Building an IoT sensor system to detect parking space occupancy using ultrasonic sensors and Raspberry Pi integration involves several steps. Here's a high-level overview of the process:

**Hardware Setup**:

Acquire the necessary components, including Raspberry Pi, ultrasonic sensors, jumper wires, and a power source (e.g., a power bank).

Connect the ultrasonic sensors to the Raspberry Pi GPIO pins using jumper wires. Typically, ultrasonic sensors have four pins: VCC, GND, Trig (trigger), and Echo. Connect these pins accordingly.

**Raspberry Pi Configuration:**

Set up your Raspberry Pi with a fresh installation of the Raspberry Pi OS.

Ensure your Raspberry Pi is connected to the internet via Wi-Fi or Ethernet.

Install any required libraries or dependencies. For ultrasonic sensors, you may need the RPi.GPIO library for Python.

**Python Script for Sensor Data Collection:**

Write a Python script to interact with the ultrasonic sensors.

Use the RPi.GPIO library to control the GPIO pins.

Configure the Trig pin to send a brief high signal to trigger the ultrasonic sensor.

Measure the time it takes for the Echo pin to receive the reflected signal and calculate the distance to the object.

Collect sensor data, including distance measurements.

**PYTHON CODE**

import RPi.GPIO as GPIO

import time

# Set GPIO mode and pins

GPIO.setmode(GPIO.BCM)

TRIG = 23

ECHO = 24

# Set up the GPIO pins

GPIO.setup(TRIG, GPIO.OUT)

GPIO.setup(ECHO, GPIO.IN)

def get\_distance():

# Send a trigger signal

GPIO.output(TRIG, True)

time.sleep(0.00001)

GPIO.output(TRIG, False)

# Wait for the echo signal

while GPIO.input(ECHO) == 0:

pulse\_start = time.time()

while GPIO.input(ECHO) == 1:

pulse\_end = time.time()

pulse\_duration = pulse\_end - pulse\_start

# Calculate distance in centimeters

distance = (pulse\_duration \* 34300) / 2

return distance

try:

while True:

distance = get\_distance()

print(f"Distance: {distance} cm")

# Send the distance data to the cloud or mobile app server

**except KeyboardInterrupt**:

GPIO.cleanup()

**Sending Data to the Cloud or Mobile App:**

To send data to the cloud, you can use IoT platforms like AWS IoT, Azure IoT, or cloud-based databases like Firebase.

To send data to a mobile app, you can use APIs and a server to relay data to connected mobile devices.

**Data Analysis and Visualization:**

Process the collected data as needed.

You can create a web-based dashboard or mobile app to visualize parking space occupancy.

**Automation and Alerts**:

Implement automation rules and alerts based on occupancy data, for example, sending notifications when parking spaces are full or available.

**Testing and Deployment**:

Test your IoT sensor system in a real parking environment.

Deploy the system to the desired parking area.

Remember to secure your Raspberry Pi and IoT setup to protect it from unauthorized access, especially if it's deployed in a public area. Additionally, consider power management to ensure your system operates reliably for an extended period.