

Spice Projects Note:

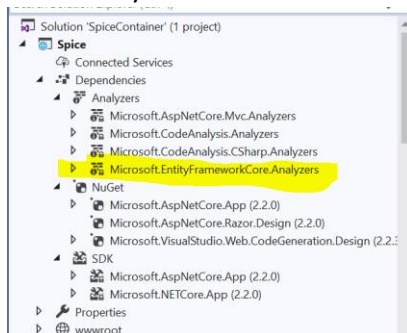
1. When Creating the Project ensure to choose asp.net core and Individual Authentication.
2. Go to startup.cs and edit the connection string with short name such as asp-Spice of data base that you want to create, highlighted below (dark)



```
1 {
2   "ConnectionStrings": {
3     "DefaultConnection": "server=LAPTOP-ALI\\SQL EXPRESS;Database=aspnet-Spice;Trusted_Connection=True;MultipleActiveResultSets=true"
4   },
5   "Logging": {
6     "LogLevel": {
7       "Default": "Warning"
8     }
9   },
10  "AllowedHosts": "*"
11 }
12
```

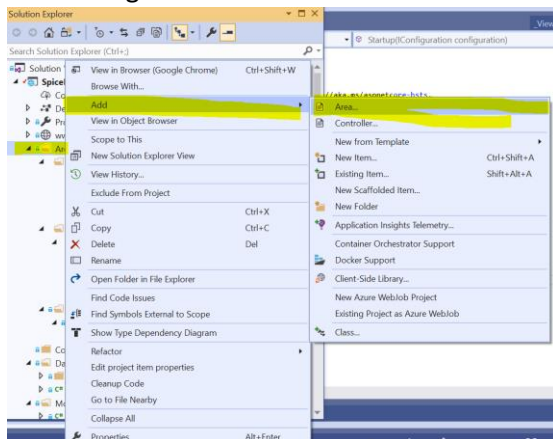
3. Open NuGet Console and type update-database, it will create database in server, If it cannot something is wrong, I have not installed any Entity framework for it to work. Make sure to change the sql server name in connection, highlighted above in yellow.

The above is necessary otherwise you may experience Entity framework errors. These packages are installed by default. See below and it works as it works with additional packages.



Creating Separate Areas for both Admin and Customer.

1. Right click and create two areas for each one (Customer and Admin)



2. You can actually can make one home, make the customer, in the startup file , in the startup file replace it with original as the following:

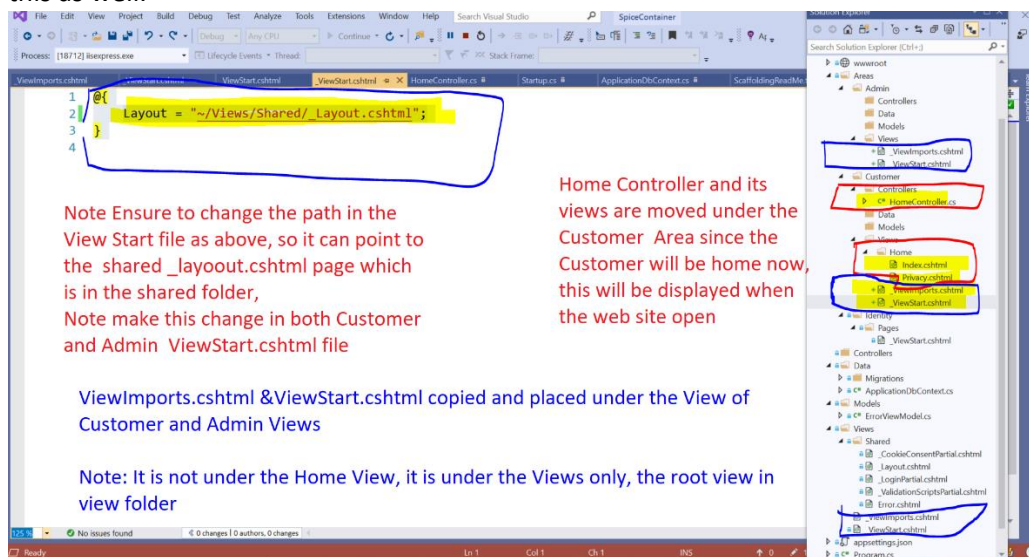
```
routes.MapRoute(
    name: "areas",
    template: "{area=Customer}/{controller=Home}/{action=Index}/{id?}");
```

```

5         }
6     }
7     else
8     {
9         app.UseExceptionHandler("/Home/Error");
10        // The default HSTS value is 30 days. You may want to change this for production s
11        app.UseHsts();
12    }
13
14    app.UseHttpsRedirection();
15    app.UseStaticFiles();
16    app.UseCookiePolicy();
17
18    app.UseAuthentication();
19
20    app.UseMvc(routes =>
21    {
22        routes.MapRoute(
23            name: "areas",
24            template: "{area=Customer}/{controller=Home}/{action=Index}/{id?}");
25    });

```

- 3.
4. Now you need to have the views and separate controller in both customer and admin. Example for customer is shown below: You may not need to copy the ViewStart.cshtml as it work without this as well.



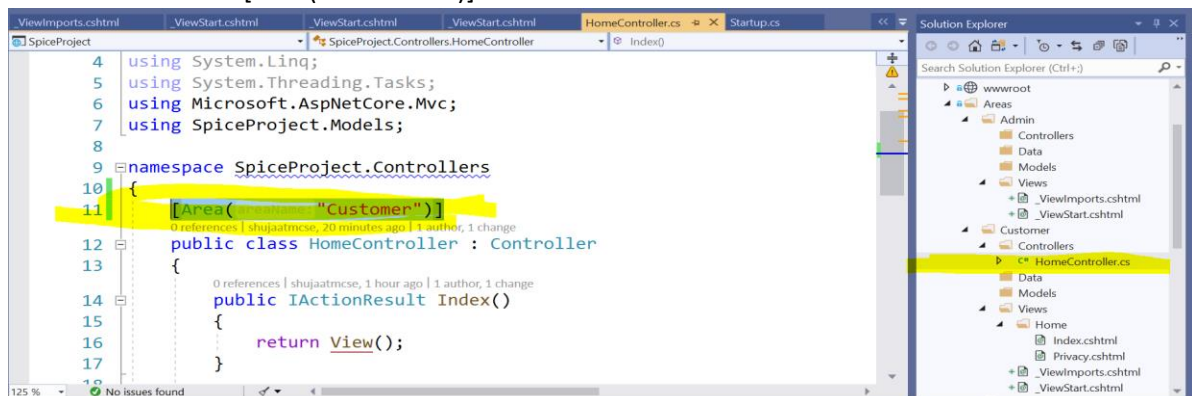
Note Ensure to change the path in the View Start file as above, so it can point to the shared _layout.cshtml page which is in the shared folder,
Note make this change in both Customer and Admin ViewStart.cshtml file

Home Controller and its views are moved under the Customer Area since the Customer will be home now, this will be displayed when the web site open

ViewImports.cshtml & ViewStart.cshtml copied and placed under the View of Customer and Admin Views

Note: It is not under the Home View, it is under the Views only, the root view in view folder

5. Now we need to tell the Home Controller which we moved to Customer about the Area. Hence do it as below: Add [Area("Customer")]

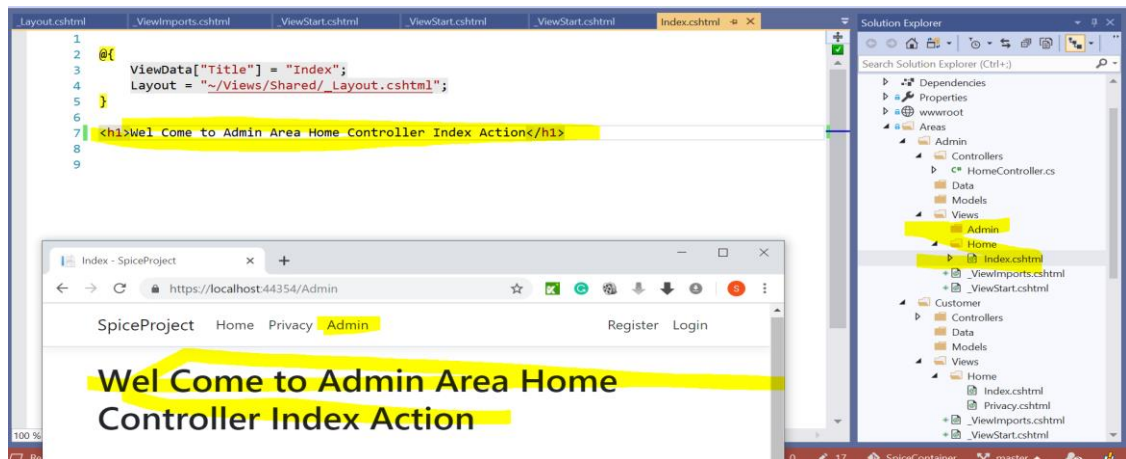


```

4 using System.Linq;
5 using System.Threading.Tasks;
6 using Microsoft.AspNetCore.Mvc;
7 using SpiceProject.Models;
8
9 namespace SpiceProject.Controllers
10 {
11     [Area("Customer")]
12     public class HomeController : Controller
13     {
14         public IActionResult Index()
15         {
16             return View();
17         }
18     }

```

- 6.
7. Below is a screen shot that shown the admin controller as well.



The complete code is on github with commits [Customer&AdminAreaWithReadMeFile](#)

Migration/Model/DBSet.

To create the Category model do the following:

1. create the Category Model Class in Model folder.
2. In your ApplicationDbContext.cs add : `public DbSet<Category> Categories { get; set; }`
3. At Nuget Console type: `add-migration MainCategoryAddedFirstMigration`
4. Type: `Update-database`

To add a foreign Key in the Code first approach, see example below how the parent class is referenced in the sub class.

```
public class SubCategory
{
    public int Id { get; set; }

    public string Name { get; set; }

    [Required]
    [Display(Name = "Category")]
    public int CategoryId { get; set; }

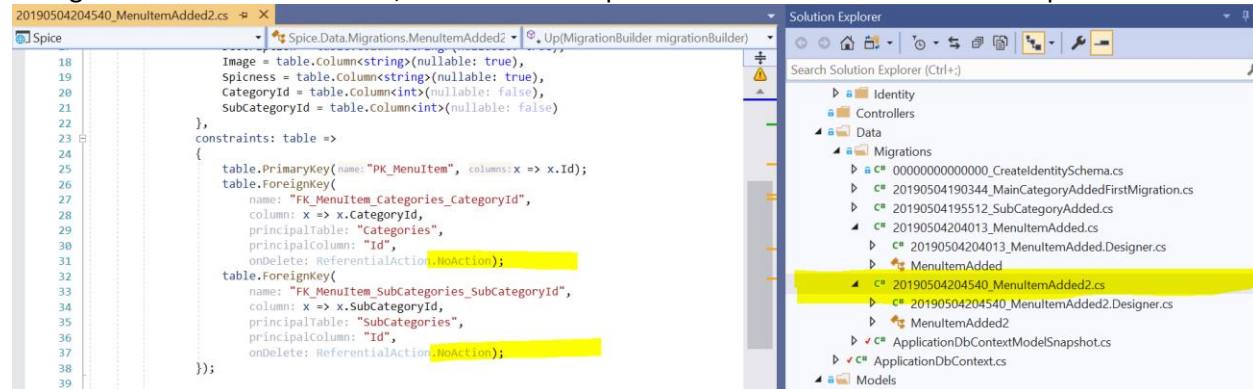
    [ForeignKey("CategoryId")] // Category is referenced to the Id of Category in //Category
                              // class, each sub category can belong to a category, the Category class
                              // is referenced after the virtual word.
    public virtual Category Category { get; set; }
}
```

//The Category Class: Note the Id in Category class is just Id

```
public class Category
{
    [Key]
    public int Id { get; set; }
    [Required]
    public string Name { get; set; }
}
```

Adding the Enum for MenuItem and its detail members:

you can get an error when updating the database at Nuget Console , so go to the context detail file and change the cascaded to NoAction, then rerun the update-database command. See example below



The error is because of Foreign Key constraint, if parent deleted what will the child do.

For Adding Enum: The `Html.GetEnumList` must be used in the view.

```
<div class="form-group"><label asp-for="Spiciness" class="control-label"></label>
<select asp-for="Spiciness" class="form-control" asp-items="Html.GetEnumSelectList<MenuItem.Espicy>()"></select></div>
```

Below is a complete class of MenuItem.

```
public class MenuItem
{
    public int Id { get; set; }
    public string Name { get; set; }
    public string Description { get; set; }
    public string Image { get; set; }
    //Enum Start
    public string Spiciness { get; set; } // We will use this as Input in View
    public enum Espicy { Na=0, NotSpicy=1, Spicy=2, HotSpicy=3} //This will be Html.get Elements
    //End,

    public decimal Price { get; set; }

    //referencing Category
    public int CategoryId { get; set; }

    [ForeignKey("CategoryId")]
    public Category Category { get; set; }

    //referencing Sub Category
    public int SubCategoryId { get; set; }

    [ForeignKey("SubCategoryId")]
    public SubCategory SubCategory { get; set; }
}
```

Rich Text Editor in the Menu Page:

On the page it self such as on the Menu Create Action Page make sure to include this.

@section Scripts

```
{
    <script>tinymce.init({
        selector:'textarea', plugins: "lists", menubar: 'file edit format'});</script>
}
```

//Note Don't include only text area, you have to include plugins, and menubar

In the Main

In the _Layout.cshtml page, In Include Enviroment add below, **note** it has API Key for localhost, You will need another key for domain name from the tiny.

```
<script
src="https://cloud.tinymce.com/5/tinymce.min.js?apiKey=koc6wzmfpb6p8drpyd2uoymbfpgbyw
p5ysjd75wr8buorhut"> </script>
```

Changing the SubCategory Select Option when Main Category is changed

1. If you are expecting JSON data then your controller should have an action which returns data in json format. In this example see the controller action below:

```
public/* JsonResult*/ async Task<IActionResult> GetSubCategories(int?
CategoryIdInController)
{
    if (CategoryIdInController == null)
    {
        return NotFound();
    }

    var subCategory = await _context.SubCategories
        .Include(s => s.Category).Where(c=>c.CategoryId==
CategoryIdInController).ToListAsync();
    if (subCategory == null)
    {
        return NotFound();
    }

    return Json(new SelectList(subCategory, "Id", "Name"));
}
```

2. In View you will basically have 2 select lists, One main which will be populated by default and the other one will be papulated based on the selction of main Category.

Code for main Category:

```
<div class="form-group">
    <label asp-for="CategoryId" class="control-label"></label>
    <select id="CategoryId" asp-for="CategoryId" class="form-control" asp-items="ViewBag.CategoryId"
onchange="GetSubCategory()">
        <option>Please Select One</option>
    </select>
</div>
```

3. Code for subcategory

```
<div class="form-group">
    <label asp-for="SubCategoryId" class="control-label"></label>
    <select id="SubCategoryId" asp-for="SubCategoryId" class="form-control">
        <option >Please select one</option>
    </select>
</div>
```

The AJAX, JQUERY AND JS Code to make it work.

@*Really important to understand how you can get different select list using, jquery, js and ajax.*@

```
<script src="http://ajax.googleapis.com/ajax/libs/jqueryui/1.8/jquery-  
ui.min.js"></script>
```

@*Ajax plugin is needed, it will work even without adding the above because it is in the_layout page*@

@*This is added to check the wiring of JQuery , if it works.*@

```
<script type="text/javascript">  
$(document).ready(function(){  
    $("#msgId").html("This is Hello World from JQuery");  
});
```

@*@* When the Category changes, we want to get the list of subcategory hence using java script for this purpose*@

```
function GetSubCategory() {  
    //alert("main category selected"); /*again to check wiring that we can  
    successfully call the function when category is changed.*/  
    console.log("Sub category function called on Category change");  
    $.ajax({  
        url: '@Url.Action("GetSubCategories", "MenuItems")', /*In the url.action  
we write the action and then controller*/  
        contentType: "application/json; charset=utf-8", //the type of content  
        datatype: "Json", // Make sure you know you are getting the data in return in  
the json format, try to check it by postman  
        //Gethttps://localhost:44362/Admin/MenuItems/GetSubCategories the data  
format will be like [{ value:1, text}, {value:2, text: cat name}, {etc}]  
        // You can also send/receive data in other formats  
        data: { CategoryIdInController: $(CategoryId).val() },// Here we write  
the Id in the controller action and the one we are sending.I have selected the Id from  
DOM using JQuery  
        success: function (data) {  
            $('#SubCategoryId').empty();/* To make the sub category list empty, make  
sure you call this method in correct syntax way*/  
            $('#SubCategoryId').append("<option value= 0>Please select one  
</option>");  
            var k = 0;  
            while (k < data.length) {  
                $('#SubCategoryId').append("<option value=" +data[k].value+">"  
+data[k].text+"</option>"); // This syntax  
                k++;  
                //The syntax above is extremely crucial , make sure you write it  
in the correct way, otherwise will go crazy.  
                console.log("value" + data[k].value);  
                console.log("text" + data[k].text);  
            }  
        },  
        error: function(reponse) {  
            alert("error : No Category found " + response);  
        }  
    });  
}
```

```
</script>
```

The Screen shot of postman, for this to work you action controller should not require Id, so remove the where condition in the GetSubCategories method in Menu item Controller.

The screenshot shows the Postman interface with a GET request to `https://localhost:44362/Admin/MenuItems/GetSubCategories`. The request is successful with a status of 200 OK and a response time of 890 ms. The response body is a JSON array of 6 subcategory objects.

```
1 {
2   {
3     "disabled": false,
4     "group": null,
5     "selected": false,
6     "text": "My Sub Category 2",
7     "value": "3"
8   },
9   {
10    "disabled": false,
11    "group": null,
12    "selected": false,
13    "text": "My Sub Category 3",
14    "value": "4"
15  },
16  {
17    "disabled": false,
18    "group": null,
19    "selected": false,
20    "text": "ANother sub2",
21    "value": "5"
22  },
23  {
24    "disabled": false,
25    "group": null,
26    "selected": false,
27    "text": "One More sub 2",
28    "value": "6"
29  }
30 }
```