**Design details for Banking System app**

**Bank end using C#,.Net core.**

**Classes**:

**Account**: Basic details of client

Account No and ClientName, LocalCurrency, CurrentBalance

**Transactions**:

TransactionId (primary Key),

Account No(Foreign Key) , DateTime, Credit, Debit, Balance

**Deposit**: Command action for credit.

This is implemented and exchange applied if Account local currency and deposit currency are different.

**Withdraw**: Command action for Debit action.

**CreateAccount:** Creates new Account

**ViewTransaction**: Command action for getting latest transaction between From and To Date for an account.

**ExchangeRate**: Exchange rest API call to get the exchange rate for depositing different currency conversions

**Design patterns:**

1. Command design pattern for actions ( Deposit, Withdraw, View Transaction, Get Balance) to separate the execution from invoker and client
2. Dependency injection Account, Exchange Rate classes, Configuration, DBContext
3. Singleton classes: Account, ExchangeRate are configured for singleton

**Database: SQL Server 2018**

Added EF using command tool

dotnet tool install --global dotnet-ef

dotnet add package Microsoft.EntityFrameworkCore.Design

dotnet ef migrations add InitialCreate

dotnet ef database update

Add migration using powershell(Package manager command) commands

a) Add-Migration InitialCreate

b) Update-Database

Database name is BankDB Added in App

**Tables created:**

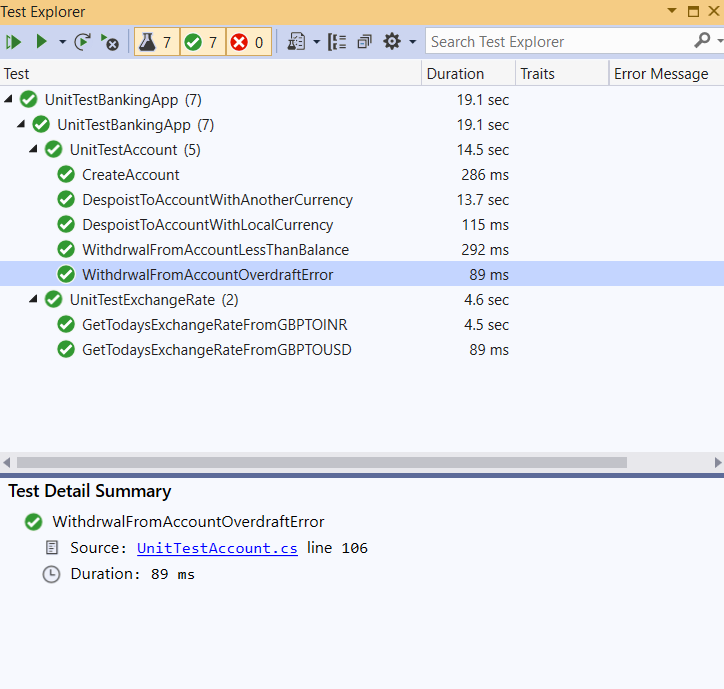
Accounts,

Transactions

**Unit Test case**

1. Using MS Test framework Sample Test case is added for Exchange rate
2. 7 test case written to test and image is attached below to show that they all have excueted.
3. These test have   
   Create account,  
   Desposit with base currency and different currency  
   Withdraw ( error handling and Positive case)

Exchange Rate REST API test



**Configuration**

1. There is DB connection string added as part of the appsettings.json

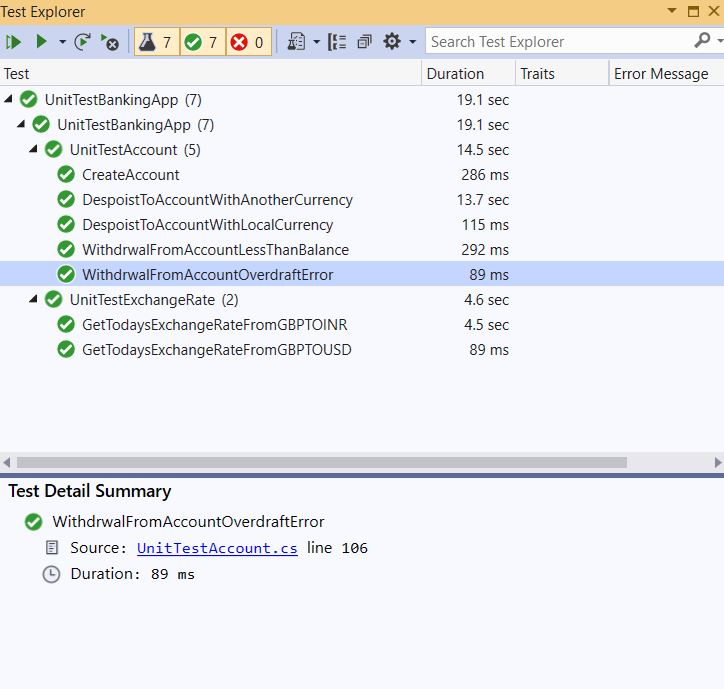
DB is loaded as part of the configure services method of start up method

1. ExchangeRateBaseURL is also added as configuration

**Error Messages:**

Overdraft error message is stored in const error message file so that any changes it will be single place. It is just sample of error messages and could be stored more.

**To Do:**

1. **Front end Angular is not implemented as I did not have time**
2. **View Transaction is implemented but no test**
3. **Import Pdf for user friendly method is not implemented**
4. **Machine readable transaction ( Could be json or xml file could be exported from transaction list)**
5. **Logging**
6. **More exception handling**