FACULTATEA: Automatică și Calculatoare

SPECIALIZAREA: Calculatoare Romana

DISCIPLINA: Sisteme Distribuite

ASSIGNMENT 3:

Remote Procedure Call (RPC)

Medication Dispenser

Îndrumător de laborator:

Claudia-Daniela Pop

Student:

Filip-Dud Mihai-Bogdan

UNIVERSITATEA TEHNICĂ DIN CLUJ-NAPOCA

1. Specificația proiectului

Aplicația are a scop implementarea unei platforme web ce va administra activitățile realizate de doctori, asistenți medicali și pacienți. Această aplicație va putea fi accesată de toate cele 3 tipuri de utilizatori, oferind pentru fiecare dintre aceștia o experiență diferită în urma unui proces de login.

Doctorul poate efectua operații de tip CRUD pe rolurile de pacienți și asistenți, CRUD pe lista de medicamente, să adauge rețete pacienților și să atribuie un asistent către un pacient.

Asistentul își poate vedea detaliile contului și, de asemenea, lista de pecienți pe care îi are în îngrijire.

Pacienții pot vedea pe pagina aferentă lor detalii personale, cât și lista de prescripții medicale pe care le au primite de la doctor.

Securitatea aplicației prevede ca niciun rol să nu poată vedea pagina altui rol fără a fi autentificat în prealabil cu un cont aferent rolului respectiv.

Pentru a implementa un sistem de monitorizare a pacienților, se va alcătui o listă a activităților desfășurate de fiecare pacient în timpul zilei care vor fi trimise printr-o coadă către aplicația principală pentru a fi prelucrate, iar în caz de o activitate suspectă, aplicația va avertiza caregiverul aferent pacientului.

Assignmentul 3 vine în completare și adaugă funcționalitatea de a avea un dispenser de medicamente inteligent. Astfel pacientul va avea o aplicație local unde va putea vedea ce medicamente trebuie să ia în ziua respectivă și într-un anumit interval orar. Dacă intervalul orar trece și nu ia pastila, butonul de take devine inactiv și va fi stocat în baza de date raspunsul corespunzător. Apelul de la aplicația locală la backend se va face prin RPC. Astfel aplicațiile împărtășesc o arhitectură comună, precum și o interfață, a cărei implementare este realizată doar pe partea de server, clientul știind doar că metodele există și pot fi folosite.

2. Arhitectura conceptuală a sistemului distribuit

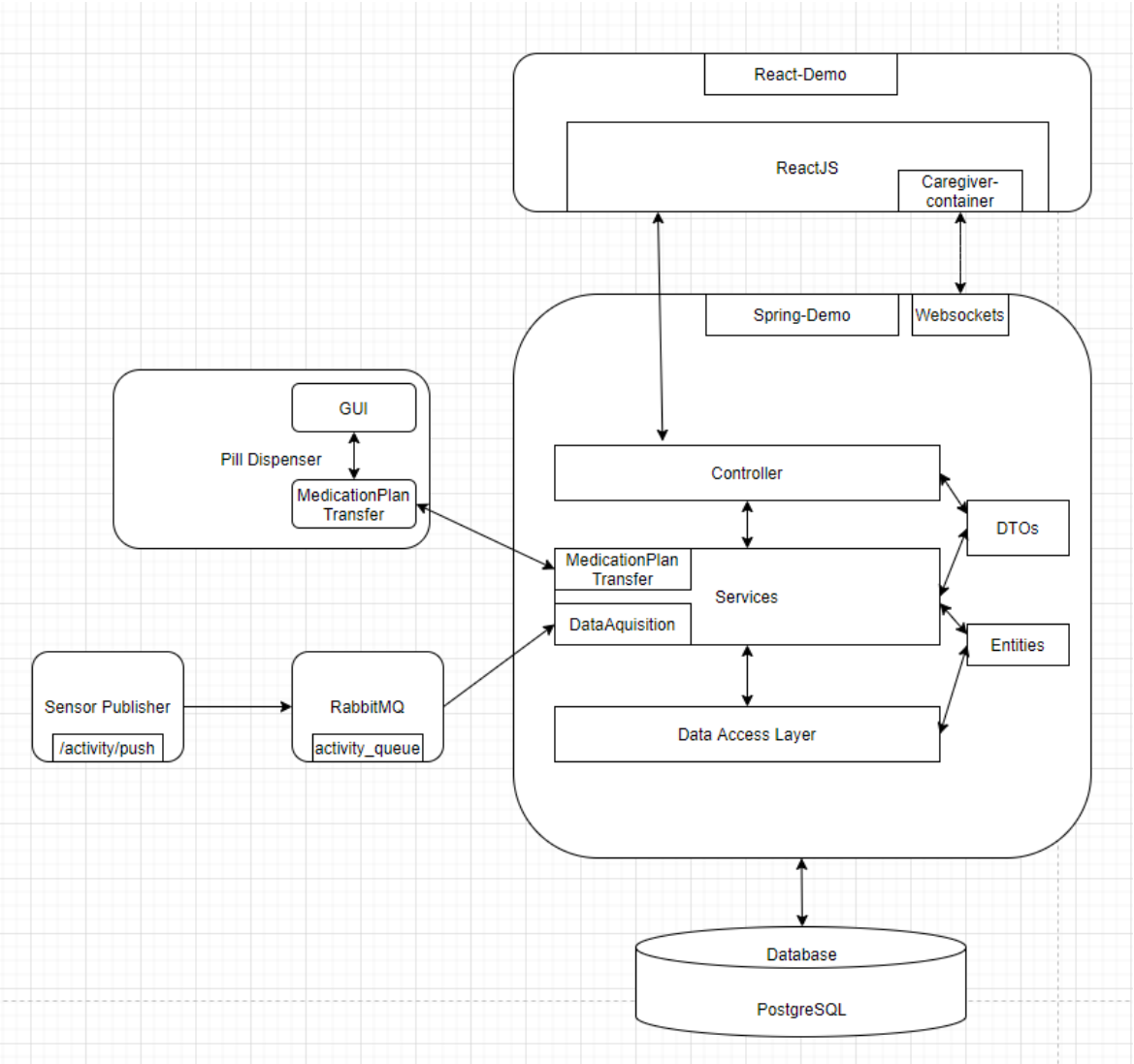
Aplicația este structurată pe 3 niveluri principale:

- Front-end-ul: reprezintă unealta de interecțiune cu utilizatorul a aplicației. Această componentă a fost implementată cu ajutorul framework-ului ReactJs ce pune la dispoziție o multitudine de unelte pentru crearea de pagini web.
- Back-end-ul: această componemtă se ocupă cu primirea datelor de la front-end prin intermediul REST-API și procesarea lor pentru a putea fi stocate mai apoi într-o bază de date cu ajutorul JPA Repository. Pentru implementarea sa, am utilizat framework-ul Spring-Boot ce rulează aplicația pe un server Tomcat integrat.
- Baza de date: aceasta stochează toate datele necesare sistemului primite de la back-end și le pune la dispoziție acestuia atunci când are nevoie de ele. Am utilizat PostgreSQL ca și platformă de gestiune a bazei de date relațională.

La acestea, se adaugă o aplicație terță pentru a simula recepționarea datelor de la senzori și trimiterea acestora prin intermediul unor mesaje manageriate de o coadă de activități implementată cu RabbitMQ. Această coadă va fi descărcată în aplicația de backend unde pentru fiecare activitate se verifică îndeplinirea condițiilor

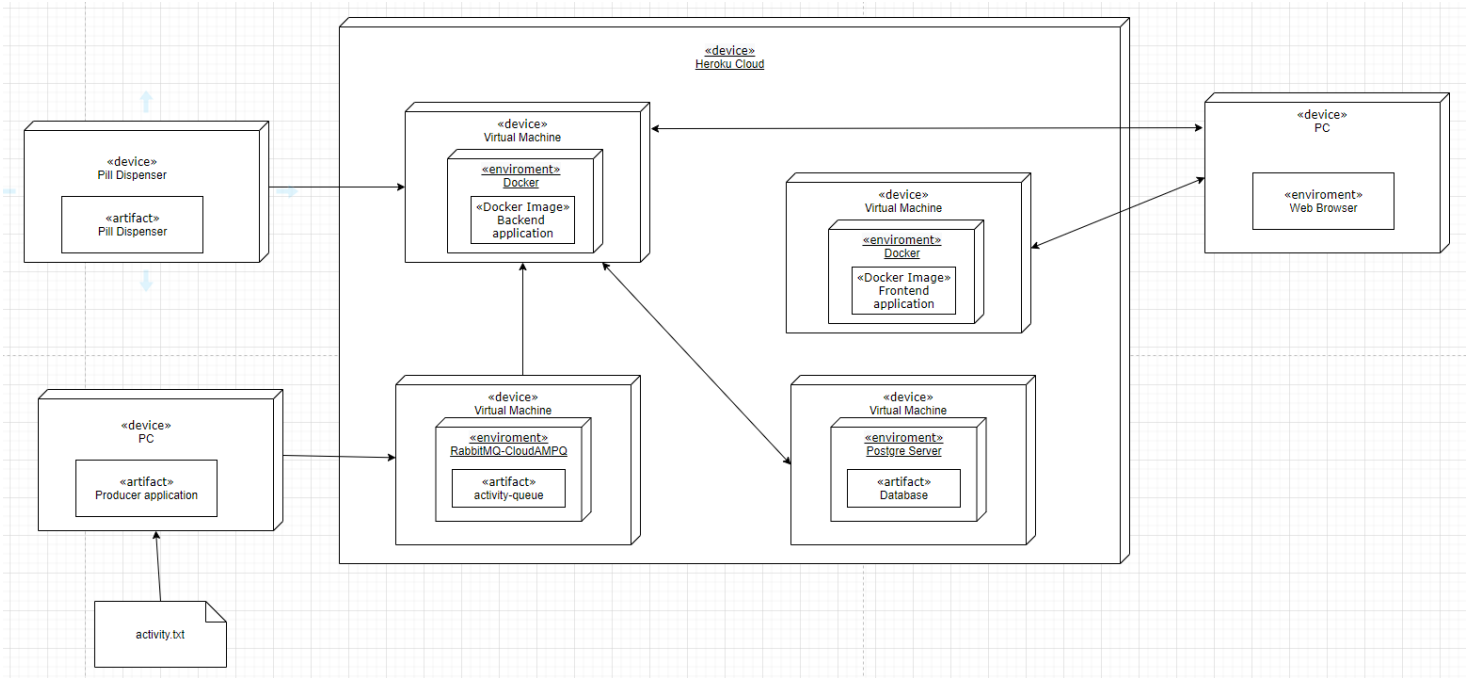
și inserarea acesteia în baza de date. În cazul observării unor neregularități, backendul va notifica caregiverul corespunzător pacientului în aplicația de frontend prin intermediul mesajelor asincrone transmise prin WebSockets.

Pentru tema 3 se mai adaugă functionalitatea de Remote Procedure Call între backend și o aplicație terță ce se ocupă cu simularea unui aparat de pill dispensing către pacient. Acesta descarcă planurile medicale active ale pacientului o data la 24 de ore și îi pune la dipoziție pacientului o interfată prin care acesta să raporteze înapoi aplicației dacă a luat sau nu medicamentul. Aplicația face constant verificări astfel încât momentul administrării medicamentului să coincidă cu intervalul orar specificat de rețetă. În caz contrar, dacă medicamentul nu a fost administrat până la ora stabilită, aplicația va informa backend-ul de acest lucru, iar in baza de date va fi inserat un status negative al administrării medicamentului. Toată comunicarea dintre backend si aplicația de Pill Dispenser se realizează prin intermediul frameworkului de RPC numit Hessian care expune interfetele de comunicare asemanător cu SpringBoot prin endpointuri.



4. Diagrama de deployment

Pentru a prezenta structura fizică a aplicației, am realizat următoarea diagramă UML de deployment:



5. Considerații generale de utilizare

Aplicația a fost amplasată pe un pipeline de Continuous Integration and Deployment ce are ca rezultat expunerea acesteia în cloud, pe platforma Heroku. Pentru a fi accesat:

- Back-end-ul: https://mihaifilipdud-backend-sd.herokuapp.com/
- Front-end-ul: https://mihaifilipdud-frontend-sd.herokuapp.com/

Logarea în aplicație ca și doctor se poate face cu username-ul: mihai.f și parola: mihai.f.

Operații pentru rolul de doctor:

- CRUD-uri pe tabele se realizaează prin apăsarea butoanelor aferente
- Asignarea unui asistent la un pacient se realizează prin selectarea radio-buttonurilor aferente celor 2 și mai apoi apăsarea pe butonul de Assign Caregiver
- Pentru a creea un medication plan pentru un pacient, trebuie să se selecteze un pacient apasând pe radio-button, să se selecteze medicationurile dorite cu ajutorul checkbox-urilor și apoi se apasă pe butonul de Add Medication Plan pentru a completa detaliile necesare.

Pentru a trimite datele despre activitati inspre coada, se pornește aplicația de producer și se accesează endpointul: http://localhost:9292/activity/push.