

## DEPARTEMENT MATHEMATIQUES ET INFORMATIQUE

# Compte Rendu

**Filière :**  
**« Génie du Logiciel et des Systèmes Informatiques Distribués »**  
**GLSID**

**Etudiant Spring**

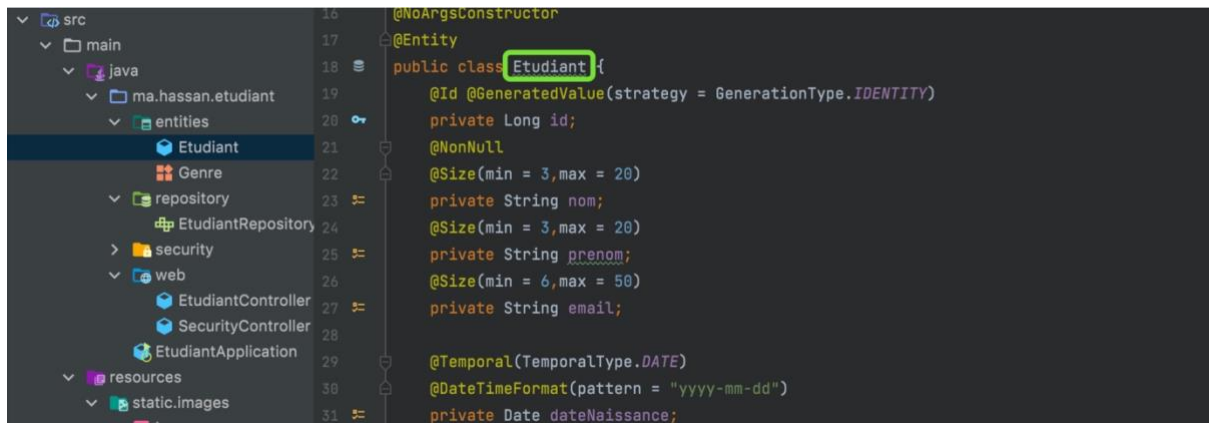
**le 23/4/2022**

Réalisé par :  
Hassan ELMAKHLOUFI

Encadré par :  
M.YOUSFI

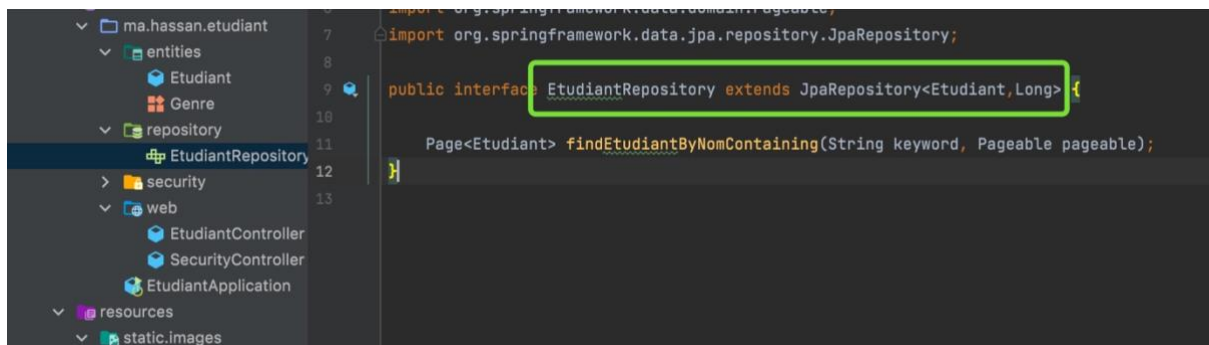
**Année Universitaire : 2021-2022**

Entités étudiant :



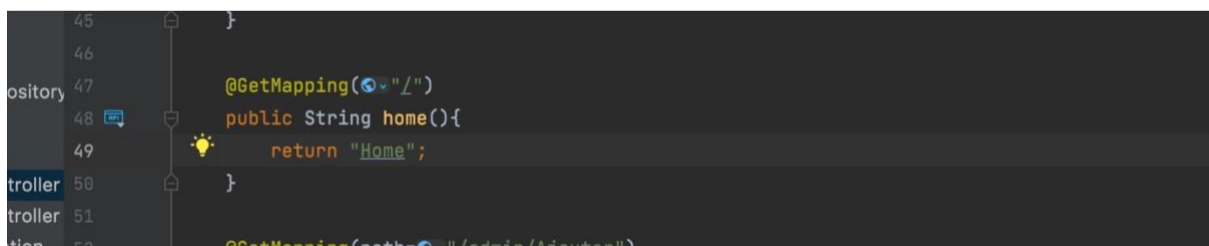
```
16 @NoArgConstructor
17 @Entity
18 public class Etudiant {
19     @Id @GeneratedValue(strategy = GenerationType.IDENTITY)
20     private Long id;
21     @NotNull
22     @Size(min = 3, max = 20)
23     private String nom;
24     @Size(min = 3, max = 20)
25     private String prenom;
26     @Size(min = 6, max = 50)
27     private String email;
28
29     @Temporal(TemporalType.DATE)
30     @DateTimeFormat(pattern = "yyyy-mm-dd")
31     private Date dateNaissance;
```

-Repository étudiant



```
7 import org.springframework.data.jpa.repository.JpaRepository;
8
9 public interface EtudiantRepository extends JpaRepository<Etudiant, Long> {
10
11     Page<Etudiant> findEtudiantByNomContaining(String keyword, Pageable pageable);
12 }
13
```

-Controller (fonction qui retourne la vue du page d'accueil)



```
45 }
46
47 @GetMapping("/")
48 public String home() {
49     return "Home";
50 }
51
52 @GetMapping("/admin/ajouter")
```

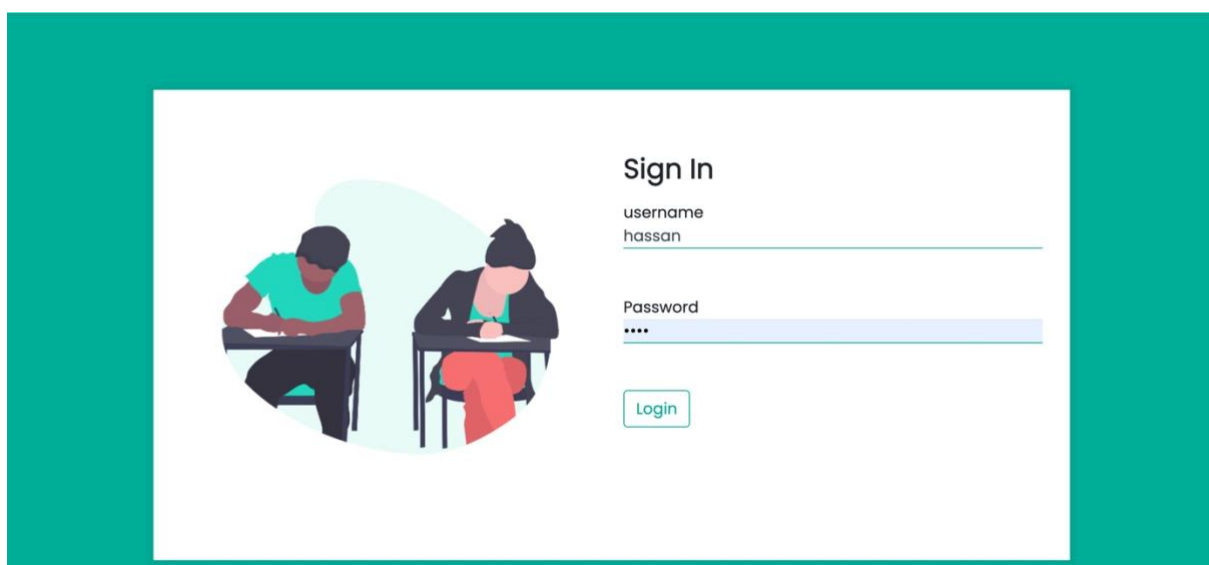
La vue HTML :

```
1  <!DOCTYPE html>
2  <html lang="en"
3      xmlns:th="http://www.thymeleaf.org"
4      xmlns:layout="http://www.ultraq.net.nz/thymeleaf/layout"
5      layout:decorate="template1">
6  <head>
7      <meta charset="UTF-8">
8      <title>Home Page</title>
9  </head>
10 <body>
11 <div layout:fragment="content">
12 <h1>Home page</h1>
13
14 </div>
15 </body>
```

Controller (la vue qui retourne index )

```
27 @GetMapping(path="/user/index")
28 public String etudiants(Model model,
29     @RequestParam(name="page",defaultValue = "0") int page,
30     @RequestParam(name="size",defaultValue = "5") int size,
31     @RequestParam(name="keyword",defaultValue = "" ) String keyword){
32     Page<Etudiant> pageetudiants = etudiantRepository.findEtudiantByNomContaining(keyword,PageRequest.of(page,size));
33     model.addAttribute( attributeName: "etudiantList",pageetudiants.getContent());
34     model.addAttribute( attributeName: "pages",new int[pageetudiants.getTotalPages()]);
35     model.addAttribute( attributeName: "currentPage",page);
36     model.addAttribute( attributeName: "keyword",keyword);
37
38     return "etudiant";
39 }
40
```

Page d'authentification



Le fichier HTML du page index

```

5  layout:decorate="template1">
6  <head>
7      <meta charset="UTF-8">
8      <title>Title</title>
9      <link rel="stylesheet" href="/webjars/bootstrap/5.1.3/css/bootstrap.min.css">
10 </head>
11 <body>
12 <div layout:fragment="content">
13
14     <div class="container mt-lg-5">
15         <div class="card">
16             <div class="card-header" style="background-color: #2c3e50; color: white; padding: 5px;>
17                 <div>liste des etudiants</div>
18
19                 <div> <a style="color: white; text-decoration: none;" th:href="@{/admin/Ajouter}">Ajouter</a></div>
20
21                 <form method="get" class="d-flex" style="background-color: #2c3e50; color: white; padding: 5px;" th:action="@{/user/index}">
22
23
24

```

Home
mohamed ▾

liste des etudiants							Ajouter	
							<input type="text"/>	chercher
id	nom	Prenom	Email	Date	Genre	Regle		
2	ilyass	elmakhloufi	ilyas@gmail.com	2022-01-11	femme	false	Delete	Edit
3	ahmed	elmakhloufi	ahmed@gmail.com	2022-04-11	Homme	true	Delete	Edit
4	hassan	elmakhloufi	hassan@gmail.com	2022-04-11	Homme	true	Delete	Edit
5	ilyas	elmakhloufi	ilyas@gmail.com	2022-04-11	Homme	true	Delete	Edit
6	ahmed	elmakhloufi	ahmed@gmail.com	2022-04-11	Homme	true	Delete	Edit

0
1
2
3
4
5
6
7
8
9
10

## Controller Ajout

```

51
52 @GetMapping(path="/admin/Ajouter")
53 public String Ajouter(Model model){
54     model.addAttribute( attributeName: "etudiant", new Etudiant());
55     model.addAttribute( attributeName: "mode", attributeValue: "new");
56     return "EtudiantForm";
57 }
58 @PostMapping(path="/admin/saveEtudiant")
59 public String saveEtudiant(@Valid Etudiant etudiant, BindingResult bindingResult){
60     if(bindingResult.hasErrors()){
61         return "EtudiantForm";
62     }
63     etudiantRepository.save(etudiant);
64     return "redirect:/user/index";
65 }

```

## Formulaire d'ajout

Ajouter un etudiant

Id :

Nom :

Prenom :

Email :

Date Naissance :

jj/mm/aaaa

Genre :

Male

Contoller modification :

```
@PostMapping("/admin/saveEtudiant")
public String saveEtudiant(@Valid Etudiant etudiant, BindingResult bindingResult){
    if(bindingResult.hasErrors())
        return "EtudiantForm";
    etudiantRepository.save(etudiant);
    return "redirect:/user/index";
}

@GetMapping("/admin/edit")
public String edit(Model model ,Long id){

    Etudiant etudiant = etudiantRepository.findById(id).get();
    model.addAttribute( attributeName: "etudiant",etudiant);
    model.addAttribute( attributeName: "mode", attributeValue: "edit");
    return "EtudiantForm";
}
```

Modifier l'etudiant

Id :

2

Nom :

ilyasss

Prenom :

elmakhloufi

Email :

ilyas@gmail.com

Date Naissance :

11/01/2022

Genre :

Female

Formulaire d'ajout et de modification :

```

Etudiant.java x EtudiantRepository.java x EtudiantController.java x EtudiantForm.html x login.html x Home.html x application.
3      xmlns:layout="http://www.ultraq.net.nz/thymeleaf/layout"
4      layout:decorate="template1"
5
6  <head>
7      <meta charset="UTF-8">
8      <title>Title</title>
9      <link rel="stylesheet" href="/webjars/bootstrap/5.1.3/css/bootstrap.min.css">
10 </head>
11 <body>
12
13 <div layout:fragment="content">
14 <div class="container mt-lg-5">
15 <div class="card">
16 <div class="card-header" style="background-color: #f8d7da;">
17 <div th:if="${mode}=='new'">Ajouter un etudiant</div>
18 <div th:if="${mode}=='edit'">Modifier l'etudiant</div>
19 </div>
20 <div class="card-body">
21 <div class="container">
22 <form th:action="@{saveEtudiant}" method="post">
23
24 <div class="form-group mt-lg-4">
25 <label class="control-label">Id :</label>
26 <input type="text" th:value="${etudiant.id}" name="id" class="form-control" style="width: 100%;"/>

```

Spring security :

Entité utilisateur pour l'authentification :

```

11  @Entity
12  @Data
13  @AllArgsConstructor
14  @NoArgsConstructor
15  public class AppUser {
16
17      @Id
18      private String userid ;
19      @Column(unique = true)
20      private String username ;
21      private String password ;
22      private boolean active;
23
24      @ManyToMany(fetch = FetchType.EAGER)
25      private List<AppRole> appRoles = new ArrayList<>();
26
27  }

```

Entité rôle pour l'authentification :

```

2
3  import ...
8
9  @Entity
10  @Data
11  @NoArgsConstructor
12  @AllArgsConstructor
13  public class AppRole {
14
15      @Id @GeneratedValue(strategy = GenerationType.IDENTITY)
16      private Long roleId;
17      @Column(unique = true)
18      private String roleName;
19      private String description;
20  }

```

Repository utilisateur :

```

1  package ma.hassan.etudiant.security.repositories;
2
3  import ma.hassan.etudiant.security.entities.AppUser;
4  import org.springframework.data.jpa.repository.JpaRepository;
5
6  public interface AppUserRepository extends JpaRepository<AppUser,String> {
7
8      AppUser findAppUserByUsername(String id);
9
10 }

```

Repository Role



The screenshot shows an IDE with a project structure on the left and a code editor on the right. The project structure includes a 'main' package with 'java' sub-packages: 'ma.hassan.etudiant' (containing 'entities' with 'Etudiant', 'Genre' and 'repository' with 'EtudiantRepository') and 'security' (containing 'AppRoleRepository'). The code editor shows the following Java code:

```

3  import ma.hassan.etudiant.security.entities.AppRole;
4  import org.springframework.data.jpa.repository.JpaRepository;
5
6  public interface AppRoleRepository extends JpaRepository<AppRole, Long> {
7
8      AppRole findAppRoleByRoleName(String Role);
9  }

```

Interface Security Service :

The screenshot shows an IDE with a project structure on the left and a code editor on the right. The project structure includes a 'main' package with 'java' sub-packages: 'ma.hassan.etudiant' (containing 'entities' with 'Etudiant', 'Genre' and 'repository' with 'EtudiantRepository') and 'security' (containing 'entities' with 'AppRole', 'AppUser', 'repositories' with 'AppRoleRepository', 'AppUserRepository' and 'service' with 'SecurityService', 'SecurityServiceImpl'). The code editor shows the following Java code:

```

3  import ma.hassan.etudiant.security.entities.AppUser;
4  import ma.hassan.etudiant.security.entities.AppRole;
5  import ma.hassan.etudiant.security.entities.AppUser;
6
7  public interface SecurityService {
8
9      AppUser saveNewUser(String username, String password, String rePassword);
10     AppRole saveNewRole(String roleName, String description);
11     void addRoleToUser(String username, String roleName);
12     void removeRoleFromUser(String username, String roleName);
13     AppUser loadUserByUsername(String username);
14 }

```

Une implementation de l'interface security service

The screenshot shows an IDE with a project structure on the left and a code editor on the right. The project structure includes a 'main' package with 'java' sub-packages: 'ma.hassan.etudiant' (containing 'entities' with 'Etudiant', 'Genre' and 'repository' with 'EtudiantRepository') and 'security' (containing 'entities' with 'AppRole', 'AppUser', 'repositories' with 'AppRoleRepository', 'AppUserRepository' and 'service' with 'SecurityService', 'SecurityServiceImpl', 'UserDetailsService', 'SecurityConfig'). The code editor shows the following Java code:

```

15  import org.springframework.security.crypto.password.PasswordEncoder;
16  @Service
17  @Slf4j
18  @AllArgsConstructor
19  @Transactional
20  public class SecurityServiceImpl implements SecurityService {
21
22      private AppUserRepository appUserRepository;
23      private AppRoleRepository appRoleRepository;
24      private PasswordEncoder myPasswordEncoder;
25
26      @Override
27      public AppUser saveNewUser(String username, String password, String rePassword) {
28          if (!password.equals(rePassword)) throw new RuntimeException("Password not match");
29          String hashedPWD = myPasswordEncoder.encode(password);
30
31          AppUser appUser = new AppUser();
32          appUser.setUserid(UUID.randomUUID().toString());
33          appUser.setUsername(username);
34          appUser.setPassword(hashedPWD);
35      }

```

User details service



```

15 import java.util.Collection;
16 import java.util.stream.Collectors;
17
18 @Service
19 public class UserDetailsServiceImpl implements UserDetailsService {
20     @Autowired
21     private SecurityService securityService;
22
23     @Override
24     public UserDetails loadUserByUsername(String username) throws UsernameNotFoundException {
25         AppUser appUser = securityService.loadUserByUsername(username);
26         Collection<GrantedAuthority> authorities = new ArrayList<>();
27         appUser.getAppRoles().forEach(role ->{
28             SimpleGrantedAuthority authority = new SimpleGrantedAuthority(role.getRoleName());
29             authorities.add(authority);
30         });
31
32         Collection<GrantedAuthority> authorities1 =
33             appUser.getAppRoles().stream().map(role -> new SimpleGrantedAuthority((role.getRoleName()
34                 .collect(Collectors.toList());
35
36         User user = new User(appUser.getUsername(), appUser.getPassword(), authorities1);

```

Security configuration :

```

18 public class SecurityConfig extends WebSecurityConfigurerAdapter {
19
20     @Autowired
21     DataSource dataSource;
22
23     @Autowired
24     private UserDetailsServiceImpl userDetailsService;
25
26     @Autowired
27     private PasswordEncoder passwordEncoder;
28
29     @Override
30     protected void configure(AuthenticationManagerBuilder auth) throws Exception {
31         System.out.println(passwordEncoder.encode("rawPassword: 1234"));
32
33         auth.inMemoryAuthentication().withUser("user1").password(passwordEncoder.encode(
34             // .withUser("admin1").password(passwordEncoder.encode("12345")).roles("adm
35             // auth.jdbcAuthentication().dataSource(dataSource)
36             // .usersByUsernameQuery("select username as principal ,password as credent
37             // .authoritiesByUsernameQuery("select username principal ,role as user_ro
38             // .rolePrefix("ROLE_")
39             // .passwordEncoder(passwordEncoder());
40
41         auth.userDetailsService(userDetailsService);

```

Security Controller :

```

6 @Controller
7 public class SecurityController {
8
9     @GetMapping("/403")
10     public String notAuthorized() { return "403"; }
11
12     @GetMapping("/login")
13     public String login() { return "login"; }
14
15 }

```