Prepared by:

Francisco Radilla Greer, Robert McNabb, Caleb Britten, Armando Magana, Joshua Wilding

6-2-2020

OBJECTIVES AND TASKS

Objectives

The objective is to provide a testing suite for the Caliber Test application. The testing suite should provide unit tests, integration tests, and system tests (E2E). The testing suite must allow for regression testing. Because this team is using test-driven development, the tests should provide a clear target for application development.

Tasks

The team will write backend tests using the JUnit framework. The team will write frontend tests using the Jasmine framework and automate tests using Selenium and Protractor. The test suite will have test cases defined in the test case design. The test case design will have the following sections: Use Case ID, Use Case Scenario, Test Case ID, Test, Steps to Test, Expected Result, Actual Result, Status, Defect ID, and notes. Developers will write one test per test case.

SCOPE

General

This testing suite will include unit tests. This suite will also include integration testing and system testing. Tests will be for the website frontend and website backend.

Tactics

The team will create all tests following a collaborative strategy. The team will prepare a collaborative test strategy document using Google Sheets and Google Docs. The team will use Asana to organize task progress. The team will divide labor into roughly equal sections. If a member finishes their task early, that member should assist other members with their tasks.

TESTING STRATEGY

The approach to testing emphasizes collaborative development, agile development, and test-driven development. The team will meet for daily stand-ups to communicate about project status. Team members will assist one another as needed, and communicate about any problems as soon as possible. The team will organize tasks on an Asana board. Tasks on Asana will each correspond to user stories. Test cases will derive from user stories.

Tests will follow requirements through use of a requirements traceability matrix. The requirements traceability matrix will include a Business Requirement ID Number, a Requirement Description, a Test Case ID, and a Status.

The team will measure backend test coverage using JaCoCo. The team will submit a report prepared using JaCoCo after creating all backend tests. Developers will measure the percentage of non-trivial methods with corresponding tests. Trivial methods are any simple method created as part of boilerplate code. Examples of trivial methods include basic accessors, basic modifiers, and the toString method considered part of the standard Java Bean design pattern.

Unit Testing

Definition:

Unit tests for the front end and the back end will focus on atomic portions of code. These portions of code should each reflect a single test case defined in the test strategy document. Developers will write a series of test cases for each user story. Every test case in every user story must have a unit test. The team will record all tests for their corresponding test case entry. The record for any failed tests must include notes and a defect ID.

Participants:

The testing team will be responsible for unit tests.

Methodology:

The testing team will write backend unit tests in JUnit and frontend tests in Protractor.

System and Integration Testing

Definition:

Integration testing will include any testing for more than a single test case. System testing will include any end-to-end testing.

Participants:

The testing team will write all integration tests and system tests.

Methodology:

The team will write frontend integration tests using Jasmine. The team will write backend integration tests using JUnit. End-to-end tests will be automated using Selenium.

User Acceptance Testing

Definition:

User acceptance testing will consist of black-box testing conducted by real users.

Participants:

The testing team will find volunteers to conduct user acceptance testing.

Methodology:

The testing team will find volunteers from other departments in the company to act as end-users.

Automated Regression Testing

Definition:

Regression testing is the process of running existing tests to ensure changes do not change the results of any established working code.

Participants:

The testing team will be responsible for regression testing.

Methodology:

The team will run regression tests as part of maintenance and updates. Developers will never commit code that fails regression tests to a production pipeline or source branch.

HARDWARE REQUIREMENTS

For this project a laptop or computer is needed. This device needs a 64 bit processor with 8+ GB of ram. This device can have any kind of operating system.

ENVIRONMENT REQUIREMENTS

With this device there are certain software that is needed to be able to run these tests. The first piece of software is a java JDK 8. This is used to compile and run the java application from the command line if needed. To view the code use Java Eclipse IDE. To actually run these tests, there are libraries needed. The three libraries are Junit, Natural 0.9, and Selenium.

Here are links on how to acquire these additional software

* Java JDK 8 installer: <https://www.oracle.com/technetwork/java/javase/overview/java8-2100321.html>
* Java Eclipse installer: <https://www.eclipse.org/downloads/packages/>
* JUnit: <https://junit.org/junit4/>
* TestNG: <https://testng.org/doc/>
* Natural 0.9: From eclipse marketplace
* Selenium: https://www.selenium.dev/downloads/

TEST SCHEDULE

For this project there are three milestones. Each milestone focuses on a different section of the web site. Milestone one focuses on the home page and manages batch components. This milestone will take 3 days to complete its tests. Home page will take 1 day while manages batch will take 2 days. Milestone two focuses on assessing batch, quality audit, and panel components. This milestone will take 5 days and will be split up with assess batch using 3 days while the other components need 1 day. The final milestone focuses on reports and settings panel components. This milestone will take 4 days with setting needs 3 days while 1 for reports. Each of these milestones will be using java eclipse with junit, selenium, and natural.

CONTROL PROCEDURES

Problem Reporting

Problem reporting will be an extensive procedure that this testing team will implement. Any time a bug or defect is caught during the testing it will be discussed by the entirety of the testing team, even if caught in a smaller subgroup. This will be done to ensure that no one left out in the testing subgroup of this particular bug or defect does not have a different perspective to approach or fix it. Once the testing team has decided they are incapable of fixing the bug or the defect they will send it back through the software defect life cycle to be handled by the development team.

Change Requests

When making changes to the existing project, the testing team will follow a very principled and arduous process. Making changes to the existing software should be taken seriously, and the testing team will exemplify that. Before any change is made to the existing document by the testing team, mock, unit, and regression tests will be created to ensure the individual functions work as well as when they are integrated with the existing project. The client will be notified of any changes being made before, during, and after completion. Moreover, the development team will be brought in to confer with to ensure the proper design of this code. They will, as well, be notified of the changes before, during, and after completion.

FEATURES TO BE TESTED

Home Page:

* Last Quality Audit:
  + Shows different data upon switching the state.
  + Shows unique data pertaining to that specific state.
  + Shows correct numbers when hovering over each bar on the graph.
* Weekly Progress:
  + Shows different data upon switching the state.
  + Shows unique data pertaining to that specific state.

Manage Batch Page:

* Trainees in batch button:
  + It works.
  + The number shown with it is correct.
  + The necessary information is provided.
  + Switch batch, update trainee, and delete trainee buttons work.
  + Add trainee button works.
  + Active trigger works.
* Update Batch button works
  + All necessary information is:
    - Provided
    - Appropriate
* Delete batch button works
  + Batch is deleted when you press yes.
  + Batch is NOT deleted when you hit no.
* Create batch button works
  + Can only submit when all information is:
    - Supplied that is required.
    - Formatted correctly
  + Submits when you hit submit
  + Doesn’t when you hit cancel

Assess Batch Page:

* Relevant information is displayed when you select:
  + Different years
  + Different trainers
  + Different weeks
* For each individual trainee, you can:
  + You can change a number in the provided fields.
  + The averages change correctly when you change these numbers.
  + You can enter a note in the notes section.
  + All of this above information saves when you click the save button.
* You can provide overall feedback for each batch.
* You can add a new week for a selected trainer and specific year.
* You can create an assessment:
  + Save works when you input the required fields
  + Close cancels the new assessment
* You can edit a given assessment upon clicking it
  + Save button works
  + Close button works.
  + Delete button works.
* You can import grades using a JSON format.

Quality Audit Page:

* Relevant information is displayed when you select:
  + Different years
  + Different trainers
  + Different weeks
* For each individual trainee, you can:
  + Click name and flag them
  + Click Feedback button and change the icon
  + Write notes on them.
* You can add a new week for a given batch
* You can provide overall feedback on the batch.

Reports Page:

* Relevant information is displayed when you select:
  + Different years
  + Different trainers
  + Different weeks
  + Different Trainees
* You can download the PDF of the Cumulative scores
* Appropriate scores show when hovering over the bars
* Can download the PDF for the technical skills.
* Can download the PDF of the weekly progress.

Settings Page:

* Trainers:
* You can create a trainer when:
  + All information is:
    - Included that is required
    - In an appropriate format.
  + Save button submits trainer
  + Close button cancels
* You can update a trainer when:
  + All information is:
    - Included that is required
    - In an appropriate format.
  + Save button submits trainer
  + Close button cancels
* You can delete a trainer:
  + Delete does just that
  + Close cancels the deletion
* Locations:
  + You can create a location when:
    - All information is:
      * Included that is required
      * In an appropriate format.
    - Save button submits location
    - Close button cancels
  + You can update a location when:
    - All information is:
      * Included that is required
      * In an appropriate format.
    - Save button submits location
    - Close button cancels
  + You can delete a location :
    - Delete does just that
    - Close cancels the deletion
* Category:
  + You can edit a category by clicking on it and submitting the required information
    - You can also change the status of a category in the edit menu.
  + You can create a new category when:
    - You input the type and hit save
    - Close cancels this submission.

FEATURES NOT TO BE TESTED

Home Page:

* Panel Progress
  + Reason: This part of the program is left blank.

Manage Batch page:

* Import batch.
  + Reason: No batches provided to test.

Panel Page

* Entire page
  + Reason: Nothing here yet.

Reports Page:

* Search bar:
  + Reason: Doesn’t work--nothing searches
* The PDF button next to the search bar.
  + Reason: Drop down list isn’t clickable.
* Batch QC Report
  + Reason: No information provided
* Weekly Progress:
  + Reason: No information provided
* Panel Feedback:
  + Reason: No information provided

RESOURCES/ROLES & RESPONSIBILITIES

Everyone on the team is responsible for setting up their own test environment and the test cases have been split up to all members of the team.

SCHEDULES

Deliverables

What we are delivering is the test plan and test case document and traceability matrix

DEPENDENCIES

What are some of the constraints or limitations on testing?

We are limited to the time of the project.

We aren't given any backend information so we can;t write actual unit tests since we don’t know what to call or what method names are.

RISKS/ASSUMPTIONS

Possible risk of overlap of tests between features.

Mitigated by proper communication between testing team members.

TOOLS

Eclipse IDE, TestNG, JUnit, Mockito, Cucumber, Jasmine, Protractor, Selenium, and Postman

APPROVALS

Project Manager:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_