

Faculté des Sciences Exactes et d'Informatique

Département de Mathématiques et informatique

Filière : Informatique

MEMOIRE DE FIN D'ETUDES

Pour l'Obtention du Diplôme de Master en Informatique

Option : Intelligence Artificielle pour l'internet des objets (IA4IoT)

Présenté par :

- Beressa Anis
- Meddah Ayoub

THEME :

**Proposition d'une solution basée sur les technologies IoT pour la
gestion de parking multi-niveaux**

Soutenu le :

Devant le jury composé de :

Mr Mohamed Habib Zahmani	MCA	Université de Mostaganem	Président
Mme Meriem Abid	MCA	Université de Mostaganem	Examineur
Mme Nacera Bahnes	MCB	Université de Mostaganem	Encadreur

Année Universitaire 2024-2025

Résumé

Les parkings en plusieurs niveaux représentent un trait de la mobilité urbaine moderne face à des enjeux de plus en plus complexes d'occupation, de fluidité et de gestion intelligente. Ce mémoire propose la conception et l'évaluation d'un système de gestion de stationnement utilisant les technologies de l'Internet des Objets (IoT) et de l'intelligence artificielle pour superviser, optimiser et simuler en temps réel l'allocation des places. L'ensemble du dispositif est constitué d'une architecture complète intégrant des capteurs simulés, une interface web, un module de réservation et un module de prédiction à base de XGBoost. Ce dernier modèle d'apprentissage permet de prévoir le taux d'occupation futur et de filtrer automatiquement les demandes de réservation à risque selon un seuil de congestion fixé. En parallèle, une approche expérimentale est explorée via un second modèle XGBoost permettant de proposer une place optimale pour chaque véhicule sans règles déterministes. L'objectif de ce travail est de comparer les performances des différentes stratégies dans une simulation afin d'évaluer leurs impacts sur la fluidité interne, la cohérence des affectations et l'intelligence du système global.

Abstract

Multi-level parking structures are a key component of modern urban mobility, addressing increasingly complex challenges related to space occupancy, internal flow, and intelligent management. This thesis presents the design and evaluation of a smart parking management system that leverages Internet of Things (IoT) technologies and artificial intelligence to monitor, optimize, and simulate real-time space allocation. The system is built upon a complete architecture that includes simulated sensors, a web interface, a reservation module, and a predictive engine based on the XGBoost algorithm. This learning model is used to forecast future occupancy rates and automatically filter high-risk reservation requests according to a predefined congestion threshold. In parallel, an experimental approach is explored through a second XGBoost model, which suggests an optimal parking spot for each vehicle without relying on deterministic rules. The goal of this work is to compare the performance of these different strategies in simulation in order to assess their impact on internal flow, allocation consistency, and the overall intelligence of the system.

Dédicace et remerciements

Tout d’abord, nous remercions Allah, pour nous avoir accordé la force, la patience et les moyens d’achever ce travail dans les meilleures conditions.

Nous souhaitons exprimer nos sincères remerciements à toutes les personnes qui ont contribué à la réalisation de ce projet de fin d’études.

Nous adressons une reconnaissance particulière à notre encadrante universitaire, Mme Bahnes Nacera, pour son accompagnement attentif, sa disponibilité et ses conseils précieux tout au long du projet. Son encadrement nous a été d’une grande aide dans l’avancement de notre travail.

Nous remercions également nos familles et nos amis pour leur soutien moral, leur patience et leurs encouragements constants. Leur présence à nos côtés a été essentielle dans les moments exigeants de ce parcours.

Ce mémoire leur est dédié, en signe de gratitude et de respect.

Liste des figures

Figure N : Titre de la figure	Page
Figure 01 : Barrière de parking	12
Figure 02 : Borne de paiement	12
Figure 03 : Détection et reconnaissance des plaques d'immatriculation	14
Figure 04 : Fonctionnement de la technologie LoRa	18
Figure 05: Architecture MQTT	19
Figure 06: Architecture Zigbee	19
Figure 07 : Architecture NB-IoT	20
Figure 08 : système de stationnement intelligent dans un environnement VANET	21
Figure 09 : Parking multiniveaux de Sahid Nagar, Bhubaneswar	26
Figure 10 : Intérieur du parking multiniveaux de Lusail, Qatar	27
Figure 11 : Interface de simulation du stationnement	29
Figure 12 : Visualisation du simulateur d'occupation sur campus universitaire	30
Figure 13 : diagramme d'états illustrant un cycle de vie d'une place de parking simulé	46
Figure 14 : outils utilisé	49
Figure 15 : Architecture fonctionnelle du système proposé	53
Figure 16 : organigramme de la Simulation du Trafic Véhiculaire dans le Parking Intelligent	54
Figure 17 : Log de simulation illustrant l'arrivée d'un véhicule dans le mode centre commercial	56
Figure 18: Changement de l'état d'une place (orange → rouge) dans l'interface de gestion de parking	57
Figure 19 : Interface principale du tableau de bord du gestionnaire	58
Figure 20 : Interface simuler sortie	58
Figure 21 : Interface statistiques détaillées	59
Figure 22 : Libération automatique de la place simulée au terme de la durée	60
Figure 23 : interface de connexion utilisateur	60
Figure 24 : Formulaire de réservation d'une place de stationnement	61
Figure 25 : visualisation d'une place réservée (affichée en bleu) dans l'interface gestionnaire	62
Figure 26 : interface liste réservation	62
Figure 27 : enregistrement de réservations dans la base de données MongoDB	63
Figure 28 : diagramme de séquence occupation d'une place dans le système de gestion intelligent	63
Figure 29 : Interface utilisateur – refus automatique suite a la prédiction	64
Figure 30 : Journal système au moment de la réservation	65
Figure 31 : Courbe de prédiction de l'occupation du parking	65
Figure 32 : Occupation réelle vs prédite du parking	68

Liste des tableaux

Tableau N : titre du tableau	Page
Tableau 01 : comparaison entre les capteurs	17
Tableau 02 : comparaison entre différentes technologies de communication	22
Tableau 03 : d'exemple d'application de stationnement intelligent	23
Tableau 04 : comparaison entre les trois études analysées	32
Tableau 05 : Statistiques globales de la simulation	66
Tableau 06 : Répartition des véhicules par étage	66
Tableau 07 : Comparaison entre les prédictions du modèle et les données réelles d'occupation	68
Tableau 08 : Répartition des véhicules par niveau selon l'approche d'attribution	69

Table des Matières

Introduction générale	9
Chapitre 1 : Évolution et technologies des parkings intelligents.....	10
1.1 Introduction.....	10
1.2 Historique et évolution des systèmes de stationnement.....	10
1.2.1 La gestion traditionnelle des premiers parkings.....	10
1.2.2 Problème des parkings traditionnels.....	11
1.2.3 Le progrès des parkings automatisés.....	11
1.2.3.1 Barrières automatiques.....	11
1.2.3.2 Bornes de paiement.....	12
1.2.4 Introduction au parking intelligent	13
1.2.4.1 L'intégration de l'IoT et de l'IA	13
1.2.4.1.1 Intégrations de l'IA.....	13
1.2.4.1.2 Intégrations de l'IoT.....	14
1.2.5 Les parkings multi-niveaux.....	14
1.3 Fonctionnement des parkings intelligents.....	15
1.3.1 Détection et gestion des places	15
1.3.2 Paiement et réservation.....	15
1.4 Technologies clés utilisées dans les parkings intelligents.....	16
1.4.1 Capteurs IoT.....	16
1.4.2 Communication et transmission des données.....	18
1.4.3 Intelligence Artificielle.....	22
1.4.4 Services et technologies dans les applications de stationnement.....	23
1.5 Impact et bénéfices des parkings intelligents.....	24
1.6 Conclusion.....	25
Chapitre 2 : État de l'art.....	26
2.1 Introduction.....	26
2.2 Applications concrètes du smart parkings multiniveaux dans différentes villes.....	26
2.3 Études des solutions proposées pour les parkings multi-niveaux.....	28
2.4 Contraintes spécifiques et limites des solutions existantes.....	33
2.4.1 Contraintes technologiques.....	33
2.4.2 Contraintes environnementales.....	34

2.5 Problématique.....	34
2.6 Conclusion.....	34
Chapitre 3 : Méthodologie.....	34
3.1 Introduction.....	34
3.2 Solution proposé.....	34
3.3 Préconditions et hypothèses de simulation	35
3.4 Diagramme d'états.....	37
3.5 Mécanismes d'optimisation implémentés.....	37
3.6 Intégration de l'intelligence artificielle dans la prise de décision	46
3.6.1 Objectif du modèle prédictif	46
3.6.2 Données d'entrée.....	47
3.6.3 Choix de XGBoost.....	47
3.6.4 Bibliothèques utilisées.....	47
3.7 Environnement de développement.....	48
3.7.1 Python.....	49
3.7.2 Docker.....	49
3.7.3 Eclipse Mosquitto.....	50
3.7.4 Fiware.....	50
3.7.5 IoT Agent.....	50
3.7.6 Orion Context Broker.....	51
3.7.7 MongoDB.....	51
3.7.8 React.....	51
3.7.9 Tailwind CSS.....	52
3.7.10 MQTT Explorer.....	52
3.8 Architecture du système.....	52
3.9 Fonctionnement de la simulation.....	54
3.9.1 Logique générale de la simulation.....	54

3.9.2 Déroulement d'un cycle de simulation.....	55
3.9.3 Système de réservation utilisateur.....	60
3.10 Diagramme de séquence	63
3.11 Intégration au simulateur.....	64
3.12 Courbe de prédiction générée par le modèle	65
3.13 Résultats obtenus et analyse du système.....	66
3.13.1 Statistiques globales de la simulation	66
3.13.2 Validation de la logique d'attribution verticale des places	67
3.13.3 Analyse comparée : occupation prédite et observée	67
3.13.4 Comparaison entre approche logique et modèle prédictif dans l'attribution des places	69
3.13.5 Conclusion	70
Conclusion générale	71
Perspectives futures	72
Bibliographie	73

Introduction générale :

La gestion du stationnement constitue aujourd'hui un enjeu majeur dans les zones urbaines, en raison de l'augmentation constante du nombre de véhicules et de la saturation des espaces de stationnement. Les parkings multi-niveaux représentent une réponse structurelle à cette problématique, mais leur exploitation efficace nécessite des outils intelligents et automatisés.

L'émergence des technologies liées à l'Internet des Objets (IoT) et à l'intelligence artificielle (IA) offre de nouvelles opportunités pour moderniser la gestion des parkings. Grâce à ces technologies, il devient possible de suivre l'état des places en temps réel, d'optimiser la circulation interne, de faciliter la réservation à distance et de proposer un système global plus fluide, plus économique et plus écologique.

Ce mémoire s'inscrit dans cette dynamique et propose une solution intelligente de gestion de parking multi-niveaux, basée sur une simulation réaliste et une architecture modulaire intégrant des composants IoT, une interface web et des algorithmes de prise de décision.

Ce travail est structuré en trois chapitres :

Le chapitre 1 retrace l'évolution des systèmes de stationnement, depuis les parkings traditionnels jusqu'aux parkings intelligents actuels. Il aborde les problématiques des anciens systèmes, les apports des technologies modernes (IoT, IA), et décrit les éléments clés du fonctionnement des parkings intelligents, notamment dans un contexte multi-niveau.

Le chapitre 2 est consacré à l'état de l'art des solutions existantes. Il présente plusieurs approches mises en œuvre dans différentes villes ou dans des travaux de recherche, qu'elles soient réelles ou simulées, et propose une analyse comparative de leurs objectifs, outils technologiques et performances.

Le chapitre 3 décrit la solution développée dans le cadre de ce projet. Il détaille l'architecture technique mise en place, les outils utilisés (notamment FIWARE, MQTT, MongoDB, React), ainsi que le fonctionnement de la simulation. Il met également en avant les mécanismes d'optimisation et les résultats obtenus.

Enfin, ce mémoire se termine par une conclusion générale et quelques perspectives d'amélioration.

Chapitre 01 : Évolution et technologies des parkings intelligents

1.1 Introduction :

Avec l'augmentation du nombre de véhicules, la gestion du stationnement est un enjeu majeur. Les embouteillages et la perte de temps liés à la recherche d'une place aggravent la congestion urbaine et la pollution. Les avancées technologiques, notamment l'IoT et l'IA, offrent aujourd'hui des solutions innovantes pour optimiser l'occupation des parkings et fluidifier la circulation.

Ce chapitre explore les méthodes actuelles de gestion du stationnement et les améliorations apportées par ces nouvelles technologies.

1.2 Historique et évolution des systèmes de stationnement

1.2.1 La gestion traditionnelle des premiers parkings [1] :

Le premier parcmètre de l'histoire a été mis en place au début des années 1920 et aux années 30 à Oklahoma City, dans l'État de l'Oklahoma. L'entreprise Magee-Hale Park-O-Meter a vu le jour et les habitants de la ville entière ont commencé à bénéficier du système de stationnement régulé. Avant l'avènement de cette nouvelle solution, les gens se garaient tant bien que mal et les commerces de proximité en pâtissaient lorsque des clients potentiels ne parvenaient pas à trouver une place disponible. Pendant des décennies, des années 1930 aux années 1990, peu de progrès ont été réalisés dans la gestion du stationnement malgré que beaucoup d'innovations technologiques aient vu le jour notamment dans le domaine de l'automobile et de la téléphonie mobile.

Ce n'est qu'à partir des années 2000 que les accès aux parkings ont commencé à être ouverts grâce à un flash d'une carte RFID ou une image d'une caméra de reconnaissance des plaques d'immatriculation (LPR). En outre, cette même technologie pourrait prochainement permettre aux propriétaires de parkings de surveiller la durée du stationnement et aux automobilistes de retrouver leur voiture dans un dédale de places en béton.

Avec l'évolution de la technologie dans les rues et les garages, il était logique de passer à l'intégration de ces équipements avec des logiciels en ligne. Des applications de stationnement comme Parker qui a vu le jour en 2006, ParkWhiz en 2008 et ParkMobile en 2011 offrent la possibilité aux automobilistes de réserver des emplacements sur rue, dans des parkings, à proximité des stades ou même des aéroports via leur smartphone. L'association de matériel de stationnement et de logiciel intelligent a contribué à diminuer la durée des trajets

pour les conducteurs, à améliorer l'efficacité du secteur du stationnement et à réaliser des économies tant pour les conducteurs que pour la gestion du stationnement.

1.2.2 Les problèmes des parkings traditionnels :

Le stationnement est à l'origine de 60% de la pollution urbaine. En 2020, un conducteur a passé en moyenne 20 minutes à parcourir la ville en quête d'une place de stationnement. Cette situation génère un stress important pour les conducteurs et leurs passagers, accumulant en moyenne jusqu'à 72 heures par an passées à chercher une place de stationnement. Pour tout le monde, cela entraîne une augmentation du stress, de l'anxiété et de la frustration, au détriment du temps consacré à la famille et aux loisirs, ce qui nuit considérablement à la qualité de vie des usagers. Par ailleurs, l'augmentation des durées de trajet entraîne une surconsommation d'énergie ce qui engendre une consommation excessive de carburant, augmentant ainsi les dépenses des automobilistes et affectant leur pouvoir d'achat.

Les réseaux routiers ne peuvent s'étendre à l'infini notamment dans les villes où l'espace est limité et où les centres historiques construits il y a parfois plusieurs centaines d'années, ne sont pas destinés à recevoir un afflux de véhicules. En raison de l'expansion démographique et économique de la région, Les besoins croissants en déplacements routiers, combinés à une densité de trafic de plus en plus élevée par rapport à l'espace disponible, ont largement dépassé les capacités des infrastructures dans certaines zones. En d'autres termes, la capacité des routes a atteint son maximum et le résultat direct est l'apparition et l'amplification des embouteillages aux périodes de forte affluence, mais pas uniquement.

Enfin, les impacts environnementaux liés au trafic routier figurent aujourd'hui parmi les préoccupations majeures des politiques urbaines et environnementales. En effet, selon un raisonnement basique plus un véhicule reste longtemps sur la route, plus il consomme de carburant et plus il émet de polluants, les congestions routières constituent la situation la plus préjudiciable en matière de détérioration de la qualité de l'air[2].

1.2.3 Le Progrès des parkings automatisés (barrières automatiques, bornes de paiement):

1.2.3.1 barrières automatiques:

Une barrière de stationnement automatique est essentiellement constituée d'un bras entraîné par un moteur, d'un système de contrôle, de capteurs et d'éléments sécuritaires. Le moteur est le composant clé, entraînant le bras qui gère l'accès. Habituellement conçu à partir de matériaux robustes, le bras pivotant autorise ou empêche l'accès, en réponse aux signaux du système de contrôle. Ces derniers peuvent être issus de tickets, de badges RFID ou de systèmes qui reconnaissent les plaques d'immatriculation. Les capteurs sont essentiels pour assurer le fonctionnement sûr des barrières. Ils identifient la présence de

véhicules à proximité, évitant ainsi toute fermeture accidentelle du bras. En outre, des dispositifs de sécurité tels que l'arrêt d'urgence garantissent la sécurité des utilisateurs en cas de problème[3] .



Figure 01 : barrière de parking [3]

1.2.3.2 Bornes de paiement:

Les bornes de paiement numériques ont été élaborés afin d'optimiser la procédure de paiement dans les points de vente ou les espaces d'accueil.

Selon les nécessités, ses dispositifs de paiement peuvent être mis en place à l'intérieur ou à l'extérieur, offrant ainsi aux clients une expérience variée.

Ces bornes de paiement tactiles seront pratiques dans toutes les circonstances[4].



Figure 02 : borne de paiement [4]

1.2.4 L'introduction au parking intelligent :

Les systèmes de stationnement intelligents (SPS) se basent sur diverses technologies et de recherches avancées. Ces dispositifs sont déployés dans divers contextes et possèdent une gamme de caractéristiques, qui répondent aux défis auxquels sont confrontés les automobilistes dans leur quête régulière de stationnement. SPS exploite une variété de technologies supplémentaires comme les capteurs et les algorithmes de routage. Les systèmes de stationnement intelligents et Le problème de tournées de véhicules sont étroitement liés, car la quête d'un emplacement de stationnement nécessite la définition d'un parcours qui doit être optimal en fonction du contexte dans lequel le système opère. Les mécanismes de calcul des coûts dans le parcours peuvent être compliqués et peuvent inclure différentes données susceptibles de varier dynamiquement[5].

1.2.4.1 L'intégration de l'IoT et de l'IA dans les parkings intelligents :

1.2.4.1.1Intégrations de l'IA :

L'intelligence artificielle (IA),Occupe une place importantedans ces systèmes. Tout comme la vision humaine éclaire les décisions, la vision par ordinateur alimentée par l'IA permet aux systèmes intelligents d'effectuer efficacement des tâches sur la base d'entrées.L'avènement de la reconnaissance optique de caractères (OCR) en 1974 a marqué une étape importante dans la vision par ordinateur et la reconnaissance des formes. Développée à l'origine pour l'interprétation automatique de textes imprimés dans des images, la technologie OCR (Optical Character Recognition) est aujourd'hui devenue un outil polyvalent, utilisé dans de nombreux domaines tels que la logistique, la finance, les télécommunications ou encore la gestion urbaine, en effet,au fil du temps les progrès des techniques de traitement d'image, des algorithmes d'apprentissage automatique et de la puissance de calcul ont considérablement amélioré la précision et l'efficacité des systèmes OCR. Parmi les applications concrètes des algorithmes OCR, on peut citer la **reconnaissance automatique des plaques d'immatriculation** (ANPR), qui repose sur des techniques de traitement d'image et de reconnaissance de formes afin de détecter, lire et analyser automatiquement les numéros de plaques à partir d'images fixes ou de flux vidéo capturés par des caméras[4].

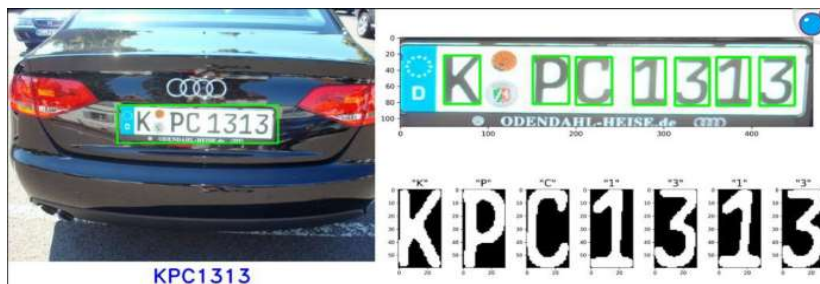


Figure 03 [5] : Détecter et reconnaître les plaques d'immatriculation des véhicules.

1.2.4.1.2Intégrations de l'IOT :

L'IOT a été introduite pour prédire l'espace de stationnement avec une analyse de régression. un système de stationnement intelligent basé sur l'IoT et le cloud qui collecte des données en temps réel, les envoie dans le cloud et recommande au conducteur une place de stationnement à proximité appropriée pour le véhicule. Le système utilise des capteurs **Raspberry Pi**, **NodeMCU**, d'identification par radiofréquence (en anglais, Radio-Frequency Identification : **RFID**) et infrarouges (**IR**). Un autre système de gestion de stationnement intelligent basé sur l'IoT utilisant la technologie LoRaWAN (Long Range Wide Area Network) pour surmonter les limites de communication et de coûts traditionnelles. Le système est composé de nœuds de détection intelligents avec des microcontrôleurs et des capteurs **Arduino UNO** pour détecter l'occupation des véhicules, en transmettant des données *via* LoRaWAN. Il propose des mises à jour de stationnement en temps réel via une interface graphique accessible et fonctionne indépendamment à l'énergie solaire[6].

1.2.5 Les parkings multi-niveaux :

Les parkings multi-niveaux représentent une solution efficace pour faire face au manque d'espace dans les zones urbaines denses. En superposant les niveaux de stationnement, ils permettent d'accueillir un grand nombre de véhicules sur une surface réduite. L'intégration de technologies intelligentes dans ces structures améliore encore leur efficacité. Grâce à des systèmes automatisés, les véhicules peuvent être garés sans intervention du conducteur, à l'aide d'ascenseurs mécaniques et de capteurs intelligents. Les données liées à l'occupation des places et aux utilisateurs sont stockées dans le cloud, et peuvent être consultées en temps réel via une application mobile. Ce type de gestion optimise l'espace, réduit la circulation inutile à l'intérieur du parking, et diminue ainsi la pollution. Cependant, ces systèmes présentent quelques limites : ils sont moins adaptés aux périodes de forte affluence et peuvent être difficiles à prendre en main pour les utilisateurs non familiers. Malgré cela, ils constituent une piste prometteuse pour une gestion de stationnement plus moderne, plus durable et mieux adaptée aux besoins actuels des villes[28].

1.3 Fonctionnement des parkings intelligents

1.3.1 Détection et gestion des places : capteurs IoT, caméras intelligentes

La technologie innovante de stationnement intelligent combinée à la connectivité IoT aide à résoudre le problème du stationnement. Les capteurs sont installés sur les parkings. Ceux-ci identifient les places de stationnement vides. Ces données IoT sont transmises via une connexion sans fil à un serveur cloud. Toutes les données du parking sont collectées et analysées en temps réel pour produire une carte des espaces disponibles

mis à la disposition de ceux qui recherchent une place libre. Les objectifs des gestionnaires de parkings privés ou les collectivités sont multiples :

- **Sécurité** : Identifier la présence d'un véhicule sur une voie réservée aux pompiers ou une sortie de secours, par exemple.
- **Accessibilité** : Simplifier l'accès aux places qui leur sont dédiées (individus en situation de handicap, employés d'une entreprise, conducteurs à une société de livraison, services communautaires...) en intégrant différentes technologies permettant de gérer à distance l'accès à ces emplacements (caméras de sécurité...).
- **Gestion du trafic** : améliorer la fluidité de la circulation en aidant les conducteurs à localiser rapidement une place de stationnement grâce à l'affichage du nombre de places libres en temps réel sur des panneaux situés à l'entrée des parkings.
- **Surveillance** : Accéder à des informations statistiques concernant le taux d'usage quotidien des parkings, les pics de fréquentation, les secteurs les plus visités, la durée moyenne d'un stationnement et ainsi de suite[7].

1.3.2 Paiement et réservation : Systèmes de paiement sans contact, réservations en ligne :

La gestion du stationnement a connu une évolution continue au fil des dernières décennies. Ce qui a débuté avec de simple distributeur automatique utilisant des pièces s'est maintenant métamorphosé en un système entièrement numérique et sans contact. Il est désormais possible de réserver, de payer et de surveiller les places de stationnement via des applications mobiles. Cette avancée a pour objectif d'optimiser le processus complet de stationnement, aussi bien pour les utilisateurs que pour les gestionnaires. De plus en plus, les systèmes automatiques classiques sont substitués par des alternatives modernes et interconnectées qui délivrent des informations en temps réel et assistent les chauffeurs dans la recherche accélérée de places de parking libres[8].

1.4 Technologies clés utilisées dans les parkings intelligents :

1.4.1 Capteurs IOT :

Généralement, un capteur convertit les phénomènes naturels en données pour des traitements et analyses ultérieurs. L'ensemble des phénomènes captés reflète la situation de l'environnement que nous tentons de suivre. Il existe plusieurs types de capteurs de stationnement intelligents :

- **Infrarouge** :Un capteur infrarouge est un dispositif électronique qui sert à détecter la longueur d'onde de la lumière de son environnement, soit par l'émission, soit par la détection du spectre infrarouge. Il est également en mesure de quantifier la chaleur dégagée par un objet et de détecter le mouvement.

Chaque capteur infrarouge est composé d'un émetteur et d'un récepteur. Chaque émetteur et récepteur possède deux registres, et un registre supplémentaire est utilisé pour surveiller la tension du LED qui indiquera l'état de la place de parking aux clients.L'IR retourne la valeur TRUE lorsqu'il ne détecte aucun objet devant lui, ce qui signifie une sortie de tension élevée (HIGH) du capteur, et FALSE lorsqu'un objet est détecté, ce qui correspond à une sortie de tension basse (LOW)[9].

- **Magnétomètre** :Un capteur de magnétométrie est employé pour repérer la présence d'un véhicule en évaluant la perturbation du champ magnétique terrestre provoquée par un objet métallique de grande taille tel qu'une automobile. Le système surveille en permanence le champ magnétique et consigne une modification lorsqu'un véhicule est détecté. Chaque magnétomètre est relié sans fil à une unité centrale de traitement qui compile les données et met à jour la disponibilité des places de stationnement en temps réel [10].

- **Ultrasonique** :Les capteurs ultrasoniques sont employés pour identifier les obstacles et les espaces vides entre les véhicules après avoir analysé le stationnement [11].

- **Caméra**: Pour obtenir des informations riches, certaines solutions déploient des caméras en tant que capteurs dans chaque place de parking. Cela ouvrira la voie à de nombreux nouveaux services intelligents. Pour maintenir un équilibre entre le coût et la qualité du service, il est suggéré d'utiliser un réseau de caméras déployées de manière autonome pour surveiller le parking, reliées par une maillage Wifi auto-installé. Une seule caméra peut surveiller une vaste zone plutôt que de se concentrer sur des places de parking spécifiques.

Grâce à l'algorithme intelligent, il est en mesure de réduire considérablement le nombre de capteurs et de caméras des systèmes existants, tout en atteignant une qualité de service supérieure[12].

- **Radar**: Une antenne radar à double lentille en ondes millimétriques (MMW) est conçue pour un système de stationnement intelligent dans le cadre de l'Internet des objets (IoT). On peut associer un radar à ondes millimétriques (MMW) à un capteur magnétique. Le radar MMW est capable de détecter la présence d'une voiture grâce au signal réfléchi. Un système de détection à double mode, comprenant un capteur magnétique et un radar MMW, peut augmenter la précision de la détection des places de stationnement tout en conservant une faible consommation d'énergie. Le radar MMW identifie la présence d'un véhicule en examinant le signal d'écho renvoyé par des objets métalliques tels que le châssis de la voiture[13].

Le tableau ci-dessous présente une comparaison des caractéristiques de différents types de capteurs utilisés dans la détection de véhicules, en tenant compte de plusieurs critères techniques et opérationnels[14] :

Tableau 01 : Comparaison entre les capteurs

Capteurs	Intrusif	Flexible	Impact Environnemental	petite Taille	Confidentialité	Installation	Contact	Précision	Coût	Détection Multiple
Mag nétomètre	✓		✓			**		*		
Ultrasons			✓		✓	*		*	*	
Caméra		✓			✓	***		**	**	✓✓
Infra rouge		✓	✓			*		*	*	
Rad ar		✓				***		**	**	✓✓

✓ : Présence d'une caractéristique * : Faible ** : Moyen *** : Élevé

1.4.2 Communication et transmission des données :

Pour transmettre les données des capteurs dans les parkings intelligents, beaucoup de technologies comme LoRaWAN pour les longues distances à faible consommation et MQTT pour une communication rapide et légère sont utilisées, souvent combinées avec la 5G et le cloud computing

- **LoRaWAN** : C'est un réseau à basse consommation qui se fonde sur la modulation des ondes radio dans la bande de fréquence 868 MHz. Les dispositifs connectés ont jusqu'à dix ans d'autonomie de batterie et ils ne peuvent transmettre que des données de petite taille (entre 0,3 et 50 kilobits par seconde de manière périodique. Grâce à des capteurs basse consommation, la technologie LoRa offre une connectivité optimale et un suivi en direct des emplacements de stationnement. Chaque emplacement est pourvu d'un capteur de

présence fonctionnant sur batterie, qui repère les déplacements des véhicules et transmet des actualisations à travers LoRa à une passerelle du réseau. Cette passerelle, reliée à un réseau LoRaWAN privé ou public, envoie les informations à l'application cloud PIN qui assure le suivi des emplacements libres et occupés[15].



Figure 04 [15] : fonctionnement de la technologie LoRa

- **MQTT:** MQTT est un protocole de communication standardisé et léger, destiné à la transmission de messages entre dispositifs IoT (capteurs, vêtements intelligents, etc.) sur des réseaux disposant d'une bande passante réduite. Il est simple à mettre en œuvre, performant et facilite la communication bidirectionnelle de données entre les dispositifs et le cloud. Parmi ses atouts, on compte :

- a) **Efficace et léger :** Optimise la bande passante tout en utilisant très peu de ressources (messages à partir de 2 octets).
- b) **Scalabilité :** Prend en charge des millions d'appareils avec une faible consommation énergétique et en code.
- c) **Fiable :** Propose trois niveaux de qualité de service (0, 1, 2) pour assurer la transmission, même en cas de réseaux instables.
- d) **Sécurisé :** Supporte le chiffrement et l'authentification (OAuth, TLS1.3, etc.).

MQTT fonctionne sur la base d'un modèle de publication et d'abonnement, dans lequel un intermédiaire (broker) coordonne les messages entre les expéditeurs (qui transmettent) et les destinataires (qui réceptionnent), sans interaction directe entre ces derniers (voir la figure 5). Ce modèle propose une séparation spatiale (absence

de nécessité de connaître la localisation), temporelle (absence de besoin de synchronisation temporelle), et de synchronisation (transmission/réception indépendantes). Le système de transmission MQTT est composé de :

- Client MQTT : Tout équipement qui utilise le protocole MQTT, qu'il soit émetteur ou récepteur.
- Agent MQTT : Gère la coordination des messages, assure l'authentification des clients et s'occupe du filtrage et de la transmission des données.
- Une liaison TCP/IP a été établie entre le client et l'agent à l'aide de messages[16].

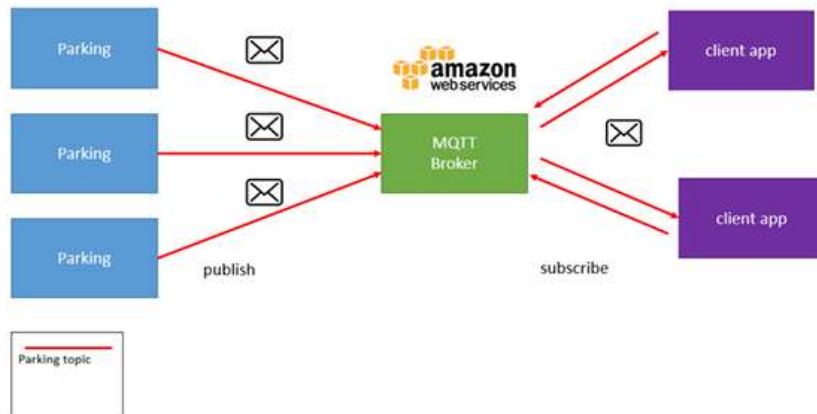


Figure 05 [17] : Architecture MQTT

- **ZigBee** : ZigBee est un appareil à basse consommation énergétique qui peut fonctionner pendant plusieurs années avec une seule pile. Cette technologie exploite la norme IEEE 802.15.4, offrant une portée en intérieur de 40 mètres et en extérieur de 120 mètres, ce qui est largement supérieur au Bluetooth et au WiFi. Cette technologie fait appel à 16 canaux et une adresse de 16 bits, permettant ainsi l'établissement de 65 536 nœuds pour la communication au sein d'un réseau. La mission première de ZigBee est de superviser et de gérer les dispositifs lors de la communication sans fil. Cette technologie offre une interface applicative, des services de sécurité et de soutien réseau au-dessus de la couche MAC (Contrôle d'Accès au Support) et la couche physique[18].

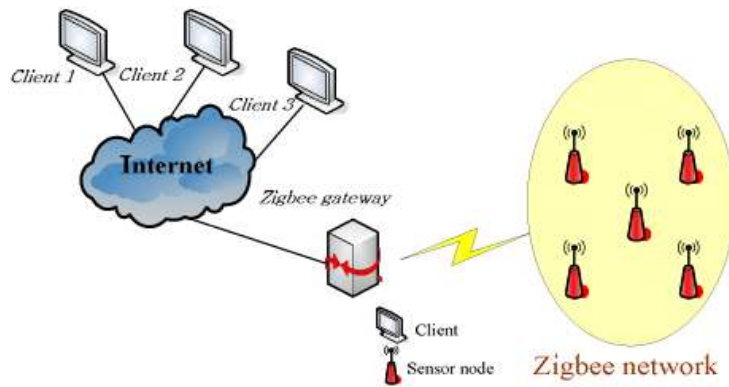


Figure 06 [18] : Architecture ZigBee.

- **WIFI** : La technologie de localisation intérieure basée sur le WiFi exploite la technique d'empreintes (fingerprinting) via des smartphones pour recueillir l'intensité des signaux. Cet algorithme, qui est le plus utilisé, comporte deux phases : la récolte hors ligne et l'établissement du positionnement en ligne. Au cours de la phase de collecte hors ligne : les signaux RSS (Received Signal Strength), recueillis par les smartphones, ainsi que les coordonnées des points de référence, seront consignés dans une base de données. On se réfère à ces données comme étant une carte radio. Durant la phase de localisation en temps réel (en ligne) : on utilise des appareils mobiles pour déterminer le signal RSS instantanément, qui est ensuite mis en relation avec la base de données afin d'identifier la position la plus précise. L'algorithme le plus couramment employé se compose de deux phases : la collecte hors ligne et le positionnement en ligne [19].

- **NB-IOT**(Narrowband Internet of Things):une technologie cellulaire novatrice, a été intégrée à la version 13 du 3GPP en vue d'une large couverture dans les contextes IoT. Cette technologie répond aux critères essentiels de l'IoT comme la souplesse de déploiement, la simplicité des dispositifs, une autonomie prolongée de la batterie, le support d'un important nombre d'appareils par cellule, ainsi qu'une amélioration notable de la portée au-delà des technologies cellulaires actuelles. Ces bénéfices font de NB-IoT une technologie idéale pour le stationnement intelligent, la lecture intelligente des compteurs, la surveillance intelligente, le suivi logistique, l'agriculture intelligente et diverses autres applications de l'Internet des objets[20].

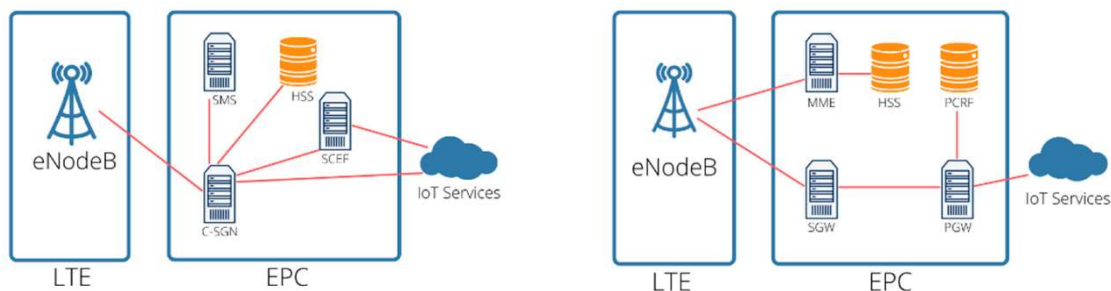


Figure 07 [21] : Architecture NB-IOT.

- **VANET:** Le VANET est une technologie qui favorise les communications entre véhicules. Les technologies VANET permettent aux véhicules de communiquer entre eux pour garantir la sécurité du conducteur. De plus, les véhicules peuvent transmettre des informations à l'Unité de Bordure de Route (UBR) concernant la synchronisation des feux de circulation et la gestion des files d'attente [22]. Néanmoins, cette technologie est exposée à plusieurs risques de sécurité, comme les failles logicielles, les algorithmes non protégés, les connexions sans fil instables, une faible capacité de cryptage, des défaillances matérielles et des soucis associés aux logiciels et systèmes d'exploitation[23] .

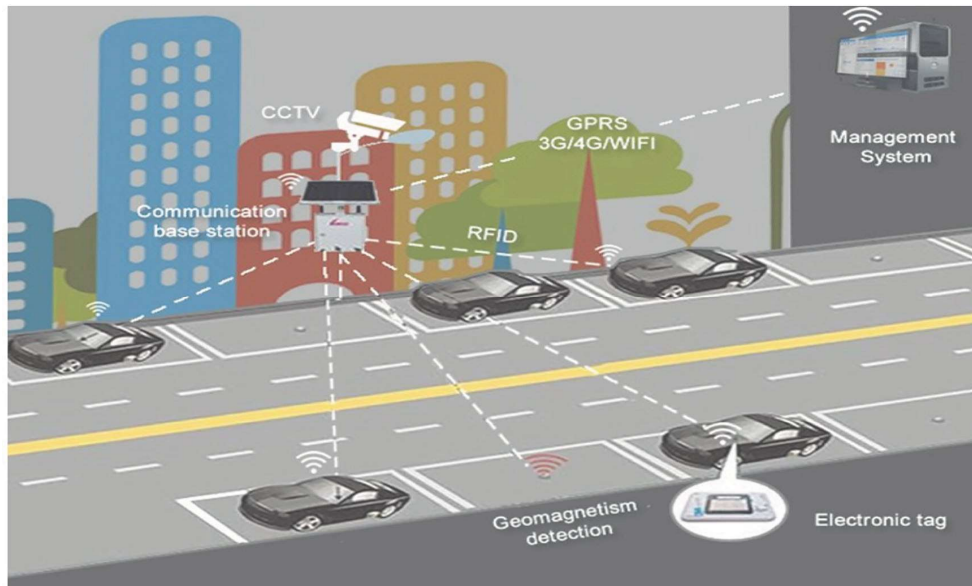


Figure 08 [22] : Système de stationnement intelligent dans un environnement VANET

Le tableau suivant présente une comparaison des technologies de communication précédemment abordées, en fonction des critères tels que la portée, la topologie de réseau et le spectre de fréquence[23] :

Tableau 02 : comparaison entre différentes technologies de communication

Protocole	Portée (Range)	Topologie de réseau (Network Topologie)	Spectre (Spectrum)
Lora WAN	Longue portée : 5 km (urbain), 20 km (rural)	Étoile (Star)	Libre (Free)
NB-IoT	Longue portée : 1 km (urbain), 10 km (rural)	Étoile (Star)	Licencié (Licensed)

Protocole	Portée (Range)	Topologie de réseau (Network Topologie)	Spectre (Spectrum)
ZigBee	Courte portée : 10-100 m	Maillage (Mesh)	Libre (Free)
WiFi	Courte portée : 20-50 m (intérieur), jusqu'à 100 m (extérieur)	Étoile (Star) ou Point à Point	Libre (2,4 GHz, 5 GHz)

1.4.3 Intelligence Artificielle:

L'intelligence artificielle et l'analyse de données massives jouent un rôle important dans la gestion du stationnement.

Les capteurs intelligents et les applications mobiles connectées, dont la popularité ne cesse de croître, pourraient en premier lieu proposer aux utilisateurs une expérience enrichie. Il est effectivement possible que la réservation anticipée d'une place, l'identification en temps réel des places disponibles et les paiements sans contact deviennent normaux, ce qui diminuerait par conséquent le stress lié à la recherche d'une place de parking, particulièrement dans les villes très fréquentées.

Par ailleurs, l'intelligence artificielle (IA) et le big data pourraient se voir attribuer un rôle crucial dans la transformation des parkings. Ces technologies peuvent effectivement permettre d'évaluer en temps réel le trafic routier et de prévoir les besoins en fonction des horaires et des événements particuliers. Par exemple, dans un parking, les dispositifs basés sur l'IA pourraient gérer automatiquement les places disponibles en fonction des arrivées et des départs des vols de l'aéroport. Cela contribuerait à éviter les congestions du côté des clients [24].

1.4.4 Services et technologies dans les applications de stationnement :

L'utilisation des applications de stationnement apporte un bénéfice considérable aux citoyens en leur permettant de gagner du temps et d'éviter des frustrations. Ces solutions visent principalement à identifier et à réserver des places de stationnement disponibles dans la zone. Ils sont aussi capables d'offrir des paiements électroniques, des systèmes de localisation par GPS et divers autres services. Les caractéristiques ci-dessous sont indispensables et représentent les composantes cruciales de chaque application dédiée au stationnement :

- **Suivi GPS :** Pour faciliter l'utilisation de l'application, il est indispensable d'activer la localisation par GPS. Ce dispositif facilite la localisation du véhicule et l'évaluation de la distance vers le stationnement le plus proche.
- **Recherche :** L'approche de recherche devrait être la plus simple possible. Par exemple, l'idée est que la solution permette de localiser le stationnement le plus proche en un seul clic suite au développement d'une application dédiée à la recherche de places de parking.
- **Réservation :** Cette fonctionnalité offre la possibilité de réserver un emplacement pour se garer. L'utilisateur a la possibilité de dénicher un lieu qui s'adapte à son budget et de le régler à l'avance sur une base quotidienne, hebdomadaire ou mensuelle. Cela évite de faire des réservations tous les jours.
- **Alignement des prix :** Les applications de stationnement ont généralement la capacité de juxtaposer les tarifs de divers lieux de stationnement. Ainsi, les utilisateurs peuvent rapidement identifier l'endroit le plus abordable à proximité[25].
- **Différents types de paiements :** Pour garantir le confort et la sécurité des utilisateurs, il est crucial de leur offrir diverses méthodes de paiement. Des plateformes comme PayPal devraient être intégrées [13].

Le tableau ci-dessous représente des exemples d'application de stationnement ainsi que leurs pays et les différents capteurs et technologies utilisées [26] :

Tableau 03 : d'exemple d'application de stationnement intelligent

Application de stationnement intelligent	Pays	Capteurs/technologiesutilisé
Park.ME	Autriche, Allemagne	Capteurs souterrains
SmartParking	Nouvelle-Zélande	Capteurs souterrains, RFID
ParkMe	Japon, États-Unis, Royaume-Uni, Allemagne, Brésil	Capteurs souterrains

Application de stationnement intelligent	Pays	Capteurs/technologiesutilisé
ParkAssist	États-Unis	Capteurs intelligents M4, reconnaissance de plaques (LPR)
SpotHero	États-Unis	Capteurs souterrains
EasyPark	Canada	Capteurs souterrains
PaybyPhone	France	Capteurs souterrains
ParkMobile	États-Unis	Capteurs souterrains
AppyParking	Royaume-Uni	Magnétomètre

1.5Impact et bénéfices des parkings intelligents :

Une problématique partagée par les centres-villes à l'échelle mondiale est que la recherche d'un emplacement pour garer son véhicule s'avère fréquemment être une opération longue et exaspérante pour les automobilistes. Cela non seulement contribue aux congestions routières, mais amplifie aussi les indices de pollution et de tension nerveuse. Les systèmes de stationnement intelligents exploitent la technologie pour surmonter ces obstacles et offrir aux automobilistes une expérience de stationnement fluide. Ces dispositifs comprennent une diversité de technologies et de solutions qui améliorent l'efficacité, la commodité et la durabilité du stationnement.

Les systèmes de stationnement intelligents intègrent souvent des stratégies de tarification dynamique. Durant les périodes de forte affluence, quand l'exigence de places de stationnement est élevée, des modifications tarifaires peuvent être effectuées dans le but d'encourager la rotation et de garantir l'accès. En revanche, les périodes creuses peuvent bénéficier de prix abaissés pour attirer davantage de clients. Ce système de tarification favorise non seulement l'optimisation des revenus pour les gestionnaires de stationnement, mais incite aussi à une exploitation plus efficiente des emplacements de parking.

Ces systèmes présentent de multiples bénéfices pour les gestionnaires de stationnement. En maximisant l'exploitation de l'espace et en appliquant une tarification dynamique, les opérateurs ont la possibilité d'accroître

leurs recettes. Par ailleurs, les fonctionnalités de suivi en direct et de gestion à distance diminuent la nécessité d'avoir du personnel sur place, entraînant par conséquent une réduction supplémentaire des coûts d'exploitation.

L'impact des systèmes de stationnement intelligents s'étend au-delà des conducteurs individuels et des opérateurs de stationnement. En réduisant la congestion du trafic, ces systèmes contribuent à l'amélioration globale de la mobilité urbaine. Avec moins de voitures à la recherche de places de stationnement, les routes deviennent moins encombrées, ce qui permet de réduire les temps de trajet, de réduire la consommation de carburant et d'améliorer la qualité de l'air[27] .

1.6 Conclusion :

Ce chapitre a retracé l'évolution des parkings, depuis les premières formes de gestion traditionnelle jusqu'aux systèmes intelligents intégrant les technologies de l'Internet des Objets et de l'intelligence artificielle. Il a présenté les technologies clés mobilisées (capteurs, communication, interfaces, algorithmes) ainsi que les bénéfices attendus en matière de fluidité du trafic, d'efficacité énergétique et d'optimisation des ressources. Le chapitre suivant présentera un état de l'art des solutions existantes, en mettant en évidence les approches concrètes déployées, les architectures adoptées et les principales limitations rencontrées dans le domaine des parkings intelligents multi-niveaux.

Chapitre 02 : état de l'art

2.1 Introduction :

Devant le problème de plus en plus aigu du parking en ville, diverses solutions ont été proposées pour mieux gérer le stationnement, en particulier dans les environnements urbains très denses. Grâce aux nouvelles possibilités qu'offre la technologie, les concepteurs ont créé des systèmes intégrés capables de capter, d'interpréter et d'optimiser la gestion de places de stationnement. Aujourd'hui, diverses idées de parking du futur font partie de cette réalité, comme les capteurs IoT qui capturent l'information, ou les plateformes intelligentes qui guident l'utilisateur jusqu'à la place de parking vide et adaptée. Dans ce chapitre nous allons présenter des solutions intelligentes existantes de stationnement, en particulier les parkings multiniveaux. Vous pourrez découvrir ici quelle est la particularité de ce type de parking, quels sont les avantages et les inconvénients, surtout en ce qui concerne les spécificités techniques des parkings à plusieurs niveaux.

2.2 Applications concrètes du smart parkings multiniveaux dans différentes villes :

L'idée des smart parkings a connu un développement progressif, notamment sur le plan infrastructurel, devenant de plus en plus complexe jusqu'à l'émergence des parkings multi-niveaux. Ces derniers sont aujourd'hui largement répandus, en particulier dans les aéroports, les centres commerciaux ou encore dans les zones urbaines à forte densité. Ce type de stationnement nécessite des systèmes de gestion sophistiqués, capables de fonctionner dans un environnement fermé, souvent réparti sur plusieurs étages. Afin de répondre à ces enjeux, diverses solutions techniques ont été mises en œuvre à travers le monde. Ci-dessous, nous présentons quelques exemples de villes qui ont déployé des systèmes de smart parkings multiniveaux :

1. Bhubaneswar, Inde [29] : Dans le cadre de son programme de développement en tant que ville intelligente, Bhubaneswar a mis en œuvre un système de stationnement intelligent basé sur une infrastructure de parking multiniveaux (en anglais, MLCP : *Multi-Levels Car Parking*) dans la zone commerciale de Saheed Nagar (voir la figure 9). Ce projet vise à répondre à la demande croissante de stationnement dans un secteur fortement fréquenté par des centres commerciaux, bureaux administratifs et établissements publics.

La structure est conçue pour accueillir entre **255 et 256** véhicules, avec une organisation sur plusieurs niveaux : un sous-sol inférieur, un sous-sol supérieur, un rez-de-chaussée, et plusieurs étages supérieurs. Tandis que les niveaux inférieurs sont destinés au stationnement, les étages supérieurs sont réservés à des espaces commerciaux, optimisant ainsi l'utilisation du terrain disponible. Le système est semi-automatisé et comprend plusieurs fonctionnalités modernes : contrôle d'accès, rampes d'entrée et de sortie, ascenseurs pour véhicules,

caméras de vidéosurveillance, signalisation directionnelle, éclairage optimisé, et zones piétonnes sécurisées. Ce type d'infrastructure contribue à désengorger la voirie en réduisant le stationnement en surface, tout en assurant confort et sécurité pour les usagers.



Figure 09 :Parking multiniveaux de Sahid Nagar, Bhubaneswar [30]

2. Lusail, Qatar [31] : À Lusail, les parkings multiniveaux des centres commerciaux comme Place Vendôme et Lagoon Mall utilisent des capteurs, des caméras PTZ et des panneaux numériques pour afficher en temps réel les places disponibles à chaque étage, réduisant ainsi le temps de recherche des conducteurs. Un système de reconnaissance des plaques d'immatriculation automatise les entrées et sorties, tandis que des applications mobiles permettent de réserver des places, de payer à l'avance et de localiser son véhicule, offrant une expérience fluide dans ces structures complexes situées dans des zones à forte affluence touristique et commerciale.



Figure 10 :Intérieur du parking multiniveaux de Lusail, Qatar [31]

En complément des exemples précédemment présentés, d'autres villes telles qu'Amsterdam, Barcelone ou Vienne ont également mis en place des projets intégrant la gestion intelligente du stationnement, notamment dans des structures multiniveaux. Ces systèmes s'appuient sur des capteurs, des caméras, des panneaux dynamiques et des applications mobiles afin d'optimiser l'occupation des places, réduire la congestion et améliorer la fluidité dans les zones à forte densité. Bien que la configuration précise (souterrain, en surface ou à niveaux) varie selon les contextes urbains, ces initiatives reflètent l'évolution vers une gestion intelligente et centralisée du stationnement dans les grandes villes [31].

1.3 Études des solutions proposées pour les parkings multi-niveaux :

Parmi les études analysées, celle de Fung réalisée en 2017 [32] se distingue par son approche fondée sur la simulation d'un système de stationnement intelligent, appliquée à un environnement urbain réel. Dans ce travail, Fung a développé une solution de stationnement intelligent fondée sur la simulation, dans un cadre urbain réel : la ville de Dublin. L'objectif principal était d'atténuer la congestion routière et de réduire les émissions liées à la recherche de stationnement, en s'appuyant sur la technologie de communication inter-véhicules (V2V), via les réseaux Ad-hoc véhiculaires (VANET). Pour cela, l'auteur a utilisé une combinaison des outils **SUMO**, **OMNeT++** et **VEINS** afin de modéliser le trafic, la communication réseau et l'interaction entre les véhicules dans un environnement représentatif.

Les données utilisées proviennent de la ville de Dublin : trafic mesuré en avril 2016 et taux d'occupation des parkings en février 2017. La simulation a été réalisée avec différents niveaux de charge et de population de

véhicules. L'un des apports majeurs de l'étude est l'utilisation de trois approches de dissémination des données : un modèle exact, un modèle par clusters (groupes de parkings), et un modèle **“live”** basé sur la communication entre véhicules en temps réel. Ce dernier est le plus léger computationnellement, tout en maintenant de bonnes performances.

Sur le plan des résultats obtenus, la réduction des émissions de CO₂ est particulièrement notable. Par exemple, lors de la simulation avec un échantillon réduit à 90 %, les émissions moyennes par véhicule dans le scénario de base étaient de **703.98 g/km**, contre **639.88 g/km** avec le système VANET, soit une diminution d'environ **9 %**. De plus, le nombre de véhicules ayant dû être redirigés est passé de **76 à 64**, confirmant une meilleure efficacité de la recherche de stationnement.

Cette étude constitue un apport essentiel à la recherche sur les smart parkings basés sur la simulation, en particulier pour les environnements urbains denses et les structures de stationnement multi-niveaux.

Cependant, l'étude fait état de plusieurs limites. Les simulations sont lourdes en calculs, nécessitant plusieurs jours d'exécution, et certaines n'ont pas pu aboutir en raison de contraintes matérielles. Par ailleurs, le système repose sur l'hypothèse d'une coopération totale entre véhicules, ce qui peut différer du comportement réel des conducteurs. Enfin, bien que réaliste, la simulation n'est pas encore validée par une implémentation réelle sur le terrain.

Une autre approche complémentaire est proposée par Zhao et al en 2018 [33], qui combine cette fois une simulation multi-agents avec un algorithme d'optimisation, appliquée à un véritable parking à deux niveaux situés sur le campus de Tongji à Shanghai. Leur méthodologie diffère de celle de Fung, notamment par l'intégration de données réelles et d'un modèle comportemental détaillé des usagers.

Dans cette étude, Zhao et al proposent une approche avancée de gestion de parkings multi-niveaux basée sur une **simulation multi-agents couplée à une optimisation par recuit simulé**. L'étude s'intéresse à la conception de stratégies de gestion optimales dans de grands parkings à forte densité, en tenant compte de la diversité des usagers (durée de stationnement, préférences, etc.) ainsi que des contraintes spatiales et temporelles. L'objectif principal est d'optimiser des indicateurs tels que le **temps de recherche**, le **temps de trajet** et la **distance de marche** dans un parking à double étage de **l'université Tongji**.

La méthodologie repose sur une **structure bi-niveau** : le niveau supérieur est un modèle d'optimisation (sous MATLAB) visant à maximiser le bénéfice social global du système, tandis que le niveau inférieur est un **modèle de micro-simulation multi-agents** développé dans **NetLogo**, qui reproduit le comportement des conducteurs dans le parking (prise de décision, recherche de place, mouvement, etc.). Pour calibrer les comportements simulés, les auteurs ont réalisé une enquête de préférence déclarée auprès de **367 usagers** dans

différents lieux publics (universités, aéroports, centres commerciaux), en modélisant les préférences à l'aide d'un **modèle logit binaire**. En parallèle, les données observées sur le terrain (entrées/sorties, durées de stationnement) issues du système intelligent de gestion du parking de Tongji ont été utilisées pour alimenter la simulation.

Les résultats obtenus sont significatifs : après optimisation, le temps moyen de déplacement dans le parking est réduit de **13,5 %**, le temps de recherche de stationnement diminue de **45,9 %**, et la distance moyenne de marche est réduite de **13,5 %**. L'efficacité du modèle est validée par un test statistique (t-test) comparant les sorties de la simulation avec les données réelles, montrant une très bonne concordance entre les deux distributions. L'étude démontre ainsi que la mise en place de zones à durée de stationnement limitée, stratégiquement réparties, permet d'adapter efficacement l'offre de stationnement à une demande variable selon les profils d'utilisateurs.

Enfin, les auteurs soulignent que ce type d'approche fournit un outil puissant pour évaluer différentes politiques de stationnement et concevoir des stratégies de gestion sur mesure, en tenant compte à la fois du comportement des usagers et des contraintes d'exploitation. L'utilisation conjointe de données réelles, d'un modèle comportemental, et d'une simulation multi-agents rend cette méthode particulièrement adaptée à l'optimisation des parkings complexes, notamment dans des environnements multi-niveaux.



Figure 11[33]:Interface de simulation du stationnement.

Une autre étude, proposée par Kharbeche et al. (2019)[34], s'intéresse à une simulation centrée sur le comportement des conducteurs et la gestion des réservations, à travers un système qui modélise l'occupation

des parkings de manière dynamique. Les auteurs développent un simulateur d'occupation pour un système de stationnement intelligent, dans le but d'évaluer diverses politiques de réservation et de gestion des places de stationnement. Le simulateur repose sur une modélisation multi-agents, dans laquelle les conducteurs sont représentés comme des agents autonomes suivant des plans d'activité journaliers, avec plusieurs points d'arrêt nécessitant une place de stationnement.

Le cas d'étude utilisé est un campus universitaire, incluant plusieurs parkings de capacités différentes. La base de données utilisée provient de configurations réalistes de stationnement (capacités, horaires, distances), et permet de simuler des conflits de réservation, des retards, ou encore des absences d'utilisation. Le système prend également en compte des mécanismes de garantie de réservation afin d'éviter que des places restent bloquées inutilement.

Les résultats obtenus montrent que le taux de réussite des réservations peut atteindre jusqu'à 90 % lorsque des stratégies de gestion incluant des délais de sécurité sont appliquées. La simulation montre aussi que les conducteurs parcourent une distance plus courte pour se garer, grâce à une meilleure répartition des places disponibles., grâce à une affectation optimale des places en fonction de la localisation et de la disponibilité. De plus, la distance moyenne parcourue entre la place de stationnement et la destination finale est réduite, ce qui contribue à améliorer l'expérience utilisateur. L'introduction de marges de sécurité permet également de limiter les conflits de réservation, en maintenant un équilibre entre la fiabilité du système et l'utilisation efficace des ressources disponibles

L'illustration suivante présente un aperçu du simulateur utilisé dans l'étude, appliqué à un campus universitaire, avec visualisation des flux de véhicules et de l'occupation des parkings.

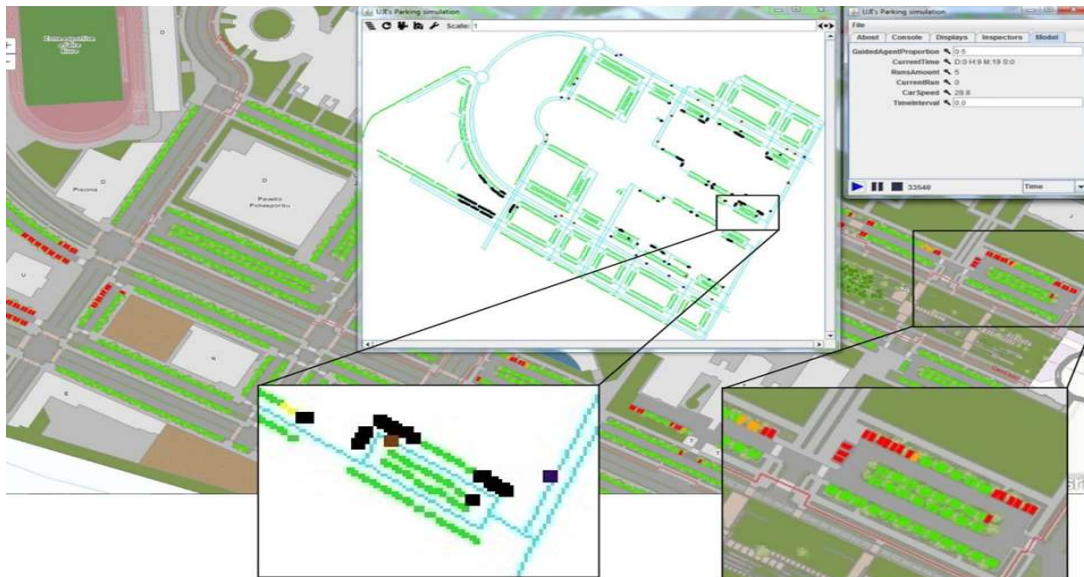


Figure 12 : Visualisation d'occupation du système de smart parking appliqué à un campus universitaire [35].

Ce simulateur constitue un outil d'aide à la décision pertinent pour les gestionnaires d'infrastructures, leur permettant de tester et d'optimiser des stratégies avant tout déploiement réel, notamment dans des contextes complexes comme les parkings multi-niveaux ou les zones à usage mixte. Le tableau ci-dessous présente une comparaison synthétique des trois études analysées, en mettant en évidence leurs approches méthodologiques, leurs objectifs, ainsi que les principaux résultats et limites identifiés.

Il convient de souligner que les trois études détaillées précédemment ne représentent qu'une partie des travaux existants sur la gestion intelligente des parkings multi-niveaux. De nombreuses autres recherches récentes proposent des approches variées et complémentaires. Par exemple, **Liu et al. (2022) [54]** ont conçu un système basé sur l'Internet des objets (IoT) pour la détection en temps réel de l'occupation des places à plusieurs niveaux, combinant capteurs physiques et communication sans fil [36]. **Ahmed et al. (2021) [55]** s'appuient quant à eux sur des techniques d'intelligence artificielle, en particulier les réseaux de neurones profonds, pour prédire la disponibilité des places dans des environnements urbains complexes [37]. **Kim et al. (2023) [56]** proposent enfin un système de navigation intelligent, capable d'orienter dynamiquement les conducteurs vers les emplacements libres en tenant compte du trafic et des niveaux de congestion [38]. Néanmoins, les études de **Fung (2017)**, **Zhao et al. (2018)** et **Kharbeche et al. (2019)** ont été privilégiées ici en raison de leur richesse méthodologique, mais aussi parce qu'elles offrent une meilleure lisibilité et une compréhension plus accessible dans le cadre de ce mémoire. Elles constituent ainsi des cas représentatifs pour illustrer les différentes dimensions de la simulation appliquée aux parkings intelligents multi-niveaux.

Tableau 04 : comparaison entre les trois études analysées

Auteur(s)	Année	Type de recherche	Méthodologie	Objectif principal	Résultats clés	Limites/ Observations
Fung, W.	2017	Simulation (VANET)	SUMO+ OMNeT++ + VEINS (scénario Dublin)	Réduire les émissions et la congestion grâce à la communication V2V	↓ émissions CO ₂ (-9 %) ; ↓ cas d'échec de stationnement ; amélioration de la fluidité	Simulation coûteuse ; suppose une coopération totale entre véhicules
Zhaoet	2018	Simulation	MATLAB +	Optimiser	↓	Modèle complexe ;

Auteur(s)	Année	Type de recherche	Méthodologie	Objectif principal	Résultats clés	Limites/ Observations
al.		appliquée à un cas réel + Optimisation	NetLogo (parking réel à deux niveaux)	l'attribution des places selon la durée de stationnement	recherche (-45,9 %) ; ↓ marche (-13,5 %) ; validation statistique par test t	nécessite données comportementales fiables
Kharbèche et al.	2019	Simulation multi-agents	Modèle d'agents + scénarios sur campus	Tester des politiques de réservation et de gestion dynamique	Taux de succès jusqu'à 90 % ; ↓ marche et trajets ; bonne stabilité du système	Sensible aux paramètres (ex. délais de réservation) ; nécessite ajustement selon l'usage

2.4 Contraintes spécifiques des parkings multi-niveaux et limites des solutions existantes :

2.4.1 Contraintes technologiques:

Les parkings multi-niveaux posent des défis importants pour la navigation autonome en raison de leur structure fermée, qui bloque le signal GPS. Même les systèmes de navigation avancés, combinant odométrie des roues et capteurs inertiels, échouent à localiser précisément un véhicule, car ils subissent des décalages, notamment sur la hauteur, obligeant à recourir à des capteurs laser coûteux pour compenser. De plus, la complexité tridimensionnelle de ces parkings, avec leurs étages multiples et leurs rampes, rend les cartes 2D inadéquates pour détecter les obstacles en 3D ou naviguer sur des surfaces inclinées, nécessitant des cartes multi-niveaux plus sophistiquées, ce qui accroît la complexité et les coûts[36].

2.4.2 Contraintes environnementales:

D'un point de vue environnemental, les parkings à plusieurs étages sont souvent confrontés à un manque criant de ventilation, ce qui provoque une concentration élevée de monoxyde de carbone (CO), bien au-delà des seuils acceptables. En effet, les niveaux dépassent régulièrement la norme professionnelle fixée à 50 ppm sur une période de 8 heures, atteignant parfois 170 % de cette limite dans les espaces intérieurs, mettant ainsi en danger la santé des employés qui y travaillent. Bien que plusieurs approches aient permis d'améliorer la gestion des parkings, il reste encore des difficultés, notamment pour aider les conducteurs à trouver rapidement une place libre et à circuler de manière fluide dans les parkings à plusieurs niveaux[37]. À partir de ce constat, une question centrale se dégage :

Comment améliorer la gestion d'un parking multi-niveaux à l'aide des technologies IoT, en réduisant le temps de recherche d'une place et en facilitant la circulation interne ?

2.6 Conclusion :

Ce chapitre a présenté différentes approches pour la gestion des parkings intelligents, avec un focus sur les structures multiniveaux. Les études analysées ont mis en lumière plusieurs avancées, mais aussi des limites, notamment en ce qui concerne la recherche rapide de places et la circulation interne.

Ces constats ont conduit à formuler une problématique centrée sur l'optimisation de la gestion des parkings multi-niveaux à l'aide des technologies IoT. Cette réflexion ouvre la voie au développement de la solution proposée, détaillée dans le chapitre suivant.

CHAPITRE 03 :Méthodologie

3.1 Introduction :

Ce chapitre décrit la démarche suivie pour concevoir et simuler une solution intelligente dédiée à la gestion d'un parking multi-niveaux, en s'appuyant sur les technologies liées à l'Internet des Objets (IoT). Dans une logique d'optimisation, notre démarche vise à améliorer principalement deux aspects : réduire le temps de recherche d'une place libre et limiter la congestion à l'entrée ou à l'intérieur du parking. Pour atteindre ces objectifs, nous présentons d'abord les outils et environnements technologiques utilisés dans le cadre de la simulation, notamment les langages, plateformes et protocoles. Ensuite, nous décrivons l'architecture globale du système, le fonctionnement des capteurs simulés, ainsi que le scénario de simulation mis en place pour évaluer les performances de notre solution.

3.2 solution proposée :

La solution développée dans le cadre de ce mémoire consiste en un système intelligent de gestion de parking multi-niveaux, reposant sur une architecture IoT modulaire et extensible. Elle vise à optimiser l'occupation des places, fluidifier la circulation interne et améliorer la gestion des réservations, grâce à l'intégration de capteurs simulés, d'un moteur de réservation, et d'un module d'intelligence artificielle.

Le système se compose de plusieurs briques fonctionnelles :

- **Simulation dynamique du parking**

Une simulation a été développée pour injecter virtuellement des véhicules, simuler des entrées/sorties et reproduire les flux de circulation dans différentes conditions.

- **Interface web de visualisation**

Elle permet d'afficher en temps réel l'état du parking, les niveaux, les places disponibles, occupées ou réservées, ainsi que les informations des véhicules présents.

- **Base de données centralisée**

Une base MongoDB enregistre les événements de simulation, les réservations, les entrées/sorties et les informations contextuelles utiles à l'analyse.

- **Fonctions d'optimisation**

Des fonctions logiques permettent d'attribuer les places en fonction de la durée de stationnement, notamment via l'algorithme `find_free_unreserved_spot()`.

- **Modèle prédictif XGBoost**

Un modèle d'apprentissage supervisé est intégré pour prédire le taux d'occupation futur. Il permet de refuser automatiquement certaines réservations si une surcharge est anticipée.

- **Modèle expérimental d'affectation intelligente**

Un second modèle XGBoost est testé pour attribuer directement une place idéale selon les données historiques, sans appliquer de règles déterministes.

Cette solution a été entièrement développée sous forme de simulateur afin de permettre une validation en environnement contrôlé, avec la possibilité de tester différents scénarios d'usage et de comparer les performances des différentes approches.

3.3 Préconditions et hypothèses de simulation :

Avant le déclenchement de la simulation, plusieurs hypothèses et conditions initiales ont été définies afin de garantir un déroulement cohérent et contrôlé du scénario simulé.

Hypothèses de base :

- Tous les capteurs (infrarouge, magnétomètre, caméra) sont supposés parfaitement fonctionnels et synchronisés, sans panne ni décalage. Les capteurs ultrasoniques sont simulés dans l'architecture, mais leur traitement visuel n'est pas développé, faute de capteurs physiques réels.
- Le serveur Orion Context Broker est toujours disponible et ne subit aucune interruption.
- Le comportement des véhicules suit une logique simulée contrôlée, incluant des délais réalistes de déplacement, d'entrée et de sortie, sans événements imprévus externes (ex. : panne, accident).
- Les réservations sont valides et cohérentes (pas de double réservation possible).
- Les niveaux de parking sont accessibles sans blocage ou panne mécanique.
- Un véhicule peut exceptionnellement échouer à entrer (avec une probabilité de 0,5 %) pour simuler une erreur de barrière.
- Le temps de trajet jusqu'à une place dépend du niveau visé, d'un délai de navigation aléatoire, et d'un facteur de congestion basé sur le taux d'occupation du parking.
- La sortie d'un véhicule peut inclure un dépassement léger et aléatoire de la durée prévue (jusqu'à 60 secondes), simulant un petit retard de départ.
- Les véhicules ayant effectué une réservation arrivent exactement à l'heure programmée, sans retard.

- Tout véhicule occupant temporairement une place réservée la libère toujours avant l'heure prévue de la réservation.
- Tout véhicule occupant temporairement une place réservée quitte cette place toujours avant l'heure prévue de la réservation.

Paramètres fixés :

- La durée de stationnement est générée aléatoirement entre 20 et 80 minutes.
- Les probabilités d'arrivée sont définies en fonction de plages horaires.
- Le nombre de places est limité à 80 places et réparti sur 4 niveaux (niveaux 1,2,3 et 4).
- La répartition des véhicules dépend de la durée de stationnement : les longues durées sont orientées vers les niveaux supérieurs (3 et 4).
- La simulation couvre trois types de scénarios selon le profil du bâtiment simulé : Mall, Office et Résidence.

Ce que la simulation ne gère pas :

- Pas de panne matérielle (capteurs, réseau ou composants).
- Pas de conduite irrégulière (vitesse excessive, refus d'entrer, etc.).
- Pas d'erreur de saisie de la part de l'utilisateur.
- Pas de gestion de files d'attente ou de priorités entre véhicules.
- Pas de replanification dynamique des places ou des niveaux en cas de surcharge.

3.4 Diagramme d'états

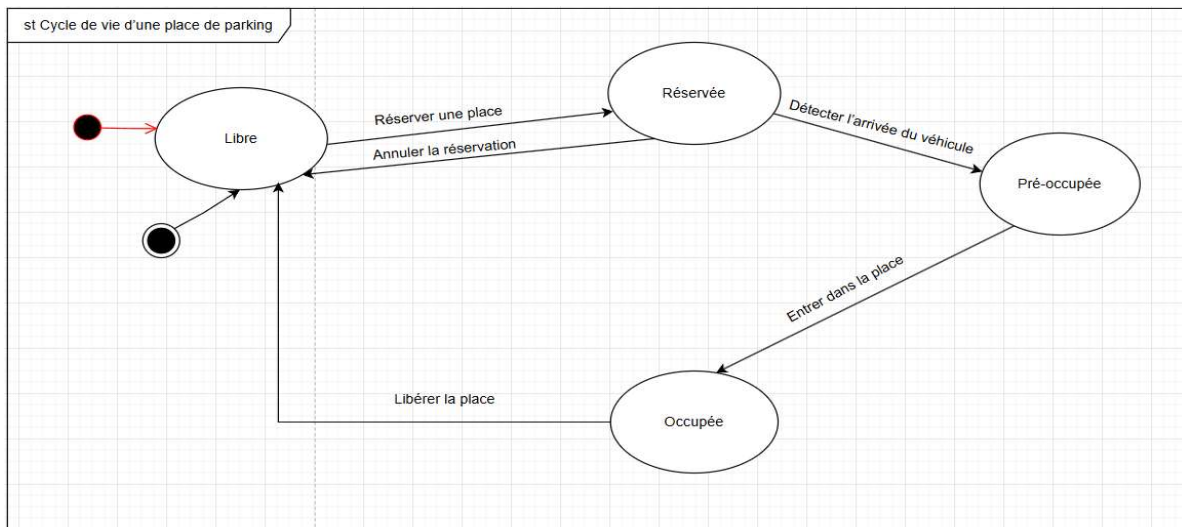


Figure 13 : Diagramme d'états illustrant le cycle de vie d'une place de parking simulée.

Ce diagramme d'états illustre les différentes transitions possibles d'une place de stationnement au sein du système, depuis l'état initial « Libre » jusqu'à « Occupée », en passant par les phases intermédiaires telles que « Réservée » et « Pré-occupée ». Il modélise le comportement dynamique de la place en fonction des actions de l'utilisateur et du déroulement de la simulation

3.5 Mécanismes d'optimisation implémentés :

Afin d'améliorer la fluidité, la robustesse et l'intelligence de la gestion du stationnement simulé, plusieurs mécanismes d'optimisation ont été implémentés dans le code. Ces mécanismes ne se limitent pas à l'attribution des places, mais s'étendent à la gestion dynamique des réservations, à la prévention des conflits d'accès, à la fiabilité des données, et à l'adaptation du comportement du simulateur selon l'activité.

Certains de ces mécanismes reposent sur des fonctions dédiées, développées spécifiquement pour répondre à des besoins critiques comme la réduction de la congestion, l'automatisation des entrées réservées ou encore la détection d'anomalies. D'autres sont intégrés dans la logique centrale du simulateur, comme le choix de la place en fonction de la durée ou l'ajustement du délai de simulation.

La suite de cette section présente les principales fonctions d'optimisation retenues, accompagnées d'une explication de leur rôle et de leur impact dans le fonctionnement global du système.

- **Fonction : Recherche d'une place libre non réservée :**

Cette fonction optimise l'attribution des places de stationnement en tenant compte à la fois de la disponibilité actuelle, des réservations futures et de la durée prévue du stationnement. Elle évite les conflits d'affectation, respecte les priorités selon la durée (courte ou longue) et répartit les véhicules de façon stratégique dans les différents niveaux du parking. Cela améliore l'efficacité d'utilisation de l'espace et réduit les risques de sur-occupation ou d'erreurs.

- **Principe de fonctionnement :**

La fonction suit une séquence de filtrage en plusieurs étapes :

- Elle commence par récupérer la liste complète des places via `get_all_spots()`, puis filtre uniquement celles dont l'état est libre (`s == "free"`) et qui ne sont pas déjà sélectionnées par d'autres véhicules dans le cycle en cours (grâce à `already_taken`).

- Ensuite, elle exclut les places qui sont concernées par une réservation imminente (via `get_reservation_start_ts`). Si la place est réservée dans moins de "`stay_duration`" secondes, elle est éliminée du choix.

- En fonction de la durée du stationnement ($\text{stay_duration} \geq 1800$ secondes soit 30 minutes), elle détermine un ordre de priorité :

- Longue durée (≥ 30 min) → étages 3 & 4
- Courte durée (< 30 min) → étages 1 & 2

- Elle privilégie les places situées sur les étages adaptés. Si aucune n'est disponible, elle retourne une place libre quelconque parmi celles filtrées.

- **Intégration dans le système :**

La fonction **`find_free_unreserved_spot (...)`** est définie dans le module `parking_flow.py` sous le nom de, et appelée dans `simulate_traffic_poisson.py` chaque fois qu'un véhicule est injecté dans le parking (`simulate_car_entry`), ou lorsqu'une réservation arrive à échéance (`reservation_watcher`). Elle constitue donc un maillon critique de la logique d'attribution cohérente des places.

- **Algorithme: `find_free_unreserved_spot (...)`**

Entrées :

stay_duration : entier (durée prévue de stationnement en secondes)

already_taken : ensemble de chaînes (IDs déjà affectés)

Sortie :

Spot : chaîne ou None (place libre compatible)

Début

spots $\leftarrow$ get_all_spots()

now_ts $\leftarrow$ temps_actuel()

free_spots $\leftarrow \emptyset$

Pour chaque s dans spots faire

Si s.status = "free" et s.id $\notin$ already_taken alors

res_start $\leftarrow$ get_reservation_start_ts(s)

Si res_start = None ou (now_ts + stay_duration $\leq$ res_start) alors

Ajouter s à free_spots

Fin Si

Fin Si

Fin Pour

Si free_spots = $\emptyset$ alors

Retourner None

Fin Si

Si $\text{stay_duration} \geq 1800$ alors

$\text{preferred_levels} \leftarrow \{3, 4\}$

Sinon

$\text{preferred_levels} \leftarrow \{1, 2\}$

Fin Si

$\text{preferred} \leftarrow \{s \in \text{free_spots} \mid s.\text{level} \in \text{preferred_levels}\}$

Si $\text{preferred} \neq \emptyset$ alors

Retourner $\text{élément_aleatoire}(\text{preferred})$

Sinon

Retourner $\text{élément_aleatoire}(\text{free_spots})$

Fin Si

Fin

- **Fonction :Simulation de l'entrée d'un véhicule :**

Cette fonction simule l'arrivée d'un véhicule dans des conditions proches du réel, en optimisant le processus d'entrée. Elle sélectionne une place adaptée, gère la réservation temporaire, prend en compte la congestion du parking pour estimer le temps de trajet, et active les capteurs dans l'ordre attendu. En assurant une entrée fluide, sans conflit, et en maintenant l'état du système à jour, elle contribue directement à l'optimisation du flux de véhicules dans le parking.

- **Principe de fonctionnement :**

- La fonction commence par générer un véhicule aléatoire avec une durée de stationnement entre 20 et 80 minutes.

- Elle utilise ensuite `find_free_unreserved_spot()` pour trouver une place libre compatible (en tenant compte des réservations).

- Dès qu'une place est trouvée, elle est verrouillée temporairement (statut pre-occupied) pour éviter une double affectation.
- Un temps de déplacement est simulé, en tenant compte :
 - Du niveau de la place,
 - D'un temps de navigation aléatoire,
 - D'un facteur de congestion basé sur le taux de remplissage global.
- Une fois arrivé, le véhicule active successivement les capteurs de la place :
 - IR,
 - Magnétomètre,
 - Caméra.
- Des données ultrasoniques sont également publiées entre les places adjacentes.
- La place passe alors à l'état final occupied et l'événement est enregistré dans MongoDB.
- **Intégration dans le système :**

La fonction `simulate_car_entry(...)` est utilisée par la boucle principale du simulateur (`simulate_traffic_new.py`) pour injecter un véhicule dans le système. Elle est déclenchée soit de manière individuelle (`entry_worker()`), soit en groupe via un burst.

- **Algorithme : `simulate_car_entry (...)`**

Entrées :

`Active_cars` : liste de véhicules actuellement présents dans le parking

`Claimed_spots` : ensemble de places déjà réservées dans le même cycle (optionnel)

Sortie:

Tuple (`car_id`, `spot_id`, `level`, `spot_number`) ou None

Début

`Car` ← `générer_véhicule_aleatoire ()`

`car.model` ← `générer_modele_aleatoire()`

car.size ← générer_taille_aleatoire()

car.stay_duration ← valeur_aléatoire_comprise_entre(1200, 4800)

car_id ← générer_identifiant_unique()

Si erreur_barrière_probabiliste(0.5 %) alors

Afficher ("Erreur : barrière bloquée")

Retourner None

Fin Si

spot ← find_free_unreserved_spot(car.stay_duration, claimed_spots)

Si spot = None alors

Afficher ("Aucune place disponible")

Retourner None

Fin Si

spot_id ← spot.id

level ← spot.level

spot_number ← spot.number

Si claimed_spots ≠ None alors

Ajouter spot_id à claimed_spots

Fin Si

Pour chaque v dans active_cars faire

Si v.spot_id = spot_id alors

Retourner None

Fin Si

Fin Pour

```

set_spot_status (spot_id, "pre-occupied")

travel_time ← 15

travel_time ← travel_time + calculer_délai(parétagé(level)

travel_time ← travel_time + valeur_aléatoire_comprise_entre (5, 10)

congestion ← taux_occupation_parking ()

travel_time ← travel_time + (congestion × 2)

attendre(travel_time)

activer_capteur ("infrared", spot_id, "occupied")

activer_capteur ("magnetometer", spot_id, "occupied")

activer_capteur ("camera", spot_id, "vehicle_detected")

set_spot_status (spot_id, "occupied")

publier_ultrasons_adjacent(spot_id)

Retourner (car_id, spot_id, level, spot_number)

```

Fin

- **Fonction : Simulation de l'entrée d'un véhicule réservé**

Cette fonction permet de sélectionner une place de stationnement libre et adaptée à la durée du séjour. Elle garantit la cohérence et la fiabilité des réservations en empêchant la double occupation, en respectant les créneaux définis, et en intégrant la détection des capteurs dans le flux normal. Cela optimise la gestion anticipée des ressources du parking.

- **Principe de fonctionnement :**

- La fonction est appelée lorsqu'un véhicule ayant effectué une réservation arrive à l'heure prévue.
- Elle récupère la place réservée (spot_id) et vérifie qu'elle n'est pas déjà occupée par un véhicule actif.
- Un identifiant unique est généré pour le véhicule (car_id), et un délai de quelques secondes est simulé pour représenter l'attente à la barrière.

➤ La place est immédiatement verrouillée temporairement (statut pre-occupied) afin de garantir qu'aucun autre véhicule ne puisse y accéder.

➤ Un temps de déplacement est simulé vers la place, calculé en fonction :

- D'un temps de base fixe,
- D'un délai aléatoire imitant la navigation interne,
- D'un ralentissement lié au taux d'occupation global du parking.

• Une fois le véhicule « arrivé » virtuellement à sa place, les capteurs de détection sont déclenchés successivement : capteur IR, magnétomètre et caméra.

➤ Des données de capteurs ultrasoniques sont également publiées vers les places adjacentes.

➤ Enfin, la place passe à l'état final "occupé, et les informations du véhicule sont enregistrées dans le système.

○ **Intégration dans le système :**

La fonction `simulate_reserved_entry(...)` est appelée par le module **reservation_watcher** lorsque l'heure d'une réservation est arrivée. Elle permet d'injecter le véhicule dans le parking de manière contrôlée, en respectant les contraintes de réservation, tout en simulant un comportement réaliste (temps de déplacement, activations capteurs). Elle s'intègre dans le cycle de simulation global aux côtés de `simulate_car_entry()`.

○ **Algorithme : `simulate_reserved_entry(...)`**

Entrées :

`spot_id` : chaîne (identifiant de la place réservée)

`Car_model` : chaîne (modèle du véhicule)

`Stay_duration` : entier (durée prévue de stationnement en secondes)

`Active_cars` : liste (véhicules actifs dans le parking)

Sortie :

Tuple (spot_id, données_véhicule) ou None

Début

Car_id $\leftarrow$ générer_identifiant_unique()

entry_time $\leftarrow$ temps_actuel()

Pour chaque véhicule v dans active_cars faire

Si v.spot_id = spot_id alors

Retourner None

Fin Si

Fin Pour

afficher("Véhicule réservé : ", car_model, " ID : ", car_id)

attendre(valeur_aléatoire_comprise_entre(2, 4))

set_spot_status(spot_id, "pre-occupied")

Base_time $\leftarrow$ 15

navigation_delay $\leftarrow$ valeur_aléatoire_comprise_entre(5, 10)

level $\leftarrow$ get_level_from_spot_id(spot_id)

level_delay $\leftarrow$ niveau_to_délai(level)

congestion $\leftarrow$ taux_occupation_parking()

congestion_delay $\leftarrow$ congestion $\times$ 2

travel_time $\leftarrow$ base_time + navigation_delay + level_delay + congestion_delay

attendre(travel_time)

```

activer_capteur("infrared", spot_id, "occupied")

activer_capteur("magnetometer", spot_id, "occupied")

activer_capteur("camera", spot_id, "vehicle_detected")

set_spot_status(spot_id, "occupied")

publier_ultrasons_adjacent(spot_id)

arrival_time ← temps_actuel()

car_size ← générer_taille_vehicule_depuis_modele(car_model)

données_vehicule ← {

    "car_id" : car_id,

"model": car_model,

    "size": car_size,

    "travel_time": travel_time,

    "stay_duration": stay_duration,

    "entry_time": entry_time,

    "arrival_time": arrival_time

}

Retourner (spot_id, données_vehicule)

Fin

```

- **Utilisation de threads pour améliorer la fluidité du système :**

Deux fonctions techniques sont exécutées en parallèle sous forme de threads afin de fluidifier l'exécution de la simulation :

- La fonction **entry_worker()** permet de lancer une entrée de véhicule dans un thread séparé, ce qui autorise plusieurs arrivées simultanées sans bloquer la boucle principale.
- La fonction **reservation_watcher()** tourne en arrière-plan et surveille en continu les réservations prévues. Lorsqu'un créneau arrive à échéance, elle déclenche automatiquement l'entrée du véhicule réservé.

Les fonctions présentées dans cette section constituent le cœur des mécanismes d'optimisation intégrés au simulateur. Elles permettent non seulement une gestion intelligente et contextuelle des ressources (places, durées, réservations), mais contribuent également à renforcer la fluidité du trafic, la cohérence des données et la prévention des conflits. Certaines de ces fonctions, telles que la sélection dynamique des places ou le traitement automatisé des entrées réservées, incarnent directement des stratégies d'optimisation adaptative. D'autres, bien que techniques, facilitent l'exécution asynchrone et la stabilité de la simulation. Ainsi, l'ensemble de ces composants participe à la création d'un système de stationnement simulé à la fois réactif, robuste et aligné avec les contraintes d'un environnement réel.

3.6 Intégration de l'intelligence artificielle dans la prise de décision :

Dans le cadre de ce projet, l'intelligence artificielle a été mobilisée comme outil d'aide à la décision dans le système de gestion du stationnement. L'objectif est de rendre certaines fonctions plus intelligentes, plus réactives et mieux adaptées à des contextes variables.

À ce jour, un premier modèle a été intégré pour prédire l'occupation future du parking et ainsi filtrer les réservations à risque. Cette section présente la logique, les données et l'intégration du module de prédiction actuellement déployé dans le simulateur.

3.6.1 Objectif du modèle prédictif :

Le modèle développé permet d'anticiper le taux d'occupation global du parking à un instant donné dans le futur afin d'anticiper les surcharges. Dans le cas où un utilisateur souhaite réserver une place pour une heure donnée, le système questionne ce modèle afin d'évaluer le niveau d'occupation escompté. Si ce niveau dépasse (ou approche) un seuil jugé critique, à savoir 80 %, alors la réservation n'est pas validée, sinon elle est acceptée. Ce mécanisme permet de réduire les conflits d'accès, notamment aux heures de pointe, et par là même de gérer les ressources de stationnement avec plus de fluidité et de fiabilité.

3.6.2 Données d'entrée et préparation :

Le modèle s'appuie sur les enregistrements de données stockées dans MongoDB, par l'intermédiaire des logs des événements d'entrée et de sortie des véhicules, qui sont transformés en séries temporelles par tranche de 5 minutes (nombre de places occupées à chaque instant).

Les données sont ensuite enrichies par des variables temporelles et comportementales, notamment :

- Heure et jour de la semaine (sin/cos encodés)
- Indicateurs de pics d'affluence (soirée, nuit, week-end...)
- Pondérations horaires (plus élevées en période critique) historique court (lags 1 à 6 → occupation en 30 minutes précédentes)

L'ensemble est structuré pour permettre au modèle de capturer les régularités temporelles et les comportements cycliques d'occupation.

3.6.3 Choix du modèle : XGBoost [37]

Le modèle retenu est XGBoost (eXtreme Gradient Boosting), est un algorithme basé sur un ensemble d'arbres de décision. Ce choix s'appuie sur plusieurs critères :

- Performant sur les données structurées et tabulaires
- Capacité à gérer les interactions non linéaires entre les variables
- Faible besoin de prétraitement (pas de normalisation poussée requise)
- Temps d'entraînement court et prédiction rapide
- Interprétable via l'importance des variables

Le modèle est utilisé ici en **régression** pour prédire une valeur continue : le nombre de places occupées à un instant donné.

3.6.4 Bibliothèques utilisées :

Le traitement dans son ensemble est implémenté en Python à l'aide de bibliothèques dédiées, dont chacune est spécifiquement dédiée à un aspect de la préparation des données, de l'apprentissage et de l'intégration du modèle prédictif :

- **Xgboost[53]** : principale bibliothèque utilisée pour l'entraînement du modèle. Cette bibliothèque permet d'implémenter l'algorithme XGBoost (eXtreme Gradient Boosting), basé sur des arbres de décision optimisés et itératifs, et permet d'entraîner des modèles de régression ou de classification performance dans le cadre pur des données tabulaires, avec des options de réglage précises.
- **Pandas [38]**: utilisée pour la manipulation et la structuration des données ; elle permet de créer et transformer des DataFrames, gérer les séries temporelles (ex : arrondi à des intervalles de 5 minutes), ajouter des variables dérivées (features) et agréger les données de manière assez facile.
- **Scikit-learn [39]** : utilisée pour la division du jeu de données (train/test split), le calcul des métriques d'évaluation, le cas échéant l'encodage des variables et éventuellement s'intégrer dans un pipeline, mais aussi compatible avec XGBoost via l'interface sklearn API, etc.
- **Matplotlib [40]** : utilisée pour la visualisation des résultats du modèle. Elle permet de tracer la courbe d'occupation réelle, les prédictions, et les comparer visuellement sur une échelle de temps. Ces graphiques aident à interpréter la précision et le comportement du modèle.
- **Numpy[41]** : bibliothèque de calcul numérique utilisée en complément pour effectuer des opérations mathématiques (vecteurs, sinus/cosinus pour l'encodage cyclique, normalisation, etc.).
- **Joblib [42]** : utilisée pour sauvegarder et recharger le modèle entraîné dans un format binaire. Cela permet de ne pas réentraîner le modèle à chaque exécution, mais de l'intégrer directement dans le simulateur.

3.7 Environnement de développement

La figure 13 montre l'ensemble de ces outils qui ont été sélectionnés pour garantir la modularité, la portabilité et la cohérence du système, tout en facilitant le développement, le test et la supervision de la solution simulée de gestion de parking multi-niveaux.



(a) logo Python [38].



(b) Logo Docker [39].

(c)

Logo Eclipse Mosquitto [40]



(d) logo MongoDB [44].



(e) logo Fiware [41].



(f) logo React [45].



(g) logo Mqtt explorer [47].

Figure 14 : Outils utilisés

3.7.1 Python

Python est un langage de programmation interprété, polyvalent et largement utilisé dans les domaines de la data science, du développement web, de l'intelligence artificielle et de l'Internet des objets (IoT). Sa syntaxe simple et lisible permet un développement rapide, ce qui en fait un outil de choix pour les projets de simulation et de prototypage rapide. Dans le cadre de ce projet, Python est utilisé pour concevoir les scripts de simulation, gérer la communication entre les modules et collecter les données à analyser [43].

3.7.2 Docker

Docker est une plateforme de virtualisation légère basée sur des conteneurs, qui permet de regrouper une application et toutes ses dépendances dans un environnement isolé, garantissant ainsi la portabilité et la cohérence entre les différents systèmes de déploiement (Merkel, 2014). Contrairement aux machines virtuelles traditionnelles, les conteneurs Docker partagent le noyau du système hôte, ce qui réduit considérablement leur

empreinte mémoire et améliore leur performance. Dans le cadre de ce projet, Docker est utilisé pour déployer plusieurs composants essentiels de la solution IoT, notamment le broker MQTT (Mosquitto), le gestionnaire de contexte (Orion Context Broker), l'interface utilisateur (WireCloud) et les agents IoT nécessaires à la communication avec les capteurs ainsi que la base de données MongoDB. Cette approche modulaire facilite la gestion, la configuration et l'orchestration des différents services nécessaires à la simulation [44].

3.7.3: Eclipse Mosquitto

Eclipse Mosquitto est un broker MQTT open source, léger et multiplateforme, conçu pour assurer la messagerie publish/subscribe entre objets connectés (Eclipse Foundation, s. d.). Conforme aux spécifications MQTT 3.1, 3.1.1 et 5.0, Mosquitto se distingue par une consommation mémoire très faible et une latence réduite, ce qui le rend particulièrement adapté aux environnements contraints de l'IoT, allant des micro-contrôleurs aux déploiements cloud. Il prend en charge la rétention de messages, la persistance sur disque, le pontage entre brokers et propose plusieurs niveaux de qualité de service (QoS 0, 1, 2) pour garantir la fiabilité des communications. Mosquitto dispose par ailleurs de mécanismes d'authentification et de chiffrement TLS, assurant la sécurité des échanges. Dans le présent projet, Mosquitto joue le rôle de cœur de la passerelle de communication : il relaie les données des capteurs simulés vers le reste de la plateforme (Orion Context Broker, base MongoDB, interface WireCloud), tout en permettant une scalabilité aisée grâce à sa compatibilité avec Docker et Docker Compose [45].

3.7.4 Fiware

FIWARE est une plateforme ouverte standardisée qui fournit un ensemble de composants réutilisables appelés Generic Enablers (GE), conçus pour le développement d'applications intelligentes dans divers domaines tels que les villes intelligentes, l'agriculture, l'énergie ou l'industrie 4.0 (FIWARE Foundation, 2025). Ces composants couvrent des fonctions essentielles comme la gestion du contexte (via l'API NGSI), l'intégration IoT, la cybersécurité, la gestion de données et l'interaction avec les utilisateurs. Dans le cadre de notre projet de gestion intelligente de parking multi-niveaux, FIWARE joue un rôle fondamental en fournissant une architecture modulaire et interopérable. Elle permet de connecter et d'orchestrer les différents services (capteurs, base de données, API, tableau de bord) au sein d'un même écosystème cohérent. Grâce à cette infrastructure, notre solution bénéficie d'une meilleure évolutivité, d'une intégration simplifiée des composants IoT comme les agents MQTT et d'une capacité à structurer les données en entités intelligentes, facilitant leur exploitation en temps réel [46].

3.7.5 IoT Agent

L'IoT Agent est un micro-service central de la plate-forme FIWARE qui fait le lien entre les protocoles IoT de terrain (dans notre cas MQTT) et les interfaces NGSI exposées par le Context Broker. Selon la documentation officielle (*IoT Agent Node Lib*, FIWARE Foundation, 2024), il assure à la fois le provisionnement automatique des dispositifs, la traduction bidirectionnelle des messages (données d'observation et commandes) et la gestion dynamique des attributs. Cette passerelle normalise ainsi des flux IoT hétérogènes vers le format NGSI, garantissant leur cohérence et leur accessibilité au sein de l'écosystème FIWARE.

Dans notre projet, l'IoT Agent joue donc un rôle clé : il permet à toutes les données remontées par les capteurs simulés d'être immédiatement prises en charge par Orion Context Broker et les autres services, assurant une intégration transparente et scalable des objets connectés [47].

3.7.6 Orion Context Broker

Orion Context Broker est le composant central de gestion de contexte de la plateforme FIWARE, conforme aux spécifications NGSI-v2. Il permet de collecter, stocker, mettre à jour et diffuser en temps réel des informations contextuelles structurées sous forme d'entités et d'attributs (FIWARE Foundation, 2025). Grâce à une API RESTful, Orion offre la possibilité de créer, modifier, interroger et supprimer des entités tout en assurant un mécanisme d'abonnement/notification qui informe automatiquement les services consommateurs à chaque modification du contexte.

Dans notre projet, Orion sert à centraliser les données contextuelles du parking (état des places, niveaux, capteurs) et à les rendre accessibles aux autres composants, ce qui permet une gestion cohérente, une visualisation en temps réel et une communication fluide entre les services [48].

3.7.7 MongoDB

MongoDB est une base de données NoSQL orientée documents, conçue pour stocker des données sous forme de documents JSON enrichis, appelés **BSON**. Elle se distingue par sa flexibilité de schéma, sa capacité à gérer de gros volumes de données non structurées, et ses performances élevées grâce à l'indexation et à la réplication horizontale (MongoDB Inc., 2025). Dans le contexte de FIWARE, MongoDB est utilisée par Orion Context Broker pour assurer la persistance des entités contextuelles.

Dans notre projet, MongoDB joue un rôle essentiel en conservant l'état de toutes les entités (places de parking, capteurs, réservations, etc.) de manière durable. Grâce à elle, les données publiées par les capteurs et gérées par Orion ne sont pas perdues lors d'un redémarrage du système, ce qui permet de relancer la simulation sans perte d'information et d'exploiter les historiques pour l'analyse ou la visualisation [49].

3.7.8 React

React est une bibliothèque JavaScript développée par Meta, conçue pour construire des interfaces utilisateurs dynamiques à partir de composants réutilisables. Grâce à son système de rendu basé sur un DOM virtuel, elle permet des performances optimales et une mise à jour fluide de l’affichage. Dans le cadre de notre projet, React est utilisé pour concevoir l’interface du tableau de bord qui affiche l’état des places de stationnement, permet les réservations et interagit avec les microservices via des appels API. Cette approche permet une visualisation temps réel et une interaction fluide avec les données simulées. La modularité de React facilite en outre l’extension future de l’application [50].

3.7.9 Tailwind CSS

Tailwind CSS est un framework CSS utilitaire qui permet de styliser les interfaces utilisateur en appliquant directement des classes prédéfinies aux éléments HTML, sans avoir à écrire de feuilles de style personnalisées. Contrairement aux frameworks traditionnels fondés sur des composants prédéfinis, Tailwind adopte une approche dite “bas niveau” qui offre une grande flexibilité de conception tout en garantissant une structure claire.

Dans notre projet, **Tailwind CSS** a été utilisé pour concevoir une interface moderne, réactive et cohérente. Il a permis de construire rapidement les différents écrans de l’application frontend, notamment le tableau de bord principal, la page de connexion des utilisateurs et l’interface dédiée aux utilisateurs connectés. Grâce à Tailwind, la mise en page (grilles, boutons, cartes) a pu être réalisée de manière efficace, avec un code plus lisible et une personnalisation facilitée [51].

3.7.10 MQTT Explorer

MQTT Explorer est un outil de visualisation graphique open source conçue pour interagir avec des brokers MQTT. Il permet d’inspecter les topics publiés, de surveiller en temps réel les messages échangés, et de publier manuellement des messages de test. Grâce à son interface claire et arborescente, il facilite la compréhension de la structure des topics et l’analyse du trafic IoT. Dans le cadre de notre projet, MQTT Explorer a été utilisé principalement à des fins de débogage et de validation des échanges entre les capteurs simulés, le broker Mosquitto et l’IoT Agent. Il a permis de vérifier que les données publiées par les capteurs sont bien structurées, correctement transmises et conformes aux attentes de la chaîne FIWARE [52].

3.8 Architecture du système :

L'architecture du système reflète la manière dont les différentes technologies ont été assemblées pour assurer une communication fluide entre les capteurs simulés, le traitement des données et leur visualisation. Elle s'appuie sur une structure en microservices interconnectés, déployés via Docker, et décrit le cheminement complet des données depuis leur émission jusqu'à leur affichage dans le tableau de bord. La figure 14 illustre l'architecture fonctionnelle du système proposé, en précisant les principaux composants ainsi que les protocoles de communication utilisés entre eux. Le fonctionnement du système repose sur une chaîne de traitement des données en temps réel, organisée autour des composants déployés dans l'environnement Docker. Les capteurs simulés publient des messages MQTT vers le broker Mosquitto, qui joue le rôle d'intermédiaire entre les capteurs et la plateforme FIWARE. Le fonctionnement du système repose sur une chaîne de traitement des données en temps réel, organisée autour des composants déployés dans l'environnement Docker. Les capteurs simulés publient des messages MQTT vers le broker Mosquitto, qui joue le rôle d'intermédiaire entre les capteurs et la plateforme FIWARE.

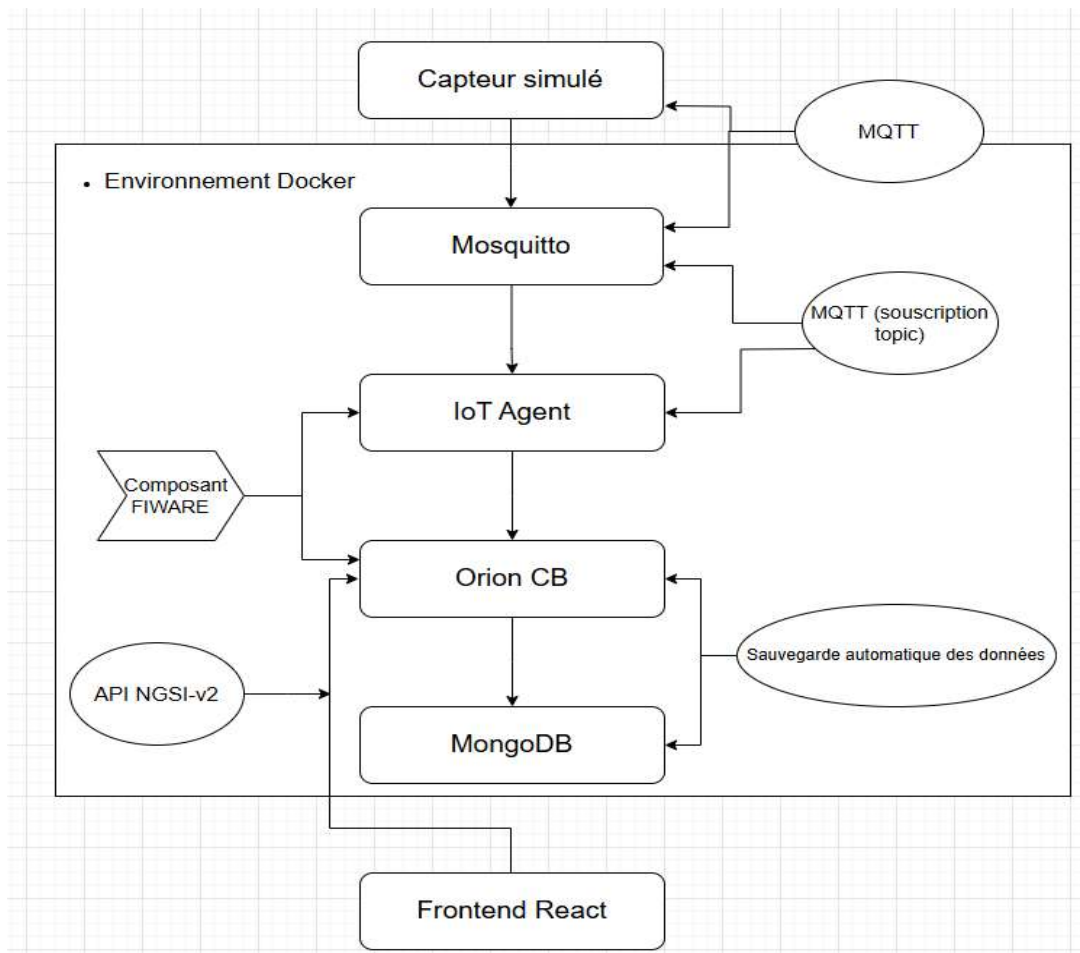


Figure 15 : Architecture fonctionnelle du système proposé.

Ces messages sont ensuite interceptés par l'IoT Agent, un composant FIWARE chargé de convertir les données en entités NGSI, compréhensibles par le gestionnaire de contexte Orion. Orion Context Broker centralise les données, les met à jour dynamiquement et les rend accessibles à l'ensemble des services connectés. Il communique directement avec MongoDB pour assurer la persistance automatique des entités. Le frontend développé en React interroge Orion via l'API NGSI-v2 pour afficher l'état du parking en temps réel. Grâce à cette architecture, chaque composant remplit un rôle spécifique tout en garantissant une communication fluide, cohérente et modulaire dans l'ensemble du système.

3.9 Fonctionnement de la simulation :

3.9.1 Logique générale de la simulation :

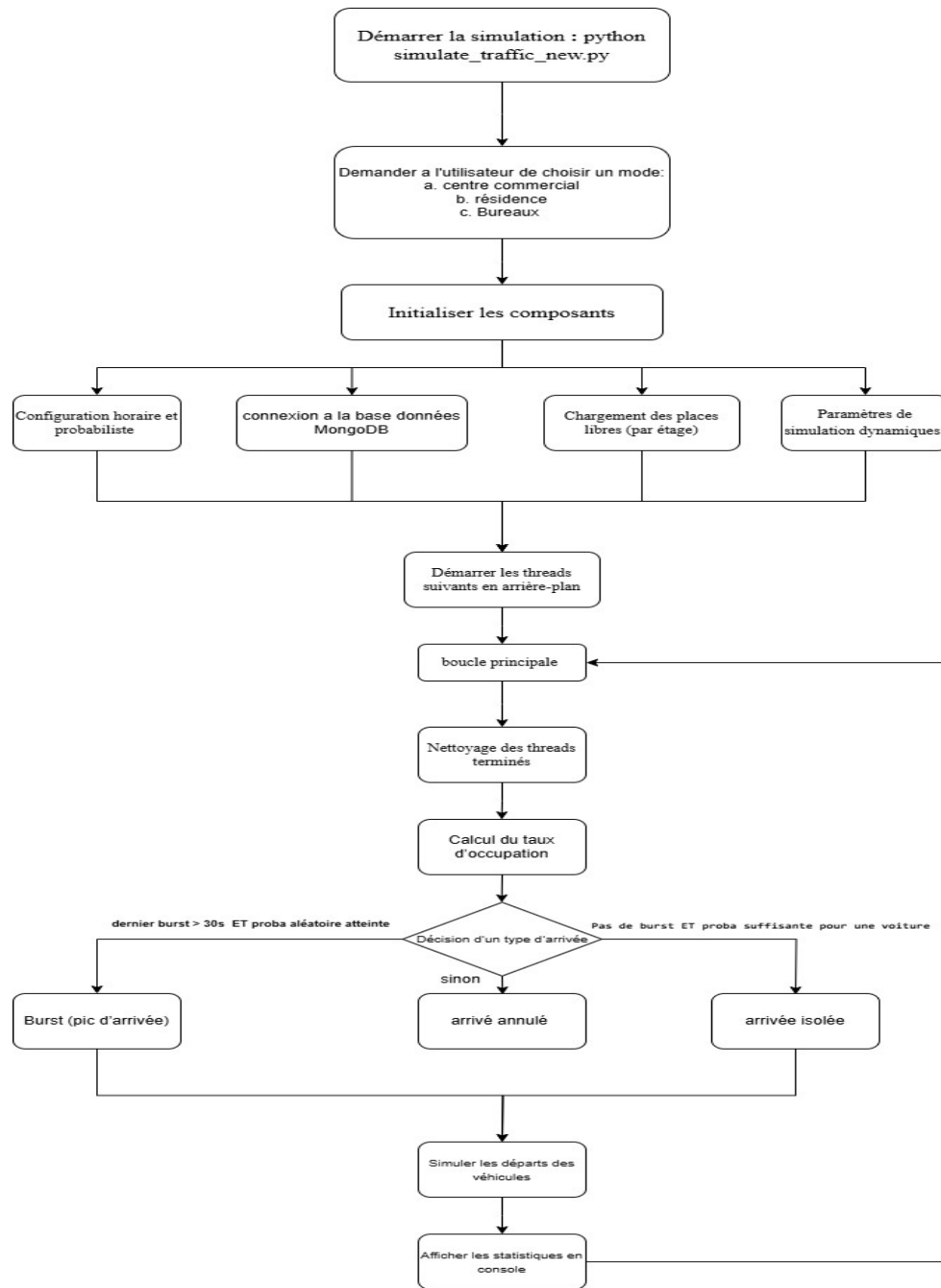


Figure 16 : Organigramme de la Simulation du Trafic Véhiculaire dans le Parking Intelligent

L'organigramme illustre le déroulement global de la simulation du trafic dans le parking intelligent. La simulation débute par l'exécution du script **simulate_traffic_new.py**, qui invite l'utilisateur à sélectionner un mode de fonctionnement parmi trois options : **centre commercial, résidence ou bureaux**. Une fois ce choix effectué, le système initialise les composants nécessaires, notamment la configuration horaire et probabiliste, la connexion à la base de données MongoDB, le chargement des places de stationnement disponibles par étage, ainsi que les paramètres dynamiques liés à la simulation.

Ensuite, les threads en arrière-plan sont lancés pour assurer la surveillance continue de certaines tâches (ex. : réservations ou états des places). La simulation entre alors dans une boucle principale au sein de laquelle plusieurs opérations sont répétées en continu. La première consiste à nettoyer les threads terminés pour éviter toute surcharge. Puis, le système calcule en temps réel le taux d'occupation du parking en comparant le nombre de véhicules actifs aux places totales.

En fonction de ce taux et du temps écoulé depuis le dernier pic d'arrivée (burst), le système détermine s'il doit simuler un nouveau pic d'arrivée de véhicules ou simplement une arrivée isolée. Un burst est déclenché si plus de 30 secondes se sont écoulées depuis le dernier et si une probabilité aléatoire est atteinte. Si ces conditions ne sont pas remplies, le système tente une arrivée isolée à travers une autre probabilité. Si aucune des deux conditions n'est satisfaite, aucune arrivée n'est simulée pour ce cycle.

Ensuite, les départs de véhicules sont simulés, et les statistiques globales (nombre de véhicules entrés/sortis, taux d'occupation, durée moyenne de stationnement, etc.) sont affichées dans la console. Ce cycle se répète jusqu'à l'interruption manuelle de la simulation par l'utilisateur.

3.9.2 Déroulement d'un cycle de simulation :

Pour illustrer concrètement le fonctionnement du système, cette section décrit un cycle complet de simulation, depuis l'arrivée d'un véhicule virtuel jusqu'à la mise à jour de l'interface gestionnaire.

Supposons qu'un véhicule virtuel arrive à 15h42 dans le mode « Mall » un samedi après-midi. Cette plage horaire est caractérisée par une affluence modérée mais constante, typique des centres commerciaux. Le simulateur détermine alors une durée de stationnement aléatoire. Dans ce scénario, la durée dépasse 30 minutes, ce qui déclenche une répartition vers les niveaux supérieurs pour éviter l'encombrement des niveaux les plus proches. Une place libre est alors choisie au troisième étage, par exemple la place numéro 12. Avant même l'arrivée du véhicule, la place sélectionnée passe temporairement à l'état « pré-occupé », ce qui la rend indisponible pour toute autre affectation durant quelques secondes. Cet état intermédiaire est représenté visuellement en orange sur le tableau de bord du gestionnaire, afin d'indiquer qu'un véhicule est en train d'y être assigné.

Le système génère un modèle de véhicule (ex. : Volkswagen Golf, size : regular), puis publie une série de messages MQTT sur des topics spécifiques simulant l'état des capteurs : capteur infrarouge, magnétomètre, caméra et capteur de statut global. En parallèle, les capteurs ultrason virtuels diffusent les distances vers les places voisines.

```

🚗 Car arriving: assigning ParkingSpot:Spot_L3_S12 (Volkswagen Golf, size: regular)
🆔 Car ID: fd6133d8-25c3-4785-93dd-16680084917a
⌚ Waiting at gate for 3.4s...
🕒 Driving to level 3... estimated time: 80.6s
📡 MQTT multi-attribute publish: /json/parkingkey/Spot_L3_S12/attrs => {'i': 'occupied'}
🔦 IR Sensor: occupied
📡 MQTT multi-attribute publish: /json/parkingkey/Spot_L3_S12/attrs => {'m': 'occupied'}
🧲 Magnetometer: occupied
📡 MQTT multi-attribute publish: /json/parkingkey/Spot_L3_S12/attrs => {'c': 'vehicle_detected'}
📷 Camera: vehicle_detected
🔍 Consistency Check | DB Entries: 1, DB Exits: 0, Active Cars: 1, Expected Occupied: 1
🕒 19:25 | Entered: 1 | Exited: 0 | Occupied: 1/80 | Avg Stay: 0s
📡 MQTT multi-attribute publish: /json/parkingkey/Spot_L3_S12/attrs => {'s': 'occupied'}
✅ Spot status set to occupied

```

Figure 17 : Log de simulation illustrant l'arrivée d'un véhicule dans le mode centre commercial.

Ces messages sont transmis via Mosquitto et interprétés par l'IoT Agent FIWARE, qui les transforme en entités NGSI. Orion Context Broker met alors à jour l'état de la place sélectionnée et conserve l'information dans MongoDB. En parallèle, l'interface de supervision développée en React interroge Orion via l'API NGSI-v2 pour afficher en temps réel les nouveaux états. La place numéro 12 au troisième étage apparaît désormais comme occupée.

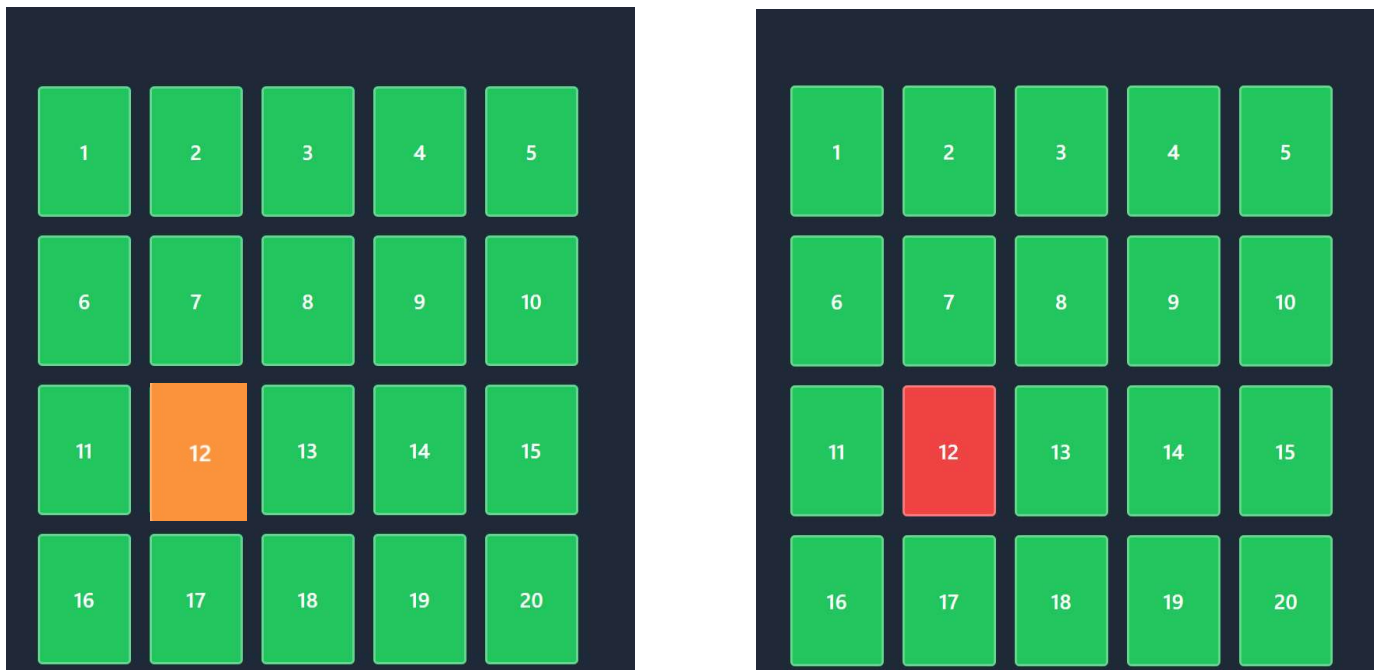


Figure 18 : Changement de l'état d'une place (orange → rouge) dans l'interface de gestion du Niveau 3

En complément, le tableau de bord React affiche également un résumé global du système dans la partie supérieure de l'interface. Cette vue permet à un gestionnaire de consulter rapidement :

- L'état de la barrière d'entrée (ouverte ou fermée)
- Le nombre total de véhicules entrés et sortis
- Le nombre de places actuellement occupées
- La durée moyenne de stationnement
- Les actions rapides telles que la visualisation des réservations, l'annulation ou encore les statistiques détaillées

La figure 18 illustre un aperçu de l'interface principale du tableau de bord du gestionnaire de parking. Des boutons permettent aussi de basculer entre les niveaux du parking, chacun étant représenté visuellement avec des codes couleur intuitifs (rouge = occupé, bleu = réservé, orange = pré-occupé, jaune = réservé et occupé, vert = libre). Toute réservation effectuée est enregistrée dans la liste des réservations comme illustré dans la figure 25.

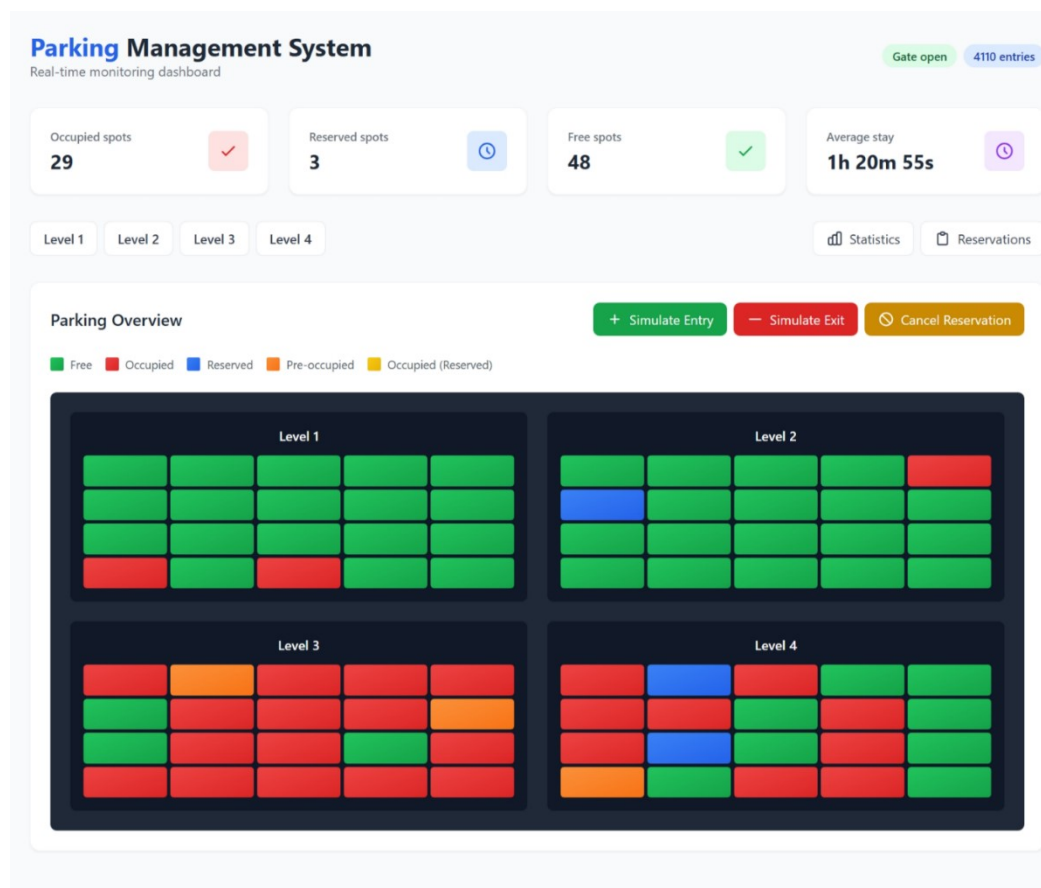


Figure 19 : Interface principale du tableau de bord du gestionnaire de parking.

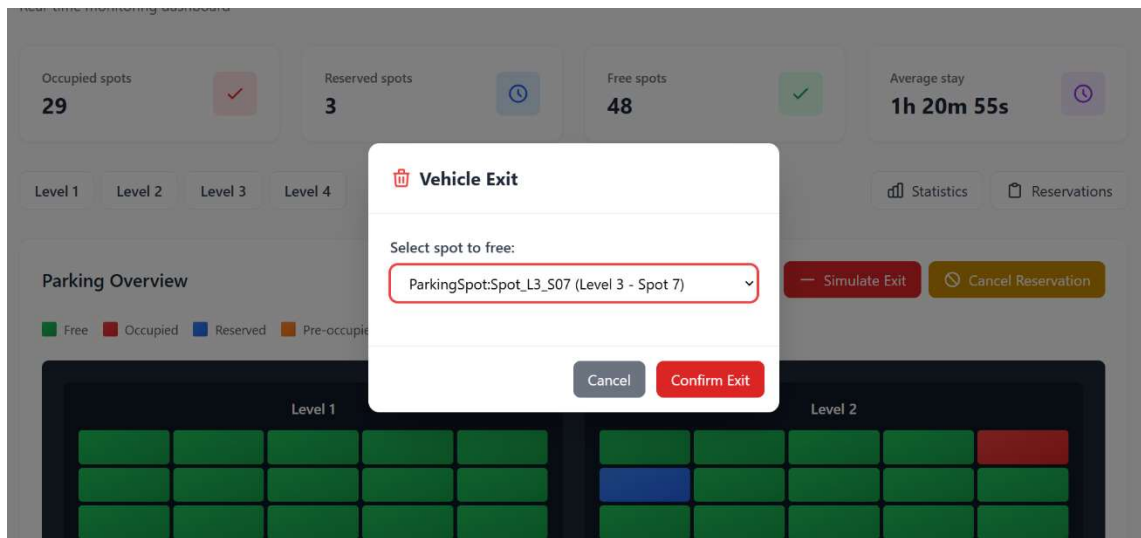


Figure 20 : interface simuler sortie.

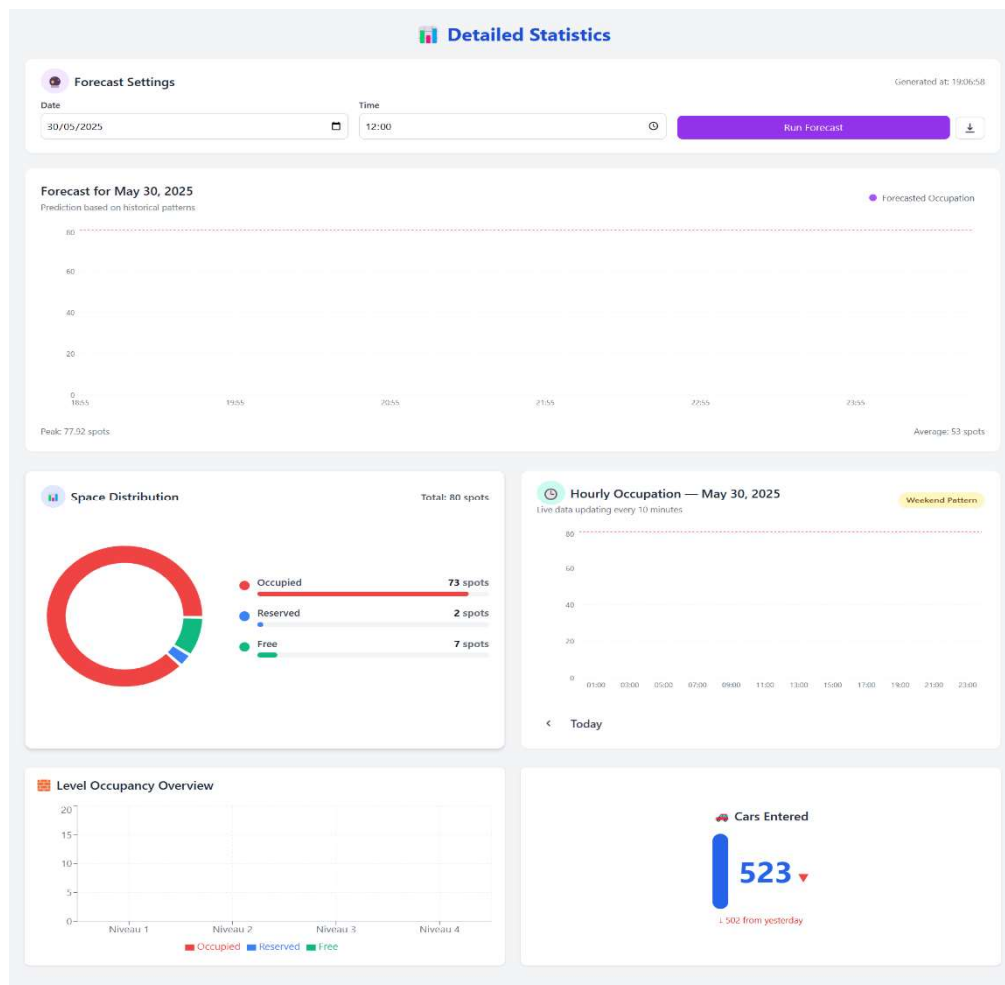


Figure 21 : interface statestique detaillé .

Après écoulement de la durée de stationnement, le simulateur déclenche la sortie du véhicule. Les messages MQTT de libération sont publiés, Orion met à jour l'état de la place, et la libération est visible sur l'interface en temps réel.

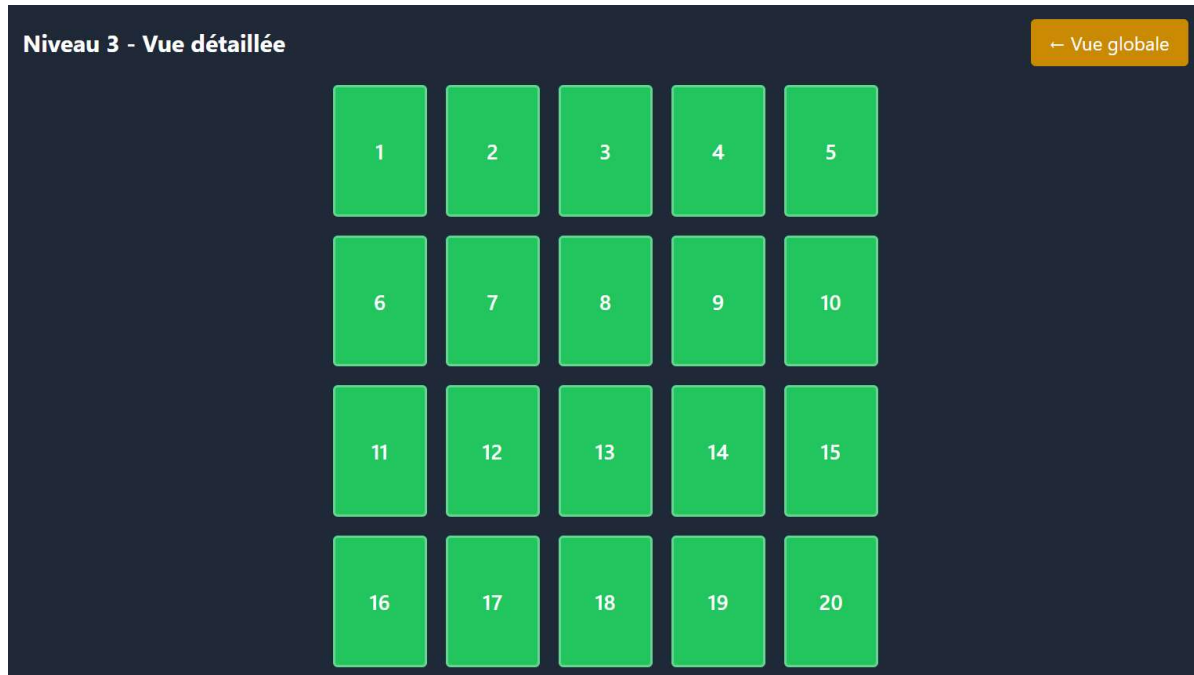


Figure 22 : Libération automatique de la place simulée au terme de la durée

3.9.3 Système de réservation utilisateur :

En plus de la simulation d'occupation classique, le système intègre un module de réservation permettant à un utilisateur de réserver une place à l'avance. Ce mécanisme ajoute une dimension plus réaliste et interactive au projet, tout en améliorant la fluidité et l'organisation du stationnement.

L'utilisateur accède à l'interface via une page d'authentification en saisissant son matricule et son mot de passe.

Figure 23 : Interface de connexion utilisateur (authentification via matricule et mot de passe)

Une fois connecté, il peut accéder à trois fonctionnalités principales :

1. Réserver une place
2. Consulter ses réservations existantes
3. Annuler une ou plusieurs réservations

Lorsqu'un utilisateur choisit de faire une réservation, un formulaire lui demande de renseigner les informations suivantes : Nom, prénom, modèle du véhicule, durée du stationnement (en minutes) et l'heure de réservation souhaitée.

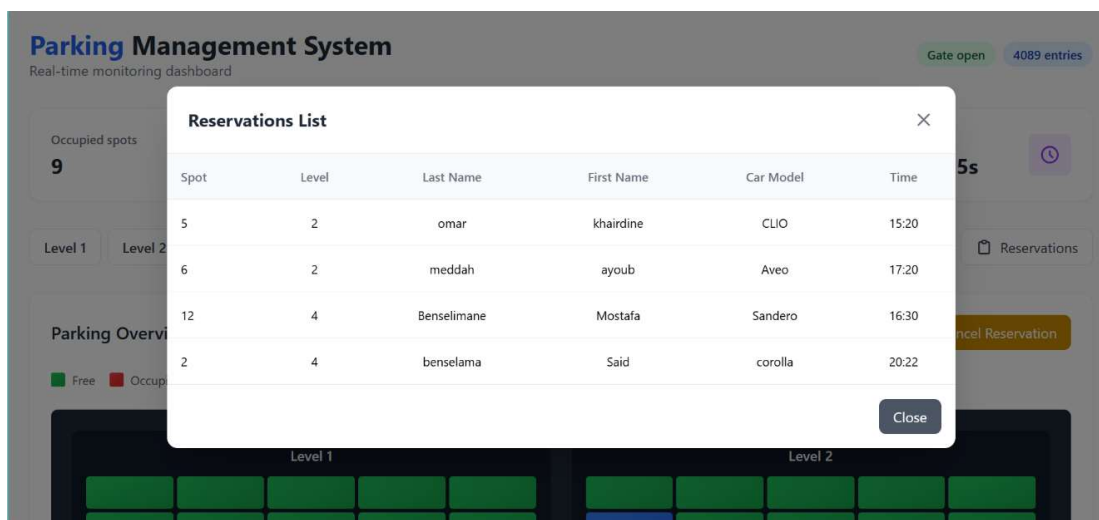
Figure 24 :Formulaire de réservation d'une place de stationnement

Le système attribue automatiquement une place selon la durée de réservation. Si la durée est inférieure ou égale à 30 minutes, la réservation est orientée vers les étages inférieurs (1 ou 2). Si la durée dépasse 30 minutes, une place est réservée dans les étages supérieurs (3 ou 4), afin de désengorger les niveaux plus accessibles. Une fois la réservation validée, la place attribuée passe à l'état « réservée » et s'affiche en bleu dans



le tableau de bord du gestionnaire, indiquant qu'elle est bloquée en attente d'arrivée. Ce statut permet d'éviter qu'un autre véhicule ne l'occupe pendant ce temps.

Figure 25: Visualisation d'une place réservée (affichée en bleu) dans l'interface gestionnaire.



Spot	Level	Last Name	First Name	Car Model	Time
5	2	omar	khairdine	CLIO	15:20
6	2	meddah	ayoub	Aveo	17:20
12	4	Benselimane	Mostafa	Sandero	16:30
2	4	benselama	Said	corolla	20:22

Figure 26 : interface liste des réservations.

Au moment de la simulation, un processus vérifie en continu si l'heure d'une réservation active est atteinte. Lorsque c'est le cas, la place réservée passe à l'état « pré-occupée » (orange) quelques secondes avant l'arrivée, puis devient « occupée » (rouge) dès que la voiture virtuelle est injectée dans le parking. Ce mécanisme garantit que la réservation bloque la place au bon moment, sans conflit avec la simulation classique.

Toutes les informations de réservation, y compris les actions de création ou d'annulation, sont enregistrées dans MongoDB pour assurer une traçabilité complète.

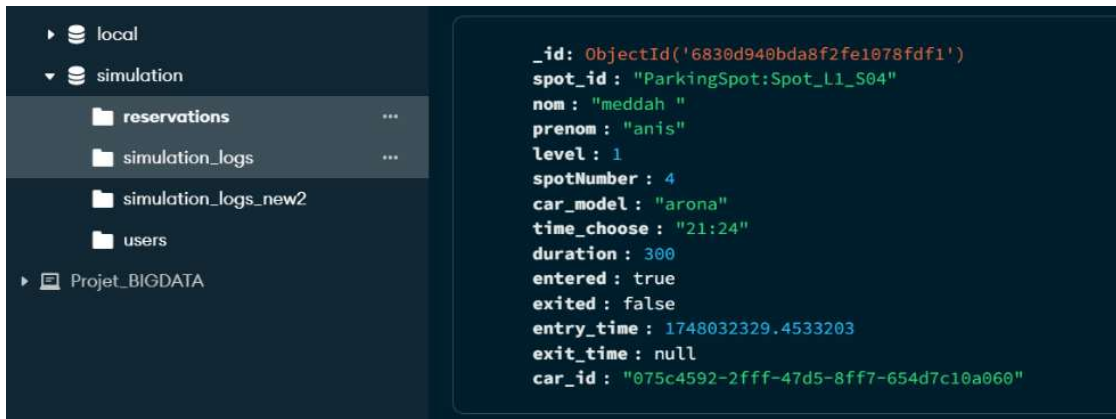


Figure 27 : Enregistrement des réservations dans la base de données MongoDB

Ce fonctionnement permet d’assurer un suivi fluide et fiable du cycle de réservation, depuis la demande initiale jusqu’à l’occupation effective de la place, tout en garantissant la synchronisation entre simulation, base de données et interface utilisateur.

3.10 Diagramme de séquence :

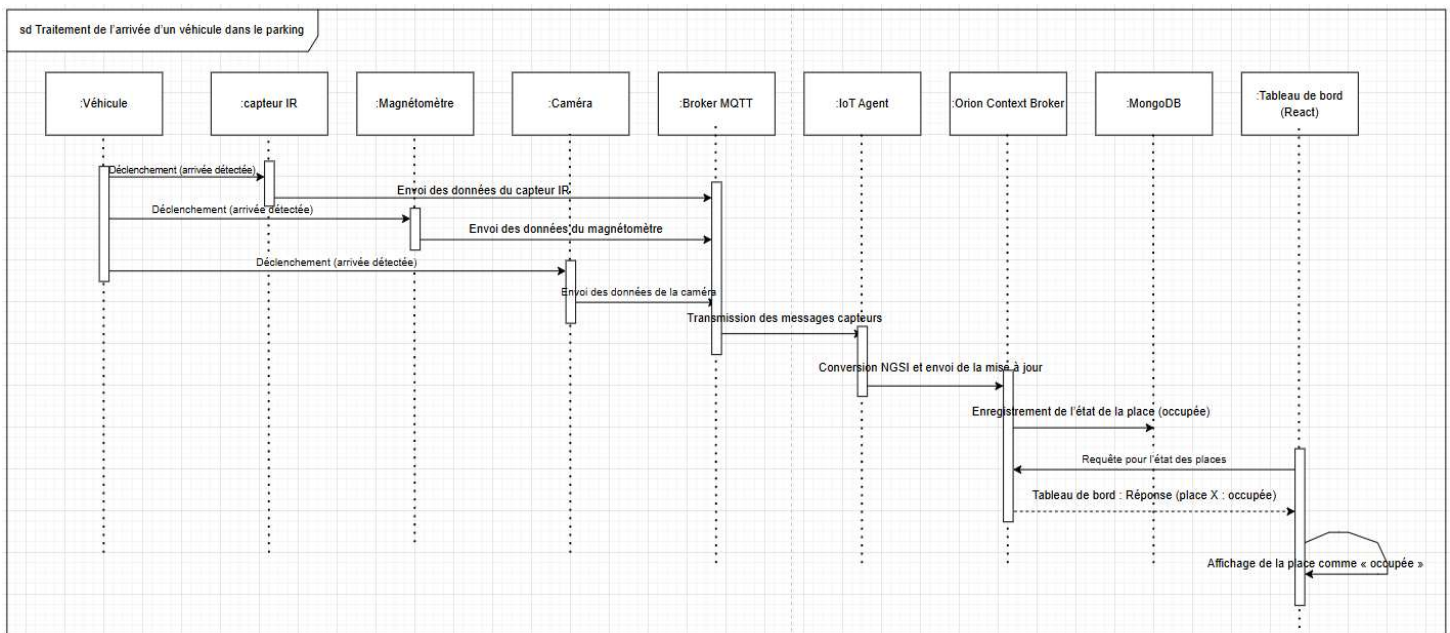


Figure 28 : Diagramme de séquence Occupation d’une place dans le système de gestion intelligent.

Ce diagramme de séquence décrit le déroulement complet de l’occupation d’une place de stationnement, depuis l’arrivée d’un véhicule détectée par les capteurs jusqu’à la mise à jour de l’interface de supervision. Il met en évidence les échanges entre les différents composants du système (capteurs, MQTT, IoT Agent, Orion, MongoDB et tableau de bord), et montre comment l’état de la place est traité et affiché en temps réel.

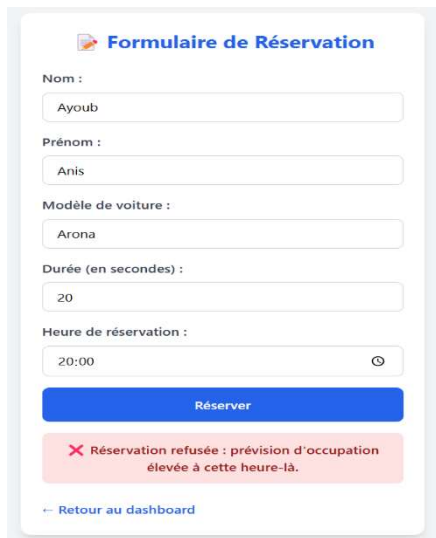
3.11 Intégration dans le simulateur :

Le modèle est intégré directement dans la logique de réservation. Lorsqu'un utilisateur envoie une requête pour réserver une place, le simulateur extrait les paramètres temporels associés (heure, jour, type de parking...) et les transmet au modèle. Celui-ci prédit le taux d'occupation attendu à ce moment-là.

Si la prédiction dépasse 80 %, la réservation est bloquée. Dans le cas contraire, elle est autorisée et enregistrée.

Ce mécanisme permet de renforcer la robustesse du système et d'introduire une forme d'intelligence adaptative dans le processus de décision, sans dépendre uniquement de l'état courant du parking.

Concrètement, lorsqu'un utilisateur soumet une demande de réservation, le simulateur interroge le modèle et retourne une décision en temps réel. Si la prédiction indique un taux d'occupation supérieur au seuil fixé, la réservation est automatiquement rejetée, comme illustré ci-dessous.



Formulaire de Réservation

Nom :
Ayoub

Prénom :
Anis

Modèle de voiture :
Arona

Durée (en secondes) :
20

Heure de réservation :
20:00

Réserver

❌ Réservation refusée : prévision d'occupation élevée à cette heure-là.

[Retour au dashboard](#)

Figure 29 : Interface utilisateur - refus automatique suite à la prédiction

En parallèle, le terminal du système affiche les détails techniques de cette prédiction, comme le montre l'extrait ci-dessous.

```

INFO: 127.0.0.1:56642 - "GET /gate-status HTTP/1.1" 200 OK
INFO: 127.0.0.1:56642 - "GET /reservations HTTP/1.1" 200 OK
INFO: 127.0.0.1:56642 - "GET /spots HTTP/1.1" 200 OK
INFO: 127.0.0.1:56642 - "GET /stats HTTP/1.1" 200 OK
INFO: 127.0.0.1:56642 - "GET /spots HTTP/1.1" 200 OK
INFO: 127.0.0.1:56642 - "GET /stats HTTP/1.1" 200 OK
INFO: 127.0.0.1:56642 - "GET /gate-status HTTP/1.1" 200 OK
INFO: 127.0.0.1:56642 - "GET /reservations HTTP/1.1" 200 OK
INFO: 127.0.0.1:61252 - "OPTIONS /reserve HTTP/1.1" 200 OK
🔵 Predicted occupancy at 2025-05-28 20:00:00+01:00: 75.42
INFO: 127.0.0.1:61252 - "POST /reserve HTTP/1.1" 200 OK

```

Figure 30 : Journal système au moment de la réservation.

3.12 Courbe de prédiction générée par le modèle :

Afin d'évaluer le comportement du modèle sur un intervalle temporel plus large, une visualisation des prédictions a été générée à partir des données simulées. Cette courbe met en parallèle, d'une part, les données d'occupation réellement observées avant l'instant de prédiction (courbe noire), et d'autre part, les valeurs prédites pour les minutes ou heures suivantes (courbe verte). Elle permet de valider la cohérence des estimations produites par le modèle XGBoost et d'observer visuellement son aptitude à suivre les tendances de fréquentation du parking. La ligne rouge verticale indique le point de bascule entre historique et projection.

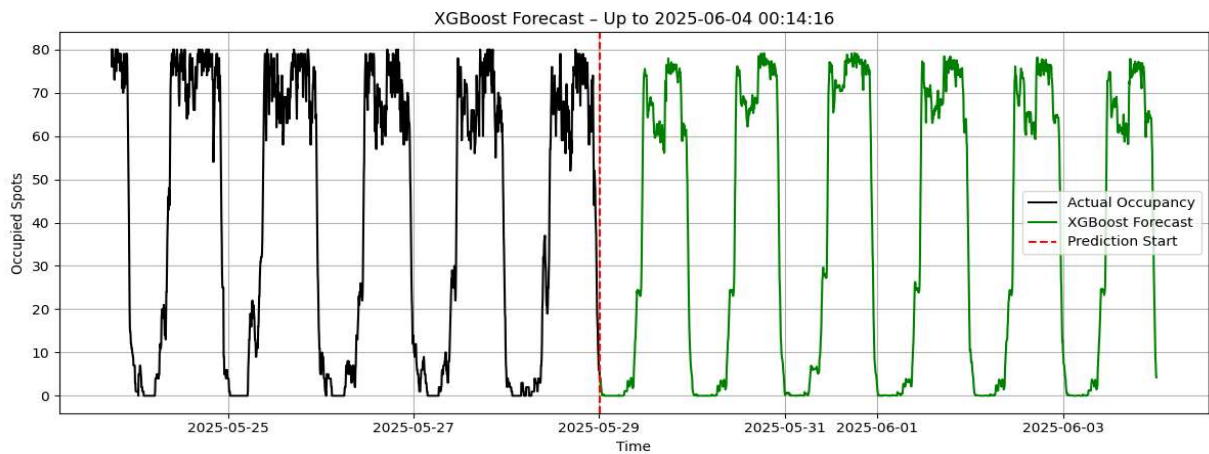


Figure 31 : Courbe de prédiction de l'occupation du parking

En parallèle du modèle prédictif d'occupation, une seconde approche expérimentale a été développée, toujours à l'aide de **XGBoost**, mais avec un objectif différent : déterminer automatiquement la place de stationnement la plus appropriée pour un véhicule donné, en fonction de son contexte d'arrivée. Ce modèle

apprend à partir de données historiques les associations les plus efficaces entre durée de stationnement, heure, jour, niveaux d'occupation par étage et affectation finale.

Contrairement à la fonction `find_free_unreserved_spot()`, qui applique une règle déterministe figée (ex. : envoyer systématiquement les séjours courts aux niveaux 1 et 2), le modèle adoptant une approche par apprentissage permet une allocation plus souple et adaptative. Il peut ainsi proposer une place optimale même dans des cas limites, comme une saturation partielle, un comportement atypique des usagers ou une répartition déséquilibrée des véhicules.

Ce modèle a été testé de manière expérimentale à petite échelle, afin d'observer sa capacité à reproduire, voire à affiner, la logique métier initialement codée. Les résultats de cette évaluation sont présentés et comparés dans la section suivante

3.13 Résultats obtenus et analyse du système

3.13.1 Statistiques globales de la simulation :

L'analyse des données collectées pendant la simulation permet de dresser un premier aperçu du fonctionnement global du système. Ces statistiques donnent une vision synthétique de la charge générée, du comportement d'occupation et de la dynamique du trafic dans le parking simulé.

Sur l'ensemble de la période de simulation de 23 mai au 29 mai 2025, le système a enregistré :

Tableau 05 – Statistiques globales de la simulation

Indicateur	Valeur
Nombre total d'entrées simulées	6 956 véhicules
Durée moyenne de stationnement estimée	49,93 minutes
Scénario de simulation utilisé	Mall

Indicateur	Valeur
	(100 %)

Tableau 06 – Répartition des véhicules par étage

Niveau du parking	Nombre de véhicules
Niveau 1	1 687
Niveau 2	1 634
Niveau 3	1 804
Niveau 4	1 831

Ces statistiques montrent une répartition relativement équilibrée entre les différents niveaux du parking, ce qui reflète une circulation fluide et réaliste dans un scénario de type centre commercial. Bien que la simulation inclue des véhicules de tailles variées, ce paramètre n'a pas encore été intégré dans la logique d'attribution des places.

3.13.2 Validation de la logique d'attribution verticale des places :

En observant les données issues de la simulation, on constate que la logique d'attribution des étages semble bien appliquée. Comme on peut le voir dans le tableau 9, les véhicules ont été répartis de manière assez équilibrée entre les différents niveaux, avec tout de même un léger surplus au niveau des étages 3 et 4.

Ce constat s'explique par le fait que la durée moyenne de stationnement dépasse 49 minutes. Il est donc logique que la majorité des voitures aient été dirigées vers les niveaux prévus pour les stationnements longs. Cela confirme que la logique d'affectation verticale fonctionne bien dans la pratique et que le simulateur applique correctement les règles définies.

3.13.3 Analyse comparée : occupation prédite et observée :

La figure suivante représente l'évolution complète de l'occupation du parking simulée et prédite sur plusieurs jours. L'analyse se concentre ici sur la période du **27 au 29 mai 2025**, durant laquelle on peut observer

un bon alignement entre les courbes réelle (en noir) en haut et prédite (en vert) en bas. Les pics d'occupation sont bien capturés, notamment en fin d'après-midi, ainsi que les périodes de relâche nocturnes. Le modèle anticipe correctement la dynamique journalière, bien que certaines transitions (notamment les descentes rapides en fin de journée) soient légèrement lissées.

Cette concordance visuelle renforce la fiabilité du modèle pour prendre des décisions automatiques dans le système de réservation, même à court terme.

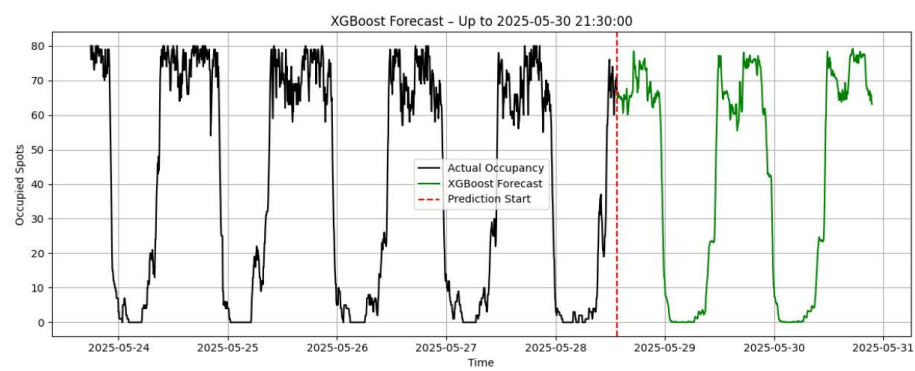


Figure 2 – Prédiction multi-jours (jusqu'au 4 juin 2025)

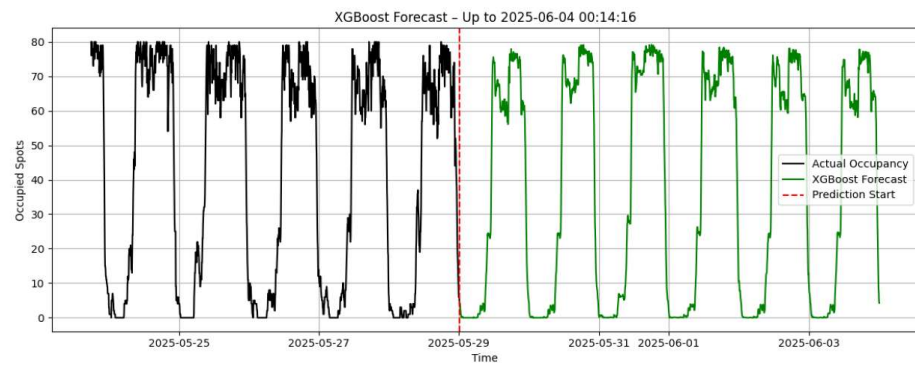


Figure32 : Occupation réelle vs prédite du parking.

Afin d'évaluer numériquement la précision du modèle, une comparaison a été effectuée entre les prédictions de l'occupation future et les valeurs réellement observées sur une même plage horaire, du **30 mai** à partir de **14h**.

Le tableau ci-dessous présente un extrait de cette comparaison :

Tableau 07 - Comparaison entre les prédictions du modèle et les données réelles d'occupation

Horodage	Prédiction	réel	Erreur absolue	Erreur relative (%)
2025-05-30 15 :30	69.31	4	4.69	6.34 %
2025-05-30 15 :35	71.43	2	0.57	0.79 %
2025-05-30 15 :40	73.15	1	2.15	3.03 %
2025-05-30 15 :45	73.15	1	2.15	3.03 %
2025-05-30 15 :50	71.89	8	3.89	5.71 %

En moyenne, l'erreur absolue s'élève à **2,87 véhicules**, soit une **erreur relative de 3,86 %**, ce qui témoigne d'une bonne fiabilité du modèle dans ce scénario.

3.13.4 Comparaison entre approche logique et modèle prédictif dans l'attribution des places :

- **Approche 1 : fonction d'optimisation classique**

La logique de la fonction `find_free_unreserved_spot()`, déjà décrite dans une section précédente, repose sur des règles déterministes en lien avec la durée estimée du stationnement, avec les longues durées en priorité affectées vers les étages supérieurs et les stationnements courts, vers les niveaux inférieurs. Si cette méthode simple et efficace peut servir de base à l'élargissement proposé, elle reste néanmoins peu souple en considérant peu de paramètres dynamiques tels que la charge réelle des étages ou l'instant horaire dans une situation complexe.

- **Approche 2 : modèle prédictif XGBoost**

Le deuxième modèle repose sur un algorithme XGBoost entraîné à partir des données historiques. Pour chaque place libre, il estime une probabilité de succès (ou de pertinence) en prenant en compte plusieurs facteurs contextuels, notamment :

- La durée prévue du stationnement (avec encodage logarithmique),

- Le moment de la journée (groupe horaire, heure sin/cos),
- Le jour de la semaine et la distinction week-end,
- Le taux d'occupation local de l'étage concerné.

À partir de ces informations, le modèle affecte un score à chaque place disponible. Le simulateur sélectionne ensuite la place ayant le meilleur compromis entre prédiction du modèle et niveau de saturation, dans une logique d'optimisation globale du parking.

Tableau 08– Répartition des véhicules par niveau selon l'approche d'attribution

Approche	Stationnement long ?	Niveau 1	Niveau 2	Niveau 3	Niveau 4
Fonction d'optimisation	Non (< 30 min)	51	34	9	5
	Oui (≥ 30 min)	136	100	775	796
IA	Non (< 30 min)	6	5	1	4
	Oui (≥ 30 min)	5	09	26	25

Il est observable ici que la logique d'un modèle d'optimisation est appliquée de façon très stricte, avec une affectation par durée, qui propulse vers les niveaux les plus élevés (**3 et 4**) les longues durées (≥30), alors que le modèle prédictif les répartit de manière plus libre et contextuelle, sans suivre une règle systématique.

On note bien une différence de régularité qui témoigne de l'approche plus adaptative de l'**IA** elle-même, qui prétend être plus contextuelle et cherche un équilibre de charge au temps où elle fonctionne.

L'approche déterministe offre une logique simple, transparente et facilement contrôlable. En revanche, le modèle prédictif permet une flexibilité plus fine, notamment en période de saturation, et peut mieux s'adapter à des comportements utilisateur imprévisibles. Ces deux stratégies ne sont pas nécessairement exclusives : une combinaison des deux (logique + IA) pourrait offrir un compromis optimal entre robustesse, lisibilité et adaptabilité.

3.13.5 Conclusion :

Ce chapitre a présenté en détail la solution technique développée pour la gestion intelligente d'un parking multi-niveaux, en exposant l'environnement de développement, l'architecture fonctionnelle, ainsi que la méthodologie de simulation mise en place. L'intégration de capteurs simulés, la gestion des communications via MQTT, la persistance des données dans MongoDB et la conception de l'interface web ont permis de construire un système modulaire et opérationnel.

Par ailleurs, les mécanismes d'optimisation tels que la répartition verticale des véhicules en fonction de la durée de stationnement et la logique de réservation ont été rigoureusement analysés. Les différentes phases de tests et d'évaluation décrites dans ce chapitre ont permis de mesurer l'impact de chaque choix technique sur le comportement global du simulateur, révélant les forces et les limites de l'approche adoptée.

L'implémentation d'un modèle prédictif basé sur XGBoost a ensuite apporté une dimension analytique supplémentaire, en offrant une prédiction dynamique du taux d'occupation et en permettant de comparer de manière critique deux stratégies d'attribution de places. Les analyses effectuées sur ces différents aspects ont ainsi permis d'établir la cohérence et l'efficacité globale du système simulé.

Conclusion générale :

Ce mémoire a présenté le développement d'une solution intelligente pour la gestion de parkings multi-niveaux, s'appuyant sur les technologies IoT et l'intelligence artificielle. À travers une simulation réaliste, nous avons démontré comment l'intégration de capteurs, de protocoles de communication légers et d'algorithmes de décision permet d'optimiser à la fois l'occupation des places, la fluidité du trafic interne, et l'expérience utilisateur.

L'approche adoptée a mis en évidence plusieurs mécanismes d'optimisation, notamment la répartition intelligente des véhicules selon la durée du stationnement, la gestion des réservations, et la prédiction de l'occupation à l'aide d'un premier modèle XGBoost. Un second modèle, également basé sur XGBoost, a été expérimenté pour attribuer directement une place optimale en fonction du contexte, apportant une nouvelle forme d'intelligence décisionnelle au système. Les résultats obtenus confirment la cohérence de la logique implémentée, ainsi que la pertinence de l'intelligence artificielle pour renforcer la robustesse et la flexibilité du système.

En conclusion, cette solution pose les bases d'un système évolutif et adaptable aux contextes urbains modernes, et ouvre la voie à de futures extensions intégrant davantage de scénarios réels, de données comportementales et de mécanismes prédictifs avancés.

Perspectives futures :

Les résultats obtenus ouvrent plusieurs pistes d'amélioration et de développement du système. Parmi les perspectives envisagées :

- **Renforcement du modèle prédictif**

L'intégration du modèle XGBoost pourrait être enrichie par l'ajout de variables contextuelles comme les événements locaux ou les habitudes comportementales. Cela permettrait d'améliorer la précision des prédictions, notamment dans des situations imprévues.

- **Adoption de modèles neuronaux avancés**

L'utilisation de réseaux neuronaux récurrents (RNN, LSTM) pourrait être explorée pour mieux capter les variations temporelles des flux de véhicules. Ces architectures permettraient de modéliser plus finement les séquences d'occupation dans le temps.

- **Mise en œuvre d'une stratégie hybride**

Une logique combinée pourrait être développée, dans laquelle le système basculerait dynamiquement entre la fonction d'optimisation classique et les recommandations IA, en fonction du taux de saturation ou du moment de la journée.

- **Authentification et expérience utilisateur enrichie**

La mise en place d'un vrai système multi-utilisateurs permettrait à chaque conducteur de se connecter avec son propre compte, de suivre son historique et de gérer ses réservations. Une application mobile dédiée pourrait également être développée pour faciliter l'accès au service à distance.

- **Validation sur données réelles**

Une évaluation du système en environnement réel (ou semi-réel) constituerait une étape clé pour valider les performances du simulateur. Cela pourrait se faire via un partenariat avec une collectivité ou un gestionnaire de parking.

- **Intégration d'un système de tarification intelligente**

Le système pourrait être étendu avec une couche tarifaire intelligente, tenant compte de la durée de stationnement, du niveau occupé, de la période (heure de pointe, week-end...), voire du profil utilisateur.

Bibliographie:

[1] Parking.net, "History of Parking," [Online]. Available: <https://www.parking.net/about-parking/history-of-parking>. [Consulté le: 05-Mar-2025]

[2] R. Jatteau, "Le stationnement urbain en ébullition," *La Tribune*, Sept. 16, 2020. [Online]. Available: <https://www.latribune.fr/opinions/tribunes/le-stationnement-urbain-en-ebullition-857555.html>. [Consulté le: 05-Mar-2025]

[3] Parking-Actus, "Fonctionnement d'une barrière de parking," [Online]. Available: <https://www.parking-actus.com/fonctionnement-barriere-parking/>. [Consulté le: 05-Mar-2025]

[4] Display Media, "Bornes de paiement digitales," [Online]. Available: <https://www.displaymedia.fr/borne-de-paiement/>. [Consulté le: 05-Mar-2025]

[5] Diaz Ogás, M. G., Fabregat, R., & Aciar, S. (2020). Survey of smart parking systems. *Applied Sciences*, 10(11), 3872. [Consulté le: 12-Mar-2025]

[6] NCBI, "Enabling smart parking for smart cities using Internet of Things (IoT) and machine learning," [Online]. Available: <https://pmc.ncbi.nlm.nih.gov/articles/PMC11784892/>. [Consulté le: 12-Mar-2025]

- [7] Application IoT, "Des capteurs de parking connectés pour une gestion intelligente des places de stationnement," [Online]. Available: <https://www.application-iot.fr/batiment-intelligent/des-capteurs-de-parking-connectes/>. [Consulté le: 12-Mar-2025]
- [8] PRM Parken, "Systèmes de paiement sans contact dans la gestion des parkings : augmenter l'efficacité et la satisfaction des usagers," [Online]. Available: <https://www.prm-parken.de/fr/kontaktlose-zahlungssysteme-im-parkraummanagement-effizienz-und-nutzerzufriedenheit-steigern/>. [Consulté le: 12-Mar-2025]
- [9] Shaikh, F. I., Jadhav, P. N., Bandarkar, S. P., Kulkarni, O. P., & Shardoor, N. B. (2016). Smart parking system based on embedded system and sensor network. *International Journal of Computer Applications*, 140(12). [Consulté le: 12-Mar-2025]
- [10] Floris, A., Girau, R., Porcu, S., Pettorru, G., & Atzori, L. (2020, September). Implementation of a magnetometer-based vehicle detection system for smart parking applications. In *2020 IEEE International Smart Cities Conference (ISC2)* (pp. 1-7). IEEE. [Consulté le: 12-Mar-2025]
- [11] Jeong, S. H., Choi, C. G., Oh, J. N., Yoon, P. J., Kim, B. S., Kim, M., & Lee, K. H. (2010). Low cost design of parallel parking assist system based on an ultrasonic sensor. *International Journal of Automotive Technology*, 11, 409-416. [Consulté le: 12-Mar-2025]
- [12] Bura, H., Lin, N., Kumar, N., Malekar, S., Nagaraj, S., & Liu, K. (2018, July). An edge based smart parking solution using camera networks and deep learning. In *2018 IEEE international conference on cognitive computing (ICCC)* (pp. 17-24). IEEE. [Consulté le: 12-Mar-2025]
- [13] Cai, Z., Zhou, Y., Qi, Y., Zhuang, W., & Deng, L. (2020). A millimeter wave dual-lens antenna for IoT-based smart parking radar system. *IEEE Internet of Things Journal*, 8(1), 418-427. [Consulté le: 13-Mar-2025]
- [14] Lin, T., Rivano, H., & Le Mouél, F. (2017). A survey of smart parking solutions. *IEEE Transactions on Intelligent Transportation Systems*, 18(12), 3229-3253. [Consulté le: 13-Mar-2025]
- [15] MOKOSmart, "Système de stationnement intelligent utilisant l'IoT," [Online]. Available: <https://www.mokosmart.com/fr/smart-parking-system-using-iot/>. [Consulté le: 13-Mar-2025]
- [16] Amazon Web Services, "Qu'est-ce que MQTT ?", [Online]. Available: <https://aws.amazon.com/fr/what-is/mqtt/> [Consulté le: 16-Mar-2025]

- [17] Herrera-Quintero, L. F., Vega-Alfonso, J., Bermúdez, D., Marentes, L. A., & Banse, K. (2019, May). ITS for Smart Parking Systems, towards the creation of smart city services using IoT and cloud approaches. In *2019 smart city symposium Prague (SCSP)* (pp. 1-7). IEEE. [Consulté le: 16-Mar-2025]
- [18] Qadir, Z., Al-Turjman, F., Khan, M. A., & Nesimoglu, T. (2018, October). ZIGBEE based time and energy efficient smart parking system using IOT. In *2018 18th mediterranean microwave symposium (MMS)* (pp. 295-298). IEEE. [Consulté le: 16-Mar-2025]
- [19] Yuan, C., Fei, L., Jianxin, C., & Wei, J. (2016, May). A smart parking system using WiFi and wireless sensor network. In *2016 IEEE International Conference on Consumer Electronics-Taiwan (ICCE-TW)* (pp. 1-2). IEEE. [Consulté le: 16-Mar-2025]
- [20] Shi, J., Jin, L., Li, J., & Fang, Z. (2017, September). A smart parking system based on NB-IoT and third-party payment platform. In *2017 17th international symposium on communications and information technologies (ISCIT)* (pp. 1-5). IEEE. [Consulté le: 16-Mar-2025]
- [21] Kadusic, E., Zivic, N., Ruland, C., & Hadzajlic, N. (2022). A smart parking solution by integrating NB-IoT radio communication technology into the core IoT platform. *Future Internet*, 14(8), 219. [Consulté le: 16-Mar-2025]
- [22] Vijayalakshmi, S., Bose, S., Logeswari, G., & Maheswaran, N. (2025). Smart parking: intelligent intrusion detection system in VANET enabled car parking system. *Automatika*, 66(2), 281-299. [Consulté le: 26-Mar-2025]
- Barriga, J. J., Sulca, J., León, J. L., Ulloa, A., Portero, D., Andrade, R., & Yoo, S. G. (2019). Smart parking: A literature review from the technological perspective. *Applied Sciences*, 9(21), 4569. [Consulté le: 26-Mar-2025]
- [23] Science Post, "Innovations technologiques et expérience du parking," [Online]. Available: <https://sciencepost.fr/innovations-technologiques-et-experience-du-parking/> [Consulté le: 26-Mar-2025]
- [24] Cleveroad, "Parking App Development: How to Build a Smart Parking System," [Online]. Available: <https://www.cleveroad.com/blog/parking-app-development/>. [Consulté le: 26-Mar-2025]
- [25] Paidi, V., Fleyeh, H., Håkansson, J., & Nyberg, R. G. (2018). *Smart parking sensors, technologies and applications for open parking lots: a review. IET Intelligent Transport Systems*. doi:10.1049/iet-its.2017.0406

[26]GCE Lab, "Smart Parking Solutions: Revolutionizing Urban Mobility and Convenience," [Online]. Available:<https://www.gcelab.com/blog/smart-parking-solutions-revolutionizing-urban-mobility-and-convenience>. [Consulté le: 26-Mar-2025]

[27]Sawant, M., Salunke, O., Raut, S., & Suman, S. (2020, April). Multi-level smart parking system. In *Proceedings of the 3rd International Conference on Advances in Science & Technology (ICAST)*. [Consulté le:10-Avr-2025]

[28]Sethy, J., Behera, S., & Panda, S. (2020). Multi-level car parking. *Journal of Shanghai Jiaotong University*. [Consulté le: 13-Avr-2025]

[29]Bhubaneswar Smart City Limited. (n.d.). *Multilevel Car Parking RajMahal*. Smart City Bhubaneswar. <https://www.smartcitybhubaneswar.gov.in/rajmahal-square> . [Consulté le: 13-Avr-2025]

[30]Tahmasseby, S. (2022). The implementation of smart mobility for smart cities: A case study in Qatar. *Civil Engineering Journal*, 8(10), 2154-2171[Consulté le: 13-Avr-2025]

[31]QDVC. (n.d.). *Lusail Car Parks*. <https://www.qdvc.com/pf/lusail-car-parks/> . [Consulté le: 13-Avr-2025]

[32]Fung, W. (2017). *Evaluating Smart Parking Performance in a Dublin Scenario* (Master's thesis, Technological University Dublin).[Consulté le: 16-Avr-2025]

[33]Zhao, X., et al. (2018). Advanced parking space management strategy design: An agent-based simulation optimization approach. *Transportation Research Record*, 2672(00), 1–10. [Consulté le: 18-Avr-2025]

[34]Kharbeche, M., Ibrahim, A., & Lee, Y. (2019). An occupancy simulator for a smart parking system [Consulté le: 22-Avr-2025]

[35]Kummerle, R., Hahnel, D., Dolgov, D., Thrun, S., & Burgard, W. (2009). *Autonomous driving in a multi-level parking structure*. *2009 IEEE International Conference on Robotics and Automation*. [Consulté le: 17-Avr-2025]

[36] Chaloulakou, A., Duci, A., &Spyrellis, N. (2002). *Exposure to Carbon Monoxide in Enclosed Multi-Level Parking Garages in the Central Athens Urban Area*. *Indoor and BuiltEnvironment*, 11(4), 191–201. [Consulté le: 17-Avr-2025]

[37] XGBoost Developers. (2023). *XGBoost Documentation* — Python Package Introduction. [Consulté le 28 mai 2025].

- [38] The pandas development team. (2023). *pandas: powerful Python data analysis toolkit*. <https://pandas.pydata.org> [Consulté le 28 mai 2025].
- [39] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... & Duchesnay, E. (2011). Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12, 2825–2830 [Consulté le 28 mai 2025].
- [40] Hunter, J. D. (2007). Matplotlib: A 2D graphics environment. *Computing in Science & Engineering*, 9(3), 90–95 [Consulté le 28 mai 2025].
- [41] Harris, C. R., Millman, K. J., van der Walt, S. J., Gommers, R., Virtanen, P., Cournapeau, D., ... & Oliphant, T. E. (2020). ArrayprogrammingwithNumPy. *Nature*, 585, 357–362 [Consulté le 28 mai 2025].
- [42] Joblib Developers. (2023). *joblib documentation* <https://joblib.readthedocs.io> [Consulté le 28 mai 2025].
- [43] Python Software Foundation. (2023). About Python. <https://www.python.org/doc/essays/blurb/>. [Consulté le: 04-Mai-2025]
- [44] Merkel, Dirk. (2014). “Docker: Lightweight Linux Containers for Consistent Development and Deployment.” *Linux Journal*, 2014(239), 2. [consulté le 5 mai 2025].
- [45] Eclipse Foundation, Eclipse Mosquitto — An Open Source MQTT Broker, <https://mosquitto.org/> [consulté le 19 mai 2025].
- [46] FIWARE Foundation. (2025). *FIWARE Catalogue: Generic Enablers and Components*, from <https://www.fiware.org/catalogue/> [consulté le 19 mai 2025].
- [47] FIWARE Foundation. (2024). *IoT Agent Node Lib documentation*, from <https://iotagent-node-lib.readthedocs.io/en/latest/> [consulté le 19 mai 2025].
- [48] FIWARE Foundation. (2025). Orion Context Broker – User & Programmer Guide, from <https://fiware-orion.readthedocs.io/en/master/> [consulté le 19 mai 2025].
- [49] MongoDB Inc. (2025). MongoDB Manual – Introduction to MongoDB, from <https://www.mongodb.com/docs/manual/introduction/> [consulté le 19 mai 2025].
- [50] Meta (2025). React – A JavaScript library for building user interfaces, from <https://react.dev/> [consulté le 19 mai 2025].

[51] Tailwind Labs. (2025). Tailwind CSS – Rapidly build modern websites without ever leaving your HTML. Retrieved May 19, 2025, from <https://tailwindcss.com/>[consulté le 19 mai 2025].

[52] Thomas Nordquist. (2025). MQTT Explorer – A comprehensive MQTT client. Retrieved May 19, 2025, from <https://mqtt-explorer.com/>[Consulté le 19 mai 2025].

[53] Chen, T., & Guestrin, C. (2016). XGBoost: A Scalable Tree Boosting System. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '16)*, pp. 785–794[Consulté le 28 mai 2025].

[54] Liu, J., Wang, H., & Zhang, Y. (2022). *An IoT-Based Multi-Level Smart Parking System for Real-Time Space Monitoring*. **IEEE Internet of Things Journal**, 9(5), 3551–3562[Consulté le 31 mai 2025].

[55] Ahmed, M., Rehman, M. H., & Jan, M. A. (2021). *Deep Learning-Based Parking Availability Prediction for Smart Cities*. **Sensors**, 21(3), 765.[Consulté le 31 mai 2025]

[56] Kim, D., Lee, S., & Choi, J. (2023). *Smart Navigation System for Urban Parking Using Real-Time Data and Route Optimization*. **Transportation Research Part C: Emerging Technologies**, 148, 104029.[Consulté le 31 mai 2025]

الجمهورية الجزائرية الديمقراطية الشعبية
وزارة التعليم العالي والبحث العلمي

جامعة عبد الحميد بن باديس - مستغانم
كلية العلوم الدقيقة والإعلام الآلي

تصريح شرفي خاص بالالتزام بقواعد النزاهة العلمية
لإنجاز البحث

أنا الممضي أدناه،

الطالب (ة): السيد
رقم التسجيل الجامعي: 202029533
الحامل لبطاقة التعريف الوطنية رقم: 160002096305226
والصادرة بتاريخ: 16-01-2019
عن:
المسجل ب: كلية العلوم الدقيقة والإعلام الآلي / قسم:
شعبة:
والمكلف بإنجاز مذكرة ماستر بعنوان:

.....
.....

أصريح بشرفي أنني ألتزم بمراعاة المعايير العلمية والمنهجية ومعايير الأخلاقيات العلمية والنزاهة الأكاديمية المطلوبة في إنجاز البحث ، وأتحمل المسؤولية الشخصية عن كل المحتوى المتضمن في البحث المذكور أعلاه .

التاريخ: 23-06-2020

إمضاء المعني



* ملحق القرار الوزاري رقم 933 المؤرخ في 28 جويلية 2016 الذي يحدد القواعد المتعلقة بالوقاية من السرقة العلمية ومكافحتها.

الجمهورية الجزائرية الديمقراطية الشعبية
وزارة التعليم العالي والبحث العلمي

جامعة عبد الحميد بن باديس-مستغانم-
كلية العلوم الدقيقة والإعلام الآلي

تصريح شرفي خاص بالالتزام بقواعد النزاهة العلمية
لإنجاز البحث

أنا الممضي أدناه،

الطالب(ة): ممدوح محمد هيثم أيوب رقم التسجيل الجامعي: 202037028662
 الحامل لبطاقة التعريف الوطنية رقم: 202037028662 والصادرة بتاريخ: 27-11-2021
 عن: مستشفى

المسجل ب: كلية العلوم الدقيقة والإعلام الآلي / قسم : ID ٥١٤٤٤ ID ٥١٤٤٤

شعبة: إعلامية ذكاء: / التخصص: الذكاء الاصطناعي للشراء ذكية
اصطناعي

والمكلف بإنجاز مذكرة ماستر بعنوان:

اقتراح حل ذكي قناعكم هذه تقنيات انترنت. ¹⁰ لماذا هذا ا حل تيسر هو ا مق
البراسات متعددة الاطراف

أصرح بشرفي أنني ألتزم بمراعاة المعايير العلمية والمنهجية ومعايير الأخلاقيات العلمية والنزاهة الأكاديمية المطلوبة في إنجاز البحث ، وأتحمل المسؤولية الشخصية عن كل المحتوى المتضمن في البحث المذكور أعلاه .

التاريخ: 23-06-2025

إمضاء المعني

Ag³

* ملحق القرار الوزاري رقم 933 المؤرخ في 28 جويلية 2016 الذي يحدد القواعد المتعلقة بالوقاية من السرقة العلمية ومكافحتها.

