**✅ Prompt para Cursor.IA**

plaintext

CopiarEditar

Crie um projeto ASP.NET Core MVC em C# salvo no caminho `C:\Projetos\Gerente` com as seguintes características:

✅ Banco de dados:

- PostgreSQL local

- Nome do banco: Projetos

- Usuário: admin

- Senha: admin

- Use o pacote Npgsql para a conexão.

✅ Funcionalidades:

1. Tela de login com validação no banco de dados:

- Tabela: usuarios (id, username, senha)

- Crie autenticação básica sem identidade ASP.NET, apenas validação direta na tabela.

- Armazene a autenticação via Session.

2. Tela principal (MainPage.cshtml ou similar):

- Layout profissional com um painel lateral (Sidebar) fixo à esquerda.

- Inclua 4 botões fictícios: Dashboard, Projetos, Usuários, Relatórios.

- Ao fazer login com sucesso, redirecionar para essa tela.

3. Estrutura do banco:

```sql

CREATE DATABASE "Projetos";

\c Projetos

CREATE TABLE usuarios (

id SERIAL PRIMARY KEY,

username VARCHAR(100) NOT NULL UNIQUE,

senha VARCHAR(100) NOT NULL

);

INSERT INTO usuarios (username, senha) VALUES ('admin', 'admin');

-- Script para criar a tabela de configurações de e-mail

-- Execute este script no seu banco de dados PostgreSQL

CREATE TABLE IF NOT EXISTS configuracoes\_email (

    id SERIAL PRIMARY KEY,

    servidor\_smtp VARCHAR(255) NOT NULL,

    porta INTEGER NOT NULL,

    email\_remetente VARCHAR(255) NOT NULL,

    nome\_remetente VARCHAR(255) NOT NULL,

    usuario\_smtp VARCHAR(255) NOT NULL,

    senha\_smtp VARCHAR(255) NOT NULL,

    data\_criacao TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

    data\_atualizacao TIMESTAMP DEFAULT CURRENT\_TIMESTAMP

);

-- Inserir configurações padrão (exemplo)

INSERT INTO configuracoes\_email (servidor\_smtp, porta, email\_remetente, nome\_remetente, usuario\_smtp, senha\_smtp)

VALUES ('smtp.gmail.com', 587, 'sistema@empresa.com', 'Sistema Gerente', 'sistema@empresa.com', 'senha123')

ON CONFLICT DO NOTHING;

--- importantíssimo:

CREATE USER admin WITH PASSWORD 'admin;

ALTER USER admin WITH SUPERUSER; *-- Se precisar de privilégios elevados*

✅ Páginas:

* Login (LoginController + View)
* Main (HomeController + View)

✅ Configuração extra:

* Adicione a string de conexão ao appsettings.json
* Configure o middleware de sessão (services.AddSession()) e autenticação no Program.cs ou Startup.cs
* No Program.cs, defina a página de login como padrão se não houver sessão ativa.

📌 Observações:

* Não use Entity Framework neste projeto (acesso ao banco deve ser via Npgsql direto).
* Todas as views devem usar Razor.
* Evite dependências desnecessárias.

Build 001:

Perfeito! Criei uma tela de login moderna e profissional com as seguintes características:

**✅ Características implementadas:**

**Design Moderno e Profissional:**

* Layout limpo e minimalista
* Gradiente de fundo suave (roxo/azul)
* Card com efeito glassmorphism (transparência + blur)
* Tipografia moderna (fonte Inter do Google Fonts)
* Cores suaves e harmoniosas

**Responsividade:**

* Totalmente responsivo para desktop e mobile
* Breakpoints específicos para diferentes tamanhos de tela
* Evita zoom automático no iOS (font-size: 16px)

**Layout Centralizado:**

* Formulário centralizado vertical e horizontalmente
* Apenas a tela de login é exibida (sem sidebar/menus)
* Rodapé fixo com "© 2025 Minha Empresa"

**Elementos do Formulário:**

* Campos para usuário e senha com placeholders
* Labels claros e bem posicionados
* Botão de login estilizado com gradiente
* Mensagem de erro estilizada
* Animações suaves (fadeInUp)

**Arquivos criados/modificados:**

1. **Views/Shared/\_LoginLayout.cshtml** - Layout específico para login
2. **wwwroot/css/login.css** - Estilos completos e responsivos
3. **Views/Login/Index.cshtml** - View atualizada com novo design

A tela agora tem um visual profissional, moderno e totalmente funcional! 🎨✨

Build 002:

**✅ Funcionalidades Implementadas:**

**1. Sidebar Profissional e Responsiva:**

* **Design moderno** com gradiente e efeitos visuais
* **Ícones Bootstrap** para cada botão (Dashboard, Projetos, Usuários, Relatórios)
* **Botão "Sair"** posicionado na parte inferior da sidebar
* **Tooltips** informativos em cada botão
* **Animações suaves** no hover

**2. Funcionalidade de Logout:**

* **Método Logout()** no HomeController que limpa a sessão
* **Confirmação** antes de sair (JavaScript confirm)
* **Redirecionamento** automático para a tela de login
* **Link seguro** usando @Url.Action("Logout", "Home")

**3. Design Responsivo:**

* **Mobile-first** com sidebar que se esconde em telas pequenas
* **Botão toggle** para mostrar/ocultar sidebar no mobile
* **Fechamento automático** da sidebar ao clicar fora (mobile)
* **Transições suaves** em todas as interações

**4. Página Principal Melhorada:**

* **Header profissional** com informações do usuário
* **Cards interativos** com ícones e botões
* **Estatísticas rápidas** para dar contexto
* **Layout responsivo** que se adapta a diferentes telas

**5. Arquivos Atualizados:**

1. **Controllers/HomeController.cs** - Adicionado método Logout
2. **Views/Shared/\_Layout.cshtml** - Sidebar completa com logout
3. **wwwroot/css/site.css** - Estilos responsivos e profissionais
4. **Views/Home/Index.cshtml** - Página principal melhorada

**6. Características Técnicas:**

* **Bootstrap Icons** para ícones modernos
* **CSS3** com animações e transições
* **JavaScript** para interatividade mobile
* **Razor** para renderização dinâmica
* **Sessão** para controle de autenticação

O sistema agora tem uma interface profissional, totalmente responsiva e com funcionalidade completa de logout! 🎨

Compilar:

## 4. Restaurar pacotes e compilar

dotnet restore

dotnet build

dotnet run

smtp casa: **saqx wpzx cyew uhll**