**DOWNLOADS NECESSÁRIOS**

Visual Studio 2022 (com ASP.NET)

VSCode

Postman

DB Browser for SQLite

Projeto inicial - <https://github.com/alura-cursos/4320-seguranca-aspnetcore>

**CONFIGURAÇÕES INICIAIS**

Instalar gerenciador de BD do .NET - Na pasta MedVoll.Web, pelo CMD

*dotnet tool install --global dotnet-ef*

Aplicar as migrations iniciais do projeto -Pelo CMD, na pasta MedVoll.Web

*dotnet ef database update*

Com o projeto aberto no Visual Strudio - Instalar Identity - Pelo Nuget

Na interface do Visual Studio -> Ferramentas -> Gerenciador de Pacotes Nuget-> Gerenciar Pacotes

Nome: Microsoft.AspNetCore.Identity.EntityFrameworkCore (v. 9.0.0)

Alterar classe ApplicationDbContext para herdar de IdentityDbContext

public **class** **ApplicationDbContext** : IdentityDbContext

Scaffolding ao ASP.NETCoreIdentity - Criar a estrutura necessária para usar o identity na aplicação (nas views Account/Login e Account/Register)

-Clique com o botão direito no projeto MedVoll.Web no Solution Explorer.

-Escolha o menu: Add > Add New Scaffolded Item.

-Na lista, selecione Identity.

-Marque os arquivos necessários:

Account\Login

Account\Register

-No campo DbContext class, selecione:ApplicationDbContext (MedVoll.Web.Data).

Gerar migração para o esquema do ASP.NET Core Identity - Pelo CMD, na pasta MedVoll.Web

dotnet ef migrations add CreateIdentitySchema

Aplicar migração ao BD

dotnet ef database update

\*Para consultar o banco de dados da aplicação

Botão direito em vollmed.db -> abrir com -> DB Browser fo SQLite

CRIAÇÃO DE USUÁRIOS DE TESTE

Criar classe Data-> IdentitySeeder

using Microsoft.AspNetCore.Identity;

namespace MedVoll.Web.Data

{

public class IdentitySeeder

{

//MÉTODO QUE CRIA DOIS USUÁRIOS DE PAPEL User

public static async Task SeedUsersAsync(IServiceProvider serviceProvider)

{

//OBJETOS UserManager E RoleManager - GESTÃO DE USUÁRIOS E PAPÉIS

var userManager = serviceProvider.GetRequiredService<UserManager<IdentityUser>>();

var roleManager = serviceProvider.GetRequiredService<RoleManager<IdentityRole>>();

// Verifica e cria a função "User", se necessário

// CRIA O PAPÉL DE User, CASO AINDA NÃO EXISTA

const string userRole = "User";

if (!await roleManager.RoleExistsAsync(userRole))

{

await roleManager.CreateAsync(new IdentityRole(userRole));

}

// COM A FUNÇÃO CreateUserAsync - CRIA DOIS USUÁRIOS ESPECÍFICOS

await CreateUserAsync(userManager, "alice@smith.com", "Password@123", userRole);

await CreateUserAsync(userManager, "bob@smith.com", "Password@123", userRole);

}

//MÉTODO QUE CRIA USUÁRIOS-DADO O OBJETO UserManager, EMAIL, SENHA E PAPÉL

private static async Task CreateUserAsync(UserManager<IdentityUser> userManager, string email, string password, string role)

{

// SE O USUÁRIO JÁ EXISTE (VERIFICADO PELO E-MAIL) NÃO CRIA

if (await userManager.FindByEmailAsync(email) != null)

{

return;

}

//SE NÃO EXISTE..

var user = new IdentityUser

{

UserName = email,

Email = email,

EmailConfirmed = true // Para evitar a necessidade de confirmação de email

};

// Cria o usuário

var result = await userManager.CreateAsync(user, password);

if (result.Succeeded)

{

// Atribui a função ao usuário

await userManager.AddToRoleAsync(user, role);

}

else

{

throw new Exception($"Erro ao criar usuário {email}: {string.Join(", ", result.Errors)}");

}

}

}

}

Program->Linha22 - builder.Services.AddDefaultIdentity.... Fazer com que a aplicação seja capaz de usar o IdentitySeeder.cs para adicionar papéis aos usuários

builder.Services.AddDefaultIdentity<IdentityUser>(options => options.SignIn.RequireConfirmedAccount = true)

.AddRoles<IdentityRole>()

.AddEntityFrameworkStores<ApplicationDbContext>();

Program-> Linha 53 e abaixo - Criar um escopo ServiceProvider que vai ser capaz de injetar dependências entre as classes do projeto

using (var scope = app.Services.CreateScope())

{

var services = scope.ServiceProvider;

try

{

await IdentitySeeder.SeedUsersAsync(services);

}

catch (Exception ex)

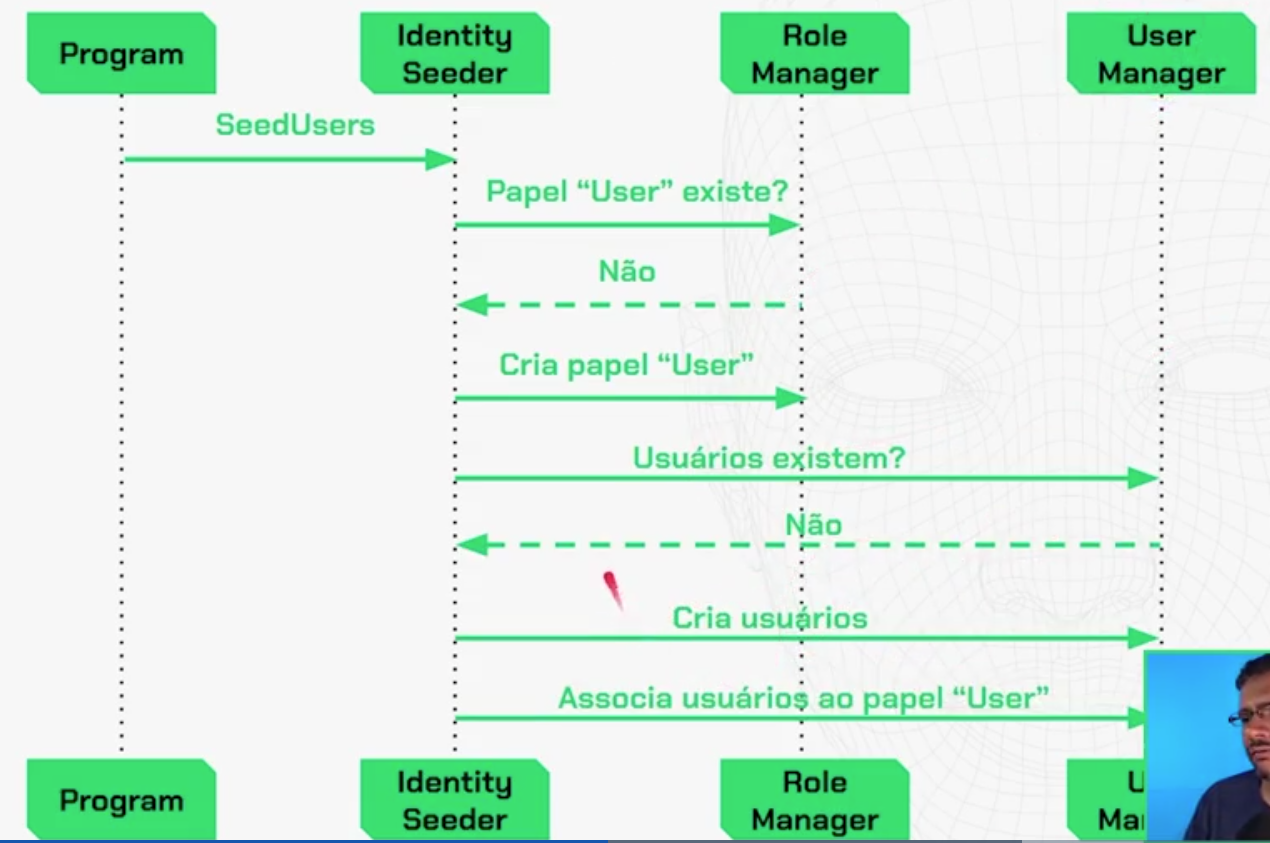
{

Console.WriteLine($"Erro ao executar o Seeder: {ex.Message}");

}

}

Diagrama de sequência - Criação de usuários teste



Como consultar as informações inseridas na BD?

Botão direito em vollmed.db -> abrir com DB Browser for SQLite-> Browse Data->

Assim é possível consultar nas tabelas AspNetUsers e AspNetRoles os usuários e papéis criados pela última execução.

Lembrando que: considerando como o código foi escrito, caso seja executado novamente, não vai duplicar os usuários/papéis ou gerar erro, dado que, caso já existam os registros, ele simplesmente não faz nada.