جامعة سيدي محمد بن عبد الله بفاس
ⵜⴰⵙⴷⴰⵡⵉⵜ ⵙⵉⴷⵉ ⵎⵓⵃⵎⴷ ⴱⵏ ⵄⴱⴷⵍⵍⴰⵀ ⴱⴼⴰⵙ
UNIVERSITÉ SIDI MOHAMED BEN ABDELLAH DE FES

كلية العلوم والتقنيات فاس
ⵜⴰⵖⵢⴻⵓⵍⵜ ⵏ ⵜⵛⴰⵜⵜⵉⵡⵉⵏ ⴰ ⵜⵓⵏⵉⵖⵉⵏ - ⴼⴰⵙ
FACULTÉ DES SCIENCES ET TECHNIQUES DE FÈS

DEPARTEMENT DE GENIE ELECTRIQUE
FILIERE : LICENCE SCIENCES ET TECHNIQUE EN GENIE ELECTRIQUE (LST GE)

PROJET DE FIN D'ETUDES

PRÉPARÉ PAR

AMINE OUEDGHIRI SAIDI & ANASS ELRHENDOR

POUR L'OBTENTION DU
DIPLOME LICENCE SCIENCES ET TECHNIQUE EN GENIE ELECTRIQUE

INTITULÉ

MISE EN PLACE D'UN PARKING INTELLIGENT

ENCADRÉ PAR :
PR. HICHAM GHENNIOUI (FSTF)

SOUTENU LE 07 JUILLET 2021, DEVANT LE JURY COMPOSÉ DE :

PR. HICHAM GHENNIOUI	FACULTE DES SCIENCES ET TECHNIQUES DE FES
PR. AICHA ALAMI HASSANI	FACULTE DES SCIENCES ET TECHNIQUES DE FES

A.U. 2020-2021

RESUME

Le rôle que joue le parking dans le secteur socioéconomique et dans le fonctionnement des activités en villes est incontournable, un rôle dont l'importance a été soulignée par Bernardo Trujillo qui popularise dans les années 1950 la formule « no parking, no business ».

Or, dans les villes, une hantise du manque de stationnement, que partagent les acteurs municipaux et les décideurs immobiliers, qui vient du fait que la demande en stationnement, qui ne cesse de s'accroitre, dépasse de loin l'offre et ce, malgré les efforts des municipalités qui ont adopté le principe selon lequel chaque activité doit assumer les besoins de stationnement qu'elle crée, par le biais de normes minimales de stationnement pour chaque nouvelle construction.

Un tel besoin a suscité notre intérêt et nous a poussés à penser à mener notre recherche dans le domaine.

En effet, à partir de ce que nous remarquons au quotidien, à savoir le manque des lieux de stationnement et surtout, quand ils se trouvent, le conducteur du véhicule passe un temps gigantesque avant de pouvoir tomber sur une place libre pour se garer.

Nous avons donc jugé important de mener notre recherche sur un tel sujet dont l'objectif sera d'essayer de créer un parking intelligent qui pourrait alléger le poids d'un tel problème pour les conducteurs.

Nous avons adopté la ***méthode constructive*** étant donné que nous sommes partis du terrain et de notre constat pour pouvoir construire le dispositif à proposer pour la résolution du problème constaté. Nous avons opté pour l'utilisation des capteurs infrarouges comme outil de travail.

Le résultat de notre recherche était la construction d'une machine susceptible d'aider à résoudre le problème de stationnement : **le parking intelligent.**

Abstract

The role that parking plays in the socio-economic sector and in the operation of activities in cities is essential, a role whose importance was emphasized by Bernardo Trujillo who popularized in the 1950s the formula "no parking, no business". However, in cities, a haunting of the lack of parking, shared by municipal actors and real estate decision-makers, which comes from the fact that the demand for parking, which continues to grow, far exceeds the supply and this, despite the efforts of municipalities that have adopted the principle according to which each activity must assume the parking needs it creates , through minimum parking standards for each new construction. Such a need aroused our interest and prompted us to think about conducting our research in the field. Indeed, from what we notice on a daily basis, namely the lack of parking spaces and especially, when they are, the driver of the vehicle spends a gigantic time before being able to fall on a free place to park. We therefore considered it important to conduct our research on such a subject whose objective will be to try to create a smart parking lot that could lighten the burden of such a problem for drivers. We have adopted the constructive method since we started from the field and our observation to be able to build the mechanism to be proposed for the resolution of the problem noted. We opted for the use of infrared sensors as a working tool. The result of our research was the construction of a machine that could help solve the parking problem: smart parking.

DEDICACES

Amine OUEDGHIRI SAIDI

Je dédie ce travail, en premier lieu, à ma chère mère qui était toujours làpour moi, à me présenter un soutien inconditionnel et à m'encourager à aller de l'avant et qui a toujours cru en mes capacités de réussite. Vraiment, ses encouragements m'ont été d'une grande aidedans les moments les plus difficiles. Qu'elle trouve ici tous les sentiments de dévouement infini.

A mon père qui a supporté mes caprices et à mon frère Younes à qui je souhaite une

vie pleine de succès et de bonheur.

ANASS ELRHENDOR

Je dédie ce projet :

à ma chère mère, à mon cher père, qui n'ont jamais cessé de formuler des prière à mon égard, de me soutenir et de m'épauler pour que je puisse atteindre mes objectifs

à mes frères, pour leurs soutiens moral et leurs conseils précieux tout au long de mes études

à mon binôme, pour être patient et plein d'énergie

à mes amis, pour leurs aides et supports dans les moments difficiles

à toute ma famille

REMERCIEMENTS

Nous voudrions dans un premier temps témoigner notre reconnaissance à notre Encadrant, Monsieur GHENNIOUI HICHAM, Professeur à la Faculté des Sciences et Techniques de Fès, et le remercier pour sa patience et sa disponibilité. Nous le remercions vivement de nous avoir encadrés, orientés par ses judicieux conseils qui ont contribué à alimenter notre réflexion tout au long de ce projet.

Membre jury

Enseignants du département de génie électrique

Nos remerciements vont également à PR. BYO pour nous avoir aider avec le matériel utilisé.

Table des matières

LISTE DES FIGURES

LISTE DES TABLEAUX

LISTE DES ABREVIATIONS/ACRONYMES

APP	Application
FST	Faculté des Sciences et Techniques
ID	Identifient
IDE	Integrated Developement Environment
IOT	Internet Of Things
IR	Infra Rouge
LSSC	Laboratoire Signaux Systèmes et Composants

CHAPITRE 1 – CONTEXTE GENERAL DU PROJET

Dans ce chapitre nous allons présenter l'environnement du stage ainsi que la problématique et l'objectif qui nous ont motivée a réaliser notre projet, Parking intelligent avant de passer a la planification du stage on utilisant le diagramme de Gantt.

1.1 PRESENTATION DU LABORATOIRE SIGNAUX, SYSTEMES ET COMPOSANTS

Le Laboratoire Signaux Systèmes & Composants (LSSC) est un laboratoire de l'Université Sidi Mohammed Ben Abdallah (USMBA), sis à la Faculté des Sciences et Techniques de Fès (FSTF). La mission du LSSC étant de CONTRIBUER ET RENFORCER LA RECHERCHE, LE DEVELOPPEMENT TECHNOLOGIQUE ET L'INNOVATION au sein l'université USMBA.
Le LSSC est monté autour de thématiques fondatrices depuis une trentaine d'années qui sont les composants électroniques, les signaux et les systèmes de télécommunications et l'ingénierie des matériaux. Ainsi les activités de recherche menées au sein du laboratoire sont pluridisciplinaires mais complémentaires, elles vont de la synthèse de matériaux et la caractérisation de leurs propriétés, leur mise en forme et leur positionnement dans des dispositifs tels les capteurs, jusqu'à la conception et l'implémentation de systèmes intelligents et autonomes, dotés de capacité de traitement à base d'intelligence artificielle pour remplir et optimiser différentes fonctions pour différentes applications (Transmission de données, Conduite, Surveillance, Sécurité, diagnostic-health monitoring, gestion de l'énergie….) **[1].**

1.2 CONTEXTE DU PROJET

1.2.1 PROBLEMATIQUE

Dans les villes, une hantise du manque de stationnement, que partagent les acteurs municipaux et les décideurs immobiliers, qui vient du fait que la demande en stationnement, qui ne cesse de s'accroitre, dépasse de loin l'offre et ce, malgré les efforts des municipalités. En effet, « Les municipalités ont alors adopté le principe selon lequel chaque activité doit assumer les besoins de stationnement qu'elle crée, par le biais de normes minimales de stationnement pour chaque nouvelle construction » Le rôle que joue le parking dans le secteur socioéconomique et dans le fonctionnement des activités en villes est incontournable, un rôle dont l'importance a été soulignée par Bernardo Trujillo qui popularise dans les années 1950 la formule « no parking, no business ».

Un tel besoin a suscité notre intérêt et nous a poussés à penser à mener notre recherche dans le domaine.

En effet, à partir de ce que nous remarquons au quotidien, à savoir le manque des lieux de stationnement et surtout, quand ils se trouvent, le conducteur du véhicule passe un temps gigantesque avant de pouvoir tomber sur une place libre pour se garer.

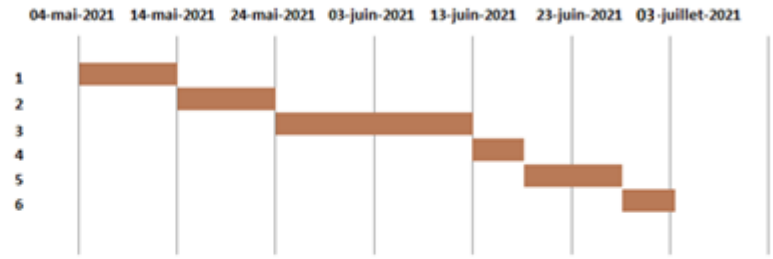
Nous avons donc juger important de mener notre recherche sur un tel sujet dont l'objectif sera d'essayer de créer un parking intelligent qui pourrait alléger le poids d'un tel problème pour les conducteurs.

1.2.2 OBJECTIFS DU PROJET

L'objectif du projet est de faciliter et économiser le temps de stationnement des utilisateurs de voitures à l'aide d'un parking intelligent réalisé avec Raspberry Pi, des capteurs infrarouge, et une application android.

1.2.3 PLANIFICATION DU PROJET

Tache	Debut	Durée	Fin
1	04-mai-2021	10	14-mai-2021
2	14-mai-2021	10	24-mai-2021
3	24-mai-2021	20	13-juin-2021
4	13-juin-2021	5	18-juin-2021
5	18-juin-2021	10	28-juin-2021
6	28-juin-2021	6	04-juillet-2021



Tache 1 : Formation raspberry et python
Tache 2 : Définition de l'architecture du parking et lister le matériel nécessaire
Tache 3 : Mise en place de la solution
Tache 4 : Établissement du plan de validation de la solution
Tache 5 : Rédaction du mémoire
Tache 6 : Préparation soutenance

Figure 1 Diagramme de Gantt

Conclusion du chapitre

Comme nous avons vu ce projet nécessite une formation dans différents langages de programmation.

Nous allons étudier différentes solutions pour réaliser un parking intelligent dans le chapitre suivant

Chapitre 2 – Etude des solutions existantes

Dans cette partie, nous avons traités quelques solutions pour réaliser un parking intelligent, nous allons comprendre le fonctionnement, les matériels utilisés, les avantages et les inconvénients de chaque solution pour étudier son efficacité.

2.1 PARKING INTELLIGENT EN CE BASANT SUR CAMERA ET RASPBERRY PI

2.1.1 MATERIEL UTILISE

Figure 2 Matériels utilises du parking avec camera

2.1.2 LE FONCTIONNEMENT DU PARKING INTELLIGENT BASE SUR CAMERA

Dans cette solution, la difficulté de trouvé une place dans le parking est résolue en ce basant sur trois étapes:

-Le traitement d'image

-Le Cloud développement

- Développement d'application mobile

La caméra est utilisé pour trouver une place dans le parking au lieu du capteur, elle est placé au centre du parking pour détecter tous les mouvements que ce trouve dans cette place, le Rasperry pi dans cette cas fait les traitements d'image en utilisant le langage Python et Open cv

Or en utilisant une technique qui s'appelle « edge détection » qui détecte les changements sur les pixels. Une fois une voiture est détectée par la camera, les données sont envoyé depuis Raspberry a un serveur, et du serveur a une application mobile que l'utilisateur installe.

Dans la figure suivante Nous allons voir le principe de fonctionnement d'un parking intelligent en se basant sur caméra et Raspberry pi

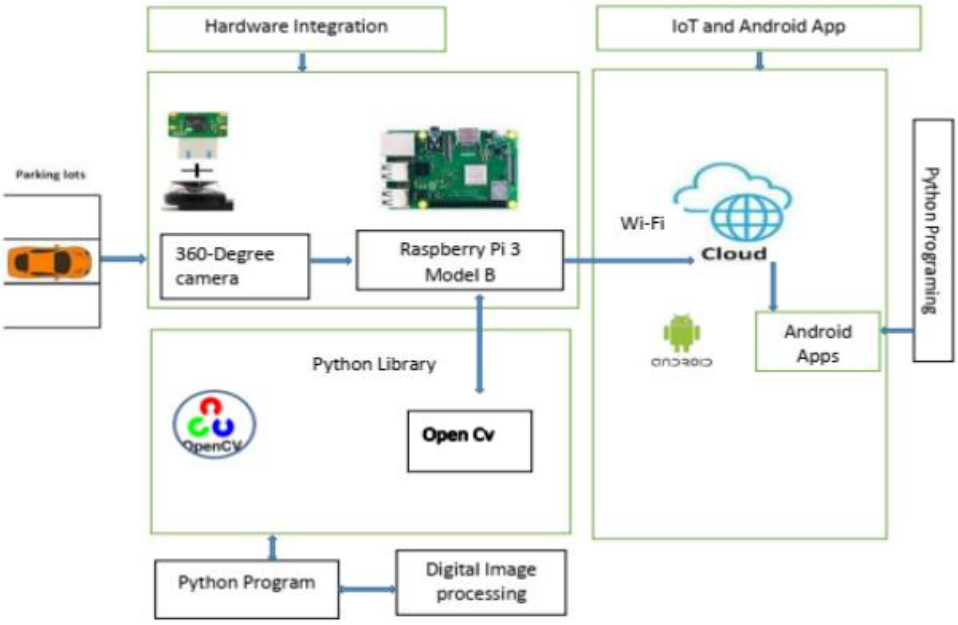


Figure 3 Principe de fonctionnement du parking avec camera [2]

2.1.3 LIMITES DE CETTE SOLUTION

Cette solution est pratique dans des conditions et des lieux plus spécifiques car cette solution est basée sur une caméra donc l'image sera claire dans les jours ensoleillés ou mois humides, par contre ses inconvénients se montre dans les jours plus humides et pluvieux, car la camera aura des difficultés à pouvoir capter des images claires.

2.2 PARKING INTELLIGENT EN CE BASANT SUR DETECTEURS ULTRASON

2.2.1 MATERIEL UTILISE

Dans la figure suivante nous allons exposer les matériels utilisés pour la construction d'un parking intelligent basé sur capteurs ultrason et un microprocesseur

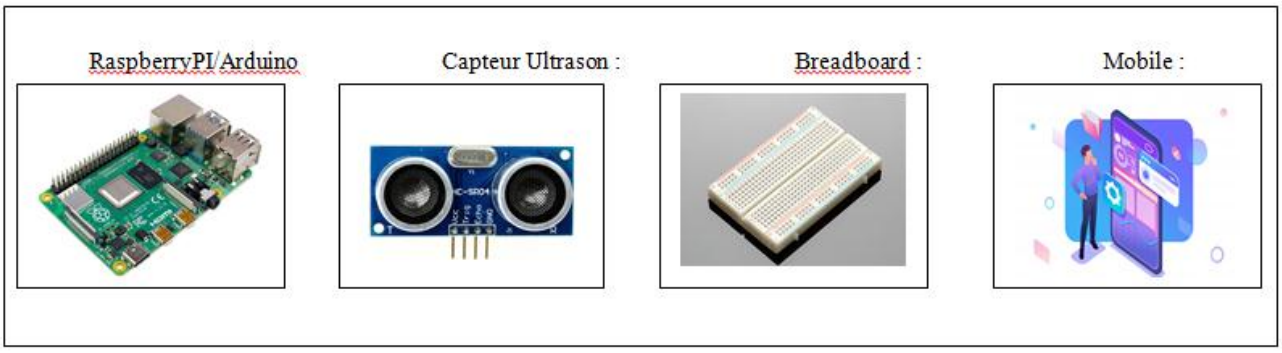


Figure 4 Matériels utilisés du parking avec ultrason

2.2.2 LE FONCTIONNEMENT DU PARKING BASE SUR CAPTEUR ULTRASON

Un capteur ultrason émet des courtes impulsions sonores avec haute fréquence, quand ces impulsions rencontrent un objet, elles se réfléchissent et reviennent au capteur.

Les données récoltées par les capteurs ultrason sont envoyées a Raspberry pi puis a un serveur pour les stocker et enfin ils sont envoyées a une application mobile que les conducteurs Install

La figure suivante montre le principe du fonctionnement d'un parking intelligent basé sur capteurs ultrason.

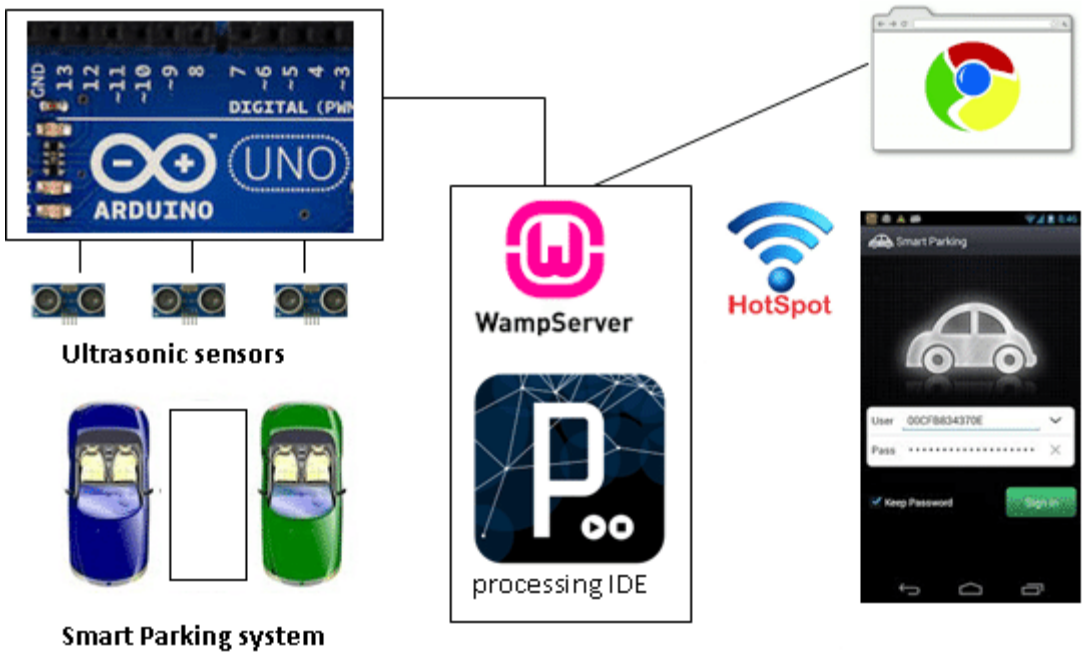


Figure 5 Fonctionnement du Parking avec capteurs ultrason [3]

2.2.3 LIMITES DE CETTE SOLUTION

Cette solution a plusieurs inconvénients :

a) Les capteurs ultrason consument beaucoup d'énergie électrique.

b) Les véhicules plus larges, et une haute consommation de courant rendent la conception d'un petit capteur difficile, donc il va falloir un gros émetteur et récepteur ultrason ainsi qu'une batterie plus grande.

c) ils ne sont pas efficaces sous une plaque de recouvrement, l'émetteur nécessite d'avoir une vue sur l'objet. Cela fonctionne très bien pour les capteurs suspendus dans les garages, mais dans le cas d'un capteur de stationnement monté au sol, il existe un risque d'endommagement mécanique de l'élément de détection à ultrasons.

2.3 PARKING INTELLIGENT EN CE BASANT SUR MAGNETOMETRE

2.3.1 MATERIEL UTILISE

Dans la figure suivante nous allons exposer les matériels utilisées dans la construction d'un parking intelligent basé sur magnétomètre et Raspberry pi.

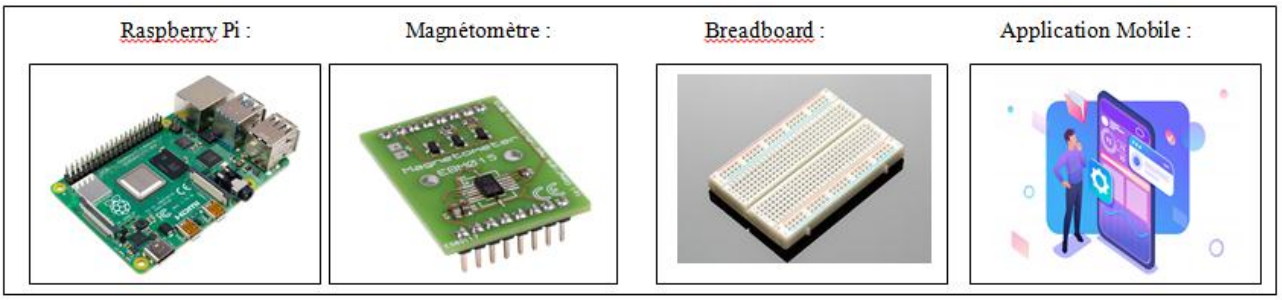


Figure 6 Matériels utilisés du parking avec magnétomètre

2.3.2 LE FONCTIONNEMENT DU PARKING BASE SUR MAGNETOMETRE

Le magnétomètre est un capteur qui mesure l'intensité du champ magnétique. Les objets contenant du fer, comme les voitures, créent une distorsion à courte portée qui peut être mesurée. L'amplitude de cette distorsion dépend du type d'alliage ferreux, de la distance au capteur et de la taille de l'objet.

Les données récoltées par le magnétomètre sont envoyées a Raspberry pi puis à un serveur pour les stocker et enfin ils sont envoyées à une application mobile que les conducteurs Install

La figure suivante montre le principe du fonctionnement d'un parking intelligent basé sur capteurs ultrason.

Nous allons démontrer la distorsion du champ magnétique produit par une voiture qui contient des objets ferromagnétique.

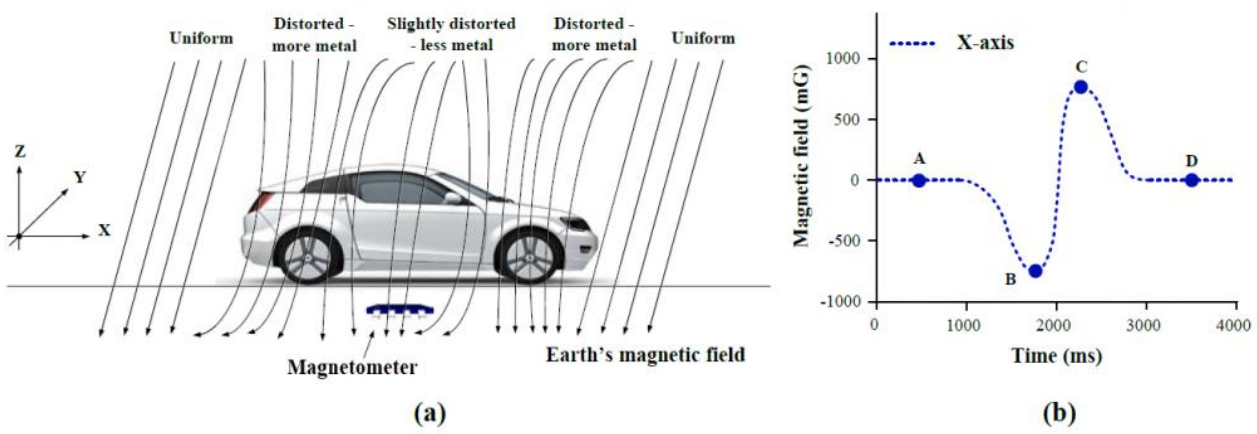


Figure 7 Distorsions créées par une voiture qui contient des objets ferromagnétiques [4]

2.3.3 LIMITES DE CETTE SOLUTION

Compte tenu de la variété des capteurs capables de détecter le champ magnétique, le processus semble au départ être une tâche simple : faire quelques expériences avec la détection de seuils, calibrer les algorithmes et le capteur de stationnement intelligent est prêt. Cependant, les choses ne sont pas si simples. Différents véhicules ont différents niveaux d'intensité de champ magnétique, et même si deux voitures peuvent se ressembler, l'une d'entre elles peut avoir une empreinte de champ magnétique beaucoup plus forte. Dans le même temps, chaque voiture a des endroits où la distorsion est minime (voir l'illustration ci-dessus). Si nous prenons des capteurs de stationnement intelligents fonctionnant sur batterie, des algorithmes de détection complexes plus avancés entraîneront une consommation d'énergie plus élevée, des taux d'échantillonnage des capteurs plus élevés et plus de temps de calcul sur le microprocesseur. Ainsi, l'un des principaux inconvénients

des capteurs de stationnement intelligents basés sur un magnétomètre est la diminution rapide de la durée de vie de la batterie avec les exigences de précision croissantes. En plus de cela, les véhicules électriques modernes n'ont souvent pas du tout de pièces ferromagnétiques. Nwave a un client de point de recharge EV qui a testé indépendamment quelques modèles de capteurs à magnétomètre avec tous les types de voitures électriques et les résultats ont été décevants – 60 % de précision de détection de véhicule au lieu de 95 % – 98 % annoncés.**[4]**

CONCULSION

Ce chapitre nous montre qu'il existe plusieurs méthodes qui permettent de realiser un parking intelligent et chaque méthode a ses avantages et ses limites.

Parmi ces méthodes il existe une que nous avons réalisée avec des capteurs IR.

Dans le chapitre suivant nous allons étudier cette solution et les étapes que nous avons suivies pour la réaliser.

CHAPITRE 3 – ETUDE DE NOTRE SOLUTION BASE SUR LA MISE EN PLACE DU PARKING INTELLIGENT AVEC RASPBERRY PI ET CAPTEUR INFRAROUGE

Le problème de parking en général est le temps perdu pour trouver une place, nous avons résolu ce problème en se basant sur IOT 'internet of things', pour communiquer la disponibilité des spots aux utilisateurs de voiture avec une application mobile,Dans ce chapitre nous allons expliquer la Méthodologie puis les matériels ,les langages et les sites web utilisés avant de conclure avec la programmation du Raspberry PI et la construction de l'application mobile.

3.1 METHODOLOGIE:

Nous allons démontrer le principe de fonctionnement de notre solution dans la figue suivante.

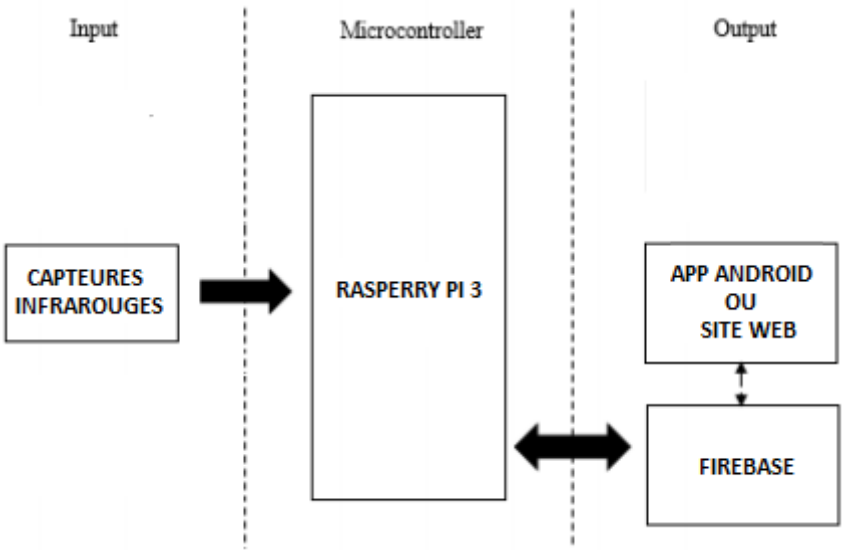


Figure 8 Principe de fonctionnement du parking avec capteurs infrarouge

3.2 MATERIEL ET LANGUAGE UTILISE

3.2.1 MATERIEL UTILISE :

Nous allons exposer les matériels utilisés dans la figure suivante :

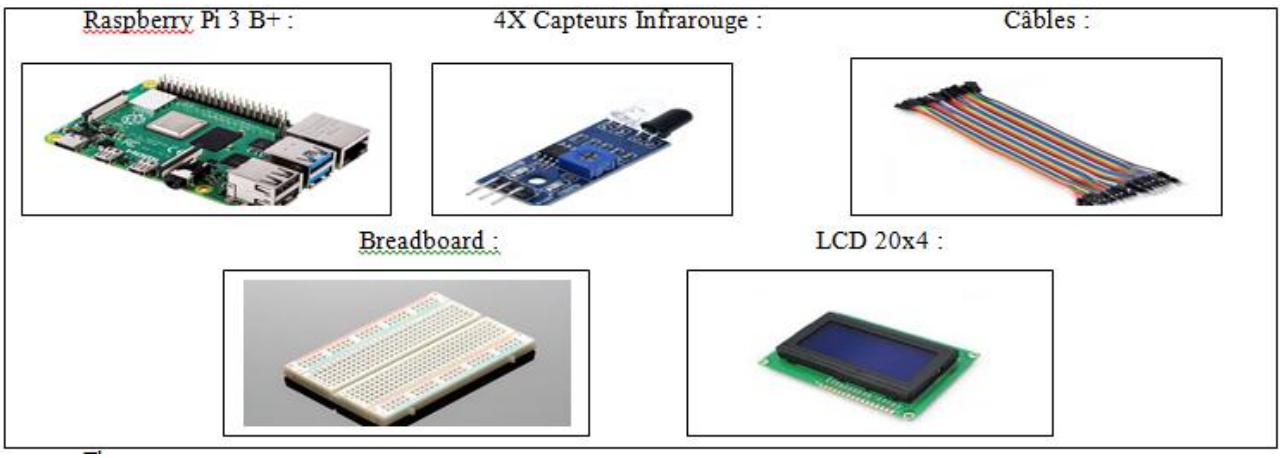


Figure 9 Matériels utilisés dans la solution proposée

3.2.2 LES LANGAGES, SITES WEB ET LOGICIEL UTILISES

Nous allons lister les langages de programmation, sites web et logiciel utilisés ainsi que leurs fonctionnalités.

Langages/site web/logiciels	Leurs utilités
Python	Programmer le Raspberry Pi pour lire les données envoyées par les capteurs IR, et pour envoyer les données collectées a Firebase et afficher un message sur LCD 20x4.
HTML & CSS	Désigne de l'application mobile
Java script	Lier et recevoir les données de Firebase puis les envoyer et les afficher dans l'application mobile
FireBase	Stocker les données envoyées par Raspberry Pi et les envoyer vers notre application mobile.
Android Studio	Aider a designer l'application mobile et la construire

Tableau 1 Fonctionnalités des langages, sites web et logiciels utilisés

3.3 REALISATION DU PROJET ET PROTOTYPE

3.3.1 CABLAGE DU RASPBERRY PI AVEC CAPTEURS IR ET LCD 20X4

Nous allons démontrer le câblage de Raspberry pi dans la figure suivante.

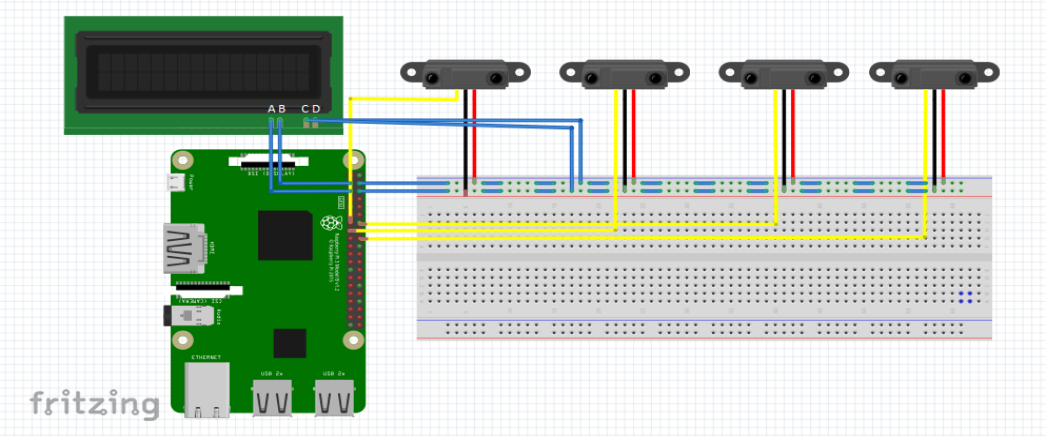


Figure 10 Câblage de Raspberry pi avec LCD et capteurs IR

Le tableau suivant explique les sorties de LCD 20x4

A	B	C	D
(SCL)	(SDA)	GND	VCC

Tableau 2 Les sorties LCD

3.3.2 PROGRAMMATION DE RASPBERRY PI

3.3.2.1 PROGRAMMATION DE RASPBERRY PI POUR LIRE LES DONNES ENVOYES PAR LES CAPTEURS IR

A l'aide de notre formation que nous avions obtenue dans le langage Python nous avions pu programmer Raspberry Pi pour lire les données envoyées par les capteurs **IR** comme le montre les figures suivantes :

Importer les librairies dans notre programme, librairies de GPIO, LCD, time, firebase :

```
1  import RPi.GPIO as GPIO
2  import time
3  import I2C_LCD_driver
4  import smbus
5  import firebase_admin
6  from firebase_admin import credentials
7  from firebase_admin import db
8
9  cred = credentials.Certificate('firebase.json')
10 firebase_admin.initialize_app(cred,{
11 'databaseURL':"https://smart-parking-c0c0e-default-rtdb.europe-west1.firebasedatabase.app/"})
12
```

Figure 11 Programme pour insérer les librairies

Déclaration des entrées de notre Raspberry pi, il y a 4 capteurs donc 4 entrées :

```
GPIO.setmode(GPIO.BCM)
GPIO.setup(23,GPIO.IN)
GPIO.setup(18,GPIO.IN)
GPIO.setup(27,GPIO.IN)
GPIO.setup(17,GPIO.IN)
```

Figure 12 Déclaration des entrées de Raspberry

Déclarer les variables pour faciliter la programmation

Nous avions assigner à chaque entrée une variable donc 4 :

```
while True:
 i=GPIO.input(23)
 a=GPIO.input(18)
 b=GPIO.input(17)
 c=GPIO.input(27)
```

Dans la figure suivante nous allons exposer quelques extraits du programme Python pour lire les données récoltées par les capteurs infrarouge :

```
27  if i==0 and a==0 and b==0 and c==0 :
28      ref = db.reference('smart parking')
29      ref.set({
30          "spot A":"indisponible","spot B":"indisponible","spot C":"indisponible","spot D":"indisponible"
31
32          })
33      mylcd.lcd_display_string(u"Welcome to Fst!        Full!")
34      time.sleep(1)
35      mylcd.lcd_clear()
36      time.sleep(0.6)
37      time.sleep(0.9)
38  elif i==0 and a==0 and b==0 and c==1 :
39      ref = db.reference('smart parking')
40      ref.set({
41          "spot A":"indisponible","spot B":"indisponible","spot C":"indisponible","spot D":"disponible"
42
43          })
44      time.sleep(0.9)
45      mylcd.lcd_display_string(u"Welcome to Fst!        1 spot disponible")
46      time.sleep(1)
47      mylcd.lcd_clear()
48      time.sleep(0.9)
```

```
188 elif i==1 and a==1 and b==1 and c==1 :
189     ref = db.reference('smart parking')
190     ref.set({
191         "spot A":"disponible","spot B":"disponible","spot C":"disponible","spot D":"disponible"
192
193         })
194     mylcd.lcd_display_string(u"Welcome to Fst!        4 spot disponible")
195     time.sleep(1)
196     mylcd.lcd_clear()
197     time.sleep(0.9)
198     time.sleep(0.9)
199
200
201 GPIO.cleanup()
```

Figure 13 Programme pour lire les données récoltées par les capteurs

Pour le programme complet voir Annexe 1,2,3,4,5,6.

3.3.2.2 AJOUTER UN PROGRAMME POUR ENVOYER LES DONNES A FIREBASE

a. C'est quoi Firebase ?

Firebase a évolué à partir d' 'Envolve', une startup antérieure fondée par 'James Tamplin' et 'Andrew Lee' en 2011.' Envolve' a fourni aux développeurs une API qui permet l'intégration de la fonctionnalité de chat en ligne dans leurs sites Web. Après avoir lancé le service de chat, 'Tamplin' et 'Lee' ont découvert qu'il était utilisé pour transmettre des données d'application qui n'étaient pas des messages de chat. Les développeurs utilisaient 'Envolve' pour synchroniser les données d'application telles que l'état du jeu en temps réel entre leurs utilisateurs. 'Tamplin' et Lee ont

décidé de séparer le système de chat et l'architecture en temps réel qui l'alimentait. Ils ont fondé Firebase en tant que société distincte en septembre 2011et l'ont lancée au public en avril 2012. **[5]**

b. Programme pour envoyer les données a Firebase

Importer les librairies de Firebase après les avoir installées sur Raspberry PI et insérer le non du fichier 'json' qui contient les informations de notre Firebase comme L'URL etc :
Firebase admin est une figure que nous avions installé sur Raspberry pi et le fichier firebase.json est un fichier unique qui contient toutes les information de notre base de données comme le montre la figure suivante:

```
5  import firebase_admin
6  from firebase_admin import credentials
7  from firebase_admin import db
8
9  cred = credentials.Certificate('firebase.json')
10 firebase_admin.initialize_app(cred,{
11 'databaseURL':"https://smart-parking-c0c0e-default-rtdb.europe-west1.firebasedatabase.app/"})
```

Figure 14 Importation des librairies Firebase

Envoyer les données récoltées par Raspberry PI :

Le programme suivant sert à créer un tableau dans notre base de données, ce tableau affiche liste les spots ainsi que leurs disponibilités :

```
    ref = db.reference('smart parking')
    ref.set({
        "spot A":"indisponible","spot B":"indisponible","spot C":"indisponible","spot D":"indisponible

        })
```

Figure 15 Envoyer les données a FireBase

3.3.2.3 PROGRAMME POUR AFFICHER LE NOMBRE DES SPOTS DISPONIBLES DANS LCD

Programme pour importer les librairies de notre LCD :

```
2  import time
3  import I2C_LCD_driver
4  import smbus
5
6
```

Figure 16 Importation des librairies LCD

Programme pour afficher le nombre des places disponibles sur LCD :

```
33      mylcd.lcd_display_string(u"Welcome to Fst!        Full!")
34      time.sleep(1)
35      mylcd.lcd_clear()
36      time.sleep(0.6)
37      time.sleep(0.9)
```

Figure 17 Programme pour afficher sur LCD

Pour le programme complet voir Annexe 1,2,3,4,5,6.

3.3.3 CREATION DE L'APPLICATION MOBILE AVEC ANDROID STUDIO

3.3.3.1 C'EST QUOI ANDROID STUDIO ?

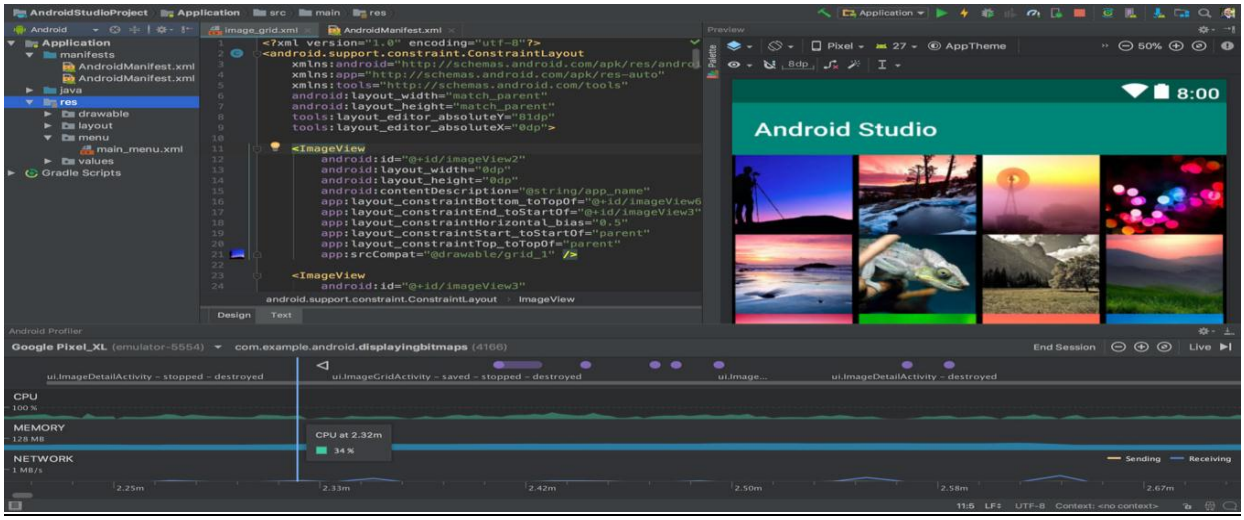


Figure 18 Interface Android studio [6]

Android Studio est l'environnement de développement intégré (IDE) officiel pour le développement d'applications Android, basé sur IntelliJ IDEA. En plus du puissant éditeur de code et des outils de développement d'IntelliJ, Android Studio offre encore plus de fonctionnalités qui améliorent votre productivité lors de la création d'applications Android **[7]**

3.3.3.2 LE DESIGNE DE L'APPLICATION MOBILE

Nous avons construit le désigne de l'application mobile avec un code XML et à l'aide de l'interface de android studio .

La figure suivante nous montre le design de l'application mobile, chaque texte est assigné avec un unique ID que nous allons utiliser pour récolter les données depuis Firebase :

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.motion.widget.MotionLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    app:layoutDescription="@xml/activity_main_scene"
    tools:context=".MainActivity">

    <TextView
        android:id="@+id/spota"
        android:layout_width="66dp"
        android:layout_height="24dp"
        android:layout_marginStart="128dp"
        android:layout_marginLeft="128dp"
        android:layout_marginTop="237dp"
        android:layout_marginEnd="216dp"
        android:layout_marginRight="216dp"
        android:text="@string/textview"
        app:layout_constraintBottom_toTopOf="@+id/spotb"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="1.0"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />
```

Figure 19 Désigne de l'application pour afficher spot A

Le programme du bouton refresh 'actualiser' pour actualiser la disponibilités des spots dans notre parking, même le bouton a un unique ID pour lui assigner le role d'actualiser les données :

```
    <Button
        android:id="@+id/button"
        android:layout_width="112dp"
        android:layout_height="45dp"
        android:layout_marginStart="125dp"
        android:layout_marginLeft="125dp"
        android:layout_marginTop="32dp"
        android:layout_marginEnd="159dp"
        android:layout_marginRight="159dp"
        android:layout_marginBottom="229dp"
        android:text="@string/refresh"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.0"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/spotb" />
```

Figure 20 Désigne du bouton actualiser

Voir Annexe 7et 8 pour le programme complet.

3.3.4 RECUPERER LES DONNES ENVOYEES A FIREBASE ET LES LIRE DANS ANDROID STUDIO

3.3.4.1 ETABLIR UNE CONNEXION ENTRE ANDROID STUDIO ET FIERBASE

Nous avions d'abord connecté Firebase avec android studio en insérant le fichier 'json' dans la section app puis nous avions importé le package de notre Firebase :

Importer le package et définir des variables pour faciliter la programmation :

```java
package com.fstf.smartp;

import ...

public class MainActivity extends AppCompatActivity {
TextView a,b,c,d;
Button btn;
DatabaseReference ref;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        a= findViewById(R.id.spota);
        b= findViewById(R.id.spotb);
        c = findViewById(R.id.spotc);
        d= findViewById(R.id.spotd);
        btn= findViewById(R.id.button);
```

Figure 21 Importation du package de la base de données et déclaration des variables

Définir le rôle de chaque texte à partir de lier son ID avec la disponibilité du spot à afficher

```java
        btn.setOnClickListener(view -> {
            ref = FirebaseDatabase.getInstance().getReference().child("smart parking");
            ref.addValueEventListener(new ValueEventListener() {
                @Override
                public void onDataChange(@NonNull DataSnapshot dataSnapshot) {
                    String spot_A= Objects.requireNonNull(dataSnapshot.child("spot A").getValue()).toString();
                    String spot_B= Objects.requireNonNull(dataSnapshot.child("spot B").getValue()).toString();
                    String spot_C= Objects.requireNonNull(dataSnapshot.child("spot C").getValue()).toString();
                    String spot_D= Objects.requireNonNull(dataSnapshot.child("spot D").getValue()).toString();
                    a.setText(spot_A);
                    b.setText(spot_B);
                    c.setText(spot_C);
                    d.setText(spot_D);


                }

                @Override
                public void onCancelled(@NonNull @org.jetbrains.annotations.NonNull DatabaseError error) {

                }

            });
        });
    }
}
```

Figure 22 Donner le rôle a chaque texte en utilisant l'ID

CONCLUSIONS & PERSPECTIVES

Nous avons pu réaliser un parking intelligent qui communique la disponibilité des spots et qui affiche le nombre des places disponible sur LCD.

Comme nous avons pu construire une base de données online, qui nous permet de récolter les informations envoyées par le Raspberry PI.

Finalement nous avons pu construire une application mobile.

L'application fonctionne bien et joue son rôle à communiquer les données effectivement

Le Raspberry PI envoi les données avec efficacité

Nous n'avions pas pu ajouter une voix qui communique la disponibilité des spots ; malheureusement le temps ne nous a pas suffit

Nous sommes intéressés par le domaine de l'intelligence artificiel, c'est pour cela que nous espérant avoir l'opportunité de réaliser un robot intelligent.

Bibliographie

[1] https://fst-usmba.ac.ma/laboratoire_ssc/ (20/6/2021)

[2] Salma, Rashidah Funke Olanrewaju, Morshidi Malik Arman, Smart Parking Guidance System Using 360o Camera and Haar-Cascade Classifier on IoT System, International Journal of Recent Technology and Engineering (IJRTE) ISSN: 2277-3878,Volume-8, Issue-2S11, September 2019 (15/5/2021)

[3] https://www.researchgate.net/ (10/6/2021)

[4] https://www.nwave.io/pros-and-cons-of-smart-parking-systems/ (4/6/2021)

[5] https://en.wikipedia.org/wiki/Firebase (7/6/2021)

[6] https://developer.android.com/studio (20/5/2021)

[7] https://developer.android.com/studio/intro (15/5/2021)

Annexe 1

Le programme pour recevoir les données envoyées par les capteurs IR, afficher le nombre de spot disponible sur LCD et envoyer les données récoltées a Firebase :

```
import RPi.GPIO as GPIO
import time
import I2C_LCD_driver
import smbus
import firebase_admin
from firebase_admin import credentials
from firebase_admin import db

cred = credentials.Certificate('firebase.json')
firebase_admin.initialize_app(cred,{
'databaseURL':"https://smart-parking-c0c0e-default-rtdb.europe-west1.firebasedatabase.app/"})



GPIO.setmode(GPIO.BCM)
GPIO.setup(23,GPIO.IN)
GPIO.setup(18,GPIO.IN)
GPIO.setup(27,GPIO.IN)
GPIO.setup(17,GPIO.IN)
mylcd = I2C_LCD_driver.lcd()
while True:
 while True:
  i=GPIO.input(23)
  a=GPIO.input(18)
  b=GPIO.input(17)
  c=GPIO.input(27)
  if i==0 and a==0 and b==0 and c==0 :
    ref = db.reference('smart parking')
    ref.set({
       "spot A":"indisponible","spot B":"indisponible","spot C":"indisponible","spot D":"indisponible"

       })
    mylcd.lcd_display_string(u"Welcome to Fst!       Full!")
    time.sleep(1)
    mylcd.lcd_clear()
    time.sleep(0.6)
```

ANNEXE 2

```
 time.sleep(0.9)
elif i==0 and a==0 and b==0 and c==1 :
   ref = db.reference('smart parking')
   ref.set({
      "spot A":"indisponible","spot B":"indisponible","spot C":"indisponible","spot D":"disponible"

      })
   time.sleep(0.9)
   mylcd.lcd_display_string(u"Welcome to Fst!        1 spot disponible")
   time.sleep(1)
   mylcd.lcd_clear()
   time.sleep(0.9)
elif i==0 and a==0 and b==1 and c==0 :
   ref = db.reference('smart parking')
   ref.set({
      "spot A":"indisponible","spot B":"indisponible","spot C":"disponible","spot D":"indisponible"

      })
   time.sleep(0.9)
   mylcd.lcd_display_string(u"Welcome to Fst!        1 spot disponible")
   time.sleep(1)
   mylcd.lcd_clear()
   time.sleep(0.9)
elif i==0 and a==0 and b==1 and c==1 :
   ref = db.reference('smart parking')
   ref.set({
      "spot A":"indisponible","spot B":"indisponible","spot C":"disponible","spot D":"disponible"

      })
   time.sleep(0.9)
   mylcd.lcd_display_string(u"Welcome to Fst!        2 spot disponible")
   time.sleep(1)
   mylcd.lcd_clear()
   time.sleep(0.9)
elif i==0 and a==1 and b==0 and c==0 :
   ref = db.reference('smart parking')
   ref.set({
      "spot A":"indisponible","spot B":"disponible","spot C":"indisponible","spot D":"indisponible"

      })
   time.sleep(0.9)
```

ANNEXE 3

```
    mylcd.lcd_display_string(u"Welcome to Fst!        1 spot disponible")
    time.sleep(1)
    mylcd.lcd_clear()
    time.sleep(0.9)
elif i==0 and a==1 and b==0 and c==1 :
    ref = db.reference('smart parking')
    ref.set({
        "spot A":"indisponible","spot B":"disponible","spot C":"indisponible","spot D":"disponible"

        })
    time.sleep(0.9)
    mylcd.lcd_display_string(u"Welcome to Fst!        2 spot disponible")
    time.sleep(1)
    mylcd.lcd_clear()
    time.sleep(0.9)
elif i==0 and a==1 and b==1 and c==0 :
    ref = db.reference('smart parking')
    ref.set({
        "spot A":"indisponible","spot B":"disponible","spot C":"disponible","spot D":"indisponible"

        })
    time.sleep(0.9)
    mylcd.lcd_display_string(u"Welcome to Fst!        2 spot disponible")
    time.sleep(1)
    mylcd.lcd_clear()
    time.sleep(0.9)
elif i==0 and a==1 and b==1 and c==1 :
    ref = db.reference('smart parking')
    ref.set({
        "spot A":"indisponible","spot B":"disponible","spot C":"disponible","spot D":"disponible"

        })
    time.sleep(0.9)
    mylcd.lcd_display_string(u"Welcome to Fst!        3 spot disponible")
    time.sleep(1)
    mylcd.lcd_clear()
    time.sleep(0.9)
elif i==1 and a==0 and b==0 and c==0 :
    ref = db.reference('smart parking')
    ref.set({
        "spot A":"disponible","spot B":"indisponible","spot C":"indisponible","spot D":"indisponible"
```

ANNEXE 4

```
    })
  time.sleep(0.9)
  mylcd.lcd_display_string(u"Welcome to Fst!        1 spot disponible")
  time.sleep(1)
  mylcd.lcd_clear()
  time.sleep(0.9)
elif i==1 and a==0 and b==0 and c==1 :
  ref = db.reference('smart parking')
  ref.set({
    "spot A":"disponible","spot B":"indisponible","spot C":"indisponible","spot D":"disponible"

    })
  time.sleep(0.9)
  mylcd.lcd_display_string(u"Welcome to Fst!        2 spot disponible")
  time.sleep(1)
  mylcd.lcd_clear()
  time.sleep(0.9)
elif i==1 and a==0 and b==1 and c==0 :
  ref = db.reference('smart parking')
  ref.set({
    "spot A":"disponible","spot B":"disponible","spot C":"indisponible","spot D":"indisponible"

    })
  time.sleep(0.9)
elif i==1 and a==0 and b==1 and c==1 :
  ref = db.reference('smart parking')
  ref.set({
    "spot A":"disponible","spot B":"indisponible","spot C":"disponible","spot D":"disponible"

    })
  time.sleep(0.9)
  mylcd.lcd_display_string(u"Welcome to Fst!        3 spot disponible")
  time.sleep(1)
  mylcd.lcd_clear()
  time.sleep(0.9)
elif i==1 and a==1 and b==0 and c==0 :
  ref = db.reference('smart parking')
  ref.set({
    "spot A":"disponible","spot B":"disponible","spot C":"indisponible","spot D":"indisponible"
```

ANNEXE 5

```
    })
  time.sleep(0.9)
  mylcd.lcd_display_string(u"Welcome to Fst!        2 spot disponible")
  time.sleep(1)
  mylcd.lcd_clear()
  time.sleep(0.9)
elif i==1 and a==1 and b==0 and c==1 :
  ref = db.reference('smart parking')
  ref.set({
     "spot A":"disponible","spot B":"disponible","spot C":"indisponible","spot D":"disponible"

     })
  time.sleep(0.9)
  mylcd.lcd_display_string(u"Welcome to Fst!        3 spot disponible")
  time.sleep(1)
  mylcd.lcd_clear()
  time.sleep(0.9)
elif i==1 and a==1 and b==1 and c==0 :
  ref = db.reference('smart parking')
  ref.set({
     "spot A":"disponible","spot B":"disponible","spot C":"disponible","spot D":"indisponible"

     })
  time.sleep(0.9)
  mylcd.lcd_display_string(u"Welcome to Fst!        3 spot disponible")
  time.sleep(1)
  mylcd.lcd_clear()
  time.sleep(0.9)
elif i==1 and a==1 and b==1 and c==1 :
  ref = db.reference('smart parking')
  ref.set({
     "spot A":"disponible","spot B":"disponible","spot C":"disponible","spot D":"disponible"

     })
  mylcd.lcd_display_string(u"Welcome to Fst!        4 spot disponible")
  time.sleep(1)
  mylcd.lcd_clear()
  time.sleep(0.9)
  time.sleep(0.9)

GPIO.cleanup()
```

ANNEXE 6

Programme du désigne de l'application :

```xml
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.motion.widget.MotionLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    app:layoutDescription="@xml/activity_main_scene"
    tools:context=".MainActivity">

    <TextView
        android:id="@+id/spota"
        android:layout_width="66dp"
        android:layout_height="24dp"
        android:layout_marginStart="128dp"
        android:layout_marginLeft="128dp"
        android:layout_marginTop="237dp"
        android:layout_marginEnd="216dp"
        android:layout_marginRight="216dp"
        android:text="@string/textview"
        app:layout_constraintBottom_toTopOf="@+id/spotb"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="1.0"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    <TextView
        android:id="@+id/spotb"
        android:layout_width="85dp"
        android:layout_height="26dp"
        android:layout_marginStart="130dp"
        android:layout_marginLeft="130dp"
        android:layout_marginTop="3dp"
        android:layout_marginEnd="215dp"
        android:layout_marginRight="215dp"
        android:layout_marginBottom="145dp"
        android:text="@string/textview"
        app:layout_constraintBottom_toTopOf="@+id/button"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.0"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/spota" />

    <Button
        android:id="@+id/button"
        android:layout_width="112dp"
        android:layout_height="45dp"
        android:layout_marginStart="125dp"
        android:layout_marginLeft="125dp"
        android:layout_marginTop="32dp"
        android:layout_marginEnd="159dp"
        android:layout_marginRight="159dp"
        android:layout_marginBottom="229dp"
        android:text="@string/refresh"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.0"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/spotb" />
```

ANNEXE 7

```
<TextView
    android:id="@+id/textView"
    android:layout_width="63dp"
    android:layout_height="25dp"
    android:layout_marginStart="50dp"
    android:layout_marginLeft="50dp"
    android:layout_marginTop="237dp"
    android:layout_marginEnd="12dp"
    android:layout_marginRight="12dp"
    android:layout_marginBottom="6dp"
    android:text="@string/spot_a"
    app:layout_constraintBottom_toTopOf="@+id/textView3"
    app:layout_constraintEnd_toStartOf="@+id/spota"
    app:layout_constraintHorizontal_bias="0.928"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent" />
```

```
<TextView
    android:id="@+id/textView2"
    android:layout_width="63dp"
    android:layout_height="20dp"
    android:layout_marginStart="68dp"
    android:layout_marginLeft="68dp"
    android:layout_marginEnd="12dp"
    android:layout_marginRight="12dp"
    android:text="@string/spot_b"
    app:layout_constraintEnd_toStartOf="@+id/spotb"
    app:layout_constraintHorizontal_bias="1.0"
    app:layout_constraintStart_toStartOf="parent"
    tools:ignore="MissingConstraints"
    tools:layout_editor_absoluteY="259dp" />
```

```
<TextView
    android:id="@+id/textView3"
    android:layout_width="70dp"
    android:layout_height="23dp"
    android:layout_marginStart="75dp"
    android:layout_marginLeft="75dp"
    android:layout_marginTop="1dp"
    android:layout_marginEnd="12dp"
    android:layout_marginRight="12dp"
    android:layout_marginBottom="2dp"
    android:text="@string/spot_c"
    app:layout_constraintBottom_toTopOf="@+id/textView4"
    app:layout_constraintEnd_toStartOf="@+id/spotc"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/textView"
    tools:ignore="MissingConstraints" />
```

ANNEXE 8

```
108
109     <TextView
110         android:id="@+id/textView4"
111         android:layout_width="68dp"
112         android:layout_height="25dp"
113         android:layout_marginStart="75dp"
114         android:layout_marginLeft="75dp"
115         android:layout_marginTop="6dp"
116         android:layout_marginEnd="12dp"
117         android:layout_marginRight="12dp"
118         android:layout_marginBottom="404dp"
119         android:text="@string/spot_d"
120         app:layout_constraintBottom_toBottomOf="parent"
121         app:layout_constraintEnd_toStartOf="@+id/spotd"
122         app:layout_constraintStart_toStartOf="parent"
123         app:layout_constraintTop_toBottomOf="@+id/textView3"
124         tools:ignore="MissingConstraints" />
125
126     <TextView
127         android:id="@+id/spotc"
128         android:layout_width="wrap_content"
129         android:layout_height="wrap_content"
130         android:layout_marginStart="12dp"
131         android:layout_marginLeft="12dp"
132         android:layout_marginTop="1dp"
133         android:layout_marginEnd="224dp"
134         android:layout_marginRight="224dp"
135         android:text="TextView"
136         app:layout_constraintBottom_toTopOf="@+id/spotd"
137         app:layout_constraintEnd_toEndOf="parent"
138         app:layout_constraintStart_toEndOf="@+id/textView3"
139         app:layout_constraintTop_toBottomOf="@+id/spotb"
140         tools:ignore="HardcodedText,MissingConstraints" />
142     <TextView
143         android:id="@+id/spotd"
144         android:layout_width="wrap_content"
145         android:layout_height="wrap_content"
146         android:layout_marginStart="12dp"
147         android:layout_marginLeft="12dp"
148         android:layout_marginEnd="224dp"
149         android:layout_marginRight="224dp"
150         android:layout_marginBottom="130dp"
151         android:text="TextView"
152         app:layout_constraintBottom_toTopOf="@+id/button"
153         app:layout_constraintEnd_toEndOf="parent"
154         app:layout_constraintStart_toEndOf="@+id/textView4"
155         app:layout_constraintTop_toBottomOf="@+id/spotc"
156         tools:ignore="HardcodedText,MissingConstraints" />
157
158 </androidx.constraintlayout.motion.widget.MotionLayout>
```