**Software Requirements and Design Document**

**For**

**Group 22**

Version 3.0

**Authors**:

Rebecca Anestad

Daniel Bryant

Cullen Hemsouvanh

Lance McMahon

Yassar Rich

# Overview (5 points)

We are developing a Web App that utilizes Spotify’s API to extend the platform’s functionality, creating a more customizable user experience. This web application interacts with the Spotify API in order to allow users to more easily share their music preferences with others, and let others know their opinions on different tracks. We utilize collaborative playlists and song recommendations to facilitate and encourage users to share their favorite songs.

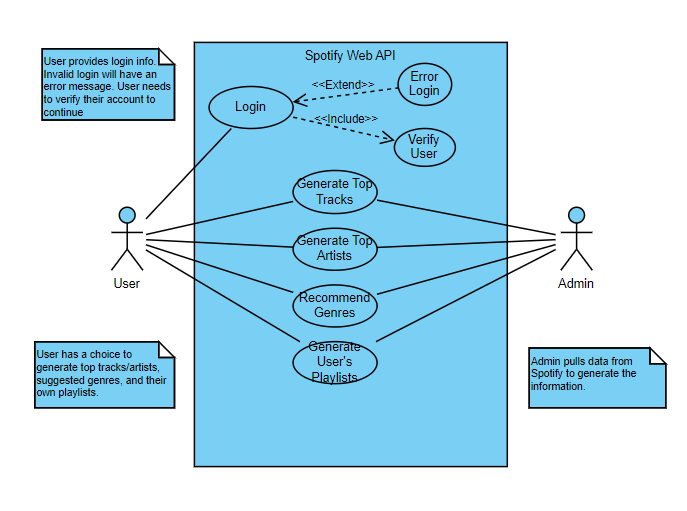
# Functional Requirements (10 points)

1. Finished - Implement a functional homepage
2. Finished - buttons within the project properly link to the correct page
3. Finished - the Spotify API needs to be correctly linked so that people can log into Spotify and have the website fully functional
4. Finished - pull from Spotify information about the users top artists, genres, and playlists, and display the data on each page accordingly
5. Medium priority - look at the users spotify history and suggest new artists, genres, and playlists
6. Low priority - (only adding if we have time) SQL database in back end to create playlists with other users and to comment on songs

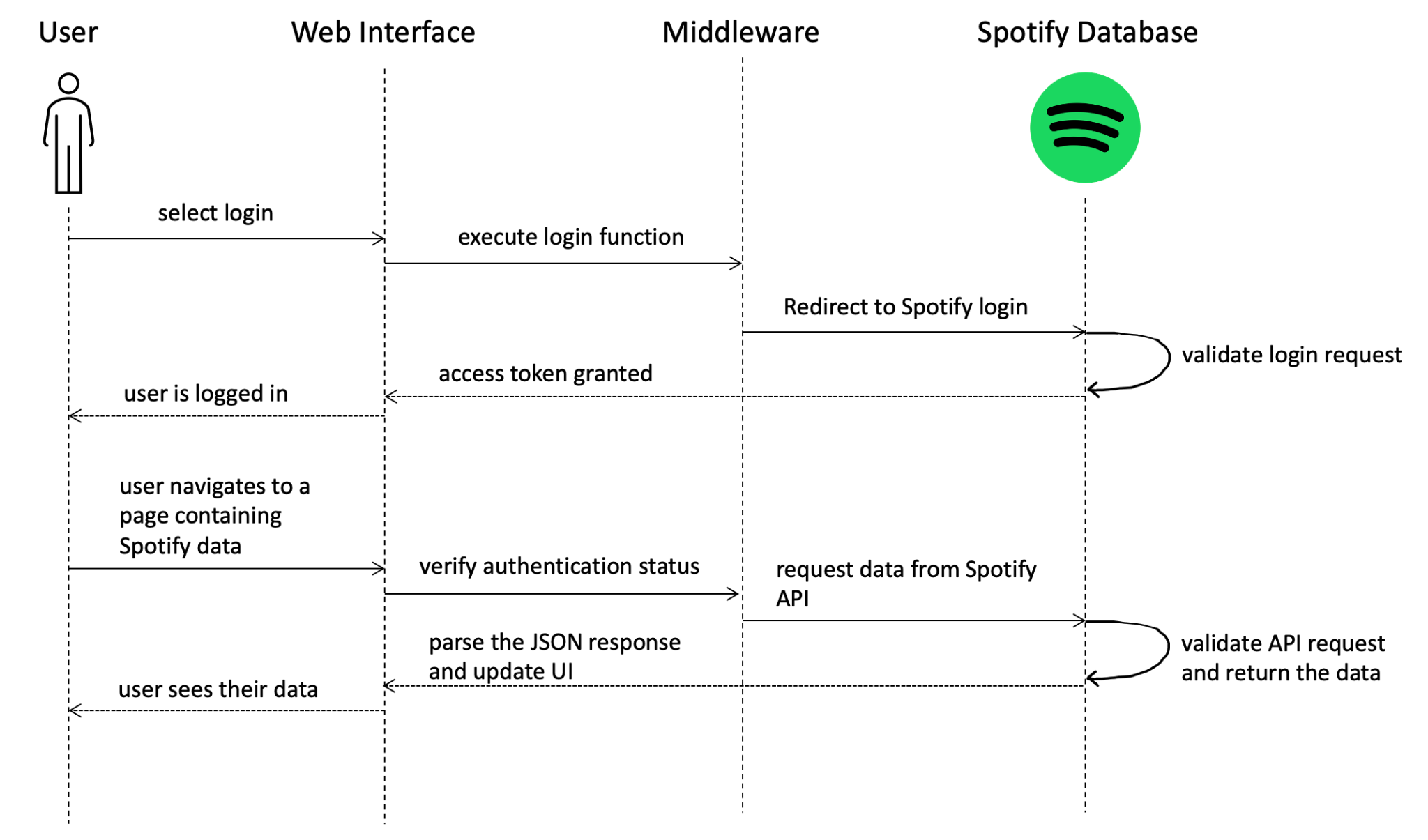
# Non-functional Requirements (10 points)

1. Account Security - Spotify OAuth must be implemented securely to prevent attacks on user’s accounts
2. Software Quality - Pages and components must function consistently after sufficient unit testing
3. Performance - Webpages must have reasonable load times, API requests should also have reasonable response times. A reasonable response time for a page to load on the desktop is less than two seconds, according to most companies.
4. Reliability - The website needs to be robust enough so that it can handle 100,000+ users in a deployment setting
5. Scalability - Backend systems and web servers must be created in such a way that lays the foundation for scaling

# Use Case Diagram (10 points)

**

# Class Diagram and/or Sequence Diagrams (15 points)

**

# Operating Environment (5 points)

The web application will be hosted on Vercel and will not be limited to a specific hardware platform, as long as the computer has the ability to access the web, they will be able to access our website.

# Assumptions and Dependencies (5 points)

If someone wants to run the project locally it is assumed that they have Node on their computer and can download all the other dependencies on our project using npm install