# 📄 Real-time IoT Data Pipeline Project Documentation Proposal

This document serves as the formal **Project Planning, Stakeholder Analysis, and Database Design** submission for the "Real-time IoT Data Pipeline" project under the **Digital Egypt Pioneers Initiative (DEPI)**.

---

## 1. Project Overview & Scope

The **Real-time IoT Data Pipeline** project establishes a robust, end-to-end data engineering solution designed to monitor and analyze simulated city sensor data, such as temperature and humidity. The primary goal is to build an **orchestrated system** that demonstrates mastery of modern data processing techniques. This includes:

- **Data Simulation and Ingestion:** Creating a Python generator to realistically simulate sensor data streams.
- **Batch Processing (ETL):** Implementing a foundational pipeline to clean, transform (e.g., flag anomalies), and load data into a structured **PostgreSQL** data warehouse for historical analysis.
- **Streaming Analytics:** Developing a real-time component to process incoming data instantly and generate **immediate alerts** when critical thresholds are breached.
- **Visualization:** Delivering a comprehensive, real-time dashboard built on **Streamlit** to provide system operators and city managers with actionable insights and monitoring capabilities.

This project is a critical demonstration of utilizing cloud-native concepts and big data architectures to support data-driven decision-making within the national digital transformation framework.

| Item | Details |
|---|---|
| **Institution** | Ministry of Communications & Information Technology (MCIT), Egypt |
| **Program** | Digital Egypt Pioneers Initiative (DEPI) |
| **Current Status** | **Implementation Complete (All Milestones)** with additional innovative features. |

| | |
|---|---|
| **GitHub Repository** | $\text{[github.com/hamed11010/data-engineering](https://github.com/hamed11010/data-engineering)}$ |
| **Leader Email** | $\text{ahmedhamedahmed911@gmail.com}$ |
| **Deadline** | November 1, 2025 |

---

## 2. Stakeholder Analysis

This project involves several key internal and external stakeholders, each with specific interests and influence.

**Internal Project Team (High Influence / High Interest)**

| Name | Role | Core Responsibilities |
|---|---|---|
| **Ahmed Hamed (Leader)** | Project Lead & Data Pipeline Developer | **Oversaw entire workflow**, environment setup (Docker, Python), API integration/CSV generation, and overall project coordination. |
| **Mathew Samy** | Data Engineer | Built **Batch ETL** scripts, defined **Database Schema**, and implemented transformation logic (averages, anomaly flagging). |
| **Zain Aldin Salah** | Streaming Specialist | Implemented **Streaming Analytics** (real-time alerts/thresholds) and planned Kafka integration. (Contributed to Milestone 3). |
| **Mohammed Esmail** | Data Visualization Engineer | Designed **Streamlit Dashboard** layout, handled plotting (temperature & humidity trends). (Contributed to Milestone 3 & 4). |

| | | |
|---|---|---|
| **Marco Nashaat** | System Integrator | Managed system orchestration, connections between ETL/Database/Dashboard, and version control setup. (Contributed to Milestone 3). |
| **Nour Hatem Mahmoud** | Documentation & Reporting Specialist | Prepared **Documentation**, summarized Technical Report, and prepared Dashboard Showcase slides. (Contributed to Milestone 4). |

**External Stakeholders (High Influence / Varying Interest)**

| Stakeholder Group | Interest | Expectations |
|---|---|---|
| **MCIT Program Management** | High | Successful, on-time completion; High-quality, robust solution; Compliance with all project requirements. |
| **Project Instructors/Mentors** | High | Demonstration of mastery in Data Pipelines, Big Data, and Cloud-Native Processing (Modules 5 & 6). |
| **Potential End Users (e.g., City Managers)** | Moderate | A reliable, **real-time dashboard** for monitoring city-wide sensor data and receiving immediate alerts. |

---

## 3. Database Design

The data ingestion process centers around storing high-volume, time-series sensor readings.

### 3.1 Technology Stack

- **Database System: PostgreSQL**
- **Deployment:** Docker container for isolated, consistent, and scalable deployment.

### 3.2 Table Schema: readings

The single-table design is optimized for fast writes and analytical querying of time-series data.

| Column Name | Data Type (PostgreSQL) | Description |
|---|---|---|
| **timestamp** | **TIMESTAMP** | Primary key candidate. The exact time the sensor reading was taken. Critical for time-series analysis. |
| **city** | **TEXT** | The location (city) where the sensor reading was recorded. |
| **temperature_c** | **FLOAT** | The temperature reading in Celsius. Used for batch averaging and real-time alerting. |
| **humidity** | **FLOAT** | The relative humidity reading. Used for batch averaging and trend analysis. |
| **pressure** | **FLOAT** | The atmospheric pressure reading. |
| **wind_speed** | **FLOAT** | The speed of the wind. |
| **weather** | **TEXT** | A description of the current weather (e.g., 'Sunny', 'Rainy'). |

---

## 4. Project Planning & Milestone Completion

The project was executed across four structured milestones, with a focus on delivering robust, production-ready components.

### 4.1 Milestone Status Summary

| Milestone | Key Deliverable(s) | Completion Status | Bonus/Innovation Feature |
|---|---|---|---|

| | | | |
|---|---|---|---|
| **1. Data Simulation** | Python Generator Script & Sample Logs | **Completed** ✅ | **CSV $\rightarrow$ Kafka-like Simulation**: The generator can write to a file *or* simulate message publishing to a queue. |
| **2. Batch ETL** | ETL Pipeline Script (Python/SQL) & Processed Dataset | **Completed** ✅ | **Anomaly Detection**: Transformation logic includes a script to flag simple temperature anomalies (e.g., outside $3\sigma$ range) before loading. |
| **3. Streaming Analytics** | Streaming Pipeline Setup & Alert Logic Code | **Completed** ✅ | **Multi-Level Alerting**: Implemented 'Warning' (high) and 'Critical' (severe) alerts for temperature threshold breaches, providing better user context. |
| **4. Dashboard & Report** | Streamlit Dashboard & Final Report/Slides | **Completed** ✅ | **Interactive Time-Range Filtering**: Added dynamic time-range selection on the dashboard for both batch-analyzed data and recent real-time views. |

**4.2 UI/UX Design (Dashboard Tool: Streamlit)**

The primary user interface is the **Streamlit Dashboard**, designed for clarity and real-time decision-making.

| Section | Content | Purpose |
|---|---|---|
| **Top Panel** | **Real-Time Alert Feed** | Displays immediate, color-coded alerts (Warning/Critical) from the streaming pipeline. |
| **Key Metrics** | Single-value cards (Kibana-style) | Displays the **last recorded** temperature, humidity, and wind speed. |

| | | |
|---|---|---|
| **Historical Trends** | Line Charts | **Batch Data View**: Plot of daily average temperature and humidity over the last 30 days. |
| **Live Chart** | Dynamic Line Chart | **Streaming Data View**: Rolling 1-hour plot of recent sensor readings, updating every 5 seconds. |

---

## 📢 Important Note on Future Development

**Please be advised that this project is subject to continuous development and innovation.** While the core requirements and milestones outlined in this document have been successfully completed, the team is actively engaged in developing and integrating additional features, performance optimizations, and advanced analytics models for future iterations. Therefore, supplementary system components or functionalities not explicitly detailed in this proposal may be present in the final deployed version.