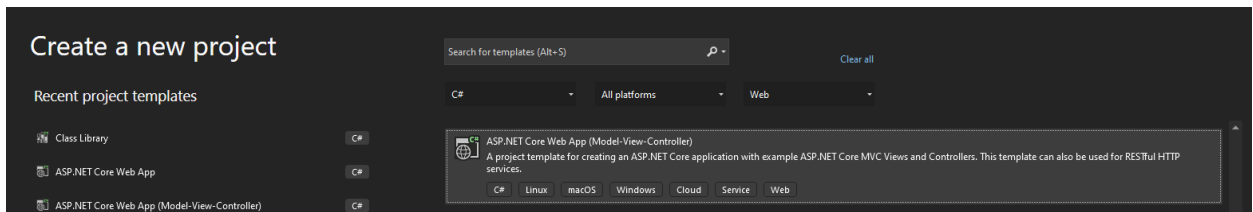
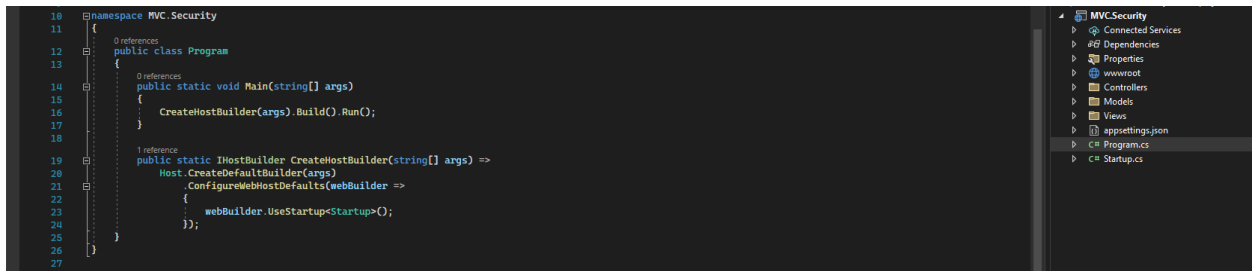


- Tạo một project với template ASP.NET Core Web App – MVC



- 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.



- Trong hàm Configure cấu hình sử dụng các middleware thêm vào pipeline request.
  - o `app.UseHttpsRedirection()` → request tự động chuyển sang https
  - o `app.UseStaticFiles()` → 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.

- 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` method that checks for authentication using `ClaimsIdentity`. The `Autos` window at the bottom displays the state of the `User` identity, including `Claims`, `Actor`, and `AuthenticationType`.

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}
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	

- 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. Điều kiện để 1 request đi qua 1 middleware là request đó có action gọi đến các handler của middleware. Ví dụ khi sử dụng thuộc tính `[Authorize]` thì request mới đi qua middleware `UseAuthorization`; khi claim user identity request sẽ đi qua middleware `UseAuthentication`.
- Debug demo:

The screenshot shows the raw request details in a web browser. The request is a GET to `/api/cms/Category/GetByPage` with various query parameters and headers.

```

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 → người dùng không được xác thực isAuthenticated = false, identity = null

```

118 // [Authorize]
119 [Route("GetByPage")]
120 [HttpGet]
121 public async Task<OutputList<ResGetCategoryId>>> 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
129     if (!CheckRoleByCode(access_key, functionCode, (int)AppEnum.Action.VIEW))
130     {
131     }
132 }

```

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).

```

118 // [Authorize]
119 [Route("GetByPage")]
120 [HttpGet]
121 public async Task<OutputList<ResGetCategoryId>>> 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
129     if (!CheckRoleByCode(access_key, functionCode, (int)AppEnum.Action.VIEW))
130     {
131     }
132 }

```

Request		Response			
Pretty	Raw	Hex	Render		
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: ?0				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					

- 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 → server trả về session

Request				Response			
Pretty	Raw	Hex		Pretty	Raw	Hex	Render
1	POST /api/login HTTP/2			1	HTTP/2 200 OK		
2	Host: localhost:44335			2	Cache-Control: no-cache, no-store		
3	Sec-Ch-Ua: "Chromium";v="103", ".Not/A)Brand";v="99"			3	Pragma: no-cache		
4	Sec-Ch-Ua-Mobile: ?0			4	Content-Type: text/html; charset=utf-8		
5	Sec-Ch-Ua-Platform: "Windows"			5	Expires: Thu, 01 Jan 1970 00:00:00 GMT		
6	Upgrade-Insecure-Requests: 1			6	Vary: Accept-Encoding		
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	Server: Microsoft-IIS/10.0		
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			8	Set-Cookie: .AspNetCore.Cookies=CfDJ80WUDBy7yHtAo2G3WAlVwmzeXU4cfLpDMeKQD_1SVDRjbZrHSLZrFQN32K7fPz7GyGIexnJMY-vh1hDWToVkyYmarOuSthVsDb800y2GzKr_xT7g2clWqrM9-YlfUBFoe2keMx1lfcNRChyxphghp_juBk_6Gyqv0NgglJGHqTDs0_SzL3s3lFmVFN:4hUhhSSDtuo0RBgtvDElGLii76R2gz_vlHrLZLxq69bHq_TnnHyranBcheItUNSUUjmaEpoq2VSS8mQteC-W8YxAuB8Rk-ftxiPRv0GGeZiDXUhpEC3LU3gMG6016jurjVFWD1E0QvBJKheE06nIfUo1Qc4FJhBlyD-4xVilsBIokt-fuBuUgwrLhVanZhv-WdiLZY-oPh00auiLBhgtvRvHdj7opmIY_WGuTcNlf-fLqYcpFmzLAV-AUxRag-Tj-c0rs6rQ05eIqmaJayZcBUQmuUS-pew3M80rzkEjcUoYDjLoSB8nuRlG_u2mTu-iKSxYXnQ; expires=Sat, 09 Jul 2022 09:08:21 GMT; path=/; secure; samesite=lax; httponly		
9	Sec-Fetch-Site: none			9	Set-Cookie: .AspNetCore.Antiforgery.G-pwIooZ2YU=CfDJ80WUDBy7yHtAo2G3WAlVwmLys188oPiPg2txH9980RkjEyRByQ6CNaD4isJFGmChHlPy6nBH7xiXisEY6ECEQh90A6PTTrlvGByvmszjqKR9-3_XI80soguheAC_KG28QstiW9s-ZKkdr33Z-UCR5oo; path=/; secure; samesite=strict; httponly		
10	Sec-Fetch-Mode: navigate			10	X-Frame-Options: SAMEORIGIN		
11	Sec-Fetch-User: ?1			11	X-Powered-By: ASP.NET		
12	Sec-Fetch-Dest: document			12	Date: Sat, 09 Jul 2022 08:58:21 GMT		
13	Accept-Encoding: gzip, deflate			13			
14	Accept-Language: en-US,en;q=0.9						
15	Content-Length: 0						
16							
17							

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

## Request

Pretty Raw Hex

```

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=
  CfDJ80WUDeY7yHtAo2G3WAlVwrmzeXU4c fiLpDMeKQD_1SVDRjb2rHSLZrFQN32K7fPz7GyG
  IexnJMY-vklkDWTovKYYmrOu5thVsDb800y2GzKr-_xT7gZclWqrM9-Y1fUBFoe2keMxI1fc
  NrCEyxphgkrp_juBk_6Gyqw0NggLjGHqYDs0_SzL3s31FmVpNr4hUkh5SDtu0u0RBgtvDE1GL
  ii76RZgz_vlHrLZLxq69bHQ_TnnHyrAnBcheltUNSULUjmEpoq2V88zQteC-W8YxAuB8RkfT
  xiPrv0GeZiDXUhpEC3LU3qMG6016jurjVFWK1EOQwBJKEheEO6nIfUo1Qc4FJhIyD-4xVil
  sBIokt fwBuXgwrLhVaNZhv-WdiLZY-oPr00auulBhgtwvRwHdj7opmIY_WGuTcNlf-flqYcp
  FmzLAV-AUXRag-Tj-c0rs6rQ05eIqmJayZcBUQnuUS-pew3M80rzkEjcUoYDjLoSB8nuR1G_
  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/w
  ebp,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	"nguyennngocdoanh1998@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.IEnum...	null
Results View	Expanding the Results View will enumerate the IEnumerable
[0]	{http://schemas.xmlsoap.org/ws/2005/05/identity/claims/name: nguyennngocdoanh1998@gmail.com}
[1]	{FullName: Nguyen Ngoc Doanh}
[2]	{http://schemas.microsoft.com/ws/2008/06/identity/claims/role: Administrator}

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