**PROJECT REPORT**

Name: Roshan Sajja

ID: 101168617

Name: Ricky Gulati

ID: 101146071

Our OpenStack instance information is given below:

Instance name: RickyGulati

ID: ab35c551-7a61-4ea3-a367-64da99a3b0e9

Floating IP: 134.117.131.185

Username: student

Password: student

Instructions for TA: All npm modules have been installed already and the project files are in home/student/public

We have deployed our server to an OpenStack instance, and we made sure its functionality is still operational on the OpenStack instance.

We designed our REST API for users properly, our web app supports all the user functionality like creating a new user, searching for a user, adding a user, removing a friend, logging in and logging out.

For our REST API we used server request paths following logical RESTful API rules by using paths like POST(/users/:uid/requests) for sending a friend request, POST(/users/:uid/friend/:reqid) for adding a friend etc. which allowed us to navigate through our webpages easily.

The user functionality that we have implemented handles bad data adequately as far as we have tested.

We also implemented the JSON API by specifying specific types of responses for html rendering and JSON use.

We also styled our webpages with advanced CSS, with minor bugs.

uses express server and pug template engine from the npm module package, which allowed us to send data to our PUG templates from the server and it also allowed us to use different URLs and methods along with making the code easier to write. It also allowed us to add session support to the server.

Unfortunately we started working on the project late, due to which we were unable to implement the CONNECT4 game logic, but we have implemented most of the user profile functionality like creating a new user, logging in, logging out, sending friend requests, rejecting friend requests, accepting them and making friends, removing friends and searching for other users.

We believe that if given more time we would be able to implement the game logic through the server using http requests and make the web pages more impressive through CSS.

I would say that we liked designing the webpages using CSS which made them look better than the normal HTML pages. Then later when we were making our business logic, we were worried about implementing all these functions, but then one after another we linked all the business logic’s user functionality to the webpages and learned how to make changes using the data in JSON files. Our best feature is making friends because we made the functions robust so that it can handle errors, it goes through a process, like first we send a request to a user who’s profile is public and then he can either accept or reject the request. Another good feature is that it’s a user friendly webpage as all things are accessible easily.