Phần chuẩn bị

- Tải các package (chú ý phù hợp với phiên bản ASP.NET core đang sử dụng)

Microsoft.AspNetCore.Diagnostics.EntityFrameWorkCore
Microsoft.EntityFrameWorkCore.SqlServer
Microsoft.EntityFrameWorkCore.Sqlite
Microsoft.EntityFrameWorkCore.Tools
Microsoft.EntityFrameWorkCore.Design
Microsoft.VisualStudio.Web.CodeGeneration.Design

- Về tạo database, chú ý đường link địa chỉ cần phải xác định đúng với máy của mình (nên dẫn đến thư mục chứa bài làm luôn cho tiện)

(NAME = N'QLHangHoa', FILENAME = N'E:\00_UNIVERSITY\TERM_05\LapTrinhWeb\DeThi_Mau_2023\QLHangHoa.mdf' , SIZE = 3072KB , MAXSIZE = UNLIMITED, FILEGROWTH = 1024KB)
LOG ON
(NAME = N'QLHangHoa_log', FILENAME = N'E:\00_UNIVERSITY\TERM_05\LapTrinhWeb\DeThi_Mau_2023\QLHangHoa_log.ldf' , SIZE = 3840KB , MAXSIZE = 2048GB , FILEGROWTH = 10%)

```
/****** Object:  Database [QLHangHoa]    Script Date: 7/6/2021 6:19:03 PM ******/
CREATE DATABASE [QLHangHoa]
 CONTAINMENT = NONE
 ON  PRIMARY
( NAME = N'QLHangHoa', FILENAME = N'E:\00_UNIVERSITY\TERM_05\LapTrinhWeb\DeThi_Mau_2023\QLHangHoa.mdf' , SIZE = 3072KB , MAXSIZE = UNLIMITED, FILEGROWTH = 1024KB )
 LOG ON
( NAME = N'QLHangHoa_log', FILENAME = N'E:\00_UNIVERSITY\TERM_05\LapTrinhWeb\DeThi_Mau_2023\QLHangHoa_log.ldf' , SIZE = 3840KB , MAXSIZE = 2048GB , FILEGROWTH = 10%)
```

Câu 1:

Tạo một dự án ASP.Net Core MVC đặt tên theo cấu trúc: HoTen_MaSV. Sử dụng file index.html để xây dựng layout với tên là HoTen_Layout

- Tạo một View HoTen_Layout.cshtml trong thư mục Views/Shared
- Copy nội dung file index.html vào trong HoTen_Layout.cshtml

1.1 (1 điểm): Sinh viên tách layout thành các phần một cách hợp lý, đặt tên theo quy định (Ví dụ như: HoTen_Header, Hoten_Footer và HoTen_MainContent) và render các phần đó trong layout.

Trong file Layout đã có chú thích các phần, chỉ cần tạo các partial view theo yêu cầu và copy từng đoạn code và paste vào đúng trong các layout thành phần.

Sau đó ở HoTen_Layout.cshmtl, thay thế:

- Các layout thành phần bằng <partial name="ten_cua_Layout_Component" />
- Riêng phần MainContent thay thế bằng @RenderBody()

1.2 (1 điểm): Ghép các thư mục tài nguyên (cs, js, images…) vào dự án theo quy định của dự án ASP.Net Core MVC.

Ở yêu cầu này chỉ cần di chuyển các tài nguyên vào đúng vị trí trong wwwroot

Câu 2:

Hiển thị các LoaiHang lên thanh điều hướng

2.1 (1 điểm): Sử dụng Entity framework, sinh các Model hỗ trợ kết nối cơ sở dữ liệu

Xác định Connection String của DB bằng 2 cách:
Cách 1: Vào tab SQL Server Object Explorer, chuột phải vào Database chọn Properties, nếu có Connection String thì copy, nếu không có thì thực hiện cách 2.
Cách 2:

- Chỉnh sửa Connection String mẫu: Scaffold-DbContext "Server=Ten_Server;Database=Ten_Database;Trusted_Connection=True;TrustServerCertificate=True;" Microsoft.EntityFrameworkCore.SqlServer -OutputDir Models -Force
- Trên thanh công cụ, Tools > NuGet Package Manager > Package Manager Console, rồi chạy dòng Connection String trên.

Nếu thực hiện đúng, thì các Model sẽ được sinh ra trong thư mục Models.

2.2 (1 điểm): Load dữ liệu từ bảng LoaiHang và hiển thị lên trên thanh điều hướng, trong đó menu Home được thay thế bằng họ tên sinh viên ví dụ: “Nguyễn Văn An”

Bước 1: Kết nối CSDL trong Program.cs - Thêm cấu hình DbContext để kết nối với cơ sở dữ liệu:

```
builder.Services.AddDbContext<QlhangHoaContext>(options =>

options.UseSqlServer(builder.Configuration.GetConnectionStrin
g("DefaultConnection")));
```

Bước 2: Tạo View Component để load dữ liệu LoaiHang
Tạo thư mục ViewComponents ngang cấp so với các thư mục Models, Views, Controllers, …
Tạo file RenderViewComponent.cs bên trong thư mục ViewComponents

```
using Microsoft.AspNetCore.Mvc;
using NguyenVietHoang_231230791.Models;

namespace NguyenVietHoang_231230791.ViewComponents
{
    public class RenderViewComponent : ViewComponent
    {
        private readonly QlhangHoaContext _context;

        public RenderViewComponent(QlhangHoaContext context)
        {
            _context = context;
        }

        public async Task<IViewComponentResult> InvokeAsync()
        {
            var loaiHangs = _context.LoaiHangs.ToList();
            return View("RenderLoaiHang", loaiHangs);
        }
    }
}
```

Giải thích:

- RenderViewComponent là class kế thừa từ ViewComponent
- ASP.NET Core tự hiểu tên ViewComponent là Render (từ phái trước của "ViewComponent")
- Dữ liệu lấy từ bảng LoaiHang trong DbContext QlhangHoaContext

Bước 3: Tạo View cho ViewComponent

- Trong thư mục View/Shared tạo thư mục con Components/Render và tạo view RenderLoaiHang.cshtml
- Nội dung bên trong được copy từ partial view thành phần Header từ Layout và chỉ thay thế phần menu:

```
<div class="menu">
     <ul>
          <!-- Thay "Home" bằng tên sinh viên -->
          <li class="active"><a href="/">Nguyễn Việt
Hoàng</a></li>

          <!-- Duyệt danh sách LoaiHang -->
          @foreach (var item in Model)
          {
               <li>
                    <a asp-controller="HangHoa"
asp-action="Index" asp-route-maloai="@item.MaLoai">
                         @item.TenLoai
                    </a>
               </li>
          }
     </ul>
</div>
```

Giải thích:

- Thẻ <li> đầu tiên thay thế “Home” bằng họ tên sinh viên.
- Các <li> sau được sinh tự động dựa trên dữ liệu từ bảng “LoaiHang”
- Sử dụng asp-controller, asp-action, asp-route-maloai để tạo link tự động.

Bước 4: Gọi View Component trong Layout
Trong Layout, phần gọi Header trước đó (<partial name=”Hoten_Footer” />) thay thế bằng

```
@await Component.InvokeAsync("Render")
```

Khi chạy ứng dụng, ViewComponent Render sẽ tự động lấy dữ liệu từ LoaiHang, render ra danh sách menu.

Câu 3:

3 (1 điểm): Load dữ liệu từ bảng HangHoa để hiển thị danh sách các HangHoa có giá trị thuộc tính gia >= 100, trong đó tiêu đề được thay thế bằng họ tên sinh viên ví dụ: “Nguyễn Văn An

Bước 1: Chỉnh sửa bên trong HomeController.cs

```
using System.Diagnostics;
using Microsoft.AspNetCore.Mvc;
using NguyenVietHoang_231230791.Models;

namespace NguyenVietHoang_231230791.Controllers
{
    public class HomeController : Controller
    {
        private readonly QlhangHoaContext _context;

        public HomeController(QlhangHoaContext context)
        {
            _context = context;
        }

        public IActionResult Index()
        {
            var data = _context.HangHoas.Where(h => h.Gia >=
100).ToList();
            return View(data);
        }
    }
}
```

Bước 2: Sửa hiển thị dữ liệu trong Views/Home/Index.cshtml

```
@model IEnumerable<NguyenVietHoang_231230791.Models.HangHoa>
@{
    ViewData["Title"] = "Home Page";
    Layout = "~/Views/Shared/NguyenVietHoang_Layout.cshtml";
}

<partial name="NguyenVietHoang_MainContent" model="Model" />
```

Bước 3: Sửa hiển thị dữ liệu bên trong
Views/Shared/NguyenVietHoang_MainContent.cshtml

```
@model IEnumerable<NguyenVietHoang_231230791.Models.HangHoa>

<div class="main">
     <div class="content">
          <!-- Begin New Product -->
          <div class="content_top">
               <div class="heading">
                    <h3>NGUYỄN VĂN AN</h3>
               </div>
               <div class="see">
                    <p><a href="#">See all Products</a></p>
               </div>
               <div class="clear"></div>
          </div>

          <div class="section group">
               @foreach (var item in Model)
               {
                    <div class="grid_1_of_4 images_1_of_4">
                         <a href="#">
                              <img
src="@Url.Content("~/images/" + item.Anh)"
alt="@item.TenHang" />
                         </a>

                         <h2>@item.TenHang</h2>
                         <div class="price-details">
                              <div class="price-number">
                                   <p><span
class="rupees">@item.Gia</span></p>
                              </div>
                              <div class="add-cart">
                                   <h4><a href="#">Add to
Cart</a></h4>
                              </div>
                              <div class="clear"></div>
                         </div>
                    </div>
               }
```

```
            </div>
            <!-- End New Product -->
        </div>
</div>
```

Câu 4:

4 (2 điểm): Kích vào từng mục trong thanh điều hướng danh sách HangHoa sẽ được lọc và hiển thị trong phần main content theo cơ chế AJAX

Bước 1: Điều chỉnh RenderLoaiHang.cshtml

Thay thế đoạn <a> thành link ảo để bắt sự kiện bằng JS

- Trước đó:

```
<a asp-controller="HangHoa" asp-action="Index"
asp-route-maloai="@item.MaLoai"> @item.TenLoai
</a>
```

- Thay thế bằng:

```
<a href="#" class="loai-link" data-maloai="@item.MaLoai">
                    @item.TenLoai
</a>
```

Bước 2: Tạo Action mới trong HomeController để xử lý lọc dữ liệu

```
[HttpGet]
public IActionResult LocHangTheoLoai(int maloai)
{
   var data = _context.HangHoas
      .Where(h => h.MaLoai == maloai && h.Gia >= 100)
      .ToList();

   return PartialView("_DanhSachHangHoa", data);
}
```

Giải thích:

- Action này trả về Partial View chứa danh sách hàng hóa lọc theo loại.
- Không return View(), vì AJAX chỉ cần nội dung HTML chèn vào

MainContent.

Bước 3: Tạo Partial View _DanhSachHangHoa.cshtml trong Views/Home với nội dung được tách ra từ phần hiển thị sản phẩm trong Hoten_MainContent.cshtml

```
@model IEnumerable<NguyenVietHoang_231230791.Models.HangHoa>

<div class="section group">
     @foreach (var item in Model)
     {
          <div class="grid_1_of_4 images_1_of_4">
               <a href="#">
                    <img src="@Url.Content("~/images/" +
item.Anh)" alt="@item.TenHang" />
               </a>
               <h2>@item.TenHang</h2>
               <div class="price-details">
                    <div class="price-number">
                         <p><span
class="rupees">@item.Gia</span></p>
                    </div>
                    <div class="add-cart">
                         <h4><a href="#">Add to Cart</a></h4>
                    </div>
                    <div class="clear"></div>
               </div>
          </div>
     }
</div>
```

Bước 4: Sửa lại Hoten_MainContent.cshtml thay thế cho phần hiển thị trước đó được tách ra.

```
<div class="ajax-container">
     @await Html.PartialAsync("_DanhSachHangHoa", Model)
</div>
```

Bước 5: Thêm AJAX Script trong Layout ở sau thẻ </body>

```
<script
src="https://code.jquery.com/jquery-3.6.0.min.js"></script>
<script>
     $(document).ready(function () {
          $(".loai-link").on("click", function (e) {
               e.preventDefault(); // Ngăn load lại trang
               var maloai = $(this).data("maloai");

               $.ajax({
                    url: '/Home/LocHangTheoLoai',
                    type: 'GET',
                    data: { maloai: maloai },
                    success: function (result) {
                         $(".ajax-container").html(result);
                    },
                    error: function () {
                         alert("Không thể tải dữ liệu hàng
hóa.");
                    }
               });
          });
     });
</script>
```

Câu 5:

5 (3 điểm): Cập nhật bảng HangHoa
5.1 (2 điểm): Xây dựng chức năng Thêm Hàng Hóa mới

Bước 1: Bổ sung 2 Action bên trong HomeController.cs

```
// GET: Hiển thị form thêm hàng hóa mới
[HttpGet]
public IActionResult Create()
{
    ViewBag.MaLoai = _context.LoaiHangs.ToList();
    return View();
```

```
}

// POST: Xử lý thêm mới hàng hóa
[HttpPost]
[ValidateAntiForgeryToken]
public IActionResult Create(HangHoa hanghoa)
{
    _context.HangHoas.Add(hanghoa);
    _context.SaveChanges();
    return RedirectToAction("Index");
}
```

Bước 2: Tạo view Create.cshtml với nội dung:

```
@model NguyenVietHoang_231230791.Models.HangHoa

@{
    ViewData["Title"] = "Thêm Hàng Hóa Mới";
}

<h2>Thêm Hàng Hóa Mới</h2>

<form asp-action="Create" method="post">
    <div>
        <label>Mã Loại</label>
        <select asp-for="MaLoai" class="form-control">
            <option value="">-- Chọn loại hàng --</option>
            @foreach (var item in ViewBag.MaLoai)
            {
                <option
value="@item.MaLoai">@item.TenLoai</option>
            }
        </select>
    </div>

    <div>
        <label>Tên Hàng</label>
        <input asp-for="TenHang" class="form-control" />
```

```
    </div>

    <div>
        <label>Giá</label>
        <input asp-for="Gia" class="form-control" />
    </div>

    <div>
        <label>Ảnh</label>
        <input asp-for="Anh" class="form-control" />
    </div>

    <button type="submit" class="btn btn-primary
mt-2">Create</button>
</form>
```

Bước 3: Thêm liên kết đến trang Thêm mới bên trong view Index.cshtml

```
<a asp-action="Create" asp-controller="Home" class="btn
btn-success mb-2">
    Thêm Hàng Hóa Mới
</a>
```

Bước 4: Thêm lệnh gọi Section Scripts trong NguyenVietHoang_Layout.cshtml trước bao đóng phần body.

```
     </div>
     <!-- End Wrap -->
     <!-- Begin Footer -->
    <partial name="NguyenVietHoang_Footer" />
     <!-- End Footer -->
     @await RenderSectionAsync("Scripts", required: false)
</body>
```

5.2 (1 điểm): Xây dựng Anotation để validate Giá phải là số và năm trong giới hạn 100 đến 5000. Tên file ảnh phải có đuôi: .jpg | .png | .gif | .tiff

Bước 1: Trong script HangHoa.cs, thêm lệnh gọi thư viện.

```
using System.ComponentModel.DataAnnotations; // 1. Thêm thư viện này
```

Bước 2: Thêm các dòng annotation.

```
// 2. Validate Giá: Từ 100 đến 5000
[Range(100, 5000, ErrorMessage = "Giá phải nằm trong khoảng từ 100 đến 5000")]
public decimal? Gia { get; set; }

// 3. Validate Ảnh: Sử dụng Regex để kiểm tra đuôi file
// Pattern giải thích: Kết thúc bằng (.) sau đó là jpg/png/gif/tiff
[RegularExpression(@"^.*\.(jpg|png|gif|tiff)$", ErrorMessage = "Tên file ảnh phải có đuôi: .jpg, .png, .gif, .tiff")]
public string? Anh { get; set; }
```