|  |  |
| --- | --- |
| **Web Services and API Development**  **Lecturer:** Julie Power  National College of Ireland – April 2017 | **Brave online Bank**  Online Bank, Mainstream Service developed in Java using NetBeans IDE  **Carlos Amaro |16106261|**  **Mariah Sonja |16104978|**  **Olga Minguett |16121856|** |

## **Introduction**

The specifications of the projects are asking to develop an application for an Online Bank. The mainstream service has to offer for the customers the functionality to create an account, access their account, transfer, withdrawal, lodge and get the account balance as well. The description below are the conditions that we received in the Project brief.

*“****Create*** *Customers should be able to create an account with the bank, and a customer who has an account should be able to add additional accounts. For example, a typical customer may have a current account and a savings account.*

***Lodgement*** *for the lodgement, a bank customer can specify the amount to lodge with the credit card that will be debited.*

***Transfer*** *For the transfer, the bank customer can specify the amount to transfer and an account to transfer to.*

***Withdrawal*** *For withdrawal, the bank customer can specify the amount to withdraw and the card that will be credited.*

***Balance*** *The customer can request a balance on any account at any time.”*

The API requirements for the Online Bank presented to us is composed of three vital points.

***“BACKEND*** *– A server implementing at least THREE of the entry points listed above (Lodgement / Transfer / Withdrawal / Balance) returning resource representations in XML and JSON.*

***BACKEND*** *– A server developed in Java which implements at least THREE of the entry points listed (Lodgement / Transfer / Withdrawal / Balance) of the API using in-memory objects. Constraints should be implemented, balances should be updated and transactions should be remembered as the API is called. This should be achieved with a database.*

***FRONTEND*** *– An HTML+ JavaScript or Mobile or Desktop client calling ALL portions of the API.”*

We developed a desktop application were the users will be able through a GUI (Maven - Java Application) to login, register, get account’s balance and add accounts (Maximum Three – pre-set condition). There are also the options of *Lodgement, Transfer* and *Withdrawal*. For this, we developed a database – SQL – within 3 tables. One table for *accounts*, another one for *customers* and the last one for *transactions*, connected with the Web resources (Maven – Web Application).

## **Development Process**

### **The RESTful API**

We used NetBeans to create the API using GlassFish Server 4.1.1 with Java EE 7 we, and linked it to our databases that was developed in MySQL Workbench. The GearHost Cloud Hosting was used to host the database and have it live thee of the team members, I case of ay change. We decided to have *Create Account*, *Transfer, Withdrawal* and *Balance* as main functionalities for the users. The customers can have their *account balance* and *transactions* updated as soon as they have insert the details in the proper GUI, in relation whit the action that they want to perform in the system. It is possible because once one transaction is processed, the databases, in connection with the server, it will be updated.

The two primary concerns that need to be addressed:

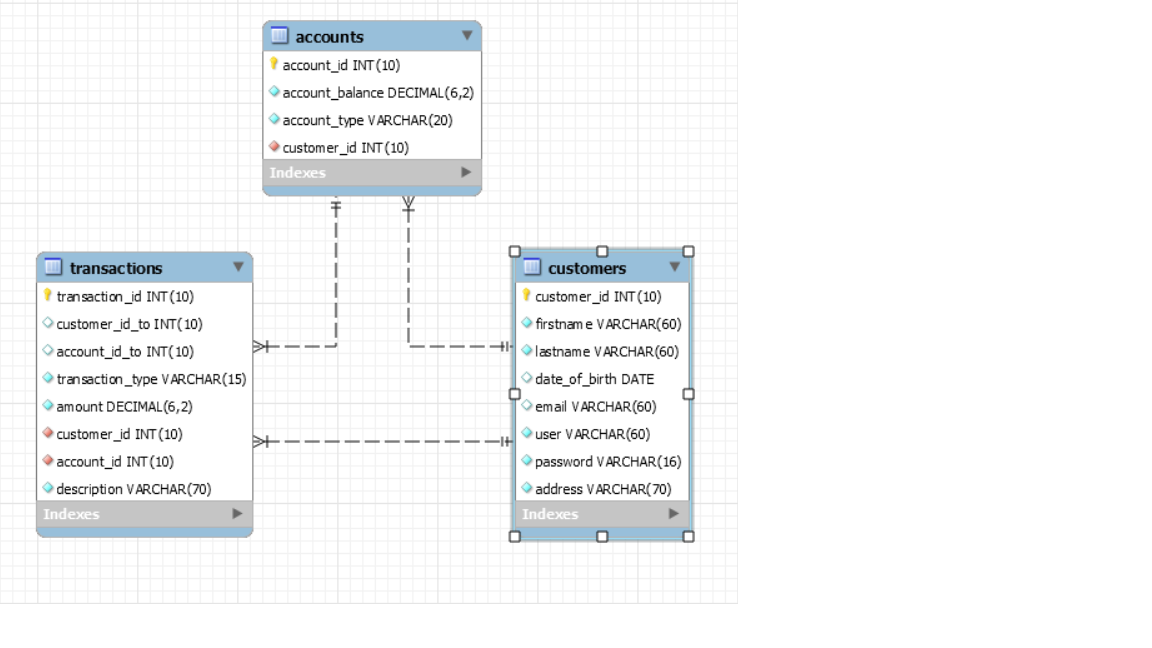
1. Preventing unauthorized users from gaining access to protected content.
2. Preventing protected content from being read while it is being transmitted.

All the required entry points are covered by our Online bank. The source code is in the main folder of this project.

### **The Database**

For the databases, we’ve created three tables: *Customers*, *Accounts* and *Transactions*. The *Customers* table has a “one to many” relationship with the *Transactions* table (A customer can make many transactions); the *Accounts* table has also a “one to many” relationship with the *Transactions* table (An account can process many transactions). Finally, our customer can have more than one account, therefore, our *Customers* table has “one to many” relationship with the *Accounts* table (A customer can have many accounts).

The following image shows the relationship between tables in the database and explore the “one to many” relationships we have talked above.



**Image 1: Database - Tables' relationship**

### **The User Interface – GUI**

The Brave Online Bank can be accessed by the users through the desktop, using the GUI. We created 7 GUIs in order to have all functionalities available for the user.

|  |  |
| --- | --- |
| * Login – if already registered; | * Register -create a new user; |
|  |  |
| * Add a new account; | * Account details and balance; |
|  |  |
| * Withdraw; | * Lodgement; |
|  |  |
| * Transfer; |  |
|  |  |

**How to use Brave Bank Online Application**

1. The user will face an interface which offers at the beginning two options, first one is Login - if they are registered already - or the second one to register as a new user in the application.
2. If they decided to login that will bring them to the Main screen which shows them details and balance of all their accounts, the main functionalities as withdrawal, lodgement and transfer between accounts and to add a new account are displayed.
3. If they decided the latter, the system bring them to another interface in which they must select which type of bank account do they want to create offering options as Current, Savings and Loan, the account number is automatically generated if the action is processed pressing ADD NEW ACCOUNT. "You have Successfully created your account!" or opposite "Account not created!", press HOME and come back to the main screen and REFRESH to check all the latest updates in that interface.
4. For WITHDRAW, the user will have the option to select between the accounts in which one the withdraw is going to be processed, enter the amount and description, press PROCESS and then HOME, in the main screen press REFRESH and the money is discounted. In the LODGEMENT, it´s going to be the same thematic than in withdraw but in this case, the money will be received and added in the bank account. For TRANSFER, the user should select between the accounts that the transfer will be processed, both accounts in their own bank accounts, add the amount and description as well as in the other functionalities before described, PROCESS and then HOME, after in the main screen press RFRESH, and check the account that increase or decrease in funds.
5. In the case, that the user select REGISTER, the new interface launch is self-descriptive and it will ask for user´s personal information like: email, first name, last name, date of birth, address, username and password (both of these last two elements important for the user) then the following interface it´s for adding an account to this new customer account. Which bring us back to the point 3, and all the functionalities displayed in that interface.
6. When the user is finished, they can press LOG OUT, to exit the application, which bring them back to the LOG IN / REGISTER interface.