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.

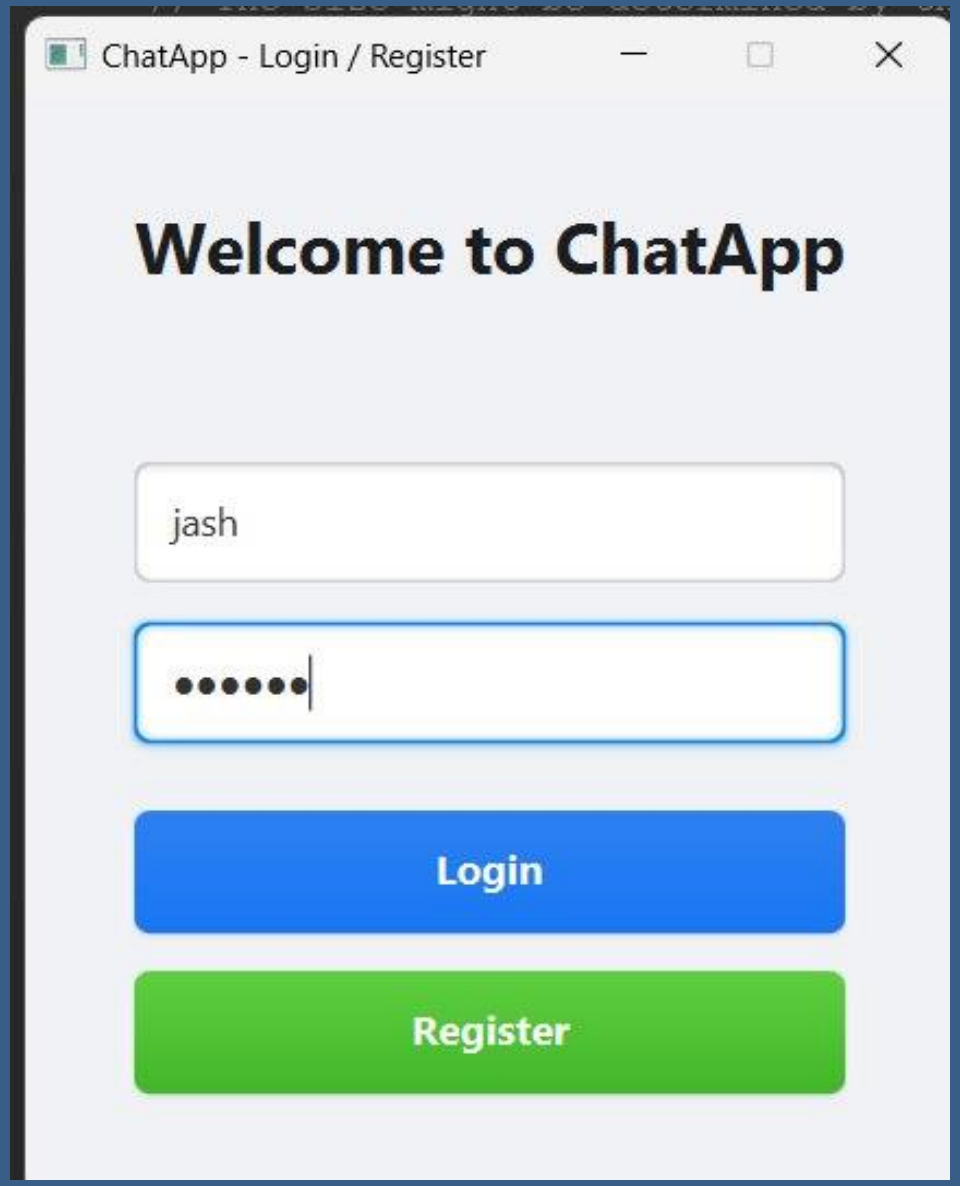
MESSAGE HANDLING MODULE: MANAGES REAL-TIME MESSAGING, MESSAGE STORAGE, AND SYNCHRONIZATION.

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

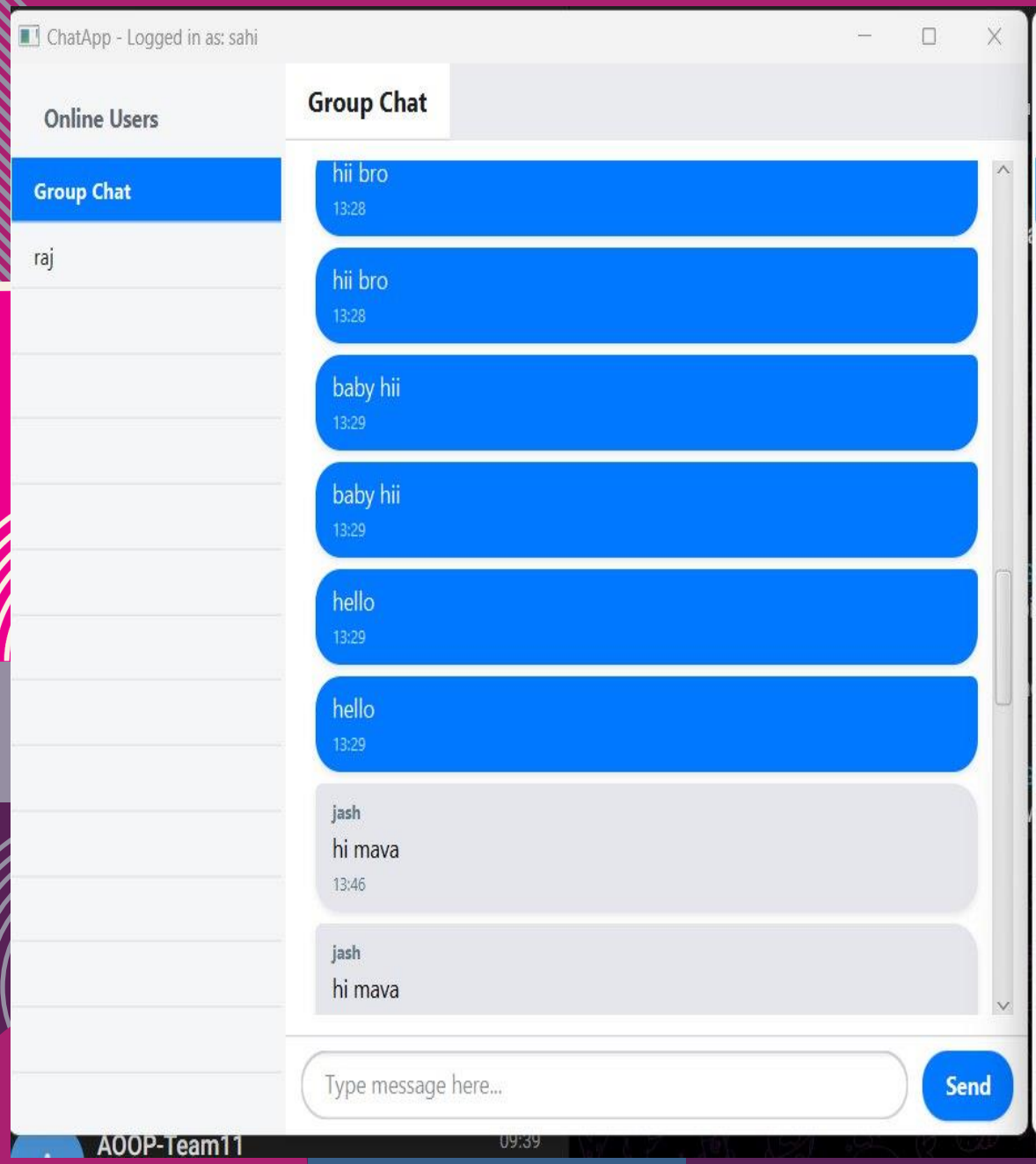
SECURITY MODULE: IMPLEMENTS ENCRYPTION, ACCESS CONTROL, AND USER PRIVACY PROTECTION.

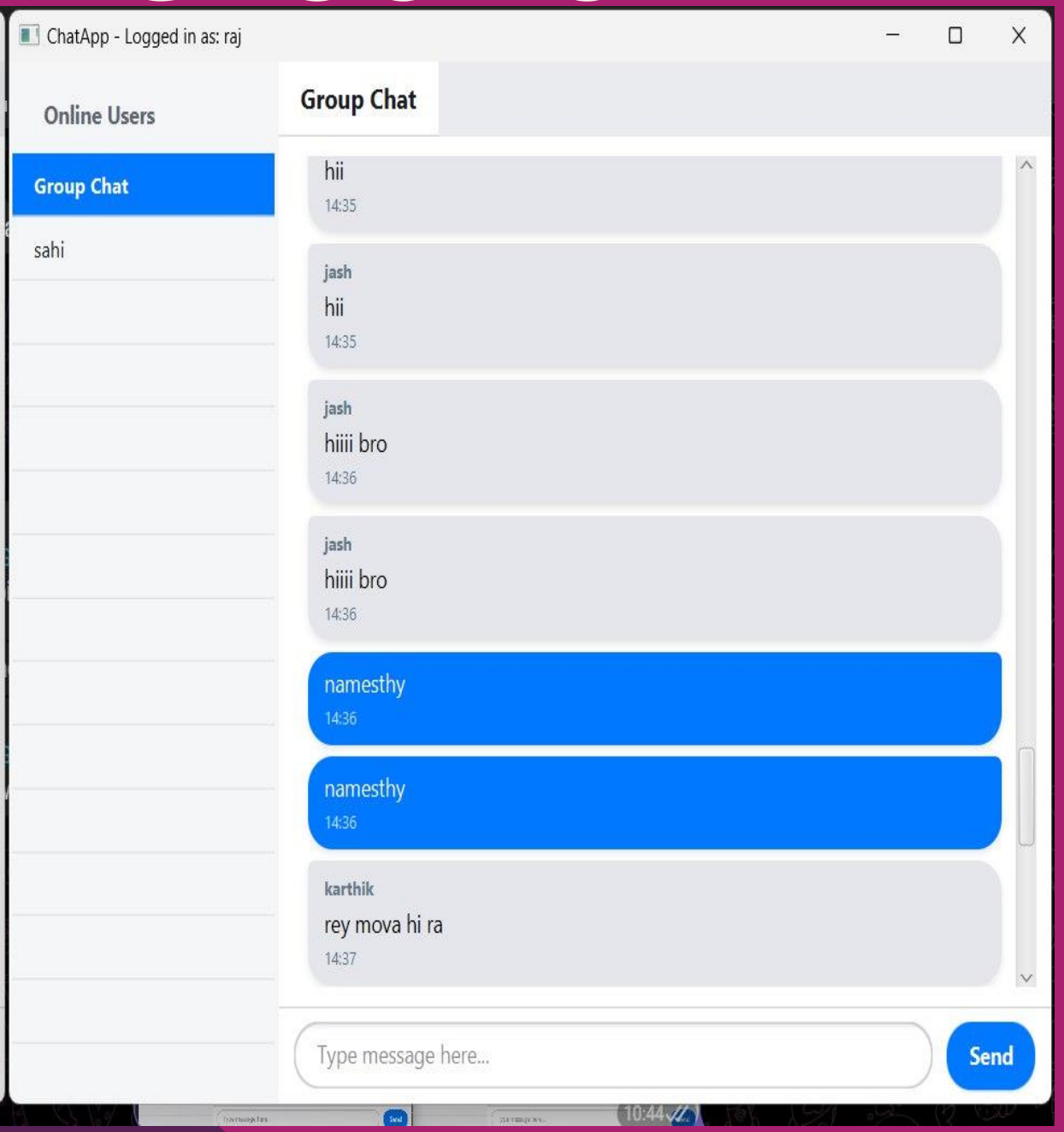
MULTIMEDIA MODULE: ENABLES FILE SHARING, MEDIA STORAGE, AND VIEWING CAPABILITIES.

OVERVIEW :

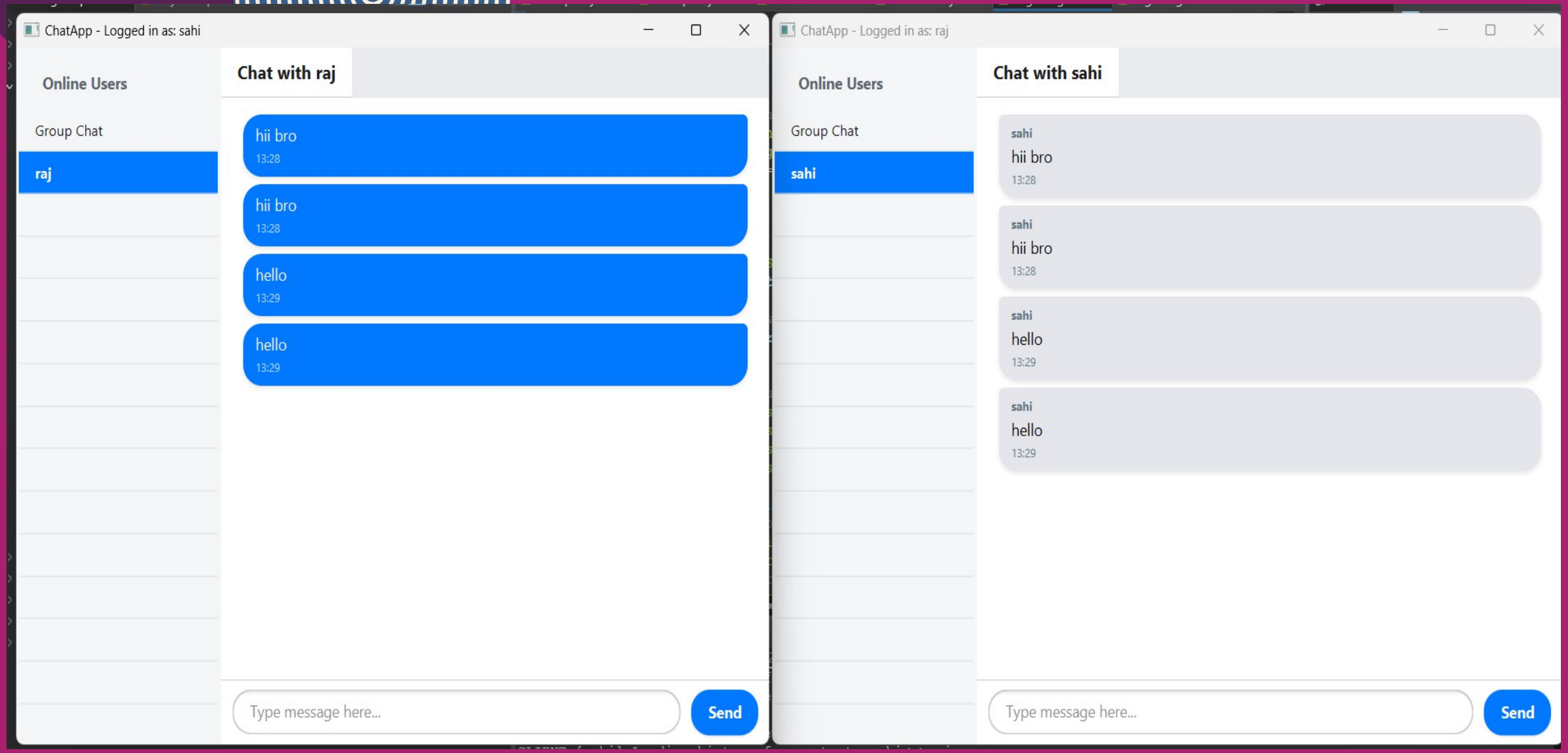

ChatApp - Login / Register
Welcome to ChatApp
jash
Login
Register

GROUP CHAT:


ChatApp - Logged in as: sahi
Online Users
Group Chat
raj
Group Chat
hii bro
13:28
hii bro
13:28
baby hii
13:29
baby hii
13:29
hello
13:29
hello
13:29
jash
hi mava
13:46
jash
hi mava
Type message here...
Send


ChatApp - Logged in as: raj
Online Users
Group Chat
sahi
Group Chat
hii
14:35
jash
hii
14:35
jash
hiiii bro
14:36
jash
hiiii bro
14:36
namesthy
14:36
namesthy
14:36
karthik
rey mova hi ra
14:37
Type message here...
Send

PERSONAL CHAT:



LOGIN PAGE CODE

example2.java | example3.java | ClientHandle... | ChatServer.java | LoginRegiste... | LoginRegiste...

```java
package com.chatapp.gui;

import javafx.application.Application;

/**
 * Main application class. Sets up the initial stage and loads the
 * login/register screen from FXML.
 */
public class LoginRegisterApp extends Application {

    @Override
    public void start(Stage primaryStage) {
        try {
            // --- Load the FXML file ---
            // Define the path relative to the current class's package
            String fxmlPath = "login-register.fxml";
            URL fxmlUrl = getClass().getResource(fxmlPath);

            // Check if the FXML file was found
            if (fxmlUrl == null) {
                System.err.println("CRITICAL ERROR: Cannot find FXML file at path: " + f
                System.err.println("Ensure '" + fxmlPath + "' is in the same package as
                                   "or check your build configuration (e.g., Maven/Gradl

                // Fallback attempt using ClassLoader (requires full path from classpath
                String absolutePath = "com/chatapp/gui/" + fxmlPath;
                System.err.println("Attempting fallback load via ClassLoader: " + absolu
                fxmlUrl = getClass().getClassLoader().getResource(absolutePath);

                if (fxmlUrl == null) {
                    // Throw an exception if still not found
                    throw new IOException("FXML file '" + fxmlPath + "' not found via ge
                } else {
                    System.out.println("Loaded FXML via ClassLoader path successfully.")
                }
            } else {
                System.out.println("Loading FXML using getResource() from: " + fxmlUrl);
            }
```

```java
            FXMLLoader loader = new FXMLLoader(fxmlUrl);
            Parent root = loader.load(); // Loads the VBox defined in FXML

            // --- Create the Scene ---
            // The size might be determined by the FXML's prefWidth/prefHeight,
            // or you can set it explicitly here if needed.
            Scene scene = new Scene(root);

            // --- Load the CSS ---
            String cssPath = "login-styles.css";
            URL cssUrl = getClass().getResource(cssPath); // Try relative path first
             if (cssUrl == null) {
                 cssUrl = getClass().getClassLoader().getResource("com/chatapp/gui/" + c
             }

            if (cssUrl != null) {
                scene.getStylesheets().add(cssUrl.toExternalForm());
                System.out.println("Login CSS loaded successfully from: " + cssUrl);
            } else {
                // Log a warning if CSS is missing, but don't stop the app
                System.err.println("Warning: Could not load CSS file 'login-styles.css'.
            }

            // --- Configure and Show the Stage ---
            primaryStage.setTitle("ChatApp - Login / Register");
            primaryStage.setScene(scene);
            primaryStage.setResizable(false); // Login window is typically not resizable
            primaryStage.show();

        } catch (IOException e) {
            // Handle errors loading FXML (critical)
            System.err.println("Failed to load application UI (FXML): " + e.getMessage()
            e.printStackTrace();
            // Show a user-friendly error dialog
            showErrorDialog("Application Load Error",
                            "Failed to load the application interface.",
                            "Could not load the main screen (login-register.fxml).\n" +
                            "Please check the application files and installation.\nEror
```

```java
                                "Please check the application files and installation.\nErro
        } catch (Exception e) {
            // Catch any other unexpected errors during startup
            System.err.println("An unexpected error occurred during application startup:
            e.printStackTrace();
             showErrorDialog("Unexpected Startup Error",
                                "An unexpected error occurred during startup.",
                                e.getMessage());
        }
    }

     /**
      * Helper method to display an error alert dialog.
      * Useful if JavaFX is partially initialized but FXML loading fails.
      * @param title Dialog window title.
      * @param header Dialog header text.
      * @param content Dialog main content message.
      */
     private void showErrorDialog(String title, String header, String content) {
        Alert alert = new Alert(Alert.AlertType.ERROR);
        alert.setTitle(title);
        alert.setHeaderText(header);
        alert.setContentText(content);
        alert.showAndWait(); // Show the dialog and wait for the user to close it
     }


    /**
     * The main entry point for the Java application.
     * Launches the JavaFX application.
     * @param args Command line arguments (not used in this application).
     */
    public static void main(String[] args) {
        System.out.println("Starting ChatApp JavaFX application...");
        // Any pre-launch setup could go here (e.g., logging config)
        launch(args); // Starts the JavaFX lifecycle (calls init(), start())
    }
}
```

BACKEND :

Console × Terminal Coverage

LoginRegisterApp [Java Application] C:\Users\allam\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.10.v20240120-1143\jre\bin\javaw.exe (26-Apr-2025, 10:45:30 am)

```
Starting ChatApp JavaFX application...
Loading FXML using getResource() from: file:/C:/Users/allam/eclipse-workspace/ChatApp/bin/com/chatapp/gui/login-register.fxml
LoginRegisterController instantiated.
LoginRegisterController initializing...
Message clearing listeners added to input fields.
Login CSS loaded successfully from: file:/C:/Users/allam/eclipse-workspace/ChatApp/bin/com/chatapp/gui/login-styles.css
```

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