MCP POC – Factory Monitoring

Overview

This repository contains a **Proof of Concept (POC)** demonstrating how **Model Context Protocol (MCP)** can be used to expose factory sensor data through a lightweight API and consume it via an agent for insights.

The POC focuses on: - Exposing sensor statistics (temperature, PPFD, volts) - Using a FastAPI-based MCP server - Consuming MCP endpoints via a rule-based agent - Following secure coding practices (no secrets committed to GitHub)

Architecture

```
[InfluxDB]
     │
     │ (Flux Queries)
     ▼
[MCP Server - FastAPI]
     │
     │ (HTTP / MCP-style endpoints)
     ▼
[Agent]
     │
     ▼
[Tabular Output + Insights]
```

Files in This Repository

File	Description
`mcp_poc.py`	MCP server implemented using FastAPI. Exposes sensor statistics endpoints.
`agent.py`	Agent that queries MCP endpoints and generates discussion-style insights.
`MCP_POC_Documentation.docx`	Detailed documentation explaining the POC design and flow.
`.gitignore`	Ensures secrets, caches, and unnecessary files are not committed.

MCP Server (`mcp_poc.py`)

The MCP server: - Connects to **InfluxDB** - Queries sensor data using **Flux** - Calculates **min**, **max**, and **latest** values per topic - Exposes REST endpoints:

Available Endpoints

Endpoint	Description
`/temperature`	Temperature statistics per sensor
`/ppfd`	PPFD (light intensity) statistics
`/volts`	Voltage statistics

Agent (`agent.py`)

The agent: - Accepts natural language-like input from the user - Identifies sensor type and metric (min / max / latest) - Calls the appropriate MCP endpoint - Displays results in a table - Generates **human-readable insights** for discussion and analysis

Example questions: - "What is the max temperature?" - "Show latest volts" - "Lowest PPFD readings"

Security & Configuration

Secrets Handling

- No tokens or secrets are committed to GitHub
- InfluxDB token is read from an environment variable

Example:

```
export INFLUX_TOKEN="<your_influx_token>"
```

Or on Windows (PowerShell):

```
$env:INFLUX_TOKEN="<your_influx_token>"
```

How to Run

1. Start the MCP Server

```
uvicorn mcp_poc:app --reload --port 5005
```

2. Run the Agent

```
python agent.py
```

Purpose of This POC

This POC demonstrates: - How MCP can decouple data access from consumers - How agents can reason over MCP-exposed tools - A secure, production-aware approach to API and agent design

It is intended for **learning, experimentation, and architectural validation**, not as a production deployment.

Future Enhancements

- Replace rule-based agent with LLM-based reasoning
- Add authentication and authorization
- Integrate visualization tools (e.g., Grafana API)
- Extend MCP tools for additional sensor types

Author

Kavita Khan
MCP POC – Factory Monitoring