# Testing Planning Document

**Table of Contents**

# Introduction

This test plan describes the testing approach and overall framework that will drive the testing of the Credentialing System Version 1.0. The document introduces:

- Test Strategy - rules the test will be based on, including
    - The givens of the project (e.g. start & end dates, objectives, assumptions)
    - Description of the process to set up a valid test (e.g. entry & exit criteria, creation of test cases, specific tasks to perform, scheduling, etc.)
- Execution Strategy - describes how the test will be performed and is a process to identify and report defects, and to fix and implement fixes
- Test Management - process to handle the logistics of the test and all the events that come up during execution (e.g. communications, risk and mitigation, team roster)

# Overview and Tasks

The Credentialing System is a tool designed to provide the Davenport scheduling department with the ability to view, create, and update relevant credential information such as instructor resumes, certifications, and degree levels without having to use various spreadsheets and local folders. The application will be accessed as a web application and will require a DU login to be able to access the information.

All information is subject to Davenport’s defined security policy, where users can only view the information they are authorized to. An authorized user can edit certain fields, maintaining the security and confidentiality of employee information.

# Scope

**General:**

This document will test certain functionalities of the MVC application. Testers will utilize certain test cases to validate input and mark the test as either passed or failed. See sections below for features to be tested and features to not be tested.

**Tactics:**

The Excel document called Testing Data is a spreadsheet that auto-generates SQL insertion code.

This data should be entered in the database for testing before the testers begin. The testers will also utilize test cases that were built for them.

# Testing Strategy

## Objectives

The objective of the test is to verify that the functionality of Credentialing System Version 1.0 works according to the specifications.

The test will execute and verify the test scripts, identify, fix and retest all high and medium severity defects per the entrance criteria, as well as prioritize lower severity defects for future fixing.

The final product of the test is twofold:

- An updated version of the software that can be released to and tested by the client (even though not all functionalities/ use cases will be implemented)
- A set of stable test scripts that can be reused for future test executions

## Assumptions

**Key Assumptions**

- Testing data is required and must be available in the system/database prior to start of testing

**General**

- Alpha Testing would be carried out once the build is ready for testing
- Performance testing is not considered for this estimation
- All the defects would come along with a snapshot JPEG format
- Project team has the knowledge and experience necessary, or has received adequate training in the system, the project and the testing processes
- The system will be treated as a black box; if the information shows correctly online and in the reports, it will be assumed that the database is working properly

**Test Principles**

- Testing will be focused on meeting the requirements (use cases) and evaluating the quality of the code and project itself
- Testing processes will be well defined, yet flexible, with the ability to change as needed
- There will be entrance and exit criteria

**Levels of Testing**

1. **Exploratory Testing**

    **PURPOSE**: The purpose of this test is to make sure critical defects are removed before the next levels of testing can start.

    **SCOPE**:  First level navigation, error handling

    **TESTERS**: Development team

    **METHOD**: This exploratory testing is carried out in the application without any test scripts and documentation

    **TIMING**: At the beginning of each cycle & each commit to GitLab

2. **Functional Testing**

    **PURPOSE**: Functional testing will be performed to check the functions of application. The functional testing is carried out by feeding the input and validates the output from the application.

    **SCOPE**:  TBD

    **TESTERS**: Development Team, Testing Team

    **METHOD**: The test will be performed according to functional scripts.

    **TIMING**: After Exploratory testing is completed.

    **TEST DELIVERABLES**:

| Testing Number | Deliverable | Authors |
|---|---|---|
| V1.0T1 | Testing Plan | Stefana Rusu |
| V1.0T2 | Functional Test Cases | Devon Fairman, Stefana Rusu, Tyler Palmer |
| V1.0T3 | Logging Defects/Bugs in Error Log | Devon Fairman, Stefana Rusu, Tyler Palmer |
| V1.0T4 | Weekly Status Report | Devon Fairman |

3. **User Acceptance Testing**

    **PURPOSE**: This test focuses on validating the business logic. It allows the end users to complete one final review of the system prior to deployment.

    **TESTERS**: End Users, Clients

**METHOD**: Since the end users are the most indicated to provide input around system needs and how the system adapts to them, it may happen that the users do some validation not contained in the scripts. Test team write the test cases based on the inputs from End Users.

**TIMING**: After all other levels of testing (Exploratory and Functional) are done. Only after this test is completed the product can be released to production.

# Test Cases

### Functional Tests

1. **V1.0T2 - Course Add**

| **Field** | **Expected Input** | **Expected Output** | **Pass/Fail** |
|---|---|---|---|
| Course Name | ENGL110 | ENGL110 | P |
| Course Title | Advanced Composition | Advanced Composition | P |
| Course Description | TBD | TBD | P |
| Course Credit Hours | 3 | 3 | P |
| Course Contact Hours | 45 | 45 | P |
| Course Subject Text | ENGL | ENGL | P |
| Course Suffix | N/A | N/A | P |

2. **V1.0T2 - Building Update**

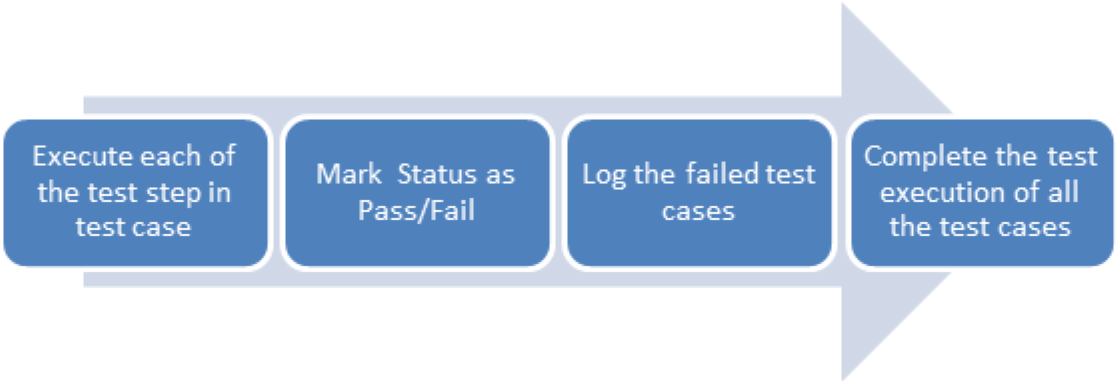
| | **Initial Data** | **Expected Input** | **Expected Output** | **Pass/Fail** |
|---|---|---|---|---|
| **Building Name** | Kalamazoo West | Kalamazoo Main | Kalamazoo Main | P |
| **Campus** | Kalamazoo | Kalamazoo | Kalamazoo | P |

3. **V1.0T2 - Room Search**

| | Expected Input | Expected Output | Pass/Fail |
|---|---|---|---|
| **Equipment** | Printer | N/A | P |
| **Software** | Visual Studio | N/A | P |
| **Campus Name** | Lettinga | Lettinga/ Lettinga | P |
| **Building Name** | N/A | Lettinga Academic/ Lettinga Maine | P |
| **Room Max Capacity** | N/A | 10-30 | P |
| **Room Description** | N/A | CSCI Lab/ COB Room | P |
| **Room ID** | N/A | 236/ 130 | P |

# Execution Strategy

### Test Execution Criteria


Execute each of the test step in test case
Mark Status as Pass/Fail
Log the failed test cases
Complete the test execution of all the test cases

(Adapted from Software Testing Help, 2014).

- Once all test cases are approved and the test environment is ready for testing, tester will start a exploratory test of the application to ensure the application is stable for testing.
- Each tester is assigned test cases.
- If any failures occur, they will be raised in the log.

# Hardware Requirements

Computers
Modems

# Test Schedule

See Project Plan.

**Major Deliverables:**

- Test Plan
- Test Cases
- Bug and Error Log
- Test Summary Reports

# Features to be Tested

- Campus
  - Create campus
  - Edit existing campus
  - View list of campuses & view details for specific campus
- Building
  - Create building
  - Edit existing building
  - View list of buildings & view details for specific building
- Room
  - Create room
  - Edit room
  - Search for room based on specific equipment and/or software
  - View list of rooms & view details for specific room
- Courses
  - Create course
  - View list of courses & view details for specific course

# Features not to be Tested

- Course
  - Edit Course is not yet functional
  - Search for Course based on specific equipment and/or software
- Main Menu

# Roles & Responsibilities

| Roles | Name | Contact Info |
|---|---|---|
| Project Manager | Stefana Rusu | srusu@email.davenport.edu |
| Test Lead | Devon Fairman | dfairman@email.davenport.edu |
| Development Lead | Stefana Rusu, Devon Fairman | |
| Technical Lead | Tyler Palmer | tpalmer12@email.davenport.edu |

# Tools & Environment

**Environment**

- The system's database server will be hosted by Davenport University.
- A windows environment with Google Chrome, Microsoft Edge, and Visual Studio 2017 should be available to the tester.

**Tools**

- The tester will use the available test cases and make sure to check all functionalities of the system.
- The bugs/ errors will be logged for feedback.

# Approvals

| | Name (In Capital Letters) | Signature | Date |
|---|---|---|---|
| 1. | STEFANA RUSU | | 12/12/18 |
| 2. | DEVON FAIRMAN | | 12/12/18 |
| 3. | TYLER PALMER | | 12/12/18 |
| 4. | BRIAN KOWALCZK | | 12/12/18 |

# References

University of Houston. (2001). Course Registration System. *University of Houston.* Retrieved from http://sce.uhcl.edu/helm/RUP_course_example/courseregistrationproject/artifacts/test/plans/test_plan_arch.htm

Software Testing Help. (2018, June 13). Sample Test Plan Document. Retrieved from https://www.softwaretestinghelp.com/test-plan-sample-softwaretesting-and-quality-assurance-templates/

Software Testing Help. (2014, February 1). Test Plan (a Real Sample). Retrieved from https://www.softwaretestinghelp.com