Phase 3: Development Part

Smart Parking

1. Hardware Setup :

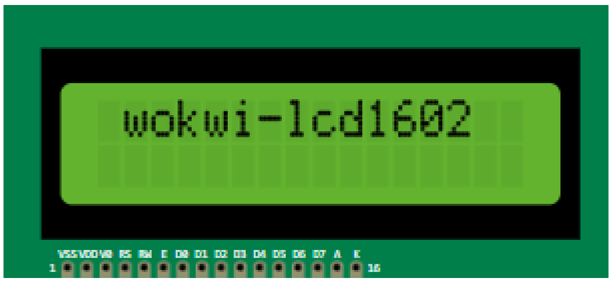
a. Ultrasonic Sensors: Connect the ultrasonic sensors to your Raspberry Pi. Typically, ultrasonic sensors have four pins: VCC, GND, TRIG, and ECHO. Connect them as follows:

- VCC to 5V on the Raspberry Pi.
- GND to GND on the Raspberry Pi.
- TRIG to a GPIO pin (e.g., GPIO17) on the Raspberry Pi.
- ECHO to another GPIO pin (e.g., GPIO18) on the Raspberry Pi.

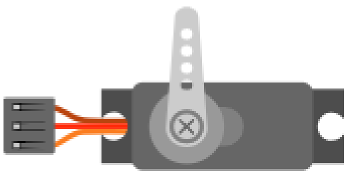
b. Raspberry Pi: Make sure your Raspberry Pi is set up with the necessary libraries and connected to the internet.

Required components :

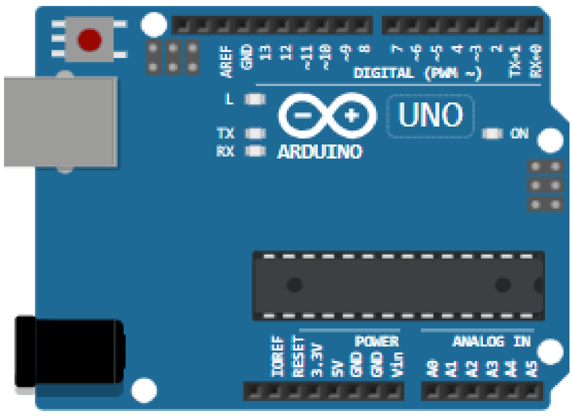
- wokwi-lcd1602 Reference



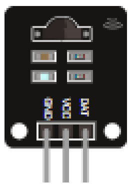
- wokwi-servo Reference



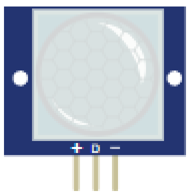
- wokwi-arduino-mega Reference



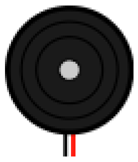
- wokwi-ir-receiver Reference



- wokwi-pir-motion-sensor Reference



- wokwi-buzzer Reference



2.Software Development :

python Script for Ultrasonic Sensors:

Create a Python script on the Raspberry Pi to interface with the ultrasonic sensors and detect parking space occupancy. Here's a sample script using the RPi.GPIO library for GPIO control and the `time` library for timing:

Python code

```
import RPi.GPIO as GPIO

import time


# Define GPIO pins for TRIG and ECHO

TRIG_PIN = 17

ECHO_PIN = 18


# Set the GPIO mode to BCM

GPIO.setmode(GPIO.BCM)


# Set TRIG as an output and ECHO as an input

GPIO.setup(TRIG_PIN, GPIO.OUT)

GPIO.setup(ECHO_PIN, GPIO.IN)


def measure_distance():
    # Send a trigger pulse
    GPIO.output(TRIG_PIN, True)
    time.sleep(0.00001)
    GPIO.output(TRIG_PIN, False)

    # Measure the time it takes for the echo pulse to return
    while GPIO.input(ECHO_PIN) == 0:
        pulse_start = time.time()

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

    pulse_duration = pulse_end - pulse_start

    # Calculate distance
    distance = (pulse_duration * 34300) / 2  # Speed of sound is 343
m/s

    return distance

try:
    while True:
        distance = measure_distance()

        # Define a threshold distance to determine occupancy (adjust
as needed)
        occupancy_threshold = 50  # Example threshold in centimeters
        if distance < occupancy_threshold:
            # Parking space is occupied, send this information to the cloud or server
            # You can use libraries like requests or MQTT to send data to the server
            # For simplicity, we'll print the occupancy status here
            print("Parking space is occupied")
        else:
            # Parking space is vacant
            print("Parking space is vacant")

        time.sleep(1)  # Adjust the interval as needed

except KeyboardInterrupt:
    GPIO.cleanup()
```

This script will continuously monitor the ultrasonic sensor and determine the parking space occupancy based on the threshold distance. You can replace the `print` statements with code to send occupancy information to your server or cloud.

3. Sending Data to Cloud or Mobile App Server :

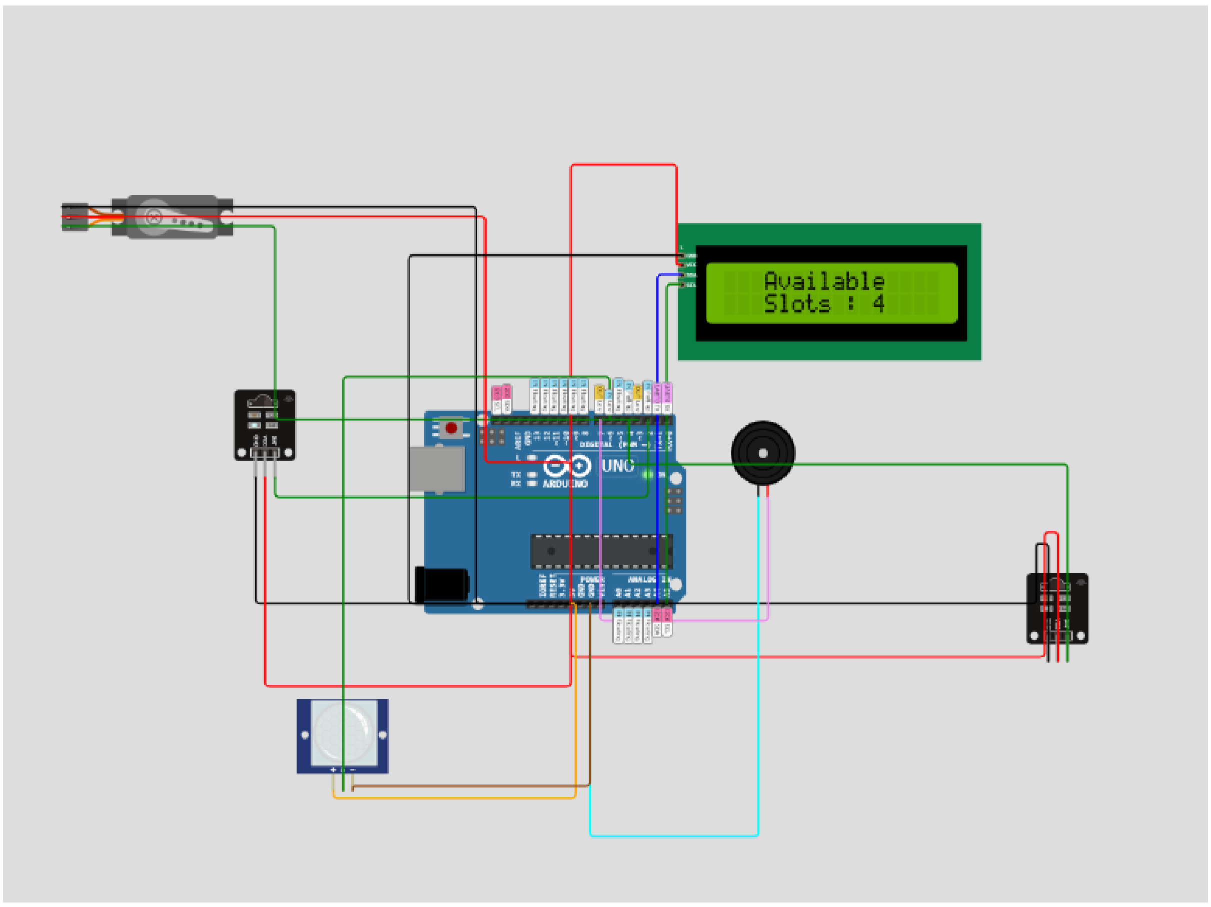
Implement the code for sending data to your cloud or mobile app server using the appropriate communication protocol (e.g., HTTP or MQTT). You'll need to set up the server or cloud infrastructure to receive and process this data.

With these components in place, your Raspberry Pi will monitor the ultrasonic sensor, detect parking space occupancy, and send the information to your server or cloud, making it a crucial part of your smart parking system.

Wokwi simulater link

: https://wokwi.com/projects/380183930050744321

Simulate our project on https://wokwi.com



Thanking You !!!