به نام خدا

راه کامل پروژه Auth System\*\* به همراه ساختار پوشه‌بندی و کلاس‌های اصلی آورده شده است:

## 🗺️ \*\*نقشه راه کامل پروژه Auth System\*\*

### \*\*مرحله 1: راه‌اندازی اولیه پروژه ✅\*\*

\* \*\*هدف:\*\* ایجاد ساختار پایه Solution و پروژه‌های مربوطه.

\* \*\*انجام شده:\*\*

\* ایجاد پوشه‌بندی اصلی (src, tests, docs, scripts, docker, ...).

\* ایجاد پروژه‌های `AuthSystem.Domain`, `AuthSystem.Application`, `AuthSystem.Infrastructure`, `AuthSystem.Shared`, `AuthSystem.Api`, `AuthSystem.UnitTests`.

\* تنظیم رفرنس‌های بین پروژه‌ها.

\* نصب بسته‌های NuGet اولیه.

### \*\*مرحله 2: پیکربندی اولیه AuthSystem.Api ✅\*\*

\* \*\*هدف:\*\* پیکربندی پروژه API برای اجرا و لاگ‌نویسی.

\* \*\*انجام شده:\*\*

\* پیکربندی `Program.cs` برای Serilog و Swagger.

\* به‌روزرسانی `appsettings.json` با تنظیمات لاگ‌نویسی و Connection String.

### \*\*مرحله 3: پیاده‌سازی لایه Domain ✅\*\*

\* \*\*هدف:\*\* تعریف Entities، Enums، ValueObjects، Exceptions و Interfaceهای سرویس‌های دامنه.

\* \*\*انجام شده:\*\*

\* ایجاد پوشه‌های `Entities`, `Enums`, `ValueObjects`, `Exceptions`, `Services` در `AuthSystem.Domain`.

\* پیاده‌سازی Entities اصلی (`BaseEntity`, `User`, `Role`, `Permission`, `UserRole`, `RolePermission`, `RefreshToken`, `LoginHistory`, `UserDevice`).

\* پیاده‌سازی `DeviceType` Enum.

\* پیاده‌سازی ValueObject `Email`.

\* پیاده‌سازی Interfaceهای سرویس‌های دامنه (`IPasswordHasher`, `ITokenGenerator`, `IEmailService`, `ISmsService`, `IDateTimeProvider`).

\* پیاده‌سازی Exceptionهای سفارشی دامنه.

### \*\*مرحله 4: پیاده‌سازی لایه Infrastructure - پیکربندی EF Core ⚠️\*\*

\* \*\*هدف:\*\* راه‌اندازی دسترسی به داده با Entity Framework Core.

\* \*\*در حال انجام /

### \*\*مرحله 5: پیاده‌سازی لایه Infrastructure - Repositoryها و DI ✅\*\*

\* \*\*هدف:\*\* پیاده‌سازی الگوی Repository برای دسترسی به داده و ثبت آنها در سیستم DI.

\* \*\*انجام شده:\*\*

1. پیاده‌سازی `IGenericRepository<T>` و `IUserRepository` در لایه `Domain`.

2. پیاده‌سازی `GenericRepository<T>` و `UserRepository` در لایه `Infrastructure`.

3. ثبت Repositoryها در `Program.cs` Api (`AddScoped`).

### \*\*مرحله 6: پیاده‌سازی لایه Application ⏳\*\*

\* \*\*هدف:\*\* پیاده‌سازی منطق کاربردی (Use Cases) با استفاده از الگوهای CQRS و Mediator.

\* \*\*وظایف:\*\*

1. پیکربندی `MediatR`, `AutoMapper`, `FluentValidation`.

2. پیاده‌سازی Features (CQRS) برای عملیات Auth (Register, Login, RefreshToken, GetUserProfile, ...).

3. پیاده‌سازی Services اپلیکیشن (مثلاً AuthService).

### \*\*مرحله 7: پیاده‌سازی OpenIddict و Endpointهای Auth ⏳\*\*

\* \*\*هدف:\*\* پیاده‌سازی سیستم احراز هویت و اجازه‌دهی با استفاده از OpenIddict.

\* \*\*وظایف:\*\*

1. پیکربندی OpenIddict در Infrastructure و Api.

2. ایجاد Endpointهای Auth (Register, Login, Logout, RefreshToken, ...).

3. ایجاد کنترلرهای API برای مدیریت کاربران، نقش‌ها و مجوزها.

4. پیاده‌سازی JWT Bearer Authentication.

### \*\*مرحله 8: تست‌نویسی و امنیت ⏳\*\*

\* \*\*هدف:\*\* اطمینان از کیفیت و امنیت کد.

\* \*\*وظایف:\*\*

1. نوشتن Unit Tests برای لایه‌های Domain و Application.

2. نوشتن Integration Tests برای APIها و سرویس‌های Infrastructure.

3. اضافه کردن Security Headers, Rate Limiting, Input Validation.

4. بررسی و رفع نقاط ضعف امنیتی.

### \*\*مرحله 9: مستندسازی و استقرار ⏳\*\*

\* \*\*هدف:\*\* آماده‌سازی پروژه برای تولید و استفاده مجدد.

\* \*\*وظایف:\*\*

1. تکمیل مستندات API (Swagger).

2. نوشتن راهنما برای توسعه‌دهندگان.

3. راه‌اندازی CI/CD.

4. کانتینریزیشن با Docker.

5. آماده‌سازی برای اجرای روی Kubernetes (در صورت نیاز).

---

## 📁 \*\*ساختار پوشه‌بندی و کلاس‌های اصلی پروژه\*\*

```

AuthSystem/

├── src/ # کد منبع اصلی

│ ├── AuthSystem.Api # لایه Presentation (API)

│ │ ├── Controllers/ # کنترلرها

│ │ │ └── v1/ # نسخه‌بندی API

│ │ │ └── AuthController.cs # (مثال) کنترلر احراز هویت

│ │ ├── Extensions/ # Extensionهای API

│ │ ├── Middleware/ # Middlewareهای سفارشی

│ │ │ └── ExceptionHandlingMiddleware.cs # (مثال)

│ │ ├── Models/ # مدل‌های API (Requests/Responses)

│ │ │ ├── Auth/ # (مثال) مدل‌های مخصوص Auth

│ │ │ │ ├── RegisterRequest.cs

│ │ │ │ └── LoginResponse.cs

│ │ ├── Filters/ # فیلترها

│ │ ├── Configuration/ # تنظیمات API

│ │ ├── HealthChecks/ # بررسی سلامت

│ │ ├── Properties/ # تنظیمات اولیه پروژه

│ │ ├── Logs/ # (پوشه خروجی لاگ‌ها - توسط Serilog)

│ │ ├── Program.cs # نقطه ورود اصلی API

│ │ ├── appsettings.json # تنظیمات محیط Production

│ │ ├── appsettings.Development.json # تنظیمات محیط Development

│ │ └── AuthSystem.Api.csproj # فایل پروژه API

│ │

│ ├── AuthSystem.Application # لایه Application (منطق کاربردی)

│ │ ├── Interfaces/ # Interfaceهای لایه Application

│ │ │ ├── Repositories/ # Interfaceهای Repository

│ │ │ │ └── IUserRepository.cs # (ارجاع به Domain)

│ │ │ ├── Services/ # Interfaceهای Service

│ │ │ │ └── INotificationService.cs # (مثال) Interface سرویس اپلیکیشنی

│ │ │ └── IUnitOfWork.cs # Interface Unit of Work

│ │ ├── Features/ # ویژگی‌ها (CQRS - Use Cases)

│ │ │ ├── Auth/ # ویژگی‌های احراز هویت

│ │ │ │ ├── Commands/ # دستورات Auth

│ │ │ │ │ ├── RegisterUser/ # (مثال) Feature ثبت نام

│ │ │ │ │ │ ├── RegisterUserCommand.cs

│ │ │ │ │ │ ├── RegisterUserCommandHandler.cs

│ │ │ │ │ │ └── RegisterUserResult.cs

│ │ │ │ │ └── LoginUser/ # (مثال) Feature ورود

│ │ │ │ │ ├── LoginUserCommand.cs

│ │ │ │ │ ├── LoginUserCommandHandler.cs

│ │ │ │ │ └── LoginUserResult.cs

│ │ │ │ └── Queries/ # پرس‌وجوهای Auth

│ │ │ │ └── GetUserProfile/ # (مثال) Feature دریافت پروفایل

│ │ │ │ ├── GetUserProfileQuery.cs

│ │ │ │ ├── GetUserProfileQueryHandler.cs

│ │ │ │ └── UserProfileDto.cs

│ │ │ ├── Users/ # ویژگی‌های کاربران

│ │ │ └── Roles/ # ویژگی‌های نقش‌ها

│ │ ├── DTOs/ # Data Transfer Objects

│ │ │ └── UserDto.cs # (مثال)

│ │ ├── Specifications/ # مشخصه‌ها برای queryهای پیچیده

│ │ ├── Exceptions/ # Exceptionهای سفارشی Application

│ │ │ └── InvalidInputException.cs # (مثال)

│ │ ├── Mappings/ # تنظیمات AutoMapper

│ │ │ └── MappingProfile.cs # (مثال) پروفایل نگاشت

│ │ ├── Validators/ # اعتبارسنجی‌ها (FluentValidation)

│ │ │ └── Auth/ # (مثال) اعتبارسنجی‌های Auth

│ │ │ ├── RegisterUserCommandValidator.cs

│ │ │ └── LoginUserCommandValidator.cs

│ │ ├── Common/ # کامپوننت‌های مشترک Application

│ │ │ ├── Behaviors/ # رفتارهای MediatR

│ │ │ │ └── ValidationBehavior.cs # (مثال) رفتار اعتبارسنجی

│ │ │ └── Models/ # مدل‌های مشترک

│ │ └── AuthSystem.Application.csproj # فایل پروژه Application

│ │

│ ├── AuthSystem.Domain # لایه Domain (منطق دامنه)

│ │ ├── Entities/ # موجودیت‌های اصلی

│ │ │ ├── BaseEntity.cs # کلاس پایه

│ │ │ ├── User.cs # Entity کاربر

│ │ │ ├── Role.cs # Entity نقش

│ │ │ ├── Permission.cs # Entity مجوز

│ │ │ ├── UserRole.cs # Entity ارتباط Many-to-Many کاربر-نقش

│ │ │ ├── RolePermission.cs # Entity ارتباط Many-to-Many نقش-مجوز

│ │ │ ├── RefreshToken.cs # Entity توکن تازه‌سازی

│ │ │ ├── LoginHistory.cs # Entity تاریخچه ورود

│ │ │ └── UserDevice.cs # Entity دستگاه کاربر

│ │ ├── ValueObjects/ # اشیاء مقدار

│ │ │ └── Email.cs # ValueObject ایمیل

│ │ ├── Enums/ # انواع شمارشی

│ │ │ └── DeviceType.cs # Enum نوع دستگاه

│ │ ├── Events/ # رویدادهای دامنه

│ │ ├── Repositories/ # Interfaceهای Repository (تعریف Contract)

│ │ │ ├── IGenericRepository.cs # Interface پایه Repository

│ │ │ └── IUserRepository.cs # Interface Repository کاربر

│ │ ├── Services/ # Interfaceهای سرویس‌های دامنه

│ │ │ ├── IPasswordHasher.cs # Interface هش‌کردن رمز عبور

│ │ │ ├── ITokenGenerator.cs # Interface تولید توکن

│ │ │ ├── IEmailService.cs # Interface ارسال ایمیل

│ │ │ ├── ISmsService.cs # Interface ارسال پیامک

│ │ │ └── IDateTimeProvider.cs # Interface ارائه زمان

│ │ ├── Specifications/ # مشخصه‌های دامنه

│ │ ├── Exceptions/ # Exceptionهای دامنه

│ │ │ ├── DomainException.cs # Exception پایه دامنه

│ │ │ ├── EmailAlreadyExistsException.cs

│ │ │ ├── UsernameAlreadyExistsException.cs

│ │ │ ├── InvalidCredentialsException.cs

│ │ │ ├── InvalidRefreshTokenException.cs

│ │ │ ├── InsufficientPermissionsException.cs

│ │ │ └── UserNotFoundException.cs

│ │ └── AuthSystem.Domain.csproj # فایل پروژه Domain

│ │

│ ├── AuthSystem.Infrastructure # لایه Infrastructure (پیاده‌سازی)

│ │ ├── Persistence/ # پایداری داده (EF Core)

│ │ │ ├── Contexts/ # Contextهای EF

│ │ │ │ └── ApplicationDbContext.cs # DbContext اصلی

│ │ │ ├── Repositories/ # پیاده‌سازی Repositoryها

│ │ │ │ ├── GenericRepository.cs # پیاده‌سازی پایه Repository

│ │ │ │ └── UserRepository.cs # پیاده‌سازی Repository کاربر

│ │ │ ├── Configurations/ # تنظیمات Entity (IEntityTypeConfiguration)

│ │ │ │ ├── UserConfiguration.cs

│ │ │ │ ├── RoleConfiguration.cs

│ │ │ │ ├── PermissionConfiguration.cs

│ │ │ │ ├── UserRoleConfiguration.cs

│ │ │ │ ├── RolePermissionConfiguration.cs

│ │ │ │ ├── RefreshTokenConfiguration.cs

│ │ │ │ ├── LoginHistoryConfiguration.cs

│ │ │ │ └── UserDeviceConfiguration.cs

│ │ │ └── Migrations/ # Migrationهای دیتابیس (EF)

│ │ ├── Services/ # پیاده‌سازی سرویس‌های دامنه

│ │ │ ├── EmailService/ # سرویس ایمیل

│ │ │ │ └── EmailService.cs # پیاده‌سازی IEmailService

│ │ │ ├── SmsService/ # سرویس پیامک

│ │ │ │ └── SmsService.cs # پیاده‌سازی ISmsService

│ │ │ ├── JwtService/ # سرویس JWT

│ │ │ │ └── JwtService.cs # پیاده‌سازی ITokenGenerator

│ │ │ └── PasswordHasher/ # سرویس هش‌کردن رمز عبور

│ │ │ └── PasswordHasher.cs # پیاده‌سازی IPasswordHasher

│ │ ├── Identity/ # پیاده‌سازی Identity/OpenIddict

│ │ ├── Cache/ # کش‌ها (Redis)

│ │ ├── Logging/ # لاگ‌نویسی (در صورت نیاز به سفارشی‌سازی بیشتر)

│ │ ├── Messaging/ # پیام‌رسانی (Event Bus)

│ │ ├── External/ # سرویس‌های خارجی (Cloud Storage, Payment)

│ │ ├── Common/ # کامپوننت‌های مشترک Infrastructure

│ │ └── AuthSystem.Infrastructure.csproj # فایل پروژه Infrastructure

│ │

│ ├── AuthSystem.Shared # مدل‌ها و DTOهای مشترک (در صورت نیاز)

│ │ ├── DTOs/

│ │ ├── Enums/

│ │ ├── Constants/

│ │ ├── Exceptions/

│ │ └── Interfaces/

│ │

│ └── AuthSystem.Messaging # کانترکت‌های پیام‌رسانی (در صورت استفاده از Message Broker)

│ ├── Events/

│ ├── Commands/

│ └── Contracts/

│

├── tests/ # تست‌ها

│ ├── AuthSystem.UnitTests # تست‌های واحد

│ │ ├── Application/ # تست‌های لایه Application

│ │ │ └── Features/ # (مثال) تست Featureهای Auth

│ │ │ └── Auth/

│ │ │ ├── RegisterUserCommandHandlerTests.cs

│ │ │ └── LoginUserCommandHandlerTests.cs

│ │ ├── Domain/ # تست‌های لایه Domain

│ │ │ └── Entities/ # (مثال) تست Entityها

│ │ │ └── UserTests.cs

│ │ ├── Infrastructure/ # تست‌های لایه Infrastructure

│ │ │ └── Persistence/ # (مثال) تست Repositoryها

│ │ │ └── Repositories/

│ │ │ └── UserRepositoryTests.cs

│ │ └── AuthSystem.UnitTests.csproj

│ │

│ ├── AuthSystem.IntegrationTests # تست‌های ادغام

│ │ ├── Api/ # تست‌های API

│ │ ├── Database/ # تست‌های دیتابیس

│ │ ├── Services/ # تست‌های ادغام سرویس‌ها

│ │ └── AuthSystem.IntegrationTests.csproj

│ │

│ ├── AuthSystem.FunctionalTests # تست‌های عملکردی (End-to-End)

│ │ └── AuthSystem.FunctionalTests.csproj

│ │

│ └── AuthSystem.LoadTests # تست‌های بار (در صورت نیاز)

│ └── AuthSystem.LoadTests.csproj

│

├── docs/ # مستندات

│ ├── api/ # مستندات API

│ │ ├── openapi/ # OpenAPI/Swagger Specification

│ │ └── postman/ # Postman Collections

│ ├── architecture/ # مستندات معماری

│ │ ├── diagrams/ # دیاگرام‌ها (UML, C4, etc.)

│ │ └── decisions/ # ADRها (Architectural Decision Records)

│ └── guides/ # راهنماها (Developer Guide, User Guide)

│

├── scripts/ # اسکریپت‌ها

│ ├── build/ # اسکریپت‌های build

│ ├── deploy/ # اسکریپت‌های استقرار

│ ├── database/ # اسکریپت‌های دیتابیس (Seed Data)

│ └── dev/ # اسکریپت‌های توسعه

│

├── docker/ # فایل‌های Docker

│ ├── development/ # محیط توسعه (Dockerfile, docker-compose)

│ ├── production/ # محیط تولید (Dockerfile, docker-compose)

│ └── k8s/ # فایل‌های Kubernetes (Helm Charts, Manifests)

│

├── .github/ # GitHub Actions (CI/CD)

│ └── workflows/ # Workflows

│

├── .devcontainer/ # Dev Container Config (برای GitHub Codespaces یا VS Code Dev Containers)

├── .vscode/ # تنظیمات VS Code

├── .config/ # تنظیمات پروژه (dotnet-tools.json)

├── .gitignore # فایل‌هایی که نباید در Git ذخیره بشن

├── AuthSystem.sln # فایل Solution اصلی

├── global.json # تنظیمات سطح solution (نسخه SDK)

└── README.md # توضیحات کلی پروژه

```