**IT7742 Advanced Programming**

**Assessment 3: Source Control and Documentation**

**Task 2:**

**public void AddAccount(string name, int id, decimal fee)**

Adds a new account using name, email, phone, staff variable in form – adds to List<Account>

**public void AddCustomer(string name, string email, int phone, bool staff)**

Adds a new customer using name, id, fee variable in form – adds to List<Customer>

**public Customer RemoveCustomer(int id)**

Find a customer by id from the list in Controller in a for loop

**public Account RemoveAccount(int id)**

Find an account by id from the list in Controller in a for loop

**public void RemoveC(int id)**

Find a customer using RemoveCustomer & remove from controller list

**public void RemoveA(int id)**

Find an account using RemoveAccount & remove from controller list

**public decimal Deposit(decimal amount, decimal balance)**

Takes amount and balance variables from form, throw exception if amount under 0 otherwise add amount to balance

**public decimal Withdraw(decimal amount, decimal balance)**

Takes amount and balance variables from form, throw exception if amount more than balance otherwise subtract amount to balance

**public decimal Interest(decimal balance, decimal interest)**

Takes interest and balance variables from form, throw exception if balance under threshold (where needed) otherwise calculate interest and add to balance

**public Account GetAccount(int id)**

Takes id variable from form, compare id in controller list and return account

**public Customer GetCustomer(int id)**

Takes id variable from form, compare id in controller list and return customer

**public decimal[] Transfer(decimal amount, int id1, int id2)**

Takes amount and 2 id variables from form, throws exception if amount under or equal to 0 or if both ids are the same. Gets the 2 account variables using GetAccount() method, subtracts balance from fromaccount and adds to toaccount. Returns both balances via index of decimal, can be called with varname[index].

**WriteBinaryData()**

Creates a formatting object, create IO stream to write to .bin, seralize custlist and singleton customer then close. Interacts with read in order to get the binary data.

**WriteBinaryDataAccount()**

Creates a formatting object, create IO stream to write to .bin, seralize acclist and singleton account then close. Interacts with read in order to get the binary data.

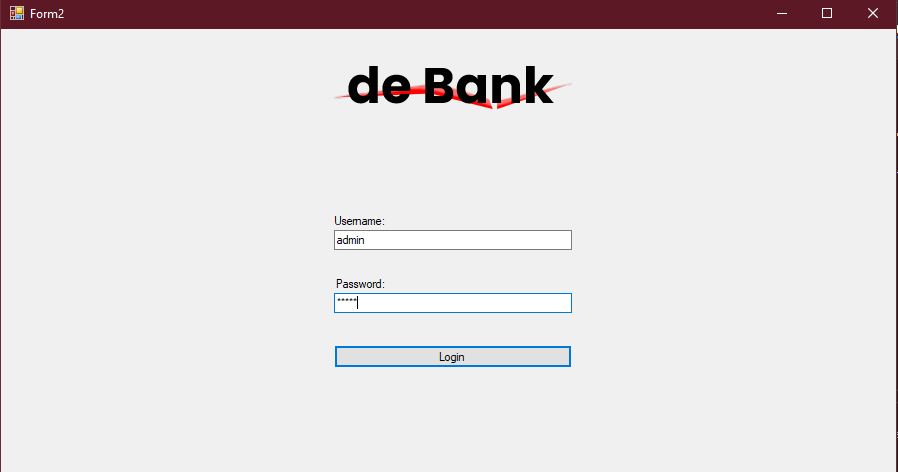
**Task 7:**

**User guide**

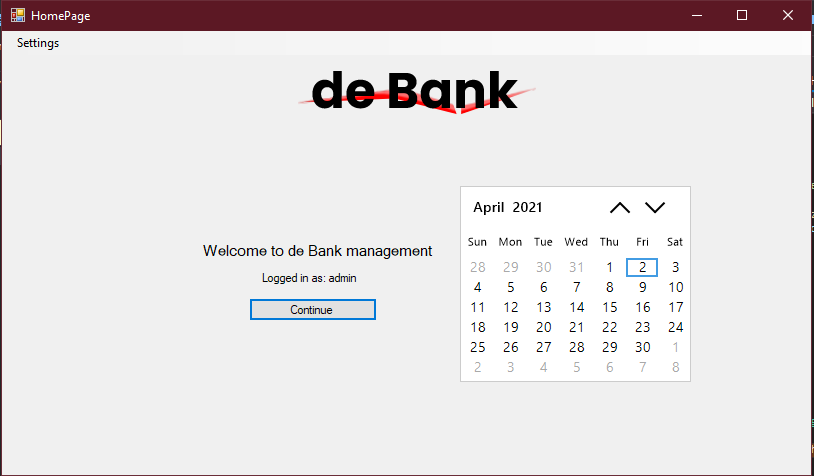
Functionality:

* Login
* CRUD functionality for customers
* CRUD functionality for accounts
* Account transactions: Withdraw, Deposit, Interest, Transfer account to account
* Search both customer and account records

User will login to the application:

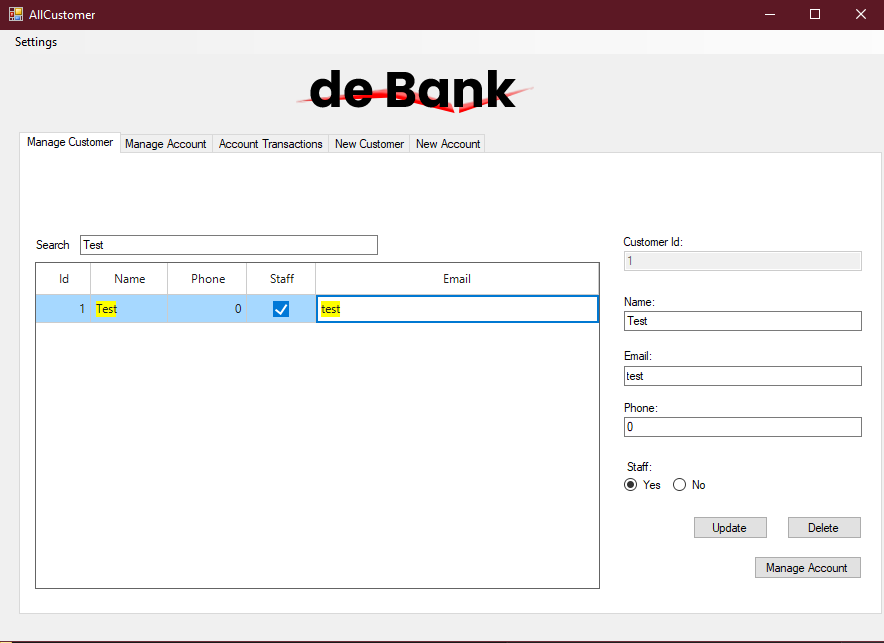


Homepage will be shown, continue button takes the user to the main form:



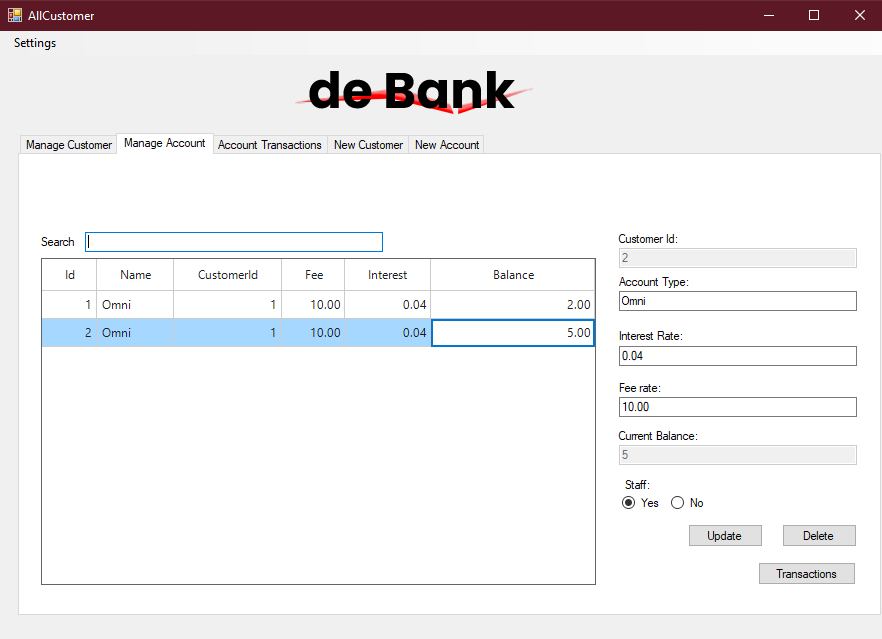
The manage customer tab has the following features:

* Search for records
* Show customer records that have been added
* Updating of customer records
* Deletion of customer records
* When a record is selected, manage account button can be clicked and then taken to the next tab



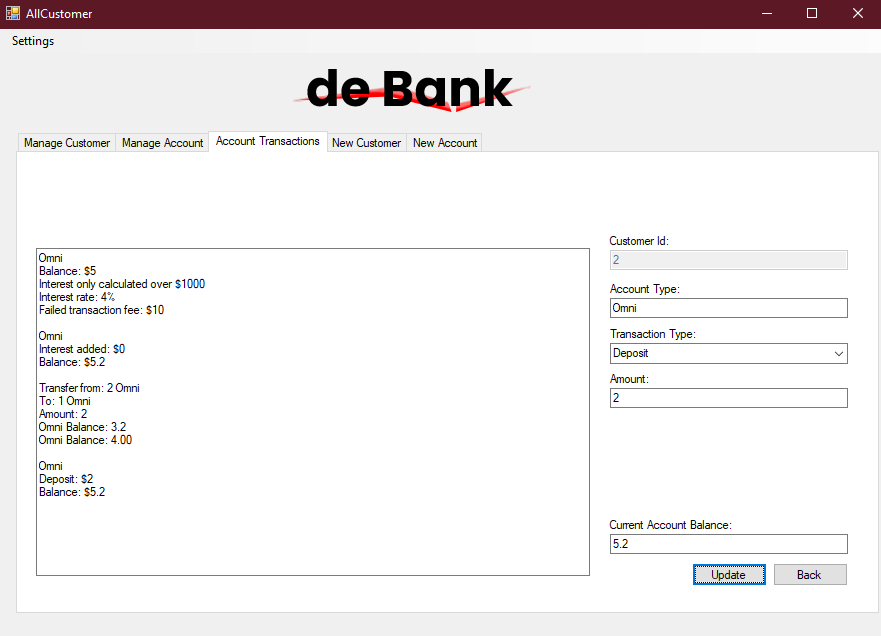
The manage accounttab has the following features:

* Search for records
* Show account records that have been added for selected customers
* Updating of account records
* Deletion of account records
* When a record is selected, transacton button can be clicked and then taken to the next tab

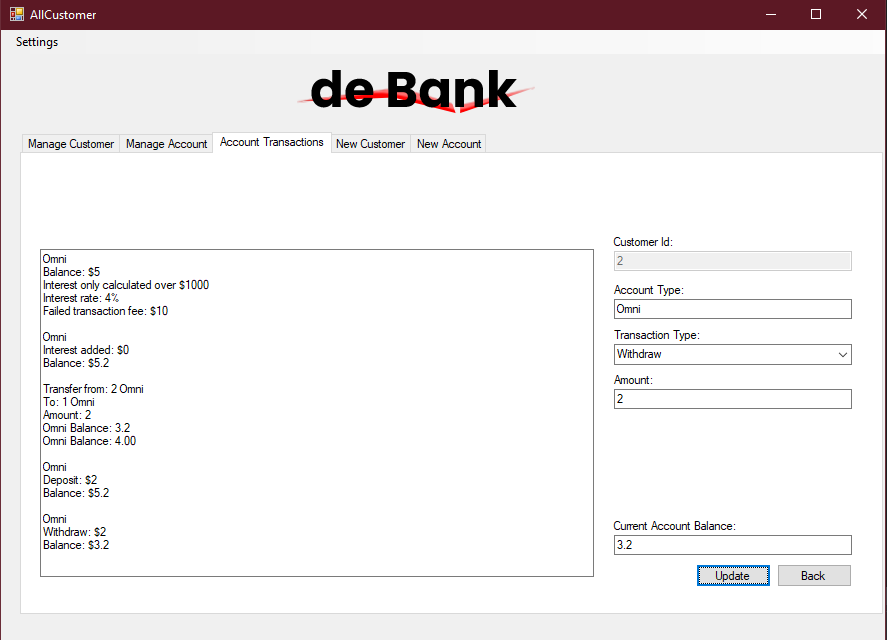


Transactions tab has the options based on type selected:

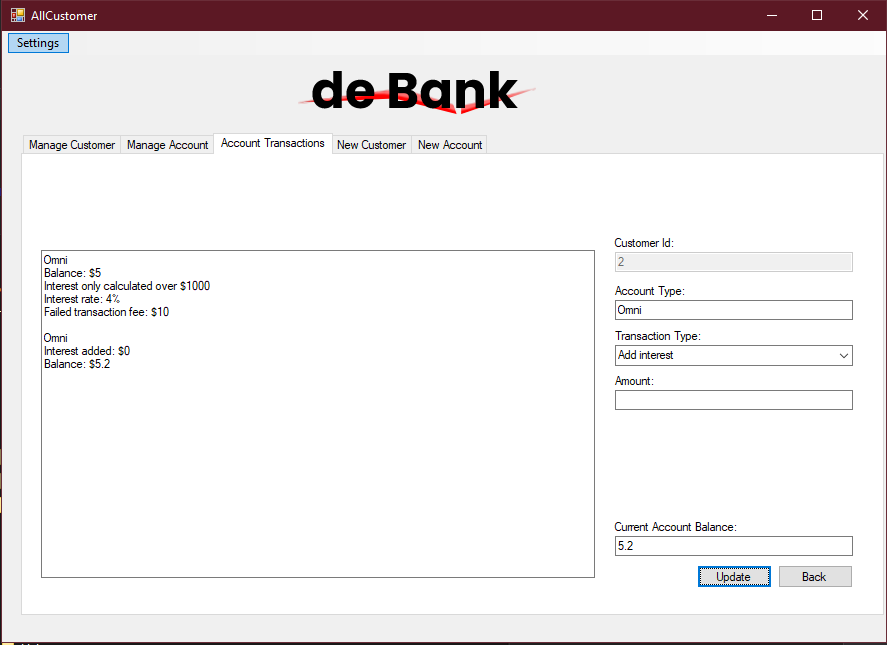
Deposit takes amount and shows info based on what has happened, amount is added to balance by clicking the update button.



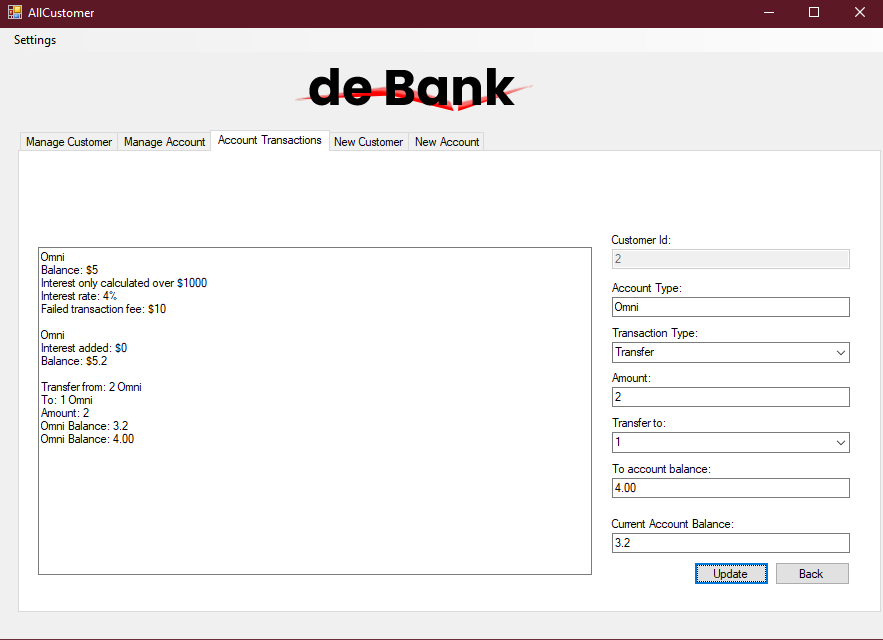
Withdraw takes amount and shows info based on what has happened, amount is subtracted from balance by clicking the update button.



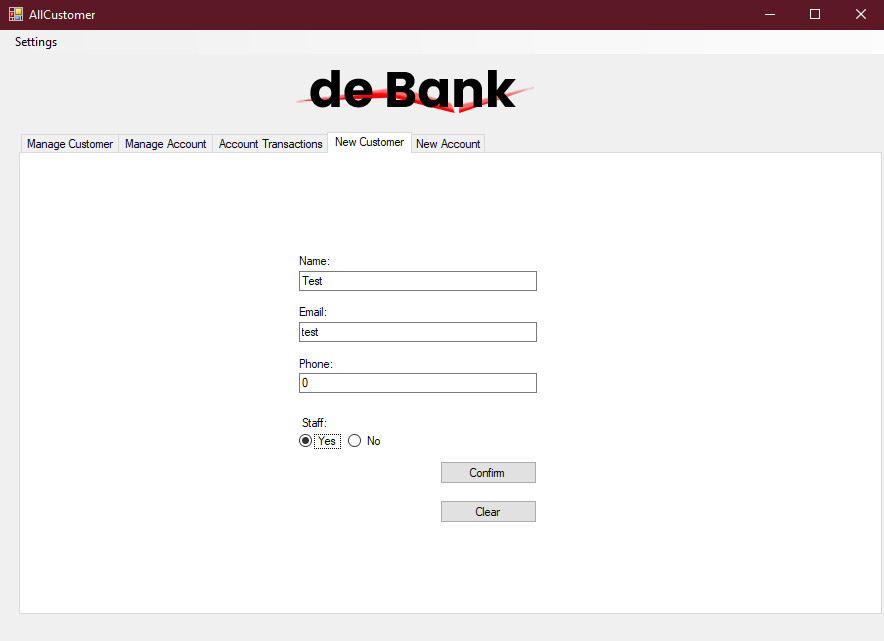
Interest is added by clicking the update button based on set variables, info is shown based on what has happened.



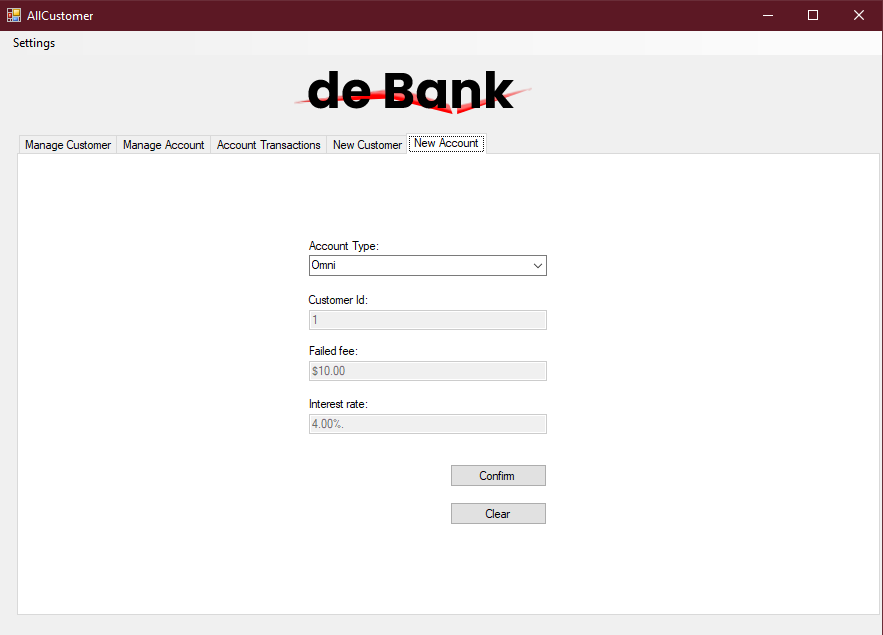
When transfer is selected from type, the transfer to box will appear. When an account is selected from this the balance will appear. When update button is clicked the amount will be transferred.



Self explanatory really, adds a new customer.



New account can be made but can only be made when a customer is selected.



Fresh open flow:

Login > Homepage > New customer > Manage customer > new account > transactions

Next open:

Login > Homepage > Manage customer > Transactions or new or whatever needed

**Task 8**

Given account <name> is selected when I transfer <amount> to <type> then <type> is transferred to

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Amount | Type | As intended |
| Omni | 400 | Omni | Yes |
| Omni | 100 | Lifestyle | Yes |
| Lifestyle | 10 | Lifestyle | Yes |
| Lifestyle | 400 | Omni | Yes |
| Everyday | 100 | Everyday | Yes |
| Everyday | 10 | Omni | Yes |

Given account <name> is selected when I transfer <amount> to <type> then <type> is transferred to

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Amount | Type | As intended |
| Omni | 400 | Everyday | Yes |
| Omni | 100 | Lifestyle | Yes |
| Lifestyle | 10 | Everyday | Yes |
| Lifestyle | 0 | Omni | Yes |
| Everyday | 0 | Lifestyle | Yes |
| Everyday | 0 | Omni | Yes |

Given customer is not <staff> when <amount> transferred is more than <balance> then the account is charged a <fee> and the <fee> should be withdrawn from the <balance>

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Staff | Amount | Balance | Fee | As intended |
| false | 100 | 0 | 10 | Yes |
| false | 3 | 0 | 10 | Yes |
| false | 105 | 5 | 10 | Yes |
| false | 44 | 901 | 10 | Yes |

Given customer is <staff> when <amount> transferred is more than <balance> then the account is charged a <fee> and the <fee> should be withdrawn from the <balance>

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Staff | Amount | Balance | Fee | As intended |
| false | 100 | 0 | 5 | Yes |
| false | 3 | 0 | 5 | Yes |
| false | 105 | 5 | 5 | Yes |
| false | 44 | 901 | 5 | Yes |