Software Samurai

Sound Sensei

Team 6: Sound Samurai’s
Members: Jesse Javana, Alysha McCullough, Anna Mikhailenko, Ismail Bilmece, Gwendelen Cady

Description & Technologies & Roles

Description:

- This user friendly application gives millions to control and listen to music all over the world.
- A third party playlist curation application.

Technologies:

- **Design Tools:**
- GIMP
- **Development Tools:**
- Draw.io, GitHub, IDE, Spotify API
- **Management Tool:**
- Discord, Trello, OneNote - meeting

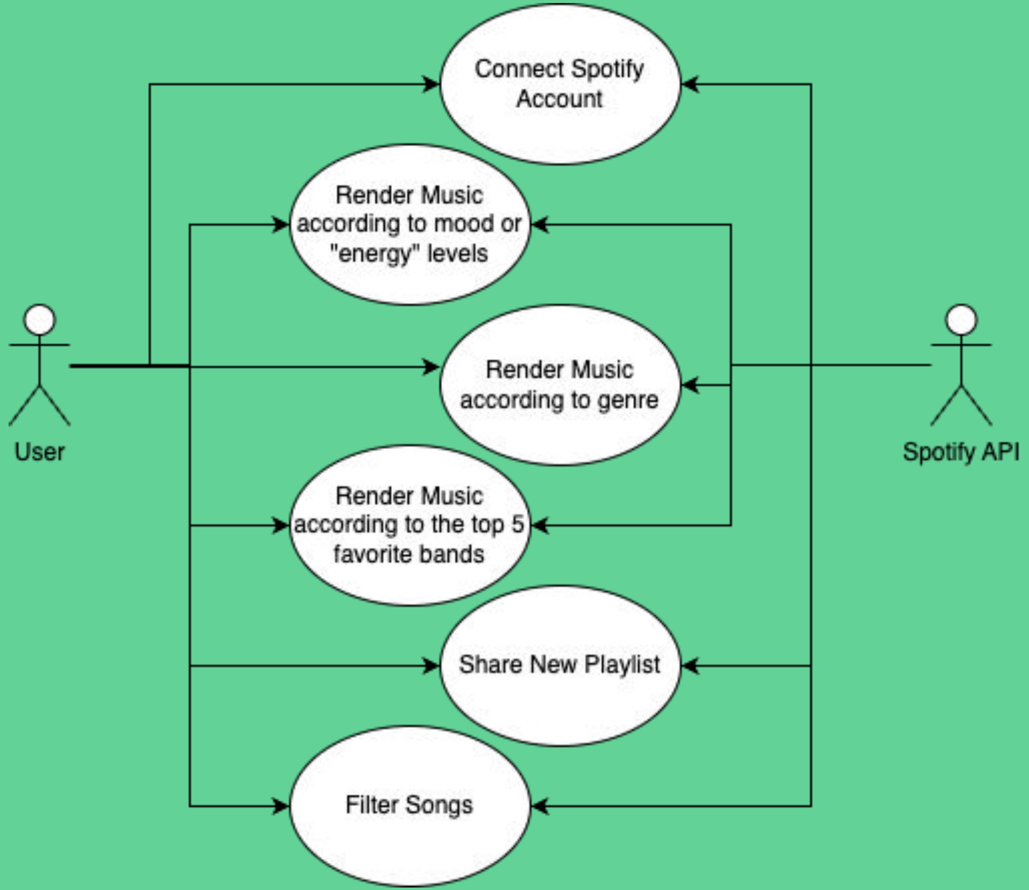
Roles:

- Designer (Gwen)
- User Interface (Anna/Alysha)
- Backend (Jesse/ Ismail)

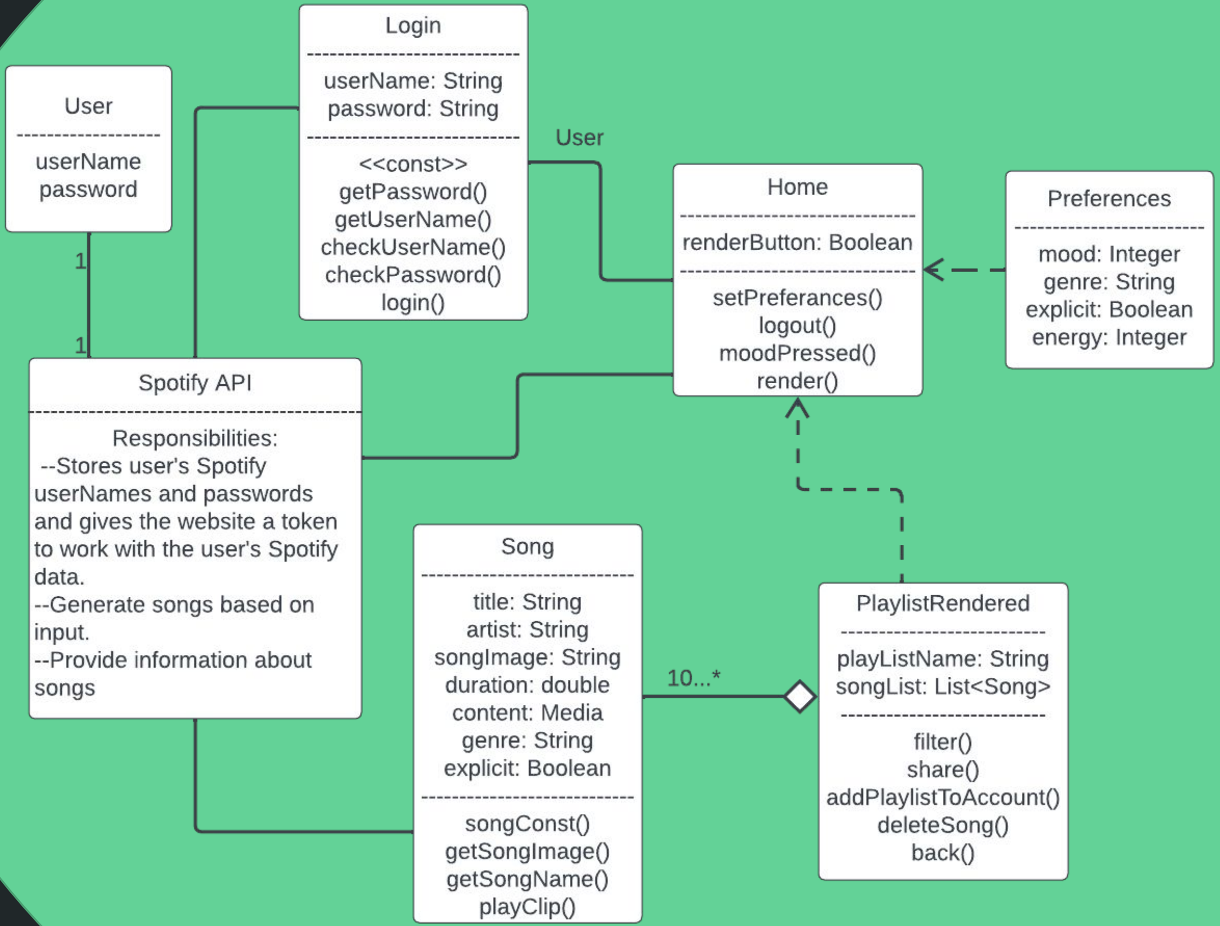
Requirements (Summary)

- The website will curate a new playlist based on user inputs and their listening preferences, using multiple different aspects of Spotify's API.
- There will be a feature that will share the playlist by copying the link to the playlist.
- The website will be able to generate a new playlist in a timely manner using proper and efficient coding.
- There shall be a feature that allows users to add the newly created playlist to their spotify account, using Spotify's API.

Use Case Diagram

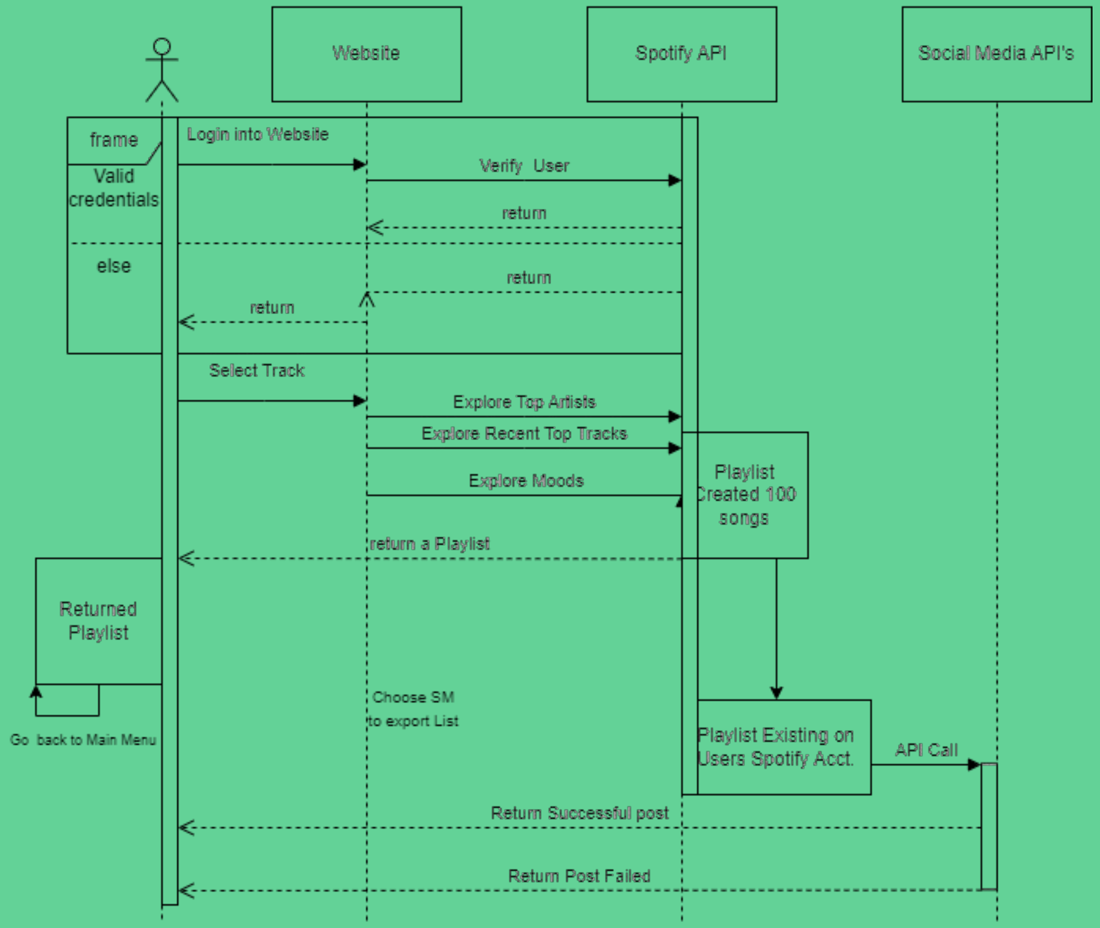

Connect Spotify Account
Render Music according to mood or "energy" levels
User
Render Music according to genre
Spotify API
Render Music according to the top 5 favorite bands
Share New Playlist
Filter Songs

Class Diagram

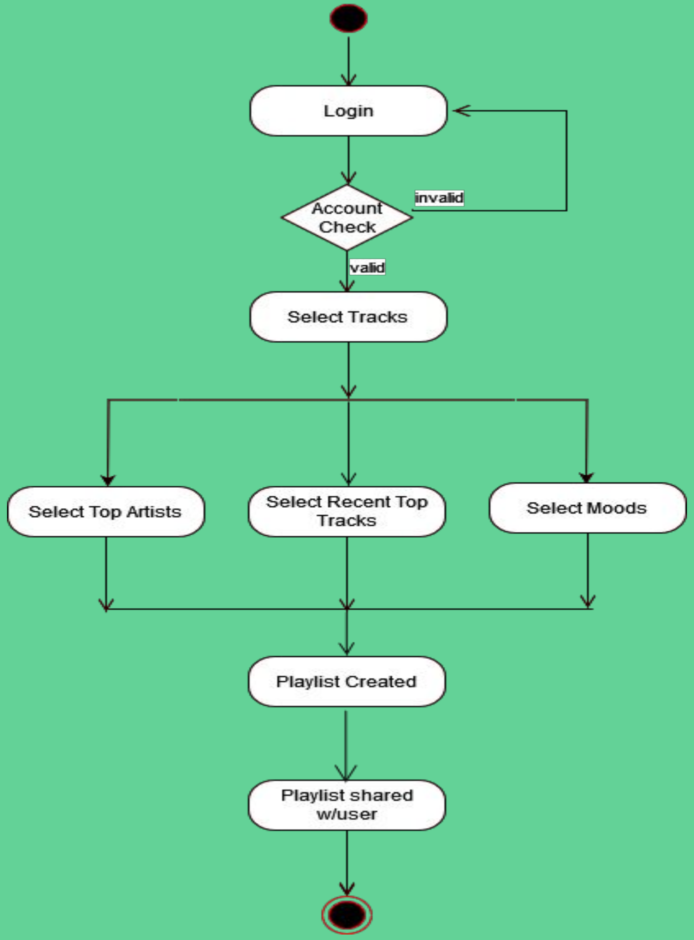


Sequence Diagram

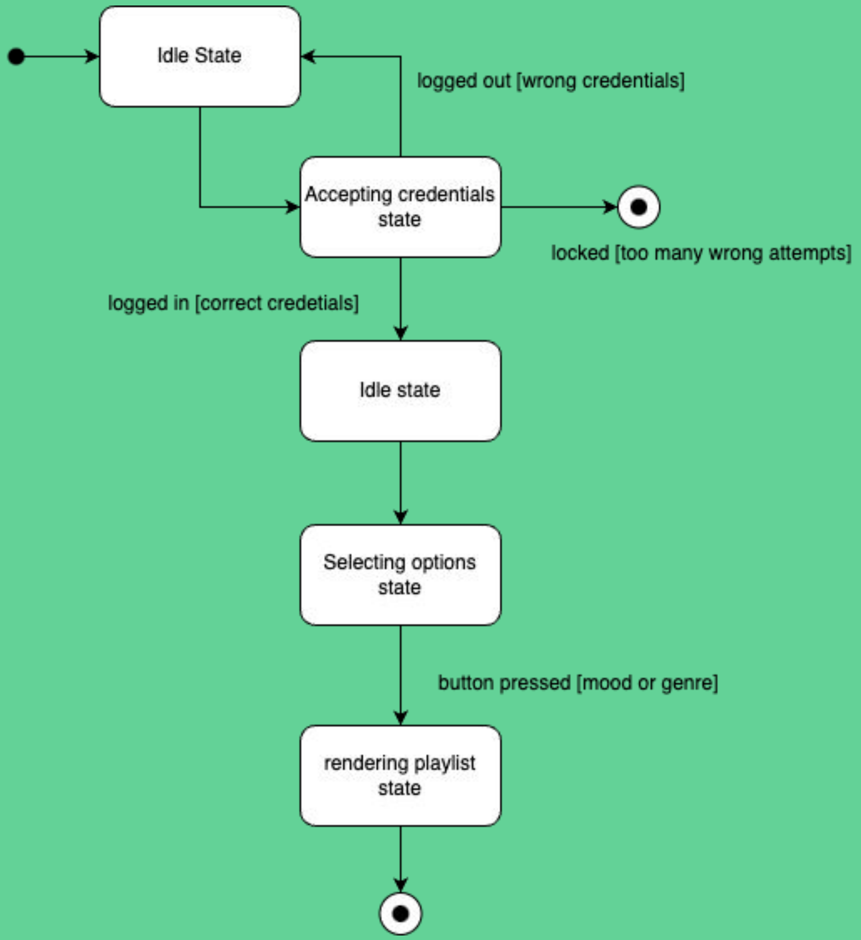
Key
return: Message to user
SM: Social Media
Square: playlist data



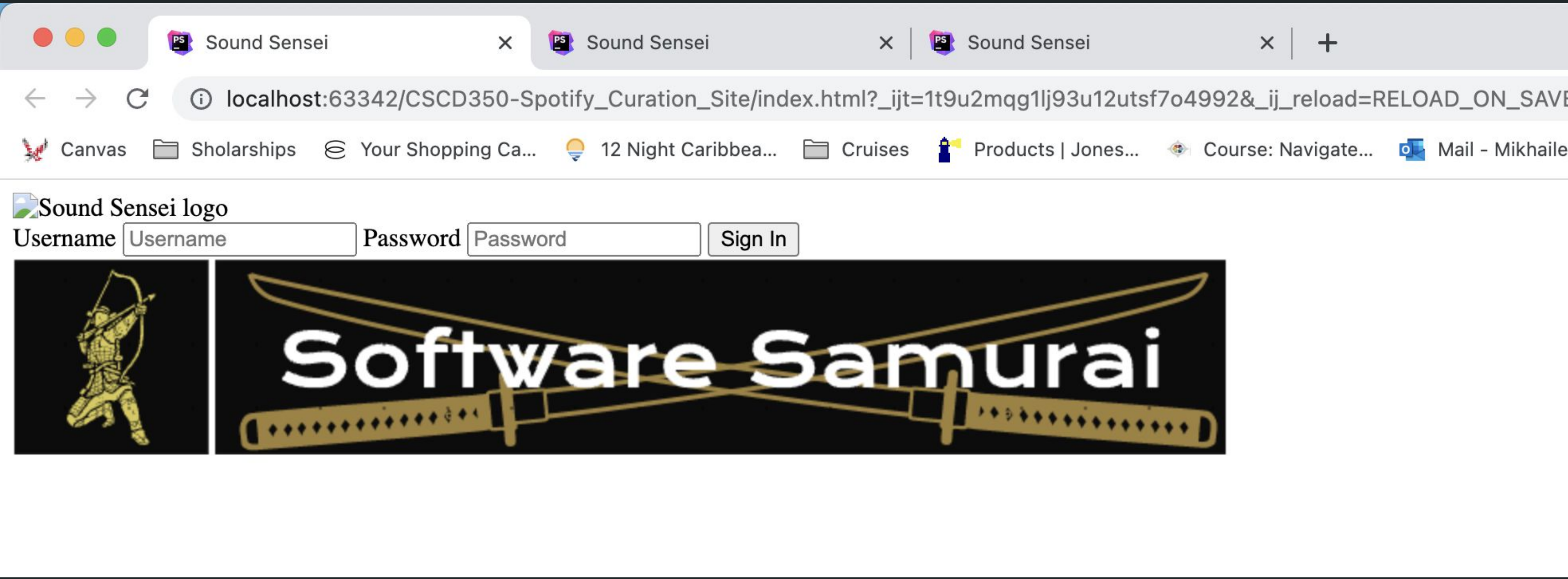
Activity Diagram



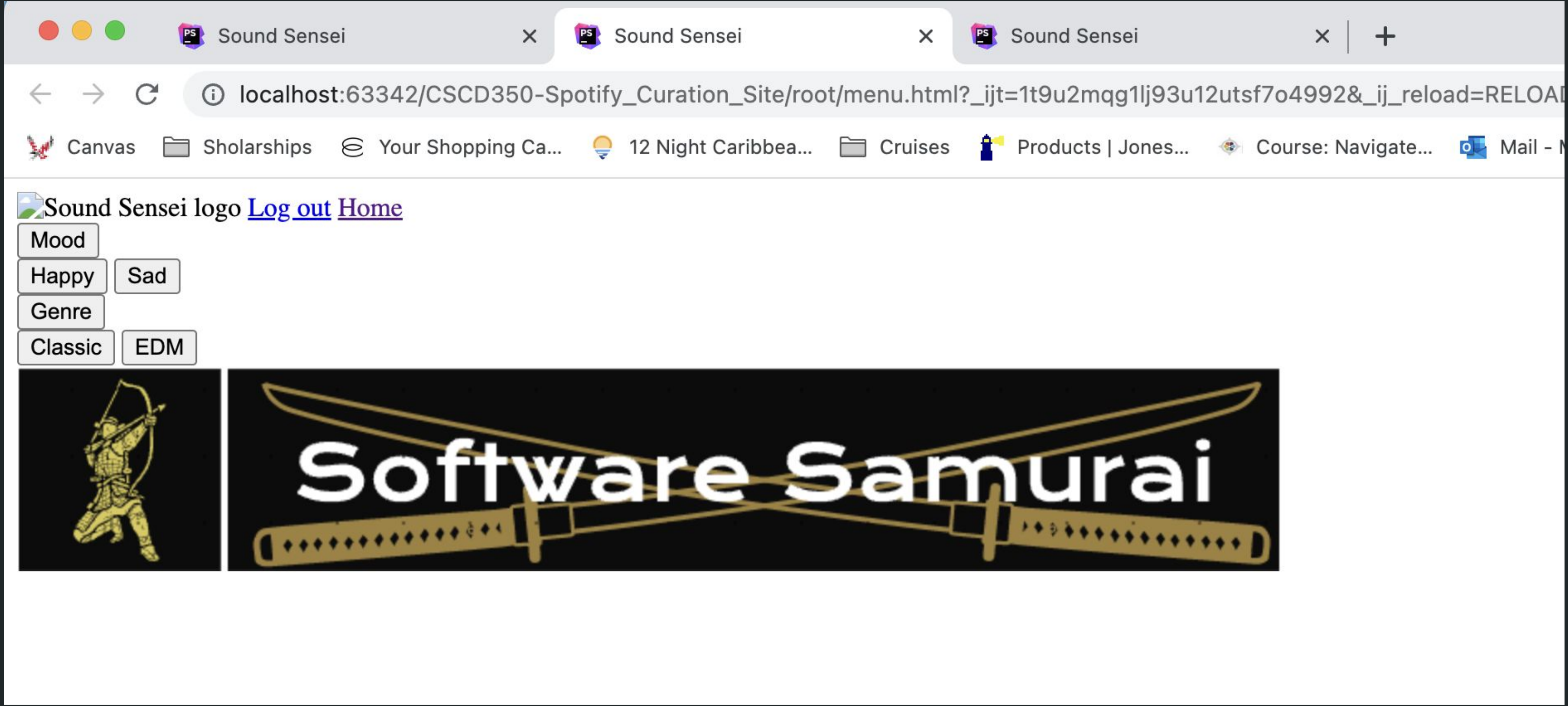
State Chart Diagram


Idle State
logged out [wrong credentials]
Accepting credentials state
locked [too many wrong attempts]
logged in [correct credetials]
Idle state
Selecting options state
button pressed [mood or genre]
rendering playlist state

Web Page Screenshots

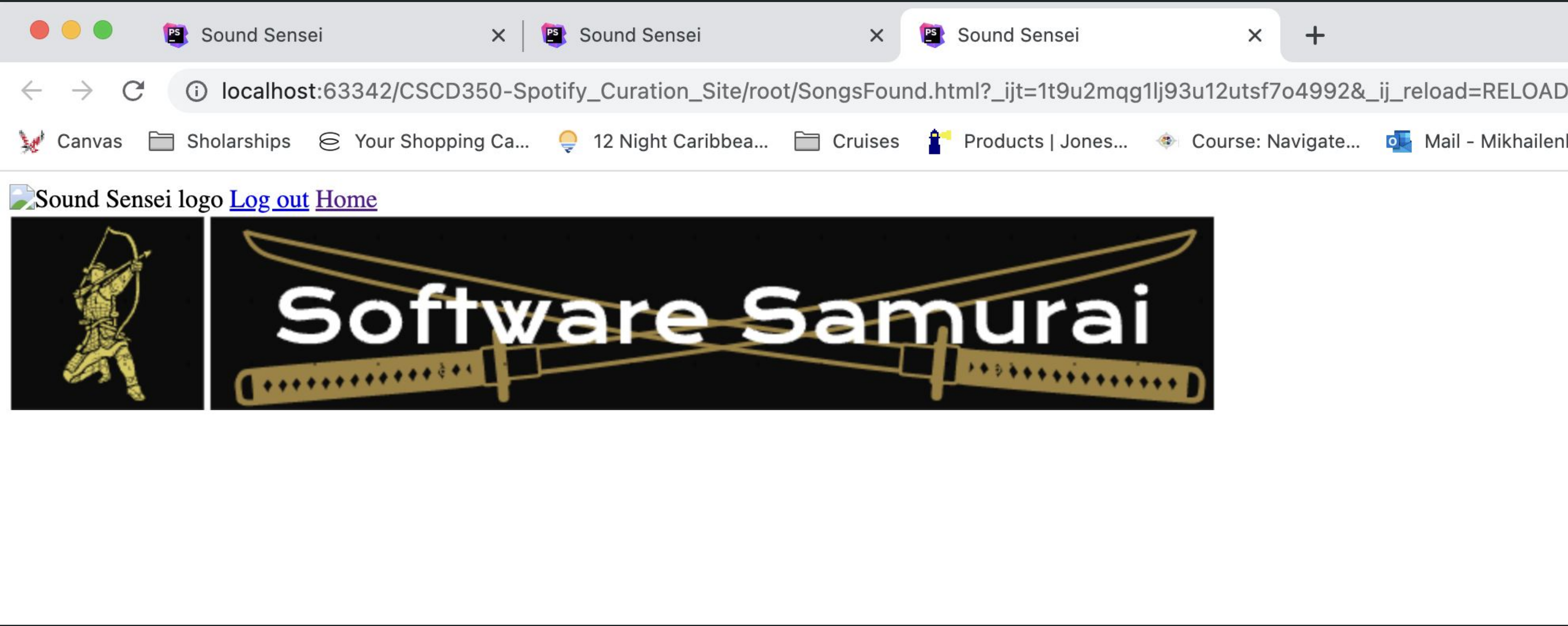


Web Page Screenshots



Sound Sensei logo Log out Home

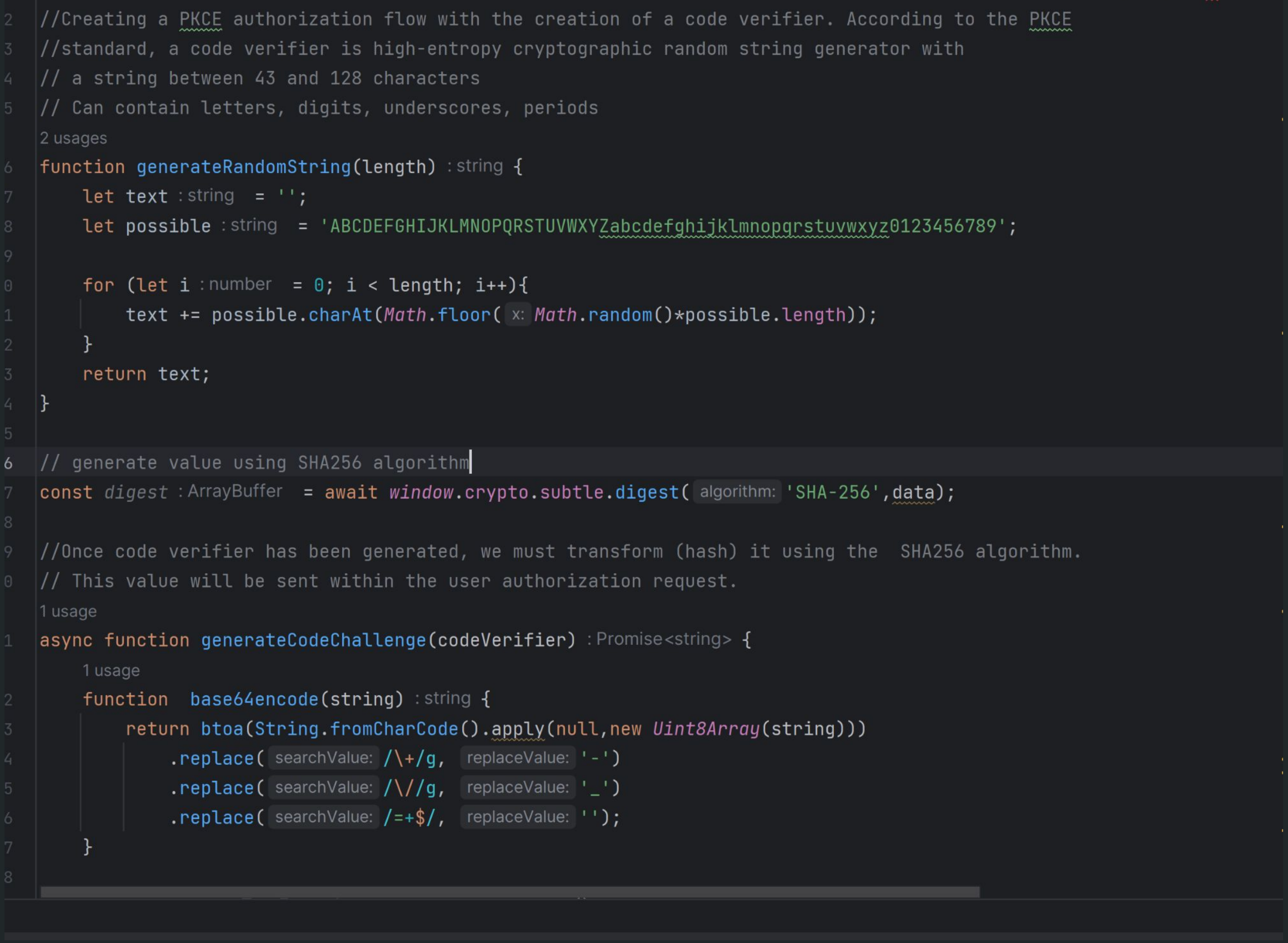
Web Page Screenshots


Sound Sensei × Sound Sensei × Sound Sensei × +

localhost:63342/CSCD350-Spotify_Curation_Site/root/SongsFound.html?_ijt=1t9u2mqg1lj93u12utsf7o4992&_ij_reload=RELOAD

Canvas Sholarships Your Shopping Ca... 12 Night Caribbea... Cruises Products | Jones... Course: Navigate... Mail - Mikhailen

Sound Sensei logo Log out Home



```
//Creating a PKCE authorization flow with the creation of a code verifier. According to the PKCE
//standard, a code verifier is high-entropy cryptographic random string generator with
// a string between 43 and 128 characters
// Can contain letters, digits, underscores, periods
2 usages
function generateRandomString(length) : string {
    let text : string  = '';
    let possible : string  = 'ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789';

    for (let i : number  = 0; i < length; i++){
        text += possible.charAt(Math.floor( x: Math.random()*possible.length));
    }
    return text;
}

// generate value using SHA256 algorithm
const digest : ArrayBuffer  = await window.crypto.subtle.digest( algorithm: 'SHA-256',data);

//Once code verifier has been generated, we must transform (hash) it using the  SHA256 algorithm.
// This value will be sent within the user authorization request.
1 usage
async function generateCodeChallenge(codeVerifier) : Promise<string> {
    1 usage
    function  base64encode(string) : string {
        return btoa(String.fromCharCode().apply(null,new Uint8Array(string)))
            .replace( searchValue: /\+/g,  replaceValue: '-')
            .replace( searchValue: /\//g,  replaceValue: '_')
            .replace( searchValue: /=+$/,  replaceValue: '');
    }

```