DPS915/VBA544 Winter 2014 Assignment 2

Due Date: Tuesday April 08 2014 (Only A and A+ students will be contacted by me for a one-on-one demonstration after the due date. Demonstration of these students will be on Friday April 11 2014 in your lab period)

Overview

This assignment is part two of the two-part **retail store**. This second part deals with creating a website for the retail store. This website will make use of the business logic you built in assignment 1. Your assignment 2 does the following:

- creates a database driven website using ASP.Net MVC, Entity Framework, and SQL Server (Local DB).
- integrates the .Net library you created in assignment 1 to work with ASP.Net MVC.
- permits your database to import and export application database data to and from JSON.
- provides unit tests for ASP.Net MVC 5 website. The unit tests must demonstrate the website requirements proving that it can store data to a SQL Server database, handle invalid data appropriately and handle the CRUD requirements correctly as described in the requirements below.
- provides unit tests for the library code too. Any new code written after assignment 1 must have its corresponding unit tests.

Learning Outcomes

- Understand ASP.Net MVC 5 (model, view and controller) programming paradigm.
- Understand Entity Framework, particularly Code First method of creating databases.
- Handle multi-language development in .Net: use both VB.Net and C# code.
- Understand the benefits of re-using code previously created with Object Oriented Design Practices.
- Understand communication between library and client on error and event situations.
- Create a polished presentation of the library and the website.
- Understand how Test Driven Development integrates with websites.

Team Work

You may work in groups of (maximum) two students for this assignment or if you prefer you may do this assignment by yourself. All students in the same group must be in the same program, i.e. BSD students can only form groups with other BSD students (if you want to work with a student across programs let me know ahead of time).

Business Requirements

Your website has two modes of access:

- **non-admin user:** can see items in the store. Non-admin users can only add new orders but within a newly created order they can add/remove/update items.
- **admin:** can do everything a non-member does, in addition can add/remove/update store items and update and delete orders.

Transfer the functionality of your POS app (from assignment 1) over to the web. This means:

- permit a user to manage their orders (add/remove/list order items).
- only allow a manager (admin) to add/remove/edit items in a store. A manager can also add/remove/edit orders. (Optional) While add/delete/update of customers is not a requirement, it would a nice thing to have. If you decide to have customers, then ensure customers can only add/modify/delete/list their own orders.

In addition to the above requirements your website has the following functionality:

- home page of a non-admin user shows a summary list of all items (sort alphabetically by item name). On the summary list only show item names and their prices on the home page. Clicking on details of an item shows its details with the option of adding that item to a shopping cart.
- home page of an admin shows a summary list of all orders (most recent order at the top). On the summary list of the admin home page, show only the order number (optional if you implement more than one non-admin user (customer) then show customer name) and the total value of the order, clicking on the details link of an order shows its full details.
- when the website starts up, it reads a JSON file created by your assignment 1 to populate the database. Allow saving of data to a JSON file too.
- let the shopping cart have a discount feature. Let the rules for the discount be checked by the library.
- when data is entered, check the data at the client (browser) level (use DataAnnotations to do this). In addition the exception messages thrown by your DLL are trapped by MVC and displayed on the browser. So for example if your discount has an formula of XXXDDMM where XXX is three letters and DDMM is day and month, then you can check the discount formula being entered correctly in the client, but only the library will ensure that the letters match a defined set of sequences (like WIN,SPR,SUM,FAL) and DDMM is a valid date (upto 7 days from the current date).

Make the following changes to your original library design:

- Add at least two extra fields to your item. For example if your store sells books add Publisher and Year Published. Write simple rules to manage these fields (example: year published cannot be before Gutenberg press, or in case of e-books before computers came into existence)
- Add at least one extra field to your order. For example add order placed date and time. Provide a simple rule for this functionality.
- Update your library to read JSON files created by your assignment 1, so the fields that were added later initialize them to safe default values in keeping with the rules you establish.

Design a website using ASP.Net MVC to do the following

- Create/Read/Update/Delete new records that are stored in SQL Server database.
- Prohibit invalid data from entering the local data storage (SQL Server database).
- The website must use the library code to do its reading and writing to and from the database.
- Use test driven methodology to add or change library functionality and only after it was adequately tested, add communication to the website.
- Make the website look and feel nice. Follow recommended guidelines for creating the user friendly websites. The code written in the web interface must be fairly light and be only a thin wrapping layer over the DLL.
- The specifications of the application are deliberately kept flexible to give you freedom to control the implementation.

NOTE

The business requirements could change depending on the difficulty of the assignment, so its possible (though unlikely) that more requirements would be added, or deleted, but its also possible (and highly likely) that requirements might be changed a little to permit the principle being used without duplication and needless complexity. In other words, the website might be made to be as real world as possible yet keeping it simple enough for a one term assignment.

Library: Object Oriented Design

Keep the code from the previous library but show how you have reused the older code to handle the new requirements. The important steps in this phase are the following:

- Make sure that the unit tests work to prevent accidental breakages after you redesign your library. Keep a backup of your code before you proceed to make changes.

- When making changes, create new objects that reuse code from the existing library. Alternatively redesign the library to streamline it with how you think it best to solve this assignment.
- be language independent, this means any .Net programming language should be able to use it. Your library contains both VB.Net and C# code.
- the code is well documented and commented. This means the code follows the guidelines for quality code outlined on my home page.
- it throws exceptions (on error) and generates events (on success) for library users to trap and handle
- it uses collection classes, preferably generic collections, and regular expressions in VB.Net
- it reads and writes data to and from a JSON file. Additionally the library can take data from a locally stored file and update a local SQL Server database.

ASP.Net MVC website

Create a local website that has the following:

- Allow a user to browse (and optionally search) through the store items.
- Allow a manage to browse (and optionally search) through the store and order items.
- Allow a user to add and remove items from their order.
- Allow a manager to add and remove items and orders from the store.

The website must be easy and intuitive to use. The user interface must have the following criteria:

- follows good website design, make it user friendly for all browsers.
- when data is being entered, check it at the client (browser) level and at the DLL level.
- an about webpage that has details about the company, design, etc.

Test Driven Development

Write unit test to check every function that is in the library. Specifically your unit tests should prove the following:

- library permits use of valid data only (show that invalid data tests fail)
- exceptions work as expected (exceptions thrown on invalid data are valid tests)
- events work as expected (prove that events are only raised at appropriate conditions)

Write unit tests to check every CRUD feature of the ASP.Net MVC 4 website. Specifically your unit tests should prove the following:

- Create, Read, Update, and Delete works with valid and invalid data
- Import of valid and invalid JSON data. When invalid JSON data is found during import prevent it from entering the database.

Your unit tests must prove with confidence that your assignment works according to the business requirements. If you unit tests are meaningless and ambiguous you will lose marks or your assignment might not be accepted.

TDD is based on Agile Methodology of software development. This means you will add new functionality to your business layer by following these steps:

1. Add a test
2. Run and watch the new test fail
3. Write some code
4. Run again and watch all test pass
5. Refactor to remove redundancy

Project report and demonstration

The project report (PDF file) must contain the following items:

- Your name, student ID, screen captures of your program, and a writeup.
- The writeup contains an overview of your assignment
- Include two bullet lists: one for features and another for known issues.
- Answers the following two questions:
  - What did you learn by doing this assignment?
  - What were some of the problems/challenges you had in doing this assignment.
- Your comments and feedback.

Getting A+ or higher

- Grade A+ students should clearly indicate why they think their assignment is worth an A+. Read my feedback for your assignment 1 very carefully.
- Grade A+ students will be contacted by me for an in-person demonstration.

BSD Students

In addition to the above requirements, all BSD students working by themselves or in a group must provide the following additional answers/information:

- Design/Rationalization Summary: What was your contribution to the assignment? What alternatives did you explore before making your choice and why did you do what you did?
- Independent Research: Include at least one item that involved independent research. This has to be something not done in class/labs/course notes. Every student must have their own research contribution.

The answer to both points can be summarized in one, or more, pages but every BSD student must have their own contribution neatly identified. Preference is given to code-based or unit testing based contributions as opposed to theory or implementation of user interface controls.

BSD students who do not have their own contribution attached to the project will not be given credit for the assignment.