Rapport du projet JAVA
GESTION BANCAIRE

Créer par :
MOHAMED LARBI BENJELLOUN
MOHAMMED ABIDOU
Encadré par :
P .EZZATI

Introduction

Ce système de gestion bancaire est une application Java qui permet à un utilisateur de :

- Créer un compte bancaire.
- Se connecter à un compte existant.
- Effectuer des dépôts et des retraits.
- Consulter le solde de son compte.
- Effectuer des virements entre comptes.
- Se déconnecter pour se connecter à un autre compte.

Le système utilise une interface graphique (Swing) pour interagir avec l'utilisateur et une base de données simple pour stocker les informations sur les comptes. Son objectif principal est de simplifier les opérations bancaires en automatisant les processus manuels et en offrant une interface utilisateur intuitive et modulaire.
Ce projet est également conçu pour être facilement extensible, permettant l'ajout futur de nouvelles fonctionnalités bancaires.

Analyse du Problème

Lors de l'élaboration de ce système, plusieurs problèmes ont été identifiés et résolus :

1. **Le stockage des comptes n'était pas fiable** : Un fichier texte a été utilisé pour stocker les données des comptes et assurer leur persistance. Cela garantit que les données restent accessibles même après la fermeture de l'application.
2. **Pas de fonctionnalité de virement** : Une classe Virement a été ajoutée pour permettre les transferts entre comptes. Cette fonctionnalité a été conçue pour traiter les transferts en toute sécurité et en validant les fonds disponibles.
3. **Navigation limitée** : Un bouton "Se déconnecter" a été ajouté dans le menu pour permettre à l'utilisateur de se reconnecter avec un autre compte. Cette option améliore l'expérience utilisateur en ajoutant de la flexibilité.

Ce rapport détaille chaque classe, son rôle, et son fonctionnement afin de fournir une documentation complète et utile pour toute équipe désireuse de modifier ou de maintenir le projet.

Classes et Fonctionnalités

1. Classe Compte

Rôle : Modéliser un compte bancaire avec ses propriétés principales et fournir des méthodes pour gérer le solde. Cette classe représente l'unité centrale du système, autour de laquelle toutes les opérations sont effectuées.

Propriétés :

- prenom : Le prénom du titulaire.
- nom : Le nom de famille du titulaire.
- idCompte : Un identifiant unique pour le compte.
- solde : Le solde actuel du compte.
- motDePasse : Le mot de passe pour l'accès au compte.

Méthodes importantes :

- deposer(double montant) : Ajoute un montant au solde.
- retirer(double montant) : Retire un montant du solde.

Cette classe inclut des validations de base pour garantir que les opérations sur le solde sont sûres.

2. Classe BaseDeDonnees

Rôle : Gérer le stockage et la récupération des comptes dans un fichier texte. Elle permet d'assurer la persistance des données utilisateur et garantit que les informations ne sont pas perdues lorsque l'application est fermée.

Propriétés :

- fichier : Le fichier texte qui contient les données des comptes.

Méthodes importantes :

- chargerComptes() : Lit les données à partir du fichier et recrée les objets Compte.

- sauvegarderComptes(ArrayList<Compte> comptes) : Sauvegarde les données des comptes dans le fichier.

Cette classe est un exemple simple mais efficace de gestion de la persistance, en utilisant un format texte facilement lisible et modifiable.

3. Classe Connexion

Rôle : Fournir une interface pour que les utilisateurs puissent se connecter à leurs comptes ou créer un nouveau compte. Elle constitue la première interaction de l'utilisateur avec l'application.

Fonctionnalités :

- Demande l'identifiant et le mot de passe pour se connecter.
- Vérifie si l'identifiant et le mot de passe correspondent à un compte existant.
- Permet à l'utilisateur de créer un nouveau compte s'il n'a pas d'identifiant.

En cas d'erreur d'authentification, un message clair est affiché à l'utilisateur.

4. Classe CreerCompte

Rôle : Permettre à l'utilisateur de créer un nouveau compte avec les informations nécessaires. Cette classe ajoute un nouvel utilisateur à la base de données et gère les validations requises.

Fonctionnalités :

- Demande le prénom, le nom, le solde initial et un mot de passe.
- Valide les entrées (ex. : solde initial doit être positif, mot de passe doit être composé de 4 chiffres).
- Crée un nouvel objet Compte et l'ajoute à la base de données.

Des validations rigoureuses sont implémentées pour minimiser les erreurs et garantir une création de compte correcte.

5. Classe Menu

Rôle : Offrir des options pour gérer un compte une fois connecté. Le menu est le point d'accès principal pour toutes les opérations disponibles sur un compte.

Fonctionnalités :

- Permet d'effectuer un dépôt (avec la classe Depot).
- Permet de retirer des fonds (avec la classe Retrait).
- Permet de consulter le solde.
- Permet d'effectuer un virement vers un autre compte (avec la classe Virement).
- Fournit un bouton "Se déconnecter" pour revenir à l'écran de connexion.

Cette classe sert de passerelle entre l'utilisateur et les opérations bancaires. Elle simplifie la navigation et améliore l'expérience utilisateur.

6. Classe Depot

Rôle : Permettre à l'utilisateur de déposer de l'argent sur son compte.

Fonctionnalités :

- Demande le montant à déposer.
- Ajoute le montant au solde du compte.
- Sauvegarde les modifications dans la base de données.

Le dépôt est validé pour s'assurer que les montants sont logiques et positifs.

Conclusion

Ce système de gestion bancaire est conçu pour être fiable, modulaire et facile à utiliser. Chaque classe remplit un rôle spécifique pour garantir la clarté et la maintenabilité du projet. L'intégration des fonctionnalités telles que la création de comptes, les dépôts, retraits, virements, et la navigation entre les comptes permet de couvrir les besoins essentiels d'une gestion bancaire. Grâce à son architecture extensible, ce système peut être facilement amélioré pour inclure des fonctionnalités plus avancées à l'avenir.