Intern Program: Week 5 Assignment

**Purpose:** This document outlines the assignment that will be handed out following the first review for the intern program. This project is to be worked on while you complete the assigned readings for Week 5, and must be completed for your Week 5 review.

# Week 5 Requirements:

This week will require users to build the following elements:

* **StoreFront Data Layer:** Begin refactoring the project into a tiered enterprise architected application.
* **Orders Admin:** Provides administrators with the ability to see all orders in the system.
* **Order Admin Details:** This provides administrators with the ability to review orders and move them down the workflow.
* **Place Order:** This page will handle guiding the user through the process of crafting the order that they are going to be placing.
* **Product Admin Details**: This page will have image upload capability added this week for site administrators.
* **Search / Product Admin:** These pages will be updated to display the image that was uploaded using the Product Admin Details page.
* **GitHub:** Make sure to commit your changes to the repository when you are done.

## StoreFront.Data:

This new project will serve as an application independent data layer for communication between the database and all applications. Beginning with this assignment, no new stored procedures or SQL data sources should be created; further previous data sources should be converted from SQL data sources to object data sources as time allows.

## Orders Admin:

This page provides a listing of all the orders in the application. This will be a grid, which displays 50 rows to a page. The grid will display the following columns:

* OrderID
* Username
* OrderDate
* Total
* Status

The grid will also provide an edit link which will allow users to open a details page displaying all details about an order.

## Order Admin Details:

This page will display all the information relevant to an order. It should be broken up using tabs, and will display the following tabs:

* General
  + Should provide the ability to view a summary page of the order information. This includes the following fields:
    - OrderID
    - Username
    - EmailAddress
    - OrderDate
    - Total
    - Status
  + This tab should also, depending on the status display a button to move it to the next status in the workflow. See the workflow section to define how the workflow should be setup with roles.
* Address
  + This tab should use a user control or partial view to display the billing address of the customer.
* Products
  + This page should display a grid showing the Products that make up the order, showing the following items as columns
    - OrderProductID
    - ProductID
    - ProductName
    - Quantity
    - Price
  + Additiionally the user should have the ability to change the quantity via a textbox in the grid. They should also have a link that allows them to remove items from the order.

## Place Order:

After a user clicks the “Place Order” button, they will be taken to a page where they are asked to complete various steps:

* Step 1 – Address: This screen will display a dropdown list showing all of the addresses for this user. They can select any of those options, or they can select the option in the dropdown of “Enter New Address”. If the user selects this option, a panel on the page will become visible that allows them to enter in a full address. They then can click a “Save” button in the lower right hand corner to move to the next screen.
* Step 2 – Confirm Order: Finally this screen will show in read-only form, the following information.
  + Address
  + Grid of the order contents

Additionally it will provide a button “Place Order” that allows the user to place their order. Afterward, it would provide a confirmation page.

* Step 3 – Order Submitted: A screen that is displaying a confirmation message about an order being submitted.

# Technical Requirements:

The following are the technical requirements for each page in this assignment: Below is a listing of the approaches that are to be used to satisfy the business requirements above.

## StoreFront.Data:

A new project will be added to the solution, called “StoreFront.Data”, and this will be a class library. This will handle the data operations for the remainder of this project. A reference to this class library should be added to your existing StoreFront project.

Inside this project, a new EntityFramework edmx or “ADO.NET Entity Data Model” should be added. And inside that data model, all of the existing tables within the database should be added.

Additionally the following classes should be created within the StoreFront.Data project.

* InventoryRepository
  + SearchProducts(string searchText): This should return a list of class “Product”, that have name or description containing the attached search text.
  + GetProduct(int id): This should return an instance of the custom class “Product”, which will have all the fields in the “Product” class in entity framework. This will be populated inside the method and returned.
* OrderRepository
  + GetOrder(int id): Should return an instance of the “Order” object, which is the row in the table with the corresponding id. It should use entity framework to retrieve this.

## Orders Admin:

The following are the requirements for this page for this week:

* This page may be developed in MVC or Windows Forms.
* This page will provide a grid that will be populated using the newly created data layer.
* All grid columns are annotated above.

## Order Admin Details:

The following are the requirements for this page for this week:

* If using Windows Forms:
  + This page will use the AJAX control toolkit’s tab container to handle the tabs.
  + The UserControl for addresses will be called “DisplayAddress.ascx”.
* If using MVC:
  + This page will utilize jQuery UI to create tabs
  + The address should be a partial view in the Views/Shared folder, or a subfolder thereof.

## Place Order:

The following are the requirements for this page for this week:

* If using Windows Forms:
  + This page will use panels to enclose each step in the wizard.
  + Update panels should be utilized to make postbacks completely transparent to the end user.
* If using MVC:
  + This page will make use of hidden sections and JavaScript to make data transfer transparent to the user.

## Product Admin Details:

The following changes will be made to the ProductAdminDetails.aspx page this week:

* A file upload control will be added to populate the ImageFile control on the Product table. This should store just the name of the image, and the file itself will be saved inside a folder in the project called “ProductImages”.

## Search / Products Admin:

The following changes will be made to the Search and ProductsAdmin pages this week:

* The grids will be updated to utilize the load the images for users to see within the grids, using the files saved to “ProductImages”.

## GitHub

Ensure that all your changes are committed to your GitHub repository for review.