**Code:**

import javax.swing.\*;

import java.awt.\*;

import java.util.\*;

import java.util.List;

class Message {

private String sender;

private String recipient;

private String text;

private Date timestamp;

public Message(String sender, String recipient, String text) {

this.sender = sender;

this.recipient = recipient;

this.text = text;

this.timestamp = new Date();

}

@Override

public String toString() {

return "[" + timestamp + "] " + sender + ": " + text;

}

}

public class ChatAppGUI {

private static List<String> usernames = new ArrayList<>();

private static List<String> passwords = new ArrayList<>();

private static Map<String, List<Message>> chats = new HashMap<>();

private static Set<String> onlineUsers = new HashSet<>();

private JFrame frame;

private JTextArea chatArea;

private JTextField inputField;

private JTextField usernameField;

private JPasswordField passwordField;

private JList<String> userList;

private DefaultListModel<String> userListModel;

private JPanel cardPanel;

private CardLayout cardLayout;

private String currentUser;

public static void main(String[] args) {

SwingUtilities.invokeLater(ChatAppGUI::new);

}

public ChatAppGUI() {

createAndShowGUI();

}

private void createAndShowGUI() {

frame = new JFrame("Chat App");

frame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

frame.setSize(800, 600);

frame.setMinimumSize(new Dimension(800, 600));

cardLayout = new CardLayout();

cardPanel = new JPanel(cardLayout);

JPanel loginPanel = createLoginPanel();

JPanel chatPanel = createChatPanel();

cardPanel.add(loginPanel, "Login");

cardPanel.add(chatPanel, "Chat");

frame.getContentPane().add(cardPanel);

frame.setVisible(true);

}

private JPanel createLoginPanel() {

JPanel panel = new JPanel(new GridBagLayout());

GridBagConstraints gbc = new GridBagConstraints();

gbc.insets = new Insets(10, 10, 10, 10);

JLabel titleLabel = new JLabel("Welcome to Chat App");

titleLabel.setFont(new Font("Arial", Font.BOLD, 24));

gbc.gridx = 0;

gbc.gridy = 0;

gbc.gridwidth = 2;

panel.add(titleLabel, gbc);

gbc.gridwidth = 1;

gbc.gridx = 0;

gbc.gridy = 1;

panel.add(new JLabel("Username:"), gbc);

usernameField = new JTextField(15);

gbc.gridx = 1;

panel.add(usernameField, gbc);

gbc.gridx = 0;

gbc.gridy = 2;

panel.add(new JLabel("Password:"), gbc);

passwordField = new JPasswordField(15);

gbc.gridx = 1;

panel.add(passwordField, gbc);

JButton loginButton = new JButton("Login");

loginButton.addActionListener(e -> login());

gbc.gridx = 0;

gbc.gridy = 3;

panel.add(loginButton, gbc);

JButton signUpButton = new JButton("Sign Up");

signUpButton.addActionListener(e -> signUp());

gbc.gridx = 1;

panel.add(signUpButton, gbc);

return panel;

}

private JPanel createChatPanel() {

JPanel panel = new JPanel(new BorderLayout(10, 10));

chatArea = new JTextArea();

chatArea.setEditable(false);

chatArea.setFont(new Font("Monospaced", Font.PLAIN, 14));

chatArea.setLineWrap(true);

chatArea.setWrapStyleWord(true);

JScrollPane scrollPane = new JScrollPane(chatArea);

panel.add(scrollPane, BorderLayout.CENTER);

inputField = new JTextField();

inputField.setFont(new Font("Arial", Font.PLAIN, 14));

inputField.addActionListener(e -> sendMessage());

panel.add(inputField, BorderLayout.SOUTH);

userListModel = new DefaultListModel<>();

userList = new JList<>(userListModel);

userList.setSelectionMode(ListSelectionModel.SINGLE\_SELECTION);

userList.setFont(new Font("Arial", Font.PLAIN, 14));

userList.setBorder(BorderFactory.createTitledBorder("Users"));

userList.addListSelectionListener(e -> displayUserStatus());

JScrollPane userScrollPane = new JScrollPane(userList);

userScrollPane.setPreferredSize(new Dimension(150, 0));

panel.add(userScrollPane, BorderLayout.EAST);

JLabel headerLabel = new JLabel("Chat Room", JLabel.CENTER);

headerLabel.setFont(new Font("Arial", Font.BOLD, 18));

panel.add(headerLabel, BorderLayout.NORTH);

JButton logoutButton = new JButton("Logout");

logoutButton.addActionListener(e -> logout());

JPanel buttonPanel = new JPanel(new FlowLayout(FlowLayout.RIGHT));

buttonPanel.add(logoutButton);

panel.add(buttonPanel, BorderLayout.NORTH);

return panel;

}

private void login() {

String username = usernameField.getText();

String password = new String(passwordField.getPassword());

int index = usernames.indexOf(username);

if (index != -1 && passwords.get(index).equals(password)) {

if (onlineUsers.contains(username)) {

JOptionPane.showMessageDialog(frame, "User is already logged in.", "Login Failed", JOptionPane.WARNING\_MESSAGE);

return;

}

currentUser = username;

onlineUsers.add(username);

refreshUserList();

switchToChatPanel();

} else {

JOptionPane.showMessageDialog(frame, "Invalid username or password.", "Login Failed", JOptionPane.ERROR\_MESSAGE);

}

}

private void signUp() {

String username = usernameField.getText();

String password = new String(passwordField.getPassword());

if (usernames.contains(username)) {

JOptionPane.showMessageDialog(frame, "Username already exists.", "Sign Up Failed", JOptionPane.ERROR\_MESSAGE);

} else {

usernames.add(username);

passwords.add(password);

chats.put(username, new ArrayList<>());

JOptionPane.showMessageDialog(frame, "Sign-up successful. Please log in.");

}

}

private void sendMessage() {

String messageText = inputField.getText().trim();

if (messageText.isEmpty()) return;

String recipient = userList.getSelectedValue();

if (recipient == null) {

recipient = JOptionPane.showInputDialog(

frame,

"Enter the username of the recipient:",

"Select Recipient",

JOptionPane.QUESTION\_MESSAGE

);

if (recipient == null || recipient.trim().isEmpty()) {

JOptionPane.showMessageDialog(frame, "No recipient selected.", "No Recipient", JOptionPane.WARNING\_MESSAGE);

return;

}

if (!usernames.contains(recipient)) {

JOptionPane.showMessageDialog(frame, "The username does not exist.", "Invalid Recipient", JOptionPane.ERROR\_MESSAGE);

return;

}

}

Message message = new Message(currentUser, recipient, messageText);

chats.get(currentUser).add(message);

chats.get(recipient).add(message);

chatArea.append(message + "\n");

inputField.setText("");

}

private void displayUserStatus() {

String username = userList.getSelectedValue();

if (username != null) {

String status = onlineUsers.contains(username) ? "Online" : "Offline";

JOptionPane.showMessageDialog(frame, username + " is " + status, "User Status", JOptionPane.INFORMATION\_MESSAGE);

}

}

private void refreshUserList() {

userListModel.clear();

for (String username : usernames) {

userListModel.addElement(username);

}

}

private void switchToChatPanel() {

refreshUserList();

chatArea.setText("");

List<Message> userChats = chats.get(currentUser);

for (Message message : userChats) {

chatArea.append(message + "\n");

}

cardLayout.show(cardPanel, "Chat");

}

private void logout() {

if (currentUser != null) {

onlineUsers.remove(currentUser);

currentUser = null;

}

chatArea.setText("");

inputField.setText("");

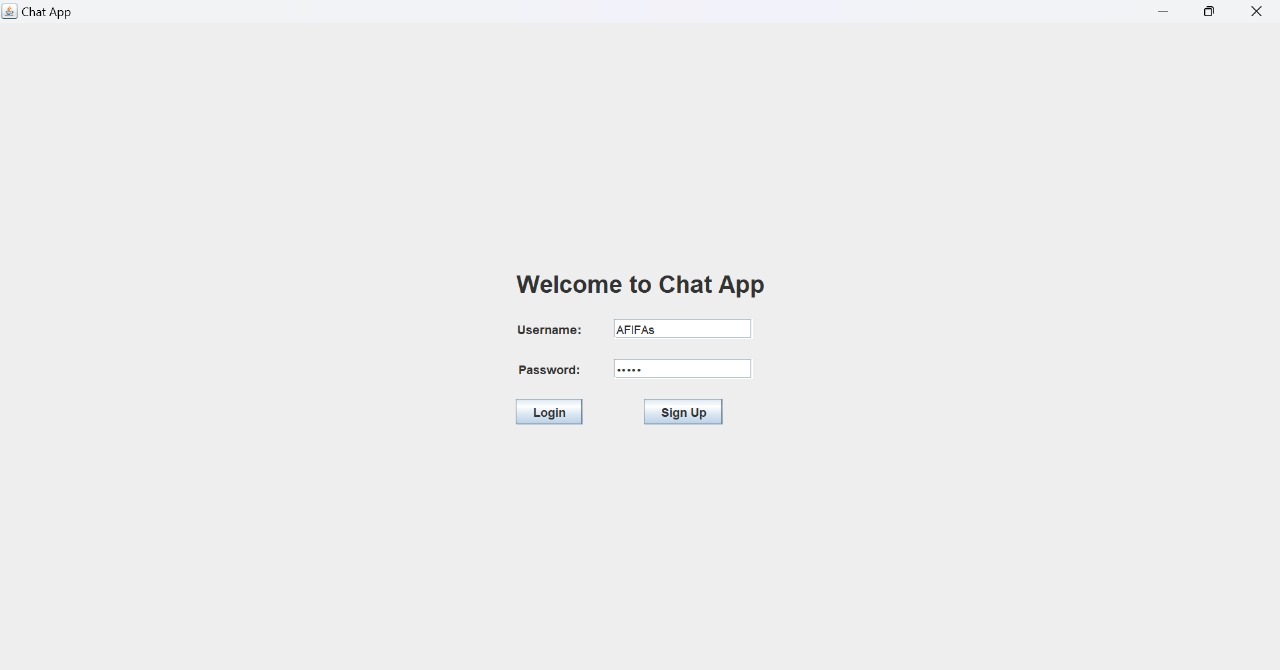
cardLayout.show(cardPanel, "Login");

}

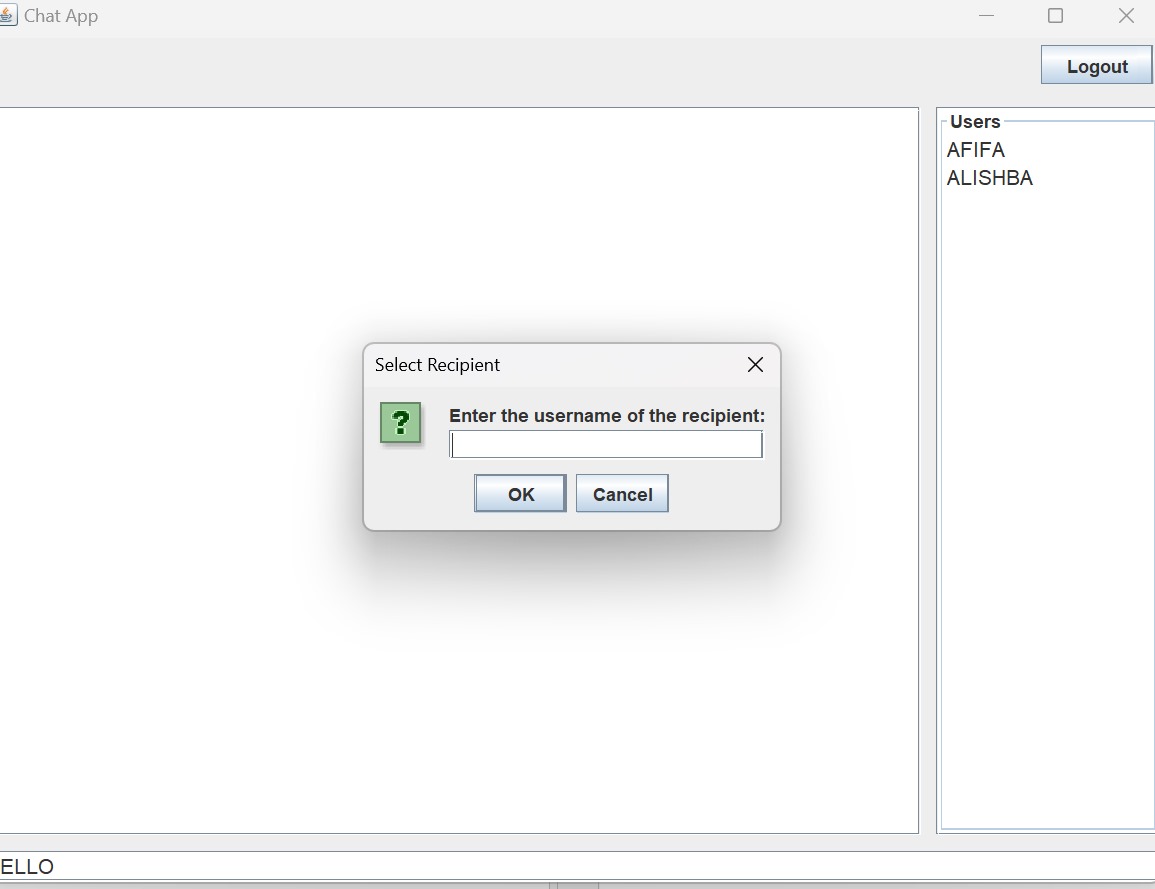
}

OUTPUT:

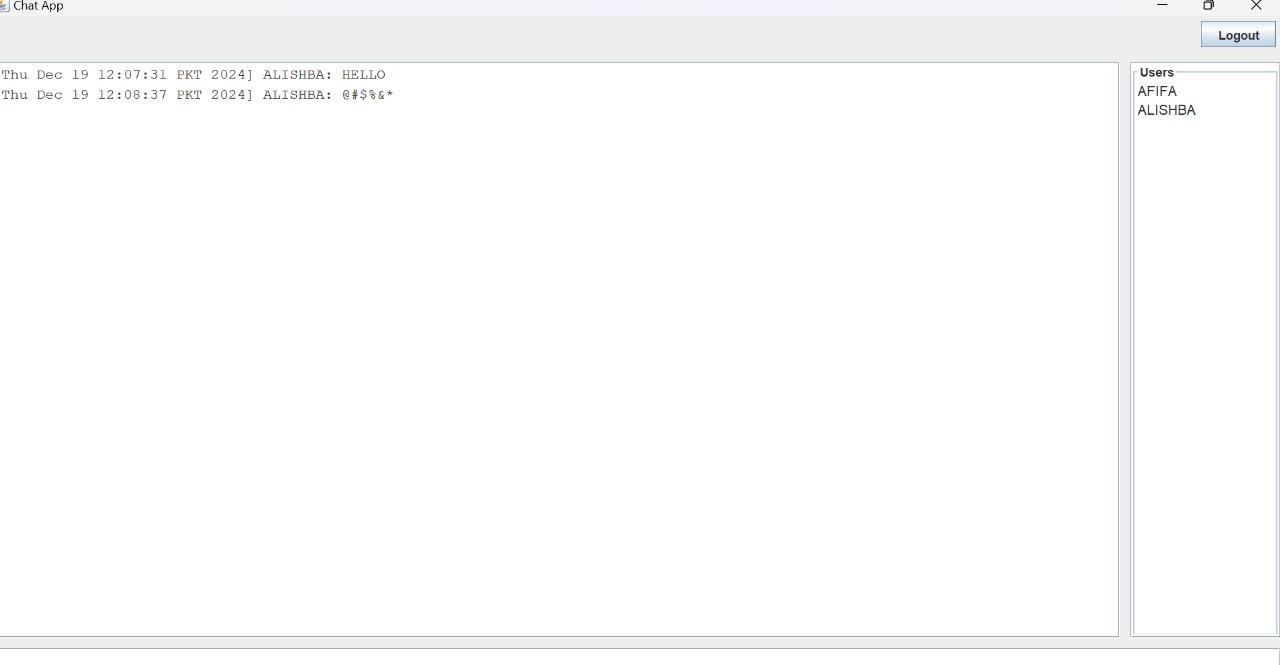
SignUP/Login:



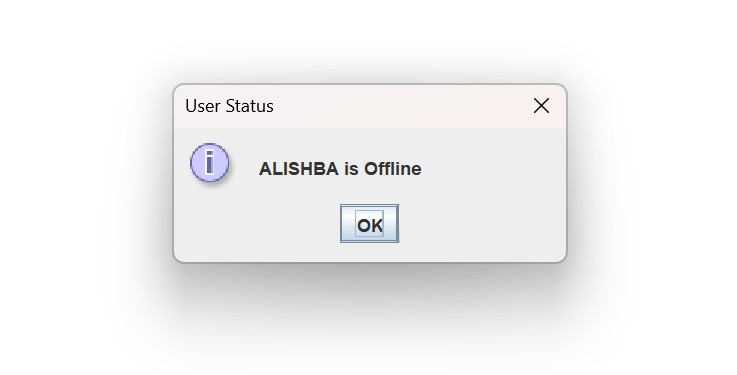
**Enter recipient username to send message:**

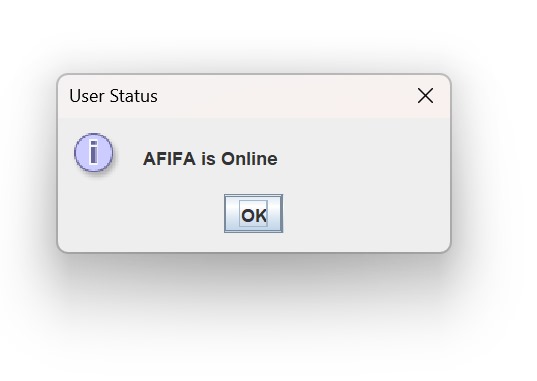
****

**Message sent :**

****

**Checking status:**

****



**Test cases for the Send message functionality:**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Tester ID** | **Test ID** | **Description** | **Test Data** | **Expected output** | **Actual output** | **Class** | **Status** |
| 001 | TC1 | **Send message** functionality | Username: "user1", Recipient: "user2", Message: "Hello user2!" | Message appears in the chat area with the correct sender ("user1") . | Message is appeared in the chat area with the correct sender ("user1"). | Valid | Pass |
| 002 | TC2 | **Send message** functionality | Username: "user1", Recipient: "user9", Message: "Hi user9!" | Warning dialog: "The user name does not exist." | Warning dialog: "The user name does not exist." | Invalid | Pass |
| 003 | TC3 | **Send message** functionality | Username: "user1", Recipient: "user2", Message: "" | No message is sent.  - Input field remains empty.  - No changes in the chat area. | No message is sent.  - Input field remains empty.  - No changes in the chat area. | Invalid | pass |
| 004 | TC4 | **Send message** functionality | Username: "user1", Recipient: None, Message: | Warning dialog: User ” “ not found. -Message is not sent. | Warning dialog: User “ ” not found.  -Message is not sent. | Invalid | Pass |
| 005 | TC5 | **Send message** functionality | Sender: "user1", Recipient: "user2", Message: "@!#%$" | Message "@!#%$" is sent from "user1" to "user2". | Message "@!#%$" is sent from "user1" to "user2". | Valid | Pass |
| 006 | TC6 | **Send message** functionality | Sender: "user1", Recipient: "user1", Message: "Test" | Message "Test" sent from "user1" to "user1". | Message "Test" is sent from "user1" to "user1". | Valid | Pass |

**Test cases for the Show Status functionality:**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Tester ID** | **Test ID** | **Decsription** | **Test Data** | **Expected output** | **Actual output** | **Class** | **Status** |
| 001 | TC1 | **Show Status** functionality | Logged-in User: "user1", Check Status for "user1" | "user1 is currently Online."  - Logged-in user should be shown as online. | "user1 is currently Online."  - Logged-in user is shown as online. | Valid | Pass |
| 002 | TC2 | **Show Status** functionality | Logged-in User: "user1", Check Status for "user3" | "user3 is Offline."  - Correct offline status is displayed. | "user3 is Offline."  - Correct offline status is displayed. | Valid | Pass |
| 003 | TC3 | **Show Status** functionality | Sender: "user1", Recipient: "user2" | Status of "user2" checked; "user2 is Offline". | Status of "user2" checked; "user2 is Offline". | Valid | Pass |
| 004 | TC4 | **Show Status** functionality | username = "user6" re-login | Dialog displays: "user6 is Online". | Dialog displays: "user6 is Online". | Valid | Pass |
| 005 | TC5 | **Show Status** functionality | Logged-in User: "user1", Logout, Check Status for "user1" | "user1 is Offline."  - Logged-out user should be shown as offline. | "user1 is Offline."  - Logged-out user is shown as offline. | Valid | Pass |