



Direction Générale des Études Technologiques
Institut Supérieur des Etudes Technologiques de Béja
Département des Technologies de l'Informatique



MEMOIRE DE PROJET DE FIN D'ÉTUDES

Présenté en vue de l'obtention du
Diplôme de Licence Appliquée en Technologies de l'Informatique
Spécialité : Systèmes embarquées et mobiles

Développement d'un système intelligent de gestion de parking basé sur la reconnaissance automatique de plaques d'immatriculation et l'analyse vidéo en temps réel

Réalisé Par :
ARIDHI RIHEM
NAHAWAND MEKNI

Encadrant Pédagogique :
M. Seif Ben Chaabene
Encadrant Entreprise :
M.Mohamed Dchia Houimli

Au sein de (Organisme d'accueil) : Kromberg & Schubert



Dédicaces

Je rends grâce à Dieu pour toutes les bénédictions qu'Il m'a accordées jusqu'à ce jour, me permettant d'atteindre cette étape et de finaliser ce travail.

Je dédie cet ouvrage à :

Mon père, Tawfik, pour son souci constant de me voir réussir, pour avoir été le premier à m'apprendre à écrire, pour sa patience inépuisable, ses sacrifices silencieux, et sa fatigue supportée dans le seul but de me voir accomplir mes études.

Merci d'être ce père exceptionnel, plein d'amour et de sagesse. Je te dois tant.

Ma mère, Mariam pour sa force, ses prières discrètes, son soutien indéfectible et ses encouragements permanents. Maman, ton amour et ton courage ont toujours été ma source de motivation.

Mon frère, Naseredine qui a toujours cru en moi et m'a soutenue pour surmonter chaque obstacle, peu importe le temps ou les circonstances.

Mon amie, Amira, pour son soutien précieux, sa présence constante, ses paroles motivantes et son courage qui m'a inspirée à persévérer. Amira, ta force m'a portée dans les moments difficiles et ton amitié est un véritable cadeau.

À mon binôme Nahawand, ta collaboration et ton dévouement ont été essentiels tout au long de ce projet. Nous avons formé une équipe exceptionnelle, et ton engagement a grandement contribué à notre réussite commune.

Aridhi RIHEM

Dédicaces

Je remercie Dieu de m'avoir donné la santé, la force et la persévérance pour mener ce travail à son terme. Je dédie ce travail

A mes chers parents Hinda et Hassen Qui n'ont, jamais cessé de m'aider, m'assister et m'encourager, ceux qui ont sacrifié leurs plus belles années pour embellir les miennes, je vous dois ma réussite, aucun mot ne serait assez pour témoigner de l'étendue des sentiments que j'éprouve à leur égard « Que dieu vous garde »

A ma chère sœur Nayrouz,

Ma petite sœur Pour son amour, son soutien constant et sa compréhension. Nayrouz, ta présence à mes côtés me donne la force de surmonter tous les obstacles. Ta patience et ton affection inébranlable ont été essentielles à la réussite de ce travail.

A ma chère sœur Nour

Aucun mot ne peut décrire les sentiments de fraternité et d'amitié qui nous attachent. Qu'nos soyons à la hauteur des sacrifices et des ambitions de nos parents.

À ma binôme Rihem,

Pour son bon cœur. Ta collaboration et ton dévouement ont été essentiels tout au long de ce projet. Rihem, notre complicité et notre travail d'équipe ont rendu cette aventure enrichissante et mémorable.

A mes chers amis

Qui m'ont toujours soutenu et qui étaient présents à mes cotés A tous ceux qui veulent partager ma joie, je vous dédie ce travail, le fruit de mes efforts et de longues années d'études, que vous y trouvez le couronnement de votre assistance et l'expression profonde de ma gratitude.

Meknî Nahawand

Remerciement

Nous souhaitons exprimer notre profonde gratitude à M. Mohamed Dhia Houimli, notre encadrant externe, pour son soutien inestimable et son accompagnement précieux tout au long de notre projet de fin d'études. Son expertise, sa patience et ses conseils éclairés ont grandement contribué à la réussite de notre rapport de PFE, et nous lui sommes extrêmement reconnaissants pour sa collaboration.

À notre encadrant interne M. Seif Ben Chaabene, nous le remercions pour son capacité à nous guider et à nous inspirer tout au long du processus a été essentielle pour surmonter les défis et atteindre nos objectifs. Ses commentaires constructifs et ses suggestions ont considérablement amélioré la qualité de notre travail, et nous lui sommes très reconnaissants pour son investissement personnel dans notre projet.

Nous tenons également à remercier chaleureusement toute l'équipe IT qui nous a accueillis et soutenus pendant cette période. Leur accueil bienveillant, leur esprit d'équipe et leur ouverture d'esprit ont créé un environnement propice à l'apprentissage et à la collaboration. Leur engagement à partager leurs connaissances et leur expérience a été une source d'inspiration pour nous, et nous sommes honorés d'avoir pu faire partie de leur équipe.

Nous sommes profondément reconnaissants envers nos enseignants à l'ISET de Beja, qui ont excellement rempli leur noble mission d'enseigner les bases de l'informatique. Nous les remercions non seulement pour les connaissances qu'ils nous ont transmises, mais aussi pour la fierté et l'ambition qu'ils ont suscitées en nous.

Nos respects et nos remerciements les plus distingués vont également aux membres du jury qui ont accepté d'évaluer notre travail.

Enfin, nous exprimons notre sincère gratitude à toutes les personnes qui ont contribué de près ou de loin à la réalisation de ce travail et pour toute l'aide qu'elles nous ont apportée.

Sommaire

Introduction Générale	10
Chapitre 1 : Cadre général du projet	12
Introduction	12
1.Présentation de l'entreprise d'accueil	12
1.1. Présentation générale	12
1.2. Service de kromberg & Schubert.....	12
1.3. Engagements de kromberg & Schubert	13
1.4. Distributeur Et Intégrateur De Solution De sécurité électrique	13
2. Analyse et étude de l'existant	15
2.1. Etude de l'existant	15
2.2. Description de l'existant.....	15
2.3. Critique de l'existant.....	15
2.4. Solution proposée	16
3.Choix méthodologique	16
3.1. Méthodologie de conception : Scrum.....	16
3.2. Le rôle de Scrum.....	17
3.3. Concept Scrum	18
4. Langage de modélisation UML	19
4.1. Langage de modélisation SYSML.....	20
Conclusion.....	20
Chapitre 2 : Etude conceptuelle : Sprint 0	22
Introduction	22
1. Analyse des besoins	22
1.1. Identification des acteurs	22
1.2. Besoins fonctionnels	22
1.3. Besoins non fonctionnels	24
1.4. Diagramme de cas d'utilisation globale.....	24
2. Pilotage du projet avec SCRUM	25
2.1. Equipes et rôles	25
2.2. Backlog Product	26
2.3. Planification des sprints	27
2. Etude et choix électronique	28
2.1. Choix du microcontrôleur	28
2.2. Choix du capteur (Caméra).....	29

2.3. Différence entre Caméra analogique et Caméra IP	30
3. Environnement matériel	30
3.1. Carte Raspberry Pi.....	30
3.2. Carte ESP32 DevKitC.....	31
3.3. Périphériques électroniques utilisés.....	32
3.4. Caméra Dahua.....	33
4. Environnement logiciel	34
5. Diagramme de classe	38
6. Architecture système	39
7. Diagramme de déploiement.....	40
Conclusion.....	41
Chapitre 3 : Sprint 1	42
Introduction	42
1. Backlog Sprint 1	42
2. Expression des besoins	43
2.1. Diagramme de cas d'utilisation sprint 1.....	43
2.2. Description textuelle des cas d'utilisation de sprint 1	44
3. Conception	47
3.1. Diagramme de séquence détaillé	48
4. Configuration l'environnement de travail	50
4.2. Configuration du Firebase.....	52
5. Tests des fonctionnalités	54
5.1. Annotation de données	54
5.2. Test d'OpenCV, CVZone et Ultralytics.....	55
5.3. Exécution de la notebook d'entraînement sur Google Colab	56
5.4. Test du modèle sur nouvelle données.....	57
5.5. Test des fonctionnalités de l'application.....	58
6. Réalisation.....	59
Conclusion.....	61
Chapitre 4 : Sprint 2	63
Introduction	63
1. Backlog Sprint 2	63
2. Expression des besoins	66
2.1. Diagramme des cas d'utilisation de sprint 2	66
2.2. Description textuelle de cas d'utilisation sprint 2	67
3. Configuration de l'environnement de développement	71

3.1. Installation du Raspberry Pi OS avec Raspberry Pi Imager	71
3.2. Créer l'environnement virtuel sur la carte Raspberry pi.....	72
4. Diagramme de Bloc	74
5. Algorithme de programmation pour traitement d'image et reconnaissance de plaque.....	75
6. Algorithme de Programmation du Contrôle d'Accès au Parking.....	77
8. tests et Réalisation	81
Conclusion.....	84
Conclusion Générale et Perspectives	86
Webographie.....	88

Liste des figures

Figure 1 : Les solutions de sécurité électronique intégrés par KROCHU.....	14
Figure 2: organigramme de l'entreprise.....	14
Figure 3: Cycle de vie de la méthodologie Scrum.....	17
Figure 4: équipe de Scrum.....	17
Figure 5: Les différents rôles de Scrum et le déroulement du processus	19
Figure 6: UML	19
Figure 7: SysML	20
Figure 8: Diagramme de cas d'utilisation globale	25
Figure 9: Les différents types de Caméras	30
Figure 10: : Carte Raspberry pi 4 modèle B	31
Figure 11: Carte ESP32 DevKit	31
Figure 12: Caméra IP	33
Figure 13: Diagramme de Classe	39
Figure 14: Architecture système	40
Figure 15:Diagramme de déploiement	41
Figure 16 : Diagramme de cas d'utilisation sprint 1	44
Figure 17: Diagramme de séquence authentification.....	48
Figure 18: :Diagramme de séquence Inscription.....	49
Figure 19: Diagramme de séquence Ajout employé.....	50
Figure 20: opencv et cvzone.....	51
Figure 21: yolo et easyocr	52
Figure 22: labelImg	52
Figure 23: Realtime DB	53
Figure 24: firebase authentification.....	53
Figure 25: connexion Firebase a l'Android studio	53
Figure 26: . Installation du NodeMailer	54
Figure 27: l'interface graphique du labelImg.....	55
Figure 28: resultat d'annotation.....	55
Figure 29: résultat d'intégration	56
Figure 30: fichier data.....	57
Figure 31: exécution de la commande d'entraînement	57
Figure 32: génération de la modèle	57
Figure 33: Test du modèle	58
Figure 34: mail de réinitialisation	59
Figure 35: Accueil Dynamique, Authentification et Inscription	60
Figure 36: Réinitialisation, Mettre à jour employée et administrateur.....	60
Figure 37: Ajout employée, Voir liste des employée et administrateur.....	61
Figure 38: test final sprint 1	61
Figure 39: Diagramme de cas d'utilisation Sprint2.....	67
Figure 40: Installation du fois Raspberry Pi Imager	71
Figure 41: choix de modèle et OS.....	72
Figure 42: terminal de la Raspberry Pi	73
Figure 43: préférences de l'IDE Arduino.....	73
Figure 44: : Diagramme de Bloc	74
Figure 45: Organigramme d'Algorithme de Reconnaissance	76

Figure 46: Organigramme d'Algorithme de Contrôle	78
Figure 47: Organigramme Architecture d'intégration	81
Figure 48: Interface de détection de plaque à l'entrée du parking.....	82
Figure 49: Figure 49: Séquence de détection de la sortie du véhicule	83
Figure 50: Interface Tableau de bord et Historique	83
Figure 51: Prototype réel de contrôle de barrière	84
Figure 52: Structure finale de la base de données Firebase	84

Liste des tableaux

Tableau 1: Backlog Produc	27
Tableau 2: Planification des sprints.....	28
Tableau 3: Choix du microcontrôleur	29
Tableau 4: Caractéristique de la carte Raspberry	31
Tableau 5: Caractéristique de la carte ESP32 DEVKIT	32
Tableau 6: Périphériques électroniques	33
Tableau 7: Caractéristique de la caméra	34
Tableau 8: environnement matériel.....	34
Tableau 9: Environnement Logiciel.....	38
Tableau 10: Backlog Sprint 1	43
Tableau 11: Description textuelle de cas S'inscrire.....	45
Tableau 12: Description textuelle de cas Gérer son compte personne	45
Tableau 13: Description textuelle de cas Récupérer le mot de passe.....	46
Tableau 14: Description textuelle de cas Gérer plaques, employés autorisés et liste administrateurs.....	46
Tableau 15: Description textuelle de cas Associer plaques aux véhicules	46
Tableau 16: Description textuelle de cas S'authentifier.....	47
Tableau 17: Backlog Sprint 2	66
Tableau 18: Description textuelle de cas d'utilisation Détection et accès automatique avec affichage	68
Tableau 19: Description textuelle de cas d'utilisation Prendre une décision (autorisé / non autorisé) et suivre le nombre de places disponibles.....	69
Tableau 20: Description textuelle de cas d'utilisation Consulter l'historique des véhicules et Accéder au tableau de bord	70
Tableau 21: Description textuelle de cas d'utilisation Afficher un message via écran LCD	71
Tableau 22: Description de l' Organigramme d'Algorithme de Reconnaissance.....	77
Tableau 23: Description d'Algorithme de Contrôle	80

Introduction Générale

La sécurité constitue une priorité fondamentale pour toute organisation, visant à protéger les individus, les biens matériels et les données sensibles. Dans le contexte spécifique d'un parking d'entreprise, cette exigence prend une dimension critique : assurer la sûreté des lieux et des personnes, prévenir les intrusions, les actes de vandalisme, le vol de véhicules ou encore l'accès non autorisé. La mise en place de solutions de surveillance intelligentes devient alors un levier stratégique pour renforcer la gestion et la sécurité des infrastructures.

C'est dans ce cadre que s'inscrit notre projet de fin d'études. Grâce à notre formation à l'Institut Supérieur des Études Technologiques de Béja, nous avons eu l'opportunité de concrétiser nos compétences théoriques et pratiques à travers un projet professionnel mené en partenariat avec **Kromberg & Schubert**, une entreprise industrielle dont le site de Béja dispose d'un parking nécessitant une meilleure sécurisation et une gestion automatisée des accès.

L'objectif principal de notre travail a été de concevoir et d'implémenter une solution complète de surveillance intelligente basée sur la reconnaissance automatique des plaques d'immatriculation. En s'appuyant sur des technologies comme l'intelligence artificielle, le traitement d'image, l'embarqué et l'IoT.

Ce projet représente bien plus qu'un simple système de vidéosurveillance : il s'inscrit dans une démarche d'innovation technologique et de transformation digitale des entreprises industrielles, alliant sécurité, efficacité opérationnelle et amélioration de l'expérience utilisateur.

Le présent rapport est structuré comme suit :

- **Le premier chapitre** présente le **cadre général du projet**, à travers une description de l'entreprise d'accueil, une analyse critique de l'existant et les solutions proposées. Il introduit également la méthodologie Scrum adoptée et les outils de modélisation utilisés.
- **Le deuxième chapitre**, intitulé **Sprint 0**, est dédié à l'**étude conceptuelle** du projet : identification des besoins, acteurs, élaboration des premiers modèles UML, choix matériels et logiciels, et définition de l'architecture globale du système.

- **Le troisième chapitre** traite du **Sprint 1**, centré sur le développement des fonctionnalités de détection d'objets et de plaques via YOLO et EasyOCR, le développement de l'application mobile, ainsi que les premiers tests réalisés.
- **Le quatrième chapitre** est consacré au **Sprint 2**, où l'accent est mis sur l'implémentation embarquée sur Raspberry Pi et ESP32, l'intégration des composants matériels, la gestion des accès au parking et l'optimisation de l'algorithme.

Nous conclurons ce rapport par une synthèse des résultats obtenus, les limites rencontrées, ainsi que les perspectives d'évolution futures pour rendre le système encore plus intelligent et évolutif.

Chapitre 1 : Cadre général du projet

Introduction

Ce chapitre présente la société d'accueil **Kromberg & Schubert**, ses services ainsi que son engagement. Le sujet est ensuite replacé dans son cadre général à travers l'étude de l'existant, suivie d'une analyse critique permettant d'identifier les limites du système actuel. Cette analyse aboutit à la proposition d'une solution idéale visant à améliorer les performances de l'existant. Enfin, le choix de la méthodologie adoptée pour la réalisation du travail est exposé et le langage de modélisation des objets.

1. Présentation de l'entreprise d'accueil

1.1. Présentation générale

Kromberg & Schubert est une entreprise internationale d'origine allemande qui est spécialisée à la fabrication des câblages pour automobiles.

Kromberg & Schubert Béja : Établie en 2008, la filiale de Béja représente l'un des sites stratégiques du groupe. Actuellement, cette usine s'étend sur 35 000 mètres carrés et emploie 4 000 travailleurs. Dans une perspective d'expansion ambitieuse, l'entreprise a récemment acquis un terrain de 8 hectares, dont 4 200 mètres carrés seront consacrés à de nouvelles installations. Cette extension, prévue pour 2025, permettra la création de 4 000 emplois supplémentaires, portant la capacité totale à 8 000 employés et cadres d'ici 2026. [1]

1.2. Service de kromberg & Schubert

Kromberg & Schubert (KROSCHU) dispose d'une organisation structurée autour de dix services clés :

- Le **service qualité** dédie aux conformités des produits et la contrôle des matières premier, produit finis selon plusieurs certifications ISO.
- Le **Service Production** est divisé en Unités Automobiles de Production (UAP) spécialisées dans la coupe, la préparation et l'assemblage, géré par le système KROSY.
- Les **Ressources Humaines** assurent la formation des nouvelles recrues et la qualification du personnel, avec un système de badges indiquant les compétences acquises.
- Le **Service Informatique** maintient l'infrastructure technologique, gérant programmation, développement logiciel et résolution des problèmes techniques.

- L'**Engineering** planifie les nouveaux projets, élaborant des dossiers de consultation incluant calculs de temps et investissements nécessaires.
- La **Logistique** analyse les commandes clients et planifie la production, distinguant commandes fermes et prévisionnelles, tandis que l'Administration De Vente (ADV) assure le suivi des commandes.
- Les services **Finance, Achat, Maintenance et Bâtiments** complètent cette organisation en gérant respectivement les approvisionnements, les achats divers, l'entretien des machines et les infrastructures

1.3. Engagements de kromberg & Schubert

Kromberg & Schubert démontre son engagement de la bonne qualité de production à travers un processus de production rigoureusement structuré pour la fabrication de systèmes de câblage électrique automobile. L'entreprise maintient un flux de production optimal grâce à une organisation stratégique en zones spécialisées : la zone de coupe (centre névralgique desservant toutes les UAP), la zone de préparation (pour traitements spécifiques), la zone de montage (où s'effectue l'assemblage sur planches mobiles ou fixes), la zone de contrôle électrique (vérification des normes techniques) et la zone de contrôle final (inspection visuelle avant conditionnement). Cette méthodologie assure l'efficacité et la satisfaction client et garantissant que chaque câblage livré répond aux standards les plus exigeants de l'industrie automobile internationale.

1.4. Distributeur Et Intégrateur De Solution De sécurité électrique

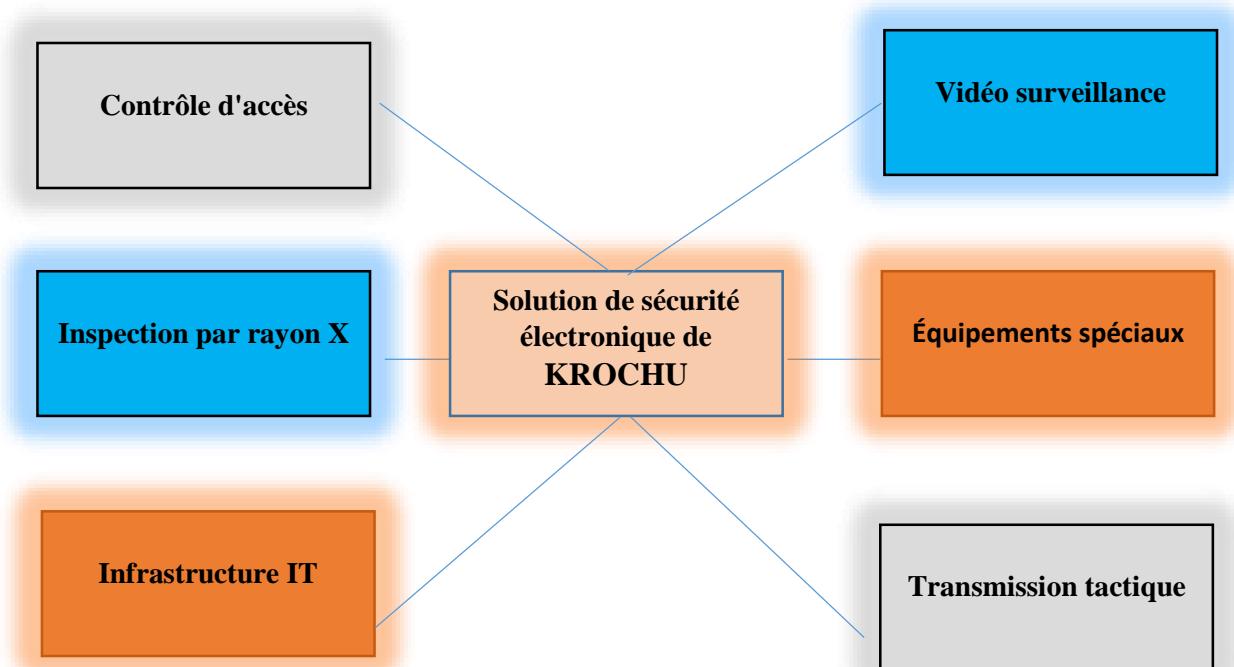


Figure 1 : Les solutions de sécurité électronique intégrés par KROCHU

1.5. Organigramme de l'entreprise

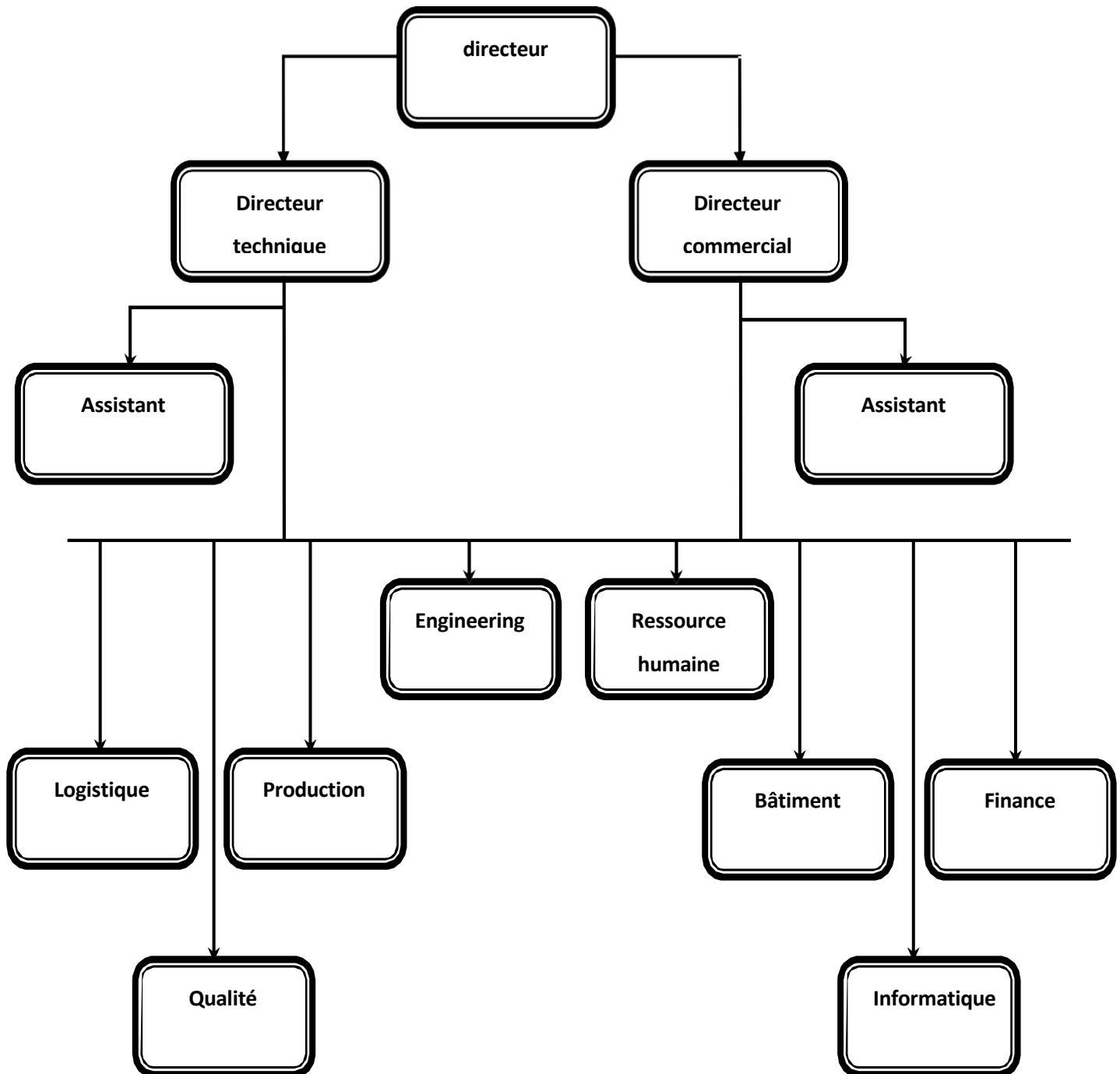


Figure 2: organigramme de l'entreprise

2. Analyse et étude de l'existant

2.1. Etude de l'existant

L'analyse de l'existant nous permet de mieux comprendre la situation et identifier les points forces et faiblesse d'un produit déjà existent qu'ils sont interdépendants et essentiels pour la réussite d'un projet. Afin de mieux comprendre et définir clairement les besoins du client. Cette étude permettra de collecter toutes les informations relatives à l'architecture actuelle et de prendre en compte ces éléments lors de la conception et de la réalisation de notre projet afin de proposer une solution qui répond spécifiquement aux besoins identifiés

2.2. Description de l'existant

Actuellement, la gestion du parking de l'entreprise Kromberg & Schubert Béja est entièrement manuelle, sans aucun système numérique ou automatisé. Les véhicules y accèdent librement selon l'ordre d'arrivée, sans contrôle ni suivi en temps réel. Cette approche provoque une saturation fréquente des places disponibles, en particulier en période de forte affluence, ce qui perturbe l'organisation globale de l'entreprise.

Les employés sont souvent contraints de stationner devant les bâtiments ou dans des zones non prévues à cet effet, ce qui complique la circulation interne et nuit à la fluidité des accès. De plus, l'absence d'identification ou de vérification à l'entrée permet à des véhicules non autorisés d'accéder au site, exposant l'entreprise à des risques de sécurité. Cette gestion non régulée génère également un sentiment d'injustice entre les employés et reflète un besoin urgent de modernisation à travers une solution intelligente et automatisée.

2.3. Critique de l'existant

La gestion actuelle du parking au sein de l'entreprise Kromberg & Schubert Béja présente plusieurs lacunes majeures qui soulignent l'urgence de la modernisation du système :

- Absence de contrôle et de suivi numérique, la gestion totalement manuelle du stationnement ne permet ni traçabilité des véhicules, ni supervision en temps réel. Cela limite gravement la capacité de l'entreprise à gérer efficacement les ressources disponibles et à anticiper les besoins.
- Manque de sécurité, l'absence d'un système de contrôle d'accès permet à des véhicules non autorisés d'entrer sur le site. Cette faille de sécurité représente un risque sérieux, notamment en matière de protection des biens et des personnes.

- Retard technologique, à une époque où l'automatisation et la numérisation sont devenues des standards, la persistance d'un système manuel renvoie une image de retard technologique. Cela peut affecter la perception externe de l'entreprise et limiter son attractivité pour les talents ou les partenaires.

Cette analyse critique met en évidence les impacts organisationnels, sécuritaires et sociaux de la gestion actuelle. Elle justifie pleinement la mise en œuvre d'une solution intelligente et automatisée de gestion de parking.

2.4. Solution proposée

Notre proposition de projet de stationnement à **Kromberg & Schubert Béja** pour répondre aux besoins de surveillance et de gestion du parking de l'entreprise est de concevoir un système capable de reconnaître les plaques d'immatriculation des voitures par l'implémentation d'un module intelligent intégré aux caméras de surveillance existantes qui sera conçu pour ajouter une couche d'intelligence en utilisant des techniques avancées. L'objectif principal est de développer une solution capable de détecter et d'analyser en temps réel les plaques d'immatriculation des véhicules entrant dans le parking à partir du flux vidéo des caméras de surveillance. Grâce à cela, il est possible de vérifier automatiquement si un véhicule est autorisé à entrer ou non, en tenant compte des places disponibles. Les informations sont ensuite envoyées avec une plateforme en ligne pour assurer un suivi. Le responsable peut également gérer les données, consulter l'état des lieux à partir d'une application sur téléphone. Le système agit ensuite pour autoriser ou refuser l'accès, en déclenchant certains éléments comme l'ouverture d'une barrière, l'allumage d'un voyant ou la diffusion d'un signal sonore.

3.Choix méthodologique

3.1. Méthodologie de conception : Scrum

Scrum est une méthodologie agile de gestion de projet dont le choix répond parfaitement à nos exigences. Elle facilite la collaboration entre les membres de l'équipe et permet une adaptation rapide aux changements survenant au cours du développement. Cette approche favorise la livraison de produits en amélioration continue, créant ainsi une valeur ajoutée constante tout en assurant la satisfaction des utilisateurs finaux. Grâce à sa structure itérative et ses principes de transparence, Scrum nous permet d'optimiser notre processus de conception et d'atteindre efficacement nos objectifs de qualité et de délais.

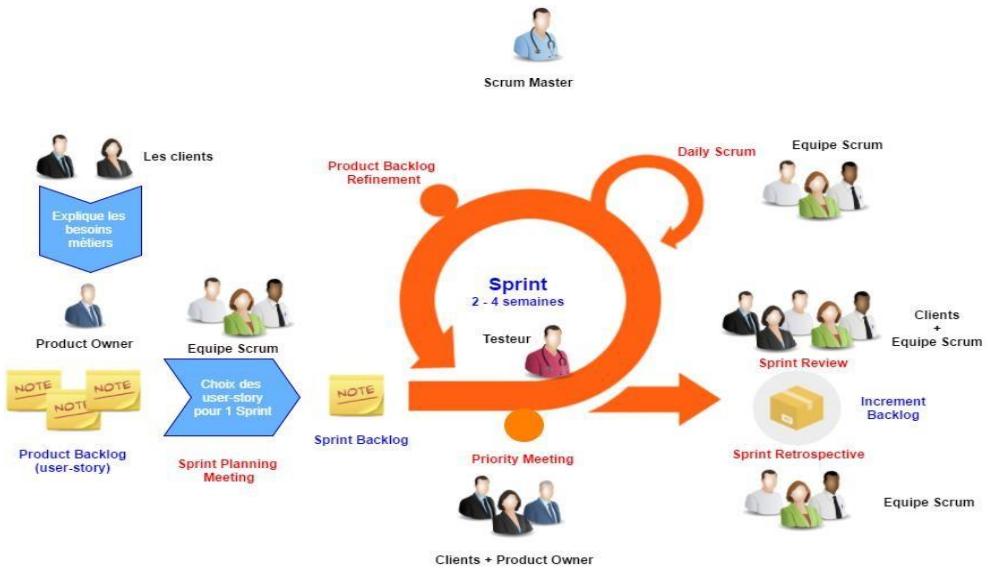


Figure 3: Cycle de vie de la méthodologie Scrum

3.2. Le rôle de Scrum

Scrum présente trois rôles : Product Owner, Scrum Master et les membres de l'équipe de développement.

3.2.1. L'équipe de développement

L'équipe de développement rassemble les personnes qui réalisent le travail. À première vue, vous pouvez penser que l'équipe de développement est composée d'ingénieurs. Mais ce n'est pas toujours le cas. Selon le Guide Scrum, l'équipe de développement peut comprendre toutes sortes de personnes, y compris des designers, des rédacteurs, des programmeurs, et bien d'autres. [2]

Généralement, ces personnes possèdent toutes les compétences nécessaires pour livrer un produit potentiellement utilisable.

La figure 4 présente l'équipe scrum :

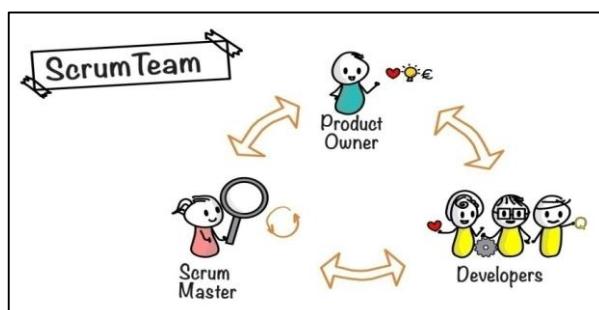


Figure 4: équipe de Scrum

Product Owner : Responsable de maximiser la valeur du produit, il définit les priorités du backlog, (besoins d'utilisateurs) clarifie les exigences et représente les intérêts des parties prenantes. Il possède une vision claire du produit final et prend les décisions concernant les fonctionnalités à développer

Scrum Master : Facilitateur et coach qui aide l'équipe à comprendre et appliquer les principes Scrum. Guider l'équipe pour résoudre les problèmes, protège l'équipe des interférences externes et favorise un environnement propice à l'auto-organisation, assure la bonne communication entre les membres. Il n'est pas un manager traditionnel mais plutôt un servant-leader.

3.3. Concept Scrum

Les concepts de la méthode Scrum sont les suivants :

- ✓ **Création du Product Backlog :** le Product Owner prépare une liste exhaustive des fonctionnalités, améliorations et corrections à apporter au produit. Cette liste évolutive représente la référence du projet.
- ✓ **Sprint Planning :** l'équipe Scrum organise une réunion pour planifier le sprint (d'une durée variant de 2 à 4 semaines), en sélectionnant les éléments du Product Backlog à intégrer dans le Sprint Backlog, selon leur priorité.
- ✓ **Daily Scrum :** c'est une réunion quotidienne de 15 minutes maximum où chaque membre de l'équipe répond aux questions suivantes :

Qu'avez-vous fait hier ?

Que ferez-vous aujourd'hui ?

Le Scrum Master discute rapidement des problèmes qui bloquent les membres de l'équipe.

- ✓ **Exécution du Sprint :** c'est la phase de développement pendant laquelle l'équipe travaille sur les éléments du Sprint Backlog afin de livrer un incrément du produit.
- ✓ **Sprint Review :** à la fin du sprint, l'équipe organise une réunion pour présenter l'incrément réalisé aux parties prenantes et ajuster le Product Backlog en cas de nouveaux besoins ou changements.

- ✓ **Sprint Rétrospective** : L'équipe identifie les points d'amélioration pour le sprint suivant afin d'éviter les erreurs précédentes et d'optimiser sa performance.

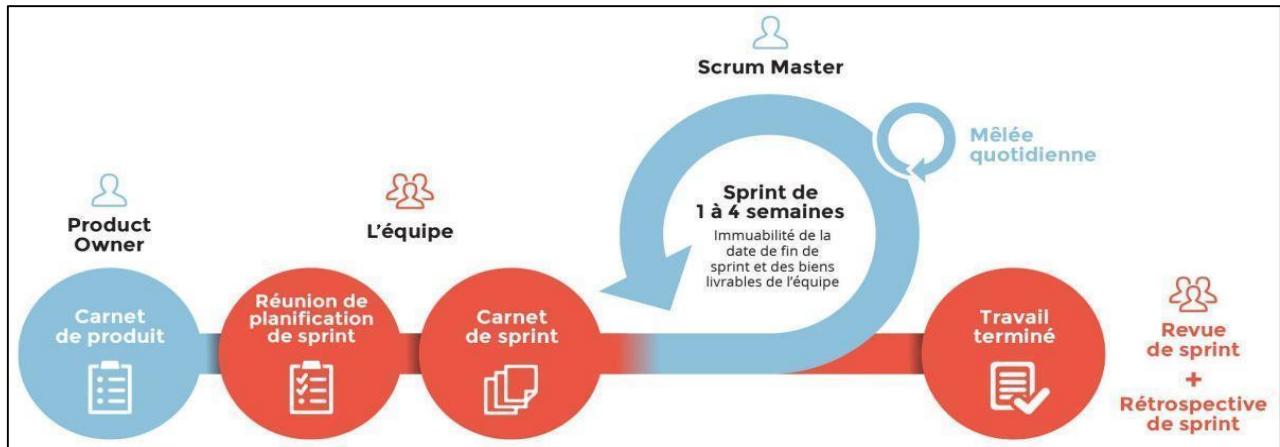


Figure 5: Les différents rôles de Scrum et le déroulement du processus

4. Langage de modélisation UML

Le langage UML (Unified Modeling Language, ou visuel standardisé utilisé en génie logiciel pour **modéliser, spécifier, visualiser et documenter** les aspects d'un système logiciel, et riche sémantiquement et syntaxiquement. Il est destiné à l'architecture, la conception et la mise en œuvre de systèmes logiciels complexes par leur structure aussi bien que leur comportement. L'UML a des applications qui vont au-delà du développement logiciel, notamment pour les flux de processus dans l'industrie.

Il ressemble aux plans utilisés dans d'autres domaines et se compose de différents types de diagrammes. Dans l'ensemble, les diagrammes UML décrivent la limite, la structure et le comportement du système et des objets qui s'y trouvent.

L'UML n'est pas un langage de programmation, mais il existe des outils qui peuvent être utilisés pour générer du code en plusieurs langages à partir de diagrammes UML.

L'UML a une relation directe avec l'analyse et la conception orientée objet. [3]



Figure 6: UML

4.1. Langage de modélisation SYSML

SysML est le nouveau langage de modélisation défini par l'OMG. Il peut être vu comme une extension d'UML destinée à la modélisation d'un large spectre de systèmes complexes, incluant les systèmes automobiles. [4] tels que les systèmes automobiles.

Dans le contexte de la modélisation d'un système automobile, SysML présente plusieurs avantages par rapport à UML

- ✓ SysML permet une capture exhaustive des exigences, contrairement à UML qui se concentre généralement sur les diagrammes de cas d'utilisation.
- ✓ SysML permet une modélisation précise des performances et des contraintes mécaniques grâce à ses diagrammes paramétriques, contrairement à UML.
- ✓ Avec SysML, la logique de contrôle et du logiciel embarqué est spécifiée de manière détaillée à l'aide de versions étendues des diagrammes d'activité et d'états d'UML. Cela offre une représentation précise de la logique de fonctionnement dans les systèmes embarqués complexes.
- ✓ SysML permet une modélisation complète du système automobile, incluant les aspects physiques, logiciels et organisationnels, grâce à ses nombreux diagrammes structurels et comportementaux

En résumé, SysML offre une approche plus complète et adaptée à la modélisation des systèmes complexes tels que les automobiles, en permettant une capture détaillée des exigences, des performances et des contraintes mécaniques, ainsi qu'une spécification précise de la logique de contrôle et du logiciel embarqué



Figure 7: SysML

Conclusion

Au cours de ce chapitre, **kromberg & Schubert**, l'organisme d'accueil, a été présenté, suivi d'une analyse approfondie de l'existant permettant d'identifier les problèmes et les enjeux du projet.

Une solution adaptée a ensuite été proposée. Aussi , la méthodologie **SCRUM** ainsi que le langage de modélisation **SysML**, choisis pour la réalisation du projet, ont été introduits. Ces éléments posent les bases nécessaires pour aborder le deuxième chapitre, consacré à la préparation des fondations de projet.

Chapitre 2 : Etude conceptuelle : Sprint 0

Introduction

Dans ce chapitre, une analyse des besoins fonctionnels et non fonctionnels associés au projet sera présentée. Ensuite, la méthode à adopter afin de modéliser les besoins auxquels doit répondre la solution sera déterminée, et finalement, le chapitre se clôturera avec la conception des besoins et une conclusion.

1. Analyse des besoins

1.1. Identification des acteurs

Un acteur représente une entité qui interagit avec le système. Elle peut être une personne ou un autre système informatique ou ensemble dispositif interconnectées échangeant les informations et faire des réactions entre eux.

Les acteurs principaux sont :

- ✓ Administrateur
- ✓ Conducteur
- ✓ Système intelligent de détection(SID)
- ✓ Système de contrôle barrière (SCB)

1.2. Besoins fonctionnels

Un besoin fonctionnel est un besoin spécifiant une action qu'un système doit être capable d'effectuer, sans considérer à aucune contrainte physique. C'est un besoin du point de vue de l'utilisateur. Les besoins fonctionnels sont focalisés sur le métier des utilisateurs. La définition des besoins fonctionnels consiste à définir les points précis du cahier des charges et le besoin primaire du client. Ces besoins conduisent à l'élaboration des modèles de cas d'utilisation car ils définissent ce que le système doit faire d'où les fonctions du système à développer. [5]

En tant qu'administrateur, je peux

- ✓ S'inscrire et s'authentifier
- ✓ Gérer son compte personnel (profil, mot de passe)
- ✓ Récupérer son mot de passe
- ✓ Ajouter, modifier ou supprimer :
 - Les plaques d'immatriculation autorisées

- La liste des employés autorisés
- ✓ Gérer les plaques d'immatriculation associées aux véhicules
- ✓ Consulter :
 - Les problèmes signalés
 - L'historique des véhicules (entrées, sorties, horaires)
 - Le flux vidéo en temps réel
 - Les informations de son propre compte
 - La liste des autres administrateurs
- ✓ Accéder au **tableau de bord pour voir :**
 - Le nombre de places disponibles
 - La dernière plaque détectée
 - Les derniers employés détectés
 - La liste des plaques détectées aujourd'hui avec date d'entrée et de sortie

En tant que conducteur je peux:

- ✓ S'inscrire dans le système.

En tant que SID je peux

- ✓ Déetecter les plaques d'immatriculation (à l'aide du modèle)
- ✓ Capturer de flux vidéo en temps réel.
- ✓ Faire le traitement de flux vidéo en temps réel
- ✓ Communiquer avec une base de données.
- ✓ Gérer des décisions (autorisé/non autorisé),
- ✓ Suivi des places disponibles.

En tant que SCB, je peux :

- ✓ Afficher les messages personnalisés et le nombre de places disponibles.
- ✓ Fermer ou Ouvrir la barrière selon le cas (entrée sortie)
- ✓ Allumer la LED verte ou rouge selon l'autorisation.
- ✓ Activation de buzzer selon l'autorisation

1.3. Besoins non fonctionnels

Les besoins non fonctionnels sont des exigences qui ne sont pas liées directement aux fonctionnalités du système, mais qui ont une incidence sur la qualité et les performances du système. Il est important de dégager ces besoins sa fin de garantir une meilleure qualité du produit. Dans le cas de notre solution, les besoins non fonctionnels que nous avons distingués sont :

- ✓ Sécurité : la sécurité est un aspect crucial pour tout parking. Nos applications doivent assurer la sécurité et la confidentialité des données de tous les utilisateurs qui doivent être stockées de manière sécurisée avec Authentification obligatoire pour accéder aux fonctions admin.
- ✓ Respect de l'environnement : le parking doit être conçu de manière à minimiser l'impact sur l'environnement. Cela peut inclure l'utilisation de panneaux solaires pour l'éclairage ou la récupération des eaux de pluie pour l'arrosage des plantes.
- ✓ Simplicité : Notre solution doit être non-compliquée afin de simplifier son utilisation par le citoyen.
- ✓ Performance : Le temps de réponse ne doit pas dépasser quelques secondes pour éviter les problèmes d'insatisfaction de l'utilisateur.
- ✓ Robustesse : L'application doit être fonctionnelle indépendamment de toutes les circonstances (panne, problème de mise à jour...) relative à l'utilisateur.

1.4. Diagramme de cas d'utilisation globale

Le diagramme Use-Case, appelé diagramme de cas d'utilisation en français, fait partie des diagrammes de comportement du langage Unified Modelling Language, UML en abrégé, avec les systèmes et processus de programmation objet ou encore les processus métier. UML n'est donc pas un langage de programmation, mais un langage de modélisation. C'est une méthode standardisée qui décrit un système en cours de conception, ou déjà existant. Cela se fait à l'aide de diagrammes, dans lesquels tous les objets impliqués sont structurés et liés les uns aux autres. De manière générale, un diagramme de cas d'utilisation permet de modéliser le comportement d'un système suite à l'interaction d'un acteur externe. Ainsi, nous allons présenter un

diagramme de cas d'utilisation global qui mettra en évidence les principaux services offerts par l'application pour permettre aux acteurs d'interagir avec celle-ci. [6]

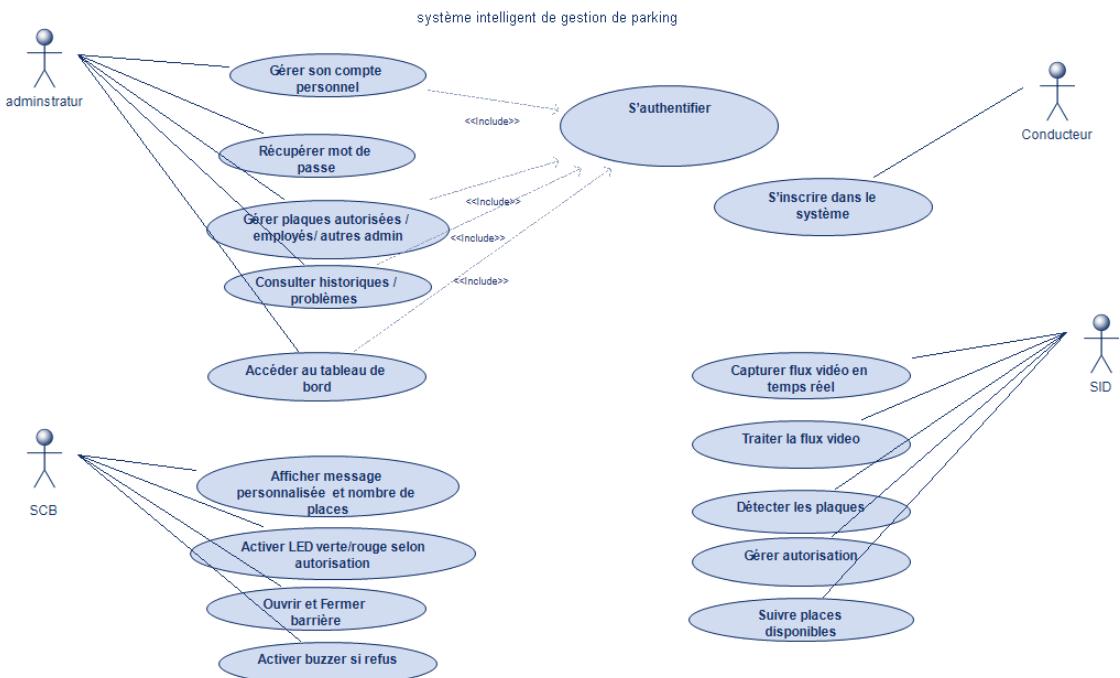


Figure 8: Diagramme de cas d'utilisation globale

2. Pilotage du projet avec SCRUM

Dans cette section on va présenter l'équipe Scrum et son rôle clairement défini, ainsi que sur le Backlog produit et la planification des sprints.

2.1. Equipes et rôles

La réussite d'un projet Scrum repose sur une définition précise des rôles et responsabilités. L'identification claire des membres occupant chaque fonction constitue un prérequis essentiel à la fluidité des processus et à l'efficacité de la méthodologie, l'équipe de ce projet est constituée de :

- Product Owner: M. Mohamed Dhia Houimli
- Scrum Master : M. Seif Ben Chaabene
- Development team: Mekni Nahawand /Aridhi Rihem

2.2. Backlog Product

	ID	User story	Priorité	Estimation	sprint
Administrateur	1	En tant qu'administrateur, je peux m'inscrire et m'authentifier	Elevé	4jrs	Sprint1
Administrateur	2	En tant qu'administrateur, je peux gérer mon compte (profil, mot de passe)	Moyenne	1jrs	Sprint1
Administrateur	3	En tant qu'administrateur je dois pouvoir gérer les employés , les autres administrateur et les plaques d'immatriculation autorisés	Elevé	7jrs	Sprint1
Administrateur	4	En tant qu'administrateur, je peux récupérer mon mot de passe	Elevé	1jrs	Sprint1
SID	5	En tant que caméra, je peux capturer le flux vidéo en temps réel	Elevé	4jrs	Sprint1
SID	6	En tant que système intelligent de détection, je peux détecter les plaques	Elevé	30jrs	Sprint1
SID	7	En tant que système intelligent de détection, je peux traiter le flux vidéo	Elevé	5jrs	Sprint1
SID	8	En tant que conducteur, je peux être détecté automatiquement à l'entrée/sortie	Elevé	4jrs	Sprint2
SID	9	En tant que conducteur, je peux accéder si ma plaque est autorisée	moyenne	1jrs	Sprint2

SID	10	En tant que système intelligent , je peux prendre une décision (autorisé / non autorisé)	Elevé	1h	Sprint2
SID	11	En tant que système intelligent, je peux suivre le nombre de places disponibles	faible	1h	Sprint2
Administrateur	12	En tant qu'administrateur, je peux accéder au tableau de bord	Elevé	3jrs	Sprint2
Administrateur	13	En tant qu'administrateur, je peux consulter l'historique des véhicules	moyenne	1jrs	Sprint2
SCB	14	En tant que système de contrôle barrière, je peux ouvrir/fermer la barrière	Elevé	2jrs	Sprint2
SCB	15	En tant que système de contrôle barrière, je peux afficher des messages personnalisé via LCD	faible	1jrs	Sprint2
SCB	16	En tant que système de contrôle barrière, je peux activer LED verte/rouge et buzzer	moyenne	2h	Sprint2

Tableau 1: Backlog Produc

2.3. Planification des sprints

	Fonctionnalités	Période
Sprint 1	<ul style="list-style-type: none"> • Entraînement du modèle YOLOv8n • Capturer le flux vidéo en temps réel • Traiter des flux vidéo • Déetecter les plaques • Intégration OCR • S'authentifier 	De 15/02/2025 A 31/03/2025

	<ul style="list-style-type: none"> • S'inscrire • Gestion du compte personnel • Récupération du mot de passe • Tableau de bord • Gestion des plaques d'immatriculation, d'employées et administrateurs 	
Sprint 2	<ul style="list-style-type: none"> • Développement système principal • Développement de système de contrôle physique • Déploiement du programme principal sur la carte Raspberry pi • Tests d'intégration complète en temps réel • Accéder au tableau de bord en temps réel • Consulter l'historique des véhicules 	De 01/04/2025 A 23/05/2025

Tableau 2:Planification des sprints

2. Etude et choix électronique

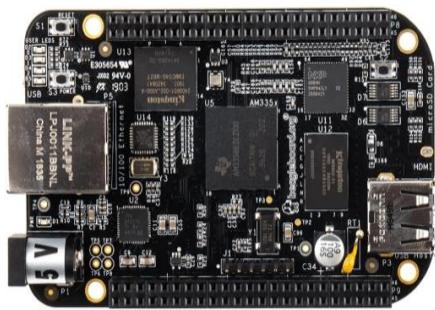
2.1. Choix du microcontrôleur

Dans le cadre de notre projet de développement d'un système de reconnaissance de plaques d'immatriculation le choix du microcontrôleur est une étape déterminante et importantes, nous avons effectué une étude approfondie pour sélectionner notre choix électronique. Parmi les cartes électroniques les plus utilisées sur le marché, on trouve :

 Carte ESP32	Après l'examen de plusieurs cartes électroniques, dont le Raspberry Pi 4, le BeagleBone Black, le Jetson Nano et l'ESP32, en tenant compte de leurs caractéristiques techniques, de leur disponibilité, de leurs coûts, nous avons conclu que le Raspberry Pi 4
---	---



Carte Arduino



Carte BeagleBone Black



Carte Jetson Nano



Carte Raspberry pi

répondait le mieux à nos critères. Sa puissance de calcul, sa connectivité, sa taille compacte, son coût abordable et le vaste support de sa communauté en font un choix idéal. Sa capacité à traiter efficacement le flux vidéo en temps réel, à exécuter des modèles de détection comme YOLO et à s'intégrer facilement aux caméras de surveillance existantes en fait la solution optimale pour notre projet. En complément, nous avons choisi d'utiliser l'ESP32 comme microcontrôleur secondaire pour gérer les éléments physiques du système, tels que l'écran LCD, les LED, la barrière automatique et les signaux de commande. Grâce à sa faible consommation d'énergie, sa connectivité Wi-Fi intégrée et sa compatibilité, l'ESP32 peut recevoir en temps réel les données transmises depuis le Raspberry Pi via la base de données cloud, assurant ainsi une interaction rapide et fiable avec les dispositifs matériels sur le terrain.

Tableau 3: Choix du microcontrôleur

2.2. Choix du capteur (Caméra)

Dans le cadre de notre projet, nous avons évalué plusieurs options de caméras, notamment les caméras IP, les caméras analogiques et les caméras de vidéosurveillance classiques.



Figure 9: Les différents types de Caméras

2.3. Différence entre Caméra analogique et Caméra IP

- Caméra IP : Offre une connectivité réseau avancée pour un accès à distance et une intégration facile avec d'autres systèmes de surveillance. Elle fournit généralement une résolution élevée, idéale pour une reconnaissance précise des plaques d'immatriculation. Cependant, son coût initial peut être plus élevé que les autres types de caméras.
- Caméra analogique : Propose un coût initial généralement moins élevé et une installation simplifiée dans les systèmes de vidéosurveillance existants. Toutefois, sa résolution plus faible peut compromettre la qualité d'image et la précision de la reconnaissance des plaques d'immatriculation. De plus, elle offre moins de fonctionnalités avancées que les caméras IP.

Par conséquent, nous avons opté pour une caméra IP, car elle répond mieux à nos exigences en termes de qualité d'image, de compatibilité avec le traitement IA embarqué sur le Raspberry Pi, et de facilité d'intégration dans un système connecté en temps réel.

3. Environnement matériel

3.1. Carte Raspberry Pi



Figure 10: : Carte Raspberry pi 4 modèle B

Processeur	Broadcom BCM2711, SoC 64 bits quad-core Cortex-A72 (ARM v8) à 1,5 GHz
Mémoire	LPDDR4 de 2 Go
GPIO	GPIO standard à 40 broches
Connectivité	Réseau local sans fil IEEE 802.11b / g / n / ac de 2,4 GHz et 5,0 GHz Bluetooth 5.0, BLE Gigabit Ethernet 2 ports USB 3.0 2 ports USB 2.0
Vidéo et son	2 × ports micro HDMI (jusqu'à 4Kp60 pris en charge) 2-voix MIPI DSI display port 2-voix MIPI CSI camera port Connecteur 4-contact Audio stéréo et Vidéo composite
Multimédia	H.265 (décodage 4Kp60) ; H.264 (décodage 1080p60, encodage 1080p30) ; OpenGL ES, 3.0 graphiques
Alimentation	5V DC via un connecteur USB-C (minimum 3A) 5V DC via le connecteur GPIO (minimum 3A) Alimentation par Ethernet (PoE) – besoin de la carte d'extension PoE HAT en option
Environnement	Température de fonctionnement 0–50°C

Tableau 4: Caractéristique de la carte Raspberry

3.2. Carte ESP32 DevKitC



Figure 11: Carte ESP32 DevKit

Processeur	Dual-core Tensilica LX6, jusqu'à 240 MHz
Mémoire RAM	520 Ko SRAM intégrée
Mémoire Flash	16 Mo de mémoire Flash externe (souvent 4 Mo par défaut)

GPIO	34 broches GPIO, compatibles avec ADC, PWM, UART, I2C, SPI
Connectivité	Wi-Fi 802.11 b/g/n, Bluetooth v4.2 BR/EDR et BLE
Interfaces	3x UART, 3x SPI, 2x I2C, 2x DAC, 18x ADC 12-bit, PWM, CapSense, SDIO
Alimentation	3.3V (via régulateur à partir de 5V micro-USB)
Environnement	-40°C à +125°C
USB	Micro-USB pour alimentation et programmation
Tensions des GPIO	3.3V TTL (non tolérant au 5V sans adaptation)
Dimensions	Environ 51 mm x 25 mm

Tableau 5: Caractéristique de la carte ESP32 DEVKIT

3.3. Périphériques électroniques utilisés

Périphériques électroniques	Description
 LCD i2C 16x2	Un écran LCD est un dispositif d'affichage qui utilise des cristaux liquides pour afficher des caractères, chiffres ou graphiques. Il existe plusieurs types d'écrans, comme le LCD 16x2 (16 caractères sur 2 lignes) [7]

	Les LEDs sont des diodes électroluminescentes utilisées comme indicateurs lumineux. Elles consomment peu d'énergie et s'allument selon la polarité.[8]
	Un servomoteur est un moteur contrôlable en position, souvent utilisé pour le mouvement angulaire précis. Il contient un moteur, un système de réduction et un circuit de contrôle. [9]
	Le buzzer est un petit composant électromécanique ou piézoélectrique qui produit un son ou un bip lorsqu'il est activé [10]

Tableau 6: Périphériques électroniques

3.4. Caméra Dahua



Figure 12: Caméra IP

Standard	TVP/IP
Résolution	1920 x 1080 - 1080p 1280 x 1024 - 1.4 Mpx 1280 x 960 - 1.3 Mpx 1280 x 720 - 720p
Angle de vue	99 ° ... 37 ° (données du fabricant) 91 ° ... 34 ° (nos tests)
Prise de carte mémoire :	Prise en charge des cartes Micro SD jusqu'à 128GB (enregistrement local possible)
Adresse IP par défaut :	192.168.1.108

WEB Server	Intégré
Alimentation	PoE (802.3af), 12 V DC / 810 mA

Tableau 7: Caractéristique de la caméra

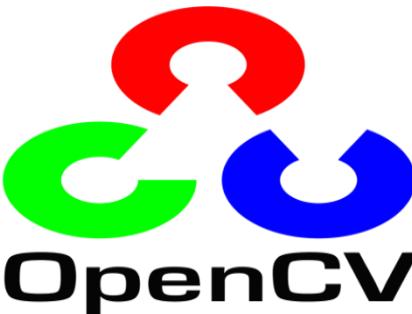
Lors du développement de notre projet, nous avons utilisé des ordinateurs personnels présentant les caractéristiques ci-dessous :

Lenovo IdeaPad	 <small>Kelaptop.com</small>	<ul style="list-style-type: none"> • Processeur : Intel Core i7 • Mémoire RAM : 16 Go • Disque dure : 512 GO SSD • OS : Windows 11
DELL Inspiron		<ul style="list-style-type: none"> • Processeur : Intel Core i5 • Mémoire RAM : 12 Go • Disque dure1 : 516 GO SSD • Disque dure2 : 1TO HDD • OS : Windows 10

Tableau 8: environnement matériel

4. Environnement logiciel

Après une analyse minutieuse et une comparaison entre divers logiciels disponibles, nous avons sélectionné celui qui correspondait le mieux à nos exigences pour notre environnement de travail logiciel.

Logiciel	Description
	OpenCV est une bibliothèque open source utilisée pour le traitement d'images et la vision par ordinateur. Elle propose des fonctionnalités telles que le traitement d'images en temps réel, la détection et la reconnaissance d'objets, avec une interface conviviale pour plusieurs langages de programmation. [11]



Android Studio est l'IDE officiel de Google pour le développement d'applications Android. Il offre des fonctionnalités telles que l'édition de code, le débogage, la conception d'interfaces utilisateur et la compilation .[12]



Python est un langage de programmation interprété, polyvalent et facile à apprendre, utilisé ici pour le traitement d'images, la détection d'objets et la communication avec les bases des données. Il est très utilisé dans les domaines de l'IA, l'IoT et le développement web. [13]



YOLO (You Only Look Once) est un modèle de détection d'objets en temps réel qui détecte et localise plusieurs objets dans une image ou une vidéo.[14]



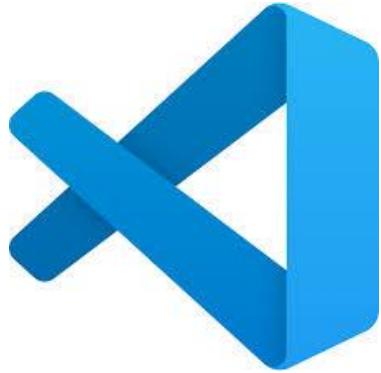
EasyOCR est une bibliothèque Python qui permet de faire de la reconnaissance optique de caractères (OCR), c'est-à-dire extraire du texte à partir d'images

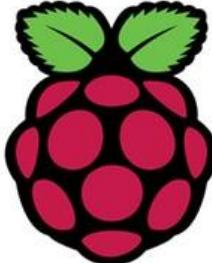
[15]



Firebase est une plateforme de développement mobile et web créée par Google. Elle fournit plusieurs services prêts à l'emploi pour créer des applications modernes, sans avoir besoin de tout coder soi-même côté serveur.

[16]

	<p>VS Code est un éditeur de code léger et puissant utilisé pour développer, exécuter et déboguer le script Python du système de reconnaissance des plaques. Il supporte de nombreuses extensions utiles.</p> <p>[17]</p>
	<p>Cvzone est une bibliothèque Python qui simplifie l'utilisation d'Open CV, en proposant des fonctions de dessin d'interface graphique comme les rectangles arrondis, les textes, etc.</p> <p>[18]</p>
 LabelImg	<p>LabelImg est une application graphique Python (PyQt5) permettant de créer des annotations pour les images, utilisées dans l'entraînement de modèles de détection d'objets. Elle permet de dessiner des boîtes englobantes (bounding boxes) autour des objets présents dans les images et de les étiqueter avec le nom de leur classe.</p>
	<p>NodeMailer est une bibliothèque Node.js qui permet d'envoyer des e-mails facilement à partir de serveurs ou d'applications.[19]</p>

	<p>Google Colab (ou Colaboratory) est un environnement de développement gratuit en ligne fourni par Google, basé sur Jupyter Notebook.</p> <p>Il permet d'écrire et d'exécuter du code Python directement dans le navigateur [20]</p>
	<p>L'Arduino IDE (Integrated Development Environment) est un logiciel libre qui permet d'écrire, compiler et téléverser du code sur des cartes Arduino (Uno, Mega, Nano...) [21]</p>
	<p>Raspberry Pi Imager est un outil officiel permettant d'installer facilement un système d'exploitation sur une carte microSD</p>
	<p>VNC Viewer est un logiciel qui permet de contrôler à distance un Raspberry Pi via une interface graphique. Il facilite l'accès à l'écran du Raspberry Pi depuis un autre appareil connecté au même réseau.</p>
	<p>Modelio est un outil de modélisation UML/BPMN open source qui permet de concevoir et de documenter des systèmes logiciels. Il est souvent utilisé dans les projets informatiques [22]</p>

	<p>Canva est un outil de design et de publication en ligne dont la mission est de permettre à tout le monde de créer et de publier sans limites</p>
	<p>Le langage C, est un langage de programmation impératif et polyvalent, créé à l'origine par Dennis Ritchie aux Bell Téléphone Laboratoires. C'est un langage de bas niveau qui offre un contrôle précis sur la mémoire et le matériel, ce qui en fait un choix populaire pour les systèmes d'exploitation, les applications embarquées, et les programmes de calcul scientifique.</p>
	<p>Thonny est un langage de programmation, mais un environnement de développement intégré (IDE) pour le langage Python. Il est conçu pour les débutants en programmation et facilite l'écriture, l'exécution et le débogage de code Python.</p>

Tableau 9: Environnement Logiciel

5. Diagramme de classe

Le diagramme de classes est un type de diagramme utilisé en UML (Unified Modeling Language) pour modéliser la structure statique d'un système logiciel. C'est un outil essentiel pour la conception orientée objet, permettant de visualiser comment les objets interagissent et s'organisent dans le système. [23]

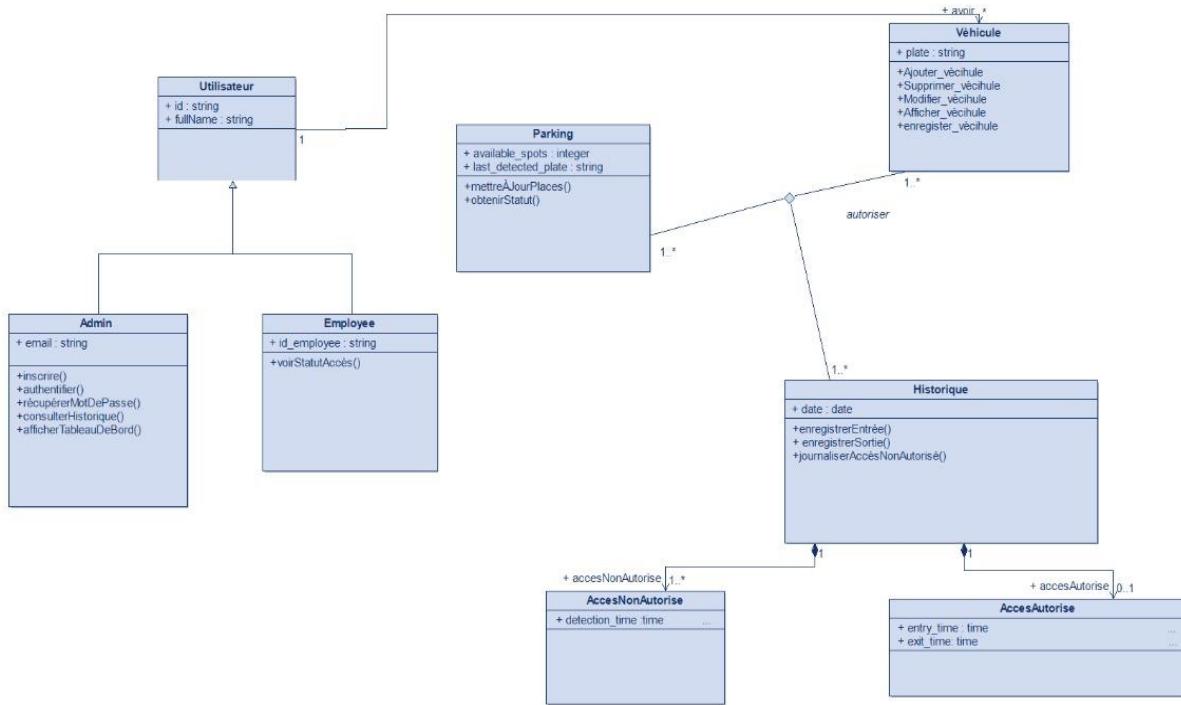


Figure 13: Diagramme de Classe

6. Architecture système

Le système de gestion de parking intelligent repose sur une architecture distribuée, moderne et évolutive. Il est conçu autour de composants matériels (caméra, Raspberry Pi 4, ESP32) interconnectés via une base de données cloud Firebase, ainsi qu'une application mobile dédiée aux administrateurs.

L'application mobile est développée à l'aide du framework Android Studio et communique avec Firebase à l'aide du Firebase SDK, permettant une interaction en temps réel. Grâce à cette plateforme, l'administrateur peut récupérer, créer, modifier et supprimer les données liées au système (plaques d'immatriculation autorisées, employés, historiques ...).

Le cœur de traitement local est assuré par le Raspberry Pi 4, qui agit comme une passerelle intelligente. Il reçoit le flux vidéo provenant des caméras de surveillance, exécute les algorithmes de reconnaissance automatique des plaques, puis transmet les résultats à Firebase pour stocker. Il est également chargé de déclencher des actions physiques en communiquant avec le microcontrôleur ESP32 via Firebase.

Le ESP32 exécute les actions physiques en fonction des commandes reçues : affichage de messages personnalisés sur un écran LCD, activation des LEDs verte ou rouge selon l'autorisation, émission d'un signal sonore via un buzzer, et commande d'ouverture ou de

fermeture de la barrière automatique. Il récupère ces instructions en temps réel depuis Firebase afin d'assurer une synchronisation fluide avec l'état global du système.

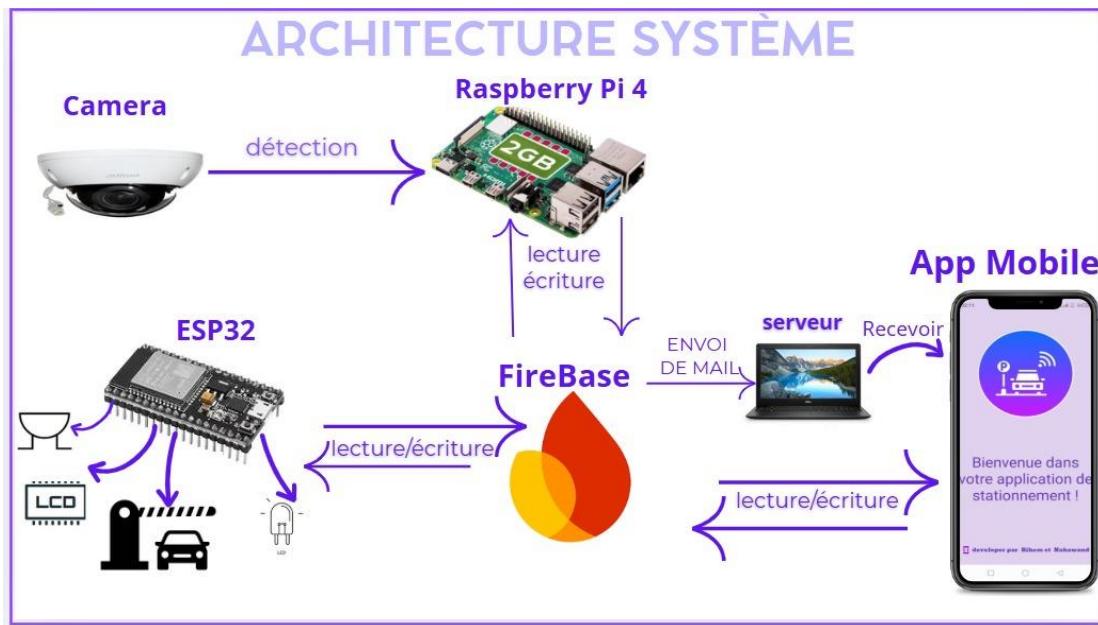


Figure 14: Architecture système

7. Diagramme de déploiement

Un diagramme de déploiement est un diagramme qui montre l'architecture physique d'un système pendant l'exécution. Il décrit comment les composants logiciels sont déployés sur l'infrastructure matérielle (serveurs, PC, capteurs, carte électroniques.)

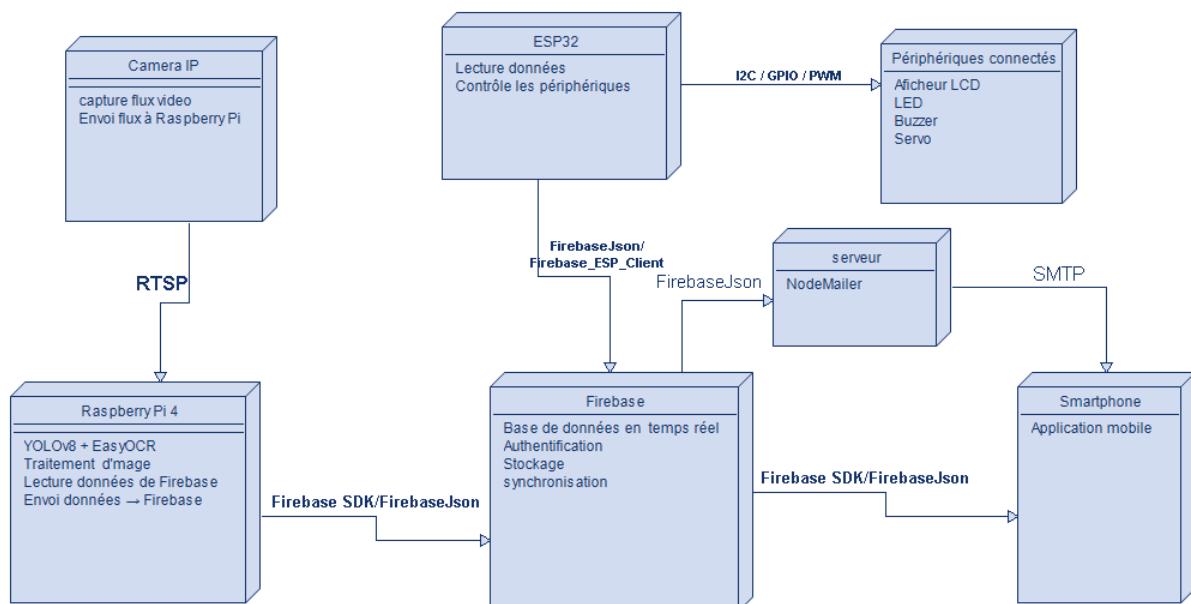


Figure 15:Diagramme de déploiement

Conclusion

Dans cette section, une stratégie de développement a été élaborée en identifiant les exigences fonctionnelles et non fonctionnelles du projet, ainsi que les différents rôles des utilisateurs. Le Backlog du système a également été établi et les sprints à venir ont été planifiés en détail. L'étape suivante consiste au lancement de la première version, qui sera abordé dans le prochain chapitre.

Chapitre 3 : Sprint 1

Introduction

Ce chapitre présente une explication détaillée de l'analyse, la conception et de mise en œuvre la première partie. Cette sprint a pour objectif d'entraîner un modèle câble de détecter les plaques d'immatriculation et réaliser l'interface utilisateur (application mobile).

1. Backlog Sprint 1

Ce backlog présente les sous taches de premier sprint :

User Story	Sous-Tâches	Priorité
En tant que SID, je peux capturer le flux vidéo en temps réel et détecter les plaques	<ul style="list-style-type: none">- Collecter des données (images de plaques d'immatriculation).- Installer labelImg- Annoter l'ensemble des données (format YOLO).- Charger données sur le drive- Entrainer un modèle YOLO sur Google Colab (notebook structuré). - Tester le modèle entraîné sur de nouvelles données.- Réentraîner le modèle avec Data Augmentation si besoin.- Capturer un flux vidéo réel (caméra ou vidéo de test).- Installer open cv et cvzone et ultralytics- Définir une zone spécifique de détection .- Installer EasyOCR pour lecture des plaques.- Développer le code pour lire correctement les plaques reconnues.	H
En tant qu'administrateur, je peux m'inscrire et m'authentifier	<ul style="list-style-type: none">- Créer l'interface d'inscription et de connexion- Créer la base de données (Firebase Authentication / Realtime DB).- Implémenter Firebase Auth (sécurité + logique de	H

	<p>connexion).</p> <ul style="list-style-type: none"> - Relier le frontend (Android) au backend Firebase. - Sauvegarder l'état de connexion avec SharedPreferences (mobile). 	
En tant qu'administrateur je dois pouvoir gérer les employés , les autres admins et les plaques d'immatriculation autorisés	<ul style="list-style-type: none"> - Concevoir la structure de la base de données (utilisateurs, rôles, plaques). - Développer les interfaces de gestion (ajout, suppression, modification). - Implémenter les fonctionnalités CRUD (Create, Read, Update, Delete). - Tester toutes les fonctionnalités . 	M
En tant qu'administrateur, je peux récupérer mon mot de passe	<ul style="list-style-type: none"> - Installer et configurer NodeMailer (pour envoi de mail). - Relier Firebase Auth à un serveur NodeMailer local - Développer l'interface de réinitialisation de mot de passe. - Tester le processus de récupération sur l'application 	M

Tableau 10: Backlog Sprint 1

2. Expression des besoins

2.1. Diagramme de cas d'utilisation sprint 1

Ce diagramme de cas d'utilisation pour sprint 1 décrit les différents acteurs et leurs cas d'utilisation, fournissant une compréhension claire du fonctionnement de l'application et la méthode de déction des plaques.

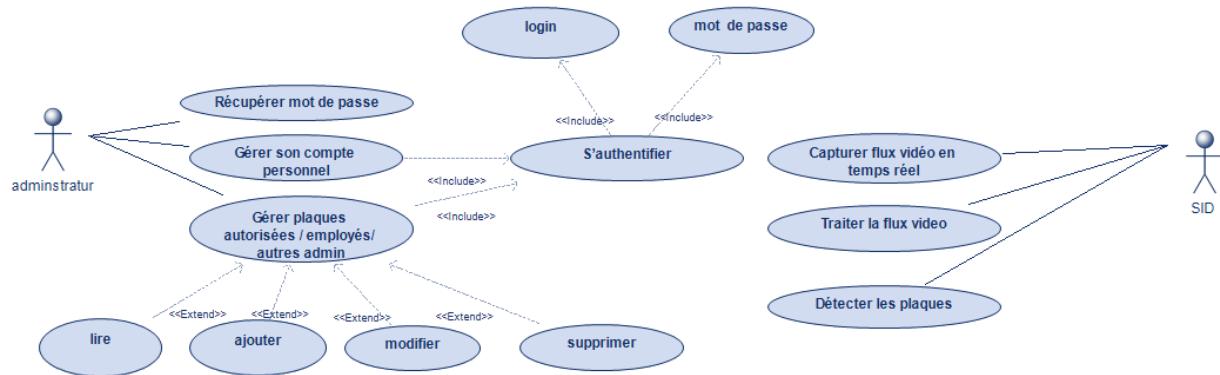


Figure 16 : Diagramme de cas d'utilisation sprint 1

2.2. Description textuelle des cas d'utilisation de sprint 1

Cas d'utilisation « S'inscrire »	
Objectif	❖ Créer un compte administrateur pour accéder à l'application mobile.
Acteur	Administrateur
Pré-conditions	❖ L'administrateur possède une adresse e-mail valide
Scénario nominal	<ul style="list-style-type: none"> L'administrateur saisit ses informations (nom, e-mail, mot de passe). Le système vérifie l'exactitude des données saisies. Si les données sont valides et l'adresse e-mail n'est pas déjà utilisée : <ul style="list-style-type: none"> ✓ Le système crée un nouveau compte. ✓ Un message de confirmation est affiché.
Scénario alternative	<ul style="list-style-type: none"> Le nom, l'e-mail ou le mot de passe est incorrect ou manquant. Le système affiche un message d'erreur : "Veuillez remplir correctement tous les champs." ou "Cette adresse e-mail est déjà utilisée."
Post-conditions	❖ Le compte administrateur est créé.

	❖ L'administrateur peut se connecter à l'application.
--	---

Tableau 11: Description textuelle de cas S'inscrire

Cas d'utilisation « Gérer son compte personnel »	
Objectif	❖ Voir les informations personnelles ou modifier le mot de passe
Acteur	Administrateur
Pré-conditions	❖ Être connecté
Scénario nominal	<ul style="list-style-type: none"> • L'administrateur accède à la page infos admins • Il peut voir ses infos ou changer son mot de passe • Le système met à jour les données
Scénario alternative	<ul style="list-style-type: none"> • Si la modification échoue le système affiche message d'erreur (problème de connexion, mot de passe invalide)
Post-conditions	❖ Les modifications sont enregistrées

Tableau 12: Description textuelle de cas Gérer son compte personne

Cas d'utilisation «Récupérer le mot de passe »	
Objectif	❖ Réinitialiser le mot de passe en cas d'oubli
Acteur	Administrateur
Pré-conditions	❖ l'administrateur ne peut pas accéder au tableau de bord
Scénario nominal	<ul style="list-style-type: none"> • L'admin clique sur "Mot de passe oublié" • Il entre son email • Il reçoit un lien de réinitialisation
Scénario alternative	<ul style="list-style-type: none"> • Si l'email n'existe pas dans la base Message d'erreur s'affiche
Post-conditions	❖ Le nouveau mot de passe est enregistré et l'administrateur peut accéder au tableau de bord

Tableau 13: Description textuelle de cas Récupérer le mot de passe

Cas d'utilisation «Gérer plaques, employés autorisés et liste administrateurs »	
Objectif	❖ Gérer les autorisations d'accès
Acteur	Administrateur
Pré-conditions	❖ Être connecté
Scénario nominal	<ul style="list-style-type: none"> • Ajouter, modifier ou supprimer une plaque/employé/admin • L'admin entre les données nécessaires • Le système enregistre les modifications
Scénario alternative	<ul style="list-style-type: none"> • Si données invalides (Message d'erreur et rejet de l'opération)
Post-conditions	❖ La base des autorisations est mise à jour

Tableau 14: Description textuelle de cas Gérer plaques, employés autorisés et liste administrateurs

Cas d'utilisation « Associer plaques aux véhicules (modifier) »	
Objectif	❖ Lier une plaque à un employé
Acteur	Administrateur
Pré-conditions	❖ L'administrateur accéder au page voire liste des employées
Scénario nominal	<ul style="list-style-type: none"> • L'admin sélectionne un employé • Il modifie la plaque
Scénario alternative	<ul style="list-style-type: none"> • Si les champs est vide un message d'erreur s'affiche
Post-conditions	❖ L'association est enregistrée

Tableau 15: Description textuelle de cas Associer plaques aux véhicules

Cas d'utilisation « S'authentifier »

Objectif	❖ Accéder à l'application mobile avec un compte existant.
Acteur	Administrateur
Pré-conditions	<ul style="list-style-type: none"> ❖ Le compte administrateur existe. ❖ L'administrateur connaît son e-mail et mot de passe.
Scénario nominal	<ul style="list-style-type: none"> • L'administrateur saisit son e-mail et mot de passe. • Le système vérifie les informations. • Si les données sont valides, l'accès est autorisé.
Scénario alternative	<ul style="list-style-type: none"> • L'e-mail ou le mot de passe est incorrect. • Le système affiche un message d'erreur :"Identifiants incorrects."ou "Échec de connexion à la base de données."
Post-conditions	<ul style="list-style-type: none"> ❖ L'administrateur est redirigé vers le tableau de bord de l'application.

Tableau 16: Description textuelle de cas S'authentifier

Cas d'utilisation « Traiter la vidéo / Déetecter plaques »	
Objectif	❖ Identification
Acteur	Système principal
Pré-conditions	<ul style="list-style-type: none"> ❖ Avoir entrainé un modèle pour déctecter les plaques
Scénario nominal	<ul style="list-style-type: none"> • Le programme utilise modèle YOLO et OCR pour lire la plaque
Scénario alternative	<ul style="list-style-type: none"> • Si le modèle est mauvais entraîné le résultat est vide
Post-conditions	<ul style="list-style-type: none"> ❖ Plaque détectée

Tableau 17 : Description textuelle de cas Traiter la vidéo / Déetecter plaques

3. Conception

Dans cette phase du sprint, la conception constitue la deuxième étape. Elle implique l'examen et la discussion des diagrammes de séquences détaillés.

3.1. Diagramme de séquence détaillé

Les diagrammes de séquences sont la représentation graphique des interactions entre les acteurs et le système selon un ordre chronologique dans la formulation Unified Modeling language.

➤ Diagramme de séquence détaillé du cas d'utilisation « authentification »

L'administrateur ou l'employé initie le processus d'authentification en saisissant ses identifiants de connexion, à savoir son adresse e-mail et son mot de passe. Le système procède alors à la vérification de la validité de ces informations. Si les identifiants sont corrects, l'utilisateur est autorisé à accéder à l'interface correspondante. En cas d'erreur (identifiants invalides ou incorrects), l'accès est refusé et un message d'erreur s'affiche, invitant l'utilisateur à réessayer. Ce cycle peut se répéter jusqu'à la saisie de données valides permettant une connexion réussie. Le déroulement de cette interaction est représenté dans le diagramme de séquence associé.

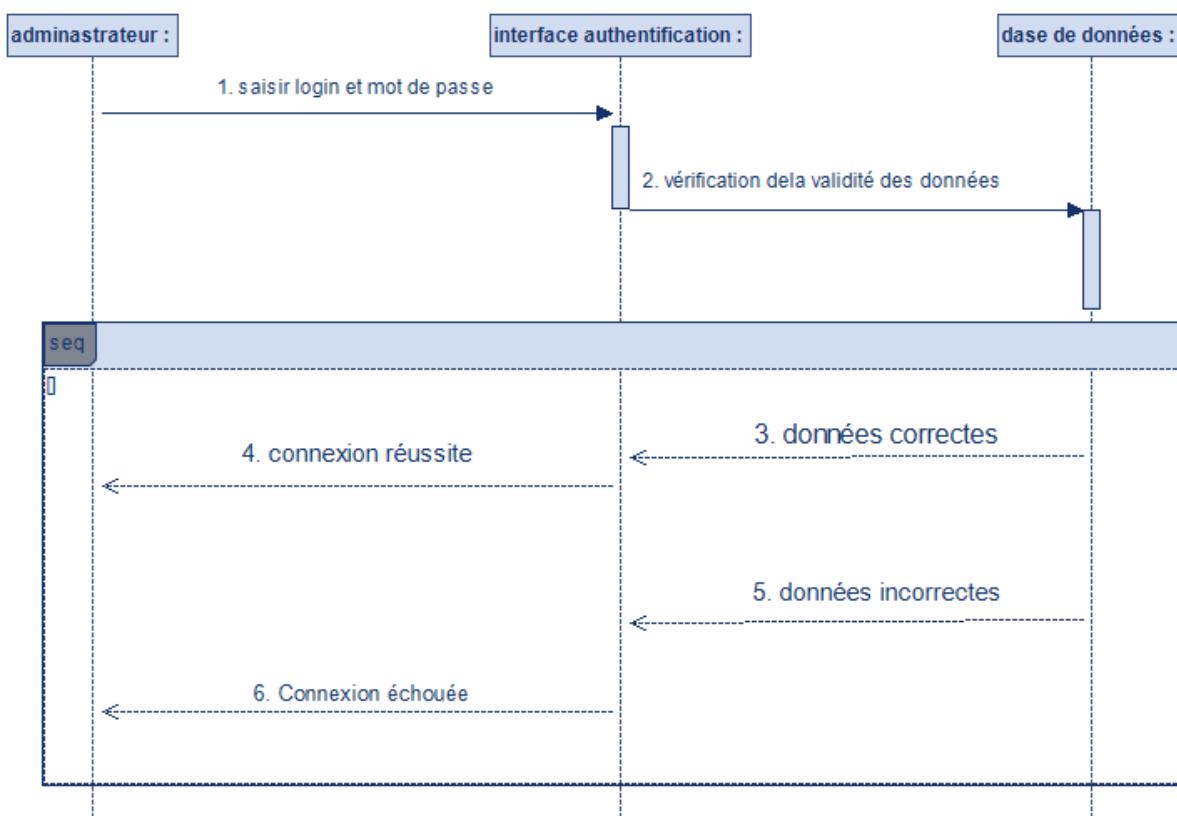


Figure 17: Diagramme de séquence authentification

➤ Diagramme de séquence détaillé du cas d'utilisation « Inscription »

Le processus d'inscription débute lorsque l'administrateur remplit les champs requis du formulaire, notamment son nom, son adresse e-mail et un mot de passe. Le système entame alors une phase de validation des données saisies. Cette vérification consiste à s'assurer que l'adresse e-mail est bien unique et au format valide, et que tous les champs obligatoires ont été correctement complétés. Si toutes les données sont conformes, le système enregistre l'administrateur et crée un nouveau compte. En revanche, si des erreurs sont détectées (champs manquants, e-mail déjà utilisé, etc.), le système affiche un message d'erreur invitant l'administrateur à corriger les informations erronées. Ce processus de vérification se répète jusqu'à ce que toutes les conditions soient remplies, permettant ainsi la finalisation de l'inscription. Cette interaction est représentée de manière détaillée dans le diagramme de séquence correspondant.

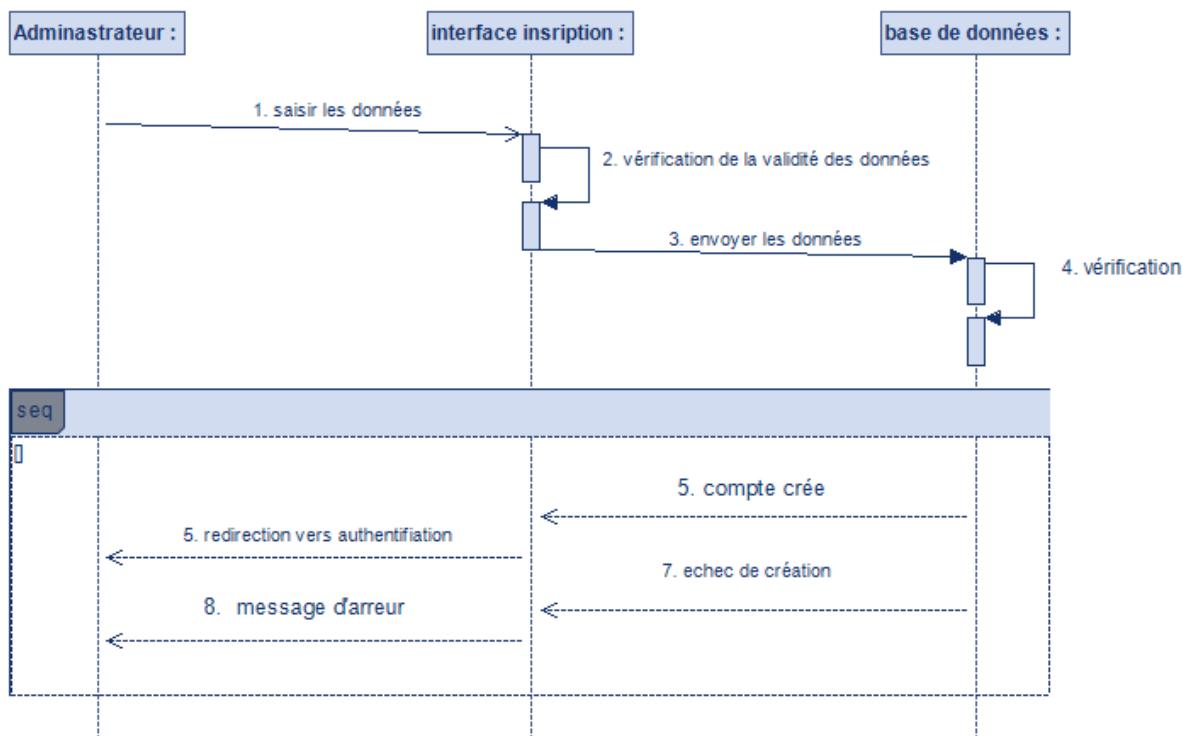


Figure 18: :Diagramme de séquence Incription

➤ Diagramme de séquence détaillé du cas d'utilisation « Ajout employé »

Le cas d'utilisation « Ajouter un employé » permet à un administrateur authentifié d'enregistrer un nouvel employé dans le système. Une fois connecté, l'administrateur accède à l'interface dédiée et saisit les informations nécessaires, telles que le nom et la plaque d'immatriculation de l'employé. L'interface procède alors à une vérification de la validité des données saisies. Si les informations sont correctes, elles sont envoyées à la base de données pour y être enregistrées.

Une fois l'enregistrement effectué avec succès, le système notifie l'administrateur que l'employé a été ajouté. En revanche, si la validation échoue ou si une erreur survient lors de l'enregistrement (comme un problème de connexion ou des données invalides), le système affiche un message d'erreur à l'administrateur. Tel que représenté dans le diagramme séquentiel

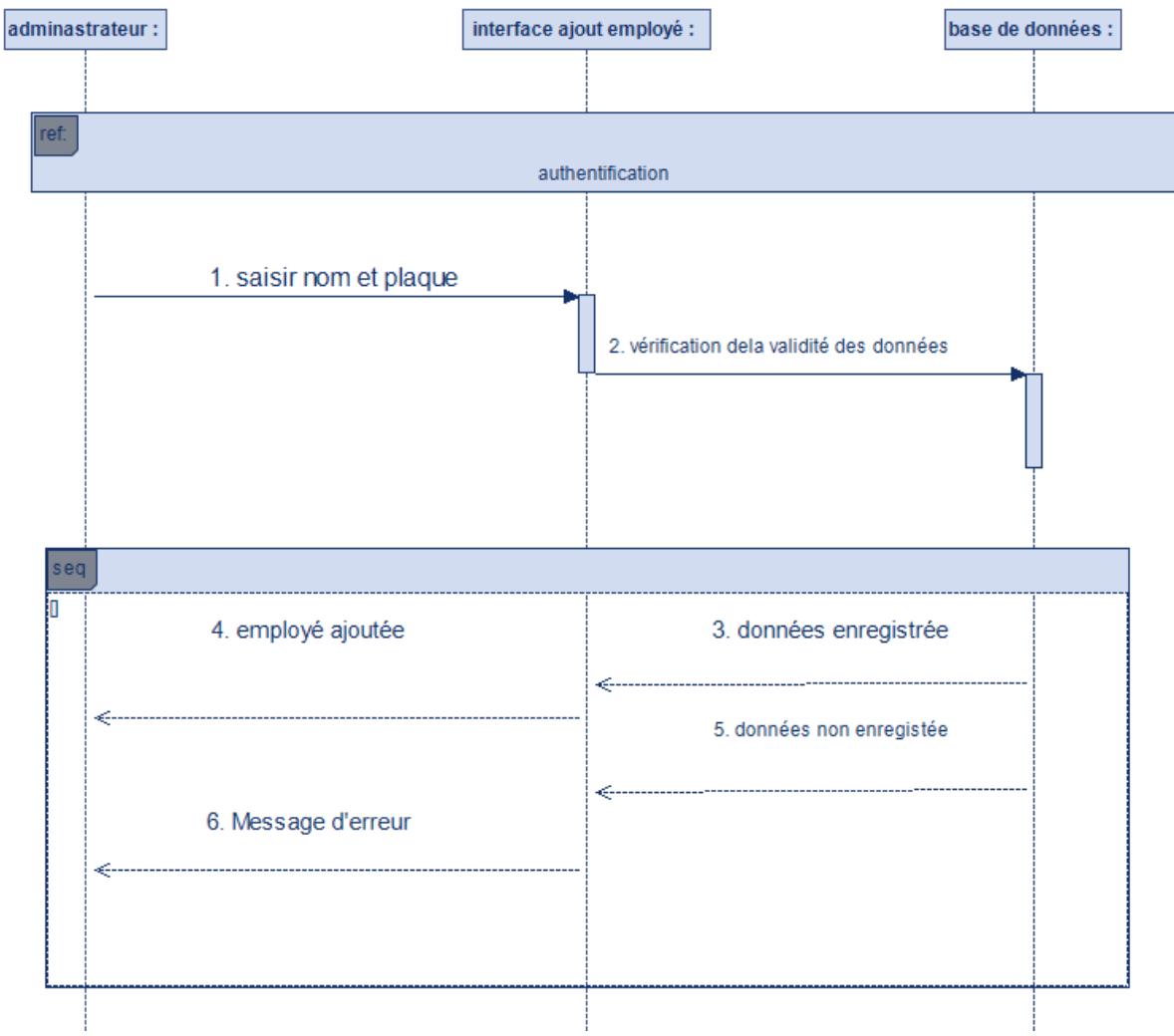


Figure 19: Diagramme de séquence Ajout employé

4. Configuration l'environnement de travail

Afin de développer notre système de vision par ordinateur basé sur YOLO pour la détection en temps réel, il est essentiel de préparer un environnement de travail adéquat comprenant les outils et bibliothèques nécessaires.

4.1. Installation des bibliothèques

Avant d'installer les différentes bibliothèques nécessaires, un environnement virtuel a été créé afin de structurer notre espace de travail et d'éviter les conflits de dépendances. Une fois l'environnement activé, nous avons exécuté les commandes suivantes dans le terminal :

- **pip install labelImg**
- **pip install opencv-python**
- **pip install cvzone**
- **pip install ultralytics**
- **pip install easyocr**

Ces installations permettent d'obtenir toutes les dépendances et versions requises pour démarrer le développement de notre projet dans de bonnes conditions.

Voici comment vérifier la présence des bibliothèques dans l'environnement virtuel :

```
(base) C:\Users\DELL>conda activate computer
(computer) C:\Users\DELL>pip show opencv-python
Name: opencv-python
Version: 4.11.0
Summary:
Home-page:
Author:
Author-email:
License:
Location: c:\users\dell\anaconda3\envs\computer\lib\site-packages
Requires:
Required-by: cvzone, ultralytics

(computer) C:\Users\DELL>pip show cvzone
Name: cvzone
Version: 1.6.1
Summary: Computer Vision Helping Library
Home-page: https://github.com/cvzone/cvzone.git
Author: Computer Vision Zone
Author-email: contact@computervision.zone
License: MIT
Location: c:\users\dell\anaconda3\envs\computer\lib\site-packages
Requires: numpy, opencv-python
Required-by:
```

Figure 20: opencv et cvzone

```
(computer) C:\Users\DELL>pip show ultralytics
Name: ultralytics
Version: 8.3.96
Summary: Ultralytics YOLO 🦄 for SOTA object detection, multi-object tracking, instance segmentation, pose estimation and image classification.
Home-page: https://ultralytics.com
Author:
Author-email: Glenn Jocher <glenn.jocher@ultralytics.com>, Jing Qiu <jing.qiu@ultralytics.com>
License: AGPL-3.0
Location: c:\users\dell\anaconda3\envs\computer\lib\site-packages
Requires: matplotlib, numpy, opencv-python, pandas, pillow, psutil, py-cpuinfo, pyyaml, requests, scipy, seaborn, torch, torchvision, tqdm, ultralytics-thop
Required-by:

(computer) C:\Users\DELL>pip show easyocr
Name: easyocr
Version: 1.7.2
Summary: End-to-End Multi-Lingual Optical Character Recognition (OCR) Solution
Home-page: https://github.com/jaidedai/easyocr
Author: Rakpong Kittinaradorn
Author-email: r.kittinaradorn@gmail.com
License: Apache License 2.0
Location: c:\users\dell\anaconda3\envs\computer\lib\site-packages
Requires: ninja, numpy, opencv-python-headless, Pillow, pyclipper, python-bidi, PyYAML, scikit-image, scipy, Shapely, torch, torchvision
Required-by:
```

Figure 21: yolo et easyocr

```
(computer) C:\Users\DELL>pip show labelImg
Name: labelImg
Version: 1.8.6
Summary: LabelImg is a graphical image annotation tool and label object bounding boxes in images
Home-page: https://github.com/tzutalin/labelImg
Author: TzuTa Lin
Author-email: tzu.ta.lin@gmail.com
License: MIT license
Location: c:\users\dell\anaconda3\envs\computer\lib\site-packages
Requires: lxml, PyQt5
Required-by:

(computer) C:\Users\DELL>
```

Figure 22: labelImg

4.2. Configuration du Firebase

Avant de commencer le développement de l'application mobile, il est essentiel de concevoir la structure de la base de données ainsi que de choisir les services adaptés à notre besoin. Dans ce projet, nous avons opté pour Firebase.

Les étapes principales de configuration sont les suivantes :

1. Création d'un projet Firebase via la console Firebase
2. Ajout d'une base de données Realtime Database pour stocker et synchroniser les données en temps réel.
3. Activation de Firebase Authentication en choisissant le mode de connexion par adresse email et mot de passe.
4. Téléchargement du fichier google-services.json contenant les informations de configuration (API Key, project ID, ...) nécessaires à l'intégration de Firebase dans l'application Android.
5. Connexion du projet Firebase à Android Studio, en important le fichier google-services.json et en modifiant les fichiers build.gradle pour inclure les dépendances requises.

Les figures suivant montrant la création de base de données en temps réel et Firebase authentication et la connexion de l'Android studio a la base.

The screenshot shows the Firebase Realtime Database interface for a project named "SmartParkingApp". The left sidebar has a "Realtime Database" section selected. The main area displays a tree view of the database structure:

```

https://smartparkingapp-48ba1-default.firebaseio.com
  +-- admins
  +-- employees
    +-- EMP001
      +-- employee_name: "Rihem Aridhi"
      +-- id: "001"
      +-- plate_number: "DL7D5017"
    +-- EMP002
      +-- employee_name: "Nahawand Mekni"
  
```

Figure 23: Realtime DB

The screenshot shows the Firebase Authentication interface for the same project. The left sidebar has an "Authentication" section selected. The main area shows the "Méthode de connexion" (Method of connection) tab, which lists a provider for email/password:

Fournisseur	État
Adresse e-mail/Mot de passe	Activé

Figure 24: firebase authentication



Your Android Studio project is connected
to your Firebase Android app

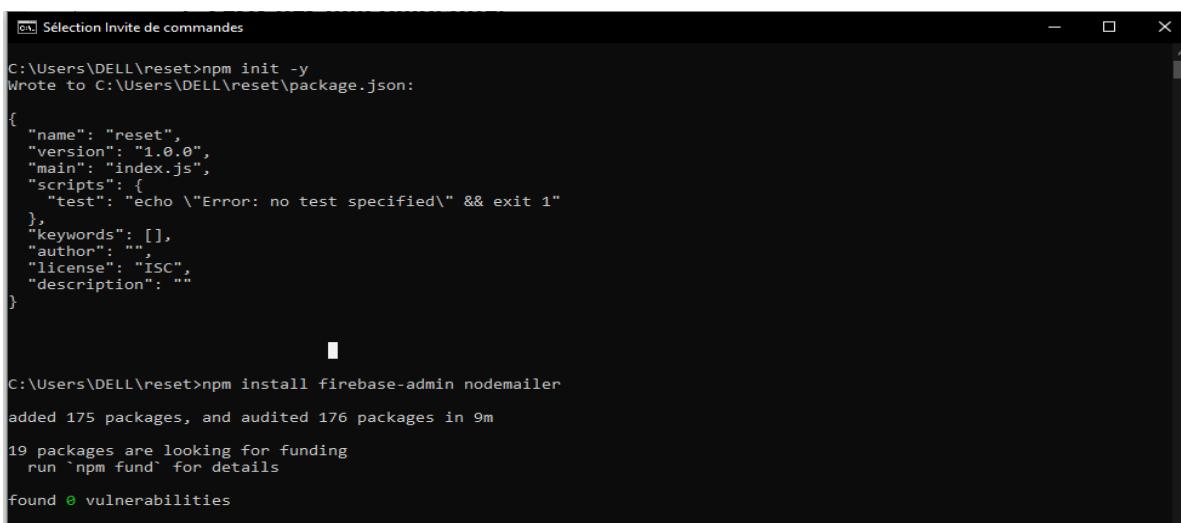
You can now use Firebase in your project! Go back to Android Studio to start using one of the Firebase SDKs.

Figure 25: connexion Firebase a l'Android studio

4.3. Installation du NodeMailer

Nous avons rencontré une limitation avec Firebase Authentication, qui ne permet pas l'envoi de plusieurs types d'e-mails personnalisés. Pour résoudre ce problème, nous avons intégré **NodeMailer**, une bibliothèque Node.js permettant d'envoyer des e-mails personnalisés depuis un serveur local.

Nous avons également installé **firebase-admin** afin d'établir une connexion sécurisée avec Firebase. Cela nous permet de déclencher l'envoi d'e-mails en fonction des événements dans la base de données. Cette solution offre plus de flexibilité pour répondre aux besoins spécifiques de notre application.



```
C:\Users\DELL\reset>npm init -y
Wrote to C:\Users\DELL\reset\package.json:

{
  "name": "reset",
  "version": "1.0.0",
  "main": "index.js",
  "scripts": {
    "test": "echo \\\"Error: no test specified\\\" && exit 1"
  },
  "keywords": [],
  "author": "",
  "license": "ISC",
  "description": ""
}

C:\Users\DELL\reset>npm install firebase-admin nodemailer
added 175 packages, and audited 176 packages in 9m
19 packages are looking for funding
  run `npm fund` for details
found 0 vulnerabilities
```

Figure 26: . Installation du NodeMailer

5. Tests des fonctionnalités

5.1. Annotation de données

Après avoir collecter les données, nous avons utilisé l'outil **LabelImg** pour annoter manuellement les plaques d'immatriculation présentes dans les images de notre donnée. Cet outil permet de dessiner des boîtes englobantes autour des objets d'intérêt, ici les plaques, et d'enregistrer les coordonnées correspondantes dans un format exploitable par les modèles d'apprentissage automatique.

Dans le terminal, l'exécution de la commande `labelImg` dans l'environnement virtuel active l'interface graphique de l'outil.

Comme illustré dans la figure suivant, nous avons sélectionné la région contenant la plaque d'immatriculation sur une image d'un véhicule. Cette annotation est associée à l'étiquette "**numberplate**". Une fois l'image annotée, le fichier correspondant au format `.txt` est généré, contenant les coordonnées exactes des boîtes englobantes.



Figure 27: l'interface graphique du labelImg

Voici la sortie générée, où chaque ligne représente une annotation sous forme de tuples contenant le label et les coordonnées des coins de la boîte. Ces données sont ensuite utilisées pour l'entraînement du modèle.

```
PS C:\Users\DELL\anaconda3\envs\computer> labelImg
[('numberplate', [(404, 292), (788, 292), (788, 378), (404, 378)], None, None, False)]
[('numberplate', [(413, 274), (811, 274), (811, 376), (413, 376)], None, None, False)]
[('numberplate', [(451, 273), (833, 273), (833, 372), (451, 372)], None, None, False)]
[('numberplate', [(519, 264), (899, 264), (899, 354), (519, 354)], None, None, False)]
[('numberplate', [(576, 261), (968, 261), (968, 361), (576, 361)], None, None, False)]
[('numberplate', [(667, 268), (1077, 268), (1077, 379), (667, 379)], None, None, False)]
[('numberplate', [(669, 312), (1047, 312), (1047, 415), (669, 415)], None, None, False)]
[('numberplate', [(586, 307), (955, 307), (955, 387), (586, 387)], None, None, False)]
```

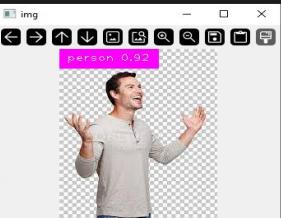
Figure 28: résultat d'annotation

5.2. Test d'OpenCV, CVZone et Ultralytics

Après avoir installé les bibliothèques nécessaires (OpenCV, CVZone et Ultralytics), nous avons testé le bon fonctionnement de l'algorithme de détection d'objets YOLOv8 sur une image locale.

L'image ci-dessus illustre ce test, où une personne a été détectée avec succès. Le modèle a identifié l'objet "**personne**" en traçant une boîte englobante autour de lui, accompagnée d'une étiquette indiquant la classe et le taux de confiance (**0.92**). CVZone a été utilisé pour afficher l'étiquette de manière claire et esthétique, tandis qu'OpenCV a servi au traitement et à l'affichage de l'image. Ce test confirme que l'intégration de YOLOv8 avec OpenCV et CVZone fonctionne correctement.

```
matricole > test.py > ...
1 import cv2
2 from ultralytics import YOLO
3 import cvzone
4 model = YOLO("yolov8n.pt")
5 img = cv2.imread("hand.jpg")
6 results = model(img)
7 classNames = model.names
8
9
10 for r in results:
11     for box in r.boxes:
12
13         x1, y1, x2, y2 = map(int, box.xyxy)
14         w, h = x2 - x1, y2 - y1
15
16         conf = round(float(box.conf[0]), 2)
17
18         cls = int(box.cls[0])
19         label = classNames[cls] if classNames else str(cls)
20


    A screenshot of a computer vision application interface. It shows a man with his hands raised, wearing a grey long-sleeved shirt and blue jeans. The background is a checkered pattern. Several bounding boxes are drawn around the man's head, hands, and torso. A pink box highlights the head area with the text "Person 0.92". Another small green box highlights a hand with the coordinates "(x=153, y=10) - R:252 G:252 B:252". The interface includes standard window controls (minimize, maximize, close) and a toolbar with various icons.
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS GITLENS

0: 640x480 1 person, 157.9ms
Speed: 6.7ms preprocess, 157.9ms inference, 2.2ms postprocess per image at shape (1, 3, 640, 480)
PS C:\Users\DELL\anaconda3\envs\computer> & c:/Users/DELL/anaconda3/envs/computer/python.exe c:/Users/.../matricole/test.py

0: 640x480 1 person, 177.9ms

Figure 29: résultat d'intégration

5.3. Exécution de la notebook d'entraînement sur Google Colab

Après avoir annoté notre dataset, puis chargé les images et annotations au format YOLO sur Google Colab, nous avons utilisé un notebook bien structuré pour lancer l'entraînement du modèle YOLOv8. Le fichier `data.yaml` a été soigneusement configuré pour spécifier le chemin des dossiers `training` et `validation`, ainsi que le noms des classes.

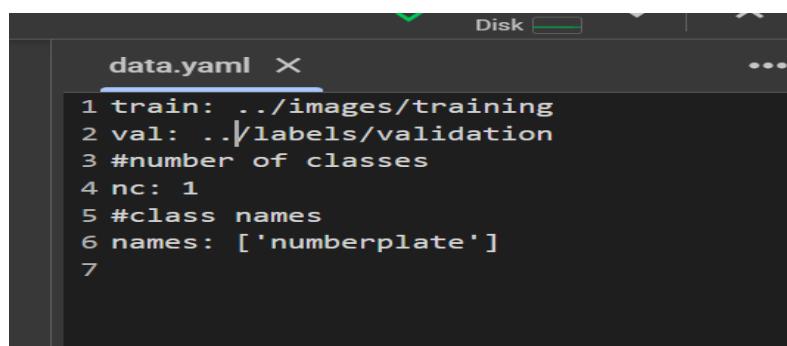


Figure 30: fichier data

Ensuite, nous avons utilisé la commande suivante, cette commande permet d'entraîner un modèle YOLOv8 small pendant 100 époques, avec des images redimensionnées à 800×800 pixels. Le paramètre plots=True génère automatiquement des graphiques d'évolution des performances, tels que la précision, la perte ou encore la matrice de confusion. L'environnement Google Colab nous a permis de bénéficier d'un GPU gratuit pour accélérer le processus d'entraînement. À la fin de l'apprentissage, un modèle .pt est généré et peut être téléchargé pour être testé sur nouvelles images ou intégré dans une application.

```
inflating: rihem2/rihem1/labels/validation/car_32.txt
1 min %cd (HOME)
!yolo task=detect mode=train model=yolov8s.pt data=/content/datasets/rihem2/rihem1/data.yaml epochs=100 imgsz=800 plots=True
↳ inloading https://github.com/ultralytics/assets/releases/download/v8.3.0/yolov8s.pt to 'yolov8s.pt'...
% 21.5M/21.5M [00:00<00:00, 176MB/s]
ralytics 8.3.127 🚀 Python-3.11.12 torch-2.6.0+cu124 CUDA:0 (Tesla T4, 15095MiB)
```

Figure 31: exécution de la commande d'entraînement

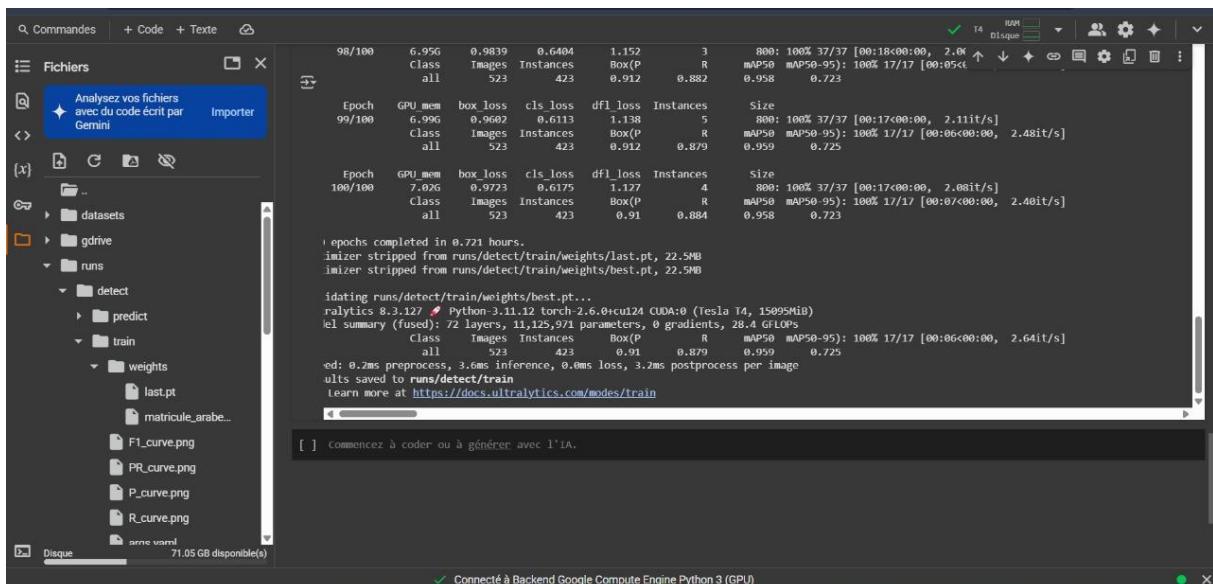


Figure 32: génération de la modèle

5.4. Test du modèle sur nouvelle données

Après avoir téléchargé le modèle entraîné, nous avons procédé à un test sur une nouvelle plaque d'immatriculation qui n'avait pas été utilisée lors de l'entraînement. Le modèle a détecté correctement la classe correspondante, démontrant sa capacité de généralisation. Lors de ce test, il a identifié la plaque avec un taux de confiance de 0.83, ce qui confirme une bonne performance dans des conditions réelles ou semi-réelles. Ce test valide la précision de notre

modèle et montre qu'il est prêt à être intégré dans une application de reconnaissance automatique de plaques d'immatriculation.

Figure 33: Test du modèle

5.5. Test des fonctionnalités de l'application

- #### • Test de l'authentification et Nodemailer

Nous avons commencé par tester le processus d'authentification (connexion / inscription). Lorsqu'un utilisateur souhaite réinitialiser son mot de passe, il peut le faire via un lien de réinitialisation envoyé automatiquement par Nodemailer. Ce module, exécuté depuis un serveur local (le PC agit comme serveur), est connecté à Firebase Authentication. Lors du test, après saisie d'une adresse e-mail dans l'application, un e-mail de réinitialisation a bien été envoyé et reçu, confirmant la communication entre l'application, Firebase et le serveur Nodemailer.

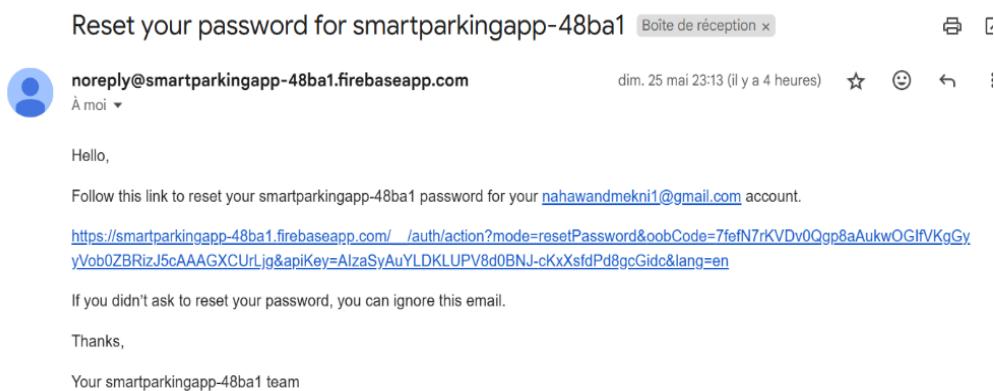


Figure 34: mail de réinitialisation

- **Test sur les jeux de données**

Nous avons également testé toutes les fonctionnalités CRUD (Create, Read, Update, Delete) sur les données des employés :

- ✓ Ajout d'un nouvel employé via un formulaire (nom complet, matricule,).
- ✓ Affichage de la liste complète des employés dans une interface claire.
- ✓ Mise à jour des données d'un employé sélectionné.
- ✓ Suppression d'un employé avec confirmation.
- ✓ Modifier le mot de passe
- ✓ Voir liste d'administrateurs
- ✓ Sauvegarder l'état de connexion
- ✓ Déconnexion

Chaque action est synchronisée avec Firebase Realtime Database, garantissant une mise à jour instantanée des données. Les tests ont été effectués avec différents jeux de données fictifs, ce qui nous a permis de valider le bon fonctionnement de l'application dans divers scénarios.

6. Réalisation

Les figures suivant représentent les Interfaces de l'application

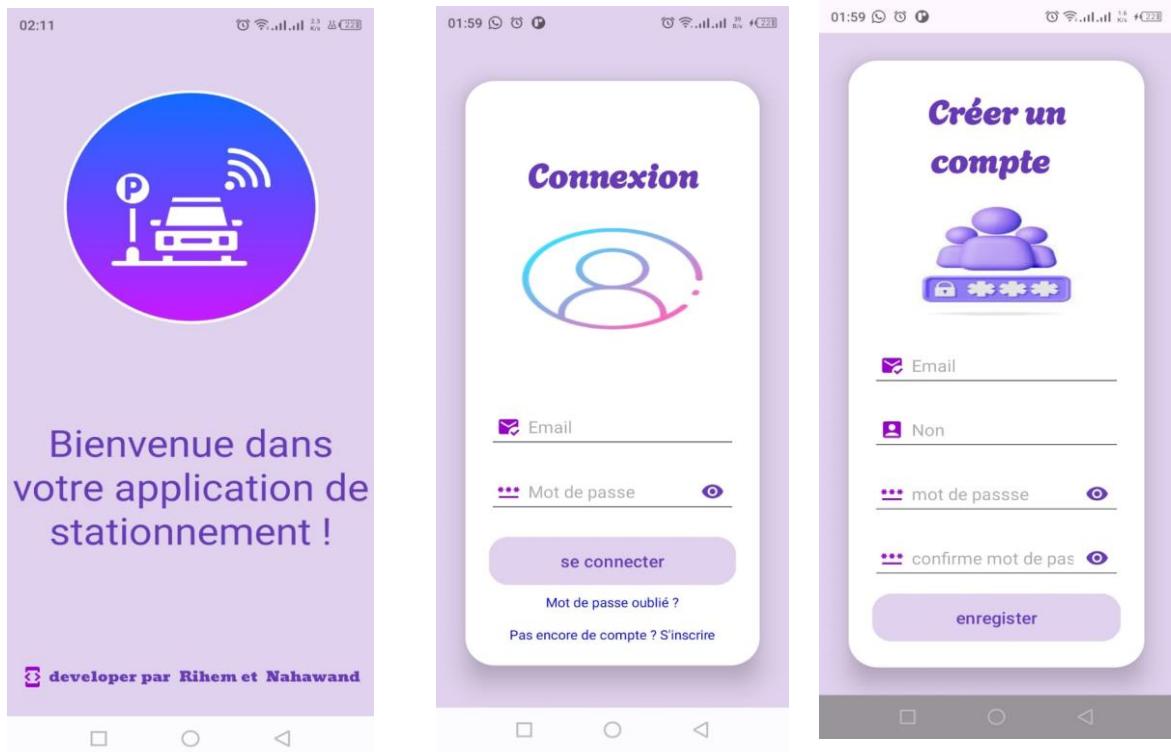


Figure 35: Accueil Dynamique, Authentification et Inscription

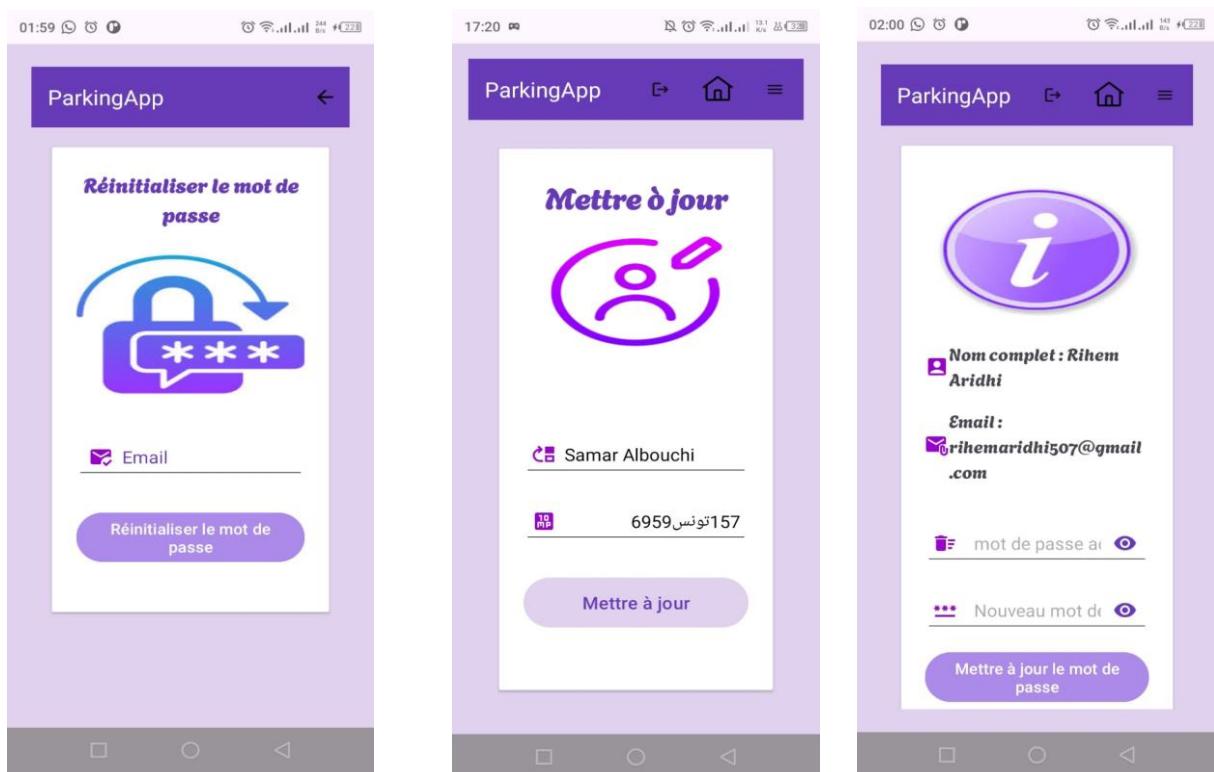


Figure 36: Réinitialisation, Mettre à jour employée et administrateur

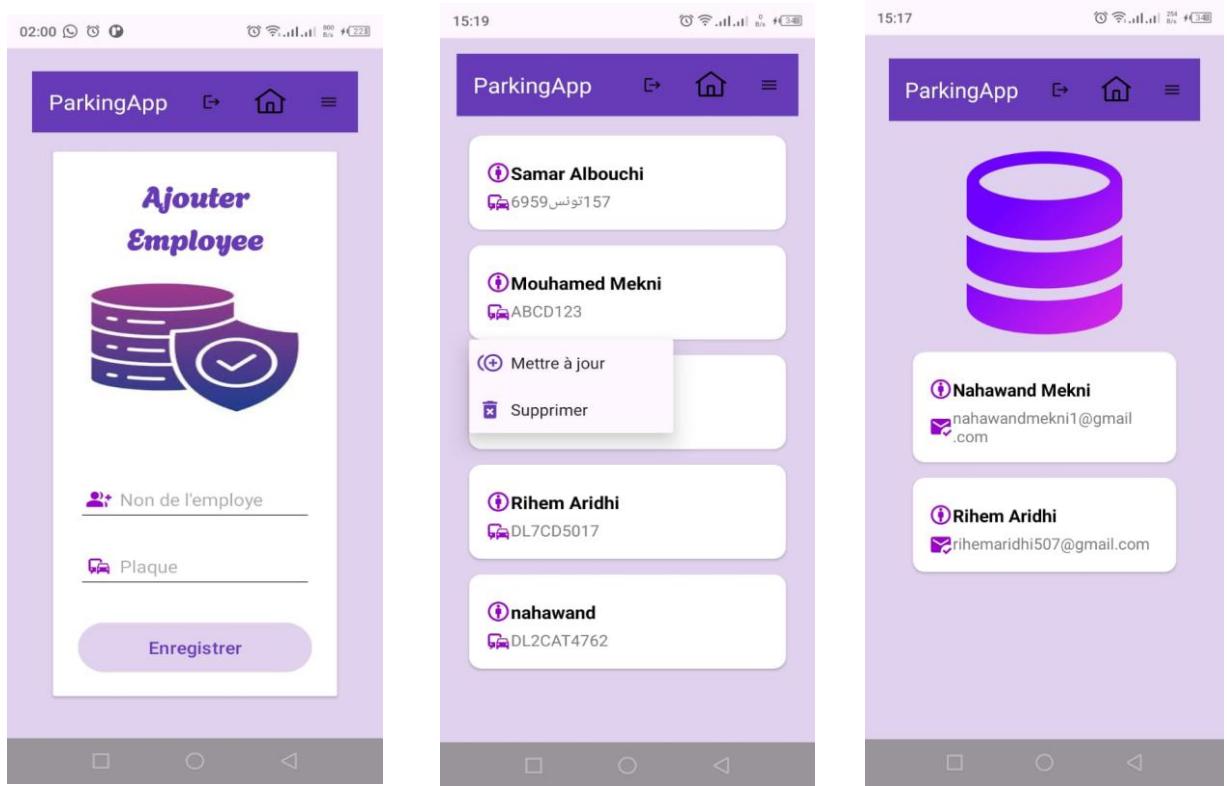


Figure 37: Ajout employée, Voir liste des employée et administrateur

- Voici le résultat final du programme de sprint 1 :

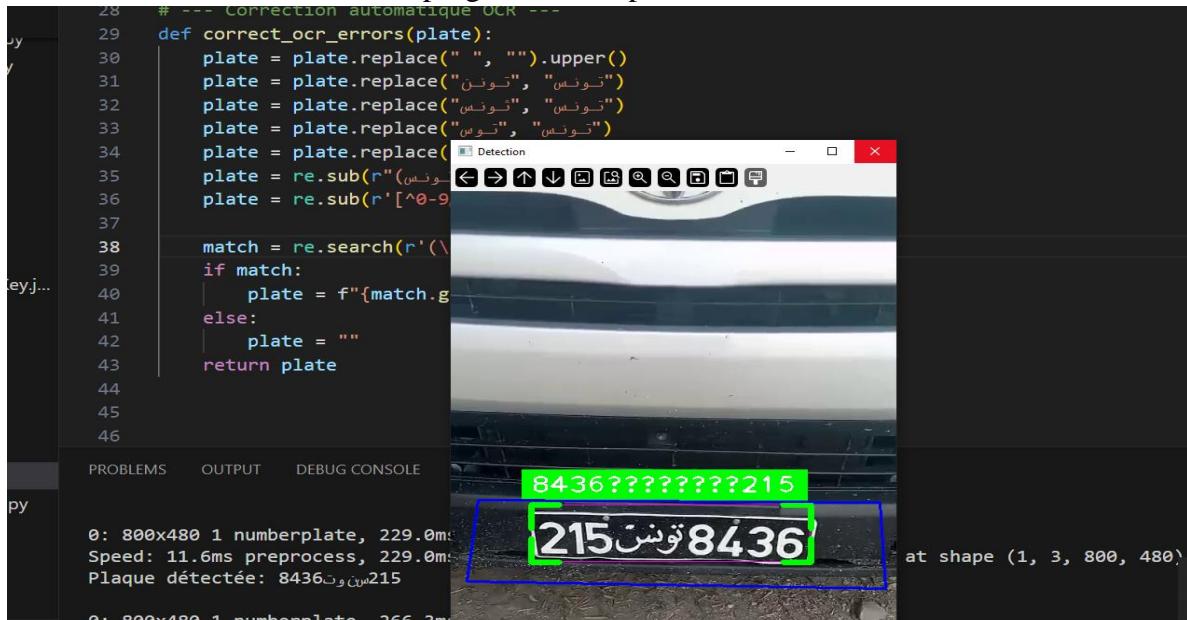


Figure 38: test final sprint 1

Conclusion

Au cours de ce chapitre, plusieurs étapes clés ont été réalisées. Tout d'abord, le backlog sprint a été organisé afin de définir les user stories à mettre en œuvre pour cette phase. Ensuite, un

diagramme de cas d'utilisation et un diagramme de séquence ont été élaborés pour représenter ce sprint. Par la suite, des captures d'écran des tests ont été effectuées. Enfin, les interfaces créées dans la partie réalisation ont été illustrées. Ce chapitre a également marqué l'initialisation du Sprint 2, posant les bases pour la suite du développement.

Chapitre 4 : Sprint 2

Introduction

Ce chapitre est consacré à la réalisation du deuxième sprint du projet, le sprint se concentre sur le développement du système principal et du contrôle physique, ainsi que sur le développement et les tests des interfaces administrateur (tableau de bord, historique). Le déploiement du système principal sur la carte Raspberry Pi 4 est également effectué. Enfin, une intégration complète est réalisée à travers plusieurs tests approfondis afin de valider le bon fonctionnement et la fiabilité du système.

1. Backlog Sprint 2

Ce Backlog décrit les sous-tâches de notre deuxième sprint

User Story	Sous-Tâches	Priorité
En tant que SID, je peux prendre une décision (autorisé / non autorisé)	-Concevoir la structure de base de données -installer firebase Admin Développer le code principal afin de de satisfaire ces conditions : - les plaques autorisées depuis base de données. - Comparer la plaque détectée à les plaques autorisées récupérer. -prendre une décision selon la comparaison : accès autorisé ou refusé. -Enregistrer l'état de l'accès (autorisé/non autorisé) avec date et heure -Tester avec des plaques autorisées et non autorisées -optimiser la performance du code	H
En tant que conducteur, je dois être détecté automatiquement à l'entrée/sortie	Développer le code principal afin de satisfaire ces conditions : - Générer un événement (entrée ou sortie) -envoyer les données nécessaires à la base de données en temps réel . -Tester la détection automatique avec plusieurs cas réels	H

<p>En tant que SID, je peux suivre le nombre de places disponibles)</p>	<p>-Développer le code principal afin de satisfaire les conditions suivantes :</p> <ul style="list-style-type: none"> • Initialiser un compteur représentant le nombre de places disponibles. • Mettre à jour ce compteur à chaque entrée ou sortie d'un véhicule. • Envoyer la valeur actualisée du compteur vers la base de données Firebase en temps réel. • Gérer le cas du parking complet (lorsqu'il n'y a plus de places disponibles). <p>-Tester les différents scénarios (parking plein, sortie sans entrée, double détection, etc.) pour valider la fiabilité du système.</p>	<p>M</p>
<p>En tant que conducteur, je peux accéder si ma plaque est autorisée</p>	<p>-Connecter un servomoteur à l'ESP32</p> <p>-Ajouter les données nécessaires dans la base de données Firebase pour gérer l'état d'autorisation et la commande de la barrière.</p> <p>-Modifier le code principal afin d'intégrer la gestion du servomoteur.</p> <p>-Développer le code de contrôle du servomoteur afin de satisfaire les conditions suivantes :</p> <ul style="list-style-type: none"> • Contrôler le servomoteur pour ouvrir ou fermer la barrière. • Lire l'état d'autorisation depuis Firebase et prendre une décision (autoriser ou refuser l'ouverture). • Intégrer une temporisation automatique après l'ouverture pour refermer la barrière au bout d'un certain délai. 	

	<p>-Tester la séquence complète : détection de la plaque → vérification Firebase → commande du servomoteur → temporisation → fermeture.</p>	
En tant que SCB, je peux afficher des messages personnalisé via LCD	<p>-Développer le code principal pour envoyer les données nécessaires à Firebase</p> <p>- Câbler et tester l'afficheur LCD I2C sur l'ESP32</p> <p>Développer le code de contrôle afin de satisfaire les conditions suivantes :</p> <ul style="list-style-type: none"> • Lire depuis Firebase le nombre actuel de places disponibles ou occupées. • Afficher dynamiquement un message sur l'écran LCD. • Mettre à jour les données en temps réel selon les entrées détectées. <p>-Tester l'ensemble des fonctionnalités : détection → autorisation → contrôle du servomoteur → mise à jour Firebase → affichage sur l'écran LCD.</p>	M
En tant que SCB , je peux activer LED verte/rouge et buzzer	<p>-Câbler et tester les composants physiques sur l'ESP32.</p> <p>- Développer les fonctions accesAutorise() et accesRefuse() dans le code de contrôle .</p> <p>- Implémenter la lecture en temps réel depuis Firebase.</p> <p>- Synchroniser l'affichage avec les signaux physiques.</p> <p>- Tester différents scénarios de plaques détectées.</p> <p>-Vérifier la fiabilité et la sécurité physique du système.</p>	M

En tant qu'administrateur, je peux accéder au tableau de bord	<ul style="list-style-type: none"> -Mettre à jour la structure de la base de données Firebase -Développer le principal pour envoyer les données en temps réel (plaque, date, heure, statut, ...) -Implémenter la lecture des données dans l'application Android (Java) -Afficher les données dynamiquement dans l'application (nombre de places, dernière plaque, liste de plaques détecte aujourd'hui,) -Gérer les erreurs de lecture et de synchronisation Firebase -Optimiser la fréquence des mises à jour -Tester la réactivité de l'affichage mobile 	H
En tant qu'administrateur, je peux consulter l'historique des véhicules	<ul style="list-style-type: none"> -Optimiser le code principal : <ul style="list-style-type: none"> • Ajouter une sauvegarde automatique à chaque détection de plaque. • Structurer les enregistrements dans Firebase. • Inclure : plaque, date, heure, autorisation, type d'événement (entrée/sortie). -Implémenter le code Java Android pour récupérer les données depuis Firebase -Tester les fonctionnalités 	H

Tableau 17: Backlog Sprint 2

2. Expression des besoins

2.1. Diagramme des cas d'utilisation de sprint 2

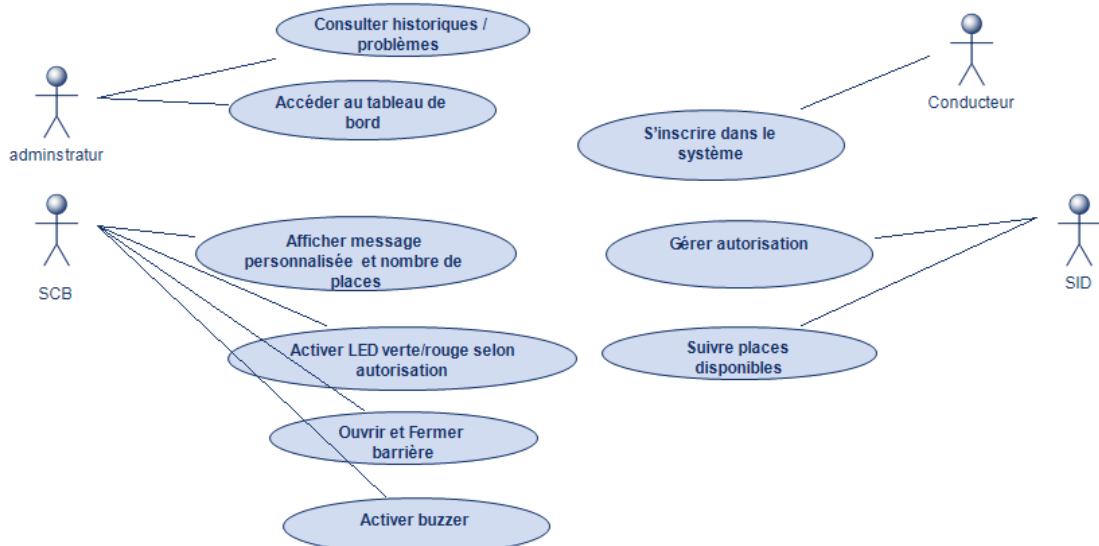


Figure 39: Diagramme de cas d'utilisation Sprint2

2.2. Description textuelle de cas d'utilisation sprint 2

Cas d'utilisation « Détection et accès automatique avec affichage »	
Objectif	<ul style="list-style-type: none"> ❖ Déetecter automatiquement un véhicule à l'entrée ou à la sortie. ❖ Autoriser l'accès si la plaque est reconnue, et afficher un message personnalisé avec le nombre de places disponibles.
Acteur	SID/SCB
Pré-conditions	<ul style="list-style-type: none"> ❖ Le système de détection (caméra, traitement vidéo, OCR, connexion Firebase) est opérationnel. ❖ L'écran est fonctionnel et connecté au système. ❖ Un véhicule approche de la zone d'entrée ou de sortie.
Scénario nominal	<ul style="list-style-type: none"> • Le système active la caméra et capture le flux vidéo en temps réel. • Une image de la plaque d'immatriculation est capturée. • Le système principal effectue: <ul style="list-style-type: none"> • La détection de la plaque via YOLO. • L'extraction du texte par OCR. • La vérification de l'autorisation via Firebase. • Si la plaque est autorisée: <ul style="list-style-type: none"> ✓ Le nom du conducteur est récupéré. ✓ Un message de bienvenue personnalisé est affiché sur l'écran LCD

	<ul style="list-style-type: none"> ✓ Le nombre de places disponibles est affiché ✓ La barrière s'ouvre automatiquement. ✓ L'événement est enregistré dans la base de données. • Si la plaque n'est pas reconnue: <ul style="list-style-type: none"> ✓ Un message d'erreur s'affiche (Accès refusé). ✓ Un signal d'alerte sonore est activé. ✓ L'événement est enregistré
Scénario alternative	<ul style="list-style-type: none"> • Échec de connexion au système (Firebase) :Message : «Échec de chargement des données »
Post-conditions	<ul style="list-style-type: none"> ❖ Le conducteur est soit autorisé à entrer/sortir, soit l'accès est refusé. ❖ Un message est affiché sur l'écran (accueil personnalisé ou erreur). ❖ Le nombre de places disponibles est affiché et mis à jour. ❖ L'événement est enregistré (plaqué, heure, type : entrée/sortie, statut).

Tableau 18: Description textuelle de cas d'utilisation Détection et accès automatique avec affichage

Cas d'utilisation « prendre une décision (autorisé / non autorisé) et suivre le nombre de places disponibles »	
Objectif	<ul style="list-style-type: none"> ❖ Prendre une décision (autorisé / non autorisé) ❖ Suivre le nombre de places disponibles
Acteur	SID
Pré-conditions	<ul style="list-style-type: none"> ❖ Le système de détection est en fonctionnement (caméra, réseau, base de données Firebase). ❖ Un véhicule se présente à l'entrée ou à la sortie
Scénario nominal	<ul style="list-style-type: none"> • Le système détecte une plaque via caméra + YOLO • Il extrait le texte via OCR • Il vérifie si la plaque est autorisée dans Firebase • Si autorisée : <ul style="list-style-type: none"> ✓ Affiche un message personnalisé ✓ Ouvre la barrière (servo moteur) ✓ Active la LED verte • Si non autorisée :

	<ul style="list-style-type: none"> ✓ Affiche un message d'erreur ✓ Active le buzzer ✓ Active la LED rouge • Il met à jour le nombre de places disponibles selon l'événement (entrée/sortie) • Il enregistre l'événement dans Firebase
Scénario alternative	<ul style="list-style-type: none"> • Erreur de connexion Firebase : affiche « Échec de chargement des données »
Post-conditions	<ul style="list-style-type: none"> ❖ La décision est prise (autorisé/refusé) ❖ Le nombre de places disponibles est mis à jour L'événement est enregistré

Tableau 19: Description textuelle de cas d'utilisation Prendre une décision (autorisé / non autorisé) et suivre le nombre de places disponibles

Cas d'utilisation « Consulter l'historique des véhicules et Accéder au tableau de bord »	
Objectif	<ul style="list-style-type: none"> ❖ Suivre l'activité du parking
Acteur	Administrateur
Pré-conditions	<ul style="list-style-type: none"> ❖ L'administrateur est connecté avec succès. ❖ Le système a accès aux données enregistrées dans Firebase (historique, état du parking).
Scénario nominal	<ul style="list-style-type: none"> • L'administrateur accède à son espace personnel via l'application mobile. • Le système principal se connecte à Firebase et récupère en temps réel les données suivantes au tableau de bord avec une mise à jour dynamique: <ul style="list-style-type: none"> ✓ Nombre total de places du parking. ✓ Nombre de places libres et occupées actuellement. ✓ Liste des plaques détectées aujourd'hui avec leurs heures d'entrée/sortie. • L'administrateur peut visualiser les données en temps réel. • L'administrateur clique sur la section « Historique ». • Le système interroge la base de données Firebase et affiche la liste des événements enregistrés.

	<ul style="list-style-type: none"> Le tableau de l'historique contient les colonnes suivantes: <ul style="list-style-type: none"> ✓ Plaque ✓ Date et heure (Entrée / Sortie) ✓ Statut (non Autorisé) Depuis cette interface, l'administrateur peut: Supprimer un ou plusieurs événements de l'historique.
Scénario alternative	<ul style="list-style-type: none"> Échec de connexion au système (Firebase) :Message : «Échec de chargement des données » → page vide.
Post-conditions	<ul style="list-style-type: none"> ❖ Le tableau de bord, Historique sont affichées avec des données actualisées en temps réel

Tableau 20: Description textuelle de cas d'utilisation Consulter l'historique des véhicules et Accéder au tableau de bord

Cas d'utilisation « Afficher un message via écran LCD et Ouvrir/Fermer la barrière avec l'activation de LED verte/rouge et le buzzer »	
Objectif	<ul style="list-style-type: none"> ❖ Afficher un message via écran LCD ❖ Ouvrir/Fermer la barrière ❖ Activer la LED verte/rouge et le buzzer
Acteur	SCB
Pré-conditions	<ul style="list-style-type: none"> ❖ Le système intelligent de détection a pris une décision (autorisé / refusé) ❖ Communication établie avec Firebase
Scénario nominal	<ul style="list-style-type: none"> Le système reçoit la décision du système principal (via Firebase) Si autorisé : <ul style="list-style-type: none"> ✓ Affiche « Bienvenue M.X » sur l'écran LCD ✓ Active la LED verte ✓ Ouvre la barrière (servo moteur) Si non autorisé : <ul style="list-style-type: none"> ✓ Affiche « Accès refusé » ✓ Active la LED rouge ✓ Active le buzzer pendant quelques secondes ✓ Barrière reste fermée

Scénario alternative	<ul style="list-style-type: none"> • Erreur de connexion Firebase : le système s'affiche « Échec de chargement des données »
Post-conditions	<ul style="list-style-type: none"> ❖ Message est affichée ❖ Les Composants physiques (barrière, LEDs, buzzer) sont activés selon la décision

Tableau 21: Description textuelle de cas d'utilisation Afficher un message via écran LCD

3. Configuration de l'environnement de développement

3.1. Installation du Raspberry Pi OS avec Raspberry Pi Imager

La figure suivant illustre le téléchargement de logiciel Raspberry Pi Imager

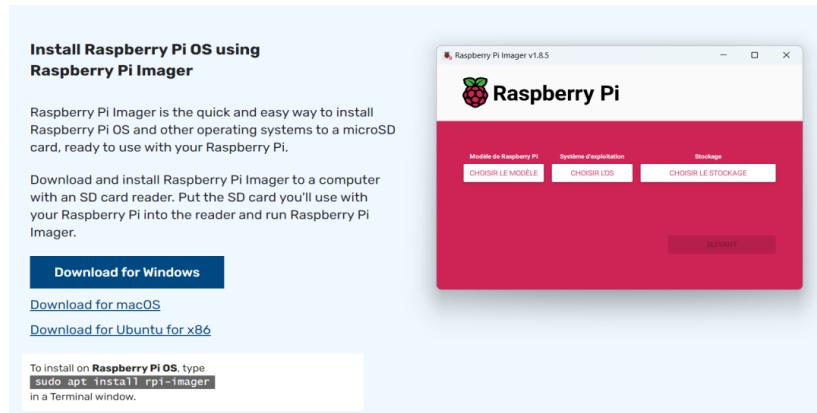


Figure 40: Installation du fois Raspberry Pi Imager

Une fois Raspberry Pi Imager installé et lancé, nous avons procédé à la sélection du modèle de carte Raspberry Pi utilisé (ici, le modèle Raspberry Pi 4). Ensuite, nous avons choisi le système d'exploitation adéquat, à savoir la version 64 bits de Raspberry Pi OS. Après avoir vérifié les options, le système a été automatiquement téléchargé et transféré sur une carte SD via une simple pression sur le bouton « CONTINUER ». Cette étape permet de préparer la carte SD pour le démarrage et le fonctionnement du Raspberry Pi

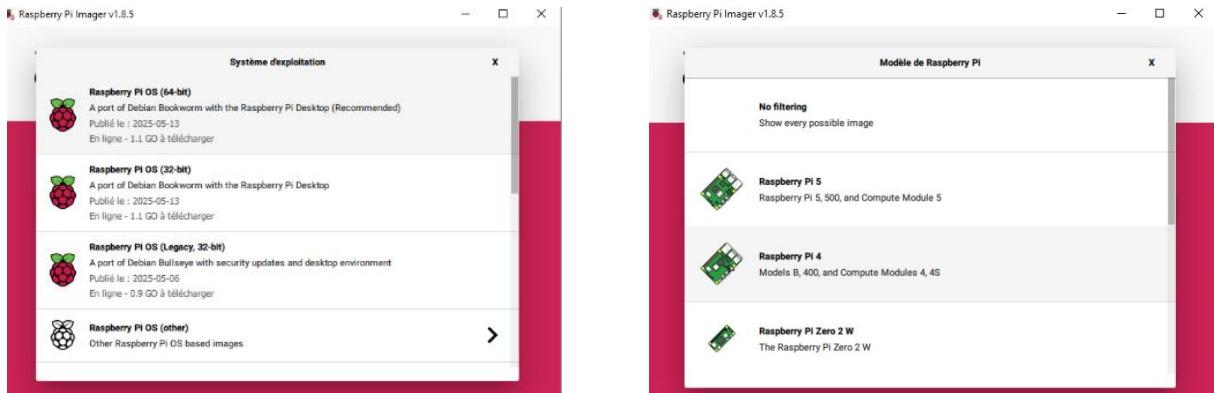


Figure 41: choix de modèle et OS

3.2. Créer l'environnement virtuel sur la carte Raspberry pi

Lors du déploiement du programme sur la carte Raspberry Pi, la création d'un environnement virtuel Python est une étape essentielle permettant d'isoler les dépendances nécessaires au projet et d'éviter les conflits avec les bibliothèques du système. Cette procédure consiste à générer un environnement spécifique au projet à l'aide de la commande **python3 -m venv venv_parking**, puis à l'activer via **source venv_parking/bin/activate**. Une fois activé, l'environnement est prêt à recevoir les bibliothèques indispensables à l'exécution du programme, telles que ultralytics pour l'utilisation de **YOLO**, **opencv-python** pour le traitement d'image, **cvzone** pour les outils de vision par ordinateur, **firebase-admin** pour la communication avec la base de données Firebase, et **EasyOCR** pour la lecture automatique des plaques d'immatriculation. Ces bibliothèques sont installées via la commande **pip install** suivie du nom du paquet. Après l'installation, des tests sont effectués pour s'assurer que tous les modules sont correctement importés et fonctionnels.

La figure suivante illustre l'activation de l'environnement virtuel et l'affichage d'informations sur certaines bibliothèques installées dans le terminal de la Raspberry Pi :

```

raspberry@pfe:~$ cd /home/raspberry/projet_parking
raspberry@pfe:~/projet_parking$ source venv_parking/bin/activate
(venv_parking) raspberry@pfe:~/projet_parking$ pip show easyocr cvzone
Name: easyocr
Version: 1.7.2
Summary: End-to-End Multi-Lingual Optical Character Recognition (OCR) Solution
Home-page: https://github.com/jaieddai/easyocr
Author: Rakpong Kittinaradorn
Author-email: r.kittinaradorn@gmail.com
License: Apache License 2.0
Location: /home/raspberry/projet_parking/venv_parking/lib/python3.11/site-packages
Requires: ninja, numpy, opencv-python-headless, Pillow, pyclipper, python-bidi, PyYAML, scikit-image, scipy, Shapely, torch, torchvision
Required-by:
...
Name: cvzone
Version: 1.6.1
Summary: Computer Vision Helping Library
Home-page: https://github.com/cvzone/cvzone.git
Author: Computer Vision Zone
Author-email: contact@computervision.zone
License: MIT
Location: /home/raspberry/projet_parking/venv_parking/lib/python3.11/site-packages
Requires: numpy, opencv-python
Required-by:
(venv_parking) raspberry@pfe:~/projet_parking$ 

```

Figure 42: terminal de la Raspberry Pi

3.3. Configuration l'environnement de développement de la carte ESP32

La configuration de l'environnement de développement de la carte ESP32 a été réalisée à l'aide de l'IDE Arduino. Après avoir installé l'IDE, le support de la carte ESP32 a été ajouté via le gestionnaire de cartes en utilisant l'URL d'Espressif. Une fois la carte ESP32 sélectionnée et connectée, les bibliothèques nécessaires telles que LiquidCrystal_I2C, ESP32Servo, WiFi et Firebase ESP Client ont été installées pour permettre le fonctionnement des composants physiques (LCD, servo, LEDs, buzzer) et la communication avec Firebase. Enfin, le code a été téléchargé sur la carte, permettant le contrôle physique du système de parking intelligent.

La figure suivante représente la configuration des préférences de l'IDE Arduino, notamment l'ajout de l'URL pour la gestion des cartes ESP32 :

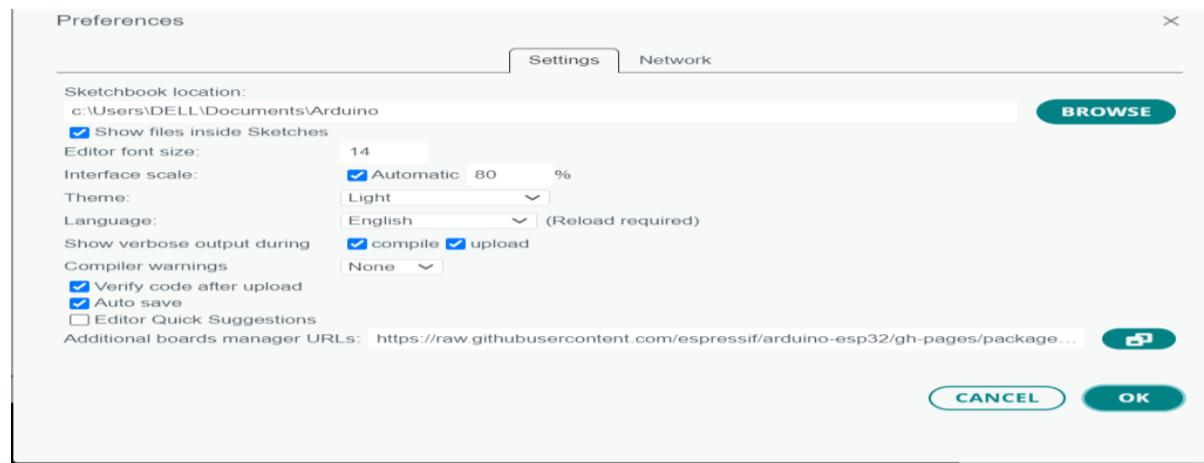


Figure 43: préférences de l'IDE Arduino

4. Diagramme de Bloc

Un diagramme de blocage est une représentation graphique simplifiée d'un système complexe. Il montre les composants principaux (ou blocs) du système, les relations entre eux, ainsi que la circulation des données ou signaux. Ce type de diagramme est très utilisé dans les domaines de l'électronique, de l'automatique, de l'informatique embarquée, et de la modélisation de systèmes.

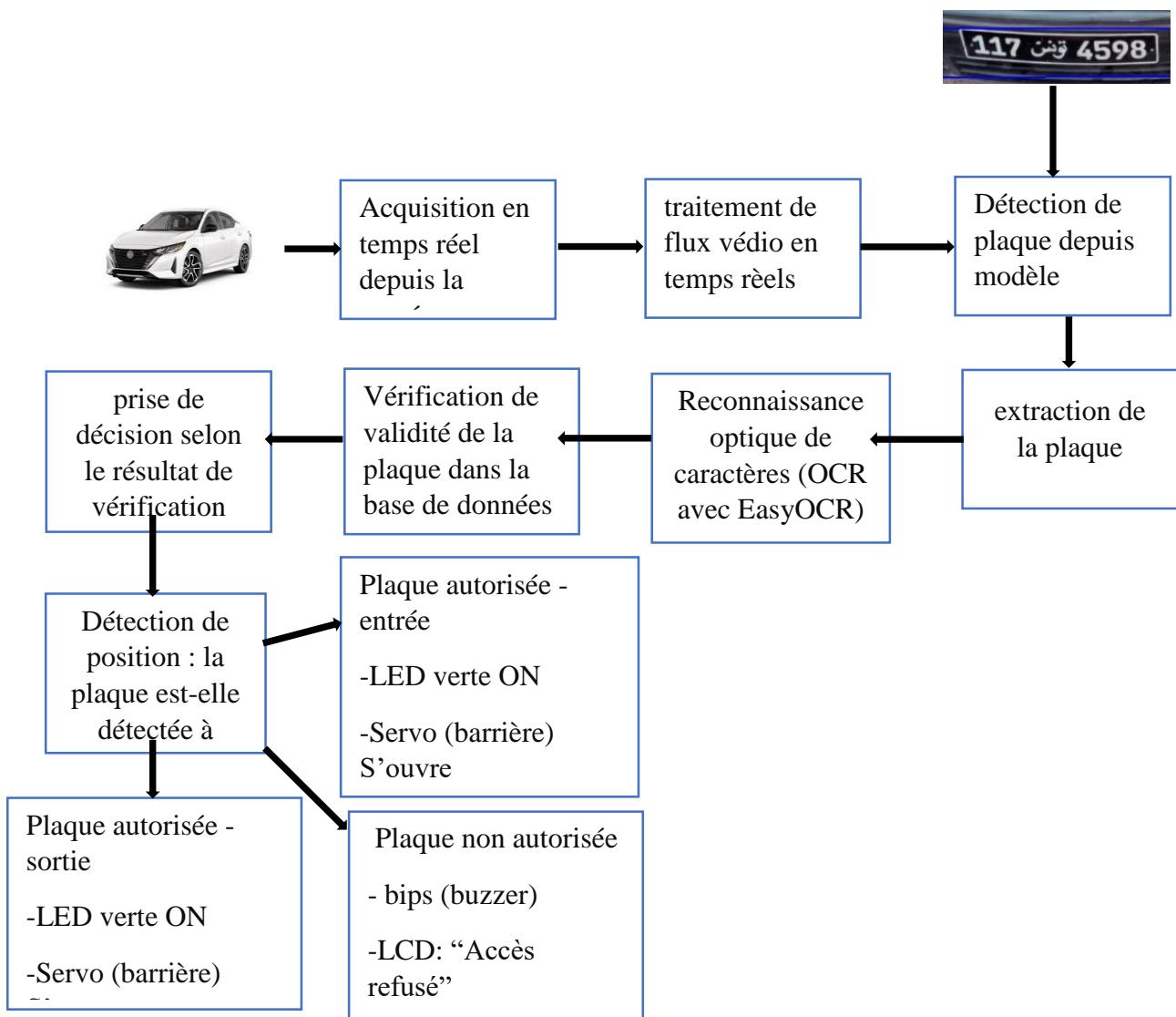


Figure 44: : Diagramme de Bloc

Ce diagramme de bloc résume le fonctionnement d'un système automatique de reconnaissance de plaques d'immatriculation : une caméra capte le flux vidéo en temps réel, un modèle détecte les plaques, puis EasyOCR extrait le texte. Ce texte est comparé à une base de données pour autoriser ou refuser l'accès. Selon le résultat, le système contrôle la barrière, les LEDs, le buzzer et l'affichage sur écran LCD, garantissant sécurité et automatisation sans intervention humaine

5. Algorithme de programmation pour traitement d'image et reconnaissance de plaque

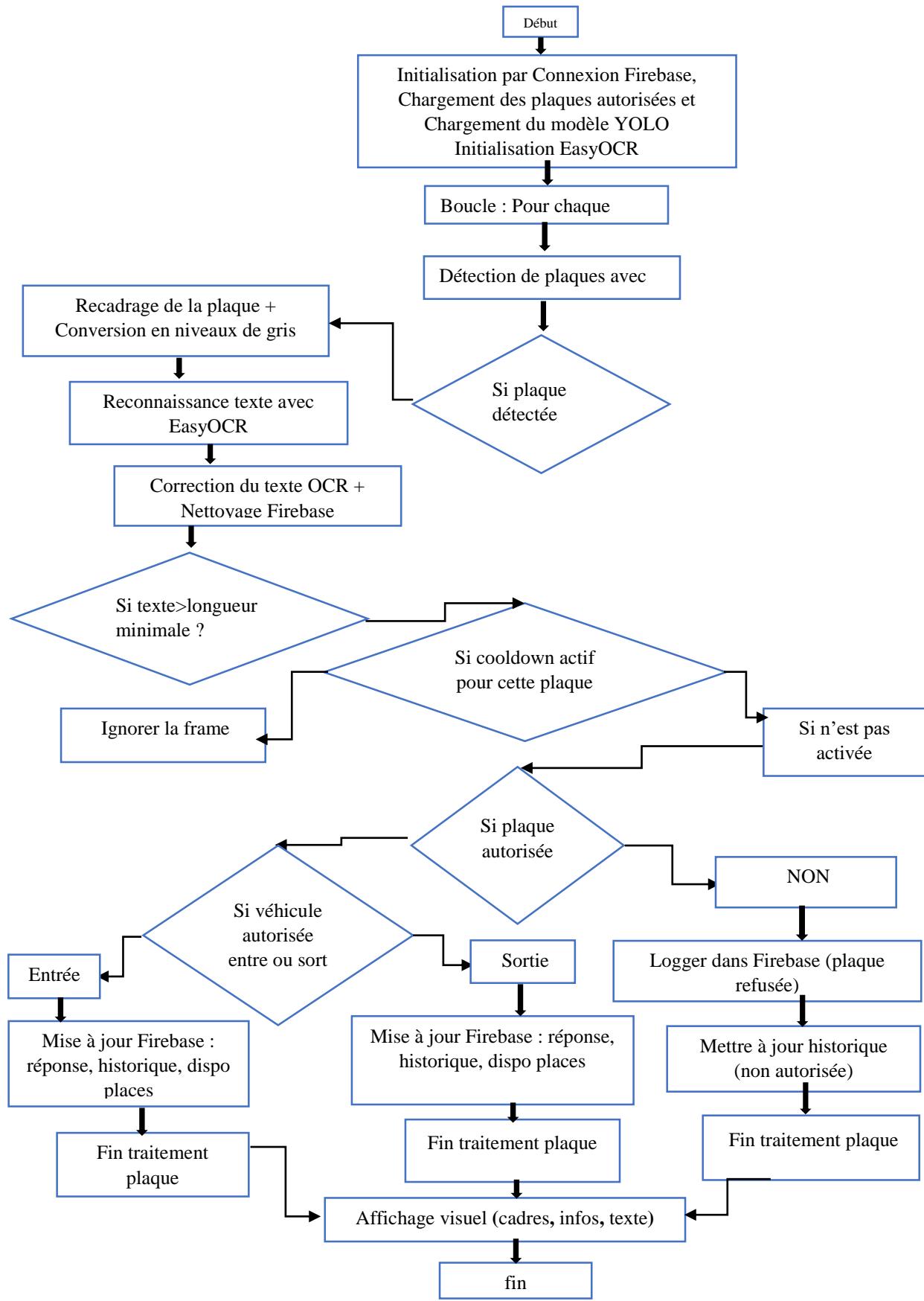


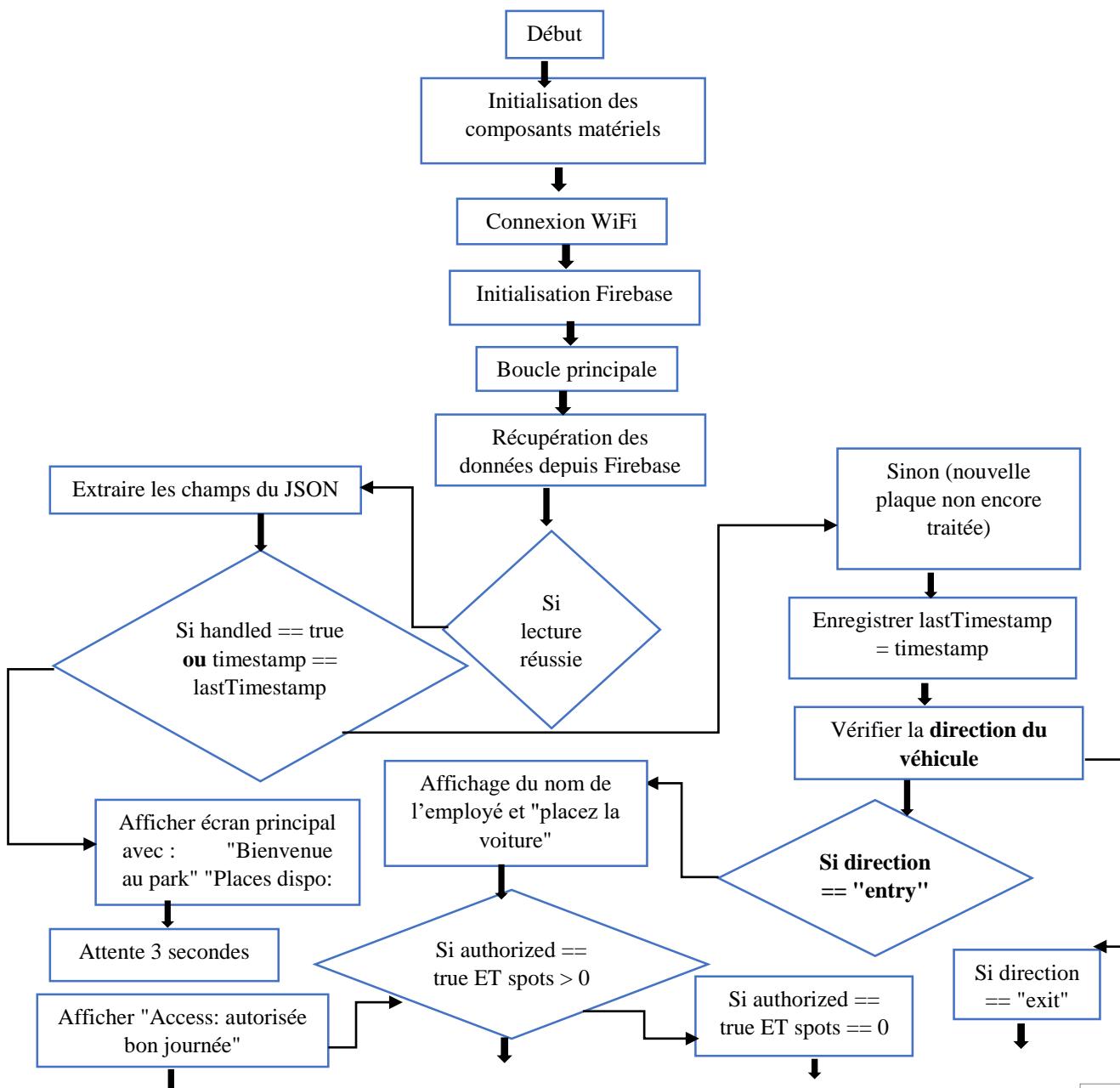
Figure 45: Organigramme d'Algorithme de Reconnaissance

Début	L'algorithme commence.
	Administrateur
Initialisation	<ul style="list-style-type: none"> ➤ Chargement du modèle YOLO pour la détection des plaques. ➤ Connexion à la base de données Firebase. ➤ Chargement des plaques autorisées depuis Firebase. ➤ Initialisation du module EasyOCR ➤ Ouverture du flux vidéo pour le traitement en temps réel.
Boucle sur chaque image (frame)	<ul style="list-style-type: none"> ➤ L'algorithme traite chaque image du flux vidéo en temps réel. ➤ Le modèle YOLOv8 analyse l'image pour détecter la présence de plaques d'immatriculation.
Vérification de la zone d'entrée	<p>L'algorithme vérifie si la plaque détectée se trouve dans la zone définie d'entrée.</p> <p>Si la plaque est en dehors de cette zone, elle est ignorée</p>
Recadrage et conversion en niveaux de gris	<ul style="list-style-type: none"> ➤ La région contenant la plaque est extraite de l'image. ➤ L'image de la plaque est convertie en niveaux de gris pour faciliter l'OCR.
Reconnaissance de texte (OCR)	<ul style="list-style-type: none"> ➤ EasyOCR est utilisé pour extraire le texte de la plaque d'immatriculation.
Correction du texte OCR	<ul style="list-style-type: none"> ➤ Des corrections sont appliquées sur le texte pour corriger les erreurs ➤ Nettoyage des caractères non pertinents et standardisation du texte
Vérification du cooldown	<ul style="list-style-type: none"> ➤ L'algorithme vérifie si cette plaque a déjà été récemment traitée (cooldown actif). ➤ Si oui, la détection est ignorée pour éviter les doublons
Comparaison avec la base de données	<ul style="list-style-type: none"> ➤ Le texte reconnu est comparé aux plaques autorisées dans Firebase.
Si la plaque est autorisée	<ul style="list-style-type: none"> ➤ Cas "Entrée" : <ul style="list-style-type: none"> • Mise à jour de Firebase : réponse autorisée, historique mis à jour, nombre de places disponibles décrémenté.

	<ul style="list-style-type: none"> ➤ Cas "Sortie" : • Mise à jour de Firebase : réponse autorisée, historique mis à jour, nombre de places disponibles
Si la plaque n'est pas autorisée	<ul style="list-style-type: none"> ➤ Enregistrement de la tentative dans l'historique. • Mise à jour de Firebase.
Affichage visuel	<ul style="list-style-type: none"> ➤ Des cadres sont affichés autour des plaques détectées avec des messages d'état sur l'écran.
Fin	<ul style="list-style-type: none"> ➤ L'algorithme se termine

Tableau 22: Description de l' Organigramme d'Algorithme de Reconnaissance

6. Algorithme de Programmation du Contrôle d'Accès au Parking



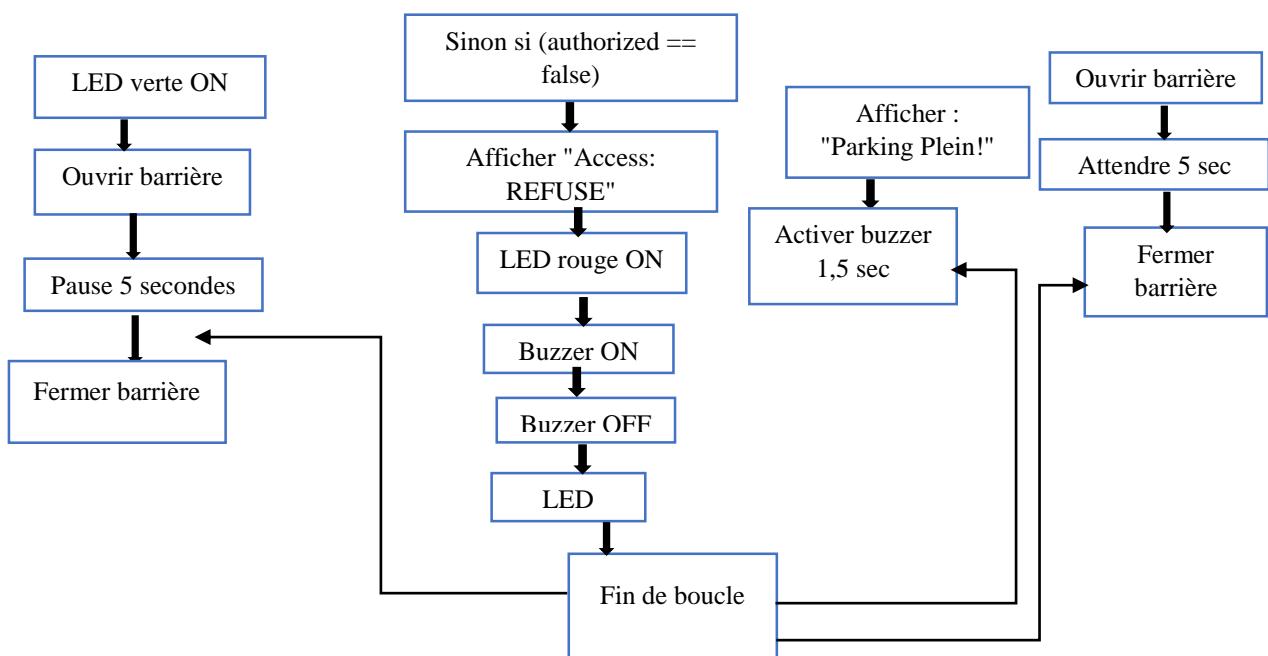


Figure 46: Organigramme d'Algorithme de Contrôle

Début	L'algorithme commence.
Initialisation des composants matériels	➤ L'ESP32 initialise les composants connectés : écran LCD I2C, servo moteur, LEDs, buzzer.
Connexion WiFi	➤ L'ESP32 se connecte au réseau WiFi
Initialisation de Firebase	➤ L'ESP32 se connecte à Firebase pour lire les données des plaques détectées.
Boucle principale	➤ L'algorithme entre dans une boucle infinie pour surveiller les nouvelles plaques détectées.
Récupération des données depuis Firebase	➤ L'ESP32 lit les données JSON de Firebase concernant la plaque détectée.

Si lecture réussie	<ul style="list-style-type: none"> ➤ Si la lecture est correcte, les champs sont extraits : handled, timestamp, authorized, direction, employee_name, spots
Éliminer les caractères non alphanumériques	<ul style="list-style-type: none"> ➤ Les caractères non alphanumériques extraits par l'OCR sont éliminés.
Si handled == true ou timestamp == lastTimestamp	<ul style="list-style-type: none"> ➤ Afficher sur l'écran LCD: <ul style="list-style-type: none"> • « Bienvenue au park » • « Places dispo: X » ➤ Attendre 3 secondes ➤ Repartir au début de la boucle
Sinon (nouvelle plaque non encore traitée)	<ul style="list-style-type: none"> ➤ L'événement est nouveau: <ul style="list-style-type: none"> • Enregistrer lastTimestamp = timestamp • Vérifier la direction du véhicule
Si direction == "entry"	<ul style="list-style-type: none"> ➤ Afficher le nom de l'employé et « placez la voiture » ➤ Le système informe l'utilisateur que le véhicule est détecté.
Si authorized == true ET spots > 0	<ul style="list-style-type: none"> ➤ Afficher sur LCD: « Access: autorisée bon journée » ➤ LED verte ON ➤ Ouvrir la barrière (servo ON) ➤ Pause 5 secondes ➤ Fermer la barrière (servo OFF)
Sinon si authorized == false	<ul style="list-style-type: none"> ➤ Afficher: « Access: REFUSÉ » ➤ LED rouge ON ➤ Activer buzzer pendant 1.5 secondes ➤ Éteindre buzzer ➤ Rester sur LED rouge

Si authorized == true ET spots == 0	<ul style="list-style-type: none"> ➤ Afficher: « Parking Plein! » ➤ Activer buzzer pendant 1.5 secondes
Si direction == "exit"	<ul style="list-style-type: none"> ➤ Ouvrir la barrière (servo ON) ➤ Attendre 5 secondes ➤ Fermer la barrière (servo OFF)
Fin	<ul style="list-style-type: none"> ➤ L'algorithme se termine

Tableau 23: Description d'Algorithme de Contrôle

7. Integration entre Raspberry Pi, ESP32 et Application mobile via Firebase.

Ce schéma illustre l'intégration technique d'un système intelligent de gestion de parking basé sur Firebase. Tous les composants – Raspberry Pi, ESP32 et application mobile – sont connectés via Wi-Fi pour une synchronisation en temps réel. Le Raspberry Pi capture et analyse les images pour détecter les plaques, puis envoie les données (texte, heure, direction) vers Firebase. L'ESP32 lit ces données via Firebase pour exécuter les actions physiques : ouverture de barrière, affichage LCD, activation des LEDs ou du buzzer selon l'autorisation. Firebase sert de plateforme cloud centrale pour stocker les plaques, gérer les accès, l'historique et sécuriser les échanges via Firebase Auth et Security Rules. L'application mobile permet à l'administrateur de superviser et gérer le système à distance, en consultant en temps réel les entrées/sorties, le nombre de places disponibles et l'historique, tout en assurant une connexion sécurisée. Ce système garantit une automatisation complète, une sécurité renforcée et un contrôle distant efficace.

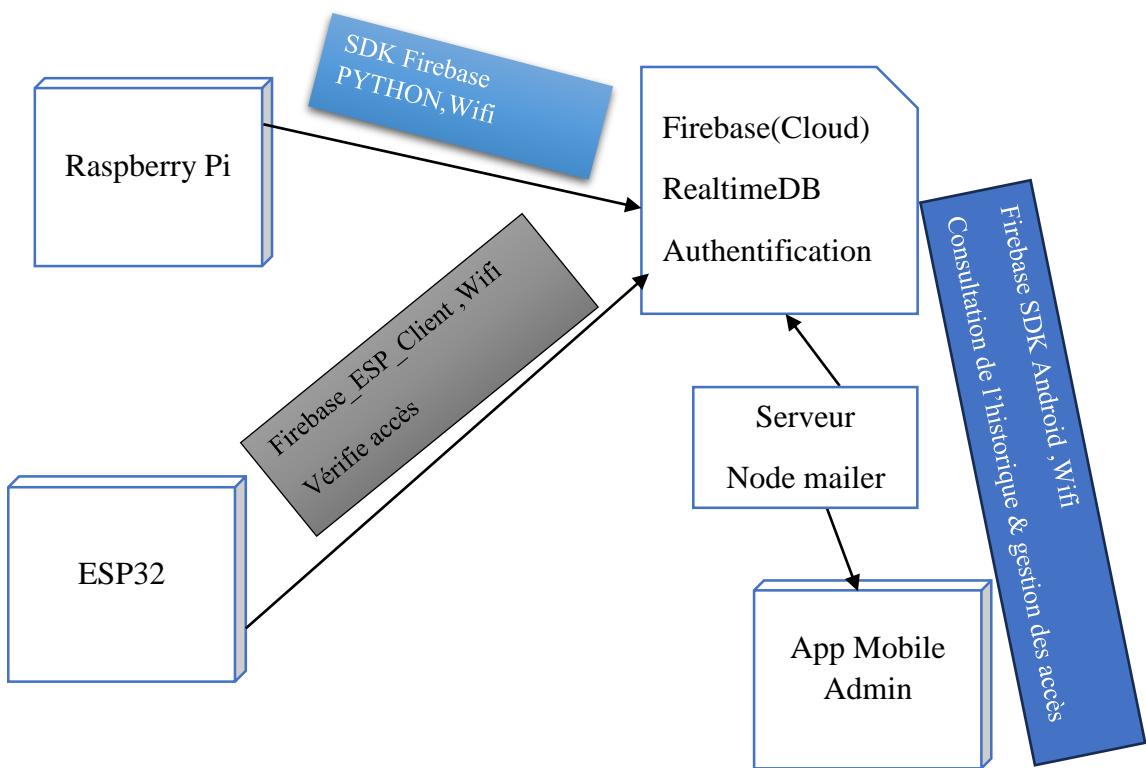


Figure 47: Organigramme Architecture d'intégration

8. tests et Réalisation

Après le développement et l'optimisation des algorithmes du système, ainsi que le déploiement sur la carte Raspberry Pi 4, nous avons procédé à un **test d'intégration complet** entre les différents composants : **ESP32**, **Raspberry Pi**, **base de données Firebase**, et **application mobile**.

Ces tests ont permis de vérifier :

- La communication en temps réel entre les modules,
- La détection automatique des véhicules à l'entrée et à la sortie,
- La gestion de l'ouverture de la barrière,
- La mise à jour dynamique du nombre de places disponibles,
- La consultation de l'historique des plaques détectées dans l'application mobile.

Divers scénarios ont été simulés pour garantir le bon fonctionnement global du système dans différentes conditions.

Les figures suivantes illustrent les interfaces réalisées et les prototypes mis en œuvre :

Cette figure montre l'exécution du script Python sur Thonny IDE sur Raspberry Pi 4, avec détection de la plaque d'immatriculation, affichage du nombre de places disponibles (3/4), et enregistrement dans Firebase via le fichier serviceAccountKey.json.

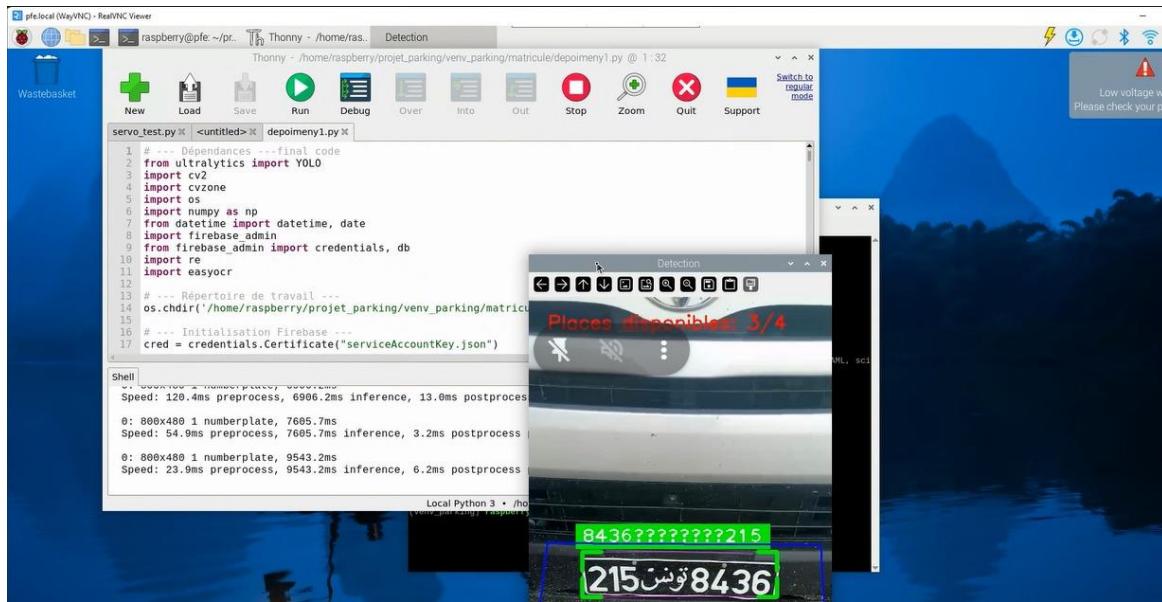


Figure 48: Interface de détection de plaque à l'entrée du parking

Cette figure illustre l'exécution du script sur la carte Raspberry Pi, montrant la détection automatique de la sortie d'un véhicule via la reconnaissance de plaque. À la suite de cette détection, le système incrémentera le nombre de places disponibles dans le parking et effectue une mise à jour en temps réel de la base de données Firebase.

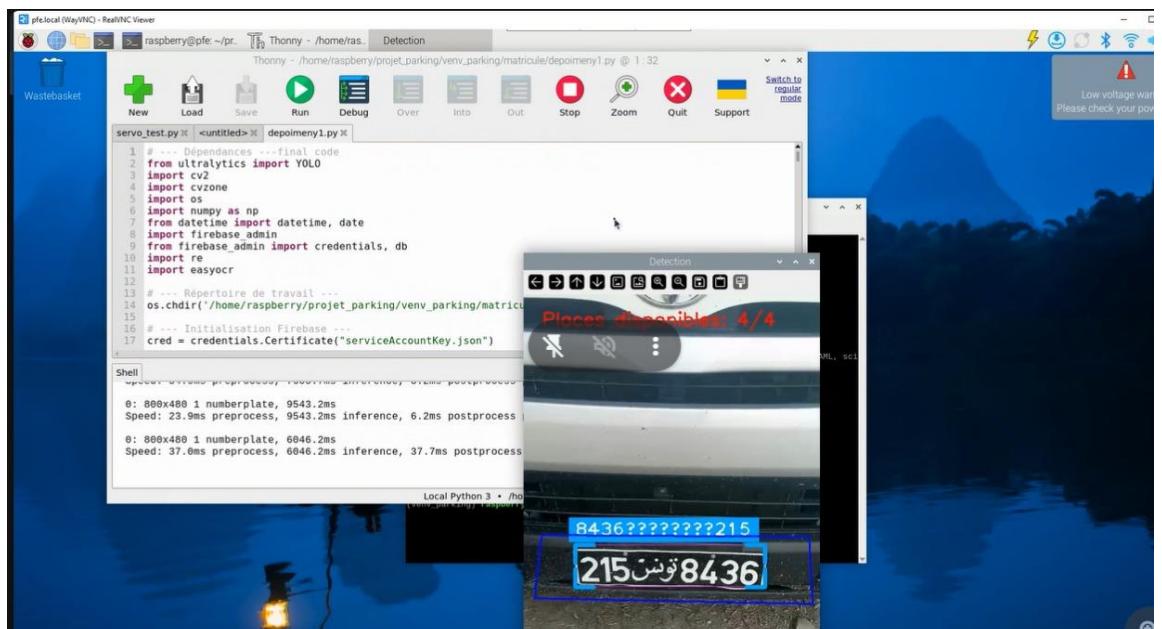


Figure 49: Séquence de détection de la sortie du véhicule

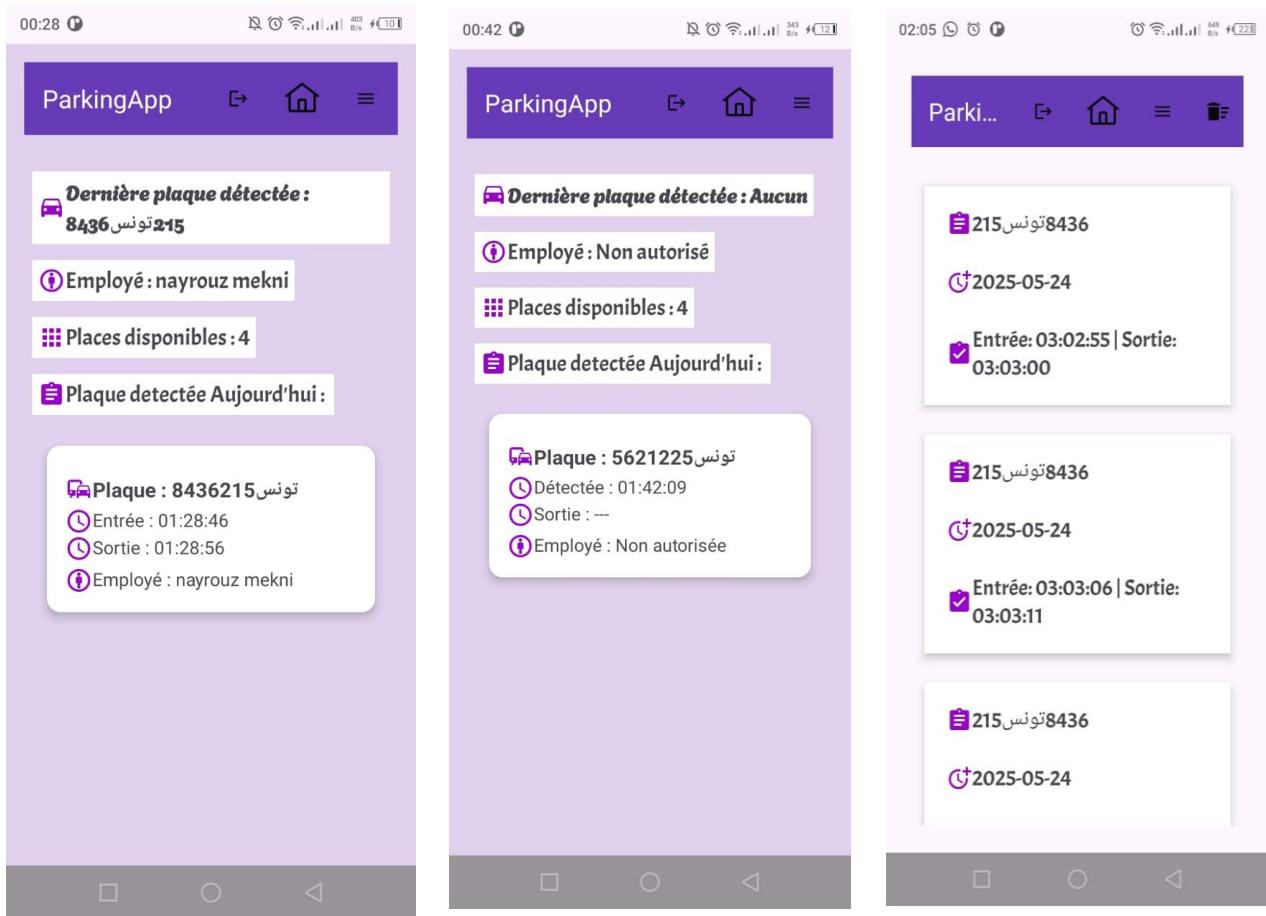


Figure 50: Interface Tableau de bord et Historique



Figure 51: Prototype réel de contrôle de barrière

Figure 52: Structure finale de la base de données Firebase

Conclusion

Durant ce chapitre, nous avons tout d'abord organisé notre backlog de sprint afin de définir les user stories à implémenter dans le sprint 2. Ensuite, nous avons conçu le diagramme de cas d'utilisation correspondant aux fonctionnalités ciblées. Nous avons également détaillé les différentes étapes techniques nécessaires à l'avancement du projet, allant de la configuration

de l'environnement jusqu'à l'intégration des modules. Enfin, la partie réalisation a permis de mettre en valeur les résultats obtenus à travers une série de captures d'écran illustrant le fonctionnement global du système.

Cette étape marque ainsi la clôture du sprint 2, avec une version fonctionnelle du système conforme aux objectifs fixés

Conclusion Générale et Perspectives

Ce projet de fin d'études, réalisé au sein de l'entreprise **Kromberg & Schubert**, avait pour objectif principal la mise en place d'un système intelligent de gestion d'accès au parking, basé sur la reconnaissance automatique des plaques d'immatriculation à l'aide de caméras de surveillance.

L'approche adoptée, fondée sur la méthodologie **Scrum**, nous a permis d'organiser efficacement les différentes étapes du développement, depuis l'analyse des besoins jusqu'à la conception, l'implémentation, les tests fonctionnels, ainsi que le déploiement sur **matériel embarqué**.

Nous avons débuté par une étude approfondie de l'existant et de l'environnement de travail, ce qui nous a permis de mieux cerner les contraintes et de proposer une solution adaptée. Nous avons ensuite modélisé l'ensemble du système à l'aide des langages **UML** et **SYSML**, entraîné le modèle d'intelligence artificielle **YOLO** pour la détection des plaques, conçu les algorithmes nécessaires, et développé l'interfaçage avec **Firebase**, ainsi qu'une application mobile pour la **gestion et la visualisation en temps réel**.

Une partie essentielle de notre travail a porté sur l'intégration matérielle, notamment via l'utilisation de la **carte Raspberry Pi 4**, dédiée au traitement d'image et à l'exécution des modèles de détection, et de la **carte ESP32**, utilisée pour le contrôle d'accès (barrière, capteurs infrarouges, écran LCD, etc.). Ce déploiement embarqué nous a permis de concevoir une solution complète, intelligente et autonome, parfaitement adaptée à un environnement industriel.

Ce projet nous a permis de consolider nos compétences techniques, notamment en **vision par ordinateur**, en **intelligence artificielle embarquée**, en **développement mobile**, ainsi qu'en **systèmes IoT**. Il constitue une base solide pour la mise en œuvre future de solutions industrielles intelligentes.

Pour une évolution future du système, plusieurs améliorations peuvent être envisagées :

- **Intégration d'un module de détection d'incidents** (accidents, incendies, etc.) dans les zones critiques du parking.

- **Détection plus précise des zones d'entrée et de sortie**, afin d'optimiser la gestion du flux de véhicules et d'améliorer la fiabilité du comptage.
- **Amélioration de l'interface mobile**, avec un **système de gestion des alertes en temps réel** (notifications, journalisation, priorisation).

Nous espérons que ce travail contribuera à fournir une **solution durable, intelligente et évolutive** pour la gestion du parking de l'entreprise **Kromberg & Schubert**, et nous remercions sincèrement tous les encadrants et responsables qui nous ont accompagnés tout au long de ce projet.

Webographie

[1] Consulté le 15/02/2025

<https://lapresse.tn/2025/01/12/tunisie-kromberg-schubert-lusine-de-beja-se-developpe-et-genere-plus-de-8000-emplois-en-2026>

[2] Consulté le 17/02/2025

<https://www.atlassian.com/fr/agile/scrum/roles>

[3] Consulté le 22/02/2025

<https://www.lucidchart.com/pages/fr/langage-uml>

[4] Consulté le 22/02/2025

<https://www.omg.sysml.org/LMO08-vfinale-BelloirEtAl.pdf>

[5] Consulté le 22/02/2025

https://www.memoireonline.com/11/12/6484/m_Conception-et-realisation-dune-application-de-gestion-des-marches-par-appel-doffres-au-sein6.html

[6] Consulté le 01/03/2025

<https://www.ionos.fr/digitalguide/sites-internet/developpement-web/diagramme-de-cas-de-utilisation>

[7] Consulté le 10/04/2025

<https://components101.com/displays/16x2-lcd-module>

[8] Consulté le 10/04/2025

<https://learn.sparkfun.com/tutorials/light-emitting-diodes-leds/all>

[9] Consulté le 10/04/2025

<https://www.raspberrypi.com/documentation/accessories/camera.html>

[10] Consulté le 10/04/2025

<https://components101.com/buzzer-module>

[11] Consulté le 15/04/2025

<https://opencv.org>

[12] Consulté le 15/04/2025

<https://developer.android.com/studio>

[13] Consulté le 15/04/2025

<https://www.python.org>

[14] Consulté le 15/04/2025

<https://docs.ultralytics.com/fr>

[15] Consulté le 15/04/2025

<https://www.geeksforgeeks.org/python-easyocr-module/>

[16] Consulté le 15/04/2025

<https://firebase.google.com>

[17] Consulté le 15/04/2025

<https://code.visualstudio.com>

[18] Consulté le 15/02/2025

<https://github.com/cvzone/cvzone>

[19] Consulté le 15/02/2025

<https://nodemailer.com/about/>

[20] Consulté le 15/02/2025

<https://colab.research.google.com>

[21] Consulté le 15/04/2025

<https://www.arduino.cc/en/software>

[22] Consulté le 15/02/2025

<https://www.modelio.org/documentation.html>

[23] Consulté le 10/05/2025

<https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-class-diagram/>