

Please either download the Visual Studio solution from Github location at <https://github.com/CardWebCoder/CWS.SimpleBank> and open the solution in Visual Studio 2019. (or clone directly from Visual Studio 2019)

Database is host in Azure service. The connection string is

```
data source=cwinterview101.database.windows.net;initial catalog=Interview101;user id=CardWebCoder;password=Coder@1234;MultipleActiveResultSets=True
```

Solution description

Simple Bank provides the following financial products to customers;

- Certificate of Deposit
- Loan
- Credit Card

Each customer can have many of these products or none. Please see the edmx diagram (in CWS.SimpleBank.Data project) to see the database structure.

Please run the project after it is loaded, part of the website is already completed. You will see a list of customers by clicking the top menu option “Customers”. There are 2 customers in the database for this exercise.

There is a “Accounts” link on the right side of each customer. It should take user to see a list of accounts this customer has.

Task 1:

In customer’s account list screen, list the accounts by product type (ie, Deposit, Loan and Credit). Based on product type, each row needs to display data elements relevant to the product.

The change should be done in Views/Customer/Accounts.chtml

The view has the table heading for each product section completed, please write the necessary code to display the account data.

Additional Information

The Model in this view is a **Customer** object, which contains an **Accounts** objects. Accounts is a list of **Account** that is a base class for various sub classes; they are **CertificateDeposit**, **CreditCardAccount** and **LoanAccount**.

Task 2:

Implement `IAccountProcessor.CalculateInterest` in **DepositProcessor** classes. See `CWS.SimplBank\Models`.

This formula is to calculate the amount of interest an account will earn in number of “days”, which is one of the input parameters.

Deposit account interest is calculated by compounding interest. The `presentValue` is account’s current balance. calculate daily rate using account’s APY divided by 365.

Interest Amount is

$$\text{presentValue} * (1 + \text{dailyRate}) ** \text{days} - \text{presentValue}$$

Task 3:

Complete the needed action to calculate interest.

Update `AccountController.CalculateInterest(customerNumber, accountNumber, days)` to

1. Get account object of `LoanAccount`, `CertificateDeposit` or `CreditCardAccount`
2. Return the interest amount by the appropriate processor. Develop additional implementation to apply proper pattern if necessary.

To test and verify, use browser with the following uri

<http://localhost:{portnumber}/account/calculateinterest/customerNumber=1&accountNumber=10>. Change the `customerNumber` and `accountNumber` to test other product types

Additional Information

1. `BankService.GetCustomer(int)` returns a `Customer` object that contains customer’s accounts. Use this to get necessary account object
2. Determine the best approach to instantiate the appropriate implementation of `AccountProcessor` to calculate the interest amount.

Task 4:

Complete Calculator view in `Views/Account/Calculator.cshtml`.

When user clicks a button, this screen should make a call to `AccountController.CalculateInterest` and display the result in `#interest-amount`.

Test by using uri

<http://localhost:{portnumber}/account/calculator?customerNumber=1&accountNumber=10&days=30>. Use other query string value to see various product types.

Task 5:

Update Views/Customer/Accounts.chnl to add an Interest link next to each account. When user clicks Interest link, display Calculator in a modal dialog box.