*GTU Department of Computer Engineering*

CSE 222 - Spring 2023

Homework 1 Report

*DOĞUKAN TAŞTAN 1901042627*

Detailed system requirements

**login(account[] allAccounts):** The login function basically performs the login process by making the boolean value in the account object true. It also checks if the user who is already logged in tries to log in again and takes all account objects and does not allow another user to log in if the boolean value is true.

**addPost(post obj):** The AddPost function must be called by a logged in user, otherwise it will not be able to perform the operation. It adds the incoming post object to the post array in the account.

**viewPosts(account obj):** The viewPosts function shows the user's posts, but this function is not directly accessible. The profile must be viewed first and then this function must be called.

**viewInteractions(account obj):** The viewInteractions function prints the like and comment information of the user's posts.

**viewProfile(account obj):** The viewProfile function scrolls through the array inside the incoming object and prints the information to the screen. However, before it works, it checks if the user has blocked the other user, if so, the operation does not take place.

**logout():** The logout () function terminates the session of a logged in user by making the boolean value false again. Again, as in login, it prevents misuse by performing control operations.

**follow(account obj):** The Follow function adds the incoming object to the follow array of the called object.

**checkMessageBox():** This function prints the number of messages in the inbox and outbox.

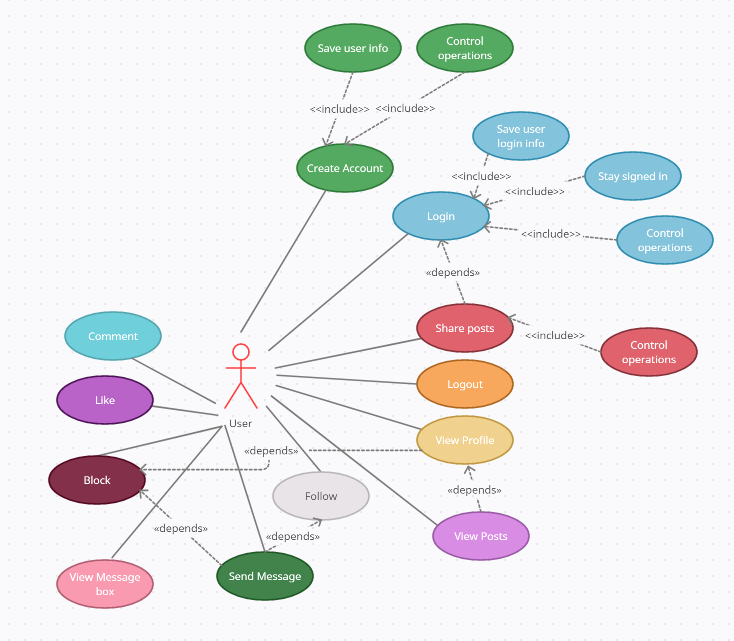
**viewInbox():** Prints incoming messages in detail.

**block(account obj):** This function adds the incoming object to the blocked array

**addLike(like obj):** This method of the post object adds the like object to the like array inside the post object..

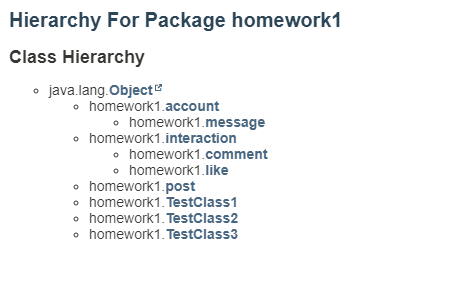
**addComment:** This method of the post object adds the comment object to the comment array inside the post object..

* 1. The Project use case diagram



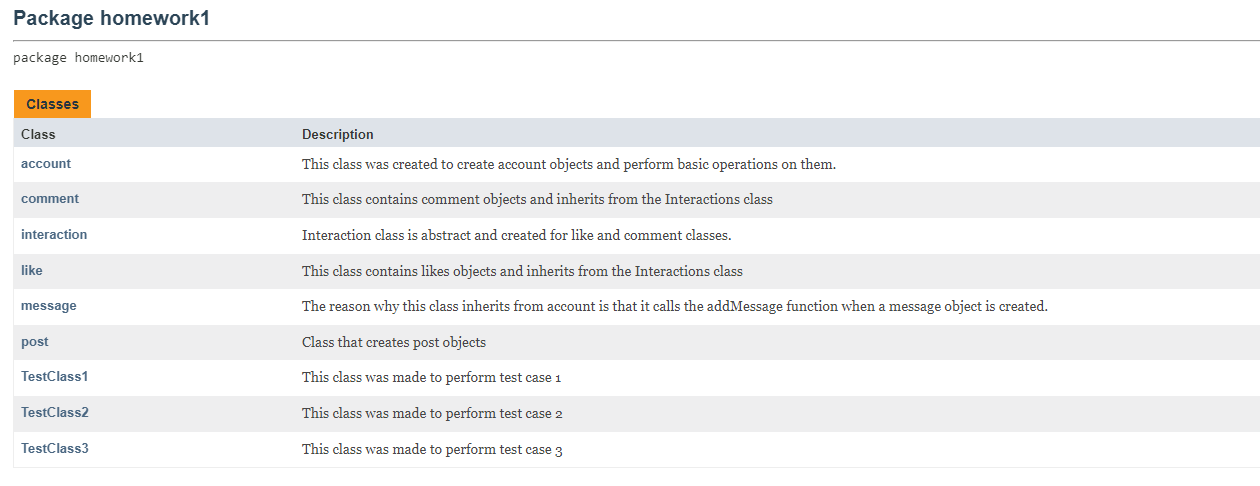
* 1. Problem solutions approach

While designing the project, it was found appropriate to inherit from 2 different classes.

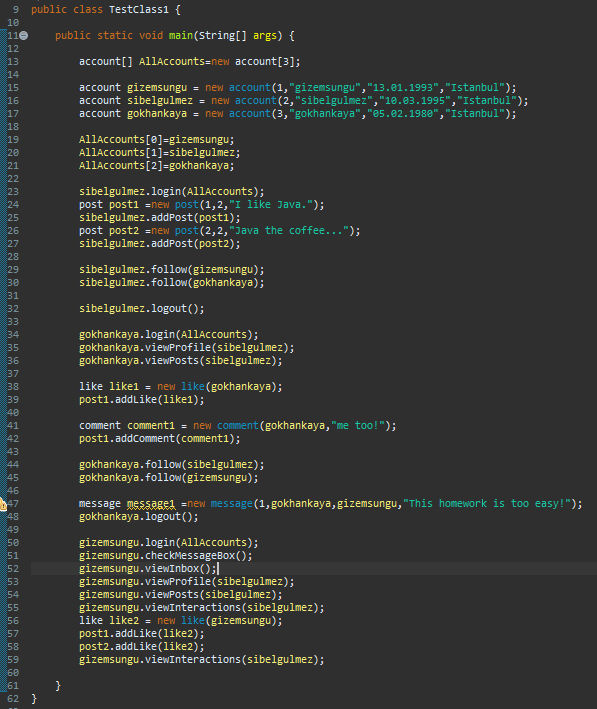
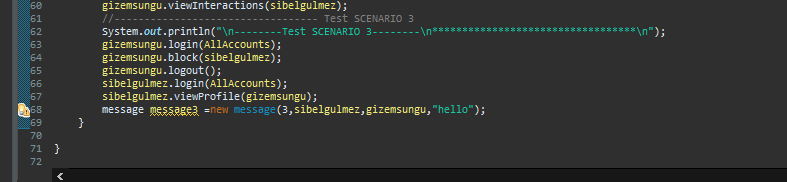


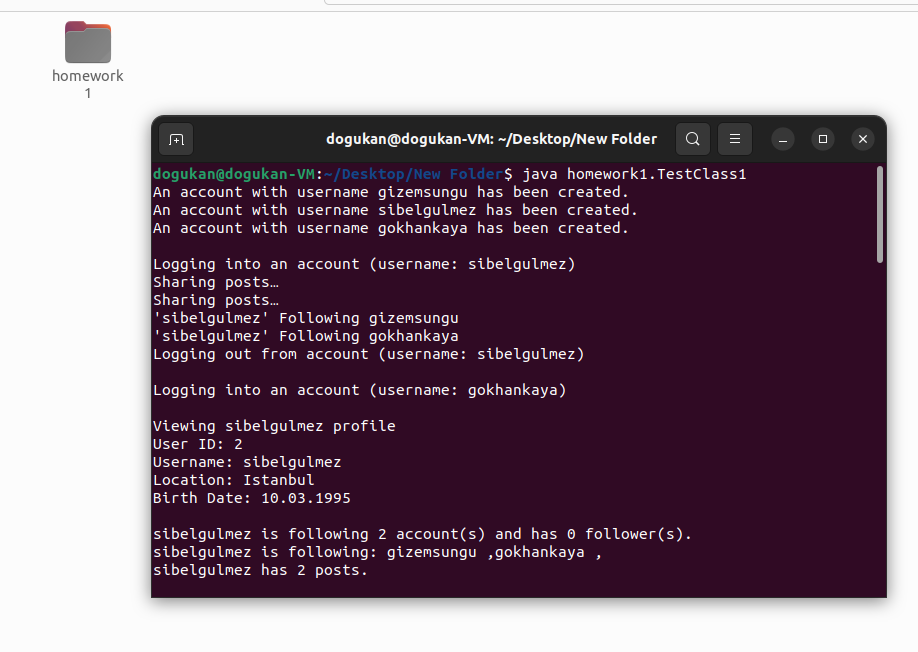
The message class inherits from account, the main reason for this is that after creating the message object in the constructor, it needs to be added to the array in the account object without additional processing.

Interaction class is the base class of like and comment classes.



The code structure consisting of 6 main classes and 3 test classes provided a solution to the problems.

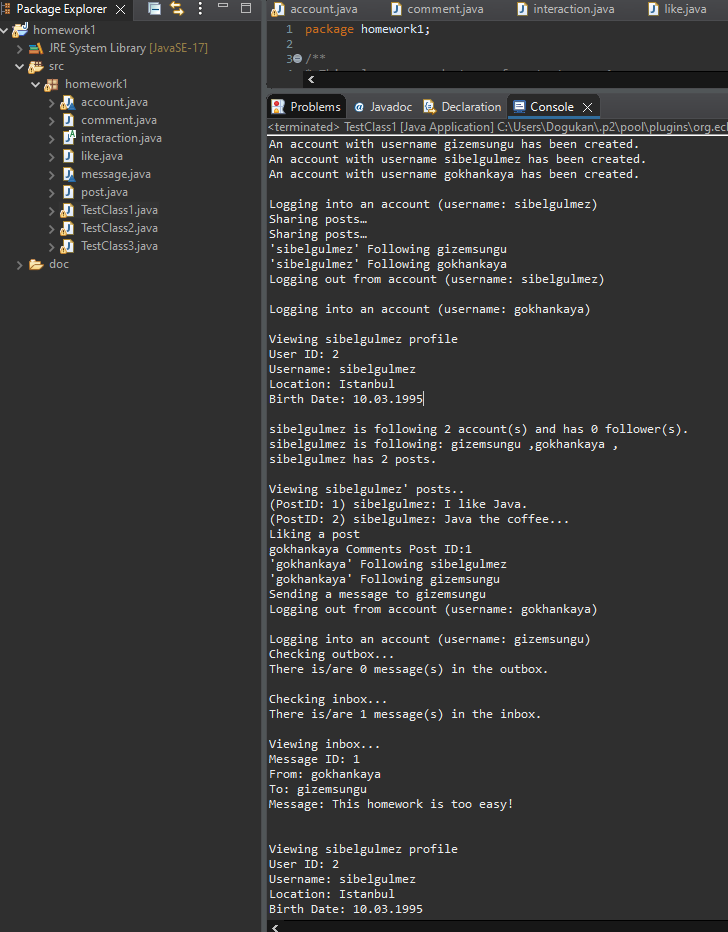
* 1. Test cases
  2. 3 test cases were created with different test classes.
  3. TestClass 1:
  4. 
  5. TestClass 2:
  6. 
  7. TestClass 3:
  8. 

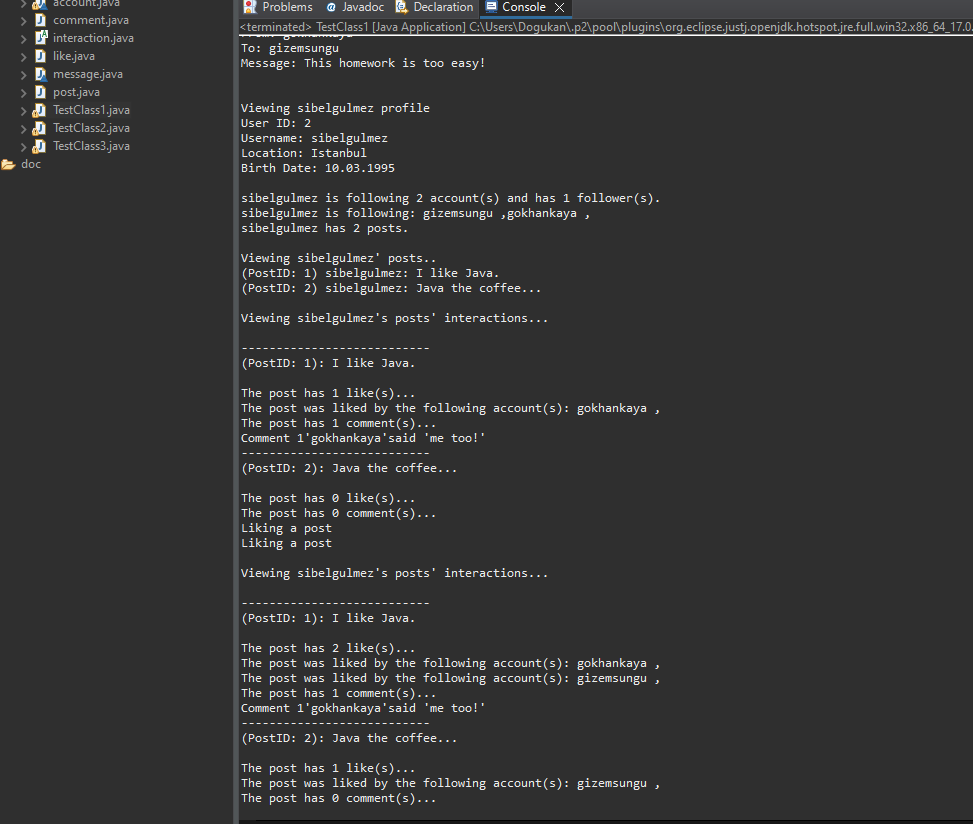


Screenshot showing it running on Linux

Running command and results

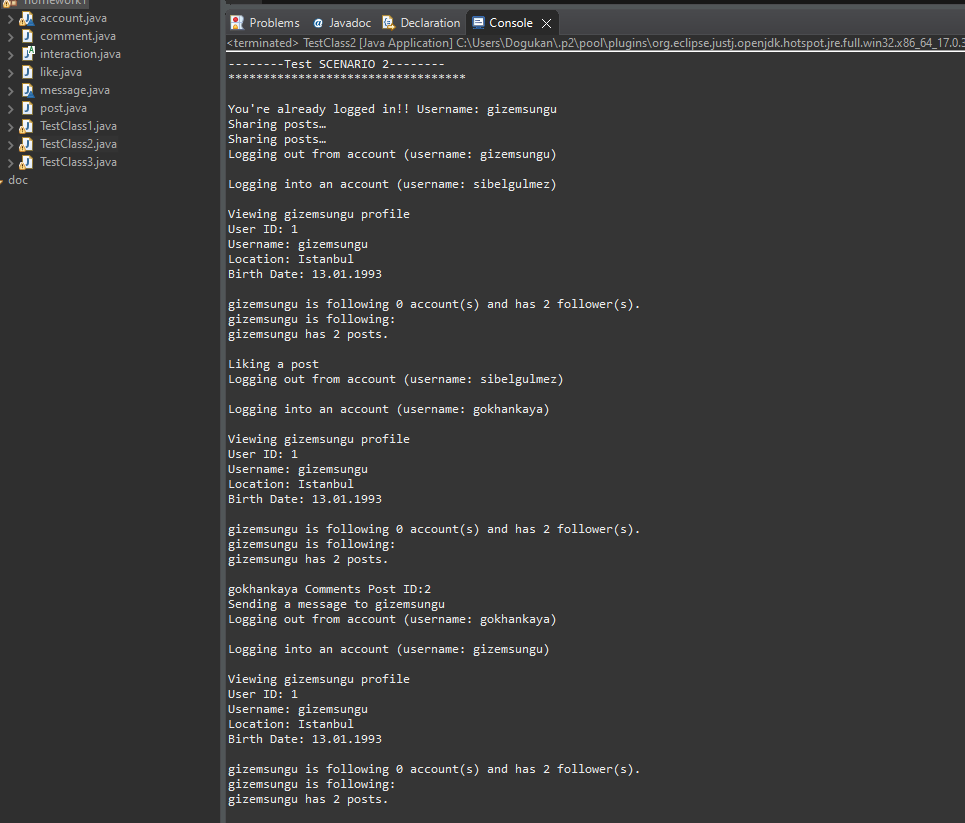
TestClass1 :

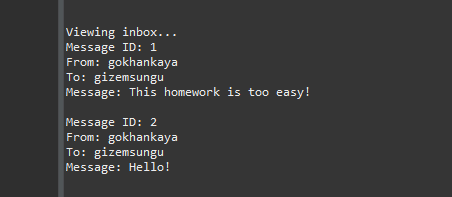




Test 1 results are as above and agree with the pdf.

TestClass 2 :





TestClass3 :

