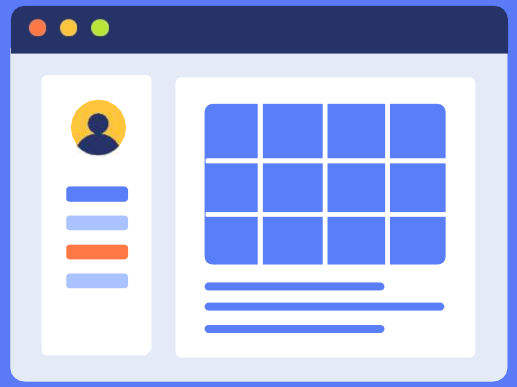
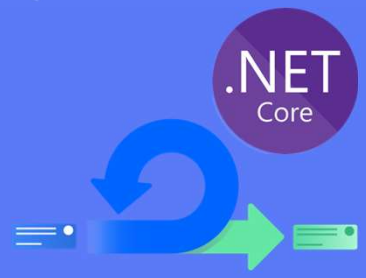
P.PORTO
ESCOLA SUPERIOR DE TECNOLOGIA E GESTÃO

Laboratório de desenvolvimento de software





Licenciatura em Engenharia Informática 2022/2023

P.PORTO
ESCOLA SUPERIOR DE TECNOLOGIA E GESTÃO

ASP.NET CORE - WEBAPI E JWT

Um ponto importante no desenvolvimento de webapis está realcionado com a segurança e controlo de acesso de utilizadores

É preciso não esquecer que autenticação ≠ autorização

- Autenticação é um processo no qual um utilizador fornece credenciais que são comparadas às armazenadas num sistema operacional, base de dados, aplicação ou outro recurso. Se eles corresponderem, os utilizadores serão autenticados com êxito e poderão executar ações para as quais estão autorizados durante um processo de autorização.
- Autorização refere-se ao processo que determina o que um utilizador pode fazer.

ASP.NET CORE - WEBAPI E JWT

O ASP.NET Core oferece ferramentas e bibliotecas para proteger as aplicações, incluindo identitity provider integrados

Pode-se no entanto usar identity serviços de terceiros, como Facebook, Twitter ou LinkedIn.

Umas das bibliotecas com suporte nativo é o JWT

ASP.NET CORE - WEBAPI E JWT

JSON Web Token (JWT) é um open standard (RFC 7519) que define um método compacto e autocontido para transmitir com segurança informações entre as partes num objeto JSON.

As JWTs podem ser assinados utilizando um segredo (com o algoritmo HMAC) ou um par de chaves pública/privada usando RSA ou ECDSA.

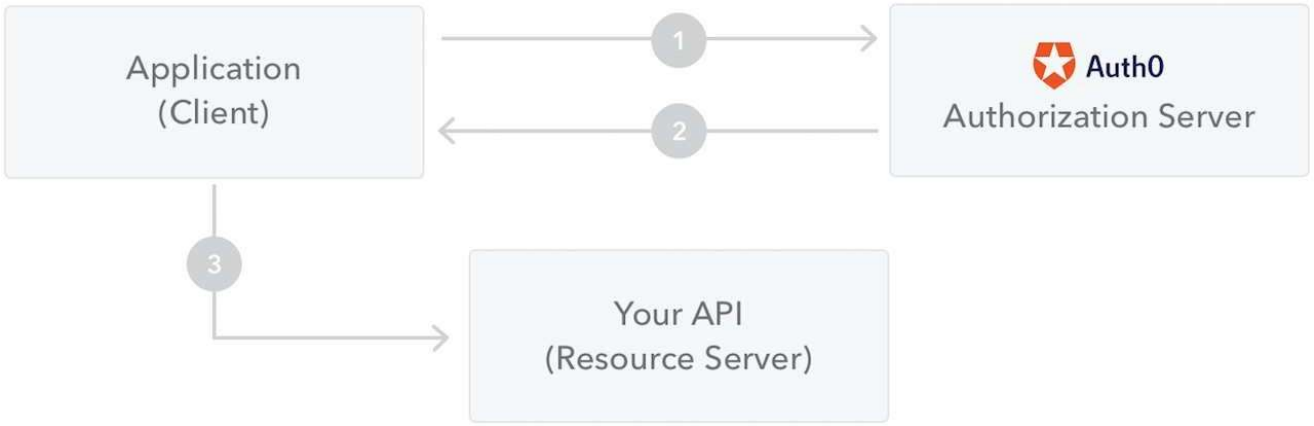
ASP.NET CORE - WEBAPI E JWT

Embora JWTs possam ser criptografadas para também fornecer sigilo entre as partes, iremos focar-nos no use de signed tokens

Os signed tokens podem verificar a integridade da informação contidas nele, enquanto os tokens criptografados ocultam essas declarações de outras partes

Quando os tokens são assinados usando pares de chaves pública/privada, a assinatura também certifica que a parte que é prorietária da chave privada é aquela que a assinou.

ASP.NET CORE - WEBAPI E JWT



Fonte: https://jwt.io/introduction/

ASP.NET CORE - WEBAPI E JWT

Configuração de uma aplicação para uso de autenticação com JWT no ficheiro Startup.cs

Verificar o uso de serviços como:

- CORS
- MVC
- JWTBearer

```
public void ConfigureServices(IServiceCollection services)
{
    services.AddDbContext<HelloWorldContext>(opt =>
        opt.UseInMemoryDatabase("PeopleList"));
    services.AddMvc().SetCompatibilityVersion(CompatibilityVersion.Version_2_1);
    services.AddAutoMapper();
    services.AddCors();


    // configure strongly typed settings objects
    var appSettingsSection = Configuration.GetSection("AppSettings");
    services.Configure<AppSettings>(appSettingsSection);

    // configure jwt authentication
    var appSettings = appSettingsSection.Get<AppSettings>();
    var key = Encoding.ASCII.GetBytes(appSettings.Secret);
    services.AddAuthentication(x =>
    {
        x.DefaultAuthenticateScheme = JwtBearerDefaults.AuthenticationScheme;
        x.DefaultChallengeScheme = JwtBearerDefaults.AuthenticationScheme;
    })
    .AddJwtBearer(x =>
    {
        x.Events = new JwtBearerEvents
        {
            OnTokenValidated = context =>
            {
                var userService = context.HttpContext.RequestServices.GetRequiredService<IUserService>();
                var userId = int.Parse(context.Principal.Identity.Name);
                var user = userService.GetById(userId);
                if (user == null)
                {
                    // return unauthorized if user no longer exists
                    context.Fail("Unauthorized");
                }
                return Task.CompletedTask;
            }
        };
        x.RequireHttpsMetadata = false;
        x.SaveToken = true;
        x.TokenValidationParameters = new TokenValidationParameters
        {
            ValidateIssuerSigningKey = true,
            IssuerSigningKey = new SymmetricSecurityKey(key),
            ValidateIssuer = false,
            ValidateAudience = false
        };
    });

    // configure DI for application services
    services.AddScoped<IUserService, UserService>();

}
```

ASP.NET CORE - WEBAPI E JWT

Continuação de configuração do ficheiro Startup.cs

```
public void Configure(IApplicationBuilder app, IHostingEnvironment env)
{
    if (env.IsDevelopment())
    {
        app.UseDeveloperExceptionPage();
    }
    else
    {
        app.UseHsts();
    }

    app.UseHttpsRedirection();

    // global cors policy
    app.UseCors(x => x
        .AllowAnyOrigin()
        .AllowAnyMethod()
        .AllowAnyHeader()
        .AllowCredentials());

    app.UseAuthentication();

    app.UseMvc();
}
```

ASP.NET CORE - WEBAPI E JWT

O código dos nossos controllers deve verificar a necessidade de fazer autenticação de utilizadores

Deve também no seu construtor ser passados todos os serviços a serem usados no processo de autenticação

```
namespace HelloWorld.Controllers
{
    [Authorize]
    [ApiController]
    [Route("[controller]")]
    0 references
    public class UsersController : ControllerBase
    {
        7 references
        private IUserService _userService;
        5 references
        private IMapper _mapper;
        2 references
        private readonly AppSettings _appSettings;

        0 references
        public UsersController(
            IUserService userService,
            IMapper mapper,
            IOptions<AppSettings> appSettings)
        {
            _userService = userService;
            _mapper = mapper;
            _appSettings = appSettings.Value;
        }
```

ASP.NET CORE - WEBAPI E JWT

Podemos adicionar exceções à regra de autenticação para métodos que o justifiquem

```
[AllowAnonymous]
[HttpPost("register")]
0 references
public IActionResult Register([FromBody]UserDto userDto)
{
    // map dto to entity
    var user = _mapper.Map<User>(userDto);

    try
    {
        // save
        _userService.Create(user, userDto.Password);
        return Ok();
    }
    catch(AppException ex)
    {
        // return error message if there was an exception
        return BadRequest(new { message = ex.Message });
    }
}
```

ASP.NET CORE - WEBAPI E JWT

Método de autenticação para validar utilizadores

```
[AllowAnonymous]
[HttpPost("authenticate")]
0 references
public IActionResult Authenticate([FromBody]UserDto userDto)
{
    var user = _userService.Authenticate(userDto.Username, userDto.Password);

    if (user == null)
        return BadRequest(new { message = "Username or password is incorrect" });

    var tokenHandler = new JwtSecurityTokenHandler();
    var key = Encoding.ASCII.GetBytes(_appSettings.Secret);
    var tokenDescriptor = new SecurityTokenDescriptor
    {
        Subject = new ClaimsIdentity(new Claim[]
        {
            new Claim(ClaimTypes.Name, user.Id.ToString())
        }),
        Expires = DateTime.UtcNow.AddDays(7),
        SigningCredentials = new SigningCredentials(new SymmetricSecurityKey(key),
        SecurityAlgorithms.HmacSha256Signature)
    };
    var token = tokenHandler.CreateToken(tokenDescriptor);
    var tokenString = tokenHandler.WriteToken(token);

    // return basic user info (without password) and token to store client side
    return Ok(new {
        Id = user.Id,
        Username = user.Username,
        FirstName = user.FirstName,
        LastName = user.LastName,
        Token = tokenString
    });
}
```

ASP.NET CORE - WEBAPI E JWT

O exemplo descrito uso também outros recursos como:

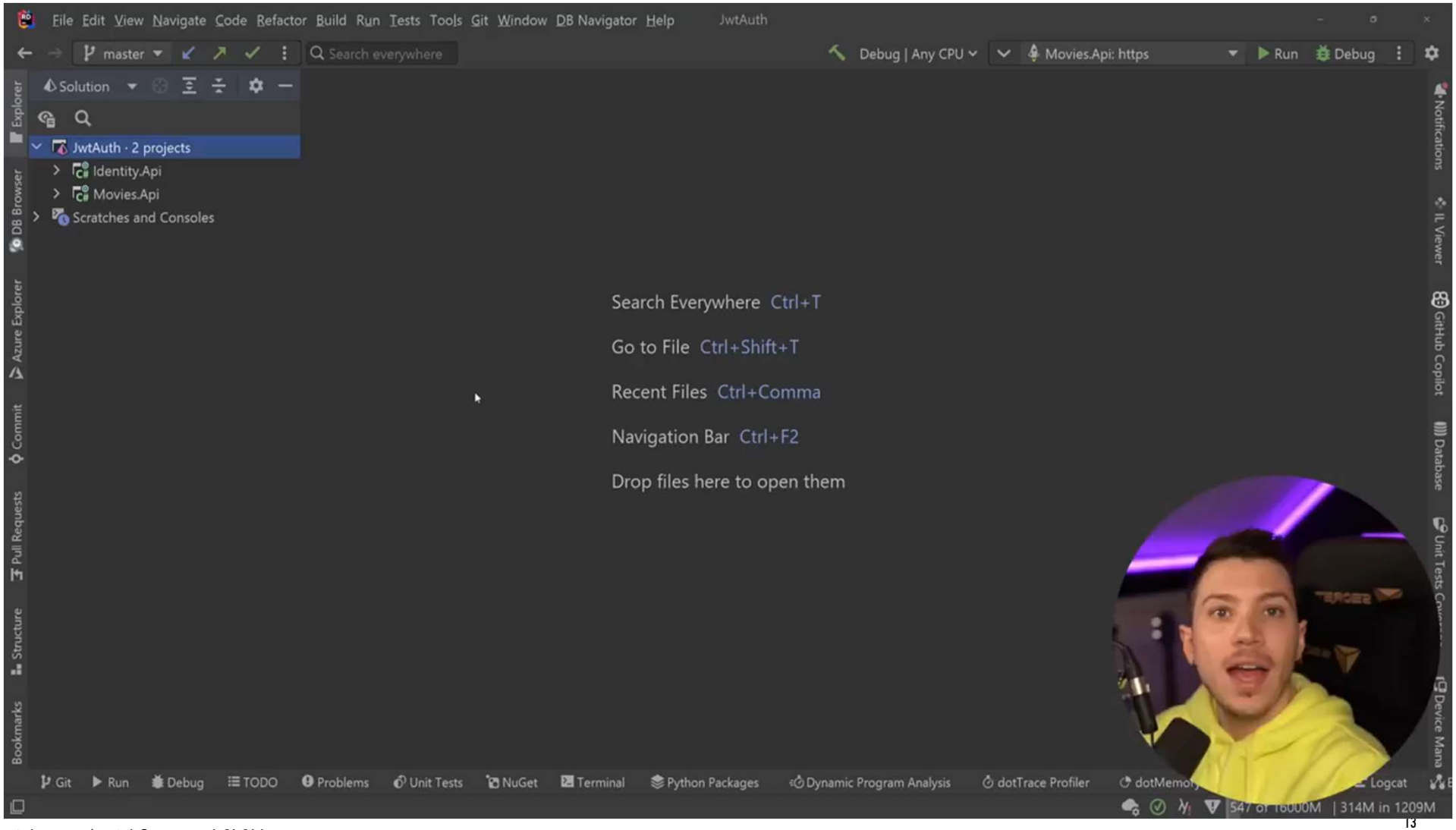
- Dependency injection
- AutoMaper

Será demonstrado o seu uso na demonstração prática

Deveremos também importar estas bibliotecas para o projeto

```
<Project Sdk="Microsoft.NET.Sdk.Web">
  <PropertyGroup>
    <TargetFramework>netcoreapp2.1</TargetFramework>
  </PropertyGroup>
  <ItemGroup>
    <PackageReference Include="AutoMapper" Version="6.0.2" />
    <PackageReference Include="AutoMapper.Extensions.Microsoft.DependencyInjection" Version="2.0.1" />
    <PackageReference Include="Microsoft.AspNetCore.App" />
  </ItemGroup>
</Project>
```

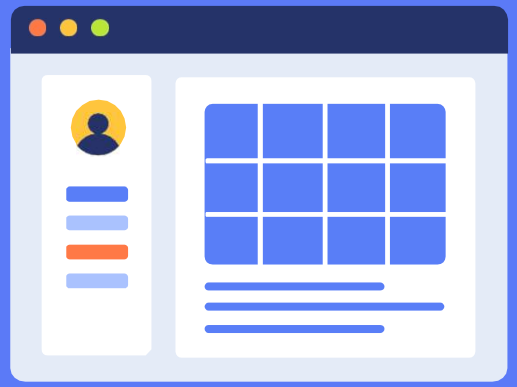
ASP.NET CORE - WEBAPI E JWT

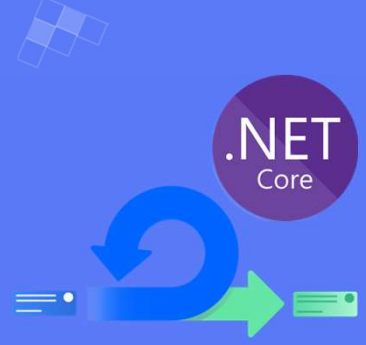


https://www.youtube.com/watch?v=mgeuh8k3l4g

P.PORTO
ESCOLA SUPERIOR DE TECNOLOGIA E GESTÃO

Laboratório de desenvolvimento de software





Licenciatura em Engenharia Informática 2022/2023

P.PORTO
ESCOLA SUPERIOR DE TECNOLOGIA E GESTÃO