

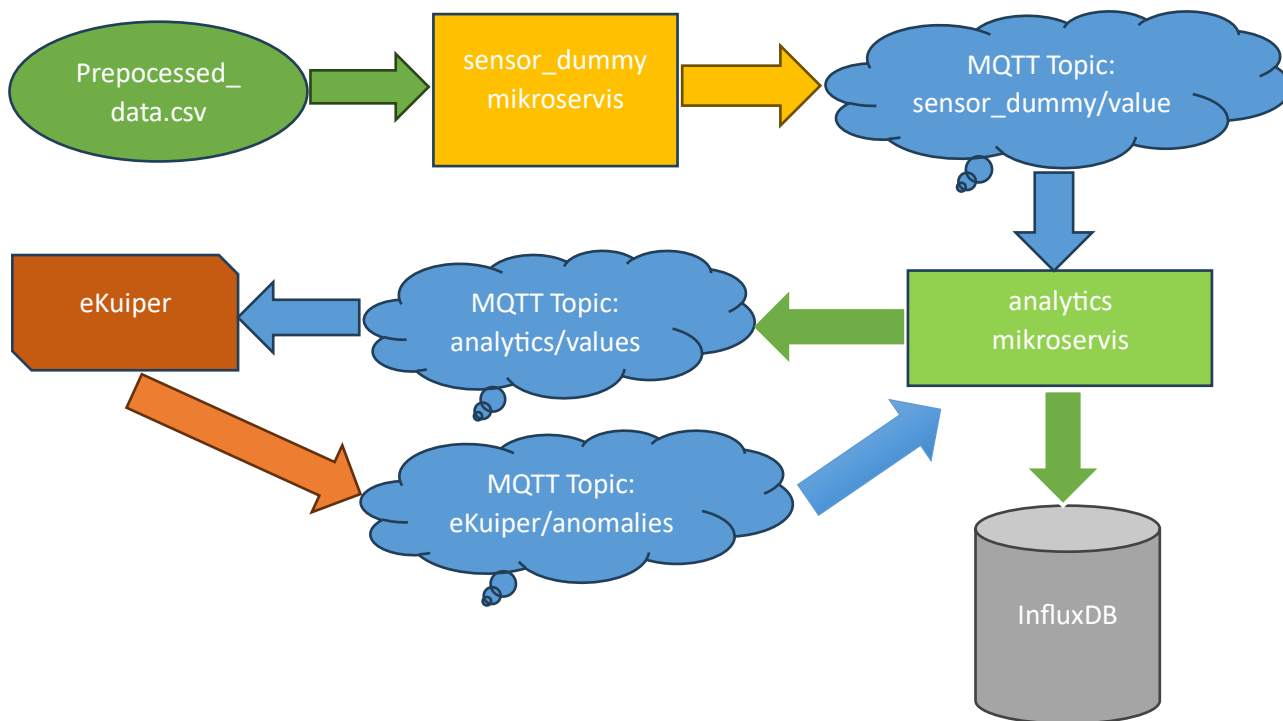
Projekat 2

Cilj ovog projekta je da se implementira mikroservisna arhitektura u razlicitim tehnologijama, kao što su: .NET, NodeJs, Java/Spring Boot, Python/Flask.

Prvi servis koji treba da se razvije je Sensor Dummy i on treba da sa senzora čita podatke i da te podatke šalje na određeni topic MQTT brokera.

Drugi servis koji treba da se implementira nazvan je Analytics, on treba da se pretplati na MQTT message broker i da dobije podatke koji su emitovani na topic od strane prvog servisa. Podatke koje dobija treba da se smeštaju u NoSQL bazu podataka InfluxDB.

Arhitektura sistema



Sensor Dummy mikroservis

Sensor Dummy je implementiran u programskom jeziku JavaScript i koristi Node.js okruženje. Potreban je dodatan paket za rad sa MQTT i za čitanje podataka iz CSV fajla. Podatke koje čita ovaj mikroservis šalje ih na topic pod imenom: **sensor_dummy/values**.

Bitno je da se ovaj mikroservis pokrene tek nakon što se pokrene EMQX kako bi podaci mogli da se šalju.

Za MQTT broker je izabran EMQX broker.

Analytics mikroservis

Glavni zadatak ovog mikroservisa jeste da pročita podatke koje Sensor Dummy šalje na topic sa imenom **sensor_dummy/values** odgovarajućeg MQTT brokera. Zatim te podatke koje čita upisuje na **analytics/values** topic, preko koga se pročitani podaci šalju eKuiper-u.

Bitno je da se ovaj mikroservis pokrene tek nakon što se pokrene EMQX kako bi mogao da pročitati podatke, a i poslao eKupier-u.

Implementiran je u .NET-u verzija 6.0 i koristi dodatne pakete za rad sa MQTT-om i sa InfluxDB bazom podataka.

eKuiper

Nakon izvršavanja docker-compose komande potrebno je da se ode na adresu gde se nalazi eKuiper:

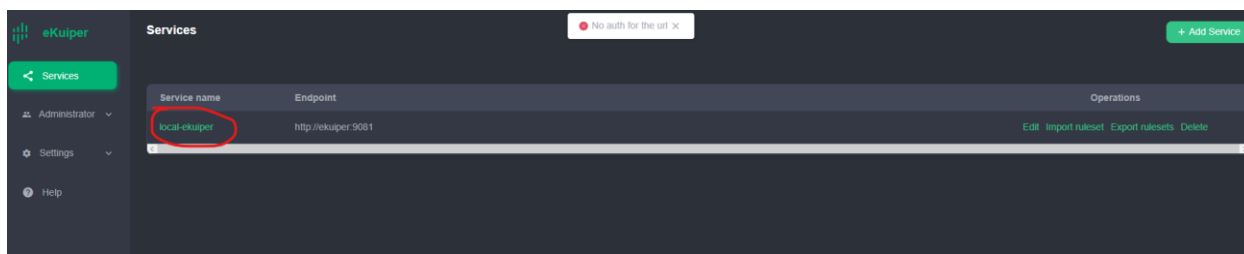
<http://localhost:9082>

Odlaskom na ovu adresu potrebno izlazi login forma gde je potrebno da se unese sledeće:

Username: admin

Password: public

Nakon što smo ulogovani je potrebno da pritisne na local-ekuiper i tu izvrši dalja konfiguracija kao na slici:



Nakon pritiska na local-ekuiper izlazi deo gde treba da se napravi stream podataka pritiskom na dugme create stream i treba da se popuni sledećim podacima:

Stream name: projekat2soa

Type: mqtt

Data Source: analytics/values (preko kog topic-a dobija podatke)

Ostala polja ostaviti kako je bilo pri otvranju forme.

Nakon što se popune podaci potrebno je da se to submituje pritiskom na dugme submit.

Sledeći korak je da se napravi pravilo koje će da u zavisnosti od uslova da aktivira alarm (da pošalje podatke ako se ne upali, ili da signalizira ako podaci nisu dobri), to se biranjem kartice sa imenom Rules, pa onda pritiskom na dugme Create Rule. Otvara se dijalog koji treba da se popuni na sledeći način:

Rule ID: eKuiperRule

Name: The alarm is triggered when the frame length is greater than 256

SQL: SELECT *
 FROM projekat2soa
 WHERE frame_len < 257

* Rule ID

eKuiperRule

Name

The alarm is triggered when the frame length is greater than 256

* SQL

```
1 SELECT *
2 FROM projekat2soa
3 WHERE frame_len < 257
```

* Actions

+ Add

Takodje je ovde potrebno da se dalje pritisne na dugme za dodavanje akcije. Popunjavanje dijaloga treba da izgleda na sledeći način:

MQTT topic: eKuiper/anomalies (ovo je ime topic-a gde se šalju podaci i analytics mikroservis je prijavljen na ovaj topic)

Add action

* Sink

Documentation

mqtt

Resource ID

+ Add sink template

Select

* MQTT broker address

tcp://10.14.42.11:1883

* MQTT topic

eKuiper/anomalies

MQTT ClientID

MQTT protocol version

3.1.1

Ostale podatke ostaviti kako su i bili prilikom otvaranja dijaloga. Onda pritiskom na dugme Test connection testiramo da li je konekcija dobra i idemo na dugme submit. Nakon ovog je preostalo da se podesi baza podataka.

InfluxDB konfiguracija

Smeštanje podataka koji su prošli ono pravilo koje je postavljeno kod eKuiper-a vrši se u InfluxDB bazu podataka. Kako bi konfigurisali bazu potrebno je da se uradi sledeće:

- Da se ode na adresu <http://localhost:8086> i da se kreira nalog na sledeći način:
 Username: admin
 Password: adminadmin
 Organization: organization
 Bucket: iot2
- Nakon toga je potrebno da se ode na stranicu Data Explorer kako bi se prikazali podaci koji su smešteni u bazi:

Graph

CUSTOMIZE

Local

SAVE AS

table	_measurement	_field	_value	_start	_stop	_time	frame_number
mean	group	string	no group	dateTime:RFC3339	dateTime:RFC3339	dateTime:RFC3339	string
0	network	eth_dst	1.67276E+14	2023-08-28T16:34:15.801Z	2023-08-28T17:34:15.801Z	2023-08-28T17:32:44.724Z	22
1	network	eth_dst	8.7972E+13	2023-08-28T16:34:15.801Z	2023-08-28T17:34:15.801Z	2023-08-28T17:32:47.722Z	23
2	network	eth_dst	1.67276E+14	2023-08-28T16:34:15.801Z	2023-08-28T17:34:15.801Z	2023-08-28T17:32:50.725Z	24
3	network	eth_dst	1.67276E+14	2023-08-28T16:34:15.801Z	2023-08-28T17:34:15.801Z	2023-08-28T17:32:53.728Z	25

Query 1 (0.0%)

View Raw Data

CSV

Past 1h

QUERY BUILDER

SUBMIT

```

1 from(bucket: "iot2")
2   |> range(start: v.timeRangeStart, stop: v.timeRangeStop)
3   |> filter(fn: (r) => r["_measurement"] == "network")
4   |> filter(fn: (r) => r["_field"] == "eth_dst" or r["_field"] == "eth_src" or r["_field"] == "frame_time")
5   |> group(columns: ["_measurement", "_field", "frame_number"])
6   |> yield(name: "mean")
  
```

Filter Functions...

Transformations

Functions