

BỘ GIÁO DỤC VÀ ĐÀO TẠO
ĐẠI HỌC CÔNG NGHỆ TP.HCM

T HỰC HÀNH LẬP TRÌNH WEB

Biên Soạn:

Nguyễn Đình Ánh

Nguyễn Huy Cường

Dương Thành Phết

THỰC HÀNH LẬP TRÌNH WEB



* 1 . 2 0 2 1 . C M P 3 7 6 *

Các ý kiến đóng góp về tài liệu học tập này, xin gửi về e-mail của ban biên tập:
tailieuhoctap@hutech.edu.vn

MỤC LỤC

MỤC LỤC	I
HƯỚNG DẪN	II
BÀI 1: LÀM QUEN VỚI ỨNG DỤNG WEB ASP.NET MVC	1
1.1 CƠ BẢN VỀ MVC 5 VÀ CÁC YÊU CẦU CĂN THIẾT ... T WEB APPLICATION MVC 5 ..	1
1.1.1 Yêu cầu.....	1
1.1.2 Các bước cài đặt MVC5.....	1
TÓM TẮT	7
CÂU HỎI ÔN TẬP	7
BÀI 2: VIẾT TRANG WEB QUẢN LÝ SÁCH VỚI ASP.NET MVC	8
2.1 TẠO MODEL BOOK	8
2.2 RÀNG BUỘC DỮ LIỆU NHẬP	17
TÓM TẮT	19
CÂU HỎI ÔN TẬP	19
BÀI 3: THỰC HÀNH THEO DỰ ÁN WEBSITE BIG-SCHOOL	20
3.1 CÔNG NGHỆ SỬ DỤNG	20
3.2 MÔ TẢ YÊU CẦU.....	21
3.3 PHẦN MỀM	21
3.4 PHÂN TÍCH YÊU CẦU CỦA ỨNG DỤNG BIGSCHOOL	21
3.4.1 Xác định các usecase trong dự án.....	21
3.4.2 Phân tích use case.....	22
3.4.3 Thứ tự use-case sẽ thực hiện	23
3.5 XÂY DỰNG ỨNG DỤNG BẰNG ASP.NET MVC	23
3.5.1 Tạo cơ sở dữ liệu theo mô hình EF Code First.....	23
3.5.2 Use-case: Thêm mới khóa học	28
3.5.3 Ghi đè CSS mặc định của Bootstrap	44
3.5.4 Thêm mới thuộc tính trong Asp.Net Identity Users	46
3.5.5 Hiển thị danh sách các khóa hoặc sắp diễn ra tại trang chủ	47
3.5.6 Bổ sung thuộc tính Name vào màn hình đăng ký, ... (Form Sign up)	49
3.5.7 Chỉnh sửa giao diện trang chủ: danh sách khóa học sắp diễn ra	51
3.5.8 Use-case: Đăng ký tham dự khóa học	53
3.5.9 Use-case theo dõi Giảng viên	60
3.5.10 Xem danh sách khóa học đăng ký tham dự	65
3.5.11 Use-case Quản lý Khóa học (Xóa, cập nhật)	69
3.5.12 Use-case thông báo (notification)	80
3.5.13 Tách file javascript trong ứng dụng	85
TÓM TẮT	88
CÂU HỎI ÔN TẬP	89
TÀI LIỆU THAM KHẢO	90

HƯỚNG DẪN

MÔ TẢ MÔN HỌC

Học phần được xây dựng với các nội dung: kiến thức tổng quan về lập trình ứng dụng web; cách thức phát triển ứng dụng web dùng ASP.NET MVC cơ bản và nâng cao.

NỘI DUNG MÔN HỌC

- Bài 1: Làm quen với ứng dụng Web ASP.NET MVC
- Bài 2: Viết trang Web quản lý sách với ASP.NET MVC
- Bài 3: Thực hành theo dự án Website Big-School

KIẾN THỨC TIỀN ĐỀ

Môn học thực hành lập trình web đòi hỏi sinh viên có nền tảng về kỹ thuật lập trình, lập trình hướng đối tượng, tiếp xúc nhiều với internet.

YÊU CẦU MÔN HỌC

Người học cần đi học đầy đủ, đọc các nội dung sẽ được học trước khi đến lớp, làm các bài tập vẽ nhà và đảm bảo thời gian tự học ở nhà.

CÁCH TIẾP NHẬN NỘI DUNG MÔN HỌC

Để học tốt môn này, sinh viên cần ôn tập các bài đã học, trả lời các câu hỏi và làm đầy đủ bài tập; đọc trước bài mới và tìm thêm các thông tin liên quan đến bài học.

Đối với mỗi bài học, sinh viên đọc trước mục tiêu và tóm tắt bài học, sau đó đọc nội dung bài học. Kết thúc mỗi ý của bài học, sinh viên trả lời câu hỏi ôn tập và kết thúc toàn bộ bài học, sinh viên làm các bài tập.

PHƯƠNG PHÁP ĐÁNH GIÁ MÔN HỌC

Môn học được đánh giá như sau:

- Điểm thực hành (100%): Hình thức thi thực hành, phù hợp với quy chế đào tạo và tình hình thực tế tại nơi tổ chức học tập.

BÀI 1: LÀM QUEN VỚI ỨNG DỤNG WEB ASP.NET MVC

Bài này giúp người học nắm được các nội dung sau:

- *Ứng dụng Web ASP.NET MVC 5 là gì?*
- *Cài đặt môi trường và công cụ để phát triển ứng dụng Web ASP.NET MVC 5*
- *Lập trình cơ bản ứng dụng Website với ASP.NET MVC 5*

1.1 CƠ BẢN VỀ MVC 5 VÀ CÁC YÊU CẦU CẦN THIẾT ĐỂ VIẾT WEB APPLICATION MVC 5

1.1.1 Yêu cầu

MVC 5 yêu cầu .Net 4.5 trở lên và chạy trên nền tảng hệ điều hành

- Windows Vista Sp2
- Windows 7
- Windows 8
- Windows 10

1.1.2 Các bước cài đặt MVC5

Mặc định khi cài đặt Visual Studio 2013 hoặc Visual Studio 2015 thì các thành phần của MVC 5 đã được bao gồm. Trong trường hợp bạn đã cài Visual Studio 2012 mà muốn bổ sung MVC5 thì có thể vào địa chỉ website <http://www.microsoft.com/en-us/download/41532> tải gói nâng cấp và cài đặt theo hướng dẫn.

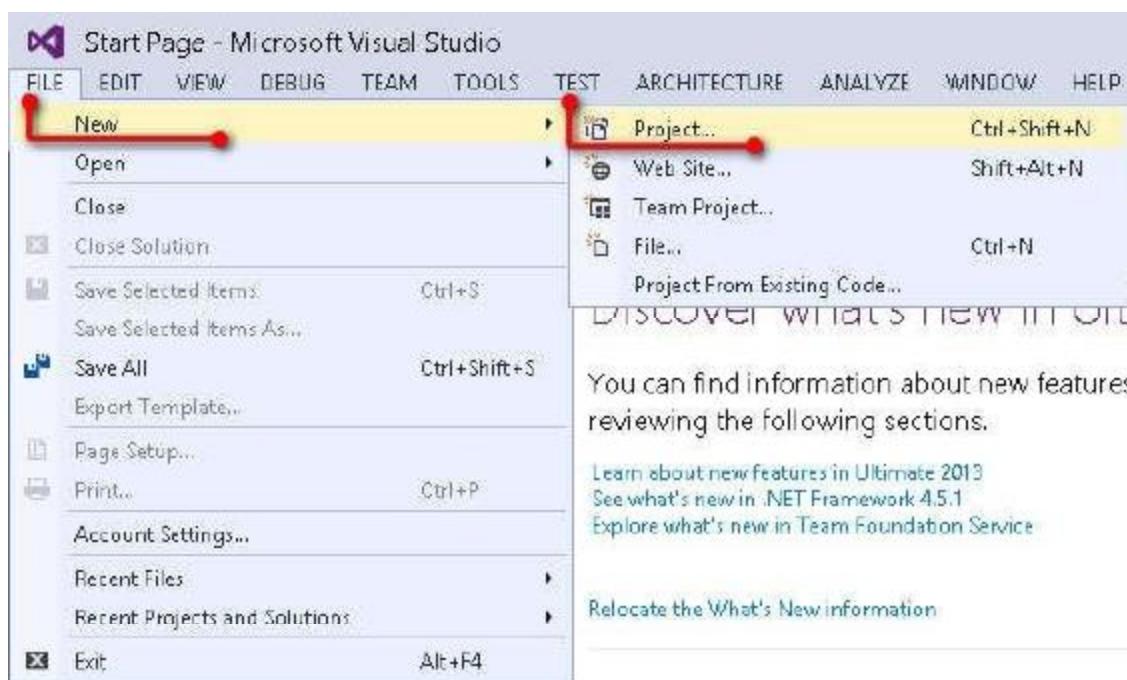
Chú ý: Tất cả các hình ảnh trong các bài LAB đều được thực hiện trên **Visual Studio 2013 & Visual Studio 2015**

- Bước 1: Tạo mới ứng dụng ASP.NET MVC 5

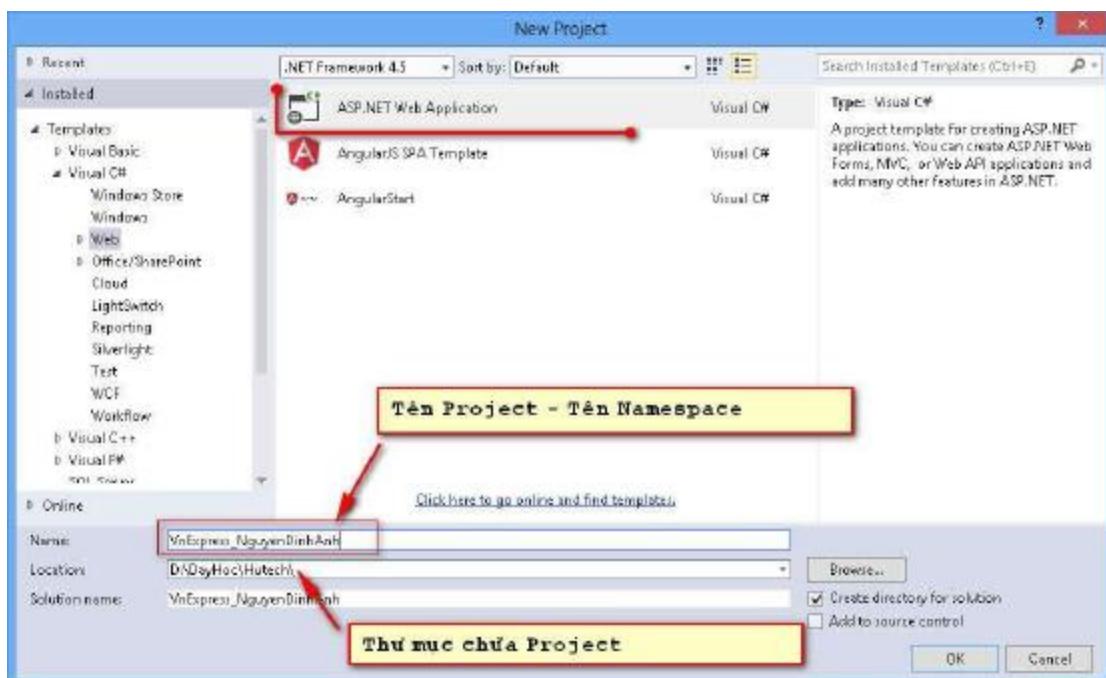
Bạn có thể tạo mới ứng dụng web ASP MVC 5 trên cả hai phiên bản Visual Studio 2013, 2015 hoặc Visual Studio 2013, 2015 Express.

Để tạo mới một ứng dụng MVC

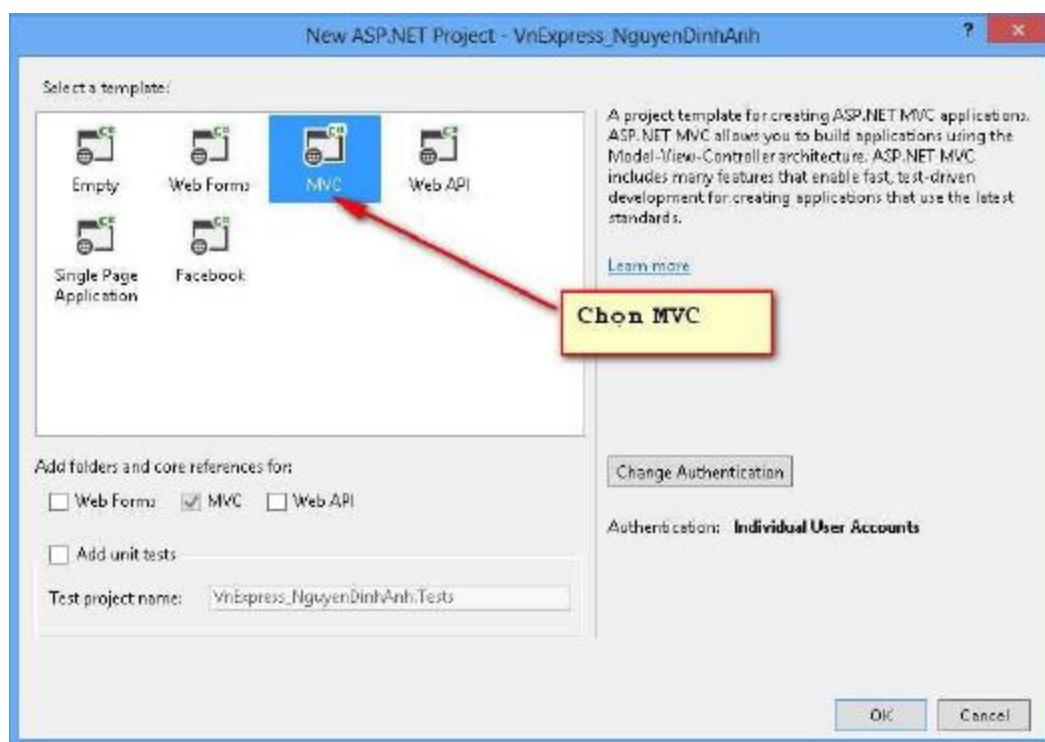
Chọn File → New Project



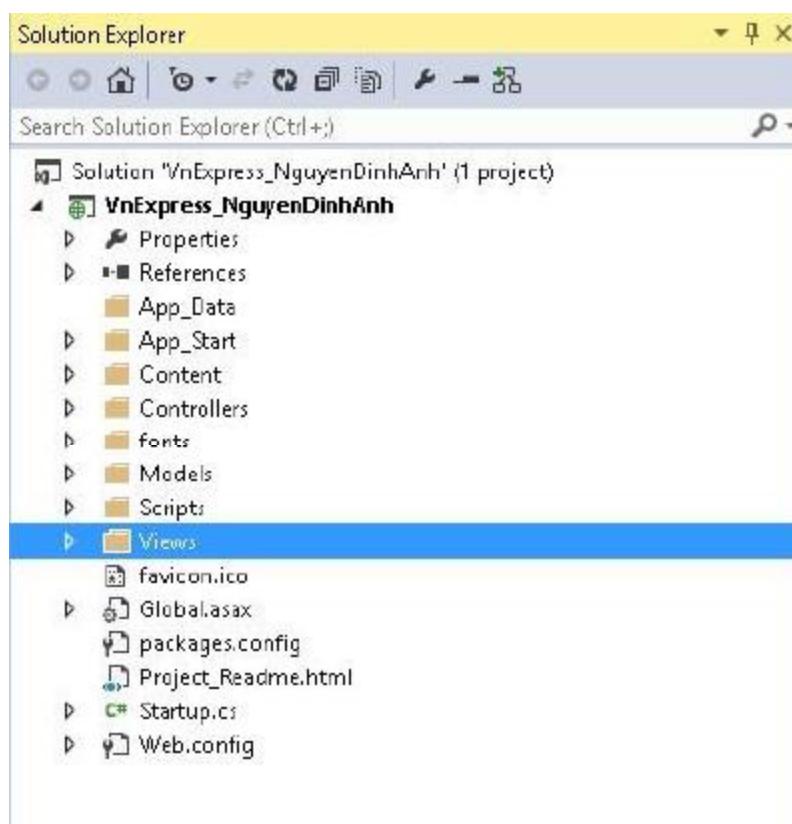
Trong phần các mẫu Teamplate đã được cài đặt, chọn **Visual C# → Web Template**



Sau khi tạo mới **Web Application MVC5**, cửa sổ tiếp theo hiện lên chọn **MVC**



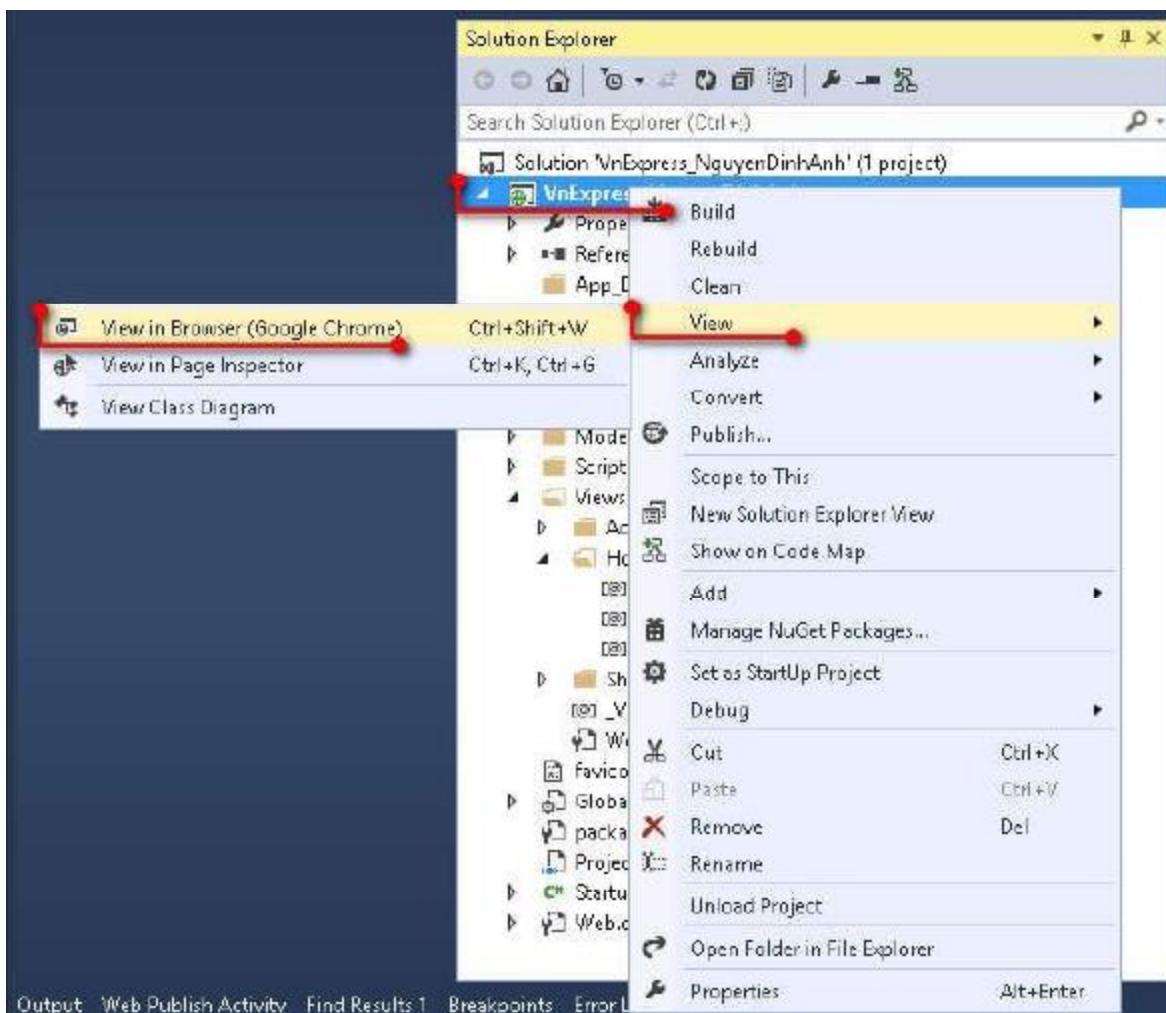
Sau khi chọn xong các bước trên, bạn có được ứng dụng mẫu **MVC Application** với các thành phần mở rộng được thêm vào sẵn, cấu trúc như hình



Cấu trúc của một MVC Web Application

Thư mục	Mô tả
/Controllers	Thư mục chứa các lớp xử lý gọi là Controller
/Models	Thư mục chứa các lớp đối tượng Model
/Views	Thư mục đặt các mẫu giao diện View
/Scripts	Thư mục chứa các bộ thư viện javascript được tích hợp vào sẵn của MVC hoặc là nơi bạn tổ chức và đặt các thư viện Javascript mà mình tự phát triển
/fonts	Chứa các mẫu font của thư viện Bootstrap
/Contents	Đặt CSS , hình ảnh hoặc các file javascript
/App_Data	Chứa database local
/App_Start	Các lớp cấu hình của ứng dụng như Routing, Bundling (hỗ trợ nén CSS, javascript để giảm kích thước)

Nhấn chuột phải lên **project** và chọn **View/ View in Browser** để chạy website



Trang mặc định được chạy là **Views/Home/Index.cshtml**, cấu hình quy định trang đầu tiên được chạy của ứng dụng cấu hình tại lớp **RouteConfig**

```

4  using System.Web;
5  using System.Web.Mvc;
6  using System.Web.Routing;
7
8  namespace VnExpress_NguyendinhAnh
9  {
10     public class RouteConfig
11     {
12         public static void RegisterRoutes(RouteCollection routes)
13         {
14             routes.IgnoreRoute("{resource}.axd/{pathInfo}");
15
16             routes.MapRoute(
17                 name: "Default",
18                 url: "{controller}/{action}/{id}",
19                 defaults: new { controller = "Home", 
20                               action = "Index", id = UrlParameter.Optional }
21             );
22         }
23     }
24 }

```

Home (Controller)
Index (Method Index trong HomeController)

Cấu trúc xử lý để chạy một trang web MVC mặc định là :

`http://localhost:Port/{Controller}/{Action}/{Id}`



TÓM TẮT

Ứng dụng ASP.NET MVC sử dụng mô hình MVC trong xử lý các yêu cầu, không sử dụng Page Life cycle như Web Forms. Trên ASP.NET MVC, Model State lưu giữ trạng thái cố định trong suốt quá trình post back, bạn cũng có thể sử dụng ViewBag, ViewData, Temp data để quản lý thông tin trạng thái. Mã lệnh được phân chia rõ ràng theo mô hình MVC, giúp cho việc nâng cấp, bảo trì được dễ dàng, thuận tiện. ASP.NET MVC không lưu giữ thông tin trạng thái, do đó tập tin kết xuất tối ưu hơn rất nhiều, giúp cho cải thiện tốc độ ứng dụng web trong thực tế một cách đáng kể. Lập trình MVC, bạn phải nắm vững kiến thức về HTML, CSS và Javascript để biết triển khai front-end, xây dựng các xử lý và kết hợp các thành phần với back-end cho hiệu quả.

CÂU HỎI ÔN TẬP

Câu 1: Mô hình MVC là gì? Các thành phần của MVC?

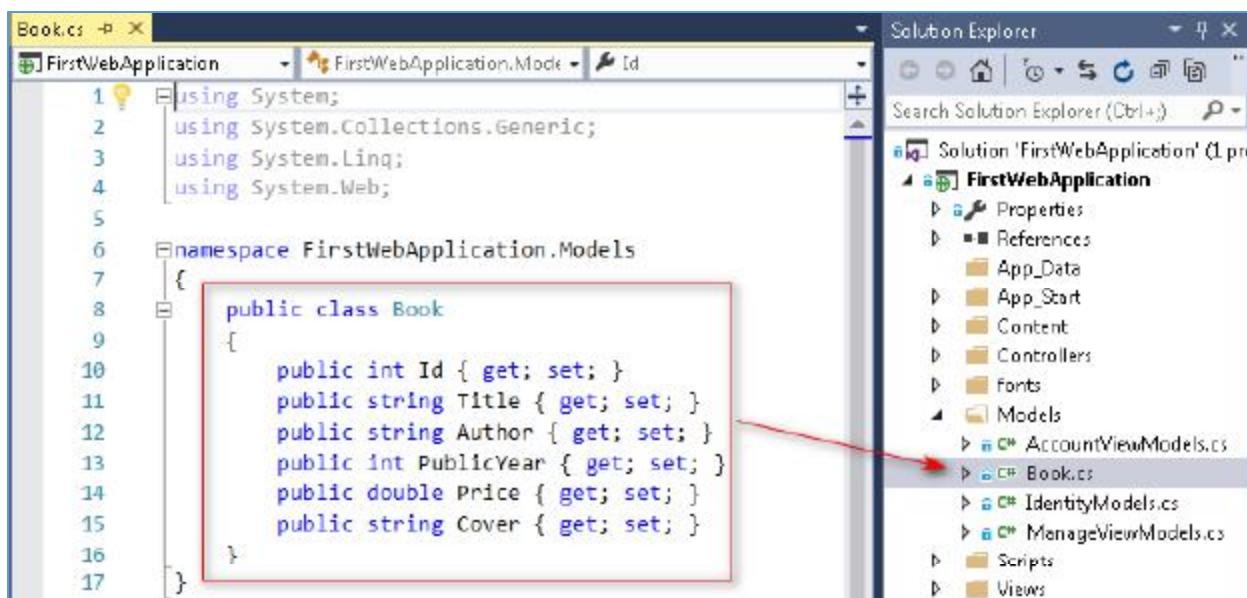
Câu 2: Luồng đi trong mô hình MVC như thế nào?

BÀI 2: VIẾT TRANG WEB QUẢN LÝ SÁCH VỚI ASP.NET MVC

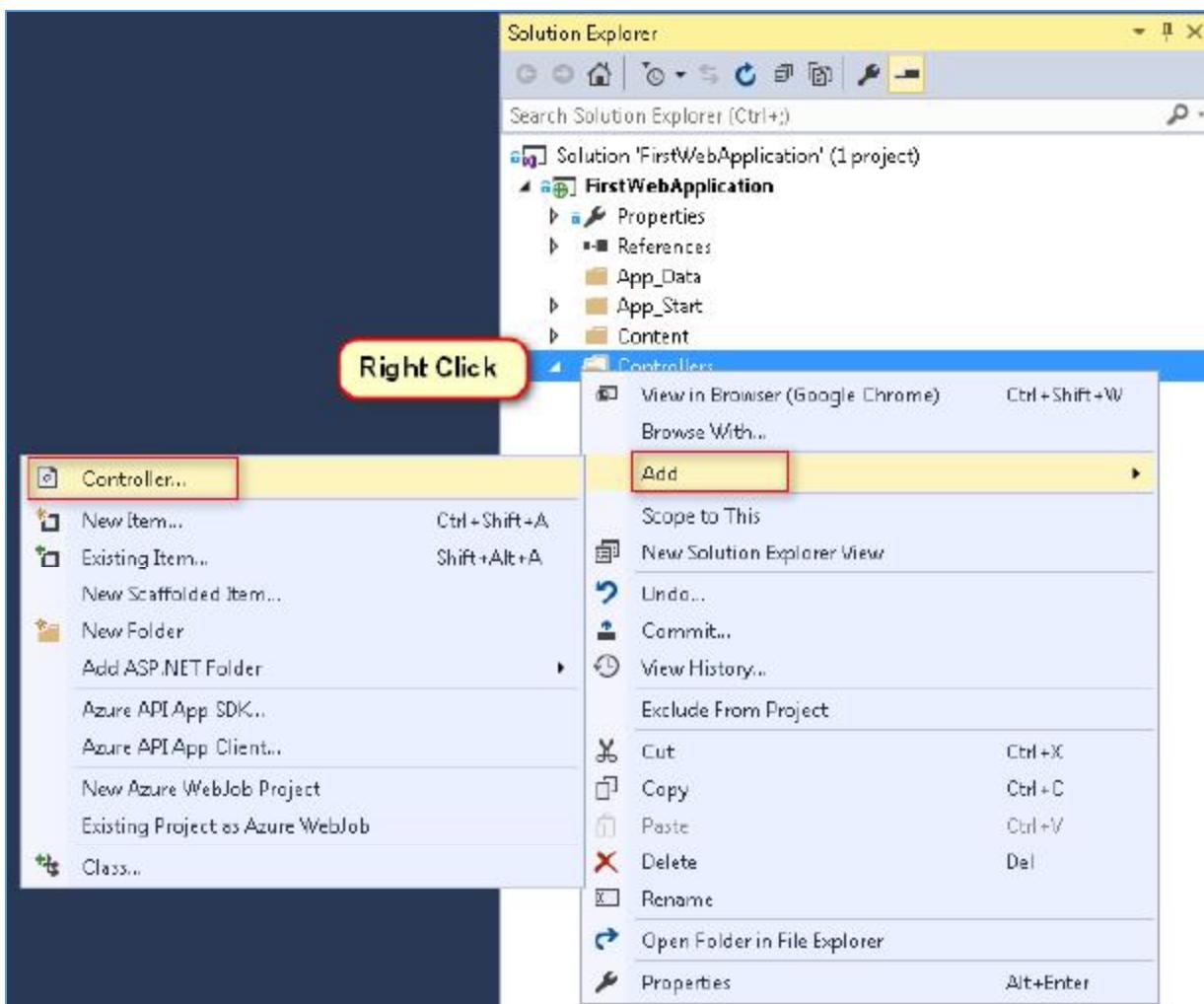
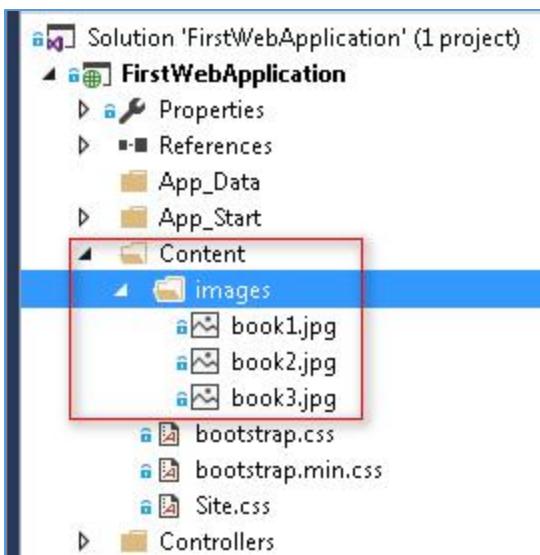
Bài này giúp người học nắm được các nội dung sau:

- *Lập trình cơ bản ứng dụng Website với ASP.NET MVC 5*
- *Hiểu được cơ chế làm việc của Model, View, Controller*

2.1 TẠO MODEL BOOK



File Hình ảnh bìa sách (Sinh viên tự lấy một số hình ảnh đưa vào thư mục như hình để demo)

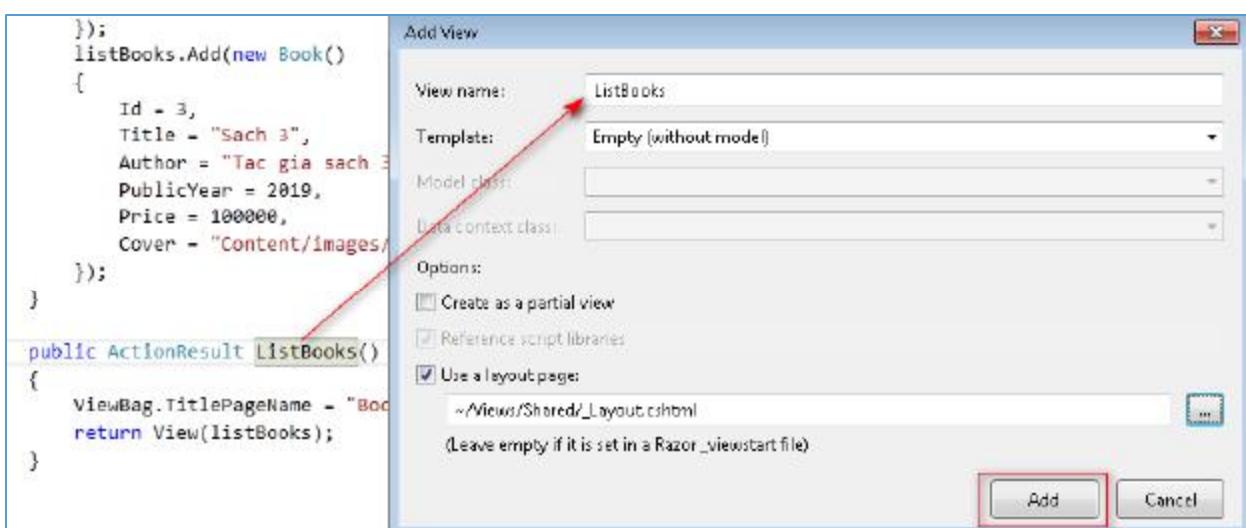


Đặt tên BooksController.cs

```
public class BooksController : Controller
{
    private List<Book> listBooks;
    public BooksController()
    {
        listBooks = new List<Book>();
        listBooks.Add(new Book()
        {
            Id = 1,
            Title = "Sách 1",
            Author = "Tác giả sách 1",
            PublicYear = 2017,
            Price = 40000,
            Cover ="Content/images/book1.jpg"
        });
        listBooks.Add(new Book()
        {
            Id = 2,
            Title = "Sách 2",
            Author = "Tác giả sách 2",
            PublicYear = 2018,
            Price = 50000,
            Cover = "Content/images/book2.jpg"
        });
        listBooks.Add(new Book()
        {
            Id = 3,
            Title = "Sách 3",
            Author = "Tác giả sách 3",
            PublicYear = 2019,
            Price = 100000,
            Cover = "Content/images/book3.jpg"
        });
    }

    public ActionResult ListBooks()
    {
        ViewBag.TitlePageName = "Book view page";
        return View(listBooks);
    }
}
```

Nhấn chuột phải vào hàm **ListBooks**, chọn **Add View...**



File Views/Books/ListBooks.cshtml

```

@model IEnumerable<FirstWebApplication.Models.Book>
{
    ViewBag.Title = "ListBooks";
    Layout = "~/Views/Shared/_Layout.cshtml";
}



## @ViewBag.TitlePageName



@foreach (var item in Model)
{
    <div class="col-lg-4 col-md-4 col-sm-6 col-xs-12">
        <div class="panel panel-default">
            <div class="panel panel-heading">@item.Title</div>
            <div class="panel-body">
                <p>
                    <strong>Author: </strong> @item.Author
                </p>
                <p>
                    <strong>Public Year: </strong> @item.PublicYear
                </p>
                <p>
                    <strong>Price: </strong> @item.Price
                </p>
                <p>
                    
                </p>
            </div>
            <div class="panel-footer clearfix">
                <div class="pull-right">
                    <a href="#" class="btn btn-primary">Order</a>
                    <a href="#" class="btn btn-default">Read more...</a>
                </div>
            </div>
        </div>
    }


```

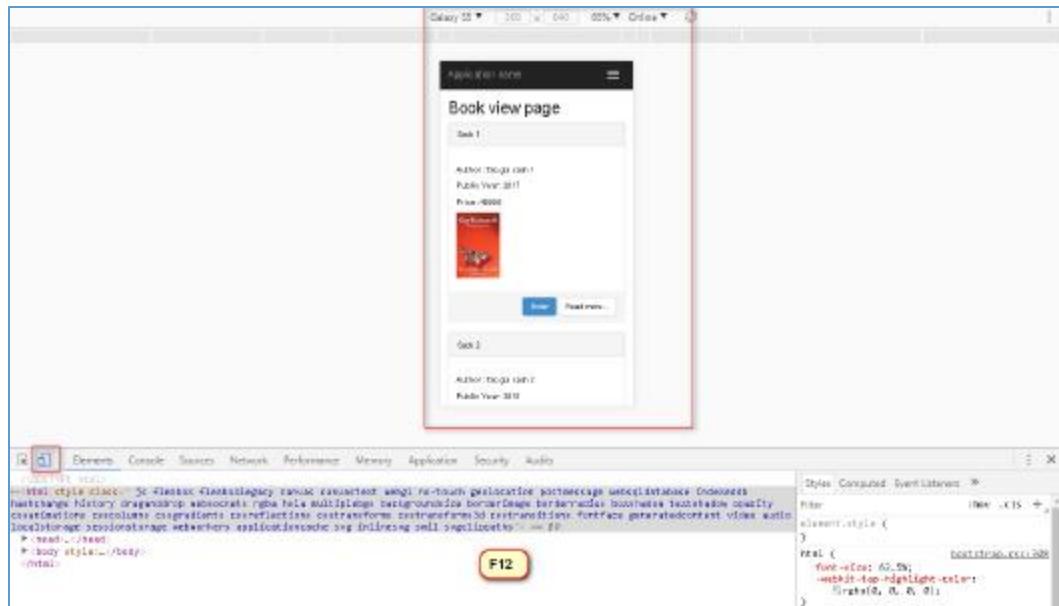
File App_Start/RouteConfig.cs

```
public class RouteConfig
{
    public static void RegisterRoutes(RouteCollection routes)
    {
        routes.IgnoreRoute("{resource}.axd/{*pathInfo}");

        routes.MapRoute(
            name: "Default",
            url: "{controller}/{action}/{id}",
            defaults: new { controller = "Books", action = "ListBooks", id =
                UrlParameter.Optional }
        );
    }
}
```

Xem kết quả khi chạy **website**

Xem kết quả trên thiết bị di động

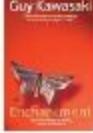
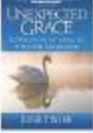


Xem chi tiết một cuốn sách (**Book Detail**)

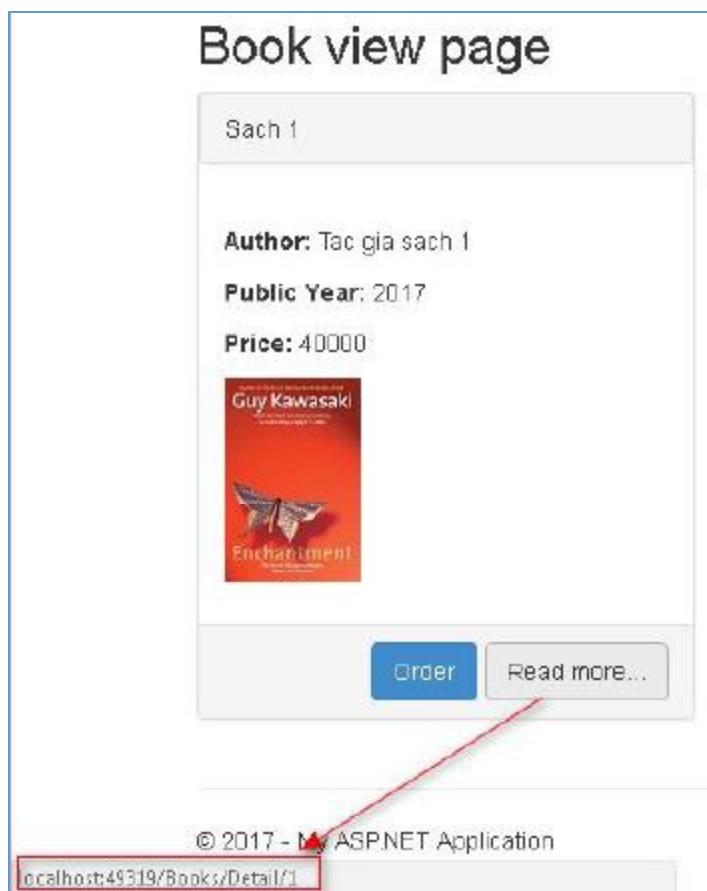
Chỉnh sửa file Views/Books/ListBook.cshtml

```
<div class="row">
    @foreach (var item in Model)
    {
        <div class="col-lg-4 col-md-4 col-sm-6 col-xs-12">
            <div class="panel panel-default">
                <div class="panel-heading">@item.Title</div>
                <div class="panel-body">
                    <p>
                        <strong>Author: </strong> @item.Author
                    </p>
                    <p>
                        <strong>Public Year: </strong> @item.PublicYear
                    </p>
                    <p>
                        <strong>Price: </strong> @item.Price
                    </p>
                    <p>
                        
                    </p>
                </div>
                <div class="panel-footer clearfix">
                    <div class="pull-right">
                        <a href="#" class="btn btn-primary">Order</a>
                        <a href="#" class="btn btn-default">Read more...</a>
                        @Html.ActionLink("Read more...", "Detail", new { id=@item.Id }, new { @class= "btn btn-default" })
                    </div>
                </div>
            </div>
        </div>
    }
</div>
```

Book view page

Sách 1	Sách 2	Sách 3
Author: Tác giả sách 1 Public Year: 2017 Price: 40000  Order Read more... Read more...	Author: Tác giả sách 2 Public Year: 2018 Price: 50000  Order Read more... Read more...	Author: Tác giả sách 3 Public Year: 2019 Price: 100000  Order Read more... Read more...

Xem Link



Code behind (Controller):

```
public ActionResult ListBooks()
{
    ViewBag.TitlePageName = "Book view page";
    return View(listBooks);
}

public ActionResult Detail(int? id)
{
    if (id == null)
    {
        return HttpNotFound();
    }
    Book book = listBooks.Find(s => s.Id == id);

    if (book == null)
        return HttpNotFound();

    return View(book);
}
```

Add View dialog:

Screenshot of the 'Add View' dialog in Visual Studio. The 'View name:' dropdown is set to 'Detail'. The 'Template:' dropdown is set to 'Details'. The 'Model class:' dropdown is set to 'Book (FirstWebApplication.Models)'. Other options like 'Create as a partial view' and 'Use a layout page' are checked.

Trang Views/Books/Detail.cshtml

```
@model FirstWebApplication.Models.Book

{
    ViewBag.Title = "Detail";
    Layout = "~/Views/Shared/_Layout.cshtml";
}

<h2>@Html.DisplayFor(model => model.Title)</h2>
<div class="col-sm-12 col-md-12">
    <div class="thumbnail">
        
    </div>
    <div class="caption">
        <p>@Html.DisplayNameFor(model => model.Author): @Html.DisplayFor(model => model.Author)</p>
        <p>@Html.DisplayNameFor(model => model.PublicYear): @Html.DisplayFor(model => model.PublicYear)</p>
        <p>@Html.DisplayNameFor(model => model.Price): @Html.DisplayFor(model => model.Price)</p>
        <p>
            <a href="#" class="btn btn-primary" role="button">
                Buy
            </a>
            @Html.ActionLink("Back", "ListBooks", "", new { @class = "btn btn-info" })
        </p>
    </div>
</div>
</div>
```

Trang chỉnh sửa Sách (Edit Book)

File Views/Books/ListBooks.cshtml

```
@model IEnumerable<FirstWebApplication.Models.Book>
@{
    ViewBag.Title = "ListBooks";
    Layout = "~/Views/Shared/_Layout.cshtml";
}

<h2>@ViewBag.TitlePageName</h2>

<div class="row">
    @foreach (var item in Model)
    {
        <div class="col-lg-4 col-md-4 col-sm-6 col-xs-12">
            <div class="panel panel-default">
                <div class="panel panel-heading">
                    @item.Title
                    <@Html.ActionLink("Edit", "Edit", new { id = @item.Id }, new { @class = "pull-right" })>
                </div>
                <div class="panel-body">...</div>
                <div class="panel-footer clearfix">...</div>
            </div>
        </div>
    }
</div>
```

File BooksController.cs

```

public ActionResult Detail(int? id)
{
    if (id == null)
    {
        return HttpNotFound();
    }
    Book book = listBooks.Find(s => s.Id == id);

    if (book == null)
        return HttpNotFound();

    return View(book);
}

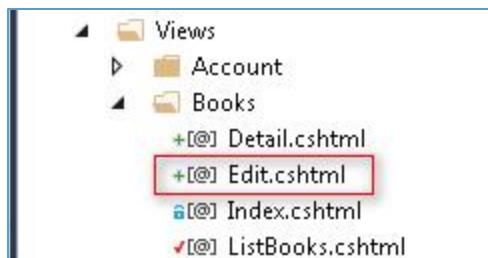
public ActionResult Edit(int? id)
{
    if (id == null)
    {
        return HttpNotFound();
    }
    Book book = listBooks.Find(s => s.Id == id);

    if (book == null)
        return HttpNotFound();

    return View(book);
}

```

Kiểm tra file **Views/Books/Edit.cshtml** đã tồn tại



Chạy ứng dụng, kiểm tra kết quả

Viết hàm lưu kết quả sau khi chỉnh sửa

2.2 RÀNG BUỘC DỮ LIỆU NHẬP

File Models/Book.cs

```
using System.ComponentModel.DataAnnotations;

namespace FirstWebApplication.Models
{
    public class Book
    {
        public int Id { get; set; }

        [Required(ErrorMessage = "Please input Title")]
        public string Title { get; set; }

        [Required(ErrorMessage = "Please input Author")]
        [StringLength(50, ErrorMessage = "Author less than 50 characters")]
        public string Author { get; set; }

        public int PublicYear { get; set; }

        public double Price { get; set; }

        public string Cover { get; set; }
    }
}
```

File Views/Books/Edit.cshtml

```
@model FirstWebApplication.Models.Book

@{
    ViewBag.Title = "Edit";
    Layout = "~/Views/Shared/_Layout.cshtml";
}



## Edit



@using (Html.BeginForm())
{
    @Html.AntiForgeryToken()

    <div class="form-horizontal">...</div>
}

<div>
    @Html.ActionLink("Back to List", "Index")
</div>

@section Scripts {
    @Scripts.Render("~/bundles/jqueryval")
}
```

Chạy ứng dụng và kiểm tra kết quả

- ❖ Viết hàm lưu kết quả

File Controllers/BooksController.cs

```
[HttpPost]
[ValidateAntiForgeryToken]
public ActionResult Edit(Book book)
{
    if (ModelState.IsValid)
    {
        try
        {
            var editBook = listBooks.Find(b => b.Id == book.Id);
            editBook.Title = book.Title;
            editBook.Author = book.Author;
            editBook.Cover = book.Cover;
            editBook.Price = book.Price;
            editBook.PublicYear = book.PublicYear;
            return View("ListBooks", listBooks);
        }
        catch (Exception ex)
        {
            //Log exception ex
            return HttpNotFound();
        }
    }
    else
    {
        ModelState.AddModelError("", "Input Model Not Valid!");
        return View(book);
    }
}
```

Chạy ứng dụng để kiểm tra kết quả, lưu thành công sẽ trả về trang danh sách với kết quả mới

TÓM TẮT

Mô hình MVC và các thành phần bên trong của MVC.

Model (M): Là bộ phận có chức năng lưu trữ toàn bộ dữ liệu của ứng dụng. Bộ phận này là một cầu nối giữa 2 thành phần bên dưới là View và Controller. Model thể hiện dưới hình thức là một cơ sở dữ liệu hoặc có khi chỉ đơn giản là một file XML bình thường. Model thể hiện rõ các thao tác với cơ sở dữ liệu như cho phép xem, truy xuất, xử lý dữ liệu,...

View (V): Đây là phần giao diện (theme) dành cho người sử dụng. Nơi mà người dùng có thể lấy được thông tin dữ liệu của MVC thông qua các thao tác truy vấn như tìm kiếm hoặc sử dụng thông qua các website.

Thông thường, các ứng dụng web sử dụng MVC View như một phần của hệ thống, nơi các thành phần HTML được tạo ra. Bên cạnh đó, View cũng có chức năng ghi nhận hoạt động của người dùng để tương tác với Controller. Tuy nhiên, View không có mối quan hệ trực tiếp với Controller, cũng không được lấy dữ liệu từ Controller mà chỉ hiển thị yêu cầu chuyển cho Controller mà thôi.

Controller (C): Bộ phận có nhiệm vụ xử lý các yêu cầu người dùng đưa đến thông qua view. Từ đó, Controller đưa ra dữ liệu phù hợp với người dùng. Bên cạnh đó, Controller còn có chức năng kết nối với model.

CÂU HỎI ÔN TẬP

Câu 1: Ưu & nhược điểm của MVC?

Câu 2: Ứng dụng mô hình MVC vào lập trình?

BÀI 3: THỰC HÀNH THEO DỰ ÁN WEBSITE BIG - SCHOOL

Bài này giúp người học nắm được các nội dung sau:

- Xây dựng giao diện người sử dụng: HTML, CSS, Bootstrap
- Xây dựng các cấu trúc tái sử dụng
- Mô hình Entity Framework code First work-flow
- Restful APIs
- Lập trình Ajax
- Bảo mật, phân quyền
- Thiết kế phần mềm hướng đối tượng
- Kiến trúc hệ thống phần mềm
- Kiểm thử tự động (Automated Testing)

3.1 CÔNG NGHỆ SỬ DỤNG

- ASP.NET MVC 5
- Entity Framework 6
- ASP.NET Identity
- Bootstrap
- Bootbox js
- Jquery

3.2 MÔ TẢ YÊU CẦU

Xây dựng mạng xã hội quản lý và theo dõi các khóa học trực tuyến (gọi tắt là BigSchool). Webstie Mạng xã hội học tập BigSchool cung cấp các chức năng cơ bản sau:

- Người học có thể dễ dàng theo dõi các khóa học của Giảng viên yêu thích. Thêm khóa học vào danh sách đăng ký tham dự. Theo dõi Giảng viên để nhận các thông tin về khóa học mới.
- Người học có thể xem tất cả các khóa học sắp được mở theo giảng viên hoặc tìm kiếm theo các thông tin như tên giảng viên, chủ đề khóa học, địa điểm khóa học.
- Giảng viên có thể đăng ký mở khóa học, bao gồm các thông tin ngày/ giờ, địa điểm và chủ đề của khóa học. Giảng viên có trang quản lý khóa học để có thể chỉnh sửa, cập nhật khóa học.

3.3 PHẦN MỀM

Sử dụng các phần mềm sau để thực hành:

- IDE: Visual studio community 2015
- MSSQL Server >= 2008
- [Tùy chọn] cài đặt các Extension cho VS 2015 (Tool → Extension & Update)
 - ReSharper (free 30 trial license)
 - Productivity Power Tools 2013 (free)
 - Web Essentials 2015 (free)

3.4 PHÂN TÍCH YÊU CẦU CỦA ỨNG DỤNG BIGSCHOOL

3.4.1 Xác định các usecase trong dự án

- BigSchool là một dự án giúp quản lý và theo dõi khóa học
- Giảng viên đăng nhập và quản lý khóa học. Khi thêm một khóa học có các thông tin địa điểm, thời gian, chủ đề khóa học
- Giảng viên có trang quản lý khóa học bao gồm: Thêm, sửa, xóa khóa học

- Học viên xem các khóa học, tìm kiếm theo tên giảng viên, chủ đề, địa điểm.
- Học viên có thể xem chi tiết một khóa học và đăng ký tham dự
- Học viên có thể theo dõi Giảng viên. Khi theo dõi Giảng viên nào đó, học viên có thể xem các khóa học sắp tới của giảng viên đó trong BigSchool Feed.

3.4.2 Phân tích use case

3.4.2.1 Chứng thực người dùng (sử dụng ASP.NET Identity Package, được tích hợp trong ASP.NET MVC5)

Bao gồm các chức năng cơ bản:

- Đăng ký
- Đăng nhập
- Đăng xuất
- Thay đổi mật khẩu
- Thay đổi thông tin cá nhân

3.4.2.2 Khóa học

- Thêm khóa học
- Danh sách khóa học sắp diễn ra
- Sửa khóa học
- Xóa khóa học
- Xem tất cả khóa học
- Tìm kiếm
- Xem chi tiết khóa học

3.4.2.3 Đăng ký tham gia khóa học

- Đăng ký tham dự
- Hủy đăng ký tham dự

- Xem các khóa học đã đăng ký tham dự

3.4.2.4 Theo dõi

- Theo dõi một giảng viên
- Bỏ theo dõi
- Danh sách người theo dõi
- Trang tin theo dõi (Big School Feed)

3.4.3 Thứ tự use-case sẽ thực hiện

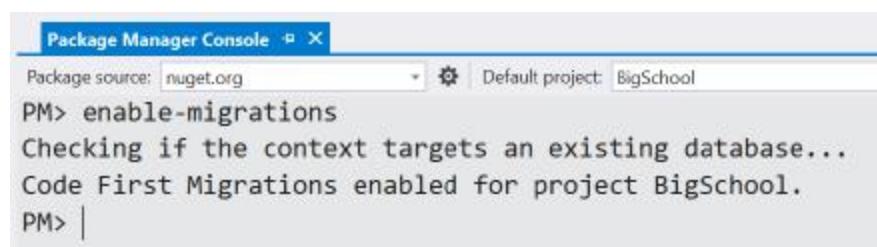
1	2	3	4	5
Thêm khóa học	Xem danh sách khóa học	Chỉnh sửa, xóa khóa học		
	Danh sách các khóa học diễn ra	Đăng ký tham dự	Xem các khóa học đã đăng ký tham dự	Hủy đăng ký tham dự
		Theo dõi giảng viên	Danh sách người theo dõi	Bỏ theo dõi
		Tìm kiếm	Big school Feed	
		Xem chi tiết khóa học		

3.5 XÂY DỰNG ỨNG DỤNG BẰNG ASP NET MVC

Tạo mới Project ASP.NET MVC trong IDE Visual studio 2015. Đặt tên project là BigSchool, và thực hiện theo hướng dẫn:

3.5.1 Tạo cơ sở dữ liệu theo mô hình EF Code First

- Tool/ Nuget package manager/ package manager console
 - enable-migrations



```
Package Manager Console  ▾ X
Package source: nuget.org   Default project: BigSchool
PM> enable-migrations
Checking if the context targets an existing database...
Code First Migrations enabled for project BigSchool.
PM> |
```

Mỗi sự thay đổi trên domain model → sẽ tạo mới bản log migration và chạy migration để cập nhật database.



- Mở file Models/IdentityModel.cs

```

namespace BigSchool.Models

// You can add profile data for the user by adding more properties.
// Please visit http://go.microsoft.com/fwlink/?LinkId=317594
public class ApplicationUser : IdentityUser
{
    public async Task<ClaimsIdentity> GenerateUserIdentityAsync(IUserManager manager)
    {
        // Note the authenticationType must match the one defined in CookieAuthenticationOptions.AuthenticationType
        var userIdentity = await manager.CreateIdentityAsync(
            this, DefaultAuthenticationTypes.ApplicationCookie);
        // Add custom user claims here
        return userIdentity;
    }
}
  
```

- IdentityUser là một phần của ASP.NET Identity, hỗ trợ xử lý các chức năng cơ bản liên quan tới việc xác thực người dùng như đăng ký, đăng nhập, đổi mật khẩu, phân quyền,... vv
- Mở file **Web.config** cấu hình trỏ liên kết tới máy chủ chạy cơ sở dữ liệu, với tên tài khoản và mật khẩu đăng nhập vào hệ quản trị CSDL.

```

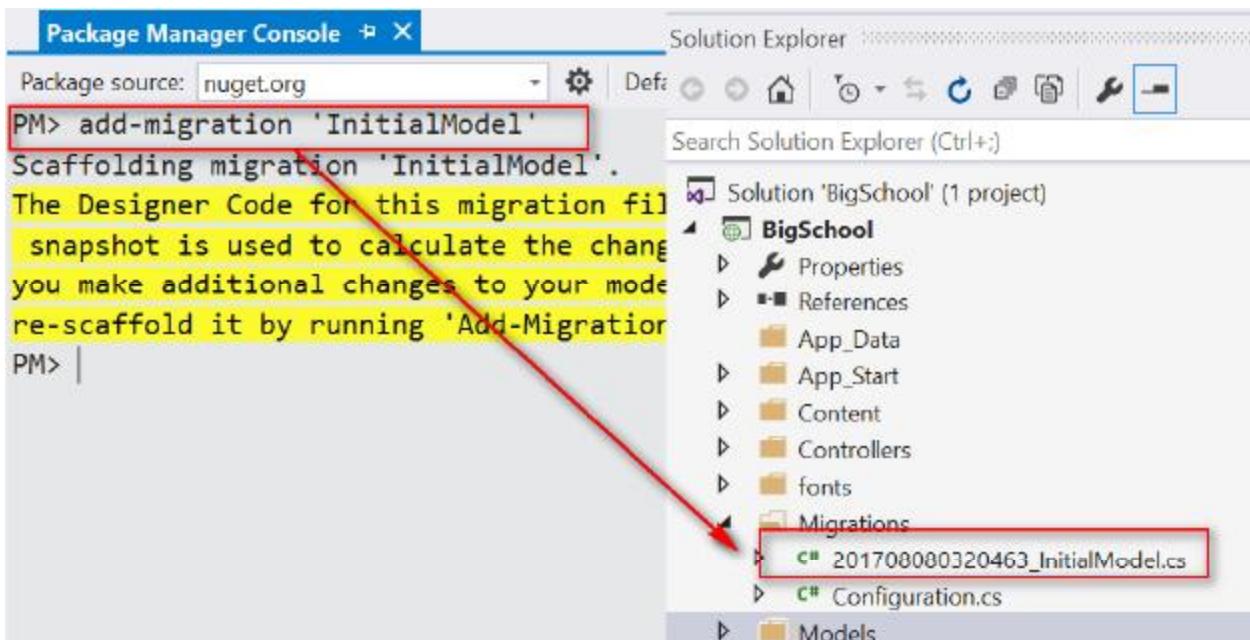
<connectionStrings>

<add name="DefaultConnection" connectionString="Data Source=DESKTOP-
AT2MHH8;Initial Catalog=BigSchool;user id = sa; pwd = anh@; Integrated Security=True" providerName="System.Data.SqlClient" />

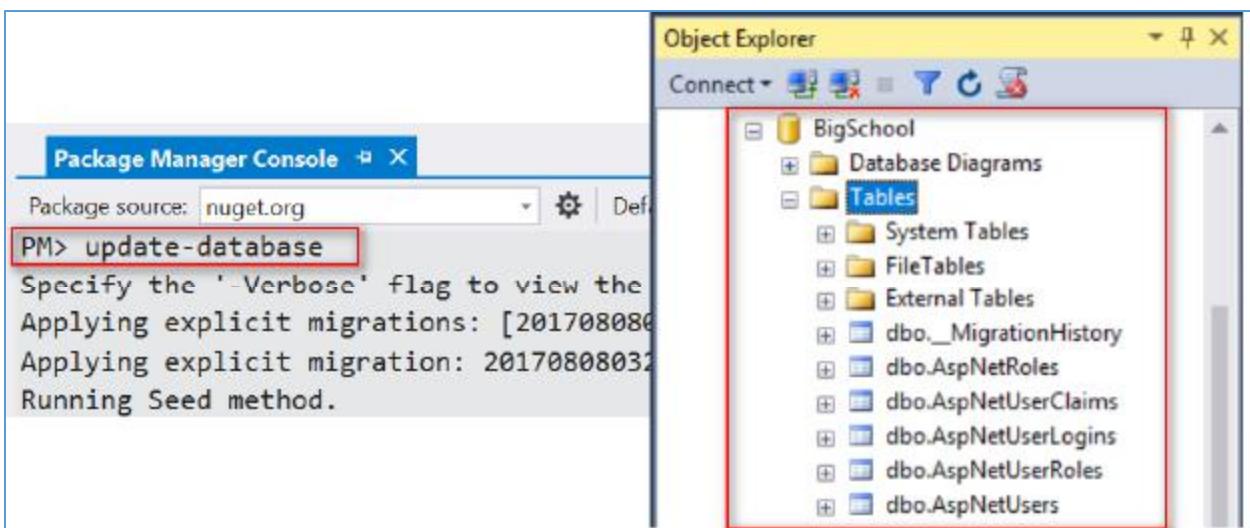
</connectionStrings>
  
```

- Mở Package Manager Console để tạo các Model của User

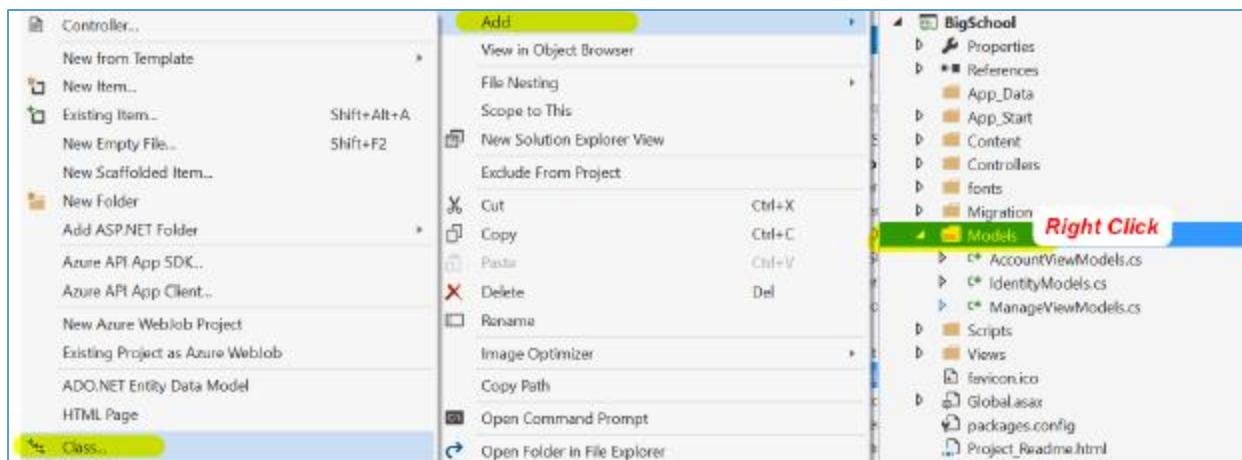
- Add-migration 'InitialModel'



- Mở Package Manager Console chạy lệnh
 - update-database: Tạo bảng trong CSDL MSSQL Server theo file migration vừa mới tạo ở bước trên (sau khi chạy lệnh này, mở chương trình quản lý database để kiểm tra CSDL BigSchool vừa được tạo)



- Tạo các Domain class của ứng dụng: Nhấn chuột phải vào thư mục Models → Add → Class: đặt tên là **Course.cs**



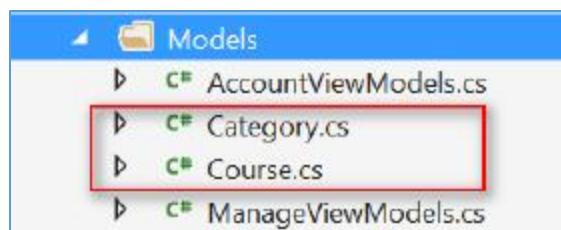
- Nội dung file **Course.cs**

```
namespace BigSchool.Models
{
    public class Course
    {
        public int Id { get; set; }

        public ApplicationUser Lecturer { get; set; }
        [Required]
        public string LecturerId { get; set; }
        [Required]
        [StringLength(255)]
        public string Place { get; set; }
        public DateTime DateTime { get; set; }
        public Category Category { get; set; }
        [Required]
        public byte CategoryId { get; set; }
    }

    public class Category
    {
        public byte Id { get; set; }
        [Required]
        [StringLength(255)]
        public string Name { get; set; }
    }
}
```

- Tách Class **Category** thành file mới, nằm trong thư mục **Models**



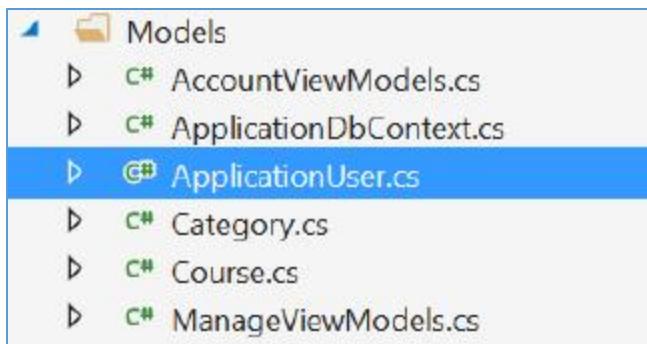
- Mở File Models/IdentityModel.cs

- Tách Class ApplicationDbContext ra một file riêng, thêm domain class Course và Category.

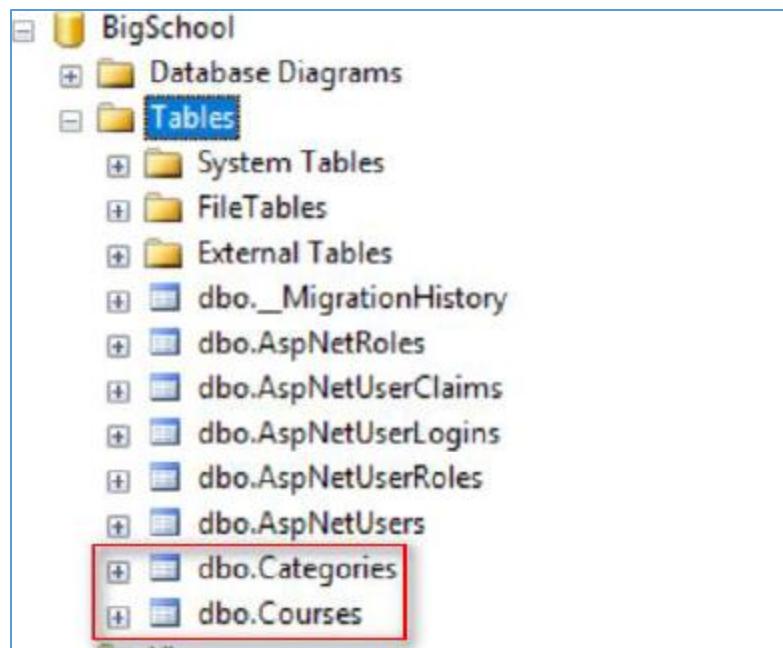
```
public class ApplicationDbContext : IdentityDbContext<ApplicationUser>
{
    public DbSet<Course> Courses { get; set; }
    public DbSet<Category> Categories { get; set; }
    public ApplicationDbContext()
        : base("DefaultConnection", throwIfV1Schema: false)
    {
    }

    public static ApplicationDbContext Create()
    {
        return new ApplicationDbContext();
    }
}
```

- Đổi tên File **Models/IdentityModes.cs** thành tên mới **Models/ApplicationUser.cs** (tên lớp và tên file nên giống nhau để dễ quản lý theo dõi thay đổi)
- Kết quả thư mục **Models** có



- Mở package Manager Console để tạo bản cập nhật database
 - add-migration 'CreateTableCourse'
 - update-database
- Kết quả (xem database)



3.5.2 Use-case: Thêm mới khóa học

Một khóa học bao gồm các thông tin như: Chủ đề (chọn từ danh sách), địa điểm, thời gian.

3.5.2.1 Tạo mẫu dữ liệu các chủ đề của khóa học

- Mở package manager control
 - add-migration PopulateCategoryTable
- Mở file migration PopulateCategoryTable.cs

```

public partial class PopulateCategoryTable : DbMigration
{
    public override void Up()
    {
        Sql("INSERT INTO CATEGORIES (ID, NAME) VALUES (1, 'Development')");
        Sql("INSERT INTO CATEGORIES (ID, NAME) VALUES (2, 'Business')");
        Sql("INSERT INTO CATEGORIES (ID, NAME) VALUES (3, 'Marketing')");
    }

    public override void Down()
    {
    }
}
  
```

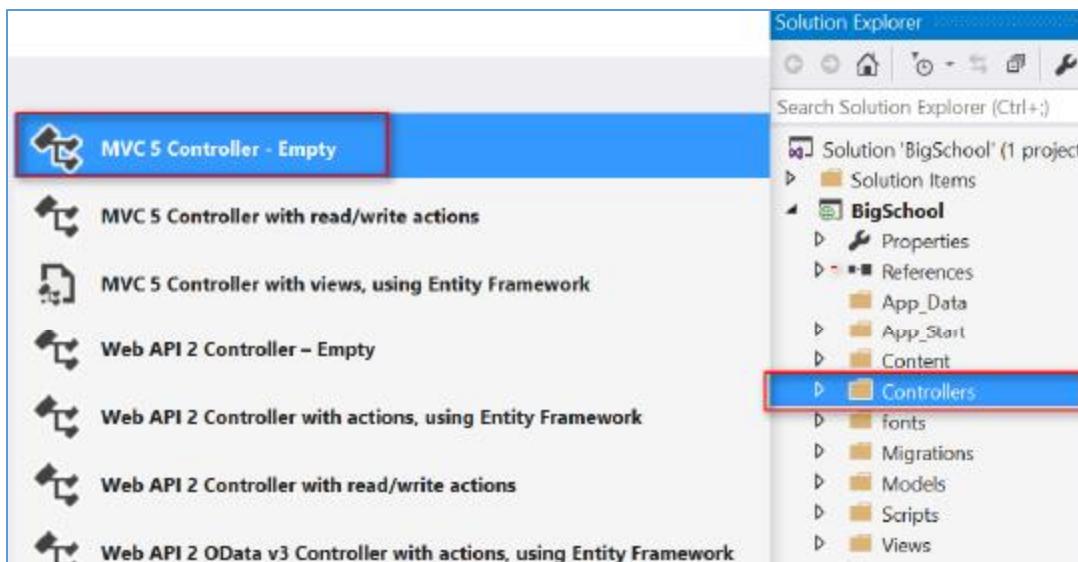
- Mở Package manager control, cập nhật dữ liệu cho bảng Category
 - update-database

- Kiểm tra dữ liệu trong bảng **Category**

Id	Name
1	Development
2	Business
3	Marketing

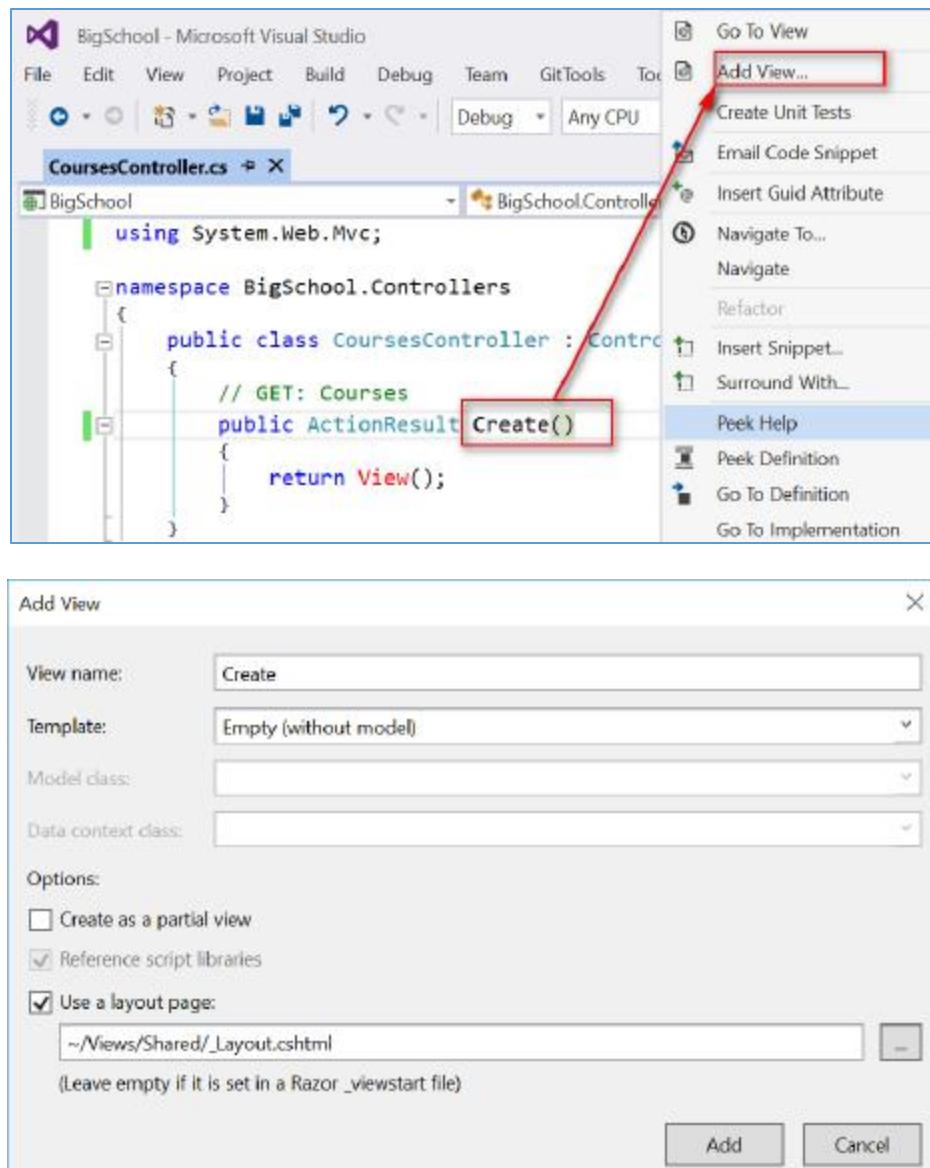
3.5.2.2 Tạo Form thêm khóa học với Bootstrap

- Nhấn chuột phải vào thư mục Controller → Chọn Add → Controller → Chọn MVC5 Empty Controller

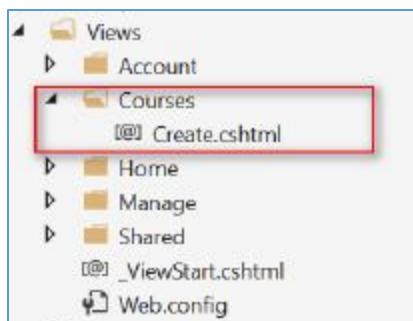


- Đặt tên: **CoursesController** → Nhấn Enter để thêm **Controller**

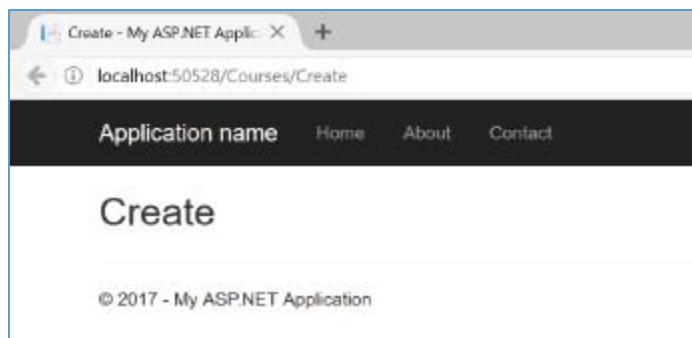
- Đổi tên ActionResult Index → Create, nhấp chuột phải vào tên Create → Chọn Add View



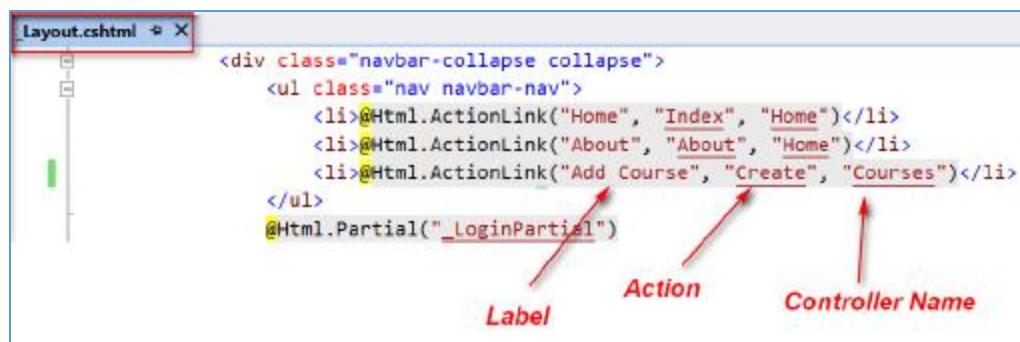
- Thư mục mới được tạo ra Views/Courses và tên View vừa tạo bước trên Create.cshtml



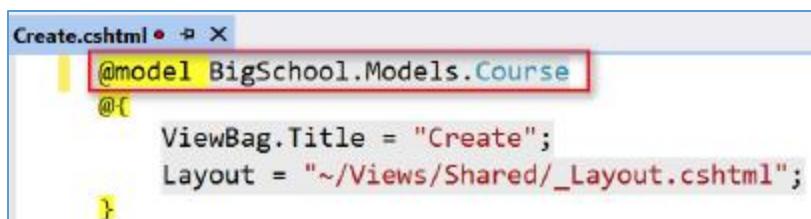
- Mở file **Views/Courses/Create.cshtml**, nhấn **F5** để xem kết quả từ trình duyệt



- Sửa menu Contact → Thành menu Create Course
 - Mở file Views/Shared/_Layout.cshtml



- Mở trình duyệt, nhấn F5 để xem lại kết quả
- Mở file Views/Courses/Create.cshtml, chỉ định Model là Course

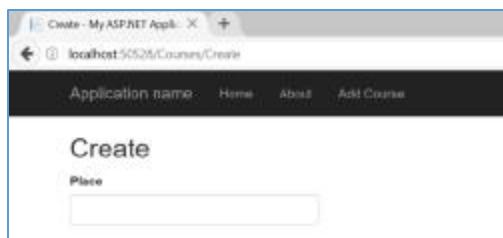


- Thêm các thành phần của khóa học như: địa điểm, thời gian

```
@model BigSchool.Models.Course
@{
    ViewBag.Title = "Create";
    Layout = "~/Views/Shared/_Layout.cshtml";
}

<h2>Create</h2>
<form>
    <div class="form-group">
        @Html.LabelFor(m => m.Place)
        @Html.TextBoxFor(m => m.Place, new { @class = "form-control" })
    </div>
</form>
```

- Được kết quả



- Xây dựng Form với Bootstrap (xem thêm bootstrap tại địa chỉ: <http://getbootstrap.com/>)



Sử dụng ViewModel

- Tiếp tục xây dựng Form khóa học

```
<h2>Create</h2>
<form>
  <div class="form-group">
    @Html.LabelFor(m => m.Place)
    @Html.TextBoxFor(m => m.Place, new { @class = "form-control" })
  </div>
  <div class="form-group">
    @Html.LabelFor(m=>m.DateTime)
  </div>
  <div class="form-group"></div>
  <div class="form-group"></div>
  <div class="form-group"></div>
  <div class="form-group"></div>
</form>
```

- Tại Trường **DateTime** của Khóa học, cho phép người dùng nhập vào 2 giá trị (**ngày học và thời gian học**)

- Tạo mới **folder** tên là **ViewModels** tại thư mục gốc
- Tạo File ViewModels/CourseViewModel.cs với nội dung:

```
public class CourseViewModel
{
    public string Place { get; set; }
    public string Date { get; set; }
    public string Time { get; set; }
}
```

- Sửa trang View thêm khóa học **Views/Course/Create.cshtml** (sử dụng model mới CourseViewModel)

```
@model BigSchool.ViewModels.CourseViewModel
ViewBag.Title = "Create";
Layout = "~/Views/Shared/_Layout.cshtml";
}

<h2>Create</h2>
<form>
    <div class="form-group">
        @Html.LabelFor(m => m.Place)
        @Html.TextBoxFor(m => m.Place, new { @class = "form-control" })
    </div>
    <div class="form-group">
        @Html.LabelFor(m=>m.Date)
        @Html.TextBoxFor(m => m.Date, new { @class = "form-control" })
    </div>
    <div class="form-group">
        @Html.LabelFor(m=>m.Time)
        @Html.TextBoxFor(m => m.Time, new { @class = "form-control" })
    </div>
    <div class="form-group"></div>
    <div class="form-group"></div>
    <div class="form-group"></div>
</form>
```

Thêm danh sách lựa chọn (Drop – down List) danh mục của khóa học

- Chính sửa file ViewModels/CourseViewModel.cs

```
public class CourseViewModel
{
    public string Place { get; set; }

    public string Date { get; set; }

    public string Time { get; set; }

    public byte Category { get; set; }
    public IEnumerable<Category> Categories { get; set; }

    public DateTime GetDateTime()
    {
        return DateTime.Parse(string.Format("{0} {1}", Date, Time));
    }
}
```

- Chính sửa trang View Khóa học: **Views/Course/Create.cshtml**

```
<form>
    <div class="form-group">
        @Html.LabelFor(m => m.Place)
        @Html.TextBoxFor(m => m.Place, new { @class = "form-control" })
    </div>
    <div class="form-group">
        @Html.LabelFor(m => m.Date)
        @Html.TextBoxFor(m => m.Date, new { @class = "form-control" })
    </div>
    <div class="form-group">
        @Html.LabelFor(m => m.Time)
        @Html.TextBoxFor(m => m.Time, new { @class = "form-control" })
    </div>
    <div class="form-group">
        @Html.LabelFor(m => m.Category)
        @Html.DropDownListFor(m => m.Category,
            new SelectList(Model.Categories, "Id", "Name"), "", new { @class = "form-control" })
    </div>
</form>
```

- Mở file Controllers/CourseController.cs

```
public class CoursesController : Controller
{
    private readonly ApplicationDbContext _dbContext;

    public CoursesController()
    {
        _dbContext = new ApplicationDbContext();
    }
    // GET: Courses
    public ActionResult Create()
    {
        var viewModel = new CourseViewModel
        {
            Categories = _dbContext.Categories.ToList()
        };
        return View(viewModel);
    }
}
```

- Chạy ứng dụng và xem kết quả

localhost:50528/Courses/Create

Application name Home About Add Course

Create

Place

Date

Time

Category

- Development
- Business
- Marketing

- ❖ Thêm nút button Lưu kết quả trên Form (views/Course/Create.cshtml)

```
<div class="form-group">
    @Html.LabelFor(m => m.Category)
    @Html.DropDownListFor(m => m.Category,
        new SelectList(Model.Categories, "Id", "Name"), "", new { @class = "form-control" })
</div>
<button type="submit" value="Save" class="btn btn-primary">Save</button>
```

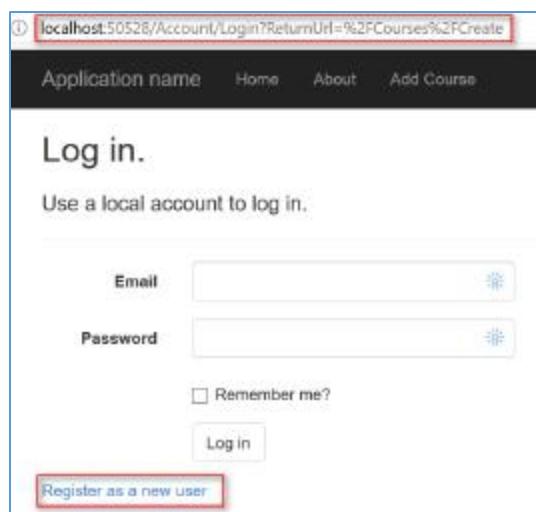
- ❖ Lưu dữ liệu khóa học

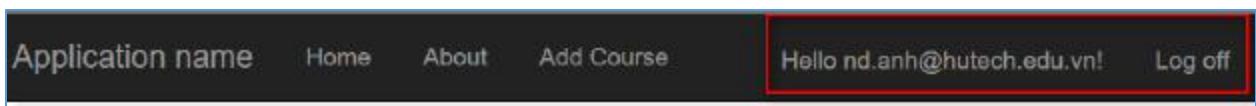
Xác thực quyền đăng nhập với ASP.NET Identity. Chỉ cho phép người dùng có tài khoản và đã đăng nhập vào hệ thống mới có thể thêm khóa học.

- File Controllers/CourseController.cs

```
// GET: Courses
[Authorize]
public ActionResult Create()
{
    var viewModel = new CourseViewModel
    {
        Categories = _dbContext.Categories.ToList()
    };
    return View(viewModel);
}
```

- Chạy ứng dụng website, vào trang thêm khóa học (<http://localhost:50528/Courses/Create>), hệ thống sẽ yêu cầu đăng nhập.
- Tạo tài khoản mới (**Register as a new user**) và đăng nhập trước khi vào trang thêm mới khóa học





- File Views/Course/Create.cshtml

```
@using (Html.BeginForm("Create", "Courses"))
{
    <div class="form-group">
        @Html.LabelFor(m => m.Place)
        @Html.TextBoxFor(m => m.Place, new { @class = "form-control" })
    </div>
    <div class="form-group">
        @Html.LabelFor(m => m.Date)
        @Html.TextBoxFor(m => m.Date, new { @class = "form-control" })
    </div>
    <div class="form-group">
        @Html.LabelFor(m => m.Time)
        @Html.TextBoxFor(m => m.Time, new { @class = "form-control" })
    </div>
    <div class="form-group">
        @Html.LabelFor(m => m.Category)
        @Html.DropDownListFor(m => m.Category,
            new SelectList(Model.Categories, "Id", "Name"), "", new { @class = "form-control" })
    </div>
    <button type="submit" value="Save" class="btn btn-primary">Save</button>
}
```

Annotations on the code:

- A red box highlights the first line: `@using (Html.BeginForm("Create", "Courses"))`. A red arrow points from this box to the text 'Action Method'.
- A red box highlights the last line: `<button type="submit" value="Save" class="btn btn-primary">Save</button>`. A red arrow points from this box to the text 'Controller Name'.
- The text 'Action Method' is written in red next to the highlighted line.
- The text 'Controller Name' is written in red next to the highlighted line.

- File Controllers/CoursesController.cs

```
[Authorize]
[HttpPost]
public ActionResult Create(CourseViewModel viewModel)
{
    var course = new Course
    {
        LecturerId = User.Identity.GetUserId(),
        DateTime = viewModel.GetDateTime(),
        CategoryId = viewModel.Category,
        Place = viewModel.Place
    };
    _dbContext.Courses.Add(course);
    _dbContext.SaveChanges();

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

- Chạy ứng dụng web, kiểm tra kết quả (<http://localhost:50528/Courses/Create>)

The screenshot shows a 'Create' form with the following fields:

- Place:** Hutech
- Date:** 27/6/2017
- Time:** 07:00
- Category:** Development

A blue 'Save' button is at the bottom.

- ❖ Ràng buộc dữ liệu nhập trên form thêm khóa học

Server-side Validation

- File ViewModels/CourseViewModel.cs

```
public class CourseViewModel
{
    [Required]
    public string Place { get; set; }

    [Required]
    public string Date { get; set; }

    [Required]
    public string Time { get; set; }

    [Required]
    public byte Category { get; set; }
}
```

Hiển thị thông báo nếu không nhập dữ liệu có ràng buộc (bắt buộc nhập)

- File Views/Course/Create.cshtml

```
@using (Html.BeginForm("Create", "Courses"))
{
    <div class="form-group">
        @Html.LabelFor(m => m.Place)
        @Html.TextBoxFor(m => m.Place, new { @class = "form-control" })
        @Html.ValidationMessageFor(m => m.Place)
    </div>
    <div class="form-group">
        @Html.LabelFor(m => m.Date)
        @Html.TextBoxFor(m => m.Date, new { @class = "form-control" })
        @Html.ValidationMessageFor(m => m.Date)
    </div>
    <div class="form-group">
        @Html.LabelFor(m => m.Time)
        @Html.TextBoxFor(m => m.Time, new { @class = "form-control" })
        @Html.ValidationMessageFor(m => m.Time)
    </div>
    <div class="form-group">
        @Html.LabelFor(m => m.Category)
        @Html.DropDownListFor(m => m.Category,
            new SelectList(Model.Categories, "Id", "Name"), "", new { @class = "form-control" })
        @Html.ValidationMessageFor(m => m.Category)
    </div>
}
```

- File **Controllers/CoursesController.cs** (Kiểm tra dữ liệu nhập phía Server, trước khi lưu vào cơ sở dữ liệu, nếu nhập sai yêu cầu lỗi sẽ được gửi lại trang đang thao tác)

```
public ActionResult Create(CourseViewModel viewModel)
{
    if (!ModelState.IsValid)
    {
        viewModel.Categories = _dbContext.Categories.ToList();
        return View("Create", viewModel);
    }
    var course = new Course
    {
        LecturerId = User.Identity.GetUserId(),
        DateTime = viewModel.GetDateTime(),
        CategoryId = viewModel.Category,
        Place = viewModel.Place
    };
    _dbContext.Courses.Add(course);
    _dbContext.SaveChanges();

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

- Chạy ứng dụng web, thêm khóa học (chú ý: không nhập địa điểm hoặc các trường ràng buộc thì sẽ báo lỗi)

The screenshot shows a 'Create' form with four fields: 'Place', 'Date', 'Time', and 'Category'. The 'Place' field is highlighted with a red border and contains the error message 'The Place field is required.'. The other fields ('Date', 'Time', 'Category') are filled with valid data: '27/6/2017', '07:00', and 'Marketing' respectively. A 'Save' button is at the bottom.

Custom validation

- Ràng buộc giá trị nhập ngày tháng của khóa học, phải lớn hơn ngày hiện tại và đúng định dạng kiểu (**dd/MM/yyyy**)
- Tạo mới file **ViewModels/FutureDate.cs** nằm trong thư mục **ViewModels** với nội dung

```
public class FutureDate : ValidationAttribute
{
    public override bool IsValid(object value)
    {
        DateTime dateTime;
        var isValid = DateTime.TryParseExact(Convert.ToString(value),
            "dd/MM/yyyy",
            CultureInfo.CurrentCulture,
            DateTimeStyles.None,
            out dateTime);

        return (isValid && dateTime > DateTime.Now);
    }
}
```

- File ViewModels/CustomViewModel.cs

```
public class CourseViewModel
{
    [Required]
    public string Place { get; set; }

    [Required]
    [FutureDate]
    public string Date { get; set; }

    [Required]
    public string Time { get; set; }
```

- Chạy ứng dụng web, kiểm tra ràng buộc custom validation tại ô Ngày, tháng khóa học (ngày nhập phải đúng định dạng dd/mm/yyyy và phải lớn hơn ngày hiện tại)

Create

Place

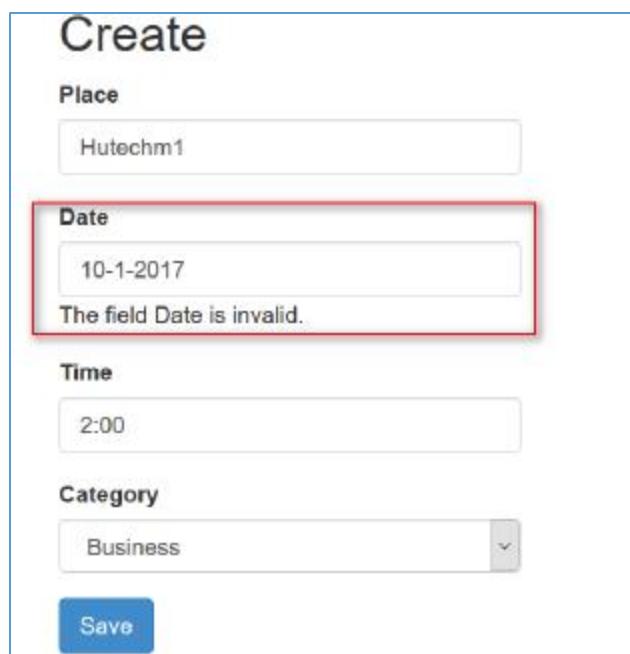
Date

The field Date is invalid.

Time

Category

Save



- Tương tự tạo Custom Validation cho ô nhập Time (Tạo mới file đặt tên là ValidTime.cs nằm trong thư mục ViewModels) và chỉnh sửa file CourseViewModel.cs (ViewModels/CourseViewModel.cs)

```

public class ValidTime : ValidationAttribute
{
    public override bool IsValid(object value)
    {
        DateTime dateTime;
        var isValid = DateTime.TryParseExact(Convert.ToString(value),
            "HH:mm",
            CultureInfo.CurrentCulture,
            DateTimeStyles.None,
            out dateTime);
        return isValid;
    }
}

public class CourseViewModel
{
    [Required]
    public string Place { get; set; }

    [Required]
    [FutureDate]
    public string Date { get; set; }

    [Required]
    [ValidTime]
    public string Time { get; set; }
}

```

- Chạy ứng dụng web, nhập giá trị vào ô **Time** không hợp lệ và kiểm tra kết quả.
- ❖ Client-side Validation (kiểm tra ràng buộc dữ liệu nhập ngay phía client trước khi gửi về server)
- File Views/Course/Create.cshtml

```

<h2>Create</h2>
@using (Html.BeginForm("Create", "Courses"))
{
    <div class="form-group">...</div>
    <div class="form-group">...</div>
    <div class="form-group">...</div>
    <div class="form-group">...</div>
    <button type="submit" value="Save" class="btn btn-primary">Save</button>
}
@section scripts
{
    @Scripts.Render("~/bundles/jqueryval");
}

```

- Chạy ứng dụng, kiểm tra kết quả

The screenshot shows a browser developer tools interface with the Network tab selected. At the top, there are validation messages for two fields:

- Time**: The Time field is required.
- Category**: The Category field is required.

Below the validation messages is a blue "Save" button. The Network tab has several sub-options: All, HTML, CSS, JS, XHR, Fonts, Images, Media, Flash, WS, and Other. The "All" option is selected. At the bottom of the Network tab, there are filters for Status, Method, File, Domain, and Cause.

Below the developer tools, there are two bullet points:

- Perform a request or Reload the page ***Không gửi request tới server***
- Click on the button to start performance analysis.

❖ BẢO MẬT

Phòng chống lỗi tấn công **CROSS SITE REQUEST FORGERY - CSRF** (Dữ liệu được gửi lên server, không phải từ web page trong hệ thống)

- File Views/Course/Create.cshtml, File Controllers/CoursesController.cs

The screenshot shows the `Create.cshtml` view and the `CoursesController.cs`.

Create.cshtml:

```

<h2>Create</h2>
@using (Html.BeginForm("Create", "Courses"))
{
    @Html.AntiForgeryToken()
    <div class="form-group">
        @Html.LabelFor(m => m.Place)
        @Html.TextBoxFor(m => m.Place, new { })
        @Html.ValidationMessageFor(m=>m.Place)
    </div>
    <div class="form-group">...</div>
    <div class="form-group">...</div>
    <div class="form-group">...</div>
    <button type="submit" value="Save" class='</button>
}
  
```

CoursesController.cs:

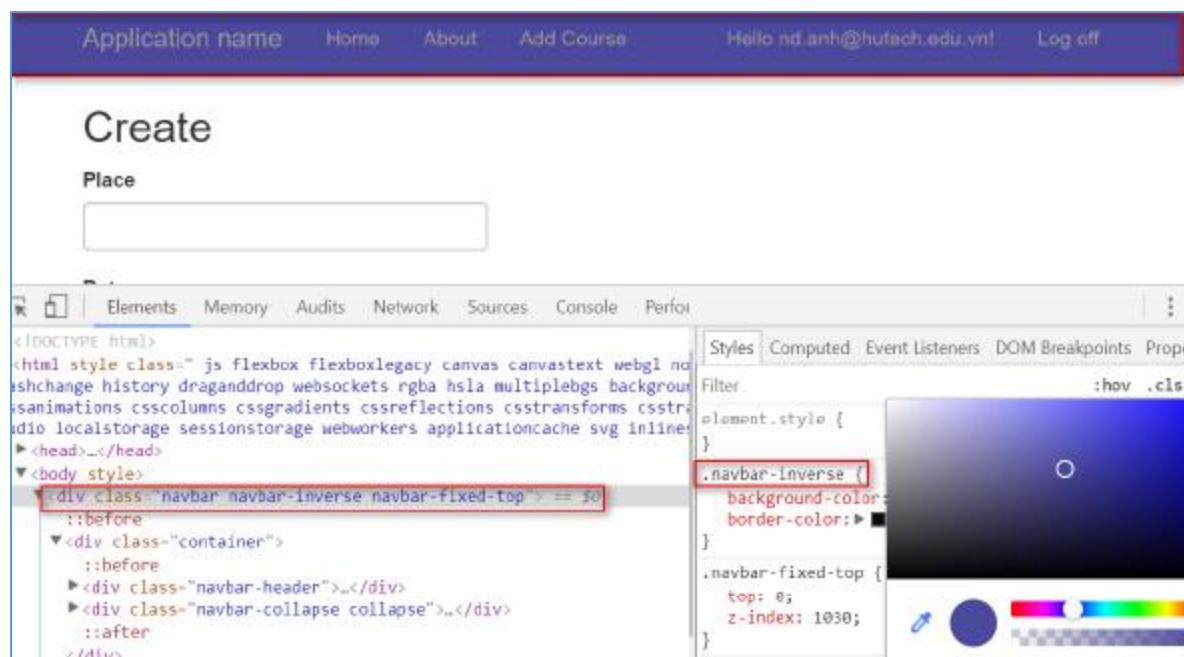
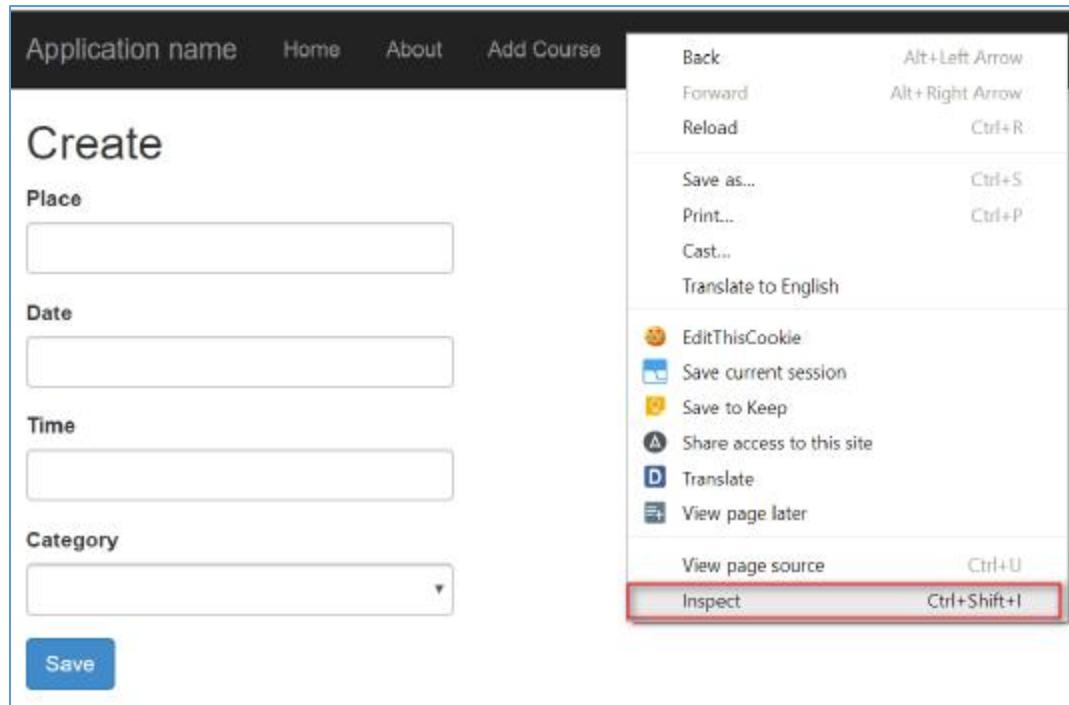
```

[Authorize]
[HttpPost]
[ValidateAntiForgeryToken]
public ActionResult Create(CourseView
{
    if (!ModelState.IsValid)
    {
        viewModel.Categories = _dbCo
        return View("Create", viewModel);
    }
    var course = new Course
    {
        Name = viewModel.Name,
        Description = viewModel.Description,
        Place = viewModel.Place,
        CategoryId = viewModel.Cat
    }
    _dbCourses.Add(course);
    _dbCourses.SaveChanges();
    return RedirectToAction("Index");
}
  
```

A red arrow points from the `@Html.AntiForgeryToken()` line in the view to the `[ValidateAntiForgeryToken]` attribute in the controller action.

3.5.3 Ghi đè CSS mặc định của Bootstrap

- Mở website bằng trình duyệt **Chorme**, nhấn chuột phải vào vùng muốn thay đổi CSS, chọn **Inspect**



- Điều chỉnh các thành phần CSS trên giao diện phù hợp theo mong muốn
- Copy nội dung CSS đã điều chỉnh, mở file **Contents/Site.css** hoặc tạo file CSS mới để ghi đè các giá trị CSS mặc định của Bootstrap

- File Contents/Site.css

```

Site.css  X
/*
Override the default bootstrap Nav bar by Ánh Nguyễn*/
.navbar-inverse {
    background-color: #d41d50;
    border-color: #fbf6f6;
}
.navbar-inverse .navbar-nav > li > a {
    color: white;
}
.navbar-inverse .navbar-brand {
    color: white;
}

```

- Chạy lại ứng dụng để xem sự thay đổi giao diện

Điều chỉnh giao diện trang khung của website (Layout)

- File Views/Shared/_LoginPartial.cshtml

```

using (Html.BeginForm("LogOff", "Account", FormMethod.Post, new { id = "logoutForm", @class = "navbar-right" }))
{
    @Html.AntiForgeryToken()

    <ul class="nav navbar-nav navbar-right">
        <li class="dropdown">
            <a href="#" class="dropdown-toggle" data-toggle="dropdown" role="button" aria-haspopup="true" aria-expanded="false">@User.Identity.GetUserName() <span class="caret"></span></a>
            <ul class="dropdown-menu">
                <li><a href="#">Course I'm Going</a></li>
                <li><a href="#">Lecture I'm Flowing</a></li>
                <li role="separator" class="divider"></li>
                <li><a href="javascript:document.getElementById('logoutForm').submit()">Log off</a></li>
            </ul>
        </li>
    </ul>
}

```

Tham khảo lấy code tạo dropdown list từ trang Document của bootstrap

- Kết quả thanh menu



3.5.4 Thêm mới thuộc tính trong Asp.Net Identity Users

Các thuộc tính mặc định của bảng ASPNETUSERs (Bảng này chưa thông tin người dùng trong hệ thống như Giảng viên hoặc Học viên)

Column Name	Type	Properties
<code>Id</code>	<code>nvarchar(128)</code>	<code>PK, not null</code>
<code>Email</code>	<code>nvarchar(256)</code>	<code>null</code>
<code>EmailConfirmed</code>	<code>bit</code>	<code>not null</code>
<code>PasswordHash</code>	<code>nvarchar(max)</code>	<code>null</code>
<code>SecurityStamp</code>	<code>nvarchar(max)</code>	<code>null</code>
<code>PhoneNumber</code>	<code>nvarchar(max)</code>	<code>null</code>
<code>PhoneNumberConfirmed</code>	<code>bit</code>	<code>not null</code>
<code>TwoFactorEnabled</code>	<code>bit</code>	<code>not null</code>
<code>LockoutEndDateUtc</code>	<code>datetime</code>	<code>null</code>
<code>LockoutEnabled</code>	<code>bit</code>	<code>not null</code>
<code>AccessFailedCount</code>	<code>int</code>	<code>not null</code>
<code>UserName</code>	<code>nvarchar(256)</code>	<code>not null</code>

Bổ sung thêm thuộc tính Name (họ tên) vào bảng

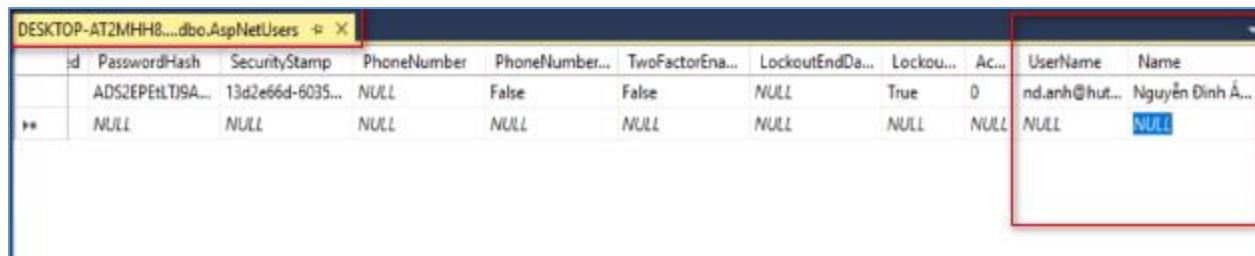
- File Models/ApplicationUser.cs

```

public class ApplicationUser : IdentityUser
{
    [Required]
    [StringLength(255)]
    public string Name { get; set; }
    public async Task<ClaimsIdentity> GenerateUserIdentityAsync(UserManager<ApplicationUser> manager)
    {
        // Note the authenticationType must match the one defined in CookieAuthenticationOptions.AuthenticationType
        var userIdentity = await manager.CreateIdentityAsync(this, DefaultAuthenticationType);
        // Add custom user claims here
        return userIdentity;
    }
}

```

- Mở Package Control Manager chạy lệnh
 - add-migration AddNameColumnTo ApplicationUser
 - update-database
- Kết quả, một cột mới được thêm vào CSDL. Nhập giá trị cho cột **Name** (ex: Nguyễn Đình Ánh)



ID	PasswordHash	SecurityStamp	PhoneNumber	PhoneNumberConfirmed	TwoFactorEnabled	LockoutEndDate	LockoutEnabled	AccessFailedCount	UserName	Name
ADS2EPELTJ9A...	13d2e66d-6035...	NULL	False	False	NULL	True	0	nd.anh@hut...	Nguyễn Đình Á...	NULL
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

3.5.5 Hiển thị danh sách các khóa hoặc sắp diễn ra tại trang chủ

- File Controllers/HomeController.cs

```
public class HomeController : Controller
{
    private ApplicationDbContext _dbContext;

    public HomeController()
    {
        _dbContext = new ApplicationDbContext();
    }

    public ActionResult Index()
    {
        var upcommingCourses = _dbContext.Courses
            .Include(c => c.Lecturer)
            .Include(c => c.Category)
            .Where(c => c.DateTime > DateTime.Now);

        return View(upcommingCourses);
    }
}
```

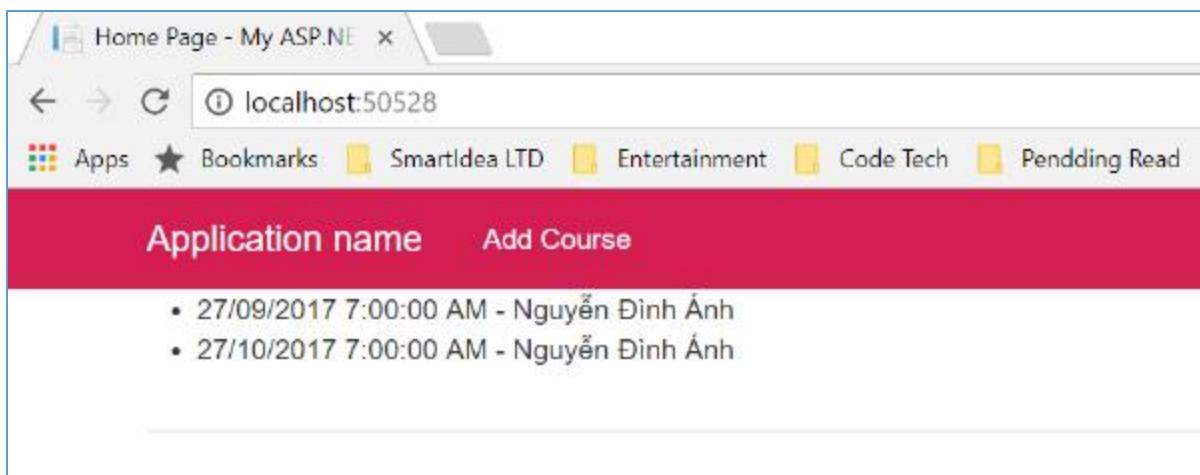
- File Views/Home/Index.cshtml

```
@model IEnumerable<BigSchool.Models.Course>
@{
    ViewBag.Title = "Home Page";
}


@foreach (var course in Model)
{
    <li>
        @course.DateTime - @course.Lecturer.Name
    </li>
}


```

- Chạy ứng dụng web, xem kết quả trang chủ (<http://localhost:50528/>)



3.5.6 Bổ sung thuộc tính Name vào màn hình đăng ký, cho phép người dùng nhập tên (Form Sign up)

- File Models/AccountViewModels.cs, Class RegisterViewModel

```
public class RegisterViewModel
{
    [Required]
    [EmailAddress]
    [Display(Name = "Email")]
    public string Email { get; set; }

    [Required]
    [StringLength(100, ErrorMessage = "The {0} must be at :")]
    [DataType(DataType.Password)]
    [Display(Name = "Password")]
    public string Password { get; set; }

    [DataType(DataType.Password)]
    [Display(Name = "Confirm password")]
    [Compare("Password", ErrorMessage = "The password and :")]
    public string ConfirmPassword { get; set; }

    [Required]
    [StringLength(255)]
    public string Name { get; set; }
}
```

- File Views/Account/Register.cshtml

```
@using (Html.BeginForm("Register", "Account", FormMethod.Post, new { @class = "form-hori" })
{
    @Html.AntiForgeryToken()
    <h4>Create a new account.</h4>
    <hr />
    @Html.ValidationSummary("", new { @class = "text-danger" })
    <div class="form-group">
        @Html.LabelFor(m => m.Email, new { @class = "col-md-2 control-label" })
        <div class="col-md-10">
            @Html.TextBoxFor(m => m.Email, new { @class = "form-control" })
        </div>
    </div>
    <div class="form-group">
        @Html.LabelFor(m => m.Name, new { @class = "col-md-2 control-label" })
        <div class="col-md-10">
            @Html.TextBoxFor(m => m.Name, new { @class = "form-control" })
        </div>
    </div>
    <div class="form-group">
        @Html.LabelFor(m => m.Password, new { @class = "col-md-2 control-label" })
        <div class="col-md-10">
            @Html.PasswordFor(m => m.Password, new { @class = "form-control" })
        </div>
    </div>
}
```

- File Controllers/AccountController.cs

```
// POST: /Account/Register
[HttpPost]
[AllowAnonymous]
[ValidateAntiForgeryToken]
public async Task<ActionResult> Register(RegisterViewModel model)
{
    if (ModelState.IsValid)
    {
        var user = new ApplicationUser { UserName = model.Email, Email = model.Email, Name = model.Name };
        var result = await UserManager.CreateAsync(user, model.Password);
        if (result.Succeeded)
        {
            await SignInManager.SignInAsync(user, isPersistent:false, rememberBrowser:false);
        }
    }
}
```

- Chạy ứng dụng web, trang đăng nhập để xem kết quả
(<http://localhost:50528/Account/Register>)

The screenshot shows a registration form titled 'Register.' with the sub-instruction 'Create a new account.' Below the title. The form fields are as follows:

- Email: dinhanhvnn@gmail.com
- Name: NGUYEN DINH ANH (highlighted with a red border)
- Password: (redacted)
- Confirm password: (redacted)
- Register button

3.5.7 Chính sửa giao diện trang chủ: danh sách khóa học sắp diễn ra

- File Views/Home/index.cshtml

```



    @foreach (var course in Model)
    {
        <li>
            <div class="date">
                <div class="month">
                    @course.DateTime.ToString("MM")
                </div>
                <div class="day">
                    @course.DateTime.ToString("dd")
                </div>
            </div>
            <div class="details">
                <span class="lecturer">
                    @course.Lecturer.Name
                </span>
                <span class="category">
                    @course.Category.Name
                </span>
            </div>
        </li>
    }
}

```

- File Content/Site.css

```
/*Course layout*/
.course {
    list-style: none;
}
.course > li {
    position: relative;
    margin-bottom: 30px;
}
.course > li .date {
    text-align: center;
    background-color: #d41d50;
    color: white;
    width: 60px;
    border-radius: 8px;
}
.course > li .details {
    position: absolute;
    top: 0;
    left: 70px;
}
```

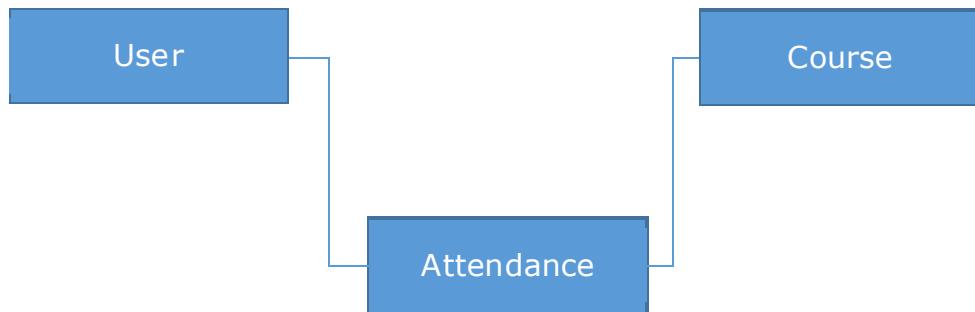
- Chạy ứng dụng web để xem kết quả (trang chủ)

The screenshot shows a list of courses on a web page. Each course item has a red rounded rectangle containing the date (e.g., 09/27, 10/27, 09/29, 10/23, 05/03) followed by the student's name and course name.

Date	Name	Course
09/27	Nguyễn Đình Ánh	Development
10/27	Nguyễn Đình Ánh	Business
09/29	NGUYEN DINH ANH	Business
10/23	NGUYEN DINH ANH	Business
05/03	NGUYEN DINH ANH	Business

3.5.8 Use-case: Đăng ký tham dự khóa học

Quan hệ nhiều – nhiều giữa hai bảng **User** và **Course**, Tạo bảng mới **Attendance** (Tham dự)



- Tạo mới File Models/Attendance.cs

```

public Course Course { get; set; }

[Key]
[Column(Order = 1)]
public int CourseId { get; set; }

public ApplicationUser Attendee { get; set; }

[Key]
[Column(Order = 2)]
public string AttendeeId { get; set; }
  
```

- Sửa Models/ApplicationDbContext.cs

```

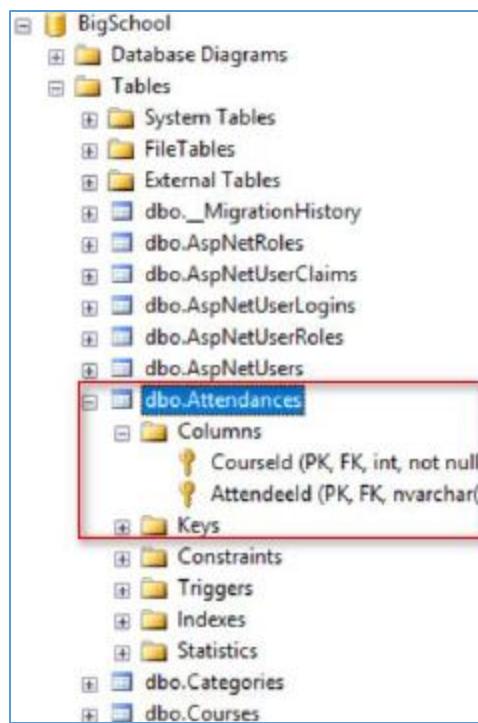
public class ApplicationDbContext : IdentityDbContext<ApplicationUser>
{
    public DbSet<Course> Courses { get; set; }
    public DbSet<Category> Categories { get; set; }
    public DbSet<Attendance> Attendances { get; set; }

    public ApplicationDbContext()
        : base("DefaultConnection", throwIfV1Schema: false)
    {
    }

    public static ApplicationDbContext Create()
    {
        return new ApplicationDbContext();
    }

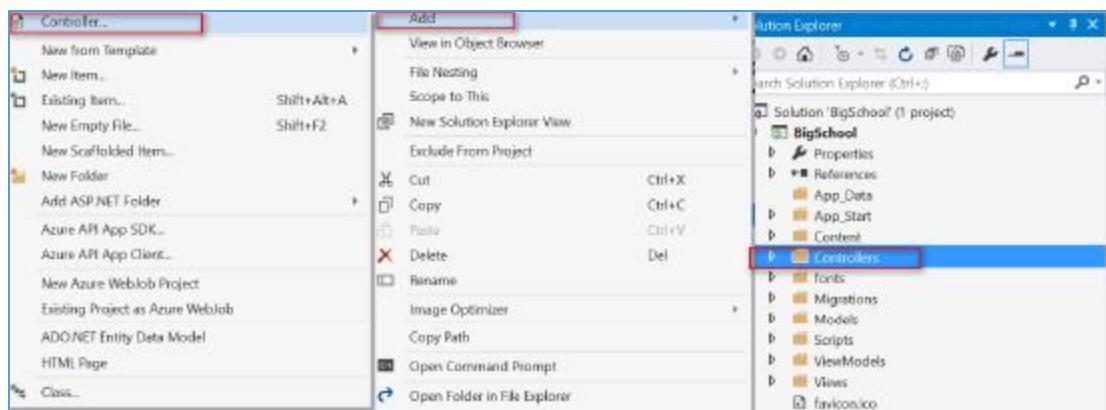
    protected override void OnModelCreating(DbModelBuilder modelBuilder)
    {
        modelBuilder.Entity<Attendance>()
            .HasRequired(a => a.Course)
            .WithMany()
            .WillCascadeOnDelete(false);
        base.OnModelCreating(modelBuilder);
    }
}
  
```

- Mở Package Manager Console chạy lệnh
 - Add - Migration AddAttendance
 - Update – database



Chức năng Học viên đăng ký tham dự khóa học (sử dụng Ajax & Restful API để không phải tải lại trang)

- Thêm mới File **Controllers/AttendancesController.cs** → Loại template: Web API 2 Controller



- Chọn Web API 2 Controller - Empty, đặt tên **AttendancesController**
- File Readme.txt hướng dẫn Enable Web API trong project

```

AttendanceController.cs readme.txt X
Visual Studio has added the full set of dependencies for ASP.NET Web API 2 to project 'BigSchool'.
The Global.asax.cs file in the project may require additional changes to enable ASP.NET Web API.

1. Add the following namespace references:
using System.Web.Http;
using System.Web.Routing;

2. If the code does not already define an Application_Start method, add the following method:
protected void Application_Start()
{
}

3. Add the following lines to the beginning of the Application_Start method:
GlobalConfiguration.Configure(WebApiConfig.Register);

```

- Mở File **Global.asax**, copy đoạn code phía trên đưa vào với nội dung

```

public class MvcApplication : System.Web.HttpApplication
{
    protected void Application_Start()
    {
        GlobalConfiguration.Configure(WebApiConfig.Register);
        AreaRegistration.RegisterAllAreas();
        FilterConfig.RegisterGlobalFilters(GlobalFilters.Filters);
        RouteConfig.RegisterRoutes(RouteTable.Routes);
        BundleConfig.RegisterBundles(BundleTable.Bundles);
    }
}

```

- File Controllers/AttendancesController.cs

```

[Authorize]
public class AttendancesController : ApiController
{
    private ApplicationDbContext _dbContext;

    public AttendancesController()
    {
        _dbContext = new ApplicationDbContext();
    }

    [HttpPost]
    public IHttpActionResult Attend([FromBody] int courseId)
    {
        var attendance = new Attendance
        {
            CourseId = courseId,
            AttendeeId = User.Identity.GetUserId()
        };

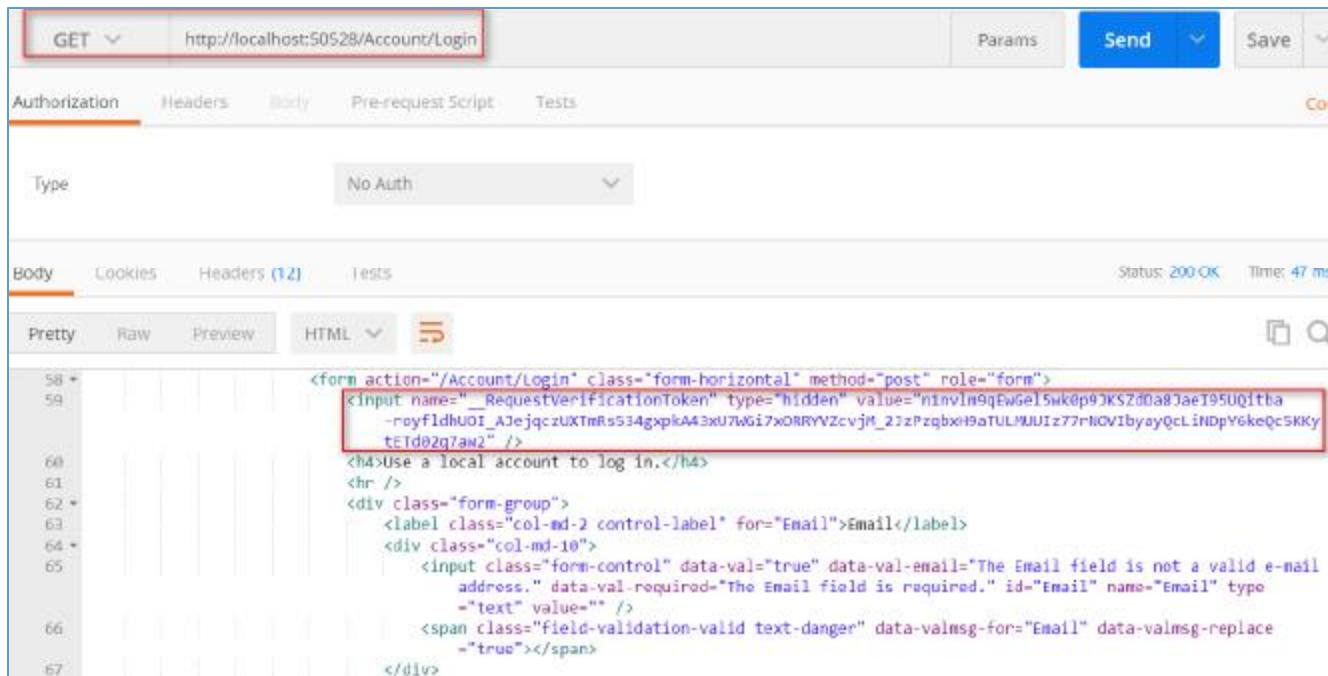
        _dbContext.Attendances.Add(attendance);
        _dbContext.SaveChanges();

        return Ok();
    }
}

```

Sử dụng công cụ Postman để Test API

- Link tải Postman (<https://www.getpostman.com/>)
- Lấy thông tin đăng nhập tài khoản, nhấn nút SEND



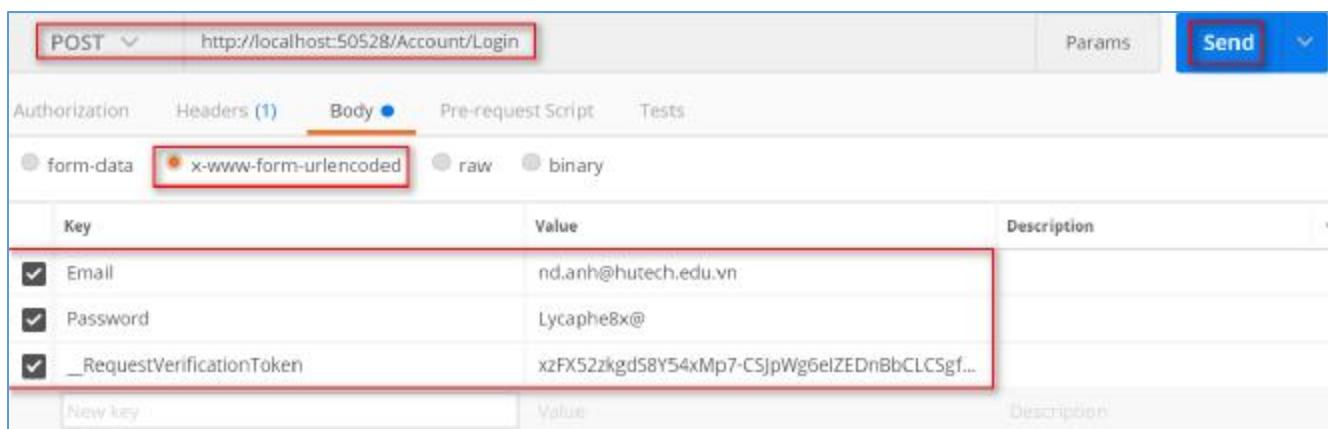
The screenshot shows the Postman interface with a GET request to `http://localhost:50528/Account/Login`. The response body contains the HTML source code of a login page. A specific line of code containing a long, randomly generated string is highlighted with a red box:

```

58 <input name="__RequestVerificationToken" type="hidden" value="ninvlm9qfTwGelsWk0p9JKS2dDa8JaeI95UQitha
59 -royfldhUOI_AjejqczUXtMRs534gxplA43xU7wGi7xORRYVZcvjM_2JzPzqbxH9aTULMUUiZ77rNOVibyayQcLiNDpy6keQc5KKY
60 t1d02q7am2" />
61
62 <h4>Use a local account to log in.</h4>
63 <hr />
64 <div class="form-group">
65   <label class="col-md-2 control-label" for="Email">Email</label>
66   <div class="col-md-10">
67     <input class="form-control" data-val="true" data-val-email="The Email field is not a valid e-mail
address." data-val-required="The Email field is required." id="Email" name="Email" type
="text" value="" />
     <span class="field-validation-valid text-danger" data-valmsg-for="Email" data-valmsg-replace
     = "true"></span>
   </div>

```

- Lấy thông tin bảo mật Cross-Site Request Forgery (CSRF) (một phần của ASP.NET Identity để bảo mật, phòng tránh lỗi CSRF), nhấn nút SEND



The screenshot shows the Postman interface with a POST request to `http://localhost:50528/Account/Login`. The Body tab is selected, showing form-data with three fields: `Email`, `Password`, and `__RequestVerificationToken`. The `__RequestVerificationToken` field is highlighted with a red box.

Key	Value	Description
Email	nd.anh@hutech.edu.vn	
Password	Lycaphe8x@	
__RequestVerificationToken	x2FX52zkgd58Y54xMp7-CSjpWg6elZEDnBbCLC5gf...	

- Mở Postman và nhập các thông tin

POST http://localhost:50528/api/Attendances

Authorization (Basic Auth)

Type: Basic Auth

Username: nd.anh@hutech.edu.vn

Password: *****

Headers (1): Content-Type: application/json

Body (raw JSON): { "AttendanceId": 1, "CourseId": 1 }

- Nhấn nút **SEND** và xem kết quả kiểm tra test API khi thêm một dòng đăng ký mới vào CSDL

SQLQuery1.sql - DES...BigSchool (sa (54))*

SELECT * FROM Attendances

Results

	CourseId	AttendeeId
1	1	e7ea7086-db4b-4a23-adf4-9450ec2997cb

- Tại màn hình **Postman** trên, nếu nhấn **SEND** thêm một lần nữa, sẽ có hai dòng dữ liệu giống nhau được chèn vào **CSDL**. Như vậy, sẽ có lỗi xảy ra

POST http://localhost:50528/api/Attendances

Params

Send

Authorization (Basic Auth): nd.anh@hutech.edu.vn

Body (raw JSON): { "AttendanceId": 1, "CourseId": 1 }

Status: 500 Internal Server Error

Body (Pretty):

```

    .SaveChangesInternal(SaveOptions options, Boolean executeInExistingTransaction)\r\n      at System.Data.Entity.Core.ObjectContext.SaveChanges(SaveOptions options)\r\n      at System.Data.Entity.Internal.InternalContext.SaveChanges()\r\n
11+     "InnerException": {
12       "Message": "An error has occurred.",
13       "ExceptionMessage": "Violation of PRIMARY KEY constraint 'PK_dbo.Attendances'. Cannot insert duplicate key in object 'Attendances'. The duplicate key value is (1, e7ea7086-db4b-4a23-adf4-9450ec2997cb). The statement has been ter

```

- ❖ Xử lý lỗi khi thêm hai dòng dữ liệu giống nhau vào CSDL
- File Controllers/AttendancesController.cs

```
[HttpPost]
public IHttpActionResult Attend([FromBody] int courseId)
{
    var userId = User.Identity.GetUserId();
    if (_dbContext.Attendances.Any(a => a.AttendeeId == userId && a.CourseId == courseId))
        return BadRequest("The Attendance already exists!");

    var attendance = new Attendance
    {
        CourseId = courseId,
        AttendeeId = userId
    };

    _dbContext.Attendances.Add(attendance);
    _dbContext.SaveChanges();

    return Ok();
}
```

- **Build** Ứng dụng và thử Test lại kết quả với **Postman**

The screenshot shows the Postman interface with a POST request to `http://localhost:50528/api/Attendances`. The response status is `400 Bad Request`, and the message is `"Message": "The Attendance already exists!"`.

- ❖ Thêm nút nhấn đăng ký tham gia khóa học

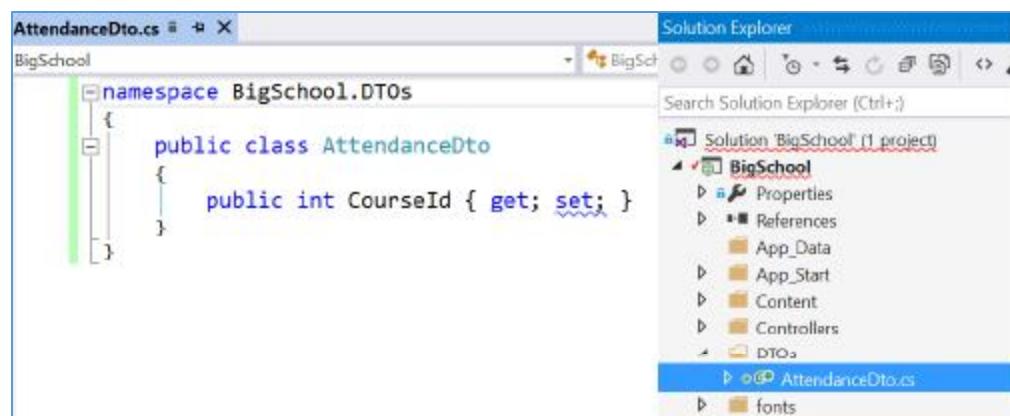
- File Views/Home/Index.cshtml

```

<div class="details">
    <span class="lecturer">
        @course.Lecturer.Name
    </span>
    <span class="category">
        @course.Category.Name
    </span>
    <button data-course-id="@course.Id" class="btn btn-default btn-sm js-toggle-attendance">Going?</button>
</div>
</li>
}
</ul>
@section scripts
{
    <script>
        $(document).ready(function () {
            $(".js-toggle-attendance").click(function (e) {
                var button = $(e.target);
                $.post("/api/attendances", { courseId: button.attr("data-course-id") })
                    .done(function() {
                        button
                            .removeClass("btn-default")
                            .addClass("btn-info")
                            .text("Going");
                    })
                    .fail(function() {
                        alert("Something failed!");
                    });
            });
        });
    </script>
}

```

- Tạo thư mục **DTOs** nằm tại thư mục gốc ứng dụng. Tạo file **DTOs/AttendanceDto.cs** (DTO: Data transfer object, đóng gói và trao đổi dữ liệu đối tượng giữa các tầng của ứng dụng)



- File Controllers/AttendancesController.cs

```
[HttpPost]
public IHttpActionResult Attend(AttendanceDto attendanceDto)
{
    var userId = User.Identity.GetUserId();
    if (_dbContext.Attendances.Any(a => a.AttendeeId == userId && a.CourseId == attendanceDto.CourseId))
        return BadRequest("The Attendance already exists!");

    var attendance = new Attendance
    {
        CourseId = attendanceDto.CourseId,
        AttendeeId = userId
    };

    _dbContext.Attendances.Add(attendance);
    _dbContext.SaveChanges();

    return Ok();
}
```

- Chạy ứng dụng và kiểm tra kết quả



3.5.9 Use-case theo dõi Giảng viên

- Tạo file Models/Following.cs

```
public class Following
{
    [Key]
    [Column(Order = 1)]
    public string FollowerId { get; set; }

    [Key]
    [Column(Order = 2)]
    public string FolloweeId { get; set; }

    public ApplicationUser Follower { get; set; }
    public ApplicationUser Followee { get; set; }
}
```

- File Models/ApplicationUser.cs

```
public class ApplicationUser : IdentityUser
{
    [Required]
    [StringLength(255)]
    public string Name { get; set; }

    public ICollection<Following> Followers { get; set; }
    public ICollection<Following> Followees { get; set; }

    public ApplicationUser()
    {
        Followers = new Collection<Following>();
        Followees = new Collection<Following>();
    }
}
```

- File Models/ApplicationDbContext.cs

```
public class ApplicationDbContext : IdentityDbContext<ApplicationUser>
{
    public DbSet<Course> Courses { get; set; }
    public DbSet<Category> Categories { get; set; }
    public DbSet<Attendance> Attendances { get; set; }
    public DbSet<Following> Followings { get; set; }

    public ApplicationDbContext()
        : base("DefaultConnection", throwIfV1Schema: false)
    {
    }

    public static ApplicationDbContext Create()
    {
        return new ApplicationDbContext();
    }

    protected override void OnModelCreating(DbModelBuilder modelBuilder)
    {
        modelBuilder.Entity<Attendance>()
            .IsRequired(a => a.Course)
            .WithMany()
            .WillCascadeOnDelete(false);

        modelBuilder.Entity< ApplicationUser >()
            .HasMany(u => u.Followers)
            .WithRequired(f => f.Followee)
            .WillCascadeOnDelete(false);

        modelBuilder.Entity< ApplicationUser >()
            .HasMany(u => u.Followees)
            .WithRequired(f => f.Follower)
            .WillCascadeOnDelete(false);
    }

    base.OnModelCreating(modelBuilder);
}
```

- Mở Package Manager Console
 - Add-migration AddFollowingsTable
 - Update-database
- File DTOs/FollowingDto.cs

```
public class FollowingDto
{
    public string FolloweeId { get; set; }
}
```

- File Controllers/FollowingsController.cs

```
public class FollowingsController : ApiController
{
    private readonly ApplicationDbContext _dbContext;

    public FollowingsController()
    {
        _dbContext = new ApplicationDbContext();
    }

    [HttpPost]
    public IHttpActionResult Follow(FollowingDto followingDto)
    {
        var userId = User.Identity.GetUserId();
        if (_dbContext.Followings.Any(f => f.FollowerId == userId && f.FolloweeId == followingDto.FolloweeId))
            return BadRequest("Following already exists!");

        var following = new Following
        {
            FollowerId = userId,
            FolloweeId = followingDto.FolloweeId
        };

        _dbContext.Followings.Add(following);
        _dbContext.SaveChanges();

        return Ok();
    }
}
```

- File Views/Home/Index.cshtml

```

<div class="details">
    <span class="lecturer">
        @course.Lecturer.Name
        <button class="btn btn-default btn-sm js-toggle-follow" data-user-id="@course.LecturerId">Follow</button>
    </span>
    <span class="category">
        @course.Category.Name
    </span>
    <button data-course-id="@course.Id" class="btn btn-default btn-sm js-toggle-attendance">Going?</button>
</div>
</li>
</ul>
@section scripts
{
    <script>
        $(document).ready(function () {
            $(".js-toggle-attendance").click(function (e) {
                var button = $(e.target);
                $.post("/api/attendances", { courseId: button.attr("data-course-id") })
                    .done(function () {
                        button
                            .removeClass("btn-default")
                            .addClass("btn-info")
                            .text("Going");
                    })
                    .fail(function () {
                        alert("Something failed!");
                    });
            });
            $(".js-toggle-follow").click(function (e) {
                var button = $(e.target);
                $.post("/api/followings", { followeeId: button.attr("data-user-id") })
                    .done(function () {
                        button.text("Following");
                    })
                    .fail(function () {
                        alert("Something failed");
                    });
            });
        });
    </script>
}

```

- Chạy ứng dụng và kiểm tra kết quả



- ❖ Nhấn nút F12 để kiểm tra các hành động Follow và Going? Nếu người dùng chưa đăng nhập

- Tạo File ViewModels/CoursesViewModel.cs

```
public class CoursesViewModel
{
    public IEnumerable<Course> UpcommingCourses { get; set; }
    public bool ShowAction { get; set; }
}
```

- File Controllers/HomeController.cs

```
public ActionResult Index()
{
    var upcommingCourses = _dbContext.Courses
        .Include(c => c.Lecturer)
        .Include(c => c.Category)
        .Where(c => c.DateTime > DateTime.Now);

    var viewModel = new CoursesViewModel
    {
        UpcommingCourses = upcommingCourses,
        ShowAction = User.Identity.IsAuthenticated
    };

    return View(viewModel);
}
```

- File Views/Home/Index.cshtml

```
@model BigSchool.ViewModels.CoursesViewModel
@{
    ViewBag.Title = "Home Page";
}




@foreach (var course in Model.UpcommingCourses)
{
    <li>...</li>
}



@course.Lecturer.Name
    @if (Model.ShowAction)
    {
        <button class="btn btn-link btn-sm js-toggle-button">
            ...
        </button>
    }


    @course.Category.Name
    @if (Model.ShowAction)
    {
        <button data-course-id="@course.Id" class="btn btn-link btn-sm js-toggle-button">
            ...
        </button>
    }


```

- Chạy ứng dụng và kiểm tra kết quả

Application name	Add Course	Register	Log in
10 27	Nguyễn Đình Ánh Business		
05 03	Ánh Nguyễn Business		

3.5.10 Xem danh sách khóa học đăng ký tham dự

- File Controllers/CoursesController.cs

```
[Authorize]
public ActionResult Attending()
{
    var userId = User.Identity.GetUserId();

    var courses = _dbContext.Attendances
        .Where(a => a.AttendeeId == userId)
        .Select(a => a.Course)
        .Include(l => l.Lecturer)
        .Include(l => l.Category)
        .ToList();

    var viewModel = new CoursesViewModel
    {
        UpcommingCourses = courses,
        ShowAction = User.Identity.IsAuthenticated
    };

    return View(viewModel);
}
```

- File Views/Courses/Attending.cshtml (Copy nội dung hiển thị giống như trang Views/Home/Index.cshtml)

```
@model BigSchool.ViewModels.CoursesViewModel
@{
    ViewBag.Title = "Attending";
    Layout = "~/Views/Shared/_Layout.cshtml";
}



## Attending




@foreach (var course in Model.UpcommingCourses)
{
    <li>...</li>
}

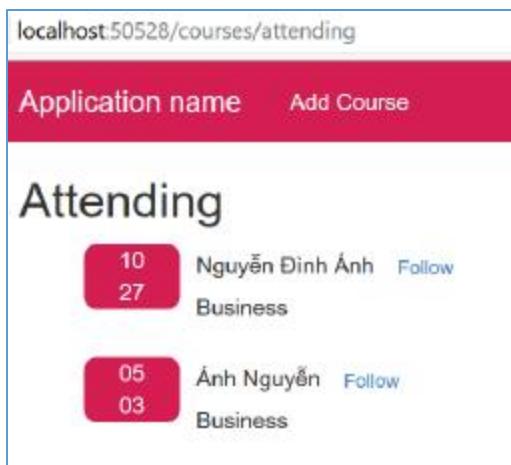
```

- File Views/Shared/_LoginPartial.cshtml

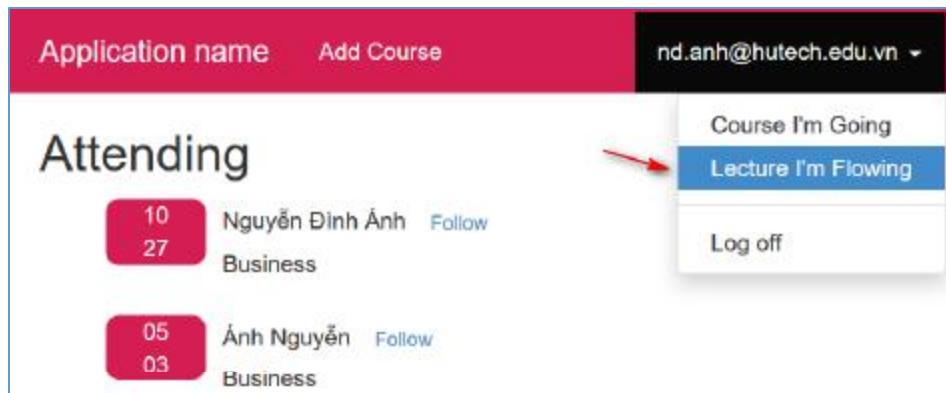
```
using (Html.BeginForm("LogOff", "Account", FormMethod.Post, new { id = "logoutForm", @class = "navbar-right" }))
{
    @Html.AntiForgeryToken()

    <ul class="nav navbar-nav navbar-right">
        <li class="dropdown">
            <a href="#" class="dropdown-toggle" data-toggle="dropdown" role="button" aria-haspopup="true" aria-expanded="false">@User.Identity.GetUserName() <span class="caret"></span></a>
            <ul class="dropdown-menu">
                <li><a href="@Url.Action("Attending", "Courses")">Course I'm Going</a></li>
                <li><a href="#">Lecture I'm Flowing</a></li>
                <li role="separator" class="divider"></li>
                <li><a href="javascript:document.getElementById('logoutForm').submit()">Logout</a></li>
            </ul>
        </li>
    </ul>
}
```

- Chạy ứng dụng và kiểm tra kết quả

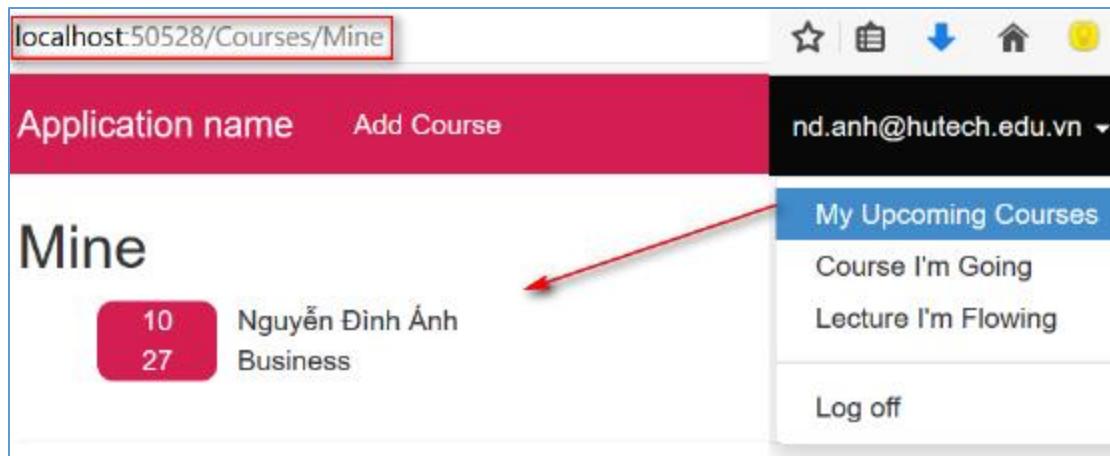


Bài tập: Thực hiện chức năng xem danh sách giảng viên đang theo dõi



Bài tập: Lần đầu danh sách được tải, đổi màu nút **Follow, Going** nếu giảng viên hoặc khóa học đã theo dõi hoặc tham gia. Cho phép thay đổi tình trạng bỏ theo dõi giảng viên, bỏ tham gia đăng ký khóa học bất kỳ.

Bài tập: Xem danh sách các khóa học được tạo bởi tài khoản người dùng hiện tại



- File Controllers/CoursesController.cs

```
[Authorize]
public ActionResult Mine()
{
    var userId = User.Identity.GetUserId();
    var courses = _dbContext.Courses
        .Where(c => c.LecturerId == userId && c.DateTime > DateTime.Now)
        .Include(l=>l.Lecturer)
        .Include(c=>c.Category)
        .ToList();

    return View(courses);
}
```

- File Views/Courses/Mine.cshtml

```
@model IEnumerable<BigSchool.Models.Course>
@{
    ViewBag.Title = "Mine";
    Layout = "~/Views/Shared/_Layout.cshtml";
}
<h2>Mine</h2>
<ul class="course">
    @foreach (var course in Model)
    {
        <li>
            <div class="date">
                <div class="month">
                    @course.DateTime.ToString("MM")
                </div>
                <div class="day">
                    @course.DateTime.ToString("dd")
                </div>
            </div>
            <div class="details">
                <span class="lecturer">
                    @course.Lecturer.Name
                </span>
                <span class="category">
                    @course.Category.Name
                </span>
            </div>
        </li>
    }
</ul>
```

Bài tập: Chức năng Tạo mới khóa học, sau hành động lưu thành công sẽ cho chuyển sang trang **My Upcomming Course**.

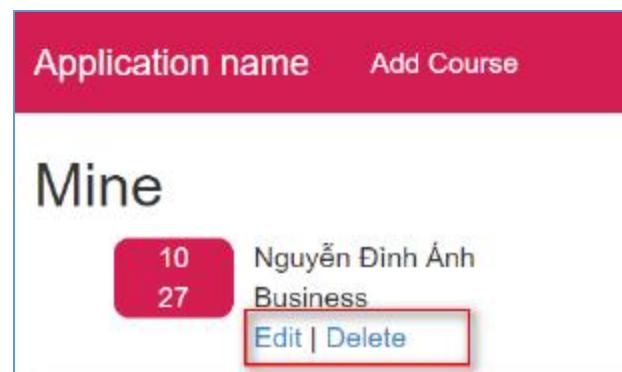
3.5.11 Use-case Quản lý Khóa học (Xóa, cập nhật)

- File Views/Courses/Mine.cshtml

```
<ul class="course">
    @foreach (var course in Model)
    {
        <li>
            <div class="date">...</div>
            <div class="details">...</div>
            <div class="action">
                <a href="">Edit</a> |
                <a href="">Delete</a>
            </div>
        </li>
    }
</ul>
```

- File Content/Site.css, thêm CSS

```
.course > li .action {
    display: none;
    left: 70px;
    position: absolute;
}
.course > li:hover .action {
    display: block;
}
```



- File Controllers/CoursesController.cs

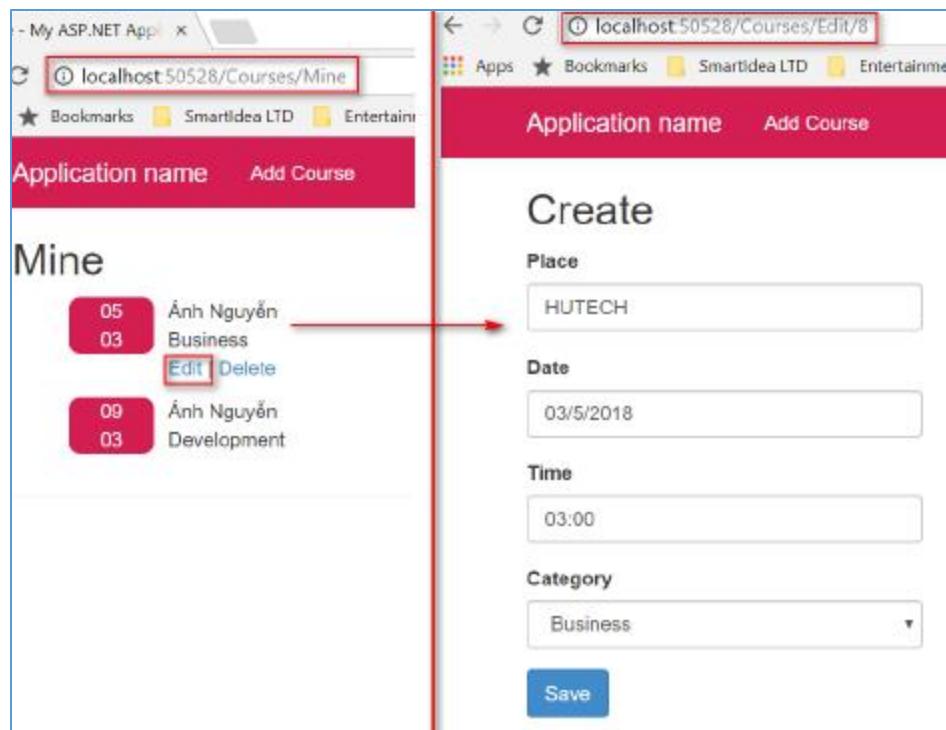
```
[Authorize]
public ActionResult Edit(int id)
{
    var userId = User.Identity.GetUserId();
    var course = _dbContext.Courses.Single(c => c.Id == id && c.LecturerId == userId);

    var viewModel = new CourseViewModel
    {
        Categories = _dbContext.Categories.ToList(),
        Date = course.DateTime.ToString("dd/M/yyyy"),
        Time = course.DateTime.ToString("HH:mm"),
        Category = course.CategoryId,
        Place = course.Place
    };
    return View("Create", viewModel);
}
```

- File Views/Courses/Mine.cshtml

```
@model IEnumerable<BigSchool.Models.Course>
@{
    ViewBag.Title = "Mine";
    Layout = "~/Views/Shared/_Layout.cshtml";
}
<h2>Mine</h2>
<ul class="course">
    @foreach (var course in Model)
    {
        <li>
            <>...</>
            <>...</>
            <div class="action">
                <a href="@Url.Action("Edit", "Courses", new { id = course.Id })">
                    Edit
                </a>
                |
                <a href="">Delete</a>
            </div>
        </li>
    }
</ul>
```

- Chạy ứng dụng, kiểm tra kết quả



- ❖ Update thông tin khóa học
- File ViewModels/CourseViewModel.cs

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

    [Required]
    public string Place { get; set; }

    [Required]
    [FutureDate]
    public string Date { get; set; }

    [Required]
    [ValidTime]
    public string Time { get; set; }

    [Required]
    public byte Category { get; set; }
    public IEnumerable<Category> Categories { get; set; }
    public string Heading { get; set; }
    public string Action
    {
        get { return (Id != 0) ? "Update" : "Create"; }
    }
}
```

- File Controllers/CoursesController.cs

```
public ActionResult Create()
{
    var viewModel = new CourseViewModel
    {
        Categories = _dbContext.Categories.ToList(),
        Heading = "Add Course"
    };
    return View(viewModel);
}

[Authorize]
public ActionResult Edit(int id)
{
    var userId = User.Identity.GetUserId();
    var course = _dbContext.Courses.Single(c => c.Id == id && c.LecturerId == userId);

    var viewModel = new CourseViewModel
    {
        Categories = _dbContext.Categories.ToList(),
        Date = course.DateTime.ToString("dd/M/yyyy"),
        Time = course.DateTime.ToString("HH:mm"),
        Category = course.CategoryId,
        Place = course.Place,
        Heading = "Edit Course",
        Id = course.Id
    };
    return View("Create", viewModel);
}

[Authorize]
[HttpPost]
[ValidateAntiForgeryToken]
public ActionResult Update(CourseViewModel viewModel)
{
    if (!ModelState.IsValid)
    {
        viewModel.Categories = _dbContext.Categories.ToList();
        return View("Create", viewModel);
    }
    var userId = User.Identity.GetUserId();
    var course = _dbContext.Courses.Single(c => c.Id == viewModel.Id && c.LecturerId == userId);

    course.Place = viewModel.Place;
    course.DateTime = viewModel.GetDateTime();
    course.CategoryId = viewModel.Category;

    _dbContext.SaveChanges();

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

- File Views/Courses/Create.cshtml

```
@model BigSchool.ViewModels.CourseViewModel
{
    ViewBag.Title = "Create";
    Layout = "~/Views/Shared/_Layout.cshtml";
}

<h2>@Model.Heading</h2>
@using (Html.BeginForm(Model.Action, "Courses"))
{
    @Html.AntiForgeryToken()
    @Html.HiddenFor(m=>m.Id)
    <div class="form-group">...</div>
    <div class="form-group">...</div>
    <div class="form-group">...</div>
    <div class="form-group">...</div>
    <button type="submit" value="Save" class="btn btn-primary">Save</button>
}
```

- Chạy ứng dụng và kiểm tra kết quả
- Đổi tên Views/Courses/Create.cshtml → Views/Courses/CourseForm.cshtml



```
// GET: Courses
[Authorize]
public ActionResult Create()
{
    var viewModel = new CourseViewModel
    {
        Categories = _dbContext.Categories.ToList(),
        Heading = "Add Course"
    };
    return View("CourseForm", viewModel);
}
```

*Tương tự cho các
method bên dưới
View --> CourseForm*

```
[Authorize]
[HttpPost]
[ValidateAntiForgeryToken]
public ActionResult Edit(int id){...}
```

```
[Authorize]
[HttpPost]
[ValidateAntiForgeryToken]
public ActionResult Update(CourseViewModel viewModel){...}
```

```
[Authorize]
[HttpPost]
[ValidateAntiForgeryToken]
public ActionResult Create(CourseViewModel viewModel){...}
```

❖ Xóa khóa học

Xóa tạm thông tin khóa học, khi cần có thể dễ dàng khôi phục

- File Models/Course.cs

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

    public bool IsCanceled { get; set; }  

    public ApplicationUser Lecturer { get; set; }
    [Required]
    public string LecturerId { get; set; }

    [Required]
    [StringLength(255)]
    public string Place { get; set; }
    public DateTime DateTime { get; set; }

    public Category Category { get; set; }
    [Required]
    public byte CategoryId { get; set; }
}
```

- Mở Package manager console

- Add-migration AddIsCanceledColumnToCourse

- Update-database

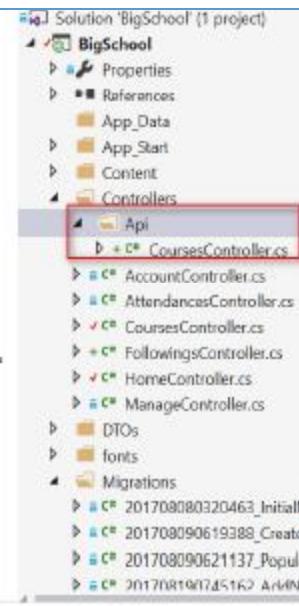
- Tạo File Controllers/Api/CoursesController.cs (Web API 2 Controller)

```
public class CoursesController : ApiController
{
    public ApplicationDbContext _dbContext { get; set; }

    public CoursesController()
    {
        _dbContext = new ApplicationDbContext();
    }

    [HttpDelete]
    public IHttpActionResult Cancel(int id)
    {
        var userId = User.Identity.GetUserId();
        var course = _dbContext.Courses.Single(c => c.Id == id && c.LecturerId == userId);
        if (course.IsCanceled)
            return NotFound();
        course.IsCanceled = true;
        _dbContext.SaveChanges();

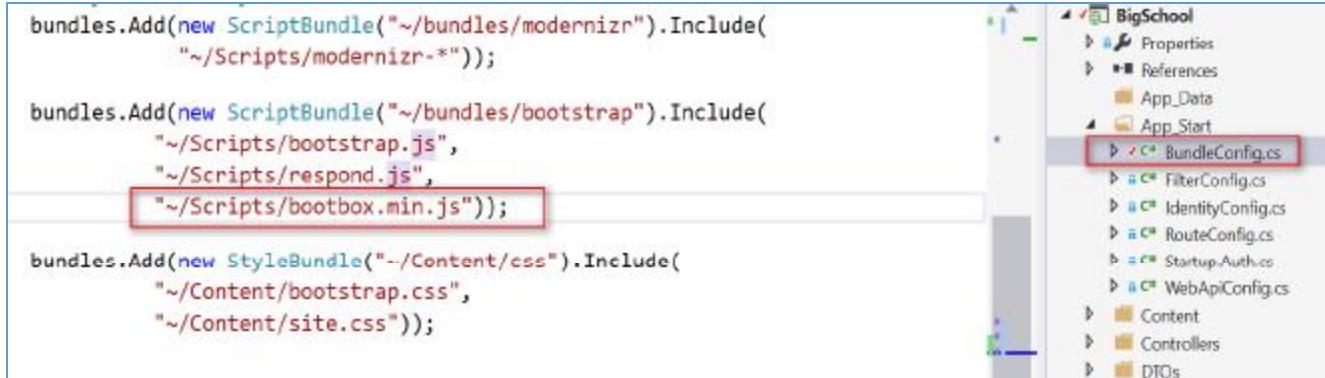
        return Ok();
    }
}
```



❖ Sử dụng bootstrap modals

Download thư viện Bootbox.js tại <http://bootboxjs.com/>, copy file thư viện bootbox.js vào thư mục Scripts

- File App_Start/BundleConfig.cs



```
bundles.Add(new ScriptBundle("~/bundles/modernizr").Include(
    "~/Scripts/modernizr-*"));

bundles.Add(new ScriptBundle("~/bundles/bootstrap").Include(
    "~/Scripts/bootstrap.js",
    "~/Scripts/respond.js",
    "~/Scripts/bootbox.min.js"));

bundles.Add(new StyleBundle("~/Content/css").Include(
    "~/Content/bootstrap.css",
    "~/Content/site.css"));
```

- File Views/Courses/Mine.cshtml

```
<div class="action">
    <a href="@Url.Action("Edit", "Courses", new { id = course.Id })">
        Edit
    </a>
    |
    <a href="#" class="js-cancel-course" data-course-id="@course.Id">
        Cancel
    </a>
</div>
</li>
}
</ul>
@section scripts
{
    <script>
        $(document).ready(function () {
            $(".js-cancel-course").click(function (e) {
                e.preventDefault();
                var link = $(e.target);
                bootbox.confirm({
                    title: "Cancel Course?",
                    message: "Are you sure to cancel?",
                    buttons: [
                        {
                            cancel: {
                                label: '<i class="fa fa-times"></i> Cancel'
                            },
                            confirm: {
                                label: '<i class="fa fa-check"></i> Confirm'
                            }
                        }
                    ],
                    callback: function (result) {
                        if (result) {
                            $.ajax({
                                url: "/api/courses/" + link.attr("data-course-id"),
                                method: "DELETE"
                            })
                            .done(function () {
                                link.parents("li").fadeOut(function () {
                                    $(this).remove();
                                });
                            })
                            .fail(function () {
                                alert("Something failed!");
                            });
                        }
                    }
                });
            });
        });
    </script>
}
```

- Chạy website và kiểm tra kết quả



localhost:50528/Courses/Mine

Application name Add Course

Mine

12 22	Nguyễn Thị Nhàn Development
----------	--------------------------------

© 2017 - My ASP.NET Application Ánh Nguy

Cancel Course?

Are you sure to cancel?

Bài tập: Chỉ hiển thị các khóa học chưa được hủy trong màn hình danh sách các khóa học (trang chủ và trang khóa học của tôi)

- ❖ Sử dụng bootstrap label hiển thị cảnh báo khóa học đã bị hủy trong màn hình theo dõi khóa học

localhost:50528/Courses/Attending

Application name Add Course 1Nd.anh@hutech.edu.vn

Attending

12 27	Đặng Thị Thúy Nga Follow Marketing
12 27	Nguyễn Đinh Ánh Follow Business

- Chính sửa Database

	Id	LecturerId	Place	DateTime	CategoryId	IsCanceled
	2	e7ea7086-db4b...	Khoa CNTT HU...	2017-12-27 07:00:00.000	2	True
	8	672eabee-7e35...	HUTECH ++	2017-12-27 07:00:00.000	3	True
	16	e7ea7086-db4b...	Khoa CNTT HU...	2017-12-27 07:00:00.000	2	True
	17	672eabee-7e35...	HCM	2017-12-27 07:00:00.000	1	True
	18	e93c5741-47e5...	HA NOI 1	2018-12-22 09:45:00.000	1	True

Attending.cshtml

```


    @course.Lecturer.Name

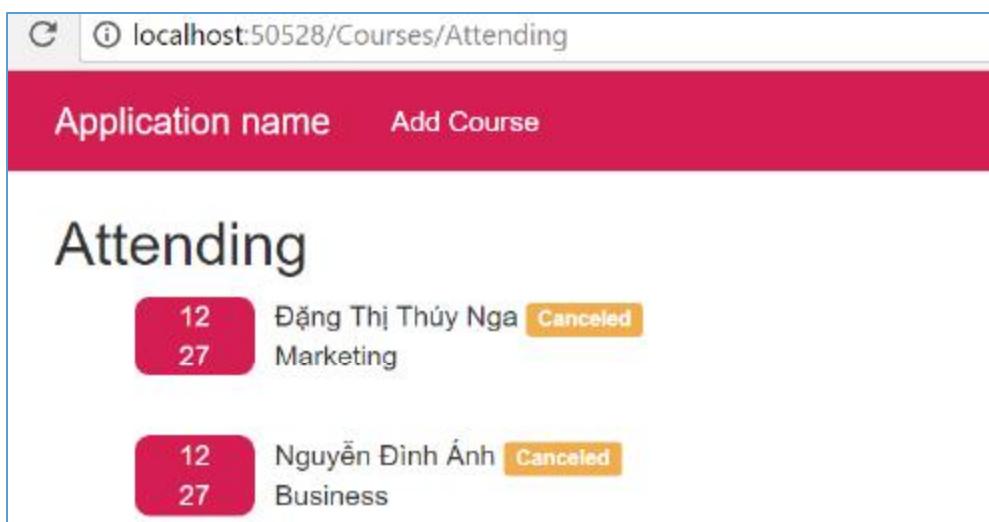

@if (course.IsCanceled)
{
    <span class="label label-warning">Canceled</span>
}

@if (Model.ShowAction && !course.IsCanceled)
{
    <button class="btn btn-link btn-sm js-toggle-follow" data-user-id="@course.LecturerId">Follow</button>
}
</span>

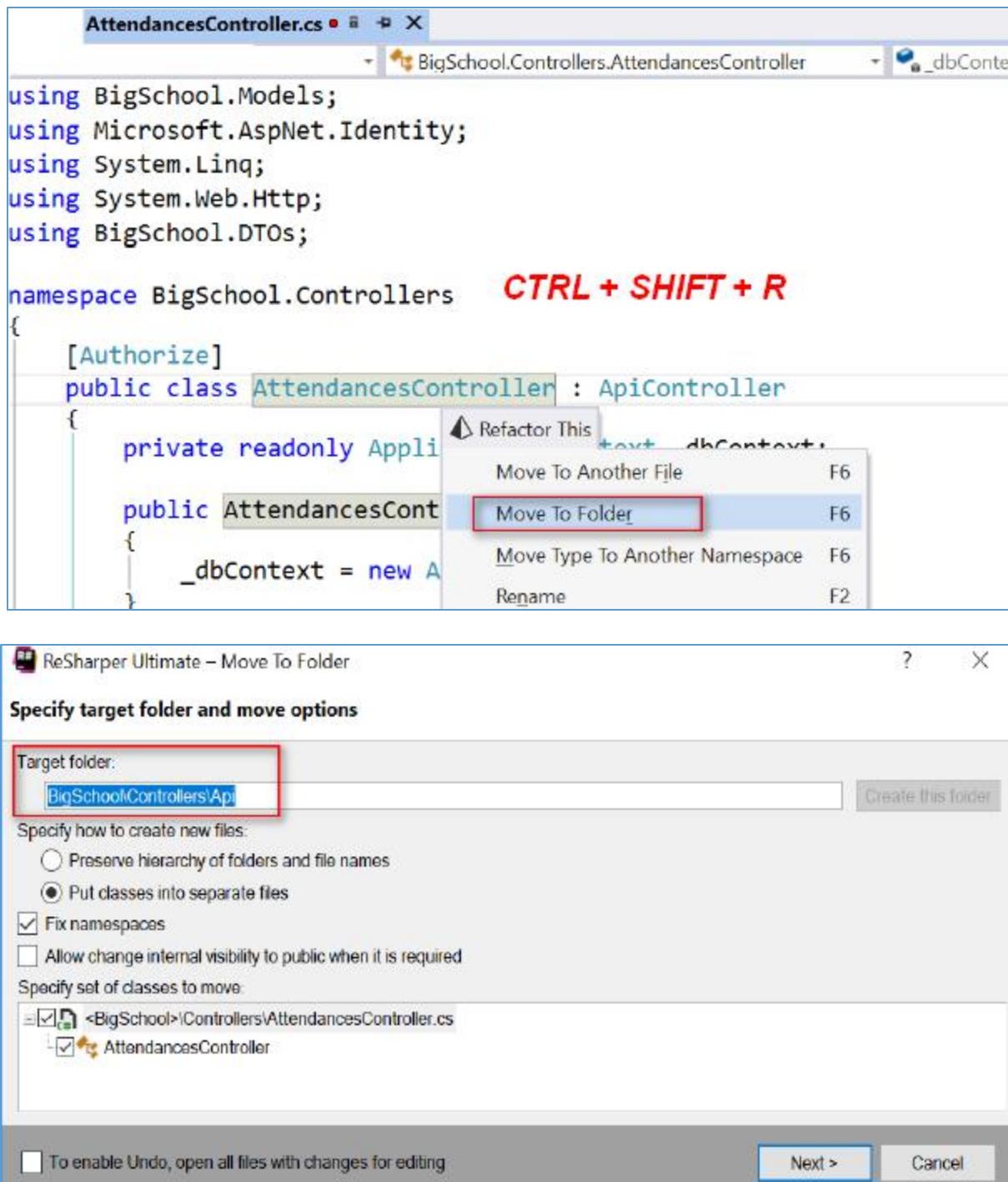
    @course.Category.Name

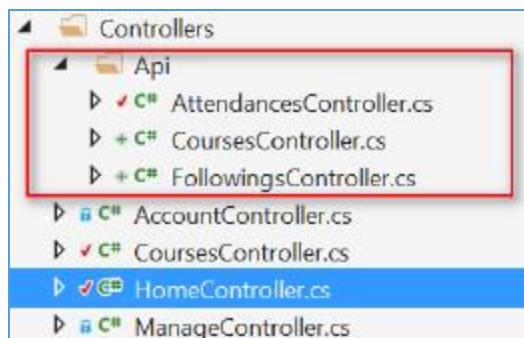

```

- Kết quả



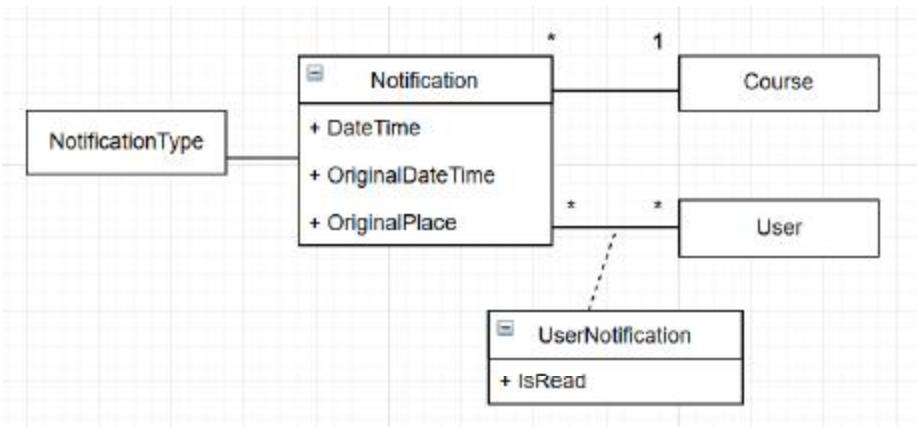
- ❖ Di chuyển các API vào thư mục API (lưu ý: nếu sử dụng Reshaper thì dùng tổ hợp phím ctrl + shift + r)





3.5.12 Use-case thông báo (notification)

Thông báo khi người dùng thay đổi thông tin khóa học bao gồm: Thêm, xóa, sửa khóa học



- Models/NotificationType.cs
- Models/Notification.cs
- Models/UserNotification.cs

```

public class UserNotification
{
    [Key]
    [Column(Order = 1)]
    public string UserId { get; set; }

    [Key]
    [Column(Order = 1)]
    public int NotificationId { get; set; }

    public ApplicationUser User { get; set; }
    public Notification Notification { get; set; }
}
  
```

```
public enum NotificationType
{
    CourseCanceled = 1,
    CourseUpdated = 2,
    CouorseCreated = 3
}
```

```
public class UserNotification
{
    [Key]
    [Column(Order = 1)]
    public string UserId { get; set; }

    [Key]
    [Column(Order = 2)]
    public int NotificationId { get; set; }

    public ApplicationUser User { get; set; }
    public Notification Notification { get; set; }

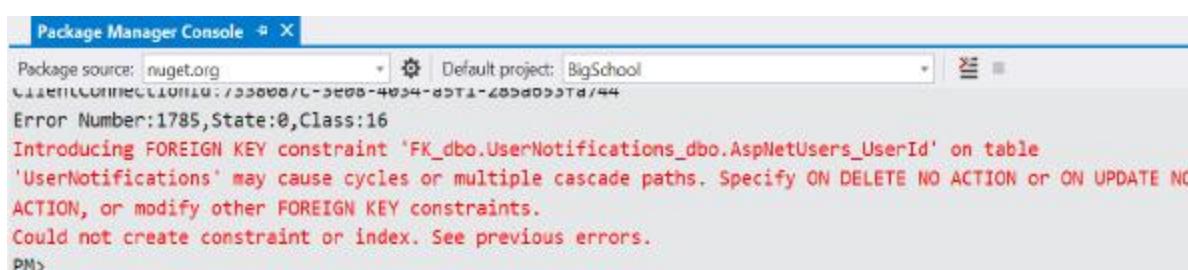
    public bool IsRead { get; set; }
}
```

- Cập nhật Database

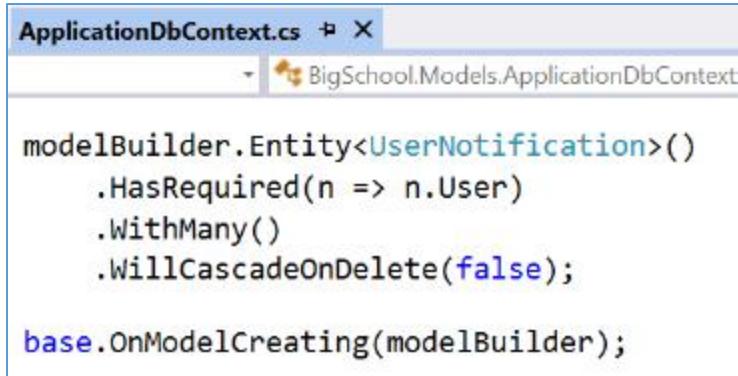
```
public class ApplicationDbContext : IdentityDbContext<ApplicationUser>
{
    public DbSet<Course> Courses { get; set; }
    public DbSet<Category> Categories { get; set; }
    public DbSet<Attendance> Attendances { get; set; }
    public DbSet<Following> Followings { get; set; }
    public DbSet<Notification> Notifications { get; set; }
    public DbSet<UserNotification> UserNotifications { get; set; }
}
```

Add-migration "AddNotification"

Update Database



- Models/ApplicationDbContext.cs



```

ApplicationDbContext.cs  ✘ X
BigSchool.Models.ApplicationDbContext

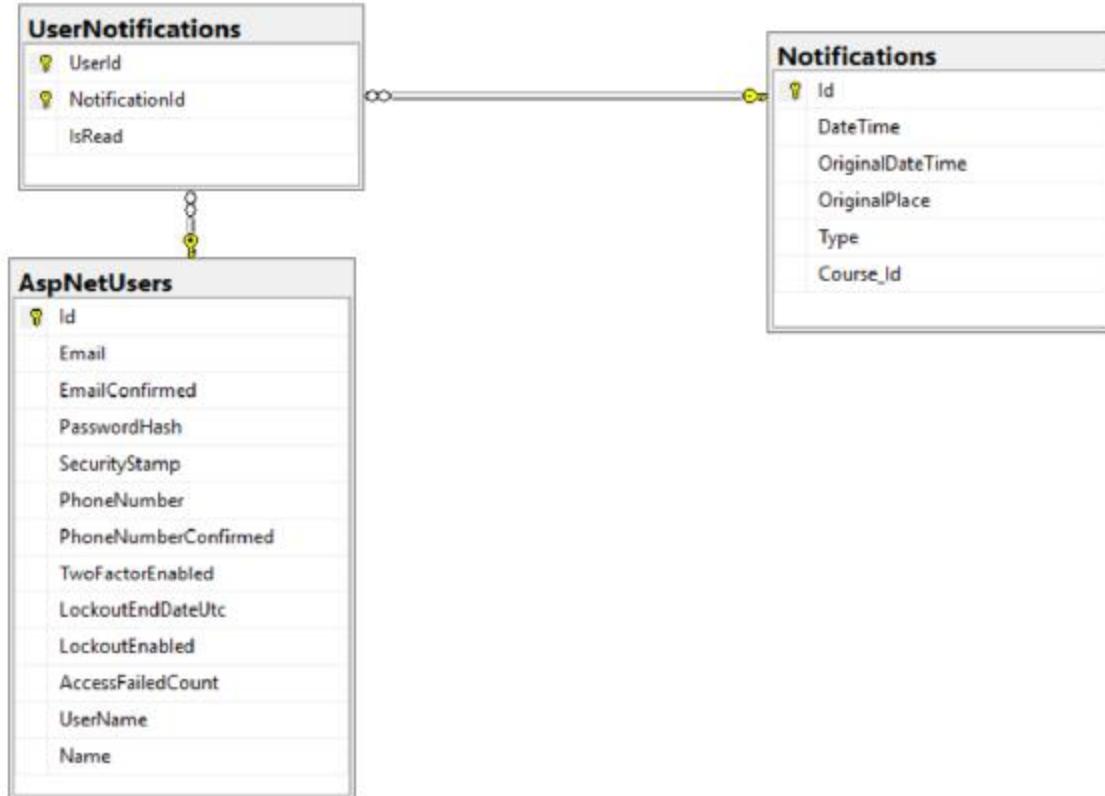
modelBuilder.Entity<UserNotification>()
    .HasRequired(n => n.User)
    .WithMany()
    .WillCascadeOnDelete(false);

base.OnModelCreating(modelBuilder);

```

Add-migration "AddNotification" -force

Update Database



- ❖ Hủy (cancel) khóa học (course) sẽ gửi thông báo cho các thành viên đã đăng ký tham dự khóa học đó
- Controllers/Api/CoursesController.cs

```
CoursesController.cs • X
BigSchool.Controllers.Api.CoursesController Cancel(int id)
[System.Web.Http.HttpDelete]
public IHttpActionResult Cancel(int id)
{
    var userId = User.Identity.GetUserId();
    var course = _dbContext.Courses.Single(c => c.Id == id
                                                && c.LecturerId == userId);
    if (course.IsCanceled)
        return NotFound();
    course.IsCanceled = true;

    //Add notification
    var notification = new Notification()
    {
        DateTime = DateTime.Now,
        Course = course,
        Type = NotificationType.CourseCanceled
    };

    var attendees = _dbContext.Attendances
        .Where(a => a.CourseId == course.Id)
        .Select(a => a.Attendee)
        .ToList();

    foreach (var attendee in attendees)
    {
        var userNotification = new UserNotification()
        {
            User = attendee,
            Notification = notification
        };
        _dbContext.UserNotifications.Add(userNotification);
    }

    _dbContext.SaveChanges();

    return Ok();
}
```

ĐÁP ÁN BÀI TẬP

- File Views/Home/Index.cshtml

```

@section scripts
{
    <script>
        $(document).ready(function () {
            $(".js-toggle-attendance").click(function (e) {
                var button = $(e.target);
                if (button.hasClass("btn-default")) {
                    $.post("/api/attendances", { courseId: button.attr("data-course-id") })
                        .done(function () {
                            button
                                .removeClass("btn-default")
                                .addClass("btn-info")
                                .text("Going");
                        })
                        .fail(function () {
                            alert("Something failed!");
                        });
                } else {
                    $.ajax({
                        url: "/api/attendances/" + button.attr("data-course-id"),
                        method: "DELETE"
                    })
                        .done(function () {
                            button
                                .removeClass("btn-info")
                                .addClass("btn-default")
                                .text("Going?");
                        })
                        .fail(function () {
                            alert("Something failed");
                        });
                }
            });
            $(".js-toggle-follow").click(function (e) {
                var button = $(e.target);
                $.post("/api/followings", { followeeId: button.attr("data-user-id") })
                    .done(function () {
                        button.text("Following");
                    })
                    .fail(function () {
                        alert("Something failed");
                    });
            });
        });
    </script>
}

```

Ghi Chú: Thực hiện tương tự cho chức năng theo dõi (follow)

- File Controllers/AttendancesController.cs

```
[HttpDelete]
public IHttpActionResult DeleteAttendance(int id)
{
    var userId = User.Identity.GetUserId();

    var attendance = _dbContext.Attendances
        .SingleOrDefault(a => a.AttendeeId == userId && a.CourseId == id);

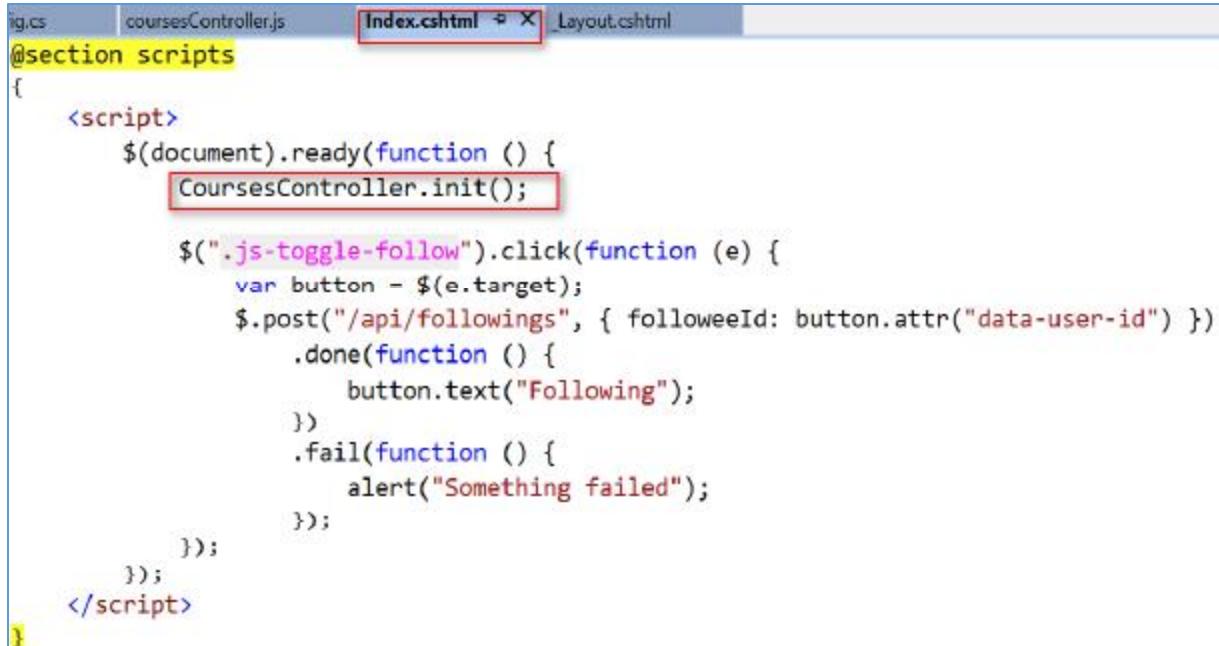
    if (attendance == null)
        return NotFound();

    _dbContext.Attendances.Remove(attendance);
    _dbContext.SaveChanges();

    return Ok(id);
}
```

3.5.13 Tách file javascript trong ứng dụng

- File Views/Home/Index.cshtml



```
@section scripts
{
    <script>
        $(document).ready(function () {
            CoursesController.init();

            $(".js-toggle-follow").click(function (e) {
                var button = $(e.target);
                $.post("/api/followings", { followeeId: button.attr("data-user-id") })
                    .done(function () {
                        button.text("Following");
                    })
                    .fail(function () {
                        alert("Something failed");
                    });
            });
        });
    </script>
}
```

- Tạo thư mục Scripts/app → Tạo file Script/app/coursesController.js

```
var CoursesController = function () {
    var button;
    var init = function () {
        $(".js-toggle-attendance").click(toggleAttendance);
    };
    var toggleAttendance = function (e) {
        button = $(e.target);
        if (button.hasClass("btn-default")) {
            createAttendance();
        }
        else {
            deleteAttendance();
        }
    };
    var createAttendance = function () {
        $.post("/api/attendances", { courseId: button.attr("data-course-id") })
            .done(done)
            .fail(fail);
    };
    var deleteAttendance = function () {
        $.ajax({
            url: "/api/attendances/" + button.attr("data-course-id"),
            method: "DELETE"
        })
        .done(done)
        .fail(fail);
    };
    var done = function () {
        var text = (button.text() == "Going") ? "Going?" : "Going";
        button.toggleClass("btn-info").toggleClass("btn-default").text(text);
    };
    var fail = function () {
        alert("Something failed");
    };
    return {
        init: init
    }
}();
```

- File App_start/BundleConfig.cs

```
public class BundleConfig
{
    // For more information on bundling, visit http://go.microsoft.com/fwlink/?LinkId=301862
    public static void RegisterBundles(BundleCollection bundles)
    {
        bundles.Add(new ScriptBundle("~/bundles/app").Include(
                    "~/scripts/app/coursesController.js"
                ));

        bundles.Add(new ScriptBundle("~/bundles/jquery").Include(
                    "~/Scripts/jquery-{version}.js"));

        bundles.Add(new ScriptBundle("~/bundles/jqueryval").Include(
                    "~/Scripts/jquery.validate*"));
    }
}
```

- File Views/Shared/_Layout.cshtml

```
Index.cshtml coursesController.js Index.cshtml Layout.cshtml
@RenderBody()


---


<footer>
    <p>© 2017 - My ASP.NET Application Ánh Nguyễn<br></p>
</footer>
</div>

@Scripts.Render("~/bundles/jquery")
@Scripts.Render("~/bundles/bootstrap")
@Scripts.Render("~/bundles/app")
@RenderSection("scripts", required: false)
</body>
```

TÓM TẮT

Bootstrap là một framework cho phép thiết kế website responsive nhanh hơn và dễ dàng hơn. Bootstrap là bao gồm các HTML templates, CSS templates và Javascript tạo ra những cái cơ bản có sẵn như: typography, forms, buttons, tables, navigation, modals, image carousels và nhiều thứ khác. Trong bootstrap có thêm các plugin Javascript trong nó, giúp cho việc thiết kế website responsive dễ dàng hơn và nhanh chóng hơn.

Trong ASP.NET MVC ta có 2 cách tiếp cận chính đó là *database first* và *code first*, với 1 bài toán yêu cầu nghiệp vụ thay đổi liên tục và phát triển nhanh chóng thì *code first* sẽ là mô hình tiếp cập phù hợp hơn cả. Với cách tiếp cận *code-first*, Entity Framework sẽ tạo các đối tượng bảng cơ sở dữ liệu dựa trên model mà bạn tạo để biểu diễn dữ liệu ứng dụng.

RESTful API là một tiêu chuẩn dùng trong việc thiết kế các API cho các ứng dụng web để quản lý các resource. *RESTful* là một trong những kiểu thiết kế API được sử dụng phổ biến nhất ngày nay. Trọng tâm của REST quy định cách sử dụng các HTTP method (như GET, POST, PUT, DELETE...) và cách định dạng các URL cho ứng dụng web để quản các resource. Với các ứng dụng web được thiết kế sử dụng *RESTful*, lập trình viên có thể dễ dàng biết được URL và HTTP method để quản lý một resource. Bạn cũng cần lưu ý bản thân *RESTful* không quy định logic ứng dụng và *RESTful* cũng không giới hạn bởi ngôn ngữ lập trình ứng dụng. Bất kỳ ngôn ngữ lập trình (hoặc framework) nào cũng có thể áp dụng *RESTful* trong việc thiết kế API cho ứng dụng web.

CÂU HỎI ÔN TẬP

Câu 1: Bootstrap là gì? Tại sao bạn nên sử dụng Bootstrap?

Câu 2: AJAX là gì và nó hoạt động như thế nào?

Câu 3: Sự khác biệt giữa REST & RESTful?

Câu 4: Mô tả vai trò và kiến trúc của ASP.NET Identity?

TÀI LIỆU THAM KHẢO

1. Pro ASP.NET MVC 5 (2013), ISBN-10: 1430265299, Apress; 5th ed. edition
2. ASP.NET MVC 5 - Building a Website with Visual Studio 2015 and C Sharp (2016), ISBN-10: 1535167866, CreateSpace Independent Publishing Platform
3. ASP.NET MVC with Entity Framework and CSS (2016), ASIN: B01LYSLEMA, Apress; 1st ed. edition