SkiPool:

Team Members (Name, GitHub Username, Email):

- Manav Shah, mash8527, manav.shah@colorado.edu
- Kerem Gurkan, CoderLLamaPhone, kerem.gurkan@colorado.edu
- Jeremy Huang, sandsowkd, jeremy.huang@coloado.edu
- Colin Wallace, cwallyy, colin.wallace-1@colorado.edu
- Shane Wierl, shwi6838, shwi6838@colorado.edu
- Rogan Tinjum, rogantinjum, roti8360@colorado.edu

Project Description:
SkiPool is a ride-sharing app designed to make winter mountain trips more affordable, accessible, and environmentally friendly while fostering an inclusive community of outdoor enthusiasts. It connects riders seeking transportation with drivers already heading to the mountains, helping users save on expenses like gas, parking, and tolls. This not only lowers costs but also reduces the environmental impact and opens access to those without their own transportation.

The website lets passengers filter by ski resort, ski pass type, or price, and allows ratings for both drivers and passengers. Users can create a post as a driver or sign up as a passenger. Drivers specify the date of travel, price per passenger, number of available seats, and ski resort. Passengers provide their pickup address and phone number to coordinate timing.

Beyond cost savings, SkiPool builds community in a space traditionally limited by high expenses. By allowing users to rate and review each other, it promotes trust, accountability, and safety. Reliable and respectful members are rewarded, while poor behavior is discouraged, ensuring a safer, more enjoyable way to share mountain adventures.
Version Control: https://github.com/CoderLLamaPhone/SkiPool

Github Project Board: https://github.com/users/CoderLLamaPhone/projects/2/views/1

Demo Video: https://youtu.be/t-Nspdhy29g

Contributions:

Colin - I made the login and register completely work with the database, created the profile page, made the profile page dynamic with the ability to edit their personal information and add or delete the cars they claim as theirs. Once that finished, I created a search page to search for users that you want to interact with, and view their profile and have the option to message them by clicking on the “message” button on their profile. This takes them to a chatroom between the two people. I also did the Live Demo for the presentation and filmed the youtube Demo.

Rogan - I started the project by creating the package.json file with essential project details. I then built the home page, which highlights key features and links to the registration page. I added action calls in index.js for navigation and included an app.use call for local resources. After applying styling and making the app mobile-friendly, I wrote positive and negative test cases for the /register and /profile routes in server.spec.js and responses in index.js to ensure functionality. Finally, I helped create the presentation, working on slides covering our app’s description, future enhancements, and the architecture diagram.

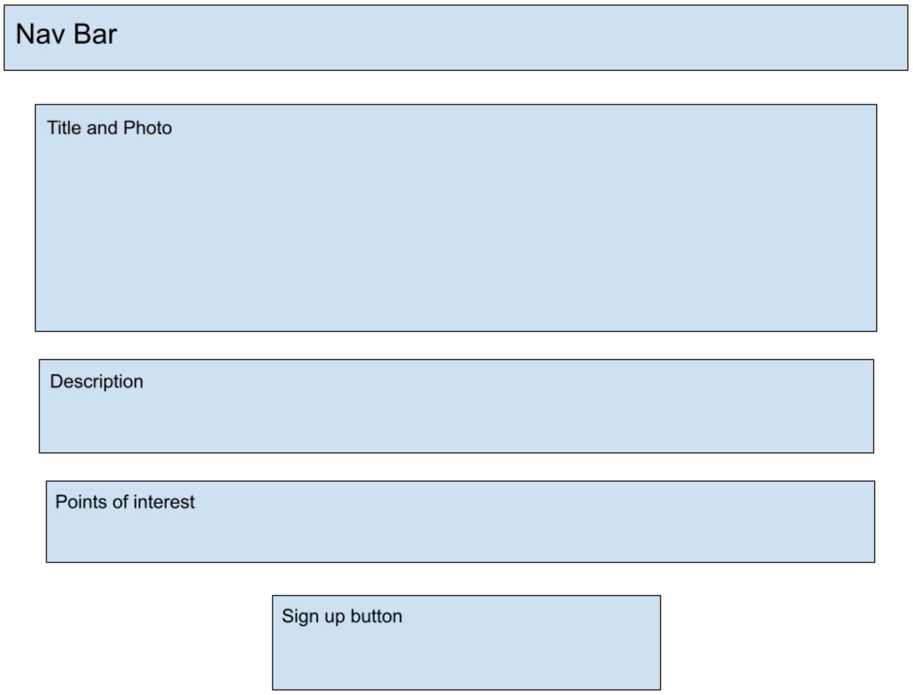
Manav - I worked on the navigation bar and implemented the partials initially. I then spent the majority of the remaining time working on the driver page. This includes the pop-up modal that appears when the “Create Drive” button is clicked, all the endpoints for the driver page in index.js, and the correct trip appearing under “Your Trips” once created. I also implemented the “Delete” button, including the endpoint, which allows users to delete a trip they created before it is completed. Besides this, I also created the UAT-plans.txt, completed the login and add a drive tests, completed Lab-9-Kickstart.txt, and worked on the demo video.

Shane - I worked on the use-case diagrams and helped with the wireframes and basic UI layout of the app during the planning phase. I set-up the profile page skeleton and base forms, and connected the backend express to our statics folders for images. I did most of the styles, and cross-designed pages for consistency. I also designed a logo on Illustrator for our app, and chose fonts and colors for the visual design of the app. I created a back-button in the chatrooms, and made a ton of UI friendly changes on pages for buttons, forms, and links. I reviewed many pull requests, and I helped with various bug fixes, backend routes, and navigation issues. I wrote and presented our tech stack and tools for the live presentation.

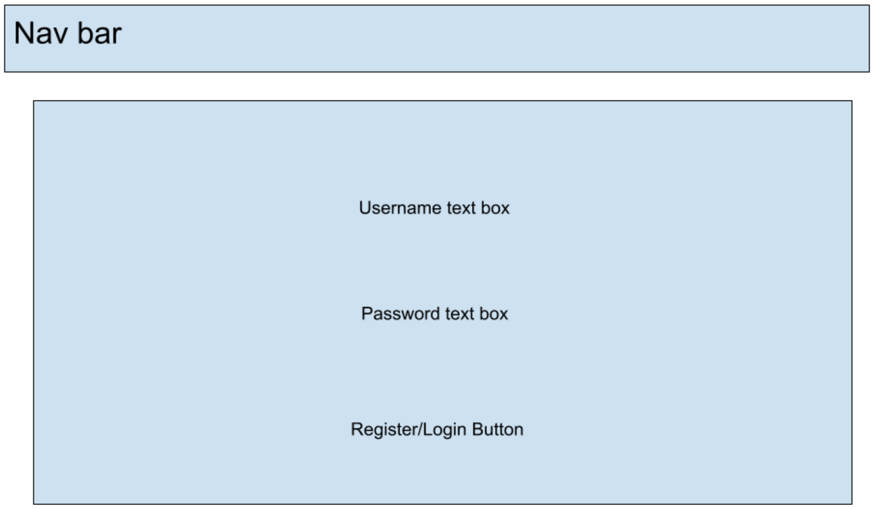
Jeremy - I first worked on the "Find a Ride" page and implemented partials. Initially, I created sample ride data to test the functionality of the page and built features such as filtering rides by resort, departure date, ski pass, and more. Then, I developed a feature to calculate the total number of available seats for each ride, ensuring that sign-ups from different user accounts could not exceed the ride's seat limit. I also styled the entire page to match the overall design of the site. Afterward, I built an "Edit Ride" feature that allowed users to cancel or update their ride details, again enforcing the seat limit logic. Additionally, I implemented a "Submit Rating" feature that became available once the ride date had passed, allowing users to rate their drivers out of five stars and handling the rating calculations. Toward the end of the project, I also contributed to preparing the final presentation, specifically working on the tools and challenges pages.

Kerem - I began working on the project with setting up branch protections and best practices for the team regarding our version control. I then set up the docker file and created an ER diagram to share with the group and make sure every feature had everything it needed. I then implemented this and added fake data and assisted other members with technical issues experienced. I then worked managing pull requests and doing integration tests and fixing known bugs. I then added the entirety of the chat room system and made it live reload. Later the group decided to add multi member group chats which resulted in the database and a number of endpoints needing to be refactored. I oversaw the live deployment of the server and troubleshooted a number of problematic test cases. I also updated the readme for the repository and delivered the pitch and future prospects of the presentation.

Wireframes:

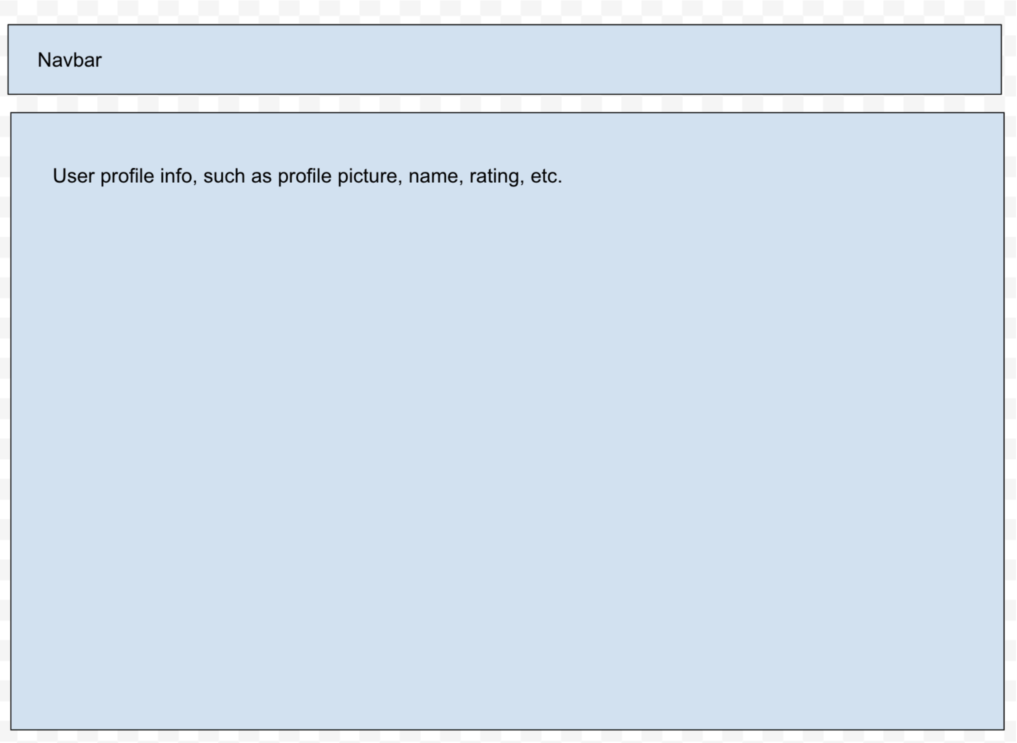


The Home Page -



Register/Login Page -

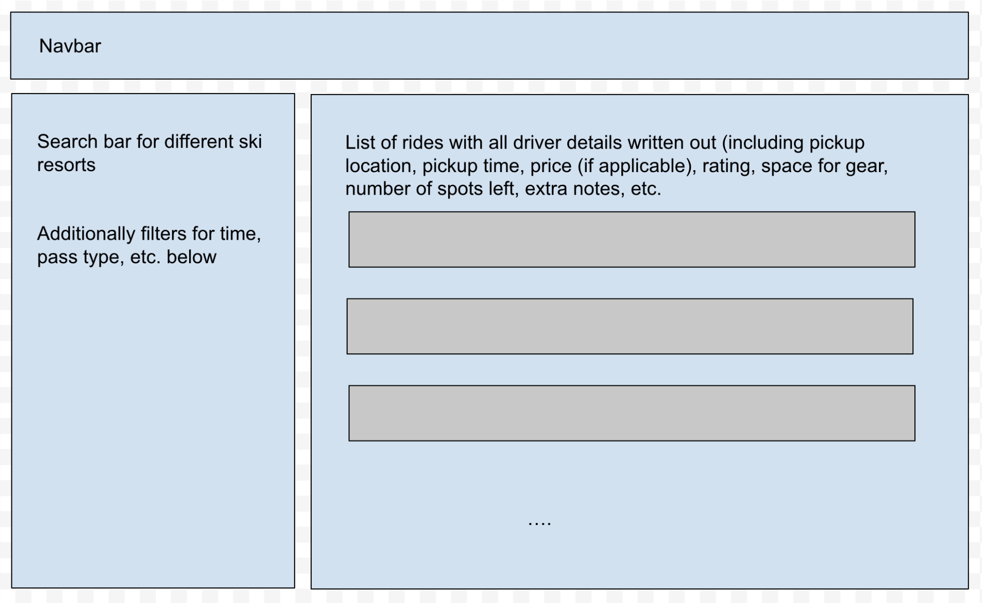
Profile page -



Navbar

User profile info, such as profile picture, name, rating, etc.

Find A Ride Page -



Navbar

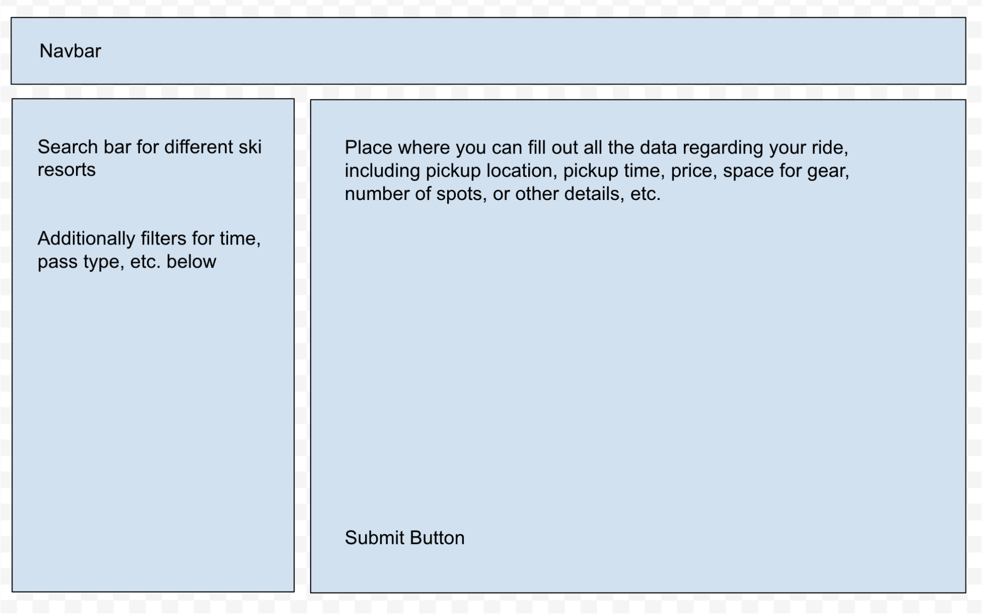
Search bar for different ski resorts

Additionally filters for time, pass type, etc. below

List of rides with all driver details written out (including pickup location, pickup time, price (if applicable), rating, space for gear, number of spots left, extra notes, etc.

….

Drive Post Page -



Navbar

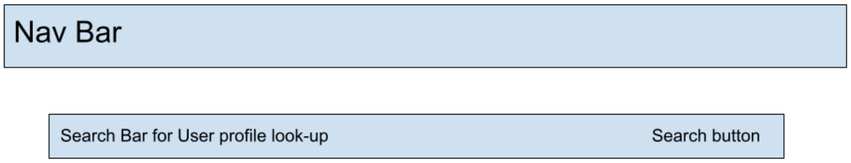
Search bar for different ski resorts

Additionally filters for time, pass type, etc. below

Place where you can fill out all the data regarding your ride, including pickup location, pickup time, price, space for gear, number of spots, or other details, etc.

Submit Button

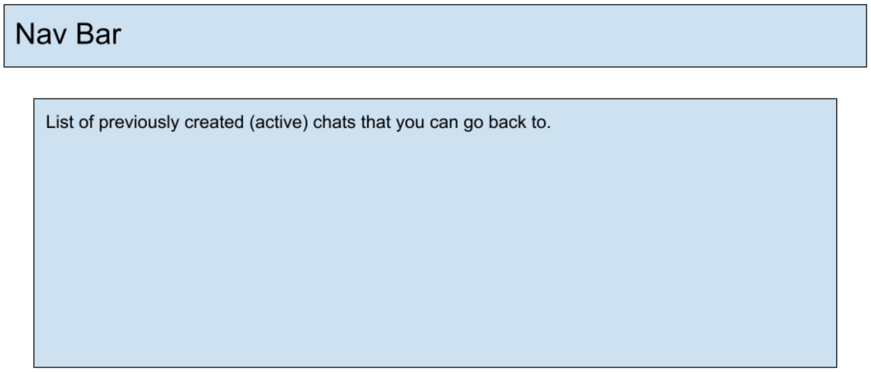
Search For Users Page -



Nav Bar

Search Bar for User profile look-up

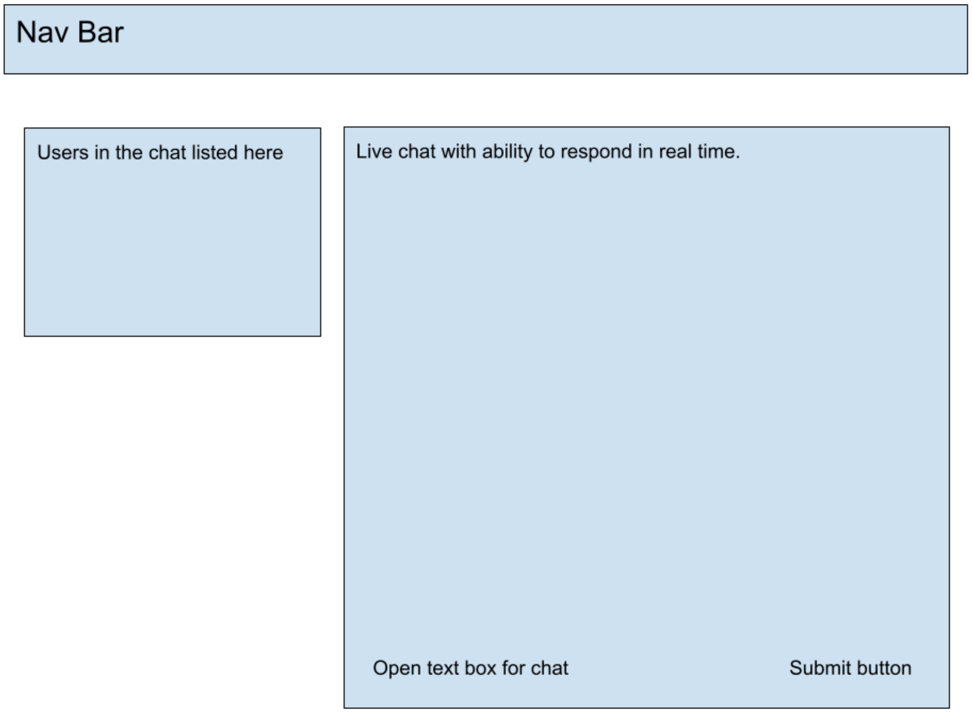
Search button



Nav Bar

List of previously created (active) chats that you can go back to.

Message Hub Page -



Nav Bar

Users in the chat listed here

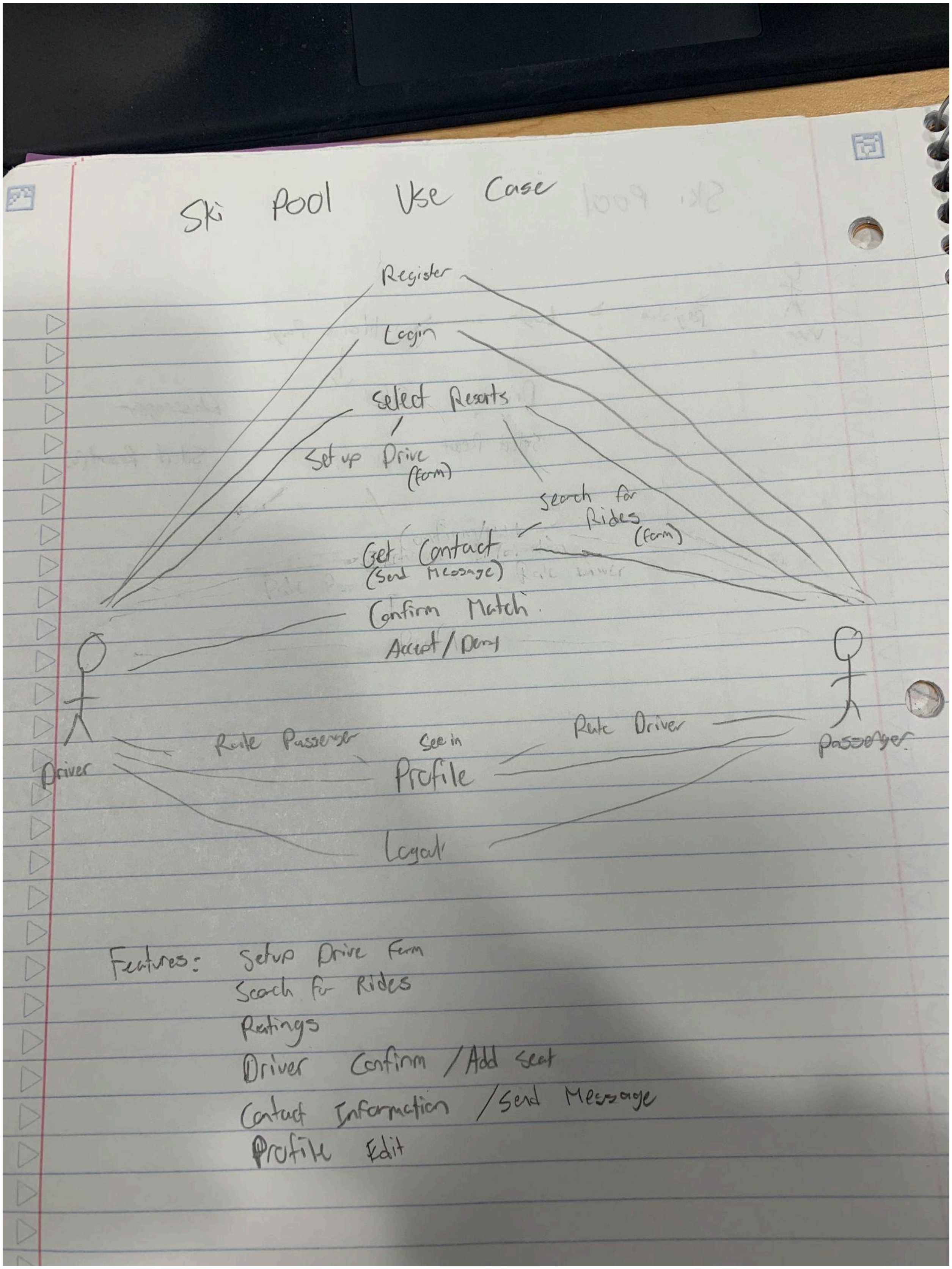
Live chat with ability to respond in real time.

Open text box for chat

Submit button

Chat Page -

Use Case Diagram:



Ski Pool Use Case

Features: Setup Drive Form
Search for Rides
Ratings
Driver Confirm / Add Seat
Contact Information / Send Message
Profile Edit

Test Plan Observations:

- Login/register test:

  Actual Test Results:

Username	Password	Actual Outcome
test_username	test_password	Successful login
test_username	incorrect	Login fails
incorrect	test_password	Login fails and redirects to register page
does_not_exist	does_not_exist	Login fails and redirects to register page

  Observations:
  There is a minor discrepancy between the actual outcome and the expected outcome. The expected outcome prompts failed logins to display an error message. However, after completing this test, we decided that redirecting users to the register page for failed login attempts is more appropriate than simply displaying an error message. Besides this, all login attempts executed as expected.

- Find a Ride filter test:

  Actual Test Results:

Filter	Actual Outcome
Set ski resort to Aspen Snowmass	Only rides that go to Aspen Snowmass appear
Set pass type to Ikon	Only rides that go to ski resorts that are on the Ikon pass are shown
Set departure date to 4/30/2025	Only rides that depart on 4/30/2025 show up
Set maximum price to $50	Only rides that cost $50 or less appear
Set minimum available seats to 3	Only rides that have at least 3 seats available are shown

  Observations:
  All filter test outcomes match their corresponding expected outcome in UAT-plans.txt.

- Add a Drive test:

  Actual Test Results:

Drive	Actual Outcome
Create a new drive with the following information: Pickup Location: Engineering Center	A trip with the same information appears under "Your Trips," and

Destination Resort: Arapahoe Basin Departure Date: 4/30/2025 Available Seats: 3 Gear Space: Available Additional Info: Test Pass Type: Ikon	the trip appears on the Find a Ride page.

Observations:
All the drive test outcomes match their corresponding expected outcome in UAT-plans.txt, however, the test data that was inputted is slightly different. Since this test was created in UAT-plans.txt before the create drive modal was fully complete, some of the fields described in the test do not exist in the create drive modal. For example, gear space and pass type do not exist in the create drive modal as it is assumed that there is space for gear since the website is meant for carpooling to go skiing and pass type is determined automatically by the destination resort. Furthermore, the create drive modal has some fields that are not described in the test plan such as price, estimated departure time, estimated return time, and car. Despite this, the same test was executed and the actual outcome was the same as the expected outcome.