



# **Rapport de projet IA2R** **Interface Web**

*Tuteur : Mr Samir Youcef*

Antoine Gaultier

Gaël Stabarin

Robin Martineau

2020-2021



UNIVERSITÉ  
DE LORRAINE

LORRAINE INP  
vos talents se lèvent à l'Est



## Sommaire

<b>PARTIE A</b> - PARTIE A - Présentation du projet .....	3
<b>PARTIE B</b> - PARTIE B - Interface graphique .....	4
<b>PARTIE C</b> - PARTIE C - Espace client et connexion .....	4
<b>PARTIE D</b> - PARTIE D - Cryptage des mots de passe .....	7
<b>PARTIE E</b> - PARTIE E - Base de donnée musicale .....	8
<b>PARTIE F</b> - PARTIE F - Conclusion .....	122

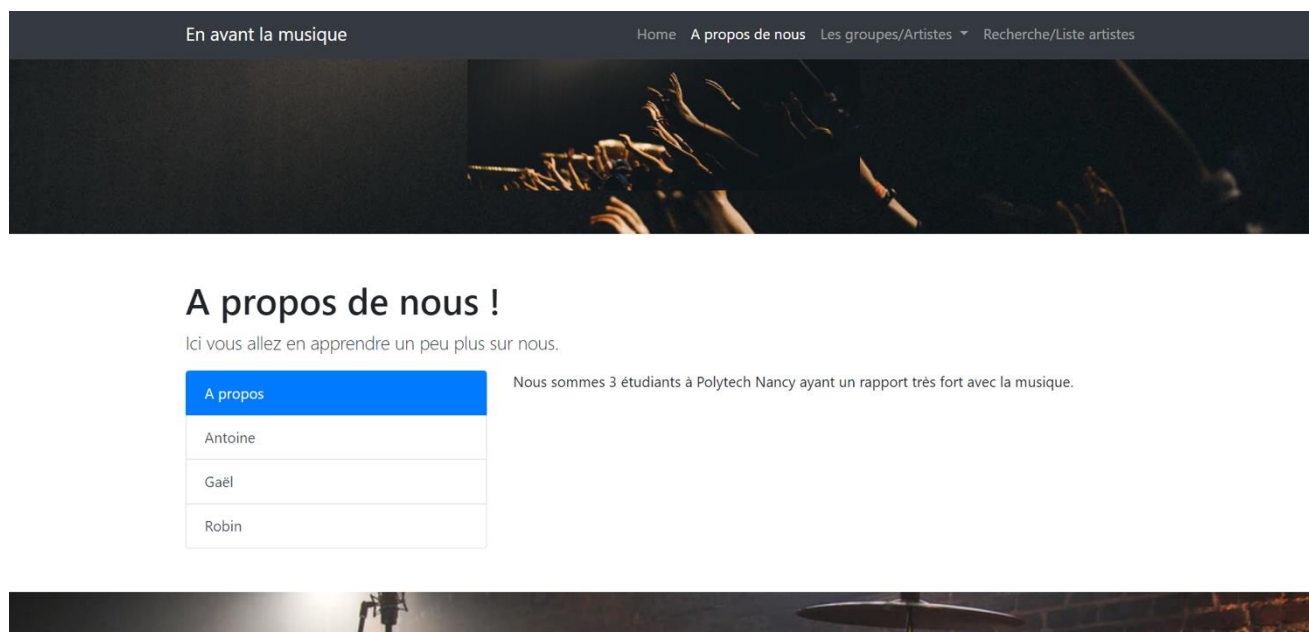


## PARTIE A - PRESENTATION DU PROJET

Tout d'abord, nous avons souhaité développer notre projet autour de la musique car c'est une thématique qui nous tenait à cœur. Nous avons donc décidé de créer une interface web dont le but est de mettre en avant les talents musicaux émergents dans les différents styles de musique. Ainsi, la personne allant sur notre site internet peut découvrir et même écouter des nouveaux artistes en fonction de son style préféré.

Pour accéder à ces fonctionnalités, il suffit de se connecter ou de créer un compte pour ensuite explorer les différentes fonctionnalités. La plateforme se divise en plusieurs pages web qui sont accessibles via plusieurs onglets.

Vous pouvez apercevoir ci-dessous un aperçu du site avec la section « à propos de nous » qui dresse un petit résumé de nos motivations :

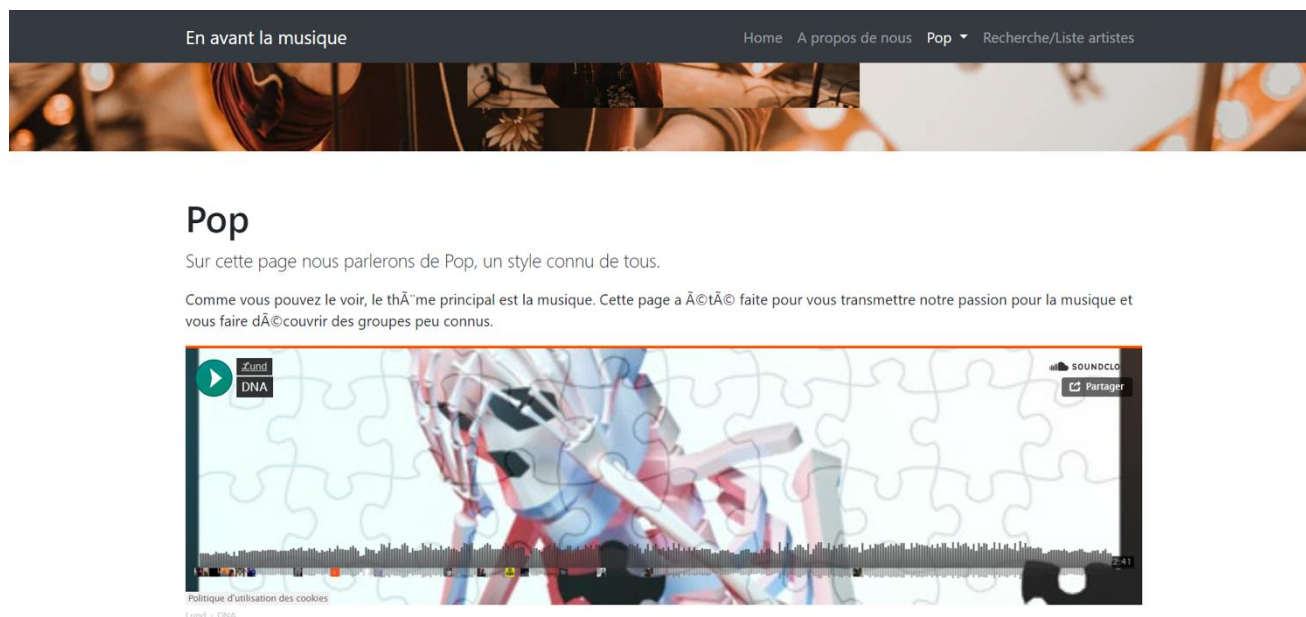




## PARTIE B - INTERFACE GRAPHIQUE

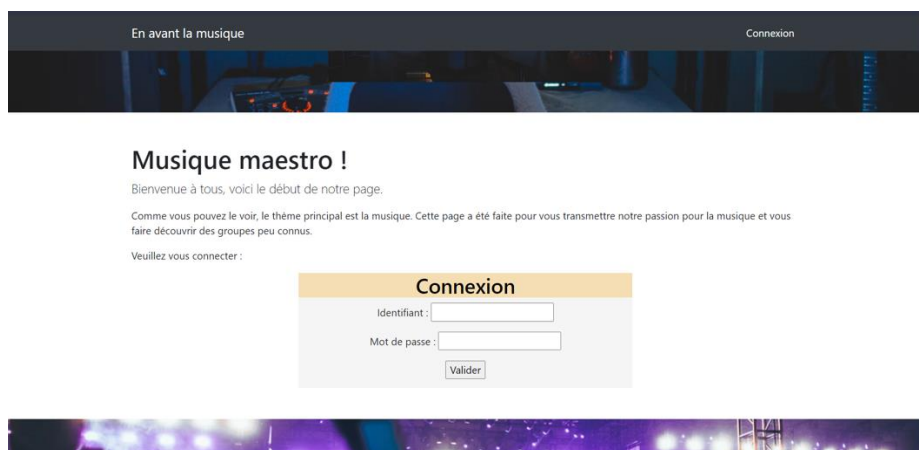
Ensuite, pour la partie visuelle nous avons voulu faire un design sobre et épuré tout en mettant bien en avant le thème musical. Ainsi, nous avons utilisé un Template Bootstrap que nous avons modifié et personnalisé selon nos envies. On peut remarquer que chaque style de musique a un onglet avec un design qui lui correspond. De plus, nous avons souhaité ajouter des players soundcloud qui permettent d'écouter directement la musique des artistes choisis.

Nous avons aussi ajouté des tableaux interactifs comme sur la page « à propos ».



## PARTIE C - ESPACE CLIENT ET CONNEXION

Une des parties les plus importantes de notre interface web et l'espace de connexion ou d'inscription qui permet d'accéder au reste de la plateforme. Cette partie est celle qui nous a pris le plus de temps à développer. Chaque nouveau client arrive sur une page index qui est la page d'authentification où un identifiant et un mot de passe lui est demandé :





Cette fonctionnalité est gérée par un premier servlet qui valide ou non l'accès au reste de la plateforme grâce à la condition True/False qui se trouve dans la méthode connectAccount.

```
protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
    // TODO Auto-generated method stub
    if(request.getParameter("login")!=null) {
        HttpSession session = request.getSession(true);

        // Recuperer les identifiants et les stocker dans la variable de session
        String Pseudonyme = request.getParameter("login");
        session.setAttribute(Pseudonyme, "login");
        ConnexionAccount connexionAccount= new ConnexionAccount();

        //Utilisation et redirection des identifiants dans l'autre Servlet
        if(connexionAccount.connectAccount(request)) {
            getServletContext().getRequestDispatcher("/accueil.jsp").forward(request, response);
        }else{
            getServletContext().getRequestDispatcher("/index.jsp").forward(request, response);
        }
    }
}
```

Ainsi, nous récupérons les informations rentrées par le client que nous allons stocker dans 2 variables id et pass. Nous allons crypter la variable pass pour qu'elle puisse être similaire au mot de passe crypté associé au nom de compte rentré. Ces deux variables vont être comparées à la base de données stockant tous les comptes des clients ainsi que leur mot de passe crypté. Ainsi, tant que la variable resultSet.next() contient quelque chose, la boucle While vérifie les informations rentrées par le client. On vérifie si un identifiant est similaire à la variable id, s'il est similaire on compare ensuite le password et le pass. Si les deux sont identiques alors on peut retourner true. Sinon on retourne false.

On voit sur le screen au-dessus que si un true est retourné, alors on a accès à la page accueil.jsp qui débloque ainsi l'accès à toutes les autres pages. Sinon nous restons sur la page index.jsp où nous devons de nouveau nous identifier.

Sur cette page nous pouvons aussi Supprimer notre compte si nous le voulons, en comparant de nouveau le mot de passe et l'identifiant.

```
public boolean connectAccount (HttpServletRequest request) throws NoSuchAlgorithmException {
    ResultSet resultSet = null;
    this.seConnector();
    String id = request.getParameter("login");
    String plaintext = request.getParameter("pass");
    MessageDigest m = MessageDigest.getInstance("MD5");
    m.reset();
    m.update(plaintext.getBytes());
    byte[] digest = m.digest();
    BigInteger bigInt = new BigInteger(1,digest);
    String hashtext = bigInt.toString(16);
    while(hashtext.length() < 32 ){
        hashtext = "0"+hashtext;
    }
    try {
        Statement statement =connection.createStatement();
        resultSet = statement.executeQuery("SELECT * FROM `password`");
        while(resultSet.next()) {
            if(resultSet.getString("identifiant").equals(id)) {
                if(resultSet.getString("password").equals(hashtext)) {
                    return true;
                }
            }
        }
    }
}
```



Un client est défini par une classe Account ci-dessous.

```
1 package fr.polytech.connexion;
2
3 public class Account {
4     private String identifiant;
5     private String password;
6     public Account(String identifiant, String password) {
7         this.identifiant = identifiant;
8         this.password = password;
9     }
10    //@Override
11    //public String toString() {
12    //    return "Etudiant [numero=" + numero + ", nom=" + nom + ", prenom=" + prenom + "];
13    //}
14    public String getIdentifiant() {
15        return identifiant;
16    }
17    public Account() {
18        super();
19        // TODO Auto-generated constructor stub
20    }
21    public void setIdentifiant(String identifiant) {
22        this.identifiant = identifiant;
23    }
24    public String getPassword() {
25        return password;
26    }
27    public void setPassword(String password) {
28        this.password = password;
29    }
30 }
```

Par ailleurs, si un client est nouveau, il a la possibilité de se créer un compte en rentrant simplement un identifiant et un mot de passe qui sera ajouté à la base de données.

## Création de compte

### Creation compte

Identifiant :

Mot de passe :

Valider

### Suppression compte

Identifiant :

Mot de passe :



```
}  
public void createAccount (Account account) throws NoSuchAlgorithmException {  
    this.seConnecter();  
    //faible d'injection SQL  
    try {  
        PreparedStatement preparedStatement;  
        preparedStatement = this.connection.prepareStatement("INSERT INTO `password`(`identifiant`, `password`) VALUES (?,?);");  
  
        String plaintext = account.getPassword();  
  
        MessageDigest m = MessageDigest.getInstance("MD5");  
        m.reset();  
        m.update(plaintext.getBytes());  
        byte[] digest = m.digest();  
        BigInteger bigInt = new BigInteger(1,digest);  
        String hashtext = bigInt.toString(16);  
        while(hashtext.length() < 32 ){  
            hashtext = "0"+hashtext;  
        }  
        preparedStatement.setString(1, account.getIdentifiant());  
        preparedStatement.setString(2, hashtext);  
        preparedStatement.executeUpdate();  
  
    } catch (SQLException e) {  
        // TODO Auto-generated catch block  
        System.out.println(e.getMessage());  
    }  
}
```

## PARTIE D - CRYPTAGE DES MOTS DE PASSE

Nous avons souhaité apporter une sécurité supplémentaire avec le cryptage des informations fournies par les clients du site.

+ Options	
identifiant	password
test	098f6bcd4621d373cade4e832627b4f6
Robin	9df89da5bc1f8636bfb7ff96866aef45

Nous avons ainsi décidé d'utiliser un code qui hache avec la méthode « MD5 » qui reconvertie ensuite le hachage en string. Ce code est ainsi utilisé dans chaque méthode allant dans la base de données pour comparer les mots de passes hachés contenus dans la base de données et les mots de passes hachés contenu dans les variables pour les comparer entre eux.



```
try {
    PreparedStatement preparedStatement;
    preparedStatement = this.connection.prepareStatement("INSERT INTO `password` (`identifiant`, `password`) VALUES (?,?);");

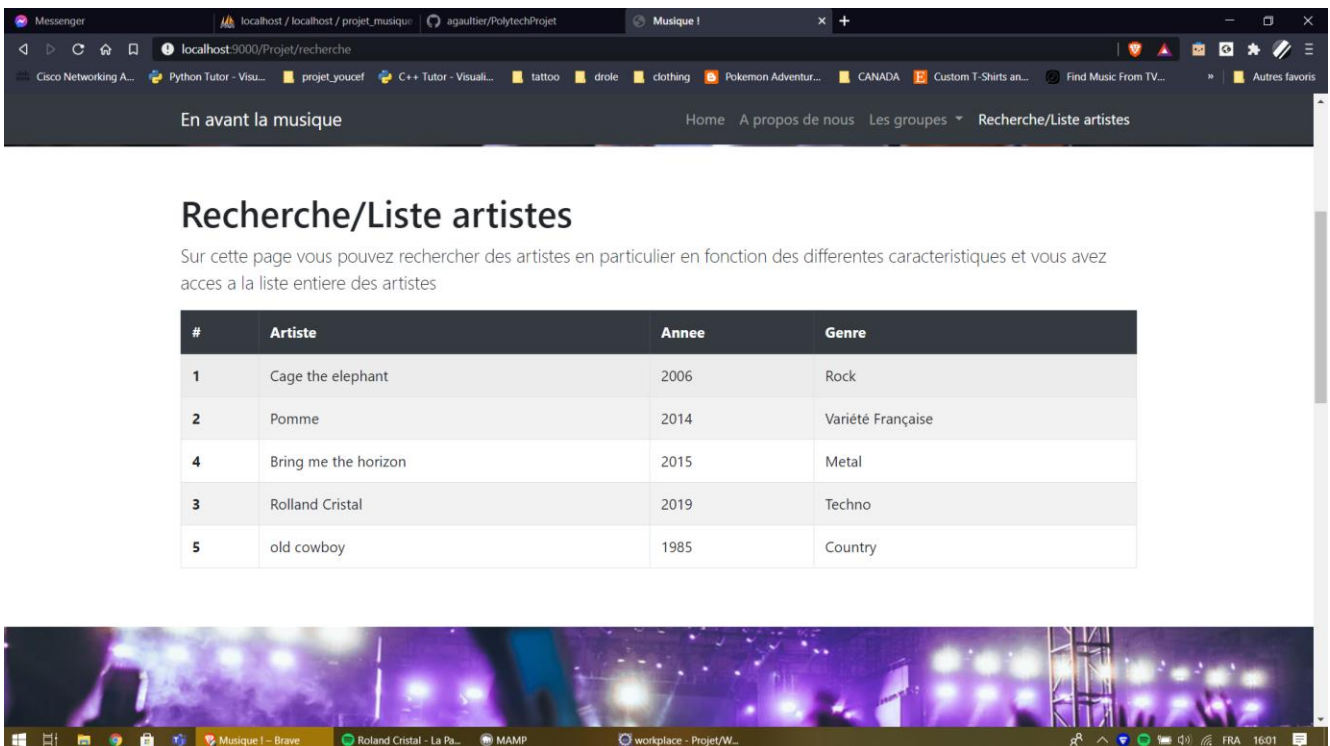
    String plaintext = account.getPassword();

    MessageDigest m = MessageDigest.getInstance("MD5");
    m.reset();
    m.update(plaintext.getBytes());
    byte[] digest = m.digest();
    BigInteger bigInt = new BigInteger(1, digest);
    String hashtext = bigInt.toString(16);
    while (hashtext.length() < 32) {
        hashtext = "0" + hashtext;
    }
    preparedStatement.setString(1, account.getIdentifiant());
    preparedStatement.setString(2, hashtext);
    preparedStatement.executeUpdate();

} catch (SQLException e) {
    // TODO Auto-generated catch block
    System.out.println(e.getMessage());
}
```

## PARTIE E - BASE DE DONNEE MUSICALE

Une des principales fonctionnalités de notre interface est la bibliothèque musicale qui a été créée à l'aide d'une base de données MySQL et d'un tableau bootstrap. Cette base de données peut être modifiée par les utilisateurs. En effet, ils ont la possibilité d'ajouter, de modifier ou de supprimer un artiste.



Recherche/Liste artistes

Sur cette page vous pouvez rechercher des artistes en particulier en fonction des différentes caractéristiques et vous avez accès à la liste entière des artistes

#	Artiste	Annee	Genre
1	Cage the elephant	2006	Rock
2	Pomme	2014	Variété Française
4	Bring me the horizon	2015	Metal
3	Roland Cristal	2019	Techno
5	old cowboy	1985	Country





## Ajoutez/Modifiez/Supprimez vos groupes preferes

### Ajout d'un groupe

Genre :

Artiste :

Annee :

Identifiant :

Valider

### Modification d'un groupe

Genre :

Artiste :

Annee :

Identifiant :

Valider

### Suppression d'un groupe

Identifiant :

Valider

Voici la méthode post du servlet Controller\_Musique qui nous a permis de récupérer ces différentes informations.

On peut voir les méthodes qui permettent de modifier, ajouter ou modifier, en fonction des paramètres rentrés dans les différents champs.

```
protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
    // TODO Auto-generated method stub
    //on définit un objet de la classe métier ... on fait appel a la méthode ajouterUnEtudiant

    if (request.getParameter("genre")!=null){
        Musique musique = new Musique();
        musique.setAnnee(Integer.parseInt(request.getParameter("annee")));
        musique.setArtiste(request.getParameter("artiste"));
        musique.setGenre(request.getParameter("genre"));
        musique.setIdentifiant(Integer.parseInt(request.getParameter("id")));

        Data_Musique musiquesListe = new Data_Musique();
        musiquesListe.ajouterUneMusique(musique);
    }
    if (request.getParameter("idSuppr")!=null) {
        Musique musique = new Musique();
        musique.setIdentifiant(Integer.parseInt(request.getParameter("idSuppr")));

        Data_Musique musiquesListe = new Data_Musique();
        musiquesListe.supprimerUneMusique(musique);
    }

    if (request.getParameter("artisteModif")!=null) {
        Musique musique = new Musique();
        musique.setAnnee(Integer.parseInt(request.getParameter("anneeModif")));
        musique.setArtiste(request.getParameter("artisteModif"));
        musique.setGenre(request.getParameter("genreModif"));
        musique.setIdentifiant(Integer.parseInt(request.getParameter("idModif")));
    }
}
```



Voici la méthode dans la classe jsp qui permet d'afficher le tableau de musiques :

```
<!-- Content section -->
<section class="py-5">
  <div class="container">
    <h1>Recherche/Liste artistes</h1>
    <p class="lead">Sur cette page vous pouvez rechercher des artistes en particulier en fonction des
    différentes caractéristiques et vous avez accès à la liste entière des artistes</p>

    <table class="table table-striped table-bordered table-hover">
      <thead class="thead-dark">
        <tr>
          <th scope="col">#</th>
          <th scope="col">Artiste</th>
          <th scope="col">Année</th>
          <th scope="col">Genre</th>
        </tr>
      </thead>
      <tbody>
        <tr>
          <th scope="row">${musique.identifiant}</th>
          <td>${musique.artiste}</td>
          <td>${musique.annee}</td>
          <td>${musique.genre}</td>
        </tr>
      </tbody>
    </table>
  </div>
</section>
```

Voici le code de la page jsp qui permet d'afficher le formulaire pour modifier ou ajouter des musiques :

```
5 <!-- Content section -->
6 <section class="py-5">
7   <div class="container">
8     <h1>Ajoutez/Modifiez/Supprimez vos groupes préférés</h1>
9
10    <form method="post" name="Form" action="recherche"
11      style="width:50%;margin:auto;background-color:whitesmoke;padding-bottom:15px;">
12
13      <h2 style="text-align:center;color:black;background-color:wheat;">Ajout d'un groupe</h2>
14      <p style="text-align:center;">Genre : <input type="text" name="genre" /></p>
15      <p style="text-align:center;">Artiste : <input type="text" name="artiste" /></p>
16      <p style="text-align:center;">Année : <input type="text" name="annee" /></p>
17      <p style="text-align:center;">Identifiant : <input type="text" name="id" /></p>
18
19      <p style="text-align:center;width:50%;margin:auto;"><input type="submit" name="Valider" value="Valider"/></p>
20    </form>
21
22    <form method="post" name="Form" action="recherche"
23      style="width:50%;margin:auto;background-color:whitesmoke;padding-bottom:15px;">
24
25      <h2 style="text-align:center;color:black;background-color:wheat;">Modification d'un groupe</h2>
26      <p style="text-align:center;">Genre : <input type="text" name="genreModif" /></p>
27      <p style="text-align:center;">Artiste : <input type="text" name="artisteModif" /></p>
28      <p style="text-align:center;">Année : <input type="text" name="anneeModif" /></p>
29      <p style="text-align:center;">Identifiant : <input type="text" name="idModif" /></p>
30
31      <p style="text-align:center;width:50%;margin:auto;"><input type="submit" name="Valider" value="Valider"/></p>
32    </form>
```



Les méthodes ci-dessous sont celles qui gèrent la connexion à notre base de données mySql et permettent d'afficher toutes les musiques:

```
public ArrayList<Musique> afficherToutesLesMusiques(){  
    ArrayList<Musique> resultat = new ArrayList<Musique>();  
    //chargement du driver my sql  
    try {  
        Class.forName("com.mysql.cj.jdbc.Driver"); //exception surveillée ...  
    } catch (ClassNotFoundException e) {  
        // TODO Auto-generated catch block  
        System.out.println("le serveur n'est pas chargé");  
    }  
  
    //se connecter a la base de donnée  
    this.seConnecter();  
    Connection connection = null;  
    Statement statement=null;  
    ResultSet resultSet = null;  
  
    try {  
        connection = DriverManager.getConnection("jdbc:mysql://localhost/projet_musique?useUnicode=true&useJDBCCompliantTimezoneShift=t  
        statement = connection.createStatement();  
        //executer une requête et récupérer le contenu dans l'objet resultSet  
        resultSet = statement.executeQuery("SELECT * FROM `musiques`");  
  
        //parcourir resultSet pour récupérer les données  
        while(resultSet.next()) {  
            int Identifiant = resultSet.getInt("Identifiant");  
            String Genre = resultSet.getString("Genre");  
            String Artiste = resultSet.getString("Artiste");  
            int Annee = resultSet.getInt("Annee");  
            resultat.add(new Musique(Genre, Artiste, Annee, Identifiant));  
        }  
    }  
}
```

Ici nous avons les méthodes qui permettent d'ajouter ou de modifier une musique :

```
}  
public void ajouterUneMusique (Musique musique) {  
    this.seConnecter();  
    //faible d'injection SQL  
    try {  
        PreparedStatement preparedStatement =this.connection.prepareStatement("INSERT INTO `musiques` (`Genre`, `Artiste`, `Annee`, "  
        + "`Identifiant`) VALUES (?, ?, ?, ?);");  
        preparedStatement.setString(1, musique.getGenre());  
        preparedStatement.setString(2, musique.getArtiste());  
        preparedStatement.setInt(3, musique.getAnnee());  
        preparedStatement.setInt(4, musique.getIdentifiant());  
        //mettre a jour et executer la requête  
        preparedStatement.executeUpdate();  
    } catch (SQLException e) {  
        // TODO Auto-generated catch block  
        System.out.println(e.getMessage());  
    }  
}  
  
public void modifierUneMusique (Musique musique) {  
    this.seConnecter();  
    try {  
        PreparedStatement preparedStatement = this.connection.prepareStatement("UPDATE `musiques` SET `Genre`=?,`Artiste`=?, "  
        + "`Annee`=?, `WHERE Identifiant=?");  
        preparedStatement.setString(1, musique.getGenre());  
        preparedStatement.setString(2, musique.getArtiste());  
        preparedStatement.setInt(3, musique.getAnnee());  
        preparedStatement.setInt(4, musique.getIdentifiant());  
  
        preparedStatement.executeUpdate();  
    }  
}
```



## PARTIE F - CONCLUSION

Pour conclure, nous avons réussi à modéliser et à mettre en place la majorité de nos idées concernant cette interface web. En effet, nous avons mis en place une plateforme avec une belle mise en page qui permet de gérer des connexions et des inscriptions clients mais aussi la gestion complète d'une bibliothèque musicale par des utilisateurs qui souhaitent faire découvrir des nouveaux talents musicaux. Nous avons apprécié travailler sur ce projet car la thématique nous tenait à cœur et cela a permis de mettre en application ce que nous avons vu en cours dans le cadre d'une application réelle.