



Mục tiêu: Làm quen với ASP.NET Web Application & MVC design pattern

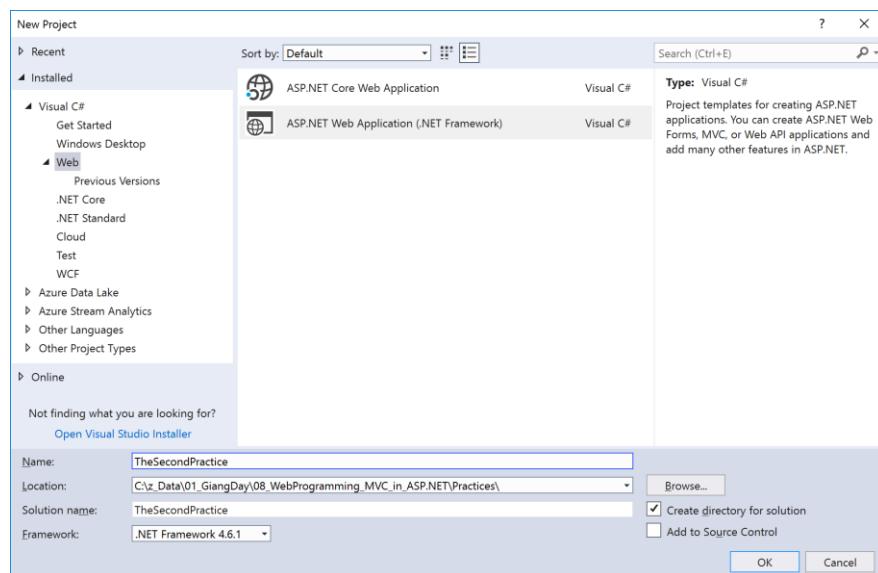
Bài 1:

- Tạo ASP.NET Web application
- Sử dụng Bootstrap cho giao diện [Manage NuGet – Copy by manual]
- Làm quen với Controller và View

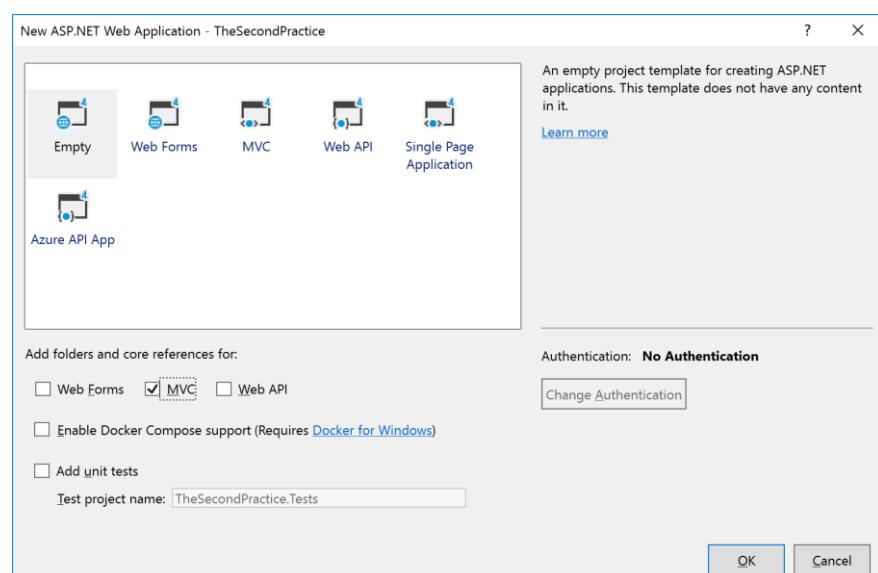
I/- Tạo ứng dụng web với Bootstrap framework

1/- Tạo mới một MVC Project

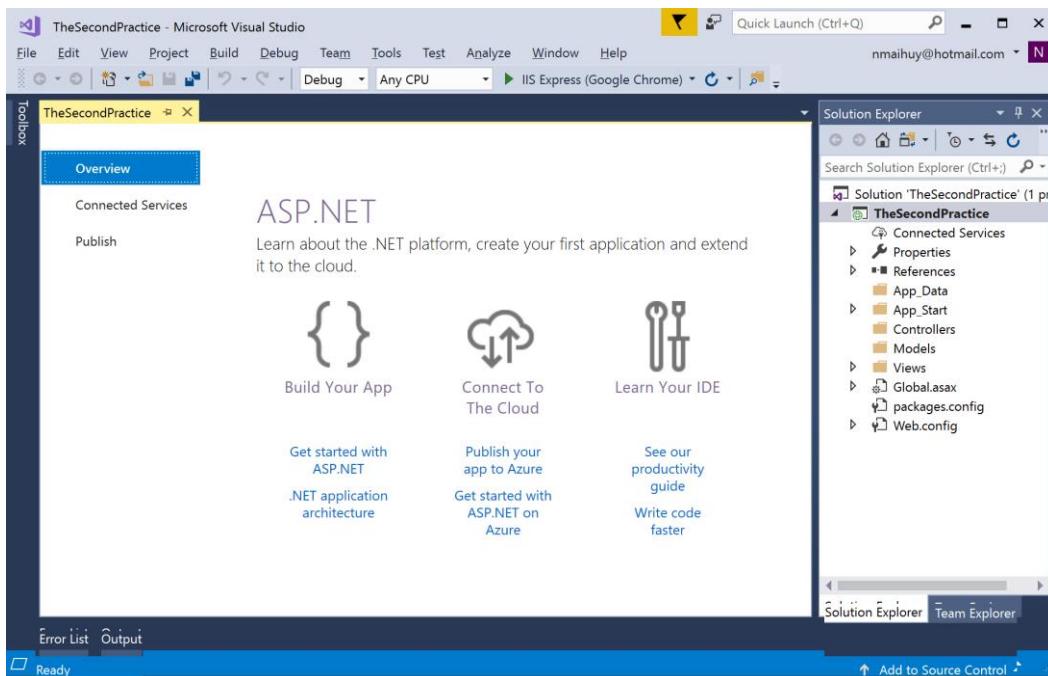
[File]New / Project



Đặt tên cho ứng dụng là TheSecondPractice -> Click vào nút OK



Chọn loại ứng dụng là **Empty**, đánh dấu mục chọn **MVC** để xác định áp dụng MVC cho ứng dụng rồi nhấn OK để hoàn tất. Giao diện của ứng dụng web trong chế độ design của visual studio sẽ thể hiện như hình sau

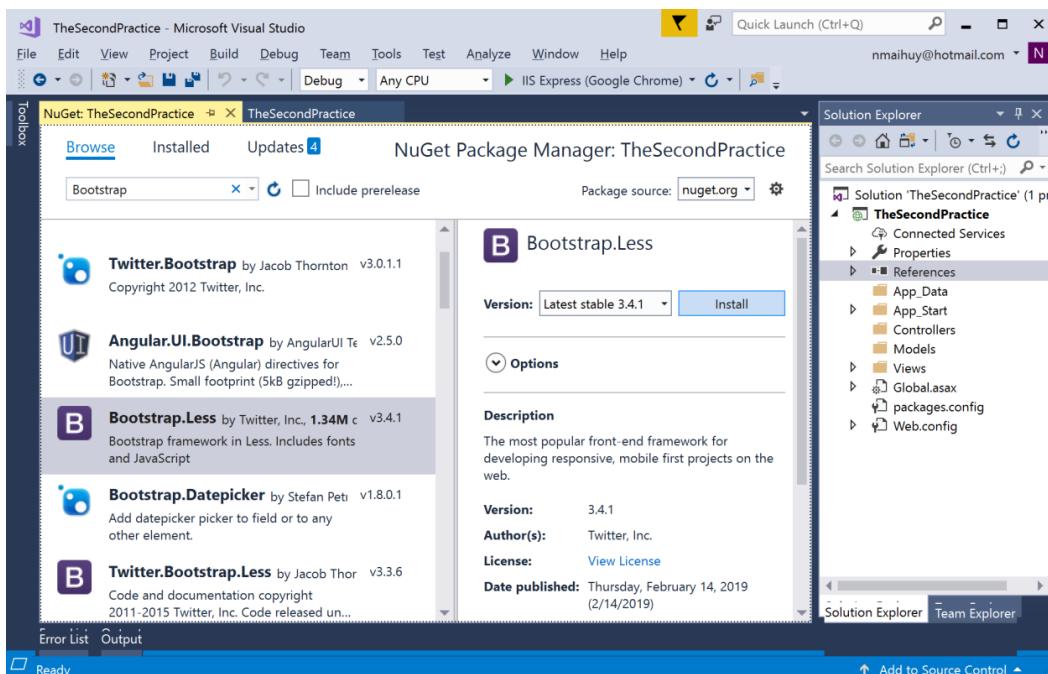


2/- Sử dụng Bootstrap framework cho giao diện

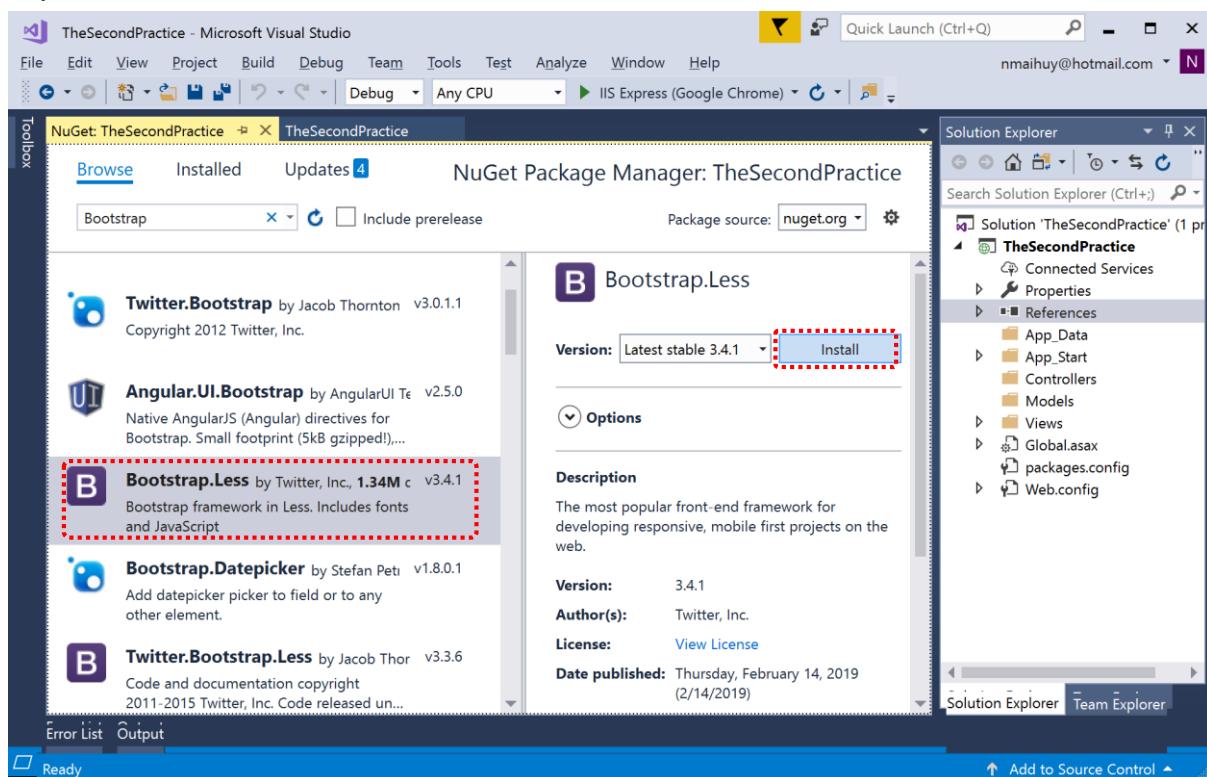
Để xây dựng giao diện của ứng dụng dựa trên Bootstrap framework, có thể thực hiện bằng một trong 2 cách

- Manage NuGet packages
- Thao tác thủ công bằng cách sao chép trực tiếp

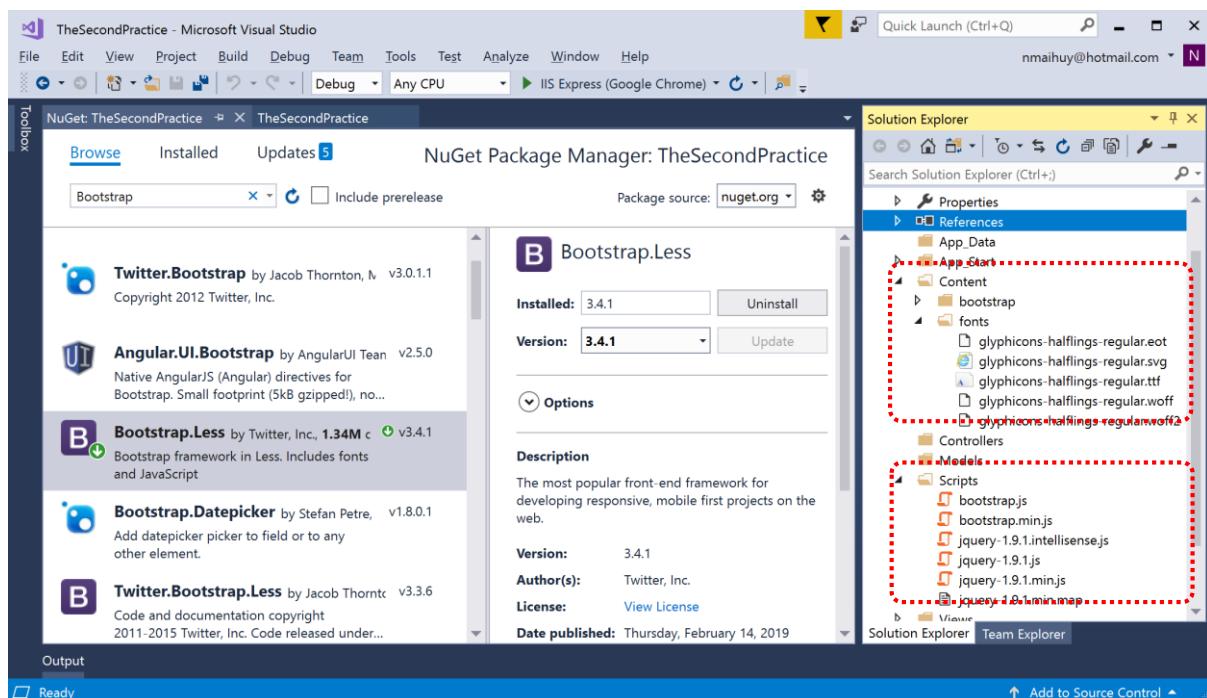
Tại khu vực “**Solution explorer**” của project hiện tại, right click vào mục **References** rồi chọn lệnh “**Manage NuGet packages ...**”. Lúc này, tính năng “Nuget ...” sẽ xuất hiện như hình minh họa dưới đây



Tại thẻ “Browser”, gõ từ khóa “Bootstrap” trong khung tìm kiếm, bạn sẽ thấy các thư viện có tên chứa từ khóa hiện ra danh sách ở phía dưới (*xem hình minh họa*). Hãy chọn thư viện muốn sử dụng cho dự án của mình trong danh sách có trong khung bên trái của vùng làm việc, và chọn nút Install trong phần màn hình phía bên phải để cài vào dự án hiện hành

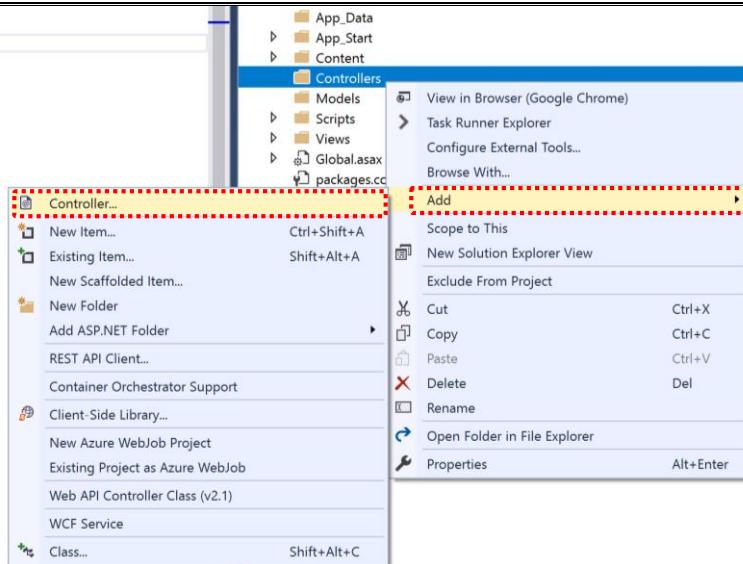


Sau khi hoàn tất, trong cấu trúc thuộc “Solution Explorer”, ta sẽ thấy có thêm một số thư mục có liên quan. VD: Content, Script, ...

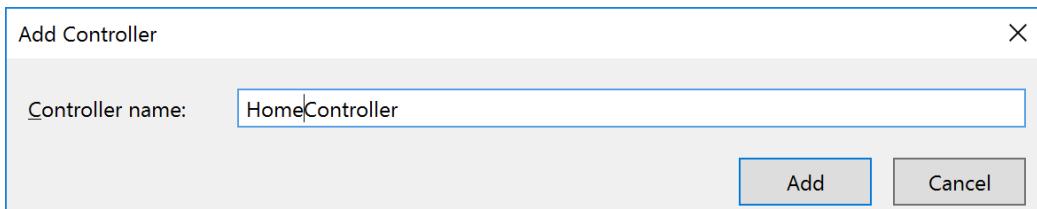


3/- Tạo Controller – View cho trang đầu tiên

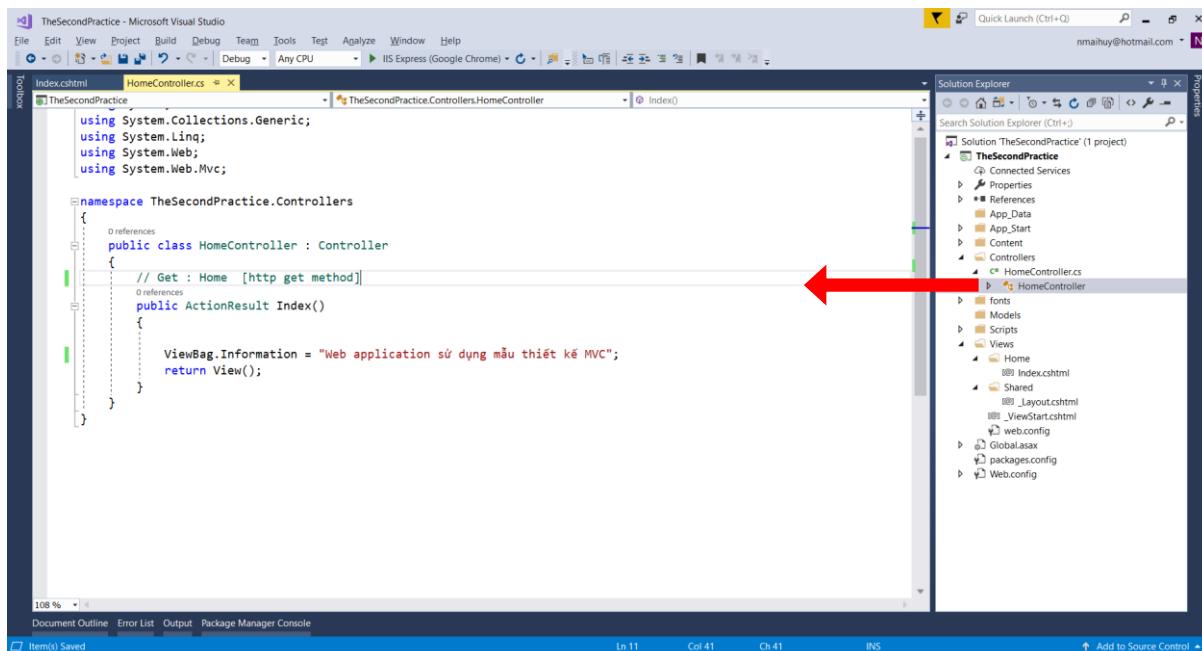
Để tạo controller cho dự án, hãy nhấn nút phải chuột vào folder có tên controllers trong khung Solution explorer, sau đó chọn “Add -> Controller ...”



Rồi đặt tên cho controller cần tạo là “Home”, lưu ý *hậu tố* của Controller muốn tạo

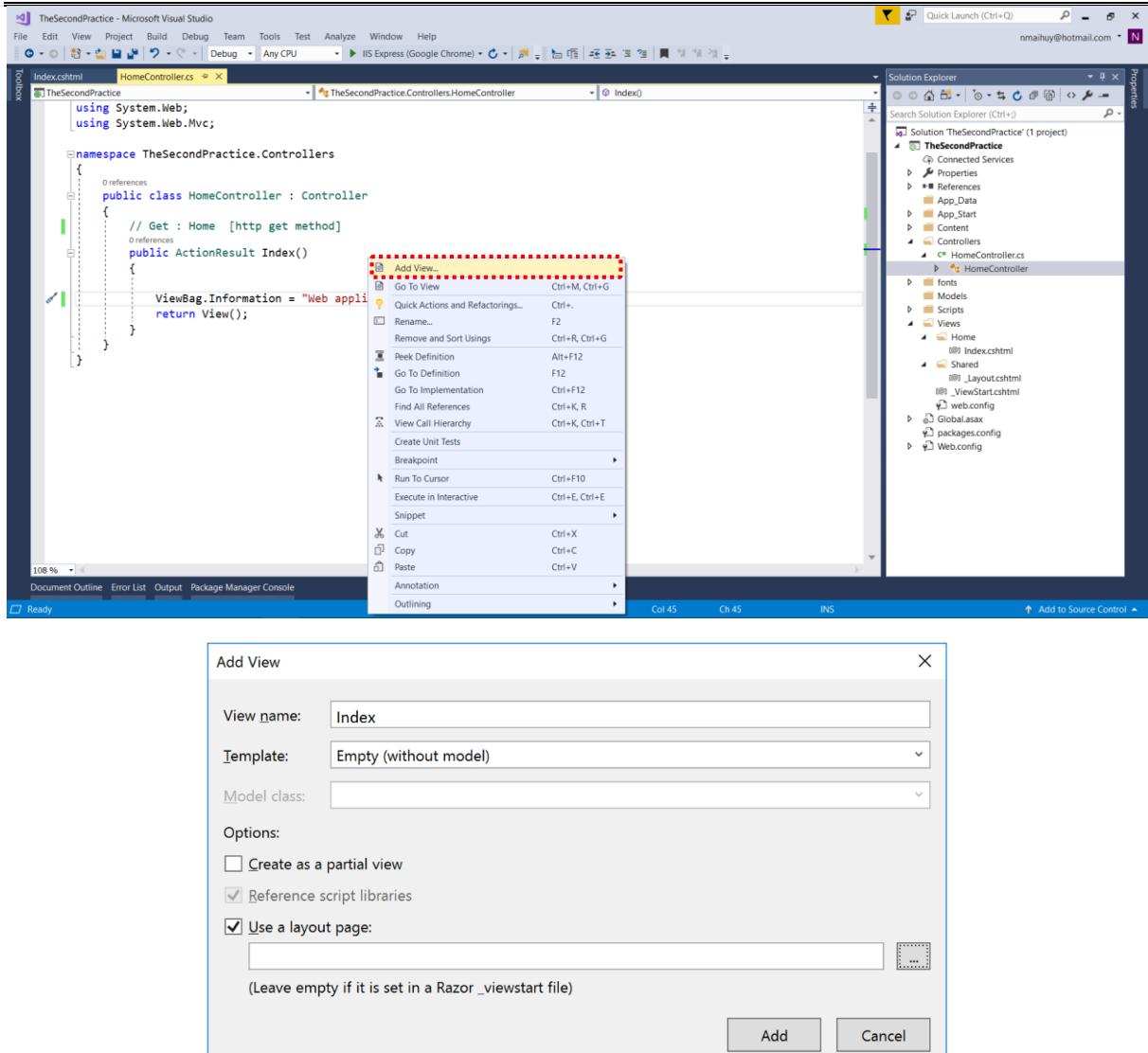


Giao diện trong design mode sau khi tạo thành công controller có dạng như sau

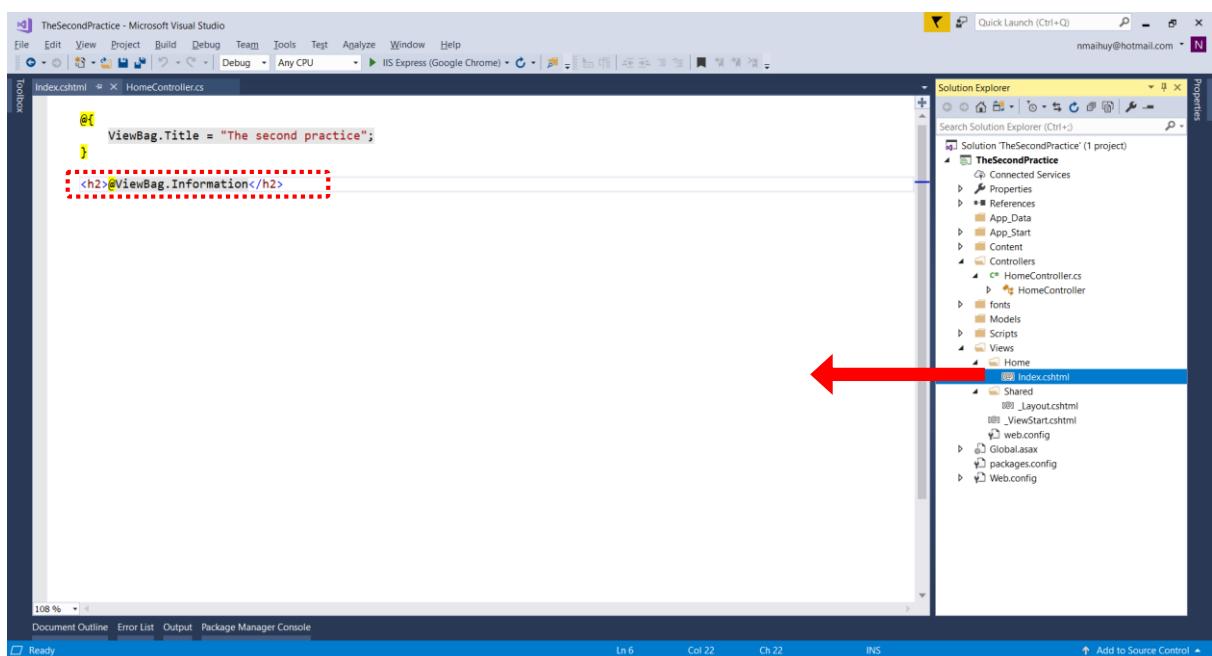


Để tạo View, nhấn nút phải chuột trên **ActionResult method**, sau đó chọn lệnh “**Add view ...**” từ Shortcut menu

Tiếp theo đặt tên cho view cần tạo (*Mặc nhiên, View cần tạo có tên là Index.cshtml*). Để nhận dữ liệu truyền từ Controller cho view, ta sẽ sử dụng ViewBag với thuộc tính là Information, (*Lưu ý dấu @ phía trước đối tượng ViewBag khi bạn muốn gọi trên View*). Hãy quan sát các hình minh họa ở trang bên



Giao diện trong Design mode của view khi tạo thành công có hình như minh họa sau

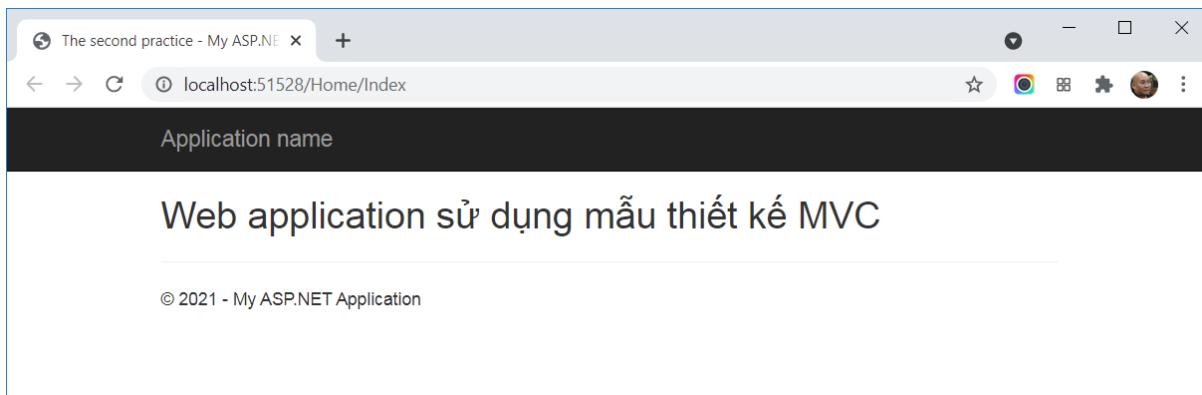


Ở minh họa này, tôi sử dụng tag h2 của HTML để trình bày thông tin được truyền từ controller dựa trên đối tượng ViewBag

4/- Chạy chương trình

Để biên dịch và chạy chương trình, có thể thực hiện một trong các cách sau

- Nhấn vào nút lệnh “IIS Express” trên thanh công cụ  (*Trên máy của bạn có thể chia tên của trình duyệt khác, không phải Google Chrome*)
- [menu] Debug \ Start debugging (*hoặc Start without debugging*)
- Nhấn F5 (*Hoặc Ctrl + F5*)



Lưu ý:

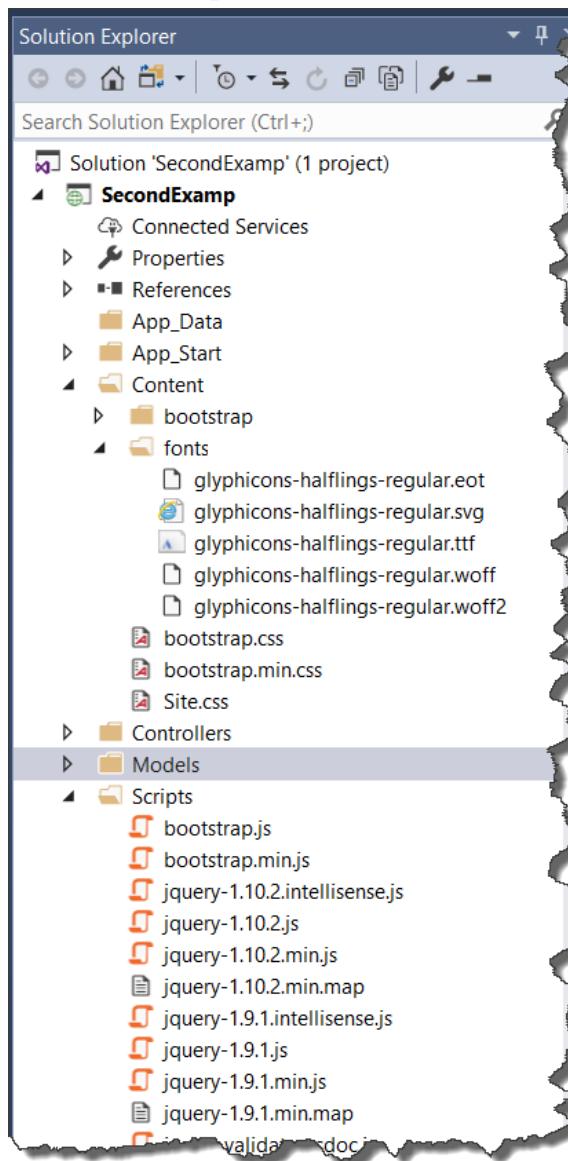
- Khi chạy chương trình web (*là dự án mà bạn đang làm việc*), quá trình này được chia làm 3 giai đoạn chính
 - o Khởi chạy & giả lập máy chủ web (IIS)
 - o Biên dịch chương trình & tải mã thực thi lên máy chủ web
 - o Chạy trình duyệt & tính năng tương ứng của project
- Do đó, có thể mất một khoảng thời gian nhất định trước khi thấy được chương trình của mình chạy trên trình duyệt. Khoảng thời gian này nhiều hay ít tùy thuộc vào mức độ dựa án của bạn lớn hay nhỏ (*có nhiều dữ liệu, tính năng, ...*)



Mục tiêu: Sử dụng *Html helper* cho việc xây dựng giao diện, *Model class*

Bài 1:

- Tạo ASP.NET Web application (*SecondExamp*)
- Cấu hình sử dụng Bootstrap framework cho ứng dụng web
- Cấu trúc của project **SecondExamp**



Yêu cầu:

- Sử dụng HTML helper để xây dựng các giao diện nhập dữ liệu (*hình trang bên*)
- Tạo các Model class hỗ trợ cho giao diện đã xây dựng
- Tạo các controllers cần thiết cho ứng dụng

User information New Pupil Student interface

Enter data for the student object in the program

You can add or edit the information of the Student

Name: Nguyễn Quang Hưng

Gender:

How old are you: 18

Your address here:

[New student](#)

[Back to List](#)

© 2021 - My ASP.NET Application

Giao diện nhập thông tin sinh viên

- Mã số sinh viên
- Họ tên
- Giới tính
- Tuổi
- Địa chỉ
- Quê quán

User information New Pupil Student interface

Pupil in the system

You can add or edit the information of the pupil

Name:

Age:

IsNewlyAccount:

Password:

DoB:

[Save](#)

[Back to List](#)

© 2021 - My ASP.NET Application

Giao diện nhập thông tin học sinh

- Mã học sinh
- Họ tên
- Tuổi
- Giới tính
- Địa chỉ
- Ngày sinh
- Tài khoản mới
- Mật khẩu

User information New Pupil Student interface

You can add or edit the information of the User

Account: nmquang

Password: mquang09

Name: Nguyễn Minh Quang

Age: 12

IsNewlyAccount:

IsActive:

IsDeleted:

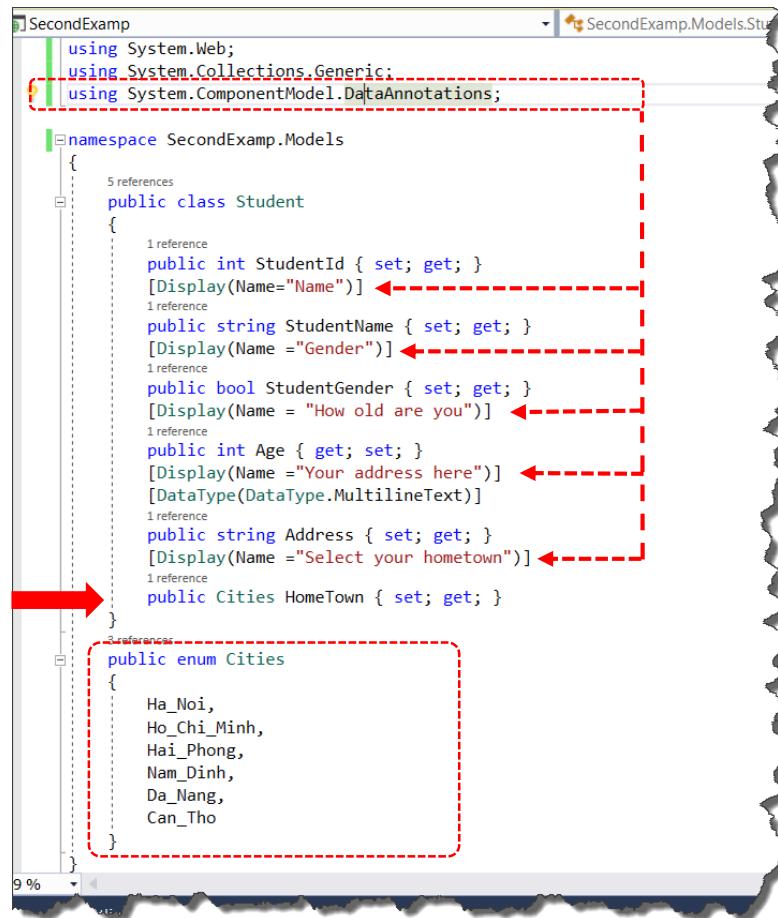
© 2021 - My ASP.NET Application

Giao diện thể hiện thông tin tài khoản

- Tên tài khoản
- Mật khẩu
- Họ tên chủ tk
- Tuổi
- Tài khoản mới
- Đã kích hoạt
- Đã xóa

I/- Model class

1 - Student class



```

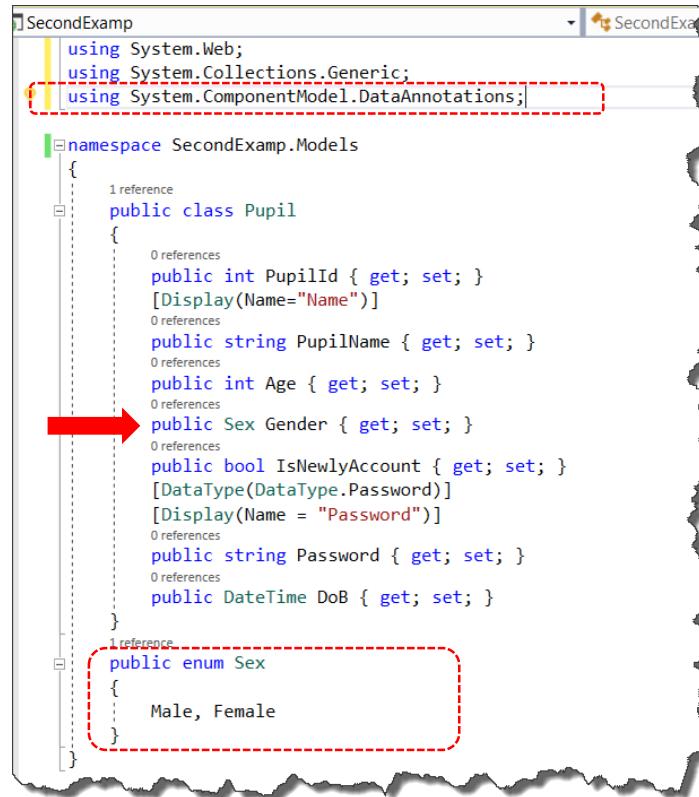
SecondExamp
using System.Web;
using System.Collections.Generic;
using System.ComponentModel.DataAnnotations;

namespace SecondExamp.Models
{
    public class Student
    {
        public int StudentId { set; get; }
        [Display(Name = "Name")]
        public string StudentName { set; get; }
        [Display(Name = "Gender")]
        public bool StudentGender { set; get; }
        [Display(Name = "How old are you")]
        public int Age { get; set; }
        [Display(Name = "Your address here")]
        [DataType(DataType.MultilineText)]
        public string Address { set; get; }
        [Display(Name = "Select your hometown")]
        public Cities HomeTown { set; get; }
    }

    public enum Cities
    {
        Ha_Noi,
        Ho_Chi_Minh,
        Hai_Phong,
        Nam_Dinh,
        Da_Nang,
        Can_Tho
    }
}

```

2 - Pupil class



```

SecondExamp
using System.Web;
using System.Collections.Generic;
using System.ComponentModel.DataAnnotations;

namespace SecondExamp.Models
{
    public class Pupil
    {
        public int PupilId { get; set; }
        [Display(Name = "Name")]
        public string PupilName { get; set; }
        public int Age { get; set; }
        public Sex Gender { get; set; }
        public bool IsNewlyAccount { get; set; }
        [DataType(DataType.Password)]
        [Display(Name = "Password")]
        public string Password { get; set; }
        public DateTime DoB { get; set; }
    }

    public enum Sex
    {
        Male, Female
    }
}

```

3 - User

```

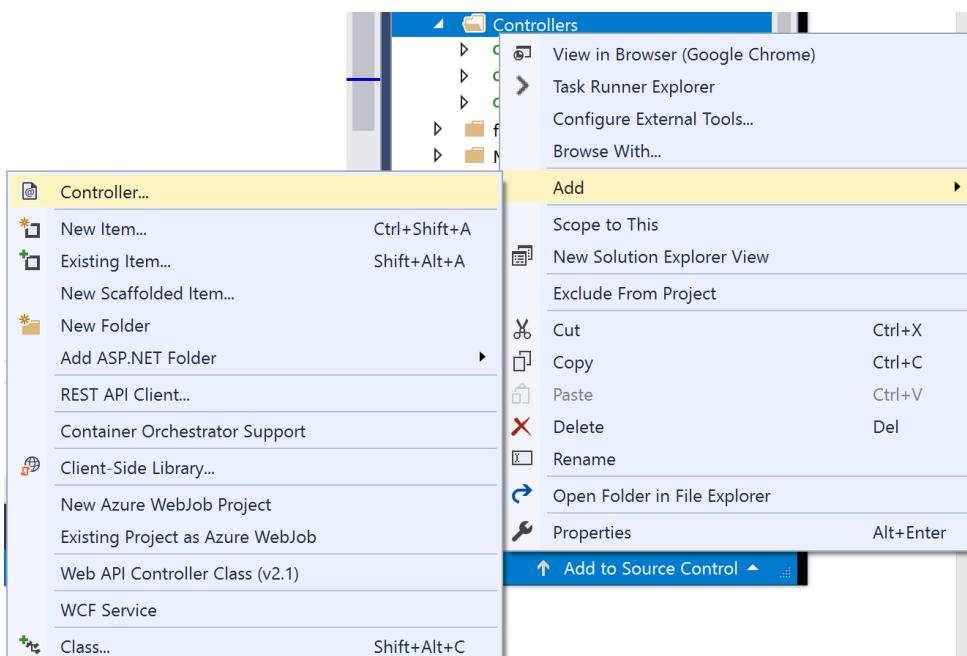
SecondExamp
using System.Web;
using System.Collections.Generic;
using System.ComponentModel.DataAnnotations;

namespace SecondExamp.Models
{
    public class User
    {
        public string Account { set; get; }
        public string Password { set; get; }
        [Display(Name ="Name")]
        public string UserName { set; get; }
        public int Age { set; get; }
        public bool IsNewlyAccount { set; get; }
        public bool IsActive { set; get; }
        public bool IsDeleted { set; get; }
    }
}

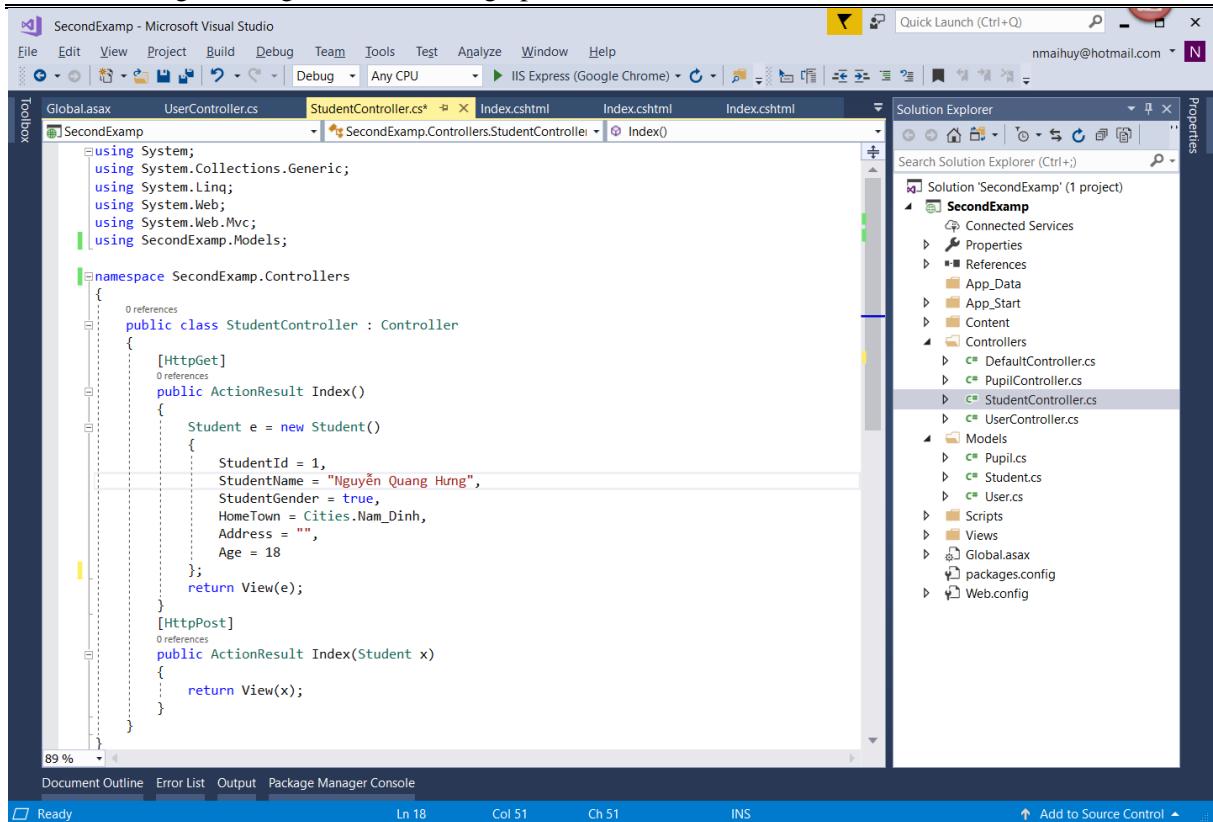
```

II/- Controller & View

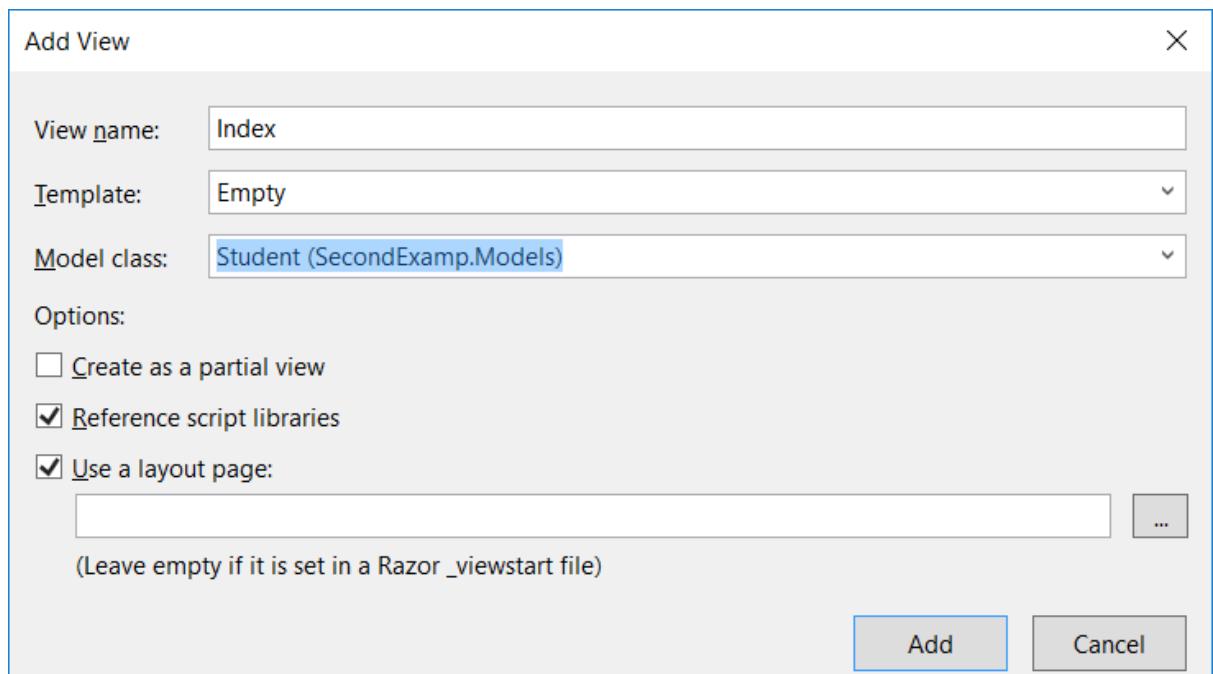
1 – Student :: Controller – View



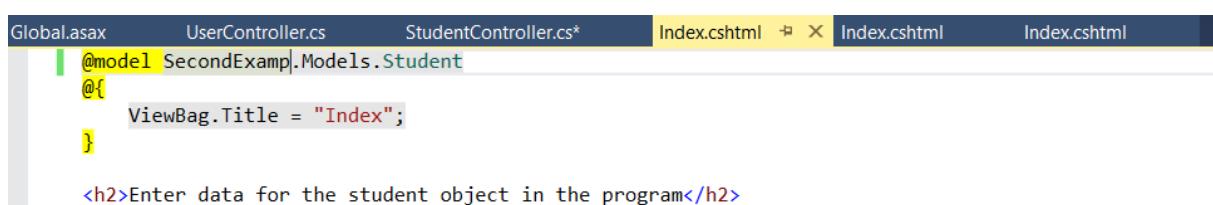
Đặt tên cho Controller là ***StudentController***, điều chỉnh nội dung với 2 **ActionResult** phục vụ cho các phương thức **Get** và **Post** dựa trên tham số truyền (*Sử dụng các Anotation **HttpGet** và **HttpPost***)



Để tạo view, ta nhấp nút phải vào ActionResult method và chọn lệnh Add view, và sử dụng **Strongly typed helper** tại mục chọn Model class



Lúc này, View được tạo bởi **Strongly typed helper** có nội dung như hình minh họa sau
(Lưu ý khai báo @model để truy xuất model type ở trên view)



```

@using (Html.BeginForm())
{
    @Html.AntiForgeryToken()
    <div class="form-horizontal">
        <hr />
        @Html.ValidationSummary(true, "", new { @class = "text-danger" })
        <div class="form-group">
            @Html.LabelFor(model => model.StudentName, htmlAttributes: new { @class = "control-label col-md-2" })
            <div class="col-md-10">
                @Html.EditorFor(model => model.StudentName, new { htmlAttributes = new { @class = "form-control" } })
                @Html.ValidationMessageFor(model => model.StudentName, "", new { @class = "text-danger" })
            </div>
        </div>
        <div class="form-group">
            @Html.LabelFor(model => model.StudentGender, htmlAttributes: new { @class = "control-label col-md-2" })
            <div class="col-md-10">
                <div class="checkbox">
                    @Html.EditorFor(model => model.StudentGender)
                    @Html.ValidationMessageFor(model => model.StudentGender, "", new { @class = "text-danger" })
                </div>
            </div>
        </div>
        <div class="form-group">
            @Html.LabelFor(model => model.Age, htmlAttributes: new { @class = "control-label col-md-2" })
            <div class="col-md-10">
                @Html.EditorFor(model => model.Age, new { htmlAttributes = new { @class = "form-control" } })
                @Html.ValidationMessageFor(model => model.Age, "", new { @class = "text-danger" })
            </div>
        </div>
        <div class="form-group">
            @Html.LabelFor(model => model.Address, htmlAttributes: new { @class = "control-label col-md-2" })
            <div class="col-md-10">
                @Html.EditorFor(model => model.Address, new { htmlAttributes = new { @class = "form-control" } })
                @Html.ValidationMessageFor(model => model.Address, "", new { @class = "text-danger" })
            </div>
        </div>
        <div class="form-group">
            <div class="col-md-offset-2 col-md-10">
                <input type="submit" value="Create" class="btn btn-default" />
            </div>
        </div>
    </div>
    <div>
        @Html.ActionLink("Back to List", "Index")
    </div>
<script src "~/Scripts/jquery-1.10.2.min.js"></script>
<script src "~/Scripts/jquery.validate.min.js"></script>
<script src "~/Scripts/jquery.validate.unobtrusive.min.js"></script>

```

2 – Pupil :: Controller – View

Sinh viên tự thực hiện tạo Controller và View cho Model học sinh (*Pupil class*)

3 – User :: Controller - View

Đối với model User, chúng ta làm quen với việc tạo view, sử dụng templated html helper và truyền dữ liệu từ controller cho view như hướng dẫn sau

- Anotation cho ActionResult là **HttpPost**
- Tạo 1 object của lớp user và thiết lập thông tin cho object này
- Chuyển object vừa tạo cho view dưới dạng tham số thông qua lệnh return View(object)

Để mô tả cho việc tạo ActionResult phục vụ cho việc truyền dữ liệu là một model object từ controller ra view, Hãy quan sát thông qua hình minh họa sau đây

The screenshot shows the Microsoft Visual Studio interface. The code editor displays the `UserController.cs` file, which contains the following code:

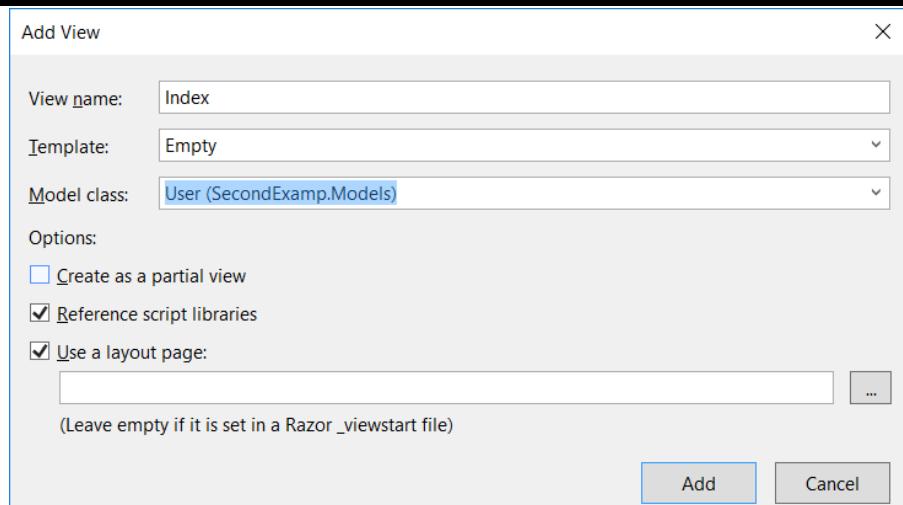
```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.Mvc;
using SecondExamp.Models;

namespace SecondExamp.Controllers
{
    public class UserController : Controller
    {
        [HttpPost]
        public ActionResult Index()
        {
            User x = new User()
            {
                Account = "nmquang",
                Password = "nmquang09",
                UserName = "Nguyễn Minh Quang",
                Age = 12,
                IsNewlyAccount = true,
                IsDeleted = false,
                IsActivated = true
            };
            return View(x);
        }
    }
}

```

The Solution Explorer on the right shows the project structure for 'SecondExamp' with files like DefaultController.cs, PupilController.cs, StudentController.cs, and UserController.cs.



The screenshot shows the Microsoft Visual Studio interface. The code editor displays the `Index.cshtml` file, which contains the following code:

```

@model SecondExamp.Models.User

@{
    ViewBag.Title = "The user information";
}

<style>
    label {
        text-align: right;
    }
    .text-box {
        display: block; width: 100%; height: 34px; padding: 6px 12px;
        font-size: 14px; line-height: 1.428571429; color: #555; vertical-align: middle;
        background-color: #fff; border: 1px solid #ccc; border-radius: 4px;
        -webkit-box-shadow: inset 0 1px 1px rgba(0,0,0,.8%);
        box-shadow: inset 0 1px 1px rgba(0,0,0,.8%);
        -webkit-transition: border-color ease-in-out .15s,box-shadow ease-in-out .15s;
        transition: border-color ease-in-out .15s,box-shadow ease-in-out .15s;
    }
</style>
<h2>You can add or edit the information of the User</h2>
@using (Html.BeginForm())
{
    @Html.EditorForModel()
}

```

The Solution Explorer on the right shows the project structure for 'SecondExamp' with files like Default.cshtml, _ViewStart.cshtml, web.config, Global.asax, packages.config, and Web.config.

Lưu ý đổi với việc định dạng cho class trên giao diện khi dùng EditorForModel (*quan sát trong chế độ Inspect element*), chúng ta có thể thực hiện style css đổi với các css class như: “editor-label”, “editor-field”, ...

The screenshot shows a browser window with the URL `localhost:49855/User/Index`. The page title is "User information". The main content area displays a form for adding or editing user information. The form fields include:

- Account**: A text input field containing "nmquang".
- Password**: A text input field containing "mquang09".
- Name**: A text input field containing "Nguyễn Minh Quang".
- Age**: A text input field containing "12".
- IsNewlyAccount**: A checked checkbox.
- IsActive**: A checked checkbox.
- IsDeleted**: An unchecked checkbox.

At the bottom of the form, there is a copyright notice: "© 2021 - My ASP.NET Application".

To the right of the browser window, the developer tools are open, specifically the "Elements" tab. The "Styles" panel is visible, showing the CSS rules applied to the selected element. The element being inspected is a label for the "Name" input field, which has the class "editor-label". The CSS rule for this label is:

```
label {
    font-family: "Helvetica Neue", Helvetica, Arial, sans-serif;
    font-size: 14px;
    line-height: 1.428571429;
    color: #333;
    background-color: #fff;
}
```

The developer tools also show the full HTML structure of the page, including the navigation bar and the form itself.



Mục tiêu: Mở rộng các thành phần chính trong MVC Design pattern

Bài 1:

Hãy tạo một ứng dụng web phục vụ cho việc tính toán diện tích, chu vi của các hình cơ bản trong hình học phẳng: **Hình tròn**, **Hình chữ nhật**, **Hình tam giác**. Giao diện của chương trình bao gồm 4 trang chức năng như các hình minh họa tại các mục 1.1, 1.2, 1.3, 1.4. Để hoàn thành bài tập này, sinh viên cần thực hiện các công việc sau:

- Tạo ASP.NET Web application
- Tạo các thành phần Model – View – Controller
- Tạo ứng dụng minh họa cho việc tính toán
- Tham khảo hướng dẫn để hiểu rõ cách thực hiện đối với các thành phần Model – View – Controller của project

1.1/- Default page

The Model-View-Controller (MVC) architectural pattern separates an application into three main groups of components: Models, Views, and Controllers. This pattern helps to achieve separation of concerns. Using this pattern, user requests are routed to a Controller which is responsible for working with the Model to perform user actions and/or retrieve results of queries. The Controller chooses the View to display to the user, and provides it with any Model data it requires.

This delineation of responsibilities helps you scale the application in terms of complexity because it's easier to code, debug, and test something (model, view, or controller) that has a single job. It's more difficult to update, test, and debug code that has dependencies spread across two or more of these three areas. For example, user interface logic tends to change more frequently than business logic. If presentation code and business logic are combined in a single object, an object containing business logic must be modified every time the user interface is changed. This often introduces errors and requires the retesting of business logic after every minimal user interface change.

© 2019 - My ASP.NET Application

1.2/- Circle page

Input your radius

Calculate Continue ...

© 2019 - My ASP.NET Application

1.3/- Triangle page

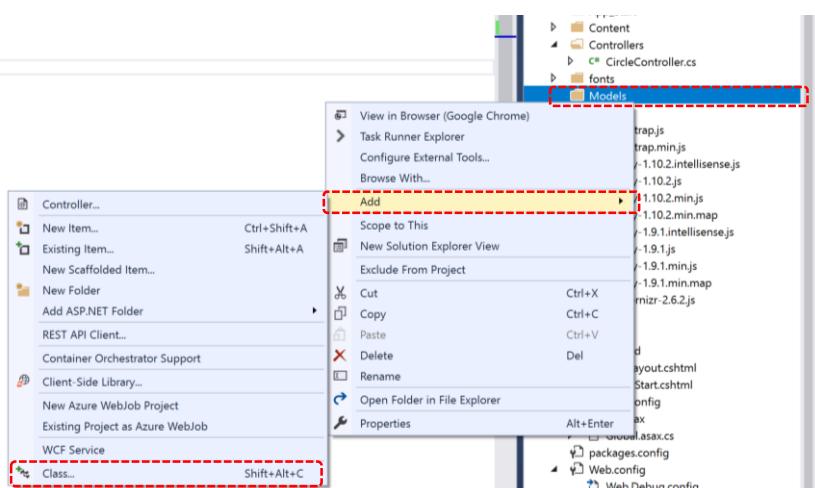
The screenshot shows a web browser window titled "Index - My ASP.NET Application" with the URL "localhost:51581/Triangle". The page has a header with links: "Home page", "Circle shape", "Triangle shape", and "Rectangle shape". Below the header, the title "Acreage - Perimeter of the triangle" is displayed. There are three input fields labeled "The first edge", "The second edge", and "The third edge", each with a corresponding text input box. At the bottom are two buttons: a blue "Calculate" button and a white "Continue ..." button.

1.4/- Rectangle page

The screenshot shows a web browser window titled "Index - My ASP.NET Application" with the URL "localhost:51581/Rectangle". The page has a header with links: "Home page", "Circle shape", "Triangle shape", and "Rectangle shape". Below the header, the title "Acreage - Perimeter of the rectangle" is displayed. There are two input fields labeled "Width of the rectangle" and "Length of the rectangle", each with a corresponding text input box. At the bottom are two buttons: a blue "Calculate" button and a white "Continue ..." button. At the very bottom of the page, there is a copyright notice: "© 2019 - My ASP.NET Application".

I/- Model class

Nhấn nút phải chuột vào folder Models có trong vùng làm việc Solution explorer và chọn lệnh Add -> Class. Sau đó đặt tên cho class cần tạo, đồng thời định nghĩa các thành phần của class (*properties, methods*)



1 – Circle class

The screenshot shows the Microsoft Visual Studio interface. The main window displays the `Circle.cs` file under the project `TheThirdPractice`. The code defines a `Circle` class with properties for radius and methods to calculate area and perimeter. The Solution Explorer on the right shows the project structure, including the `Models` folder which contains `Circle.cs`, `Rectangle.cs`, and `Triangle.cs`.

```

using System;
using System.Linq;
using System.Web;
using System.Collections.Generic;

namespace TheThirdPractice.Models
{
    /// <summary> Circle Model :: Create by Nguyễn Mai Huy
    public class Circle
    {
        // 6 references
        public double Radius { get; set; }
        /// <summary>
        /// Default constructor
        /// </summary>
        public Circle()
        {
            this.Radius = 0;
        }
        /// <summary> The method to calculate the square of the Circle
        public double acreage()
        {
            return this.Radius * this.Radius * Math.PI;
        }
        /// <summary> Phương thức cho phép tính chu vi của hình tròn dựa vào công thức d ...
        public double perimeter()
        {
            return 2 * this.Radius * Math.PI;
        }
    }
}

```

2 – Rectangle class

The screenshot shows the Microsoft Word ribbon. The document contains the XML documentation for the `Rectangle` class, which includes summaries for the class itself and its properties (`width` and `height`), a constructor, and methods for calculating area and perimeter.

```

/// <summary>
/// Rectangle Model :: Create by Nguyễn Mai Huy
/// </summary>
public class Rectangle
{
    /// <summary>
    /// Chiều rộng hình chữ nhật
    /// </summary>
    public int width { get; set; }
    /// <summary> Chiều dài hình chữ nhật
    public int height { get; set; }
    public Rectangle() ...
    /// <summary>
    /// Phương thức cho phép tính diện tích của hình chữ nhật
    /// </summary>
    /// <returns></returns>
    public int acreage()
    {
        return this.width * this.height ;
    }
    /// <summary>
    /// Phương thức cho phép tính chu vi của hình chữ nhật
    /// </summary>
    /// <returns></returns>
    public double perimeter()
    {
        return 2 * (this.width+this.height);
    }
}

```

3 – Triangle class

```

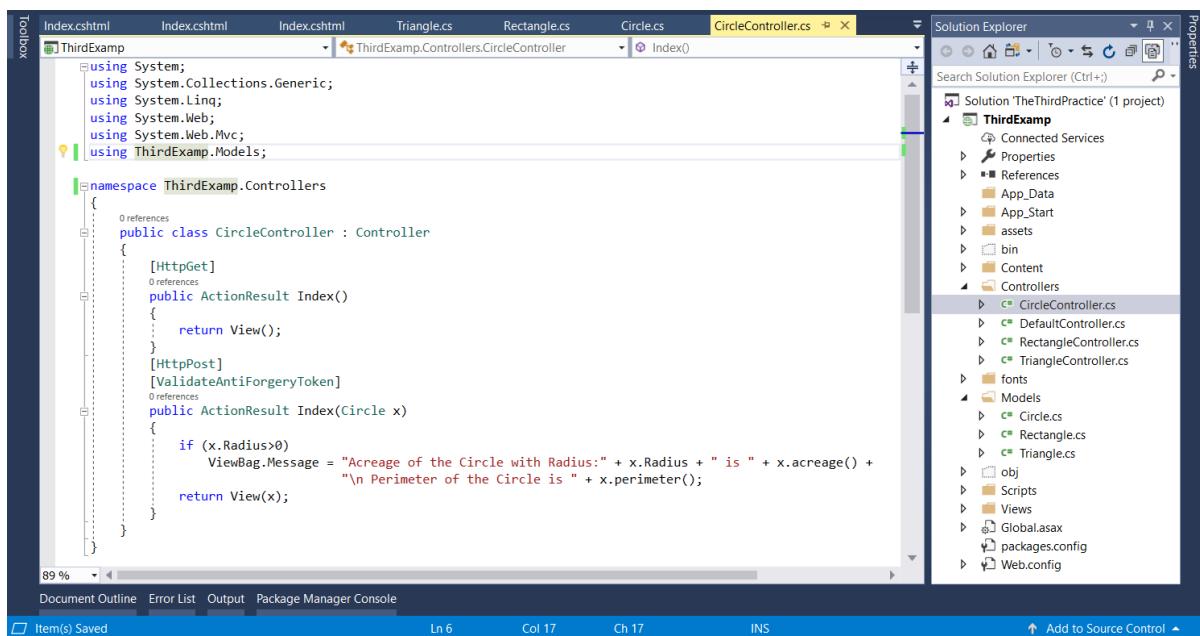
/// <summary>
/// Triangle Model :: Create by Nguyễn Mai Huy
/// </summary>
1 reference
public class Triangle
{
    /// <summary> Thông số cho cạnh thứ Nhất của Tam giác
    5 references
    public float firstEdge { set; get; }
    /// <summary> Thông số cho cạnh thứ Hai của Tam giác
    5 references
    public float secondEdge { set; get; }
    /// <summary> Thông số cho cạnh thứ Ba của Tam giác
    5 references
    public float thirdEdge { set; get; }
    /// <summary> Default constructor
    0 references
    public Triangle()...
    /// <summary> Kiểm tra xem 3 cạnh có thỏa mãn đối với 1 tam giác hay không
    0 references
    public bool isTriangle()
    {
        return (this.firstEdge + secondEdge > this.thirdEdge) &&
               (this.secondEdge + this.thirdEdge > firstEdge) &&
               (this.firstEdge + this.thirdEdge > this.secondEdge);
    }
    /// <summary> Phương thức cho phép tính chu vi của hình tam giác
    1 reference
    public float perimeter()
    {
        return (this.firstEdge+this.secondEdge+this.thirdEdge);
    }
    /// <summary> Phương thức cho phép tính diện tích của hình tam giác
    0 references
    public float acreage()
    {
        float p = this.perimeter();
        return (float)Math.Sqrt(p*(p-firstEdge)*(p-secondEdge)*(p-thirdEdge));
    }
}

```

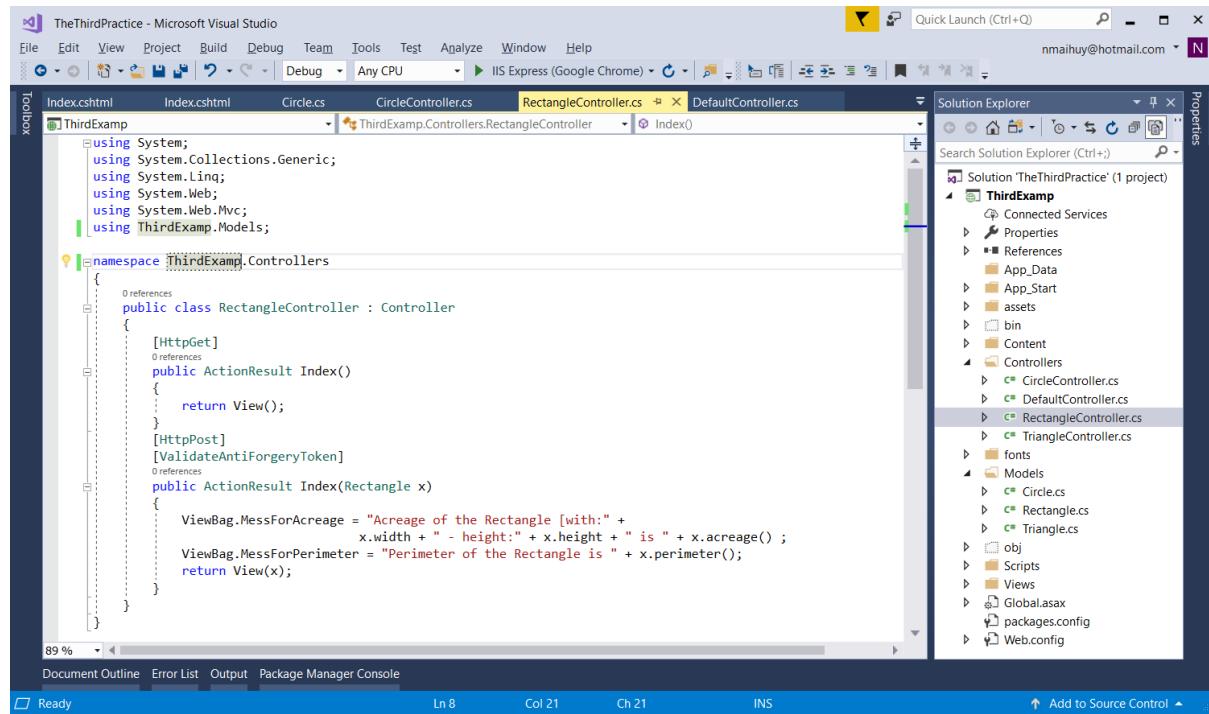
II/- Controllers

Đối với controller của một mục tiêu, chúng ta sẽ sử dụng 2 **ActionResult** khác nhau cho 2 phương thức thương tác giữa client và server là **HttpGet** vào **HttpPost**. Ngoài ra, để truyền thông tin cho View, ta sẽ sử dụng đối tượng **ViewBag**, riêng với **ActionResult** phục vụ cho **HttpPost**, chúng ta sẽ khai báo Anotation **ValidateAntiForgeryToken** cho phương thức này (*Sinh viên cần trả lời câu hỏi vì sao nên có 2 ActionResult phục vụ cho Get – Post và ValidateAntiForgeryToken trong giới lý thuyết*)

1 – CircleController



2 – RectangleController



The screenshot shows the Microsoft Visual Studio interface with the 'TheThirdPractice' project open. The 'RectangleController.cs' file is selected in the Solution Explorer. The code defines a controller for rectangles:

```

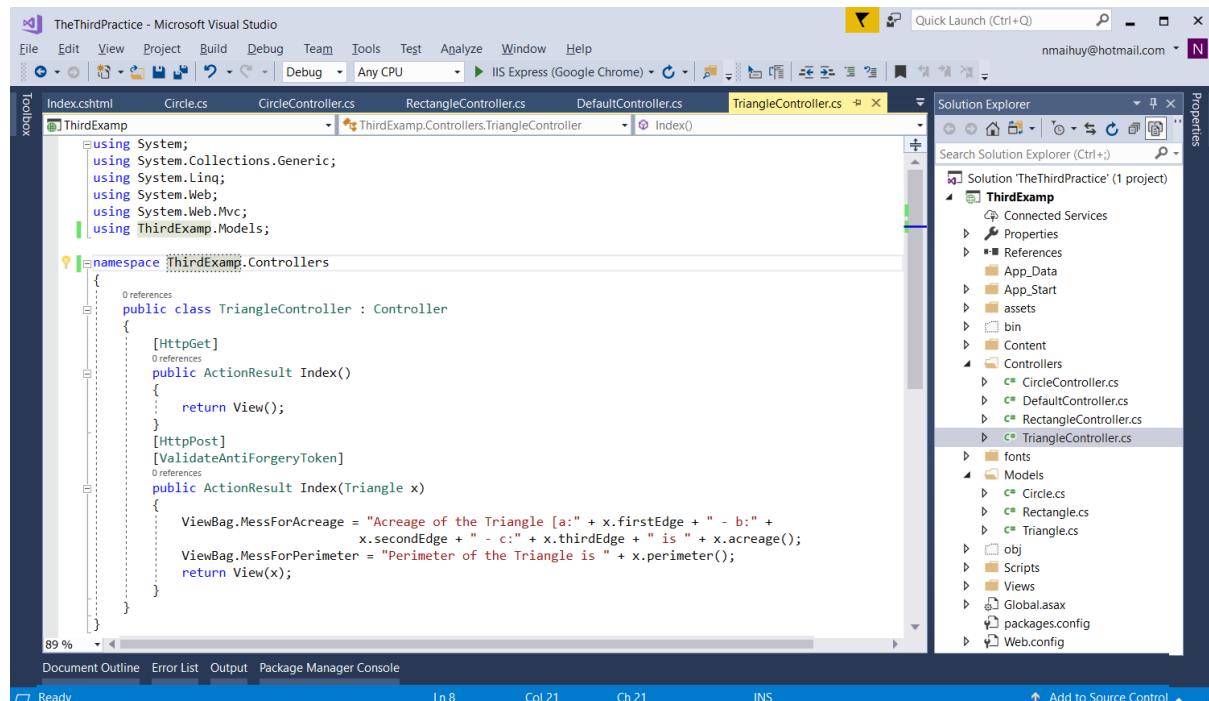
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.Mvc;
using ThirdExam.Models;

namespace ThirdExam.Controllers
{
    public class RectangleController : Controller
    {
        [HttpGet]
        public ActionResult Index()
        {
            return View();
        }

        [HttpPost]
        [ValidateAntiForgeryToken]
        public ActionResult Index(Rectangle x)
        {
            ViewBag.MessForAcreage = "Acreage of the Rectangle [with:" + x.width + " - height:" + x.height + " is " + x.acreage() + "]";
            ViewBag.MessForPerimeter = "Perimeter of the Rectangle is " + x.perimeter();
            return View(x);
        }
    }
}

```

3 – TriangleController



The screenshot shows the Microsoft Visual Studio interface with the 'TriangleController.cs' file selected in the Solution Explorer. The code defines a controller for triangles:

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.Mvc;
using ThirdExam.Models;

namespace ThirdExam.Controllers
{
    public class TriangleController : Controller
    {
        [HttpGet]
        public ActionResult Index()
        {
            return View();
        }

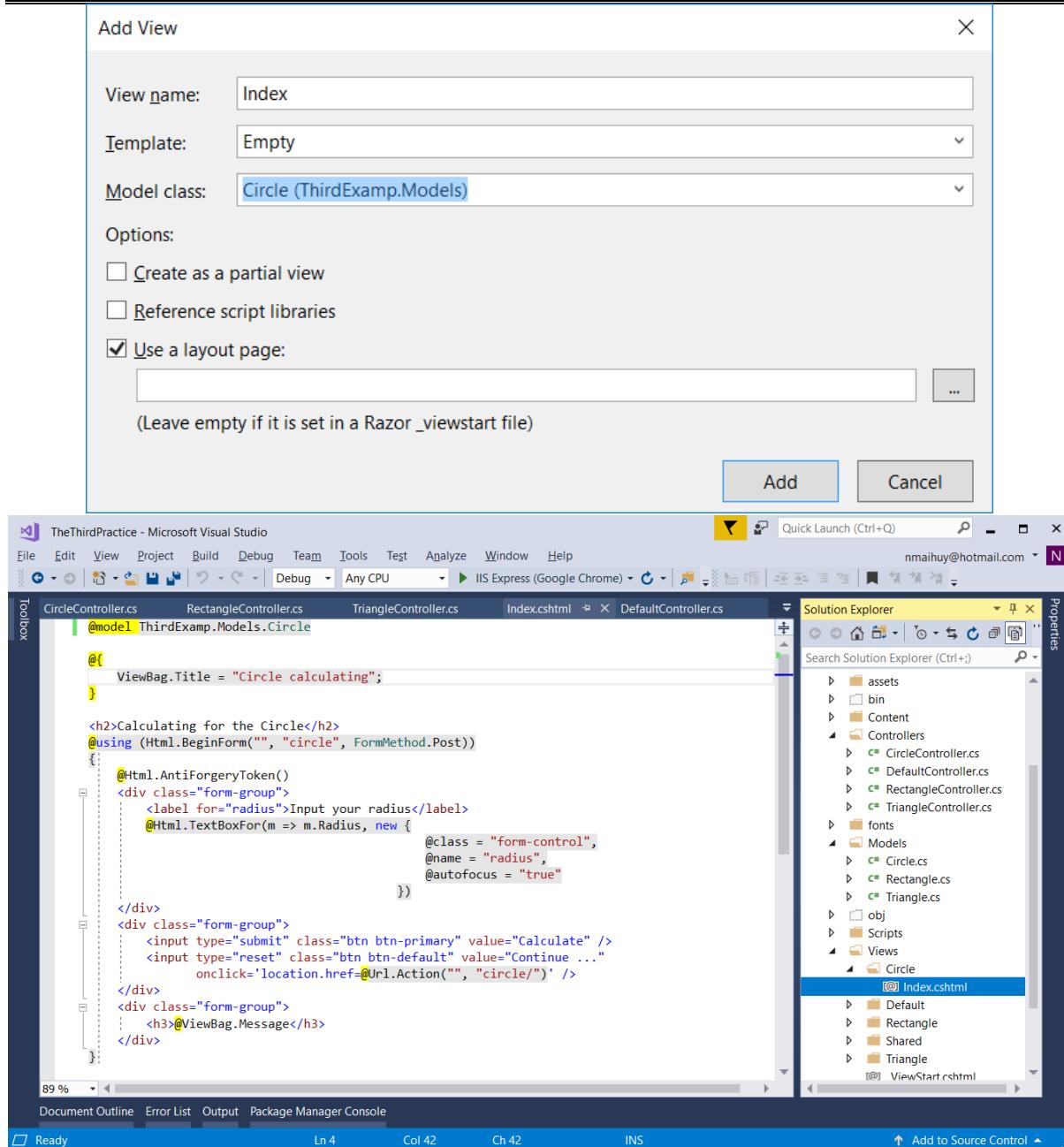
        [HttpPost]
        [ValidateAntiForgeryToken]
        public ActionResult Index(Triangle x)
        {
            ViewBag.MessForAcreage = "Acreage of the Triangle [a:" + x.firstEdge + " - b:" + x.secondEdge + " - c:" + x.thirdEdge + " is " + x.acreage() + "]";
            ViewBag.MessForPerimeter = "Perimeter of the Triangle is " + x.perimeter();
            return View(x);
        }
    }
}

```

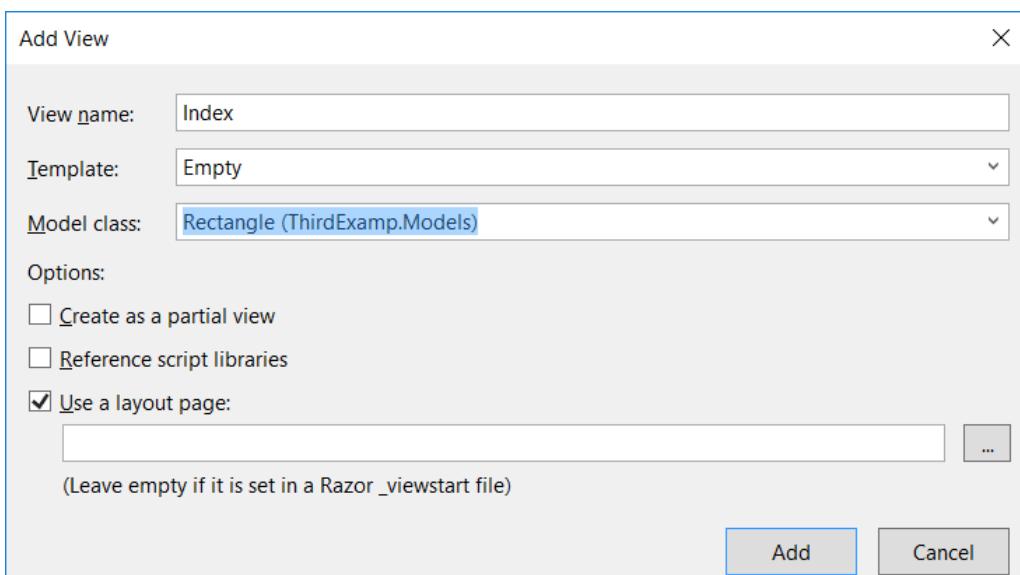
III/- Views

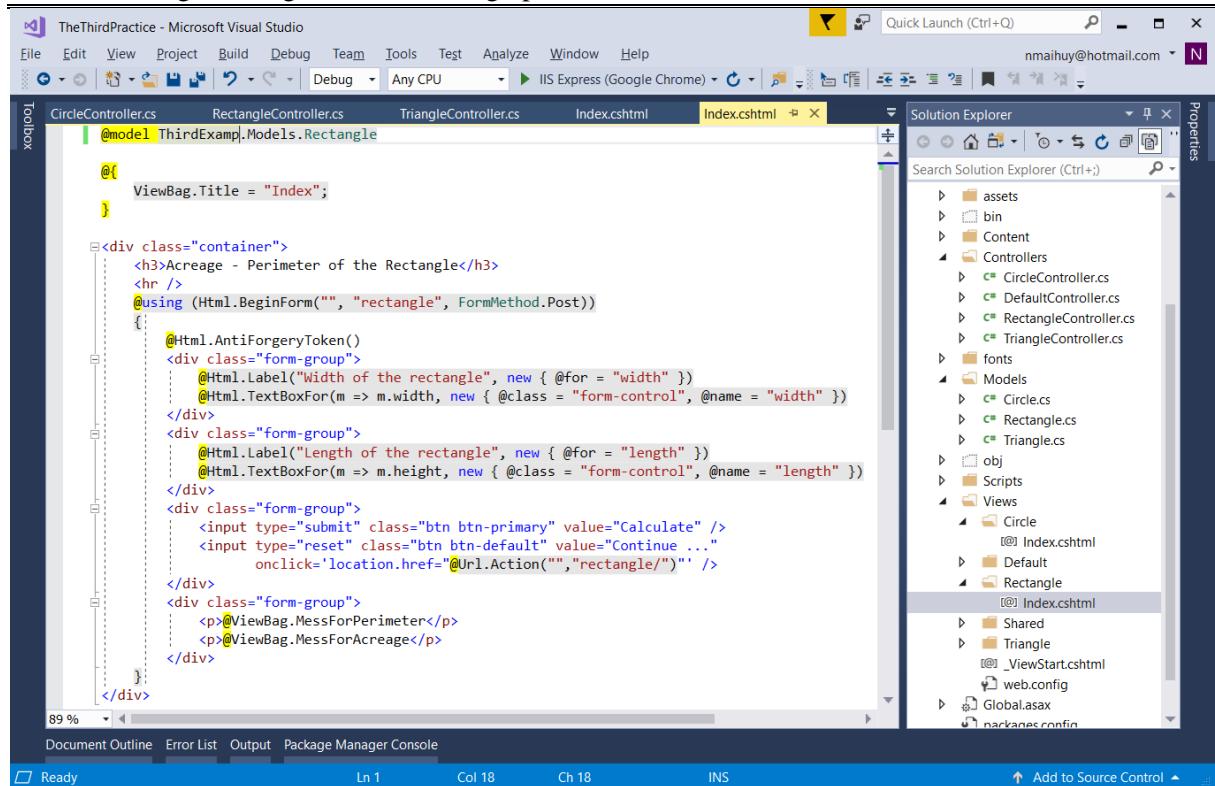
Để tạo view cho chức năng, tại ActionResult trong controller tương ứng, nhấn nút phải chuột chọn Add view

1 – Circle view

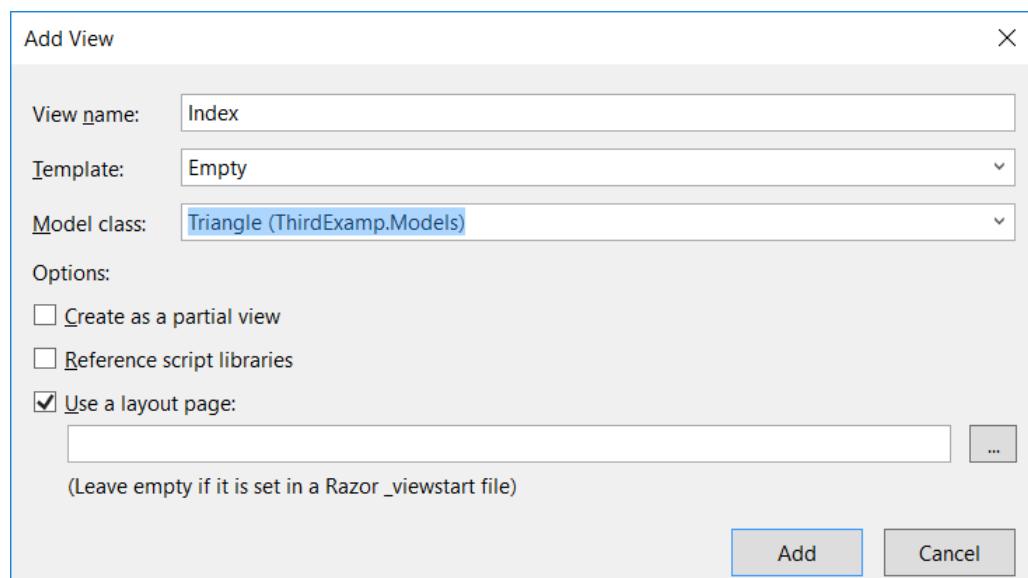


2 – Rectangle view





3 – Triangle View



```

@model ThirdExam.Models.Triangle

@{ ViewBag.Title = "Index"; }



### Acreage - Perimeter of the triangle</h3><hr /> @using (Html.BeginForm("", "triangle", FormMethod.Post)) { @Html.AntiForgeryToken() <div class="form-group"> @Html.Label("The first edge", new { @for = "first" }) @Html.TextBoxFor(m => m.firstEdge, new { @class = "form-control", @name = "first" }) </div> <div class="form-group"> @Html.Label("The second edge", new { @for = "second" }) @Html.TextBoxFor(m => m.secondEdge, new { @class = "form-control", @name = "second" }) </div> <div class="form-group"> @Html.Label("The third edge", new { @for = "third" }) @Html.TextBoxFor(m => m.thirdEdge, new { @class = "form-control", @name = "third" }) </div> <div class="form-group"> <input type="submit" class="btn btn-primary" value="Calculate" /> <input type="reset" class="btn btn-default" value="Continue ..." onclick="location.href=@Url.Action("", "triangle")" /> </div> <div class="form-group"> <p>@ViewBag.MessForPerimeter</p> <p>@ViewBag.MessForAcreage</p> </div> }


```

4 - _Layout – Default

```

<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8" />
        <meta name="viewport" content="width=device-width, initial-scale=1.0" />
        <title>@ViewBag.Title - My ASP.NET Application</title>
        <link href="~/Content/Site.css" rel="stylesheet" type="text/css" />
        <link href="~/Content/bootstrap.min.css" rel="stylesheet" type="text/css" />
        <script src="~/Scripts/modernizr-2.6.2.js"></script>
    </head>
    <body>
        <div class="navbar navbar-inverse navbar-fixed-top">
            <div class="container">
                <div class="navbar-header">
                    <a class="navbar-brand" href="#">Home page
                    <a class="navbar-brand" href="#">Circle shape
                    <a class="navbar-brand" href="#">Triangle shape
                    <a class="navbar-brand" href="#">Rectangle shape
                </div>
                <div class="navbar-collapse collapse">
                    <ul class="nav navbar-nav">
                        <li><a href="#">Home</a></li>
                        <li><a href="#">Circle</a></li>
                        <li><a href="#">Triangle</a></li>
                        <li><a href="#">Rectangle</a></li>
                    </ul>
                </div>
            </div>
        </div>
        <div class="container body-content">
            @RenderBody()
            <hr />
            <footer>...</footer>
        </div>
        <script src="~/Scripts/jquery-1.10.2.min.js"></script>
        <script src="~/Scripts/bootstrap.min.js"></script>
    </body>
</html>

```

The screenshot shows the Microsoft Visual Studio interface with the title "TheThirdPractice - Microsoft Visual Studio". The main window displays the code for "Index.cshtml" under the "Views/Default" folder. The code includes a ViewBag assignment and a section explaining the MVC pattern's separation of concerns. The Solution Explorer on the right shows the project structure with files like "CircleController.cs", "DefaultController.cs", "RectangleController.cs", "TriangleController.cs", "Circle.cs", "Rectangle.cs", and "Triangle.cs" in the "Models" folder, along with "Index.cshtml" files for each view.

```

@{
    ViewBag.Title = "Index";
}



## Example to calculate the Acreage - Perimeter of the basic shape



The Model-View-Controller (MVC) architectural pattern separates an application into three main groups of components: Models, Views, and Controllers. This pattern helps to achieve separation of concerns. Using this pattern, user requests are routed to a Controller which is responsible for working with the Model to perform user actions and/or retrieve results of queries. The Controller chooses the View to display to the user, and provides it with any Model data it requires.



This delineation of responsibilities helps you scale the application in terms of complexity because it's easier to code, debug, and test something (model, view, or controller) that has a single job. It's more difficult to update, test, and debug code that has dependencies spread across two or more of these three areas. For example, user interface logic tends to change more frequently than business logic. If presentation code and business logic are combined in a single object, an object containing business logic must be modified every time the user interface is changed. This often introduces errors and requires the retesting of business logic after every minimal user interface change.






```

Fourth week :: 4

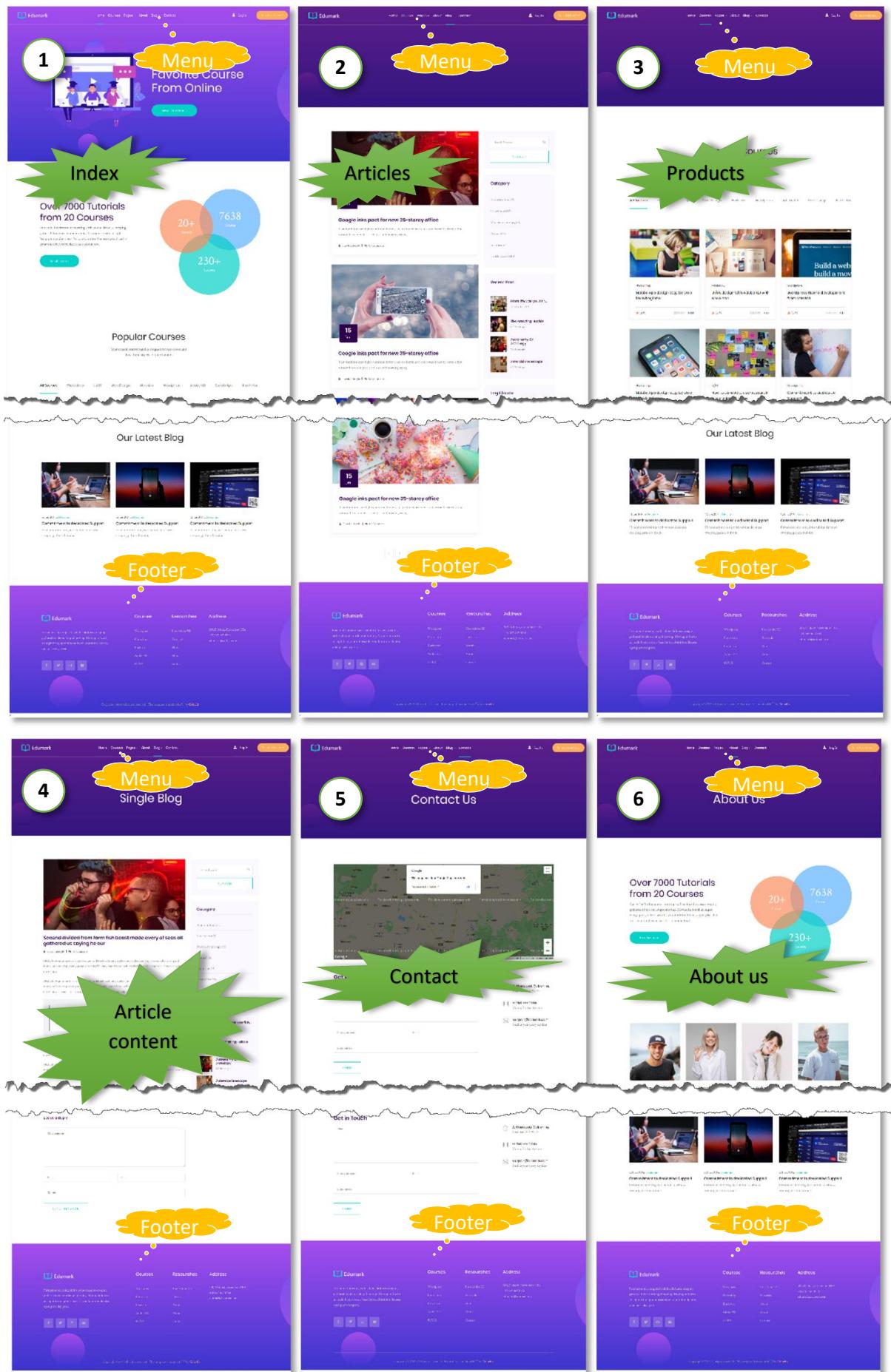
Mục tiêu: Áp dụng template cho dự án web [Public pages – Private pages]

Bài 1:

Yêu cầu của bài tập này hướng đến việc triển khai một mẫu thiết kế đã được thực hiện ở Front-end cho ứng dụng web trong thực tế, cho project xây dựng bởi ASP.NET MVC

- Phân tích và thiết kế các chức năng cần thiết cho một ứng dụng với giao diện chia làm 2 phần
 - o **Public pages:** Bao gồm các chức năng phục vụ cho mục tiêu “công khai”, tức là tất cả người dùng truy cập từ internet đều có thể truy xuất và sử dụng được các chức năng này. VD: Trang chủ chủ của ứng dụng web, các trang đưa tin: nội dung, danh sách bài viết, sản phẩm ..., các trang hỗ trợ người dùng: FAQs, Feedback, Contact, About us, ...
 - o **Private pages, Administrative pages :** các trang chức năng phục vụ cho mục đích quản trị, các chức năng này chỉ có thể sử dụng sau khi người dùng đã xác thực danh tính (*đăng nhập*) và đã được cấp quyền, để làm thực hiện một số công việc như: Duyệt bài, Đăng tin (*Upload bài viết*), cập nhật thông tin sản phẩm, danh sách đơn hàng, danh sách khách hàng, ...
- Chuẩn bị giao diện bằng một trong hai cách sau
 - o **Xây dựng** 2 bộ giao diện để sử dụng cho các chức năng thuộc nhóm public pages và private page (*nếu sinh viên muốn tự thực hiện dựa trên kiến thức và kỹ năng đã học trong web layout*)
 - o **Tìm kiếm** giao diện miễn phí (*free template*) từ cộng đồng web thông qua internet, có thể tìm kiếm các mẫu giao diện được cộng đồng thiết kế web cung cấp miễn phí (*có thể mua bản có tính phí nếu có điều kiện*) thông qua các từ khóa VD: “*nice web templates*”, “*web templates for sale*”, “*administrative website templates for free*” ... tùy theo mục tiêu cần thực hiện của mỗi người
- Thiết kế & xây dựng các trang chức năng phục vụ cho cả 2 mục tiêu công khai và quản trị
 - o **Công khai – Public pages:** Cân xứng các trang chức năng phục vụ cho mục đích cung cấp thông tin mang tính quảng bá, một hệ thống các trang công khai thường bao gồm các trang như: Home page, Articles, Article content, Products, Product details, Contact, About us, ... tùy thuộc vào mục đích và định hướng của việc cung cấp thông tin của ứng dụng web cần xây dựng mà sinh viên cần triển khai cho phù hợp.
 - o **Quản trị - Private pages, Administrative pages:** Để phục vụ cho việc cung cấp thông tin, hệ thống cần có các chức năng phục vụ cho việc quản trị nội dung (*Content Management System*). Những trang chức năng này chủ yếu phục vụ cho các mục đích như: Đăng bài, đưa tin, kiểm duyệt thông tin thu nhận được từ quá trình hoạt động của các chức năng công khai (*Kiểm duyệt bình luận, Xác nhận và kích hoạt đơn hàng, ...*), tạo tài khoản, xử lý đơn hàng, ... Thông thường các trang chức năng phục vụ cho mục tiêu quản trị nội dung sẽ phát sinh lệ thuộc theo thiết kế của các trang công khai (*Tùy thuộc theo mục đích hoạt động của ứng dụng*)
- Dựa vào các phân tích và thiết kế từ các công việc đã liệt kê ở trên, sinh viên cần tạo một ứng dụng web bằng ASP.NET sử dụng MVC Framework cùng với các trang chứng năng cần thiết, dựa trên một mẫu thiết kế đã chọn đối với public pages và đối với administrative pages.
- Yêu cầu của bài tập là xây dựng hoàn chỉnh hệ thống các trang công khai cùng với các trang phục vụ cho quản trị để mô tả toàn bộ các tương tác đặc trưng trên hệ thống ứng dụng, chuẩn bị cho các bài tập trong phần 2. Do đó, chỉ cần tạo các controller và view tương ứng với mỗi chức năng cùng với các liên kết cần thiết là đạt yêu cầu.

I/- Phân tích giao diện, xác định các thành phần dùng chung giữa các trang



II/- Kiểm tra phần tử & xác định các thành phần khác biệt

1/- Trang chủ: Index.html

The screenshot shows the Edumark website's homepage. The main content area is highlighted with a red dashed box. To the right, the browser's developer tools are open, showing the DOM structure and CSS styles. A large, stylized starburst graphic with the text "Different areas" is overlaid on the bottom right of the page.

2/- Danh sách sản phẩm: courses.html

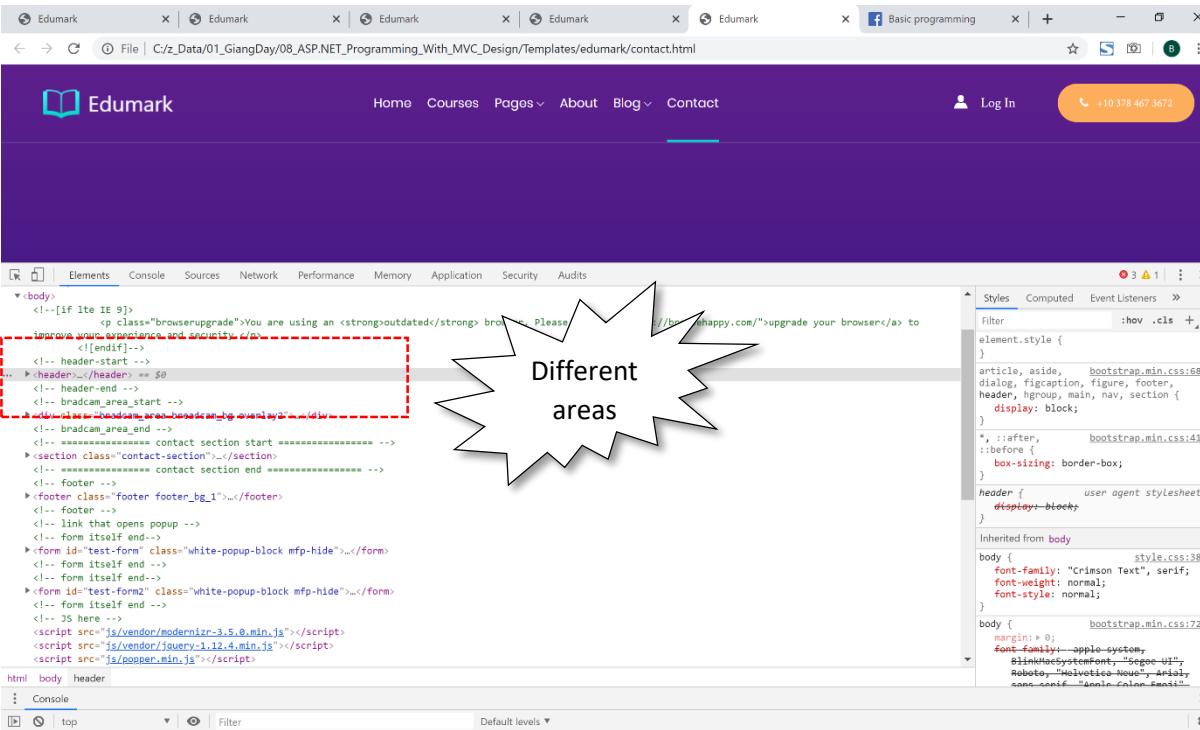
The screenshot shows the Edumark website's "Popular Courses" page. The main content area is highlighted with a red dashed box. The browser's developer tools are open, showing the DOM structure and CSS styles. A large, stylized starburst graphic with the text "Different areas" is overlaid on the bottom right of the page.

3/- Danh sách bài viết: blog.html

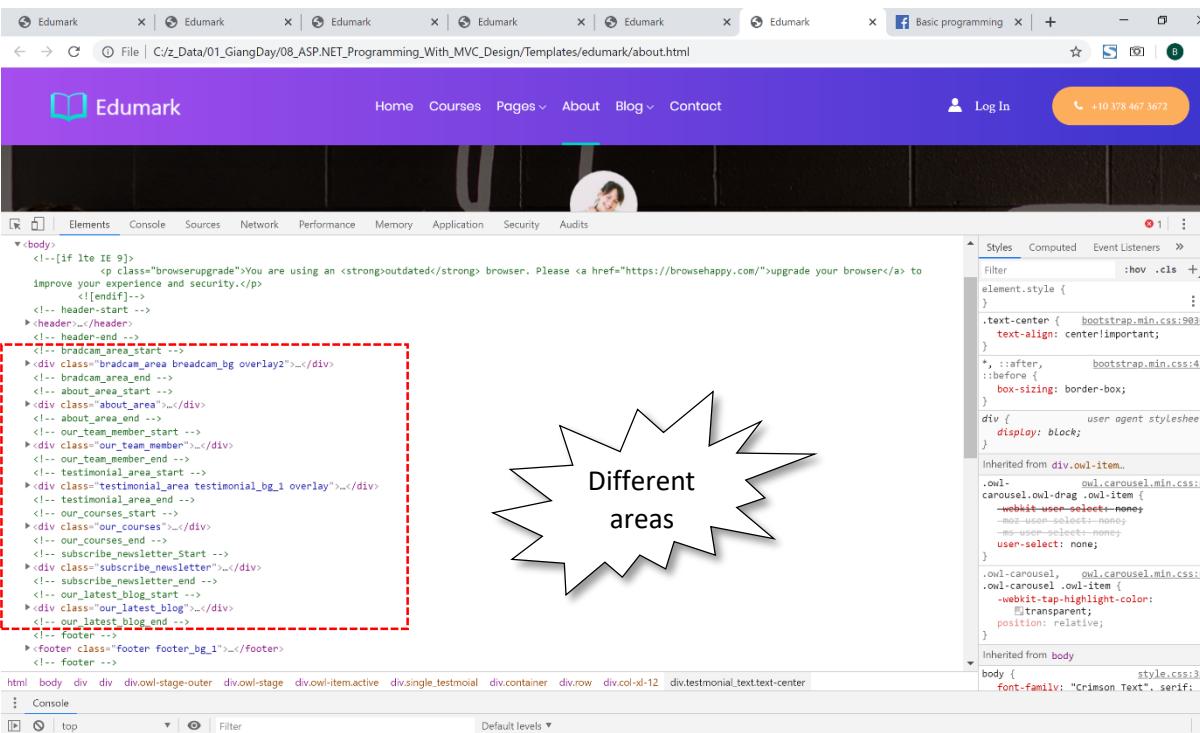
The screenshot shows the browser developer tools with the 'Elements' tab selected. A red box highlights a specific section of the DOM code, specifically a conditional comment for Internet Explorer 9. To the right, the browser's right-hand panel displays the CSS styles applied to the page, with a large starburst graphic labeled 'Different areas' pointing towards both the highlighted code and the styles panel.

4/- Nội dung bài viết: single-blog.html

The screenshot shows the browser developer tools with the 'Elements' tab selected. A red box highlights a specific section of the DOM code, specifically a conditional comment for Internet Explorer 9. To the right, the browser's right-hand panel displays the CSS styles applied to the page, with a large starburst graphic labeled 'Different areas' pointing towards both the highlighted code and the styles panel.

5/- Thông tin liên hệ: contact.html


The screenshot shows a browser window with multiple tabs open, all displaying the same page. The main content is the 'contact.html' page from the 'Edumark' template. A red dashed box highlights the 'header-start' and 'header-end' sections of the HTML code. A large, hand-drawn style starburst graphic with the text 'Different areas' points to this highlighted code block. The browser's developer tools are visible on the right side, showing the styles and computed values for various elements.

6/- Giới thiệu về đơn vị chủ quản: about.html


The screenshot shows a browser window with multiple tabs open, all displaying the same page. The main content is the 'about.html' page from the 'Edumark' template. A red dashed box highlights the 'header-start' and 'header-end' sections of the HTML code. A large, hand-drawn style starburst graphic with the text 'Different areas' points to this highlighted code block. The browser's developer tools are visible on the right side, showing the styles and computed values for various elements.

III/- Chuyển đổi Template cho các trang công khai (Public pages)

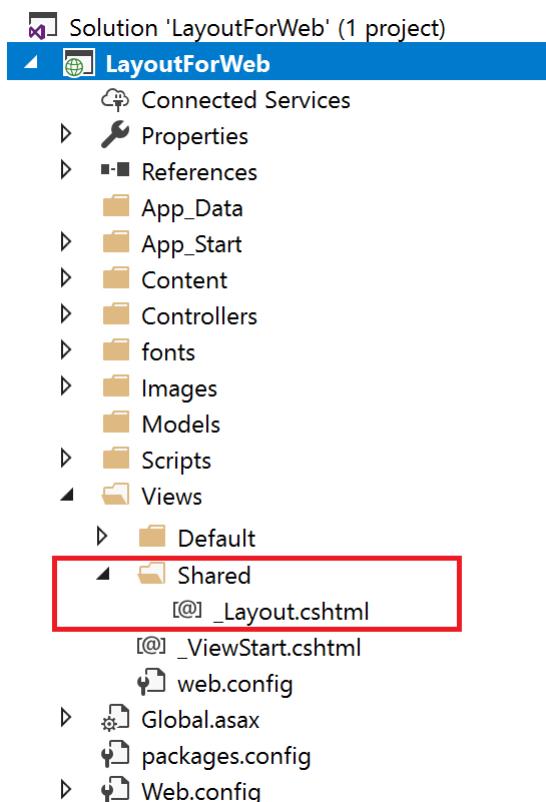
1/- Tạo trang chủ :: Home page

Trang chủ là trang quan trọng nhất trong số các trang thuộc loại public page, trang chức năng này cung cấp hầu hết các loại thông tin đặc trưng, nhằm tiếp cận khách truy cập thông qua internet theo hướng đa dạng, tiện lợi và thân thiện. Để tạo trang chủ cho dự án của mình, chúng ta thực hiện thông qua các bước sau

- ✓ Tạo một Controller với tên mặc định là Default
- ✓ Tạo một View tương ứng với controller ở trên để phục vụ cho việc hiển thị

Lưu ý: Vì thông tin cần thể hiện trên trang chức năng này chủ yếu được tổng hợp từ các loại dữ liệu khác nhau, do đó có thể không nhất thiết phải tạo model riêng cho nó.

Bây giờ, quan sát trong cấu trúc folder **View**, ngoài thư mục Default chứa view của controller vừa tạo ở trên, chúng ta sẽ thấy xuất hiện thư mục Shared với tập tin **_Layout.cshtml** dùng để chứa các mã html dùng chung cho các trang công khai và nhiệm vụ trước tiên là chúng ta là phải chuyển những mã dùng chung đã được phân tích trước đó vào trong tập tin này. Hãy quan sát hình minh họa dưới đây



2/- Cập nhật mã dùng chung cho _Layout.cshtml

2.1/- Cập nhật thông tin meta tag, link tag, style, ...

Tiến hành chuyển các mã khai báo từ trong tập tin **Index.html** (có trong template cần sử dụng cho các trang công khai của dự án) vào tập tin **_Layout.cshtml**. Hãy quan sát hình minh họa sau

```

<!DOCTYPE html>
<html class="no-js lang="zxx">
  <head>
    <meta charset="utf-8">
    <meta http-equiv="x-ua-compatible" content="ie=edge">
    <title>Edumark</title>
    <meta name="description" content="">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <!-- <link rel="manifest" href="site.webmanifest"> -->
    <link rel="shortcut icon" type="image/x-icon" href="img/favicon.png">
    <!-- Place favicon.ico in the root directory -->
    <!-- CSS here -->
    <link rel="stylesheet" href="css/bootstrap.min.css">
    <link rel="stylesheet" href="css/owl.carousel.min.css">
    <link rel="stylesheet" href="css/magnific-popup.css">
    <link rel="stylesheet" href="css/font-awesome.min.css">
    <link rel="stylesheet" href="css/themify-icons.css">
    <link rel="stylesheet" href="css/nice-select.css">
    <link rel="stylesheet" href="css/flaticon.css">
    <link rel="stylesheet" href="css/giigo.css">
    <link rel="stylesheet" href="css/animate.css">
    <link rel="stylesheet" href="css/slicknav.css">
    <link rel="stylesheet" href="css/style.css">
    <!-- <link rel="stylesheet" href="css/responsive.css"> -->
  </head>
  <body>
    <!--[if lte IE 9]>
      <p class="browserupgrade">You are using an <strong>outdated</strong> browser. Please <a href="https://browsehappy.com/">
        <![endif]-->
  </body>

```

Hình 2.1.1: Các mã khai báo có trong tag head của tập tin index.html thuộc template

```

_Layout.cshtml*  index.html  DefaultController.cs  NuGet: LayoutForWeb
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta name="description" content="">
    <title>@ViewBag.Title - My ASP.NET Application</title>
    <meta http-equiv="x-ua-compatible" content="ie=edge">
    <!-- <link rel="manifest" href="site.webmanifest"> -->
    <link rel="shortcut icon" type="image/x-icon" href="img/favicon.png">
    <!-- Place favicon.ico in the root directory -->
    <!-- CSS here -->
    <link rel="stylesheet" href="css/bootstrap.min.css">
    <link rel="stylesheet" href="css/owl.carousel.min.css">
    <link rel="stylesheet" href="css/magnific-popup.css">
    <link rel="stylesheet" href="css/font-awesome.min.css">
    <link rel="stylesheet" href="css/themify-icons.css">
    <link rel="stylesheet" href="css/nice-select.css">
    <link rel="stylesheet" href="css/flaticon.css">
    <link rel="stylesheet" href="css/giigo.css">
    <link rel="stylesheet" href="css/animate.css">
    <link rel="stylesheet" href="css/slicknav.css">
    <link rel="stylesheet" href="css/style.css">
    <!-- <link rel="stylesheet" href="css/responsive.css"> -->
  </head>
  <body>
    <!--[if lte IE 9]>
      <p class="browserupgrade">You are using an <strong>outdated</strong> browser. Please <a href="https://browsehappy.com/">
        <![endif]-->
  </body>

```

Hình 2.1.2: Các mã khai báo có trong tag head được chuyển vào _layout.cshtml

2.2/- Cập nhật các mã html dùng chung cho các trang công khai

Mã html thuộc khu vực đầu trang như Menu, navigation bar, ...

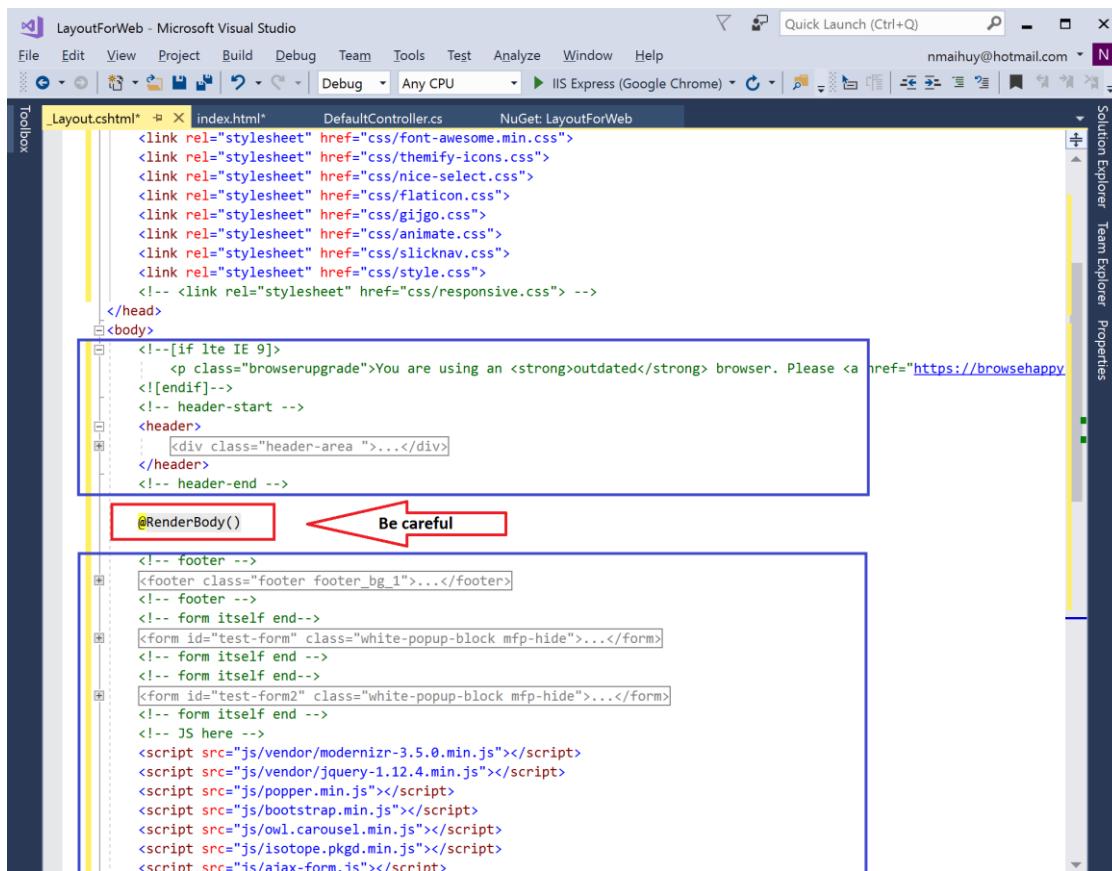
```

_Layout.cshtml*  index.html*  DefaultController.cs  NuGet: LayoutForWeb
<link rel="stylesheet" href="css/slicknav.css">
<link rel="stylesheet" href="css/style.css">
<!-- <link rel="stylesheet" href="css/responsive.css"> -->
</head>

<body>
  <!--[if lte IE 9]>
    <p class="browserupgrade">You are using an <strong>outdated</strong> browser. Please <a href="https://browsehappy.com/">
      <![endif]-->
  <!-- header-start -->
  <header>
    <div class="header-area ">...</div>
  </header>
  <!-- header-end -->

```

Mã html dành cho khu vực cuối trang,



```

<head>
    <link rel="stylesheet" href="css/font-awesome.min.css">
    <link rel="stylesheet" href="css/themify-icons.css">
    <link rel="stylesheet" href="css/nice-select.css">
    <link rel="stylesheet" href="css/flaticon.css">
    <link rel="stylesheet" href="css/gijgo.css">
    <link rel="stylesheet" href="css/animate.css">
    <link rel="stylesheet" href="css/slicknav.css">
    <link rel="stylesheet" href="css/style.css">
    <!-- link rel="stylesheet" href="css/responsive.css" -->
</head>
<body>
    <!--[if lte IE 9]>
        <p class="browserupgrade">You are using an <strong>outdated</strong> browser. Please <a href="https://browsehappy.com">upgrade today!</a>
    <![endif]-->
    <!-- header-start -->
    <header>
        <div class="header-area ">...</div>
    </header>
    <!-- header-end -->
    <!-- RenderBody() --> Be careful
    <!-- footer -->
    <footer class="footer footer_bg_1">...</footer>
    <!-- footer -->
    <!-- form itself end-->
    <form id="test-form" class="white-popup-block mfp-hide">...</form>
    <!-- form itself end -->
    <!-- form itself end-->
    <form id="test-form2" class="white-popup-block mfp-hide">...</form>
    <!-- form itself end -->
    <!-- JS here -->
    <script src="js/vendor/modernizr-3.5.0.min.js"></script>
    <script src="js/vendor/jquery-1.12.4.min.js"></script>
    <script src="js/popper.min.js"></script>
    <script src="js/bootstrap.min.js"></script>
    <script src="js/owl.carousel.min.js"></script>
    <script src="js/isotope.pkgd.min.js"></script>
    <script src="js/ajax-form.js"></script>

```

Lưu ý: Hãy cẩn thận với html helper **@RenderBody()**, vì đây chính là phần liên kết để đưa thông tin khác biệt từ các trang khác nhau của ứng dụng để thực hiện quá trình rendering, tạo ra trang web hoàn chỉnh trước khi gửi xuống máy truy cập

2.3/- Sao chép CSS, JavaScript, hình ảnh và cập nhật đường dẫn

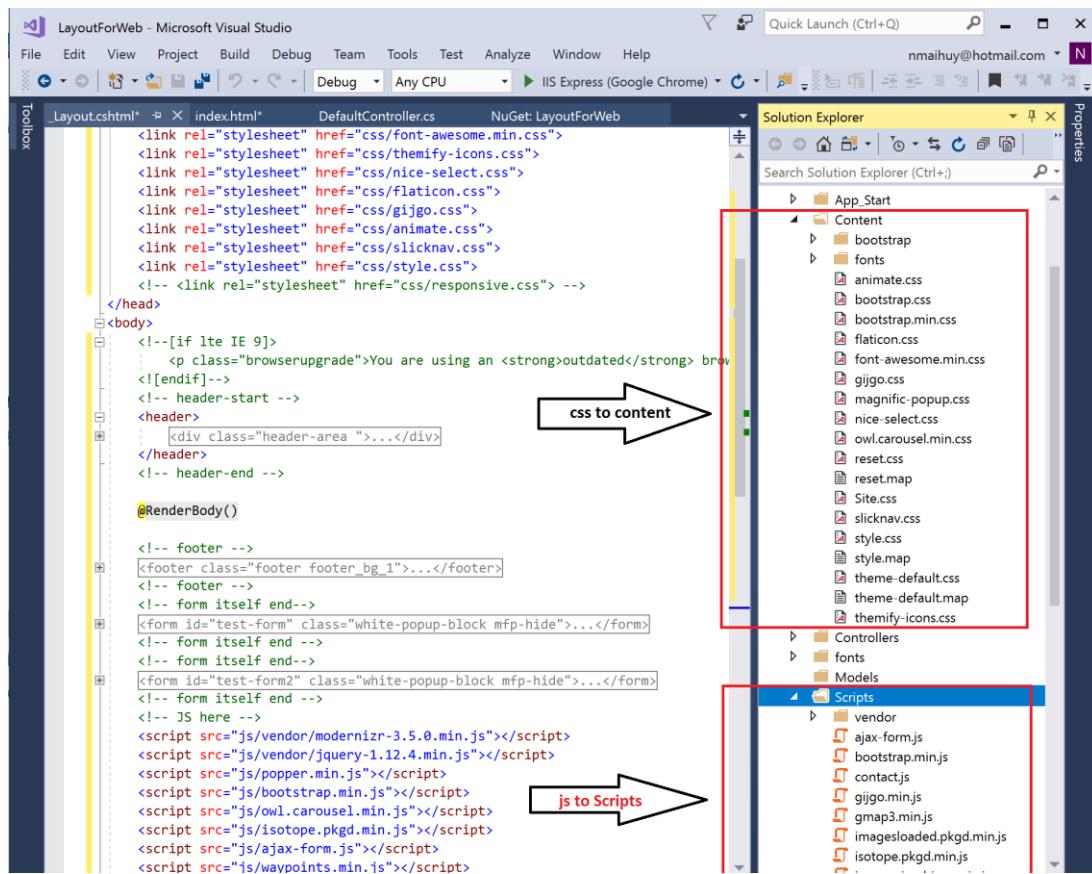
Khi xây dựng giao diện, thông thường bộ phận front-end xây dựng các folder phục vụ cho lưu trữ các tập tin hỗ trợ định dạng (CSS), kiểu chữ (font), mã kịch bản (JavaScript), Hình ảnh minh họa (Images) trong một số các thư mục đặc biệt của template. Trong minh họa này, các thư mục trên lần lượt có tên là: css, fonts, js, img. Bạn cần chuyển những thư viện này vào trong project của mình.

Name	Date modified	Type	Size
207 Edu mark DOC	10/1/2019 5:34 AM	File folder	
css	10/1/2019 5:34 AM	File folder	
fonts	10/1/2019 5:34 AM	File folder	
img	10/1/2019 5:34 AM	File folder	
js	10/1/2019 5:35 AM	File folder	
scss	10/1/2019 5:35 AM	File folder	
about.html	10/5/2019 7:41 PM	Chrome HTML Docu...	28 KB
blog.html	10/5/2019 7:41 PM	Chrome HTML Docu...	32 KB
contact.html	10/5/2019 7:41 PM	Chrome HTML Docu...	49 KB
contact_process.php	6/17/2019 9:34 AM	PHP File	2 KB
course_details.html	10/5/2019 7:41 PM	Chrome HTML Docu...	34 KB
courses.html	10/5/2019 7:41 PM	Chrome HTML Docu...	126 KB
elements.html	10/5/2019 7:41 PM	Chrome HTML Docu...	41 KB
index.html	10/5/2019 7:41 PM	Chrome HTML Docu...	129 KB
readme.txt	9/24/2019 11:23 PM	Text Document	1 KB
single-blog.html	10/5/2019 7:41 PM	Chrome HTML Docu...	35 KB

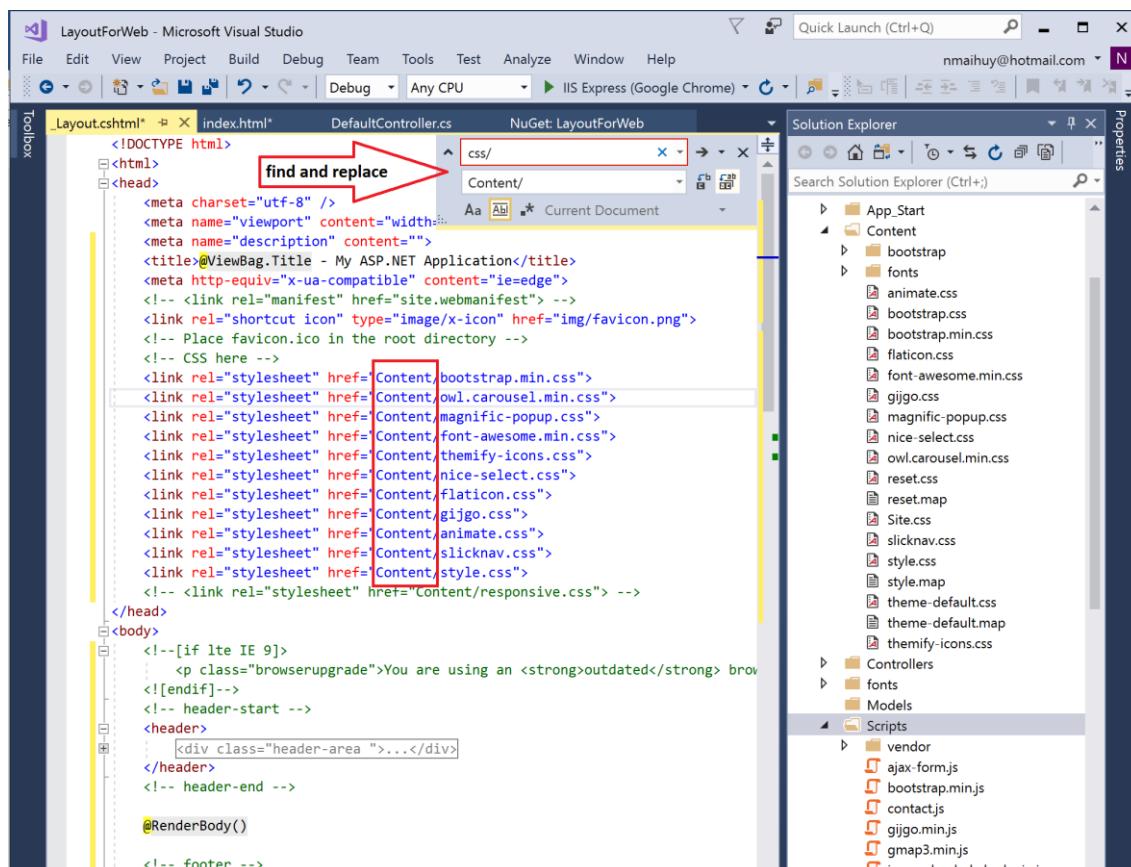
Hình 2.3.1: Các folder chứa CSS, font, JavaScript, images có trong template

Việc chuyển đổi dữ liệu từ template từ template vào project được thực hiện như sau:

Css → content folder, fonts → Scripts

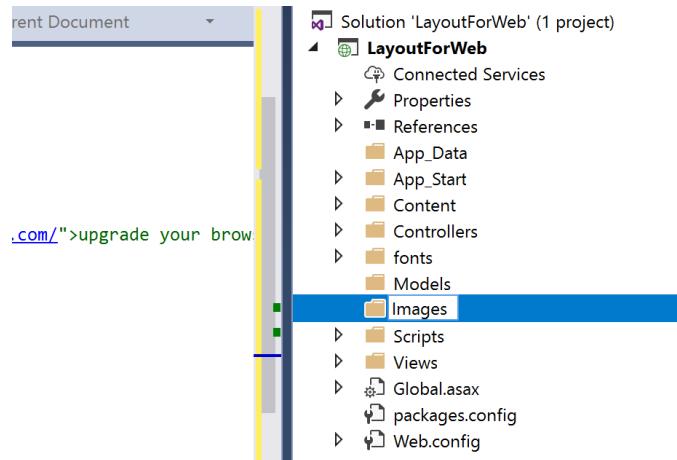


Sau khi chuyển các thư viện từ template vào trong project, chúng ta sẽ gặp phải một vấn đề, đó chính là đường dẫn. Bạn cần phải cập nhật đường dẫn lại cho phù hợp

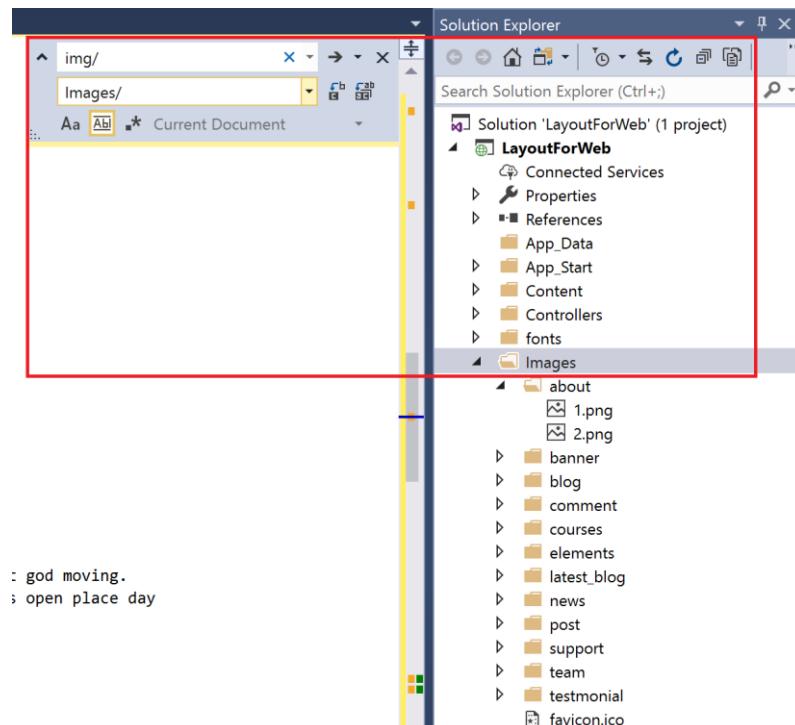


Trong hình minh họa trên, chúng ta cần phải cập nhật tất cả các khai báo cho css bằng content (*Bởi vì trước đó, chúng ta đã chép các tập tin phục vụ cho định dạng từ folder css trong template vào thư mục content của project*). Hãy tiến hành sửa đường dẫn cho khác khai báo kinh bản bằng cách tìm và thay thế tên của thư mục js bằng Scripts đối với tất cả các script tag đã dùng để khai báo jQuery hoặc JavaScript tại vị trí cuối cùng của trang.

Đối với hình ảnh minh họa sử dụng trên template ban đầu cũng cần phải cập nhật đường dẫn theo cách tương tự. Tiến hành tạo thư mục Images trong project để lưu hình ảnh.



Sau đó sao chép những hình ảnh cần thiết từ thư mục img có trong template vào thư mục Images vừa tạo ở trên trong project của bạn, rồi tiến hành cập nhật lại đường dẫn cho các img tag cho phù hợp với đường dẫn mới trong project.

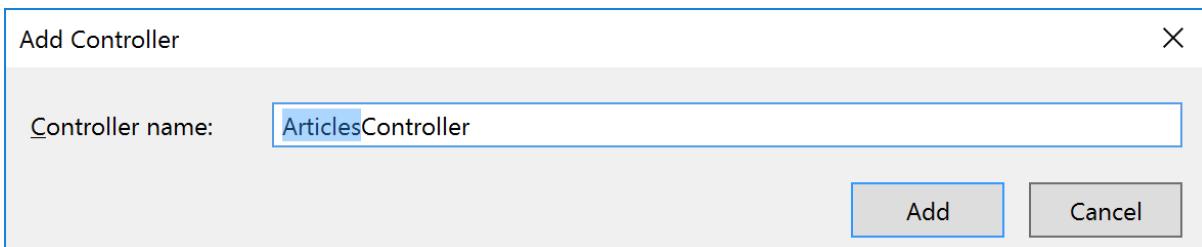


3/- Cập nhật mã trên View của trang chủ

Dựa vào phần mã html của tập tin index.html có trong template, tiến hành cập nhật những nội dung đặc trưng của trang này (*Những vùng không tin khác biệt đã được xác định trong quá trình phân tích được mô tả trong phần trước*) cho view của trang chủ (thuộc nhóm public pages)

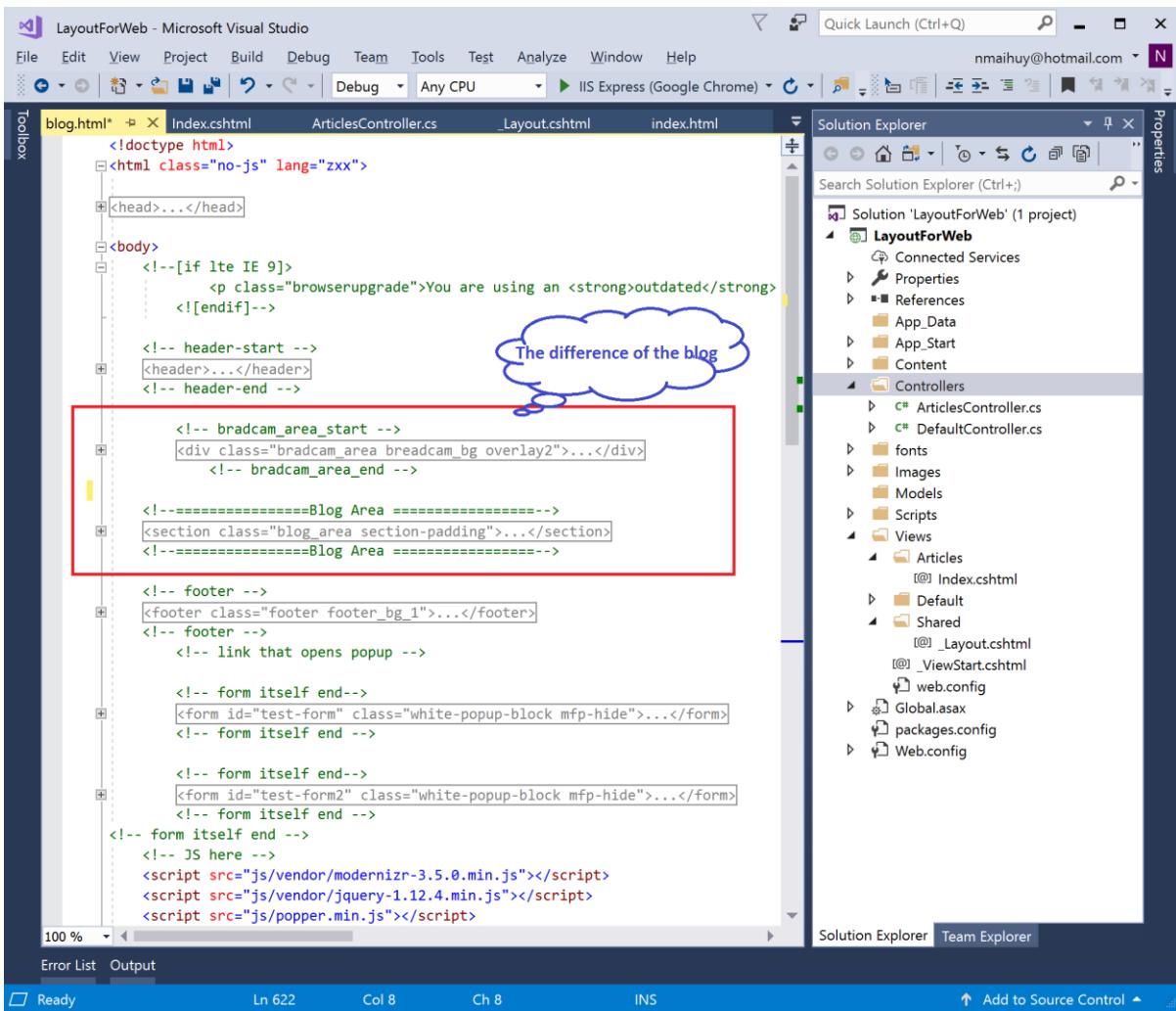
4/- Tạo danh sách bài viết

4.1/- Controller của chức năng hiển thị danh sách bài viết



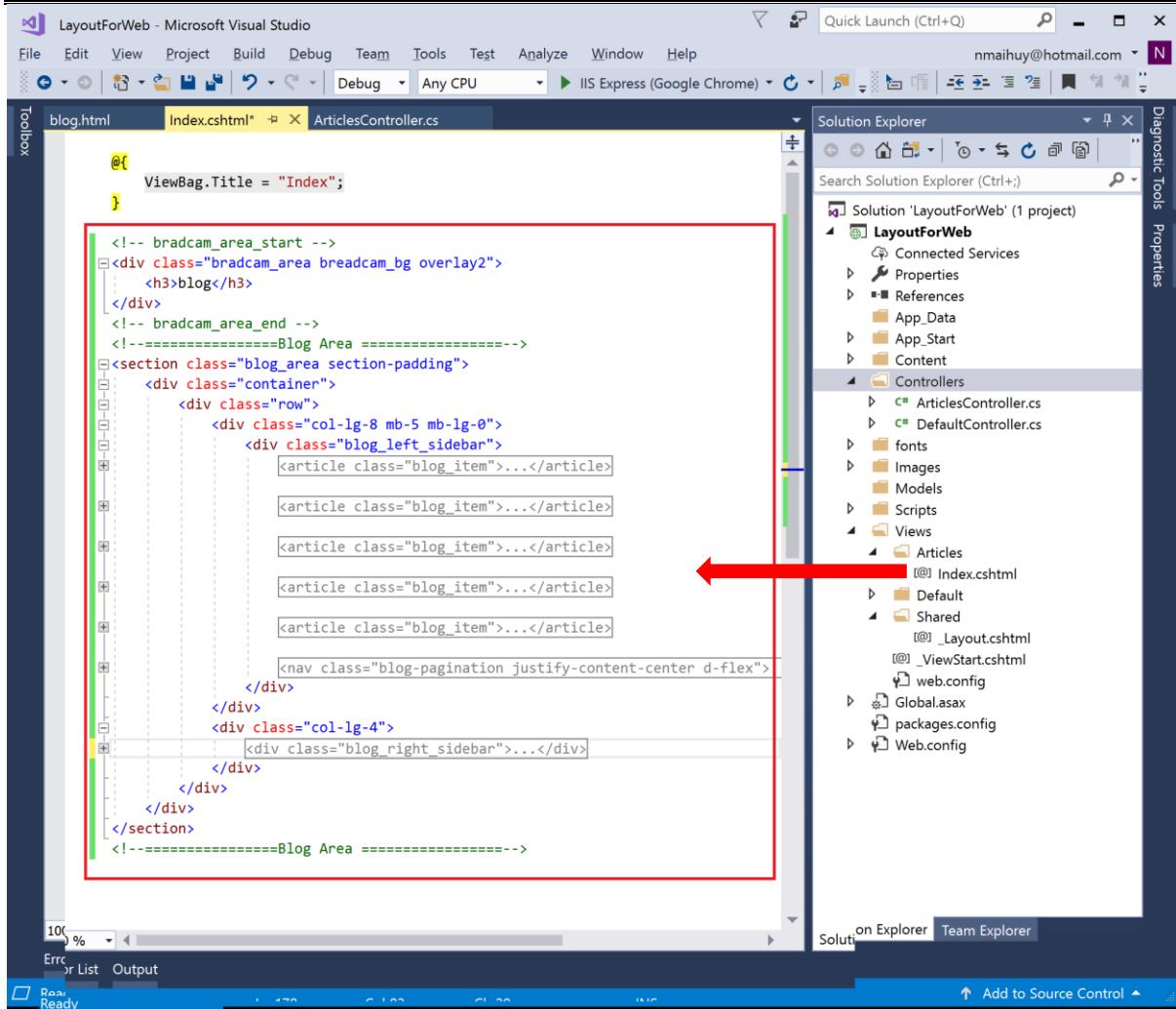
4.2/- View hiển thị dữ liệu danh sách tin, bài viết

Tạo view có tên Artiles, tương ứng với controller đã tạo trong mục 4.1 . Sau đó chép phần mã khác biệt của trang này từ trong tập tin blog.html vào trong view vừa tạo. Hãy quan sát phần nội dung khác biệt của blog.html trong hình minh họa dưới đây



Hình 4.2.1: Minh họa phần mã khác biệt trong blog.html của template

Phần mã html sau khi được đưa vào view, sẽ thể hiện trong hình minh họa 4.2.2 ở trang bên



Hình 4.2.2 Minh họa cho nội dung của Articles.cshtml (View thể hiện danh sách bài viết)

5/- Xây dựng các public pages còn lại

Tương tự như các hướng dẫn ở trên, hãy tiến hành tạo các trang công khai khác của project theo thiết kế của bạn (VD: *Nội dung bài viết, Danh sách sản phẩm, nội dung sản phẩm, Liên hệ, ...*)

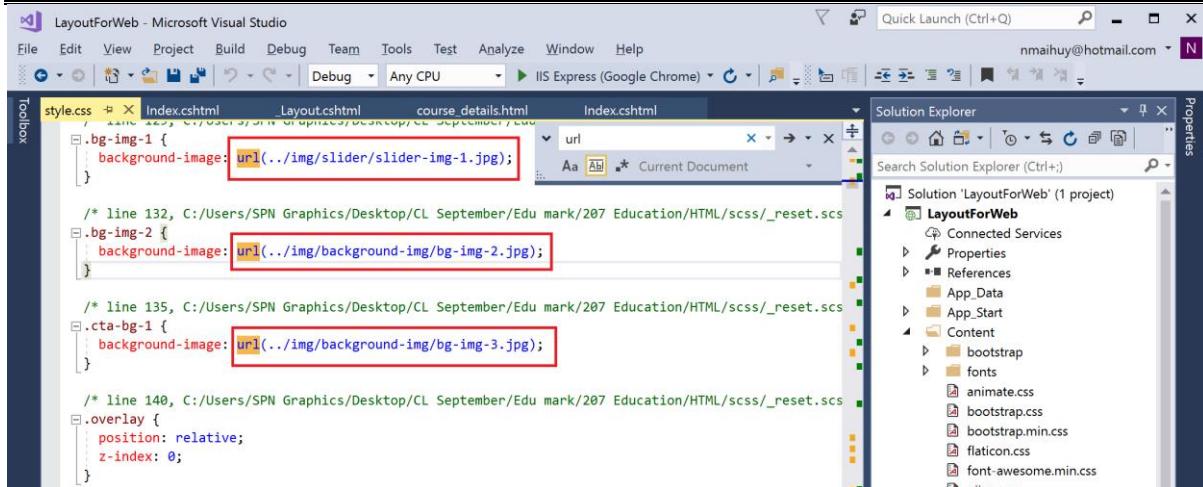
6/- Một số vấn đề về tài nguyên tham chiếu

Sau khi chuyển mã html từ mẫu thiết kế đã được layout bởi front-end, các lập trình viên phát triển ứng dụng web thường gặp một số lỗi liên quan đến tài nguyên tham chiếu phục vụ cho:

- Hiển thị theo định dạng đã xây dựng, không thể hiện được đúng định dạng của mã CSS đã xây dựng trước đó
- Không chạy được các kịch bản đã được lập trình bởi JavaScript như mẫu thiết kế ban đầu
- Không hiển thị được hình ảnh, font chữ viết, ...

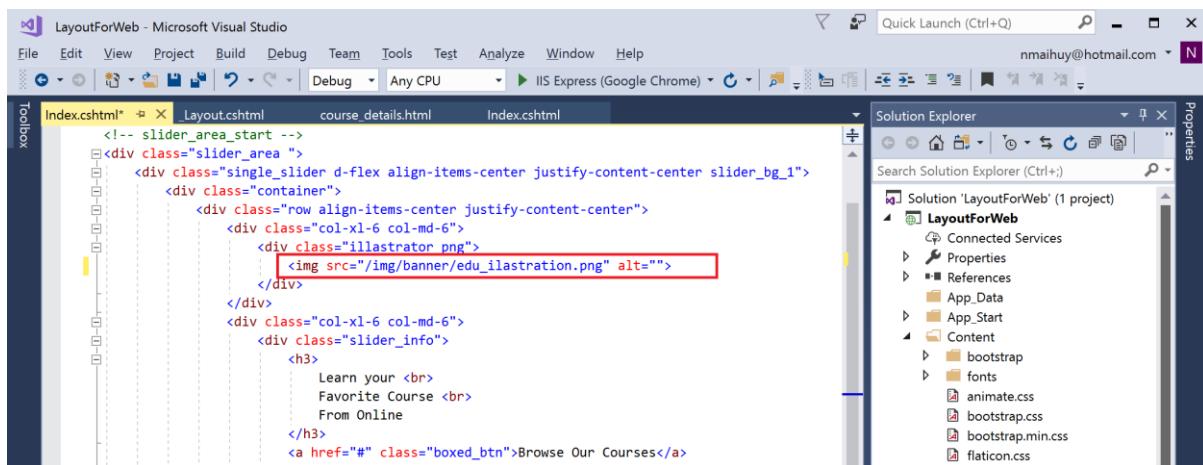
Những lỗi này xảy ra, thường do nguyên nhân tham chiếu tài nguyên không chính xác. Hãy quan sát hình minh họa 6.1, 6.2

- Lỗi khi tham chiếu dữ liệu trong tập tin css



Hình 6.1 Lỗi đường dẫn tham chiếu bên trong tập tin chứa mã định dạng CSS

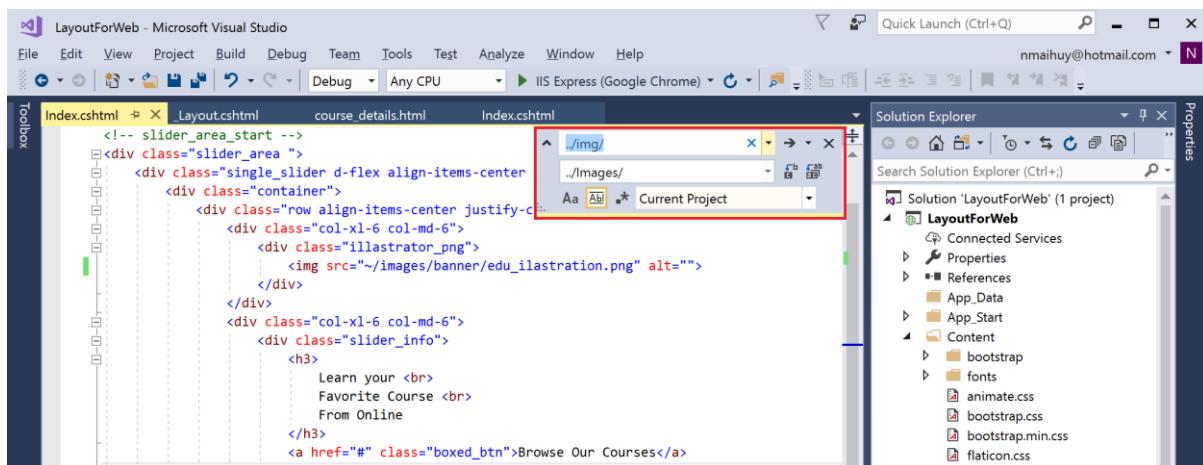
- Đường dẫn tham chiếu đến hình minh họa trong tập tin index.html



Hình 6.2: Lỗi tham chiếu hình ảnh do không phù hợp với vị trí lưu trữ trong project

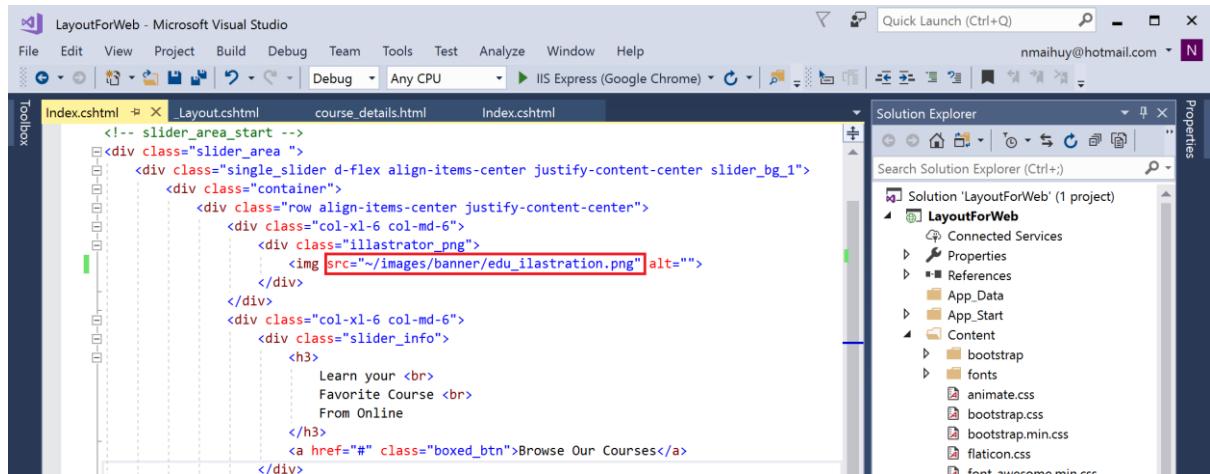
Một số thao tác nên dùng:

- Sử dụng chức năng tìm kiếm và thay thế (trong menu Edit, hoặc nhấn Ctrl+h) để tìm kiếm tất cả các đường dẫn bị sai trong các mã tham chiếu thuộc HTML hoặc CSS và thay lại cho đúng. Ví dụ trong hình minh họa 6.3, chúng ta tìm “./img/” thay bằng “./Images/”, do sau khi chuyển từ template sang project hiện hành, tên của thư mục chứa hình ảnh đã thay bằng Images



Hình 6.3 Minh họa việc sử dụng chức năng tìm và thay thế để sửa lại đường dẫn

- Ở trên, chúng ta sửa lại đường dẫn theo cách dùng đường dẫn tương đối (*Relative path*). Có thể sử dụng đường dẫn tuyệt đối trong lập trình web dựa vào ký hiệu “~”. Đây là ký hiệu đại diện cho domain hiện hành mà ứng dụng đang hoạt động. Như vậy, để truy xuất tới thư mục Images có trong thư mục gốc của domain hiện hành ta sẽ dùng đường dẫn truy xuất tài nguyên hình ảnh như sau “~/Images/tên_file_hình”



Hình 6.4 Minh họa việc sử dụng đường dẫn tuyệt đối để trỏ tới hình cần sử dụng

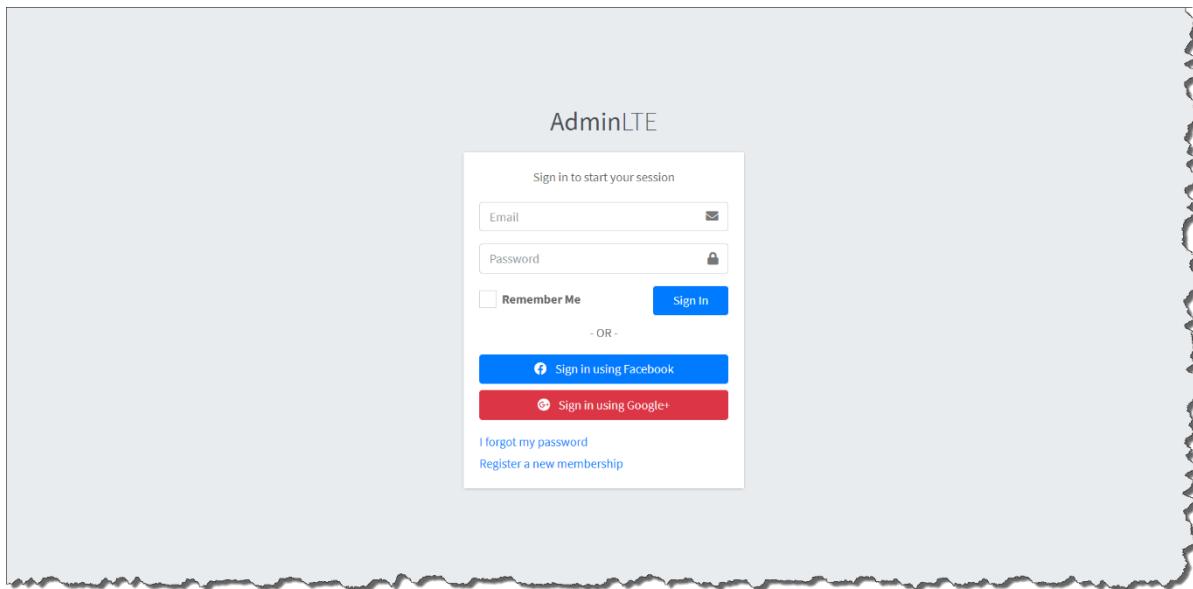
Lưu ý:

Thông thường, khi thiết kế và layout giao diện web, đội ngũ front-end thường có thói quen sử dụng đường dẫn tương đối (*Relative path*) để xác định tài nguyên tham chiếu. Tuy nhiên, trong lập trình web thì lập trình viên lại thường sử dụng đường dẫn tuyệt đối (*Absolute path*) thay vì đường dẫn tương đối mà các bạn đã quen thuộc khi layout giao diện

IV/- Thực hiện xây dựng các trang phục vụ cho quản trị

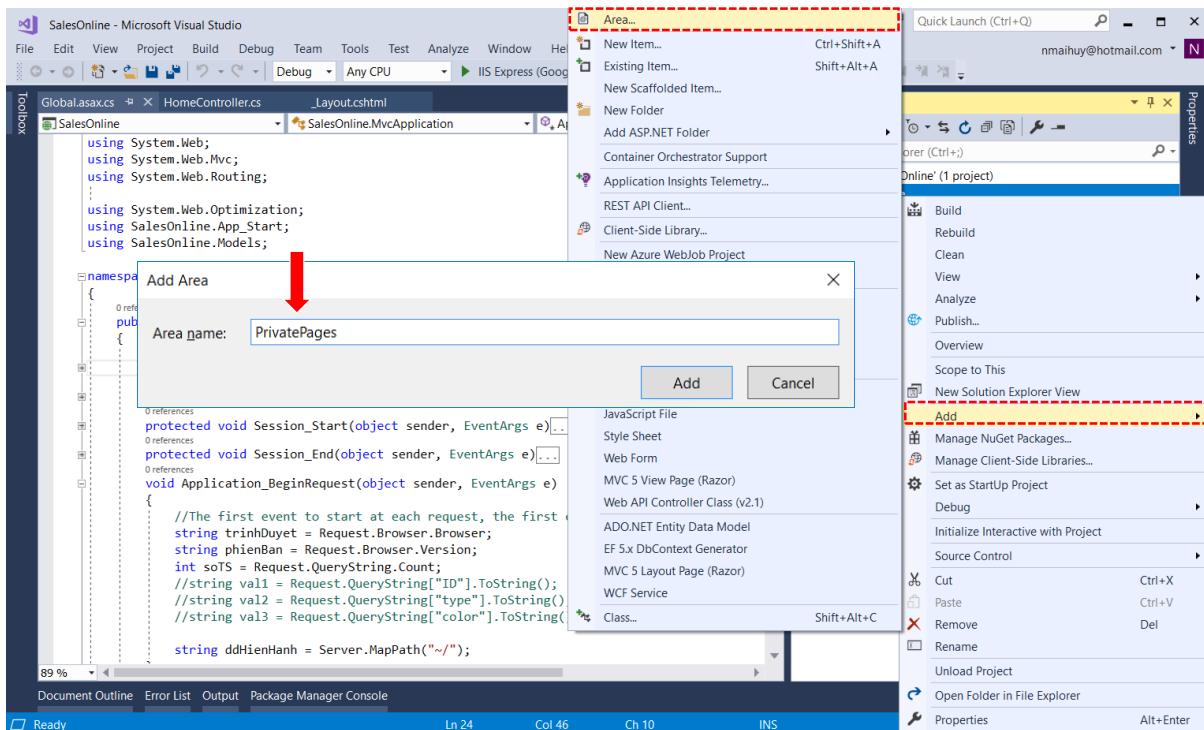
1/- Login page

Trang chức năng này nên đặt ngoài gốc, không dùng _Layout.cshtml chung với các trang quản trị



2/- Các trang dùng cho quản trị - Private pages, Administration pages

Các trang dùng cho quản trị nên được tạo và quản lý trong một area độc lập và tách biệt so với hệ thống các trang công khai. Để tạo area, nhấn nút phải vào tên của project trong khung Solution explorer -> Add -> Area rồi đặt tên cho “Area name”, trong hình minh họa dưới đây, tôi đặt tên của area này là PrivatePages



Sau khi tạo PrivatePages area thành công, ta sẽ thấy cấu trúc của thành phần này cũng bao gồm các thành phần như Models-Views-Controllers nhưng hoàn toàn độc lập với các thành phần tương tự ở bên ngoài project

```

using System.Web.Mvc;

namespace SalesOnline.Areas.PrivatePages
{
    public class PrivatePagesAreaRegistration : AreaRegistration
    {
        public override string AreaName
        {
            get
            {
                return "PrivatePages";
            }
        }

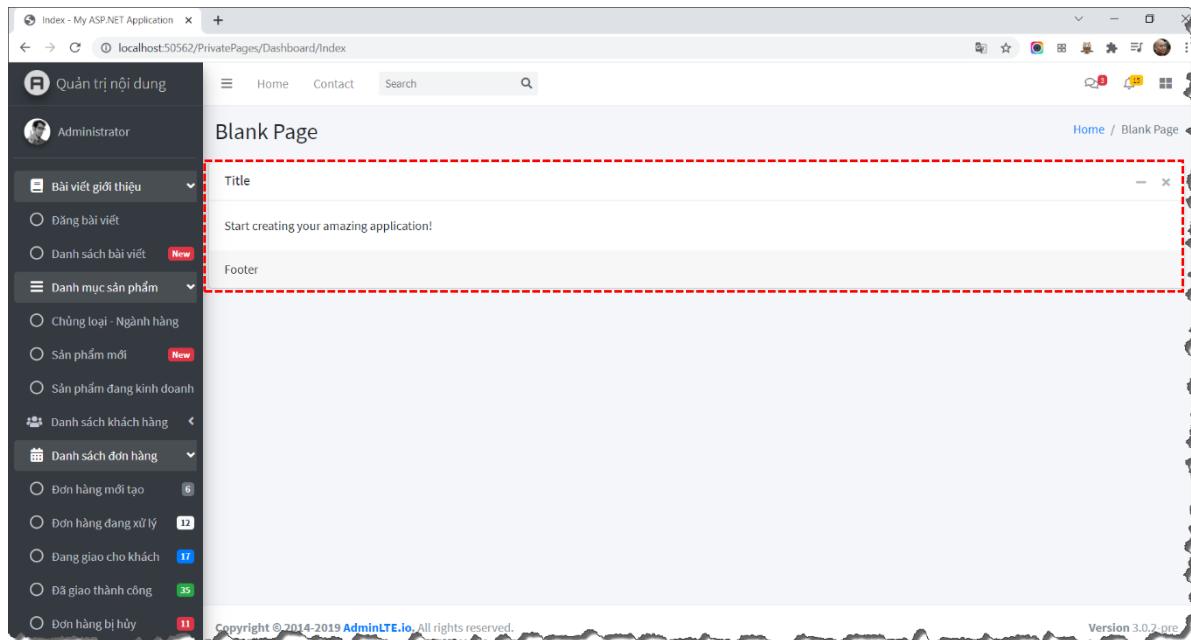
        public override void RegisterArea(AreaRegistrationContext context)
        {
            context.MapRoute(
                "PrivatePages_default",
                "PrivatePages/{controller}/{action}/{id}",
                new { action = "Index", id = UrlParameter.Optional }
            );
        }
    }
}

```

PrivatePages area phục vụ cho quản trị

Trong minh họa của bài tập này, chúng ta dùng template AdminLTE 3 cho giao diện của các trang quản trị.

Bước tiếp theo, cần tạo `_Layout.cshtml` dựa vào “**blank page**” có trong template, do trang này chỉ chứa các thành phần dùng chung cho toàn bộ giao diện của các trang chức năng khác. Trong đó, Left side bar và Top navigation đã được tạo sẵn



Cấu trúc của Left side bar (*giải sứ*) được thiết kế với 4 mục tiêu cùng 2 cấp thẻ hiện cho việc liên kết với các chức năng như mô tả sau

- ❖ Bài viết giới thiệu
 - ✓ Đăng bài viết
 - ✓ Danh sách bài viết

- ❖ Danh mục sản phẩm

- ✓ Chủng loại – Ngành hàng
- ✓ Sản phẩm mới
- ✓ Sản phẩm đang kinh doanh

❖ Danh sách khách hàng

- ✓ Khách hàng mới
- ✓ Danh sách khách hàng

❖ Danh sách đơn hàng

- ✓ Đơn hàng mới tạo
- ✓ Đơn hàng đang xử lý
- ✓ Đang giao cho khách
- ✓ Đã giao thành công
- ✓ Đơn hàng bị hủy

Sinh viên tham khảo cách xây dựng giao diện của các trang mẫu trong danh sách: IU elements, Forms, Tables, ... để xây dựng giao diện của các trang quản trị như các mẫu mô tả sau đây

2.1/- Đăng bài – Đưa tin

The screenshot shows a web-based administrative interface for creating a product introduction article. The left sidebar contains navigation links: Quản trị nội dung, Administrator, Bài viết giới thiệu, Danh mục sản phẩm, Danh sách khách hàng, and Danh sách đơn hàng. The main content area has two tabs: Thông tin chính and Hình đại diện. The Thông tin chính tab contains fields for Tên bài viết (Title), Nội dung tóm tắt (Summary), Ngày đăng (Publish Date), Tài khoản (Account), Kiểm duyệt bài viết (Review status) with an unchecked checkbox for Đã duyệt (Approved), Số lần đọc (Number of reads) set to 125, and a rich text editor for Nội dung bài viết (Article content). The Hình đại diện tab displays a preview image of a woman holding shopping bags and includes a File input field with Choose file, Browse, and Upload buttons.

2.2/- Danh mục bài viết

Mã bài viết	Tên bài viết	Ngày đăng	Tài khoản	Đã duyệt	Số lần đọc
01	Xu hướng thời trang hè	09:30 23/06/2021	HoangTuan	true	135
02	Sức khỏe người tiêu dùng	09:30 23/06/2021	HoangTuan	true	976
03	Giới trẻ thích công nghệ gì	09:30 23/06/2021	HoangTuan	true	10256
04	Những đột phá của công nghệ 2021	09:30 23/06/2021	HoangTuan	true	20094
05	Bữa ăn gia đình mùa Covid	09:30 23/06/2021	HoangTuan	true	561
06	Phó tổng thống Mỹ thăm việt nam	09:30 23/06/2021	HoangTuan	true	209
07	Cấp độ dịch ở các quận huyện TP.HCM	09:30 23/06/2021	HoangTuan	true	325
08	Giá vàng trên thế giới tăng cao	09:30 23/06/2021	HoangTuan	true	451
09	Bài thuốc dân gian chữa viêm họng	09:30 23/06/2021	HoangTuan	true	671
10	Hai năm kinh tế trong vòng xoáy Covid	09:30 23/06/2021	HoangTuan	true	710

Showing 1 to 10 of 20 entries

Footer: Copyright © 2014-2019 AdminLTE.io. All rights reserved. Version 3.0.2-pre

2.3/- Danh mục sản phẩm

Sản phẩm 1

Nội dung sản phẩm thứ nhất
dùng để minh họa cho thông tin
tóm lược về sản phẩm, đây là
thông tin lấy từ thuộc tính tóm
tắt nội dung

[Delete](#) [Modify](#) [Inactive](#)

Sản phẩm 2

Nội dung sản phẩm thứ nhất
dùng để minh họa cho thông tin
tóm lược về sản phẩm, đây là
thông tin lấy từ thuộc tính tóm
tắt nội dung

[Delete](#) [Modify](#) [Inactive](#)

Sản phẩm 3

Nội dung sản phẩm thứ nhất
dùng để minh họa cho thông tin
tóm lược về sản phẩm, đây là
thông tin lấy từ thuộc tính tóm
tắt nội dung

[Delete](#) [Modify](#) [Inactive](#)

Sản phẩm 4

Nội dung sản phẩm thứ nhất
dùng để minh họa cho thông tin
tóm lược về sản phẩm, đây là
thông tin lấy từ thuộc tính tóm
tắt nội dung

[Delete](#) [Modify](#) [Inactive](#)

Sản phẩm 5

Nội dung sản phẩm thứ nhất
dùng để minh họa cho thông tin
tóm lược về sản phẩm, đây là
thông tin lấy từ thuộc tính tóm
tắt nội dung

[Delete](#) [Modify](#) [Inactive](#)

Sản phẩm 6

Nội dung sản phẩm thứ nhất
dùng để minh họa cho thông tin
tóm lược về sản phẩm, đây là
thông tin lấy từ thuộc tính tóm
tắt nội dung

[Delete](#) [Modify](#) [Inactive](#)

Sản phẩm 7

Nội dung sản phẩm thứ nhất
dùng để minh họa cho thông tin
tóm lược về sản phẩm, đây là
thông tin lấy từ thuộc tính tóm
tắt nội dung

[Delete](#) [Modify](#) [Inactive](#)

Sản phẩm 8

Nội dung sản phẩm thứ nhất
dùng để minh họa cho thông tin
tóm lược về sản phẩm, đây là
thông tin lấy từ thuộc tính tóm
tắt nội dung

[Delete](#) [Modify](#) [Inactive](#)

Sản phẩm 9

Nội dung sản phẩm thứ nhất
dùng để minh họa cho thông tin
tóm lược về sản phẩm, đây là
thông tin lấy từ thuộc tính tóm
tắt nội dung

[Delete](#) [Modify](#) [Inactive](#)

Footer: Copyright © 2014-2019 AdminLTE.io. All rights reserved. Version 3.0.2-pre

2.4/- Thông tin sản phẩm

Thông tin sản phẩm mới

Hình minh họa cho sản phẩm

Tên sản phẩm

Nội dung tóm tắt

Giá bán

Đơn vị tính Select

Thông tin khuyến mãi

Ngày đăng

Tài khoản

Kiểm duyệt sản phẩm

Số lần xem

File input

Choose file Browse Upload

Giới thiệu chi tiết về sản phẩm

Lưu thông tin sản phẩm

Copyright © 2014-2019 AdminLTE.io. All rights reserved.

Version 3.0.2-pre

2.5/- Thông tin khách hàng

Thông tin của khách hàng

Stt	Số điện thoại	Tên khách hàng	Giới tính	Địa chỉ	Số lần mua	Email
1	0913.12.31.12	Trần Thanh Bình	Nam	112 Lê Thánh Tôn, Q.10, TP.HCM	23	thanhbinh@gmail.com
2	0908.24.25.26	Nguyễn Trần Ý Lan	Nữ	1071 Cách Mạng Tháng Tám, Q.3, TP.HCM	42	ctylan12@gmail.com
3	0918.234567	Ngô Minh Tùng	Nam	501 Điện Biên Phủ, Q.3, TP.HCM	9	minhtung1280@gmail.com
4	0933.12.12.12	Phạm Tuấn Minh	Nam	421/32 Xô Viết Nghệ Tĩnh, Q.Bình Thạnh, TP.HCM	11	tuanminh@gmail.com
5	0916.27.27.27	Nguyễn Thanh Minh Nguyệt	Nữ	36 Hậu Giang, Q.6, TP.HCM	58	nguyetminh@gmail.com
6	0903.22.33.44	Võ Quang Cường	Nam	171 Kinh Dương Vương, Q.6, TP.HCM	81	quangcuong@gmail.com
7	0913.641576	Nguyễn Minh Quang	Nam	1024/4 Cách Mạng Tháng Tám, Q.3, TP.HCM	31	minhquang1206@gmail.com
8	0913.68.68.68	Hà Minh Thành	Nam	11 Nguyễn Trãi, Q.5, TP.HCM	7	minhthanh@gmail.com
9	0908.11.11.11	Ngô Thanh Mỹ Yến	Nữ	971 Hoàng Văn Thụ, Q.Tân Bình, TP.HCM	12	myyen1208@gmail.com
10	0903.00.00.45	Đỗ Thị Mỹ Linh	Nữ	127 Trần Hưng Đạo, Q.5, TP.HCM	78	mylinh@yahoo.com

Showing 1 to 10 of 15 entries

Previous 1 2 Next

Footer

Copyright © 2014-2019 AdminLTE.io. All rights reserved.

Version 3.0.2-pre

2.6/- Quản lý đơn hàng

The screenshot shows a left sidebar with navigation items like 'Quản trị nội dung', 'Administrator', 'Bài viết giới thiệu', 'Danh mục sản phẩm', 'Danh sách khách hàng', and 'Danh sách đơn hàng'. The main content area has a header 'Danh sách đơn hàng mới' and a breadcrumb 'Home / Danh sách đơn hàng'. Below is a table listing 10 orders:

Số đơn hàng	Tên khách hàng	Ngày giao	Địa chỉ	Thành tiền	Nhóm lệnh
1	Trần Quốc Tú	12/08/2021	112 Lê Thánh Tôn, Q.10, TP.HCM	541000	Hủy đơn Chi tiết Kích hoạt
2	Nguyễn Minh Quang	17/09/2021	472/4 Cách Mạng Tháng Tám, Q.3, TP.HCM	759000	Hủy đơn Chi tiết Kích hoạt
3	Phan Minh Thành	17/09/2021	32 Xô Viết Nghệ Tĩnh, Q.Bình Thạnh, TP.HCM	272000	Hủy đơn Chi tiết Kích hoạt
4	Hà Tuấn Tú	17/09/2021	12 Ngõ Tất Tố, Q.Bình Thạnh, TP.HCM	1125000	Hủy đơn Chi tiết Kích hoạt
5	Trần Hoàng Vũ	19/09/2021	112 Lê Thánh Tôn, Q.10, TP.HCM	541000	Hủy đơn Chi tiết Kích hoạt
6	Trần Quốc Tú	19/09/2021	212 Điện Biên Phủ, Q.3, TP.HCM	778000	Hủy đơn Chi tiết Kích hoạt
7	Nguyễn Thanh Loan Vy	22/09/2021	11 Nguyễn Thiện Thuật, Q.3, TP.HCM	1450000	Hủy đơn Chi tiết Kích hoạt
8	Nguyễn Hoàng Tùng	23/09/2021	112 Lê Thánh Tôn, Q.10, TP.HCM	1201000	Hủy đơn Chi tiết Kích hoạt
9	Ngô Thanh Hương	23/09/2021	35 Nguyễn Văn Đậu, Q.Bình Thạnh, TP.HCM	375000	Hủy đơn Chi tiết Kích hoạt
10	Nguyễn Quang Hưng	23/09/2021	112 Lê Thánh Tôn, Q.10, TP.HCM	141000	Hủy đơn Chi tiết Kích hoạt

Showing 1 to 10 of 12 entries

Footer: Copyright © 2014-2019 AdminLTE.io. All rights reserved.

Version 3.0.2-pre

* Lưu ý:

- Để thực hiện giao diện của các trang chức năng phục vụ cho mục đích quản trị nội dung (*Như các hình minh họa ở trên*), sinh viên cần tham khảo các trang mô tả cho việc sử dụng các thành phần dành cho : Form, Table, Chart, ... trong template Admin LTE để từ đó sử dụng Html helper với css class cho phù hợp.
- Một số mẫu minh họa trong template Admin LTE:

The screenshot shows a left sidebar with 'AdminLTE 3' and 'Alexander Pierce' logged in. Under 'Pages', 'Invoice' is selected. The main content area shows an invoice for 'Admin, Inc.' to 'John Doe' dated 2/10/2014. The invoice details and payment methods are listed.

Invoice

Note:
This page has been enhanced for printing. Click the print button at the bottom of the invoice to test.

AdminLTE, Inc. Date: 2/10/2014
From: Admin, Inc.
795 Folsom Ave, Suite 600
San Francisco, CA 94107
Phone: (804) 123-5432
Email: info@almasedeedstudio.com

To: John Doe
795 Folsom Ave, Suite 600
San Francisco, CA 94107
Phone: (555) 539-1037
Email: john.doe@example.com

Invoice #007612

Order ID: 4F3S8J
Payment Due: 2/22/2014
Account: 968-34567

Qty	Product	Serial #	Description	Subtotal
1	Call of Duty	455-981-221	El snort testosterone trophy driving gloves handsome	\$64.50
1	Need for Speed IV	247-925-726	Wes Anderson umami biodiesel	\$50.00
1	Monsters DVD	735-845-642	Terry Richardson helvetica tousled street art master	\$10.70
1	Grown Ups Blue Ray	422-568-642	Tousled lomo letterpress	\$25.99

Payment Methods: Amount Due 2/22/2014

The screenshot shows a product page for 'LOWA Men's Renegade GTX Mid Hiking Boots'. On the left is a sidebar with navigation links like Dashboard, Widgets, Layout Options, Charts, UI Elements, Forms, Tables, Examples, Calendar, Gallery, Mailbox, Pages (selected), Invoice, and Profile. The main content area features a large image of the hiking boot, its name, a review section with placeholder text, color and size selection dropdowns, and a price of \$80.00.

The screenshot shows a 'General Form' example page. The sidebar includes 'Forms' (selected) under Examples. The main content area displays various form input examples: 'Quick Example' (Email address, Password, File input, Checkbox), 'General Elements' (Text, Textarea), 'Input with success' (green border with checkmark), 'Input with warning' (orange border with exclamation mark), and 'Input with error' (red border with X). A note at the bottom right indicates 'checkbox checked' and 'radio checked'.

The screenshot shows a 'DataTables' example page. The sidebar includes 'Tables' (selected) under Examples. The main content area displays a table titled 'DataTable with minimal features & hover style' showing browser compatibility across different platforms and engines.

Rendering engine	Browser	Platform(s)	Engine version	CSS grade
Gecko	Firefox 1.0	Win 98+ / OSX.2+	1.7	A
Gecko	Firefox 1.5	Win 98+ / OSX.2+	1.8	A
Gecko	Firefox 2.0	Win 98+ / OSX.2+	1.8	A
Gecko	Firefox 3.0	Win 2k+ / OSX.3+	1.9	A
Gecko	Camino 1.0	OSX.2+	1.8	A
Gecko	Camino 1.5	OSX.3+	1.8	A
Gecko	Netscape 7.2	Win 95+ / Mac OS 8.6-9.2	1.7	A
Gecko	Netscape Browser 8	Win 98SE+	1.7	A
Gecko	Netscape Navigator 9	Win 98+ / OSX.2+	1.8	A
Gecko	Mozilla 1.0	Win 95+ / OSX.1+	1	A