Project Report for Team 014-5, Jicama B

CSCI3308: Software Development Methods and Tools

Group Members:

James Nguyen

Cara Wang

Benedikt Safigan

George Fisher

Mason Chansamone

Table of Contents:

Project Title
Spotigang

Project Description
Spotigang is a social feed platform with a focus on sharing opinions about different songs. Users are able to create an account on the Spotigang website and post reviews about songs, along with updating their profile in various ways, such as profile pictures and status messages (inspired by Instagram Notes) that can indicate the song a user is currently listening to. Additionally, users can send friend requests to each other, being able to see their review posts and status messages. By posting a review about a song that includes a song name, rating, and text body for the post, other users can view a home feed that allows them to like and dislike posts, along with commenting on these posts to continue discussion.

The core of this application's music theme is utilizing the free Spotify API. Setting a "Listening To: {song}" as your status message, along with posting a review on a song, requires selecting a song, where a drop down menu is displayed when searching for a song that utilizes the Spotify API. These features require a song from the Spotify API to be chosen, ensuring that the social aspect is intertwined with the music-opinion aspect. Future improvements involve utilizing this concept more- for example, being able to see what a user is listening to in real time utilizing a linked Spotify account, better Spotify metadata about songs/albums on reviews, being able to message friends, and an exclusive feed with reviews specifically from friends.

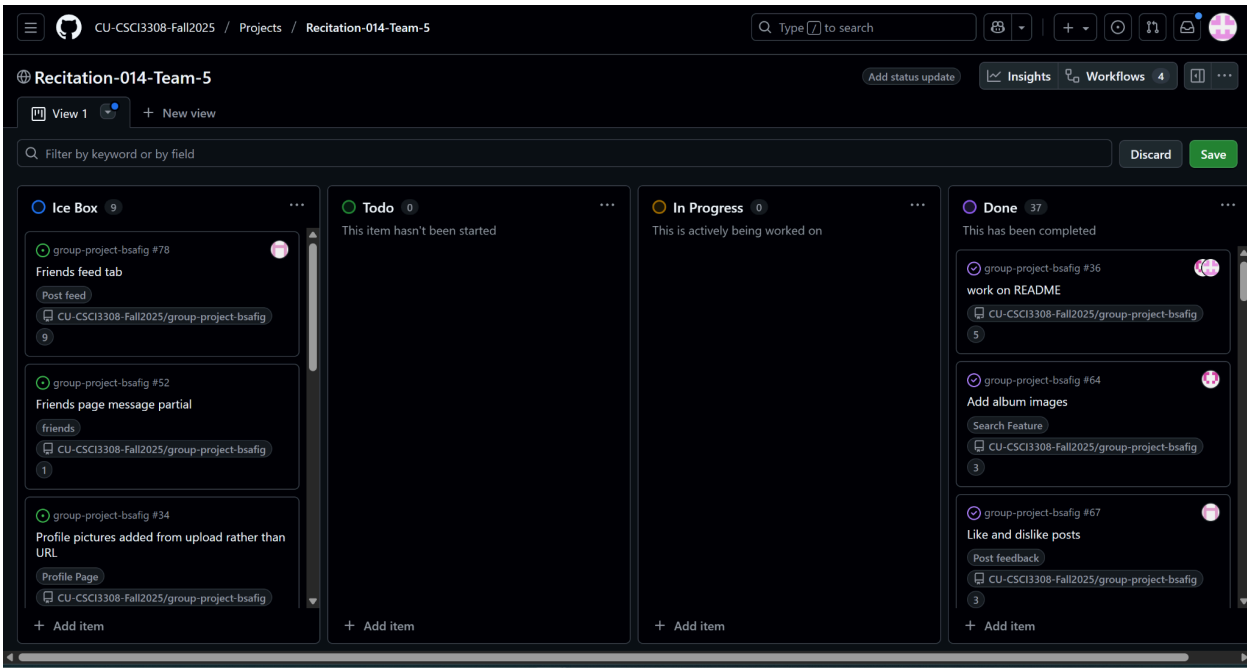
Git Repository & Project Tracker & Demo Video

- Link to GitHub Repository:
    - https://github.com/CU-CSCI3308-Fall2025/group-project-bsafig

- Link to GitHub Project Board Project Tracker:
    - https://github.com/orgs/CU-CSCI3308-Fall2025/projects/17/views/1?filterQuery=

- Link to YouTube Video displaying a demo of Spotigang:

    - [CSCI3308 Spotigang: Demo Video]

- A screenshot of the project in the GitHub Project Tracker. Any items still in the Ice Box are additional features that, if time permitted, would be in our expanded scope.
    - 

Contributions

Individual Group Member Contributions:

- **James** primarily worked on backend Node.js endpoints, including the beginnings of the registration and login pages, the core functionality of the profile and settings pages, the full stack creation of the status message, and linking the Spotify search functionality created by Cara to include proper enforcement for Post and Status creation. James also worked on the README and project logistics, including writeups/documents, Render deployment, and weekly release notes.
- **Benedikt** worked on the initial creation of the PostgreSQL database, along with the necessary pg-promise, containerization, and index.js setup (middleware) for the start of the project. Ben also worked on the full stack creation of the friends page, including the ability to send, search, and cancel friend requests. He also

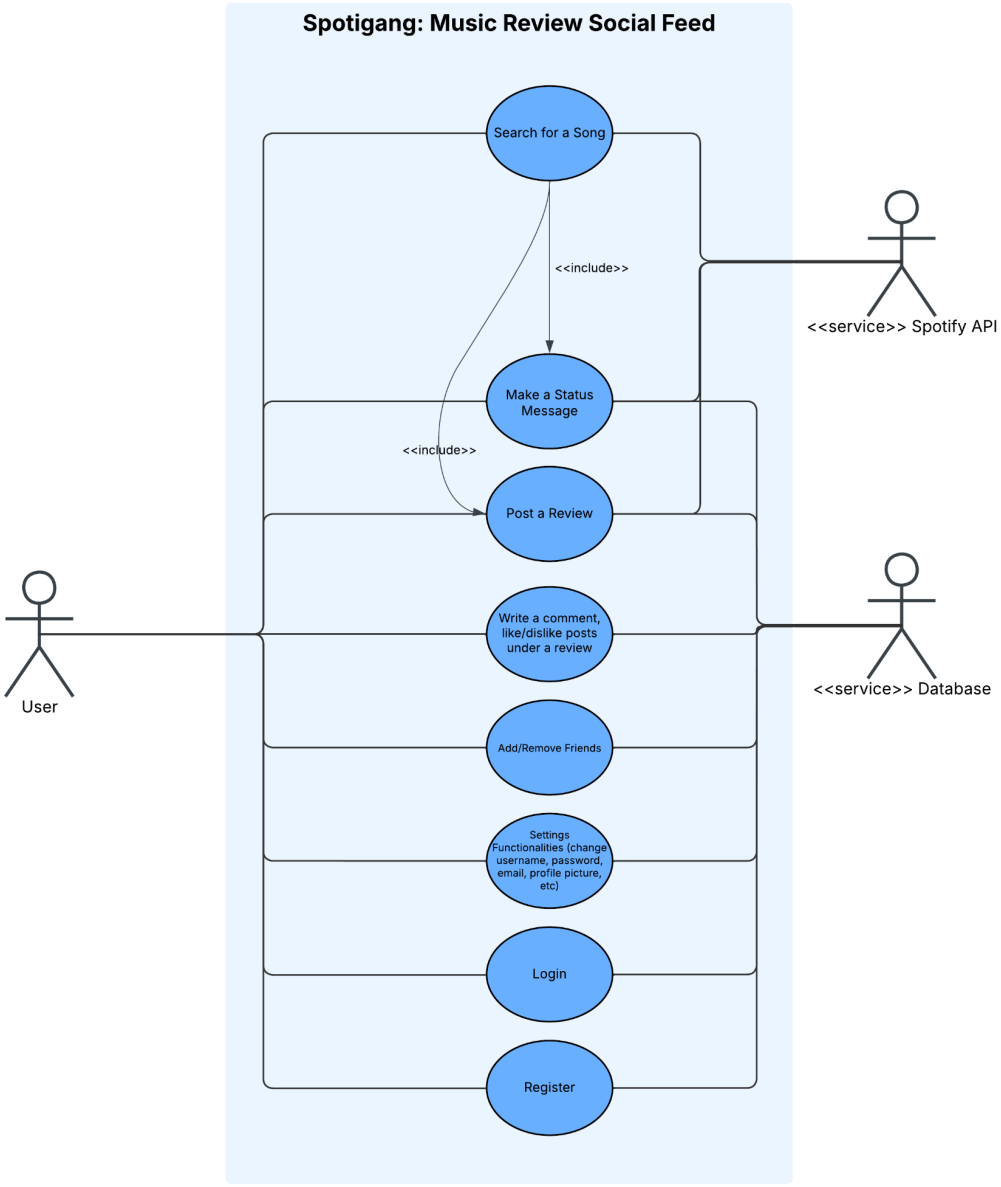
worked on adding created posts onto the Home page, along with all of the comment and post reaction (like/dislike) functionality.

- **Cara** worked on the the frontend for both the Profile and Settings pages, along with creating the necessary partials for pages like the navigation bar and footers. Cara also worked on the post functionality, changing the originally partial post into a full separate page. She also implemented the usage of the external Spotify API for posts and status creation. Finally, she also fixed dark mode to be fully functional.
- **George** worked on the implementation of the friends page and friends page, working on displaying the list of friends. George worked on displaying the list of friends, along with being able to view the profile pages of different friends. This also involved being able to display the list of other profile's friends. George also worked on the initial development of Dark Mode, which Cara finalized the implementation of.
- **Mason** worked on the backend of profile, settings, and posts. Mason worked on showing posts and the friend count in the profile page properly, fixing core bugs with proper display counts, along with adding extra functionality (update email) to the settings page. Mason implemented the deletion and updating of posts, along with visual clarity through removing unnecessary visuals and buttons for the registration pages.

Use Case Diagram

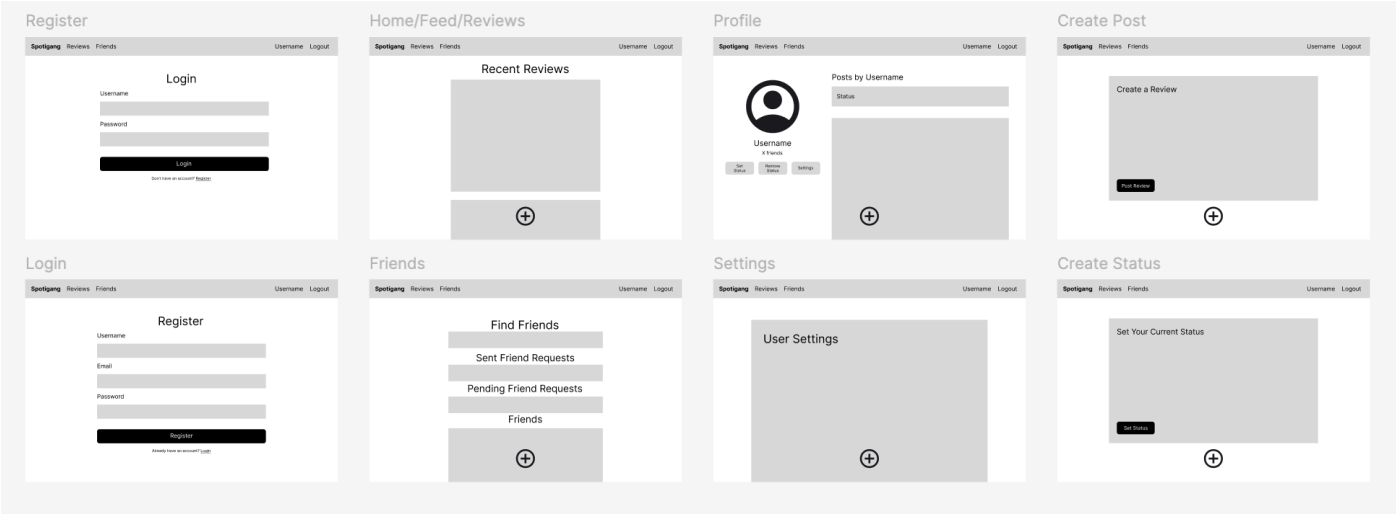
The use case diagram utilized in the initial proposal (Lab 8) is now out of date for the scope of the project. The new use case diagram below contains core changes, especially with the amount of functionality. The main feature comes from the fact that the Spotify API is solely used to search for songs in the posts/status creation, which reduced the scope/ability of the Spotify API as the project progressed; on the other hand, there are

many new features our first UML diagram did not initially cover.



Wireframes

Additional wireframes were created to include the creation of new pages as the scope of our project grew.



Test Results

We had two testers for testing our UAT plan. Both testers were other CU Boulder students with a non-cs (but engineering) background, and were not related to CSCI3308. We provided them with the test cases provided in the UAT plan, providing them necessary data if needed, like account info. The test results described are generally what we saw between both of the testers, consolidated into one set of observations. Additionally, to have further testing on top of the UAT plan described in our Lab 10 writeup, an additional test was created for the purposes of the first step: logging in. The following cases tested are described below, and the observations associated with each step.

1. Logging In

- *Test Data:* Using a test account called ‘test_account’ with password ‘123’
- *Expected Results:* No error should occur when logging in. The user should be redirected to the home page with the navbar and footer present.

We observed little issues when using the login page. As the default Spotigang app redirects to the login page, it is self explanatory for the user to enter the information we gave them,

and click the login button. When testing with a password that does not work, the user was unable to login (expected). Otherwise, the user was then redirected to the home page, and the footer (+ button to create a post) was generated.

2. <u>Sending a Friend Request</u>
- *Test Data*: The user sends a friend request to an existing user, typing in the letter 'a' should show the user 'abc'.
- *Expected Results:* No error occurs. Alert occurs that informs the user that a request was sent. The request will appear in the 'sent requests' section.

The users were successfully able to navigate to the friends page. When seeing the large "Find Friends" button, it was intuitive for the user to search for friends. The users successfully searched up 'a' and found the user 'abc', sending them a friend request. The alert that shows to inform a user that a request was sent was slightly confusing for the user experience, as it is directly followed by the page re-rendering to show the Sent Friend Requests, which can be confusing when it seems like the page didn't visually change much. If this were an app with more of a focus on frontend and UX development, then adding a better or smoother indicator for friend requests would be helpful.

3. <u>Accepting a Friend Request</u>
- *Test Data:* The user logs out of 'test_account'. The user will login to the account 'abc' with password '123', navigate to the friends page, and accept the friend request.
- *Expected Results:* Visually, the request should exist in the 'pending requests' section. No error occurs. Alert occurs that informs the user that the request was accepted. User should now appear in the active friends list. If a status is assigned to the friend, it should show the status.

The users had no difficulty with logging out of test_account and logging into abc. The users navigated to the friends page, and accepted the pending friend request. Similarly, the typical JavaScript alert was once again slightly confusing and not helpful for the user experience. Other than that, there were no bugs present, and the friend 'test_account' was now present in the active friends list.

4. <u>Posting a Review</u>
- *Test Data:* The user clicks the + at the bottom of the screen (footer). The user enters a song, and selects it from the Spotify API integrated search. The user posts a rating between 0-10. The user clicks post.
- *Expected Results:* No error occurs. The post appears under the Home feed as the most

recent post. The post appears under the user's posts in the Profile page.

The users found the + button as the footer a bit unintuitive- while a plus symbol is a pretty typical button for the creation of a post, without enough context of what the app was, it was easy for the users to not really consider what the plus symbol would do- and whether or not it was even a button. This is an issue with communication about *what* the app is within the home pages or even before logging in, and also with the user experience design. After opening the Create Post page, the users were slightly confused by the initial server delays of searching for a Spotify song. Other than that, the rating and review textboxes had no issues, and when posted, the post successfully showed under the home feed and the user's posts in the profile page.

Deployment

Link to the Render application for the deployment of Spotigang:
https://spotigang.onrender.com/login