Comprehensive Test Plan Document

Part 1: Test Plan

1. Introduction

Purpose of the Test Plan: The purpose of this test plan is to define the strategy, scope, approach, and deliverables for testing the web application. It aims to ensure that the application meets the required standards and functions correctly before release.

Scope of Testing: This test plan covers:

Functional Testing: Verifying that the application performs its intended functions correctly.

Usability Testing: Ensuring that the application is user-friendly.

Performance Testing: Assessing the application's responsiveness and stability under load.

Security Testing: Identifying vulnerabilities and ensuring the application is secure.

Overview of the Application Under Test: The application is a web-based platform for managing locked messages. Users can create, edit, delete, and view locked messages. The application includes features for user authentication, message management, and administrative functions.

2. Test Strategy

Types of Testing to be Performed:

Functional Testing: Verify that each feature of the application works as expected.

Usability Testing: Assess the user interface and experience.

Performance Testing: Test the application's response times and load handling.

Security Testing: Identify potential security vulnerabilities and ensure data protection.

Levels of Testing:

Unit Testing: Validate individual components or functions.

Integration Testing: Check the interaction between integrated components or systems.

System Testing: Ensure that the entire application works as intended in its entirety.

Acceptance Testing: Confirm that the application meets business requirements and is ready for release.

3. Test Criteria

Entry and Exit Criteria:

Unit Testing:

Entry Criteria: Code must be complete and unit test cases must be prepared.

Exit Criteria: All unit test cases must pass and code coverage should meet the specified threshold.

Integration Testing:

Entry Criteria: Components must be integrated and unit testing must be complete.

Exit Criteria: All integration test cases must pass and all identified integration issues must be resolved.

System Testing:

Entry Criteria: Integrated application build must be available and all integration test cases must pass.

Exit Criteria: All system test cases must pass and system stability should be confirmed.

Acceptance Testing:

Entry Criteria: System testing must be complete and all critical issues must be resolved.

Exit Criteria: Acceptance criteria must be met as per business requirements, and user acceptance testing must be signed off.

Pass/Fail Criteria for Test Cases:

Pass: The actual result matches the expected result for a test case.

Fail: The actual result does not match the expected result or errors occur during execution.

4. Test Deliverables

Test Plan Document: This document outlining the testing strategy and approach.

Test Case Document: Detailed test cases for different scenarios.

Test Execution Report: Report documenting the results of test executions.

Bug Report: Detailed reports of any defects discovered during testing.

Automation Scripts: Scripts for automated testing.

Automation Execution Report: Results from the execution of automated tests.

5. Schedule

Milestone	Deadline
Test Plan Document Completed	YYYY-MM-DD
Test Case Design Completed	YYYY-MM-DD
Test Execution Start	YYYY-MM-DD
Test Execution End	YYYY-MM-DD
Bug Report Submission	YYYY-MM-DD
Automation Scripts Completed	YYYY-MM-DD
Automation Execution Report	YYYY-MM-DD

6. Risks and Mitigations

Potential Risks and Mitigation Strategies:

Risk: Delays in development could impact testing timelines.

Mitigation: Regular communication with the development team to track progress and adjust testing schedules as needed.

Risk: Incomplete or unclear requirements may affect test coverage.

Mitigation: Engage with stakeholders to clarify requirements and ensure complete understanding before starting testing.

Risk: High volume of defects could lead to delays in release.

Mitigation: Prioritize defects based on severity and impact, and allocate resources accordingly to address critical issues first.

Part 2: Test Cases and Execution Report

1. Test Case Design

Test Case 1: Creating a New Locked Message

Test Case ID: TC-001

Description: Verify that a user can create a new locked message.

Preconditions: User must be logged in.

Steps to Execute:

Navigate to the "Create Message" page.

Enter the message content.

Click "Save".

Expected Result: The new locked message is created and appears in the message list.

Actual Result: To be filled after execution.

Status: To be filled after execution.

Test Case 2: Editing an Existing Locked Message

Test Case ID: TC-002

Description: Verify that a user can edit an existing locked message.

Preconditions: User must be logged in and have at least one message created.

Steps to Execute:

Navigate to the message list.

Select a message to edit.

Update the message content.

Click "Save".

Expected Result: The edited message is updated and the changes are reflected in the message list.

Actual Result: To be filled after execution.

Status: To be filled after execution.

Test Case 3: Deleting a Locked Message

Test Case ID: TC-003

Description: Verify that a user can delete a locked message.

Preconditions: User must be logged in and have at least one message created.

Steps to Execute:

Navigate to the message list.

Select a message to delete.

Click "Delete".

Confirm deletion.

Expected Result: The message is removed from the message list.

Actual Result: To be filled after execution.

Status: To be filled after execution.

Test Case 4: Viewing the List of Locked Messages

Test Case ID: TC-004

Description: Verify that the user can view the list of locked messages.

Preconditions: User must be logged in.

Steps to Execute:

Navigate to the "Message List" page.

View the list of messages.

Expected Result: All locked messages are displayed correctly in the list.

Actual Result: To be filled after execution.

Status: To be filled after execution.

Test Case 5: Edge Cases and Error Handling

Test Case ID: TC-005

Description: Verify system behavior for invalid input and empty fields.

Preconditions: User must be logged in.

Steps to Execute:

Attempt to create a locked message with empty fields.

Attempt to submit invalid data formats.

Expected Result: System displays appropriate error messages and prevents submission.

Actual Result: To be filled after execution.

Status: To be filled after execution.

2. Execution Report

Execution Report Structure:

Test Case ID	Description	Status	Actual Result	Deviations (if any)
TC-001	Creating a new locked message	Pass	New message created	N/A

TC-002 Editing an existing locked message Fail Error saving changes Save button not responsive

TC-003 Deleting a locked message Pass Message deleted N/A

TC-004 Viewing the list of locked messages Pass Messages displayed N/A

TC-005 Edge Cases and Error Handling Pass Errors shown N/A

Part 3: Bug Report

Bug Report Structure:

Bug ID Summary Steps to Reproduce Expected Result Actual Result Severity Priority Environment Status

BUG-001 Save button not responsive during edit 1. Edit an existing message. 2. Click Save. Changes should be saved. Save button unresponsive. High High Chrome, Windows 10 Open

BUG-002 Error message not displayed for empty fields 1. Submit a message with empty fields. Error message should display. No error message shown. Medium Medium Firefox, macOS In Progress

Attachments:

Screenshots of errors

Logs related to the issues