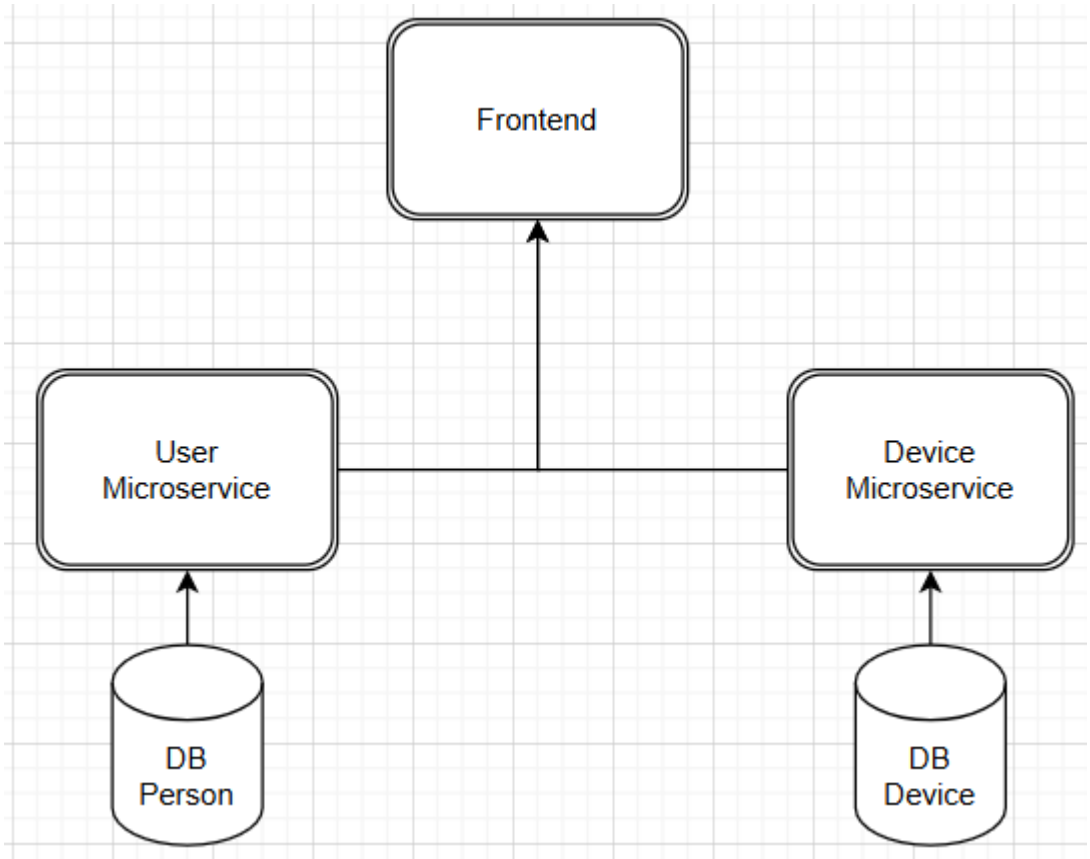
Energy Management System

Contents

1. Introducere

Energy Management System este o aplicație distribuită formată din două microservicii (gestionarea utilizatorilor și a dispozitivelor) și o interfață web pentru utilizatori. Aplicația este destinată administrării conturilor utilizatorilor și a dispozitivelor inteligente de măsurare a energiei asociate acestora. Sistemul include două tipuri de utilizatori: administrator și client. Administratorul poate crea și gestiona conturi de utilizator, dispozitive și asignarea dispozitivelor utilizatorilor, iar clientul poate vizualiza doar dispozitivele asignate.

2. Arhitectura Conceptuală a Sistemului



Sistemul folosește o arhitectură de tip microservicii, cu două microservicii REST (gestionarea utilizatorilor și gestionarea dispozitivelor) și un front-end realizat in React. Fiecare microserviciu este responsabil pentru o funcționalitate specifică și dispune de

o bază de date proprie, cu sincronizarea informațiilor între acestea pentru asocierea utilizatori-dispozitive.

Componente

User Management Microservice: Gestionează informațiile utilizatorilor, inclusiv datele de autentificare și rolurile (administrator sau client).

Device Management Microservice: Gestionează informațiile despre dispozitivele de măsurare a energiei.

Frontend: Interfața de utilizator care permite autentificarea și realizarea operațiilor pentru fiecare rol.

Database: Fiecare microserviciu are propria sa bază de date.

3. Funcționalități API și Frontend

3.1. Microserviciul User Management

API-uri Principale

Autentificare utilizator

- Endpoint: POST /login
- Body: { "username": "admin", "password": "password" }

CRUD Utilizatori (Administrator)

- Creare utilizator: POST /person
- Citire utilizatori: GET /person
- Actualizare utilizator: PUT /person/{userId}
- Ștergere utilizator: DELETE /person/{userId}

API Frontend

- Autentificare și Redirecționare: După autentificare, utilizatorul este redirecționat automat la pagina corespunzătoare rolului său (pagina administratorului pentru administrare, respectiv pagina clientului pentru vizualizarea dispozitivelor).

3.2. Microserviciul Device Management

API-uri Principale

CRUD Dispozitive (Administrator)

- Creare dispozitiv: POST /device
- Citire dispozitive: GET /device
- Device-urile unui utilizator: GET /device/client/{clientId}
- Actualizare dispozitiv: PUT /device/{deviceId}
- Ștergere dispozitiv: DELETE /device/{deviceId}

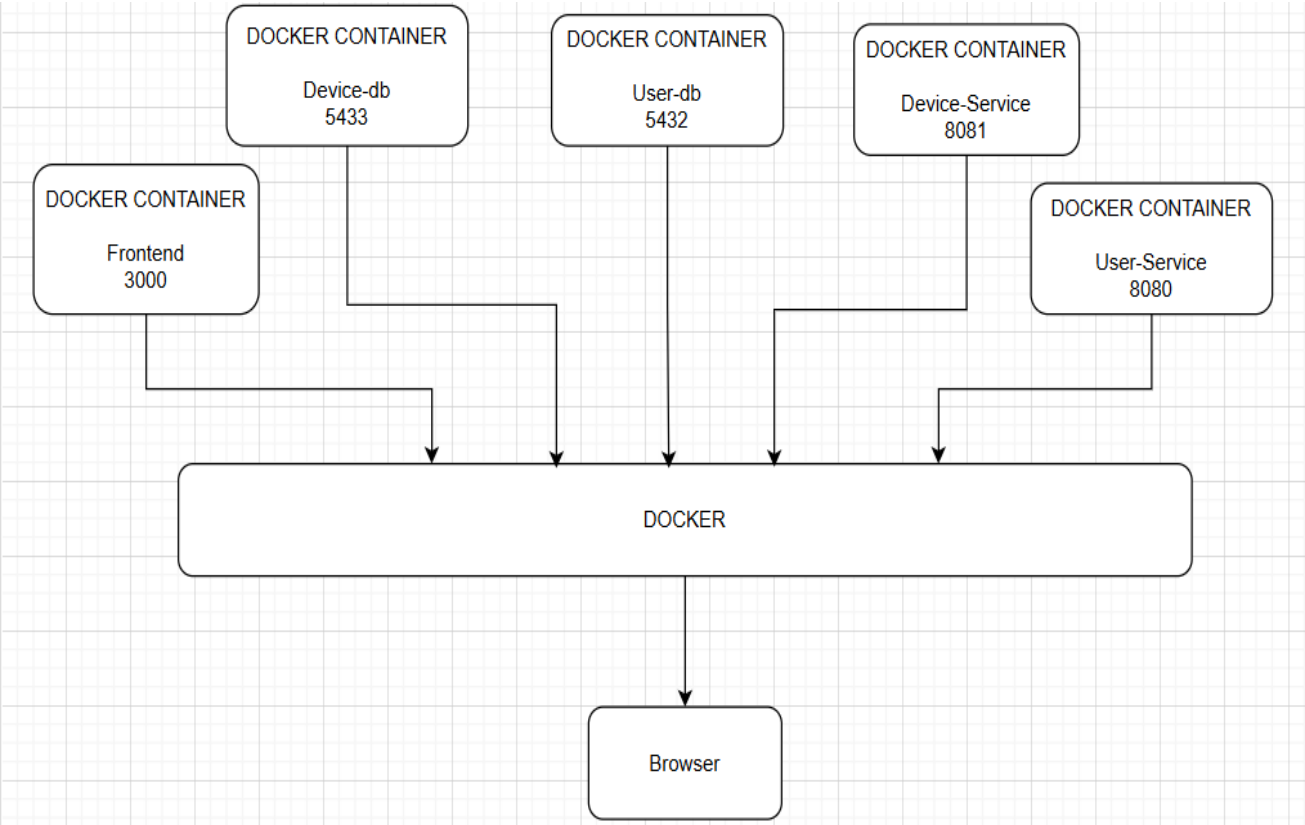
Mapare Utilizator-Dispozitiv (Administrator)

- Body: { "userId": "uuid_user", "deviceId": "uuid_device" }

3.3. Sincronizarea Datelor între Microservicii

Implementăm un mecanism de sincronizare pentru a propaga modificările asupra utilizatorilor între microservicii. Acest mecanism poate foloseste apeluri REST.

4. Diagrama de deployment



3.2. Microserviciul de Monitorizare și Comunicare

Descriere generală

Microserviciul de Monitorizare și Comunicare gestionează colectarea, procesarea și notificarea datelor primite de la dispozitivele inteligente. Sistemul utilizează RabbitMQ pentru comunicare asincronă între componente, asigurând scalabilitate și performanță.

Simulatorul de dispozitive inteligente

Simulatorul este o aplicație desktop care citește periodic valori dintr-un fișier `sensor.csv`. Datele sunt trimise către RabbitMQ sub formă de mesaje JSON care includ un timestamp local, un `device_id` unic și valoarea măsurată. Fiecare instanță a simulatorului are configurat un identificator unic al dispozitivului, asociat unui utilizator.

Brokerul de mesaje RabbitMQ

RabbitMQ este utilizat pentru a transmite datele de la simulatoare către microserviciu. Comunicarea se bazează pe cozi și topicuri, ceea ce permite procesarea eficientă a datelor și sincronizarea între microservicii.

Microserviciul de Monitorizare și Comunicare

Microserviciul include un consumator RabbitMQ care preia mesaje din coadă. Mesajele sunt procesate pentru a calcula consumul total de energie pentru fiecare oră, iar datele rezultate sunt stocate într-o bază de date. Dacă un consum depășește limita maximă definită, microserviciul notifică utilizatorul printr-un server WebSocket.

Sincronizarea datelor între microservicii

Sincronizarea între Microserviciul de Monitorizare și cel de Gestionare a Dispozitivelor se realizează prin topicuri RabbitMQ. Aceasta permite actualizarea automată a informațiilor despre dispozitive în baza de date.

Aplicația client

Aplicația client afișează notificări în timp real și oferă o interfață grafică pentru vizualizarea istoricului consumului de energie. Utilizatorii pot selecta o zi din calendar și vizualiza consumul orar pentru acea zi sub formă de grafice. Notificările apar imediat în interfață prin intermediul conexiunii WebSocket.

Configurarea și rularea sistemului

Configurarea începe cu inițializarea RabbitMQ și configurarea cozilor și topicurilor necesare. Simulatorul este pornit cu un fișier de configurare care setează `device_id`-ul corespunzător. Microserviciul de monitorizare este lansat pentru a procesa datele din coadă, iar aplicația client este deschisă în browser pentru vizualizarea notificărilor și datelor istorice.

Rulare simultană a mai multor simulatoare

Sistemul permite rularea simultană a mai multor simulatoare, fiecare reprezentând un dispozitiv distinct. Datele sunt procesate individual pentru fiecare dispozitiv, iar notificările și graficele sunt afișate în timp real în interfața utilizatorului.

Deployment Diagram:

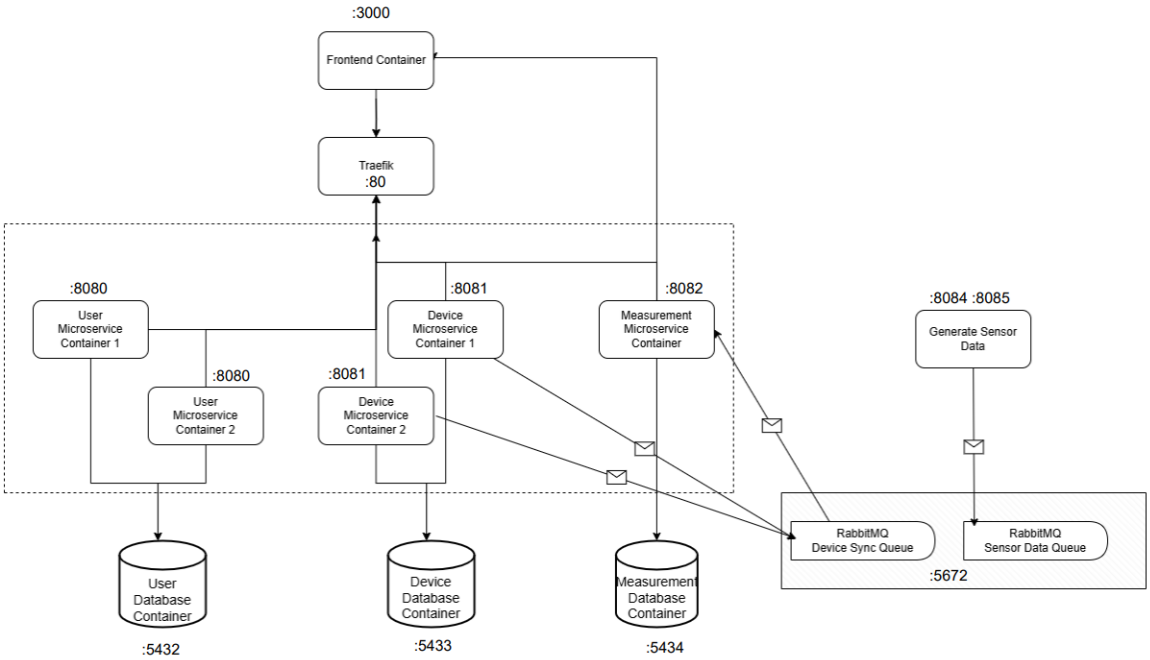
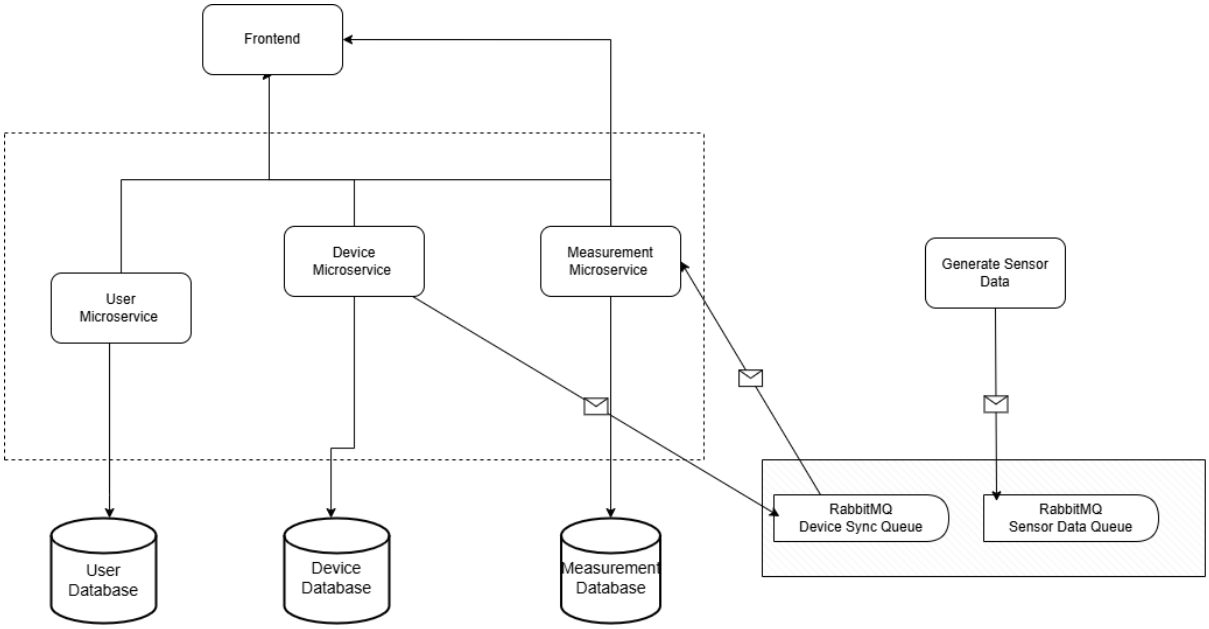


Diagrama de clase



3.3. Chat Microservice și Componenta de Autorizare pentru Sistemul de Management al Energiei

1.1 Descriere generală a arhitecturii distribuite

Sistemul de Management al Energiei utilizează o arhitectură bazată pe microservicii care sunt interconectate prin intermediul unor tehnologii specifice pentru a asigura scalabilitate, securitate și performanță. Aceste microservicii sunt implementate astfel încât să îndeplinească cerințele de comunicare în timp real și de securitate ale utilizatorilor.

Componentele principale ale arhitecturii sunt:

- **Microserviciul de Chat**: Acesta se ocupă de gestionarea comunicării dintre utilizatori și administratorii sistemului. Microserviciul utilizează WebSockets pentru a asigura comunicarea bidirecțională și în timp real între utilizatori și administratori.
    - *Funcționalitate*: Permite trimiterea de mesaje între utilizatori și administratori, notificarea la citirea mesajelor și altele.
- **Componenta de Autorizare**: Oferă acces securizat la sistemul distribuit prin utilizarea Spring Security și OAuth2 JWT pentru autentificare și autorizare. Această componentă este integrată în Microserviciul de Management al Utilizatorilor, permițând autentificarea utilizatorilor și accesul doar pentru aceia autorizați.
    - *Funcționalitate*: Asigură că doar utilizatorii autentificați pot accesa chat-ul și alte microservicii ale sistemului.
- **Microserviciul de Management al Utilizatorilor**: Gestionează înregistrarea și autentificarea utilizatorilor, fiind esențial pentru autentificarea și autorizarea accesului la restul microserviciilor.
    - *Funcționalitate*: Permite gestionarea utilizatorilor și autentificarea acestora în cadrul sistemului.
- **Microservicii de Dispozitive**: Aceste microservicii colectează date de la dispozitivele inteligente și permit utilizatorilor să interacționeze cu aceste date prin interfețe API.

Tehnologii utilizate:

- **WebSockets**: Asigură comunicarea bidirecțională între utilizatori și administratori, garantând mesajele în timp real.
- **Spring Security OAuth2 JWT**: Asigură autentificarea utilizatorilor și autorizarea acestora pentru a accesa diferite servicii din sistem.

- **RabbitMQ**: Permite comunicarea asincronă între microservicii, contribuind la procesarea eficientă a mesajelor.
- **Docker**: Permite containerizarea microserviciilor, ceea ce asigură portabilitatea și ușurința implementării în medii diverse.

Fluxul general de comunicare:

- Utilizatorul trimite un mesaj prin interfața de chat.
- Mesajul este procesat de microserviciul de chat și redirecționat către administratorul sistemului.
- Administratorul poate răspunde în timp real prin WebSocket.
- Notificările sunt trimise utilizatorului și administratorului atunci când un mesaj este citit sau când unul dintre aceștia începe să scrie un mesaj.

1.2 Componentele arhitecturale și interacțiunile dintre acestea

Arhitectura este organizată pe mai multe niveluri, fiecare componentă având un rol clar definit:

- **Front-end (Aplicația web de chat)**: Aceasta permite utilizatorilor să interacționeze cu sistemul printr-o interfață de chat. Utilizatorii pot trimite și primi mesaje, iar interfața notifică utilizatorii atunci când administratorul citește mesajul sau începe să scrie un răspuns.
- **Microserviciul de Chat**: Este responsabil pentru procesarea și gestionarea mesajelor transmise între utilizatori și administratori. Utilizează WebSockets pentru a transmite mesajele în timp real. Acesta este și punctul central în care sunt gestionate notificările legate de citirea mesajelor și scrierea acestora.
- **Componenta de Autorizare**: Integrarea cu Spring Security și OAuth2 JWT asigură că doar utilizatorii autentificați pot accesa chat-ul și alte microservicii. Această componentă gestionează permisiunile de acces și protejează datele sensibile.

Deployment Diagram

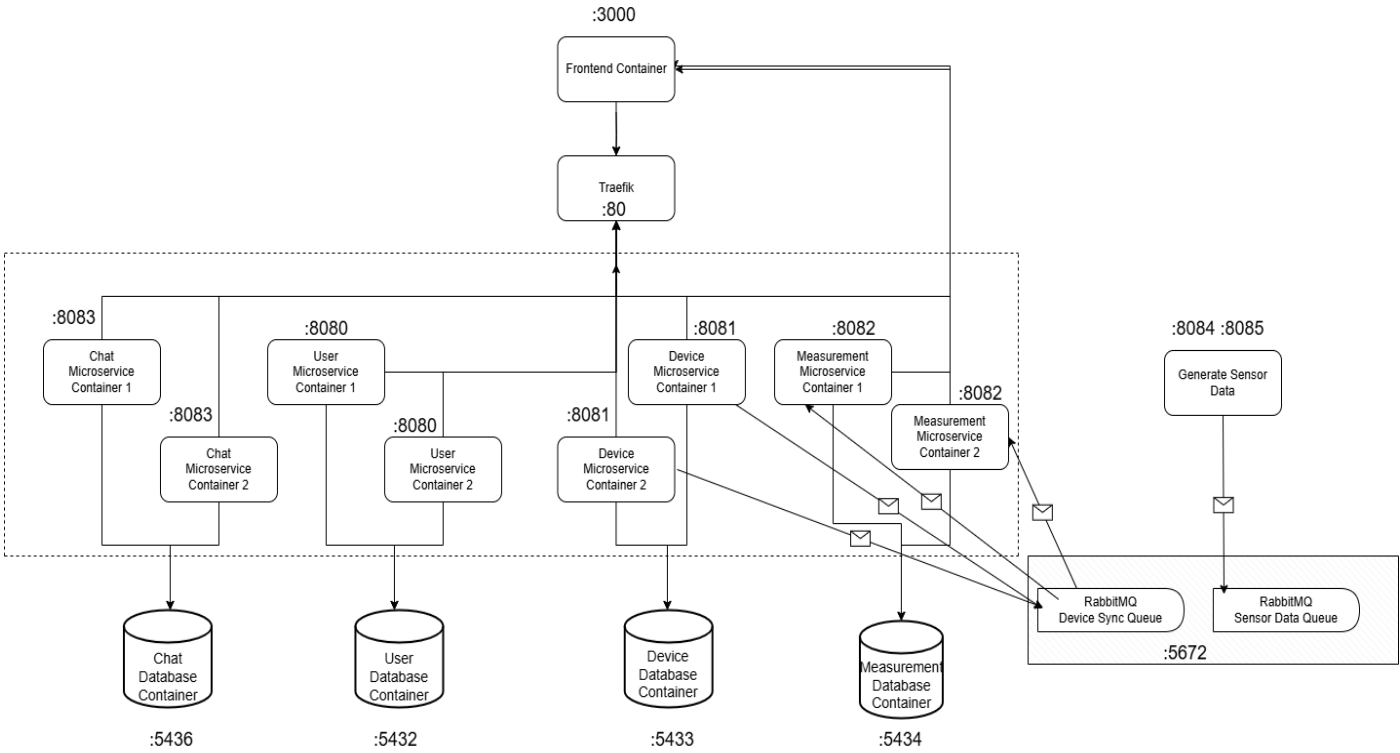


Diagrama de clase