LITZLER Marco

DE ABREU Quentin

ROLLAND Julia

Rapport Projet gl52   
Prototype Gestion Magasin ÉlectromÉnagÉ

UV GL52           P21

Professeur : Abderrafiaa Koukam

Table des matières

[1- Introduction 4](#_Toc74675476)

[Buts et destinataires 4](#_Toc74675477)

[Présentations générale du document 4](#_Toc74675478)

[2- Description générale 5](#_Toc74675479)

[Environnement ou contexte du système 5](#_Toc74675480)

[Caractéristique des utilisateurs 5](#_Toc74675481)

[Les contraintes principales de développement 5](#_Toc74675482)

[Hypothèse de travail 5](#_Toc74675483)

[3- Besoin fonctionnels 6](#_Toc74675484)

[Nom du cas d'utilisation 6](#_Toc74675485)

[4- Spécification des structures de données 6](#_Toc74675486)

[5- Spécifications des interfaces externes 6](#_Toc74675487)

[Interface matériel/logiciel 6](#_Toc74675488)

[Interface logiciel/logiciel 6](#_Toc74675489)

[Interface Homme/logiciel 6](#_Toc74675490)

[6- Les besoins en performance 6](#_Toc74675491)

[7- Les contraintes de développement 6](#_Toc74675492)

[8- Références 6](#_Toc74675493)

[9- Index 6](#_Toc74675494)

[10- Annexes 13](#_Toc74675495)

# Introduction

## Buts et destinataires

L’objectif de ce projet est de proposer un prototype d’un logiciel permettant la gestion d’un magasin d’électroménager. Celui-ci doit faciliter le bon déroulement de la vente et du service après-vente du magasin.

L’organisme commanditaire souhaite remettre au gout du jour son outil déjà existant, dans l’objectif d’avoir une meilleure gestion de leurs ressources et de simplifier l’exécution des différentes tâches de leurs employé.

## Présentations générale du document

Ce document présente donc dans un premier temps le projet d’un point de vue générale à l’aide d’une brève description et de diagrammes UML.

# Description générale

## Environnement ou contexte du système

Contexte : Spécification en langage naturel

Le logiciel permet de gérer un magasin d’électroménager en trois parties : la vente, le SAV, le stock.

La vente :

Concernant la partie vente. Un client veut acheter un article disponible dans le magasin. Pour ce faire, un membre du personnel prend les références de l’article et du client (S’il n’est pas déjà dans le fichier client). Le client peut payer par carte ou en espèces.

Le service après-vente (SAV) :

Concernant la partie SAV. Un client a un problème ou souhaite rendre le produit qu’il a acheté. Un membre du personnel prend les références de l’article et du client. Puis il vérifie la date d’achat du produit, si celui-ci est garanti ou non. Le produit est ensuite étudié  pour voir s’il est neuf (pour un retour) ou s’il est défectueux comme indiqué par le client. En fonction de l’état du produit, on a plusieurs cas possibles :

Il est repris et on rembourse ou donne un avoir au client.

Il est envoyé en réparation puis donné au client.

Il n’est pas réparable. S’il en reste en stock, on donne un nouveau produit au client sinon on le rembourse ou lui donne un avoir.

Il n’a pas de défaut, on le rend au client.

Possibilité de prêter un produit pendant la durée de la réparation.

Le stock :

Concernant la gestion du stock. Possibilité de voir le stock en temps réel de l’ensemble du magasin. Possibilité de modifier le stock par exemple lors d’un inventaire.

## Caractéristique des utilisateurs

Le client :

Il n’est pas l’utilisateur direct du système, ses informations transitent par un employé de la boutique avant d'être intégrées dans le logiciel. Il est un acteur extérieur du système.

Les employés :

Les employés de l’entreprise sont polyvalents et assurent  différents rôles liés à différentes fonctions du logiciel (vente, SAV, gestion du stock).

## Les contraintes principales de développement

???

## Hypothèse de travail

???

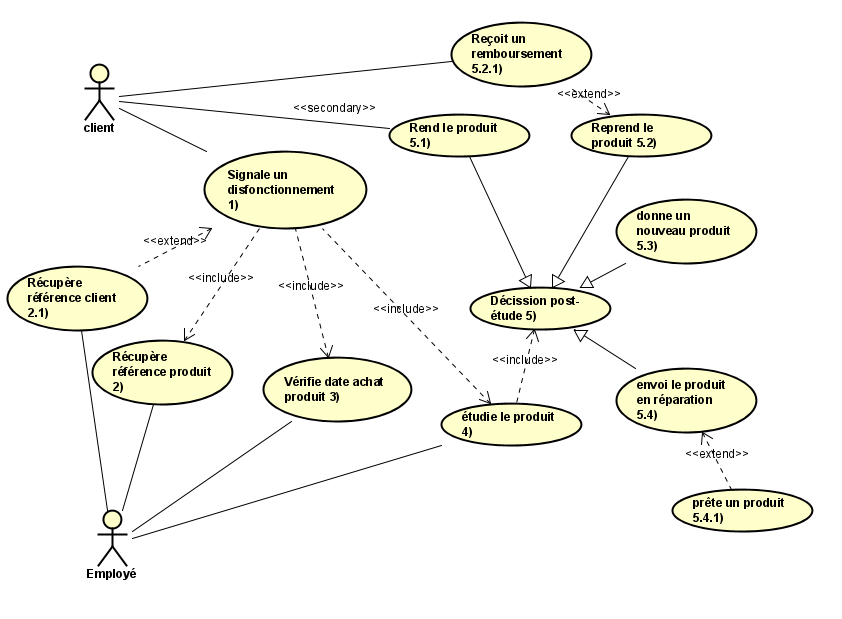
# Besoin fonctionnels

## Nom du cas d'utilisation : Diagramme cas d’utilisation

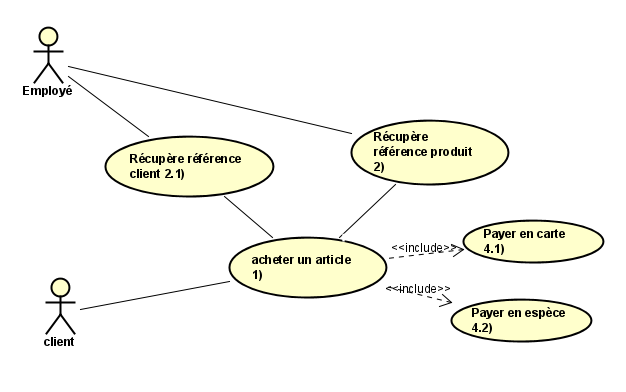
Faire tableau récapitulatif pour chaque cas d’un diagramme

|  |  |
| --- | --- |
| Use Case ID |  |
| Nom |  |
| Premier Acteur |  |
| Deuxième Acteur |  |
| Description |  |
| Pré-Conditions |  |
| Post-Conditions |  |
| Événements principaux |  |
| Evènements exceptionnels |  |

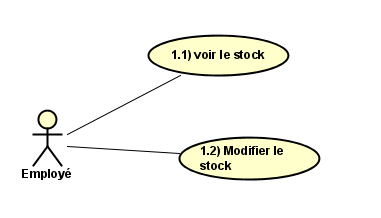
### SAV



### Vente



### Stock



# Spécification des structures de données

Une image contenant texte, noir, métal, capture d’écran

Description générée automatiquement

Figure 1 : Diagramme de classe - package model

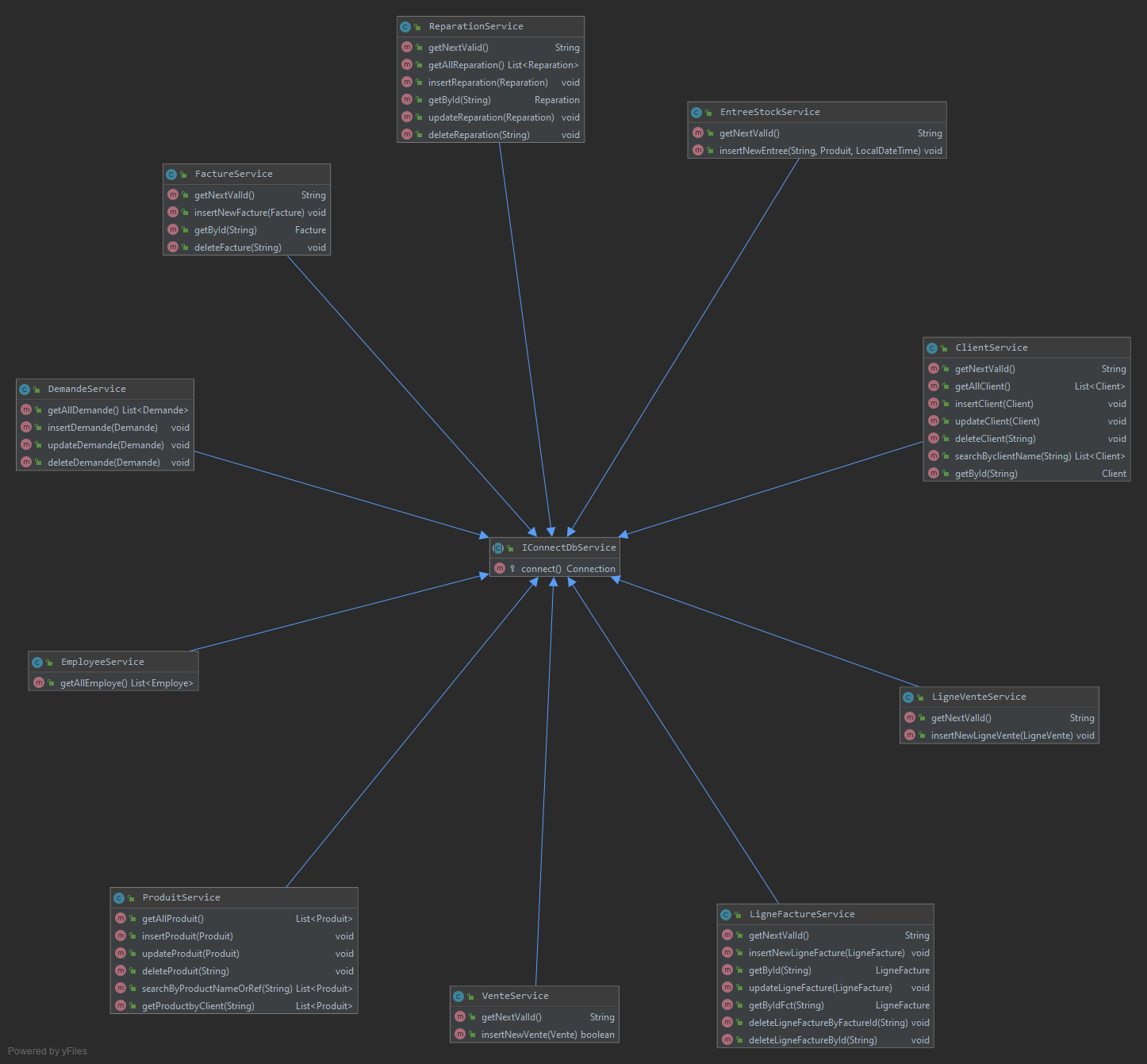


Figure 2: diagramme de classe - package service

Une image contenant texte, noir, capture d’écran

Description générée automatiquement

Figure 3: diagramme de classe - package controller

Une image contenant texte

Description générée automatiquement

Figure 4: diagramme de classe - package ui

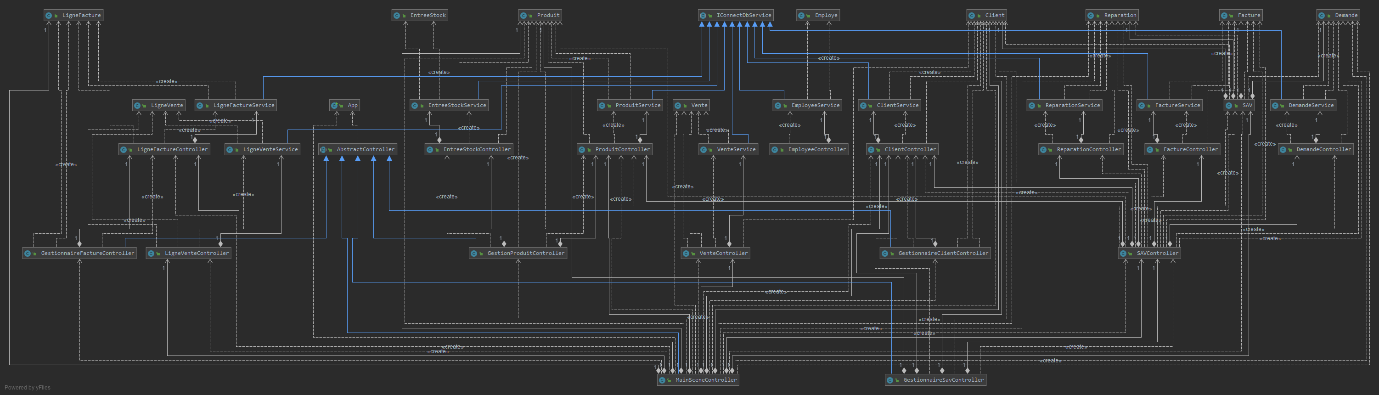


Figure 5 : diagramme de classe - projet complet

# Spécifications des interfaces externes

## Interface matériel/logiciel

## Interface logiciel/logiciel

## Interface Homme/logiciel

# Les besoins en performance

# Les contraintes de développement

# Références

# Index

Package ui

File App

private static Parent loadFXML(String fxml) throws IOException

public static void main(String[] args)

public static void setRoot(String fxml) throws IOException

public void start(@SuppressWarnings("exports") Stage stage) throws IOException

public static FXMLLoader getFxmlLoader()

File controller\AbstractController

public void switchToMainScene() throws IOException

File controller\GestionnaireClientController

private Client setClient(String id)

private boolean checkTextFieldIsNull()

private void handleClientAdresse(ActionEvent event)

private void handleClientCp(ActionEvent event)

private void handleClientEmail(ActionEvent event)

private void handleClientFirstName(ActionEvent event)

private void handleClientLastName(ActionEvent event)

private void handleClientPays(ActionEvent event)

private void handleClientPhone(ActionEvent event)

private void handleClientRue(ActionEvent event)

private void handleClientVille(ActionEvent event)

private void handleClientnum(ActionEvent event)

private void isNew() throws IOException

private void modifyClient() throws IOException

private void newClient() throws IOException

public void initialize(URL location, ResourceBundle resources)

public void setDataForUpdate(@SuppressWarnings("exports") Client client)

public void switchToMainScene() throws IOException

public void switchToMainSceneValidate() throws IOException

File controller\GestionnaireFactureController

private LigneFacture setLigneFacture(String id)

private void handleDesRepTxtField(ActionEvent event)

private void handlePrixRepTxtField(ActionEvent event)

private void modifyLigneFacture() throws IOException

private void newLigneFacture() throws IOException

public void setDataForUpdate(@SuppressWarnings("exports") LigneFacture ligneFacture)

public void setNumFct(String numFct)

File controller\GestionnaireSavController

private void handleReparationDescription(ActionEvent event)

private void handleReparationNature(ActionEvent event)

private void handleReparationNumEmployee(ActionEvent event)

private void newSAV() throws IOException

public void getSearchClientVente()

public void setClientProduct()

File controller\GestionProduitController

private Produit setProduit() throws IOException

private void handleProductDescription(ActionEvent event)

private void handleProductHTPrice(ActionEvent event)

private void handleProductQuantity(ActionEvent event)

private void handleProductReference(ActionEvent event)

private void handleProductTTCPrice(ActionEvent event)

private void modifyProduit() throws IOException

private void newProduit() throws IOException

public void setDataForUpdate(@SuppressWarnings("exports") Produit produit)

File controller\MainSceneController

private Vente setSell()

private void setSellLine(String numVte, String refProd, String qtt)

private void switchToGestionnaireClient() throws IOException

public MainSceneController()

public void addFactureLine() throws IOException

public void cancelSell()

public void changeMontantVenteValue()

public void changeSavTextFieldValues()

public void createClient() throws IOException

public void createProduit() throws IOException

public void createReparation() throws IOException

public void deleteClient() throws IOException

public void deleteFactureLine()

public void deleteProduit() throws IOException

public void deleteReparation()

public void getSearchProductStock()

public void getSearchProductVente()

public void modifyClient() throws IOException

public void modifyFactureLine() throws IOException

public void modifyProduit() throws IOException

public void modifyReparation() throws IOException

public void selectClientVenteInfo()

public void sell()

Package model

File Client

public Client()

public String getAdresse()

public String getCpCli()

public String getMailCli()

public String getNomCli()

public String getNumCli()

public String getPaysCli()

public String getPrenomCli()

public String getRueCli()

public String getTelCli()

public String getVilleCli()

public String toString()

public void setCpCli(String cpCli)

public void setMailCli(String mailCli)

public void setNomCli(String nomCli)

public void setNumCli(String numCli)

public void setPaysCli(String paysCli)

public void setPrenomCli(String prenomCli)

public void setRueCli(String rueCli)

public void setTelCli(String telCli)

public void setVilleCli(String villeCli)

File Employe

public Employe()

public String getCpEmp()

public String getMailEmp()

public String getNomEmp()

public String getNumEmp()

public String getPaysEmp()

public String getPrenomEmp()

public String getRueEmp()

public String getTelEmp()

public String getVilleEmp()

public void setCpEmp(String cpEmp)

public void setMailEmp(String mailEmp)

public void setNomEmp(String nomEmp)

public void setNumEmp(String numEmp)

public void setPaysEmp(String paysEmp)

public void setPrenomEmp(String prenomEmp)

public void setRueEmp(String rueEmp)

public void setTelEmp(String telEmp)

public void setVilleEmp(String villeEmp)

File Produit

public Produit()

public Produit(String referenceProd, String descriptionProduct, String quantity, String prixHTT, String prixTTC, String tVA)

public String getDesProd()

public String getPrixHTProd()

public String getPrixTTCProd()

public String getQteProd()

public String getRefProd()

public String getTVAProd()

public boolean equals(Object obj)

public final String getDesProdProperty()

public final String getPrixHTProdProperty()

public final String getPrixTTCProdProperty()

public final String getQteProdProperty()

public final String getRefProdProperty()

public final String getTVAProdProperty()

public int hashCode()

public void setDesProd(String desProd)

public void setPrixHTProd(String prixHTProd)

public void setPrixTTCProd(String prixTTCProd)

public void setQteProd(String qteProd)

public void setRefProd(String refProd)

public void setTVAProd(String tVAProd)

File sav\Demande

public Demande()

public Demande(String numRep, String numCli, String natureRep, String descRep, String refProd, String numFct)

public String getDescRep()

public String getNatureRep()

public String getNumFct()

public String getNumRep()

public void setDescRep(String descRep)

public void setNatureRep(String natureRep)

public void setNumRep(String numRep)

File sav\Reparation

public Reparation()

public Reparation(String numRep, String etatRep)

public String getEtatRep()

public void setEtatRep(String etatRep)

File sav\SAV

public Client getClient()

public Demande getDemande()

public Facture getFacture()

public Reparation getReparation()

public SAV()

public SAV(Client client, Demande demande, Reparation reparation, Facture facture)

public void setClient(Client client)

public void setDemande(Demande demande)

public void setFacture(Facture facture)

public void setReparation(Reparation reparation)

File sav\facture\Facture

public Facture()

public Facture(String numFct, String dateFct, String numEmp)

public String getDateFct()

public void setDateFct(String dateFct)

File sav\facture\LigneFacture

public LigneFacture()

public LigneFacture(String idLigneFct, String numFct, String mttLigneFct, String desLigneFct)

public String getDesLigneFct()

public String getIdLigneFct()

public String getMttLigneFct()

public void setDseLigneFct(String desLigneFct)

public void setIdLigneFct(String idLigneFct)

public void setMttLigneFct(String mttLigneFct)

File stock\EntreeStock

public EntreeStock()

public EntreeStock(String idEntr, String qteEnStock, String refProduct, String dateEntree)

public String getDateEntree()

public String getIdEntr()

public String getQteEnStock()

public String getRefProduct()

public void setDateEntree(String dateEntree)

public void setIdEntr(String idEntr)

public void setQteEnStock(String qteStock)

public void setRefProduct(String refProduct)

File vente\LigneVente

public LigneVente()

public LigneVente(String id,String qteLigne, String refProd, String numVte)

public String getId()

public String getNumVte()

public String getQteLigne()

public void setId(String id)

public void setNumVte(String numVte)

public void setQteLigne(String qteLigne)

File vente\Vente

public String getDate()

public String getMontantVte()

public Vente()

public Vente(String numVte, String numCli, String numEmp, String date, String montantVte)

public void setDate(String date)

public void setMontantVte(String montantVte)

Package controller

File ClientController

public Client getById(String numCli)

public ClientController()

public List<Client> getAllClient()

public List<Client> searchByClientName(String searchName)

public String getNextValId()

public void deleteClient(Client client)

public void insertNewClient(Client client)

public void modifyClient(Client client)

File EmployeeController

public EmployeeController()

File ProduitController

public void modifyProduit(Produit produit)

public void modifyProduitList(List<Produit> producList)

public List<Produit> getAllProduit()

public List<Produit> getProductbyClient(String numCli)

public List<Produit> searchByProductNameOrRef(String searchName)

public ProduitController()

public void deleteProduit(String referenceProd)

public void insertNewProduit(Produit produit)

File sav\DemandeController

public DemandeController()

public List<Demande> getAllDemande()

public void deleteDemande(Demande demande)

public void insertNewDemande(Demande demande)

public void modifyDemande(Demande demande)

File sav\ReparationController

public Reparation getById(String numRep)

public ReparationController()

public void deleteReparation(String numRep)

public void insertNewReparation(Reparation reparation)

public void modifyReparation(Reparation reparation)

File sav\SAVController

public List<Produit> getProductbyClient(String numCli)

public List<SAV> getAllSav()

public void createSAV(Client client, Produit produit , String descRep, String natureRep, String numEmp)

public void deleteSAV(Demande demande)

public void modifySAV(Reparation reparation, Demande demande)

File sav\facture\FactureController

public Facture getById(String numFct)

public FactureController()

public void deleteFacture(String numFct)

public void insertNewFacture(Facture facture)

File sav\facture\LigneFactureController

public LigneFacture getById(String idLigneFct)

public LigneFacture getByIdFct(String numFct)

public LigneFactureController()

public void deleteLigneFactureByFactureId(String numFct)

public void deleteLigneFactureById(String idLigneFct)

public void insertNewLigneFacture(LigneFacture ligneFacture)

public void updateLigneFacture(LigneFacture ligneFacture)

File stock\EntreeStockController

public EntreeStockController()

public void insertNewEntree(Produit produit)

File vente\LigneVenteController

public LigneVenteController()

public void insertNewLigneVente(String numVte, String refProd, String qtt)

File vente\VenteController

public Vente insertNewVente(Client client, String numEmp, float mttTotal)

public VenteController()

Package service

File ClientService

public ClientService()

public List<Client> searchByclientName(String searchName)

public int compare(Client c1, Client c2)

public void deleteClient(String numCli)

public void insertClient(Client client)

public void updateClient(Client client)

File EmployeeService

public EmployeeService()

public List<Employe> getAllEmploye()

File IConnectDbService

protected Connection connect()

File ProduitService

public ProduitService()

public int compare(Produit c1, Produit c2)

public void deleteProduit(String refProd)

public void insertProduit(Produit produit)

public void updateProduit(Produit produit)

File sav\DemandeService

public DemandeService()

public int compare(Demande d1, Demande d2)

public void insertDemande(Demande demande)

public void updateDemande(Demande demande)

File sav\ReparationService

public List<Reparation> getAllReparation()

public ReparationService()

public String getNextValid()

public int compare(Reparation r1, Reparation r2)

public void insertReparation(Reparation reparation)

public void updateReparation(Reparation reparation)

File sav\facture\FactureService

public FactureService()

public Facture getById(String numFct)

public void deleteFacture(String numFct)

public void insertNewFacture(Facture facture)

File sav\facture\LigneFactureService

public LigneFactureService()

public LigneFacture getById(String idLigneFct)

public LigneFacture getByIdFct(String numFct)

public void deleteLigneFactureByFactureId(String numFct)

public void deleteLigneFactureById(String idLigneFct)

public void insertNewLigneFacture(LigneFacture ligneFacture)

public void updateLigneFacture(LigneFacture ligneFacture)

File stock\EntreeStockService

public EntreeStockService()

public void insertNewEntree(String nextValId, Produit produit, LocalDateTime now)

File vente\VenteService

public VenteService()

public boolean insertNewVente(Vente vente)

File vente\LigneVenteService

public LigneVenteService()

public int compare(LigneVente c1, LigneVente c2)

public void insertNewLigneVente(LigneVente ligneVente)

# Annexes

Outils :

* Maven
* Javafx
* SqLite Database 3

Environnement de développement :

* Eclipse IDE for Enterprise Java and Web Developers - 2021-03
* JavaFx Scene Builder 2.0