REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE

MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE

UNIVERSITE MOULOUDMAMMERI, TIZI-OUZOU

FACULTE DE GENIE ELECTRIQUE ET DE L'INFORMATIQUE

DEPARTEMENT D'ELECTRONIQUE

Mémoire de fin d'études
De MASTER ACADEMIQUE

Domaine : **Science et Technologie**

Filière: **Electronique**

Spécialité: **Electronique des systèmes embarques**

Présenté par :

MIMOUN Sarah

SLIMANI Amel

Thème :

Réalisation et conception d'un système de gestion d'un parking via une application mobile

Mémoire soutenu publiquement le 03/07/2025. Devant le jury composé de:

Mr ATTAF Youcef ..... UMMTO, Président

Mr OUALLOUCHE Fethi.....UMMTO, Encadreur

Mr LAZRI Mourad..... UMMTO, Examinateur

Année universitaire : **2024/ 2025**

Remerciements

Nous remercions tout d'abord Allah tout puissant qui nous a donné la santé, le courage et la patience afin de pouvoir accomplir ce modeste travail.

Nous tenons à présenter nos remerciements à ***Mr OUALLOUCHE.F,*** *notre promoteur pour sa précieuse aide, ses orientations et le temps qu'il nous a accordé et d'avoir accepté l'encadrement de ce mémoire.*

Nos remerciements vont aussi aux membres du jury qui nous feront l'honneur de juger notre travail.

Enfin nous n'oublions pas nos enseignants qui ont supervisé notre enseignement tout au long du cycle d'étude.

Dédicace

A mon très cher père

Autant de phrases et d'expressions aussi éloquentes sont-elles ne sauraient exprimer ma gratitude et
ma reconnaissance. Tu as su m'inculquer le sens de la responsabilité, de l'optimisme et de la confiance en soi face aux difficultés de la vie. Tes conseils ont toujours guidé mes pas vers la réussite.

Ta patience sans fin, ta compréhension et ton encouragement sont pour moi le soutien indispensable que t'as toujours su m'apporter. Je te dois ce que je suis aujourd'hui et ce que je serai demain et je
ferai de mon mieux pour rester ta fierté et ne jamais te décevoir. Qu'ALLAH le tout puissant te préserve, t'accorde Santé, bonheur et te protège de tout mal, Je t'aime papa

A ma très chère mère

*A ma mère**,** ma force, mon rayon du soleil et ma source de vie d'amour et d'affection qui m'a appris le sens de responsabilité, à qui je dois la réussite, pour l'éducation qu'elle m'a offerte, avec tous les*
moyens et au prix de tous les sacrifices qu'elle a consentis à mon égard, pour le sens du devoir qu'elle

m'a enseigné depuis mon enfance. Je te remercie pour la femme forte que t'es. Je te dédie ce modeste travail. Qu'ALLAH te protège et te donne la santé, le bonheur et longue vie, Je t'aime maman

***A mon très cher** adorable **frère Lyes** mon pilier, mon ange gradient et mon fidèle compagnon dans les moments les plus délicats de cette vie mystérieuse et **ma grande sœur** d'amour **Tina** la prunelle de mes*
yeux Une sœur comme on ne peut trouver nulle part ailleurs. Que dieu vous garde pour nous. Je vous aime très fort

***A ma grande mère chérie** Qui m'a accompagné par ses prières, sa douceur, puisse Dieu lui prêter longue Vie et beaucoup de santé et de bonheur dans les deux vies.*

***A la mémoire de mes grand- père et ma grande mère paternelle,** J'aurais tant aimé que vous soyez présents. Que Dieu ait vos âmes dans sa sainte miséricorde*

À mes chères tantes,

Veuillez trouver dans ce travail l'expression de mon respect le plus profond et Mon affection la plus Sincère

À mes cousines chéries,

Merci d'être toujours là, avec vos rires, votre amour et votre folie. La famille c'est précieux et vous en êtes la plus belle preuve. Je vous aime fort

Et a tous mes amis(es) merci d'être vous-même je vous aime.

SARAH

Dédicace

À mes chers parents,

Je dédie ce travail de tout cœur à mes parents, mes piliers, mes modèles, mes premiers soutiens et ma plus grande force. Merci pour votre présence, votre amour inconditionnel, votre aide précieuse, et pour avoir toujours cru en moi. Que Dieu vous protège.

À mes chers frères et sœurs,

*À ma grande sœur, **Sabrina**, et à mon grand frère, **Ali**, qui m'ont toujours soutenue tout au long de mon parcours et m'ont donné l'espoir d'avancer sans jamais abandonner.*

*À mon petit frère, **Anis**, et à ma petite sœur, **Imen**, qui m'ont appris ce que signifie être une grande sœur et qui m'ont donné une belle raison de me dépasser chaque jour.*

À mes chers oncles, tantes,

*Un grand merci à mes **tantes et oncles**, pour leur gentillesse, leurs encouragements et leur présence bienveillante.*
*Une pensée toute particulière à **mon oncle Noureddine**, pour son soutien unique et sa confiance inébranlable.*

*À **un ami spécial**, même si la distance nous sépare, tu m'as encouragée à prendre mes propres décisions, celles qui peuvent changer mon avenir pour le meilleur. Grâce à toi, j'ai découvert une nouvelle manière d'être curieuse, ouverte, et d'embrasser l'apprentissage avec passion.*

*À mon ancienne école, **l'ENST**, qui m'a parfois fait souffrir, mais qui m'a aussi offert de précieuses expériences, de belles amitiés et des souvenirs impérissables. Elle restera gravée dans ma mémoire, avec ses hauts et ses bas, comme une étape essentielle de mon parcours.*

AMEL

Sommaire :

Table des figures :

Liste des tableaux :

Liste des abréviations :

QR	Quick Response (Réponse rapide)
IoT	Internet of Things (Internet des objets)
RAM	Random Access Memory
PWM	Pulse Width Modulation
OTP	One-Time Programmable
FRAM	Ferroelectric Random Access Memory
USB	Universal Serial Bus
ROM	Read-Only Memory
SRAM	Static Random Access Memory
RTC	Real-Time Clock
SPI	Serial Peripheral Interface
SoC	System on Chip
BLE	Bluetooth Low Energy
GND	Ground
VCC	Voltage Common Collector
IR	Infrared
LED	Light Emitting Diode
I2C	Inter-Integrated Circuit
IDE	Integrated Development Environment
RX	Receive
TX	Transmit
MIT	Massachusetts Institute of Technology
PCB	Printed Circuit Board
HTTPS	HyperText Transfer Protocol Secure
MAJ	Mise à jour
CVV	Card Verification Value
CIB	Carte Interbancaire
SATIM	Société d'Automatisation des Transactions Interbancaires et de Monétique

Introduction générale

Avec l'augmentation de la population mondiale, de plus en plus de personnes vivent en ville. Cela pose plusieurs défis, et les villes doivent s'adapter en améliorant leurs infrastructures et leurs services pour mieux accueillir leurs habitants. Il existe également une tendance croissante dans les applications de la ville intelligente qui peuvent aider à améliorer pour réduire les problèmes des villes. Dans les villes intelligentes nous trouvons beaucoup de difficultés et des obstacles à surmonter notamment la demande de places de stationnement qui dépasse l'offre disponible, les usagers se retrouvent parfois dans l'obligation de garer de façon anarchique, ce qui aggrave la circulation et provoque des embouteillages en ville. L'un des plus importants facteurs pour régler cette situation c'est le parking. Avec l'urbanisation croissante et l'augmentation continue du nombre de véhicules en circulation, la recherche d'une place de stationnement disponible en milieu urbain est devenue un véritable défi quotidien pour de nombreux automobilistes. Ce problème engendre non seulement une perte considérable de temps, mais également une consommation excessive de carburant, contribuant ainsi à l'augmentation de la pollution et du stress des usagers. Dans ce contexte, le concept de ville intelligente (Smart City) [1] émerge comme une solution globale visant à améliorer la qualité de vie des citoyens à travers l'intégration des technologies de l'information et de la communication dans la gestion des infrastructures urbaines. Le stationnement intelligent constitue l'un des piliers de cette transformation, en proposant des solutions innovantes pour optimiser l'utilisation des espaces disponibles et fluidifier la circulation. Le présent projet s'inscrit pleinement dans cette démarche. Il porte sur la conception et la réalisation d'un système de gestion de parking intelligent, basé sur un microcontrôleur ESP32 [2], reconnu pour ses capacités de connectivité sans fil et ses performances adaptées aux systèmes embarqués. L'objectif est de développer une solution permettant de détecter en temps réel les places de stationnement disponibles et d'informer les conducteurs via un affichage local et une application mobile Android. Ce système vise à réduire significativement le temps de recherche d'un emplacement libre, tout en contribuant à une gestion plus efficace des parkings urbains.

Dans le présent mémoire, nous avons commencé par une introduction générale, suivie de trois chapitres :

- **Le premier chapitre :** consiste à donner une idée générale sur les parkings intelligents, leurs définitions, leurs types, leurs avantages et inconvénients.
- **Le deuxième chapitre :** présente la description du fonctionnement du système et les outils matériels et logiciels utilisés.

- **Le troisième chapitre :** Nous présenterons les différentes étapes à suivre afin d'avoir un système fonctionnel. De plus, nous allons mettre en évidence les différents tests et les résultats obtenus.

Enfin nous terminons notre mémoire avec une conclusion qui résume notre travail et montre ses limites puis donne des perspectives pour des travaux futurs.

Chapitre I :Généralités sur les parkings

I. Introduction :

L'augmentation importante du nombre de voiture qui circule dans la route surtout en ville entraine des embouteillages. Beaucoup de voiture sont mal garés, les parkings sont mal gérés. Il faut un nouveau moyen pour minimiser l'espace et assuré des bonnes conditions de stationnement (la sécurité, le temps). Ce qui mène l'idée de la réalisation de ce projet de gestion de parking.

À l'ère de la technologie avancée, les parkings intelligents apparaissent comme une solution moderne pour mieux gérer la mobilité en ville. Ce ne sont plus seulement des endroits pour se garer, mais de vraies solutions pour mieux gérer l'espace et la circulation, en intégrant des technologies avancées telles que des capteurs, des caméras, des logiciels de gestion de données et des communications en temps réel. Ces parkings rendent le stationnement plus facile, améliorent la sécurité, aident à réduire les embouteillages et protègent l'environnement. En résumé, grâce à la technologie les parkings intelligents changent ces espaces essentiels en endroits connectés et intelligents, ils permettent d'améliorer l'organisation des villes et de les rendre plus efficaces.

II. Problématique de stationnement :

Le stationnement n'est pas seulement pour les propriétaires du quartier ou de la ville, mais pour le grand public et la durée de l'arrêt n'est pas spécifique, et c'est ce qui provoque les problèmes. Malgré la présence de parkings il y a beaucoup d'accidents et de problèmes qui se produisent entre les conducteurs.

Parmi ces problèmes de stationnement on peut mentionner :

- **Manque de places disponibles :**
  Dans des lieux ou y'a beaucoup de monde comme les centres ville, les zones commerciales il n y'a souvent pas assez de place de stationnement pour tout le monde. Cela peut par exemple faire perdre du temps aux conducteurs qui tournent pour trouver une place et causer des embouteillages.
- **Mauvaise gestion des lieux existants :**
  Des places libres qui ne sont pas bien utilisés en raison d'un manque d'information en temps réel ou des systèmes pour le guidage. Cela peut entrainer que certaines places soient occupées trop longtemps par des véhicules non autorisés.

- **Problème d'accès :**

  Les entrées et sorties du parking peuvent des fois prendre du temps (lentes) a cause d'une mauvaise organisation. Cela peut parfois saturer les points d'accès.

Pour réduire ces problèmes, le système de stationnement intelligent est une solution efficace, car il aide les touristes, les employés ou toute personne ne connaissant pas la ville à réserver une place de stationnement facilement et de manière fiable.

III. Définition d'un parking :

Un parking ou bien un parc de stationnement, est un espace aménagé en plein air ou dans un bâtiment destiné au stationnement de véhicules. Il peut être public ou privé situé à l'extérieur en étage ou encore en sous-sol. On les retrouve souvent près des bâtiments publics comme les gares ou les aéroports, autour des lieux de travail, des centres commerciaux…[3].

Le symbole signalétique presque universel signifiant « parking » est un « P » majuscule blanc sur un disque ou un carré bleu.

Figure I.01 : Un parc de stationnement

IV. Les différents types de parking :

Il existe différents types de parkings, adaptés à des besoins spécifiques selon leur emplacement, leur usage, leur structure ou leur mode de gestion [4]. Voici une liste des principaux types de stationnements :

IV.1. Parkings en surface :

Ce sont des parkings en plein air, sans couverture, situés au niveau de sol. Peuvent être public (par exemple : parking d'un supermarché) ou privé (par exemple : parking d'entreprise) sont facile à construire. Ces parkings sont généralement utilisés pour un stationnement temporaire ou de courte durée. La figure I.02 montre à quoi ressemble un parking en surface.

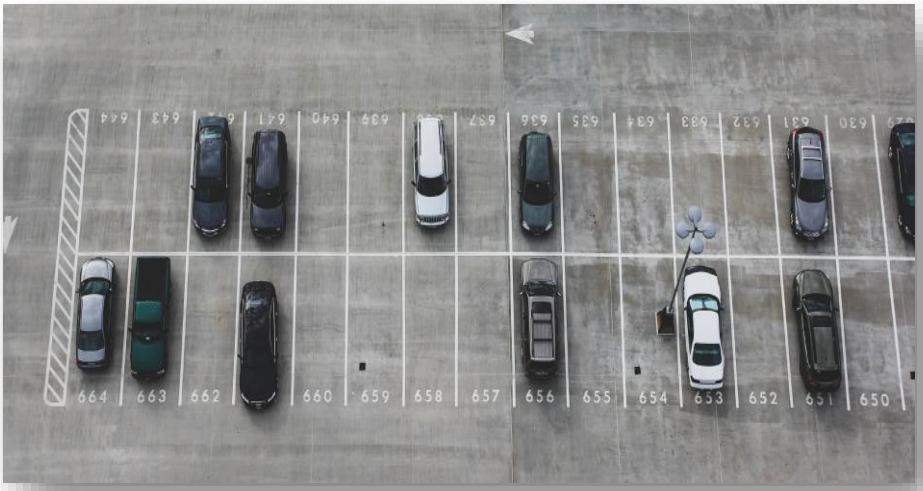


Figure I.02 : Parking en surface

IV.2. Parkings souterrains :

Parkings situés sous le niveau du sol, souvent sous des bâtiments de certaines zones d'activités ou des places publiques. Ce qui permet d'utiliser l'espace de manière plus efficace dans des villes ou la place est limitée. Chaque niveau de parking sous terrain ressemble à un parking classique mais avec quelques différences comme :

- L'air est plus confiné et souvent plus pollué
- Le sol n'est pas nettoyé naturellement par la pluie
- Un revêtement spécifique peut être utilises

Des rampes facilitent le passage d'un niveau à l'autre, tandis que des ascenseurs ou des escaliers permettent aux conducteurs, une fois garés, de retourner à la surface. Dans plusieurs pays, dont la France, les parkings souterrains sont devenus une exigence pour la construction d'immeubles d'habitation dans certaines zones urbaines définies. Des normes strictes régissent des aspects tels que la ventilation, la prévention des incendies et les issues de secours pour assurer la sécurité des occupants. La figure I.03 illustre un parking souterrain.

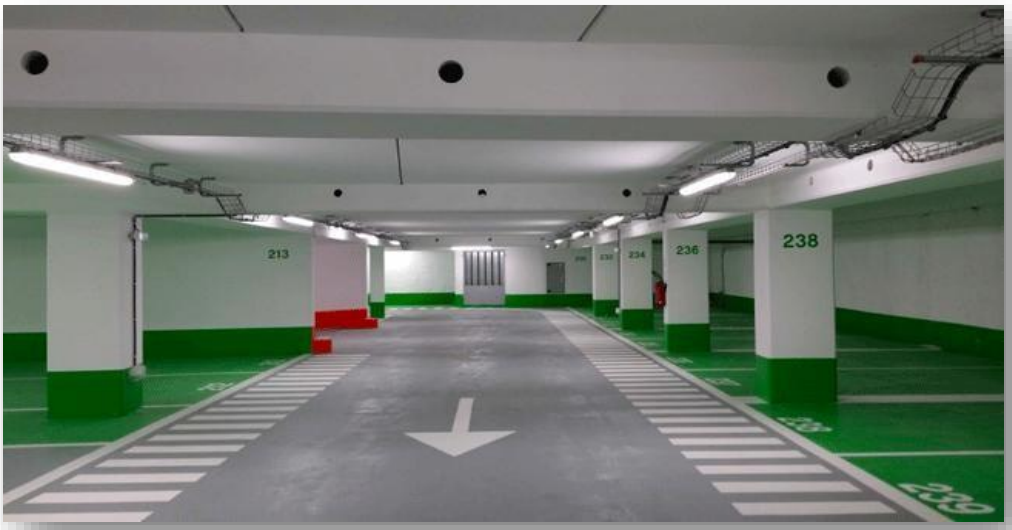


Figure I.03 : Parking souterrain

IV.3. Parkings en étages :

Bâtiments (structure) en hauteur à plusieurs étages conçues pour maximiser le nombre de places (accueillir plus de personnes) sur une petite surface, peuvent être ouverts ou fermés. Il fonctionne comme un parking souterrain à la différence qu'il ne demande pas de lourds travaux de creusement. Il comporte aussi des rampes pour permettre aux véhicules de monter ou de descendre entre les étages ainsi que des ascenseurs et des escaliers pour les piétons.

Un exemple de parking en étages est montré sur la Figure I.04 :

Figure I.04 : Parking en étages

IV.4. Parkings relais :

Ce sont des parcs qui se trouvent en périphérie des villes, sont financés par les collectivités car leur emplacement stratégique améliore la mobilité globale. Les parcs relais sont en effet positionnés près des transports en commun, la logique de cette implantation est d'encourager les conducteurs à laisser leur voiture est à utiliser les Trams, bus, métro…, afin de réduire la circulation dans les centres urbains et l'impact écologique et économique de l'automobile (réduction de la pollution de l'air, réduction des dépenses en essence). La figure I.05 montre un exemple de parking relais.

Figure I.05 : parking relais

V. Les parkings automatisés ou parkings intelligents :

Gérer et organiser le stationnement des véhicules motorisés a toujours été un point sensible, que de nombreuses entreprises ont essayé de régler en proposant des alternatives innovantes. C'est notamment le cas des parcs de stationnement automatisés, qui permettent de gérer automatiquement le stationnement des véhicules à l'intérieur d'espaces entièrement dédiés à cette mission.[5]

Le développement des systèmes de stationnement intelligents résulte d'une évolution remarquable dans la gestion des espaces de stationnement. Leur évolution témoigne d'une volonté constante de répondre aux exigences des villes modernes.

V.1. Historique :

Bien que Les systèmes de stationnement automatisés puisse sembler être une invention récente, l'histoire du stationnement automatisé remonte à plus de 100 ans. Nés en France au début du siècle dernier, leur histoire peut être marquée par trois moments importants :

– **En 1905**, à l'occasion de l'Exposition universelle, le Garage **Rue de Ponthieu**, premier parking automatisé, est conçu à Paris. La force motrice était le besoin d'augmenter de façon drastique la capacité d'une petite zone, répondant ainsi aux demandes du marché de l'époque.

– **L'année 1920** voit l'émergence du design de la grande roue, également connu sous le nom de « **Paternoster** ». La structure permettait de garer jusqu'à 8 véhicules dans un seul compartiment, optimisant ainsi grandement l'espace.

– **En 1950** à Washington DC, l'avancée technologique donne naissance au premier parking automatisé sans chauffeur, un modèle qui s'imposera rapidement dans le monde entier. Aujourd'hui, les systèmes de stationnement automatisés répondent principalement aux besoins des zones urbaines et suburbaines, où la demande de zones de stationnement est considérablement plus élevée que l'espace disponible [6].

Les figures I.06 et I.07 montrent respectivement les parking Ponthieu et Paternoster.

Figure I.06 : (Garage Rue de Ponthieu – Paris 1905)

Figure I.07 : (Paternoster – Chicago 1923)

V.2. Technologies clés pour la gestion intelligente du stationnement :

Aujourd'hui, la transition vers des parkings intelligents est déjà bien engagée, comme en témoignent les robots-voituriers capables de stationner les véhicules de manière autonome. La prochaine étape consistera à permettre aux voitures autonomes de se garer elles-mêmes, sans aucune intervention humaine. Cette évolution est désormais reconnue comme essentielle par les villes et les gestionnaires de parkings à travers le monde. Elle repose sur de nombreuses avancées technologiques, dont certaines occupent une place centrale dans le développement de la prochaine génération de parkings intelligents.

Parmi les innovations technologiques majeures, on retrouve les capteurs de détection de véhicules, qui permettent de surveiller en temps réel l'occupation des places de stationnement. À cela s'ajoutent les systèmes de communication sans fil qui assurent une transmission rapide et fiable des données. L'analyse des mégadonnées joue également un rôle crucial, en fournissant des informations précieuses pour optimiser la gestion du stationnement.

Par ailleurs, l'amélioration de l'accès à Internet permet désormais aux conducteurs de rechercher des places de stationnement en ligne. Dans le même temps, le développement des applications mobiles facilite grandement la localisation, la réservation et le paiement des places disponibles. L'ensemble de ces avancées technologiques contribue à transformer en profondeur la gestion et l'utilisation des parkings, en renforçant leur efficacité et en améliorant l'expérience des usagers [7].

V.3. Les caractéristiques des parkings intelligents :

Les caractéristiques d'un parking intelligent reposent sur l'utilisation de technologies avancés afin d'optimiser la gestion des emplacements de stationnement, renforcer l'expérience utilisateur et relever les défis de l'urbain. Voici quelques-unes de ces caractéristiques :

- Réservation via application mobile : Les utilisateurs se réservent à l'avance une place de stationnement à travers une application mobile simple, en sélectionnant un emplacement disponible et en terminant le paiement (uniques ou d'abonnement).
- Accès sécurisé automatique : L'accès à le stationnement est géré par un système de validation par code QR ou code numérique unique, produit lors de la réservation, qui permet de faire ouvrir la barrière uniquement pour les codes corrects.
- Capteurs de détection en temps réel : Des capteurs infrarouges ou équivalents surveillent l'état des places de stationnement (libres ou occupées), en informant l'application et les systèmes de gestion pour une mise à jour instantanée.
- Affichage dynamique des places : Un afficheur à l'entrée du parking affiche en temps réel la disponibilité des places de stationnement, permettant aux clients n'ayant pas accès à l'application de voir l'état du stationnement.
- Connectivité haute performance : Les réseaux sans fil assure une transmission rapide des informations entre capteurs, application mobile et systèmes de contrôle pour permettre une gestion fluide.

- Application mobile multifonctionnelle : L'application permet de rechercher des emplacements, de réserver, de payer et de générer des codes d'accès (QR ou numérique), offrant une expérience utilisateur simplifiée et centralisée.
- Support à véhicules autonomes : Le système est conçu pour être compatible avec les futures technologies de stationnement autonome, permettant aux véhicules de se garer automatiquement sans intervention humaine via une infrastructure connectée.

V. L'internet des objets (IoT) :

L'Internet des Objets (IoT) désigne un réseau d'objets physiques intégrant des capteurs, des logiciels et d'autres technologies leur permettant de se connecter à Internet pour échanger des données. Ces objets peuvent interagir entre eux ou avec des systèmes centralisés afin d'automatiser ou de surveiller des tâches spécifiques.

Les objets connectés fonctionnent en collectant des données via des capteurs, qu'ils envoient ensuite à un serveur ou à une plateforme cloud. Ces données peuvent être analysées pour déclencher des actions ou être visualisées à distance via une interface utilisateur [8].

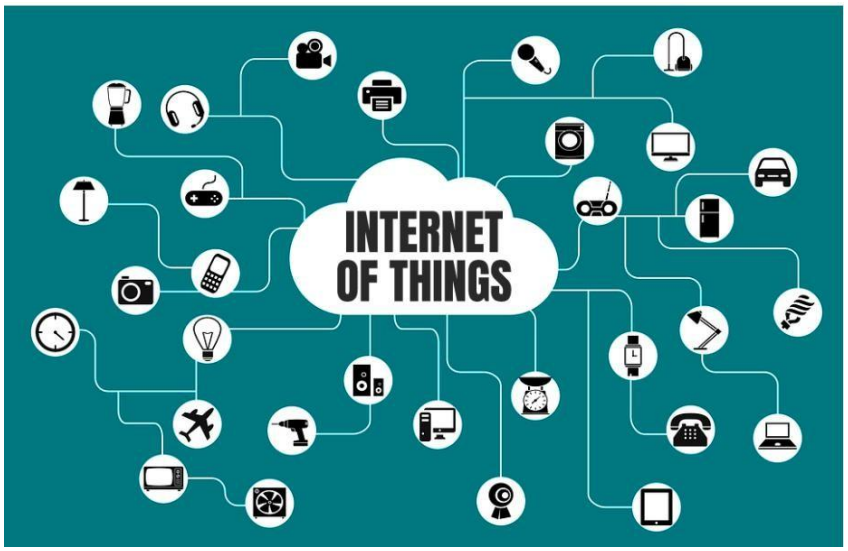


Figure I.08 : Internet des objets

IV.1. Domaines d'application :

L'IoT est présent dans de nombreux domaines : la maison connectée, la santé, l'agriculture, l'industrie et les villes intelligentes. Dans le contexte du stationnement, par exemple l'IoT permet de détecter en temps réel la disponibilité des places et de guider les véhicules.

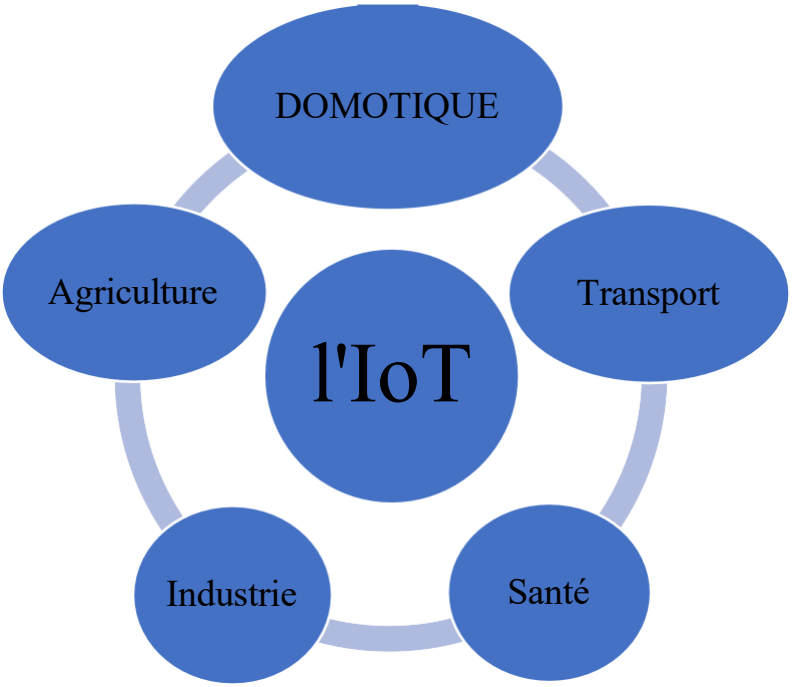


Figure I.09 : Domaine d'IOT

IV.2. Applications de l'IoT dans le cadre du parking intelligent :

Dans le contexte d'un parking intelligent, l'IoT permet :

- **Surveillance en temps réel :** Les capteurs détectent l'occupation des places et envoient les données à une base de données
- **Accessibilité :** Les utilisateurs accèdent aux informations via une application mobile, facilitant la recherche de places disponibles.
- **Automatisation :** Les systèmes IoT peuvent gérer automatiquement l'allocation des places ou déclencher des alertes en cas d'anomalies.
- **Flexibilité :** Le système peut être enrichi par d'autres fonctionnalités, comme la réservation de places ou l'intégration avec des systèmes de paiement.

VI. Identification de notre projet :

Le projet vise à concevoir un système de stationnement intelligent à des fins de maximisation du contrôle de la place de stationnement via une solution numérique et automatique . Ce système s'appuie sur une application mobile qui permet aux utilisateurs de réserver une place de stationnement en choisissant une place disponible , d'effectuer un paiement via l' application et de recevoir un code numérique unique pour valider leur réservation . À l' entrée du parking l'utilisateur peut saisir le code l'accès.

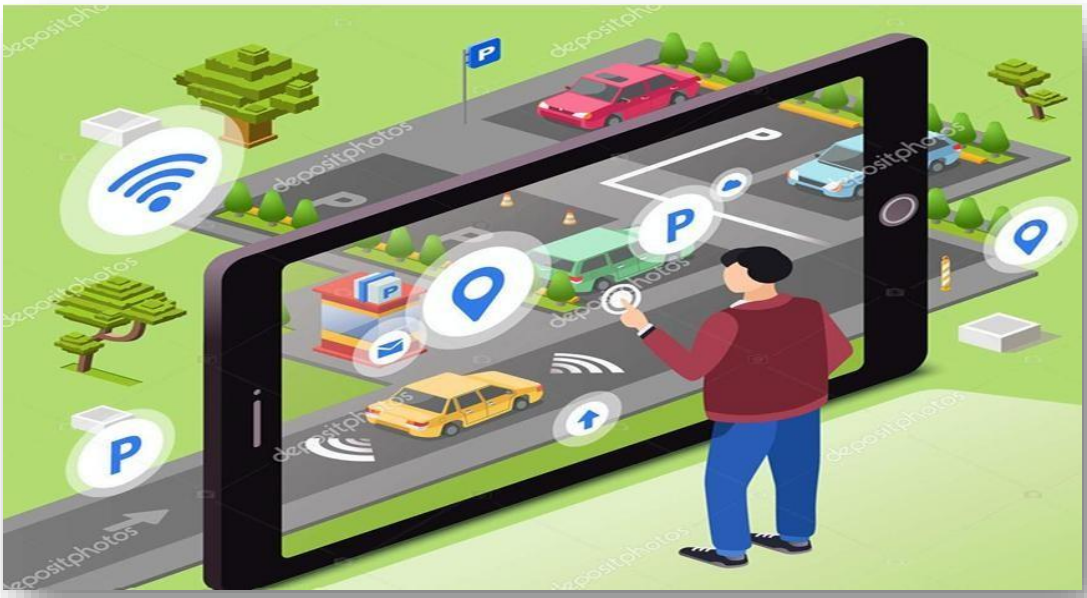


Figure I.10: Le parking intelligent

L'objectif principal du notre projet est de développer un système de stationnement intelligent, accessible et maniable, qui simplifie la réservation et l'accès au stationnement pour les usagers et optimise l'utilisation des places existantes. Le système doit :

- Améliorer l'expérience utilisateur en proposant une réservation rapide et sécurisée via une application mobile, adaptée aux besoins des conducteurs (visiteurs, travailleurs, habitants).
- Automatisez l'accès au parking grâce à des codes uniques, garantissant une entrée fluide et sécurisée.
- Faciliter l'information en temps réel sur la disponibilité des lieux via un afficheur, rendant le système inclusif pour les utilisateurs non équipés de l'application.

- Le système permet une gestion efficace du stationnement en réduisant principalement le temps de recherche de place, ce qui diminue les encombrements.

VII. Les avantages et les inconvénients du projet parking intelligent :

Les avantages :

- Les opérateurs de parkings peuvent surveiller en temps réel les niveaux d'occupation et la gestion des ressources, ce qui facilite la maintenance préventive et l'efficacité opérationnelle [9].
- Guider et donner les informations pour les usagers et les touristes sur le stationnement disponible et non occupé.
- Utiliser facilement des places de stationnement.
- Augmenter l'activité et le déplacement plus librement dans la ville en utilisant les technologies modernes.
- Gagner en temps lors de la recherche de l'espace libre pour stationnement.
- Diminuer de la pollution et l'utilisation de l'essence et l'émission de gaz toxiques.

Les inconvénients :

- Les systèmes de parking intelligents dépendent d'une connexion réseau fiable, et les pannes de réseau, bugs logiciels ou capteurs défectueux peuvent rendre le système inutilisable, perturbant ainsi la gestion du parking
- Certains systèmes de stationnement intelligent reposent sur des applications mobiles pour la réservation et le paiement, ce qui peut pénaliser les conducteurs ne disposant pas de smartphone ou d'accès à Internet.

VIII. Conclusion :

Dans ce chapitre, il était question pour nous de faire l'état des connaissances dans le domaine des parkings intelligents. Ce dernier joue un rôle très important dans les villes, car il assure une bonne mobilité, cependant il règle de nombreux problèmes, des problèmes de pollution, d'encombrement, et aussi il améliore les niveaux des services en ville.

Ce chapitre prépare le terrain pour les prochaines étapes de l'étude, qui porteront sur les aspects techniques et pratiques afin de concevoir un système adapté aux besoins des utilisateurs.

Chapitre II : Conception du système Parking intelligent

I. Introduction :

La phase de conception constitue une étape cruciale dans le développement d'un système électronique, car elle permet de traduire les besoins identifiés en solutions techniques cohérentes et fiables. Dans le cadre de notre projet de réalisation d'un parking intelligent, la conception vise à structurer un système de gestion de parking automatisé, basé sur l'utilisation d'une carte à microcontrôleur, de capteurs pour connaitre l'environnement de notre système et d'actionneurs [10].

Les objectifs principaux de notre système sont l'automatisation des processus d'entrée et de sortie des véhicules, la gestion efficace des places de stationnement et la fourniture d'informations claires aux utilisateurs. Pour garantir la robustesse et la clarté de la conception, des schémas blocs et des schémas matériels seront utilisés, offrant une représentation visuelle et structurée des interactions entre les composants du système.

II. Partie Matérielle :

II.1. Schéma bloc :

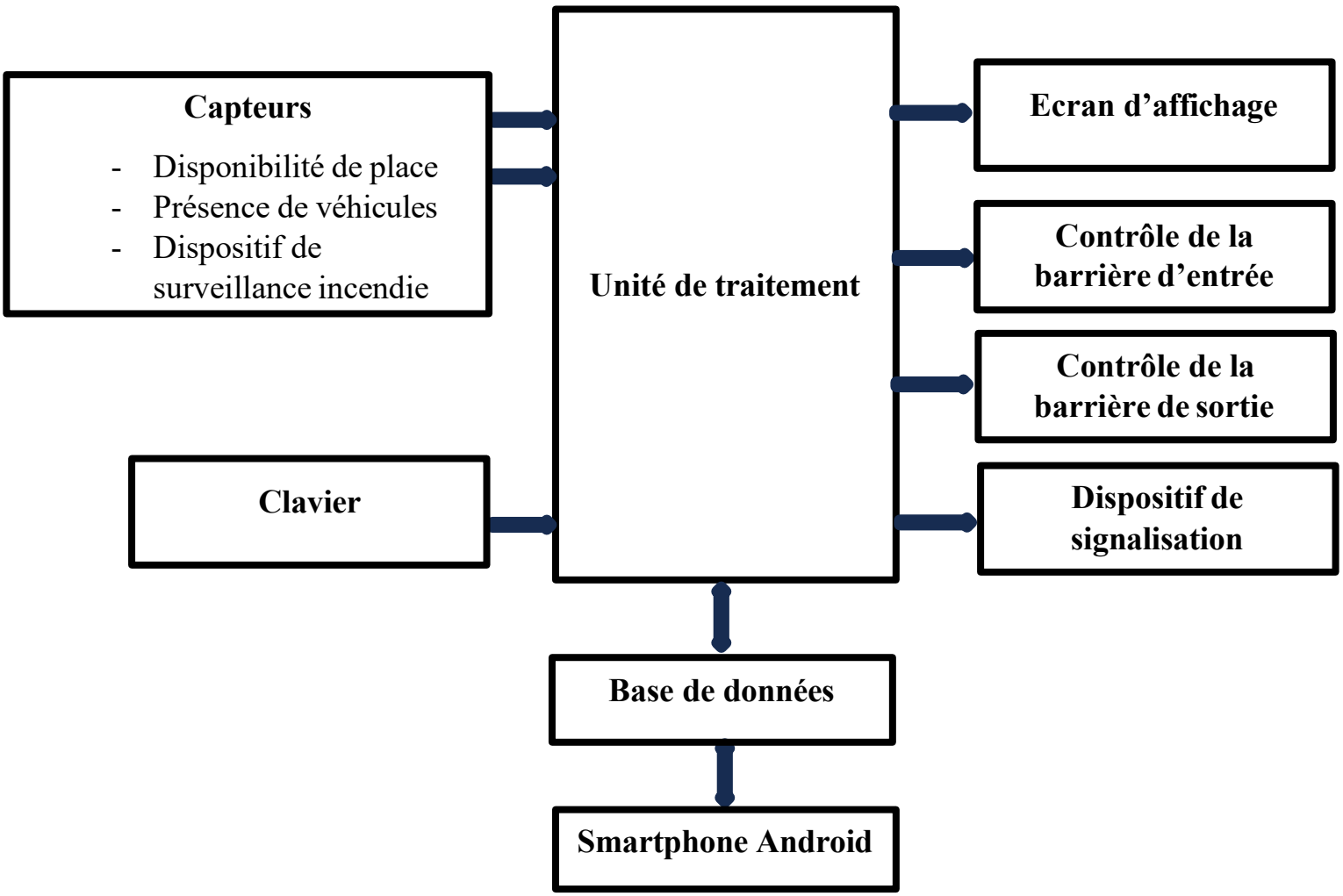


Figure II.01 : Schéma bloc de notre projet

II.2. L'unité de traitement :

Pour les besoins de notre projet, nous avons opté pour la carte ESP32. Celle-ci est une carte de développement électronique open-source, équipée du microcontrôleur ESP32 de chez Espressif Systems. Très appréciée pour les applications en domotique et IoT (Internet des Objets), elle se distingue par sa connectivité sans fil (Wi-Fi et Bluetooth) et ses hautes performances de calcul.

Les caractéristiques courantes de la carte **ESP32** sont :

- Une gamme de chips avec des performances différentes :
  ESP32, C1, C2, C3, S1, S2, S3

- Microcontrôleur **ESP32** à double cœur avec un processeur principal à 160 MHz et un coprocesseur à 240 MHz
- Support de la connectivité Wi-Fi 802.11b/g/n et Bluetooth v4.2
- 32 Mo de mémoire flash et 520 Ko de mémoire RAM
- 34 broches d'entrée/sortie, dont 18 PWM et 8 entrées analogiques
- Support de l'OTP (One-Time Programmable) pour la mémoire flash et de la mémoire FRAM (Ferroelectric Random Access Memory)
- Alimentation via USB ou 3,3 V externe

Et voici quelques possibilités en mémoire [11] :

- Mémoire interne

  – 448 KB Internal ROM

  – 520 KB Internal SRAM

  – 8 KB RTC FAST Memory

  – 8 KB RTC SLOW Memory

- Mémoire externe :
  La mémoire SPI hors puce peut être mappée dans l'espace d'adressage disponible en tant que mémoire externe. Des parties de la mémoire intégrée peuvent être utilisées comme cache transparent pour cette mémoire externe.

– Supports up to 16 MB off-Chip SPI Flash.

– Supports up to 8 MB off-Chip SPI SRAM.

La carte **ESP32** est compatible avec une grande variété de langages de programmation, tels que C, C++, Python et MicroPython. Elle est également compatible avec de nombreux frameworks de développement tels que Arduino et Espressif IoT Development Framework (ESP-IDF).

Figure II.02 : Carte de développement ESP-32 38PIN

II.2.1. Les types de module Espressif :

Espressif Systems propose une gamme variée de modules basés sur ses systèmes sur puce (SoC) comme l'ESP8266, l'ESP32 et leurs variantes, conçus principalement pour les applications IoT avec connectivité Wi-Fi, Bluetooth, et parfois d'autres protocoles comme Zigbee ou Thread.

Étant donné que nous utilisons l'ESP32 dans notre projet, voici les principaux types qu'elle prend en charge :

- ✓ ESP32-WROOM
- ✓ ESP32-WROVER
- ✓ ESP32-MINI
- ✓ ESP32-PICO
- ✓ ESP32-S2
- ✓ ESP32-S3
- ✓ ESP32-C3
- ✓ ESP32-C6
- ✓ ESP32-H2
- ✓ ESP32-P4

II.2.2. Comparaison rapide des modules ESP32 :

Module	**SoC**	**Cœurs**	**Wi-Fi**	**Bluetooth**	**Autres Protocoles**	**Utilisation principale**
ESP32-WROOM	ESP32-D0WD(Q6)	Dual-core Xtensa LX6	802.11 b/g/n	BT 4.2, BLE	-	IoT général
ESP32-WROVER	ESP32-D0WD	Dual-core Xtensa LX6	802.11 b/g/n	BT 4.2, BLE	PSRAM	AIoT, streaming
ESP32-S2	ESP32-S2	Monocœur Xtensa LX7	802.11 b/g/n	BLE	USB OTG	Interfaces utilisateur

ESP32-S3	ESP32-S3	Dual-core Xtensa LX7	802.11 b/g/n	BLE 5.0	AI vectorielle	AIoT, vision
ESP32-C3	ESP32-C3	Monocœur RISC-V	802.11 b/g/n	BLE 5.0	-	IoT compact
ESP32-C6	ESP32-C6	Monocœur RISC-V	Wi-Fi 6	BLE 5.0	Zigbee, Thread	Maisons intelligentes
ESP32-H2	ESP32-H2	Monocœur RISC-V	-	BLE 5.0	Zigbee, Thread	Basse consommation
ESP32-P4	ESP32-P4	Dual-core RISC-V	-		NPU, H264	IA, vision

Tableau II.1: Comparatif des principaux modules ESP32

II.3. Esp32 WROOM-32 :

Dans notre projet on s'est intéressé à l'ESP32 WROOM-32 :

Figure II.03: Carte esp32 WROOM-32

L'ESP32-WROOM-32 possède 38 broches. Les définitions des différentes broches sont données par le tableau 2 ci-dessous [12] :

Nom	N°	Type	Fonction
GND	1	P	Masse (Ground)
3V3	2	P	Alimentation 3.3V
EN	3	I	Signal d'activation du module (actif à l'état haut)
SENSOR_VP	4	I	GPIO36, ADC1_CH0, RTC_GPIO0
SENSOR_VN	5	I	GPIO39, ADC1_CH3, RTC_GPIO3
GPIO34	6	I	GPIO34, ADC1_CH6, RTC_GPIO6
GPIO35	7	I	GPIO35, ADC1_CH7, RTC_GPIO5
GPIO32	8	I/O	GPIO32, XTAL_32K_P, ADC1_CH4, RTC_GPIO4
GPIO33	9	I/O	GPIO33, XTAL_32K_N, ADC1_CH5, RTC_GPIO9
GPIO25	10	I/O	GPIO25, DAC_1, ADC2_CH8, RTC_GPIO6, EMAC_RXD1
GPIO26	11	I/O	GPIO26, DAC_2, ADC2_CH9, RTC_GPIO7, EMAC_RXD0
GPIO27	12	I/O	GPIO27,ADC2_CH7,RTC_GPIO17,TOUCH7, MTCK, EMAC_RX_DV
GPIO14	13	I/O	GPIO14, ADC2_CH6, TOUCH6, RTC_GPIO16, MTMS, HSPICLK, HS2_CLK, SD_CLK
GPIO12	14	I/O	GPIO12, ADC2_CH5, TOUCH5, RTC_GPIO15, MTD1, HSPID, HS2_DATA2, SD_DATA2
GND	15	P	Masse
GPIO13	16	I/O	GPIO13, ADC2_CH4, TOUCH4, RTC_GPIO14, MTCK, HSPICLK, HS2_DATA3, SD_DATA3
GPIO9*	17	I/O	GPIO9, SD_DATA2, EMAC_RX_ER
GPIO10*	18	I/O	GPIO10, SD_DATA3, SPIHD, HS1_DATA2, U1RXD
GPIO11*	19	I/O	GPIO11, SD_CMD, SPICS0, HS1_DATA3, U1TXD
GPIO6*	20	I/O	GPIO6, SD_CLK, SPICLK, HS1_CLK, U1RTS
GPIO7*	21	I/O	GPIO7, SD_DATA0, SPIQ, HS1_DATA0, U2TXD
GPIO8*	22	I/O	GPIO8, SD_DATA1, SPID, HS1_DATA1, U2CTS
GPIO15	23	I/O	GPIO15, ADC2_CH3, TOUCH3, RTC_GPIO13, HS2_CMD
GPIO2	24	I/O	GPIO2,ADC2_CH2,TOUCH2,RTC_GPIO12, HS2_DATA0, SD_DATA0

GPIO0	25	I/O	GPIO0,ADC2_CH1,TOUCH1,RTC_GPIO11,CLK_OUT1,EMAC_TX_CLK
GPIO4	26	I/O	GPIO4,ADC2_CH0,TOUCH0, RTC_GPIO10, CLK_OUT2, EMAC_TX_ER
GPIO16	27	I/O	GPIO16, HS1_DATA4, U2RXD, EMAC_CLK_OUT
GPIO17	28	I/O	GPIO17, HS1_DATA5, U2TXD, EMAC_CLK_OUT_180
GPIO5	29	I/O	GPIO5, VSPIQ, HS1_DATA6, EMAC_RX_CLK
GPIO18	30	I/O	GPIO18, VSPICLK, HS1_DATA7
GPIO19	31	I/O	GPIO19, VSPIHD, EMAC_TXD0
GND	32	P	Masse
GPIO21	33	I/O	GPIO21, VSPIHD, EMAC_TX_EN
RXD0	34	I/O	GPIO3, U0RXD, CLK_OUT2
TXD0	35	I/O	GPIO1, U0TXD, CLK_OUT3, EMAC_RXD2
GPIO22	36	I/O	GPIO22, VSPIWP, U0RTS, EMAC_TXD1
GPIO23	37	I/O	GPIO23, VSPIHD, HS1_STROBE
GND	38	P	Masse

Tableau II.2: Pin Définitions

Les broches GPIO6, GPIO7, GPIO8, GPIO9, GPIO10, GPIO11, sont connectées à la mémoire flash SPI intégrée sur le module et ne sont pas recommandées pour d'autres utilisations.

La disposition des différentes broches de l'ESP 32 WROOM citées dans le tableau précédent est donnée par la figure ci-dessous représentant la vue de dessus.

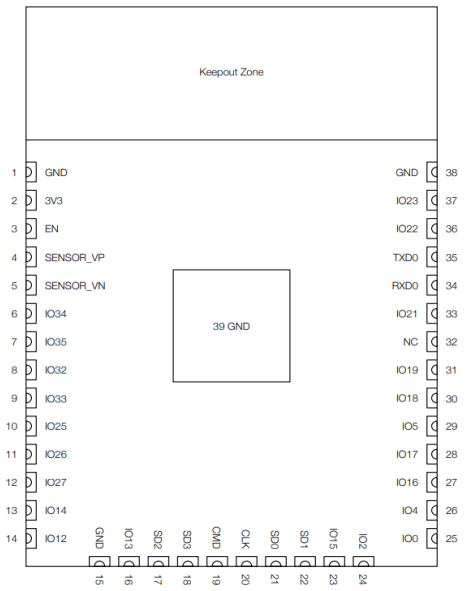


Figure II.04 : Disposition des broches de l'ESP32-WROOM-32 (vue de dessus)

Le choix de l'utilisation de la carte ESP32 dans le système que nous comptons réaliser est motivé par le fait que :

- Elle intègre le Wi-Fi, ce qui facilite la connexion avec mon application mobile.
- Elle dispose d'un nombre suffisant de GPIO pour connecter les capteurs que nous utilisons et plus généralement tous les composants que nous souhaitons utilisés.
- Elle représente une solution économique, performante et facile à programmer.
- Elle convient parfaitement à notre application mobile, car elle ne nécessite ni une précision temps réel extrême, ni une consommation énergétique minimale.
- Elle peut être programmée avec Arduino, un environnement simple à utiliser et que nous maîtrisons bien. De plus, nous l'avons déjà employée dans plusieurs projets, et Arduino inclut généralement toutes les bibliothèques requises.
- Par rapport à son coût, qui est relativement faible, elle offre un excellent rapport qualité-prix.

II.4. Capteurs :

Un capteur est un dispositif de prélèvement d'informations qui élabore, à partir d'une grandeur physique, une autre grandeur physique de nature différente (généralement **électrique**) afin de fournir une indication ou une mesure exploitable. C'est à partir du moment où l'on a su capter une grandeur physique et exploiter ses caractéristiques, qu'on a pu créer des systèmes automatiques intelligents, capables de se contrôler tout seuls, sans avoir besoin de l'intervention humaine [13].

- Entrée : grandeur physique

  C'est la valeur mesurée par le capteur (comme la position, la pression, la température, le niveau, le déplacement, etc.). Cette grandeur donne une information utile au système qui va l'analyser ou l'utiliser.
- Sortie : grandeur électrique

  C'est le signal produit par le capteur. Il est sous forme électrique pour être facilement utilisé par les appareils de mesure ou les systèmes automatiques. Ce signal peut être de différents types :

  -Analogique : il varie en continu avec le temps (ex. : une tension qui change selon la température).

  -Logique : il ne prend que deux états possibles (0 ou 1), comme un interrupteur (on appelle ça aussi "tout ou rien").

  -Numérique : il donne une valeur précise codée, souvent avec plusieurs chiffres (ex. : un capteur qui envoie la valeur 127 au lieu de juste 0 ou 1).

Les détecteurs font partie de la famille des capteurs, leur fonction est de transformer la grandeur physique d'entrée en une grandeur logique. L'information en sortie d'un détecteur est de type tout ou rien, 0 ou 1 logique.

Exemple de détection : Présence, absence, passage, position, proximité

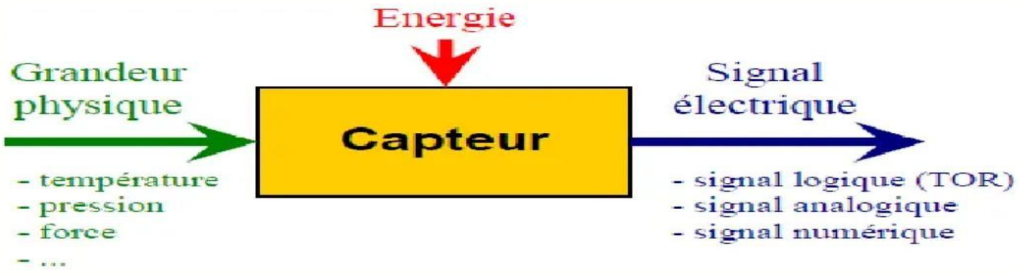


Figure II.05: Principe de fonctionnement d'un capteur

II.4.1. Capteur de disponibilité de place et de présence des véhicules:

La détection de la disponibilité des places et de la présence des véhicules est assurée par un capteur. Le capteur que nous allons utiliser est un capteur infrarouge.

Un capteur infrarouge est un composant électronique utilisé pour capter le rayonnement infrarouge (IR) émis ou réfléchi par des objets. Le rayonnement infrarouge est une partie du spectre électromagnétique, des longueurs d'onde qui se situent juste au-delà de la lumière visible, typiquement comprises entre 700 nm et 1 mm . Ce type de détecteur peut être utilisé pour repérer la présence, la distance, la température ou le mouvement d' un objet, en fonction du principe selon lequel tout corps émet naturellement un rayonnement infrarouge en fonction de sa température. Nous différencions essentiellement deux sortes de capteurs infrarouges : les capteurs actifs, qui émettent un faisceau IR et observent sa réflexion, et les capteurs passifs, qui enregistrent directement le rayonnement infrarouge produit par les corps . Par sécurité et faible coût, les capteurs infrarouges sont largement appliqués à des domaines variés comme la sécurité, l' automatisation , la robotique ou le domaine médical.

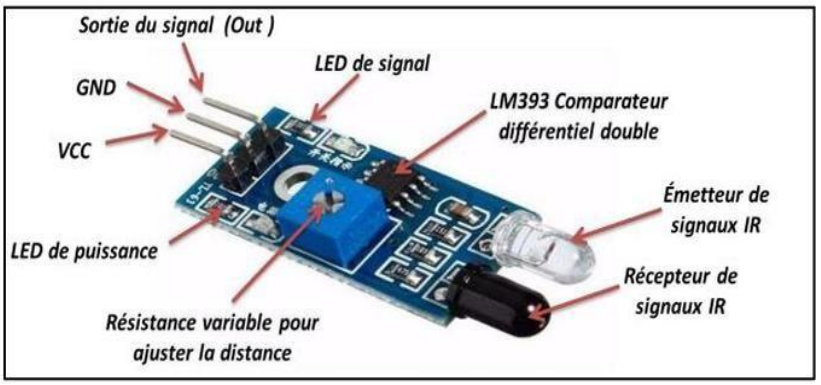


Figure II.06: Structure d'un Capteur IR

II.4.1.1. Principe de fonctionnement d'un capteur infrarouge :

Le principe de fonctionnement d' un capteur infrarouge repose sur la détection du rayonnement infrarouge, un type d'énergie électromagnétique naturellement émise par tous les corps en fonction de leur température. Ce rayonnement infrarouge invisible pour l' œil humain peut être mesuré afin d'en extraire des informations telles que la présence, le mouvement ou la température d' un objet. Il existe deux principaux types de capteurs infrarouges , chacun avec un principe de fonctionnement différent :

- **Capteur infrarouge actif** : fonctionne sur la base de l'émission et de la détection d'un rayonnement infrarouge . Il se compose principalement de deux parties :

- Un émetteur IR : souvent une LED IR (diode infrarouge électroluminescente).
- Un récepteur : comme une photodiode ou un phototransistor, sensible aux infrarouges.

Le travail se déroule en plusieurs étapes :

- **Emission du signal :** L' émetteur IR émet un faisceau d' infrarouge vers une zone cible.
- **Réflexion du signal :** Lorsque ce faisceau atteint un objet, une partie du rayonnement est réfléchie.
- **Réception du signal réfléchi :** Le récepteur capte le rayonnement infrarouge réfléchi par l'objet.
- **Traitement du signal :** L'intensité du signal reçu est analysée pour déterminer la présence, la proximité ou la distance de l' objet détecté.

Ce principe permet de détecter un objet sans contact physique. Plus l'objet est proche, plus le signal réfléchi est fort. Ce type de capteur est largement utilisé pour la détection d'obstacles, les robots suiveurs de ligne, les compteurs de passage, ou les systèmes de contrôle automatique.

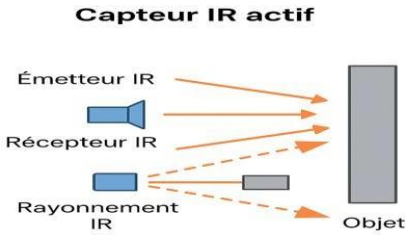


Figure II.07 : Principe de fonctionnement d'un capteur IR actif

- **Capteur infrarouge passif (PIR) :** Contrairement au capteur actif, le capteur passif n'émet aucun rayonnement. Il se contente de recevoir le rayonnement infrarouge naturellement émis par les objets chauds, notamment les êtres vivants. Le capteur PIR détecte les variations du rayonnement IR dans son champ de vision, ce qui lui permet de repérer un mouvement ou une présence humaine. Ce principe est utilisé dans les systèmes de sécurité, comme les détecteurs de mouvement.

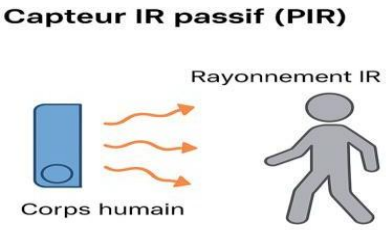


Figure II.08: Principe de fonctionnement d'un capteur IR passif

II.4.1.2. Les avantages et limite d'un capteur IR :

Les capteurs IR présentent les avantages suivants :

- Faible coût
- Faible consommation d'énergie
- Non sensibles à la lumière visible
- Détection sans contact
- Temps de réponse rapide
- Faciles à intégrer

Toutefois, ce type de capteurs ont des limites dues à :

- Sensibilité à la température ambiante
- Portée limitée
- Sensibilité aux matériaux
- Dépendance à l'alignement (capteurs actifs)
- Incapacité à identifier les objets

II.4.2. Dispositif de surveillance incendie:

Pour renforcer la sécurité de notre parking, un dispositif de surveillance incendie a été ajouté au système. Le dispositif utilisé est un détecteur de flamme.

Un détecteur de flamme est un type de capteur capable de détecter la présence d'une flamme et d'y répondre. Ces détecteurs ont la capacité d'identifier les liquides sans fumée et la fumée qui peut créer un feu ouvert. Par exemple, les détecteurs de flamme sont largement utilisés dans les chaudières, car ils peuvent détecter la chaleur, la fumée et le feu. Ces dispositifs peuvent également détecter un incendie en fonction de la température et du mouvement de l'air. Les détecteurs de flammes utilisent la technologie des ultraviolets (UV) ou des infrarouges pour identifier les flammes, ce qui signifie qu'ils peuvent donner l'alerte en moins d'une seconde. Le détecteur de flammes réagit à la détection d'une flamme en fonction de son installation. Il peut par exemple déclencher une alarme, désactiver la conduite de carburant ou même activer un système d'extinction d'incendie [14].

Figure II.09: Détecteur de flamme

II.4.2.1. Principe de fonctionnement d'un détecteur de flamme :

Les détecteurs de flamme utilisent généralement des capteurs infrarouges pour détecter le rayonnement spécifique émis par les flammes. En présence d'une flamme, le capteur détecte le rayonnement infrarouge et génère un signal électrique. Ce signal est ensuite traité par le système de détection incendie pour déclencher une alarme ou activer des mesures d'extinction [15].

II.4.2.2. Les avantages et limites des détecteurs de flammes :

Les détecteurs de flamme présentent les avantages suivants :

- Détection rapide
- Fiabilité
- Capacité à fonctionner dans des environnements variés

- Réduction des risques
- Intégration moderne

Toutefois, ce type de détecteurs ont des limites dues à :

- Sensibilité aux interférences
- Coût élevé
- Nécessité d'entretien régulier

II.4.3. Clavier pour introduire des informations

Notre parking intelligent doit interagir avec l'utilisateur (le client). A cet effet, nous allons utiliser un clavier permettant d'obtenir cette interaction.

Le clavier à utiliser est un clavier matriciel 4x4. Il est constitué par un groupe de boutons alignés en 4 lignes et en 4 colonnes, qui font toute une matrice. Chaque bouton est à l' intersection d'une ligne et d'une colonne. Lorsqu'un utilisateur appuie sur un bouton, il établit un contact électrique entre une ligne et une colonne spécifique, ce qui permet au système de détecter la touche en question .

Figure II.10 : Clavier matricielle 4x4

II.4.3.1. Principe de fonctionnement d'un clavier matricielle 4x4 :

Son mode de fonctionnement est basé sur un processus appelé balayage matriciel (scanning) :

- Le microcontrôleur configure soit les lignes soit les colonnes comme des sorties et les autres entrées avec pull-up.
- Il active une ligne ou une colonne (à la fois) en passant par la mise en l'état bas (LOW), les autres étant en l'état haut (HIGH).

- Il lit ensuite l'état des colonnes ou des lignes en entrée : Si une touche est pressée, un état bas est détecté sur une colonne, indiquant une connexion entre cette colonne et la ligne activée.
- En répétant ce processus pour chaque ligne (ou colonne), le microcontrôleur peut identifier précisément quelle touche est pressée

Exemple :

- Le microcontrôleur met R1 = LOW, R2, R3, R4 = HIGH.
- Il lit : C1, C2, C3, C4. Si C2 = LOW, alors la touche (R1, C2) est pressée.
- Ensuite, il passe à R2 = LOW et recommence la lecture.

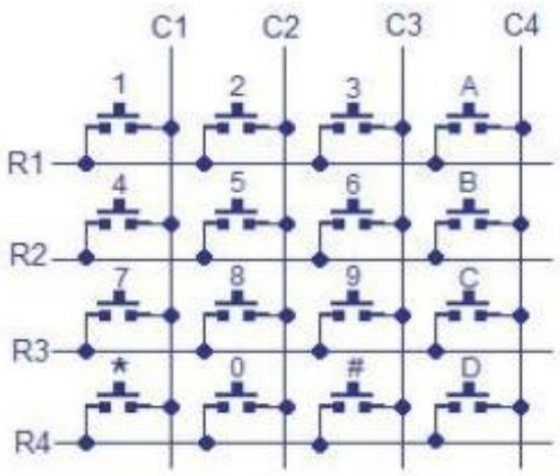


Figure II.11: Schéma d'un clavier matricielle 4x4

II.4.3.2. Les avantages et limites d'un clavier matricielle :

Avantages :

- Réduction du nombre de broches
- Coût faible
- Facilité d'intégration
- Taille compacte

Limites :

- Détection limitée à une touche à la fois
- Pas de retour tactile avancé
- Vitesse de détection dépend du scan

- Pas adapté aux interfaces complexes

II.5. Les actionneurs :

II.5.1. Définition d'un actionneur :

Un actionneur est un système qui transforme l'énergie qui lui est fournie (qu'il reçoit) en un phénomène physique (en chaleur, champs magnétique, lumière, mouvement, position, pression, son) qui fournit un travail. L'actionneur fait partie de la partie opérative [16].

Par exemple :

- Lumière à partir d'un courant électrique (diode électroluminescente, lampe...).
- Sons à partir d'un courant électrique (vibreur, avertisseur sonore...).
- Chaleur à partir d'un courant électrique (résistance chauffante).
- Rayonnement infra-rouge à partir d'un courant électrique (diode émissive Infrarouge).

II.5.2. Contrôle de barrière:

Pour automatiser le contrôle de la barrière d'entrée et de sortie du parking, un servomoteur a été intégré au système.

Un servomoteur est un type de moteur spécialement conçu pour réaliser des mouvements rotatifs ou linéaires avec une grande précision. Il fonctionne grâce à un système de rétroaction (ou retour d'information) qui lui permet d'ajuster sa position en temps réel en fonction d'un signal de commande. Ce signal détermine la position cible, et le servomoteur s'y déplace avec exactitude. Grâce à ce mécanisme, il offre un contrôle précis et fluide des mouvements, ce qui le rend essentiel dans les applications où le positionnement précis est crucial, comme en robotique, en automatisation ou en modélisme [17].

Figure II.12 : Un servomoteur SG90

II.5.2.1. Principe de fonctionnement d'un servomoteur :

- **Signal de contrôle :** Il constitue le déclencheur du fonctionnement du servomoteur. Ce signal électrique, envoyé par le contrôleur, indique la position ou le mouvement désiré. Il prend généralement la forme d'une impulsion à largeur variable une technique appelée modulation de largeur d'impulsion (MLI) permettant de coder la position cible.

- **Moteur et mécanisme d'engrenage :** Après réception du signal de commande, le circuit interne du servomoteur alimente le moteur afin qu'il atteigne la position demandée. Celui-ci est généralement couplé à un réducteur, dont le rôle est d'augmenter le couple et de permettre un contrôle précis du mouvement et de la vitesse de l'arbre de sortie.

- **Système de retour d'information :** Ce système joue un rôle essentiel dans la précision du servomoteur en surveillant en temps réel la position actuelle de l'arbre de sortie. Il utilise généralement des capteurs, comme des potentiomètres ou des encodeurs, pour comparer la position réelle à la position cible et ajuster le mouvement si nécessaire.

- **Correction des erreurs :** Le cœur du fonctionnement du servomoteur réside dans son circuit de commande. Celui-ci compare en continu la position actuelle, fournie par le système de retour, avec la position cible définie par le signal de commande. En cas de décalage appelé signal d'erreur le contrôleur ajuste la puissance envoyée au moteur afin de corriger la position jusqu'à ce que celle-ci corresponde exactement à la consigne.

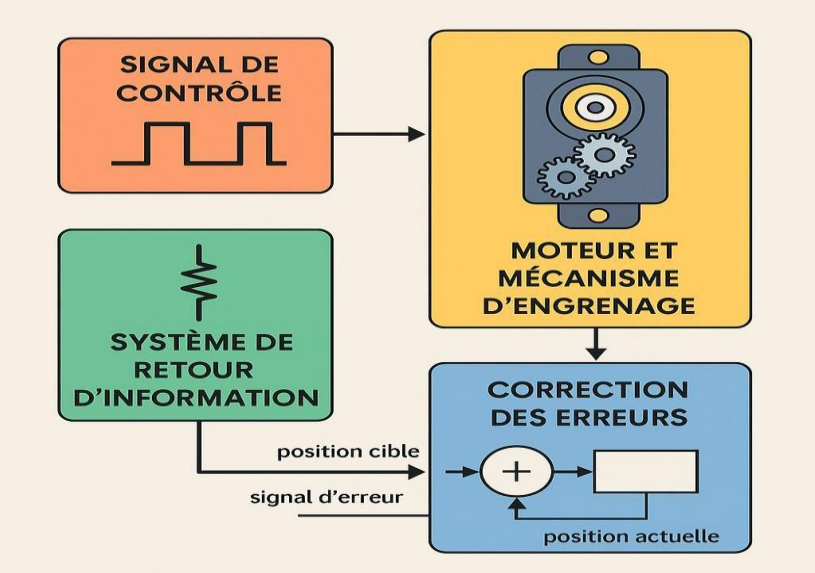


Figure II.13 : Principe de fonctionnement d'un servomoteur

II.5.2.2. Les avantages et limites d'un servomoteur :

Avantages :

- Précision de positionnement
- Contrôle de la vitesse et de l'angle
- Réponse rapide
- Fonctionnement fluide
- Faible consommation en repos

Limites :

- Angle de rotation restreint
- Sensible aux surcharges
- Couple limité
- Bruit en fonctionnement

II.5.3. Un écran d'affichage :

Afin d'informer l'utilisateur en temps réel sur l'état du parking nous allons utiliser un écran d'affichage. Ce dernier est un afficheur LCD.

Un afficheur LCD (Liquid Crystal Display, ou affichage à cristaux liquides en français) est un dispositif électronique qui peut être contrôlé par une interface de communication I2C utilisé pour afficher des informations sous forme visuelle généralement du texte, des chiffres ou des symboles. Il fonctionne grâce aux cristaux liquides, qui ont la particularité de modifier la direction de la lumière lorsqu'ils sont soumis à un champ électrique, ce qui permet de faire apparaître ou disparaître des segments visibles à l'écran.

Dans les systèmes électroniques et embarqués, l'afficheur LCD joue un rôle essentiel en servant de moyen de communication visuelle entre le dispositif et l'utilisateur. Il permet à ce dernier de recevoir des informations claires sur l'état ou le fonctionnement du système, comme les valeurs mesurées par des capteurs, des messages d'alerte, des menus de navigation ou encore des instructions.

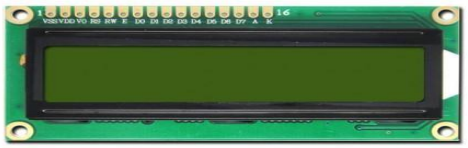


Figure II.14 : Un afficheur LCD 16x2

Module I2C :

I2C permet une communication bidirectionnelle entre un maître (ex. : microcontrôleur) et un ou plusieurs esclaves (ex. : capteurs, écrans) via deux lignes :

- **SDA** (Serial Data) : pour les données.
- **SCL** (Serial Clock) : pour l'horloge synchronisant la communication.

Figure II.15: Module I2C

II.5.4. Dispositif de signalisation :

Afin d'alerter l'utilisateur en temps réel en cas de détection d'une flamme, nous utiliserons un dispositif électronique de signalisation sonore, le buzzer. Ce dernier conçu pour produire un son de bourdonnement lorsqu'il est alimenté. Largement utilisé dans les ordinateurs, imprimantes, photocopieuses, alarmes, jouets électroniques, dispositifs électroniques automobiles, téléphones, minuteurs et autres appareils électroniques ou de signal sonore.

Figure II.16: Un buzzer

Voici le symbole électrique d'un buzzer. Il possède deux broches avec des pôles positif et négatif. Le signe « + » sur la surface représente l'anode, tandis que l'autre broche est la cathode.

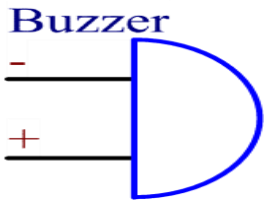


Figure II.17: le symbole électrique d'un buzzer

Il faut vérifier les broches du buzzer :la plus longue est l'anode et la plus courte est la cathode. Assurez-vous de ne pas les inverser lors du branchement, sinon le buzzer ne produira pas de son.

III. Partie Logicielle :

III.1. Arduino IDE :

L'Arduino IDE(Integrated Development Environment) est un environnement de développement intégré open source, développé spécifiquement pour programmer les cartes électroniques de la série Arduino et des cartes compatibles, telle que l'ESP32. Il s'agit d'un logiciel léger et accessible, intuitif, très apprécié dans les milieux de l' électronique embarquée, de la robotique et de l'Internet des objets.

L'IDE Arduino autorise les utilisateurs à coder , à compiler et à télécharger directement du code sur une carte microcontrôleur depuis un ordinateur. Il simplifie grandement le développement en raison d' une interface conviviale et d' un ensemble d' outils inclusifs facilitant le codage .

Le langage utilisé dans l'IDE Arduino est basé sur le langage C/C ++, mais a été simplifié pour le rendre plus accessible , notamment aux débutants. La plupart des fonctions sont prédéfinies et intégrées dans des bibliothèques, ce qui permet de développer des projets complexes sans connaissances approfondies en programmation de bas niveau .

III.1.1. Présentation du logiciel:

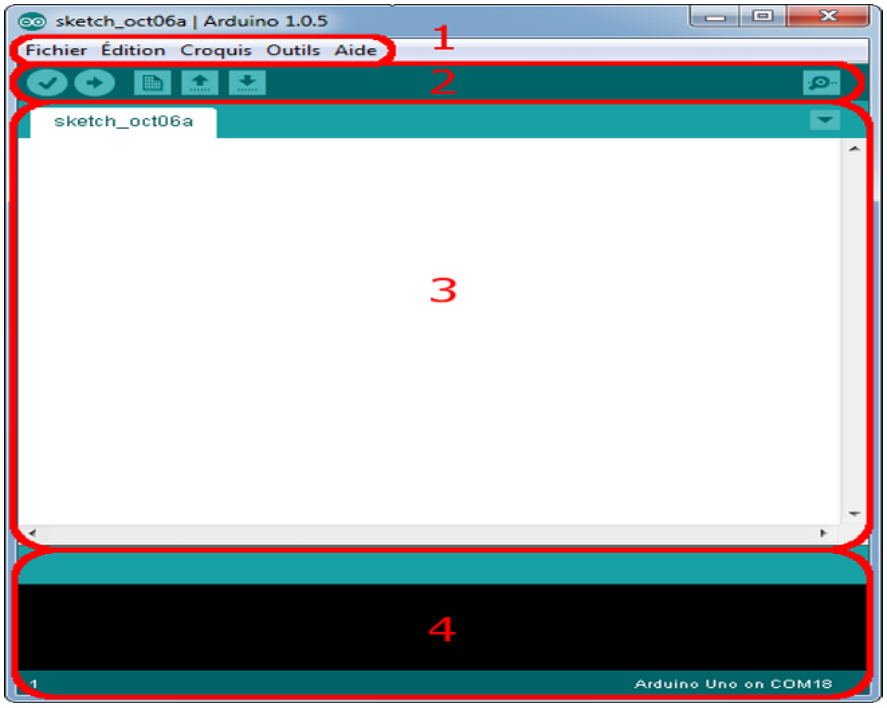


Figure II.**18:** Présentation de logiciel Arduino

- Barre 1 : **Barre de menu** : Ce sont les options de configuration du logiciel tel que :

  - Créer un nouveau programme ou en ouvrir un existant.

  - Enregistrer le document en cours ou choisir un emplacement pour l'enregistrement.

  - Afficher une liste déroulante contenant les noms d'exemples de programmes existants,

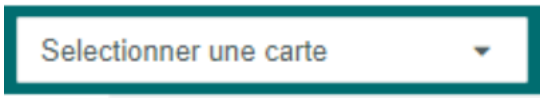
- Barre 2 : **Barre d'outils** : Il comporte les boutons nécessaires à la programmation de nos cartes.

Figure II.**19 :** Présentation des boutons

Vérifier (Verify) : vérifier les erreurs dans le code

Charge (Upload) : compiler le code et charge le programme sur la carte Arduino

Téléverser avec option avancer ou préférences



Sélectionner une carte

Traceur série : Permet d'afficher graphiquement les données envoyées sue le port série

Serial Monitor : permet d'accéder au port série (en RX/TX) et d'afficher les données textuellement envoyées par la carte

- Barre 3 : **Zone d'édition du code (éditeur)** : Ce bloc va contenir le programme que nous allons créer.
- Barre 4 : **Console de sortie** : Affiche les messages de compilation, de téléversement ou les messages envoyés par la carte.

III.1.2. Le code :

Avec Arduino, nous devons utiliser un code minimal lorsque l'on crée un programme. Ce code permet de diviser le programme que nous allons créer en deux grosses parties.

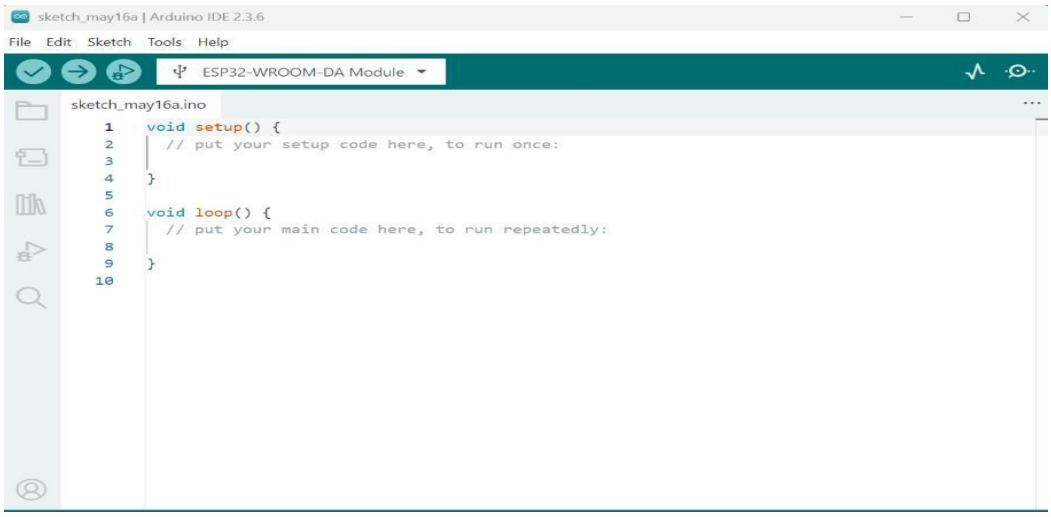


Figure II.20 : Le code

- Void setup () :fonction d'initialisation de la carte.
- Void loop () fonction principale qui se répète (s'exécute) à l'infini .

III.1.3. Installation de l'ESP32 :

Ce menu apparaît lorsque tu vas dans :

File > Preferences

Très important pour ESP32 !

C'est ici qu'on ajoute l'URL suivante pour installer les cartes ESP32 :

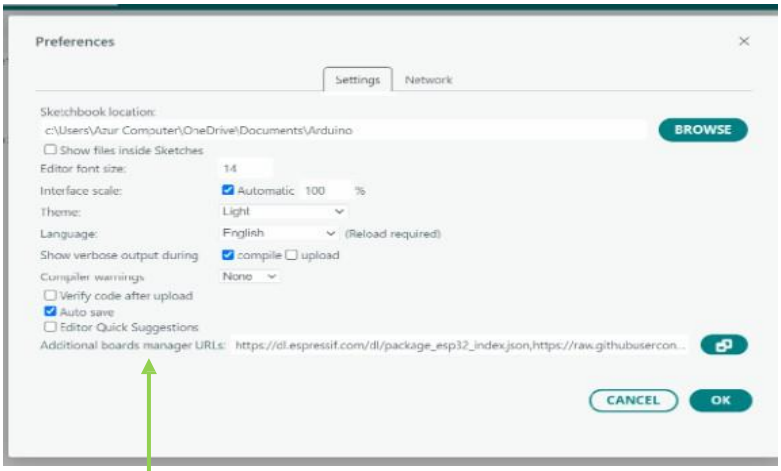


Figure II.21 : Interface Preferences

Les liens à copier dans les préférences de l'IDE Arduino pour installer l'ESP32 sont : https://dl.espressif.com/dl/package_esp32_index.json, https://raw.githubusercontent.com/espressif/arduino-esp32/gh-pages/package_esp32_index.json

III.1.4. Installation des bibliothèques :

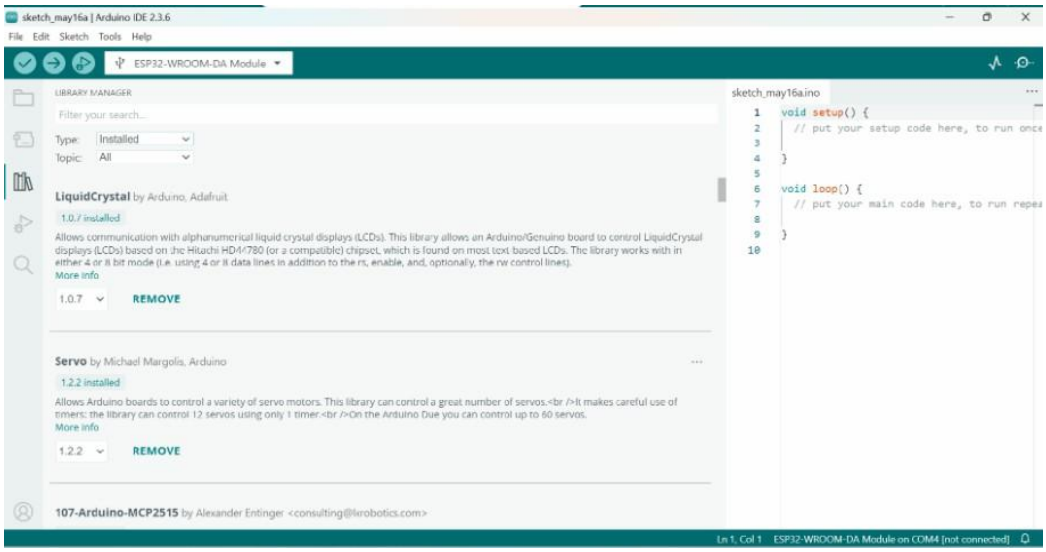


Figure II.22 : Interface instalation bibliothéque

Ce menu apparaît lorsque tu vas dans :

Clique sur **Croquis > Inclure une bibliothèque > Gérer les bibliothèques…**

Une fenêtre nommée **Library Manager** s'ouvre.

Dans la barre de recherche, tape le nom de la bibliothèque que tu veux installer

Sélectionne la version souhaitée (par défaut la plus récente).

Clique sur **Installer**.

III.1.5. Ajout de bibliothèques dans l'Arduino IDE :

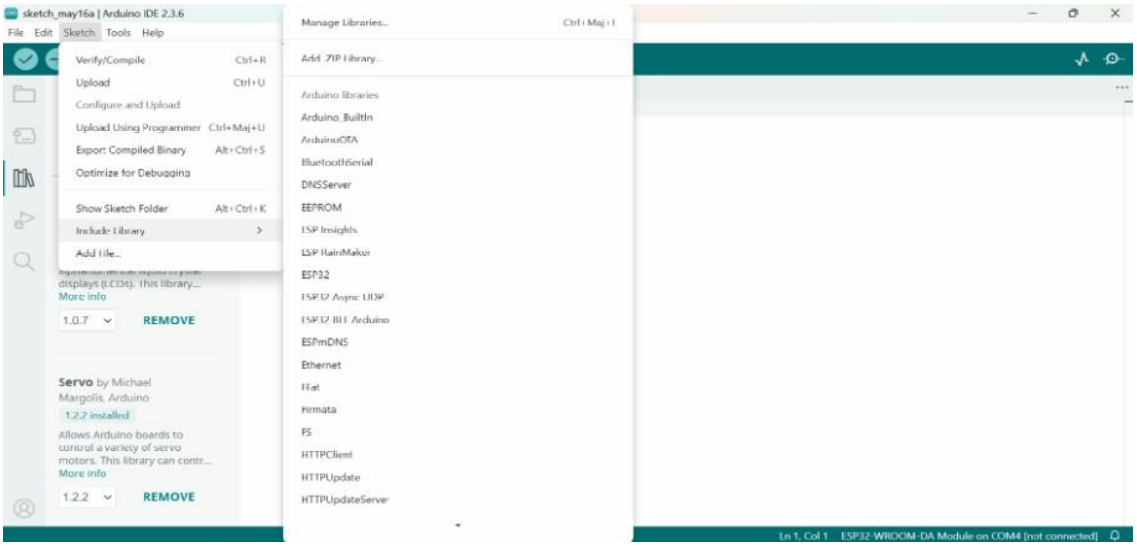


Figure II.23 : Interface insertion bibliothèque

Ce menu apparaît lorsque tu vas dans :

Croquis (Sketch) > Inclure une bibliothèque (Include Library)

Il sert à ajouter des bibliothèques logicielles à mon projet Arduino (appelé *sketch*), pour utiliser des fonctionnalités avancées sans devoir les programmer manuellement. C'est une étape importante pour utiliser des composants spécifiques.

III.1.6. La sélection de la carte et du port série :

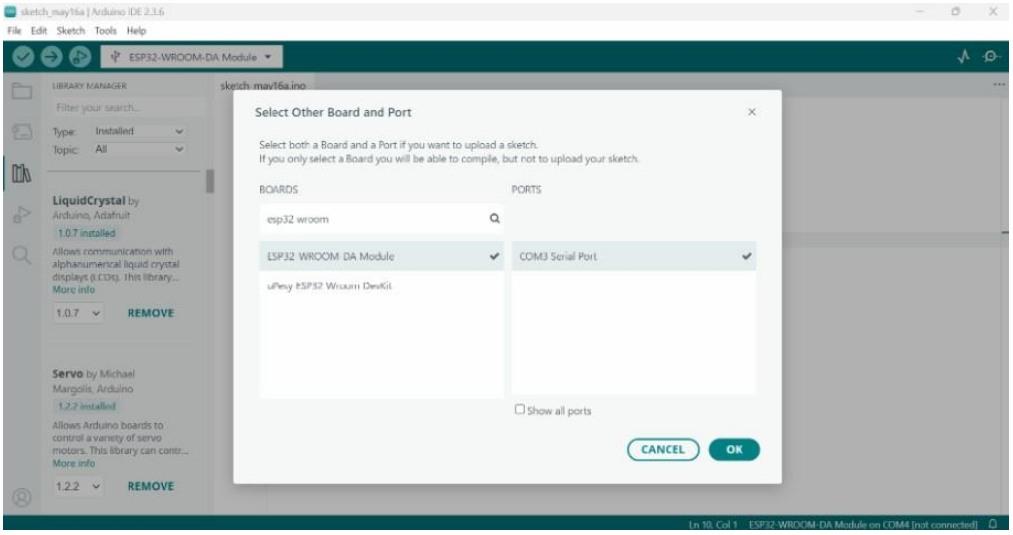


Figure II.24 : Interface de sélection de Board et port

Ce menu apparaît lorsque tu vas dans :

Va dans **Outils > Type de carte > Sélectionner une autre carte …** ou clique directement sur la barre en haut (là où le nom de la carte est affiché).

Une fenêtre s'ouvre avec les différentes **cartes et ports disponibles**.

Sélectionne :

- La **carte**.
- Le **port COM** auquel la carte est connectée.

Clique sur **OK**.

Dans notre cas, nous devons choisir la carte ESP32 WROOM DA Module ainsi que le port choisi .

Il est également possible d'effectuer cette opération de la manière suivante :

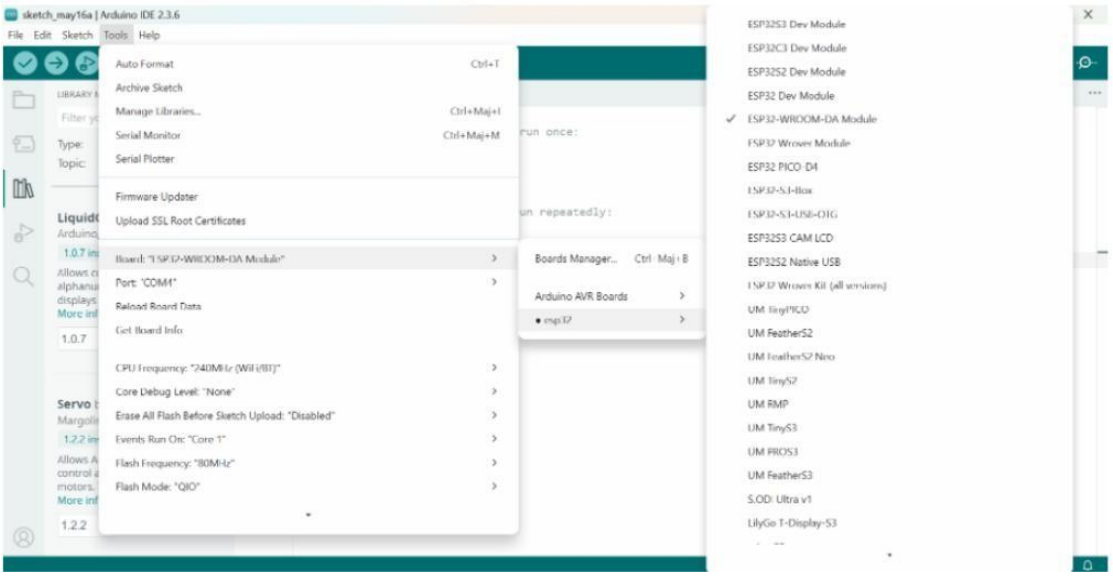


Figure II.25 : Sélection de Board et port

Cliquer sur **(Tools)** (*Outils*).

Passer la souris sur **(Board:)** (Type de carte) :

→ Une liste déroulante s'affiche avec les cartes disponibles.

Choisir la carte utilisée dans le projet (ESP32-WROOM-DA-Module)

Si la carte souhaitée n'apparaît pas :

- Cliquer sur **(Board Manager…)**
- Chercher la carte (par exemple : ESP32)
- Cliquer sur (**Install**) pour l'ajouter.

III.1.7. Transfert des programmes vers la carte :

On suppose qu'un programme et bien écrit (il ne contienne pas des erreurs) se trouve dans la fenêtre **EDITEUR.** Pour une première programmation de la carte, aller dans le menu **File>Exemples>Digital>Blink :** un programme s'ouvre avec du code dans la fenêtre éditeur.

Par la suit on vérifie si le programme et correcte ont appuyant sur le bouton **Verify** pour lancer la vérification.

Si tout se passe correctement, aucun message d'erreur ne doit apparaître dans la console, et la zone de message doit afficher **"Done Compiling"**, ce qui confirme que la vérification s'est effectuée avec succès.

Avant de téléverser le programme sur la carte, il convient de s'assurer que la carte et le port appropriés ont été correctement sélectionnés, comme indiqué dans les étapes précédentes.

Une fois que le bon port série et la bonne carte est sélectionné, on clique sur le Bouton **Téléverser** dans la barre d'outils, ou bien on sélectionne menu **fichier > Transférer vers la carte.**

Une fois le transfert terminé, le logiciel affiche soit un message confirmant la réussite de l'opération, soit des messages d'erreur en cas de problème**.**

III.2. MIT app inventor :

MIT App Inventor est une application web open source. Elle permet aux utilisateurs de créer des applications Android. L'application MIT Inventor a été fourni par Google, et maintenu par le Massachusetts Institute of Technology (MIT) dont les inventeurs sont Mark Friedman et le MIT professeur Hal Abelson. Elle a été conçue pour permettre à toute personne, même sans expérience en programmation, de créer facilement des applications fonctionnelles grâce à une interface graphique intuitive basée sur le principe du glisser-déposer.

MIT App Inventor permet de concevoir des applications en associant des composants visuels (boutons, textes, images, capteurs, etc.) à des blocs de programmation visuelle. Cette approche élimine la nécessité d'écrire du code complexe, tout en conservant une grande puissance fonctionnelle. Elle est particulièrement utile pour le prototypage rapide et les projets de systèmes embarqués, comme les systèmes de surveillance ou les objets connectés utilisant le Bluetooth.

Pour accéder à la plateforme et commencer à créer des applications, rendez-vous sur : https://appinventor.mit.edu

III.2.1. Interface de MIT App Inventor :

L'environnement de développement de MIT App Inventor est composé de deux parties principales : le Designer (pour concevoir l'interface utilisateur) et l'éditeur de blocs (Blocks Editor) pour programmer les comportements de l'application.

III.2.1.1. Le designer :

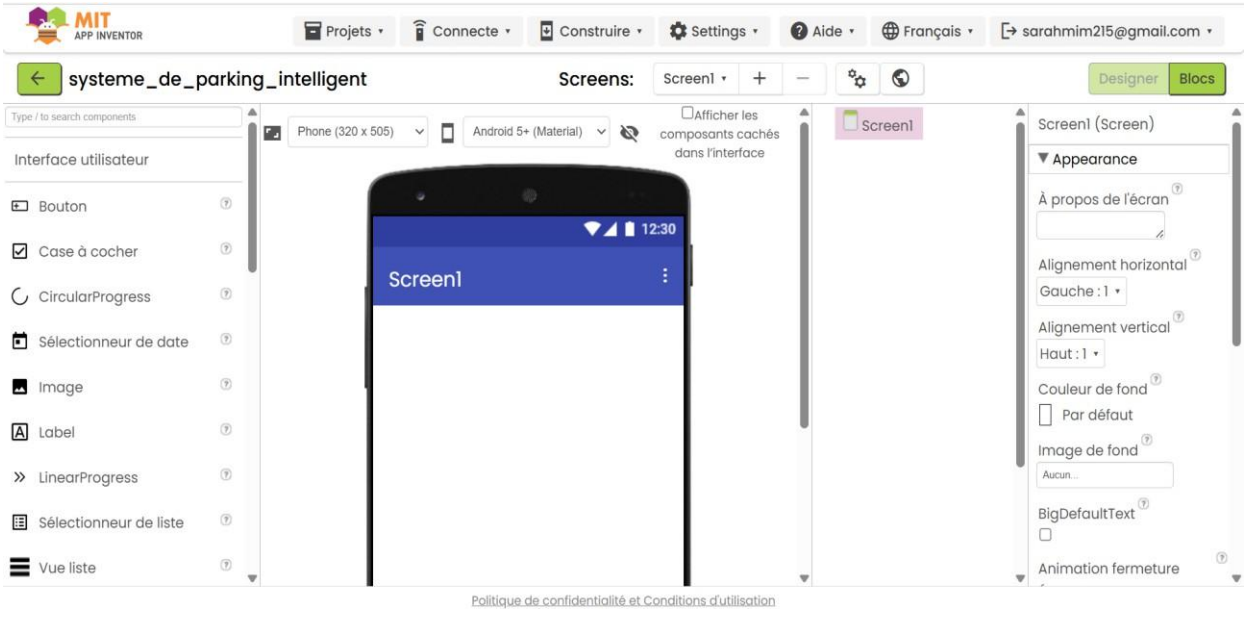


Figure II.26: fenêtre de création de l'interface sur App Inventor

Le Designer est l'espace où l'on construit l'apparence visuelle de l'application. Il contient plusieurs zones principales, chacune jouant un rôle spécifique dans la création de l'interface utilisateur. Voici les éléments essentiels :

- **Palette:** Contient tous les composants disponibles que l'utilisateur peut utiliser dans son application. Il suffit de glisser les composants depuis la palette vers l'écran de l'application.
- **Viewer (Aperçu) :** Au centre de la fenêtre, c'est la zone principale où l'on visualise l'écran du téléphone et où l'on dispose les composants pour construire l'interface.
- **Components (Composants) :** Située en bas à droite, cette zone affiche la liste des composants ajoutés à l'application, à la fois visuels et non-visuels. On peut les sélectionner, les renommer ou les organiser.
- **Properties (Propriétés) :** Placée à droite, elle permet de configurer les propriétés du composant sélectionné dans le Viewer ou la liste des composants. Par exemple, on peut modifier le texte d'un bouton, sa couleur, sa taille, ou encore le nom interne du composant.
- **Menu supérieur (Barre d'outils) :** En haut de la page, elle contient les fonctions générales : enregistrer, exporter l'application, accéder à l'éditeur de blocs, se connecter à un appareil mobile pour tester l'application, etc.

III.2.1.2. Blocks Editor :

L'éditeur de blocs est la partie dédiée à la programmation logique de l'application. Il permet de créer des scripts à l'aide de blocs colorés représentant des instructions, des événements, des conditions ou des boucles. MIT App Inventor utilise une approche visuelle on assemble des blocs logiques comme des pièces de puzzle pour définir le comportement de l'application. Cette méthode facilite la compréhension du fonctionnement de l'application et évite les erreurs de syntaxe propres au code classique.

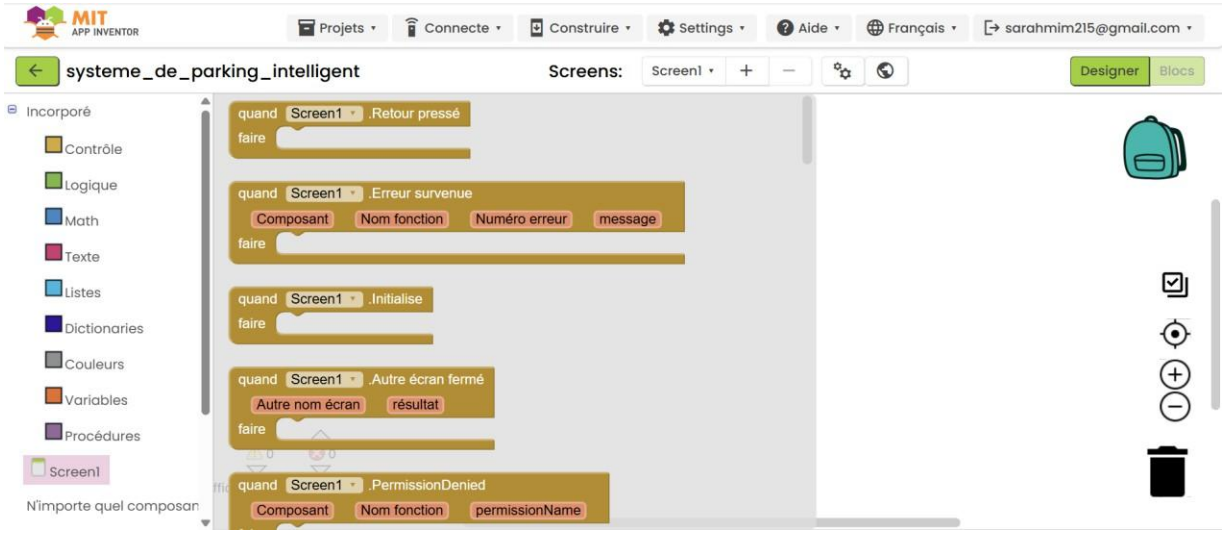


Figure II.**27:** Interface de l'éditeur de blocs

III.2.2. Tester l'application :

Après avoir conçu l'interface utilisateur et programmé la logique dans MIT App Inventor, il est important de tester l'application en temps réel afin de s'assurer qu'elle fonctionne correctement. MIT App Inventor permet cela de manière simple grâce à l'application mobile MIT AI2 Companion, disponible gratuitement sur Android. Voici les étapes à suivre :

- **Installer l'application MIT AI2 Companion :** Télécharger et installer l'application MIT AI2 Companion depuis le Google Play Store sur le smartphone Android. Cette application agit comme un pont entre le téléphone et la plateforme de développement.
- **Connexion à l'interface MIT App Inventor :** Sur l'interface web de MIT App Inventor (dans le navigateur), cliquer sur le bouton **"Connect"** dans le menu du haut. Puis sélectionner **"AI Companion"**.
- **Scanner le QR code :** Un QR code s'affiche à l'écran de l'ordinateur. Il suffit alors d'ouvrir l'application MIT AI2 Companion sur le téléphone, puis utiliser la fonction scanner pour lire le QR code affiché. En alternative, il est aussi possible de saisir manuellement un code à 6 caractères afficher juste au-dessus du QR code.
- **Tester en temps réel :** Une fois le smartphone connecté à MIT App Inventor via l'application MIT AI2 Companion , l'application en cours de développement devient instantanément visible sur l'écran du téléphone . L'utilisateur peut alors prévisualiser et tester l'interface ainsi que la réponse des blocs programmés en temps réel. Toute

modification effectuée sur le site que ce soit dans Designer ou dans l'éditeur de blocs est immédiatement synchronisée avec le téléphone . Cela permet de tester, corriger et ajuster l'application rapidement et facilement , sans avoir à la recompiler à chaque modification .

III.3. Base de données :

Pour assurer un bon fonctionnement du système de parking intelligent, il est essentiel d'établir un lien entre le microcontrôleur et l'application mobile. Ce lien permet d'échanger des informations en temps réel. Dans notre projet, ce lien est assuré par une base de données en ligne qui joue un rôle central dans la communication entre les deux parties. Nous avons choisi d'utiliser Firebase [18].

Firebase est une plateforme de développement d'applications mobiles et web proposée par Google, qui offre un ensemble complet de services backend permettant de créer des applications performantes, sans avoir à gérer soi-même un serveur.

Elle propose plusieurs fonctionnalités, dont les plus utilisées sont :

- **Firebase Realtime Database** : une base de données en temps réel qui permet de stocker et de synchroniser les données entre les utilisateurs et les appareils instantanément. Elle est particulièrement utile dans les projets connectés, comme notre projet où l'état des places de stationnement peut changer à tout moment.
- **Firebase Authentication** : pour gérer facilement l'identification et la connexion des utilisateurs via email, mot de passe ou réseaux sociaux.
- **Firebase Cloud Messaging (FCM)** : pour envoyer des notifications push aux utilisateurs, même lorsque l'application est fermée. Cela peut être utilisé.

Dans le cadre de ce projet, Firebase a été utilisé comme passerelle entre le système de stationnement géré par la carte ESP32 et l'application mobile, permettant une communication fluide et instantanée des données relatives à l'occupation des places.

III.4. Fritzing :

Fritzing est une suite logiciel open-source dont la vocation est l'électronique. L' équipe Fritzing a créé Fritzing et permet la conception assistée par ordinateur dans le contexte de l'électronique, et surtout pour les makers et les amateurs passe amateur . Fritzing est composé majoritairement de trois vues intégrées : vue schématique, vue breadboard et vue PCB. De nombreuses

communautés, écoles et amateurs d'électronique sont à l'emploi de cette suite logicielle. En plus de sa vogue, Fritzing présente plusieurs autres points de force :

- Un logiciel intuitif et accessible, idéal pour les débutants.
- Une communauté active et un large partage de ressources.
- Un outil de prototypage visuel qui facilite la transition entre le concept et la réalisation matérielle.

Fritzing est la méthode utilisée dans cette recherche pour la conception et la simulation du système. Il est surtout connu par ses capacités à créer des schémas électriques fidèles et des prototypes visuels sur breadboard. Outre cela , ce logiciel permet la conception des circuits imprimés (PCB) et la simulation des connexions, qui servent à repérer les erreurs en phase de conception. Indirectement , les schémas dessinés avec Fritzing peuvent être appliqués dans les documentations grâce à son contrôle précis des aspects graphiques des circuits. Son interface se présente comme suit :

Figure II.28 : Interface Fritzing

IV. Conclusion :

Au cours de ce chapitre, nous avons fourni une vue d'ensemble sur le projet, en détaillant ses deux parties principales. Dans la partie matérielle, nous avons décrit les composants utilisés, notamment la carte ESP32, ainsi que les capteurs et actionneurs associés. Ensuite, dans la partie logicielle, nous avons expliqué l'utilisation du logiciel Arduino pour la programmation de la carte, ainsi que le développement de l'application mobile via MIT App Inventor.

Enfin, nous avons mis en lumière l'intégration de ces éléments pour assurer le bon fonctionnement du système.

Chapitre III :Réalisation et tests

I. Introduction :

Dans la continuité des chapitres précédents, où une vue globale sur les parkings intelligents ainsi que les composants utilisés ont été exposée, le présent chapitre est consacré à la réalisation concrète du système développé. Le projet mis en œuvre allie une application mobile dédiée à la réservation et au paiement avec un système de gestion physique automatisée d'un parking sécurisé.

II. Présentation du projet :

Ce projet a pour objectif de concevoir un système de gestion intelligente de stationnement, combinant une application mobile intuitive avec une infrastructure connectée, afin d'optimiser l'expérience utilisateur et la fluidité du trafic dans les parkings. Grâce à l'intégration de capteurs, d'un microcontrôleur ESP32 et d'une base de données en ligne, ce système permet une réservation en temps réel, une détection automatique des véhicules et une gestion sécurisée des accès.

L'utilisateur débute par la création d'un compte via l'application, dans lequel il renseigne ses informations personnelles ainsi que celles de son véhicule. Une fois connecté, il accède à une interface lui permettant de consulter les places disponibles en temps réel, de sélectionner une place et de définir l'heure d'arrivée et de sortie, sous réserve que l'heure d'entrée soit d'au moins une heure postérieure à la réservation. Après validation et paiement, l'application génère un code de réservation unique, à usage unique, et valide uniquement pendant la période définie lors de la réservation.

Lorsqu'un utilisateur finalise sa réservation via l'application, l'interface affiche la place réservée en orange, indiquant visuellement qu'elle est bloquée pour un utilisateur mais pas encore occupée physiquement. Les places déjà occupées apparaissent en rouge, tandis que les places entièrement libres sont affichées en vert. Seules les places vertes sont cliquables et donc sélectionnables lors de la réservation, assurant ainsi une gestion fluide et cohérente du stationnement. Cette distinction colorée améliore l'ergonomie de l'application et permet au conducteur de visualiser en un coup d'œil la disponibilité du parking.

Sur site, le conducteur saisit ce code au niveau de l'entrée du parking via un clavier numérique. Si le code est reconnu comme valide (grâce à une correspondance via Firebase), la barrière d'entrée se lève automatiquement. L'utilisateur se dirige alors vers la place qui lui a été attribuée, chaque emplacement étant équipé d'un capteur infrarouge détectant la présence du véhicule, et transmettant son état à l'application à travers la base de données en ligne.

Pour des raisons de sécurité, le parking est également équipé de capteurs de flamme. La Figure III.01 illustre les différentes actions déclenchées automatiquement en cas de détection d'un début d'incendie :

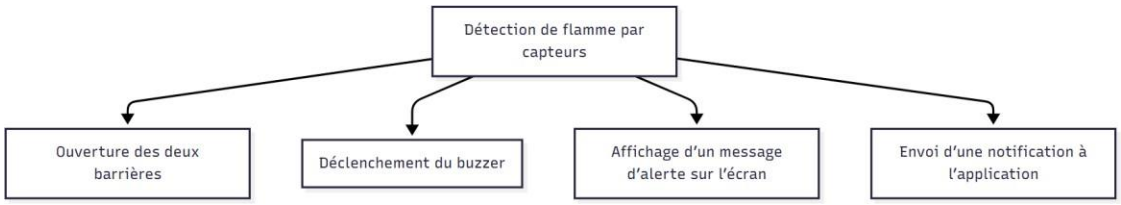
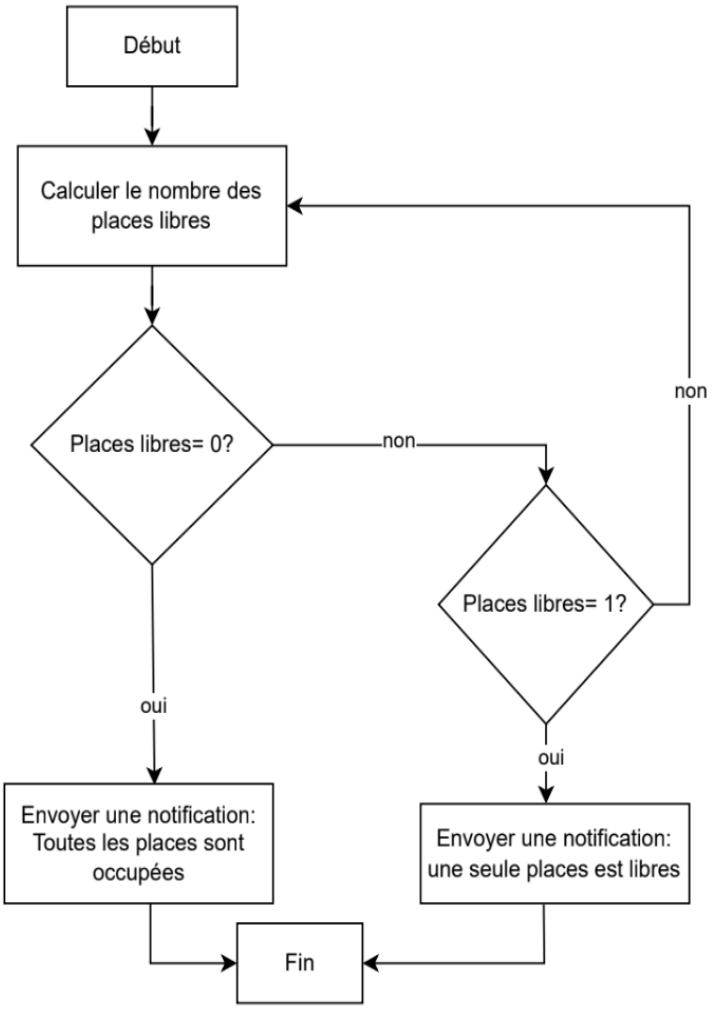


Figure III.01 : Les actions déclenchées dans le cas de détection de flamme.

En complément, un autre mécanisme de notification est mis en place selon l'état d'occupation du parking. Lorsque toutes les places sont réservées ou occupées, une notification est envoyée aux utilisateurs pour signaler que le parking est complet. Lorsqu'il ne reste qu'une seule place ou lorsqu'une place vient d'être libérée, une notification est également envoyée pour indiquer cette disponibilité.

Cette logique permet de maintenir les utilisateurs informés en temps réel et d'optimiser la gestion des places. La Figure III.02 illustre ce processus décisionnel basé sur l'état d'occupation.



La Figure III.02 : Organigramme de gestion des notifications selon l'occupation du parking

L'ensemble du système repose sur un microcontrôleur ESP32, chargé d'interagir en temps réel avec les différents capteurs, actionneurs et la base de données. Toute modification de l'état des capteurs est transmise vers Firebase, assurant ainsi une synchronisation constante entre l'état physique du parking et l'application mobile.

III. Conception de parking intelligent :

III.1. Les étapes de créations l'application :

Comme on a déjà expliquer dans le chapitre 2, notre application a été créé dans le site MIT APP INVENTOR qui se compose de deux interfaces : l'une pour la visualisation et l'insertion des composants et l'autre pour la construction des blocs de programmation.

Notre application comprend plusieurs screens, chacun jouant un rôle spécifique dans le parcours utilisateur. Ces screens sont détaillés comme suit :

- **Screen de bienvenue :**

C'est le tout premier screen que voit l'utilisateur. Il donne le ton de l'application avec un visuel soigné : label "Bienvenue à notre application", image significative d'un parking, et un bouton "Suivant". L'objectif ici est simple : accueillir l'utilisateur dans un environnement clair et professionnel, tout en créant une première impression positive.

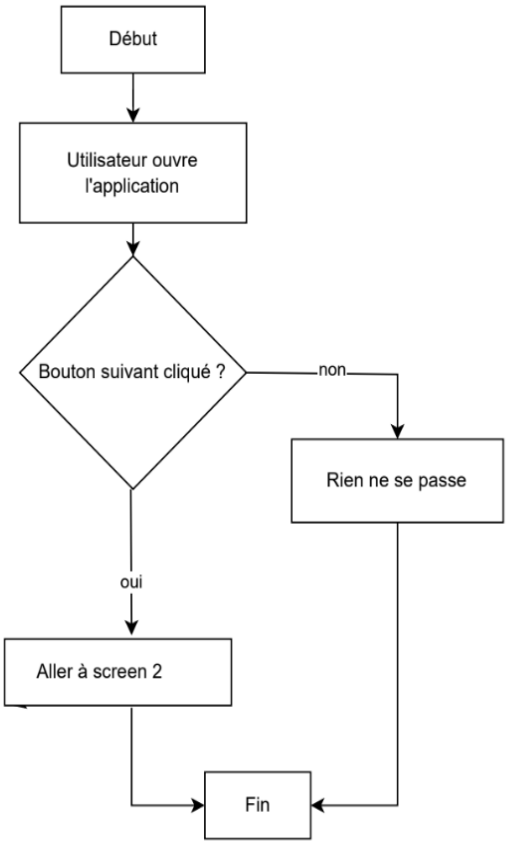


Figure III.03 : Organigramme de fonctionnement de Screen 1

Figure III.04 : Interface de Screen 1

- **Screen de choix : Inscription ou Connexion :**

Dès que l'utilisateur appuie sur "Suivant", il accède à un screen proposant deux chemins distincts :

- Créer un compte s'il s'agit d'un premier usage,
- Se connecter s'il a déjà un profil.

Ce point de passage est fondamental pour garantir un parcours utilisateur logique et fluide.

Figure III.05 : Interface de Screen 2

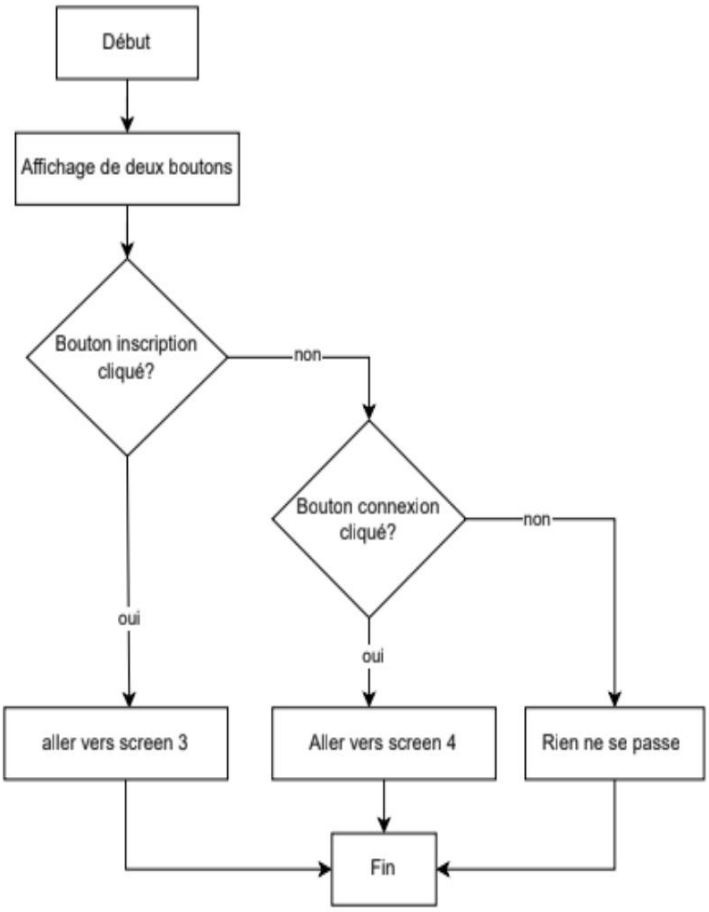


Figure III.06 : Organigramme de fonctionnement de Screen 2

Screen d'inscription :

Sur cet écran, l'utilisateur est invité à renseigner ses informations :

- Son nom complet
- Le type de véhicule
- Le numéro d'immatriculation
- Le numéro de contrat d'assurance
- Un mot de passe

Une fois le formulaire complété, les données sont stockées pour permettre l'accès futur. Des blocs logiques sont utilisés pour valider les champs et stocker les informations dans TinyDB.

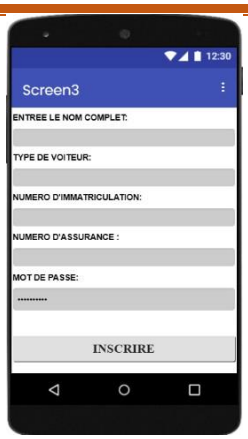


Figure III.07 : Interface de Screen 3

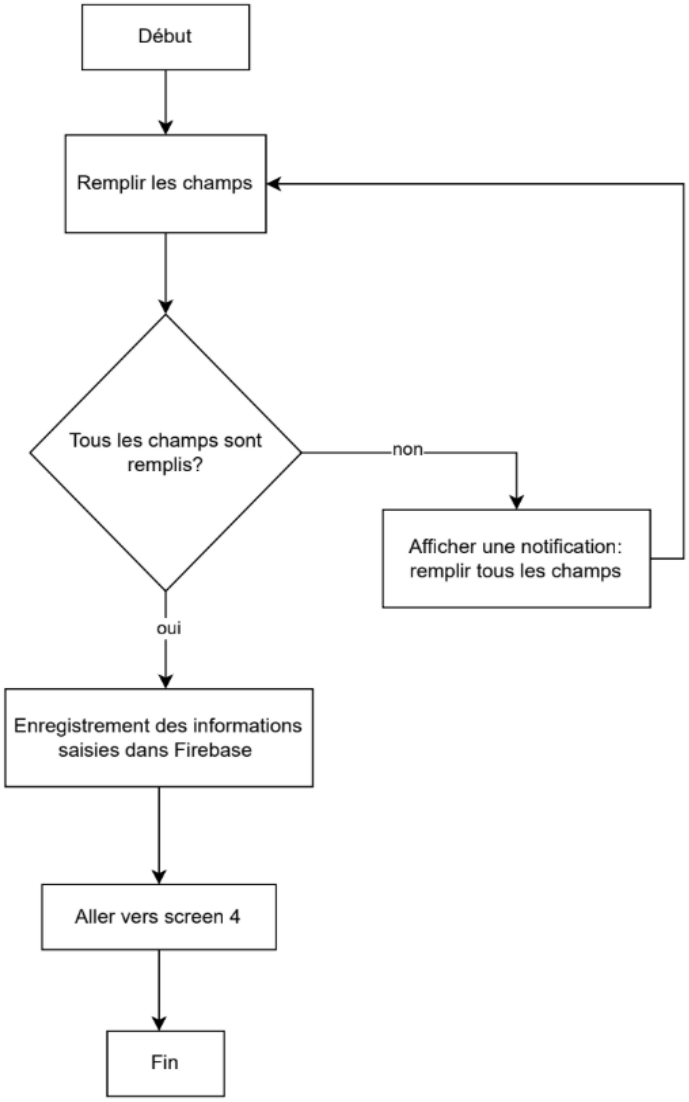


Figure III.08 : Organigramme de fonctionnement de Screen 3

Screen de connexion :

L'écran de connexion est conçu pour être simple et intuitif.

L'utilisateur renseigne son nom complet et son mot de passe. Le système vérifie automatiquement les informations saisies en les comparant aux données enregistrées.

- Si l'un des champs est vide, un message d'alerte s'affiche pour inviter l'utilisateur à compléter tous les champs obligatoires.
- Si les informations ne correspondent pas, un message d'erreur informe l'utilisateur qu'il y a une erreur au niveau de l'identifiant ou du mot de passe.

Figure III.09 : Interface de Screen 4

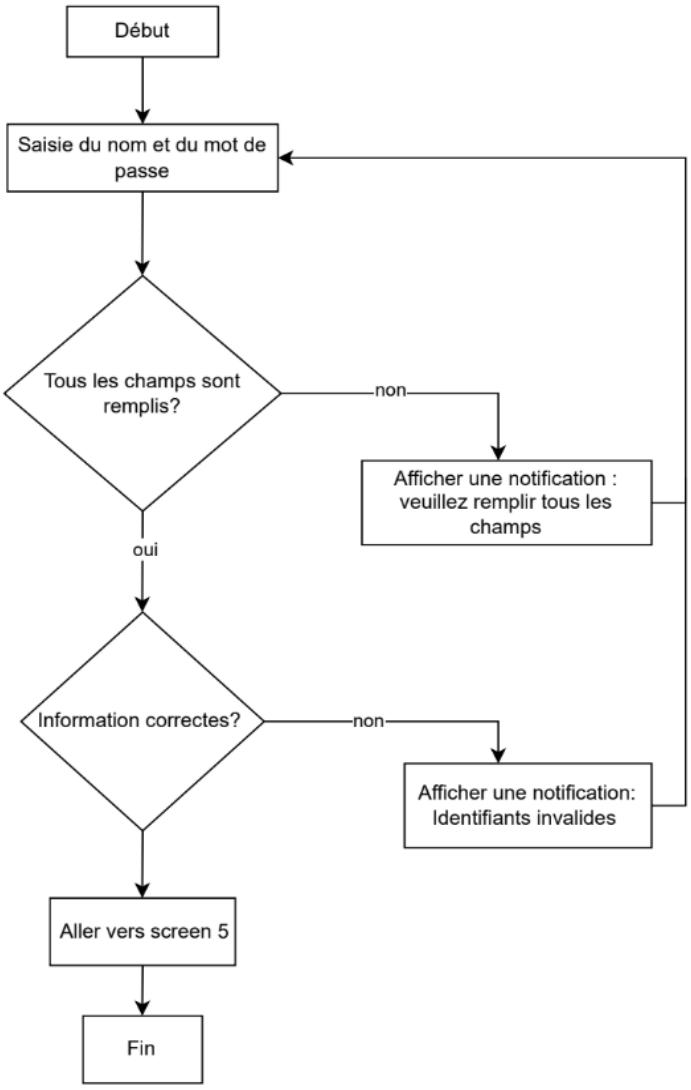


Figure III.10 : Organigramme de fonctionnement de Screen 4

Screen de choix : Réservation ou historique

Une fois connecté, l'utilisateur accède au screen principal qui regroupe deux fonctions essentielles :

- La réservation d'une place de parking,
- La consultation de l'historique de ses réservations précédentes.

Deux boutons principaux permettent de naviguer facilement vers l'action souhaitée. Cet écran constitue le point de départ fonctionnel de l'application.

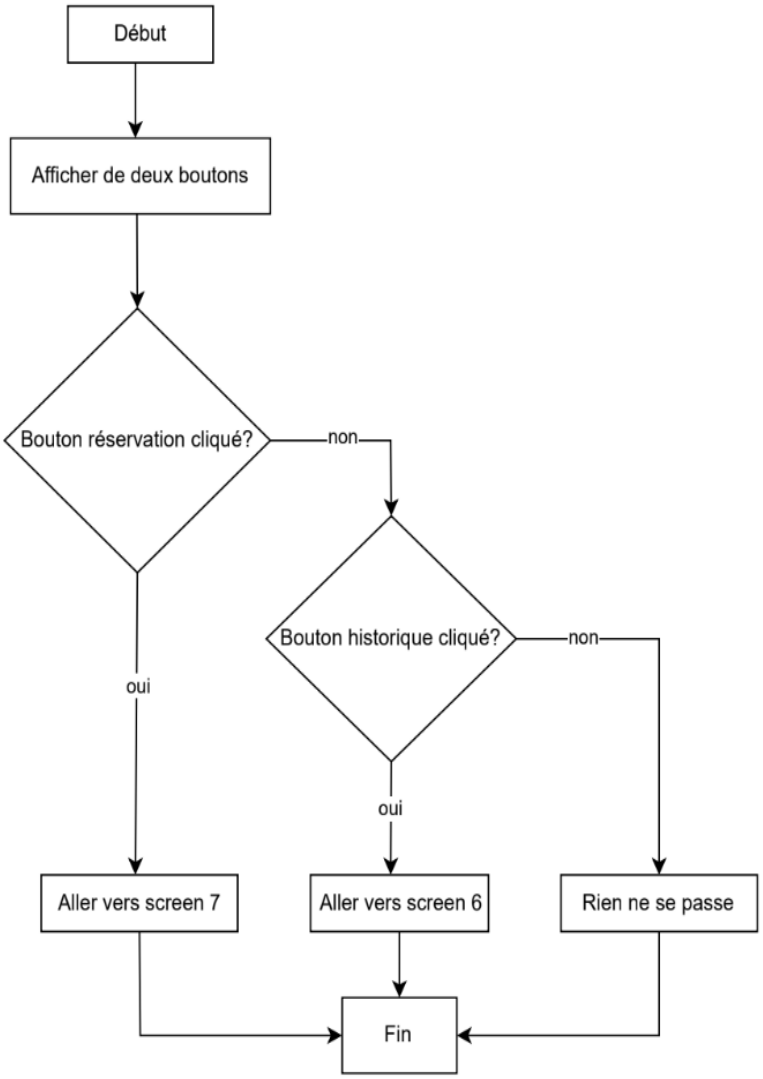


Figure III.11 : Organigramme de fonctionnement de Screen 5

Figure III.12 : Interface de Screen 5

Screen de réservation :

Ce screen affiche la liste des places de parking, avec leur état actuel :

- Libre (en vert),
- Occupée (en rouge),
- Réservée (en orange).

Seules les places libres en vertes sont cliquables et donc sélectionnables lors de la réservation, assurant ainsi une gestion fluide et cohérente du stationnement.

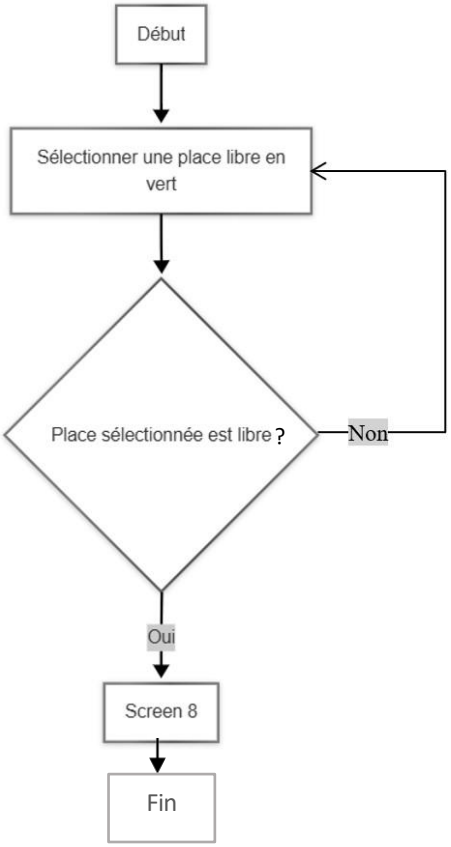


Figure III.13 : Organigramme de fonctionnement de Screen 7

Figure III.14 : Interface de Screen 7

Screen de sélection de la date et de l'heure

Sur ce screen, l'utilisateur choisit :

- La date du jour.
- L'heure d'entrée et de sortie souhaitées.

L'application utilise ces informations pour calculer automatiquement la durée totale du stationnement, puis en déduit le prix à payer selon la règle suivante :

- 100 DA par heure.
- 1 DA par minute supplémentaire.

Des blocs logiques intégrés dans l'application assurent une vérification rigoureuse des horaires saisis, notamment en imposant que l'heure d'entrée soit au minimum une heure après l'heure actuelle, cela garantit la cohérence de la réservation et un délai de préparation suffisant.

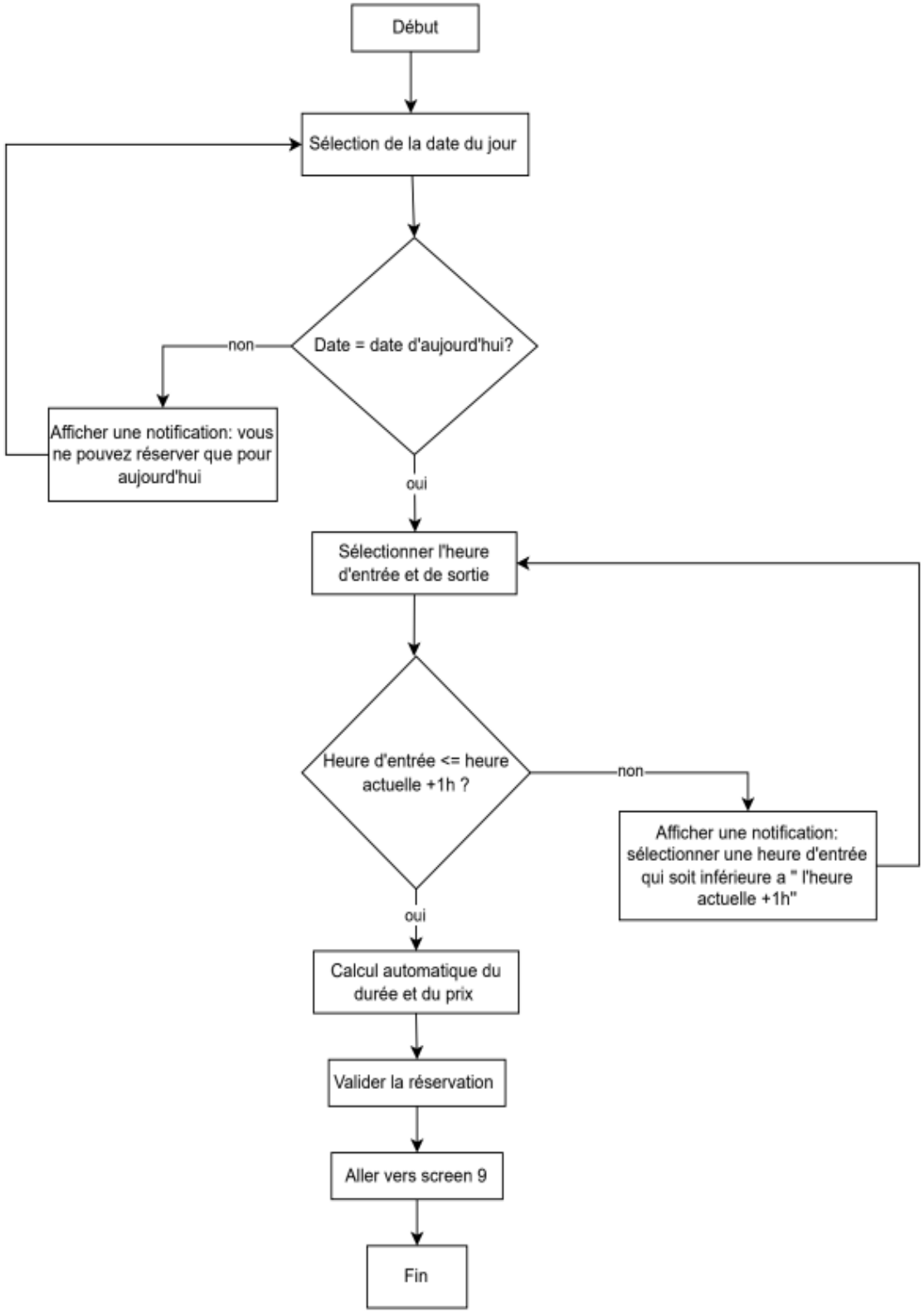


Figure III.15 : Organigramme de fonctionnement de Screen 8

Figure III.16 : Interface de Screen 8

- **Screen de paiement :**

Dans l'interface actuelle de l'application, un screen de paiement provisoire est mis en place à titre démonstratif. Il présente un formulaire contenant les informations classiques d'une carte bancaire, telles que :

- Nom figurant sur la carte.
- Numéro de la carte.
- Date d'expiration (MM/AA).
- CVV.

Figure III.17 : Interface de Screen 9

Intégration future du paiement en ligne via cartes CIB et Edahabia :

Dans le cadre d'une intégration réelle, deux solutions locales sont envisagées en Algérie pour offrir un service de paiement électronique sécurisé :

- **Paiement par carte CIB (SATIM)** :

L'intégration avec la plateforme SATIM permet d'accepter les paiements par carte bancaire CIB. Cela nécessite :

  - Une immatriculation légale (registre de commerce ou équivalent)
  - Une demande d'adhésion à SATIM
  - Un site web sécurisé avec HTTPS où sera intégrée la page de paiement

Une fois approuvée, SATIM fournit une passerelle de paiement sous forme de lien sécurisé, intégrable directement dans l'application à l'aide d'un composant WebViewer. L'utilisateur accède à une page de paiement intégrée en cliquant sur "Payer".

- **Paiement par carte Edahabia – via Algérie Poste**

Les détenteurs de cartes Edahabia peuvent quant à eux effectuer leur règlement via le service Baridi E-paiement, proposé par Algérie Poste. L'intégration suit une procédure similaire :

  - Faire une demande d'adhésion auprès d'Algérie Poste

- Obtenir, après validation, une interface de paiement en ligne (formulaire ou lien sécurisé)
- Intégrer cette interface dans l'application via le WebViewer

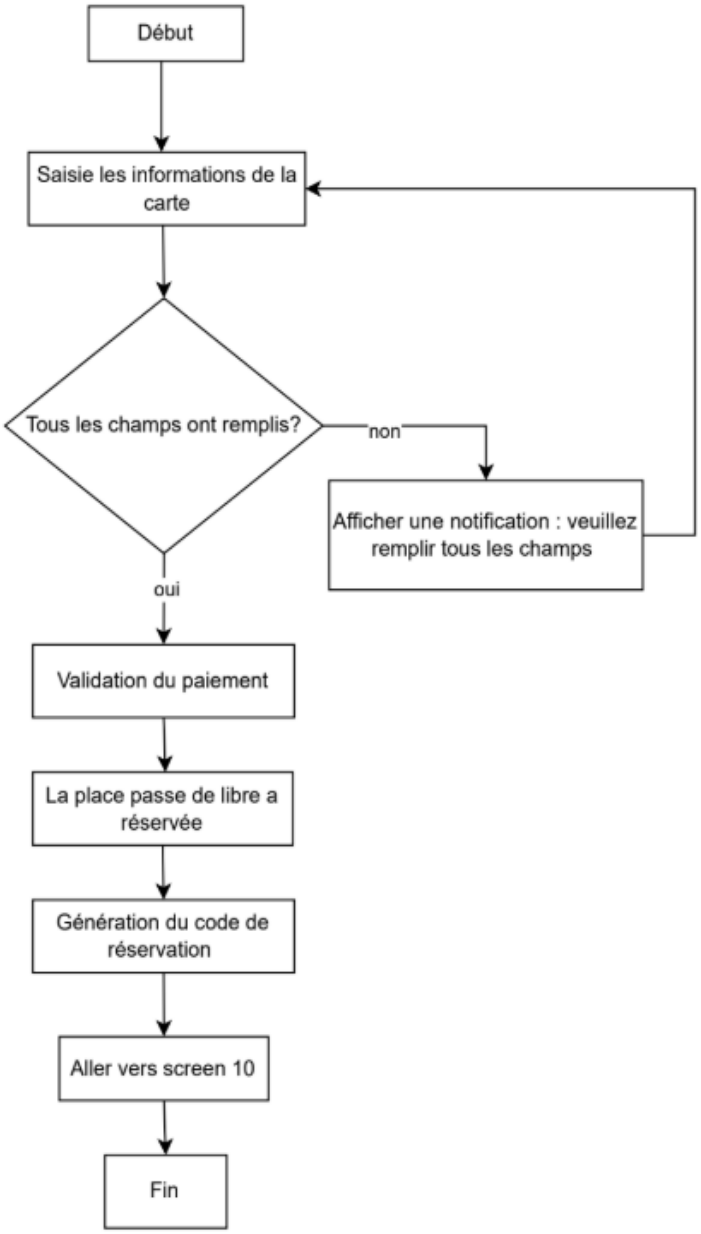


Figure III.18 : Organigramme de fonctionnement de Screen 9

Screen de génération de code de réservation :

Une fois le paiement validé et la réservation confirmée, un code d'accès aléatoire composé de 4 chiffres est automatiquement généré.

Ce code sert de clé d'entrée au parking et garantit un accès sécurisé à l'utilisateur.

Figure III.19 : Interface de Screen 10

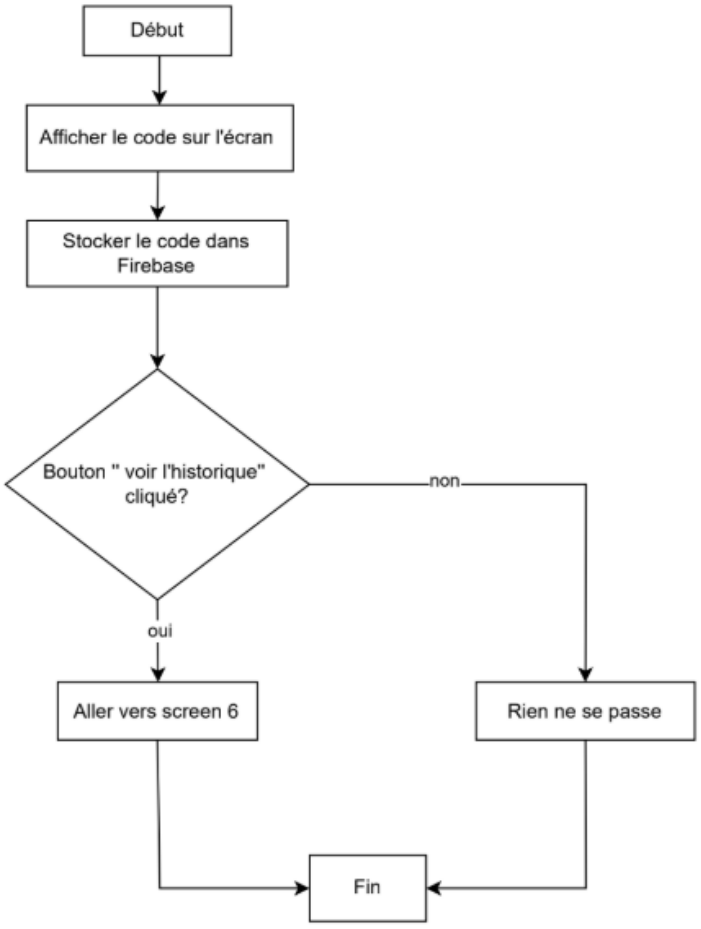


Figure III.20: Organigramme de fonctionnement de Screen 10

Screen d'historique

L'historique affiche toutes les réservations effectuées par l'utilisateur, avec les détails suivants: place réservée, date de réservation, heure d'entrée, heure de sortie, prix total et le code de réservation.

Cela permet à l'utilisateur de suivre ses usages et de justifier ses paiements si nécessaire. Les données sont stockées localement à l'aide des blocs TinyDB de MIT App Inventor.

Figure III.21 : Interface de Screen 6

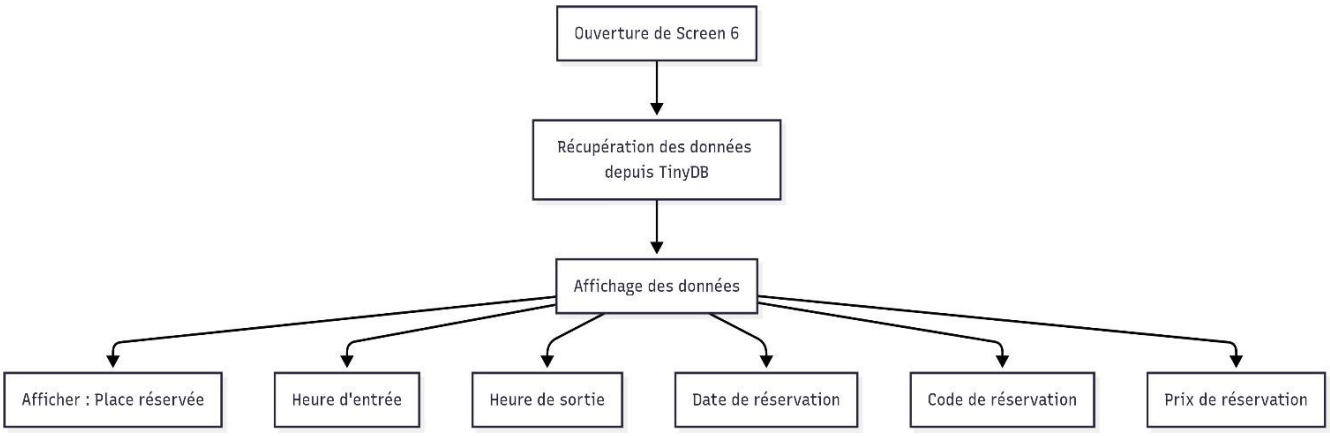


Figure III.22: Processus d'affichage de l'historique de réservation dans Screen 6

III.2. Schémas de branchements et programmations :

Afficheur LCD :

Le branchement se fait comme suit :

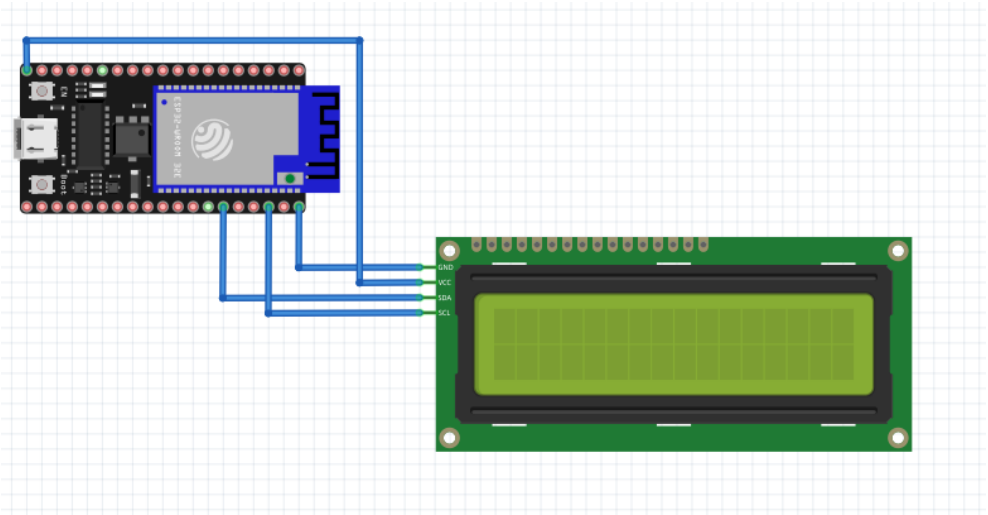


Figure III.23: Branchement de l'afficheur LCD I2C a la carte ESP32

Les broches associées sont mentionnées dans le tableau suivant :

Broche du capteur	**Rôle**	**Broche ESP32**
VCC	Alimentation à 5 V	5V
GND	Masse	GND
SDA	Données I2C	GPIO21
SCL	Horloge I2C	GPIO22

Tableau III.1 : Branchement de l'afficheur LCD I2C a la carte ESP32

Un exemple de programmation d'un afficheur LCD I2C :

FLAMME | Arduino IDE 2.3.6

File Edit Sketch Tools Help

ESP32-WROOM-DA Module

FLAMME.ino

```
#include <Wire.h>
#include <LiquidCrystal_I2C.h>

LiquidCrystal_I2C lcd(0x27, 16, 2);

void setup() {
  // Initialisation de la communication série (facultatif pour debug)
  Serial.begin(115200);

  // Initialisation de l'écran LCD
  lcd.init();            // Démarre l'écran
  lcd.backlight();       // Allume le rétroéclairage
  // Affiche le message de bienvenue
  lcd.setCursor(0, 0);
  lcd.print("Bienvenue au");
  lcd.setCursor(0, 1);
  lcd.print("Parking");
}

void loop() {
}
```

Figure III.24 : Exemple de programmation de l'afficheur

Capteur de flamme :

Le branchement se fait comme suit :

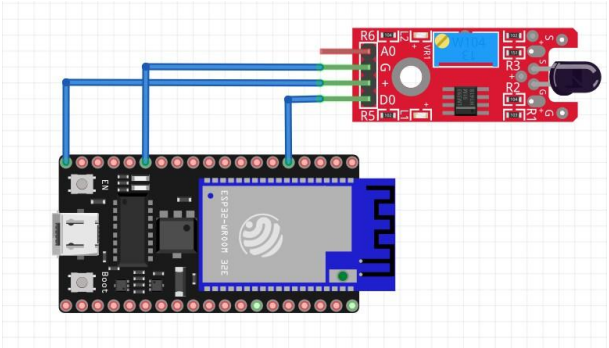


Figure III.25: Branchement de capteur de flamme a la carte ESP32

Dans notre conception, nous avons intégré deux capteurs de flamme afin d'assurer une couverture complète de la zone de stationnement en cas d'incendie, renforçant ainsi la sécurité des conducteurs et de leurs véhicules.

Les broches associées sont mentionnées dans le tableau suivant :

Broche du capteur de flamme	**Rôle**	**Broche ESP32**
VCC *(tous les capteurs)*	Alimentation à 5 V	5V *(partagé)*
GND *(tous les capteurs)*	Masse	GND *(partagé)*
D0 (capteur 1)	Sortie numérique (HIGH/LOW)	GPIO36
D0 (capteur 2)	Sortie numérique (HIGH/LOW)	GPIO39
A0 *(facultatif) (capteur 1et 2)*	Sortie analogique	NON

Tableau III.2 : Branchement des capteurs de flamme à la carte ESP32

Exemple de programmation de deux capteurs de flamme et un buzzer :

FLAMME | Arduino IDE 2.3.6

File Edit Sketch Tools Help

ESP32-WROOM-DA Module

FLAMME.ino

```
#define FLAME_SENSOR_1 36
#define FLAME_SENSOR_2 39
#define BUZZER_PIN     15

void setup() {
  Serial.begin(115200);
  pinMode(FLAME_SENSOR_1, INPUT);
  pinMode(FLAME_SENSOR_2, INPUT);
  pinMode(BUZZER_PIN, OUTPUT);
  Serial.println("Systeme de detection pret");
}
void loop() {
  bool flamme1 = digitalRead(FLAME_SENSOR_1) == LOW;
  bool flamme2 = digitalRead(FLAME_SENSOR_2) == LOW;

  if (flamme1 || flamme2) {
    digitalWrite(BUZZER_PIN, HIGH);
    Serial.println("!!! ALERTE FLAMME !!!");
  } else {
    digitalWrite(BUZZER_PIN, LOW);
    Serial.println("Pas de flamme");
  }
  delay(500);
}
```

Figure III.26 : Exemple de programmation des capteurs de flamme

Capteur Infrarouge :

Le branchement de capteur Infrarouge se fait comme suit :

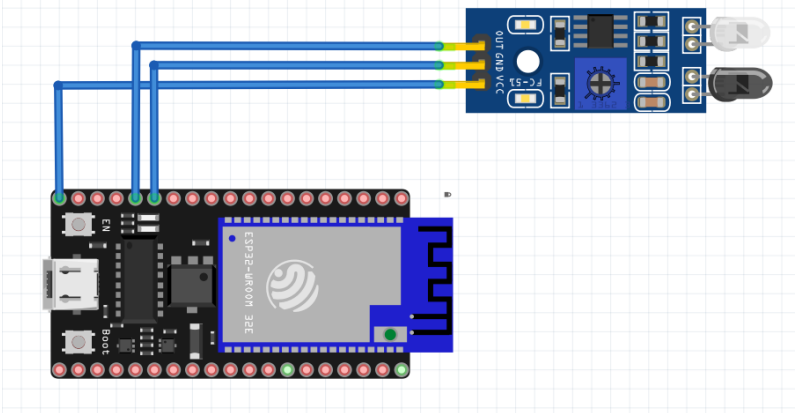


Figure III.27 : Branchement d'un capteur Infrarouge à la carte ESP32

Lors de la phase de réalisation, cinq capteurs infrarouges ont été utilisés : quatre positionnés sur les emplacements de stationnement pour détecter la présence de véhicules, et un cinquième installé à la sortie afin d'identifier les départs et déclencher automatiquement l'ouverture de la barrière de sortie.

Les broches associées des 5 capteur Infrarouge sont mentionnées dans le tableau suivant :

Broche du capteur IR	**Rôle**	**Branchement ESP32**
VCC *(tous les capteurs)*	Alimentation 5 V	5V *(partagé)*
GND *(tous les capteurs)*	Masse	GND *(partagé)*
D0 (capteur 1)	Sortie numérique	GPIO2
D0 (capteur 2)	Sortie numérique	GPIO5
D0 (capteur 3)	Sortie numérique	GPIO13
D0 (capteur 4)	Sortie numérique	GPIO14
D0 (capteur 5)	Sortie numérique	GPIO4

Tableau III.3 : Branchement des 5 capteurs Infrarouge à la carte ESP32

Voici un exemple de programmation de 5 capteurs Infrarouge connectés à un ESP32, ce code affiche dans le moniteur série le nombre de places libres :

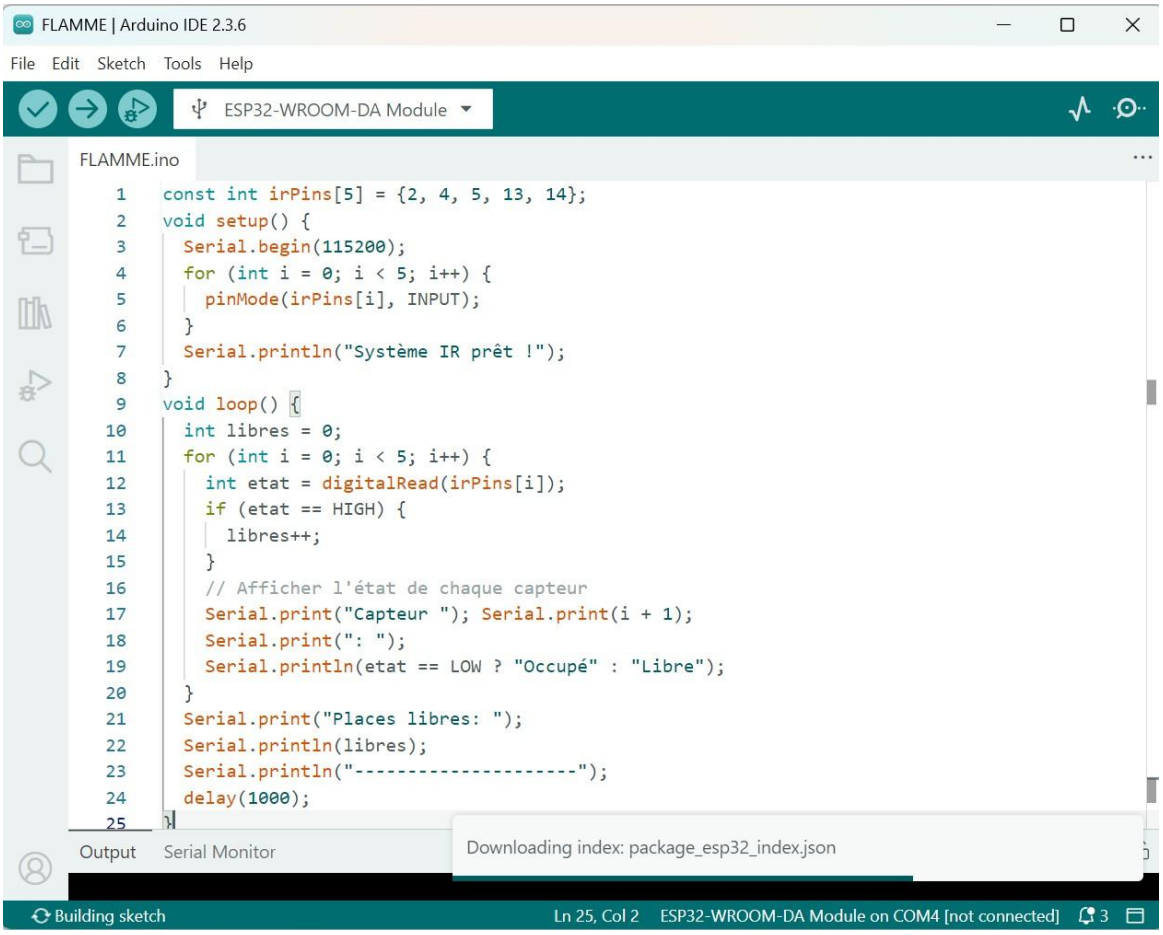


Figure III.28 : Exemple de programmation de 5 capteurs Infrarouge

Clavier matriciel 4x4 :

Le branchement d'un clavier matriciel 4x4 avec l'ESP32 :

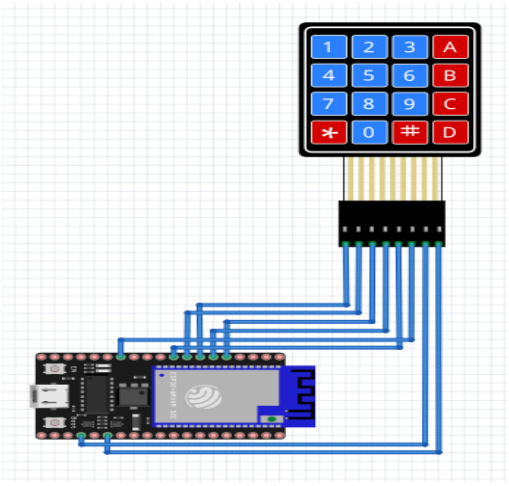


Figure III.29 : Branchement d'un clavier matriciel 4x4 à la carte ESP32

Les broches associées sont mentionnées dans le tableau suivant :

Broche du clavier	**Rôle**	**Branchement ESP32**
R1	Ligne 1	GPIO32
R2	Ligne 2	GPIO33
R3	Ligne 3	GPIO25
R4	Ligne 4	GPIO26
C1	Colonne 1	GPIO27
C2	Colonne 2	GPIO19
C3	Colonne 3	GPIO18
C4	Colonne 4	GPIO23

Tableau III.4 : Branchement d'un clavier matriciel 4x4 à la carte ESP32

Voici un exemple de programmation d'un clavier matriciel 4x4 :

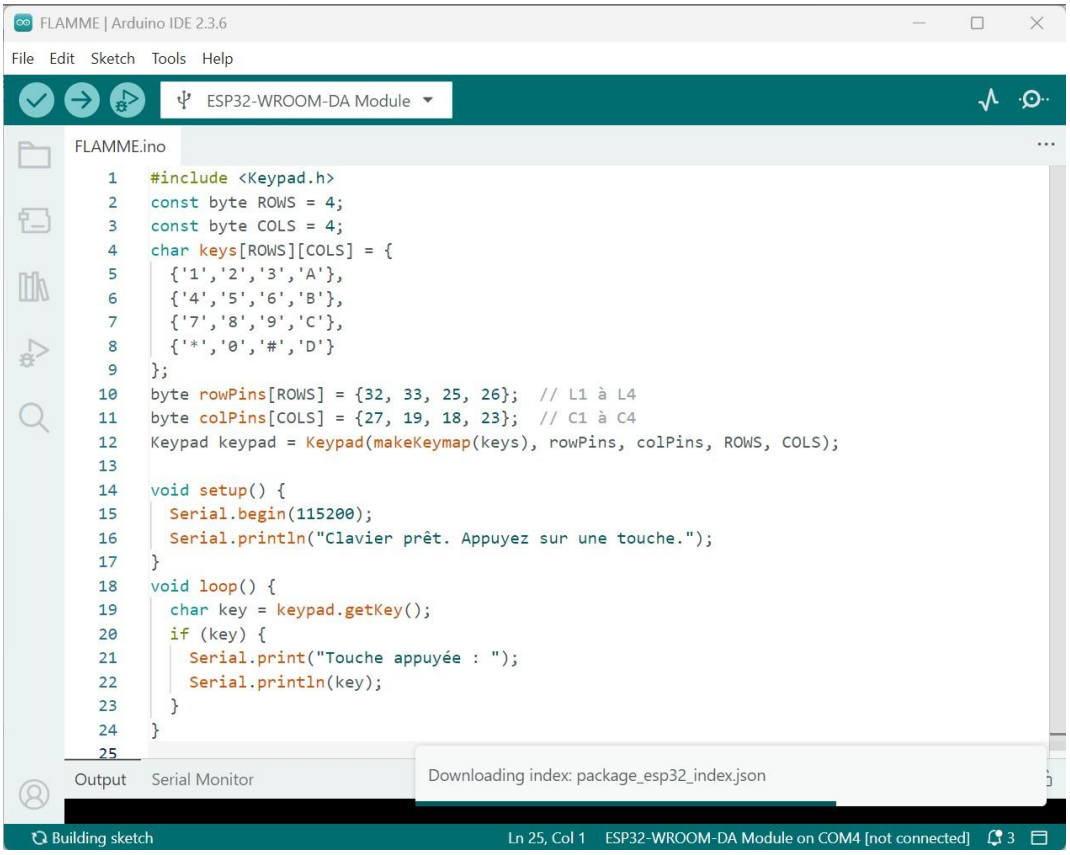


Figure III.30 : Exemple de programmation d'un clavier matriciel 4x4

Servomoteur :

Le branchement se fait comme suit :

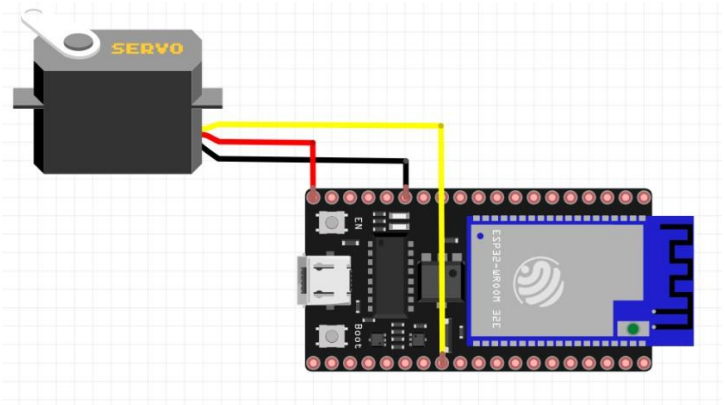


Figure III.31 : Branchement d'un Servomoteur à la carte ESP32

Nous avons utilisé deux servomoteurs dans notre système : l'un pour contrôler la barrière d'entrée, et l'autre pour la barrière de sortie du parking.

Voici le branchement des deux servomoteurs :

Servomoteur	**Signal (PWM)**	**Branchement ESP32**
Servo 1	Signal	GPIO16
Servo 2	Signal	GPIO17
VCC (Les 2 servomoteur)	Alimentation	5V *(partagé)*
GND (Les 2 servomoteur)	Masse	GND *(partagé)*

Tableau III.5 : Branchement des servomoteurs à la carte ESP32

Le programme suivant est un exemple de fonctionnement du deux Servomoteurs qui part de la position 180° à 90° puis revenir à l'état initial après 3 secondes :

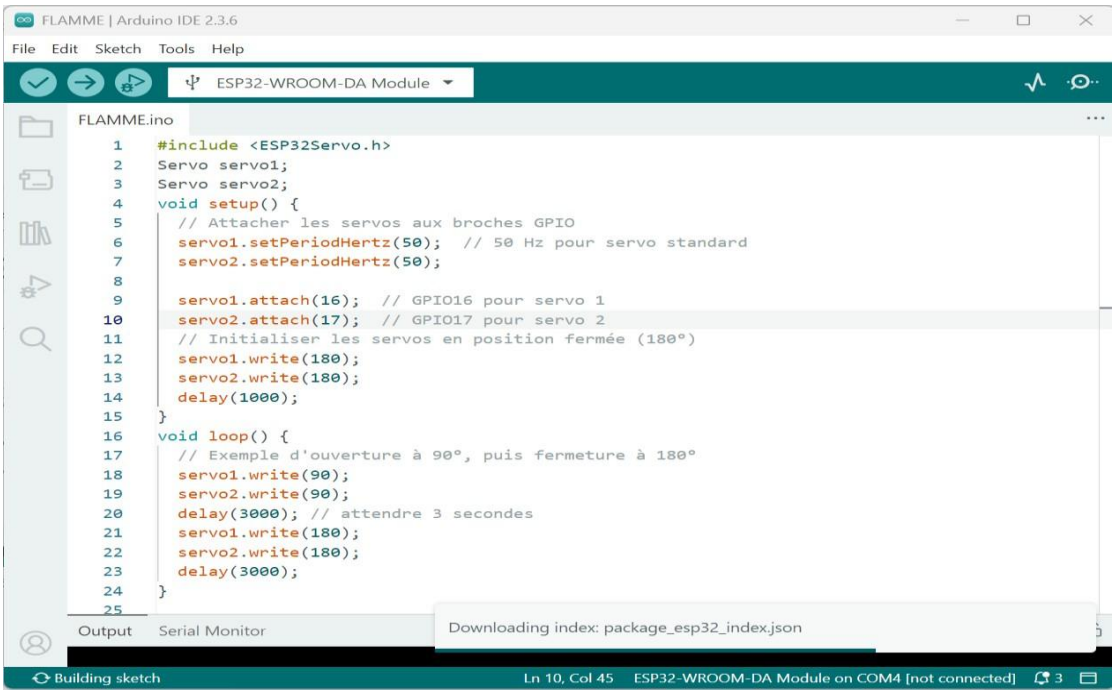


Figure III.32 : Exemple de programmation de deux servomoteurs

Buzzer :

Le branchement se fait comme suit :

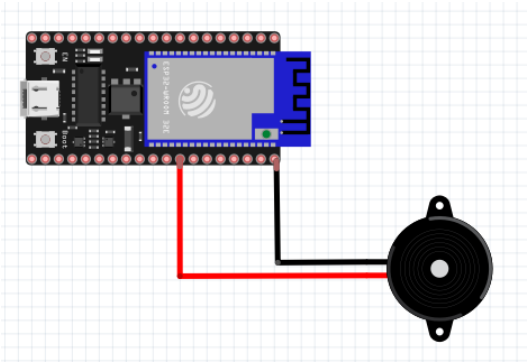


Figure III.33 : Branchement d'un buzzer à la carte ESP32

Le buzzer est utilisé pour se déclencher automatiquement en cas de détection d'incendie, afin d'émettre une alerte sonore immédiate et avertir les usagers du danger.

Broche du buzzer	Rôle	Branchement ESP32
+	Signal (ON/OFF)	GPIO12
–	Masse	GND

Tableau III.6 : Branchement d'un buzzer à la carte ESP32

Exemple de programmation d'un buzzer :

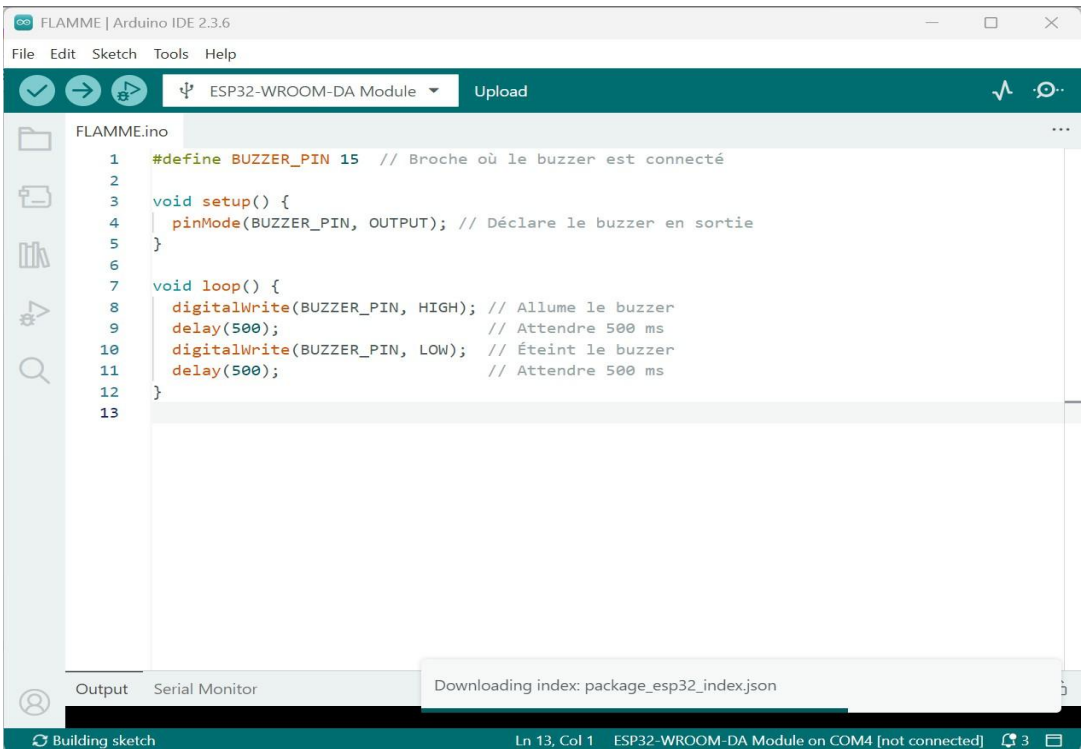


Figure III.34 : Exemple de programmation d'un buzzer

III.3. Assemblage globale :

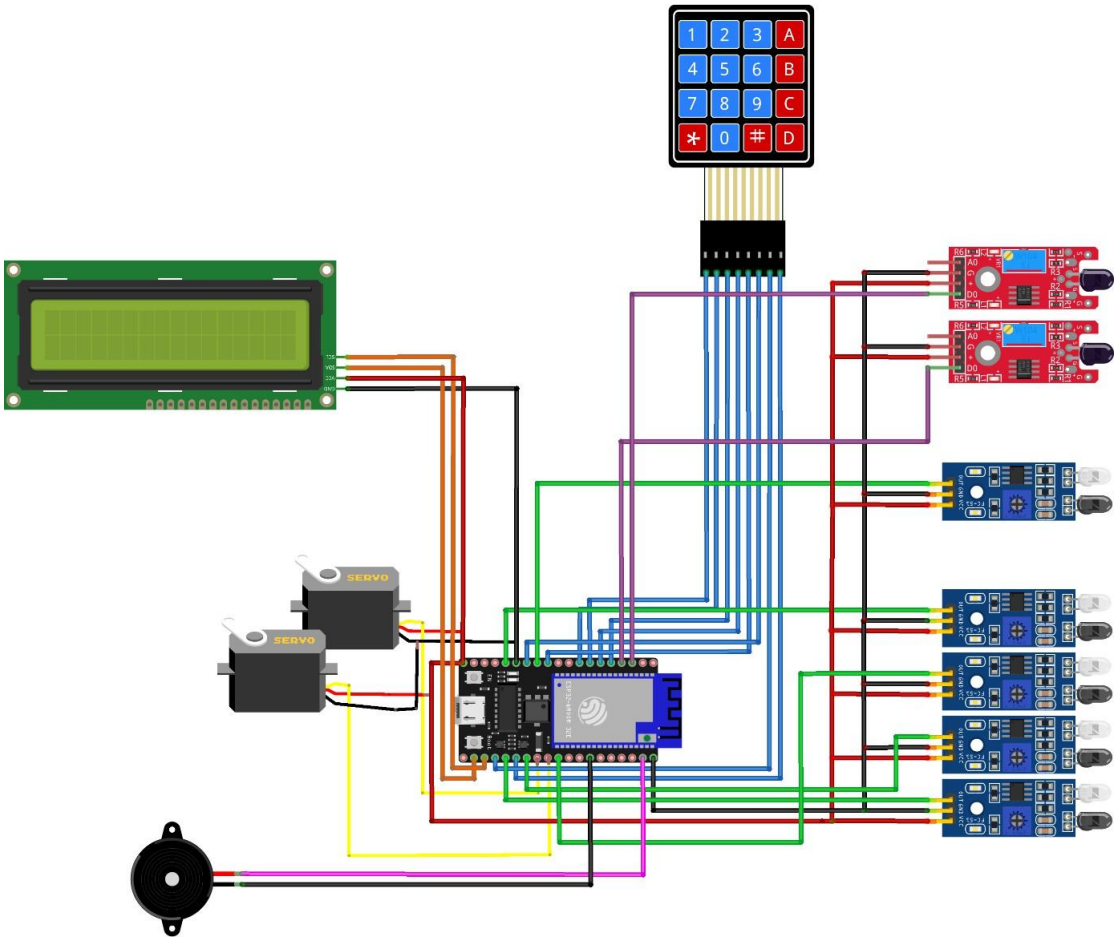


Figure III.35 : Exemple de programmation d'un buzzer

IV. Présentation de la maquette :

Dans cette partie, nous présentons une vue d'ensemble de la maquette de notre système de gestion intelligente de parking. Par la suite, des tests seront réalisés pour évaluer le bon fonctionnement du système.

IV.1. Présentation globale :

La maquette de notre parking intelligent se compose des éléments suivants :

- Une entrée et une sortie distinctes pour les véhicules, assurant une circulation fluide.
- Quatre places de stationnement, chacune équipée d'un capteur infrarouge permettant de détecter la présence ou l'absence d'un véhicule. Lorsqu'un véhicule est détecté par un

capteur ou qu'une place est réservée via l'application mobile, celle-ci est automatiquement décrémentée du nombre de places libres affiché.

- Deux capteurs de flamme sont installés pour détecter tout début d'incendie. En cas de détection, un buzzer se déclenche automatiquement pour alerter du danger, les deux barrières s'ouvrent simultanément pour permettre une évacuation rapide, et une notification d'alerte est envoyée à l'application mobile.

- Un afficheur indique en temps réel le nombre de places disponibles, mis à jour selon l'occupation détectée ou les réservations effectuées.

- Un clavier matriciel permet à l'utilisateur de saisir son code de réservation, lequel s'affiche sous forme de caractères masqués (*) sur un écran LCD, afin de garantir la confidentialité du code. Ce code n'est valide que pendant la période réservée, c'est-à-dire entre l'heure d'entrée et l'heure de sortie spécifiées lors de la réservation via l'application mobile. De plus, le code devient invalide après sa première utilisation, empêchant toute tentative d'accès ultérieure avec le même code. Si le code est valide, la barrière d'entrée s'ouvre automatiquement pour permettre l'accès au parking.

- À la sortie, un capteur infrarouge détecte la présence d'un véhicule en approche. Lorsque le véhicule est détecté, la barrière de sortie s'ouvre automatiquement.

Les deux figures ci-dessous illustrent la disposition des composants utilisés dans notre maquette:

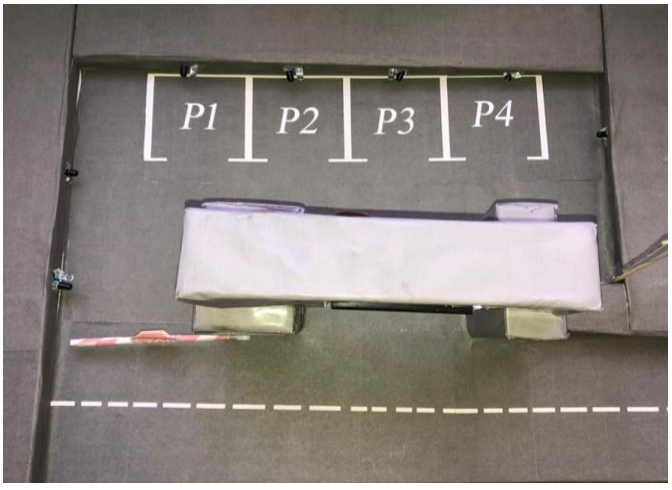


Figure III.36 : Vue de dessous de la maquette.

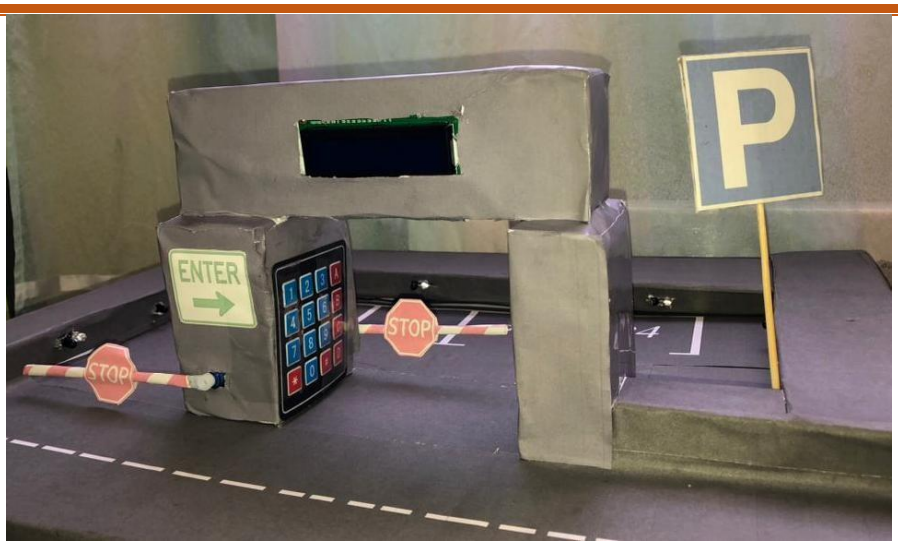


Figure III.37: Vue de face de la maquette.

IV.2. Présentation de programmation de maquette :

Afin de comprendre le fonctionnement global du système de gestion intelligente de parking, une série d'organigrammes a été réalisée. Ces schémas permettent de représenter de manière claire et structurée les différentes étapes logiques exécutées par le microcontrôleur ESP32.

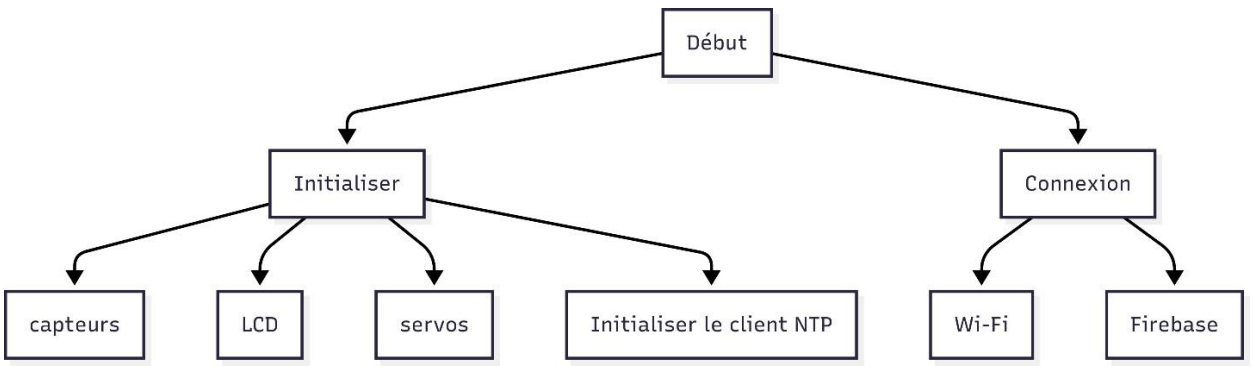


Figure III.38 : Étapes d'initialisation du système ESP32 au démarrage

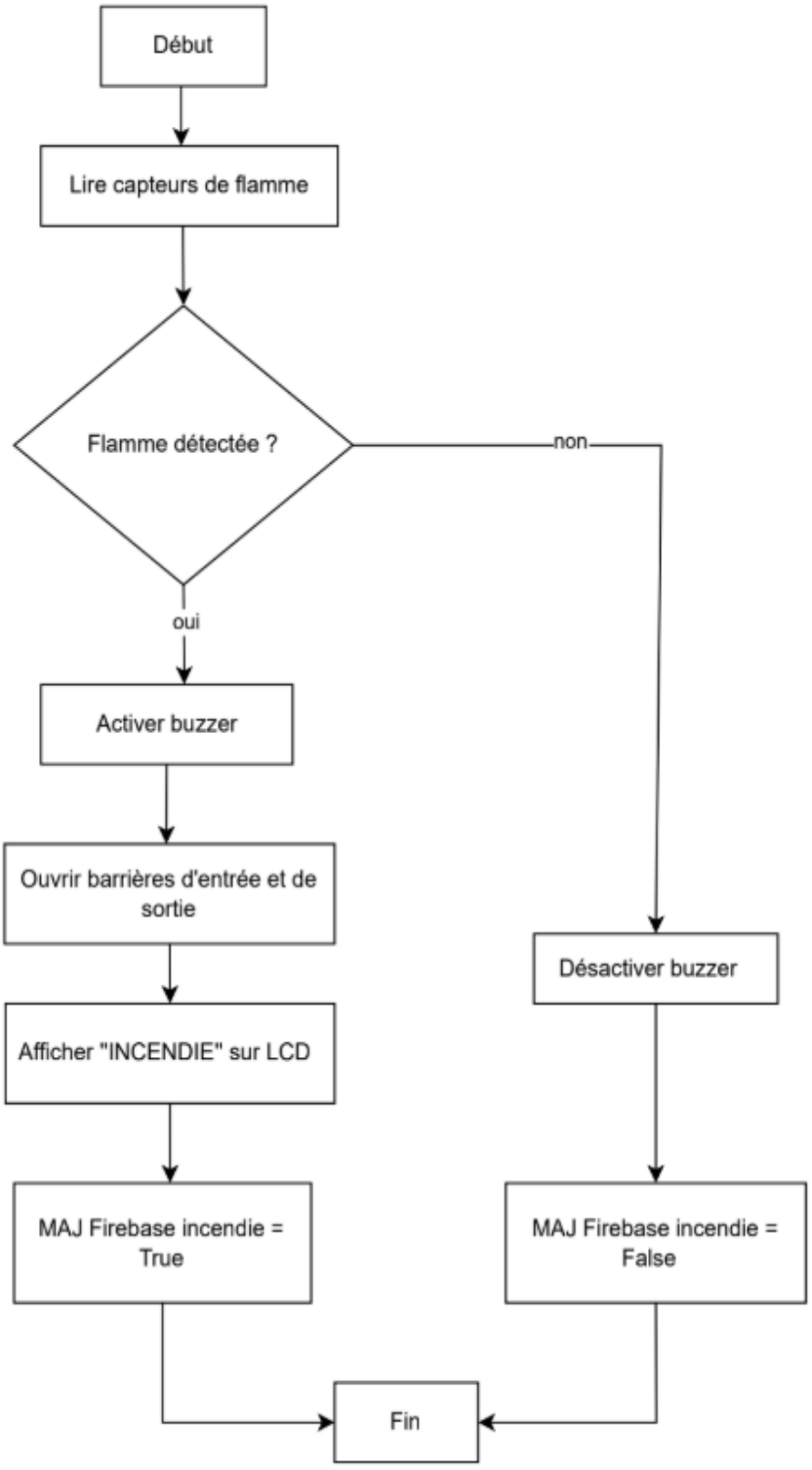


Figure III.39 : Organigramme de Détection et Gestion d'Incendie

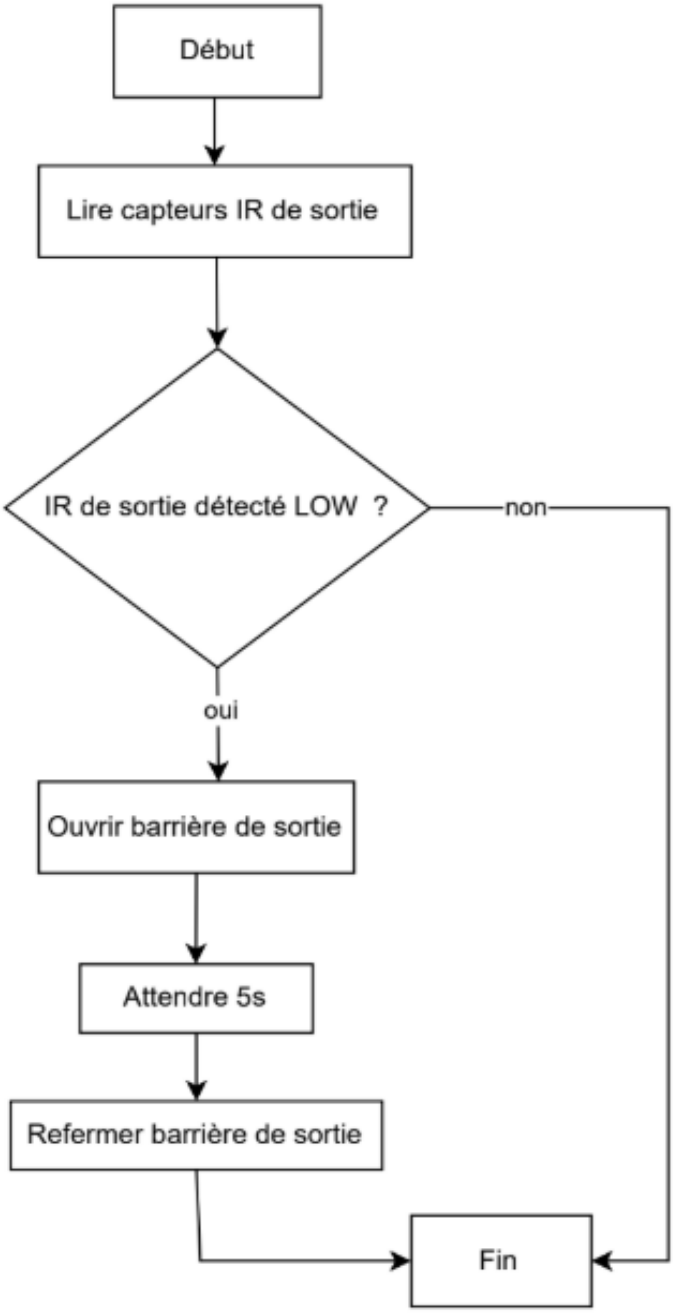


Figure III.40 : Organigramme d'Ouverture Automatique de la Barrière de Sortie

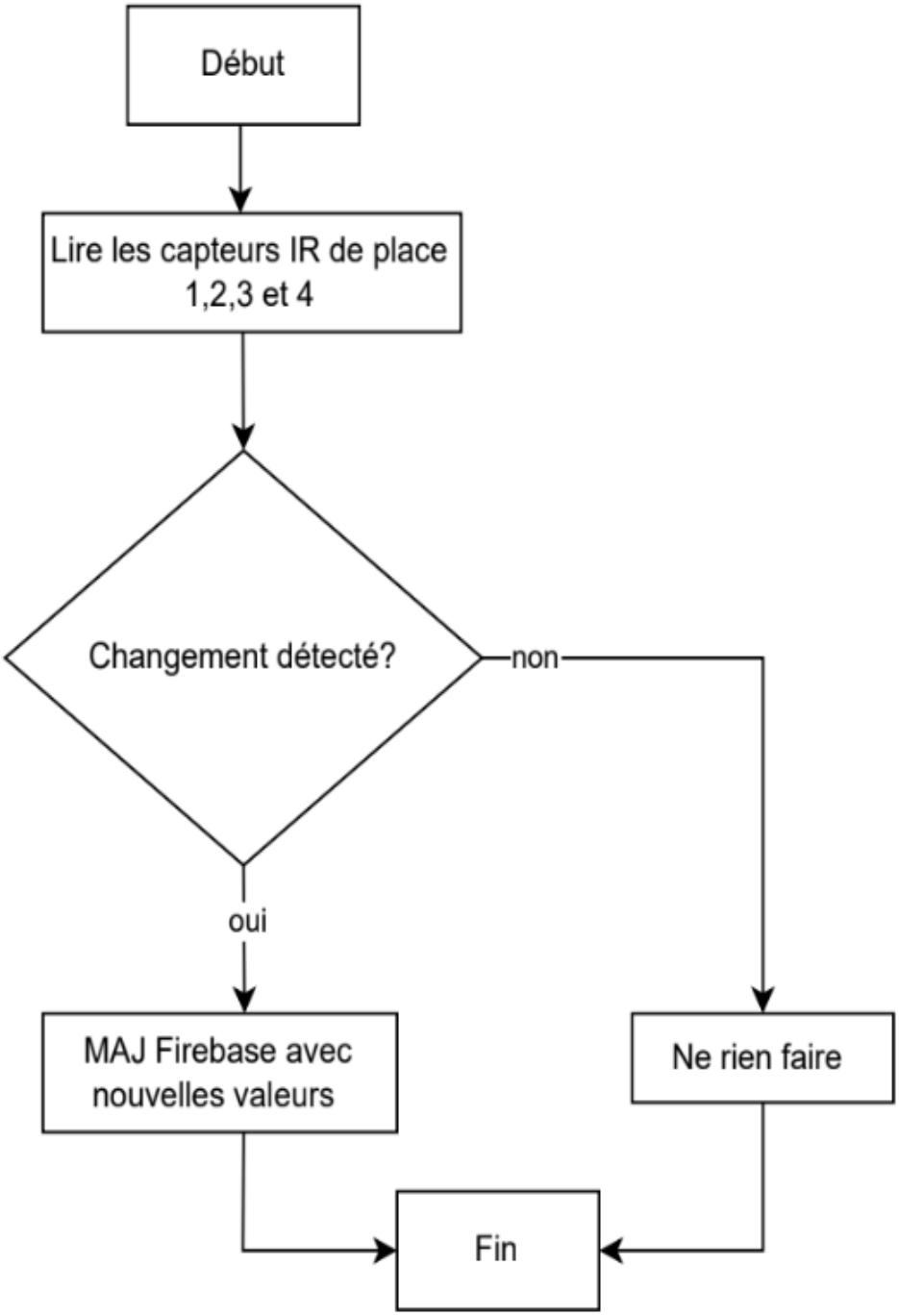


Figure III.41 : Organigramme de Mise à Jour des Capteurs Infrarouges

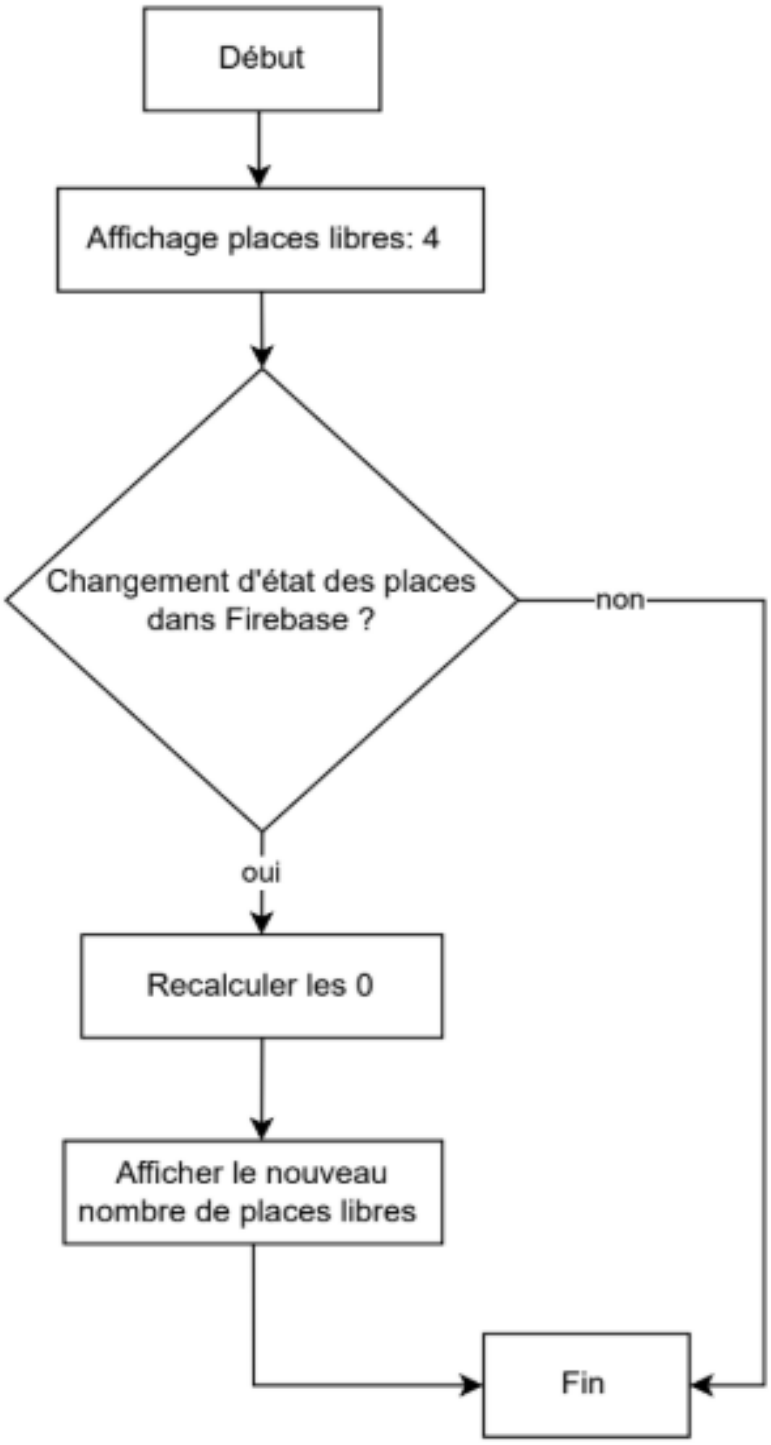


Figure III.42 : Organigramme du Calcul du Nombre de Places Libres

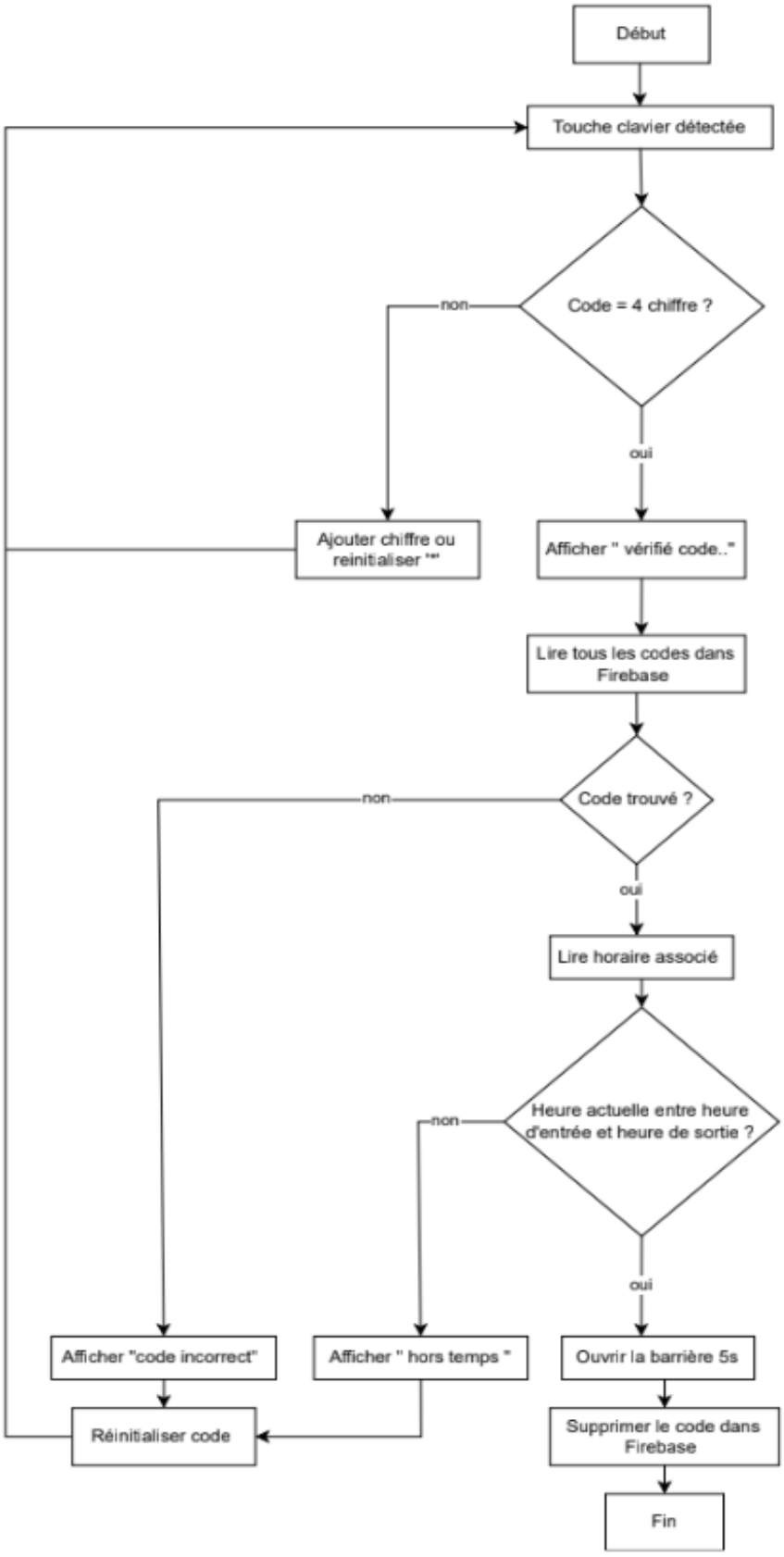


Figure III.43 : Organigramme de Saisie et Vérification du Code de Réservation

IV.3. Tests et résultats :

- **Test 1:** Cas du parking vide

Lorsque le parking est vide, aucun véhicule n'est présent et aucune réservation n'a été faite. L'afficheur montre que toutes les places sont disponibles. Les capteurs infrarouges ne détectent aucune présence. Le système est prêt à accueillir de nouveaux utilisateurs.

Figure III.44: Affichage sur la maquette et dans l'application — Parking vide.

- **Test 2:** Cas d'une réservation via l'application

Lorsqu'un utilisateur réserve une place à partir de l'application mobile, cette place est automatiquement bloquée. Le nombre de places libres affiché diminue. L'utilisateur reçoit un code de réservation qu'il utilisera à l'entrée du parking.

Figure III.45: Affichage sur la maquette et dans l'application — Après une réservation.

- **Test 3:** Cas d'un arrive d'un véhicule :

Lorsque le véhicule arrive, l'utilisateur saisit son code sur le clavier. Si le code est correct la barrière d'entrée s'ouvre et le code est immédiatement supprimé de la base Firebase. Sinon, un message indiquant que le code est invalide s'affiche sur l'écran et la barrière reste fermée. Une fois le véhicule garé, un capteur détecte sa présence ce qui permet au système de mettre à jour l'état de la place de parking dans l'application.

Figure III.46 : Arriver d'un véhicule

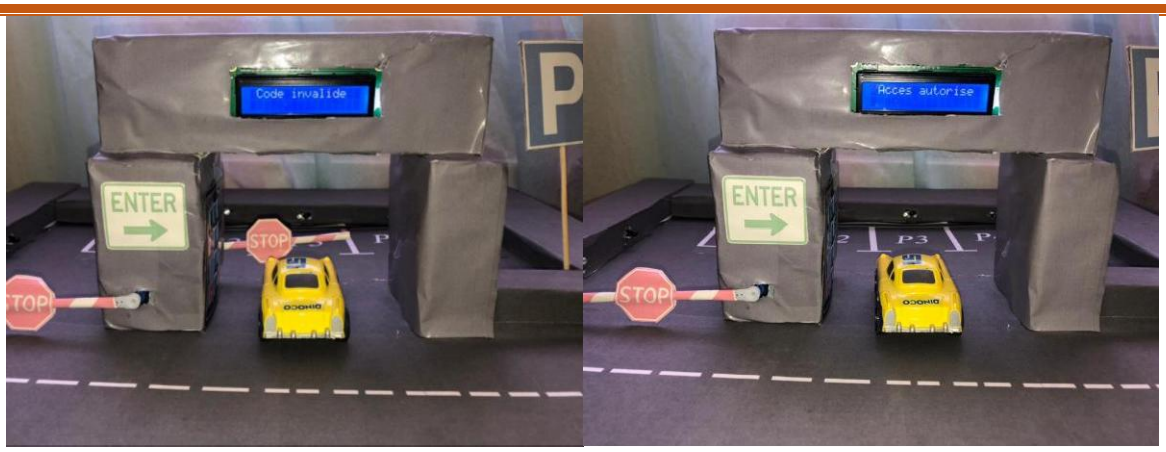


Figure III.47 : Vérification de code d'accès**.**

Figure III.48: Affichage dans l'application — Stationnement d'un véhicule.

- **Test 4:** Cas de deuxième réservation :

Lorsqu'un autre utilisateur réserve une place, l'état et la couleur de celle-ci changent dans l'application, et le nombre de places libres affiché sur l'écran passe à 2.

Figure III.49 : Affichage maquette et application — Deuxième réservation.

Figure III.50 : Vue de dessous — Deuxième stationnement.

- **Test 5:** Cas où le parking est plein

Lorsque toutes les places sont occupées ou réservées, le système affiche "0 place libre" sur l'écran, et toutes les places apparaissent en rouge dans l'application. Aucune nouvelle réservation ou entrée n'est autorisée. Une notification est envoyée aux utilisateurs pour les prévenir que le parking est plein.

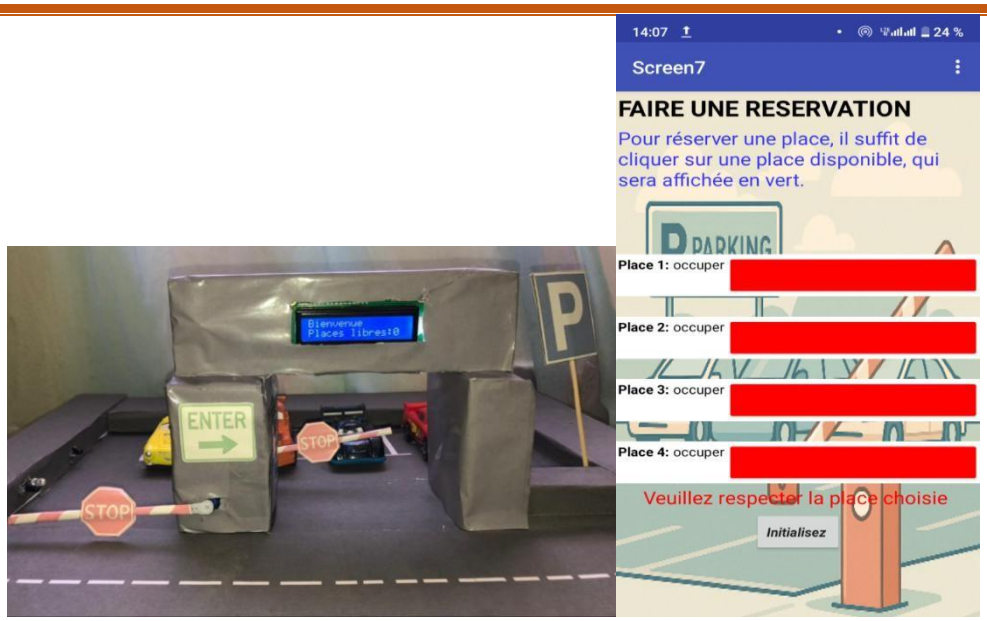


Figure III.51 : Affichage maquette et application — Parking plein.

Figure III.52 : Vue de dessous — Parking plein.

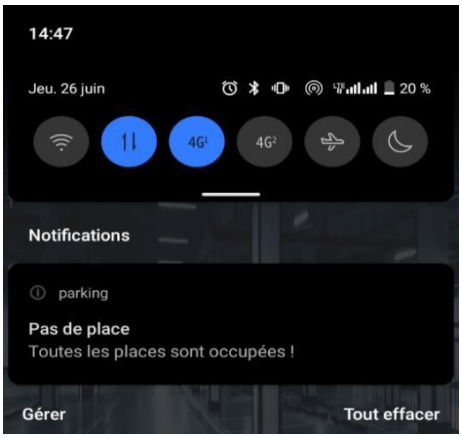


Figure III.53: Notification envoyée — Parking plein.

- **Test 6 :** Cas où une place se libère

Lorsqu'un véhicule quitte le parking, le capteur détecte l'absence. Le nombre de places libres augmente. Une notification est envoyée aux utilisateurs pour les informer qu'une place est disponible.

Figure III.54: Libération d'une place de stationnement.

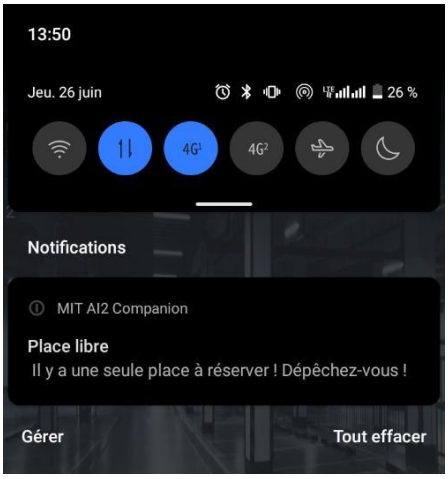


Figure III.55 : Notification envoyée — Place libre.

- **Test 7 :** Cas d'un incendie

Si un capteur de flamme détecte un départ de feu, le buzzer se déclenche. Les barrières d'entrée et de sortie s'ouvrent automatiquement pour permettre une évacuation rapide. Une alerte est envoyée à l'application pour informer les utilisateurs du danger.

Figure III.56: La maquette en cas d'incendie.

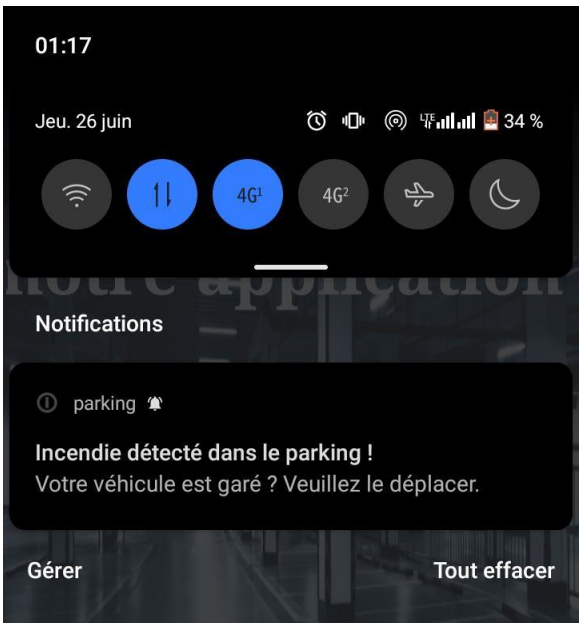


Figure III.57 : Notification envoyée — Incendie détecté dans le parking.

- **Test 8 :** Cas où l'utilisateur n'a pas de connexion :

Ce test simule une situation où l'utilisateur ne dispose pas de connexion Internet sur son téléphone au moment de la réservation. Dans ce cas, il ne peut ni consulter l'état des places disponibles, ni effectuer une réservation via l'application. Pour pallier ce problème, un administrateur est présent sur le site du parking. L'utilisateur peut alors se rendre directement auprès de l'administrateur, qui effectue la réservation à sa place via un accès dédié à la plateforme. Une fois la réservation validée, l'administrateur remet à l'utilisateur un code de réservation qu'il pourra saisir à l'entrée du parking. Ce test permet de vérifier que le système

reste fonctionnel même en l'absence de connexion du côté utilisateur, grâce à la présence humaine d'un gestionnaire sur place.

V. Conclusion :

Dans ce chapitre, nous avons montré comment notre système de gestion de parking intelligent a été réalisé. Nous avons expliqué la création de l'application mobile, le rôle de chaque composant utilisé dans la maquette, et comment ils fonctionnent ensemble grâce à la carte ESP32 et la base de données Firebase.

Les différents tests que nous avons faits prouvent que le système fonctionne correctement dans plusieurs situations : réservation, entrée, sortie, détection d'incendie, et mise à jour automatique du nombre de places disponibles.

Ce projet montre qu'il est possible d'améliorer la gestion des parkings en combinant technologie mobile et objets connectés, pour offrir un service plus pratique, plus rapide et plus sécurisé aux utilisateurs.

Conclusion générale

Notre travail portait sur la conception et la réalisation d'un système de gestion d'un parking contrôlé à distance via une application mobile. Ce système capable de détecter automatiquement l'occupation des places de stationnement et d'en informer l'utilisateur en temps réel via une application mobile, qui permet aussi de réserver une place ainsi que de simuler un paiement sans transfert d'argent. L'objectif de ce travail était de résoudre les problèmes de stationnement que nous avons mentionnés, et il a permis d'apporter une solution adaptée à ces difficultés.

Pour atteindre cet objectif, nous avons utilisé plusieurs composants technologiques : un microcontrôleur (ESP32) et plusieurs capteurs l'un pour détecter la disponibilité des places, un autre pour la présence de véhicules, et un autre pour la détection d'incendies ainsi que plusieurs actionneurs dont un pour contrôler la barrière, un autre pour afficher l'état du parking, et un dispositif de signalisation. Une application mobile développée via MIT App Inventor échange en temps réel avec le système via Firebase.

La réalisation de notre système de parking intelligent s'est déroulée en plusieurs étapes bien définies. Tout d'abord, nous avons procédé à la programmation de la carte ESP32 en utilisant l'environnement Arduino IDE. Ensuite, nous avons mis en place l'intégration avec la base de données Firebase, qui nous permet de stocker en temps réel des différentes informations. Firebase agit comme un pont entre le système embarqué (ESP32) et l'application mobile. Chaque changement est immédiatement synchronisé dans la base de données. Nous avons ensuite conçu l'application mobile via la plateforme MIT App Inventor. Cette application permet à l'utilisateur de consulter à distance les places disponibles dans le parking et de faire des réservations. Nous avons également ajouté une fonctionnalité de notification push, notamment en cas de détection d'incendie grâce au capteur de flamme, ou lorsqu'une seule place est libre ou bien toute les places sont occupées.

Concernant la maquette, nous avons réalisé l'installation des capteurs infrarouges au niveau de chaque place de stationnement afin de détecter la présence d'un véhicule. Le capteur de flamme a été positionné à un endroit stratégique du parking pour détecter rapidement tout début d'incendie. Les servomoteurs ont été configuré pour gérer automatiquement l'ouverture et la fermeture de la barrière, enfin l'écran LCD a été programmé pour afficher en temps réel l'état du parking.

Après la mise en œuvre complète du système, plusieurs tests ont été réalisés pour vérifier son bon fonctionnement, par exemple lors de réservations, en cas d'incendie dans le parking, lorsque toutes les places sont occupées, ou encore lorsqu'une place se libère. Les résultats obtenus sont globalement satisfaisants. Le système détecte correctement la présence ou l'absence d'un véhicule grâce aux capteurs infrarouges installés sur chaque place. Les données sont envoyées avec succès vers la base de données Firebase en temps réel, permettant à l'utilisateur de consulter en temps réel l'état du parking via l'application mobile. De plus, des notifications push sont envoyées dans des cas spécifiques.

Nous avons effectué plusieurs essais pour vérifier le bon fonctionnement de l'ensemble des composants. Le système est globalement fiable, bien que nous ayons observé une certaine lenteur lorsque la connexion Internet est faible, ce qui peut légèrement affecter la mise à jour des données. Malgré cette limitation, le système s'est montré fiable, efficace et réactif dans des conditions normales d'utilisation.

Même si le projet est fonctionnel, des améliorations restent possibles comme :

- Ajouter des caméras pour renforcer la sécurité dans le parking.[19]
- Intégrer un système de lecture automatique des plaques d'immatriculation pour identifier les véhicules plus facilement et de faciliter l'accès au parking.[20]

Les références bibliographiques :

[1] HAËNTJENS Jean. *Smart city, ville intelligente : quels modèles pour demain ?* Paris, La Documentation française, 2021.

[2] ELEKTOR Publishing. *MicroPython pour microcontrôleurs – Projets avec Thonny-IDE, uPyCraft-IDE et ESP32*, Éditions Elektor, 2022.

[3] WIKIPÉDIA. (21 mai 2024). *Parking*. Dans Wikipédia, l'encyclopédie libre. Consulté le 15 avril 2025, à l'adresse : https://fr.wikipedia.org/wiki/Parking

[4] FONDATION DES PARKINGS. *Les parkings sous la loupe*, Genève, Fondation des Parkings, 2024.

[5] ORNIKAR. (s.d.). *Les parkings automatisés*, Ornikar. Consulté le 24 avril 2025, à l'adresse:

https://www.ornikar.com/permis/conseils-conduite/stationnement/amenagements/parkings-automatises

[6] RAKOTOSALAMA Lova Tsilavo. (20 octobre 2023). *Gestion automatisée de parking* (Mémoire de licence professionnelle, École Supérieure Polytechnique d'Antananarivo, Université d'Antananarivo). Ministère de l'Enseignement Supérieur et de la Recherche Scientifique, République de Madagascar.

[7] RANDRIAMANALINA Jean Aimé. *Système de gestion d'un parking intelligent*, mémoire de licence, Mention Électronique, Université d'Antananarivo, École Supérieure Polytechnique d'Antananarivo, sous la direction de M. RAMASOMBOHITRA Nivonjy Nomen'Ahy, soutenu le 7 décembre 2023.

[8] SALEH Imène. (2018). *Internet des Objets (IdO) : Concepts, Enjeux, Défis et Perspectives*. Revue Internet des Objets, ISTE OpenScience.

[9] MANOUVRIER Julien. (s.d.). *Smart parking : tout savoir sur le parking intelligent*. Zenpark. Consulté le 20 mai 2025, à l'adresse :

https://zenpark.com/blog/entreprises/smart-parking-parking-intelligent/

[10] ROMAIN Olivier, & BOURDEL Emmanuel. *Conception des systèmes électroniques intelligents : architectures, communications et intégration embarquée*. Paris : ESIEE Éditions, 2023.

[11] ESPRESSIF SYSTEMS. (s.d.). *ESP32 Programmer's memory model*. Espressif Developer Zone. Consulté le 21 mai 2025, à l'adresse : https://developer.espressif.com/blog/esp32-programmers-memory-model

[12] ESPRESSIF SYSTEMS. (mai 2019). *ESP32-WROOM-32 Datasheet* (Version 3.9). Consulté le 21 mai 2025, à l'adresse :

https://www.espressif.com/sites/default/files/documentation/esp32-wroom-32_datasheet_en.pdf

[13] ASCH Georges, & POUSSERY Bernard. *Les capteurs en instrumentation industrielle*, 8e édition, Paris, Dunod, 2017.

[14] PARET Dominique. *Capteurs et systèmes de détection : Principes, technologies et applications*, Paris, Dunod, 2016.

[15] WINSEN SENSOR ELECTRONICS. (s.d.). *What is a flame sensor? A comprehensive guide to fire detection*. Consulté le 25 mai 2025, à l'adresse :

https://www.winsen-sensor.com/knowledge/what-is-a-flame-sensor.html

[16] ACADÉMIE DE BORDEAUX. (2019, mai). *IP 2 3 C6 D Capteur actionneur interface* [Document PDF]. Consulté le 25 mai 2025, à l'adresse :

https://ent2d.ac-bordeaux.fr/disciplines/sti-college/wp-content/uploads/sites/63/2019/05/IP-2-3-C6-MF-Capteur-actionneur-interface.pdf

[17] ADVANCED MOTION CONTROLS. (s.d.). *Servomoteurs* [Page web]. AMC. Consulté le 27 mai 2025, à l'adresse :

https://www.a-m-c.com/fr/servomoteur/

[18] RANDOM NERD TUTORIALS. (2025, 21 avril). *ESP32: Getting Started with Firebase (Realtime Database)*. Consulté le 27 mai 2025, à l'adresse :

https://randomnerdtutorials.com/esp32-firebase-realtime-database/

[19] MYCONNECT. (s.d.). Caméra de surveillance pour parking. MyConnect. Consulté le 7 juillet 2025, à l'adresse :

https://myconnect.fr/entreprise-videosurveillance/parking/

[20] DITTA Allah, AHMED Muhammad Maroof, MAZHAR Tehseen, SHAHZAD Tariq, ALAHMED Yazan, & HAMAM Habib. (2025). Number plate recognition smart parking management system using IoT. Measurement: Sensors, Elsevier. Consulté le 7 juillet 2025, à l'adresse: https://ouci.dntb.gov.ua/en/works/9jLq1rZ4/