SAÉ 2.01

Documentation technique

Sommaire :

1) Présentation de l'application

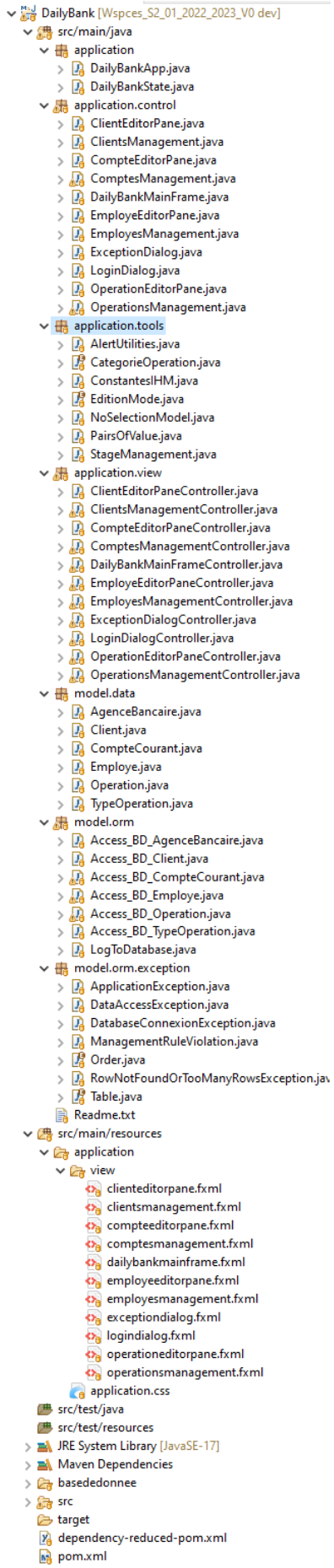
La banque DailyBank souhaite développer une application de gestion des comptes clients en utilisant JAVA-Oracle pour remplacer plusieurs outils obsolètes. Cette initiative s'inscrit dans le cadre de la restructuration de ses services bancaires.

Le travail sera basé sur une application existante nommée "Daily Bank" à laquelle nous allons apporter de nouvelles fonctionnalités pour répondre aux nouveaux besoins de la banque.

Cette application sera déployée dans les 100 agences que compte son réseau.

Le rôle de l'application va être de gérer des comptes bancaires, de pouvoir débiter ainsi que créditer soit par virement bancaire soit par une personne physique auprès d'un guichet de la banque.

2) Architecture

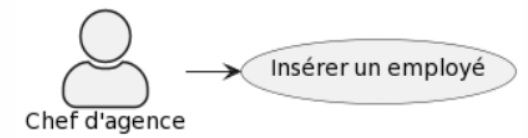

DailyBank [Wspces_S2_01_2022_2023_V0 dev]
src/main/java
application
DailyBankApp.java
DailyBankState.java
application.control
ClientEditorPane.java
ClientsManagement.java
CompteEditorPane.java
ComptesManagement.java
DailyBankMainFrame.java
EmployeEditorPane.java
EmployesManagement.java
ExceptionDialog.java
LoginDialog.java
OperationEditorPane.java
OperationsManagement.java
application.tools
AlertUtilities.java
CategorieOperation.java
ConstantesIHM.java
EditionMode.java
NoSelectionModel.java
PairsOfValue.java
StageManagement.java
application.view
ClientEditorPaneController.java
ClientsManagementController.java
CompteEditorPaneController.java
ComptesManagementController.java
DailyBankMainFrameController.java
EmployeEditorPaneController.java
EmployesManagementController.java
ExceptionDialogController.java
LoginDialogController.java
OperationEditorPaneController.java
OperationsManagementController.java
model.data
AgenceBancaire.java
Client.java
CompteCourant.java
Employe.java
Operation.java
TypeOperation.java
model.orm
Access_BD_AgenceBancaire.java
Access_BD_Client.java
Access_BD_CompteCourant.java
Access_BD_Employe.java
Access_BD_Operation.java
Access_BD_TypeOperation.java
LogToDatabase.java
model.orm.exception
ApplicationException.java
DataAccessException.java
DatabaseConnexionException.java
ManagementRuleViolation.java
Order.java
RowNotFoundOrTooManyRowsException.jav
Table.java
Readme.txt
src/main/resources
application
view
clienteditorpane.fxml
clientsmanagement.fxml
compteeditorpane.fxml
comptesmanagement.fxml
dailybankmainframe.fxml
employeeditorpane.fxml
employesmanagement.fxml
exceptiondialog.fxml
logindialog.fxml
operationeditorpane.fxml
operationsmanagement.fxml
application.css
src/test/java
src/test/resources
JRE System Library [JavaSE-17]
Maven Dependencies
basededonnee
src
target
dependency-reduced-pom.xml
pom.xml

3) Fonctionnalités

a) Version 1

i) Créer des employés (A. B.)

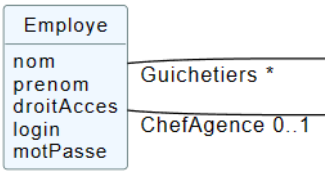
Use case de la fonctionnalité :



Classes concernées :

application.control.EmployesManagement (nouveauEmploye)
application.view.EmployesManagementController
(doNouveauEmploye)
model.orm.Acces_BD_Employe (insertEmploye)

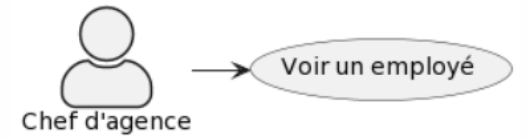
Partie du diagramme de classes concernée :



Lorsque le chef d'agence ajoute un nouvel employé, il est ajouté dans la base de données et dans la liste des employés. Pour plus de détails, voir doc utilisateur.

ii) Voir les employés (A. B.)

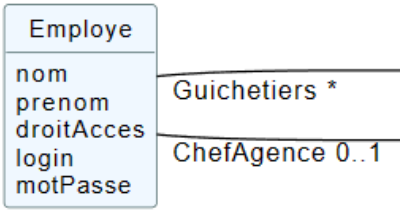
Use case de la fonctionnalité :



Classes concernées :

application.control.EmployesManagement (getlisteEmployes)
application.view.EmployesManagementController (doRechercher)
model.orm.Acces_BD_Employe (getEmploye)

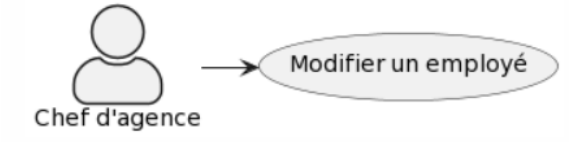
Partie du diagramme de classes concernée :



Lorsque le chef d'agence utilise la fonctionnalité rechercher, la liste des employés est affichée. Pour plus de détails, voir doc utilisateur.

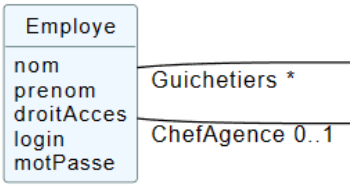
iii) Modifier les employés (H. B.)

Use case de la fonctionnalité :



Classes concernées : application.control.EmployesManagement (modifierEmploye)
application.view.EmployesManagementController (doModifierEmploye)
model.orm.Acces_BD_Employe (updateEmploye)

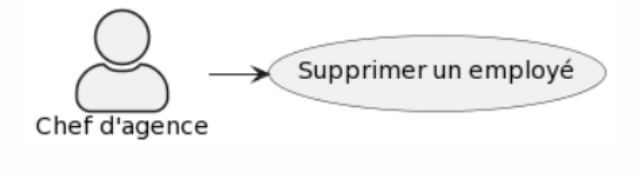
Partie du diagramme de classes concernée :



Lorsque le chef d'agence modifie un employé, les champs modifiés sont changés et les autres sont laissés tels quels. L'identifiant unique de l'employé ne peut cependant pas être modifié pour éviter d'avoir des employés en double. Pour plus de détails, voir doc utilisateur.

iv) Effacer les employés (H. B.)

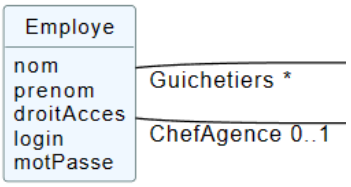
Use case de la fonctionnalité :



Classes concernées :

application.control.EmployesManagement (supprimerEmploye)
application.view.EmployesManagementController (doSupprimerEmploye)
model.orm.Acces_BD_Employe (removeEmploye)

Partie du diagramme de classes concernée :



Lorsque le chef d'agence supprime un employé, celui-ci est supprimé de la base de données ainsi que de la liste des employés. Pour plus de détails, voir doc utilisateur.

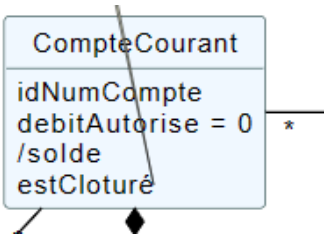
v) Créer un compte (B. S.)

Use case de la fonctionnalité :

Classes concernées:

application.control.ComptesManagement(creerNouveauCompte)
model.orm.Access_BD_CompteCourant(insertCompteCourrant)
application.view.ComptesManagementController(doNouveauCompte)

Partie du diagramme de classes concernée :



Lorsque le guichetier crée un nouveau compte, il est ajouté dans la base de données et le débit autorisé est de 0 par défaut. Pour plus de détails, voir doc utilisateur.

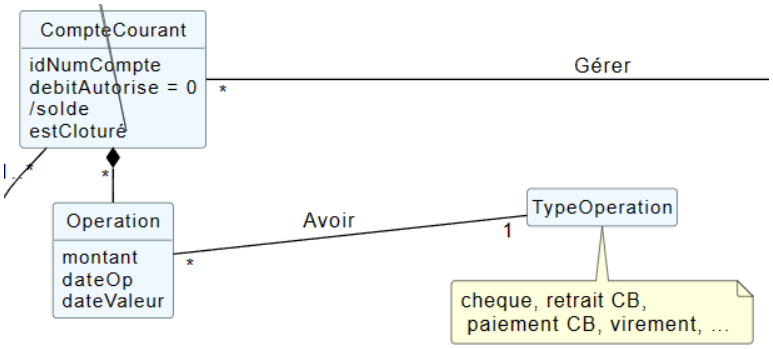
vi) Créditer / Débiter sur un compte (B. S.)

Use case de la fonctionnalité :

Classes concernées:

application.view.OperationsManagementController (doCredit)
model.orm.Access_BD_Operation (insertCredit)
application.control.OperationsManagement (enregistrerCredit)
application.view.OperationEditorPaneController (doAjouter -> case crédit , displayDialog -> case crédit)

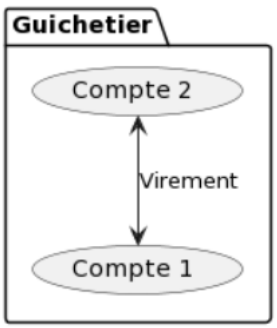
Partie du diagramme de classes concernée :



Lorsque un guichetier effectue un crédit ou un débit sur un compte, le solde de ce compte est modifié, la base de données est modifiée en conséquence. Pour plus de détails, voir doc utilisateur.

vii) Effectuer un virement (J. M.)

Use case de la fonctionnalité :



Classes concernées :

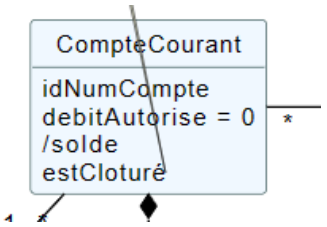
viii) Clôturer un compte (J. M.)

Use case de la fonctionnalité :

Classes concernées :

model.orm.Access_BD_CompteCourant (cloturerCompteCourant)
application.view.ComptesManagementController(doSupprimerCompte)
application.view.ComptesManagementController(validateComponentState)
application.view.OperationManagementController(validateComponentState)

Partie du diagramme de classes concernées :



Lorsque le guichetier clôture un compte, on ne peut plus effectuer d'opérations sur ce compte mais il reste dans la base de données. Pour plus de détails, voir doc utilisateur.

b) Version 2 (en cours de développement)

i) Débit exceptionnel

ii) Simuler emprunt

iii) ...