

Weather Data Streaming with Apache Kafka

Erickson Figueroa

Executive Summary:

The Weather Data Processing System collects, processes, and analyzes weather data from multiple cities and stores it in a PostgreSQL database. The system consists of four main components: producer, consumer, weather data module, and weather summary module.

Producer Component:

- The producer component fetches weather data from the OpenWeatherMap API for specified cities.
- It uses a Kafka producer to send the weather data to a Kafka topic named "weather-data."
- City names and weather data are retrieved using the `get_weather_data` function from the `weather_data` module.
- If the weather data fetches successfully, it sends it to the Kafka topic, and logging generates records for success, warning, and error events.

Consumer Component:

- The consumer component listens to the "weather-data" Kafka topic and retrieves weather data messages.
- Retrieved weather data is inserted into a PostgreSQL database table named "weather_detail."
- Generate logging records for successful data insertion and any encountered errors.

Weather Data Module:

- The `weather_data` module provides functionality to retrieve weather data for a given city from the OpenWeatherMap API.
- The `get_weather_data` function constructs the API URL, makes a GET request to the API, and parses the JSON response to extract relevant weather information.
- Negative temperature values are handled by converting them to strings and adding a "-" sign if they are negative.

Weather Summary Module:

- The `weather_summary` module summarizes the weather data stored in the "weather_detail" table.
- It calculates the minimum and maximum values for each city's temperature, wind speed, and humidity.

- The summarized data is formatted and inserted into a PostgreSQL table named "weather_summary."
- Implemented error handling to catch and log exceptions during the summary process.

Closing: The Weather Data Processing System provides a solution for collecting, storing, and summarizing weather data. It leverages Kafka for real-time data streaming and PostgreSQL for data persistence and integrates error logging to ensure system reliability. With its modular architecture and comprehensive functionality, the system effectively meets weather data management and analysis requirements.

Zookeeper service

```
adminuser@erickson-lab:~/Downloads/kafka_2.13-3.6.1$ bin/zookeeper-server-start.sh zookeeper.properties
```

```
adminuser@rickson-lab: ~$ bin/zookeeper-server-start.sh config/zookeeper.properties
2024-02-06 15:13:53.766 INFO Reading configuration from: config/zookeeper.properties (org.apache.zookeeper.server.quorum.QuorumPeerConfig)
2024-02-06 15:13:53.772 WARN config/zookeeper.properties is relative. Prepend ./ to indicate that you're sure! (org.apache.zookeeper.server.quorum.QuorumPeerConfig)
2024-02-06 15:13:53.779 INFO clientPortAddress is 0.0.0.0:2181 (org.apache.zookeeper.server.quorum.QuorumPeerConfig)
2024-02-06 15:13:53.779 INFO secureClientPort is not set (org.apache.zookeeper.server.quorum.QuorumPeerConfig)
2024-02-06 15:13:53.780 INFO observerMasterPort is not set (org.apache.zookeeper.server.quorum.QuorumPeerConfig)
2024-02-06 15:13:53.780 INFO metricsProvider.className is org.apache.zookeeper.metrics.impl.DefaultMetricsProvider (org.apache.zookeeper.server.quorum.QuorumPeerConfig)
2024-02-06 15:13:53.780 INFO autopurge.snapshotInCount sets to 3 (org.apache.zookeeper.server.DataDirCleanupManager)
2024-02-06 15:13:53.786 INFO autopurge.purgeInterval is 0 (org.apache.zookeeper.server.DataDirCleanupManager)
2024-02-06 15:13:53.786 INFO Purge task is not scheduled. (org.apache.zookeeper.server.DataDirCleanupManager)
2024-02-06 15:13:53.786 WARN Either no config or no quorum defined in config, running in standalone mode (org.apache.zookeeper.server.quorum.QuorumPeerMain)
2024-02-06 15:13:53.789 INFO Log4j 1.2 jmx support not found; jmx disabled. (org.apache.zookeeper.jmx.ManagedUtil)
2024-02-06 15:13:53.791 INFO Reading configuration from: config/zookeeper.properties (org.apache.zookeeper.server.quorum.QuorumPeerConfig)
2024-02-06 15:13:53.791 WARN config/zookeeper.properties is relative. Prepend ./ to indicate that you're sure! (org.apache.zookeeper.server.quorum.QuorumPeerConfig)
2024-02-06 15:13:53.791 INFO clientPortAddress is 0.0.0.0:2181 (org.apache.zookeeper.server.quorum.QuorumPeerConfig)
2024-02-06 15:13:53.792 INFO secureClientPort is not set (org.apache.zookeeper.server.quorum.QuorumPeerConfig)
2024-02-06 15:13:792 INFO observerMasterPort is not set (org.apache.zookeeper.server.quorum.QuorumPeerConfig)
2024-02-06 15:13:53.792 INFO metricsProvider.className is org.apache.zookeeper.metrics.impl.DefaultMetricsProvider (org.apache.zookeeper.server.quorum.QuorumPeerConfig)
2024-02-06 15:13:53.793 INFO Starting server (org.apache.zookeeper.server.ZooKeeperServerMain)
2024-02-06 15:13:53.848 INFO ServerMetrics initialized with provider org.apache.zookeeper.metrics.impl.DefaultMetricsProvider@691a7fbf (org.apache.zookeeper.server.ServerMetrics)
2024-02-06 15:13:53.862 INFO ACL digest algorithm is: SHA1 (org.apache.zookeeper.server.auth.DigestAuthenticationProvider)
2024-02-06 15:13:53.862 INFO Zookeeper.DigestAuthenticationProvider.enabled = true (org.apache.zookeeper.server.auth.DigestAuthenticationProvider)
2024-02-06 15:13:53.873 INFO zookeeper.snapshot.trust.empty = false (org.apache.zookeeper.server.persistence.FileTxnSnapLog)
2024-02-06 15:13:53.907 INFO (org.apache.zookeeper.server.ZooKeeperServer)
2024-02-06 15:13:53.908 INFO (org.apache.zookeeper.server.ZooKeeperServer)
2024-02-06 15:13:53.908 INFO (org.apache.zookeeper.server.ZooKeeperServer)
2024-02-06 15:13:53.908 INFO (org.apache.zookeeper.server.ZooKeeperServer)
2024-02-06 15:13:53.908 INFO (org.apache.zookeeper.server.ZooKeeperServer)
2024-02-06 15:13:53.909 INFO (org.apache.zookeeper.server.ZooKeeperServer)
2024-02-06 15:13:53.909 INFO (org.apache.zookeeper.server.ZooKeeperServer)
2024-02-06 15:13:53.910 INFO (org.apache.zookeeper.server.ZooKeeperServer)
2024-02-06 15:13:53.910 INFO (org.apache.zookeeper.server.ZooKeeperServer)
2024-02-06 15:13:53.910 INFO (org.apache.zookeeper.server.ZooKeeperServer)
2024-02-06 15:13:53.915 INFO Server environment:zookeeper.version=3.8.3-6ad6364c7c0bcfd6e452d54ebfa3058989ba56, built on 2023-10-05 10:34 UTC (org.apache.zookeeper.server.ZooKeeperServer)
2024-02-06 15:13:53.915 INFO Server environment:host.name=rickson-lab (org.apache.zookeeper.server.ZooKeeperServer)
2024-02-06 15:13:53.919 INFO Server environment:java.version=17.0.2 (org.apache.zookeeper.server.ZooKeeperServer)
2024-02-06 15:13:53.919 INFO Server environment:java.vendor=Private Build (org.apache.zookeeper.server.ZooKeeperServer)
2024-02-06 15:13:53.919 INFO Server environment:java.home=/usr/lib/jvm/java-17-openjdk-amd64 (org.apache.zookeeper.server.ZooKeeperServer)
2024-02-06 15:13:53.919 INFO Server environment:java.class.path=/home/adminuser/Downloads/kafka_2.13-3.6.1/bin/.:/lib/activation-1.1.1.jar:/home/adminuser/Downloads/kafka_2.13-3.6.1/bin/.:/lib/apollinaire-repackaged-2.6.1.jar:/home/adminuser/Downloads/kafka_2.13-3.6.1/bin/.:/lib/argparse4l-0.7.0.jar:/home/adminuser/Downloads/kafka_2.13-3.6.1/bin/.:/lib/audience-annotations-0.12.0.jar:/home/adminuser/Downloads/kafka_2.13-3.6.1/bin/.:/lib/caffeine-2.9.3.jar:/home/adminuser/Downloads/kafka_2.13-3.6.1/bin/.:/lib/checker-qual-3.19.0.jar
```

Kafka service

```
adminuser@erickson-lab:~/Downloads/kafka_2.13-3.6.1$ bin/kafka-server-start.sh config/server.properties
```

```
[2024-02-06 15:14:59,625] INFO Registered kafka:type=kafka.Log4jController MBean (kafka.utils.Log4jControllerRegistrations)
[2024-02-06 15:15:00,255] INFO Setting -Djdk.tls.rejectClientInitiatedRenegotiation=true to disable client-initiated TLS renegotiation (org.apache.zookeeper.common.XS509Util)
[2024-02-06 15:15:00,435] INFO Registered signal handlers for TERM, INT, HUP (org.apache.kafka.common.utils.LoggingSignalHandler)
[2024-02-06 15:15:00,438] INFO starting (kafka.server.KafkaServer)
[2024-02-06 15:15:00,443] INFO Connecting to zookeeper on localhost:2181 (kafka.server.KafkaServer)
[2024-02-06 15:15:00,479] INFO [ZooKeeperClient Kafka server] Initializing a new session to local host:2181. (Kafka.ZooKeeper.ZooKeeperClient)
[2024-02-06 15:15:00,487] INFO Client environment:zookeeper.version=3.8.3-6add6364c70bcf0d4e52d5defba3058098ab56, built on 2023-10-05 10:34 UTC (org.apache.zookeeper.ZooKeeper)
[2024-02-06 15:15:00,487] INFO Client environment:host.name=erickson-lab (org.apache.zookeeper.ZooKeeper)
[2024-02-06 15:15:00,488] INFO Client environment:java.version=17.0.9 (org.apache.zookeeper.ZooKeeper)
[2024-02-06 15:15:00,488] INFO Client environment:java.vendor=Private Build (org.apache.zookeeper.ZooKeeper)
[2024-02-06 15:15:00,488] INFO Client environment:java.home=/usr/lib/jvm/java-17-openjdk-amd64 (org.apache.zookeeper.ZooKeeper)
[2024-02-06 15:15:00,489] INFO Client environment:java.class.path=/home/adm/nuser/Downloads/kafka_2.13-3.6.1/bin/.:/lib/activation-1.1.1.jar:/home/adm/nuser/Downloa
1/bin/.:/Libs/apollance-repackaged-2.6.1.jar:/home/adm/nuser/Downloads/kafka_2.13-3.6.1/bin/.:/Libs/argparse4-0.7.0.jar:/home/adm/nuser/Downloads/kafka_2.13-3.6.1/bin/.:/Libs
nce-annotations-0.12.0.jar:/home/adm/nuser/Downloads/kafka_2.13-3.6.1/bin/.:/Libs/caffeine-2.9.3.jar:/home/adm/nuser/Downloads/kafka_2.13-3.6.1/bin/.:/Libs/checker-qual-3.19.0.f
adnuser/Downloads/kafka_2.13-3.6.1/bin/.:/Libs/commons-beanutils-1.9.4.jar:/home/adm/nuser/Downloads/kafka_2.13-3.6.1/bin/.:/Libs/commons-cli-1.4.jar:/home/adm/nuser/Downlo
1/bin/.:/Libs/commons-codec-1.15.0.jar:/home/adm/nuser/Downloads/kafka_2.13-3.6.1/bin/.:/Libs/commons-collections4-4.4.jar:/home/adm/nuser/Downloads/kafka_2.13-3.6.1/bi
pgAdmin4_operator-1.11.0.jar:/home/adm/nuser/Downloads/kafka_2.13-3.6.1/bin/.:/Libs/commons-lang3-3.8.1.jar:/home/adm/nuser/Downloads/kafka_2.13-3.6.1/bin/.:/Libs/commons-logg
2.jar:/home/adm/nuser/Downloads/kafka_2.13-3.6.1/bin/.:/Libs/commons-validator-1.7.jar:/home/adm/nuser/Downloads/kafka_2.13-3.6.1/bin/.:/Libs/connect-api-3.6.1.jar:/home/adm/nuse
nloads/kafka_2.13-3.6.1/bin/.:/Libs/connect-basic-auth-extension-3.6.1.jar:/home/adm/nuser/Downloads/kafka_2.13-3.6.1/bin/.:/Libs/connect-json-3.6.1.jar:/home/adm/nuser/Downloa
2.13-3.6.1/bin/.:/Libs/connect-mirror-3.6.1.jar:/home/adm/nuser/Downloads/kafka_2.13-3.6.1/bin/.:/Libs/connect-mirror-client-3.6.1.jar:/home/adm/nuser/Downloads/kafka_2.13-3.6
./Libs/connect-runtime-3.6.1.jar:/home/adm/nuser/Downloads/kafka_2.13-3.6.1/bin/.:/Libs/connect-transforms-3.6.1.jar:/home/adm/nuser/Downloads/kafka_2.13-3.6.1/bin/.:/Libs/err
nce-annotations-2.10.0.jar:/home/adm/nuser/Downloads/kafka_2.13-3.6.1/bin/.:/Libs/hk2-api-2.6.1.jar:/home/adm/nuser/Downloads/kafka_2.13-3.6.1/bin/.:/Libs/hk2-locator-2.6.1.jar:/h
adm/nuser/Downloads/kafka_2.13-3.6.1/bin/.:/Libs/hk2-utils-2.6.1.jar:/home/adm/nuser/Downloads/kafka_2.13-3.6.1/bin/.:/Libs/jackson-annotations-2.13.5.jar:/home/adm/nuser/Downloa
kafka_2.13-3.6.1/bin/.:/Libs/jackson-core-2.13.5.jar:/home/adm/nuser/Downloads/kafka_2.13-3.6.1/bin/.:/Libs/jackson-databind-2.13.5.jar:/home/adm/nuser/Downloads/kafka_2.13-3.6.1/b
Libs/jackson-dataformat-csv-2.13.5.jar:/home/adm/nuser/Downloads/kafka_2.13-3.6.1/bin/.:/Libs/jackson-datatype-jdk8-2.13.5.jar:/home/adm/nuser/Downloads/kafka_2.13-3.6.1/bin/.:/L
ckson-jaxrs-base-2.13.5.jar:/home/adm/nuser/Downloads/kafka_2.13-3.6.1/bin/.:/Libs/jackson-jaxrs-json-provider-2.13.5.jar:/home/adm/nuser/Downloads/kafka_2.13-3.6.1/bin/.:/Libs
on-module-jaxb-annotations-2.13.5.jar:/home/adm/nuser/Downloads/kafka_2.13-3.6.1/bin/.:/Libs/jackson-module-scala-2.13-2.13.5.jar:/home/adm/nuser/Downloads/kafka_2.13-3.6.1/bin/.:/L
s/jakarta.activation-api-1.2.2.jar:/home/adm/nuser/Downloads/kafka_2.13-3.6.1/bin/.:/Libs/jakarta.annotation-api-1.3.5.jar:/home/adm/nuser/Downloads/kafka_2.13-3.6.1/bin/.:/Libs t
ta.inject-2.6.1.jar:/home/adm/nuser/Downloads/kafka_2.13-3.6.1/bin/.:/Libs/jakarta.validation-api-2.0.2.jar:/home/adm/nuser/Downloads/kafka_2.13-3.6.1/bin/.:/Libs/jakarta.ws.rs-
1.6.jar:/home/adm/nuser/Downloads/kafka_2.13-3.6.1/bin/.:/Libs/jakarta.xml.bind-api-2.3.3.jar:/home/adm/nuser/Downloads/kafka_2.13-3.6.1/bin/.:/Libs/javasst-3.29.2-GA.jar:/hom
nuser/Downloads/kafka_2.13-3.6.1/bin/.:/Libs/javax.activation-api-1.2.0.jar:/home/adm/nuser/Downloads/kafka_2.13-3.6.1/bin/.:/Libs/javax.activation-api-1.3.2.jar:/home/adm/nuser/Downl
ods/kafka_2.13-3.6.1/bin/.:/Libs/javax.servlet-api-3.1.0.jar:/home/adm/nuser/Downloads/kafka_2.13-3.6.1/bin/.:/Libs/javax.rs-api-2.1.1.jar:/home/adm/nuser/Downloads/kafka_2.
2.13-3.6.1/bin/.:/Libs/jersey-container-servlet-2.39.1.jar:/home/adm/nuser/Downloads/kafka_2.13-3.6.1/bin/.:/Libs/jersey-container-servlet-core-2.39.1.jar:/home/adm/nuser/Downloa
kafka_2.13-3.6.1/bin/.:/Libs/jersey-container-servlet-2.39.1.jar:/home/adm/nuser/Downloads/kafka_2.13-3.6.1/bin/.:/Libs/jersey-container-servlet-core-2.39.1.jar:/home/adm/nuser/Downloa
kafka_2.13-3.6.1/bin/.:/Libs/jetty-http-11.0.20.jar:/home/adm/nuser/Downloads/kafka_2.13-3.6.1/bin/.:/Libs/jetty-io-11.0.20.jar:/home/adm/nuser/Downloads/kafka_2.13-3.6.1/bin/.:/Libs
jetty-server-11.0.20.jar:/home/adm/nuser/Downloads/kafka_2.13-3.6.1/bin/.:/Libs/jetty-util-11.0.20.jar:/home/adm/nuser/Downloads/kafka_2.13-3.6.1/bin/.:/Libs/jetty-webapp-11.0.20.ja
```

Creating a Kafka topic

```
(kafkaenv) adminuser@erickson-lab:~/Downloads/kafka_2.13-3.6.1$ ./bin/kafka-topics.sh --create --topic weather-data --bootstrap-server localhost:9092 --replication-factor 1 --partitions 2

(kafkaenv) adminuser@erickson-lab:~/Downloads/kafka_2.13-3.6.1$ ./bin/kafka-topics.sh --list --bootstrap-server localhost:9092
__consumer_offsets
quickstart-events
weather-data
(kafkaenv) adminuser@erickson-lab:~/Downloads/kafka_2.13-3.6.1$
```

The producer's performance (obtaining serialized data from OpenWeatherMap API and send to the consumer)

```
(kafkaenv) adminuser@erickson-lab:~/kafka_app_venv$ /home/adminuser/kafka_app_venv/kafkaenv/bin/python /home/adminuser/kafka_app_venv/kafkaenv/kafka_weather_app/producer.py
{'city_name': 'Winnipeg', 'current temperature': '1.42', 'current wind speed': 4.63, 'current humidity': 98}
{'city_name': 'Vancouver', 'current temperature': '4.69', 'current wind speed': 7.72, 'current humidity': 87}
{'city_name': 'Winnipeg', 'current temperature': '1.78', 'current wind speed': 5.14, 'current humidity': 98}
{'city_name': 'Vancouver', 'current temperature': '4.69', 'current wind speed': 7.72, 'current humidity': 87}
{'city_name': 'Winnipeg', 'current temperature': '1.42', 'current wind speed': 4.63, 'current humidity': 98}
{'city_name': 'Vancouver', 'current temperature': '4.69', 'current wind speed': 7.72, 'current humidity': 87}
{'city_name': 'Winnipeg', 'current temperature': '1.55', 'current wind speed': 5.14, 'current humidity': 98}
{'city_name': 'Vancouver', 'current temperature': '4.69', 'current wind speed': 7.72, 'current humidity': 87}
{'city_name': 'Winnipeg', 'current temperature': '1.42', 'current wind speed': 4.63, 'current humidity': 98}
{'city_name': 'Vancouver', 'current temperature': '4.69', 'current wind speed': 7.72, 'current humidity': 87}
{'city_name': 'Winnipeg', 'current temperature': '1.78', 'current wind speed': 5.14, 'current humidity': 98}
{'city_name': 'Vancouver', 'current temperature': '4.69', 'current wind speed': 7.72, 'current humidity': 87}
{'city_name': 'Winnipeg', 'current temperature': '1.42', 'current wind speed': 4.63, 'current humidity': 98}
{'city_name': 'Vancouver', 'current temperature': '4.69', 'current wind speed': 7.72, 'current humidity': 87}
{'city_name': 'Winnipeg', 'current temperature': '1.55', 'current wind speed': 5.14, 'current humidity': 98}
{'city_name': 'Vancouver', 'current temperature': '4.69', 'current wind speed': 7.72, 'current humidity': 87}
{'city_name': 'Winnipeg', 'current temperature': '1.42', 'current wind speed': 4.63, 'current humidity': 98}
{'city_name': 'Vancouver', 'current temperature': '4.69', 'current wind speed': 7.72, 'current humidity': 87}
{'city_name': 'Winnipeg', 'current temperature': '1.78', 'current wind speed': 5.14, 'current humidity': 98}
{'city_name': 'Vancouver', 'current temperature': '4.69', 'current wind speed': 7.72, 'current humidity': 87}
{'city_name': 'Winnipeg', 'current temperature': '1.42', 'current wind speed': 4.63, 'current humidity': 98}
{'city_name': 'Vancouver', 'current temperature': '4.69', 'current wind speed': 7.72, 'current humidity': 87}
{'city_name': 'Winnipeg', 'current temperature': '1.55', 'current wind speed': 5.14, 'current humidity': 98}
{'city_name': 'Vancouver', 'current temperature': '4.69', 'current wind speed': 7.72, 'current humidity': 87}
{'city_name': 'Winnipeg', 'current temperature': '1.42', 'current wind speed': 4.63, 'current humidity': 98}
{'city_name': 'Vancouver', 'current temperature': '4.69', 'current wind speed': 7.72, 'current humidity': 87}
{'city_name': 'Winnipeg', 'current temperature': '1.78', 'current wind speed': 5.14, 'current humidity': 98}
{'city_name': 'Vancouver', 'current temperature': '4.69', 'current wind speed': 7.72, 'current humidity': 87}
{'city_name': 'Winnipeg', 'current temperature': '1.42', 'current wind speed': 4.63, 'current humidity': 98}
```

The consumer's performance (obtain the deserialized data from the producer and insert it into the database)

```
(kafkaenv) adminuser@rickson-lab:~/kafka_app_venv$ /home/adminuser/kafka_app_venv/kafkaenv/bin/python /home/adminuser/kafka_app_venv/kafkaenv/kafka_weather_app/consumer.py
/home/adminuser/kafka_app_venv/kafkaenv/kafka_weather_app/weather_summary.py:16: UserWarning: pandas only supports SQLAlchemy connectable (engine/connection) or database string URI or sql
tes DBAPI2 connection. Other DBAPI2 objects are not tested. Please consider using SQLAlchemy.
  weather_df = pd.read_sql(query, conn)
Received weather data: {'city_name': 'Winnipeg', 'current_temperature': '1.55', 'current_wind_speed': 3.6, 'current_humidity': 98}
The weather data were inserted into the database
Received weather data: {'city_name': 'Vancouver', 'current_temperature': '4.85', 'current_wind_speed': 6.17, 'current_humidity': 87}
The weather data were inserted into the database
Received weather data: {'city_name': 'Vancouver', 'current_temperature': '4.85', 'current_wind_speed': 6.17, 'current_humidity': 87}
The weather data were inserted into the database
Received weather data: {'city_name': 'Winnipeg', 'current_temperature': '1.55', 'current_wind_speed': 3.6, 'current_humidity': 98}
The weather data were inserted into the database
Received weather data: {'city_name': 'Vancouver', 'current_temperature': '4.69', 'current_wind_speed': 7.72, 'current_humidity': 87}
The weather data were inserted into the database
Received weather data: {'city_name': 'Winnipeg', 'current_temperature': '1.42', 'current_wind_speed': 4.63, 'current_humidity': 98}
The weather data were inserted into the database
Received weather data: {'city_name': 'Vancouver', 'current_temperature': '4.69', 'current_wind_speed': 7.72, 'current_humidity': 87}
The weather data were inserted into the database
Received weather data: {'city_name': 'Winnipeg', 'current_temperature': '1.55', 'current_wind_speed': 5.14, 'current_humidity': 98}
The weather data were inserted into the database
Received weather data: {'city_name': 'Vancouver', 'current_temperature': '4.69', 'current_wind_speed': 7.72, 'current_humidity': 87}
The weather data were inserted into the database
Received weather data: {'city_name': 'Winnipeg', 'current_temperature': '1.42', 'current_wind_speed': 4.63, 'current_humidity': 98}
The weather data were inserted into the database
Received weather data: {'city_name': 'Vancouver', 'current_temperature': '4.69', 'current_wind_speed': 7.72, 'current_humidity': 87}
The weather data were inserted into the database
Received weather data: {'city_name': 'Winnipeg', 'current_temperature': '1.42', 'current_wind_speed': 4.63, 'current_humidity': 98}
The weather data were inserted into the database
Received weather data: {'city_name': 'Vancouver', 'current_temperature': '4.69', 'current_wind_speed': 7.72, 'current_humidity': 87}
The weather data were inserted into the database
Received weather data: {'city_name': 'Winnipeg', 'current_temperature': '1.78', 'current_wind_speed': 5.14, 'current_humidity': 98}
The weather data were inserted into the database
Received weather data: {'city_name': 'Vancouver', 'current_temperature': '4.69', 'current_wind_speed': 7.72, 'current_humidity': 87}
The weather data were inserted into the database
Received weather data: {'city_name': 'Vancouver', 'current_temperature': '4.69', 'current_wind_speed': 7.72, 'current_humidity': 87}
```

Table to capture summary data (weather_summary)

```
22 select * from weather_summary order by 1 desc
```

Data Output Messages Notifications

	id [PK] integer	city character varying (100)	temperature_deg_c character varying (100)	wind_speed_km_h character varying (100)	humidity_percentage character varying (100)	received_at timestamp without time zone
1	12	Winnipeg	-1.11 to 1.92	3.6 to 4.58	70 to 98	2024-02-08 06:31:24.233372
2	11	Vancouver	4.85 to 5.01	6.17 to 6.71	86 to 87	2024-02-08 06:31:24.233372

Table to capture the complete data without summarizing (weather_detail)

```
25 select * from weather_detail
```

```
26
```

Data Output Messages Notifications

	id [PK] integer	city_name character varying (100)	temperature character varying (100)	wind_speed character varying (100)	humidity character varying (100)	received_at timestamp without time zone
1	1	Winnipeg	1.92	3.6	98	2024-02-08 04:29:52.326443
2	2	Vancouver	5.01	6.71	86	2024-02-08 04:29:52.335367
3	3	Winnipeg	1.92	3.6	98	2024-02-08 04:29:52.343267
4	4	Vancouver	5	6.71	87	2024-02-08 04:29:52.34836
5	5	Winnipeg	1.92	3.6	98	2024-02-08 04:29:52.352756
6	6	Vancouver	5	6.71	87	2024-02-08 04:29:52.355707
7	7	Winnipeg	1.92	3.6	98	2024-02-08 04:29:52.358019
8	8	Vancouver	5	6.71	86	2024-02-08 04:29:52.360764
9	9	Winnipeg	1.92	3.6	98	2024-02-08 04:29:52.362917
10	10	Winnipeg	1.92	3.6	98	2024-02-08 04:29:52.366994
11	11	Winnipeg	1.92	3.6	98	2024-02-08 04:29:52.369207
12	12	Vancouver	5.01	6.71	86	2024-02-08 04:29:52.371673
13	13	Vancouver	5	6.71	87	2024-02-08 04:29:52.373255
14	14	Winnipeg	1.92	3.6	98	2024-02-08 04:29:52.375045
15	15	Vancouver	5	6.71	87	2024-02-08 04:29:52.377009

Table structures

```

> postgres 1 -- Table of all weather insert details
> weather_app 2 CREATE TABLE weather_detail (
> Casts 3 id SERIAL PRIMARY KEY,
> Catalogs 4 city_name VARCHAR(100),
> Event Triggers 5 temperature VARCHAR(100),
> Extensions 6 wind_speed VARCHAR(100),
> Foreign Data Wrappers 7 humidity VARCHAR(100),
> Languages 8 received_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
> Publications 9 );
> Schemas 10
> Subscriptions 11 -- Table to summary the data
> Login/Group Roles 12 CREATE TABLE weather_summary (
> Tablespaces 13 id SERIAL PRIMARY KEY,
14 city VARCHAR(100),
15 temperature_deg_c VARCHAR(100),
16 wind_speed_km_h VARCHAR(100),
17 humidity_percentage VARCHAR(100),
18 received_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
19 );
20
21

```

Producer's log file:

```

2024-02-08 06:20:57.694 - DEBUG - Sending request ProduceRequest v7(transactional_id=None, required_acks=1, timeout=30000, topics=[(topic='weather-data', partitions=[(partition=0, message=
2024-02-08 06:20:57.695 - DEBUG - <brkcrConnection node id=0 host=erickson-lab:9092 <connected> [IPv4 ('127.0.1.1', 9092)]> Request 4: ProduceRequest v7(transactional_id=None, required_acks=1, timeout=30000, topics=[(topic='weather-data', partitions=[(partition=0, message=
2024-02-08 06:20:57.695 - INFO - Weather data sent to Kafka successfully for Winnipeg
2024-02-08 06:20:57.697 - DEBUG - Starting new HTTP connection (1): api.openweathermap.org:80
2024-02-08 06:20:57.701 - DEBUG - Received correlation id: 4
2024-02-08 06:20:57.705 - DEBUG - Processing response ProduceResponse v7
2024-02-08 06:20:57.706 - DEBUG - <brkcrConnection node id=0 host=erickson-lab:9092 <connected> [IPv4 ('127.0.1.1', 9092)]> Response 4 (10.869741429019336 ms): ProduceResponse v7(topics=[(topic='weather-data', partitions=[(partition=0, error_code=0, offset=4235, timestamp=1, log_start_offset=0, message_offset=0)])])
2024-02-08 06:20:57.707 - DEBUG - Parsing produce response: ProduceResponse v7(topics=[(topic='weather-data', partitions=[(partition=0, error_code=0, offset=4235, timestamp=1, log_start_offset=0, message_offset=0)])])
2024-02-08 06:20:57.707 - DEBUG - Produced messages to topic-partition TopicPartition(topic='weather-data', partition=0) with base offset 4235 log start offset 0 and error None.
2024-02-08 06:20:57.860 - DEBUG - http://api.openweathermap.org:80 "GET /data/2.5/weather?appid=bdad30df8e8bbae91ddf2c8ad7f446d6q=Vancouver&units=metric HTTP/1.1" 200 476
2024-02-08 06:20:57.861 - DEBUG - Sending (key=None value={'city name': 'Vancouver', 'current temperature': '4.69', 'current wind speed': '7.72', 'current humidity': '87'}) headers={} to TopicPartition(topic='weather-data', partition=0)
2024-02-08 06:20:57.861 - DEBUG - Allocating a new 16384 byte message buffer for TopicPartition(topic='weather-data', partition=0)
2024-02-08 06:20:57.861 - DEBUG - Waking up the sender since TopicPartition(topic='weather-data', partition=0) is either full or getting a new batch
2024-02-08 06:20:57.861 - INFO - Weather data sent to Kafka successfully for Vancouver.
2024-02-08 06:20:57.862 - DEBUG - Nodes with data ready to send: (0)
2024-02-08 06:20:57.862 - DEBUG - Created 1 produce requests: (0: ProduceRequest v7(transactional_id=None, required_acks=1, timeout=30000, topics=[(topic='weather-data', partitions=[(partition=0, message=
2024-02-08 06:20:57.862 - DEBUG - Sending Produce Request: ProduceRequest v7(transactional_id=None, required_acks=1, timeout=30000, topics=[(topic='weather-data', partitions=[(partition=0, message=
2024-02-08 06:20:57.863 - DEBUG - <brkcrConnection node id=0 host=erickson-lab:9092 <connected> [IPv4 ('127.0.1.1', 9092)]> Request 5: ProduceRequest v7(transactional_id=None, required_acks=1, timeout=30000, topics=[(topic='weather-data', partitions=[(partition=0, message=
2024-02-08 06:20:57.865 - DEBUG - Received correlation id: 5
2024-02-08 06:20:57.866 - DEBUG - Processing response ProduceResponse v7
2024-02-08 06:20:57.866 - DEBUG - <brkcrConnection node id=0 host=erickson-lab:9092 <connected> [IPv4 ('127.0.1.1', 9092)]> Response 5 (3.409862518310547 ms): ProduceResponse v7(topics=[(topic='weather-data', partitions=[(partition=0, error_code=0, offset=4236, timestamp=1, log_start_offset=0, message_offset=0)])])
2024-02-08 06:20:57.867 - DEBUG - Parsing produce response: ProduceResponse v7(topics=[(topic='weather-data', partitions=[(partition=0, error_code=0, offset=4236, timestamp=1, log_start_offset=0, message_offset=0)])])
2024-02-08 06:20:57.867 - DEBUG - Produced messages to topic-partition TopicPartition(topic='weather-data', partition=0) with base offset 4236 log start offset 0 and error None.
2024-02-08 06:21:02.165 - DEBUG - Starting new HTTP connection (1): api.openweathermap.org:80
2024-02-08 06:21:03.163 - DEBUG - http://api.openweathermap.org:80 "GET /data/2.5/weather?appid=bdad30df8e8bbae91ddf2c8ad7f446d6q=Winnipeg&units=metric HTTP/1.1" 200 539
2024-02-08 06:21:03.165 - DEBUG - Sending (key=None value={'city name': 'Winnipeg', 'current temperature': '1.42', 'current wind speed': '4.63', 'current humidity': '98'}) headers={} to TopicPartition(topic='weather-data', partition=0)
2024-02-08 06:21:03.166 - DEBUG - Allocating a new 16384 byte message buffer for TopicPartition(topic='weather-data', partition=0)
2024-02-08 06:21:03.167 - DEBUG - Waking up the sender since TopicPartition(topic='weather-data', partition=0) is either full or getting a new batch
2024-02-08 06:21:03.168 - DEBUG - Nodes with data ready to send: (0)
2024-02-08 06:21:03.168 - DEBUG - Created 1 produce requests: (0: ProduceRequest v7(transactional_id=None, required_acks=1, timeout=30000, topics=[(topic='weather-data', partitions=[(partition=0, message=
2024-02-08 06:21:03.169 - DEBUG - Sending Produce Request: ProduceRequest v7(transactional_id=None, required_acks=1, timeout=30000, topics=[(topic='weather-data', partitions=[(partition=0, message=

```

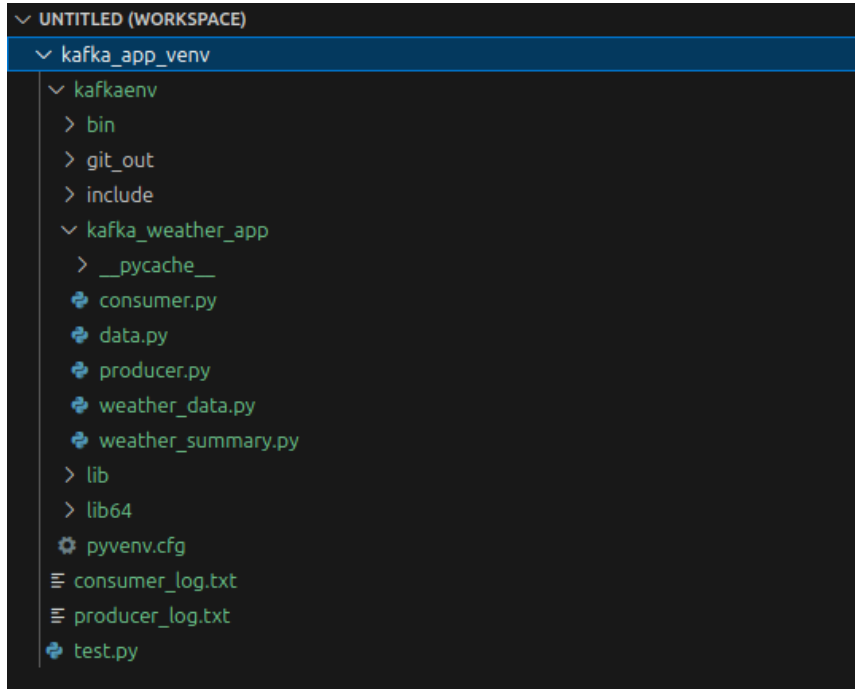
Consumer's log file:

```

2024-02-08 04:52:25.217 - INFO - Weather data inserted into PostgreSQL successfully!
2024-02-08 04:52:25.222 - INFO - Weather data inserted into PostgreSQL successfully!
2024-02-08 04:52:25.228 - INFO - Weather data inserted into PostgreSQL successfully!
2024-02-08 04:52:25.235 - INFO - Weather data inserted into PostgreSQL successfully!
2024-02-08 04:52:25.244 - INFO - Weather data inserted into PostgreSQL successfully!
2024-02-08 04:52:25.246 - INFO - Weather data inserted into PostgreSQL successfully!
2024-02-08 04:52:25.250 - INFO - Weather data inserted into PostgreSQL successfully!
2024-02-08 04:52:25.252 - INFO - Weather data inserted into PostgreSQL successfully!
2024-02-08 04:52:25.255 - INFO - Heartbeat: weather-data-group[55] kafka-python-2.0.2-ce6659e5-1f07-4b51-98d2-bcd437638019
2024-02-08 04:52:25.256 - DEBUG - Sending request HeartbeatRequest v1(group='weather-data-group', generation_id=55, member_id='kafka-python-2.0.2-ce6659e5-1f07-4b51-98d2-bcd437638019')
2024-02-08 04:52:25.256 - DEBUG - <brkcrConnection node id=coordinator-0 host=erickson-lab:9092 <connected> [IPv4 ('127.0.1.1', 9092)]> Request 5: HeartbeatRequest v1(group='weather-data-group', generation_id=55, member_id='kafka-python-2.0.2-ce6659e5-1f07-4b51-98d2-bcd437638019')
2024-02-08 04:52:25.259 - INFO - Weather data inserted into PostgreSQL successfully!
2024-02-08 04:52:25.261 - INFO - Weather data inserted into PostgreSQL successfully!
2024-02-08 04:52:25.269 - INFO - Weather data inserted into PostgreSQL successfully!
2024-02-08 04:52:25.272 - INFO - Weather data inserted into PostgreSQL successfully!
2024-02-08 04:52:25.275 - INFO - Weather data inserted into PostgreSQL successfully!
2024-02-08 04:52:25.280 - INFO - Weather data inserted into PostgreSQL successfully!
2024-02-08 04:52:25.283 - INFO - Weather data inserted into PostgreSQL successfully!
2024-02-08 04:52:25.288 - INFO - Weather data inserted into PostgreSQL successfully!
2024-02-08 04:52:25.295 - INFO - Weather data inserted into PostgreSQL successfully!
2024-02-08 04:52:25.300 - INFO - Weather data inserted into PostgreSQL successfully!
2024-02-08 04:52:25.312 - INFO - Weather data inserted into PostgreSQL successfully!
2024-02-08 04:52:25.314 - DEBUG - Sending offset-commit request with [TopicPartition(topic='weather-data', partition=0): OffsetAndMetadata(offset=2396, metadata=''), TopicPartition(topic='weather-data', partition=0): OffsetAndMetadata(offset=2396, metadata='')]
2024-02-08 04:52:25.315 - DEBUG - Sending request OffsetCommitRequest v2(consumer_group='weather-data-group', consumer_group_generation_id=55, consumer_id='kafka-python-2.0.2-ce6659e5-1f07-4b51-98d2-bcd437638019')
2024-02-08 04:52:25.315 - DEBUG - <brkcrConnection node id=coordinator-0 host=erickson-lab:9092 <connected> [IPv4 ('127.0.1.1', 9092)]> Request 6: OffsetCommitRequest v2(consumer_group='weather-data-group', generation_id=55, consumer_id='kafka-python-2.0.2-ce6659e5-1f07-4b51-98d2-bcd437638019')
2024-02-08 04:52:25.316 - DEBUG - Received correlation id: 5
2024-02-08 04:52:25.317 - DEBUG - Processing response HeartbeatResponse v1
2024-02-08 04:52:25.317 - DEBUG - <brkcrConnection node id=coordinator-0 host=erickson-lab:9092 <connected> [IPv4 ('127.0.1.1', 9092)]> Response 5 (61.07807159423828 ms): HeartbeatResponse v1
2024-02-08 04:52:25.318 - DEBUG - Received successful heartbeat response for group weather-data-group

```

Project structure



Sample API response result from OpenWeatherMap:

api.openweathermap.org/data/2.5/weather?appid=bdadc30df8e8bbae91ddf2c8ad7f446d&q=Winnipeg&units=metric	
JSON	Raw Data
Save	Copy
Collapse All	Expand All
Filter JSON	
<pre> lon: -97.147 lat: 49.8844 weather: 0: id: 300 main: "Drizzle" description: "light intensity drizzle" icon: "09n" 1: id: 701 main: "Mist" description: "mist" icon: "50n" 2: id: 600 main: "Snow" description: "light snow" icon: "13n" base: "stations" main: temp: 1.52 feels_like: -3.09 temp_min: 0.58 temp_max: 2.75 pressure: 995 humidity: 98 visibility: 4023 wind: speed: 5.14 deg: 30 snow: th: 0.11 clouds: all: 100 dt: 1707397562 sys: type: 1 id: 738 country: "CA" sunrise: 1707488308 sunset: 1707435227 timezone: -21600 id: 6183235 name: "Winnipeg" cod: 200 </pre>	

Source Code

Weather_data.py (to get the data from the OpenWeatherMap API):

```
import requests

# API key from openweathermap.org
API_KEY = "bdadc30df8e8bbae91ddf2c8ad7f446d"

# Base URL for the OpenWeatherMap API
BASE_URL = "http://api.openweathermap.org/data/2.5/weather?appid="

# Constant to define the parameter units as metrics (Celsius)
CELSIUS = "metric"

def convert_temperature(temperature):
    """
    Convert temperature to a string and add "-" if it's negative.

    Args:
    temperature (float): The temperature value.

    Returns:
    str: The temperature as a string.
    """
    return str(temperature) if temperature >= 0 else "-" + str(abs(temperature))

def get_weather_data(city_name):
    """
    Retrieve weather data for a given city from the OpenWeatherMap API.

    Args:
    city_name (str): The name of the city.

    Returns:
    dict: A dictionary containing the weather data, or None if the city is not found.
    """

    # Construct the complete URL
    url = f"{BASE_URL}{API_KEY}&q={city_name}&units={CELSIUS}"

    # Make a GET request to the OpenWeatherMap API
    response = requests.get(url)

    # Check if the city is found
    if response.status_code == 200:
```

```

# Parse the JSON response
data = response.json()

# Extract relevant weather information
weather_data = {
    "city_name": city_name,
    "current_temperature": convert_temperature(data["main"]["temp"]),
    "current_wind_speed": data["wind"]["speed"],
    "current_humidity": data["main"]["humidity"]
}
return weather_data
else:
    print(f"Error retrieving weather data: {response.status_code}")
    return None

```

producer.py (to get the data from the OpenWeatherMap API and send to Kafka topic):

```

import logging
from kafka import KafkaProducer
import json
import time
from weather_data import get_weather_data

# Configure logging
logging.basicConfig(filename='producer_log.txt', level=logging.DEBUG,
                    format='%(asctime)s - %(levelname)s - %(message)s')

# Kafka producer settings
producer = KafkaProducer(bootstrap_servers=['localhost:9092'],
                        value_serializer=lambda x: json.dumps(x).encode('utf-8'))

# List of city names to get the weather data
city_names = ["Winnipeg", "Vancouver"]

# Process to send the weather data to kafka consumer
if __name__ == "__main__":
    while True:
        for city_name in city_names:
            try:
                # Get weather data
                weather_data = get_weather_data(city_name)
                print(weather_data)
                if weather_data:
                    # Send weather data to Kafka
                    producer.send("weather-data", weather_data)
                    logging.info(f"Weather data sent to Kafka successfully for {city_name}")
            else:

```



```

logging.warning(f"City {city_name} not found.")
except Exception as e:
    logging.error(f"An error occurred sending the data for {city_name}: {str(e)}")

# Wait for 5 seconds before sending the next request
time.sleep(5)

```

consumer.py (to get the data from the producer and insert to the database):

```

import logging
from kafka import KafkaConsumer
import json
import psycopg2
from weather_summary import weather_summary

# Configure logging
logging.basicConfig(filename='consumer_log.txt', level=logging.DEBUG,
format='% (asctime)s - % (levelname)s - % (message)s')

# Kafka consumer settings
consumer = KafkaConsumer('weather-data',
group_id='weather-data-group',
bootstrap_servers=['localhost:9092'],
value_deserializer=lambda x: json.loads(x.decode('utf-8')))

# PostgreSQL connection settings
conn = psycopg2.connect(dbname='weather_app', user='postgres', password='admin@01',
host='localhost')
cur = conn.cursor()

if __name__ == "__main__":
    for message in consumer:
        weather_data = message.value
        print("Received weather data:", weather_data)

# Insert weather data into PostgreSQL database
try:
    cur.execute("""
INSERT INTO weather_detail (city_name, temperature, wind_speed, humidity)
VALUES (%s, %s, %s, %s)
""", (weather_data['city_name'], weather_data['current_temperature'],
weather_data['current_wind_speed'], weather_data['current_humidity']))
    conn.commit()
    print("The weather data were inserted into the database")
    logging.info("Weather data inserted into PostgreSQL successfully!")
except Exception as e:
    logging.error("Error inserting weather data into PostgreSQL: %s", str(e))

```

```
cur.close()
conn.close()
```

After processing the data, call the weather_summary function to summarize the final data
weather_summary()

weather_summary.py (to summarize the final data into the weather_summary table):

```
import pandas as pd
import psycopg2

def weather_summary():
    try:
        # PostgreSQL connection settings
        conn = psycopg2.connect(dbname='weather_app', user='postgres',
                                password='admin@01', host='localhost')

        # Query to fetch weather details
        query = """
        SELECT city_name, temperature, wind_speed, humidity
        FROM weather_detail
        """

        # Fetch data from PostgreSQL into a DataFrame
        weather_df = pd.read_sql_query(query, conn)

        # Process the data to calculate min and max values for each city
        summary_df = weather_df.groupby('city_name').agg({
            'temperature': ['min', 'max'],
            'wind_speed': ['min', 'max'],
            'humidity': ['min', 'max']
        }).reset_index()

        # Format the data as required
        summary_df.columns = ['City', 'Min Temperature, deg C', 'Max Temperature,
                               deg C', 'Min Wind speed, km/h', 'Max Wind speed, km/h', 'Min Humidity, %',
                               'Max Humidity, %']
        summary_df['Min Temperature, deg C'] = summary_df['Min Temperature, deg
        C'].astype(str) + " to " + summary_df['Max Temperature, deg
        C'].astype(str)
        summary_df['Min Wind speed, km/h'] = summary_df['Min Wind speed,
        km/h'].astype(str) + " to " + summary_df['Max Wind speed,
        km/h'].astype(str)
```

```

summary_df['Min Humidity, %'] = summary_df['Min Humidity, %'].astype(str)
+ " to " + summary_df['Max Humidity, %'].astype(str)
summary_df.drop(columns=['Max Temperature, deg C', 'Max Wind speed, km/h',
'Max Humidity, %'], inplace=True)

# Insert the summarized data into the weather_summary table
with conn.cursor() as cur:
for _, row in summary_df.iterrows():
cur.execute("""
INSERT INTO weather_summary (City, Temperature_deg_c, Wind_speed_km_h,
Humidity_percentage)
VALUES (%s, %s, %s, %s)
""", (row['City'], row['Min Temperature, deg C'], row['Min Wind speed,
km/h'], row['Min Humidity, %']))
conn.commit()

# Close the database connection
conn.close()
except Exception as e:
print(f"An error occurred: {e}")

# Calling the function weather_summary to summarize the final data
weather_summary()

```