



NAOUR
BOUTAINA

JEE : SPRING SECURITY, MVC,JPA,THYMELEAF, SPRING DATA

4-ème année IIR G21

Emsi les
orangers



Compte-rendu



Introduction



Énoncé



Conception



Code source



Captures écran de l'exécution



Conclusion

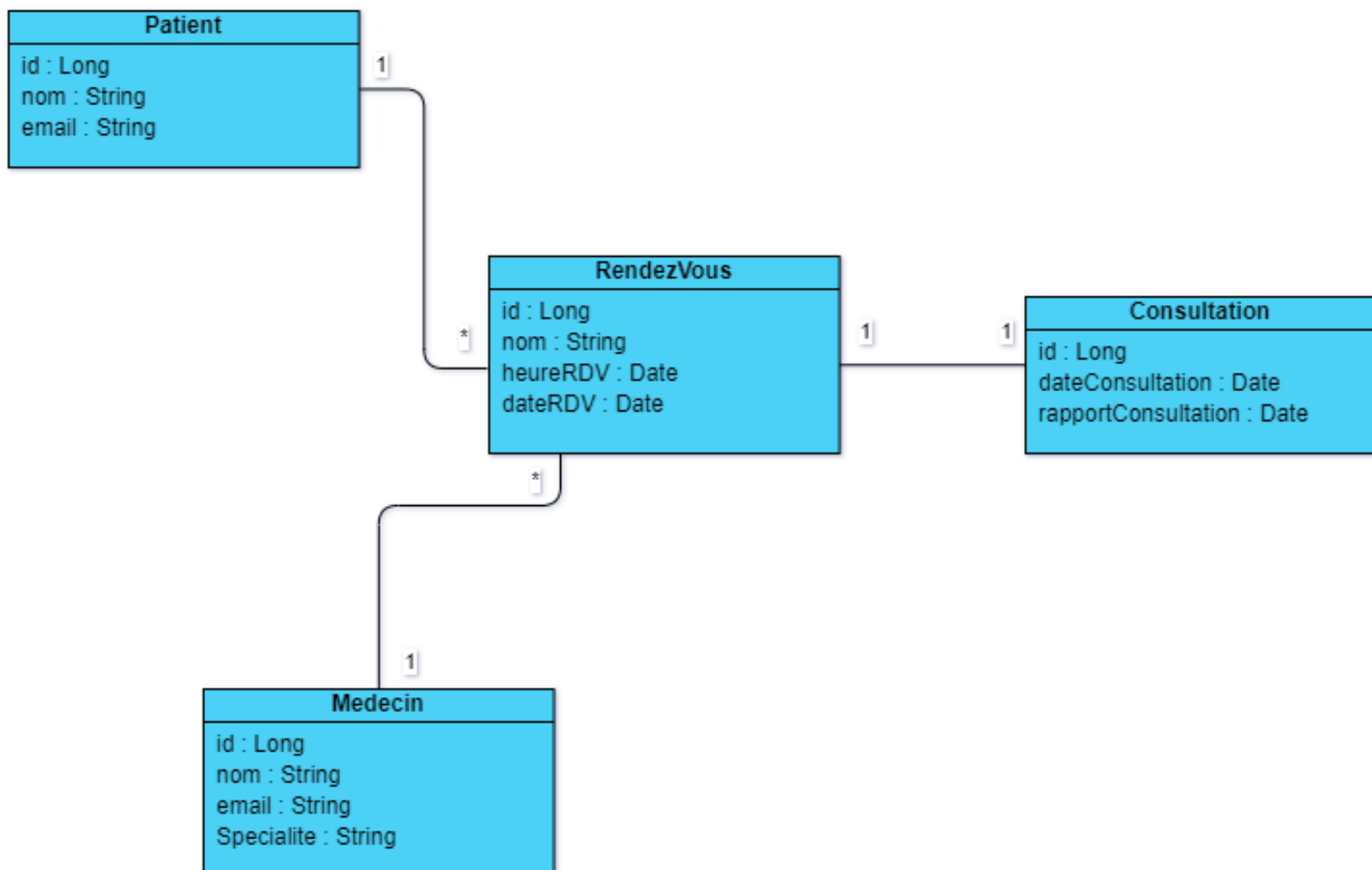
Dans cette partie on verra comment sécuriser l'accès à l'application qui permet de gérer les patients en utilisant Spring Security . L'accès à l'application nécessite

Une authentification avec un username et un mot de passe, un utilisateur peut avoir plusieurs rôles :

- ✓ `ROLE_USER` : permettant de chercher des patients
- ✓ `ROLE_ADMIN` : permettant d'ajouter ,Mettre à jour de supprimer les patients

Objectif : Créer une application qui permet de gérer des patients, des médecins et des rendez vous en utilisant Spring ,MySQL.

1. On souhaite gérer les rendez-vous des consultations des patients effectuées par des médecins.
2. Ajouter les dépendance maven de spring security
3. Chaque Rendez-vous concerne un patient et un médecin.
4. Pour chaque rendez-vous on associe une seule consultation issue de rendez-vous.
5. Un patient peut prendre plusieurs rendez vous



CODE SOURCE

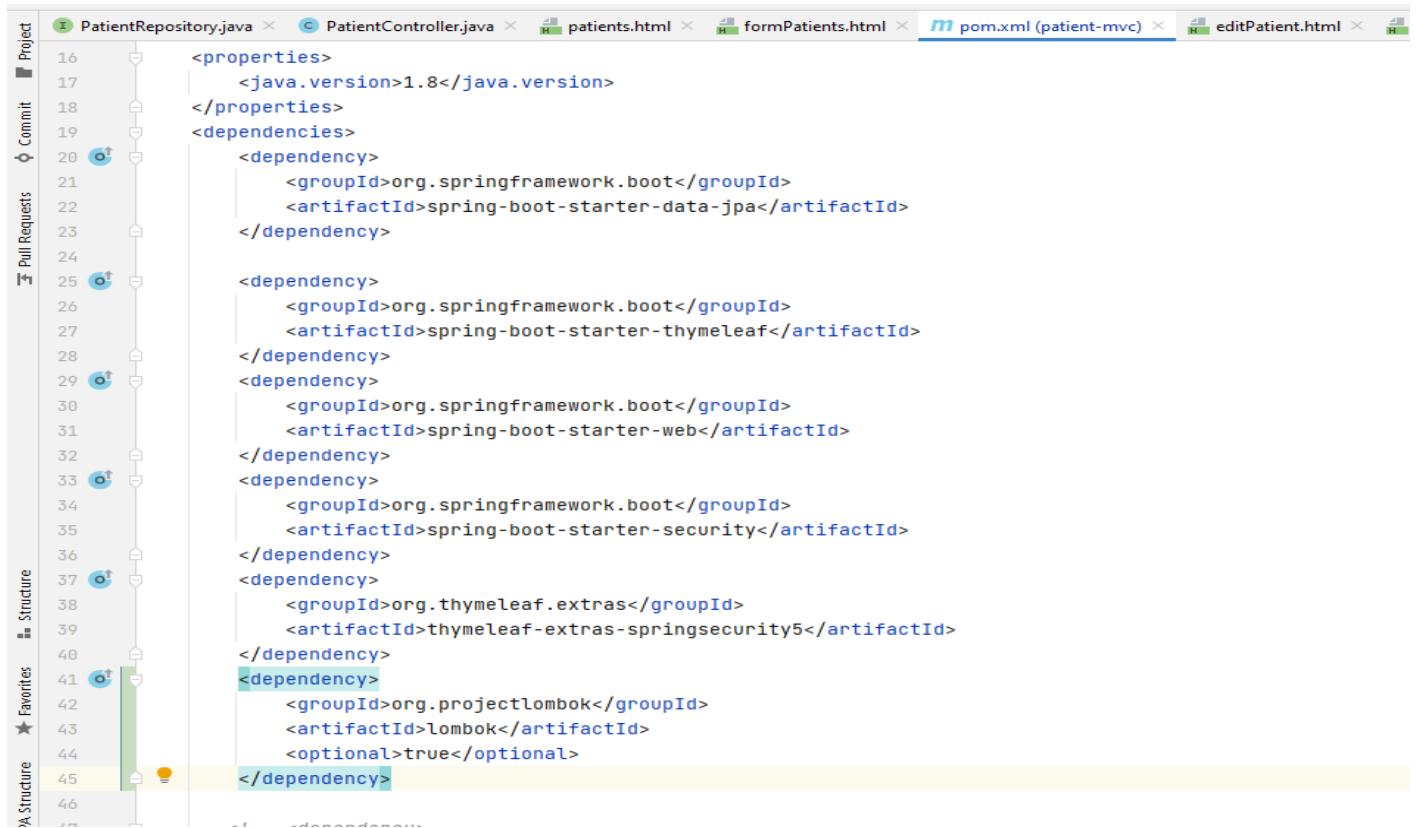
Lien du code source en GitHub :

https://github.com/bout-aina/jee_tp/tree/main/patient-mvc

Onglet 4

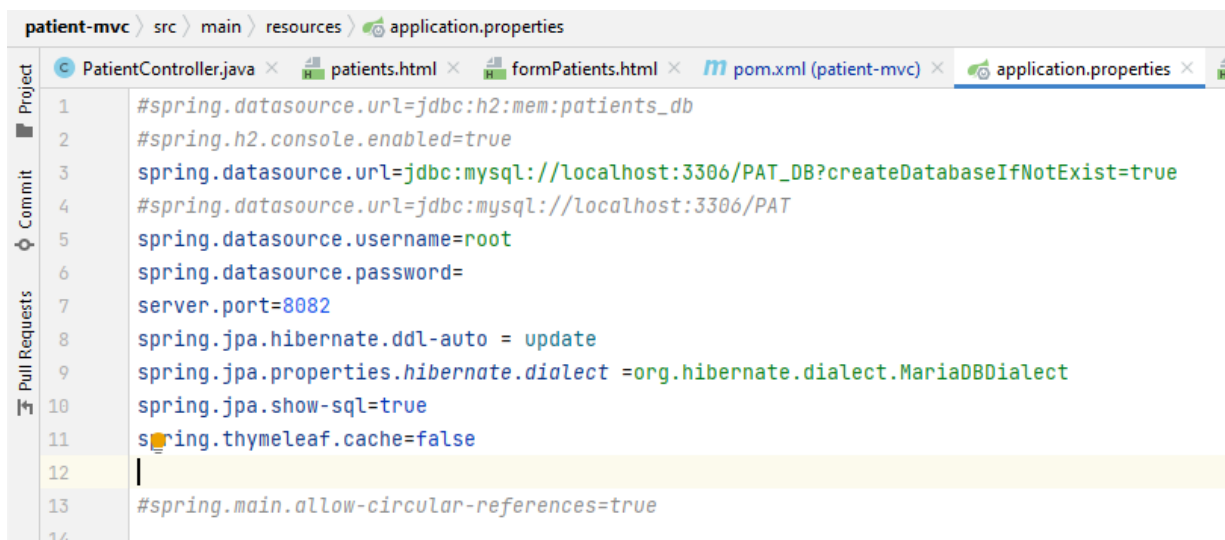
SCREEN SHOTS

A. Ajouter les dépendance maven de spring security



```
16 <properties>
17   <java.version>1.8</java.version>
18 </properties>
19 <dependencies>
20   <dependency>
21     <groupId>org.springframework.boot</groupId>
22     <artifactId>spring-boot-starter-data-jpa</artifactId>
23   </dependency>
24
25   <dependency>
26     <groupId>org.springframework.boot</groupId>
27     <artifactId>spring-boot-starter-thymeleaf</artifactId>
28   </dependency>
29   <dependency>
30     <groupId>org.springframework.boot</groupId>
31     <artifactId>spring-boot-starter-web</artifactId>
32   </dependency>
33   <dependency>
34     <groupId>org.springframework.boot</groupId>
35     <artifactId>spring-boot-starter-security</artifactId>
36   </dependency>
37   <dependency>
38     <groupId>org.thymeleaf.extras</groupId>
39     <artifactId>thymeleaf-extras-springsecurity5</artifactId>
40   </dependency>
41   <dependency>
42     <groupId>org.projectlombok</groupId>
43     <artifactId>lombok</artifactId>
44     <optional>true</optional>
45   </dependency>
46 </dependencies>
```

B. Application.properties:



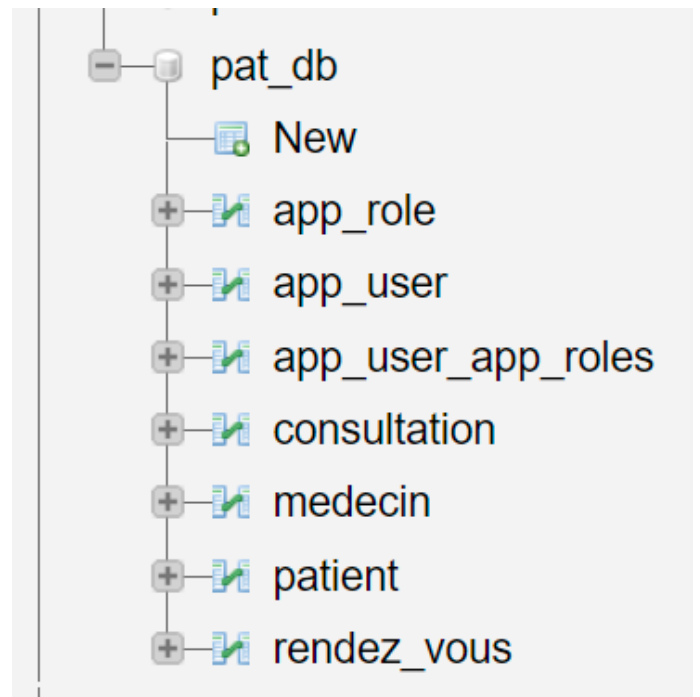
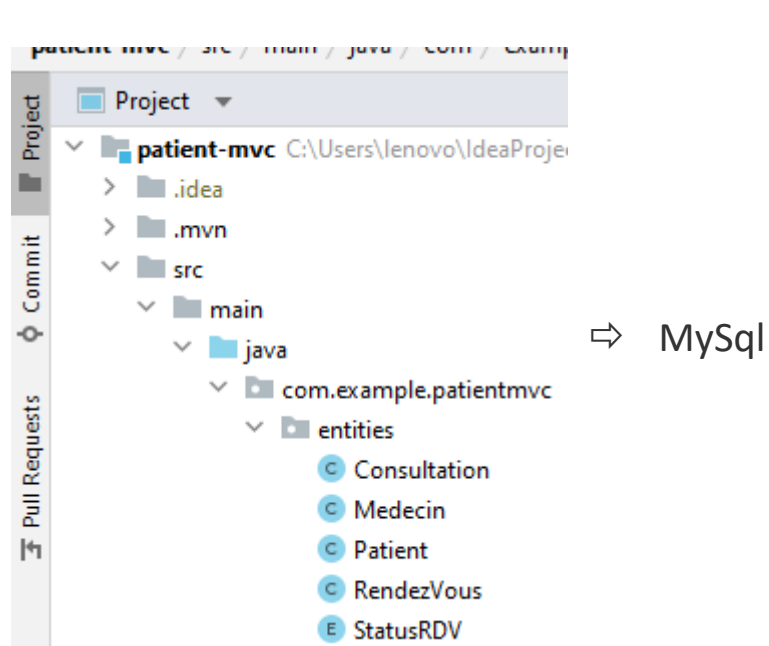
```
patient-mvc > src > main > resources > application.properties
1 #spring.datasource.url=jdbc:h2:mem:patients_db
2 #spring.h2.console.enabled=true
3 spring.datasource.url=jdbc:mysql://localhost:3306/PAT_DB?createDatabaseIfNotExist=true
4 #spring.datasource.url=jdbc:mysql://localhost:3306/PAT
5 spring.datasource.username=root
6 spring.datasource.password=
7 server.port=8082
8 spring.jpa.hibernate.ddl-auto = update
9 spring.jpa.properties.hibernate.dialect =org.hibernate.dialect.MariaDBDialect
10 spring.jpa.show-sql=true
11 spring.thymeleaf.cache=false
12
13 #spring.main.allow-circular-references=true
14
```

SCREEN SHOTS

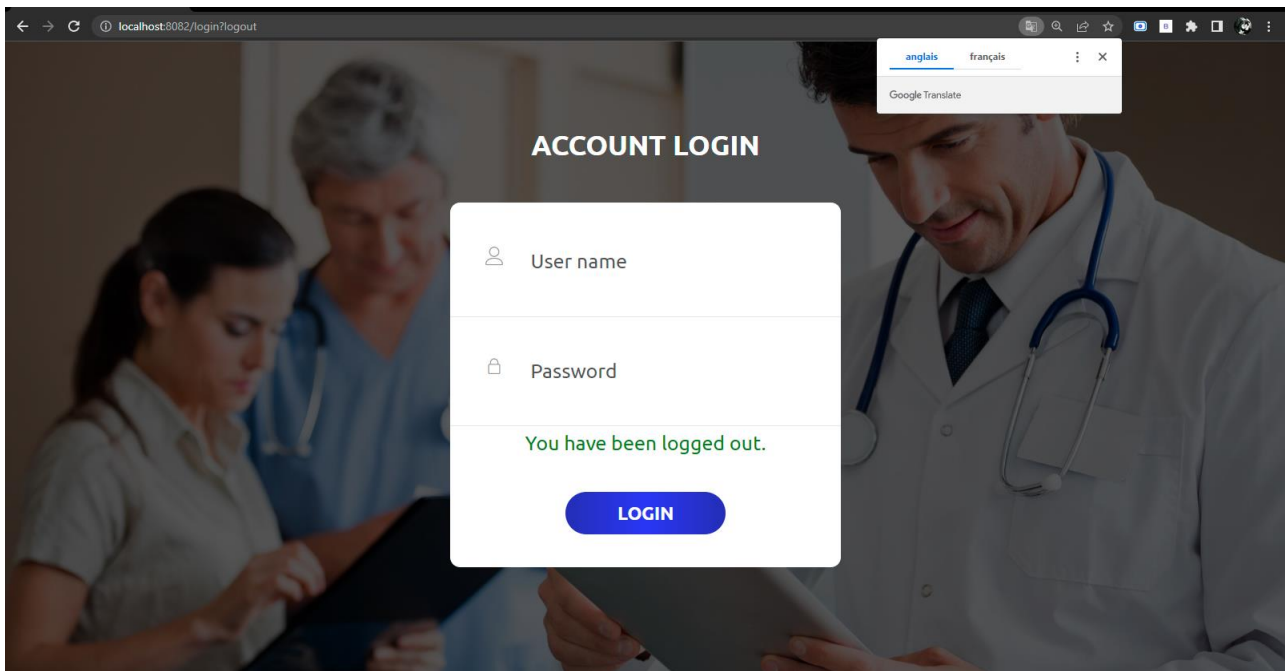
C. Ajouter des utilisateurs pour login selon leur role :

```
52  
53  
54  
55 // @Bean  
56 CommandLineRunner saveUsers(SecurityService securityService)  
57 {  
58  
59     return args -> {  
60         securityService.saveNewUser( username: "mohamed", password: "1234", rePassword: "1234");  
61         securityService.saveNewUser( username: "yasmine", password: "1234", rePassword: "1234");  
62         securityService.saveNewUser( username: "hassan", password: "1234", rePassword: "1234");  
63         securityService.saveNewRole( roleName: "USER", description: "");  
64         securityService.saveNewRole( roleName: "ADMIN", description: "");  
65         securityService.addRoleToUser( username: "mohamed", roleName: "USER");  
66         securityService.addRoleToUser( username: "mohamed", roleName: "ADMIN");  
67         securityService.addRoleToUser( username: "yasmine", roleName: "USER");  
68         securityService.addRoleToUser( username: "hassan", roleName: "USER");  
69  
70  
71     };  
72 }  
73
```

D. Création des entités dans la base de donnée



⇒ Tester les interfaces

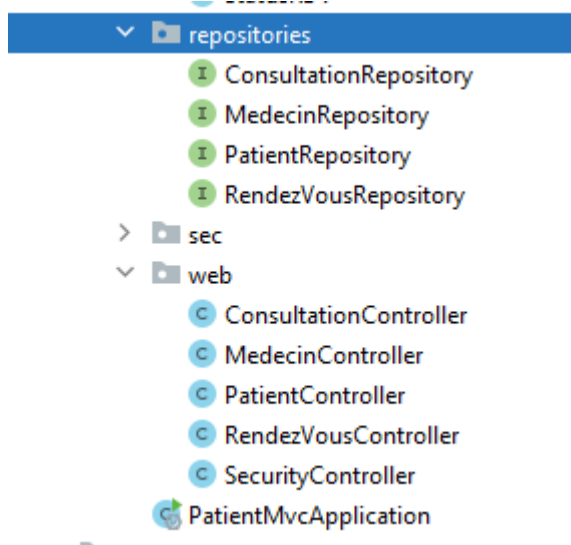
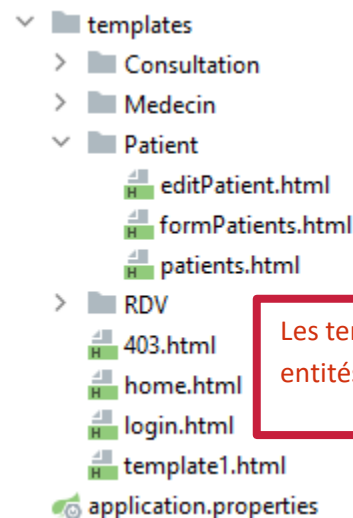


⇒ Page login première page qui s'affiche

Il ya deux types d'utilisateurs user et admin :

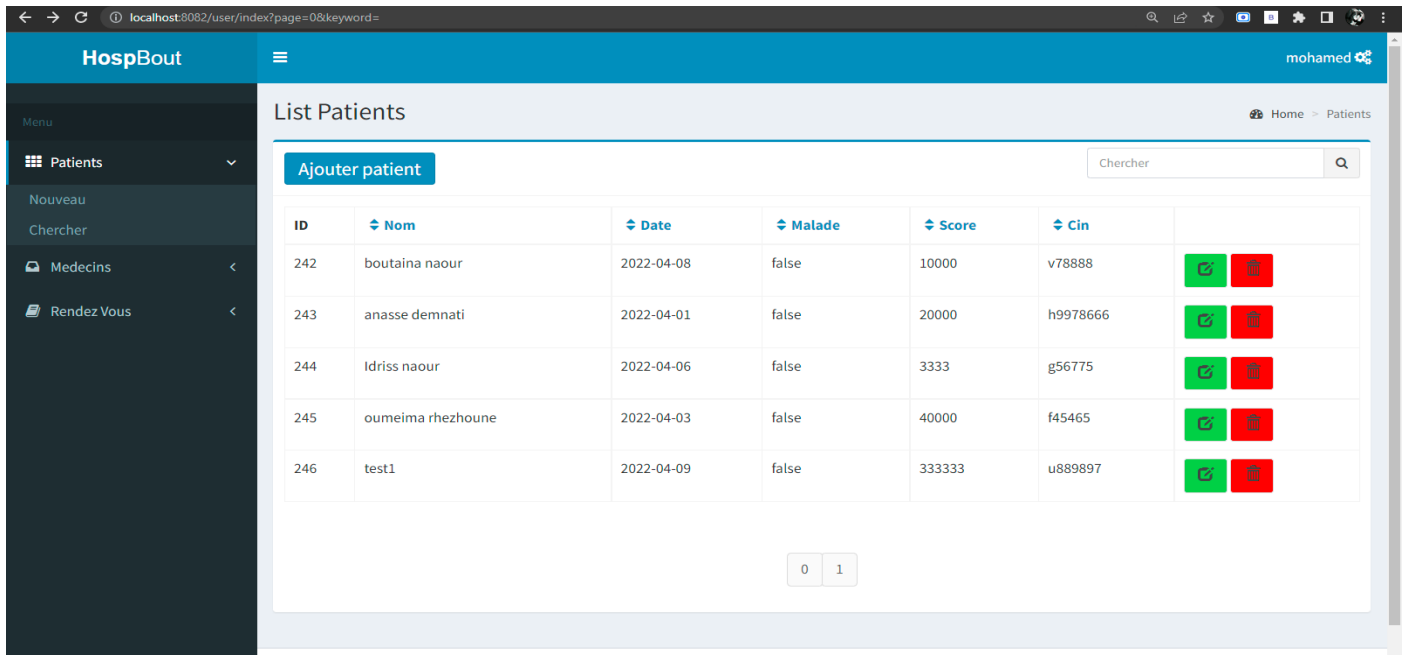
⇒ Admin a le droit de consulter , ajouter, supprimer et chercher les patients , rendez vous et les medecins

⇒ User peut juste consulter et chercher les patients , rendez vous et les medecins

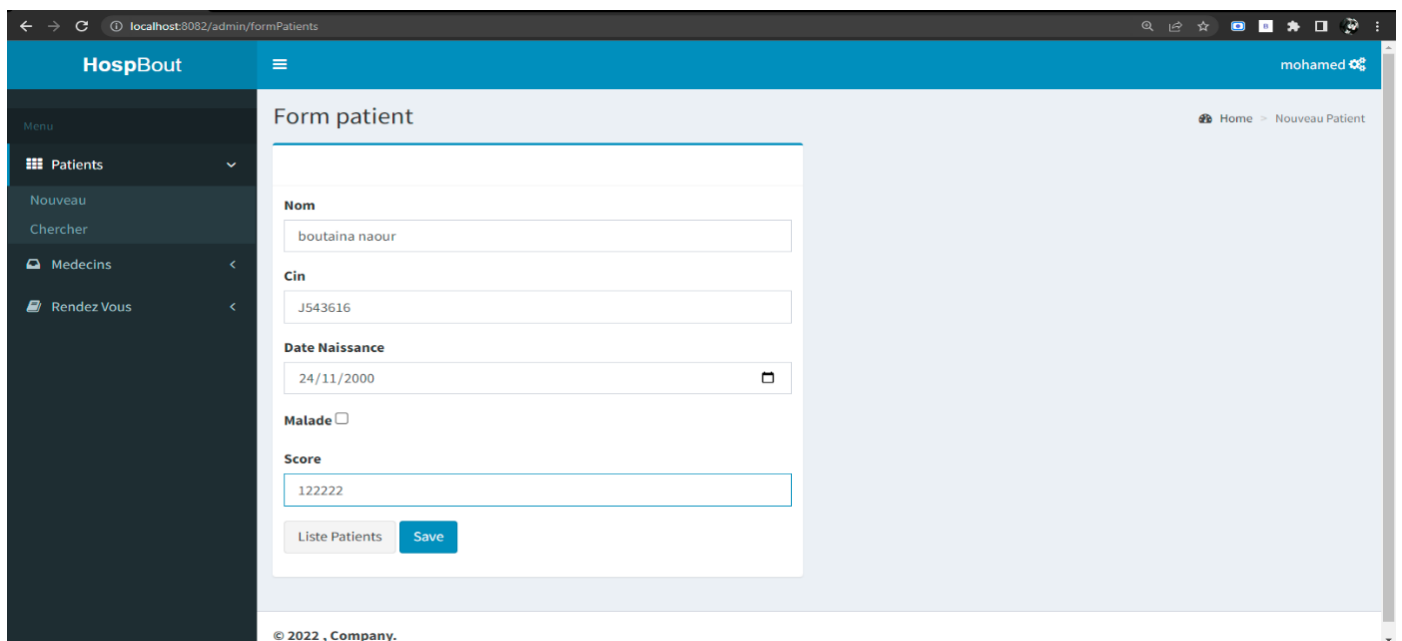


I. Partie Admin :

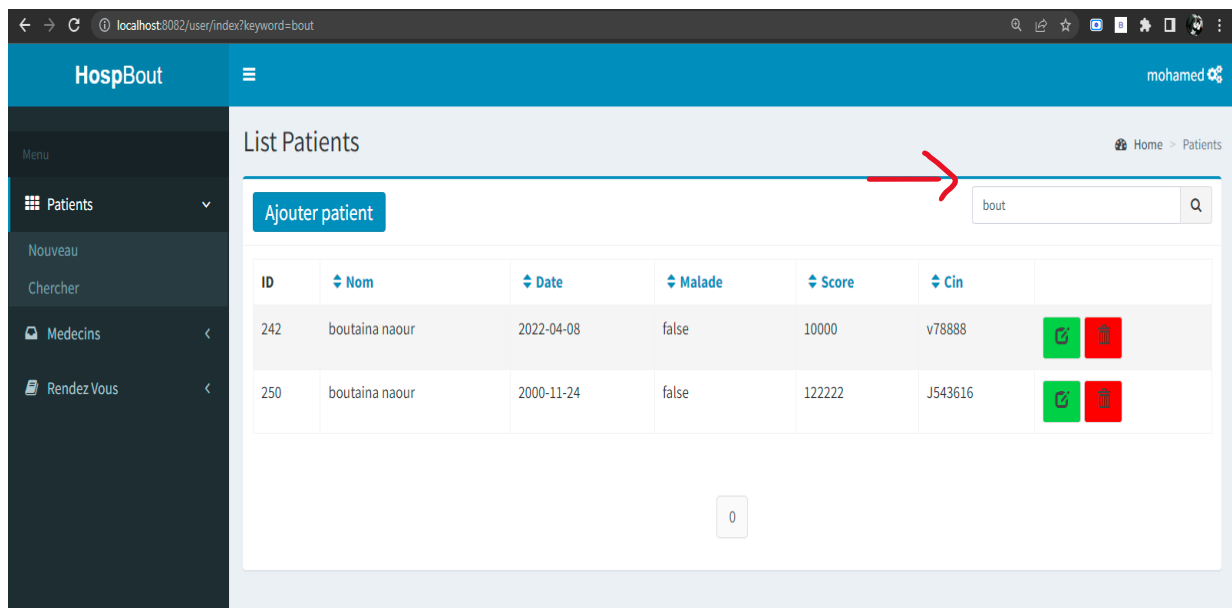
SCREEN SHOTS



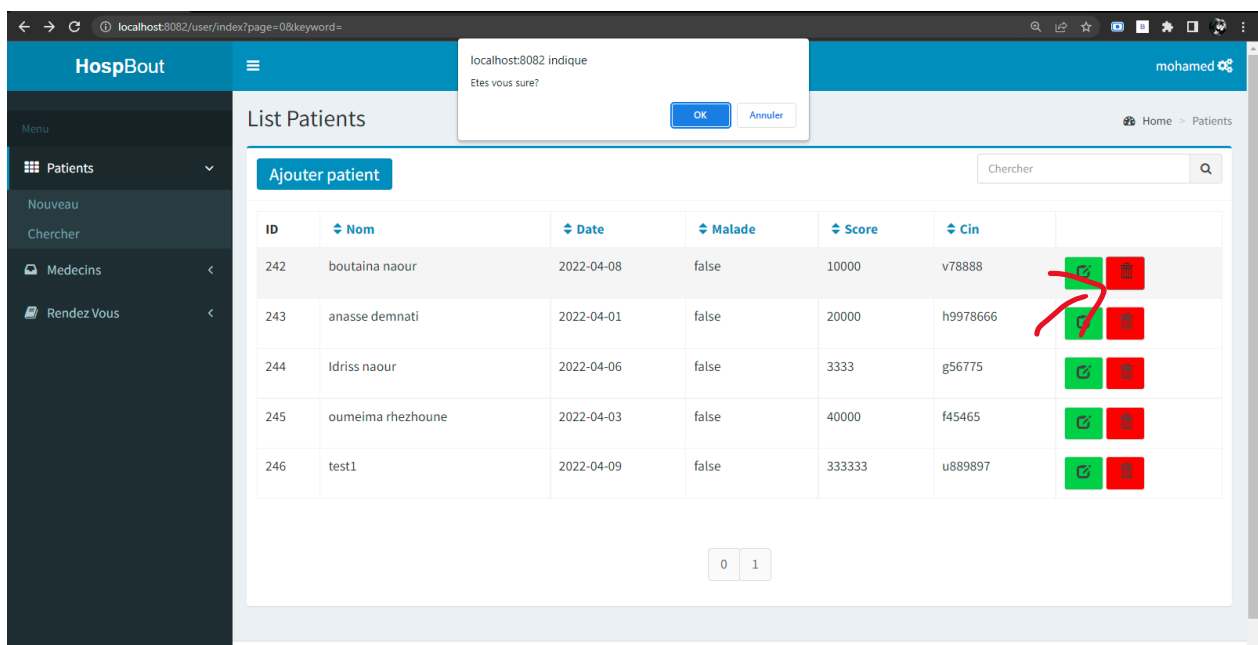
- List de tous les patients avec la possibilité de modifier , supprimer et ajouter un patient



- Formulaire pour l'ajout d'un nouveau patient



- La recherche par nom du patient



- Supprimer après la confirmation

HospBout

Menu

Patients

Nouveau

Chercher

Medecins



Rendez Vous

Admin/editPatient?id=242

List Patients

Ajouter patient

Chercher

ID	Nom	Date	Malade	Score	Cin	
242	boutaina naour 22	2022-04-08	false	10000	e24546	 

Edit Patient

Nom

boutaina naour 22

Cin

e24546

Date Naissance

08/04/2022

Malade

☐

Score

10000

Save

- Modification d'un patient

- Médecin : même CRUD que patient

HospBout

Menu

Patients

Medecins

Nouveau

Chercher





Rendez Vous

localhost:8082/user/indexMedecin?keyword=idris

List Medecins

Ajouter medecin

idris











ID	Nom	Email	Specialite	
2	idris	idris@gmail.com	Medecin Général	 
6	idris	idris@gmail.com	Medecin Général	 

0

List Medecins

Ajouter medecin

Chercher

ID	Nom	Email	Specialite	
6	idris	idris@gmail.com	Medecin Général	 
7	anasse	anasse@gmail.com	Medecin Général	 
8	ismail	ismail@gmail.com	Medecin Général	 
10	nouveau1	nv@gmail.com	Medecin général	 
11	nouveau medecin	nv@gmail.com	Medecin génichologue	 

0 1

HospBout

Menu

Patients

Medecins

Nouveau

Chercher

Rendez Vous

localhost:8082/admin/formMedecins

Form medecin

Nouveau Medecin

Nom

nouveau medecin

Email

nv@gmail.com

Specialité

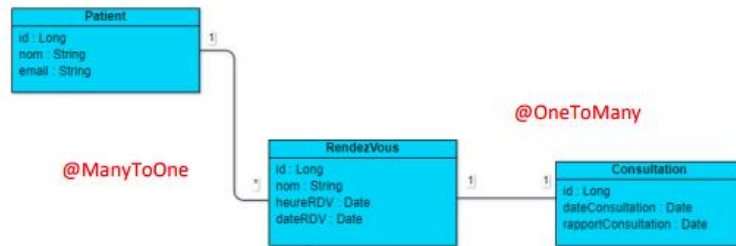
Medecin génichologue

Liste Medecins

Save

Liste contenant les spécialité disponible

- Rendez vous : possède une relation many to one pour le patient et le médecin c.-à-d chaque patient ou bien médecin peut avoir un ou plusieurs rendez vous



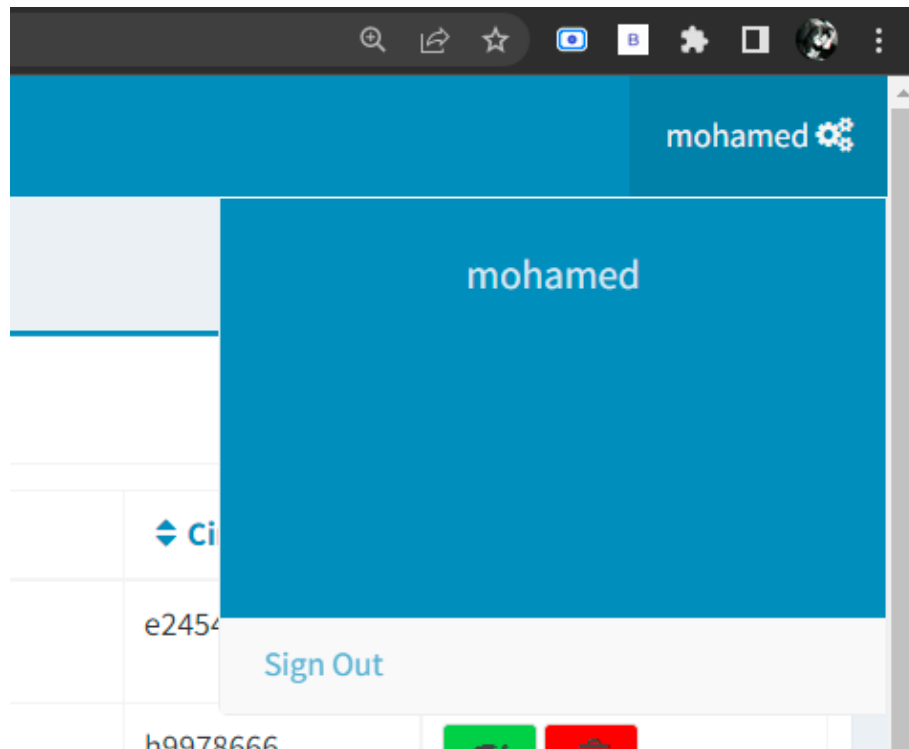
```

Patient.java
13 @AllArgsConstructor
14
15 public class Patient {
16     @Id
17     @GeneratedValue(strategy = GenerationType.IDENTITY)
18
19     private Long id;
20     private String nom;
21     @Temporal(TemporalType.DATE)
22     private Date dateNaissance;
23     private boolean malade;
24     @OneToMany(mappedBy = "patient", fetch = FetchType.LAZY)
25     private Collection<RendezVous> rendezVous;
26
27 }
  
```

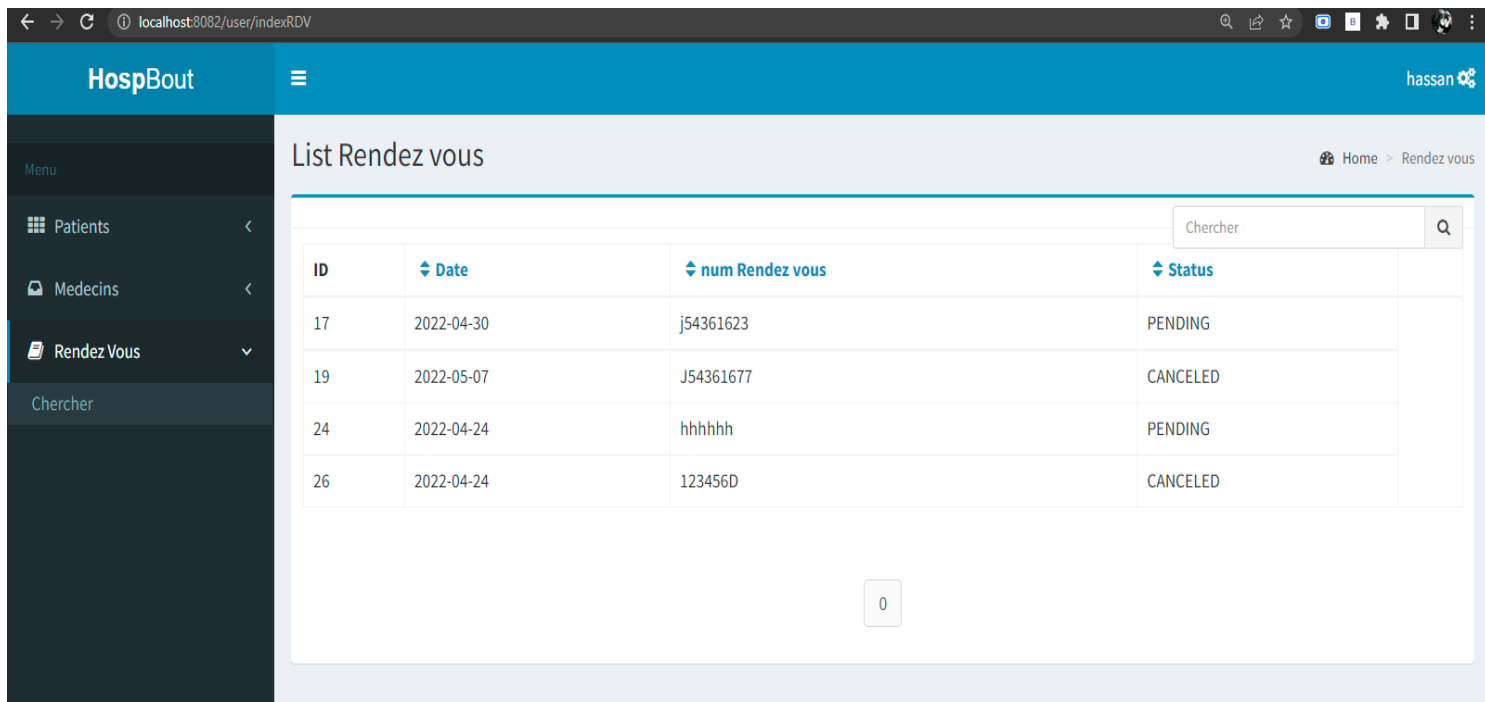
une liste pour pouvoir
getter les patients et les médecins

Pour Edit il peut modifier que ces champs

- Logout



II. Partie User : il peut que voir la liste des patients , des médecins et des rendez vous



HospBout

Menu

Patients

Medecins

Chercher

Rendez Vous

hassan

List Medecins

Home > Medecins

Chercher

Q

ID	Nom	Email	Specialite
1	boutaina	boutynaour8@gmail.com	Medecin cardiologue
2	idriss	idriss@gmail.com	Medecin Général
3	anasse	anasse@gmail.com	Medecin Général
4	ismail	ismail@gmail.com	Medecin Général
5	boutaina	boutynaour8@gmail.com	Medecin Général

01

HospBout

Menu

Patients

Chercher

Medecins

Rendez Vous

hassan

List Patients

Home > Patients

Chercher

Q

ID	Nom	Date	Malade	Score	Cin
242	boutaina naour 22	2022-04-08	true	10000	e24546
243	anasse demnati	2022-04-01	false	20000	h9978666
244	Idriss naour	2022-04-06	false	3333	g56775
245	oumeima rhezhoune	2022-04-03	false	40000	f45465
247	test2	2022-04-14	false	5555555	j78888

01

Alors Spring Security est un module de Spring qui permet de sécuriser les applications Web , configure aussi des filtres (springSecurityFilterChain) qui permet d' intercepter les requêtes HTTP et de vérifier si l' utilisateur authentifié dispose des droits d' accès à la ressource demandée. Les actions du contrôleur ne seront invoquées que si l' utilisateur authentifié dispose de l' un des rôles attribués à l' action. Spring Security peut configurer les rôles associés aux utilisateurs en utilisant différentes solution (Mémoire, Base de Données SQL, LDAP, etc..),C' était le but pour le quel on a travaillé avec Spring Security.