**TEST PLAN DOCUMENT OF** **OangeHRM**

**Objectives**

The primary objective of this test plan is to validate the functionality, usability, performance, and reliability of the OrangeHRM application. This includes ensuring that all key modules such as login, employee management, leave management, time tracking, recruitment, and performance are functioning according to specified requirements. The testing process aims to identify and report bugs or inconsistencies, verify fixes, and ensure that the system meets quality standards before deployment. The ultimate goal is to deliver a robust and error-free HR management system to end users.

**Scope**

This test plan covers the functional and non-functional testing of the OrangeHRM application. The testing will focus on the core modules of the system, including but not limited to:

* User Login and Authentication
* Employee Management (Add/Edit/Delete Employees)
* Leave Management (Apply, Approve/Reject Leave Requests)
* Time Tracking and Attendance
* Recruitment Module (Job Posting, Applications, Candidate Selection)
* Performance Management
* Admin Module Settings
* Dashboard and Reporting Features

The testing will include manual test cases execution and defect reporting. Both positive and negative test scenarios will be considered.

**Environment** **Test**

The OrangeHRM application will be tested in the following environment:

**Operating System:**

* Windows 10
* Ubuntu 24.04 LTS
* **Web Browsers:** Google Chrome (v136), Mozilla Firefox (v125)
* **Server Environment:** Apache Tomcat 9
* **Database:** MySQL
* **Application URL:** http://test.orangehrm.local
* **Test Tools:** Selenium IDE (for automation), Jira (for defect tracking)
* **Other Requirements:** Stable internet connection, test data for employee records

**Defect** **Reporting**

Jira

**Testing** **Types**

In this project, the following types of testing will be performed to ensure the quality and reliability of the OrangeHRM application:

**1**. **Functional Testing ( black box testing):** Verify that all features and functionalities of the application work as per the requirements.

**2. Regression Testing:** Ensure that new code changes do not adversely affect the existing functionalities

**3. Performance Testing:** Assess the application’s responsiveness, stability, and scalability under expected load conditions.

**4. Security Testing:** Identify vulnerabilities and ensure data protection and secure access.

**5. Usability Testing:** Evaluate the user interface and user experience for ease of use and intuitiveness.

**6. Smoke Testing:** quick check to ensure the basic features of a software build work correctly

**7. Integration Testing:** Test the interaction between different modules and external systems.

**8. System Testing:** It is end-to-end testing of the complete application to ensure all modules work together as expected.

**9. Load Testing:** This testing checks how the application performs under expected user load and identifies performance issues.

**10. Automation Testing:** It involves using tools to automatically run test cases to save time and improve efficiency.

**11. Unit Testing ( white box testing):** Testing of individual software components by developers to ensure they work correctly before integration

**12. Sanity Testing:** Quick check to verify that recent changes work and core functions are not broken.

**13. End-to-End Testing:** Testing the full application flow to ensure all parts work together correctly.It simulates real user scenarios from start to finish.

**14. User acceptance testing (UAT):**

* UAT is performed by the actual users or clients to verify that the software meets their business requirements.
* It is the final testing phase before the software is released to production.

**15. Stress testing:**

* Testing to see how the system performs under extreme or heavy load.
* It checks system stability and identifies breaking points.

**16. Cross Browser Testing:** Testing the application on different browsers to ensure consistent behavior and appearance (Chrome, Firefox, Edge, Safari)

**Schedule**

|  |  |  |  |
| --- | --- | --- | --- |
| **Activity** | **Start Date** | **End Date** | **Description** |
| Test Plan Preparation | 10 Oct | 11 Oct | Test Scope, objectives and Strategy |
| Test Case Design | 12 Oct | 15 Oct | Test Case (Login, PIM, Leave) |
| Test Environment Setup | 16 Oct | 16 Oct | Stable build install on QA server of OrangeHRM |
| Smoke Testing | 17 Oct | 17 Oct | Basic testing of High-level functionality |
| Test Execution | 18 Oct | 25 Oct | Run the test cases |
| Defecting Reporting and Retesting | 26 Oct | 02 Nov | After fixing the bug |
| Regression Testing | 03 Nov | 04 Nov | Ensures that new changes have not broken existing functionality |
| Final Reports and test closure | 05 Nov | 06 Nov | Final test Summary |

**Entry** **criteria**

* The Functional Requirements Document (FRD) has been reviewed and approved
* The Test Plan has been reviewed and signed off by stakeholders.
* All test cases have been designed, reviewed, and are ready for execution.
* The QA environment is fully configured and accessible.
* A stable build (Build 1.0 or later) has been deployed to the QA environment.
* Required test data is available and accessible.
* All necessary tools (e.g., defect tracking, test management) are in place.

**Exit** **criteria**

* All planned test cases have been executed.
* All critical and high severity defects have been fixed and successfully retested.
* No open blocker or showstopper defects remain.
* Regression testing has been completed with acceptable results.
* The test summary report has been prepared and reviewed.
* Formal sign-off has been received from the QA lead and relevant stakeholders

**Testing** **process**

* QA will execute case
* QA will report bugs in JIRA
* QA will assign bug to Project Manager

**Fixing**

Developer will fix the assigned bug and assign it to QA

**Verification**

QA will verify the fix

**Development** **test** **plan**

1. Test Environment (QA)

Used by the QA team for executing functional, integration, and regression test cases.

All builds will be deployed here for formal test execution.

1. Staging Environment

Mimics production. Used for final round of sanity or UAT (User Acceptance Testing).

1. Production Environment

Final live system for end-users

No formal QA testing, only post-release smoke checks or monitoring.

**Error** **Management**

All defects found during testing will be logged, tracked, and managed using a defect tracking tool (e.g., Jira). Defects will be prioritized by severity and status until resolved and verified.

**Configuration** **Management**

All code, builds, test cases, and environment settings will be version controlled using tools like Git to ensure consistency and traceability throughout the testing process.

**Issue** **and** **Assumption**

* Testing needs
* Test system for desktop application
* Printer
* Card Reader
* Mobile
* Access to DB

**Risk**

Certain items may currently be out of scope, but may be accommodated depending on budget availability and managerial approval

**Approval**

The following individuals have reviewed and approved this Test Plan. Their signatures indicate agreement with the testing approach and authorization to proceed

Project Manager = Ahmad Khan

QA Lead = Sana Khan

QA Manager = Ali Hassan