



UNIVERSITÀ
di CAMERINO

Data Analytics on Elasticsearch for IoT

University of Camerino
School of Science and Technology
Data Analytics Course



Students:

Giulia Morelli
Jessica Piccioni



Supervisor:

Prof. Massimo Callisto

TABLE OF CONTENTS

01

Project Description

Project description,
Objectives and Critical Points

02

Technologies

Technologies used in the
project

03

Implementation

Implementation phases of the project

04

Results and Future works

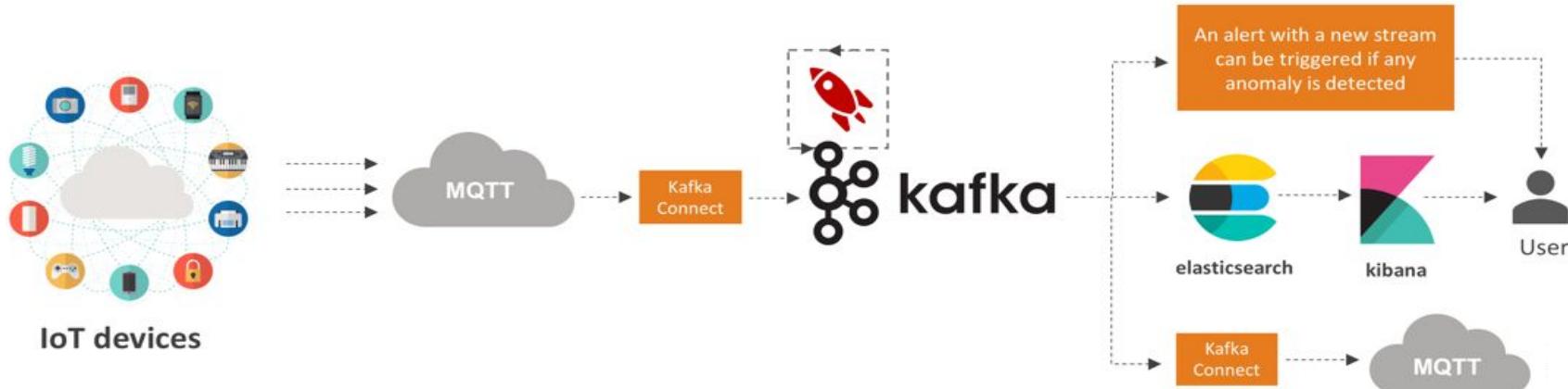
Results obtained and
future works

PROJECT DESCRIPTION



PROJECT DESCRIPTION

The project aims to monitor IoT data coming from the sensors installed in the Computer Science department and classrooms of Camerino University. Data must be ingested with Kafka, sent in real time to Elasticsearch and displayed on a visualization dashboard defined in Kibana.



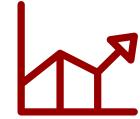
OBJECTIVES



The **main objective** of the project is to create an IoT data visualization dashboard based on Elasticsearch



Take MQTT data in real time from the Filippetti sensors and send it to a Kafka broker



Filter IoT data in order to make it understandable in Elasticsearch



Create a dashboard that displays the environmental data of the Filippetti sensors in Kibana.

CRITICAL POINTS



The **critical points** of the project were the following:



Taking sensor data using Filebeat with Logstash or Confluent Platform with Kafka?

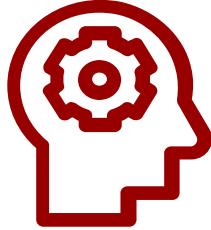


Filtering data on Confluent Platform or on Elasticsearch?



What kind of data should be displayed on the dashboard?

OUR SOLUTION



The **solutions** of the critical points of the project were the following:

Confluent Platform is the most efficient solution.
Filebeat does not support WSS protocol

Filter the data with KsqlDB in Confluent Platform

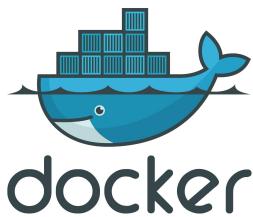
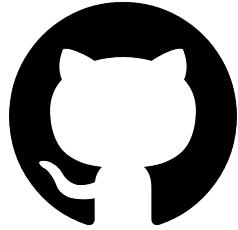
Creating a dashboard showing the most significant data gathered in the classroom

TECHNOLOGIES



TECHNOLOGIES

- **Ubuntu 21.04 64 bit**
- **Java 11.0**
- **PuTTY 0.74**
- **Docker 20.10**
- **Elasticsearch 7.10**
- **Kibana 7.10**
- **KsqlDb 0.17**
- **Github**
- **Confluent Platform 6.2**



TECHNOLOGIES

Confluent Platform allows us to make a streaming analytics and filtering data that comes from Flippetti broker.



Elasticsearch allows us to save the data that comes from the kafka broker.



Kibana is used to create the dashboard for visualizing the data.



IMPLEMENTATION



IMPLEMENTATION PHASES



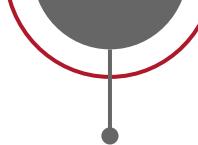
Set Elasticsearch and Kibana environment



Send the data from Kafka Broker to Elasticsearch and create Kibana Dashboard



Start Confluent Platform, create the topics, add the connector and filter mqtt data



SETTING ELASTIC AND KIBANA ENVIRONMENT



- 1) Start Elasticsearch and Kibana with docker-compose

```
docker-compose up
```

- 2) Create an index in Elasticsearch

```
curl -X PUT http://hostname:9200/index_name
```

SETTING ELASTIC AND KIBANA ENVIRONMENT



- 3) Create a Pipeline in Elasticsearch to convert the date field from a string format into a data format

```
curl -X PUT
"hostname:9200/_ingest/pipeline/my-pipeline?pretty" -H
'Content-Type: application/json' -d'
{
  "description" : "...",
  "processors" : [
    {
      "date" : {
        "field" : "TZ",
        "formats" : ["date_optional_time"],
        "timezone" : "Europe/London"
      }
    }
  ]
}'
```

- 4) Set the pipeline as default pipeline for the index

```
curl -X PUT
"hostname:9200/index_name/_settings" -H
'Content-Type: application/json' -d'
{
  "settings": {
    "index.default_pipeline": "my-pipeline"
  }
}'
```

SETTING CONFLUENT PLATFORM ENVIRONMENT

2

- 1) Launch Confluent from any directory with the underneath command

```
confluent local services start
```

- 2) Once all services are [UP] we can proceed navigating to the Control Center web interface at <http://hostname:9021/> and select the cluster

The screenshot shows the Confluent Control Center home page. At the top, there's a dark header with the Confluent logo. Below it, a banner displays the number of healthy and unhealthy clusters. The main area has a search bar and a card for the single healthy cluster, which is named "controlcenter.cluster" and is labeled as "Running". The card provides an overview of the cluster's metrics.

Category	Value
Brokers	1
Partitions	321
Topics	61
Production	18.25KB/s
Consumption	14.01KB/s

SETTING CONFLUENT PLATFORM ENVIRONMENT

2

3) Install the source and sink connectors with the commands

```
confluent-hub install  
confluentinc/kafka-connect-mqtt:latest
```

```
confluent-hub install  
confluentinc/kafka-connect-elasticsearch:latest
```

4) Create the topic you need

Topics

Topic name	Status	Partitions	Production (last 5 mins)	Consumption (
default_ksql_processing_log	Healthy --	1	--	--
mqtt_data	Healthy --	1	--	--

SETTING CONFLUENT PLATFORM ENVIRONMENT



5) Add the Mqtt source connector to take the data in real time from the Filippetti broker and put it into the new topic

Add Connector

1. Setup connection —— 2. Test and verify

```
{  
  "name": "mqtt-source",  
  "config": {  
    "name": "mqtt-source",  
    "connector.class": "io.confluent.connect.mqtt.MqttSourceConnector",  
    "tasks.max": "1",  
    "mqtt.server.uri": "wss://cloud.smartplatform.io/mqtt",  
    "mqtt.username": "serviceuser@unicam.it",  
    "mqtt.password": "*****",  
    "kafka.topic": "mqtt",  
    "mqtt.topics": "/unicam.it/jz/device/snapshot/0000/edv/#"  
  }  
}
```

[Launch](#)

[Back](#)

[Download connector config file](#)

SETTING CONFLUENT PLATFORM ENVIRONMENT

2

- 6) Create and Write a Stream using KsqlDB to filter the mqtt data and create a new topic with the filtered data

ksqlDB

Editor Flow Streams Tables Persistent queries Settings

```
1 CREATE STREAM stream1(uuid varchar KEY, ref varchar, type varchar, m array<struct<tz varchar,
2 k varchar, v double, u varchar>>)
3 WITH (kafka_topic='mqtt', value_format='JSON_SR');
4
5 CREATE STREAM filtered_data AS
6 SELECT uuid, ref, type, EXPLODE(m)->tz AS tz, EXPLODE(m)->k AS measure, EXPLODE(m)->v AS value,
7 EXPLODE(m)->u AS unit
8 FROM stream1
9 EMIT CHANGES;
```

SEND DATA TO ELASTICSEARCH AND CREATE DASHBOARD



- 1) Add the Elasticsearch sink connector to send filtered data to Elasticsearch.

Add Connector

1. Setup connection —— 2. Test and verify

```
{  
  "name": "elastic-Connector",  
  "config": {  
    "value.converter.schema.registry.url": "http://localhost:8081",  
    "value.converter.schemas.enable": "true",  
    "key.converter.schema.registry.url": "http://localhost:8081",  
    "name": "elastic-Connector",  
    "connector.class": "io.confluent.connect.elasticsearch.ElasticsearchSinkConnector",  
    "key.converter": "org.apache.kafka.connect.storage.StringConverter",  
    "value.converter": "io.confluent.connect.json.JsonSchemaConverter",  
    "errors.tolerance": "all",  
    "topics": "FILTERED_DATA",  
    "connection.url": "http://192.168.178.36:9200",  
    "key.ignore": "true",  
    "behavior.on.malformed.documents": "ignore"  
  }  
}
```

[Launch](#)

[Back](#)

[Download connector config file](#)

SEND DATA TO ELASTICSEARCH AND CREATE DASHBOARD

3

2) Open Kibana at <http://hostname:5601> and create the index pattern

Stack Management -> Index Pattern -> Create index Pattern -> write the name with the same name of your index in which there are the sensors data -> select @timestamp as field.

The screenshot shows the 'Create index pattern' page within the Elastic Stack Management interface. The left sidebar lists various management sections: Ingest, Data, Alerts and Insights, and Kibana. The main content area is titled 'Create index pattern' and contains instructions about matching sources. A form for defining the index pattern is shown, with the 'Index pattern name' field containing 'index-name-*'. Below it, a note says 'Use an asterisk (*) to match multiple indices. Spaces and the characters \, /, ?, *, <, >, | are not allowed.' There is also an option to 'Include system and hidden indices' with a checked radio button. A note states 'Your index pattern can match any of your 12 sources.' At the bottom, three index patterns are listed: 'abc' (Index), 'exploded_base' (Index), and 'filtered_data' (Index). A 'Next step >' button is visible on the right.

SEND DATA TO ELASTICSEARCH AND CREATE DASHBOARD

3

3) After selecting the index pattern you can create your custom dashboard.

The image shows two screenshots of the Kibana interface. The top screenshot displays the 'Dashboards' page with a central card titled 'Create your first dashboard'. It includes a brief description: 'You can combine data views from any Kibana app into one dashboard and see everything in one place.' Below the description is a blue button labeled '+ Create new dashboard'. The bottom screenshot shows the 'Editing New Dashboard' screen. On the left, there's a sidebar with 'Search' and 'Add filter' options. The main area has a placeholder 'Add an existing or new object to this dashboard' and a 'Create new' button. To the right, a 'New Visualization' section is open, featuring a search bar and a grid of visualization icons. A title 'Select a visualization type' is at the top of the grid, followed by the text 'Start creating your visualization by selecting a type for that visualization.' Below this is a section titled 'Try Lens, our new, intuitive way to create visualizations.' with a 'Go to Lens' button. The grid contains 16 visualization types: Lens, Area, Controls, Data Table, Gauge, Goal, Heat Map, Horizontal Bar, Line, Maps, Markdown, Metric, Pie, TSVB, Tag Cloud, and Timeline.

RESULTS AND FUTURE WORK

FORWARD



SENSORS DATA VS FILTERED DATA IN ELASTICSEARCH

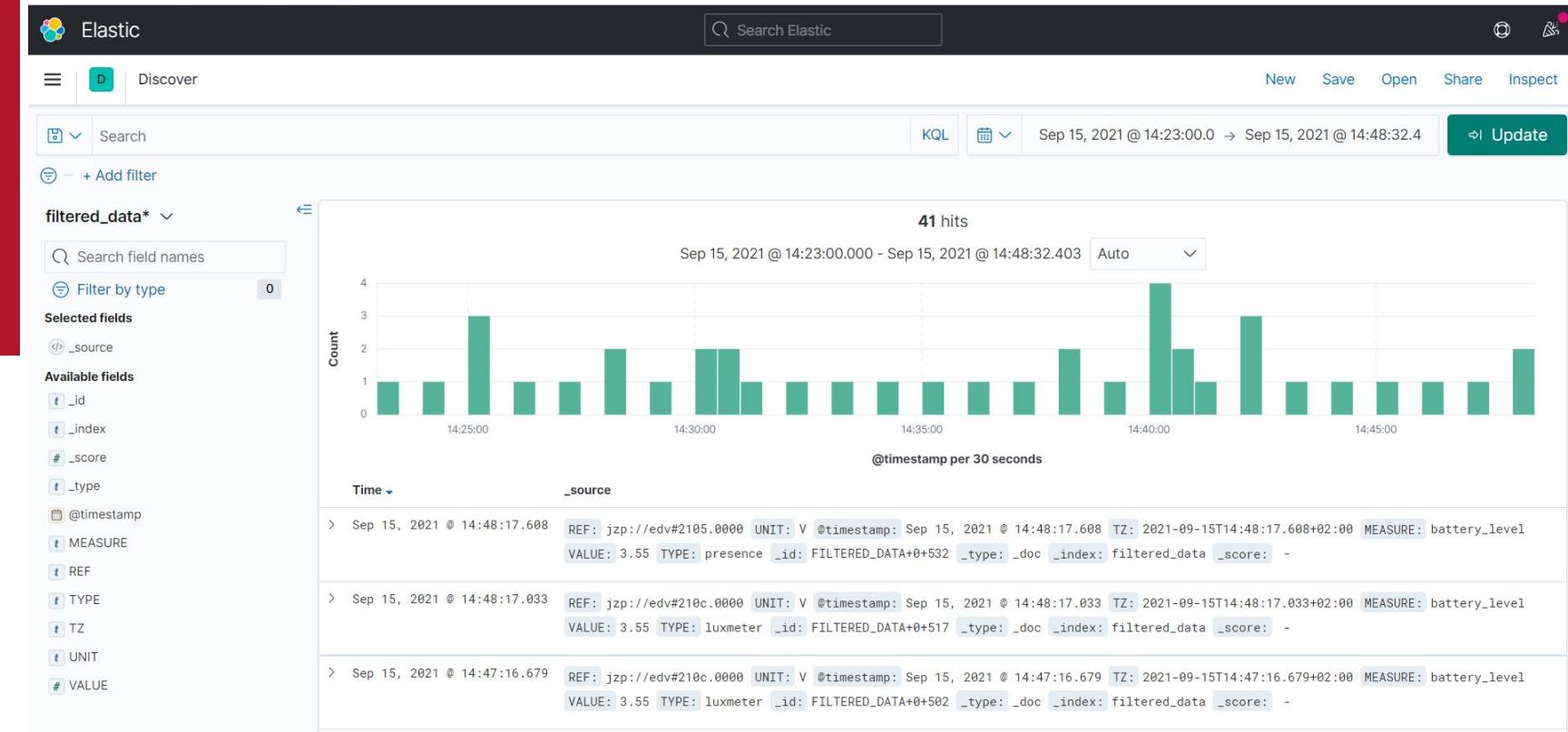
```
{  
  "t": timestamp in secondi  
  "tz": timestamp in HHMMDDtHHMMSS  
  "uuid": identificativo univoco del messaggio  
  "cuid":  
  "ref": "jzp://edv#0503.0000" identificativo fisico del device  
  "type": "presence" tipo di messaggio del sensore  
  "cat": "0610"  
  "sn": "integer(0, 255)",  
  "m": [  
    {  
      "t": "nowTimestamp()",  
      "tz": "now()",  
      "k": "device_temperature" tipo di misura: temperatura del device  
      "v": "double(0, 40)", valore della misura  
      "u": "C" unità di misura  
    },
```

200 - OK 128 ms

```
2  "took" : 1,  
3  "timed_out" : false,  
4  "shards" : {  
5    "total" : 1,  
6    "successful" : 1,  
7    "skipped" : 0,  
8    "failed" : 0  
9  },  
10 "hits" : {  
11   "total" : {  
12     "value" : 497,  
13     "relation" : "eq"  
14   },  
15   "max_score" : 1.0,  
16   "hits" : [  
17     {  
18       "_index" : "filtered_data",  
19       "_type" : "_doc",  
20       "_id" : "FILTERED_DATA+0+0",  
21       "score" : 1.0,  
22       "source" : {  
23         "REF" : "jzp://edv#210c.0000",  
24         "UNIT" : "V",  
25         "@timestamp" : "2021-09-15T14:23:09.219+02:00",  
26         "TZ" : "2021-09-15T14:23:09.219+02:00",  
27         "MEASURE" : "battery_level",  
28         "VALUE" : 3.55,  
29         "TYPE" : "luxmeter"  
30     }  
31   },  
32 ]  
33 }
```



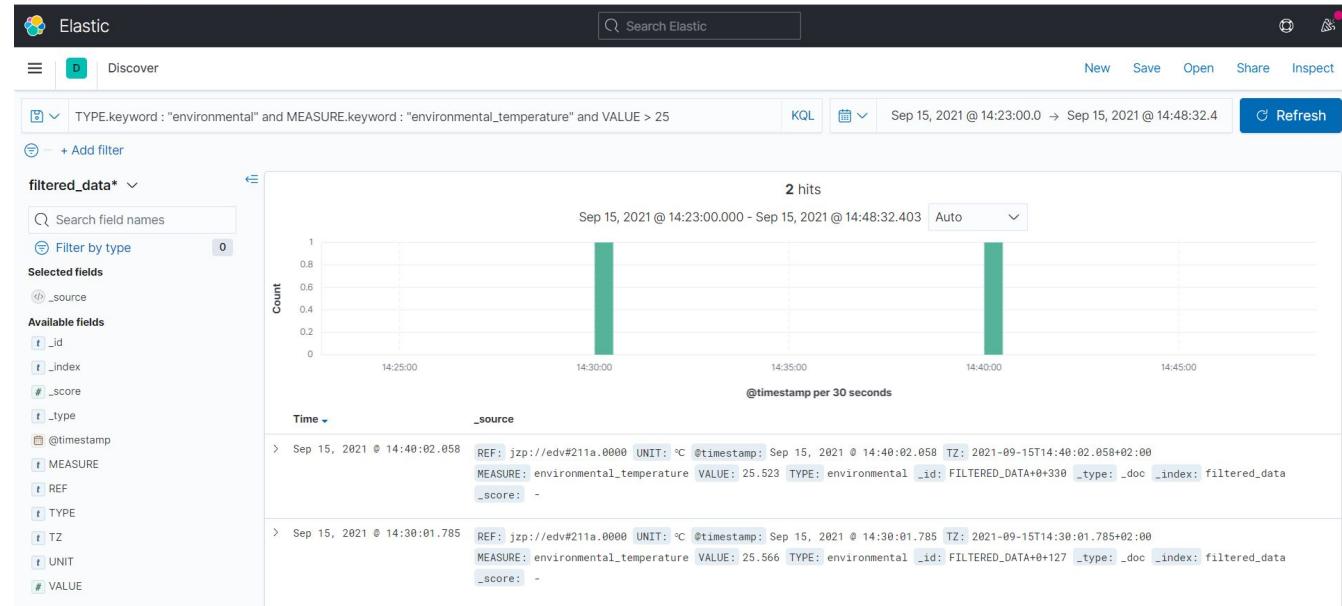
DISCOVER VIEW IN KIBANA



DISCOVER VIEW IN KIBANA

KQL QUERY 1

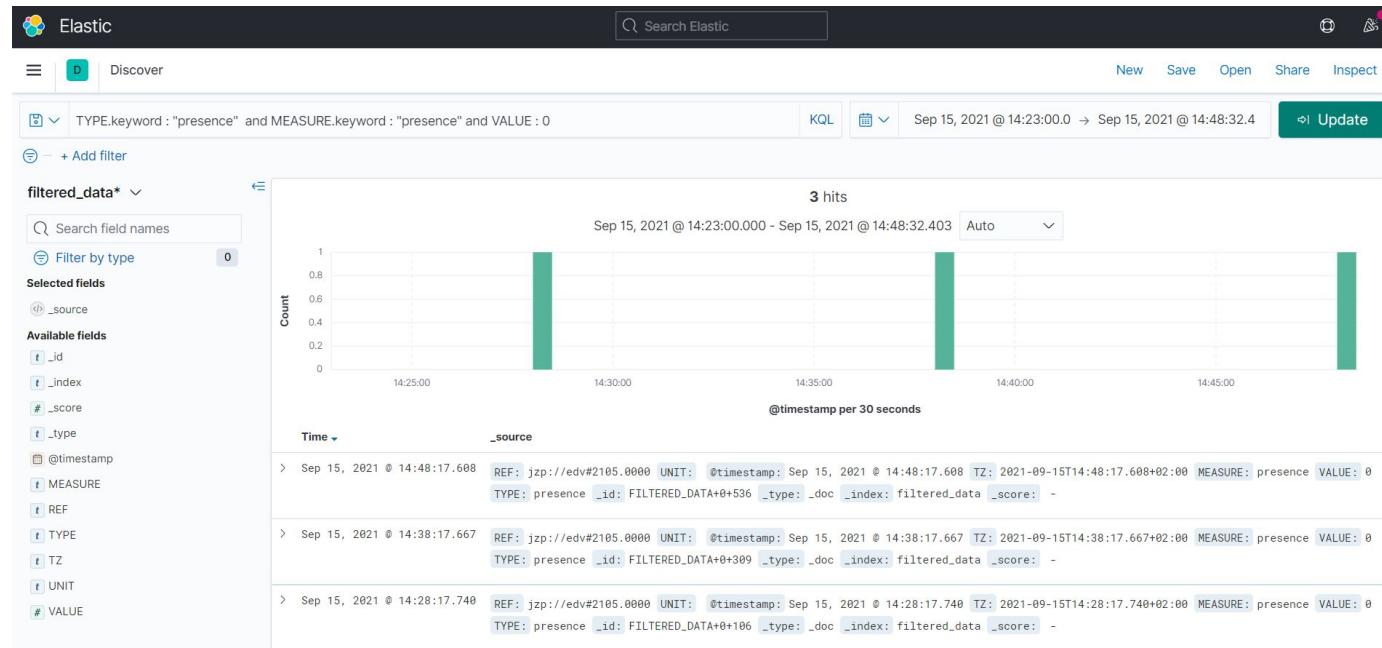
```
TYPE.keyword : "environmental" and  
MEASURE.keyword : "environmental_temperature"  
and VALUE > 25
```



DISCOVER VIEW IN KIBANA

KQL QUERY 2

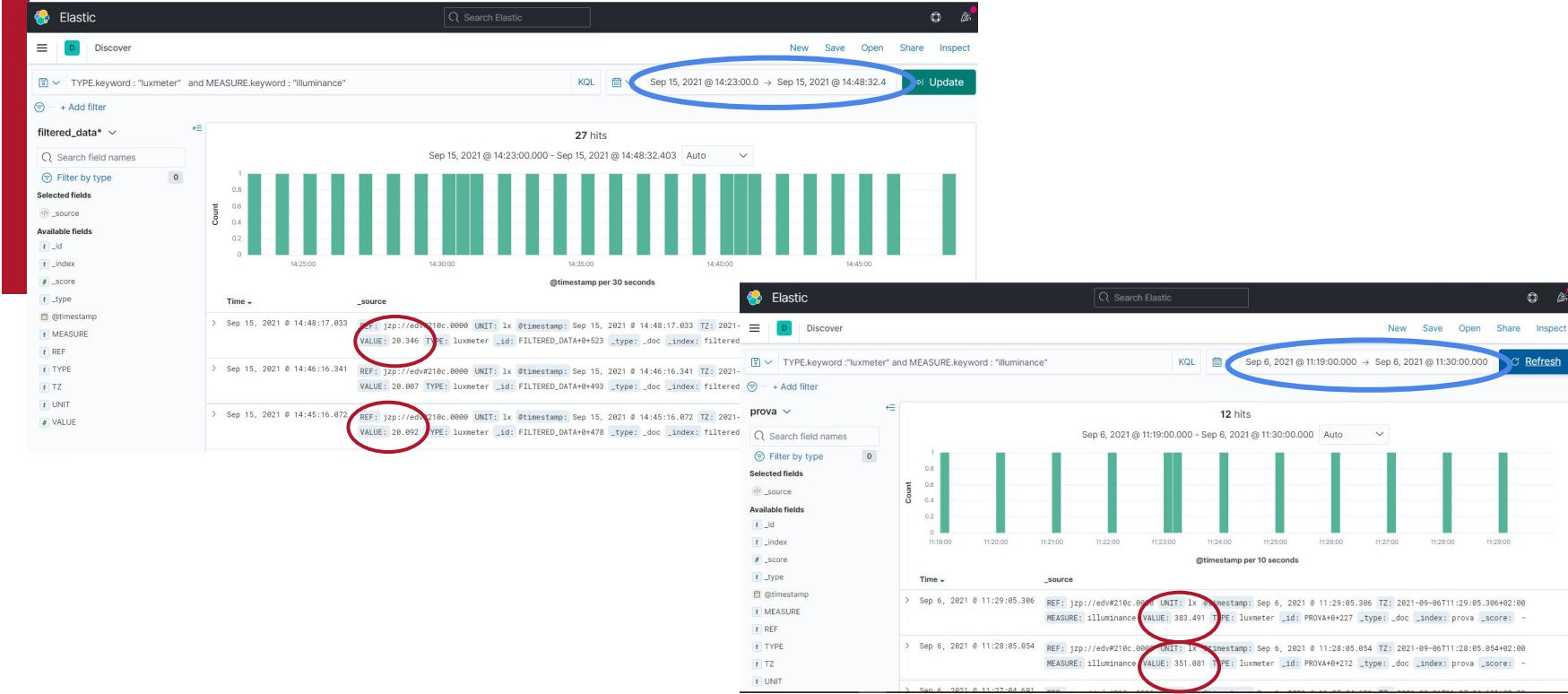
```
TYPE.keyword : "presence" and  
MEASURE.keyword : "presence" and VALUE : 0
```



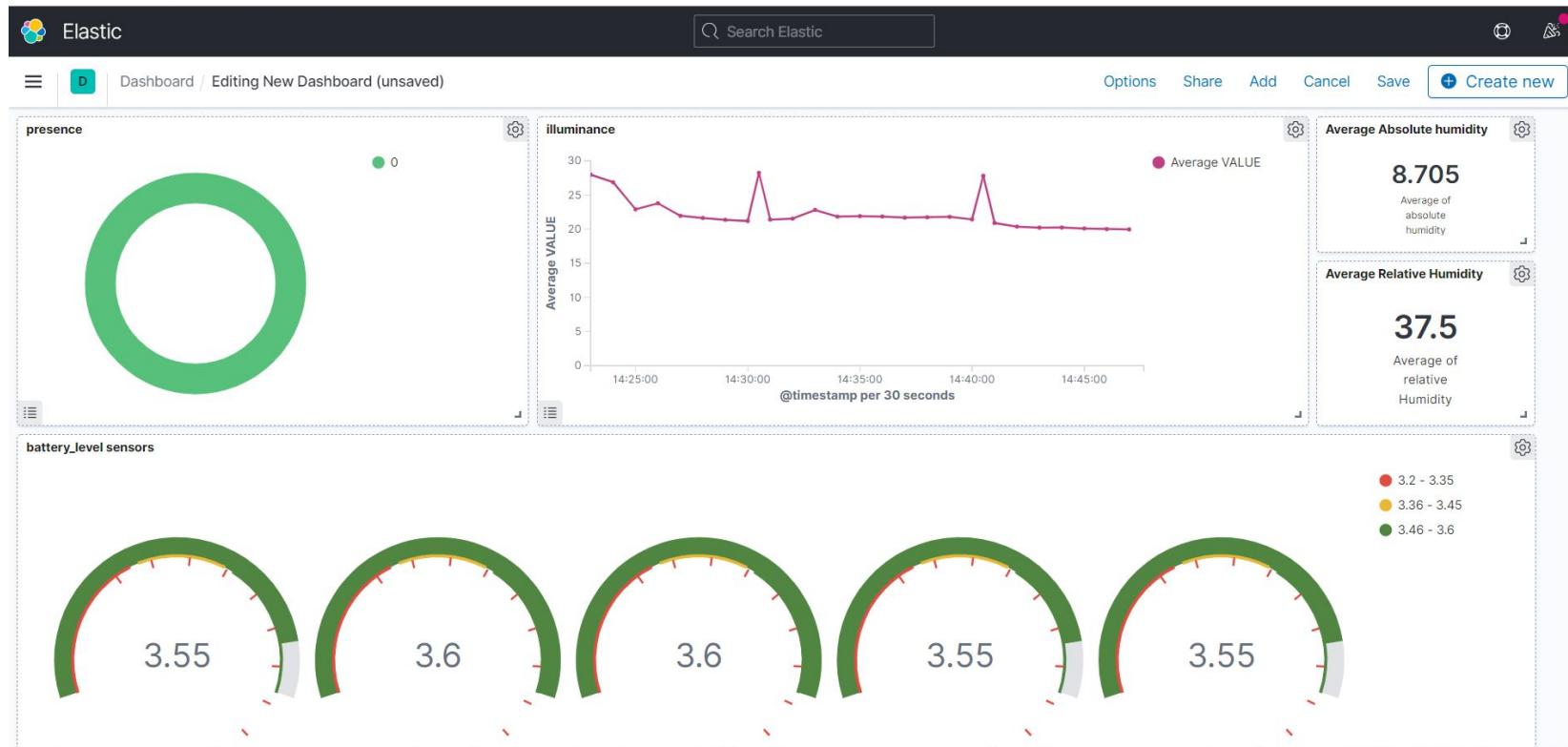
DISCOVER VIEW IN KIBANA

KQL QUERY 3

TYPE.keyword : "luxmeter" and
MEASURE.keyword : "illuminance"



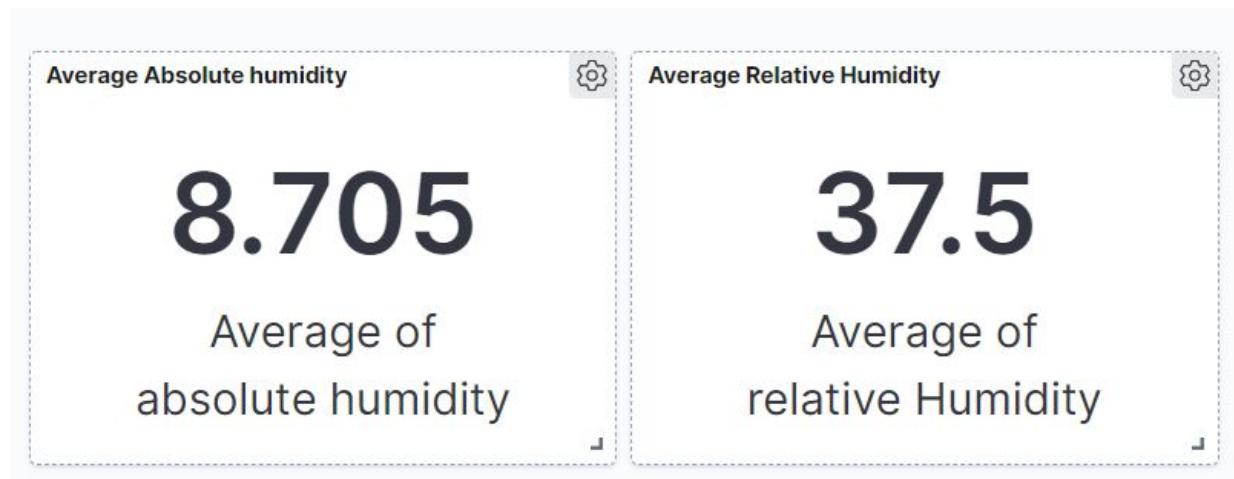
DASHBOARD IN KIBANA



DASHBOARD IN KIBANA

Graphic for the visualization of the absolute and relative humidity:

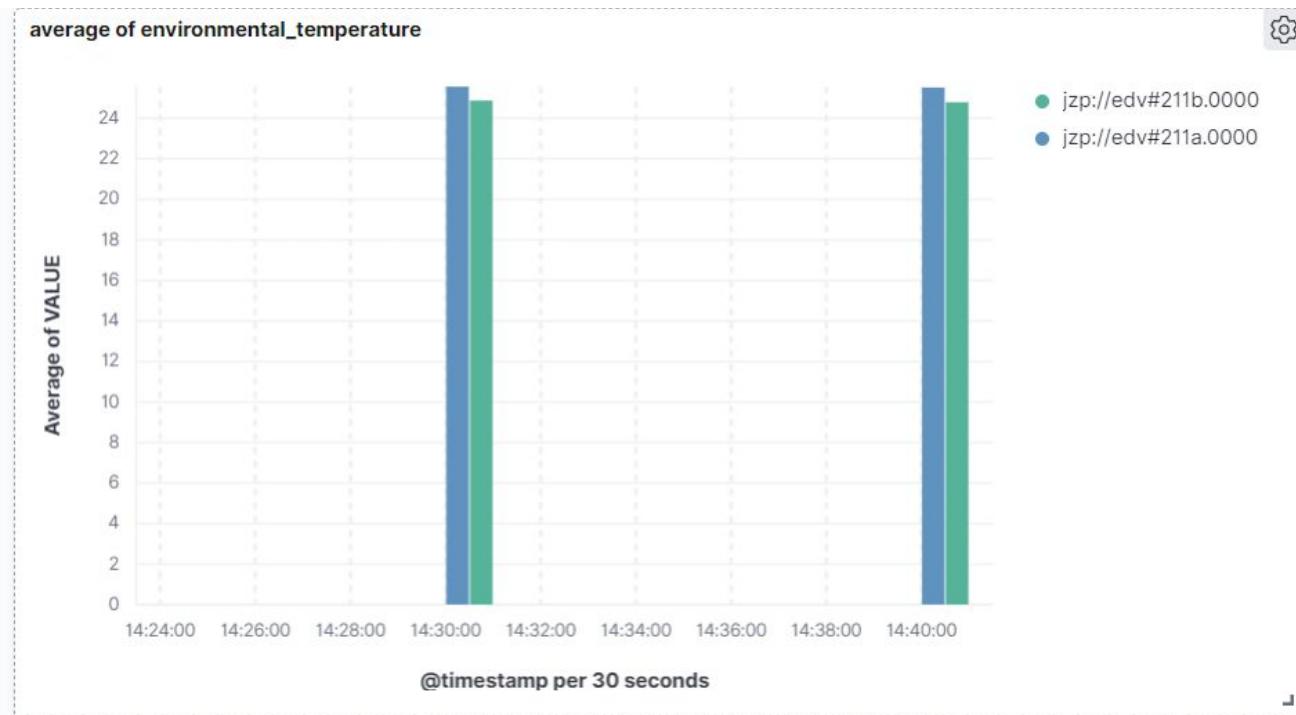
Use metric chart to display the absolute and relative humidity in the classrooms.



DASHBOARD IN KIBANA

Graphic for the visualization of the average of the environmental temperature in the classrooms

Use a Histogram chart to display the average of the environmental temperature in the classrooms, grouped by each sensor



DASHBOARD IN KIBANA

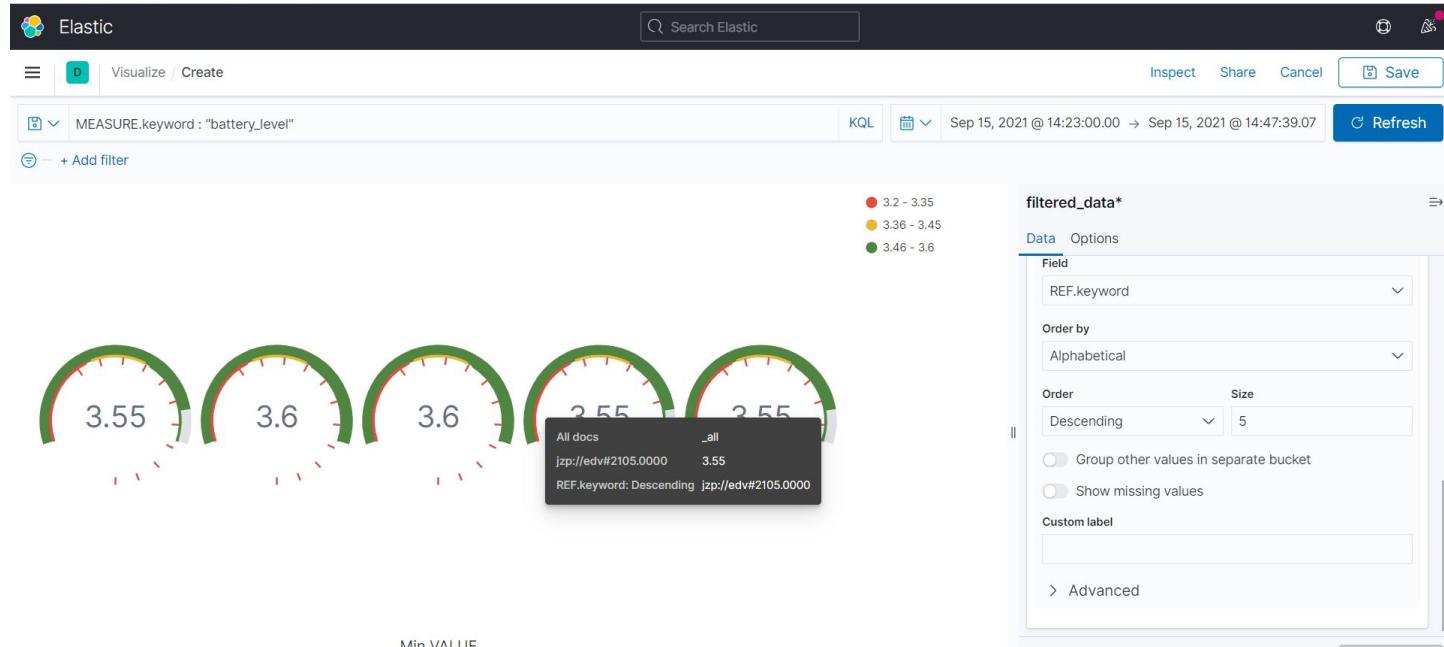
Graphic for the visualization of each sensor battery level

Use a Gauge graphic where we set 3 levels of battery with a color:

red (3.2-3.35) -> low battery level

yellow (3.36-3.45) -> medium battery level

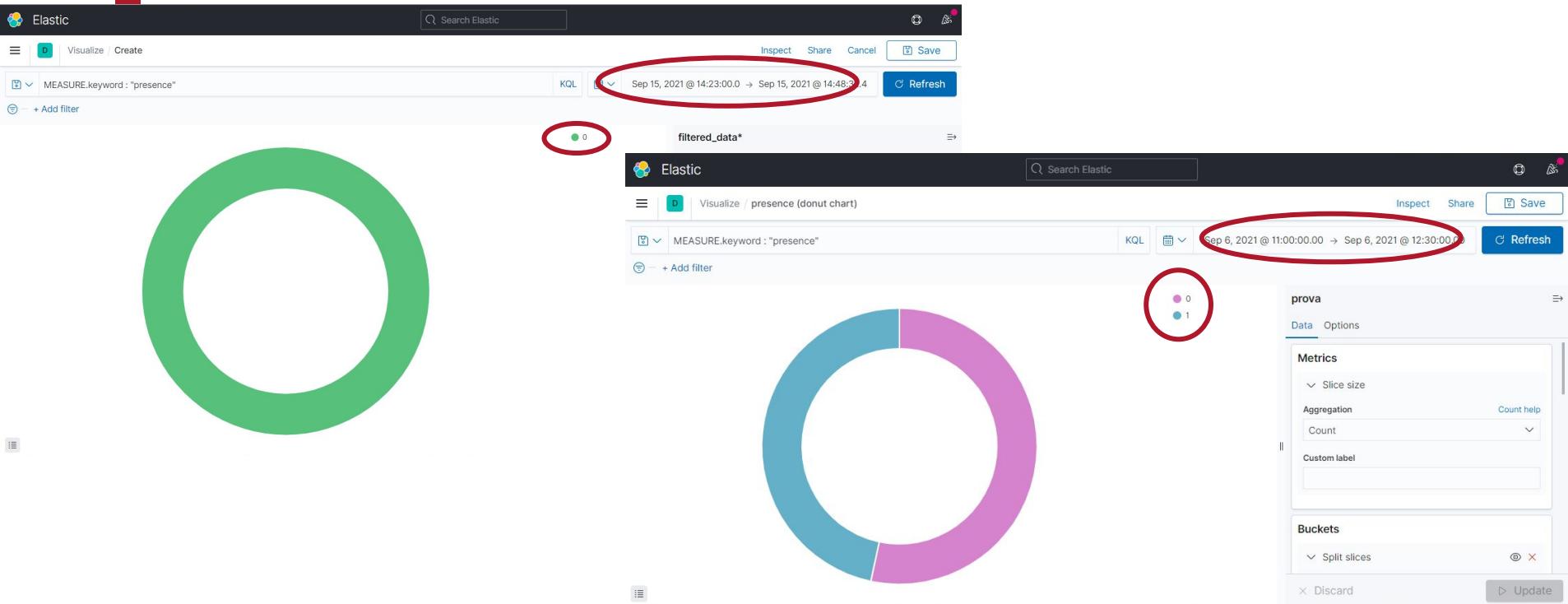
green (3.46-3.6) -> high battery level



DASHBOARD IN KIBANA

Graphic for the visualization of the presence in the classrooms:

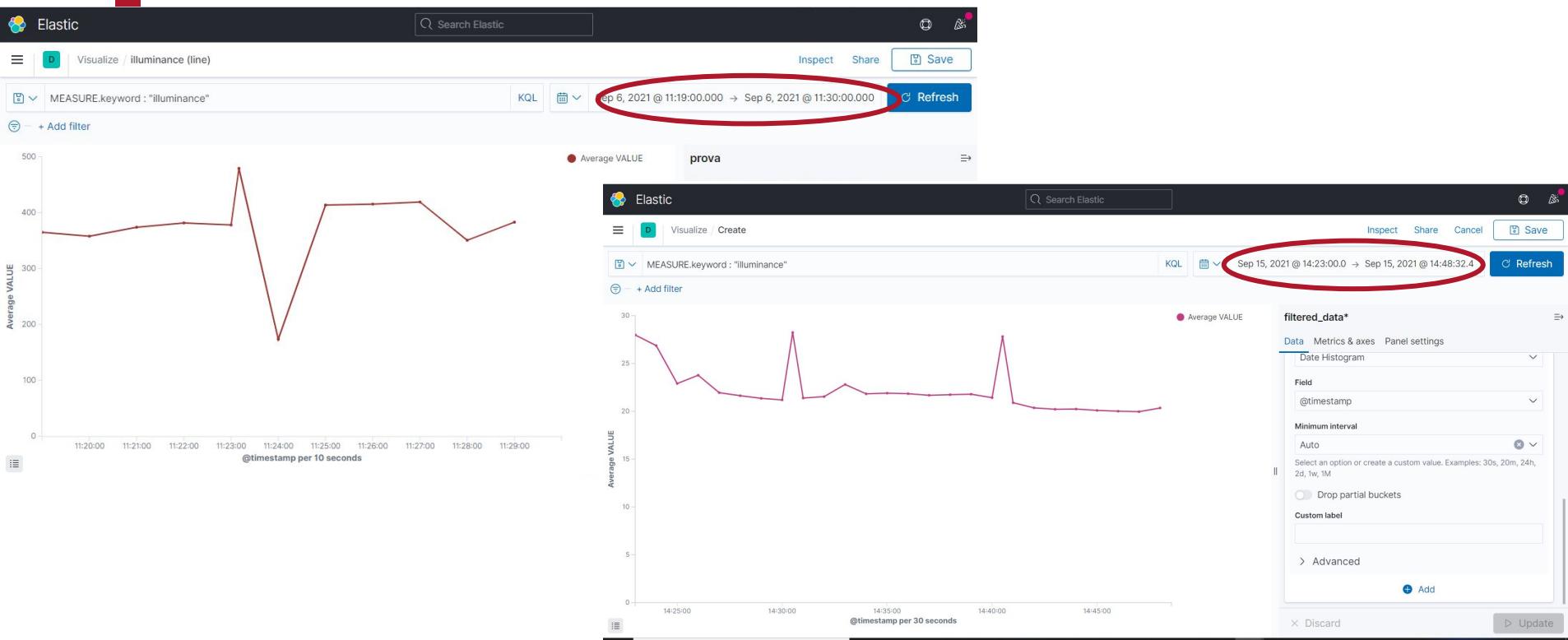
Use a Pie chart to show the presence in the classrooms. 0 is for no presence and 1 for the presence.



DASHBOARD IN KIBANA

Graphic for the visualization of the illuminance in the classrooms:

Use a Line chart to display illuminance in the classrooms.





Confluent Cloud



cloud

Using Confluent Cloud with
Elastic Cloud



If Confluent Platform
supports the version 0.20 of
KsqlDB, use directly the
type timestamp without
using the pipeline



Create new kind of
dashboards in Kibana that
focus on specific data.



FUTURE WORKS

Thanks For The Attention

Does anyone have any questions?



giulia.morelli@stundeti.unicam.it
jessica.piccioni@studenti.unicam.it



<https://github.com/jessicapicc/elk-IoT>