**Products and Categories Management System**

Contents

[Backend Architecture 2](#_Toc188054417)

[Project Overview 2](#_Toc188054418)

[Directory Structure 2](#_Toc188054419)

[API: 2](#_Toc188054420)

[Application: 3](#_Toc188054421)

[Common: 3](#_Toc188054422)

[Domain**:** 3](#_Toc188054423)

[Infrastructure**:** 3](#_Toc188054424)

[FrontEnd Architecture 4](#_Toc188054425)

[Architecture Overview 4](#_Toc188054426)

[Directory Structure &Description of Each Folder: 4](#_Toc188054427)

[Core 4](#_Toc188054428)

[Guards 4](#_Toc188054429)

[Interceptors 4](#_Toc188054430)

[Layout 4](#_Toc188054431)

[ Services 4](#_Toc188054432)

# GitHub Repo

Backend: <https://github.com/fatmazayedsayed/ProductManagement-BE>

FrontEnd: <https://github.com/fatmazayedsayed/ProductManagement-FE>

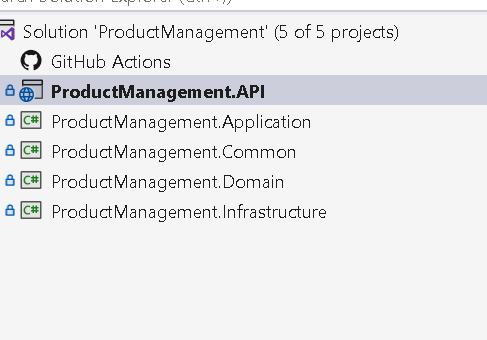
Admin account:

Username: admin

password: admin

# Backend Architecture

## Project Overview

* **Project Name and Purpose**: **ProductManagement** the project handles all APIs related to admin portal and user portal.
* **Technology Stack**:  NET Core web API, UOW, FastEndpoints ,Code first, JWT,Nlog
* **Architecture Overview**:   monolithic architecture DDD

## Directory Structure

Solution is divided into 5 layers

### API:

It has all endpoints that will be called from frontend

- authentication: handle user login

-category: handle category CRUD operations

-Product: handle product CRUD operations

-dashboard: handle system quick overview

-extensions: handle registration for handlers that used during integration with application layer

-filters: global exception handling

-program.cs: register and configure FastEndPoint and swagger

### A screenshot of a computer Description automatically generatedApplication:

Handling the core business of the project.

- Each entity in the project operations are separated in its related folder, (using the same folder name in API for easy access)

Each command is in a separated folder with its related DTO used and it’s handler.

- abstraction folder contains all system interfaces

-identity folder handle login and authentication and JWT generation

### Common:

Contains helpers and enums

### Domain**:**

Contains database entities

### Infrastructure**:**

Contains database migrations,

repositories to interact with database

Data seeding to initialize database with basic data

# A screenshot of a computer Description automatically generatedFrontEnd Architecture

## Architecture Overview

  The project follows a **Modular Architecture**. This design approach allows for clean separation of concerns and ease of maintenance by organizing the application into distinct feature modules.

* **Dependencies:**
* **Angular Core:** **17.2.0:**

@angular/animations, @angular/common, @angular/compiler, @angular/core

## Directory Structure &Description of Each Folder:

Core:  Contains core functionality, including:

Guards: Such as auth-guard for route protection.

Interceptors: For managing token.

Layout: Manages reusable UI components like header ,side navigation, and home page.

* Services: Contains services for API endpoint communication.