Python Home Task

# Sensor Data Monitoring and Alerting System

Instructions:

* Write clean and coherent code
* Keep SOLID and readable code
* If you make any assumption, please comment it in the readme file
* On any question or doubt, feel free to reach out

Objective:

Create a Python-based service to monitor sensor data. The system should be able to read sensor data from various types of sensors (temperature, humidity, pressure, etc.), validate the data based on configured rules, and alert an alert service if invalid data is detected. The alert service should notify on invalid data.

Requirements:

Sensor Connectors: Create classes for different types of sensors (e.g., TemperatureSensor, HumiditySensor, PressureSensor). Each sensor connector should simulate receiving data from a real sensor (you can generate random values).

Configuration File: Use a JSON or YAML configuration file to specify which sensors are connected and what constitutes valid data for each sensor.

Main Service: The main service should read the configuration file to determine which sensors to monitor and what data is considered valid. It should continuously read data from the sensors and validate it.

Alert Service: Create a separate service responsible for handling alerts. The main service should notify the alert service if it detects invalid data.

Inter-Service Communication: Use any communication protocol to enable messaging between the services. For example, you can use HTTP RESTful APIs or message queues.

Clean Code: Code should be clean, well-organized, and should include comments explaining the important parts.

Technical Specs:

Sensor Connector: Implement at least 3 different types of sensor connectors.

Configuration File Example:

Json



Main Service:

Read and parse the configuration file.

Initialize the corresponding sensor connectors based on the configuration.

Continuously read data and validate against the configured valid range.

If invalid data is found, send an alert to the Alert Service.

Alert Service:

Receive alerts from the main service and log them.

Bonus Points:

* Asynchronous Programming: Use asynchronous programming constructs for better performance.
* Testing: Include unit tests for each component.
* Enhance alerting service with different types of alerting methods (email, slack)