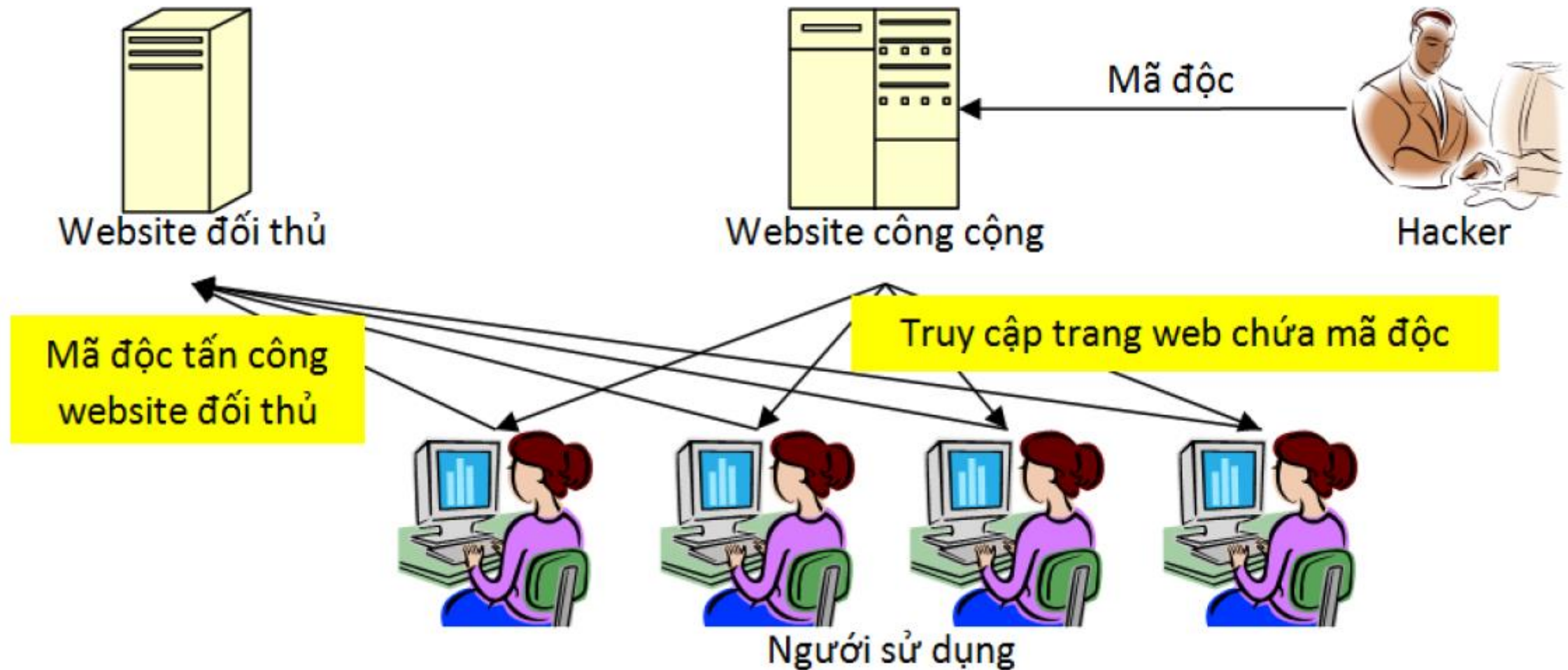


Chủ đề 5. BẢO MẬT VÀ ĐỊNH TUYẾN TRONG ASP.NET MVC

1. Bảo mật trong ASP.NET MVC

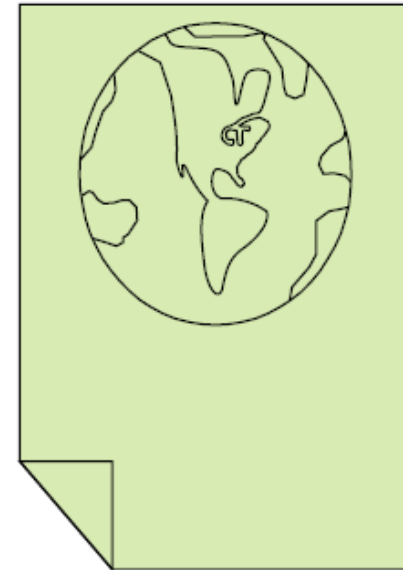
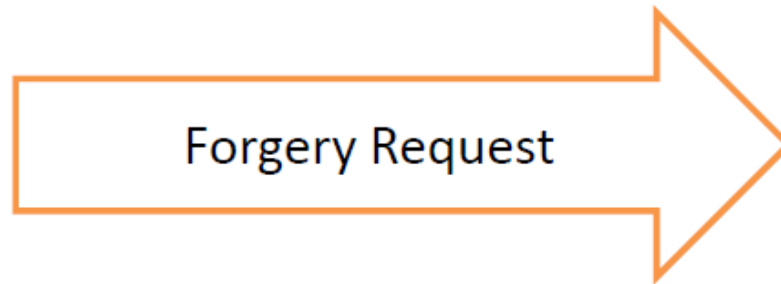
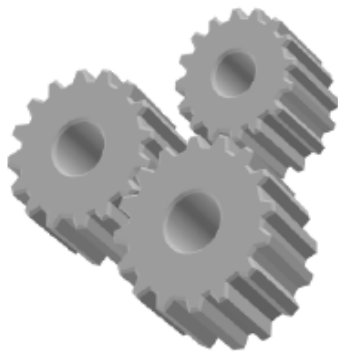
- Tấn công bằng cách nhúng mã script
- Anti Forgery (gửi tự động)
- Authentication & Authorization (kiểm soát đăng nhập và phân quyền)

a. Tấn công website bằng cách nhúng mã script vào một trang web của site khác



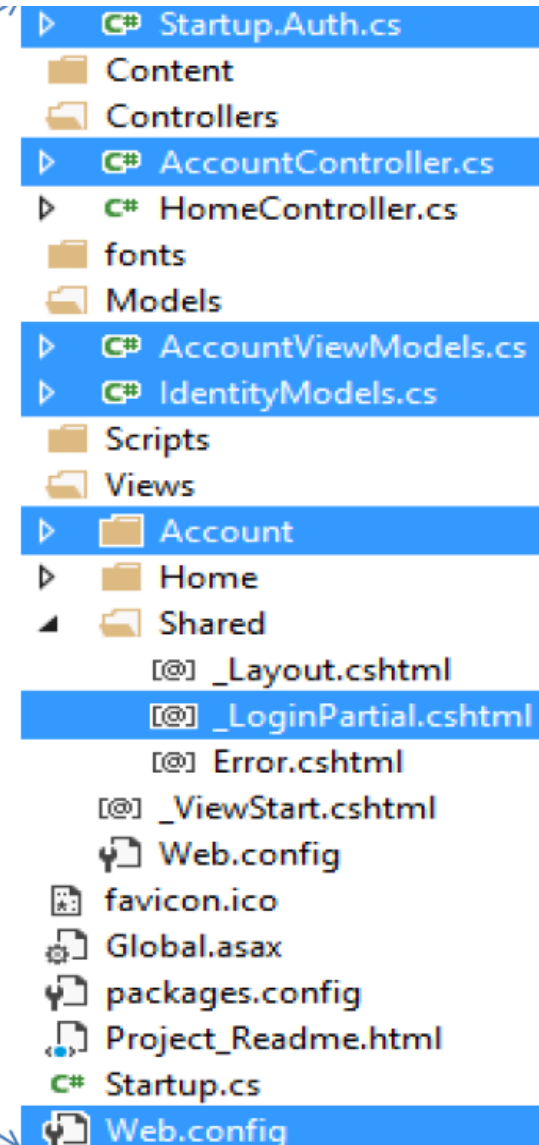
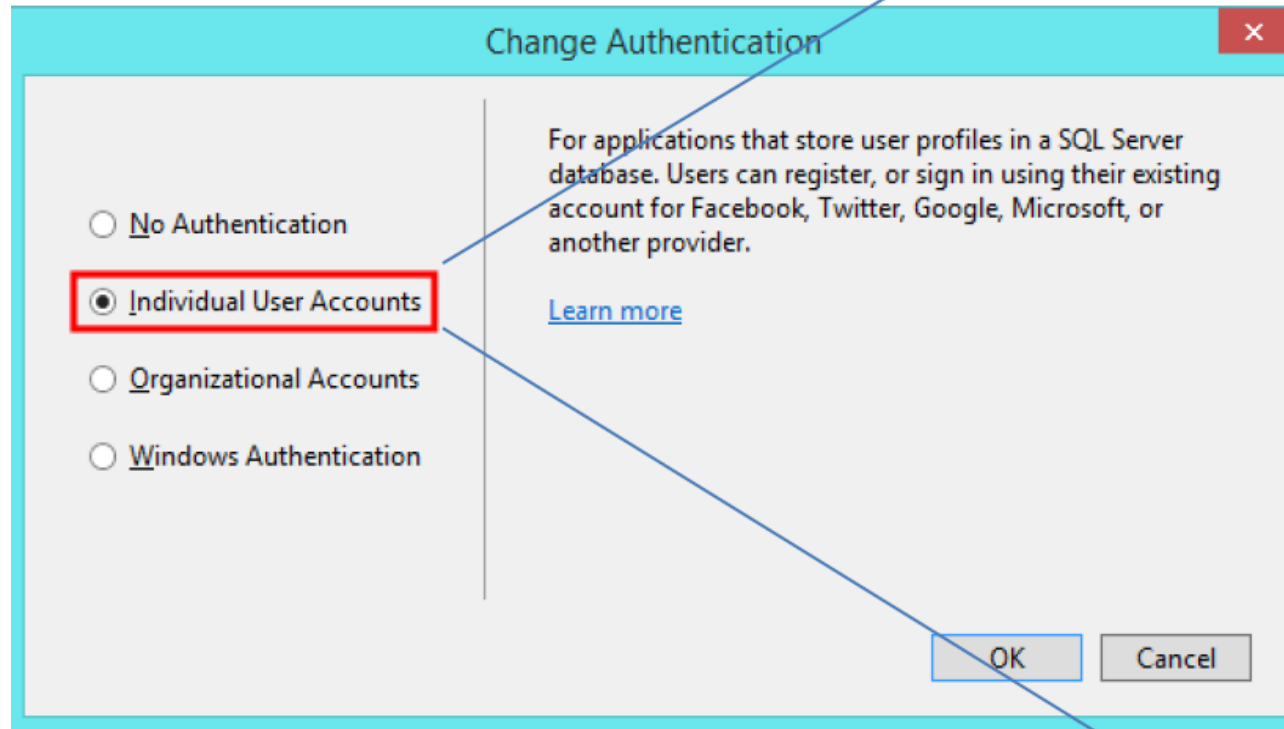
b. Gửi tự động (AntiForgery)

- ❑ Giả mạo request để gửi dữ liệu tự động đến server để thực hiện hành động một cách tự động



c. Authentication & Authorization

- ❑ Kiểm soát đăng nhập
- ❑ Kiểm soát vai trò



Các thành phần Security:

❑ Startup.Auth.cs:

✎ Cấu hình trang đăng nhập và các nguồn đăng nhập bên ngoài

❑ AccountController.cs:

✎ Định nghĩa các action đăng nhập, đăng xuất, đăng ký, đổi mật khẩu

❑ AccountViewModels.cs:

✎ Các model buộc dữ liệu với giao diện

❑ IdentityModels.cs:

✎ Model dữ liệu security như User, Role, UserInRoles

❑ Account/*.cshtml:

✎ các view liên quan đến các action trong AccountController

❑ Shared/_LoginPartial.cshtml:

✎ view thành phần nhúng vào layout

❑ Web.config:

✎ chứa khai báo chuỗi kết nối đến CSDL

Tổ chức AccountController:

Buộc phải đăng nhập mới sử dụng các action của controller này

[Authorize]

```
public class AccountController : Controller
{
    public AccountController()
        : this(new UserManager<ApplicationUser>(
            new UserStore<ApplicationUser>(new ApplicationDbContext()))))
    {
    }

    public AccountController(UserManager<ApplicationUser> userManager)
    {
        UserManager = userManager;
    }

    public UserManager<ApplicationUser> UserManager { get; private set; }

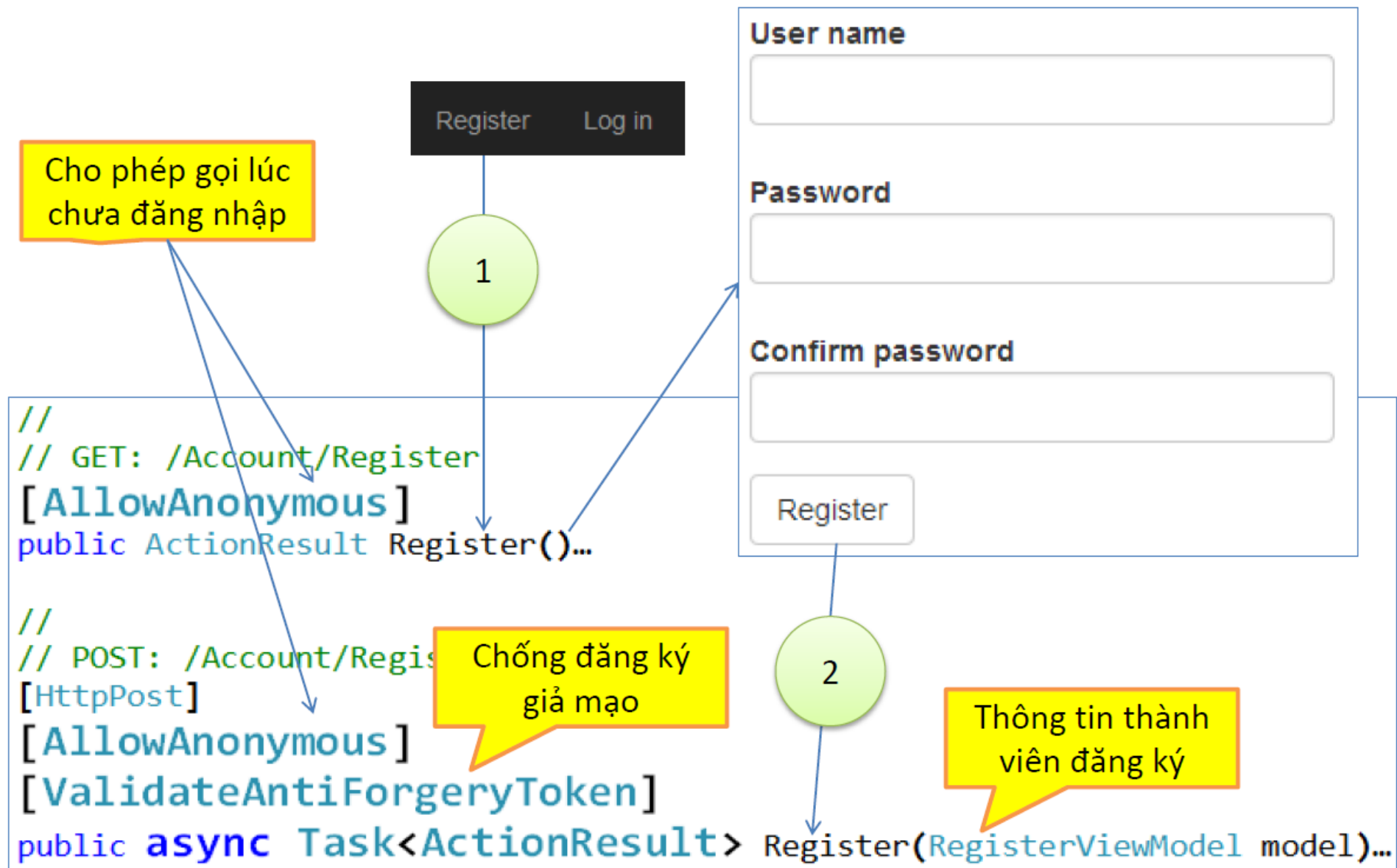
    ...các action và mã hỗ trợ khác...
}
```

DbContext làm việc với CSDL thành viên

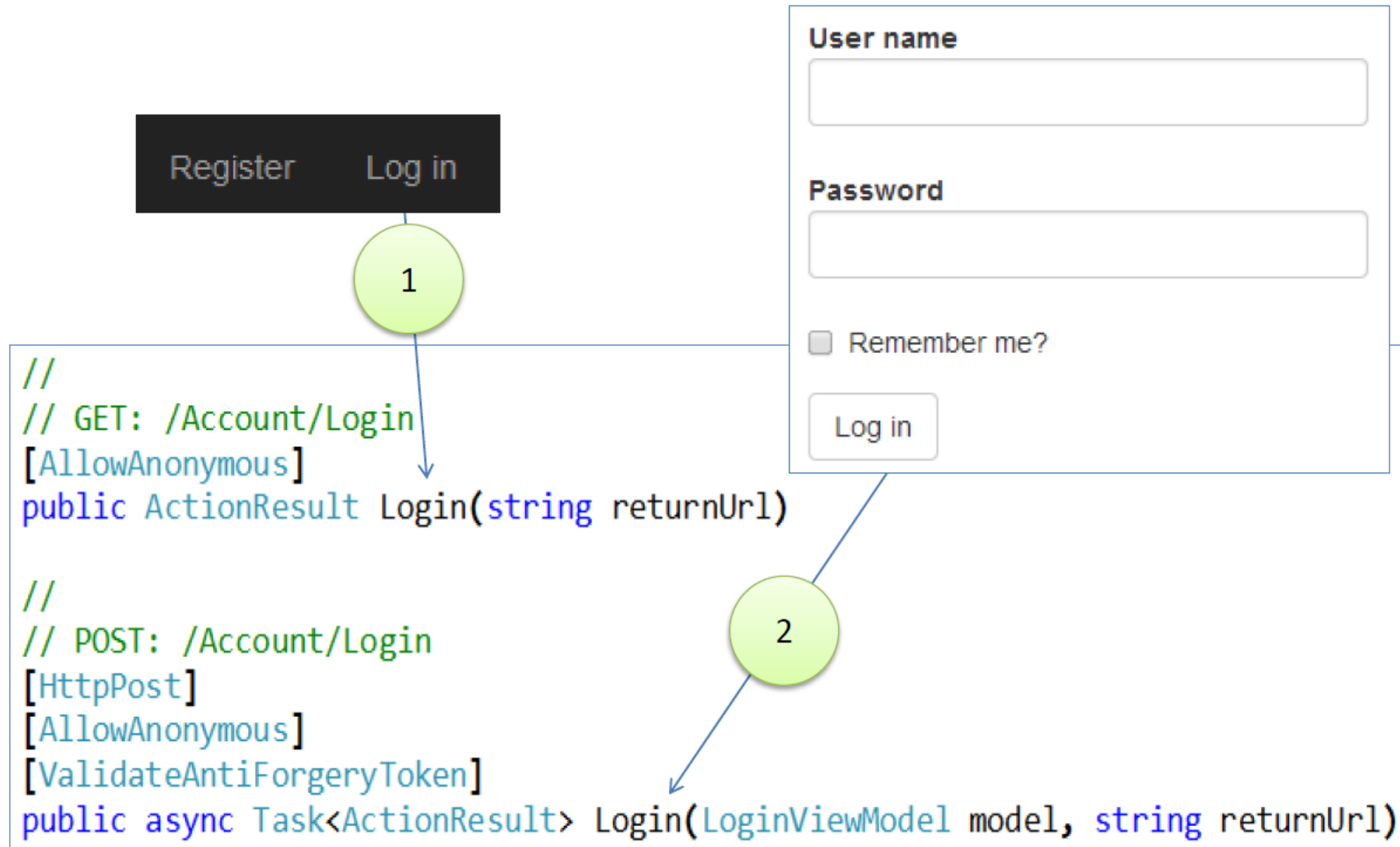
Sử dụng để quản lý thành viên

Các action được bảo vệ

Register Action:



Login Action:



Account mở rộng:

Use a local account to log in.

User name

Password

☐ Remember me?

Log in

Use another service to log in.

There are no external authentication services configured. See [this article](#) for details on setting up this ASP.NET application to support logging in via external services.

```
app.UseFacebookAuthentication(  
    appId: "1467885686787500",  
    appSecret: "157fca68bad034ffc50ef2da294362e8");  
  
app.UseGoogleAuthentication();
```

User name

Password

☐ Remember me?

Log in

Use another service to log in.

Google

Facebook

Change Password và LogOff:

```
//  
// POST: /Account/LogOff  
[HttpPost]  
[ValidateAntiForgeryToken]  
public ActionResult LogOff()
```

Hello nnghiem! Log off

1

```
//  
// GET: /Account/Manage  
public ActionResult Manage(ManageMessageId? message)
```

```
//  
// POST: /Account/Manage  
[HttpPost]  
[ValidateAntiForgeryToken]  
public async Task<ActionResult> Manage(ManageUserViewModel model)
```

Current password

New password

Confirm new password

Change password

2

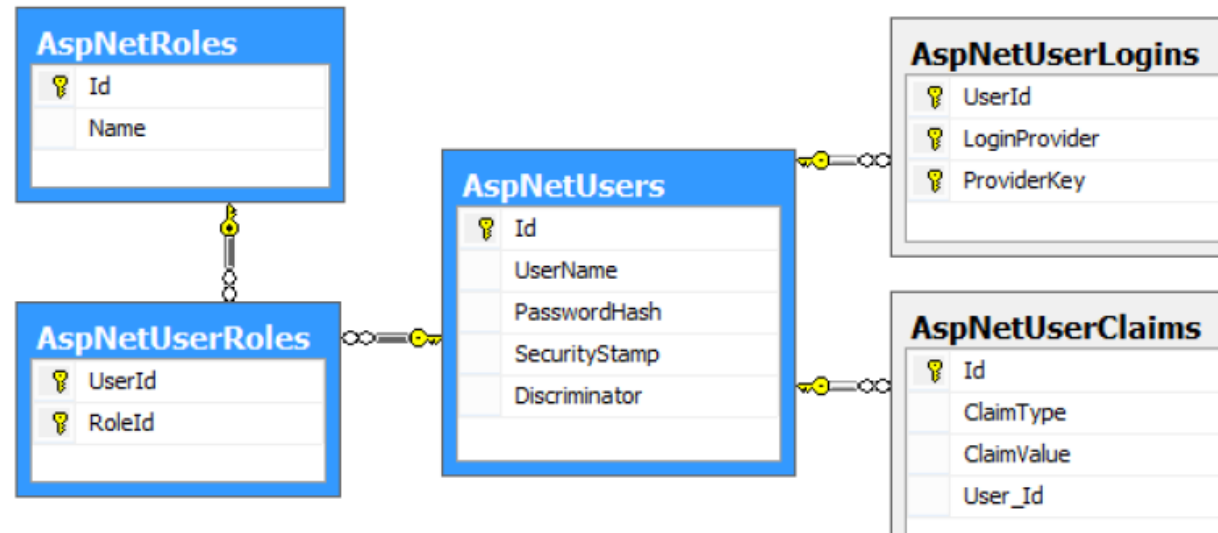
IdentityModel.cs

- ❑ Mô hình CSDL security
- ❑ ApplicationUser thông tin thành viên
- ❑ ApplicationDbContext là DbContext làm việc với CSDL thông qua kết nối DefaultConnection được khai trong Web.config.

```
public class ApplicationUser : IdentityUser
{
}

public class ApplicationDbContext : IdentityDbContext<ApplicationUser>
{
    public ApplicationDbContext(): base("DefaultConnection"){ }
}
```


CSDL thành viên:



Bảng	Mô tả	Thực thể
AspNetUsers	Quản lý thông tin thành viên	IdentityUser
AspNetRoles	Quản lý vai trò	IdentityRole
AspNetUserRoles	Phân quyền – user nào có vai trò gì	IdentityUserRole
AppNetUserLogins	Thông tin thêm của tài khoản ngoài	IdentityUserLogin
AspNetUserClaims	Thông tin thêm của tài khoản Active Directory	IdentityUserClaim

Các thực thể:

Phương thức/Thuộc tính	Mô tả
Id: string	Mã chuỗi tự tăng
UserName: string	Tên đăng nhập
Email: string	Email
PhoneNumber: string	Số điện thoại
Roles: ICollection<IdentityUserRole>	Danh sách roles của user



Thuộc tính/Phương thức	Mô tả
UserId	Mã thành viên
RoleId	Mã vai trò

Thuộc tính	Mô tả
Id: string	Mã vai trò
Name: string	Tên vai trò
Users: ICollection<IdentityUserRole>	Danh sách user thuộc vai trò này

Security API:

❑ IdentityDbContext

✎ Quản lý CSDL thành viên

❑ UserManager<IdentityUser>

✎ Quản lý thành viên

❑ RoleManager<IdentityRole>

✎ Quản lý vai trò

❑ IAuthenticationManager

✎ Thông báo cho hệ thống về việc đăng nhập và đăng xuất của thành viên

Khởi tạo:

❑ DB = new **IdentityDbContext()**

✎ Tạo DbContext

❑ UM = new **UserManager**<IdentityUser>(new
UserStore<IdentityUser>(db))

✎ Tạo UserManager

❑ RM = new **RoleManager**<IdentityRole>(new
RoleStore<IdentityRole>(db))

✎ Tạo RoleManager

❑ AU = **HttpContext.GetOwinContext().Authentication**

✎ Lấy authenticate

UserManager<IdentityUser> API

❑ Các thao tác quản lý thành viên đăng nhập

Phương thức/Thuộc tính	Mô tả
Users	Danh sách thành viên
Create()	Tạo thành viên mới
Delete()	Xóa thành viên
Update()	Cập nhật thông tin thành viên
Find()	Tìm theo thông tin đăng nhập (username, password)
FindById()	Tìm theo mã
FindByEmail()	Tìm theo email
FindByName()	Tìm theo username
ChangePassword()	Đổi mật khẩu
AddPassword()	Cấp mật khẩu mới
RemovePassword()	Xóa mật khẩu
CreateIdentity()	Tạo một ClaimIdentity kết hợp với một user
GetRoles()	Lấy danh sách vai trò của một user
AddToRole()	Thêm vai trò cho 1 user
RemoveFromRole()	Xóa vai trò khỏi 1 user
IsUserInRole()	Kiểm tra một vai trò của một user

❑ Các thao tác quản lý vai trò thành viên

Thuộc tính/Phương thức	Mô tả
Roles	Danh sách vai trò
Create(), CreateAsync()	Tạo vai trò mới
Delete(), DeleteAsync()	Xóa vai trò
Update(), UpdateAsync()	Cập nhật vai trò
FindById(), FindByIdAsync()	Tìm vai trò theo mã
FindByName(), FindByNameAsync()	Tìm vai trò theo tên
RoleExists(), RoleExistsAsync()	Kiểm tra sự tồn tại của vai trò

IAuthenticationManager API:

❑ Thao tác chính của IAuthenticationManager

Thuộc tính/Phương thức	Mô tả
SignIn()	Thông báo đăng nhập đến hệ thống
SignOut()	Thông báo đăng xuất đến hệ thống
GetExternalLoginInfoAsync()	Lấy thông tin đăng nhập bên ngoài

Nâng cấp và mở rộng:

- ❑ Mở rộng thông tin người dùng
- ❑ Kích hoạt tài khoản qua email
- ❑ Quên mật khẩu
- ❑ ApplicationUser CRUD

Mở rộng thông tin thành viên:

User name	<input type="text" value="nnghiem"/>
Password	<input type="password" value="....."/>
Confirm password	<input type="password" value="....."/>
PhoneNumber	<input type="text" value="84913745789"/>
Email	<input type="text" value="nghiemn04@gmail.com"/>
Full name	<input type="text" value="Nguyễn Nghiệm"/>
Photo	<input type="text" value="nghiemn.gif"/>
<input type="button" value="Register"/>	

```
public class ApplicationUser : IdentityUser
{
    [StringLength(50), Required]
    public string FullName { get; set; }

    [StringLength(50)]
    public string Photo { get; set; }

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

```
var user = new ApplicationUser()
{
    UserName = model.UserName,
    Email = model.Email,
    FullName = model.FullName,
    PhoneNumber = model.PhoneNumber,
    Photo = model.Photo
};

var result = await UserManager.CreateAsync(user, model.Password);
```

SONGLONG.Mvc5A...dbo.AspNetUsers			
	Column Name	Data Type	Allow Nulls
🔑	Id	nvarchar(128)	<input type="checkbox"/>
	FullName	nvarchar(50)	<input type="checkbox"/>
	Photo	nvarchar(50)	<input checked="" type="checkbox"/>
	Email	nvarchar(256)	<input checked="" type="checkbox"/>
	EmailConfirmed	bit	<input type="checkbox"/>
	PasswordHash	nvarchar(MAX)	<input checked="" type="checkbox"/>
	SecurityStamp	nvarchar(MAX)	<input checked="" type="checkbox"/>
	PhoneNumber	nvarchar(MAX)	<input checked="" type="checkbox"/>
	PhoneNumberConfirmed	bit	<input type="checkbox"/>
	TwoFactorEnabled	bit	<input type="checkbox"/>
	LockoutEndDateUtc	datetime	<input checked="" type="checkbox"/>
	LockoutEnabled	bit	<input type="checkbox"/>
	AccessFailedCount	int	<input type="checkbox"/>
	UserName	nvarchar(256)	<input type="checkbox"/>

Kích hoạt qua tài khoản email:

The diagram illustrates the email activation process flow. It starts with a registration form on the left, followed by an email inbox showing a welcome message, and ends with a login form on the right. Orange arrows indicate the flow from registration to the inbox and then to the login form.

Registration Form:

User name	nnghiem
Password
Confirm password
PhoneNumber	84913745789
Email	nnghiemn04@gmail.com
Full name	Nguyễn Nghiệm
Photo	nnghiemn.gif
<button>Register</button>	

Email Inbox:

COMPOSE | Welcome mail | **Inbox** x

Inbox
Starred

Web Master <kentphp1@gmail.com>
to me ▾
[Activate](#)

Login Form:

User name	<input type="text"/>
Password	<input type="password"/>
<input type="checkbox"/> Remember me?	
<button>Log in</button>	

1. Đăng ký

```
UserManager.CreateAsync(user, model.Password);
```

2. Gửi mail

```
XMail.Send(to, subject, body);
```

3. Kích hoạt

```
var user = UserManager.FindByName(userName);  
user.Activated = true;  
UserManager.Update(user);
```

4. Đăng nhập

```
if (user.Activated)  
{  
    await SignInAsync(user, model.RememberMe);  
    return RedirectToLocal(returnUrl);  
}  
else  
{  
    ModelState.AddModelError("", "The account hasn't been activated !");  
}
```

Nhận lại mật khẩu:

ForgotPassword

User Name	<input type="text" value="nnghiem2015"/>
Email Registered	<input type="text" value="nnghiemn@fpt.edu.vn"/>
<input type="button" value="Get Password"/>	

```
var token = Guid.NewGuid().ToString();  
XMail.Send(to, subject, token);  
  
userManager.RemovePassword(user.Id);  
userManager.AddPassword(user.Id, token);
```

Mã token

ResetPassword

User Name	<input type="text" value="nnghiem2015"/>
Reset Token	<input type="text" value="96bd1d4d-4204-4b27-be1b-6b04146f7a"/>
New Password	<input type="password" value="....."/>
Confirm Password	<input type="password" value="....."/>
<input type="button" value="Reset Password"/>	

```
userManager.ChangePassword(user.Id, token, newPassword);
```

Quản lý người dùng:

User Edition User List

User Name

Password

FullName

Email

Phone Number

Photo

☐ Activated

- ❑ UserManager.Users
- ❑ UserManager.Create(user, pwd)
- ❑ UserManager.FindByName(id);
- ❑ UserManager.Update(user)
- ❑ UserManager.Delete(user)

User Edition		User List				
User Name	Password	Full Name	Email	Phone Number	Photo	Activated?
lththao	*****	Lê Thị Hương Thảo	lycato@gmail.com	0918355888		No Edit
nnghiem2015	*****	Nguyễn Nghiệm	nghiemn@fpt.edu.vn	84913745789	nghiemn.gif	Yes Edit

Quản lý vai trò:

Role Edition

Role List

Id	Name	
57bf48ef-055a-4ede-8156-82a0d95b4e0f	Administrator	Edit
ff6760b8-303f-42ea-9505-9b1db67dc69c	User	Edit

Role Edition

Role List

Role Id

e3da8a05-fe53-429b-9bc7-224068695ff

Role Name

Insert

Update

Delete

New

Quyền chứng thực (Authorization)

User Name	Full Name	Roles
lththao	Lê Thị Hương Thảo	<input type="checkbox"/> Administrator <input checked="" type="checkbox"/> User
nnghiem2015	Nguyễn Nghiệm	<input checked="" type="checkbox"/> Administrator <input type="checkbox"/> User

❑ Khai báo

- ~~✗~~ `[Authorize(Roles = "Administrator, User")]`
- ~~✗~~ `[Authorize(Users = "nnghiem2015, lththao")]`
- ~~✗~~ `[Authorize(Users = "lththao", Roles="User")]`

❑ Lập trình

- ~~✗~~ `User.IsInRole(role)`
- ~~✗~~ `User.Identity.Name`

Kiểm soát chức năng:

```
if (User.IsInRole("Director"))  
{  
    // Công việc của giám đốc  
}
```

Lập trình theo Role

Khai báo theo Role và User

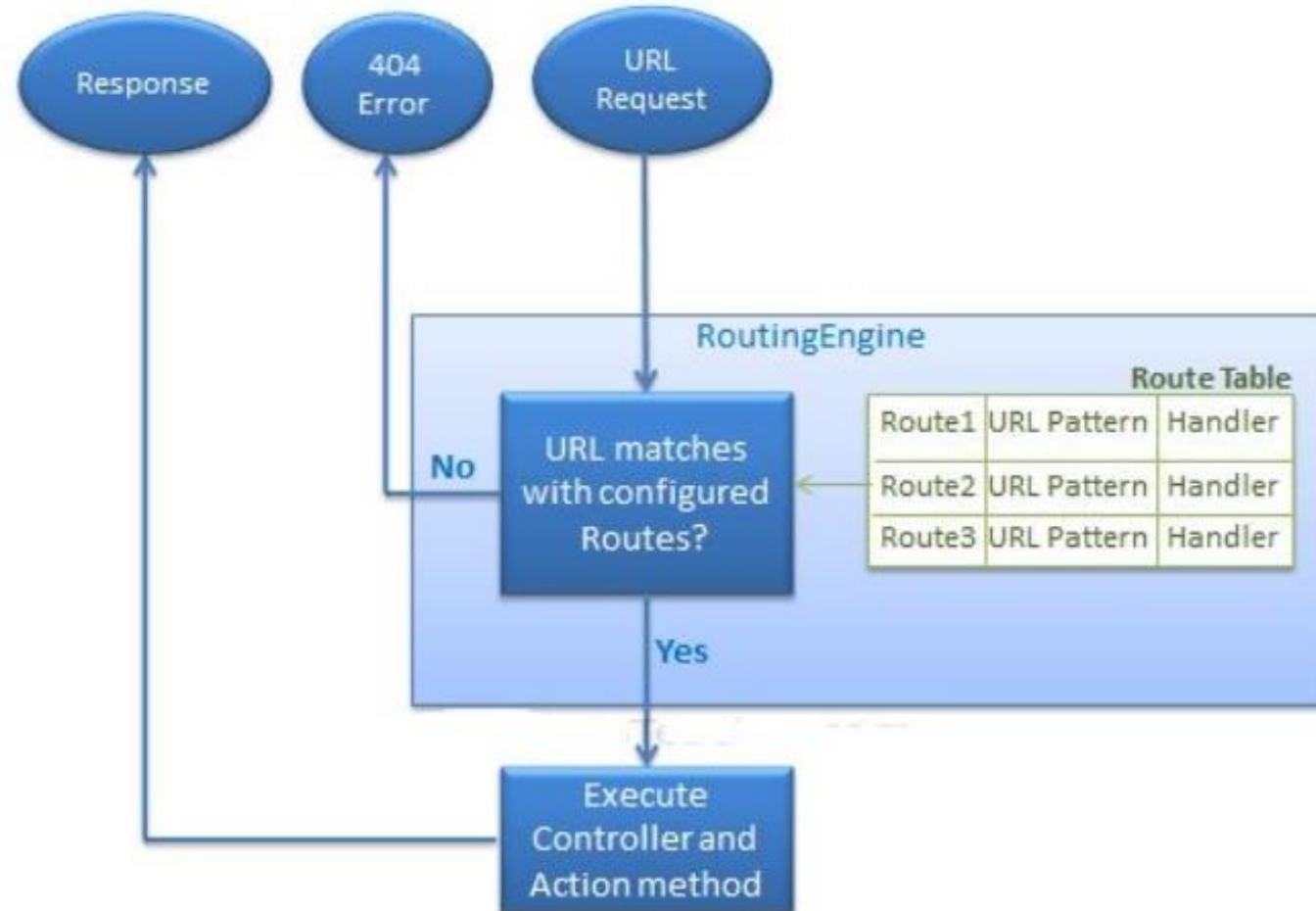
```
[Authorize]  
public class CRUDController : Controller  
{  
    public ActionResult Index(){...}  
    public ActionResult Insert(model){...}  
    public ActionResult Update(model){...}  
    public ActionResult Edit(id){...}  
  
    [Authorize(Roles="Admin", Users="vip")]  
    public ActionResult Delete(id){...}  
}
```

```
if (User.Identity.Name == "nghiemn")  
{  
    // Công việc của tài khoản nghiemn  
}
```

Lập trình theo User

2. Định tuyến (Routing)

- Route (Tuyến) xác định mẫu URL và thông tin xử lý.



Cấu hình tuyến trong ASP.NET MVC

- Mỗi ứng dụng phải cấu hình ít nhất một tuyến đã được cấu hình mặc định.
- Cấu hình tuyến trong RouteConfig.cs (thư mục App_Start)

```
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 }
        );
    }
}
```

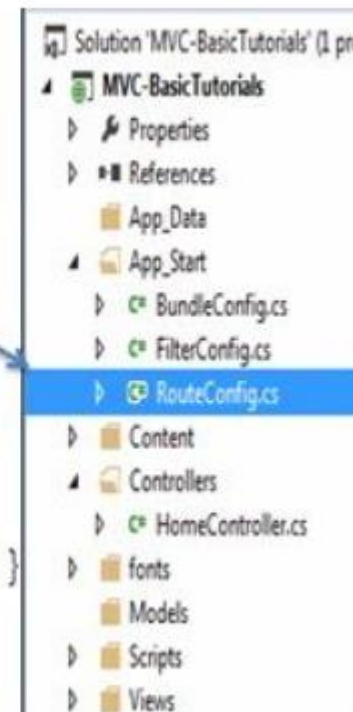
Route to ignore

Route name

URL Pattern

Defaults for Route

RouteConfig.cs



Giải thích:

- Tuyến được cấu hình bằng phương thức mở rộng:

`MapRoute()` của `RouteCollection`, trong đó tên tuyến là "Default", mẫu URL của tuyến là `{controller}/{action}/{id}` và giá trị mặc định cho tham số controller, phương thức hành động và tham số id.

Giá trị tham số mặc định chỉ định controller, phương thức hành động hoặc giá trị của tham số id sẽ được sử dụng nếu chúng không tồn tại trong URL yêu cầu gửi đến.

Ví dụ nhiều tuyến:

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

        routes.MapRoute(
            name: "Student",
            url: "student/{id}",
            defaults: new { controller = "Student", action = "Index" }
        );

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

Copy

Đăng ký tuyến trong ASP.NET MVC

- Đăng ký tuyến vào sự kiện `Application_Start()` trong `Global.asax`:

```
public class MvcApplication : System.Web.HttpApplication
{
    protected void Application_Start()
    {
        RouteConfig.RegisterRoutes(RouteTable.Routes);
    }
}
```