Title: Mixify

Who:
Vikki Wong (viwo4708, viwo4708@colorado.edu),
Luke Sellmayer (luse9638, luse9638@colorado.edu),
Justin Jamrowski (juja4571, juja4571@colorado.edu),
Freddy Linn (frli5468, frli5468@colorado.edu)
Lillian Duarte (lidu5648, lidu5648@colorado.edu)

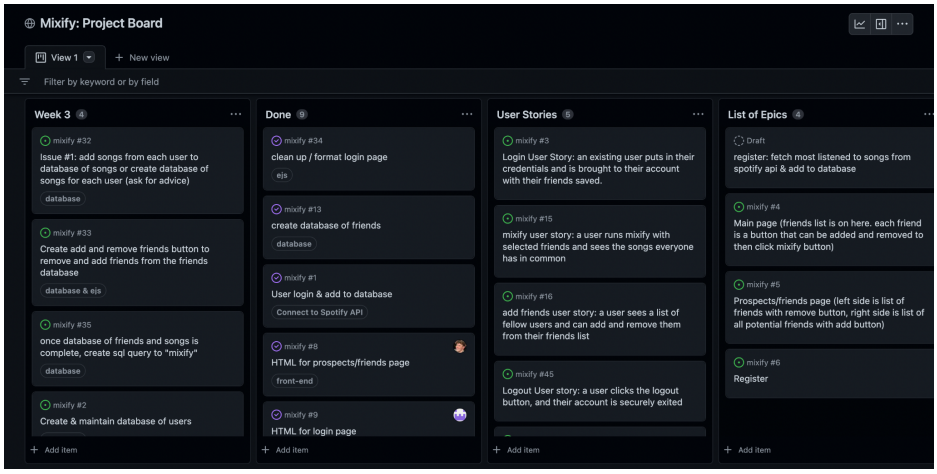
Project Description:
For Spotify users who are curious about what songs their friends love, Mixify is an app that lets you see what songs (most listened to) you and a friend have in common.

Our project, Mixify, is a website that you can use once you log into your Spotify account and allow it to be linked with Mixify. Once logged in and linked, Mixify stores everyone's top 50 most listened-to songs collected from the Spotify API in our backend database. You can add any friend that has logged into Mixify themselves. Then, you can Mix with that friend you have added! Once you Mix, it will pop up the songs you and your friend have in common.

On the backend, we use the Spotify API to pull the appropriate information needed to bring this application to life. This includes spotifyUserID, spotifyDisplayName, spotifyAccessToken, etc; then through GET and POST calls we were able to grab the data needed and display it to our users.

Our goal with this project was to create something that everyone would use and love. To bring people together with the click of a button!

Project Tracker - GitHub project board:
Link to our GitHub project board: https://github.com/users/luse9638/projects/1



Video:
Link to video of application:
https://drive.google.com/file/d/1GeNOdw63iDWNRRGml6HTD-tKNMeRwyoX/view?usp=sharing

The demonstration video is also linked in our ReadMe.md in Github.

VCS:
Link to our git Repository: https://github.com/luse9638/mixify

Contributions:
Lillian: I was the project manager and my main coding responsibility was on the backend working on creating the database tables and making the POST & SQL requests for getting information. I helped keep our group organized / helped direct what people worked on and provided support wherever needed. I also helped with getting the results from the SQL request onto the page using EJS. The main tools I used were NodeJS and SQL.

Vikki:
I implemented the initial connection to the Spotify API in index.js for connecting our application to the user's Spotify account that all the other API data requests were based on. I also did the front-end coding for the homepage, as well as the Mixify results. I also refined the design of the navbar. I created wireframes for the prospects page. The main tools I used were Javascript, HTML, CSS, NodeJS, and Figma.

Freddy:

My contributions were mostly on the front-end and related to UI/UX design. I created the wireframes for the Mixify page as well as the results page after a mix was completed. Afterwards, most of my work was related to the ejs pages. I implemented the home.ejs page as well as the mixify.ejs page, and started the outline for the results.ejs page. This included writing HTML for the content, styling with CSS and Bootstrap, and then tying in JavaScript code through EJS. I also helped contribute to the logout feature and tweaked areas of the partials.

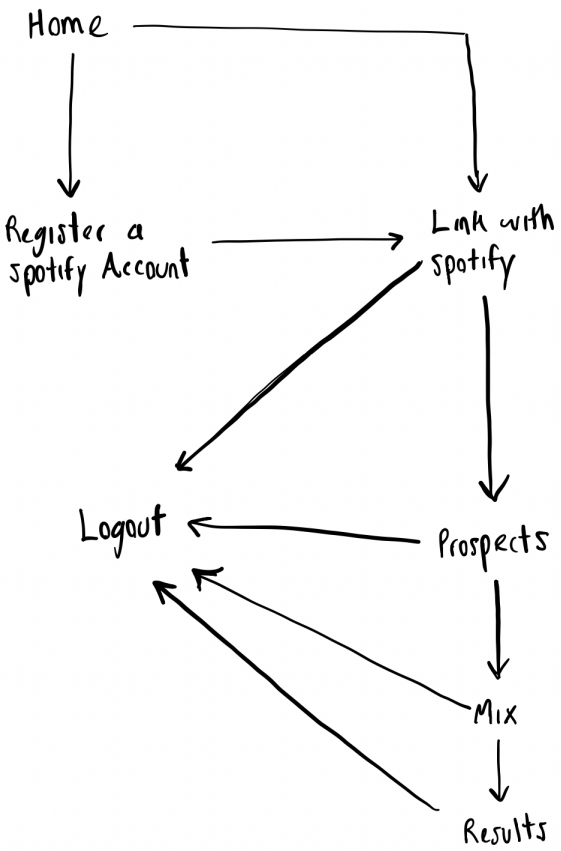
Luke:

The work I did on the Mixify website mainly involved working with the backend and the Spotify API. I implemented the Spotify login process that would allow us to access the user's data as well as actually requesting that data once we had permission to access it. Additionally, I implemented the queries that would store that data in the database so that it could be used by the front end of the website. Part of this included implementing the dynamically updating friends list on prospects.ejs that queries the database to show current and prospective friends.

Justin:

The work I did on the Mixify website consisted mostly of front-end work. Vikki and I implemented the Link with Spotify page, and I implemented the Register with Spotify page if the user does not have a Spotify account but wants to use Mixify. As well as, handling the taking of notes and the submission of meeting minutes and the project report that is done weekly. The technologies I worked with include Bootstrap, CSS, and implementing the link.ejs and registerSpotify.ejs pages.

Use Case Diagram:



Test results:

The test plan we created consisted of testing out our Login with Spotify feature, Register a Spotify Account feature, and Add a friend feature.

Login with Spotify feature

Test Case: Test that the Link with Spotify Button takes the user to Spotify to link their account.

Observations:
What are the users doing? Attempting to link their Spotify account with Mixify.
What is the user's reasoning for their actions? Wanting to use Mixify.
Is their behavior consistent with the use case? Yes

Test Case: Test that after you link your account it will take you back to the main site.

Observations:
What are the users doing? Going back to Mixify after linking their Spotify account
What is the user's reasoning for their actions? Now the user wants to use Mixify
Is their behavior consistent with the use case? Yes

Register a Spotify Account feature

Test Case: Clicking the button will take you to Spotify's website to create a Mixify account

Observations:
What are the users doing? Going to Spotify to create an account.
What is the user's reasoning for their actions? The user does not have a Spotify account already, so this makes it easier for them to create one than to open a new tab and search to make one.
Is their behavior consistent with the use case? Yes

Adding a friend feature

Test Case: Clicking the "add" button next to a username correctly adds that user to our friend lists in the database

Observations:
What are the users doing? Adding their friend on Mixify
What is the user's reasoning for their actions? The user wants to add a friend so they can use the website for its purpose of "Mixifying".
Is their behavior consistent with the use case? Yes

Test Case: Clicking the "remove" button next to a username correctly adds that user to our friend lists in the database

What are the users doing? The user wants to remove a friend on Mixify.
What is the user's reasoning for their actions? The user may not be friends with that person anymore or accidentally added them.

Is their behavior consistent with the use case? Yes

Deployment:

To run it locally:

1. Start docker-compose up
2. Open two browsers
3. Go to the local host in each browser and log into Spotify
4. On whichever browser, go to the prospects page and add the other user who logged in on the other browser
5. Then go to Mixify and hit mix!