# International Institute of Information Technology, Bangalore

## Internet of Things Project

## Smart Parking System

**Under the Guidance of**
**Prof. Jyotsna Bapat**

**May 23, 2021**


Biri Arya Vardhan Reddy IMT2018017

Chebrolu Suchith kumar IMT2018019

# Introduction :-

This report presents smart parking system, our project on the Internet of Things. The concept of smart city is all about using technologies to solve our daily problems. This project is a good example how the Internet of Things can be used for applications of smart city.

If you are living in a city, you probably had cases when you are looking too long for a parking lot. With this project, everything can be quick and simple. Vacant parking lots are shown in the application which uses sensors in IOT technologies to receive information about lots in real time.

The solution we provide can successfully handle the issue of parking on several levels. Easy search for parking lots will help millions of drivers save their precious time, working most effectively for extra-large and multi-story parking spaces. With this solution, smart parking can optimize traffic flows in entire cities and save valuable time of many citizens.

# Parts and tools used :-

*Hardware:*

1. Ultrasonic sensor :
Ultrasonic sensing is one of the best ways to sense proximity with high reliability. Ultrasonic sensors work by sending out a sound wave at a frequency above the range of human hearing. The transducer of the sensor acts as a microphone to receive and send the ultrasonic sound.

Ultrasonic sensors transmit sound waves toward a target and measure the time it took for the reflected waves to return to the receiver. Then, distance measurement is done, based on the measurement of time-of-flight (time lapse between the sending and receiving of the ultrasonic pulse) and the speed of sound. It is a electronic device that can convert reflected sound into an electrical signal for processing. In this project, we have used ultrasonic sensor with the micro-controller platform, Raspberry pi.

2. Raspberry pi :
The Raspberry Pi is a cheap computer that runs Linux, and provides a set of GPIO (general purpose input/output) pins. It allows us to control electronic components for physical computing and explore the Internet of Things (IoT).

As well as controlling outputs with the GPIO pins, the Raspberry Pi can collect data about the outside world using sensors. Other sensors can be added to the pins of the Pi, either as individual components connected to GPIO pins, or through an add-on board. As mentioned earlier, in this project, we will be using the Raspberry Pi

with ultrasonic sensors to detect the presence of objects (cars).

3. Jumper wires (F to F) :
Jumper wires are simply wires that have connector pins at each end, allowing them to be used to connect two points to each other without soldering.

*Software :*

1. Coding language:
We are using Python version 3 in Raspberry Pi. For the application, we are using Flutter, Python Flask along with HTML.

2. Cloud platform:
We are using the ThingSpeak cloud to store the status values of parking lots.

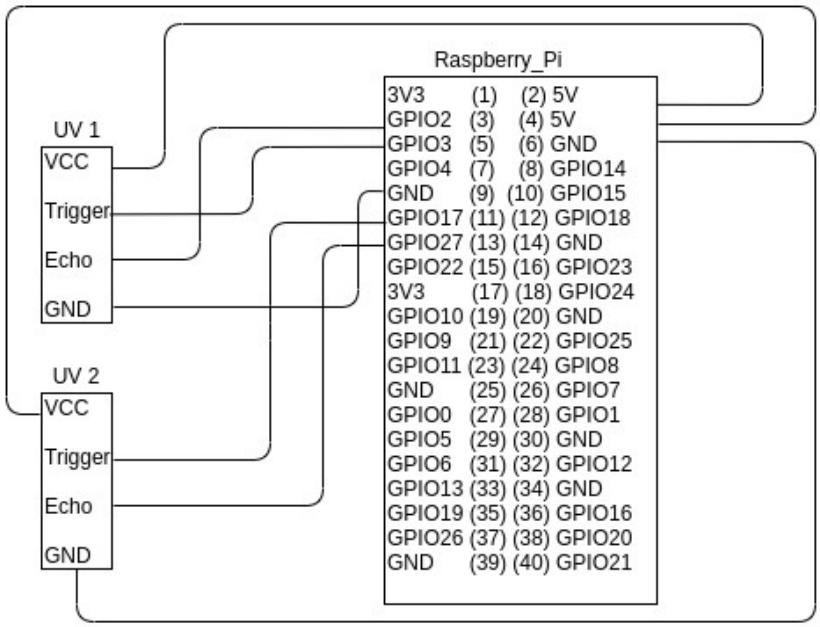
# Proposed Solution :

Let's look at the working of the smart parking system. Ultrasonic sensors are placed at the center of each parking lot. A sensor detects a parked car by measuring the distance to the nearest obstacle, in our case to the car bottom. The measurements are periodically sent to the micro-controller (Raspberry Pi). Then, after processing the readings, the sensor states are determined and sent to the cloud (ThingSpeak) where they're stored. When a driver accesses the system with an application, sensor state is read and displayed. The state represents if the parking lot is free, occupied or not sure. Here is the algorithm:
If the sensor detects nothing up to 50 centimeters, the status is set to free and shown to the user. The occupied range is between 10 and 50 centimeters. Below 10, the status is dirty. It means the sensor might be covered with mud or an obstacle and requires checking. All of these states can be easily configured.
The Raspberry Pi code:

```
import urllib.request
import requests
import threading
import json
import random
import time
import board
import adafruit_hcsr04
sonar1 = adafruit_hcsr04.HCSR04(trigger_pin=board.D3, echo_pin=board.D2)
sonar2 = adafruit_hcsr04.HCSR04(trigger_pin=board.D17, echo_pin=board.D27)
# Define a function to get status of parking lots from the distance
def get_status(distance):
    if(distance<10):
        return 2
    elif(distance<50):
        return 1
    else:
        return 0
# Define a function that will post on server every 15 Seconds
def thingspeak_post():
    threading.Timer(15,thingspeak_post).start()
    dist1=sonar1.distance
    dist2=sonar2.distance
    URl='https://api.thingspeak.com/update?api_key='
    KEY='BLDX37OWD7PY0WXI'
    val1=get_status(dist1)
    val2=get_status(dist2)
    HEADER='&field1={}&field2={}'.format(val1,val2)
    NEW_URL=URl+KEY+HEADER
    data=urllib.request.urlopen(NEW_URL)
    print("Distance1:",dist1,",Status1:",val1)
    print("Distance2:",dist2,",Status2:",val2)
if __name__ == '__main__':
    print("Status: Occupied-1 , Vacant-0 , Not sure-2")
    thingspeak_post()

```

The circuit diagram:




We initially tested our hardware side of things by connecting the raspberry pi with ThingSpeak. We then pulled the readings from the cloud using a simple flask app, which then processed the readings to decide occupancy, and pushed it into a basic html script.

Our main application was built using flutter. Flutter is Google’s UI toolkit to build fast and neat applications. It is natively compiled and can be used to build iOS, web and android applications from a single codebase.

Flutter builds apps as a combination of pages and widgets. Each page has a certain set of widgets, and routes are defined to connect the different pages. Our app has two pages, one with the home screen, which has a button that triggers the shift to the other page, which holds information about the parking lot (the view page).

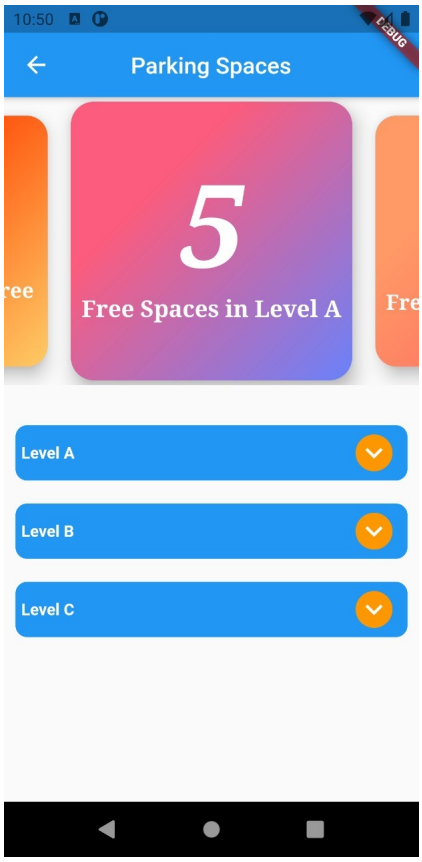
For a comprehensive look into the app, assume a parking lot with three levels and 10 spots in each level. The view page can be divided into two parts, the top half shows us statistics about free spots in each level, while the bottom half shows which specific spots are available.

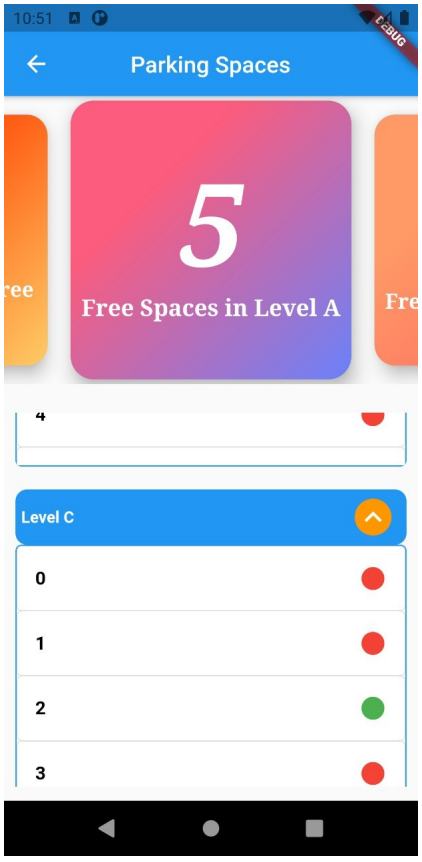
To connect the flutter application and our flask application, we make use of REST API, a framework that allows us to use http requests to access and transfer data. The flutter application is connected to the local host, through which it pulls the data pushed out by flask. The data is pulled in the form of a JSON string, which is then modified for use. Flutter does this using a module called “async”, which enables flutter to wait for a response from a function.

In the current state of the application, given that we have only a couple of “physical parking lots”, we have randomized the occupancy of the remaining parking lots, and every lot gets updated whenever we toggle the list view of the different levels. The occupancy statistics in the top half of the page also get updated every 5 seconds, made possible by the periodic timer present in the “async” module.

## Results :

Here is how the final application looks like:


The results can also be seen in the demo video we made for this project. The demo video covers all aspects of the project in more detail. The link to the demo video:

https://drive.google.com/file/d/1ZoZVYiwj44QhwED0uWxkFfHvUnnCa2qu/view?usp=drivesdk

The demo basically has two parts, the hardware side and the software side. The hardware side includes explanations of how the readings from the ultrasonic sensors are recorded, how the algorithm is implemented, and how the status states of parking lots are uploaded to the ThingSpeak cloud. The software side shows how

the application looks and how the states are read from the cloud and displayed in the application.

# Conclusion :

Future study:

When we talk about this particular application, in the future, instead of just using one type of sensors, we can install cameras in the parking areas and use image processing along with ultrasonic sensors. We would also like to add an other components to it, where users can reserve parking spots before hand, and even find their cars in the parking spot. We would like to try integrating our application with a map interface like Google maps. These would make the current app into a robust system that can be scaled easily.

The project is just not limited to the smart parking where we show if a car is present in a parking lot or not. With the same software and hardware, it can be extended to other cases where we need to say if an object is present or not, like whether a human is sitting on a seat or not in movie halls, if a patient is on the bed or not in hospitals, if a train has arrived on the platform or not in railway station, if an animal is in its cage or not in the zoo and so on.

Summary:
Using Raspberry Pi and ultrasonic sensors, we were able to build an application that shows if parking lots are occupied or not in real time. The challenges we faced on the software end of things include the completely new syntax and style of coding. Flutter is quite unconventional at first due to the presence of an overwhelming codebase, and this took time to sink in. Connecting to the local host also was challenging as the implementation to this has often changed a lot over the time flutter has existed, making the task of finding up to date documentation difficult. Since both of us are from software background, we had to put in extra effort to learn how to interact with hardware components and make the circuit connections.

Final thoughts:
This was a learning experience for us, as we were introduced to new technologies and new APIs (interfacing with Pi and ultrasonic sensors, application development with flutter, flask). We believe this project will improve the efficiency and functionality of vehicle parking spaces and make lives easier for citizens. We believe this experience will help increase our familiarity with the working of developing applications and the Internet of Things, and are thankful for the opportunity.

# References :

[1] Using ultrasonic distance sensors with Python :- https://learn.adafruit.com/ultrasonic-sonar-distance-sensors/python-circuitpython

[2] Using ThingSpeak cloud with Python :-
Retrieving data: https://youtu.be/Q3NPt9fxS5A
Sending data: https://youtu.be/whXaVYSXItQ

[3] Flutter :-

https://flutter.dev/docs

[4] Flask and Flutter integration :-
https://stackoverflow.com/questions/65375920/connecting-my-python-backend-with-my-flutter-frontend-using-flask

[5] Repeating function in dart :-
https://stackoverflow.com/questions/14946012/how-do-i-run-a-reoccurring-function-in-dart

***Thank you***