

## Compte rendu du TP6 :

La classe **patient**

```
@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;
    private int score;
}
```

La classe **patientRepository** qui contient les méthodes pour de traitement de la table patient

```
public interface PatientRepository extends JpaRepository<Patient, Long> {
    Page<Patient> findByNomContains(String kw, Pageable pageable);
}
```

La classe **patientController** :

La méthode **patients** qui permet de récupérer la liste des patients par mot clé

```
public class PatientController {
    private PatientRepository patientRepository;

    @GetMapping(path = "/index")
    public String patients(Model model,
        @RequestParam(name = "page", defaultValue = "0") int page,
        @RequestParam(name = "size", defaultValue = "5") int size,
        @RequestParam(name = "keyword", defaultValue = "") String keyword ){
        Page<Patient> pagePatients=patientRepository.findByNomContains(keyword, PageRequest.of(page, size));
        model.addAttribute( attributeName: "listPatients",pagePatients.getContent());
        model.addAttribute( attributeName: "pages", new int[pagePatients.getTotalPages()]);
        model.addAttribute( attributeName: "currentPage", page);
        model.addAttribute( attributeName: "keyword", keyword);
        return "patients";
    }
}
```

La méthode **delete** qui permet de supprimer un patient

```
@GetMapping("/delete")
public String delete(Long id, String keyword, int page) {
    patientRepository.deleteById(id);
    return "redirect:/index?page="+page+"&keyword="+keyword;
}
```

La méthode **home** pour afficher la page accueil

```
@GetMapping("/")
public String home() { return "redirect:/index"; }
```

La méthode **listPatients** pour afficher la liste des patients en général

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

La classe **properties** qui contient les informations sur la connexion à la base de données

```
spring.datasource.url = jdbc:mysql://localhost:3306/patients?createDatabaseIfNotExist=true
spring.datasource.username = root
spring.datasource.password =
server.port=8083
spring.jpa.hibernate.ddl-auto = create
spring.jpa.properties.hibernate.dialect = org.hibernate.dialect.MariaDBDialect
spring.jpa.show-sql=true
```

La page html pour afficher le résultat dans une page web

```
<!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 mt-2">
    <div class="card">
        <div class="card-header">Liste des Patients</div>
        <div class="card-body">
            <form method="get" th:action="@{index}">
                <label>Key Word</label>
                <input type="text" name="keyword" th:value="${keyword}">
                <button type="submit" class="btn btn-primary">Chercher</button>
            </form>
            <table class="table">
                <thead>
                    <tr>
                        <th>ID</th>
                        <th>Nom</th>
                        <th>DateNaissance</th>
                        <th>Malade</th>
                        <th>Score</th>
                    </tr>
```

La classe Main

```
public static void main(String[] args) { SpringApplication.run(Tp6Application.class, args); }

@Bean
CommandLineRunner commandLineRunner(PatientRepository patientRepository){
    return args -> {
        patientRepository.save(
            new Patient( id: null, nom: "soukaina", new Date(), malade: false, score: 12)
        );
        patientRepository.save(
            new Patient( id: null, nom: "salma", new Date(), malade: false, score: 12)
        );
        patientRepository.save(
            new Patient( id: null, nom: "soukaina1", new Date(), malade: false, score: 12)
        );
        patientRepository.save(
            new Patient( id: null, nom: "salma1", new Date(), malade: false, score: 12)
        );
        patientRepository.findAll().forEach(p ->{
            System.out.println(p.getNom());
        });
    };
};
```

## La classe Security

```
@EnableWebSecurity
public class securityConfig extends WebSecurityConfigurerAdapter {

    @Override
    protected void configure(AuthenticationManagerBuilder auth) throws Exception {
        PasswordEncoder passwordEncoder=passwordEncoder();
        String encodedPWD=passwordEncoder.encode( rawPassword: "1234");
        //stocker en mémoire les utilisateurs qui ont acces à l'application
        auth.inMemoryAuthentication().withUser( username: "user").password(encodedPWD).roles("USER");
        auth.inMemoryAuthentication().withUser( username: "admin").password(passwordEncoder.encode( rawPassword: "5678"));
    }

    @Override
    protected void configure(HttpSecurity http) throws Exception {
        //demander à spring d'utiliser un formulaire d'authentification
        http.formLogin();
        // cette page ne nécessite pas une permission
        http.authorizeRequests().antMatchers( ...antPatterns: "/").permitAll();
        //ces paths sont accessibles juste pour les utilisateurs qui ont role ADMIN
        http.authorizeRequests().antMatchers( ...antPatterns: "/admin/**").hasRole("ADMIN");
        //ces paths sont accessibles juste pour les utilisateurs qui ont role USER
        http.authorizeRequests().antMatchers( ...antPatterns: "/user/**").hasRole("USER");
        //toute requete http nécessite une authentification
        http.authorizeRequests().anyRequest().authenticated();
        http.exceptionHandling().accessDeniedPage("/403");
    }
}
```

Security Controller :

```
@Controller
public class securityController {

















































    @GetMapping("/403")
    public String notAuthorized() { return "403"; }
}
```

## La forme d'ajout d'un patient

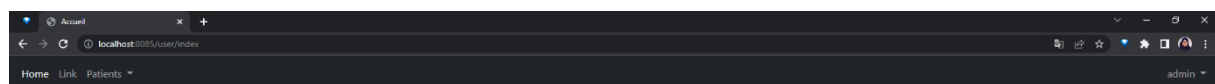
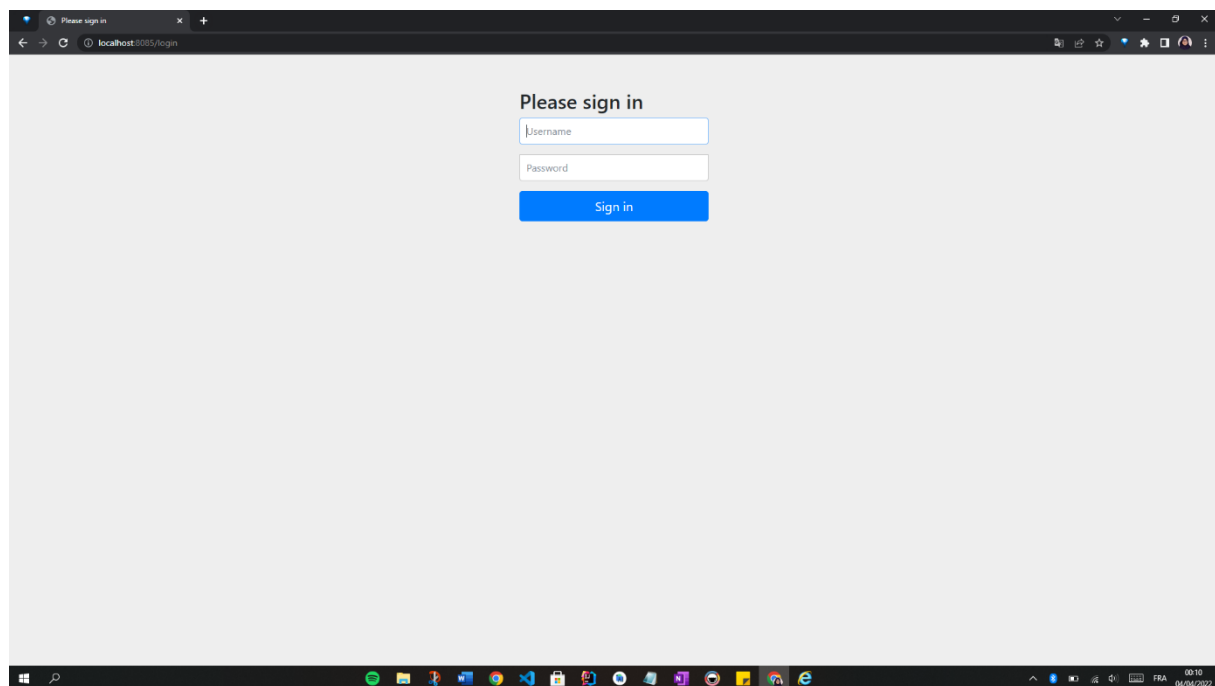
```
<div layout:fragment="content1">
  <div class="col-md-6 offset-3">
    <form method="post" th:action="@{/admin/save}">
      <div class="mb-3">
        <label for="exampleNom" class="form-label">Nom</label>
        <input type="text" class="form-control" id="exampleNom" name="nom" th:value="${patient.nom}">
        <span th:errors="${patient.nom}"></span>
      </div>
      <div class="mb-3">
        <label for="exampleDate" class="form-label">Date de naissance</label>
        <input type="date" class="form-control" id="exampleDate" name="datenaissance" th:value="${patient.datenaissance}">
        <span th:errors="${patient.datenaissance}"></span>
      </div>
      <div class="mb-3 form-check">
        <input type="checkbox" class="form-check-input" id="exampleCheck1" name="malade" th:checked="${patient.malade}">
        <label class="form-check-label" for="exampleCheck1">Malade</label>
        <span th:errors="${patient.malade}"></span>
      </div>
      <div class="mb-3">
        <label for="exampleScore" class="form-label">Score</label>
        <input type="text" class="form-control" id="exampleScore" name="score" th:value="${patient.score}">
        <span th:errors="${patient.score}"></span>
      </div>
      <button type="submit" class="btn btn-primary">Submit</button>
    </form>
  </div>
</div>
```

## La table dans la base de données

Options

					id	datenaissance	malade	nom	score		
<input type="checkbox"/>		Éditer		Copier		Supprimer	1	2022-03-27	0	soukaina	12
<input type="checkbox"/>		Éditer		Copier		Supprimer	2	2022-03-27	0	salma	12
<input type="checkbox"/>		Éditer		Copier		Supprimer	3	2022-03-27	0	soukaina1	12
<input type="checkbox"/>		Éditer		Copier		Supprimer	4	2022-03-27	0	salma1	12
<input type="checkbox"/>		Éditer		Copier		Supprimer	5	2022-03-27	0	soukaina	12
<input type="checkbox"/>		Éditer		Copier		Supprimer	6	2022-03-27	0	salma	12
<input type="checkbox"/>		Éditer		Copier		Supprimer	7	2022-03-27	0	soukaina1	12
<input type="checkbox"/>		Éditer		Copier		Supprimer	8	2022-03-27	0	salma1	12
<input type="checkbox"/>		Éditer		Copier		Supprimer	9	2022-03-27	0	soukaina	12
<input type="checkbox"/>		Éditer		Copier		Supprimer	10	2022-03-27	0	salma	12
<input type="checkbox"/>		Éditer		Copier		Supprimer	11	2022-03-27	0	soukaina1	12
<input type="checkbox"/>		Éditer		Copier		Supprimer	12	2022-03-27	0	salma1	12
<input type="checkbox"/>		Éditer		Copier		Supprimer	13	2022-03-27	0	soukaina	12
<input type="checkbox"/>		Éditer		Copier		Supprimer	14	2022-03-27	0	salma	12
<input type="checkbox"/>		Éditer		Copier		Supprimer	15	2022-03-27	0	soukaina1	12
<input type="checkbox"/>		Éditer		Copier		Supprimer	16	2022-03-27	0	salma1	12

## Affichage



Liste des Patients

Key Word  [Chercher](#)

ID	Nom	DateNaissance	Malade	Score		
1	soukaina	2022-04-04	false	120	<a href="#">Delete</a>	<a href="#">Edit</a>
2	salma	2022-04-04	false	120	<a href="#">Delete</a>	<a href="#">Edit</a>
3	soukaina	2022-04-04	false	120	<a href="#">Delete</a>	<a href="#">Edit</a>
4	salma	2022-04-04	false	120	<a href="#">Delete</a>	<a href="#">Edit</a>
5	soukaina	2022-04-04	false	120	<a href="#">Delete</a>	<a href="#">Edit</a>

[0](#) [1](#)



