# Web application desinged and develop in containerized environment

* Index
* Architecture
  + Streaming bus
    - Apache Kafka
  + Micorservices
  + Monolithic Service
* Technologies
  + Kubernetes
  + Helm Charts
  + Java 11
    - Spring Boot
    - JBoss (?) WildFly(?)
  + React JS
  + Time Series Database
  + Relational Database
* Security & Privacy
  + JWT
    - Not exposure of the token (?)
  + 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 currently used in these 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 them in different conditions. My work was to develop a system that could handle and communicate with many **IOT devices** with the best performance possible, I also want to describe the procedures for the release of the web application and demonstrate the efficiency of them. For this last point was strictly necessary the use of a **Version control service**.

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