

DEPARTEMENT MATHÉMATIQUES ET INFORMATIQUE

Activité Pratique N°3

Filière :

« Génie du Logiciel et des Systèmes Informatiques Distribués »

GLSID

JPA Hibernate Spring Data

Réalisé par :

Najat ES-SAYYAD

Année Universitaire : 2022-2023

Introduction

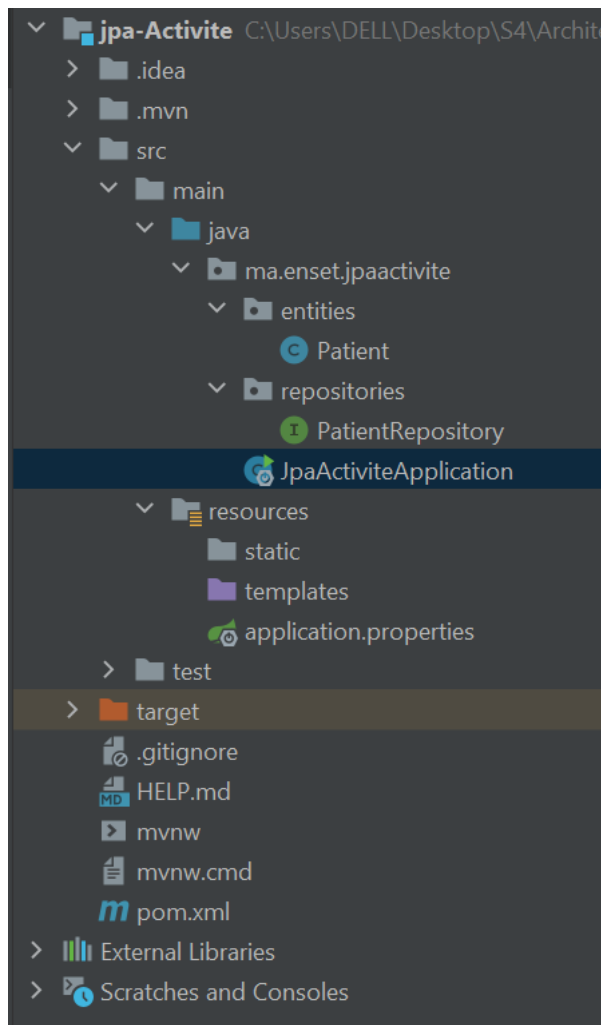
Dans cette activité, nous allons créer un projet Spring Initializer avec les dépendances nécessaires pour notre application, à savoir JPA, H2, Spring Web et Lombok. Ensuite, nous allons créer une entité JPA nommée Patient avec des attributs tels que l'id, le nom, la date de naissance, l'état de santé, et le score. Nous allons ensuite configurer l'unité de persistance dans le fichier application.properties pour que notre application puisse stocker et récupérer les données des patients.

Nous allons créer une interface JPA Repository basée sur Spring data pour effectuer les opérations CRUD (Create, Read, Update, Delete) sur notre entité Patient. Nous allons tester quelques opérations de gestion de patients telles que l'ajout de patients, la consultation de tous les patients, la consultation d'un patient en particulier, la recherche de patients, la mise à jour d'un patient et la suppression d'un patient.

Nous allons également migrer notre base de données H2 vers MySQL pour mieux gérer les données de notre application.

Enfin, nous allons reprendre les exemples précédents et appliquer les mêmes concepts de création d'entités JPA, de configurations de l'unité de persistance et de création d'interfaces JPA Repository pour les entités Médecin, rendez-vous, consultation.

L'architecture du projet :



L'entité JPA Patient ayant les attributs :

- id de type Long
- nom de type String
- dateNaissanec de type Date
- malade de type boolean
- score de type int

```

1 package ma.enset.jpaaactivite.entities;
2
3 import ...
4     16 usages
5
6 @Entity
7 @Data // lombok va ajouter les getters et setters
8 @NoArgsConstructor @AllArgsConstructor
9
10 public class Patient {
11     no usages
12     @Id @GeneratedValue(strategy = GenerationType.IDENTITY)
13     private Long id ;
14     no usages
15     @Column(length = 50)
16     private String nom ;
17     no usages
18     @Temporal(TemporalType.DATE)
19     private Date dateNaissance ;
20     no usages
21     private boolean malade ;
22     no usages
23     private int score ;
24
25 }

```

Configurer l'unité de persistance dans le fichier application.properties

```

1 spring.datasource.url=jdbc:h2:mem:patient-db
2 spring.h2.console.enabled=true
3 server.port=8082

```

Créer l'interface JPA Repository basée sur Spring data

```

1 package ma.enset.jpaaactivite.repositories;
2
3 import ...
4     2 usages
5
6 public interface PatientRepository extends JpaRepository<Patient, Long> {
7     1 usage
8     List<Patient> findByMalade(boolean m);
9     1 usage
10    Page<Patient> findByMalade(boolean m, PageRequest pageable);
11    1 usage
12    List<Patient> findByMaladeAndScoreLessThan(boolean m ,int score);
13
14    1 usage
15    List<Patient> findByMaladeIsTrueAndScoreLessThan(int score);
16    1 usage
17    List<Patient> findByDateNaissanceBetweenAndMaladeIsTrueOrNomLike(Date d1, Date d2, String mc);
18    1 usage
19    @Query("select p from Patient p where p.nom like :x and p.score < :y")
20    List<Patient> recherchePatients(@Param("x") String nom ,@Param("y")int scoreMin);
21
22 }

```

L'interface Web H2

English ▼ [Preferences](#) [Tools](#) [Help](#)

Login

Saved Settings: Generic H2 (Embedded) ▼

Setting Name: Save Remove

Driver Class:

JDBC URL:

User Name:

Password:

Connect Test Connection

La base de donnée sous H2:

← → ↻ ⓘ localhost:8082/h2-console/login.do?jsessionid=391ee1bf51014b5767f139bab4de7065

🎵 | 🛠️ | ☒ Auto commit | 🔄 | Max rows: 1000 | 🟢 | 🟡 | 🔴 | 📄 | Auto complete: Off | Auto select: On | ⓘ

📁 jdbc:h2:mem:patient-db

- 📄 PATIENT
 - 📄 ID
 - 📄 DATE_NAISSANCE
 - 📄 DATE
 - 📄 MALADE
 - 📄 NOM
 - 📄 CHARACTER VARYING(50)
 - 📄 SCORE
 - 📄 Indexes
- 📄 INFORMATION_SCHEMA
- 📄 Users

📘 H2 2.1.214 (2022-06-13)

Run Run Selected Auto complete Clear SQL statement:

SELECT * FROM PATIENT PATIENT

Important Commands

Tester quelques opérations de gestion de patients :

- Ajouter des patients
- Consulter tous les patients

localhost:8082/h2-console/login.do?jsessionid=...

Auto Max 1000 Auto complete Auto select

jdbc:h2:mem:patient-db
PATIENT
INFORMATION_SCHEMA
Users
H2 2.1.214 (2022-06-13)

Run Run Selected Auto complete Clear SQL statement:
SELECT * FROM PATIENT

SELECT * FROM PATIENT;

ID	DATE_NAISSANCE	MALADE	NOM	SCORE
1	2023-03-29	FALSE	Najat	870
2	2023-03-29	TRUE	Imane	100
3	2023-03-29	FALSE	Salma	234

(3 rows, 3 ms)

Edit

- supprimer un patient

localhost:8082/h2-console/login.do?jsessionid=...

Auto Max 1000 Auto complete Auto select

jdbc:h2:mem:patient-db
PATIENT
INFORMATION_SCHEMA
Users
H2 2.1.214 (2022-06-13)

Run Run Selected Auto complete Clear SQL statement:
SELECT * FROM PATIENT

SELECT * FROM PATIENT;

ID	DATE_NAISSANCE	MALADE	NOM	SCORE
1	2023-03-29	FALSE	Najat	870
3	2023-03-29	FALSE	Salma	234

(2 rows, 2 ms)

Edit

- Ajouter des patients

localhost:8082/h2-console/login....

Auto Max 1000 Auto complete Auto select

jdbc:h2:mem:patient-db

PATIENT

INFORMATION_SCHEMA

Users

H2 2.1.214 (2022-06-13)

Run Run Selected Auto complete Clear SQL statement:

SELECT * FROM PATIENT

SELECT * FROM PATIENT;

ID	DATE_NAISSANCE	MALADE	NOM	SCORE
1	2023-03-29	FALSE	Najat	870
3	2023-03-29	FALSE	Najat	61
4	2023-03-29	FALSE	Najat	49
5	2023-03-29	FALSE	Najat	40
6	2023-03-29	FALSE	Najat	82
7	2023-03-29	FALSE	Najat	22
8	2023-03-29	FALSE	Najat	86
9	2023-03-29	FALSE	Najat	79
10	2023-03-29	FALSE	Najat	1
11	2023-03-29	FALSE	Najat	10
12	2023-03-29	FALSE	Najat	84
13	2023-03-29	FALSE	Najat	73
14	2023-03-29	FALSE	Najat	10
15	2023-03-29	FALSE	Najat	98
16	2023-03-29	FALSE	Najat	68
17	2023-03-29	FALSE	Najat	72
18	2023-03-29	FALSE	Najat	54
19	2023-03-29	FALSE	Najat	31

Migrer de H2 Database vers MySQL

```
1 spring.datasource.url=jdbc:mysql://localhost:3306/DBP?createDatabaseIfNotExist=true
2 spring.datasource.username=root
3 spring.datasource.password=
4 server.port=8082
5 spring.jpa.hibernate.ddl-auto=update
6 spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MariaDBDialect
7 spring.jpa.show-sql=true
```

Résultat

```
-----  
1  
Najat  
2023-03-29  
96  
false  
-----
```

```
2  
Najat  
2023-03-29  
36  
false  
-----
```

```
3  
Najat  
2023-03-29  
55  
false  
-----
```

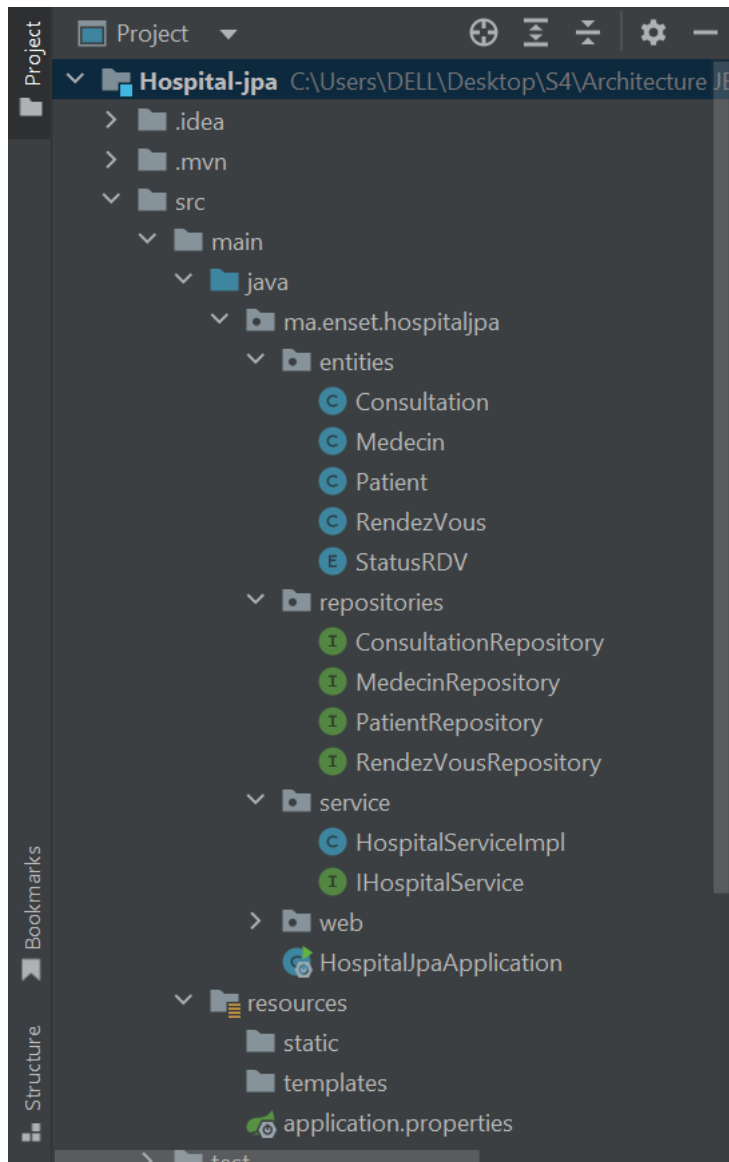
```
4  
Najat  
2023-03-29  
33  
false  
-----
```

```
5  
Najat  
2023-03-29  
74  
false  
=====
```

```
Najat  
false
```


Reprendre les exemples du Patient, Médecin, rendez-vous, consultation

L'architecture du projet :



Les entités JPA :

Entité Patient :

```
1 package ma.enset.hospitaljpa.entities;
2
3 import ...
4
5
6
7
8
9
10 @Entity
11 @Data @NoArgsConstructor @AllArgsConstructor
12 public class Patient {
13     no usages
14     @Id @GeneratedValue(strategy = GenerationType.IDENTITY)
15     private Long id ;
16     no usages
17     @Column(length = 50)
18     private String nom ;
19     no usages
20     @Temporal(TemporalType.DATE)
21     private Date dateNaissance ;
22     no usages
23     private boolean malade ;
24     no usages
25     @OneToMany(mappedBy = "patient", fetch=FetchType.LAZY)
26     private Collection<RendezVous> rendezVous;
27 }
```

Entité Médecin :

```
1 package ma.enset.hospitaljpa.entities;
2
3 import ...
4
5
6
7
8
9
10 @Entity
11 @Data @NoArgsConstructor @AllArgsConstructor
12 public class Medecin {
13     no usages
14     @Id
15     @GeneratedValue(strategy = GenerationType.IDENTITY)
16     private Long id;
17     no usages
18     private String nom ;
19     no usages
20     private String email;
21     no usages
22     private String specialite;
23     no usages
24     @OneToMany(mappedBy = "medecin" , fetch = FetchType.LAZY)
25     @JsonProperty(access = JsonProperty.Access.WRITE_ONLY)
26     private Collection<RendezVous> rendezVous;
27 }
```

Entité Consultation :

```
1 package ma.enset.hospitaljpa.entities;
2
3 import ...
4     11 usages
5
6 @Entity
7 @Data @NoArgsConstructor @AllArgsConstructor
8 public class Consultation {
9     no usages
10
11     @Id
12     @GeneratedValue(strategy = GenerationType.IDENTITY)
13     private Long id;
14     no usages
15
16     private Date dateConsultation;
17     no usages
18
19     private String rapport;
20     no usages
21
22     @OneToOne
23     @JsonProperty(access = JsonProperty.Access.WRITE_ONLY)
24     private RendezVous rendezVous;
25
26 }
```

Entité Rendez-vous :

```
10 @Entity
11 @Data
12 @NoArgsConstructor @AllArgsConstructor
13 public class RendezVous {
14     no usages
15
16     @Id // @GeneratedValue(strategy = GenerationType.IDENTITY)
17     // private Long id;
18     private String id;
19     no usages
20
21     private Date date;
22     no usages
23
24     @Enumerated(EnumType.STRING)
25     private StatusRDV status;
26     no usages
27
28     @ManyToOne
29     @JsonProperty(access = JsonProperty.Access.WRITE_ONLY)
30     private Patient patient;
31     no usages
32
33     @ManyToOne
34     private Medecin medecin;
35     no usages
36
37     @OneToOne(mappedBy = "rendezVous")
38     private Consultation consultation;
39
40 }
```

```

1 package ma.enset.hospitaljpa.entities;
2
3 public enum StatusRDV {
4     PENDING,
5     CANCELED,
6     DONE
7 }

```

Les interfaces JPA :

PatientRepository :

```

1 package ma.enset.hospitaljpa.repositories;
2
3 import ma.enset.hospitaljpa.entities.Patient;
4 import org.springframework.data.jpa.repository.JpaRepository;
5
6 public interface PatientRepository extends JpaRepository<Patient, Long> {
7     Patient findByNom(String name);
8 }
9

```

MedecinRepository :

```

1 package ma.enset.hospitaljpa.repositories;
2
3 import ma.enset.hospitaljpa.entities.Medecin;
4 import org.springframework.data.jpa.repository.JpaRepository;
5
6 public interface MedecinRepository extends JpaRepository<Medecin, Long> {
7     Medecin findByNom(String name);
8 }
9
10

```

ConsultationRepository :

```
1 package ma.enset.hospitaljpa.repositories;
2
3 import ma.enset.hospitaljpa.entities.Consultation;
4 import org.springframework.data.jpa.repository.JpaRepository;
5
6 public interface ConsultationRepository extends JpaRepository<Consultation, Long> {
7 }
8
```

RendezVousRepository :

```
1 package ma.enset.hospitaljpa.repositories;
2
3 import ma.enset.hospitaljpa.entities.RendezVous;
4 import org.springframework.data.jpa.repository.JpaRepository;
5
6 public interface RendezVousRepository extends JpaRepository<RendezVous, String> {
7 }
8
```

L'interface Service :

```
1 package ma.enset.hospitaljpa.service;
2
3 import ma.enset.hospitaljpa.entities.Consultation;
4 import ma.enset.hospitaljpa.entities.Medecin;
5 import ma.enset.hospitaljpa.entities.Patient;
6 import ma.enset.hospitaljpa.entities.RendezVous;
7
8 public interface IHospitalService {
9     Patient savePatient(Patient patient);
10    Medecin saveMedecin(Medecin medecin);
11    RendezVous saveRDV(RendezVous rendezVous);
12    Consultation saveConsultation(Consultation consultation);
13
14 }
```

L'implémentation de l'interface Service :





































```
1 package ma.enset.hospitaljpa.service;
2
3 import ...
4
15 no usages
16 @Service
17 @Transactional
18 public class HospitalServiceImpl implements IHospitalService {
19     no usages
20     public HospitalServiceImpl(PatientRepository patientRepository, MedecinRepository medecinRepository,
21                               RendezVousRepository rendezVousRepository,
22                               ConsultationRepository consultationRepository) {
23         this.patientRepository = patientRepository;
24         this.medecinRepository = medecinRepository;
25         this.rendezVousRepository = rendezVousRepository;
26         this.consultationRepository = consultationRepository;
27     }
28
29     2 usages
30     private PatientRepository patientRepository;
31     2 usages
32     private MedecinRepository medecinRepository;
33     2 usages
34     private RendezVousRepository rendezVousRepository;
35     2 usages
36     private ConsultationRepository consultationRepository;
37
38     1 usage
39     @Override
40     public Patient savePatient(Patient patient) { return patientRepository.save(patient); }
41
42     1 usage
43     @Override
44     public Medecin saveMedecin(Medecin medecin) { return medecinRepository.save(medecin); }
45
46     1 usage
47     @Override
48     public RendezVous saveRDV(RendezVous rendezVous) {
49         rendezVous.setId(UUID.randomUUID().toString());
50         return rendezVousRepository.save(rendezVous);
51     }
52
53     1 usage
54     @Override
55     public Consultation saveConsultation(Consultation consultation) {
56         return consultationRepository.save(consultation);
57     }
58 }
59
60 }
```

```

16 1 usage
   @SpringBootApplication
17  public class HospitalJpaApplication {
      no usages
18  public static void main(String[] args) {
19      SpringApplication.run(HospitalJpaApplication.class, args);}
20
21      /*...*/
22
23  }
24
25  no usages
26
27  @Bean
28  CommandLineRunner start(IHospitalService hospitalService,
29                          PatientRepository patientRepository,
30                          MedecinRepository medecinRepository,
31                          RendezVousRepository rendezVousRepository){
32
33      return args -> {
34          Stream.of( ...values: "Mohamed", "Hassan", "Najat")
35                  .forEach(name->{
36                      Patient patient=new Patient();
37                      patient.setNom(name);
38                      patient.setDateNaissance(new Date());
39                      patient.setMalade(false);
40                      hospitalService.savePatient(patient);
41                  });
42
43          Stream.of( ...values: "aymane", "hanane", "yasmine")
44                  .forEach(name->{
45                      Medecin medecin=new Medecin();
46                      medecin.setNom(name);
47                      medecin.setEmail(name+"@gmail.com");
48                      medecin.setSpecialite(Math.random()>0.5?"Cardio":"Dentiste");
49                      hospitalService.saveMedecin(medecin);
50                  });
51
52          Patient patient=patientRepository.findById(1L).orElse( other: null);
53          Patient patient1=patientRepository.findByNom( name: "Mohamed");
54
55          Medecin medecin=medecinRepository.findByNom( name: "yasmine");
56
57          RendezVous rendezVous=new RendezVous();
58          rendezVous.setDate(new Date());
59          rendezVous.setStatus(StatusRDV.PENDING);
60          rendezVous.setMedecin(medecin);
61          rendezVous.setPatient(patient);
62          RendezVous saveRDV= hospitalService.saveRDV(rendezVous);
63          System.out.println(saveRDV.getId());
64
65          // RendezVous rendezVous1=rendezVousRepository.findById(1L).orElse(null);
66
67          RendezVous rendezVous1=rendezVousRepository.findAll().get(0);
68
69      }
70
71  }
72
73  }
74
75  }
76
77  }
78
79  }
80
81  }
82
83  }
84
85  }
86
87  }
88
89  }
90
91  }
92
93  }
94
95  }
96
97  }
98
99  }
100

```

Base de donnée :

	jdbc:h2:mem:hospitaljpa
	CONSULTATION
	ID
	DATE_CONSULTAION
	RAPPORT
	RENDEZ_VOUS_ID
	 Indexes
	MEDECIN
	ID
	EMAIL
	NOM
	SPECIALITE
	 Indexes
	PATIENT
	ID
	DATE_NAISSANCE
	MALADE
	NOM
	 Indexes
	RENDEZ_VOUS
	ID
	DATE
	STATUS
	MEDECIN_ID
	PATIENT_ID
	 Indexes
	 INFORMATION_SCHEMA
	 Users
	 SA
	H2 2.1.214 (2022-06-13)

Résultat :

```
[
  {
    "id": 1,
    "nom": "Mohamed",
    "dateNaissance": "2023-04-26",
    "malade": false,
    "rendezVous": [
      {
        "id": "7bcdee8c-7590-428c-bc47-a1f382f833d9",
        "date": "2023-04-26T20:42:03.132+00:00",
        "status": "PENDIND",
        "medecin": {
          "id": 3,
          "nom": "yasmine",
          "email": "yasmine@gmail.com",
          "specialite": "Dentiste"
        },
        "consultation": {
          "id": 1,
          "dateConsultaion": "2023-04-26T20:42:03.178+00:00",
          "rapport": "Rapport de la consultation ....."
        }
      }
    ]
  },
  {
    "id": 2,
    "nom": "Hassan",
    "dateNaissance": "2023-04-26",
    "malade": false,
    "rendezVous": []
  },
  {
    "id": 3,
    "nom": "Najat",
    "dateNaissance": "2023-04-26",
    "malade": false,
    "rendezVous": []
  }
]
```