

Air Quality Monitoring System

Using Apache Kafka

By:

Vasumathi R

RA2412044015107

AIDS-B Section

SRMIST, Chennai.

AIM:

To design and implement a real-time air quality monitoring system that continuously collects, processes, and streams air quality data using Apache Kafka to enable actionable insights for environmental monitoring and public health awareness.

INTRODUCTION:

The Air Quality Monitoring System leverages Apache Kafka to collect, process, and stream real-time pollution data. This scalable system provides actionable insights through dashboards and alert mechanisms, empowering timely interventions. By addressing health and sustainability challenges, it highlights technology's role in advancing environmental monitoring and public awareness.

PROCEDURE:

Step 1: Install Apache Kafka in WSL

Step 2: Create Kafka Topic

Step 3: Install Required Libraries

Step 4: Get Open Air Quality Map API Key

Step 5: Create the Kafka Producer

Step 6: Create the Kafka Consumer

Step 7: Create a Real-Time Dashboard Using Streamlit

EXECUTION and SCREENSHOTS

Step 1: Install Apache Kafka in WSL

If you haven't already installed Kafka, follow these steps:

1.1 Download and Extract Kafka

```
wget https://archive.apache.org/dist/kafka/3.4.1/kafka_2.13-3.4.1.tgz
```

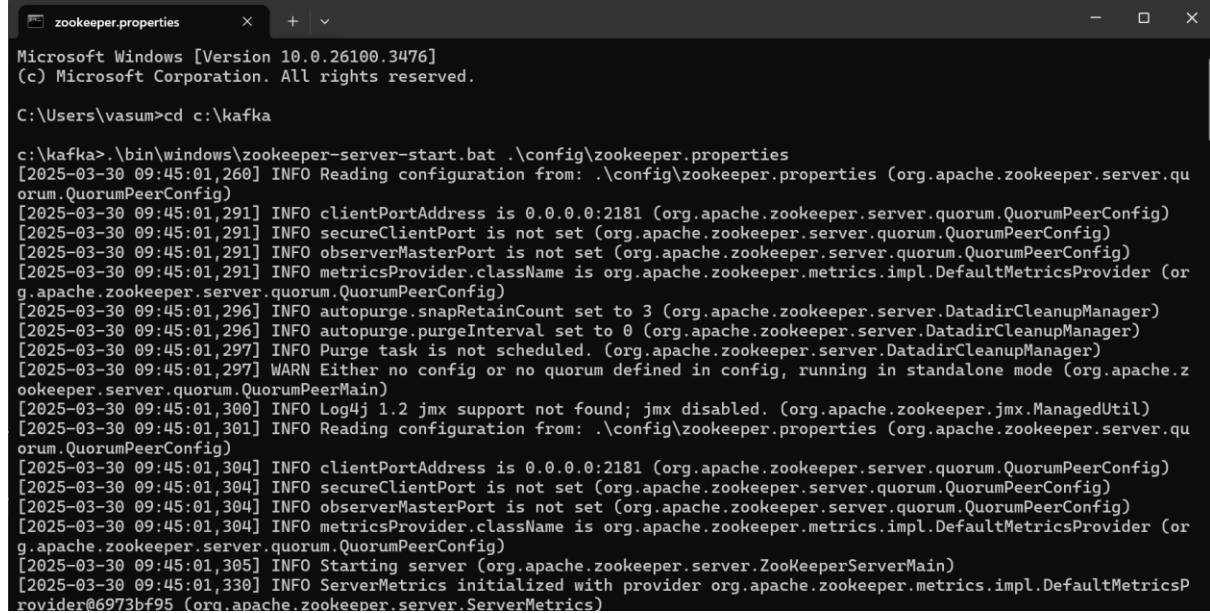
```
tar -xvzf kafka_2.13-3.4.1.tgz (Extract)
```

```
mv kafka_2.13-3.4.1 kafka (Store)
```

1.2 Start Zookeeper

```
cd kafka
```

```
bin/zookeeper-server-start.sh config/zookeeper.properties
```



```
Microsoft Windows [Version 10.0.26100.3476]
(c) Microsoft Corporation. All rights reserved.

C:\Users\vasum>cd c:\kafka

c:\kafka>.\bin\windows\zookeeper-server-start.bat .\config\zookeeper.properties
[2025-03-30 09:45:01,260] INFO Reading configuration from: .\config\zookeeper.properties (org.apache.zookeeper.server.quorum.QuorumPeerConfig)
[2025-03-30 09:45:01,291] INFO clientPortAddress is 0.0.0.0:2181 (org.apache.zookeeper.server.quorum.QuorumPeerConfig)
[2025-03-30 09:45:01,291] INFO secureClientPort is not set (org.apache.zookeeper.server.quorum.QuorumPeerConfig)
[2025-03-30 09:45:01,291] INFO observerMasterPort is not set (org.apache.zookeeper.server.quorum.QuorumPeerConfig)
[2025-03-30 09:45:01,291] INFO metricsProvider.className is org.apache.zookeeper.metrics.impl.DefaultMetricsProvider (org.apache.zookeeper.server.quorum.QuorumPeerConfig)
[2025-03-30 09:45:01,296] INFO autopurge.snapRetainCount set to 3 (org.apache.zookeeper.server.DatadirCleanupManager)
[2025-03-30 09:45:01,296] INFO autopurge.purgeInterval set to 0 (org.apache.zookeeper.server.DatadirCleanupManager)
[2025-03-30 09:45:01,297] INFO Purge task is not scheduled. (org.apache.zookeeper.server.DatadirCleanupManager)
[2025-03-30 09:45:01,297] WARN Either no config or no quorum defined in config, running in standalone mode (org.apache.zookeeper.server.quorum.QuorumPeerMain)
[2025-03-30 09:45:01,300] INFO Log4j 1.2 jmx support not found; jmx disabled. (org.apache.zookeeper.jmx.ManagedUtil)
[2025-03-30 09:45:01,301] INFO Reading configuration from: .\config\zookeeper.properties (org.apache.zookeeper.server.quorum.QuorumPeerConfig)
[2025-03-30 09:45:01,304] INFO clientPortAddress is 0.0.0.0:2181 (org.apache.zookeeper.server.quorum.QuorumPeerConfig)
[2025-03-30 09:45:01,304] INFO secureClientPort is not set (org.apache.zookeeper.server.quorum.QuorumPeerConfig)
[2025-03-30 09:45:01,304] INFO observerMasterPort is not set (org.apache.zookeeper.server.quorum.QuorumPeerConfig)
[2025-03-30 09:45:01,304] INFO metricsProvider.className is org.apache.zookeeper.metrics.impl.DefaultMetricsProvider (org.apache.zookeeper.server.quorum.QuorumPeerConfig)
[2025-03-30 09:45:01,305] INFO Starting server (org.apache.zookeeper.server.ZooKeeperServerMain)
[2025-03-30 09:45:01,330] INFO ServerMetrics initialized with provider org.apache.zookeeper.metrics.impl.DefaultMetricsProvider@6973bf95 (org.apache.zookeeper.server.ServerMetrics)
```

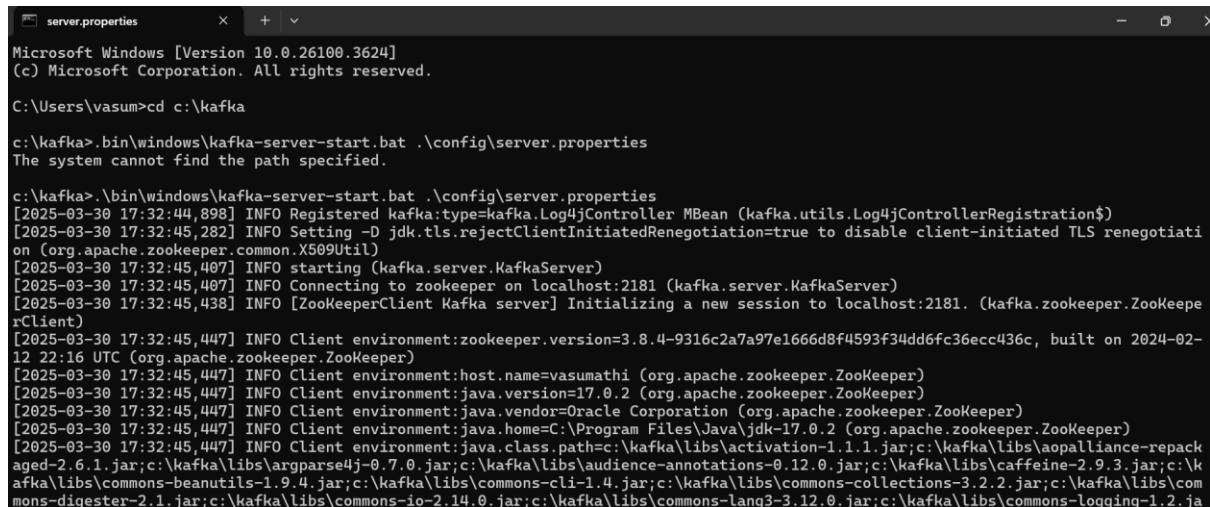
1.3 Start Kafka Server

```
In a new terminal:
```

```
cd kafka
```

```
bin/kafka-server-start.sh config/server.properties
```

```
Server Running
```



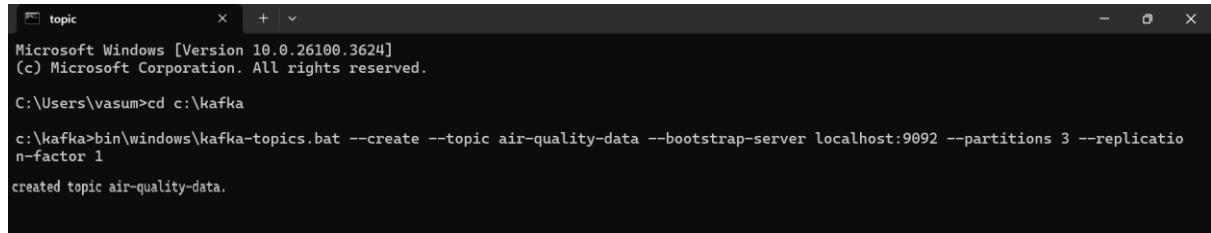
```
Microsoft Windows [Version 10.0.26100.3624]
(c) Microsoft Corporation. All rights reserved.

C:\Users\vasum>cd c:\kafka

c:\kafka>.\bin\windows\kafka-server-start.bat .\config\server.properties
The system cannot find the path specified.

c:\kafka>.\bin\windows\kafka-server-start.bat .\config\server.properties
[2025-03-30 17:32:44,898] INFO Registered kafka:type=kafka.Log4jController MBean (kafka.utils.Log4jControllerRegistration$)
[2025-03-30 17:32:45,282] INFO Setting -D jdk.tls.rejectClientInitiatedRenegotiation=true to disable client-initiated TLS renegotiation (org.apache.zookeeper.common.X509Util)
[2025-03-30 17:32:45,407] INFO starting (kafka.server.KafkaServer)
[2025-03-30 17:32:45,407] INFO Connecting to zookeeper on localhost:2181 (kafka.server.KafkaServer)
[2025-03-30 17:32:45,438] INFO [ZooKeeperClient Kafka server] Initializing a new session to localhost:2181. (kafka.zookeeper.ZooKeeperClient)
[2025-03-30 17:32:45,447] INFO Client environment:zookeeper.version=3.8.4-9316c2a7a97e1666d8f4593f34dd6fc36ecc436c, built on 2024-02-12 22:16 UTC (org.apache.zookeeper.ZooKeeper)
[2025-03-30 17:32:45,447] INFO Client environment:host.name=vasumathi (org.apache.zookeeper.ZooKeeper)
[2025-03-30 17:32:45,447] INFO Client environment:java.version=17.0.2 (org.apache.zookeeper.ZooKeeper)
[2025-03-30 17:32:45,447] INFO Client environment:java.vendor=Oracle Corporation (org.apache.zookeeper.ZooKeeper)
[2025-03-30 17:32:45,447] INFO Client environment:java.home=C:\Program Files\Java\jdk-17.0.2 (org.apache.zookeeper.ZooKeeper)
[2025-03-30 17:32:45,447] INFO Client environment:java.class.path=c:\kafka\libs\activation-1.1.1.jar;c:\kafka\libs\apool-alliance-repacked-2.6.1.jar;c:\kafka\libs\argparse4j-0.7.0.jar;c:\kafka\libs\audience-annotations-0.12.0.jar;c:\kafka\libs\caffeine-2.9.3.jar;c:\kafka\libs\commons-beanutils-1.9.4.jar;c:\kafka\libs\commons-cli-1.4.jar;c:\kafka\libs\commons-collections-3.2.2.jar;c:\kafka\libs\commons-digester-2.1.jar;c:\kafka\libs\commons-io-2.14.0.jar;c:\kafka\libs\commons-lang3-3.12.0.jar;c:\kafka\libs\commons-logging-1.2.jar
```

Step 2: Create Kafka Topic

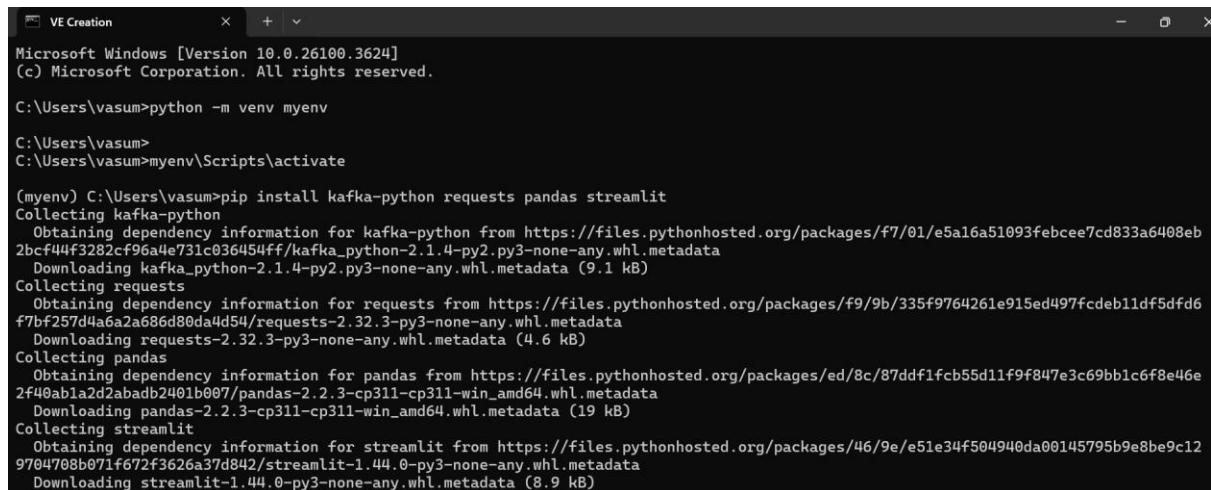


```
topic
Microsoft Windows [Version 10.0.26100.3624]
(c) Microsoft Corporation. All rights reserved.

C:\Users\vasum>cd c:\kafka
c:\kafka>bin\windows\kafka-topics.bat --create --topic air-quality-data --bootstrap-server localhost:9092 --partitions 3 --replication-factor 1
created topic air-quality-data.
```

Step 3: Install Required Libraries

Virtual Environment Creation



```
VE Creation
Microsoft Windows [Version 10.0.26100.3624]
(c) Microsoft Corporation. All rights reserved.

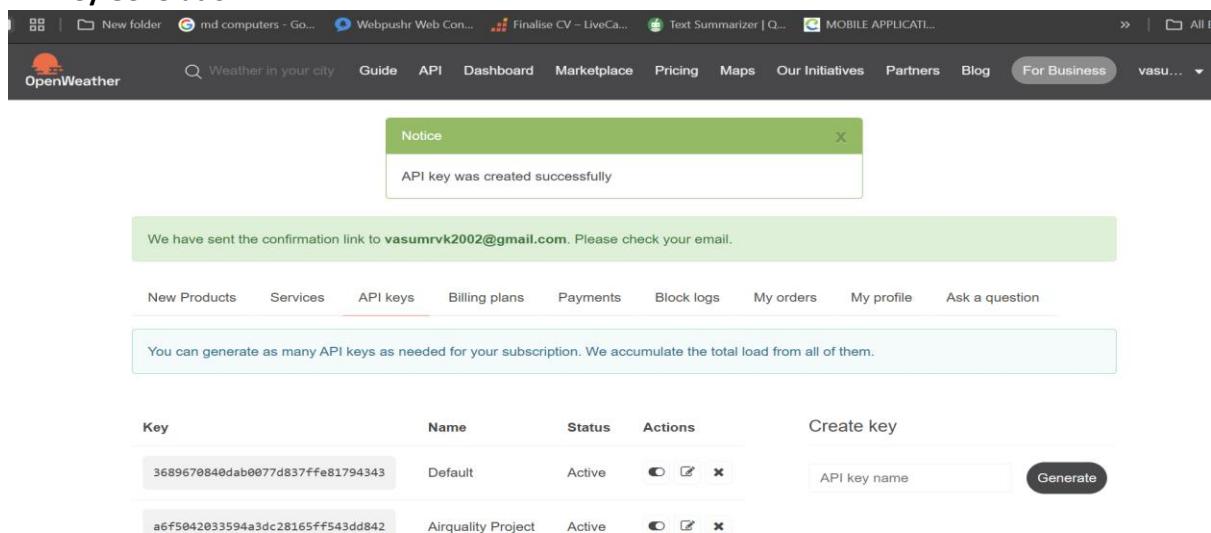
C:\Users\vasum>python -m venv myenv
C:\Users\vasum>
C:\Users\vasum>myenv\Scripts\activate

(myenv) C:\Users\vasum>pip install kafka-python requests pandas streamlit
Collecting kafka-python
  Obtaining dependency information for kafka-python from https://files.pythonhosted.org/packages/f7/01/e5a16a51093febcee7cd833a6408eb2bcf44f3282cf96a1e731c036454ff/kafka_python-2.1.4-py2.py3-none-any.whl.metadata
    Downloading kafka_python-2.1.4-py2.py3-none-any.whl.metadata (9.1 kB)
Collecting requests
  Obtaining dependency information for requests from https://files.pythonhosted.org/packages/f9/9b/335f9764261e915ed497fcdeb11df5dfd6f7bf257d4a6a2a686d80da4d54/requests-2.32.3-py3-none-any.whl.metadata
    Downloading requests-2.32.3-py3-none-any.whl.metadata (4.6 kB)
Collecting pandas
  Obtaining dependency information for pandas from https://files.pythonhosted.org/packages/ed/8c/87ddf1fcbb55d11f9f847e3c69bb1c6f8e46e2f40ab1a2d2abdb2401b007/pandas-2.2.3-cp311-cp311-win_amd64.whl.metadata
    Downloading pandas-2.2.3-cp311-cp311-win_amd64.whl.metadata (19 kB)
Collecting streamlit
  Obtaining dependency information for streamlit from https://files.pythonhosted.org/packages/46/9e/e51e34f504940da00145795b9e8be9c1297047688b671f672f3626a37d842/streamlit-1.44.0-py3-none-any.whl.metadata
    Downloading streamlit-1.44.0-py3-none-any.whl.metadata (8.9 kB)
```

Step 4: Get Open Air Quality Map API Key

1. Go to https://home.openweathermap.org/users/sign_up
2. Sign up for a free account
3. Go to API keys → Generate a new API key

API Key Generation:



The screenshot shows the OpenWeather API key generation interface. A green notice box at the top right says "API key was created successfully". Below it, a green bar at the bottom left says "We have sent the confirmation link to vasumrvk2002@gmail.com. Please check your email." The main content area has a table with columns: Key, Name, Status, Actions, and Create key. There are two rows: one for a Default key (Status: Active) and another for an Airquality Project key (Status: Active). Both keys have checkboxes for edit and delete, and a "Generate" button next to the "Create key" column.

Key	Name	Status	Actions	Create key
3689670840dab0077d837ffe81794343	Default	Active	<input type="checkbox"/> <input checked="" type="checkbox"/> <input type="checkbox"/>	<input type="text" value="API key name"/> <input type="button" value="Generate"/>
a6f5042033594a3dc28165ff543dd842	Airquality Project	Active	<input type="checkbox"/> <input checked="" type="checkbox"/> <input type="checkbox"/>	

Step 5: Create the Kafka Producer

Kafka Producer: (Producer.py)

```
import json
import requests
from kafka import KafkaProducer
import time

# OpenWeatherMap API details
API_KEY = 'a6f5042033594a3dc28165ff543dd842'
CITY = 'Chennai' # City name
LAT, LON = 13.0827, 80.2707 # Latitude and Longitude of Chennai

# API URL to fetch air quality data
URL =
f"http://api.openweathermap.org/data/2.5/air_pollution?lat={LAT}&lon={LON}&appid={API_KEY}"

# Kafka Producer configuration
producer = KafkaProducer(
    bootstrap_servers='localhost:9092', # Kafka server
    value_serializer=lambda v: json.dumps(v).encode('utf-8') # Convert data to JSON
)

def fetch_air_quality_data():
    """Fetch real-time air quality data from OpenWeatherMap API"""
    try:
        response = requests.get(URL)
        data = response.json()
        return data
    except Exception as e:
        print(f"Error fetching data: {e}")
        return None

# Send data to Kafka at regular intervals
while True:
    air_data = fetch_air_quality_data()
```

```

if air_data:

    producer.send('air-quality-data', value=air_data)

    print(" ✅ Data sent to Kafka:", air_data)

time.sleep(10) # Fetch data every 10 seconds

```

```

producer
+ ~
0': 20.6, 'nh3': 5.38}, 'dt': 1742909693}]}
✅ Data sent to Kafka: {'coord': {'lon': 77.5946, 'lat': 12.9716},
'list': [{main': {'aqi': 23, 'components': {'co': 634.19, 'no':
0, 'no2': 22.28, 'o3': 90.84, 'so2': 7.39, 'pm2_5': 16.34, 'pm1
0': 20.6, 'nh3': 5.38}, 'dt': 1742909693}]}
✅ Data sent to Kafka: {'coord': {'lon': 77.5946, 'lat': 12.9716},
'list': [{main': {'aqi': 23, 'components': {'co': 634.19, 'no':
0, 'no2': 22.28, 'o3': 90.84, 'so2': 7.39, 'pm2_5': 16.34, 'pm1
0': 20.6, 'nh3': 5.38}, 'dt': 1742909693}]}
✅ Data sent to Kafka: {'coord': {'lon': 77.5946, 'lat': 12.9716},
'list': [{main': {'aqi': 23, 'components': {'co': 634.19, 'no':
0, 'no2': 22.28, 'o3': 90.84, 'so2': 7.39, 'pm2_5': 16.34, 'pm1
0': 20.6, 'nh3': 5.38}, 'dt': 1742909693}]}
✅ Data sent to Kafka: {'coord': {'lon': 77.5946, 'lat': 12.9716},
'list': [{main': {'aqi': 23, 'components': {'co': 634.19, 'no':
0, 'no2': 22.28, 'o3': 90.84, 'so2': 7.39, 'pm2_5': 16.34, 'pm1
0': 20.6, 'nh3': 5.38}, 'dt': 1742909693}]}
✅ Data sent to Kafka: {'coord': {'lon': 77.5946, 'lat': 12.9716},
'list': [{main': {'aqi': 23, 'components': {'co': 634.19, 'no':
0, 'no2': 22.28, 'o3': 90.84, 'so2': 7.39, 'pm2_5': 16.34, 'pm1
0': 20.6, 'nh3': 5.38}, 'dt': 1742909693}]}
✅ Data sent to Kafka: {'coord': {'lon': 77.5946, 'lat': 12.9716},
'list': [{main': {'aqi': 23, 'components': {'co': 634.19, 'no':
0, 'no2': 22.28, 'o3': 90.84, 'so2': 7.39, 'pm2_5': 16.34, 'pm1
0': 20.6, 'nh3': 5.38}, 'dt': 1742909693}]}

```

Step 6: Create the Kafka Consumer

Consumer: (consumer.py)

```

from kafka import KafkaConsumer

import json

import pandas as pd


# Corrected Kafka consumer configuration

TOPIC_NAME = 'air-quality-data' # 🤝 Updated topic name

BOOTSTRAP_SERVERS = 'localhost:9092'

# Create a Kafka consumer

consumer = KafkaConsumer(
    TOPIC_NAME,
    bootstrap_servers=BOOTSTRAP_SERVERS,
    value_deserializer=lambda x: json.loads(x.decode('utf-8')),
    auto_offset_reset='earliest', # Read from the beginning if no offset is committed
    enable_auto_commit=True,
)

```

```
group_id='air-quality-group'  
)  
print("  Listening for incoming air quality data...")  
# Process incoming data  
for message in consumer:  
    air_data = message.value  
    print(f"Received data: {air_data}")
```

Step 7: Create a Real-Time Dashboard

Using StreamlitDashboard Creation:

```
import streamlit as st  
  
import pandas as pd  
  
import plotly.express as px  
  
# Page title  
  
st.title("Air Quality Monitoring Dashboard")  
  
# Sample air quality data  
  
data = {  
    "Timestamp": ["2025-03-25 18:00", "2025-03-25 18:05", "2025-03-25 18:10"],  
    "PM2.5": [12, 15, 18],  
    "PM10": [30, 35, 40],  
    "Ozone": [45, 50, 55],  
    "NO2": [60, 65, 70],  
    "CO": [75, 80, 85],  
    "SO2": [90, 95, 100]  
}
```

```
"PM2.5": [15.01, 16.34, 18.56],  
"PM10": [18.56, 20.60, 22.78],  
"CO": [594.14, 634.19, 650.43],  
"NO2": [21.08, 22.28, 23.34],  
"O3": [97.28, 90.84, 88.23],  
"AQI": [2, 2, 3]  
}  
  
# Create DataFrame  
  
df = pd.DataFrame(data)  
  
# Display data as table  
  
st.write("### 🌳 Raw Air Quality Data")  
  
st.dataframe(df)  
  
# Plot AQI over time  
  
fig = px.line(df, x="Timestamp", y="AQI", title="Air Quality Index Over Time")  
st.plotly_chart(fig)  
  
# Plot PM2.5 and PM10 levels  
  
fig_pm = px.line(df, x="Timestamp", y=["PM2.5", "PM10"], title="PM2.5 and PM10 Levels Over Time")  
st.plotly_chart(fig_pm)  
  
st.success("✅ Dashboard Updated Successfully!")
```

```
Command Prompt - streamlit  +  ~
You can now view your Streamlit app in your browser.

Local URL: http://localhost:8501
Network URL: http://192.168.0.202:8501

D:\air-quality-monitoring-system\dashboard\dashboard.py:16: SyntaxWarning: ...
    valid escape sequence '\d'
    file_path = "D:\air-quality-monitoring-system\dashboard\dashboard.py"
    Stopping...

D:\air-quality-monitoring-system\dashboard>streamlit run dashboard.py

You can now view your Streamlit app in your browser.

Local URL: http://localhost:8501
Network URL: http://192.168.0.202:8501

Stopping...

D:\air-quality-monitoring-system\dashboard>streamlit run dashboard.py

You can now view your Streamlit app in your browser.

Local URL: http://localhost:8501
Network URL: http://192.168.0.202:8501
```

Air Quality Monitoring System

📅 Today's Date: 2025-03-30 21:34:46

🕒 No CSV file found. Waiting for real-time data!

📊 Sample Air Quality Data

	Component	Concentration
0	CO	500.0000
1	NO	0.1000
2	NO2	20.0000
3	O3	95.0000
4	SO2	7.0000
5	PM2.5	15.0000
6	PM10	18.0000
7	NH3	3.0000

💡 Developed by: Air Quality Monitoring Team 🚀

CONCLUSION:

The Air Quality Monitoring System powered by Apache Kafka effectively addresses the need for real-time environmental data processing. By providing actionable insights and timely alerts, it empowers decision-makers to combat air pollution and safeguard public health. This project demonstrates the transformative impact of technology in building a sustainable and healthier future.