Name: Yinhla Abundence

Surname: Baloyi

Student No: ST10474409

Module code: PROG 5121

Module: Programming 1A

QULIFICATION: BACHELOR OF INFORMATION TECHNOLOGY IN BUSINESS SYSTEM

ASSIGNMENT TYPE: ASSIGNMENT 2

GITHUB REPOSITORY LINK

ST10474409_ChatApp.java (Main)

```java
package st10474409_chatapp;

import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
import java.util.regex.Pattern;

public class ST10474409_ChatApp {
    private String username;
    private String password;
    private String cellNumber;
    private String firstName;
    private String lastName;
    private boolean isLoggedIn = false;

    // === GUI Entry Point ===
    public static void main(String[] args) {
        SwingUtilities.invokeLater(() -> {
            ST10474409_ChatApp app = new ST10474409_ChatApp();
            app.showMainMenu();
        });
    }
```

```java
// === GUI: Main Menu ===
private void showMainMenu() {
    JFrame frame = new JFrame("QuickChat Messaging Apllication");
    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    frame.setSize(400, 300);
    frame.setLayout(new BorderLayout());


    JLabel welcomeLabel = new JLabel("Welcome to QuickChat.",
SwingConstants.CENTER);
    welcomeLabel.setFont(new Font("Arial", Font.BOLD, 18));


    JButton sendBtn = new JButton("Send Messages");
    JButton recentBtn = new JButton("Show Recently Sent Messages");
    JButton storedBtn = new JButton("View Stored Messages");
    JButton quitBtn = new JButton("Quit");


    JPanel buttonPanel = new JPanel(new GridLayout(4, 1, 10, 10));
    buttonPanel.add(sendBtn);
    buttonPanel.add(recentBtn);
    buttonPanel.add(storedBtn);
    buttonPanel.add(quitBtn);


    frame.add(welcomeLabel, BorderLayout.NORTH);
    frame.add(buttonPanel, BorderLayout.CENTER);


    // === Event Listeners ===
    sendBtn.addActionListener(e -> sendMessagesGUI());
    recentBtn.addActionListener(e -> JOptionPane.showMessageDialog(frame,
"Coming Soon."));
```

```java
        storedBtn.addActionListener(e -> viewStoredMessagesGUI());

        quitBtn.addActionListener(e -> {

            JOptionPane.showMessageDialog(frame, "Thank you for using QuickChat.
Goodbye!");

            frame.dispose();

        });


        frame.setLocationRelativeTo(null);

        frame.setVisible(true);

    }


    // === GUI: Send Messages ===

    private void sendMessagesGUI() {

        JFrame sendFrame = new JFrame("Send Message");

        sendFrame.setSize(400, 400);

        sendFrame.setLayout(new GridLayout(7, 1, 10, 10));


        JTextField recipientField = new JTextField();

        JTextArea messageArea = new JTextArea();

        JButton sendBtn = new JButton("Send Message");

        JButton storeBtn = new JButton("Store Message");

        JButton discardBtn = new JButton("Discard Message");


        sendFrame.add(new JLabel("Recipient Cell Number (with +code):"));

        sendFrame.add(recipientField);

        sendFrame.add(new JLabel("Message (max 250 characters):"));

        sendFrame.add(new JScrollPane(messageArea));

        sendFrame.add(sendBtn);
```

```java
        sendFrame.add(storeBtn);

        sendFrame.add(discardBtn);


        sendFrame.setLocationRelativeTo(null);

        sendFrame.setVisible(true);


        // Message object

        Message message = new Message();


        // === Action Listeners ===

        sendBtn.addActionListener(e -> {

            String recipient = recipientField.getText().trim();

            String text = messageArea.getText().trim();


            if (message.checkRecipientCell(recipient) == 0) {

                JOptionPane.showMessageDialog(sendFrame, "Invalid phone number. Must
include international code.", "Error", JOptionPane.ERROR_MESSAGE);

                return;

            }

            if (text.length() > 250) {

                JOptionPane.showMessageDialog(sendFrame, "Message exceeds 250
characters.", "Error", JOptionPane.ERROR_MESSAGE);

                return;

            }


            message.setRecipient(recipient);

            message.setMessage(text);

            String result = message.sentMessage(1); // Send
```

```java
            JOptionPane.showMessageDialog(sendFrame, result + "\n\n" +
message.printMessages(), "Message Sent", JOptionPane.INFORMATION_MESSAGE);

        });


    storeBtn.addActionListener(e -> {

        String recipient = recipientField.getText().trim();

        String text = messageArea.getText().trim();


        if (message.checkRecipientCell(recipient) == 0 || text.isEmpty()) {

            JOptionPane.showMessageDialog(sendFrame, "Please fill in all fields correctly.",
"Error", JOptionPane.ERROR_MESSAGE);

            return;

        }


        message.setRecipient(recipient);

        message.setMessage(text);

        String result = message.sentMessage(3); // Store

        JOptionPane.showMessageDialog(sendFrame, result, "Stored",
JOptionPane.INFORMATION_MESSAGE);

        });


    discardBtn.addActionListener(e -> {

        recipientField.setText("");

        messageArea.setText("");

        JOptionPane.showMessageDialog(sendFrame, "Message discarded.");

        });

    }


  // === GUI: View Stored Messages ===
```

```java
private void viewStoredMessagesGUI() {

    JFrame viewFrame = new JFrame("Stored Messages");

    viewFrame.setSize(400, 300);


    JTextArea messagesArea = new JTextArea();

    messagesArea.setEditable(false);

    messagesArea.setText(JSONHandler.getAllMessages());


    viewFrame.add(new JScrollPane(messagesArea));

    viewFrame.setLocationRelativeTo(null);

    viewFrame.setVisible(true);

}


// === Validation Methods ===

public boolean checkUsername(String username) {

    return username.length() <= 5 && username.contains("_");

}


public boolean checkPasswordComplexity(String password) {

    if (password.length() < 8) return false;

    if (!Pattern.compile("[A-Z]").matcher(password).find()) return false;

    if (!Pattern.compile("[0-9]").matcher(password).find()) return false;

    return Pattern.compile("[^A-Za-z0-9]").matcher(password).find();

}


public boolean checkCellPhoneNumber(String cellNumber) {

    String pattern = "^\\+\\d{1,3}\\d{7,10}$";

    return Pattern.matches(pattern, cellNumber);
```

```java
    }


    // === Login ===

    public boolean loginUser(String enteredUsername, String enteredPassword) {

        return enteredUsername.equals(this.username) &&
enteredPassword.equals(this.password);

    }


    public String returnLoginStatus(boolean isSuccessful) {

        if (isSuccessful) {

            return "Welcome " + firstName + " " + lastName + ", it is great to see you again.";

        }

        return "Username or password incorrect, please try again.";

    }
}
```

Message.java
```java
package st10474409_chatapp;


import java.util.Random;


public class Message {
    private String messageID;

    private int messageCount;

    private String recipient;

    private String message;

    private String messageHash;

    private String status; // "sent", "stored", "discarded"
```

```java
private static int totalMessagesSent = 0;

private static int messageCounter = 0;


public Message() {

    this.messageID = generateMessageID();

    this.messageCount = ++messageCounter;

    this.status = "pending";

}


// Generate random 10-digit message ID

private String generateMessageID() {

    Random rand = new Random();

    long id = 1000000000L + (long)(rand.nextDouble() * 9000000000L);

    return String.valueOf(id);

}


public boolean checkMessageID() {

    return this.messageID.length() == 10;

}


public int checkRecipientCell(String recipient) {

    // Check if number starts with international code and has proper length

    if (recipient.startsWith("+") && recipient.length() <= 13 && recipient.length() >= 11) {

        String numberPart = recipient.substring(1);

        if (numberPart.matches("\\d+")) {

            return 1; // Success

        }

    }
```

```java
        return 0; // Failure

    }


    public String createMessageHash() {

        String firstTwo = messageID.substring(0, 2);


        // Extract first and last words from message

        String[] words = message.split(" ");

        String firstWord = words.length > 0 ? words[0].toUpperCase() : "";

        String lastWord = words.length > 1 ? words[words.length - 1].toUpperCase() :
firstWord;


        return firstTwo + ":" + messageCount + ":" + firstWord + lastWord;

    }


    public String sentMessage(int choice) {

      switch (choice) {

        case 1: // Send Message

          totalMessagesSent++;

          this.status = "sent";

          // Store in JSON

          JSONHandler.storeMessage(this);

          return "Message successfully sent.";

        case 2: // Disregard Message

          this.status = "discarded";

          return "Press 0 to delete message.";

        case 3: // Store Message

          this.status = "stored";
```

```java
        // Store in JSON

        JSONHandler.storeMessage(this);

        return "Message successfully stored.";

      default:

        return "Invalid option.";

  }

}


public String printMessages() {

  return "MessageID: " + messageID +

      "\nMessage Hash: " + messageHash +

      "\nRecipient: " + recipient +

      "\nMessage: " + message +

      "\nStatus: " + status;

}


public static int returnTotalMessages() {

  return totalMessagesSent;

}


// Getters and Setters

public String getMessageID() { return messageID; }

public int getMessageCount() { return messageCount; }

public String getRecipient() { return recipient; }

public void setRecipient(String recipient) { this.recipient = recipient; }

public String getMessage() { return message; }

public void setMessage(String message) {

  this.message = message;
```

```java
        this.messageHash = createMessageHash();
    }

    public String getMessageHash() { return messageHash; }

    public String getStatus() { return status; }

    public void setStatus(String status) { this.status = status; }
}
```

JSONHandler.java

```java
package st10474409_chatapp;

import java.io.*;

import java.text.SimpleDateFormat;

import java.util.Date;

import java.util.ArrayList;

import java.util.List;

public class JSONHandler {
    private static final String FILE_NAME = "messages.json";

    public static void storeMessage(Message message) {
        try {
            // Read existing messages
            List<String> existingMessages = readAllMessages();

            // Create JSON object for new message
            String messageJson = createMessageJSON(message);
```

```java
        // Add new message to list

        existingMessages.add(messageJson);


        // Write back to file

        writeMessagesToFile(existingMessages);


        System.out.println("Message stored in JSON file: " + FILE_NAME);


    } catch (IOException e) {

        System.err.println("Error storing message: " + e.getMessage());

    }

}


private static String createMessageJSON(Message message) {

    StringBuilder json = new StringBuilder();

    json.append(" {\n");

    json.append("   \"messageID\":
\"").append(escapeJSON(message.getMessageID())).append("\",\n");

    json.append("   \"messageCount\":
").append(message.getMessageCount()).append(",\n");

    json.append("   \"recipient\":
\"").append(escapeJSON(message.getRecipient())).append("\",\n");

    json.append("   \"message\":
\"").append(escapeJSON(message.getMessage())).append("\",\n");

    json.append("   \"messageHash\":
\"").append(escapeJSON(message.getMessageHash())).append("\",\n");

    json.append("   \"status\":
\"").append(escapeJSON(message.getStatus())).append("\",\n");

    json.append("   \"timestamp\": \"").append(new SimpleDateFormat("yyyy-MM-dd
HH:mm:ss").format(new Date())).append("\"\n");
```

```java
        json.append("  }");
        return json.toString();
    }


    private static String escapeJSON(String text) {
        if (text == null) return "";
        return text.replace("\\", "\\\\")
                .replace("\"", "\\\"")
                .replace("\b", "\\b")
                .replace("\f", "\\f")
                .replace("\n", "\\n")
                .replace("\r", "\\r")
                .replace("\t", "\\t");
    }


    private static List<String> readAllMessages() throws IOException {
        List<String> messages = new ArrayList<>();


        File file = new File(FILE_NAME);
        if (!file.exists()) {
            return messages;
        }


        try (BufferedReader reader = new BufferedReader(new FileReader(file))) {
            String line;
            StringBuilder currentMessage = new StringBuilder();
            boolean inMessage = false;
```

```java
            while ((line = reader.readLine()) != null) {

                line = line.trim();


                if (line.equals("[")) continue;

                if (line.equals("]")) break;


                if (line.equals("{")) {

                    inMessage = true;

                    currentMessage = new StringBuilder();

                    currentMessage.append("{\n");

                } else if (line.equals("},") || line.equals("}")) {

                    if (inMessage) {

                        currentMessage.append("}");

                        messages.add(currentMessage.toString());

                        inMessage = false;

                    }

                } else if (inMessage) {

                    currentMessage.append(line).append("\n");

                }

            }

        }


        return messages;

    }


    private static void writeMessagesToFile(List<String> messages) throws IOException {

        try (FileWriter file = new FileWriter(FILE_NAME)) {

            file.write("[\n");
```

```java
        for (int i = 0; i < messages.size(); i++) {

            file.write(messages.get(i));

            if (i < messages.size() - 1) {

                file.write(",");

            }

            file.write("\n");

        }


        file.write("]");

        file.flush();

    }

}


public static String getAllMessages() {

    try {

        List<String> messages = readAllMessages();

        if (messages.isEmpty()) {

            return "No messages stored.";

        }


        StringBuilder sb = new StringBuilder();

        sb.append("Stored Messages:\n");

        sb.append("===============\n");


        for (String messageJson : messages) {

            // Simple parsing to extract key information

            String[] lines = messageJson.split("\n");
```

```java
        String messageID = extractValue(lines, "messageID");

        String recipient = extractValue(lines, "recipient");

        String status = extractValue(lines, "status");

        String timestamp = extractValue(lines, "timestamp");


        sb.append("ID: ").append(messageID)

          .append(" | To: ").append(recipient)

          .append(" | Status: ").append(status)

          .append(" | Time: ").append(timestamp)

          .append("\n");

      }

      return sb.toString();


    } catch (IOException e) {

      return "Error reading messages: " + e.getMessage();

    }

}


private static String extractValue(String[] lines, String key) {

  for (String line : lines) {

    if (line.contains("\"" + key + "\":")) {

      int start = line.indexOf(":") + 1;

      String value = line.substring(start).trim();

      if (value.startsWith("\"")) {

        value = value.substring(1, value.length() - 1);

      }

      // Remove trailing comma if present

      if (value.endsWith(",")) {
```

```java
                value = value.substring(0, value.length() - 1);

            }

            return unescapeJSON(value);

        }

    }

    return "N/A";

}


private static String unescapeJSON(String text) {

    return text.replace("\\\"", "\"")

            .replace("\\\\", "\\")

            .replace("\\n", "\n")

            .replace("\\r", "\r")

            .replace("\\t", "\t")

            .replace("\\b", "\b")

            .replace("\\f", "\f");

}

}
```