**OrangeHRM Application**



**qUALITY aSSURANCE tEST STRATEGY**

**Document History:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Version** | **Date** | **Author** | **Reviewer** | **Approver** |
| OrangeHRM OS 5.5 | 14-Sept-23 | Pulkit Purohit  Shweta Misal  Saba Karigar  Siddhant Pednekar | Gwen Iarussi |  |

**Table of Contents**

|  |
| --- |
| **Chapter 1: Introduction** |
| 1.1. Purpose |
| 1.2. Project Overview |
| 1.3. Goals & Objectives |
| 1.4. QA Testing Process Flow |
| 1.5. System Model |
| **Chapter 2: Scope of Testing** |
| 2.1. Overall, Scope of the Application |
| 2.2. High-Value/Risk Areas |
| 2.3. In Scope of The Project |
| 2.4. Type of Testing Included |
| 2.5. Type of Testing Excluded |
| **Chapter 3:** **Automation Script Development Process** |
| 3.1. Automation Script Development Process Flow Diagram |
| **Chapter 4: Test Approach** |
| 4.1. Test Specification |
| 4.2. Automation Execution Cycle |
| 4.3. Test Environment or tool selection |
| 4.4. FrameworkSelection |
| 4.5. Test Script Development/Maintenance |
| 4.6. Test Reporting |
| 4.7. Automation Test Case Naming Conventions and Folder Structures |
| **Chapter 5: Risk Management** |
| 5.1. Assumptions |
| 5.2. Constraints |
| 5.3. Automation Risks and Mitigation Plans |
| **Chapter 6: Defect Management** |
| 6.1. Defect Capturing and Assignment |
| 6.2. Defect Severity |
| 6.3. Defect Priority |

# **Chapter 1: Introduction**

## 1.1. Purpose

The purpose of this document is to provide a high-level strategy for the OrangeHRM Testing Project. This document is one of the agreed deliverables of the OrangeHRM Testing project. The document explains the detailed level scope, approach, entry and exit criteria, defect management process, dependencies, test case naming conventions, test deliverables, etc. for the OrangeHRM testing project.

## 1.2. Project Overview

OrangeHRM is an open-source web based human resource management software that provides a comprehensive HR (Human Resource) management solution for small to medium-sized businesses. This application is used by organizations to manage and automate their HR processes, by providing features such as employee information management, leave management, time and attendance tracking, performance management, and more.

The purpose of this testing project is to verify the functionality and reliability of the OrangeHRM application, and it will involve testing of the various modules and features of the software to ensure that they are functioning as expected. Also, it will involve testing the application for change requests, regression testing, and ad-hoc SIT/regression testing based on the request from stakeholders.

## 1.3. Goals & Objectives

### Product Quality Goals

Test automation goals are to check the behavior of application, objectives are centered on facilitating the test process and boosting its efficiency.

The major goals of the automation testing are as follows

* **Scalability**: Ensure that the OrangeHRM platform can handle a growing number of users, modules, and data while maintaining the correctness of its functionality.
* **Reliability**: Confirm the proper functioning of each HRM (Human Resource Management) component within the OrangeHRM platform, both independently and when integrated with other modules.
* **Efficiency**: Identify redundant test cases within the HRM system to automate repetitive tasks and streamline the testing process.
* **Regression Testing**: Automate regression testing for critical HRM functionalities to reduce manual efforts and ensure the stability of the system after updates or changes.

### Testing Objectives

1. **End-to-End Functionality Testing**: Verify that all aspects of the OrangeHRM system, from employee data management to payroll processing, are functioning correctly.
2. **Scope Definition and Risk Mitigation**: Clearly define the scope of testing, including which modules and features will be tested, and identify potential risks to the project's successful completion. Develop mitigation plans for these risks.
3. **Reporting**: Regularly report on the status of testing activities, including test progress, defects found, and test results. Ensure transparency in reporting to facilitate decision-making.
4. **Consistency with Standards**: Ensure that the testing process aligns with established standards and follows documented processes to maintain quality and consistency.
5. **Quality Assurance**: Continuously assess and confirm that the OrangeHRM system meets the expected quality standards and complies with HRM best practices.
6. **Regression Testing**: Automate the testing of critical end-to-end scenarios in the OrangeHRM system to validate that recent changes or updates do not introduce defects

**1.4. QA Testing Process Flow**

1. **Release Calendar**: Establish a release calendar that outlines the planned release dates for new versions or updates of the OrangeHRM platform.
2. **Requirement Gathering**: Collaborate with stakeholders, such as HR professionals and software developers, to gather detailed requirements for testing each module and feature.
3. **Scope Definition and Coordination**:

* Define the scope of testing for the OrangeHRM project, specifying which modules and functionalities will be tested.
* Coordinate with different regions or entities that will be using OrangeHRM and ensure their specific needs are considered.

1. **Test Planning and Execution**:

* Create test cases tailored to the OrangeHRM platform or adapt existing ones.
* Update the test management system (e.g., ALM (Application Lifecycle Management) - Application Lifecycle Management) with test cases and requirements.
* Prepare test data sets representative of real-world scenarios.
* Execute tests following a well-defined test plan.
* Provide daily status reports to keep stakeholders informed about testing progress.

1. **Defect Triage/Resolution and Retest**:

* Log and prioritize defects found during testing.
* Collaborate with the development team to resolve defects.
* Retest defects after they have been fixed to ensure they are resolved.

1. **QA SIT Sign-off**: After successful testing and defect resolution, obtain sign-off from the Quality Assurance team, indicating that the OrangeHRM system is ready for release.

# **Chapter 2: Scope of Testing**

OrangeHRM is a comprehensive human resource management software that typically covers a wide range of HR-related functionalities. The scope of such an application typically includes:

## **2.1.** Overall, Scope of the Application**:** -

## Employee Management: This includes features for adding, updating, and maintaining employee records, including personal details, employment history, and documents.

## Leave Management: Managing employee leave requests, approvals, and tracking leave balances.

## Time and Attendance: Recording and managing employee attendance and work hours.

## Payroll: Calculating employee salaries, tax deductions, and managing payroll processes.

## Recruitment: Posting job vacancies, managing applications, and conducting interviews.

## Performance Management: Assessing employee performance through evaluations, goal setting, and performance reviews.

## Training and Development: Managing training programs and tracking employee skills and development.

## Benefits Administration: Administering employee benefits such as health insurance, retirement plans, and other perks.

## Reports and Analytics: Generating reports and analytics related to HR data.

## 2.2. High-Value/Risk Areas

* **Admin Module:** The Admin Module provides you with full control of all settings that affect the action of your OrangeHRM implementation.
* **PIM (Personal Information Module):** This module maintains all relevant employee-related information. All information about an employee can be entered here. Information captured in this module is utilized by all other modules, thus eliminating data redundancy. The PIM Module will be available to the admin with full control and supervisors with restricted access showing his subordinates.
* **Leave Module:** The Leave Module automates the HR administrative tasks of recording leave and controlling these against leave policies defined in the HR system. The module provides flexibility in allowing you to define several types of Leave, including Annual Leave, Sick Leave, Travel leave etc. The Leave Module can send notifications to covering officers and allows you to record, track leave and view leave history.
* **Time Module:** Business-critical operations require reliable tracking and control to maximize profits and reduce operational costs. A time management tool is one of the vital employee work time management features that make the entire difference between successful HR-Management and a weak one. The Time module automates the time tracking process. While allowing the employee to define and

submit their time sheets the supervisors can approve/reject or even modify them.

* **Dashboard:** Verify that the system generates accurate and useful reports on employee data, including attendance, performance, and payroll.

## 2.3. In Scope of the project

|  |  |  |
| --- | --- | --- |
| **Module** | **Quantitative Data** | **Qualitative Data** |
| **Admin Module** | This module covers 90% of all user interactions based on historical usage data. | It includes critical features such as employee onboarding and data updates, making it a high-priority area. |
| **PIM** | Manages employee data records. | Data privacy and security are paramount. |
| **Leave Module** | Approximately 80% of employees regularly use this module to request leaves. | Ensuring accurate leave management is crucial for employee satisfaction. |
| **Time Module** | Tracks employee hours and attendance.  Employee time entries: 1,000/day. | Precision needed for payroll calculations.  Real-time updates for scheduling. |
| **Dashboard** | The dashboard module covers 95% of critical KPIs, with 85% of employees engaging daily and real-time updates for data from multiple sources. | Users report positive feedback on the dashboard's ease of use and its positive impact on decision-making. |

## 2.4. Type of Testing Included

**Functional Testing:** Testing all modules for functional correctness, including data validation, calculations, and feature interactions.

**Integration Testing:** Testing the integration points between different modules and any external systems.

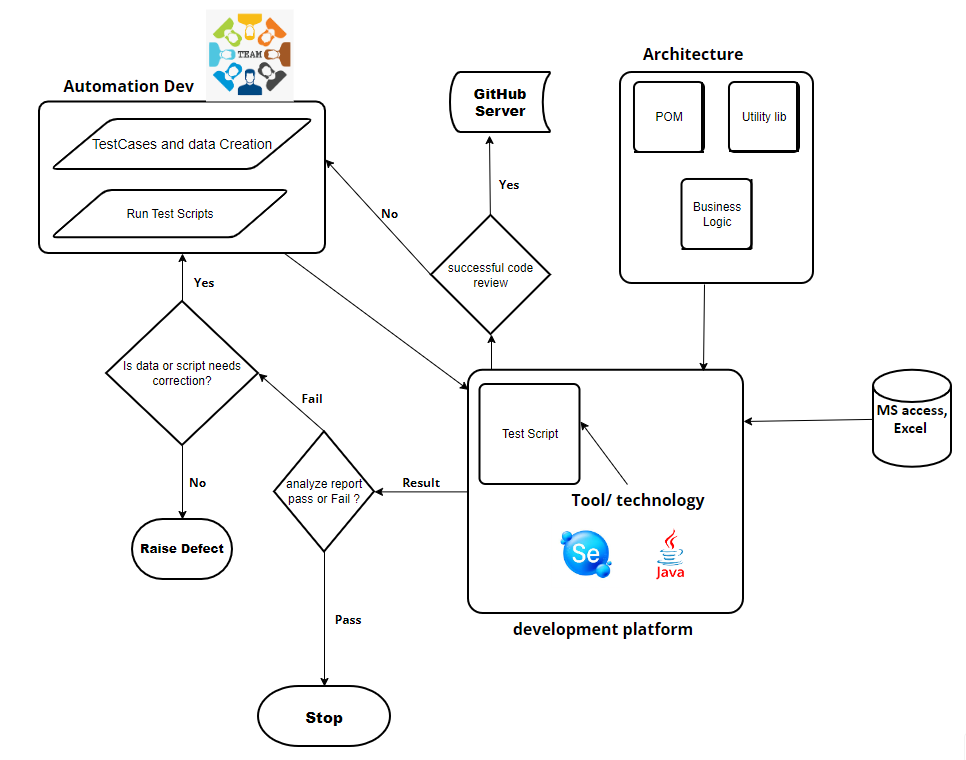
**Regression Testing:** Regularly retesting all modules to catch any unintended side effects of code changes.

## 2.5. Type of Testing Excluded

* API and Database testing
* Compatibility testing
* Manual Regression Testing

# **Chapter 3: Automation Script Development Process**

## 3.1. Automation Script Development Process Flow Diagram



# **Chapter 4: Test Approach**

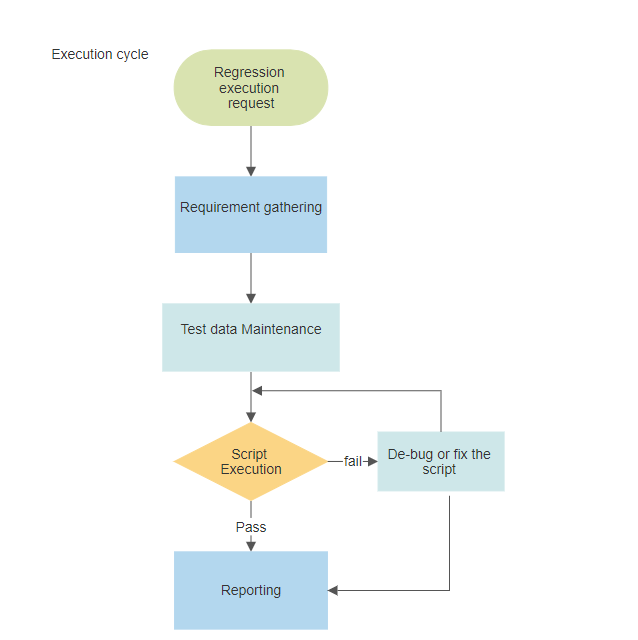
## 4.1. Test Specification

Regression suite comprises of below modules:

|  |  |  |
| --- | --- | --- |
| Modules | Tester Name | Module\_TestCase\_Count |
| PIM | Siddhant | 16 |
| Leave | Shweta | 17 |
| Time | Saba | 15 |
| Dashboard | Pulkit | 47 |
| Login | Pulkit | 5 |
| Total Test cases | | 100 |

|  |  |  |  |
| --- | --- | --- | --- |
| [Orange Application](file:///C:\Users\u1251792\AppData\Local\Microsoft\Windows\INetCache\Content.MSO\BBE09175.xlsx#RANGE!A1) | | | |
| **Details/Test Scenario Complexity** | **Low** | **Medium** | **High** |
| No of Test Case | 17 | 26 | 40 |
| Script Development for each TC (in hrs.) | 4 | 6 | 8 |
| **Total Script development Efforts (in hrs)** | 68 | 156 | 320 |
| Review (Development + functional team) | 3.4 | 7.8 | 16 |
| Review incorporation | 3.4 | 7.8 | 16 |
| **Batch Dry Run** | 60 | | |
| **Script development total efforts** | 658 | | |
| **Additional Efforts(in Hrs)** | | | |
| Framework Modification | 33 | | |
| Contingency | 165 | | |
| **Total Efforts(in hrs.)** | 856 | | |
| Total Efforts based on skill competency | **1070** | | |
| Total Efforts in person(in Day's) | 134 | | |

## 4.2. Automation Execution Cycle



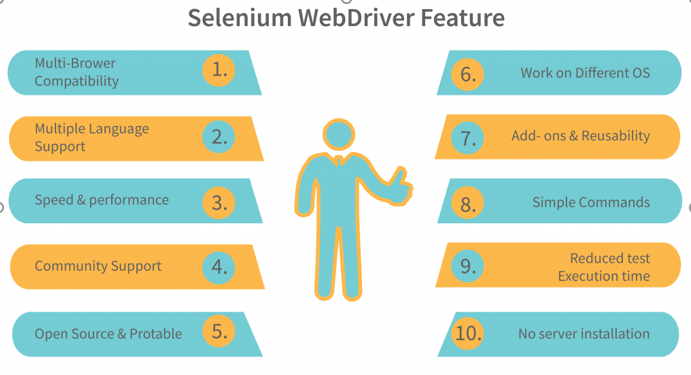
## 4.3. Test Environment and tool selection

In this step, check the feasibility of application which needs to be automated. Feasibility checks should be performed before starting automation testing, this includes shortlisting of the relevant test cases for automation. Identifying the right automation testing tool is critical as automation testing is highly dependent on the tool.

|  |  |  |
| --- | --- | --- |
| Comparison Basis | UFT (QTP) | Selenium WebDriver |
| Software Type | It is a Desktop based Application. | It is a set of APIs. |
| Cost | Paid Tool. You need to purchase a license to use it. It is available as seat-based and concurrent, which costs more. | Open Source/Free tool. You can download and use it for free. |
| Application Type | It supports web, mobile, API, hybrid, RPA, and enterprise apps. | Selenium can be used only for Web-based applications. It is a major disadvantage of selenium over QTP. |
| Supported Languages | VBS (Visual Basic Script) | Java, C#, Ruby, Python, Perl PHP, JavaScript, R etc. |
| Supported Browsers | Chrome, Firefox, Safari, IE, and Edge | IE, Edge, Firefox, Chrome, Safari, Opera, Headless browsers |
| Supported Operating Systems | Only Microsoft Windows | It supports Microsoft Windows, Apple OS X, Linux |
| Test Execution | Combine it with Micro Focus ALM to execute tests synchronously and ALM is also paid one. | It can run tests synchronously with the integration of test frameworks such as TestNG which is free of cost. |
| Support | Since it is a paid tool, so they provide proper support. | Since it is an open-source tool, no professional support is provided but it has good community support. |

We have selected **Selenium** for its compatibility with our application.

* Open Source
* Supports Web Based Application
* Supports most of the majorly used languages. **Java**, C#
* Supports most of the majorly used browsers. (**Chrome, Edge**)
* Good Community support.



### Test Environment: -

The automation testing will be executed in the following environment:

|  |  |  |
| --- | --- | --- |
| Items | Tools | Version |
| Browsers |  | Microsoft Edge  Google Chrome |
| Test Management | ALM | 15.05 |
| Database | MS access, Excel as Database |  |
| Documentation Tools |  | Word, Excel, PDF |
| Automation Tool | UFT | UFT 2021 v |
| Code Repository | GitHub |  |
| Automation Reporting | Selenium testNG default reporting |  |
| VBScript Language |  |  |

## 4.4. Framework Selection

We will be using **Hybrid Framework** because of below advantages,

* Code Reusability
* Supports Page Object Model
* Supports Data Driven Framework
* Customized Reporting (Extent Reports, Spark Reports)
* Exception Handling
* Parallel Execution (Future Scope)

## 4.5. Test Script Development/Maintenance

* **Clear Test Data Management Approach:**

Every time a change is made to the application, refresh the database. Build a new set of data every time you run the scripts. A clear test data management approach can help in maintaining the test automation suite and contribute to bringing a clear ROI.

* **Regular health check of automation suites:**

Get regular health checks of the automation suites. Regular checks will ensure automation suites perform as anticipated and give the best results.

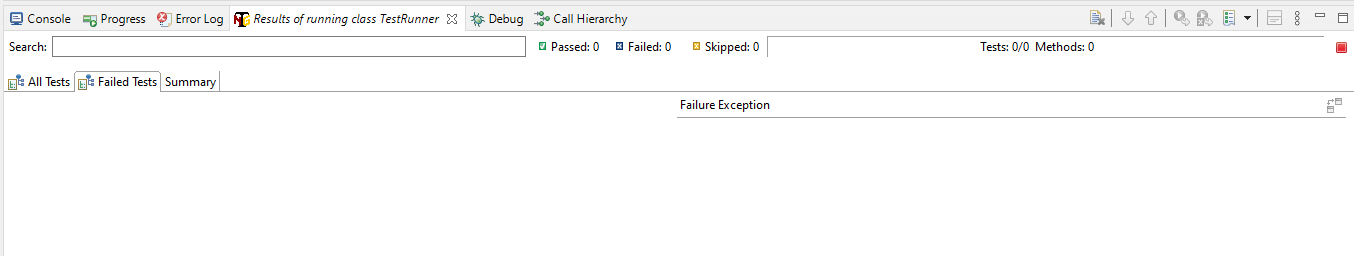
* **Impact analysis of any new enhancement in the application:**

The analysis inspects the proposed change to recognize modules to be created, modified, or rejected and to estimate the effort associated with implementing the change. Based on the analysis, QA should be planned to update the impacted automated scripts during the release itself.

## 4.6. Test Reporting

The use of Selenium, testNG default reporting.

Below is the snapshot of test reporting



## 4.7. Automation Test Case Naming Conventions and Folder Structures

-Pending

# **Chapter 5: Risk Management**

## 5.1. Assumptions

1. **Application Stability**: The OrangeHRM open-source demo application is assumed to be stable and not prone to frequent, disruptive changes during the automation testing process.
2. **Access to Required Resources**: The automation testing team has access to the necessary hardware, software, and infrastructure resources required for test automation, including test environments and test data.
3. **Application Documentation**: Sufficient documentation and information about the OrangeHRM open-source demo application's architecture, functionality, and APIs are available to support test script development and testing activities.
4. **Collaboration and Communication**: Effective collaboration and communication channels are established among team members, developers, and other stakeholders to address issues, share updates, and track progress.
5. **Stakeholder Support**: Stakeholders, including project managers and decision-makers, are supportive of the automation testing efforts and are willing to allocate resources and address issues as they arise.
6. **Infrastructure Stability**: Test automation tools and software licenses are assumed to be in compliance with legal and organizational requirements.
7. **Data Privacy and Security Compliance**: Test data used for automation testing activities is assumed to be properly sanitized, anonymized, or masked to comply with data privacy and security regulations.

## 5.2. Automation Risks and Mitigation Plans

|  |  |
| --- | --- |
| **Risks** | **Mitigation Plans** |
| The possibility that automation Testing’s allotted time will be cut short. | Organize test cases by priority, and test the high priority test cases first, leaving the lower test cases until time permits their execution. |
| Test scripts become outdated and require frequent updates due to application changes | Implement a robust test script design and structure that is easy to maintain.  Use version control for test scripts to track changes and collaborate effectively. |
| Application issue such as Browser compatibility, Object Identification, synchronization issue may be encountered due to new change deployed in the application | Work around with issues to overcome them. |
| The loss of existing data occurs due to the system being refreshed every hour. | Execute E2E test script every time |
| Ensuring the availability of appropriate and clean test data for automated testing | Establish a test data management strategy that includes data generation, data masking, and data provisioning.  Automate data setup and teardown processes as part of test scripts. |
| Automated tests may not cover all critical test scenarios or edge cases. | Conduct regular test coverage reviews and gap analysis.  Use code and test coverage analysis tools to identify untested areas. |
| Automated tests may encounter intermittent failures or instability. | Rerun failed tests automatically to verify the stability of reported issues.  Implement explicit waits and robust error-handling mechanisms. |

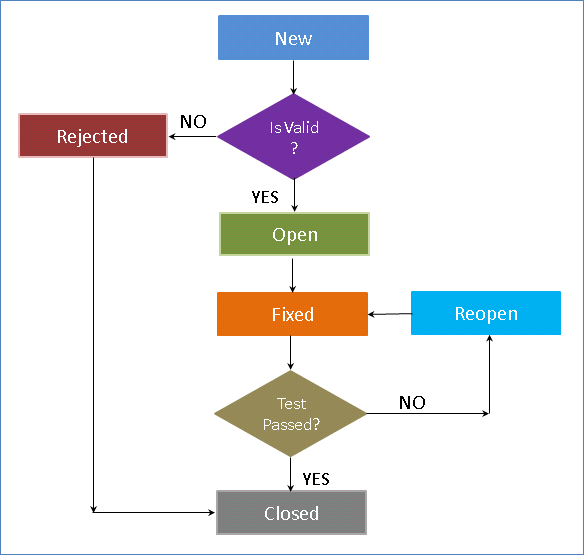
# **Chapter 6: Defect Management**

ALM is used for reporting, tracking, and managing Defects.

## 6.1. Defect Capturing and Assignment

Test team accesses ALM through the web and reports the test execution results and defects into this tool daily in the test execution phase. The defects are assigned to the defined SPOCs in ALM for resolution. The testers will retest the defects upon resolution and change the status of the defect to Closed or Reopened in ALM. The development team will investigate the reopened defects. This iteration process will continue until the tester successfully verifies the fix and updates the defect to Verified/Closed status.

The Defect life cycle is explained below



|  |  |
| --- | --- |
| Status | Explanation |
| New | When a defect is posted, the default status is ‘New’ |
| Open | When the defect is accepted by developers it is moved to ‘Open’ Status |
| Rejected | When the defect is rejected by developers it is moved to ‘Rejected’ Status |
| Fixed | When the defect is fixed by developers it is moved to ‘Fixed’ Status. Testers would pick up all defects for testing that are in status ‘Fixed’. |
| Reopen | If the testing has failed, the defect is moved to ‘Reopen’ status |
| Closed | If the testing has passed, the defect is moved to ‘Closed’ Status. |

## 6.2. Defect Severity

Indicate the impact each bug has on testing efforts or users and administrators of the application under test. This information is used by developers and management on the basis for assigning priority of work on defects

The bugs in Issue track will be classified as follows:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **#** | **Severity** | **Functional Issue** | **Configuration/ security Issue** | **Usability Issue** | **Data\*** | **Workflow/ logic** |
| **1** | **Critical** | Showstopper issue, user is unable to interact with major functionality. System crashes while performing functionality | Showstopper, user is unable to access role appropriate functionality or has inappropriate access causing a possible compliance or regulatory issue | N/A | Showstopper issue, Data Corrupted, unable to save, unable to refresh | Showstopper, user is unable to execute critical functionality following primary workflow as prescribed by use case |
| **2** | **High** | Major functionality does not work as expected or limited; errors display. Minor functionality is unavailable | Configuration does not match spec with impact major functionality. | Missing icons or visual cues, Major navigation issue | Large Data elements are available but is missing or wrong data is populated | User is unable to execute secondary functionality following primary workflow as prescribed by use case. Workflow logic is incorrect |
| **3** | **Medium** | Major functionality works but exhibits odd/unexpected behaviors. Minor functionality does not work as expected or limited; errors display. | Configuration does not match spec with impact to major functionality | Major user interface issue (color, text, navigation, display of pop-up, printing) | Large Data elements are populated incorrectly: wrong order, sort or format. Small data elements are missing or wrong. | Workflow logic is confusing or counter intuitive |
| **4** | **Low** | Minor functionality works but exhibits odd/unexpected behaviors. |  | Minor user interface issues (colors, typos, alignment, text of error messages) | Cosmetic data issues: font size, alignment | Navigation/ prompts are mislabeled or not user friendly. |

## 6.3. Defect Priority

Indicates the importance or urgency of fixing a defect.

|  |  |
| --- | --- |
| **Priority** | **Description** |
| **1-Urgent** | Key functionality has a defect and needs urgent resolution |
| **2-High** | This defect comes after Urgent as it needs less attention but still more than other remaining priority categories. |
| **3-Medium** | A defect with minor severity that need not be fixed right away as it does not cause any significant functionality issues or business. These defects are fixed after Urgent and High priority defects are removed |
| **4-Low** | A defect that does not have any major impact on functionality of software and hence does not need any immediate attention. It can be repaired in future or once higher priority defects are fixed. All the Low severity defects fall into this category. |