Test Plan Document

for

Indiana University Southeast
School of Natural Sciences

Career Tracker

Table of Contents

1. Introduction

This document outlines the unit testing, user acceptance testing, and system testing to be employed prior to implementation of the Career Tracker application.

Unit testing will be completed using the Jasmine unit testing framework which allows for automated testing of critical application functionality and by manual testing. System, API, and database testing will be completed using various methods to be detailed within the testing documentation.

2. Business Background

Career Tracker is primarily a web application that allows a user to view, edit, add, and logically delete records related to university alumni data. This data is used by the University for various research efforts. The product also consists of backend server-side processes that allow for file processing as well as processing of CRUD requests received via the HTTP endpoints.

This testing effort is designed to ensure users are able to utilize the frontend user interface to perform the required operations within the specifications of the application using hardware/software consistent with the stated system requirements.

3. Test Objectives

The objectives of this testing effort are as follows:

- As thoroughly and efficiently as is reasonable, evaluate the application using specific test cases in order to ensure it is operating within stated requirements.
- Identify instances where the application's operation deviates from the stated requirements, allowing changes to be made to rectify such defects.
- Provide customer with documentation outlining in detail the functionality tested, the results of the testing, and actions that were taken to resolve any instances where the result was not as expected.

4. Scope

The scope of this testing effort specifically includes the following:

- Automated unit testing of functionality used in the operation of the application where such testing has been determined to be essential to ensure the product delivered operates in a way consistent with the stated requirements except where such testing is not possible or more efficiently accomplished by manual unit testing.
- Manual unit testing where it is determined such testing is essential and more viable or efficient than automated unit testing.
- System and API testing using various methods to evaluate application performance and to ensure application is operating in a way consistent with the stated requirements.

The components to be tested include:

- Record Maintenance
    - Data retrieval functions (frontend and backend)
    - Data validation functions (frontend and backend)
    - Associated API/database functions
- Data Imports/Exports
    - Import file processing functions (frontend and backend), including upload, file parsing, validation, and data commits
    - Export file processing, including criteria processing, file creation, and download
- Login/Authentication
    - Authentication functions (frontend and backend)
    - Authorization and session variable initialization functions (frontend and backend)

The scope of this testing effort specifically excludes the following:

- Comprehensive evaluation of non-functional components or attributes of the application.
- Testing of applications or libraries used in the development of the application beyond ensuring they allow the application to operate in a way consistent with the stated requirements.
- Evaluation of functionality within the application that is not essential for its operation within the stated requirements.

5. Test Types Identified

The types of tests to be used within this testing effort include:

- **Unit testing** – automated or manual testing of specific functionality within the application
- **System testing** – testing of entire application, including the validation of functionality using specific inputs and testing of the user experience
- **API testing** – testing of specific APIs used in the application, such as HTTP endpoints, file parsing, file generation, and interactions between application and database

6. Problems Perceived

While testing issues have not been seen as of this writing, problems expected to potentially hinder testing efforts are as follows:

- **Lack of common testing environment** – each development team member maintains their own copy of the application. This could result in inconsistent testing results due to testers potentially using different application/library versions.
- **Lack of testing time/resources** – as testing users for this application are very limited in quantity, it is possible lack of time may result in decision to forego some testing if the risk is found to be within acceptable limits.
- **Inconsistent testing experience/knowledge** – as the testing team consists of persons with varying levels of testing experience, it is conceivable that inconsistency in test coverage/results analysis may result.

7. Architecture

The architecture of the application test plan is comprised of the following:

Unit Testing

Automated

All functions that allow application to operate in a way consistent with the documented requirements and where such testing is determined viable and more efficient than manual unit testing.

Manual

All functions that allow application to operate in a way consistent with the documented requirements and where automated testing is determined to not be viable or is less efficient than manual testing.

System Testing

Manual and/or automated testing of the overall application to ensure it is operating in a way consistent with the documented requirements.

API Testing

Manual and/or automated testing to ensure all APIs utilized in the application allow it to operate in a way consistent with the stated requirements.

Test Case Execution

Test cases associated with this plan will be executed by members of the development team. The results of these tests will be logged within the test result documentation. All test results found to deviate from the expected result will be logged and defects prioritized for resolution.

8. Environment

The recommended system specifications for the operation of the application are as follows:

Client (User Interface)

Web Browsers (as consistent with Angular browser support):

Google Chrome – latest version

Firefox – latest version

Internet Explorer – version 11 or later

Safari – latest version

Note: This application was designed for use on a laptop or desktop computing device. The use of mobile devices is not recommended.

Operating system/memory/storage:

Refer to system requirements for above mentioned client applications

Display:

A resolution of 1280 x 800 or greater

Server

Node JS:

Version 13 or later

MySQL:

Version 8.0.18 or later

Libraries:

Angular version 8.2.14 or later
Node CSV
Passport JS

Operating system/memory/storage/other:

Refer to system requirements for above mentioned server applications/libraries

Automated unit testing to be completed using Karma version 4.1.0.

9. Assumptions

The following assumptions have been made in regards to this testing effort:

- Conditions and associated test results in test environment will be similar to those observed in the production environment.
- All instances where application is operating is a way inconsistent with the documented requirements will require root cause identification and resolution.
- Limited regression testing is likely necessary after a defect in the application has been rectified.
- Defects and test results to be documented and available for review in testing results documentation.

10. Functionality

The functionality targeted for testing as related to this test plan include:

Client Application (Web Interface)

- Ability for user to authenticate using username/password credentials and appropriate session variables are set
- Inability of a user that fails to authenticate using valid username/password credentials to access application resources, including HTTP endpoints
- Ability for a user to log out of the application, clearing session variables
- Ability for user to navigate site to access application functionality
- Ability for a user to search for alumni/employer/graduate school records using specified criteria
- Ability for a user to add or edit a record, including applicable data validation and commission to backend database
- Ability for a user to logically delete a record
- Ability for a user to upload an input file for server-side processing and receive any errors or a confirmation that the process completed successfully
- Ability for a user to download a file containing application data based upon specified criteria

Server Applications

- Ability for server to appropriately respond to HTTP resource requests
- Ability for server to establish a user session when a user successfully authenticates
- Ability for server to end a user session when log out functionality is invoked
- Ability for server to parse, validate data, commit data, and return results related to an uploaded input file

- Ability for server to prepare and return an output file containing records meeting specific criteria

11. Risks and Mitigations

The following risks and associated mitigations have been identified as related to this testing effort:

- **Risk:** Given that each testing user is using their own copy of the application, there is potential for inaccurate test results due to user not using most recent version of the application or associated libraries
    - **Mitigations:**
        - Test users to ensure that they are using the most recent version of the application and associated libraries before conducting tests
        - Test cases to be assigned to testing user in a way that reduces testing of overlapping components by multiple users
- **Risk:** Due to end user not being involved in testing effort, there is potential that tests do not accurately capture business expectations as related to documented requirements
    - **Mitigations:**
        - End user to be consulted/made aware of testing details, including tests identified, test criteria used, and results
        - Engagement with end user to ensure functional and non-functional attributes of web application are consistent with expectations
- **Risk:** Due to lack of time/resources, some testing may not be completed as expected
    - **Mitigations:**
        - Test cases to be distributed to test users in a way that helps ensure work is equitable among team members
        - Test users to keep other users consistently abreast of progress and obstacles so resources can be re-allocated accordingly
        - Some testing may be forgone if deemed not essential to application operating in a way consistent with documented requirements

12. Test Strategy

A high-level overview of the test strategy for components of the application is provided below.

Automated Unit Testing will be used to evaluate, in whole or part, functionality related to the following requirements:

Client Application (Web Interface)

- Ability for user to authenticate using username/password credentials and appropriate session variables are set
- Ability for a user to search for alumni/employer/graduate school records using specified criteria
- Ability for a user to add or edit a record, including applicable data validation and commission to backend database
- Ability for a user to logically delete a record

Server Applications

- Ability for server to appropriately respond to HTTP resource requests
- Ability for server to parse, validate data, commit data, and return results related to an uploaded input file
- Ability for server to prepare and return an output file containing records meeting specific criteria

Manual Unit Testing will be used to evaluate, in whole or part, functionality related to the following requirements:

Client Application (Web Interface)

- Ability for user to authenticate using username/password credentials and appropriate session variables are set
- Ability for a user to search for alumni/employer/graduate school records using specified criteria
- Ability for a user to add or edit a record, including applicable data validation and commission to backend database
- Ability for a user to logically delete a record
- Inability of a user that fails to authenticate using valid username/password credentials to access application resources, including HTTP endpoints
- Ability for a user to log out of the application, clearing session variables
- Ability for user to navigate site to access application functionality

- Ability for a user to upload an input file for server-side processing and receive any errors or a confirmation that the process completed successfully
- Ability for a user to download a file containing application data based upon specified criteria

Server Applications

- Ability for server to appropriately respond to HTTP resource requests
- Ability for server to parse, validate data, commit data, and return results related to an uploaded input file
- Ability for server to establish a user session when a user successfully authenticates
- Ability for server to end a user session when log out functionality is invoked
- Ability for server to parse, validate data, commit data, and return results related to an uploaded input file
- Ability for server to prepare and return an output file containing records meeting specific criteria

API Testing will be used to evaluate, in whole or part, API-related functionality related to the following requirements:

Client Application (Web Interface)

- Ability for user to authenticate using username/password credentials and appropriate session variables are set
- Ability for a user to search for alumni/employer/graduate school records using specified criteria
- Ability for a user to add or edit a record, including applicable data validation and commission to backend database
- Ability for a user to logically delete a record

Server Applications

- Ability for server to appropriately respond to HTTP resource requests
- Ability for server to parse, validate data, commit data, and return results related to an uploaded input file
- Ability for server to parse, validate data, commit data, and return results related to an uploaded input file
- Ability for server to prepare and return an output file containing records meeting specific criteria

System Testing will be used to evaluate the overall application, utilizing the functionality of multiple components, and will consist of manual and/or automated tests.

13. Deliverables

The deliverables as related to this test effort are as follows:

- Test cases executed, documented, and any defects logged as related to testing of the aforementioned features
- Detailed testing results document outlining tests execute, expected and actual results, and additional actions taken, if applicable
- Resolution of defects and re-testing related to essential functionality of the application

14. Test Team Organization

The test team will be organized as such that the tester of a specific functionality is generally the same person who developed it; this is expected to reduce time needed to identify appropriate tests, develop tests, and resolve defects.

Import and export file processing and associated web application functionality – William Ntumba

Web Application navigation/search/edit/add/delete alumni/employers/graduate schools functionality – Scott Shrout

User authentication related functionality – Anthony Freitas

15. Schedule

As is generally consistent with the course schedule, the schedule for testing is currently as follows:

Unit Testing (Automated and Manual): 12/4/2019 – 1/15/2020

API Testing: 12/4/2019 – 2/5/2020

System Testing: 1/16/2020 – 2/5/2020

The testing team is expected to communicate weekly regarding progress and any hinderances.

16. Defect Classification Criteria

Any features found to be defective during the testing effort must be classified by their criticality in having the application perform in a way consistent with the documented requirements. The defect classifications and associated criteria are as follows:

Low – does not impact the ability for the application to function in a way consistent with the documented requirements; may include cosmetic only issues

Medium – impacts the ability for the application to function in a way consistent with the document requirements, but resolution is not critical to further testing

High – impacts the ability for the application to function in a way consistent with the document requirements; other testing dependent upon defect being resolved; impacts a single application component

Critical – impacts the ability for the application to function in a way consistent with the document requirements; other testing dependent upon defect being resolved; impacts multiple application components

17. Release Criteria

The following criteria must be met prior to the introduction of a new release into the production environment:

- All components essential to the operation of the application consistent with the documented requirements must be tested with appropriate document of test criteria and results
- All defects of classification medium, high, or critical must be resolved or otherwise mitigated
- Approval from end user as related to test results and delivered application must be received