EvaluRate

The Hack Jacks

Dylan Grayson
Conner Swann
Brandon Paree
Brian Saganey

Test Plan Document

April 19, 2016
V 1.0

Table of Contents

Introduction

EvaluRate is a fairly complex web application that aggregates data from multiple different users, and has a strict permissions hierarchy. It also relies on the usability of a front end application that faces two distinct user groups. For these reasons, our primary focus for testing EvaluRate will be on:

- Unit testing - Testing for code correctness
- Usability testing - Testing for usability from target users

Our unit testing can check our math, and our permissions, to ensure the application is functioning as intended, and the usability testing will give us feedback on how well our user interface is working for its intended users.

Unit Testing

Since EvaluRate is built with Meteor.JS, all unit testing is performed by the node-based unit testing framework Mocha. Using the framework, we will be testing EvaluRate in two major ways. At the method-level, we will be performing simple unit testing. However, due to the closely integrated nature of the client and server, simple unit testing does not adequately capture the range of possible component interactions. In other words, testing only on the server or only the client is insufficient, so Meteor provides a testing environment that lets us load both client and server code to their respective areas while also isolating those components from the rest of the application. In this way, we will also be performing simple integration tests to ensure that while the individual components work correctly, they also work together in a manner that is expected. The two main components of the application that need to be tested are as follows:

- Result Scores - The weighted average of evaluation data for each member on a team.
- Permissions - The restriction and denial of different administrative actions on any given unit.

Result Score Testing

EvaluRate features a visual results section that tabulates data from evaluations and displays a calculated average based on certain data points in accordance with some weighting metric. The application depends on the correctness of these values, so they will be thoroughly tested on all built in evaluations, with a wide range of user inputs.

Permissions Testing

The anonymous nature of peer evaluations in EvaluRate means that our organizational units need a robust permissions system. This is to ensure that our users never see what their peers rated them, as this would defeat the purpose of anonymous peer evaluations. The permissions system includes two different types of administrators, and every organizational unit contains its own set of permissions pertaining to both types of administrator. The testing for this will include many different organizational structures with many different arrangements of administrators.

Usability Testing

In order to decide how the usability for EvaluRate should be tested, we first had to consider who would be using the application. There are two obvious divisions in the functions of the user interface. The first function facilitates administrators/managers of projects/teams to issue evaluations to the members in those teams, and receive the data for viewing. The second function allows the members of those teams to fill out and submit the evaluations. The usability testing for EvaluRate will include user feedback, and in-lab user testing from two user groups we have easy access to:

- Professors: They fit right in our user group for project/team administrators, as they often assign group projects, and may desire peer evaluations.
- Students: They are the user group that will be put into teams on projects, therefore taking evaluations.

These two user groups will provide necessary feedback that will inform as to whether or not our user interface is working.

Professors

Clearly in our case, the user interface for professors is far more complicated than it is for students. For that reason, we plan to spend the majority of our usability testing time with professors, to make sure the process of creating a project and assigning an evaluation makes sense to them. The professors should find it easy to form a hierarchy with one or more classes corresponding to classes in reality. They should then be able to add the appropriate students to the appropriate classes, and create a project with teams consisting of those members. The professor should then be able to issue an evaluation to that project, and see resulting data as students complete the evaluation. We will test this usability through expert reviews with professors. By walking them through the interface and taking notes on where their understanding breaks down, we can fix any major issues in the interface.

Students

The user interface for students is very simple as long as they do not administer any team. The most important aspect of this portion of the user interface is comprehensive notifications. Students should be notified when an evaluation is made available, and when it is approaching it’s due date. We will test this functionality by doing very brief and informal walkthroughs with fellow students.