# OPC-UA Data Pipeline: Final Project Assignment

## Overview

For your final project, you will design and implement a complete end-to-end data pipeline that collects real-time industrial data from an OPC-UA simulation server, processes it through multiple layers, and visualizes the results in a monitoring dashboard. This project demonstrates your understanding of IoT data ingestion, stream processing, time-series databases, and data visualization.

In total, there are 100 points to be gained in this assignment. The points are split into the following categories:

| **Category** | **# of points** |
|---|---|
| OPC-UA simulation server | 30 |
| Data Pipeline | 20 |
| TimescaleDB table structure | 5 |
| TimescaleDB continuous aggregate | 15 |
| Grafana Dashboard | 15 |
| Documentation | 15 |

## Project Objectives

By completing this project, you will:

- Build a working OPC-UA simulation server
- Design and implement a multi-stage data pipeline with message brokers and event enrichment
- Configure a time-series database for efficient storage and querying
- Create real-time visualizations of industrial data
- Package all parts of the data Pipeline in a single docker compose setup for easy testing & deployment.

# Project components

## OPC-UA Simulation Server (30 Points)

Implement a functional OPC-UA simulation server in a programming language of your choice. Your server must accurately represent all fields specified in one of the machine descriptions provided during the course. The server should simulate realistic data variations and maintain stable connections for continuous data transmission.

## Data Pipeline (20 Points)

Your data pipeline must consist of four integrated components:

**OPC-UA to MQTT Agent:** Create an agent that reads data from your OPC-UA simulation server and converts it to MQTT protocol for transmission to a message broker.

**MQTT Broker:** Set up and configure an MQTT broker that acts as the central hub for initial data distribution.

**Hydration Agent:** Implement an agent that retrieves context data from Redis, enriches incoming MQTT events with this contextual information, and publishes the enriched events to a Kafka queue.

**Kafka to Database Agent:** Build an agent that consumes messages from your Kafka queue and writes the data to your PostgreSQL database.

All components must work together seamlessly, with proper error handling and data validation at each stage.

## TimescaleDB table structure (5 Points)

Create a correctly configured TimescaleDB hypertable optimized for your time-series data. The table schema should reflect the fields from your machine description and include appropriate data types, constraints, and indexes. Clearly document your table structure, including all columns, their purposes, and any design decisions you made.

## Continuous aggregate (15 Points)

Set up a continuous aggregate on your TimescaleDB hypertable that performs regular time-based aggregations of your data (e.g., hourly, daily). This aggregate should compute meaningful statistics such as averages, minimums, maximums, or other relevant metrics. Document your aggregation strategy and explain how it supports your visualization needs.

## Grafana Dashboard (15 Points)

Create a Grafana dashboard that visualizes data from your TimescaleDB table. Your dashboard must include:

**Short-term visualization:** Graphs showing recent data (e.g., last 24 hours) to monitor current system behavior

**Long-term visualization:** Graphs showing historical trends (e.g., last month or longer) to identify patterns and trends

Your dashboard should be intuitive, well-labeled, and use appropriate chart types for your data.

## Project documentation (15 Points)

Provide comprehensive documentation that includes:

**OPC-UA Server Overview:** A description of all available fields in your OPC-UA simulation server and what they represent.

**System Architecture:** Links to all relevant user interfaces (Redpanda console, Grafana dashboard, etc.) and a complete list of ports used by each service.

**Database Schema:** A clear explanation of your TimescaleDB table structure, including all columns and their meanings.

**Continuous Aggregate Details:** Documentation of how your continuous aggregate is configured, what time intervals it uses, and what metrics it computes.

**Grafana Configuration:** Descriptions of your dashboard panels, the queries used, and how to interpret the visualizations.

## Submission Requirements

Submit your project as a complete package including:

- All source code for the OPC-UA server and pipeline agents
- Database schema and setup scripts
- Grafana dashboard configuration (exported as JSON)
- Complete documentation file (PDF or Markdown format)
- A README file with instructions for running your System
- A docker-compose.yaml file to run the entire stack

For your submission you can either

a) Provide a link to a Github repository containing all the relevant files
b) Upload all required files as a zip archive

## Final remarks

- All components should run reliably for extended periods without manual Intervention
- Provide a Dockerfile for your OPC-UA Server so that it can be built during testing
- Test your entire pipeline end-to-end before submission to ensure data flows correctly through all stages
- Include comments in your code explaining complex logic