Ecole Normale Supérieure de l'Enseignement Technique Mohammedia

Université Hassan II de Casablanca

ENSET

ا لمدرسة العليا لأساتذة التعليم التقني المحمدية
جامعة الحسن الثاني بالدار البيضاء

DEPARTEMENT MATHEMATIQUES ET INFORMATIQUE

Compte rendu

- Travaux Pratique : 2- -

« Architecture JEE »

JPA
HIBERNATE ET SPRING DATA

Réalisé par :

Ikram Berradi

Encadré par :

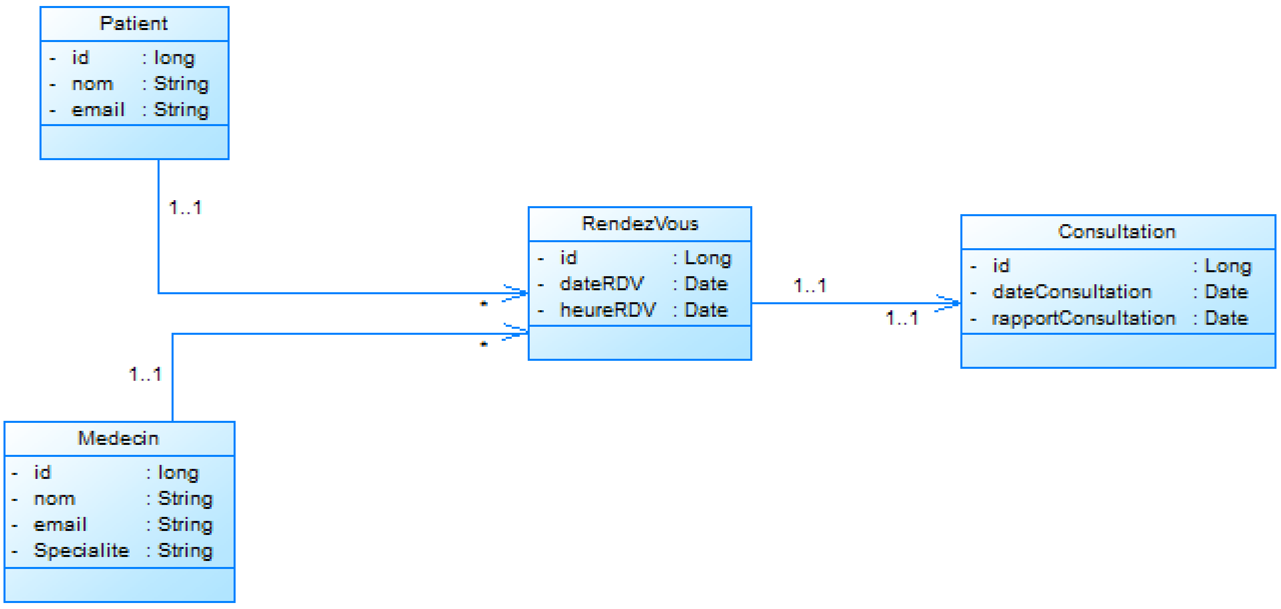
Mohamed YOUSSFI

GLSID II-2022/2021

Mapping objet relationnel avec JPA, Hibernate et Spring Data

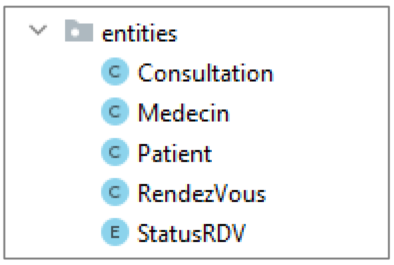
❖ Association OneToMany, ManyToOne, OneToOne

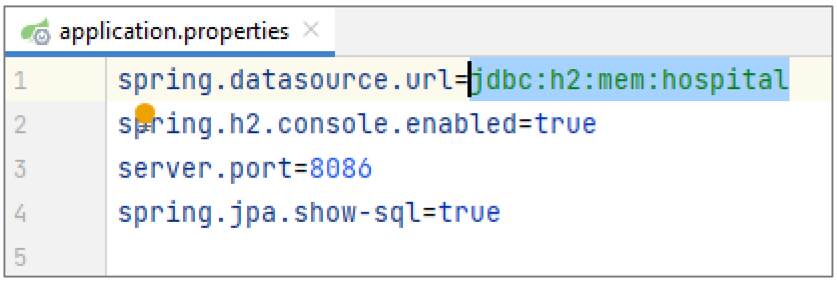
- On souhaite gérer les rendez-vous des consultations des patients effectuées par des médecins.



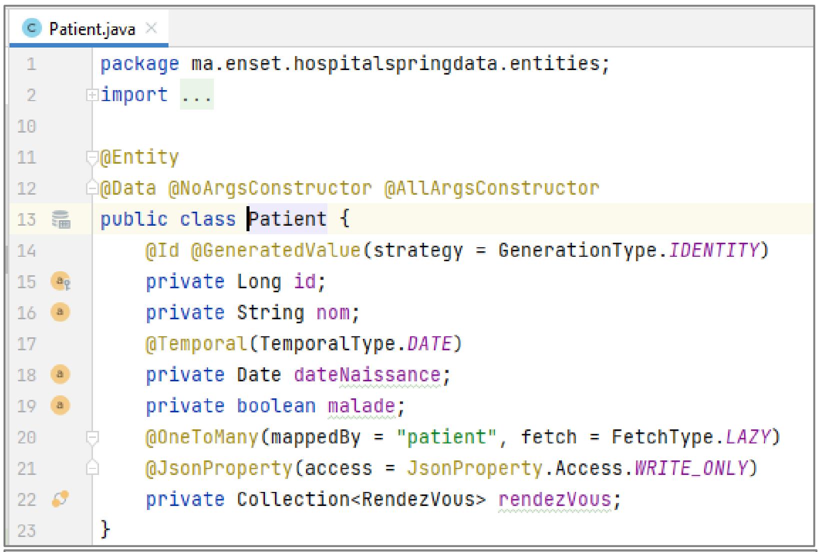
- Chaque Rendez-vous concerne un patient et un médecin.
- Pour chaque rendez-vous on associe une seule consultation issue de rendez-vous.
- Un Patient peut prendre plusieurs rendez-vous

➢ Entities JPA

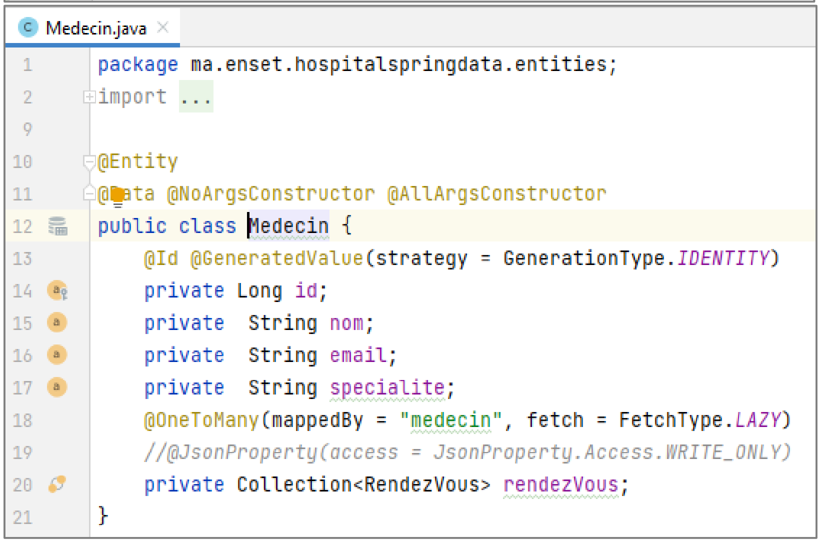




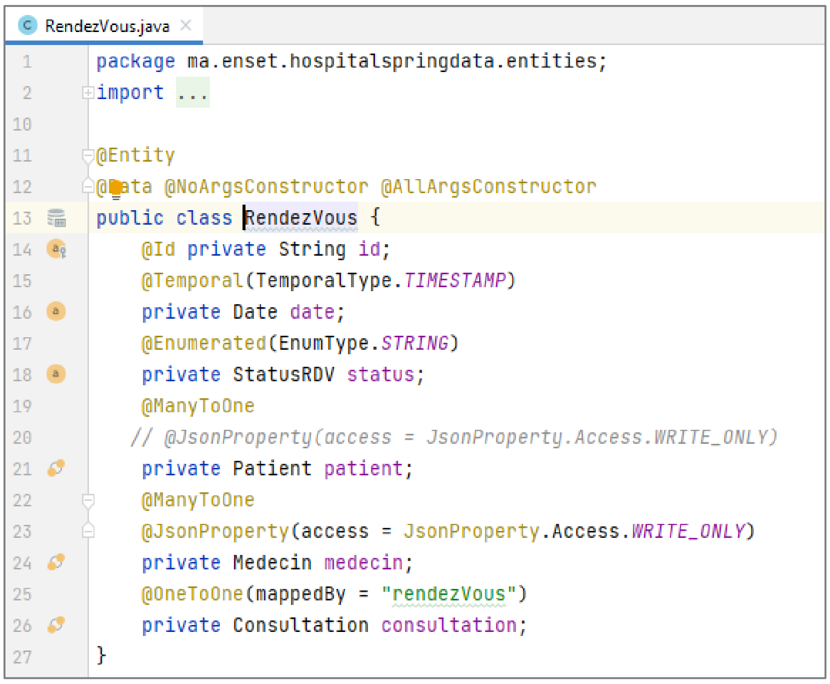
```
spring.datasource.url=jdbc:h2:mem:hospital
spring.h2.console.enabled=true
server.port=8086
spring.jpa.show-sql=true
```


Patient.java

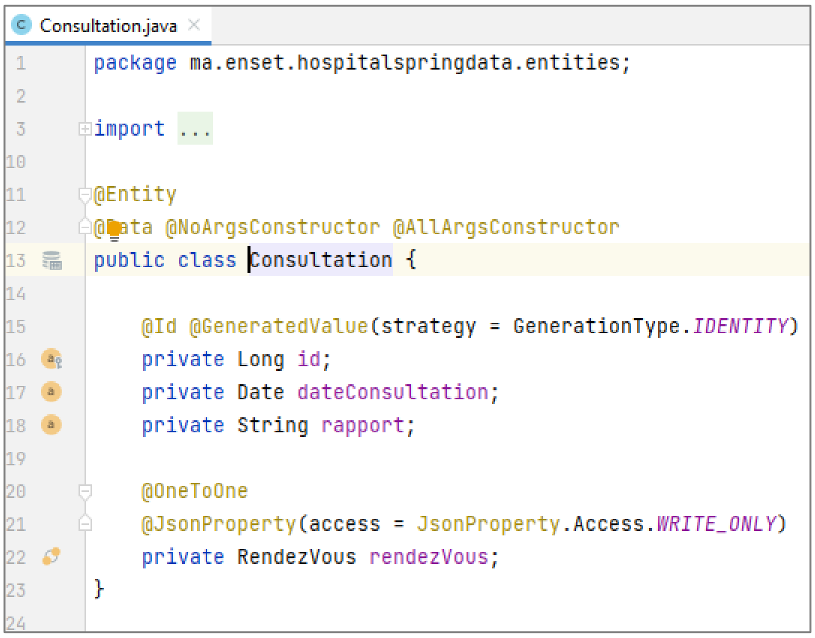
```java
package ma.enset.hospitalspringdata.entities;
import ...

@Entity
@Data @NoArgsConstructor @AllArgsConstructor
public class Patient {
    @Id @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;
    private String nom;
    @Temporal(TemporalType.DATE)
    private Date dateNaissance;
    private boolean malade;
    @OneToMany(mappedBy = "patient", fetch = FetchType.LAZY)
    @JsonProperty(access = JsonProperty.Access.WRITE_ONLY)
    private Collection<RendezVous> rendezVous;
}
```


Medecin.java

```java
package ma.enset.hospitalspringdata.entities;
import ...

@Entity
@Data @NoArgsConstructor @AllArgsConstructor
public class Medecin {
    @Id @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;
    private  String nom;
    private  String email;
    private  String specialite;
    @OneToMany(mappedBy = "medecin", fetch = FetchType.LAZY)
    //@JsonProperty(access = JsonProperty.Access.WRITE_ONLY)
    private Collection<RendezVous> rendezVous;
}
```

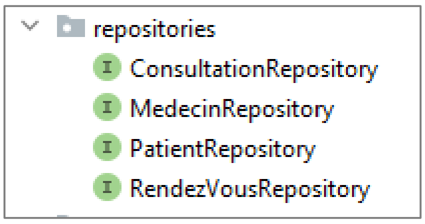

RendezVous.java

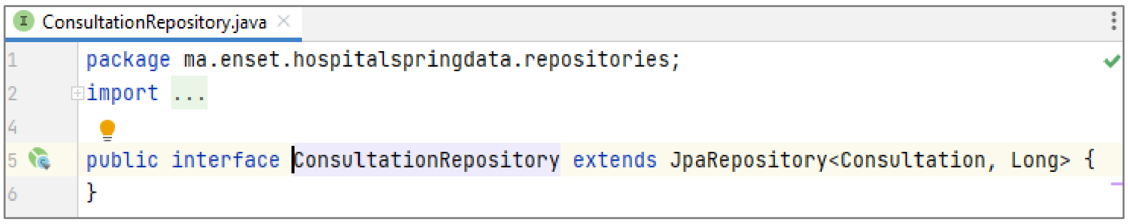
```java
package ma.enset.hospitalspringdata.entities;
import ...

@Entity
@Data @NoArgsConstructor @AllArgsConstructor
public class RendezVous {
    @Id private String id;
    @Temporal(TemporalType.TIMESTAMP)
    private Date date;
    @Enumerated(EnumType.STRING)
    private StatusRDV status;
    @ManyToOne
   // @JsonProperty(access = JsonProperty.Access.WRITE_ONLY)
    private Patient patient;
    @ManyToOne
    @JsonProperty(access = JsonProperty.Access.WRITE_ONLY)
    private Medecin medecin;
    @OneToOne(mappedBy = "rendezVous")
    private Consultation consultation;
}
```



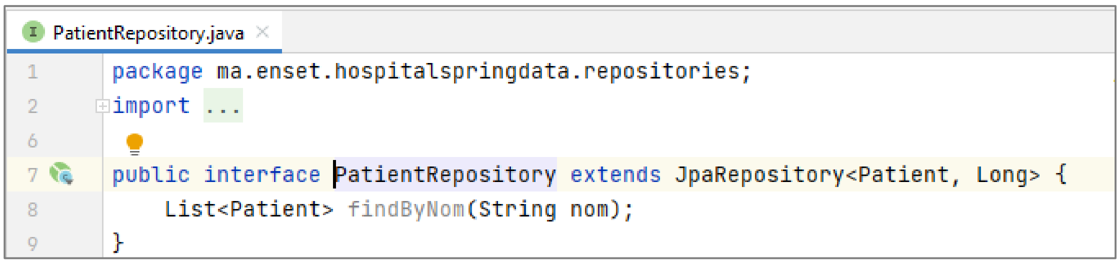
```java
package ma.enset.hospitalspringdata.entities;

import ...

@Entity
@Data @NoArgsConstructor @AllArgsConstructor
public class Consultation {

    @Id @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;
    private Date dateConsultation;
    private String rapport;

    @OneToOne
    @JsonProperty(access = JsonProperty.Access.WRITE_ONLY)
    private RendezVous rendezVous;
}
```

➢ Interfaces DAO basées sur Spring Data

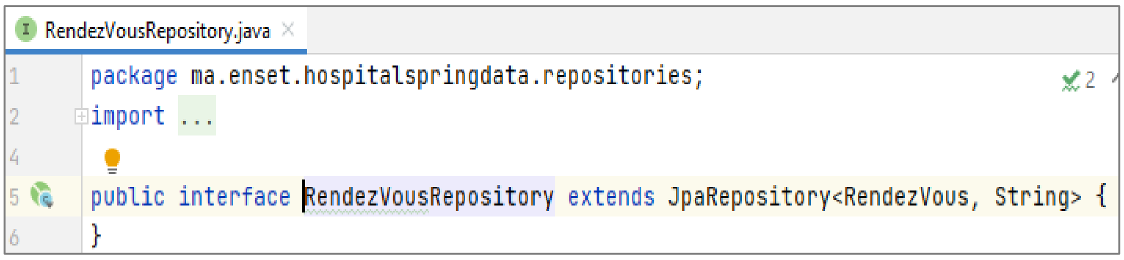




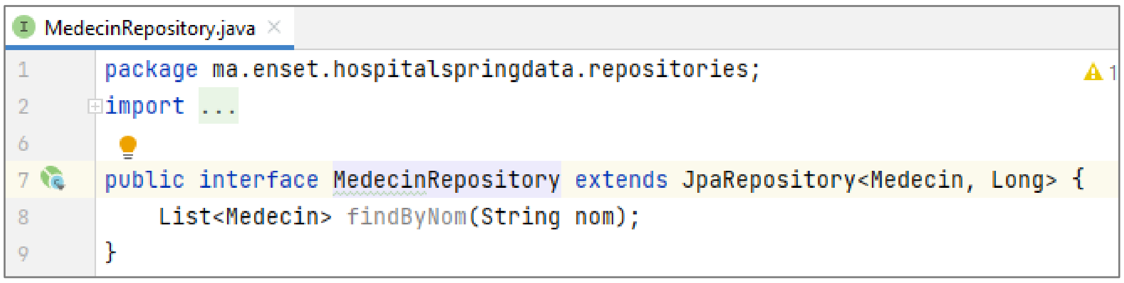
```java
package ma.enset.hospitalspringdata.repositories;
import ...

public interface ConsultationRepository extends JpaRepository<Consultation, Long> {
}
```



```java
package ma.enset.hospitalspringdata.repositories;
import ...

public interface PatientRepository extends JpaRepository<Patient, Long> {
    List<Patient> findByNom(String nom);
}
```



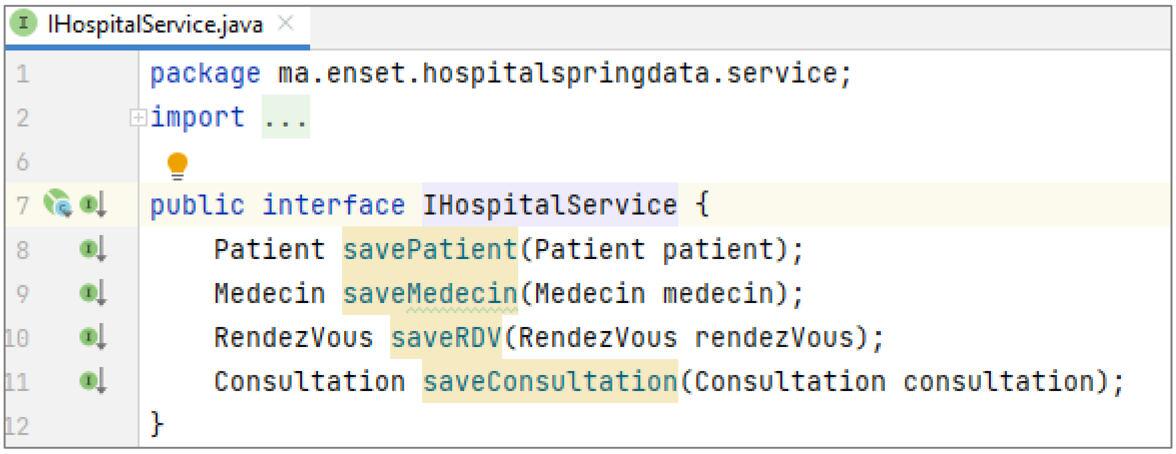
```java
package ma.enset.hospitalspringdata.repositories;
import ...

public interface RendezVousRepository extends JpaRepository<RendezVous, String> {
}
```



```java
package ma.enset.hospitalspringdata.repositories;
import ...

public interface MedecinRepository extends JpaRepository<Medecin, Long> {
    List<Medecin> findByNom(String nom);
}
```

➢ La couche service (métier)

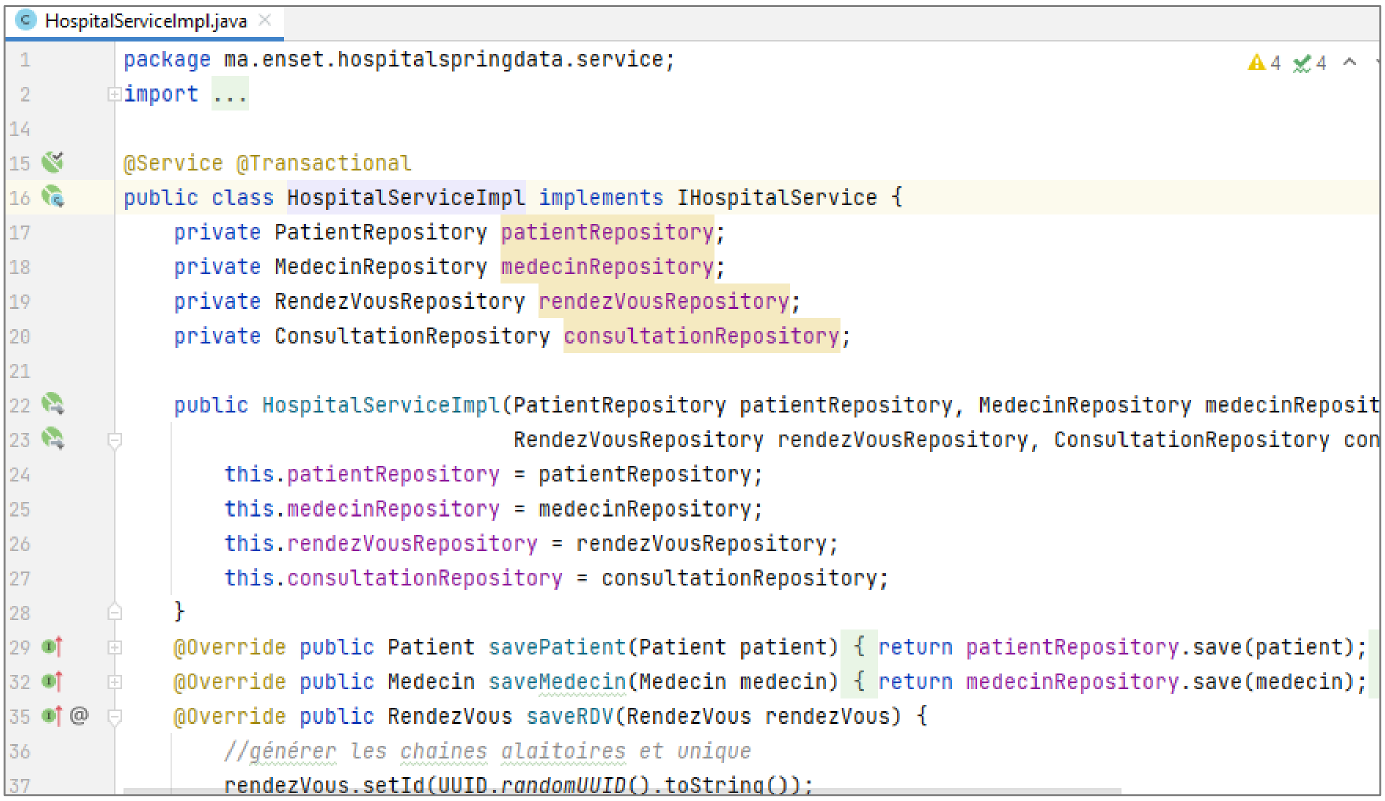
➢ Interface



IHospitalService.java

```java
package ma.enset.hospitalspringdata.service;
import ...

public interface IHospitalService {
    Patient savePatient(Patient patient);
    Medecin saveMedecin(Medecin medecin);
    RendezVous saveRDV(RendezVous rendezVous);
    Consultation saveConsultation(Consultation consultation);
}
```

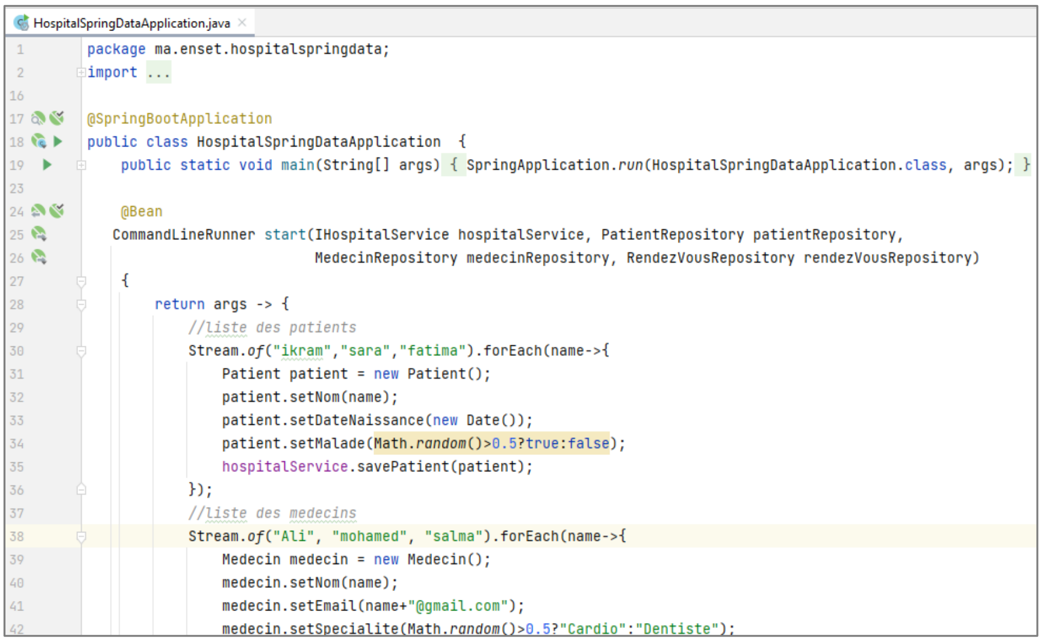
➢ Implémentation



HospitalServiceImpl.java

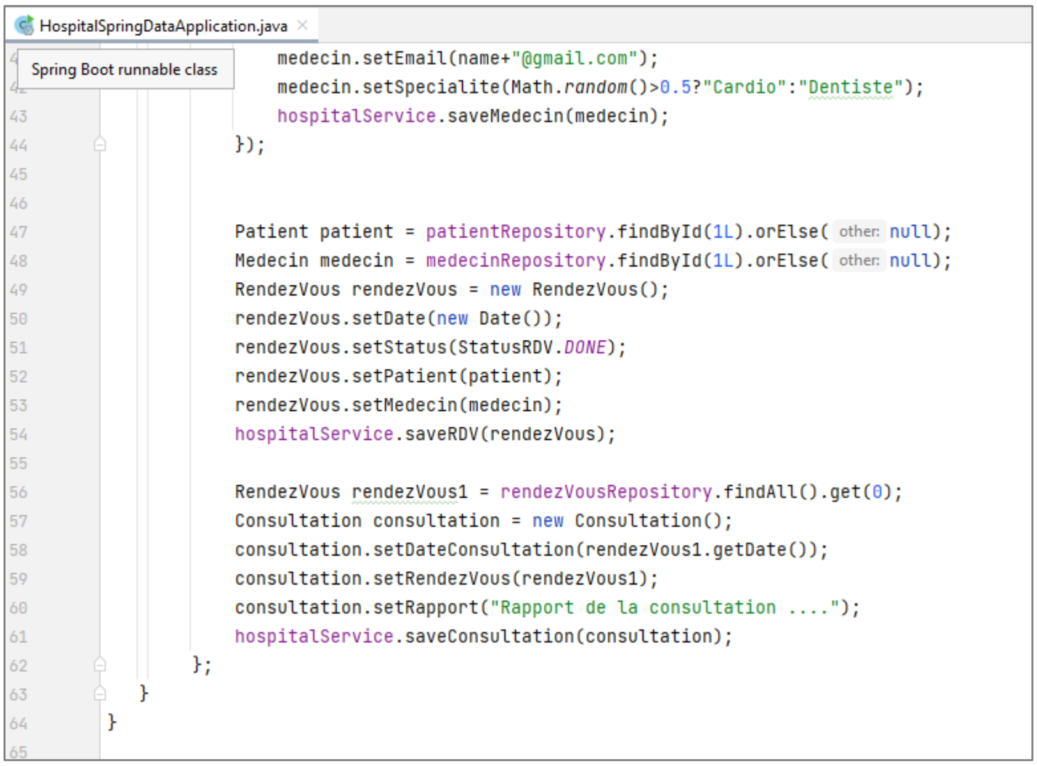
```java
package ma.enset.hospitalspringdata.service;
import ...

@Service @Transactional
public class HospitalServiceImpl implements IHospitalService {
    private PatientRepository patientRepository;
    private MedecinRepository medecinRepository;
    private RendezVousRepository rendezVousRepository;
    private ConsultationRepository consultationRepository;

    public HospitalServiceImpl(PatientRepository patientRepository, MedecinRepository medecinReposit
                               RendezVousRepository rendezVousRepository, ConsultationRepository con
        this.patientRepository = patientRepository;
        this.medecinRepository = medecinRepository;
        this.rendezVousRepository = rendezVousRepository;
        this.consultationRepository = consultationRepository;
    }
    @Override public Patient savePatient(Patient patient) { return patientRepository.save(patient);
    @Override public Medecin saveMedecin(Medecin medecin) { return medecinRepository.save(medecin);
    @Override public RendezVous saveRDV(RendezVous rendezVous) {
        //générer les chaines alaitoires et unique
        rendezVous.setId(UUID.randomUUID().toString());
```

➢ Application



HospitalSpringDataApplication.java

```java
package ma.enset.hospitalspringdata;
import ...

@SpringBootApplication
public class HospitalSpringDataApplication  {
    public static void main(String[] args) { SpringApplication.run(HospitalSpringDataApplication.class, args); }

    @Bean
   CommandLineRunner start(IHospitalService hospitalService, PatientRepository patientRepository,
                           MedecinRepository medecinRepository, RendezVousRepository rendezVousRepository)
    {
        return args -> {
            //liste des patients
            Stream.of("ikram","sara","fatima").forEach(name->{
                Patient patient = new Patient();
                patient.setNom(name);
                patient.setDateNaissance(new Date());
                patient.setMalade(Math.random()>0.5?true:false);
                hospitalService.savePatient(patient);
            });
            //liste des medecins
            Stream.of("Ali", "mohamed", "salma").forEach(name->{
                Medecin medecin = new Medecin();
                medecin.setNom(name);
                medecin.setEmail(name+"@gmail.com");
                medecin.setSpecialite(Math.random()>0.5?"Cardio":"Dentiste");
```

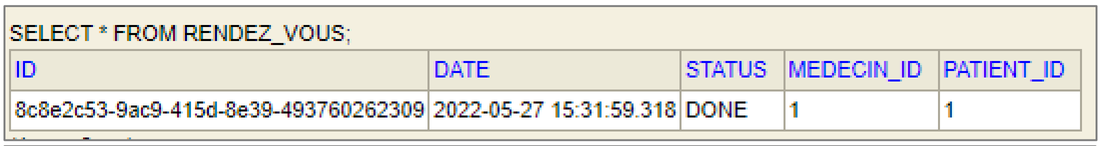


HospitalSpringDataApplication.java

Spring Boot runnable class

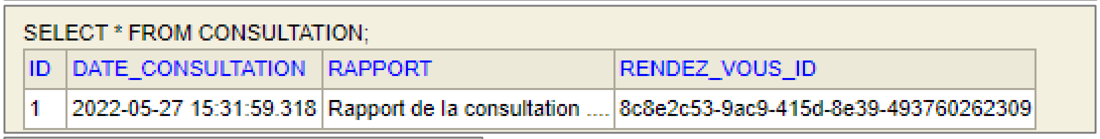
```java
                medecin.setEmail(name+"@gmail.com");
                medecin.setSpecialite(Math.random()>0.5?"Cardio":"Dentiste");
                hospitalService.saveMedecin(medecin);
            });


            Patient patient = patientRepository.findById(1L).orElse( other: null);
            Medecin medecin = medecinRepository.findById(1L).orElse( other: null);
            RendezVous rendezVous = new RendezVous();
            rendezVous.setDate(new Date());
            rendezVous.setStatus(StatusRDV.DONE);
            rendezVous.setPatient(patient);
            rendezVous.setMedecin(medecin);
            hospitalService.saveRDV(rendezVous);

            RendezVous rendezVous1 = rendezVousRepository.findAll().get(0);
            Consultation consultation = new Consultation();
            consultation.setDateConsultation(rendezVous1.getDate());
            consultation.setRendezVous(rendezVous1);
            consultation.setRapport("Rapport de la consultation ....");
            hospitalService.saveConsultation(consultation);
        };
    }
}
```

➢ Résultat



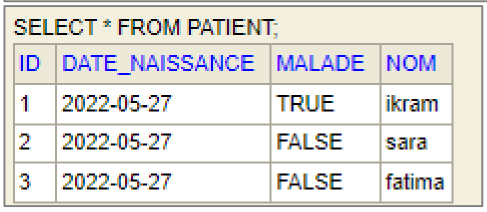
SELECT * FROM RENDEZ_VOUS;

ID	DATE	STATUS	MEDECIN_ID	PATIENT_ID
8c8e2c53-9ac9-415d-8e39-493760262309	2022-05-27 15:31:59.318	DONE	1	1



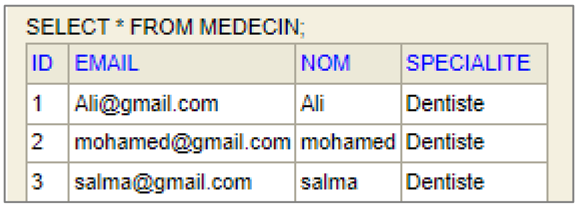
SELECT * FROM CONSULTATION;

ID	DATE_CONSULTATION	RAPPORT	RENDEZ_VOUS_ID
1	2022-05-27 15:31:59.318	Rapport de la consultation	8c8e2c53-9ac9-415d-8e39-493760262309



SELECT * FROM PATIENT;

ID	DATE_NAISSANCE	MALADE	NOM
1	2022-05-27	TRUE	ikram
2	2022-05-27	FALSE	sara
3	2022-05-27	FALSE	fatima

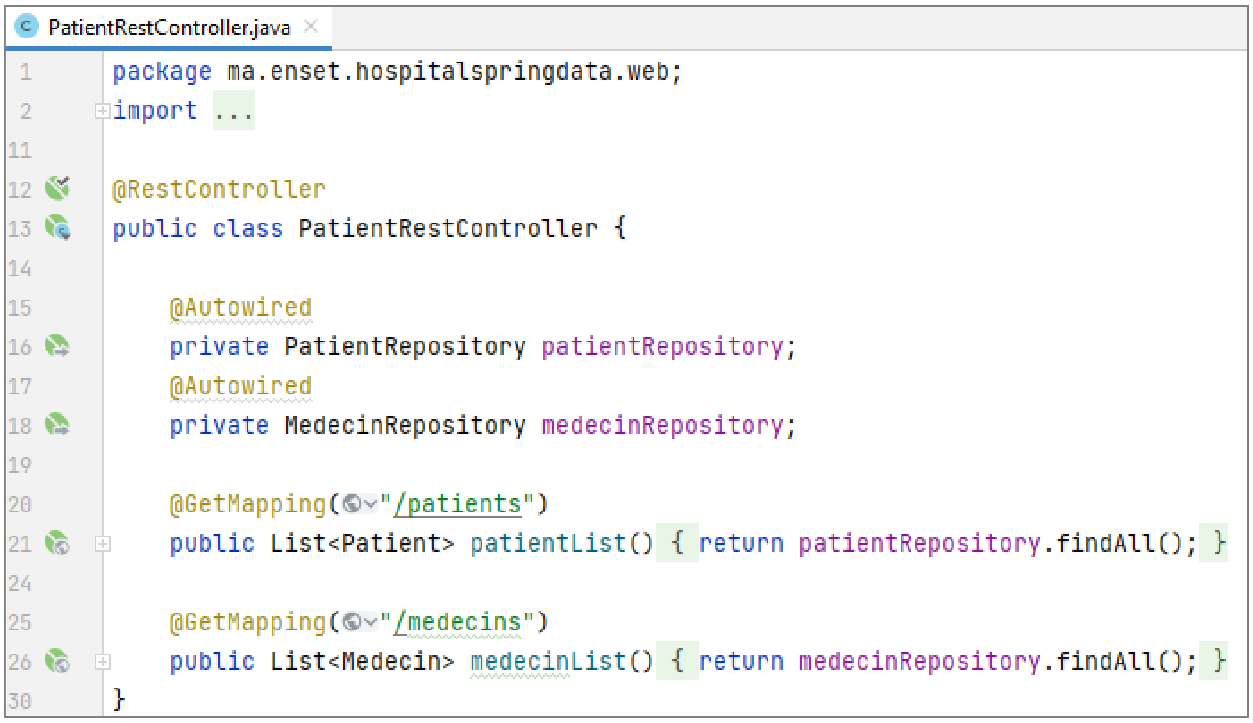


SELECT * FROM MEDECIN;

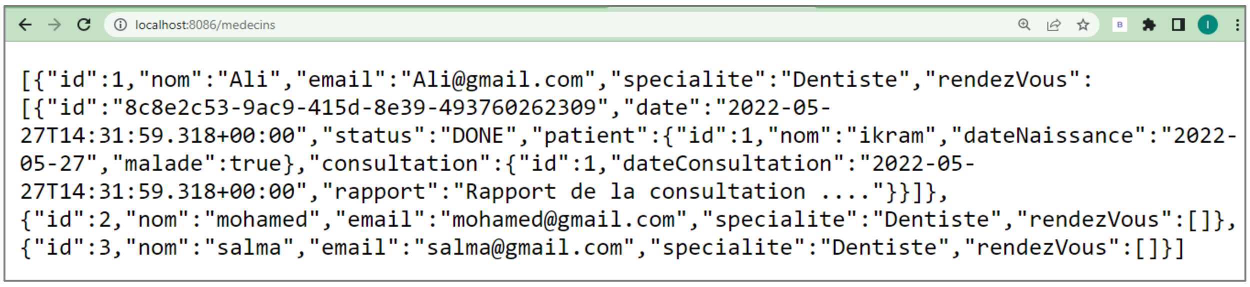
ID	EMAIL	NOM	SPECIALITE
1	Ali@gmail.com	Ali	Dentiste
2	mohamed@gmail.com	mohamed	Dentiste
3	salma@gmail.com	salma	Dentiste

➢ La couche web

• Le contrôleur Spring MVC



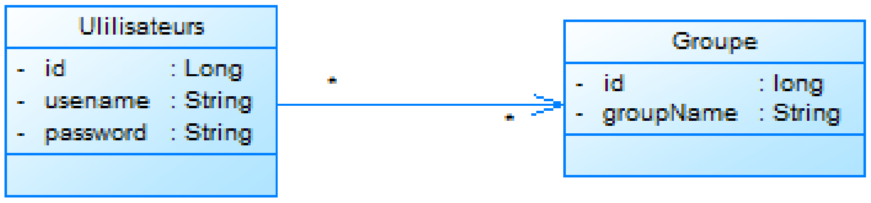
```java
package ma.enset.hospitalspringdata.web;
import ...

@RestController
public class PatientRestController {

    @Autowired
    private PatientRepository patientRepository;
    @Autowired
    private MedecinRepository medecinRepository;

    @GetMapping("/patients")
    public List<Patient> patientList() { return patientRepository.findAll(); }

    @GetMapping("/medecins")
    public List<Medecin> medecinList() { return medecinRepository.findAll(); }
}
```



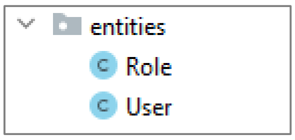
localhost:8086/medecins

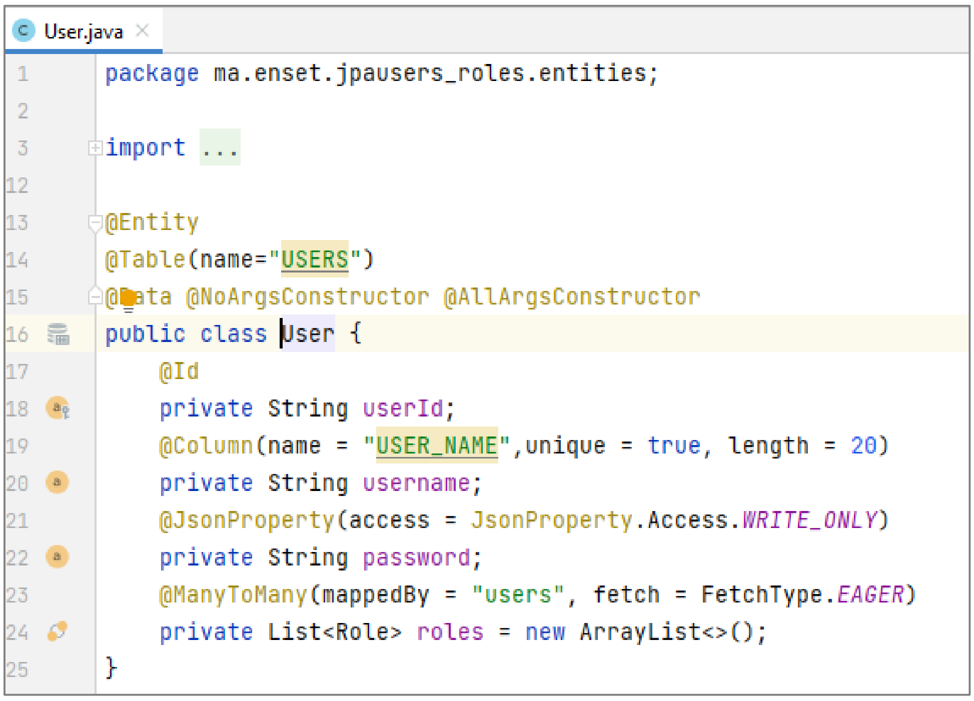
```
[{"id":1,"nom":"Ali","email":"Ali@gmail.com","specialite":"Dentiste","rendezVous":
[{"id":"8c8e2c53-9ac9-415d-8e39-493760262309","date":"2022-05-
27T14:31:59.318+00:00","status":"DONE","patient":{"id":1,"nom":"ikram","dateNaissance":"2022-
05-27","malade":true},"consultation":{"id":1,"dateConsultation":"2022-05-
27T14:31:59.318+00:00","rapport":"Rapport de la consultation ...."}}]},
{"id":2,"nom":"mohamed","email":"mohamed@gmail.com","specialite":"Dentiste","rendezVous":[]},
{"id":3,"nom":"salma","email":"salma@gmail.com","specialite":"Dentiste","rendezVous":[]}]
```

❖ Association ManyToMany

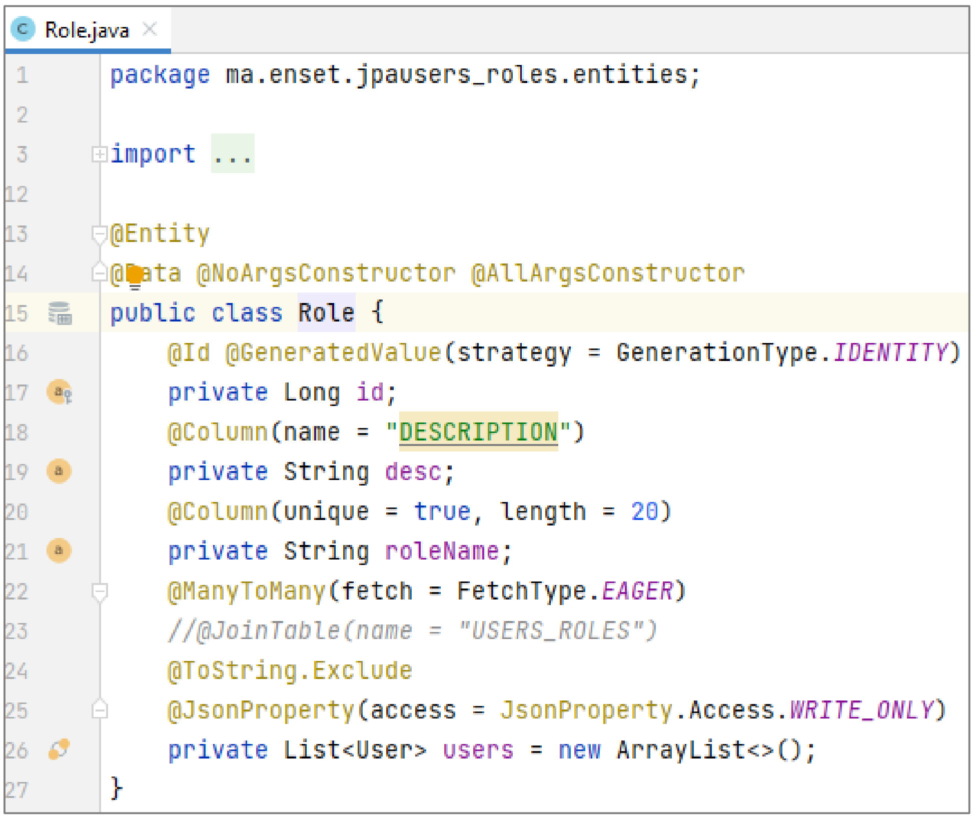


➢ Entities JPA



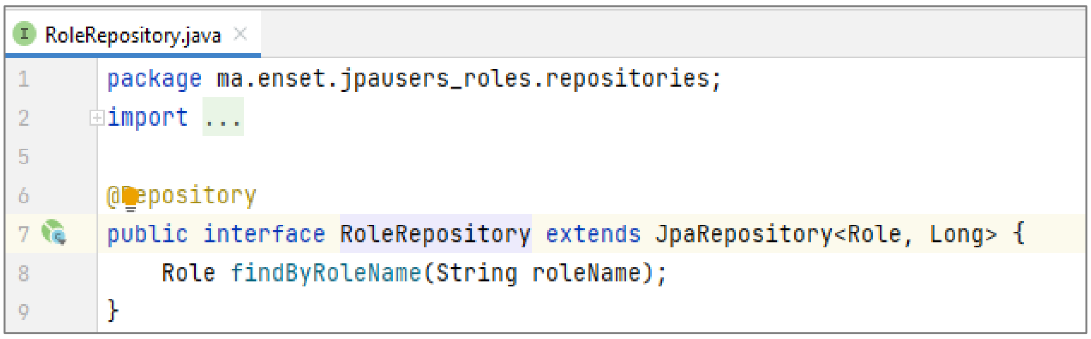

User.java

```java
package ma.enset.jpausers_roles.entities;

import ...

@Entity
@Table(name="USERS")
@Data @NoArgsConstructor @AllArgsConstructor
public class User {
    @Id
    private String userId;
    @Column(name = "USER_NAME",unique = true, length = 20)
    private String username;
    @JsonProperty(access = JsonProperty.Access.WRITE_ONLY)
    private String password;
    @ManyToMany(mappedBy = "users", fetch = FetchType.EAGER)
    private List<Role> roles = new ArrayList<>();
}
```

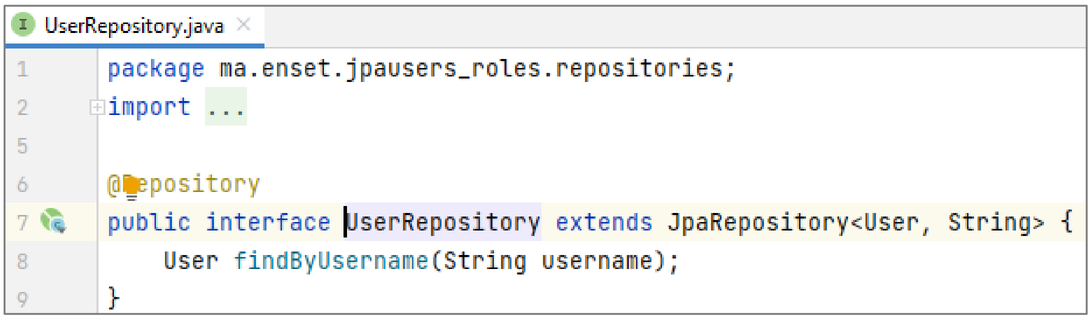

Role.java

```java
package ma.enset.jpausers_roles.entities;

import ...

@Entity
@Data @NoArgsConstructor @AllArgsConstructor
public class Role {
    @Id @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;
    @Column(name = "DESCRIPTION")
    private String desc;
    @Column(unique = true, length = 20)
    private String roleName;
    @ManyToMany(fetch = FetchType.EAGER)
    //@JoinTable(name = "USERS_ROLES")
    @ToString.Exclude
    @JsonProperty(access = JsonProperty.Access.WRITE_ONLY)
    private List<User> users = new ArrayList<>();
}
```

➢ Interfaces DAO basées sur Spring Data


RoleRepository.java

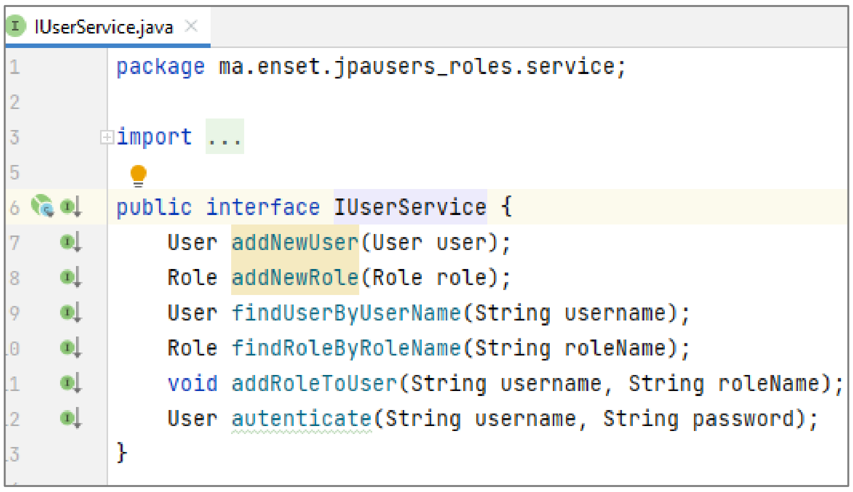
```java
package ma.enset.jpausers_roles.repositories;
import ...

@Repository
public interface RoleRepository extends JpaRepository<Role, Long> {
    Role findByRoleName(String roleName);
}
```


UserRepository.java

```java
package ma.enset.jpausers_roles.repositories;
import ...

@Repository
public interface UserRepository extends JpaRepository<User, String> {
    User findByUsername(String username);
}
```

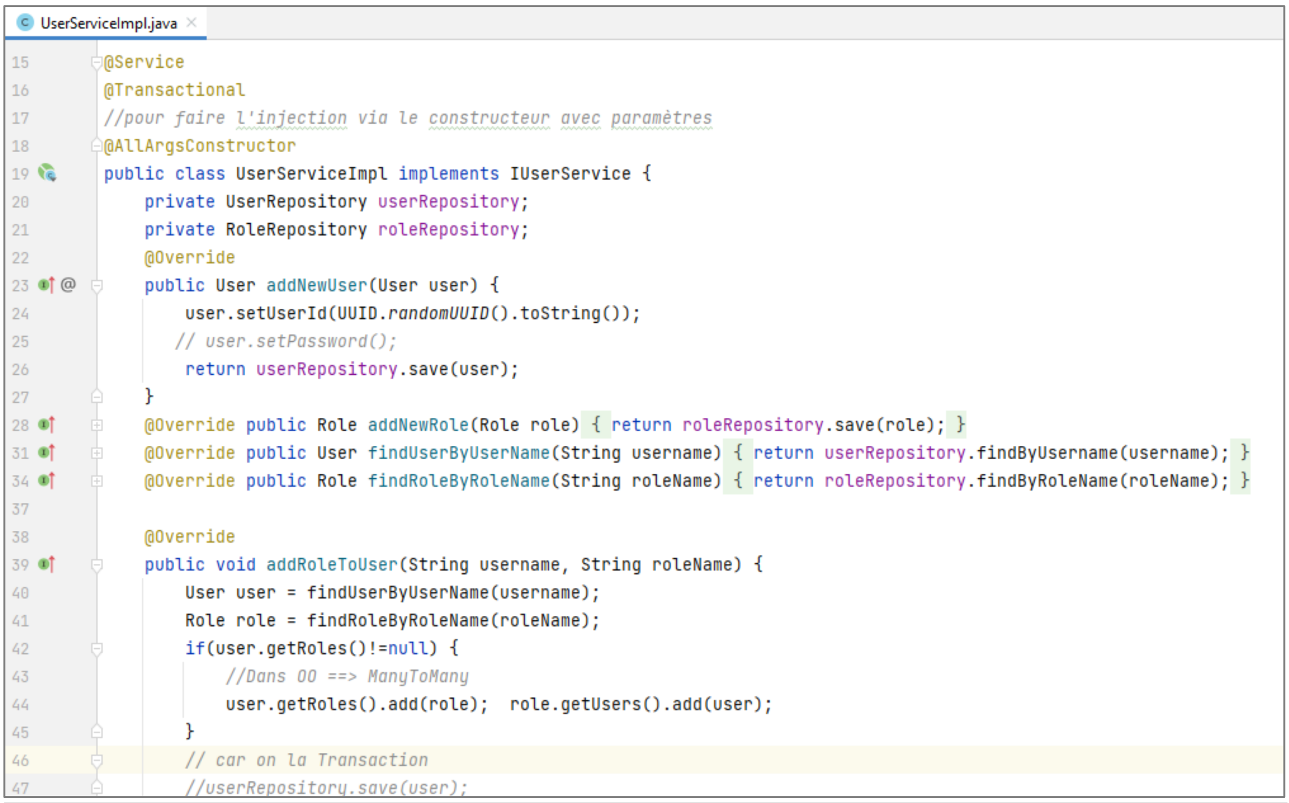
➢ La couche service (métier)

➢ Interface

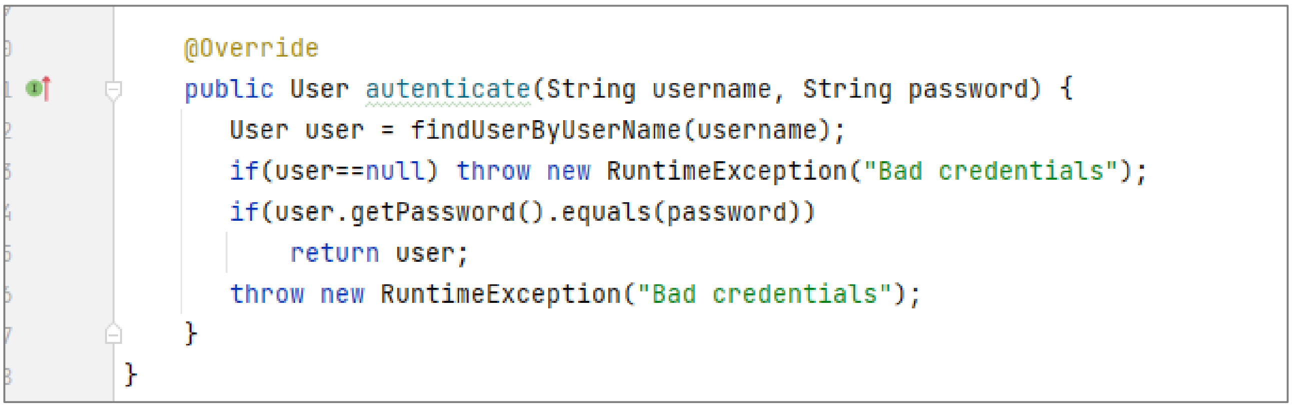

IUserService.java

```java
package ma.enset.jpausers_roles.service;

import ...

public interface IUserService {
    User addNewUser(User user);
    Role addNewRole(Role role);
    User findUserByUserName(String username);
    Role findRoleByRoleName(String roleName);
    void addRoleToUser(String username, String roleName);
    User autenticate(String username, String password);
}
```

➢ Implémentation

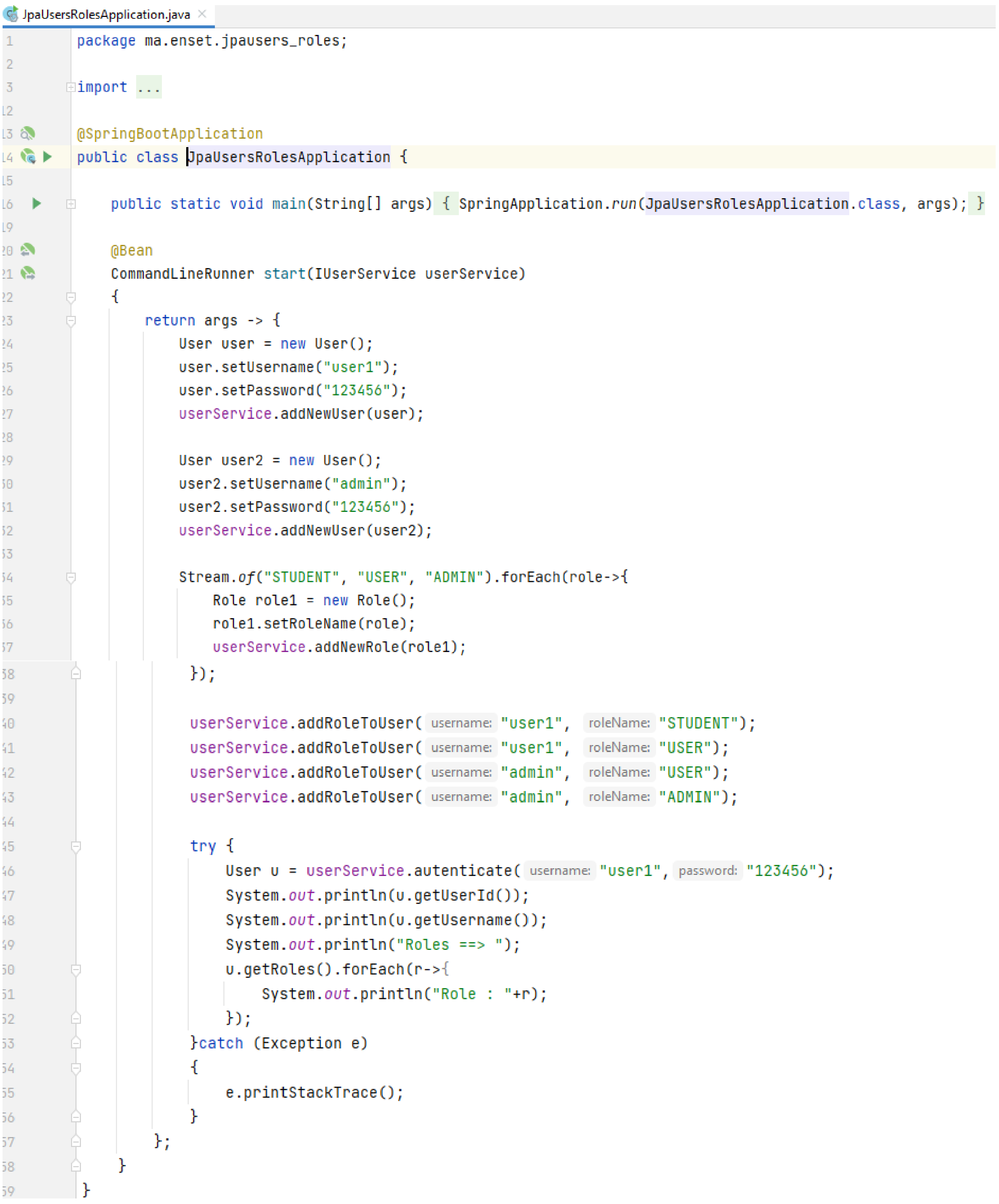

UserServiceImpl.java

```java
@Service
@Transactional
//pour faire l'injection via le constructeur avec paramètres
@AllArgsConstructor
public class UserServiceImpl implements IUserService {
    private UserRepository userRepository;
    private RoleRepository roleRepository;
    @Override
    public User addNewUser(User user) {
        user.setUserId(UUID.randomUUID().toString());
       // user.setPassword();
        return userRepository.save(user);
    }
    @Override public Role addNewRole(Role role) { return roleRepository.save(role); }
    @Override public User findUserByUserName(String username) { return userRepository.findByUsername(username); }
    @Override public Role findRoleByRoleName(String roleName) { return roleRepository.findByRoleName(roleName); }

    @Override
    public void addRoleToUser(String username, String roleName) {
        User user = findUserByUserName(username);
        Role role = findRoleByRoleName(roleName);
        if(user.getRoles()!=null) {
            //Dans OO ==> ManyToMany
            user.getRoles().add(role);  role.getUsers().add(user);
        }
        // car on la Transaction
        //userRepository.save(user);
```



```java
    @Override
    public User autenticate(String username, String password) {
        User user = findUserByUserName(username);
        if(user==null) throw new RuntimeException("Bad credentials");
        if(user.getPassword().equals(password))
            return user;
        throw new RuntimeException("Bad credentials");
    }
}
```

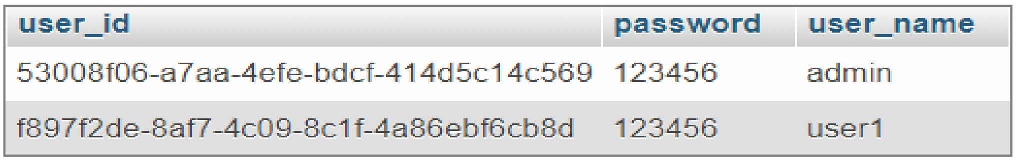
➢ Application



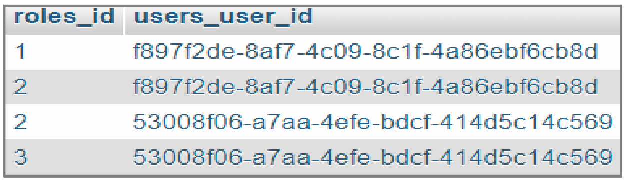
JpaUsersRolesApplication.java

```java
package ma.enset.jpausers_roles;

import ...

@SpringBootApplication
public class JpaUsersRolesApplication {

    public static void main(String[] args) { SpringApplication.run(JpaUsersRolesApplication.class, args); }

    @Bean
    CommandLineRunner start(IUserService userService)
    {
        return args -> {
            User user = new User();
            user.setUsername("user1");
            user.setPassword("123456");
            userService.addNewUser(user);

            User user2 = new User();
            user2.setUsername("admin");
            user2.setPassword("123456");
            userService.addNewUser(user2);

            Stream.of("STUDENT", "USER", "ADMIN").forEach(role->{
                Role role1 = new Role();
                role1.setRoleName(role);
                userService.addNewRole(role1);
            });

            userService.addRoleToUser( username: "user1",  roleName: "STUDENT");
            userService.addRoleToUser( username: "user1",  roleName: "USER");
            userService.addRoleToUser( username: "admin",  roleName: "USER");
            userService.addRoleToUser( username: "admin",  roleName: "ADMIN");

            try {
                User u = userService.autenticate( username: "user1", password: "123456");
                System.out.println(u.getUserId());
                System.out.println(u.getUsername());
                System.out.println("Roles ==> ");
                u.getRoles().forEach(r->{
                    System.out.println("Role : "+r);
                });
            }catch (Exception e)
            {
                e.printStackTrace();
            }
        };
    }
}
```

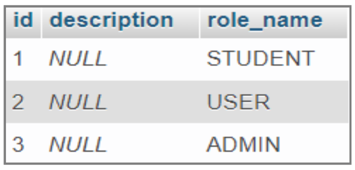
➢ Résultat



user_id	password	user_name
53008f06-a7aa-4efe-bdcf-414d5c14c569	123456	admin
f897f2de-8af7-4c09-8c1f-4a86ebf6cb8d	123456	user1



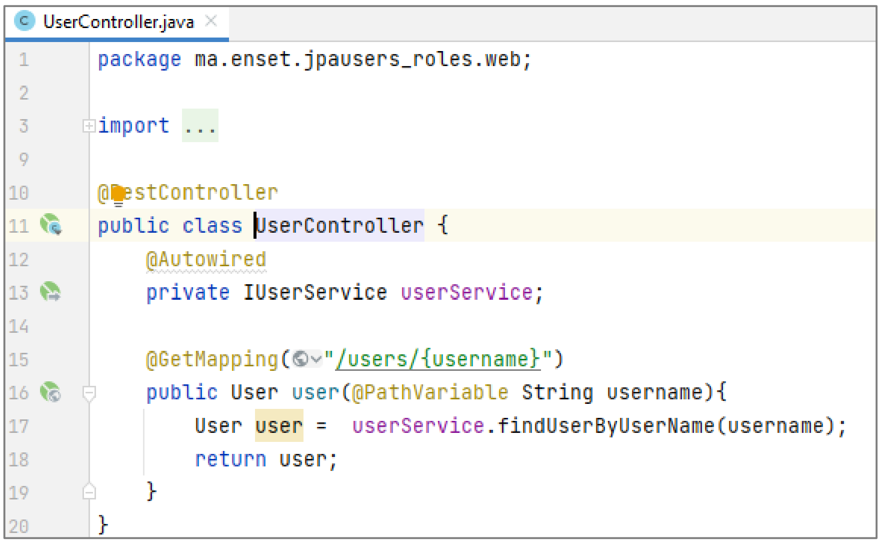
roles_id	users_user_id
1	f897f2de-8af7-4c09-8c1f-4a86ebf6cb8d
2	f897f2de-8af7-4c09-8c1f-4a86ebf6cb8d
2	53008f06-a7aa-4efe-bdcf-414d5c14c569
3	53008f06-a7aa-4efe-bdcf-414d5c14c569



id	description	role_name
1	*NULL*	STUDENT
2	*NULL*	USER
3	*NULL*	ADMIN

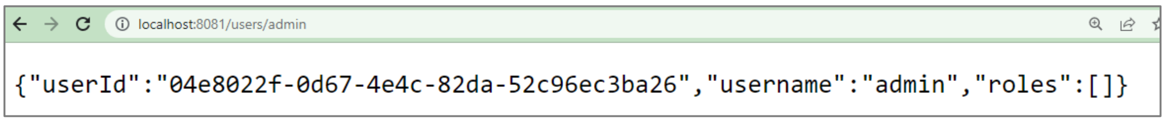
➢ La couche web

• **Le contrôleur Spring MVC**



UserController.java

```java
package ma.enset.jpausers_roles.web;

import ...

@RestController
public class UserController {
    @Autowired
    private IUserService userService;

    @GetMapping("/users/{username}")
    public User user(@PathVariable String username){
        User user =  userService.findUserByUserName(username);
        return user;
    }
}
```



localhost:8081/users/admin

```
{"userId":"04e8022f-0d67-4e4c-82da-52c96ec3ba26","username":"admin","roles":[]}
```