April 6, 2016

TutorMe – Project Report

|  |
| --- |
| Team Members |

|  |  |
| --- | --- |
| Member Name | CDF ID |
| Felicia Hioe |  |
| Flavio da Silva |  |
| Rodrigo Guimarães | C5longhi |
| Zhi Chen |  |
|  |  |
| Application Concept | |

TutorMe is a platform to connect tutors and tutored. The idea is to have a platform where students can be tutored by other students to acquire the most diverse skills and to improve their performance in school. As a first version, only high school subjects are available to be tutored, but this can soon and easily be expanded to many areas of the human knowledge.

Each person is able to sign up (either trough TutorMe own sign up system or using Facebook). This process includes choosing all the free-times the person has to tutor other people and choosing which subjects you are capable of tutoring.

After logging in, the user is granted with a homepage where he can search for other users, as well as search for tutors by subject in which he is interested. The user is also able to access his messages, view/edit his profile and review a tutor on his homepage.

When searching for a tutor, the user is able to select a particular user, where he can see the tutor profile and reviews. There, he can also choose to see the schedule to book and appointment or go to the chat system, where he can communicate with the tutor.

The application also has an admin page, for database administration purposes.

|  |
| --- |
| A - Application Design |

The application is built using a Node.JS server, a mongo database and the following frameworks: express, socket.io, mongoose, cookie-parser, cookie-signature and compression.

The main structure of the server is to use a RESTFUL API, where the URL routes are redirected to the correct part of the program using the router of the express framework. Pages are fetched using the get method of the router, and some posting to the database is done by using the post method. Also, some communication is done using the socket.io framework, that creates a socket permitting clients send and receive information to/from the server. Each message received by the socket.io framework has an identifier that makes it possible to treat different messages with different code snippets. The socket.io socket is used to transfer information that requires JSON objects to be transferred, such as the objects containing a user schedule or the object with the complete person information, when it is signing up.

Pages that require the user to be logged in to have access do this by checking a cookie. When a user first logs in, the server generates a signed cookie and saves to the user’s browser. Each time a user accesses a page that requires the user to be logged in, the server reads the request cookie and checks for the username and cookie validity. If the page requires admin privileges, this is where the server checks too. The cookie stores the username of the user only, and a verification string to check for validity.

Each route renders the correct view according to the request parameters, and the differences in the page are passed by HTML variables. These HTML variables are parsed on the JavaScript folder and the page can dynamically change itself to display the username of the user that is currently logged in, different database information and others.

|  |
| --- |
| B - Security |

|  |
| --- |
| C - Performance |

|  |
| --- |
| D - Video |

|  |
| --- |
| E – MORE |