

Lab 8: Authentication trong ASP.NET Core Web API

Bài 1: Các bước thực hiện:

Tạo Project Asp net core web api.

Tiến hành cài đặt các Package sau:

Microsoft.AspNetCore.Identity.EntityFrameworkCore (phiên bản 8.x.x)

Microsoft.EntityFrameworkCore.Tools,

Microsoft.EntityFrameworkCore.SqlServer

Thêm class **UserContext.cs** như sau:

```
public class UsersContext : IdentityUserContext<IdentityUser>
{
    public UsersContext(DbContextOptions<UsersContext> options) : base(options)
    {
    }
    protected override void OnConfiguring(DbContextOptionsBuilder options)
    {
        options.UseSqlServer("Server = .; Database = Lab8Db; Trusted_Connection = True; MultipleActiveResultSets = True; TrustServerCertificate = True;");
    }
}
```

Thêm một số cấu hình cài đặt thông qua Service AddIdentity khi đăng ký tài khoản vào file Program.cs như sau:

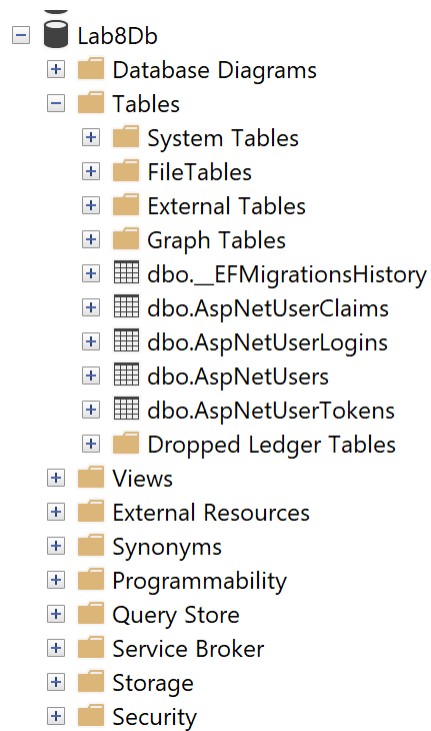
```
builder.Services.AddDbContext<UsersContext>();
builder.Services.AddIdentityCore<IdentityUser>(options =>
{
    options.SignIn.RequireConfirmedAccount = false;
    options.User.RequireUniqueEmail = true;
    options.Password.RequireDigit = false;
    options.Password.RequiredLength = 6;
    options.Password.RequireNonAlphanumeric = false;
    options.Password.RequireUppercase = false;
    options.Password.RequireLowercase = false;
})
.AddEntityFrameworkStores<UsersContext>();
```

Mở Package Manager Console chạy các lệnh sau:

Add-migration InitDb

Update-database

Mở SQL Server, kiểm tra các Table được tạo:



Thêm mới một số class vào project như sau:

Class **RegistrationRequest.cs**

```
public class RegistrationRequest
{
    [Required]
    public string Email { get; set; } = null!;

    [Required]
    public string Username { get; set; } = null!;

    [Required]
    public string Password { get; set; } = null!;
}
```

Class **AuthRequest.cs**

```
public class AuthRequest
{
    public string Email { get; set; } = null!;

    public string Password { get; set; } = null!;
}
```

Class **AuthResponse.cs**

```
public class AuthResponse
{
    public string Username { get; set; } = null!;
    public string Email { get; set; } = null!;
}
```

Thêm Web Api Controller: **AuthController** như sau:

```
public class AuthController : ControllerBase
{
    private readonly UserManager<IdentityUser> _userManager;
    private readonly UsersContext _context;
    public AuthController(UserManager<IdentityUser> userManager, UsersContext context)
    {
        _userManager = userManager;
        _context = context;
    }
}
```

```
[HttpPost]
[Route("register")]
public async Task<IActionResult> Register(RegistrationRequest request)
{
    if (!ModelState.IsValid)
    {
        return BadRequest(ModelState);
    }

    var result = await _userManager.CreateAsync(
        new IdentityUser { UserName = request.Username, Email = request.Email },
        request.Password
    );

    if (result.Succeeded)
    {
        request.Password = "";
        return CreatedAtAction(nameof(Register), new { email = request.Email }, request);
    }

    foreach (var error in result.Errors)
    {
        ModelState.AddModelError(error.Code, error.Description);
    }
    return BadRequest(ModelState);
}
```

```

[HttpPost]
[Route("login")]
public async Task<ActionResult<AuthResponse>> Authenticate([FromBody] AuthRequest request)
{
    if (!ModelState.IsValid)
    {
        return BadRequest(ModelState);
    }

    var managedUser = await _userManager.FindByEmailAsync(request.Email);

    if (managedUser == null)
    {
        return BadRequest("Bad credentials");
    }

    var isPasswordValid = await _userManager.CheckPasswordAsync(managedUser, request.Password);

    if (!isPasswordValid)
    {
        return BadRequest("Bad credentials");
    }

    var userInDb = _context.Users.FirstOrDefault(u => u.Email == request.Email);
    if (userInDb is null)
        return Unauthorized();

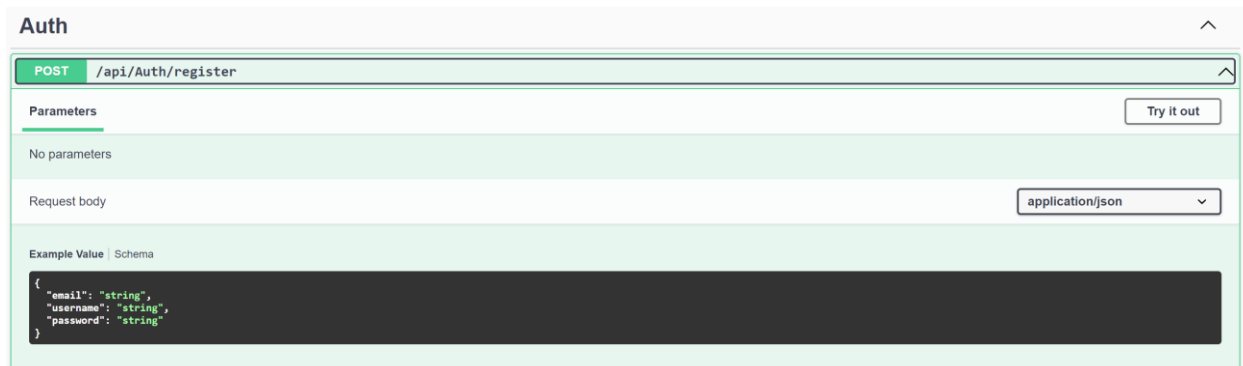
    return Ok(new AuthResponse
    {
        Username = userInDb.UserName,
        Email = userInDb.Email,
    });
}

```

Chạy Project (Ctrl + F5) kiểm thử API đã tạo sử dụng Swagger được tích hợp:

Đăng ký tài khoản:

Bấm vào Try it out, nhập các thông tin email, username, password để đăng ký tài khoản



Đăng nhập:

POST

/api/Auth/login

Parameters

Try it out

No parameters

Request body

application/json

Example Value | Schema

```
{
  "email": "string",
  "password": "string"
}
```

Sinh viên kiểm tra các trường hợp lỗi:

- Đăng ký không đầy đủ thông tin (thiếu địa chỉ Email, thiếu Username, thiếu Password)
- Đăng ký tài khoản bị trùng địa chỉ email đã có
- Đăng ký tài khoản Password không đúng yêu cầu (tối thiểu 6 ký tự)
- Đăng nhập tài khoản không đúng Username, Password

Các lỗi thông báo:

Code

Details

400

Undocumented

Error: response status is 400

Response body

```
{
  "DuplicateUserName": [
    "Username 'abc' is already taken."
  ]
}
```

Download

Code

Details

400

Undocumented

Error: response status is 400

Response body

```
{
  "DuplicateEmail": [
    "Email 'abc@gmail.com' is already taken."
  ]
}
```

Download

Server response

Code

Details

400

Undocumented

Error: response status is 400

Response body

Bad credentials

Download

Response headers

```
content-type: text/plain; charset=utf-8
date: Mon, 31 Mar 2025 13:06:54 GMT
server: Kestrel
```

Bài 2: Tiếp tục thực hiện trên Project ở bài 1, cài đặt thêm các Package sau:

Microsoft.AspNetCore.Authentication.JwtBearer (8.0.14)

System.IdentityModel.Tokens.Jwt

Mở file appsettings.json thêm vào cấu hình Jwt sau:

```
"Jwt": {  
  "Key": "a30ba53c-1430-4a3e-a3f8-a3e0f9ea80db",  
  "Issuer": "mycompany.com",  
  "Audience": "https://localhost:5261"  
}
```

Trong Class **AuthResponse.cs** thêm mới một property sau (được sử dụng để lưu giá trị token được sinh ra):

```
public string Token { get; set; } = null!;
```

Thêm cấu hình xác thực token JWT vào file Program.cs như sau:

```
var jwtIssuer = builder.Configuration.GetSection("Jwt:Issuer").Get<string>();  
var jwtKey = builder.Configuration.GetSection("Jwt:Key").Get<string>();  
var jwtAudience = builder.Configuration.GetSection("Jwt:Audience").Get<string>();  
builder.Services.AddAuthentication(JwtBearerDefaults.AuthenticationScheme)  
    .AddJwtBearer(options =>  
    {  
        options.TokenValidationParameters = new TokenValidationParameters  
        {  
            ValidateIssuer = true,  
            ValidateAudience = true,  
            ValidateLifetime = true,  
            ValidateIssuerSigningKey = true,  
            ValidIssuer = jwtIssuer,  
            ValidAudience = jwtAudience,  
            IssuerSigningKey = new SymmetricSecurityKey(Encoding.UTF8.GetBytes(jwtKey))  
        };  
    });
```

Khai báo thêm biến **_config** và thêm vào constructor của class **AuthController.cs** hiện tại để có thể đọc dữ liệu config từ file appsettings.json:

```
private readonly IConfiguration _config;
public AuthController(UserManager<IdentityUser> userManager, UsersContext context,
    IConfiguration config)
{
    _userManager = userManager;
    _context = context;
    _config = config;
}
```

Thêm đoạn code sau vào class **AuthController.cs** để tạo Token khi User đăng nhập:

```
var jwtSettings = _config.GetSection("Jwt");
var securityKey = new SymmetricSecurityKey(Encoding.UTF8.GetBytes(jwtSettings["Key"]));
var credentials = new SigningCredentials(securityKey, SecurityAlgorithms.HmacSha256);

List<Claim> claims = new List<Claim> {
    new Claim(type: "MyClaim", value: "Demo")
};

var tokenOptions = new JwtSecurityToken
(
    issuer: jwtSettings["Issuer"],
    audience: jwtSettings["Audience"],
    claims: claims,
    expires: DateTime.Now.AddMinutes(120),
    signingCredentials: credentials
);
var token = new JwtSecurityTokenHandler().WriteToken(tokenOptions);
return Ok(new AuthResponse
{
    Username = userInDb.UserName,
    Email = userInDb.Email,
    Token = token
});
```

Thêm mới một **Web Api controller** và đặt tên **TestController** để kiểm tra xác thực sử dụng Token (lưu ý thêm attribute `[Authorize]` cho Controller này)

```
[Authorize]
[Route("api/[controller]")]
[ApiController]
0 references
public class TestController : ControllerBase
{
    [HttpGet]
    0 references
    public IActionResult Get()
    {
        return Ok("My test data");
    }
}
```

Sử dụng **Postman** kiểm tra request đến **TestController** sử dụng xác thực **JWT** (lưu ý thêm Bearer "Token" vào Header của request).

Ví dụ:

Đăng ký tài khoản

Sử dụng JWT trả về sau khi User đăng nhập

Params	Authorization	Headers (8)	Body	Pre-request Script	Tests	Settings	Cookies
Headers 7 hidden							
	Key	Value	Description	... Bulk Edit Presets			
<input checked="" type="checkbox"/>	Authorization	Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ...					
	Key	Value	Description				

Nhập Email và Password theo định dạng JSON vào phần Body của Request.

Kiểm tra dữ liệu trả về thành công (Mã: 200OK, dữ liệu trả về từ Action Method Get: “My test data”)

BodyCookiesHeaders (4)Test Results

200 OK

Raw

Preview

Visualize

1My test data

Kiểm tra các trường hợp lỗi:

Chưa đăng nhập

Không đúng mã JWT gửi kèm

BodyCookiesHeaders (4)Test Results🔄




401 Unauthorized • 16 ms

📄 Raw ▾

▶ Preview

🔄 Visualize ▾

1

Body	Cookies	Headers (4)	Test Results	🔄
405 Method Not Allowed				
 Raw	 Preview	 Visualize		