Activité Pratique N°2 - ORM, JPA Hibernate Spring Data

1.

2. Créer un projet Spring Initializer avec les dépendances JPA, H2, Spring Web et Lombock

3. 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

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

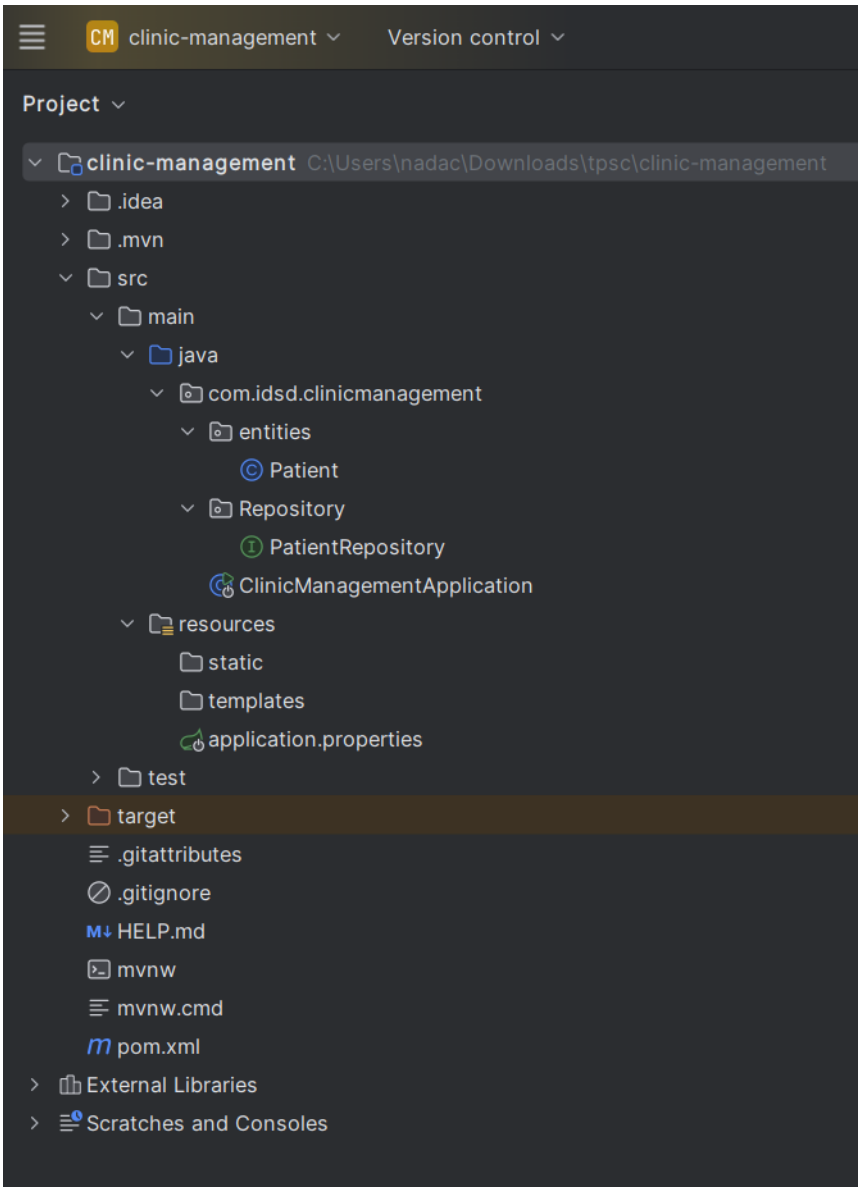
5. Créer l'interface JPA Repository basée sur Spring data

6. Tester quelques opérations de gestion de patients :

- Ajouter des patients
- Consulter tous les patients
- Consulter un patient
- Chercher des patients
- Mettre à jour un patient
- supprimer un patient

7. Migrer de H2 Database vers MySQL

8. Reprendre les exemples du Patient, Médecin, rendez-vous, consultation, users et roles


clinic-management
Version control
Project
clinic-management C:\Users\nadac\Downloads\tpsc\clinic-management
.idea
.mvn
src
main
java
com.idsd.clinicmanagement
entities
Patient
Repository
PatientRepository
ClinicManagementApplication
resources
static
templates
application.properties
test
target
.gitattributes
.gitignore
HELP.md
mvnw
mvnw.cmd
pom.xml
External Libraries
Scratches and Consoles

```java
package com.idsd.clinicmanagement.entities;
import jakarta.persistence.*;
import lombok.*;
import java.util.Date;
@Entity
@Data @NoArgsConstructor @AllArgsConstructor @Builder
public class Patient {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;

    private String nom;

    @Temporal(TemporalType.DATE)
    private Date dateNaissance;

    private boolean malade;

    private int score;
}
```

```java
package com.idsd.clinicmanagement.Repository;

import com.idsd.clinicmanagement.entities.Patient;
import org.springframework.data.jpa.repository.JpaRepository;
import org.springframework.stereotype.Repository;
import java.util.List;

@Repository  2 usages
public interface PatientRepository extends JpaRepository<Patient, Long> {
    // Recherche des patients par nom
    List<Patient> findByNomContains(String nom);  1 usage
}
```

```java
package com.idsd.clinicmanagement;
import com.idsd.clinicmanagement.entities.Patient;
import com.idsd.clinicmanagement.Repository.PatientRepository;
import org.springframework.data.repository.Repository;
import org.springframework.boot.CommandLineRunner;
import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
import org.springframework.context.annotation.Bean;
import java.util.Date;
import java.util.Optional;
@SpringBootApplication
public class ClinicManagementApplication {

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

    @Bean
    CommandLineRunner demo(PatientRepository repo) {
        return String[] args -> {
            // Ajouter des patients
            repo.save(new Patient( id: null,  nom: "Alice", new Date(),  malade: false,  score: 85));
            repo.save(new Patient( id: null,  nom: "Nada", new Date(),  malade: true,  score: 24));
            repo.save(new Patient( id: null,  nom: "Amira ", new Date(),  malade: true,  score: 25));

            // Consulter tous les patients
            System.out.println("Tous les patients:");
            repo.findAll().forEach(System.out::println);

            // Consulter un patient par ID
            Optional<Patient> patientOptional = repo.findById(1L);
            if (patientOptional.isPresent()) {
                Patient p = patientOptional.get();
                System.out.println("Patient id=1: " + p);
```

```java
                System.out.println("Patient id=1: " + p);

                // Mettre à jour le patient
                p.setScore(95);
                repo.save(p);
            } else {
                System.out.println("Patient avec id=1 non trouvé.");
            }

            // Chercher des patients par nom
            System.out.println("Patients contenant 'Al':");
            repo.findByNomContains("Al").forEach(System.out::println);

            // Supprimer un patient (id=2)
            if (repo.existsById(2L)) {
                repo.deleteById(2L);
                System.out.println("Patient avec id=2 supprimé.");
            } else {
                System.out.println("Patient avec id=2 non trouvé pour suppression.");
            }
        };
    }
}
```

```
# Application Name
spring.application.name=clinicManagement

# Connexion MySQL
spring.datasource.url=jdbc:mysql://localhost:3306/clinic_db?useSSL=false&serverTimezone=UTC
spring.datasource.username=root
spring.datasource.password=

# JPA / Hibernate Configuration
spring.jpa.hibernate.ddl-auto=update
spring.jpa.show-sql=true
spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MySQL8Dialect
spring.jpa.properties.hibernate.format_sql=true

# Thymeleaf (si utilisé)
spring.thymeleaf.cache=false
```