InvMIS (Inventory Management System)

Step-by-Step Guide with Database: InventoryDB

Task #01: Project Setup

```
- Create solution `InvMIS`.
- Add projects:
  - InvMIS.API (Web API project)
  - InvMIS.Domain (Entities, Core Models)
  - InvMIS.Application (Interfaces, Business Logic)
  - InvMIS.Infrastructure (DbContext, Repository)

References:
- InvMIS.API → Application, Infrastructure, Domain
- InvMIS.Application → Domain
- InvMIS.Infrastructure → Application, Domain
```

Task #02: Install EF Core & Setup DbContext

```
- Install EF Core packages in Infrastructure project:
  dotnet add package Microsoft.EntityFrameworkCore
  dotnet add package Microsoft.EntityFrameworkCore.Design
  dotnet add package Npgsql.EntityFrameworkCore.PostgreSQL

- Create Data/InvMISDbContext.cs inside Infrastructure project.
```

Task #03: Define Entities

```
Inside Domain/Entities:
- Product.cs
- Category.cs
- Supplier.cs
- Stock.cs
- User.cs
```

Task #04: Repository Pattern

```
- Create IRepository<T> in Application/Interfaces.
- Implement Repository<T> in Infrastructure/Repositories.
```

Task #05: Configure DI & Services

```
Update Program.cs in API to register DbContext, Repositories, and Services.
Example:
services.AddScoped<IProductService, ProductService>();
```

Task #06: ProductController

Create API Controller in API/Controllers/ProductController.cs with CRUD endpoints.

Task #07: Swagger Setup

- Add Swashbuckle.AspNetCore package.
- Update Program.cs with UseSwagger() & UseSwaggerUI().

Task #08: appsettings.json (Database Config)

Set connection string:

```
"ConnectionStrings": {
  "DefaultConnection": "Host=...;Database=InventoryDB;..."
}
```

Task #09: JWT Authentication

- Add Jwt settings in appsettings.json.
- Configure Authentication in Program.cs.

Task #10: Role-based Authorization

- Add [Authorize(Roles="Admin")] in controllers.
- Update appsettings.json Jwt:Audience = InvMISAPIUsers.

Task #11: Migrations

Run inside Infrastructure:

- dotnet ef migrations add InitialCreate -p InvMIS.Infrastructure -s InvMIS.API
- dotnet ef database update -p InvMIS.Infrastructure -s InvMIS.API

Task #12: ProductController Update

Add async CRUD with validation, return IActionResult with Ok/NotFound/BadRequest responses.

Task #13: Global Exception Handling

Add middleware in Program.cs to handle errors globally and return JSON error response.

Task #14: Validation

Use DataAnnotations in Domain Entities (e.g., [Required], [StringLength]).

Task #15: User & Identity

Extend DbContext with Identity.
Configure User entity for authentication & authorization.

Final Testing Guide

1. Run `dotnet build` to ensure solution builds without errors.
2. Run migrations: `dotnet ef database update -p InvMIS.Infrastructure -s InvMIS.API`.
3. Press F5 or run `dotnet run` from API project.
4. Open Swagger UI at https://localhost:<port>/swagger.
5. Test CRUD for Products, Categories, Suppliers, Stocks.
6. Test JWT authentication by logging in and calling secured endpoints with Bearer token.
7. Verify role-based authorization (Admin vs User).

Project Reference Map

InvMIS.API → Application, Infrastructure, Domain
InvMIS.Application → Domain
InvMIS.Infrastructure → Application, Domain
InvMIS.Domain → (No dependencies)