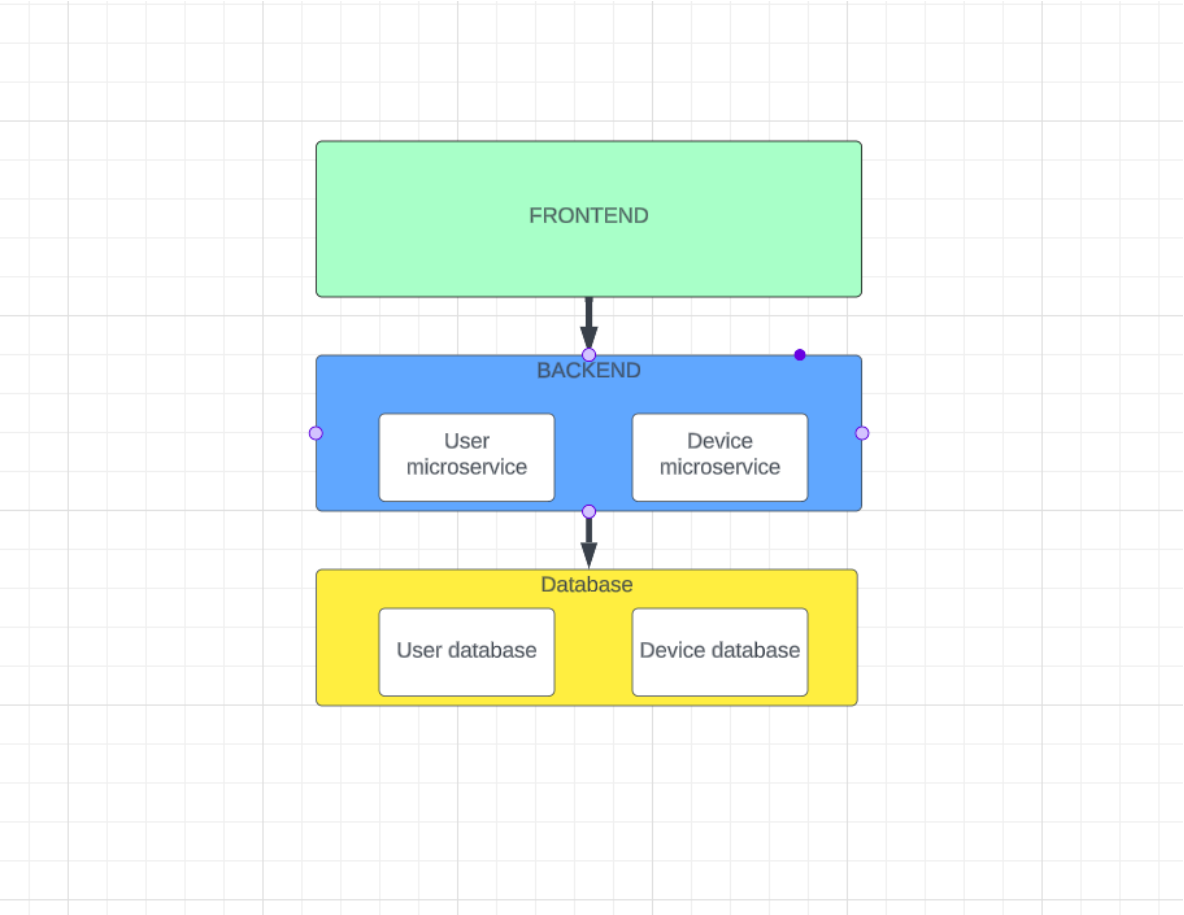
Energy management system

Beiusanu Horia

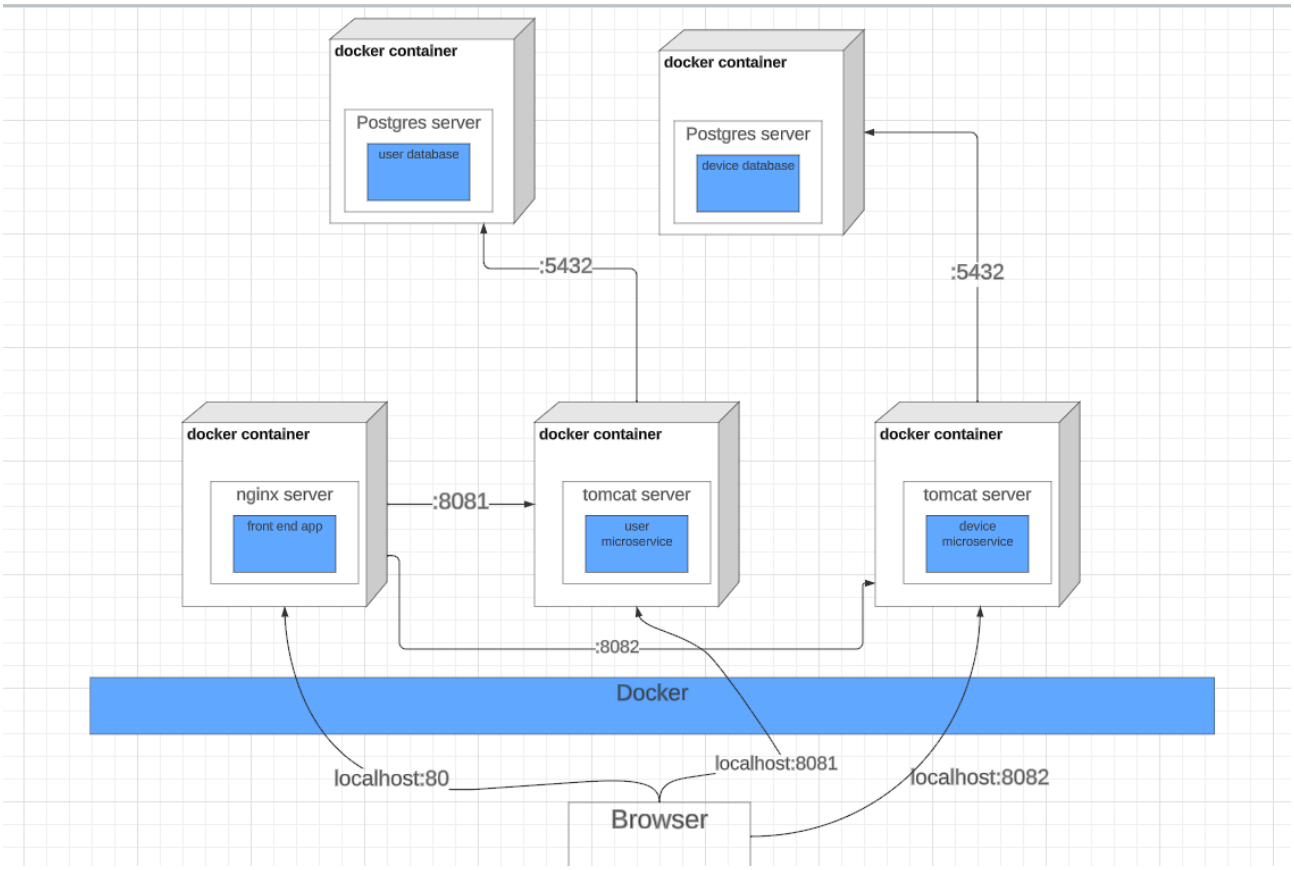
Arhitectura 3-layer



Aplicatia web foloseste arhitectura 3-layer, daca ar fi sa o privim abstractizat top-view. Mai in detaliu, backendul foloseste o arhitectura pe microservicii, mai exact din 2: user si device management. Microserviciile comunica intre ele prin HTTP si produc side-effects in bazele de date. Fiecare microserviciu are la randul lui cate o baza de date si comunica strict doar cu cea asignata lui. Aceasta arhitectura pe microservicii ne permite sa decuplam functionalitatea, si sa permitem echipe diferite sa lucreze si sa evolueze in mare parte indepependent. De asemenea permita ca fiecare microserviciu sa fie implementat in orice fel de tehnologie, neexistand nicio cuplare intre cele folosite de catre servicii diferite. Drept comparatie, arhitectura monolitica ofera initial o mare viteza de development si o performanta mai ridicata, dar aceste degradeaza semnificativ odata cu evolutia proiectului. Aceasta decuplare permite development independent, dar si o scalare mult mai eficienta, fiecare serviciu putand fi scalat independent, in functie de nevoie, in special prin tehnologii de orchestrate precum Kubernetes.

Sarcina frontendului de a gasi mereu ip-ul si portul corespunzator pentru un anumit request, s-ar putea simplifica prin introducerea unui gateway intre frontened si backend.

Diagrama de deployment



Pentru deployment am folosit tehnologia de containerizare Docker. Din browser serviciile se pot accesa prin port-urile externe ale containerelor, dar containerele intre ele vor comunica prin porturile lor interne. Fiecare micro serviciu detine o imagine proprie si se instatiaza intr-un container. Pentru fiecare microserviciu exista si un container care se ocupa cu baza de date corespunzatoare. Toate aceste containere sunt configurate sa comunice printr-un network extern care trebuie creat inainte de pornire. Acesti pasi sunt descrisi in sectiunea ‘pasi pentru build’. In acest network, containerele se adreseaza unul altuia prin numele containerului si prin portul expus de catre acestea. Serverul DNS integrat in docker preia responsabilitatea de a mapa numele containerului la ipul adevarat al acestuia.

Pasii pentru build:

Configurare initiala: docker network create mynetwork

Pentru microservicii:

Din root-ul proiectului, se executa:

1) ./gradlew clean build –x test –parallel
2) Docker build –t [user-app/device-app] .

Pentru frontend:

Din root-ul proiectului, se executa

1) npm run build

2) docker build –t energy-fe .

Pasii pentru executie:

Pentru a porni toate containerele deodata, din root-ul microserviciului se executa comanda:

docker-compose up --build