

**DISTRIBUTED SYSTEMS**

**Assignment 2**

**Asynchronous Communication**

**Sensor Monitoring System and Real Time Notification**

**Online Energy Utility Platform**

Nume: Hitu Octavian

Grupa:30242

Cuprins

1. Specificatiile Proiectului
2. Instrumente utilizate
3. Functionalitati
4. Securitate
5. Diagrama UML
6. Designul architectural
   1. Baza de date
   2. Diagrama de clase
   3. Diagramad e deployment

7. Dezvolari ulterioare

8. Concluzie

9. Bibliografie

**Specificatiile proiectului**

Aplicatia pe care am cret-o ajuta la managerierea utlizatorilor, a dispozitivelor si a masuratorilor dintr-o firma. In aceasta aplicatie exista doua tipuri de utilizatori: administrator si utilizator normal. Cel dintai poate face toate operatiile CRUD pe toti utilizatorii aplicatiei, pe dispositive si pe masuratori, iar cel de-al doilea isi poate vizualiza propriile dispozitive, deoarece fiecarei persoane ii pot fi atribuite una sau mai multe dispozitive. Am adaugat simularea unui client care citeste date dint-un fisier csv si le incarca intr-o coada RabbitMQ. La aceasta coada se conecteaza doi clienti care citesc mesajele si se afiseaza in timp real.

**Instrumente Utilizate**

Backend-ul aplicatiei a fost realizat cu Spring Boot in Intellij IDEA. Aici am folosit ca limbaj de programare java . Nu a fost nimic greu deoarece pe parcursul facultatii am mai implementat aplicatii folosind acest limbaj. Ca si frontend aplicatia a fost realizata in VisualStudio Code cu React. Ca limbaj aici am folosit typescript deoarece sunt familiarizat cu el din alte proiecte si ajuta foarte mult la reducerea erorilor in cee ace priveste tipul obiectelor folosite. Ca baza de date am folosit postgres. A fost prima data cand am folosit acest tip de baza de date, dar fiind una relationala nu a fosts greu de inteles.

La aceasta tema am folosit RabbitMQ , folosind o coada situate in cloud care ajuta la conectarea dintre Sender si receiver.

**Functionalitati**

Aplicatia se deschide cu pagina de login. Aici utilizatorii se pot loga ca si client sau administrator. Ca si client , aplicatie nu este foarte permisiva, deoarece el isi poate vizualiza doar dispozitivele care ii sunt atribuite.

Ca si administrator aplicatia ii permite mult mai multe functionalitati. Dupa logare el este trimis in pagina de utilizatori . Aici el poate vizualiza , adauga, sterge, si modificate toti utilizatorii aplicatiei. Adminstratorul dispune de un header cu ajutorul caruia poate naviga printre pagini. In pagina de dispozitive el poate face aceleasi fuctionalitati ca la utiliztaori, dar in momentul in care adauga un nou dispozitiv el trebuie sa i-l atribuie unui utilizator. La masuratori este la fel ca la dispozitive. In partea din dreapta sus se gaseste butonul de log out care trimite utilizatorul la pagina de login.

Clasa sender din noul proiect create va citi din csv datele. Va lua fiecare linie , o dace un string si o fa trimite la coada. Fiecare masuratoare se se trimite la o secunda . Acest mesaj se primeste de catre Receiver dni aplicatie si se afiseaza in consola.

**Securitate**

Prima interactiune a utilizatorului cu aplicatia este pagina de logare. Daca nu se efectueaza logarea , atunci niciun utilizator nu poate avea access la nicio alta pagina. Odata cu efectuarea logarii in functie de tipul de utilizator clientul va fi directionat spre pagina de client, iar adminsitartorul spre pagina de administrator. Daca clientul incearac sa acceseze din URL pagina administratorului acesta va fi directionat spre pagina sa , iar daca acest luvru nu merge si reuseste sa ajunga nu se va afisa nimic. Acest lucru este identic si din partea administratorului.

**Diagrama UML Use Case**

Diagram

Description automatically generated

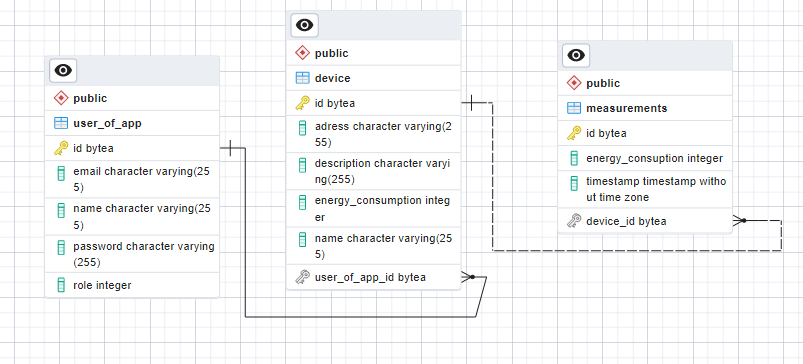
Diagram

Description automatically generated

**Designul architectural**

Niciun proiect nu poate incepe fara un design structural bun. Acesta ne ajuta in dezvoltarea aplicatiei, deoarece noi in design ne stabilim toate tipurile de funtionalitati si ce componente avem nevoie pentru a ajunge la produsul final. Programatorii respectand-ul vor avea un parcurs mult mai usor in implementarea aplicatiei.

**Baza de date**



Ca baza de date am folsoit Postgres. In baza de date a aplicatiei vom gasi 3 tabele.

Prima tabela este cea de utilzatori. Care contine ca si coloane nume, email, parola si rol. Coloana de rol m-a ajutat la Securitate deoarece am reusit sa fac directionarea intre pagini si restrictionarile utilizatorilor.

Graphical user interface, text, application, email

Description automatically generated In cea de-a doua tabela se regasesc device-urile care pe langa coloanele sale specifice se va regasi un field care continue o entitate de utilizator, deoarece fiecare device are un utilzator. Intre aceste doua tabele exista o relatie de one-to-many. Table

Description automatically generated

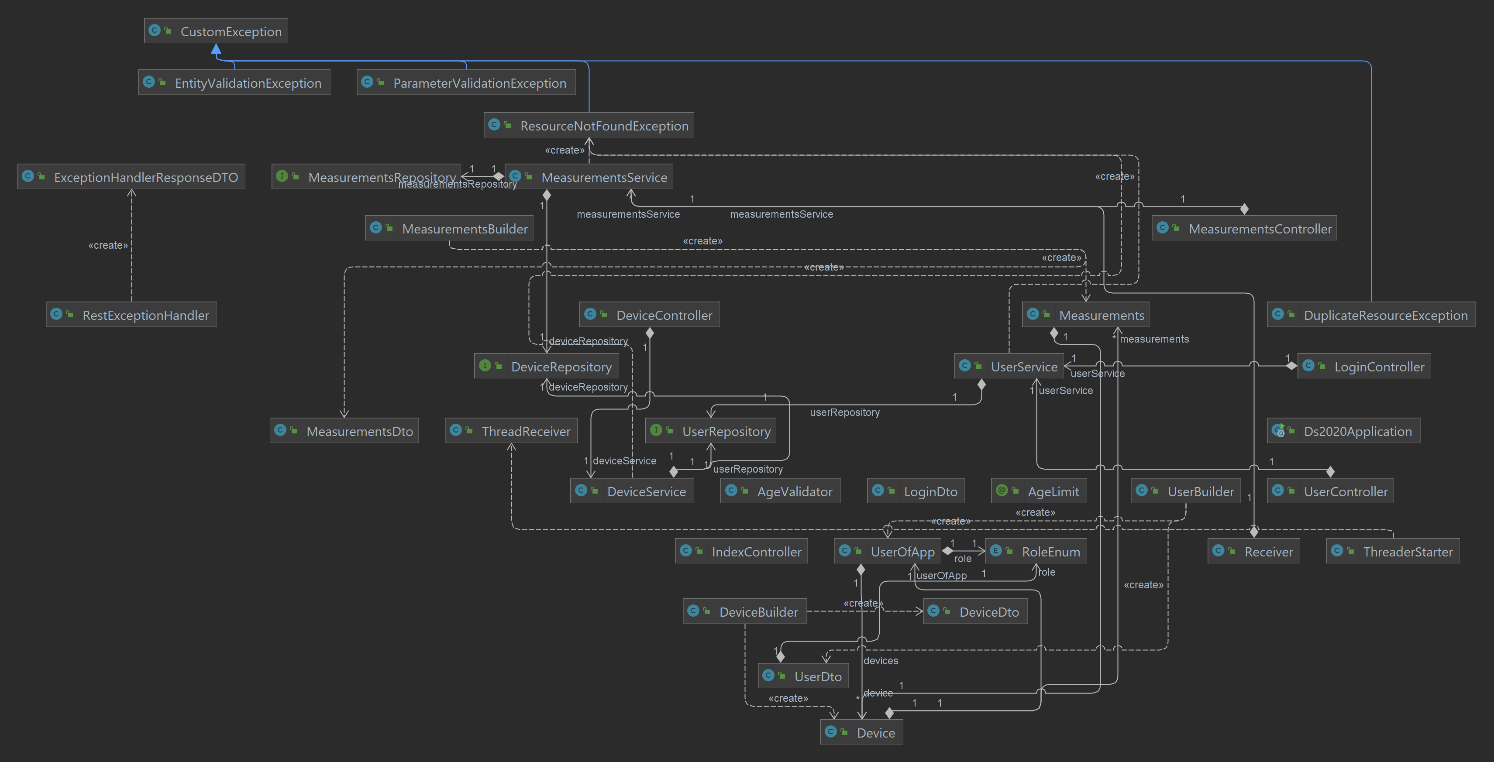
Acelasi tip de relatie se gaseste si intre tabela de masuratori si cea device-uri. Fiecare masuratoare are o entitate de dispozitiv pentru a se face legatura. Daca din baza de date se va sterge un utilizator, atunci se va intampla o cascada , iar toate dispozitivele atribuite lui vor fi sterge ,iar cu toate masuratorile se va intampla la fel.

Table

Description automatically generated

**Diagrama de clase**

Aceasta diagrama de clase este generate de Intellij. Aici se pot gasi toate clasele si legaturile dintre ele.



**Dezvolatari ulterioare**

Ca si imbunatatiri ale aplicatiei se poate implementa o pagina de inregstrare. Logarea si securitatea pot fi mult imbunatatite. Aplicatia ar putea fi mult mai buna daca clientii ar putea sa se inregistreze. Designul aplicatiei ar putea fi modificat , deoarece el a fost facut dupa placul meu, iar e lar trebui sa fie pentru clienti.

**Concluzie**

Din partea mea cea mai dificila parte a fost cea de deploy. Desi a fost unul local , fiind prima data mi s-a parut greu. Legat de frontend si backend nu mi s-a parut nimic complicat deoarece am mai create aplicatii.