**Experiment No. 7**

**Aim:** Design test scenarios and test cases for your SRS.

**Performance:**

1. **Create test scenario and test cases for your case study**
2. **Use the following template for the Test Scenario**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test scenario ID | Requirement- reference document index | Test scenario description | Importance | No. of test cases |
|  |  |  |  |  |

**Column #1: Test scenario ID**  
Every entity in our testing process has to be uniquely identifiable. So, every test scenario has to be assigned an ID. The rules to follow while assigning this ID have to be defined. For the sake of this article we are going to follow the naming convention as: TS(prefix that stands for Test Scenario) followed by ‘\_’ , module name MI(my Info module of the Orange HRM project) followed by ‘\_’ and then the sub section (eg: MIM for My info module, P for photograph and so on)followed by a serial number. An example would be: “TS\_MI\_MIM\_01”.

**Column #2: Requirement**  
It helps that when we create a test scenario we should be able to map it back to the section of the SRS document where we picked it from. If the requirements have IDs we could use that. If not section numbers or even page numbers of the SRS document from where we identified a testable requirement will do.

**Column #3: Test scenario description**  
A one liner specifying ‘what to test’. We would also refer to it as test objective.

**Column #4: Importance**  
This is to give an idea about how important certain functionality is for the AUT. Values like high, medium and low can be assigned to this field. You could also choose a point system, like 1-5, 5 being most important, 1 being less important. Whatever the value this field can take, it has to be pre-decided.

**Column #5: No. of Test cases**  
A rough estimate on how many individual test cases we might end up writing that one test scenario. **For example**: To test the login- we include these situations: Correct username and password. Correct username and wrong password. Correct password and wrong username. So, validating the login functionality will result in 3 test cases.

1. **Use the following template for the Test Cases**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test case ID** | **Test Objective** | **Precondition** | **Steps:** | **Test data** | **Expected result** | **Post-condition** |

**Test case ID**: Unique ID for each test case. Follow some convention to indicate types of test. E.g. ‘TC\_UI\_1′ indicating ‘user interface test case #1′.

**Test Objective**: Describe what the test case is actually going to test.

**Pre-condition**: Any prerequisite that must be fulfilled before execution of this test case. List all pre-conditions in order to successfully execute this test case.

**Dependencies**: Mention any dependencies on other test cases or test requirement.

**Test Steps**: List all test execution steps in detail. Write test steps in the order in which these should be executed. Make sure to provide as much details as you can. Tip – to efficiently manage test case with lesser number of fields use this field to describe test conditions, test data and user roles for running test.

**Test Data**: Use of test data as an input for this test case. You can provide different data sets with exact values to be used as an input.

**Expected Result**:  What should be the system output after test execution? Describe the expected result in detail including message/error that should be displayed on screen.

**Post-condition**: What should be the state of the system after executing this test case?

**Actual result**: Actual test result should be filled after test execution. Describe system behavior after test execution.

**Conclusion:**

Thus, we are able to create test scenarios and test cases for our case study.