

**American International University-Bangladesh (AIUB)**

**Course: Software Quality and Testing**

**Group No: 1**

**Section: B**

**Project Title:** Developing a Test Plan for Student Registration System for a University.

**Group Members:**

|  |  |
| --- | --- |
| **Name** | **ID** |
| **Md. Abdullah Al Noman Mazumder** | **16-32079-2** |
| **Jishnu Saha** | **16-32020-2** |
| **Fahim, Hasan Shahriar** | **16-32025-2** |
| **Jaman, Sarder Sakib Ibnay** | **16-31988-2** |

CONTENT:

1) Test Plan Identifier

2) References

3) Introduction

4) Test Items

5) Software Risk Issues

6) Features to be Tested

7) Features not to be Tested

8) Approach

9) Item Pass/Fail Criteria

10) Test Deliverables

11) Remaining Test Tasks

12) Environmental Needs

13) Staffing and Training Needs

14) Responsibilities

15) Schedule

16) Planning Risks and Contingencies

17) Approvals

1. **Test Plan Identifier:**

Atest plan is a document describing software testing scope and activities. The project is to implement a robust web-based system for online course registration. It will allow student from all over the world to access the system and perform registration. This test plan documents the testing activities for the QARS development activities. The programmers will complete the application unit testing. There will be two testers and one test manager. In order to keep track of the most current version of our test plan, we should assign it an identifying number. A test plan identifier is a unique company generated number used to identify a version of a test plan, it’s level and the version of software that it pertains to keep in mind that test plans are like other software documentation – they are dynamic in nature and therefore must be kept up-to-date. If we don’t give an identifier number, then when anybody will see this they may thought that the plan was created but never changed and probably never used and also sometimes they may think that the plan was created only to satisfy International Standards Organization (ISO) or Capability Maturity Model (CMM) guidelines.

1. **References:**

Books notes:

* EEE recommended practice for software requirements specifications - IEEE Std 830-1998

* Software requirement specification ver. 1.1 august 29 2003, Michael J. Reaver, of Jacksonville State University

* Project management for information system ± James Cadle and Donald Yeasts

* Metrics and Models in Software Quality Engineering, 2nd Edition, Stephen H. Kan

* CSC 504 Dr. Rachides Class Notes.

* Indian Institute of Information Technology and Management ± Kerala, Techno
* Park Software Architecture Document, WyIT431, V1.0, 1999, Wylie College IT.
* Software Architecture Document, WyIT431, V1.0, 1999, Wylie College IT.

Websites:

* <http://www.wikipedia.org/>
* <http://www.adu.ac.ae/en>
* https://ndeva.auckland.ac.nz/ndeva/guest/guest\_frameset.asp (the university of Auckland)
* http://www.ox.ac.uk/admissions/ ( university of Oxford)

* http://www.projectperfect.com.au/info\_risk\_mgmt.php

* <http://isb.wa.gov/tools/pmframework/examples/mlsexample.aspx#assumptions>
* <https://www.bartleby.com/essay/Software-Requirement-Specification-for-Course-Registration-System-P3R3XYEK6ZZA>
* <https://brainmass.com/computer-science/processor-architectures/system-design-document-university-registration-system-256512>

1. **Introduction:**

The project is to implement a robust web-based system for online course registration where student from all over the world can access the system and perform registration. Test plan is the project plan for the testing work to be done. It is not a test design specification, a collection of test cases or a set of test producers. By writing a test plan, it guides our thinking. Writing a test plan forces us to confront challenges that await us and focus our thinking on important topic. Failure to allow enough time for system test. Since the delay comes at the end of the schedule no one is aware of schedule trouble until almost the delivery date. By using a template for writing test plans helps us remember the important challenges. The test plan helps us to manage changes. During early phase of the project, as we gather more information, we revise our plans. As the project evolves and situations change, we adapt our plans. By adapting the plan at major milestone helps us to keep testing aligned with project needs. As we run the tests, we make final adjustments to our plans based on the results. With

1. **Test Items:**

Data and Database Integrity Testing:

Verify access to Course Catalogue Database.

Verify simultaneous record read accesses.

Verify lockout during Course Catalogue updates.

Verify correct retrieval of update of database data.

Function Testing:

* The system shall interface with the Course Catalogue Database System.
* The system shall support the data format.
* The server component of the system shall operate from any Server and shall run under all Operating System.
* The client component of the system shall operate on any personal computer.
* The system shall integrate with legacy system (course catalogue database) which operates on the Main Frame.

User Interface Testing:

* Verify ease of navigation through a sample set of screens.
* Verify sample screens conform to GUI standards.
* The System shall be easy-to-use.
* The desktop user-interface shall be Windows 95/98 compliant.

Load Testing:

* Verify system response when loaded with 10000 logged on students.
* Verify system response when 500 simultaneous student accesses to the registration.

Security and Access Control Testing:

* Verify Logon from a local PC.
* Verify Logon from a remote PC.
* Verify Login security through user name and password mechanisms.

1. **SOFTWARE RISK ISSUES:**

A risk actually, in one sense, the flip side of assumption. With an assumption, we expect something to happen. With a risk we ask what will we do if something does not happen, or how do we increase that probability that something will happen. Put in other words a risk is something that may happen and if it does, will have a positive or negative impact on the project. A risk has a probability of less then 100%, and above 0%. It must be a chance to happen or it is not a risk. We measure risks by looking at their probability and impact. The probability may be highly likely, modest likely and not likely. The impact may be large, moderate or small.

* Integration of code written by different developers working separately.
* To remove Bumping when it works or student try to use it.
* Control properly when any student tries to take a course and his/her data is lost before register that course.
* Control properly when heavy number of students can access the system simultaneously.
* Security, performance and reliability.

1. **FEATURES TO BE TESTED**

The Following features are going to be tested.

Student View:

## Valid student

## A student must be valid. He/she must be log in with proper authentication before he/she go for registration. Moreover, a student under probation is not allowed to registration. He/she must meet to his instructor for registration. So, we need to test is the system validate this criterion properly.

## Close a course

If the registration count of a course reaches the limited count it will be closed. It should not visible for registration.

## Closed course cannot be registered

If a course is closed, then that course cannot be registered. In case, if the page is loaded with register button visible but another student may register that course while loading the page. And that cause the course may reach the closed phase so we need to test if the course has not already closed when registered button is clicked.

## Show available course for registration

## A course will be shown if and only if the prerequisite and the is matched or it is not closed. So we need to test this prerequisite criterion is matched as well as it is not closed already for the courses which is shown in the registration section.

## Register a course

We have to test if the courses are shown in the registration section students are being able to register that course without any problem.

## Time clash

if a student registers a course it should not be clash with another course. So that must be tested if the system is preforming proper validation for time clash among courses taken before.

## Only one section from a course can be registered

We must test if the system is switching the registration to second course when a student selects another section of a course which he/she already registered for a section before.

In that case, test case **#2** and **#6** must be performed.

## Drop course

## We must test if student can drop a course which they have registered until they confirm the registration.

## Registration must be in credit limit range

Student must register total credit which is between the minimum and maximum limit. So we have to check when a student clicked confirm button the system is validating the limits.

# Instructor View:

1. View registration count in a specific course

We have to test if an instructor is availing to see the current status of a course. For example, registration limit, current registration etc.

## 2. Registration in complicated case

We must test if an instructor can register a student for courses if any complexity occurs.

# Department Head View:

## Set limit

Department head should be avail to change the limit of a course. It must be checked.

1. Cancel courses, open reserved course, add new course

We must test if Department head is availing to cancel courses, open reserved courses and add new courses.

1. Change timing

We must check if the timing of a course can be changed by head and the changes is also affect in the student view.

## Change prerequisite

We must check if the department head is availing to change the prerequisite of a course.

System view:

1. Server load

Server load must be tested. As a huge number of student will hit on the server at a same time the server may be get down. So, we must test the capability that the server can maintain, if that capability is adequate for our system.

1. Data base integrity

We must test database integrity as the data is associated to courses will be fetched from database.

1. **FEATURES NOT TO BE TESTED**
2. Network issue

We don’t need to test network issue as it is not project feature.

1. Electricity issue

We don’t need to test electricity issue as it is not project feature.

1. The features, which has been tasted before successfully.
2. **Approach:**

|  |  |  |  |
| --- | --- | --- | --- |
| Testing Level | Testing Technique | Performed By | Cause Behind Testing |
| Unit Testing | White-box Testing | Programmer | All logical decision and their execution path should be verified. It will remove possible faults and error in the raw code. |
| Integration Testing | Grey box testing | Programmer and Tester | This testing will find out the defects if any between different modules while integrating. It is useful because it takes the black-box technique and also combines the technique of white-box for addressing the defects found. |
| System Testing | Black-box Testing | Tester | This testing includes spectrum of testing such as functionality and performance. After integrating all the modules system testing is conducted to evaluate the system’s correctness associated with its specified requirements in the SRS documents. |
| Acceptance Testing | Black-box Testing | Stakeholders | As this testing is done by stakeholders, ultimately this will ensure if the system include all the feature wanted by users is implemented as well as it will ensure users satisfaction. |

1. **Item Pass/Fail criteria**
2. Unit test level

All test cases must be passed completely. All the problematic logical issue should be passed in the test.

1. Integration Testing level

All the feature should work properly, so the test should produce correct result according to the test case as defined.

1. System Testing and Acceptance Testing level

Code coverage tool indicate all code covered.

1. **Test Deliverables:**
2. Test Plan

This document deals with what needs to be done in User acceptance testing (UAT).

1. Test Designs

The UAT Acceptance Criteria.

1. Test Cases

The values input and results expected from tests.

1. Test Item Transmittal Reports

Developers handover report.

1. Test Logs

The results of running the tests.

1. Incident Reports

Observations of unexpected results.

1. Incident Report Logs

Summary of Incident Reports.

1. Test Summary Report

Summary of testing.

1. The test data

Test data is the data that is used by the testers to run the test cases.

1. Test strategy

It is a document which captures the approach on how we go about testing the product and achieve the goals. It is normally derived from the Business Requirement Specification (BRS).

1. **Remaining test tasks:**

These areas needed to identify to avoid any confusion should defects be reported back on those future functions. This will also allow the users and testers to avoid incomplete functions and prevent waste of resources chasing non-defects.

1. Create Acceptance Test Plan.

2. Create System/Integration Test Plan.

3. Define Unit Testing rules and Procedures.

4. Define Turnover procedures for each level.

1. **Environmental needs**:

1.System and applications

2.Test data

3.Database server

4.Automated testing tools: JIRA, Selenium, Appium.

5.Bug tracking tools: Bugzilla, Trac, Redmine

6.Requirement tracking tools: Microsoft word and Microsoft Excel

7.Operating system: Windows 7 or above

8.Microsoft Office 2010 or above

9.Browser

10.Hardware includes Server Operating system.

1. **STAFFING AND TRAINING NEEDS:**

It is preferred that there will be at least one full time tester assigned to the project for the system/integration and acceptance testing phases of the project. This will require assignment of a person part time at the beginning of the project to participate in reviews etc... and approximately four months into the project they would be assigned full time. If a separate test person is not available, the project manager/test manager will perform this role. In order to provide complete and proper testing the following areas need to be addressed in terms of training:

1. The developers and tester(s) will need to be trained on the basic operations of the JIRA interface.

2.The administration staff will require training on the new screens and reports.

3.At least one developer and operations staff member needs to be trained on the installation and control of the project package.

4. Just-In-Time (JIT) approach can be followed for training needs and plan to schedule. Because there is often a temptation to attend training too far in advance of its usage when the test team has apparent slack. Doing this introduces the risk of the training being forgotten by the time it's needed.

5.The test team often requires the support and skills of other team members not directly part of the test team. Appropriate availability of System Administrators, Database Administrators, and Developers who are required to enable the test effort.