CIT AI Tech Consulting



Department of AI and DS

Coimbatore Institute of Technology

**Muti user Chat System (MCS)**

**Software Test Document**

**Version: 2.0d**

Document Id: SEP2-STD-V 2.0 d-20231208

Author(s): GOKUL, RANJITH, DINESH KUMAR, VEL MURUGAN

Reviewed By : RANJITH, DINESH KUMAR, VEL MURUGAN

Date: 08-12-2023

**Modification History**

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Modifications** | **Reason** | **Version** |
| 02/10/2023 | Create a draft version of the document. | Adapted from various IEEE standards on software requirements specification | 0.7d |
| 15/10/2023 | Add guidelines and faked examples as italics. | For students to follow | 0.8d |
| 20/10/2023 | Minor amendments | As recommended by reviewer | 0.9d |
| 22/10/2023 | Modify document content for the MCS | Adapted for Assignment 1 of Semester 3 | 1.0d |
| 02/12/2023 | Modify document content for testing documentation purpose | Adapted for Assignment 2 of Semester 3 | 2.0d |

Table 1.0 Modification History

**Contents**

[1. Introduction 1](#_Toc20718127)

[1.1 Document Purpose 1](#_Toc20718128)

[1.2 About the Project 1](#_Toc20718129)

[1.3 Document Scope 1](#_Toc20718130)

[1.4 System Overview 1](#_Toc20718131)

[1.5 Document Overview 1](#_Toc20718132)

[2. Software Test Plan 2](#_Toc20718133)

[2.1 User Acceptance Test Plan 2](#_Toc20718134)

[2.2 Functional Test Plan 3](#_Toc20718135)

[3. Software User Acceptance Test Plan 3](#_Toc20718136)

[3.1 Test case design for “Validation” 3](#_Toc20718137)

[3.1.1 Test Case Specification and Design for” Group Chat Creation” 3](#_Toc20718138)

[3.1.2 Test Case Specification and Design for” Multimedia File Sharing” 4](#_Toc20718139)

[3.1.3 Test Case Specification and Design for” Message Editing and Deletion” 4](#_Toc20718140)

[3.2 Test case design for “Public And Private Chat” 4](#_Toc20718141)

[3.2.1 Test Case Specification and Design for” Joining a Public Chat Room” 4](#_Toc20718142)

[3.2.2 Test Case Specification and Design for” Sending Public Messages” 4](#_Toc20718143)

[3.3 Test case design for “Voice Chat” 4](#_Toc20718141)

[3.3.1 Test Case Specification and Design for” Voice Chat Activation” 4](#_Toc20718142)

[3.3.2 Test Case Specification and Design for” Microphone and Speaker Verification” 4](#_Toc20718143)

[3.4 Test case design for “Video Chat” 4](#_Toc20718141)

[3.4.1 Test Case Specification and Design for” Initiate Video Call” 4](#_Toc20718142)

[3.4.2 Test Case Specification and Design for” Accept Video Call” 4](#_Toc20718143)

[3.5 Test case design for “Add User” 4](#_Toc20718141)

[3.5.1 Test Case Specification and Design for” Adding a User Successfully” 4](#_Toc20718142)

[3.5.2 Test Case Specification and Design for” Adding a User Already in the System” 4](#_Toc20718143)

[3.6 Test case design for “Remove User” 4](#_Toc20718141)

[3.6.1 Test Case Specification and Design for” User Removal Process” 4](#_Toc20718142)

[3.6.2 Test Case Specification and Design for” Confirmation Prompt for User Removal” 4](#_Toc20718143)

[4. Software Functional Test Plan 5](#_Toc20718145)

[4.1 This section Test Case Design for “Registering to the Chat System” 5](#_Toc20718146)

[4.1.1 Test Case Specification and Design for Registering to the chat system 3](#_Toc20718138)

[4.1.2 Test Case Specification and Design for Registering to the chat system 4](#_Toc20718139)

[4.1.3 Test Case Specification and Design for Forgot Password 4](#_Toc20718140)

[4.2 Test Case Design for Login Into The System 5](#_Toc20718146)

[4.2.1 Test Case Specification and Design for Login to the chat system 3](#_Toc20718138)

[4.2.2 Test Case Specification and Design for Registering to the chat system 4](#_Toc20718139)

[4.2.3 Test Case Specification and Design for account information update 4](#_Toc20718140)

[4.3 Test Case Design for “Alert system” 5](#_Toc20718146)

[4.3.1 Test Case Specification and Design for Alert system in the chat 3](#_Toc20718138)

[4.3.2 Test Case Specification and Design for alert system to the chat system 4](#_Toc20718139)

[4.4 Test Case Design for “File permission in the multi chat system.” 5](#_Toc20718146)

[4.4.1 Test Case Specification and Design for file permission in the chat 3](#_Toc20718138)

[4.4.2 Test Case Specification and Design for alert system to the chat system 4](#_Toc20718139)

[4.5 Test Case Design for “Chat search filter in the multi chat system.” 5](#_Toc20718146)

[4.5.1 Test Case Specification and Design for chat search filter in the chat 3](#_Toc20718138)

[4.5.2 Test Case Specification and Design for alert system to the chat system 4](#_Toc20718139)

[5. Requirements Traceability 6](#_Toc20718153)

[References 7](#_Toc20718154)

[Appendix A: Definitions, Acronyms, and Abbreviations 8](#_Toc20718155)

[Appendix B: Contributions 8](#_Toc20718156)

# Introduction

## Document Purpose

This document presents the software test plan prepared by CIT AI Tech Consulting (henceforth referred to as “MCS”) developed for the Multimedia concern (henceforth referred to as “Client”). Specifically, the document details the user acceptance test plans and the functional test plans for the computer software to be developed by CIT AI Tech for the Client.

This document will be subject to formal and informal reviews by the CIT AI Tech development team and the Client, and will form the basis for ongoing testing of the software being developed by CIT AI Tech , to meet the requirements of the Client.

This document forms part of the main deliverable from the software testing design activity. It is intended to be a detailed documentation for the CIT AI Tech development team to continue the quality assurance tasks such as testing as required. Sections 2.1 and 3 constitute the user acceptance test plans, whereas Sections 2.2 and 4 constitute the functional test plans of testing the MCS.

## About the Project

The aim of this project is to perform the ‘Software Test Design’ activity only. All tasks associated with this activity shall be performed based on the theoretical support provided as part of the ‘HIT1031 Introduction to Software Engineering’ course. It is assumed that other related or required activities involved in the development of the system shall be completed under different project(s).

## Document Scope

The document's scope encompasses test case specifications and designs for various functionalities within a chat system, covering registration, chat creation, multimedia sharing, user management, alert systems, file permissions, and search filters (sections 3.1 to 4.5). It outlines detailed test scenarios, expected behaviours, and environments for testing these features. However, it does not include detailed implementation procedures, code-level testing, or specific performance metrics and benchmarks for scalability or load testing in the system. The document aims to provide a comprehensive testing roadmap for functional and system-level validation while excluding low-level implementation details and specific performance metrics.

## System Overview

The ‘Multi user Chat System’ (MCS) is a system that allows users to discuss various information’s and report facility usages in the community of the Client.

Document Overview

This document, the Software Test Document (STD), identifies the software test plan and the details of the testing to be carried out for the MCS.

This document has *X* major sections and *Y* appendices (*students to complete*):

1. **Introduction** (Section 1)provides an overview of the entire MCS system and the MCS Software Test Document.
2. **Software Test Plan** (Section 2) documents the approaches used in different stages of testing from functional testing to user acceptance testing.
3. **Software User Acceptance Test Plan** (Section 3) describes the details of the user acceptance testing of the software.
4. **Software Functional Test Plan** (Section 4) describes the details of testing the functionalities of the system to ensure the software behaves as described in SRS.
5. **Traceability of Requirements**(Section 5) documents the related requirements per test cases in this document.
6. **Appendices:** Supplementary information providing additional context, glossary, or detailed specifications to complement the main test plan.
7. **Test Plan Review Document:** A comprehensive assessment outlining identified faults, corrective actions, and a checklist validating the effectiveness and coverage of the test plan sections (3.1 to 4.5).

# 2. Software Test Plan

This section describes the testing plans for testing the MCS software. In general, this includes the user acceptance test (UAT) and the functional testing. Moreover, it is assumed that the next level of testing is performed only when every test plan in the previous testing level has been carried out without any error.

## User Acceptance Test Plan

This section documents a high level view of the user acceptance test plan and traces the functionalities under test against the SRS of the software. In particular, the user acceptance test plan addresses the user acceptance criteria in the SRS.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Functionality** | **Requirements in SRS** | **Description/Main Responsibilities** | **Feature to be tested** | **Test Spec. and Design** |
| Validation | Section 3.3 | Allow users to validate the multi-user chat system. | Group chat creation | Section 3.1.1 |
|  |  |  | Multimedia file sharing | Section 3.1.2 |
|  |  |  | Message editing and deletion | Section 3.1.3 |
| Public And Private Chat | Section 3.4 | Allow users to send public and private chats. | Joining a public room | Section 3.2.1 |
|  |  |  | Sending public messages | Section 3.2.2 |
| Voice Chat | Section 3.5 | Allow users to send voice messages. | Voice chat activation | Section 3.3.1 |
|  |  |  | Microphone input and Speaker output | Section 3.3.2 |
| Video Chat | Section 3.6 |  | Initiate video call | Section 3.4.1 |
|  |  |  | Accept video call | Section 3.4.2 |
| Add User | Section 3.7 | Allow users to add user to group chat. | Adding new user | Section 3.5.1 |
|  |  |  | Adding existing user | Section 3.5.2 |
| Remove User | Section 3.8 | Allow users to remove user from group chats. | User removal process | Section 3.6.1 |
|  |  |  | Conformation prompt for user | Section 3.6.2 |

Table 2.1 User Acceptance Test Plan Table

## Functional Test Plan

This section documents a high level view of the functional test plan of the software. Moreover, it traces the functionalities under test against the SRS of the software.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Functionality** | **Requirements in SRS** | **Description** | **Feature to be tested** | **Test Spec. and Design** |
| Registration | Section 3.1 | Allow users to register to new account. | Registering to chat system | Section 4.1.1 |
|  |  |  | Entering the chat system | Section 4.1.2 |
|  |  |  | Checking the password and changing it | Section 4.1.3 |
| Login | Section 3.2 | Allow users to login in already existing account. | Login to system | Section 4.2.1 |
|  |  |  | Entering the chat system | Section 4.2.2 |
|  |  |  | Account information update | Section 4.2.3 |
| Alert System | Section 3.9 | Allow users to receive alert messages through notification. | Getting Alert messages | Section 4.3.1 |
|  |  |  | Getting proper notification for alert | Section 4.3.2 |
| File Permission | Section 3.10 | Get Permission to access user’s files | Getting permission to access files | Section 4.4.1 |
| Chat Search Filter | Section 3.11 | Allow user to search chat using keyword. | To check filter is working correctly | Section 4.5.1 |
|  |  |  | To access searched keyword in data base | Section 4.5.2 |

Table 2.2 Functional Test Plan Table

**3. Software User Acceptance Test Plan**

**3.1 Test case design for “Validation”**

**3.1.1 Test Case Specification and Design for”Group Chat Creation”**

**Objective:** Ensure users can create and participate in group chats.

**Features to be tested:**

* Ability for users to create a new group chat.
* Adding existing users to the newly created group chat.
* Setting a name and optional description for the group chat.

**Requirements:**

* User must be logged in to the chat system.
* User must have permission to create group chats.
* The system must be able to handle at least 10 users in a single group chat.

**Expected Behavior:**

* Upon clicking the "Create Group Chat" button, a new form should appear for entering group details.
* The form should allow entering a group name, description, and selecting existing users to be added.
* After submitting the form, a new group chat should be created and the user should be automatically added to it.
* All selected users should receive an invitation to join the group chat.

|  |  |  |  |
| --- | --- | --- | --- |
| Test ID | Input | Environment Needs | Output Expected |
| 3.1.1.1 | User clicks on "Create Group Chat" button. | Chat system is available and accessible. | Form for creating a new group chat appears. |
| 3.1.1.2 | User enters a group name, description, and selects existing users. | Chat system is online and connected to the internet. | System validates the entered information and displays a confirmation message. |
| 3.1.1.3 | User submits the form. | Chat system is connected to the database server. | New group chat is created and the user is added to it. |
| 3.1.1.4 | User clicks on "Invite More Users" button. | Chat system is online and connected to the internet. | System displays a list of available users to invite. |
| 3.1.1.5 | User selects additional users and clicks on "Send Invitations". | Chat system is connected to the messaging server. | System sends invitations to the selected users. |
| 3.1.1.6 | Invited user receives a notification and accepts the invitation. | Chat system is running on the user's device. | User is added to the group chat and can see its messages. |

Table 1.1.1 - Testcase Design for "validation"

**3.1.2 Test Case Specification and Design for”****Multimedia File Sharing”**

**Objective:** Validate the ability to share multimedia files (images, videos, documents) in the chat system.

**Features to be tested:**

* Sending and receiving multimedia files (images, videos, audios) within a chat conversation.
* File preview functionality for images and videos.
* Downloading and uploading multimedia files.
* File size limitations for uploads.
* Error handling for invalid file formats or exceeding file size limits.

**Requirements:**

* The system must be able to handle different multimedia file formats (e.g., JPG, PNG, MP4, MP3).
* File previews must be generated quickly and accurately.
* Downloading and uploading must be seamless and efficient.
* The system should effectively handle file size limitations and display clear error messages for invalid formats or exceeding limits.

**Expected Behaviour:**

* Users can select and send multimedia files from their device or online storage.
* Sent files are displayed in the chat conversation with a preview thumbnail for images and videos.
* Users can click on the preview to download the file.
* Downloaded files are saved to the user's device in the original format.
* The system displays clear and informative error messages for invalid file formats or exceeding file size limits.

|  |  |  |  |
| --- | --- | --- | --- |
| **Test ID** | **Inputs** | **Required Environment** | **Output Expected** |
| 3.1.2.1 | Image file in JPG format | Stable internet connection, chat conversation open with another user | Image preview displayed correctly, download option available |
| 3.1.2.2 | Video file in MP4 format | Stable internet connection, chat conversation open with another user | Video preview starts automatically with play/pause controls, download option available |
| 3.1.2.3 | Audio file in MP3 format | Stable internet connection, chat conversation open with another user | Audio file starts playing automatically with play/pause controls, download option available |
| 3.1.2.4 | File exceeding size limit | Stable internet connection, chat conversation open with another user | Error message displayed indicating file size limit exceeded |
| 3.1.2.5 | Unsupported file format | Stable internet connection, chat conversation open with another user | Error message displayed indicating unsupported file format |

Table 3.1.2 - Testcase Design for "Multimedia file sharing"

**3.1.3 Test Case Specification and Design for”****Message Editing and Deletion”**

**Objective:** Validate the ability to edit and delete messages

**Features to be tested:**

* Editing sent messages.
* Deleting sent messages.
* Confirmation messages for editing and deleting actions.
* Undoing edits and deletions.

**Requirements:**

* Users should be able to edit their own sent messages within a specific time frame (e.g., 5 minutes).
* Users should be able to delete their own sent messages.
* Deleting a message should remove it from the chat history for all participants.
* Users should be notified about the editing and deleting actions of other participants.
* Users should be able to undo edits and deletions within a short period (e.g., 15 seconds).

**Expected behavior:**

* When a user edits a message, the updated message should be displayed in the chat history for all participants.
* When a user deletes a message, it should be removed from the chat history and replaced with a notification indicating the action.
* Users should see a confirmation message before editing or deleting a message.
* Undoing an edit or deletion should restore the message to its previous.

|  |  |  |  |
| --- | --- | --- | --- |
| Test ID | Inputs | Required Environment | Expected Output |
| 3.1.3.1 | User edits a sent message within the time limit. | Active chat session with existing sent message. | Message is updated in the chat history and participants are notified. |
| 3.1.3.2 | User attempts to edit a sent message outside the time limit. | Active chat session with existing sent message exceeding time limit. | Error message indicating that editing is no longer allowed. |
| 3.1.3.3 | User deletes a sent message. | Active chat session with existing sent message. | Message is removed from the chat history and replaced with a deletion notification. |
| 3.1.3.4 | User attempts to delete a message sent by another participant. | Active chat session with existing message sent by another participant. | Error message indicating that deleting others' messages is not allowed. |
| 3.1.3.5 | User undoes a recent edit or deletion. | Active chat session with recent editing or deletion action. | Message is restored to its previous state. |

Table 3.1.3 - Testcase Design for "Message editing and deletion"

**3.2 Test case design for “Public And Private Chat”**

**3.2.1 Test Case Specification and Design for”****Joining a Public Chat Room”**

**Objective:** Ensure users can join a public chat room successfully.

Test Cases for Joining a Public Chat Room in a Multi-User Chat System

**Features to be tested:**

* The ability for a user to successfully join a public chat room.
* The display of available public chat rooms in the user interface.
* The visibility of the chat history and current participants within the joined chat room.
* The ability for the user to send and receive chat messages within the joined chat room.

**Requirements:**

* The user must have an active account and be logged in to the chat system.
* The user must have a valid internet connection.
* There must be at least one public chat room available to join.

**Expected behavior:**

* When the user clicks on a public chat room, they should be automatically joined to the room.
* The chat history and current participants should be displayed in the chat room window.
* The user should be able to send and receive chat messages in the chat room window.
* New messages should be displayed in real-time for all participants in the chat room.

**Test Case Design Table**

|  |  |  |  |
| --- | --- | --- | --- |
| Test ID | Inputs | Required Environment | Expected Output |
| 3.2.1.1 | User clicks on a public chat room. | Active account, logged in user, internet connection, available public chat room. | User is automatically joined to the public chat room. |
| 3.2.1.2 | User sends a chat message. | Joined public chat room, valid message content. | Chat message is sent and displayed in the chat room window. |
| 3.2.1.3 | User receives a chat message from another participant. | Joined public chat room, another participant sends a message. | Chat message is received and displayed in the chat room window. |
| 3.2.1.4 | User attempts to join a non-existent public chat room. | User clicks on a non-existent public chat room link. | An error message is displayed, indicating that the chat room does not exist. |
| 3.2.1.5 | User attempts to join a public chat room with a disabled account. | User with a disabled account attempts to join a public chat room. | An error message is displayed, indicating that the user's account is disabled. |

Table 3.2.1 - Testcase Design for "Joining a public chat room"

**3.2.2 Test Case Specification and Design for”****Sending Public Messages”**

**Objective:** Verify that users can send messages in a public chat room.

**Features to be tested:**

* Sending public messages to all users in the chat room
* Sending messages with different lengths and formats (text, emojis, URLs)
* Editing and deleting sent public messages

**Requirements:**

* User must be logged in and connected to the chat room
* Text area must be available to compose messages
* Public message button or equivalent functionality must be present
* Messages must be displayed in the chat room for all users to see
* Sent messages must be editable and deletable within a reasonable timeframe

**Expected behavior:**

* When a user types a message and clicks the public message button, the message should be sent to all users in the chat room and displayed in real-time.
* Users should be able to edit their sent public messages until a specific time limit has passed.
* Users should be able to delete their sent public messages.

**Test Case Design:**

|  |  |  |  |
| --- | --- | --- | --- |
| Test ID | Inputs | Environment | Expected Output |
| 3.2.2.1 | Text message | Stable internet connection, Connected to chat room | Public message displayed in chat room with correct username and message content |
| 3.2.2.2 | Emoji message | Stable internet connection, Connected to chat room | Public message displayed in chat room with emojis rendered correctly |
| 3.2.2.3 | URL message | Stable internet connection, Connected to chat room | Public message displayed in chat room with URL clickable and opening the linked webpage |
| 3.2.2.4 | Edit public message | Public message sent within the time limit | Edited message displayed in chat room with updated content |
| 3.2.2.5 | Delete public message | Public message sent within the time limit | Public message disappears from chat room history |

Table 3.2.2 - Testcase Design for "Sending public message"

**3.3 Test case design for “Voice Chat”**

**3.3.1 Test Case Specification and Design for”****Voice Chat Activation”**

**Objective:** Verify that users can activate voice chat during a conversation.

**Features to be tested:**

* User can activate voice chat by pressing a dedicated button.
* User can deactivate voice chat by pressing the button again.
* User's voice is transmitted clearly and without significant delay to other participants.
* Only the voice of the user who activated voice chat is transmitted.
* Background noise is minimized during voice transmission.

**Requirements:**

* The voice chat button should be clearly visible and easy to locate.
* User should receive immediate audio confirmation upon activating and deactivating voice chat.
* Voice quality should be high enough for clear and intelligible communication.
* System should effectively suppress background noise and other unwanted sounds.
* User should be able to speak naturally without voice clipping or distortion.

**Expected behavior:**

* When the voice chat button is pressed, the user's microphone is activated and their voice is transmitted to other participants.
* When the voice chat button is pressed again, the microphone is deactivated and the user's voice is no longer transmitted.
* Other participants should hear the user's voice clearly and without significant delay.
* Background noise should be minimal and not interfere with communication.
* The user should be able to speak naturally without any noticeable audio issues.

|  |  |  |  |
| --- | --- | --- | --- |
| Test ID | Inputs | Required Environment | Expected Output |
| 3.3.1.1 | Press voice chat button | Active internet connection, working microphone | User's microphone is activated, audio confirmation is received, other participants hear user's voice. |
| 3.3.1.2 | Press voice chat button again | Active internet connection, working microphone | User's microphone is deactivated, audio confirmation is received, other participants no longer hear user's voice. |
| 3.3.1.3 | Speak while voice chat is active | Active internet connection, working microphone | Other participants hear user's voice clearly and without significant delay. |
| 3.3.1.4 | Speak with background noise (e.g., music, conversation) | Active internet connection, working microphone | Other participants hear user's voice with minimal background noise. |
| 3.3.1.5 | Speak at different volumes (soft, loud) | Active internet connection, working microphone | User's voice is transmitted at the appropriate volume, without clipping or distortion. |

Table 3.3.1 - Testcase Design for "Voice chat Activation"

**3.3.2 Test Case Specification and Design for”****Microphone and Speaker Verification”**

**Objective:** Ensure the system correctly detects and uses the microphone and speaker.

**Features to be tested:**

* Microphone input capture clarity and accuracy.
* Speaker output audio quality and clarity.
* User identification through microphone and speaker verification.

**Requirements:**

* The system should capture microphone input with minimal distortion and background noise.
* The system should output clear and intelligible audio through speakers.
* The system should accurately identify users based on their voice and speaker characteristics.

**Expected behavior:**

* Users should be able to speak clearly and naturally into the microphone and be understood by other users.
* Users should be able to hear other users speaking clearly and without distortion.
* The system should consistently identify users correctly, even in noisy environments or with multiple users speaking.

**Test Case Design**

|  |  |  |  |
| --- | --- | --- | --- |
| Test ID | Input | Environment Needs | Expected Output |
| 3.3.2.1 | User speaks clearly into the microphone. | Background noise is minimal. | Microphone input should be clear and undistorted. |
| 3.3.2.2 | User speaks in a noisy environment. | Background noise is significant. | Microphone input should be understandable, but may have some background noise. |
| 3.3.2.3 | Multiple users speak simultaneously. | Multiple voices are present in the environment. | The system should identify each user accurately based on their individual voice characteristics. |
| 3.3.2.4 | User whispers into the microphone. | The environment is quiet. | The microphone should capture the whisper clearly. |
| 3.3.2.5 | User speaks with a non-standard accent or dialect. | The system has been trained on diverse voice samples. | The system should still identify the user correctly. |

Table 3.3.2 - Testcase Design for "Microphone and speaker verification"

**3.4 Test case design for “Video Chat”**

**3.4.1 Test Case Specification and Design for”****Initiate Video Call”**

**Objective:** Ensure users can initiate a video call successfully.

User Acceptance Testing for Initiate Video Call in Multiple User Chat System

**Features to be tested:**

* Initiating a video call with one or multiple participants from a chat group.
* Accepting or rejecting incoming video call requests.
* Switching between video and audio-only call during the call.
* Muting or unmuting the microphone and camera during the call.
* Ending the video call.

**Requirements:**

* Users should be able to see a list of online participants in the chat group.
* The video call should start with the user's own camera and microphone enabled by default.
* Participants should be able to see and hear each other clearly during the call.
* The call should be stable with minimal audio and video lag.
* Users should be able to easily switch between video and audio-only mode.
* Users should be able to mute and unmute their microphone and camera individually.
* Users should be able to end the call for themselves or all participants.

**Expected Behavior:**

* When a user clicks the "Initiate Video Call" button, a pop-up window should appear allowing them to select participants from the chat group.
* Clicking on a participant's name should add them to the video call list.
* Once all participants have been selected, clicking the "Start Call" button should initiate the video call.
* Incoming video call requests should appear as notifications on the user's screen.
* Users should be able to accept or reject the call by clicking the appropriate buttons.
* During the call, users should see a live video feed of themselves and other participants.
* Clicking the "Switch to Audio" button should disable the user's camera and continue the call in audio-only mode.
* Clicking the "Mute Microphone" or "Mute Camera" buttons should disable the respective function.
* Clicking the "End Call" button should terminate the call for all participants.

**Test Case Design:**

|  |  |  |  |
| --- | --- | --- | --- |
| Test ID | Inputs | Required Environment | Expected Output |
| 3.4.1.1 | Initiate a video call with one participant | Chat group with at least two online users | Video call starts with both users seeing and hearing each other clearly. |
| 3.4.1.2 | Accept an incoming video call request | Active internet connection | User joins the video call and can see and hear other participants. |
| 3.4.1.3 | Switch between video and audio-only mode during the call | Video call with at least two participants | User's video is disabled/enabled, and audio continues/mutes smoothly. |
| 3.4.1.4 | Mute microphone and camera during the call | Video call with at least two participants | Other participants cannot hear the user's audio or see their video. |
| 3.4.1.5 | End the video call | Video call with at least two participants | Video call ends for all participants. |

Table 3.4.1 - Testcase Design for "Initial video call"

**3.4.2 Test Case Specification and Design for”****Accept Video Call”**

**Objective:** Verify that users can accept an incoming video call.

**Features to be tested:**

* Accepting an incoming video call from another user within the chat system.
* Checking video and audio quality during the video call.
* Muting/unmuting microphone and turning on/off camera during the video call.
* Minimizing and maximizing the video call window.
* Ending the video call.

**Requirements:**

* Users should be able to accept incoming video calls with a single click.
* Video and audio quality should be clear and without significant delays.
* Users should be able to mute/unmute their microphone and turn on/off their camera during the video call.
* Users should be able to minimize the video call window and continue using other applications.
* Users should be able to end the video call at any time.

**Expected behavior:**

* When a user receives an incoming video call, a notification should appear on their screen with the caller's name and avatar.
* Clicking on the "Accept" button should connect the users and start the video call.
* Video and audio should be transmitted smoothly between users.
* Clicking on the microphone icon should mute the user's microphone and clicking again should unmute it.
* Clicking on the camera icon should turn off the user's camera and clicking again should turn it on.
* Clicking on the minimize button should shrink the video call window to a smaller size and clicking on the maximize button should restore it to its original size.
* Clicking on the "End Call" button should end the video call.

**Test Case Design Table:**

|  |  |  |  |
| --- | --- | --- | --- |
| Test ID | Inputs | Environment Needs | Expected Output |
| 3.4.2.1 | User receives an incoming video call. | Stable internet connection. | Video call notification appears on user's screen. |
| 3.4.2.2 | User clicks on the "Accept" button. | High-quality webcam and microphone. | Video call connects and video and audio are transmitted clearly. |
| 3.4.2.3 | User clicks on the "Mute" button. | None | User's microphone is muted and a microphone icon with a slash appears. |
| 3.4.2.4 | User clicks on the "Minimize" button. | None | Video call window shrinks to a smaller size. |
| 3.4.2.5 | User clicks on the "End Call" button. | None | Video call ends and the user returns to the chat window. |

Table 3.4.2 - Testcase Design for "Accept video call"

**3.5 Test case design for “Add User”**

**3.5.1 Test Case Specification and Design for”****Adding a User Successfully”**

**Objective:** Ensure users can be successfully added to the chat system.

User Acceptance Test for Adding a User Successfully in a Multi-User Chat System

**Features to be tested:**

* User creation functionality
* User data validation
* User account activation

**Requirements:**

* Users must be able to create new accounts by providing valid information such as username, email address, and password.
* User data must be validated for format and correctness.
* New user accounts must be activated before the user can access the chat system.

**Expected behavior:**

* When a user attempts to create a new account, the system should:
  + Prompt the user for required information (username, email address, password).
  + Validate the entered information for format and correctness.
  + If the information is valid, create a new user account and send an activation email.
  + If the information is invalid, display clear and concise error messages.
* Once the user receives the activation email, clicking the activation link should activate their account and allow them to log in to the chat system.

**Test Case Design:**

|  |  |  |  |
| --- | --- | --- | --- |
| Test ID | Inputs | Environment Needs | Expected Output |
| 3.5.1.1 | Valid username, valid email address, valid password | Stable internet connection, access to email | New user account is created successfully, activation email is sent |
| 3.5.1.2 | Empty username field | Stable internet connection | Error message is displayed indicating missing username |
| 3.5.1.3 | Invalid email address format | Stable internet connection | Error message is displayed indicating invalid email address format |
| 3.5.1.4 | Existing username | Stable internet connection | Error message is displayed indicating username is already taken |
| 3.5.1.5 | Clicking activation link in email | Stable internet connection | User account is activated and login is allowed |

Table 3.5.1 - Testcase Design for "Adding a user successfully"

**3.5.2 Test Case Specification and Design for”****Adding a User Already in the System”**

**Objective:** Verify that the system handles attempts to add a user who is already part of the chat system**.**

**Features to be tested:**

* Adding a user already present in the system.
* Validation of user presence before adding.
* Error handling for duplicate user addition.

**Requirements:**

* Users should be able to add other users to the chat system.
* The system should validate if the user being added already exists.
* If the user already exists, an appropriate error message should be displayed.

**Expected Behavior:**

* When a user attempts to add an existing user, the system should display a message indicating that the user is already present.
* The system should not allow adding the same user twice.

**Test Case Design**

|  |  |  |  |
| --- | --- | --- | --- |
| Test ID | Inputs | Environment Needs | Expected Output |
| 3.5.2.1 | Existing user username | User list | Message: "User already exists." |
| 3.5.2.2 | Non-existent user username | User list | User added successfully to the chat system. |
| 3.5.2.3 | Empty username | User list | Error message: "Please enter a username." |
| 3.5.2.4 | User tries to add themselves | User list | Message: "You cannot add yourself to the chat." |
| 3.5.2.5 | User attempts to add with invalid characters | User list | Error message: "Invalid username format." |

Table 3.5.2 - Testcase Design for "Adding a user already in the system"

**3.6 Test case design for “Remove User”**

**3.6.1 Test Case Specification and Design for”****User Removal Process”**

**Objective:** Verify that users can be successfully removed from the chat system.

**Features to be tested:**

* Ability to remove users from a chat conversation.
* Confirmation process before user removal.
* Notification to removed user and remaining participants.
* Removal of user history from the conversation.

**Requirements:**

* Only authorized users (e.g., chat owner, moderator) can remove users.
* Users must have a valid reason for removing someone.
* Removed users should not be able to rejoin the conversation.
* Usernames and avatars of removed users should be hidden from the conversation history.

**Expected behavior:**

* When authorized users attempt to remove a user, a confirmation pop-up should appear.
* The pop-up should clearly state the action being taken and its consequences.
* The user should have the option to cancel or confirm the removal.
* Upon confirmation, the user should be removed from the conversation and notified.
* A notification should also be sent to remaining participants about the removal.
* The user's history and profile information should be hidden from the chat conversation.

**Test Case Design:**

|  |  |  |  |
| --- | --- | --- | --- |
| Test ID | Inputs | Required Environment | Expected Output |
| 3.6.1.1 | User attempts to remove another user. | Authorized user logged in. | Confirmation pop-up appears, stating the action and consequences. |
| 3.6.1.2 | User cancels removal confirmation. | Confirmation pop-up open. | Removal process is aborted, user remains in the chat. |
| 3.6.1.3 | Authorized user confirms removal. | Confirmation pop-up open. | User is removed from the chat and notified. |
| 3.6.1.4 | Removed user attempts to rejoin the chat. | Removed user attempts to access the conversation. | Access denied, message stating removal notification displayed. |
| 3.6.1.5 | User checks conversation history. | Authorized user views the chat history. | Removed user's username and avatar are hidden, messages attributed to "Removed User". |

Table 3.6.1 - Testcase Design for "User removal process"

**3.6.2 Test Case Specification and Design for”****Confirmation Prompt for User Removal”**

**Objective:** Ensure that a confirmation prompt is displayed before removing a user.

Confirmation Prompt for User Removal in Multiple User Chat System: User Acceptance Testing (UAT)

**Features to be tested:**

* The user interface element for initiating user removal.
* The confirmation prompt displayed before user removal.
* The functionality of confirming and canceling user removal.

**Requirements:**

* The user interface element for initiating user removal should be clearly visible and accessible.
* The confirmation prompt should clearly state the consequences of user removal.
* The user should have the option to confirm or cancel user removal.
* After confirming user removal, the user should be removed from the chat and relevant notifications should be sent to other participants.
* After canceling user removal, the user should remain a participant in the chat.

**Expected behavior:**

* When a user clicks the user removal element, a confirmation prompt should appear.
* The prompt should clearly state the username and chat name being removed.
* The prompt should offer two buttons: "Confirm Removal" and "Cancel."
* Clicking "Confirm Removal" should remove the user from the chat and send notifications to other participants.
* Clicking "Cancel" should close the prompt without making any changes.

**Test Case Design**

|  |  |  |  |
| --- | --- | --- | --- |
| Test ID | Inputs | Required Environment | Expected Output |
| 3.6.2.1 | User clicks the user removal element. | Multiple user chat session. | Confirmation prompt appears with username and chat name. |
| 3.6.2.2 | User clicks "Confirm Removal." | Multiple user chat session. | User is removed from the chat. Notification sent to other participants. |
| 3.6.2.3 | User clicks "Cancel." | Confirmation prompt displayed. | Confirmation prompt closes. User remains in the chat. |
| 3.6.2.4 | User attempts to remove non-existent user. | Multiple user chat session. | Error message displayed stating the user cannot be found. |
| 3.6.2.5 | User attempts to remove themselves from the chat. | Multiple user chat session. | Confirmation prompt appears with user's own username and chat name. User can still choose to remove themselves or cancel. |

Table 3.6.2 - Testcase Design for "Confirmation prompt for user removal"

# 4 Software Functional Test Plan

This section describes the details of the test plan for the system testing which includes functional testing and performance testing.

## Test Case Design for “Registering to the Chat System”

This section describes the specifications of the test cases for registering to the chat system.

## 4.1.1 Test Case Specification and Design for Registering to the chat system

**Features to be tested**: To test whether the user is able to register themselves to the chat system properly.

**Requirements**: Proper details of the user who is registering to the system.

**Expected Behaviour**: The system create the account with the given details and store the details in the database.

**Test Case Design**:

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Id.** | **Inputs required** | **Environment needs** | **Output expected** |
| 4.1.1.1 | **Personal Information:** Typically, you'll need to provide basic personal details such as your name, email address, and sometimes a username. | **Device Compatibility:** Ensure that the chat system is compatible with your device (e.g., computer, smartphone) and operating system. | **Account Confirmation:** After providing the necessary inputs, you may receive an email or SMS for account verification. Follow the provided instructions to confirm your registration. |
| 4.1.1.2 | **Authentication Credentials:** Create a secure password or use multi-factor authentication to secure your account. | **Internet Connection:** A stable internet connection is usually required for real-time communication. | **Access to Chat System:** Upon successful registration, you should gain access to the chat system and its features. |
| 4.1.1.3 | **Contact Information:** Some systems may require additional contact information for communication purposes*.* | **Browser or App:** Depending on the platform, you may need a specific web browser or app to access the chat system. | **User Interface:** You'll interact with the chat interface to send/receive messages, join channels or groups, and manage your contacts. |
| 4.1.1.4 | **Profile Settings:** Depending on the chat system, you might be asked to set preferences, such as profile pictures or status messages. | **Privacy Settings:** Review and configure privacy settings based on your preferences. | **Notifications:** Depending on the system, you might receive notifications for messages, friend requests, or other relevant activities. |

Table 4.1.1 – Test Case Design for registering to the chat system

### 4.1.2 Test Case Specification and Design for Registering to the chat system

### Features to be tested: After the registration of the account the user should be able to enter to the chat system.

**Requirements**: The entered details of the user must meet the conditions of the multi chat system.

**Expected Behaviour**: Upon successful registration, you should gain access to the chat system and its features

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Id.** | **Inputs required** | **Environment needs** | **Output Expected** |
| 4.1.2.1 | Name, E-mail, Username | Compatible Device | Access to the chat system. |
| 4.1.2.2 | Secured Password | Device Compatibility | Ability to login successfully |
| 4.1.2.3 | Optional :Phone Number | Internet Connection | Enhanced Profile Configuration. |

Table 4.1.2 – Test Case Design for registering to chat system

### 4.1.3 Test Case Specification and Design for Forgot Password:

**Features to be tested**: To check whether the user is able to change the password if he does not remember the password.

**Requirements**: Entered details must be valid and not invalid.

**Expected Behaviour**: Password should be changed successfully.

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Id.** | **Inputs required** | **Environment needs** | **Output Expected** |
| 4.1.3.1 | Valid Email Address | Ensure the system is accessible and the "Forgot Password" functionality is enabled. | The user should receive a password reset email. |
| 4.1.3.2 | Password Reset Link Expiration | Ensure the system is accessible and the "Forgot Password" functionality is enabled. | The system should display a message indicating that the password reset link has expired, and the user needs to request a new one. |
| 4.1.3.3 | Security Measures | Ensure the system has security measures in place to prevent abuse, such as rate limiting on password reset requests. | The system should prevent abuse by implementing security measures like rate limiting. |

Table 4.1.3 – Test Case Design for Forgot password

## Test Case Design for Login Into The System:

## This section describes the specifications of the test cases for Login to the chat system.

## 4.2.1 Test Case Specification and Design for Login to the chat system

**Features to be tested**: To test whether the user is able to Login to the system.

**Requirements**: Proper login details of the registered account.

**Expected Behaviour**: The system should authenticate the entered details and should aloe the user to enter into the multi chat system.

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Id.** | **Inputs required** | **Environment needs** | **Output expected** |
| 4.2.1.1 | Enter valid username and password.. | **Internet Connection:** A stable internet connection is usually required for real-time communication | The user should be logged in and redirected to the system's dashboard or chat interface. |
| 4.2.1.2 | Invalid Login - Incorrect Username | **Internet Connection:** A stable internet connection is usually required for real-time communication. | The system should display an error message indicating that the username is incorrect. |
| 4.2.1.3 | Session Timeout | **Internet Connection:** A stable internet connection is usually required for real-time communication. | The system should prompt the user to log in again due to session timeout. |
| 4.2.1.4 | Evaluate the login feature for accessibility compliance. | **Internet Connection:** A stable internet connection is usually required for real-time communication.. | The login process should be accessible and navigable for users with disabilities.. |

Table 4.2.1 – Test Case Design for Login to chat system

### 4.2.2 Test Case Specification and Design for Registering to the chat system

### Features to be tested: After the registration of the account the user should be able to enter to the chat system.

**Requirements**: Proper details of the user who is registering to the system.

**Expected Behaviour**: Upon successful registration, you should gain access to use the chat system and its features.

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Id.** | **Inputs required** | **Environment needs** | **Output Expected** |
| 4.2.2.1 | Verify if the login functionality works across different browsers. | Perform login tests using various browsers (Chrome, Firefox, Safari, etc.).. | The login feature should work consistently across different browsers without any discrepancies. |
| 4.2.2.2 | Verify the system's response to an incorrect username. | Ensure the system is accessible and the "Forgot Password" functionality is enabled. | The system should display an error message indicating that the username is incorrect. |
| 4.2.2.3 | Security Measures | Ensure the system has security measures in place to prevent abuse, such as rate limiting on password reset requests. | The system should prevent abuse by implementing security measures like rate limiting. |

Table 4.2.2 – Test Case Design for Registering to chat system

### 4.2.3 Test Case Specification and Design for account information update:

**Features to be tested**: To ensure that the user is able to update their profile or the account.

**Requirements**: The old entered details must be known by the user.

**Expected Behaviour**: Account details updated successfully.

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Id.** | **Inputs required** | **Environment needs** | **Output Expected** |
| 4.2.3.1 | Old account information (Username, Email, Password). | **Internet Connection:** A stable internet connection is usually required for real-time communication. Access to the system with the account change feature.  Test user account with the necessary permissions. | System allows the user to change the information by redirecting to the profile update page. |
| 4.2.3.2 | New account information (Updated Username, Email, Password). | **Internet Connection:** A stable internet connection is usually required for real-time communication. Access to the system with the account change feature.  Test user account with the necessary permissions. | System displays a success message confirming the account information update and the system reflects the changes in the user's account. |
| 4.2.3.3 | Current Password.  New Password (not meeting the password strength criteria). | **Internet Connection:** A stable internet connection is usually required for real-time communication.Access to the system with the account change feature.Test user account with the necessary permissions. | System displays an error message indicating that the password does not meet the strength requirements.  User is prevented from changing the password until the criteria are met. |

Table 4.2.3 – Test Case Design for Account Information update

## Test Case Design for “Alert system”

This section describes the specifications of the test cases for alert system in the chat system.

## 4.3.1 Test Case Specification and Design for Alert system in the chat

**Features to be tested**: To test whether the user is able to get alert messages.

**Requirements**: Proper instructions to be given to the system for alerting.

**Expected Behaviour**: The system should alert the user on any suspicious behaviour.

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Id.** | **Inputs required** | **Environment needs** | **Output expected** |
| 4.3.1.1 | Multi-chat system data (messages, user interactions). | **Access to message logs and user data.**. | Immediate alert activation upon detection of trigger event.  Customizable alert types based on user preferences.  Clear indication of the triggering event (message, user involved). |
| 4.3.1.2 | Simulate a trigger event with specific parameters. | **Internet Connection:** A stable internet connection is usually required for real-time communication. | System detects the event and initiates the alert. |
| 4.3.1.3 | Change user preferences for alert type. | **Internet Connection:** A stable internet connection is usually required for real-time communication. | Alerts are displayed according to the updated preferences. |
| 4.3.1.4 | Generate multiple trigger events in quick succession. | **Internet Connection:** A stable internet connection is usually required for real-time communication. | Alerts are activated promptly for each event. |

Table 4.3.1 – Test Case Design for Alert system in chat

### 4.3.2 Test Case Specification and Design for alert system to the chat system

### Features to be tested: Whether the user is getting proper notification of alert or not.

**Requirements**: The entered details of the user must meet the conditions of the multi chat system.

**Expected Behaviour**: Alert to the user.

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Id.** | **Inputs required** | **Environment needs** | **Output Expected** |
| 4.3.2.1 | Test alert dismissal and user interaction options. | User Interaction | Alerts can be dismissed, and users can interact with them as intended. |
| 4.3.2.2 | extreme scenarios (e.g., very long messages, rapid influx of messages). | Device Compatibility | Alert system handles edge cases gracefully without errors or delays. |
| 4.3.2.3 | Detect and respond to specific trigger events. | Internet Connection | Detect and respond to specific trigger events. |

Table 4.3.2 – Test Case Design for alert system to chat system

## Test Case Design for “File permission in the multi chat system.”

This section describes the specifications of the test cases for file permission in the chat system.

## 4.4.1 Test Case Specification and Design for file permission in the chat

**Features to be tested**: To ask permission from the user to access the file.

**Requirements**: proper permission requirements.

**Expected Behaviour**: Permission alert to the user’s notification.

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Id.** | **Inputs required** | **Environment needs** | **Output expected** |
| 4.4.1.1 | File permissions must be enforced at the server level to ensure security. | **Internet Connection:** A stable internet connection is usually required for real-time communication. | Full access to all files. |
| 4.4.1.2 | Permissions should include read, write, and execute rights. | **Internet Connection:** A stable internet connection is usually required for real-time communication. | Able to read files, but not modify or execute. |
| 4.4.1.3 | Admin Access | **Internet Connection:** A stable internet connection is usually required for real-time communication. | Alerts are displayed according to the updated preferences. |
| 4.4.1.4 | Regular user credentials with write permissions | **Internet Connection:** A stable internet connection is usually required for real-time communication. | Able to modify files, but not read or execute. |

Table 4.4.1 – Test Case Design for file permission in multi-chat system

### 4.4.2 Test Case Specification and Design for alert system to the chat system

**Features to be tested**: To ask permission from the user to access the file.

**Requirements**: proper permission requirements.

**Expected Behaviour**: Permission alert to the user’s notification.

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Id.** | **Inputs required** | **Environment needs** | **Output Expected** |
| 4.4.2.1 | Regular User Write Access | Ensure server configuration supports user roles and file permissions.  Simulate concurrent user interactions to test scalability. | Able to read files, but not modify or execute. |
| 4.4.2.2 | unauthorized actions and ensure appropriate error | Test cases should include scenarios with varying network conditions to ensure robustness. | Test scenarios where users attempt unauthorized actions and ensure appropriate error messages are displayed. |
| 4.4.2.3 | Detect and respond to specific trigger events. | Test cases should include scenarios with varying network conditions to ensure robustness. | File system responses should align with the granted permissions.  System logs should record and report an unauthorized access attempts. |

Table 4.4.2 – Test Case Design for alert system to chat system

## Test Case Design for “Chat search filter in the multi chat system.”

This section describes the specifications of the test cases for file permission in the chat system.

## 4.5.1 Test Case Specification and Design for chat search filter in the chat

**Features to be tested**: Whether the filter in the search is working correctly.

**Requirements**: Entering proper matching words used while searching before.

**Expected Behaviour**: Suggesting the expected result in the search box.

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Id.** | **Inputs required** | **Environment needs** | **Output expected** |
| 4.5.1.1 | Enter a specific keyword that exists in one of the chat conversations. | Multiple chat conversations with diverse content. | Relevant chat messages containing the specified keyword. |
| 4.5.1.2 | Enter the name of a participant in one of the chat conversations. | Chats with known participants. | Chat messages involving the specified participant. |
| 4.5.1.3 | Enter a keyword that does not exist in any chat. | Diverse chat content. | No search results. |
| 4.5.1.4 | Enter the name of a participant not present in any chat. | Chats with known participants. | No search results. |

### 4.5.2 Test Case Specification and Design for search filter to the chat system

**Features to be tested**: Whether the filter in the search is working correctly.

**Requirements**: Entering proper matching words used while searching before.

**Expected Behaviour**: Suggesting the expected result in the search box.

**Test Case Design**:

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Id.** | **Inputs required** | **Environment needs** | **Output Expected** |
| 4.5.2.1 | Leave the search field empty. | Chats with diverse content. | All chat messages are displayed, indicating no filtering. |
| 4.5.2.2 | Enter a long search query. | Chats with diverse content. | Relevant messages matching the long search query. |
| 4.5.2.3 | Enter a specific keyword that exists in one of the chat conversations. | Multiple chat conversations with diverse content. | Relevant chat messages containing the specified keyword. |

**Features to be tested**: Whether the filter in the search is working correctly.

**Requirements**: Entering proper matching words used while searching before.

**Expected Behaviour**: Suggesting the expected result in the search box.

**Test Case Design**:

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Id.** | **Inputs required** | **Environment needs** | **Output Expected** |
| 4.5.3.1 | A database with a variety of chat conversations. | Ensure a stable network connection for real-time search. | Display chat messages matching the search criteria. |
| 4.5.3.2 | Multiple user accounts with distinct participants in chats. | Ensure a stable network connection for real-time search. | instantaneous display of search results as the user inputs the search query. |
| 4.5.3.3 | Enter a keyword that does not exist in any chat. | Diverse chat content. | No search results. |

# 5. Requirements Traceability

Requirements traceability is a crucial aspect of the software development process that ensures every functional requirement is systematically linked to its origin, be it a higher-level requirement, user need, or business objective. This traceability matrix provides clarity on how each requirement contributes to the overall goals of the system. Below is a concise one-page summary of the requirements traceability for Functional Requirements outlined in Section 3 of SRS.

|  |  |  |
| --- | --- | --- |
| **Req. ID** | **Origin/Source** | **Linked To** |
| **FR-001** | User Interaction | UI Design Specifications |
| **FR-002** | Message Handling | Communication Module Specifications |
| **FR-003** | Group Management | Group Management Module Specifications |
| **FR-004** | Security Requirements | Security Module Specifications |
| **FR-005** | Reporting and Analysis | Reporting Module Specifications |

Table 5.1-Requirements Traceability

**Detailed Explanation:**

**FR-001 (User Interaction):**

Source: Derived from user needs and usability studies.

Linked to: Corresponds to the detailed UI design specifications documented in the UI Design Specifications document.

**FR-002 (Message Handling):**

Source: Identified based on the necessity for efficient message processing.

Linked to: Directly tied to the specifications outlined in the Communication Module Specifications document.

**FR-003 (Group Management):**

Source: Originates from the requirement to manage groups effectively.

Linked to: Associated with the Group Management Module Specifications, detailing how the system will handle group-related functionalities.

**FR-004 (Security Compliance):**

Source: Originates from security and compliance standards.

Linked to: Directly associated with the Security Module Specifications, ensuring that the system adheres to necessary security measures.

**FR-005 (Reporting and Analytics):**

Source: Identified based on the need for reporting and analytics features.

Linked to: Corresponds to the Analytics Module Specifications, detailing how reporting and analytics functionalities will be implemented in the system.

**Traceability Insights:**

**Completeness:** Each functional requirement is linked to its specific source, ensuring that no requirement is overlooked or disconnected from its origin.

**Alignment with Design and Modules:** Traceability highlights the seamless connection between functional requirements and subsequent design specifications. This ensures that the designed system aligns precisely with user expectations.

**Modular Development:** The traceability matrix supports a modular development approach by clearly mapping each requirement to its relevant module. This enhances the understanding of how different components contribute to the fulfillment of user needs.

**Security Integration:** FR-004 emphasizes the integration of security measures within the system, ensuring compliance with industry standards and regulations.

**Performance Monitoring:** FR-005 focuses on reporting and analytics, linking to the Analytics Module Specifications to guarantee the implementation of robust reporting features and performance analytics.

This extended traceability matrix enhances the overall understanding of how the functional requirements align with specific modules, providing a comprehensive view of the development process and ensuring that all user needs are addressed through systematic and traceable design decisions.

# Appendix A: Definitions, Acronyms, and Abbreviations

This section describes any general information that helps to understand this software test document. This section contains an alphabetical listing of all acronyms, abbreviations, and their meanings as used in this document and a list of any terms and definitions needed to understand this document.

Table 3 Definitions, Acronyms, and Abbreviations

|  |  |
| --- | --- |
| **Term/Acronym** | **Definition/Description** |
| Functional Testing | Validates that each function of the software application operates as intended, ensuring all features meet specified requirements. |
| Performance Testing | The activity of testing the software as a whole to verify that its performance (e.g. response time) matches its SRS. |
| Requirement | A condition or capability needed by a user to solve a problem or achieve an objective. |
| Specification | A document that prescribes, in a complete, precise, verifiable manner, the requirements, design, behaviours or other characteristic of a system or system components. |
| Software Requirements Specification (SRS) | It is a technical oriented software requirements document. |
| Software Test Document (STD) | A document that describes test plans and test case specifications for testing the software. |
| System Testing | The testing of the software application as a whole to see the software works according to its specification. This can be further divided into functional testing and performance testing. |
| User Acceptance Testing (UAT) | Involves end-users testing the software to confirm it meets their needs and requirements before final deployment. |
| White-Box Testing | Examines the internal logic, structure, and code of the software to ensure it functions correctly at the programming level. |
| Black-Box Testing | Assesses the functionality of a software application without examining its internal code, focusing on inputs and outputs to validate the expected behavior. |
| Verification | Verification ensures that a product is designed and implemented correctly, confirming adherence to specified requirements. |
| Validation | Validation ensures the product meets the customer's needs by assessing its functionality and performance against user expectations. |

# Appendix B: Contributions

This section shall contain the names of all the contributors to this document. The sections that each contributor has worked on shall be described in the table below.

|  |  |
| --- | --- |
| **Contributor Name** | **Sections Worked On** |
| Gokul P | Document Template Development, Document Preparation(1 – Appendices),  Introduction, Requirements Traceability, Test Plan Review Document |
| Ranjith T | Functional Test plan, Test Cases and Specifications, review |
| Dinesh Kumar R | User Acceptance Test plan, Index, Formatting all tables, review |
| Vel Murugan T | Software Test plan, Review report |

Table 6-Students Contribution