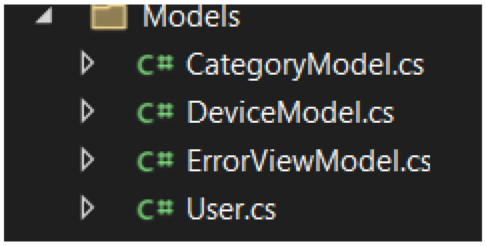
Report Lab 2 – CSE 422

I. Project structure:

This web will apply MVC model in ASP Dotnet Core and combine with MySQL to save and retrieve data. This web provides CRUD features for Category Model, Device Model and User Model and other features such as search devices by categoryId, device status, device name and device code.

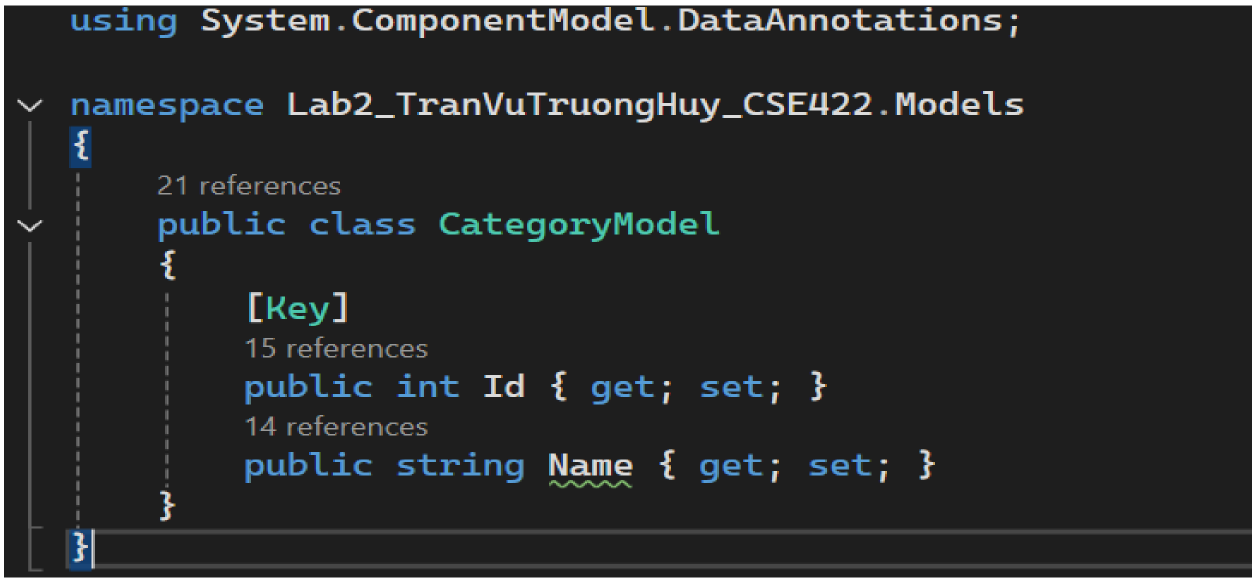
II. Implement:

Folder Models will contain all entity which can help us to connect with MySQL.

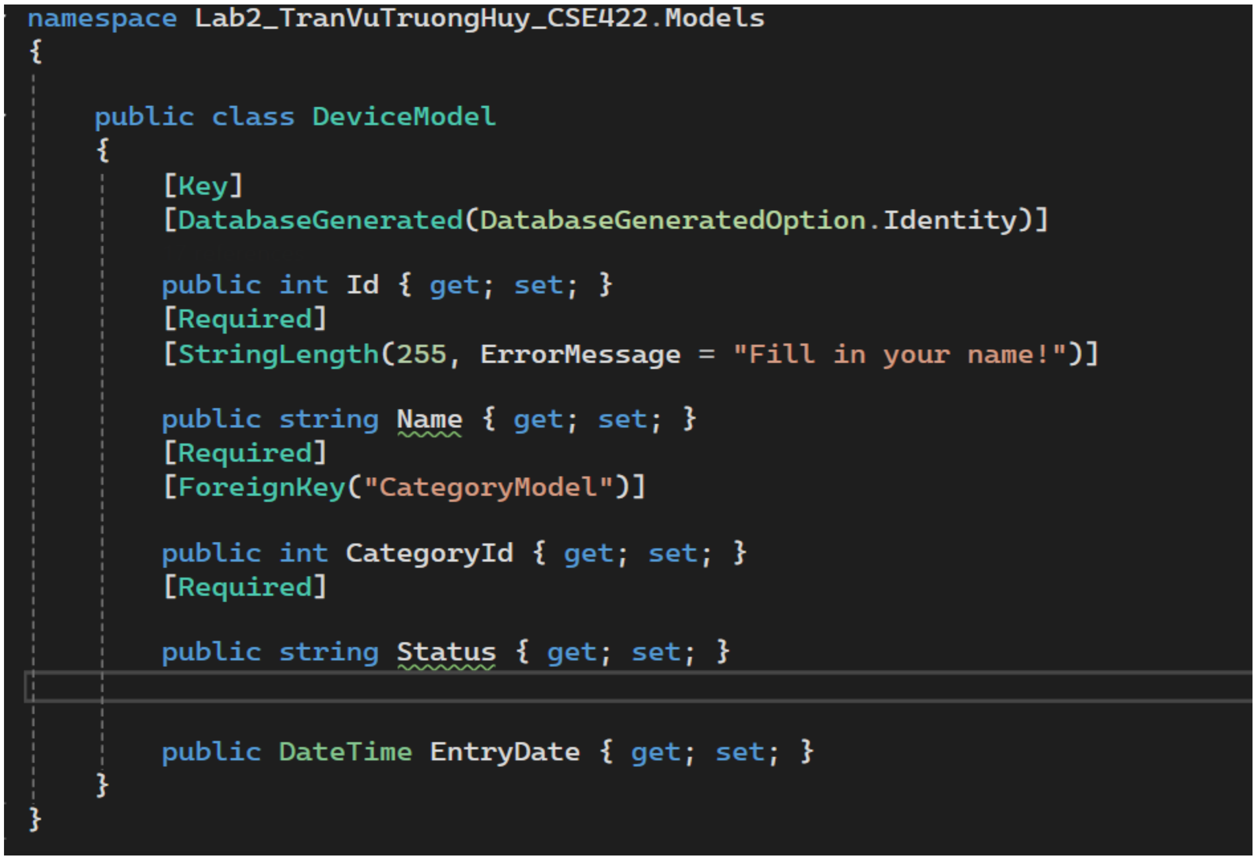


```
Models
    CategoryModel.cs
    DeviceModel.cs
    ErrorViewModel.cs
    User.cs
```

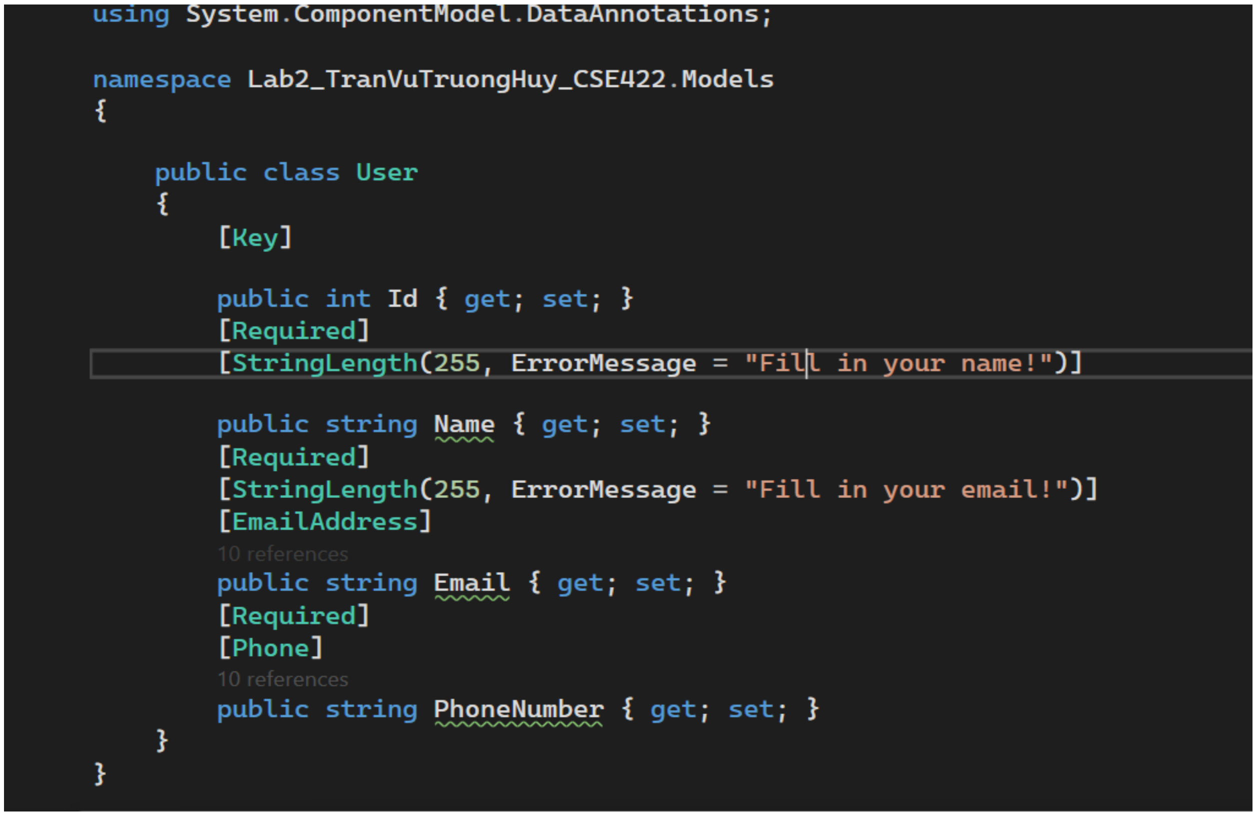
Firstly, I will create these models with attributes below:



```
using System.ComponentModel.DataAnnotations;

namespace Lab2_TranVuTruongHuy_CSE422.Models
{
    21 references
    public class CategoryModel
    {
        [Key]
        15 references
        public int Id { get; set; }
        14 references
        public string Name { get; set; }
    }
}
```

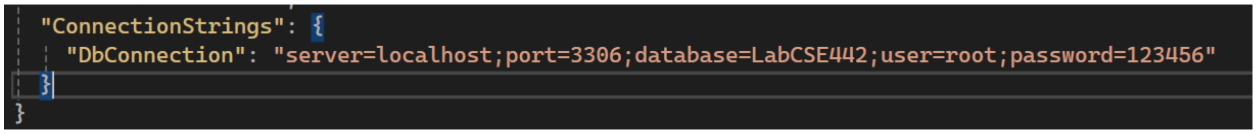


```
namespace Lab2_TranVuTruongHuy_CSE422.Models
{

    public class DeviceModel
    {
        [Key]
        [DatabaseGenerated(DatabaseGeneratedOption.Identity)]

        public int Id { get; set; }
        [Required]
        [StringLength(255, ErrorMessage = "Fill in your name!")]

        public string Name { get; set; }
        [Required]
        [ForeignKey("CategoryModel")]

        public int CategoryId { get; set; }
        [Required]

        public string Status { get; set; }


        public DateTime EntryDate { get; set; }
    }
}
```

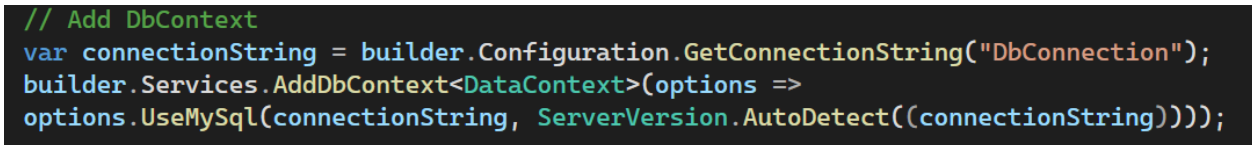


```
using System.ComponentModel.DataAnnotations;

namespace Lab2_TranVuTruongHuy_CSE422.Models
{

    public class User
    {
        [Key]

        public int Id { get; set; }
        [Required]
        [StringLength(255, ErrorMessage = "Fill in your name!")]

        public string Name { get; set; }
        [Required]
        [StringLength(255, ErrorMessage = "Fill in your email!")]
        [EmailAddress]
        10 references
        public string Email { get; set; }
        [Required]
        [Phone]
        10 references
        public string PhoneNumber { get; set; }
    }
}
```

Secondly, I will connect to the database:

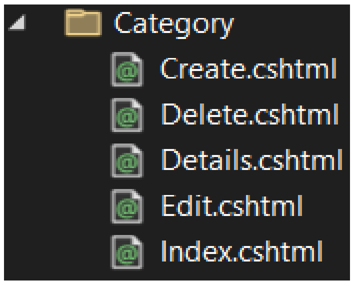


```
  "ConnectionStrings": {
    "DbConnection": "server=localhost;port=3306;database=LabCSE442;user=root;password=123456"
  }
}
```

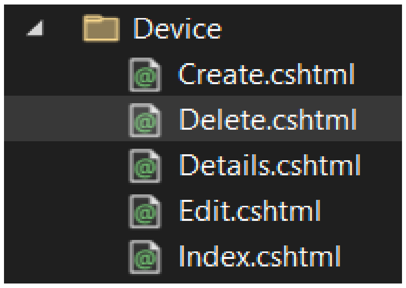


```
// Add DbContext
var connectionString = builder.Configuration.GetConnectionString("DbConnection");
builder.Services.AddDbContext<DataContext>(options =>
options.UseMySql(connectionString, ServerVersion.AutoDetect((connectionString))));
```

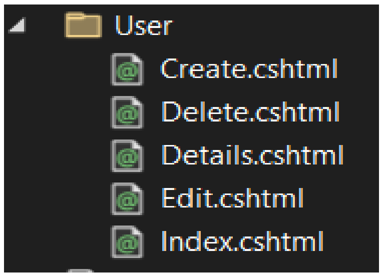
Thirdly, I will create views in folder view which will later combine with the controllers to show in the website (CRUD):



```
Category
    Create.cshtml
    Delete.cshtml
    Details.cshtml
    Edit.cshtml
    Index.cshtml
```



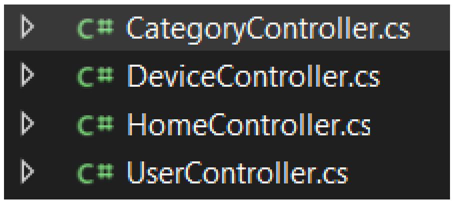
```
Device
    Create.cshtml
    Delete.cshtml
    Details.cshtml
    Edit.cshtml
    Index.cshtml
```



```
User
    Create.cshtml
    Delete.cshtml
    Details.cshtml
    Edit.cshtml
    Index.cshtml
```

These files will contain html and C# code inside to help show data in the website and each folder will need a controller to manage the path.

Lastly, I will create controllers:



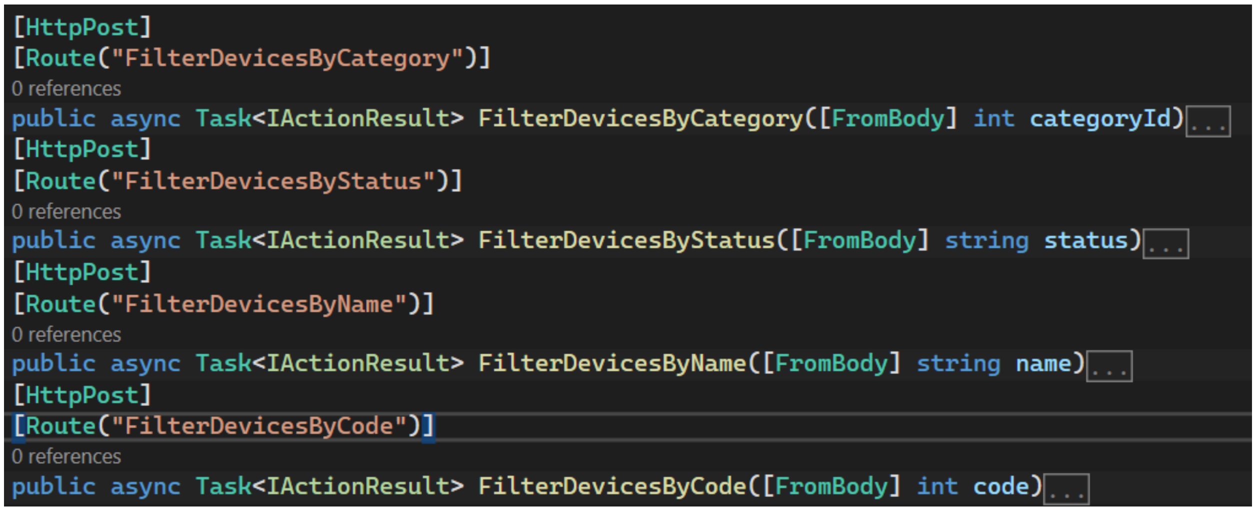
```
CategoryController.cs
DeviceController.cs
HomeController.cs
UserController.cs
```

Each controller is corresponded to each view folder and it will have functions with names the same as cshtml files in view folders.

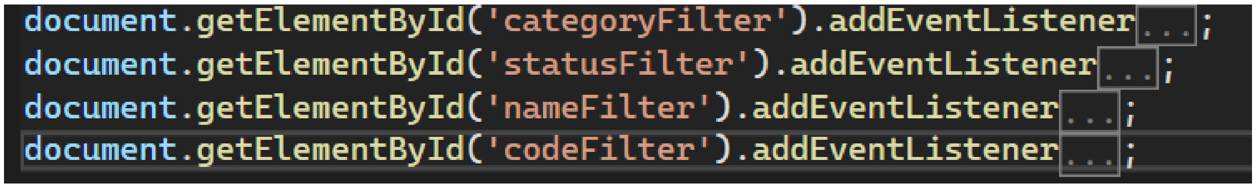
These controllers are used to define the path in url.

III. Filter features:

I will use fetch + javascript to do these feature (when people choose a categoryId, a name, a code or a status, then click on filter buttons, I will pass the parameter of it to the controller, then I will query under the database to get the list of the products and display on the website)



```
[HttpPost]
[Route("FilterDevicesByCategory")]
0 references
public async Task<IActionResult> FilterDevicesByCategory([FromBody] int categoryId)...
[HttpPost]
[Route("FilterDevicesByStatus")]
0 references
public async Task<IActionResult> FilterDevicesByStatus([FromBody] string status)...
[HttpPost]
[Route("FilterDevicesByName")]
0 references
public async Task<IActionResult> FilterDevicesByName([FromBody] string name)...
[HttpPost]
[Route("FilterDevicesByCode")]
0 references
public async Task<IActionResult> FilterDevicesByCode([FromBody] int code)...
```



```
document.getElementById('categoryFilter').addEventListener...;
document.getElementById('statusFilter').addEventListener...;
document.getElementById('nameFilter').addEventListener...;
document.getElementById('codeFilter').addEventListener...;
```