1. **Introduzione: parla della nascita del progetto e che cos’è l’elaborato della tesi**

My thesis paper talks about a prototype of an industrial web application. This project was created with the support of algoWatt, that is the company where I’m employed, after the conclusion of a project in the same context.

1. **Architettura: spiegare i macrogruppi con breve descrizione**

The architecture of this prototype is the one in the figure. It can be divedied in few main-groups, each of them composed by some components.

* 1. The Input Flow group contains the components dedicated to the communication with the IOT devices.
  2. The Data Management group allow a user to configure and maintain a registry of all the IOT devices employed in the system.
  3. The UI group expose the dedicated web pages to a user in order to perform the actions offered by the Data Management group.
  4. The Auth and user profiling group is the sub-system in charge to verify if the interacting user is allowed to navigate and perform action on the system
  5. The Monitoring group gives a complete overview of the health status of the whole envrionment and give views to display data produced by the system

1. **Tecnologie core: K8s e Helm per il deploy dell’applicazione**

To release and handle the deployment of the application are used two foundamental technologies:

* 1. **K8s: descrivere brevemente le risorse usate su ogni componente**

wich is a container orchestartor, it provides many resources to deploy components. For example: a Deployment describe how to create a Pod and wich policy use to maintain a consistent state for the application in case of failures. A Service is used to instruct K8s to create a DNS address to allow a Pod to recieve requests from others.

* 1. **Soffermarsi sugli operator K8s**

an Important resource of K8s is the Operator wich is an application-specific controller that can help to package, deploy, and manage a Kubernetes application.

* 1. Helm is a Package Manager for Kubernetes that helps a developer to release an application in a faster way

1. **Kafka:**

the basic technology employed to create the application is Apache Kafka.

* 1. **spiega cos’è e perché è impiegata**

it is a event-driven streming bus with very high performaces, it is used to distribute messages through the whole system with very low latency.

* 1. **vantaggi usati**

Kafka main functionaliy is the usage of topic to distinguish the context of a message, Kafka goal is to dispatch these messages from the producers to the consumers. A Producer is an actor that has the right to send messages in a chosen topic. The Consumer is a different actor that waits to recieve any message from a specific topic. A Consumer Group identify more Consumer that wait for messages in the same topic with the same group-id, Kafka will distribute this message to a single consumer inside the group using a policy like round robin.

* 1. **tecnologie di contorno**

Kafka works with Zookeper and JMX, Zookeeper is used by Kafka brokers to determine which broker is the leader of a given partition and topic and perform leader elections, JMX is a Java extension library configured to exports statistics about Kafka performances and resource consumption.

1. **Input Flow:**

The Input Flow component members has the goal of producing or getting data for the system.

* 1. **Python perché è usato**

All the components are developed in Python, because its malleability fit better for this project.

* 1. **Simulatore di termometro, anemometro, e solarimetri**

For this prototype I created some emulator that generate pseudo-random data of thermometers, anemometers, rain gauges and solarimeters, in order to test the functionality of the application.

* 1. **Spiega cosa sostituisce il simulatore**

This emulator take the place of components that can be called Adapters, that is a dedicated software developed for allow the communication of real physical device with Kafka and so to distribute data inside the application. This Adapters can be developed in Python because there are libraries that allows communication to a low level hardware as C++, and so it can be possible to allow a non IOT device to communicate with Kafka.

* 1. **Parla del manager**

The last Python component to discuss of is the Adapter Manager, that is a dedicated software who checks if there are any communication issue between any Adapters and Kafka. If any situation happens, the manager notify it to the rest of the system almost in real-time.

1. **Data Management:**

The Data Management group has the aim to handle the registry of the used IOT devices in the system and modify the configuration of them.

* 1. **Base dati Mysql vantaggi**

The registry data are stored in a MySQL database. It is used beacause the free version has the necessary functionality used in this application and the sucurity mechanism fit perfectly in the context of the app.

* 1. **Spring Boot vantaggi**

All the services provided in the system are developed using Spring Boot, it is a Java framework with a lot of battle-tested functions that ease the job of a developer. Spring Boot allow a developer to concentrate more in writing the functionalities instead of configuring them.

* + 1. **Perché divisione dei microservizi**

I chose to develop more services with less functionalities in order to distribute in a better way the communication traffic and reduce the possibility of bottleneck due a heavy computation.

* + 1. **Scopi dei microrservizi**

For this reason I created a REST service that is dedicated to the HTTP REST traffic and store the configurations of the IOT devices in the Database.

The Socket Service have the aim of performing real-time communication between UI and the system, for example: to show alarms in case of communication issue of an Adapters or getting the initial configuration of a IOT device.

The Configuration Scheduler is the only service that non interact with the UI because its objective is to syncronize the desired configuration selected by the User with the IOT devices

1. **UI: perché React, vantaggi**

The UI is where a user can actually apply the possibilities offered by the application, it is developed in React and some other libraries are involved, like Axios for the Http Rest requests, Socks-Js for opening a socket communication with the socket service and AntD for an easiest creation of responsive react components. React was choosen because it offer the possibility to reuse already created React components and so to speed up the developiment process.

1. **Authentication:**

In the context of the industrial application, is very important to protect the sensitive information abuot the production prcesses and the users allowed to check them. In this groups are used two compoents:

* 1. **Keycloack: cos’è e com’è usato**

In order to pretect the data, Keycloack is employed- It is a Java open source identity and access management solution that already implements all the requirements for the main security protocols for the Authentication and Authorization processes. In my solution it is configured to allow the user creation just to an Admin User, the created users will have roles that define wich functionalities can be used inside the application.

* 1. **Gateway:**

The second compont is a Spring Boot Gateway.

* + 1. **Spring Boot router ignorate**

Spring Boot offers a functionality to develop a Gatway that works exaclty as a router, this allow to redirect the traffic to the correct service by definig some simple rules.

* + 1. **funzionalità usate per SSO**

But this component is enriched to communicate with Keycloak after a user perform a correct login. This gateway integrate a web Session where it is putted a JWT token negotiated with Keycloack, in the JWT are stored the sensitive information of the logged user, like the role in the system. As soon as this token is valid, the web session will be maintained active and the user can navigate inside the UI, otherwise, it must perform a login again in order to access the application.

1. **Monitoring:**

This groups contains some common componets that can be found in every web application of this kind since they are implemented in a very malleable way and still perform a very good job in every context. All the employed components in this group has the aim to collect and dispaly information and data about the application.

* 1. **Influx: vantaggi e uso**

Influx is an open-source time series database with very high performances used to collect the data distributed by the Adapters.

* 1. **Telegraf: funzionalità e uso**

The Telegraf is a foudamental and malleable component that can be used to collect information from any configured source and store it in any configured destination. In this application it’s source is Kafka and its destination is Influx, the Telegraf consumes messages from the same topic where the Adapter produce them.

* 1. **CAdvisor: scopo**

Cadvisor is deployed to retrieve metrics detailed for all the pods in the application

* 1. **NodeExporter: scopo**

NodeExporter do the same objective as Cadvisor, but it collect the metrics just for the physical host machine where the application is running.

* 1. **JMXExporter: scopo**

The JMX is the component explained before during the Kafka presentation

* 1. **Prometheus: scopo**

This last three subjects are used by Promethes to collect all theee metrics and share them. Promethes implemeents a timeseries DB where it puts all the scraped values from the configured sources and offer web api to query all theese metrics.

* 1. **Grafana: scopo e uso**

Finally, Grafana is the components with the aim to display data from the configured sources with the creation of dashboard and panels. In this application Grafana provides more dashboards to display data for the sources Prometheus and Influx, so it offers a view for the resource consumptions for the hosting machine, a view for the resource consumption detaild for every pod in the application, few views about the resource used by Kafka and the Kafka resources employed for the communications. The last dashboard shows data generated by the adapters.

1. **Tests:**

In order to ensure the stability and security of the application, many tests are implemented

* 1. **Unit test**

The unit tests are tests performed on a single functionality, they ensure that the tested functionality behave as expected depending on the given input of the tested functionality. For the spring boot applications it is used the library Junit5, for the Python components it us used the combination of libraries Pytest and unittest

* 1. **Integration test**

In this application are implemented also some integration tests, wich has the aim to ensure that different components interact in a expected way each other without interrupt the correct flow of the application.

* 1. **Cypress, cos’è**

Cypress is a NodeJs library that allow a developer to perform both unit tests for react component and integration tests. After creating a test case, cypress offer a GUI where it is possible to show in realt-time which actions are performed during the test. In this application, the tests are made to verify that every interaction the user made with the UI, correspond to a spefici response from the system, there are alsotest that verify that a user without the correct role to interact with the UI cannot perform the same actions as a different user with the role.

1. **Pipeline CI/CD: scopi, test e release**

All the work that I described can be stored in a Git Repository, this allow to apply versions on the apllication and so to release it, or simply merge each contribue of the developers. To release an app, CI/CD pipeline are involved. A pipeline describe a set of instructions that has to be made in order to distribute an application, and so it apply a version, test the application, build the deliverable and realese it. The pipeline is automatically execute when a developer push his contributes to the remote repository by a dedicated job in a machine dedicated to perform the actions described by the pipelne.