PROG5121

Assignment 3

ST10472114

Gareth De Jager

Link for personal GitHub

https://github.com/GDJCodes/ST10472114-Assignment-3

Link for GitHub Classroom

https://github.com/IIEVCPMB/programming-1a-poe-GDJCodes

Below is the code first for login and then for message class and after is me manually testing with the tests provided in POE:

Login Code:

```
package com.mycompany.login;


/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template
 */


import java.io.FileNotFoundException;
import javax.swing.*;




/**
 *
 * @author garet
 */
```

```
public class Login {

static String firstName;
  static String lastName;
  static String username;
  static String password;
  static String cellPhone;



   public static boolean checkUserName() {
     return username.contains("_") && username.length() <= 5;
  }

  public static boolean checkPasswordComplexity() {
     return password.length() > 8 && password.matches(".*[A-Z].*") &&
password.matches(".*\\d.*") && password.matches(".*[!@#$%^&*()_+=<>?].*");
  }

  public static boolean checkCellPhoneNumber() {
     return cellPhone.matches("\\+27\\d{9}");
  }

  public static String registerUser() {
     firstName = JOptionPane.showInputDialog("Please enter your first name: ");
     if(firstName == null){
        JOptionPane.showMessageDialog(null, "Cancelled");
        System.exit(0);
     }
     lastName = JOptionPane.showInputDialog("Please enter your last name: ");
```

```
    if(lastName == null){
      JOptionPane.showMessageDialog(null, "Cancelled");
      System.exit(0);
    }

    while(true){
      username = JOptionPane.showInputDialog("Please enter your username: \n- Contains an underscore. \n- No more than 5 characters long.");
    if(username == null)return "Registration cancelled";
    if(checkUserName()){
      JOptionPane.showMessageDialog(null, "Welcome " + firstName + " " + lastName + " it is great to see you.");
    } else {
      JOptionPane.showMessageDialog(null, "Username is incorrectly formatted, please ensure that your username contains an underscore and is no more than five characters in length.");
      continue;
      }
    password = JOptionPane.showInputDialog("Please enter a password: \n- At least eight characters long. \n- Contain a capital letter. \n- Contain a number. \n- Contain a special character.");
    if(password == null) return "Registraiton cancelled";
    if(checkPasswordComplexity()){
      JOptionPane.showMessageDialog(null, "Password successfully captured.");
    }else{
    JOptionPane.showMessageDialog(null, "Password is not correctly formatted, please ensure that the password contains at least eight characters, a capital letter, a number, and a special character.");
    continue;
    }
    cellPhone = JOptionPane.showInputDialog("Please enter a valid cell phone number starting with the international country code: ");
    if(cellPhone == null) return "Registraiton cancelled";
    if(checkCellPhoneNumber()){
      JOptionPane.showMessageDialog(null, "Cell number successfully captured.");
    }else{
      JOptionPane.showMessageDialog(null, "Cell number is incorrectly formatted or does not contain an international code, please correct the number and try again.");
    }
    return "User has been registered successfully!";
  }
  }
public String getPassword(){
  return password;
}


public static void main(String[] args) {
      String registrationMessage = registerUser();
    JOptionPane.showMessageDialog(null, registrationMessage);



    if (registrationMessage.contains("successfully")) {
      boolean loginSuccess = false;
      while (!loginSuccess) {
      String loginUsername = JOptionPane.showInputDialog("Enter username to log in: ");
      String loginPassword = JOptionPane.showInputDialog("Enter password: ");

      loginSuccess = loginUser(loginUsername, loginPassword);
      JOptionPane.showMessageDialog(null, returnLoginStatus(loginSuccess));

      if(!loginSuccess){
        int retry = JOptionPane.showConfirmDialog(null, "Would you like to try again?", "login failed.", JOptionPane.YES_NO_OPTION);
        if (retry != JOptionPane.YES_OPTION)
          break;
      }
      }
    }
    Login login = new Login();
    Message chatApp = new Message();


    try{
    chatApp.run(login);
    }catch(FileNotFoundException e){
      JOptionPane.showMessageDialog(null, "Message not found: " + e.getMessage());
    }
}
public static boolean loginUser(String loginUsername, String loginPassword){
    return loginUsername.equals(username) && loginPassword.equals(password);
  }
  public static String returnLoginStatus(boolean loginSuccess) {
    if (loginSuccess) {
      return "A successful login.";
    } else {
      return "A failed login.";
    }
  }



}
```

Message class code:

```
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template
 */
package com.mycompany.login;
```

```
import java.awt.Dimension;
import java.io.BufferedReader;
import java.io.File;
import java.io.FileNotFoundException;
import java.io.FileReader;
import java.io.FileWriter;
import java.io.IOException;
import java.util.ArrayList;
import java.util.List;
import java.util.Random;
import javax.swing.JOptionPane;
import javax.swing.JScrollPane;
import javax.swing.JTextArea;
import org.json.JSONArray;
import org.json.JSONException;
import org.json.JSONObject;

/**
 *
 * @author garet
 */
public class Message {

    private ArrayList<String> draftMessages = new ArrayList<>();
    private ArrayList<String> disregardedMessages = new ArrayList<>();

    public void feedbackPrompt(){
    String filled = "\u2605";
    String empty = "\u2606";
```

```
  String[] stars = {
    filled + empty + empty + empty + empty + " - Poor",
    filled + filled + empty + empty + empty + " - Fair",
    filled + filled + filled + empty + empty + " - Good",
    filled + filled + filled + filled + empty + " - Very Good",
    filled + filled + filled + filled + filled + " - Excellent"
  };

  String rating = (String) JOptionPane.showInputDialog(
      null,
  "How would you rate your experience",
      "Rate us!",
      JOptionPane.QUESTION_MESSAGE,
      null,
      stars,
      stars[2]
  );

  if(rating != null){
    JOptionPane.showMessageDialog(null, "Thank you for the feedback! \n You rated us: " +
rating);
  }
}
public boolean checkMessageID(String id) {
    return id != null && id.matches("\\d{10}") ;
  }
  public static int checkRecipientCell(String number){
    if (number == null)
      return 0;
    if (number.startsWith("+27") && number.length() == 12 &&
number.substring(3).matches("\\d{9}")){
      return 1;
```

```
        }return 0;
    }
    public void startChat(){
        JOptionPane.showMessageDialog(null, "Welcome to QuickChat");


    }
    public void run(Login login) throws FileNotFoundException {


        JOptionPane.showMessageDialog(null, "Login Successful. Welcome, " + Login.username +
"!");
        boolean running = true;
        while (running){
            String[] options = {
                "1 - Send Message",
                "2 - Show recently sent messages",
                "3 - Messages",
                "4 - Quit"
            };
            String choiceString = (String) JOptionPane.showInputDialog(null,
                    "Choose an option:",
                    "Main menu",
                    JOptionPane.PLAIN_MESSAGE,
                    null,
                    options,
                    options[0]
                    );
            if (choiceString == null){
                feedbackPrompt();
                System.exit(0);
                continue;
            }
            int choice = Integer.parseInt(choiceString.substring(0, 1));
            switch (choice){
                case 1 -> sentMessage(Login.username);
                case 2 -> JOptionPane.showMessageDialog(null, "Coming soon.");
                case 3 -> messagesMenu();
                case 4 -> {
                    JOptionPane.showMessageDialog(null, "Session ended.");
                    deleteMessagesOnExit();
                    feedbackPrompt();
                    running = false;
                }

                default -> JOptionPane.showMessageDialog(null, "Invalid option");
            }


        }
    }
    private List<String> loadMessagesFromJson(){
    List<String> messages = new ArrayList<>();

    try{
        File file = new File("messages.json");
        if(!file.exists()){
        file.createNewFile();
        return messages;
    }
        try (BufferedReader reader = new BufferedReader(new FileReader(file))) {
            String line;
            while ((line = reader.readLine()) !=null){
```

```
        try{
          JSONObject obj = new JSONObject(line);
          String from = obj.optString("from");
          String to = obj.optString("to");
          String content = obj.optString("message");
          messages.add("From: " + from + " to " + to + ": " + content);
        }catch(JSONException je){
          System.out.println("Error reading! " + je.getMessage());
        }
      }     }
}catch(IOException e){
  JOptionPane.showMessageDialog(null, "Error reading file: " + e.getMessage());
}
return messages;
}



  public void sentMessage(String username){


    String recipient;
    while(true){
      recipient = JOptionPane.showInputDialog("Enter recipients number: ");


      if (recipient == null) return;


  if (recipient.matches("\\+27\\d{9}")){
    break;
  }
  else{
    JOptionPane.showMessageDialog(null,"Recipient cannot be empty.");
    return;
  }
}
  int messageCount = 0;
  try{
    String countInput = JOptionPane.showInputDialog("How many messages would you like to
send?");
    if (countInput == null)


      System.exit(0);


    messageCount = Integer.parseInt(countInput);


    if (messageCount <= 0) {
      JOptionPane.showMessageDialog(null, "Enter number.");
      return;
    }
  }catch(NumberFormatException e){
        JOptionPane.showMessageDialog(null, "Invalid number");
        return;
        }


  for (int i = 1; i <= messageCount; i++){
  String message = JOptionPane.showInputDialog("Enter the message: ");
  if (message == null || message.trim().isEmpty()){
    disregardedMessages.add("Disregarded message!");
    return;
  }


  String messageId = checkMessageID();
  String messageHash = createMessageHash(messageId, i, message);
  String fullMessage = "[Message ID: " + messageId + "}\nHash: " + messageHash +
      "\nFrom: " + username + " to " + recipient + ": " + message;
  String[] options = {"Send", "Disregard", "Store for later" };
  int choice = JOptionPane.showOptionDialog(
      null,
      "What would you like to do?" + fullMessage,
      "Confirm Message",
      JOptionPane.DEFAULT_OPTION,
      JOptionPane.QUESTION_MESSAGE,
      null,
      options,
      options[0]
      );
      if(options == null){
        JOptionPane.showMessageDialog(null, "Cancelled!");
      }


      JSONObject messageObj = new JSONObject();
      messageObj.put("Message ID", messageId);
      messageObj.put("Message Hash", messageHash);
      messageObj.put("from",username);
      messageObj.put("to", recipient);
      messageObj.put("message", message);

      switch(choice){
        case 0 -> {
          try(FileWriter file = new FileWriter("messages.json", true)){
            file.write(messageObj.toString() + "\n");
            JOptionPane.showMessageDialog(null, "Message sent!");
          }catch(IOException e){
```

```
                        JOptionPane.showMessageDialog(null, "Failed to send: " + e.getMessage());
                    }
            }
                case 1 -> JOptionPane.showMessageDialog(null, "Message Discarded!");
                case 2 -> {
                    try(FileWriter file = new FileWriter("drafts.json", true)){
                        file.write(messageObj.toString() + "\n");
                        JOptionPane.showMessageDialog(null, "Message saved to drafts.");
                    }catch(IOException e){
                        JOptionPane.showConfirmDialog(null, "Failed to save draft: " + e.getMessage());
                    }
            }

                default -> JOptionPane.showMessageDialog(null, "Invalid choice!");
            }

}
    }

    public void messagesMenu() throws FileNotFoundException{
        boolean inMessagesMenu = true;



        while (inMessagesMenu){
            String[] messageOptions = {
                "1 - Display senders and recipients",
                "2 - Display longest message",
                "3 - Message ID: ",
                "4 - Recipient number: ",
                "5 - Hash number",
  "6 - Display message report",
  "7 - More options",
  "8 - Back"
};
String msgChoiceString = (String)JOptionPane.showInputDialog(
null,
    "Choose an option: ",
    "Back to main Menu",
    JOptionPane.PLAIN_MESSAGE,
    null,
    messageOptions,
    messageOptions[0]
);
if(msgChoiceString == null)
  break;
int msgChoice = Integer.parseInt(msgChoiceString.substring(0, 1));

switch (msgChoice) {
  case 1 -> displaySendersAndRecipients();
  case 2 -> displayLongestMessage();
  case 3 -> searchMessageID();
  case 4 -> searchRecipientNumber();
  case 5 -> searchHashNumber();
  case 6 -> displayMessageReport();
  case 7 -> moreOptionsMenu();
  case 8 -> {
    feedbackPrompt();
    deleteMessagesOnExit();
    inMessagesMenu = false;
  }
  default -> JOptionPane.showMessageDialog(null, "Invalid choice");
```

```
            }
        }
    }

    public void displaySendersAndRecipients(){
        List<String> messages = loadMessagesFromJson();

        if(messages.isEmpty()){
            JOptionPane.showMessageDialog(null, "No messages sent!");
        }
        StringBuilder summary = new StringBuilder("Senders and Recipients:\n\n");

        for (String msg : messages){
            String from = extractBetween(msg, "From: "," to ");
            String to = extractBetween(msg,"to ", ":");
            summary.append("From: ").append(from).append(" To: ").append(to).append("\n");
        }

        JTextArea textArea = new JTextArea(summary.toString());
        textArea.setEditable(false);
        JScrollPane scrollPane = new JScrollPane(textArea);
        scrollPane.setPreferredSize(new Dimension(500, 300));



        JOptionPane.showMessageDialog(null, scrollPane, "Sender and Recipeint list: ",
JOptionPane.INFORMATION_MESSAGE);
    }

    private String extractBetween(String text, String start, String end){
        int startIndex = text.indexOf(start);
        int endIndex = text.indexOf(end, startIndex + start.length());
        if(startIndex == -1 || endIndex == -1)
            return "N/A";
        return text.substring(startIndex + start.length(), endIndex).trim();
    }

    private void displayLongestMessage(){
        List<String> messages = loadMessagesFromJson();

        if(messages.isEmpty()){
            JOptionPane.showMessageDialog(null, "No messages sent!");
        }

        String longestMessage = "";
        for (String message : messages){
            if(message.length()>longestMessage.length()){
                longestMessage = message;
            }
    }
        if (longestMessage.trim().isEmpty()){
            JOptionPane.showMessageDialog(null, "Longest message is empty!");
        }
        JTextArea textArea = new JTextArea("Longest message: \n\n" + longestMessage);
        textArea.setEditable(false);
        JScrollPane scrollPane = new JScrollPane(textArea);
        scrollPane.setPreferredSize(new Dimension(500, 200));



        JOptionPane.showMessageDialog(null, scrollPane, "Longest Message",
JOptionPane.INFORMATION_MESSAGE);
    }
```

```
  private void searchMessageID(){
    List<String> messages = loadMessagesFromJson();


    if(messages.isEmpty()){
      JOptionPane.showMessageDialog(null, "No messages sent!");
    }
    String input = JOptionPane.showInputDialog("Enter message ID " + (messages.size() - 1) + ":
");


    if (input == null) return;
      try {int id = Integer.parseInt(input);
      if (id < 0 || id >= messages.size()){
        JOptionPane.showMessageDialog(null, "Invalid message ID. Try again!");
        return;
      }
      String message = messages.get(id);
      String to = extractBetween(message, "to ", ":");
      String content = message.substring(message.indexOf(":") + 1).trim();


      JOptionPane.showMessageDialog(null,
          "Message ID: " + id + "\nRecipient: " + to + "\nMessage: " + content, "Message details",
          JOptionPane.INFORMATION_MESSAGE
      );
    }
      catch(NumberFormatException e){
        JOptionPane.showMessageDialog(null, "Please enter a valid message ID!");
      }
  }


  public void searchRecipientNumber(){
    List<String> messages = loadMessagesFromJson();
```

```
        if(messages.isEmpty()){
          JOptionPane.showMessageDialog(null, "No messages sent!");
        }
        String number = JOptionPane.showInputDialog("Enter recipient number: ");

        if(number == null || number.trim().isEmpty()) return;

        StringBuilder result = new StringBuilder();

        for (String msg : messages){
          String to = extractBetween(msg, "to ", ":");
          if(to.equalsIgnoreCase(number.trim())){
            result.append(msg).append("\n");
          }
        }
        if (result.length() == 0){
          JOptionPane.showMessageDialog(null, "No messages for: " + number);
        }

        JTextArea textArea = new JTextArea("Messages sent to: " + number + ":\n\n" +
result.toString());
        textArea.setEditable(false);
        JScrollPane scrollPane = new JScrollPane(textArea);
        scrollPane.setPreferredSize(new Dimension(500, 300));



        JOptionPane.showMessageDialog(null, scrollPane, "Search results",
JOptionPane.INFORMATION_MESSAGE);
      }
      private void searchHashNumber(){
        List<String> messages = loadMessagesFromJson();
```

```
    if(messages.isEmpty()){
      JOptionPane.showMessageDialog(null, "No messages sent!");
    }

    String input = JOptionPane.showInputDialog("Enter message Hash: ");

    if(input == null || input.trim().isEmpty()){
      try{
        int hashFind = Integer.parseInt(input);
        String messageFound = null;

        for (String msg : messages){
          if(msg.hashCode() == hashFind){
            messageFound = msg;
            break;
          }
        }
        if (messageFound == null){
          JOptionPane.showMessageDialog(null, "No message found using this Hash!");
        }
        int option = JOptionPane.showConfirmDialog(null, "Message: \n\n" + messageFound +
"\n\n Do you want to delete the message?", "Delete",
            JOptionPane.YES_NO_OPTION);
        if (option == JOptionPane.YES_OPTION){
          messages.remove(messageFound);
          JOptionPane.showMessageDialog(null, "Deleted");
        }
      }catch(NumberFormatException e){
        JOptionPane.showMessageDialog(null, "Invalid Hash entered, retry");
      }
    }


}
public void displayMessageReport(){
    List<String> messages = loadMessagesFromJson();


    if(messages.isEmpty()){
        JOptionPane.showMessageDialog(null, "No messages sent!");
    }
    StringBuilder report = new StringBuilder("Full sent message report: \n\n");


    for (int i = 0; i<messages.size(); i++){
        String msg = messages.get(i);
        String from = extractBetween(msg, "From: ", " to ");
        String to = extractBetween(msg, "to ", ":");
        String content = msg.substring(msg.indexOf(":") + 1).trim();


        report.append("Message ID: ").append(i).append("\n");
        report.append("Hash: ").append(msg.hashCode()).append("\n");
        report.append("From: ").append(from).append("\n");
        report.append("To: ").append(to).append("\n");
        report.append("Message: ").append(content).append("\n");
        report.append("-----------------------------------\n");
    }
    JTextArea textArea = new JTextArea(report.toString());
    textArea.setEditable(false);
    JScrollPane scrollPane = new JScrollPane(textArea);
    scrollPane.setPreferredSize(new Dimension(500, 400));
    JOptionPane.showMessageDialog(null, scrollPane, "Message report!",
JOptionPane.INFORMATION_MESSAGE);

  }


  public void moreOptionsMenu() throws FileNotFoundException{


    boolean inMoreOptionsMenu = true;


    while(inMoreOptionsMenu){
      String [] moreOptions = {
        "1 - List sent messages",
        "2 - List all disregarded messages",
        "3 - List all stored messages",
        "4 - List all message Hashes",
        "5 - List all message ID's",
        "6 - Back"
      };
      String choiceStr = (String) JOptionPane.showInputDialog(
      null, "Choose option: ",
      "More Options",
      JOptionPane.INFORMATION_MESSAGE, null,
      moreOptions,
      moreOptions[0]
          );


      if (choiceStr == null)
        break;


      int choice = Integer.parseInt(choiceStr.substring(0, 1));


      switch (choice){
```

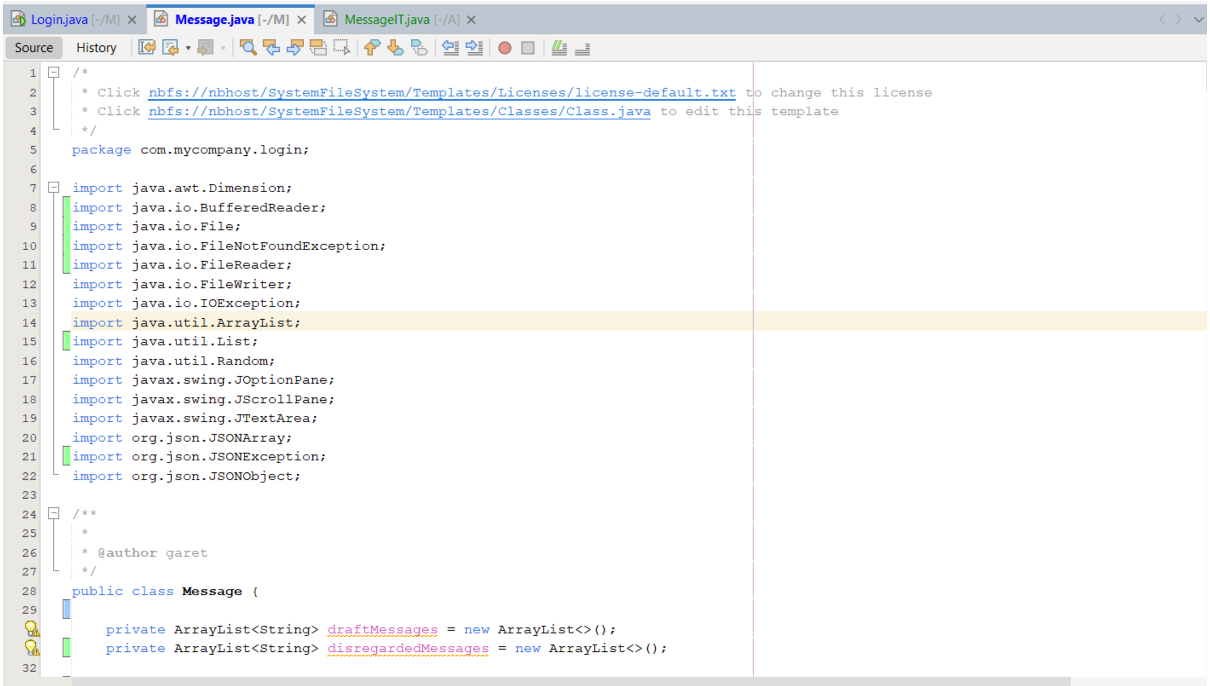
```
                case 1 -> listSentMessages();
                case 2 -> listDisregardedMessages();
                case 3 -> listStoredMessages();
                case 4 -> listMessageHashes();
                case 5 -> listMessageIDs();
                case 6 -> {
                    feedbackPrompt();
                    deleteMessagesOnExit();
                    inMoreOptionsMenu = false;
            }
                default -> JOptionPane.showMessageDialog(null, "Invalid option,retry!");
        }
    }

}

public void listSentMessages(){
    List<String> messages = loadMessagesFromJson();

    if(messages.isEmpty()){
        JOptionPane.showMessageDialog(null, "No messages sent!");
    }

    StringBuilder arrayDisplay = new StringBuilder("Sent messages: \n\n");

    for (int i = 0; i < messages.size(); i++){
        arrayDisplay.append("[").append(i).append("]").append(messages.get(i)).append("\n");
    }

    JTextArea textArea = new JTextArea(arrayDisplay.toString());
    textArea.setEditable(false);
    JScrollPane scrollPane = new JScrollPane(textArea);
    scrollPane.setPreferredSize(new Dimension(500, 300));



    JOptionPane.showMessageDialog(null, scrollPane, "Sent messages",
JOptionPane.INFORMATION_MESSAGE);
    }

  public void listDisregardedMessages(){
    List<String> messages = loadMessagesFromJson();

    if(messages.isEmpty()){
      JOptionPane.showMessageDialog(null, "No disregarded messages!");
    }

    StringBuilder disregarded = new StringBuilder("Disregarded messages: \n\n");

    for (int i = 0; i < disregardedMessages.size(); i++){

disregarded.append("[").append(i).append("]").append(disregardedMessages.get(i)).append("\n
");
    }

    JTextArea textArea = new JTextArea(disregarded.toString());
    textArea.setEditable(false);
    JScrollPane scrollPane = new JScrollPane(textArea);
    scrollPane.setPreferredSize(new Dimension(500, 300));



    JOptionPane.showMessageDialog(null, scrollPane, "Disregarded messages.",
JOptionPane.INFORMATION_MESSAGE);
  }
  public void listMessageHashes(){
    List<String> messages = loadMessagesFromJson();

    if(messages.isEmpty()){
      JOptionPane.showMessageDialog(null, "No messages!");
    }

    StringBuilder output = new StringBuilder("Message hashes: \n\n");

    for(int i = 0; i < messages.size(); i++){
      String message = messages.get(i);
      int hash = message.hashCode();
      output.append("[").append(i).append("]").append(hash).append("\n");
    }

    JTextArea textArea = new JTextArea(output.toString());
    textArea.setEditable(false);
    JScrollPane scrollPane = new JScrollPane(textArea);
    scrollPane.setPreferredSize(new Dimension(500, 300));


    JOptionPane.showMessageDialog(null, scrollPane, "Message Hashes", JOptionPane.INFORMATION_MESSAGE);
  }

  public void listMessageIDs(){
    List<String> messages = loadMessagesFromJson();
```

```
        if(messages.isEmpty()){
            JOptionPane.showMessageDialog(null, "No messages!");
        }

        StringBuilder output = new StringBuilder("Message ID's: \n\n");

        for(int i = 0; i < messages.size(); i++){
            output.append("Message ID's: ").append(i).append("\n");
        }
        JTextArea textArea = new JTextArea(output.toString());
        textArea.setEditable(false);
        JScrollPane scrollPane = new JScrollPane(textArea);
        scrollPane.setPreferredSize(new Dimension(500, 300));



        JOptionPane.showMessageDialog(null,scrollPane, "Message ID's",
JOptionPane.INFORMATION_MESSAGE);
    }




    @SuppressWarnings("empty-statement")



private String checkMessageID(){
        Random rand = new Random();
        StringBuilder id = new StringBuilder();
        for (int i = 0; i < 10; i++){
            id.append(rand.nextInt(10));
        }
        return id.toString();
```

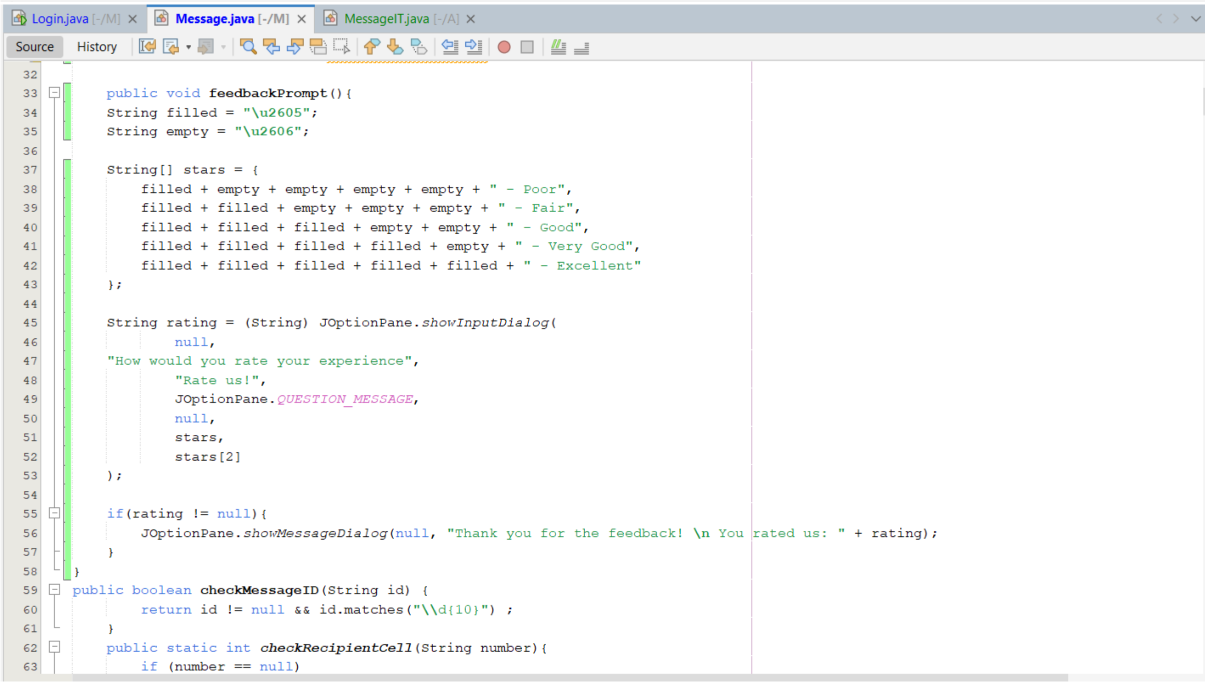
```
}
private String createMessageHash(String messageId, int messageNumber, String messageContent){
    String firstTwoDigits = messageId.length() >= 2 ? messageId.substring(0, 2) : messageId;


    String[] words = messageContent.trim().split("\\s+");
    String firstWord = words.length > 0 ? words[0] : "";
    String lastWord = words.length > 1 ? words[words.length - 1] : firstWord;


    return firstTwoDigits + ":" + messageNumber + ":" + firstWord + lastWord;
}
private void printMessage(){
    List<String> messages = loadMessagesFromJson();


    if (messages.isEmpty()){
        JOptionPane.showMessageDialog(null, "No messages sent");
        return;
    }
    StringBuilder messageList = new StringBuilder("Sent messages:\n\n");


    for (String message : messages){
        messageList.append(message).append("\n\n");
    }


    JTextArea textArea = new JTextArea(messageList.toString());
    textArea.setEditable(false);
    JScrollPane scrollPane = new JScrollPane(textArea);
    scrollPane.setPreferredSize(new Dimension(500, 400));



    JOptionPane.showMessageDialog(null, scrollPane, "Chat history", JOptionPane.INFORMATION_MESSAGE);
```

```
}



public void listStoredMessages() throws FileNotFoundException{


      JSONArray messages = new JSONArray();


  try(BufferedReader reader = new BufferedReader(new FileReader("messages.json"))){
    String line;
    while((line = reader.readLine()) != null){
      JSONObject obj = new JSONObject(line);
      messages.put(obj);
    }
  }catch(FileNotFoundException e){
    JOptionPane.showMessageDialog(null, "JSON file not found!");
  }catch(IOException e){
    JOptionPane.showMessageDialog(null, "Error reading file!" + e.getMessage());
  }


     if(messages.isEmpty()){
      JOptionPane.showMessageDialog(null, "No stored messages!");
    }
     StringBuilder output = new StringBuilder("Stored Messages: \n\n");


     for(int i = 0; i<messages.length();i++){
       JSONObject msg = messages.getJSONObject(i);
       output.append("[").append(i).append("]");
       output.append("From: ").append(msg.optString("from")).append(" ");
       output.append("To: ").append(msg.optString("to")).append("\n");
       output.append("Message: ").append(msg.optString("message")).append("\n");

       output.append("---------------------\n");



     }

     JTextArea textArea = new JTextArea(output.toString());

     textArea.setEditable(false);

     JScrollPane scrollPane = new JScrollPane(textArea);

     scrollPane.setPreferredSize(new Dimension(500, 300));



     JOptionPane.showMessageDialog(null, scrollPane, "Stored Messages",
JOptionPane.INFORMATION_MESSAGE);

    }


 private void deleteMessagesOnExit(){

   File messagesFile = new File("messages.json");

   if(messagesFile.exists()){

     if(messagesFile.delete()){

       JOptionPane.showMessageDialog(null, "Messages deleted!");

     }else{

       JOptionPane.showMessageDialog(null, "Deletion failed!");

     }

   }

 }


}
```

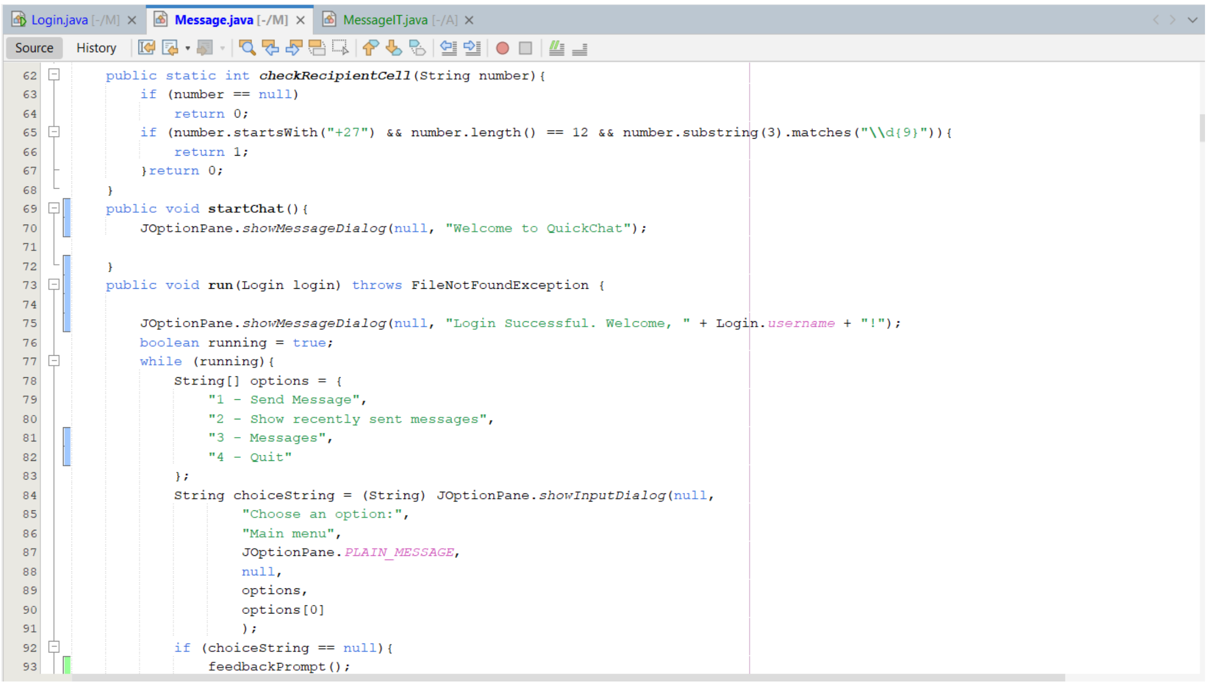
Screenshots of code:



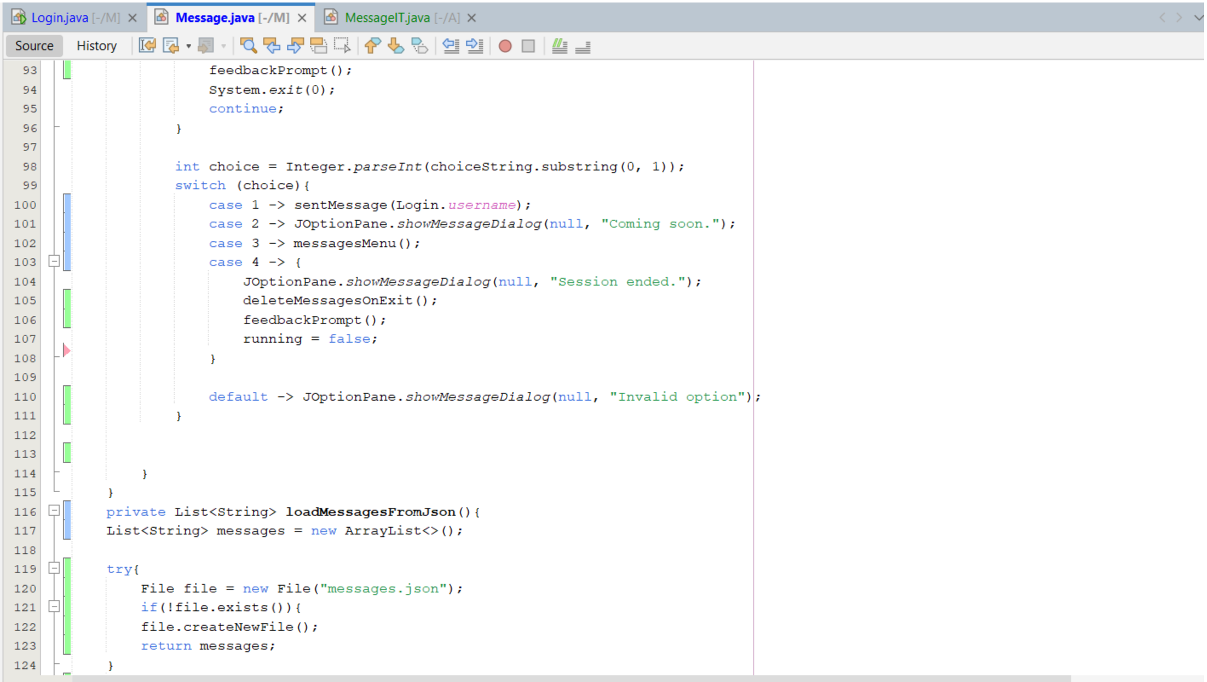
```
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template
 */
package com.mycompany.login;

import java.awt.Dimension;
import java.io.BufferedReader;
import java.io.File;
import java.io.FileNotFoundException;
import java.io.FileReader;
import java.io.FileWriter;
import java.io.IOException;
import java.util.ArrayList;
import java.util.List;
import java.util.Random;
import javax.swing.JOptionPane;
import javax.swing.JScrollPane;
import javax.swing.JTextArea;
import org.json.JSONArray;
import org.json.JSONException;
import org.json.JSONObject;

/**
 *
 * @author garet
 */
public class Message {

    private ArrayList<String> draftMessages = new ArrayList<>();
    private ArrayList<String> disregardedMessages = new ArrayList<>();

```



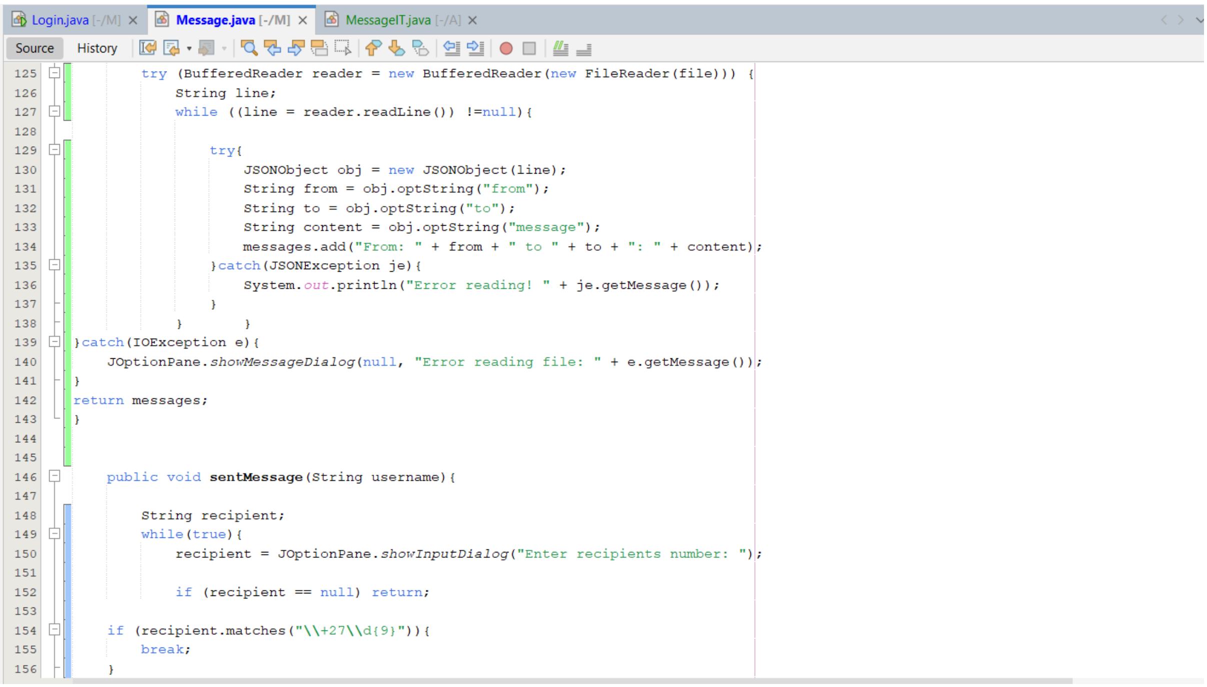
```

    public void feedbackPrompt(){
    String filled = "\u2605";
    String empty = "\u2606";

    String[] stars = {
        filled + empty + empty + empty + empty + " - Poor",
        filled + filled + empty + empty + empty + " - Fair",
        filled + filled + filled + empty + empty + " - Good",
        filled + filled + filled + filled + empty + " - Very Good",
        filled + filled + filled + filled + filled + " - Excellent"
    };

    String rating = (String) JOptionPane.showInputDialog(
            null,
    "How would you rate your experience",
            "Rate us!",
            JOptionPane.QUESTION_MESSAGE,
            null,
            stars,
            stars[2]
    );

    if(rating != null){
        JOptionPane.showMessageDialog(null, "Thank you for the feedback! \n You rated us: " + rating);
    }
}
public boolean checkMessageID(String id) {
        return id != null && id.matches("\\d{10}") ;
    }
    public static int checkRecipientCell(String number){
        if (number == null)
```



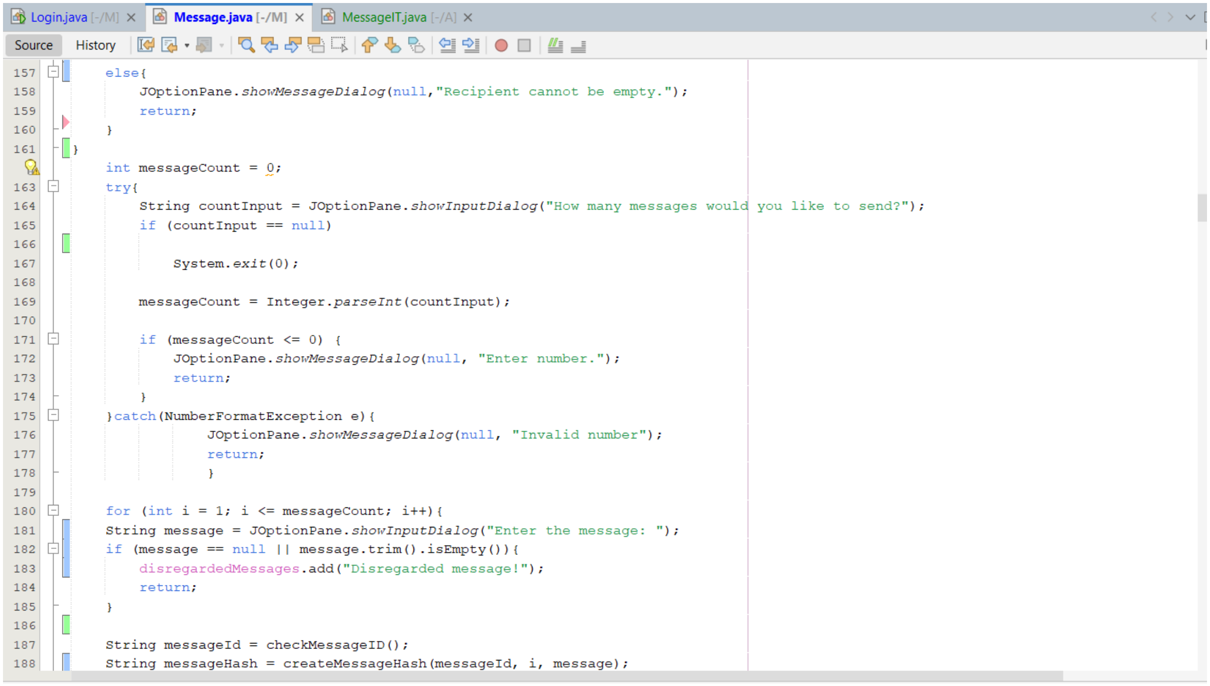
```java
    public static int checkRecipientCell(String number){
        if (number == null)
            return 0;
        if (number.startsWith("+27") && number.length() == 12 && number.substring(3).matches("\\d{9}")){
            return 1;
        }return 0;
    }
    public void startChat(){
        JOptionPane.showMessageDialog(null, "Welcome to QuickChat");

    }
    public void run(Login login) throws FileNotFoundException {

        JOptionPane.showMessageDialog(null, "Login Successful. Welcome, " + Login.username + "!");
        boolean running = true;
        while (running){
            String[] options = {
                "1 - Send Message",
                "2 - Show recently sent messages",
                "3 - Messages",
                "4 - Quit"
            };
            String choiceString = (String) JOptionPane.showInputDialog(null,
                    "Choose an option:",
                    "Main menu",
                    JOptionPane.PLAIN_MESSAGE,
                    null,
                    options,
                    options[0]
                    );
            if (choiceString == null){
                feedbackPrompt();
```



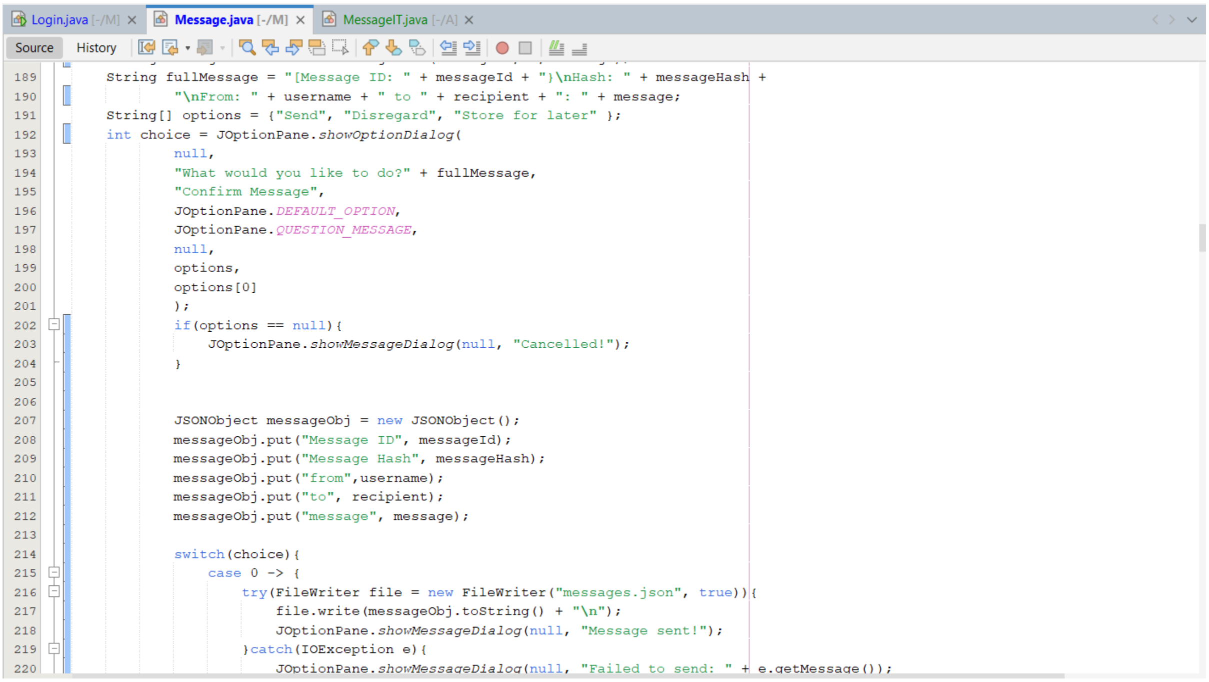
```java
                feedbackPrompt();
                System.exit(0);
                continue;
            }

            int choice = Integer.parseInt(choiceString.substring(0, 1));
            switch (choice){
                case 1 -> sentMessage(Login.username);
                case 2 -> JOptionPane.showMessageDialog(null, "Coming soon.");
                case 3 -> messagesMenu();
                case 4 -> {
                    JOptionPane.showMessageDialog(null, "Session ended.");
                    deleteMessagesOnExit();
                    feedbackPrompt();
                    running = false;
                }

                default -> JOptionPane.showMessageDialog(null, "Invalid option");
            }


        }
    }
    private List<String> loadMessagesFromJson(){
    List<String> messages = new ArrayList<>();

    try{
        File file = new File("messages.json");
        if(!file.exists()){
        file.createNewFile();
        return messages;
    }
```



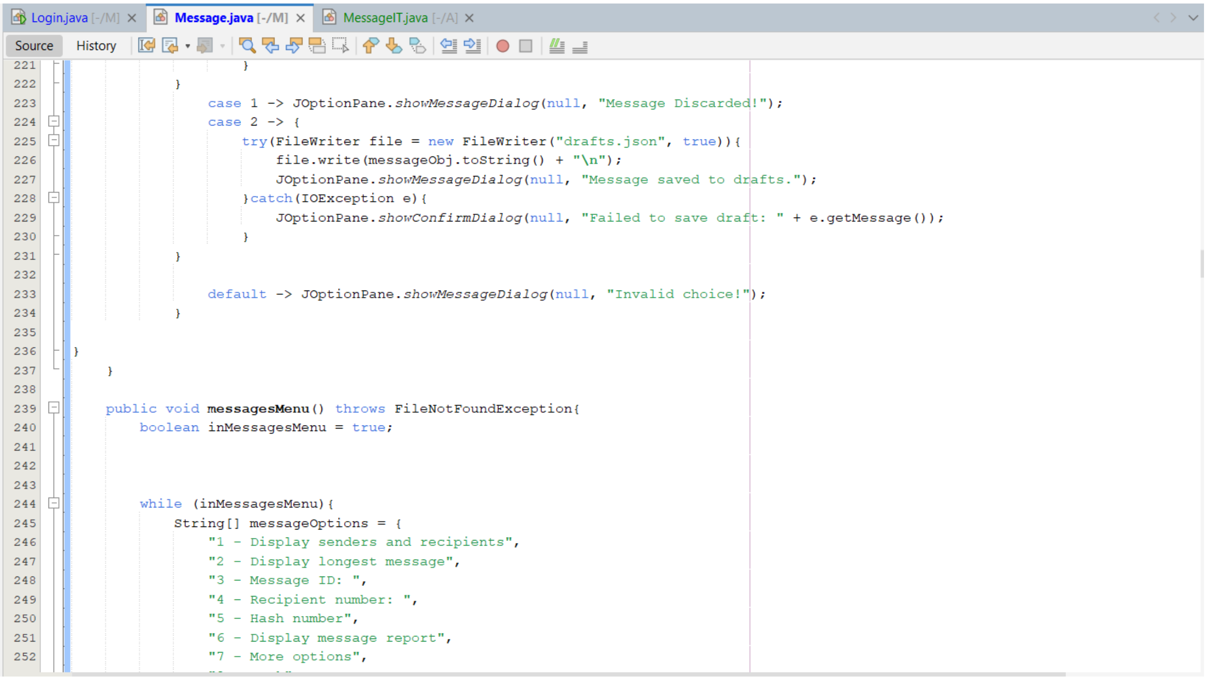
```
        try (BufferedReader reader = new BufferedReader(new FileReader(file))) {
            String line;
            while ((line = reader.readLine()) !=null){

                try{
                    JSONObject obj = new JSONObject(line);
                    String from = obj.optString("from");
                    String to = obj.optString("to");
                    String content = obj.optString("message");
                    messages.add("From: " + from + " to " + to + ": " + content);
                }catch(JSONException je){
                    System.out.println("Error reading! " + je.getMessage());
                }
            }        }
}catch(IOException e){
    JOptionPane.showMessageDialog(null, "Error reading file: " + e.getMessage());
}
return messages;
}


    public void sentMessage(String username){

        String recipient;
        while(true){
            recipient = JOptionPane.showInputDialog("Enter recipients number: ");

            if (recipient == null) return;

    if (recipient.matches("\\+27\\d{9}")){
        break;
    }
```



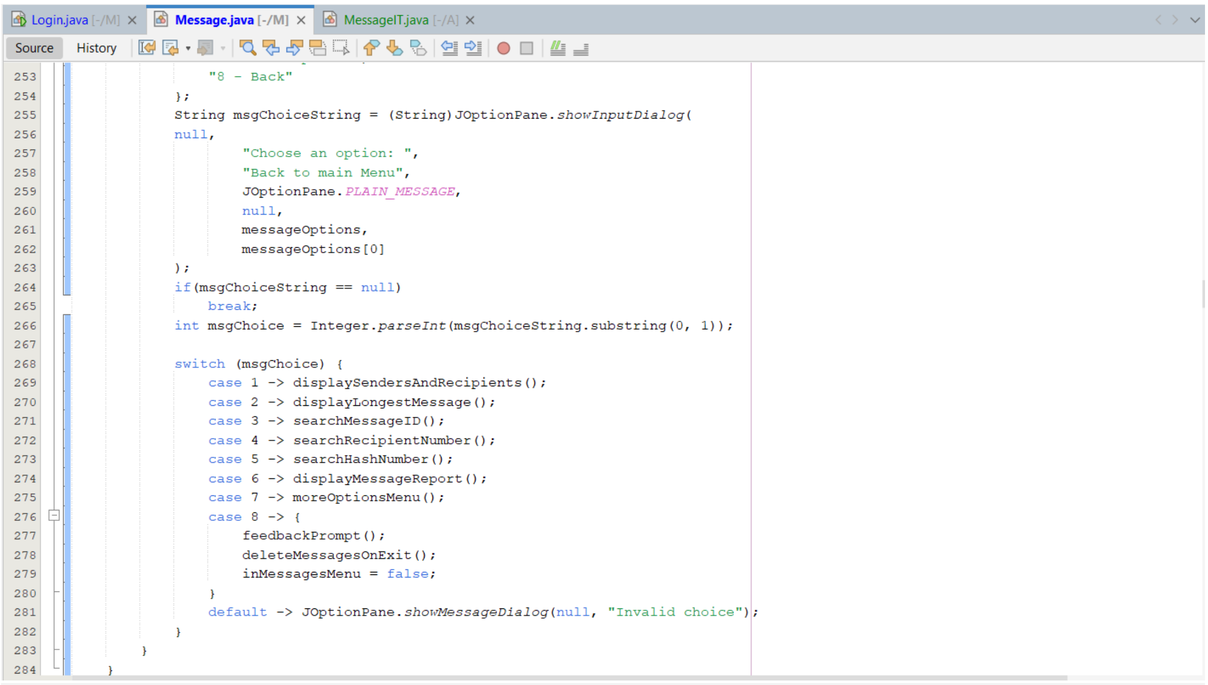
```
    else{
        JOptionPane.showMessageDialog(null,"Recipient cannot be empty.");
        return;
    }
}
    int messageCount = 0;
    try{
        String countInput = JOptionPane.showInputDialog("How many messages would you like to send?");
        if (countInput == null)

            System.exit(0);

        messageCount = Integer.parseInt(countInput);

        if (messageCount <= 0) {
            JOptionPane.showMessageDialog(null, "Enter number.");
            return;
        }
    }catch(NumberFormatException e){
                JOptionPane.showMessageDialog(null, "Invalid number");
                return;
                }

    for (int i = 1; i <= messageCount; i++){
    String message = JOptionPane.showInputDialog("Enter the message: ");
    if (message == null || message.trim().isEmpty()){
        disregardedMessages.add("Disregarded message!");
        return;
    }

    String messageId = checkMessageID();
    String messageHash = createMessageHash(messageId, i, message);
```



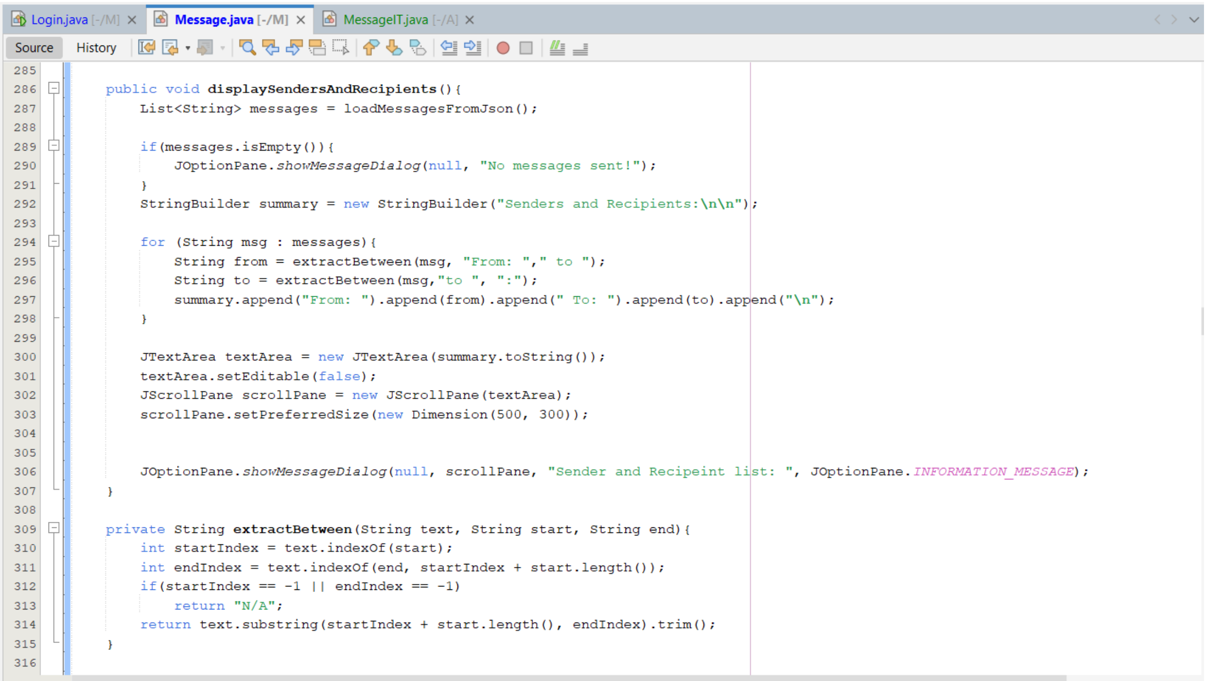
```java
    String fullMessage = "[Message ID: " + messageId + "}\nHash: " + messageHash +
            "\nFrom: " + username + " to " + recipient + ": " + message;
    String[] options = {"Send", "Disregard", "Store for later" };
    int choice = JOptionPane.showOptionDialog(
            null,
            "What would you like to do?" + fullMessage,
            "Confirm Message",
            JOptionPane.DEFAULT_OPTION,
            JOptionPane.QUESTION_MESSAGE,
            null,
            options,
            options[0]
            );
            if(options == null){
                JOptionPane.showMessageDialog(null, "Cancelled!");
            }


            JSONObject messageObj = new JSONObject();
            messageObj.put("Message ID", messageId);
            messageObj.put("Message Hash", messageHash);
            messageObj.put("from",username);
            messageObj.put("to", recipient);
            messageObj.put("message", message);

            switch(choice){
                case 0 -> {
                    try(FileWriter file = new FileWriter("messages.json", true)){
                        file.write(messageObj.toString() + "\n");
                        JOptionPane.showMessageDialog(null, "Message sent!");
                    }catch(IOException e){
                        JOptionPane.showMessageDialog(null, "Failed to send: " + e.getMessage());
```



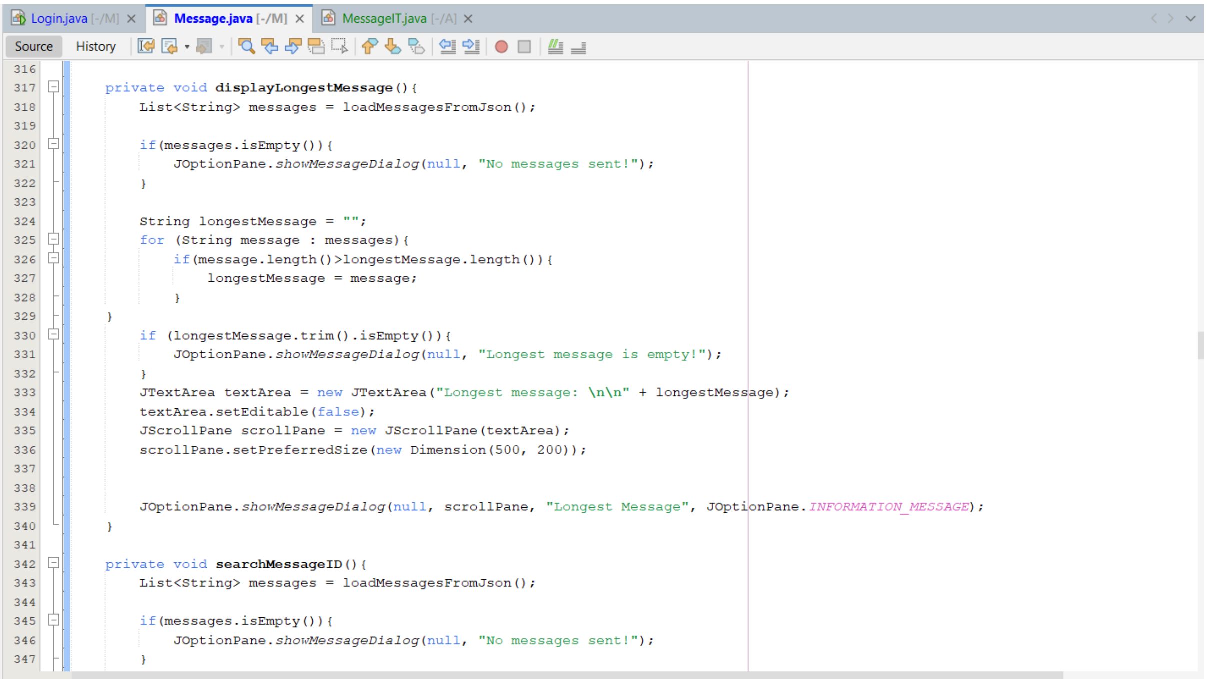
```java
                    }
            }
                case 1 -> JOptionPane.showMessageDialog(null, "Message Discarded!");
                case 2 -> {
                    try(FileWriter file = new FileWriter("drafts.json", true)){
                        file.write(messageObj.toString() + "\n");
                        JOptionPane.showMessageDialog(null, "Message saved to drafts.");
                    }catch(IOException e){
                        JOptionPane.showConfirmDialog(null, "Failed to save draft: " + e.getMessage());
                    }
            }

                default -> JOptionPane.showMessageDialog(null, "Invalid choice!");
            }

}
    }

    public void messagesMenu() throws FileNotFoundException{
        boolean inMessagesMenu = true;



        while (inMessagesMenu){
            String[] messageOptions = {
                "1 - Display senders and recipients",
                "2 - Display longest message",
                "3 - Message ID: ",
                "4 - Recipient number: ",
                "5 - Hash number",
                "6 - Display message report",
                "7 - More options",
```



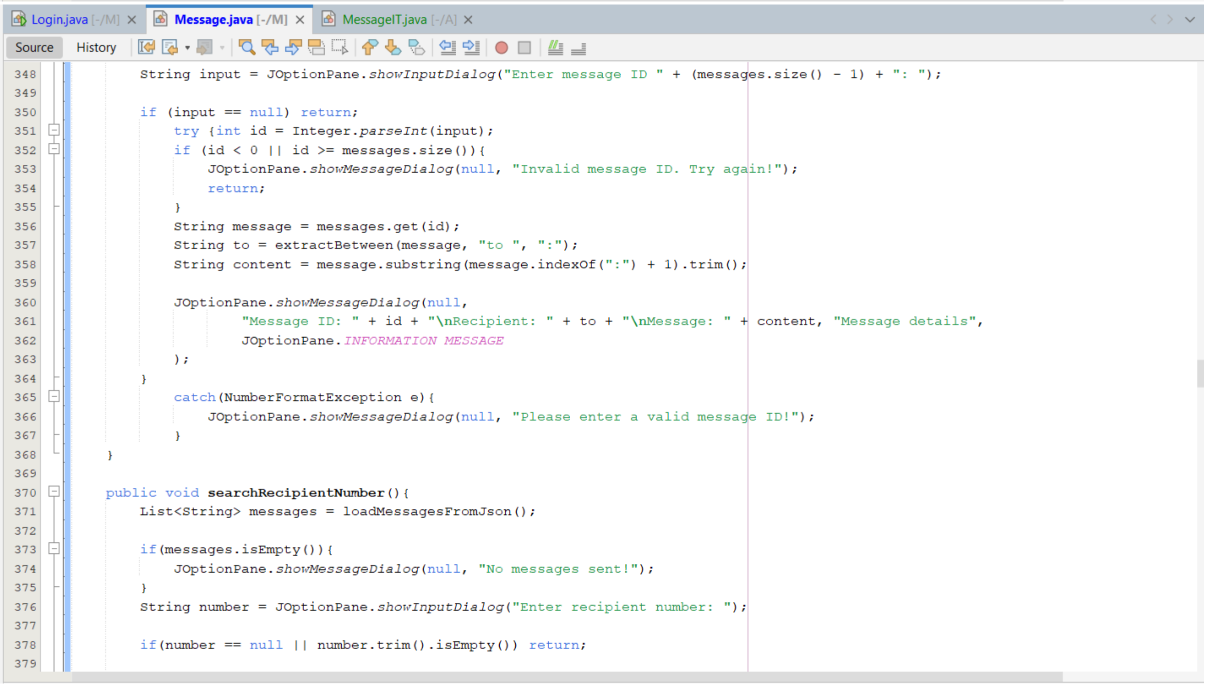
```java
                "8 - Back"
            );
            String msgChoiceString = (String)JOptionPane.showInputDialog(
            null,
                    "Choose an option: ",
                    "Back to main Menu",
                    JOptionPane.PLAIN_MESSAGE,
                    null,
                    messageOptions,
                    messageOptions[0]
            );
            if(msgChoiceString == null)
                break;
            int msgChoice = Integer.parseInt(msgChoiceString.substring(0, 1));

            switch (msgChoice) {
                case 1 -> displaySendersAndRecipients();
                case 2 -> displayLongestMessage();
                case 3 -> searchMessageID();
                case 4 -> searchRecipientNumber();
                case 5 -> searchHashNumber();
                case 6 -> displayMessageReport();
                case 7 -> moreOptionsMenu();
                case 8 -> {
                    feedbackPrompt();
                    deleteMessagesOnExit();
                    inMessagesMenu = false;
                }
                default -> JOptionPane.showMessageDialog(null, "Invalid choice");
            }
        }
    }
```



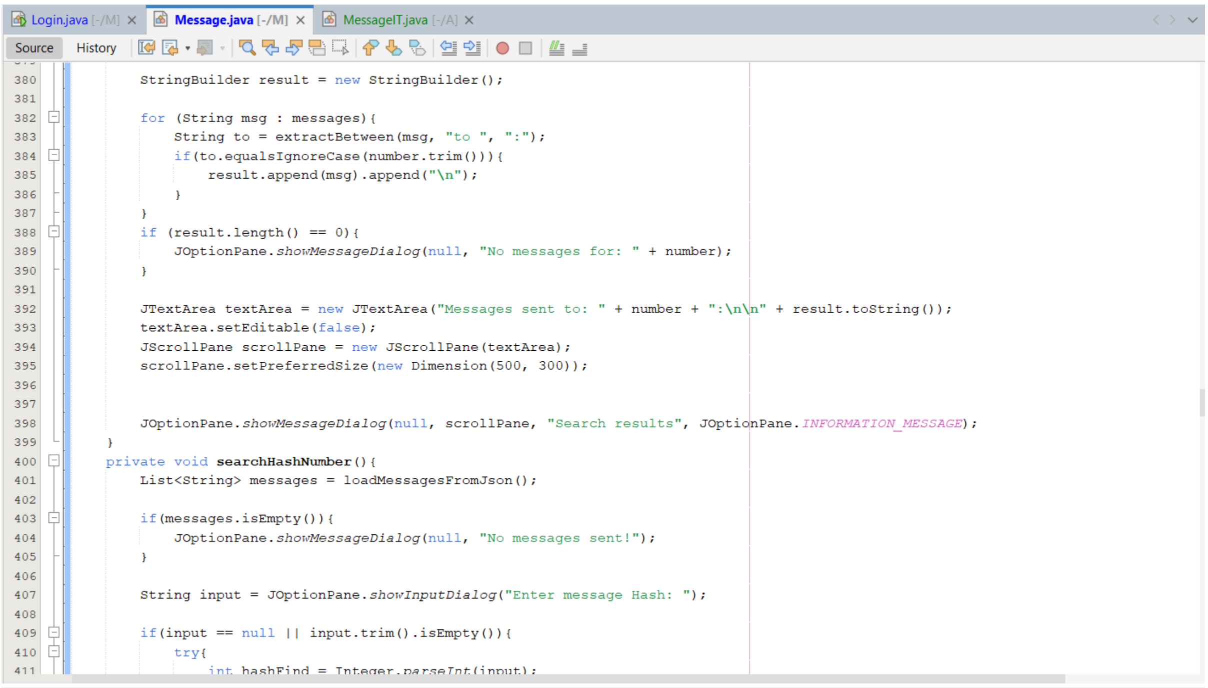
```java

    public void displaySendersAndRecipients(){
        List<String> messages = loadMessagesFromJson();

        if(messages.isEmpty()){
            JOptionPane.showMessageDialog(null, "No messages sent!");
        }
        StringBuilder summary = new StringBuilder("Senders and Recipients:\n\n");

        for (String msg : messages){
            String from = extractBetween(msg, "From: "," to ");
            String to = extractBetween(msg,"to ", ":");
            summary.append("From: ").append(from).append(" To: ").append(to).append("\n");
        }

        JTextArea textArea = new JTextArea(summary.toString());
        textArea.setEditable(false);
        JScrollPane scrollPane = new JScrollPane(textArea);
        scrollPane.setPreferredSize(new Dimension(500, 300));


        JOptionPane.showMessageDialog(null, scrollPane, "Sender and Recipeint list: ", JOptionPane.INFORMATION_MESSAGE);
    }

    private String extractBetween(String text, String start, String end){
        int startIndex = text.indexOf(start);
        int endIndex = text.indexOf(end, startIndex + start.length());
        if(startIndex == -1 || endIndex == -1)
            return "N/A";
        return text.substring(startIndex + start.length(), endIndex).trim();
    }

```



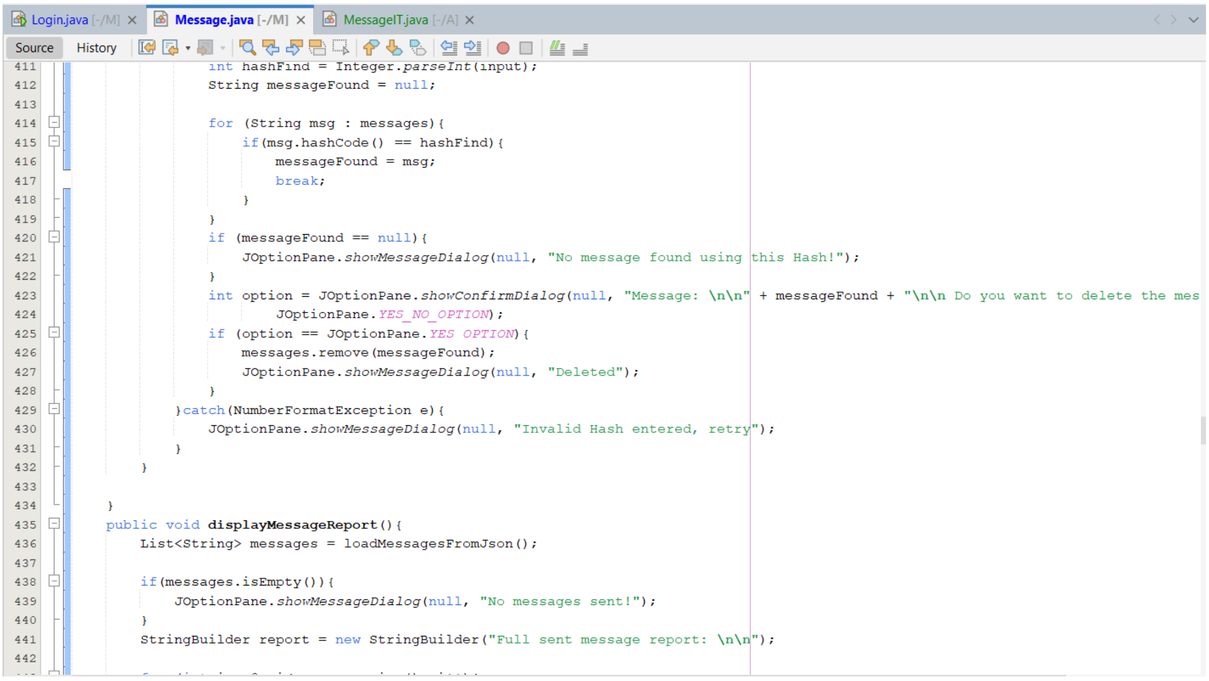
```
    private void displayLongestMessage(){
        List<String> messages = loadMessagesFromJson();

        if(messages.isEmpty()){
            JOptionPane.showMessageDialog(null, "No messages sent!");
        }

        String longestMessage = "";
        for (String message : messages){
            if(message.length()>longestMessage.length()){
                longestMessage = message;
            }
    }
        if (longestMessage.trim().isEmpty()){
            JOptionPane.showMessageDialog(null, "Longest message is empty!");
        }
        JTextArea textArea = new JTextArea("Longest message: \n\n" + longestMessage);
        textArea.setEditable(false);
        JScrollPane scrollPane = new JScrollPane(textArea);
        scrollPane.setPreferredSize(new Dimension(500, 200));


        JOptionPane.showMessageDialog(null, scrollPane, "Longest Message", JOptionPane.INFORMATION_MESSAGE);
    }

    private void searchMessageID(){
        List<String> messages = loadMessagesFromJson();

        if(messages.isEmpty()){
            JOptionPane.showMessageDialog(null, "No messages sent!");
        }
```



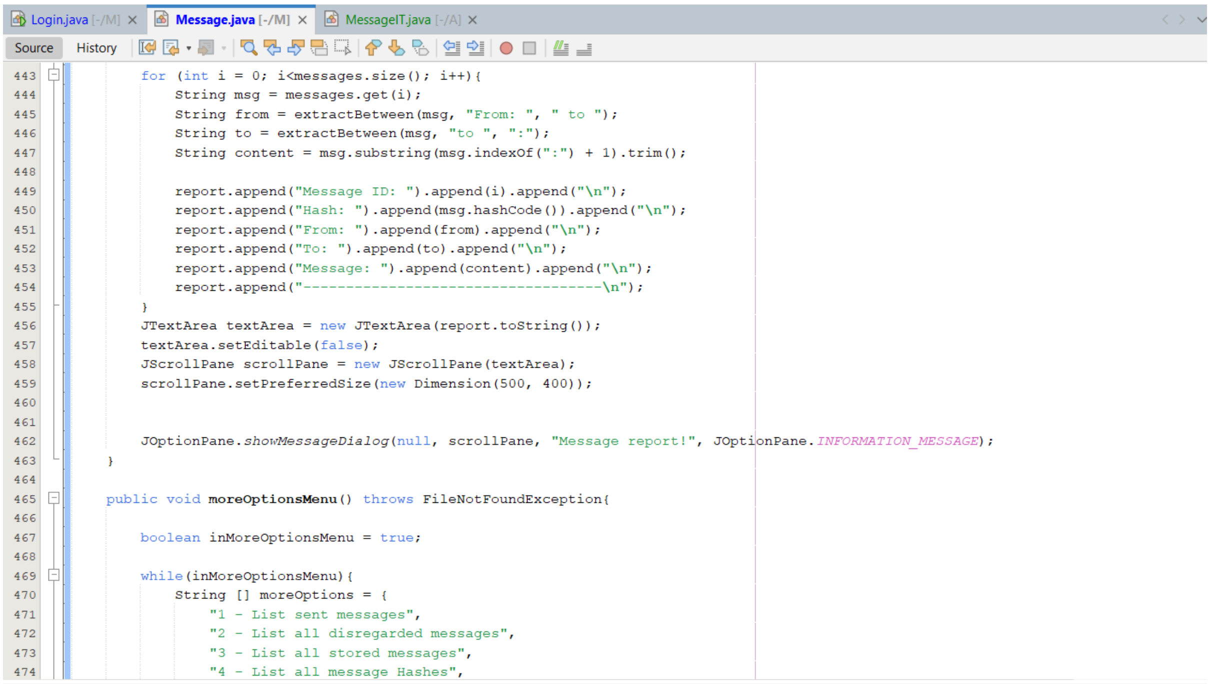
```
        String input = JOptionPane.showInputDialog("Enter message ID " + (messages.size() - 1) + ": ");

        if (input == null) return;
            try {int id = Integer.parseInt(input);
            if (id < 0 || id >= messages.size()){
                JOptionPane.showMessageDialog(null, "Invalid message ID. Try again!");
                return;
            }
            String message = messages.get(id);
            String to = extractBetween(message, "to ", ":");
            String content = message.substring(message.indexOf(":") + 1).trim();

            JOptionPane.showMessageDialog(null,
                    "Message ID: " + id + "\nRecipient: " + to + "\nMessage: " + content, "Message details",
                    JOptionPane.INFORMATION_MESSAGE
            );
        }
            catch(NumberFormatException e){
                JOptionPane.showMessageDialog(null, "Please enter a valid message ID!");
            }
    }

    public void searchRecipientNumber(){
        List<String> messages = loadMessagesFromJson();

        if(messages.isEmpty()){
            JOptionPane.showMessageDialog(null, "No messages sent!");
        }
        String number = JOptionPane.showInputDialog("Enter recipient number: ");

        if(number == null || number.trim().isEmpty()) return;

```



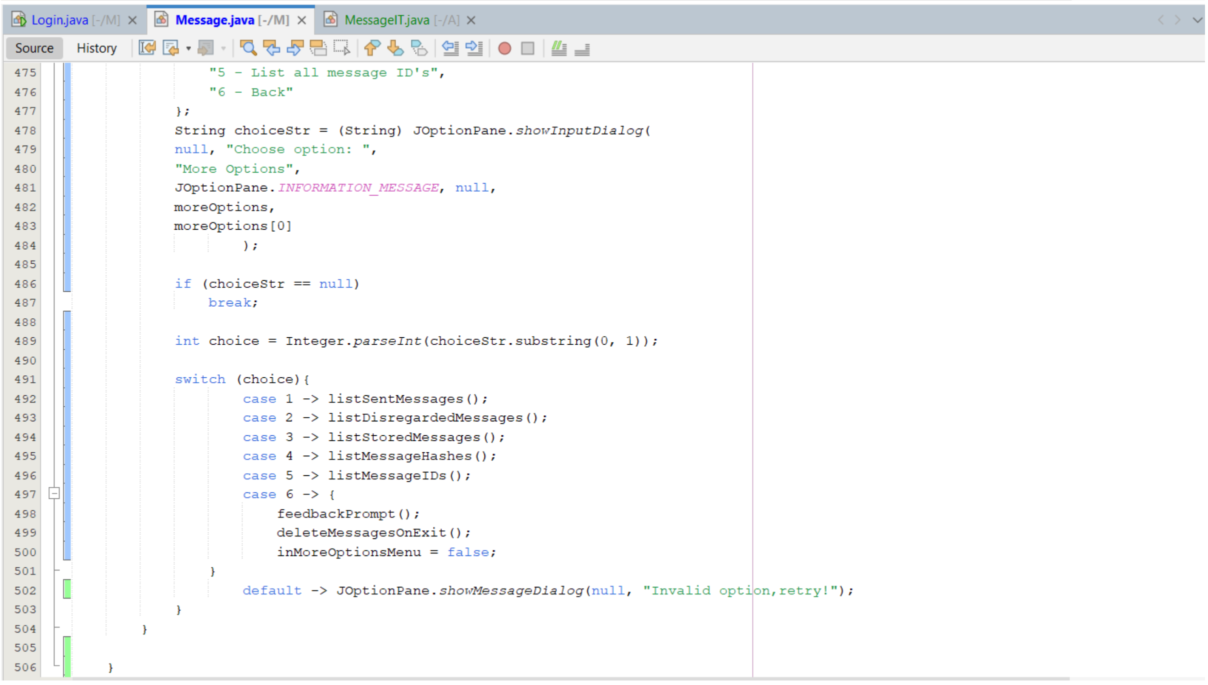
```java
        StringBuilder result = new StringBuilder();

        for (String msg : messages){
            String to = extractBetween(msg, "to ", ":");
            if(to.equalsIgnoreCase(number.trim())){
                result.append(msg).append("\n");
            }
        }
        if (result.length() == 0){
            JOptionPane.showMessageDialog(null, "No messages for: " + number);
        }

        JTextArea textArea = new JTextArea("Messages sent to: " + number + ":\n\n" + result.toString());
        textArea.setEditable(false);
        JScrollPane scrollPane = new JScrollPane(textArea);
        scrollPane.setPreferredSize(new Dimension(500, 300));


        JOptionPane.showMessageDialog(null, scrollPane, "Search results", JOptionPane.INFORMATION_MESSAGE);
    }
    private void searchHashNumber(){
        List<String> messages = loadMessagesFromJson();

        if(messages.isEmpty()){
            JOptionPane.showMessageDialog(null, "No messages sent!");
        }

        String input = JOptionPane.showInputDialog("Enter message Hash: ");

        if(input == null || input.trim().isEmpty()){
            try{
                int hashFind = Integer.parseInt(input);
```



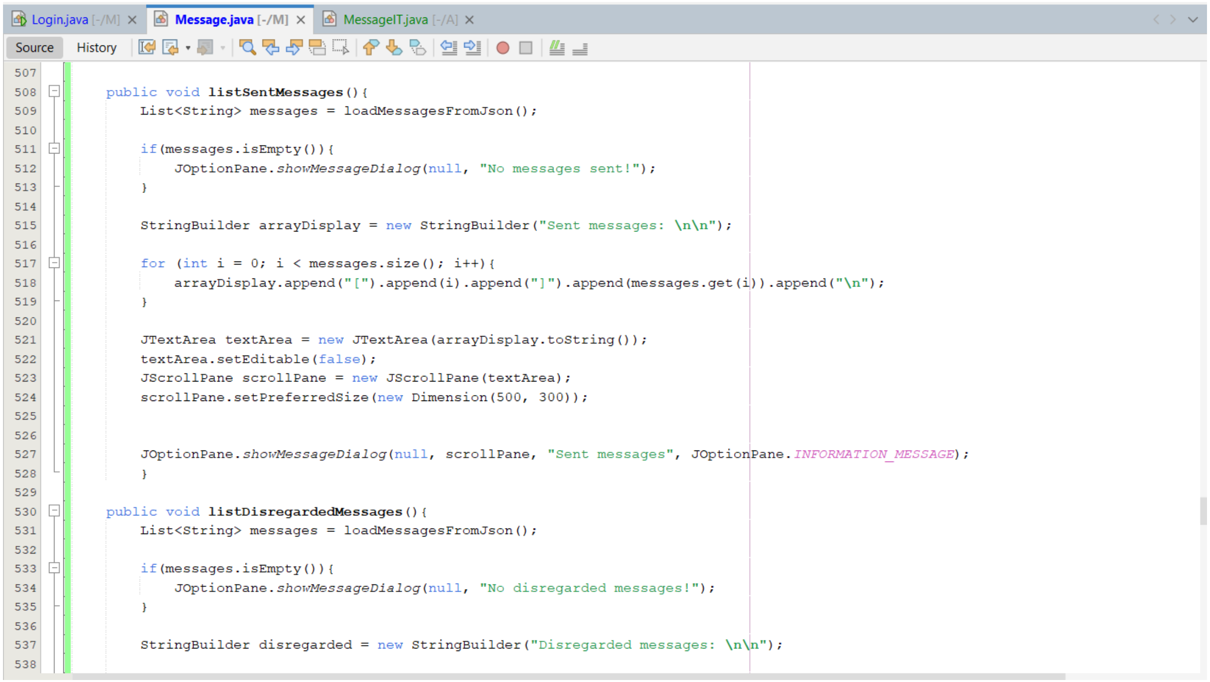
```java
                int hashFind = Integer.parseInt(input);
                String messageFound = null;

                for (String msg : messages){
                    if(msg.hashCode() == hashFind){
                        messageFound = msg;
                        break;
                    }
                }
                if (messageFound == null){
                    JOptionPane.showMessageDialog(null, "No message found using this Hash!");
                }
                int option = JOptionPane.showConfirmDialog(null, "Message: \n\n" + messageFound + "\n\n Do you want to delete the mes
                        JOptionPane.YES_NO_OPTION);
                if (option == JOptionPane.YES_OPTION){
                    messages.remove(messageFound);
                    JOptionPane.showMessageDialog(null, "Deleted");
                }
            }catch(NumberFormatException e){
                JOptionPane.showMessageDialog(null, "Invalid Hash entered, retry");
            }
        }

    }
    public void displayMessageReport(){
        List<String> messages = loadMessagesFromJson();

        if(messages.isEmpty()){
            JOptionPane.showMessageDialog(null, "No messages sent!");
        }
        StringBuilder report = new StringBuilder("Full sent message report: \n\n");

```



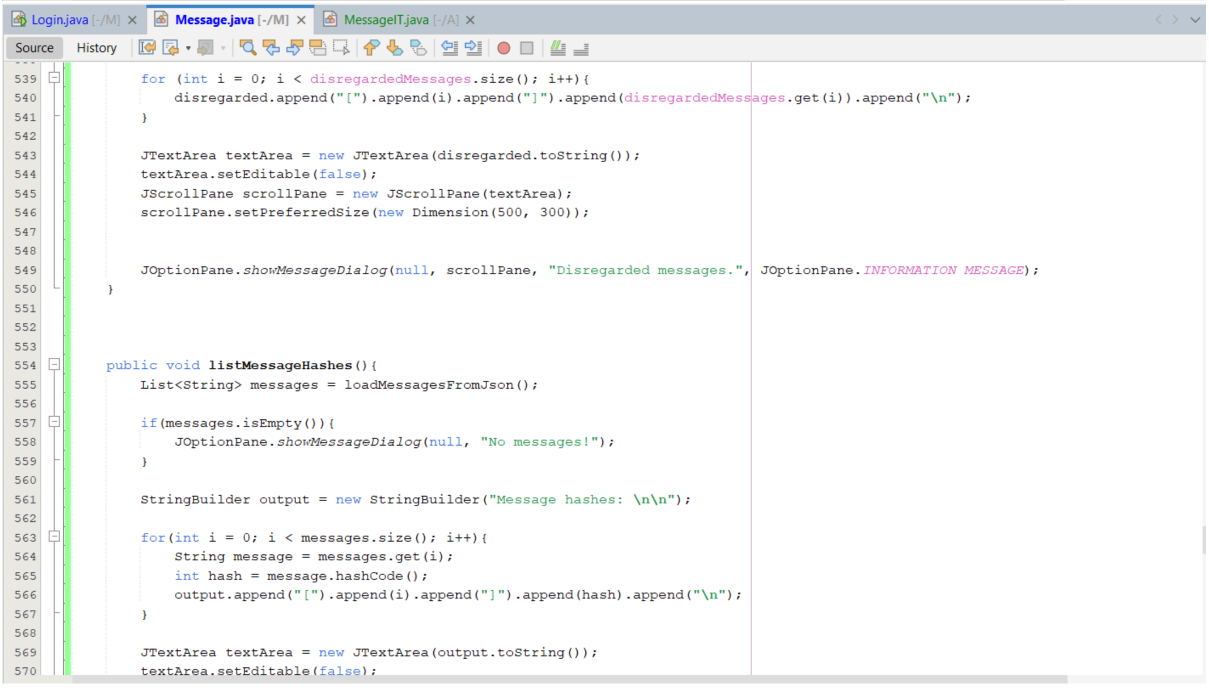
```java
        for (int i = 0; i<messages.size(); i++){
            String msg = messages.get(i);
            String from = extractBetween(msg, "From: ", " to ");
            String to = extractBetween(msg, "to ", ":");
            String content = msg.substring(msg.indexOf(":") + 1).trim();

            report.append("Message ID: ").append(i).append("\n");
            report.append("Hash: ").append(msg.hashCode()).append("\n");
            report.append("From: ").append(from).append("\n");
            report.append("To: ").append(to).append("\n");
            report.append("Message: ").append(content).append("\n");
            report.append("-----------------------------------\n");
        }
        JTextArea textArea = new JTextArea(report.toString());
        textArea.setEditable(false);
        JScrollPane scrollPane = new JScrollPane(textArea);
        scrollPane.setPreferredSize(new Dimension(500, 400));


        JOptionPane.showMessageDialog(null, scrollPane, "Message report!", JOptionPane.INFORMATION_MESSAGE);
    }

    public void moreOptionsMenu() throws FileNotFoundException{

        boolean inMoreOptionsMenu = true;

        while(inMoreOptionsMenu){
            String [] moreOptions = {
                "1 - List sent messages",
                "2 - List all disregarded messages",
                "3 - List all stored messages",
                "4 - List all message Hashes",
```



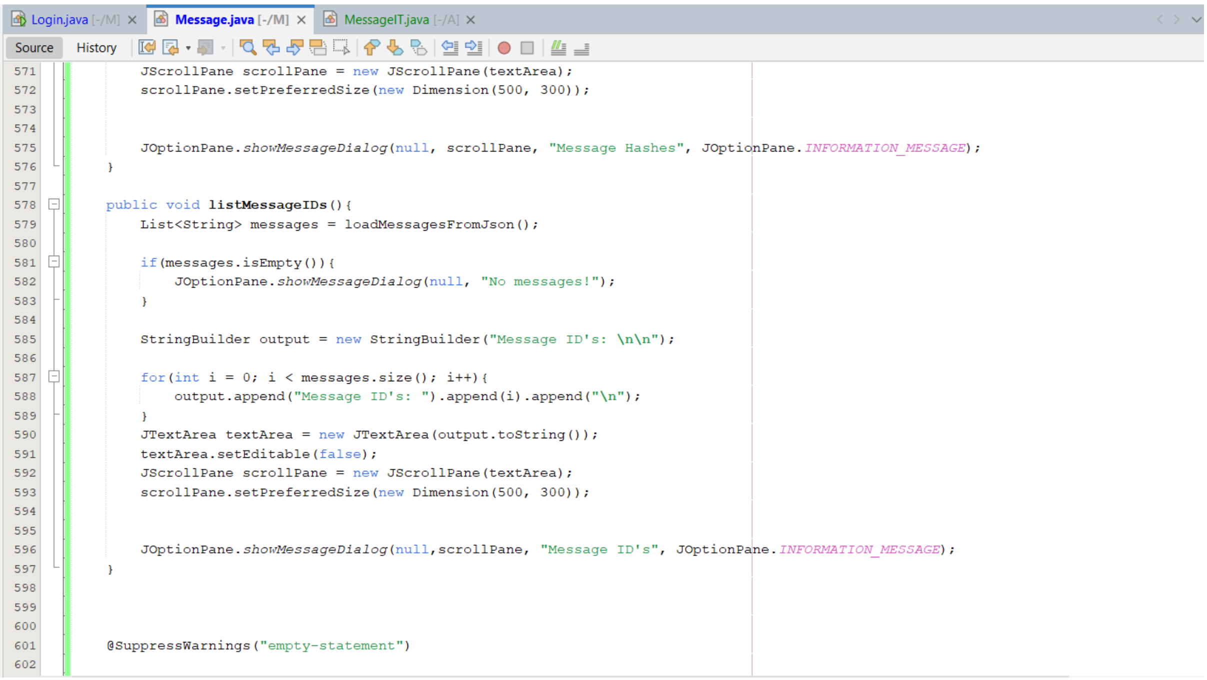
```java
                "5 - List all message ID's",
                "6 - Back"
            };
            String choiceStr = (String) JOptionPane.showInputDialog(
            null, "Choose option: ",
            "More Options",
            JOptionPane.INFORMATION_MESSAGE, null,
            moreOptions,
            moreOptions[0]
                    );

            if (choiceStr == null)
                break;

            int choice = Integer.parseInt(choiceStr.substring(0, 1));

            switch (choice){
                    case 1 -> listSentMessages();
                    case 2 -> listDisregardedMessages();
                    case 3 -> listStoredMessages();
                    case 4 -> listMessageHashes();
                    case 5 -> listMessageIDs();
                    case 6 -> {
                        feedbackPrompt();
                        deleteMessagesOnExit();
                        inMoreOptionsMenu = false;
                }
                    default -> JOptionPane.showMessageDialog(null, "Invalid option,retry!");
            }
        }

    }
```



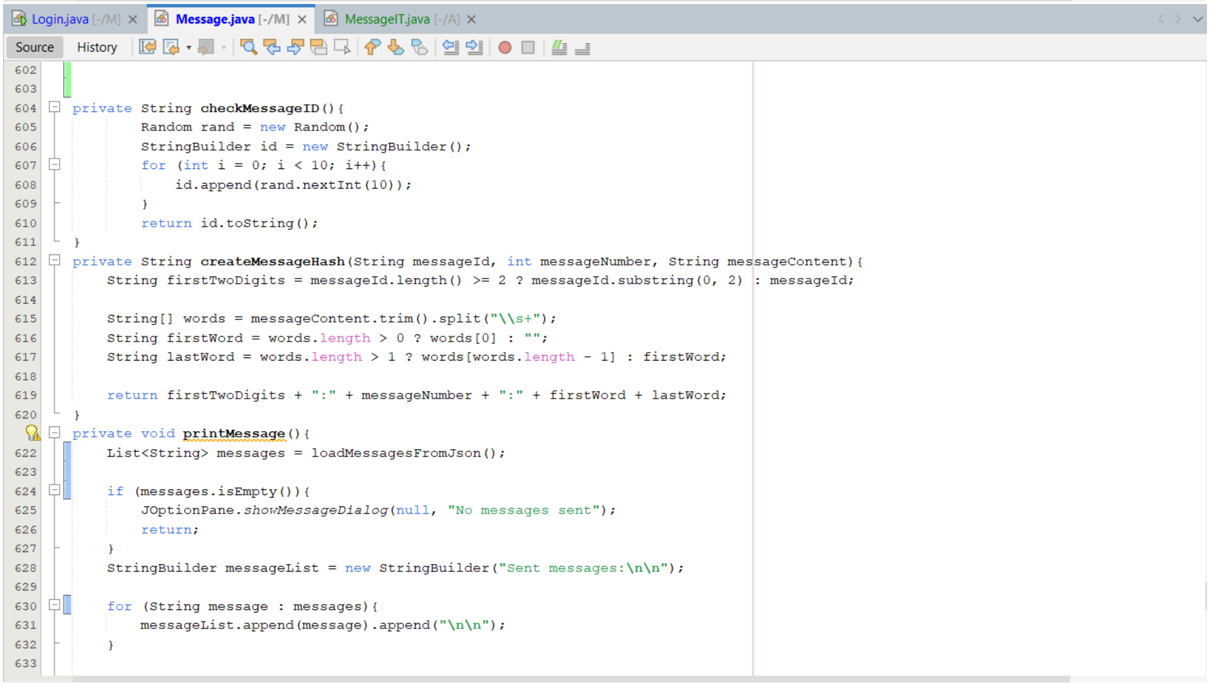
```

    public void listSentMessages(){
        List<String> messages = loadMessagesFromJson();

        if(messages.isEmpty()){
            JOptionPane.showMessageDialog(null, "No messages sent!");
        }

        StringBuilder arrayDisplay = new StringBuilder("Sent messages: \n\n");

        for (int i = 0; i < messages.size(); i++){
            arrayDisplay.append("[").append(i).append("]").append(messages.get(i)).append("\n");
        }

        JTextArea textArea = new JTextArea(arrayDisplay.toString());
        textArea.setEditable(false);
        JScrollPane scrollPane = new JScrollPane(textArea);
        scrollPane.setPreferredSize(new Dimension(500, 300));


        JOptionPane.showMessageDialog(null, scrollPane, "Sent messages", JOptionPane.INFORMATION_MESSAGE);
    }

    public void listDisregardedMessages(){
        List<String> messages = loadMessagesFromJson();

        if(messages.isEmpty()){
            JOptionPane.showMessageDialog(null, "No disregarded messages!");
        }

        StringBuilder disregarded = new StringBuilder("Disregarded messages: \n\n");

```



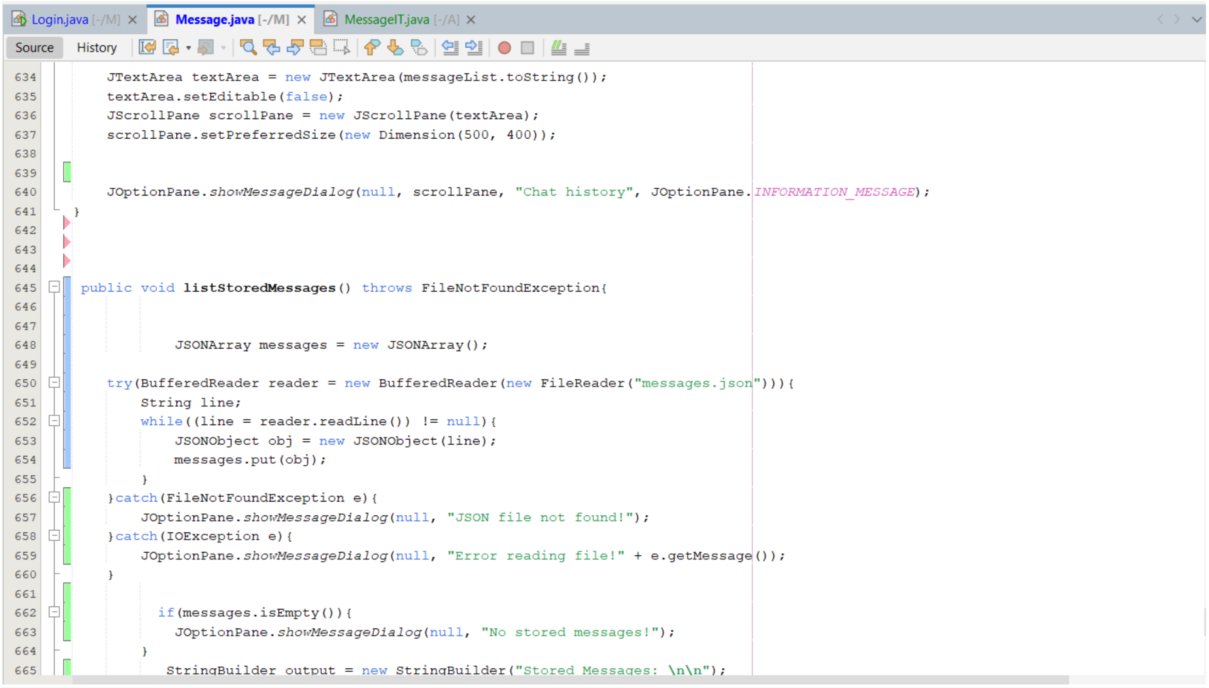
```
        for (int i = 0; i < disregardedMessages.size(); i++){
            disregarded.append("[").append(i).append("]").append(disregardedMessages.get(i)).append("\n");
        }

        JTextArea textArea = new JTextArea(disregarded.toString());
        textArea.setEditable(false);
        JScrollPane scrollPane = new JScrollPane(textArea);
        scrollPane.setPreferredSize(new Dimension(500, 300));


        JOptionPane.showMessageDialog(null, scrollPane, "Disregarded messages.", JOptionPane.INFORMATION_MESSAGE);
    }



    public void listMessageHashes(){
        List<String> messages = loadMessagesFromJson();

        if(messages.isEmpty()){
            JOptionPane.showMessageDialog(null, "No messages!");
        }

        StringBuilder output = new StringBuilder("Message hashes: \n\n");

        for(int i = 0; i < messages.size(); i++){
            String message = messages.get(i);
            int hash = message.hashCode();
            output.append("[").append(i).append("]").append(hash).append("\n");
        }

        JTextArea textArea = new JTextArea(output.toString());
        textArea.setEditable(false);
```



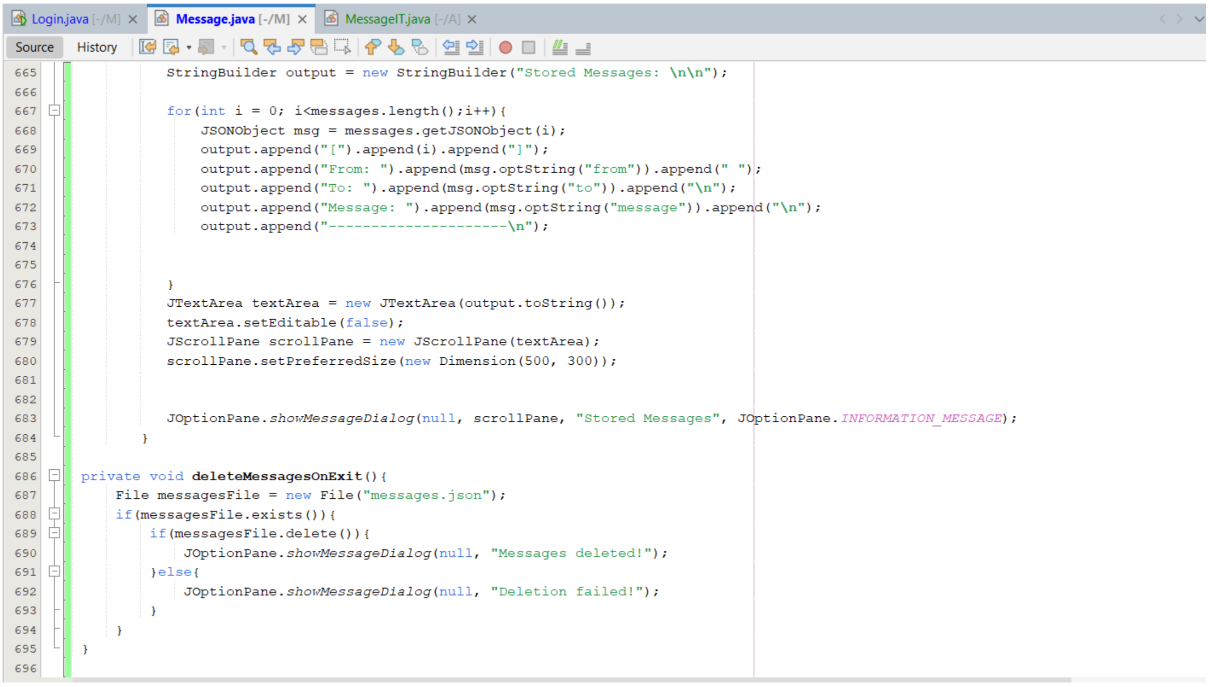
```java
        JScrollPane scrollPane = new JScrollPane(textArea);
        scrollPane.setPreferredSize(new Dimension(500, 300));


        JOptionPane.showMessageDialog(null, scrollPane, "Message Hashes", JOptionPane.INFORMATION_MESSAGE);
    }

    public void listMessageIDs(){
        List<String> messages = loadMessagesFromJson();

        if(messages.isEmpty()){
            JOptionPane.showMessageDialog(null, "No messages!");
        }

        StringBuilder output = new StringBuilder("Message ID's: \n\n");

        for(int i = 0; i < messages.size(); i++){
            output.append("Message ID's: ").append(i).append("\n");
        }
        JTextArea textArea = new JTextArea(output.toString());
        textArea.setEditable(false);
        JScrollPane scrollPane = new JScrollPane(textArea);
        scrollPane.setPreferredSize(new Dimension(500, 300));


        JOptionPane.showMessageDialog(null,scrollPane, "Message ID's", JOptionPane.INFORMATION_MESSAGE);
    }



    @SuppressWarnings("empty-statement")

```



```java


private String checkMessageID(){
        Random rand = new Random();
        StringBuilder id = new StringBuilder();
        for (int i = 0; i < 10; i++){
            id.append(rand.nextInt(10));
        }
        return id.toString();
}
private String createMessageHash(String messageId, int messageNumber, String messageContent){
    String firstTwoDigits = messageId.length() >= 2 ? messageId.substring(0, 2) : messageId;

    String[] words = messageContent.trim().split("\\s+");
    String firstWord = words.length > 0 ? words[0] : "";
    String lastWord = words.length > 1 ? words[words.length - 1] : firstWord;

    return firstTwoDigits + ":" + messageNumber + ":" + firstWord + lastWord;
}
private void printMessage(){
    List<String> messages = loadMessagesFromJson();

    if (messages.isEmpty()){
        JOptionPane.showMessageDialog(null, "No messages sent");
        return;
    }
    StringBuilder messageList = new StringBuilder("Sent messages:\n\n");

    for (String message : messages){
        messageList.append(message).append("\n\n");
    }

```



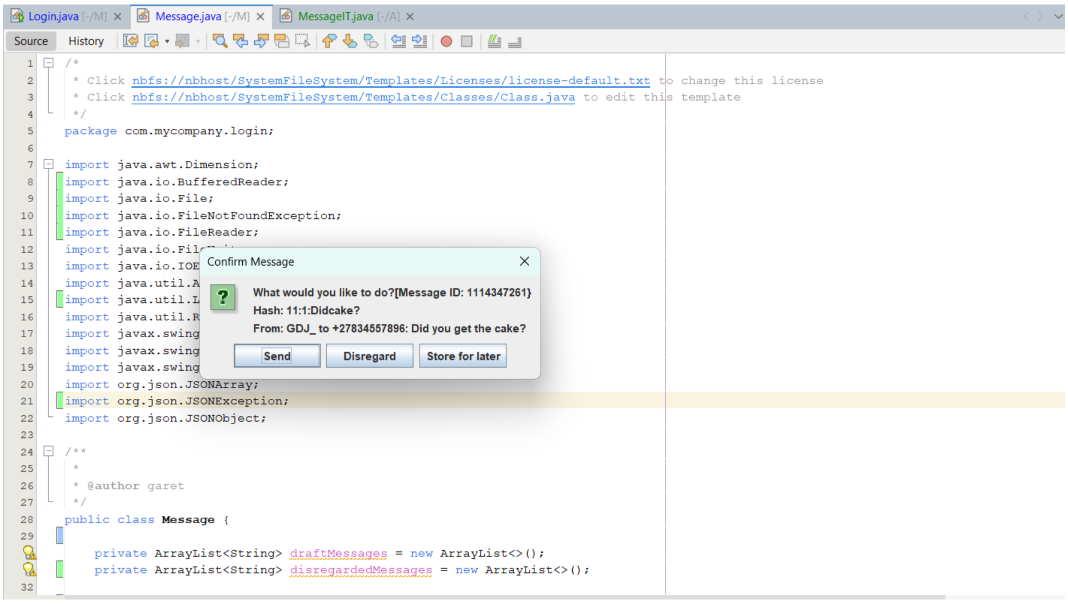
```
   JTextArea textArea = new JTextArea(messageList.toString());
   textArea.setEditable(false);
   JScrollPane scrollPane = new JScrollPane(textArea);
   scrollPane.setPreferredSize(new Dimension(500, 400));


   JOptionPane.showMessageDialog(null, scrollPane, "Chat history", JOptionPane.INFORMATION_MESSAGE);
}



public void listStoredMessages() throws FileNotFoundException{


           JSONArray messages = new JSONArray();

   try(BufferedReader reader = new BufferedReader(new FileReader("messages.json"))){
       String line;
       while((line = reader.readLine()) != null){
           JSONObject obj = new JSONObject(line);
           messages.put(obj);
       }
   }catch(FileNotFoundException e){
       JOptionPane.showMessageDialog(null, "JSON file not found!");
   }catch(IOException e){
       JOptionPane.showMessageDialog(null, "Error reading file!" + e.getMessage());
   }

         if(messages.isEmpty()){
           JOptionPane.showMessageDialog(null, "No stored messages!");
       }
          StringBuilder output = new StringBuilder("Stored Messages: \n\n");
```



```
          StringBuilder output = new StringBuilder("Stored Messages: \n\n");

          for(int i = 0; i<messages.length();i++){
              JSONObject msg = messages.getJSONObject(i);
              output.append("[").append(i).append("]");
              output.append("From: ").append(msg.optString("from")).append(" ");
              output.append("To: ").append(msg.optString("to")).append("\n");
              output.append("Message: ").append(msg.optString("message")).append("\n");
              output.append("---------------------\n");


          }
          JTextArea textArea = new JTextArea(output.toString());
          textArea.setEditable(false);
          JScrollPane scrollPane = new JScrollPane(textArea);
          scrollPane.setPreferredSize(new Dimension(500, 300));


          JOptionPane.showMessageDialog(null, scrollPane, "Stored Messages", JOptionPane.INFORMATION_MESSAGE);
       }

private void deleteMessagesOnExit(){
    File messagesFile = new File("messages.json");
    if(messagesFile.exists()){
        if(messagesFile.delete()){
            JOptionPane.showMessageDialog(null, "Messages deleted!");
        }else{
            JOptionPane.showMessageDialog(null, "Deletion failed!");
        }
    }
}
```

Screenshots of testing:

Test data 1_ confirmation of the message:



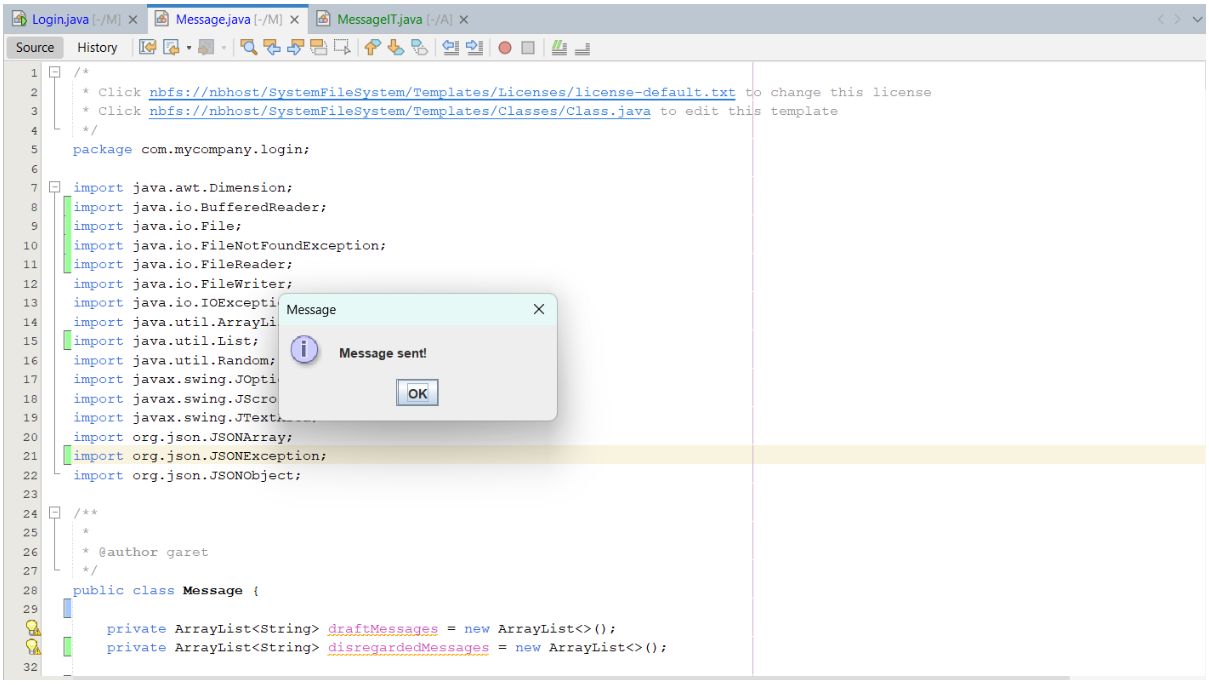
```
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template
 */
package com.mycompany.login;

import java.awt.Dimension;
import java.io.BufferedReader;
import java.io.File;
import java.io.FileNotFoundException;
import java.io.FileReader;
import java.io.Fil
import java.io.IOE
import java.util.A
import java.util.L
import java.util.R
import javax.swing
import javax.swing
import javax.swing
import org.json.JSONArray;
import org.json.JSONException;
import org.json.JSONObject;

/**
 *
 * @author garet
 */
public class Message {

    private ArrayList<String> draftMessages = new ArrayList<>();
    private ArrayList<String> disregardedMessages = new ArrayList<>();
```

Confirm Message

What would you like to do?[Message ID: 1114347261}
Hash: 11:1:Didcake?
From: GDJ_ to +27834557896: Did you get the cake?

Send | Disregard | Store for later

Test data 1_ message sent:



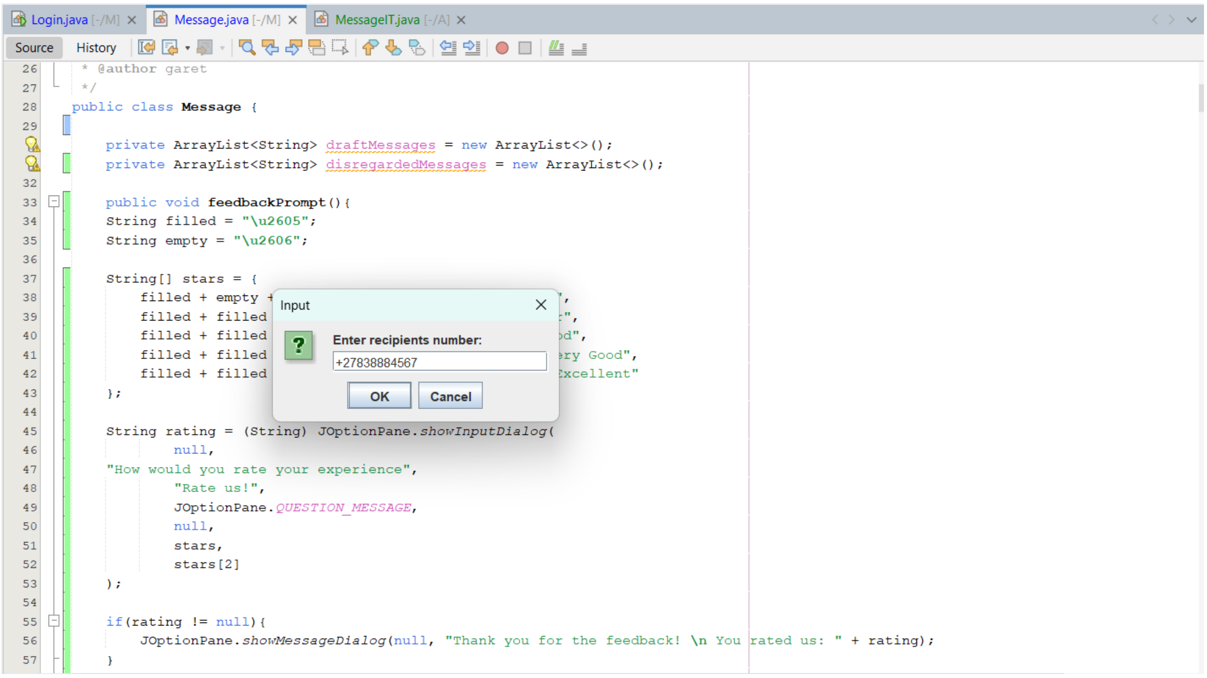
```
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template
 */
package com.mycompany.login;

import java.awt.Dimension;
import java.io.BufferedReader;
import java.io.File;
import java.io.FileNotFoundException;
import java.io.FileReader;
import java.io.FileWriter;
import java.io.IOExcepti
import java.util.ArrayLi
import java.util.List;
import java.util.Random;
import javax.swing.JOpti
import javax.swing.JScro
import javax.swing.JText
import org.json.JSONArray;
import org.json.JSONException;
import org.json.JSONObject;

/**
 *
 * @author garet
 */
public class Message {

    private ArrayList<String> draftMessages = new ArrayList<>();
    private ArrayList<String> disregardedMessages = new ArrayList<>();
```

Message

Message sent!

OK

Test data 2_3_4_5_ recipient number:



```
 * @author garet
 */
public class Message {

    private ArrayList<String> draftMessages = new ArrayList<>();
    private ArrayList<String> disregardedMessages = new ArrayList<>();

    public void feedbackPrompt(){
    String filled = "\u2605";
    String empty = "\u2606";

    String[] stars = {
        filled + empty +                                ',
        filled + filled                                 :",
        filled + filled                                 od",
        filled + filled                                 ery Good",
        filled + filled                                 Excellent"
    };

    String rating = (String) JOptionPane.showInputDialog(
            null,
    "How would you rate your experience",
            "Rate us!",
            JOptionPane.QUESTION_MESSAGE,
            null,
            stars,
            stars[2]
    );

    if(rating != null){
        JOptionPane.showMessageDialog(null, "Thank you for the feedback! \n You rated us: " + rating);
    }
```

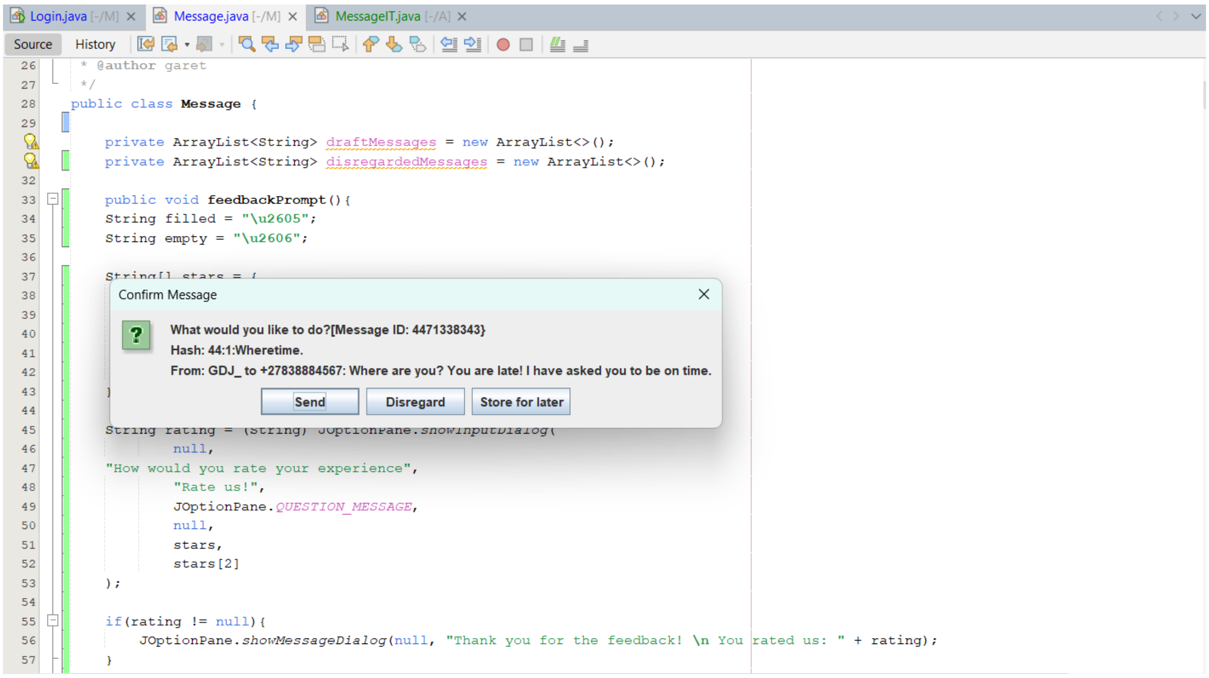
Input

Enter recipients number:

+27838884567

OK   Cancel

Test data 2_ message:



```
 * @author garet
 */
public class Message {

    private ArrayList<String> draftMessages = new ArrayList<>();
    private ArrayList<String> disregardedMessages = new ArrayList<>();

    public void feedbackPrompt(){
    String filled = "\u2605";
    String empty = "\u2606";

    String[] stars = {

    String rating = (String) JOptionPane.showInputDialog(
            null,
    "How would you rate your experience",
            "Rate us!",
            JOptionPane.QUESTION_MESSAGE,
            null,
            stars,
            stars[2]
    );

    if(rating != null){
        JOptionPane.showMessageDialog(null, "Thank you for the feedback! \n You rated us: " + rating);
    }
```

Confirm Message

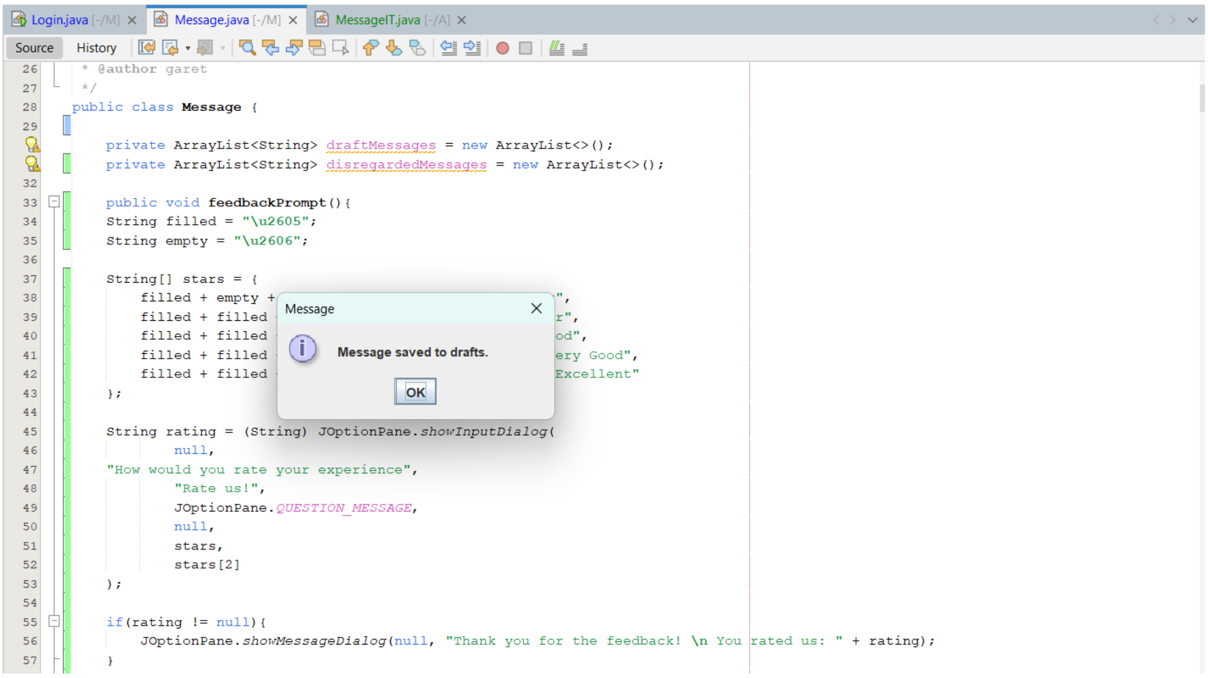
What would you like to do?[Message ID: 4471338343}

Hash: 44:1:Wheretime.

From: GDJ_ to +27838884567: Where are you? You are late! I have asked you to be on time.

Send   Disregard   Store for later

Test data 2_ saved message:

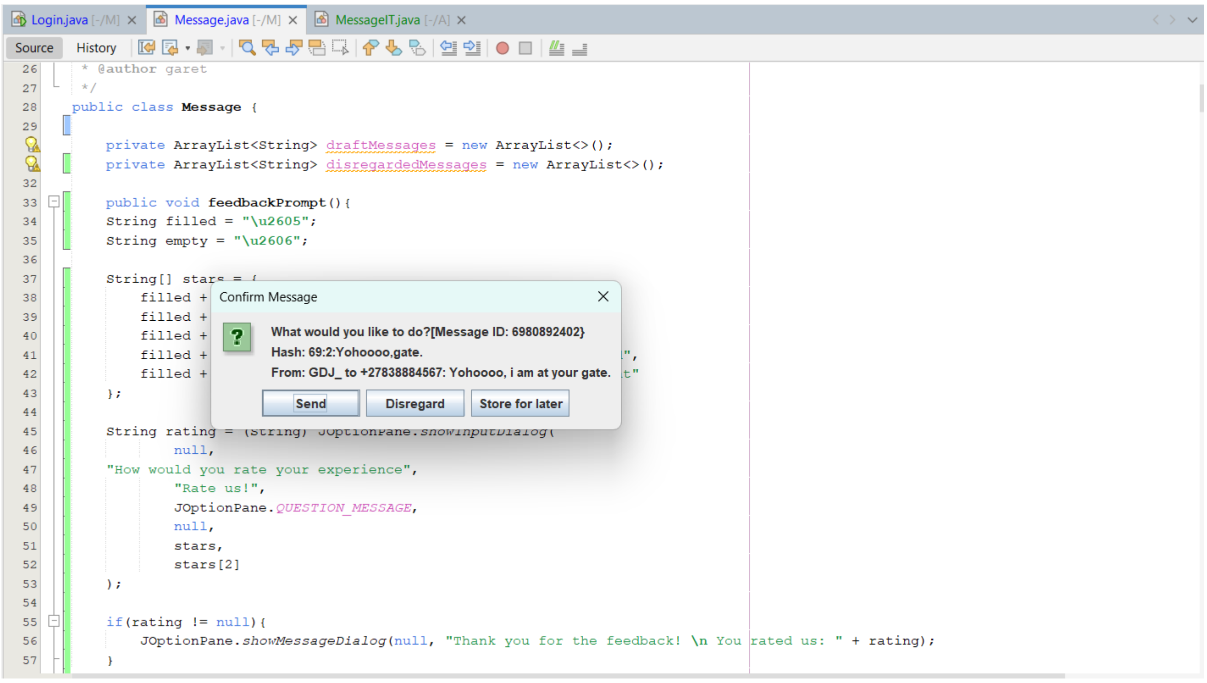


```
 * @author garet
 */
public class Message {

    private ArrayList<String> draftMessages = new ArrayList<>();
    private ArrayList<String> disregardedMessages = new ArrayList<>();

    public void feedbackPrompt(){
    String filled = "\u2605";
    String empty = "\u2606";

    String[] stars = {
        filled + empty +                                  ",
        filled + filled                                   r",
        filled + filled                                   od",
        filled + filled                                   ery Good",
        filled + filled                                   Excellent"
    };

    String rating = (String) JOptionPane.showInputDialog(
            null,
    "How would you rate your experience",
            "Rate us!",
            JOptionPane.QUESTION_MESSAGE,
            null,
            stars,
            stars[2]
    );

    if(rating != null){
        JOptionPane.showMessageDialog(null, "Thank you for the feedback! \n You rated us: " + rating);
    }
```

Message

Message saved to drafts.

Test data 3_ message:

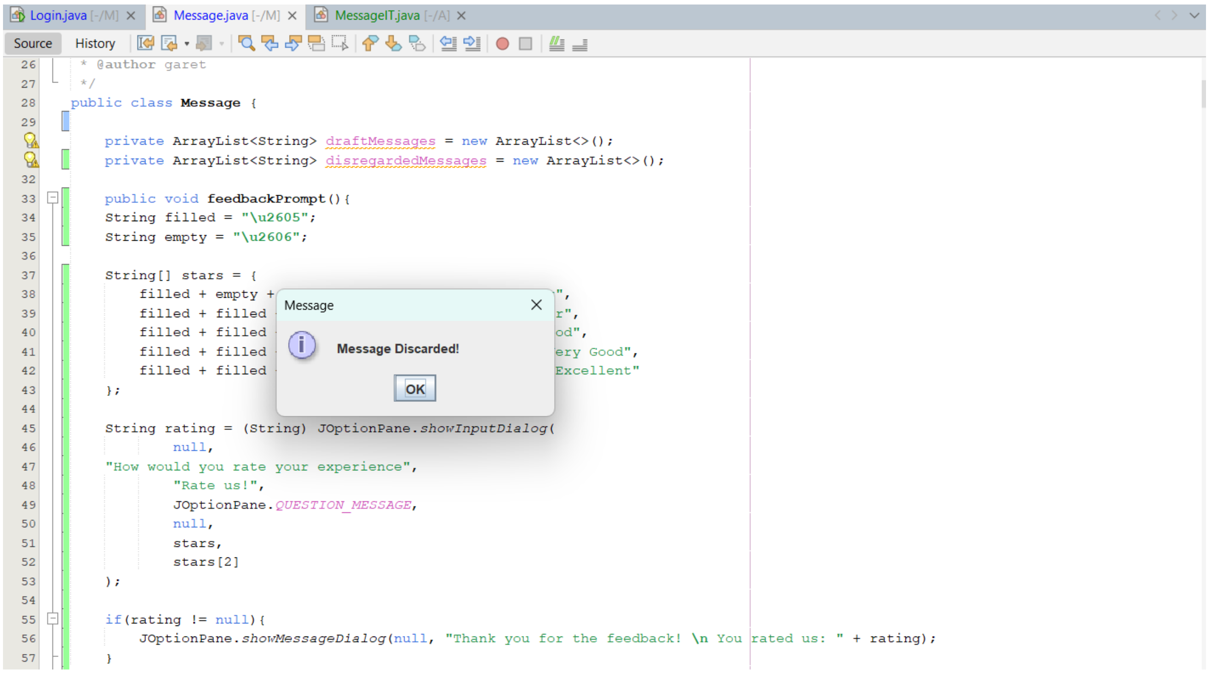


```
 * @author garet
 */
public class Message {

    private ArrayList<String> draftMessages = new ArrayList<>();
    private ArrayList<String> disregardedMessages = new ArrayList<>();

    public void feedbackPrompt(){
    String filled = "\u2605";
    String empty = "\u2606";

    String[] stars = {
        filled +                                                   ",
        filled +                                                   t"
        filled +
        filled +
        filled +
    };

    String rating

            null,
    "How would you rate your experience",
            "Rate us!",
            JOptionPane.QUESTION_MESSAGE,
            null,
            stars,
            stars[2]
    );

    if(rating != null){
        JOptionPane.showMessageDialog(null, "Thank you for the feedback! \n You rated us: " + rating);
    }
```

Confirm Message

What would you like to do?[Message ID: 6980892402}
Hash: 69:2:Yohoooo,gate.
From: GDJ_ to +27838884567: Yohoooo, i am at your gate.

Test data 3_ message discarded:

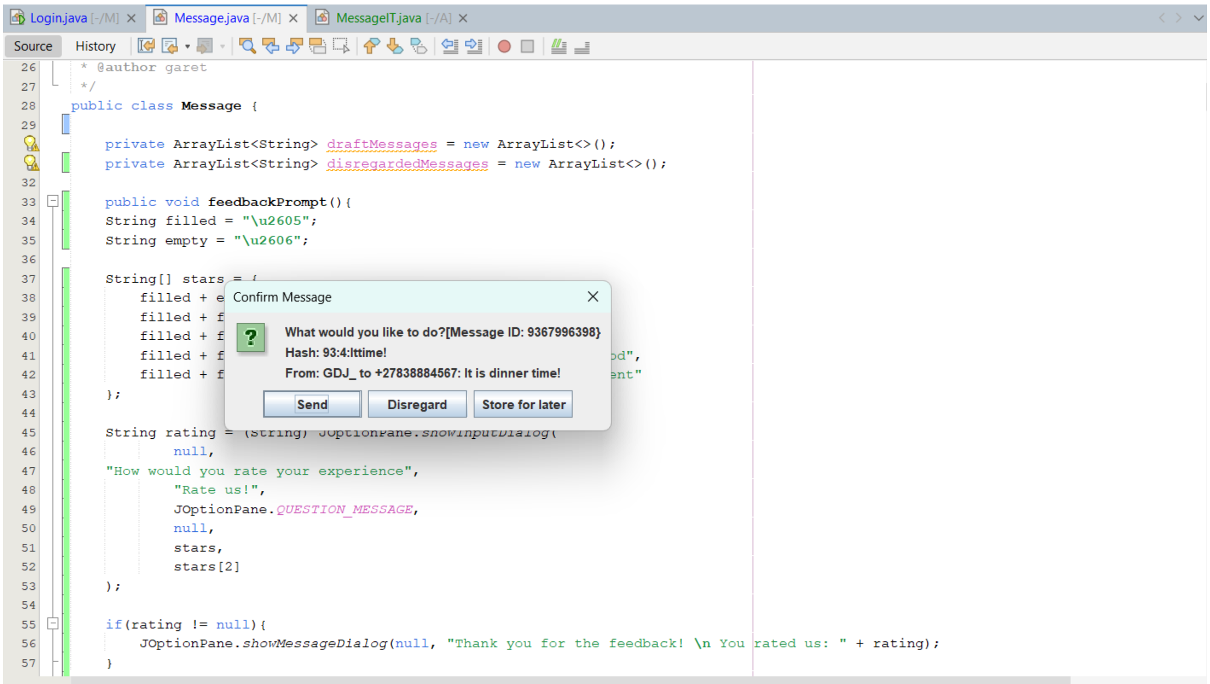


```
 * @author garet
 */
public class Message {

    private ArrayList<String> draftMessages = new ArrayList<>();
    private ArrayList<String> disregardedMessages = new ArrayList<>();

    public void feedbackPrompt(){
    String filled = "\u2605";
    String empty = "\u2606";

    String[] stars = {
        filled + empty +                      ",
        filled + filled                       r",
        filled + filled                       od",
        filled + filled                       ery Good",
        filled + filled                       Excellent"
    };

    String rating = (String) JOptionPane.showInputDialog(
            null,
    "How would you rate your experience",
            "Rate us!",
            JOptionPane.QUESTION_MESSAGE,
            null,
            stars,
            stars[2]
    );

    if(rating != null){
        JOptionPane.showMessageDialog(null, "Thank you for the feedback! \n You rated us: " + rating);
    }
```

Message

Message Discarded!

Test data 4_ message:

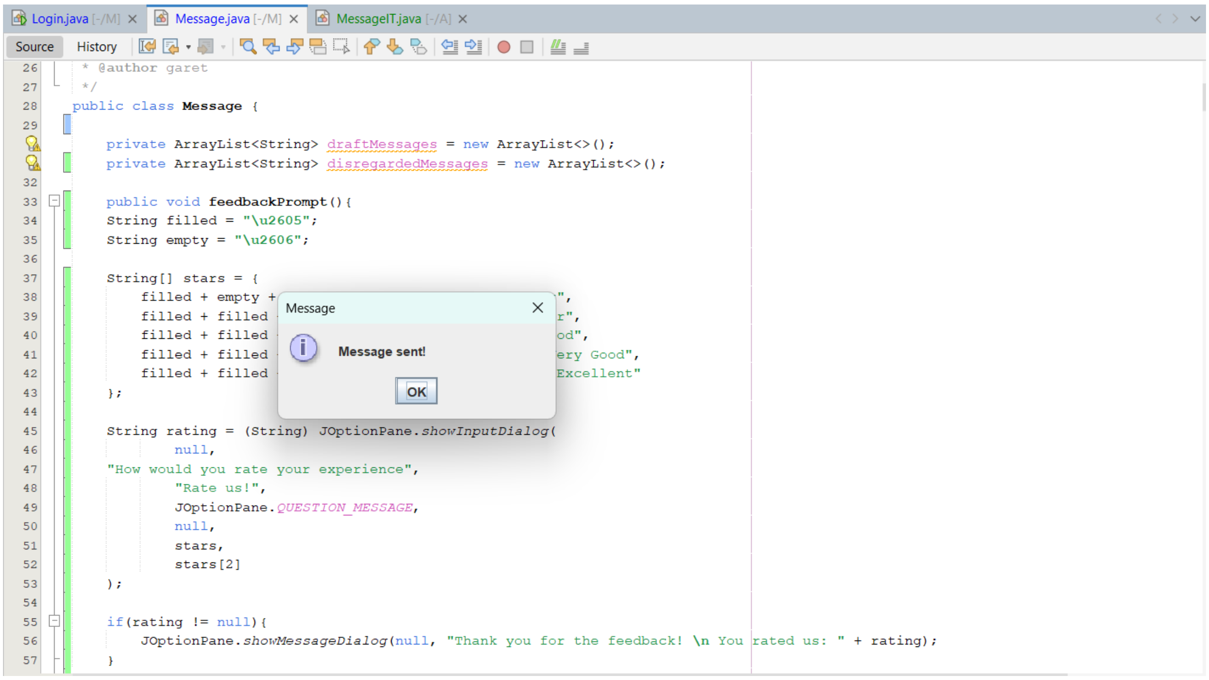


```
 * @author garet
 */
public class Message {

    private ArrayList<String> draftMessages = new ArrayList<>();
    private ArrayList<String> disregardedMessages = new ArrayList<>();

    public void feedbackPrompt(){
    String filled = "\u2605";
    String empty = "\u2606";

    String[] stars = {
        filled + e                                          
        filled + f
        filled + f
        filled + f                                          od",
        filled + f                                          ent"
    };

    String rating = (String) JOptionPane.showInputDialog(
            null,
    "How would you rate your experience",
            "Rate us!",
            JOptionPane.QUESTION_MESSAGE,
            null,
            stars,
            stars[2]
    );

    if(rating != null){
        JOptionPane.showMessageDialog(null, "Thank you for the feedback! \n You rated us: " + rating);
    }
```

Confirm Message

What would you like to do?[Message ID: 9367996398}
Hash: 93:4:Ittime!
From: GDJ_ to +27838884567: It is dinner time!

Test data 4_ message sent:

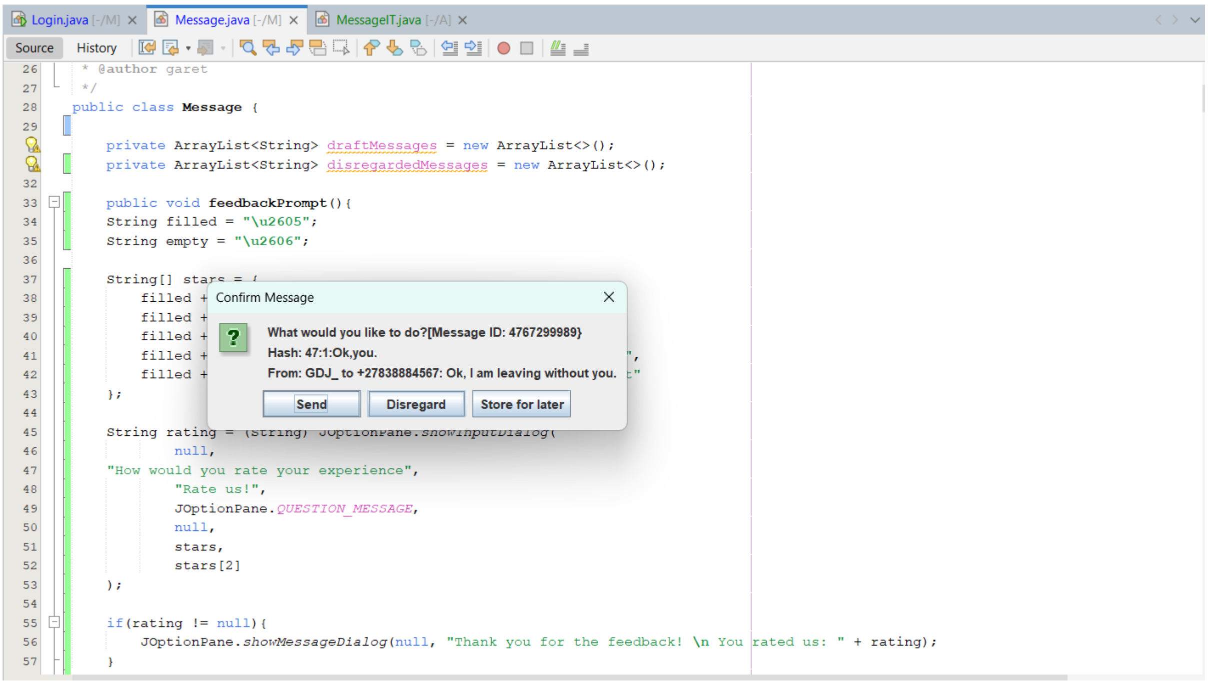


```
 * @author garet
 */
public class Message {

    private ArrayList<String> draftMessages = new ArrayList<>();
    private ArrayList<String> disregardedMessages = new ArrayList<>();

    public void feedbackPrompt(){
    String filled = "\u2605";
    String empty = "\u2606";

    String[] stars = {
        filled + empty +                                  ",
        filled + filled                                   r",
        filled + filled                                   od",
        filled + filled                                   ery Good",
        filled + filled                                   Excellent"
    };

    String rating = (String) JOptionPane.showInputDialog(
            null,
    "How would you rate your experience",
            "Rate us!",
            JOptionPane.QUESTION_MESSAGE,
            null,
            stars,
            stars[2]
    );

    if(rating != null){
        JOptionPane.showMessageDialog(null, "Thank you for the feedback! \n You rated us: " + rating);
    }
```

Message

Message sent!

Test data 5_ message:

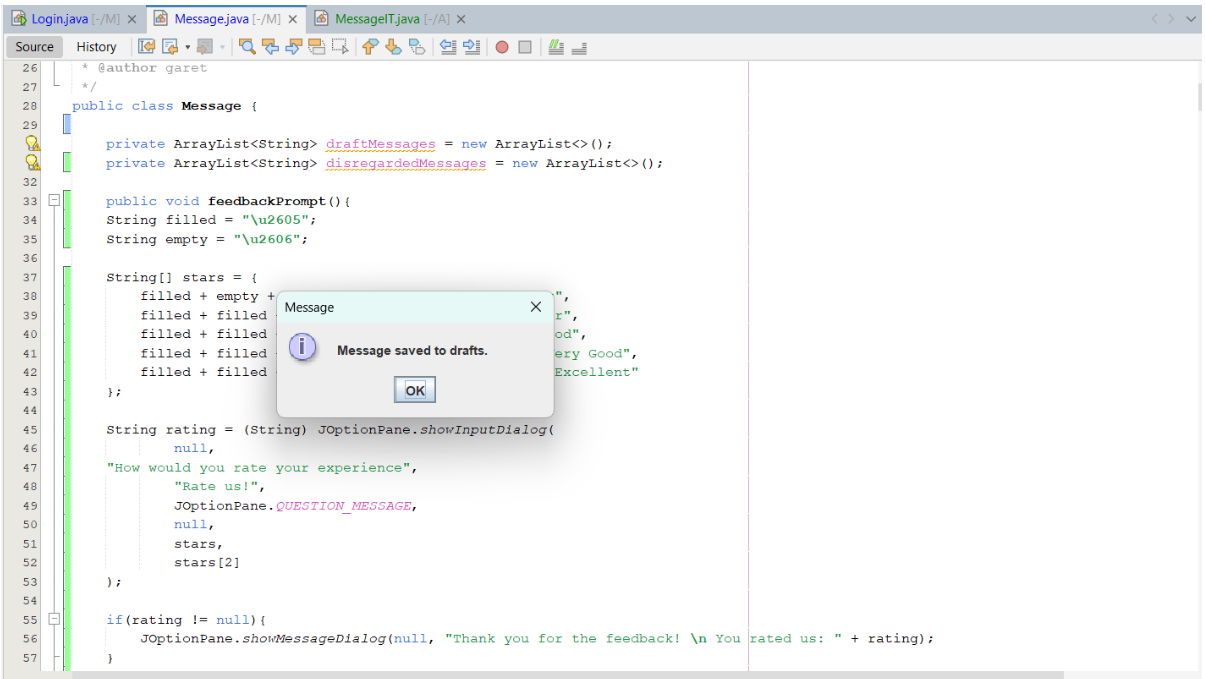


```
 * @author garet
 */
public class Message {

    private ArrayList<String> draftMessages = new ArrayList<>();
    private ArrayList<String> disregardedMessages = new ArrayList<>();

    public void feedbackPrompt(){
    String filled = "\u2605";
    String empty = "\u2606";

    String[] stars = {
        filled +
        filled +
        filled +
        filled +                                                    ",
        filled +                                                    t"
    };

    String rating = (String) JOptionPane.showInputDialog(
            null,
    "How would you rate your experience",
            "Rate us!",
            JOptionPane.QUESTION_MESSAGE,
            null,
            stars,
            stars[2]
    );

    if(rating != null){
        JOptionPane.showMessageDialog(null, "Thank you for the feedback! \n You rated us: " + rating);
    }
```

Confirm Message

What would you like to do?[Message ID: 4767299989}
Hash: 47:1:Ok,you.
From: GDJ_ to +27838884567: Ok, I am leaving without you.

Test data 5_ saved message:



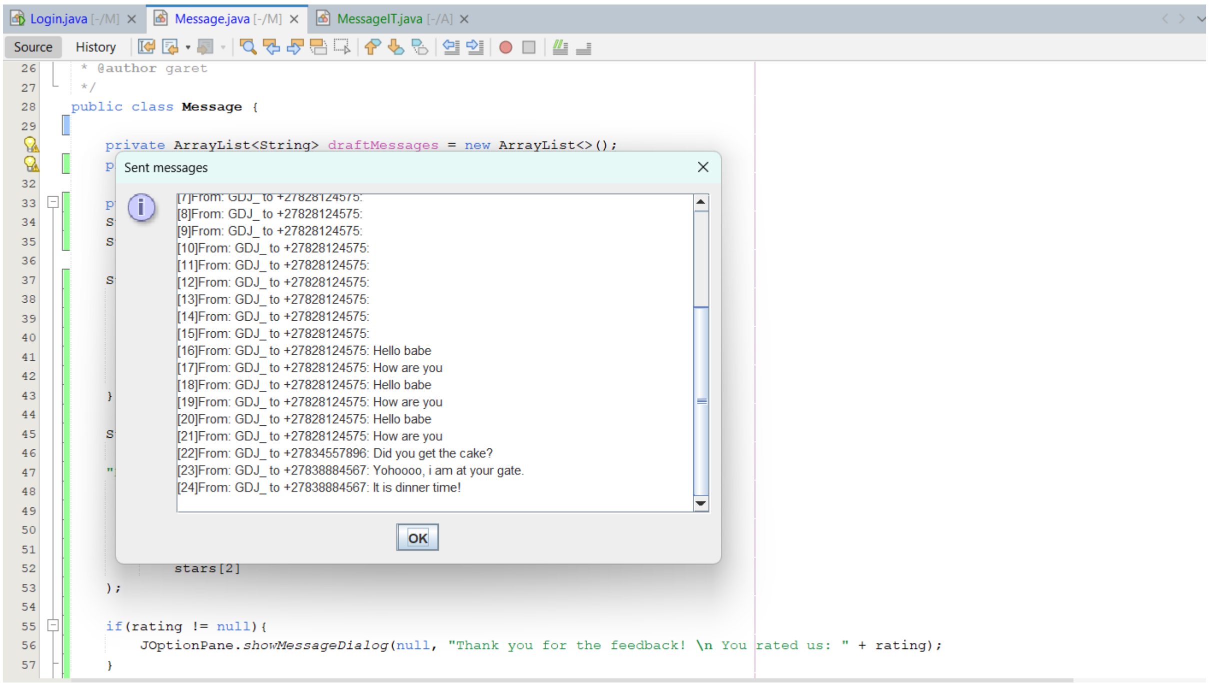
```
 * @author garet
 */
public class Message {

    private ArrayList<String> draftMessages = new ArrayList<>();
    private ArrayList<String> disregardedMessages = new ArrayList<>();

    public void feedbackPrompt(){
    String filled = "\u2605";
    String empty = "\u2606";

    String[] stars = {
        filled + empty +                                  ",
        filled + filled                                   r",
        filled + filled                                   od",
        filled + filled                                   ery Good",
        filled + filled                                   Excellent"
    };

    String rating = (String) JOptionPane.showInputDialog(
            null,
    "How would you rate your experience",
            "Rate us!",
            JOptionPane.QUESTION_MESSAGE,
            null,
            stars,
            stars[2]
    );

    if(rating != null){
        JOptionPane.showMessageDialog(null, "Thank you for the feedback! \n You rated us: " + rating);
    }
```

Message

Message saved to drafts.

OK

sent message page:



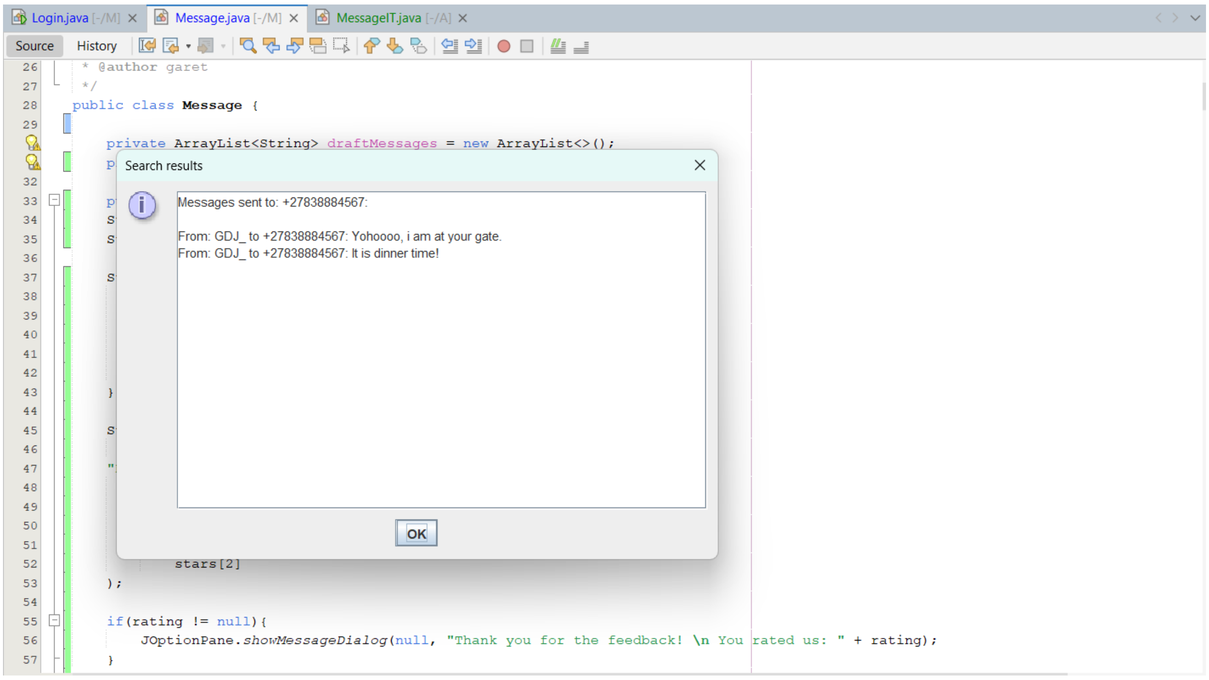
Sent messages

```
[7]From: GDJ_ to +27828124575:
[8]From: GDJ_ to +27828124575:
[9]From: GDJ_ to +27828124575:
[10]From: GDJ_ to +27828124575:
[11]From: GDJ_ to +27828124575:
[12]From: GDJ_ to +27828124575:
[13]From: GDJ_ to +27828124575:
[14]From: GDJ_ to +27828124575:
[15]From: GDJ_ to +27828124575:
[16]From: GDJ_ to +27828124575: Hello babe
[17]From: GDJ_ to +27828124575: How are you
[18]From: GDJ_ to +27828124575: Hello babe
[19]From: GDJ_ to +27828124575: How are you
[20]From: GDJ_ to +27828124575: Hello babe
[21]From: GDJ_ to +27828124575: How are you
[22]From: GDJ_ to +27834557896: Did you get the cake?
[23]From: GDJ_ to +27838884567: Yohoooo, i am at your gate.
[24]From: GDJ_ to +27838884567: It is dinner time!
```

OK

```
 * @author garet
 */
public class Message {

    private ArrayList<String> draftMessages = new ArrayList<>();
            stars[2]
    );

    if(rating != null){
        JOptionPane.showMessageDialog(null, "Thank you for the feedback! \n You rated us: " + rating);
    }
```

recipient search page:



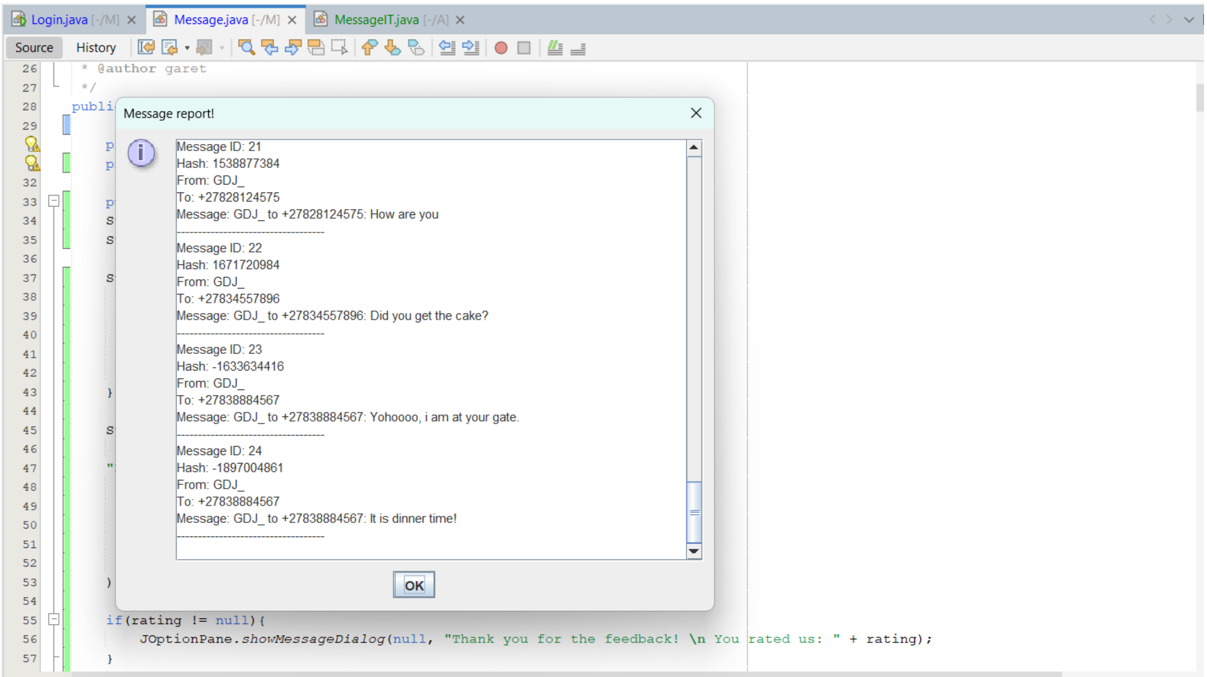
```
 * @author garet
 */
public class Message {

    private ArrayList<String> draftMessages = new ArrayList<>();
```

Search results

```
Messages sent to: +27838884567:

From: GDJ_ to +27838884567: Yohoooo, i am at your gate.
From: GDJ_ to +27838884567: It is dinner time!
```

```
            stars[2]
    );

    if(rating != null){
        JOptionPane.showMessageDialog(null, "Thank you for the feedback! \n You rated us: " + rating);
    }
```

message report page:



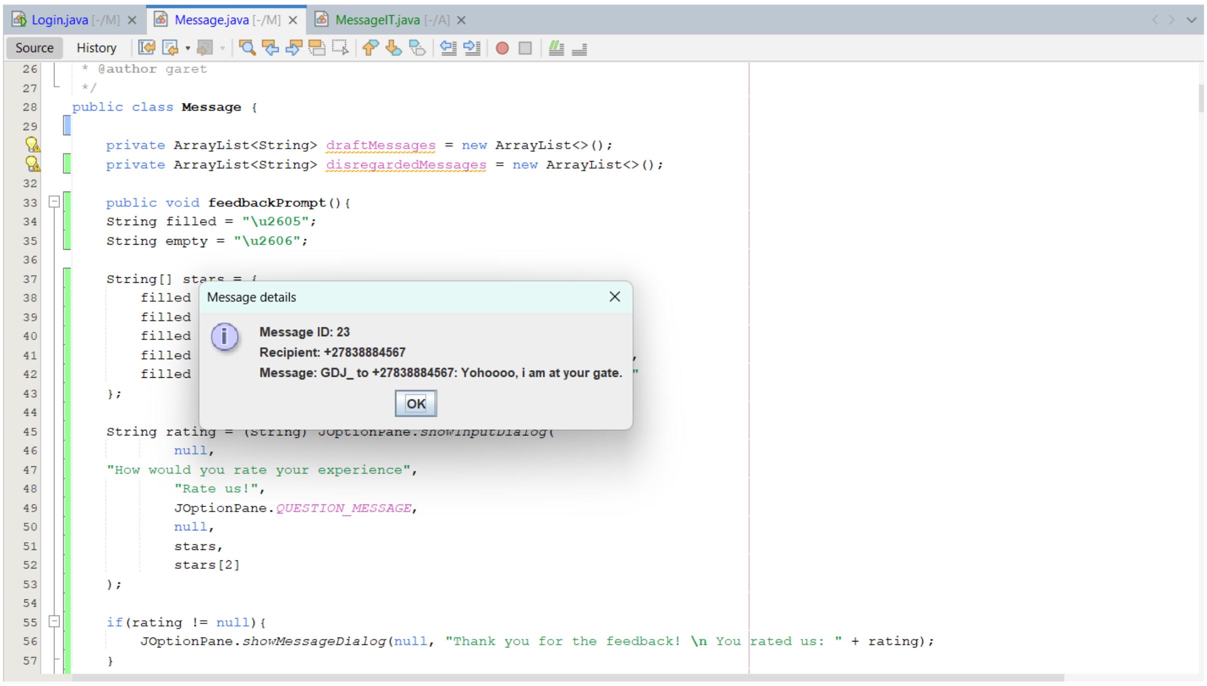
```
 * @author garet
 */
```

Message report!

```
Message ID: 21
Hash: 1538877384
From: GDJ_
To: +27828124575
Message: GDJ_ to +27828124575: How are you
----------------------------------
Message ID: 22
Hash: 1671720984
From: GDJ_
To: +27834557896
Message: GDJ_ to +27834557896: Did you get the cake?
----------------------------------
Message ID: 23
Hash: -1633634416
From: GDJ_
To: +27838884567
Message: GDJ_ to +27838884567: Yohoooo, i am at your gate.
----------------------------------
Message ID: 24
Hash: -1897004861
From: GDJ_
To: +27838884567
Message: GDJ_ to +27838884567: It is dinner time!
----------------------------------
```

```
    if(rating != null){
        JOptionPane.showMessageDialog(null, "Thank you for the feedback! \n You rated us: " + rating);
    }
```

message ID page:



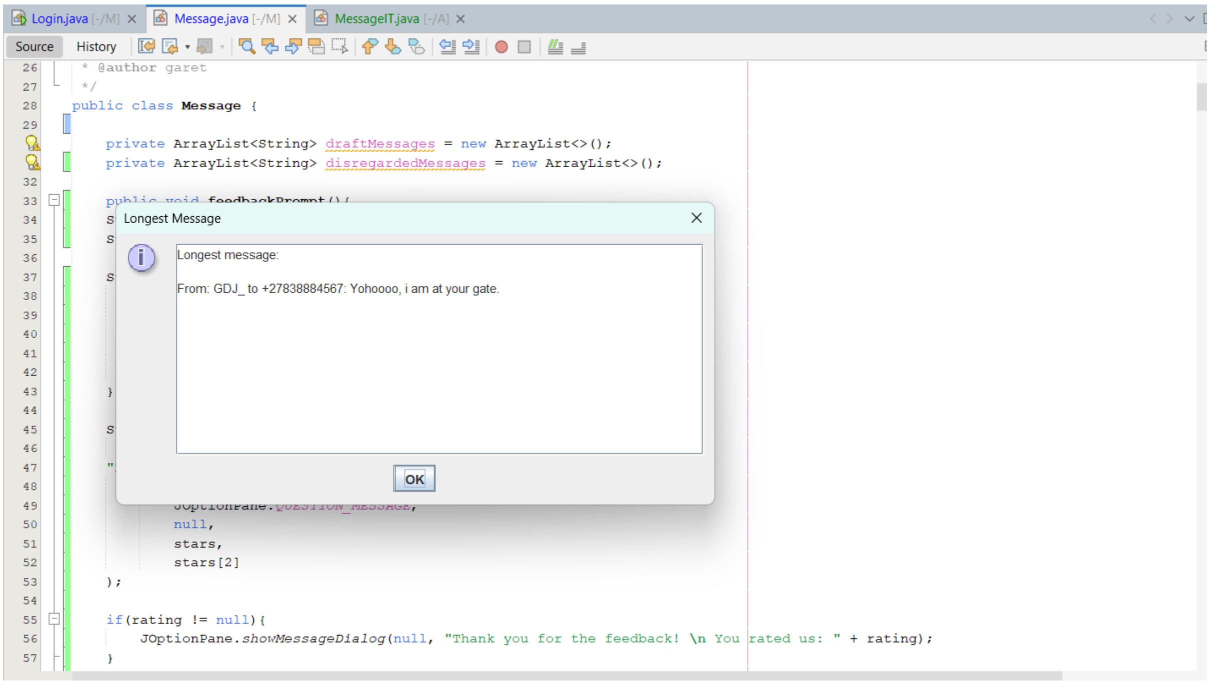
```
 * @author garet
 */
public class Message {

    private ArrayList<String> draftMessages = new ArrayList<>();
    private ArrayList<String> disregardedMessages = new ArrayList<>();

    public void feedbackPrompt(){
    String filled = "\u2605";
    String empty = "\u2606";

    String[] stars = {
        filled
        filled
        filled
        filled                                              ,
        filled                                              "
    };

    String rating = (String) JOptionPane.showInputDialog(
            null,
    "How would you rate your experience",
            "Rate us!",
            JOptionPane.QUESTION_MESSAGE,
            null,
            stars,
            stars[2]
    );

    if(rating != null){
        JOptionPane.showMessageDialog(null, "Thank you for the feedback! \n You rated us: " + rating);
    }
```

Message details

Message ID: 23
Recipient: +27838884567
Message: GDJ_ to +27838884567: Yohoooo, i am at your gate.

OK

longest message page:



```
 * @author garet
 */
public class Message {

    private ArrayList<String> draftMessages = new ArrayList<>();
    private ArrayList<String> disregardedMessages = new ArrayList<>();

    public void feedbackPrompt(){
    S
    S

    S






    }

    S

    "

            JOptionPane.QUESTION_MESSAGE,
            null,
            stars,
            stars[2]
    );

    if(rating != null){
        JOptionPane.showMessageDialog(null, "Thank you for the feedback! \n You rated us: " + rating);
    }
```

Longest Message

Longest message:

From: GDJ_ to +27838884567: Yohoooo, i am at your gate.

OK