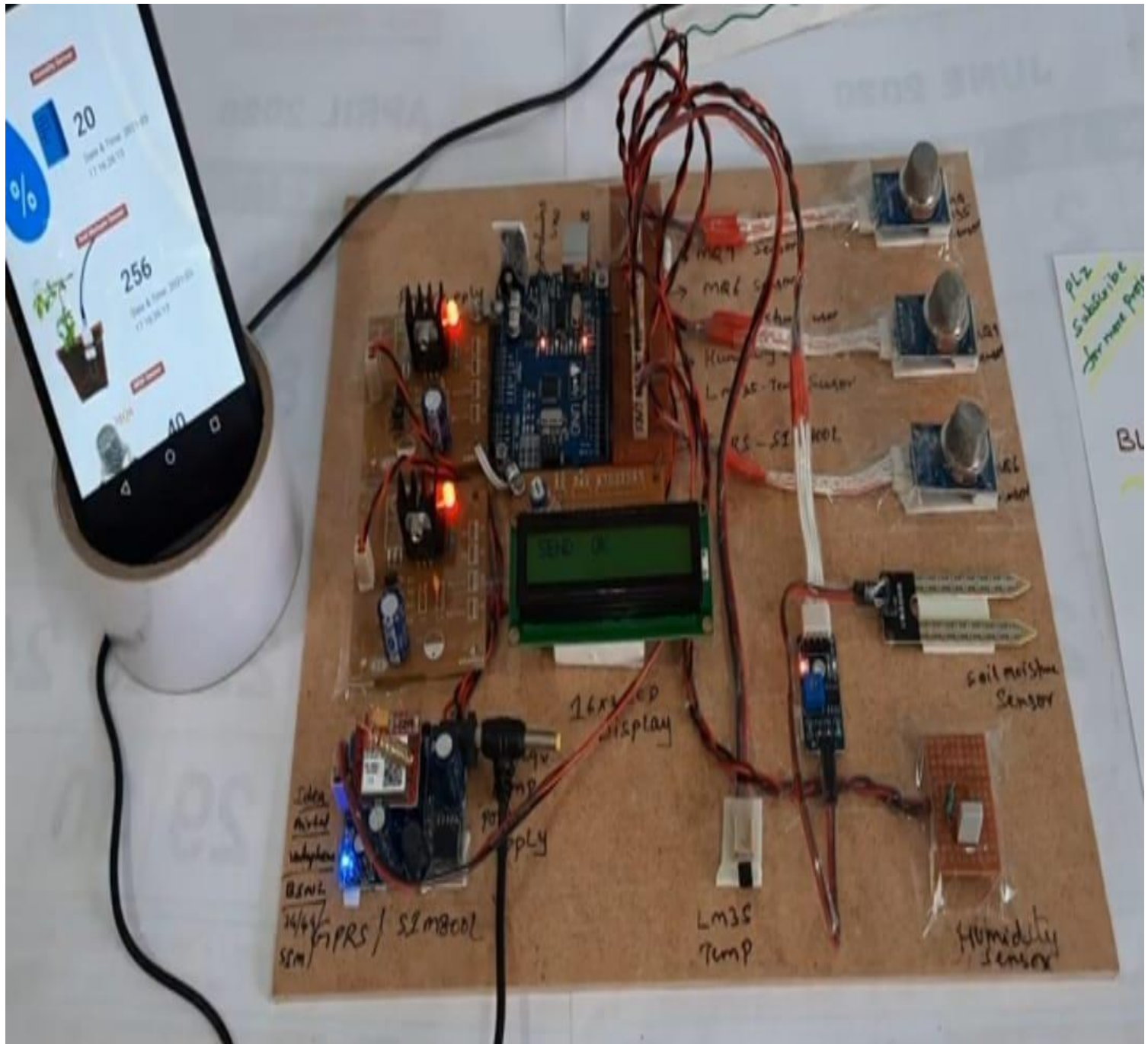


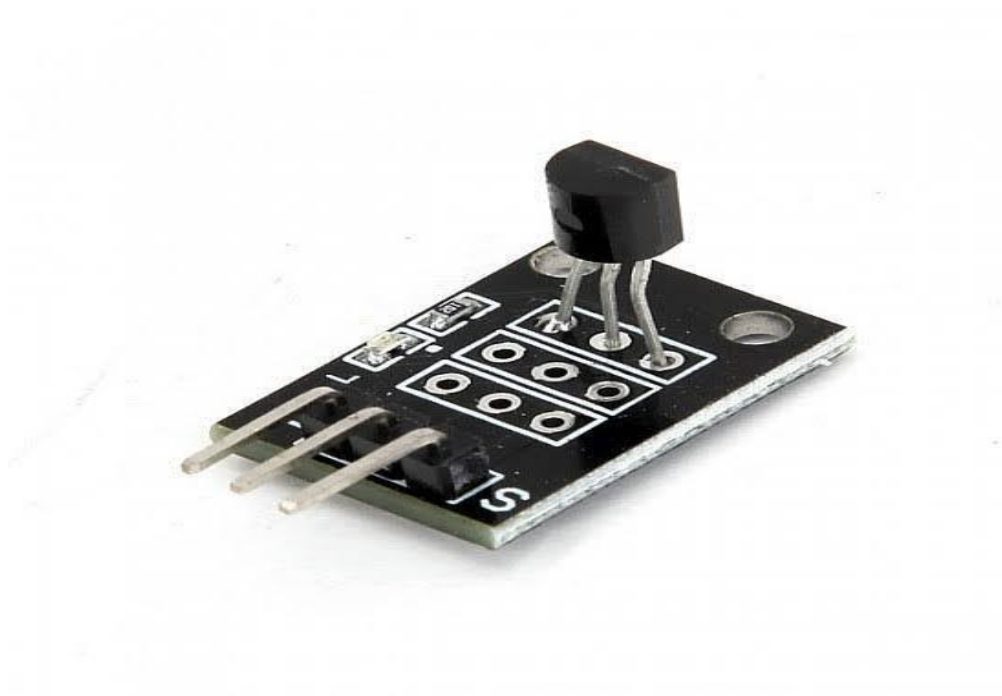
Environmental monitoring-phase 3

PROJECT DESIGN:



COMPONENTS :

■TEMPERATURE SENSOR :



▪ **CAPACITIVE HUMIDITY SENSORS :**



▪ **MQ 135 SENSOR:**



▪MQ 6 GAS SENSOR :



▪MQ 9 GAS SENSOR :



CODE:

Adafruit IO Environmental Monitor for Feather or Raspberry Pi -

an internet-enabled environmental monitor

*# Import standard python modules import
time*

import Adafruit Blinka

import board import

busio

import CircuitPython sensor libraries

import adafruit_sgp30 import

adafruit_veml6070

from adafruit_bme280 import basic as adafruit_bme280

import Adafruit IO REST client

from Adafruit_IO import Client, Feed, RequestError

loop timeout, in seconds.

LOOP_DELAY = 10

Set to your Adafruit IO key.

Remember, your key is a secret,

so make sure not to publish it when you publish this code!

ADAFRUIT_IO_KEY = 'YOUR_AIO_KEY'

Set to your Adafruit IO username.

(go to <https://accounts.adafruit.com> to find your username)

```
ADAFRUIT_IO_USERNAME = 'YOUR_AIO_USERNAME'
```

```
# Create an instance of the REST client
```

```
aio = Client(ADAFRUIT_IO_USERNAME, ADAFRUIT_IO_KEY)
```

```
try: # if we already have the feeds, assign them.
```

```
    tvoc_feed = aio.feeds('tvoc')
```

```
    eCO2_feed = aio.feeds('eco2')    uv_feed
```

```
    = aio.feeds('uv')
```

```
    temperature_feed = aio.feeds('temperature')
```

```
    humidity_feed = aio.feeds('humidity')    pressure_feed
```

```
    = aio.feeds('pressure')    altitude_feed =
```

```
    aio.feeds('altitude') except RequestError: # if we don't,
```

```
    create and assign them.
```

```
    tvoc_feed = aio.create_feed(Feed(name='tvoc'))
```

```
    eCO2_feed = aio.create_feed(Feed(name='eco2'))    uv_feed
```

```
    = aio.create_feed(Feed(name='uv'))    temperature_feed =
```

```
    aio.create_feed(Feed(name='temperature'))
```

```
    humidity_feed = aio.create_feed(Feed(name='humidity'))
```

```
    pressure_feed = aio.create_feed(Feed(name='pressure'))
```

```
    altitude_feed = aio.create_feed(Feed(name='altitude'))
```

```

# Create busio I2C
i2c = busio.I2C(board.SCL, board.SDA) #
Create VEML6070 object.
uv = adafruit_veml6070.VEML6070(i2c) # Create
BME280 object. bme280 =
adafruit_bme280.Adafruit_BME280_I2C(i2c)
bme280.sea_level_pressure = 1013.25 # Create SGP30
object using I2C.
sgp30 = adafruit_sgp30.Adafruit_SGP30(i2c) sgp30.iaq_init()
sgp30.set_iaq_baseline(0x8973, 0x8aae)

# Sample VEML6070 def
sample_VEML(): for _
in range(10):
uv_raw = uv.uv_raw
return uv_raw

while True:
    print('Reading sensors...')
# Read SGP30.
    eCO2_data = sgp30.eCO2
    tvoc_data = sgp30.TVOC

```


Read VEMML6070.

uv_data = sample_VEMML()

Read BME280.

temp_data = bme280.temperature #

convert temperature (C->F) temp_data

*= int(temp_data) * 1.8 + 32*

humid_data = bme280.humidity

pressure_data = bme280.pressure

alt_data = bme280.altitude

print('sending data to adafruit io...')

Send SGP30 Data to Adafruit IO.

print('eCO2:', eCO2_data)

aio.send(eCO2_feed.key, eCO2_data)

print('tvoc:', tvoc_data)

aio.send(tvoc_feed.key, tvoc_data)

time.sleep(2)

Send VEMML6070 Data to Adafruit IO.

```
print('UV Level: ', uv_data)  
aio.send(uv_feed.key, uv_data)    time.sleep(2)  
# Send BME280 Data to Adafruit IO.  
print('Temperature: %0.1f C' % temp_data)  
aio.send(temperature_feed.key, temp_data)  
print("Humidity: %0.1f %" % humid_data)  
aio.send(humidity_feed.key, int(humid_data))  
time.sleep(2)  
print("Pressure: %0.1f hPa" % pressure_data)  
aio.send(pressure_feed.key, int(pressure_data))  
print("Altitude = %0.2f meters" % alt_data)  
aio.send(altitude_feed.key, int(alt_data))    # avoid timeout  
from adafruit io    time.sleep(LOOP_DELAY * 60)
```