Lab 6: Lập Trình Web ASP.NET CORE MVC (tiếp theo)

Bài 1: Kết nối dữ liệu đến Database SQL Server

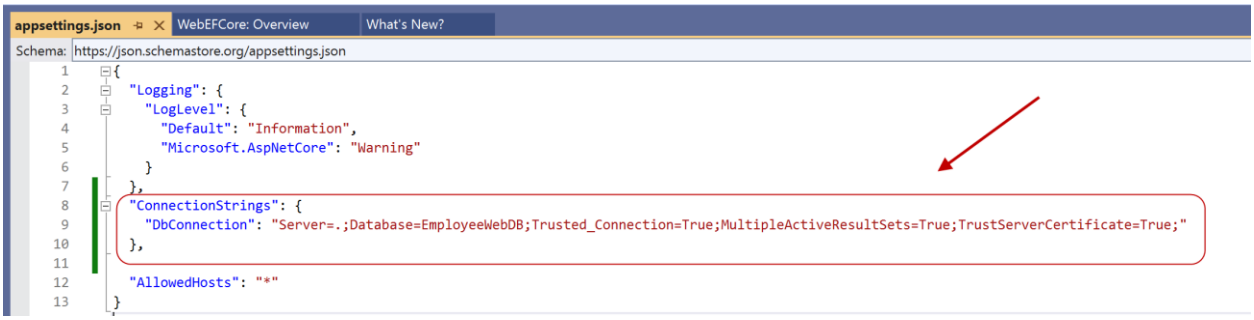
Tạo mới Project ASP.NET Core MVC. Sử dụng NuGet Package Manager cài đặt các Package sau:

Microsoft.EntityFrameworkCore.SqlServer

Microsoft.EntityFrameworkCore.Tools

Thêm chuỗi kết nối sau vào file appsettings.json

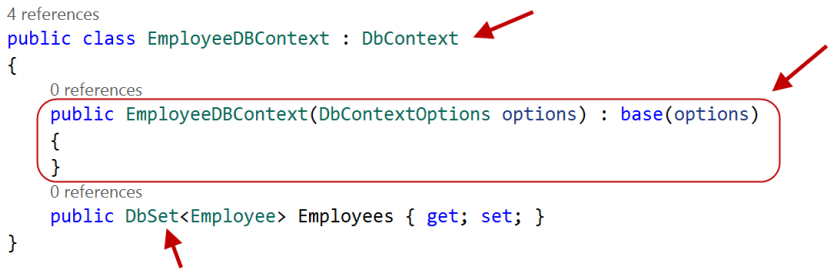
```
"ConnectionStrings": {

    "DbConnection":"Server=
YourMachineName;Database=YourDB;Trusted_Connection=True;MultipleActiveResultSets=True;TrustServer
Certificate=True;"

}
```



Thêm mới class: **Customer** vào Models.

Tạo mới thư mục: Data, trong thư mục này thêm mới class: **CustomerDBContext.cs** (class này kế thừa từ lớp **DBContext**).

Ví dụ tương tự:



Thêm các câu lệnh sau trong file Program.cs để kết nối đến Database: (cần phải using Microsoft.EntityFrameworkCore; trong file Program.cs)

```
// Add services to the container.
var connectionString = builder.Configuration.GetConnectionString("DbConnection");
builder.Services.AddDbContext<YourDBContext >(options => options.UseSqlServer(connectionString));
```

Lưu ý: hai dòng lệnh thêm vào phải nằm trên lệnh var app = builder.Build(); trong file Program.cs

Chạy lệnh Add-migration và Update-database.

Mở SQL Server

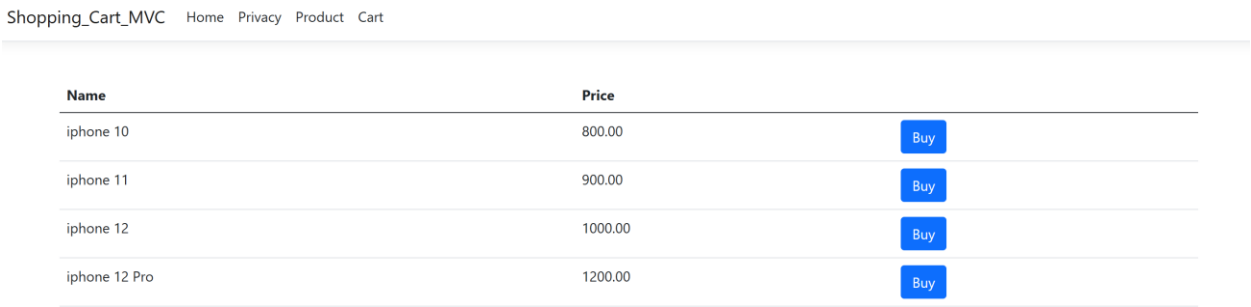
Chèn một số dòng dữ liệu vào bảng Customer đã được tạo như sau:

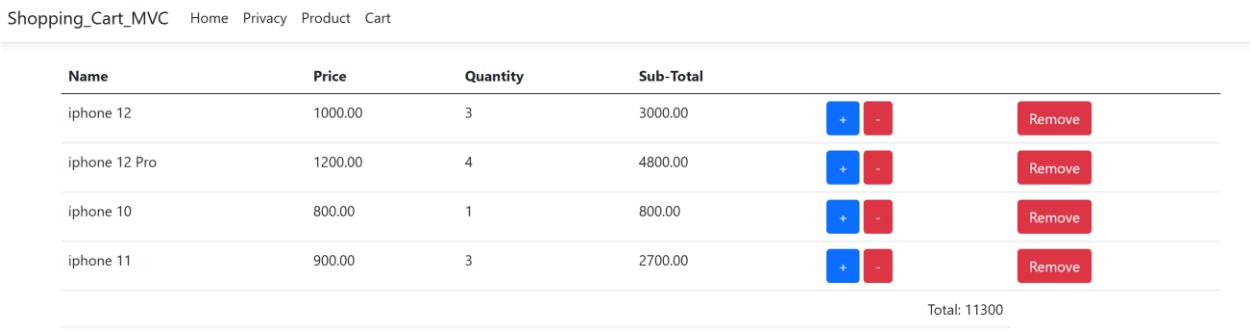
Id	CustomerName	OrderDate
1	John Doe	2023-07-01
2	John Smith	2023-07-02

Đọc dữ liệu từ Database, hiển thị lên giao diện Web Razor Page.

Lưu dự án.

Bài 2: Tạo project ASP MVC Net Core. Tạo các trang Product và Cart như sau:





Hướng dẫn:

Thêm class Model:

```
public class Product
{
    public string Sku { get; set; }
    public string Name { get; set; }
    public decimal Price { get; set; }

}
```

Thêm mới CartController và các Action Method trong Controller này như sau:

```
public class CartController : Controller
 {
     private readonly IProductService _productService;

     public CartController(IProductService productService)
     {
         _productService = productService;
     }

     public IActionResult Index()
     {
         var cart = HttpContext.Session.Get<List<Product_Item>>("cart");
         if (cart != null)
         {
             ViewBag.total = cart.Sum(s => s.Quantity * s.Product.Price);
         }
         else
         {
             cart = new List<Product_Item>();
             ViewBag.total = 0;
         }

         return View(cart);
     }

     public IActionResult Buy(string sku)
     {
         var product = _productService.GetProduct(sku);
         var cart = HttpContext.Session.Get<List<Product_Item>>("cart");

         if (cart == null) //no item in the cart
         {
             cart = new List<Product_Item>();
             cart.Add(new Product_Item { Product = product, Quantity = 1 });
         }
         else
         {
             int index = cart.FindIndex(w => w.Product.Sku == sku);

             if (index != -1) //if item already in the
             {
                 cart[index].Quantity++; //increment by 1
             }
             else
             {
                 cart.Add(new Product_Item { Product = product, Quantity = 1 });
             }
         }

         HttpContext.Session.Set<List<Product_Item>>("cart", cart);
         return RedirectToAction("Index");
     }
     public IActionResult Add(string sku)
     {
         var product = _productService.GetProduct(sku);
         var cart = HttpContext.Session.Get<List<Product_Item>>("cart");

         int index = cart.FindIndex(w => w.Product.Sku == sku);
         cart[index].Quantity++; //increment by 1

         HttpContext.Session.Set<List<Product_Item>>("cart", cart);
         return RedirectToAction("Index");
     }

     public IActionResult Minus(string sku)
     {
         var product = _productService.GetProduct(sku);
         var cart = HttpContext.Session.Get<List<Product_Item>>("cart");

         int index = cart.FindIndex(w => w.Product.Sku == sku);

         if (cart[index].Quantity == 1) //last item of a product
         {
             cart.RemoveAt(index); //remove it
         }
         else
         {
             cart[index].Quantity--; //reduce by 1
         }


         HttpContext.Session.Set<List<Product_Item>>("cart", cart);
         return RedirectToAction("Index");
     }
     public IActionResult Remove(string sku)
     {
         var product = _productService.GetProduct(sku);
         var cart = HttpContext.Session.Get<List<Product_Item>>("cart");

         int index = cart.FindIndex(w => w.Product.Sku == sku);
         cart.RemoveAt(index);

         HttpContext.Session.Set<List<Product_Item>>("cart", cart);

         return RedirectToAction("Index");
     }
 }
```

Thêm View cho Action method Index của CartController như sau:

```
@model IEnumerable<Shopping_Cart_MVC.Models.Product_Item>
<table class="table">
    <thead>
        <tr>
            <th>@Html.DisplayNameFor(model => model.Product.Name)</th>
            <th>@Html.DisplayNameFor(model => model.Product.Price)</th>
            <th>@Html.DisplayNameFor(model => model.Quantity)</th>
            <th>Sub-Total</th>
            <th></th>
            <th></th>
        </tr>
    </thead>
    <tbody>
        @foreach (var cart in Model)
        {
        <tr>
            <td>
                @Html.DisplayFor(modelItem => cart.Product.Name)
            </td>
            <td>
                @Html.DisplayFor(modelItem => cart.Product.Price)
            </td>
            <td>
                @Html.DisplayFor(modelItem => cart.Quantity)
            </td>
            <td>
                @Html.DisplayFor(modelItem => (cart.SubTotal))
            </td>
            <td>
                @Html.ActionLink("+", "Add", "Cart", new { sku = cart.Product.Sku }, new { @class = "btn btn-primary" })
                @Html.ActionLink("-", "Minus", "Cart", new { sku = cart.Product.Sku }, new { @class = "btn btn-danger" })

            </td>
            <td>
                @Html.ActionLink("Remove", "Remove", "Cart", new { sku = cart.Product.Sku }, new { @class = "btn btn-
danger" })
            </td>
        </tr>
        }
        <tr>
            <td colspan="5" style="text-align:right">
                Total: @ViewBag.total
            </td>
        </tr>
    </tbody>
</table>
```

Sinh viên thực hiện các phần còn lại của Project để hoàn thành bài thực hành.

Chạy chương trình, kiểm tra các chức năng và lưu dự án.

Bài 3: Cấu hình routing trong ASP NET Core

3. 1: Cấu hình Attribute Routing

1. Tạo controller **BlogController** với các action sau:
    - **GetAllPosts:** Truy cập bằng URL **/posts**.
    - **GetPostById:** Truy cập bằng URL **/posts/{id:int}** (ràng buộc **id** là số nguyên).
    - **SearchPosts:** Truy cập bằng URL **/posts/search/{keyword?}** (tham số **keyword** tùy chọn).
2. Thêm route prefix **/blog** cho toàn bộ các action trong **BlogController**.

3.2: Tùy chỉnh Route Constraints

1. Tạo action **GetUsersByRole** trong **UserController**:
    - URL pattern: **/users/{role:alpha}** (ràng buộc **role** chỉ chứa chữ cái).
    - Test với URL hợp lệ (ví dụ: **/users/admin**) và không hợp lệ (ví dụ: **/users/123**).