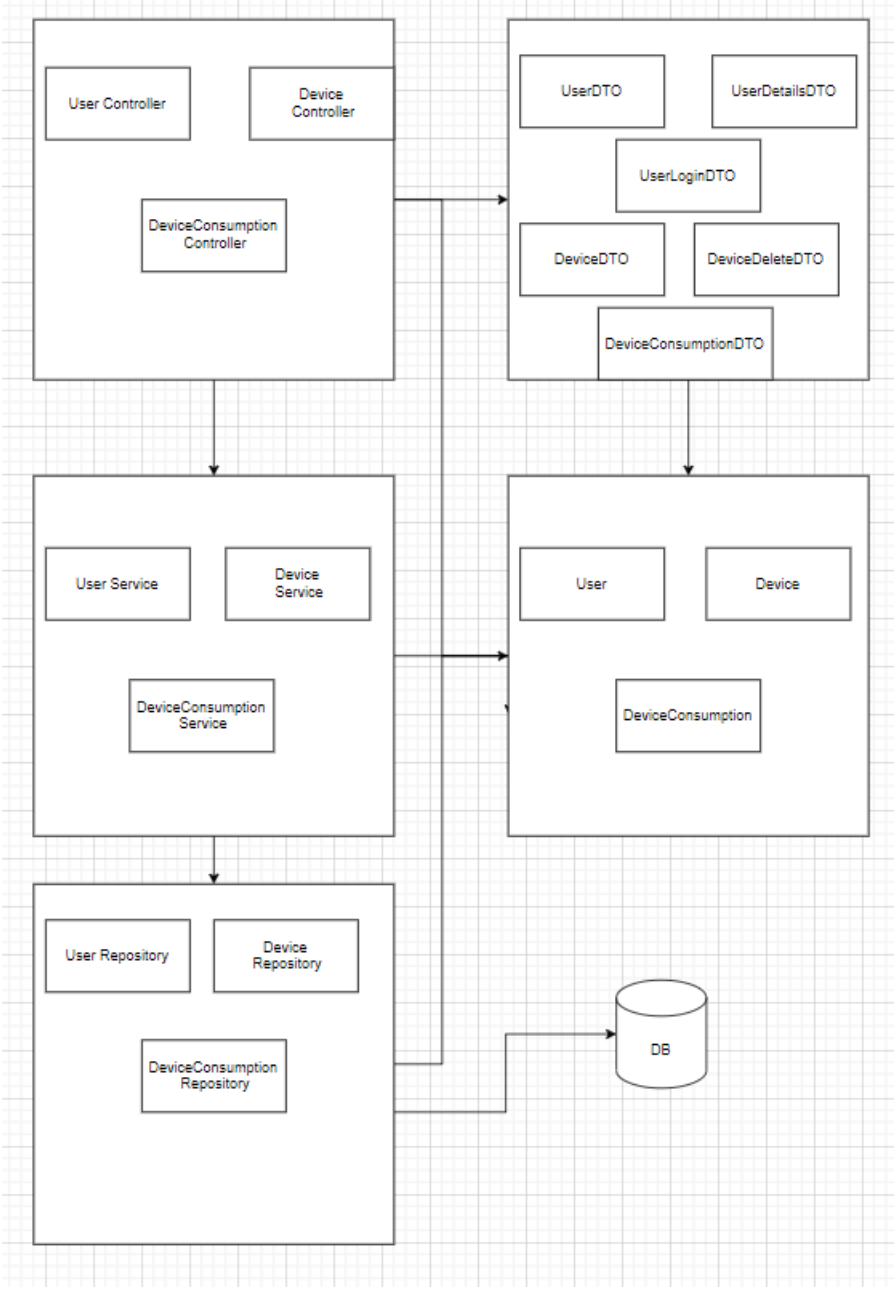
Documentatie Tema1

Tarta Manuel Vasile

Grupa 30641

1. Arhitectura Conceptuala



La aceasta aplicatie am 3 controllere, UserController in care am requesturile HTTP de vizualizare a userilor, adaugare, update si stergere. Acestea apeleaza metodele necesare din service, si returneaza fie o lista de useri, fie id-ul userului care a fost sters de exemplu. In cazul in care operatia nu se poate efectua se returneaza o exceptie care poate fi vizualizata pe frontend. DeviceController merge exact pe aceleasi idei, doar ca pe tabela de devices, iar DeviceConsumptionController are doar un GET in functie de numele device-ului pentru care dorim

Si la service-uri, am tot 3, pentru user, device si device consumption. Fiecare metoda din service are o mica logica proprie, de exemplu gaseste device-ul dupa nume si dupa face o alta operatie din repository. Tot din services se si arunca exceptiile cand este nevoie. In services realizam si conversia intre DTOs si entities (fie primim un dto si trimitem la repository un entity, fie primim un entity si returnam spre controller un DTO.

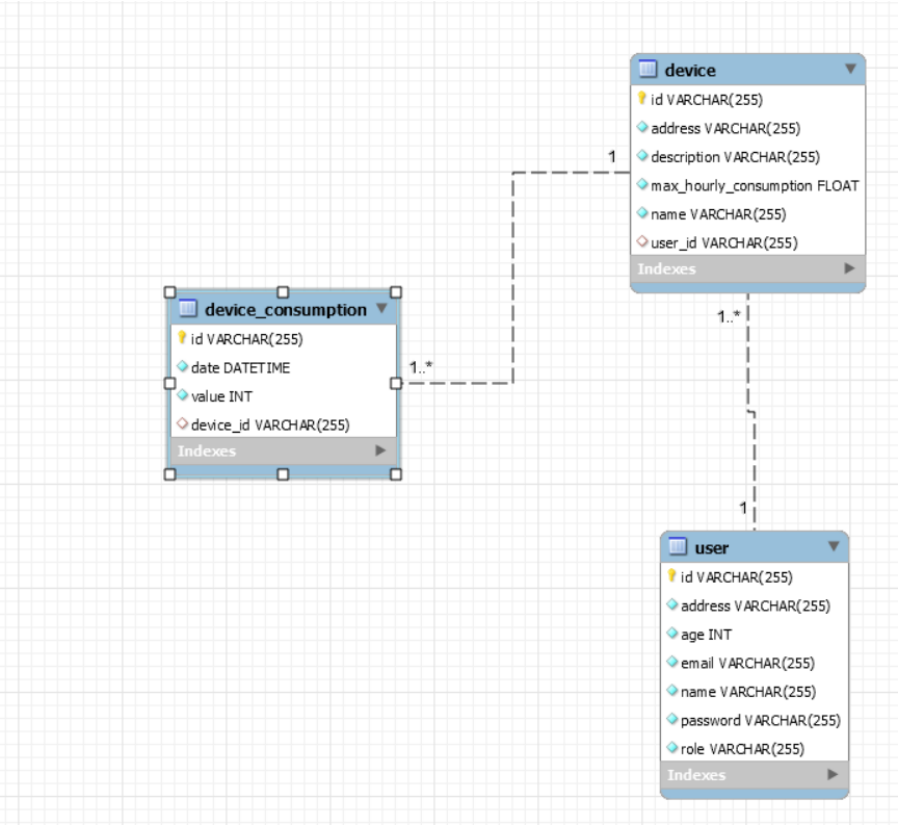
Ca DTO-uri, la User am un DTO pentru vizualizare, unul pentru POST la operatia de Login. La Device am un DTO pentru vizualizare si unul pentru Delete in care am doar numele device-ului pe care vreau sa il sterg.

In Builder am conversii din entity in DTO si din DTO in entity pentru toate combinatiile posibile.

De asemenea, am entities pentru fiecare tabela. Fiecare entity este o interfata care extinde JpaRepository. Asa avem predefinite operatiile basic pe acea tabela. Am mai facut si unele query-uri explicite, cum ar fi sa gasesc device-ul in functie de nume si id-ul userului.

Aplicatia de frontend obtine informatiile de care are nevoie prin intermediul requesturilor HTTP. Are nevoie de un endpoint, care este locul unde poate accesa metodele de la un anumit controller (de exemplu /user). Primeste inapoi un result care contine o lista, un element, un ID, orice returneaza acea metoda din controller, si un status prin care putem verifica daca operatia s-a realizat cu succes sau nu.

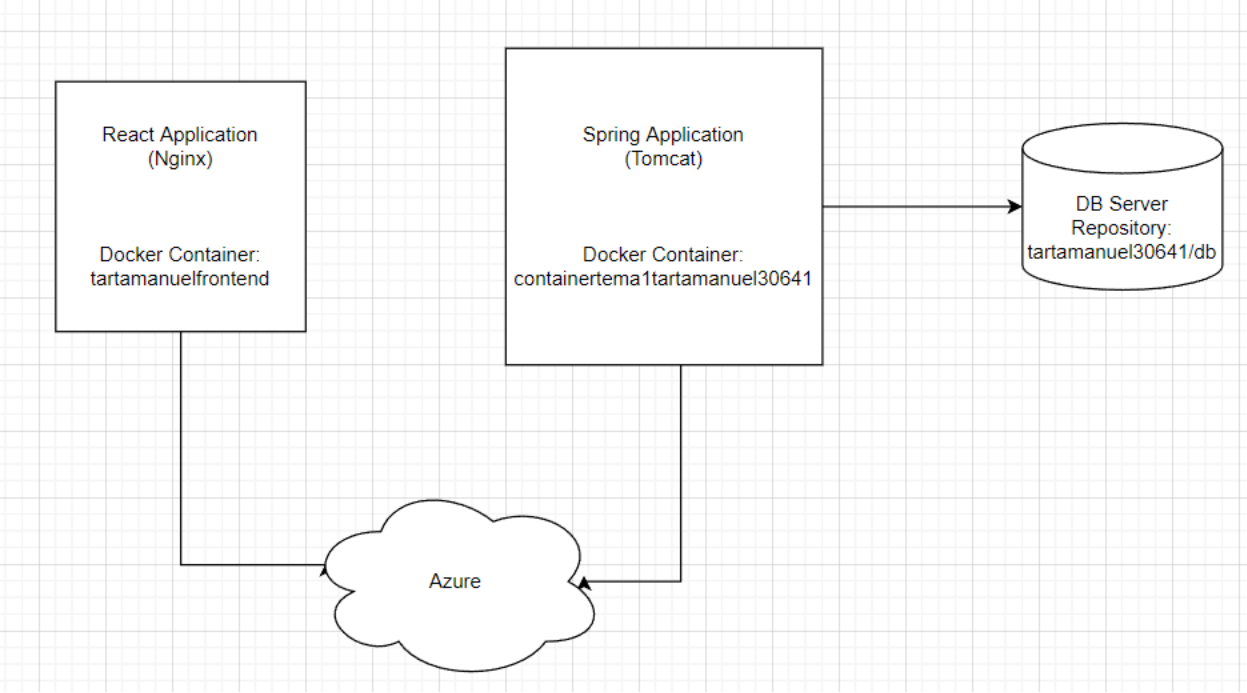
2. Design-ul bazei de date



In baza de date am 3 tabele: una care stocheaza un user care este definit prin adresa, varsta, email, parola, nume, si rol (client sau admin), tabela device care este intr-o relatie many-to-one cu tabela user (un user poate avea mai multe devices) si inca o tabela device_consumption in care stochez consumurile unui anumit device, deci tot o relatie many-to-one uc device.

Astfel, un user isi va putea vedea consumurile pentru fiecare device pe care il detine, putand fi stocate consumuri pe mai multe ore, din mai multe zile. Un anumit consum este definit prin data si ora la care a fost inregistrat, valoarea si device-ul pentru care a fost inregistrat. Consumul nu trebuie sa depaseasca valoarea maxima pentru acel device.

3. Deployment Diagram



Am facut deploy pe Azure atat pentru aplicatia de backend, cat si de frontend. Aplicatia de backend este stocata impreuna cu baza de date in containerul containertema1tartamanuel30641. La fiecare repornire a containerului si la fiecare operatie de release, datele din baza de date sunt sterse, si primim o alta adresa de IP pentru accesul pe web. Pentru a accesa aplicatia de backend de pe cloud, vom folosi acea adresa de ip, plus portul 8080.

Aplicatia de frontend se afla in containerul tartamanuelfrontend si poate fi accesata pe portul 3000, cu o adresa de ip, ca si la backend.

Pentru ca aplicatia sa functioneze, trebuie doar ca cele 2 containere sa fie pornite si sa cunoastem adresele ip care ne-au fost acordate.