ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN
VNUHCM - UIT

LẬP TRÌNH WEB

CHƯƠNG 2
LÀM VIỆC VỚI CONTROLLER

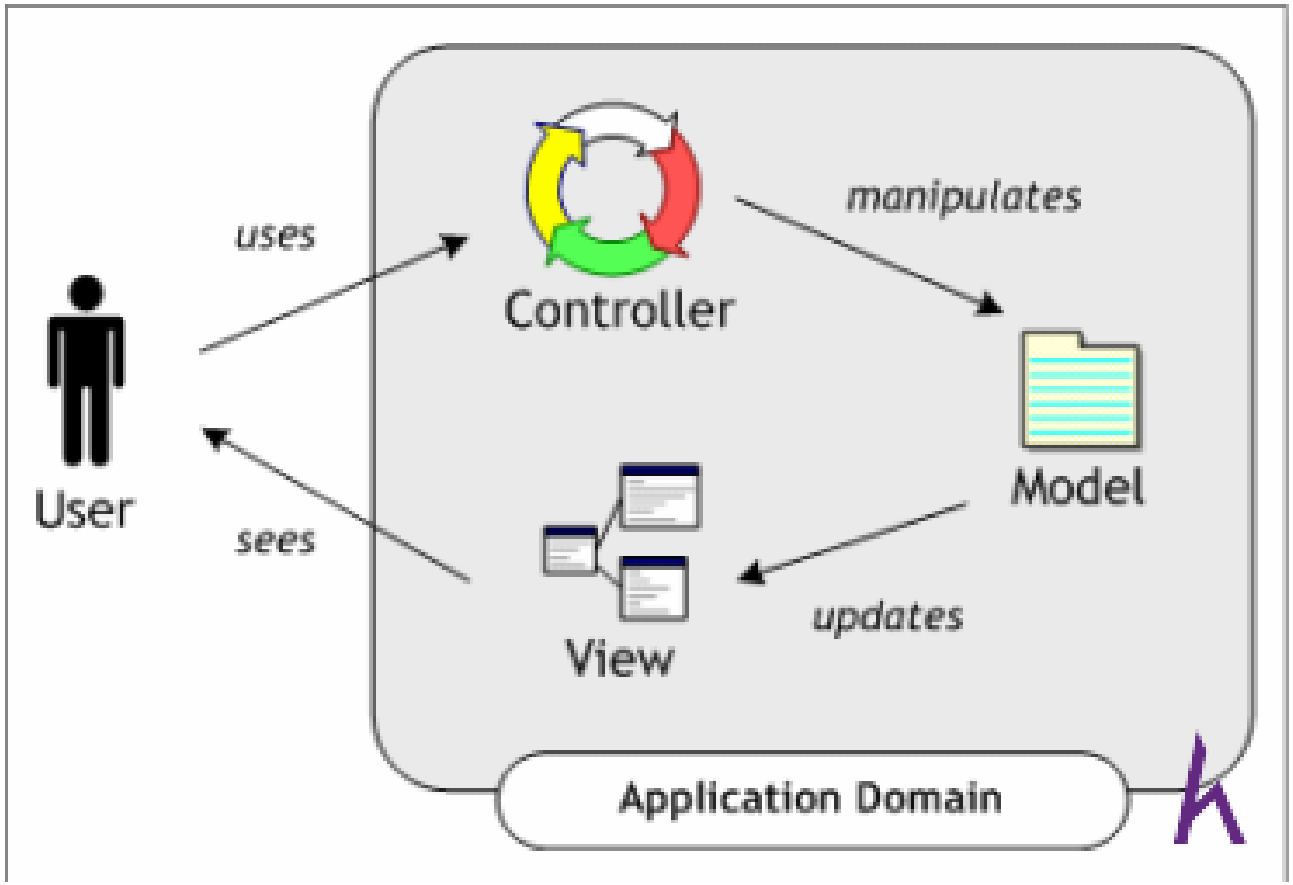
Email: dungta@uit.edu.vn

NỘI DUNG

1. Kiến trúc MVC
2. Khái niệm về Controller
3. Tạo và sử dụng Controller
4. Làm việc với các Action methods
5. Cơ chế Routing trong ASP.NET MVC
6. Cách chuyển hướng trong action - Redirect

1. KIẾN TRÚC MVC

❖ MVC (Model – View - Controller)



2. KHÁI NIỆM VỀ CONTROLLER

- Controller trong Framework MVC xử lý mọi yêu cầu URL đến. Controller là một lớp trong namespace System.Web.Mvc.Controller. Lớp Controller chứa các phương thức public gọi là các phương thức Action. Controller và phương thức action của nó xử lý các yêu cầu trình duyệt đến, lấy dữ liệu mô hình cần thiết và trả về cho trình duyệt.

2. KHÁI NIỆM VỀ CONTROLLER

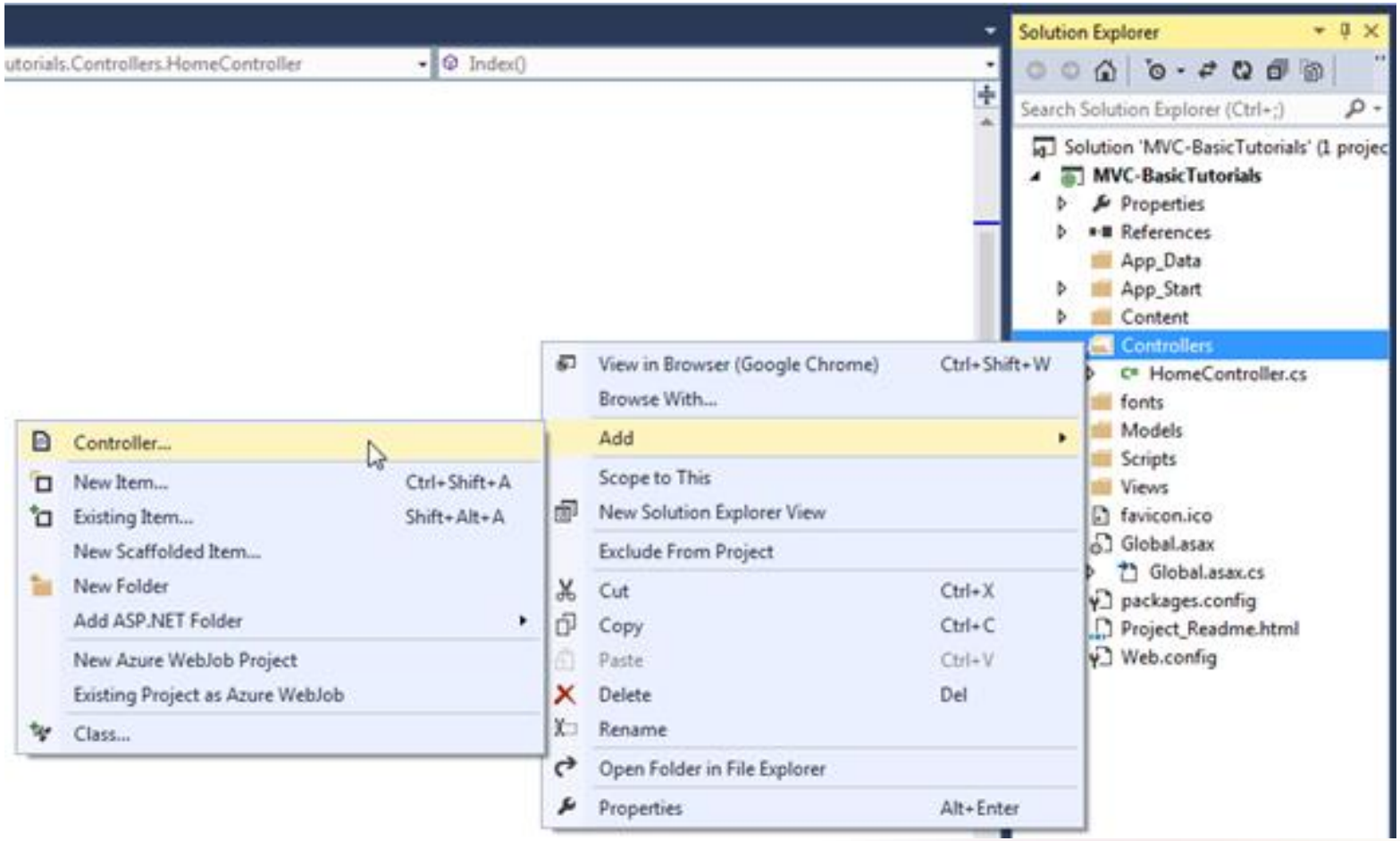
- Trong ASP.NET MVC, mọi tên lớp của trình điều khiển phải kết thúc bằng một từ "Controller".
- Ví dụ: bộ điều khiển cho trang chủ phải là HomeController và bộ điều khiển cho sinh viên phải là StudentController. Ngoài ra, mỗi lớp Controller được đặt trong thư mục Controller của cấu trúc thư mục MVC.

2. KHÁI NIỆM VỀ CONTROLLER

- Controller có 3 trách nhiệm chính:
  - Nhận request
  - Dựng model
  - Gửi trả response

3. TẠO VÀ SỬ DỤNG CONTROLLER

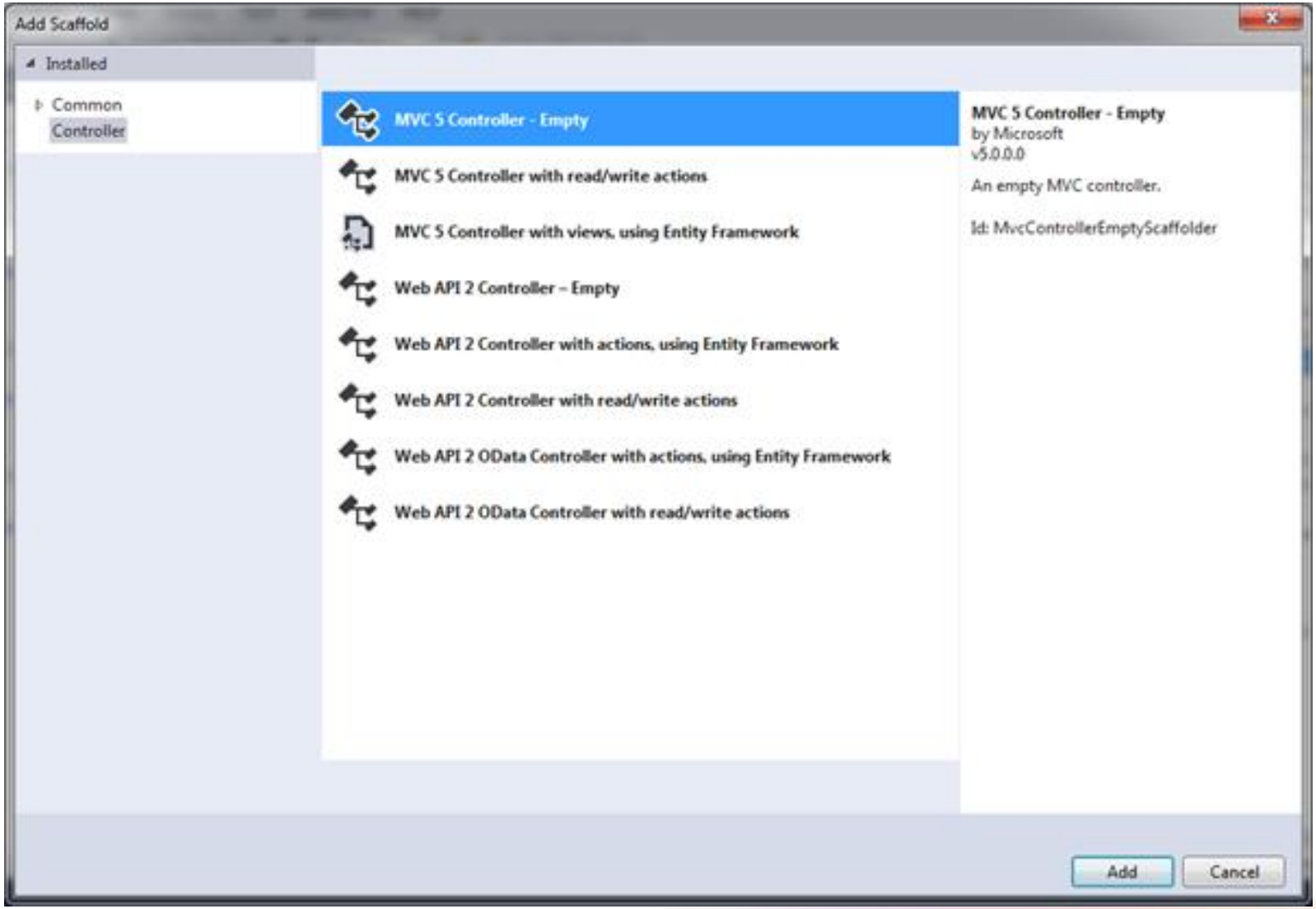
❖ Thêm mới Controller:



3. TẠO VÀ SỬ DỤNG CONTROLLER

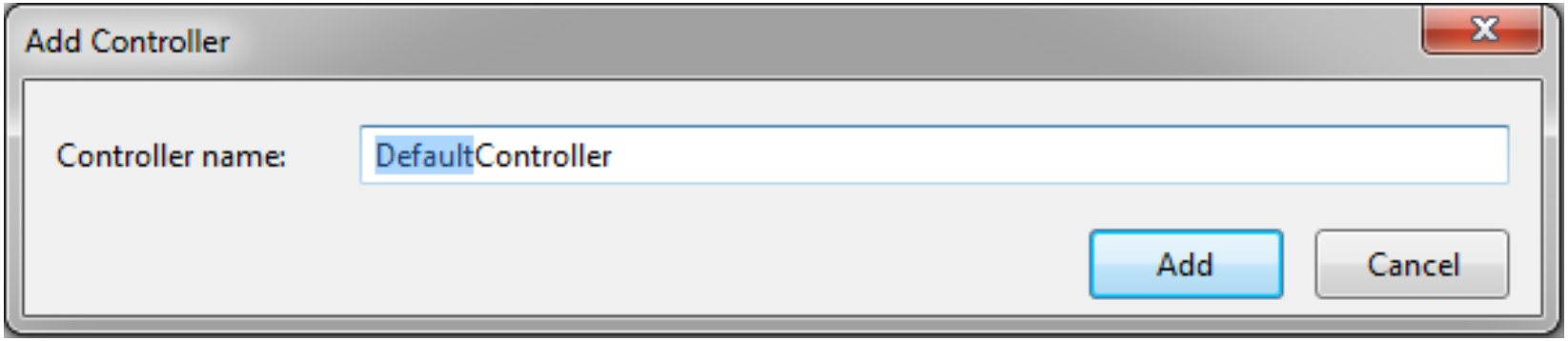
- Ở hộp thoại Add Scaffold, chọn MVC 5 Controller – Empty và nhấn Add. Ở bước này, do chúng ta mới làm quen với Controller cho nên chúng ta sử dụng mẫu Controller dạng Empty, tức là nội dung trống. Từ đó, chúng ta sẽ viết 1 số đoạn code làm quen. Ở những bài tiếp theo, các bạn sẽ làm quen các dạng Controller khác.

3. TẠO VÀ SỬ DỤNG CONTROLLER



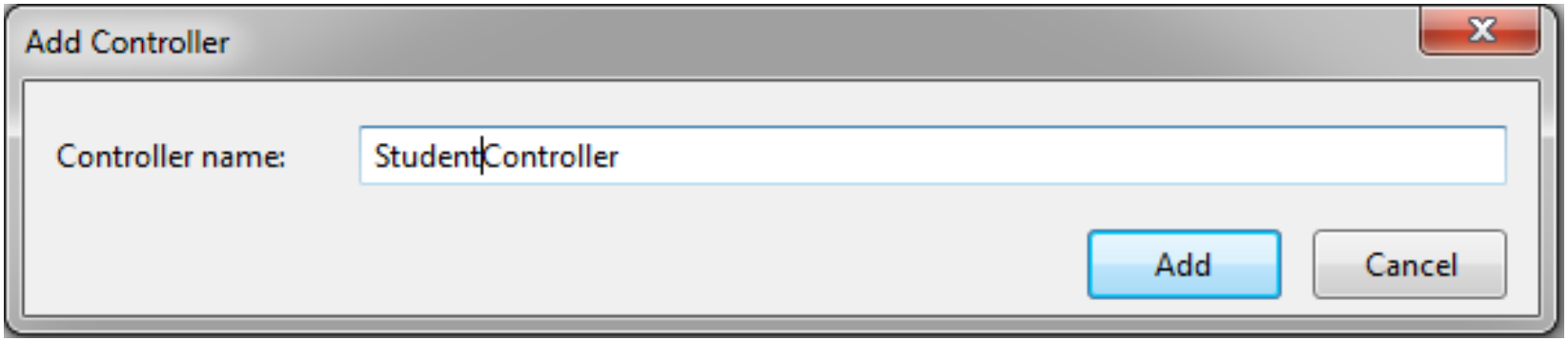
3. TẠO VÀ SỬ DỤNG CONTROLLER

- Thêm Scaffold chứa các mẫu khác nhau để tạo bộ điều khiển mới. Chúng ta sẽ tìm hiểu về các mẫu khác sau. Hiện tại, chọn "MVC 5 Controller - Empty" và nhấp vào Add. Nó sẽ mở hộp thoại Add điều khiển như hình sau:



3. TẠO VÀ SỬ DỤNG CONTROLLER

- Sau đó, đặt tên Controller là StudentController, lưu ý bạn nên để tiền tố "Controller" cuối cùng khi đặt tên để dễ phân biệt lớp code nào là Controller.



3. TẠO VÀ SỬ DỤNG CONTROLLER

- Ở bước này, thư mục Controller, lớp StudentController.cs được tạo ra với nội dung mặc định như trong hình sau:

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

namespace MVC_BasicTutorials.Controllers
{
    public class StudentController : Controller
    {
        // GET: Student
        public ActionResult Index()
        {
            return View();
        }
    }
}
```

3. TẠO VÀ SỬ DỤNG CONTROLLER

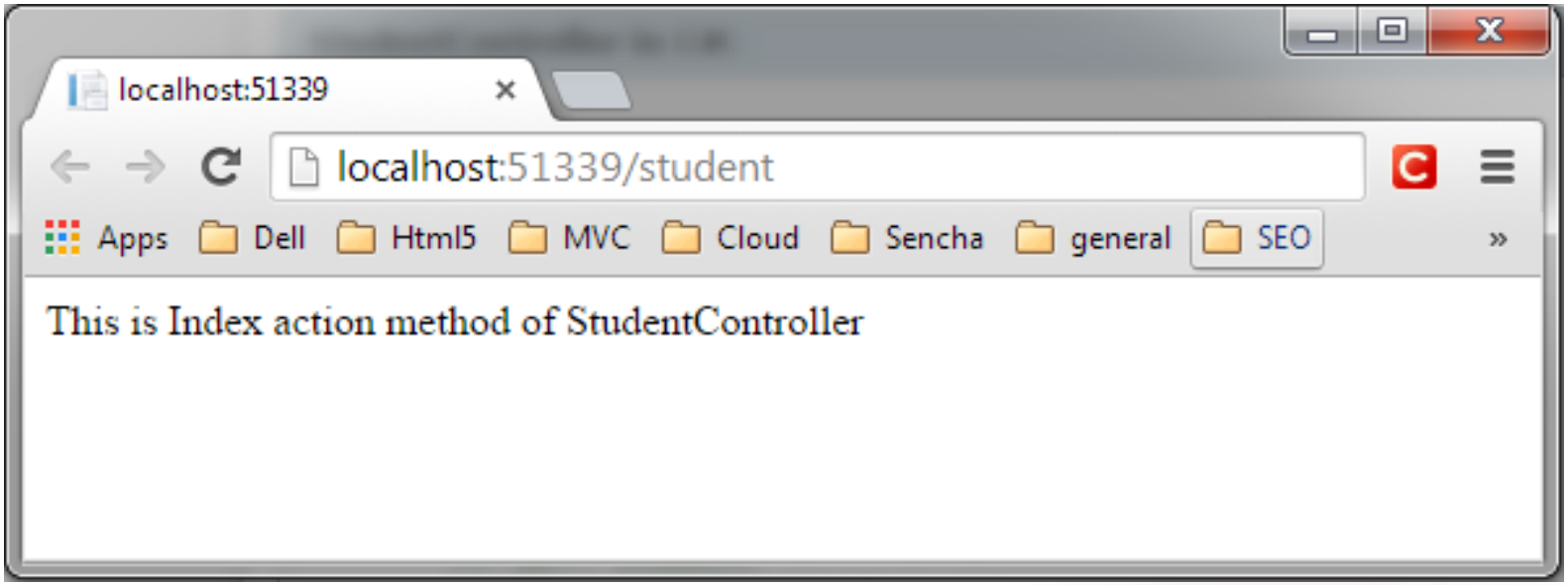
- Tiếp theo, bạn thêm đoạn mã vào tập tin StudentController.cs như sau:

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

namespace MVC_BasicTutorials.Controllers
{
    public class StudentController : Controller
    {
        // GET: Student
        public string Index()
        {
                return "This is Index action method of StudentController";
        }
    }
}
```

3. TẠO VÀ SỬ DỤNG CONTROLLER

- Bạn có thể tạo nhiều phương thức thực thi ở tập tin StudentController.cs tùy ý. Tiếp theo, bạn thực thi ứng dụng bằng cách nhấn F5 hoặc Ctrl + F5 (chế độ không cần Debug) để xem kết quả.



3. TẠO VÀ SỬ DỤNG CONTROLLER

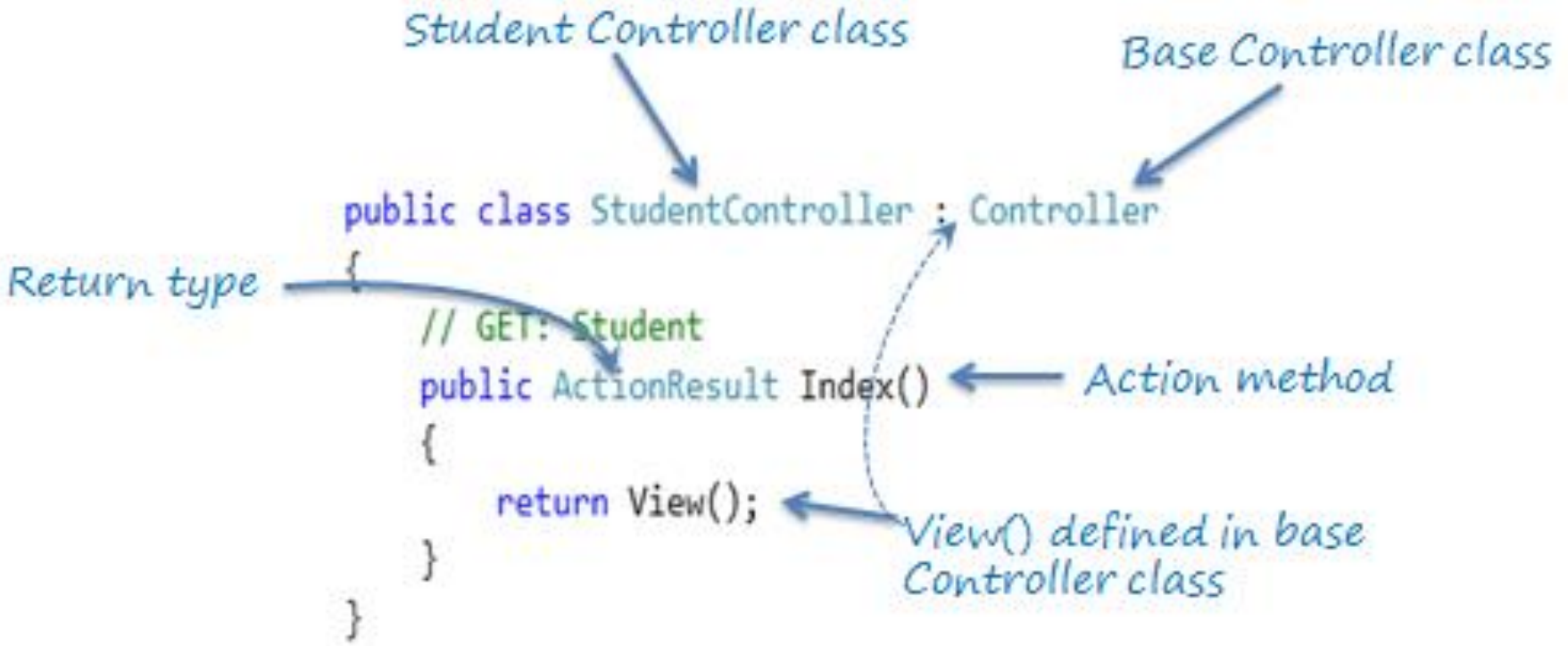
- Controller xử lý các yêu cầu của URL gửi đến. Controller MVC gửi yêu cầu đến controller và phương thức hành động dựa trên URL và các Controller được định cấu hình.
- Controller kế thừa từ System.Web.Mvc.Controller.
- Một Controller mới có thể được tạo bằng các mẫu scaffolding khác nhau. Bạn có thể tạo mẫu scaffolding tùy ý.

4. ACTION METHODS

- Tất cả các phương thức public của lớp Controller được gọi là Phương thức hành động (Action). Chúng giống như bất kỳ phương thức bình thường nào khác với các hạn chế sau:
  - Phương thức Action phải là public. Không thể private hay protected
  - Phương thức Action không thể nạp chồng (overloaded)
  - Phương thức Action không thể là phương thức static.

4. ACTION METHODS

- Ví dụ sau về phương thức action là Index của lớp StudentController



4. ACTION METHODS

❖ Phương thức Action mặc định:

- Mỗi controller có phương thức action mặc định được cài đặt trong lớp RouteConfig. Phương thức Index là phương action mặc định, được cài đặt mặc định.
- Route mặc định:

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

4. ACTION METHODS

- ActionResult:
  - MVC framework chứa nhiều lớp kết quả khác nhau, có thể được trả về từ một phương thức hành động. Có các lớp kết quả đại diện cho các loại phản hồi khác nhau, chẳng hạn như html, tệp, chuỗi, json, javascript,… Bảng sau liệt kê tất cả các lớp kết quả có sẵn trong ASP.NET MVC.

4. ACTION METHODS

Result Class	Description
ViewResult	Trả về HTML
EmptyResult	Không trả về cái gì cả.
ContentResult	Trả về chuỗi ký tự
FileContentResult/ FilePathResult/ FileStreamResult	Trả về nội dung của tập tin
JavaScriptResult	Trả về JavaScript script.
JsonResult	Trả về một JSON có thể sử dụng trong AJAX
RedirectResult	Chuyển hướng người dùng
RedirectToRouteResult	Chuyển hướng hành động khác.
PartialViewResult	Trả về HTML từ Partial view
HttpUnauthorizedResult	Trả về lỗi và HTTP Code

4. ACTION METHODS

- Lớp ActionResult là một lớp cơ sở của tất cả các lớp kết quả ở trên, do đó, nó có thể là kiểu trả về của các phương thức hành động trả về bất kỳ loại kết quả nào được liệt kê ở trên. Tuy nhiên, bạn có thể chỉ định lớp kết quả phù hợp làm kiểu trả về của phương thức hành động.

4. ACTION METHODS

- Các tham số của phương thức Action:
  - Mỗi phương thức hành động có thể có các tham số đầu vào như các phương thức bình thường.
  - Các tham số có thể là kiểu dữ liệu nguyên thủy hoặc tham số kiểu phức tạp như trong ví dụ ở slide sau.

4. ACTION METHODS

❖ Các tham số của phương thức Action:

Ví dụ: Các tham số của phương thức Action

```
[HttpPost]
public ActionResult Edit(Student std)
{
    // update student to the database

    return RedirectToAction("Index");
}

[HttpDelete]
public ActionResult Delete(int id)
{
    // delete student from the database whose id matches with specified id

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

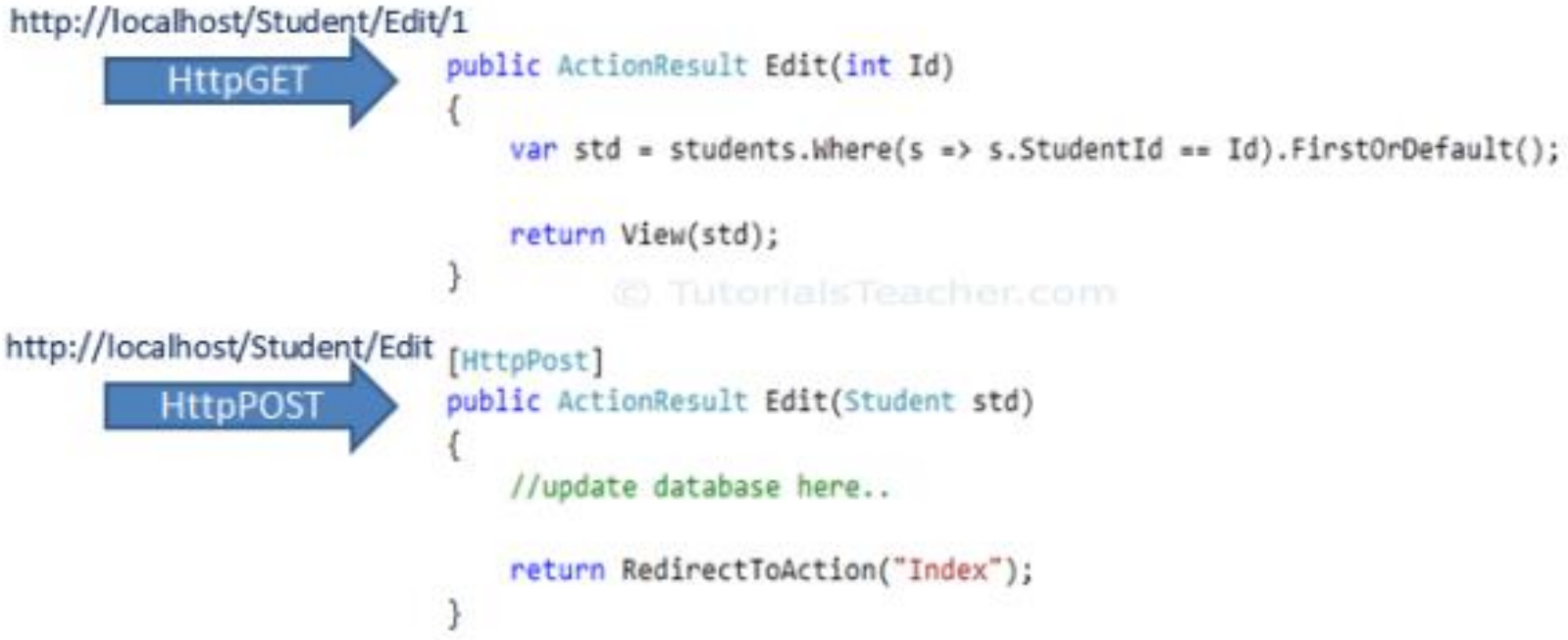
4. ACTION METHODS

❖ ActionVerb:

- Một phương thức action khi được định nghĩa có thể được gọi theo Get và Post. Cách này cho phép lập trình viên có thể định nghĩa hai phương thức khác nhau nhưng cùng tên, một phương thức có thể dùng Http Get và phương thức kia thì đáp có thể dùng HttpPost.
- MVC Framework hỗ trợ nhiều ActionVerb, như HttpGet, HttpPost, HttpPut, HttpDelete, HttpOptions & HttpPatch.

4. ACTION METHODS

- ActionVerb:
  - Ví dụ minh họa sử dụng HttpGet và HttpPost khi thực hiện nghiệp vụ chỉnh sửa dữ liệu Student.



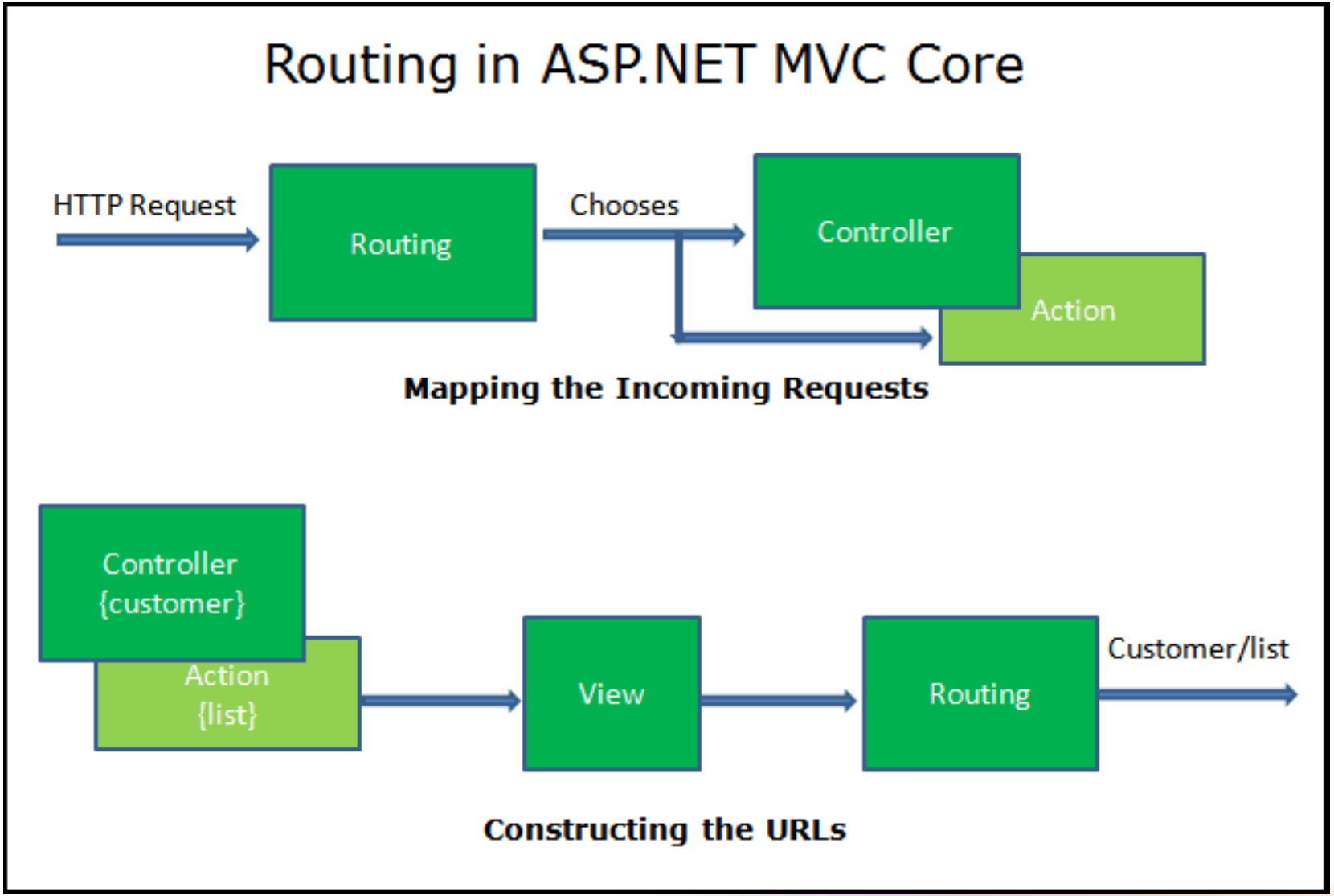
4. ACTION METHODS

Phương thức Http	Mô tả
GET	Các tham số trên URL (query string) tự động được thêm vào như các tham số. HttpGet được dùng để nhận một resource từ server.
POST	HttpPost attribute giới hạn action method chấp nhận HTTP Request sử dụng Post verb. Post verb được dùng để tạo mới bản ghi.
PUT	HttpPut attribute giới hạn action method chỉ chấp nhận các HTTP Request sử dụng Put verb. Put verb được dùng để cập nhật hoặc tạo mới tài nguyên.
HEAD	HttpHead attribute giới hạn action method chỉ chấp nhận HTTP Request sử dụng Head verb. Head verb được dùng để nhận các HTTP Header. Phương thức này được định danh cho GET ngoại trừ các serrver không trả về message body.
OPTIONS	HttpOptions attribute giới hạn action method chỉ chấp nhận các request sử dụng Options verb. Method này nhận thông tin về tùy chọn giao tiếp được hỗ trợ bởi web server.
DELETE	HttpDelete attribute giới hạn Action method chỉ chấp nhận HTTP Request sử dụng Delete verb. Delete verb được dùng để xóa tài nguyên đang tồn tại
PATCH	HttPatch attribute giới hạn action method chỉ nhận các HTTP Request sử dụng Options verb. Method nà sử dụng cho toàn bộ hoặc một phần việc cập nhật tài nguyên.

5. CƠ CHẾ ROUTING TRONG ASP.NET

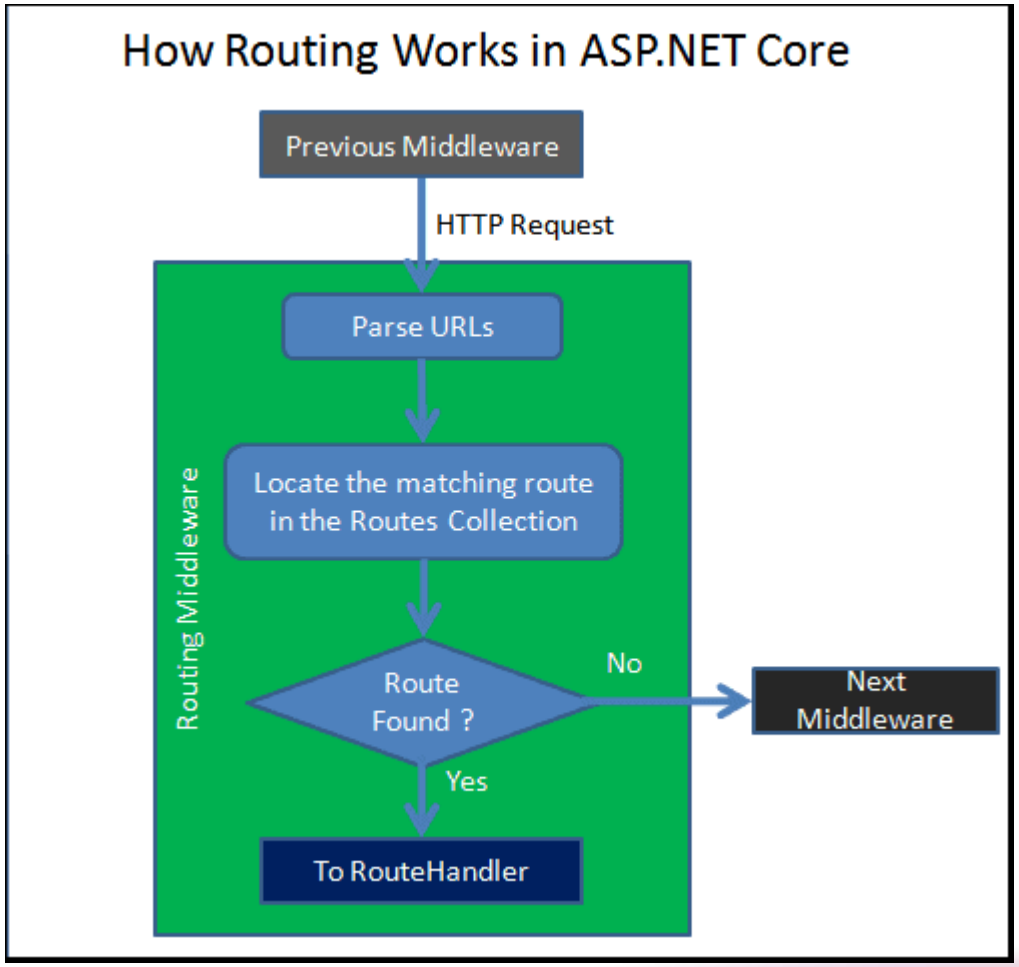
- Một trong các thành phần quan trọng nhất của kiến trúc MVC là cơ chế routing (định tuyến). Nó là cơ chế quyết định xem Controller nào sẽ được xử lý request nào.
- Routing là gì?
- Routing có 2 trách nhiệm chính:
  - Nó map request đến vào Controller Action.
  - Tạo ra URL đầu ra tương ứng với Controller action.

5. CƠ CHẾ ROUTING TRONG ASP.NET



5. CƠ CHẾ ROUTING TRONG ASP.NET

❖ Routing làm việc ra sao?



5. CƠ CHẾ ROUTING TRONG ASP.NET

- Mỗi Route bao gồm các thông tin như tên, mẫu URL (URL pattern) hay còn gọi là template url, thông tin controller action mặc định và ràng buộc (constraints). URL pattern được so sánh với URL đến xem có đúng mẫu không. Một ví dụ của URL pattern là: {controller=Home}/{action=Index}/{id?}

- Route được định nghĩa trong Microsoft.AspNetCore.Routing.

5. CƠ CHẾ ROUTING TRONG ASP.NET

❖ Route Collection là gì?

- Route Collection là một tập hợp tất cả ác Route trong ứng dụng. Một ứng dụng sẽ lưu một tập hợp các route ở một nơi duy nhất trong bộ nhớ. Các Route này sẽ thêm vào collection khi ứng dụng khởi động.
- Route Module sẽ tìm kiếm một Route match với URL request đến trong mỗi một Route của Route Collection. Route Collection được định nghĩa trong Microsoft.AspNetCore.Routing.

5. CƠ CHẾ ROUTING TRONG ASP.NET

❖ Route Handler là gì?

- Route Handler là một thành phần quyết định sẽ làm gì với Route. Khi cơ chế routing tìm được một Route thích hợp cho một request đến, nó sẽ gọi đến RouteHandler và gửi route đó cho RouteHandler xử lý.
- Route Handler là class triển khai từ interface IRouteHandler. Trong ASP.NET Core thì Route được xử lý bởi MvcRouteHandler.

5. CƠ CHẾ ROUTING TRONG ASP.NET

- MVCRouteHandler
  - Đây là Route Handler mặc định của ASP.NET Core MVC Middleware. MVCRouteHandler được đăng ký khi đăng ký MVC Middleware. Bạn có thể ghi đè việc này bằng cách tự tạo cho mình một custom implementation của Route Handler.
  - MVCRouteHandler được định nghĩa trong namespace: Microsoft.AspnetCore.Mvc

6. CÁCH CHUYỂN HƯỚNG TRONG ACTION - REDIRECT

1. Chuyển hướng qua action khác cùng Controller
2. Chuyển hướng action khác Controller
3. Chuyển hướng tới một URL cụ thể
4. Chuyển hướng bằng RouteName

6. CÁCH CHUYỂN HƯỚNG TRONG ACTION - REDIRECT

❖ Chuyển hướng qua action khác cùng Controller:

```
public ActionResult Index()
{
    return RedirectToAction("actionName");
}
```

6. CÁCH CHUYỂN HƯỚNG TRONG ACTION - REDIRECT

❖ Chuyển hướng qua action khác cùng Controller:

```
public ActionResult TrangChu()
{
    // Chuyển hướng qua action khác, cùng controller
    return RedirectToAction("DangNhap");
}

0 references
public ActionResult GioiThieu()
{
    return View();
}

0 references
public ActionResult DangNhap()
{
    return View();
}
```

6. CÁCH CHUYỂN HƯỚNG TRONG ACTION - REDIRECT

❖ Chuyển hướng qua action khác khác Controller:

```
public ActionResult Index()
{
    return RedirectToAction("actionName", "controllerName");
}
```

6. CÁCH CHUYỂN HƯỚNG TRONG ACTION - REDIRECT

❖ Chuyển hướng tới một URL:

```
public ActionResult Index()
{
    return Redirect("URL");
}
```

❖ Ví dụ:

```
public ActionResult Index()
{
    return Redirect("https://www.google.com/");
}
```

6. CÁCH CHUYỂN HƯỚNG TRONG ACTION - REDIRECT

❖ Chuyển hướng bằng RouteName:

```
public ActionResult Index()
{
    return RedirectToRoute("routeName");
}
```

```
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 = "Home",
                action = "Index", id = UrlParameter.Optional }
        );
    }
}
```

Email: dungta@uit.edu.vn