SMART PARKING USING IOT

Hardware setup:

IoT-based smart parking system deployment requires integrating various devices, sensors, and microcontrollers.

- ◇ Raspberry Pi
- ◇ Ultrasonic Sensors (HC-SR04)
- ◇ Breadboard and jumper wires.

Data collection:

some use QR codes to identify available parking spots, while others use sensors to detect when a car leaves a parking spot.

IOT Platform:

can be a microcontroller transmitting data to the cloud environment or a Bluetooth beacon.

With its help, consumers can control parking locally.

Data Storage:

The received data is stored in a database. Depending on the system's design, the data may be stored locally or in a cloud-based solution.

Data Analysis:

The stored data can be used for analytics purposes.

This includes trends in parking space usage, peak hours, and other insights that can be valuable for urban planning.

Early warning system:

Sensor technology to monitor parking spaces and provides real-time information to drivers About availability of parking spots.

User interface:

Users can access the parking information through a mobile app or a website.

This interface shows them the availability of parking spaces in real-time.

Python Script:

```python
import RPi.GPIO as GPIO
import time
  # Set GPIO pins
TRIG_PIN = 23
```

```
ECHO_PIN = 24

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

 def get_distance():
    GPIO.output(TRIG_PIN, True)
    time.sleep(0.00001)
    GPIO.output(TRIG_PIN, False)

    pulse_start = time.time()
    while GPIO.input(ECHO_PIN) == 0:
        pulse_start = time.time()

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

    pulse_duration = pulse_end - pulse_start
```

```
    distance = pulse_duration * 17150
    distance = round(distance, 2)

    return distance
```

♧ Connect the ultrasonic sensor to the Raspberry Pi.

♧ Run the Python program on the Raspberry Pi.

♧Run the Flask application on the server.

This is a simplified example. In a real-world scenario, you would need to add more functionality, handle multiple parking spaces, implement real-time updates, and integrate with a database for storage. Additionally, consider security aspects and error handling in a production environment.