تمرین: پروژه Backend با C# و PostgreSQL در VS Code

مشخصات پروژه

- \*\*نام پروژه\*\*: `PostgresBackendApp`

- \*\*نوع پروژه\*\*: ASP.NET Core Web API

- \*\*پایگاه داده\*\*: PostgreSQL

- \*\*عملیات\*\*: Insert، Update، Delete و Select بر روی جدول محصولات

مراحل انجام تمرین با VS Code

1. نصب پیش‌نیازها

- \*\*.NET SDK\*\*: اطمینان حاصل کنید که \*\*.NET SDK\*\* را نصب کرده‌اید.

- \*\*PostgreSQL\*\*: PostgreSQL را نصب کنید و اطمینان حاصل کنید که در حال اجراست.

- \*\*VS Code\*\*: Visual Studio Code را نصب کنید.

- \*\*C# Extension\*\*: افزونه C# را در VS Code نصب کنید.

1. ایجاد پروژه جدید
2. باز کردن ترمینال\*\*:

در VS Code، ترمینال را باز کنید.

1. ایجاد پروژه جدید\*\*:

فرمان `dotnet new webapi -n PostgresBackendApp` را اجرا کنید تا پروژه جدیدی ایجاد شود.

به پوشه پروژه وارد شوید با `cd PostgresBackendApp`.

1. نصب بسته‌های مورد نیاز

از `dotnet add package Dapper` و `dotnet add package Npgsql` برای نصب Dapper و Npgsql استفاده کنید.

1. تنظیم پایگاه داده
2. ایجاد پایگاه داده و جدول

از \*\*pgAdmin\*\* یا \*\*psql\*\* برای ایجاد یک پایگاه داده جدید به نام `DapperExampleDb` و جدول `Products` استفاده کنید.

1. قرار دادن داده‌های نمونه

با استفاده از دستورات SQL، چند محصول نمونه به جدول اضافه کنید.

1. ایجاد مدل

یک فایل به نام `Product.cs` در پوشه `Models` ایجاد کنید که ویژگی‌های محصول را تعریف کند.

1. ایجاد Repository

یک فایل به نام `ProductRepository.cs` در پوشه `Repositories` ایجاد کنید که متدهای CRUD را پیاده‌سازی کند.

1. ایجاد Controller

یک فایل به نام `ProductsController.cs` در پوشه `Controllers` ایجاد کنید که API endpoints را برای عملیات CRUD تعریف کند.

1. تنظیم Dependency Injection

در فایل `Program.cs`، Repository را به DI Container اضافه کنید.

1. پیکربندی رشته اتصال

در فایل `appsettings.json`، رشته اتصال به پایگاه داده PostgreSQL را اضافه کنید.

1. تست API

از ابزارهایی مانند \*\*Postman\*\* یا \*\*cURL\*\* برای تست API خود استفاده کنید و عملیات Insert، Update، Delete و Select را انجام دهید.