CS 4530 Final Project: User Registration

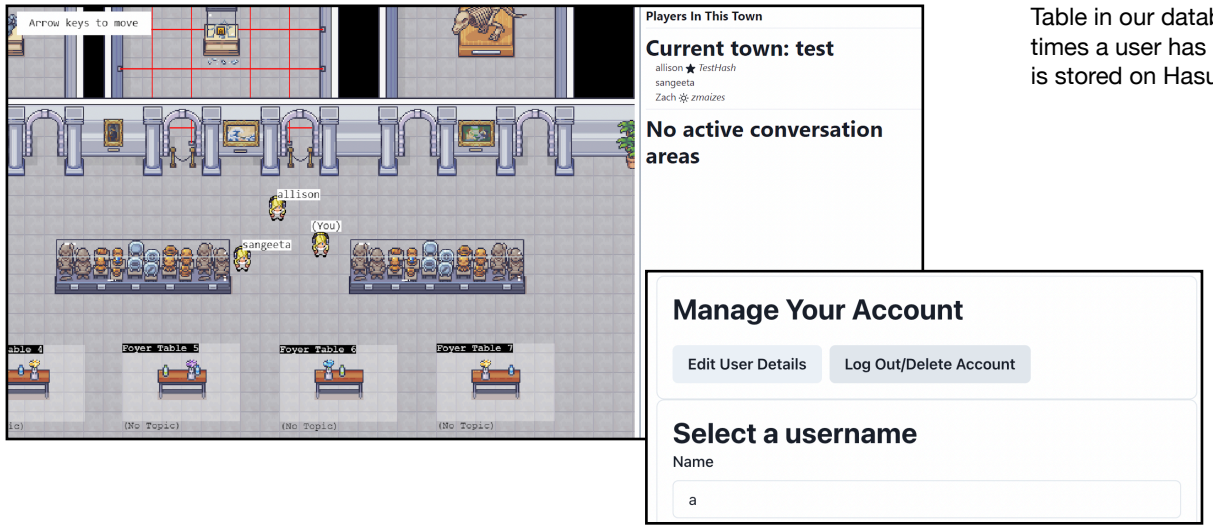
Group 2C - Allison, Zachary, Sangeeta

Our Feature: User Registration and Login

In the original release of Covey.Town, users could select a username, connect to a town, where they can walk around and interact with other users through voice calls, video calls, and newly implemented conversation areas. We realized that some users who frequently use the Covey.Town app may want to save their user profiles. Also, users can choose any username, even one that is already being used by another player. This could be inconvenient to returning users trying to find others they may already frequently connected with on the app, since multiple users may have the same username.

We developed a feature that allows users to register with a unique username, a nickname and a password. Registered users can also select an avatar. If a user does not want to register, they can still continue to use the app as a guest, but they will not be able to select an avatar. A registered user will also not be logged out of their session if they refresh the page or close and reopen the browser. There is also an option to logout or delete account, once one is created.

Demo and Source Code



Our demo is available at: https://main--lighthearted-mermaid-653606.netlify.app/ and our source code is located at: https://github.com/neu-cs4530-s22/team-project-2c. In the left image above, Allison and Zachary are registered players with usernames ‘TestHash’ and ‘zmaizes’ and nicknames ‘allison’ and ‘Zach’ respectively. They both also have avatars next to their nickname since they are registered. ‘sangeeta’ is not a registered user and is using the app with the original guest functionality. The image to the right is what the home page will look like if you are registered. You will be able to edit your avatar or log out/delete your account in the Manage your Account section of the main page.

Our Technology

We implemented the user registration feature using the existing codebase and by connecting the backend to a database through REST API calls. This Postgres database was created using Hasura Cloud. The front end user registration and login implementation was done using React states based on login status. Based on whether the user was logged in, react was able to render different modals, allowing users to log in as a guest or log out as a user, among other options. ‘Account name’, ‘username’, and ‘avatar’ field values were added to the player’s constructor to account for players who are registered. We also used local storage to save a players credentials so they do not get logged out from their session unless another login with the same credentials are detected or they manually log out.

Username	Nickname	Password	LoginIdx	Token
TestHash	allison	54410cdc2af6e19331609beae42204dd668e26c4	46	b9ddd77e82c8f63afe11
usr13	nic	pas	12	usr1311
TestUser	testingUgh	testPassword	43	token1
test1	test	b444ac06613fc8d63795be9ad0beaf55011936ac	2	100c4e57374fc998e571
test2user	test2nick	test2pass	1	test2user
user	nickname	pass	3	token

Table in our database that stores player username, nickname, password, loginidx (number of times a user has logged in), and token (generated upon user registration). Our Postgres database is stored on Hasura Cloud.

Future Work

Since we assumed that the main benefit of using Hasura Cloud is the access to instant GraphQL API’s, we spent more time than we originally anticipated on determining how to use GraphQL on our database and ultimately were not able to figure it out and resorted to switching back to REST API calls. Future work might include implementing the API calls to the Postgres database using GraphQL, rather than Rest.

Lastly, we struggled to think of a way to update a player’s avatar in the time that we were given. A player can currently choose a number between 1-5, each corresponding to a different avatar, which is then shown besides the players’ nickname. A next step could be allowing a registered user to update their player icon that moves throughout the town.

Now that we have a database that stores user profiles, the functionality of users can be extended. Registered users should be able to connect with other users by adding and removing friends, that would then be displayed in a new section on the screen showing a user all of their friends. Also, a user could have could have a stats/achievements section based on days active or any other criteria.