

مكتب التكوين المهني وإنعاش الشغل

L'Office de la Formation Professionnelle et de la
Promotion du travail



مكتب التكوين المهني
 وإنعاش الشغل
الطريق الأمثل للمستقبل

Rapport de projet de fin d'étude

Développement d'une Application web
pour la gestion de stationnement

**Spécialité : Développement Digital
Option Web Full Stack**

Année professionnelle : 2024-2025

Réalisé par :

Amine EL Bekaroui

Encadré par:

Mr. Ismail Barhdadi

Dédicace

Nous nous trouvons incapables de trouver les mots pour exprimer notre gratitude et nos remerciements à tous ceux qui nous ont aidés de près ou de loin pour faire réussir notre travail, et nous dédions ce rapport :

A tous les membres de nos familles pour leurs sacrifices, leurs encouragements, et pour leur soutien matériel et morale tout au long de notre formation.

Au directeur et directrice de notre institut (I.S.T.A) pour leurs efforts qui nous a aidé dans notre formation.

A nos amis qui n'ont cessé de nous aider et de nous conseiller pour accomplir notre rapport.

A nos formateurs et formatrices qui ont fait leurs efforts pour nous aider à travailler dans les bonnes conditions et pour l'aide qu'ils ont toujours porté aux stagiaires.



Remerciement

Ce travail n'aurait jamais pu se concrétiser sans l'aide et le soutien de plusieurs personnes que nous souhaitons vivement remercier et à qui nous dédions ce travail.

Monsieur **Barhdadi Ismail** notre encadrant, qui n'a pas cessé de nous prodiguer ses conseils et qui n'a épargné aucun effort pour contribuer à la réussite de notre travail.

Nos formateurs et formatrices à l'institut pour leurs Responsabilités et leurs conseils continus grâce auxquels nous avons pu arriver à ce travail
Notre Institut qui nous a donné l'occasion d'acquérir une formation professionnelle.



Sommaire

Dédicace	1
Remerciement.....	2
Sommaire.....	3
Liste des figures.....	5
Liste des tableaux	6
Chapitre 1 : Cadre générale du projet.....	8
I. Cadre générale du projet :	8
II. Présentation de sujet :	8
1. Présentation de projet :.....	8
2. Le but à atteindre :	8
3. Travail demandé :.....	9
III. Planification du projet :.....	9
Chapitre 2 : Analyse et spécification des besoins.....	12
I. Etude préalable :	12
1. Etude de l'existant :.....	12
2. Solutions envisagées :	12
II. Spécifications des besoins :	13
1. Les besoins fonctionnels :.....	13
Chapitre 3 : Etude conceptuelle	16
Introduction	16
I. Cycle de vie de développement de projet :	16
II. Langage UML :.....	17
a. Présentation de langage UML :	17
b. Intérêt de la modélisation :	17
III. Conception avec UML :	19
a. Outil de modélisation :.....	19
b. Modélisation avec les diagrammes cas d'utilisation :	20
Chapitre 3 : Réalisation.....	32
Introduction	32
I. Environnement de travail :.....	32

a.	Environnement de développement intégré :	32
II.	Choix de développement :	35
1.	Le modèle :	36
2.	La vue :	36
3.	Le contrôleur :	37
III.	Les technologies utilisées :	37
IV.	Les interfaces graphiques :	39
?	Page d'authentification :	39
?	Interface de profil d'utilisateur :	40
?	Partie de gérant :	41
a.	Page de consulter factures :	41
b.	Interface d'affichage des factures payée et impayée d'un adhérent :	42
c.	Interface de rapport de consommation d'un adhérent :	43
d.	Interface de liste des utilisateurs :	44
e.	Interface de voir des informations d'un utilisateur :	45
f.	Interface de modifier des informations d'un utilisateur :	45
g.	Interface confirmation de suppression d'un utilisateur :	48
h.	Interface d'ajout des utilisateurs :	49
i.	Interface de gérer les tranches :	50
?	Partie de responsable de paiement :	Error! Bookmark not defined.
a.	Interface de liste des adhérents :	Error! Bookmark not defined.
b.	Interface de modifier un adhérent :	Error! Bookmark not defined.
c.	Interface Ajouter adhérent :	Error! Bookmark not defined.
d.	Interface de supprimer un adhérent :	Error! Bookmark not defined.
e.	Interface de gestion des village de commune :	Error! Bookmark not defined.
f.	Interface de payer de facture :	Error! Bookmark not defined.
?	Partie de technicien :	Error! Bookmark not defined.
a.	Interface du déposer un consommation :	Error! Bookmark not defined.
b.	Interface de modification de consommation	Error! Bookmark not defined.
Glossaire		52
Webographie		53
Conclusion		54

Liste des figures

Figure 1 Méthode agile	17
Figure 2Diagram de cas d'utilisation	21
Figure 3 Diagram de cas d'utilisation de parametrage des tranches	Error! Bookmark not defined.
Figure 8 diagramme de classe	24
Figure 9 Diagramme de sequence d'authentification	27
Figure 10 Diagramme de séquence <>paramétrage des tranches>.....	28
Figure 11 Diagramme de classe de gestion des utilisateurs	Error! Bookmark not defined.
Figure 12 Diagramme de séquence de <>Consulter les statistiques>.....	29
Figure 17 Logo de l'éditeur Visual studio code	32
Figure 18 Logo de serveur XAMPP	33
Figure 19 Schéma du Modèle MVC	36
Figure 20 Logo de framework Laravel.....	37
Figure 21 Logo de framework Bootstrap.....	38
Figure 22 Page d'authentification	39
Figure 23 Interface nouveau voiture	40
Figure 24interface dashboard.....	41
Figure 25 Interface Admin create.....	42
Figure 26 Interface de Admin Liste.....	43
Figure 27 Interface de create un utlisateur	44
Figure 28 Interface de voir les listes d'un utilisateur	45
Figure 29 Interface de create categorie de voiture.....	46
Figure 30 liste des category.....	47
Figure 31 interface de create vericle	48
Figure 32 Interface de Liste Vehicule.....	49
Figure 33 Interface de Vehicule in liste et Ajouter new Vehicules.....	50
Figure 34 Interface de vehicule out list.....	51

Liste des tableaux

Tableau 1 planification temporelle	9
Tableau 2 Cas d'utilisation de paramétrage des tranches	22

Cadre du projet

Chapitre 1 : Cadre générale du projet

Dans ce chapitre nous mettons notre travail dans son contexte général nous présentons le cadre général de notre projet.

I. Cadre générale du projet :

Notre projet consiste en le développement d'une application de gestion de stationnement destinée aux communes urbaines et rurales. Cette application a pour objectif de moderniser et d'optimiser le processus de gestion du stationnement dans ces communes, en remplaçant les méthodes manuelles et archaïques utilisées jusqu'à présent. En effet, les communes rurales rencontrent souvent des difficultés dans la gestion des places de stationnement, la facturation et le contrôle des véhicules stationnés, ce qui entraîne des erreurs et une inefficacité globale..

II. Présentation de sujet :

1. Présentation de projet :

Le projet vise à développer une application de gestion de stationnement spécifiquement conçue pour répondre aux besoins des communes urbaines et rurales. Cette application permettra de moderniser les processus de gestion du stationnement, en remplaçant les méthodes manuelles et obsolètes utilisées jusqu'à présent. Son objectif est d'optimiser la gestion des places de stationnement, la facturation et le contrôle des véhicules stationnés, afin d'améliorer l'efficacité et de réduire les erreurs dans ces tâches cruciales. En fournissant une solution technologique adaptée aux particularités des communes, cette application contribuera à améliorer la gestion globale du stationnement et à assurer une utilisation plus efficace et durable des espaces disponibles pour les habitants de ces communautés.

2. Le but à atteindre :

Le but ultime de notre projet est de fournir une application de gestion de stationnement fonctionnelle et conviviale pour les communes urbaines et rurales. Cette solution simplifiera les processus de gestion et optimisera l'utilisation des espaces de stationnement. Elle sera efficace, sécurisée et adaptée aux besoins spécifiques de chaque commune, tout en étant facile à utiliser pour les utilisateurs.

3. Travail demandé :

Pour atteindre notre objectif, nous avons élaboré un plan de travail détaillé.

Ce plan comprend différentes phases, telles que :

- l'analyse des besoins des communes rurales en matière de gestion de l'eau.
- la conception de l'application en utilisant des outils tels que UML.
- le développement du backend en utilisant le langage de programmation PHP et le Framework Laravel.
- la création de l'interface utilisateur en utilisant des langages et des outils tels que HTML, CSS, Bootstrap et JavaScript.

Nous prévoyons également :

- des phases de test, de débogage et de déploiement pour nous assurer que l'application fonctionne correctement et répond aux attentes des utilisateurs.

III. Planification du projet :

La planification du projet a joué un rôle clé dans sa réussite. Nous avons établi un plan détaillé, en identifiant les différentes étapes du développement de l'application, depuis l'analyse des besoins jusqu'à la phase de déploiement et de test. Nous avons également défini des objectifs intermédiaires et des délais pour chaque étape, ce qui nous a permis de suivre notre progression et de garantir la livraison du projet dans les délais impartis.

Mois	Février 2024				Mars 2024				Avril 2024				Mai 2024			
Semaine	Semaine 1	Semaine 2	Semaine 3	Semaine 4	Semaine 1	Semaine 2	Semaine 3	Semaine 4	Semaine 1	Semaine 2	Semaine 3	Semaine 4	Semaine 1	Semaine 2	Semaine 3	Semaine 4
Recherche et documentation	■	■	■	■	■	■	■	■	■	■	■	■				
Analyse des besoins et Spécification				■	■	■	■	■								
Conception																
Développement																
Tests et validation													■	■	■	■
Élaboration du rapport									■	■	■	■	■	■	■	■

Tableau 1 planification temporelle

Résumé :

Dans ce chapitre introductif, nous avons pris le temps de situer notre projet dans son contexte général, offrant ainsi une vue d'ensemble claire de notre démarche. Nous avons commencé par présenter le cadre général du projet, mettant en lumière les enjeux et les défis auxquels nous sommes confrontés dans notre domaine d'intervention.

Ensuite, nous avons offert une brève présentation de notre projet, exposant ses objectifs principaux et les solutions que nous envisageons d'apporter pour répondre aux besoins identifiés. Cette présentation nous a permis de définir clairement le but à atteindre, servant de guide tout au long de notre démarche.

Nous avons également décrit le travail demandé dans le cadre de ce projet, mettant en évidence les différentes tâches et activités que nous devrons entreprendre pour atteindre nos objectifs. Cette analyse nous a permis de mieux comprendre les exigences du projet et d'organiser notre travail de manière efficace.

Analyse et spécifications des besoins

Chapitre 2 : Analyse et spécification des besoins

Introduction :

Ce chapitre est dédié à l'analyse et aux spécifications des besoins pour notre application de gestion de stationnement destinée aux communes urbaines et rurales. L'objectif principal de cette étape est de comprendre en détail les exigences et les attentes des utilisateurs afin de concevoir une solution adaptée et répondant à leurs besoins spécifiques.

I. Etude préalable :

Dans notre étude préalable, nous avons identifié une solution envisagée pour répondre aux besoins de gestion du stationnement dans les communes urbaines et rurales. Cette solution consiste en le développement d'une application web conviviale, spécifiquement conçue pour la gestion efficace du stationnement.

1. Etude de l'existant :

a. Description de l'existant :

Nous décrivons en détail les procédures et les outils utilisés dans la gestion du stationnement des communes urbaines et rurales. Cela comprend la manière dont les places de stationnement sont gérées, la facturation manuelle, le contrôle manuel des véhicules stationnés, ainsi que les registres et les documents utilisés.

b. Critique de l'existant :

Nous évaluons les limitations et les problèmes rencontrés dans l'existant, tels que les erreurs humaines, l'inefficacité des processus, la difficulté de suivi et de contrôle, ainsi que les risques de perte de données.

2. Solutions envisagées :

L'application web offre plusieurs avantages par rapport aux méthodes de gestion manuelle traditionnelles. Voici quelques-unes des fonctionnalités clés que notre application web propose :

- **Gestion des adhérents :** L'application permet aux utilisateurs de gérer facilement les informations des conducteurs. Ils peuvent enregistrer de nouveaux conducteurs, mettre à jour leurs informations personnelles, et supprimer les conducteurs si nécessaire. Cela élimine la nécessité de maintenir des registres physiques et facilite la recherche et la gestion des conducteurs.
- **Suivi des véhicules stationnés :** L'application permet de suivre et d'analyser les véhicules stationnés. Les utilisateurs peuvent visualiser les tendances de stationnement, détecter les places occupées de manière prolongée, et prendre des mesures pour optimiser l'utilisation des espaces. Cela facilite la gestion des places de stationnement disponibles et aide à identifier les problèmes potentiels tels que les véhicules abandonnés.
- **Sécurité des données :** Sécurité des données : L'application web garantit la sécurité des données des conducteurs. Les informations sensibles sont stockées de manière sécurisée et l'accès aux données est contrôlé par des mécanismes d'authentification et d'autorisation. Cela assure la confidentialité et l'intégrité des informations des conducteurs

II. Spécifications des besoins :

1. Les besoins fonctionnels :

Nous identifions les fonctionnalités essentielles que notre application doit offrir. Cela peut inclure la gestion des utilisateurs, la création de factures automatisées, le suivi des véhicules stationnés, la génération de rapports, etc. Nous détaillons chaque fonctionnalité et ses interactions avec les utilisateurs.

2. Les besoins non fonctionnels :

Nous définissons les critères de qualité et les exigences non fonctionnelles de notre application. Cela peut inclure la sécurité des données, la convivialité de l'interface utilisateur, la performance du système, la compatibilité avec différents appareils, etc.

Résumé :

Ce chapitre d'analyse et de spécifications des besoins nous permet de comprendre en profondeur les enjeux et les attentes des communes urbaines et rurales en matière de gestion du stationnement. En identifiant les problèmes existants et en spécifiant les fonctionnalités nécessaires, nous sommes en mesure de concevoir une application qui répondra efficacement aux besoins des utilisateurs. Dans le chapitre suivant, nous aborderons la conception de l'application en utilisant les informations recueillies lors de cette analyse approfondie.

Etude conceptuelle

Chapitre 3 : Etude conceptuelle

Introduction

Penser avant d'agir, faire des plans avant de construire, concevoir d'abord, développer ensuite c'est la démarche qui doit être suivre lors du développement d'une application et pour réussir n'importe quel projet.

En effet, La conception d'un système informatique est une étape très importante qui va influencer la qualité et la fiabilité de toute application.

D'abord, nous allons commencer ce chapitre par l'explication du modèle de cycle de vie de projet qu'on a choisi. Ensuite nous allons passer à la partie de conception détaillée où nous présentons l'architecture globale de l'application. Enfin nous citons les différents diagrammes de cas d'utilisation, de séquences et le diagramme de classes.

I. Cycle de vie de développement de projet :

Le cycle de vie d'un projet est une séquence d'étapes distinctes qui permet de gérer et d'organiser le développement ou la mise en œuvre d'un projet, depuis sa conception initiale jusqu'à sa clôture finale. Chaque étape du cycle de vie est caractérisée par des activités spécifiques et des livrables associés. Le cycle de vie du projet peut varier en fonction de la méthodologie ou de l'approche de gestion de projet utilisée.

Pour fournir une meilleure réalisation, nous avons présenté la méthode agile est une approche itérative et collaborative du développement de logiciels qui se concentre sur la flexibilité, l'adaptabilité et la livraison continue de fonctionnalités utilisables.

Voir la figure 1

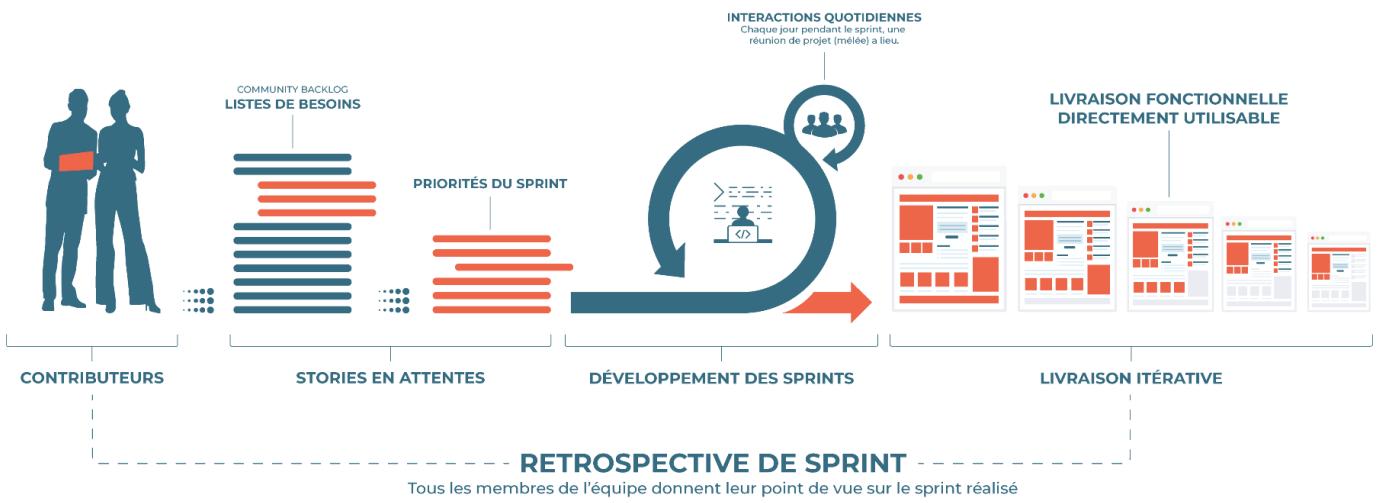


Figure 1 Méthode agile

II. Langage UML :

a. Présentation de langage UML :

UML (en anglais Unified Modeling Language, « langage de modélisation unifié ») est un langage graphique de modélisation des données et des traitements. C'est une formalisation non-propriétaire de la modélisation objet utilisée en génie logiciel. UML spécifie plusieurs objectifs qui font un outil exact de communication :

- Comprendre et décrire les besoins.
- Spécifier un système.
- Établir l'architecture logicielle.

b. Intérêt de la modélisation :

L'utilisation de la modélisation conceptuelle dans le développement des systèmes d'information permet de prendre en compte les besoins des applications d'une façon plus adéquats et de présenter d'une manière abstraite certains aspects des systèmes physiques et humains.

c. Les avantages d'UML :

- UML est un langage formel et standardisé.
 - Gain de précision.
 - Motivation à l'utilisation d'outils.
 - Gagne de stabilité et de fixité.
- UML est un support de communication adéquat et compétitif
 - Il éclaire et facilite la compréhension de représentation abstraite complexe.
 - Son caractère plurivalent et sa souplesse en font un langage universel.
 - UML a pour objectif de spécifier, édifier et documenter les systèmes à base de logiciel.
 - UML n'est pas une méthode mais une notation qui laisse la liberté de la conception.
 - UML est un langage qui permet de modéliser tous les types de systèmes informatiques mais, qui nécessite toutefois une méthodologie de conception.

UML normalise les concepts objet, sa notion graphique permet d'exprimer une solution objet, ce qui simplifie la comparaison et l'appréciation des solutions. UML cadre l'analyse objet, il permet non seulement de représenter les concepts objets, mais il sous-entend une démarche d'analyse qui permet de reproduire une solution objet de manière itérative, grâce aux diagrammes, qui supportent l'abstraction. Un diagramme UML est une représentation graphique, et à chaque vue correspondent des diagrammes qui sont répartis selon leurs aspects statiques ou dynamiques :

➤ **Statique :**

- Cas d'utilisation
- Classes
- Composants
- Objets
- Déploiement

➤ **Dynamique (comportementaux) :**

- Séquences
- Activité
- État-transition
- Collaboration

➤ **Fonctionnel :**

- Cas d'utilisation
- Collaboration



Ces diagrammes ne sont pas nécessairement tous produits à l'occasion d'une modélisation. Les plus utilisables sont les diagrammes d'activités, de classes, de cas d'utilisation, d'objets, d'états transitions et de séquence. Les diagrammes de composants, de déploiement et de communication sont surtout utiles pour la maîtrise d'étude à qu'ils permettent de formaliser les contraintes de la réalisation et les solutions.

III. Conception avec UML :

a. Outil de modélisation :



option pour une familiarisation à la modélisation. Cependant, seule une version Windows est disponible.

StarUML est un logiciel de modélisation UML (Unified Modeling Language) open source qui peut remplacer dans bien des situations des logiciels commerciaux et coûteux comme Rational Rose¹ ou Together². Étant simple d'utilisation, nécessitant peu de ressources système, supportant UML 2, ce logiciel constitue une excellente option pour une familiarisation à la modélisation.

b. Modélisation avec les diagrammes cas d'utilisation :

Le diagramme de cas d'utilisation permet de déterminer les possibilités d'interférence entre le système et les acteurs, c'est-à-dire déterminer toutes les fonctionnalités que doit fournir le système. Il permet aussi de délimiter ce dernier.

- Chaque usage effectué par les acteurs est représenté par un cas d'utilisation.
- Chaque cas d'utilisation symbolise une fonctionnalité qui leur est offerte afin d'engendrer le résultat attendu.
- Le diagramme de cas d'utilisation décrit l'interaction entre le système et l'acteur en déterminant les besoins de l'utilisateur et tout ce que doit faire le système pour l'acteur.

a. Diagramme de cas d'utilisation général :

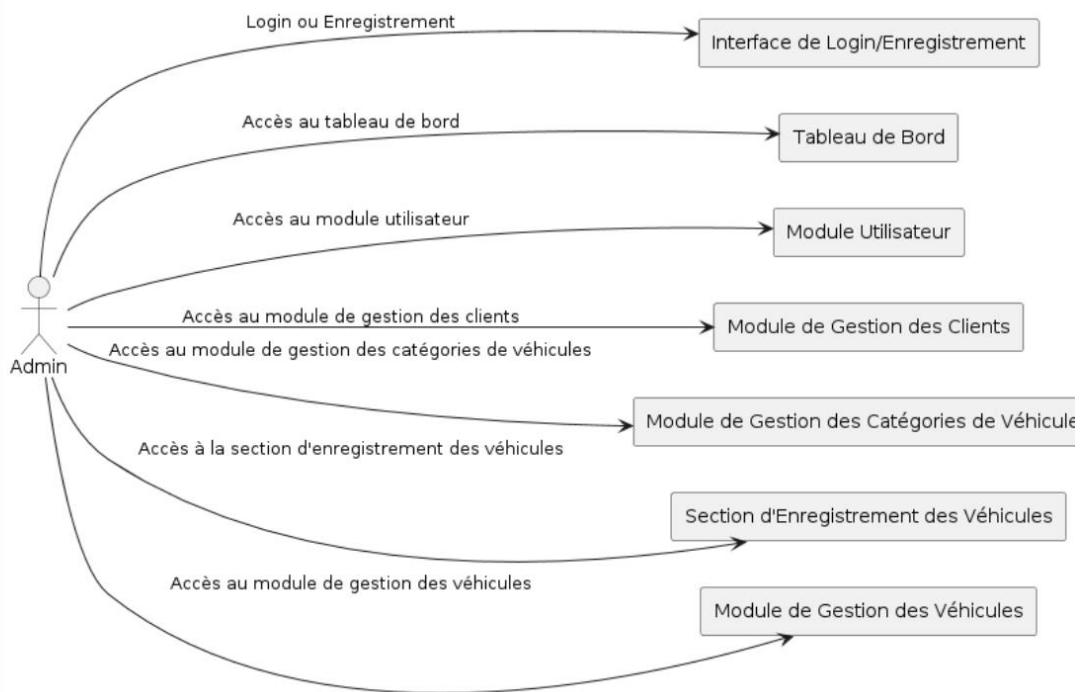


Figure 2 Diagram de cas d'utilisation

La figure 2 représente le diagramme de cas d'utilisation global de notre projet, où les acteurs sont : Administrateur, Utilisateur.

- **Administrateur** : permet de gérer les utilisateurs et leurs rôles, d'accéder au tableau de bord, au module de gestion des clients, au module de gestion des catégories de véhicules, à la section d'enregistrement des véhicules et au module de gestion des véhicules.
- **Utilisateur** : permet d'accéder à l'interface de login/d'enregistrement, au tableau de bord, au module utilisateur et au module de gestion des véhicules.

Cas d'utilisation	Paramétrage des tranches
Acteur	<ul style="list-style-type: none">• Admin
Pré condition	<ul style="list-style-type: none">• Authentification
Description	<ul style="list-style-type: none">• L'Admin modifier et ajouter de chaque tranche.• Consulter le diagramme graphique des tranches.

Tableau 2 Cas d'utilisation de paramétrage des tranches

a. Diagram de classe :

Le diagramme de classe est un diagramme UML qui contient des classes, des interfaces, des packages et leurs relations, et qui fournit une vue logique de tout ou partie d'un système informatique.

On construit un diagramme de classes pour simplifier l'interaction des objets d'un système qu'on est en train de modéliser. Ces diagrammes expriment la structure statique d'un système en termes de classes et de relations entre eux. Une classe décrit un ensemble d'objets et une association décrit un

ensemble de liens. Un diagramme de classe n'exprime rien de spécifique concernant les liens d'un objet particulier, mais il décrit, le lien potentiel entre un objet et d'autres objets.

b. Présentation des classes :

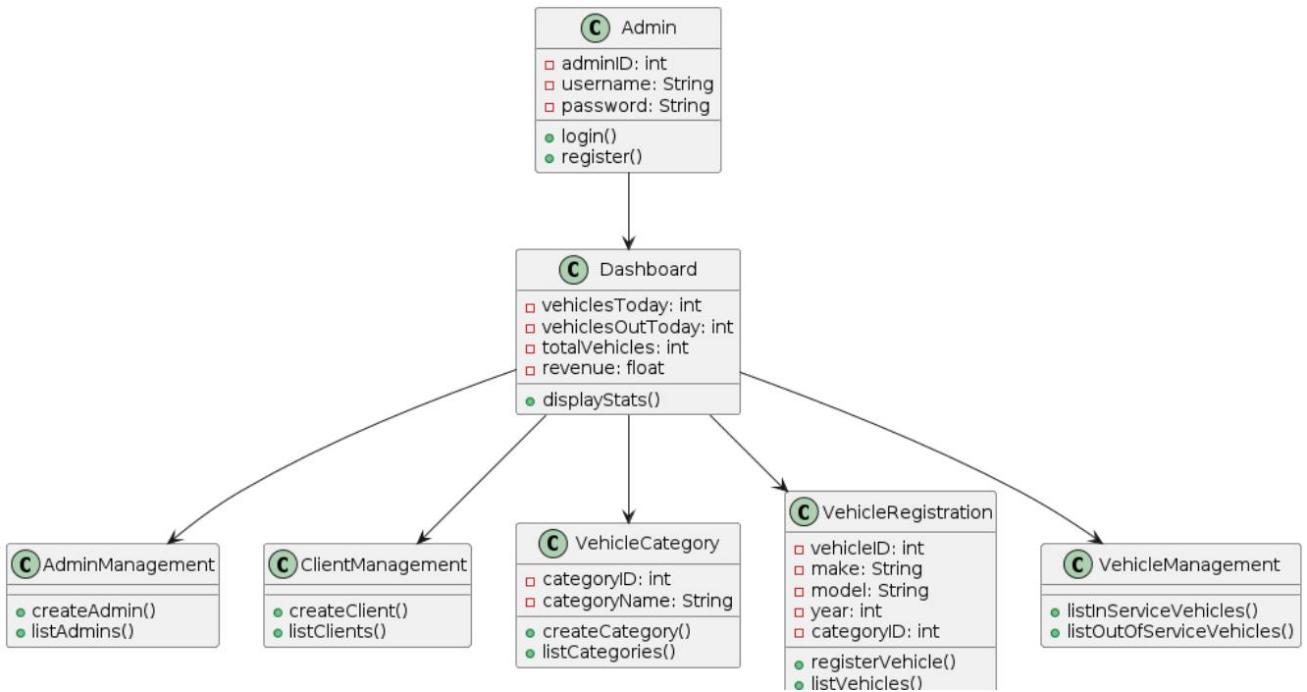


Figure 3 diagramme de classe

1. **Classe "Admin"** : Cette classe Admin définit un administrateur avec et deux méthodes : login pour permettre à l'administrateur de se connecter en vérifiant son nom d'utilisateur et son mot de passe, et register qui pourrait être utilisée pour s'enregistrer dans le système, mais ce n'est pas toujours nécessaire selon le contexte
2. **Classe " Dashboard"** : Cette classe représente un tableau de bord dans un système de gestion des véhicules. Elle contient des attributs tels que le nombre de véhicules aujourd'hui, le nombre de véhicules sortis aujourd'hui, le nombre total de véhicules et le revenu total. Le tableau

de bord est utilisé pour afficher des statistiques importantes sur l'état du système.

3. **Classe "VehicleRegistration" :** Cette classe VehicleRegistration représente l'enregistrement des véhicules dans un système de gestion. Elle contient des attributs tels que l'identifiant du véhicule, la marque, le modèle, l'année et l'identifiant de la catégorie. La classe inclut une liste de véhicules enregistrés et est utilisée pour stocker et gérer les informations relatives aux véhicules
4. **Classe "Véhicule Management" :** Cette classe gère les véhicules dans un système. Elle contient une liste de véhicules et deux méthodes principales pour lister les véhicules en service et li pour lister les véhicules hors service. Chaque véhicule est représenté par un objet de la classe, qui contient ses attributs permet d'ajouter de nouveaux véhicules à la liste.
5. **Classe " Client Management " :** Cette classe Client Management gère les clients dans un système. Elle contient une liste de clients et permet de stocker et de gérer les informations de chaque client, telles que l'identifiant du client, son nom, son adresse, son email et son numéro de téléphone.
6. **Classe "Admin Management" :** représente la gestion des administrateurs dans un système. Elle permet de créer et de gérer les administrateurs, incluant des fonctionnalités pour ajouter de nouveaux administrateurs et lister les administrateurs existants. Les administrateurs sont stockés dans une liste et chaque administrateur est représenté par un objet contenant un identifiant, un nom d'utilisateur et un mot de passe. Cette classe centralise la gestion

a. Diagramme de séquence :

a. Définition :

Le diagramme de séquence, décrit les scénarios de chaque cas d'utilisation en mettant l'accent sur la chronologie des opérations en interaction avec les objets. Un diagramme de séquence montre une interférence présentée en séquence dans le temps. En particulier, il montre aussi les objets qui participent à l'interaction par leur "ligne de vie" et les messages qu'ils échangent présentés en séquence dans le temps.

Voici quelques notions de base du diagramme :

- Scénario : une liste d'actions qui décrivent une interaction entre un acteur et le système.
- Interaction : un comportement qui comprend un ensemble de messages échangés par un ensemble d'objets dans un certain contexte pour accomplir une certaine tâche.
- Message : Un message définit une communication particulière entre des lignes de vie (objets ou acteurs).

b. Diagramme de séquence <>Authentification>> :

Le diagramme qui suit, présente l'enchaînement de la phase d'authentification.

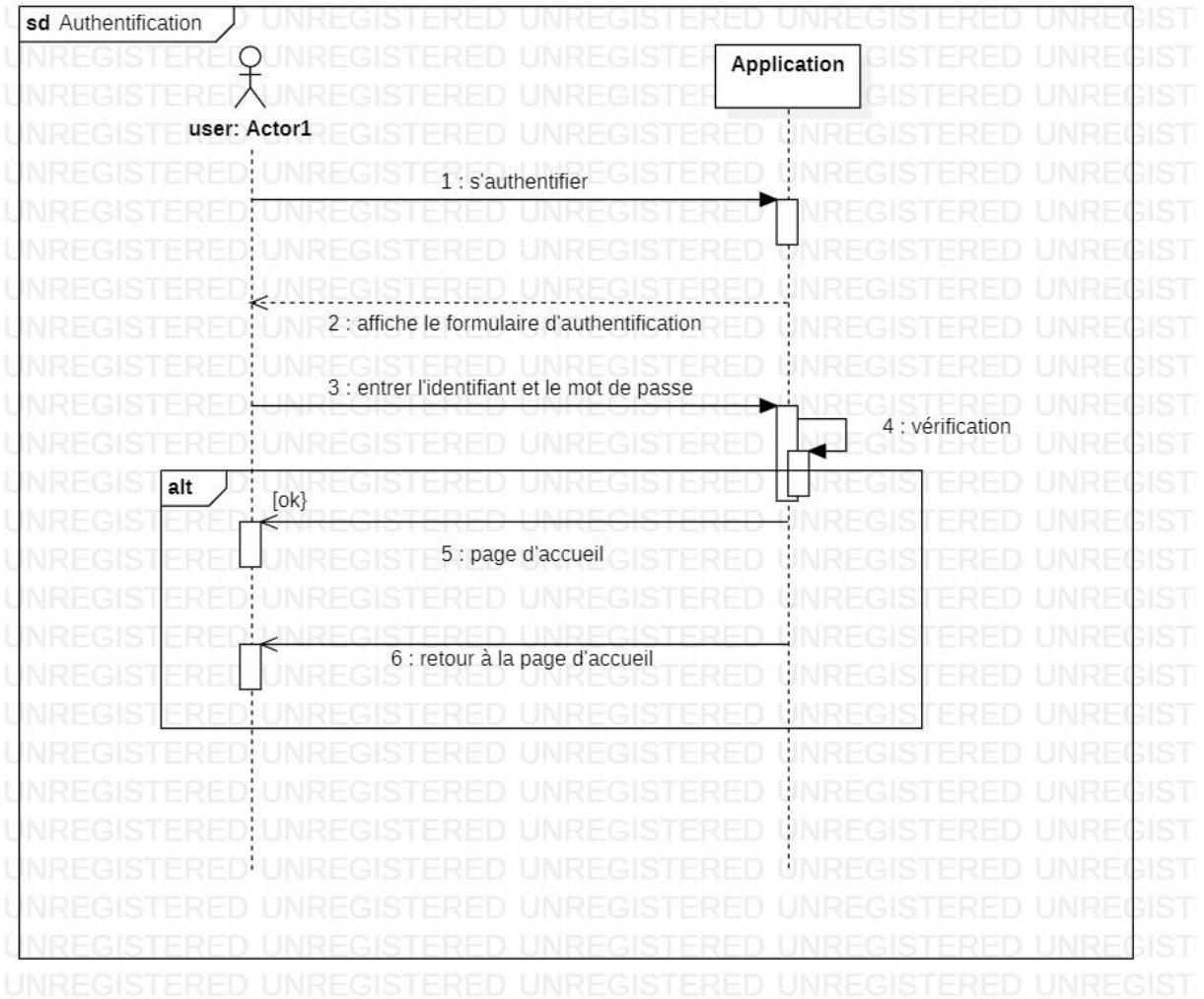


Figure 4 Diagramme de sequence d'authentification

Avant d'accéder à l'accueil de l'application et faire l'ensemble des autres scénarios l'utilisateur doit se connecter en utilisant son login et son mot de passe.

Objectif : la sécurité et la confidentialité de l'accès à l'application
Description:

Pour accéder à l'application, Admin doit tout d'abord s'identifier par son login et son mot de passe via le système qui prend en charge de vérifier les champs saisis par le technicien dans la base de données. S'il est accepté, donc il aura accès au système et aux applications du menu correspondant. Sinon, il doit vérifier ses données et s'identifie de nouveau s'il a déjà s'inscrire.

c. Diagramme de séquence de <<paramétrage des tranches>> :

Le cas d'utilisation "Configuration des zones de stationnement" permet à l'administrateur d'accéder à l'interface de l'application dédiée à la gestion du stationnement et de configurer les différentes zones de stationnement avec leurs tarifs associés. Les zones de stationnement permettent de définir des emplacements spécifiques de stationnement et d'appliquer des tarifs différents en fonction de ces emplacements

d. Diagramme de séquence :

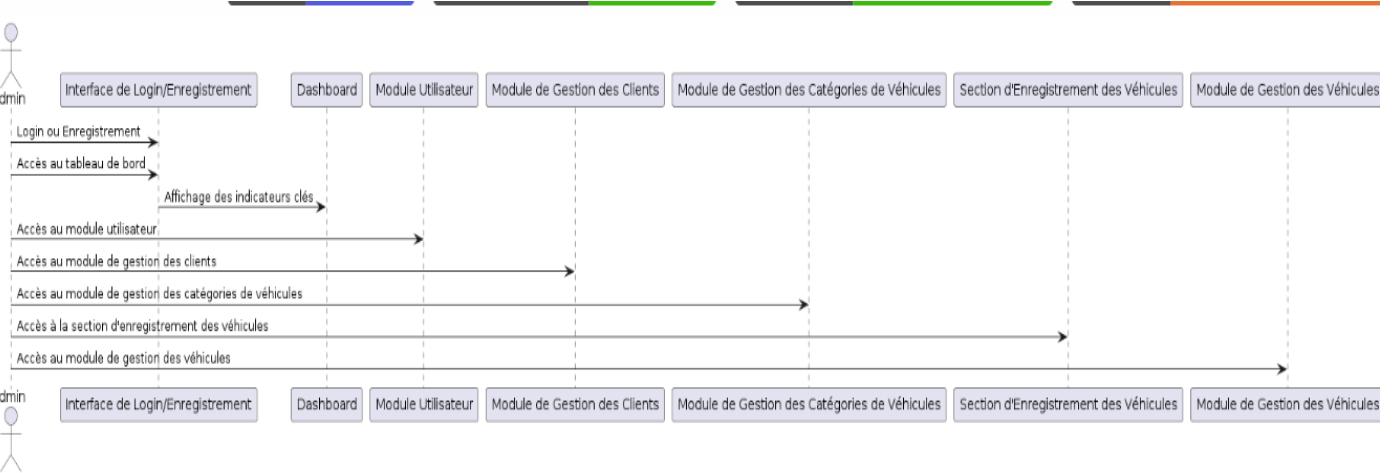


Figure 5 Diagramme de séquence <<paramétrage des tranches>>

Ce diagramme représente la gestion des utilisateurs dans notre application, géré par le gérant, décrit les interactions entre le gérant et les différentes fonctionnalités du système pour gérer les utilisateurs. Le gérant se connecte à l'application, effectue des actions telles que la consultation de la liste des utilisateurs, la recherche d'utilisateurs spécifiques, la modification des informations des utilisateurs existants et l'ajout de nouveaux utilisateurs. L'application effectue les vérifications nécessaires, récupère les informations de la base de données, et met à jour les données en conséquence. Le diagramme de séquence permet de visualiser de manière séquentielle ces interactions entre le gérant et le système, en montrant comment les informations sont échangées pour gérer efficacement les utilisateurs de notre application.

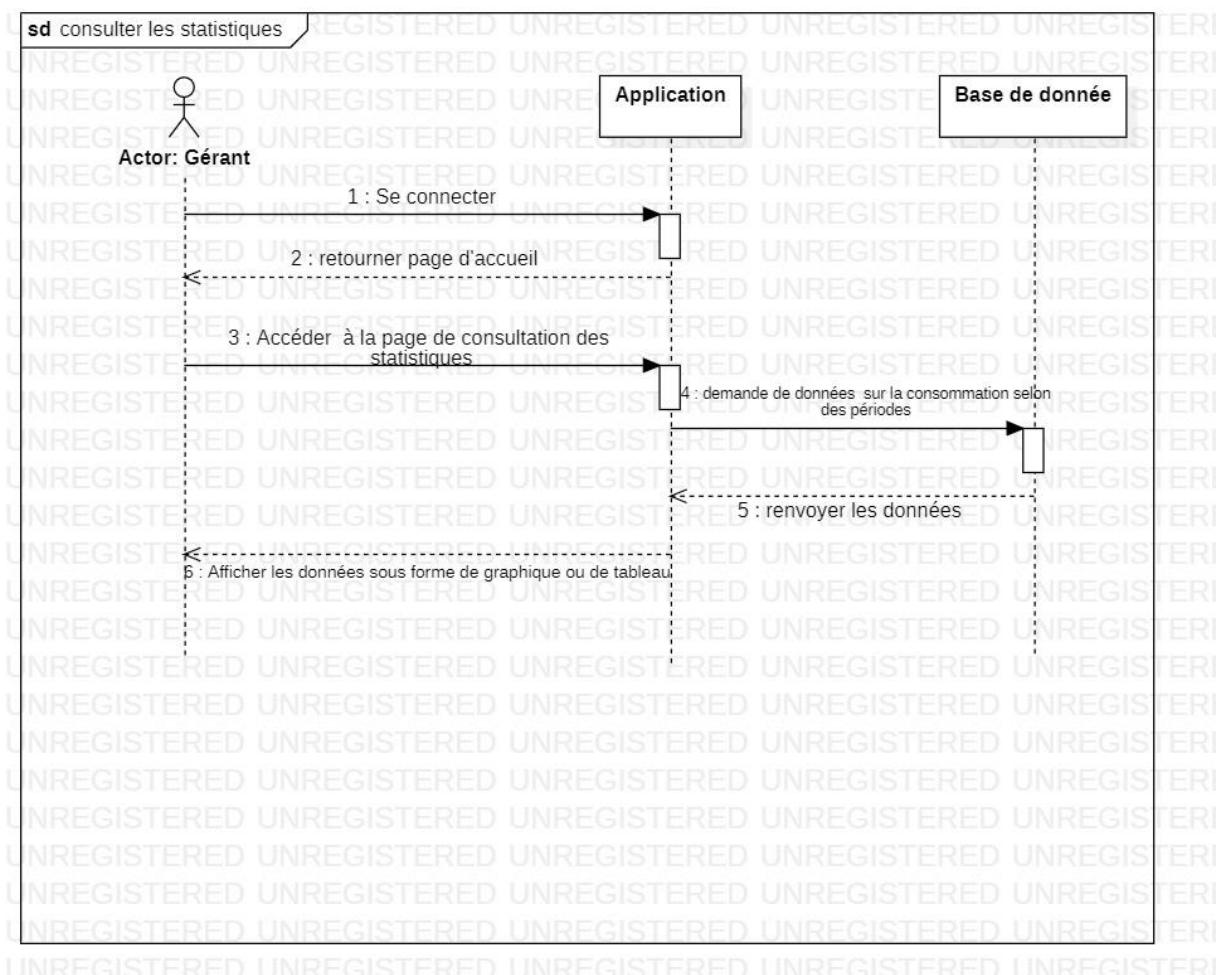


Figure 6 Diagramme de séquence de <<Consulter les statistiques>>

Le diagramme de séquence de consultation des statistiques, qui affiche les données sous forme de diagrammes graphiques, représente le flux d'interaction entre l'utilisateur et l'application lors de la consultation des statistiques. Le gérant lance l'application et accède à la fonctionnalité de consultation des statistiques, où les données pertinentes sont récupérées à partir de la base de données. L'application analyse ces données et les transforme en diagrammes graphiques, qui sont ensuite affichés au gérant. L'application met à jour les graphiques en fonction des actions de gérant, en reconfigurant les données et en les représentant visuellement. Lorsque gérant quitte la fonctionnalité, l'application termine la session. Le diagramme de séquence illustre de manière séquentielle ces interactions, montrant comment les données sont traitées et affichées sous forme de graphiques pour faciliter la consultation des statistiques.

Réalisation

Chapitre 3 : Réalisation

Introduction

Nous arrivons maintenant à la phase ultime. Cette dernière partie est la plus importante puisqu'elle met en réalité toute la théorie précédente. Dans un premier temps nous présentons l'environnement de réalisation sur le plan logiciel. Dans un second temps nous présentons quelques interfaces de notre application web.

I. Environnement de travail :

a. Environnement de développement intégré :

Visual Studio Code est un éditeur de code simplifié, qui est gratuit et développé en open source par Microsoft. Il fonctionne sous Windows, mac OS et Linux. Il fournit aux développeurs à la fois un environnement de développement intégré avec des outils permettant de faire avancer les projets techniques, de l'édition, à la construction, jusqu'au débogage.



Figure 7 Logo de l'éditeur Visual studio code

b. Outil d'administration de la base de données

L'implémentation de notre application se fera avec MYSQL sous l'environnement **XAMPP**. Il installe et configure automatiquement un environnement de travail complet sous Windows permettant de mettre en œuvre toute la puissance qu'offrent le langage dynamique PHP et son support efficace des bases de données.

XAMPP comprend le serveur Web Apache, PHP, MySQL, Perl, le serveur FTP et phpMyAdmin, le serveur de messagerie Mercury et le serveur JSP Tomcat.



Figure 8 Logo de serveur XAMPP

c. Langage de programmations :

❖ PHP :

Le PHP, pour Hypertext Preprocessor, désigne un langage informatique, ou un langage de script, utilisé principalement pour la conception de sites web dynamiques. Il s'agit d'un langage de programmation sous licence libre qui peut donc être utilisé par n'importe qui de façon totalement gratuite.

Créé au début des années 1990 par le Canadien et Groenlandais **Rasmus Lerdorf**, le langage PHP est souvent associé au serveur de base de données MySQL et au serveur Apache. Avec le système d'exploitation Linux, il fait partie intégrante de la suite de logiciels libres LAMP.

Sur un plan technique, le PHP s'utilise la plupart du temps côté serveur. Il génère du code HTML, CSS ou encore XHTML, des données (en PNG, JPG, etc.) ou encore des fichiers PDF. Il fait, depuis de nombreuses années, l'objet d'un développement spécifique et jouit aujourd'hui une bonne réputation en matière de fiabilité et de performances.

❖ HTML :

HTML (Hyper Text Markup Language / langage hypertexte) est le langage dans lequel sont écrites les pages du web. Un site web est constitué d'un ou plusieurs documents HTML. Pour se déplacer entre les pages dans nos modules on passe par l'intermédiaire d'hyperliens. Pour ajouter des objets graphiques on utilise le HTML d'autre part pour tester des pages web html en local, il suffit d'ouvrir le fichier dans un navigateur. Le HTML n'est pas un langage de programmation comme C++.

❖ JavaScript :

JavaScript est un langage de programmation de scripts principalement utilisé pour les pages web interactives comme les pages HTML. JavaScript est exécuté sur l'ordinateur de l'internaute par le navigateur lui-même. C'est une extension du langage HTML qui est incluse dans le code. Ce langage est un langage de programmation qui permet d'apporter des améliorations au langage HTML en permettant

d'exécuter des commandes. Ce code est directement écrit dans la page HTML, c'est un langage peu évolué qui ne permet aucune confidentialité au niveau des codes. Dans l'application nous avons codé plusieurs fonctions JavaScript par exemple : pour l'interaction des pages en envoyant des variables dans l'adresse URL pour filtrer le résultat de la requête en utilisant la méthode POST ou GET.

❖ CSS :

Les CSS, Cascading Style Sheets (feuilles de styles en cascade), permettent de mettre en forme des documents web, type page HTML ou XML. Par l'intermédiaire de propriétés d'apparence (couleurs, bordures, polices, etc.) et de placement (largeur, hauteur, côté à côté, dessus, dessous, etc.), le rendu d'une page web peut être entièrement modifié sans aucun code supplémentaire dans la page web. Les feuilles de styles ont d'ailleurs pour objectif principal de séparer le contenu de la page de son apparence visuelle

II. Choix de développement :

Pour le développement, nous avons appliqué le modèle MVC. Ce paradigme divise l'IHM (Interface Homme Machine) en un modèle (modèle de données) une vue (la présentation, l'interface utilisateur) et un contrôleur (la logique de contrôle, et la gestion des événements / synchronisation), chacun a un rôle bien précis.

L'architecture MVC ne résout pas tous les problèmes. Elle fournit souvent une première approche qui peut ensuite être adaptée et elle offre aussi un cadre pour structurer une application.

Ce maître d'architecture impose la séparation entre les données, la présentation et les traitements, ce qui nous donne trois parties fondamentales dans l'application : le modèle, la vue et le contrôleur.

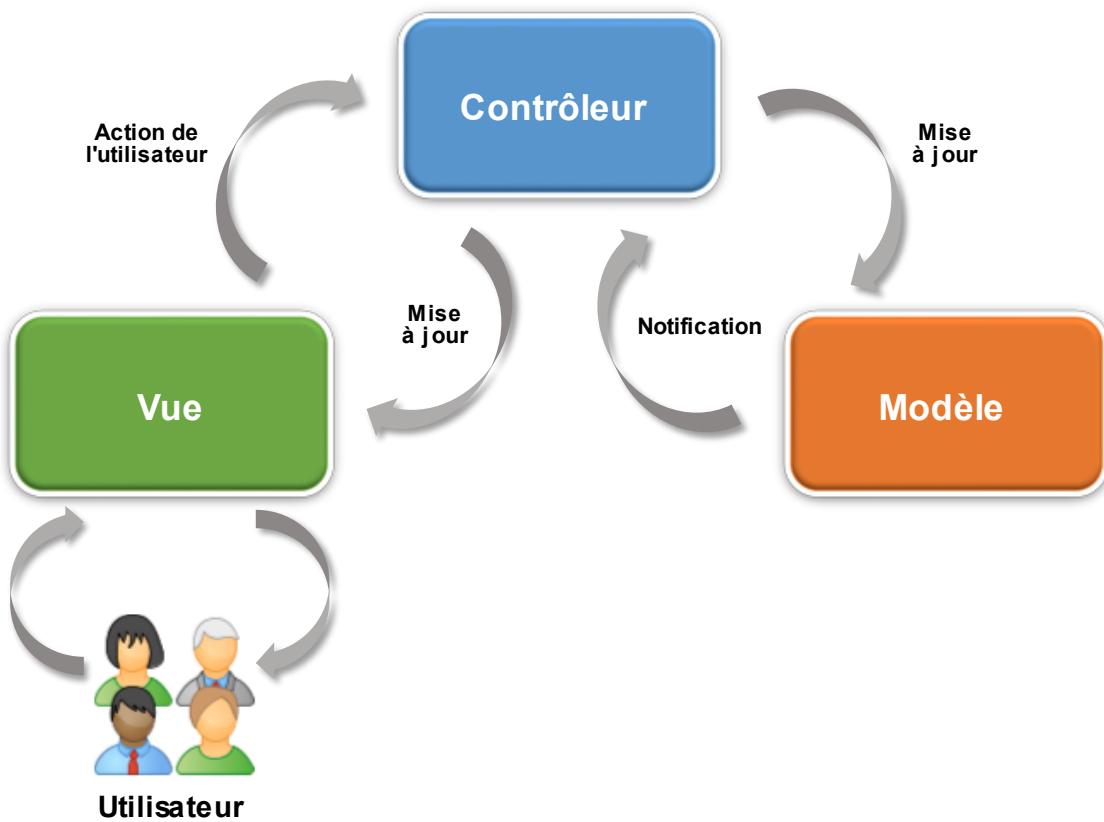


Figure 9 Schéma du Modèle MVC

1. Le modèle :

Le modèle indique le comportement de l'application : traitements des données, interactions avec la base de données. Il décrit l'emplacement des données manipulées par l'application et assure la gestion de ces données et garantit leur intégrité. Dans le cas spécifique d'une base de données, c'est le modèle qui la contient. Le modèle offre des méthodes pour mettre à jour ces données (ajout, suppression, changement de valeur). Il offre aussi des méthodes pour récupérer ces données. Les résultats renvoyés par le modèle sont dénués de toute présentation.

2. La vue :

La vue correspond à l'interface avec laquelle l'utilisateur interagit. Sa première tâche est de présenter les résultats renvoyés par le modèle. Sa seconde tâche est de recevoir toutes les actions de l'utilisateur (sélection d'une entrée, boutons, etc.). Ces événements sont envoyés au contrôleur. La vue n'effectue

aucun traitement, elle se satisfait d'afficher les résultats des traitements effectués par le modèle et d'interagir avec l'utilisateur.

3. Le contrôleur :

Le contrôleur prend en charge la gestion des événements de synchronisation pour mettre à jour la vue ou le modèle et les synchroniser. Il reçoit tous les événements de l'utilisateur et enclenche les actions à effectuer. Si une action nécessite un changement des données, le contrôleur demande la modification des données au modèle, et ce dernier informe la vue que les données ont changée pour qu'elle les mette à jour.

III. Les technologies utilisées :

❖ Laravel :

Laravel est un framework open source écrit en PHP, conçu pour le développement web. Laravel utilise une architecture MVC (Modèle-Vue-Contrôleur) pour organiser le code et faciliter la maintenance. Il offre également de nombreuses fonctionnalités intégrées, telles que l'authentification, la validation des formulaires, la gestion des sessions, la gestion des files d'attente, la gestion des emails, etc. qui facilitent le développement d'applications web complexes.

L'une des caractéristiques les plus appréciées de Laravel est son système de routage qui permet de définir facilement les URLs et les actions correspondantes dans l'application. Il offre également une base de données ORM (Object-Relational Mapping) appelée Eloquent qui permet de manipuler les données de la base de données de manière très simple.



Figure 10 Logo de framework Laravel

❖ Bootstrap :

Bootstrap est un framework front-end open source créé par Twitter pour le développement web. Il fournit des outils, des composants et des styles CSS préconçus pour aider les développeurs à créer des sites web et des applications web responsives rapidement et facilement.

Bootstrap utilise une grille de mise en page flexible et réactive qui permet aux développeurs de créer des designs adaptables à toutes les tailles d'écran, des ordinateurs de bureau aux téléphones mobiles. Il offre également de nombreux composants prêts à l'emploi tels que des menus de navigation, des boutons, des formulaires, des alertes, des modales, etc. qui peuvent être personnalisés en fonction des besoins de l'application.



Figure 11 Logo de framework Bootstrap

IV. Les interfaces graphiques :

L'interface graphique est une partie très importante pour la réalisation d'une application convenable offrant un certain plaisir à l'utilisateur lors de sa navigation. Ainsi, ce critère peut faire la différence entre une application et un autre bien qu'elles aient les mêmes fonctionnalités.

Voici un ensemble de captures d'écrans sur les principaux points d'entrées de l'application :

⇒ Page d'authentification :

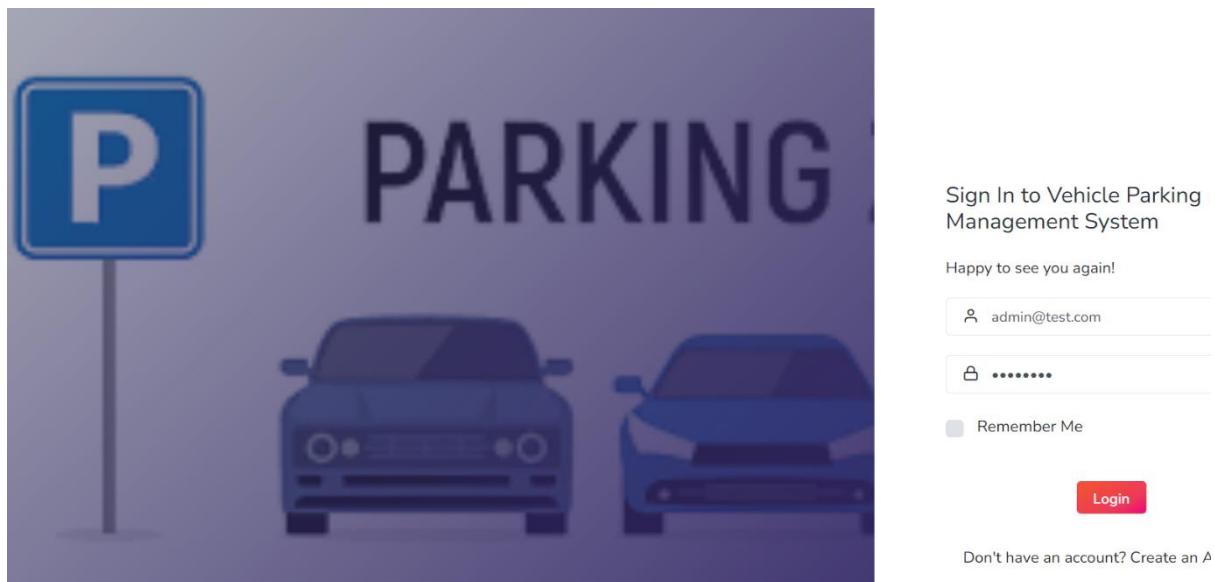


Figure 12 Page d'authentification

Pour pouvoir accéder aux différentes fonctionnalités de l'application, l'administrateur doit taper son login et son mot de passe dans les champs correspondants. Une fois que le client a cliqué sur le bouton « Login », le système vérifie les données entrées. En cas d'échec, il réaffiche la page D'authentification avec un message d'erreur. Si le Login et le mot de passe sont acceptables, le système passe au menu principal.

⇒ Interface de Créez un nouveau compte si vous n'en avez pas:

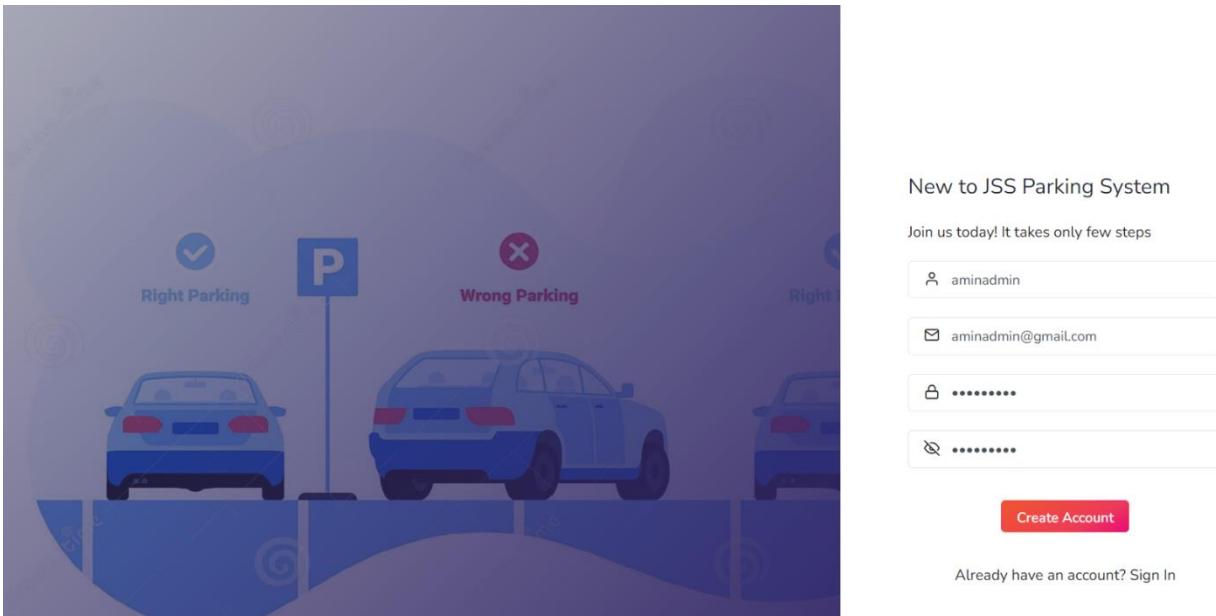


Figure 13 Interface nouveau compte

L'interface de création d'un nouveau compte offre à l'utilisateur la possibilité de créer un compte s'il n'en possède pas déjà un. Cette interface permet à l'utilisateur de fournir les informations nécessaires pour créer son compte, telles que son nom, son adresse e-mail, son numéro de téléphone et toute autre information requise. Une fois que l'utilisateur a saisi ses informations, il peut soumettre le formulaire de création de compte pour finaliser le processus. Cette interface fournit une expérience conviviale et intuitive pour permettre à de nouveaux utilisateurs de rejoindre la plateforme facilement et rapidement.

① Partie de gérant :

a. Page de Dashboard :

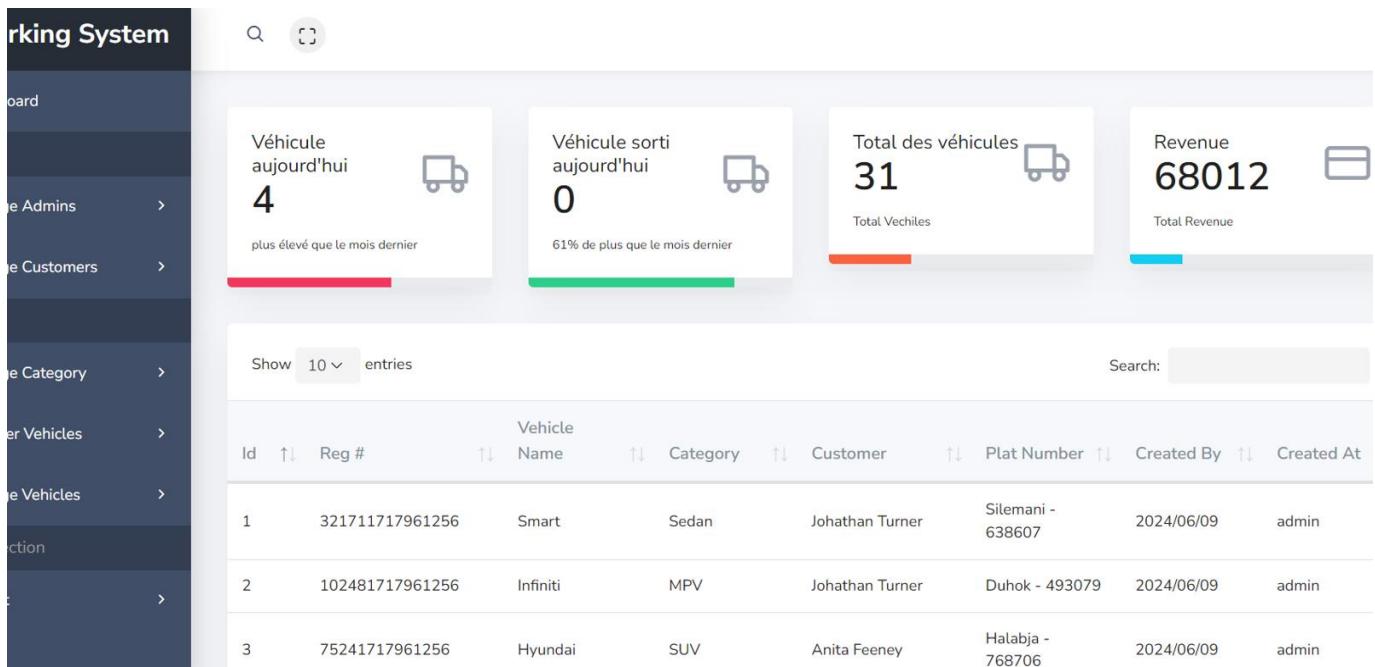


Figure 14 interface Dashboard

L'interface du tableau de bord pour la gestion du stationnement est un outil centralisé qui permet au responsable du stationnement d'accéder et de consulter les informations relatives à la gestion des emplacements de stationnement. Cette interface offre une vue d'ensemble claire et organisée des différents aspects de la gestion du stationnement, permettant au responsable de surveiller efficacement l'état global du système.

À travers cette interface, le responsable du stationnement peut visualiser des indicateurs clés tels que le nombre total de places de stationnement, le taux d'occupation actuel, les revenus générés, etc. De plus, le tableau de bord offre la possibilité de consulter des données détaillées sur les transactions de stationnement, les emplacements occupés, les paiements reçus et d'autres informations pertinentes. Cela permet au responsable du stationnement d'avoir une vue d'ensemble complète de la situation et de prendre les décisions appropriées pour assurer un fonctionnement efficace du système de gestion du stationnement.

b. Int Manage Admins cérat :

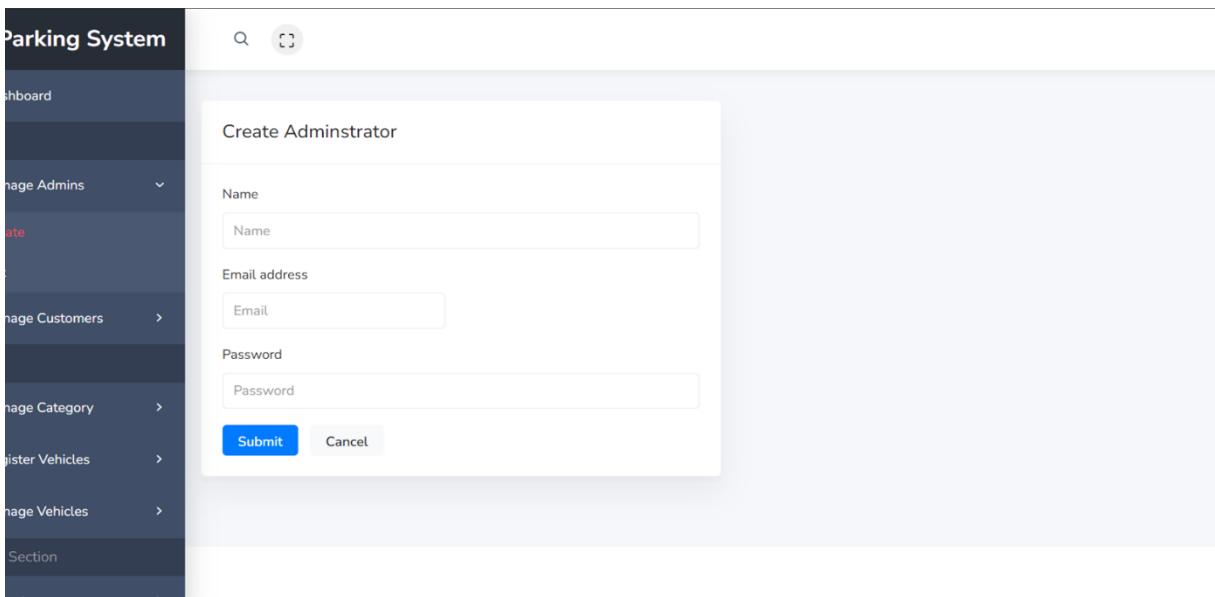


Figure 15 Interface Admins cérat

Lorsque vous sélectionnez un utilisateur dans l'application de gestion du stationnement, l'interface affiche les transactions associées à cet utilisateur, en distinguant clairement les transactions payées et les transactions impayées. Cette fonctionnalité permet à l'administrateur de voir toutes les transactions spécifiques à cet utilisateur et de suivre facilement leur statut de paiement.

De même, les transactions impayées sont également affichées dans une section distincte. Elles sont présentées avec les informations essentielles telles que la date de la transaction, le montant dû et toute autre information pertinente. Cette distinction visuelle entre les transactions payées et impayées permet à l'administrateur de repérer rapidement les transactions en attente de paiement et de prendre les mesures appropriées pour relancer les paiements en souffrance..

c. Interface de Admin List :

The screenshot shows the 'Admin List' page of a parking system application. The left sidebar has a dark theme with white text and icons. It includes links for Dashboard, Manage Admins (selected), Manage Customers, Manage Category, Register Vehicles, Manage Vehicles, and a Section link. The main content area has a light gray background. At the top, there's a search bar and a refresh icon. Below it, the title 'Admin List' is displayed next to a car icon. To the right, there are navigation links for Home, Tables, and Details. The main section is titled 'Administrators List'. It features a table with columns: Id, Name, Email, and Created At. The table contains three entries:

Id	Name	Email	Created At	Operation
1	admin	admin@test.com	2024/06/09	
2	amine	amine@gmail.com	2024/06/09	
3	aminadmin	aminadmin@gmail.com	2024/06/11	

Below the table, a message says 'Showing 1 to 3 of 3 entries'. At the bottom right, there are buttons for 'Previous' and '1'.

Figure 16 Interface de Admin List

L'interface de l'application de gestion du stationnement affiche une liste des administrateurs avec leurs informations associées. Cette liste offre une vue claire et organisée de tous les administrateurs du système, permettant à l'administrateur principal de gérer efficacement les utilisateurs et leurs rôles.

d. Interface de crée un utilisateur:

The screenshot shows a user interface for creating a new administrator. On the left, there is a dark sidebar menu with various options like 'Dashboard', 'Manage Admins', 'Manage Customers', 'Manage Category', 'Manage Vehicles', and 'Section'. The main area has a title 'Create Adminstrator' and contains four input fields: 'Name' (with placeholder 'Name'), 'Email address' (with placeholder 'Email'), 'Phone' (with placeholder 'Phone'), and 'Company' (with placeholder 'Company'). Below these fields is a large 'Address' input area with a scroll bar. At the bottom of the form are two buttons: 'Submit' (in blue) and 'Cancel'.

Figure 17 Interface de crée un utilisateur

L'interface de création d'utilisateur offre à l'administrateur la possibilité de créer de nouveaux utilisateurs dans le système. Cette interface fournit un formulaire convivial où l'administrateur peut saisir les informations nécessaires pour créer un nouvel utilisateur.

Le formulaire comprend des champs pour le nom de l'utilisateur, son adresse e-mail, son numéro de téléphone, son adresse physique et d'autres détails pertinents. De plus, l'interface permet à l'administrateur de définir le rôle et les autorisations de l'utilisateur nouvellement créé, lui permettant d'accéder aux fonctionnalités appropriées dans le système.

e. Interface de voir les listes de utilisateur :

Id	Name	Email	Address	Company	Phone	Created
1	Verdie Torp	myles.kuvalis@hayes.com	84292 McLaughlin Shore Suite 223 Lueilwitzview, WV 54177	Durgan Inc	866.830.6347	2024/06/01
2	Nedra Haag	ebaumbach@gmail.com	887 Bethany Harbor Apt. 110 Miracleview, WV 60550	Bernhard, Borer and Leffler	1-855-225-6917	2024/06/01
3	Anita Feeney	hillard14@gmail.com	3386 Corkery Squares Suite 131 Humbertofurt, VA 14855-9492	Nolan and Sons	1-800-882-1213	2024/06/01
4	Mr. Craig VonRueden V	ashley.williamson@kautzer.com	906 Keira Burgs Lake Anorlfurt, MS 23558	Barrows, Wisoky and Berrnaum	(866) 680-8643	2024/06/01

Figure 18 Interface de voir des listes d'un utilisateur

L'interface de visualisation des listes d'utilisateurs offre à l'administrateur une vue complète et organisée de tous les utilisateurs enregistrés dans le système. Cette interface présente les utilisateurs sous forme de liste, avec leurs informations principales affichées de manière claire et concise.

Chaque utilisateur est répertorié avec des détails tels que son nom, son adresse e-mail, son numéro de téléphone et éventuellement d'autres informations pertinentes. De plus, l'interface offre des fonctionnalités de filtrage et de recherche, permettant à l'administrateur de trouver rapidement des utilisateurs spécifiques en fonction de certains critères.

f. Interface de création catégorie de voiture :

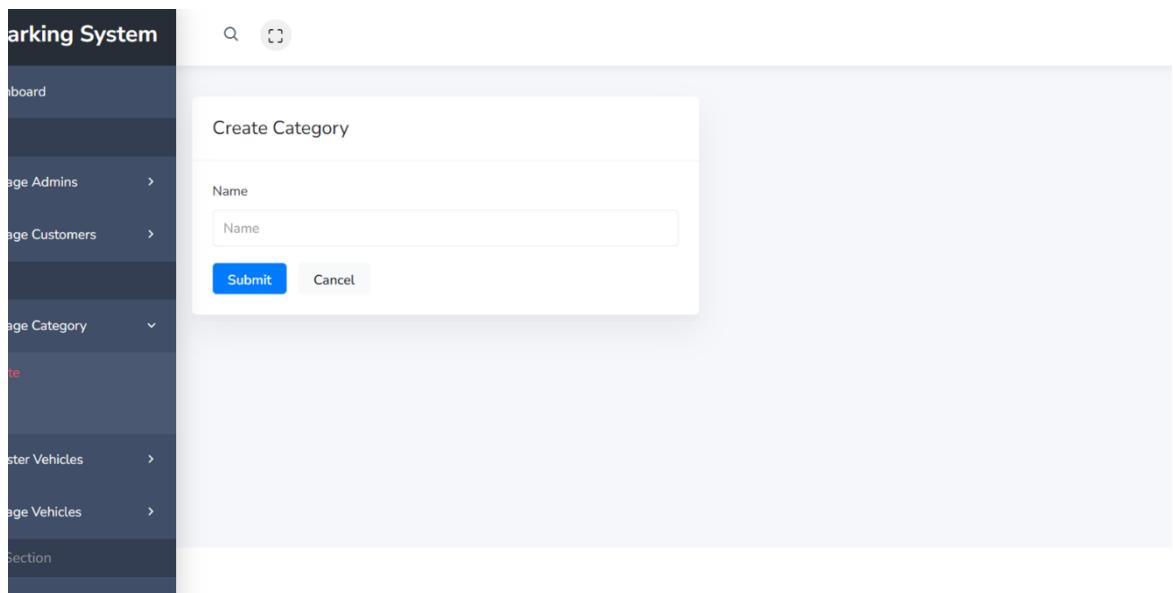


Figure 19 Interface de création catégorie de voiture

L'interface de création de catégorie de voiture offre à l'administrateur la possibilité de définir de nouvelles catégories pour les véhicules dans le système de gestion du stationnement. Cette interface présente un formulaire où l'administrateur peut saisir les détails de la nouvelle catégorie.

Category List

Id	Name	Created By	Created At	Operation
1	Van	admin	2024/06/09	
2	Taxi	admin	2024/06/09	
3	Bus	admin	2024/06/09	
4	Hatchback	admin	2024/06/09	
5	Sedan	admin	2024/06/09	
6	SUV	admin	2024/06/09	

Figure 20 list des category

g. Interface de création Véhicule :

The screenshot shows a user interface for creating a vehicle. On the left, there is a sidebar with a dark background containing navigation items: 'Dashboard', 'Manage Admins', 'Manage Customers', 'Manage Category', 'Manage Vehicles', 'Manage Vehicles', and 'Logout'. The main area has a light gray background and is titled 'Create Vehicle'. It contains several input fields and dropdown menus:

- 'Registration Number' field with placeholder 'Registration Number Auto'.
- 'Category' dropdown menu with placeholder 'Select'.
- 'Customer Name' dropdown menu with placeholder 'Select'.
- 'Vehicle Name' field with placeholder 'Vehicle Name'.
- 'Vehicle Plat Number' field with placeholder 'Vehicle Plat Number'.
- 'Parking Duration (in hours)' field with placeholder 'Parking Duration'.
- 'Parking Charges (IQD)' field with placeholder 'Parking Charges'.

At the bottom left is a blue 'Submit' button, and at the bottom right is a 'Cancel' link.

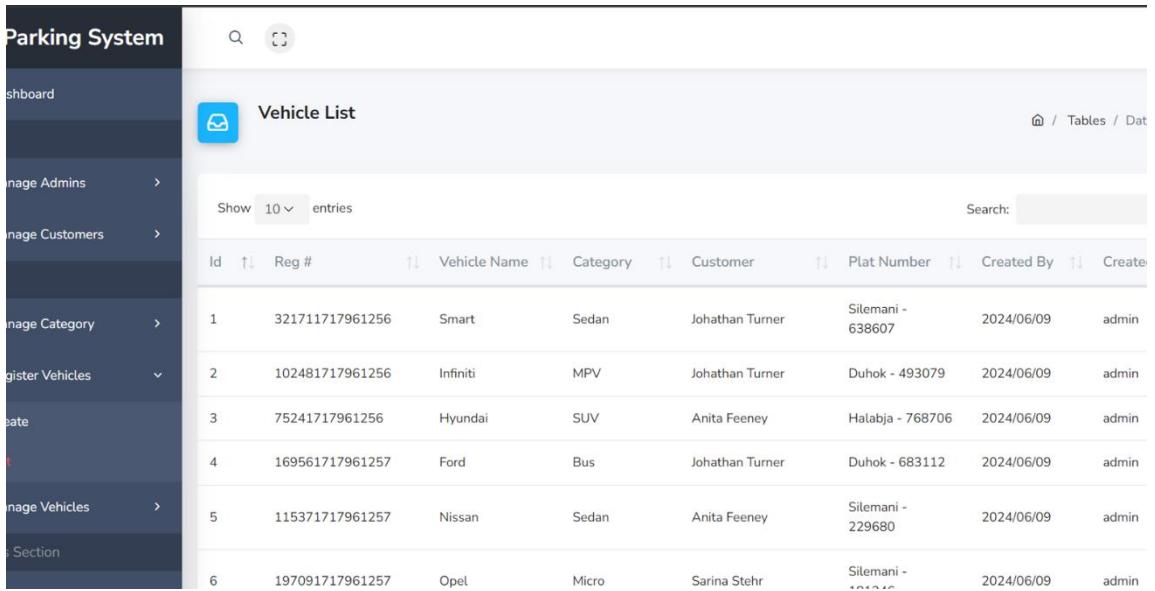
Figure 21 interface de création Véhicule

L'interface de création de véhicule offre à l'administrateur la possibilité d'ajouter de nouveaux véhicules au système de gestion du stationnement. Cette interface présente un formulaire où l'administrateur peut saisir les détails du nouveau véhicule.

Le formulaire comprend généralement des champs pour les informations suivantes :

- Marque du véhicule
- Modèle du véhicule
- Année de fabrication
- Catégorie du véhicule (sélectionnée à partir des catégories préalablement définies)
- Autres détails spécifiques selon les besoins du système

h. Interface de Liste Véhicule :



The screenshot shows the 'Vehicle List' page of a web application. The left sidebar has a dark blue background with white text and icons. It includes links for Dashboard, Manage Admins, Manage Customers, Manage Category, Register Vehicles, Create, Manage Vehicles, and a Section link. The main area has a light gray header with a search bar and a vehicle icon. Below the header is a table titled 'Vehicle List' with 6 rows of data. The table columns are: Id, Reg #, Vehicle Name, Category, Customer, Plat Number, Created By, and Create. The data in the table is as follows:

Id	Reg #	Vehicle Name	Category	Customer	Plat Number	Created By	Create
1	321711717961256	Smart	Sedan	Johathan Turner	Silemani - 638607	2024/06/09	admin
2	102481717961256	Infiniti	MPV	Johathan Turner	Duhok - 493079	2024/06/09	admin
3	75241717961256	Hyundai	SUV	Anita Feeney	Halabja - 768706	2024/06/09	admin
4	169561717961257	Ford	Bus	Johathan Turner	Duhok - 683112	2024/06/09	admin
5	115371717961257	Nissan	Sedan	Anita Feeney	Silemani - 229680	2024/06/09	admin
6	197091717961257	Opel	Micro	Sarina Stehr	Silemani - 123456	2024/06/09	admin

Figure 22 Interface de Liste Véhicule

L'interface de liste des véhicules offre à l'administrateur une vue complète et organisée de tous les véhicules enregistrés dans le système de gestion du stationnement. Cette interface présente les véhicules sous forme de liste, avec leurs détails principaux affichés de manière claire et concise.

Chaque véhicule est répertorié avec des informations telles que la marque, le modèle, l'année de fabrication, la catégorie du véhicule, et éventuellement d'autres détails pertinents. De plus, l'interface offre des fonctionnalités de filtrage et de recherche, permettant à l'administrateur de trouver rapidement des véhicules spécifiques en fonction de certains critères.

i. Interface de Véhicules in List et Ajouter new Véhicules :

The screenshot shows the 'Vehicles In List' page of a parking system. On the left, there's a sidebar with navigation links: Dashboard, Site Admins, Site Customers, Site Category, Manage Vehicles, and Vehicles. The main area has a title 'Vehicles In List' with a car icon and a red 'Add New Vehicle' button. Below it, there are two tabs: 'Current In Vehicles' (selected) and 'Vehicles History'. A search bar and a 'Show 10 entries' dropdown are also present. The main content is a table with columns: Id, Reg #, Vehicle Name, Parking Area, Parking Number, Created At, and Created By. It lists four vehicles: Mazda (Blok C, 32698), Smart (Blok B, 5372), Daihatsu (Blok A, 5391), and Rover (Blok C, 1741). At the bottom, it says 'Showing 1 to 4 of 4 entries'.

Id	Reg #	Vehicle Name	Parking Area	Parking Number	Created At	Created By
1	148341717961257	Mazda	Blok C	32698	2024/06/09 00:00 AM	admin
2	321711717961256	Smart	Blok B	5372	2024/06/09 00:00 AM	admin
3	243471717961257	Daihatsu	Blok A	5391	2024/06/09 00:00 AM	admin
4	305891717961257	Rover	Blok C	1741	2024/06/09 00:00 AM	admin

Figure 23 Interface de Véhicules in List et Ajouter new Véhicules

L'interface des véhicules en liste affiche tous les véhicules actuellement enregistrés dans le système de gestion du stationnement. Chaque véhicule est présenté avec ses détails principaux, tels que la marque, le modèle, l'année de fabrication et la catégorie. Cette interface permet à l'utilisateur de visualiser rapidement tous les véhicules disponibles dans le système.

En plus de la liste des véhicules existants, l'interface propose également une option pour ajouter un nouveau véhicule. L'utilisateur peut accéder à cette fonctionnalité en cliquant sur un bouton "Ajouter un nouveau véhicule" ou un lien similaire. En cliquant sur cette option, l'utilisateur est dirigé vers un formulaire où il peut saisir les détails du nouveau véhicule à ajouter, tels que la marque, le modèle, l'année de fabrication, la catégorie.

Id	Reg #	Vehicle Name	Parking Area	Parking Number	Created At	Created By	Operation	Action
1	235211717961257	Bentley	Blok B	8199	2024/06/09 00:00 AM	admin	<input type="checkbox"/>	<input type="checkbox"/>
2	75241717961256	Hyundai	Blok A	2631	2024/06/09 00:00 AM	admin	<input type="checkbox"/>	<input type="checkbox"/>
3	187791717961257	Daewoo	Blok C	23351	2024/06/09 00:00 AM	admin	<input type="checkbox"/>	<input type="checkbox"/>
4	141461717961257	Nissan	Blok A	32826	2024/06/09 00:00 AM	admin	<input type="checkbox"/>	<input type="checkbox"/>
5	197091717961257	Opel	Blok D	31853	2024/06/09 00:00 AM	admin	<input type="checkbox"/>	<input type="checkbox"/>

Figure 24 Interface de véhicule out List

L'interface de la liste des véhicules hors service affiche tous les véhicules qui sont actuellement indisponibles ou hors d'usage dans le système de gestion du stationnement. Chaque véhicule est présenté avec ses détails principaux, tels que la marque, le modèle, l'année de fabrication et la catégorie, ainsi que des informations supplémentaires sur la raison de leur indisponibilité.

a. Glossaire

- UML : Unified Modeling Language
- PHP : Personal Home Page
- MVC : model-view-controller
- IDE : integrated development environment

Webographie

<https://fr.slideshare.net/sof1105/rapport-projet-de-fin-dtude-dveloppent-dune-application-web-avec-symfony2>

date consultation : entre 09/05/2024 et 15/05/2024

<https://stackoverflow.com/>

date consultation : 20/05/2024

<https://laravel.com/docs/8.x>

date consultation : 29/05/2024

Conclusion

Représente une solution innovante et efficace pour surmonter les défis associés à la gestion traditionnelle des parkings. Grâce à l'utilisation de technologies de pointe et à une approche de développement agile, nous avons réussi à créer une application qui simplifie la gestion des véhicules, la tarification et le suivi de l'utilisation des places de stationnement. Cette application contribue à réduire les erreurs administratives, à améliorer l'efficacité opérationnelle et à garantir la sécurité des données des utilisateurs. Elle marque une avancée significative dans l'amélioration de la gestion du stationnement urbain, offrant des avantages en termes de facilité d'utilisation, de rapidité et de fiabilité. En résumé, notre application favorise une gestion plus efficace, transparente et sécurisée des parkings, répondant aux besoins spécifiques des municipalités urbaines.