# 06 Review Exercise Energy System API

## Overview

In this assignment, you will develop a REST API for managing an energy system. We can register buildings, sensors and sensor logs. We can also get the stats of the sensor logs.

**This assignment reflects the level of the exam but does NOT cover everything you are expected to know.**

A **building** has the following properties:

- Id
- Name
- Address
- City
- Country
- PostalCode
- Sensors (

**List of sensors for the building**)

A **sensor** has the following properties:

- Id
- Name
- Type (Temperature, CO2, etc.)
- Unit (Example: °C, °F, kWh, etc.)

A **sensorlog** has the following properties:

- Id
- BuildingId
- SensorId

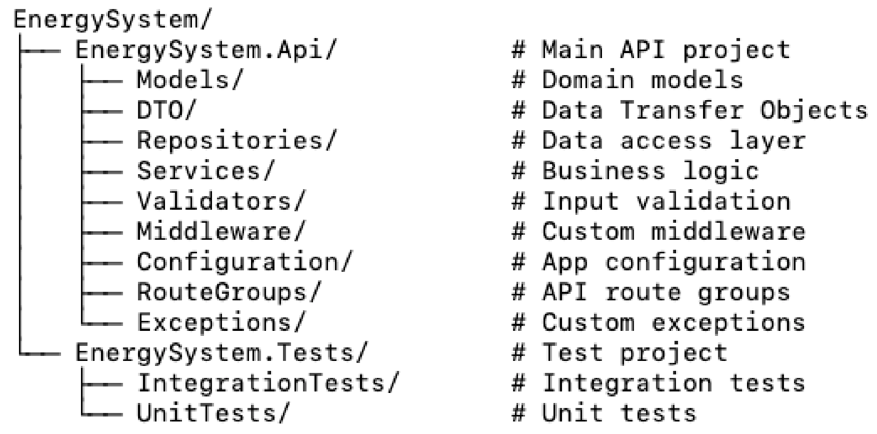
- Value
- Unit
- Timestamp

The application has the following endpoints:
- POST /api/buildings - Create a building
- GET /api/buildings - Get all buildings
- GET /api/buildings/{id} - Get a building by id
- POST /api/buildings/sensors - Create a sensor for a building
- POST /api/sensorlogs - Create a sensor log
- GET /api/sensorlogs - Get all sensor logs
- GET /api/sensorlogs/stats/{buildingId}/{sensorId}  Get the stats of a sensor log for the given building and sensor. The result contains the min, max and average of the sensor logs for the given building and sensor.

Use **DTOs**, **RouteGroups** and **custom exceptions** (like SensorNotFoundException, BuildingNotFoundException, etc.) where possible.

You can only use
**2 collections in the database: buildings and sensorlogs.** The **sensors** are stored in the database **together** with the **building**.

The project structure looks like this:



```
EnergySystem/
├── EnergySystem.Api/           # Main API project
│   ├── Models/                 # Domain models
│   ├── DTO/                    # Data Transfer Objects
│   ├── Repositories/           # Data access layer
│   ├── Services/               # Business logic
│   ├── Validators/             # Input validation
│   ├── Middleware/             # Custom middleware
│   ├── Configuration/          # App configuration
│   ├── RouteGroups/            # API route groups
│   └── Exceptions/             # Custom exceptions
└── EnergySystem.Tests/         # Test project
    ├── IntegrationTests/       # Integration tests
    └── UnitTests/              # Unit tests
```

The root namespace is **EnergySystem.Api.**

**Technical Requirements**

- Use .NET 9
- Use MongoDB as the database
- Implement services and repositories for business logic and data access
- Use FluentValidation for input validation
- Return proper HTTP status codes and error responses
- Apply dependency injection throughout the application
- Use AutoMapper for object mapping
- Write unit tests and integration tests