



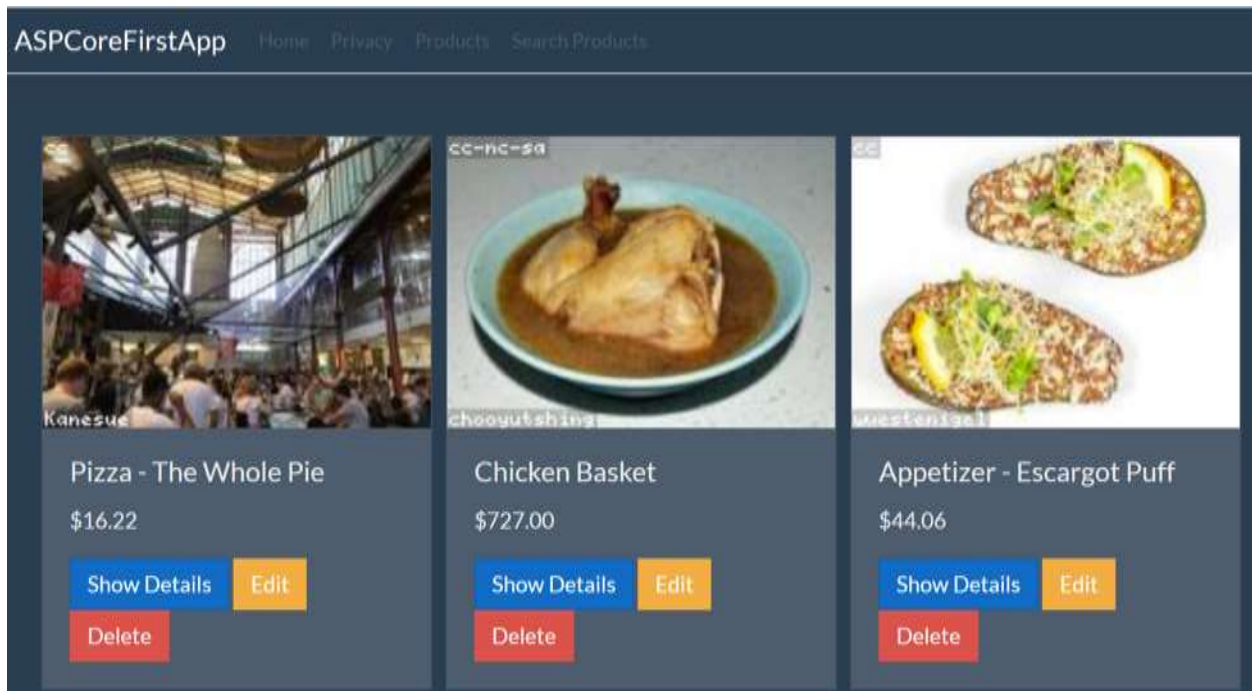
Part 4 Bootstrap Modal

Goal:

Show the details of a product and perform the edit function on the same page, without having to reload or show a new view. We will use AJAX updates and a **Modal dialog box** in the Products App

Review Current State

This activity continues with the Products Application. After completing the previous tutorial, your products app should have a gallery of products with a **Details**, **Edit** and **Delete** button on each card.

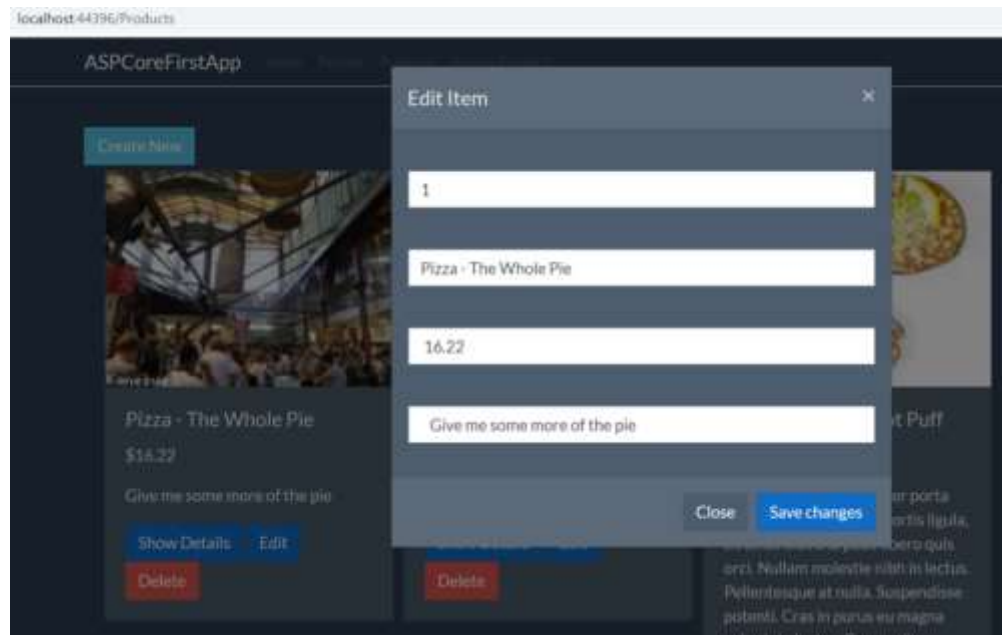


Edit Modal

The objective is to display the details of an item on a popup window without having to jump to another page.

A **modal** is like a dialog box or popup box where the focus of the application is centered on a **data entry form** or yes-no-cancel type of input.

To accomplish a modal in a web page, some fancy trickery in CSS is required. Fortunately, Bootstrap does all of this CSS work for us. We mostly copy and paste from their example code. We will notice several “modal” Bootstrap class names in the code.

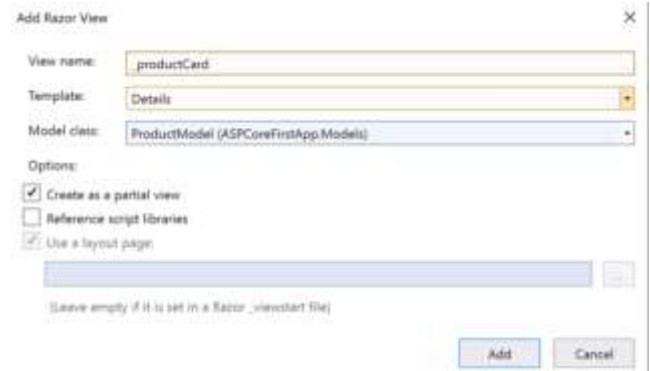


How it works

- **Modals** are built with HTML, CSS, and JavaScript. They're positioned over everything else in the document and remove scroll from the <body> so that modal content scrolls instead.
- Clicking on the modal “backdrop” will automatically close the modal.
- Bootstrap only supports one modal window at a time.
- Modals use **position: fixed**, which can sometimes be a bit particular about its rendering. Whenever possible, place your modal HTML in a top-level position to avoid potential interference from other elements. You'll likely run into issues when nesting a **.modal** within another fixed element.
- Due to **position: fixed**, there are some caveats with using modals on mobile devices.

Instructions

1. Create a new **partial view** inside the Views > Products folder.
 - a. Name the View **_productCard**. The underscore character **_** is a common convention to indicate this is a partial view.
 - b. Use the **Details** template
 - c. Choose **ProductModel** as the Model class.
2. From the **Index.cshtml** file, copy the entire `<div id="prod-card">` section. We will replace most of the HTML code of the **_productCard** file with this version of product details.



```
Index.cshtml*  _productView.cshtml  ProductController.cs
1  @model IEnumerable<Activity2.Models.ProductModel>
2
3  <a href="/product/showcreateform" >Create new product</a>
4
5  <div class="container d-flex flex-wrap">
6      @foreach (var item in Model)
7      {
8          var s = item.Name;
9
10         var firstWord = s.IndexOf(" ") > -1 ? s.Substring(0, s.IndexOf(" ")) : s;
11         <div class="card" style="width: 18rem;">
12             
13             <div class="card-body">
14                 <h5 class="card-title">@item.Name</h5>
15                 <p class="card-text">@Html.DisplayFor(modelItem => item.Price)</p>
16                 <a href="/product/ShowOneProduct/@item.Id" class="btn btn-primary">Show Details</a>
17                 <a href="/product/ShowEditForm/@item.Id" class="btn btn-warning">Edit</a>
18                 <a href="/product/DeleteOne/@item.Id" class="btn btn-danger">Delete</a>
19             </div>
20         </div>
21     }
22 </div>
23
24
25
```

3. Paste the contents into the **_productCard** file and change all of the **@Item** statements to **@Model** ("Model" not "Modal")

```
Index.cshtml*  _productView.cshtml  ProductController.cs
1  @model Activity2.Models.ProductModel
2  @{
3      var s = @Model.Name;
4
5      var firstWord = s.IndexOf(" ") > -1 ? s.Substring(0, s.IndexOf(" ")) : s;
6  }
7
8  <div class="card" style="width: 18rem;">
9      
10     <div class="card-body">
11         <h5 class="card-title">@Model.Name</h5>
12         <p class="card-text">@Html.DisplayFor(modelItem => Model.Price)</p>
13         <a href="/product/ShowOneProduct/@Model.Id" class="btn btn-primary">Show Details</a>
14         <a href="/product/ShowEditForm/@Model.Id" class="btn btn-warning">Edit</a>
15         <a href="/product/DeleteOne/@Model.Id" class="btn btn-danger">Delete</a>
16     </div>
17 </div>
```

The model for this view is only one product. Change **item** to **model**.

4. Modify the **Index** View to include the new View as a partial page insert.

```
Index.cshtml  _productCard.cshtml  ProductController.cs
1  @model IEnumerable<Activity2.Models.ProductModel>
2
3  <a href="/product/showcreateform" >Create new product</a>
4
5  <div class="container d-flex flex-wrap">
6      @foreach (var item in Model)
7      {
8
9          <partial name="_productCard" model="item" />
10
11      }
12  </div>
13
```

5. Run the program. The application should continue to work as before. We implemented a partial view in preparation for partial page updates which will be done via AJAX.
6. From **Bootstrap**, copy the example code for a modal dialog box. Be sure to use Bootstrap version 4.X

getbootstrap.com/docs/4.0/components/modal/

Home Documentation Examples Themes Expo Blog

Search...

Getting started
Layout
Content

Components

- Alerts
- Badge
- Breadcrumb
- Buttons
- Button group
- Card
- Carousel
- Collapse
- Dropdowns
- Forms
- Input group
- Jumbotron
- List group
- Modal
- Navs
- Navbar
- Pagination
- Popovers
- Progress

Live demo

Toggle a working modal demo by clicking the button below. It will slide down and fade in from the top of the page.

Launch demo modal

```
<!-- Button trigger modal -->
<button type="button" class="btn btn-primary" data-toggle="modal" data-target="#exampleModal">
  Launch demo modal
</button>

<!-- Modal -->
<div class="modal fade" id="exampleModal" tabindex="-1" role="dialog" aria-labelledby="exampleModalLabel" aria-hidden="true">
  <div class="modal-dialog" role="document">
    <div class="modal-content">
      <div class="modal-header">
        <h5 class="modal-title" id="exampleModalLabel">Modal title</h5>
        <button type="button" class="close" data-dismiss="modal" aria-label="Close">
          <span aria-hidden="true">&times;</span>
        </button>
      </div>
      <div class="modal-body">
        ...
      </div>
      <div class="modal-footer">
        <button type="button" class="btn btn-secondary" data-dismiss="modal">Close</button>
        <button type="button" class="btn btn-primary">Save changes</button>
      </div>
    </div>
  </div>
</div>
```

Copy this section later.

Copy this section now.

7. Add the copied code to the bottom of the **Index** View. Bootstrap recommends putting modal code first on the page, to avoid naming conflicts with id numbers. However, it works as shown here. This modal section remains hidden until the Edit button of a product is clicked.



```
1 using ASPCoreFirstApp.Models
2 @model IEnumerable<ASPCoreFirstApp.Models.ProductModel>
3
4 <a class="btn btn-info" href="@Url.Action("ShowCreate", "Products")">Create New</a>
5
6 <div id="modal-placeholder"></div>
7
8 <div class="container d-flex flex-wrap">
9     @foreach (var item in Model)
10     {
11         <div id="card-number-@item.Id">
12             <partial name="_productCard" model="item" />
13         </div>
14     }
15 </div>
16
17 <!-- Modal -->
18 <div class="modal fade" id="exampleModal" tabindex="-1" role="dialog" aria-labelledby="exampleModallabel" aria-hidden="true">
19     <div class="modal-dialog" role="document">
20         <div class="modal-content">
21             <div class="modal-header">
22                 <h5 class="modal-title" id="exampleModallabel">Modal title</h5>
23                 <button type="button" class="close" data-dismiss="modal" aria-label="Close">
24                     <span aria-hidden="true">&times;</span>
25                 </button>
26             </div>
27             <div class="modal-body">
28                 ...
29             </div>
30             <div class="modal-footer">
31                 <button type="button" class="btn btn-secondary" data-dismiss="modal">Close</button>
32                 <button type="button" class="btn btn-primary">Save changes</button>
33             </div>
34         </div>
35     </div>
36 </div>
37 </div>
```

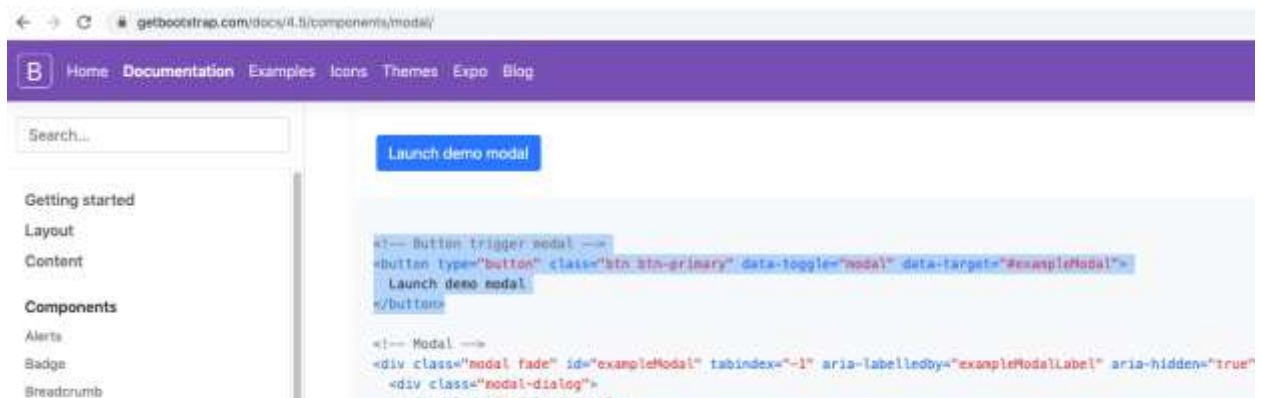
8. Rename the `<div class>` to `editForm`.


```

1  @model IEnumerable<Activity2.Models.ProductModel>
2
3  <a href="/product/showcreateform">Create new product</a>
4
5  <div class="container d-flex flex-wrap">
6      @foreach (var item in Model)
7      {
8
9          <partial name="_productCard" model="item" />
10
11      }
12  </div>
13
14  <!-- Modal -->
15  <div class="modal fade" id="editForm" tabindex="-1" aria-labelledby="exampleModallabel" aria-hidden="true">
16      <div class="modal-dialog">
17          <div class="modal-content">
18              <div class="modal-header">
19                  <h5 class="modal-title" id="exampleModallabel">Modal title</h5>
20                  <button type="button" class="close" data-dismiss="modal" aria-label="Close">
21                      <span aria-hidden="true">&times;</span>
22                  </button>
23              </div>
24              <div class="modal-body">
25                  ...
26              </div>
27              <div class="modal-footer">
28                  <button type="button" class="btn btn-secondary" data-dismiss="modal">Close</button>
29                  <button type="button" class="btn btn-primary">Save changes</button>
30              </div>
31          </div>
32      </div>
33  </div>

```

9. Copy the button text from the Bootstrap example.



10. Paste into the `_productCard` view and rename the `data-target` to **editForm** in order match the modal form..

```

Index.cshtml | productCard.cshtml | ProductController.cs
1  @model Activity2.Models.ProductModel
2  @{
3      var s = @Model.Name;
4
5      var firstWord = s.IndexOf(" ") > -1 ? s.Substring(0, s.IndexOf(" ")) : s;
6  }
7
8  <div class="card" style="width: 18rem;">
9      
10     <div class="card-body">
11         <h5 class="card-title">@Model.Name</h5>
12         <p class="card-text">@Html.DisplayFor(modelItem => Model.Price)</p>
13         <a href="/product/ShowOneProduct/@Model.Id" class="btn btn-primary">Show Details</a>
14         <a href="/product/ShowEditForm/@Model.Id" class="btn btn-warning">Edit</a>
15         <a href="/product/DeleteOne/@Model.Id" class="btn btn-danger">Delete</a>
16
17         <!-- Button trigger modal -->
18         <button type="button" class="btn btn-primary" data-toggle="modal" data-target="#editForm">
19             Launch demo modal
20         </button>
21
22     </div>
23 </div>

```

11. From the ShowEditForm page copy the four sections dealing with the data input fields.

```

ShowEditForm.cshtml | Index.cshtml | productCard.cshtml | ProductController.cs
1  @model Activity2.Models.ProductModel
2
3  <h4>ProductModel</h4>
4  <hr />
5  <div class="row">
6      <div class="col-md-4">
7          <form asp-action="ProcessEdit">
8              <div asp-validation-summary="ModelOnly" class="text-danger"></div>
9              <div class="form-group">
10                 <label asp-for="Id" class="control-label"></label>
11                 <input asp-for="Id" class="form-control" />
12                 <span asp-validation-for="Id" class="text-danger"></span>
13             </div>
14             <div class="form-group">
15                 <label asp-for="Name" class="control-label"></label>
16                 <input asp-for="Name" class="form-control" />
17                 <span asp-validation-for="Name" class="text-danger"></span>
18             </div>
19             <div class="form-group">
20                 <label asp-for="Price" class="control-label"></label>
21                 <input asp-for="Price" class="form-control" />
22                 <span asp-validation-for="Price" class="text-danger"></span>
23             </div>
24             <div class="form-group">
25                 <label asp-for="Description" class="control-label"></label>
26                 <input asp-for="Description" class="form-control" />
27                 <span asp-validation-for="Description" class="text-danger"></span>
28             </div>
29             <div class="form-group">
30                 <input type="submit" value="Create" class="btn btn-primary" />
31             </div>
32          </form>
33      </div>
34  </div>
35
36  <div>
37      <a asp-action="Index">Back to List</a>
38  </div>
39

```

12. Paste the contents into the center area of the modal body, replacing the triple dot placeholder (...)

```
ShowEditForm.cshtml | Index.cshtml | productCard.cshtml | ProductController.cs
1  @model IEnumerable<Activity2.Models.ProductModel>
2
3  <a href="/product/showcreateform">Create new product</a>
4
5  <div class="container d-flex flex-wrap">
6      @foreach (var item in Model)
7      {
8          <partial name="_productCard" model="item" />
9      }
10 </div>
11
12 <!-- Modal -->
13
14 <div class="modal fade" id="editForm" tabindex="-1" aria-labelledby="exampleModalLabel" aria-hidden="true">
15     <div class="modal-dialog">
16         <div class="modal-content">
17             <div class="modal-header">
18                 <h5 class="modal-title" id="exampleModalLabel">Modal title</h5>
19                 <button type="button" class="close" data-dismiss="modal" aria-label="Close">
20                     <span aria-hidden="true">&times;</span>
21                 </button>
22             </div>
23             <div class="modal-body">
24                 <div class="form-group">
25                     <label asp-for="Id" class="control-label"></label>
26                     <input asp-for="Id" class="form-control" />
27                     <span asp-validation-for="Id" class="text-danger"></span>
28                 </div>
29                 <div class="form-group">
30                     <label asp-for="Name" class="control-label"></label>
31                     <input asp-for="Name" class="form-control" />
32                     <span asp-validation-for="Name" class="text-danger"></span>
33                 </div>
34                 <div class="form-group">
35                     <label asp-for="Price" class="control-label"></label>
36                     <input asp-for="Price" class="form-control" />
37                     <span asp-validation-for="Price" class="text-danger"></span>
38                 </div>
39                 <div class="form-group">
40                     <label asp-for="Description" class="control-label"></label>
41                     <input asp-for="Description" class="form-control" />
42                     <span asp-validation-for="Description" class="text-danger"></span>
43                 </div>
44             </div>
45             <div class="modal-footer">
46                 <button type="button" class="btn btn-secondary" data-dismiss="modal">Close</button>
47                 <button type="button" class="btn btn-primary">Save changes</button>
48             </div>
49         </div>
50     </div>
51 </div>
52
```

13. Rename the **asp-for** tags to **for** tags. Remove the **asp-validation** lines. Give each input an id value as shown.


```

13
14 <!-- Modal -->
15 <div class="modal fade" id="editForm" tabindex="-1" aria-labelledby="exampleModallabel" aria-hidden="true">
16   <div class="modal-dialog">
17     <div class="modal-content">
18       <div class="modal-header">
19         <h5 class="modal-title" id="exampleModallabel">Modal title</h5>
20         <button type="button" class="close" data-dismiss="modal" aria-label="Close">
21           <span aria-hidden="true">&times;</span>
22         </button>
23       </div>
24       <div class="modal-body">
25         <div class="form-group">
26           <label for="Id" class="control-label"></label>
27           <input id="modal-input-id" for="Id" class="form-control" />
28         </div>
29         <div class="form-group">
30           <label for="Name" class="control-label"></label>
31           <input id="modal-input-name" for="Name" class="form-control" />
32         </div>
33         <div class="form-group">
34           <label for="Price" class="control-label"></label>
35           <input id="modal-input-price" for="Price" class="form-control" />
36         </div>
37         <div class="form-group">
38           <label for="Description" class="control-label"></label>
39           <input id="modal-input-description" for="Description" class="form-control" />
40         </div>
41       </div>
42       <div class="modal-footer">
43         <button type="button" class="btn btn-secondary" data-dismiss="modal">Close</button>
44         <button id="save-button" type="button" class="btn btn-primary btn-secondary" data-dismiss="modal">Save changes</button>
45       </div>
46     </div>
47   </div>
48 </div>
49
50
51
52

```



Diagram illustrating the removal of the input fields from the modal form. Three red boxes highlight the input fields, and blue arrows point to them with the label "removed".

14. Give the "Save changes" button an id property. This will allow us to reference the item in the javascript code.
15. Add the data-dismiss="modal" property to the "save changes" button. This will close the modal dialog when save is clicked.

```

43
44 </div>
45 </div>
46 <div class="modal-footer">
47   <button type="button" class="btn btn-secondary" data-dismiss="modal">Close</button>
48   <button id="save-button" type="button" class="btn btn-primary btn-secondary" data-dismiss="modal">Save changes</button>
49 </div>
50 </div>
51 </div>
52

```

16. Add a class name and value to the new button in the _productCard view as shown below.
17. Add an id value for the card as shown below. Further along, this will allow us to add a JavaScript click listener to the buttons and identify which item number was clicked.

```

1  @model Activity2.Models.ProductModel
2
3  @{
4      var s = @Model.Name;
5
6      var firstWord = s.IndexOf(" ") > -1 ? s.Substring(0, s.IndexOf(" ")) : s;
7
8      <div id="card-number-@Model.Id" class="card" style="width: 18rem;">
9          
10         <div class="card-body">
11             <h5 class="card-title">@Model.Name</h5>
12             <p class="card-text">@Html.DisplayFor(modelItem => Model.Price)</p>
13             <a href="/product/ShowOneProduct/@Model.Id" class="btn btn-primary">Show Details</a>
14             <a href="/product/ShowEditForm/@Model.Id" class="btn btn-warning">Edit</a>
15             <a href="/product/DeleteOne/@Model.Id" class="btn btn-danger">Delete</a>
16
17             <!-- Button trigger modal -->
18             <button type="button" class="btn btn-primary edit-product-button" value="@Model.Id" data-toggle="modal" data-target="#editForm">
19                 Launch demo modal
20             </button>
21
22         </div>
23     </div>

```

18. Add a new method, **ShowOneProductJSON**, to the **Products Controller** that will return a JSON-formatted version of a product model.

```

36
37 0 references
38  public IActionResult ShowOneProduct(int Id)
39  {
40      return View(repository.GetProductById(Id));
41  }
42
43 0 references
44  public IActionResult ShowOneProductJSON(int Id)
45  {
46      return Json(repository.GetProductById(Id));

```

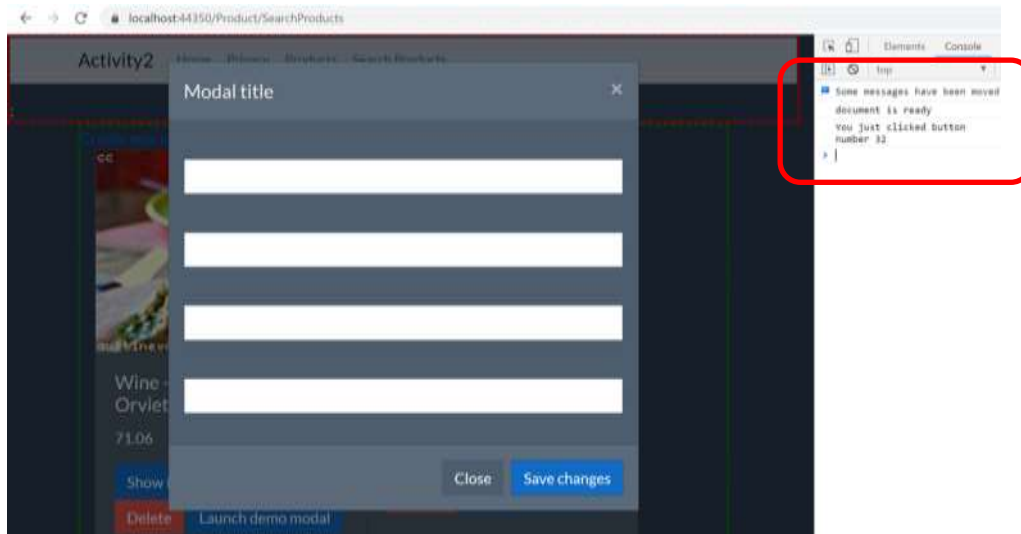
19. Add the following JavaScript code to **site.js** (inside wwwroot > js).

```

site.js  ShowEditForm.cshtml  Index.cshtml  _productCard.cshtml
Activity2 JavaScript Content Files
1  $(function() {
2      console.log("document is ready");
3      $(document).on("click", ".edit-product-button", function () {
4          console.log("You just clicked button number " + $(this).val());
5      });
6  });

```

20. Confirm that the button click function is working properly by looking at the console log messages. Each button should have the value of the product id.



21. Add the following ajax command in order to fetch a product from the controller.



22. Confirm that the edit button is fetching a product in JSON format by viewing the console log.

```

document is ready      site.js?v=w32ysSmaql...XQUuDxoMegUXUXI0:2
You just clicked button site.js?v=w32ysSmaql...XQUuDxoMegUXUXI0:4
number 49

      site.js?v=w32ysSmaql...XQUuDxoMegUXUXI0:14
{id: 49, name: "Beer - Rickards Red",
price: 12.51, description: "Fusce lacus purus, aliquet at, feugiat non, pretium ..."}
description: "Fusce lacus purus, aliquet at, feugiat non, pretium ...
id: 49
name: "Beer - Rickards Red"
price: 12.51

```

23. Modify the site.js file to fill the modal entry form with values from the data.

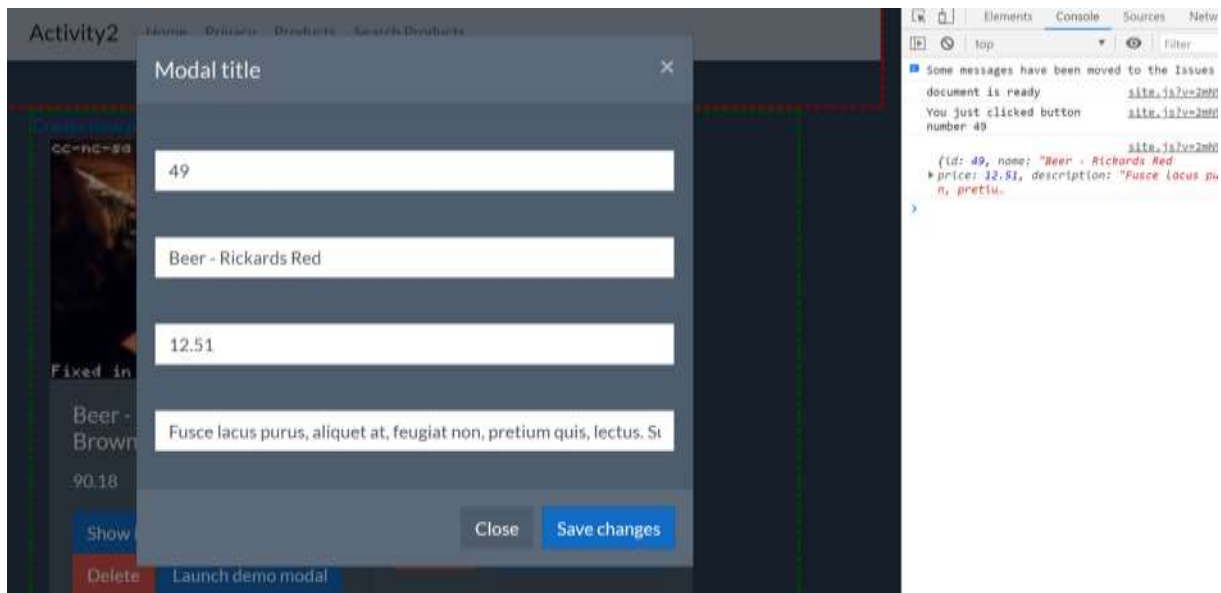
```

site.js*  ShowEditForm.cshtml  Index.cshtml  _productCard.cshtml
Activity2 JavaScript Content Files

1  $(function() {
2      console.log("document is ready");
3      $(document).on("click", ".edit-product-button", function () {
4          console.log("You just clicked button number " + $(this).val());
5
6          // fetch the product information and fill the modal form.
7          var productId = $(this).val();
8          $.ajax({
9              type: 'json',
10             data: {
11                 "id": productId
12             },
13             url: "/product/ShowOneProductJSON",
14             success: function (data) {
15                 console.log(data);
16
17                 // fill the input fields in the modal
18
19                 $("#modal-input-id").val(data.id);
20                 $("#modal-input-name").val(data.name);
21                 $("#modal-input-price").val(data.price);
22                 $("#modal-input-description").val(data.description);
23             }
24         });
25     });
26
27

```


24. Test the buttons. You should see the form filled with the information that corresponds to the product values.




25. Add the following to the site.js file in order to capture the values stored in the modal form.

```

1  $(function() {
2      console.log("document is ready");
3      $(document).on("click", ".edit-product-button", function () {
4          console.log("You just clicked button number " + $(this).val());
5
6          // fetch the product information and fill the modal form.
7          var productId = $(this).val();
8          $.ajax({
9              type: 'json',
10             data: {
11                 "id": productId
12             },
13             url: "/product/ShowOneProductJSON",
14             success: function (data) {
15                 console.log(data);
16
17                 // fill the input fields in the modal
18
19                 $("#modal-input-id").val(data.id);
20                 $("#modal-input-name").val(data.name);
21                 $("#modal-input-price").val(data.price);
22                 $("#modal-input-description").val(data.description);
23             }
24         });
25     });
26
27
28     $("#save-button").click(function () {
29         // get the values of the input fields and make a product JSON object.
30         var Product = {
31             "Id": $("#modal-input-id").val(),
32             "Name": $("#modal-input-name").val(),
33             "Price": $("#modal-input-price").val(),
34             "Description": $("#modal-input-description").val()
35         }
36         console.log("saved:");
37         console.log(Product);
38     });
39 });

```

26. Run the program and confirm that the data is being captured from the modal by looking in the console messages.



Soup - Campbells Chili Veg

1.37

Show Details Edit

Delete Launch demo modal

Some messages have been moved to the Issues panel.

document is ready [site.js?v=zTZlwG0hw2...-7no](#)

You just clicked button number 63 [site.js?v=zTZlwG0hw2...-7no](#)

```
{id: 63, name: "Soup - Campbells Chili Veg", price: 1.37, description: "Pellentesque viverra pede, Pellentesque..."}
saved: site.js?v=zTZlwG0hw2...-7no
```

```
{Id: "63", Name: "Soup - Campbells Chili Veg", Price: "1.37", Description: "Pellentesque viverra, as pellentesq..."}
```

27. Add a new method in the product controller to update a product and return a partial view.

```

52 0 references
53 public IActionResult ProcessEdit(ProductModel product)
54 {
55     repository.Update(product);
56     return View("Index", repository.AllProducts());
57 }
58 0 references
59 public IActionResult ProcessEditReturnPartial(ProductModel product)
60 {
61     repository.Update(product);
62     return PartialView("_productCard", product);
63 }

```

28. Test the application to verify that the updated data is being captured.

```

{id: 119, name: "Flour - Fast / Rapid", price: 21.48, description: "Donec dapibus. Duis at velit eu est congue elementum..."}
saved: {Id: "119", Name: "Flour", Price: "21.48", Description: "Donec dapibus. Duis at velit eu est congue elementum..."}

```

Before Edit

After Edit

29. Finally, save the update to the database and insert the data into the proper card.

```

1  $(function() {
2      console.log("document is ready");
3      $(document).on("click", ".edit-product-button", function () {
4          console.log("You just clicked button number " + $(this).val());
5
6          // fetch the product information and fill the modal form.
7          var productId = $(this).val();
8          $.ajax({
9              type: 'json',
10             data: {
11                 "id": productId
12             },
13             url: "/product/ShowOneProductJSON",
14             success: function (data) {
15                 console.log(data);
16
17                 // fill the input fields in the modal
18
19                 $("#modal-input-id").val(data.id);
20                 $("#modal-input-name").val(data.name);
21                 $("#modal-input-price").val(data.price);
22                 $("#modal-input-description").val(data.description);
23             }
24         });
25     });
26
27
28     $("#save-button").click(function () {
29         // get the values of the input fields and make a product JSON object.
30         var Product = {
31             "Id": $("#modal-input-id").val(),
32             "Name": $("#modal-input-name").val(),
33             "Price": $("#modal-input-price").val(),
34             "Description": $("#modal-input-description").val()
35         };
36         console.log("saved:");
37         console.log(Product);
38
39         // save the updated product record in the database using the controller
40         $.ajax({
41             type: 'json',
42             data: Product,
43             url: '/product/ProcessEditReturnPartial',
44             success: function (data) {
45                 // show the partial update for testing purposes.
46                 console.log(data);
47                 // replace the proper card with the new data.
48                 $("#card-number-" + Product.Id).html(data).hide().fadeIn(2000);
49             }
50         });
51     });
52 });

```

30. Run the program. You should be able to **Edit** any product on the page.
31. Take a **screen shot** of the app at this stage. Paste it into a Word document and caption the image with a brief explanation of what you just demonstrated.

Deliverables:

1. This activity has multiple parts. Complete all parts before submitting.
2. Submit a Word document with screen shots of the application being run at each stage of development. Show each screen of the output and put a caption under each picture explaining what is being demonstrated.
3. In the same document, in one paragraph, write a summary of the key concepts that were demonstrated in this lesson.
4. Submit a ZIP file of the project file. In order to save space, you can delete the bin and the obj folders of the project. These folders contain the compiled version of the application and are automatically regenerated each time the build or run commands are executed.

