Activité Pratique Spring MVC Thymeleaf

Réaliser par :

Mohamed amine KHAMMOUR

Professeur :

Mr. Mohamed YOUSSFI

Enonce d’activité

Activité Pratique Spring MVC Thymeleaf

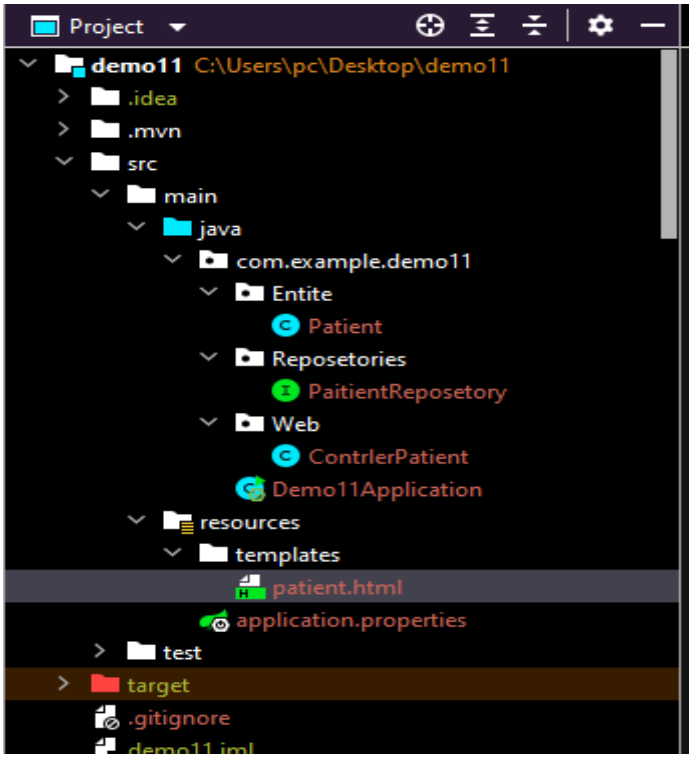
Mohamed YOUSSFI • 22 mars

100 points

Voir la vidéo :Créer une application Web JEE basée sur Spring MVC, Thylemeaf et Spring Data JPA qui permet de gérer les patients. L'application doit permettre les fonctionnalités suivantes :

- Afficher les patients
- Faire la pagination
- Chercher les patients
- Supprimer un patient
- Faire des améliorations supplémentaires

Structure du projet :



La configuration du projet :

```
#spring.datasource.url=jdbc:h2:mem:PATIENT_DB
#spring.h2.console.enabled=true
server.port=8086
spring.datasource.url=jdbc:mysql://localhost:3306/PATIENT_DB?createDatabase
IfNotExist=true
spring.datasource.username = root
spring.datasource.password =
spring.jpa.hibernate.ddl-auto = update
spring.jpa.properties.hibernate.dialect =
org.hibernate.dialect.MariaDBDialect
spring.jpa.show-sql = false
```

La class patient

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


import javax.persistence.*;
import java.util.Date;
@Data
@AllArgsConstructor
@NoArgsConstructor
@Entity
public class Patient {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id ;
    private String nom ;
    @Temporal(TemporalType.DATE)
    private Date date;
    private Boolean malad;
    private  int score;
}
```

Les dépendances du projet :

```xml
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
https://maven.apache.org/xsd/maven-4.0.0.xsd">
    <modelVersion>4.0.0</modelVersion>
    <parent>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-parent</artifactId>
        <version>2.6.5</version>
        <relativePath/> <!-- lookup parent from repository -->
    </parent>
    <groupId>com.example</groupId>
    <artifactId>demo11</artifactId>
    <version>0.0.1-SNAPSHOT</version>
    <name>demo11</name>
    <description>demo11</description>
    <properties>
        <java.version>1.8</java.version>
    </properties>
    <dependencies>
        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-data-jpa</artifactId>
        </dependency>
        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-thymeleaf</artifactId>
        </dependency>
        <!-- https://mvnrepository.com/artifact/org.webjars/bootstrap -->
        <dependency>
            <groupId>org.webjars</groupId>
            <artifactId>bootstrap</artifactId>
            <version>5.1.3</version>
```

```
        </dependency>


        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-web</artifactId>
        </dependency>
        <dependency>
            <groupId>mysql</groupId>
            <artifactId>mysql-connector-java</artifactId>
        </dependency>
<!--         <dependency>-->
<!--             <groupId>com.h2database</groupId>-->
<!--             <artifactId>h2</artifactId>-->
<!--             <scope>runtime</scope>-->
<!--         </dependency>-->
        <dependency>
            <groupId>org.projectlombok</groupId>
            <artifactId>lombok</artifactId>
            <optional>true</optional>
        </dependency>
        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-test</artifactId>
            <scope>test</scope>
        </dependency>
    </dependencies>

    <build>
        <plugins>
            <plugin>
                <groupId>org.springframework.boot</groupId>
                <artifactId>spring-boot-maven-plugin</artifactId>
                <configuration>
                    <excludes>
                        <exclude>
                            <groupId>org.projectlombok</groupId>
                            <artifactId>lombok</artifactId>
                        </exclude>
                    </excludes>
                </configuration>
            </plugin>
        </plugins>
    </build>

</project>
```

L'interface patientReposetory : contient le prototype de la fonction qui permet de retourner un patient par son nom et aussi par son scor :

```
package com.example.demo11.Reposetories;

import com.example.demo11.Entite.Patient;
import org.springframework.data.domain.Page;
import org.springframework.data.domain.Pageable;
import org.springframework.data.jpa.repository.JpaRepository;

public interface   PaitientReposetory extends JpaRepository<Patient,Long> {
```

```
    Page<Patient> findByNomContainsAndScoreEquals(String nom,int scor,
Pageable pageable);
}
```

le Reposetory Web qui contient la class controllerPatient :

```
    @GET(path="/index")
    public String patiates(Model model ,
                           @Request Param(name ="page" ,defaultValue = "0") int page,
                           @Request Param(name = "size",defaultValue = "5") int size,
                           @Request Param(name = "keyword",defaultValue ="") String keyword,
                           @Request Param(name = "scor",defaultValue ="") int scor)

    {
        Page<Patient> pagePatient= paitientRepositery.findByNomContainsAndScoreEquals(keyword,scor,PageRequest.of(page,size));
        model.addAttribute( attributeName: "PagePtient",pagePatient.getContent());
        model.addAttribute( attributeName: "nombrePages",new int[pagePatient.getTotalPages()]);
        model.addAttribute( attributeName: "currentPage",page);
        model.addAttribute( attributeName: "keyword",keyword);
        model.addAttribute( attributeName: "scor",scor);
        return "patient";
    }
```

- **Dans le chemin `ressources/templates`, j'ai créé le template `patients.html`, ce template est traité par le moteur de rendu thymeleaf, reçoit les données nécessaires à travers le controlleur.**
- **Dans ce template j'ai inclu les framework bootstrap et jquery qui sont stocké dans les ressources statiques.**

```
<!DOCTYPE html>
<html lang="en" xmlns:th="http://www.thymeleaf.org">
<head>
    <meta charset="UTF-8" >
    <title>Title</title>
    <link rel="stylesheet"
href="/webjars/bootstrap/5.1.3/css/bootstrap.min.css">


</head>
<body>

    <div class="container">
        <div class="card">
            <div class="card-header" style="text-align: center; " >
listes des patient </div>
            <div class="card-body">
                <form method="get" th:action="@{index}" style="text-
align: center; ">
                    <label>Key word</label>
                    <input type="text" name="keyword"
th:value="${keyword}">
                    <label>Scor</label>
                    <input type="number" name="scor"
th:value="${scor}">
                    <button type="submit" class="btn btn-
primary">Chercher</button>
                </form>

                <table class="table">
```

```
                <thead>
                <tr>

<th>Id</th><th>Nom</th><th>Date</th><th>Malad</th><th>Score</th>
                </tr>
                </thead>
                <tbody>
                <tr th:each="p:${PagePtient}">
                    <td th:text="${p.id}"></td>
                    <td th:text="${p.nom}"></td>
                    <td th:text="${p.date}"></td>
                    <td th:text="${p.malad}"></td>
                    <td th:text="${p.score}"></td>
                    <td>
                    <a onclick="return confirm('Etes vous sure?')"
class="btn btn-danger" th:href="@{delete(id=${p.id},
keyword=${keyword}, page=${currentPage},scor=${scor})}">
                        Delete
                    </a>
                    </td>
                </tr>
                </tbody>
                </table>
                <ul class="nav nav-pills" style="text-align: center;
">
                    <li th:each="pangeN,status:${nombrePages}"
style="text-align: center; ">
                    <a  th:class="${status.index==currentPage?'btn
btn-primary ms-1':'btn btn-outline-primary ms-1'}"
                        th:text="${status.index}"

th:href="@{index(page=${status.index},keyword=${keyword},scor=${scor}
)}">

                    </a>
                    </li>
                </ul>

            </div>

        </div>
    </div>



</body>
</html>
```

Class Application :

```java
@SpringBootApplication
public class Demo11Application {

    public static void main(String[] args) { SpringApplication.run(Demo11Application.class, args); }
@Bean
    CommandLineRunner commandLineRunner(PaitientRepositery paitientRepositery){
        return args -> {
            paitientRepositery.save(new Patient( id: null, nom: "amine",new Date(), malad: false, score: 140)) ;
            paitientRepositery.save(new Patient( id: null, nom: "khalid",new Date(), malad: true, score: 90)) ;
            paitientRepositery.save(new Patient( id: null, nom: "hatim",new Date(), malad: false, score: 50)) ;
            paitientRepositery.save(new Patient( id: null, nom: "rafik",new Date(), malad: true, score: 410)) ;

            paitientRepositery.findAll().forEach(patient -> {
                System.out.println(patient.getNom());
            });

        };
}
}
```

Résultat final :