REAL-TIME CHAT APPLICATION

An Overview of Modules and UML Diagrams

Presented by: Group-11
2310030188 – Jashwanth Reddy
2310030421 – Guru Charan
2310030163 – Abhinav Reddy

INTRODUCTION :

A real-time chat application enables users to exchange messages instantly over a network, making communication faster and more interactive. The key features of a real-time chat application are:

Instant Messaging: Messages are delivered almost instantly from one user to another without delay. This is essential for providing an efficient and seamless communication experience, especially in business or customer service contexts.

Communication in Personal & Professional Settings: These apps are crucial in both informal settings, such as chatting with friends or family, and in more formal contexts, like business communications. Examples like Slack and Microsoft Teams illustrate this well.

TECHNOLOGIES USED:

Real-time chat relies on technologies like:

Web Sockets: A protocol that allows persistent, two-way communication between the client and server, making real-time message delivery possible.

Push Notifications: Used to notify users of new messages even when they aren't actively using the app.

Benefits:

Faster Communication: Real-time messaging eliminates the wait time typically involved in other communication forms (e.g., email).

Efficient Collaboration: Teams can discuss issues, exchange information, and collaborate in real time, improving productivity.

User Engagement: Instant responses and notifications keep users engaged and active in the app.

MODULES OF A REAL-TIME CHAT APPLICATION :

USER MANAGEMENT MODULE: HANDLES USER REGISTRATION, AUTHENTICATION, AND PROFILE MANAGEMENT.

CHAT MODULE: SUPPORTS ONE-ON-ONE AND GROUP MESSAGING, ENSURING SMOOTH COMMUNICATION.

MESSAGE HANDLING MODULE: MANAGES REAL-TIME MESSAGING, MESSAGE STORAGE, AND SYNCHRONIZATION.

NOTIFICATION MODULE: SENDS PUSH NOTIFICATIONS AND EMAIL ALERTS FOR USER ENGAGEMENT.

SECURITY MODULE: IMPLEMENTS ENCRYPTION, ACCESS CONTROL, AND USER PRIVACY PROTECTION.

MULTIMEDIA MODULE: ENABLES FILE SHARING, MEDIA STORAGE, AND VIEWING CAPABILITIES.

CLASS DIAGRAM:

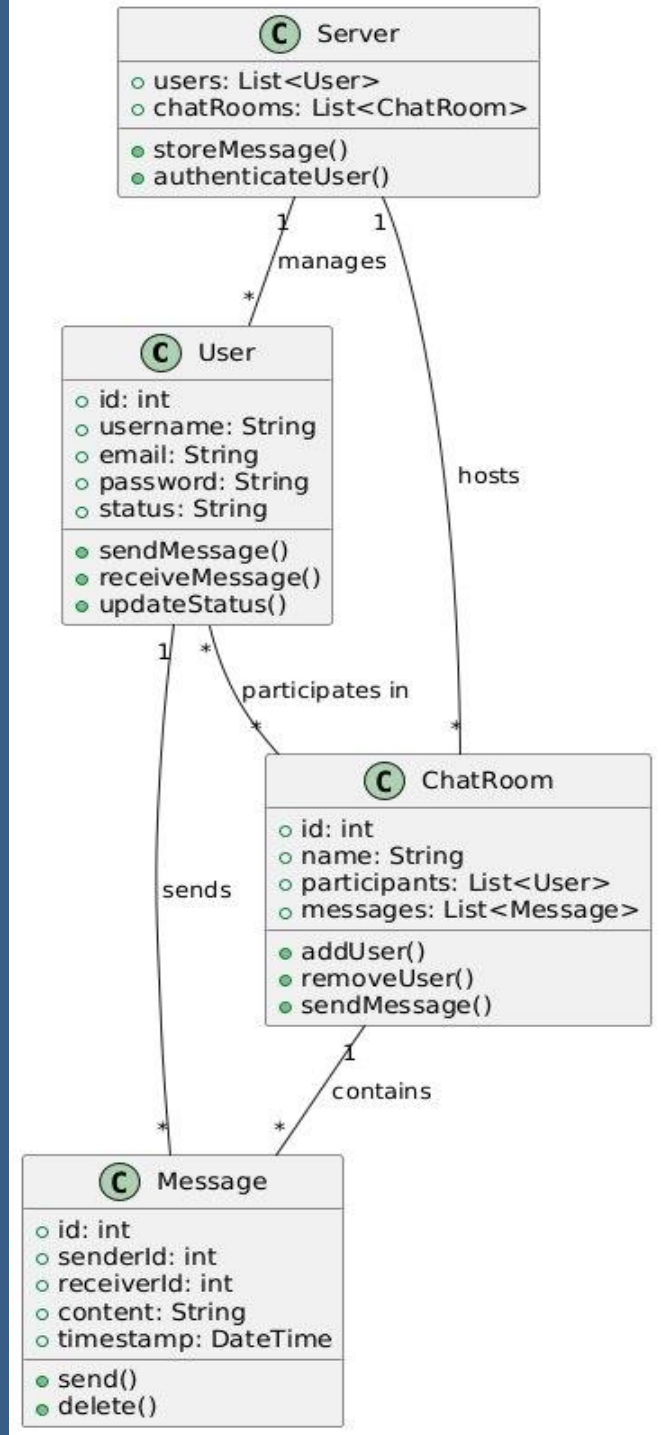
Key classes include:

Server: to store the data.

User: Attributes like username, password, and profile.

Chat Room: Represents conversations between users.

Message: Stores message content, timestamps, and sender/receiver details.



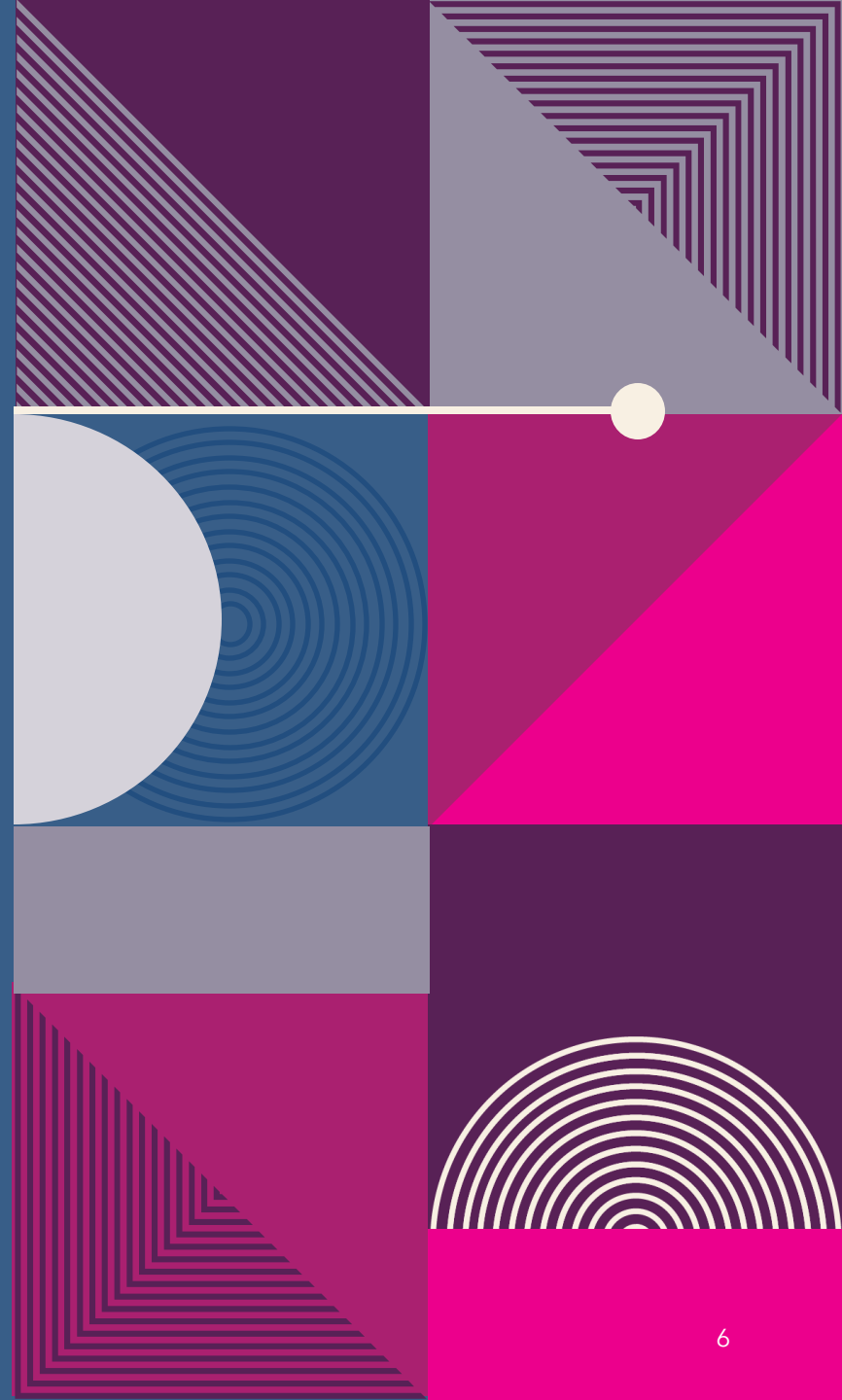
ENGAGING THE AUDIENCE

Make eye contact with your audience to create a sense of intimacy and involvement

Weave relatable stories into your presentation using narratives that make your message memorable and impactful

Encourage questions and provide thoughtful responses to enhance audience participation

Use live polls or surveys to gather audience opinions, promoting engagement and making sure the audience feel involved



USE CASE DIAGRAM

System Functionalities & User Interactions

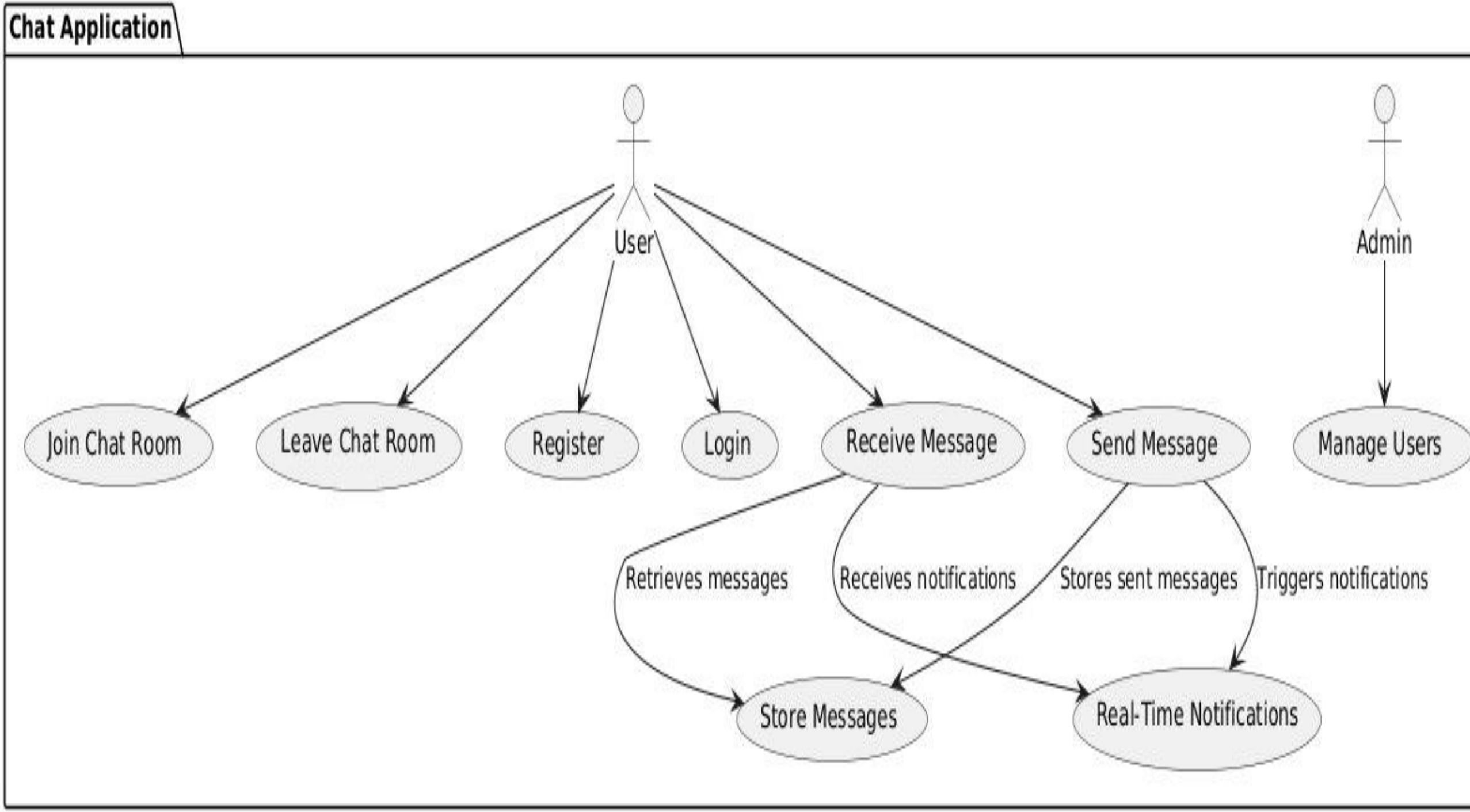
This diagram defines the scope and interactions within the real-time chat application, highlighting the key use cases and actors involved.

Actors:

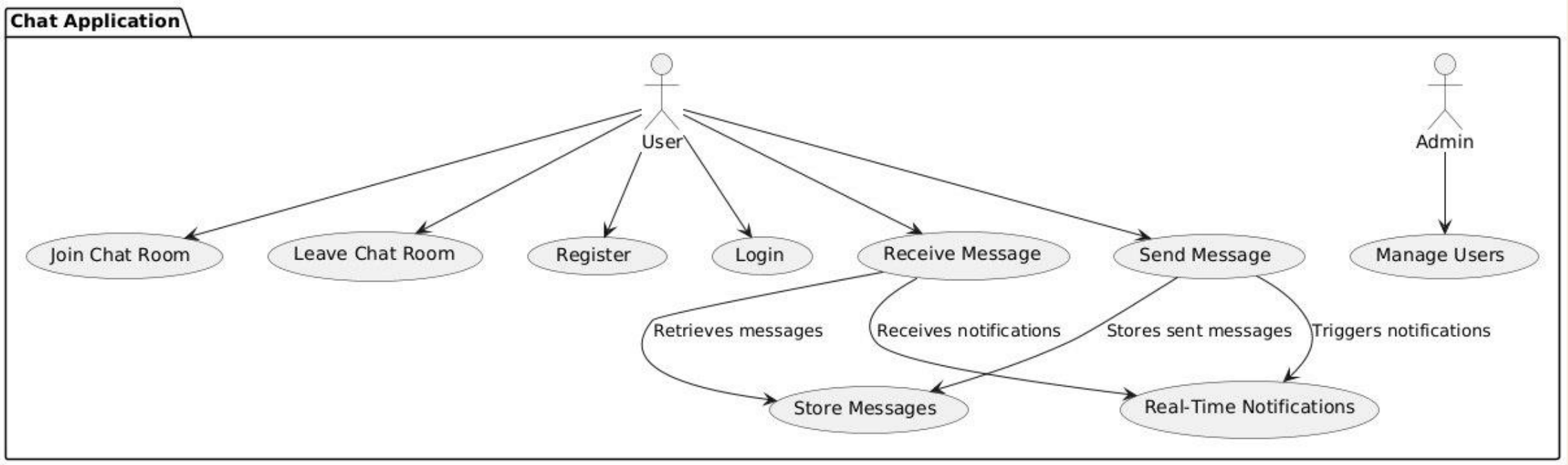
- **User**: Sends/receives messages, adds/blocks contacts, uploads/downloads media, and receives notifications.
- **Admin**: Manages user contacts, oversees media uploads, handles notifications, and manages users.
- **System**: Handles message processing, media handling, and notification delivery.

Key Use Cases:

- **Send & Receive Messages**: Real-time communication between users.
- **Add/Block Contacts**: Manage who users interact with.
- **Upload & Download Media**: Share and retrieve media like photos and videos.
- **Receive Notifications**: Notify users of new messages, media, or events.


Chat Application
User
Admin
Join Chat Room
Leave Chat Room
Register
Login
Receive Message
Send Message
Manage Users
Retrieves messages
Receives notifications
Stores sent messages
Triggers notifications
Store Messages
Real-Time Notifications

STATE CHART DIAGRAM



Represents the states and transitions of objects within the system.

Message states include: Draft: Message is being composed.

Sent: Message is sent but not yet delivered.

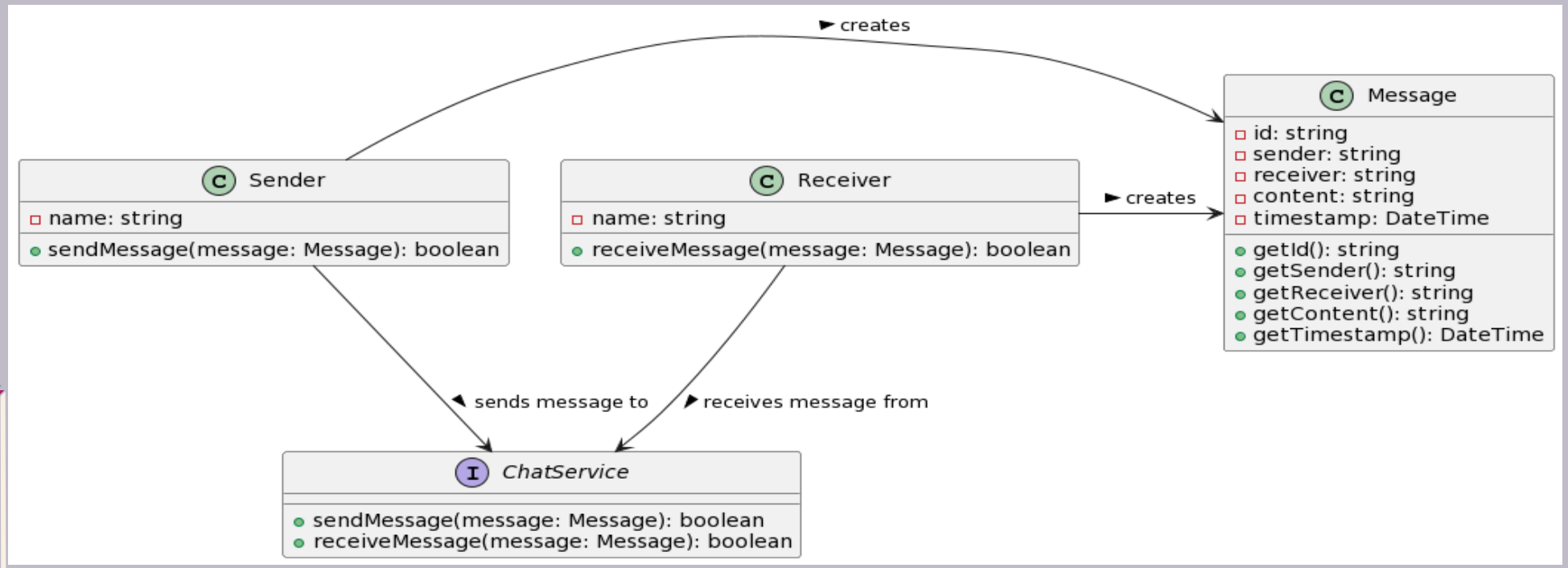
Delivered: Message reached the recipient.

Read: Recipient has seen the message.

Deleted: Message is removed from the system.

Shows object lifecycle and transitions between states.

INTERACTION DIAGRAM



Interaction Diagram Represents how objects communicate within the system.

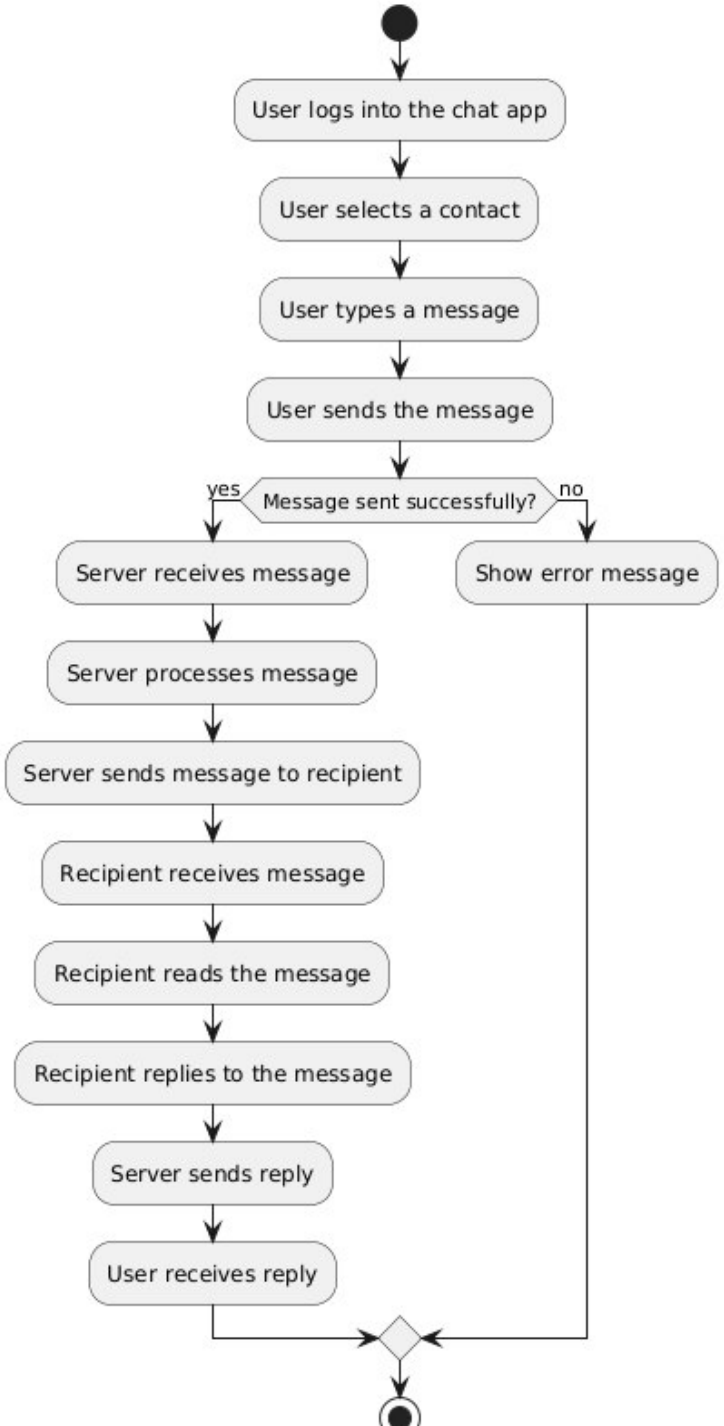
Sequence diagram example:

User sends a message.

Message is processed by the system.

Recipient receives the message.

Defines message flow, order of interactions, and data exchange.



ACTIVITY DIAGRAM:

Represents the workflow and processes of the system.

Example: Sending and receiving a message.

User composes a message.

Message is sent via the server.

Server routes the message to the recipient.

Recipient receives and reads the message.

Defines the flow of activities in a step-by-step manner.

CONCLUSION

- A real-time chat application is essential for seamless communication.
- Modules such as chat, message handling, notifications, and security enhance system functionality.
- UML diagrams provide a structured approach to designing scalable chat applications.
- Future improvements may include AI-powered chatbots, voice/video integration, and enhanced security features.

THANK YOU