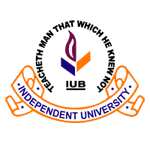
****

**Independent University, Bangladesh**

**Department of Computer Science & Engineering**

**SEN 653: Software Quality Assurance, Testing and Reliability**

**(Autumn 2020)**

**Term Paper**

**Test Plan:**

Learning Management System

**Prepared by**

Md. Tarek Hasan, 2031276

Israt Zahan Safia, 2022928

Farhina Akter, 2022927

Md. Ilias, 1910737

Nusrat Tahsin Kamaly, 1610152

Anupom Hossain Meraj, 1530851

**Date:** 20-01-2021

Table of Contents

[Introduction 3](#_Toc61193584)

[Scope 3](#_Toc61193585)

[Goals & Objectives 3](#_Toc61193586)

[Roles & Responsibilities 4](#_Toc61193587)

[Testing Methodology & Strategy 4](#_Toc61193588)

[Overview 4](#_Toc61193589)

[Level of Testing 5](#_Toc61193590)

[Usability Testing 5](#_Toc61193591)

[Unit Testing 6](#_Toc61193592)

[White Box Testing 6](#_Toc61193593)

[Black Box Testing 7](#_Toc61193594)

[System and Integration Testing 7](#_Toc61193595)

[Performance and Stress Testing 7](#_Toc61193596)

[Smoke Testing 7](#_Toc61193597)

[User Acceptance Testing 8](#_Toc61193598)

[Test completion criteria 8](#_Toc61193599)

[Batch & Beta Testing 9](#_Toc61193600)

[Final Release Test 9](#_Toc61193601)

[Test Completeness 9](#_Toc61193602)

[Test Approach (Testing techniques and models) 10](#_Toc61193603)

[Techniques 10](#_Toc61193604)

[Boundary Value Analysis (BVA) 10](#_Toc61193605)

[Equivalence Partitioning 10](#_Toc61193606)

[Decision Table Based Testing 11](#_Toc61193607)

[Error Guessing 12](#_Toc61193608)

[State Transition 12](#_Toc61193609)

[Integrating Manual And Automation Testing Techniques 13](#_Toc61193610)

[DevOps Testing 14](#_Toc61193611)

[Big Data Testing 14](#_Toc61193612)

[Risk-Based Testing Technique 15](#_Toc61193613)

[Models 16](#_Toc61193614)

[Team organogram 17](#_Toc61193615)

[Hardware & Environment Requirements 18](#_Toc61193616)

[Recommended System Requirements 18](#_Toc61193617)

[Test Environment 18](#_Toc61193618)

[Types of Testing Environments 18](#_Toc61193619)

[Disk Space Requirements 18](#_Toc61193620)

[Milestones / Deliverables 19](#_Toc61193621)

[Assumption & Risk 20](#_Toc61193622)

[Tools 21](#_Toc61193623)

[Approval 22](#_Toc61193624)

# Introduction

An envisioned learning management system for higher education will include courses from different universities around the country, training courses, workshops, seminars by well-known persons with a very reasonable cost so that students can afford from anywhere anytime across the country according to their desire and demands will help to accomplish the initiative. Education will be barrier-free, there will be no restriction on gaining knowledge thus the youth of Bangladesh will be equipped with the knowledge to face the challenges of the 21st Century. It will also promote economic growth by contributing to public-private partnerships, the rapid growth of IT industries, research, and development. More youth will be involved in research and development. The purpose of this project is to test each and every features of lbs in order to guarantee the customer satisfaction.

## Scope

The key features of our LMS are Registration process, Login, Course Registration, Enrolment, User profile. These requirements need to be met in order to guarantee customer satisfaction. So, we will be focusing on these requirements in our quality assurance process. Basically, All the functionalities will be checked to make sure that customers can not find any bugs in our system.

## Goals & Objectives

* Finding defects which may get created by the programmer while developing the software.
* Gaining confidence in and providing information about the level of quality.
* To prevent defects.
* To make sure that the end result meets the business and user requirements.
* To ensure that it satisfies the BRS that is Business Requirement Specification and SRS that is System Requirement Specifications.
* To gain the confidence of the customers by providing them a quality product.

## Roles & Responsibilities

|  |  |
| --- | --- |
| **Roles** | **Responsibility** |
| Author | The role of Author is appointed to the programmer or designer who is responsible for producing the program or document that is being inspected. |
| Moderator | The moderator is the leader of the inspection and s/he is responsible for planning the inspection as well as coordinating it. |
| Reader | The role of a reader is a simple one, but not insignificant. They present the code or document at an inspection meeting, where they read the document one by one. |
| Recorder | The Recorder or Scribe is the participant who is responsible for documenting the defects found during the inspection process. |
| Inspector | During the process of software inspection, inspectors are allotted the task of examining the work product to identify possible defects. |

# Testing Methodology & Strategy

## Overview

Learner wants a perfect Learning management system (LMS), which assed the full cycle of manual testing. The Test Plan has been created to facilitate communication within the team members. This document describes approaches and methodologies that will apply to the unit, integration and system testing of the Learning management system. It includes the objectives, test responsibilities, entry and exit criteria, scope, schedule major milestones, entry and exit criteria and approach. This document has clearly identified what the test deliverables will be, and what is deemed in and out of scope.

## Level of Testing

There are following four level of testing will be conduct in the system:

* Unit Testing,
* Integration Testing,
* System Testing,
* Acceptance Testing

## Usability Testing

Usability testing represents all activities that focus on observing users testing a product or service in certain working conditions. The goal for observer is to watch, listen and take notes while the participants are trying to complete tasks.

The ultimate purpose is to identify usability problems, collect qualitative and quantitative data and to determine participants’ satisfaction and dissatisfaction with the product or service. Generally, the usability testing takes two forms.

**Formative testing:** While the product is still in the development process, with a goal to diagnose the problems and try to fix them before representing it to the consumers.

**Summative testing:** commonly when the product is finished, with a goal to validate if the product meets the consumer’s requirements.

There is some plan for usability testing following:

• Title page

• Table of Contents

• List of Illustration, Abbreviation & Figures

• Executive Summary

• Problem Statement and Test Objectives

• Methodology

• User Profiles

• Participant Incentive

• Screeners

• Scenarios & Tasks

• Evaluation Methods

• Test environment and Equipment

• Presentation of Findings

• Pre-test Questionnaire, Post-task & Post-test Questionnaire

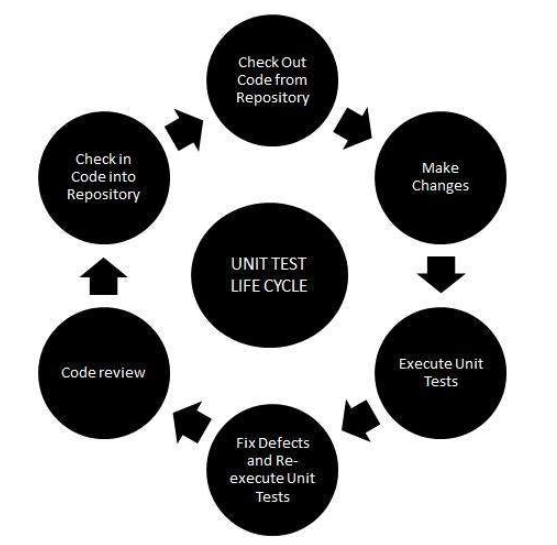
• Recommendations

• Appendices

## Unit Testing

In this type of software testing, individual units or components of a software are tested. Unit Testing is done during the “**Learning Management system**” development (coding phase) of an application by the developer. Unit Tests isolate a section of code and verify its correctness. A unit may be an individual function, method, procedure, module, or object.

**Life cycle of unit testing:**



## White Box Testing

To focuses on the “**Learning Management System**” internal structure, design and coding of “**Learning Management System**” are tested to verify flow of input-output and to improve design, usability and security. In white box testing, code is visible to testers so it is also called Clear box testing, Open box testing, Transparent box testing, Code-based testing and Glass box testing.

## Black Box Testing

To focuses on the functional requirements of the “**Learning Management System**”. It enables one to derive sets of input conditions that will fully exercise all functional requirements for a program.

## System and Integration Testing

**System Testing:** For a complete, integrated system to evaluate the system's compliance with its specified requirements of “**Learning Management System**”

**Integration:** To constructing the program structure while conducting test to uncover errors associated with interacting. In Report, integration testing includes the testing Report from “**Learning Management System**” respective location(s).

## Performance and Stress Testing

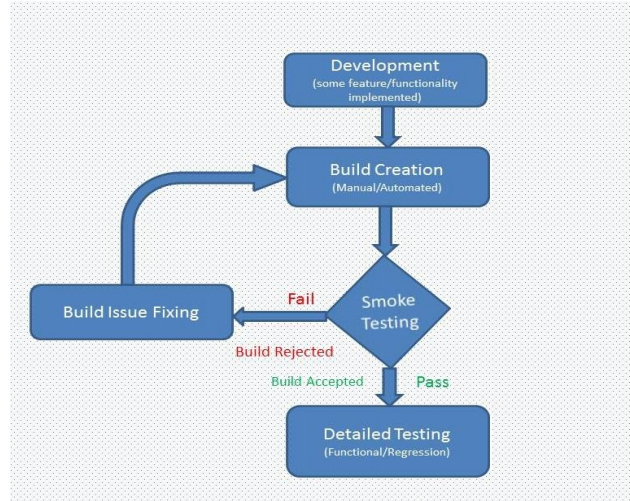
Performance Testing:

* Check the optimal time the page is loaded of **“Learning Management System”**
* Check the operation of the system under load **“Learning Management System”**

## Smoke Testing

The purpose of smoke testing of **“Learning Management System”** to determine whether the new software build is stable or not.

**Deployment Life Cycle for smoke testing:**



**There are three Technique of smoke testing:**

* Manual Approach creating test case for “**Learning Management System**” and run manually.
* Automation Approach run automation tools for “**Learning Management System**”
* Hybrid Approach combination of manual and automation approaches “**Learning Management System**”.

## User Acceptance Testing

The purpose behind user acceptance testing is to conform that system is developed according to the specified user requirements and is ready for operational use. Acceptance testing is carried out at two levels - Alpha and Beta Testing. User acceptance testing (UAT) will be done at the Client.

## Test completion criteria

**Test completion criteria**

* Any blocker issue for a feature will be taken care of immediately before testing the next one.
* The testing of the modules will be completed when the defects are fixed and re-tested.

**Unresolved Issues and Risks**

* These are risks to the schedule, scope, or quality of the test effort.
* The more defects or issues found the more time will be needed for fixing and retesting. This may interrupt the testing schedule as well as delivery within planned timeline of the delivery.
* The feedback (issue fixing/not fixing) from developers on Test defects, issues, remarks need to be well communicated. If not, Project delivery timeline can be hampered.
* If planned/allocated resources are not available for the test, then the testing date and product release date can be hampered.

## Batch & Beta Testing

Before “**Learning Management System**” is finally released to a wide audience. The objective is to uncover as many bugs or usability issues as possible in this controlled setting. Taking feedback from different stack holder for improving the system.

## Final Release Test

Before finally release the **“Learning Management System”** testing full process of the system.

## Test Completeness

When all the testers in your team are exhausted and when all the planned tests have been executed. In quality assurance testing technique in which all scenarios or data is tested for testing. In other word when you get positive result from all the test cases, you can consider completeness of your test

# Test Approach (Testing techniques and models)

## Techniques

### Boundary Value Analysis (BVA)

In boundary value analysis we will test at the boundaries between partitions. It includes maximum, minimum, inside or outside boundaries, typical values and error values.

It is generally seen that a large number of errors occur at the boundaries of the defined input values rather than the centre. It is also known as BVA and gives a selection of test cases which exercise bounding values.

The black box testing technique complements equivalence partitioning. This software testing technique is based on the principle that, if a system works well for these particular values then it will work perfectly well for all values which come between the two boundary values.

**Guidelines for Boundary Value analysis**

* If an input condition is restricted between values x and y, then the test cases should be designed with values x and y as well as values which are above and below x and y.
* If an input condition is a large number of values, the test case should be developed which needs to exercise the minimum and maximum numbers. Here, values above and below the minimum and maximum values are also tested.
* Apply guidelines 1 and 2 to output conditions. It gives an output which reflects the minimum and the maximum values expected. It also tests the below or above values.

**Example:**

In the registration form the user must put a valid email, without accurate email system will not accept request.

Password length must be greater than 6 characters furthermore with special characters.

Length & Email are the boundaries for the system to accept requests.

**Consider Minimum, Just above the minimum, A nominal value, Just below the maximum, Maximum**

Equivalence Partitioning or Equivalence Class Partitioning is a type of black box testing technique which can be applied to all levels of software testing like unit, integration, system, etc. In this technique, input data units are divided into equivalent partitions that can be used to derive test cases which reduces time required for testing because of the small number of test cases.

* It divides the input data of software into different equivalence data classes.
* You can apply this technique, where there is a range in the input field.

**Example 1: Equivalence and Boundary Value**

* Let's consider the behaviour of Registering for courses
* Students can register for 1 to 5 are considered valid. A success message is shown.
* While value 11 to 99 are considered invalid for registration and an error message will appear, **"Only 5 Course can be registered at a time"**

**Here is the test condition**

1. Any Number greater than 5 entered in the Course Registration (let say 11) is considered invalid.
2. Any Number less than 1 that is 0 or below, then it is considered invalid.
3. Numbers 1 to 5 are considered valid
4. Any 3 Digit Number say -100 is invalid.

We cannot test all the possible values because if done, the number of test cases will be more than 100. To address this problem, we use the equivalence partitioning hypothesis where we divide the possible values of tickets into groups or sets as shown below where the system behaviour can be considered the same.

### Decision Table Based Testing

A decision table is also known as the Cause-Effect table. This software testing technique we will use for functions which respond to a combination of inputs or events.

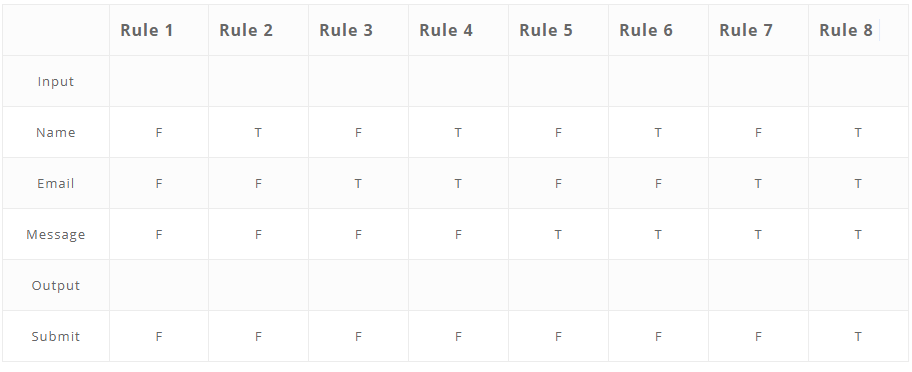
For example, a submit button should be enabled if the user has entered all required fields and disable if not.

For every function, you need to create a table and list down all types of combinations of inputs and its respective outputs. This helps to identify a condition that is overlooked by the tester.

**Following are steps to create a decision table:**

* Enlist the inputs in rows
* Enter all the rules in the column
* Fill the table with the different combination of inputs
* In the last row, note down the output against the input combination.

**Example**: A submit button in a contact form is enabled only when all the inputs are entered by the end user.



### Error Guessing

is a software testing technique based on guessing the error which can prevail in the code. The technique is heavily based on the experience where the test analysts use their experience to guess the problematic part of the testing application. Hence, the test analysts must be skilled and experienced for better error guessing moreover must have the product domain knowledge.

The technique counts a list of possible errors or error-prone situations. Then the tester writes a test case to expose those errors. To design test cases based on this software testing technique, the analyst can use the past experiences to identify the conditions.

**Guidelines for Error Guessing:**

* The test should use the previous experience of testing similar applications
* Understanding of the system under test
* Knowledge of typical implementation errors
* Remember previously troubled areas
* Evaluate Historical data & Test results
* Having Product Domain Knowledge

### State Transition

In State Transition technique changes in input conditions change the state of the Application Under Test (AUT). This testing technique allows the tester to test the behaviour of an AUT. The tester can perform this action by entering various input conditions in a sequence. In State transition technique, the testing team provides positive as well as negative input test values for evaluating the system behaviour.

**Guideline for State Transition:**

* State transition should be used when a testing team is testing the application for a limited set of input values.
* The technique should be used when the testing team wants to test a sequence of events which happen in the application under test.

**Example:**

In the following example, if the user enters a valid password in any of the first three attempts the user will be able to log in successfully. If the user enters the invalid password in the first or second try, the user will be prompted to re-enter the password. When the user enters password incorrectly 3rd time, the action has taken, and the account will be blocked.

### Integrating Manual And Automation Testing Techniques

By this we will get the benefit most by combining the efforts of both-manual as well as automation testing techniques. By integrating automation testing while conducting manual tests, your productivity and efficiency can increase by 10 folds! This is why it is important to recognize and segregate the test case that can be automated.

**Examples of such test cases may include:**

* Test cases that are most often needed
* Test cases which are the most time consuming
* Test cases that have a critical need for accuracy but have scope for human errors

For example, these test cases make up around 25% of your test plan. This will mean a 25% reduction of manual efforts and time consumption, Cloud-Based Testing Technique Cloud based testing includes the use of cloud-based tools for testing web, installed applications and web. These tools are used to match the environments and user traffic with the real-world.

A few tips to effectively implement the cloud-based testing techniques are:

* **Set Objectives-** This testing proves to be advantageous only if you have a clear objective set for your business needs. It requires cooperation between testers and developers for conducting all tests throughout the SDLC.
* **Creating Test Strategy-** Before transporting your project onto the cloud, determine the tests you need to perform, the time they will take and the risks involved in them. This will help you get an estimated idea of the testing budget.
* **Plan the Infrastructure-** Create test strategies that align with the infrastructure requirements needed for building the test environment.
* **Selecting a Provider-** To select the best provider, compare the quality, reliability and security being offered by them.
* **Determine Level of Access-** To conduct cloud-based testing many testers must have access to the cloud. Therefore, determine how many and who all can have access to it, so as to prevent the generation of additional costs from service overuse.

### DevOps Testing

‘*Development and Operations’* is a development methodology that integrates all development functions including development and operations in the same cycle. This methodology when used in software testing facilitates the testers to combine test cases, test automation and test designs so as to verify changes within the code while avoiding the product from crashing during the development phase.

A few DevOps testing technique tips include:

* Test execution must be lean
* Test cases required for particular builds need to be developed
* Standardize all the environment requirements for testing and automate the deployments
* Set exit criteria for every run to facilitate the go/no-go decision for production
* The testers must have the ability to use different types of automation techniques over different cross-platform environments.

### Big Data Testing

Big Data testing helps in ensuring that the quality of data is uncompromising. It’s going to be a widely used testing technique this year because of the decreasing costs of data storage. In this type of testing, testers verify whether the terabytes of data have been successfully processed using supportive components like commodity clusters or not.

A few examples of the test cases in big data testing are:

* Determine if the correct alter mechanisms, such as Mail on alert, are executed.
* Check whether errors and exceptions are properly displayed with appropriate error messages so that error/exception handling becomes easy.
* Implementing integration testing for complete workflow, from data ingestion till its storage or visualization.
* Performance testing for different parameters of processing random data and monitoring parameters like time taken in execution of particular metrics.

### Risk-Based Testing Technique

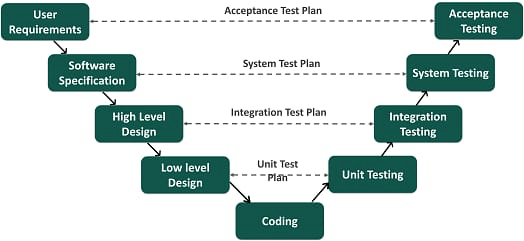
This type of testing is implemented with the aim of finding out the critical errors as soon as possible with least cost. Here, functionalities are prioritized and tested according to the level to which they are error prone.

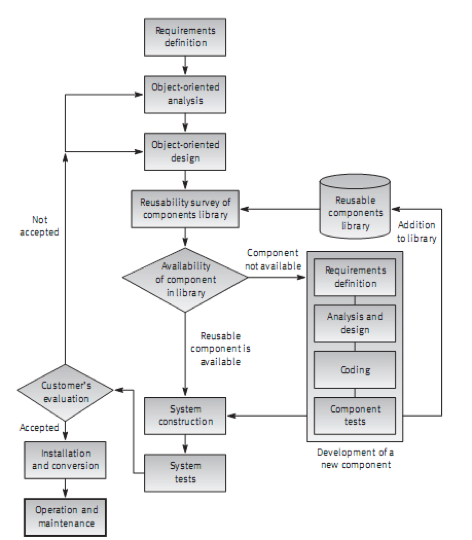
The steps to follow for effective risk-based testing include:

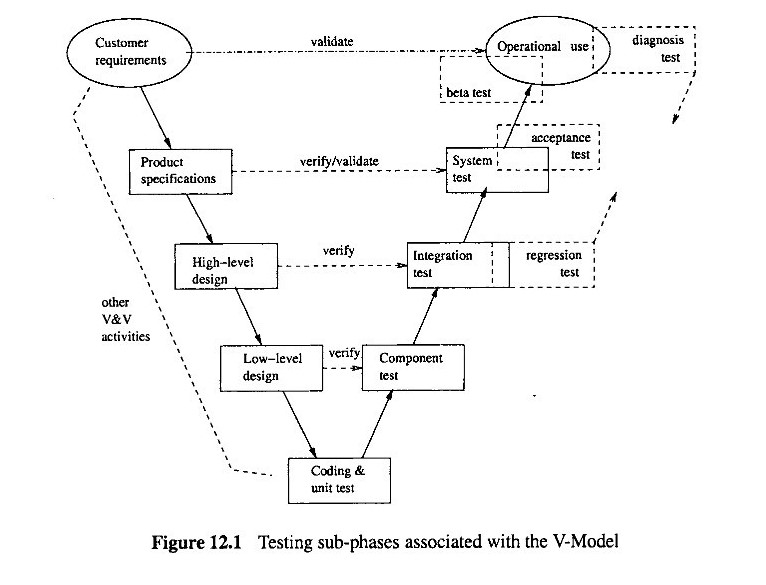
* Identify and prioritize risks
* Create test plans accordingly
* Eliminate or add risks according to results obtained

## Models

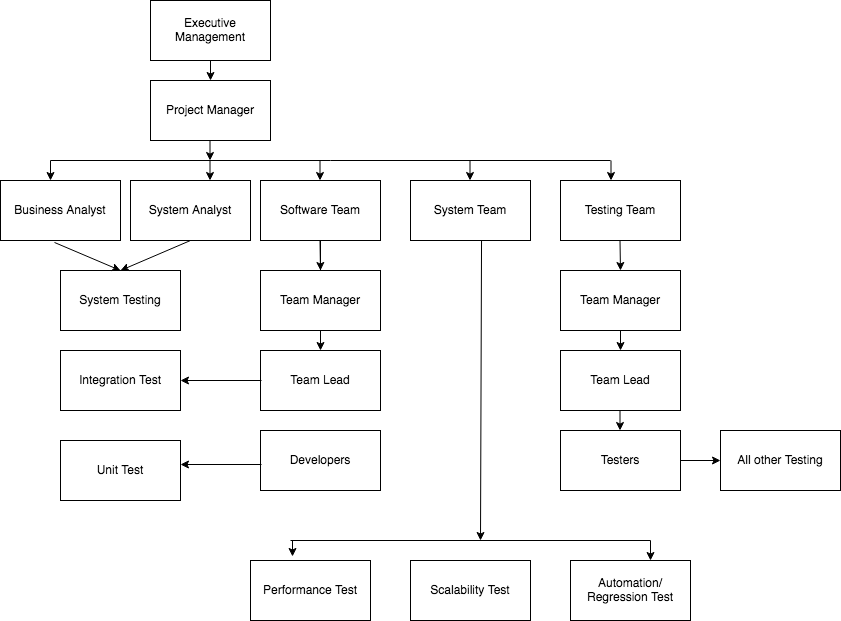
As per our system we are using the V-Model with OOP approach, In each step of V-model we will go through the OOP model.







# Team organogram



# Hardware & Environment Requirements

### Recommended System Requirements

* A 64-bit operating system like Windows or Linux.
* Latest Browsers.
* Intel Core Intel Core i7 ( 10th generation).
* 32 GB of RAM or more.
* 1 TB of free disk space for installation, plus extra space for temporary files during test runs (see below).
* 1280 × 1024 or higher display resolution.
* Mouse or other pointing device.
* Load balancer
* Test Server
* Few other automation tools...will write

### Test Environment

* System and applications.
* **Test** data.
* Test Database server.
* Configure the environment
* Front-end running **environment**.
* Client operating system.
* Browser.
* Hardware includes Server Operating systems.
* Network.
* Documentation required like reference documents/configuration guides/installation guides/ user manuals

### Types of Testing Environments

* Integration Testing Environment.
* Performance Testing Environment.
* Security Testing Environment.
* Chaos Testing Environment.

### Disk Space Requirements

Test Complete Platform creates and uses temporary files during test runs. The free disk space required by Test Complete Platform depends on your project size. You need at least 500 MB of free space on the system disk to run small projects, and 1–2 GB of free disk space to run large projects.

**Software Environment**

Web Server - Apache

Database - PostgresSQL

OS - Linux

Browser - Chrome/FireFox

PHP: version 7

**Test Bed:**

A test environment that has been prepared with test data. The test data helps you verify test cases that require a certain data setup.

# Milestones / Deliverables

The focus of quality is on the deliverables of the project. Quality assurance monitors project deliverables to verify that the deliverables are of acceptable quality, are complete, and correct. The following table identifies the major deliverables and/or milestones of the project to be evaluated for satisfactory quality level. Quality control activities to monitor these objectives are detailed in the next section, Quality Assurance Activities.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Project Milestone/Activity** | **Deliverables / Evaluation Products** | | **Timeframe** | |
| OCS Design Review |  | Submission material |  | |
|  | Final OCS Review Letter, no outstanding issues | |  |  |
|  |  |  |  | |
| SAW Integration |  | SAW connection documentation |  | |
|  | Testing and project acceptance | |  |  |
|  |  |  |  | |
| Pre-Data Migration |  | Field mapping |  | |
| Preparation |  | Domain/subdomain design |  |  |
|  | Development of Migration Workbook | |  |  |
|  |  | Identify migration content |  |  |
|  | Preparation check-lists | |  |  |
|  |  |  |  | |
| Stakeholder Outreach |  | Process Walks |  | |
| Pre-Configuration |  | Regular email communications |  |  |
|  |  | Regular working sessions |  |  |
|  |  | Monthly OCC meetings |  |  |
|  |  |  |  | |
| Stakeholder Outreach |  | Weekly email communications |  | |
| Post-Configuration |  | Weekly working sessions |  |  |
|  |  | Monthly OCC meetings |  |  |
|  |  |  |  | |
| STAGE Domain |  | Finalize Migration Workbook |  | |
| Configuration |  | STAGE environment accessible |  |  |
|  |  |  |  | |
| ALTSA HOTS |  | Project plans |  | |
|  |  | STAGE configuration |  |  |
|  |  | Training |  |  |
|  |  | UAT |  |  |
|  |  | PROD configuration |  |  |
|  |  |  |  | |
| HRMS/AD Data Feeds |  | Stakeholder requirements |  | |
|  |  | Identify HRMS/AD fields |  |  |
|  | Extract/transform/load design and plan | |  |  |
|  | Testing and project acceptance | |  |  |
|  |  |  |  | |
| Customer Agency |  | Training completed |  | |
| Readiness |  | Migrated data validation |  |  |
|  | Manual data input/loading adequately | |  |  |
|  |  | complete |  |  |
|  |  |  |  | |
| Initial Data Migration |  | Data validation |  | |
|  |  |  |  | |
| Monthly Executive |  | Monthly Status Report |  | |
| Steering Committee |  | Quality Assurance Report |  |  |
| Meetings |  | Updated OCIO Dashboard |  |  |
|  |  |  |  | |
| Secondary Data |  | Data validation |  | |
|  |  |  |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| Migration |  |  |  |
|  |  |  |  |
| Administrator Training |  | Sessions scheduled to adequately address |  |
|  |  | skill levels, context, and volumes |  |
|  |  |  |  |
| User Acceptance |  | Enterprise sign-off |  |
| Testing |  | Customer agencies sign-off |  |
|  |  |  |  |
| Go-Live Readiness |  | Go-Live Readiness Criteria |  |
|  |  | OCIO Briefing |  |
|  | Black-out period | |  |
|  |  |  |  |
| Deployment |  | Final data migration validation |  |
|  |  | Maestro “deactivation” |  |
|  |  |  |  |
| Stabilization |  | Weekly status reports |  |
|  |  |  |  |
| Closure Reporting |  | Post Closure Report (QA) |  |
|  | Post Implementation Review (PM) | |  |
|  |  |  |  |

The following identifies quality assurance actions and activities to evaluate milestones and activities defined above

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **Project Processes** |  |  | **Quality Assurance Activity** | |  |  | Frequency/Interval | |  |
|  |  |  |  |  |  |  |  |  |  |  |
|  | Project | |  |  | Perform QA Readiness Assessment | |  |  | Once per project (fall 2019) | |
|  | Planning | |  |  | Develop project QA Plan | |  |  | Once per project (spring 2020) | |
|  |  |  |  | Review project planning documents | | |  |  | Continuous for project duration | |
|  |  |  |  |  | Attend project planning/coordination | |  |  | Continuous for project duration | |
|  |  |  |  |  | meetings | |  |  |  |  |
|  | Technical | |  |  | Attend design sessions | |  |  | Continuous for project duration | |
|  | Design | |  |  | Review documentation | |  |  | Continuous for project duration | |
|  |  |  |  | Interview core project team members | | |  |  | Intermittent for project duration | |
|  |  | |  |  |  | |  |  | | |
|  | Data Migration | |  |  | Attend design sessions | |  | Continuous until initial data migration | | |
|  |  |  |  |  | Review documentation | |  | (April 2020) | | |
|  |  |  |  | Interview core project team members | | |  |  |  |  |
|  |  | |  |  |  | |  |  | | |
|  | Systems | |  |  | Design review | |  | Per occurrence | | |
|  | Integration | |  |  | Review documentation | |  |  |  |  |
|  |  |  |  | Interview core project team members | | |  |  |  |  |
|  |  | |  |  |  | |  |  | | |
|  | Production | |  |  | Check-in with ESC members | |  | One per member, August 2020 | | |
|  | Implementaion | |  |  | Test as extended user | |  |  | August 2020 | |
|  |  |  |  |  |  |  |  |  |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| (STAGE) | Interview core team members post- | Monthly from April – August | |
|  | implementation |  | 2020 |
|  | Vendor interview |  | July 2020 |
|  |  |  | |
| Go-Live | Check-in with ESC members | One per member August 2020 | |
| Readines | Test as extended user |  | August 2020 |
|  | Interview core team members post- | Monthly from April – August | |
|  | implementation |  | 2020 |
|  |  |  | |
| Risk | Review with core project team and vendor | Monthly prior to Executive Leadership | |
| Management |  | Meeting | |
|  |  |  | |
| OCIO Gated | Review with core project team and vendor | Monthly prior to Executive Leadership | |
| Project |  | Meeting and per Gate | |
| Processes |  |  |  |
|  |  |  | |
| Change | Review with core project team and vendor | Monthly prior to Executive Leadership | |
| Management/ |  | Meeting | |
| Training |  |  |  |
|  |  |  | |
| Stabilization | Interview core project team members | Once per project | |
|  | Interview ESC members |  |  |
|  |  |  | |
| Close project | Deliver Closeout Report | Once per project | |
| with post |  |  |  |
| project review |  |  |  |
|  |  |  |  |

# Assumption & Risk

The modern LMS expresses a particular set of assumptions about the delivery of learning. These assumptions include:

1. Knowledge is best delivered in a discrete, time-bound, linear set of experiences known as “courses.”
2. Courses are designed by an individual instructor or a professional learning designer.
3. Learners are consumers of instructor defined content, assessment, and activities. They may share some work products, but only when and if allowed by the instructor.
4. Learners are responsible for the cognitive load of interacting with the course.
5. Only instructors design or select the learning experiences for a given course instance.

These assumptions create a limited option space for any designer. If we design a learning environment that doesn’t look like a course, it will be challenging to find a market.

Learning Management System (LMS) responds to risk by taking ownership of the systems security problem and through implementing controls to manage it. To govern effectively, management may wish to adopt a preexisting LMS model to implement best practices to manage its assets and to provide transparency over their decisions. Within the scope of governance is the practice of risk analysis and management’s response to justify the costs of safeguards.

There should be a close tie between the outcomes of analysis and the requirements of countermeasures to be considered, how well potential risks and failures have been analyzed and then addressed. There should be a security manager to assume the risks and failing aspects of this LMS project.

Risk management incorporates an understanding of the vulnerability of the project to the consequences of various threats and hazards. The goal is to decrease the exposure of the project to risks through management actions.

Security managers deal with risk in the following ways:

• **Risk Assumption**— A risk contingency plan can be developed for the project that defines the actions taken, the resource plans, and the factor that triggers an action should a given risk occur. This option accepts the potential risk and continues assuming the contingency plan lowers the risk to an acceptable level (low cost). Risk Avoidance: We can bring down the risk by removing the cause or reducing the consequences (some cost).

• **Risk Limitation**— To limit the risk by implementing specific changes planned activities in the project. This approach should be pursued when the risk cannot be dealt with any other way or will be too costly to the project.

• **Risk Transfer**— We can transfer the risk by using other techniques to offset the loss. One example would be to purchase insurance. The potential impact of a risk event can be transferred by insuring a product or department against the liability of damage.

• **Risk Probability**— The likelihood the risk event will occur.

• **Potential Consequences**— The severity of the consequences on cost or schedule allows the risk to occur.

Choose the risk elements that have the greatest possible impact on cost or schedule. The risk probability is a number between 0 and 1, and the consequences are expressed in dollars so the risk event impact is in dollars.

# Tools

It is not possible to sell an LMS to a learning provider that does not conform to these assumptions. So LMS developers will not develop an alternative. The value chain assumptions of the users constrain the tools themselves.

Three vital tools of support service for running a LMS:

**Administrative Tools**

These tools enable the admin to effectively use the LMS.

* tools which allow authentication of user
* specific user tools to assist in specific access as per the roles
* ability to add or remove students
* host and domain services, etc.

**Course Delivery Tools**

      These tools help the teacher/instructor to efficiently deliver the course.

* These tools allow automated testing management
* System services for importing and exporting tests followed by statistical analysis of test results
* Online marking tools, online grade book
* Monitoring the progress of user as per the course requirements, student tracking feature etc.

**Content Delivery Tools**

* Accessibility compliance for all types of learners
* Compliance to WAIWCAG 1.0 AAA guidelines
* Course templates, customized tools and feel, instructional design tools,
* Instructional standard compliance
* Content authoring and integration in text, audio, video, graphics, animation and multimedia platforms
* Provision for blogs and Wikipedia information, etc.

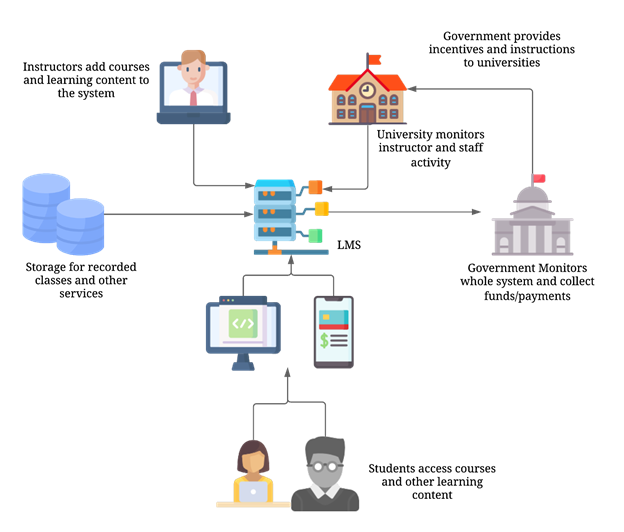
The selection of any LMS should be in conformity with all the above standards, requirements and components thereby fulfilling the requirement of providing ICT based learner support services to students. Teachers should be able to respond individually to their students. They may use it for either online or blended learning. It can be used to totally or partially replace face-to-face teaching.

Open knowledge platform content is made available to even non registered users. It allows institution to be able to use their campus administration to import student data. An efficient LMS help educators /instructors to create effective online learning communities and create online courses with opportunities for rich interaction.

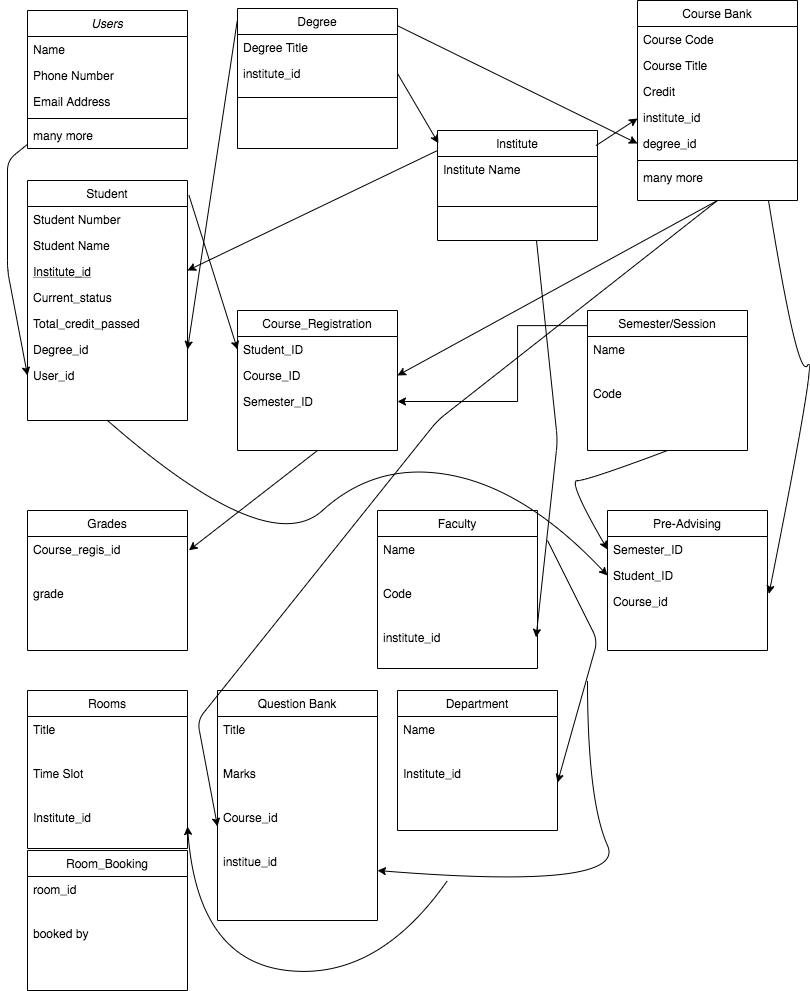
# Approval

|  |  |  |
| --- | --- | --- |
| Name | Position | Signature |
| Md Tarek Hasan |  |  |
| Israt Zahan Safia |  |  |
| Farhina Akter |  |  |
| Md. Ilias |  |  |
| Nusrat Tahsin Kamaly |  |  |
| Anupom Hossain Meraj |  |  |

**Rich Picture**



**ERD**

****