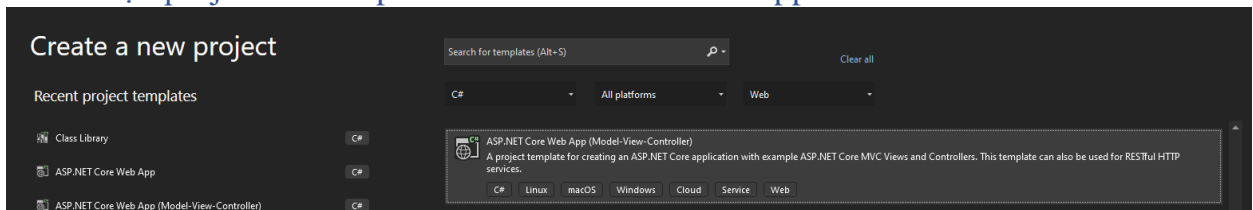


Contents

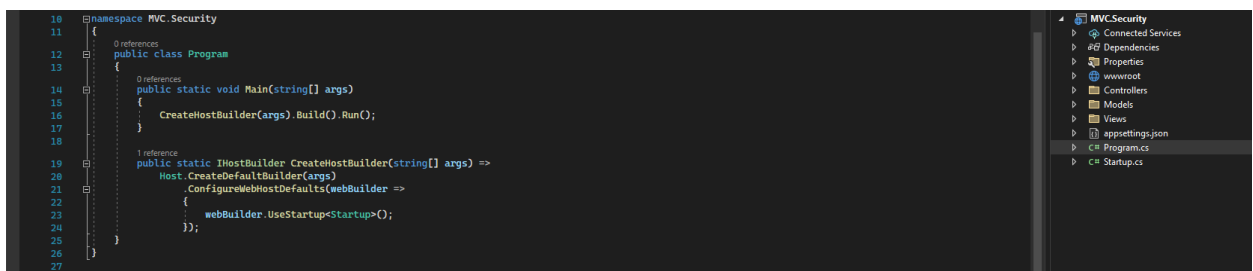
I. Authentication và Authorization cơ bản trong ASP.NET core MVC.....	1
1. Tạo project với template ASP.NET Core Web App – MVC.....	1
2. App.useAuthentication() – with Bearer token	2
3. App.useAuthorization().....	3
4. Debug demo:.....	4
II. ASP.NET Core Authentication with Cookie:.....	5
III. Một số vấn đề khác khi review source code.....	8
1. Entity Framework (EF).....	8
2. Services addControllers(), addControllersWithViews(), addRazorPages(), addMvc()	9
3. .NET Ecosystem: .Net / .Net Core / .Net Standard	9
4. .NET SDK vs .NET Runtime.....	10
5. Target framework	10

I. Authentication và Authorization cơ bản trong ASP.NET core MVC

1. Tạo project với template ASP.NET Core Web App – MVC



- Cấu trúc thư mục sẽ có những phần chính như wwwroot (web root folder), Models, Views, Controllers.



- Project bắt đầu chạy từ hàm Main và chạy đến class Startup.

```

38 public void Configure(IApplicationBuilder app, IWebHostEnvironment env)
39 {
40     if (env.IsDevelopment())
41     {
42         app.UseDeveloperExceptionPage();
43     }
44     else
45     {
46         app.UseExceptionHandler("/Home/Error");
47         // The default HSTS value is 30 days. You may want to change this for production scenarios, see https://aka.ms/aspnetcore-hsts.
48         app.UseHsts();
49     }
50     app.UseHttpsRedirection();
51     app.UseStaticFiles();
52
53     app.UseRouting();
54
55     app.UseAuthentication();
56     app.UseAuthorization();
57
58     app.UseEndpoints(endpoints =>
59     {
60         endpoints.MapControllerRoute(
61             name: "default",
62             pattern: "{controller=Home}/{action=Index}/{id?}");
63     });
64 }

```

- Trong hàm Configure cấu hình sử dụng các middleware thêm vào pipeline request.
 - o app.UseHttpsRedirection() à request tự động chuyển sang https
 - o app.UseStaticFiles() à middleware xử lý các request gọi đến file tĩnh, file không thực thi được như html, txt,...
 - o app.UseRouting() sẽ định tuyến cho các api request chạy đến các endpoint được map ở bên trong middleware UseEndpoints
 - o Ví dụ:

```

15 namespace IOIT.Identity.Api.Controllers.ApiApp
16 {
17     [Route("api/app/[controller]")]
18     [ApiController]
19     public class UserController : BaseController
20     {
21         [HttpPost("login")]
22         public async Task<IActionResult> Login([FromBody] LoginUserAppQuery command)
23         {
24             var resData = await _mediator.Send(command);
25
26             return Res(new DefaultResponse().Success(
27                 resData,
28                 ApiConstants.MessageResource.ACCTION_SUCCESS,
29                 ApiConstants.StatusCode.Success200
30             ));
31     }
32 }

```

➔ Api login là: /api/app/user/login. Class UserController thì sẽ bỏ Controller đi, đến action login.

2. App.useAuthentication() – with Bearer token

- Khi sử dụng middleware UseAuthentication() ta phải đăng ký service và các handle trong hàm ConfigureServices trong class Startup. Ví dụ sử dụng token Bear cho việc xác thực.

```

249 string domain = Configuration["AppSettings:JwtIssuer"];
250 var authenticationProviderKey = "TestKey";
251 JwtSecurityTokenHandler.DefaultInboundClaimTypeMap.Clear();
252 services.AddAuthentication(options =>
253 {
254     options.DefaultAuthenticateScheme = authenticationProviderKey;
255     options.DefaultScheme = JwtBearerDefaults.AuthenticationScheme; // Bear
256     options.DefaultChallengeScheme = JwtBearerDefaults.AuthenticationScheme; // Bear
257 })
258 .AddJwtBearer(authenticationProviderKey, cfg =>
259 {
260     cfg.RequireHttpsMetadata = false;
261     cfg.SaveToken = true;
262     cfg.TokenValidationParameters = new TokenValidationParameters
263     {
264         ValidIssuer = domain,
265         ValidAudience = domain,
266         IssuerSigningKey = new SymmetricSecurityKey(Encoding.UTF8.GetBytes(Configuration["AppSettings:JwtKey"])),
267         ClockSkew = TimeSpan.Zero // remove delay of token when expire
268     };
269 });

```

- Các request sẽ đi qua middleware này, token được validate, thông tin người dùng ở context hiện tại được lưu trong thuộc tính User, đặc biệt quan tâm đến trường isAuthenticated. Nếu token invalid biến isAuthenticated sẽ là false

The screenshot shows the Visual Studio IDE with the `CategoryController.cs` file open. The code defines a `GetByPage` action that uses `[Authorize]` and `[Route("GetByPage")]`. The action body retrieves the user identity and claims, and then sets the `isAuthenticated` property of the `User` object.

The Autos window below the code shows the state of the `User` object. The `isAuthenticated` property is highlighted in blue and has a value of `true`.

Name	Value
Microsoft.AspNetCore.Mvc.ControllerBase.User.get returned	(System.Security.Claims.ClaimsPrincipal)
System.Security.Claims.ClaimsPrincipal.Identity.get returned	(System.Security.Claims.ClaimsIdentity)
(ClaimsIdentity)User.Identity	(System.Security.Claims.ClaimsPrincipal)
User	(System.Security.Claims.ClaimsPrincipal)
Claims	(System.Security.Claims.ClaimsPrincipal.<get_Claims>d_21)
CustomSerializationData	null
Identities	Count = 1
Identity	(System.Security.Claims.ClaimsIdentity)
Actor	null
AuthenticationType	"AuthenticationTypes.Federation"
BootstrapContext	null
Claims	Count = 22
CustomSerializationData	null
isAuthenticated	true
Label	null
Name	"ez"
NameClaimType	"http://schemas.xmlsoap.org/ws/2005/05/identity/claims/name"
RoleClaimType	"http://schemas.microsoft.com/ws/2008/06/identity/claims/role"
Static members	

3. App.useAuthorization()

- Khi sử dụng middleware `UseAuthorization()` và sử dụng thuộc tính `[Authorize]` trên Controller hoặc trên action, server sẽ kiểm tra biến `isAuthenticated` trong User identity. Nếu chưa xác thực thì server trả về luôn response `UN_AUTHORIZE` (Lưu ý: nếu sử dụng thuộc tính `[Authorize]` nhưng không khai báo middleware `UseAuthorization()` thì server sẽ không kiểm tra người dùng đã authenticate hay chưa)
- Tùy vào từng request mà có thể đi qua hết các middleware hoặc đi qua 1 vài hoặc ko đi qua middleware nào.
 - o Request sẽ đi qua các middleware cho đến khi có Reponse tương ứng được trả về.

- Các Request có thể đi qua các middleware mà ko có sự tác động gì đến HttpContext vì request ko gọi đến Handler nào trong middleware đó.
- 1 request bị tác động khi đi qua 1 middleware khi request đó có action gọi đến các handler của middleware.
- Ví dụ khi sử dụng thuộc tính [Authorize] thì request đi qua middleware UseAuthorization và gọi đến AuthorizationHandler; khi claim user identity request sẽ đi qua middleware UseAuthentication và gọi đến AuthenticationHandler (verify token, cookies, claim user identity).
- Các Request thường bị rẽ nhánh để đến middleware cuối (trả về Response) khi đi qua middleware useEndpoints() (hoặc các middleware có chức năng map request đến action trong api controller)
- Đọc thêm middleware trong ASP.NET core: <https://xuanthulab.net/asp-net-core-cao-middleware-va-dang-ky-vao-pipeline-cua-ung-dung-web-c-csharp.html>

4. Debug demo:

Request

PrettyRawHex

⌵

↵

☰

```

1 GET /api/cms/Category/GetByPage?page=4&page_size=10&query=1=1&order_by=Id%20Desc HTTP/2
2 Host: localhost:44308
3 Sec-Ch-Ua: "Chromium";v="103", ".Not/A) Brand";v="99"
4 Accept: text/plain
5 Sec-Ch-Ua-Mobile: ?0
6 Authorization: Bearer faketoken
7 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
  Chrome/103.0.5060.53 Safari/537.36
8 Sec-Ch-Ua-Platform: "Windows"
9 Sec-Fetch-Site: same-origin
10 Sec-Fetch-Mode: cors
11 Sec-Fetch-Dest: empty
12 Referer: https://localhost:44308/swagger/index.html
13 Accept-Encoding: gzip, deflate
14 Accept-Language: en-US,en;q=0.9
15

```

- Khi gửi 1 request với invalid token à người dùng không được xác thực isAuthenticated = false, identity = null

The screenshot shows a C# controller action in Visual Studio. The code is as follows:

```

118 // [Authorize]
119 [Route("GetByPage")]
120 [HttpGet]
121 public async Task<Output<List<ResGetCategoryById>>> GetByPage([FromQuery] GetCategoryByPagingQuery command)
122 {
123     var identity = (ClaimsIdentity)User.Identity;
124     string access_key = identity.Claims.Where(c => c.Type == "AccessKey").Select(c => c.Value).SingleOrDefault();
125     int userId = int.Parse(identity.Claims.Where(c => c.Type == "UserId").Select(c => c.Value).SingleOrDefault());
126     var type = identity.Claims.Where(c => c.Type == "Type").Select(c => c.Value).SingleOrDefault() ?? "0";
127     var projectId = identity.Claims.Where(c => c.Type == "ProjectId").Select(c => c.Value).SingleOrDefault() ?? "-1";
128     if (!CheckRoleByCode(access_key, functionCode, (int)AppEnum.Action.VIEW))
129     {
130     }
131 }

```

The debugger state shows the following variables:

Name	Value
Microsoft.AspNetCore.Mvc.ControllerBase.User.get returned	{System.Security.Claims.ClaimsPrincipal}
System.Security.Claims.ClaimsPrincipal.Identity.get returned	{System.Security.Claims.ClaimsIdentity}
(ClaimsIdentity)User.Identity	{System.Security.Claims.ClaimsIdentity}
User	{System.Security.Claims.ClaimsPrincipal}
User.Identity	{System.Security.Claims.ClaimsIdentity}
access_key	null
identity	{System.Security.Claims.ClaimsIdentity}
Actor	null
AuthenticationType	null
BootstrapContext	null
Claims	Count = 0
CustomSerializationData	null
IsAuthenticated	false
Label	null
Name	null
NameClaimType	"http://schemas.xmlsoap.org/ws/2005/05/identity/claims/name"
RoleClaimType	"http://schemas.microsoft.com/ws/2008/06/identity/claims/role"
Static members	
Non-Public members	

- Khi cho request này đi qua middleware Authorization thì sẽ bị chặn luôn (dòng 118).

The screenshot shows the same C# controller action as above, but with a red dot on line 118, indicating the point where the request is blocked by the Authorization middleware.

The screenshot shows a browser's developer tools with the following request and response:

Request			Response			
	Pretty	Raw		Pretty	Raw	Hex
1	GET	/api/cms/Category/GetByPage?page=4&page_size=10&query=1=1&order_by=Id&20Desc	HTTP/2	1	HTTP/2 401 Unauthorized	
2	Host:	localhost:44308		2	Server:	Microsoft-IIS/10.0
3	Sec-Ch-Ua:	"Chromium";v="103", ".Not/A) Brand";v="99"		3	Www-Authenticate:	Bearer error="invalid_token"
4	Accept:	text/plain		4	X-Powered-By:	ASP.NET
5	Sec-Ch-Ua-Mobile:	70		5	Date:	Thu, 30 Jun 2022 11:53:56 GMT
6	Authorization:	Bearer faktoken		6		
7	User-Agent:	Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/103.0.5060.53 Safari/537.36		7		
8	Sec-Ch-Ua-Platform:	"Windows"				
9	Sec-Fetch-Site:	same-origin				
10	Sec-Fetch-Mode:	cors				
11	Sec-Fetch-Dest:	empty				
12	Referer:	https://localhost:44308/swagger/index.html				
13	Accept-Encoding:	gzip, deflate				
14	Accept-Language:	en-US,en;q=0.9				
15						
16						

II. ASP.NET Core Authentication with Cookie:

- Khai báo Middleware và đăng ký Service trong Startup.cs

```

var cookiePolicyOptions = new CookiePolicyOptions
{
    MinimumSameSitePolicy = SameSiteMode.None,
    Secure = CookieSecurePolicy.Always,
    HttpOnly = Microsoft.AspNetCore.CookiePolicy.HttpOnlyPolicy.Always,
};

app.UseHttpsRedirection();
app.UseStaticFiles();

app.UseRouting();
app.UseAuthentication();
app.UseCookiePolicy(cookiePolicyOptions);
app.UseAuthorization();

```

```

services.AddAuthentication(CookieAuthenticationDefaults.AuthenticationScheme)
    .AddCookie(options =>
    {
        options.ExpireTimeSpan = TimeSpan.FromMinutes(20);
        options.SlidingExpiration = true;
        options.AccessDeniedPath = "/Forbidden/";
    });

```

- Trong api login

```

[HttpPost("login")]
0 references
public async Task<ActionResult> Login(string email, string password)
{
    if (ModelState.IsValid)
    {
        var user = 1; //await AuthenticateUser(Input.Email, Input.Password);
        if (user == null)
        {
            ViewBag.errorr = "Login Failed";
            return RedirectToAction("Login");
        }

        var claims = new List<Claim>
        {
            new Claim(ClaimTypes.Name, "nguyennngocdoanh1998@gmail.com"), //user.Email,
            new Claim("FullName", "Nguyen Ngoc Doanh"), //user.FullName,
            new Claim(ClaimTypes.Role, "Administrator"),
        };

        var claimsIdentity = new ClaimsIdentity(
            claims, CookieAuthenticationDefaults.AuthenticationScheme);

        var authProperties = new AuthenticationProperties
        {
            ExpiresUtc = DateTimeOffset.UtcNow.AddMinutes(10),
            IsPersistent = true,
        };

        await HttpContext.SignInAsync(
            CookieAuthenticationDefaults.AuthenticationScheme,
            new ClaimsPrincipal(claimsIdentity),
            authProperties);
    }
}

```

- Khi gửi request post login à server trả về cookies

Request

PrettyRawHex

1

POST /api/login HTTP/2

2

Host: localhost:44335

3

Sec-Ch-Ua: "Chromium";v="103", ".Not/A)Brand";v="99"

4

Sec-Ch-Ua-Mobile: ?0

5

Sec-Ch-Ua-Platform: "Windows"

6

Upgrade-Insecure-Requests: 1

7

User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/103.0.5060.53 Safari/537.36

8

Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9

9

Sec-Fetch-Site: none

10

Sec-Fetch-Mode: navigate

11

Sec-Fetch-User: ?1

12

Sec-Fetch-Dest: document

13

Accept-Encoding: gzip, deflate

14

Accept-Language: en-US,en;q=0.9

15

Content-Length: 0

16

17

Response

PrettyRawHexRender

1

HTTP/2 200 OK

2

Cache-Control: no-cache, no-store

3

Pragma: no-cache

4

Content-Type: text/html; charset=utf-8

5

Expires: Thu, 01 Jan 1970 00:00:00 GMT

6

Vary: Accept-Encoding

7

Server: Microsoft-IIS/10.0

8

Set-Cookie: .AspNetCore.Cookies=CfDj80WUDEy7yHtAo2G3WAlVwrnzeXU4cfiLpDMeKQD_15VDRjbZrHSLZrFQN32K7fPz7GyGIexnJMY-vkLkDWTovKYrOu5thVsDb800y2GzKr-xT7gZclWqrM9-YlFUBFoeCkeMxIlfcNkCRyxpbghp_juBk_6GyqW0NggljGHqYDs0_5zL3s3lFmVpNr4hUkh5SDtu0u0RBgtvDElGLii76RZgz_vlHrLZLxq69bHQ_TnnHyranBcheltUNSULUjmEpoq2VSS8zQteC-W8YxAuB8RkftxiPRv0GeZiDXUhpEC3LU3gMG6016jurjVFWXlE0QwBJKEheE06nIfUo1Qc4FJhBIyD-4xVilSBloktfwBuXgwrLhVaNZhV-WdiLZY-oPh00auu1BhgtvwrWdHj7opmIY_WGuTcNlf-flqYcpFmzLAV-AUXRag-Tj-c0rs6rQ05eIqmaJayZcBUQnuUS-pew3M80rzkEjcUoYDjLoSB8nuRLG_u2mTu-iK9xYXzQ; expires=Sat, 09 Jul 2022 09:08:21 GMT; path=/; secure; samesite=lax; httponly

9

Set-Cookie: .AspNetCore.Antiforgery.G-pwIocZ2YU=CfDj80WUDEy7yHtAo2G3WAlVwrnzeXU4cfiLpDMeKQD_15VDRjbZrHSLZrFQN32K7fPz7GyGIexnJMY-vkLkDWTovKYrOu5thVsDb800y2GzKr-xT7gZclWqrM9-YlFUBFoeCkeMxIlfcNkCRyxpbghp_juBk_6GyqW0NggljGHqYDs0_5zL3s3lFmVpNr4hUkh5SDtu0u0RBgtvDElGLii76RZgz_vlHrLZLxq69bHQ_TnnHyranBcheltUNSULUjmEpoq2VSS8zQteC-W8YxAuB8RkftxiPRv0GeZiDXUhpEC3LU3gMG6016jurjVFWXlE0QwBJKEheE06nIfUo1Qc4FJhBIyD-4xVilSBloktfwBuXgwrLhVaNZhV-WdiLZY-oPh00auu1BhgtvwrWdHj7opmIY_WGuTcNlf-flqYcpFmzLAV-AUXRag-Tj-c0rs6rQ05eIqmaJayZcBUQnuUS-pew3M80rzkEjcUoYDjLoSB8nuRLG_u2mTu-iK9xYXzQ; expires=Sat, 09 Jul 2022 09:08:21 GMT; path=/; secure; samesite=lax; httponly

10

X-Frame-Options: SAMEORIGIN

11

X-Powered-By: ASP.NET

12

Date: Sat, 09 Jul 2022 08:58:21 GMT

13

- Sử dụng Cookie này cho việc xác thực người dùng trong các request sau

Request

PrettyRawHex

1

GET /api/register HTTP/2

2

Host: localhost:44335

3

Sec-Ch-Ua: "Chromium";v="103", ".Not/A)Brand";v="99"

4

Sec-Ch-Ua-Mobile: ?0

5

Sec-Ch-Ua-Platform: "Windows"

6

Upgrade-Insecure-Requests: 1

7

Cookie: .AspNetCore.Cookies=CfDj80WUDEy7yHtAo2G3WAlVwrnzeXU4cfiLpDMeKQD_15VDRjbZrHSLZrFQN32K7fPz7GyGIexnJMY-vkLkDWTovKYrOu5thVsDb800y2GzKr-xT7gZclWqrM9-YlFUBFoeCkeMxIlfcNkCRyxpbghp_juBk_6GyqW0NggljGHqYDs0_5zL3s3lFmVpNr4hUkh5SDtu0u0RBgtvDElGLii76RZgz_vlHrLZLxq69bHQ_TnnHyranBcheltUNSULUjmEpoq2VSS8zQteC-W8YxAuB8RkftxiPRv0GeZiDXUhpEC3LU3gMG6016jurjVFWXlE0QwBJKEheE06nIfUo1Qc4FJhBIyD-4xVilSBloktfwBuXgwrLhVaNZhV-WdiLZY-oPh00auu1BhgtvwrWdHj7opmIY_WGuTcNlf-flqYcpFmzLAV-AUXRag-Tj-c0rs6rQ05eIqmaJayZcBUQnuUS-pew3M80rzkEjcUoYDjLoSB8nuRLG_u2mTu-iK9xYXzQ

8

User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/103.0.5060.53 Safari/537.36

9

Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9

10

Sec-Fetch-Site: none

11

Sec-Fetch-Mode: navigate

12

Sec-Fetch-User: ?1

13

Sec-Fetch-Dest: document

14

Accept-Encoding: gzip, deflate

15

Accept-Language: en-US,en;q=0.9

16

17

- Kết quả: người dùng được xác thực, claim được thông tin User đã login

▲ User	{System.Security.Claims.ClaimsPrincipal}
▶ Claims	{System.Security.Claims.ClaimsPrincipal.<get_Claims>d_22}
CustomSerializationData	null
Identities	Count = 1
▲ Identity	{System.Security.Claims.ClaimsIdentity}
▶ Actor	null
AuthenticationType	"Cookies" 🔍
BootstrapContext	null
Claims	{System.Security.Claims.ClaimsIdentity.<CombinedClaimsIterator>d_38}
CustomSerializationData	null
ExternalClaims	Count = 0
IsAuthenticated	true
Label	null
Name	"nguyenngocdoanh1998@gmail.com" 🔍
NameClaimType	"http://schemas.xmlsoap.org/ws/2005/05/identity/claims/name" 🔍
RoleClaimType	"http://schemas.microsoft.com/ws/2008/06/identity/claims/role" 🔍

▲ Claims	{System.Security.Claims.ClaimsIdentity.<CombinedClaimsIterator>d_38}
System.Collections.Gen...	null
System.Collections.IEnu...	null
Results View	Expanding the Results View will enumerate the IEnumerable
[0]	{http://schemas.xmlsoap.org/ws/2005/05/identity/claims/name: nguyenngocdoanh1998@gmail.com}
[1]	{FullName: Nguyen Ngoc Doanh}
[2]	{http://schemas.microsoft.com/ws/2008/06/identity/claims/role: Administrator}

- Reference:
 - a. <https://docs.microsoft.com/en-us/aspnet/core/security/authentication/cookie?view=aspnetcore-6.0>
 - b. <https://github.com/doanhnn/MVC.Api.git>

III. Một số vấn đề khác khi review source code

1. Entity Framework (EF)

```

namespace MVC.API.Entities
{
    3 references | 0 changes | 0 authors, 0 changes
    public class DB_Entities : DbContext
    {
        1 reference | 0 changes | 0 authors, 0 changes
        public DB_Entities() : base("Server=localhost\\SQLEXPRESS;Database=master;Trusted_Connection=True;") { }

        2 references | 0 changes | 0 authors, 0 changes
        public DbSet<User> Users { get; set; }

        0 references | 0 changes | 0 authors, 0 changes
        protected override void OnModelCreating(DbModelBuilder modelBuilder)
        {
            modelBuilder.Entity<User>().ToTable("Users");
            modelBuilder.Conventions.Remove<PluralizingTableNameConvention>();
            base.OnModelCreating(modelBuilder);
        }
    }
}

```

- `using System.Data.Entity;`
- `using System.Data.Entity.ModelConfiguration.Conventions;`
- Hàm khởi tạo sẽ kết nối đến SQLServer, database master sử dụng connectionString `"Server=localhost\\SQLEXPRESS;Database=master;Trusted_Connection=True;"`
- `DbSet<User> Users` → bảng Users có model User
- Tạo 1 đối tượng `DB_Entities` và sử dụng


```

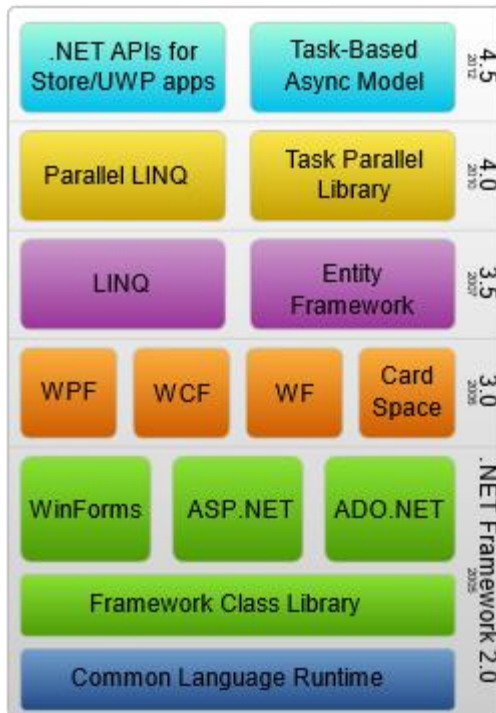
namespace MVC.API.Controllers
{
    [ApiController]
    [Route("/[controller]")]
    0 references | 0 changes | 0 authors, 0 changes
    public class ApiController : Controller
    {
        private DB_Entities _db = new DB_Entities();

        //[Authorize]
        [HttpGet("register")]
        0 references | 0 changes | 0 authors, 0 changes
        public ActionResult Register()...

        [HttpPost("register")]
        0 references | 0 changes | 0 authors, 0 changes
        public ActionResult Register(User _user)
        {
            if (ModelState.IsValid)
            {
                var check = _db.Users.FirstOrDefault(s => s.Email == _user.Email);
                if(check == null)
                {
                    _db.Configuration.ValidateOnSaveEnabled = false;
                    _db.Users.Add(_user);
                    _db.SaveChanges();
                    return RedirectToAction("Index");
                }
                else
                {
                    ViewBag.error = "Email already exists";
                    return View();
                }
            }
            return View();
        }
    }
}

```

2. Services addControllers(), addControllersWithViews(), addRazorPages(), addMvc()
 - <https://dotnettutorials.net/lesson/difference-between-addmvc-and-addmvccore-method/>
3. .NET Ecosystem: .Net / .Net Core / .Net Standard
 - .NET Framework là 1 nền tảng lập trình và cũng là môi trường để thực thi ứng dụng trên Windows.



- 2 thành phần chính trong .NET Framework là Framework Class Library (FCL) và Common Language Runtime (CLR)
- FCL là các thư viện hỗ trợ sẵn, lập trình viên chỉ cần học các sử dụng và dùng lại để xây dựng app
- CLR như 1 sandbox là môi trường chạy ứng dụng. CLR còn hỗ trợ 1 số tính năng như manager security, manager memory, handler exception, garbage collection,...
- Khi compiler app, IDE sẽ dịch ra 1 ngôn ngữ trung gian - Microsoft Intermediate Language (MSIL). Khi run app, CLR sẽ dịch MSIL ra mã máy để thực thi.
- WinForms: windows desktop app
- ASP.NET: web app
- ADO.NET: connect DB (Oracle/MySQL)

- .NET Core là 1 framework độc lập với .NET nhưng được phát triển từ .NET, với mục đích có thể chạy trên đa nền tảng (Windows/Linux/MacOS...)
 - o Không hỗ trợ winforms
 - o CoreCLR
- .NET Standard là 1 bộ đặc tả kỹ thuật (specification) của những API chung trong các .NET framework. Có thể hiểu .NET Standard là dạng Interface còn .NET, .NET Core, Xamarin/Mono là các Implementation khác nhau
 - o .NET, .NET Core chạy trên máy Desktop
 - o Xamarin/Mono chạy trên Android/iOS

4. .NET SDK vs .NET Runtime

- .NET SDK: là những công cụ hỗ trợ cho việc build/run app (CLI, IDE Visual Studio)
- .NET Runtime: là những thứ cần cho việc run app (đã bao gồm trong .NET SDK)

5. Target framework

```

MVC.API.csproj ➤ ✕
<Project Sdk="Microsoft.NET.Sdk.Web">
  <PropertyGroup>
    <TargetFramework>netcoreapp5.0</TargetFramework>
  </PropertyGroup>
</Project>

```

- Examples:
 - o .NET Standard → netstandard2.1
 - o .NET Framework → net48
 - o .NET 5+ (and .NET Core) → netcoreapp3.1, net5.0/netcoreapp5.0

- <https://docs.microsoft.com/en-us/dotnet/standard/frameworks>