UNIVERSITATEA TEHNICĂ DIN CLUJ-NAPOCA

Energy Management System

DISTRIBUTED SYSTEMS

Nume: Muresan Ioana Danina
Grupa: 30643

Facultatea de Automatica si Calculatoare

27 Noiembrie 2023

Cuprins

1 Arhitectura conceptuala a sistemului

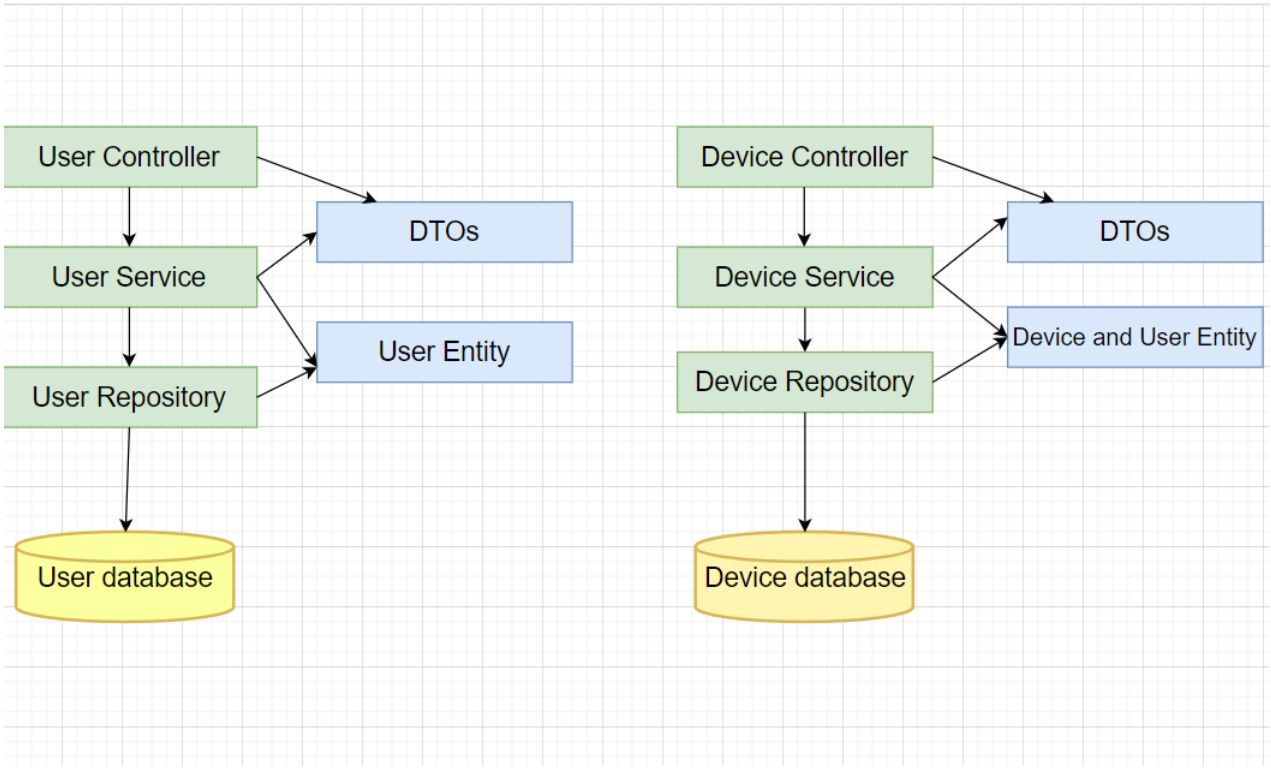
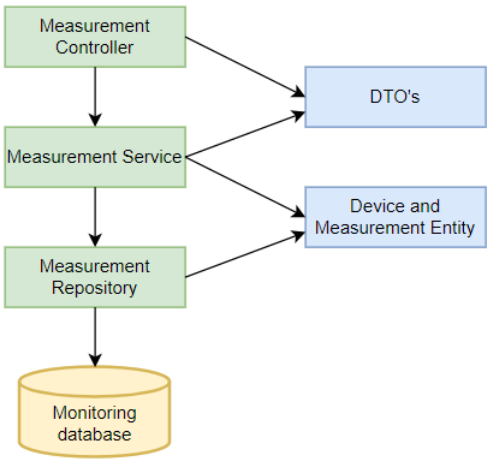


Figura 1: Arhitectura conceptuala

Am adaugat pentru tema 2



De asemenea,am modificat arhitectura conceptuala pentru tema2.

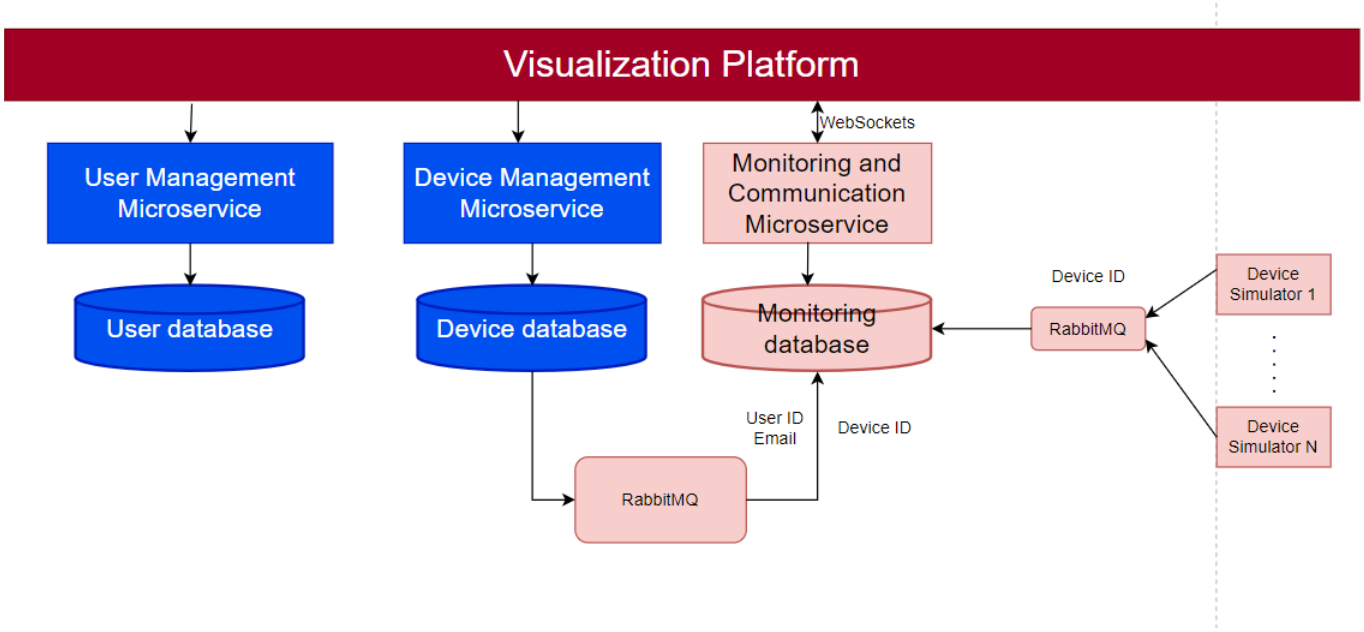


Figura 2: Arhitectura conceptuala

2 Diagrama UML de deploy

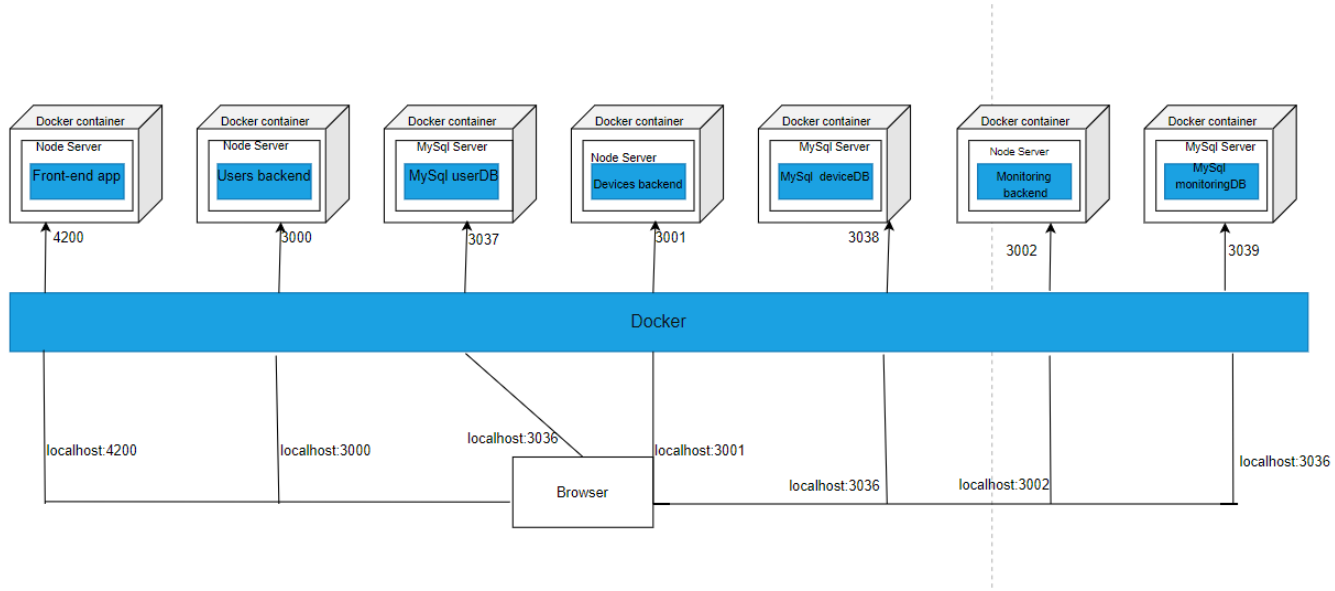


Figura 3: Arhitectura conceptuala

3 Readme

3.1 Descriere

Această aplicație constă din trei backend-uri (micro-servicii) Nest.js, fiecare cu propria bază de date,pentru care am utilizat MySql, proiectate pentru administrarea utilizatorilor și dispozitivelor lor inteligente de măsurare a energiei și un frontend Angular care facilitează interacțiunea cu aceste servere.Sistemul poate fi accesat de către două tipuri de utilizatori după un proces de autentificare: administrator (manager) și clienți.

Pentru tema 2 am implementat si o aplicatie Smart Metering Device Simulator ca Producator de mesaje. Aceasta simuleaza un contor inteligent citind datele de energie dintr-un fisier sensor.csv (adica, o valoare la fiecare 10 minute) si trimite date sub forma ¡ timestamp, deviceid, measurement-value ¿ catre Broker de mesaje (adica, o coada). Marca temporala este preluata de la ceasul local, iar deviceid este unic pentru fiecare instanta a Simulatorului dispozitivului de masurare inteligenta si corespunde dispozitivului id a unui utilizator din baza de date (asa cum este definita in Tema 1). Simulatorul de dispozitiv este dezvoltat ca o aplicatie autonoma (adica, aplicatie desktop).

3.2 Micro-servicii

Pentru utilizarea celor doua micro-servicii de Nest, se va proceda astfel: se verifica initial daca exista Node.js pe calculatorul respectiv, in caz contrar de va instala ruland intr-un terminal **install node.js**, se verifica daca este instalat **npm**. Pe urma,se deschide fisierul in care este salvat proiectul si un terminal.Se ruleaza **npm install** pentru a descarca toata librariile,dependintele utilizate. Apoi se ruleaza comanda **npm run start:dev** sau **npm run start** pentru a porni aplicatia. Daca totul functioneaza ok, terminalul va afisa **LOG [NestApplication] Nest application successfully started**. Micro-serviciul de user ruleaza pe portul localhost:3000 ,micro-serviciul de monitoring pe localhost:3002 iar micro-serviciul de deviceuri ruleaza pe portul localhost:3001. Ambele baza de date ruleaza pe portul 3036. Microserviciul de monitorizare și comunicare are o componentă Message Consumer care va procesa măsurătorile pentru a calcula consumul total de energie pe oră și îl va stoca în Bază de date. Dacă consumul total de energie pe oră calculat depășește maximul definit de dispozitiv valoare (așa cum este definită în Tema 1) notifică în mod asincron utilizatorul pe interfața sa web.

3.3 Frontend

Se navigheaza in directorul unde este salvat frontendul si se ruleaza , intr-un terminal, comanda **npm install** pentru instalarea dependintelor Angular. Apoi se ruleaza comanda **ng serve** pentru a porni aplicatia. Se deschide localhost:4200 intr-un browser web. Aplicatia se deschide cu o pagina de login, in care utilizatorul trebuie sa isi introduca credentialele(email si parola) apoi este redirectionat la pagina specifica atributiilor pe care le are (admin sau client).

3.4 Sensor Simulator

Pentru a porni o instata de Sensor Simulator se configureaza in deviceid.txt id-ul deviceului si intr-un terminal se ruleaza: **node smart-metering-device-simulator.ts deviceid.txt**. Daca se doreste pornirea in paralel a mai multor instante, se creaza alte fisiere de configurare de unde se citeste id-ul deviceului si se ruleaza ,transmitand in linia de comanda numele fisierului de unde se citeste. Cand valoarea citita depaseste valoarea maxima pentru device-ul respectiv, userul primeste o notificare(prin websocket).

3.5 Docker

Aplicația beneficiază de un fișier docker-compose.yml care grupează cele 7 servicii într-un singur container Docker. Acest container permite rularea aplicației fără a fi nevoie să porniți fiecare aplicație individual din mediul de dezvoltare (IDE).Acest fișier docker-compose.yml definește configurația containerelor Docker pentru fiecare serviciu din aplicație, inclusiv bazele de date, backend-urile pentru managementul utilizatorilor și dispozitivelor, precum și frontend-ul Angular. Fiecare serviciu este configurat pentru a utiliza imagini Docker, a expune porturi necesare și a asigura dependențe corecte între servicii.

4 Bibliografie

https://blog.back4app.com/how-to-deploy-a-nest-js-application/

https://docs.nestjs.com/first-steps

https://dev.to/gustavocontreiras/how-to-create-a-dockerized-full-stack-environment-with-mysql-nes

https://angular.io/tutorial/tour-of-heroes/toh-pt0

https://customer.cloudamqp.com/instance

https://backendcommunity.com/how-to-integrate-socket-io-with-angular-and-nest-js/