**CS5003 - Programming Projects - Assignment 2: Craic**

Student number: 130018883 Words: 485

For this project, I worked mainly on the server side scripting. My first task was to find a way of acknowledging when a user was or was not signed into application; I researched this and found cookies and session variables the options available. I opted for a session variable, and added some lines to our code that set this upon login and deleted it upon logout. As a team we then developed a function that ran on window load to check for this variable, and alter the page accordingly.

When working on the server side, and testing the application, I noticed that a lot of MongoDB connections were remaining open after use; e.g. sometimes over 100 if the application was running for a few minutes. I brought this to the attention of the team and we investigated; finding that MongoDB was by default opening a pool of five connections. We found how to manually open a single connection, and edited the code accordingly

After this, I made an addendum to the registration process; while registration worked correctly, there was no validation performed on the user details, meaning duplicate usernames were possible, something we wished to prevent. I added the checkUsername() function to be called before userRegister() on the server side, that returned an error if the desired username was already taken.

My next task was to streamline the following process. Functions to follow and unfollow users had been added, had associated buttons on the front end, and worked as expected. However, users could follow/unfollow themselves, and one user could follow/unfollow another user numerous times, which is not what we wanted. Upon investigation and testing, I decided that the most elegant solution would be to hide the follow and unfollow buttons from a users own page, and only show a single button on other users pages, depending on if they were following them or not. I added the checkFriendship() function to the series of callbacks on the server side so that the client side had access to this information. If true (i.e. logged in user follows viewed user), then the follow button was hidden and the unfollow was shown; if false the converse was applied.

I added a favicon to the HTML file to improve the appearance, and replace the default page icon displayed in the tab. I also went through the server side JavaScript and cleaned up the code, removed the debugging console.log()’s, and making it more structured, and ordered, as well as adding brief comments for all functions.

Finally, in the interest of security I looked for a solution that would encrypt or hash the passwords when stored in our database. I found the password-hash package to suit perfectly, and be reasonably straightforward to use. I altered the registering and logging in code to make an account for this, and only stored password information in the database after encryption.