Activité N° : 2

JPA Hibernate, Spring Data

Réalisé Par :

Manal Namir

Encadré Par :

Mohamed YOUSSFI

Année Universitaire 2022 – 2023

Créer 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

```java
import lombok.Data;
import lombok.NoArgsConstructor;

import java.util.Date;
18 usages
@Entity
@Data
@NoArgsConstructor
@AllArgsConstructor

// classe pour gerer les patients
public class Patient {
    no usages
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;
    no usages
    @Column(length = 50 )
    private String nom;
    no usages
    @Temporal(TemporalType.DATE)
    private Date dateNaissance;
    no usages
    private boolean malade;
    no usages
    private int score;
}
```

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

```
spring.datasource.url=jdbc:mysql://localhost:3306/DBP?createDatabaseIfNotExist=true
Spring.datasource.username=root
Spring.datasource.password=
#spring.h2.console.enabled=true
server.port=8082
spring.jpa.hibernate.ddl-auto=update
spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MariaDBDialect
spring.jpa.show-sql=true
```

Créer l'interface JPA Repository basée sur Spring data

ent.java | ApiApApplication.java | PatientRepository.java | application.properties | pom.xml (api-ap)

```
import org.springframework.data.domain.Pageable;
import org.springframework.data.jpa.repository.JpaRepository;
import org.springframework.data.jpa.repository.Query;
import org.springframework.data.repository.query.Param;

import java.util.Date;
import java.util.List;

// le mapping objet relationnel
2 usages
public interface PatientRepository extends JpaRepository<Patient, Long> {
    //afficher que les patients malades et la pagination dans une methode creer
    no usages
    public List<Patient> findByMalade(boolean m);
    1 usage
    public Page<Patient> findByMalade(boolean m, Pageable pageable);
    no usages
    List<Patient> findByMaladeAndScoreLessThan(boolean m,int score);
    no usages
    List<Patient> findByMaladeIsTrueAndScoreLessThan(int score);
    no usages
    List<Patient> findByDateNaissanceBetweenAndMaladeIsTrueOrNomLike(Date d1,Date d2,String mc);
    // autre solution que des meths long
    no usages
    @Query("select p from Patient p where p.dateNaissance between :x and :y or p.nom like :z")
    List<Patient> chercherPatients(@Param("x") Date d1, @Param("y")Date d2, @Param("z")String nom);
    1 usage
    @Query("select p from Patient p where p.nom like :x and p.score<:y")
    List<Patient> chercherPatients(@Param("x") String nom,@Param("y")int scoreMin);
}
```

Tester quelques opérations de gestion de patients :

```java
@SpringBootApplication
public class ApiApApplication implements CommandLineRunner {
    6 usages
    @Autowired
    private PatientRepository patientRepository;
    no usages
    public static void main(String[] args) {

        SpringApplication.run(ApiApApplication.class, args);
    }

    @Override
    public void run(String... args) throws Exception {
        for (int i = 0;i < 100; i++){
            patientRepository.save(new Patient( id: null, nom: "manal",new Date(),Math.random()>0.5?true:false,(int)(Math.random()*100)));
        }

        Page<Patient> patients= patientRepository.findAll(PageRequest.of( page: 0, size: 10));//pagination
        System.out.println("total des pages :"+patients.getTotalPages());
        System.out.println("total des elements :"+patients.getTotalElements());
        System.out.println("numero page :"+patients.getNumber());
        List<Patient> content = patients.getContent();
        Page<Patient> byMalade = patientRepository.findByMalade( m: true,PageRequest.of( page: 0, size: 4));
        List<Patient> patientList=patientRepository.chercherPatients( nom: "%m%", scoreMin: 40);
```

```java
    byMalade.forEach(p->{
        /*content.forEach(p->{*/
        System.out.println("**********************");
        System.out.println(p.getId());
        System.out.println(p.getNom());
        System.out.println(p.getScore());
        System.out.println(p.getDateNaissance());
        System.out.println(p.isMalade());
    });
    System.out.println("**********************");
    Patient patient = patientRepository.findById(1L).orElse( other: null);//ou bien getReferenceById(new Long(1))
    if(patient!=null){
        System.out.println(patient.getNom());
        System.out.println(patient.isMalade());
    }
    patient.setScore(870);
    patientRepository.save(patient);
    //patientRepository.deleteById(1L);
}
```