
Quality Assurance Plan

HR Management Application

Assignment II

Advance Software Quality Assurance

19020882 – K.D.Y.K. Weerasinghe

1 Introduction

1.1 PURPOSE

This document outlines the testing strategy and approach for validating the HR Management features of actiTime application to ensure its functionality, performance, and security.

1.2 PROJECT OVERVIEW

The project detailed in this test plan is, ActiTime. It is a software tool that provides time-tracking/timesheet and management solutions for businesses and organizations of various types and sizes. Users can manage leaves, maintain records of the duration of time spent on different projects and activities, and create analytical reports to gain a deeper knowledge and analysis of productivity and resource use. The platform is intended to improve overall efficiency in teams and organizations by streamlining project management procedures.

2 Scope

2.1 IN-SCOPE

1. **Login Functionality** - Verify that HR users can log in with valid credentials.
2. **View Employee Profile** - Ensure HR personnel can view employee profiles.
3. **Review leaves and attendance reports** - Validate HR personnel can access and review leave and attendance reports.
4. **Approve/Reject timesheets** - Validate HR personnel can review and approve/reject timesheets submitted by employees

2.2 OUT-OF-SCOPE

1. Customer and project management functionality
2. Creation, viewing and management of tasks.
3. Create new user functionality.
4. Work assignments functionality.
5. Lock time tracking feature.
6. Financial report generation.
7. Employee work related report generation excluding leave and attendance reports.

3 Testing Strategy

3.1 PRODUCT/APPLICATION/SOLUTION RISKS

Risks	Criticality	Mitigation Strategy
Unauthorized access to the system without proper credentials.	High	<ul style="list-style-type: none">○ Implement mechanisms such as two-factor authentication for secure logins.○ Perform frequent security testing to identify vulnerabilities and implement security patches.

		<ul style="list-style-type: none"> ○ Identify and address any loopholes in the login process.
Data breach. Unauthorized access to sensitive information.	High	<ul style="list-style-type: none"> ○ Encrypt employee data and sensitive information to prevent unauthorized access or theft. ○ Implement strict access controls to guarantee that sensitive data is only accessed by authorized personnel.
Unauthorized alterations to system data.	High	<ul style="list-style-type: none"> ○ Implement strict access controls and permission settings to restrict unauthorized alterations. ○ Create a multi-level approval procedure to guarantee that any adjustments made to data such as timesheet are legitimate. ○ Conduct regular audits to review the integrity of data.
Poor performance of system under heavy usage	High	<ul style="list-style-type: none"> ○ Perform load tests and enhance system efficiency to manage increased user engagement and data processing requirements. ○ To mitigate potential performance limits, use effective caching mechanisms and scalable hardware.
Failure to comply with regulatory standards and legal requirements	High	<ul style="list-style-type: none"> ○ Update company policies on a regular basis to guarantee compliance with regulations. ○ Conduct regular compliance audits and inspections to detect and correct violations.
Service disruption due to system downtime or outages	High	<ul style="list-style-type: none"> ○ Set up backup servers and other solutions to provide continuous access to leave and attendance records. ○ Create a comprehensive disaster recovery strategy to reduce downtime. ○ Perform regular system maintenance and assessments to discover and rectify any potential issues.
Loss of data due to system failures	High	<ul style="list-style-type: none"> ○ Back up employee profiles on a regular basis to avoid data loss in the case of a system breakdown.

		<ul style="list-style-type: none"> ○ Implement effective data recovery procedures to restore any lost or damaged employee data.
Data inconsistencies due to User errors in data entry	Low	<ul style="list-style-type: none"> ○ Implement user-friendly interfaces to ease data entry. ○ Implement automated data validation checks to ensure accuracy and consistency ○ Provide thorough user training on data entry best practices.
Integration Issues (Incompatible browsers/devices)	Medium	<ul style="list-style-type: none"> ○ Perform comprehensive compatibility tests and establish effective communication channels with third-party vendors for smooth integration.

3.2 LEVEL OF TESTING

Test Type	Description
Functional Testing	Aims to validate that the software functions align with specified requirements, examining inputs and outputs to confirm their expected behavior.
Regression Testing	Involves retesting the software to ensure no adverse effect to the functionality of existing features, after alterations have been made.
Non-Functional Testing	Evaluates performance, usability, reliability, and other non-functional aspects of the software to ensure it meets established quality standards.

3.2.1 FUNCTIONAL TESTING

Test Type	Description
Unit Testing	Individual components or modules are tested to ensure they adhere to specified requirements and expected functionality.
Integration Testing	Examines the interaction and cooperation between software modules or components to validate seamless integration without compromising overall system functionality.
System Testing	Evaluates the complete software system to ensure that it is operating in line with the specified requirements and standards.

User Acceptance Testing (UAT)	Involves testing the software from the end user's perspective to verify if it meets the user requirements and works as intended. UAT ensures that the software is ready for deployment.
Smoke Testing	Conducted as a preliminary assessment to verify critical software functionalities before comprehensive testing, enabling early identification of major issues.

3.2.2 REGRESSION TESTING

Test Type	Description
Progressive Regression testing	Involves creating new test cases for modified specifications, facilitating independent execution within the updated program.
Selective Regression Testing	Targeted testing using a subset of test cases to minimize retesting effort and cost when incorporating new code.
Retest-all regression testing	Reevaluation of the entire system to ensure the functionality remains intact after any modifications or fixes.
Corrective Regression testing	Reuses existing test cases for unchanged specifications, efficiently identifying faults or bugs.

3.3.3 NON-FUNCTIONAL TESTING

Test Type	Description
Performance testing	The software's performance under various conditions, such as workload, response time, and stability, is evaluated.
Usability Testing	Assesses the software's user-friendliness, intuitiveness, and overall user experience.
Security Testing	Identifies software vulnerabilities to safeguard data and resources from potential threats and unauthorized access.

4. Test Approach

4.1 TEST DESIGN APPROACH

A comprehensive and multifaceted testing approach will be undertaken for the HR process requirements ensuring that all aspects of the application are thoroughly examined and validated. A combination of analytical and directed test methodologies alongside several test design techniques will be utilized to achieve maximum test coverage and efficiency.





The following test design techniques will be used,

1. **Equivalence Partitioning**
E.g., Verify the accuracy of data presentation by testing the access to leave and attendance reports for various employee categories, such as those on leave or on duty.
2. **Boundary Value Analysis**
E.g., Test the system's response when there are no timesheets to approve or reject.
3. **Decision Table Testing**
E.g., Create a decision table with valid and invalid login credentials to verify all possible combinations and their respective outcomes
4. **State Transition Testing**
E.g., Verify the transition from the timesheet review to the final approval/rejection confirmation.
5. **Exploratory Testing**
E.g., Verify the user experience is intuitive and error messages are user-friendly.

4.2 EXECUTION STRATEGY

4.3.1 ENTRY CRITERIA

- The entry criteria refer to the desirable conditions in order to start test execution
- Entry criteria are flexible benchmarks. If they are not met, the test team will assess the risk, identify mitigation actions, and provide a recommendation.

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

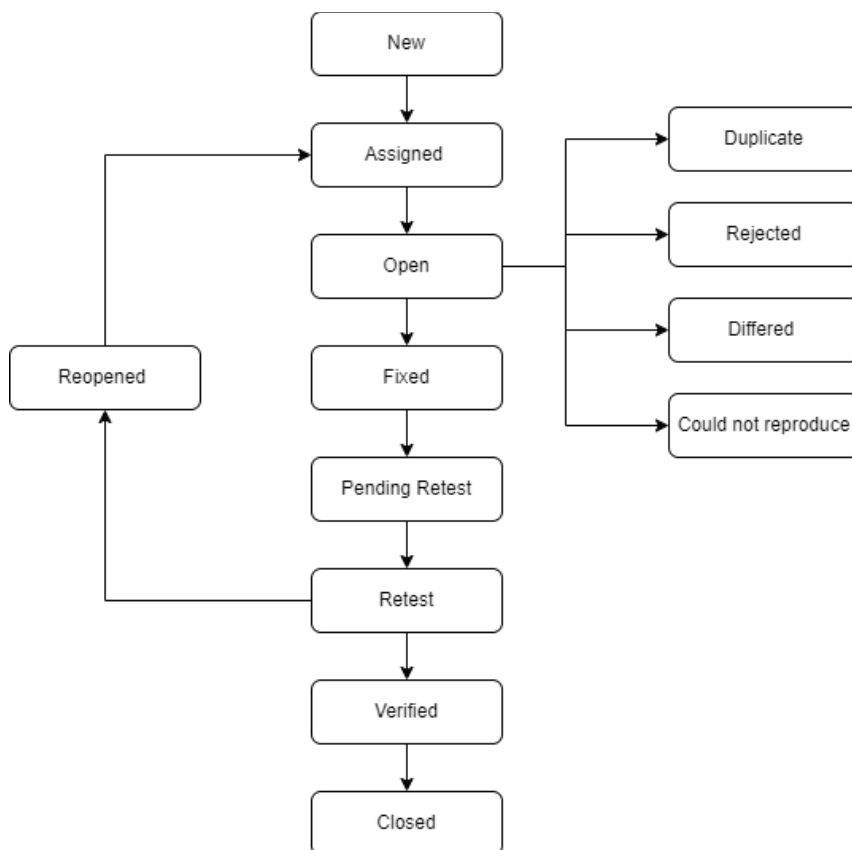
4.3.2 EXIT CRITERIA

- The exit criteria are the desirable conditions that need to be met in order proceed with the implementation.
- Exit criteria are flexible benchmarks. If they are not met, the test team will assess the risk, identify mitigation actions and provide a recommendation.

Exit Criteria	Conditions	Comments
---------------	------------	----------

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

3.3 DEFECT MANAGEMENT



- It is expected that the testers execute all the scripts in each of the cycles described above.

- The defects will be tracked through Defect Tracker or Spreadsheet.
- It is the tester's responsibility to open the defects, retest and close them.

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

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

5.2 ROLES AND RESPONSIBILITIES

QA Manager

- Develop a comprehensive testing strategy including goals, resources, and scope.
- Allocate resources and team members to appropriate testing domains and organize team testing efforts.
- Ensure that the testing process aligns with quality standards, industry best practices, and company policies.
- Facilitate effective communication among stakeholders and testing levels.

QA Leads

- Oversee the development of test cases and ensure they meet the requirements.
- Coordinate and monitor the execution of test cases in accordance with the plan.
- Inform the QA Manager on progress, issues, and any deviations from the test plan.

Senior QA Engineers

- Create detailed test cases based on requirements, covering various scenarios.
- Implement test cases, record and validate detected flaws, and resolve them.
- Create and maintain thorough test documentation, test scripts and test results.

QA Engineers

- Implement test cases, record and validate detected flaws, and resolve them.
- Assist with the preparation and maintenance of test data for test cases.
- Create and maintain clear and accurate test documentation, including test execution records.

6. Test Schedule

Test Activity	Week																	
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
Test Planning																		
Test Environment Setup																		
Test Case Design																		
Test Data Preparation																		
Test Execution																		
Defect Reporting and Tracking																		
Regression Testing Phase																		
User Acceptance Testing (UAT)																		
Final Testing and Sign-off Phase																		

7. Test Reporting

7.1. TEST REPORTING APPROACH

#	Report Name	Owner	Audience	Frequency
1	TEST PROGRESS REPORT			
2	TEST PROGRESS REPORT			

7.2. QUALITY MATRICES

- Defect density quantifies the number of defects per unit of code, indicating code quality and defect detection effectiveness.
- Test coverage metrics assess the scope of testing to ensure complete coverage across crucial areas.
- Test pass/fail rates indicate overall test success and areas needing improvement.
- The percentage of automated test cases, which shows the proportion of all test cases that are automated.
- Test Case Effectiveness evaluates the effectiveness of test cases in defect detection by taking into account criteria such as the proportion of problems identified and the average time required.

8. Test Environment Requirements

Test Environment Requirements include specific conditions, hardware, software, and configurations necessary for accurate and reliable testing that closely mimics the production environment.

1. **Hardware Requirements:** Machines with sufficient processing power and RAM, matching the specifications of the target users' systems, to effectively handle the application's workload.
2. **Software requirements:** For cross-platform compatibility testing, use operating systems like Windows, macOS, and Linux, evaluate the application's performance across various browsers such as Chrome, Firefox, and Edge, and manage and manipulate test data effectively using Database Systems like MySQL, PostgreSQL, or equivalents.
3. **Network Requirements:** Consistent internet connectivity to assess the application's functionality under real-time network situations.
4. **Testing tools:** Utilize Jira, TestRail, or similar Test Management Tools for comprehensive test case and result management, along with automated testing tools like Selenium, or similar options, for automated test case execution and regression testing, and employ Jira, Bugzilla, or similar Defect Tracking Tools for efficient defect monitoring during testing.
5. **Data Management:** Utilize Backup and Restore Tools to safeguard and restore test data in the event of any data loss during testing, and ensure adherence to data protection regulations and guidelines for maintaining data confidentiality in the test environment.

9. Dependencies and Assumptions

Dependencies

1. Test item availability
2. Availability of required resources
3. Availability of a stable testing environment

Assumptions

1. Assumed that unit testing, code integration, and system development have been concluded, and the test data will be established using the demo data available in the actiTIME platform.
2. Assumed that essential resources, including test management tools and testing staff, are allocated aligning with the project plan.
3. Assumed that the setup and configuration of the test environment, including hardware, software, and network infrastructure, will be timely.
4. Assumed that sufficient data privacy and security protocols are implemented to safeguard sensitive HR data throughout the testing process.