# Web application designed and developed in containerized environment

* Index
* State of the art
* Architecture
  + Streaming bus
    - Apache Kafka
  + Micorservices
  + Monolithic Service
* Technologies
  + Kubernetes
  + Helm Charts
  + Java 11
    - Spring Boot
    - JBoss (?) WildFly(?)
  + React JS
    - Redux
  + Time Series Database
  + Relational Database
* Security & Privacy
  + JWT
    - The cover up JWT
  + Login using JWT
    - Realms
    - Roles
    - Claims
  + Keycloak
* Devops – Continuous integration
* Unit testing
  + JUnit
  + Cypress
* Agile
  + Scrum
* Benchmarks
  + Scalability definition
  + *Streaming* Bus: Apache Kafka vs Moquitto
  + *Services*: Micro vs Monolithic
  + *Time Series DB*: Influx vs Redis (?) vs Mongo (?)
* Future developments

## Index

During my working experience in algoWatt s.p.a, I was employed to develop a very complete web application that were built with some of the advanced and challenging technologies that are currently used in the development of industrial application contexts. In this experience, I was involved in writing code for a large part of the components of the project, which I learned to manage, and which enriched my knowledge. However, there was much more to discover about the entire web application.

The idea for this master thesis project was born from this industrial web application with the goal of studying the technologies used and test their performances in different conditions. My work was to develop a system that could handle and communicate with many **IOT devices** with the best performance possible, using the technologies mentioned below.

My project relied on Kubernetes and Helm, that are software used for the deployment and the monitoring of all the component of the web application.  
Some of the components are prepared images took from public repositories, like [docker.io](http://www.docker.io/), which gave me many options and functionalities in developing the application. In particular, I used the Kafka image from the repository [quay.io](https://quay.io/repository/strimzi/kafka) deployed and maintained by the [Strimzi](https://strimzi.io/) community, this one is the core business of the entire project. Others example are Grafana and Prometheus images, that are basically used to monitoring a system and displaying some graphical representations of it.  
To complete the project, I had to develop some other images myself using frameworks like Spring Boot, libraries like React with Redux and Axios, or programming languages like Python.  
In the end, many pipelines for the Continuous Integration/Continuous Delivery are implemented and used to update the system in a faster and automatic way

All these technologies were chosen because of the better reliability, personal knowledge, and documentation, indeed, almost all of them are widely used. For example, Red Hat developed his own cloud service called **Red Hat OpenShift Streams for Apache Kafka**, which already implement a Strimzi Kafka deployment, and it’s configured using Helm Charts and Kubernetes. It’s also had a deployment for the monitoring based on Grafana and Prometheus, which allow the maintainers to check the health status of the systems in term of resources (RAM utilization, CPU utilization, Kafka topic consumption) with specific and detailed graph.

## State of the art

My work is divided in two branches that I can identify as **Backend** and **Frontend**, connected each other via the streaming bus. The Backend is the part that strictly operate with the IOT devices and collect data from them. Instead, the Frontend contains all the components dedicated to the interaction with the **final User**, which can check the entire system status and the detail of a single IOT device.

There is a component which is dedicated to the User Interface, and it was developed using React. Some other components are developed using the Spring Boot Framework for the rest services, doing some scheduled jobs and maintain a socket service.

## Technologies

My project relied on two main software for the deployment of all the component of the web application, which are Kubernetes and Helm.  
**Kubernetes** is an orchestrator, it automates all the actions for the deployment, maintenance, monitoring, scaling to easily handle every request of the customers and give a better management of the application.  
**Helm** is a package manager who speed up the process of deployment of the Kubernetes objects, configurations, and installations of every component of the project. The power of Helm and Kubernetes together is that once you have all the images and configurations ready, you can just write a single command to install or upgrade your application.

The components are deployed using different technologies that are React, Spring Boot and Python.   
**React** is a JavaScript library used to create User Interfaces for Single Page Application, in combination with other JavaScript libraries that are **Redux** and **Axios**, is possible to handle easily all the pages of the application and add the possibility to interact with the exposed Rest API.  
**Spring Boot** is a Java framework used for multiple functionalities like the exposition of the previous called Rest API, interactions with the Database, scheduled jobs, socket implementations and gateway functionality. It’s used due to his great reliability in security, modularity, and automatic configuration of predetermined functionalities.  
**Python** is a dynamic object-oriented programming language, it can be used to deploy almost everything, is widely used in Data Science, in my project it’s used to deploy some internal Rest API and communicate with the streaming buses.

In my web application there are some public images that I used to take advantage of their functionalities. An image is a prebuilt software of every kind, these images could be found in public repositories like [docker.io](http://www.docker.io). Once I found the software that I need, I created the designed container, using Helm and Kubernetes, to host the image and use his functionalities. A container is set of process isolated from the rest of the system, many containers can be host in the same machine without colliding when they use physical resources. This modularity allows the developers to speed up the usage and the distribution to the customers. Some crucial containers used in my web app are Kafka, Grafana, Prometheus, MySql, and InfluxDb.  
**Apache Kafka** is an open-source distributed event streaming platform which allows to publish, subscribe and store data flow in real-time, it allows to handle a large quantity of data form many inputs, or producer, to many outputs, or consumer.  
**Grafana** is a free multiplatform web application developed in Go, it’s used for the interactive visualization of data generated form any system. In an industrial context, Grafana give a lot of dashboards to allows the maintainers to check the status of the system in which the web application is deployed, it can be also used to simply display some data from a proper source that could be configured inside the containers itself.  
**Prometheus** is a free application for event monitoring and alerting developed in Go, it records metrics in a time series database form every configured source which expose his metrics in a specific format. Grafana supports querying Prometheus, as Grafana has Prometheus in his configured sources, along with many dashboards to display the collected metrics.

Obviously, I have some persistent data in my system that I must store, for this reason a container with a MySql image is deployed. **MySql** is an open-source relational database management system developed following the standard ANSI SQL and ODBC SQL, it’s supported by almost all the major programming languages, and it guarantees high-performance, reliability and scalability.  
**InfluxDb** is an open-source time series database developed in Go and it’s used for store time-based data such as measurements and metrics for real time analysis. It is built to ensure high throughput ingestion, compression, and real-time querying of that same data.

In the end, to maintain a backup of all the files used in my project and keep track of the changes during the development, was used a Version control system call GitHub.

Kubernetes is a widely used software created by Google, who set it open source in 2014. Some of the biggest companies who use Kubernetes together with Helm are, of course, Google, Spotify, The New York Times, and Adidas. The Java framework Spring Boot is widely used by the biggest companies like Netflix, Udemy and Trivago due to his security, scalability, and simplicity. Python, instead, is even more used than Spring Boot, especially in Data Science due to the simplicity in reading and writing code. Some big companies that use Python are Wikipedia, CERN, again Google and Spotify, Amazon, Meta, and NASA. Finally, React is a JavaScript library created by Facebook back in 2011 and it’s use by obviously Meta, Netflix, AirBnB, and Dropbox. Speaking about Databases, for first let’s look to MySql, it’s a consolidated DBMS based on SQL and used by Sony, BBC, and Uber. Instead, InfluxDb used as a Time Series DataBase is widely used by many big companies like PayPal, Adobe, CERN, Cisco, and much more.