

Hacettepe University Computer Engineering Department Information Security Lab.

Homework 2

Subject: Hashing
Due Date: 16.11.2022 23:59
Language: Java
T.A.: Ali Baran TAŞDEMİR

Introduction

You are expected to develop a simple message box system. The system will store and protect messages left by visitors. Visitors will leave a password-protected message to a registered user. Messages have a receiver (authorized user) and it is protected with a password. To view a message already left on the system the user must satisfy these conditions, 1) the message sent to the user, (in other words, authorized by the message owner) 2) the user must know the password given to the message.

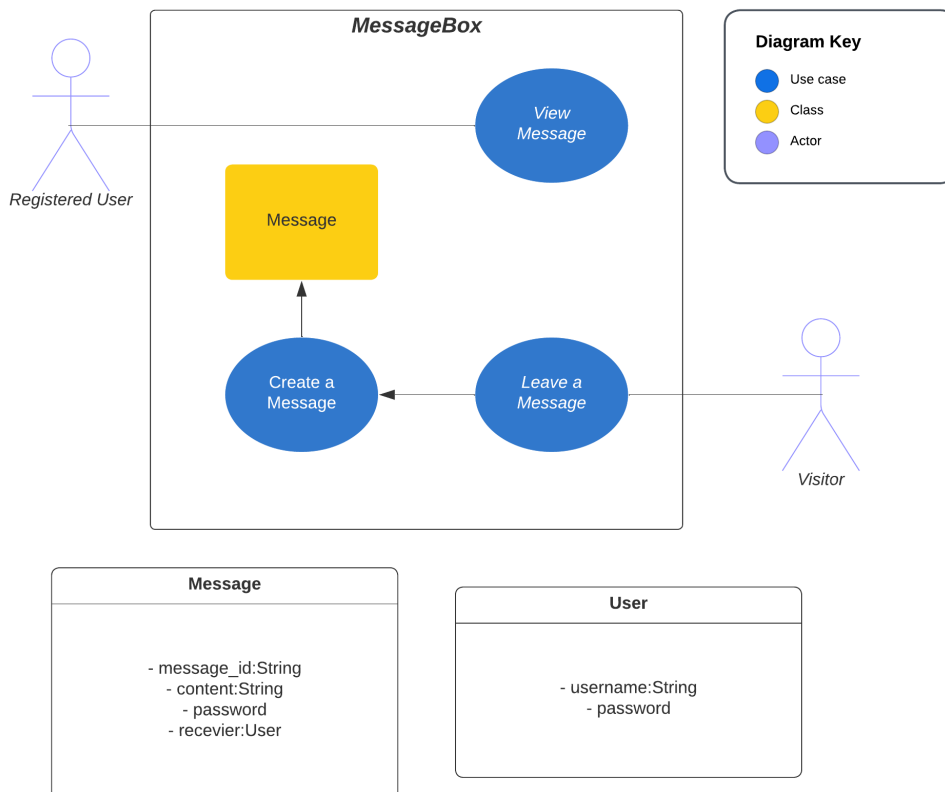


Figure 1: Use case and class diagram

Implementation Details

Message

Messages stored in the system have 4 properties; message id, content, receiver (authorized user), and password.

- Each message has a **unique** message id.
- Each message has a password and authorized user (username). The password will not be stored as raw. It must be in **hashed form**.
- Each message has a content.

User

The system has registered users. And each user has a unique username and password.

- Each user has a **unique** username.
- User passwords must be stored in **hashed form**.

Accessing/Creating Message

- A user can access a message if s/he knows the password of the message and is authorized by the sender(visitor) of the message.

- While creating a message, the visitor must provide a unique message id for the message (it should be validated by the system) and declare an authorized user (receiver). Only the authorized user can view the message.

Actors

Visitors: Visitors are anyone who has access to the program. They are not registered and they do not have any information stored on the program. They can only create a message for a user.

Users: Users are registered to the system to receive/view messages. A user has a unique username. The password of the user must be stored in hashed form. Users can view a message if s/he is authorized to view the message and knows the password of the message.

Use Cases

Leave a Message: A visitor can leave a message to the system by filling in the corresponding fields on the UI. The visitor has to provide a unique message id to the message. And this condition has to be checked by the system. If the message id is not validated the visitor shouldn't be able to create the message. The visitor has to select a receiver for the message. The receiver has to be a registered user. The created message will be accessible by this receiver later. And the visitor set a password for the message.

View a Message: Only a user can view the message. To view a message, the user has to know the message id and the message password. Also, the user must be authorized by the message (see "Leave a Message").

Requirements

For the *MessageBox* program, the requirements are listed below:

- The program must store two data files locally. The first file is *message*. The program must store registered message ids, content, and **hashed** password and authorized username in this file.
- The second file is *user*. The program must store registered usernames and their **hashed** passwords.
- Both of the datafiles must be **encrypted** to provide security. Keys for the encryption algorithm must be hard-coded in the code.
- Users can access messages by entering an id and password. If the user is authorized, s/he can see the content of the message.
- The program has 3 main functionalities; Access, Register, and View. In the *access page*, you must perform validation to give access to the user to read the message. In the *register page*, you must validate the id for messages. IDs have to be **unique**. In the *message page*, you should display the message.
- Unique fields (message id and username) must be checked by the system. The system shouldn't allow users with the same username or messages with the same message-id.
- Encrypted database files should be placed in the same directory as the program.

Graphical User Interface

The program must have a graphical user interface (GUI) using JAVA Swing. There are 4 must view for the program. These are; Home, Access, Register, and Message Views.

Home Page

The home page is the main page of the program. There should be at least two buttons to direct a user to the access page to view a message and the register page to leave a message to the system.

Access View

This page includes a form to get the credentials from the user. There should be fields for messages; message id and password, username, and user password. The page should include a "View" button. If the given credentials are correct and the user has authorization, s/he should view the message.

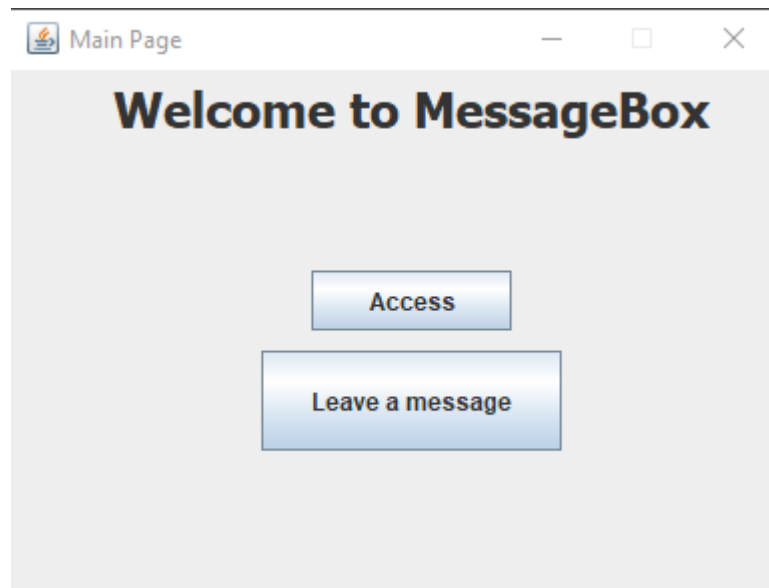


Figure 2: Home page view

Message View Page

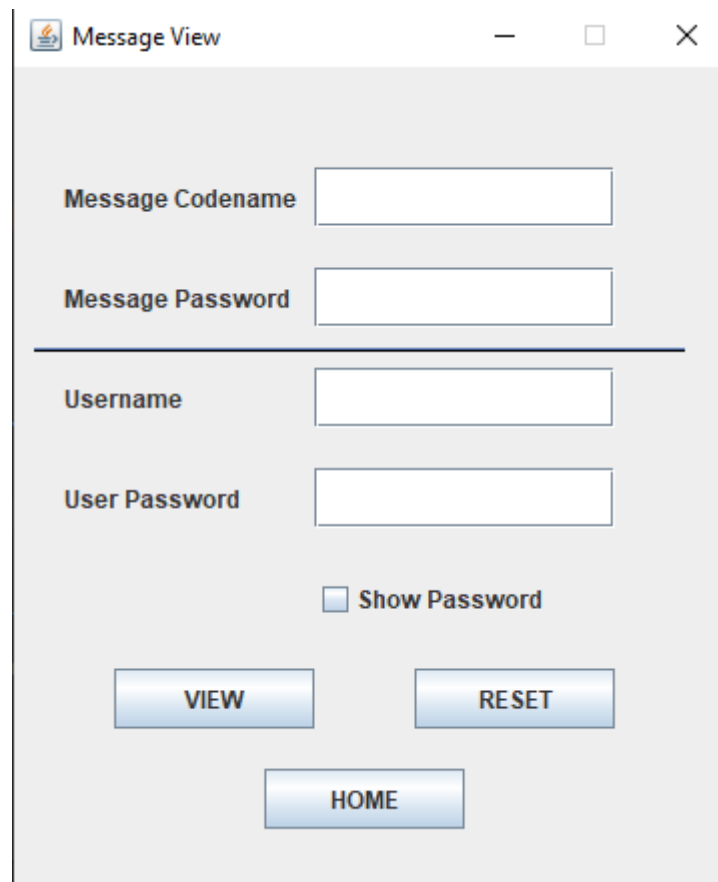
After the access page, if the user is successfully directed to the message page, s/he should view the message on the Message View page. The message should be on the frame, and there should be a button to return.

Message Register View

To create new messages, you must create a message register view that includes a registration form. This form should include corresponding fields for message content, message id, and password. Also should include the authorized username. On the form page, there should be at least two buttons, one for sending the form to create a new message, and the other one to return to the home page.

Notes

1. You must use Java Swing for the GUI.
2. Data files must be encrypted. Use *DES* algorithm for encryption.
3. You can use crypto API. For encryption/decryption you can use your codes from the previous homework.
4. You can use MD5 or SHA-256 as hashing algorithm.
5. You must submit the homework in groups of two.
6. You should prepare a report that describes your approach to the problem with the details of your implementation. You must write down all group members' names and ids. Reports will be graded too.



The image shows a screenshot of a web application window titled "Message View". The window has a standard title bar with a minimize button, a maximize button, and a close button. The main content area is light gray and contains the following elements:

- A label "Message Codename" followed by a text input field.
- A label "Message Password" followed by a text input field.
- A horizontal separator line.
- A label "Username" followed by a text input field.
- A label "User Password" followed by a text input field.
- A checkbox labeled "Show Password".
- Three buttons: "VIEW", "RESET", and "HOME". The "VIEW" and "RESET" buttons are positioned side-by-side, and the "HOME" button is centered below them.

Figure 3: Access Page View

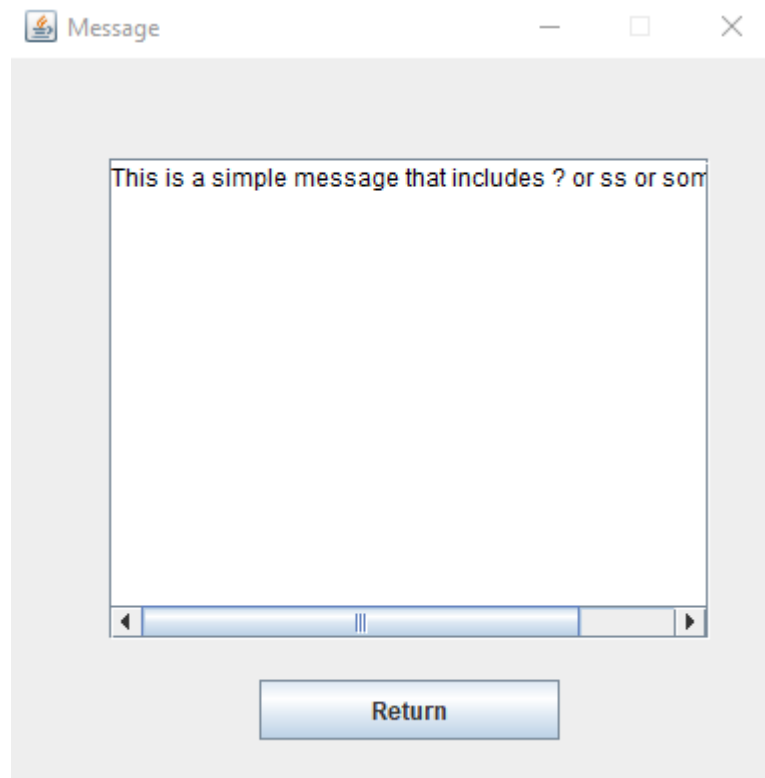


Figure 4: Message Page View

The image shows a window titled "Register Form" with standard window controls (minimize, maximize, close) in the top right corner. The form contains the following elements:

- Auth. username***: A text box containing "abt999" with a dropdown arrow on the right.
- Password***: A text box.
- Confirm Password***: A text box.
- Message codename***: A text box.
- ENTER YOUR MESSAGE***: A label next to a large, empty text area.
- Create Message**: A button.
- HOME**: A button.

Figure 5: Message Register View

7. You can ask your questions about the homework via Piazza. (www.piazza.com/hacettepe.edu.tr/fall2022/bbm465)
8. T.A. as himself has the right to partially change this document. However, the modifications will be announced in the Piazza system. In case, it is your obligation to check the Piazza course page periodically.
9. You must compile and test your code on Eclipse Platform for Windows before submission.
10. You will submit your work via the submission system.
(www.submit.cs.hacettepe.edu.tr)
The submission format is given below:
 - <group id.zip>
 - src /*.java
 - report.pdf

Policy

All work on assignments must be done with your own group unless stated otherwise. You are encouraged to discuss with your classmates about the given assignments, but these discussions should be carried out in an abstract way. That is, discussions related to a particular solution to a specific problem (either in actual code or in pseudo-code) will not be tolerated. In short, turning in someone else's work (from the internet), in whole or in part, as your own will be considered a violation of academic integrity. Please note that the former condition also holds for the material found on the web as everything on the web has been written by someone else.