Activite4:

Use Case JPA Hibernate Spring Data Many To Many Case

Réaliser par :

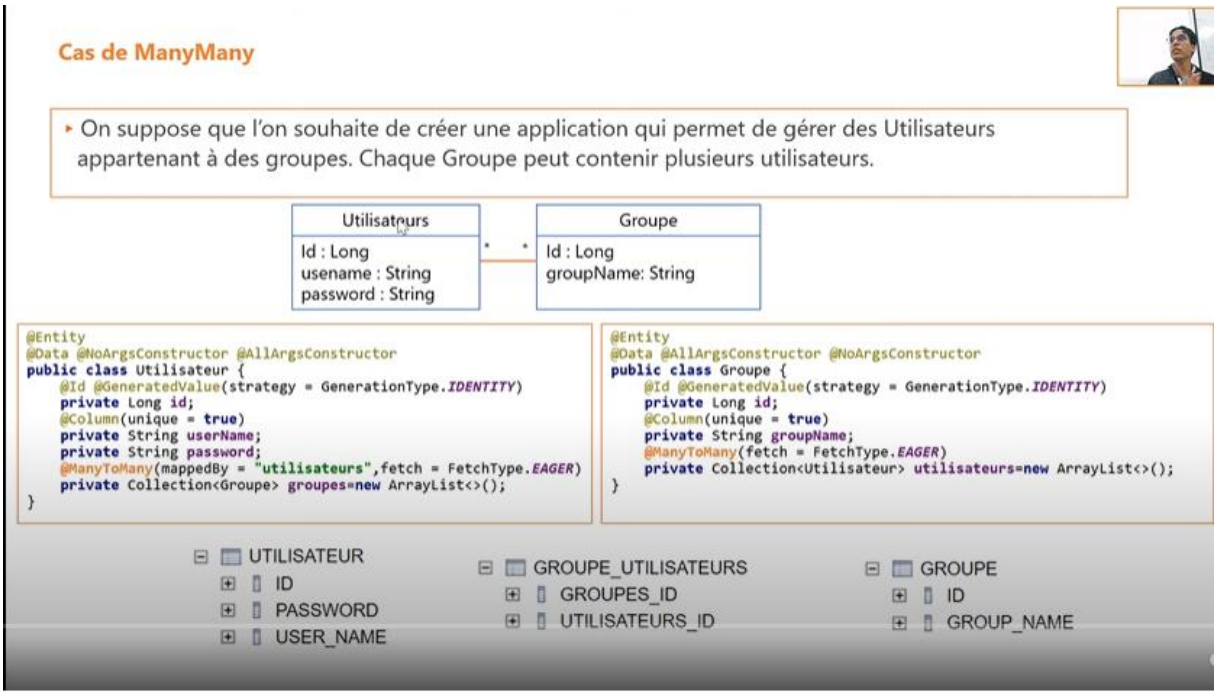
Mohamed amine

KHAMMOUR

Professeur :

Mr. Mohamed

YOUSSFI



La classe ROLE

```
package ma.enset.relationmanytomany.Entity;

import com.fasterxml.jackson.annotation.JsonProperty;
import lombok.AllArgsConstructor;
import lombok.Data;
import lombok.NoArgsConstructor;

import javax.persistence.*;
import java.util.ArrayList;
import java.util.List;
@Entity
@Data
@NoArgsConstructor
@AllArgsConstructor
public class Role {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;
    private String descri;
    @Column(unique = true, length = 20)
    private  String roleName;
    @ManyToMany(mappedBy = "roles",fetch = FetchType.EAGER)
    @JsonProperty(access = JsonProperty.Access.WRITE_ONLY)

    // @JoinTable(name = "USERS_ROLES") // pour changé le nom du table
jointure
    private List<User> users = new ArrayList<>();
}
```

La class USER

```
package ma.enset.relationmanytomany.Entity;

import lombok.AllArgsConstructor;
import lombok.Data;
```

```
import lombok.NoArgsConstructor;

import lombok.AllArgsConstructor;
import lombok.Data;
import lombok.NoArgsConstructor;

import javax.persistence.*;
import java.util.ArrayList;
import java.util.List;

@Entity
@Data @NoArgsConstructor @AllArgsConstructor
public class User {
    @Id
    private  String userId;
    @Column(unique = true, length = 20)
    private String userName;
    private String password;
    @ManyToMany(fetch = FetchType.EAGER)
    private List<Role> roles = new ArrayList<>();
}
```

l'interface RoleReposetories:

```
package ma.enset.relationmanytomany.Reposetories;

import ma.enset.relationmanytomany.Entity.Role;
import org.springframework.data.jpa.repository.JpaRepository;


public interface RoleReposetories extends JpaRepository<Role,Long>
{
    Role findByRoleName(String UserName);
}
```

l'interface UserReposetory :

```
package ma.enset.relationmanytomany.Reposetories;

import ma.enset.relationmanytomany.Entity.User;
import org.springframework.data.jpa.repository.JpaRepository;

public interface UserReposetories extends JpaRepository<User,String> {
    User findByUserName(String username);

}
```

L'interface UserService :

```
package ma.enset.relationmanytomany.Service;

import ma.enset.relationmanytomany.Entity.Role;
import ma.enset.relationmanytomany.Entity.User;

import java.security.Key;
import java.security.SecureRandom;
```

```
public interface UserService {
    User AddNewUser(User user) throws Exception;
    Role AddNewRole(Role role);
    User FindUserByUserName(String NameUser);
    Role FindRoleByRoleName(String NameRole);
    void AddRoleToUser(String UseName,String RoleName);
    User authenticate(String username, String password) throws Exception;
     Key generateKey() throws Exception;
    String decrypt(String encryptedValue) throws Exception;
   String  encrypt(String valueToEnc) throws Exception;
}
```

l'implémentation de cette interface dans la class UserServiceImp :

les fonction Encrept et decrypt permet de Hache le mot de passe de chaque User.

```
package ma.enset.relationmanytomany.Service;

import lombok.AllArgsConstructor;
import ma.enset.relationmanytomany.Entity.Role;
import ma.enset.relationmanytomany.Entity.User;
import ma.enset.relationmanytomany.Reposetories.RoleReposetories;
import ma.enset.relationmanytomany.Reposetories.UserReposetories;
import org.springframework.stereotype.Service;
import sun.misc.BASE64Decoder;
import sun.misc.BASE64Encoder;

import javax.crypto.Cipher;
import javax.crypto.spec.SecretKeySpec;
import javax.transaction.Transactional;
import java.security.Key;
import java.util.UUID;

@Service
@Transactional
@AllArgsConstructor
public class UserServiceImpl implements UserService {
    private UserReposetories userReposetories;
    private RoleReposetories roleReposetories;
    public Key generateKey() throws Exception {
        byte[] keyAsBytes;
        keyAsBytes = myEncryptionKey.getBytes(UNICODE_FORMAT);
        Key key = new SecretKeySpec(keyAsBytes, ALGORITHM);
        return key;
    }
    private static final String ALGORITHM       = "AES";
    private static final String myEncryptionKey = "ThisIsFoundation";
    private static final String UNICODE_FORMAT  = "UTF8";

    public String encrypt(String valueToEnc) throws Exception {
        Key key = generateKey();
        Cipher c = Cipher.getInstance(ALGORITHM);
        c.init(Cipher.ENCRYPT_MODE, key);
        byte[] encValue = c.doFinal(valueToEnc.getBytes());
        String encryptedValue = new BASE64Encoder().encode(encValue);
        return encryptedValue;
    }

    public String decrypt(String encryptedValue) throws Exception {
```

```
        Key key = generateKey();
        Cipher c = Cipher.getInstance(ALGORITHM);
        c.init(Cipher.DECRYPT_MODE, key);
        byte[] decordedValue = new
BASE64Decoder().decodeBuffer(encryptedValue);
        byte[] decValue = c.doFinal(decordedValue);//////////LINE 50
        String decryptedValue = new String(decValue);
        return decryptedValue;
    }


    @Override

    public User AddNewUser(User user) throws Exception {
        user.setPassword(encrypt(user.getPassword()));
    user.setUserId(UUID.randomUUID().toString());
        return userReposetories.save(user);
    }

    public Role AddNewRole(Role role) {
        return roleReposetories.save(role);
    }

    @Override
    public User FindUserByUserName(String NameUser) {
        return userReposetories.findByUserName(NameUser);
    }


    @Override
    public Role FindRoleByRoleName(String RoleName) {
        return roleReposetories.findByRoleName(RoleName);
    }

    @Override
    public void AddRoleToUser(String UseName, String RoleName) {

        Role role = FindRoleByRoleName(RoleName);
        User user = FindUserByUserName(UseName);
        user.getRoles().add(role);
        role.getUsers().add(user);

    }

    @Override
    public User authenticate(String username, String password) throws
Exception {
        User user=FindUserByUserName(username);
        if(user==null) throw new RuntimeException("user name or password
incorect");
        if(decrypt(user.getPassword()).equals(password)){
            return  user;
        }
        throw new RuntimeException("user name or password incorect");
    }
}
```

la class main :

```
package ma.enset.relationmanytomany;

import ma.enset.relationmanytomany.Entity.Role;
import ma.enset.relationmanytomany.Entity.User;
import ma.enset.relationmanytomany.Service.UserService;
import org.springframework.boot.CommandLineRunner;
import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
import org.springframework.context.annotation.Bean;

import java.util.stream.Stream;

@SpringBootApplication
public class RelationManyToManyApplication {

    public static void main(String[] args) {
        SpringApplication.run(RelationManyToManyApplication.class, args);
    }
@Bean
    CommandLineRunner Start(UserService userService){
        return args -> {
            User u1=new User();
            u1.setUserName("amine1");
            u1.setPassword("1425");
            userService.AddNewUser(u1);
            User u2=new User();
            u2.setUserName("Ayoub");
            u2.setPassword("1425fff");
            userService.AddNewUser(u2);
            Stream.of("Student","USER","ADMIN").forEach(r->{
                Role role =new Role();
                role.setRoleName(r);
                role.setDescri("my role is"+r);
                userService.AddNewRole(role);

            });
            Role role1=userService.FindRoleByRoleName("Student");
            Role role21 =userService.FindRoleByRoleName(("USER"));
            userService.AddRoleToUser("amine1", role1.getRoleName());
            userService.AddRoleToUser("amine1", role21.getRoleName());
            User user=userService.authenticate("amine1","1425");

System.out.println("==============================================");
            System.out.println(user.getUserName());
            for (Role role2: user.getRoles()) {
                System.out.println("role=====>"+role2.getRoleName());
            }

System.out.println("==============================================");

        };
}
}
```

Resultat :

```
2022-03-19 19:23:53.029  INFO 5724 --- [  restartedMain] m.e.r.RelationManyToManyApplication
==============================================
amine1
role=====>Student
role=====>USER
==============================================
```

La base de donnés :

Table	Action						Lignes	Type	Interclassement	Taille	Perte
role	Parcourir	Structure	Rechercher	Insérer	Vider	Supprimer	3	InnoDB	utf8mb4_general_ci	32,0 kio	-
user	Parcourir	Structure	Rechercher	Insérer	Vider	Supprimer	2	InnoDB	utf8mb4_general_ci	32,0 kio	-
user_roles	Parcourir	Structure	Rechercher	Insérer	Vider	Supprimer	2	InnoDB	utf8mb4_general_ci	48,0 kio	-
3 tables	Somme						7	InnoDB	utf8mb4_general_ci	112,0 kio	0 o

La table Rôle :

+ Options

			id	descri	role_name
Éditer	Copier	Supprimer	1	my role isStudent	Student
Éditer	Copier	Supprimer	2	my role isUSER	USER
Éditer	Copier	Supprimer	3	my role isADMIN	ADMIN

La table User :(mot de passe hache)

+ Options

			user_id	password	user_name
Éditer	Copier	Supprimer	422fbe6e-c9a0-4faa-baea-054d25e671fe	yS5RAEQZ9FvCB8+FAawxWg==	amine1
Éditer	Copier	Supprimer	6349551a-6e97-43e3-ba85-7f758efb92d1	lqkv83UFh0/YiqE/lLROjQ==	Ayoub

Tout cocher *Avec la sélection :* Éditer Copier Supprimer Exporter

La table User_Roles :

users_user_id	roles_id
422fbe6e-c9a0-4faa-baea-054d25e671fe	1
422fbe6e-c9a0-4faa-baea-054d25e671fe	2

La partie Web :

```
package ma.enset.relationmanytomany.Web;

import lombok.AllArgsConstructor;
import ma.enset.relationmanytomany.Entity.User;
import ma.enset.relationmanytomany.Service.UserService;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.RestController;



@AllArgsConstructor
@RestController
```

```java
public class Controller {
    private UserService userService;
    @GetMapping("/users/{username}")
    public User user(@PathVariable String username){
        User user=userService.FindUserByUserName(username);
        return  user;
    }

}
```

localhost:8086/users/amine1

Gmail YouTube Maps GitHub Student De... Diagramme vierge:... Chapitre-5-SURDEF... anglais (1) Tutoriel PHP : Cr... Messagerie ENSET... Convert HTML to W... Reading list


```
// 20220319193010
// http://localhost:8086/users/amine1

{
  "userId": "422fbe6e-c9a0-4faa-baea-054d25e671fe",
  "userName": "amine1",
  "password": "yS5RAEQZ9FvCB8+FAawxWg==",
  "roles": [
    {
      "id": 1,
      "descri": "my role isStudent",
      "roleName": "Student"
    },
    {
      "id": 2,
      "descri": "my role isUSER",
      "roleName": "USER"
    }
  ]
}
```