Research and Development

APTEAN EDGE BANK

Group Project – Microsoft .Net C# and SQL Server

Version 1.3 – June 17, 2014

Problem Statement

The Aptean Edge Bank has decided to develop a new computerized banking system to replace its archaic, but time-proven, system of spreadsheets and abacuses. You have been chosen to embark on this hazardous mission to develop the system to include a graphical user interface which promotes a "mouse-friendly" environment. Their IT Consultants have advised to use Microsoft .NET's Windows Forms technology as the platform as SQL Server 2012 as backend database. Your mission is to develop the Banking application for the Bank. The bankers are the end-users of the new interface; they are responsible for managing customers and performing typical banking operations on behalf of these customers.

This group project assignment will utilize your skills in graphical user-interface development with Microsoft .NET Windows Forms and database.

The following skills and software are required to complete this project:

- Microsoft .NET Windows Forms (WinForms) Programming
- Visual C# Programming Language
- Microsoft .NET 4.0 Framework
- Visual Studio 2010
- Graphical user-interface design and programming skills.
- SQL Server 2012
- Software development skills with databases and SQL.

Functional Requirements –

The Aptean Edge Bank banker's collective has submitted the following demands with regards to the new UX.

- When the application starts, the user should be presented with a list of all customers.
- The user can open a customer in a new modal window, from which the customer's details and accounts can be operated upon.
- All customer operations can be performed from within the new UI.
- All account operations can be performed from within the new UI.
- Errors and exceptions must be elegantly, and appropriately, handled by the new UI.
    - The application should not crash.
    - The user must be notified of errors where appropriate. (ie. including logic errors, such as insufficient funds available).

Aptean Edge Bank Functional Requirements in detail

The functional requirements of project "Aptean Triple I", version 2.0 of the Aptean Edge Bank's computerized banking system, is outlined below.

The goal of this project is to implement a realistic banking system which tracks the bank's money, from which loans can be provided to customers.

For the curious, we are implementing a variation of a fractional-reserve banking system (http://en.wikipedia.org/wiki/Fractional_reserve_banking). This system of banking describes a bank which loans money from funds deposited by its customers. The astute would notice that if the customer re-deposits the borrowed funds to the bank, the funds can be loaned once more by the bank.

The project has been simplified to not enforce a fractional-reserve, as such this banking system would probably be called a zero-reserve banking system. This is a tricky financial system, as the bank can theoretically loan an infinite amount of money if all funds are re-deposited to the same bank by the customer.

Cash

Cash are funds entering or leaving the banking system. Any amount of money that is moving in or out of the banking system is considered cash.

Accounts

An account represents an amount of money managed by the bank.

The following requirements apply to all types of accounts, unless otherwise specified.

- An account is represented by at least an ID and balance.
- An account's balance cannot be negative.
- An account cannot be closed if it has a non-zero balance.
- An account is associated with exactly one customer or the bank.

The Bank

The bank maintains a general account for itself to track the money held by the bank.

- There must exist exactly one such account for the bank.
- The bank's accounts can never be closed.

Bank's General Account: This is like a regular customer's chequing account. It represents the money held by the bank. Interest revenue earned from loans (ie. profit) would be deposited to this account.

- Money deposited by the customer would also be deposited to the bank's general account.
- Money withdrawn by the customer would also be withdrawn from the bank's general account.

    - A customer cannot withdraw funds exceeding the balance of the bank's general account. ie. The bank cannot give the customer money it doesn't have.

- Money loaned to customers would come from the bank's general account.

The Customer

The concept of a customer and a customer's accounts have changed little from the version 1.0 specifications.

Customer: A customer of the bank is represented by an ID, their name, and their join date.

Customer

The Aptean Edge Bank maintains a profile of each customer and their accounts.

Property	Data Type	Description
Customer Id	Integer	Each customer is assigned a customer identifier which is unique.
Name	String	The customer's name.
Date Joined	DateTime	The date and time when the customer joined the Aptean Edge Bank.

Customer General Accounts

A customer can have general bank accounts representing the money owned by the customer, and therefore owed by the bank.

The following requirements apply to customer general accounts:

- Each account tracks its date opened and date closed values.

- Funds can be deposited and withdrawn from a customer general account. These funds are respectively treated as cash entering and leaving the banking system.

"Aptean Integrity" Chequing Account: A customer's simple savings account. A customer can have zero or more chequing accounts.

"Aptean Intensity" Tax-Free Savings Account: Identical to the chequing account, with the exception of a maximum balance policy of $5,000. A customer can have zero or one tax-free-savings account.

Property	Data Type	Description
Account Id	Integer	Each account is assigned an account number which is unique. Account numbers of closed accounts cannot be reused.
Date Opened	DateTime	The date and time when the account was opened.
Balance	Decimal	The current balance of the account. Values will be stored with 2 decimal places. The balance cannot be negative.

In addition, an account must enforce the following behavior or detail:

- The balance must be zero before the account can be closed.
- Each account is associated with a customer. No customer can have more than one account of each type.

Chequing Account

The "Aptean Integrity" chequing account is a simple savings account from which funds can be deposited and withdrawn.

There is no upper limit to the amount of funds which can be deposited to this account.

Tax-Free Savings Account

The "Aptean Intensity" tax-free savings account is a special account from which an upper limit on the balance is enforced. This account type serves a generous government initiative where taxes cannot be incurred on funds sheltered in this account. However, the government's generosity only goes so far, the maximum balance permitted on this account is $5000.

Tip: Create the two account types as sub-classes of an abstract class "Account". Ie. "ChequingAccount" and "TFSAccount".

Customer Liability Account

A customer can have a liability account where money loaned by the bank to the customer is tracked. Positive balance in this account represents money owed by the customer.

The following requirements apply to customer liability accounts:

- A liability account cannot be closed.
- Liability accounts do *not* need to track the date when it was opened.
- Issuing Loans

    - Loans issued to the customer would add to the balance of a liability account.
    - A loan cannot be issued for an amount more than the remaining balance of the bank's general account. ie. The bank cannot loan more money than it has available.
    - Funds to issue a loan are funded from the bank's general account.
    - The customer may choose to deposit the loaned funds to a general account (possibly as a separate transaction). Otherwise, funds issued for a loan are assumed to leave the bank.

- Loan Repayment
    - Repayment of loans by the customer would subtract from the balance of a liability account.
        - The customer can repay a partial amount for a loan.
    - Funds from a loan repayment are deposited to the bank's general account.
    - The customer may choose to withdraw funds from their general account to repay a loan (possibly as a separate transaction). Otherwise, funds used for loan repayment are assumed to be cash entering the bank.

- Interest on Loans
    - Customers pay interest when they repay their loans.
        - To keep this project simple, customer's pay an arbitrary positive amount as interest on a loan.
    - Interest revenue (interest earned from loans) for the bank is deposited directly to the bank's general account.
        - ie. Interest paid for a loan does not affect the liability account's balance.

Note: To keep the project simple, the terms of a loan are not tracked by this banking system.

"Aptean Intelligence" Liability Account: A customer's liability account. A customer can have zero or one liability account.

Account Activity Ledger

The account activity ledger is a log of each action which affects the balance of an account. This log can be used to reconstruct an account's activity. It can also be used to recalculate an account's current balance.

- Each activity has an ID; IDs are issued sequentially in the order of each action as they occur.
    - A sequence of activities performed to complete a banking operation must have a sequence of consecutive IDs. ie. A customer deposit operation includes an action to deposit the amount to the customer's account and an action to deposit the amount to the bank's account.
- Each activity records the account ID, amount, timestamp, an activity code, and an optional note describing the action.
    - The activity code represents the action taken; such as a deposit or a withdrawal.
    - (Optional) If you are familiar with general ledgers in financial accounting, you may represent the amount as debit and credit amounts instead. http://en.wikipedia.org/wiki/Debits_and_credits

Banking Operations

The following operations can be performed within the Aptean Edge Bank system through an updated version 2.0 "Visual-Mouse-io UX" graphical user interface. The user of this interface are the bankers of Aptean Edge Bank.

Appropriate error messages must be displayed where appropriate.

Customer Management

Operation	Behaviour
Create New Customer	A new customer profile is created.
Read Customer Profile	An existing customer's profile are retrieved.
Read Customer List	A list of all customers is retrieved.
Read Customer's Account List	A list of all general and liability accounts for a given customer is retrieved.
Update Customer Profile	An existing customer's profile is updated (ie. name and date-joined values).

Customer General Account Management

The following operations are available for customer general accounts.

Operation	Behaviour
Create New Account	A new account (chequing or TFSA) is created for a given customer.
Read Account Details	An existing account's details are retrieved; this includes the account balance.
Close Account	Close an existing account.
Reopen Account	Reopen a closed account.
Deposit	Funds can be deposited to the account. This adjusts the account's balance accordingly.
Withdraw	Funds can be withdrawn from the account.

Customer Liability Account Operations

The following operations are available for customer liability accounts.

Operation	Behaviour
Create New Account	A new liability account is created for a given customer.
Read Account Details	An existing account's details are retrieved; this includes the account balance.
Issue Loan	A loan can be issued for the account.
Loan Repayment	A loan can be repaid for the account. An arbitrary interest amount is paid with the loan repayment.

Bank Management

The following operations are available for the bank and its accounts.

Operation	Behaviour
Read Bank's General Account	The bank's general account details are retrieved; this includes the account balance.

Activity Ledger

The following operations will be available with respect to the account activity ledger.

Operation	Behaviour
Read Account Activity	Retrieve the activity log for a given account.
Calculate Account Balance	Any account's balance can be recalculated by summing the logged amounts appropriately.

Database Design

An appropriate layout of tables, columns, fields, and data types must be designed for the database.

Consider the following as you design your database:

- Primary and foreign keys.
- Appropriate columns data types.
- Nullable and non-nullable columns.

NOTE: Please name your databases using the following pattern: "ApteanBankXXXX", where "XXXX" is your first name.

ie. Amit would name his database "ApteanBankAmit".

Unit Testing Requirements

The Federal Banking Commission demanded that the following guidelines be followed with regards to the unit tests.

- Unit tests are to be written for the computerized banking system's class library.
- Each unit test class should contain tests only for one class.
    - ie. CustomerTests.cs should only contain tests for the Customer class.
- Unit tests must be written for normal cases.
    - ie. Withdraw an amount that is less than or equal to the remaining account balance.
- Unit tests must be written for expected error cases.
    - ie. Attempt to withdraw an amount that exceeds the remaining account balance.
- Unit tests must be written for exceptional cases.

    - ie. Attempt to withdraw a nonsensical amount (ie. negative values) from an account.

- A solution must be provided to manage the test data used by the unit tests.
    - Unit tests must be repeatable.
    - Unit tests can be executed in random sequence.
- All unit tests must pass.
- The windows forms application does not need to be unit tested.

Do your best to implement as much coverage as possible.

NOTE: Consider test cases for the data-access layer as well; such as connection failures (server not found).

Deliverables

Once all assignment objectives have been completed, submit the deliverables as a zip file to your mentors.

Deliverable	Description
Code Files	All *.cs, *.csproj, and *.sln files for the assignment. The intermediate output files under the "obj" folder are not necessary.
Output Binaries	The output binaries for the assignment (*.exe & *.dll files). The *.pdb debug binaries are not necessary.
Database Backup File	Using SQL Server Management Studio, create a backup file of your database. Use the name of your database as the name of the output *.bak file. A single *.bak file should be produced.
QA Test Cases	A word document containing a list of all QA test cases.
XML API Documentation	The *.xml files generated for the API class library. Bonus points for generating a help file from the XML documentation using Sandcastle (http://shfb.codeplex.com/).
TFS References	A list of TFS references for the following items:TFS work items (tasks, use cases, marketable features, etc.) for the project. Alternatively, provide a query which would return the work items -TFS source control path to where the project's code is checked in.