Skillswap Project Report

Who:
Jake Huebner (JakeH4), Annika Polavarapu (annika-polavarapu), Evan Mohan (ProofLion), Anthony Liu (Anthony1503), Gavin Unrue (O11WL1D)

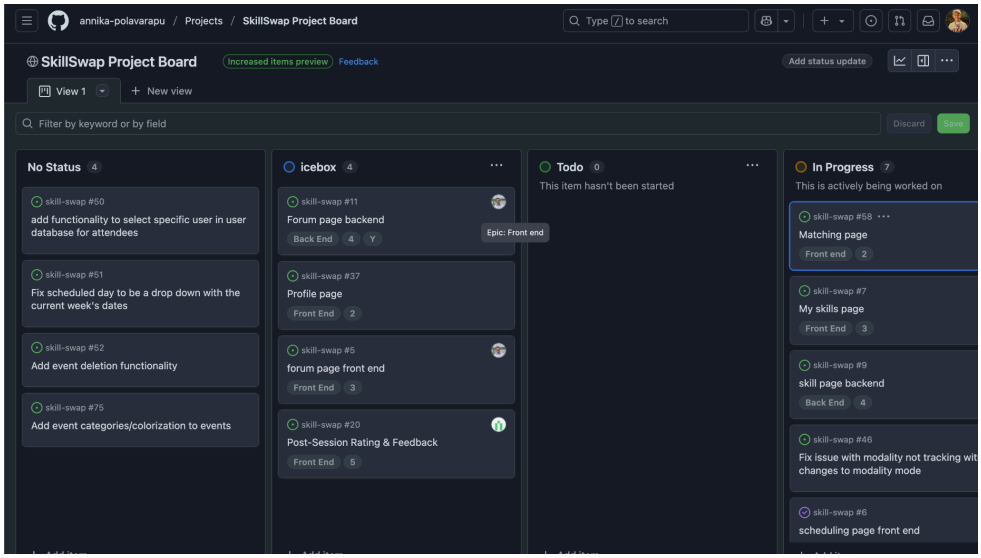
Project Description:

SkillSwap is a web application that helps people connect based on the skills they want to learn and teach. After creating an account, users can set up a profile with their skills and interests. From there, they can customize their profile by adding skills that they have and a profile picture. After that, the user can search for a skill that they want to learn and the app matches them with users who can teach them their skill while learning theirs.

The app features a matching system, private messaging, and scheduling tools, all wrapped up in a clean interface with both light and dark modes. Users can easily browse potential matches, view basic profile info, and start conversations to plan skill swaps.SkillSwap was built with Node.js, Express, Handlebars, and PostgreSQL, and is styled with Bootstrap and custom CSS for a polished but simple feel. We containerized everything with Docker to make it easy to run and deploy anywhere.

Our goal was to create something practical, user-friendly, and rewarding in which users can share their knowledge, build skills, and make real connections.
WC: 183
Link to render (also in README):
https://skill-swap-webapp.onrender.com/

Version Control:
Link to repository: https://github.com/annika-polavarapu/skill-swap
Screenshot of project tracker:



Link to project board:
https://github.com/users/annika-polavarapu/projects/1/views/1

Link to demo video:
https://github.com/annika-polavarapu/skill-swap/blob/main/projectLogs/skill-swap-video.mp4
(inside project logs folder)
OR
https://girlbossesproject.slack.com/files/U08HXT0D0DC/F08Q54YTP3L/skill-swap-video.mp4

Contributions (less than 100 words each):
Jake:

I worked mainly on the front end development. One thing I was surprised at was the difficulty of the implementation of night mode, which changes the style of the page to a darker, more relaxed mode. It was easy to create at first, but it slightly broke a lot of small things and created some unfriendly use cases for the users. Therefore, I got a lot of experience with css and styling. I also enjoyed messing with the dynamic, interactive features that made the app feel more modern and smooth.
WC: 90

Annika:

I focused a lot on building out the login, logout, and registration features, as well as setting up the backend authentication. Getting sessions to stay consistent across different parts of the app turned out to be way more detailed work than I thought it would be. There were lots of small issues that popped up when trying to make everything connect cleanly, but it helped me really understand how authentication flows work behind the scenes. I also helped keep things organized across the backend, making sure the routes and database stayed clean and easy for everyone else to use.

WC: 95

Evan:

I mostly worked on the back-end implementation of all the pages except Scheduling, login/logout and register. One thing I was surprised with, was once I had the correct code for messaging users, implementing Socket.IO was really easy and I would've liked to implement it on more pages but I ran out of time. The most annoying thing probably was getting the database to work correctly with users registering skills on the profile page, and matching with users on the matching page, it took a lot of SQL troubleshooting to get it working the way it was intended.
WC: 99

Anthony:

For my part, I mainly worked on the navigation bar dropdown, logout functionality, the profile page frontend, and making the site look cleaner overall. Getting small design details right such as making dropdowns look smooth and natural which took a lot more back-and-forth tweaking than I expected. It made me realize that front-end work isn't just about making things look good; it's about making them feel seamless too. Through all the trial and error, I definitely got better at noticing little things that affect the user's experience and figuring out how to polish them up.

WC: 92

Gavin:

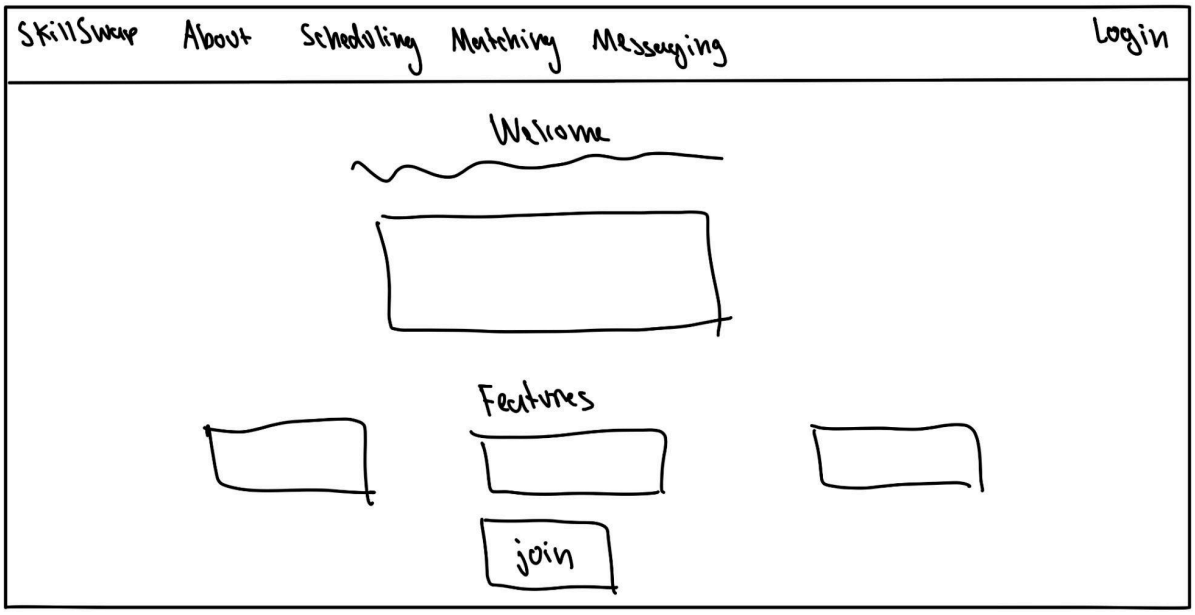
Most of my work was focused on the scheduling page. I spent a lot of time making sure the scheduling system looked clean and was easy to interact with. One of the biggest challenges was adjusting the layout so it would still feel natural in both light mode and dark mode without breaking anything. Getting all the small visual pieces to match the rest of the site took more time than I expected, but it gave me good experience with thinking about both functionality and design at the same time when building a full page from scratch.

WC: 92

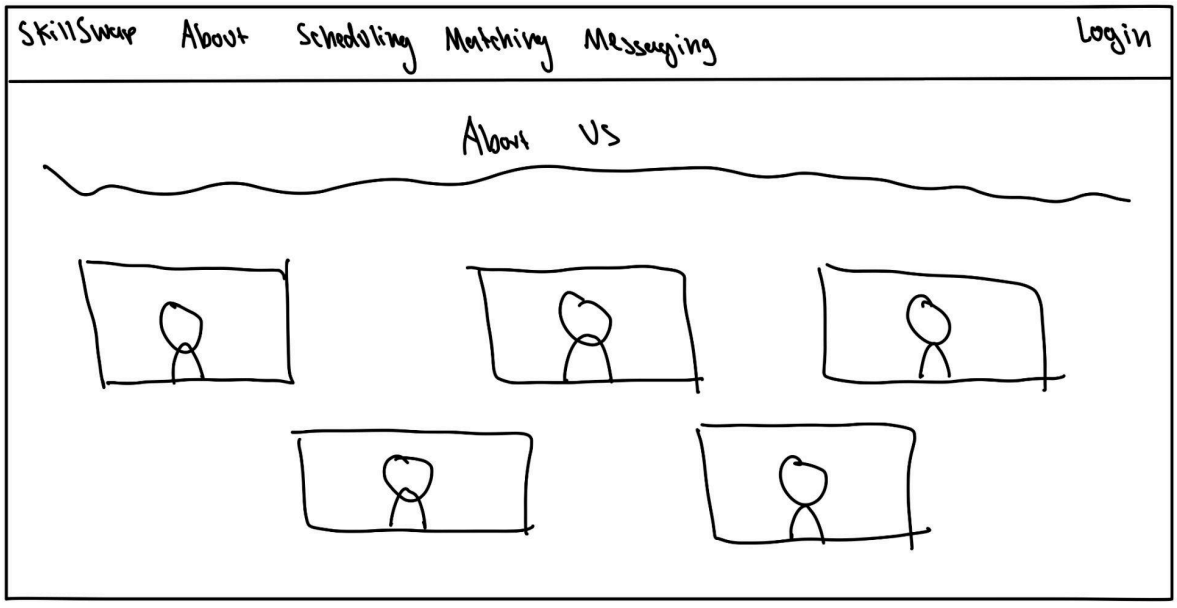
Use Case Diagrams:

Wireframes:

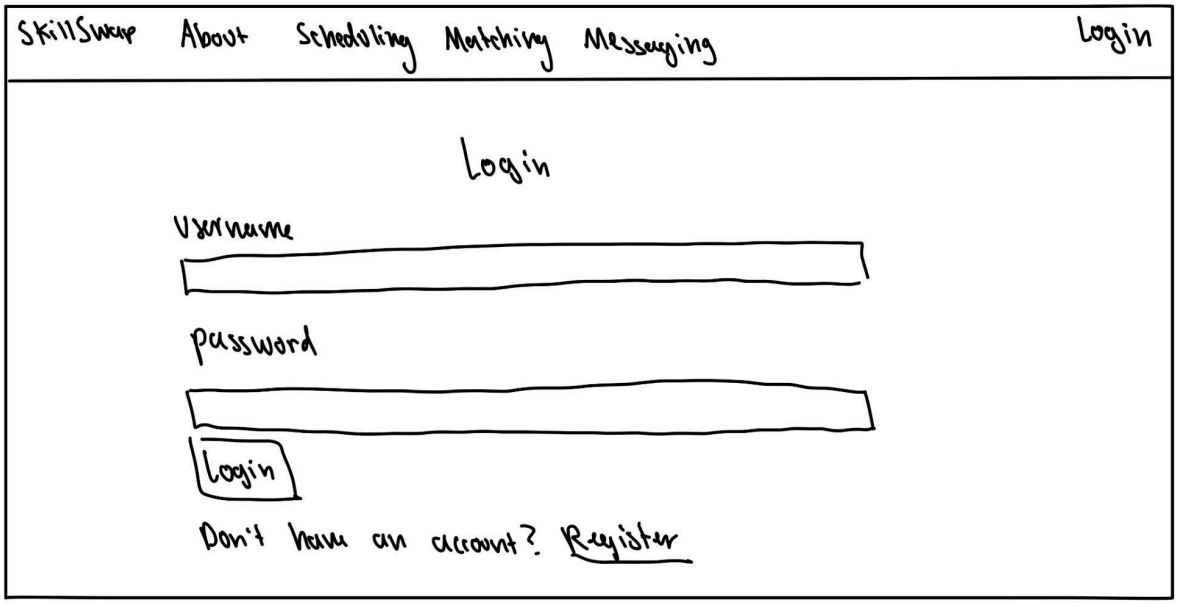
Home Page:



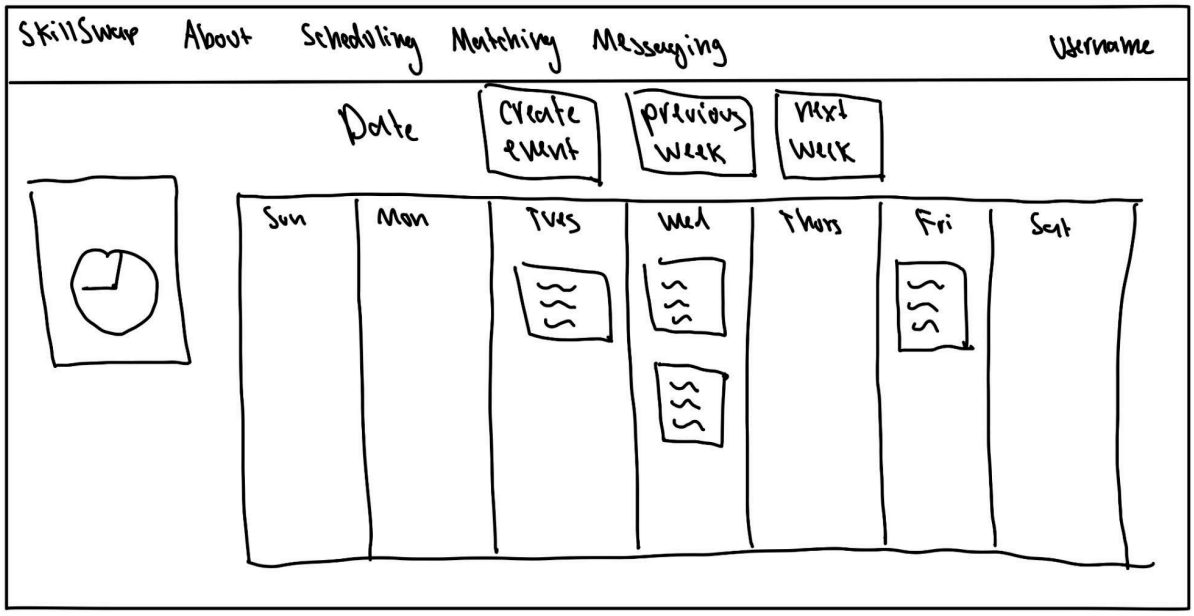
About Page:



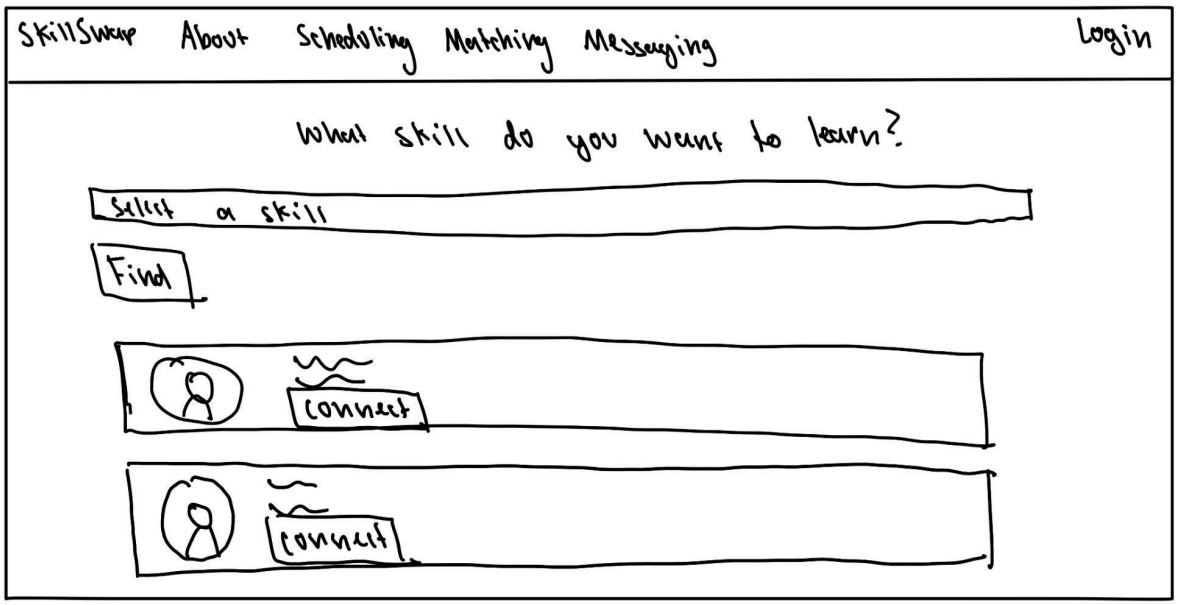
Login Page:



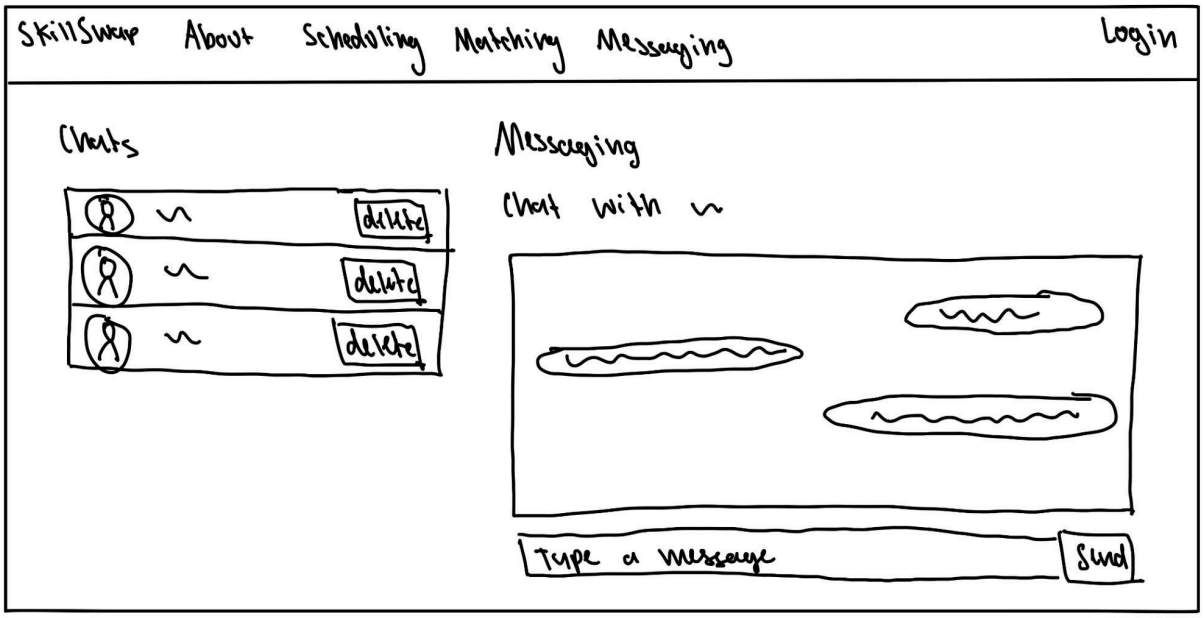
Scheduling Page



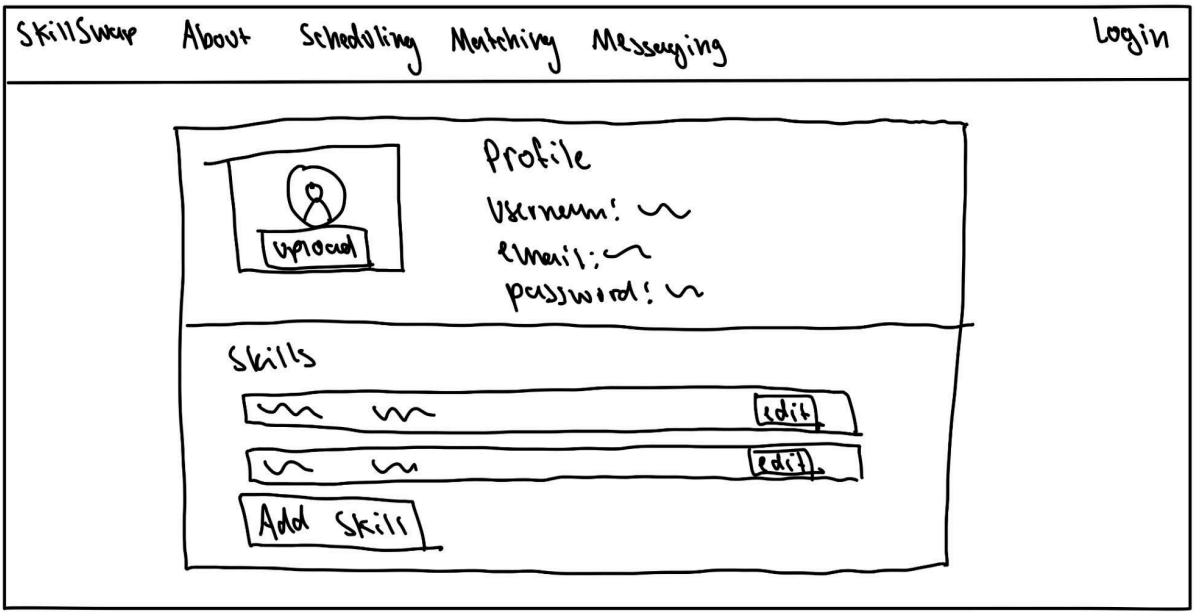
Matching Page:



Messaging Page:



Profile Page:



Test Plan Observations:
User Acceptance Test Plan

Test Environment:

- Platform: localhost via Docker
- Access: App runs at http://localhost:3000
- Testers: Team members of Skill Swap

Feature 1: Register a New User
Description:
User fills out the registration form with valid credentials.

Test Data:
- username: test123
- email: test@example.com
- password: test123

Steps:
1. Go to /register
2. Enter a valid username, email, and password
3. Submit the form

Expected Result:
User is redirected to the login page.

Feature 2: Login with Valid Credentials
Description:
User logs in using the correct credentials.

Test Data:
- username: test123
- password: test123

Steps:
1. Go to /login
2. Enter the correct username and password
3. Submit the form

Expected Result:
User is redirected to the homepage and is logged in.

Feature 3: Add a Skill on Profile
Description:

Logged-in user adds a skill with a level on their profile page.

Test Data:
- skill name: JavaScript
- expertise level: advanced

Steps:
1. Log in and go to /profile
2. Enter a skill name and level
3. Click “Add Skill”

Expected Result:
Skill is displayed on the profile page under the user’s skills list.

Acceptance Criteria:
- Forms must display success or error messages.
- Submitted data must persist and show on the next page load.
- Invalid or incomplete data should result in clear feedback to the user.

Source Code (Tag repo):
Done

ReadMe.md (link):
https://github.com/annika-polavarapu/skill-swap/blob/main/README.md
Can access the application through the link in the README.