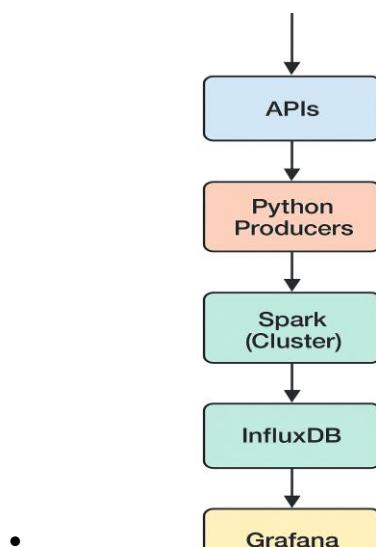


Project Title & Team

- **Project Name:** EnviroHealth-Monitor
- **Checkpoint:** Milestone 2 - Implementation & Demonstration
- **Team Members:**
 - Hussnain Amanat Ali
 - Ayman Berri
 - Muhammad Irfan

2. System Architecture

- **Description:** Briefly explain the pipeline you built.
 - *We implemented a real-time Lambda Architecture pipeline. Data is ingested from OpenAQ and Open-Meteo APIs via Kafka, processed by Apache Spark Structured Streaming on a 6-node cluster, and stored in both HDFS (Data Lake) and InfluxDB (Real-time Database) for visualization in Grafana.*
- **Diagram:**
- **API -> Python Producers -> Kafka -> Spark (Cluster) -> InfluxDB -> Grafana**



3. Infrastructure (The 6-Node Cluster)

- **Description:** We successfully configured a 6-node distributed Hadoop cluster on Azure, consisting of 1 Master node and 5 Worker nodes.

- Evidence: [Screenshot of hdfs dfsadmin -report]

- HDFS Admin Report showing 6 Live Datanodes connected to the Master.

```
# 3. Ensure InfluxDB & Grafana are running
sudo systemctl start influxdb
sudo systemctl start grafana-server
Starting namenodes on [master]
master: namenode is running as process 1506. Stop it first and ensure /tmp/hadoop-adm-mcsc-namenode.pid file is empty before retry.
Starting datanodes
localhost: datanode is running as process 1658. Stop it first and ensure /tmp/hadoop-adm-mcsc-datanode.pid file is empty before retr
y.
worker4: datanode is running as process 1634. Stop it first and ensure /tmp/hadoop-adm-mcsc-datanode.pid file is empty before retr
y.
worker5: datanode is running as process 1601. Stop it first and ensure /tmp/hadoop-adm-mcsc-datanode.pid file is empty before retr
y.
worker1: datanode is running as process 1318. Stop it first and ensure /tmp/hadoop-adm-mcsc-datanode.pid file is empty before retr
y.
worker3: datanode is running as process 1345. Stop it first and ensure /tmp/hadoop-adm-mcsc-datanode.pid file is empty before retr
y.
worker2: datanode is running as process 1349. Stop it first and ensure /tmp/hadoop-adm-mcsc-datanode.pid file is empty before retr
y.
Starting secondary namenodes [master]
master: secondarynamenode is running as process 1934. Stop it first and ensure /tmp/hadoop-adm-mcsc-secondarynamenode.pid file is em
pty before retr
y.
Starting resourcemanager
resourcemanager is running as process 2135. Stop it first and ensure /tmp/hadoop-adm-mcsc-resourcemanager.pid file is empty before r
etry.
Starting nodemanagers
localhost: nodemanager is running as process 2350. Stop it first and ensure /tmp/hadoop-adm-mcsc-nodemanager.pid file is empty befor
e retr
y.
worker1: nodemanager is running as process 1453. Stop it first and ensure /tmp/hadoop-adm-mcsc-nodemanager.pid file is empty before retr
y.
worker3: nodemanager is running as process 1480. Stop it first and ensure /tmp/hadoop-adm-mcsc-nodemanager.pid file is empty before retr
y.
worker5: nodemanager is running as process 1705. Stop it first and ensure /tmp/hadoop-adm-mcsc-nodemanager.pid file is empty before retr
y.
worker2: nodemanager is running as process 1481. Stop it first and ensure /tmp/hadoop-adm-mcsc-nodemanager.pid file is empty before retr
y.
worker4: nodemanager is running as process 1767. Stop it first and ensure /tmp/hadoop-adm-mcsc-nodemanager.pid file is empty before retr
y.
adm-mcsc@master:~$ |
```

4. Data Ingestion & Processing

Description: Two Python producers fetch live Air Quality and Weather data and publish to Kafka. A Spark Structured Streaming job consumes these topics, joins the streams based on time windows, and calculates the Health Risk Index (HRI).

```
✓ Sent Weather (Synced): {'city': 'Paris', 'temperature': 4.8, 'windspeed': 9.1, 'timestamp': '2025-11-27T08:55:56.548227'}
✓ Sent Weather (Synced): {'city': 'Paris', 'temperature': 4.8, 'windspeed': 9.1, 'timestamp': '2025-11-27T08:56:01.633756'}
✓ Sent Weather (Synced): {'city': 'Paris', 'temperature': 4.8, 'windspeed': 9.1, 'timestamp': '2025-11-27T08:56:06.726317'}
✓ Sent Weather (Synced): {'city': 'Paris', 'temperature': 4.8, 'windspeed': 9.1, 'timestamp': '2025-11-27T08:56:11.818981'}
✓ Sent Weather (Synced): {'city': 'Paris', 'temperature': 4.8, 'windspeed': 9.1, 'timestamp': '2025-11-27T08:56:16.911132'}
✓ Sent Weather (Synced): {'city': 'Paris', 'temperature': 4.8, 'windspeed': 9.1, 'timestamp': '2025-11-27T08:56:22.010872'}
✓ Sent Weather (Synced): {'city': 'Paris', 'temperature': 4.8, 'windspeed': 9.1, 'timestamp': '2025-11-27T08:56:27.141072'}
✓ Sent Weather (Synced): {'city': 'Paris', 'temperature': 4.8, 'windspeed': 9.1, 'timestamp': '2025-11-27T08:56:32.293777'}
✓ Sent Weather (Synced): {'city': 'Paris', 'temperature': 4.8, 'windspeed': 9.1, 'timestamp': '2025-11-27T08:56:37.456678'}
✓ Sent Weather (Synced): {'city': 'Paris', 'temperature': 4.8, 'windspeed': 9.1, 'timestamp': '2025-11-27T08:56:42.545867'}
✓ Sent Weather (Synced): {'city': 'Paris', 'temperature': 4.8, 'windspeed': 9.1, 'timestamp': '2025-11-27T08:56:47.629409'}
✓ Sent Weather (Synced): {'city': 'Paris', 'temperature': 4.8, 'windspeed': 9.1, 'timestamp': '2025-11-27T08:56:52.716753'}
✓ Sent Weather (Synced): {'city': 'Paris', 'temperature': 4.8, 'windspeed': 9.1, 'timestamp': '2025-11-27T08:56:57.806116'}
✓ Sent Weather (Synced): {'city': 'Paris', 'temperature': 4.8, 'windspeed': 9.1, 'timestamp': '2025-11-27T08:57:02.891838'}
✓ Sent Weather (Synced): {'city': 'Paris', 'temperature': 4.8, 'windspeed': 9.1, 'timestamp': '2025-11-27T08:57:07.978814'}
✓ Sent Weather (Synced): {'city': 'Paris', 'temperature': 4.8, 'windspeed': 9.1, 'timestamp': '2025-11-27T08:57:13.069986'}
✓ Sent Weather (Synced): {'city': 'Paris', 'temperature': 4.8, 'windspeed': 9.1, 'timestamp': '2025-11-27T08:57:18.155981'}
✓ Sent Weather (Synced): {'city': 'Paris', 'temperature': 4.8, 'windspeed': 9.1, 'timestamp': '2025-11-27T08:57:23.246077'}
✓ Sent Weather (Synced): {'city': 'Paris', 'temperature': 4.8, 'windspeed': 9.1, 'timestamp': '2025-11-27T08:57:28.337457'}
✓ Sent Weather (Synced): {'city': 'Paris', 'temperature': 4.8, 'windspeed': 9.1, 'timestamp': '2025-11-27T08:57:33.424739'}
✓ Sent Weather (Synced): {'city': 'Paris', 'temperature': 4.8, 'windspeed': 9.1, 'timestamp': '2025-11-27T08:57:38.509791'}
✓ Sent Weather (Synced): {'city': 'Paris', 'temperature': 4.8, 'windspeed': 9.1, 'timestamp': '2025-11-27T08:57:43.601252'}
✓ Sent Weather (Synced): {'city': 'Paris', 'temperature': 4.8, 'windspeed': 9.1, 'timestamp': '2025-11-27T08:57:48.696253'}
✓ Sent Weather (Synced): {'city': 'Paris', 'temperature': 4.8, 'windspeed': 9.1, 'timestamp': '2025-11-27T08:57:53.787500'}
✓ Sent Weather (Synced): {'city': 'Paris', 'temperature': 4.8, 'windspeed': 9.1, 'timestamp': '2025-11-27T08:57:58.877007'}
✓ Sent Weather (Synced): {'city': 'Paris', 'temperature': 4.8, 'windspeed': 9.1, 'timestamp': '2025-11-27T08:58:03.968038'}
✓ Sent Weather (Synced): {'city': 'Paris', 'temperature': 4.8, 'windspeed': 9.1, 'timestamp': '2025-11-27T08:58:09.058822'}
✓ Sent Weather (Synced): {'city': 'Paris', 'temperature': 4.8, 'windspeed': 9.1, 'timestamp': '2025-11-27T08:58:14.145700'}
✓ Sent Weather (Synced): {'city': 'Paris', 'temperature': 4.8, 'windspeed': 9.1, 'timestamp': '2025-11-27T08:58:19.244658'}
✓ Sent Weather (Synced): {'city': 'Paris', 'temperature': 4.8, 'windspeed': 9.1, 'timestamp': '2025-11-27T08:58:24.334405'}
✓ Sent Weather (Synced): {'city': 'Paris', 'temperature': 4.8, 'windspeed': 9.1, 'timestamp': '2025-11-27T08:58:29.427484'}
✓ Sent Weather (Synced): {'city': 'Paris', 'temperature': 4.8, 'windspeed': 9.1, 'timestamp': '2025-11-27T08:58:34.521580'}
✓ Sent Weather (Synced): {'city': 'Paris', 'temperature': 4.8, 'windspeed': 9.1, 'timestamp': '2025-11-27T08:58:39.605488'}
```

```

    Sent Simulated Data: {"city": "Paris", "country": "FR", "parameter": "pm25", "value": 13.23, "unit": "µg/m³", "timestamp": "2025-1-27T08:57:39.492825"}
    Sent Simulated Data: {"city": "Paris", "country": "FR", "parameter": "pm25", "value": 25.67, "unit": "µg/m³", "timestamp": "2025-1-27T08:57:44.493892"}
    Sent Simulated Data: {"city": "Paris", "country": "FR", "parameter": "pm25", "value": 52.44, "unit": "µg/m³", "timestamp": "2025-1-27T08:57:49.495526"}
    Sent Simulated Data: {"city": "Paris", "country": "FR", "parameter": "pm25", "value": 7.66, "unit": "µg/m³", "timestamp": "2025-11-27T08:57:54.496831"}
    Sent Simulated Data: {"city": "Paris", "country": "FR", "parameter": "pm25", "value": 41.36, "unit": "µg/m³", "timestamp": "2025-1-27T08:59.497261"}
    Sent Simulated Data: {"city": "Paris", "country": "FR", "parameter": "pm25", "value": 47.54, "unit": "µg/m³", "timestamp": "2025-1-27T08:58:04.497742"}
    Sent Simulated Data: {"city": "Paris", "country": "FR", "parameter": "pm25", "value": 6.26, "unit": "µg/m³", "timestamp": "2025-11-27T08:58:09.498788"}
    Sent Simulated Data: {"city": "Paris", "country": "FR", "parameter": "pm25", "value": 26.74, "unit": "µg/m³", "timestamp": "2025-1-27T08:58:14.500880"}
    Sent Simulated Data: {"city": "Paris", "country": "FR", "parameter": "pm25", "value": 16.99, "unit": "µg/m³", "timestamp": "2025-1-27T08:58:18.501371"}
    Sent Simulated Data: {"city": "Paris", "country": "FR", "parameter": "pm25", "value": 42.27, "unit": "µg/m³", "timestamp": "2025-1-27T08:58:24.502442"}
    Sent Simulated Data: {"city": "Paris", "country": "FR", "parameter": "pm25", "value": 47.21, "unit": "µg/m³", "timestamp": "2025-1-27T08:58:29.503654"}
    Sent Simulated Data: {"city": "Paris", "country": "FR", "parameter": "pm25", "value": 9.01, "unit": "µg/m³", "timestamp": "2025-11-27T08:58:34.504689"}
    Sent Simulated Data: {"city": "Paris", "country": "FR", "parameter": "pm25", "value": 20.42, "unit": "µg/m³", "timestamp": "2025-1-27T08:58:39.505212"}
    Sent Simulated Data: {"city": "Paris", "country": "FR", "parameter": "pm25", "value": 53.63, "unit": "µg/m³", "timestamp": "2025-1-27T08:58:44.505709"}
    Sent Simulated Data: {"city": "Paris", "country": "FR", "parameter": "pm25", "value": 48.52, "unit": "µg/m³", "timestamp": "2025-1-27T08:58:49.507463"}
    Sent Simulated Data: {"city": "Paris", "country": "FR", "parameter": "pm25", "value": 34.46, "unit": "µg/m³", "timestamp": "2025-1-27T08:58:54.507916"}
    Sent Simulated Data: {"city": "Paris", "country": "FR", "parameter": "pm25", "value": 12.94, "unit": "µg/m³", "timestamp": "2025-1"

```

- Evidence: [Screenshot of Spark Terminal]**

Spark Structured Streaming logs confirming successful processing of micro-batches and writing to InfluxDB.

```

    Batch 26: Wrote 11 records to InfluxDB.
    Batch 28: Wrote 12 records to InfluxDB.
    Batch 30: Wrote 12 records to InfluxDB.
    Batch 32: Wrote 12 records to InfluxDB.
    Batch 34: Wrote 12 records to InfluxDB.
    Batch 36: Wrote 12 records to InfluxDB.
    Batch 38: Wrote 12 records to InfluxDB.
    Batch 40: Wrote 12 records to InfluxDB.
    Batch 41: Wrote 12 records to InfluxDB.
    Batch 43: Wrote 12 records to InfluxDB.
    Batch 44: Wrote 12 records to InfluxDB.
    Batch 46: Wrote 12 records to InfluxDB.
    Batch 47: Wrote 12 records to InfluxDB.
    Batch 49: Wrote 12 records to InfluxDB.
    Batch 50: Wrote 12 records to InfluxDB.
    Batch 52: Wrote 12 records to InfluxDB.
    Batch 53: Wrote 12 records to InfluxDB.
    Batch 55: Wrote 12 records to InfluxDB.
    Batch 56: Wrote 12 records to InfluxDB.
    Batch 58: Wrote 12 records to InfluxDB.
    Batch 59: Wrote 12 records to InfluxDB.
    Batch 61: Wrote 34 records to InfluxDB.
    Batch 62: Wrote 93 records to InfluxDB.
    Batch 63: Wrote 69 records to InfluxDB.
    Batch 64: Wrote 39 records to InfluxDB.
    Batch 65: Wrote 49 records to InfluxDB.
    Batch 66: Wrote 72 records to InfluxDB.
    Batch 67: Wrote 68 records to InfluxDB.
    Batch 68: Wrote 65 records to InfluxDB.
    Batch 69: Wrote 25 records to InfluxDB.
    Batch 70: Wrote 68 records to InfluxDB.
    Batch 71: Wrote 37 records to InfluxDB.
    Batch 72: Wrote 48 records to InfluxDB.
    Batch 73: Wrote 48 records to InfluxDB.
    Batch 74: Wrote 83 records to InfluxDB.

```

5. Data Lake Storage (The "High Quality" Feature)

- Description:** *To ensure data reproducibility, raw events are partitioned and stored in HDFS as a Data Lake.*
- Evidence: [Screenshot of hdfs dfs -ls -R /datalake]**
- HDFS Data Lake structure showing raw JSON events partitioned by year, month, day, and hour.

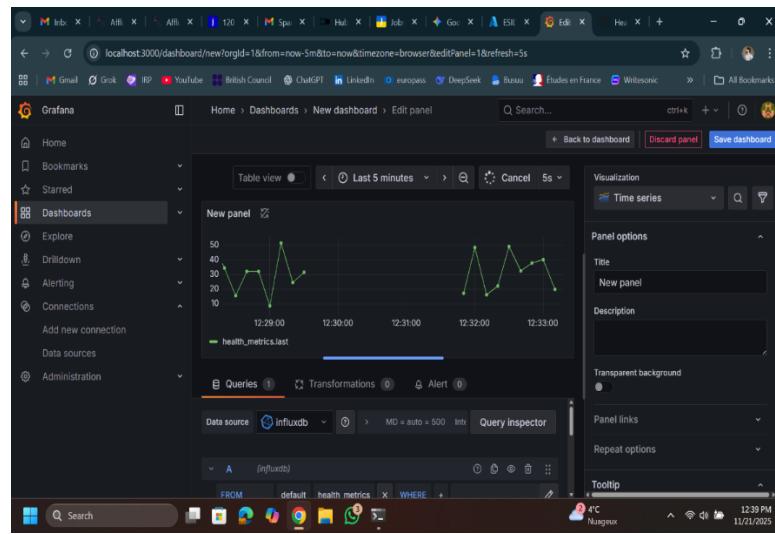
```

Windows PowerShell X Windows PowerShell X Windows PowerShell X Windows PowerShell X Windows PowerShell X
Che
-rw-r--r-- 3 adm-mscsupergroup 256 2025-11-21 14:46 /datalake/weather/_spark_metadata/1
-rw-r--r-- 3 adm-mscsupergroup 256 2025-11-21 14:46 /datalake/weather/_spark_metadata/10
-rw-r--r-- 3 adm-mscsupergroup 256 2025-11-21 15:12 /datalake/weather/_spark_metadata/11
-rw-r--r-- 3 adm-mscsupergroup 256 2025-11-21 14:46 /datalake/weather/_spark_metadata/2
-rw-r--r-- 3 adm-mscsupergroup 256 2025-11-21 14:46 /datalake/weather/_spark_metadata/3
-rw-r--r-- 3 adm-mscsupergroup 256 2025-11-21 14:46 /datalake/weather/_spark_metadata/4
-rw-r--r-- 3 adm-mscsupergroup 256 2025-11-21 14:46 /datalake/weather/_spark_metadata/5
-rw-r--r-- 3 adm-mscsupergroup 256 2025-11-21 14:46 /datalake/weather/_spark_metadata/6
-rw-r--r-- 3 adm-mscsupergroup 256 2025-11-21 14:46 /datalake/weather/_spark_metadata/7
-rw-r--r-- 3 adm-mscsupergroup 256 2025-11-21 14:47 /datalake/weather/_spark_metadata/8
-rw-r--r-- 3 adm-mscsupergroup 2288 2025-11-21 14:47 /datalake/weather/_spark_metadata/9.compact
drwxr-xr-x - adm-mscsupergroup 0 2025-11-21 14:46 /datalake/weather/year=2025
drwxr-xr-x - adm-mscsupergroup 0 2025-11-21 14:46 /datalake/weather/year=2025/month=11
drwxr-xr-x - adm-mscsupergroup 0 2025-11-21 15:18 /datalake/weather/year=2025/month=11/day=21
drwxr-xr-x - adm-mscsupergroup 0 2025-11-21 14:47 /datalake/weather/year=2025/month=11/day=21/hour=14
-rw-r--r-- 3 adm-mscsupergroup 91 2025-11-21 14:47 /datalake/weather/year=2025/month=11/day=21/hour=14/part-
G 00000-0e8f77e7-1693-4u83-8269-4d311aa85b4.c000.json
Dist 00000-16c5fcfc3-29f6-42ca-85ee-e27aaaf4c3fe0.c000.json
-rw-r--r-- 3 adm-mscsupergroup 91 2025-11-21 14:46 /datalake/weather/year=2025/month=11/day=21/hour=14/part-
00000-592d4ffe-4048-4f66-ab2f-a5bfa463abe4.c000.json
-rw-r--r-- 3 adm-mscsupergroup 91 2025-11-21 14:46 /datalake/weather/year=2025/month=11/day=21/hour=14/part-
A 00000-7276f452-2a8d-4427-98c7-e9f96aa8c3573.c000.json
-rw-r--r-- 3 adm-mscsupergroup 91 2025-11-21 14:46 /datalake/weather/year=2025/month=11/day=21/hour=14/part-
00000-8ace99dc-f028-49ad-a3b3-91a83ad68825.c000.json
-rw-r--r-- 3 adm-mscsupergroup 91 2025-11-21 14:46 /datalake/weather/year=2025/month=11/day=21/hour=14/part-
00000-cbeca489-394c-45ea-9c86-0dadbe30188.c000.json
-rw-r--r-- 3 adm-mscsupergroup 91 2025-11-21 14:47 /datalake/weather/year=2025/month=11/day=21/hour=14/part-
00000-c9788ed5-cdb7-4957-974e-86e62496e510.c000.json
202 -rw-r--r-- 3 adm-mscsupergroup 91 2025-11-21 14:46 /datalake/weather/year=2025/month=11/day=21/hour=14/part-

```

6. Visualization (The Dashboard)

- Description:** Real-time health risks are visualized on a Grafana dashboard connected to InfluxDB.
- Evidence:** [Screenshot of Grafana]
 - Real-time Grafana Dashboard displaying the calculated Health Risk Index (Gauge) and live Air Quality trends.



7. Conclusion

- We have successfully implemented an end-to-end Big Data pipeline that ingests, processes, stores, and visualizes environmental health data in real-time across a distributed cloud cluster.