SISTEME DISTRIBUITE

Energy Management System

Moşilă Luciana 30641/2

FACULTATEA DE AUTOMATICA
SI CALCULATOARE
2024

Contents

I. Conceptual architecture

Arhitectura Aplicației

Aplicația este construită pe o arhitectură pe bază de microservicii, fiecare microserviciu având responsabilități clare și comunicând între ele prin RabbitMQ, care funcționează ca un broker de mesaje asincrone. Fiecare microserviciu este independent, având propriile baze de date și funcționalități, dar colaborează printr-un sistem de mesaje pentru a asigura integritatea și performanța întregii aplicații.

Microserviciul de Management al Utilizatorilor: Gestionază autentificarea utilizatorilor și operațiunile CRUD pentru conturi. Folosește JWT pentru autentificare și autorizare, permițând accesul securizat la resursele aplicației. Acest microserviciu integrează componenta de autorizare care protejează accesul utilizatorilor și gestionează drepturile acestora pentru a accesa resursele sistemului.

Microserviciul de Dispozitive: Administrează dispozitivele inteligente, inclusiv operațiunile de adăugare, actualizare și ștergere. Acest microserviciu interacționează cu cel de Monitorizare și Comunicare pentru a actualiza datele în timp real. Prin integrarea cu RabbitMQ, se asigură comunicarea continuă cu alte microservicii.

Microserviciul de Monitorizare și Comunicare: Monitorizează consumul de energie al dispozitivelor și trimite notificări în timp real atunci când sunt depășite pragurile de consum. Datele sunt procesate și stocate în baza de date de monitorizare. Acest microserviciu este esențial pentru gestionarea alertelor și pentru interacțiunea cu utilizatorii prin WebSocket.

Simulatorul de Dispozitive Inteligente: Simulează dispozitivele inteligente, generând date de consum și trimițându-le către RabbitMQ, care sunt preluate de microserviciile de monitorizare și analiza consumului.

Frontend: Interfața utilizatorului, construită în React, permite vizualizarea consumului de energie, managementul dispozitivelor, notificările în timp real și comunicarea prin chat cu administratorii. Chat-ul se realizează prin WebSocket, permițând conversații instantanee între utilizatori și administratori.

Microserviciul de Chat: Permite comunicarea între utilizatori și administratorii sistemului, permițând utilizatorilor să pună întrebări și să primească răspunsuri în timp real. Administratorul poate conversa simultan cu mai mulți utilizatori, iar notificările sunt generate pentru indicarea când un mesaj este citit sau când utilizatorul/administratorul scrie un mesaj. Componenta de chat utilizează WebSocket pentru comunicarea instantanee între utilizator și administrator.

RabbitMQ: Asigură comunicarea asincronă între microservicii, permițând o gestionare eficientă a datelor între simulator, monitorizare și alte componente.

Securitatea Aplicației

Securitatea aplicației este esențială pentru protecția datelor utilizatorilor și pentru controlul accesului la funcționalitățile sensibile ale sistemului. Pentru a garanta că doar utilizatorii autorizați pot accesa resursele protejate, aplicația utilizează Spring Security împreună cu JWT (JSON Web Token).

2.1. Autentificare și Autorizare

Accesul utilizatorilor la aplicație este protejat printr-un sistem de autentificare bazat pe adresa de e-mail și parola. Atunci când un utilizator se autentifică, serverul validează aceste credențiale și, în cazul în care sunt corecte, generează un token JWT. Acest token este folosit ulterior pentru autentificarea cererilor de acces la microserviciile backend. Tokenul JWT este transmis în header-ul cererilor și este validat de fiecare microserviciu care protejează resursele aplicației.

2.2. Roluri și Permisiuni

Aplicația implementează un sistem de roluri care permite definirea diferitelor nivele de acces pentru utilizatori. Există două tipuri de roluri:

- **Administratorii**: Au privilegii extinse, incluzând posibilitatea de a gestiona utilizatori și dispozitive, de a adăuga, modifica și șterge date.
- **Clienții**: Au acces limitat, fiind capabili doar să vizualizeze dispozitivele și consumul de energie asociat conturilor lor.

Rolurile sunt atribuite utilizatorilor la momentul autentificării și permit aplicarea unor restricții la nivel de acces, asigurându-se astfel că utilizatorii nu pot accesa resurse pentru care nu au permisiuni.

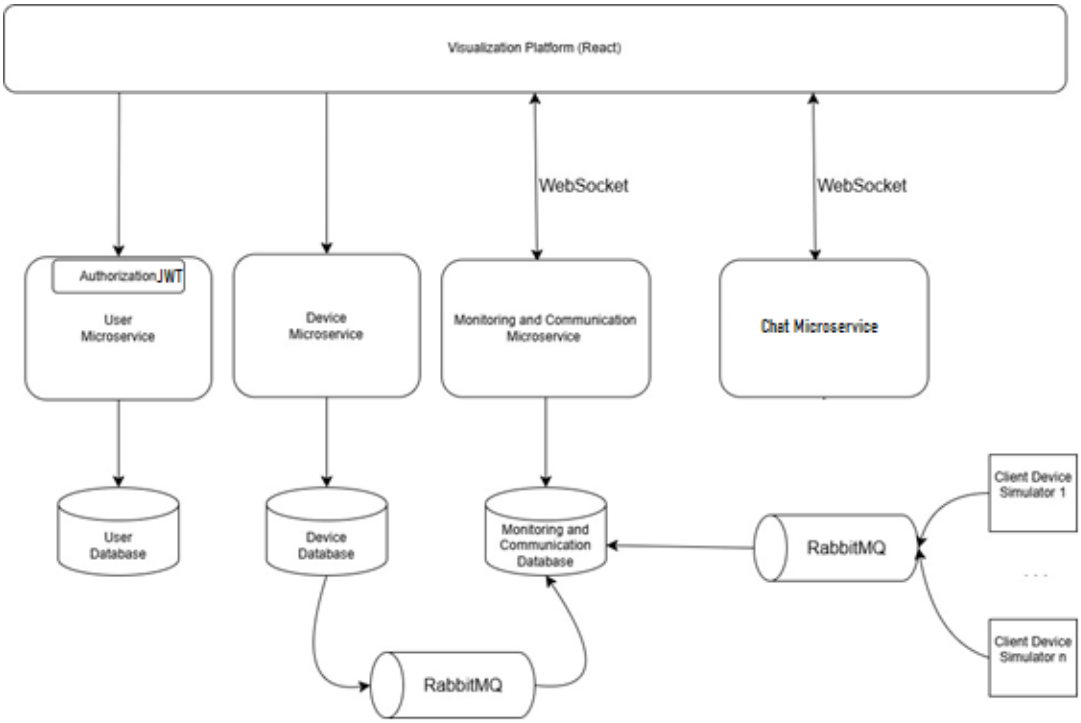
2.3. Protecția Datelor Sensibile

Pentru protecția datelor sensibile, precum parolele utilizatorilor, acestea nu sunt stocate în forma lor necriptată. Parolele sunt criptate utilizând algoritmi de criptare puternici, garantând astfel confidențialitatea acestora și protejarea lor împotriva accesului neautorizat.

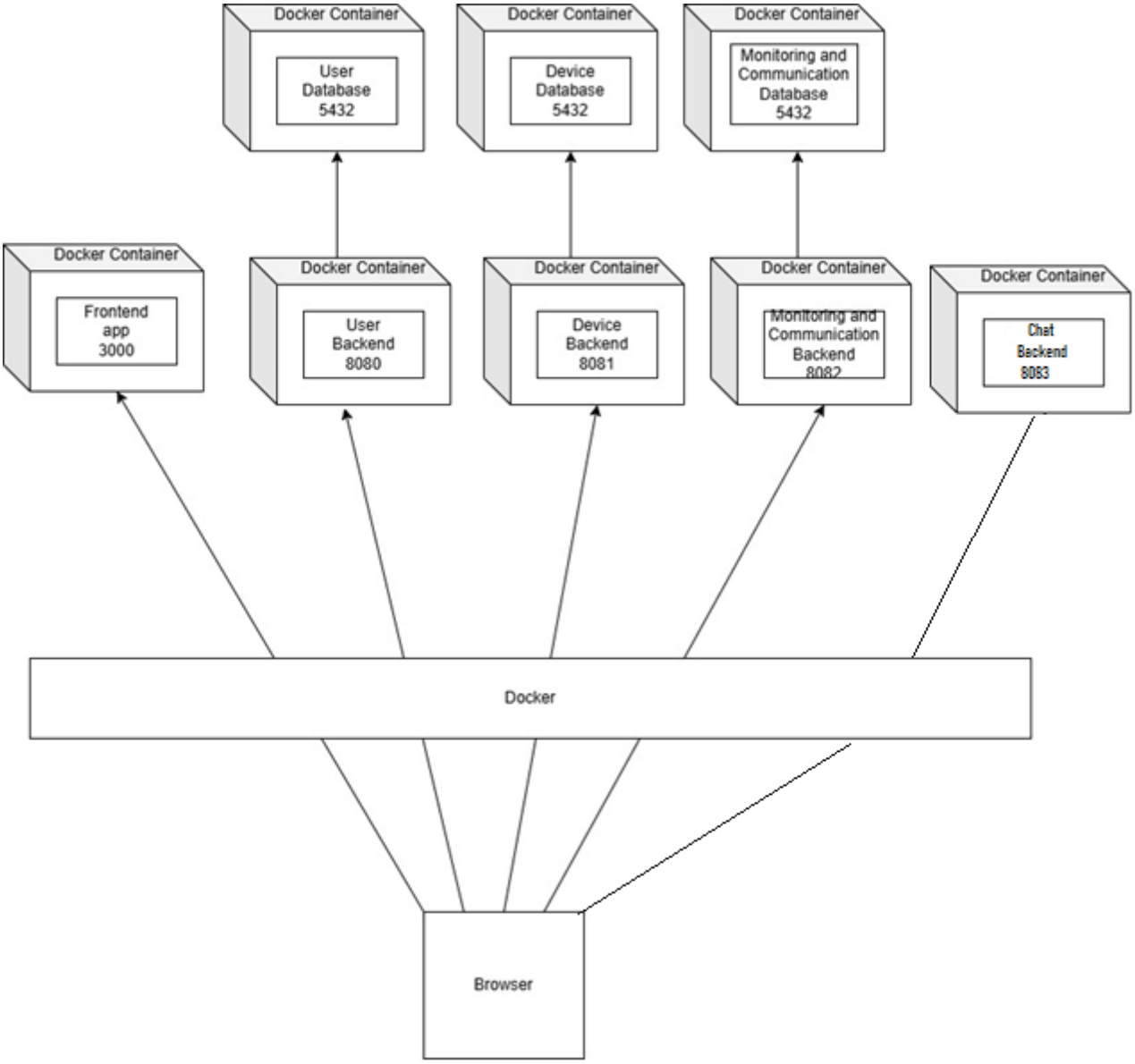
2.4. Validarea Token-urilor

Fiecare cerere care necesită acces la resurse protejate trebuie să includă un token JWT valid. Acest token este verificat de Spring Security pentru a se asigura că utilizatorul este autentificat și că nu a expirat. În cazul unui token invalid sau expirat, cererea va fi respinsă, iar utilizatorului i se va returna un răspuns de tip **401 Unauthorized**.

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 Container
Chat Backend 8083
Docker
Browser

Load Balancing and Reverse Proxy