CSCI 3308 - LuckyMoment

Project Members:

- Matt Schneider
- Alec Volkert
- Nina Vo
- Vivian Diep
- Varun Cheela
- John Kreye
- Daniel Zhu

Project Description:

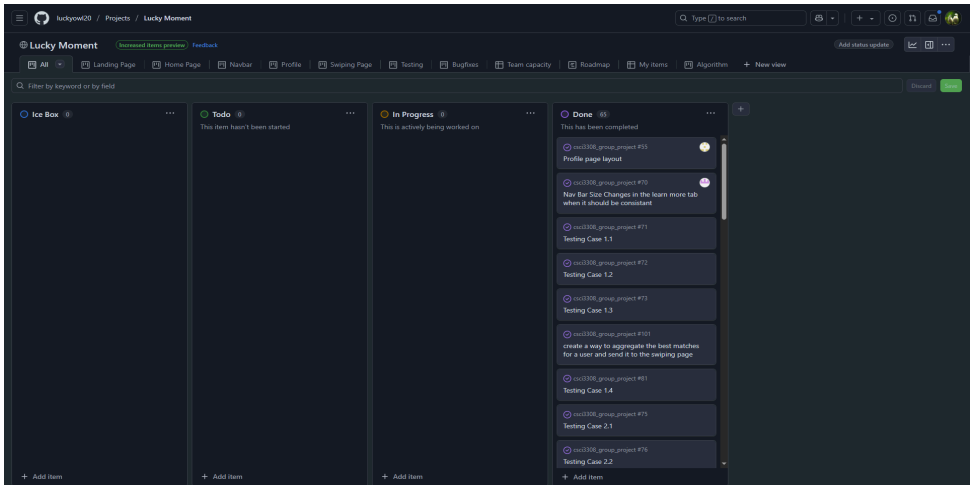
Lucky Moment is a dating and social platform designed to promote authenticity and prevent catfishing. Each day, users have a five-minute window to capture and upload a real-time photo from their webcam, giving a genuine glimpse into their daily life. To access features like swiping, messaging, and viewing posts, users must first post their daily photo.

The app matches users based on shared interests, location, and activity, offering real-time chat between matches and a daily feed of updates. You can explore new music, restaurants, and activities through the discovery page, and view potential matches' posts from the past seven days. Profiles showcase interests, a favorite song, a short bio, and more, helping users connect beyond appearances.

Built with Node.js, Express, Handlebars, and PostgreSQL, Lucky Moment also integrates Spotify and Google Maps APIs for an enriched experience. It’s hosted on Render — please allow a few minutes for the app to start if inactive.

Project Tracker - GitHub Project Board:

https://github.com/users/luckyowl20/projects/1/



Screenshot of project in project board:

Video

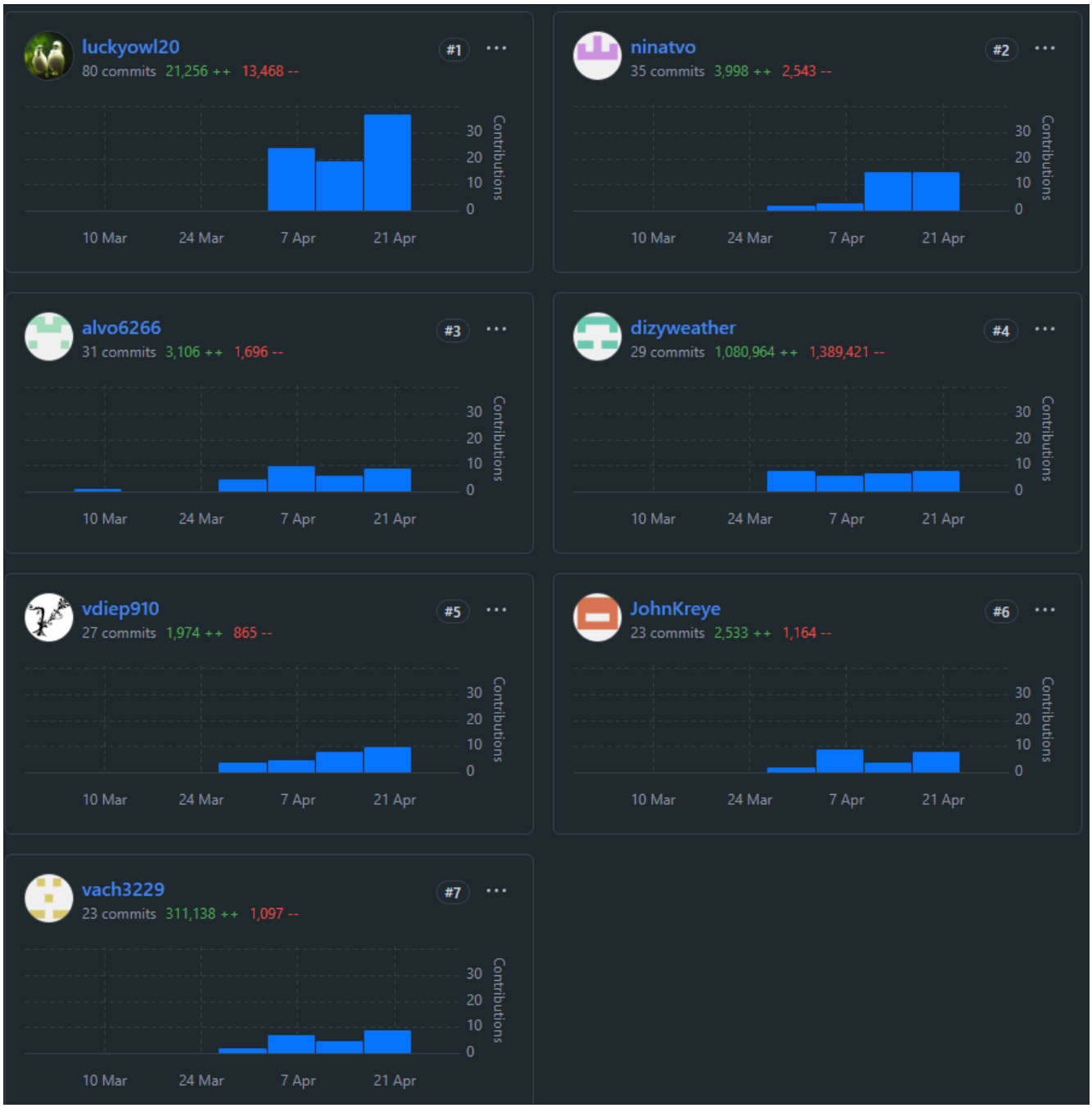
https://drive.google.com/file/d/1niDP4HJUHGEGzDtc0sD3GlJZtRuuM-Eg/view?usp=sharing

Version Control System (VCS) - GitHub

Project Repository on GitHub:
https://github.com/luckyowl20/csci3308_group_project

Contributions

GitHub Insights page for this project, includes commit history and activity.
https://github.com/luckyowl20/csci3308_group_project/graphs/contributors
Screenshot of contributions page, contributions over the duration of the project to the main branch, excluding merge commits:



Individual Contributions

Matt Schneider - luckyowl20/Matt

- Built chat page, profile page, edit profile, preferences, landing page, database schema, back-end matching algorithm, and dummy data generation script. Led styling efforts for the whole project, moved styles to css-sheets. Bug-fixing for swiping pages, discover songs page, register/login pages/warning messages. Set up test cases for lab 11, deployed the project for lab 13. Set up/maintained project directory structure, ensured consistent organization for the duration of the project (routes/css). Reviewed/approved most pull requests for new features and fixes. Used Spotify API, socket.io, PostGIS (PostgreSQL extension), Python web scrapers for dummy data images (selenium).

Alec Volkert - alvo6266

- Built the take a picture element. Incorporated it to work with each individual person's system that would pull from our database. Implemented a take profile picture feature. Implemented the supabase api. Corrected small bugs that were found within different pages. Helped build the presentation. For the phone implementation I styled pages to make sure they would work correctly on a user's phone and users computer. Implemented new features on the take picture and take profile picture so that when a user was on their phone the camera could flip.

Nina Vo - ninatvo

- Designed the basic UI framework for the website that was further polished into a global base CSS file and refined into page specific CSS files. Worked on the Explore Music page that incorporated the Spotify Web API I-frame player. Swiping functionality was added so that the user could like/dislike the song, and their opinion was added to a music table in the database along with the ID of the song. Search function was added so users could search for songs from a specific artist.

Vivian Diep - vdiep910

- Made the basic working About page with all the information and catchphrase. Made and styled the swipe function, the meet someone card, and the meet a friends card, and connected them to the user database. Connected the swipe and matching to the matching algorithm and made sure that when users match, they get added as a friend or a match. On the swipe page, added a carousel so users can view the latest week of posts

from the profile they are swiping on. Also created the LuckyMoment logo and helped with the presentation for the future enhancements.

Varun Cheela - vach3229

- Built the original landing page, feed/home page, explore-activities page, and memories/week view feature. Developed the feed so friends can view uploaded moments from friends and matches, with filters for completed profiles. Created the memories view to browse past moments. Integrated the Google Maps Places API into the explore-activities page to show local activities based on user location. Implemented like button functionality for activities and posts, updating the SQL database accordingly. Contributed to early foundational structure and styling for the project

John Kreye - JohnKreye

- Built the settings page, implemented the settings in the sql database, setting changes are deployed upon being changed thanks to handlebars. Fixed an issue with the chat where socket.io was sending messages to all active users and not just the target. Worked on blog page, used json_agg and json_build_object along with many joins, groups, and other sql statements to select database entries in a complex way necessary to achieve desired results for the blog page, used handlebars extensively to display blog data on the page. Finalized CSS styling and helped get the webpage ready for mobile usage.

Daniel Zhu - dizyweather

- Helped set up Mocha & Chai for testing. Developed the Explore Restaurants front-end with non-page-resetting like and dislike buttons on each restaurant, which updates the database. Used Google Maps Places API to query and display said restaurants near the user. Helped review pull requests, fix merge conflicts, and test branches.

Use Case Diagram

https://i.postimg.cc/fTZM0b5M/Screenshot-2025-04-28-at-10-16-28-PM.png

Wireframes

Figma of our layouts/wireframes for the website

https://www.figma.com/design/c6YNpYc8Zzvoo1Ec4WfsKJ/CSCI-3308-Project-Wire?node-id=0-1&p=f&t=UBhedqTaQkjkiRBp-0

PNG Version:
https://o365coloradoedu-my.sharepoint.com/:i:/g/personal/vidi3457_colorado_edu/EWltTXueSXBJkeIYJnEvM9gBMoy8noH0CSZq6ZFRWELSDQ?e=dbxbOn
(To zoom in, click the three dots and “view original”)

Final Database Schema Image
https://ibb.co/YBqsnjcg

Test Results

Results are included in the UAT plan listed beneath every test that details how each test was conducted and the expected results for each test.

https://github.com/luckyowl20/csci3308_group_project/blob/main/Milestones/lab11-TestingMilestone.md

Feature 1: Live chat functionality

- Test 1.1 Successful message send/receive
    - Results: Test case passed clearly with no issues users were able to send and receive messages without any issues.

- Test 1.2 Message sent while receiver is online
    - Results: This passed perfectly fine users were still able to send messages even if the other user is offline.

- Test 1.3 Sending an empty or whitespace message

    - Results: This worked, no messages that had just pure whitespace were able to be sent between users.

- Test1.4 Message sent to non friend (disallowed)
    - Results: This test fails partially. User is able to navigate to chat/(user id) and send messages to that user they are not friends/matched with. User is not able to send messages to themselves.

Feature 2: Profile editing

- Test 2.1 completely update profile with new information in each field
    - Results: All profile updates work as they should and the data is stored correctly as it should be.

- Test 2.2 Partial update (just changing bio)
    - Results: This works correctly the user can update any part of the profile that they choose and no issues will arise with other profile settings or features.

- Test 2.3 Invalid/empty inputs
    - Results: We do not allow profile pictures to be submitted as URLs, thus test case passes. URLs come from supabase database. Similarly, all empty fields are handled correctly

- Test 2.4 Unauthorized edit attempt
    - Results: No editing button shows up if users are on another user's profile meaning that this test case passes and this feature works as it should.

- Test 2.5 Profile reload reflects changes
    - Result: Data is pulled from the database and after a reload the data does stay on the profile meaning this test passes.

Feature 3: Spotify song search and selection on user profile

- Test 3.1 Successful search and selection
    - Results: User is able to search, select, and save a track. Meaning this test passes.

- Test 3.2 Empty search
    - Results: There are no results when the user has not entered any value in the search bar for the song, so this test passes.

- Test 3.3 Search with no results
    - Results: There are no search results from spotify that do not result in some type of result. This this test case somewhat fails, although it is out of our control

- Test 3.4 Song selected but modal not submitted
    - Results: The modal must be submitted otherwise the song will not be preserved so this test does pass.

- Test 3.5 Edit existing song selection
    - Results: The user is able to change their previously selected track to any track they choose so this test does pass.

- Test 3.6 Profile Load With No Song Selected
    - Results: A message saying no favorite song selected pops up so this test does pass.

Deployment

Deployed website/app: https://csci3308-group-project.onrender.com/

Our app was deployed using the deployment service Render. We deployed our website by following the instructions for the deployment lab and successfully created a database instance and deployment environment for our code to run on, within Render.
To access our application, visit the link above and start creating your account by following the instructions on the landing page. Depending on the time of access, the initial photos created for our demonstration during class presentation time may no longer be available for viewing. This is due to the fact that each set of photos only displays on the homepage for the current day and is then only visible through viewing the profile of your friends. Photos of other users are visible for a week at most, the dummy users “user(number 1-25)” with password “dev” can be used to test the functionality of our application. For example “user1” was used in the demonstration. Alternatively, you can create an account as described on the landing page.

Dummy data timestamps have been reset as of 4/27/25.