**Assignment 2**

**Sensor Monitoring System and Real-Time Notifications**

Student: Buzila-Garda Andra-Maria

Profesor indrumator: Liana Toderean

1. Arhitectura conceptuala

Aplicatia creata cuprinde 2 parti (proiecte mai mici):

* Prima parte incorporeaza MessageProducer-ul. MessageProducer-ul este cel care citeste din fisierul “sensor.csv” si trimite datele intr-un format JSON catre Broker. MessageBroker face parte tot din acest proiect si reprezinta de fapt RabbitMQ, adica trimiterea mesajelor catre Consumer.
* A doua parte contine MessageConsumer-ul care, nu doar ca primeste datele dar le si prelucreaza si anume, mesajul primit in format JSON va fi parsat. Cu datele obtinute se creeaza intanta de clasa, care este trimisa catre repository si ulterior, salvata in baza de date. Toata aceasta procedura se executa odata la 10 secunde. Acesta este intervalul de timp in care se trimit datele prin rabbit.

Consumer

RabbitMQ

Producer

1. Descrierea Implementarii

Pentru partea de Backend am folosit Java Spring, iar pentru frontend React. Am creat

o aplicatie care trimite notificari de fiecare data cand se citesc date dintr-un fisier .csv. Am creat un obiect Json cu 3 field-uri, unul cu timestamp (data si timpul curente), unul cu id-ul device-ului si inca unul cu energyConsumption a device-ului.Acest obiect a fost trimis intr-o coada cu RabbitMQ de unde a fost luat, iar datele au fost parsate astfel incat sa se poata introduce in baza de date periodic. Intervalul de timp a fost setat in Sender, iar reveiver-ul l-a parsat. In frontend am creat o noua pagina unde am afisat notificarile. Din frontend am facut un call catre backend pentru a genera acele notificari in functie de valoarea variabilei energyConsumption. Am comparat valoarea cu valoarea maxima pe care am atribuit-o device-ului respective in Assignment 1. In final am afisat notificarile in real-time intr-un panel.

1. Deployment

Am realizat partea de deploy folosind Docker local pentru backend si frontend, iar

coada creata a fost introdusa in cloud. In codul pentru backend, in ambele proiecte, am pus link-ul de la acea coada. Fisierele Dockerfile si docker-compose au fost preluate de la Assignment-ul 1.

Cloud

docker

localhost:3000

Frontend

Postgresql port:5432

Ds2022\_30641\_buzila\_andra\_assignment\_2 - docker

QUEUE

Backend port:8083

Pentru a rula proiectul, in docker se pornesc containerele, iar proiectul de unde se trimit datele se porneste separat. Datorita faptului ca acea coada creata este pe cloud, accesul la ea se face mult mai usor.

Scopul acestui proiect este de a ne familiariza cu trimiterea asincrona de mesaje de la server la client si invers si de a invata sa lucram cu noi tehnologii, cum ar fi RabbitMq.