

EMSI

# Java Avancé

Réalisé par Loubna Ait-Hra

---

## Table des matières :

1	Introduction Générale .....	1
1.1	Contexte du projet.....	1
1.2	Problématique .....	1
1.3	Objectifs.....	1
2	Partie I : Analyse et Conception .....	1
2.1	Spécification des besoins .....	1
2.1.1	Besoins Fonctionnels.....	1
2.1.2	Besoins Non-Fonctionnels .....	1
2.2	Conception UML.....	2
2.2.1	Diagramme de Classes.....	2
2.2.2	Explication Générale.....	2
2.3	Conception de la Base de Données.....	3
3	Environnement Technique .....	3
4	Architecture et Implémentation.....	3
4.1	Architecture logicielle (MVC).....	3
4.2	Design Patterns.....	3
4.3	Extraits de code clés .....	3
5	Interface Utilisateur et Tests.....	4
5.1	Présentation des interfaces .....	4
5.2	Scénarios de Test .....	13
6	Conclusion et Perspectives.....	14
7	Webographie / Bibliographie .....	15



---

## 1 Introduction Générale

### 1.1 Contexte du projet

Ce projet s'inscrit dans le cadre du module Java Avancé. Il vise à simuler un système moderne de gestion de parking urbain.

### 1.2 Problématique

La gestion manuelle des parkings entraîne des pertes de temps, une mauvaise optimisation de l'espace et une difficulté pour les conducteurs à trouver des places disponibles en temps réel.

### 1.3 Objectifs

- ✓ Authentification sécurisée avec rôles spécifiques (Admin/Conducteur).
- ✓ Gestion CRUD des infrastructures de parking.
- ✓ Système de réservation en temps réel avec suivi de disponibilité.
- ✓ Dashboard analytique pour l'administrateur (Places libres).

## 2 Partie I : Analyse et Conception

### 2.1 Spécification des besoins

#### 2.1.1 Besoins Fonctionnels

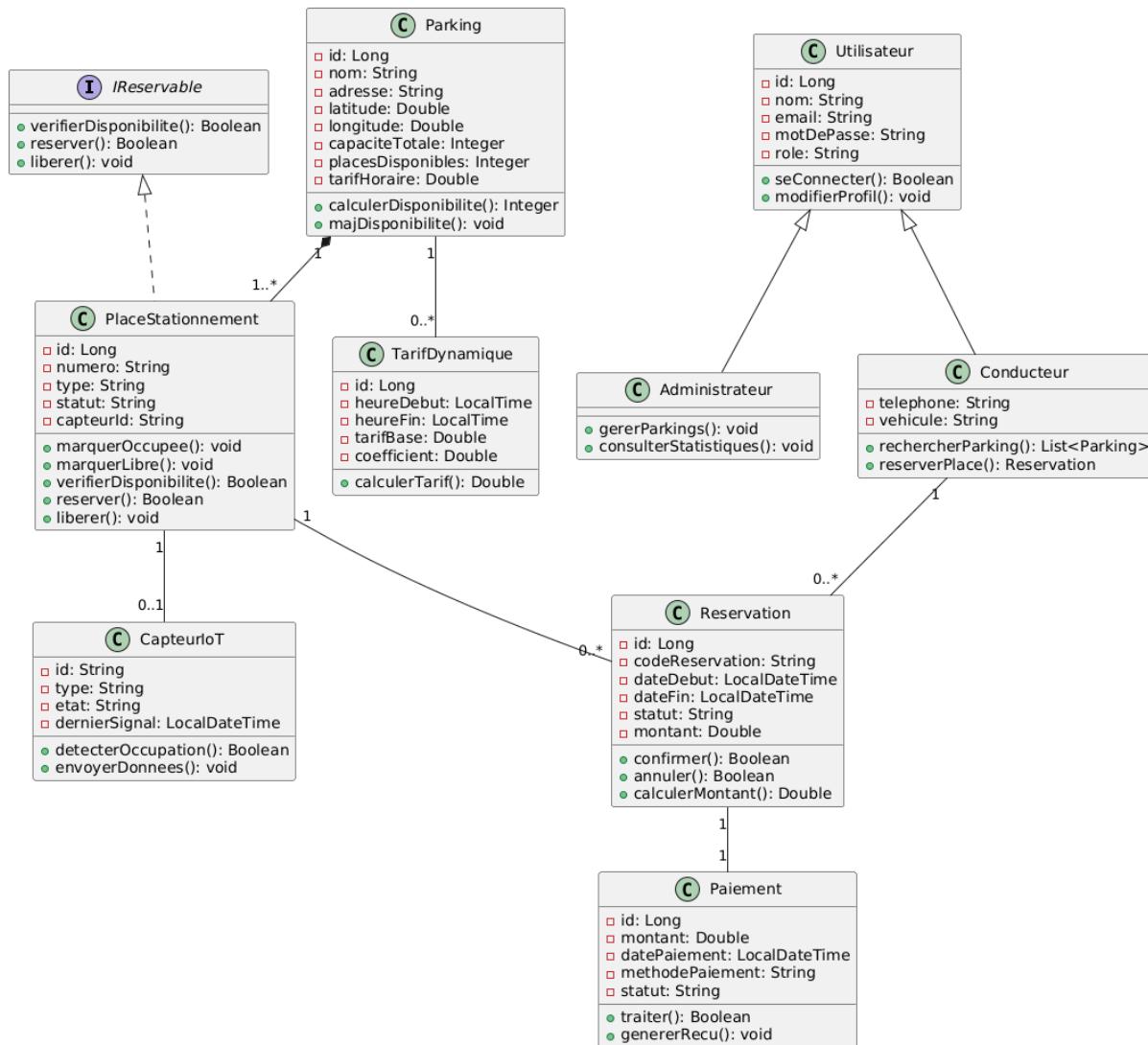
- ✓ Se connecter au système.
- ✓ Visualiser la liste des parkings et leur capacité libre.
- ✓ Réserver/Annuler une place (Conducteur).
- ✓ Gérer les parkings et capteurs (Admin).

#### 2.1.2 Besoins Non-Fonctionnels

- ✓ Persistance des données via Hibernate.
- ✓ Interface graphique intuitive sous JavaFX.
- ✓ Gestion automatique des comptes de test au démarrage.

## 2.2 Conception UML

### 2.2.1 Diagramme de Classes



### 2.2.2 Explication Générale

Le diagramme de classe structure le système autour de quatre piliers majeurs :

- ✓ **Gestion des Utilisateurs** : Un système d'héritage lie `Utilisateur` à `Administre` et `Conducteur`, permettant une gestion centralisée des accès.
- ✓ **Infrastructure de Parking** : La classe `Parking` agrège plusieurs `PlaceStationnement`. Chaque place est associée à un `CapteurIoT` pour le suivi en temps réel et à un `TarifDynamique` pour le calcul automatique des coûts.
- ✓ **Processus de Réservation** : L'interface `IReservable` garantit que les places respectent le cycle de vie (vérifier, réserver, libérer). La classe `Reservation` fait le pont entre le `Conducteur` et la `PlaceStationnement`.
- ✓ **Finalisation** : Chaque `Reservation` aboutit à un `Paiement` unique, assurant la traçabilité financière des transactions.

## 2.3 Conception de la Base de Données

**MLD :**Tables `Utilisateur`, `Administrateur`, `Conducteur`, `Parking`, `PlaceStationnement`, `Reservation`, `Paiement`, `CapteurIoT`.

**PK/FK :**Utilisation de l'Identity pour les IDs et de contraintes `ON DELETE CASCADE` pour les sous-classes.

## 3 Environnement Technique

- ✓ Langage : Java 17 (JDK).
- ✓ **IDE :** IntelliJ IDEA / VS Code.
- ✓ **Gestion de projet :** Maven (Gestion des dépendances Hibernate, MySQL Connector, JavaFX).
- ✓ **SGBD :**MySQL 8.0.
- ✓ **Bibliothèques tierces :**
  - ✓ Hibernate ORM (Persistance).
  - ✓ JavaFX (Interface graphique).
  - ✓ MySQL Connector (JDBC).

## 4 Architecture et Implémentation

### 4.1 Architecture logicielle (MVC)

Organisation des packages :

- `ma.emsi.entities` : Entités persistantes mappées par Hibernate (XML).
- `ma.emsi.ui` : Contrôleurs JavaFX (`LoginController`, `MainController`).
- `ma.emsi.util` : Utilitaires (`HibernateUtil`, `DatabaseSeeder`).
- `ma.emsi.resources` : Fichiers FXML et CSS.

### 4.2 Design Patterns

- **Singleton :** `HibernateUtil` garantit une instance unique de `SessionFactory`.
- **Session-per-request :** Ouverture/fermeture propre des sessions Hibernate dans les contrôleurs.
- **Data Seeding :** `DatabaseSeeder` assure la cohérence des comptes sans intervention manuelle.

### 4.3 Extraits de code clés

- Vérification de rôle personnalisée :

Java

```
if (user != null && user.getMotDePasse().equals(password)) {  
  
    String dbRole = user.getRole().toUpperCase();
```

```
App.setCurrentUser(user);
```

```
App.setRoot("MainView");}
```

- Calcul dynamique des places libres (HQL) :

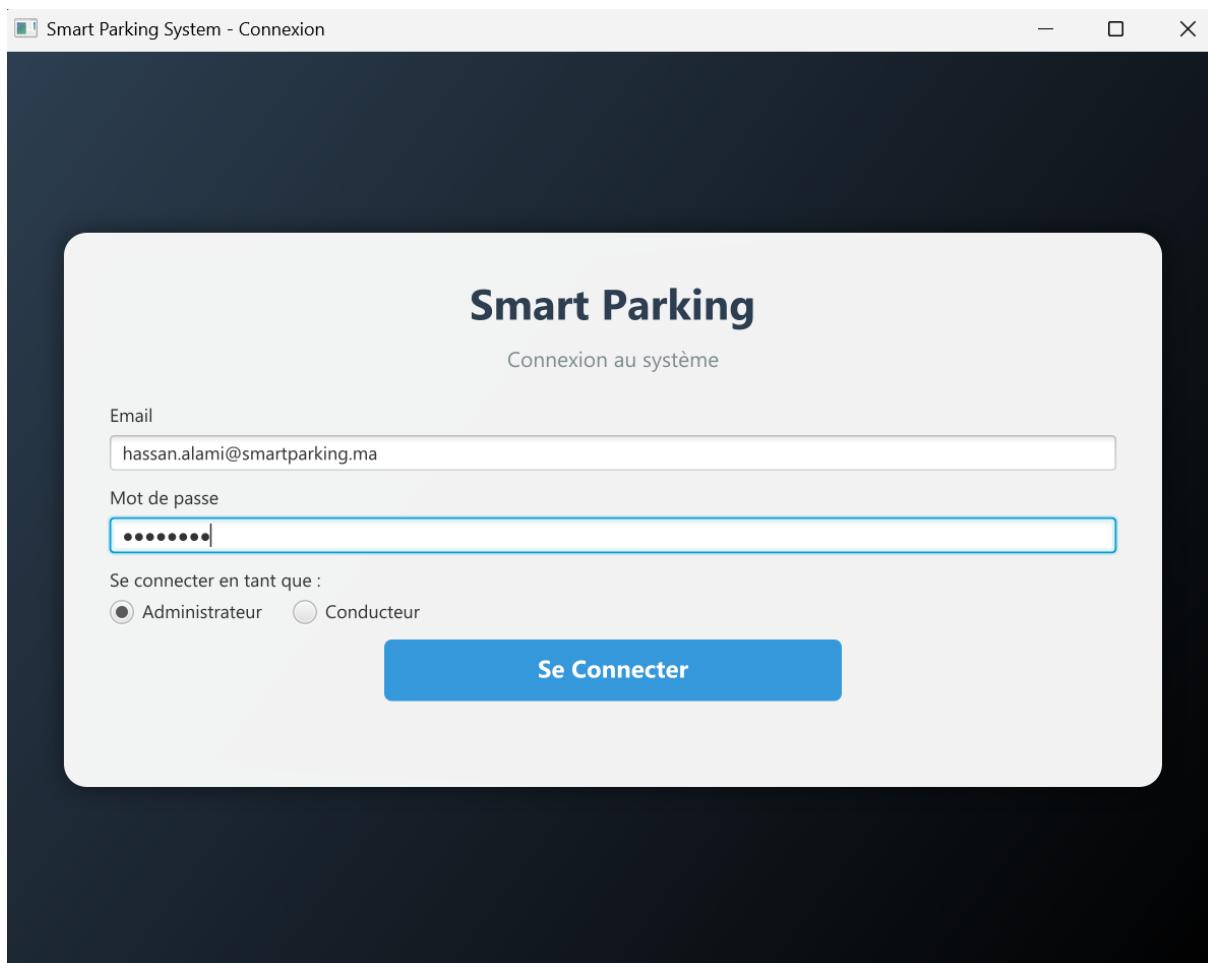
Java

```
session.createQuery("select count(p) from PlaceStationnement p where p.parking.id = :pid and p.estDisponible = true")
```

## 5 *Interface Utilisateur et Tests*

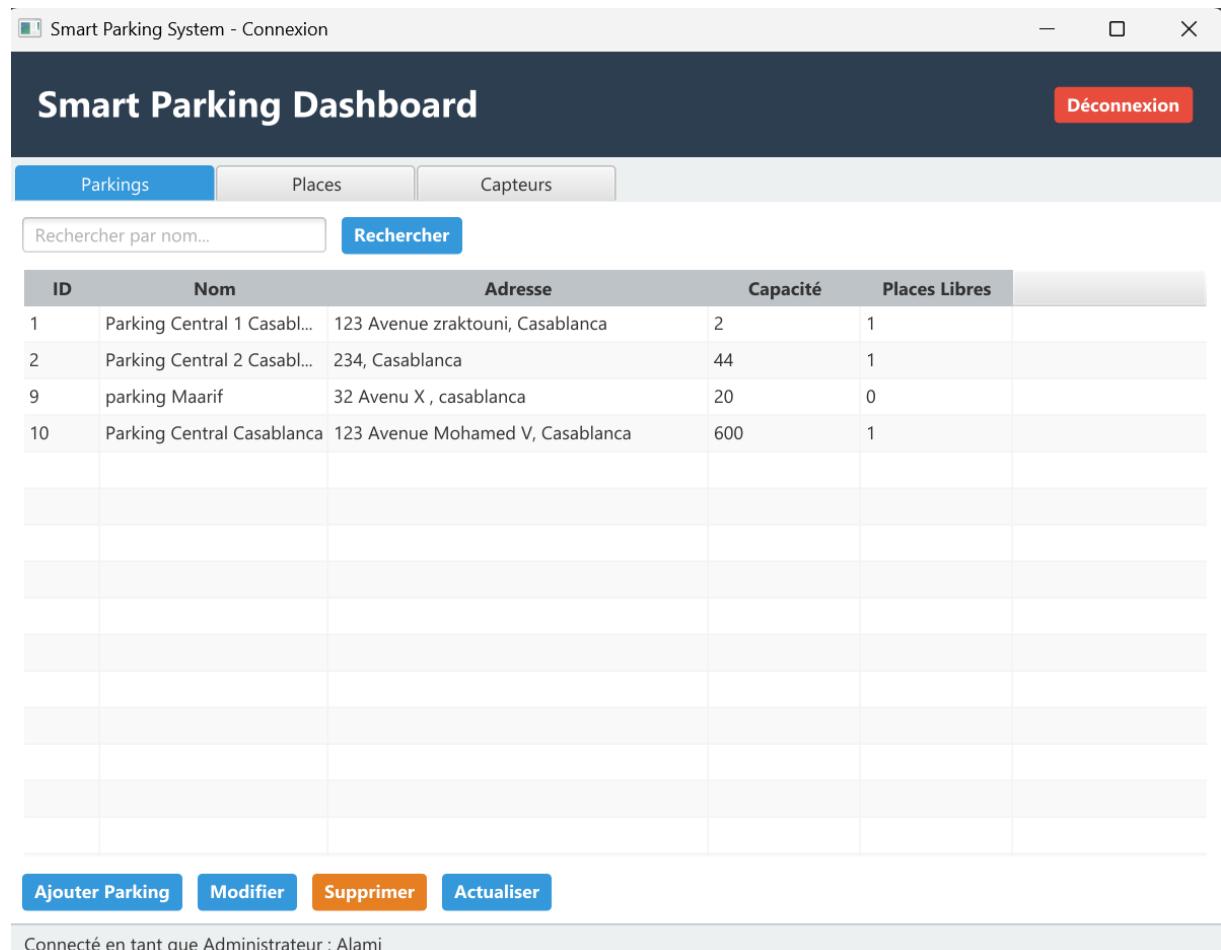
### 5.1 Présentation des interfaces

- Login



- **Description\*\*** : Premier point d'entrée de l'application permettant d'identifier l'utilisateur et son rôle.
- **Fonctionnalités :**
  - Saisie sécurisée de l'email et du mot de passe.

- Gestion personnalisée des erreurs (ex: "Utilisateur non trouvé", "Mot de passe incorrect").
- Redirection intelligente vers le tableau de bord correspondant au rôle (Admin ou Conducteur).
- Dashboard Admin



The screenshot shows the 'Smart Parking Dashboard' interface. At the top, there's a header bar with the title 'Smart Parking Dashboard' and a 'Déconnexion' button. Below the header, there are three tabs: 'Parkings' (selected), 'Places', and 'Capteurs'. A search bar with placeholder 'Rechercher par nom...' and a 'Rechercher' button are located below the tabs. The main area is a table with columns: ID, Nom, Adresse, Capacité, and Places Libres. The table contains five rows of parking information:

ID	Nom	Adresse	Capacité	Places Libres
1	Parking Central 1 Casabl...	123 Avenue zraktouni, Casablanca	2	1
2	Parking Central 2 Casabl...	234, Casablanca	44	1
9	parking Maarif	32 Avenu X , casablanca	20	0
10	Parking Central Casablanca	123 Avenue Mohamed V, Casablanca	600	1

At the bottom of the dashboard, there are four buttons: 'Ajouter Parking' (blue), 'Modifier' (blue), 'Supprimer' (orange), and 'Actualiser' (blue). A status message 'Connecté en tant que Administrateur : Alami' is displayed at the bottom.

**Description:** Interface dédiée à la supervision complète de l'infrastructure.

**Fonctionnalités:**

**Gestion des Parkings:** Tableau interactif avec options d'ajout, de modification et de suppression.

**Indicateur de Disponibilité :** Colonne "Places Libres" recalculée dynamiquement pour chaque site.

**Suivi des Capteurs :** Liste des capteurs IoT avec leur état réel (Opérationnel/En panne).

✓ Modifier

Smart Parking System - Connexion

## Smart Parking Dashboard

Déconnexion

Parkings Places Capteurs

Rechercher par nom... Rechercher

ID Nom

1	Parking Central 1 Casabla...	123 Avenue zrak...
2	Parking Central 2 Casabl...	234, Casablanca
9	parking Maarif	32 Avenu X , cas...
10	Parking Central Casablanca	123 Avenue Mo...

Modifier un Parking

Modifiez les informations du parking : Parking Central 1 Casablanca

Nom:

Adresse:

Capacité:

Enregistrer Annuler

Ajouter Parking Modifier Supprimer Actualiser

Connecté en tant que Administrateur : Alami

✓ Supprimer

Smart Parking System - Connexion

## Smart Parking Dashboard

Déconnexion

Parkings Places Capteurs

Rechercher par nom... Rechercher

ID	Nom	Places Libres
1	Parking Central 1 Casabl...	1
2	Parking Central 2 Casabl...	2
9	parking Maarif	3
10	Parking Central Casablanca	1

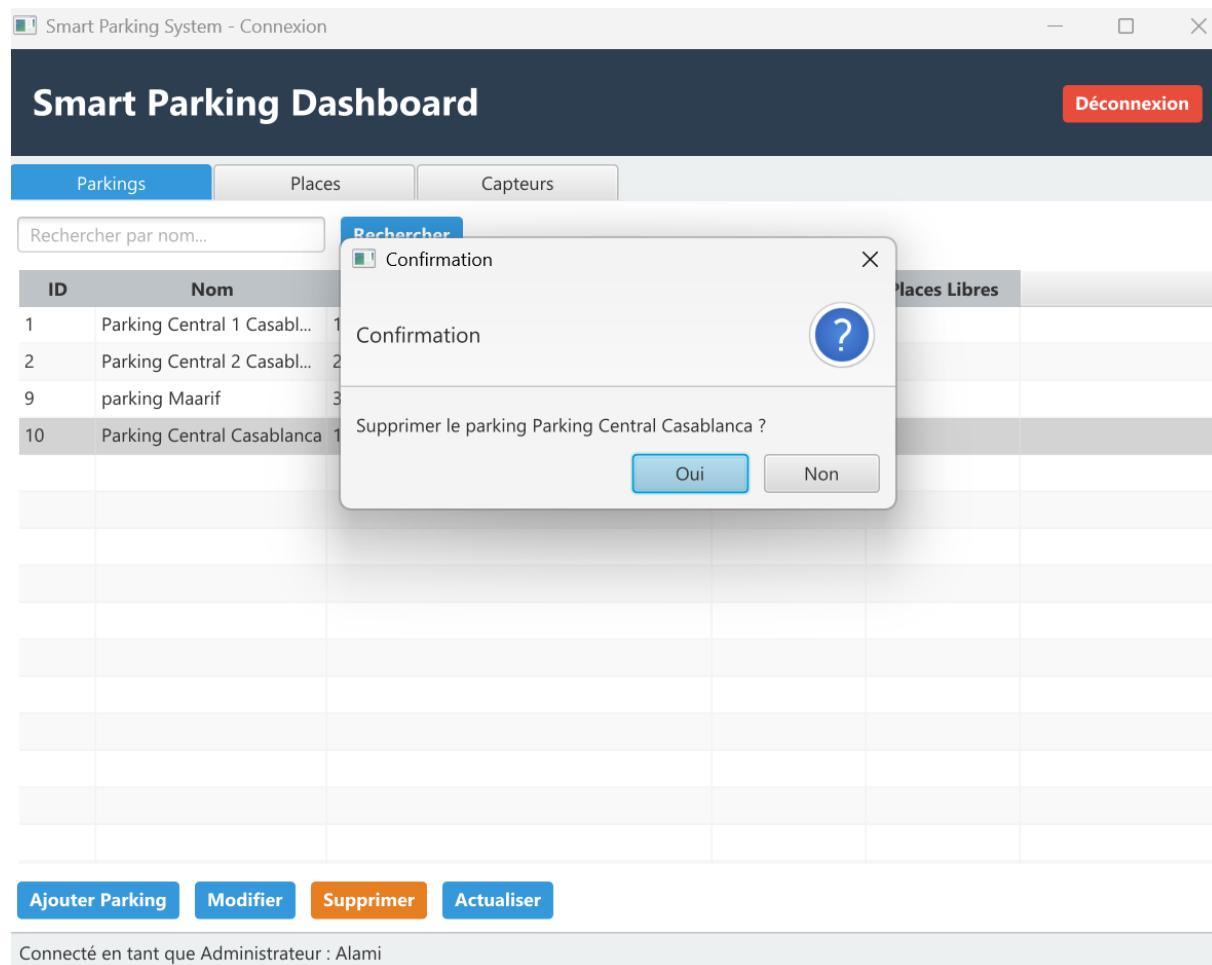
Confirmation

Supprimer le parking Parking Central Casablanca ?

Oui Non

Ajouter Parking Modifier Supprimer Actualiser

Connecté en tant que Administrateur : Alami



✓ Ajouter

Smart Parking System - Connexion

## Smart Parking Dashboard

Déconnexion

Parkings Places Capteurs

Rechercher par nom...

ID	Nom
1	Parking Central 1 Casabl...
2	Parking Central 2 Casabl...
9	parking Maarif
10	Parking Central Casablanca

Ajouter un Parking

Entrez les informations du nouveau parking

Nom:

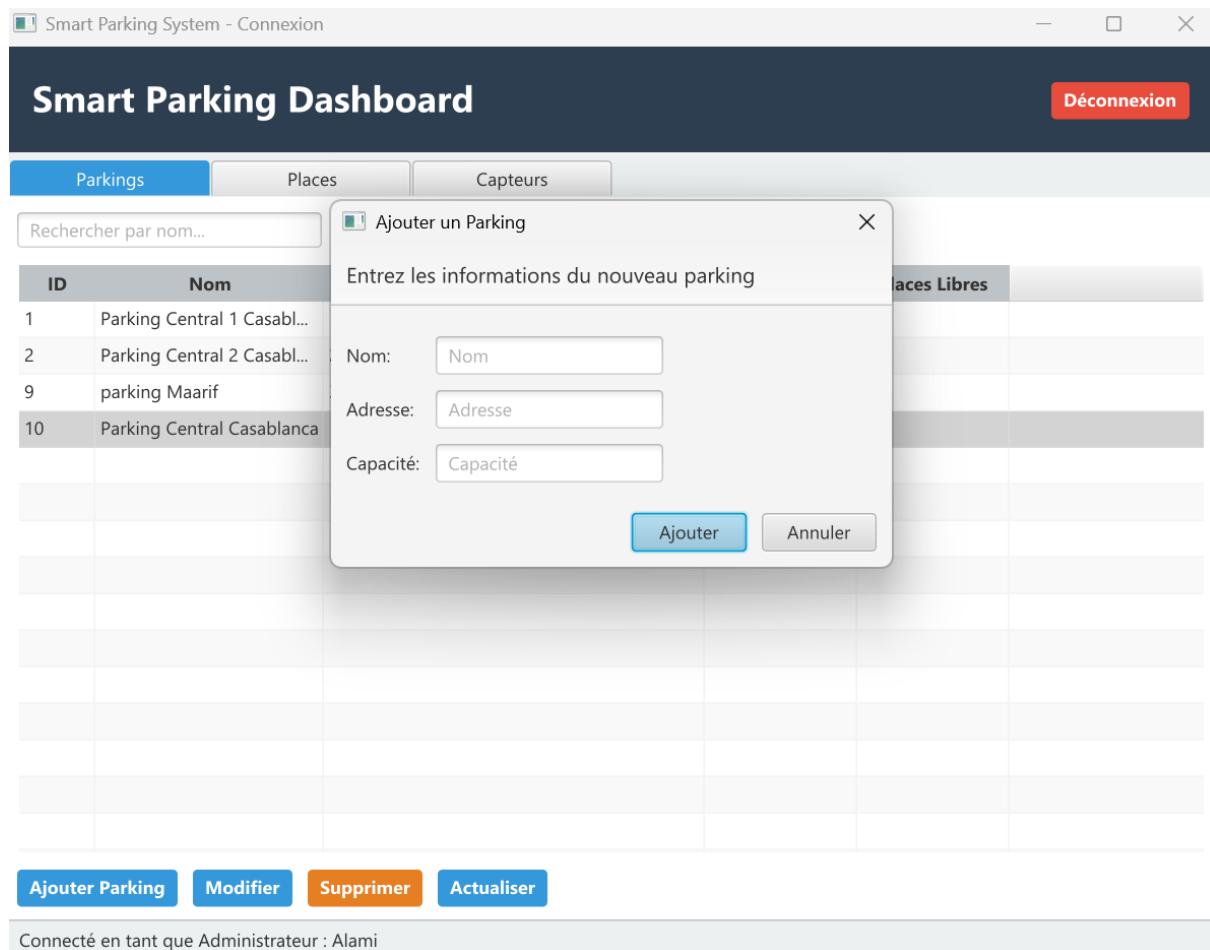
Adresse:

Capacité:

Ajouter Annuler

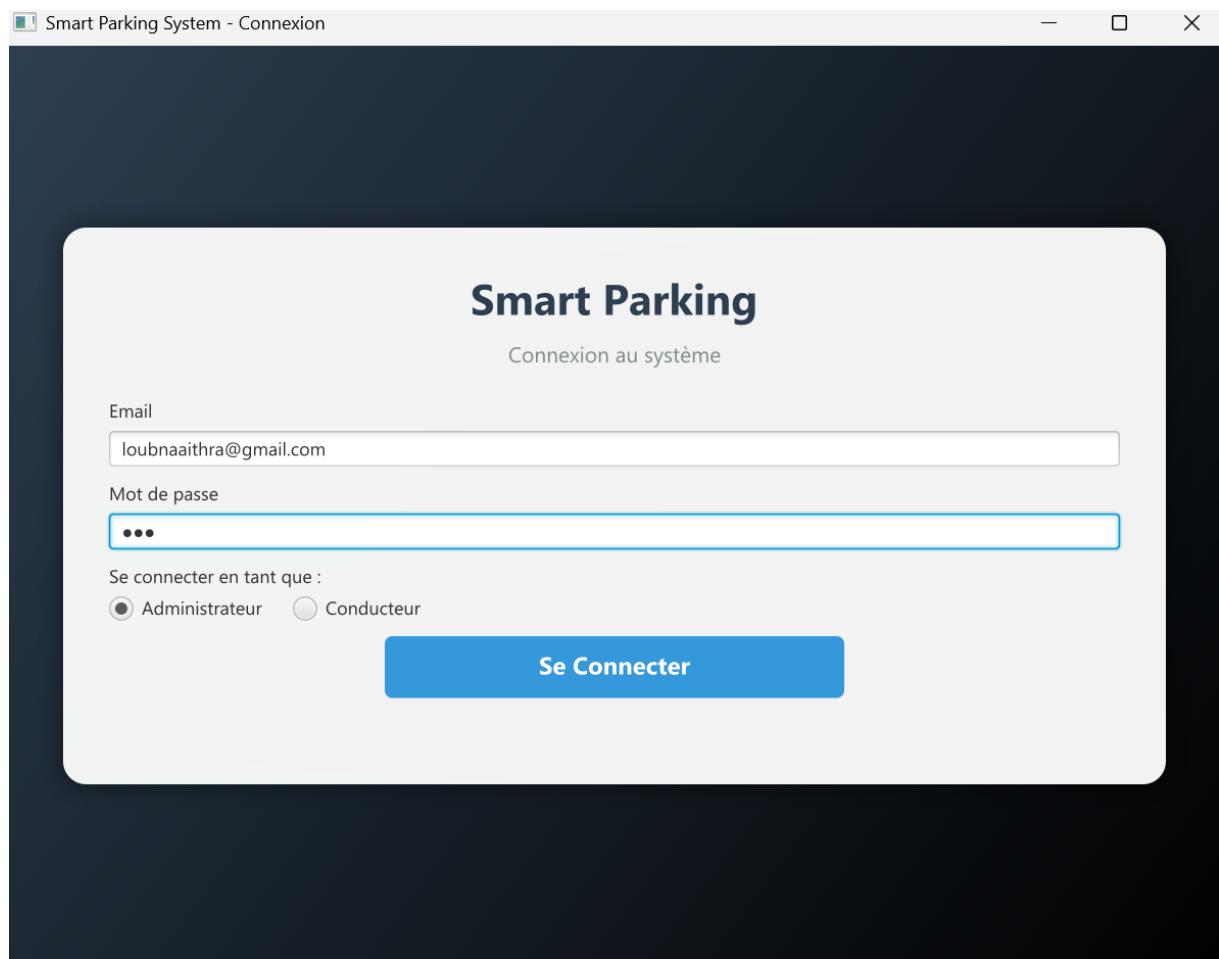
Ajouter Parking Modifier Supprimer Actualiser

Connecté en tant que Administrateur : Alami



✓ Place

- Interface Conducteur



**Description:** Interface fluide centrée sur l'utilisateur final et la mobilité.

**Fonctionnalités :**

**Disponibilité des Places** : Affichage clair avec les statuts "Libre" ou "Occupée".

**Réservation Instantanée** : Système permettant de bloquer une place en un clic (statut Persistant).

**Expérience Personnalisée** : Les places réservées par l'utilisateur actuel s'affichent avec le texte \*\*"Ma Réservation"\*\* en rouge pour une visibilité immédiate.

**Annulation** : Possibilité de libérer la place à tout moment pour la rendre à nouveau disponible.



Smart Parking System - Connexion

## Smart Parking Dashboard

Déconnexion

ID	Numéro	Type	Statut	Parking
1	A-001	STANDARD	Occupée	Parking Central 1 Casabl...
2	A-002	HANDICAP	Libre	Parking Central 1 Casabl...
3	A-001	STANDARD	Occupée	Parking Central 2 Casabl...
4	A-002	HANDICAP	Libre	Parking Central 2 Casabl...
15	A-001	STANDARD	Occupée	Parking Central Casablanca
16	A-002	HANDICAP	Libre	Parking Central Casablanca

Réserver   Annuler   Actualiser

Connecté en tant que Conducteur : loubna

✓ Reservation

Smart Parking System - Connexion

## Smart Parking Dashboard

Déconnexion

Parkings					Places
ID	Numéro	Type	Statut	Parking	
1	A-001	STANDARD	Occupée	Parking Central 1 Casabl...	
2	A-002	HANDICAP	Libre	Parking Central 1 Casabl...	
3	A-001	STANDARD	Occupée	Parking Central 2 Casabl...	
4	A-002	HANDICAP	Libre	Parking Central 2 Casabl...	
15	A-001	STANDARD	Occupée	Parking Central Casablanca	
16	A-002	HANDICAP	<b>Ma Réservation</b>	Parking Central Casablanca	

 Succès

Réservation effectuée

i

Place A-002 réservée avec succès.

OK

**Réserver** **Annuler** **Actualiser**

Capteurs actualisés (1)

## 5.2 Scénarios de Test

**Nominal :** Loubna réserve une place -> Status devient "Ma Réservation".

**Erreur :** Tentative de réservation d'une place déjà occupée -> Alerte bloquante.

**Restauration :** Annulation d'une réservation -> La place redevient "Libre".

## ***6 Conclusion et Perspectives***

- \* **Bilan technique :** Toutes les fonctionnalités Java Avancé (Hibernate, JavaFX, Architecture multi-couches) sont opérationnelles.
- \* **Difficultés :** La synchronisation entre les entités Hibernate et l'affichage JavaFX a nécessité une gestion fine des sessions.
- \* **Perspectives :** Ajout d'un système de paiement par carte (module 'Paiement') et intégration de notifications en temps réel.

## 7 Webographie / Bibliographie

Documentation Hibernate : [hibernate.org](<https://hibernate.org/>)

Tutoriels JavaFX : [openjfx.io](<https://openjfx.io/>)

Cours EMSI Java Avancé.