**Name : Sunil Shrestha**

Test Plan Document

# Introduction

1. **Purpose:**

This test plan describes the testing approach and overall framework that will drive the testing of the Student Management System. 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.
* Execution Strategy: describes how the test will be performed and 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 .

1. **Project Overviews:**

Student Management system is the application that record all the list of students . The application allows the Administrator to add, update, search and delete the record of student. This application help to mitigate the problem occur in traditional(manually) record system .

This application provides detail record of the student and the score they obtained so the teacher can focus on the student who obtain less score and provide more guidance . Student Management system help the teacher to analyze the data of the student and provide students proper guidance according to the score they have obatins.

# Reference

* Mockups
* Requirements Documents (Project plan , functional requirements)

# Scope

## In Scope :

## Different type of testing will be performed for each module . Type of testing involved for this project include :

1. **Functionality Testing** :

Functional Testing will be carried out where the functionalities are verified and validated according to the various requirements.

1. **UI Testing** :

-Check Screen Validations

-Verify All Navigations

-Check usability Conditions

-Verify Data Integrity

-Verify the object states

-Verify the date Field and Numeric Field Format

1. **Regression Testing:**

Regression Testing will be done to assure that changes in any module don’t have ripple effect on related modules which will be delivered or enhanced.

1. **Sanity Testing :**

Sanity testing is performed after receiving a software build, with minor changes in code, or functionality, to ascertain that the bugs have been fixed and no further issues are introduced due to these changes .

1. **API Testing :**

API enables communication and data exchange between two separate software systems. A software system implementing an API contains functions/subroutines which can be executed by another software system.

## Out Scope :

* 1. **Security Testing :**

Will not be considered for now but may have to do in the future

* 1. **Performance Testing:**
     1. Load Testing : Checks the application's ability to perform under anticipated user loads. The objective is to identify performance bottlenecks before the software application goes live.
     2. Stress Testing : Involves testing an application under extreme workloads to see how it handles high traffic or data processing. The objective is to identify the breaking point of an application.

# Feature to be tested

The feature that to be tested are given below :

1. To check the functionality like login of the administrator, view list page of students, view sorting list of students according to the name and score .
2. Functionality like searching the individual student details according to the name of student and score they have obtained .
3. To ensure that data exchange and data integration between the systems.
4. To ensure that student details export to CSV format.

# Testing Strategy / Approach

A Test Strategy is a plan for defining the testing approach, and it answers to questions like what you want to get done and how you are going to accomplish it.

## Test data preparation :

Test data is actually the input given to a software program. It represents data that affects or is affected by the execution of the specific module. Test data is used for positive testing, typically to verify that a given set of input to a given function produces an expected result and also for negative testing to test the ability of the program to handle unusual, extreme, exceptional, or unexpected input.

Test Data can be Generated -

* Manually
* Mass copy of data from production to testing environment
* Mass copy of test data from legacy client systems
* Automated Test Data Generation Tools

In functionality Testing functional test cases can have test data meeting following criteria -

**No data:** Check system response when no data is submitted

**Valid data:** Check system response when Valid test data is submitted

**Invalid data:** Check system response when InValid test data is submitted

**Illegal data format:** Check system response when test data is in an invalid format

**Boundary Condition Dataset:** Test data meeting boundary value conditions

**Equivalence Partition Data Set:** Test data qualifying your equivalence partitions.

**Decision Table Data Set:** Test data qualifying your decision table testing strategy

**State Transition Test Data Set:** Test data meeting your state transition testing strategy

**Use Case Test Data:** Test Data in-sync with your use cases.

## Functionality Testing:

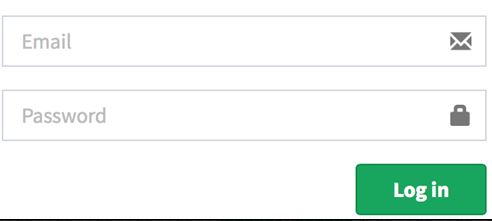
## Functionality testing is a type of software testing whereby the system is tested against the functional requirements/specifications.

Some of the approaches for our application to conduct functional testing are decision table testing , positive and negative testing .

### Decision Table Testing

Decision table testing is a software testing technique used to test system behavior for different input combinations. This is a systematic approach where the different input combinations and their corresponding system behavior (Output) are captured in a tabular form.

One of the case that occur in our application is login i.e the user have to login to go to the dashboard . The decision table for login screen is



The condition is simple if the user provides correct username and password the user will be redirected to the dashboard. If any of the input is wrong, an error message will be displayed.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Conditions | Rule 1 | Rule 2 | Rule 3 | Rule 4 |
| Username (T/F) | F | T | F | T |
| Password (T/F) | F | F | T | T |
| Output (E/D) | E | E | E | D |

Where,

T – Correct username/password

F – Wrong username/password

E – Error message is displayed

D– Dashboard screen is displayed

Interpretation:

Case 1 – Username and password both were wrong. The user is shown an error message.

Case 2 – Username was correct, but the password was wrong. The user is shown an error message.

Case 3 – Username was wrong, but the password was correct. The user is shown an error message.

Case 4 – Username and password both were correct, and the user navigated to dashboard

### Positive and Negative Testing

Positive testing is performed on the system by providing the valid data as input. It checks whether an application behaves as expected with positive inputs. This test is done to check the application that does what it is supposed to do.

Example: There is a text box in an application which can accept only numbers. Entering values up to 99999 will be acceptable by the system and any other values apart from this should not be acceptable. To do positive testing, set the valid input values from 0 to 99999 and check whether the system is accepting the values.

Negative Testing is performed on the system by providing invalid data as input. It checks whether an application behaves as expected with the negative inputs. This is to test the application does not do anything that it is not supposed to do so.

Example : Negative testing can be performed by entering characters A to Z or from a to z. Either software system should not accept the values or else it should throw an error message for these invalid data inputs.

## API Testing :

There are mainly 4 methods involve in API Testing like GET, POST, Delete, and PUT.

**GET-** The GET method is used to extract information from the given server using a given URI. While using GET request, it should only extract data and should have no other effect on the data.

**POST**- A POST request is used to create a new entity. It can also be used to send data to the server, for example, customer information, file upload, etc. using HTML forms.

**PUT-** Create a new entity or update an existing one.

**DELETE-** Removes all current representations of the target resource given by a URI.

API testing requires an application to interact with API. To test an API, you require two things,

* Testing Tool/Framework to drive the API
* Manually writing the code to test the API

Rest API can be tested with tools like:

* Advanced Rest Client
* Postman-Rest Client
* Curl in Linux

# Test Item

* Login module
* Students list view module
* Student details view module
* Searching module
* Sorting by name and score
* CSV export

# Risks and Assumptions

## Assumption

1. The test plan will meet the requirement of the client.
2. The test plan will cover all user stories.
3. Login will be successful by providing valid credentials.
4. Searching and Sorting names, score will be covered in test plan.
5. Test data are sufficient .

## Risk and Mitigation Plan

## 

|  |  |  |  |
| --- | --- | --- | --- |
| S.N | Risk | Priority | Mitigation Plan |
| 1 | Test data are insufficient | High | Testa data should be prepared according to the user requirement , stakeholder communication and data validation with the client |
| 2 | Change in the requirement | High | Strong communication between client and development team |
| 3 | Login is not successful | medium | The user should type correct username and password according to field format |
| 4 | Details of students list are not in view page | Low | API testing should conduct again and again i.e should check frontend connection with backend. |

# Tools

1. Selenium (automation tool)
2. Ghostlab(cross browser testing tool)
3. Postman (API testing tool)

# Test Deliverable

1. Test plan document.
2. Test cases.
3. Summary Reports.
4. Metrics.
5. Bug report .
6. Test data .

# Testing Environment

|  |  |
| --- | --- |
| Environment | Testers |
| Browser | IE,firefox, mozilla |
| OS version | Windows 10 , 8 , Linux, Mac |
| Database | MySQL, ORACAL |
| Programming language | JAVA |

# 

# 

### 

### 