NOISE POLLUTION MONITORING PHASE\_3

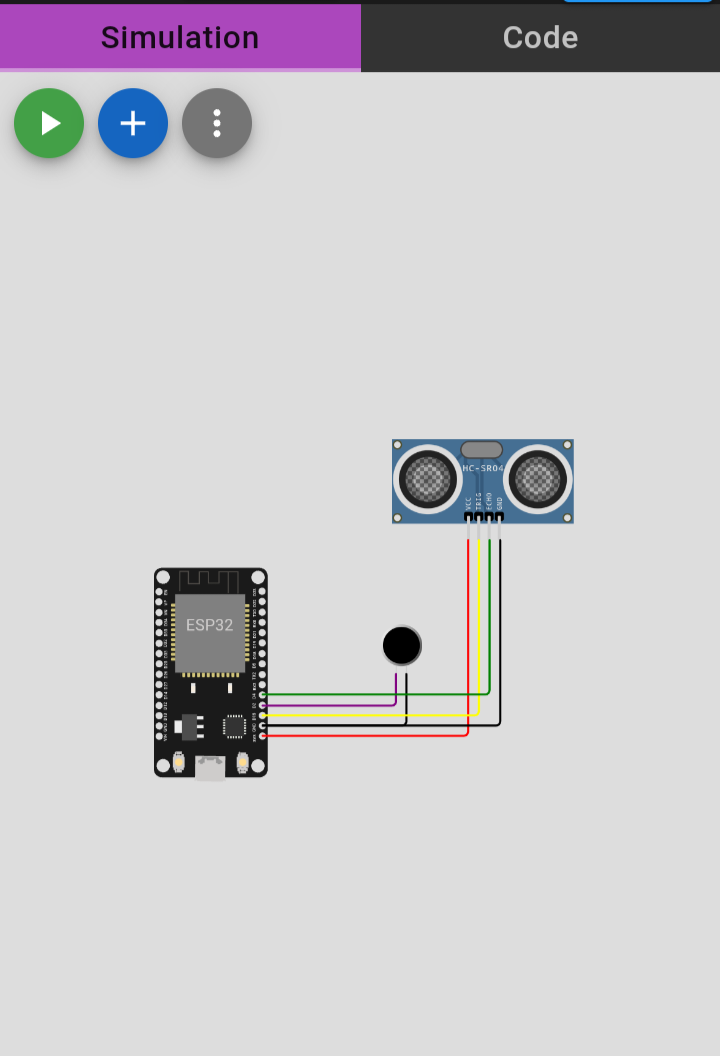
PHASE\_3 Development part 1

Noise pollution is a prevalent issue in urban environments, and with the help of Arduino and Python, we can develop a cost-effective solution to monitor noise levels and analyze noise pollution statistics.We'll utilize the Arduino Uno for data collection as it offers an easy interface for various sensors.To collect noise data, we'll integrate a sound sensor (e.g., KY-038) with the Arduino Uno.The Arduino Uno will be responsible for collecting noise data through the sound sensor.

Programming :

In the code, given below we are using MicroPython to read analog input from a pin using the machine.ADC module and printing the readings to the console.

the code is written in MicroPython, a version of Python designed for microcontroller and embedded systems programming.



CODE :

import machine

import time

import urequests

import ujson

import network

import math

wifi\_ssid ='Sheki'

wifi\_password = 'jesuslovesyou'

wifi = network.WLAN(network.STA\_IF)

wifi.active(True)

wifi.connect(wifi\_ssid, wifi\_password)

# Wait for Wi-Fi connection

while not wifi.isconnected():

pass

ultrasonic\_trig = machine.Pin(15, machine.Pin.OUT)

ultrasonic\_echo = machine.Pin(4, machine.Pin.IN)

microphone = machine.ADC(2)

calibration\_constant = 2.0

noise\_threshold = 50

firebase\_url = 'https://noise-pollution-monitor-default-rtdb.firebaseio.com/'

firebase\_secret ='XxdMfHMgQvXxQ2vjAOQoCrEAxEa6FX1ClIEHrChz'

def measure\_distance():

ultrasonic\_trig.value(1)

time.sleep\_us(10)

ultrasonic\_trig.value(0)

pulse\_time = machine.time\_pulse\_us(ultrasonic\_echo, 1, 30000)

distance\_cm = (pulse\_time / 2) / 29.1

return distance\_cm

def measure\_noise\_level():

noise\_level = microphone.read()

noise\_level\_db = 20 \* math.log10(noise\_level / calibration\_constant)

return noise\_level, noise\_level\_db

def send\_data\_to\_firebase(distance, noise\_level\_db):

data = {

"Distance": distance,

"NoiseLevelDB": noise\_level\_db

}

url = f'{firebase\_url}/sensor\_data.json?auth={firebase\_secret}'

try:

response = urequests.patch(url, json=data)

if response.status\_code == 200:

print("Data sent to Firebase")

else:

print(f"Failed to send data to Firebase. Status code: {response.status\_code}")

except Exception as e:

print(f"Error sending data to Firebase: {str(e)}")

try:

while True:

distance = measure\_distance()

noise\_level, noise\_level\_db = measure\_noise\_level()

print("Distance: {} cm, Noise Level: {:.2f} dB".format(distance, noise\_level\_db))

if noise\_level\_db > noise\_threshold:

print("Warning: Noise pollution exceeds threshold!")

send\_data\_to\_firebase(distance, noise\_level\_db)

time.sleep(1)

except KeyboardInterrupt:

print("Monitoring stopped")

EXPLANATION OF THE ABOVE CODE:

The code above is a Python code to monitor noise pollution using an ultrasonic sensor and a microphone. It also sends the measured data to Firebase Realtime Database.

First few lines import the necessary libraries, such as `machine`, `time`, `urequests`, `ujson`, `network`, and `math`.

The following lines define the Wi-Fi credentials and connect to the Wi-Fi network.

Later the lines define the pins for the ultrasonic sensor and the microphone.

After that lines define the calibration constant and the noise threshold.

Then this function measures the distance using the ultrasonic sensor. It triggers the sensor and measures the pulse width of the echo signal. The distance is then calculated using the following formula:

```

distance\_cm = (pulse\_time / 2) / 29.1

```

Later the function measures the noise level using the microphone. It reads the analog value from the microphone and converts it to a decibel value using the following formula:

```

noise\_level\_db = 20 \* math.log10(noise\_level / calibration\_constant)

```

Then the function sends the measured data to Firebase Realtime Database. It uses the `urequests` library to make a PATCH request to the Firebase URL with the distance and noise level data.

After that comes the main loop of the program. It continuously measures the distance and noise level, and sends the data to Firebase. If the noise level exceeds the threshold, it prints a warning message.

Short procedure:

\* Install the necessary Python libraries.

\* Replace the `wifi\_ssid` and `wifi\_password` variables with your own Wi-Fi credentials.

\* Replace the `firebase\_url` and `firebase\_secret` variables with your own Firebase URL and secret.

\* Connect the ultrasonic sensor and the microphone to the appropriate pins on your microcontroller.

After these steps we can run the code and it will start monitoring noise pollution. The measured data will be sent to Firebase Realtime Database, where you can view and analyze it.

OUTPUT LINK :

https://wokwi.com/projects/379675165725791233

