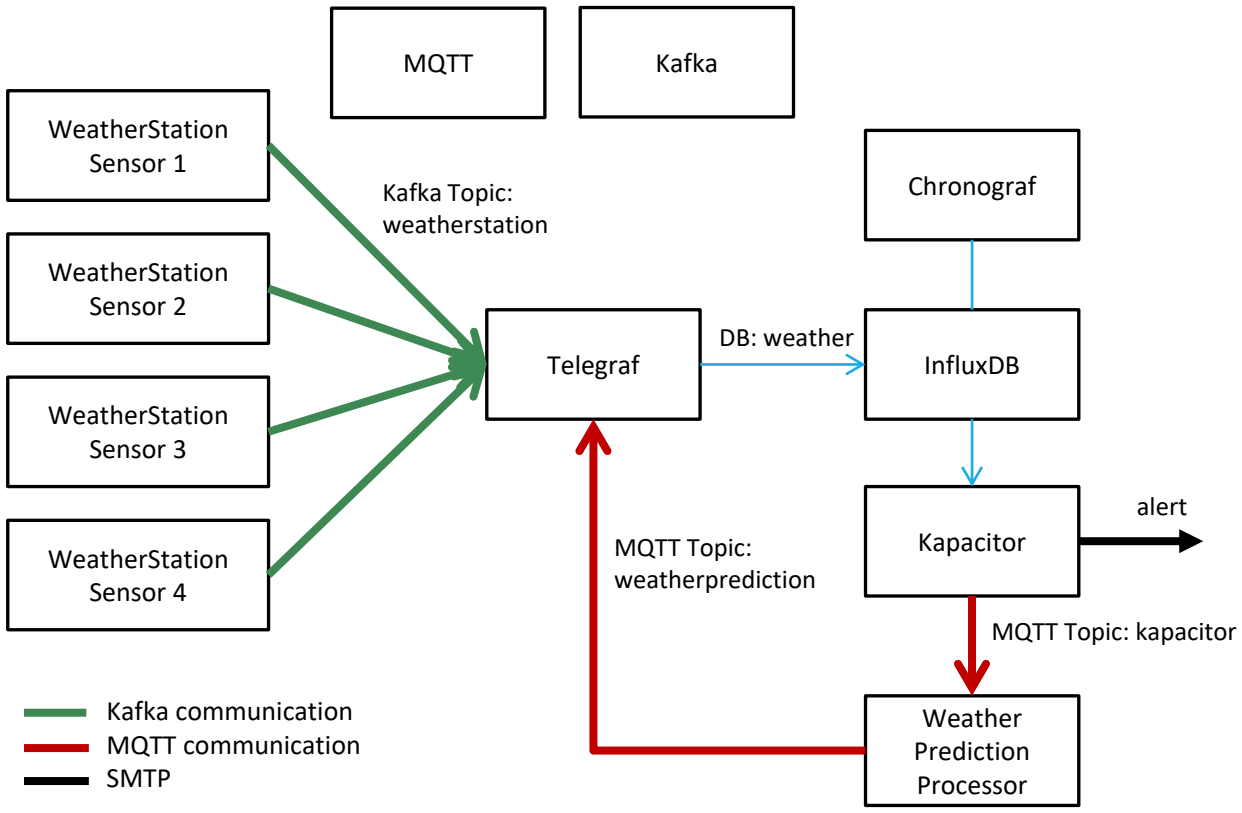
Assignment 2

COMPUTING SYSTEMS AND INFRASTRUCTURES

(SISTEMAS E INFRAESTRUTUTURAS DE COMPUTAÇÃO)

Assignment 2

- The aim of this exercise is to develop a Weather Station that includes:
  - Weather sensors that collect temperature, rain, uvindex and wind sensors dada
  - Time series database that stores all data
  - Weather prediction processor that analyses in real time the sensors data and produces weather predictions for the next day
  - Sends weather alerts to an external email
  - Visualization dashboards to monitor the weather and predictions
- Create a **Docker compose** file that is able to instantiate the required goals by instantiating and interconnecting the required services:
  - Message broker (RabbitMQ)
  - Kafka broker
  - TICK stack
  - Python script – WeatherStation
  - Python script – WeatherPredictionProcessor



WeatherStation

- Python docker with python script to simulate a weather station that periodically reads data from temperature, rain, uvindex and wind sensors.
- WeatherStation service should set 4 replicas to deploy the four dockers.
- Script should send every second sensor data to the platform. The senser data must be read from a file (“sensors_sample_data.csv”) – to be created by the students. This sample file sensors data should be sent to the platform in loop, and in each loop the values should be multiplied by a random number between 0.7 and 1.3.
- Both script and sample data files should be bind from the local filesystem to the containers.
- The sensor data is sent to the platform through Kafka in the topic “weatherstation”. The message must follow the Influxdb line protocol and send the four sensors data in the same message. The measurement name should be “weatherMesasurement”.

WeatherPredictionProcessor

- Python docker with python script to simulate a prediction service that receives frequent real-time average weather measurements and creates weather predictions for the following day.
- Script should receive the real-time weather measurements and calculate the last 5 values received average. If the average decreases from the previous calculated value, the new prediction is 90% of the average value, but if the average increases, the new prediction is 110% of the average value.
- Frequent real-time average weather measurements are received from MQTT topic “kapacitor”.
- Weather predictions must be sent every 10 seconds to MQTT topic “weatherprediction” following the Influxdb line protocol. The measurement name should be “predictionMesasurement”.

Influxdb 1.x

- The influxdb should store all time series data in the database “weather”.

Kapacitor

- Kapacitor configuration must be done in a configuration file that is bind to the Kapacitor docker.
- The TICKscripts should be loaded from a bind directory.
- Create two TICKscripts:
  - Alert script: if temperature > 30ºC or < 5ºC send alert to your DEI email (recommended to use DEI SMTP server)
  - Processor script:
    - Calculate the mean temperature in periods of 30 seconds, every 1 second
    - Send the calculated mean value to MQTT topic “kapacitor”
- Use stream for both TICKscripts
- Do not submit the assignment with the requirement to use Chronograf to configure or load the TICKscripts.

Chronograf

- Connect Chronograf to Influxdb and Kapacitor.
- Create one visualization dashboard to allow the monitor of the weather and the weather predictions. This dashboard must include at least one visualization type of: graph, table, stat, gauge.
- Create one Chronograf weather alert when rain is higher than 5.
- These Chronograf configurations must be included in the project files

Docker

- Rely only on official images used in classes to develop this exercise (RabbitMQ, bitnami/kafka, Python, etc.)
- Create the required Dockerfile to build custom images
- Usage of volumes is required when appropriate to have persistent data
- Usage of networks is required to isolate as much as possible the different services
- Only services that are require to publish ports should publish them
- Address the containers dependency requirements (create custom health checks if required)
- The compose file must be self contained (to not depend on existing custom images, volumes or networks)
- No configuration can be dependent on the operating system or on absolute file paths. Use only relative paths to your assignment folder.

Notes

- The assignment is to be developed in groups of 2 students
- The assignment is to be submitted in Inforestudante in a ZIP file containing all required files
- Students must subscribed to one of the available defence time slot available in Inforestudante

- Submission deadline: 3 of December
- Defence: 12 of December