Student Lap Tracker

Test Plan

Patrick Oyarzun, Joe DeHart
Vaishali Patel, Duke Meche, Robert Knott

ABSTRACT

This document describes the techniques, tools, and reasoning behind testing the Lap Tracker application.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

INTRODUCTION

This document describes the techniques, tools, and reasoning behind testing the Lap Tracker application. The nature of a client-server MVC model is that all functionality is implemented on the server and exposed through a thin layer on the client. Because of this, the test plan is composed of two major parts, first are integration tests which are carried out entirely from the client side. These are meant to test the system as a whole and are meant to know nothing of the implementation. The integration tests also include a penetration-testing section to ensure no unauthorized data access or retrieval is possible.

The second major part is the set of unit tests for the internal system implementation. Whenever possible, these tests are still carried out as ‘black-box’ tests. However, some particular tests may need to use knowledge of the implementation in order to verify the application invariants are maintained. This is mostly due to the datastore work that happens internally.

REQUIREMENTS/SPECIFICATIONS-BASED SYSTEM LEVEL TEST CASES

Integration Tests

Authentication

Visit Homepage without Authentication

- Expected Output
    - Since the request handlers are decorated with either @login_required or @admin_required, the user will be redirected to a google account login page.

Visit Homepage with Authentication

- Expected Output
    - A list of links for each of the other pages: import, export, lap tracking, and milestone configuration.

Visit All Pages Similarly

Logout, repeat

Import

Upload Correct Workbook

- Expected Output
    - After importing an excel spreadsheet, the contents of the file will be displayed on the page, if processed successfully.

Upload Workbook with mismatched teacher names

- Expected Output
    - Displays an error message that the teacher name does not match with the database record.

Upload Workbook with mismatched student names

- Expected Output
    - Displays an error message that the student name does not match with the database record.

Export

Download All Students

- Expected Output
    - Displays the data of all the students, if drop down menu selection is all students.

Download Single Student

- Expected Output:
    - If drop down menu selection is the student id of any particular student, then it will only display the data of a particular student associated with that selected id.

Tracking UI

Scan Barcode

- Expected Output

- After scanning a student's ID barcode, a success message is displayed and the scan is added into the Recent Scan window.

Undo a repeated scan

- Expected Output
  - If a scan was unnecessary or repeated by accident, the user can remove the scan by clicking the delete button on the scan's success message in the Recent Scan window

Milestones

Display Milestones

- Expected Output
  - A list/table of the milestones that are currently active.

Add Milestone

- Expected Output
  - A new row with the milestone specifications is added to the list/table. A success message appears briefly to let the user know the addition was successful.

Delete Milestone

- Expected Output
  - The specified milestone is removed from the list/table. A success message appears briefly to let the user know the removal was successful.

Modifying Milestone

- Expected Output
  - The milestone's specifications are changed to the user's new requirements. A success message appears briefly to let the user know the change was successful.

Cron Emails

Summary Email of Milestone Progress (Weekly Fridays after school)

- Expected Output
  - This will be called weekly after school and will send an email to the teacher containing a milestone progress report which may look like the following:

  Weekly Milestone Progress Report:

Teacher Name	Laps	Laps to Go Until Next MileStone
Duke Meche	61	39
Vaishali Patel	74	26
...		
Patrick Oyarzun	79	21

Trigger Milestone (Daily weekdays after school)

- Expected Output
  - In administrator console:

The Google App Engine administrator console provides a way to check the functionality of each cron job that is active. It provides us with the latest scheduled tasks, when they are scheduled to be executed, and whether or not the job was successful.

/crontask	every day 16:30 (US/Central)
daily milestone check	2014/11/07 16:30:00 **on time Success**

- Other:
    1. If the milestones have not been reached by any student, nothing is done.
    2. If the milestones have been reached, the teacher will be notified via email which teacher has passed the milestones and which class should be rewarded.

Unit Tests for Internal Server Implementation

Import

Parse Correctly formatted Workbook

- Expected Output
    - The datastore contains a Teacher entity for each teacher in the spreadsheet.
    - The datastore contains a Student entity for each student in the spreadsheet.
    - Each student entity has the correct counts for track 1 and track 2.

Parse Workbook with mismatched teacher names

- Expected Output
    - Nothing is stored in the datastore.
    - An error message is returned naming the teacher which was not able to be matched.

Parse Workbook with mismatched student names

- Expected Output
    - Nothing is stored in the datastore.
    - An error message is returned naming the teacher which was not able to be matched.

Export

Download All Students

- Expected Output
    - A single Excel Workbook (.xlsx) is generated and it contains the entire contents of the database in the same format as the original uploaded spreadsheet.

Download Single Student

- Expected Output
    - The student object is returned singularly. This functionality does not generate an Excel Workbook as that would be superfluous for a single row.

Tracking UI

Scan Barcode

- Expected Output

Undo a repeated scan

- Expected Output

Milestones

List Milestones

- Expected Output

Add Milestone

- Expected Output
    - The new milestone is added to the milestone table in the Datastore.

Delete Milestone

- Expected Output
    - The milestone chosen by the user is removed from the Datastore.

Modifying Milestone

- Expected Output
    - Milestones are changed to match new requirements in the Datastore.

Cron Emails

Summary Email of Milestone Progress (Weekly Fridays after school)

- Expected Output
    - Success message in admin console under cron jobs.

Trigger Milestone (Daily weekdays after school)

- Expected Output
    - Nothing if the milestones have not been met.
    - If the milestones are met, an email is sent to the teacher with the name(s) of the teacher(s) whose class has passed the milestones.

TRACEABILITY OF TEST CASES TO USE CASES

Each of the original three use cases is covered by at least 1 integration test and several unit tests internally. These include Import, Export, and Track. This verifies that the spec is being followed. Additionally, the access control requirements are verified through some additional integration testing.

TECHNIQUES FOR TEST GENERATION

The integration tests are all treated as black-box tests. This is a consequence of a client-server architecture. In order to verify the client side implementation, we must not

depend on any implementation detail. The client has no access to the internal application state without passing through the MVC architecture on the server.

The unit tests on the other hand must use internal implementation knowledge to verify the pre and post conditions of all functions are correct. It is not possible to verify a database table was properly updated without knowing which table is involved in the transaction.

Test quality is measured along a single axis: coverage. A test with 100% coverage is the ideal test. Coverage is a measure of how thorough a given test is. For example, any code branch which is not executed during the test process is considered to not be covered by the test suite. Not only is this a good measure of test quality, it is a simple technique which allows us to find uncovered code paths and fill in the gaps.

Where relevant, application invariants are listed along side the test cases for that section of code. Automated unit tests and user-driven integration tests validate that those invariants are maintained.

EVIDENCE THE TEST CASES, DOCUMENT HAVE BEEN PLACED UNDER CONFIGURATION MANAGEMENT

REFERENCES