
Quality Assurance Plan

Actitime

3rd November 2023

Asela Pathirage
19020538

1 Introduction

1.1 PURPOSE

The purpose of this test plan is to ensure that the HR Management application functions as intended and meets the specified requirements. Specifically, the purpose is to verify that HR users can successfully log in with valid credentials, access employee profiles, review leave and attendance reports, and perform approval or rejection of timesheets without encountering critical issues or unexpected behaviour. This testing is essential to confirm the application's reliability, security, and user-friendliness for HR personnel.

1.2 PROJECT OVERVIEW

The Actitime HR Management application stands as a robust platform, central to HR operations, offering essential features such as employee profile management, comprehensive leave and attendance tracking, and a streamlined process for timesheet approval or rejection. In a bid to ensure the application's efficiency, the project's primary objectives revolve around three core areas: functionality, security, and performance. Functional testing is instrumental in confirming that HR users can seamlessly navigate and utilize these key features, with a focus on user-friendliness and precise execution. Security assessments are carried out to fortify the application's defenses against potential threats, encompassing penetration testing, security scans, and code reviews to pinpoint and rectify vulnerabilities. Moreover, rigorous performance testing examines the application's responsiveness, scalability, and load-handling capabilities, guaranteeing it can withstand real-world usage patterns and meet the performance standards required for smooth HR operations, thereby securing a robust and reliable HR management system.

In summary, the Actitime HR Management application serves as a cornerstone of HR functions, and the project is dedicated to assuring that it functions flawlessly, upholds data security, and performs optimally, thereby ensuring the application aligns with the requirements of HR professionals and safeguards data integrity and confidentiality.

2 Scope

2.1 IN-SCOPE

- User Authentication:

In-scope testing involves validating the login functionality to ensure HR users can successfully log in with valid usernames and passwords. This includes checking the authentication process for accuracy, security, and user-friendliness. Confirm that the login process works as intended and securely grants access to authorized HR users.

- **Access to Employee Profiles:**

This testing verifies that HR users can access and view employee profiles within the application. It includes testing the interface, permissions, and data accuracy related to employee profiles. Ensure that HR users can access and view employee profiles with the appropriate permissions.

- **Review of Leave and Attendance Reports:**

Includes evaluating the functionality for HR to review leave and attendance reports. This involves checking the accuracy of data and the usability of the report review feature. Confirm that HR can efficiently review leave and attendance reports as required.

- **Approval/Rejection of Timesheets:**

This aspect of testing ensures that HR can perform actions like approving or rejecting timesheets within the application. It encompasses testing the user interface for these actions, data validation, and permissions. Verify that HR can efficiently approve or reject timesheets according to established workflows.

2.2 OUT-OF-SCOPE

- **User Registration:**

Out-of-scope items include testing the registration process for creating new HR user accounts. This involves the registration interface, user verification, and associated security measures. User registration testing is not within the current testing scope.

- **Integration with other systems:**

Out-of-scope testing excludes evaluating the applicability of the HR Management system to integrate with other systems, such as third-party applications or data sources. Integration testing may involve data exchange, API compatibility, and workflow integration. Testing for integration with external systems is not part of the current testing scope.

- **Performance Testing:**

Performance and load testing, which includes assessing the application's response time and scalability under various conditions, are out-of-scope for this testing effort. Performance testing involves load testing, stress testing, and benchmarking. Performance and load testing are not part of the current testing scope.

- **Compatibility Testing:**

Compatibility testing, which checks the application's compatibility with a wide range of browsers and devices, is not within the scope of this testing phase. Compatibility testing on multiple platforms is not part of the current testing scope.

3 Testing Strategy

3.1 PRODUCT/APPLICATION/SOLUTION RISKS

Risks	Criticality	Mitigation Strategy
User Authentication Issues	High	Thoroughly test the login functionality, conduct user acceptance testing (UAT), and provide clear login instructions and support for users.
Unauthorized Access	High	Implement strict access controls and permissions. Conduct thorough testing of user roles and privileges. Regularly review and update access policies.
Data Privacy and Security Breach	High	Employ encryption and secure data storage practices. Conduct security audits and penetration testing regularly. Ensure compliance with data protection regulations
Incomplete or Inaccurate Test Data	Medium	Establish a comprehensive test data strategy, including test data generation and validation. Implement data masking or anonymization where necessary.
HR functions like employee profile viewing, leave approval, and attendance report generation may not work correctly.	High	Conduct thorough functional testing and create comprehensive test cases.

Browser Compatibility. The application may not work consistently across different web browsers.	Medium	Test the application on various browsers and versions to ensure compatibility.
---	--------	--

3.2 LEVEL OF TESTING

Test Type	Description
Functional Testing	<p>Functional testing ensures that the HR Management application's functions work as specified in the requirements. This testing type will encompass verifying features like user authentication, access to employee profiles, leave and attendance report functionality, and timesheet approval/rejection.</p> <p>The objective of functional testing is to validate that the application's functions perform correctly and meet user expectations.</p>
Regression Testing	<p>Regression testing is conducted to ensure that new code changes or feature additions do not introduce new defects or break existing functionality. It involves retesting of previously validated functionality.</p> <p>The objective of regression testing is to confirm that recent changes have not negatively impacted the application's existing features.</p>
Security Testing	<p>Security testing focuses on identifying vulnerabilities and weaknesses in the application's security. It includes activities like penetration testing, vulnerability scanning, and access control testing to ensure data privacy and protect against unauthorized access.</p> <p>The objective of security testing is to ensure that the application is secure and can withstand potential security threats.</p>

Usability Testing	<p>Usability testing evaluates the application's user-friendliness. It assesses how easy it is for HR users to navigate the interface, perform tasks, and accomplish their objectives within the application.</p> <p>The objective of usability testing is to ensure that the application provides an intuitive and efficient user experience.</p>
User Acceptance Testing (UAT)	<p>UAT involves HR users testing the application to determine if it meets their specific needs and requirements. It is typically the final stage of testing before deployment.</p> <p>The objective of UAT is to validate that the application meets HR user expectations and is ready for production use.</p>

4. Test Approach

4.1 TEST DESIGN APPROACH



Test Design Strategy:

Our test design strategy for the HR Management application is designed to ensure comprehensive test coverage while efficiently utilizing available resources. To achieve this, we will employ various test design techniques that align with our testing objectives. The following test design techniques will be used, and here's how they will be applied:



1. **Equivalence Partitioning:**
Equivalence partitioning will be applied to user authentication testing, specifically to validate the login functionality. We will divide the input space into equivalence classes such as valid usernames, valid passwords, and invalid inputs to ensure that each class is adequately tested.
2. **Boundary Value Analysis:**
Boundary value analysis will also be applied to user authentication testing. It helps in identifying potential issues related to the boundary conditions of input values. This is crucial for checking the application's response to the edge cases of valid and invalid inputs.
3. **Use Case-Based Testing:**
Use case-based testing will be used for access to employee profiles, review of leave and attendance reports, and approval/rejection of timesheets. We will design test cases based on typical usage scenarios and user interactions to ensure that critical user journeys are thoroughly tested.
4. **Exploratory Testing**
Exploratory testing will be used to uncover unexpected defects and issues that may not be covered by predefined test cases. Testers will explore the application, attempting various actions and operations, simulating real-world user behavior.
5. **State Transition Testing:**
State transition testing will be applied to scenarios where the application's behavior changes based on specific conditions, such as leave approval/rejection workflows. We will model and test these state transitions thoroughly.

4.2 EXECUTION STRATEGY

4.3.1 Entry Criteria

Entry Criteria	Conditions	Comments
<i>Test environment(s) is available</i>		
<i>Test data is available</i>		
<i>Code has been merged successfully</i>		
<i>Development has completed unit testing</i>		
<i>Test cases and scripts are completed, reviewed and approved by the Project Team</i>		
<i>Test Team Readiness</i>		
<i>Test Tools and Resources</i>		

3.2.2 Exit criteria

Exit Criteria	Conditions	Comments
<i>100% Test Scripts executed</i>		
<i>90% pass rate of Test Scripts</i>		
<i>No open Critical and High severity defects</i>		
<i>All remaining defects are either cancelled or documented as Change Requests for a future release</i>		
<i>All expected and actual results are captured and documented with the test script</i>		
<i>All test metrics collected based on reports from daily and Weekly Status reports</i>		
<i>All defects logged in -Defect Tracker/Spreadsheet</i>		
<i>Test environment cleanup completed and a new back up of the environment</i>		
<i>Documentation Updated</i>		

3.3 DEFECT MANAGEMENT

Defects found during the Testing should be categorized as below:

Severity	Impact
1 (Critical)	<ul style="list-style-type: none">▪ <i>Functionality is blocked and no testing can proceed</i>▪ <i>Application/program/feature is unusable in the current state</i>
2 (High)	<ul style="list-style-type: none">▪ <i>Functionality is not usable and there is no workaround but testing can proceed</i>
3 (Medium)	<ul style="list-style-type: none">▪ <i>Functionality issues but there is a workaround for achieving the desired functionality</i>
4 (Low)	<ul style="list-style-type: none">▪ <i>Unclear error message or cosmetic error which has minimum impact on product use.</i>

5. Test Team Structure

5.1 TEAM STRUCTURE

Team - Actitime QA Bots

#	Role	Resource Count
1	QA Manager	1
2	QA Leads	1
3	Senior QA Engineers	2
4	QA Engineers	3

5.2 ROLES AND RESPONSIBILITIES

QA Manager

Role: The QA Manager is responsible for the overall management and coordination of the testing effort. They provide strategic direction, leadership, and oversight to the entire testing team.

Responsibilities:

- Develop the test strategy and test plan.
- Allocate resources and define project timelines.
- Manage communication with stakeholders.
- Ensure adherence to quality standards and best practices.
- Oversee risk management and mitigation.

QA Leads

Role: QA Leads assist the QA Manager in managing specific aspects of the testing process, including test design, test execution, and defect management.

Responsibilities:

- Develop detailed test plans and test cases.
- Coordinate testing activities within their assigned area.
- Mentor and guide junior team members.
- Assist in risk assessment and mitigation.
- Ensure that testing objectives are met.

Senior QA Engineers

Role: Senior QA Engineers are experienced testers responsible for the detailed planning and execution of tests, as well as providing valuable insights to the team.

Responsibilities:

- Develop test cases, test scripts, and test data.
- Execute test cases and document results.
- Conduct complex test scenarios, including performance and security testing.
- Assist in identifying and analyzing defects.
- Contribute to continuous improvement of testing processes.

QA Engineers

Role: QA Engineers are responsible for hands-on testing activities, including executing test cases, verifying defects, and ensuring the quality of the HR Management application.

Responsibilities:

- Execute test cases and report test results.
- Verify and document defects, including providing detailed steps to reproduce.
- Participate in regression testing and re-testing efforts.
- Collaborate with the team to ensure the application meets quality standards.
- Provide feedback to improve test cases and test data.

6. Test Schedule

1. Test Phase Start Date: 01/11/2023

2. Test Phase End Date: 21/04/2024

3. Test Activities:

a. Test Planning and Preparation:

- Start Date: 01/11/2023

- End Date: 15/11/2023

- Description: This initial phase involves developing the test strategy, test plan, and defining the test scope and objectives. It includes resource allocation, tool selection, and environment setup.

b. Test Design:

- Start Date: 15/11/2023

- End Date: 15/12/2023

- Description: During this phase, detailed test cases, test scripts, and test data are designed based on the requirements and the testing strategy.

c. Test Environment Setup:

- Start Date: 15/12/2023

- End Date: 18/12/2023

- Description: This involves configuring the test environment, including hardware, software, databases, and network configurations to mirror the production environment as closely as possible.

d. Test Execution:

- Start Date: 18/12/2023

- End Date: 15/01/2024

- Description: The actual execution of test cases and scenarios takes place during this phase, encompassing functional, security, performance, usability, and integration tests.

e. Defect Management:

- Start Date: 15/01/2024

- End Date: 15/02/2024

- Description: Throughout the testing phase, defects are identified, reported, and managed. Defects are resolved and re-tested in this stage.

f. Regression Testing:

- Start Date: 15/02/2024

- End Date: 01/03/2024

- Description: After defect resolution, regression testing is conducted to ensure that existing functionality remains intact.

g. User Acceptance Testing (UAT):

- Start Date: 01/03/2024
- End Date: 31/03/2024
- Description: HR users will perform UAT to validate that the application meets their specific needs and requirements.

4. Milestones:

a. Test Plan Approval:

- Date: 15/11/2023
- Description: Formal approval of the test plan by relevant stakeholders.

b. Mid-Test Phase Review:

- Date: 25/12/2023
- Description: A review meeting to assess the progress of testing activities and address any issues or concerns.

c. End of Test Execution:

- Date: 15/01/2024
- Description: Completion of test case execution.

d. Defect Resolution Completion:

- Date: 15/02/2024
- Description: All defects are resolved, verified, and closed.

e. UAT Completion:

- Date: 31/03/2024
- Description: User Acceptance Testing (UAT) is successfully completed.

5. Dependencies:

-

6. Resource Allocation:

- Team members
- Repository access
- System access
- Database access

7. Contingency Plan:

- 3 weeks is allocated in the schedule.

7. Test Reporting

7.1. TEST REPORTING APPROACH

#	Report Name	Owner	Audience	Frequency
1	Test Summary Report	Project Manager	Project Stakeholders, QA Manager	Weekly
2	Defect Report	QA Manager	Development Team, QA team	Daily, followed by weekly summaries
3	Security Testing Report	Senior QA Engineer	Security Team, QA team	Once every two weeks, after each security testing iteration

7.2. QUALITY MATRICES

1. Test Coverage Matrix

The test coverage matrix measures the percentage of requirements, functionalities, or code paths that have been tested. It helps ensure that all aspects of the application have been adequately covered by test cases.

Measurement: Calculated as the ratio of tested items to the total number of items (e.g., requirements or code lines). Formula: $(\text{Tested Items} / \text{Total Items}) * 100\%$.

2. Defect Density Matrix

The defect density matrix quantifies the number of defects identified in a specific module, feature, or component, providing insights into the quality and stability of each area.

Measurement: Calculated as the ratio of defects to the size of the module, feature, or component. Formula: $(\text{Defect Count} / \text{Module Size})$.

3. Test Execution Progress Matrix

This matrix tracks the progress of test execution over time, reflecting the percentage of

completed test cases and the remaining work.

Measurement: Monitored weekly or at regular intervals, showing the percentage of test cases executed and their status (e.g., Pass/Fail/Blocked).

4. Test Effectiveness Matrix

The test effectiveness matrix assesses the ability of test cases to detect defects. It indicates how many defects were discovered by the tests and their distribution by severity.

Measurement: Percentage of defects identified by the test cases out of the total defects discovered.

5. Test Case Pass Rate Matrix

The test case pass rate matrix measures the percentage of test cases that have passed successfully, indicating the stability and reliability of the application.

Measurement: Calculated as the ratio of passed test cases to the total number of executed test cases. Formula: $(\text{Passed Test Cases} / \text{Total Executed Test Cases}) * 100\%$.

6. Test Completion Matrix

This matrix tracks the completeness of testing efforts by measuring the percentage of planned test cases that have been executed, allowing project managers to assess the overall testing progress.

Measurement: Calculated as the ratio of executed test cases to the total planned test cases. Formula: $(\text{Executed Test Cases} / \text{Total Planned Test Cases}) * 100\%$.

7. Defect Closure Rate Matrix

The defect closure rate matrix evaluates the rate at which identified defects are resolved and closed. It provides insights into the efficiency of the defect management process.

Measurement: Calculated as the ratio of closed defects to the total identified defects. Formula: $(\text{Closed Defects} / \text{Total Identified Defects}) * 100\%$.

8. Test Cycle Time Matrix

Test cycle time measures the time taken to complete a testing cycle or regression cycle. It helps evaluate the efficiency of testing activities and identify potential bottlenecks.

Measurement: Calculated as the time taken to complete a testing cycle.

9. User Acceptance Testing (UAT) Satisfaction Matrix

The UAT satisfaction matrix assesses the satisfaction level of HR users during user acceptance testing, providing insights into the application's user-friendliness and alignment with user needs.

Measurement: Gathered through user surveys or feedback forms to evaluate user satisfaction and identify areas for improvement.

8. Test Environment Requirements

1. Hardware and Software Requirements

- **Server Hardware:** A dedicated server or servers with the necessary processing power, memory, and storage capacity to host the HR Management application and its associated databases.
- **Operating Systems:** Supported server and client operating systems, such as Windows Server, Linux, and various versions of Windows for client devices.
- **Database:** A suitable database system for storing application data
- **Web Browsers:** A range of web browsers, such as Chrome, Firefox, Safari, and Edge, to verify cross-browser compatibility.
- **Network Infrastructure:** A stable and secure network environment, with adequate bandwidth, to simulate real-world network conditions

2. Test Data Requirements

- **Sample Data:** Sample employee profiles, leave records, and timesheets representative of real-world usage to validate application functionality.
- **Sensitive Data:** Anonymized sensitive data for security testing, ensuring that sensitive information is adequately protected during access and transmission.

3. Testing Tools and Software

- **Test Management Tool:** A dedicated test management tool Jira, to manage test cases, defects, and reporting.
- **Automation Tools:** Automated testing tools such as Selenium, JMeter
- **Browser Developer Tools:** Web browser developer tools to inspect and debug web applications during testing.

4. Test Environment Configuration

- Isolation: Ensuring that the test environment is isolated from the production environment to prevent data leaks or interference with live operations.
- Data Backup and Restore: Regular backups of the test environment data to facilitate recovery and restoration in case of data corruption or testing failures.

5. Access and Permissions

- User Accounts: User accounts with different roles and permissions for testing various scenarios.
- Admin Access: Administrative access to configure and manage the application settings.

6. Monitoring and Reporting

- Logging and Monitoring Tools: Logging and monitoring tools to track system and application behaviour, performance, and security.
- Reporting Tools: Tools for generating reports on test execution, defect status, and overall test progress.

9. Dependencies and Assumptions

Dependencies

- Test Data Availability: Testing relies on the availability of test data, including employee profiles, leave records, and timesheets.
- Test Environment Readiness: The test environment, including the hardware, software, and network infrastructure, must be configured and ready for use before testing.
- Resource Allocation: The availability of testing resources, including QA team members, testing tools, and hardware, is a critical dependency. Any resource constraints can affect the testing process.

Assumptions:

- Data Integrity: The accuracy and integrity of test data, especially sensitive HR-related data, are assumed. It is expected that the data provided for testing reflects real-world scenarios and is suitable for validation.
- Availability of Test Environments: It is assumed that the test environments will be available and properly configured as per the test environment requirements.
- Timely Issue Resolution: The prompt resolution of identified defects and issues is assumed.