Near Real-Time PM2.5 Data Pipeline with Prefect, Streamlit, and ARIMA Forecasting

📌 Project Overview

This project is a final assignment for the **DSI321: Big Data Infrastructure** course. It builds an automated data pipeline that collects, processes, and visualizes near real-time PM2.5 air quality data in Bangkok using **Prefect**, **Docker**, and **Streamlit**.

Beyond data ingestion and visualization, the project implements **time series forecasting** with **ARIMA models** to predict PM2.5 levels and help stakeholders anticipate air quality trends.

📊 Motivation

Air pollution, especially PM2.5, remains a critical health issue in Bangkok. This project aims to provide an end-to-end system that not only tracks real-time data but also forecasts future pollution levels to support early warnings and planning.

🚀 Features

✅ Automated ETL Pipeline (Prefect)

- Hourly scheduled ingestion of AQI and PM2.5 data from Bangkok monitoring stations via Air4Thai API
- Data cleaning, transformation, and storage in CSV and Parquet formats
- Automated scheduling of ingestion and forecasting flows using Prefect

✅ Data Version Control (LakeFS)

- Git-like version control for datasets with branching, committing, and rollback capabilities
- Safe experimentation on data lakes without impacting production data
- Ensures reproducibility, auditability, and data integrity

✅ Forecasting (ARIMA)

- Time series modeling and hourly forecasting of both PM2.5 and AQI values per station
- Predictions visualized alongside historical data in the dashboard

✅ Visualization Dashboard (Streamlit)

- Near Real-time and forecasted PM2.5 and AQI interactive charts
- Station selection, time range filtering, and geographic AQI heatmaps across Bangkok districts
- Auto-refresh every 60 seconds for up-to-date information

✅ Containerized Environment (Docker)

- Includes Prefect orchestration, LakeFS data versioning, Streamlit UI, and JupyterLab
- Ensures consistent and reproducible deployment across environments

---

📊 Data Schema

The following table describes the structure of the processed dataset used for forecasting and visualization in the Streamlit dashboard. This schema is a refined version of the full dataset stored in LakeFS.

Column	Data Type	Description
`timestamp`	datetime	Timestamp of the measurement
`stationID`	string	Unique station identifier
`nameTH`	string	Station name in Thai
`areaTH`	string	Area name in Thai
`district`	string	District name
`lat`	float	Latitude
`long`	float	Longitude
`AQI.aqi`	int	Air Quality Index (0–500)
`PM25.value`	float	PM2.5 concentration (μg/m³)

🔧 Technologies Used

Tool	Purpose
Prefect	Orchestration and Scheduling
Docker	Containerization
Streamlit	Visualization
LakeFS	Data Versioning
ARIMA	Forecasting Model
Air4Thai PM2.5 API	data source
JupyterLab	Development & experimentation

✅ Data Quality Assurance

To ensure the dataset used in this project is accurate and reliable, we applied a thorough set of quality checks. These checks help maintain data integrity throughout the data ingestion and forecasting workflows, especially when working with real-time air quality data from multiple stations.

- The dataset contains at least **1,000 total records** across all stations
- Each station has at least **24 consecutive hours of data**

- Overall data completeness is greater than **90%** across all fields
- No columns have an `object` **data type**
- There are no **duplicate rows** within any station's data

📓 **For full details on the quality checks:**
▶ Check the complete notebook here: **check_data_quality.ipynb**

🛠 How to Run

1. **Clone the Repository**

```
git clone https://github.com/chawi177/dsi321_2025.git
cd dsi321_2025
```

2. **Start Docker Services**

```
docker-compose up --build -d
```

3. **Access Local Services**

- Prefect UI: http://localhost:4200
- Streamlit Dashboard: http://localhost:8502
- JupyterLab: http://localhost:8888
- LakeFS: http://localhost:8001

🔧 **Default login for LakeFS**:

Username: `access_key`
Password: `secret_key`

> ❗ **Before proceeding**, create a LakeFS repository (one-time setup):

```
lakectl repo create lakefs://dust-concentration
```

4. 📤 Upload Initial Data to LakeFS (Required Before Forecasting & Dashboard)

You'll need to upload initial `.parquet` data into LakeFS so that the dashboard and forecast pipelines can function properly.

First, open a shell inside the Jupyter container:

```
docker exec -it 321repo-jupyter-1 bash
```

Then, run the upload script:

```
python upload.py
```

This script will:

- Locate the most recent folder inside `/home/jovyan/data/data.parquet/year=*/month=*/day=*/`
- Upload the latest day's `.parquet` files to the `dust-concentration` repository in LakeFS
- Overwrite existing files if necessary

5. ☑ Generate Initial Forecast Data (Required for Dashboard)

Before the dashboard can display forecast data, ensure LakeFS contains both real-time and forecasted datasets.

Option A: Run Scripts Manually via CLI

Enter the Jupyter container shell:

```
docker exec -it 321repo-jupyter-1 bash
```

Then run the necessary scripts:

```
python getdata.py
python forecast.py
```

Option B: Trigger Flows from the Prefect UI

If you have already deployed the flows using `deploy.py` and `deploy_ml.py`, you can also trigger them manually from the Prefect UI.

Navigate to http://localhost:4200, select each flow, and click **"Quick Run"** to execute.

6. ⏱ (Optional) Schedule Flows with Prefect

You can automate the ingestion and forecasting flows to run every hour using Prefect.

Deploy the Ingestion Flow (runs at minute 25 every hour)

Enter the Jupyter container shell:

```
docker exec -it 321repo-jupyter-1 bash
```

Then run this script:

```
python deploy.py
```

Deploy the Forecasting Flow (runs at minute 27 every hour)

Enter the Jupyter container shell:

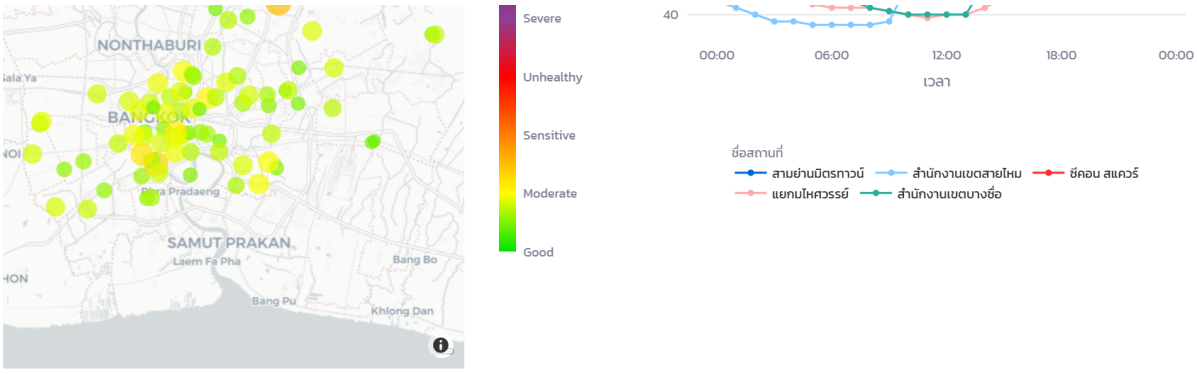
```
docker exec -it 321repo-jupyter-1 bash
```

Then run this script:

```
python deploy_ml.py
```

> ✅ These flows will execute automatically every hour if the Prefect Worker is active and new data is available in LakeFS.

📊 Streamlit Dashboard Overview

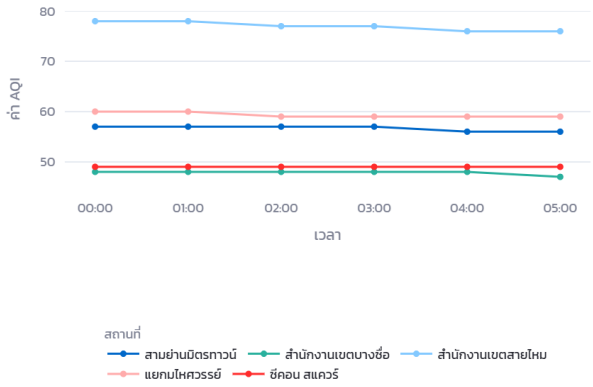


พยากรณ์คุณภาพอากาศล่วงหน้า

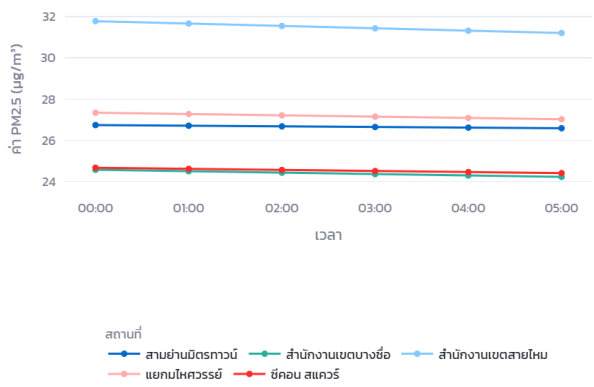
เลือกสถานที่

สำนักงานเขตสายไ... × | แยกมไหศวรรย์ × | สามย่านมิตรทาวน์ × | ซีคอน สแควร์ × | สำนักงานเขตบาง... ×

พยากรณ์ AQI



พยากรณ์ PM2.5



ข้อมูลทั้งหมด (ชั่วโมงล่าสุด)

timestamp	nameTH	district	AQI.aqi	PM25.value
18/05/2025 23:00	มหาวิทยาลัยราชภัฏบ้านสมเด็จเจ้าพระยา	ธนบุรี	29	16.5
18/05/2025 23:00	ริมถนนทางหลวงหมายเลข 3902	บางขุนเทียน	40	20.9
18/05/2025 23:00	การเคหะชุมชนห้วยขวาง	ดินแดง	53	25.5
18/05/2025 23:00	โรงเรียนนนทรีวิทยา	ยานนาวา	36	19.1
18/05/2025 23:00	โรงพยาบาลจุฬาลงกรณ์	ปทุมวัน	43	22.2
18/05/2025 23:00	การไฟฟ้าย่อยธนบุรี	ธนบุรี	43	22.1
18/05/2025 23:00	สถานีตำรวจนครบาลโชคชัย	วังทองหลาง	44	22.5
18/05/2025 23:00	การเคหะชุมชนดินแดง	ดินแดง	45	23.1
18/05/2025 23:00	กรมประชาสัมพันธ์	พญาไท	22	13.4
18/05/2025 23:00	โรงเรียนบดินทรเดชา (สิงห์ สิงหเสนี)	วังทองหลาง	34	18.5
18/05/2025 23:00	สำนักงานเขตบึงกุ่ม	บึงกุ่ม	21	12.4
18/05/2025 23:00	สำนักงานเขตคลองสามวา	คลองสามวา	43	22.3
18/05/2025 23:00	สำนักงานเขตจอมทอง	จอมทอง	32	17.6

The dashboard provides a city-wide overview of real-time and forecasted air quality in Bangkok.

🔗 Components:

- **Station Selector**: Choose a station to view details
- **Real-time Scorecard**: Latest AQI and PM2.5 for selected station
- **Citywide Averages**: Average AQI and PM2.5 across all stations

- **Color Map**: Map of AQI levels with color-coded bubbles
- **Line Chart**: AQI Line Chart for the most polluted station
- **Forecast Line Chart**: Multi-station forecast for AQI and PM2.5
- **Data Table**: All current readings from every station

Forecasting Logic (ARIMA)

We forecast **both AQI and PM2.5** values for each station using manually configured ARIMA models (`order=(1, 0, 1)`), implemented with the `statsmodels` package. Forecasts are generated hourly and stored in LakeFS:

```
lakefs://dust-concentration/main/forecast/forecast.parquet
```

Key Points:

- Forecast horizon: 6 hours into the future per station
- Separate ARIMA(1,0,1) models are trained for both PM2.5 and AQI
- Stations with fewer than 24 hourly records are skipped
- Outlier stations (e.g., with constant data) are excluded
- Forecasts are rounded (AQI) or kept as float (PM2.5) and saved back to LakeFS
- Forecast results are visualized in the Streamlit dashboard

Repository Structure

```
.
├── data/                           # Data directory with Parquet and schema
│   ├── data.parquet/                 # Partitioned Parquet files (LakeFS-style)
│   │   └── year=2025/
│   │       └── month=5/
│   │           └── day=XX/
│   │               └── hour=XX/
│   │                   └── <uuid>.parquet
│   ├── SCHEMA.md                     # Dataset schema documentation
│   └── check_data_quality.ipynb      # Notebook for validating data quality
│
├── pipeline/                       # Python scripts for ingestion & forecasting
│   ├── bangkok_districts.geojson     # GeoJSON file for Bangkok map visualization
│   ├── deploy.py                     # Prefect flow: fetch real-time data from
API
│   ├── deploy_ml.py                  # Prefect flow: run ARIMA forecasts per
station
│   ├── forecast.py                   # ARIMA model for forecasting
│   ├── getdata.py                    # Script to retrieve and transform API data
│   └── savedata.py                   # Dowload entire LakeFS repository contents
to local
│   └── upload.py                     # Upload latest day's `.parquet` files to
LakeFS
│
```

```
├── prefect/                          # Prefect-related configs and Docker setup
│   ├── Dockerfile.jupyter              # Dockerfile for JupyterLab environment
│   ├── Dockerfile.prefect-worker       # Dockerfile for Prefect flow worker
│   ├── requirements.txt                # Python dependencies for flows
│   └── wait-for-server.sh              # Helper script to wait for services
│
├── visualization/                    # Streamlit dashboard implementation
│   ├── .streamlit/
│   │   └── config.toml                   # Streamlit UI configuration (theme,
layout)
│   └── app.py                          # Main Streamlit dashboard application
│   ├── img/                              # Images for README or dashboard preview
│       └── dashboard_demo.png/             # Screenshot of the Streamlit dashboard
│
├── .gitignore                        # Git ignored files list
├── README.md                         # Main project documentation
└── docker-compose.yml                # Docker Compose to run all services
```

📩 Contact

For questions, feedback, or collaboration inquiries, feel free to reach out:

- ✉️ Email: chawisa.wann@gmail.com

DSI321: BIG DATA INFRASTRUCTURE | 2025