Java Bank Application

Created by: Bar Luca-Narcis & Cornea Mihai, Group 30421

**Table of Contents:**

**1.Brief Description**

**2. Use-Cases**

**3. Solution Presentation**

**4.Conclusion and further improvements**

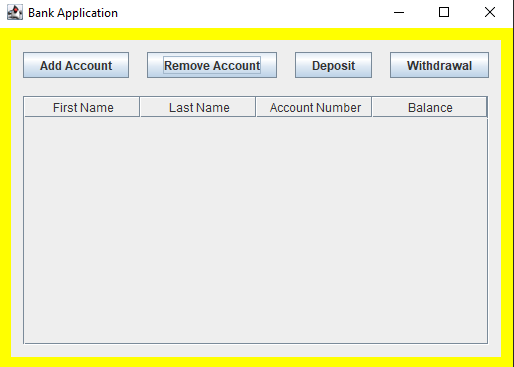
*1.Brief Description*

We have created a (pretty simple) bank application, which allows the user to add an account (by entering the name, CNP and the initial deposit), delete an account and withdraw or deposit a certain amount of money(in Lei). It has a simple and intuitive GUI to allow the user to interact with all of its different functions.

*2.Use-Cases*

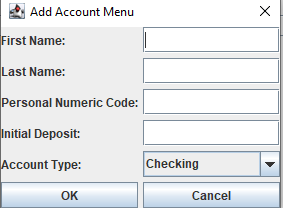
1. **Main Menu:**

Basically the MainMenu function is the main function of our project. In it, we initialise the menu JFrame and all the corresponding buttons that will launch the other interfaces of the application.

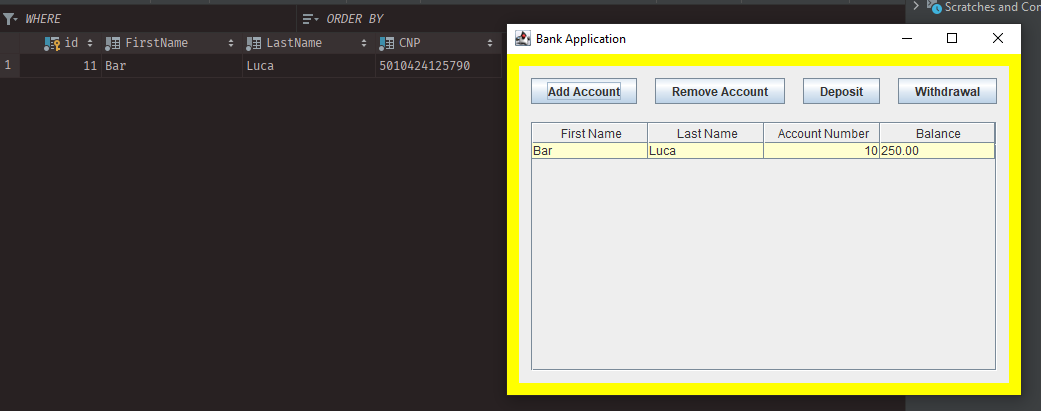
****

From here we can select between adding an account, removing an account, depositing and withdrawing. The last 2 options can be done only if an account is present and is selected.

**B) Add Account**



User presses Add Account and is prompted with this window in which he adds his/her first and last name ,the CNP ,the initial deposit and selects the type of account, then presses confirm,after which it is added in the table and database(if CNP has 13 digits and the initial deposit is at least 100 for Checking and 200 for Savings. Afterwards ,the menu is refreshed to display the newly added account.

****

Here we can see that we have successfully added an account.

**C)Remove Account:**

When User selects the desired account and the presses the Remove Account button, the selected account will be removed from both the table and database. The Main Menu table will also be refreshed.

1. **Deposit**

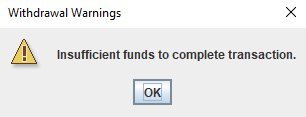


User selects the account then it is prompted to insert the desired amount. Then a confirmation windows as the one above is displayed. Upon pressing the “Yes” button, the window closes and the balance is updated in both the table and database ,and the table is refreshed, displaying the new balance.

**E)Withdraw**

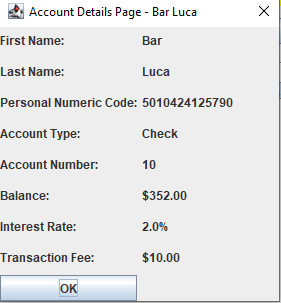


When the user presses the Withdraw button, it is prompted to enter an amount and then a confirmation window is displayed once again After confirmation, the table and database are updated and the table is refreshed, displaying the new balance. But here, if the desired amount+transaction fee is greater than current balance, an error is displayed as below.

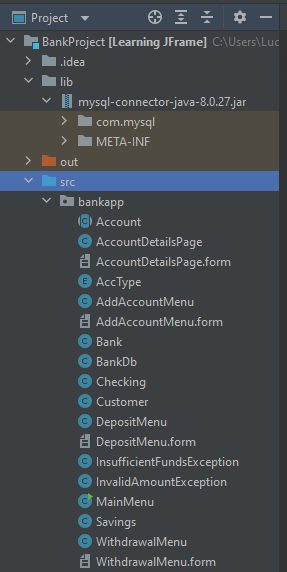


**F)Account Details**

When user double-clicks one of the account,I t is prompted with another window which displays the selected account’s details.

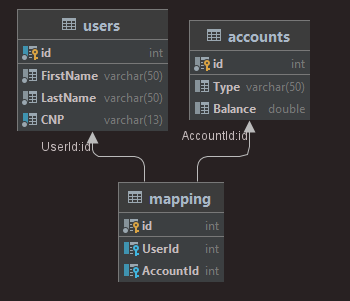


3.Solution Presentation



The Working behind the project is quite simple. Our main class is the main menu, which has a GUI implemented. Depending on the type of operation selected, the user is shown the corresponding menu: AccountDetailsPage, AddAccountMenu, DepositMenu or WithdrawalMenu, which are all part of the GUI. All of them are shown by using ActionLister’s ActionPerformed method. The Bank class takes care of all the operations (adding or deleting accounts, depositing or withdrawing) and by using the database in the BankDb class, sends the variables needed for the specific sql queries(e.g. the details of the newly added accounts, the id of the account to be deleted, the id and the sum needed for withdrawal of depositing). The BankDb class also has methods that return the specific details of the accounts. AccType is an Enum which displays the different types of account(Undef,Checking or Savings). And the corresponding Checking and Savings classes extend the abstract class Account and implement its getAccountType method to return the type corresponding to them.

For our application we have chosen to implement the database in MySQL with the following tables :

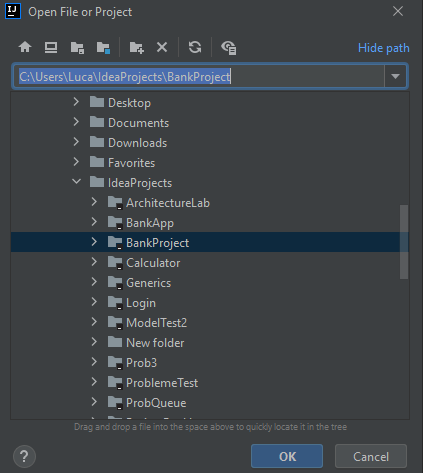


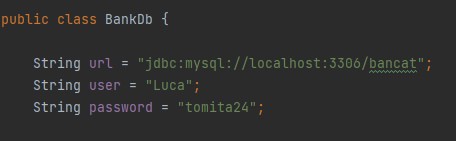
We’ve also implemented two new Exception classes in case an invalid number is inserted(for deposit or withdrawal) or the funds are insufficient( in case of withdrawal).

How do we set up the Project?

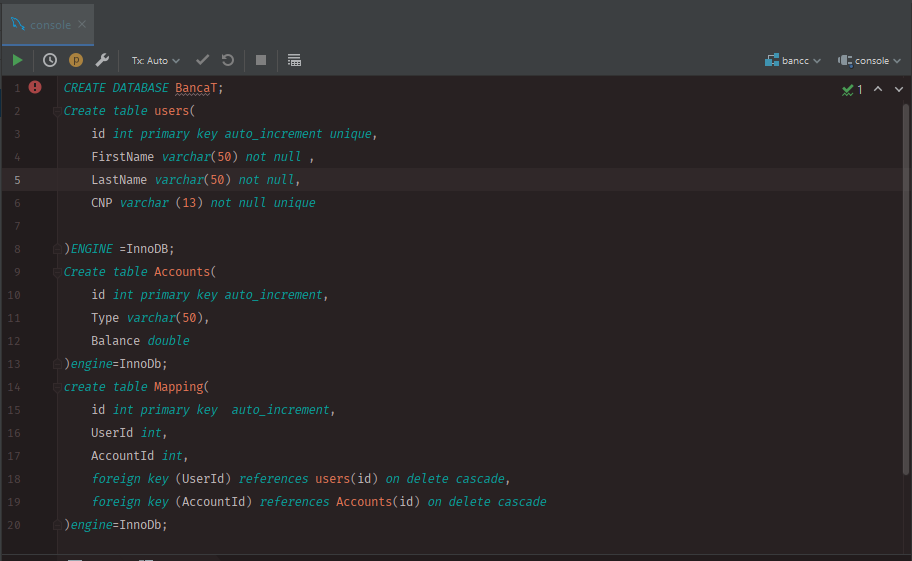
Well, it’s quite easy. All you need to do is:

1.open the folder as a project in your IDE of choice(IntelliJ IDEA is recommended).



2.Afterwards ,in the BankDb class, you need to specify the URL, username and password for your database: 

3.In the project folder,there is a Queries.txt file included with the queries for creating a database with the same name and tables which we used. Of course, these can be changed but if you change the name of the tables or rows, you should replace them in the code as well.( We’ve also added the mysql java connector library to the project, because ,without it, the database will not connect to your project!)



4. After the preparation is done,you can simply build and run the project and test it.

4.Conclusions and further improvement

To sum it all up, the project is a rather simple and easy to use bank application which of course can be improved. It can obviously be used in a bank to manage customers’accounts. Using the same structure, though, other applications for account management can be easily created quite easily . Some improvements would be things like adding accounts in a different currency, transferring money between accounts from the same bank or another bank(with a certain tax, of course),adding more than one account of each type per person and maybe a currency exchange option.

# 