Milestone 4
Team: Yahooligans!

Automated Test Cases-

- We are using TravisCI to implement continuous integration in our web app
- **https://travis-ci.org/Yahoooligans/ScrumMasterLove.com/builds**

User Acceptance Test Plans

- Feature #1 – Successfully register a new user
  - User activity –
    - The user accesses the website
    - They click a button to sign up
    - They fill out the necessary information on the sign-up form
    - They are prompted to fill out a questionnaire to find matches
    - They fill out the necessary information on the questionnaire
  - Data –
    - Server.js spins up
    - Index.js handles a GET request and Index.ejs is rendered
    - A button to sign up is clicked and a route to Questionnaire.js is executed
    - Questionnaire.js renders Questionnaire.ejs and a form is displayed.
    - The user fills out the form and questionnaire.js handles the POST request and stores all data, profile info and answer to the questions, into the database.
- Feature #2 – Successfully view and edit their profile
  - User activity –
    - The user selects the link to show their profile
    - They are able to view all their information
    - They are also able to edit their information by clicking a button
    - They can change info and then click a button to save
    - Their information is updated and can now view it
  - Data –
    - Server.js spins up
    - Index.js handles a GET request and renders Index.ejs
    - A button to login is clicked and a route to login.js is executed
    - Login.js renders login.ejs and the user is able to enter their info and login
    - The user is redirected to their homepage
    - From the homepage the user can select to view their profile
    - The link routes to profile.js which renders in profile.ejs
    - The information is then pulled from the database and the user is then able to see their information
    - They also have the option to edit their info
    - If so, index.js handles a GET and POST request to edit the user's info
    - They can change information and then click a button to save their changes in the database

        - The changes are recorded, the user is redirected back to profile, and the newly added/subtracted/changed information is displayed on their profile

- Feature #3 – Successfully view their matches
    - User activity –
        - After filling out the questionnaire, a user should be given an array of up to 10 matches
        - They should be able to click the link and get redirected to a match’s page.
        - Here they should be able to people who answered the questionnaire within a certain percentage
    - Data –
        - Server.js spins up
        - Index.js handles a GET request and renders Index.ejs
        - A button to login is clicked and a route to login.js is executed
        - Login.js renders login.ejs and the user is able to enter their info and login
        - The user is redirected to their homepage
        - From the homepage the user can select to view their matches
        - The link routes to matches.js which renders in matches.ejs
        - This page displays up to 10 of the users matches based on a percent closeness in response based on the questionnaire.