🧩 Inventory Management System

Project 1 – Programming Fundamentals (CA-PRFND)

📘 Introduction

This project is a **prototype Inventory Management System** developed in **C# using .NET 9 and Visual Studio Code**.
It captures and manages inventory items using **EF Core and SQLite**, exposing a **RESTful API** with Swagger/OpenAPI support.

📊 Program Flow (Mermaid Diagram)

```
flowchart TD
    A[Start API] --> B[Swagger / Root Endpoint]
    B --> C{Select Endpoint}

    C -->|GET items| D[Fetch all items from DB]
    D --> E[Return JSON list]

    C -->|GET item by ID| F[Fetch item by ID]
    F --> G{Item Exists?}
    G -->|Yes| H[Return Item JSON]
    G -->|No| I[Return 404 Not Found]

    C -->|POST items| J[Receive Item JSON]
    J --> K{Validate Input}
    K -->|Valid| L[Insert into DB]
    K -->|Invalid| M[Return 400 Bad Request]
    L --> N[Return Created Response]
```

🧰 Setup Instructions

Prerequisites • .NET 9 SDK • Visual Studio Code or Visual Studio • SQLite CLI (optional)

Build & Run

cd InventoryAPI dotnet restore dotnet build dotnet run

API will run on: • HTTPS: https://localhost:7255 • HTTP: http://localhost:5091

Database Migrations

dotnet ef migrations add InitialCreate --project InventoryAPI dotnet ef database update --project InventoryAPI

💾 Database Model

Item.cs

public class Item { public int Id { get; set; } public string FirstName { get; set; } public string LastName { get; set; } public double Price { get; set; } }

InventoryDbContext.cs

using Microsoft.EntityFrameworkCore;

public class InventoryDbContext : DbContext { public InventoryDbContext(DbContextOptions options) : base(options) { }

```
public DbSet<Item> Items { get; set; }
```

}

⚙️ API Endpoints

Endpoint Method Description / GET Health check / Root message /items GET Fetch all items /items/{id} GET Fetch a single item by ID /items POST Add a new item

Swagger UI: https://localhost:7255/swagger

🔧 Development Highlights • Minimal API with ASP.NET Core • EF Core SQLite integration • Input validation for IDs and prices • Async/await for database operations • Swagger/OpenAPI for endpoint testing

---

🧩 Folder Structure

InventoryAPI/ │ ├── Program.cs ├── Item.cs ├── InventoryDbContext.cs ├── appsettings.json ├── appsettings.Development.json ├── Properties/ ├── bin/ ├── obj/ └── InventoryAPI.csproj

---

👨‍💻 Author

Marc Cavada Programming Fundamentals – CDI College Project: CA_PRFND – Inventory Management System

---

Next, we can **convert this markdown into a PDF**. On macOS or VS Code, here are two simple options:

Option 1 – VS Code Markdown PDF extension

1. Install `Markdown PDF` extension.
2. Open this README `.md` file.
3. Press `Cmd+Shift+P` → `Markdown PDF: Export (pdf)`

Option 2 – Using Pandoc CLI

```
brew install pandoc
pandoc README.md -o InventoryAPI.pdf --pdf-engine=xelatex
```