**Assignment 4**

Halasz Lorand Daniel

Grupa 30242

1. **Descriere proiect**

Primul assignment a constat in crearea unei platforme online, care se ocupa de managementul pacientilor, a persoanelor care se ocupa de acestia si a planurilor de medicamentatie. Sistemul poate fi accesat de trei tipuri de utilizatori: doctori, pacienti si persoane care au grija de acestia. Tipul utilizatorilor se stabileste pe baza unui proces de logare, fiecare cont creat va avea asociat un rol, astfel incat un utilizator va avea anumite capabilitati in functie de acest rol. Utilizator se conecteaza la platforma pe baza unui username si a unei parole.

Un pacient este definit de:

* Id
* Nume
* Data de nastere
* Email
* Sexul
* Adresa
* Istoric medical

Persoana care are grija de pacienti este definit de:

* Id
* Nume
* Data de nastere
* Email
* Sexul
* Adresa
* Lista cu pacientii asociati

Medicamentul este definit de:

* Id
* Nume
* Efecte secundare
* Dozaj

Assignmentul 4 consta in crearea a doua web service-uri, in doua limbaje de programare diferite, unul pentru doctor si unul pentru caregiver. Cele doua web service-uri ar trebui sa aiba urmatoarele functionalitati:

* Service-ul pentru doctor:
  + Vizualizarea istoricului unui pacient sub forma de chart pentru activitati zilnice
  + Vizualizarea medicamentelor luate si neluate din plan medical
  + Vizualizarea activitatilor zilnice pentru un pacient si posibilitatea de a modifica statusul acestora din anomalii in normale
  + Posibilitatea de a adauga recomandari
* Service-ul pentru caregiver:
  + Vizualizarea recomandarilor pentru pacienti

Pentru implementarea proiectului s-au folosit urmatoarele tehnologii:

* SOAP
* .NET
* GRPC
* RabbitMQ
* WebSockets
* Servicii REST pe partea de backend (Java Spring REST)
* Framework-ul ReactJS pentru partea de frontend (bazat pe limbajul JavaScript)
* Hibernate
* Maven

Stil arhitectural de dezvoltare a aplicațiilor Web axat asupra reprezentarii datelor. Serviciile REST sunt servicii web bazate pe metodele HTTP. Metodele principale care sunt folosite în definirea acestor servicii sunt:

* POST: upload-ul unei noi resurse (creare sau modificare)
* PUT: crearea unei noi resurse
* GET: cititrea unei resurse fără a modifica resursa
* DELETE: stergerea unei resurse

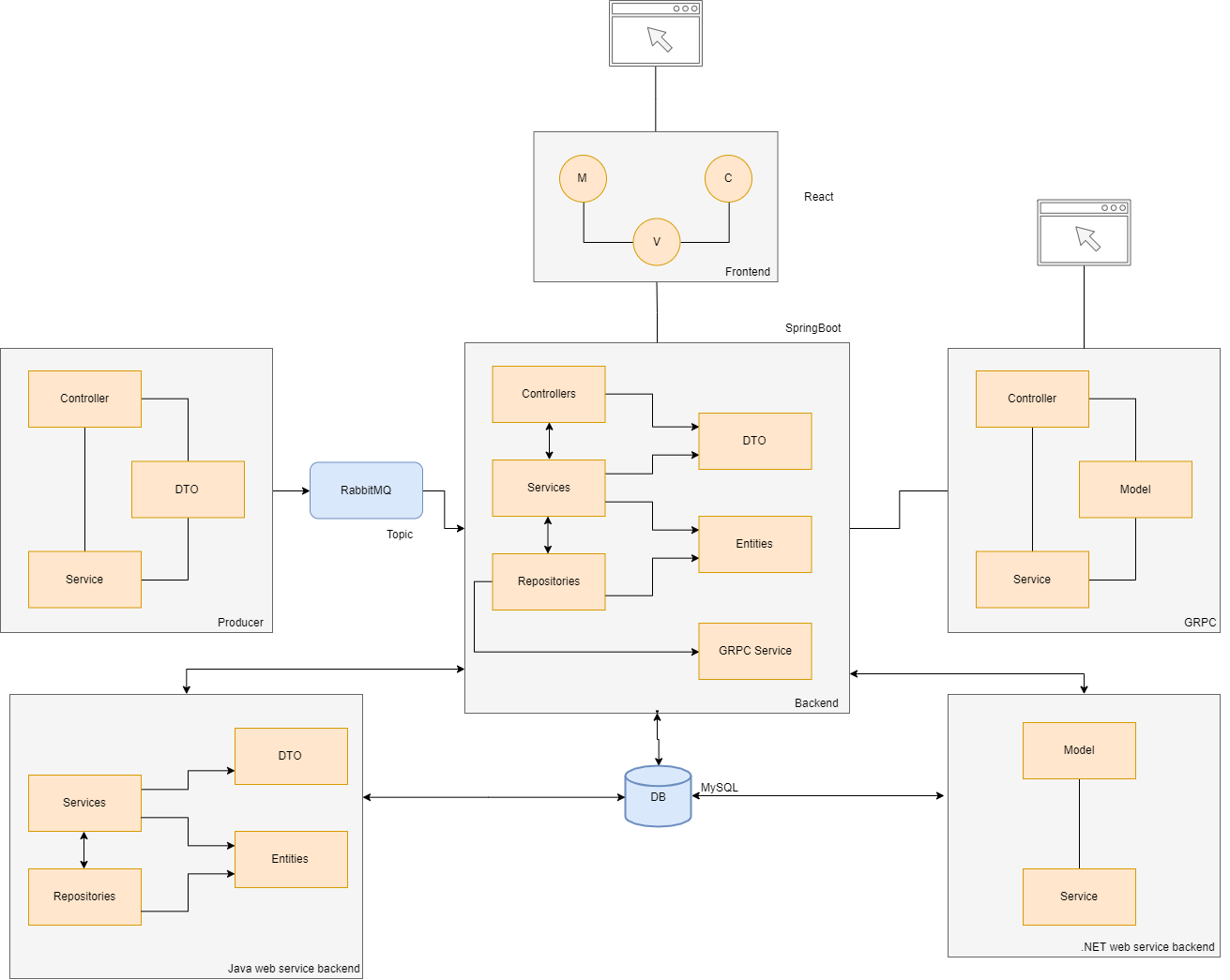
1. **Consideratii teoretice**

Hibernate este un framework Java foarte popular, care are ca rol facilitarea accesului la baze de date si conectarea intre lumea obiectuala si lumea tabelelor relationale.

Modelul client-server este o arhitectura de aplicatie distribuita care partajează task-urile sau sarcinile de lucru intre furnizorii unor resurse sau unor servicii, numiti servere, si elementele care solicita aceste servicii, numite clienți. Adesea, serverele si clientii comunica printr-o retea de calculatoare, de obicei prin Internet, pe hardware separate, dar, de asemenea, atat clientul cat si serverul pot rula la fel de bine si pe acelasi sistem fizic. Un server ruleaza unul sau mai multe programe server, care partajeaza resursele existente cu clientii. In schimb, clientul nu partajeaza niciuna dintre resursele proprii, dar solicita continut sau serviciu de la un server. Prin urmare, clientii initiaza sesiuni de comunicare cu serverele si asteaptă mesajele acestora. Pentru menținerea legaturii intre cei doi, indiferent de pauzele care intervin, se foloseste conceptul de sesiune, care de cele mai multe ori este limitata in timp.

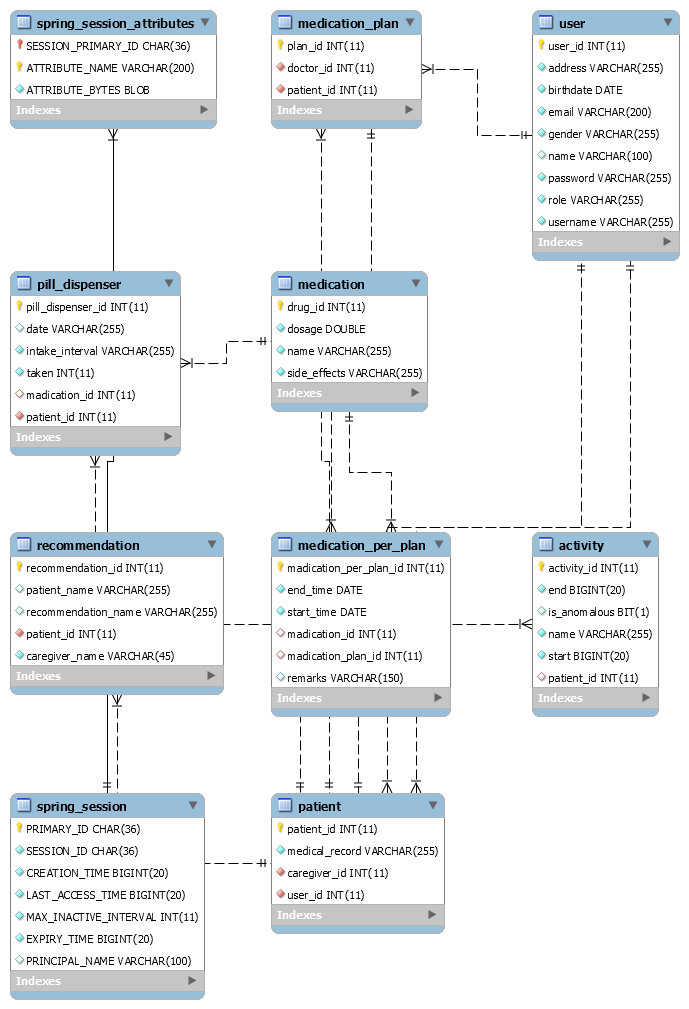
1. **Arhitectura conceptuala a sistemului distribuit**

Acest sistem este o aplicatie web, de tip client-server construita utilizand arhitectura pe layere.



1. **Diagrama DB**

Pentru crearea bazei de date, am folosit framework-ul hibernate, care face ca entitatile din codul Java sa se mapeze pe tabelele bazelor de date, folosind adnotatiile specific acestora. Baza de date a fost create conform cerintelor din assignment.



1. **Diagrama de deployment**

Cu ajutorul diagramelor de deployment se vor putea prezenta arhitectura fizica pe care va fi implementat sistemul, calculatoarele, device-urile (referite ca nodurile sistemului), impreuna cu conexiunile dintre ele. Componentele si obiectele executabile sunt alocate in interiorul nodurilor, ceea ce ne va permite o vizualizare a unitatilor care se vor executa pe fiecare nod.

