Title: Rinder
Who: Justin Goh, Austin Schene, Jonny Lunney, Kevin Yang, Benjamin Hyde, Alexis Marez
Project Description:

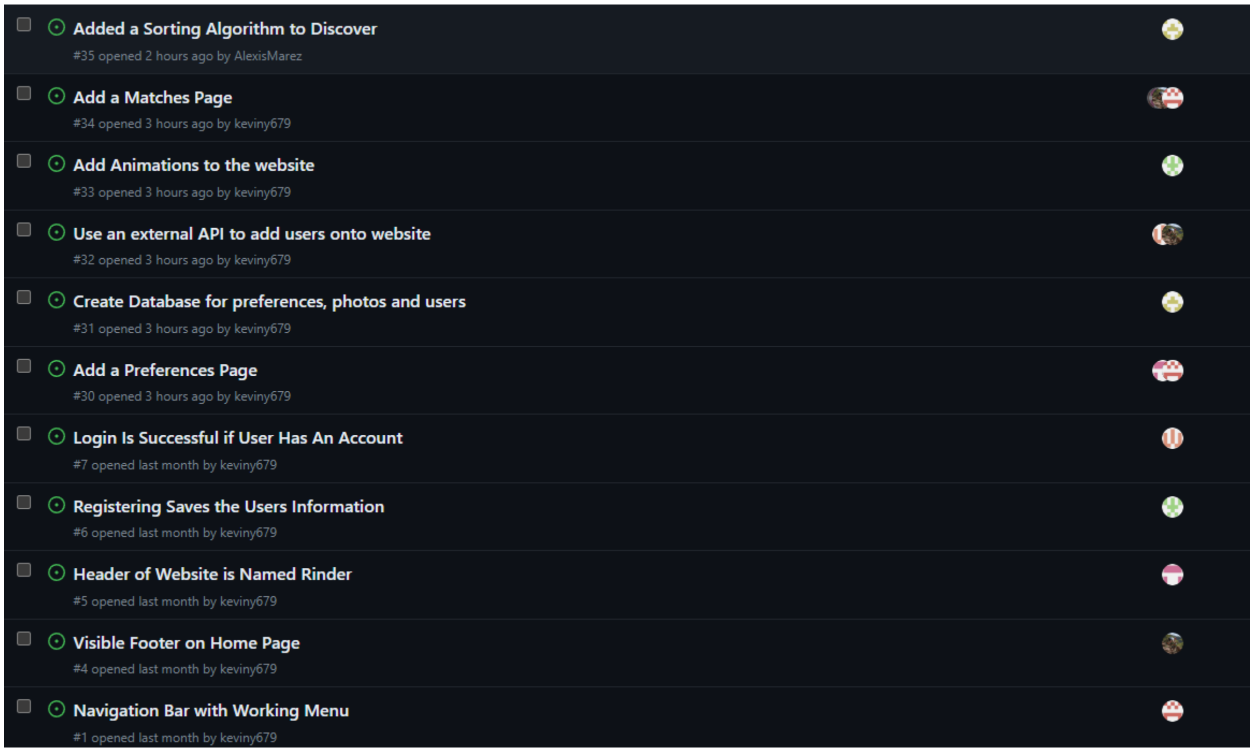
Introducing "Rinder" - the innovative solution to the age-old challenge of finding the perfect roommate. Rinder is a user-friendly app designed to streamline the roommate search process, inspired by the popular dating app Tinder. With Rinder, users can create detailed profiles that showcase their lifestyle preferences, housing needs, and personal interests. Aspiring roommates can then browse through these profiles, liking any profile they come across and reach out. This intuitive interface allows users to quickly identify potential roommates who align with their living preferences, making the arduous task of finding the ideal living companion easier and more enjoyable.

Project Tracker:

Link to project: https://github.com/orgs/CU-CSCI3308-Fall2023/projects/44

Screenshot of showing project in project tracker:

Project Board: https://github.com/orgs/CU-CSCI3308-Fall2023/projects/44/views/1



Added a Sorting Algorithm to Discover
#35 opened 2 hours ago by AlexisMarez

Add a Matches Page
#34 opened 3 hours ago by keviny679

Add Animations to the website
#33 opened 3 hours ago by keviny679

Use an external API to add users onto website
#32 opened 3 hours ago by keviny679

Create Database for preferences, photos and users
#31 opened 3 hours ago by keviny679

Add a Preferences Page
#30 opened 3 hours ago by keviny679

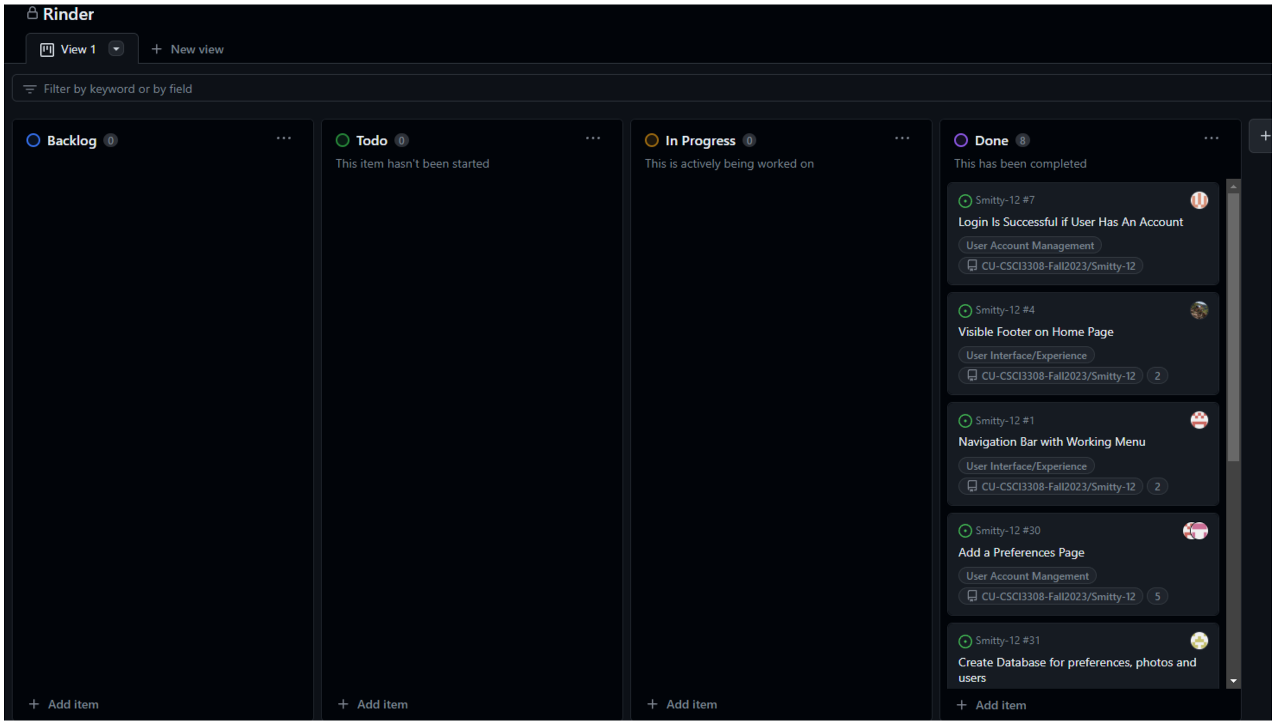
Login Is Successful if User Has An Account
#7 opened last month by keviny679

Registering Saves the Users Information
#6 opened last month by keviny679

Header of Website is Named Rinder
#5 opened last month by keviny679

Visible Footer on Home Page
#4 opened last month by keviny679

Navigation Bar with Working Menu
#1 opened last month by keviny679


Rinder

View 1 | New view

Filter by keyword or by field

Backlog 0

Todo 0
This item hasn't been started

In Progress 0
This is actively being worked on

Done 8
This has been completed

Smitty-12 #7
Login Is Successful if User Has An Account
User Account Management
CU-CSCI3308-Fall2023/Smitty-12

Smitty-12 #4
Visible Footer on Home Page
User Interface/Experience
CU-CSCI3308-Fall2023/Smitty-12 2

Smitty-12 #1
Navigation Bar with Working Menu
User Interface/Experience
CU-CSCI3308-Fall2023/Smitty-12 2

Smitty-12 #30
Add a Preferences Page
User Account Mangement
CU-CSCI3308-Fall2023/Smitty-12 5

Smitty-12 #31
Create Database for preferences, photos and users

+ Add item

Video:

Link: https://youtu.be/dIlmwNzl3DU

VCS:

Link to repo: https://github.com/CU-CSCI3308-Fall2023/Smitty-12

Contribution:

- Justin:
    - Designed the user, user/:userId, and liked_back route in index.js. Created the create.ejs and users.ejs page. Created the modal to open up information of users. Made the matches table in the backend. Made preferences and users be displayed on discover/users route. Helped reformat api call so that each user has a unique profile photo. Wrote report, made slides, deployed to Microsoft Azure

- Kevin
    - Created and designed the home page so that when the logo on the nav bar is clicked, it will redirect so the user can access login/reg easier. Coded the backend for the preferences page which is the /update_preferences route. Inserted the logo and designed navbar/menu.ejs. Helped make liked_back and liked_you route. Also designed the login/register and preferences page to look more professional. Created the user stories to keep track of what we have done.

- Austin
    - Managed the Github as Scrum Master, mostly making sure the merges were correct and we were accurately updating the files properly. Worked on the API routes and backend code in index.js. Contributed to the likes functionality and the

preferences functionality. Created the discover page with Jonny to use our API. Created the very sweet logo you see on our website. Helped make the likes and made the matches backend.

- Jonny
    - Spearheaded the discover page, which is a remote api call to a site that finds people that are available for being potential roommates. Coded the discover page to display cards with all of the information from the remote api, and implemented the modal buttons to align in each card. Wrote the tests for the website for lab 11 in the server.spec.js file. Set up the code skeleton at the start of the project, and set up the proper directory structure for the website. Coded the docker yaml file and package.json file.
- Benjamin
    - All animations that appear on the website were done by me, using an additional remorse library anime.js which can call any html element in javascript to be moved around on the screen. So much so that I was able to create a “background” ejs file to be used as a partial, to be called into any html page to be used as a background “layer” to make a pattern for the background layer. Manually creating elements for any pixel on the screen that follows the slope lined slope on the screen as well as the translation and perpendicular slope. Then those elements are called into an array in javascript to be moved on screen by the animation library.
- Alexis
    - Created our sql database with multiple foreign keys and differing data types. Also created the sorting algorithm in discover to sort the users based on the current users preferred age and preferred gender. Worked on index.js to get the information for the current preferences. Edited the logout page and the discover page. Made sure the preferences page update the discovery algorithm to search for the new preferences. Connected each user to their preferences.

Test plan:
Unit Testing: Mocha and Chai
Test cases: Login, Register,
Test environment: VS Code, Google Chrome, Node

	Positive Test Case	Expected Test Result (Positive)	Negative Test Case	Expected Test Result (Negative)	

Login	The user should be able to successfully login to the website if their login credentials are valid. This test is to ensure that the login function works as we intended and users are able to gain access to the other features of the website.	Should send a redirect status 302 if credentials are actually stored.	Checks to make sure that our system correctly handles incomplete login submissions for users attempting to login. The goal is the user cannot login without first providing all essential information which would be the username and password. Redirect to error message if user inputs wrong username or password.	The user should see an error message and the page should stay on login.	
Register	This test case is to confirm that a user can successfully register a new account into the database, if the register is successful then the user should be redirected to the login page. This test case is designed to check if someone is actually inserting a new account into our database of users.	The user is redirected to the login page so they can login once the registration is successful.	The goal of this test case is to ensure the system properly handles the case of a user registering with a username that already exists in the database. The user should be prompted to enter a new username or password and the page does not redirect to the login page until a unique username is provided.	The system should stay on the register page with an error about duplicate usernames showing up.	
Discover	When successfully logged in the user the discover page presents users with information about potential roommates. This test is to ensure users can explore suitable roommates that fit their needs.	The user should be able to see a list of places looking for roommates.	N/A	N/A	

Matches	This test case is to ensure the matches work. If it works, the request should contain an array of data that contains likes you and matches. Likes you is if they sent you a like. Matched is if you both send each other a like.	Should append to the array and display on the screen.	The goal of this test case is to make sure the system handles the reading of likes and matches properly.	The page wouldn't render anything for matches or likes.
update_preferences	This test case makes sure the preferences completely update and can change with each	Should update the database with new	The goal of this case is to make sure the system handles updating the	Returns a json "Internal server error"

	profile logged in and changed through the create path.	preferences set.	preferences correctly.	
users	This test case makes sure that users and preferences are joined and sends the data through ejs and renders it into the page.	Updates the results with usersWithPrefs.	The goal of this case is to choose the number of users out of the database along with their preferences.	Returns a console error and a status 500 error.
Save_and _referesh	This test case makes it so that it inserts matches from liked users to the liker into the database.	Redirects users to the page /users.	The goal of this case is to make sure the database addition of matches is done correctly.	Returns the error in console
liked_you	This test case fetches the people who have liked the logged in user and shows it on the page.	Renders the matches page and passes on the likedYouUserIds to the page and console logged the people.	This case is used to return the list of users that liked you.	Returns error in console
liked_back	This test case queries the database to find users that the current users has liked and users who have the current user liked.	It finds the intersection of two sets and logs the result. If successful it renders a page displaying matched users.	This case is used to check if the response from liked_back/matches is working.	Returns a 500 internal service error response if it fails.
logout	This test case is used to see if the user is successfully logged out	Destroys session and renders a simple logout page	N/A	N/A