Capstone Project:

Smart Manufacturing: Fuel Efficiency and Turbine Health Analytics on Azure Cloud

Business Scenario

Project Overview

This capstone project simulates a real-world industrial IoT analytics workflow using high-frequency gas turbine sensor data. Learners will develop end-to-end data engineering, transformation, analytics, machine learning, and deployment pipelines.

The project replicates operational scenarios involving fuel consumption optimization, early failure detection, anomaly analysis, and performance benchmarking in manufacturing.

Business Context

Modern manufacturing environments—especially in sectors like shipping, aerospace, and energy—rely heavily on real-time monitoring of complex machines like gas turbines. These systems are instrumented with dozens of sensors, generating vast volumes of telemetry data related to temperature, torque, pressure, fuel flow, etc.

Proper ingestion, transformation, and analysis of this data can help businesses:

- Detect anomalies and pre-empt machine failures.
- Analyze fuel inefficiency or overuse under different operational loads.
- Benchmark turbines based on wear and tear (decay coefficients).
- Predict component degradation for proactive maintenance.
- Visualize turbine health and drive data-driven interventions.

Project Objectives

1. Develop Python scripts and SQL models to ingest and analyze time-series turbine sensor data.
2. Design ETL pipelines and data stores (SQL/NoSQL) to manage large sensor logs.
3. Perform multivariate analysis and visualize turbine behavior and fuel usage.
4. Build predictive models for failure detection and anomaly analysis.
5. Deploy the solution on Azure Cloud using Blob Storage, Data Factory, Synapse, Databricks, and Cosmos DB.

Project Dataset

Sensor Telemetry Dataset (CSV format)

Field Name	Description
Lp	Lever position
V	Ship speed (knots)
GTT	Gas Turbine shaft torque (kN·m)
GTn	Gas Turbine revolutions per minute (rpm)
GGn	Gas Generator revolutions per minute (rpm)
Ts	Starboard Propeller Torque (kN)
Tp	Port Propeller Torque (kN)
T48	HP Turbine exit temperature (°C)
T1	Compressor inlet air temperature (°C)
T2	Compressor outlet air temperature (°C)
P48	HP Turbine exit pressure (bar)
P1	Compressor inlet pressure (bar)
P2	Compressor outlet pressure (bar)
Pexh	Exhaust gas pressure (bar)
TIC	Turbine Injection Control (%)
mf	Fuel flow rate (kg/s)
decay_coeff_comp	Compressor decay coefficient
decay_coeff_turbine	Turbine decay coefficient

Solution Outline

1. Data Ingestion & Preprocessing

- Read time-series CSV sensor logs using Python.
- Clean missing, null, and anomalous sensor values.
- Normalize and smooth readings for visual clarity.
- Calculate derived KPIs (e.g., torque differential, temperature-pressure ratios).

2. Data Modeling & Storage

- Create SQL schema to store structured sensor data.
- Use MongoDB for flexible anomaly logs or maintenance alerts.
- Build entity relationships based on turbine ID and timestamps.

3. ETL & Aggregation

- Build modular ETL pipelines using Python and PySpark.
- Aggregate metrics per turbine/hour/day (fuel usage, temp spikes).
- Apply window functions for rolling average or sensor drift detection.

4. Multivariate Analytics & KPI Reporting

- Explore feature relationships: fuel flow vs torque, decay vs pressure.
- Identify underperforming turbines based on energy efficiency.
- Visualize temporal changes using Power BI dashboards.

5. Predictive Modeling

- Classify turbine health using logistic regression or decision trees.
- Cluster operating conditions via K-Means.
- Use decay coefficients to estimate maintenance needs.

6. Cloud Deployment (Azure)

- Ingest raw files into Azure Blob Storage.
- Build transformation pipelines using Azure Data Factory.
- Store structured data in Azure Synapse and unstructured in Cosmos DB.
- Analyze logs via Azure Databricks notebooks.
- Visualize turbine performance trends via Power BI Service.

Capstone Phases with learner Tasks & Deliverables

Phase 1: Programming Skill and Data Extraction using Python

Objective: Build foundational coding logic for sensor data ingestion and anomaly detection.

- Develop algorithms and flowcharts to simulate edge-device streaming logic.
- Modular Python code to clean raw sensor data.
- Apply exception handling and logging for invalid readings.
- Use Pandas, NumPy for EDA and visualization.
- Generate unit test cases using PyTest.

Phase 2: Software Engineering Principles & Data Architecture

Objective: Translate business requirements into scalable system design.

- Understand SDLC, SRS, and HLD using turbine monitoring requirements.
- Create UML diagrams for component architecture.
- Design SRS for predictive maintenance and KPI generation system.

Phase 3: SQL, Data Structures & Visual Summaries

Objective: Build relational schema and derive KPIs from structured sensor logs.

- Design normalized tables for sensor_readings, anomalies, turbine_metadata.
- Write SQL queries: joins, aggregates, window functions.
- Visualize output trends (fuel usage, decay) via Seaborn/Matplotlib.

Phase 4: ETL, PySpark, and NoSQL Design

Objective: Create pipelines and flexible storage for semi-structured logs.

- Develop PySpark jobs to ingest and transform sensor data.
- Store alerts and maintenance flags in MongoDB.
- Create timestamp-based ETL aggregations (hourly, daily, weekly).

Phase 5: REST APIs and FastAPI Integration

Objective: Make sensor predictions available to external teams via APIs.

- Build FastAPI-based endpoints for turbine health status and anomaly alerts.
- Integrate SQLAlchemy + FastAPI to serve live turbine data.
- Swagger docs to expose model predictions and ingestion status.

Phase 6: Cloud Deployment on Azure (ADF, Synapse, Power BI)

Objective: Operationalize the analytics system on Azure.

- Store raw data in Blob and processed in Delta Tables (Databricks).
- Automate ingestion using Azure Data Factory pipelines.
- Use Synapse SQL to perform interactive queries.
- Build Power BI dashboard for turbine metrics (fuel usage, decay, status).

Final Deliverables Summary

Stage	Key Deliverables
1	Python ingestion scripts, KPI calculator functions, test cases
2	SRS, HLD, UML diagrams for turbine monitoring platform
3	SQL schema, dashboard plots, anomaly reports

4	PySpark ETL job, MongoDB alert repository
5	REST APIs with FastAPI, model prediction endpoints
6	Azure pipelines (ADF), dashboards (Power BI), analytics notebooks (Databricks)