Ecole Normale Supérieure de l'Enseignement Technique Mohammedia

Université Hassan II de Casablanca

ENSET

ا لمدرسة العليا لأساتذة التعليم التقني المحمدية

جامعة الحسن الثاني بالدار البيضاء

Département Mathématiques et Informatique

Cycle Ingénieur

«GLSID»

COMPTE-RENDU:
Activité Pratique Spring MVC Thymeleaf

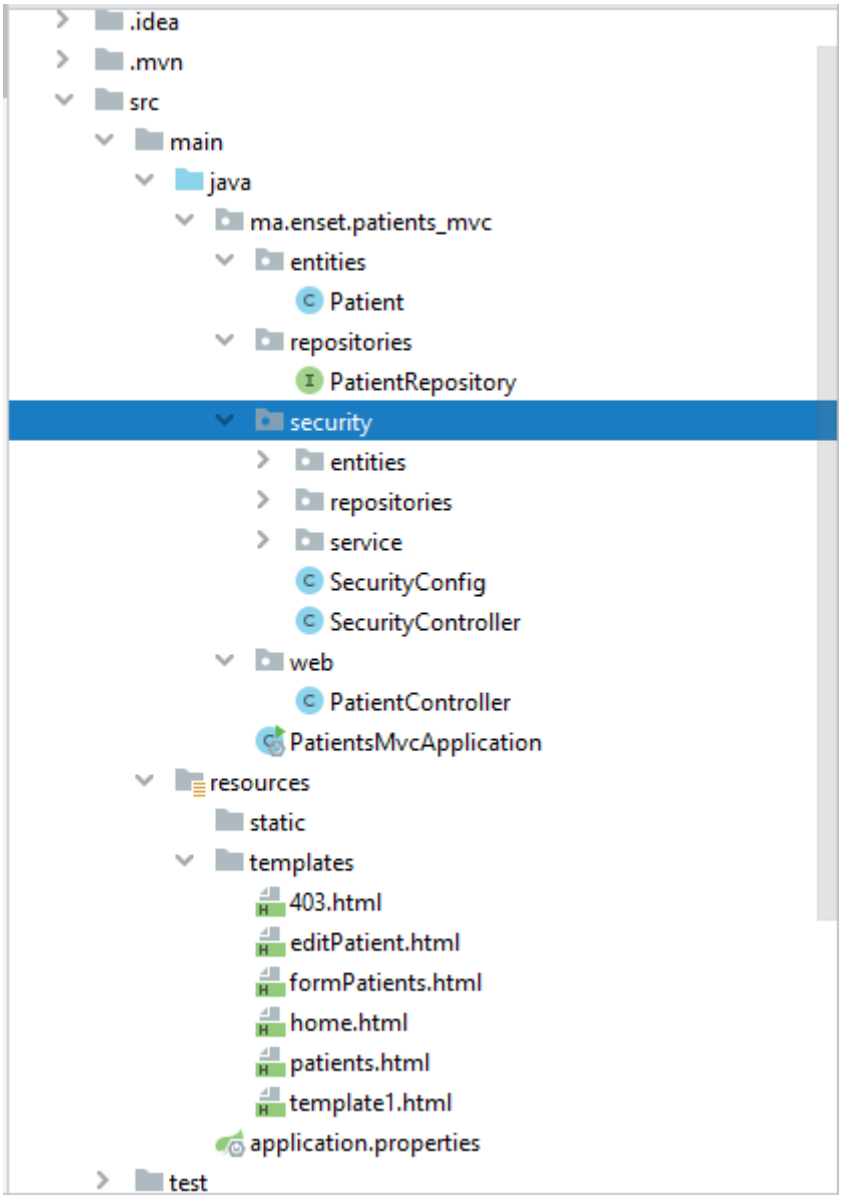
Réalisé par: Tarmoun Amal

Classe : GLSID2

Encadré par:
Mr. Mohammed EL YOUSSEFI

Année universitaire : 2021 / 2022

1- Cas de Patient



Le package Entities :

La classe Patient

```java
package ma.enset.patients_mvc.entities;

import ...
@Entity
@Data  @AllArgsConstructor
public class Patient {
    @Id @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id ;
    @NotEmpty
    @Size(min=4 , max =40)
    private String nom;
    @Temporal(TemporalType.DATE)
    @DateTimeFormat(pattern = "yyyy-MM-dd")
    private Date dateNaissance ;
    private boolean malade ;
    @DecimalMin("100")
    private int score;
    public Patient() {
    }
}

```

Le package Repository

L'interface PatienRepository

```java
package ma.enset.patients_mvc.repositories;

import ma.enset.patients_mvc.entities.Patient;
import org.springframework.data.domain.Page;
import org.springframework.data.domain.Pageable;
import org.springframework.data.jpa.repository.JpaRepository;

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

```

Le package web

La classe PatientController

```java
@Controller
public class PatientControllen {
    private PatientRepository patientRepository;

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

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

        return "home";
    }
    @GetMapping("/patients")
    @ResponseBody
    public List<Patient> ListPatients() { return patientRepository.findAll(); }
    @GetMapping("/admin/formPatients")
    public String formPatients(Model model){
        Patient patient = new Patient();
        if (patient != null) {
            model.addAttribute( attributeName: "p", patient);
        } else {
            model.addAttribute( attributeName: "p", new Patient());
        }
        return "formPatients";
    }
```

```java
    @PostMapping(path = "/admin/save")
    public String save( @Valid  @ModelAttribute Patient patient, BindingResult bindingResult ,Model model ,
                        @RequestParam(defaultValue = "0") int page ,
                        @RequestParam(defaultValue = "") String keyword){
        model.addAttribute( attributeName: "p", patient);
        if(bindingResult.hasErrors()) {
            return "formPatients";
        }

        patientRepository.save(patient);
        return "redirect:/user/index?page="+page+"&keyword="+keyword;
    }

    @GetMapping("/admin/editPatient")
    public String editPatient(Model model , Long id , String keyword , int page ){
        Patient patient =patientRepository.findById(id).orElse( other: null);
        if(patient==null) throw new RuntimeException("patient introuvable");
        model.addAttribute( attributeName: "p" ,patient);
        model.addAttribute( attributeName: "page", page);
        model.addAttribute( attributeName: "keyword", keyword);
        return "editPatient";
    }
}
```

La classe PatientsMvcApplication :

```java
package ma.enset.patients_mvc;

import ...

@SpringBootApplication
public class PatientsMvcApplication {

    public static void main(String[] args) { SpringApplication.run(PatientsMvcApplication.class, args); }
    @Bean
    CommandLineRunner commandLineRunner(PatientRepository patientRepository){
        return args -> {
            patientRepository.save(new Patient( id: null, nom: "amal",new Date(), malade: false, score: 1000));
            patientRepository.save(new Patient( id: null, nom: "ikram",new Date(), malade: true, score: 590));
            patientRepository.save(new Patient( id: null, nom: "kawtar",new Date(), malade: false, score: 199));
            patientRepository.save(new Patient( id: null, nom: "soumia",new Date(), malade: true, score: 339));


            patientRepository.findAll().forEach(p->{
                System.out.println(p.getNom());
            });


        };
    }

    @Bean
    PasswordEncoder passwordEncoder() { return new BCryptPasswordEncoder(); }
    @Bean
    CommandLineRunner saveUsers(SecurityService securityService){
        return args -> {
            securityService.saveNewUser( username: "fatiha" , password: "1234", repassword: "1234");
            securityService.saveNewUser( username: "douha" , password: "1234", repassword: "1234");
            securityService.saveNewUser( username: "hiba" , password: "1234", repassword: "1234");



//            securityService.saveNewRole("USER","");
//            securityService.saveNewRole("ADMIN","");
//
            securityService.addRoleToUser( username: "fatiha", roleName: "USER");
            securityService.addRoleToUser( username: "fatiha", roleName: "ADMIN");
            securityService.addRoleToUser( username: "douha", roleName: "USER");
            securityService.addRoleToUser( username: "hiba", roleName: "USER");

        };
    }

}
```

Security :

Entities:

AppRole:

```java
@Entity
@Data @AllArgsConstructor @NoArgsConstructor
public class AppRole {
    @Id @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long roleId ;
    @Column(unique = true)
    private String role;
    private String description;
}
```

AppUser:

```java
@Entity
@Data @AllArgsConstructor @NoArgsConstructor
public class AppUser {
    @Id
    private String userId;
    @Column(unique = true)
    private String username ;
    private String password ;
    private Boolean active ;

    @ManyToMany(fetch = FetchType.EAGER)
    private List<AppRole> appRoleList = new ArrayList<>();
}
```

Repositories :

```java
public interface AppRoleRepository extends JpaRepository<AppRole, Long> {
    AppRole findByRole(String rolname );
}
```

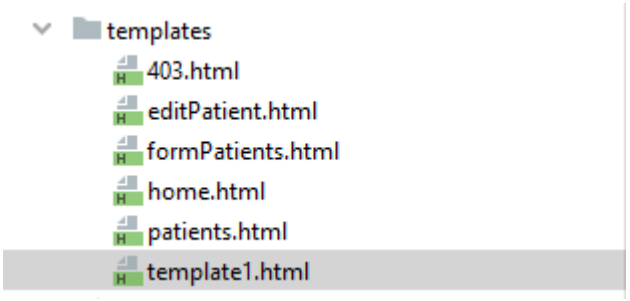
```java
public interface AppUserRepository extends JpaRepository<AppUser,String> {
    AppUser findByUsername(String username);
}
```

Services :

```java
@Configuration // instancier  en premier lieu
@EnableWebSecurity
public class SecurityConfig extends WebSecurityConfigurerAdapter {

    @Autowired
    private DataSource dataSource ;
    @Autowired
    private UserDetailsServiceIml userDetailsServiceIml;
    @Autowired
    private PasswordEncoder passwordEncoder;


    @Override // strategy pour que spring security va chercher les users
    protected void configure(AuthenticationManagerBuilder auth) throws Exception {


//        String ecodedPassword=passwordEncoder.encode("1234");
//        System.out.println(ecodedPassword);
       // preciser comment spring va chercherr les user et les roles ( bdd ou user en memoire ou
       // annuaire en  entreprise)
//        auth.inMemoryAuthentication().withUser("user1").password(ecodedPassword).roles("USER");
//        auth.inMemoryAuthentication().withUser("user2").password(passwordEncoder.encode("1111")).roles("USER");
//        auth.inMemoryAuthentication().withUser("admin").password(passwordEncoder.encode("2345"))
//        .roles("USER","ADMIN"); // user qui on le droit d'acceder a l'application seront stockerr en memoire
//    auth.jdbcAuthentication()
//            .dataSource(dataSource)
```

```
//          .dataSource(dataSource)
//          .usersByUsernameQuery("select username as principal, password as credentials,active from users where u
//          .authoritiesByUsernameQuery("Select username as principal, role as role from users_roles where usernam
//          .rolePrefix("ROLE_")
//          .passwordEncoder(passwordEncoder);

    auth.userDetailsService(userDetailsServiceIml);

}
@Override
protected void configure(HttpSecurity http) throws Exception {
    http.formLogin();
    http.authorizeRequests().antMatchers( ...antPatterns: "/").permitAll();
    http.authorizeRequests().antMatchers( ...antPatterns: "/admin/**").hasAuthority("ADMIN");
    http.authorizeRequests().antMatchers( ...antPatterns: "/user/**").hasAuthority("USER");
    http.authorizeRequests().antMatchers( ...antPatterns: "/webjars/**").permitAll();
    http.authorizeRequests().anyRequest().authenticated(); // toutes les rrequetes http neccecite une  authenti
    http.exceptionHandling().accessDeniedPage("/403");
}
```

Thymeleaf:



Les Interfaces :

Login :

Nouveau patient

Nom

Date Naissance

mm/dd/yyyy

Malade

Score

0

Save

Chercher un Patient :

List des patients

key word chercher

ID	Nom	Date	Malade	Score		
1	amal	2022-03-28	false	12	Delete	Edit
2	ikram	2022-03-28	true	565	Delete	Edit
3	kawtar	2022-03-28	false	199	Delete	Edit
6	ikram	2022-03-28	true	5655	Delete	Edit
7	kawtar	2022-03-28	false	199	Delete	Edit

0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23
24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45
46 47 48 49 50 51 52 53 54 55