SMART PARKING

IOT_PHASE 3

Hardware Components:

Ultrasonic Sensors: To detect parking space occupancy, you will need ultrasonic sensors. These sensors measure the distance to objects and are commonly used in IoT projects.

Raspberry Pi: You'll use a Raspberry Pi as the central controller to collect and send data to the cloud or a mobile app.

Internet Connectivity: Ensure that your Raspberry Pi is connected to the internet. You can use Wi-Fi or Ethernet for this purpose.

Step 1: Assemble Hardware:

Connect the ultrasonic sensors to the Raspberry Pi. Typically, ultrasonic sensors have four pins: VCC, GND, Trig (Trigger), and Echo. Connect these pins to the corresponding GPIO pins on the Raspberry Pi.

Step 2: Set Up the Raspberry Pi:

Make sure your Raspberry Pi is running an up-to-date operating system. You can use Raspberry Pi OS or any compatible Linux distribution.

Install Python on the Raspberry Pi, if not already installed.

Step 3: Python Script for Data Collection:

Write Python scripts to collect data from the ultrasonic sensors. You can use the RPi.GPIO library to interface with the GPIO pins. Here's an example script for reading data from an ultrasonic sensor and printing it:

Program:

```
import RPi.GPIO as GPIO

import time


# Set GPIO pin numbers
```

```
TRIG_PIN = 23
ECHO_PIN = 24

# Setup GPIO
GPIO.setmode(GPIO.BCM)
GPIO.setup(TRIG_PIN, GPIO.OUT)
GPIO.setup(ECHO_PIN, GPIO.IN)

try:
    while True:
        # Trigger the sensor
        GPIO.output(TRIG_PIN, True)
        time.sleep(0.00001)
        GPIO.output(TRIG_PIN, False)

        # Wait for the echo
        while GPIO.input(ECHO_PIN) == 0:
            pulse_start = time.time()

        while GPIO.input(ECHO_PIN) == 1:
            pulse_end = time.time()

        # Calculate distance
        pulse_duration = pulse_end - pulse_start
        distance = (pulse_duration * 34300) / 2

        print(f"Distance: {distance} cm")
        time.sleep(1)  # Adjust the delay as needed

except KeyboardInterrupt:
    GPIO.cleanup()
```

Step 4: Send Data to the Cloud/Server:

To send the collected data to the cloud or a mobile app server, you need to implement the communication part. This can involve making HTTP requests, using MQTT, or other protocols. You will also need a cloud server or mobile app server to receive and process this data.

Depending on the specific protocol and server you choose, the code will differ. For example, to send data using HTTP, you can use the requests library in Python.

Remember, this is a basic example, and you'll need to adapt the code to your specific requirements, sensors, and cloud/server setup. It's essential to secure your system and consider the architecture for scaling and reliability as you move forward in your project.

****