****

**DOCUMENTATIE**

**Aplicatie E-Health**

**Varvara Raluca Ana-Maria**

**Tudor Mihai Nicusor**

**Grupa 30232**

**Profesor Laborator: Delia Mitrea**

# Specificatii si analiza sistemului

Construiti un web-site pe domeniul e-health care sa permita monitorizarea la distanta a starii bolnavilor (simptome, temperatura, tensiune arteriala, etc), fiecare pacient avand medicul propriu si avand stabilite unul sau mai multe diagnostic, precum si antecedente personale, respeciv heredocolaterale (ale rudelor).

In aceasta aplicatie vor exista useri care trebuie sa se inregistreze, acestia fiind ori pacienti, ori doctori. In apgina principala a apliactiei vor putea sa se inregistreze sau daca deja sunt inregistrati vor putea sa se logheze, si la logare ii vor duce in pagina specifica in functie de ce rol au, pacient sau doctor. Pacientii apoi vor putea sa isi introduce starea (temperature, tensiunea si simptomele) pentru a putea fi monitorizati de doctori, si de asemenea vor putea sa isi introduca informatii personale suplimentare care ar putea fi folositoarea la alegerea unui diagnostic si a unei retete. Doctorii vor putea allege un pacient de pe lista pacientilor lor (un pacient are doar un doctor) si sa ii verifice starea, apo isa ii ofere un diagnostic si o reteta dupa consultarea tuturor informatiilor importante.

# Precizarea limbajului e programare

Limbajele de programare in care o sa lucram sunt:

* Java
* Spring
* HTML
* CSS
* JavaScript
* MySQL

Sistem de operare: Windows 10.

Java am ales deoarece este un limbaj de programare orientat-obiect, puternic tipizat, si este o tehnologie pe care o stapanim bine de la cursuri precedente, si datorita proiectelor realizate la materia tehnici de programare. Aceasta tehnologie este extrem de folosita, si de acea dupa acest proiect vom dobandi cunostinte care ne vor putea ajuta foarte mult in viitor. De asemenea am ales Java pentru multe alte avantaje pe care le are: este un limbaj foarte sigur (nu foloseste pointeri expliciti), este usor de mentinut (programul poate fi executat pe orice tip de masina si atunci nu mai exista costul portabilitatii).

Spring am ales doarece este un framework care ajuta foarte mult la programarea in Java si aduce extrem de multe avantaje la acest limbaj de programare. Spring se focuseaza pe viteza, simplicitate si productivitate si este cel mai popular Java framework. De asemenea, lucrand in acest mediu vom dobandi multe informatii foarte folositoare.

HTML si CSS si JavaScript am ales deoarece sunt niste limbaje foarte populare si usor de utilizat folosite in proiectarea aplicatiilor web, astfel ne vom putea dezvolta abilitatile de dezvoltare a acestui tip de aplicatii.

MySQL este un limbaj care lucreaza cu baze de date pe care l-am studiat in anul scolar anterior si in care am create de asemenea un proiect, si avem deja cunostinte in acest limbaj. In proiect am folosit si Java, deci comunicarea intre aceste 2 limbaje ne este deja cunoscuta.

# Fundamente teoretice

In primul rand, cea mai importanta parte a proiectului este folosirea unei arhitecturi client server, pentru implementarea aplicatiei de E-Health. Pe server se gaseste baza de date si aplicatia, iar clientul acceseaza browserul si clientii pot comunica intre ei prin intermediul serverului, prin trimitere de mesaje.

Modelul client-server este o structură care distribuie sarcini între furnizorii de resurse sau servicii numite servere și elementele care solicită resurse sau servicii, numite clienți. Adesea, clienții și serverele se află pe mașini fizice diferite și comunică printr-o rețea, dar atât clientul cât și serverul pot sta în același sistem sau pot comunica prin alte mijloace. Prin extensie, un sistem fizic ce rulează unul sau mai multe programe server, care distribuie resursele existente clienților, este denumit tot server. Clienții inițiază sesiuni de comunicare cu serverele, care așteaptă cereri.

Caracteristica client-server descrie relația programelor care colaborează într-o aplicație. Componenta server oferă o funcție sau un serviciu unuia sau mai multor clienți, care inițiază cereri pentru astfel de servicii. Serverele sunt clasificate în funcție de serviciile pe care le oferă. De exemplu, un server web servește pagini web și un server de fișiere servește fișiere de calculator, noi in acest proiect vom implementa un srever web pentru comunicarea intre doctori si pacienti. Partajarea resurselor unui server constituie un serviciu.

De asemenea, proiectul va fi implementat ca si un Software as a Service (Saas).

Software as a Service (sau SaaS) este o modalitate de a furniza aplicații prin Internet — ca serviciu. În loc să instalați și să întrețineți software, îl accesați pur și simplu prin Internet, eliberându-vă de gestionarea complexă a software-ului și hardware-ului. Aplicațiile SaaS sunt uneori numite software bazat pe web, software la cerere sau software găzduit. Oricare ar fi numele, aplicațiile SaaS rulează pe serverele unui furnizor SaaS. Furnizorul gestionează accesul la aplicație, inclusiv securitatea, disponibilitatea și performanța.

# Proiectarea aplicatiei

## Descrierea arhitecturii sistemului

Arhitectura sistemului aplicatiei, la fel ca si tehnologiile utilizate, poate fi separata in doua parti principale: arhitectura de frontend si arhitectura de backend.

Frontend-ul se bazeaza pe afisarea in browser, in functie de linkul accesat, a paginilor corespunzatoare construite in HTML, CSS si JavaScript. Aceste pagini datorita sablonului

oferit de motorul Thymeleaf va face conexiunea la suprafata cu partea de backend a aplicatiei.

Backend-ul, implementat in Spring Boot, va consta din mai multe module: model, controller, exception and error, service, repository and validation.

Aceste module se gasesc in ambele configuratii - Server si Client:

* **Model** - consta din toate clasele care pot fi definite ca tabele in baza de date
* **Controller** - gestioneaza cererile API, pregateste modele si returneaza raspunsuri
* **Exception and Error** - toate exceptiile/errorile care pot fi aruncate/interceptate de aplicatie sunt definite aici
* **Repository** - reprezinta stratul de acces la date, servind ca o modalitate de a incapsula stocarea, substragerea si cautarea.
* **Service** - aici gasim toata logica de manevrare a aplicatiei
* **Validation** - inainte de a fi introdus date de catre user trebuie ca acestea sa fie acceptabile pentru aplicatie, verificari ce au loc in acest modul.

Modelul de design utilizat in aceasta parte este de Model-View-Controller. Prin acest lucru trimitem solicitari catre gestionatori cu mapari configurabile,

rezolutii de vizualizare, precum si suport pentru incarcarea datelor.

In urmatoarea figura se poate sesiza intreaga structura a proiectului.

Diagram

Description automatically generated

## Identificarea functionalitatilor aplicatiei

Diagram

Description automatically generated

Diagrama de use case este prezentata mai sus, toate functionalitatile se pot determina din cerinta proiectului. Am mai adaugat noi functionalitatea de adaugare de reteta de catre doctor si de vizualizare a retetei de client.

Userul poate sa:

* se inregistreze
* sa se logheze

Dupa inregistrare se face diferenta intre 2 tipuri de useri: pacienti si doctori.

Pacientul poate sa:

* isi updateze starea, adica: temperature, tensiunea si simptomele si sa creeze o consultatie
* sa isi adauge informatii suplimentare, adica: antecedente si heredocolaterale
* sa isi vizualizeze diagnosticul
* sa isi vizualizeze reteta

Doctorul poate sa:

* vizualizeze consultatia unui pacient ales
* sa ii adauge un diagnostic
* si sa ii adauge o reteta

## Diagrama de clase

Diagrama Proiect\_Client

Diagram

Description automatically generated

Diagrama Proiect-Server

Diagram

Description automatically generated

## Diagrame de interactiune: activitate, secventa, colaborare

Diagrama de activitate

Diagram

Description automatically generated

Diagrama secventiere client logare + creare consultative si doctor +finalizare consultatie

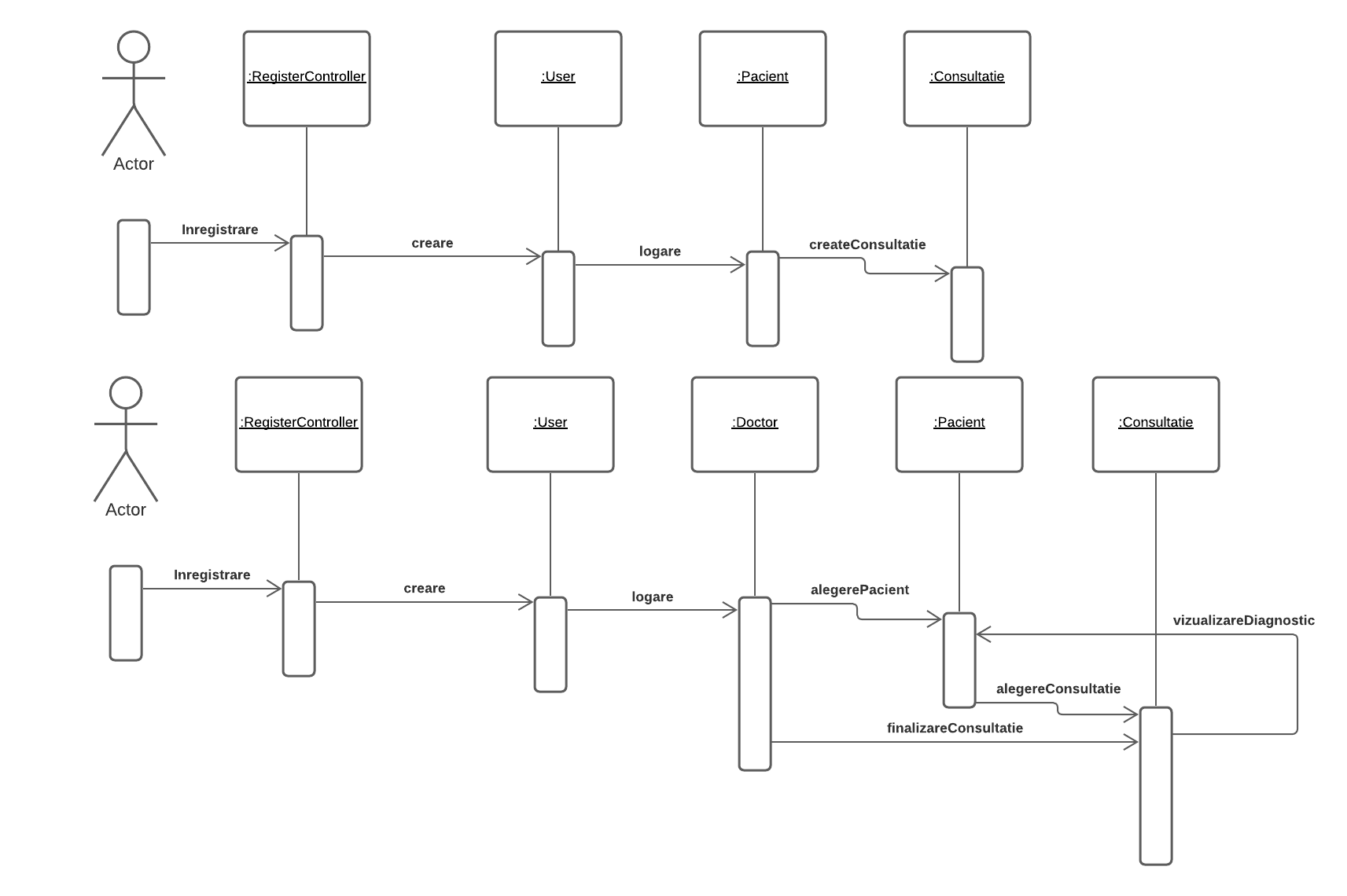


Diagrama colaborare inregistrare/logare

Diagram

Description automatically generated

Diagrama colaborare pacient doctor

Diagram

Description automatically generated

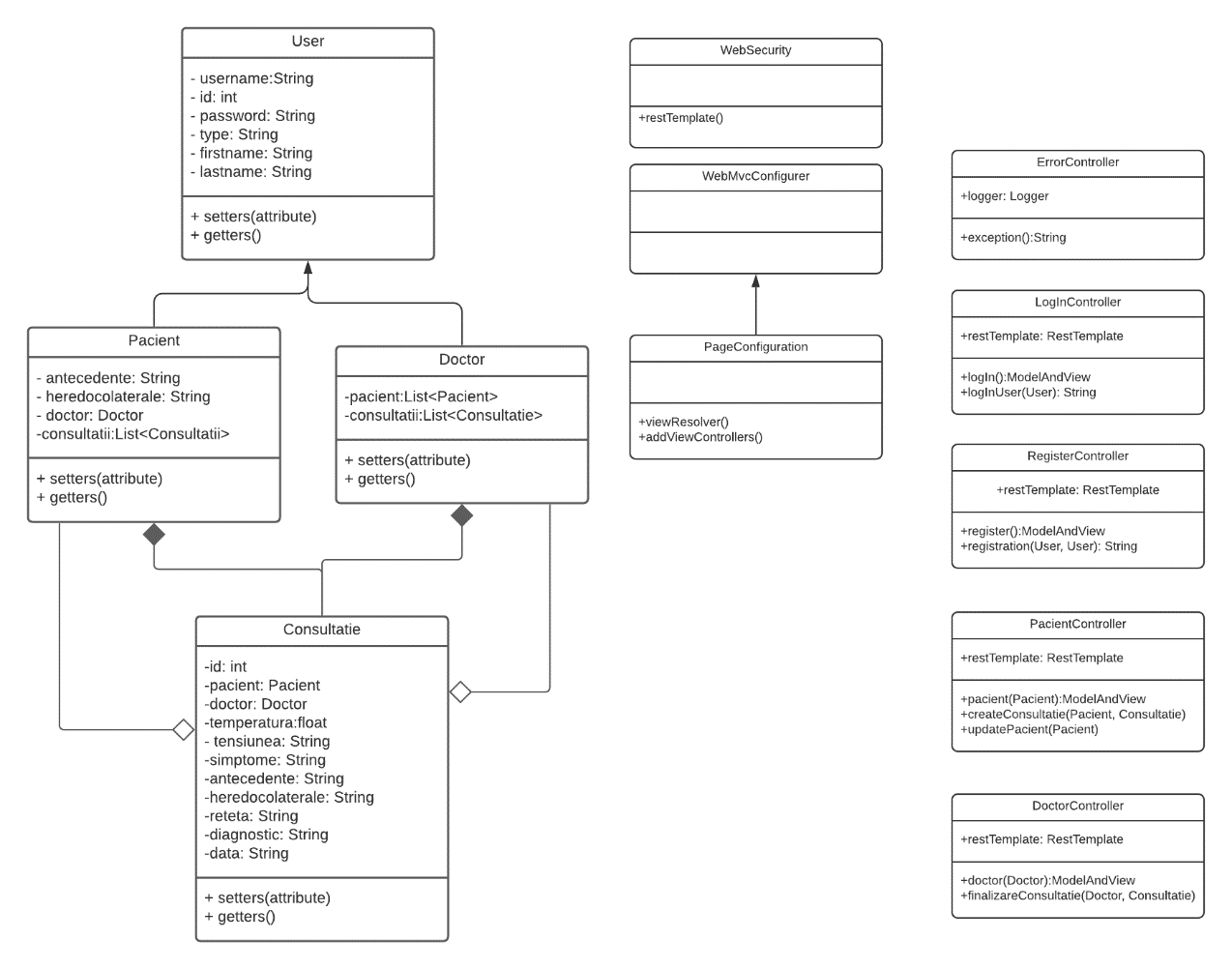
## Detalii de implementare: diagrama bazei de date, diagrama de clase cu atribute şi metode, diagrama de module şi componente

Diagram

Description automatically generated

Diagram

Description automatically generated



Diagram

Description automatically generated

Diagram

Description automatically generated

Diagram

Description automatically generated with medium confidence

## Cazuri de testare

Se acceseaza pagina de logare -> http://localhost:8082/login

I. Daca userul are cont:

1. apasa pe butonul de logare:

1.1. pop-up legat de faptul ca nu a introdus datele

1.2. eroare la introducere datelor

1.3. revine la pasul I

1.4. merge la pasul 4

2. introduce usernameul:

2.1. merge la pasul 1

3. introduce username-ul si parola:

3.1. merge la pasul 1

4. logarea a avut succes:

4.1. redirectionare la pagina de pacient -> http://localhost:8082/pacient

4.2. redirectionare la pagina de doctor -> http://localhost:8082/doctor

II. Daca userul nu are cont:

1. apasa pe butonul de register:

1.1. redirectare la pagina de inregistrare -> http://localhost:8082/login/register

2. completare username:

2.1. apasare pe butonul de register

2.2. eroare la complentarea datelor incomplet

2.3. revine la pasul 1.1.

3. completare username si parola:

3.1. apasare pe butonul de register

3.2. selectat automat ca si pacient

3.3. username deja existent

3.4. revenire la pasul 1.1.

3.5. selectare manuala ca si pacient/doctor

3.6. username nou

3.7. merge la pasul 4

4. inregistrarea a avut succes:

4.1. redirectare la pagina de logare -> http://localhost:8082/login

4.2. merge la pasul I

Diagram

Description automatically generated

# Testarea propriu-zisa a aplicatiei

Testarea s-a realizat din browser pentru fiecare tip de utilizator. Cu tot cu logare a unui utilizator deja existent, sau inregistrare a unui utilizator nou. Prima oara ne vom loga cu 2 utilizatori diferiti, unul pacient altul doctor, deja existenti, doctoral este mihai si are parola 12345, iar pacientul este ana si are parola 12345.

Graphical user interface, website

Description automatically generated

Dupa ce apasam pe butonul de logIn ne va deschide pagina specifica unui doctor.

Graphical user interface, application

Description automatically generated

Aici acesta poate sa isi aleaga pacientul dorit, momentan exista numai Ana Valea

Graphical user interface

Description automatically generated with low confidence

Iar apoi poate sa aleaga o consultatie de-a ei si sa introduca diagnosticul si reteta.

Graphical user interface, application

Description automatically generated

Graphical user interface, application

Description automatically generated

Graphical user interface, application

Description automatically generated

Si va apasa pe butonul de finalizare a consultatiei.

La fel va fi si pentru pacient, ne intoarcem la pagina de login:

Graphical user interface, website

Description automatically generated

Si dupa ce apasam pe butonul de log in ne va redirectiona catre pagina specifica unui pacient

Graphical user interface, application

Description automatically generated

Aici ea poate sa creeze o consultatie:

Graphical user interface

Description automatically generated

iar daca nu sunt completate campurile cum trebuie se va primi un mesaj de eroare.

Sau poate sa isi actualizeze datele completand ultimele campuri si apasand pe butonul de actualizare:

Graphical user interface, application

Description automatically generated

Dupa ce a create o consultatie corect, si dupa ce doctorul poate sa o completeze aceasta poate sa isi vada diagnosticul din dropBoxul “selectati consultatie Dorita” si va vedea si ce simptome a avut pentru acea consultatie:

Graphical user interface, application

Description automatically generated

Pentru inregistrarea unui user din pagina de login, fara sa trebuiasca sa completeze un camp apasa pe butonul de inregistrare si va fi redirectionat catre pagina de inregistrare:

Graphical user interface, website

Description automatically generated

Daca va allege doctor, dupa ce va apasa pe varianta doctor din dropbox, comboboxul pentru alegere a unui doctor va disparea.

Graphical user interface, application

Description automatically generated

iar la pacient , acesta va trebui sa isi aleaga un doctor la care va fi asignat.

Graphical user interface, application, website

Description automatically generated

# Manual de instalare si utilizare

Deoarece este o aplicatie software as a service, pentru a rula aplicatia nu este necesara o instalare a ei, ci doar accesarea serverului local intr-un browser la alegerea utilizatorului atat timp cat aplicatia java pentru server ruleaza, dar si cea pentru client.

Pentru a utiliza aplicatia prima oara trebuie sa fie pornite serverul si clientul, sa ruleze pe masina pe care vrea sa se acceseze aplicatia, si sa intre intr-un browser unde sa acceseze local hostul folosit de aplicatie:



Apoi, acesta, daca nu are un cont deja, se va inregistra cum am prezentat mai sus in cazurile de testare:

* isi alege un username
* isi introduce o parola
* isi introduce prenumele
* si isi introduce numele
* din combobox alege tipul de user care va fi: pacient sau doctor
* daca este pacient va trebui sa aleaga din dropboxul de la mijloc un doctor caruia sa ii fie asignat, daca este doctor acel camp va disparea
* va apasa pe butonul de inregistrare

Pentru logare sunt necesare urmatorii pasi:

* introducerea username-ului
* introducerea parolei
* apasarea pe butonul de login

Ca pacient, pentru a creea o consultatie:

* introducerea temperaturii care este de 2 cifre urmata de un punct si de alte doua cifre (de exemplu 37.60) altfel va primi un mesaj de avertizare
* introducerea tensiunii care este de de 2 cifre sau 3 urmata de un “/” si de alte 2 cifre
* introducerea simptomelor
* apasarea pe butonul de creare consultatie

Pentru vizualizarea unui diagnostic din dropboxul de alegere a consultatiei trebuie numai sa apese pe consultatia dorita si apoi i se vor complete toate campurile cu ce simptome avea in momentul respective, dar sic ampul de diagnostic si reteta.

De asemenea, poate sa isi actualizeze informatiile antecedente si heredocolaterale completand cele 2 campuri si apasarea pe butonul de actualizare date.

Pentru doctor, pentru a putea complete un diagnostic si o reteta trebuie executati urmatorii pasi dupa logare care se face ca mai sus:

* alegerea din dropbox a unui pacient care este asignat lui
* alegerea din urmatorul dropbox a unei consultatii create de acesta
* vizualizarea temperaturii, tensiunii, simptomelor si antecedentelor
* completarea campurilor de diagnostic si reteta
* apasarea pe butonul de finalizare a consultatiei

# Concluzii si dezvoltari ulterioare

Cu ajutorul acestui proiect ne-am obisnuit cu arhitectura client server, o arhitectura foarte folosita mai ales in aplicatiile web, si am invatat multe tehnologii noi. Am invatat sa utilizam frameworkul Spring, un framework foarte folosit, mai ales in aplicatiile care folosesc Java, si legarea la baza de date. De asemenea, am invatat ce este software as a service si cum sa cream o aplicatie de acest tip. Am invatat mai multe despre interfetele aplicatiilor web create cu html, si cum sa le imbunatatim folosind css, in plus, cum sa le adaugam functionalitati mai complicate impreuna cu javascript si java.

Ca si dezvoltari ulterioare, s-ar putea muta aceasta aplicatie intr-o aplicatie pentru Android si iOS, deoarece ar fi mult mai la indemana pentru toata lumea, toata lumea utilizeaza un telefon, si fiecare ar putea sa isi faca o consultatie oriunde ar fi, telefonul toata lumea il are la el, pe cand laptopul sau computerul nu.

S-ar mai putea adauga un sistem de consultatii de urgenta, care sa il atentioneze pe doctor in cazul in care pacientul a patit ceva grav si are nevoie urgenta de ajutor. De asemenea, s-ar mai putea implementa in plus o sectiune de chat intre doctor si pacient pentru consultatii care au nevoie de mai multe detalii, sau daca doctorul are nevoie de informatii punctuale in plus.

De asemenea, s-ar mai putea adauga un agent de inteligenta artificiala care sa tina evidenta conversatiilor si consultatiilor si sa creeze un profil fiecarui pacient, care il va putea ajuta pe doctor in descoperirea unor boli sau detlaii despre starea pacientului, care sunt mai punctuale, sau mai rare.

# Bibliografie

* <https://www.jetbrains.com/help/idea/class-diagram.html>
* <https://spring.io/projects/spring-boot>
* <https://www.salesforce.com/in/saas/>
* **Modelling Interactions and Behaviour -LlosengCh08E2**
* <https://stackoverflow.com/questions/6262472/multiple-lines-of-input-in-input-type-text>
* <https://www.w3schools.com/html/html_forms.asp>
* <https://www.w3schools.com/howto/howto_css_responsive_form.asp>
* <https://www.w3schools.com/js/js_events.asp>
* <https://www.w3schools.com/js/js_date_methods_set.asp>
* <https://www.baeldung.com/java-connect-mysql>