

**UNIVERSITÉ CADI AYAD**

**ECOLE SUPERIEURE DE TECHNOLOGIE-SAFI**

**DUT GÉNIE INFORMATIQUE**

Compte rendu

TP 3 : Système de Gestion des Congés

(Genericite,MVC, DAO, Swing,E/S)

Réalisée par : Enseigné par :

SAISSI Zahra M.EL ABDELLAOUI Said

Année universitaire :2024/2025

Table des matières

[Introduction 2](#_Toc4128)

[Objectifs 2](#_Toc21776)

[Lien GitHub: 2](#_Toc1543)

[Étapes de Création du Projet 3](#_Toc1023)

[1. 1ère étape : Création de l’interface génerique et son implémentation 3](#_Toc26351)

[1.1. Interface DataImportExport: 3](#_Toc14661)

[1.2. Importation des Données Employés depuis un Fichier Texte 3](#_Toc23491)

[1.3. Exportation des Données Employés depuis un Fichier Texte 4](#_Toc18174)

[Code : 4](#_Toc28570)

[2. 2ème étape : Couche Model -Logique metier 5](#_Toc20363)

[Code : 5](#_Toc6233)

[3. 3ème étape : Couche View - Interface graphique 5](#_Toc25146)

[3.1. Les boutons d'importation et d'exportation 5](#_Toc15656)

[Code: 6](#_Toc21085)

[3.2. L'action d'importation et d'exportation 6](#_Toc29103)

[Code: 6](#_Toc17428)

[3.3. Les méthodes d'exportation et d'importation des données (exportData et importData) 7](#_Toc5188)

[Code: 7](#_Toc1098)

[4. 4ème étape : Main - Application principale 8](#_Toc11879)

[Code : 8](#_Toc11702)

[Réalisation 10](#_Toc12865)

[1. Page Login: 10](#_Toc27370)

[2. Authentification: 10](#_Toc17668)

[3. Affichage des employes et conges: 10](#_Toc26023)

[4. Exportation des employes: 11](#_Toc1371)

[5. Exportation des conges: 13](#_Toc13846)

[6. Importation des employes: 14](#_Toc30286)

[-On importe un fichier testemplye\_Importation: 14](#_Toc30876)

[-On clique sur importer: 15](#_Toc14912)

[-On choisi notre fichier: 15](#_Toc10072)

[-on l’ouvre: 15](#_Toc26359)

[7. Importation des conges: 16](#_Toc31890)

[-On suit les memes etapes: 16](#_Toc17103)

[Conclusion 19](#_Toc3489)

# Introduction

Dans le cadre de l'amélioration du système de gestion des congés, l'entreprise SEA a décidé d'intégrer des mécanismes d'entrées/sorties (E/S) afin d'ajouter des fonctionnalités d'importation et d'exportation des données. Ces nouvelles fonctionnalités, basées sur le modèle MVC et l'architecture DAO, permettront de renforcer la flexibilité et l'efficacité de l'application en facilitant la gestion des employés et des congés via des fichiers externes.

# Objectifs

L'objectif principal est d'étendre l'application existante pour inclure des fonctionnalités avancées d'import/export des données. Plus précisément, il s'agit de permettre :

* L'exportation de la liste complète des employés.
* L'importation de nouvelles données concernant les employés.
* L'exportation ciblée des employés bénéficiant de congés.

# Lien GitHub:

**https://github.com/saissizahra/GestionRessourcesHumain.git**

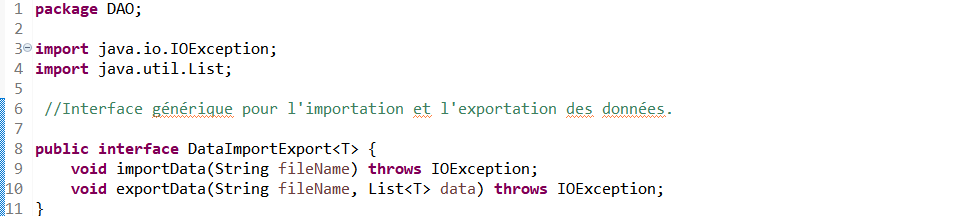
# Étapes de Création du Projet

## **1ère étape : Création de l’interface génerique et son implémentation**

### **Interface DataImportExport:**

Dans cette première étape, nous mettons en place une interface générique DataImportExport qui définit les méthodes nécessaires pour importer et exporter des données. L'objectif est de permettre à l'application de lire des données à partir de fichiers (par exemple, CSV ou TXT) et d'exporter des informations vers ces fichiers, tout en restant flexible pour différents types de données grâce à la généricité.

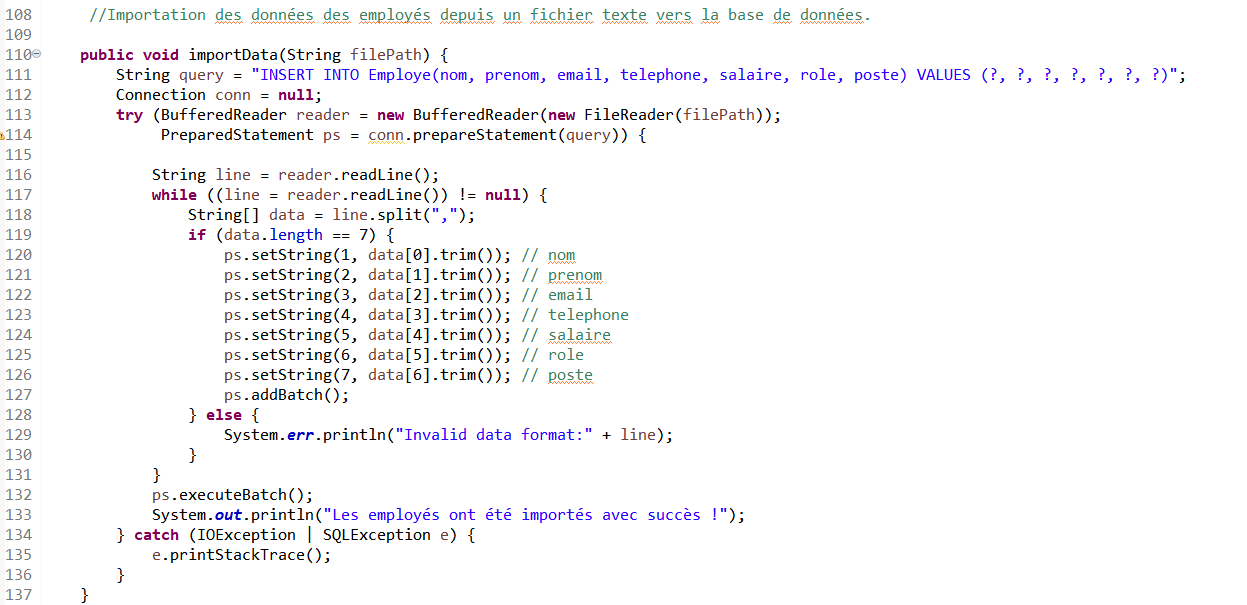
**Code :**



### **Importation des Données Employés depuis un Fichier Texte**

Dans la classe EmployeDAOImpl, cette méthode permet d'importer les données des employés depuis un fichier texte dans la base de données. Le fichier est lu ligne par ligne avec un BufferedReader. Chaque ligne est ensuite divisée en différentes valeurs (nom, prénom, email, etc.) à l'aide de la méthode split. Si le format est valide (7 éléments), ces valeurs sont insérées dans la base de données via une requête INSERT préparée. Après avoir ajouté toutes les lignes valides au lot de la requête, l'insertion est effectuée en une seule opération avec ps.executeBatch(). Si une ligne a un format incorrect, un message d'erreur est affiché.

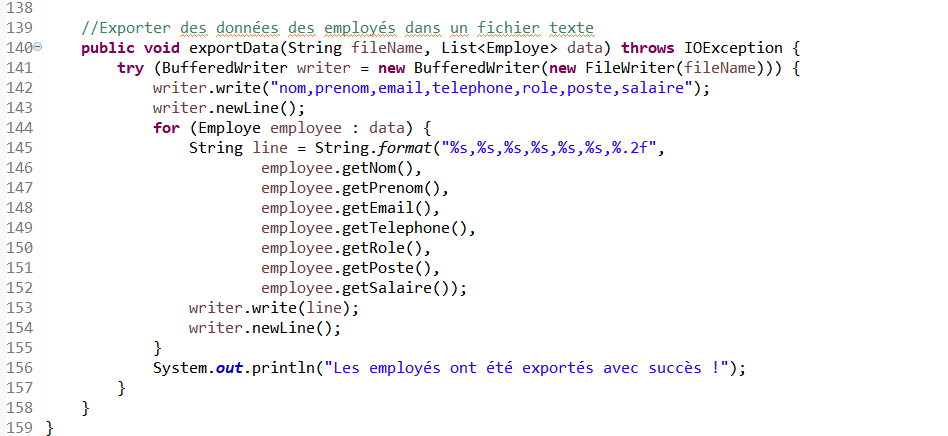
**Code :**



### **Exportation des Données Employés depuis un Fichier Texte**

Dans cette méthode, située également dans la classe EmployeDAOImpl, les données des employés sont exportées vers un fichier texte à l'aide d'un BufferedWriter. Le fichier est créé ou ouvert avec un FileWriter, et la première ligne contient les en-têtes des colonnes (nom, prénom, email, etc.). Ensuite, pour chaque employé dans la liste, une ligne est formatée avec les informations correspondantes (nom, prénom, email, etc.) et écrite dans le fichier. Après chaque ligne, un saut de ligne est ajouté pour séparer les enregistrements. Une fois l'exportation terminée, un message confirme le succès de l'opération.

**Code :**

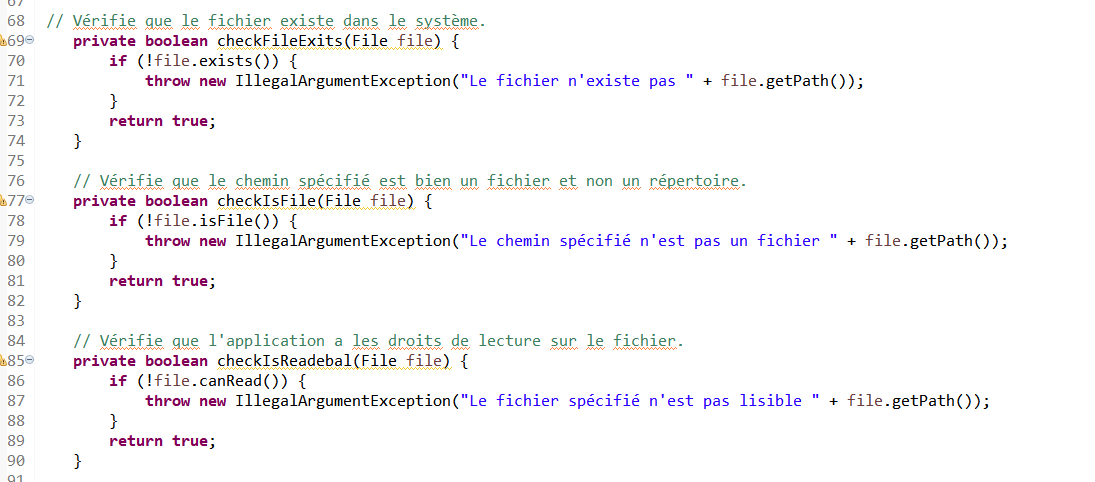


## **2ème étape : Couche Model -Logique metier**

Dans cette étape, située dans la couche **Model**, plusieurs vérifications sont mises en place pour garantir que le fichier spécifié est valide avant d'être utilisé pour l'import ou l'export des données.

1. **Vérification de l'existence du fichier** : La méthode checkFileExits vérifie si le fichier existe bien dans le système. Si le fichier n'existe pas, une exception est levée avec un message d'erreur approprié.
2. **Vérification du type de fichier** : La méthode checkIsFile s'assure que le chemin spécifié pointe bien vers un fichier et non un répertoire. Si ce n'est pas le cas, une exception est levée.
3. **Vérification des droits de lecture** : La méthode checkIsReadebal vérifie que le fichier est lisible par l'application. Si les droits de lecture sont insuffisants, une exception est levée.

**Code :**



1. **3ème étape : Couche View - Interface graphique**
   1. **Les boutons d'importation et d'exportation**

Dans l'interface de gestion des employés, deux boutons ont été ajoutés pour faciliter les opérations d'importation et d'exportation des données. Le bouton "Importer" permet à l'utilisateur de charger un fichier contenant des informations, comme des données sur les employés, et de les ajouter au modèle de la table en affichant les nouvelles informations dans l'interface. De même, le bouton "Exporter" permet de sauvegarder les données actuelles de la table dans un fichier.

**Code:**



* 1. **L'action d'importation et d'exportation**

L'action d'importation est déclenchée lorsque l'utilisateur clique sur le bouton "Importer". Cela ouvre une fenêtre de sélection de fichier, et une fois un fichier choisi, les données sont lues ligne par ligne et ajoutées à la table. Un message de succès informe l'utilisateur de la réussite de l'importation.

L'action d'exportation se lance lorsque l'utilisateur clique sur le bouton "Exporter". Il ouvre une fenêtre pour sélectionner l'emplacement et le nom du fichier, puis les données de la table sont sauvegardées dans le fichier choisi. Un message de confirmation est affiché pour informer l'utilisateur de la réussite de l'exportation.

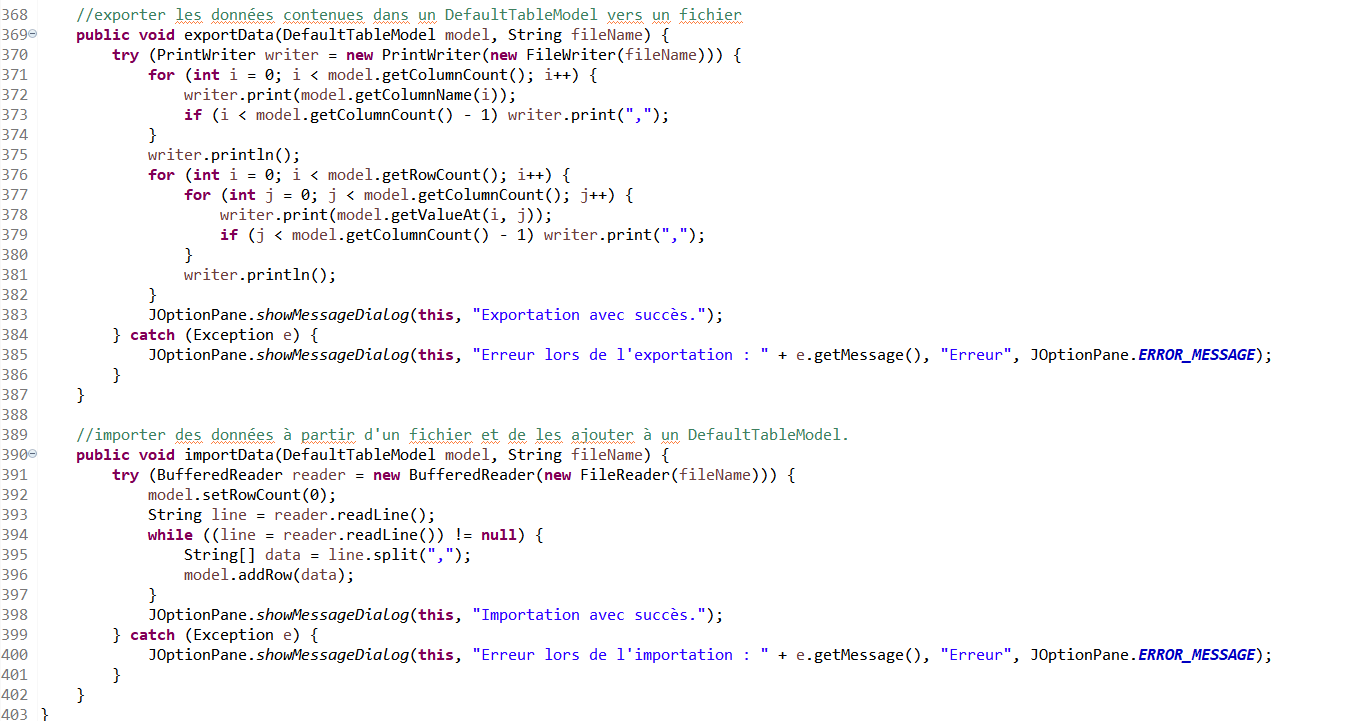
**Code:**



* 1. **Les méthodes d'exportation et d'importation des données (exportData et importData)**

Les méthodes exportData et importData gèrent respectivement l'exportation et l'importation des données dans l'application. La méthode exportData prend en charge l'enregistrement des données actuelles de la table dans un fichier. Elle commence par écrire les noms des colonnes, suivis des valeurs des lignes, séparées par des virgules. Une fois l'exportation terminée, un message de succès s'affiche pour informer l'utilisateur de la réussite de l'opération. En revanche, la méthode importData s'occupe de charger les données depuis un fichier. Elle lit chaque ligne du fichier, extrait les valeurs séparées par des virgules et les ajoute à la table. Les anciennes données sont supprimées avant l'importation pour éviter toute confusion avec les nouvelles données.

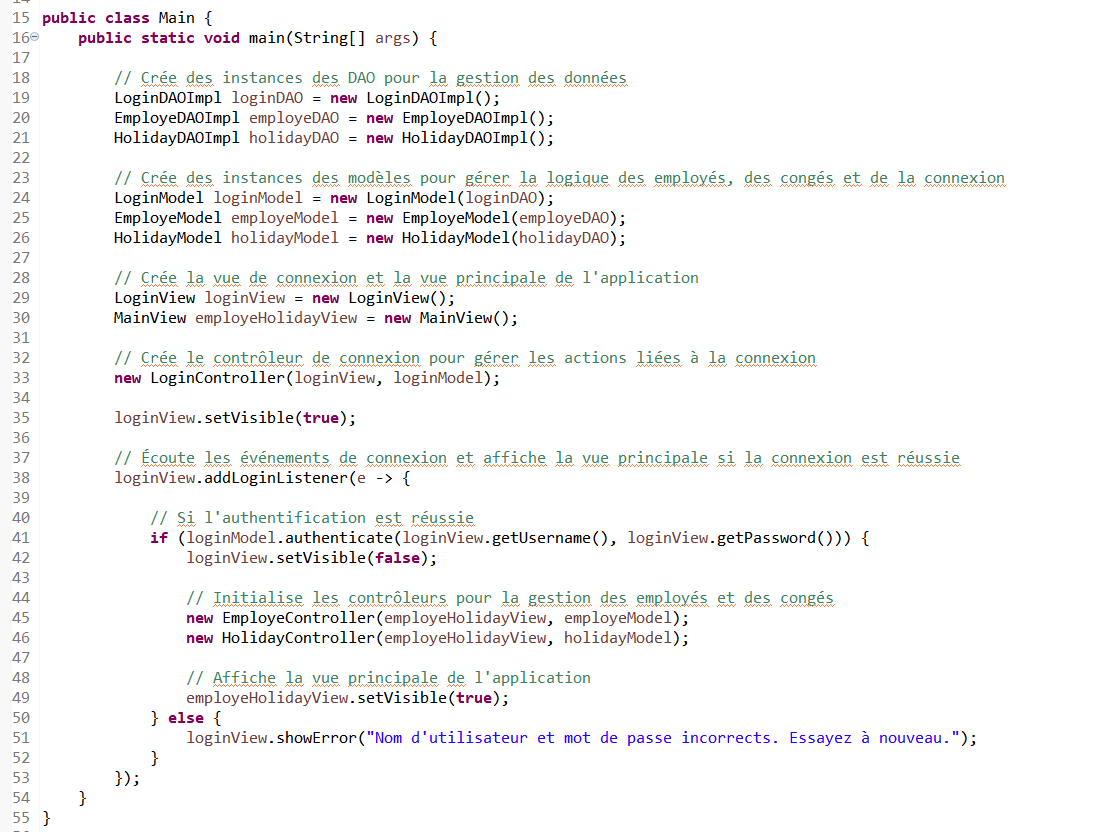
**Code:**



## **4ème étape : Main - Application principale**

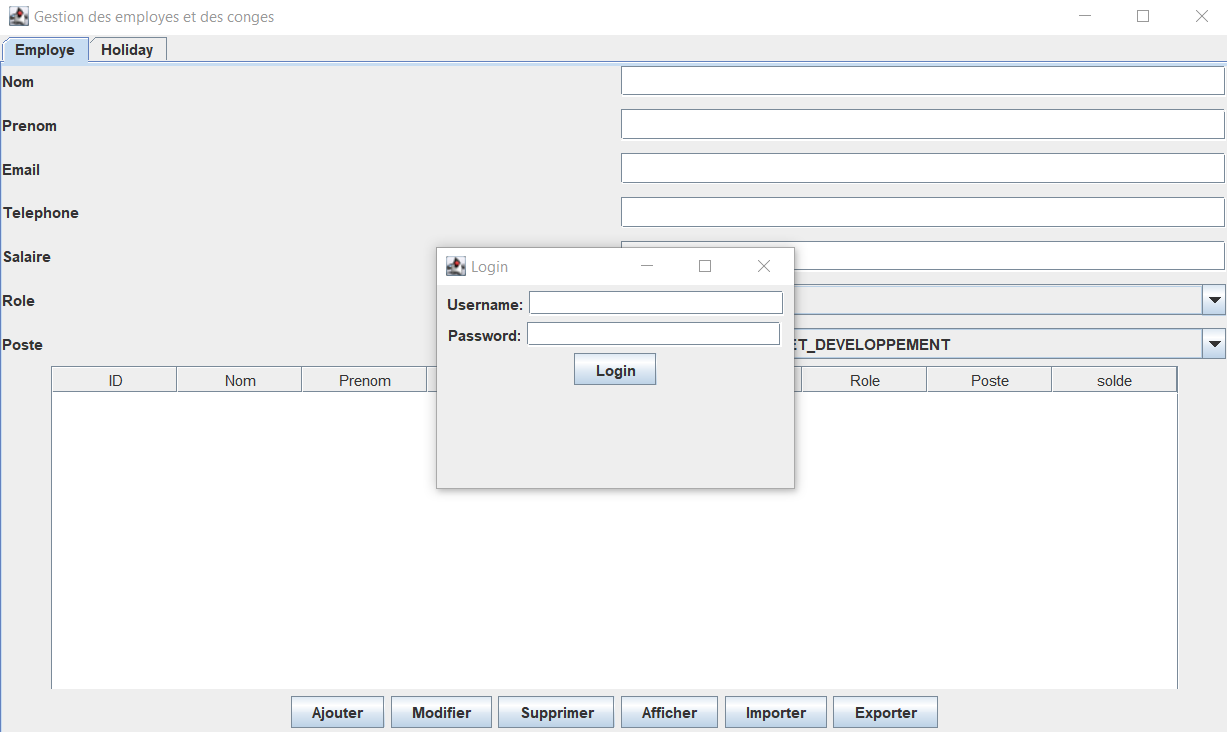
La classe Main initialise l'application en créant les objets nécessaires pour la gestion de la connexion des employés et leurs congés. Il crée des instances des classes de gestion des données (DAO), des modèles qui traitent la logique métier, et des vues pour afficher l'interface utilisateur. Lorsqu'un utilisateur entre ses informations de connexion, le contrôleur vérifie si celles-ci sont correctes. Si la connexion réussit, la vue principale apparaît, permettant de gérer les employés et les congés. En cas d'échec, un message d'erreur est affiché.

**Code :**

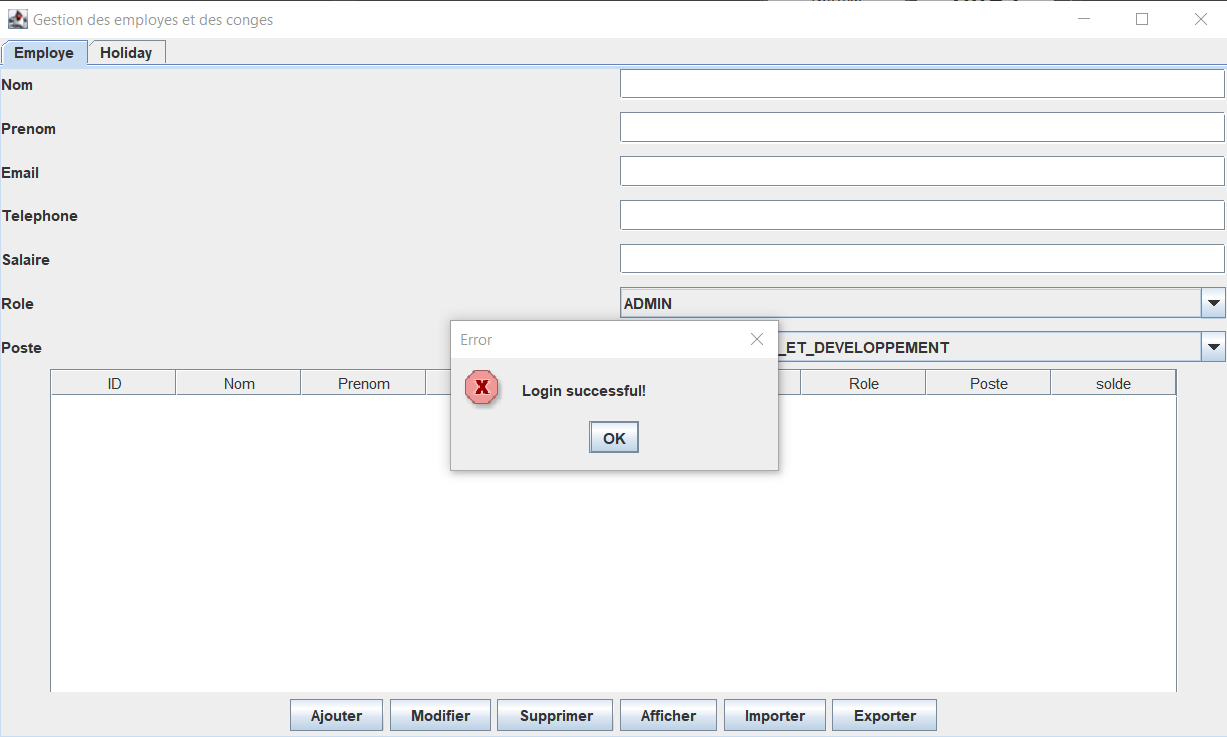


# Réalisation

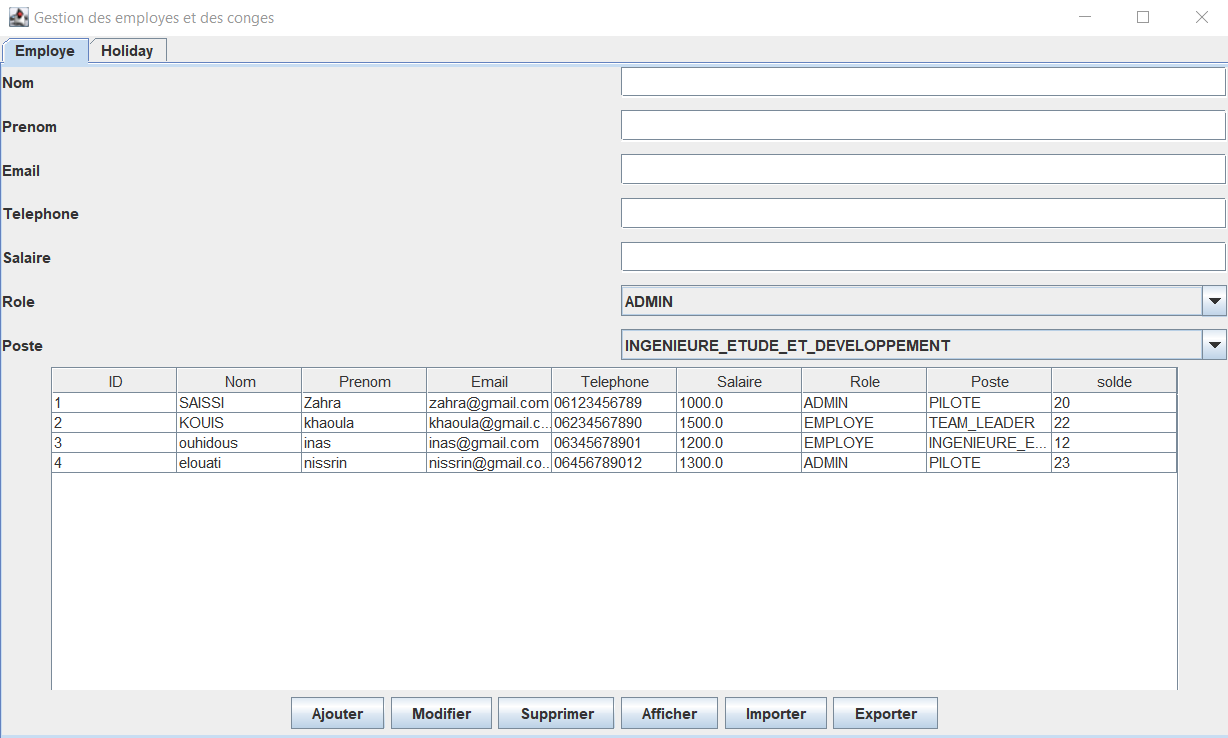
1. **Page Login:**

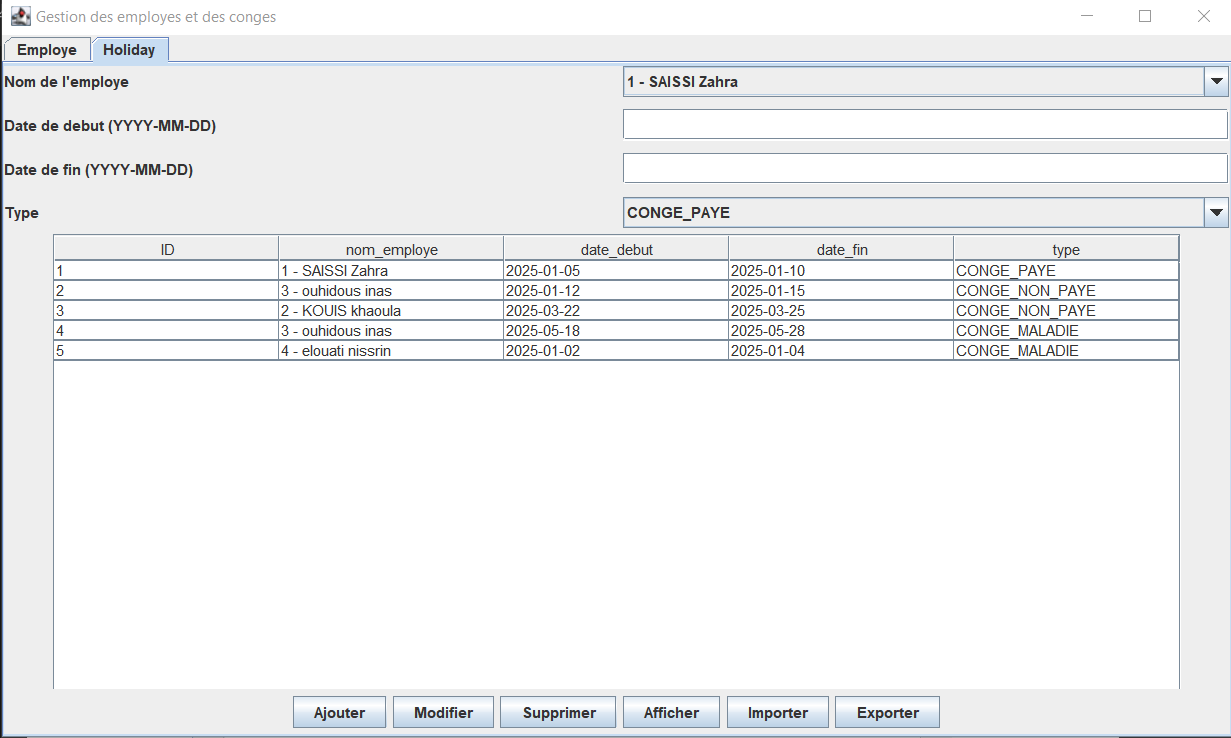


1. **Authentification:**



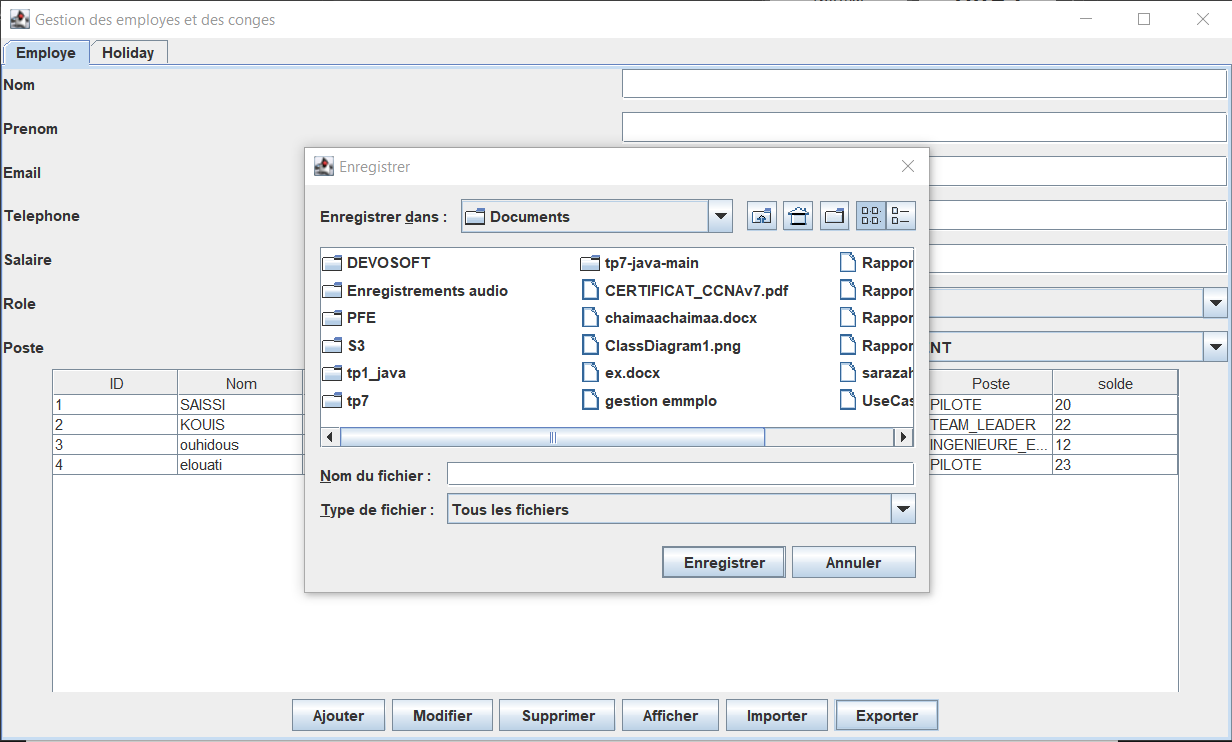
1. **Affichage des employes et conges:**



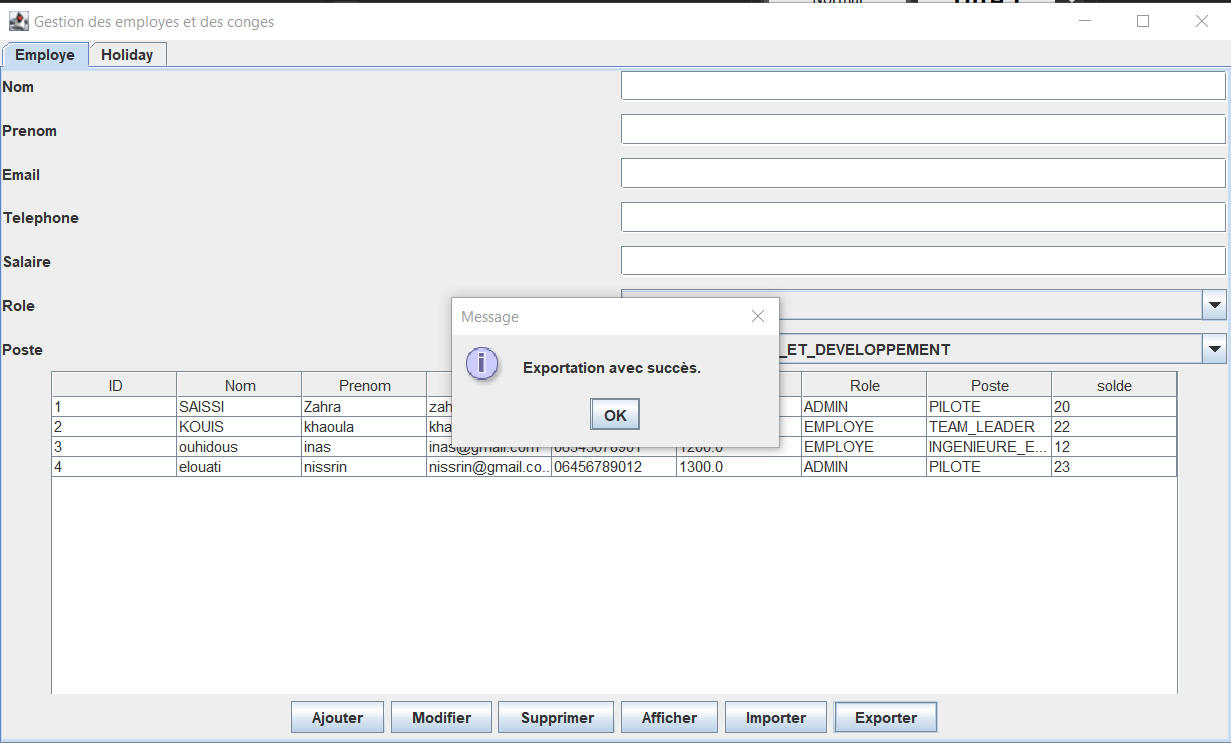


1. **Exportation des employes:**

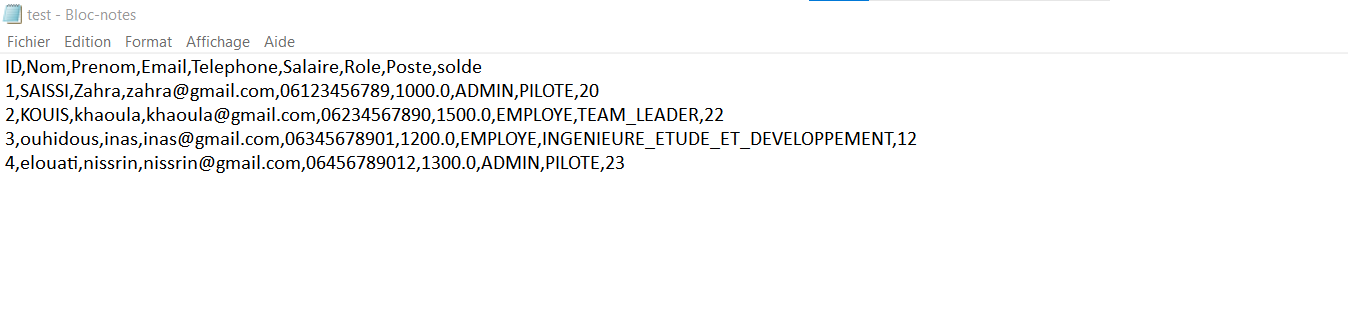
**-** Une fois en clique sur le boutton exporter



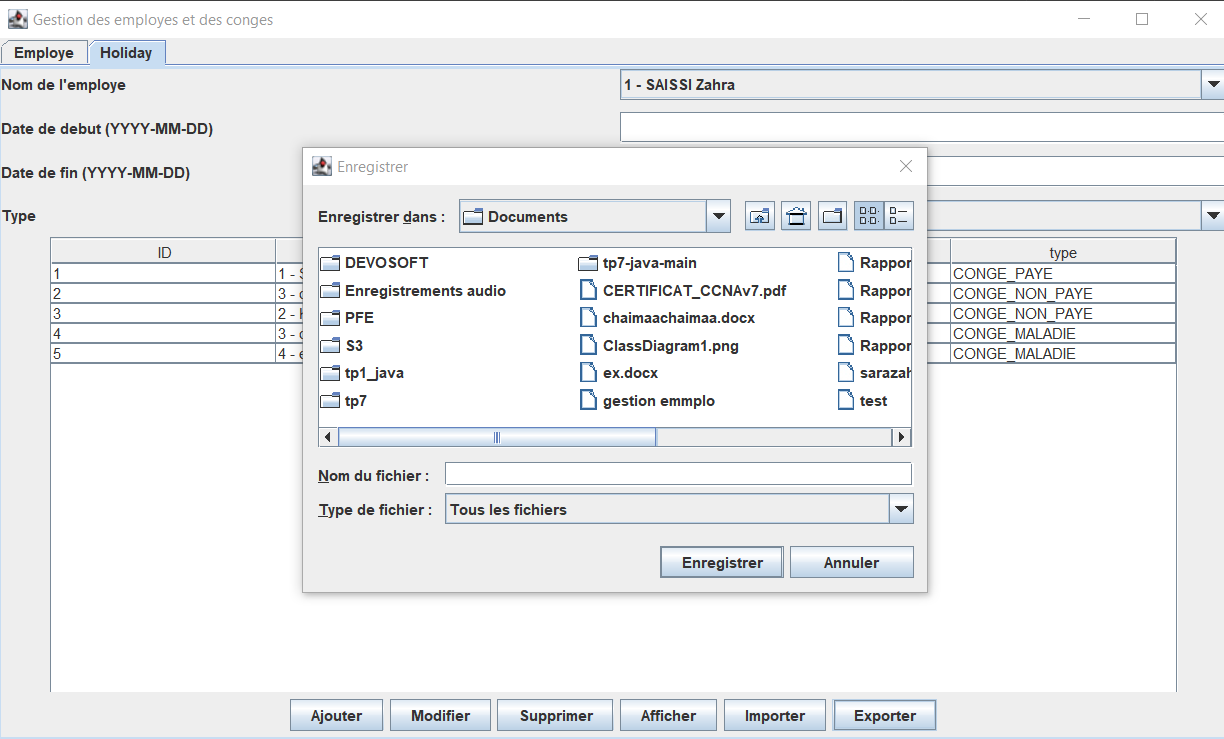
-En l’enregistre sous le nom test:

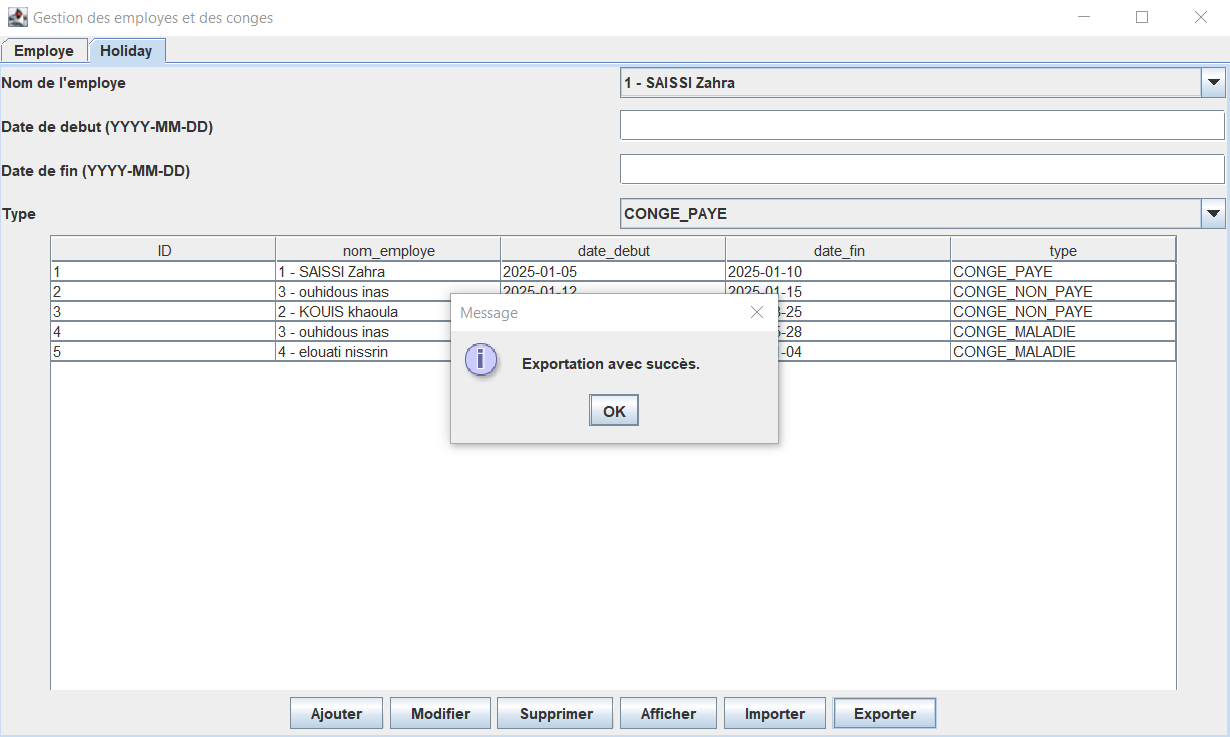


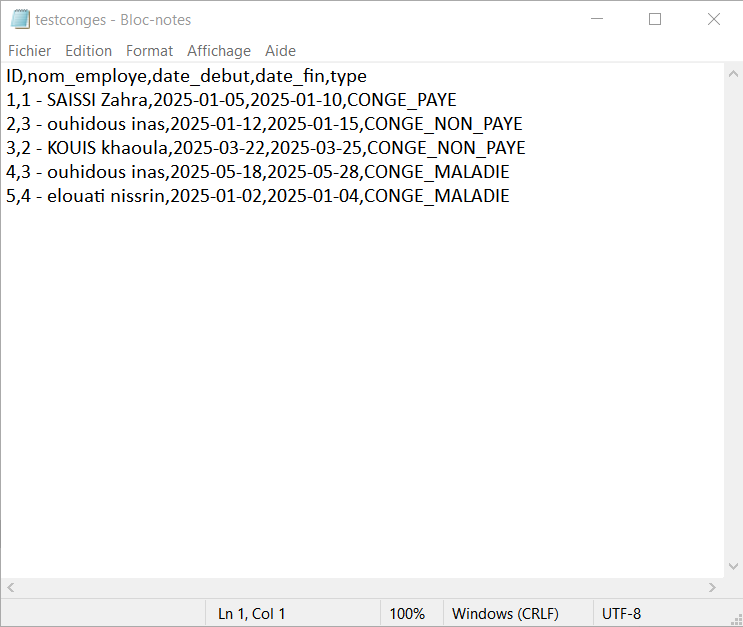
- Le fichier test s’affiche dans le bloc note comme si-dessous:



1. **Exportation des conges:**

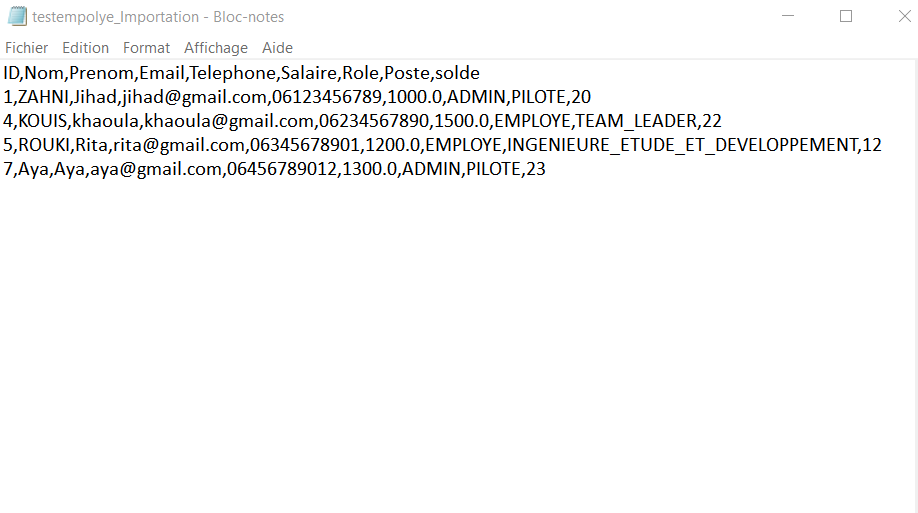




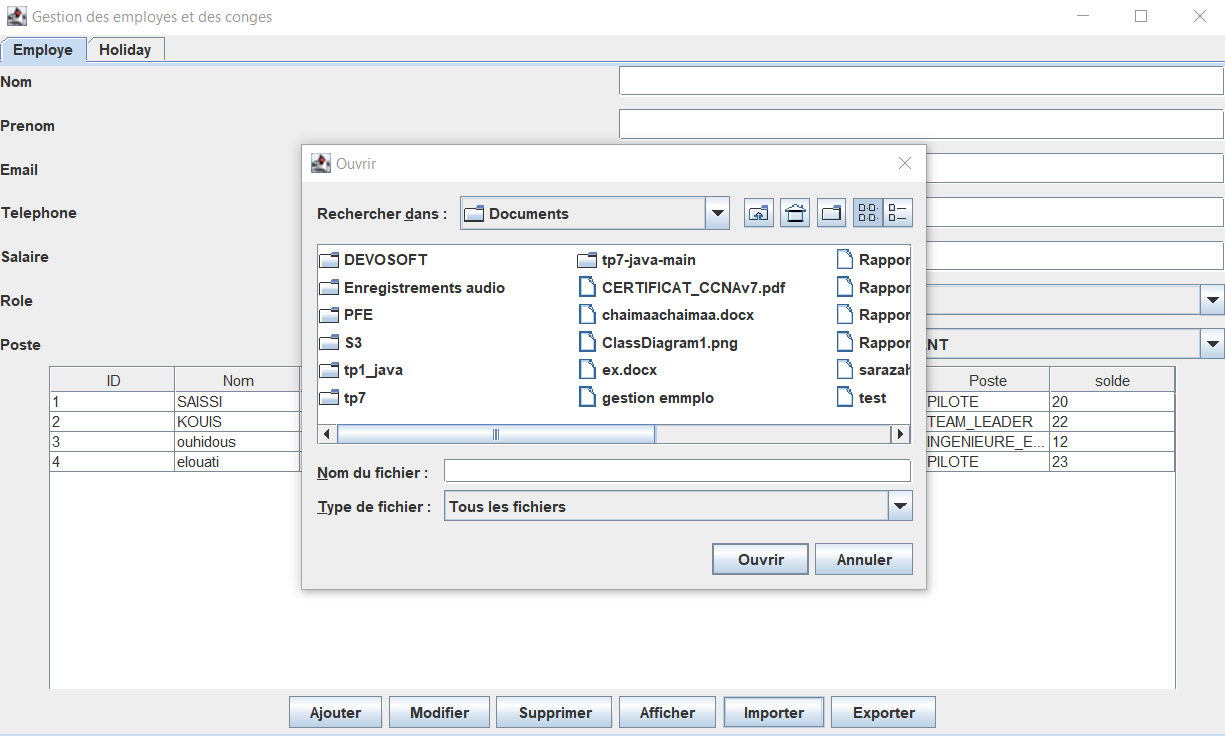


1. **Importation des employes:**

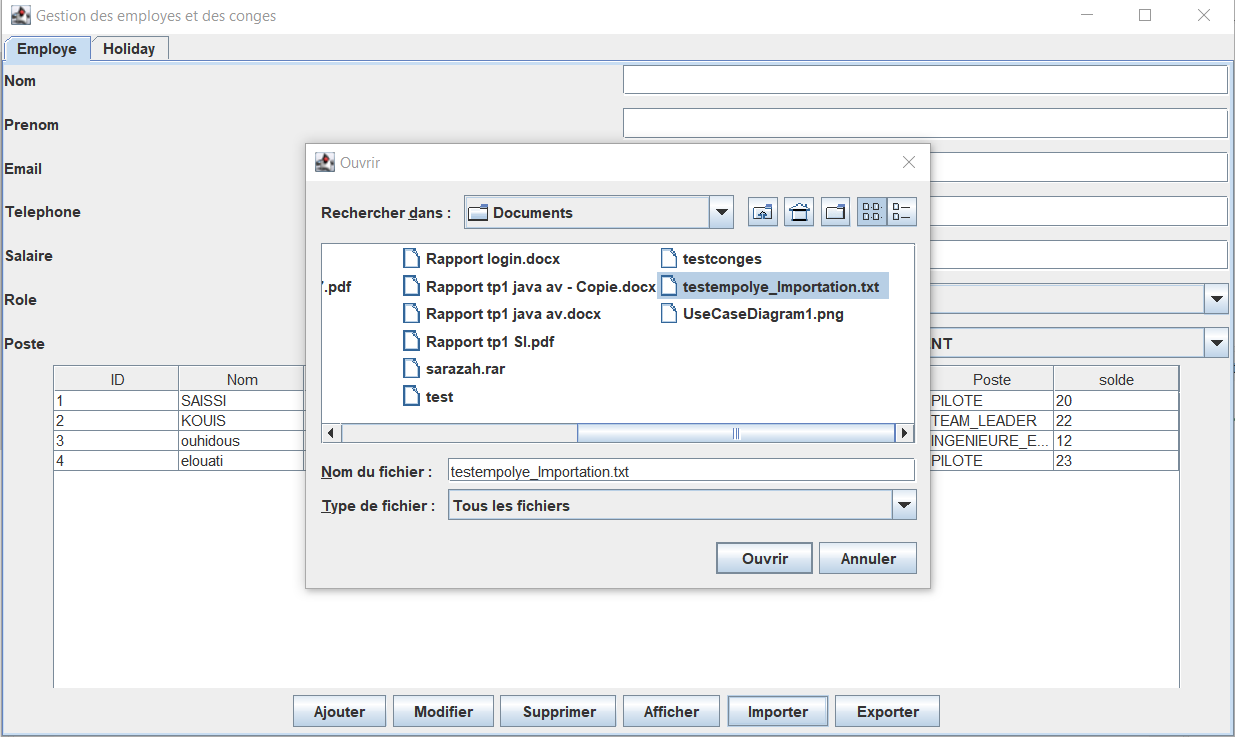
**-**On importe un fichier testemplye\_Importation:



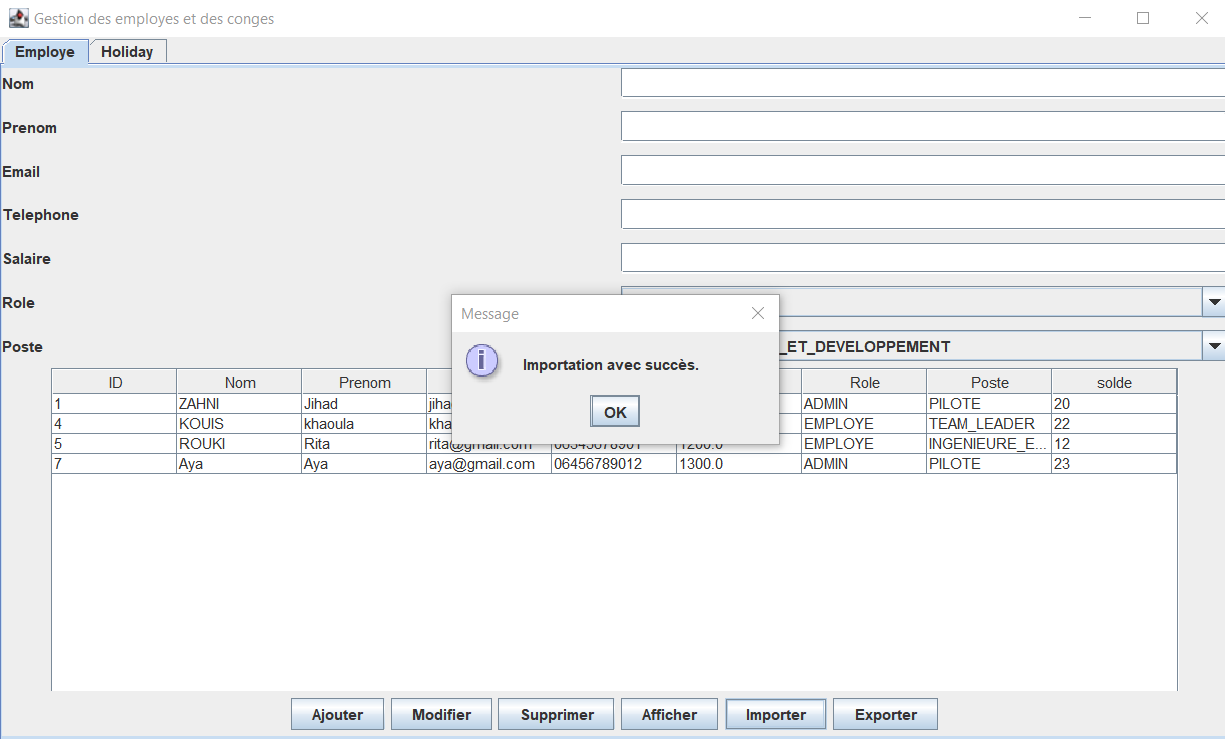
-On clique sur importer:

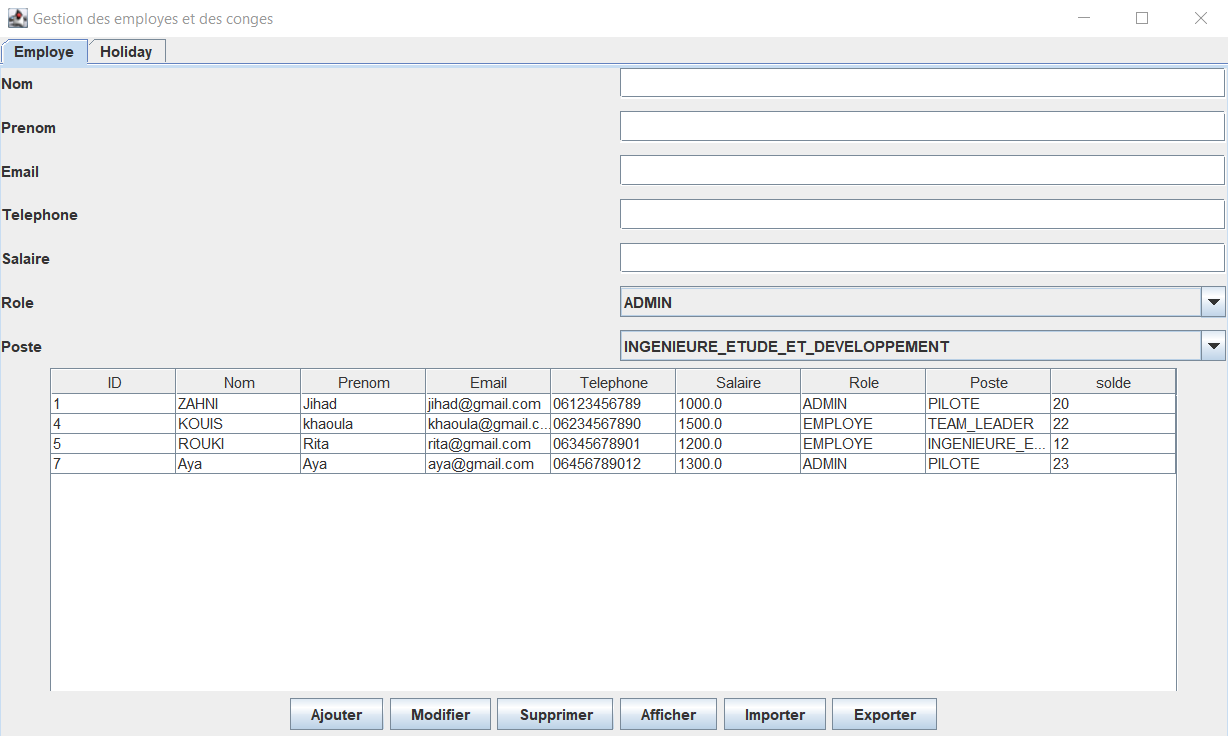


-On choisi notre fichier:



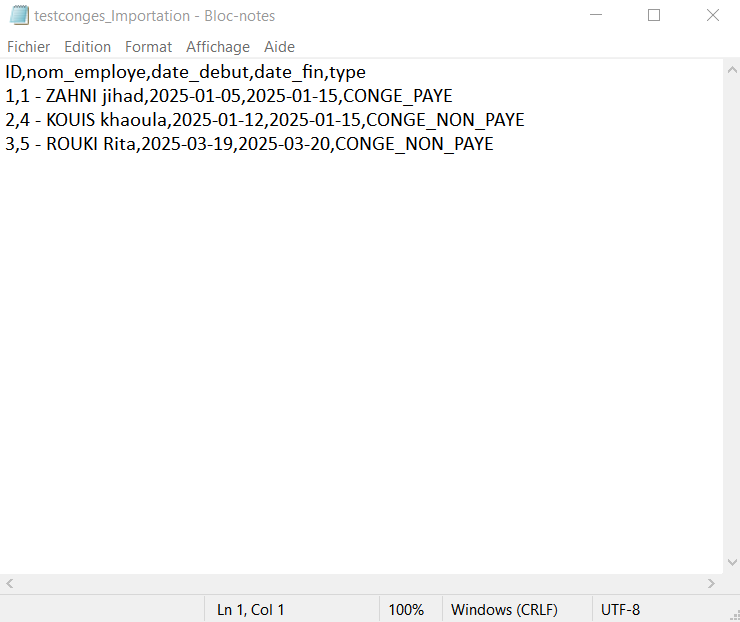
-on l’ouvre:

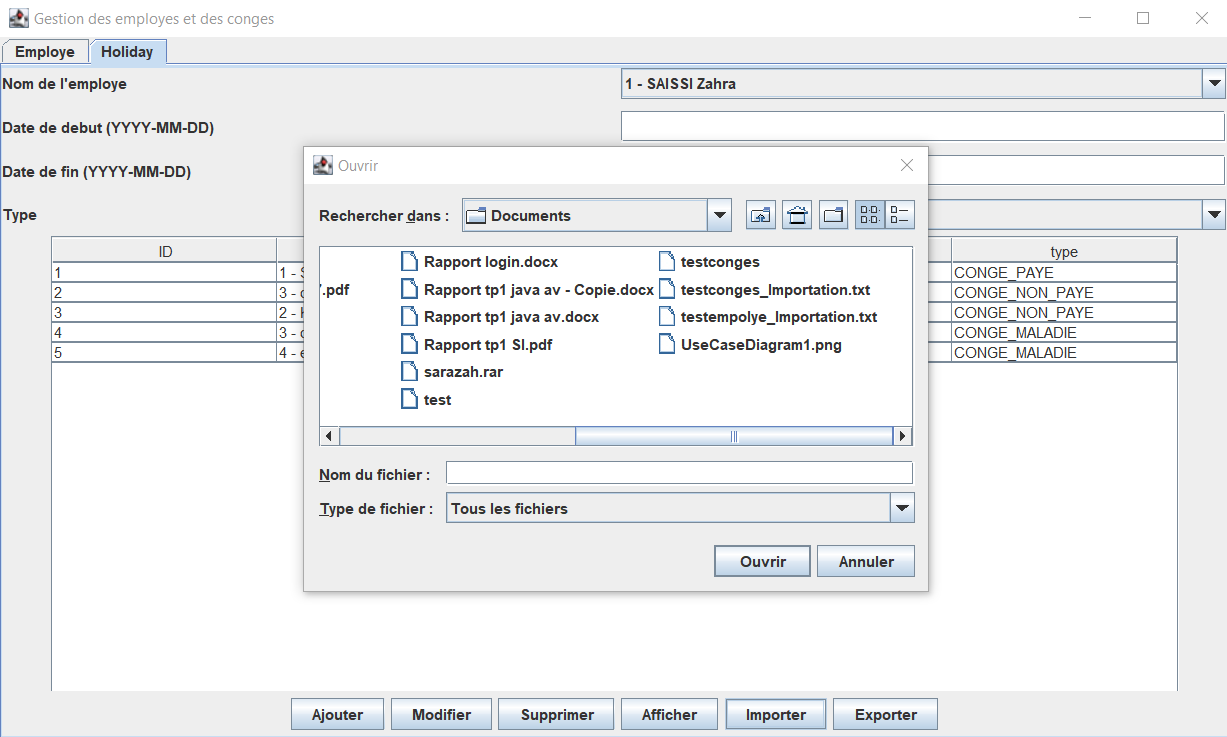


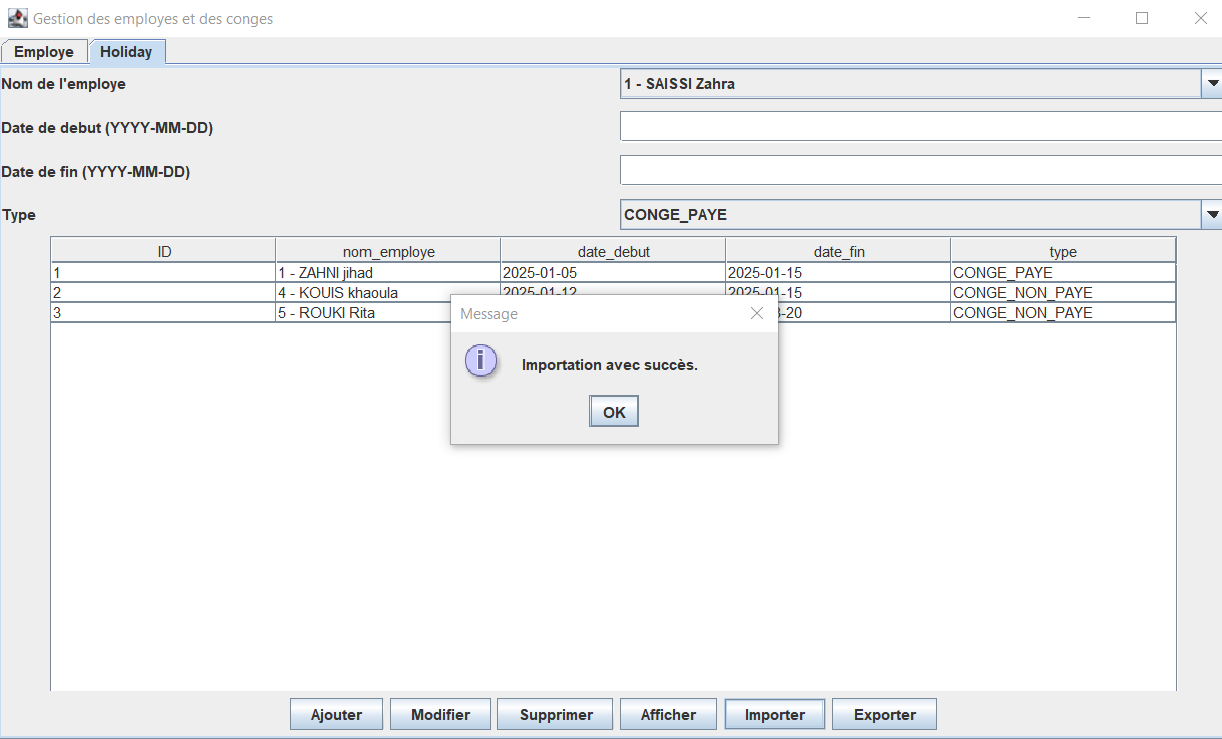


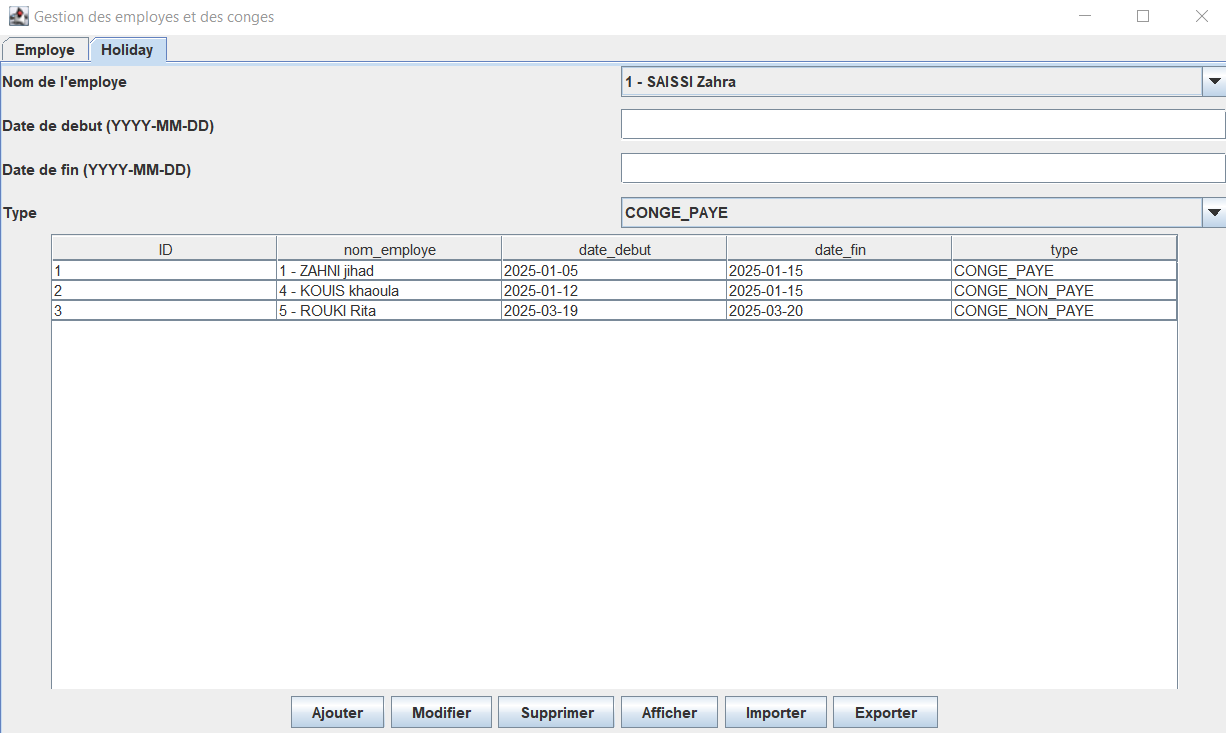
1. **Importation des conges:**

**-**On suit les memes etapes:









**Conclusion**

L'ajout des fonctionnalités d'E/S à l'application de gestion des congés représente une étape cruciale pour améliorer son efficacité et sa convivialité. Grâce à l'intégration des mécanismes d'import/export, l'application devient plus flexible et adaptée aux besoins des utilisateurs. Cette extension, conforme au modèle MVC et à l'approche DAO, garantit une architecture bien structurée et maintenable, tout en offrant une gestion optimisée des données liées aux employés et à leurs congés.