300583 Web Systems Development

PRACTICAL SET 1 DUE: 9PM 21 AUG 2020

In Practical Set 1, you will create a simple website by ASP.NET Core Razor pages for a fictitious pizza shop named "Gourmet Pizza". This pizza shop sells four types of pizzas, which are detailed in the following table:

Pizza Type	Pizza Name	Price
1	BBQ Beef	$10.50
2	Chicken and Pineapple	$8.50
3	Pepperoni Feast	$9.00
4	Vegetarian	$7.00

Table 1: Pizza Details

Steps for completing this practical set

1. Create an ASP.NET Core Razor Pages project named "GourmetPizza" with Visual Studio.
2. In 'Solution Explorer', create a folder named "PizzaTest" under the "Pages" folder of your project.
3. Create a Razor page named "Index" under the "PizzaTest" folder. Modify the automatically-generated 'Index.cshtml' file to display the Table 1 above in an HTML table. That is, when entering the following URL https://localhost:xxx/PizzaTest into the browser after running this project, the Table 1 above should be displayed in the browser.
4. In 'Solution Explorer', create a folder named 'images' under the 'wwwroot' folder of your project. Then, create an image file for each of the four pizza types and store these four files under the 'images' folder. You can download images from Internet and then modify them. When searching images on the Internet, you should use royalty-free image websites such as Pixabay (www.pixabay.com) and Unsplash (www.unsplash.com) to avoid copyright issues. The four images should all:
    - Have a resolution of 800 X 450 pixels.
    - Be in one image format such as jpeg, bmp, png, etc.
    - Roughly illustrate their corresponding pizza types.

Hint: "wwwroot" is the folder for storing static contents such as images, js, css, etc of your project. This folder can be referred to by the "~" character in URL. For example, the <img> element to display the image "wwwroot/images/xxx.jpg" should be written as in HTML document.

5. Create another Razor page named 'Purchase' under the 'PizzaTest' folder. This page should process the following kind of queries:

    https://localhost:xxx/PizzaTest/Purchase?PizzaType=3&PizzaCount=4

This processing requires you to modify the OnGet() function in the 'Purchase.cshtml.cs' file as follows:

a. Add two function parameters: int PizzaType and int PizzaCount
b. Calculate the total price for purchasing pizzas of PizzaType with the quantity of PizzaCount. The pricing should be based on the Table 1 above.
c. Pass the following values to the content file 'Purchase.cshtml' by the ViewData dictionary:
    i. The pizza name corresponding to the PizzaType based on Table 1.
    ii. The total price to pay.

Hint: the easiest way to implement this function is to use the "switch" statement in C#. See https://www.w3schools.com/cs/cs_switch.asp.

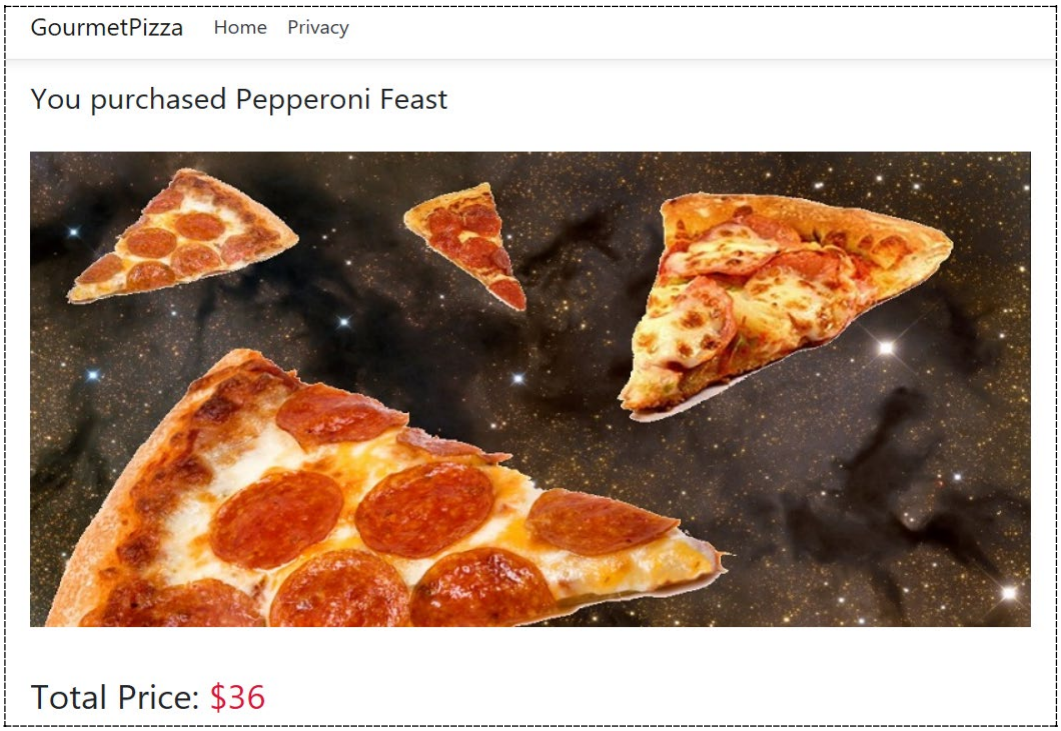
6. Modify the content file 'Purchase.cshtml' such that it can:
    a. Display the name of the pizza purchased in an <h4> heading.
    b. Display the image corresponding to that pizza name underneath the pizza name.
    c. Display the total price to pay in an <h3> heading below the pizza image.

Hint: You need to embed C# code into the content file with Razor syntax. Again a 'switch' statement gives you the most efficient implementation. Inside this 'switch' statement, you decide the image file name for the pizza and store it into a string variable, say, 'imagefile'. Then, in the <img> tag, you can do `<img src="~/images/@imagefile"/>`.

Also, look at our sample project for Lecture 2.

An exemplar testing result for step 6

Enter the URL https://localhost:xxx/PizzaTest/Purchase?PizzaType=3&PizzaCount=4, and the following should be roughly displayed:



7. Modify the "Index" Razor page directly under the "Pages" folder to implement the following:
    a. Add a carousel that uses the four pizza images created in Step 4. (hint: to make the images occupy the full width, you need to add a CSS style "width:100%" for the images; you can refer to our sample project in lecture 3)
    b. The images should be rotated every 3 seconds. (hint: you can use the "data-interval" attribute of carousel to achieve this)
    c. The captions should indicate the type of the pizza.
8. Create a Model class for pizzas. The class name should be "Pizza". The properties of this class should reflect the columns in Table 1. The property names are of your choice. For example, you can use a property named either "ID" or "PizzaID" to match the "Pizza Type" column.
9. Create a 'Pizzas' folder under the 'Pages' folder. Then, use scaffolding to create the Razor pages for CRUD operations on the Pizza database table under the 'Pizzas' folder.
10. Modify the ConfigureServices() method in Startup.cs file such that SQLite is injected as the DBMS instead of the SQL Server LocalDB. The name of the database file should be "PizzaShop.db".
11. Use migration to create the PizzaShop.db as the database.
12. Modify the "_Layout.cshtml" file to add a hyperlink with the text "Pizza Index" into the navigation bar. This hyperlink should point to the Index page under the 'Pizzas' folder. (hint: refer to our sample project accompanying lecture 4).
13. Modify the "Index" Razor page directly under the "Pages" folder further to implement the following:
    a. The area beneath the carousel is divided into two combined columns. The first combined column consists of 8 columns in the 12-column grid system, and the second combined column consists of the remaining 4 columns.
    b. The two combined columns should be stacked vertically if the screen width is less than 768px.
    c. The first combined column should contain a welcome message to this Gourmet Pizza shop; the second combined column should contain the following links arranged vertically in a bullet list.
        - Home: link to the Index page under 'Pages' folder.
        - Pizzas Index: link to the Index page under the 'Pages/Pizzas' folder.
        - Privacy: link to the Privacy page under the 'Pages' folder.
14. Run the GourmetPizza project and use the web interface from this project to add the data in Table 1 into the database.

An exemplar testing result after completing this practical set

Click the "Pizzas Index" link in the navigation bar, the following should appear roughly. Note that since the Pizza ID column is taken as the primary key, it won't appear by default.

Checkpoint (2 marks)

The Prac1 Checkpoint will occur in your registered practical class in week 3. At this checkpoint you should complete steps 1-3.

Marking criteria:

- Steps 1 and 2: 1 mark.
- Step 3: 1 mark.

Any mistake in a bullet point above will results in a zero mark for that bullet point. No decimal mark is available.

Final Submission (12 marks)

Please follow the steps below to submit your completed Prac1 to vUWS:

- Zip up the entire GourmetPizza folder and its sub-folders, which contain all your files for your project.
- Name the ZIP file *Surname_studentid*_Prac1.zip
- Submit the zip file:
  - In VUWS, go to 'Practicals'
  - Click on 'Prac1 Submission'; this link will be made available closer to the deadline.
  - Attach your zip file and submit

Marking Criteria: Please see the rubric for Prac1 in the **My Grades** page on vUWS.

Notes:

- You can submit the zip file multiple times until the due time. The latest submission will be marked.
- Work submitted after the due time will incur a 10% per day late penalty as per university policy **https://policies.westernsydney.edu.au/document/view.current.php?id=227**
- **Based on the policy, the 10% above means the 10% of the worth of the assessment, not your final score. So if the worth of the assessment is 10 marks and you scored 5 marks with a 5-day lateness, then you'll receive 0 mark.**