



HACETTEPE UNIVERSITY
COMPUTER ENGINEERING DEPARTMENT

BBM465 INFORMATION SECURITY LAB. - 2022 FALL

Assignment 2 - Message Box

November 23, 2022

Student names:

Abdullah Mert DİNÇER
Yiğit Emir IŞIKÇI

Student Numbers:

2200356016
2200356028

1 Problem Definition

For this project, we were expected to design a message program that any user is able to register a message for a registered user. And any registered user is able to view messages if they know the credentials of that specific message.

2 Solution Implementation

First we've designed the pages to be used in our program. We were going to have 4 screens. One for "home screen", one for "message register screen", one for "access screen", and lastly one for "view message screen".

1st Screen: Home Screen

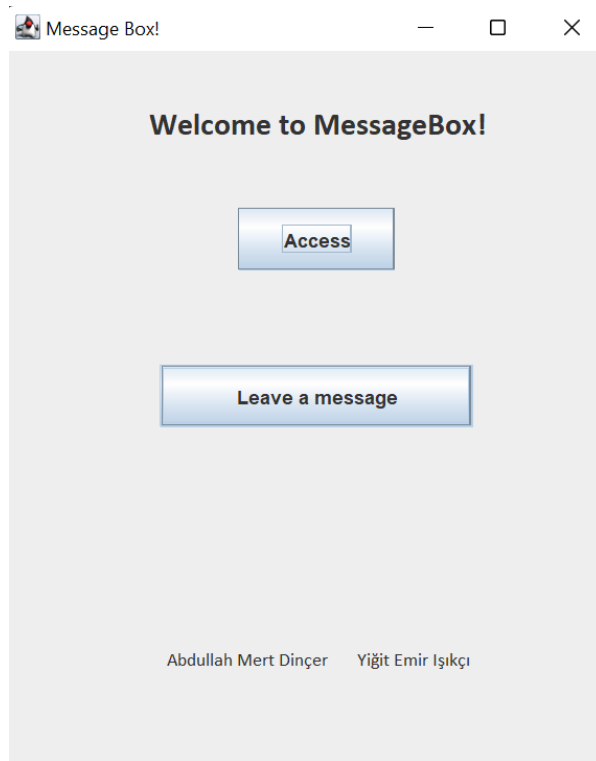
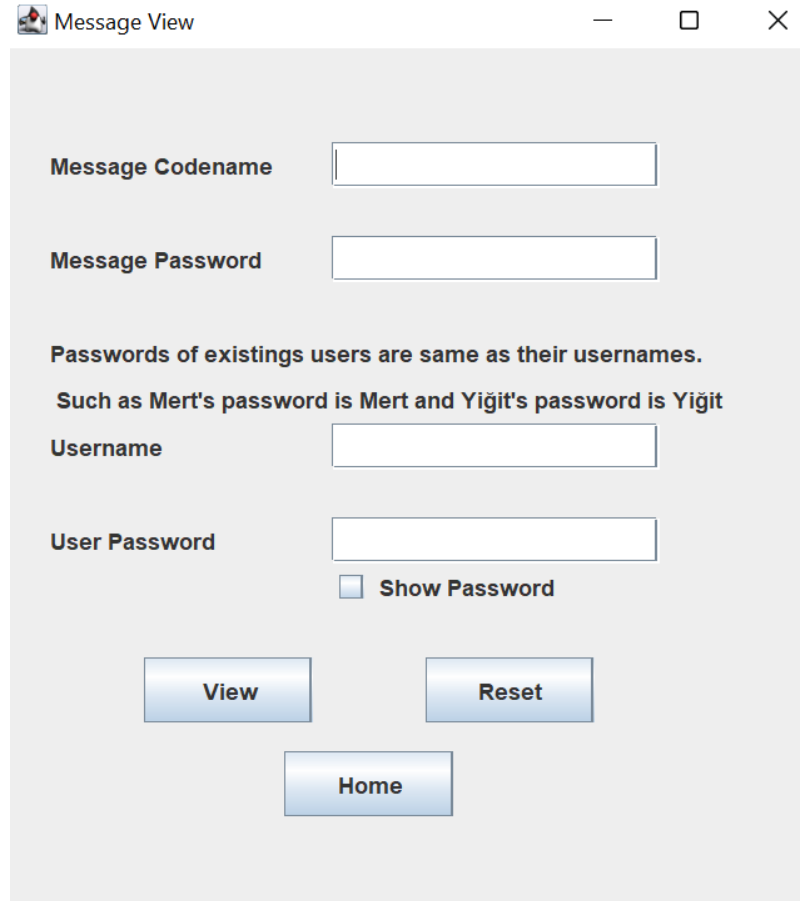


Figure 1: Home Screen

There are 2 buttons on home screen which navigates to other 2 screens. "Access" button navigates to access screen and "Leave a message" button navigates to message register screen.

2nd Screen: Access Screen



Message Codename

Message Password

Passwords of existings users are same as their usernames.
Such as Mert's password is Mert and Yiğit's password is Yiğit

Username

User Password

☐ **Show Password**

View **Reset**

Home

Figure 2: Access Screen

Access screen allows individuals to see messages which are written to them. For instance let's say I'm Mert and some other person told me he left me a message yesterday and gave me the credentials like the codename of the message and password of the message. Because I know the codename and the password of that message I can read my private message by entering my username and password data. Then the program checks if such codename exists in encrypted messages data file. If yes, next step is to check if the passwords match. If yes, next step is to check whose message is this? Like am I authorized to see the content of that message if yes and my password is legit then the content of the message is shown right after I press "View" button.

Important: Users passwords are the same as their usernames. Such as if username is Mert then his password is also Mert. If the username is Arda his password is Arda as well.

3rd Screen: View Screen

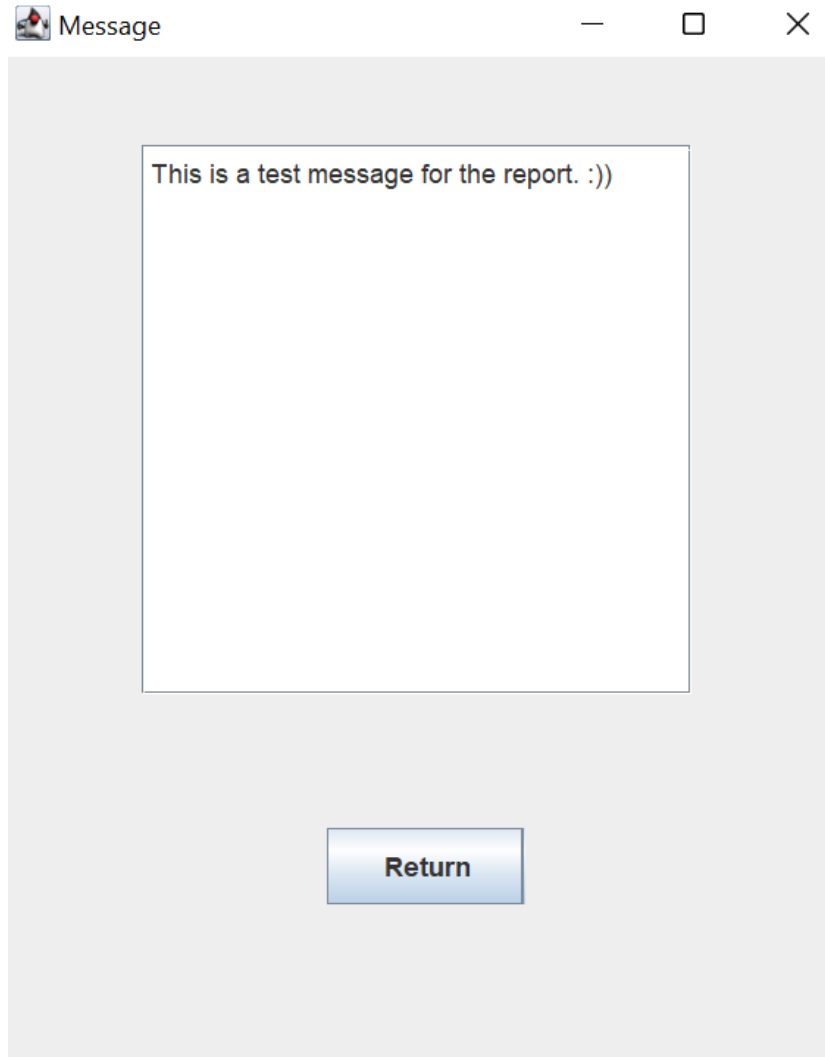
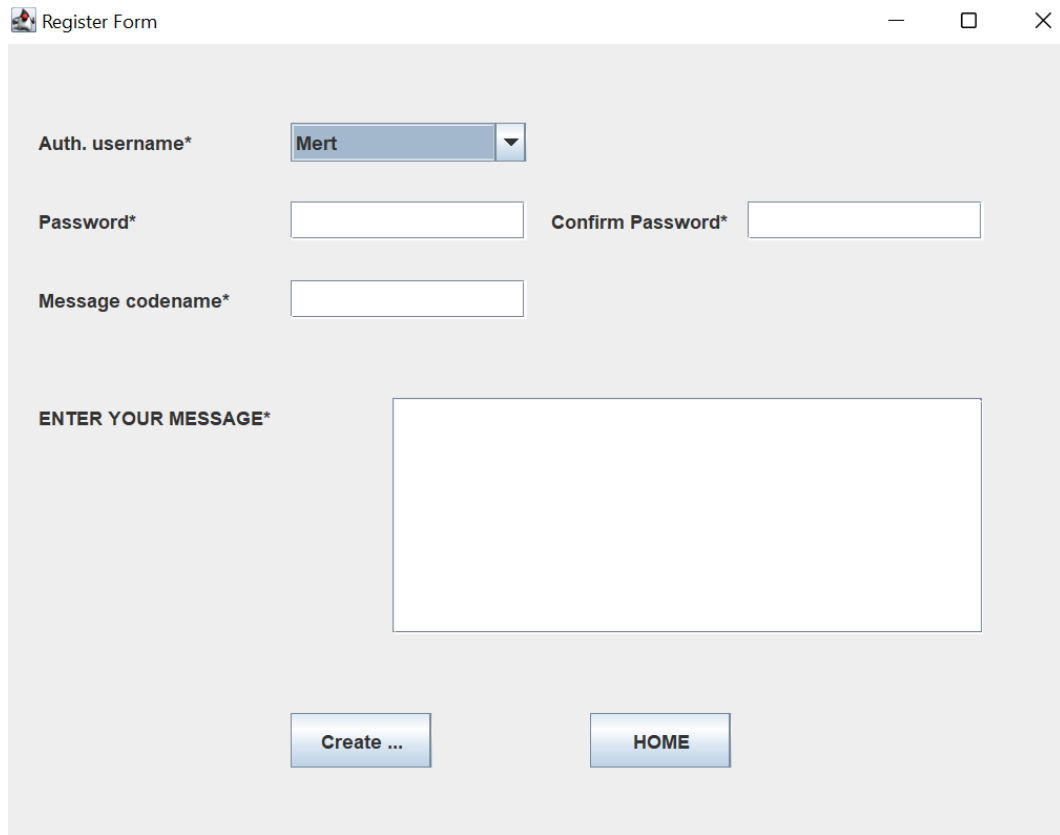


Figure 3: View Screen

After the view button is pressed some conditions are checked like do the message codename exist, password of message is correct or the user is authorized to see its contents etc. If all these conditions are met then another screen called "View Screen" appears and the content is shown to user.

4th Screen: Register Message Screen



The image shows a web application window titled "Register Form". It contains the following elements:

- Auth. username*:** A dropdown menu with "Mert" selected.
- Password*:** A text input field.
- Confirm Password*:** A text input field.
- Message codename*:** A text input field.
- ENTER YOUR MESSAGE*:** A large text area for writing the message.
- Create ...:** A button to submit the form.
- HOME:** A button to navigate back to the home screen.

Figure 4: Register Message Screen

If someone wants to leave a message for a registered user, he/she will press the "Leave a message" button in the home screen and will be navigated to the Register Message Screen. In this page someone is able to leave a message by entering some private credentials and the contents of the message. First you must choose who you are writing to. Then after you chose the recipient, you need to enter a password for your private message. This password will be hashed and kept in an encrypted file. After you've entered your password you need to confirm it by typing again. If they're matched you can type a codename for your message. Codenames must be unique and if the codename you've chosen already exists you will get an error message and your message will not be recorded. After this step if all your credentials are valid then you are able to write your message for the recipient. When you're done with all these steps press the "Create..." button and your message will be recorded in an encrypted file securely if your credentials are valid.

3 Technical Implementations

In this section we are going to mention about how passwords and other contents are kept securely in data files.

3.1 Hashing

It's not very convenient to keep passwords directly in a data file. Though, it's more suitable to keep passwords in a hashed form and store them that way. So in this program users' passwords and the passwords for messages are not kept in plain form rather they're hashed by using SHA256 algorithm and stored after getting hashed. That way if the encrypted data is stolen or decrypted, passwords are still not reversible and not reachable.

```
1 public class SHA256Hasher {
2
3     public static String hashIt(String message) {
4
5         String hashKey = "6fb0ebc3b23c927184376723f7466c3d";
6
7         Mac sha256_HMAC;
8
9         try {
10             sha256_HMAC = Mac.getInstance("HmacSHA256");
11
12             SecretKeySpec secret_key = new SecretKeySpec(hashKey.getBytes
13                 (), "HmacSHA256");
14
15             sha256_HMAC.init(secret_key);
16
17             String hash = Base64.getEncoder().encodeToString(sha256_HMAC.
18                 doFinal(message.getBytes()));
19
20             return hash;
21         }
22         catch (NoSuchAlgorithmException | InvalidKeyException e) {
23
24             e.printStackTrace();
25         }
26         return null;
27     }
28 }
```

3.2 Encrypting/Decrypting

2 data files are created to store some permanent data such as the users's and messages' credentials data. These 2 files contains critical and sensitive informations. So we must protect these 2 files by encrypting them. We solved this problem by using DES encryption algorithm. Every line of the content of both files gets encrypted and written to the encrypted file. So data becomes protected against attackers. Private keys are not stored in permanent memory such as HDD or SSD. They are isolated in the RAM reserved for the program while it's running.

```
28 private static void doCrypto(int cipherMode, String key, File inputFile,
29                               File outputFile) throws CryptoException {
30     try {
31         Key secretKey = new SecretKeySpec(key.getBytes(), ALGORITHM);
32
33         Cipher cipher = Cipher.getInstance(TRANSFORMATION);
34
35         cipher.init(cipherMode, secretKey);
36
37         FileInputStream inputStream = new FileInputStream(inputFile);
38
39         byte[] inputBytes = new byte[(int) inputFile.length()];
40
41         inputStream.read(inputBytes);
42
43         byte[] outputBytes = cipher.doFinal(inputBytes);
44
45         FileOutputStream outputStream = new FileOutputStream(
46             outputFile);
47
48         outputStream.write(outputBytes);
49
50         inputStream.close();
51
52         outputStream.close();
53     } catch (NoSuchPaddingException | NoSuchAlgorithmException
54             | InvalidKeyException | BadPaddingException
55             | IllegalBlockSizeException | IOException ex) {
56         throw new CryptoException("Error encrypting or decrypting file
57             ", ex);
58     }
```

4 Notes

Important: Users passwords are the same as their usernames. Such as if username is Mert then his password is also Mert. If the username is Arda his password is Arda as well.

References

- <https://docs.oracle.com/>
- <https://stackoverflow.com/>
- <https://jenkov.com/tutorials/java-cryptography/index.html>
- <https://www.javatpoint.com/java-swing>