CSE-4E3 Designing Data Intensive Applications

Project: Weather Stations Monitoring

Team Members

AbdElaziz Mohamed AbdElaziz	19015941
Veronica Romany Hanna	19016156
Mark Ehab Latif	19016213

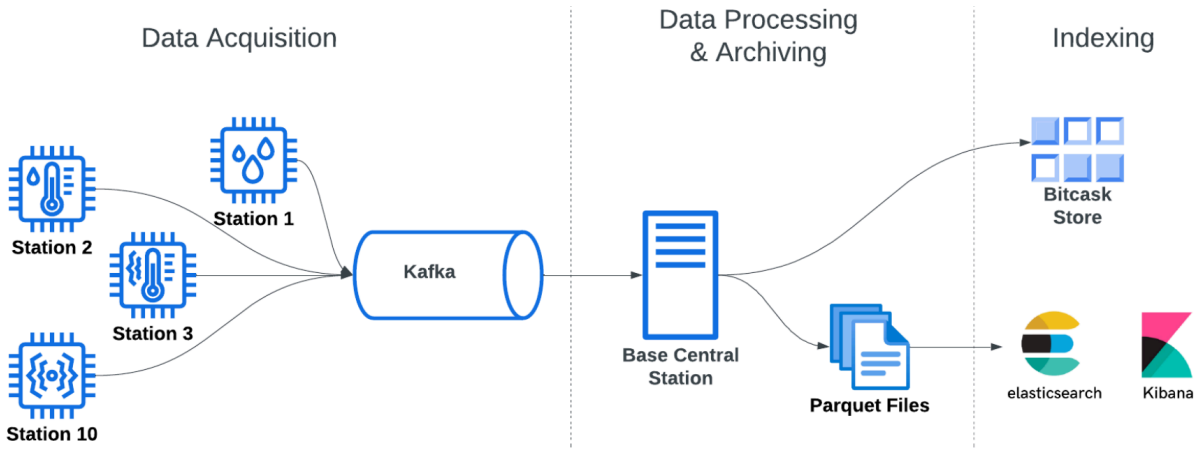
Problem Statement

The Internet of Things (IoT) is an important source of data streams in the modern digital world.The “Things” are huge in count and emit messages in very high frequency which flood the global internet. Hence, efficient stream processing is inevitable.

One use case is the distributed weather stations use case. Each “weather station” emits readings for the current weather status to the “central base station” for persistence and analysis. In this project, you will be required to implement the architecture of a weather monitoring system.

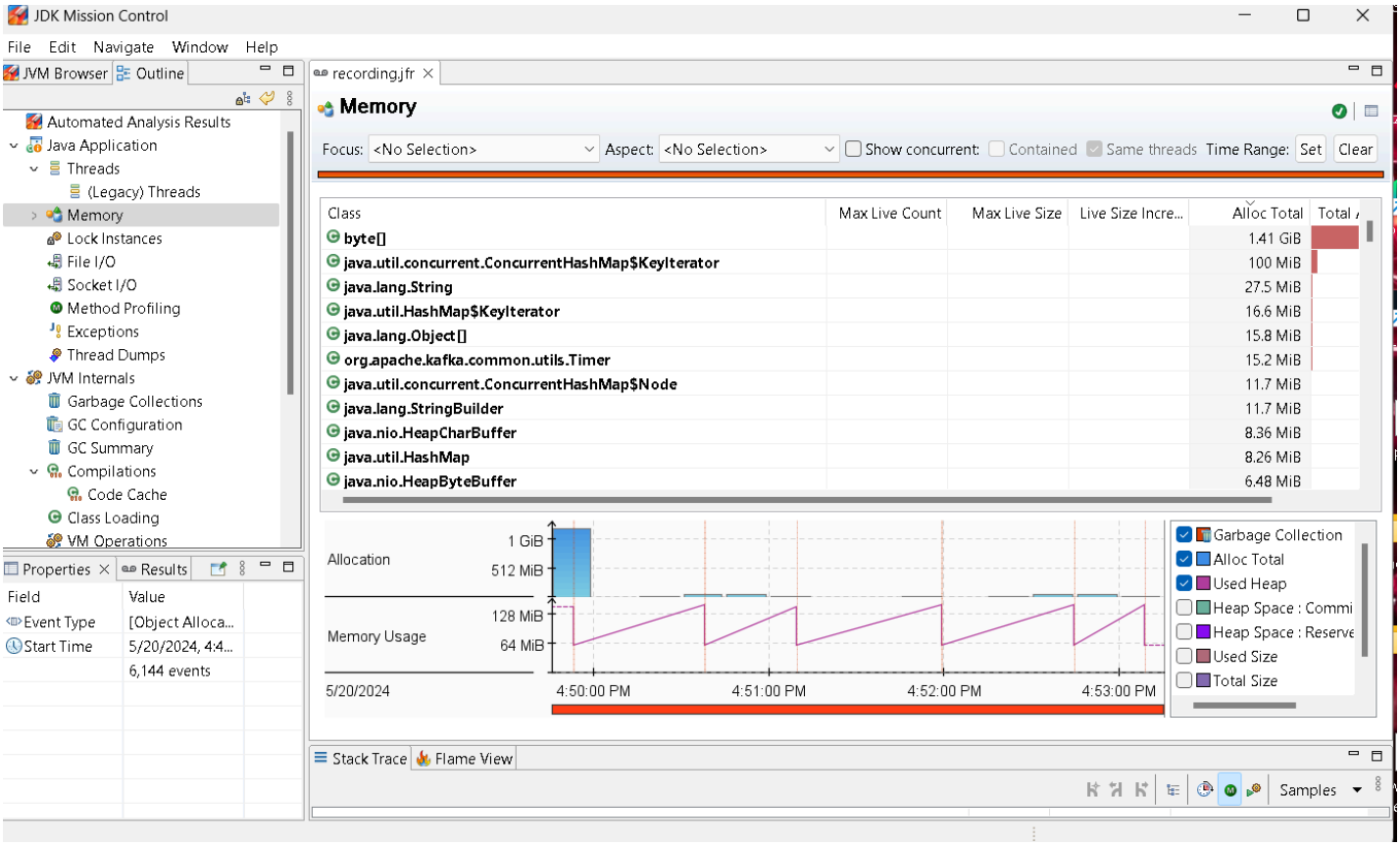
The system is composed of three stages:

- **Data Acquisition:** multiple weather stations that feed a queueing service (Kafka) with their readings.
- **Data Processing & Archiving:** The base central station is consuming the streamed data and archiving all data in the form of Parquet files.
- **Indexing:** two variants of index are maintained
  - Key-value store (Bitcask) for the latest reading from each individual station.
  - ElasticSearch / Kibana that are running over the Parquet files.

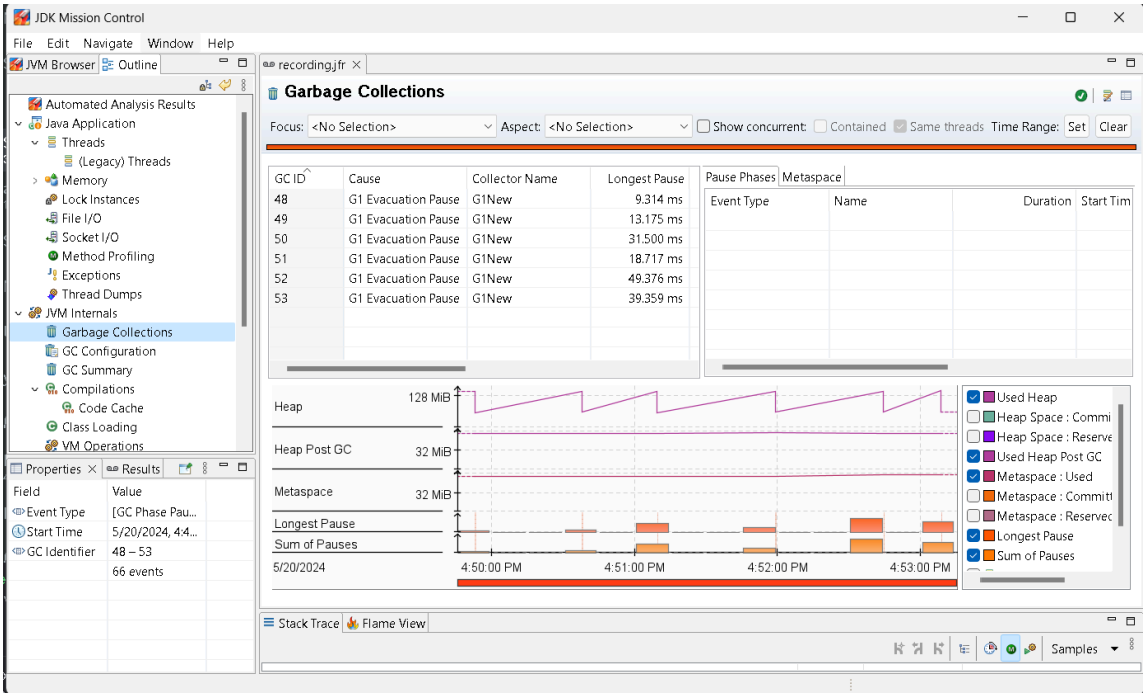


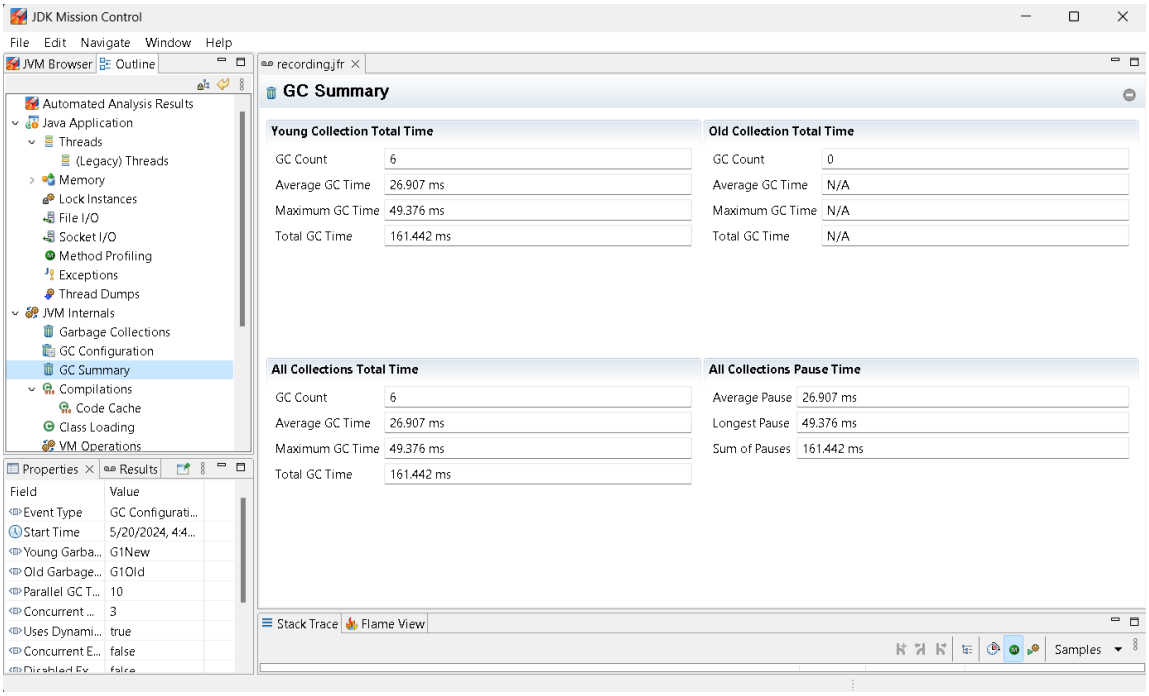
Profile Central Station using JFR

- Top 10 Classes with highest total memory

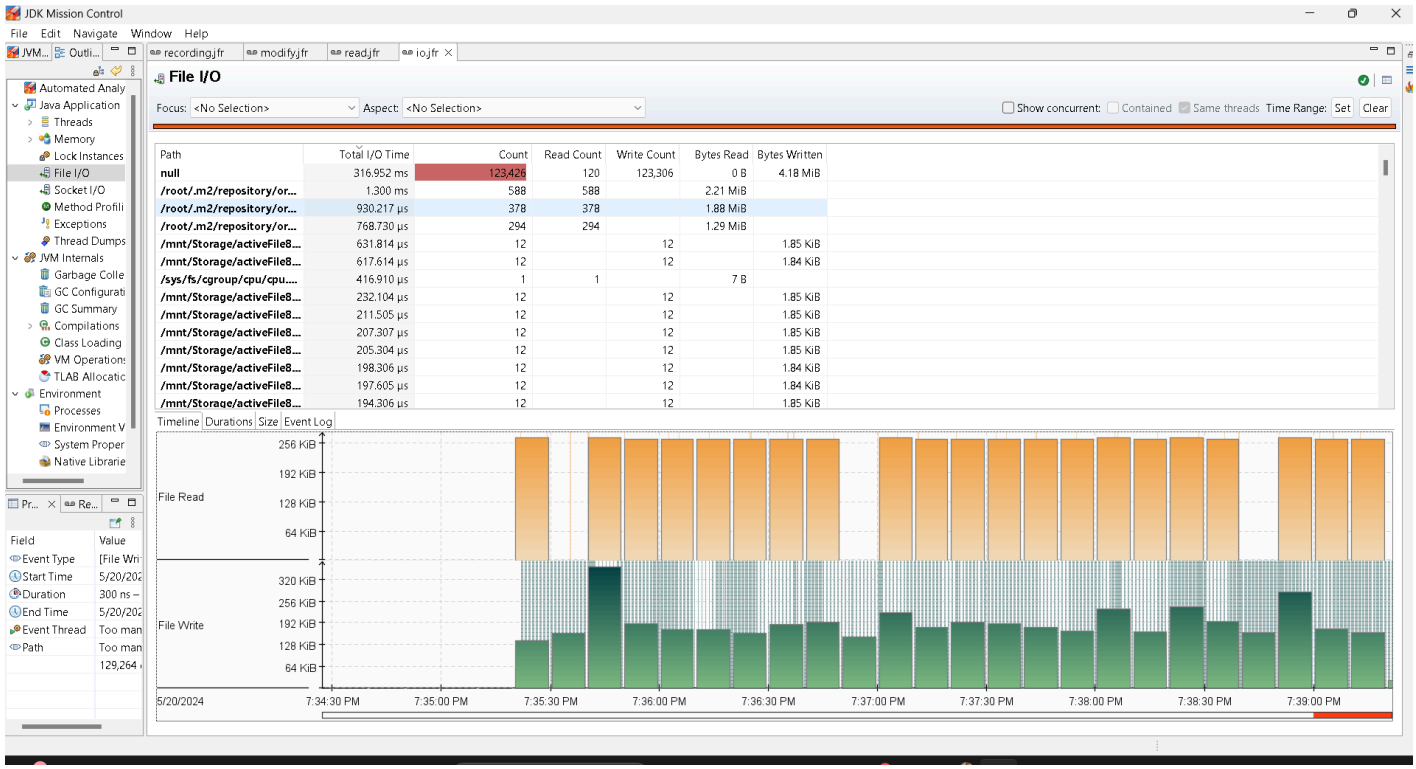


- GC pauses count
- GC maximum pause duration





- List of I/O operations



Docker & Kubernetes files

- K8s yaml file
    - Station 1

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: station1
  labels:
    app: station1
spec:
  replicas: 1
  selector:
    matchLabels:
      app: station1
  template:
    metadata:
      labels:
        app: station1
    spec:
      containers:
        - name: station1
          image: abdelaziz89/weather-station:v2.0
          ports:
            - containerPort: 8080
          env:
            - name: StationId
              value: "1"
            - name: latitude
              value: "31.2018"
            - name: longitude
              value: "29.9158"
```

- Kafka Processor

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: kafkaprocessor
  labels:
    app: kafkaprocessor
spec:
  replicas: 1
  selector:
    matchLabels:
      app: kafkaprocessor
  template:
    metadata:
      labels:
        app: kafkaprocessor
    spec:
      containers:
        - name: kafkaprocessor
          image: abdelaziz89/kafkaprocessor:v3.0
          ports:
            - containerPort: 8080
```

- Central Server

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: centralserver
  labels:
    app: centralserver
spec:
  replicas: 1
  selector:
    matchLabels:
      app: centralserver
  template:
    metadata:
      labels:
        app: centralserver
    spec:
      containers:
        - name: newserver
          image: abdelaziz89/central-server:v21.0
          ports:
            - containerPort: 8080
          volumeMounts:
            - mountPath: "/mnt/Volume"
              name: parquet
            - mountPath: "/mnt/Storage"
              name: parquet
      volumes:
        - name: parquet
          persistentVolumeClaim:
            claimName: serverstorage
```

- Persistent Volume

```yaml
apiVersion: v1
kind: PersistentVolume
metadata:
  name: storage
spec:
  storageClassName: manual
  capacity:
    storage: 2Gi
  accessModes:
    - ReadWriteOnce
  hostPath:
    path: "/data"
```

- Persistent Volume Claim

```yaml
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: serverstorage
spec:
  accessModes:
    - ReadWriteOnce
  storageClassName: manual
  resources:
    requests:
      storage: 2Gi
  volumeName: storage
```

- kibana

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: kibana-deployment
  labels:
    app: kibana
spec:
  replicas: 1
  selector:
    matchLabels:
      app: kibana
  template:
    metadata:
      labels:
        app: kibana
    spec:
      containers:
      - name: kibana
        image: kibana:8.4.0
        ports:
        - containerPort: 5601
        env:
        - name: ELASTICSEARCH_HOSTS
          value: http://elasticsearch-service:9200
---
apiVersion: v1
kind: Service
metadata:
  name: kibana-service
  labels:
    app: kibana
spec:
  ports:
  - port: 5601
    targetPort: 5601
  selector:
    app: kibana
```

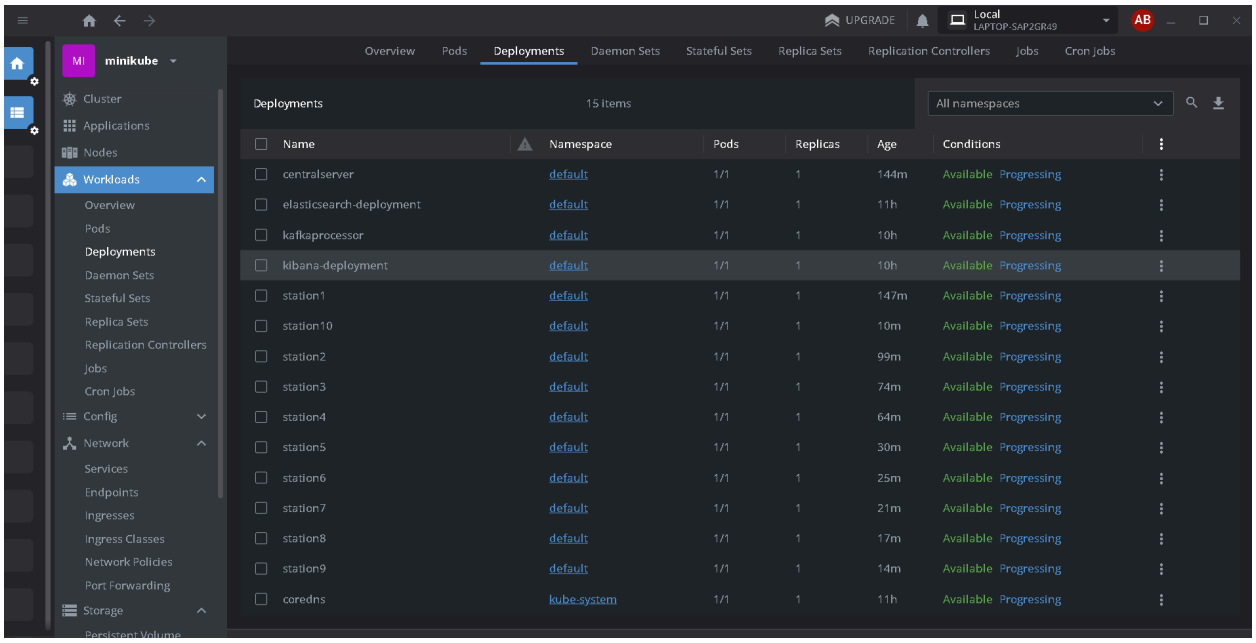
- Elastic Search

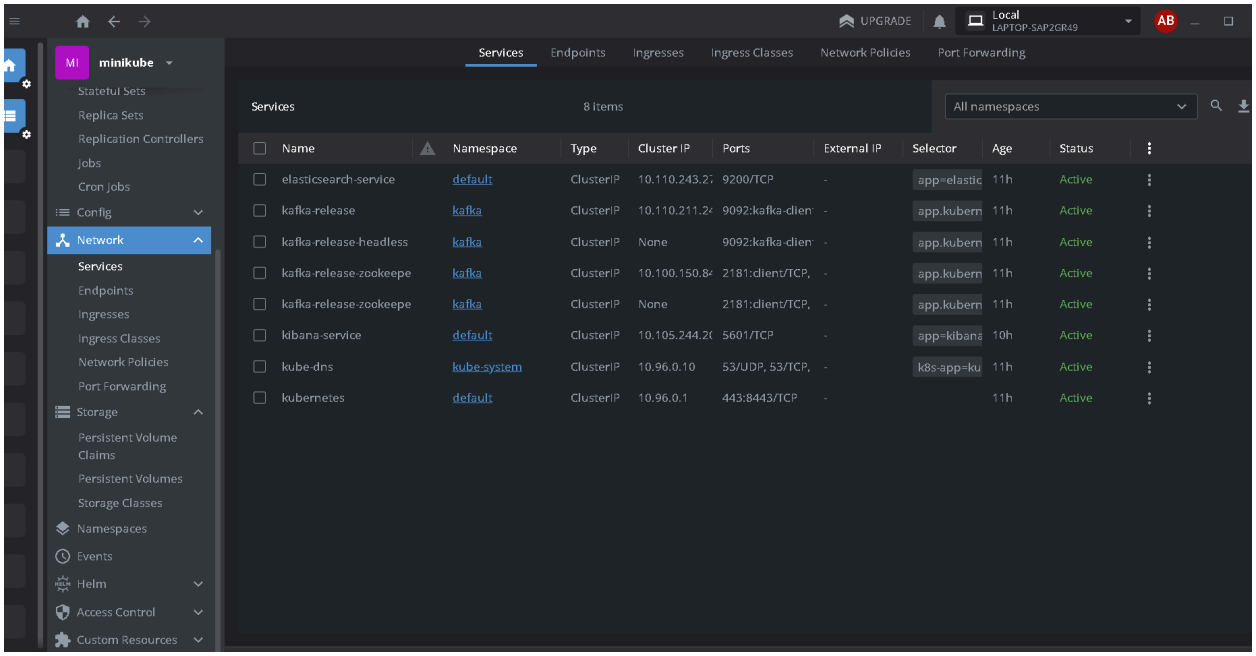
```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: elasticsearch-deployment
  labels:
    app: elasticsearch
spec:
  replicas: 1
  selector:
    matchLabels:
      app: elasticsearch
  template:
    metadata:
      labels:
        app: elasticsearch
    spec:
      containers:
      - name: elasticsearch
        image: elasticsearch:8.4.0
        ports:
        - containerPort: 9200
        env:
        - name: xpack.security.enabled
          value: "false"
        - name: discovery.type
          value: single-node
        - name: ES_JAVA_OPTS
          value: "-Xms512m -Xmx512m"
---
apiVersion: v1
kind: Service
metadata:
  name: elasticsearch-service
  labels:
    app: elasticsearch
spec:
  ports:
  - port: 9200
    targetPort: 9200
  selector:
    app: elasticsearch
```

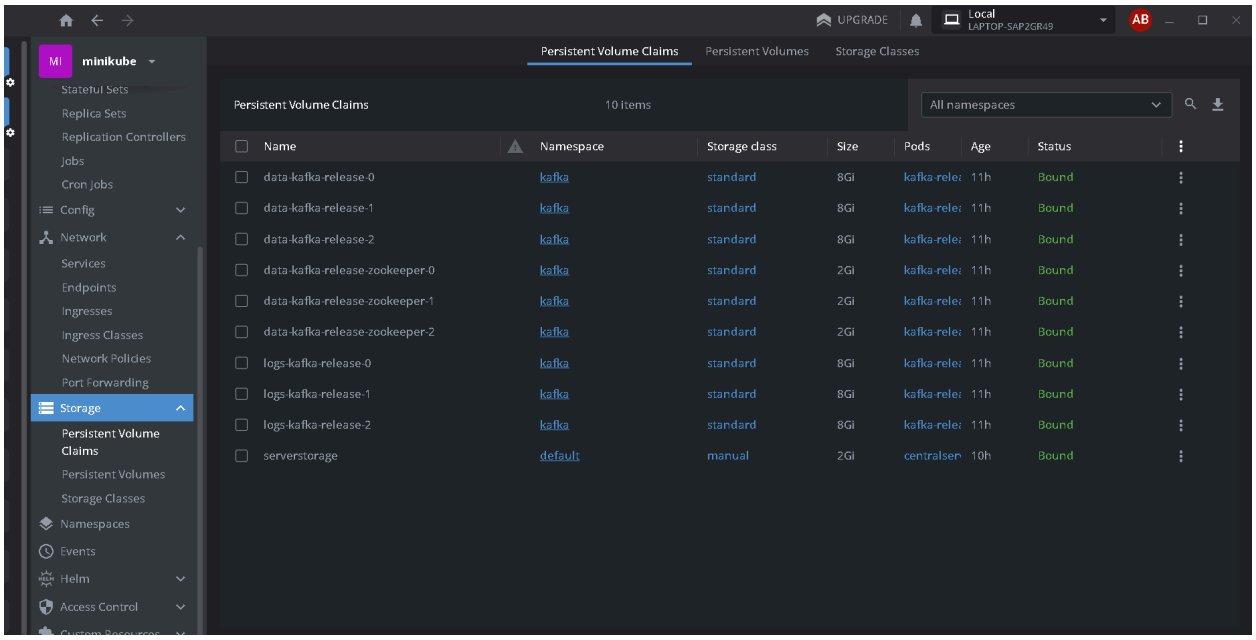
- Weather Station Dockerfile
- Central Server Dockerfile

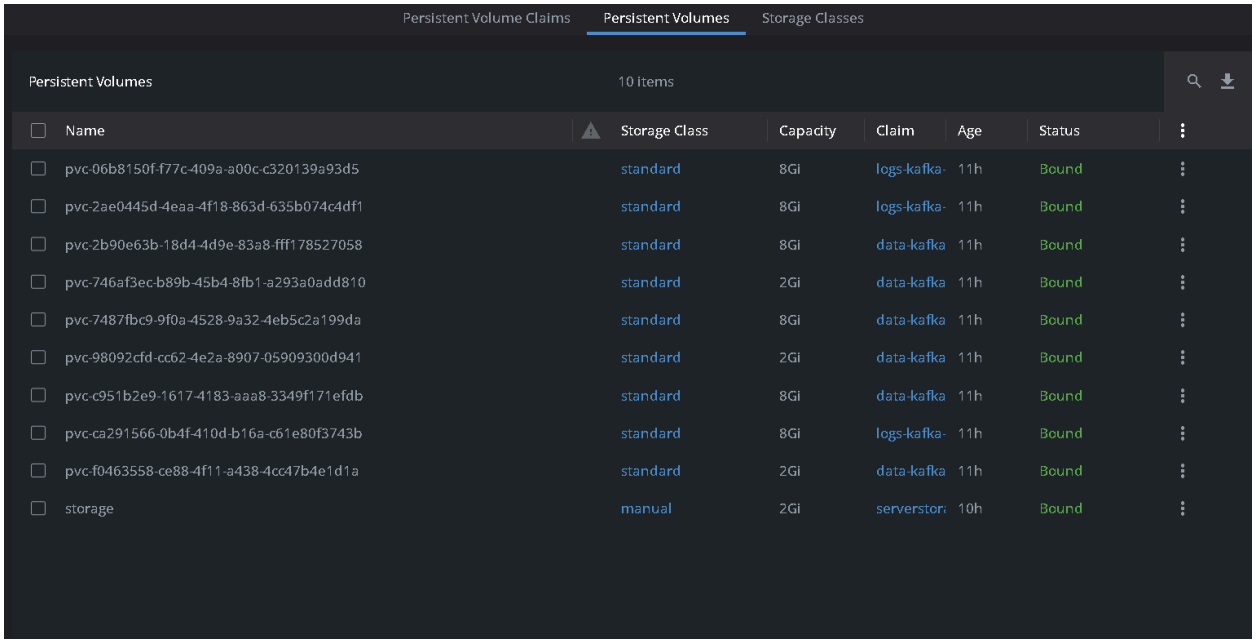
```
FROM maven:3.9.2-eclipse-temurin-17-alpine

WORKDIR /CentralServer
EXPOSE 8080
# Copy the Maven project files into the Docker container
COPY src ./src
COPY pom.xml .

# Copy the Avro schema file into the Docker container
COPY src/main/java/BaseCentralStation/weather_record.avsc
/CentralServer/src/main/java/BaseCentralStation/weather_record.avsc

# Build the Maven project inside the Docker container
RUN mvn clean install

# Specify the command to run your Java application
CMD ["mvn", "exec:java"]
```

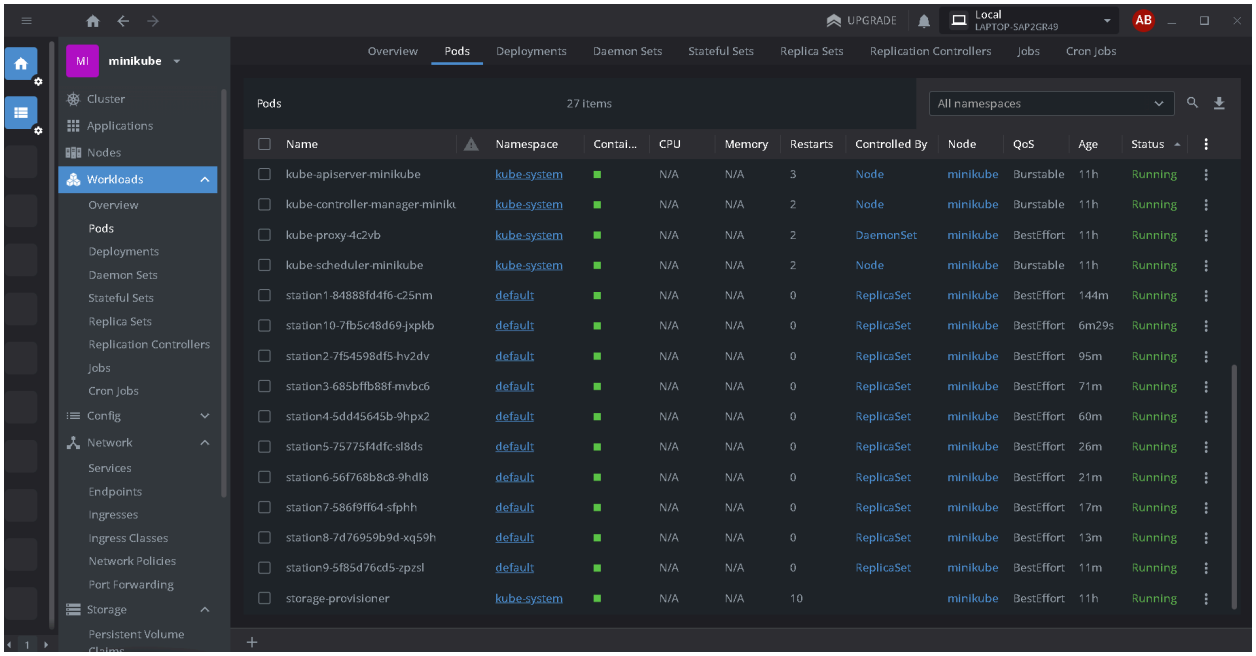
Kubernetes GUI






UPGRADE
Local
LAPTOP-SAP2GR49
AB
MI
minikube
Stateful Sets
Replica Sets
Replication Controllers
Jobs
Cron Jobs
Config
Network
Services
Endpoints
Ingresses
Ingress Classes
Network Policies
Port Forwarding
Storage
Persistent Volume Claims
Persistent Volumes
Storage Classes
Namespaces
Events
Helm
Access Control
Persistent Volume Claims
Persistent Volumes
Storage Classes
Persistent Volume Claims
10 items
All namespaces
Name
Namespace
Storage class
Size
Pods
Age
Status
data-kafka-release-0 kafka standard 8Gi kafka-relea 11h Bound
data-kafka-release-1 kafka standard 8Gi kafka-relea 11h Bound
data-kafka-release-2 kafka standard 8Gi kafka-relea 11h Bound
data-kafka-release-zookeeper-0 kafka standard 2Gi kafka-relea 11h Bound
data-kafka-release-zookeeper-1 kafka standard 2Gi kafka-relea 11h Bound
data-kafka-release-zookeeper-2 kafka standard 2Gi kafka-relea 11h Bound
logs-kafka-release-0 kafka standard 8Gi kafka-relea 11h Bound
logs-kafka-release-1 kafka standard 8Gi kafka-relea 11h Bound
logs-kafka-release-2 kafka standard 8Gi kafka-relea 11h Bound
serverstorage default manual 2Gi centralser 10h Bound

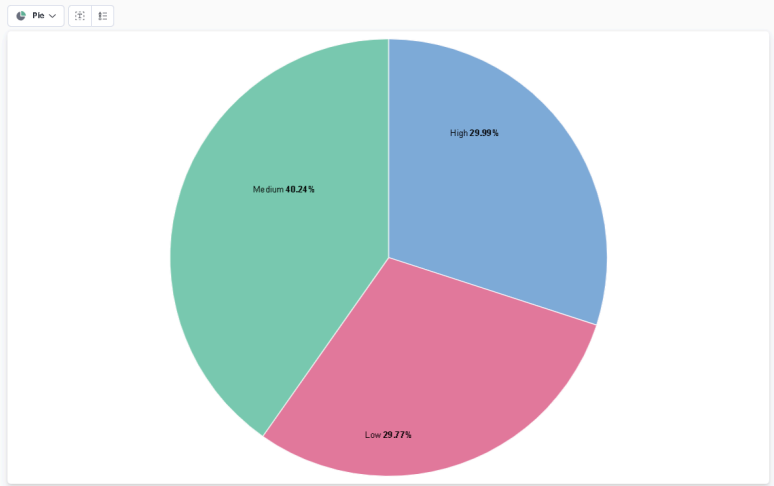

Persistent Volume Claims
Persistent Volumes
Storage Classes
Persistent Volumes
10 items
Name
Storage Class
Capacity
Claim
Age
Status
pvc-06b8150f-f77c-409a-a00c-c320139a93d5 standard 8Gi logs-kafka- 11h Bound
pvc-2ae0445d-4eaa-4f18-863d-635b074c4df1 standard 8Gi logs-kafka- 11h Bound
pvc-2b90e63b-18d4-4d9e-83a8-fff178527058 standard 8Gi data-kafka 11h Bound
pvc-746af3ec-b89b-45b4-8fb1-a293a0add810 standard 2Gi data-kafka 11h Bound
pvc-7487fbc9-9f0a-4528-9a32-4eb5c2a199da standard 8Gi data-kafka 11h Bound
pvc-98092cfd-cc62-4e2a-8907-05909300d941 standard 2Gi data-kafka 11h Bound
pvc-c951b2e9-1617-4183-aaa8-3349f171efdb standard 8Gi data-kafka 11h Bound
pvc-ca291566-0b4f-410d-b16a-c61e80f3743b standard 8Gi logs-kafka- 11h Bound
pvc-f0463558-ce88-4f11-a438-4cc47b4e1d1a standard 2Gi data-kafka 11h Bound
storage manual 2Gi serverstor 10h Bound



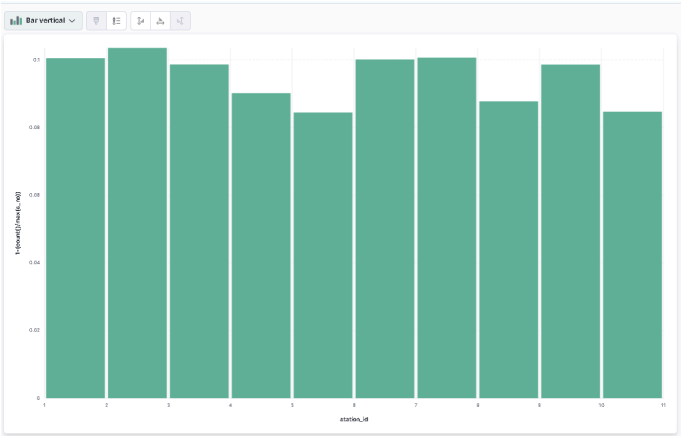

```
Downloaded from central: https://repo.maven.apache.org/maven2/org/ow2/asm/asm-tree/9.6/asm-tree-9.6.jar (52 kB at 22 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/codehaus/plexus/plexus-utils/4.0.0/plexus-utils-4.0.0.jar (192 kB at 66 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/ow2/asm/asm/9.6/asm-9.6.jar (124 kB at 44 kB/s)
log4j:WARN No appenders could be found for logger (org.apache.kafka.clients.producer.ProducerConfig).
log4j:WARN Please initialize the log4j system properly.
log4j:WARN See http://logging.apache.org/log4j/1.2/faq.html#noconfig for more info.
Raining detected: {"station_id":1,"s_no":21,"weather":{"temperature":74,"humidity":71,"wind_speed":16},"status_timestamp":1716224400,"battery_status":"Mediu
m"}
Raining detected: {"station_id":1,"s_no":22,"weather":{"temperature":73,"humidity":75,"wind_speed":15},"status_timestamp":1716228000,"battery_status":"Mediu
m"}
Raining detected: {"station_id":1,"s_no":23,"weather":{"temperature":73,"humidity":78,"wind_speed":16},"status_timestamp":1716231600,"battery_status":"Mediu
m"}
Raining detected: {"station_id":1,"s_no":24,"weather":{"temperature":72,"humidity":79,"wind_speed":16},"status_timestamp":1716235200,"battery_status":"Low"}
Raining detected: {"station_id":10,"s_no":24,"weather":{"temperature":74,"humidity":76,"wind_speed":8},"status_timestamp":1716235200,"battery_status":"Low"}
Raining detected: {"station_id":1,"s_no":45,"weather":{"temperature":74,"humidity":71,"wind_speed":16},"status_timestamp":1716224400,"battery_status":"High"
}
Raining detected: {"station_id":1,"s_no":46,"weather":{"temperature":73,"humidity":75,"wind_speed":15},"status_timestamp":1716228000,"battery_status":"Low"}
Raining detected: {"station_id":1,"s_no":47,"weather":{"temperature":73,"humidity":78,"wind_speed":16},"status_timestamp":1716231600,"battery_status":"Mediu
m"}
Raining detected: {"station_id":1,"s_no":48,"weather":{"temperature":72,"humidity":79,"wind_speed":16},"status_timestamp":1716235200,"battery_status":"High"
}
Raining detected: {"station_id":10,"s_no":48,"weather":{"temperature":74,"humidity":76,"wind_speed":8},"status_timestamp":1716235200,"battery_status":"Low"}
Raining detected: {"station_id":1,"s_no":69,"weather":{"temperature":74,"humidity":71,"wind_speed":16},"status_timestamp":1716224400,"battery_status":"Mediu
m"}
Raining detected: {"station_id":1,"s_no":70,"weather":{"temperature":73,"humidity":75,"wind_speed":15},"status_timestamp":1716228000,"battery_status":"High"
}
Raining detected: {"station_id":1,"s_no":71,"weather":{"temperature":73,"humidity":78,"wind_speed":16},"status_timestamp":1716231600,"battery_status":"High"
}
Raining detected: {"station_id":1,"s_no":72,"weather":{"temperature":72,"humidity":79,"wind_speed":16},"status_timestamp":1716235200,"battery_status":"Mediu
m"}
Raining detected: {"station_id":10,"s_no":72,"weather":{"temperature":74,"humidity":76,"wind_speed":8},"status_timestamp":1716235200,"battery_status":"High"
}
Raining detected: {"station_id":1,"s_no":93,"weather":{"temperature":74,"humidity":71,"wind_speed":16},"status_timestamp":1716224400,"battery_status":"Low"}
Raining detected: {"station_id":1,"s_no":94,"weather":{"temperature":73,"humidity":75,"wind_speed":15},"status_timestamp":1716228000,"battery_status":"Mediu
m"}
Raining detected: {"station_id":1,"s_no":95,"weather":{"temperature":73,"humidity":78,"wind_speed":16},"status_timestamp":1716231600,"battery_status":"High"
}
Raining detected: {"station_id":1,"s_no":96,"weather":{"temperature":72,"humidity":79,"wind_speed":16},"status_timestamp":1716235200,"battery_status":"High"
}
Raining detected: {"station_id":10,"s_no":96,"weather":{"temperature":74,"humidity":76,"wind_speed":8},"status_timestamp":1716235200,"battery_status":"Low"}
Raining detected: {"station_id":1,"s_no":117,"weather":{"temperature":74,"humidity":71,"wind_speed":16},"status_timestamp":1716224400,"battery_status":"High
```

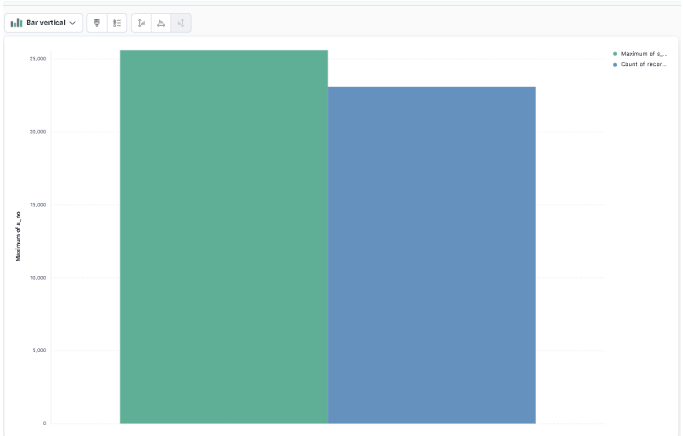
Kibana visualization confirming

- Battery status distribution (30% low - 40% medium - 30% high)



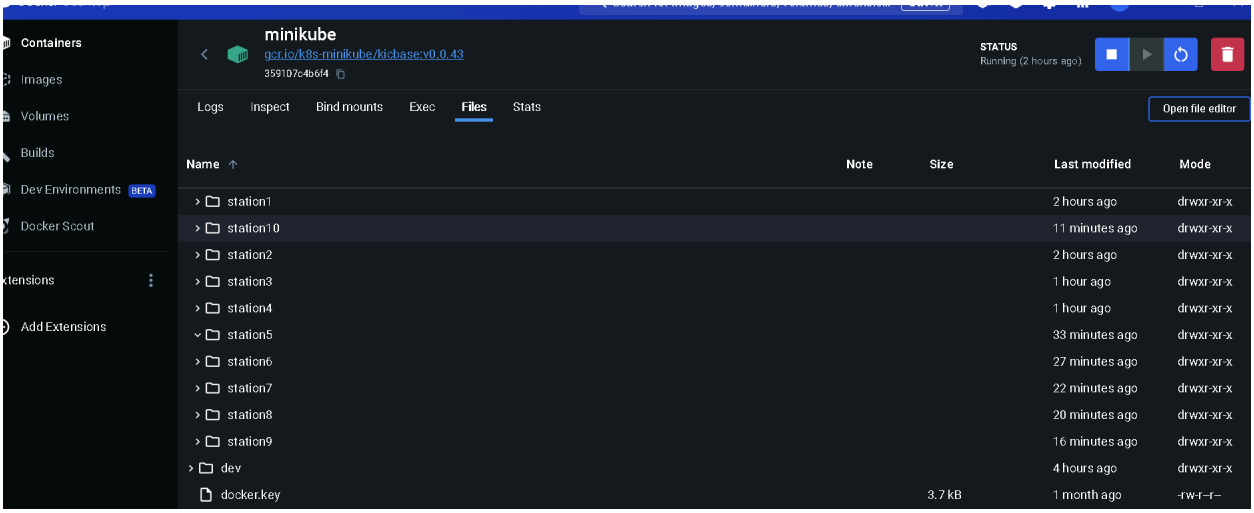
- 10% dropped messages





Sample Parquet File

- Parquet File Directory



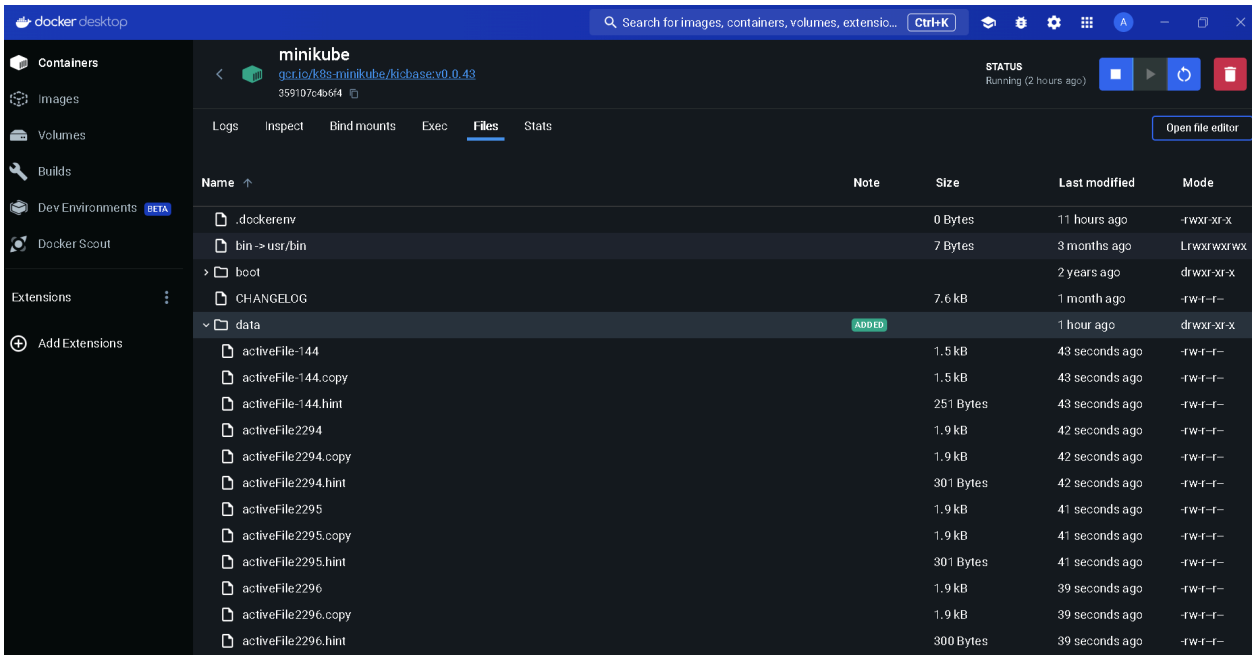
- ## Parquet File Sample

  Project_1716203944681.parquet

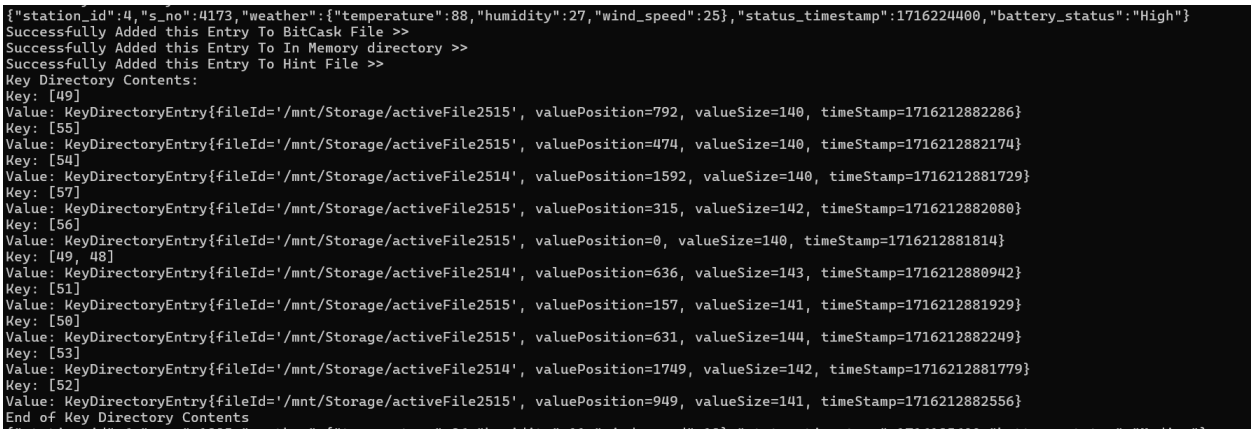
station_id - i32	s_no - i32	weather - struct[3]	status_timestamp - i64	battery_status - str
1	1	{82,44,25}	1716152400	Medium
1	2	{82,45,26}	1716156000	Low
1	3	{82,45,26}	1716159600	Medium
1	4	{82,44,24}	1716163200	Low
1	5	{82,48,22}	1716166800	Low
1	6	{82,49,20}	1716170400	Medium
1	7	{82,49,16}	1716174000	Medium
1	8	{82,53,18}	1716177600	High
1	9	{80,57,23}	1716181200	Low
1	10	{78,51,15}	1716184800	Medium
1	11	{79,49,13}	1716188400	Medium
1	12	{83,39,11}	1716192000	High
1	13	{89,23,16}	1716195600	Medium
1	14	{81,43,23}	1716199200	Medium
1	15	{78,52,23}	1716202800	Medium
1	16	{78,51,21}	1716206400	High
1	17	{78,52,20}	1716210000	High
1	18	{77,55,19}	1716213600	Medium

Sample BitCask Riak LSM directory

- Files in directory



- BitCask In Memory Key Value Directory Sample



BitCask File Entries

- Active File Entry

```
public class BitCaskEntry   {
    private long timeStamp;
    private int keyLength;
    private int valueLength ;
    private byte [] key;
    private byte [] value;
}
```

- Hint File Entry

```
public class HintFileEntry {
    private long tstamp;
    private int keySize ;
    private int valueSize ;
    private long valuePosition;
    private byte[] key;
}
```

- Key Directory In Memory Entry

```
public class KeyDirectoryEntry {
    private String fileId ;
    private long valuePosition;
    private int valueSize ;
    private long timeStamp;
}
```

Bonus

- Open-Meteo

```java
public class OpenMeteoAPI {
    private static final String OPEN_METEO_API_URL = "https://api.open-meteo.com/weather";

    private final KafkaProducer<String, String> kafkaProducer;
    private final Random random = new Random();
    public OpenMeteoAPI(KafkaProducer<String, String> kafkaProducer) {
        this.kafkaProducer = kafkaProducer;
    }
    public int seq =1 ;

    public void fetchAndPublishWeatherData(int StationId,double latitude, double longitude) {
        try {
            while(true)
            {
                // Construct API URL with latitude, longitude, and other parameters
                String apiUrl = "https://api.open-meteo.com/v1/forecast?latitude=" +
                        latitude + "&longitude=" + longitude +
                        "&hourly=relativehumidity_2m,windspeed_80m,temperature_80m&" +
                        "current_weather=true&temperature_unit=fahrenheit&timeformat=unixtime" +
                        "&forecast_days=1&timezone=Africa%2FCairo";

                // Create HTTP connection
                URL url = new URL(apiUrl);
                HttpURLConnection connection = (HttpURLConnection) url.openConnection();
                connection.setRequestMethod("GET");
                // Check response code
                int responseCode = connection.getResponseCode();
```

- Enterprise Integration Patterns

1. Adapter

```
// Parse JSON response
JSONObject jsonObject = new JSONObject(response.toString());
System.out.println(jsonObject);
// Extract hourly data object
JSONObject hourlyData = jsonObject.getJSONObject("hourly");

// Extract arrays for each parameter
JSONArray relativeHumidityArray = hourlyData.getJSONArray("relativehumidity_2m");
JSONArray windSpeedArray = hourlyData.getJSONArray("windspeed_80m");
JSONArray temperatureArray = hourlyData.getJSONArray("temperature_80m");
JSONArray timeArray = hourlyData.getJSONArray("time");
```

2. Aggregator

```
 if (recordCount >= BATCH_SIZE) {
      // Write the batch to a Parquet file
      try {
          writeRecordsToParquet(stationId, stationRecords, avroSchema);
     } catch (IOException e) {
          e.printStackTrace();
          // Optionally handle the exception, e.g., skip the batch
     } finally {
          stationRecords.clear();
          recordCount = 0;
    }
}
```

3. Pipe and Filter

```java
    private static boolean isRaining(String value) {
        try {
            // Parse the JSON message
            JSONObject json = new JSONObject(value);

            // Extract humidity from the weather data
            JSONObject weatherData = json.getJSONObject("weather");
            int humidity = weatherData.getInt("humidity");

            // Check if humidity is higher than 70%
            if (humidity > 70) {
                // Print value if it's raining
                System.out.println("Raining detected: " + value);
                String message = "Raining detected: " + value;
                producer.send(new ProducerRecord<>("processor", message));
                return true;
            }
        } catch (Exception e) {
            // Handle parsing or any other exceptions
            e.printStackTrace();
        }
        return false;
    }
```

4. Envelope Wrapper

```java
kafkaProducer.send(new ProducerRecord<>("Project", Integer.toString(StationId),kafkaMessage.toString()));
kafkaProducer.flush();
```

```java
for (ConsumerRecord<String, String> record : records) {
                GenericRecord avroRecord = createAvroRecord(record.value(), avroSchema);
                recordCount++;
                System.out.println(record.value());
                int stationId = (int) avroRecord.get("station_id");
                List<GenericRecord> stationRecords = stationRecordsMap.get(stationId);
                stationRecords.add(avroRecord);
```

5. Polling Consumer

```
ConsumerRecords<String, String> records = consumer.poll(Duration.ofMillis(100));
```

6. Enricher Content

```
  // Method to build weather message with specified attributes
  public static JSONObject buildWeatherMessage(int stationId, int sNo, int relativeHumidity, int windSpeed, int
temperature, long time) {

        JSONObject kafkaMessage = new JSONObject();
        kafkaMessage.put("station_id", stationId);
        kafkaMessage.put("s_no", sNo);
        kafkaMessage.put("battery_status", getRandomBatteryStatus());
        kafkaMessage.put("status_timestamp", time);
        JSONObject weatherData = new JSONObject();
        weatherData.put("humidity", relativeHumidity);
        weatherData.put("temperature", temperature);
        weatherData.put("wind_speed", windSpeed);
        kafkaMessage.put("weather", weatherData);
        return kafkaMessage;
    }
```

Github Repo

https://github.com/Abdelaziz25/Weather-Stations-Monitoring