

A New Reliable ATM

OOPT Phase 2050 & 2060

Construct & Testing

Project Team T6

201411140 권성완

201511247 김선정

201510436 허윤아

201510285 조수빈

Date

2018-05-23

2050 Construct

Activity 2051. Implement Class & Methods Definitions

Type	Class
Name	MainSystem
Purpose	유저의 입력과 Offer,Account를 연결하는 클래스
Overview(Class)	Insert(), inputPassword(), printRecentTransaction(), printTransactionReceipt(), deposit(), withdraw(), transfer(), exchange(), loan(), doForcedTermination(), printError() takeCharge() depositWithoutBank() checkBalance(), payUtilityBill()
Cross Reference	Usecase: R1, R2, R3, R4, R5, R6, R7, R8, R9, R10, R11, R12, R13,R15 Functional:R1.1,R1.2,R1.3,R1.4,R1.5,R1.6,R1.7,R1.8,R2.1,R2.2,R2.3,R2.4,R3.1,R5 .1
Input(Method)	-
Output(Method)	-
Abstract operation(Method)	
Exceptional Courses of Events	-

Type	Method
Name	Insert()
Purpose	유저의 매체 입력을 받는다.
Cross Reference	Usecase: 9 Functional :R2.1
Input(Method)	-
Output(Method)	void
Abstract operation(Method)	유저의 통장/카드번호와 지로번호를 입력받는다.
Exceptional Courses of Events	-

Type	Method
Name	inputPassword()
Purpose	ATM의 기능을 이용하기 위한 password를 입력받아 withdraw 등에 사용할 수 있게 한다.

Cross Reference	Usecase: 3, 4, 5, 6, 7, 8, 12, 14 Functional: R1.3, R1.4, R1.5, R1.6, R1.7, R1.8, R2.4, R4.1,
Input(Method)	
Output(Method)	int password
Abstract operation(Method)	유저가 비밀번호를 입력하고 이를 저장한다
Exceptional Courses of Events	-

Type	Method
Name	printTransactionReceptit()
Purpose	거래명세서를 출력한다..
Cross Reference	Usecase: 10 Functional: R2.2
Input(Method)	
Output(Method)	void
Abstract operation(Method)	유저가 비밀번호를 입력하고 이를 저장한다
Exceptional Courses of Events	-

Type	Method
Name	deposit()
Purpose	예금한다.
Cross Reference	Usecase: 1 Functional: R1.1
Input(Method)	
Output(Method)	void
Abstract operation(Method)	유저가 입금할 금액을 입력하고 이를 account에 저장한다.
Exceptional Courses of Events	-

Type	Method
Name	depositWithoutBank()
Purpose	무통장 입금을 한다.
Cross Reference	Usecase: 2 Functional: R1.2
Input(Method)	-
Output(Method)	void
Abstract operation(Method)	통장이나 카드가 없어도 송금을 한다. 송금받은 계좌의 변동 사항을 account에 저장한다.
Exceptional Courses of Events	-

Type	Method
Name	withdraw()
Purpose	출금한다
Cross Reference	Usecase: 3 Functional: R1.3
Input(Method)	
Output(Method)	void
Abstract operation(Method)	유저가 출금할 금액을 입력하고 그 금액만큼의 변동사항을 account에 저장한다.
Exceptional Courses of Events	-

Type	Method
Name	transfer()
Purpose	송금한다
Cross Reference	Usecase: 4 Functional: R1.4
Input(Method)	
Output(Method)	void
Abstract operation(Method)	유저가 송금할 금액을 입력하고 그 금액만큼의 변동사항을 유저의 account와 송금 받는 사람의 account에 저장한다.
Exceptional Courses of Events	-

Type	Method
Name	exchange()
Purpose	환전한다
Cross Reference	Usecase: 5 Functional: R1.5
Input(Method)	
Output(Method)	void
Abstract operation(Method)	유저가 환전할 금액을 입력하고 그 금액만큼의 환전 후 변동사항을 account에 저장한다.
Exceptional Courses of Events	-

Type	Method
Name	loan()

Purpose	대출한다
Cross Reference	Usecase: 6 Functional: R1.6
Input(Method)	
Output(Method)	Void
Abstract operation(Method)	유저가 대출할 금액을 입력하고 그 금액만큼의 변동사항을 account에 저장한다.
Exceptional Courses of Events	-

Type	Method
Name	doForcedTermination()
Purpose	초기화면으로 돌아간다
Cross Reference	Usecase: 12 Functional: R 2.4
Input(Method)	
Output(Method)	void
Abstract operation(Method)	에러가 났을 경우나 선택한 기능이 끝난 경우 초기화면으로 돌아간다.
Exceptional Courses of Events	-

Type	Method
Name	printError(int errorType)
Purpose	에러를 출력한다
Cross Reference	Usecase: 11 Functional: R2.3
Input(Method)	Int errorType
Output(Method)	void
Abstract operation(Method)	errorType(error code)에 따라 맞는 에러를 출력한다.
Exceptional Courses of Events	-

Type	Method
Name	takeCharge()
Purpose	수수료를 부과한다
Cross Reference	Usecase: 13 Functional: R3.1
Input(Method)	
Output(Method)	void

Abstract operation(Method)	수수료를 등급에 맞게 계산하여 부과하고 그에 따른 변동사항을 계좌에 저장한다.
Exceptional Courses of Events	-

Type	Method
Name	checkBalance()
Purpose	거래내역을 확인한다
Cross Reference	Usecase: 8 Functional: R1.8
Input(Method)	
Output(Method)	void
Abstract operation(Method)	최근 계좌의 거래내역을 출력한다.
Exceptional Courses of Events	-

Type	Method
Name	payUtilityBill()
Purpose	공과금을 수납한다
Cross Reference	Usecase: 7 Functional: R1.7
Input(Method)	
Output(Method)	void
Abstract operation(Method)	국가계좌로 공과금을 납부한다
Exceptional Courses of Events	-

Type	Class
Name	Account
Purpose	MainSystem과 offer의 Database(test file)사이의 정보를 연결하는 클래스. MainSystem은 Account클래스의 정보를 변경하고, 이 Account클래스의 정보를 offer의 Database에 저장한다.
Overview(Class)	getPassword(), getLimit(), getBalance(), getIsLocked(), getBank(), getRate(), getDept(), setLimit(int limit), setBalance(int balance), setDept(int dept), setIsLocked(Boolean isLocked), setRate(int rate), setBank(int bank), setPassword(int password), checkPassword(int password)
Cross Reference	Usecase: 12, 14, 16, 17 Functional: R2.4, R4.1, R6.1, R6.2
Input(Method)	-

Output(Method)	-
Abstract operation(Method)	
Exceptional Courses of Events	-

Type	Method
Name	getPassword()
Purpose	Account의 password를 가져온다.
Cross Reference	Usecase: 14, 16,17 Functional: R4.1. R6.1, R6.2
Input(Method)	-
Output(Method)	int password
Abstract operation(Method)	account의 password를 읽고 리턴한다.
Exceptional Courses of Events	-

Type	Method
Name	getLimit()
Purpose	Account의 limit를 가져온다.
Cross Reference	Usecase: 13, 16,17 Functional: R3.1 R6.1, R6.2
Input(Method)	-
Output(Method)	int limit
Abstract operation(Method)	account의 대출 한도를 리턴한다.
Exceptional Courses of Events	-

Type	Method
Name	getBalance()
Purpose	Account의 balance를 가져온다.
Cross Reference	Usecase: 8,16,17 Functional: R1.8, R6.1, R6.2
Input(Method)	-
Output(Method)	int balance
Abstract operation(Method)	account의 남은 잔액을 리턴한다.
Exceptional Courses of Events	-

Type	Method
Name	getIsLocked()

Purpose	Account의 isLocked를 가져온다.
Cross Reference	Usecase: 15, 16,17 Functional: R5.1, R6.1, R6.2
Input(Method)	-
Output(Method)	Boolean isLocked
Abstract operation(Method)	account가 잠긴 상태인지 리턴한다.
Exceptional Courses of Events	-

Type	Method
Name	getBank()
Purpose	Account의 bank를 가져온다.
Cross Reference	Usecase: 16,17 Functional: R6.1, R6.2
Input(Method)	-
Output(Method)	String bank
Abstract operation(Method)	account의 은행 정보를 리턴한다.
Exceptional Courses of Events	-

Type	Method
Name	getRate()
Purpose	Account의 rate를 가져온다.
Cross Reference	Usecase: 13, 16,17 Functional: R3.1 R6.1, R6.2
Input(Method)	-
Output(Method)	int rate
Abstract operation(Method)	account의 등급을 리턴한다.
Exceptional Courses of Events	-

Type	Method
Name	getDept()
Purpose	Account의 dept를 가져온다.
Cross Reference	Usecase: 16,17 Functional: R6.1, R6.2
Input(Method)	-
Output(Method)	String bank
Abstract operation(Method)	account의 빚 금액을 리턴한다.

Exceptional Courses of Events	-
--------------------------------------	---

Type	Method
Name	setLimit(int limit)
Purpose	Account의 limit를 설정한다.
Cross Reference	Usecase: 13, 16,17 Functional: R3.1 R6.1, R6.2
Input(Method)	int limit
Output(Method)	void
Abstract operation(Method)	account의 대출 한도를 설정한다.
Exceptional Courses of Events	-

Type	Method
Name	setBalance(int balance)
Purpose	Account의 balance를 설정한다.
Cross Reference	Usecase: 8,16,17 Functional: R1.8, R6.1, R6.2
Input(Method)	int balance
Output(Method)	void
Abstract operation(Method)	account의 남은 잔액을 설정한다.
Exceptional Courses of Events	-

Type	Method
Name	setDept(int dept)
Purpose	Account의 dept를 설정한다.
Cross Reference	Usecase: 16,17 Functional: R6.1, R6.2
Input(Method)	Int dept
Output(Method)	void
Abstract operation(Method)	Account의 대출로 발생한 빚을 설정한다.
Exceptional Courses of Events	-

Type	Method
Name	setIsLocked(boolean isLocked)
Purpose	Account의 isLocked를 설정한다.
Cross Reference	Usecase: 15, 16,17

	Functional: R5.1, R6.1, R6.2
Input(Method)	Boolean isLocked
Output(Method)	Void
Abstract operation(Method)	Account가 transitionLock 상태인지 설정한다.
Exceptional Courses of Events	-

Type	Method
Name	setRate(int rate)
Purpose	Account의 rate를 설정한다.
Cross Reference	Usecase: 13, 16,17 Functional: R3.1 R6.1, R6.2
Input(Method)	int rate
Output(Method)	Void
Abstract operation(Method)	account의 신용등급을 설정한다.
Exceptional Courses of Events	-

Type	Method
Name	setBank(int bank)
Purpose	Account의 bank를 설정한다.
Cross Reference	Usecase: 16,17 Functional: R6.1, R6.2
Input(Method)	int bank
Output(Method)	void
Abstract operation(Method)	Int로 offer code같은 정보를 받아 string인 bank를 설정한다.
Exceptional Courses of Events	-

Type	Method
Name	setPassword(int password)
Purpose	Account의 password를 설정한다.
Cross Reference	Usecase: 14, 16,17 Functional: R4.1. R6.1, R6.2
Input(Method)	int password
Output(Method)	void
Abstract operation(Method)	Account의 비밀번호를 설정한다.
Exceptional Courses of Events	-

Type	Method
Name	checkPasword(int password)
Purpose	Account의 password와 입력한 password가 일치하는지 확인한다.
Cross Reference	Usecase: 14, 16,17 Functional: R4.1. R6.1, R6.2
Input(Method)	int password
Output(Method)	boolean
Abstract operation(Method)	withdraw같은 과정 중에 유저가 입력한 password와 account에 설정된 password가 일치하는지 확인하고 true/false를 리턴한다.
Exceptional Courses of Events	-

Type	Class
Name	Offer
Purpose	Database(text file)을 관리하고 atm(main system)에 정보를 넘겨주는 클래스
Overview(Class)	Offer(int company), updateDatabase(Account account), checkValid(Account account), readDatabase(Account account)
Cross Reference	Usecase: Functional:
Input(Method)	-
Output(Method)	-
Abstract operation(Method)	
Exceptional Courses of Events	-

Type	Constructor
Name	Offer(int company)
Purpose	Int company의 값에 따라 맞는 Offer를 생성한다.
Cross Reference	-
Input(Method)	Int company
Output(Method)	-
Abstract operation(Method)	Int company값에 따라 각각 다른 offer를 생성하는 생성자
Exceptional Courses of Events	-

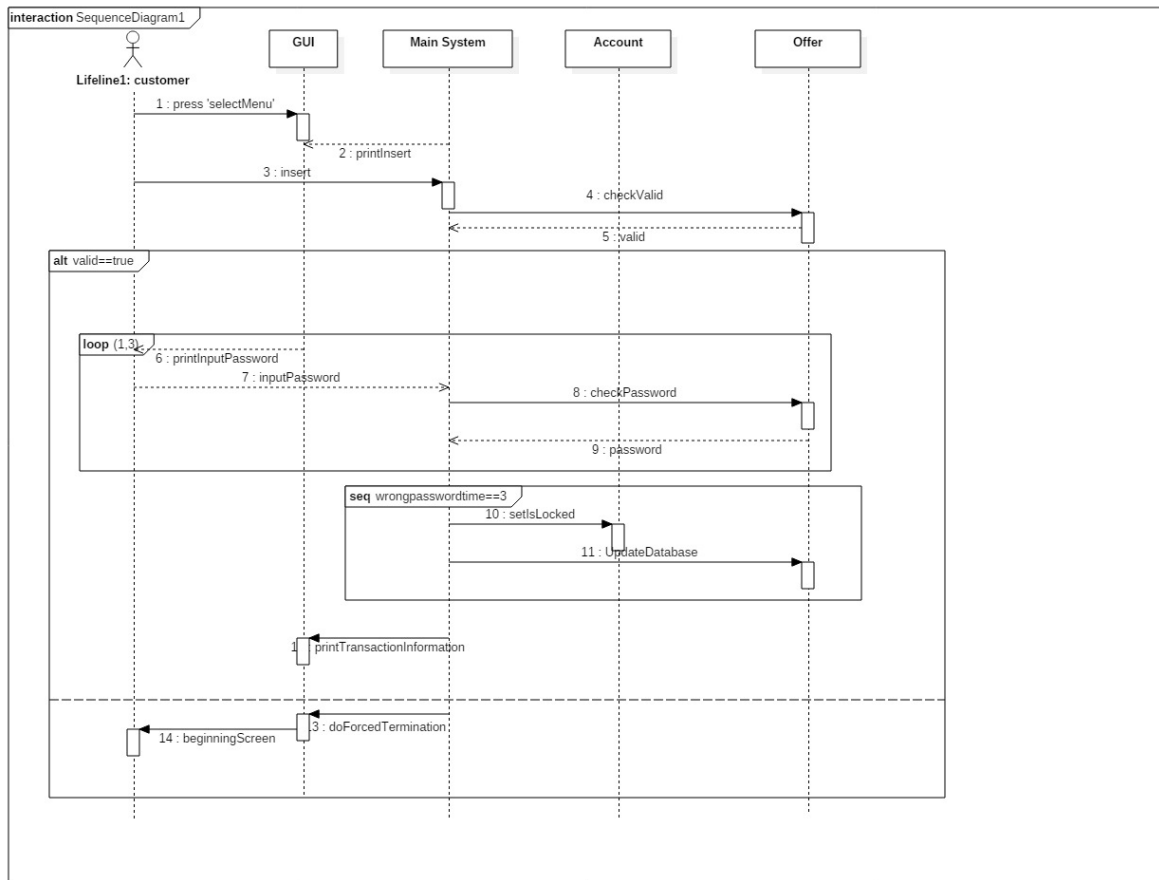
Type	Method
Name	updateDatabase(Account account)

Purpose	Database를 업데이트한다
Cross Reference	Usecase: 17 Functional: R6.2
Input(Method)	Account account
Output(Method)	Int
Abstract operation(Method)	Database를 account객체의 정보와 같게 저장하고, error code를 int로 반환한다.
Exceptional Courses of Events	-

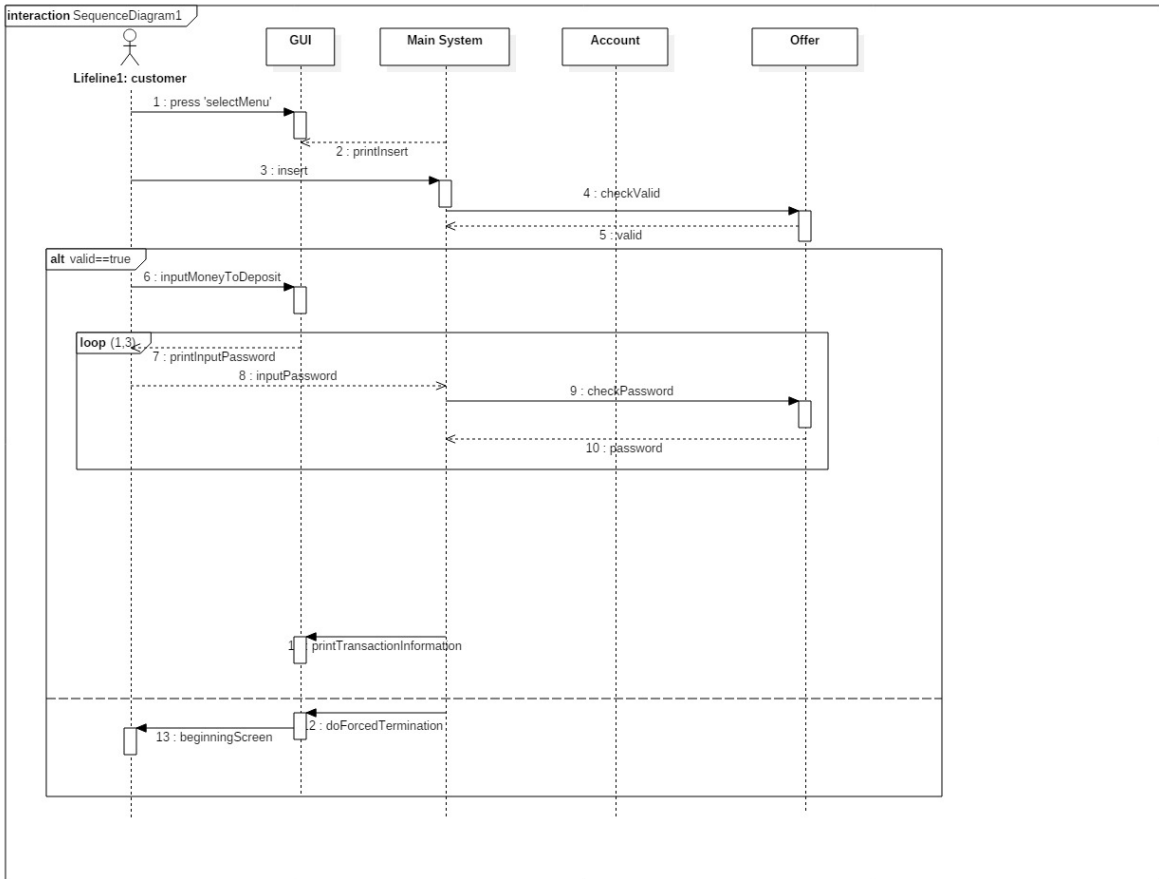
Type	Method
Name	checkValid(Account account)
Purpose	Account의 유효성을 체크하고 database의 내용을 Account에 저장한다.
Cross Reference	Usecase: 16 Functional: R6.1
Input(Method)	Account account
Output(Method)	Int
Abstract operation(Method)	account객체를 database의 정보와 같게 저장하고, error code를 int로 반환한다.
Exceptional Courses of Events	-

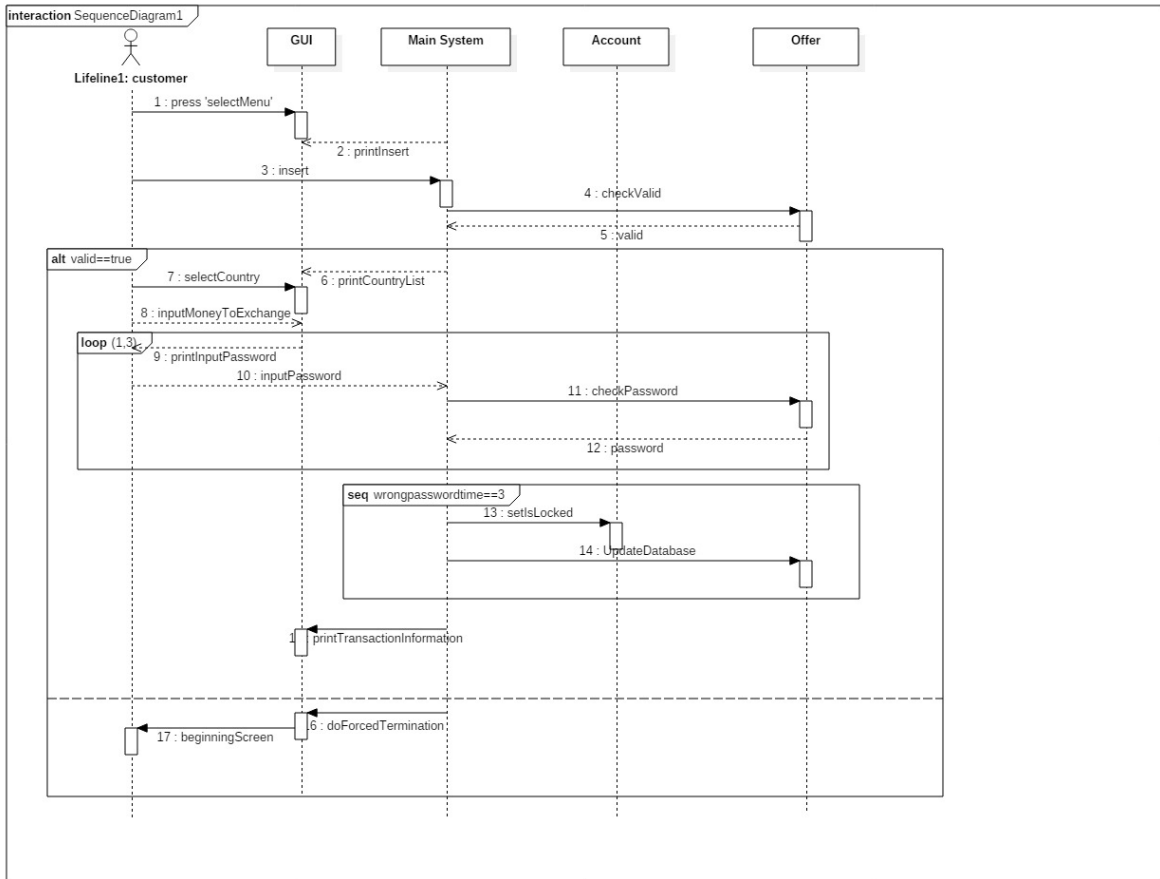
Type	Method
Name	readDatabase(Account account)
Purpose	Database(txt)파일의 정보를 읽어온다.
Cross Reference	Usecase: 17 Functional: R6.2
Input(Method)	Account account
Output(Method)	Int
Abstract operation(Method)	Database(txt)파일의 정보를 읽어오고 error code를 int로 반환한다.
Exceptional Courses of Events	-

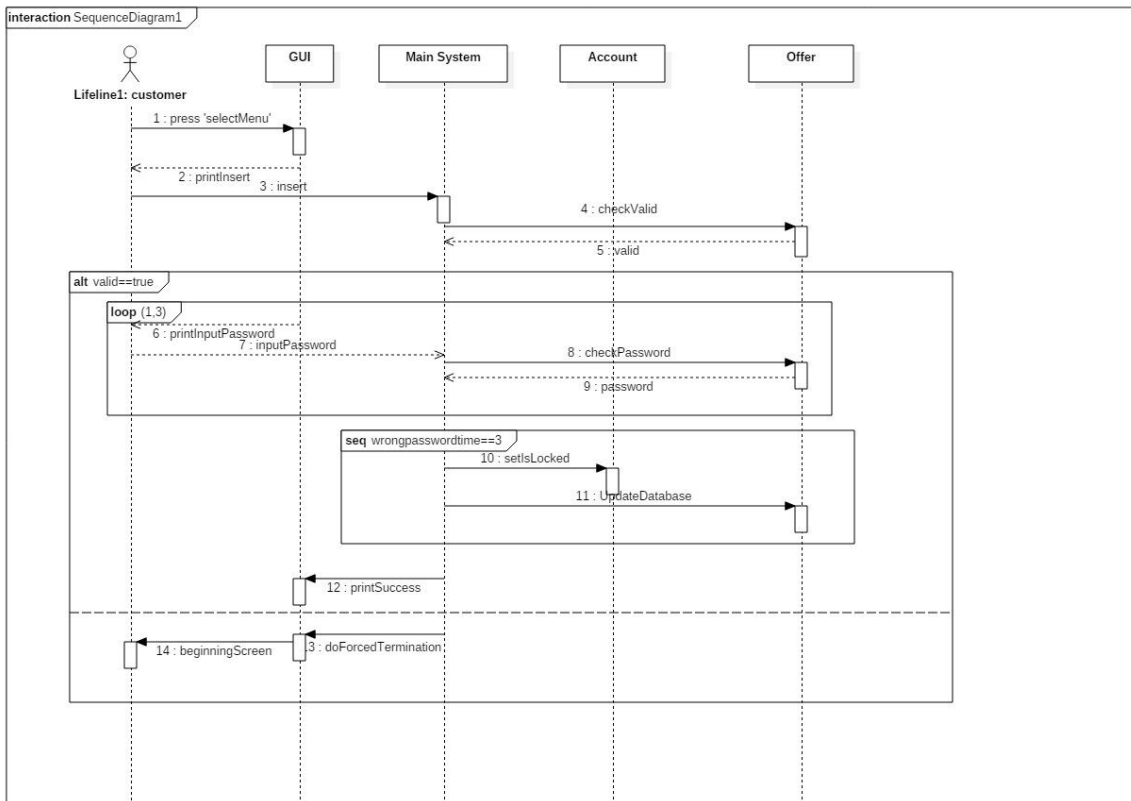
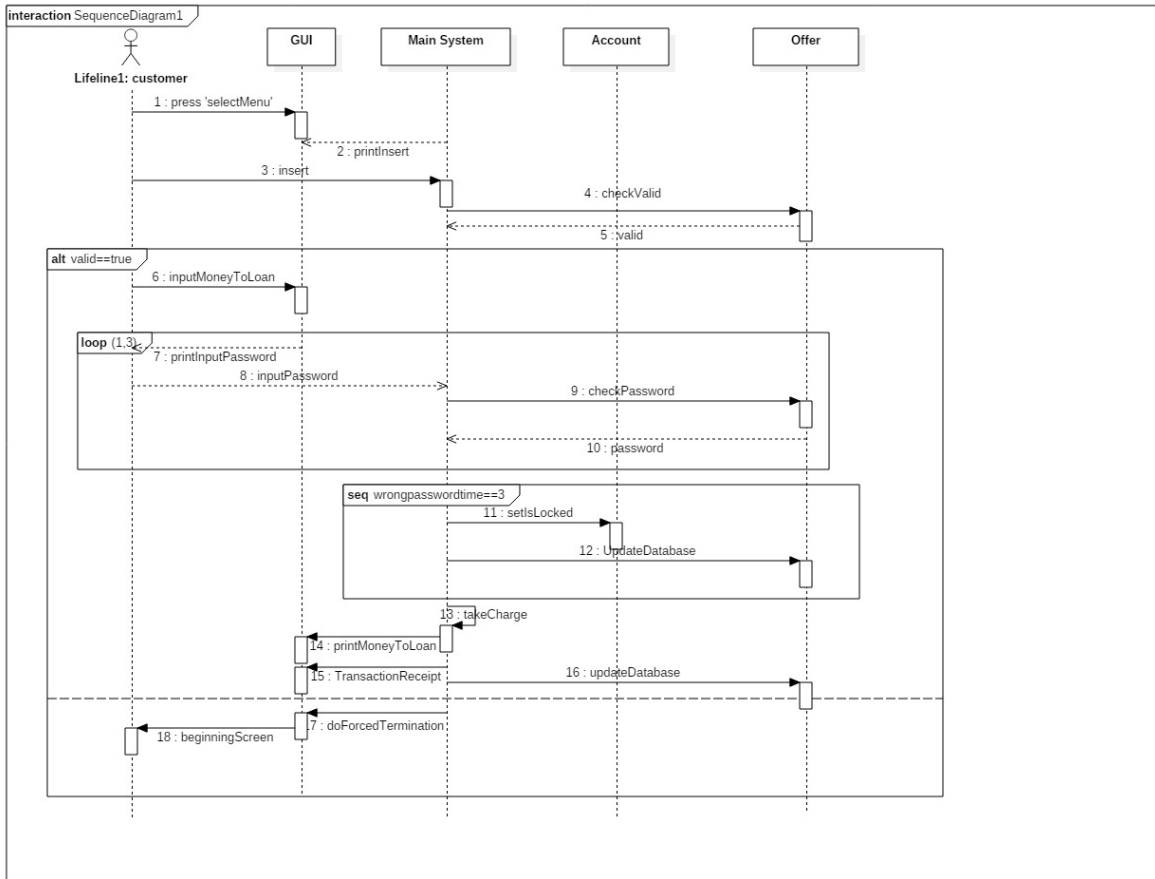
Activity 2052. Implements Windows



Name	selectMenu
Responsibilities	메뉴에서 원하는 목적의 버튼을 누른다
Type	GUI
Cross Reference	R1.1,R1.2,R1.3,R1.4,R1.5,R1.6,R1.7,R1.8
Notes	버튼을 누른다
Pre-Conditions	
Post-Conditions	다음 화면으로 이동한다.







Post-Conditions	다음 화면으로 이동한다.
-----------------	---------------

Activity 2055. Write Unit Test Code

(현재까지 구현된 부분까지 테스트했습니다)

1. Check Valid Test

```
@Test
public void testCheckValid() {
    Offer test = new Offer(0);
    Account account = new Account();
    account.setBank("국민은행");
    account.setAccountNumber("23456789");
    assertEquals(true, test.checkValid(account));
}
```

2. Update Database Test

```
@Test
public void testUpdateDatabase() {
    Offer test = new Offer(0);
    Account testaccount = new Account();
    testaccount.setBank("국민은행");
    testaccount.setAccountNumber("23456789");
    testaccount.setBalance(222080);
    testaccount.setIsLocked(false);
    testaccount.setName("권성환");
    testaccount.setPassword(4752);
    testaccount.setRate(3);
    testaccount.setLog("");
    assertEquals(true, test.updateDatabase(testaccount));
}
```

3. Read Database Test

```
@Test
public void testReadDatabase() {
    Account testaccount = new Account();
    testaccount.setBank("국민은행");
    testaccount.setAccountNumber("23456789");
    testaccount.setBalance(222080);
    testaccount.setIsLocked(false);
    testaccount.setName("권성환");
    testaccount.setPassword(4752);
    testaccount.setRate(3);
    Account account = new Account();
    Offer test = new Offer(0);
    account.setBank("국민은행");
    account.setAccountNumber("23456789");
    test.readDatabase(account);
    assertEquals(testaccount.getBalance(), account.getBalance());
}
```

2060 Testing

(현재까지 구현된 곳까지 진행했습니다)

Activity 2061. Unit Testing

```

1 package Junit;
2
3 import static org.junit.Assert.*;
9
10 public class Filetest {
11
12     @Test
13     public void testCheckValid() {
14         Offer test = new Offer(0);
15         Account account = new Account();
16         account.setBank("국민은행");
17         account.setAccountNumber("23456789");
18         assertEquals(true, test.checkValid(account));
19     }
20
21     @Test
22     public void testUpdateDatabase() {
23         Offer test = new Offer(0);
24         Account testaccount = new Account();
25         testaccount.setBank("국민은행");
26         testaccount.setAccountNumber("23456789");
27         testaccount.setBalance(222000);
28         testaccount.setIsLocked(false);
29         testaccount.setName("권성원");
30         testaccount.setPassword(4752);
31         testaccount.setRate(3);
32         testaccount.setLog("");
33         assertEquals(true, test.updateDatabase(testaccount));
34     }
35
36     @Test
37     public void testReadDatabase() {
38         Account testaccount = new Account();
39         testaccount.setBank("국민은행");
40         testaccount.setAccountNumber("23456789");
41         testaccount.setBalance(222000);
42         testaccount.setIsLocked(false);
43         testaccount.setName("권성원");
44         testaccount.setPassword(4752);
45         testaccount.setRate(3);
46         Account account = new Account();
47         Offer test = new Offer(0);
48         account.setBank("국민은행");
49         account.setAccountNumber("23456789");
50         test.readDatabase(account);
51         assertEquals(testaccount.getBalance(), account.getBalance());
52     }
53
54 }
55

```

Activity 2062. System Testing

No.	Test 항목	Description	Use Case	System Function
1	Deposit 기능	Deposit을 선택하고 과정을 진행했을 때 제대로 과정이 진행되는지 확인한다	Deposit	R1.1
2	Withdraw 기능	Withdraw를 선택하고 과정을 진행했을 때 제대로 과정이 진행되는지 확인한다	Withdraw	R1.3
3	Insert 기능	각 기능을 선택하고 거래 method가 잘 insert되는지 확인한다	Insert	R2.1
4	Print Error 기능	잘못된 정보를 입력할 경우 오류 메시지가 출력되는지 확인한다	Print Error	R2.3
4-1	Print Error-잔액 부족	출금 시 잔액이 부족할 경우 확인 가능한지 확인한다	Print Error	R2.3
5	Check Password	패스워드를 잘못 입력했을 경우 제대	Check Password	R4.1

	기능	로 오류 메시지를 출력하는지 확인한다		
6	Check Password 기능	패스워드를 제대로 입력했을 경우 계좌에 제대로 access되는지 확인한다	Check Password	R4.1
7	Forced Termination 기능	잘못된 정보를 3회보다 더 많이 입력했을 경우 거래를 즉시 종료하는지 확인한다	Do Forced Termination	R2.4
8	Transaction Lock 기능	비밀번호를 3회보다 더 많이 잘못 입력했을 경우 계좌가 잠기는지 확인한다	Transaction Lock	R5.1
8-1	Transaction Lock 기능	이미 잠긴 계좌의 경우 거래가 불가능한지 확인한다	Transaction Lock	R5.1
9	Update DB 기능	거래가 종료된 이후 DB가 제대로 update되는지 확인한다	Update Database	R6.2
10	Check Validation 기능	거래 method가 거래 가능한지 불가능한지 확인한다	Check Validation	R6.1
11	Print Transaction Receipt 기능	거래가 완료된 이후 거래 명세서가 출력되는지 확인한다	Print Transaction Receipt	R2.2
12	Check Balance 기능	거래 내역을 출력할 수 있는지 확인한다	Check Balance	R1.8