Packages

```
src/main/java
      ie.michaelbank
      ie.michaelbank.dao
      ie.michaelbank.mappers
      ie.michaelbank.services
```

Dependcies in POM.XML

```
 <properties>

            <!-- Generic properties -->
            <java.version>1.8</java.version>
            <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
            <project.reporting.outputEncoding>UTF-
8</project.reporting.outputEncoding>

            <!-- Spring -->
            <spring-framework.version>5.2.0.RELEASE</spring-framework.version>

            <!-- Hibernate / JPA -->
            <hibernate.version>4.2.1.Final</hibernate.version>

            <!-- Logging -->
            <logback.version>1.0.13</logback.version>
            <slf4j.version>1.7.5</slf4j.version>

            <!-- Test -->
            <junit.version>4.11</junit.version>

      </properties>


<!-- Spring and Transactions -->
            <dependency>
                  <groupId>org.springframework</groupId>
                  <artifactId>spring-context</artifactId>
                  <version>${spring-framework.version}</version>
            </dependency>
            <dependency>
                  <groupId>org.springframework</groupId>
                  <artifactId>spring-tx</artifactId>
                  <version>${spring-framework.version}</version>
            </dependency>
            <!-- JDBC -->
            <dependency>
                <groupId>org.springframework</groupId>
                <artifactId>spring-jdbc</artifactId>
                <version>${spring-framework.version}</version>
            </dependency>

            <!-- https://mvnrepository.com/artifact/com.h2database/h2 -->
            <dependency>
                <groupId>com.h2database</groupId>
                <artifactId>h2</artifactId>
                <version>1.4.199</version>
                <!-- <scope>test</scope> -->
            </dependency>
```

Nice feature which showing queries

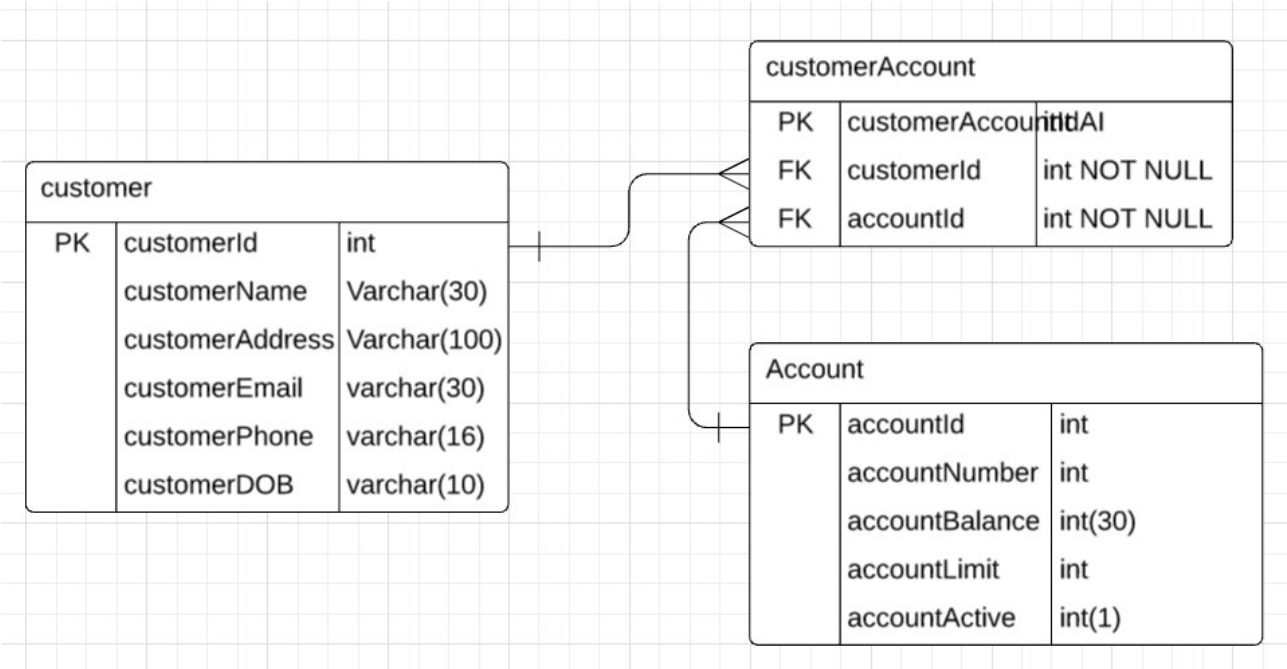
```
		<!-- Logging with SLF4J & LogBack -->
	<!--		<dependency>
			<groupId>org.slf4j</groupId>
			<artifactId>slf4j-api</artifactId>
			<version>${slf4j.version}</version>
			<scope>compile</scope>
		</dependency>
		<dependency>
			<groupId>ch.qos.logback</groupId>
			<artifactId>logback-classic</artifactId>
			<version>${logback.version}</version>
			<scope>runtime</scope>
		</dependency>-->
```

Setting Beans

```
src/main/resources
	beans.xml
```

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
	xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
	xmlns:context="http://www.springframework.org/schema/context"
	xmlns:jdbc="http://www.springframework.org/schema/jdbc"
	xmlns:c="http://www.springframework.org/schema/c"
	xmlns:p="http://www.springframework.org/schema/p"
	xsi:schemaLocation="http://www.springframework.org/schema/jdbc
http://www.springframework.org/schema/jdbc/spring-jdbc-4.3.xsd
		http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans.xsd
		http://www.springframework.org/schema/context
http://www.springframework.org/schema/context/spring-context-4.3.xsd">

	<context:component-scan base-package="ie.michaelbank"></context:component-
scan>

	<jdbc:embedded-database id="dataSource" type="H2">
		<jdbc:script location="schema.sql"/>
		<jdbc:script location="data.sql"/>
	</jdbc:embedded-database>

	<bean id="jdbcTemplate"
class="org.springframework.jdbc.core.JdbcTemplate" autowire="byType"/>

</beans>
```

Database Entity Realitionship



Creating tables

```sql
CREATE TABLE customer (
    customerId int(11) NOT NULL AUTO_INCREMENT,
    customerName varchar(30) NOT NULL,
    customerAddress varchar(100) NOT NULL,
    customerEmail varchar(50) NOT NULL,
    customerPhone varchar(16) NOT NULL,
    customerDOB varchar(10) NOT NULL,
    PRIMARY KEY(customerId)
);

CREATE table  account (
    accountId int(11) NOT NULL AUTO_INCREMENT,
    accountNumber int(11) AUTO_INCREMENT,
    accountBalance int(30) default 0,
    accountLimit int(11) NOT NULL default 500,
    accountActive int(1) NOT NULL default 1,
    PRIMARY KEY (accountID)
);


CREATE TABLE customerAccount (
    customerAccountId int(11) NOT NULL AUTO_INCREMENT  PRIMARY KEY,
    customerID      int(11) NOT NULL,
    accountID       int(11) NOT NULL,
    FOREIGN KEY (customerId) REFERENCES customer (customerId),
    FOREIGN KEY (accountId) REFERENCES account (accountId)
);
```

```sql
INSERT INTO customer(customerName, customerAddress, customerEmail,customerPhone, customerDOB) VALUES
                  ('John Doe', '68 Bovoyage Bulvar Dublin','John.doe@email.com'     ,'0012003456765','01-01-2000'),
                  ('Peter White', '24 Chedar Avenue Dublin','Peter.white@email.com','0012098345098','24-03-1960'),
                  ('Jane Silver', '123 Green Street Dublin','Jane.silver@email.com','0012083459384','06-08-1980');

INSERT INTO account(accountNumber, accountLimit,accountBalance) VALUES
                                          (100001, 500,  110000);
INSERT INTO account(accountLimit,accountBalance) VALUES (500, 1100000),
                                                        (600,  379800);

INSERT INTO customerAccount(customerId, accountId) VALUES (1,1), (2,2), (3,3);
```

Criteria for quries

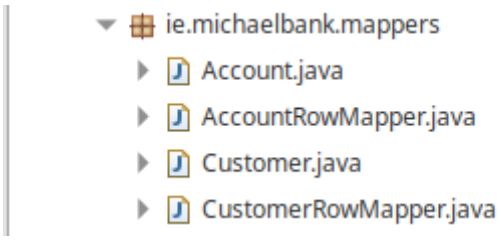
1 Register new custmer become a bank customer i.e. register with the bank
Enter name:
Enter address:
Enter DOB:

```
      Enter email:
      Enter phone:

            int saveAndReturnCustomerId(String customerName, String
customerAddress,String customerEmail, String customerPhone, String customerDOB);

 2 Create a new account + register customer
      Enter the overdraft (debit) limit:
            int saveAndReturnAccountId(int accountLimit);

      Join customer to the account
      J add account to customer
            Enter the phone number or D-O-B:
            Customer getCustomerFromDOBorPhone(String customerDOBorPhone);


      N create new customer
            int saveAndReturnCustomerId(String customerName, String
customerAddress,String customerEmail, String customerPhone, String customerDOB);

      addAccountToCustomer(int accountId, int customerId);


 3 Add person to an account (for joint accounts)
      --list existing account and name

      Enter Phone or DOB:
            int getCustomerIdFromPhone(String customerPhone);
      Enter account no:
            int getAccountIdFromAccountNumber(int accountNumber);


      boolean addCustomerToAcount(int customerId, int accountId);


 4 view customers acounts - view his/her own accounts
      Enter the acoount number

      Account findById(int accountId);
      Account findByAccountNumber(int accountNumber);

 5 withdraw money from his/her account, subject to an overdraft limit
      Enter the account number:
      Enter the ammount of withdraw:
      Account findByAccountNumber(int accountNumber);
      int updeteBalanceWithAccountNumber(int accountNumber);


 6 Deposit money into his/her account
      Enter the account number:
      Enter the deposit:
      int updeteBalanceWithAccountNumber(int accountNumber);

 7 Transfer money from one account to another
      Enter the account number you transfer FROM:
      Enter the account number you transfer TO:
      Enter the ammount:
      int transferFromAccountToAccount(int accountNumberFrom,
                              int accountNumberTo, int ammount);
```

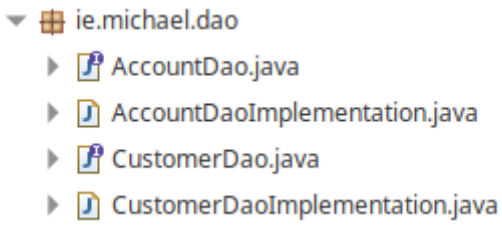
```
 8 Close an account
      Enter the acoount number you like to close:
      boolean udateCloseAccountWithAccountNumber(int accountNumber);

 9 Total amount of money deposited in the bank
      int getAllAmountInBank();
   The number of accounts with deposits over €10,000
      int findCountOfAccountsOver10000();
 0 EXIT the app


extra
      List<Account> findAllAccountsWithCustomer();
      List<Customer> findAllCustomersWithAccount();
```

The above text is color coded to easily find I which object it will be placed in the code.

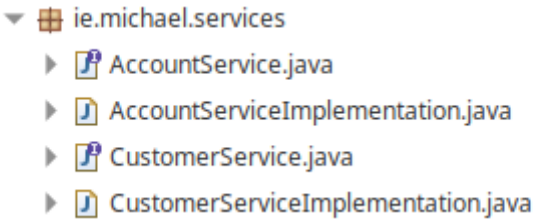
Creating mappers layer



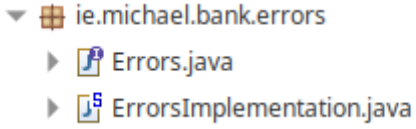
Creating dao layer



Creating Service layer



Creating Erros layer



Bean Errors

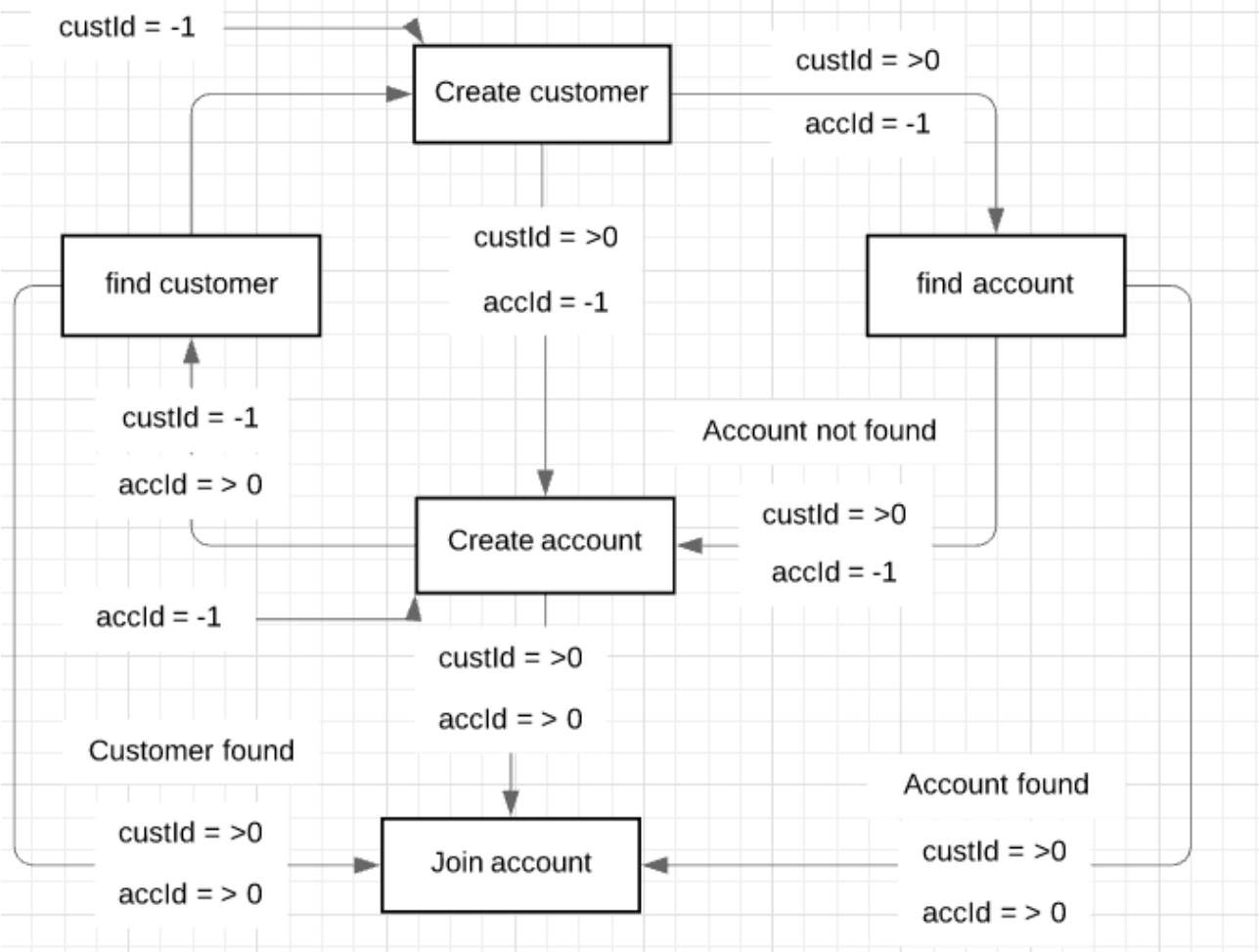
All error messages are in beans.xml

```
<bean class="ie.michael.bank.errors.ErrorsImplementation"
    id="customerNotFound" c:errorName="**** CUSTOMER NOT FOUND ****">
</bean>
```

Bean Error implementation

```
Errors error;
Customer cust = customerService.getACustomerByItsPhoneorDOB("06-08-198");
if(cust == null) {
    error = (Errors) context.getBean("customerNotFound");
    System.out.println(error.toString());
}
```

Create Customer, Create Account, Join Customer with Account (Existing to New, multiple Accounts)



The diagram above showing functionality. To join a customer with an account, we have to obtain both customer id and account id. If one of these ids doesn't exist, the program prompts to create it.

However, if one of those ids exist we can create a customer with multiple accounts or a single account with multiple users.

Testing APP

Main Menu

```
 SELECT A NUMBER
 [1] Register new custmer become a bank customer i.e. register with the bank
 [2] Create a new account
 [3] Add person to an account (for joint accounts)
 [4] view customers acounts - view his/her own accounts

 [5] withdraw money from his/her account, subject to an overdraft limit
 [6] Deposit money into his/her account
 [7] Dransfer money from one account to another
 [8] Close an account

 [9] Total amount of money deposited in the bank
     The number of accounts with deposits over €10,000
 [0] EXIT the app

ENTER THE NUMBER:
```

1 Register new customer

```
ENTER THE NUMBER: 1
REGISTER NEW CUSTOMER
Enter the NAME (max 30):John Silver
Enter the ADDRESS (max 100):467 Pirate Bay, California
Enter the DOB (DD-MM-YYYY): 01-01-1000
Enter the EMAIL (max 30):n/a
Enter the Phone (9-16 digits):123456789

You ENTERED:
Name   : John Silver
Address: 467 Pirate Bay, California
DOB    : 01-01-1000
Email  : n/a
Phone  : 123456789

Press ENTER to SAVE
Start over? (Y) Exit without save (E)
```

Firstly, the form must be filled, can not be blank or longer than stated.
The phone number and DOB must be filled correctly, otherwise, it can't be saved and the error message is thrown.

```
**** YOUR ENTRY CONTAINS ERRORS ****
Start over? (Y) Exit without save (E)
```

User has two options E – Exit or Y (or any character) – start over

Creating Account

```
CREATE NEW ACCOUNT
Enter the deposit     :1000
Enter overdraft limit :100
false

You ENTERED:
Deposit        : 1000
Overdraft limit: 100

Press ENTER to SAVE
Start over? (Y)
```

1 register NEW customer + NEW account
- **create customer**
- **create account**
- **join together**
- **display**

```
=================================================================

(4) ACCOUNT NUMBER: 100004    BALANCE: 1000.0    overdraft: 1000    [ACTIVE]
    (5) NAME: bob sdfsfd    ADDRESS: sdfsfd    EMAIL: sdfsfd    PHONE: 987654321    DOB: 03-23-2000

=================================================================

(5) NAME: bob sdfsfd    ADDRESS: sdfsfd    EMAIL: sdfsfd    PHONE: 987654321    DOB: 03-23-2000
    (4) ACCOUNT NUMBER: 100004    BALANCE: 1000.0    overdraft: 1000    [ACTIVE]
```

2 create NEW account EXISTING customer
- **create new account**
- **find customer**

  The account can be found by account number, if prompted wrong it can be found by phone or DOB (in case of DOB multiple customers may be returned) we can specify the right customer by customer-id

- **add account to customer**
- **display (1 customer has multiple accounts)**

```
(2) NAME: Peter White    ADDRESS: 24 Chedar Avenue Dublin    EMAIL: Peter.white@email.com    PHONE: 0012098345098    DOB: 24-03-1960
    (2) ACCOUNT NUMBER: 100002    BALANCE: 11000.0    overdraft: 500    [ACTIVE]
    (4) ACCOUNT NUMBER: 100004    BALANCE: 100.0    overdraft: 100    [ACTIVE]
    (5) ACCOUNT NUMBER: 100005    BALANCE: 10.0    overdraft: 1000    [ACTIVE]
******JOINED*****
```

3 add customer to account (join account)

Join accoun
create customer
find account
join together
display (1 account has multiple customers)

```
(2) ACCOUNT NUMBER: 100002    BALANCE: 11000.0    overdraft: 500    [ACTIVE]
    (2) NAME: Peter White    ADDRESS: 24 Chedar Avenue Dublin    EMAIL: Peter.white@email.com    PHONE: 0012098345098    DOB: 24-03-1960
    (4) NAME: asdad    ADDRESS: asdads    EMAIL: sdfsdf    PHONE: 1232456789    DOB: 01-01-1000
```

Because of connected methods, we will always get an option to create an account if the account number doesn't exist, creating customers if we can't find account number, phone number or DOB.

```
***SAVED***JOIN CUSTOMER WITH ACCOUNT
Create New account?                              (N)
Add Customer to existing account? (Join Account) (A)a
Enter the account number:100002
******JOINED*****
```

Function "Find"

```
JOIN ACCOUNT WITH CUSTOMER (customer with more accounts)
   Register New User?  (R)
   FIND existing User? (F)
f
Find CUstomers by account number
Enter the account number
100004
Find Customers by Phone or D-O-B
Enter the phone number or DOB (DD-MM-YYYY):
```

Firstly, the function is asking for an account number and an ID is returned. However, if the account number is not found, the -1 triggers next search.

When a phone or DOB search is used, there could be more than one user with the same DOB.
In that case, multiple customers are listed and "accountant" has to choose the right ID.
If even then is not right ID selected the Find method is terminated and returning to the main menu.

4 View account

```
ENTER THE NUMBER: 4
Enter the account number:100002
(2) ACCOUNT NUMBER: 100002    BALANCE: 11000.0    overdraft: 500    [ACTIVE]
Press enter to conttinue...
```

5 Withdraw from account

```
ENTER THE NUMBER: 5
WIDTHDRAW MONEY FROM AN ACCOUNT
Enter the account number:10003
***ACCOUNT NOT FOUND***
```

wrong account number

```
ENTER THE NUMBER: 5
WIDTHDRAW MONEY FROM AN ACCOUNT
Enter the account number:100003
(3) ACCOUNT NUMBER: 100003    BALANCE: 3798.0    overdraft: 600    [ACTIVE]
Amount you like to widthdraw (maximum 4398.0): 5000
Insuficience founds
```

debiting too much

```
ENTER THE NUMBER: 5
WIDTHDRAW MONEY FROM AN ACCOUNT
Enter the account number:100003
(3) ACCOUNT NUMBER: 100003    BALANCE: 3798.0    overdraft: 600    [ACTIVE]
Amount you like to widthdraw (maximum 4398.0): 500
Will be debeted:500.0
Are you sure? (Y)

***TRANSACTION CANCELED***
```

withdraw

```
ENTER THE NUMBER: 5
WIDTHDRAW MONEY FROM AN ACCOUNT
Enter the account number:100003
(3) ACCOUNT NUMBER: 100003    BALANCE: 3798.0    overdraft: 600    [ACTIVE]
Amount you like to widthdraw (maximum 4398.0): 500
Will be debeted:500.0
Are you sure? (Y)
y
***ACCOUNT DEBETED***
```

The account must be ACTIVE otherwise the transaction can not be done

Checking by viewing account

```
ENTER THE NUMBER: 4
Enter the account number:100003
(3) ACCOUNT NUMBER: 100003    BALANCE: 3298.0    overdraft: 600    [ACTIVE]
Press enter to conttinue...
```

6 Deposit Money

```
ENTER THE NUMBER: 6
DEPOSIT MONEY TO ACCOUNTEnter the account number:100001
(1) ACCOUNT NUMBER: 100001    BALANCE: 1100.0    overdraft: 500    [ACTIVE]
Amount you like to deposit: 2000
Will be lodged:2000.0
Are you sure? (Y)
y
***ACCOUNT LODGED***
```

Viewing account to confirm the transaction

```
ENTER THE NUMBER: 4
Enter the account number:100001
(1) ACCOUNT NUMBER: 100001    BALANCE: 3100.0    overdraft: 500    [ACTIVE]
Press enter to conttinue...
```

The account must be ACTIVE otherwise the transaction can not be done

7 Transfer Money

```
ENTER THE NUMBER: 7
TRANSFER MONEY
Account number you transferint FROM
Enter the account number:100001
FROM:(1) ACCOUNT NUMBER: 100001    BALANCE: 1100.0    overdraft: 500    [ACTIVE]
Enter the account number:100002
Account number you transferint TO
TO  :(2) ACCOUNT NUMBER: 100002    BALANCE: 11000.0    overdraft: 500    [ACTIVE]
Amount You Like To transfer (maximum 1600.0): 600
FROM account: 100001 Will be transfered:600.0 TO:100002
Are you sure? (Y)
y
***TRANSFER COMPLETED***
```

both accounts must set correctly, on the debited account must be sufficient funds and both accounts must be ACTIVE

checking if correct

```
ENTER THE NUMBER: 4
Enter the account number:100001
(1) ACCOUNT NUMBER: 100001    BALANCE: 500.0    overdraft: 500    [ACTIVE]
Press enter to conttinue...
```

```
ENTER THE NUMBER: 4
Enter the account number:100002
(2) ACCOUNT NUMBER: 100002    BALANCE: 11600.0    overdraft: 500    [ACTIVE]
Press enter to conttinue...
```

8 Closing account

```
ENTER THE NUMBER: 8
Enter the account number:100002
(2) ACCOUNT NUMBER: 100002    BALANCE: 11600    overdraft: 500    [ACTIVE]
really close this account? (Y)y
***ACCOUNT CLOSED***
Press enter to back into the main menu...
```

checking if an account is closed

```
ENTER THE NUMBER: 4
Enter the account number:100002
(2) ACCOUNT NUMBER: 100002    BALANCE: 11600    overdraft: 500    [CLOSED]
Press enter to continue...
```

With closed account cannot be manipulated! Crediting or debeting this account cannot be done!

```
ENTER THE NUMBER: 5
WIDTHDRAW MONEY FROM AN ACCOUNT
Enter the account number:100002
(2) ACCOUNT NUMBER: 100002    BALANCE: 11600.0    overdraft: 500    [CLOSED]
***ACCOUNT CLOSED***
```

9 Total amount of money deposited in the bank
The number of accounts with deposits over €10,000

```
BANK INFO
Total amount of money deposited in the bank is €599.800000001
The number of accounts with deposits over €10,000 is: 0
```

```
BANK INFO
Total amount of money deposited in the bank is €599.8
The number of accounts with deposits over €10,000 is: 0
```

The numbers are stored in the database as a BIGINT that's mean I am multiplying and dividing by 100 to avoid floating point issue.

Also, I think that fraction numbers shouldn’t be used for storing money

jUnit test

from console, when test is runt:

```
TEST 1
creating accout and crediting 0.04 (floating point error 0.04000000000000001)

(4) ACCOUNT NUMBER: 100004    BALANCE: 0.04    overdraft: 50    [ACTIVE]

Thus balance must be also 0.04
```

```
TEST 2
creating two new accounts and obtaining its numbers:
100004
100005
The account numbers cannot be equal
```

```
TEST 3
from given account No: 100002

(2) ACCOUNT NUMBER: 100002    BALANCE: 11000.0    overdraft: 500    [ACTIVE]

trying to widthdraw 11501.0 which is 1 more then balance+limit
receiving status "0" which means insufficient founds

When account No: 100002 balance is checked after, it should be unchanged

(2) ACCOUNT NUMBER: 100002    BALANCE: 11000.0    overdraft: 500    [ACTIVE]
```

```
TEST 4
Closing acount No: 100002

(2) ACCOUNT NUMBER: 100002    BALANCE: 11000.0    overdraft: 500    [CLOSED]

trying to widthdraw 1.0
receiving status "-1" which means account is closed

When account No: 100002 balance is checked after, it should be unchanged

(2) ACCOUNT NUMBER: 100002    BALANCE: 11000.0    overdraft: 500    [CLOSED]
```

```
TEST 5
Closing acount No: 100001

(1) ACCOUNT NUMBER: 100001    BALANCE: 1100.0    overdraft: 500    [CLOSED]

trying to lodge 1.0
receiving status "false" which means account is closed

When account No: 100001 balance is checked after, it should be unchanged

(1) ACCOUNT NUMBER: 100001    BALANCE: 1100.0    overdraft: 500    [CLOSED]
```

Mockito

```
TEST 6
An account should be set just once
```