**DOCUMENTATIE**

**TEMA 2**

NUME STUDENT: Lazăr Vlad

GRUPA: 30241

1. **Cerinta**

Proiectul presupune să dezvoltăm un Sistem de Gestionare a Energiei pentru a administra utilizatorii și dispozitivele inteligente de măsurare a energiei asociate acestora. Sistemul poate fi accesat de două tipuri de utilizatori, după un proces de autentificare: administratori și clienți. Administratorul poate efectua operațiuni CRUD (Creare, Citire, Actualizare, Ștergere) pe conturile de utilizator, pe dispozitivele inteligente de măsurare a energiei și pe asocierea utilizatorilor cu dispozitivele. Clientii pot sa isi vizualizeze dispozitivele proprii, sa vizualizeze consumul dispozitivelor lor si sunt notificati in cazul in care un dispozitiv de-al sau depaseste limita sa de consum pe ora.

* 1. Cerinte functionale:
* Autentificare utilizatori: Utilizatorii se autentifică în sistem și sunt redirecționați către pagina corespunzătoare rolului lor.
* Rolul Administrator/Manager:
  + Operatii CRUD (Creare, Citire, Actualizare, Ștergere) pe utilizatori.
  + Operatii CRUD pe dispozitive.
  + Crearea asocierilor între utilizatori și dispozitive.
* Rolul Utilizator/Client:
  + Un client poate vizualiza pe pagina sa toate dispozitivele asociate.
  + Middleware-ul orientat pe mesaje permite Simulatoarelor de Dispozitive Inteligente de Masurare sa trimita tupluri de date in format JSON.
  + Componenta consumator de mesaje a microserviciului proceseaza fiecare mesaj si notifica asincron aplicatia client folosind WebSocket.
  + Valorile de energie orara vor fi salvate de componenta consumator in baza de date a Monitorizarii.
* Restricții de acces:
  + Utilizatorii aparținând unui anumit rol nu pot accesa paginile asociate altor roluri (de exemplu, după autentificare, nu pot accesa paginile de administrare copiind linkul URL în browser).
  1. Cerinte non-funcționale
* Trei Microservicii:
  + Microserviciu pentru Gestionarea Utilizatorilor
  + Microserviciu pentru Gestionarea Dispozitivelor
  + Microserviciul pentru Monitorizarea si Comunicarea cu dispozitivele
* Securitate: Utilizarea autentificării pentru a restricționa accesul utilizatorilor la paginile de administrare.
  1. Tehnologii utilizate
* REST pentru microservicii: am folosit Java Spring REST pentru a implementa microserviciile.
* Framework-uri JavaScript pentru aplicațiile client: am utilizat framework-ul ReactJS, bazat pe JavaScript pentru dezvoltarea aplicațiilor client.
* Pentru transmiterea valorilor transmise de catre dispozitive si pentru transmiterea datelor despre dispozitive, am utilizat cozi RabbitMQ.
* Pentru comunicarea asincrona dintre microserviciul de monitorizare si comunicare cu dispozitivele am utilizat Websockets.

1. **Arhitectura conceptuala**
   1. Bazele de date

Pentru arhitectura cu doua microservicii am avut nevoie de doua baze de date, cate una pentru fiecare microserviciu.

* + 1. Baza de date pentru utilizatori

Pentru utilizatori am stocat urmatoarele informatii in baza de date:

* Id (id) – UUID
* Username (username) – VARCHAR(255)
* Parola (password) – VARCHAR(255)
* Nume (name) – VARCHAR(255)
* Rol (role) – VARCHAR(255)
  + 1. Baza de date pentru device-uri

Pentru device-uri am stocat urmatoarele informatii in baza de date:

* Id (id) – UUID
* Nume (name) – VARCHAR(255)
* Descriere (descriprion) – VARCHAR(255)
* Adresa (address) – VARCHAR(255)
* Consumul maxim pe ora (max\_hourly\_consumption) – REAL
* Id-ul proprietarului (ownerId) – UUID
  + 1. Baza de date pentru masurarori

Pentru masuratori am stocat urmatoarele informatii in baza de date:

* Id (id) – UUID
* Timestamp (timestamp) – TIMESTAMP
* Id-ul device-ului (deviceId) – UUID
* Valoarea masurata (measurementValue) - FLOAT
  1. Backend

Backend-ul este organizat în trei microservicii: Managementul Utilizatorilor, Managementul Dispozitivelor si Managementul de Monitorizare si Comunicare cu device-urile, fiecare implementat pe o arhitectură pe layere. Proiectele sunt împărțite în următoarele pachete: **Configurations, Controllers**, **DTOs**, **Entities**, **Services** și **Repositories**.

* Configurations (Configurari):
  + Are rolul de a centraliza si gestiona setarile necesare pentru configurarea aplicatiei sau a microserviciilor. Acest pachet asigura o organizare clara si separa configurarile de logica de afaceri, contribuind la flexibilitatea si mentenanta proiectului.
* Controllers (Controlere):
  + Gestionează solicitările HTTP primite, direcționându-le către serviciile adecvate pentru procesare.
  + Acționează ca punct de legătură între client și logica din backend, verificând datele de intrare și oferind răspunsuri în funcție de rezultatele obținute din serviciile backend-ului.
* DTOs (Obiecte de Transfer de Date):
  + Obiectele DTO facilitează transferul structurat al datelor între layere, protejând în același timp structurile interne.
  + Transmit doar datele necesare în solicitări și răspunsuri, contribuind la securitatea și eficiența generală.
* Entities (Entități):
  + Sunt modelele de bază care reflectă structura bazei de date, definind atributele și relațiile fiecărui obiect de date.
  + Sunt concentrate pe reprezentarea datelor și nu includ logica de afaceri.
* Services (Servicii):
  + Aici este plasată logica de afaceri principală, care gestionează regulile aplicației și interacțiunile dintre diferitele entități și layere.
  + Coordonează datele între controlere și depozite și aplică reguli specifice înainte de a manipula datele.
* Repositories (Depozite):
  + Aceste pachete gestionează accesul la date și operațiunile CRUD (Creare, Citire, Actualizare, Ștergere) pentru entități, asigurând separarea accesului la baza de date de logica aplicației.
  + Repositorie-urile permit o schimbare mai facilă a bazei de date și simplifică gestionarea datelor pentru servicii.
    1. Microserviciul de Gestionare a Utilizatorilor
* Acest microserviciu gestionează toate aspectele legate de conturile de utilizator și interacțiunile lor cu sistemul. Operatiile CRUD generale sunt dedicate administratorilor, permițându-le să creeze, actualizeze și să șteargă orice utilizator din sistem.
* In controller am gestionat operatia de autentificare.
* Utilizatorii pot avea unul dintre cele două roluri: client sau administrator. La crearea sau ștergerea unui utilizator, microserviciul inițiază cereri către Microserviciul de Gestionare a Dispozitivelor pentru a menține sincronizarea bazelor de date.
  + 1. Microserviciul de Gestionare a Dispozitivelor
* Acest microserviciu se ocupă de toate aspectele legate de dispozitive și de asocierea lor cu utilizatorii.
* Administratorii au acces complet la operațiunile CRUD pentru toate dispozitivele și gestionează legătura dintre dispozitive și proprietarii acestora.
* Un client își poate vizualiza propriile dispozitive.
  + 1. Microserviciul de Gestionare si Comunicare cu Dispozitivele
* Acest microserviciu gestioneaza toate masuratorile primite de la dispozitivele conectate (printr-o coada RabbitMQ), prelucreaza si stocheaza valorile de energie consumata pe ora in baza de date.
* Daca valoarea consumata depaseste un prag stabilit de utilizator, acesta este notificat, transmisiunea realizandu-se prin Websocket-uri.
  1. Frontend
     1. Pagina de login
* La accesarea aplicatiei, avem pagina de login. Aceasta contine doua campuri corespunzatoare username-ului si parolei. Sub acestea este butonul de submit care va verifica valididatea credentialelor si va redirectiona utilizatorul catre pagina corespunzatoare rolului lui.
  + 1. Pagina de administrator
* Pagina de administrator are in componenta doua tabele: corespunzatoare user-ilor, respectiv device-urilor. Schimbul intre cele doua poate fi realizat din dropdown list-ul din stanga sus.
* In partea superioara a tabelului exista un search-bar, unde putem introduce numele unui device sau utilizator, filtrandu-ni doar pe aceia care corespund.
* In dreapta sus exista butonul ”Add device”, respectiv ”Add user”. Prin actionarea acestor butoane, se deschide un formular cu campuri care trebuiesc completate cu datele utilizatorului, respectiv device-ului pe care vrem sa il adaugam. Dupa completarea formularului, butonul de ”Submit” devine accesibil, doar daca toate campurile sunt completate valid.
* Pentru editarea sau stergerea unui utilizator sau device, dam click pe randul din tabel corespunzator acestuia. Aceasta actiune ne va deschide un formular similar cu cel de la adaugare, insa acesta este completat cu datele clientului/device-ului pe care l-am selectat.
  + Daca dorim sa editam, modificam datele, dupa care apasam butonul de “Edit”
  + Daca dorim sa stergem, apasam butonul de “Delete”
* In coltul din stanga sus exista o pictograma, prin apasarea ei, vom fi redirectionati catre pagina de login.
  + 1. Pagina de client
* Pagina de client are in componenta un tabel, corespunzator device-urilor clientului conectat.
* In partea superioara a tabelului exista un search-bar, unde putem introduce numele unui device, filtrandu-ni doar pe aceia care corespund.
* Imediat sub tabel se afiseaza diagrama de consum a dispozitivului selectat la data aleasa. Daca utilizatorul nu a selectat dispozitivul sau data se va afisa mesajul „No measurements available”.
* Pentru generarea diagramei, utilizatorul trebuie mai intai sa selecteze data din partea de stanga jos a ferestrei, dupa care sa apese pe linia din tabel corespunzatoare dispozitivului dorit.
* In coltul din stanga sus exista o pictograma, prin apasarea ei, vom fi redirectionati catre pagina de login.
  + 1. Securitate
* Pentru asigurarea securitatii aplicatiei, am implementat path-uri securizate (un utilizator nu poate accesa alta pagina decat cea alocata rolului sau prin schimbarea url-ului).
* Pentru securitatea parolelor, transmiterea acestora in momentul logarii este criptata folosind SHA256.

1. **Diagrama de deployment**

Pentru deployment-ul aplicatiei am utilizat docker. Am upload-at fiecare parte a proiectului in cate un container, comunicarea intre ele fiind precizata in fisierul docker-compose.yml

* User application backend (userapplication.localhost)
* Device application backend (deviceapplication.localhost)
* Measurement application backend (measurementapplication.localhost)
* User database (postgres-user.localhost)
* Device database (postgres-device.localhost)
* Measurement database (postgres-measurement.localhost)
* Frontend (frontendapplication.localhost)

A diagram on a white board

Description automatically generated

1. **Manualul utilizatorului**
   1. Descarcarea si pornirea aplicatiei

* Se descarca aplicatia
* Se incarca containerele pe docker ruland comanda in fisierul proiectului:
  + docker-compose up -d –build
* Dupa ce executia s-a terminat cu succes, accesati in browser:
  + http://frontendapplication.localhost
  1. Utilizarea aplicatiei
* Pe pagina de login introduceti credentialele dumneavoastra.
* Pentru rolul de administrator:
  + Se pot vizualiza utilizatorii si device-urile in tabelele corespunzatoare
  + Se pot adauga utilizatori/device-uri „Add User”/ „Add Device”
  + Se pot edita utilizatori/device-uri
    - Selectati linia din tabel corespunzatoare utilizatorului/device-ului de editat si modificati datele din formular dupa care apasati butonul „Edit”
  + Se pot sterge utilizatori/device-uri
    - Selectati linia din tabel corespunzatoare utilizatorului/device-ului de sters apoi apasati butonul „Delete”
* Pentru rolul de client:
  + Se pot vizualiza device-urile asociate clientului logat in tabel
  + Se poate vizualiza graficul de consum al unui dispozitiv
    - Selectati data din partea de stanga jos a ferestrei
    - Selectati linia din tabel corespunzatoare dispozitivului dorit