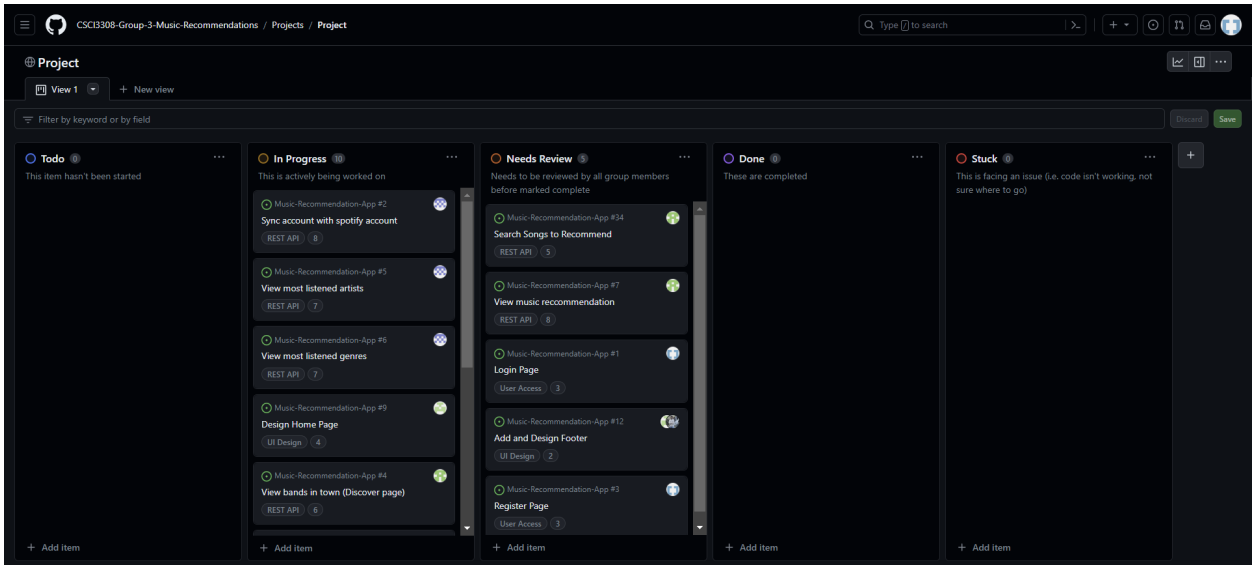
BeatBuddy

Members: Isabelle Godfrey, Ali Haroon, Olivia Hebert, Tyler Onstad-cran, Emanuele Rimini

Project Description:
BeatBuddy is a perfect app for avid music listeners who want to enhance their music experience. BeatBuddy is an app that allows users to see their favorite artists/songs, input songs to find similar ones, and search for artists playing nearby. Users can choose to either register an account with the app or not, which will change what tools are available to them. Users can use the app without logging in but functionality is limited. These users can still find recommended songs but are unable to see top artists/tracks or link their account to Spotify. Users that are logged in can connect their account to Spotify so they can view top artists/tracks, find artists playing nearby, and find recommended songs. The top tracks and artists pages also allow you to check different time periods, in the last 4 weeks, 6 months, or all-time. Users can also search for artists playing nearby and click on events which will take them to Ticketmaster where they can get tickets to those events. Users can also press the fill button to fill the discover page with any events happening soon from their top 20 artists, so they will never miss a concert.

Project Tracker - GitHub project board:

- Link to Project Tracker
- Project board as it was on 11/29:



Video: Video Link

VCS: Link to our git Repository

Contributions:

Isabelle Godfrey: Worked with Chai and Mocha for the login and register test cases in our test branch. Worked with login and register API calls, as well as redirection calls for different pages of our site. Worked mainly on the front-end, changing the appearance and interaction of the header, footer, menu, login, register, home, and profile pages. Worked with the functionality of the site, including ensuring that users who are not logged in, logged in to our site but not Spotify, and logged into our site and connected Spotify have different functionalities that are only accessible to them.

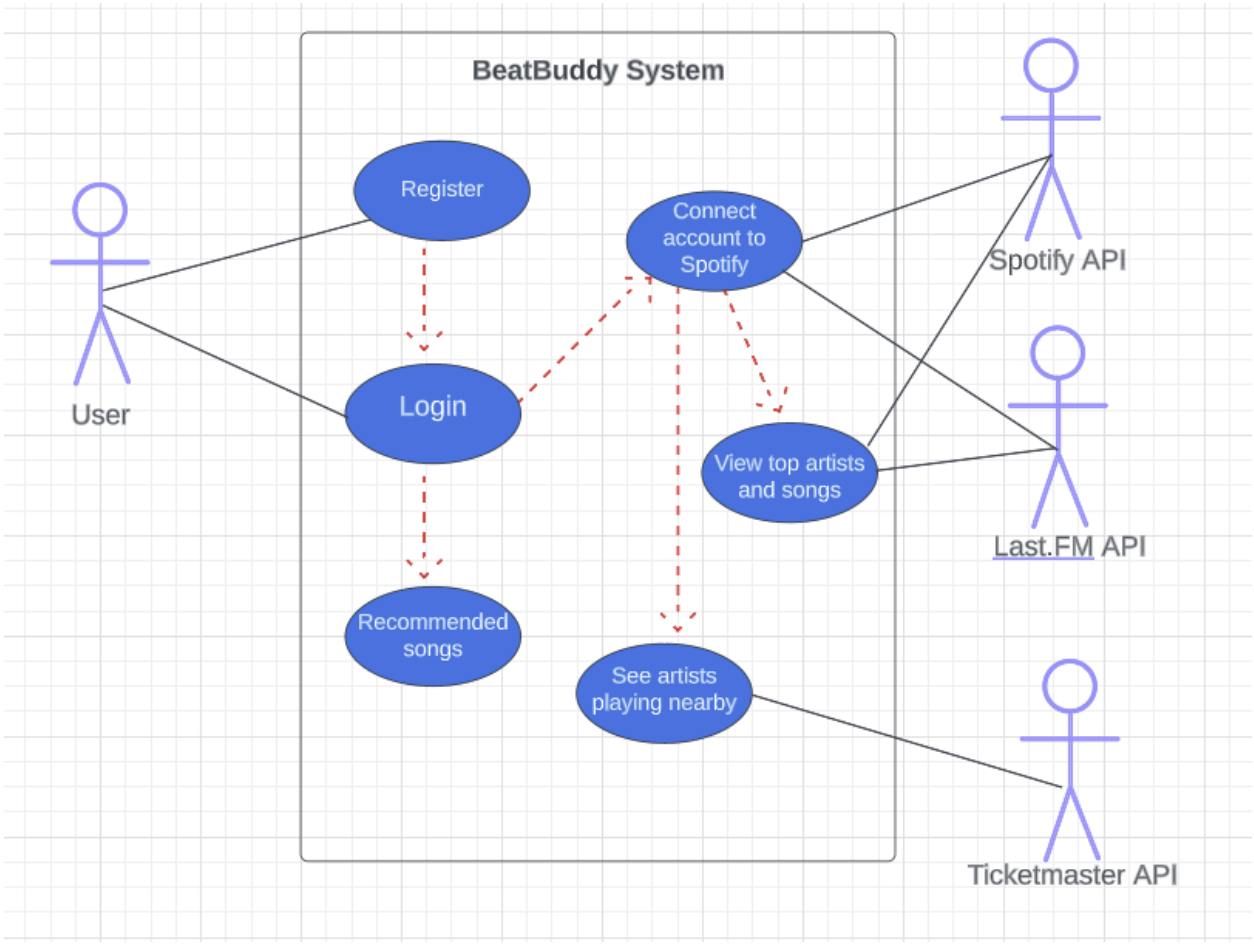
Ali Haroon: Working with the Spotify API, I focused on implementing and testing the 'topTracks' and 'topArtists' routes to enhance our website's functionality. My role also involved styling the website using CSS and Bootstrap, ensuring a cohesive and visually appealing design. Additionally, I managed version control and documentation on GitHub, including the use of automatic release notes and updating previous notes for comprehensive project tracking as we didn't do release notes at the beginning.

Olivia Hebert: I worked on the login and register pages, including the forms to submit login and registration information and error handling. I made the loginFail and registerFail pages which display an error message, for when a user logs in with incorrect information or tries to register with a username that is already taken. I also worked on the User Acceptance Testing and assisted Isabelle with Chai and Mocha testing. I helped with the database and some API routes. I took the TA meeting notes, made the Use Case and Architecture Diagrams, and wrote the user acceptance criteria for each user story.

Tyler Onstad-Cran: Used last FM api and javascript to create a search feature for songs that recommends related music. Using the Spotify API with the users token, populated the top tracks and top artists pages using Axios calls and javascript. Made these calls adjustable to different time scales. Used ticketmasters API and Spotify's API to display upcoming events based on users top artists and their location, as well as a feature to search artists by name. Created a collage feature to map an image that contains all of the users top tracks or artists in a given time period.

Emanuele Rimini: Worked with the spotify api to have the user be able to log in to their spotify account and thus getting us the api key we need for the rest of the site. I wrote the routes to get the top tracks/artists and wrote the discover code, which queries all your top artists and queries the ticketmaster api to see if they have any local events. Also worked on general site structure and linking everything.

Use Case Diagram:



Test results:

FEATURE 1: *Being able to view your top artists in different intervals of time no matter your listening history.*

Case 1: User has 0 listening history with no top artists to fetch

- Since user has 0 listening history there are no artists to display and the pages are blank (for all time intervals)

Case 2: User has less than 10 top artists but more than 0

- Each time period displays however many top artists the user has.

Case 3:User case 3: User has more than 10 top artists

- Initially we were planning on populating the top artists with the top 10 artists but the APIs allowed for up to top 50 so we populated the pages with the top 50 artists (if they have at least 50 top artists). Each time period displays accurate top artists for each time period.

FEATURE 2: *A user should be able to see concerts nearby based off their top artists*

Case 1: User has 0 listening history with no top artists/songs to fetch

- If the user selects the “fill with top artists” the page doesn’t populate with concerts as they have no top artists, it takes them to an error page. If they manually enter information the page populates with concerts matching what they input.

Case 2: User has listening history

- If the user selects the “fill with top artists” the page populate with concerts as they have top artists. They can also choose to manually enter information and the page populates with concerts matching what they input.

FEATURE 3: *A user should be able to login if they have an account or register if they don’t*

Case 1: User has an existing account and inputs username and matching password (login)

- The user is logged into the app and directed to the profile page which is updated with their info (username).

Case 2: User has an existing account and inputs a username and password that don’t match (login)

- The user is not redirected to the profile page as the login failed. Instead an error message pops up and they can try to login again (or register if needed) If they retry with correct information then they are redirected to profile page

Case 3: User doesn’t have an existing account and inputs an unused username (register)

- The user can click on “register here” which redirects them to the registration page where they can input their information, and then they are redirected to the login page to login.

Case 4: User doesn’t have an existing account and inputs a used username (register)

- The user can click on “register here” which redirects them to the registration page, when they put in their information and submit, they are not redirected to the login page. Instead an error message pops up and they can try to register again. When they retry with an unused username they are redirected to the login page.

FEATURE 4: *User should be able to receive song/artist/album recommendations based upon their history.*

Case 1: User has not linked their spotify account

- The user can enter a song name, choose the correct song, and then receive a list of similar songs.

Case 2: User has linked their spotify account, but it has no data

- The user can enter a song name, choose the correct song, and then receive a list of similar songs.

Case 3: User has linked their spotify account, and it has data

- The user can enter a song name, choose the correct song, and then receive a list of similar songs. We ended up not doing recommendations based on listening history due to the functionality of the api's we used.

Deployment: http://recitation-11-team-03.eastus.cloudapp.azure.com:3000/home