CS-UH 2012: Software Engineering
Spring 2020

CLASSROOK

Test plan

(IEEE 829-1998 Format)

0. Table Of Contents

1. Test Plan Identifier:

Name: classrook_test_master
Author and contact: engineering@classrook.com
Revision history: 1.0

2. Introduction

The purpose of this master level plan is to fully test and ensure the integrity and reliability of the software which is to be delivered to New York University Abu Dhabi (NYUAD), who is the client that ordered the software.

The software in question is Classrook, of which the full description and requirements can be reviewed in a previous document titled CLASSROOK_SRS. This piece of software consists of many different use cases and components, each of which has the possibility to inter-mingle with other use cases. This can create a lot of complexity within the functionality of managing a resource sharing platform. Therefore, it is key to develop a procedural and straightforwards test plan which can methodically analyze key important features that require the most attention. These important features will be the most useful for satisfying the academic resource sharing requirements set forth by the NYUAD stakeholders.

More information related to the use cases and the class diagrams for this project can be referenced in the following documents: Classrook_usecase and Classrook_classdiagram. While there are many minor use cases, for the purposes of the prototype delivered initially, the primary focus will be on the validity of three key use cases, that are central to the functioning of the program. The three use cases in question are:

1. View course information (Version 1.0)
2. Post course reviews (Version 1.0)

3. Upload course material (Version 1.0)
3. Download course material (Version 1.0)

3. Test Items and Scope

The items to be tested will be the interaction and responses between the end user and Classrook in attempting a particular sequence, or sequences of actions. In particular, the test cases will directly test the various method and pathways that the user can take in order to achieve the end goal of completing one of the above key use cases.

The test will primarily involve **Validation testing**. This type of testing ensures that all functional, behavioral and performance requirements of Classrook were satisfied as described within the reference document: Software Requirement Specifications (version 1.0) It can can be more detailed and involves black-box testing strategies which run against expected outputs from inputs such as in unit tests.

We will also **Acceptance testing**, which analyses a whether classrook is able overall satisfy the business requirements outlined proposed by our client. More information can be found in the reference document: Software Requirement Specifications. The details of the particular acceptance tests can be refered to within the later sections.

Items to be excluded from testing will be use cases that are outside of the scope of resources for testing, and for the purposes of this assignment, the components we were able to implement so far within the prototype. In this case, the scope of what we will be testing will be the **above four key use cases** and how they can be used in regular settings, on various hardware, and under different scenarios that may be different from the typical. Classrook should be resiliant to both technologically

inexperienced freshmen NYUAD students, and also tech-savvy CS-major students at NYUAD.

4. Features To Be Tested

Each use case can involve a number of system-level features that work in tandem in order to delvier the desired response. In this regard, we would need to test the overall feature and each individual sub-function. The objective is to satisfy these relevant features requested by the NYUAD administration in our SRS document.

1. A NYUAD student using to software in order to **view a course information**. This includes the following steps:

1. Logging into ClassRook
2. Visiting the course catalog
3. Selecting a course from the course list
4. Viewing the course page

A risk-level of this feature is set to Medium (M). This scale level out of H/M/L was chosen because of the relative damage that can happen if this feature fails. If a student could not view a course information it would result in limited damages.

2. A NYUAD student using to software in order to **post a course review**

1. Ensuring that the user is logged in.
2. Visiting the course catalog
3. Selecting a course from the course list
4. Writing a review
5. Recieving +1 app credit for future use

A risk-level of this feature is set to Medium (M). This scale level out of H/M/L was chosen because of the relative damage that can happen if this feature fails. If a student could not post a course review, the student would be able to contact the software support team (who will be providing maintenance as allocated within the

Economic_Feasibility_Report). This would be a relatively quick fix and would result in the review eventually being posted.

3. A NYUAD student using to software in order to **upload course material**
    1. Ensuring that the user is logged in.
    2. Visiting the course catalog
    3. Selecting a course from the course list
    4. Uploading course material
    5. Recieving +1 app credit for future use

The risk level of this feature is set to High (H). This is because course materials are considered a key element of Classrook's requirement. If a user is not able to upload course materials, or even worse, upload materials with errors in the file, then it could cause long term damages as the file remains on the server for a while.

4. A NYUAD student using to software in order to **download course material**
    1. Ensuring that the user is logged in.
    2. Visiting the course catalog
    3. Selecting a course from the course list
    3. Uploading course material

The risk level of this feature is set to High (H). This is because course materials are considered a key element of Classrook's requirement. Furthurmore, downloads are a key source of revenue for the platform therefore it is important that this aspect of the software works well.

5. Features Not To Be Tested

There are a number of features that will not be tested, especially features that will not be incorporated into the initial prototype version of classrook. For the purposes of this assignment we will only be testing the above four key features, all other features of classrook will not be tested including but not **limited** to:

In this scenario we will not be testing the view all course reviews feature. This is because this element is considered a extension of the Classrook core features. It is primarily used as part of the feature which can recommend courses to users. We will be including this additional course recommender through reviews in a future release.

6. Approach

To effectively test Classrook we will follow a series of important steps that will enable us to systematically conduct the test.

Special tools:
There will not be special tools required for testing. In order to interact with ClassRook the user will use the traditional IO devices such as a Keyboard and Mouse. These are familiar to the students at NYUAD.

Metrics to be collected:
The metrics to be collected will be whether ClassRook successful or unsuccessfully passes/fails the test at each level. We will also record the overall test summary at the end of each test case.

Configuration:
There will be a number of hardware configurations i.e laptop, desktop that will be tested. We will also be testing software configurations i.e OSX/Windows7/Windows10. We will test all relevant combinations of HW/SW packages in order to ensure a comprehensive coverage.

Special requirements:
The tests should be conducted by both ClassRook developers and also the NYUAD students in order to demonstrate that the software is easy to use for all kinds of end users including those new to Classrook.

The steps for the full test plan:

1. **Create the four relevant tests cases**

-> These tests would be based upon the selected four use cases that are key to Classrook and have been implemented within the initial version one prototype. Information can be found in the following reference documents. View course information (Version 1.0) Post course reviews (Version 1.0), Upload course material (Version 1.0), Download course material (Version 1.0)

-> The requirements would be analyzed from the SRS documents in order to ensure that the tests directly follow the specifications for the use cases given from the customer who ordered the software.

-> From these requirements we would create scenarios of tests where a certain pathway of action would be performed. At each step we would write unit tests that directly assess whether Classrook has produced the desired output.

-> We would determine the exact Item pass/fail citeria of the individual tests.

2. **Set up the testing environment**

-> We would set up the required software testing environment consisting of combinations of HW and SW setups are described in the previous section. The setup also includes the end-user of ClassRook which would serve as a final verification that the test is successful.

3. **Execute each test**

-> Based on the inputs and, we would run each test. After the tests are completed, the outputs are recorded.

-> At each level, we would execute sub-level tasks on which the primary task depends on.

4. **Compare outputs to expected outputs.**

-> For each verification test, the system should automatically analyze the output of ClassRook to the expected output to determine whether it has passed or failed.

-> After all the executions are performed, we would analyze the path that the software has taken, identify any error reports, and determine whether the output is correct.

->Finally we would record any invalid outcomes of the test within our Test Results document that would be later used to fix and update Classrook such that the test cases pass.

The constraints of running this approach would be resource avaliability: We would need to ensure that the relevant hardware and software tools are avaliable on site for the developers and students to quickly and efficiently execute the plan. The deadline for running the full suite of tests should not exceed more than 1 day(s).

7. Item Pass/Fail Criteria.

What are the completion criteria for Classrook's testing plan (Version 1.0)? The completion criteria occurs when each and every test and sub-test has successfully completed.

The code coverage should consist of at least 50% of the code, and the success rate should be 100%. Every operation of Classrook is important to the satisfaction of students. There is very little room for error when it comes to an application that utilizes credit transactions for academic-sensitive documents.

From the master test plan level, all the lower level test cases should be completed. Since for the purpose of this assignment we are only testing four key use cases, this document serves as the master test plan.

8. Suspension Criteria and Resumption Requirements:

The suspension criterion of this test plan could include both anticipated and unanticipated events. When these issues have been resolved, and certain resumption requirements met, then the testing can commence. If the test has begun, however, the number of defects has reached a number greater than 5, then at this point there is no purpose of continuing the test. Many errors could simply be cumulative errors from a previous defect. It would be better to not waste more resources and send the software back to the development team (along with a test report) in order to the software to be improved.

Testing can resume if ClassRook is believed by the development team to be ready to deliver. As in, the software has a high likelihood of successfully delivering the requirements by NYUAD. Testing can also resume only after all the relevant requirements laid out in the Approach section are satisfied. Such as the provision of the required Hardware (HW) demands inclding a laptop and desktop device, or the provision of required Software (SW) such as Windows 7 and a OSX

9. Test Deliverables

This plan will also lay out a number of test deliverables that will be delivered at the end of the testing procedure. This can be used both as a certification that the software has met the requirements at this current version number, and can also be used by the developers to assist furthur software devleopt and TDD approaches.

The deliverables:

1. This master testing plan document
2. Each of the test cases, including the specifications
3. The input and anticipated output data from running each use case test.
4. The recorded outputs and success/fail nature of each test log.
5. A summary detailing the process of the test, Indicating any anomalies experienced during the testing procedure.
6. A Test_Results document containing these errors and the subsequent fixes made.

10. Test tasks.

The tasks needed to accomplish each testing deliverable encompasses:

1. Developing the test plan document by careful analysis of use cases and communication with the client.

2. Creating invidual unit tests to capture the essence of each key functionality.

3. Creating the data and expected outputs by analyzing the pathways and intermediary steps to reach this point.

4. Setting up the environment and running the test.

5. Recording any incidents

6. Creating a summary reports that takes into account both internal and external groups that could potentially be driving the testing process.

11. Environmental needs

We would simply need the required Hardware(HW) and Software (HW) combinations that would be seen deployed in the end product. These resources should be easily obtainable from either the student's current equipment (macbook laptops), or purchased for testing purposes from the market.

For the purposes of this assignment, Classrook should be restricted to only the relevant aspects that have been implemented for the purpose of this first version prototype. For example, we would not need to include the unfinished duplicate material detection feature

The security of the site should be maintained in order to not leak any Intellectual Property or information i.e the source code to any unauthorized third parties.

12. Responsibilities:

The responsibility of executing this master test plan will rest primarily on the test master who will oversee the entire process. This will be assigned on the test day.

The reponsibility of running each automated and non-automated test will depend on both the developers and the independent testers (e.g a selection of NYUAD students). Each test will be executed and the output should be logged.

The responsibility for providing required training for the independent testers shall be provided by the developers of ClassRook in a manner that mirrors a real life deployment. The test master will be responsible for logging any anomalous incident(s) and creating the final testing summary documents that are to be delivered to stakeholders.

13. Staffing and training needs.

The tests will need to be performed by both the developers, and the end users (nyuad students). They should be avaliable during the testing process in order to ensure a quick and effective testing procedure.

These end-user individuals should be briefly trained by the developers on the general workings of Classrook. The developers and end-users should alrealy be familiar with the particular functionality that they expect to work within the software.

14. Schedule.

Test case ID: 1
Test Priority: Medium
Tested module: Courses information module
Test Title: **Test_view_course_information**
Test Execution date: **07/05/20**
Tester: The developers

Test case ID: 2
Test Priority: **Medium**
Tested module: **Course review module**
Test Title: **Test_post_course_review**
Test Execution date: **07/05/20**
Tester: NYUAD student

Test case ID: 3
Test Priority: **High**
Tested module:**Course material module**
Test Title: **Test_upload_course_material**
Test Execution date: **07/05/20**
Tester: The developers

Test case ID: 4
Test Priority: High
Tested module: **Course materials**
Test Title: **Test_download_course_material**
Test Execution date: **07/05/20**
Tester: The developers

15. Risks and contingencies

A smooth testing process would be ideal, however, there are possible risks in which we should have contingency plans.

Lack of personnel resources when testing is to begin: If this is the case, then the test should be paused until the personnel has arrived. If key individuals such as the test master is not avaliable then the testing should be cancelled until a desired time can be agreed upon.

Lack of availability of required hardware, software, data or tools. If this is the case, then the procurement of these missing tools should be made a top priority and relevant funding allocated for obtaining these required devices.

Changes to the original requirements or design: Testing should continue as is, and if it is later determined, such as from the testing summary report that the original requirements have deviated too much, then another test can be ordered.

16. Approvals

The developers will be responsible for first approving the results of the test outcomes. It is the first check to ensure that the software performs as required.

NYUAD will give the ultimate approval of the testing results. (After reading the testing summary prepared by the test master that is in a language comprehendable by the average person).

IEEE test plan template obtained from:
http://www.ecs.csun.edu/~rlingard/comp480/TestPlanTemplate.pdf

Test cass templates obtained from:
NYUCLASSES