Assignment 6

# Test Suite Design

FOR

# University Department Information System

Group - 58
Kaustubh Hiware - 14CS30011
Surya Midatala - 14CS30017

# Table of Contents :-

# 1. Introduction

## 1.a Test plan identifier

Master Test Plan version 1.0 for DepInfosys.
Date: 13-04-2016
Authors: Kaustubh Hiware, Surya Midatala

## 1.b Purpose

This document sketches out the test plan for the University Department Information (UDIS or informally , DepInfosys).This plan describes the nature , purpose and methodology of all testing activities. This plan is aimed at verifying the functionality and correct working of every aspect of DepInfosys which aims to keep track of academic activities of a department , along with keeping records of transactions , research work , etc. It uses Black Box Testing as well as White Box Testing to uncover the various bugs in the software , which are to be rectified later.

## 1.c Scope

The scope of the Test plan is to ensure whether the designed application meets all the given requirements. The approach described in this document assumes the user to be competent in operating the application. The document would be updated parallely with the overall product.

## 1.d References

- IEEE standard for the SRS template
  *830-1984 — IEEE Guide to Software Requirements Specifications*. 1984.
- Class slides

Also , refer to the SRS , UML and SASD documents of this software.

# 2. Features to be tested

The following features of the application are to be tested in the testing cycle:

- Add course
- Delete course
- Add student
- Manage student's courses
- Enter grades of courses
- Manage inventory
- Manage Treasury
- Manage Research/Publication
- Search transactions

# 3. Features not to be tested

The following features are trivial in their implementation and hence have been omitted from the testing cycle:

- View courses
- View items

# 4. Pass/Fail Criteria

### Pass Criteria

The test is considered to be passed if:

- the correct result is obtained.

### Fail Criteria

The test is considered to be failed if:

- It returns the wrong output
- The software crashes
- The software does not show an error message when an error occurs

# 5.Test Plan

## 5.a Black Box Testing

- **CHARACTERISTICS:**
  - Black box testing is a method of software testing that examines the functionality of an application without peering into its internal structures or workings .
  - This method of test can be applied to virtually every level of software testing: unit, integration, system and acceptance.
  - It typically comprises most if not all-higher level testing, but can also dominate unit testing as well.
  - Typical black-box test design techniques include:
    - Decision table testing
    - All-pairs testing
    - State transition tables
    - Equivalence partitioning
    - Boundary value analysis

- **Black box testing strategy:**
  - Equivalence Class Testing: It is used to minimize the number of possible test cases to an optimum level while maintains reasonable test Coverage.
  - Boundary Value Testing: Boundary value testing is focused on the values at boundaries. This technique determines whether a certain range of values is acceptable by the system or not. It is very useful in reducing the number of test cases. It is mostly suitable for the systems where input is within certain ranges.
  - Decision Table Testing: A decision table puts causes and their effects in a matrix. There is unique combination in each column.

Black box testing has its own lifecycle called Software Test Life Cycle and it is relative to every stage of Software Development Life Cycle.

1. Requirement – This is the initial stage of SDLC and in this stage requirement is gathered. Software testers also take part in this stage.
2. Test Planning & Analysis – Testing Types applicable to the project are determined. A Test Plan is created which determines possible project risks and their mitigation.

3. Design – In this stage Test cases/scripts are created on the basis of software requirement documents.
4. Test Execution - In this stage Test Cases prepared are executed. Bugs if any are fixed and re-tested.

### 5.b Testing tools

The following tools have been used in the testing cycle for performing various tests

- JUnit testing
- TrueCoverage
- Selenium
- LoadRunner
- MicroFocus(DevPartner)

# 6.Testing tasks

### 6.a Black Box Tests

As discussed earlier , black box testing requires us to do identify & test for equivalence classes . We'll be doing the same here.

## #1 : Login Testing

The software has only one actor - that is the Department Secretary .
Input : Password(String)

| **Class** | **Input** | **Expected Output** |
|---|---|---|
| Incorrect Password | Incorrect password | Error message |
| Correct password(refer README) | Correct password | Open Home tab |

## #2 : Add course

Input : Name , Professor (String) , Credit(int 1-10)

| **Class** | **Input** | **Expected Output** |
|---|---|---|
| One or more field left empty | None | Error message |
| Credit entered is invalid | String or unrealistic integer | Error message |
| Correct details | String in name , professor fields ,<br>Integer(<10) in credits | Success message |

## #3 : Add student

Input : Name , Address , Email(Strings),Phone(Number)

| **Class** | **Input** | **Expected Output** |
|---|---|---|
| One or more field left empty | None | Error message |
| Phone number entered is invalid | String | Error message |
| Correct details | String in name , address and email fields ,<br>Integer in phone | Success message |

## #4 : Enroll student

First , a student is to be selected . The available courses are then displayed of which he may be enrolled to.

Input : Mouse clicks

| **Class** | **Input** | **Expected Output** |
|---|---|---|
| No subject selected to enroll | Enroll button is clicked | Error message |

| | | |
|---|---|---|
| Subject selected to enroll | Enroll button is clicked | Update lists |

## #5 : Delete course

The user selects  a course to be deleted .The software deletes the course ,if no student is enrolled in it currently.

Input : Mouse click

| **Class** | **Input** | **Expected Output** |
|---|---|---|
| No course selected | Delete button pressed | Error message |
| Course selected which has current students | Delete button pressed | Error message |
| Course selected with no students | Delete button pressed | Success message |

## #6 : Evaluate / Enter grades

The user must first select a course , of which current students are displayed .Now , the student must be selected and the slider must be used to given the grade.

Input : Mouse clicks

| **Class** | **Input** | **Expected Output** |
|---|---|---|
| No course selected | Submit button pressed | Error message |
| Course selected but student not selected | Submit button pressed | Error message |
| Course selected and student selected | Delete button pressed | Grade in table updated |

## #7:Print student

The user must enter  valid roll number , upon which the record of the student  corresponding to that roll number is displayed.The user may give upto 5 invalid inputs at a time.

Input : Roll number(int)

| **Class** | **Input** | **Expected Output** |
|---|---|---|
| Invalid roll number | String or invalid int | Error message |
| Valid roll number entered | Int as roll number | Records of the student displayed |
| Valid roll number entered | Int as roll number and save as file button clicked | Software generates grade card |

## #8 : Add item to inventory

Input : Name , Location(String),Price (double).

| **Class** | **Input** | **Expected Output** |
|---|---|---|
| One or more fields left empty | None | Error message |
| Price entered is incorrect | String or negative double | Error regarding price |
| Correct details | String in Name , Location , Double in Price | Item added |

## #9 : Delete item

The user must first click on delete item button on inventory tab.
Input : Mouse click

| **Class** | **Input** | **Expected Output** |
|---|---|---|
| No item selected | None | Error message |

| | | |
|---|---|---|
| Item selected | Clicked on an item row | Item removed |

## #10 : Modify item details

The user must first click modify item in the inventory menu.
Input : String

| **Class** | **Input** | **Expected Output** |
|---|---|---|
| No item selected | Modify button clicked | Error message |
| Item selected and one empty field | Modify button clicked and save button clicked | Error regarding missing detail |
| Correct details | Modify button clicked & String in Name , Location | Item modified |

## #11 : Adding transaction

Input : Name , Details(String),Price (double),Type(selected).

| **Class** | **Input** | **Expected Output** |
|---|---|---|
| One or more fields left empty | None | Error message |
| Price entered is incorrect | String or negative double | Error regarding price |
| Correct details | String in Name , Location , Double in Price | Transaction added |

## #12 : Add Research / Publication

Since not every Research/Publication has a cost attached to it , if left empty , investment and/or profit is considered 0.

Input : Name , Details,Incharge(String),Price (double),Type(selected).

| **Class** | **Input** | **Expected Output** |
|---|---|---|

| | | |
|---|---|---|
| One or more string fields left empty | None | Error message |
| Price entered is incorrect | String or negative double | Error regarding price |
| Correct details | String in Name , Location,Incharge, Double in Price | Work added |

## #13 : View Research/Publication

The user must first click on view Research/Publication button.
Input : Mouse click

| **Class** | **Input** | **Expected Output** |
|---|---|---|
| No work selected | None | Error message |
| Work selected | Clicked on a work row | Details displayed |

## #14 : Search Transactions

For record sake , research / publications are also displayed.
Input(Optional) : search key(String)

| **Class** | **Input** | **Expected Output** |
|---|---|---|
| No transaction selected | None | Error message |
| Transaction selected | Clicked on a transaction row | Details displayed |
| Search key entered | String in search bar | Transactions Containing that substring should be displayed |