Programming 3 Homework

ATM Machine

Ashiq Muhammad

EDOXAM

1 Project Description

The project is mainly depending on the development of ATM machine. An Automatic Teller Machine (ATM) is a computer-based machine, connected to a network, that offers, as basic functions to users, access to bank account (balance, bank transfers) and retrieval of money.

Stakeholders:

1) User 2) System Administrator

This project is designed in such a way that the user must enter pin number. Once verified, he/she is provided a menu and he/she must enter the option provided in the menu. For example, when the user wants to withdraw cash, he/she must enter the option for withdraw and from which account he/she wants to withdraw. After the option is entered along with the respective argument, if the transaction is possible the amount will be deducted from account and a message will be displayed on the screen. If the transaction is not possible an error message will be displayed on the screen.

System Administrator, which is basically the maintenance person from the bank maintains the User's accounts. His task might be, update the user's information, add new user and delete a user etc.

Note:

This program works only for the user which are stored in database.

When we run the program user info such as pin will appear on the console for example "2341"

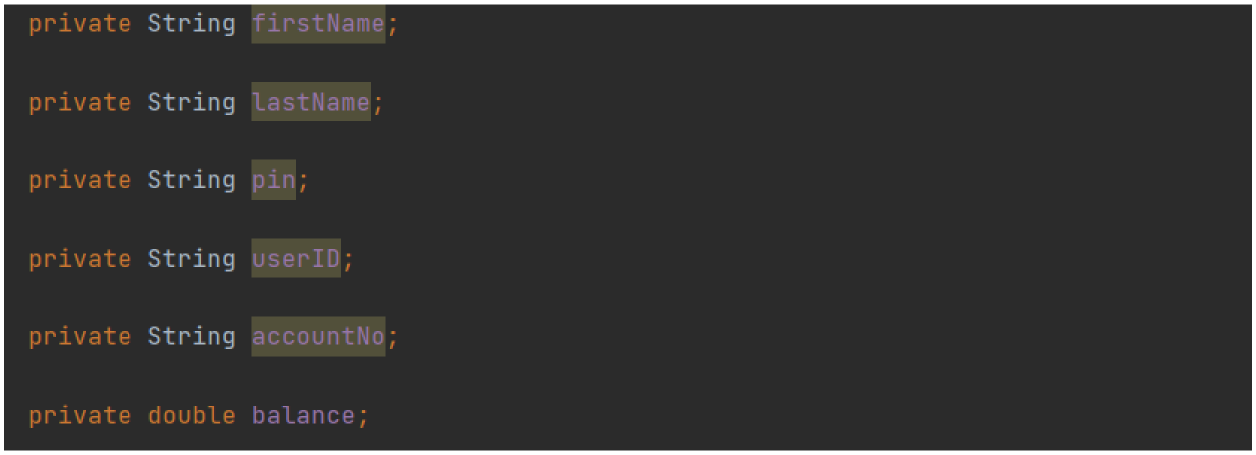
2 Development Implementation

Classes Structure

Following is the description of the class used to implement the development procedure

User

This class implements serializable. It contains the information of the user. It has following attributes

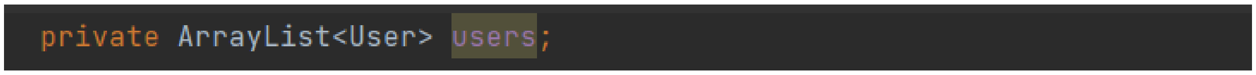


```
private String firstName;

private String lastName;

private String pin;

private String userID;

private String accountNo;

private double balance;
```

It also has different getter and setter methods to update the user information and to give access of its attributes to other classes.

Bank

The attribute it contains:



```
private ArrayList<User> users;
```

It stores the Arraylist of User object to the file("ATMFile.txt"). It also implements the task which administrator performs, such as: add new user and delete a user etc.

It also contains the main method.

ATM

The attributes for this class are:

```
    private ArrayList<User> user;
    private double balance;
    private String password;
```

It read the user's information from the file and stores in to into Arraylist of User object. Using that information, it implements the functionalities which user need to perform. For example, deposit money, change pin, withdraw cash etc.

Using information, which is implemented in the ATM class, the two more classes are created which implement the GUI part

LogInGUI

```
public class LogInGUI implements ActionListener
```

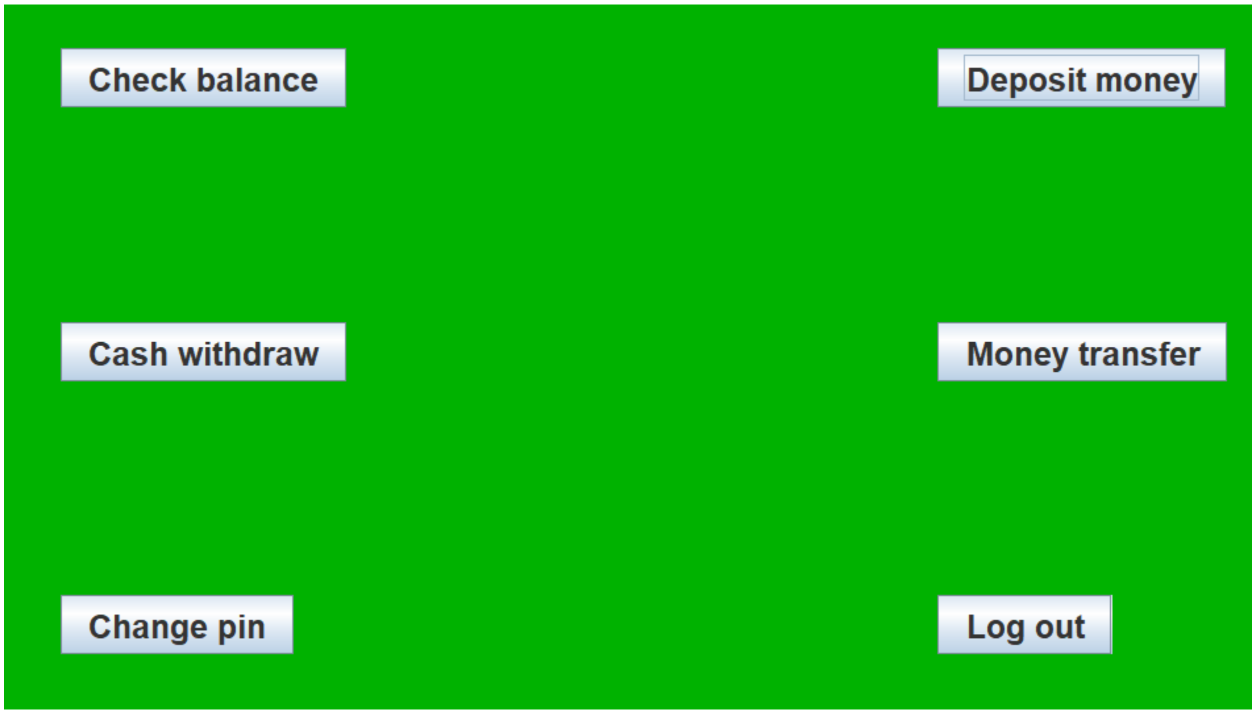
It implements the graphical user interface for log in and also for change pin.

Three JFrames are implemented here, for welcome message, for log in and for change pin.

AtmOperationGUI

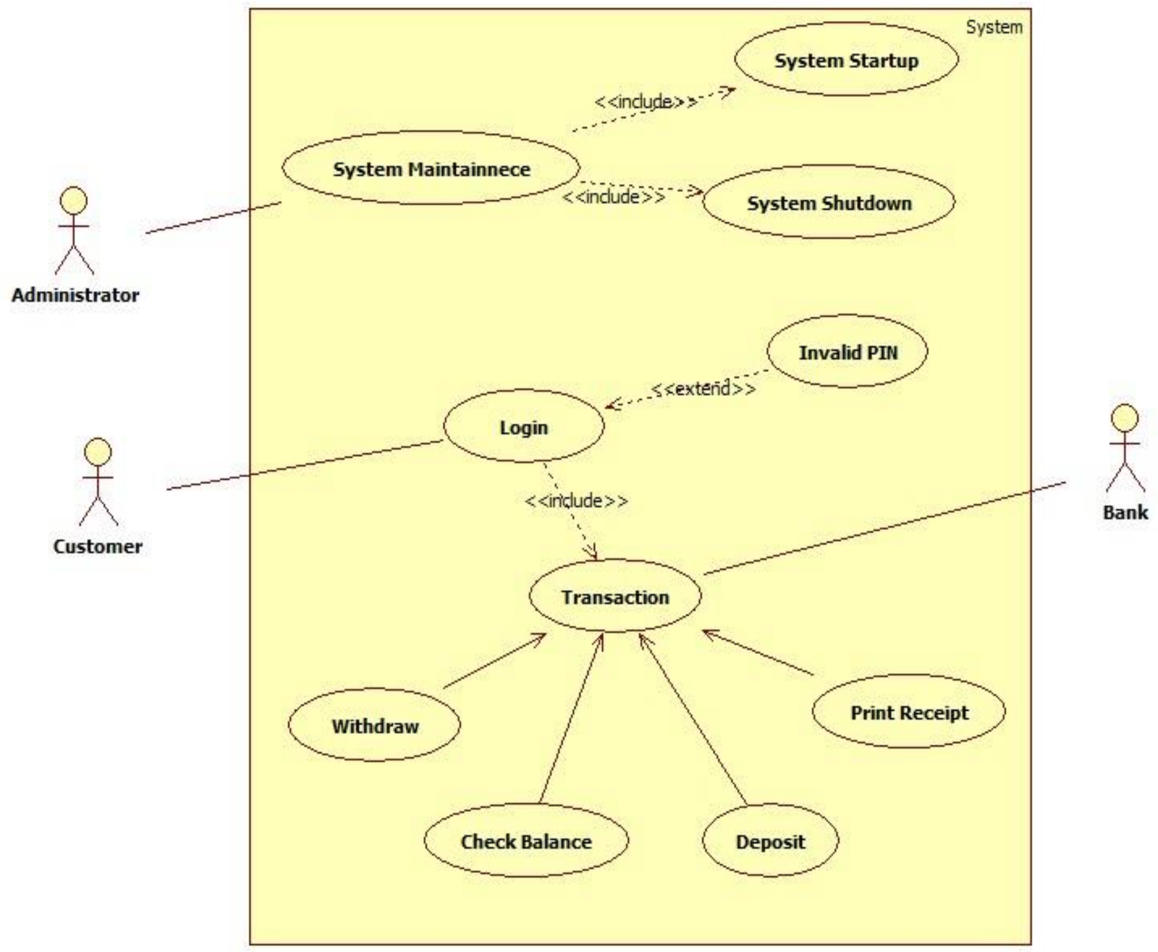
```
public class AtmOperationsGUI implements ActionListener
```

In this class the graphical user interface for the different ATM's operations is implemented.



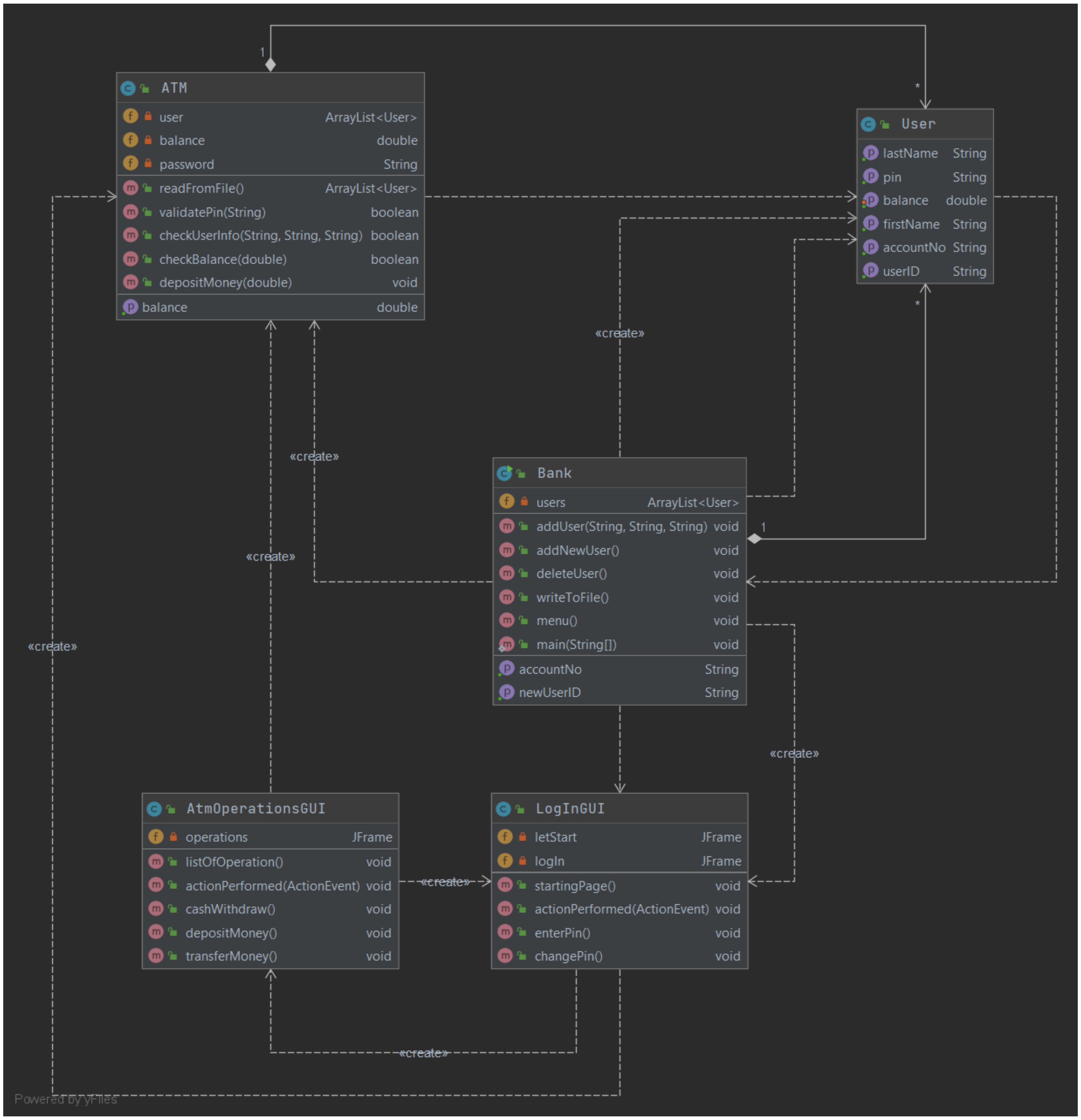
ActionListener is added to buttons shown above to implement further processes.

3 Use case Diagram

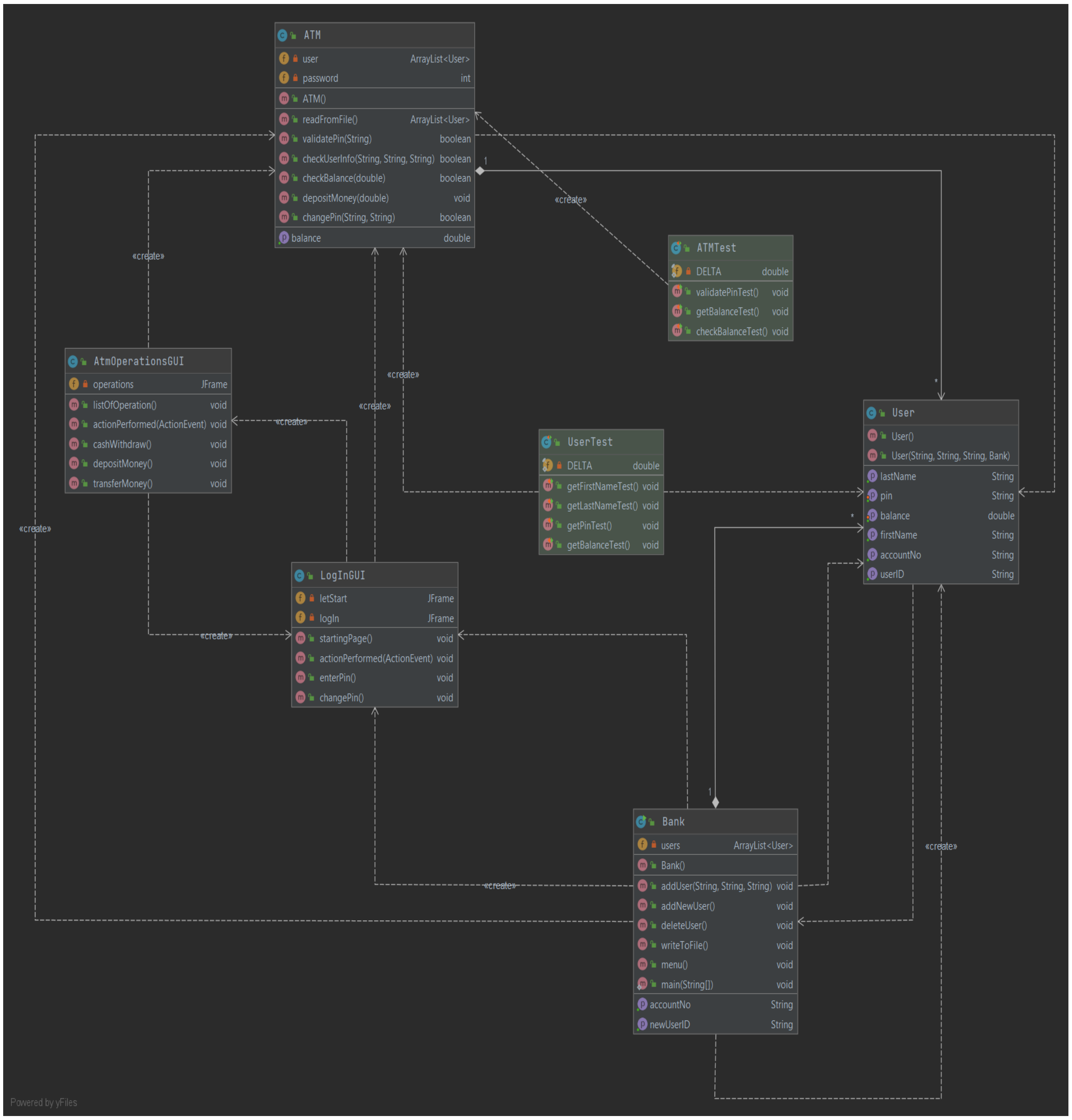


4 Class Diagram

4.1 Without Junit Test



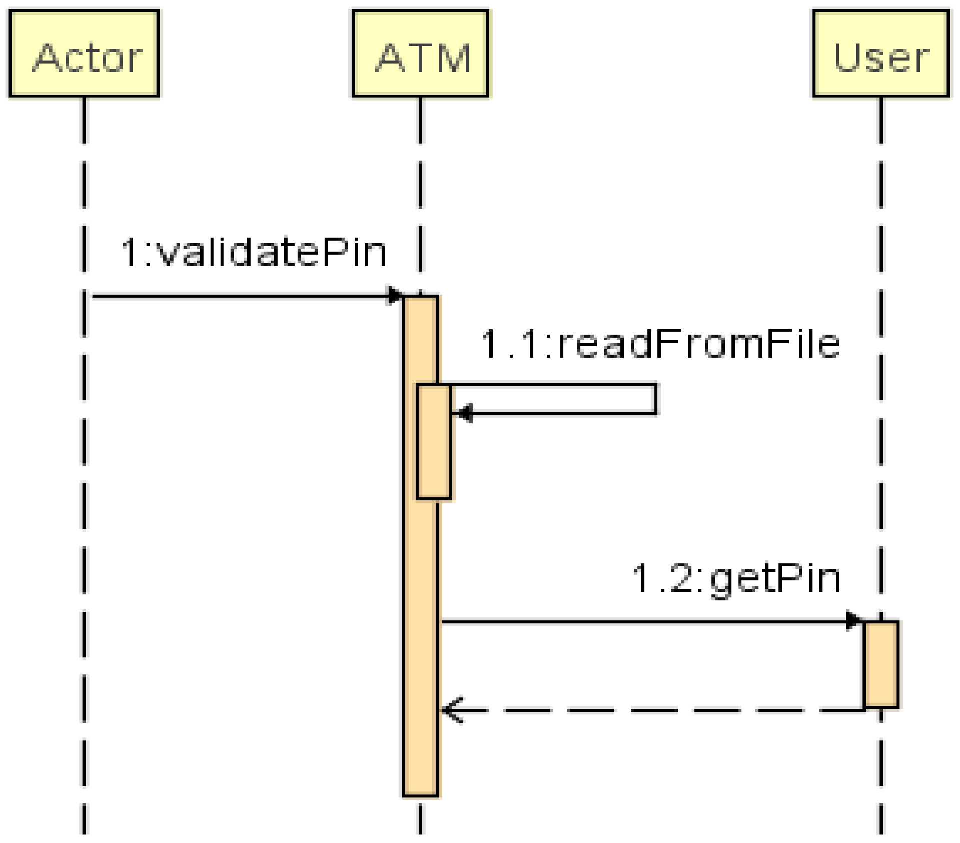
4.2 With Junit Test



5 Sequence Diagrams

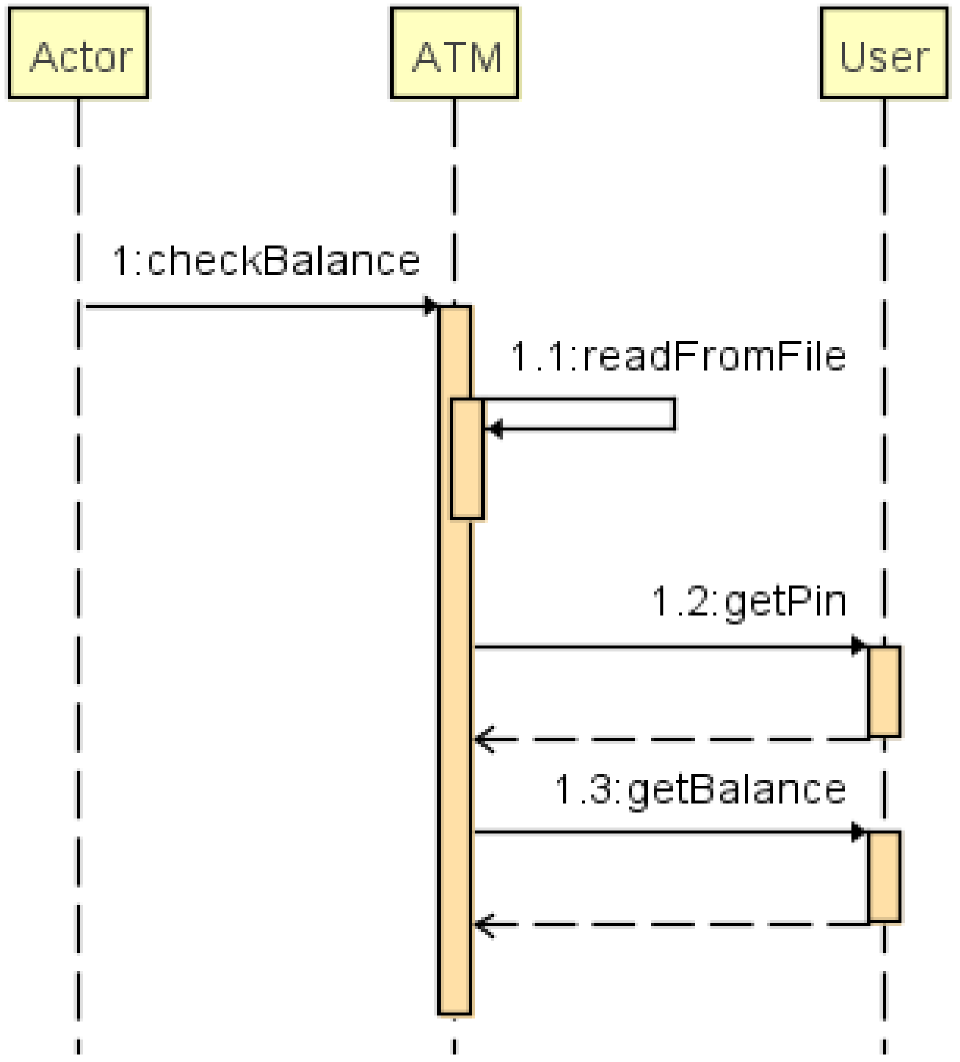
5.1

```
public boolean validatePin(String pin)
```



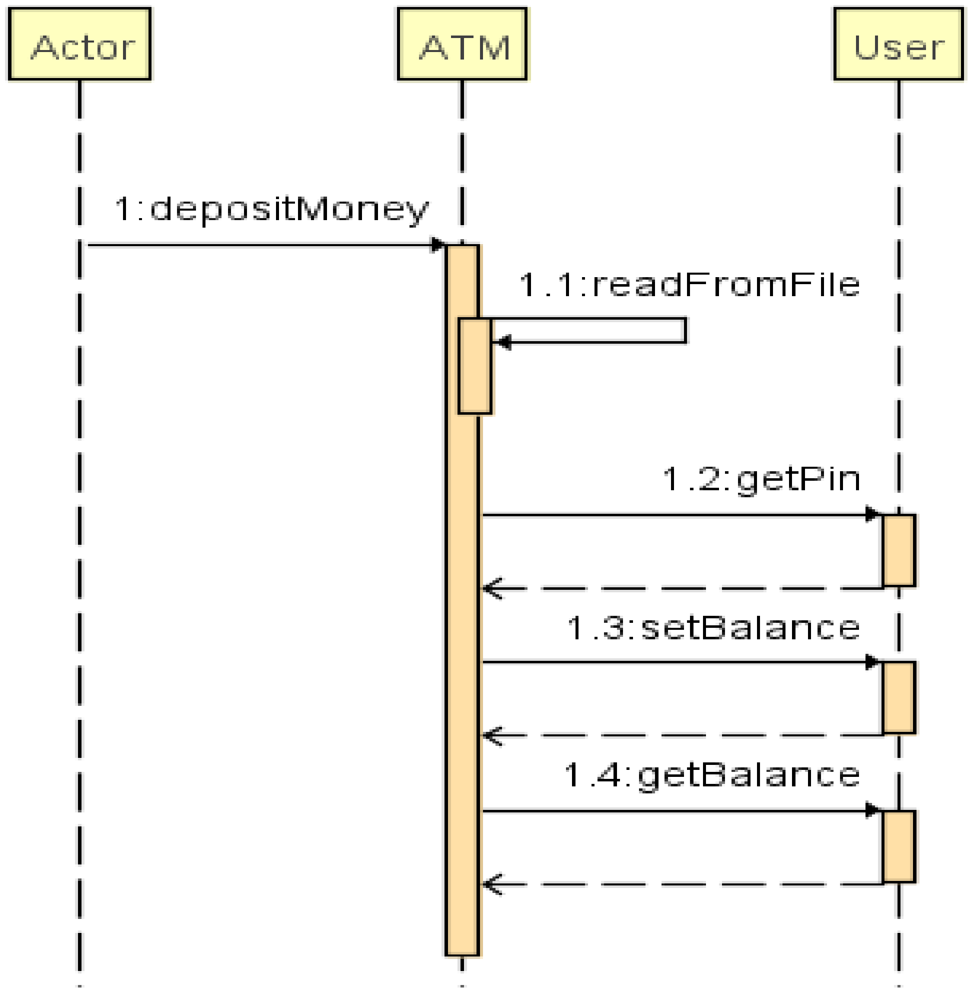
5.2

```
public  boolean checkBalance(double balance)
```

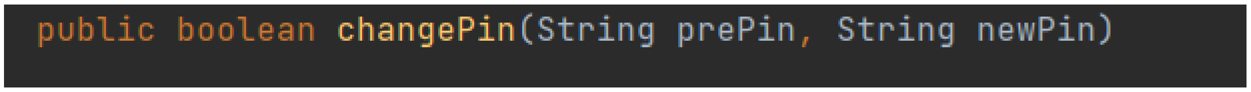


5.3

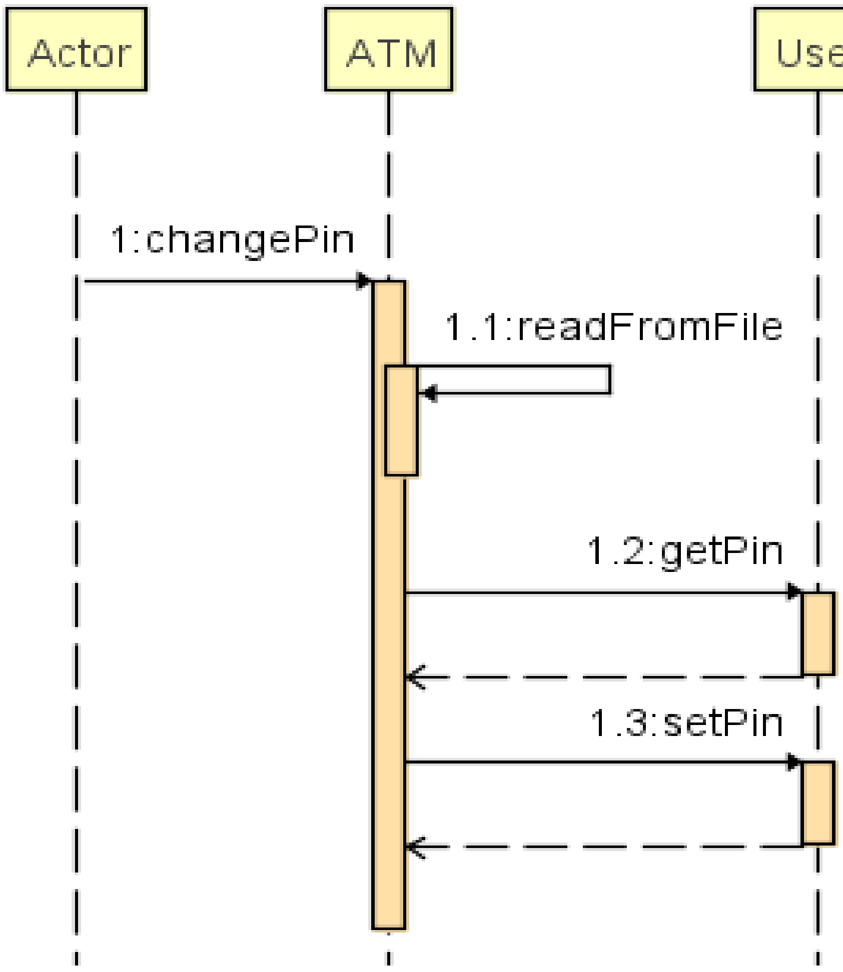
```
public void depositMoney(double money)
```



5.4

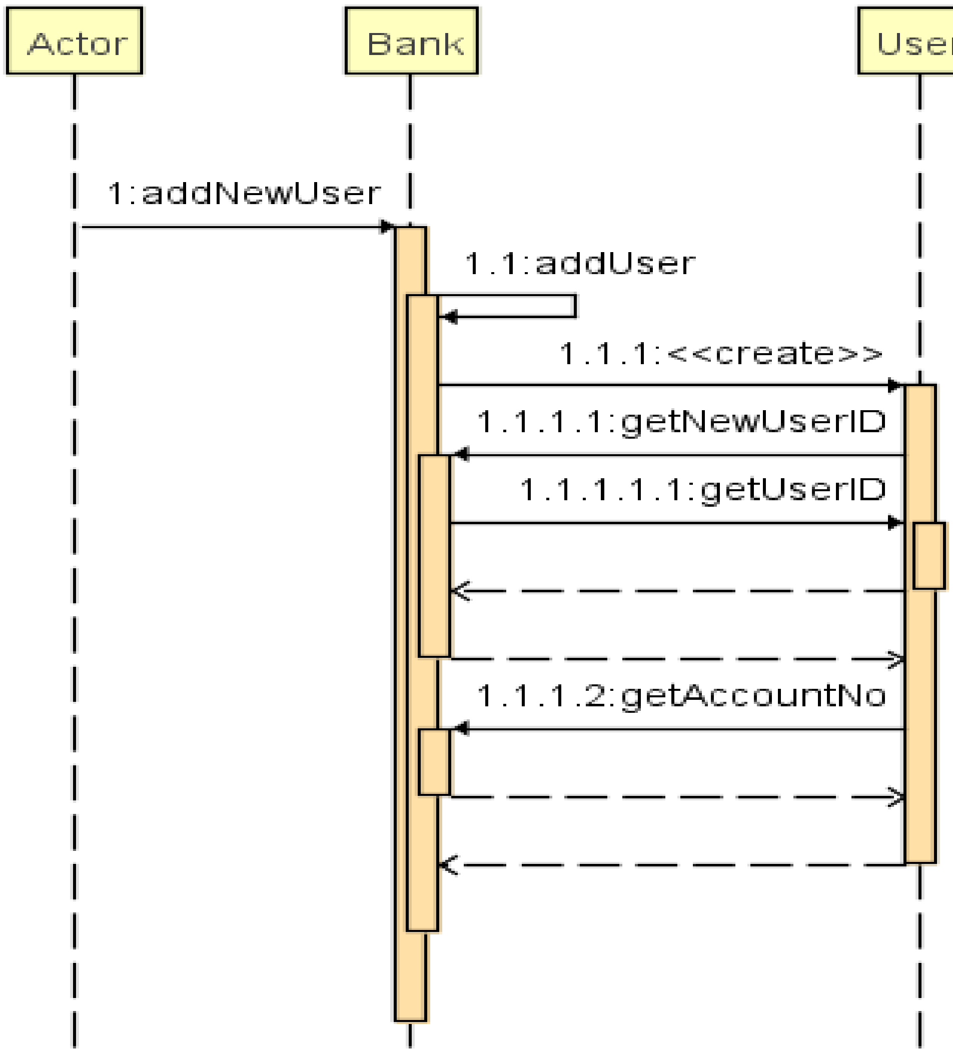


```
public boolean changePin(String prePin, String newPin)
```



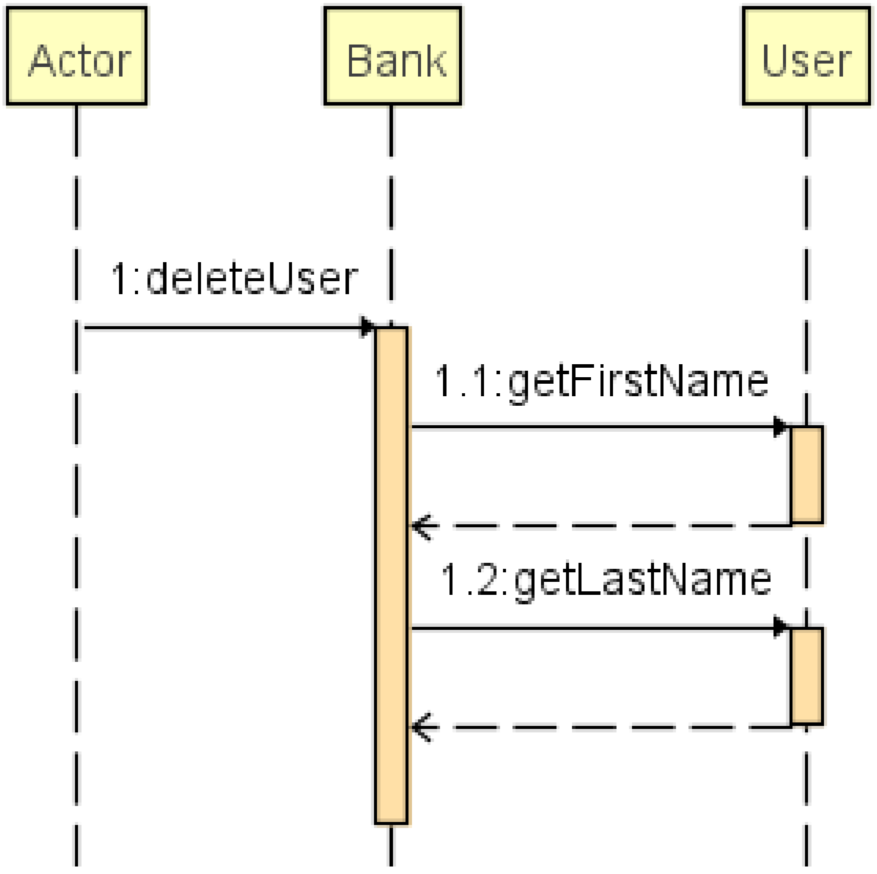
5.5

```
public void addNewUser()
```



5.6

```
public void deleteUser()
```



6 Unit Test Description

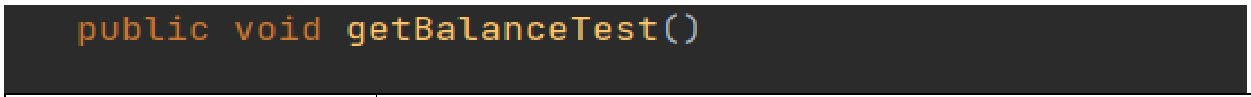
6.1 UserTest

6.1.1

```
public void validatePinTest()
```

Description	Test the method and it make sure that it works correctly
Main success scenario	Test passes if returned user's actual pin matches the expected value of pin
Alternate scenario	Test fails if return value does not match the expected value

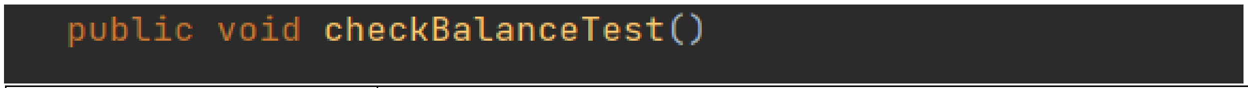
6.1.2



```
public void getBalanceTest()
```

Description	Test the method and it make sure that it works correctly
Main success scenario	Test passes if returned balance matches the expected balance of the user
Alternate scenario	Test fails if return value does not match the expected value

6.1.3

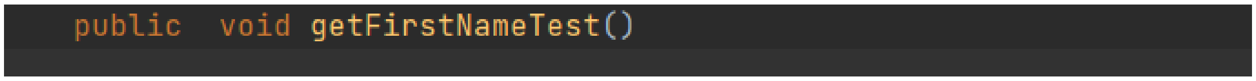


```
public void checkBalanceTest()
```

Description	Test the method and it make sure that it works correctly
Main success scenario	Test passes if returned value matches the expected value
Alternate scenario	Test fails if return value does not match the expected value

6.2 ATMTest

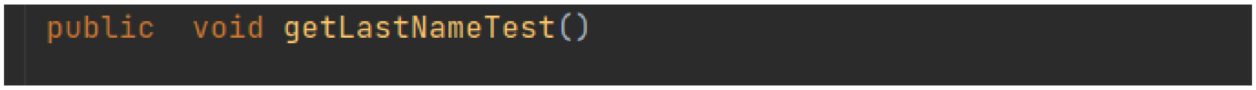
6.2.1



```
public  void getFirstNameTest()
```

Description	Test the method and it make sure that it works correctly
Main success scenario	Test passes if returned user's first name matches the expected first name
Alternate scenario	Test fails if return value does not match the expected value

6.2.2



```
public  void getLastNameTest()
```

Description	Test the method and it make sure that it works correctly
Main success scenario	Test passes if returned user's last name matches the expected first name
Alternate scenario	Test fails if return value does not match the expected value

6.2.3

```
public void getPinTest()
```

Description	Test the method and it make sure that it works correctly
Main success scenario	Test passes if returned user’s pin matches the expected pin
Alternate scenario	Test fails if return value does not match the expected value

6.2.4

```
public  void getBalanceTest()
```

Description	Test the method and it make sure that it works correctly
Main success scenario	Test passes if returned user’s balance matches the expected balance
Alternate scenario	Test fails if return value does not match the expected value