Learning hub Test Strategy

Contents

[1. Revision History 3](#_Toc471127874)

[2. Project Overview 3](#_Toc471127878)

[3. Test Strategy Purpose 4](#_Toc471127879)

[4. Test Environments 4](#_Toc471127890)

[5. Test Data 4](#_Toc471127892)

[6. Test Users 5](#_Toc471127893)

[7. Testing Responsibilities 5](#_Toc471127894)

[8. Test Types 6](#_Toc471127897)

[8.1 System Test 6](#_Toc471127900)

[8.2 User Acceptance Test (UAT) 7](#_Toc471127901)

[9. Test Deliverables 8](#_Toc471127931)

[10. Project Testing Related Tools 8](#_Toc471127933)

[11. Defect Management 9](#_Toc471127934)

[11.1 Defect reporting and resolution process 9](#_Toc471127937)

[11.2 Defect severity definitions 10](#_Toc471127939)

[11.3 Defect life cycle stage 11](#_Toc471127940)

# Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Version No.** | **Date** | **Revised By** | **Description of Change** |
| Version 1 | 24/2/2023 | Hasnaa Ahmed |  |
|  |  |  |  |
|  |  |  |  |

# Project Overview

Learning hub is a web platform that will provide tons of information so users can receive and explore knowledge in different life domain and also to be part of the community to contribute and share their knowledge

Information technology institute (ITI) has recently approved " learning hub web application" project to move forward for project initiation within the research and development (R&D) group. While " learning hub web application" is currently available, ITI believes that new technological developments will enable our team to develop a solution far superior to what is currently available.

# Test Strategy Purpose

Test strategy is a guideline to be followed to achieve the test objective and execution of test types. It deals with test objective, test environment, test approach, automation tools and strategy, contingency plan, and risk analysis.

# Test Environments

when creating test environments, we must:

* Set up database.
* Create test data.
* Select the right browsers.
* Select the right hardware and operating system.
* Configure the network.

# Test Data

1. Identify unique data required by the application such as User Ids and Passwords.
2. Prepare all types of data:

* normal data - typical, sensible data that the program should accept and be able to process.
* boundary data - valid data that falls at the boundary of any possible ranges, sometimes known as extreme data.
* erroneous data - data that the program cannot process and should not accept.

1. Attention to detail such as drop-down boxes and buttons.
2. Also determine if DB needs to be seeded with additional data.

# Test Users

Each test case will require one or more test users. Test users must be created to replicate real business users allowing defects related to authorisation profiles and delegation of duties to be identified. Using unrealistic role assignment for test users will invalidate all functional test

# Roles and Responsibilities for Test Events

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Phase** | **Activity** | **Test Lead** | **Dev Lead** | **PM** | **Comments** |
| Requirement Analysis | Providing detailed list of requirements in scope for the release | A&R | I | I |  |
| Test Case Development | Requirement analysis and test case development | A&R | C | I |  |
| Test Execution | Perform functional system testing and report results | A&R | R | C |  |
| Test Execution | Bug fixes and retest | R | A&R | C |  |
| Test Execution | Test Results Review and Sign off | C | C | R |  |
| Acceptance test execution | UAT execution | C | C | R |  |
|  | | | | | |

R – Responsibility A – Accountability C – Consulted I – Informed

# Test Types

## System Testing

|  |  |
| --- | --- |
| Purpose | SIT validates a set of business processes that define a business scenario in a comprehensive and self-contained manner on a macro level.  This is an end-to-end test of the business process. Typically, a business scenario will involve testing multiple test cases together. The primary objective of this testing is to discover errors in the integration between different modules and to verify that the modules work together correctly as one function. |
| Test Scope | * Full End to end business process. * Regression * Interface testing with interfacing systems |
| Entry Criteria | * Proper test data is available. * Test plans and test cases are reviewed and signed off. * Smoke testing has passed. |
| Exit Criteria | * Test case execution completed with 100% passed. * All defects are recorded. * No outstanding “showstopper or severe” defects. * All test results have been documented. * Coverage of code/functionality/requirements is 100% of functional requirements. |
|  | |

## User Acceptance Test (UAT)

|  |  |
| --- | --- |
| Purpose | User acceptance test is performed by business users. The users test the complete, end-to-end business processes to verify that the implemented solution performs the intended functions and satisfies the business requirements. |
| Test Scope | * UAT * Full Regression |
| Entry Criteria | * The application works functionally as defined in the specifications. * No outstanding “showstopper or severe” defects. * Entire system functioning and all new components available unless previously agreed between UAT stakeholder/Test manager and project managers. * All test cases are documented and reviewed. * System testing is done. |
| Exit Criteria | * The Acceptance Tests must be completed, with a pass rate of not less than 95%. * No outstanding “showstopper or severe” defects. * No new defects have been discovered for a week prior to Production Implementation. * All test results recorded and approved. * UAT test summary report documented and approved. |
|  | |

# 

# Test Deliverables

* Test Strategy.
* Test cases.
* Bug-tracking system report of all issues raised during testing process.
* Test Coverage measurement.
* Test summary report.

# Project Testing Related Tools

|  |  |
| --- | --- |
| **Phase/activity** | **Test tool requirement** |
| Test case documentation (Manual & Automation) | JIRA |
| Requirement Management | JIRA |
| Test execution and results reporting | JIRA |
| Defect reporting and tracking | JIRA |
| Document storage | GitHub |

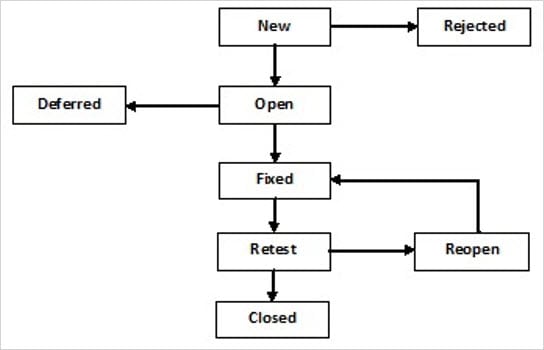
# Defect Management

Defect Review Meetings will be held on a daily. The goal of this meeting is to ensure that defects are being resolved in a timely fashion and that any issues or questions are resolved. It is at these meetings that progress tracking of defect resolution and closure is communicated.

## Defect reporting and resolution process

Development team should have access to defects section and are able to update the defect details. Testing team should be configured to send auto emails when a new defect is logged, assignee is changed and the status is moved to re-test.

Below diagram help in understand the defect life cycle process quickly and easily.



## Defect severity definitions

|  |  |  |
| --- | --- | --- |
| **Severity** | **Definition** | **Expected time for Closure** |
| **Critical** | A complete software system, or a subsystem, or software unit (program or Module) within the system lost its ability to perform its required function (=Failure) and no workaround available.  OR  Testing of a significant number of tests cannot continue without closure. | Only 1 Day |
| **Major** | The software system, or subsystem, or software unit (program or module) within the system produces Incorrect, Incomplete, or Inconsistent results. | 2 Business days |
| **Minor** | Everything that not Major or Critical | 3 Business days |

## Defect life cycle stage

As part of the Defect Life Cycle definition and Defect Management process, various Defect stages will be identified as mentioned below

| **Defect Status** | **Description** | **Required Previous Status** | **Next Possible Status** |
| --- | --- | --- | --- |
| **New** | - Defect identified and raised by a Team - Defect is not reviewed by the Assigned Team | NA | Open Rejected |
| **Open** | - Assigned Team acknowledges the defect by moving the defect to open status - No one has been assigned to analyze the defect | New | Fixed Deferred |
| **Rejected** | - An invalid defect has been logged. The defect can be rejected by the Assigned Team for various reasons  - Invalid data used by tester  - Invalid test case executed by tester  - Test steps followed by the tester were incorrect | New | Closed |
| **Fixed** | - Assigned Team moves the defect to fixed when the defect is fixed and is ready to be deployed | Open  Re-open | Re-test |
| **Re-test** | - Assigned team moves the defect to Retest when the defect has been fixed. | Fixed | Closed Re-Open |
| **Re-Open** | - If a defect in Retest Fails, then the defect is Reopened and assigned back to the previous team which fixed the defect Note: If the retest of the defect fails because of a reason different than what the defect was logged for then a new defect should be open for the new issue. The current defect shouldn't be reopened in such cases | Retest | Fixed |
| **Closed** | - Defect passes the retest and can be closed. | Retest | - |
| **Deferred** | - Defect is acknowledged by the Assigned Team and cannot be fixed with the Release timeline because of any constraints. The defect then will be deployed to production with known risk. | Open | - |