Department of Computer Science & Informatics
CSIS3734 (2021)

Assignment 2: Basic CRUD Functionality for SportsStore app

Submission deadline: Monday, 17 May 2021 @ 14h30

No late submissions will be accepted

Background Information

For this assignment you are going to add the following administration features to the existing SportsStore application:

- View list of all products.
- Add a new product.
- Edit product details.
- Delete a product.
- Client-side validation for the Administration section.
- Make use of a TempData variable to display relevant dismissable alert messages on the Administration layout page.
- A Razor Page to:
  - View a list of all orders (unshipped and shipped)
  - Mark an unshipped order as “shipped”.
- Fully functional navigation bar on the Home Page.

Please read the specifications carefully and make sure that you follow all the instructions specified in this document exactly.

Basic Requirements

- Your web app must be created using the ASP.NET Core 3.1 MVC framework.
- You must use Bootstrap 4 to set the presentation aspects of the application.
- Your personal details must be displayed in the footer of all pages.
- Use Microsoft Edge to test your application.
- This is an individual assignment. You are not allowed to collaborate with anyone or ask anyone other than the CSIS3734 lecturer or assistants for help.

Preparation

1. Download the **`Assig2_StudentFiles.zip`** file from Blackboard. This file contains the following:
   - **`Appendix.pdf`:** A document containing screen captures of all the completed pages.
   - **SportsStore app**: The initial SportsStore application to be used as the starting point for this assignment.
   - **CodeSnippets folder**: Basic markup for the new views you have to create.
2. Rename the application folder `SportsStore_Assign2_Surname_Initials` by replacing the "Surname" and "Initials" parts with your own details.

Model

Add the necessary validation attributes to the **Product** domain model to set the following:

- Required fields: Name, Description, Price and CategoryID
- Price must contain a value between `0.01` and `double.MaxValue`.
- Display name for CategoryID must be "Category".
- Add the necessary error messages (see **`Appendix.pdf`**).

Please note the following:

- You are NOT allowed to make <u>any other changes</u> to the files in the **Models** folder.
- You are also NOT allowed to make any changes to files in the **Data** folder.

Controllers

Make the necessary changes to the **Admin Controller** to implement all the required CRUD functionalities for Products. (Refer to the Background Section of this document as well as **`Appendix.pdf`** for more details in this regard).

Please note the following:

- For these implementations you must use the existing **RepositoryWrapper** and the generic repository methods whenever possible.
- Make use of try/catch blocks to handle all errors caused during changes to the database.

Razor Views

Create all the new views requested by the **Admin Controller**.

Other requirements:

- All new views must use the _AdminLayout.
- Implement client-side validation for the _AdminLayout.
- The markup to display the dismissible alert messages must be added to the _AdminLayout.
- Use the same view to Edit and Add products. (No partial views allowed here.)
- Make use of strongly-typed views where possible.
- Make use of the built-in tag helpers where possible.

Razor Page

Create a new Razor Page that can be reached via the URL "/Orders". This page should display a list of all Unshipped and Shipped orders (together with the details orfeach order). The user should be able to mark an Unshipped order as Shipped. Refer to **`Appendix.pdf`** for additional details regarding this page.

Other requirements:

- The Orders link in the drop-down menu on the main navigation bar must be linked to this page.
- You must use the existing **RepositoryWrapper** and the generic repository methods.
- This page must use the existing _CartLayout.
- You are not allowed to use any ViewModel classes as part of the Page model**.**
- The Page model class should only contain three methods: OnGet() - to handle the Get Request when the page is displayed; OnPost() - to handle the Post Request when the user selects the "Ship" button, and a method to populate the class properties for Unshipped and Shipped orders.

Submission

1. Create a **Compressed Zip Folder** of your entire application folder. (**Note:** Make sure that you have renamed your application folder as instructed in the Preparation section of this document.)

   Please note the following:

   - The filename of your zip folder should be **`SportsStore_Assign2_`*`SurnameIntials`*`.zip`**
   - You will not be submitting your SQL Server database file.

2. Submit your ZIP folder on the **"Submit Assignment 2"** page on Blackboard before the submission deadline. (See "**Practical Assignments**" page of Study Unit 3).

END OF ASSIGNMENT