Title: **Movie Meter**

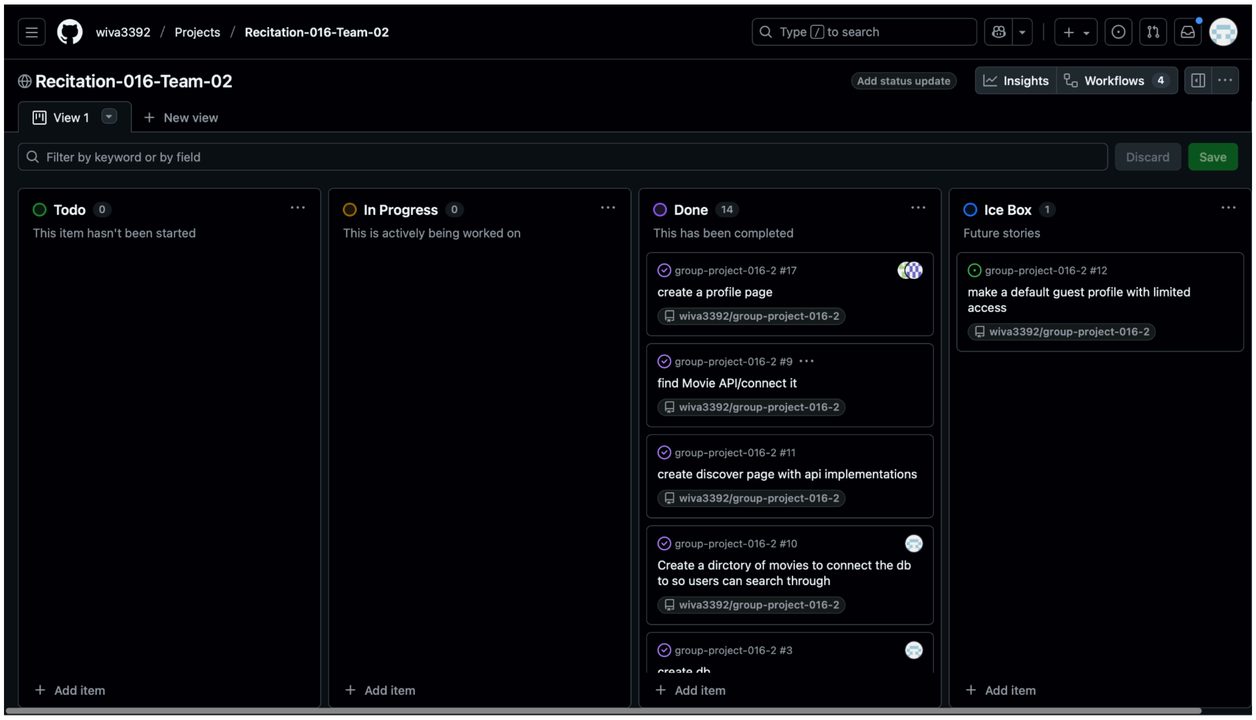
Member Details:
1. Hiale Haile (hhaile1)
2. Killian Wood (KillianJWood)
3. William Vance (wiva3392)
4. Abeneazer Getachew (abe-neezer)

Project Description:

Our group wanted to create a streamlined and easy way to review movies and to add movies to a watchlist, so we created Movie Meter. Movie Meter utilizes node.js, a third party API for movies, and a PostgreSQL database to store movies and users' reviews. It features a login and registration page in order for users to securely access their movies and reviews. This leads into a discovery page where users can find and review movies. If users want to add a movie to their watch list, each movie has an "add movie" button that moves it to their watchlist. Also, with each movie are "review" and "read" buttons. The "review" button brings users to a page that will allow them to score a movie on a scale of 1 through 10. Then below this is a text box where users can leave a typed out review to go with their score. The "read" button allows users to see their review and the reviews of other users. Finally the profile page shows users their watchlist, reviews, and their top movies based on their highest reviews. In the backend, all of this runs on a relational database that stores users' movies and reviews in tables.

Version Control: https://github.com/wiva3392/group-project-016-2

Project Tracker - GitHub project board:
https://github.com/wiva3392/group-project-016-2/projects?query=is%3Aopen



Todo 0
This item hasn't been started

In Progress 0
This is actively being worked on

Done 14
This has been completed

group-project-016-2 #17
create a profile page
wiva3392/group-project-016-2

group-project-016-2 #9
find Movie API/connect it
wiva3392/group-project-016-2

group-project-016-2 #11
create discover page with api implementations
wiva3392/group-project-016-2

group-project-016-2 #10
Create a dirctory of movies to connect the db to so users can search through
wiva3392/group-project-016-2

group-project-016-2 #3
create db

Ice Box 1
Future stories

group-project-016-2 #12
make a default guest profile with limited access
wiva3392/group-project-016-2

Demo Video:
[Movie Meter demonstration video](https://www.youtube.com/watch?v=d-JrttVZMY0)
https://www.youtube.com/watch?v=d-JrttVZMY0

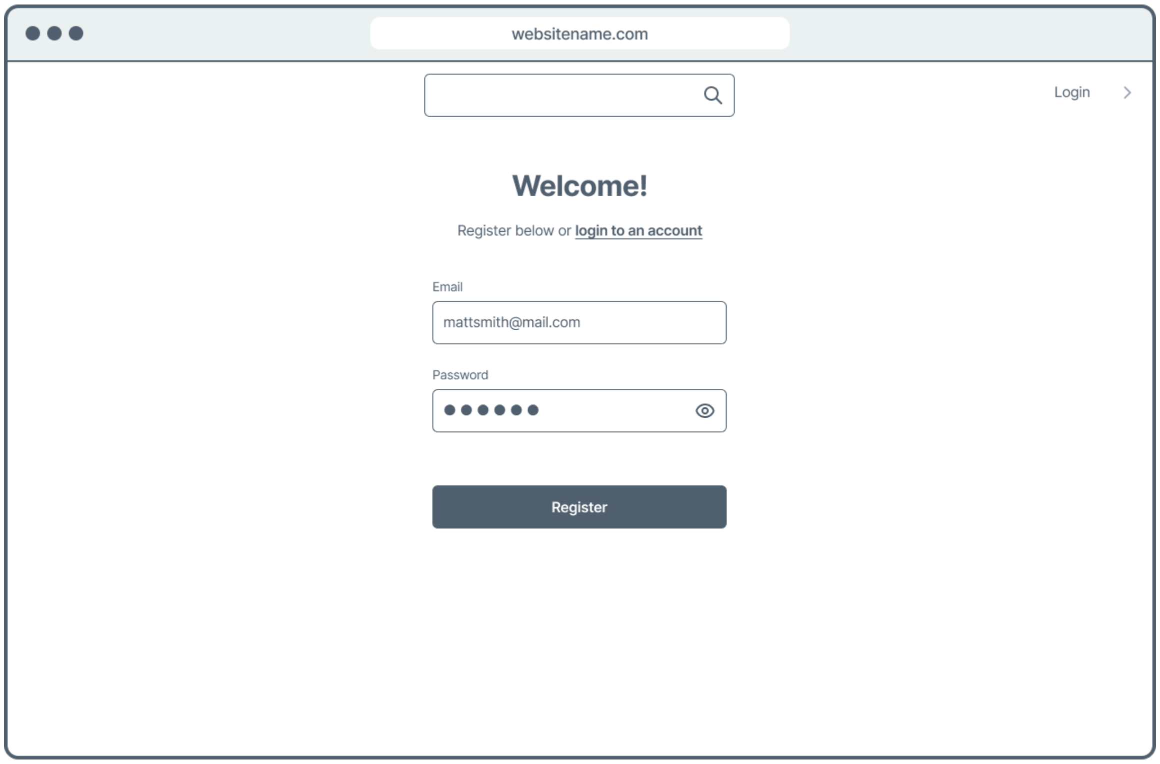
Contributions:
Hiale:
I co-designed our PostgreSQL database with Killian, refining the schema using feedback from our professor. After we selected an API, I implemented the core backend using JavaScript (Node.js + Express.js), including the GET and POST routes for adding, saving, and retrieving movies and reviews. I then built the associated **Handlebars** pages for displaying and submitting reviews. My work focused on developing the backend logic, connecting it to the database, and integrating it with the frontend templates.

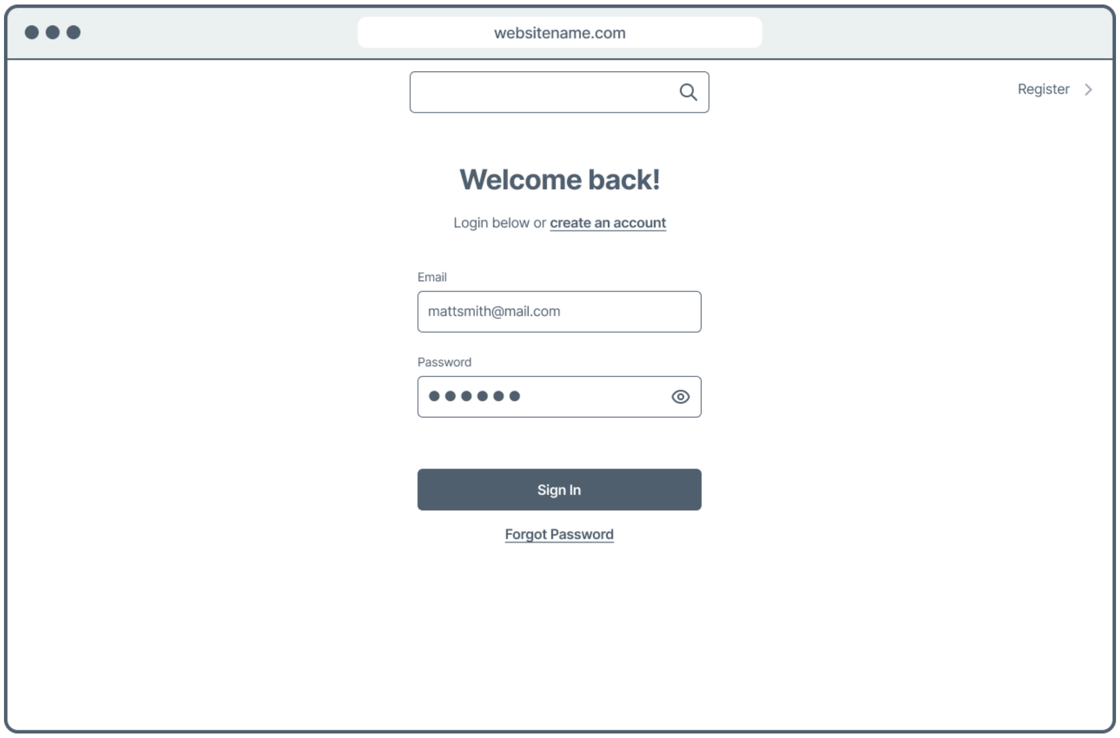
Killian:
I helped design the PostgreSQL database and created the ERD diagram for our team. I located and evaluated the external API we used, then integrated it into our backend using JavaScript (Node.js + Express.js). I implemented the initial GET requests to confirm functionality and developed the API calls for the Discover page and search feature. I also wrote the Chai testing tools to test the project. My contributions established the API integration and data flow that supported the rest of the project's features.

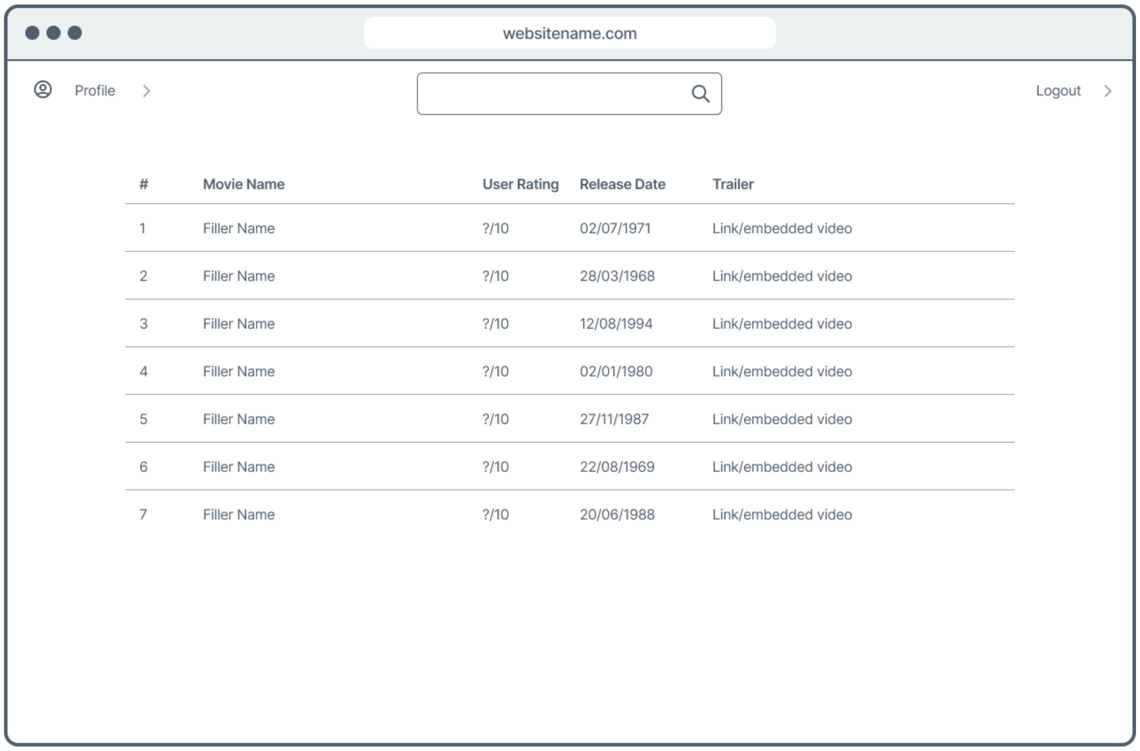
William:
I designed the functionality of the profile page using the established database. The profile page needed to track a user's reviews, watchlist, and top movies. For this, I added a user_list table to the database to join the movies and user id's table. Then I utilized this information in our index.js and profile.hbs using HTML and JavaScript (Node.js + Express.js) files to create the watchlist, reviews, and top movies sections. My contribution implemented the core functionality of the user profile and enabled streamlined tracking of each user's reviews and added movies.

Abeneazer:
For our MovieMeter project, I designed and built the front end of the website, focusing on a clean layout, readable text, and easy navigation Using CSS and HTML. I also created the login page, including the form structure and basic validation, and made sure it worked smoothly with the rest of the site also hashing the pass words for security using JavaScript (Node.js + Express.js) . My contribution gave the project a consistent look and made it easier for users to access and explore the movie features.

Wireframes:

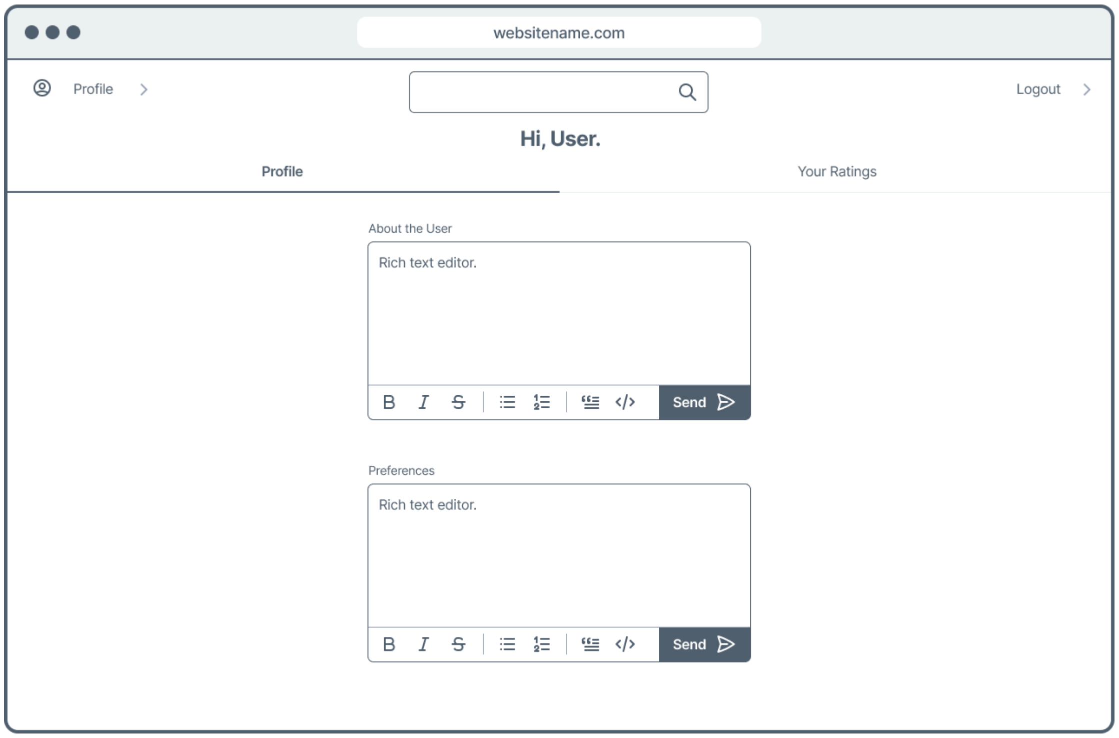
1.


2.


3. 

websitename.com

#	Movie Name	User Rating	Release Date	Trailer
1	Filler Name	?/10	02/07/1971	Link/embedded video
2	Filler Name	?/10	28/03/1968	Link/embedded video
3	Filler Name	?/10	12/08/1994	Link/embedded video
4	Filler Name	?/10	02/01/1980	Link/embedded video
5	Filler Name	?/10	27/11/1987	Link/embedded video
6	Filler Name	?/10	22/08/1969	Link/embedded video
7	Filler Name	?/10	20/06/1988	Link/embedded video

4. 

websitename.com

Hi, User.

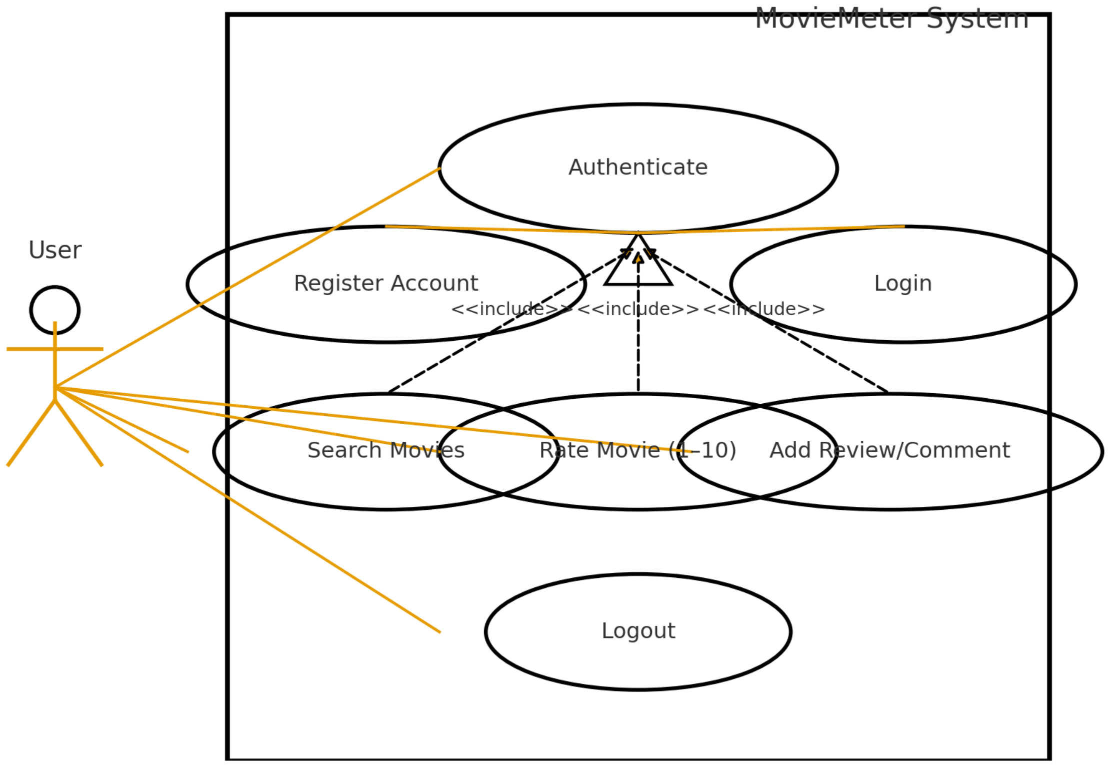
About the User

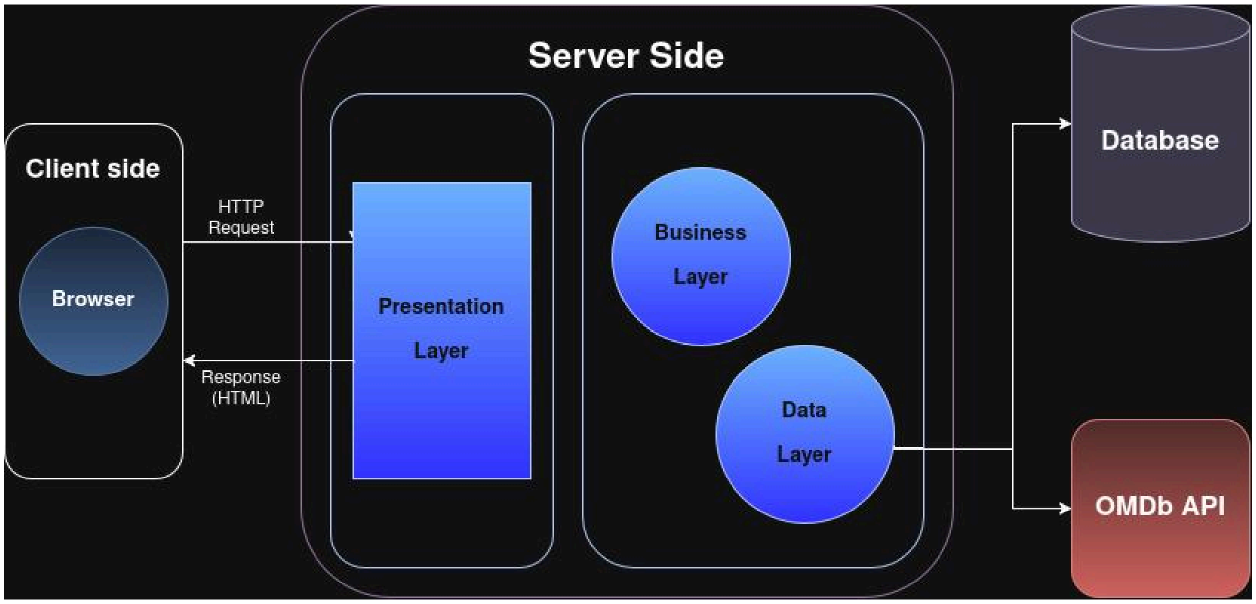
Rich text editor.

Preferences

Rich text editor.

Use Case Diagram/Architecture Diagram:





Test Cases:

As part of Lab 10, you will create a Test Plan for user testing your application. You will be required to execute that test plan to test your application present your findings in the final project report. The findings should include the following:

A description of the use cases being tested. You should be testing a minimum of 4 use cases. The audience for this activity should be someone outside of your team and preferably the class to receive objective feedback. If you are building an application in a specialized domain, you should be testing it with someone who is familiar with that domain.

Tester: Nathan M.

Test Case 1: Searching for Movies by Title Keywords

Observations

- What users did:
  Testers typed a variety of keyword combinations into the search bar, including full titles, partial phrases, and intentionally vague keywords. Many also tested multiple searches in a row to compare results.
- User reasoning:
  Users wanted to see how flexible the search system was. They expected the search to work similarly to other movie apps like IMDb or Rotten Tomatoes—meaning partial keywords or slightly misspelled words should still return results.
- Consistency with use case:
  Their behavior aligned perfectly with the intended use case of keyword-based search.
- Deviations from expected actions:
  A few testers tried entering symbols or numbers unrelated to movies (e.g., “###” or “123”), expecting the app to explain why no results appeared.
- Resulting changes made:
  After observing confusion with meaningless inputs, we added a “No results found—try different keywords”message to clarify the search outcome. This made the UI more communicative and reduced user uncertainty.

Test Case 2: Posting Reviews With Validation (blank, normal, overflow)

Observations

- What users did:
  Testers attempted to submit reviews in three forms:
  (1) empty review fields
  (2) normal reviews under 200 characters
  (3) over-limit reviews above 200 characters
  To test to see if the review limit will be enforced or not.
- User reasoning:
  They wanted to understand the boundaries of what the system allowed and how clearly validation messages guided them.
- Consistency with use case:
  Their actions directly matched the goal test review validation thoroughly.
- Deviations from expected actions:
  One tester tried to add numbers in their review in reference to the move and it initially did not work
- Resulting changes made:
  We added a function to remove numbers in the review entry to allow for users to add numbers in their review for more in depth reviews.

Test Case 3: Adding Ratings for Movies

Observations

- What users did:
  Testers added a rating 1-10 and submitted them with and without accompanying reviews. They refreshed the page and navigated to other movie pages to confirm the rating was saved.
- User reasoning:
  Users were checking whether ratings persisted in the database and whether the UI updated in real-time, similar to modern rating systems.
- Consistency with use case:
  Their interactions fully aligned with the intended behavior for rating submission.
- Deviations from expected actions:
  The profile page that finds top rated movie displayed with many decimal places which was not expected and was not what we wanted to present in the UI

- Resulting changes made:
  We implemented a small update so the profile page that only shows one decimal as opposed to many

Test Case 4: Account Registration, Login, and Navigation through Reviews

Observations

- What users did:
  Testers registered accounts, logged in, browsed movie pages, checked other users' reviews, and navigated through the layout to ensure everything made sense.
- User reasoning:
  They wanted to confirm that account creation was smooth and that browsing reviews felt intuitive. Many compared the navigation flow to letterboxd or standard review apps.
- Consistency with use case:
  Their behavior was entirely aligned with validating the usability of account creation and review navigation.
- Deviations from expected actions:
  Several users tried clicking on UI elements that was interactive but not clickable, such as hovering over movies. This showed they expected more clickable components than originally provided.
- Resulting changes made:
  We updated certain text labels into clickable links (like "View All Reviews") and adjusted spacing/button styling to make interactive elements more visually distinct. This made it clear what was actually clickable as opposed to just interactive.

ReadMe: https://github.com/wiva3392/group-project-016-2/blob/master/ReadMe.md

Deployment: https://movie-meter-xlqs.onrender.com/