

RISCORD MILESTONE 2

Jackson Wilson, Soumil Chhabra, Ronak Jain, Chinmay Arvind, Anitej Isaac Sharma

Project Overview

Riscord is a versatile communication platform, offering a seamless and feature-rich experience inspired by *Discord*. With Riscord, users can create and customize their own virtual communities, known as servers, with ease. From organizing discussions to facilitating various forms of communication through text and voice chat, Riscord empowers users to tailor their online interactions to their liking. With robust permissions management, users can ensure a secure environment within their communities. Additionally, Riscord enhances connectivity through friend lists, direct messaging, and media sharing, providing a dynamic and personalized experience for all users. After all, communication is key – and with Riscord, you can open the door to limitless connections and boundless communication possibilities.

Project Scope: A High-Level Description

Riscord aims to replicate the core features of *Discord*, offering users a synchronous communication platform where they can create and manage their own servers, establish channels for various forms of communication, and engage in activities such as text, voice, and video chat. For instance, users can create a server for photography enthusiasts dedicated to sharing photographs from around the world and building connections with fellow photography enthusiasts, where channels can be created for various photography styles and genres, and workshops and other live discussions can be hosted via the text, voice, and video chat features.

The platform also offers direct messaging capabilities, allowing users to privately communicate with friends or colleagues. Additionally, users can share multimedia content such as images, videos, stickers, GIFs, and emojis to enhance their communication experience. Riscord also provides users with moderation tools and customizable notifications, empowering each individual to shape their platform experience based on personal preferences in addition to having the ability to tailor the platform's user interface to their liking by selecting from a range of available themes.

Overall, **Riscord** aims to deliver an adaptable and engaging communication platform tailored to diverse user needs and preferences.

User Requirements

1. Users should be able to **create new servers, edit server settings, and delete servers**.
2. Users should be able to **create, rename, modify, and delete channels within servers**.
3. Users should have **access to live text and voice chat features** within servers.
4. Users should be able to **modify permissions within servers to assign roles and manage access levels**.
5. Users should **receive notifications for new events within servers and should be able to customize their notification preferences for each server**.
6. Users should be able to **send direct messages** to other users for private communication.
7. Users should have the ability to **join and leave multiple servers**.
8. Users should be able to **create an account for themselves and delete an account**.
9. Users should be able to **send friend requests to other users**.
10. Users should be able to **see the online/offline status of other users** within a server and with other users they are friends with.
11. Users should have the ability to **add media (images, videos, Stickers, GIFs, emojis, etc.) and embed links within a message** to another user or within a server.
12. Users should be able to **tag other users by user ID, and everyone within a server**.
13. Users should have the ability to **change application/server themes**.

Functional Requirements

1. Server Management

- a. Allow users to create, modify, and delete servers.
- b. *Acceptance Criteria:* Users should be able to initiate server setup, change server details, and remove a server without encountering errors.

2. Category Management

- a. Enable users to create, modify, and delete categories within servers.
- b. *Acceptance Criteria:* Users should have the ability to add new categories, rename existing categories, reorganize categories, and delete categories as needed.

3. Channel Management

- a. Allow users to create, modify, and delete text and voice channels within servers.
- b. *Acceptance Criteria:* Users should be able to add channels, adjust channel settings, and remove channels successfully.

4. Communication Tools

- a. Support live text and voice chat.
- b. *Acceptance Criteria:* Users should be able to engage in real-time communication through text and voice without experiencing noticeable lag or quality issues.

5. Permission Settings

- a. Allow server owners to modify user permissions.
- b. *Acceptance Criteria:* Server owners should be able to change permissions, and those changes should be reflected immediately for affected users.

6. Account Registration and Management

- a. Enable new users to register and existing users to modify or delete their accounts.
- b. *Acceptance Criteria:* Users should be able to register a new account, modify their account details, and delete their account as needed.

7. Server Membership

- a. Allow users to join multiple servers, leave servers, and switch between servers.
- b. *Acceptance Criteria:* Users should be able to seamlessly join or leave servers and switch the active server view without encountering system errors.

8. Social Interaction

- a. Provide a feature for users to send friend requests, manage a friends list, and display online/offline statuses.
- b. *Acceptance Criteria:* Users should be able to send friend requests, accept or decline friend requests, manage their friends list, and view the online/offline status of other users.

9. Direct Messaging

- a. Offer a direct messaging feature for private user communication.
- b. *Acceptance Criteria:* Users should be able to send, receive, and view direct messages with other users in real time.

10. Media and Link Sharing

- a. Allow users to add media and embed links in their messages.
- b. *Acceptance Criteria:* Users should be able to upload and share media files and links, which should be accessible to others in the chat.

11. Notifications

- a. Provide customizable notifications for various user activities.
- b. *Acceptance Criteria:* Users should receive notifications according to their settings and have the ability to customize what they are notified about.

12. Tagging and Mentioning

- a. Support the functionality to tag or mention users in chats.
- b. *Acceptance Criteria:* When a user is tagged, they should receive a notification, and the tag should be visible in the chat.

13. Customization

- a. Allow users to change server themes.
- b. *Acceptance Criteria:* Users should be able to change the theme of a server and see the visual changes on the user interface.

Non-Functional Requirements

1. Performance

- a. Efficiently handle user traffic to ensure smooth operation.
- b. *Acceptance Criteria:* The system should maintain responsiveness and stability even when several users are accessing various features simultaneously.

2. Usability

- a. Ensure that the platform is adaptable and easy to use for all users.
- b. *Acceptance Criteria:* Users should be able to navigate the system effortlessly or at least perform basic tasks without requiring guidance or instructions.

3. Reliability

- a. Maintain system stability and prevent crashes or failures.
- b. *Acceptance Criteria:* The system should operate without disruption during demonstrations, and even if there are disruptions, it should be able to recover from them easily.

4. Scalability

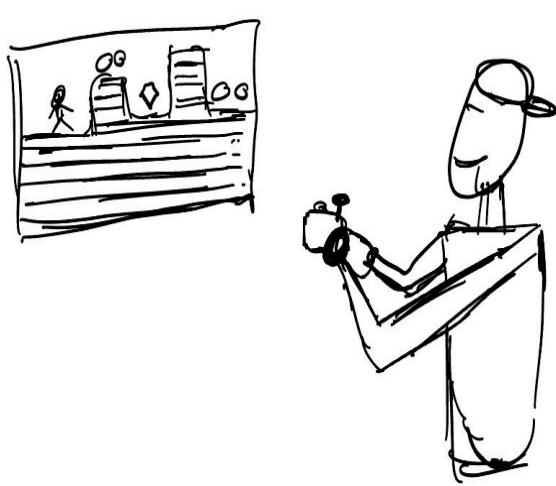
- a. Ensure that the system can accommodate increase in users and user engagement without having to manually tailor the platform each time there is a growth.
- b. *Acceptance Criteria:* The system should be able to add additional mock users, servers, categories, channels, etc. without the need for manual adjustments in the database or the platform's code base.

5. Security

- a. Protection of user data and privacy.

- b. *Acceptance Criteria:* Basic security measures such as hashed passwords should be implemented to protect user accounts and other sensitive information.
- 6. Compatibility**
- a. Ensure that the functionality of the platform remains consistent across different browsers and environments
 - b. *Acceptance Criteria:* The system should perform as expected on Chrome, Firefox, Safari, etc. during demonstrations to showcase compatibility with commonly used web browsers.

Proto-Personas: Visualizing User Interaction

<p style="text-align: center;">Alex Johnson</p> 	<p>Age: 21</p> <p>Gender: Male</p> <p>Occupation: Student (specializing in Computer Science)</p> <p>Marital Status: Single</p> <p>Interests: Video game testing and development, participating in local software competitions, playing competitive video games with friends, and listening to music.</p> <p>Personality: Ambivert, tech-savvy, competitive, and goal-oriented.</p> <p>Goals: To stay connected with friends while enjoying video games, find a platform facilitating easy communication with gaming buddies, organize gaming sessions, share gaming-related media, and enjoy listening to music together during activities.</p> <p>Tasks:</p>
---	---



- Establish and manage gaming servers for various games played with friends.
- Organize channels within servers to categorize them based on different games or gaming sessions.
- Utilize live text and voice chat features during gaming sessions.
- Adjust permissions within servers to assign roles like moderators for managing chats during gaming.
- Stay updated on friends' online/offline status to determine availability for gaming.
- Directly message friends for quick coordination or to share gaming-related content.
- Customize notifications to receive alerts for crucial messages or mentions during gaming sessions.
- Listen to music together with friends during gaming sessions or other activities.

Physical Environment: Alex primarily spends his time in his dorm room, equipped with a powerful PC and high-speed internet for gaming.

Social Environment: He shares a close bond with a group of friends, engaging in regular gaming sessions and online communication.

Technological Environment: Proficient in various gaming platforms and communication tools, Alex prefers platforms seamlessly integrating with gaming activities.

Maya Patel



Age: 35

Gender: Female

Occupation: Freelance Graphic Designer and Small Business Owner

Marital Status: Married with two children

Interests: Graphic design, entrepreneurship, cooking, yoga, and DIY home projects.

Personality: Organized, creative, entrepreneurial, nurturing, and community-oriented.

Goals: Maya seeks a platform where she can network with other professionals in her industry, showcase her design portfolio, collaborate on projects, and gain inspiration for her creative endeavors. She desires a platform that offers a supportive community, valuable resources, and opportunities for professional growth.

Tasks:

- Create and manage servers dedicated to graphic design, entrepreneurship, and other areas of interest.
- Customize channels within servers to organize discussions, share resources, and collaborate on projects effectively.
- Utilize live text and voice chat for networking, brainstorming, and seeking advice from peers.
- Adjust permissions to ensure a respectful and constructive environment within professional



channels.

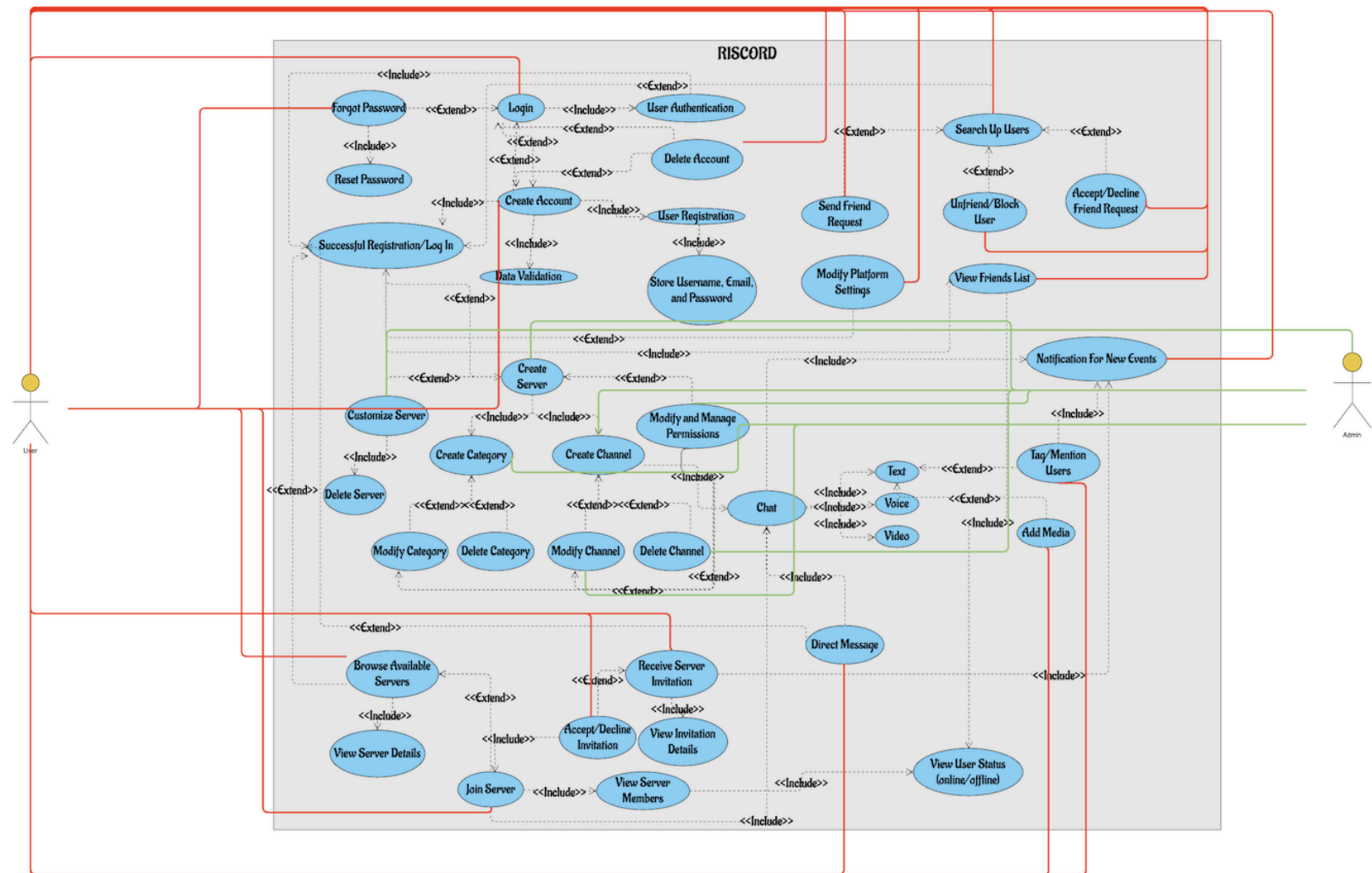
- Stay updated on industry trends, events, and opportunities by engaging with relevant channels and communities.
- Direct message fellow designers, clients, and collaborators for project discussions, feedback, and networking.
- Customize notifications to receive alerts for job postings, industry news, and important discussions within professional communities.

Physical Environment: Maya works from her home office, which is equipped with a dedicated workspace, high-speed internet, and essential design tools.

Social Environment: Actively engaged in online design communities, professional forums, and local business networks, Maya interacts with fellow designers, clients, and industry professionals to build connections and grow her business.

Technological Environment: Proficient in various design software and online collaboration tools, Maya values platforms that offer intuitive interfaces, seamless communication features, and robust networking capabilities to support her professional endeavors effectively.

Use-Case Diagram



The use case diagram details exactly how a user (non-admin) can use the diagram by lines connecting the user to the features they have access to within the Riscord web application. The same goes for the server admin, who is also a user, and whose access to features is in addition to the features that a regular user has access to (the server admin inherits access to the features that a regular user has access to). The special features made for server admins are indicated with green lines connecting the server admin to the features they have access to, which are exclusive to them and not available for just regular users of the Riscord web application.

In-depth description of features:

- *Creating channel within a server*
 - Initial Assumption: A user can create a channel within a server and assign a channel type and channel name
 - Normal Flow: User logs into account with admin privileges in the corresponding server, navigates to server settings, and creates a channel with desired inputs
 - What Can Go Wrong: User can name a channel something other than what they wanted, user can assign wrong channel type
 - Other Activities:
 - System State on Completion: Server contains new channel with set attributes
- *Modifying/Deleting channel within a server*
 - Initial Assumption: A user with admin privileges can modify a channel by deleting, modifying attributes, or changing permissions.
 - Normal Flow: User logs into account with admin privileges in the corresponding server, navigates to channel setting, changes desired attributes
 - What Can Go Wrong: Channel may be accidentally deleted
 - Other Activities:
 - System State on Completion: Server contains existing channel with new set of attributes
- *Text/Voice chatting channel within a server (with the ability to tag/mention users and add media such as images, videos, gifs, etc.)*
 - Initial Assumption: A user with access to the server and relevant channels can enter a text/voice chatroom and interact within
 - Normal Flow: User logs into account with access to relevant channels, enters the server, and begins interacting
 - What Can Go Wrong: Poor connectivity may cause issues within chatroom
 - Other Activities:
 - System State on Completion: User exists within channel while interacting with others within the channel

- *Viewing member list of each server*
 - Initial Assumption: A user with access to the server can see a list of total and active users within the server
 - Normal Flow: User enters server and can see list of users
 - What Can Go Wrong: Active status may be incorrect
 - Other Activities:
 - System State on Completion: User sees a list of all users on the same server
- *Leaving/Joining a server*
 - Initial Assumption: A user with access to the server can leave/join said server at will
 - Normal Flow: User enters server, navigates to system settings, clicks leave server. OR User receives invite and accepts
 - What Can Go Wrong: Invite may be expired
 - Other Activities: User can leave multiple servers simultaneously
 - System State on Completion: User gains/loses a server
- *Friending/Unfriending users*
 - Initial Assumption: A user can send/receive/decline friend requests as well as managing existing friends
 - Normal Flow: User enters friends tab, navigates to pending invites or friends list, accepts or removes friends as needed
 - What Can Go Wrong: Friend may be accidentally deleted
 - Other Activities: User can chat with friends
 - System State on Completion: User gains/loses a friend

- *Viewing offline/online status of other users within a server and whom a user is friends with*
 - Initial Assumption: A user can login to a server and view the status of all other members
 - Normal Flow: User enters server, views users tab, and is shown various statuses
 - What Can Go Wrong: Status shown may be incorrect
 - Other Activities:
 - System State on Completion: User sees tab containing statuses of server members
- *Directly messaging users from a server or a user that the current user is friends with*
 - Initial Assumption: User can view the profile of another user and directly message them
 - Normal Flow: User opens friends page, clicks a messaging button, types message for friend
 - What Can Go Wrong:
 - Other Activities: User can send gifs, images, files
 - System State on Completion: User is in a chatroom with friend
- *Receiving notifications about joining servers, and messages received*
 - Initial Assumption: User receives notifications based on preferences. Notifications can include DMs, server pings, or friend requests
 - Normal Flow: User receives a notification on mobile or desktop when one is triggered
 - What Can Go Wrong: User may receive unwanted pings when @everyone is used
 - Other Activities:
 - System State on Completion: Notification is sent to user

- *Modifying server permissions*
 - Initial Assumption: User with admin privileges goes to server settings and modifies permissions on a user to user basis
 - Normal Flow: Admin navigates to server settings, navigates to the relevant section, and modifies a user's permissions within the server
 - What Can Go Wrong: Admin may accidentally give a user dangerous permissions such as the ability to delete channels
 - Other Activities:
 - System State on Completion: Admin reaches a screen where they can modify a user's permissions in the server
- *Viewing friends list*
 - Initial Assumption: User can navigate to a page with a list of their friends, all of which have interactable profiles
 - Normal Flow: User navigates to friends page and is met with a list of icons representing all friends
 - What Can Go Wrong:
 - Other Activities: User can DM, add, or remove friends
 - System State on Completion: User is met with a list containing all friends
- *Creating a server*
 - Initial Assumption: User can create a server where they are automatically granted admin privileges
 - Normal Flow: User clicks on the 'new server' icon and is prompted with a screen asking for basic information. Upon completion a new server is added into their server list and they're given admin privilege
 - What Can Go Wrong:
 - Other Activities:
 - System State on Completion: User has a new server in their list as well as admin privileges for that specific one

- *Deleting a server*

- This feature enables a server admin to be able to delete a server, by opening the server settings page and verifying the user's identity before deleting the server's contents, and all the content on the server, and removing all users in the server from the server.
- Initial Assumption: User with admin privileges for a server can navigate to the server settings to delete it
- Normal Flow: Admin navigates to server settings, clicks 'delete server' and confirms on an 'are you sure' screen
- What Can Go Wrong: User may delete the wrong server by accident
- Other Activities:
- System State on Completion: Server is removed from the server list of all users within the server

- *Creating/deleting an account*

- Initial Assumption: New users are prompted to create an account and existing users can create alternate accounts
- Normal Flow: New user opens app, is prompted with a create account option, and upon creating can navigate to the profile tab to create alternate accounts using the same create account option. Within the profile settings users can also choose to delete the account, given a confirmation box before it is completed
- What Can Go Wrong:
- Other Activities:
- System State on Completion: User has a profile and the option to navigate to alternate profiles

User Stories

Note: The following user stories are also included in the Github backlog of the project.

- As a server admin, I want to be able to create, modify, and delete a server so that I can communicate with other friends in ways that are customizable, or to end communication with other users.

Acceptance Criteria:

- Server admin should have access to options to create a new server, modify server details, and delete a server.
 - When creating a new server, the admin should be prompted to provide a server name and optional profile picture.
 - Upon deletion of a server, all server data, including channels and messages, should be permanently removed.
 - Changes made to server details (name, profile picture) should be immediately reflected across the platform.
- As a server admin, I want to be able to create, modify, and delete a channel in a server so that I can effectively categorize interactions between users in a server or remove categories of communication that serve no purpose.

Acceptance Criteria:

- Server admin should be able to create a new channel or category within a server, modify existing channel settings, and delete channels.
 - When creating a new channel or category, the admin should specify the channel or category name and type (text, voice).
 - Modifications to channel or category settings should include options to change the channel or category name, type, and permissions.
 - Deleted channels or categories should be removed from the server interface, and all associated messages should be permanently deleted.
- As a server admin, I want to be able to change server themes so that I can let the server members have a different visual when they open the server.

Acceptance Criteria:

- Server admin should have the ability to change the theme of the server interface.
 - Themes should include options for color schemes, background images, and other visual elements.
 - Changes to server themes should be immediately visible to all server members upon application.
- As a regular user, I want to be able to register as a user and have a personal account so that I can access the app's features in its entirety.

Acceptance Criteria:

- New users should be able to register by providing required information such as username, email, and password.
 - Upon successful registration, users should receive a confirmation email or message.
 - Registered users should have access to all features of the application upon logging in.
- As a regular user, I want to be able to modify and delete my account so that I can stop using the application.

Acceptance Criteria:

- Users should have access to settings allowing them to modify their account details.
 - Account deletion option should be available in the user settings menu.
 - Upon deletion, the user's account and all associated data should be permanently removed from the system.
- As a regular user, I want to be able to send friend requests to other users so that I can communicate with them effectively.

Acceptance Criteria:

- Users should be able to send friend requests to other users.

- The recipient should receive a notification of the friend request and have the option to accept or decline it.
 - Upon acceptance, the users should be added to each other's friends list.
- As a regular user, I want to be able to see my friends' list so that I can communicate with them when needed.

Acceptance Criteria:

- Users should have access to a friends list displaying all accepted friend connections.
 - The friends list should include information such as friend usernames and online/offline status.
 - Users should be able to access their friends list from the application's main interface.
- As a regular user, I want to be able to modify permissions on a server if given a role so that I can enforce regulations on a server.

Acceptance Criteria:

- Users with appropriate permissions (e.g., server admins, moderators) should be able to modify user permissions within a server.
 - Changes to permissions should be immediately reflected for affected users.
 - Options for modifying permissions should be accessible from the server settings menu.
- As a regular user, I want to be able to search for other users by username so that I can send friend requests to other users.

Acceptance Criteria:

- Users should have access to a search feature allowing them to find other users by their usernames.
 - Search results should display matching usernames, and users should be able to click on a result to view the user profile.

- As a regular user, I want to be able to access online/offline status along with the member list for each server so that I can see which user is online and to communicate or send friend requests to users on the server.

Acceptance Criteria:

- Users should be able to view the online/offline status of other users within a server.
 - The member list of each server should display usernames along with their online/offline status.
 - Users should be able to access the member list from the server interface.
- As a regular user, I want to be able to directly message a user by their username so that I can communicate with a user from a server or just by username.

Acceptance Criteria:

- Users should have the ability to send direct messages to other users by entering their usernames.
 - Direct messages should be delivered in real-time, and users should receive notifications for new messages.
- As a regular user, I want to be able to attach media such as images, videos, stickers, GIFs, emojis, etc., and embed links so that I can send media to users in a server or via direct messaging.

Acceptance Criteria:

- Users should be able to attach various types of media (images, videos, stickers, GIFs, emojis) to messages.
 - Links pasted into messages should be automatically embedded with a preview.
 - Media and links should be accessible and viewable by recipients in the chat.
- As a regular user, I want to be able to receive push notifications to users involved in a server where a message was sent or via direct messaging so that I can see the messages sent as they reach the server or the direct messaging chat.

Acceptance Criteria:

- Users should receive push notifications for new messages in servers and direct messages.
 - Notifications should be delivered in real-time and should include information about the sender and message content.
- As a regular user, I want to be able to customize notification frequency so that I can make sure I get notifications as and when I want to see them.

Acceptance Criteria:

- Users should have access to settings allowing them to customize notification preferences.
 - Options should include frequency settings for different types of notifications (e.g., message mentions, friend requests).
 - Changes to notification settings should be applied immediately.
- As a regular user, I want to be able to tag/mention server members using @ so that I can talk to specific users or all the users in a server.

Acceptance Criteria:

- Users should be able to tag or mention other users in messages using the "@" symbol followed by the username.
 - Tagged users should receive notifications, and their usernames should be highlighted in the message.
- As a regular user, I want to be able to join and leave multiple servers so that I can communicate with a community of users or stop communicating with a community of users.

Acceptance Criteria:

- Users should have the ability to join multiple servers and leave servers they no longer wish to participate in.

- Joining a server should be initiated by accepting an invitation or searching for and selecting a server from a list.
 - Leaving a server should remove the user from the server's member list and channels.
- As a regular user, I want to be able to communicate via text and voice so that I can communicate effectively with other users in servers and via direct message.

Acceptance Criteria:

- Users should have access to both text and voice communication features within servers and direct messages.
- Text chat should support real-time messaging, while voice chat should provide clear audio communication.
- Both text and voice chat features should be accessible from the server interface and direct message windows.