Test Plan
For
Admin 360

Prepared By:	Alex Vaghela	Reviewed by:	
Version No:	V1.1	Issue Date:	19/11/2024

Test Plan – Admin 360	

Document Revision History*

Document Version	Submitted By	Date	Reviewed & Approved by	Notes
V1.0	Alex Vaghela	20/11/2024		
V1.1	Alex Vaghela	26/11/2024		

Test Plan – Admin 360	

Table of Content

Test Plan – Admin 360

1. INTRODUCTION

1.1 Purpose

Our goal is to make sure the admin 360 system works perfectly and meets all the requirements. To achieve this, we will conduct thorough testing of all key features, including the Login module, Dashboard module, Guest House system, and Remote Office Attendance system. The primary goal of this testing phase is to validate that these components function as expected, perform optimally, and offer a seamless user experience. By doing so, we can ensure that the admin 360 platform meets its requirements and serves as a reliable tool for administrators.

1.2 About This Document

This document outlines the test plan for the admin 360 system, a comprehensive platform designed to support administrative tasks. The primary objective of this test plan is to ensure that the system meets all functional, performance, and usability requirements. The key components of this document are Login module, Dashboard, Guest house System, Remote attendance system.

1.3 Related Documentation

- Software Requirements Specification
- End–Client requirements
- Test Summary Report

2 Scope

2.1 Application Scope

The scope of the admin 360 application testing includes

- Login
- Dashboard
  - o Guest House System
  - o Remote Office Attendance System

2.2 Test Scope

The testing scope are going to cover in on this module

- Functional testing of Login, Dashboard, Guest House System, Remote office attendance system.
- Security testing to once we find these weaknesses, we fix them to make the software more secure.
- Regression testing to test the software after making changes to ensure everything still works as expected.
- Defect management to issues found in the software, assigning them to the right people to fix, and tracking their progress until they're resolved.
- Performance Testing: Excluding comprehensive performance testing under various conditions.

1

- Useability testing to navigate, and overall, a pleasant experience.

2.3 Exclusions to the Scope

The following items are not included in the scope of this test plan

- Third-Party Integration Testing: Excluding integration tests with third-party applications or services.
- Automation Testing: Test the application by hand, not using automated tools.
- Database Performance Optimization: Not including specific performance optimization tasks for the database.

3 Approach
3.1 Traditional Testing

This is a step-by-step approach where testing happens alongside development.

- Requirement Analysis: Understanding the requirements.
- Test Planning: Deciding how the software will be going to test.
- Test cases: Create different test to check different parts of the system.
- Test Execution: Execute the cases and see the actual results.
- Test closure: Verify all test cases is complete and the system is ready for delivery.

3.2 Functional Testing

This approach is focuses on testing the system functionality to ensure it works as required.

- Functional requirements: Verify each part of the system it works as par the requirements.
- Test Execution: Verify the all-test cases positive and negative scenarios.
- Cover: Make sure every single feature of system is tested properly.

3.3 Integration Testing

Using integration testing approach have to check all parts of system work together or not.

- Module Integration: Test the different functions of system are work together.
- Interface Test: Check the data is transferred between different parts of system.
- End-to-end testing: Test the complete workflow of system for end uses.

3.4 GUI Testing

The GUI testing's role is testing the end user side and verify the elements and visuals of system.

- Usability Testing: Ensuring the interface is user-friendly and intuitive.
- Visual Consistency: Verify the elements are consistent in the system.
- Functionality Testing: Make sure all the clickable functions of system are work perfectly.

3.5 Browser Compatibility Testing

Browser Compatibility Testing verify the System works correctly on different web browsers.

- Browser Testing: Test the system on different browsers like Chrome, Firefox, Safari, Edge, etc.

- Functionality and Visual Checks: Verify the system functionality and visual elements are proper on all browsers.
- Responsive Interface: Check the system on different screen size and resolution like mobile, tablet etc.

3.6 Application Testing on Mobile Devices

Test the application on mobile devices, its properly works on mobile devices.

- Device Compatibility: Testing the system on various mobile devices and operating systems like iOS, Android, windows etc.
- Responsive Interface: Check the application's layout and elements proper on different screen sizes.

3.7 Security Testing

The main role of security testing is checking the application is protected for threads and hackers.

- Vulnerability Scanning: Verify the security weaknesses within the System.
- Access Control: Ensuring that only authorized application users have access to specific data.
- Data Encryption: Verifying that sensitive data is encrypted to keep it safe.
- Security Compliance: Make sure the software meets all the necessary security standards.

3.8 Performance Testing

Performance Testing check the application's responsiveness and stability under various conditions.

- Load Testing: The application can work smoothly even when many people are using it at the same time.
- Stress Testing: The application can handle unexpected spikes in usage or other stressful conditions.
- Response Time: The application responds quickly to user actions.
- Scalability Testing: The application can handle increasing user loads efficiently.

3.9 USER ACCEPTANCE TESTING

User Acceptance Testing (UAT) verify that the application meets the end-user requirements for deployment.

- Requirements Validation: Verifying that the System meets the business requirements.
- End-User Scenarios: Test the System in real-life situations to see how it performs.
- Feedback Collection: Take feedback and opinions from users for identifying problems for improvement.

4 Test Environment

4.1 QA/Staging Environment

The QA (Quality Assurance) environment as a space where we can test the software before it goes live. It's like a practice field where we can try out different things and make sure everything works as expected.

- Configuration: Create one set up the QA environment to be as close as possible to the actual environment where the software will be used.
- Data Setup: Use special test data to simulate real-world scenarios without affecting real data.
- Tools and Resources: For perform different types of testing have all necessary tools to get equate results.
- Stability: Ensures the environment is stable and consistent, allowing tester to run repeated tests without any interruptions.

4.2 UAT Environment

The UAT (User Acceptance Testing) Environment is configured to allow end-users to validate the application against their requirements and real-world scenarios.

- Production-Like Setup: It's sat-up like close as possible to the actual production environment.
- Real-world data: Use real-world data or scenarios to test the application.
- User Access: Designed for end-users to test the software and provide feedback.
- Acceptance Criteria: Add specific acceptance criteria that must be met for the application to be approved for production.

5 Defect Tracking Mechanism

The defect tracking process to make sure Admin 360 system works correctly, we need a way to find and fix any problems. This section explains how we do that, including the steps we take and the tools we use to identify, record, manage, and solve these issues.

For Defect tracking we use mostly **JIRA** tool.

JIRA: A comprehensive issue and project tracking tool that enables detailed defect management, such as logging, prioritizing, and resolving.

Flow of Defect Tracking:

I. **New:** A bug is found by tester and reported by tester.

II. **Assigned:** After reporting defect or bug tester give this to developer or team for fix bug.

III. **Open:** When developer open the bug and start process to fix, is show status of bug. Now it's "open"

IV. **Fixed:** Once developer done with the fixing bug then it changes status "Fixed" for bug.

V. **Retest:** After fixing of bug tester do retest on this fix to check it is work properly or not.

VI. **Reopened:** When tester test the fix or bug but it's doesn't not work as expected then tester send back to developer and make status of bug "Reopened".

VII.   **Verified:** Once tester confirm the fix is works properly, then tester mark as "Verified".

VIII.   **Closed:** When bug is fixed and it's verified by the tester and it's officially declared by tester then tester marks "Closed".

6   Test Deliverables

These are the documents we will create during the testing process for Admin 360.

I.   Test Plan
II.   Test Data
III.   Test Scenario
IV.   Test Cases
V.   Defect Report
VI.   Test Summary report
VII.   User Acceptance Testing (UAT) Report
VIII.   Test Closure Report

7   Assumptions

These are the things we are assuming to be true for our testing process:

- Stable Test Environment: The test set-up will be same and stable throughout the testing process.
- Availability of Test Data: All necessary test data will be available and correctly configured before testing begins.
- Requirements Clarity: All requirements and specifications for the Admin 360 system are clear, complete, and approved.
- Timely Fixes: Defects identified during testing will be resolved promptly by the development team.
- User Involvement: Users will be available to test the software and give feedback.

8   Risk and Mitigation

I.   **Risk:** Delays in set-up the test environment
     Mitigation: Make a detailed setup checklist to ensure everything is ready on time, use online tools to save time and do fast work

II.   **Risk:** Unclear or incomplete requirements
     Mitigation: Do regular meetings with the team to clear up any confusion and confirm them with the team.

III.   **Risk:** Problems with logging in
     Mitigation: Test different login scenarios, like entering wrong and right username or password, test the login system early to catch problems quickly.

IV. **Risk:** Issues on different devices or browsers
Mitigation: Test the application on different browsers like Chrome, Safari and Edge, and also check on devices like desktops, mobiles, and tablets.

V. **Risk:** If employee on leave during testing
Mitigation: When one employee on leave during the testing then handover his task to another team member for complete the task.

6