Dossier de projet « Parking »

Table des matières

[1 Objet du document 3](#_Toc43944455)

[2 Présentation du besoin 3](#_Toc43944456)

[3 Définition du besoin (DUC) 3](#_Toc43944457)

[4 Analyse 4](#_Toc43944458)

[4.1 UC détaillé 4](#_Toc43944459)

[4.1.1 UC Entrer parking 4](#_Toc43944460)

[4.1.2 UC Sortir parking 5](#_Toc43944461)

[4.1.3 UC Gérer parking 5](#_Toc43944462)

[4.2 Maquette des IHM de l’application parking 7](#_Toc43944463)

[4.2.1 Maquette de l’application de gestion du parking 7](#_Toc43944464)

[4.2.2 Maquette du simulateur du parking 7](#_Toc43944465)

[4.3 Diagramme de classe du domaine 8](#_Toc43944466)

[4.3.1 Modélisation de la persistance des données (MCD) 8](#_Toc43944467)

[5 Conception 9](#_Toc43944468)

[5.1 Diagramme de séquence (traduisant le scénario de l’UC détaillé) 9](#_Toc43944469)

[5.1.1 Diagramme de séquence interaction entrer dans le parking 9](#_Toc43944470)

[5.1.2 Diagramme de séquence interaction sortir du parking 10](#_Toc43944471)

[5.1.3 Diagramme de séquence interaction gérer le parking 11](#_Toc43944472)

[5.2 Diagramme de classe de conception (ajout des méthodes) 12](#_Toc43944473)

[5.2.1 Création du MLD 12](#_Toc43944474)

[5.2.2 Création de la base de données 13](#_Toc43944475)

[5.2.3 Architecture MVC avec la Technologies logicielle ORM JPA et DAO 14](#_Toc43944476)

[5.2.4 Éléments graphiques (conception) 15](#_Toc43944477)

[5.2.5 Environnement logiciel utilisé. 20](#_Toc43944478)

[5.2.6 Arborescence projet du code 20](#_Toc43944479)

[6 Les tests 21](#_Toc43944480)

[6.1 Tests unitaires 21](#_Toc43944481)

[6.1.1 Tests JUnit 21](#_Toc43944482)

[6.1.2 Tests d’intégrations 22](#_Toc43944483)

[7 Déploiement 23](#_Toc43944484)

[Bilan et perspective 24](#_Toc43944485)

# Objet du document

Rapport de l’analyse, de la conception, et de la réalisation du projet d’une application pour la gestion d’un parking avec UML et java.

# Présentation du besoin

La direction d’une société souhaite améliorer la sécurité dans leur immeuble et sur leur site, sans contrarier leurs employés.

Elle voudrait aussi empêcher les personnes qui ne font pas parti de la compagnie, d’utiliser leur parking.

Il a été décidé de délivrer des cartes d’identité à tous leurs employés, qu’ils devront porter sur le site.

Les cartes indiquent le nom, le département, et numéro de membre du personnel, et permettent l’accès au parking de la société.

Une barrière et un lecteur de carte sont placés à l’entrée du parking de voitures. Un capteur est positionné en entrée et en sortie du parking pour déterminer si une voiture est aperçue. Le conducteur d’une voiture s’approchant du parking introduit sa carte numérotée dans le lecteur de carte, celui-ci vérifiant que le numéro de la carte est bien connu du système. Si la carte est reconnue, le lecteur envoie un signal pour lever la barrière et la voiture peut entrer dans le parking.

A la sortie, il y a aussi une barrière, qui est levée quand une voiture souhaite quitter le parking.

Quand il n’y a plus de places dans le parking, un indicateur à l’entrée affiche « plein » et est remis en position « off » quand une voiture sort du parking.

Des cartes spéciales « visiteur », qui indiquent un numéro et la date courante, permettent aussi d’accéder au parking de voitures.

Livrables :

1. Une archive zip des sources
2. Une version exécutable, Jar ou EXE
3. Le document technique renseigné et commenté incluant en annexe, un manuel utilisateur précisant quoi et comment installer pour que l’application tourne sur le PC du client.

# Définition du besoin (DUC)



# Analyse

## UC détaillé

### UC Entrer parking

|  |  |
| --- | --- |
| **Pré-conditions** | Le parking a des places disponibles, la barrière fonctionne et la carte est valide. Les identifiants des cartes visiteur sont supérieurs ou égal à 100.  Les identifiants des carte salariés sont strictement inférieurs à 100. |
| **Scénarios** | **Nominale** |
| **1** | Le conducteur avec sa voiture se présente devant la borne d'entrée du parking |
| **2** | Le système via le capteur détecte le véhicule (un seul à la fois) |
| **3** | Le système demande la carte au conducteur |
| **4** | Le conducteur insère sa carte dans le système |
| **5** | Le système valide la carte du conducteur |
| **6** | Le système (via la barrière) autorise l'entrée et le feu passe au vert |
| **7** | Le conducteur entre et se gare dans le parking |
| **8** | Le système ferme le parking |
|  | Scénario Alternatif 1 : il n'y a plus de place visiteur |
| **1a** | Le parking affiche complet |
| **1b** | Le conducteur part |
|  | **Fin du scénario** |
|  | Scénario d'erreur 1 : Carte salarié non lisible ou oubliée |
| **5a** | Le système ne valide pas la carte et affiche le message "carte illisible refaite votre carte" |
| **5b** | Le salarié demande une réimpression de sa carte. |
|  | **Le scénario reprend à l'étape 4** |
|  | Scénario d'erreur 2: Panne barrière (problème technique) |
| **6a** | La barrière ne se lève pas, le système affiche "panne barrière" |
| **6b** | Le conducteur ne rentre pas et repart |
|  | **Fin du scénario** |
| **Post-conditions** | Le système a mis à jour le tableau d'occupation du parking |
| Exigence non fonctionnelle | |
| La société possède un parking 14 places. Elle attribue 10 places pour ses 10 salariés et 4 pour les visiteurs. | |
| Les places ne sont pas appareillées | |
| Les visiteurs obtiennent une carte numéroté et horodaté à faire lire dans le lecteur. | |

### UC Sortir parking

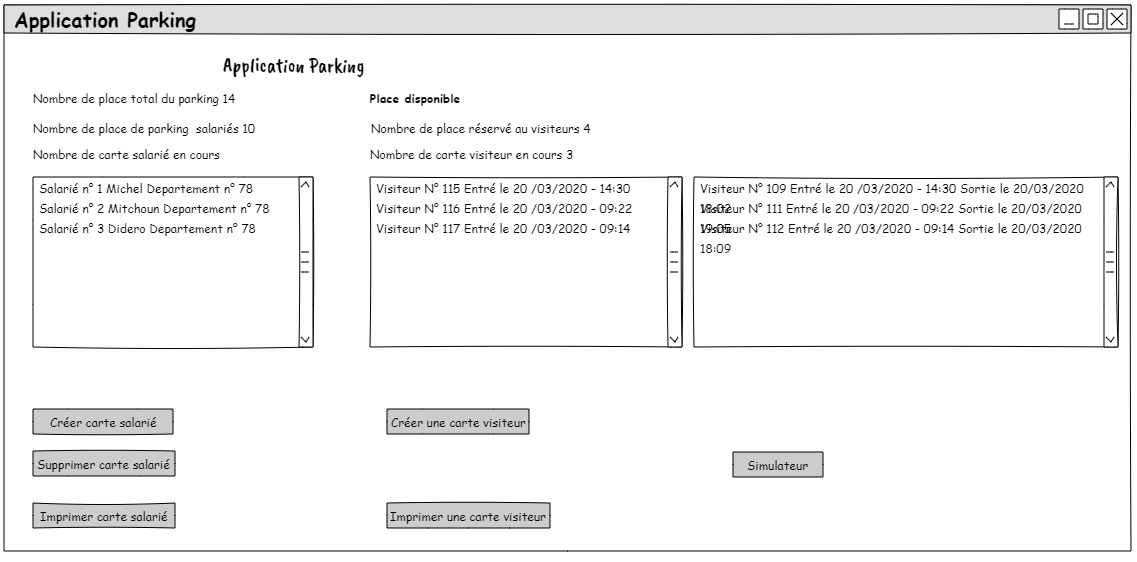
|  |  |
| --- | --- |
| **Pré-conditions** | La barrière fonctionne et la carte est valide |
| **Scénarios** | **Nominale** |
|  |  |
| **1** | Le conducteur avec sa voiture se présente devant la borne de sortie du parking |
| **2** | Le système via le capteur détecte le véhicule (un seul à la fois) |
| **3** | Le système demande la carte au conducteur |
| **4** | Le conducteur insère sa carte dans le système |
| **5** | Le système valide la carte du conducteur |
| **6** | Le système (via la barrière) autorise l'entrée et le feu passe au vert |
| **7** | Le conducteur sort du parking |
| **8** | Le système ferme le parking |
| **9** | Le système met à jour le nombre de place visiteur disponible |
|  | Scénario d'erreur 1 : Carte ou ticket non valide |
| **5a** | Le système ne valide pas la carte et affiche le message "carte illisible refaite votre carte" |
| **5b** | Le conducteur se gare et s'adressera au gardien |
|  | **Fin du scénario** |
|  | Scénario d'erreur 2 : Panne barrière |
| **5a** | La barrière ne se lève pas |
| **6a** | Le conducteur se gare et s'adressera au gardien |
|  | **Fin du scénario** |
| **Post-conditions** | Le système a mis à jour le tableau d'occupation du parking |

### UC Gérer parking

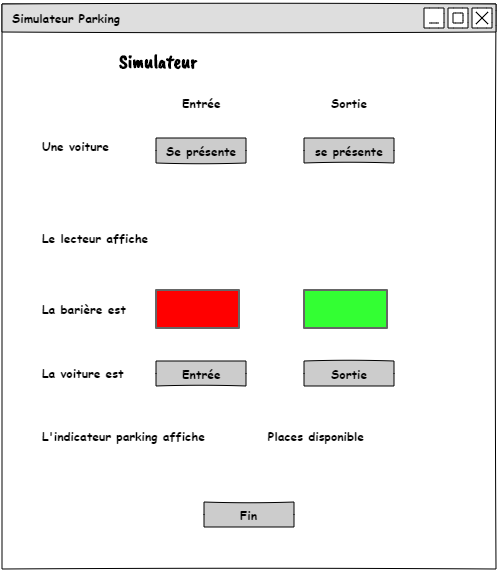
|  |  |
| --- | --- |
| **Pré-conditions** | une demande d'une nouvelle place de parking pour un salarié a été formulée. Il y est précisé le nom et le département du salarié |
| **Scénario nominale** | **Rajouter une nouvelle place de parking pour un salarié** |
| **1** | Le gestionnaire du parking reçoit la demande pour une nouvelle place de parking salarié |
| **2** | Le gestionnaire accède sa console de gestion du parking |
| **3** | Le gestionnaire demande au système de créer une nouvelle carte salarié |
| **4** | Le système vérifie qu'il reste bien une place attribuable |
| **5** | Le système transforme une place visiteur en place salarié |
| **6** | Le système génère une nouvelle carte |
| **7** | le gestionnaire transmet sa nouvelle carte au salarié |
| **Post-conditions** | Une nouvelle place de parking est attribué à un nouveau salarié |
|  | Il reste au moins une place visiteur non occupé |
| **Pré-conditions** | une demande pour supprimer une place de parking salarié a été formulée. Il y est précisé l'id de la place |
| **Scénario nominale** | **Supprimer une place de parking pour un salarié** |
| **1** | Le gestionnaire du parking reçoit la demande pour supprimer une place de parking salarié |
| **2** | Le gestionnaire accède sa console de gestion du parking |
| **3** | Le gestionnaire demande au système de supprimer une place salarié |
| **4** | Le système supprime la carte |
| **5** | Le système met à jour le nombre de place disponible |
| **Post-conditions** | Une place de parking d'un salarié est supprimée |
| **Pré-conditions** | une demande d'une nouvelle place de parking pour un visiteur. |
| **Scénario nominale** | **Rajouter une nouvelle place de parking pour un visiteur** |
| **1** | Le gestionnaire du parking reçoit la demande pour une nouvelle place de parking visiteur |
| **2** | Le gestionnaire accède sa console de gestion du parking |
| **3** | Le gestionnaire demande au système de créer une nouvelle place visiteur |
| **4** | Le système met à jour le nombre de place disponible |
| **Post-conditions** | Une nouvelle place de parking visiteur est créé |
| **Pré-conditions** | une demande pour supprimer une place de parking visiteur a été formulée. |
| **Scénario nominale** | **Supprimer une place de parking pour un visiteur** |
| **1** | Le gestionnaire du parking reçoit la demande pour supprimer une place de parking visiteur |
| **2** | Le gestionnaire accède sa console de gestion du parking |
| **3** | Le gestionnaire demande au système de supprimer une place visiteur |
| **4** | Le système met à jour le nombre de place |
| **Post-conditions** | Une place de parking visiteur est supprimée |
| **Pré-conditions** | une demande pour réimprimer une carte été formulée |
| **Scénario nominale** | **Réimprimer une carte salariée** |
| **1** | Le gestionnaire du parking reçoit la demande pour réimprimer une place de parking salarie |
| **2** | Le gestionnaire accède sa console de gestion du parking |
| **3** | Le gestionnaire demande au système de réimprimer une carte parking salarie |
| **4** | Le système imprime la carte |
| **Post-conditions** | Une place de parking visiteur est supprimée |
| **Pré-conditions** | le gérant veux voir les stationnements |
| 1 | Le gestionnaire du parking accède aux stationnements des visiteurs via la console de gestion |
| 2 | La console retourne tous les stationnements visiteurs en cours |
| 1 | Le gestionnaire du parking accède aux stationnements des salariés via la console de gestion |
| 2 | La console retourne tous les stationnements des salariés en cours |
| 1 | Le gestionnaire du parking accède aux stationnements des historiques |
| 2 | Le gestionnaire sélectionne salariés ou visiteurs |
| 3 | Le gestionnaire sélectionne la date des stationnements souhaités et valide |
| 4 | La console retourne les stationnements de la date indiquée |

## Maquette des IHM de l’application parking

### Maquette de l’application de gestion du parking

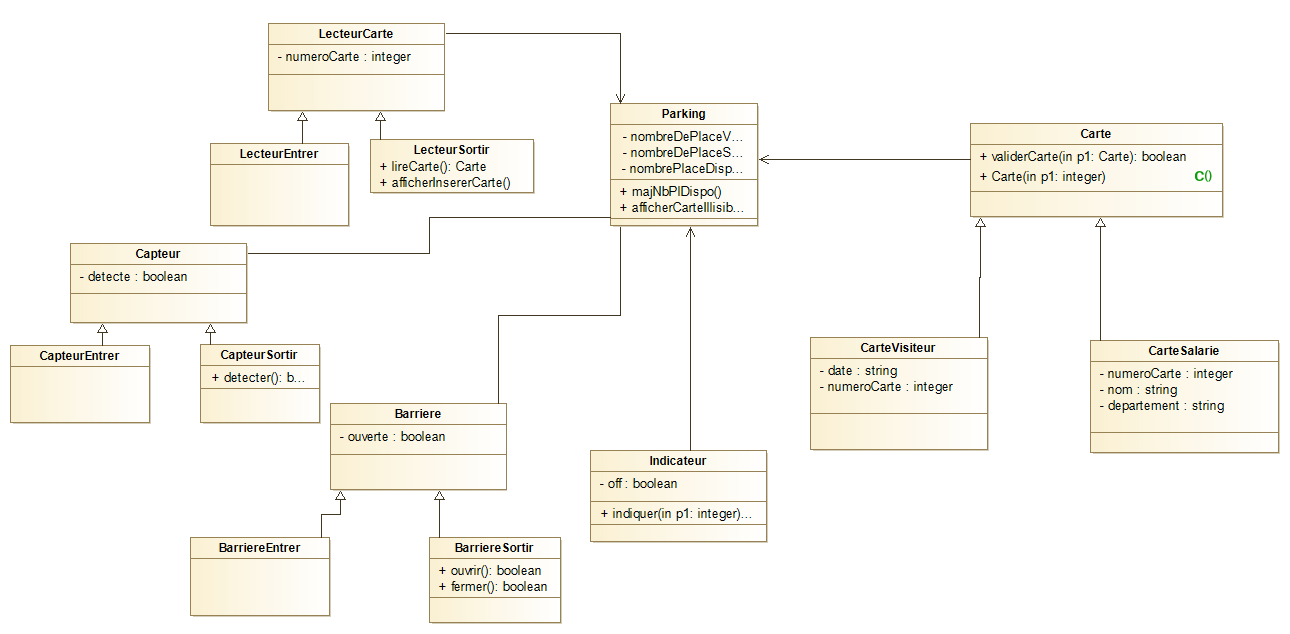


### Maquette du simulateur du parking



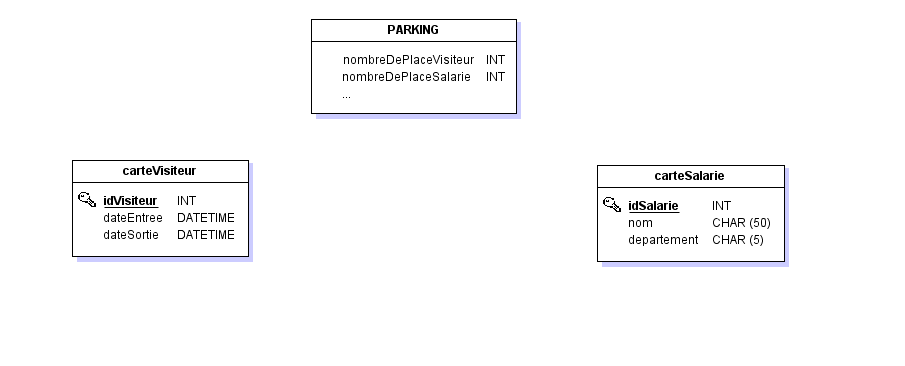
## Diagramme de classe du domaine

Première abstraction pour décrire la structure interne du système qui est composé par les classes reliés entre elle.



### Modélisation de la persistance des données (MCD)

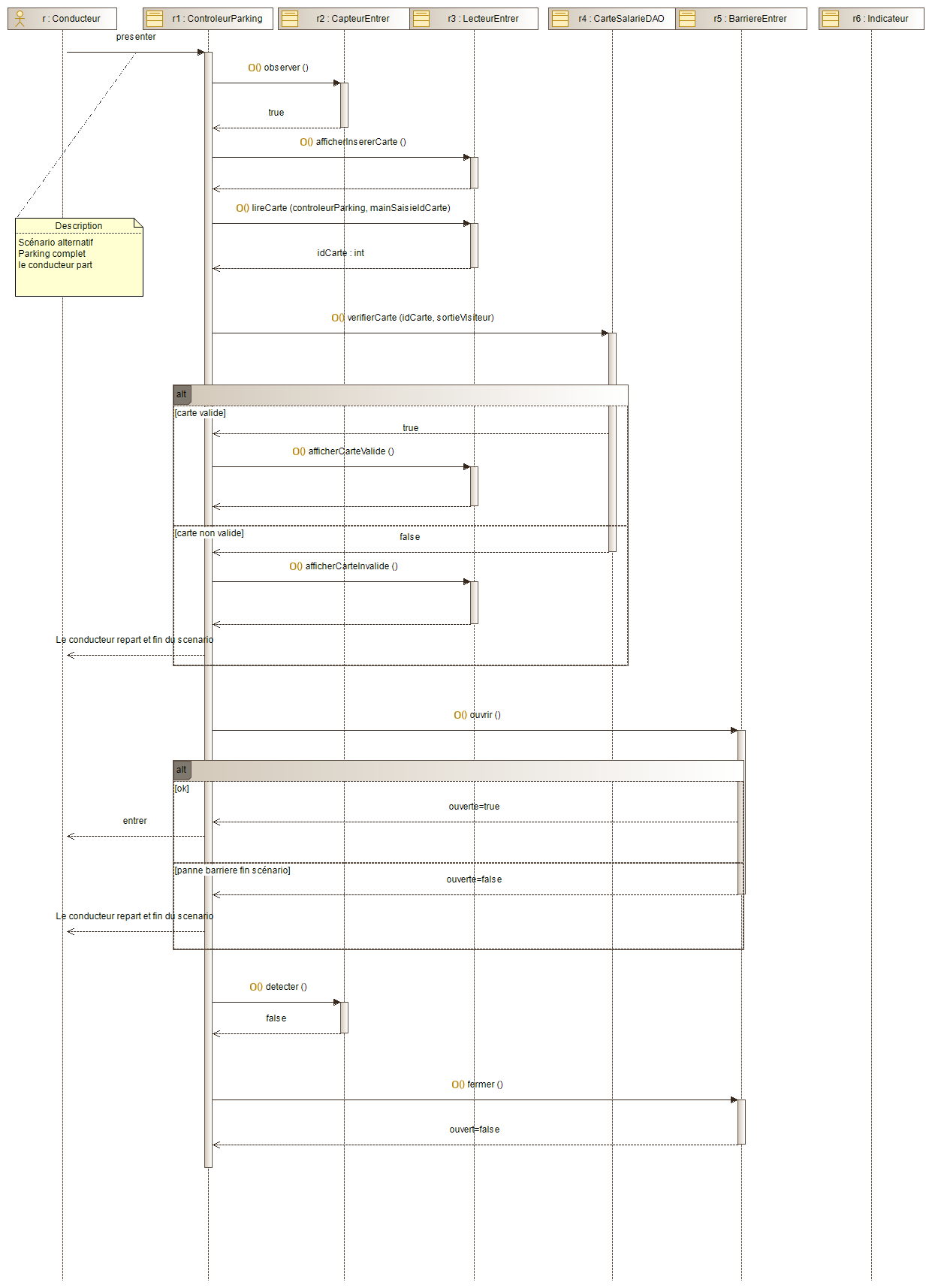
Les places de parking ne sont pas numérotées et ne peuvent donc pas être appareillées.



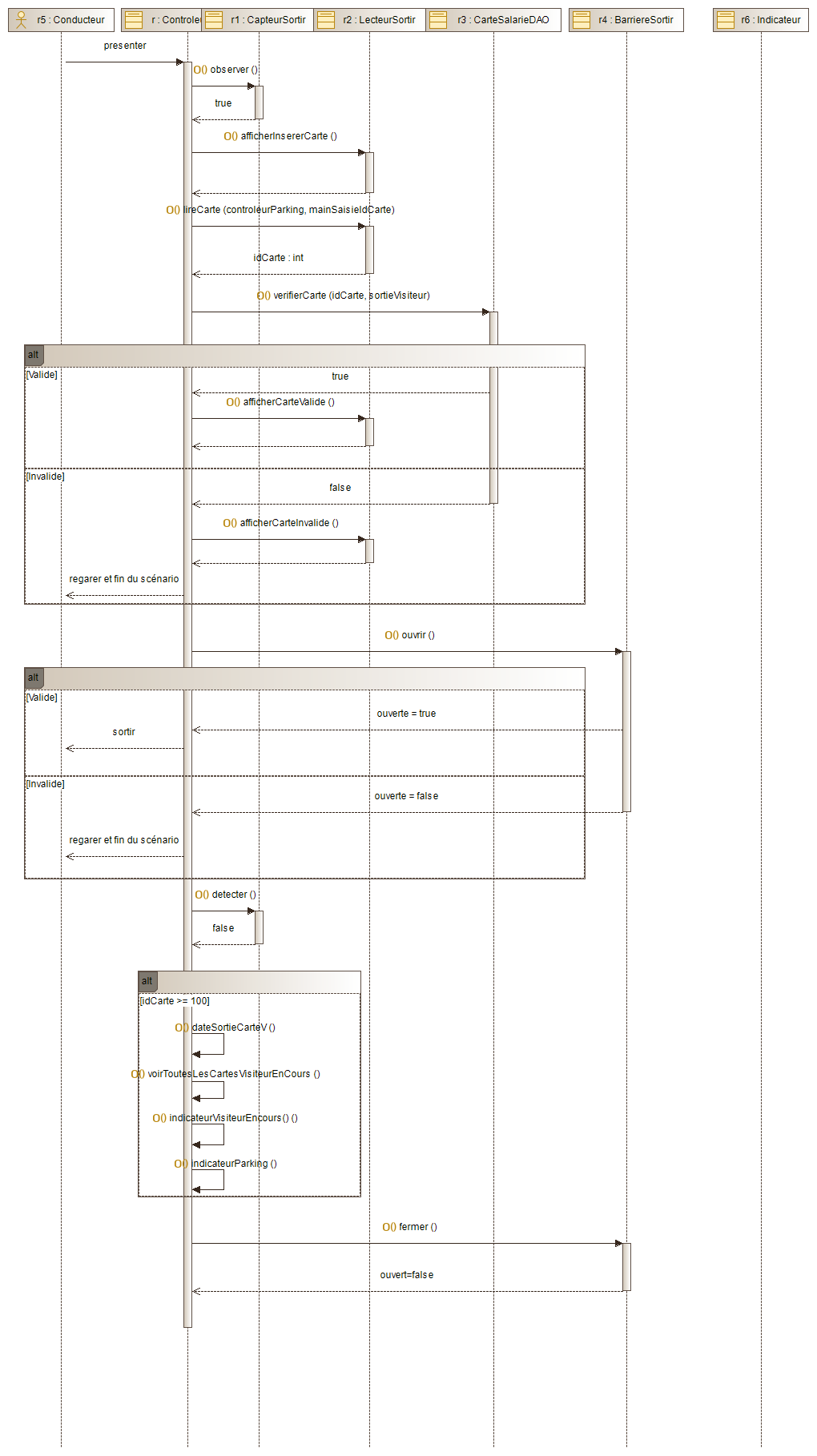
# Conception

## Diagramme de séquence (traduisant le scénario de l’UC détaillé)

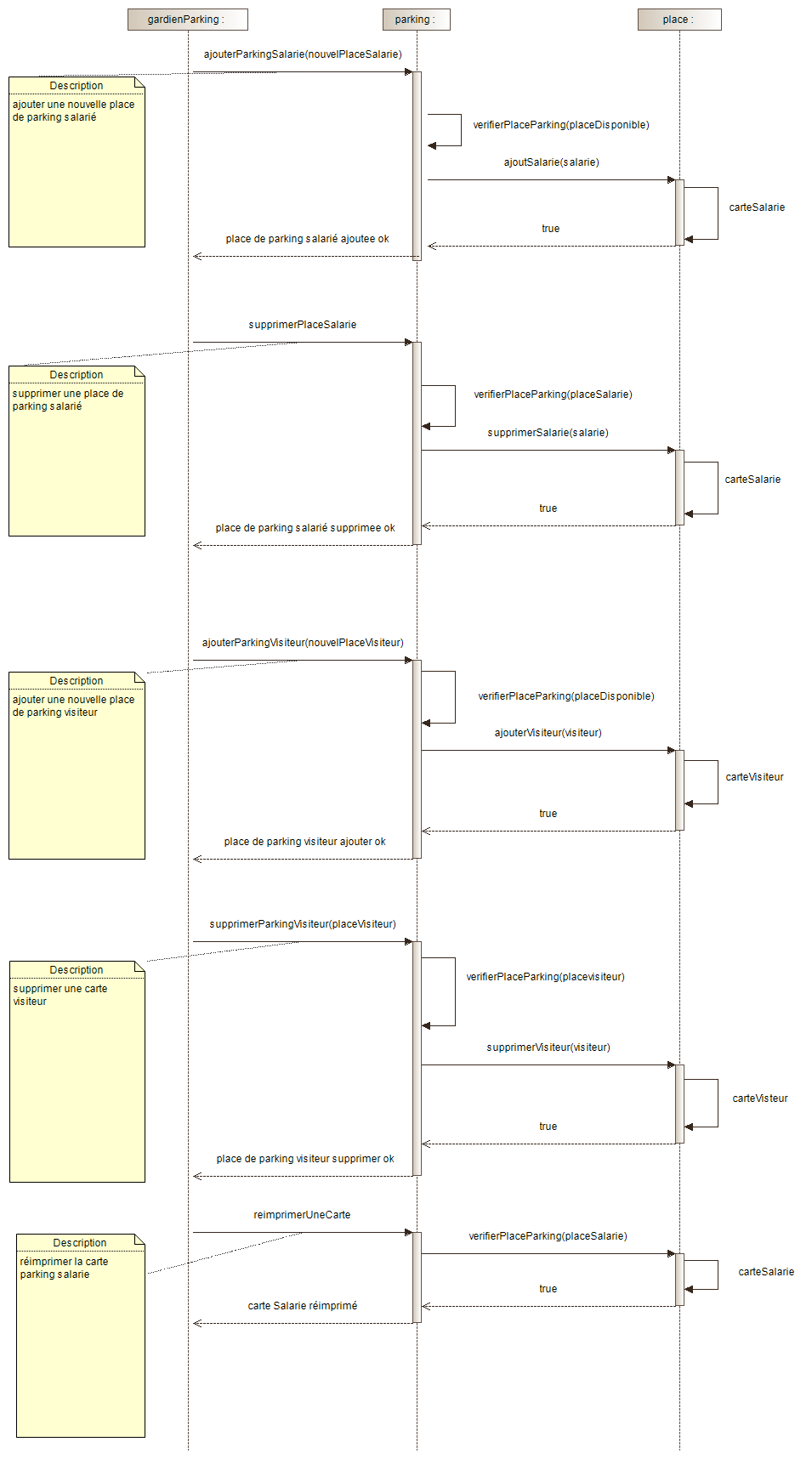
### Diagramme de séquence interaction entrer dans le parking



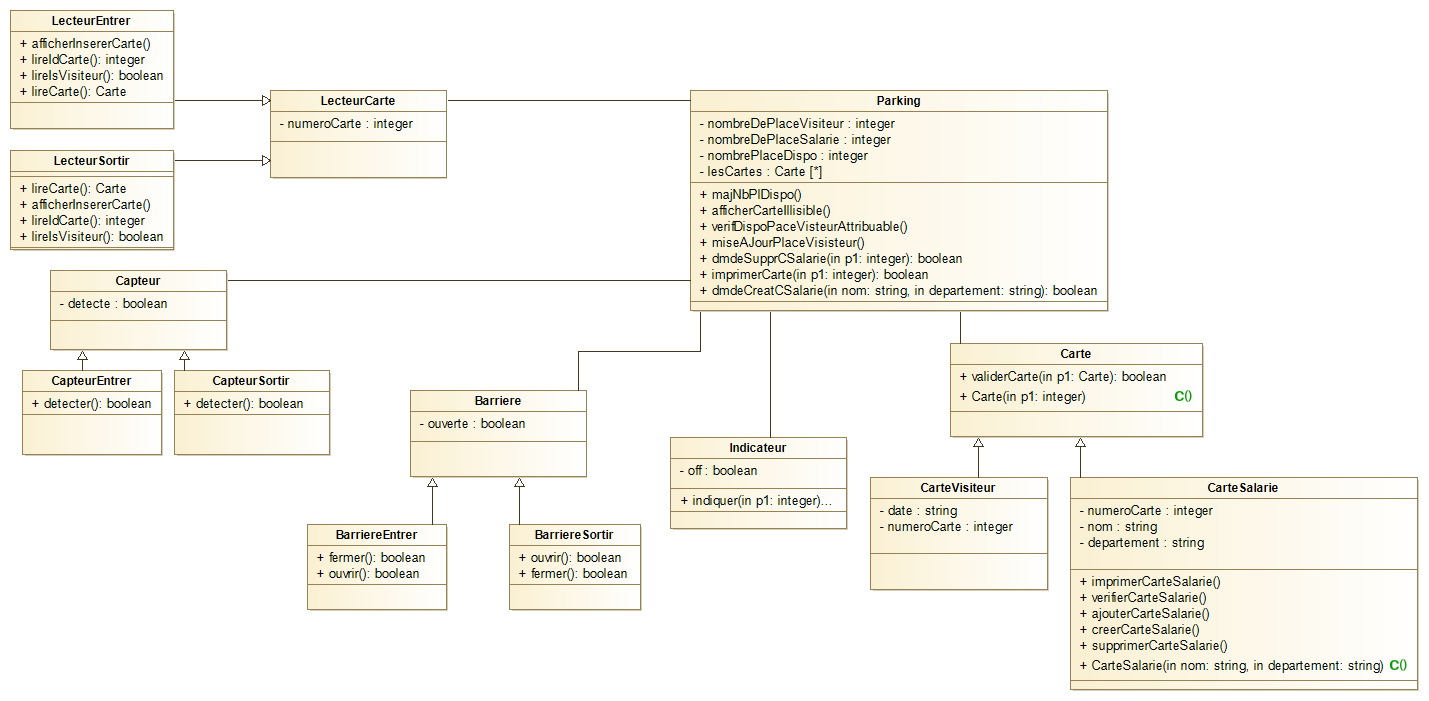
### Diagramme de séquence interaction sortir du parking



### Diagramme de séquence interaction gérer le parking

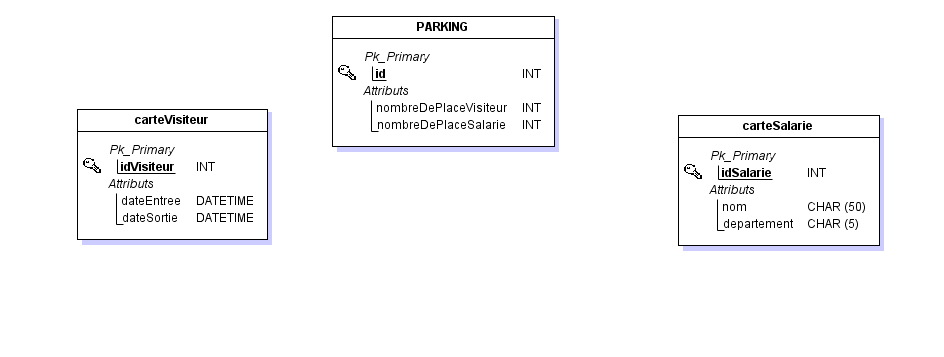


## Diagramme de classe de conception (ajout des méthodes)



### Création du MLD

Illustration du MLD :



### Création de la base de données

Script SQL de la base de données.

#------------------------------------------------------------

# Script MySQL.

#------------------------------------------------------------

#------------------------------------------------------------

# DataBASE: application\_parking

#------------------------------------------------------------

CREATE DATABASE IF NOT EXISTS application\_parking;

#------------------------------------------------------------

# Table: PARKING

#------------------------------------------------------------

CREATE TABLE PARKING(

id Int NOT NULL ,

nombreDePlaceVisiteur Int NOT NULL ,

nombreDePlaceSalarie Int NOT NULL

,CONSTRAINT PARKING\_PK PRIMARY KEY (id)

)ENGINE=InnoDB;

#------------------------------------------------------------

# Table: carteSalarie

#------------------------------------------------------------

CREATE TABLE carteSalarie(

idSalarie Int NOT NULL ,

nom Char (50) NOT NULL ,

departement Char (5) NOT NULL

,CONSTRAINT carteSalarie\_PK PRIMARY KEY (idSalarie)

)ENGINE=InnoDB;

#------------------------------------------------------------

# Table: carteVisiteur

#------------------------------------------------------------

CREATE TABLE carteVisiteur(

idVisiteur Int NOT NULL ,

date Date NOT NULL ,

dateEntree Date NOT NULL ,

dateSortie Date NOT NULL

,CONSTRAINT carteVisiteur\_PK PRIMARY KEY (idVisiteur)

)ENGINE=InnoDB;

### Architecture MVC avec la Technologies logicielle ORM JPA et DAO

La classe MainParking est la classe principale qui lance l'application, c'est le point d'entrée de l'application. Elle lance la fenêtre principale de l’application et son contrôleur.

En appuyant sur le bouton approprié, elle lance le MainCarteSalarie, la classe qui permet d'initialiser la boite de dialogue pour l'ajout d'une carte salarie. Cette fenêtre bloque les évènements d’être délivrés vers la fenêtre principale jusqu’à sa fermeture.

En appuyant sur le bouton approprié, elle lance le MainImpression, la classe qui permet d'afficher la fenêtre d'impression d'une carte. Cette fenêtre bloque les évènements d’être délivrés vers la fenêtre principale jusqu’à sa fermeture.

En appuyant sur le bouton approprié, elle lance la MainSimulateur, la classe qui sert à lancer la fenêtre et le contrôleur du simulateur. Cette dernière ne bloque pas les évènements destinés à la fenêtre principale.

Ce MainSimulateur instancie l’objet de la classe MainSaisieIdCarte qui est la classe qui sert à lancer la fenêtre et le contrôleur de la fenêtre de saisie du numéro de la carte. Cette dernière fenêtre bloque les événements d’être délivrés vers les fenêtres principales (MainParking et Simulateur).

Lors de la configuration de la connexion eclipse-link (JPA) à la base de données MySQL nous lui demandons de construire le modèle ORM (Mapping Objet-Relationnel : une classe correspond à une entité) à partir des tables pour la persistance des données. Nous rajoutons des surdéfinitions (exemple : surdéfinition de la méthode .toString()).

Pour la DAO nous écrivons nous même les méthodes, JPA fourni la classe EntityManager et les méthodes CRUD (Create, Read, Update, Delete).

### Éléments graphiques (conception)

Vue du fichier FXML de la vue de la console de gestion du parking.

<?xml version=*"1.0"* encoding=*"UTF-8"*?>

<?import javafx.geometry.Insets?>

<?import javafx.scene.control.Button?>

<?import javafx.scene.control.Label?>

<?import javafx.scene.control.ListView?>

<?import javafx.scene.layout.ColumnConstraints?>

<?import javafx.scene.layout.GridPane?>

<?import javafx.scene.layout.RowConstraints?>

<?import javafx.scene.text.Font?>

<GridPane alignment=*"CENTER"* minWidth=*"900.0"* prefHeight=*"600.0"* xmlns=*"http://javafx.com/javafx/8.0.171"* xmlns:fx=*"http://javafx.com/fxml/1"* fx:controller=*"controleur.ControleurParking"*>

<columnConstraints>

<ColumnConstraints />

<ColumnConstraints minWidth=*"200.0"* />

<ColumnConstraints minWidth=*"100.0"* />

<ColumnConstraints minWidth=*"10.0"* prefWidth=*"70.0"* />

<ColumnConstraints minWidth=*"220.0"* />

<ColumnConstraints minWidth=*"80.0"* />

<ColumnConstraints minWidth=*"10.0"* prefWidth=*"200.0"* />

<ColumnConstraints minWidth=*"10.0"* prefWidth=*"200.0"* />

</columnConstraints>

<rowConstraints>

<RowConstraints />

<RowConstraints />

<RowConstraints minHeight=*"10.0"* prefHeight=*"30.0"* />

<RowConstraints minHeight=*"10.0"* prefHeight=*"30.0"* />

<RowConstraints minHeight=*"10.0"* prefHeight=*"30.0"* />

<RowConstraints minHeight=*"10.0"* prefHeight=*"30.0"* />

<RowConstraints minHeight=*"10.0"* prefHeight=*"30.0"* />

<RowConstraints minHeight=*"10.0"* prefHeight=*"30.0"* />

<RowConstraints minHeight=*"10.0"* prefHeight=*"30.0"* />

<RowConstraints minHeight=*"10.0"* prefHeight=*"30.0"* />

<RowConstraints minHeight=*"10.0"* prefHeight=*"30.0"* />

<RowConstraints minHeight=*"10.0"* prefHeight=*"30.0"* />

<RowConstraints minHeight=*"10.0"* prefHeight=*"30.0"* />

<RowConstraints minHeight=*"10.0"* prefHeight=*"30.0"* />

<RowConstraints minHeight=*"10.0"* prefHeight=*"30.0"* />

<RowConstraints minHeight=*"10.0"* prefHeight=*"30.0"* />

<RowConstraints minHeight=*"10.0"* prefHeight=*"30.0"* />

<RowConstraints minHeight=*"10.0"* prefHeight=*"30.0"* />

<RowConstraints minHeight=*"10.0"* prefHeight=*"30.0"* />

<RowConstraints minHeight=*"10.0"* prefHeight=*"30.0"* />

<RowConstraints minHeight=*"10.0"* prefHeight=*"30.0"* />

</rowConstraints>

<children>

<Label prefHeight=*"17.0"* prefWidth=*"263.0"* text=*"Application Parking"* GridPane.columnIndex=*"2"* GridPane.columnSpan=*"3"* GridPane.rowIndex=*"1"*>

<font>

<Font name=*"System Bold"* size=*"18.0"* />

</font>

</Label>

<Label prefHeight=*"17.0"* prefWidth=*"162.0"* text=*"Liste Salarie"* GridPane.columnIndex=*"1"* GridPane.columnSpan=*"2"* GridPane.rowIndex=*"8"*>

<font>

<Font name=*"System Bold"* size=*"13.0"* />

</font>

</Label>

<Label prefHeight=*"17.0"* prefWidth=*"221.0"* text=*"Liste Visiteur En Cours"* GridPane.columnIndex=*"4"* GridPane.columnSpan=*"2"* GridPane.rowIndex=*"8"*>

<font>

<Font name=*"System Bold"* size=*"13.0"* />

</font>

</Label>

<ListView fx:id=*"lesCartesSalaries"* prefHeight=*"200.0"* prefWidth=*"200.0"* GridPane.columnIndex=*"1"* GridPane.columnSpan=*"2"* GridPane.rowIndex=*"9"* GridPane.rowSpan=*"7"*>

<GridPane.margin>

<Insets bottom=*"10.0"* left=*"10.0"* right=*"10.0"* top=*"10.0"* />

</GridPane.margin>

</ListView>

<ListView fx:id=*"lesCartesVisiteursEnCours"* prefHeight=*"200.0"* prefWidth=*"200.0"* GridPane.columnIndex=*"4"* GridPane.columnSpan=*"2"* GridPane.rowIndex=*"9"* GridPane.rowSpan=*"7"*>

<GridPane.margin>

<Insets bottom=*"10.0"* left=*"10.0"* right=*"10.0"* top=*"10.0"* />

</GridPane.margin>

</ListView>

<Button mnemonicParsing=*"false"* onAction=*"#ajouterUneCarteSalarie"* prefHeight=*"25.0"* prefWidth=*"191.0"* text=*"Créer Carte Salarie"* GridPane.columnIndex=*"1"* GridPane.columnSpan=*"2"* GridPane.rowIndex=*"17"*>

<GridPane.margin>

<Insets bottom=*"10.0"* left=*"10.0"* right=*"10.0"* top=*"10.0"* />

</GridPane.margin>

</Button>

<Button mnemonicParsing=*"false"* onAction=*"#dmdeCreatCVsiteur"* prefHeight=*"25.0"* prefWidth=*"193.0"* text=*"Créér Carte Visiteur"* GridPane.columnIndex=*"4"* GridPane.columnSpan=*"2"* GridPane.rowIndex=*"17"*>

<GridPane.margin>

<Insets bottom=*"10.0"* left=*"10.0"* right=*"10.0"* top=*"10.0"* />

</GridPane.margin>

</Button>

<Button mnemonicParsing=*"false"* onAction=*"#imprimerCarteSalarie"* prefHeight=*"25.0"* prefWidth=*"191.0"* text=*"Imprimer Carte Salarie"* GridPane.columnIndex=*"1"* GridPane.columnSpan=*"2"* GridPane.rowIndex=*"19"*>

<GridPane.margin>

<Insets bottom=*"10.0"* left=*"10.0"* right=*"10.0"* top=*"10.0"* />

</GridPane.margin>

</Button>

<Button mnemonicParsing=*"false"* onAction=*"#imprimerCarteVisiteur"* prefHeight=*"25.0"* prefWidth=*"193.0"* text=*"Imprimer Carte Visiteur"* GridPane.columnIndex=*"4"* GridPane.columnSpan=*"2"* GridPane.rowIndex=*"19"*>

<GridPane.margin>

<Insets bottom=*"10.0"* left=*"10.0"* right=*"10.0"* top=*"10.0"* />

</GridPane.margin>

</Button>

<Button mnemonicParsing=*"false"* onAction=*"#supprimerCarteSalarie"* prefHeight=*"25.0"* prefWidth=*"184.0"* text=*"Supprimer Carte Salarie "* GridPane.columnIndex=*"1"* GridPane.columnSpan=*"2"* GridPane.rowIndex=*"18"*>

<GridPane.margin>

<Insets bottom=*"10.0"* left=*"10.0"* right=*"10.0"* top=*"10.0"* />

</GridPane.margin>

</Button>

<Label text=*"Historique des visiteurs"* GridPane.columnIndex=*"6"* GridPane.rowIndex=*"8"*>

<font>

<Font name=*"System Bold"* size=*"13.0"* />

</font>

</Label>

<ListView fx:id=*"lesCartesVisiteursHistorique"* prefHeight=*"200.0"* prefWidth=*"200.0"* GridPane.columnIndex=*"6"* GridPane.columnSpan=*"2"* GridPane.rowIndex=*"9"* GridPane.rowSpan=*"7"*>

<GridPane.margin>

<Insets bottom=*"10.0"* left=*"10.0"* right=*"10.0"* top=*"10.0"* />

</GridPane.margin>

<padding>

<Insets bottom=*"10.0"* left=*"10.0"* right=*"10.0"* top=*"10.0"* />

</padding>

</ListView>

<Label text=*"Nombre de place total du parking"* GridPane.columnIndex=*"1"* GridPane.rowIndex=*"3"* GridPane.valignment=*"CENTER"* />

<Label fx:id=*"nombreTotalPlaceParking"* text=*"Label"* GridPane.columnIndex=*"2"* GridPane.rowIndex=*"3"* />

<Label text=*"Nombre de place de parking salariés"* GridPane.columnIndex=*"1"* GridPane.rowIndex=*"5"* GridPane.valignment=*"CENTER"* />

<Label fx:id=*"nombreTotalPlaceSalarie"* text=*"Label"* GridPane.columnIndex=*"2"* GridPane.rowIndex=*"5"* />

<Label text=*"Nombre de place réservé aux visiteurs"* GridPane.columnIndex=*"4"* GridPane.rowIndex=*"5"* />

<Label fx:id=*"nombreTotalPlaceVisiteur"* contentDisplay=*"CENTER"* text=*"Label"* GridPane.columnIndex=*"5"* GridPane.rowIndex=*"5"* GridPane.valignment=*"CENTER"* />

<Label fx:id=*"etatParkingVisiteur"* text=*"Etat du parking Visiteur"* GridPane.columnIndex=*"4"* GridPane.columnSpan=*"2"* GridPane.rowIndex=*"3"*>

<font>

<Font name=*"System Bold"* size=*"18.0"* />

</font></Label>

<Label text=*"Nombre de carte visiteur en cours"* GridPane.columnIndex=*"4"* GridPane.rowIndex=*"6"* />

<Label fx:id=*"nombreTotalvisiteurEnCours"* contentDisplay=*"CENTER"* text=*"Label"* GridPane.columnIndex=*"5"* GridPane.rowIndex=*"6"* GridPane.valignment=*"CENTER"* />

<Label text=*"Nombre de carte salarie en cours"* GridPane.columnIndex=*"1"* GridPane.rowIndex=*"6"* />

<Label fx:id=*"nombreTotalPlaceSalarieEncours"* text=*"Label"* GridPane.columnIndex=*"2"* GridPane.rowIndex=*"6"* />

<Button mnemonicParsing=*"false"* onAction=*"#simulateur"* text=*"Simulateur"* textFill=*"BLUE"* GridPane.columnIndex=*"6"* GridPane.halignment=*"CENTER"* GridPane.rowIndex=*"18"* />

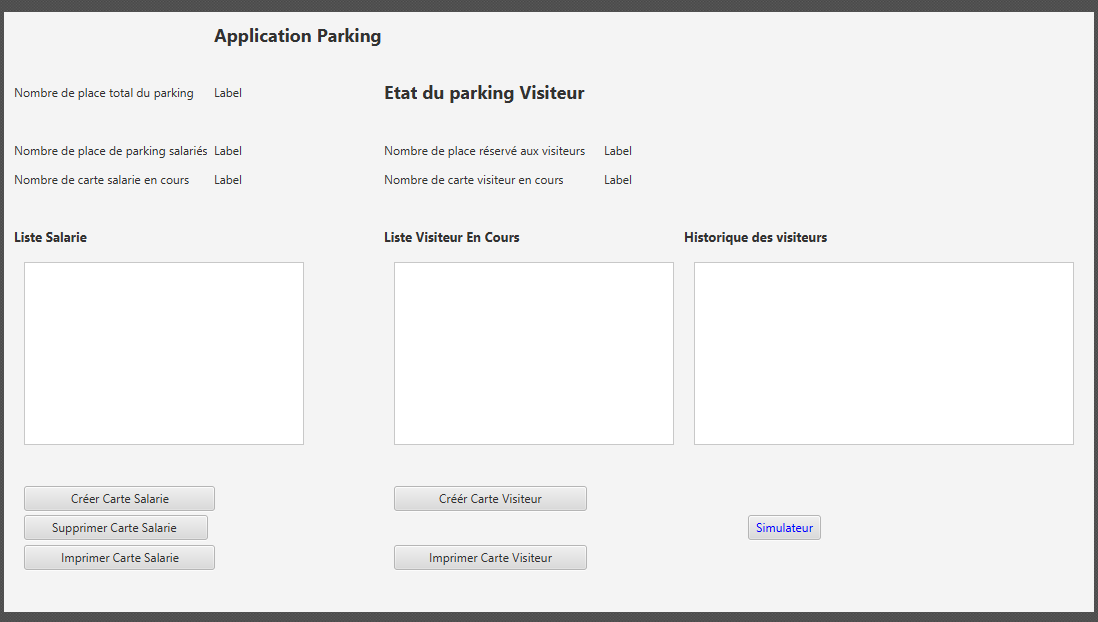
</children>

<padding>

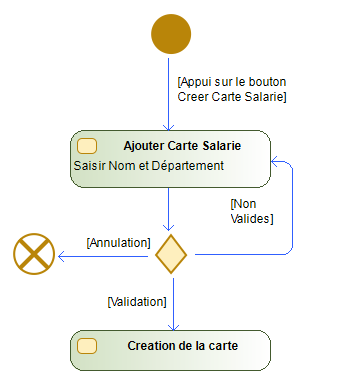
<Insets bottom=*"10.0"* left=*"10.0"* right=*"10.0"* top=*"10.0"* />

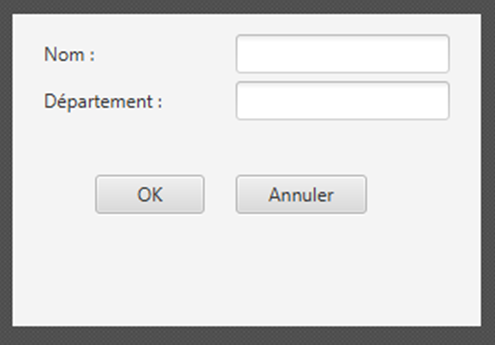
</padding>

</GridPane>

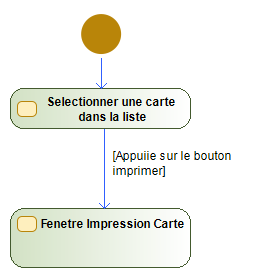
Vue affichée avec Scene Builder de la console de gestion du parking.

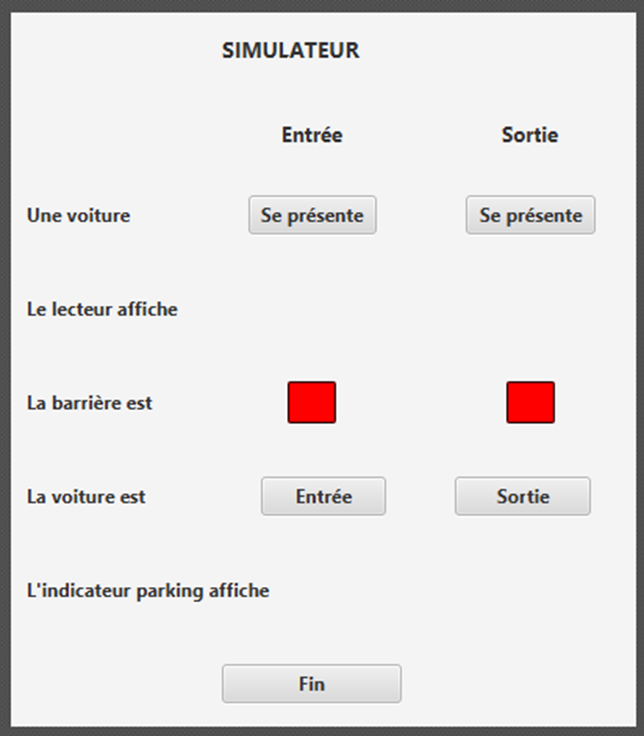
Appuie sur le bouton Créer Carte salarie

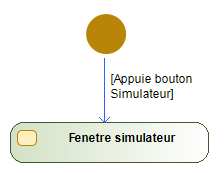




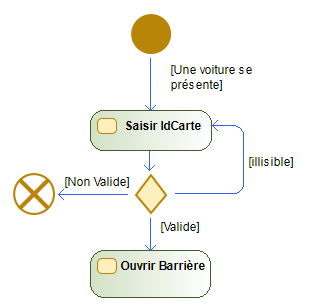
Appuie sur un des boutons Imprimer

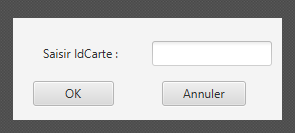


Appuie sur le bouton simulateur



Une voiture se présente





### Environnement logiciel utilisé.

Environnement serveur Xampp-portable-Windows-x64-7.4.5-0-VC15-installer

Phpmyadmin V5 pour l’administration de la BDD.

Base de données Maria DB 10.4.11

**AGL - Atelier –Génie logiciel –**

JMerise : pour la création MCD

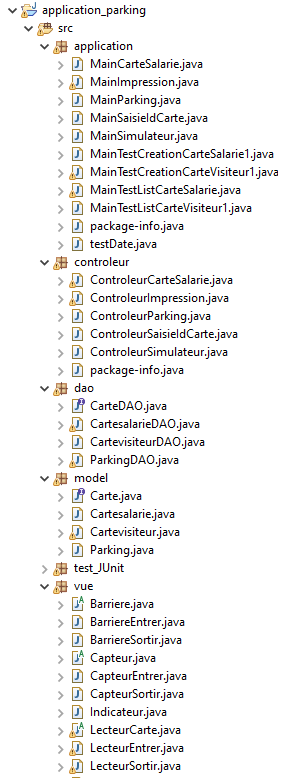
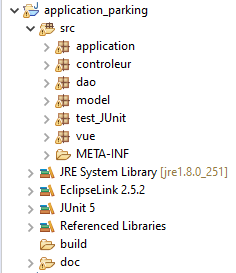
NotePad++ pour le Script SQL

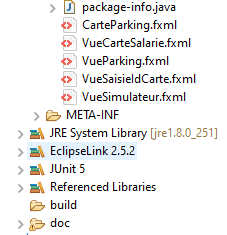
Pencil : Pour les maquettes

IDE eclipse

Langage JAVA JEE

### Arborescence projet du code



# Les tests

## Tests unitaires

### Tests JUnit

Exemple d’un test unitaire de la classe DAO Cartevisisteur pour la création d’une carte visiteur.

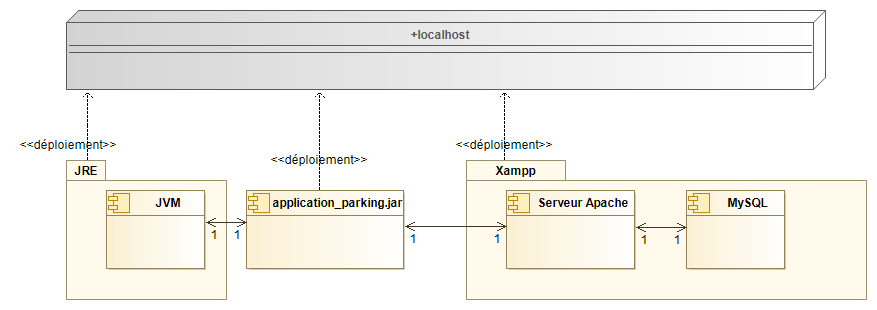


### Tests d’intégrations

Exemple d’un test d’intégration pour la création d’une carte visiteur.



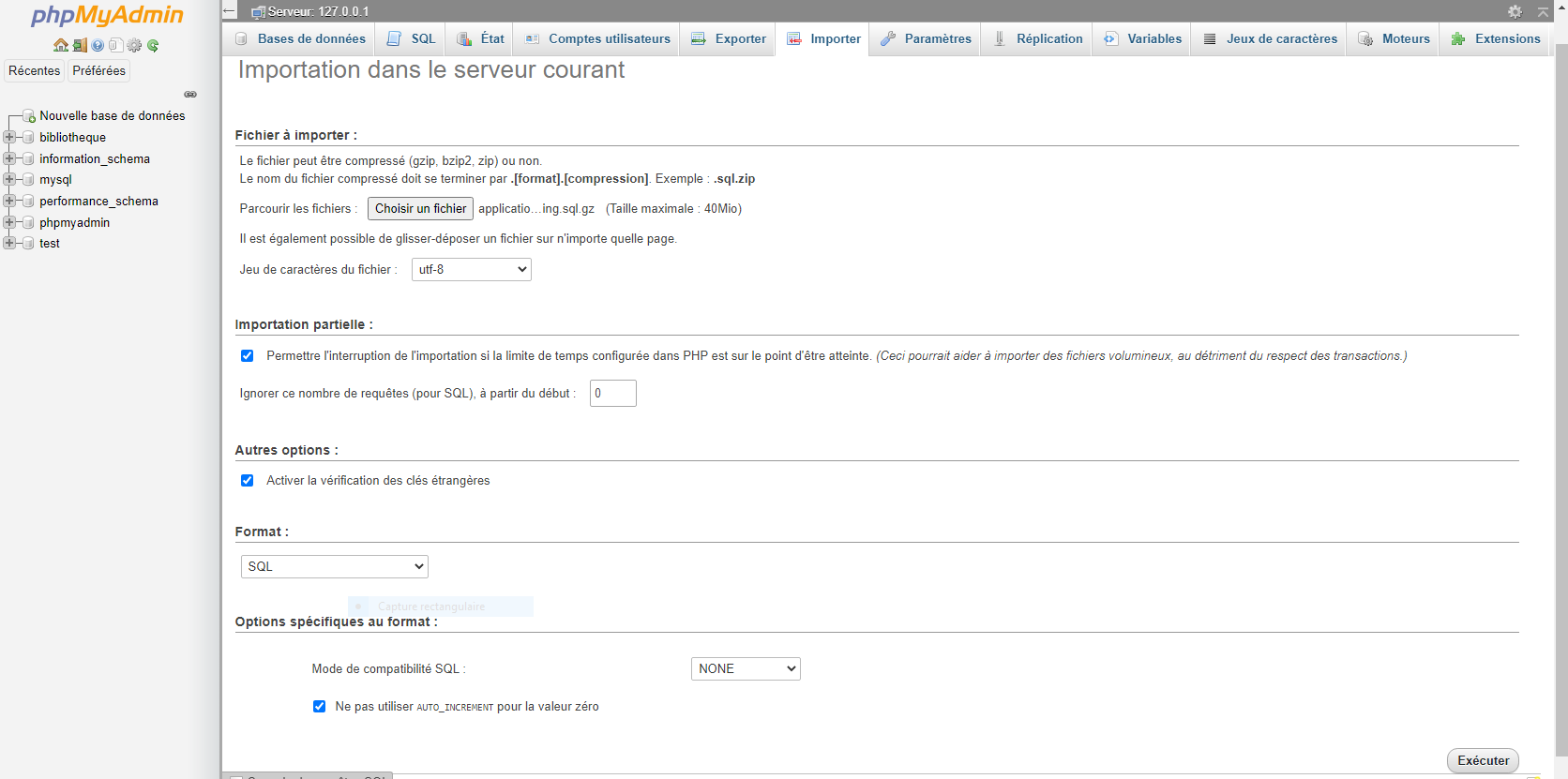
# Déploiement



Il faut tout d’abord installer le JRE.

Puis installer Xampp.

Ensuite importer la base de données jointe dans mySQL : Choisissez le fichier puis faire exécuter.



Par la suite créer l’utilisateur « admin » avec le mot de passe « admin » dans MySQL.

Après copier sur votre poste le fichier application\_parking.jar avec le répertoire application\_parking\_lib et exécuter l’application.

Pour lancer le simulateur appuier sur le bouton simulateur qui peut fonctionner indépendamment de la console de gestion.

# Bilan et perspective

Ce projet nous a permis de travailler en collaboration grâce au fil conducteur de la conception UML qui fourni un cadre de développement.

Une itération future prévoira l’utilisation d’une technologie API REST (Payara, JEE) permettant une meilleur sécurité (librairies) contre l’injection SQL.

Le frein a été constitué par le temps passé à consulter les documentations pour apprendre les méthodes. Par exemple la classe Date comporte des méthodes dépréciées et nous avons décidé de coder qu’avec des méthodes pérennes.