



**ECOLE MAROCAINE DES
SCIENCES DE L'INGENIEUR**
Membre de
HONORIS UNITED UNIVERSITIES

FILIÈRE : 4ème année IIR
CENTRE : My Youssef
Année universitaire : 2025/2026

Projet intitulé sous :

APPLICATION DE GESTION INTELLIGENTE DU STATIONNEMENT URBAIN

Réalisé par :
Amine Erradha

Encadré par : M.Abderrahim LARHLIMI

Remerciements

Nous tenons à remercier notre encadrant pédagogique **M.Abderrahim LARHLIMI** pour son accompagnement, ainsi que l'administration de l'EMSI pour les moyens mis à disposition ayant contribué à la réussite de ce projet.

Table des matières

Remerciements	2
Introduction Générale	4
1 Contexte du projet	4
2 Problématique	4
3 Objectifs	4
4 <u>Analyse et Conception</u>	5
4.1 Spécification des besoins	5
4.1.1 Besoins fonctionnels	5
4.1.2 Besoins non fonctionnels	5
5 Conception UML	6
5.1 Diagramme de classes	6
6 Environnement Technique	7
6.1 Java (JDK 17) :	7
6.2 Maven :	8
7 Architecture logicielle	9
7.1 Les objets metiers	9
8 DAO :	10
9 Les interface	11
9.1 login et authentification	11
10 <u>Interface admin pour ajouter et modifier le parking</u>	12
11 <u>interface user pour reservation de parking</u>	12
Conclusion	13

Introduction Générale

Ce projet s'inscrit dans le cadre du module **Java Avancé**. Il vise à concevoir et développer une application de gestion intelligente du stationnement urbain permettant d'optimiser la réservation des places de parking.

1 Contexte du projet

La croissance urbaine entraîne une augmentation du nombre de véhicules, rendant le stationnement difficile dans les zones urbaines.

2 Problématique

L'absence de solutions numériques efficaces entraîne une perte de temps, une congestion du trafic et une mauvaise gestion des espaces de stationnement.

3 Objectifs

Les objectifs principaux du projet sont :

- Permettre la réservation de places de parking
- Gérer la disponibilité en temps réel
- Offrir une interface simple et intuitive

4 Analyse et Conception

4.1 Spécification des besoins

4.1.1 Besoins fonctionnels

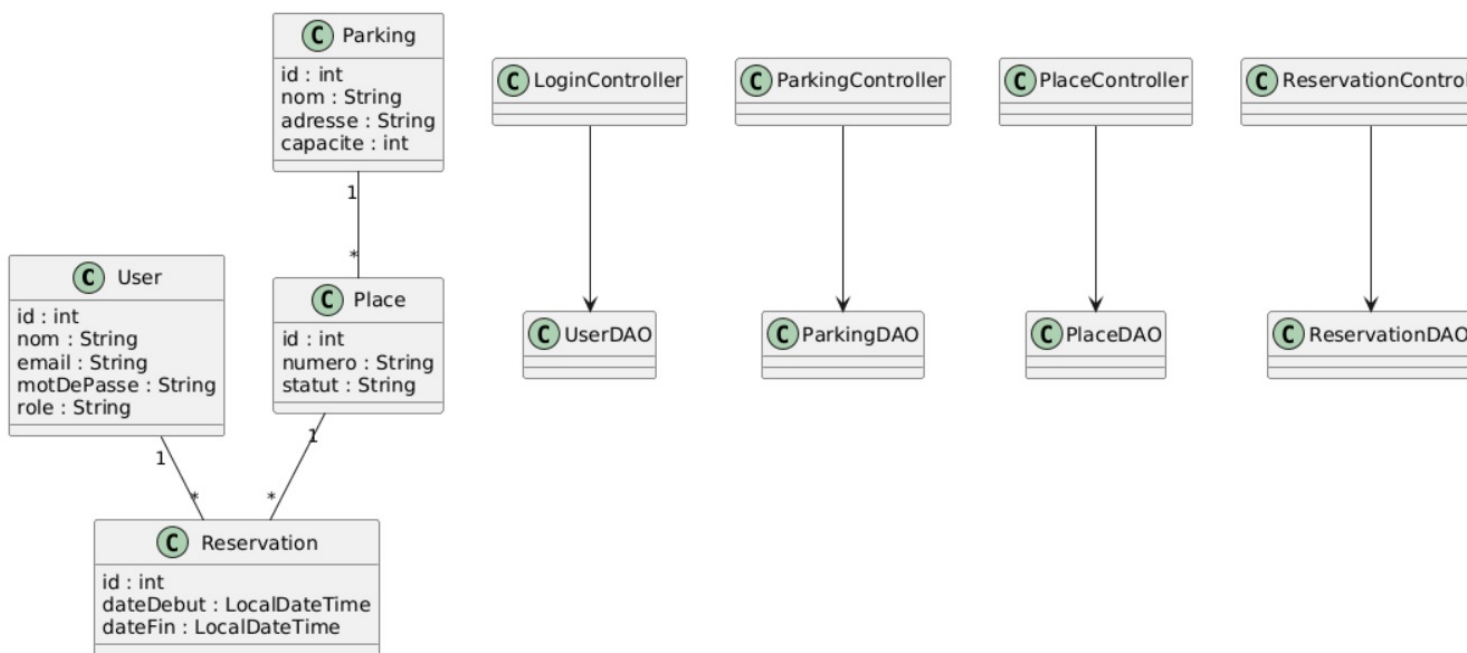
- L'utilisateur peut rechercher un parking par prix
- L'utilisateur peut consulter les détails d'un parking
- L'utilisateur peut réserver une place

4.1.2 Besoins non fonctionnels

- Sécurité des données
- Temps de réponse rapide
- Ergonomie de l'interface

5 Conception UML

5.1 Diagramme de classes



6 Environnement Technique

6.1 Java (JDK 17) :



est une version majeure avec support à long terme (LTS) du langage et de la plateforme Java, sortie en septembre 2021. Reconnue pour ses améliorations de performance, de stabilité et de sécurité, elle est recommandée pour les environnements de production modernes.

image.png

6.2 Maven :



Maven est un outil open source de gestion et d'automatisation de projet pour Java, qui simplifie des tâches comme la construction (build), la gestion des dépendances, l'exécution des tests, le packaging et le déploiement. Il utilise le fichier pom.xml pour configurer le projet et suit une approche de "convention plutôt que configuration" (convention over configuration), ce qui standardise l'arborescence des projets et facilite le travail collaboratif.

7 Architecture logicielle

7.1 Les objets metiers

```
public class Parking implements Serializable {

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Integer id;

    @Column(name = "nom", nullable = false)
    private String nom;

    @Column(name = "adresse", nullable = false)
    private String adresse;

    @Column(name = "capacite", nullable = false)
    private Integer capacite;

    @Column(name = "latitude")
    private Double latitude;

    @Column(name = "longitude")
    private Double longitude;

    @Column(name = "tarif_horaire")
    private Double tarifHoraire;

    @Column(name = "description")
    private String description;

    @OneToMany(mappedBy = "parking", cascade = CascadeType.ALL, fetch =
FetchType.LAZY)
    private Set<Place> places = new HashSet<>();
```

8 DAO :

```
public class ParkingDAO {

    public void create(Parking parking) throws SQLException {
        String sql = "INSERT INTO parkings (nom, adresse, capacite, latitude, longitude) VALUES (" + parking.getNom() + ", " + parking.getAdresse() + ", " + parking.getCapacite() + ", " + parking.getLatitude() + ", " + parking.getLongitude() + ")";

        try (Connection conn = DatabaseConnection.getConnection();
            PreparedStatement stmt = conn.prepareStatement(sql)) {

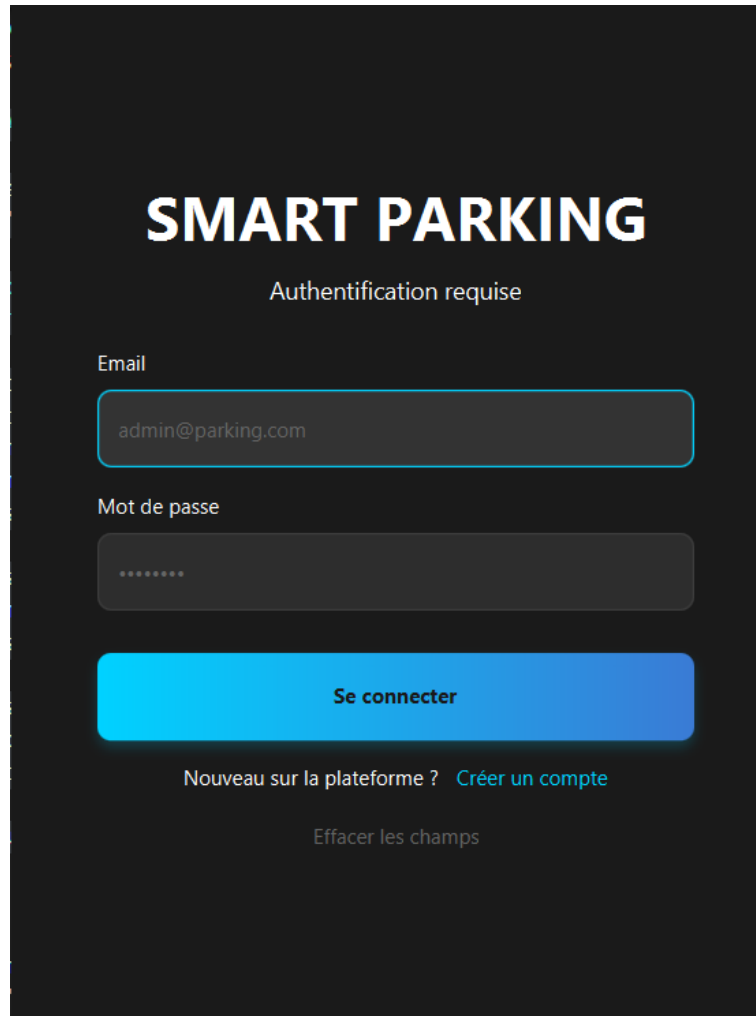
            stmt.setString(1, parking.getNom());
            stmt.setString(2, parking.getAdresse());
            stmt.setInt(3, parking.getCapacite());
            if (parking.getLatitude() != null)
                stmt.setDouble(4, parking.getLatitude());
            else
                stmt.setNull(4, java.sql.Types.DOUBLE);
            if (parking.getLongitude() != null)
                stmt.setDouble(5, parking.getLongitude());
            else
                stmt.setNull(5, java.sql.Types.DOUBLE);
            stmt.setDouble(6, parking.getTarifHoraire());
            stmt.setString(7, parking.getDescription());

            stmt.executeUpdate();

        }
    }
}
```

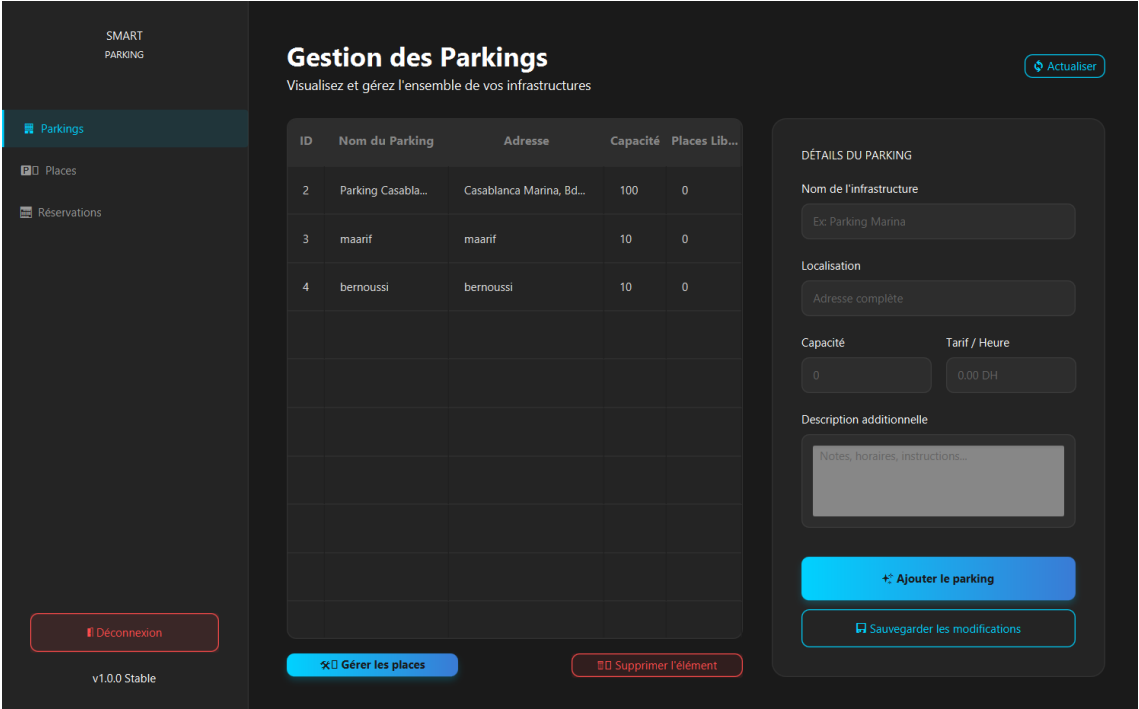
9 Les interface

9.1 login et authentication

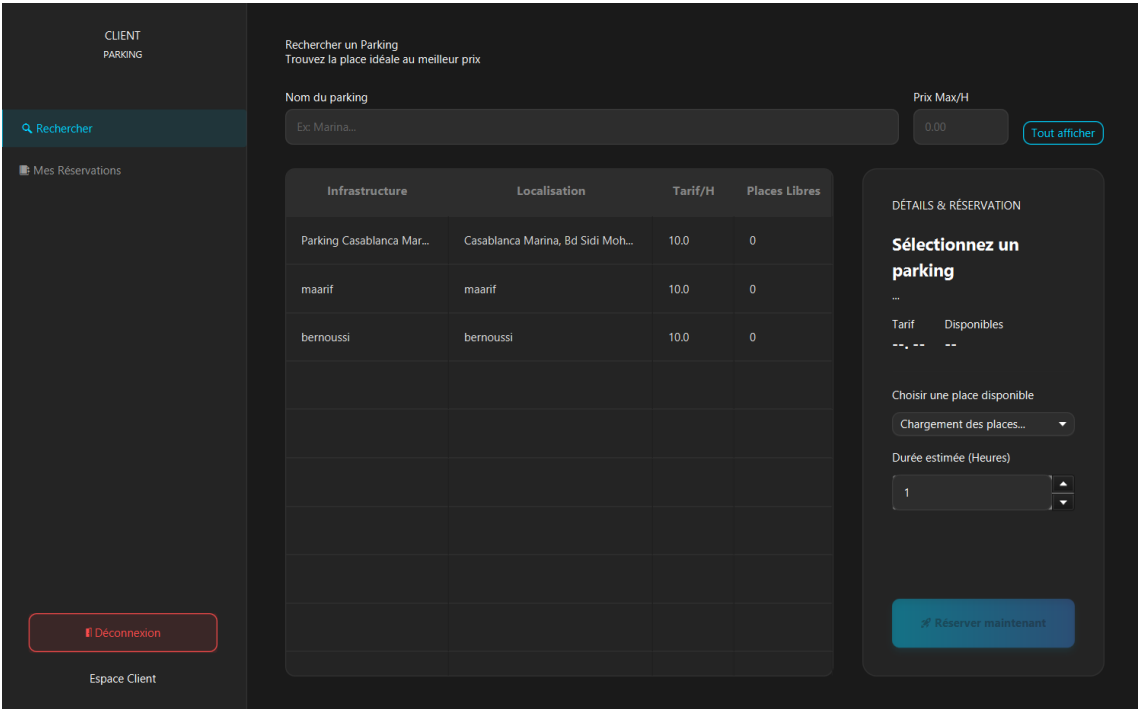
The image shows a dark-themed login interface for 'SMART PARKING'. At the top, the title 'SMART PARKING' is in large, bold, white capital letters. Below it, the text 'Authentication requise' is in a smaller, lighter font. There are two input fields: 'Email' with the placeholder 'admin@parking.com' and 'Mot de passe' with a masked password '*****'. A prominent blue button with the text 'Se connecter' is centered below the fields. At the bottom, there is a link 'Nouveau sur la plateforme ? Créer un compte' and a text link 'Effacer les champs'.

L'écran d'authentification constitue le point d'entrée principal de l'application Smart Parking. Il permet aux utilisateurs d'accéder à la plateforme de manière sécurisée en saisissant leur adresse e-mail et leur mot de passe. Une fois les informations renseignées, le bouton « Se connecter » permet de valider l'accès au système. L'interface offre également une option « Créer un compte » destinée aux nouveaux utilisateurs souhaitant s'inscrire sur la plateforme, ainsi qu'une fonctionnalité « Effacer les champs » pour réinitialiser les données saisies. Cet écran garantit la sécurité des accès et assure que seules les personnes autorisées peuvent utiliser les fonctionnalités de réservation et de gestion des parkings.

10 Interface admin pour ajouter et modifier le parking



11 interface user pour reservation de parking



Conclusion

Ce projet de Smart Parking a permis de développer une application fonctionnelle répondant aux exigences du cahier des charges, notamment l'authentification des utilisateurs, la réservation de places de parking et la gestion des réservations. L'architecture adoptée et les technologies utilisées ont assuré une application structurée et cohérente.

Sur le plan personnel, ce projet a contribué au renforcement des compétences en Java, JDBC, Maven et en programmation orientée objet, tout en développant le travail en équipe. Certaines difficultés techniques, principalement liées à la gestion de la base de données et des exceptions, ont été surmontées grâce aux tests et à la recherche.

Enfin, des améliorations futures sont envisageables, telles qu'une version web ou mobile, un renforcement de la sécurité et l'intégration de nouvelles fonctionnalités pour rendre l'application plus complète et plus performante.