Tudr: Student Focused Tutor Matching

CSCI 3308 Group Project Report

Group Members
Joe Hoertig-Paulus
Mason McGaffin
Nick Meagher
Curtis Preedom
Connor Shumate

Group Members

Name	GitHub Username
Joe Hoertig-Paulus	joehpaulus
Mason McGaffin	mamc3334
Nick Meagher	nime5333
Curtis Preedom	PreedomCL
Connor Shumate	cshumate2114

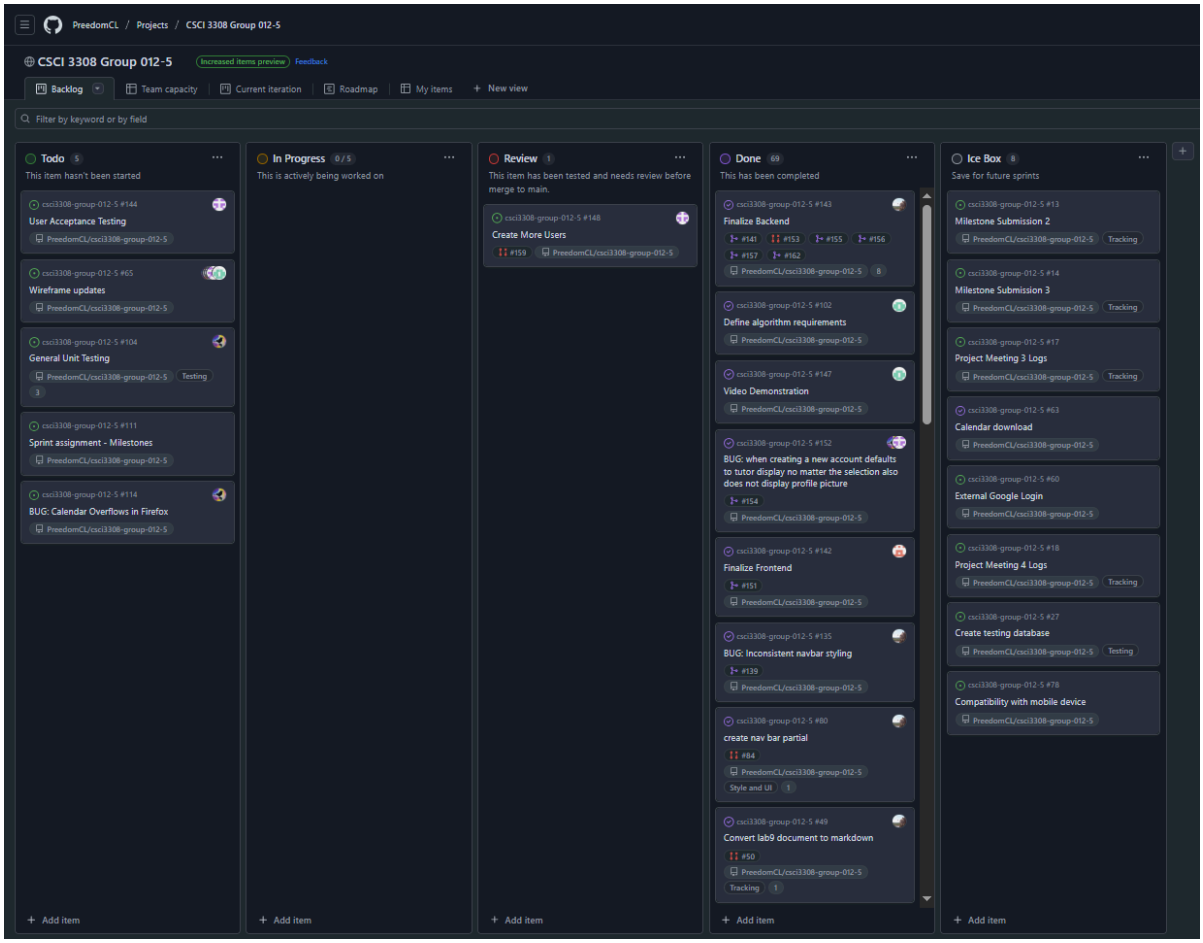
Project Description

Tudr is a web application which provides a personalized platform for university students to find tutors. Users provide information about their major, courses, learning style, and availability. These factors are used to suggest to students the tutors who best match their learning needs. After both the student and tutor have agreed to a match, students are able to schedule tutoring sessions directly through the site.

Both students and tutors may register either with their email address or Google account. When registering, students provide information about their learning preferences and tutors describe their experience and teaching style. Students are able to browse a list of suggested tutors and send a match request to any they like. Tutors can view a student's profile and approve or deny their match request. Tutors also use the in-platform calendar to publish their availability to their students. Students can view a matched tutor's calendar and request a tutoring session, which the tutor can then accept.

Project Tracker

GitHub Projects Board:
https://github.com/users/PreedomCL/projects/5



Screenshot of GitHub Projects board.

Video

https://drive.google.com/file/d/1cRLvntr2TrdASGy6R2B-Kk8CJaUQmNBP/view

VCS

Git repository hosted on GitHub:
https://github.com/PreedomCL/csci3308-group-012-5

Contributions

Joe Hoertig-Paulus
I contributed in three primary ways: backend code, UI functionality, and managing documentation. Using Node.js, Handlebars, HTML, and PostgreSQL I helped create the Home page, Login Page, Register Page, Profile Page, and some of their functionalities. I set up the profile image storage and route, helped make the profile pages adapt to tutor and student accounts, created the download calendar function (later changed to only download an event), set up routes to reflect matches on designated accounts, tested and reviewed team member's contributions, and made release notes/tags and meeting notes.

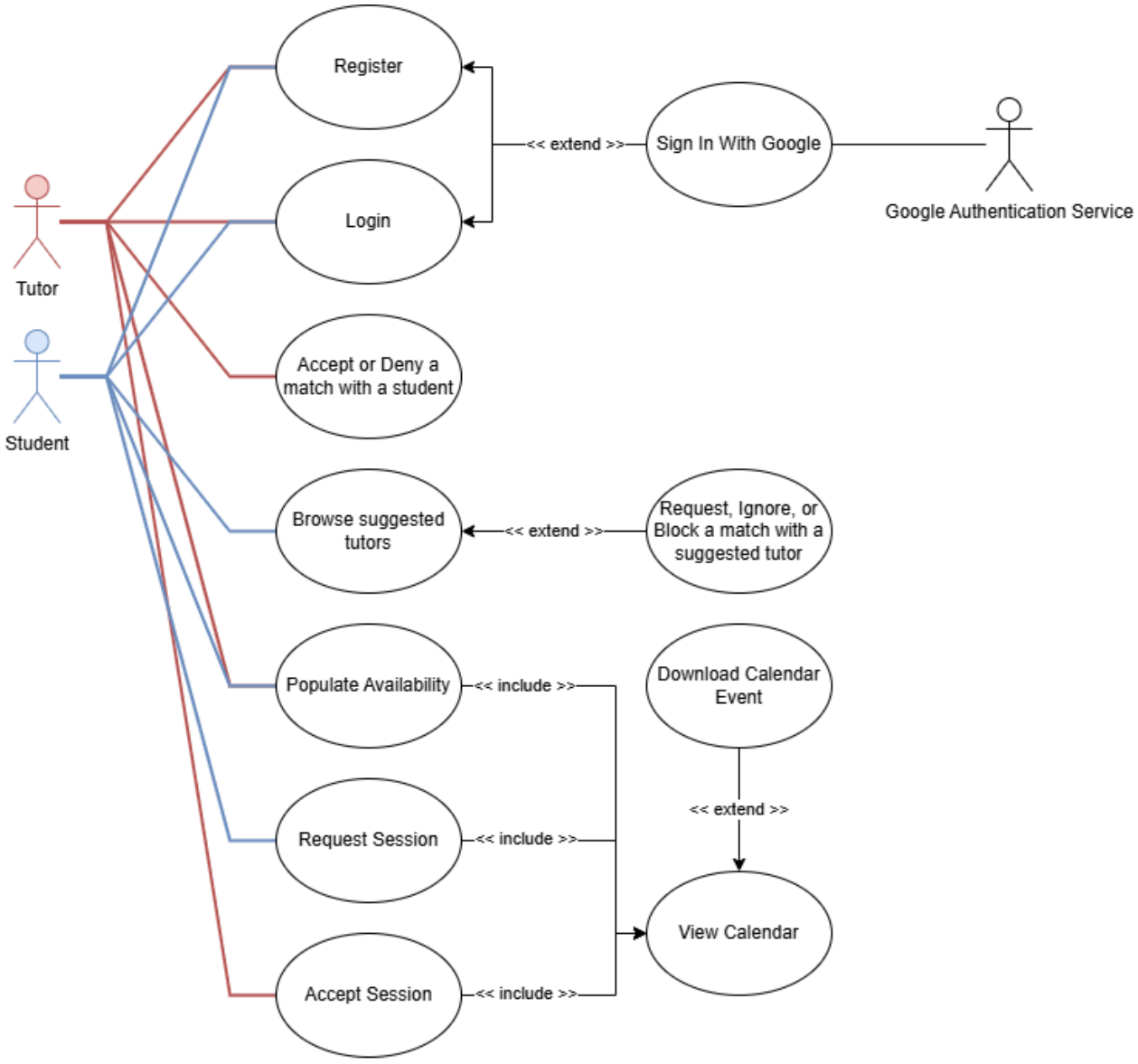
Mason McGaffin
My contribution was primarily full stack development for the calendar feature. I used the FullCalendar library as the backbone of this feature. Besides styling, I handled most of the development for the Profile Calendar and also the Match Profile modal, which were implemented as Handlebars partials. I used PostgreSQL and Node.js API calls to handle events scheduling and Calendar exports. I also helped Connor with a rework of the Matching logic to be a sort of three way handshake. I also handled the capability to schedule a tutoring session with the matched tutor.

Nick Meagher
I contributed primarily in three ways: front end development, style design, and the email notification integration.Using Node.js, Handlebars, HTML, Nodemaller, Bootstrap. My main focus was designing and implementing the style and final design of the website across all pages. My main goal with this was to make the website look professional and seamless for release, which included bug fixing and troubleshooting. Also, the implementation of the email system through Nodemailer, using a free gmail account, that will send email updates to the user about account creation and match requests.

Curtis Preedom
I contributed in two primary ways: core backend code and merge reviews. Using Node.js, Express, and PostgreSQL, I implemented the back-end for user registration and authentication. Using Google's API, I implemented the entirety of the Sign in with Google feature. Using Render I deployed the application. I spent significant time testing and reviewing my team member's contributions. I also made minor contributions to the appearance and client side functionality of certain pages.

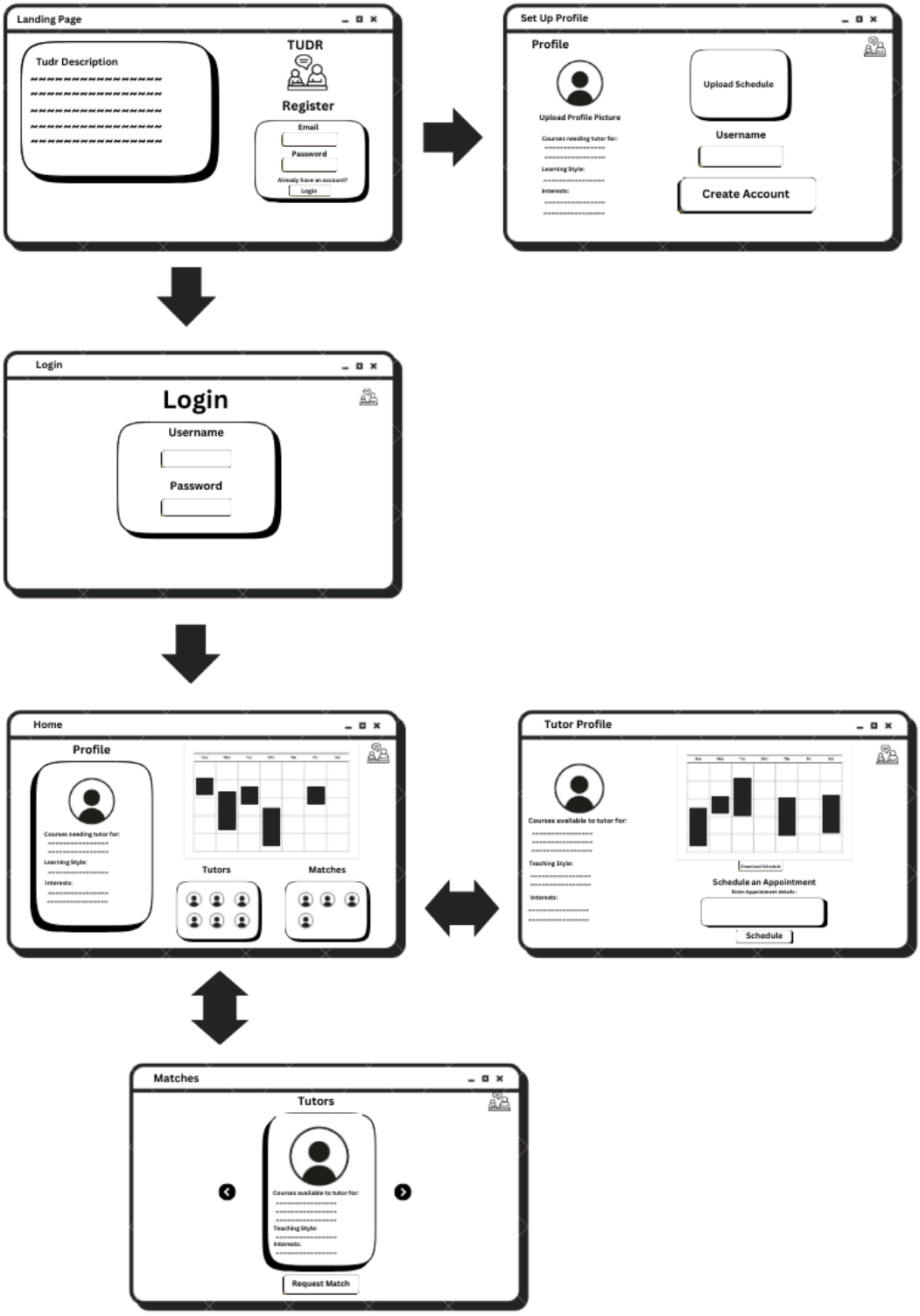
Connor Shumate
Using Node.js and PostgreSQL, I was responsible for making the matching phase of the website. I used SQL queries to select potential matches from the database, then displayed them to the user with all of the relevant data. I then made and functionalized the like and skip buttons to store matched and skipped users in a database. Finally I learned how to use indexing to handle the routes that showed only one potential match at a time. I was also in charge of setting up the file structure for our project, creating organized directory formatting.

Use Case Diagram



Wireframes

You must include all the wireframes that were created for the project. It is expected that you have at least one for each page. They can be photographs of hand-drawn images.



Test Results

Tests: found at
https://github.com/PreedomCL/csci3308-group-012-5/blob/lab11/MilestoneSubmission/uat_plan.md

Student Tutor Matching

Description: This tests the ability for students and tutors to adequately be matched. This test makes sure that students are able to request a match, tutors are able to accept that match, and the student is able to view that match.

Results from the test: The user was able to log in as a student, request the match. Then log in as a tutor and accept the match. The match was then displayed on both student and tutor profiles.

Observations:

What is the user doing? The user is following the instructions for the use case tests in the UAT plan file. They are logging in with the student credentials provided, requesting the match. Then logging out. Logging back in with the tutor credentials provided, accepting the match. Then they are viewing the match with both profiles, by logging in and out of the designated accounts.

What is the user's reasoning for their actions? The user is following the instructions in the UAT plan file adequately and is able to follow the instructions as the application is working properly.

Is their behavior consistent with the use case? Yes, their behavior is consistent with the use case.

If there is a deviation from the expected actions, what is the reason for that? No, there is no deviation from the expected actions. The user is able to log in as a student, request the match. Then log in as a tutor and accept the match. The match is displayed on both student and tutor profiles.

Did you use that to make changes to your application? If so, what changes did you make? No.

Register and Login

Description: This tests the ability for students to register and log in into Tudr. This test ensures that students are able to register with the appropriate credentials and log in using those credentials without any problems.

Results from the test: The user was able to register as a student with the provided credentials in the UAT plan. The user was also able to log in with the matching credentials to the register

portion of the test. Once redirected to the profile page after logging in, the user was able to see all of its provided information reflected accurately in the designated profile locations.

Observations:

What are the users doing? The user is following the instructions in the UAT plan file. The user is registering using the credentials provided in the UAT plan file. They have successfully registered and are redirected to the login page. They log in with the credentials provided and are now redirected to the profile page. Once on the profile page, they are able to see the information they registered with reflected on the profile page in the designated locations.

What is the user's reasoning for their actions? The user is following the UAT plan file instructions accurately and carefully.

Is their behavior consistent with the use case? The user's behavior is consistent with the use case.

If there is a deviation from the expected actions, what is the reason for that? There were no deviations from the expected actions. The user was able to follow the UAT plan file instructions completely and register, log in, and see the credentials provided reflected in the designated locations on the profile page.

Did you use that to make changes to your application? If so, what changes did you make? No.

View and Download Calendar

Description: This tests the ability for a user to view and download their calendar event by navigating to the profile page, requesting a match with the tutor, clicking on the event time, and downloading the event information, then seeing the event information accurately displayed on their local device calendar.

Results from the test: The user was able to login with the given credentials. Create an availability event as a tutor, propose a meeting as a student, accept the meeting as a tutor, and download the event both as a tutor and student (able to download both accepted meeting and pending meeting).

Observations:

What are the users doing? The user is following the instructions in the UAT plan file. The user is able to log in as a student, see calendar. User logs in as a tutor, creates an availability slot. User then logged in as a student, saw the tutor match, requested a meeting with the tutor, and downloaded the pending meeting ics file. The user logs back in as a tutor, accepts the meeting, and downloads the accepted meeting.

What is the user's reasoning for their actions? The user is following the UAT plan file instructions adequately.

Is their behavior consistent with the use case? Yes.

If there is a deviation from the expected actions, what is the reason for that? No. The user was able to download the calendar event both as a student and as a tutor.

Did you use that to make changes to your application? If so, what changes did you make? No.

Deployment

The application is deployed on Render with two instances. One instance is the PostgreSQL database, and the second is a web service running the Node.js application server. The database was initialized by establishing a remote terminal connection and running an SQL script. The web service running the application server automatically builds the most recent version of the project from GitHub.

Access the live application with a browser:
https://csci3308-group-012-5.onrender.com/