UNIVERSITATEA TEHNICĂ
DIN CLUJ-NAPOCA

SISTEME DISTRIBUITE

Energy Management System

Moşilă Luciana 30641/2

FACULTATEA DE AUTOMATICA
SI CALCULATOARE
2024

Contents

I. Conceptual architecture

1. Backend

Arhitectura backend este bazată pe microservicii care comunică printr-un mecanism asincron, utilizând RabbitMQ ca broker de mesaje. Fiecare microserviciu are o funcționalitate specifică și comunică cu celelalte microservicii sau baze de date asociate.

1.1. User Microservice

Microserviciul de management al utilizatorilor este responsabil pentru gestionarea utilizatorilor în aplicație, incluzând autentificarea, rolurile și operațiunile CRUD asupra conturilor. Acest microserviciu oferă două tipuri de utilizatori: administratorii, care au privilegii extinse pentru a gestiona dispozitivele și asocierile lor, și clienții, care pot vizualiza doar dispozitivele asociate lor. Fiecare utilizator este definit printr-un ID unic, un nume, și un rol. Administratorii pot adăuga, actualiza sau șterge utilizatori și atribui dispozitive acestora, asigurând funcționalitatea completă de gestionare a conturilor. Microserviciul comunică cu frontend-ul prin API-uri REST și asigură protecția paginilor corespunzătoare fiecărui rol, astfel încât utilizatorii să nu poată accesa secțiuni neautorizate.

1.2. Device Microservice

Microserviciul de Dispozitive este responsabil pentru administrarea dispozitivelor inteligente din cadrul sistemului, incluzând operațiuni precum adăugarea, actualizarea și ștergerea acestora. Informațiile despre fiecare dispozitiv, cum ar fi descrierea, adresa și valoarea maximă permisă pentru consumul de energie, sunt stocate în Devices Database. De asemenea, microserviciul se ocupă cu sincronizarea datelor despre dispozitive cu microserviciul de Monitorizare și Comunicare printr-un sistem bazat pe evenimente, utilizând RabbitMQ. Astfel, orice modificare a dispozitivelor sau a parametrilor acestora este trimisă în timp real către microserviciul de monitorizare, asigurând coerența datelor și un flux continuu de informații între cele două componente.

1.3. Monitoring and Communication Microservice

Microserviciul de Monitorizare și Comunicare este responsabil pentru gestionarea și analizarea consumului de energie al dispozitivelor inteligente, precum și pentru notificarea utilizatorilor în caz de depășire a pragurilor prestabilite. Acesta consumă datele transmise de simulatoarele de dispozitive prin RabbitMQ, folosind componenta Message Consumer, care calculează consumul orar total de energie și stochează rezultatele în Monitoring Database. În cazul în care consumul orar depășește valoarea maximă configurată pentru un dispozitiv, microserviciul trimite notificări în timp real către utilizatori prin WebSocket, asigurând o interacțiune rapidă și eficientă cu aplicația client. Astfel, microserviciul joacă un rol central în sistemul de management al energiei, garantând atât monitorizarea precisă, cât și alertarea proactivă a utilizatorilor.

1.4. Smart Metering Device Simulator

Simulatorul de dispozitive inteligente trimite date simulate despre consumul de energie către RabbitMQ, folosind valori preluate din fișierul sensor.csv. Acesta generează și trimite mesaje la fiecare 10 minute, fiecare mesaj fiind în format JSON și conținând un timestamp (preluat de la ceasul sistemului), un device_id unic (asociat unui utilizator din baza de date) și o valoare de consum (measurement_value). Simulatorul emulează comportamentul unui dispozitiv real, oferind date de test esențiale pentru microserviciul de monitorizare și comunicare, care procesează aceste informații pentru a calcula consumul orar și pentru a verifica depășirea pragurilor maxime. Această

componentă este fundamentală pentru validarea și testarea fluxului de date din cadrul sistemului de management al energiei.

Simulează trimiterea măsurătorilor la intervale de 10 minute către RabbitMQ, folosind un format JSON:

"timestamp": 1570654800000,

"device_id": "5c2494a3-1140-4c7a-991a-a1a2561c6bc2",

"measurement_value": 0.1

2. Frontend

2.1. Vizualizarea și Interacțiunea cu Sistemul

Frontend-ul aplicației este realizat în React, oferind utilizatorilor o interfață intuitivă pentru monitorizarea consumului energetic. Utilizatorii pot vizualiza datele prin diagrame de tip linie sau bară, care afișează consumul orar al dispozitivelor pentru o zi selectată dintr-un calendar. De asemenea, notificările în timp real sunt afișate prin WebSockets, alertând utilizatorii atunci când un dispozitiv depășește pragul maxim de consum energetic. Pe lângă diagrame, aplicația include și un tabel cu dispozitivele asociate utilizatorilor, precum și funcționalități dedicate pentru administratori, precum gestionarea utilizatorilor și dispozitivelor. Frontend-ul comunică direct cu microserviciile din backend, garantând actualizarea în timp real a datelor afișate.

3. Comunicarea RabbitMQ

Comunicarea între componentele sistemului se realizează prin RabbitMQ, care acționează ca broker de mesaje, asigurând un flux continuu de date între Simulatorul de Dispozitive și Microserviciul de Monitorizare și Comunicare. Simulatorul de dispozitive inteligente trimite măsurători în format JSON către RabbitMQ la intervale de 10 minute. RabbitMQ distribuie aceste mesaje către Message Consumer, care procesează datele primite, calculează consumul orar total de energie și îl stochează în baza de date a microserviciului de monitorizare. Dacă consumul depășește valoarea maximă permisă, Message Consumer trimite notificări prin WebSocket către utilizatori, asigurându-se că aceștia sunt alertați în timp real despre depășirea pragurilor de consum. Acest flux de date între componente este esențial pentru buna funcționare a sistemului, facilitând comunicarea eficientă și actualizarea constantă a informațiilor.

4.Cerințe Funcționale Îndeplinite

1.1. Comunicare Asincronă:

Mesajele dintre simulatorul de dispozitive și microserviciile sistemului sunt gestionate prin RabbitMQ, un broker de mesaje care permite comunicarea asincronă între componente. Aceasta asigură că datele sunt trimise și procesate fără a bloca alte operațiuni, facilitând un sistem scalabil și eficient.

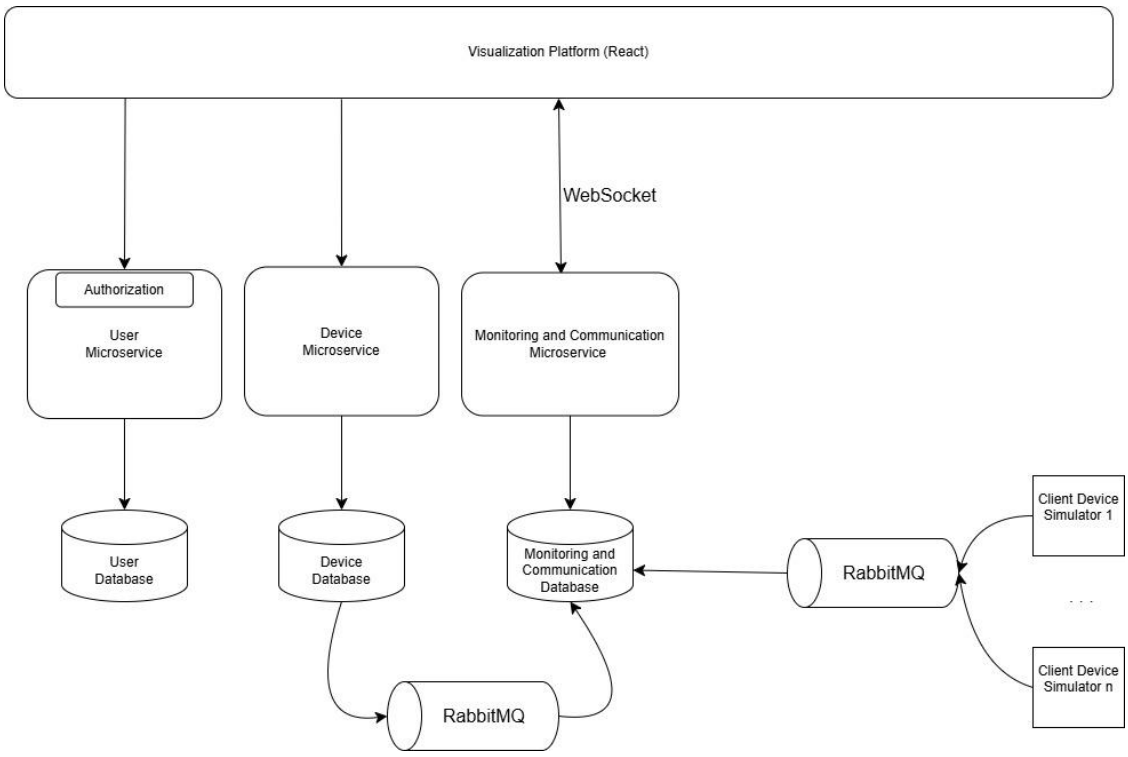
1.2. Notificări:

Utilizatorii sunt notificați în timp real dacă un dispozitiv depășește consumul maxim permis. Microserviciul de Monitorizare și Comunicare procesează măsurătorile primite, iar atunci când consumul orar depășește pragul definit pentru dispozitiv, notificările sunt trimise prin WebSocket către interfața utilizatorului.

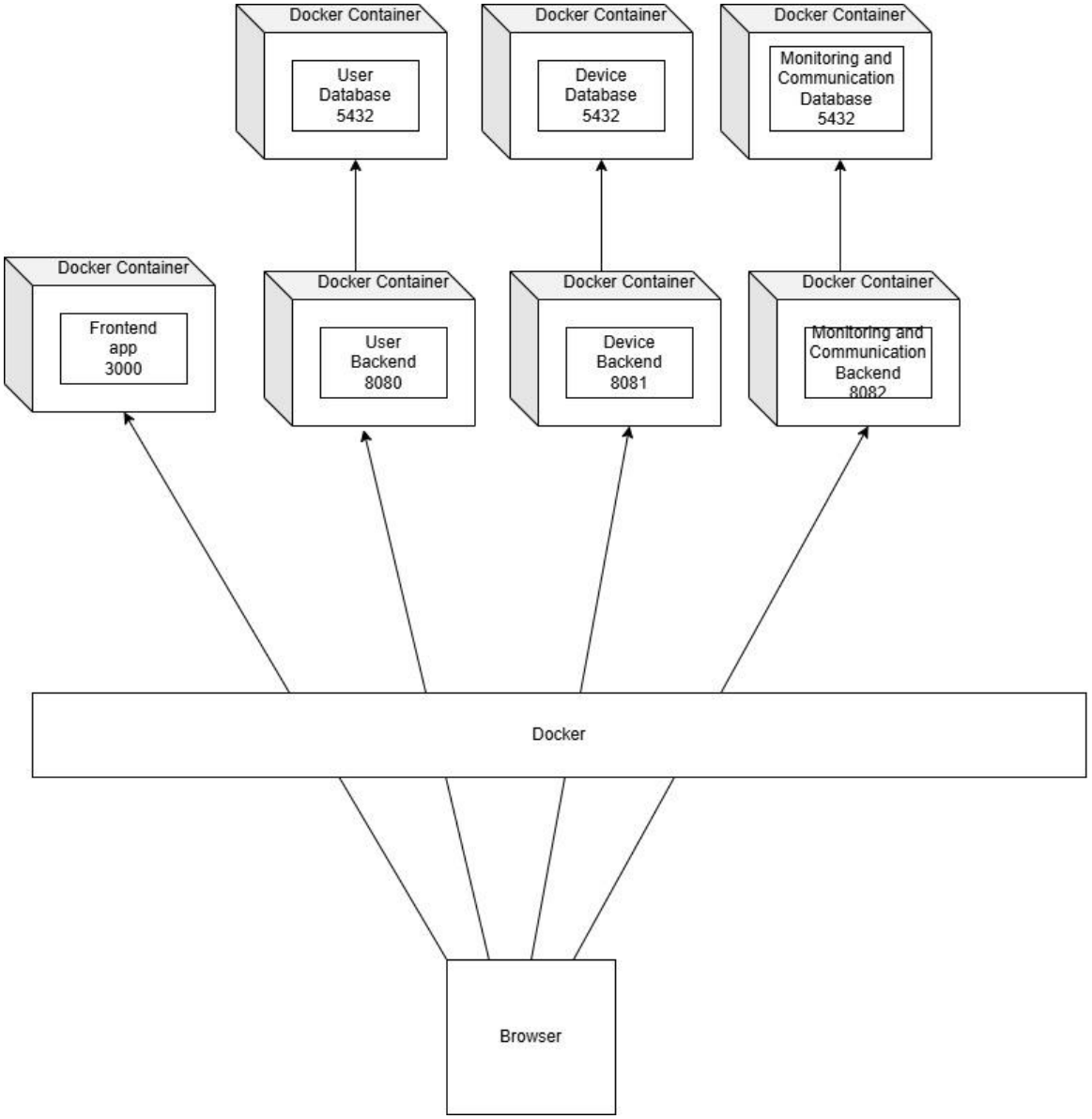
1.3. Istoricul Consumului:

Datele de consum sunt prezentate sub formă de diagrame orare, permițând utilizatorilor să vizualizeze consumul de energie pe zile selectate. Aceste diagrame oferă o reprezentare clară a consumului zilnic, facilitând analiza și monitorizarea energiei pe termen lung.

II. UML Diagram



Docker


Docker Container
User Database 5432
Docker Container
Device Database 5432
Docker Container
Monitoring and Communication Database 5432
Docker Container
Frontend app 3000
Docker Container
User Backend 8080
Docker Container
Device Backend 8081
Docker Container
Monitoring and Communication Backend 8082
Docker
Browser

Load Balancing and Reverse Proxy