
# École Supérieure de Technologie - Safi

---

# Rapport de TP1

---

## Application de Gestion des Employés avec Swing et Modèle MVC

**Filière : Génie Informatique**
**Réalisé par : Omar Akalay**
**Année Universitaire : 2024/2025**

# Contents

**Introduction** **2**

**1 Couche Modèle** **3**
1.1 Définition des Classes et Énumérations . . . . . 3
1.2 Exemple de Code pour la Classe Employee . . . . . 3

**2 Couche DAO** **5**
2.1 Connexion à la Base de Données . . . . . 5
2.1.1 Création des Tables . . . . . 5
2.1.2 Classe DBConnection . . . . . 5

**3 Interface Graphique** **6**
3.1 Conception de la Vue . . . . . 6

**4 Contrôleur** **7**
4.1 Gestion des Événements . . . . . 7

**Conclusion** **9**

# Introduction

L'objectif principal de ce projet est de développer une application de gestion des employés pour l'entreprise SEA. Cette application, basée sur l'architecture MVC (Modèle-Vue-Contrôleur), permettra une gestion centralisée et efficace des données des employés, des rôles et des postes. Ce rapport présente les étapes de conception et de développement du projet, de la création du modèle à l'implémentation de l'interface utilisateur et du contrôleur.

# Chapter 1

# Couche Modèle

## 1.1 Définition des Classes et Énumérations

Dans cette section, nous définissons les classes et énumérations nécessaires pour représenter les données de l'application :

- Classe `Employee` : représente les employés avec des attributs tels que nom, prénom, rôle, et poste.
- Énumérations `Poste` et `Role` : définissent les valeurs possibles pour les postes et les rôles.

## 1.2 Exemple de Code pour la Classe Employee

```
public class Employee {
    private int id;
    private String name;
    private String role;
    private String poste;

    public Employee(int id, String name, String role, String poste) {
        this.id = id;
        this.name = name;
        this.role = role;
        this.poste = poste;
    }

    // Getters et setters
    public int getId() { return id; }
    public void setId(int id) { this.id = id; }
    public String getName() { return name; }
    public void setName(String name) { this.name = name; }
    public String getRole() { return role; }
    public void setRole(String role) { this.role = role; }
```

```
    public String getPoste() { return poste; }
    public void setPoste(String poste) { this.poste = poste; }
}
```

# Chapter 2

# Couche DAO

## 2.1 Connexion à la Base de Données

### 2.1.1 Création des Tables

Les tables `Employee`, `Poste`, et `Role` sont définies dans une base de données MySQL. Voici un exemple de script SQL :

```
CREATE TABLE Employee (
    id INT AUTO_INCREMENT PRIMARY KEY,
    name VARCHAR(100),
    role VARCHAR(50),
    poste VARCHAR(50)
);
CREATE TABLE Poste (
    id INT AUTO_INCREMENT PRIMARY KEY,
    name VARCHAR(50)
);
CREATE TABLE Role (
    id INT AUTO_INCREMENT PRIMARY KEY,
    name VARCHAR(50)
);
```

### 2.1.2 Classe DBConnection

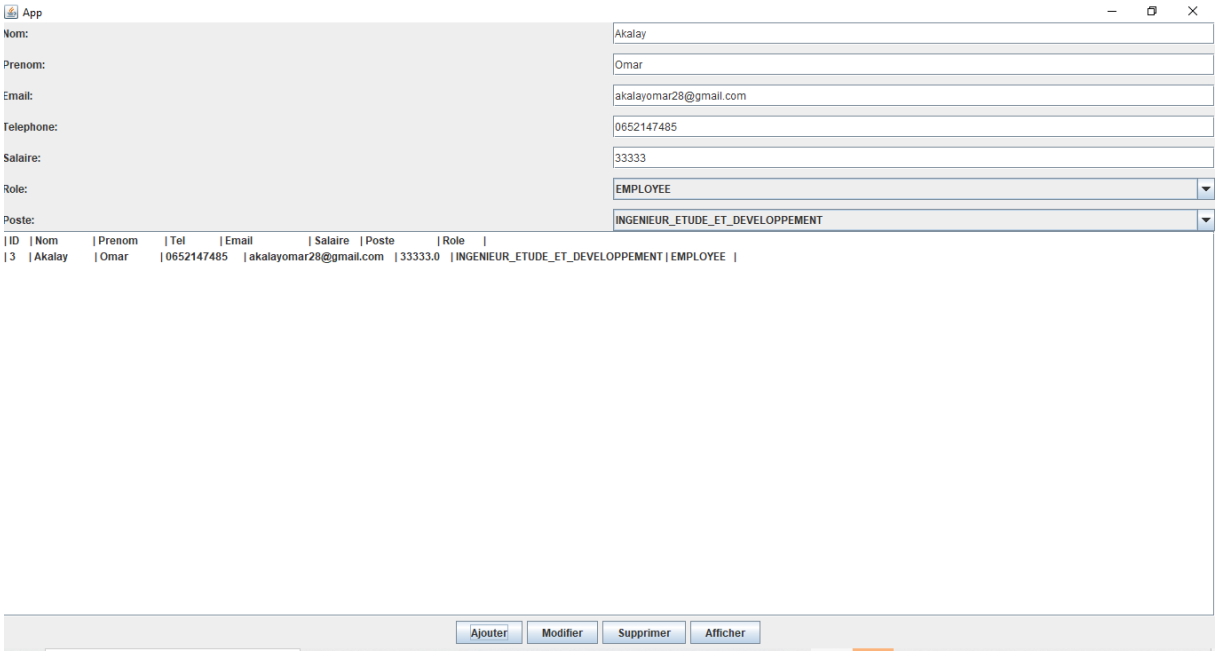
```
public class DBConnection {
    private static final String URL = "jdbc:mysql://localhost:3306/sea";
    private static final String USER = "root";
    private static final String PASSWORD = "password";

    public static Connection getConnection() throws SQLException {
        return DriverManager.getConnection(URL, USER, PASSWORD);
    }
}
```

# Chapter 3

# Interface Graphique

## 3.1 Conception de la Vue

L'interface utilisateur est réalisée avec Swing et organisée avec des `JPanel` et `JTable`.



# Chapter 4

# Contrôleur

## 4.1 Gestion des Événements

La classe `EmployeeController` gère les interactions utilisateur et la coordination avec les modèles. article [utf8]inputenc amsmath

### Ajouter un Employé

La fonction **Ajouter** permet d'enregistrer de nouvelles informations dans la base de données. Lorsqu'un utilisateur souhaite ajouter un employé, il doit remplir un formulaire contenant les champs requis tels que le nom, le rôle et le poste. Ces données sont ensuite validées pour éviter les erreurs, comme des champs vides ou des données non conformes. Une fois validées, les informations sont sauvegardées dans la table `Employee` de la base de données, avec un identifiant unique généré automatiquement. **Exemple de scénario :** Lorsqu'un nouvel employé rejoint l'entreprise, l'utilisateur utilise cette option pour intégrer ses informations au système.

### Afficher les Employés

La fonction **Afficher** permet de consulter la liste de tous les employés existants. Les données sont présentées dans une interface graphique sous forme de tableau dynamique. Cette interface inclut des fonctionnalités de tri et de recherche pour faciliter l'accès aux informations spécifiques. **Exemple de scénario :** Le gestionnaire souhaite visualiser les employés travaillant dans un département spécifique et utilise la fonction de recherche pour filtrer les résultats.

### Supprimer un Employé

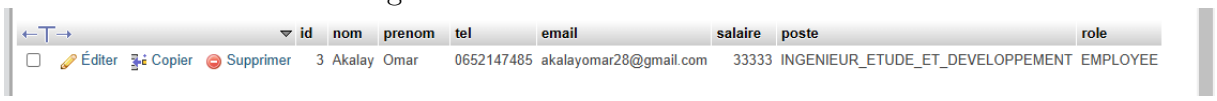
La fonction **Supprimer** permet de retirer un employé de la base de données. L'utilisateur sélectionne un employé dans la liste et confirme l'opération. Une fois supprimées, les informations ne sont plus accessibles, et cette action est irréversible. **Exemple de scénario :** Si un employé quitte l'entreprise, le gestionnaire peut supprimer ses données du système pour éviter toute confusion future.

## Modifier les Informations d'un Employé

La fonction **Modifier** permet de mettre à jour les informations d'un employé existant. Après avoir sélectionné un employé dans la liste, l'utilisateur peut accéder à un formulaire pré-rempli contenant ses données actuelles. Les modifications sont ensuite enregistrées dans la base de données, remplaçant les informations précédentes. **Exemple de scénario :** Un employé change de poste ou de rôle ; le gestionnaire utilise cette option pour actualiser les informations afin qu'elles reflètent le changement.

## Remarque

tous les modifications sont enregistrés dans la base de donnée



| | id | nom | prenom | tel | email | salaire | poste | role |
|---|---|---|---|---|---|---|---|---|
| Éditer Copier Supprimer | 3 | Akalay | Omar | 0652147485 | akalayomar28@gmail.com | 33333 | INGENIEUR_ETUDE_ET_DEVELOPPEMENT | EMPLOYEE |

# Conclusion

Le projet a permis de résoudre les problèmes de gestion des employés de l'entreprise SEA en introduisant une application centralisée et bien structurée. Les futures améliorations pourraient inclure l'ajout de nouvelles fonctionnalités, telles que la gestion des horaires et des performances des employés.