot to take text queries
   Option 3: Train our own chat bot from scratch/libraries

   Choice: ChatGPT API chat bot
   Justification: Utilizing ChatGPT API to generate a useful chatbot could get our chatbot up and running quickly. While this relies on ChatGPT to provide their services for free, creating a decision tree chatbot or training a whole new chat bot is far more work for minimal benefit compared to ChatGPT considering our lack of expertise.

9. What criteria should be used for recommending songs to add to playlists?
   Option 1: Genre, artist, songwriter, location, album, user emotion.
   Option 2: BPM-based recommendations.

   Choice: Both
   Justification: Taking more heuristics into account when giving recommendations to the user will allow for more accuracy. While creating BPM-based recommendations may take a significant amount of time, it is something that the team feels strongly about incorporating into the project in order to provide the user with a positive experience.

10. How can song uploading be handled while considering copyright and legal concerns?
    Option 1: Implement strict copyright checks.
    Option 2: Allow users to upload local files responsibly.

    Choice: Allow users to upload local files responsibly
    Justification: Since we are utilizing Spotify's backend to perform this operation, we would not be hosting the data and thus this is not our concern but rather Spotify's concern. Moreover, implementing strict copyright checks falls into the realm of music DRM which is a far more complex issue than what we could reasonably perform in a short period of time compared to Spotify. We would

however add a disclaimer so that users are aware of the legal responsibilities they hold, and our development team is not at risk for legal issues.

11. What tasks should the chatbot handle, and how should interactions be designed?
    Option 1: Providing song recommendations, song information, and user assistance
    Option 2: Expanding chatbot functionality to include automatic playlist generation, playlist modification, automatic setting changes, etc.

    Choice: Provide the basic functionality based on option 1 but if time allows expand the functionality to include option 2.
    Justification: The chatbot will provide the basic functionality that is self-enclosed with recommendations, song information and user assistance. These other than the user assistance will be calling functions that we will be implementing already. Then option 2 will mostly be implemented as well but they can fundamentally change user functionality so more work will have to be invested to avoid harming the user.

Non-Functional Issues

1. What framework should we use?
    Option 1: ReactJS
    Option 2: HTML/CSS
    Option 3: Angular
    Option 4: Vue

    Choice: ReactJS
    Justification: ReactJS is our choice since we are attempting for our homepage to have a dashboard view which means we will have a lot of repeated similar looking views. ReactJS allows us to build components and reuse them which will be very valuable in our case.

2. What backend server should we use?
    Option 1: Amazon Web Services
    Option 2: Azure
    Option 3: Google Cloud
    Option 4: Heroku

    Choice: Heroku
    Justification: For an academic project that is only a semester long with relatively inexperienced developers, Heroku is our best choice. This is because the other

platforms are complex and typically used by organizations which leads to a higher price and more complex coding. Heroku on the other hand has plenty of free addons, documentation, community support and a free tier for smaller projects like PULSE.

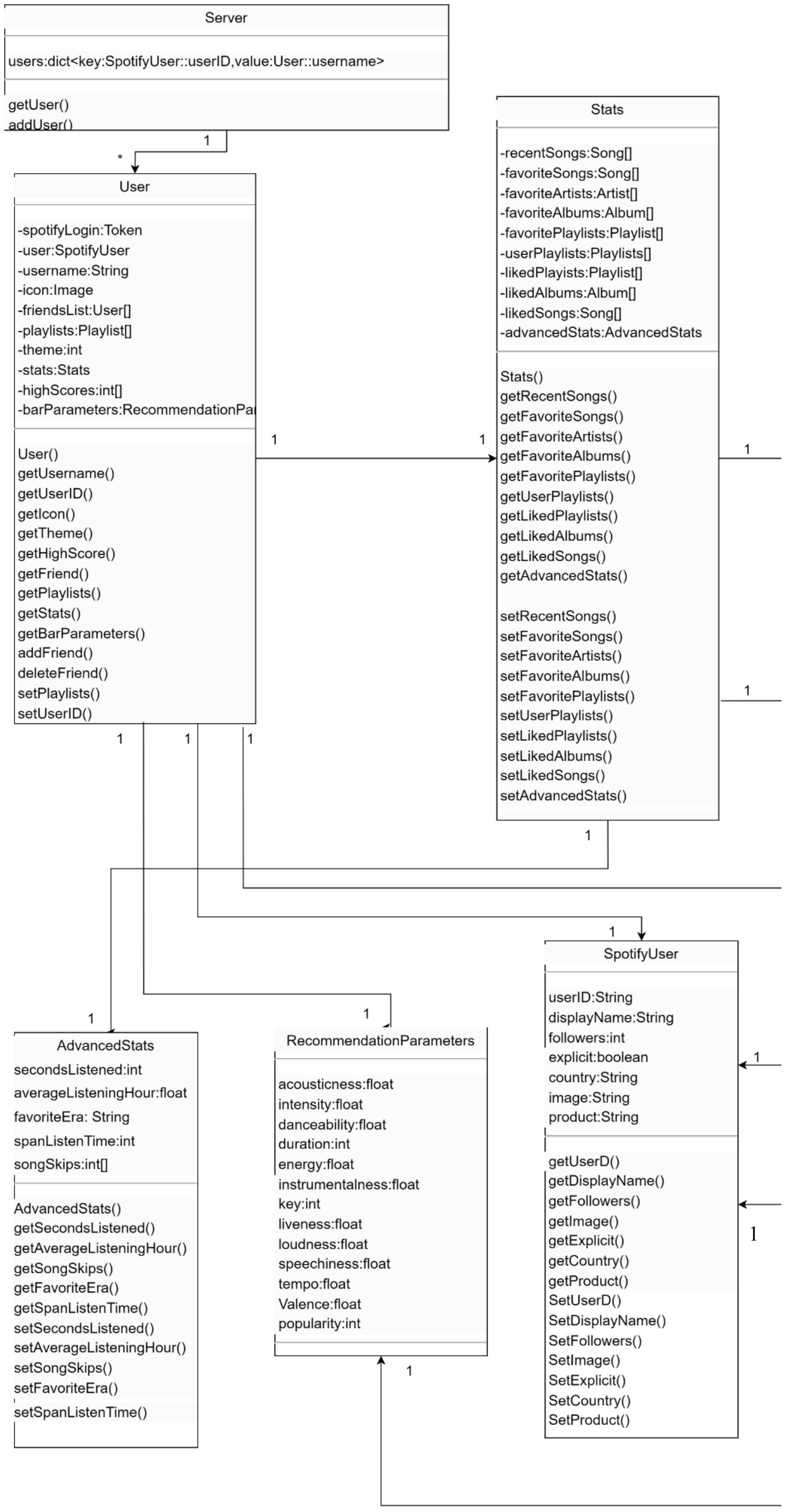
3. What database should we use?

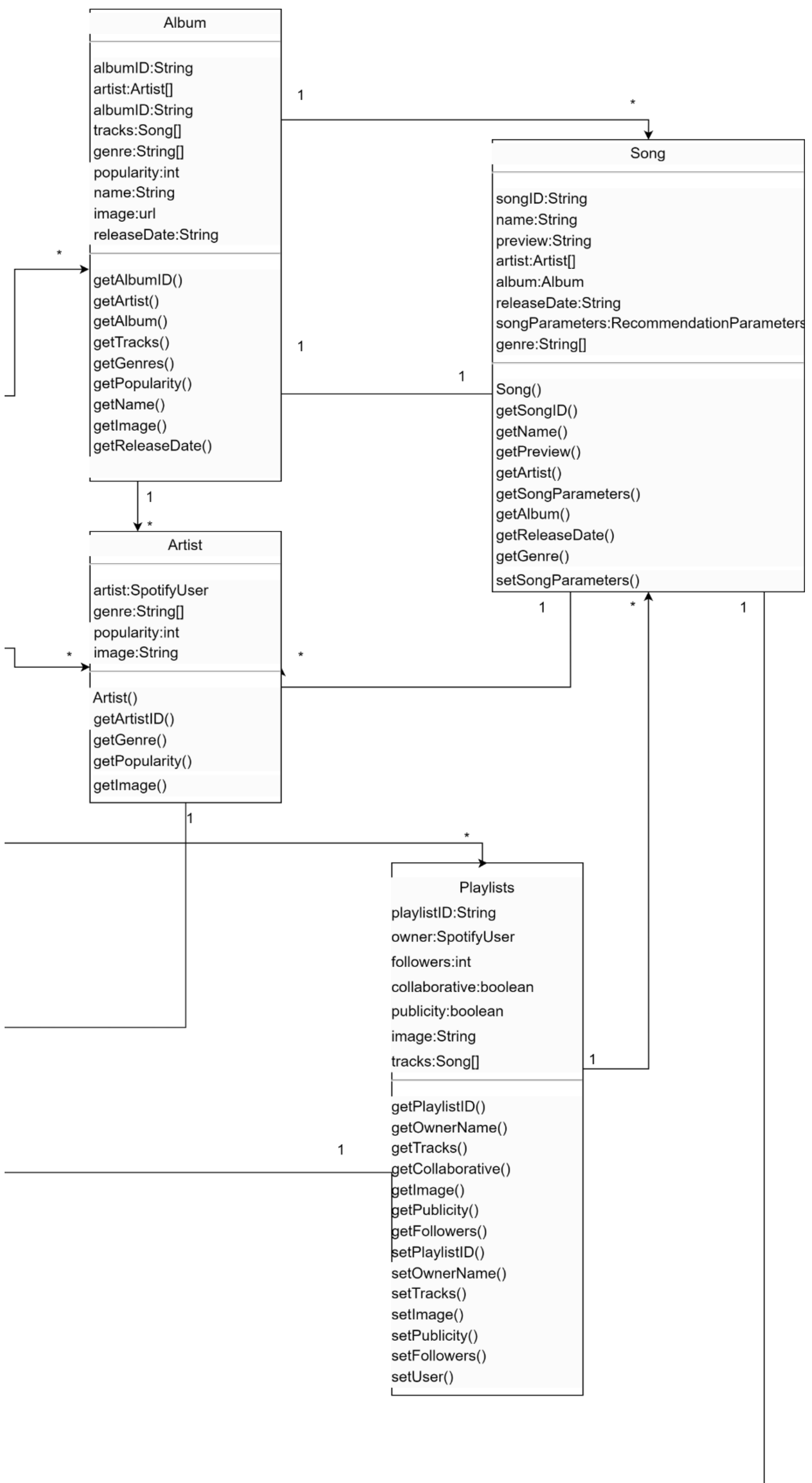
   Option 1: MongoDB
   Option 2: MySQL
   Option 3: Oracle XE
   Option 4: Redis

   Choice: MySQL
   Justification: MySQL is relatively easy to work with and it is something most of the developers on the team are familiar with. There's no reason to complicate the choice by going with something else that many people are unfamiliar with when MySQL is sufficient. Also SQL is a popular skill to have in industry and is good to learn.

CS 307 **Team 18**
Project Name: Spotify PULSE **Project Coordinator**: Alisa Garcia
Team Members: Rheaa Sharma, Noah Stern, Dane Shaffer, Alex Polivka, Jason Zheng, Collin Scott

Design Details:




Album
albumID:String
artist:Artist[]
albumID:String
tracks:Song[]
genre:String[]
popularity:int
name:String
image:url
releaseDate:String
getAlbumID()
getArtist()
getAlbum()
getTracks()
getGenres()
getPopularity()
getName()
getImage()
getReleaseDate()
Song
songID:String
name:String
preview:String
artist:Artist[]
album:Album
releaseDate:String
songParameters:RecommendationParameters
genre:String[]
Song()
getSongID()
getName()
getPreview()
getArtist()
getSongParameters()
getAlbum()
getReleaseDate()
getGenre()
setSongParameters()
Artist
artist:SpotifyUser
genre:String[]
popularity:int
image:String
Artist()
getArtistID()
getGenre()
getPopularity()
getImage()
Playlists
playlistID:String
owner:SpotifyUser
followers:int
collaborative:boolean
publicity:boolean
image:String
tracks:Song[]
getPlaylistID()
getOwnerName()
getTracks()
getCollaborative()
getImage()
getPublicity()
getFollowers()
setPlaylistID()
setOwnerName()
setTracks()
setImage()
setPublicity()
setFollowers()
setUser()
1
*
1
1
1
*
*
*
*
1
*
1
1
*
1
1

Class Diagram

- Server
  - Stores key, value pairs in a Python dictionary with the key being the unique Spotify userID and the value being a reference to the user's PULSE account
- User
  - Stores information about users who have registered with PULSE
  - Including personalized information such as username and avatar
  - Includes settings information
  - Includes statistics and game high score information
- SpotifyUser
  - Stores information specific to user's Spotify account
- Stats
  - Includes arrays for top artists, songs, playlists, etc
- Song
  - Informational class that holds information about songs and relates it to artist, and album
- Artist
  - Informational class that holds information about Spotify artists
- Album
  - Informational class that holds information about albums and relates it to artist, and songs
- Playlists
  - Object that reflects a Spotify playlists
  - Gives information about Spotify owner, number of followers
  - Includes the songs included on the playlist
- RecommendationParameters
  - Stores information about the user's parameters for searching for Spotify music recommendations
- AdvancedStats
  - Includes advanced statistics that are parsed from user uploaded Spotify data

Sequence Diagram

The sequence of events given below outlines the primary interactions between the user, server, database, and Spotify. The first thing the user does is logs in through Spotify. Once this is complete, the server requests an authentication token using the Spotify API which will allow the user to get their data through the website.

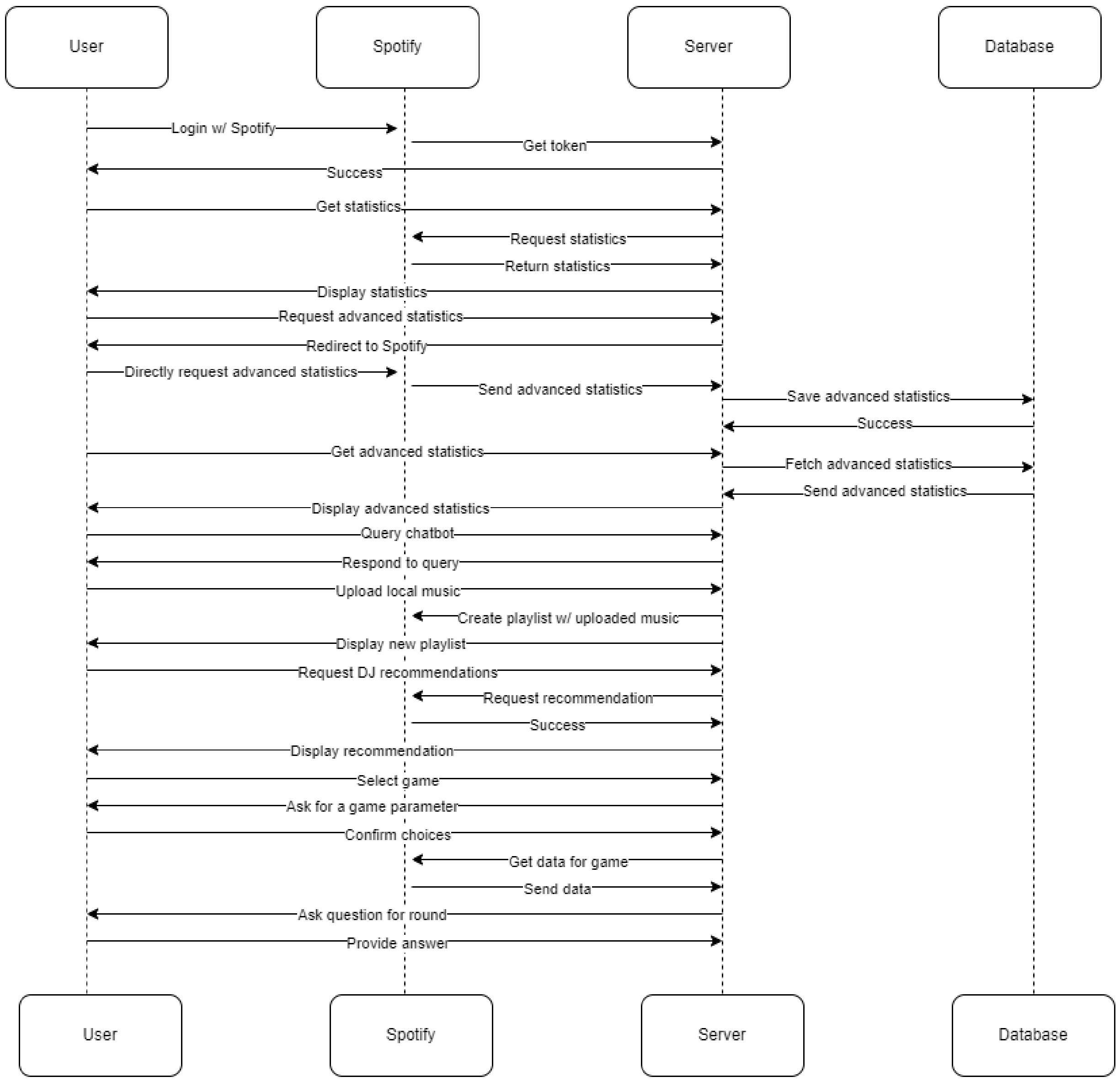
Whenever the user requests basic statistics, such as those available through the Spotify API, the server will query the API for the statistics and display them to the user. When more advanced statistics are requested, the server will prompt the user to directly request Spotify for their data. Once this data is obtained by the server, it is cached in the database. When the user wants to access their advanced statistics using this data, the server will fetch it from the database and process it before displaying the requested statistics to the user.

When the user requests recommendations, the server will interface with the Spotify API to get the relevant data. The server will then process the data, mixing the Spotify song parameters with machine-learned parameters to present the user with personalized music recommendations.

For users interested in uploading their local music collection, the server will create a playlist with the new music using the Spotify API and then display the new playlist to the user.

When the user wants to play a game, they initiate the process by specifying their preferred game and additional parameters such as the number of rounds and players. For each round, the server will interact with the API to obtain the proper data. This data will be processed and used to formulate a relevant question that will be sent to the user. The user will respond with the answer and send it back to the game. Lastly, the server will process their answer before moving onto the next round.

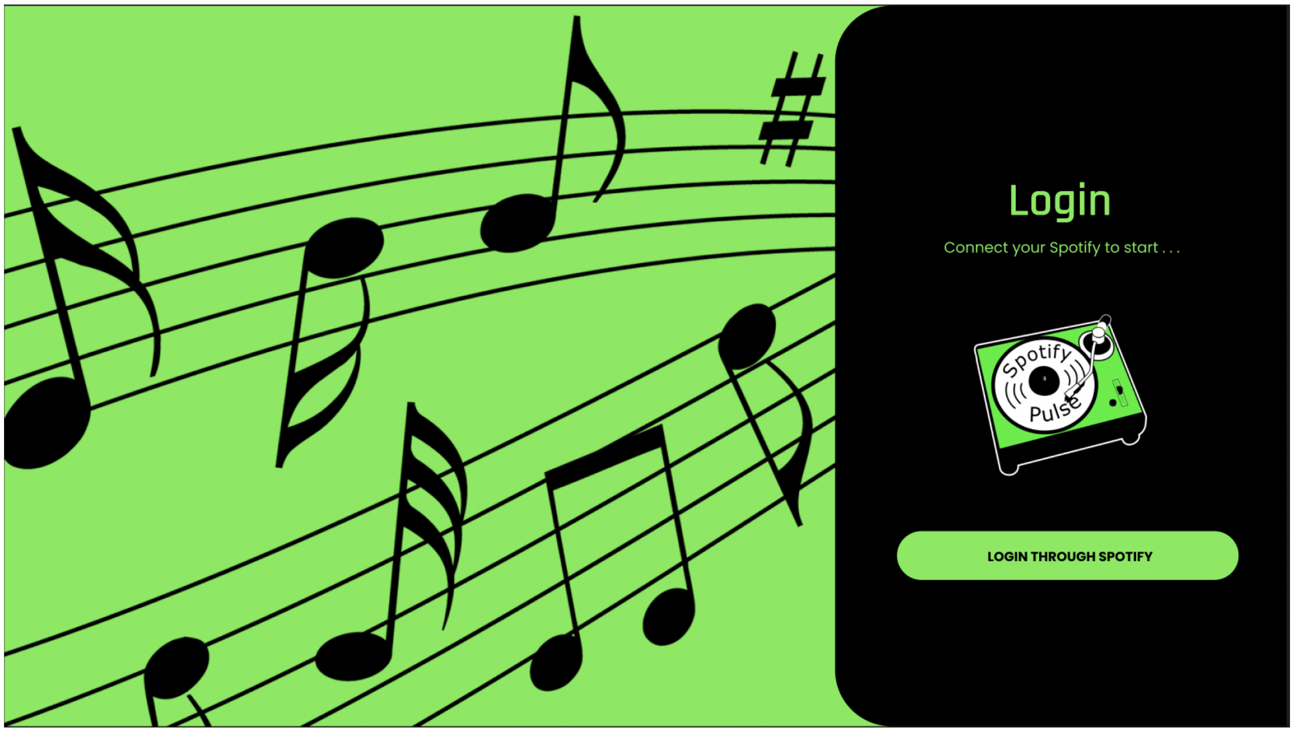
CS 307 **Team 18**
Project Name: Spotify PULSE **Project Coordinator**: Alisa Garcia
Team Members: Rheaa Sharma, Noah Stern, Dane Shaffer, Alex Polivka, Jason Zheng, Collin Scott

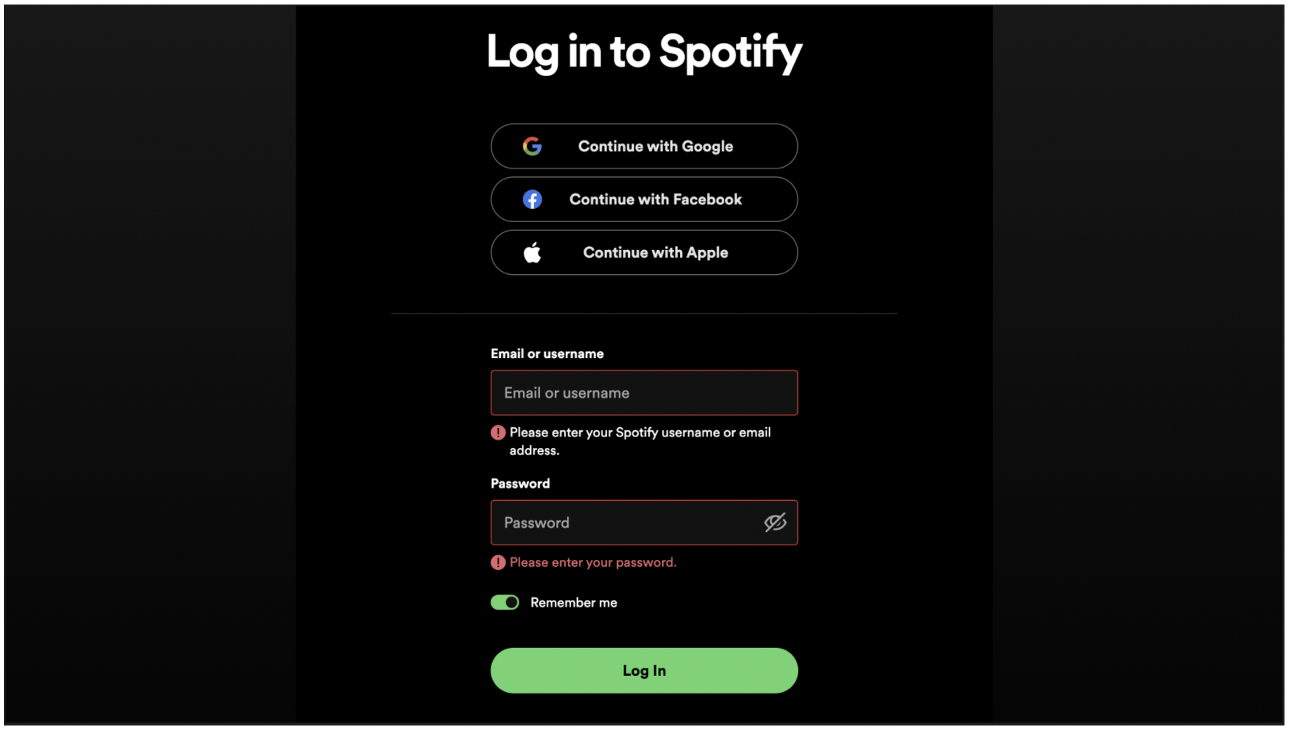


CS 307 **Team 18**
Project Name: Spotify PULSE **Project Coordinator**: Alisa Garcia
Team Members: Rheaa Sharma, Noah Stern, Dane Shaffer, Alex Polivka, Jason Zheng, Collin Scott

Mockup UI

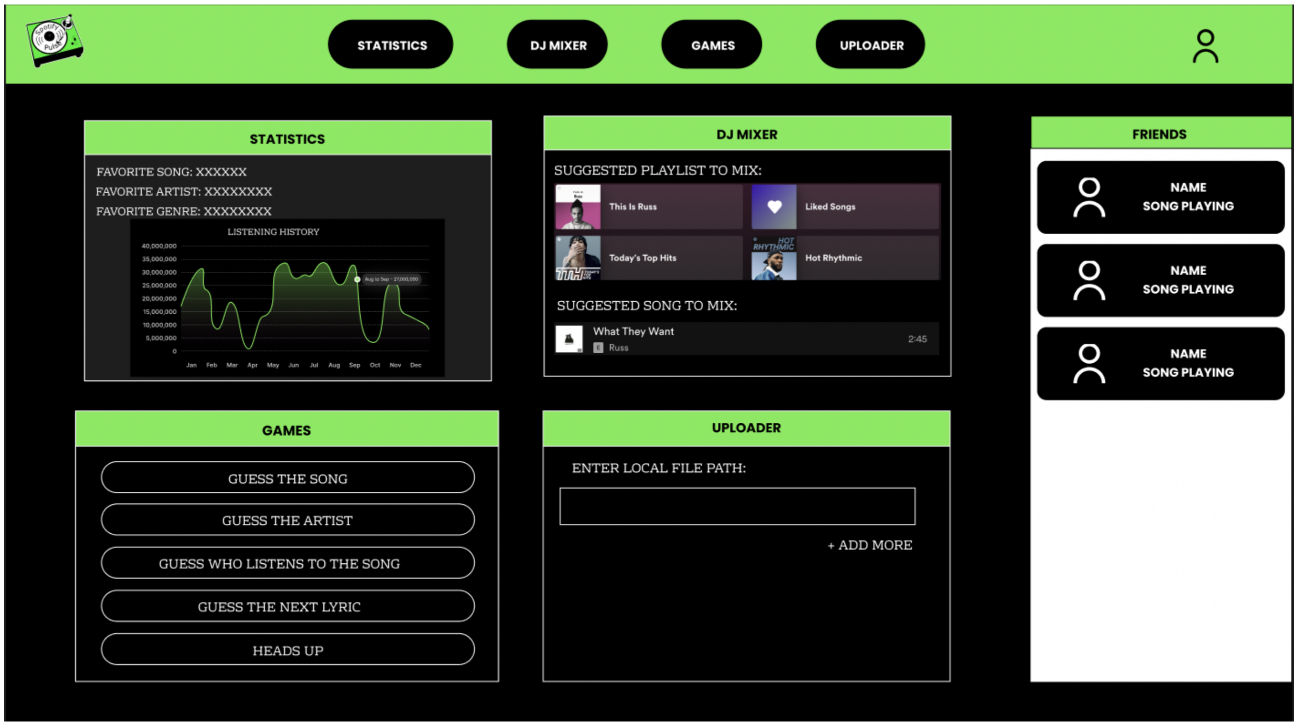
Home page/ Login page



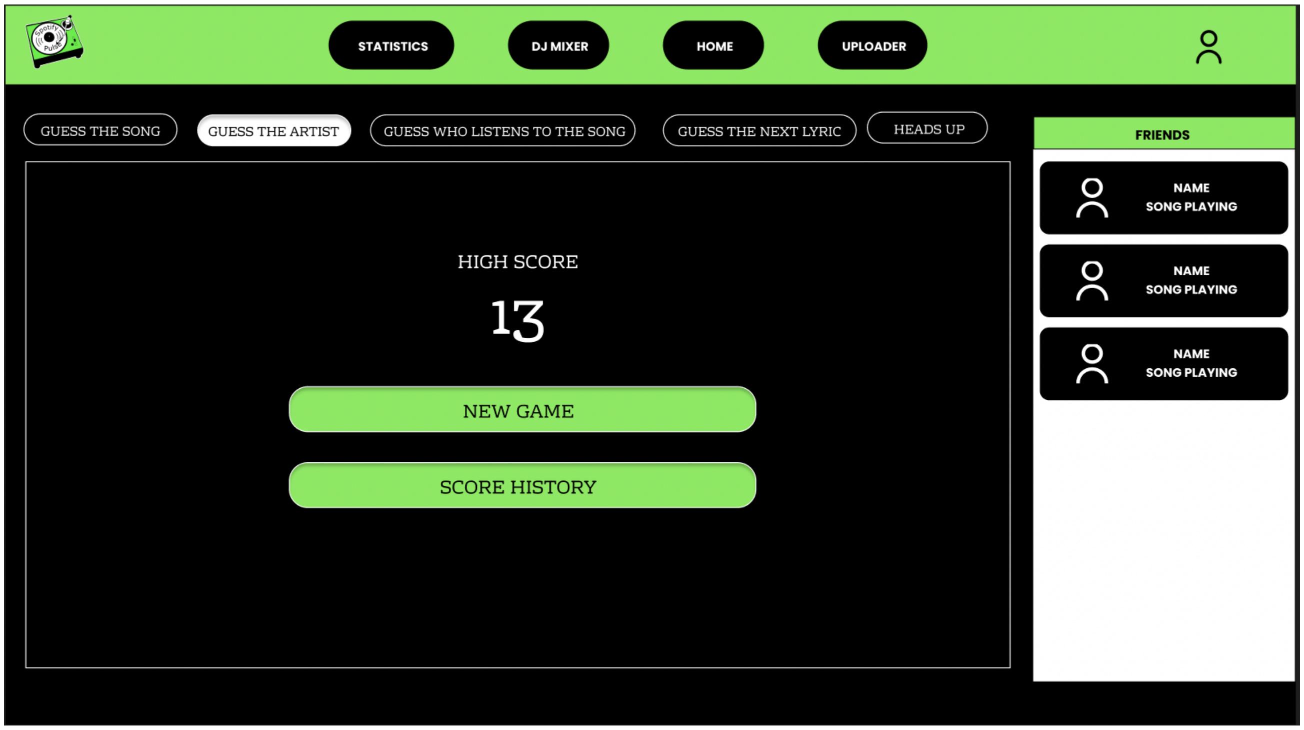


CS 307 **Team 18**
Project Name: Spotify PULSE **Project Coordinator**: Alisa Garcia
Team Members: Rheaa Sharma, Noah Stern, Dane Shaffer, Alex Polivka, Jason Zheng, Collin Scott

DashBoard Page

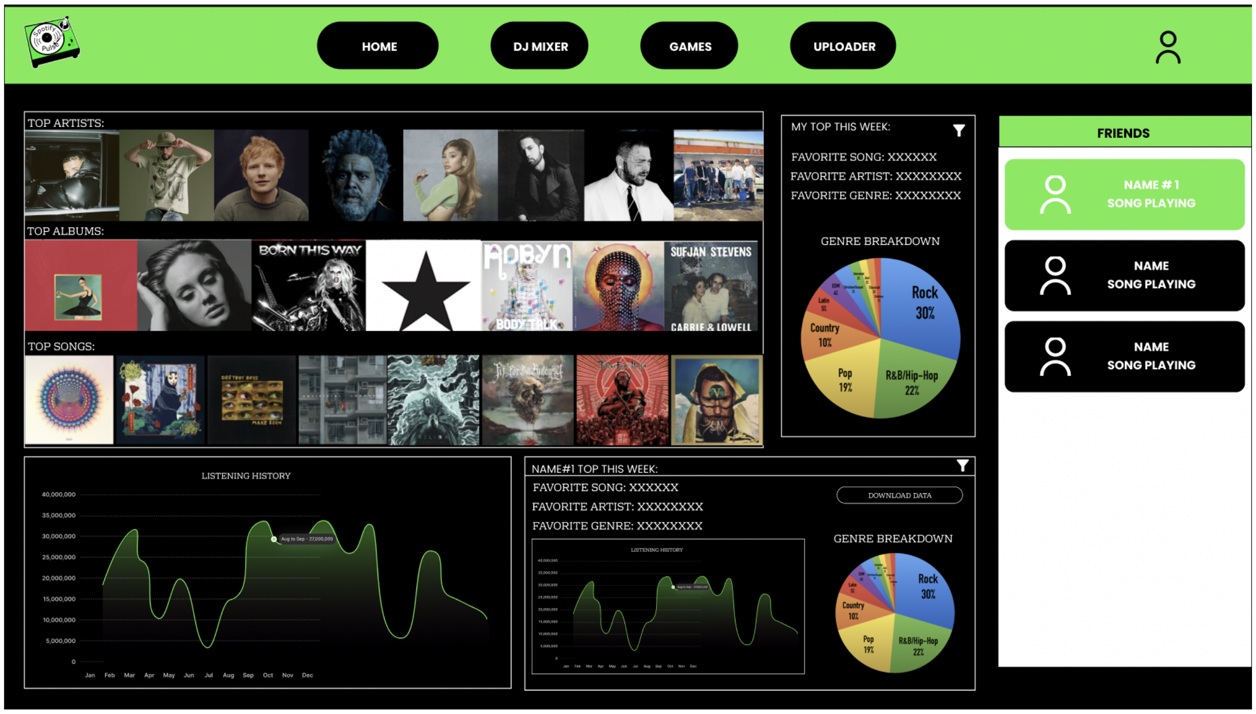


Games page



CS 307 **Team 18**
Project Name: Spotify PULSE **Project Coordinator**: Alisa Garcia
Team Members: Rheaa Sharma, Noah Stern, Dane Shaffer, Alex Polivka, Jason Zheng, Collin Scott

Statistics page



MY TOP THIS WEEK:

FAVORITE SONG: XXXXXX
FAVORITE ARTIST: XXXXXXXX
FAVORITE GENRE: XXXXXXXX

NAME#1 TOP THIS WEEK:

FAVORITE SONG: XXXXXX
FAVORITE ARTIST: XXXXXXXX
FAVORITE GENRE: XXXXXXXX