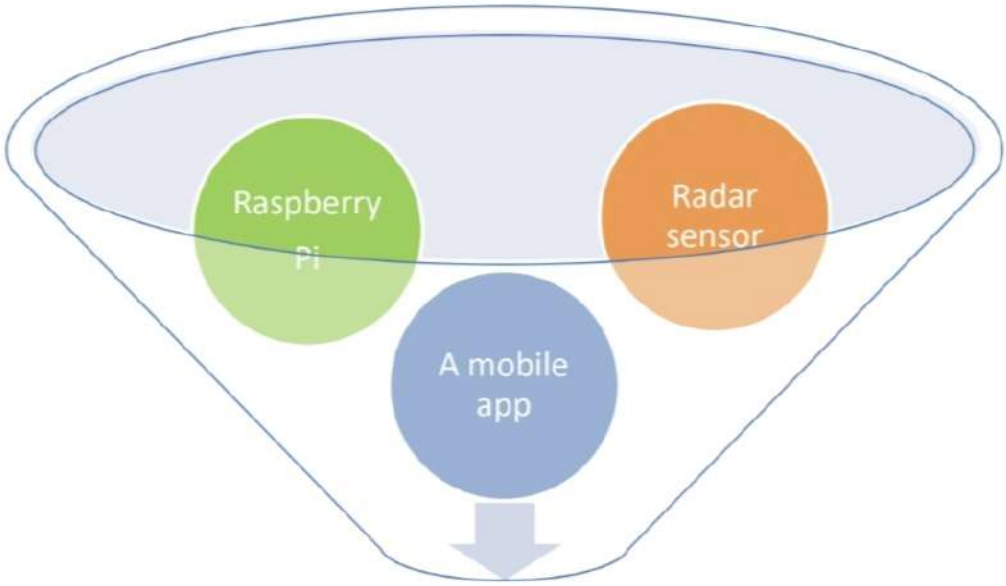
DEVELOPMENT

Introduction

- Smart Parking using iot is a technology-based system that employs iot sensors to monitor and optimize parking space availability, improving parking management and the overall user experience by providing real-time information to drivers and operators.

components



Parking space analyzer

PROCESS OF RADAR SENSOR

- **Transmitter:**
  The radar sensor contains a transmitter that emits radio waves, typically in the microwave frequency range.
- **Propagation:**
  These radio waves propagate through the air and interact with objects in their path.
- **Reflection:**
  When the radio waves encounter an object, such as a vehicle or any obstruction, they are reflected back towards the sensor.
- **Receiver:**
  The radar sensor also has a receiver that detects the reflected waves.
- **Doppler Effect (Optional):**
  Some radar sensors use the Doppler effect to detect the movement of objects. When an object is moving (like a car), the frequency of the reflected waves changes, allowing the sensor to detect motion.

- ***Analysis:***

  The radar sensor analyzes the characteristics of the reflected waves, such as the time taken for the waves to return and the frequency shift (if using Doppler). This information is used to determine the distance, size, and movement of the detected objects.

- ***Occupancy Detection:***

  Based on the analysis, the radar sensor determines whether a parking space is occupied or vacant. If an object is detected within a predefined distance range and meets certain criteria (e.g., size or movement), the space is considered occupied.

- ***Data Transmission :***

  The radar sensor may transmit the occupancy information to a central system, like a Raspberry Pi or a cloud server, for further processing and storage.

- ***Integration with Parking System:***

  The occupancy information can be integrated into a larger smart parking system, where it is made available to users through a mobile app or displayed on electronic signage.

Data transmission to raspberry from radar sensor

Wiring the Radar Sensor to Raspberry Pi:

- ***Identify GPIO Pins on Raspberry Pi:***

  Identify the GPIO (General Purpose Input/Output) pins on your Raspberry Pi. You can find GPIO pinout diagrams for your specific Raspberry Pi model online.

- ***Power Supply:***

  Connecting the power (Vcc) pin of the radar sensor to a 5V GPIO pin on the Raspberry Pi.

  Connect the ground (GND) pin of the radar sensor to a GND GPIO pin on the Raspberry Pi.

- ***Signal Pin:***

  Connecting the signal (output) pin of the radar sensor to a GPIO pin on the Raspberry Pi. This will be the pin that carries the information about detected objects.

- ***Optional Components:***

  Depending on the specifications of our radar sensor, you may need to add additional components like resistors or level shifters to ensure compatibility with the Raspberry Pi's GPIO voltage levels.

PROGRAM FOR DATA TRANSFER

```
import RPi.GPIO as GPIO
import time

# Set up GPIO
GPIO.setmode(GPIO.BCM)
radar_sensor_pin = 17  # Replace with the actual GPIO pin you used
GPIO.setup(radar_sensor_pin, GPIO.IN)

def read_radar_sensor():
    try:
        while True:
            # Read the status of the GPIO pin
            if GPIO.input(radar_sensor_pin) == GPIO.HIGH:
                return True  # Object detected
```

```
else:
            return False  # No object detected

        time.sleep(1)  # Wait for 1 second before checking again

    except KeyboardInterrupt:
        GPIO.cleanup()  # Clean up GPIO settings on Ctrl+C

if __name__ == "__main__":
    try:
        while True:
            detection_status = read_radar_sensor()

            if detection_status:
                print("Object detected!")
                # Perform actions based on the detection, e.g., transmit data to Raspberry Pi

            time.sleep(1)  # Adjust the sleep duration based on your application

    except KeyboardInterrupt:
        GPIO.cleanup()  # Clean up GPIO settings on Ctrl+C
```

PROGRAM FOR TRANSMIT DATA TO MOBILE APP FROM RASPBERRY PI

```
from flask import Flask, jsonify

app = Flask(__name__)

# Assuming 'collected_data' is a variable holding the data you want to send to the mobile app
collected_data = {"temperature": 25, "humidity": 60}

@app.route('/get_data', methods=['GET'])
def get_data():
    return jsonify(collected_data)

if __name__ == '__main__':
    app.run(debug=True, host='0.0.0.0')
```

(install flask on raspberry pi)

MOBILE SIDE PROCESS

- *HTTP Request Script (client.py):*

  Sends a GET request to the Flask server running on the Raspberry Pi to the /get data endpoint .

  Receives the response, assuming it's in JSON format, and prints the collected data.

PROGRAM FOR SHOWING DATA ON MOBILE

```
import requests

def get_data_from_raspberry_pi():
    raspberry_pi_ip = 'raspberry_pi_ip'  # Replace with the actual IP address of your Raspberry Pi
    url = f'http://{raspberry_pi_ip}:5000/get_data'  # Assuming Flask is running on the default port 5000

    try:
        response = requests.get(url)
        if response.status_code == 200:
            data_from_raspberry_pi = response.json()
            print("Data received from Raspberry Pi:")
            print(data_from_raspberry_pi)
            # Process the data as needed in your mobile app
        else:
            print(f"Error: {response.status_code}")
    except Exception as e:
        print(f"An error occurred: {e}")

if __name__ == '__main__':
    get_data_from_raspberry_pi()
```