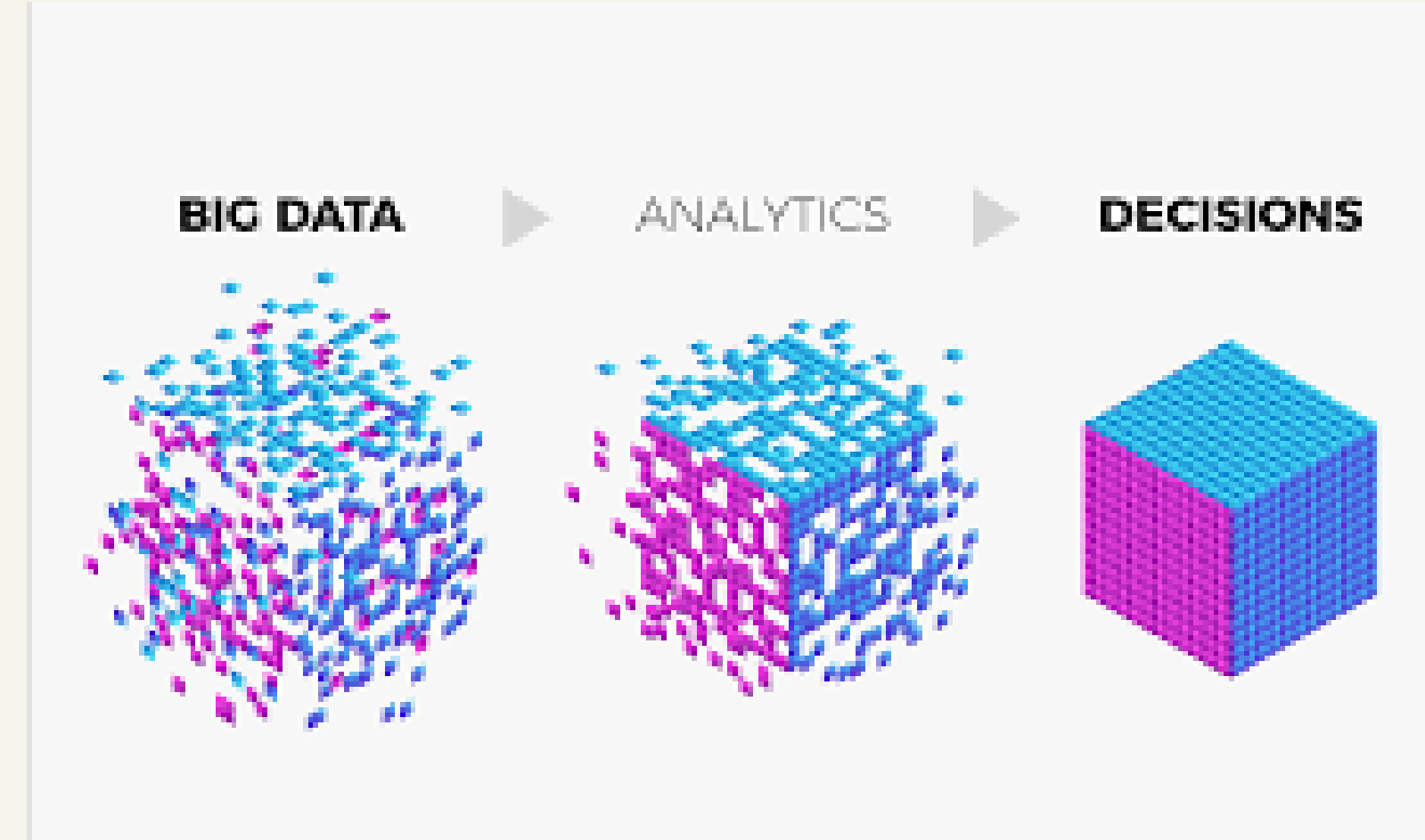


The background features three vertical stripes on the left side in shades of pink, blue, and beige. On the right side, there is a decorative pattern of small pink dots arranged in a grid-like fashion, with some dots missing to create a sparse effect.

# **BIG DATA ANALYTICS GROUP PROJECT**

**GROUP 14**

**Under the guidance of : Dr. Sonali Agarwal**



**GROUP 14**

**Mohit Kumar - IIT2020220**

**Janhavi Bawaskar - IIT2020263**

**Pratyaksh Singh - IIB2020015**

**Harsh Kant Bhatnagar - IEC2020113**

# OVERVIEW

- Problem Statement
- Introduction
- Literary Review
- Overview of requirements
- Database
- Project Hypothesis
- Methodology
- Implementation
- Result
- Timeline
- Future Scope

# PROBLEM STATEMENT

Design an InfluxDB with Apache superset model for real time smart building operations monitoring.



# INTRODUCTION

3

- **Real-time data stream processing is important in environmental monitoring systems to enable timely detection of anomalies and events.**
- **A CEP engine is a software system that processes data in a distributed manner and is used for datastream processing in environmental monitoring.**
- **This work presents a complex event processing (CEP) engine for detecting anomalies in real time**



# LITERARY REVIEW

<u>Author name</u>	<u>Year</u>	<u>Paper Title</u>	<u>Application domain</u>	<u>Achieved Performance</u>
Alejandro Buchmann, TU Darmstadt, Boris Koldehofe, Universität Stuttgart	2019	Complex Event Processing	IoT Smart Cities CEP	Studies focusing on performance metrics, scalability, and optimization techniques in CEP systems. Comparative analysis of various CEP systems
Alexander Y. Suna, Zhi Zhonga, Hoonyoung Jeongb, Qian Yanga	2019	Building complex event processing capability for intelligent environmental monitoring	Real-time Event Detection Complex event processing Predictive Analytics for Environmental Patterns	Complex event processing (CEP) engine for detecting anomalies in real time, and demonstrates it using a series of real monitoring data from the geological carbon sequestration domain.
Nithin Krishna Reghunathan	2020	Real-Time Streaming in Big Data: Kafka and Spark with SingleStore	Real-Time Streaming Big Data Technologies Stream Processing Pipelines Real-Time Analytics	Investigates the integration and capabilities of Kafka and Spark in conjunction with SingleStore to facilitate real-time streaming and processing within the realm of big data applications.
Haruna Isah, Tariq Abughofa, Sazia Mahfuz, Dharmitha Ajerla, Farhana Zulkernine, Shahzad Khan	Jan 2021	A Survey of Distributed Data Stream Processing Frameworks	Distributed Data Stream Processing Framework Analysis and Evaluation	Synthesizing and presenting existing information regarding the strengths, weaknesses, and characteristics of different distributed data stream processing frameworks based on available literature and evaluations performed by others

# REQUIREMENTS

7

## InfluxDB

- Real-time insights from any time series data with a single, purpose-built database.
- retrieval, and processing of large volumes of time-series data

## Apache Kafka

Apache Kafka is an open-source distributed streaming system used for stream processing, real-time data pipelines, and data integration at scale.

## Apache Spark

Spark is used for processing vast amounts of data in a distributed and parallel manner. It can handle both batch and real-time data processing

## Grafana

Spark is used for processing vast amounts of data in a distributed and parallel manner. It can handle both batch and real-time data processing



# DATASET

- The dataset which we will be using is Occupancy Detection Data Set UCI.
- The "Occupancy Detection Data Set" from the UCI Machine Learning Repository is a dataset that contains information related to occupancy detection in an office building. The dataset is typically used for binary classification tasks where the goal is to determine whether a room is occupied or not based on various sensor measurements.
- The attributes include: Date and Time, Temperature, Relative Humidity, Light, CO2, Humidity Ratio, Occupancy.



```
1  "date","Temperature","Humidity","Light","CO2","HumidityRatio","Occupancy"
2  "1","2015-02-04 17:51:00",23.18,27.272,426,721.25,0.00479298817650529,1
3  "2","2015-02-04 17:51:59",23.15,27.2675,429.5,714,0.00478344094931065,1
4  "3","2015-02-04 17:53:00",23.15,27.245,426,713.5,0.00477946352442199,1
5  "4","2015-02-04 17:54:00",23.15,27.2,426,708.25,0.00477150882608175,1
6  "5","2015-02-04 17:55:00",23.1,27.2,426,704.5,0.00475699293331518,1
7  "6","2015-02-04 17:55:59",23.1,27.2,419,701,0.00475699293331518,1
8  "7","2015-02-04 17:57:00",23.1,27.2,419,701.666666666667,0.00475699293331518,1
9  "8","2015-02-04 17:57:59",23.1,27.2,419,699,0.00475699293331518,1
10 "9","2015-02-04 17:58:59",23.1,27.2,419,689.333333333333,0.00475699293331518,1
11 "10","2015-02-04 18:00:00",23.075,27.175,419,688,0.00474535071966655,1
12 "11","2015-02-04 18:01:00",23.075,27.15,419,690.25,0.00474095189694268,1
13 "12","2015-02-04 18:02:00",23.1,27.1,419,691,0.00473937073052061,1
14 "13","2015-02-04 18:03:00",23.1,27.1666666666667,419,683.5,0.00475111875560951,1
15 "14","2015-02-04 18:04:00",23.05,27.15,419,687.5,0.0047337317970825,1
16 "15","2015-02-04 18:04:59",23,27.125,419,686,0.00471494214590473,1
17 "16","2015-02-04 18:06:00",23,27.125,418.5,680.5,0.00471494214590473,1
18 "17","2015-02-04 18:07:00",23,27.2,0,681.5,0.00472807794966877,0
19 "18","2015-02-04 18:08:00",22.945,27.29,0,685,0.00472795137178073,0
20 "19","2015-02-04 18:08:59",22.945,27.39,0,685,0.0047454083970941,0
21 "20","2015-02-04 18:10:00",22.89,27.39,0,689,0.00472950615591001,0
22 "21","2015-02-04 18:10:59",22.89,27.39,0,689.5,0.00472950615591001,0
23 "22","2015-02-04 18:11:59",22.89,27.39,0,689,0.00472950615591001,0
24 "23","2015-02-04 18:13:00",22.89,27.445,0,691,0.00473907551474663,0
25 "24","2015-02-04 18:14:00",22.89,27.5,0,688,0.00474864516581148,0
26 "25","2015-02-04 18:15:00",22.89,27.5,0,689.5,0.00474864516581148,0
```

# METHODOLOGY

## ● Phase 1

Create a data delivery system, that continuously produces the data. For this the same data frame is iterated again and again.

## ● Phase 2

Create a data ingestion system that delivers the data to spark, then processes the data and ingests it into influx db

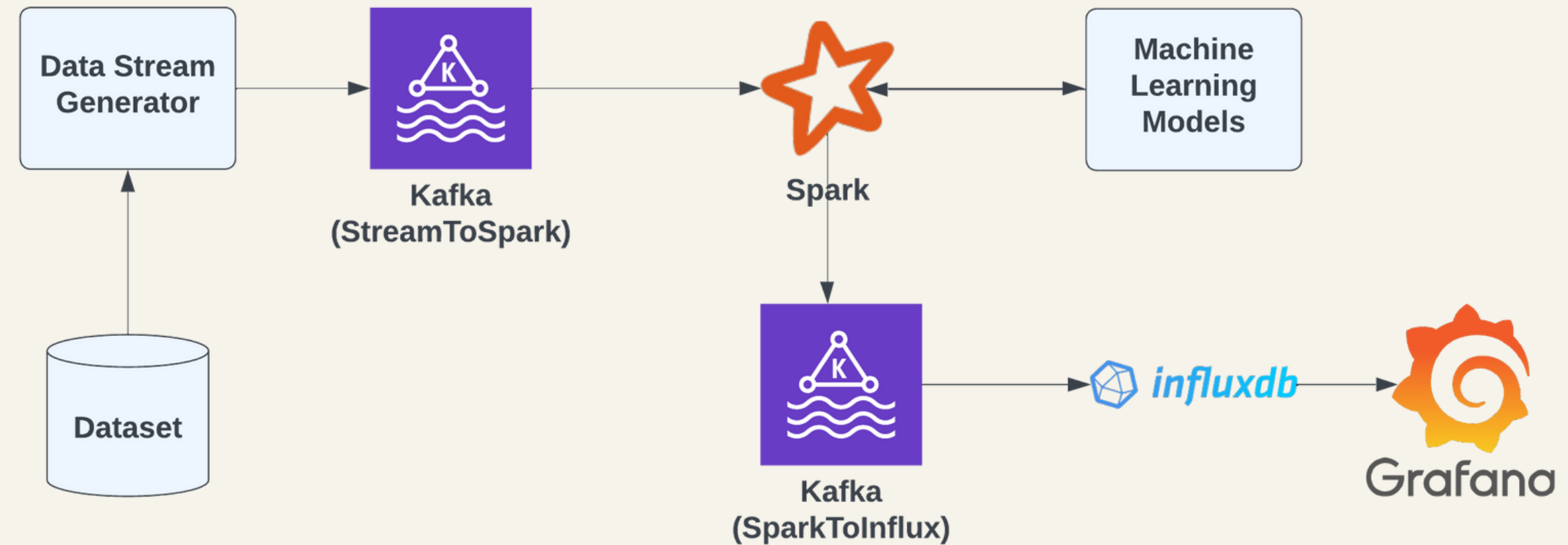
## ● Phase 3

Create a connection from influx DB to Grafana so that data can be queried and graphs can be analyzed.

## ● Phase 4

Create alerts in grafana which will help to analyse any abnormalities in the building data

# SYSTEM ARCHITECTURE



**Docker used for running kafka, zookeeper, spark, influx-db and grafana in containers**

**Docker-Compose used to configure the services**

**Kafka acting as a message broker between data source & spark and spark & influxDB**

**Spark used for real time processing and prediction**

**Multiple spark brokers running simultaneously (master-slave architecture) for efficient computation**

```
spark-master:
  image: bitnami/spark:latest
  command: bin/spark-class org.apache.spark.deploy.master.Master
  ports:
    - "9090:8080"
    - "7077:7077"
  volumes:
    - ./models:/opt/bitnami/spark/model
  networks:
    - bda_project

spark-worker-1:
  image: bitnami/spark:latest
  command: bin/spark-class org.apache.spark.deploy.worker.Worker spark://spark-master:7077
  depends_on:
    - spark-master
  environment:
    SPARK_MODE: "worker"
    SPARK_WORKER_CORES: "2"
    SPARK_WORKER_MEMORY: "2g"
    SPARK_MASTER_URL: "spark://spark-master:7077"
  volumes:
    - ./models:/opt/bitnami/spark/model
  networks:
    - bda_project

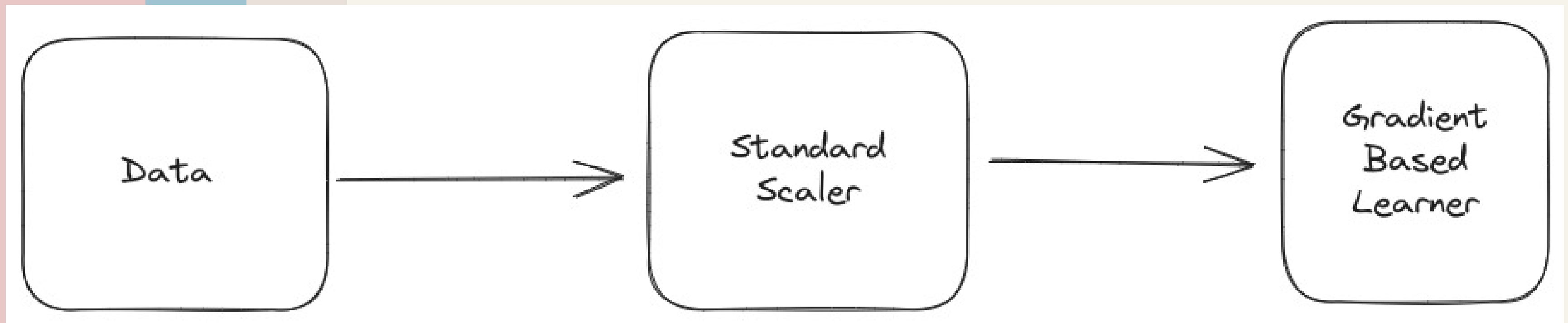
spark-worker-2:
  image: bitnami/spark:latest
```

Master Slave Configuration

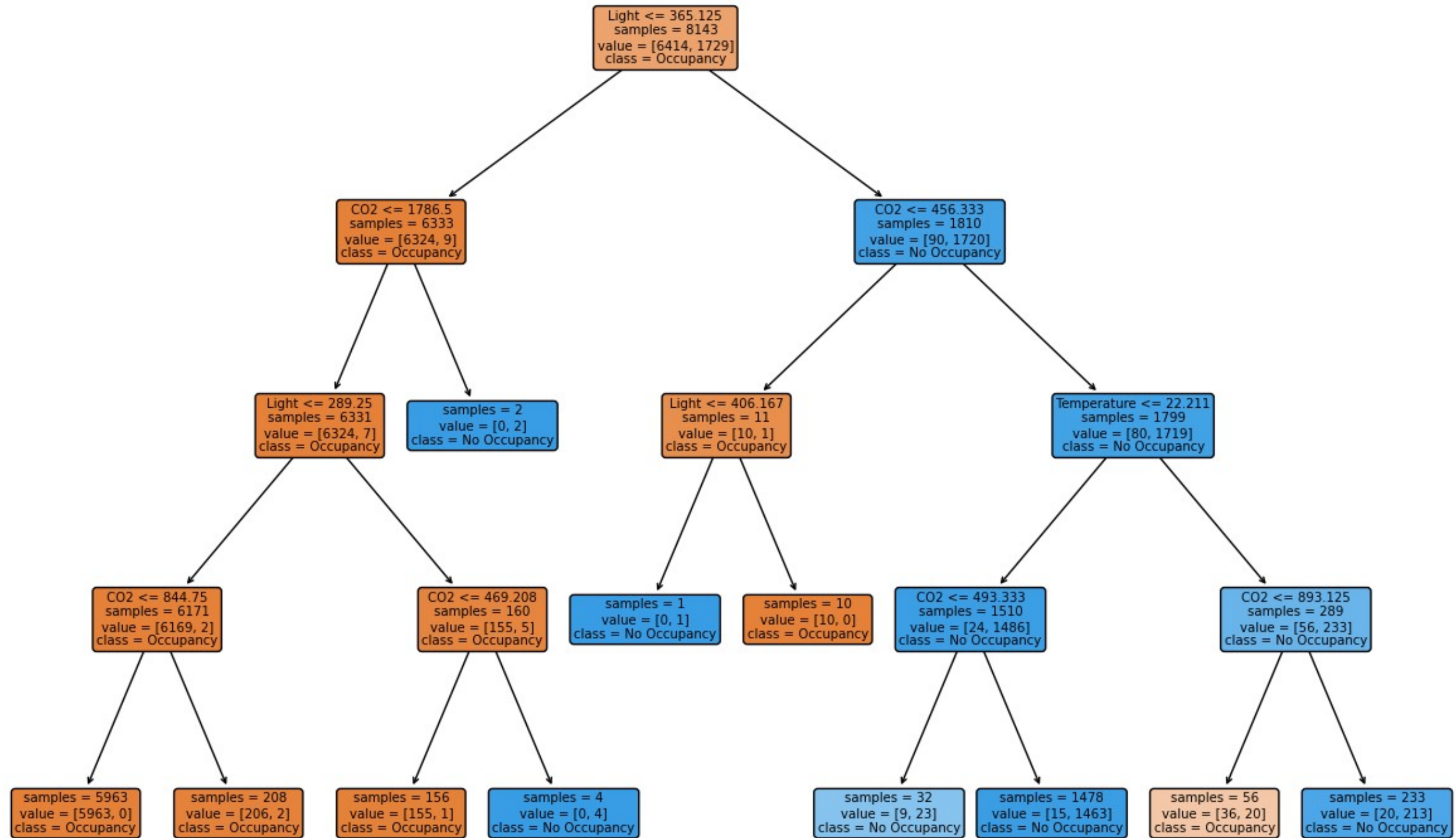
**InfluxDB is a high speed read and write noSQL database used in time series applications**

**Grafana is a data visualization tool used for creating dashboards, alerts etc**

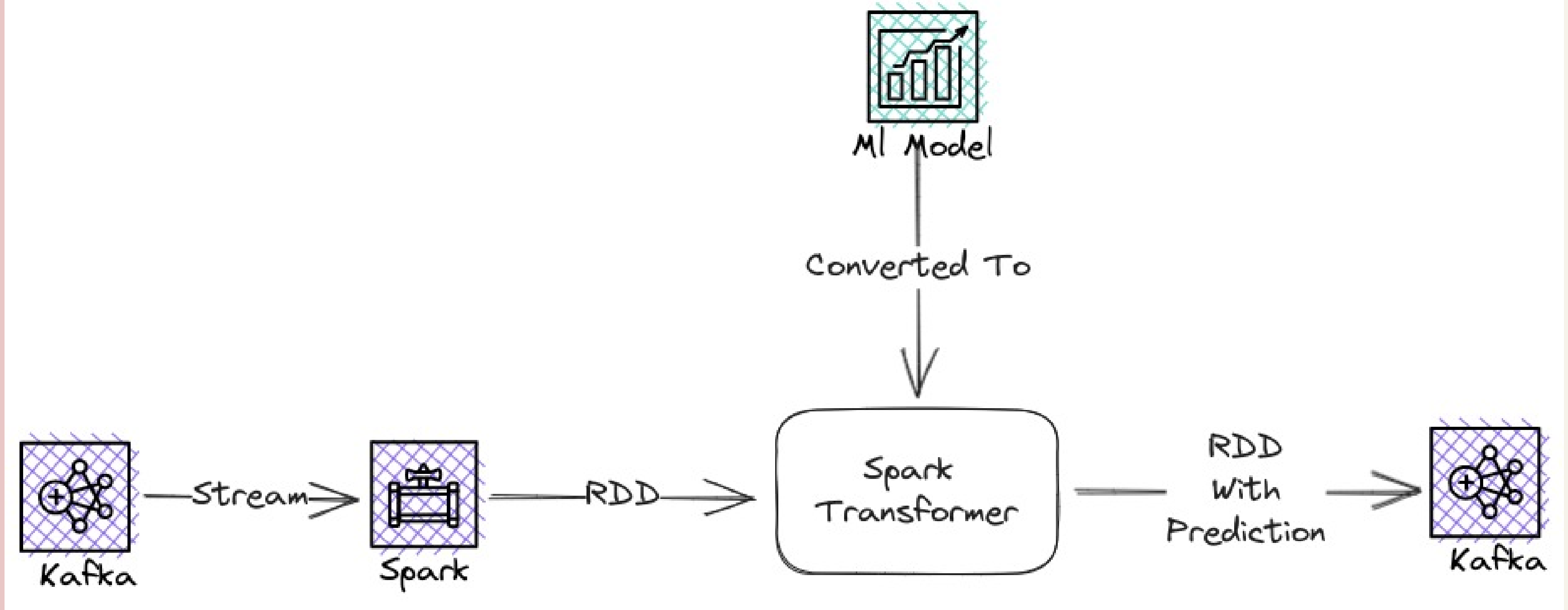
# GRADIENT BASED LEARNER



# TREE BASED LEARNER



# PREDICTION ARCHITECTURE





# LOCAL OUTLIER FACTOR

- The Local Outlier Factor (LOF) is a machine learning algorithm used for anomaly detection and outlier detection in datasets.
- It was introduced by Markus M. Breunig, Hans-Peter Kriegel, ~~Raymond T. Ng~~, and Jörg Sander in a 2000 research paper.
- LOF is particularly useful for finding anomalies in high-dimensional datasets and those with complex structures.

# GRAFANA

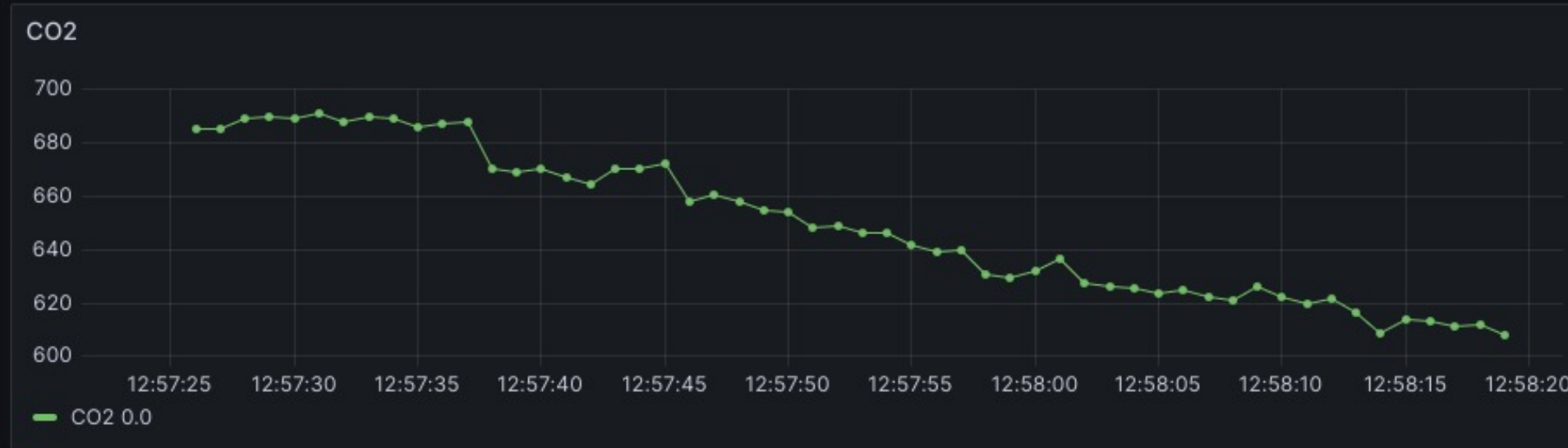
- **Visualization is instrumental for assisting knowledge discovery and decision support.**
- **Our problem statement mentions the use of Apache Superset but due to certain limitations at hand, we have chosen to move forward with Grafana as our visualization instrument.**
- **It has a rich collection of data visualization tools, easy-to-use interface for uploading and transforming data, and seamless integration with commonly used data stores.**

# GRAFANA

- Visualized data on the basis of various features.
- The features include:
  1. CO2 level
  2. Humidity
  3. Temperature
  4. Light
- Flux query language is used to display the dashboards.
- Added alerts when CO2 goes above 300. Here, 300 is the threshold value.

# SETTING UP QUERY

- Flux query language



Query 1 Transform data 0 Alert 0

Data source bda

Query options MD = auto = 940 Interval = 50ms

Query inspector

A (bda)

```
1 from(bucket: "bda")
2   |> range(start: -1m)
3   |> filter(fn: (r) => r["_measurement"] == "bda_project")
4   |> filter(fn: (r) => r["Occupancy"] == "0.0" or r["Occupancy"] == "1.0")
5   |> filter(fn: (r) => r["_field"] == "CO2")
```

Flux language syntax

Sample query

Help

# SETTING UP ALERT

State

Labels

Created

> Pending alertname CO2 Alert grafana\_folder BDA 2023-12-04 13:05:00

Query & Results

A

bda

now-10m to now

intervalMs: 1000  
maxDataPoints: 43200  
query: |-  
 from(bucket: "bda")  
 |> range(start: -10s)  
 |> filter(fn: (r) => r.\_measurement == "bda\_project" and r.\_field == "CO2")  
 |> mean()  
 |> yield(name: "average\_CO2")  
refId: A

2023-12-04 13:05:03View in Explore

Table

Series 1441.375

B

Reduce

Function

Last

Input

A

Mode

Strict

Series 1

441.375

C

Threshold

Alert condition

Input

B

Is above 300

Series 1

1

Firing

# ALERTS FIRING

## Alert rules

Rules that determine whether an alert will fire

Search by data sources ⓘ

All data sources

State

FiringNormalPending

Rule type

AlertRecording

Health

OkNo DataError

Search ⓘ

Q Search

View as

GroupedListState

1 rule1 firing

+ New alert rule

More

Grafana

▼ BDA > BDA

1 firing | 30s |

State	Name	Health	Summary	Next evaluation	Actions
> Firing for 17s	CO2 Alert	ok		in a few seconds	More

Mimir / Cortex / Loki

There are no Prometheus or Loki data sources configured.



# FINAL DASHBOARD





The background features three vertical stripes on the left side in shades of pink, blue, and beige. On the right side, there are two rectangular areas filled with a grid of small, light pink dots.

# THANK YOU

**Presented By : Group 14**