# Test Plan for Web Application

# 1. Introduction

The goal of this test plan is to describe the approach, resources, scope, and strategies for testing a web application with the following features:

* One-to-one messaging
* Group messaging
* Private messaging
* Call functionalities (video, audio, and conference calls)
* Create ticket
* Create meeting
* Add reminder

## Objectives

To assure that the web application's features all perform as intended, offering a safe and seamless user experience across all platforms and situations.

Verify each feature of the web application functions correctly and effectively.

Ensure the application is user friendly and meets the requirements.

# 2. Scope

## In-Scope

1. **Features to be tested:**

* One-to-one Messaging:

Sending, receiving, and handling one-to-one messages should all be tested.

* Group Messaging:

Verifying the creation, management, and involvement in group chats, as well as the sending and receiving of group messages.

* Private Messaging:

Ensuring access control and privacy settings for the private messaging feature.

* Call Functionalities
  + Video Calls:

Verifying the video quality, call controls (mute, camera on/off, etc.), and connection reliability.

* + Audio Calls:

Verifying call controls, audio clarity, and connection stability.

* + Conference Calls:

Ensuring the audio/video quality overall and allowing numerous participants to join, manage, and exit conference calls.

* Create Tickets:

Testing the various priority levels and status of support tickets as well as their creation, editing, assignment, and tracking.

* Create Meetings:

Ensuring the ability to send invitations, set up meetings, and work with calendar systems.

* Add Reminders:

Analyzing the process of making, managing, and setting off reminders.

1. **Testing Types**

* Functional Testing:

Make sure every feature works according to the requirements and performs as intended.

* Integration Testing:

Verify that different features work smoothly together.

* Performance Testing:

Check the responsiveness, speed, and stability of the program underload.

* Security Testing:

Verify that there are no vulnerabilities in the program including authorization, authentication, data protection.

* Cross browser Testing:

Ensure the application works consistently across various supported browsers.

* Regression Testing:

Re-run previous tests to ensure the software performs as expected after any changes.

## Out-of-Scope

* Unsupported Browsers:

Testing on an outdated browser that is not a part of the project’s requirements or targeted audience.

* Non-Functional Aspects:

Any testing that is not functional and has nothing to do with the main features of the application, such as changes to the UI design, branding, or aesthetics, but has no bearing on performance or usability.

# 3. Test Approach

## Testing Levels

The testing procedure will be divided into several stages to ensure that each aspect of the application is thoroughly covered:

* Unit Testing:

Performed by developers to verify specific code segments or components.

It focuses on confirming that every function or procedure operates properly when used in isolation.

* Integrity Testing:

Ensures that various modules or components function together as intended.

Interactions between the ticketing, meeting, messaging, calling, and reminder services will all be tested.

* System Testing:

Comprehensive testing of the entire program to guarantee that all of its feature’s function as a whole.

Includes all features that fall under in-scope, such as calls, messages, ticketing, meetings, and reminders.

* User Acceptance Testing (UAT):

Carried out with end users or stakeholders.

Confirms that the application satisfies the business requirements and is prepared for production.

**Testing Types**

* Functional Testing:

Verifies that every feature, such as call features, group and one-to-one messaging, private messaging, and call functionality—functions according to the specifications.

Will include both positive and negative test scenarios to ensure robustness.

* Integration Testing:

Ensures that the functionalities of the application work together seamlessly.

Includes scenarios of how calls and messages can be combined, as well as how meetings and reminders work together.

* System Testing:

Full system testing of the application, verifying that all components work together.

Includes end-to-end processes such as ticket management, reminders, and setting up meetings.

* Performance Testing:

Evaluates the scalability, stability, and responsiveness of the application under various load scenarios.

Includes endurance, stress, and load testing to make that the application can manage the anticipated volume of user traffic.

* Security Testing:

Focuses on finding vulnerabilities and ensuring that the program is safe.

Includes data encryption, authentication, and authorization testing as well as defense against common threats.

* Usability Testing:

Ensures that the application is easy to use and offers a satisfying user experience.

Involves testing the ease of use of the platform.

* Regression Testing:

Ensures that any additions or modifications don't interfere with already-existing features.

## Defect Management

* Recording and Monitoring Defects:

Any defects identified will be entered into the defect tracking system.

Defects will be categorized according to priority and severity.

* Retesting and Defect Resolution:

Developers will be tasked with fixing defects.

Defects will be retested once they have been rectified to make sure they have been correctly fixed without creating new problems.

# 4. Test Environment

## Configuring the Environment:

* To ensure compatibility, a range of devices and browsers will be used in test scenarios.
* Testing will take place in a QA test environment that is configured to resemble the production environment.
* Testing tools, defect tracking tools, automation tools (if applicable).

## Test Data:

* Make use of realistic test data that replicates actual situations.
* Provide information for various user roles, such as admin and regular user.

# 5. Milestones / Deliverables

## Milestones

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Task Name** | **Start** | **Finish** | **Effort** | **Comments** |
| Test Planning | 02/09/2024 | 05/09/2024 | 4 days | Planning and strategy development. |
| Review Requirements | 02/09/2024 | 03/09/2024 | 2 days | Document Review and Analysis. |
| Create Initial Test Estimates | 04/09/2024 | 04/09/2024 | 1 day | Estimation based on scope. |
| Deploy to QA test environment | 10/09/2024 | 10/09/2024 | 1 day | Estimations based on scope. |
| Functional testing | 11/09/2024 | 15/09/2024 | 5 days | Functionality Testing of the system |
| System testing | 16/09/2024 | 20/09/2024 | 5 days | End-to-End system testing. |
| Regression Testing | 20/09/2024 | 23/09/2024 | 4 days | Regression testing the previous features. |
| UAT | 24/09/2024 | 29/09/2024 | 6 days | User Acceptance Testing. |
| Final Defects and Build Testing | 30/09/2024 | 03/10/2024 | 4 days | Final testing and bug fixes. |
| Performance Testing | 04/10/2024 | 07/10/2024 | 4 days | Stress and load testing. |
| Release to Production | 08/10/2024 | 08/10/2024 | 1 day | Go-Live in production |

## Deliverables

|  |  |  |
| --- | --- | --- |
| **Deliverable** | **For** | **Milestone** |
| Test Plan | Project Manager; Test Team | 02/09/2024 |
| Test Result | Project Manager | After each iteration |
| Test Status Report | Project Manager | Weekly during Project |
| Metrics | Team Members | Ongoing |
| UAT sign off | Stakeholders | 29/09/2024 |
| Final Test Summary Report | Project Manager | 07/10/2024 |