

Trainning: Springboot & Angular 9 Trainer: Dr. Mohamed Amine MEZGHICH Period of training : 30 Hours Email : ma.mezghich@smart-it-partner.com Phone : +216 51 36 36 34	Workshop n° 3: Spring boot _ Security Layer Goals: Be able to develop your own project using spring boot framework with controllers, Actions, views and models. Manage data between views and integrate css style. Finally propose some unit tests and deploy the application on a remote server August Session 2020
--	---

What you will discover in this workshop?

1. The Spring Boot & Spring Security Layer

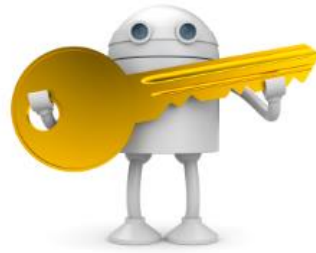
- Add dependencies to pom.xml file*
- Create User and Role entities*
- Create repositories for User and Roles entities*
- Add Configuration spring security files(2 files)*
- Add loginController.java*
- Add login/register views*
- Add roles to users from mysql database*

2. Template inheritance and protected access to some page using role.

- Create home page after have been authenticated*
- See fragment thymleaf concept*
- Show authenticated usernames, roles, logout*
- Restrict access to certain pages for ADMIN role*
- Running the project*

Some pictures to what we will perform?

[Go To Registration Page](#)



Welcome

Smart **It** Partner
We Build Smart Solutions For You

Smart IT PARTNER

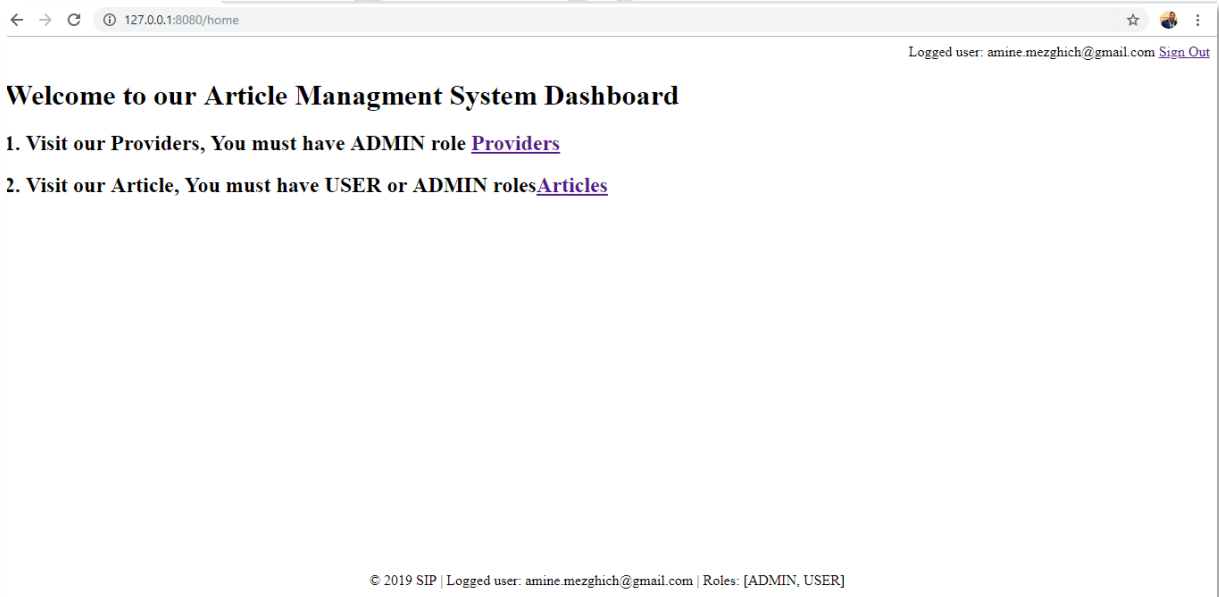
We build Smart Solutions for You

<https://smart-it-partner.com/>

[Go To Login Page](#)

Registration Form

Register User



Welcome to our Article Managment System Dashboard

1. Visit our Providers, You must have ADMIN role [Providers](#)
2. Visit our Article, You must have USER or ADMIN roles [Articles](#)

Logged user: amine.mezghich@gmail.com [Sign Out](#)

© 2019 SIP | Logged user: amine.mezghich@gmail.com | Roles: [ADMIN, USER]

Welcome to our Article Managment System Dashboard

1. Visit our Providers, You must have ADMIN role [Providers](#)
2. Visit our Article, You must have USER or ADMIN roles [Articles](#)

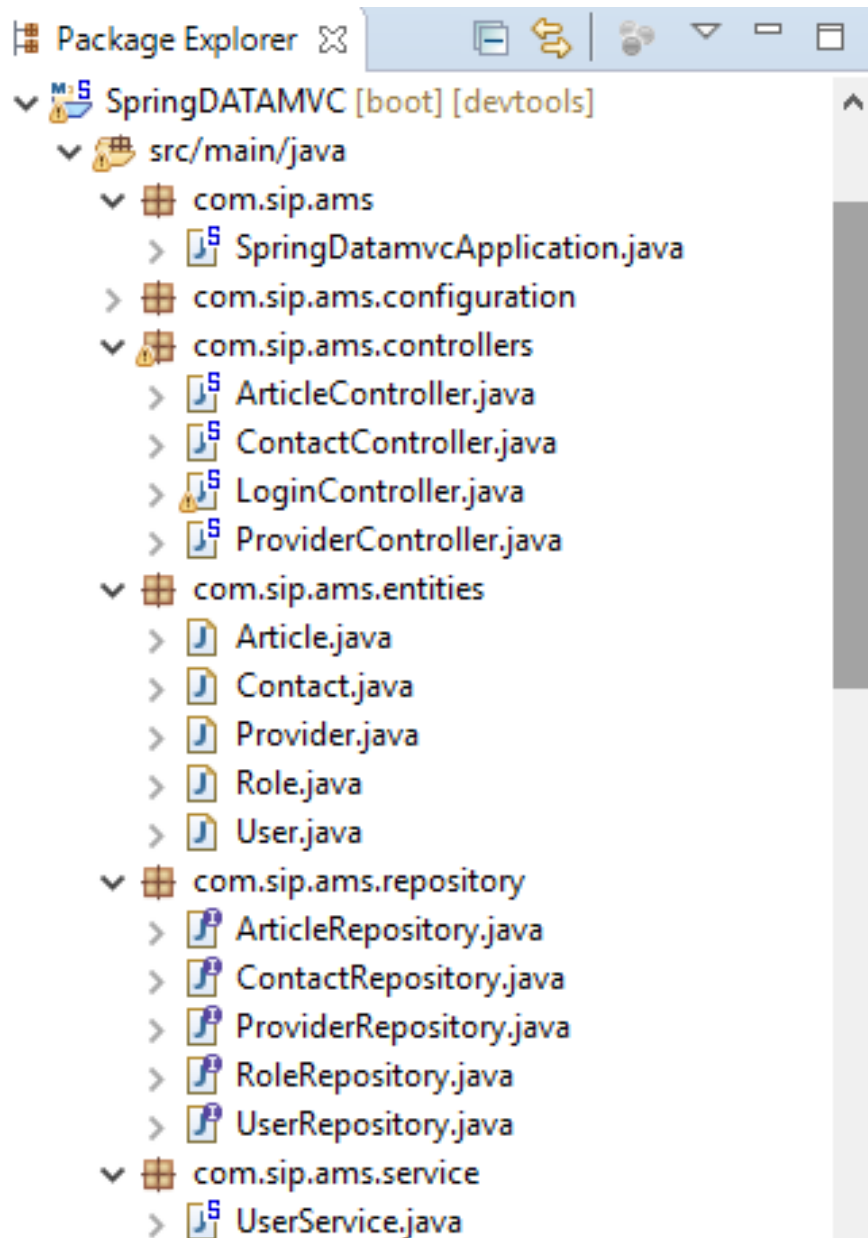
© 2019 SIP | Logged user: raouf@gmail.com | Roles: [USER]

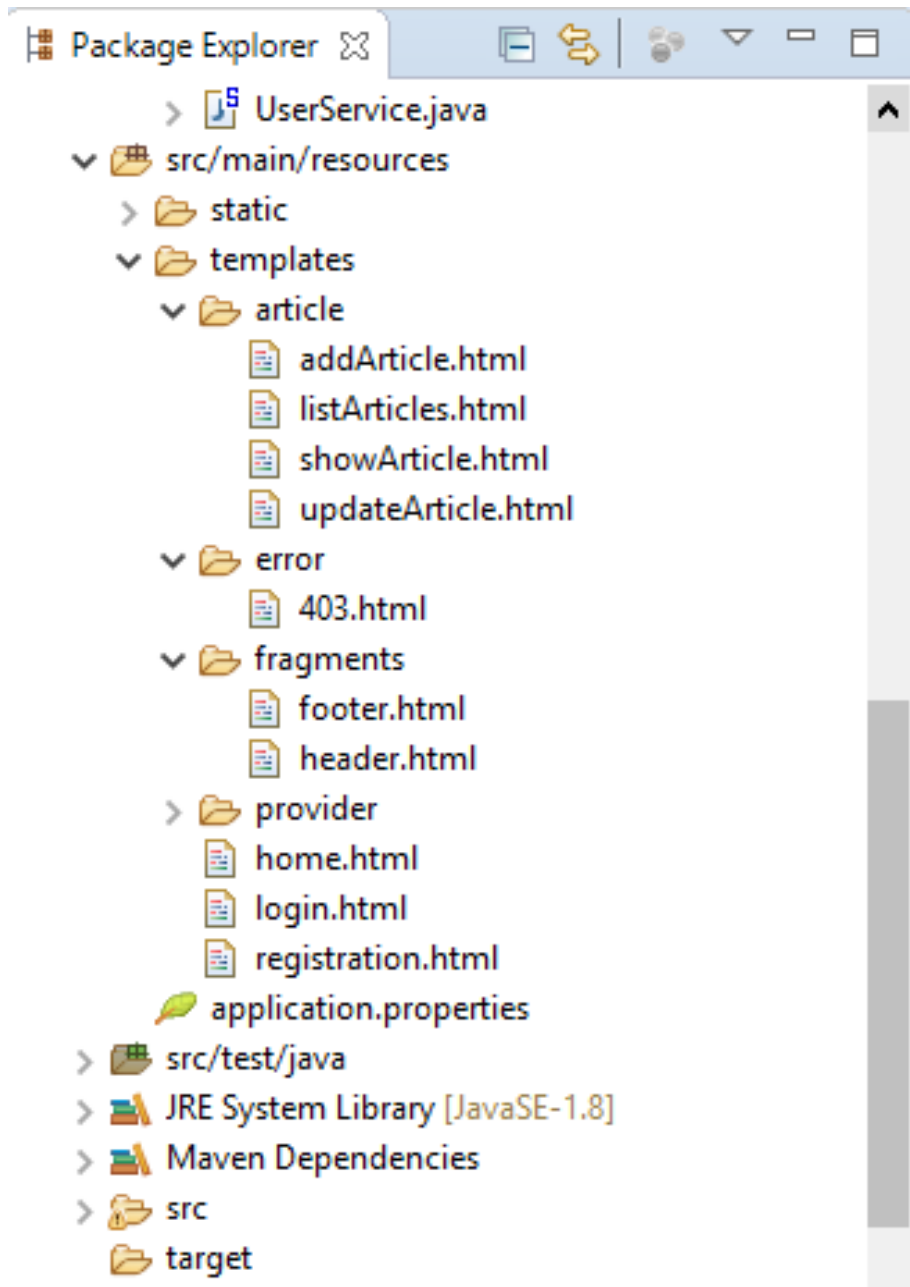
← → ↻ ⓘ 127.0.0.1:8080/provider/list

You don't have permission to visit the page
return to home page [here](#)

Smart It Partner
We Build Smart Solutions For You

Project Structure





Pom.xml file

Add the following dependency to pom.xml file

```
<dependency>
    <groupId>org.springframework.security</groupId>
    <artifactId>spring-security-web</artifactId>
</dependency>

<dependency>
    <groupId>org.thymeleaf.extras</groupId>
    <artifactId>thymeleaf-extras-
springsecurity5</artifactId>
</dependency>

    <dependency>
    <groupId>org.springframework.security</groupId>
    <artifactId>spring-security-config</artifactId>
</dependency>
```

Model Creation

Now let's create our model classes called User and Role(Entity classes). Lombok is a very useful library used to generate boilerplate code mainly for model/data objects.

User

This class includes validations based on the validations provided by Hibernate.

```

package com.sip.ams.entities;
import lombok.Data;
import org.hibernate.validator.constraints.Length;
import javax.persistence.*;
import javax.validation.constraints.Email;
import javax.validation.constraints.NotEmpty;
import java.util.Set;
@Data
@Entity
@Table(name = "user")
public class User {
    @Id
    @GeneratedValue(strategy = GenerationType.AUTO)
    @Column(name = "user_id")
    private int id;

    @Column(name = "email")
    @Email(message = "*Please provide a valid Email")
    @NotEmpty(message = "*Please provide an email")
    private String email;

    @Column(name = "password")
    @Length(min = 5, message = "*Your password must have at least
5 characters")
    @NotEmpty(message = "*Please provide your password")
    private String password;

    @Column(name = "name")
    @NotEmpty(message = "*Please provide your name")
    private String name;

    @Column(name = "last_name")
    @NotEmpty(message = "*Please provide your last name")
    private String lastName;

    @Column(name = "active")
    private int active;

    @ManyToMany(cascade = CascadeType.ALL)
    @JoinTable(name = "user_role", joinColumns = @JoinColumn(name
= "user_id"), inverseJoinColumns = @JoinColumn(name = "role_id"))
    private Set<Role> roles;

```



```
public int getId() {
    return id;
}
public void setId(int id) {
    this.id = id;
}
public String getEmail() {
    return email;
}
public void setEmail(String email) {
    this.email = email;
}
public String getPassword() {
    return password;
}
public void setPassword(String password) {
    this.password = password;
}
public String getName() {
    return name;
}
public void setName(String name) {
    this.name = name;
}
public String getLastName() {
    return lastName;
}
public void setLastName(String lastName) {
    this.lastName = lastName;
}
public int getActive() {
    return active;
}
public void setActive(int active) {
    this.active = active;
}
public Set<Role> getRoles() {
    return roles;
}
public void setRoles(Set<Role> roles) {
    this.roles = roles;
}
}
```

Role

Smart IT PARTNER

We build Smart Solutions for You

<https://smart-it-partner.com/>

```

package com.sip.ams.entities;

import lombok.Data;
import javax.persistence.*;
@Data
@Entity
@Table(name = "role")
public class Role {
    @Id
    @GeneratedValue(strategy = GenerationType.AUTO)
    @Column(name = "role_id")
    private int id;
    @Column(name = "role")
    private String role;
    public int getId() {
        return id;
    }
    public void setId(int id) {
        this.id = id;
    }
    public String getRole() {
        return role;
    }
    public void setRole(String role) {
        this.role = role;
    }
}

```

Smart IT Partner

Data Layer (JPA Repositories)

The repositories allow us to access the information stored in the data base.

UserRepository

```

package com.orsys.ams.repositories;

import org.springframework.data.jpa.repository.JpaRepository;
import org.springframework.stereotype.Repository;
import com.sip.ams.entities.User;
@Repository("userRepository")
public interface UserRepository extends JpaRepository<User,
Integer> {
    User findByEmail(String email);
}

```

RoleRepository

```
package com.orsys.ams.repositories;

import org.springframework.data.jpa.repository.JpaRepository;
import org.springframework.stereotype.Repository;
import com.orsys.ams.entities.Role;
@Repository("roleRepository")
public interface RoleRepository extends JpaRepository<Role, Integer> {
    Role findByRole(String role);
}
```

Exercise (15 min)

- 1-Develop a bootstrap form to add "ADMIN" and "USER" roles to data base.
- 2-Use a RoleContoller with crud(list, add only) operations

Correction

```
package com.orsys.ams.controllers;
import java.util.List;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestParam;

import com.orsys.ams.entities.Role;
import com.orsys.ams.repositories.RoleRepository;

@Controller
@RequestMapping("/role/")
public class RoleController {

    private final RoleRepository roleRepository;
```

```

@Autowired
public RoleController(RoleRepository roleRepository) {
    this.roleRepository = roleRepository;
}

@GetMapping("list")
public String listRoles(Model model) {

    List<Role> roles = (List<Role>) roleRepository.findAll();
    long nbr = roleRepository.count();
    if(roles.size()==0)
        roles = null;
    model.addAttribute("roles", roles);
    model.addAttribute("nbr", nbr);
    return "role/listRoles";
}

@GetMapping("add")
public String showAddRoleForm() {

    //m.addAttribute("Role",new Role("Admin"));
    return "role/addRole";
}

@PostMapping("add")
public String addRole(@RequestParam("role") String role) {

    System.out.println(role);
    Role r = new Role(role);
    Role rSaved = roleRepository.save(r);
    System.out.println("role = "+ rSaved);
    return "redirect:list";
}
}

```

listRoles.html

```
<!DOCTYPE html>
<html xmlns:th="http://www.thymeleaf.org">

<head>
<meta charset="utf-8">
<meta http-equiv="x-ua-compatible" content="ie=edge">
<title>Providers</title>
<meta name="viewport" content="width=device-width, initial-
scale=1">
<link rel="stylesheet"
      href="https://stackpath.bootstrapcdn.com/bootstrap/4.1.3/css/bo
otstrap.min.css"
      integrity="sha384-
MCw98/SFnGE8fJT3GXwEOngsV7Zt27NXFoaoApmYm81iuXoPkFOJwJ8ERdknLPMO"
      crossorigin="anonymous">
<link rel="stylesheet"
      href="https://use.fontawesome.com/releases/v5.4.1/css/all.css"
      integrity="sha384-
5sAR7xN1Nv6T6+dt2mhtzEpVJvfS3NScPQTrOxhwjIuvcA67KV2R5Jz6kr4abQsz"
      crossorigin="anonymous">
<!-- <link rel="stylesheet" href=" ../css/shards.min.css"> -->
</head>

<body>
  <div class="container my-2">
    <div class="card">
      <div class="card-body">
        <div th:switch="${providers}" class="container
my-5">
          <p class="my-5">
            <a th:href="@{'/role/add/'}" class="btn
btn-primary">
              <i class="fas fa-user-plus ml-2"> Add
Role</i></a>
          </p>
          <div class="col-md-10">
            <h2 th:case="null">No Role yet!</h2>
            <div th:case="*">
              <h2>Nombre total de role= <span
th:text="${nbr}">10</span></h2>
              <table class="table table-striped
table-responsive-md">
```

```

<thead>
  <tr>
    <th>ID</th>
    <th>Role</th>
  </tr>
</thead>
<tbody>
  <tr th:each="role :
${roles}">
    <td
th:text="${role.id}"></td>
    <td
th:text="${role.role}"></td>
  </tr>
</tbody>
</table>
</div>
</div>
</div>
</div>
</div>
</div>
</body>
</html>

```

addRole.html

```

<!DOCTYPE html>
<html xmlns:th="http://www.thymeleaf.org">

<head>
<meta charset="utf-8">
<meta http-equiv="x-ua-compatible" content="ie=edge">
<title>Add Role</title>
<meta name="viewport" content="width=device-width, initial-
scale=1">
<link rel="stylesheet"
      href="https://stackpath.bootstrapcdn.com/bootstrap/4.1.3/css/bo
otstrap.min.css"
      integrity="sha384-
MCw98/SFnGE8fJT3GXwEOngsV7Zt27NXFoaoApmYm81iuXoPkFOJwJ8ERdknLPMO"

```

Smart IT PARTNER

We build Smart Solutions for You

<https://smart-it-partner.com/>

```

        crossorigin="anonymous">
<link rel="stylesheet"
      href="https://use.fontawesome.com/releases/v5.4.1/css/all.css"
      integrity="sha384-
5sAR7xN1Nv6T6+dt2mhtzEpVJvfS3NScPQTr0xhwjIuvCA67KV2R5Jz6kr4abQsz"
      crossorigin="anonymous">
<!-- <link rel="stylesheet" href="../../css/shards.min.css"> -->
</head>

<body>
  <div class="container my-5">
    <h3>Add Role</h3>
    <div class="card">
      <div class="card-body">
        <div class="col-md-10">
          <!-- form action="#"
th:action="@{/role/add}"
          th:object="${Role}" method="post">
            <div class="row">
              <div class="form-group col-md-8">
                <label for="role" class="col-
form-label">Role</label>

                <input
th:field="*{role}" class="form-control" id="role"
                placeholder="Role">
                <span
th:if="${#fields.hasErrors('role')}" th:errors="*{role}"
                class="text-
danger"></span>

                </div>
                <div class="col-md-6">
                  <input type="submit" class="btn
btn-primary"
                  value="Add Role">
                </div>
              <div class="form-group col-md-
8"></div>

            </div>
          </form-->

```

```

        <form action="#" th:action="@{/role/add}"
method="post">
            <div class="row">
                <div class="form-group col-md-8">
                    <label for="role" class="col-
form-label">Role</label>

                    <input
                        type="text" name="role"
class="form-control" id="role"
                        placeholder="Role">

                    </div>
                    <div class="col-md-6">
                        <input type="submit" class="btn
btn-primary"
                        value="Add Role">
                    </div>
                <div class="form-group col-md-
8"></div>
            </div>
        </form>
    </div>
</div>
</div>
</div>
</body>
</html>

```

Service Layer

Now let's create our service layer. We will inject the UserRepository, RoleRepository and the BCryptPasswordEncoder into UserService .

UserService

```
package com.sip.ams.service;

import com.sip.ams.entities.Role;
import com.sip.ams.entities.User;
import com.sip.ams.repository.RoleRepository;
import com.sip.ams.repository.UserRepository;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.security.crypto.bcrypt.BCryptPasswordEncoder;
import org.springframework.stereotype.Service;
import java.util.Arrays;
import java.util.HashSet;
@Service("userService")
public class UserService {
    private UserRepository userRepository;
    private RoleRepository roleRepository;
    private BCryptPasswordEncoder bCryptPasswordEncoder;
    @Autowired
    public UserService(UserRepository userRepository,
                      RoleRepository roleRepository,
                      BCryptPasswordEncoder
bCryptPasswordEncoder) {
        this.userRepository = userRepository;
        this.roleRepository = roleRepository;
        this.bCryptPasswordEncoder = bCryptPasswordEncoder;
    }

    public User findUserByEmail(String email) {
        return userRepository.findByEmail(email);
    }

    public void saveUser(User user) {
user.setPassword(bCryptPasswordEncoder.encode(user.getPassword()));
;
        user.setActive(0);
        Role userRole = roleRepository.findByRole("USER");
        user.setRoles(new HashSet<Role>(Arrays.asList(userRole)));
        userRepository.save(user);
    }
}
```

Configuration Files

WebMvcConfig

This class defines the password encoder that we just injected in the service layer.

```
package com.sip.ams.configuration;

import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.Configuration;
import org.springframework.security.crypto.bcrypt.BCryptPasswordEncoder;
import org.springframework.web.servlet.config.annotation.WebMvcConfigurer;

@Configuration
public class WebMvcConfig implements WebMvcConfigurer {
    @Bean
    public BCryptPasswordEncoder passwordEncoder() {
        BCryptPasswordEncoder bCryptPasswordEncoder = new
        BCryptPasswordEncoder();
        return bCryptPasswordEncoder;
    }
}
```

Smart IT Partner
We Build Smart Solutions For You

SecurityConfiguration

```
package com.sip.ams.configuration;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.beans.factory.annotation.Value;
import org.springframework.context.annotation.Configuration;
import
org.springframework.security.config.annotation.authentication.builders.AuthenticationManagerBuilder;
import
org.springframework.security.config.annotation.web.builders.HttpSecurity;
import
org.springframework.security.config.annotation.web.builders.WebSecurity;
import
org.springframework.security.config.annotation.web.configuration.EnableWebSecurity;
import
org.springframework.security.config.annotation.web.configuration.WebSecurityConfigurerAdapter;
import
org.springframework.security.crypto.bcrypt.BCryptPasswordEncoder;
import
org.springframework.security.web.util.matcher.AntPathRequestMatcher;
import javax.sql.DataSource;
@Configuration
@EnableWebSecurity
public class SecurityConfiguration extends
WebSecurityConfigurerAdapter {
    @Autowired
    private BCryptPasswordEncoder bCryptPasswordEncoder;
    @Autowired
    private DataSource dataSource;
    @Value("${spring.queries.users-query}")
    private String usersQuery;
    @Value("${spring.queries.roles-query}")
    private String rolesQuery;

    @Override
    protected void configure(AuthenticationManagerBuilder auth)
        throws Exception {
        auth.
            jdbcAuthentication()
            .usersByUsernameQuery(usersQuery)
            .authoritiesByUsernameQuery(rolesQuery)
            .dataSource(dataSource)
            .passwordEncoder(bCryptPasswordEncoder);
    }
}
```

```

@Override
    protected void configure(HttpSecurity http) throws Exception {

        http.
            authorizeRequests()
                .antMatchers("/").permitAll() // accès pour tous
            users
                .antMatchers("/login").permitAll() // accès pour
            tous users
                .antMatchers("/registration").permitAll() // accès
            pour tous users
                .antMatchers("/provider/**").hasAuthority("ADMIN")

            .antMatchers("/article/**").hasAuthority("USER").anyRequest()

            .authenticated().and().csrf().disable().formLogin() // l'accès de
            fait via un formulaire

            .loginPage("/login").failureUrl("/login?error=true") // fixer la
            page login

                .defaultSuccessUrl("/home") // page d'accueil
            après login avec succès
                .usernameParameter("email") // paramètres
            d'authentifications login et password
                .passwordParameter("password")
                .and().logout()
                .logoutRequestMatcher(new
            AntPathRequestMatcher("/logout")) // route de deconnexion ici
            /logout

            .logoutSuccessUrl("/login").and().exceptionHandling() // une fois
            deconnecté redirection vers login

                .accessDeniedPage("/403");

        }

        // laisser l'accès aux ressources
        @Override
        public void configure(WebSecurity web) throws Exception {
            web
                .ignoring()
                .antMatchers("/resources/**", "/static/**",
            "/css/**", "/js/**", "/images/**");
        }
    }

```

This class is where the security logic is implemented, let's analyze the code.

application.properties file

Basically the idea of this file is to setup the configurations in a property file instead of a xml file or a java configuration class.

```
spring.jpa.hibernate.ddl-auto=update
spring.datasource.url=jdbc:mysql://localhost:3306/amsdb2?useUnicode=true&useJDBCCompliantTimezoneShift=true&useLegacyDatetimeCode=false&serverTimezone=UTC
spring.datasource.username=root
spring.datasource.password=

# =====
# = Spring Security / Queries for AuthenticationManagerBuilder
# =====
spring.queries.users-query=select email, password, active from
user where email=?
spring.queries.roles-query=select u.email, r.role from user u
inner join user_role ur on(u.user_id=ur.user_id) inner join role r
on(ur.role_id=r.role_id) where u.email=?
```



Notes:

- The properties “spring.queries.users-query” and “spring.queries.roles-query” define where the user/role information is stored.
- Update with your Database credentials.

Controller Layer

MVC Logic

```

package com.sip.ams.controllers;

import javax.validation.Valid;
import com.sip.ams.entities.User;
import com.sip.ams.service.UserService;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.security.core.Authentication;
import org.springframework.security.core.context.SecurityContextHolder;
import org.springframework.stereotype.Controller;
import org.springframework.validation.BindingResult;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;
import org.springframework.web.servlet.ModelAndView;
@Controller
public class LoginController {
    @Autowired
    private UserService userService;
    @RequestMapping(value={"/", "/login"}, method =
RequestMethod.GET)
    public ModelAndView login(){
        ModelAndView modelAndView = new ModelAndView();
        modelAndView.setViewName("login");
        return modelAndView;
    }

    @RequestMapping(value={"/home"}, method = RequestMethod.GET)
    public ModelAndView accueil(){
        ModelAndView modelAndView = new ModelAndView();
        modelAndView.setViewName("home");
        return modelAndView;
    }

    @RequestMapping(value={"/registration"}, method =
RequestMethod.GET)
    public ModelAndView registration(){
        ModelAndView modelAndView = new ModelAndView();
        User user = new User();
        modelAndView.addObject("user", user);
        modelAndView.setViewName("registration");
        return modelAndView;
    }
}

```

```

@RequestMapping(value = "/registration", method =
RequestMethod.POST)
    public ModelAndView createNewUser(@Valid User user,
BindingResult bindingResult) {
    ModelAndView modelAndView = new ModelAndView();
    User userExists =
userService.findUserByEmail(user.getEmail());
    if (userExists != null) {
        bindingResult
            .rejectValue("email", "error.user",
                "There is already a user registered
with the email provided");
    }
    if (bindingResult.hasErrors()) {
        modelAndView.setViewName("registration");
    } else {
        userService.saveUser(user);
        modelAndView.addObject("successMessage", "User has
been registered successfully");
        modelAndView.addObject("user", new User());
        modelAndView.setViewName("registration");
    }
    return modelAndView;
}
/* @RequestMapping(value="/admin/home", method =
RequestMethod.GET)
    public ModelAndView home(){
        ModelAndView modelAndView = new ModelAndView();
        Authentication auth =
SecurityContextHolder.getContext().getAuthentication();
        User user = userService.findUserByEmail(auth.getName());
        modelAndView.addObject("userName", "Welcome " +
user.getName() + " " + user.getLastName() + " (" + user.getEmail()
+ ")");
        modelAndView.addObject("adminMessage","Content Available
Only for Users with Admin Role");
        modelAndView.setViewName("admin/home");
        return modelAndView;
    }*/

@GetMapping("/403")
    public String error403() {
        return "/error/403";
    }
}

```

By default Spring Boot defines the view resolver in the next way.

- **Prefix** → resources/templates
- **Suffix** → html

Note: if you want to implement a custom view resolver you can do it using the application.properties file or the a java configuration file.

Views Layer



Home.html

```
<!DOCTYPE HTML>
<html xmlns:th="http://www.thymeleaf.org">
<head>
  <title>AMS-Dashboard</title>
```



```
</head>
<body>

<div th:replace="fragments/header :: header"/>

<div class="container">

    <div class="starter-template">
        <h1>Welcome to our Article Managment System Dashboard</h1>
        <h2>1. Visit our Providers, You must have ADMIN role <a
th:href="@{/provider/list}">Providers</a></h2>
        <h2>2. Visit our Article, You must have USER or ADMIN roles<a
th:href="@{/article/list}">Articles</a></h2>
        <!-- h2>3. Visit <a th:href="@{/about}">Normal page</a></h2-->
    </div>

</div>
<!-- /.container -->

<div th:replace="fragments/footer::footer"/>

</body>
</html>
```

login.html

```
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml"
      xmlns:th="http://www.thymeleaf.org">

<head>
    <title>Spring Security Tutorial</title>
    <link rel="stylesheet" type="text/css"
th:href="@{/css/Login.css}" />
    <link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstra
p.min.css">
    <script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.1.1/jquery.min
.js"></script>
    <script
src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/js/bootstrap.
min.js"></script>
</head>
<body>
    <form th:action="@{/registration}" method="get">
        <button class="btn btn-md btn-warning btn-block"
type="Submit">Go To Registration Page</button>
    </form>

    <div class="container">
        
        <form th:action="@{/login}" method="POST" class="form-
signin">
            <h3 class="form-signin-heading"
th:text="Welcome"></h3>
            <br/>

            <input type="text" id="email" name="email"
th:placeholder="Email"
                class="form-control" /> <br/>
            <input type="password" th:placeholder="Password"
id="password" name="password" class="form-
control" /> <br />

            <div align="center" th:if="${param.error}">
                <p style="font-size: 20; color:
#FF1C19;">Email or Password invalid, please verify</p>
            </div>
            <button class="btn btn-lg btn-primary btn-block"
name="Submit" value="Login" type="Submit"
th:text="Login"></button>
        </form>
    </div>
</body>
</html>
```

registration.html

```
<!DOCTYPE html>
<html lang="en" xmlns="http://www.w3.org/1999/xhtml"
      xmlns:th="http://www.thymeleaf.org">
<head>
    <title>Registration Form</title>
    <link rel="stylesheet" type="text/css"
th:href="@{/css/registration.css}" />
    <link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstra
p.min.css">
    <script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.1.1/jquery.min
.js"></script>
    <script
src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/js/bootstrap.
min.js"></script>
</head>
<body>
    <form th:action="@{/}" method="get">
        <button class="btn btn-md btn-warning btn-block"
type="Submit">Go To Login Page</button>
    </form>
```



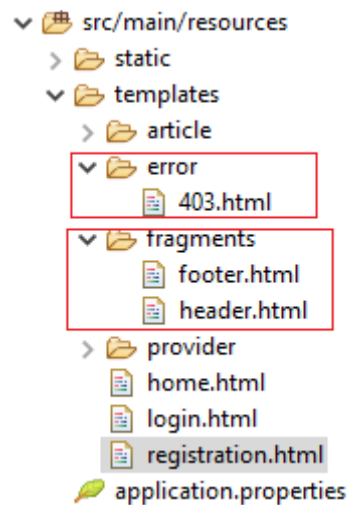
Smart It Partner
We Build Smart Solutions For You

```

<div class="container">
  <div class="row">
    <div class="col-md-6 col-md-offset-3">
      <form autocomplete="off" action="#"
th:action="@{/registration}"
      th:object="${user}" method="post"
class="form-horizontal"
      role="form">
        <h2>Registration Form</h2>
        <div class="form-group">
          <div class="col-sm-9">
            <label
th:if="${#fields.hasErrors('name')}}" th:errors="*{name}"
            class="validation-
message"></label>
            <input type="text"
th:field="*{name}" placeholder="Name"
            class="form-control" />
          </div>
        </div>
        <div class="form-group">
          <div class="col-sm-9">
            <label
th:if="${#fields.hasErrors('lastName')}}"
th:errors="*{lastName}"
            class="validation-
message"></label>
            <input type="text"
th:field="*{lastName}"
            placeholder="Last Name"
class="form-control" />
          </div>
        </div>
        <div class="form-group">
          <div class="col-sm-9">
            <input type="text"
th:field="*{email}" placeholder="Email"
            class="form-control" />
          </div>
          <label
th:if="${#fields.hasErrors('email')}}" th:errors="*{email}"
            class="validation-
message"></label>
          </div>
        </div>
        <div class="form-group">
          <div class="col-sm-9">
            <input type="password"
th:field="*{password}"
            placeholder="Password"
class="form-control" />
          </div>
          <label
th:if="${#fields.hasErrors('password')}}"
th:errors="*{password}"
            class="validation-

```

Note: By default Spring Boot will create the database structure if you have provided in the right way your MySQL credentials in the application.properties file.



For the fragments files and the error 403.file

Header.html

```
<html xmlns="http://www.w3.org/1999/xhtml"
      xmlns:th="http://www.thymeleaf.org"
      xmlns:sec="http://www.thymeleaf.org/thymeleaf-extras-springsecurity5">
<head>
</head>
<body>
<div th:fragment="header">
<div class="container" align="right">
    <span>Logged user: <span sec:authentication="name"></span>
    <a th:href="@{/logout}">Sign Out</a>
</div>
</div>
</body>
</html>
```

Footer.html

```
<html xmlns="http://www.w3.org/1999/xhtml"
      xmlns:th="http://www.thymeleaf.org"
      xmlns:sec="http://www.thymeleaf.org/thymeleaf-extras-springsecurity5">
<head>
</head>
```

```

<body>
<div th:fragment="footer">

    <div class="container" align="center" style="padding-top:400px">

        <footer>
        <!-- this is footer -->
        © 2019 SIP
            <span sec:authorize="isAuthenticated()">
                | Logged user: <span sec:authentication="name"></span> |
                Roles: <span sec:authentication="principal.authorities"></span>
            </span>
        </footer>
    </div>

</div>
</body>
</html>

```

403.html

```

<!DOCTYPE html>
<html xmlns:th="http://www.thymeleaf.org">
<head>
<meta charset="ISO-8859-1">
<title>Insert title here</title>
</head>
<body>
You don't have permission to visit the page
<br/>
return to home page <a th:href="@{/home}">here</a>
</body>
</html>

```

Register new user

<http://localhost:8088/registration>

Validations

Registration Form

*Please provide your name







*Please provide your last name

*Please provide an email

*Your password must have at least 5 characters

*Please provide your password

As you can see the password has been stored with a **Hash algorithm** due we have implemented the BCryptPasswordEncoder in our AuthenticationManagerBuilder.

+ Options				user_id	active	email	last_name	name	password
<input type="checkbox"/>	 Editor	 Copier	 Supprimer	17	1	raouf@gmail.com	Raouf	AB	\$2a\$10\$7jV1jOO113YgDtBAn2FcQ.Eq6iUj3ubqkVNkmJaE4Eu...
<input type="checkbox"/>	 Editor	 Copier	 Supprimer	16	1	amine.mezghich@gmail.com	Mezghich	Mohamed Amine	\$2a\$10\$pr5FSBNj7QZVbXJJCeZDeCR2RLZiL6MOiIKZJT4q2d...

We Build Smart Solutions For You

Exercise (30 min)

- 1-Add some modifications to enable view for role without restriction
- 2-Add modification to list the user accounts and activate some of them

Response

1-

```
.antMatchers("/role/**").permitAll()
```

2-

```
.antMatchers("/accounts/**").permitAll()
```

➤ **AccountController.java**

```
package com.orsys.ams.controllers;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestParam;

import com.orsys.ams.entities.User;
```



```
import com.orsys.ams.repositories.UserRepository;

@Controller
@RequestMapping("/accounts/")

public class AccountController {

    private final UserRepository userRepository;

    @Autowired
    public AccountController(UserRepository userRepository) {
        this.userRepository = userRepository;
    }

    @GetMapping("list")
    public String listUsers(Model model) {
        List<User> users = (List<User>) userRepository.findAll();
        long nbr = userRepository.count();
        if(users.size()==0)
            users = null;
        model.addAttribute("users", users);
        model.addAttribute("nbr", nbr);
        return "user/listUsers";
    }
}
```

}

➤ listUsers.html

```
<!DOCTYPE html>
<html xmlns:th="http://www.thymeleaf.org">

<head>
<meta charset="utf-8">
<meta http-equiv="x-ua-compatible" content="ie=edge">
<title>Roles</title>
<meta name="viewport" content="width=device-width, initial-
scale=1">
<link rel="stylesheet"
    href="https://stackpath.bootstrapcdn.com/bootstrap/4.1.3/css/bo
otstrap.min.css"
    integrity="sha384-
MCw98/SFnGE8fJT3GXwEOngsV7Zt27NXFoaoApmYm81iuXoPkFOJwJ8ERdknLPMO"
    crossorigin="anonymous">
<link rel="stylesheet"
    href="https://use.fontawesome.com/releases/v5.4.1/css/all.css"
    integrity="sha384-
5sAR7xN1Nv6T6+dt2mhtzEpVJvfS3NScPQTr0xhwjIuvcA67KV2R5Jz6kr4abQsz"
    crossorigin="anonymous">
<!-- <link rel="stylesheet" href=" ../css/shards.min.css"> -->
</head>

<body>
    <div class="container my-2">
        <div class="card">
            <div class="card-body">
                <div th:switch="${users}" class="container my-5">

                    <div class="col-md-10">
                        <h2 th:case="null">No Role yet!</h2>
                        <div th:case="*">

                            <h2>Nombre total des Users= <span
th:text="${nbr}">10</span></h2>
```

```

<table class="table table-striped
table-responsive-md">

    <thead>
        <tr>
            <th>ID</th>
            <th>Name</th>
            <th>Email</th>
            <th>Role</th>
            <th>is Active</th>
        </tr>
    </thead>
    <tbody>
        <tr th:each="user :
${users}">
            <td>
th:text="${user.id}"</td>
            <td>
th:text="${user.name}"</td>
            <td>
th:text="${user.email}"</td>
            <td>
th:text="${user.roles[0].role}"</td>
            <td>
th:text="${user.active}"</td>
        </tr>
    </tbody>
</table>
</div>
</div>
</div>
</div>
</div>
</body>

</html>

```

Nombre total des Users= 3

ID	Name	Email	Role	is Active
16	mezghich	mezghich@gmail.com	ADMIN	0
17	med	med@gmail.com	USER	1
18	amine	amine@gmail.com	USER	1

3-Add actions to activate account (20 min)

4-1-Add modification in the configuration file, for **SUPERADMIN** to be able to visit both **providers and articles resources**. (10 min)

4-2- Apporter les modifications nécessaires pour pouvoir attribuer/modifier un rôle à chaque nouvel inscrit via l'interface /accounts/list (15 min)

5-Send email if the account is being activated. (30 min)

Update the action to send email if the account is activated.

(see <https://mkkyong.com/spring-boot/spring-boot-how-to-send-email-via-smtp/>)

Responses

3-

```
<!DOCTYPE html>
<html xmlns:th="http://www.thymeleaf.org">

<head>
<meta charset="utf-8">
<meta http-equiv="x-ua-compatible" content="ie=edge">
<title>Roles</title>
<meta name="viewport" content="width=device-width, initial-
scale=1">
<link rel="stylesheet"
```

```

    href="https://stackpath.bootstrapcdn.com/bootstrap/4.1.3/css/bootstrap.min.css"
    integrity="sha384-
MCw98/SFnGE8fJT3GXwEOngsV7Zt27NXFoaoApmYm81iuXoPkFOJwJ8ERdknLPMO"
    crossorigin="anonymous">
<link rel="stylesheet"
    href="https://use.fontawesome.com/releases/v5.4.1/css/all.css"
    integrity="sha384-
5sAR7xN1Nv6T6+dt2mhtzEpVJvfS3NScPQTr0xhwjIuvcaA67KV2R5Jz6kr4abQsz"
    crossorigin="anonymous">
<!-- <link rel="stylesheet" href="../../css/shards.min.css"> -->
</head>

<body>
    <div class="container my-2">
        <div class="card">
            <div class="card-body">
                <div th:switch="${users}" class="container my-5">
                    <div class="col-md-10">
                        <h2 th:case="null">No Role yet!</h2>
                        <div th:case="*">
                            <h2>Nombre total des Users= <span
th:text="${nbr}">10</span></h2>
                            <table class="table table-striped
table-responsive-md">
                                <thead>
                                    <tr>
                                        <th>ID</th>
                                        <th>Name</th>
                                        <th>Email</th>
                                        <th>Role</th>
                                        <th>is Active</th>
                                        <th>Activer</th>
                                        <th>Désactiver</th>
                                    </tr>
                                </thead>
                                <tbody>
                                    <tr th:each="user :
${users}">
                                        <td
th:text="${user.id}"></td>

```

```

th:text="${user.name}"></td>
th:text="${user.email}"></td>
th:text="${user.roles[0].role}"></td>
th:text="${user.active}"></td>

th:href="@{/accounts/enable/{id} (id=${user.id})}">Activer</a></td>
th:href="@{/accounts/disable/{id} (id=${user.id})}">Désactiver</a></td>
</tr>
</tbody>
</table>
</div>
</div>
</div>
</div>
</div>
</div>
</body>
</html>

```

```

package com.orsys.ams.controllers;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PathVariable;

```

```
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestParam;
import org.springframework.web.bind.annotation.ResponseBody;
```

```
import com.orsys.ams.entities.Provider;
import com.orsys.ams.entities.User;
import com.orsys.ams.repositories.UserRepository;
```

```
@Controller
```

```
@RequestMapping("/accounts/")
```

```
public class AccountController {
```

```
    private final UserRepository userRepository;
```

```
    @Autowired
```

```
    public AccountController(UserRepository userRepository) {
```

```
        this.userRepository = userRepository;
```

```
    }
```

```
    @GetMapping("list")
```

```
    public String listUsers(Model model) {
```

```
        List<User> users = (List<User>) userRepository.findAll();
```

```
        long nbr = userRepository.count();
```

```
        if(users.size()==0)
```

```
            users = null;
```

```

model.addAttribute("users", users);
model.addAttribute("nbr", nbr);
return "user/listUsers";
}

```

```

@GetMapping("enable/{id}")

```

```

//@ResponseBody

```

```

public String enableUserAccount(@PathVariable ("id") int id) {

```

```

    User user = userRepository.findById(id).orElseThrow(()->new
    IllegalArgumentException("Invalid User Id:" + id));

```

```

    user.setActive(1);

```

```

    userRepository.save(user);

```

```

    return "redirect:../list";

```

```

}

```

```

@GetMapping("disable/{id}")

```

```

//@ResponseBody

```

```

public String disableUserAccount(@PathVariable ("id") int id) {

```

```

    User user = userRepository.findById(id).orElseThrow(()->new
    IllegalArgumentException("Invalid User Id:" + id));

```

```

    user.setActive(0);

```

```

    userRepository.save(user);

```

```

    return "redirect:../list";

```

```

}

```



```
}
```

4-1

```
//.antMatchers("/provider/**").hasAuthority("ADMIN")

//.antMatchers("/article/**").hasAuthority("USER").anyRequest()

.antMatchers("/provider/**").hasAnyAuthority("ADMIN", "SUPERADMIN")
    .antMatchers("/article/**").hasAnyAuthority("USER",
"SUPERADMIN").anyRequest()
```

4-2

Listusers.html

```
<!DOCTYPE html>
<html xmlns:th="http://www.thymeleaf.org">

<head>
<meta charset="utf-8">
<meta http-equiv="x-ua-compatible" content="ie=edge">
<title>Roles</title>
<meta name="viewport" content="width=device-width, initial-
scale=1">
<link rel="stylesheet"
    href="https://stackpath.bootstrapcdn.com/bootstrap/4.1.3/css/bo
otstrap.min.css"
    integrity="sha384-
MCw98/SFnGE8fJT3GXwEOngsV7Zt27NXFoaoApmYm81iuXoPkFOJwJ8ERdknLPMO"
    crossorigin="anonymous">
<link rel="stylesheet"
    href="https://use.fontawesome.com/releases/v5.4.1/css/all.css"
    integrity="sha384-
5sAR7xN1Nv6T6+dt2mhtzEpVJvfS3NScPQTr0xhwjIuvca67KV2R5Jz6kr4abQsz"
    crossorigin="anonymous">
<!-- <link rel="stylesheet" href="../../css/shards.min.css"> -->
</head>

<body>
    <div class="container my-2">
        <div class="card">
            <div class="card-body">
                <div th:switch="${users}" class="container my-5">

                    <div class="col-md-10">
```

```

<h2 th:case="null">No Role yet!</h2>
<div th:case="*">

    <h2>Nombre total des Users= <span
th:text="\${nbr}">10</span></h2>

    <table class="table table-striped
table-responsive-md">

        <thead>
            <tr>
                <th>ID</th>
                <th>Name</th>
                <th>Email</th>
                <th>Role</th>
                <th>Update Role</th>
                <th>is Active</th>
                <th>Activer</th>
                <th>Désactiver</th>
            </tr>
        </thead>
        <tbody>
            <tr th:each="user :
\${users}">

                <td>
th:text="\${user.id}"</td>

                <td>
th:text="\${user.name}"</td>
                <td>
th:text="\${user.email}"</td>
                <td>
th:text="\${user.roles[0].role}"</td>

                <td>
<form method="post"

                    <select

                        <option
value="USER">USER</option>

                        <option
value="ADMIN">ADMIN</option>

                        <option
value="SUPERADMIN">SUPERADMIN</option>

                    </select>
<br/>

```

```

<input
type="hidden" name="id" th:value="${user.id}" th:attr="name='id'"/>
<input class="btn
btn-primary" type="submit" value="Update role"/>
</form>
</td>
<td>
th:text="${user.active}"</td>

<td><a
th:href="@{/accounts/enable/{id} (id=${user.id})}">Activer</a></td>
<td><a
th:href="@{/accounts/disable/{id}
(id=${user.id})}">Désactiver</a></td>

</tr>
</tbody>
</table>
</div>
</div>
</div>
</div>
</div>
</body>
Smart It Partner
We Build Smart Solutions For You
</html>

```

Nombre total des Users= 3

ID	Name	Email	Role	Update Role	is Active	Activer	Désactiver
23	Mohamed Amine	amine.mezghich@gmail.com	SUPERADMIN	<div>USER</div>	1	Activer	Désactiver
				<div>Update role</div>			
24	admin	admin@gmail.com	ADMIN	<div>USER</div>	0	Activer	Désactiver
				<div>Update role</div>			
30	super	super@gmail.com	SUPERADMIN	<div>USER</div>	1	Activer	Désactiver
				<div>Update role</div>			

Au niveau AccountController.java

```
@PostMapping("updateRole")
//@ResponseBody
public String updateUserRole(@RequestParam ("id") int id,
    @RequestParam ("newrole")String newRole
    ) {

    User user = userRepository.findById(id).orElseThrow(()->
    new IllegalArgumentException("Invalid User Id:" + id));

    Role userRole = roleRepository.findByRole(newRole);

    user.setRoles(new HashSet<Role>(Arrays.asList(userRole)));

    userRepository.save(user);
    return "redirect:list";
}
```

5-

Pom.xml

Smart IT PARTNER

We build Smart Solutions for You

<https://smart-it-partner.com/>

```
<!-- send email -->
    <dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-mail</artifactId>
    </dependency>
```

Appliation.properties

```
spring.mail.host=smtp.gmail.com
spring.mail.port=587
spring.mail.username=username
spring.mail.password=password
```

Other properties

```
spring.mail.properties.mail.smtp.auth=true
spring.mail.properties.mail.smtp.connectiontimeout=5000
spring.mail.properties.mail.smtp.timeout=5000
spring.mail.properties.mail.smtp.writetimeout=5000
```

TLS , port 587

```
spring.mail.properties.mail.smtp.starttls.enable=true
```

- la classe principale et la fonction d'envois de mail

```
package com.orsys.ams;

import java.io.File;
import java.util.ArrayList;
import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.boot.CommandLineRunner;
import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
import org.springframework.boot.web.servlet.support.SpringBootServletInitializer;
import org.springframework.web.SpringServletContainerInitializer;

import com.orsys.ams.controllers.ArticleController;
import com.orsys.ams.controllers.ProviderController;
```

```

import org.springframework.mail.SimpleMailMessage;
import org.springframework.mail.javamail.JavaMailSender;
import javax.mail.MessagingException;
import java.io.IOException;

@SpringBootApplication
public class OrsysDataApplication extends
SpringBootServletInitializer implements CommandLineRunner{

    @Autowired
    private JavaMailSender javaMailSender;

    void sendEmail() {

        SimpleMailMessage msg = new SimpleMailMessage();
        msg.setTo("amine.mezghich@gmail.com");

        msg.setSubject("Testing from Spring Boot");
        msg.setText("Hello World \n Spring Boot Email");

        javaMailSender.send(msg);
    }

    @Override
    public void run(String... args) throws MessagingException,
IOException {

        System.out.println("Sending Email...");

        sendEmail();

        System.out.println("Done");
    }

    public static void main(String[] args) {

        new File(ArticleController.uploadDirectory).mkdir();
        new
File(ProviderController.uploadDirectoryProvider).mkdir();

```

```

SpringApplication.run(OrsysDataApplication.class, args);

/*Vehicule v = new Voiture();

String ch="";
String ch2 = null;

List<String> ls = new ArrayList();
List<String> ls2 = null;
//ls2.add("ab");
System.out.println(ls.isEmpty());
ls.add("Java");
System.out.println(ls.isEmpty());*/
}
}

```

Correction

- **Partie html**

```

<!DOCTYPE html>
<html xmlns:th="http://www.thymeleaf.org">

<head>
<meta charset="utf-8">
<meta http-equiv="x-ua-compatible" content="ie=edge">
<title>Roles</title>
<meta name="viewport" content="width=device-width, initial-
scale=1">
<link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.1.3/css/bo
otstrap.min.css"
integrity="sha384-
MCw98/SFnGE8fJT3GXwEOngsV7Zt27NXFoaoApmYm81iuXoPkFOJwJ8ERdknLPMO"
crossorigin="anonymous">
<link rel="stylesheet"
href="https://use.fontawesome.com/releases/v5.4.1/css/all.css"
integrity="sha384-
5sAR7xN1Nv6T6+dT2mhtzEpVJvfS3NScPQTrOxhwjIuvcA67KV2R5Jz6kr4abQsz"
crossorigin="anonymous">

```

```

<!-- <link rel="stylesheet" href="../../css/shards.min.css"> -->
</head>

<body>
  <div class="container my-2">
    <div class="card">
      <div class="card-body">
        <div th:switch="{users}" class="container my-5">

          <div class="col-md-10">
            <h2 th:case="null">No Role yet!</h2>
            <div th:case="*">

              <h2>Nombre total des Users= <span
th:text="{nbr}">10</span></h2>

              <table class="table table-striped
table-responsive-md">
                <thead>
                  <tr>
                    <th>ID</th>
                    <th>Name</th>
                    <th>Email</th>
                    <th>Role</th>
                    <th>Update Role</th>
                    <th>is Active</th>
                    <th>Activer</th>
                    <th>Désactiver</th>
                  </tr>
                </thead>
                <tbody>
                  <tr th:each="user :
{users}">
                    <td
th:text="{user.id}"></td>
                    <td
th:text="{user.name}"></td>
                    <td
th:text="{user.email}"></td>
                    <td
th:text="{user.roles[0].role}"></td>
                    <td>
                      <form method="post"
th:action="@{/accounts/updateRole}">

```



```

name="newrole" class="form-control">
value="USER">USER</option>
value="ADMIN">ADMIN</option>
value="SUPERADMIN">SUPERADMIN</option>
</select>
<br/>
<input
type="hidden" name="id" th:value="{user.id}" th:attr="name='id'"/>
<input class="btn
btn-primary" type="submit" value="Update role"/>
</form>
</td>
<td>
th:text="{user.active}"</td>
<td><a
th:href="@{/accounts/enable/{id}/{email} (id={user.id},
email={user.email})}">Activer</a></td>
<td><a
th:href="@{/accounts/disable/{id}/{email} (id={user.id},
email={user.email})}">Désactiver</a></td>
</tr>
</tbody>
</table>
</div>
</div>
</div>
</div>
</div>
</body>
</html>

```

- Le controleur

```
package com.orsys.ams.controllers;

import java.util.Arrays;
import java.util.HashSet;
import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestParam;
import org.springframework.web.bind.annotation.ResponseBody;

import com.orsys.ams.entities.Provider;
import com.orsys.ams.entities.Role;
import com.orsys.ams.entities.User;
import com.orsys.ams.repositories.RoleRepository;
import com.orsys.ams.repositories.UserRepository;

import org.springframework.mail.SimpleMailMessage;
import org.springframework.mail.javamail.JavaMailSender;
import javax.mail.MessagingException;
import java.io.IOException;

@Controller
@RequestMapping("/accounts/")

public class AccountController {

    private final UserRepository userRepository;
    private final RoleRepository roleRepository;

    @Autowired
    private JavaMailSender javaMailSender;

    @Autowired
```

```

    public AccountController(UserRepository userRepository,
RoleRepository roleRepository) {
        this.userRepository = userRepository;
        this.roleRepository = roleRepository;
    }

    @GetMapping("list")
    public String listUsers(Model model) {

        List<User> users = (List<User>) userRepository.findAll();
        long nbr = userRepository.count();
        if(users.size()==0)
            users = null;
        model.addAttribute("users", users);
        model.addAttribute("nbr", nbr);
        return "user/listUsers";
    }

    @GetMapping("enable/{id}/{email}")
    // @ResponseBody
    public String enableUserAccount(@PathVariable ("id") int id,
        @PathVariable ("email") String email) {

        sendEmail(email, true);
        User user = userRepository.findById(id).orElseThrow(() -
> new IllegalArgumentException("Invalid User Id:" + id));
        user.setActive(1);
        userRepository.save(user);
        return "redirect:../..../list";
    }

    @GetMapping("disable/{id}/{email}")
    // @ResponseBody
    public String disableUserAccount(@PathVariable ("id") int id,
        @PathVariable ("email") String email) {

        sendEmail(email, false);

        User user = userRepository.findById(id).orElseThrow(() -
> new IllegalArgumentException("Invalid User Id:" + id));
        user.setActive(0);
        userRepository.save(user);
        return "redirect:../..../list";
    }
}

```

```

@PostMapping("updateRole")
//@ResponseBody
public String UpdateUserRole(@RequestParam ("id") int id,
    @RequestParam ("newrole")String newRole
    ) {

    User user = userRepository.findById(id).orElseThrow(()-
>new IllegalArgumentException("Invalid User Id:" + id));

    Role userRole = roleRepository.findByRole(newRole);

    user.setRoles(new HashSet<Role>(Arrays.asList(userRole)));

    userRepository.save(user);
    return "redirect:list";
}

void sendEmail(String email, boolean state) {

    SimpleMailMessage msg = new SimpleMailMessage();
    msg.setTo(email);
    if(state == true)
    {
        msg.setSubject("Account Has Been Activated");
        msg.setText("Hello, Your account has been activated. "
            + "You can log in : http://127.0.0.1:81/login"
            + " \n Best Regards!");
    }
    else
    {
        msg.setSubject("Account Has Been disactivated");
        msg.setText("Hello, Your account has been
disactivated.");
    }
    javaMailSender.send(msg);

}

}


```

Account Has Been Activated

Boîte de réception x

 mohamed.mezghich@sesame.com.tn

À moi ▾

 anglais ▾ > français ▾ [Traduire le message](#)

Hello, Your account has been activated. You can log in : <http://127.0.0.1:81/login>
Best Regards!



Smart It Partner

We Build Smart Solutions For You