Banking System Project

Introduction

This is a simple Banking system, I just base on some main functionals such as Withdraw, Deposit, Fund Transfer, View transaction histories.

Objectives

I designed the architecture for this project base on 3 Layers pattern, Repository and OOD (SOLID). Moreover, I splitted the Identity from Presentation Layer into another layer called Security to follow DI principle and easier to make relationships between others entity model, etc... Beside, this project represent for solving the concurrency problem and adapt some non-functional requirements such as:

1. Maintainability.
2. Readability.
3. Testability.

Technologies

Asp.Net Core, EF7, AutoMapper, Jquery, Html, Css, XUnit.

How to build code

1. Open project with Visual Studio 2015(install .NET Core) or 2017.
2. After the project are openned, you will see the project details on the Solution Explorer, then right-click on the solution the choose Build to build and restore necessary package.
3. If the solution build success, the project now ready to run but before running project you should check you connectionString on the appsetting.json file on the BankSystem project is correct or not, if not, change it then open Package Manager Console, choose Default project to "BankSystem.DAL" and type 'Update-Database' command to create database.
4. Open the seed data sql file in the attachment and run it (the seed data for concurrency case, you have to run after database was created).
5. Now, the project ready to run. All you should do is deploy the web project to the server or running it on your local machine.

How to test concurrency problem

The concurrency problem happend when a data are modified from 2 differences source and it cause the missing and update wrong data information.

Solution

I used EF7 for this project. Firstly, I create a RowVersion(TimeStamp type) for the table that I need to handle Concurrency problem. Then, I everytime we update the data, It will automatically add an condition to our update query for comparing the RowVersion in db is same as the version in source data.

You can read this Link for more information.

Testing

After you finish the Build step above, you should access to the url: youdomain.com/Account/TestConcurrency. Then you will see the button named Submit after you click it, it will execute an method to create the Concurrency problem case.

For more information, please access the AccountService.cs (Service Layer), find function ConcurrencyTest and the you will see another comment for the details.

Improvement

There are many improvement for this project following below:

- Create more unit test for this project such as Presentation Layer, DAL, Service Layer.
- Improving security and UI for user.
- Giving comment on every function in the project to make sure other developer in team or future team-mate can understand it.
- Applying UnitOfWork pattern to manage data transaction.

Thanks for reading this.;)