**The IoT agricultural soil nutrient and weather station system**

**英文版**

**Introduction**

In order to monitor the growth environment of **crops outdoors and greenhouses**, I have designed an IoT agricultural monitoring system. The system utilizes Unihikers as the main board, combined with **RS485 soil sensors**, temperature and humidity sensors, CO2 sensors, UV light sensors, etc., collecting **soil nutrients at different nodes, including soil temperature, humidity, PH, and NPK**. At the same time, the system collects data on air temperature, humidity, CO2, and UV light. Additionally, cameras are also equipped for taking photos of crop growth.

Unihikers **wirelessly** upload the data to the local server of Lattepanda through **WiFi**. We can view real-time data using the Flask web. Moreover, the system not only records environmental data in greenhouses but also includes an **Unihiker outdoor weather station**, which monitors essential environmental data outdoors in real-time, including wind direction, wind speed, temperature, humidity, pressure, and rainfall.

All of the data uploaded to the cloud can support real-time analysis of soil nutrients, UV light, and other environmental data. It can also provide data labeling and learning foundations for future AI development. The program is written in **Python**, with functions such as automatically running script on startup and automatic network reconnection. The Lattepanda server can display the online status and data of different nodes.

|  |  |
| --- | --- |
| Monitoring lettuce in the greenhouse | weather station |

|  |  |
| --- | --- |
| RS485 sensor monitors soil data | A monitoring node |

|  |  |
| --- | --- |
| A monitoring node for greenhouse | Remote viewing data on LattePanda |

**Hardware list:**

1. 土壤氮磷钾传感器 x5
2. Lattepanda 3 delta x1
3. Unihiker x6
4. 二氧化碳传感器 SEN0536 x3
5. 环境光传感器 SEN0540 x3
6. 气象站 SEN0186 x1
7. 485转UART模块 DFR0845 x5
8. 温湿度传感器 SEN0334 x1
9. Unihiker扩展板 MBT0008 x6
10. Power Adapter FIT0639 x6
11. SCI x 6
12. USB camera x5
13. Router x1
14. Waterproof enclosure x6

**System functions:**

- Real-time monitoring of major environmental indicators in the "one-meter field"(soil parameters: soil temperature and humidity, soil pH, NPK; environmental data: CO2, temperature and humidity, UV light).

- Crop monitoring: automatic image capture and save, automatically displaying the latest crop images.

- Weather station monitoring: real-time monitoring of outside weather conditions (wind direction, wind speed, temperature, air pressure, and rainfall).

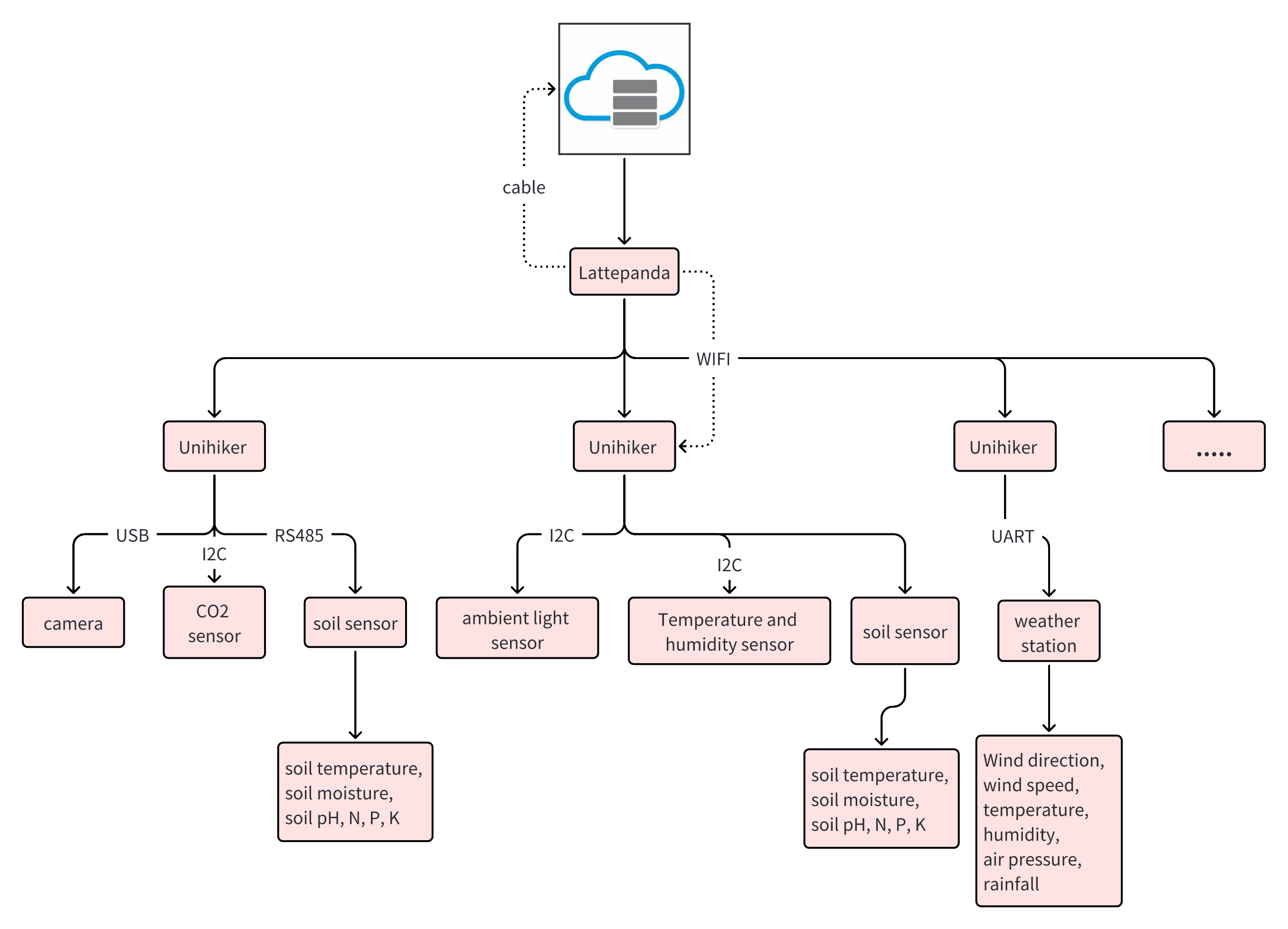
- Displaying the data in the cloud.

- Online warning for offline node.



**点击图片可查看完整电子表格**

**Solution diagram**



**Product specifications**

4.1 **土壤传感器：**

4.2 **Lattepanda 3 delta：**

* CPU: Intel® Celeron® N5105
* Core: 2.0~2.9GHz Quad-Core, Four-Thread
* Graphics: Intel® UHD Graphics (Frequency: 450 – 800MHz)
* Memory: LPDDR4 8GB 2933MHz
* Storage: 64GB eMMC V5.1
* Expansion Slots:  
   1x M.2 M Key, PCIe 3.0 2x, Supports NVMe SSD  
   1x M.2 B Key, USB 2.0, USB 3.0, SIM, SATA(Option support PCIe 3.0 1x), Supports 4G & 5G module
* Connectivity:  
   1x Wi-Fi 6 @ 2.4GHz &5 GHz(160MHz)  
   1x Bluetooth 5.2  
   1x Intel Gigabit Ethernet
* USB Ports:  
   2x USB 3.2 Gen1 Type A  
   1x USB 3.2 Gen2 Type A  
   1x USB Type C, Supports PD, DP, USB 2.0  
   1x USB 2.0 Pin Header
* Display:  
   1x HDMI 2.0b: Up to 4096x2160 @ 60Hz HDR  
   1x DP 1.4: Up to 4096x2160 @ 60Hz HDR  
   1x eDP: Extendable Touch Displays up to 1920\*1080
* TPM: built-in TPM(2.0)
* Co-processor: Arduino Leonardo ATMEGA32U4
* Audio: Microphone + Headphone Combo Connector
* GPIO & Other Features:  
   12x Analog Inputs\*  
   Up to 23x Digital Input/Output (7 PWM)\*  
   1x UART  
   1x I2C  
   1x SPI  
   1x Audio Connector  
   1x 4-Pin RS232 Header  
   1x Fan Port (4 Pin 1.25mm PWM 5V）  
   1x 4-Pin Header (Power and Switch)
* OS Support: Windows 10 & Windows 11 & Linux
* Dimension: 125 x 78 x 16mm
* Operating Temperature: 0°C~75°C

4.3 **Unihiker:**

* CPU: Quad-Core ARM Cortex-A35, up to 1.2GHz
* RAM: 512MB
* Flash: 16GB
* OS: Debian
* Wi-Fi: 2.4G
* BT: Bluetooth 4.0
* Screen: 2.8inch, 240×320, Touch Screen
* MCU: GD32VF103
* Sensor: Button, Microphone, Light Sensor, Accelerometer Sensor, Gyroscope Sensor
* Actuator: Led, Buzzer
* Port: USB Type-C, USB-A, Gravity 3pin&4pin port, Edge connector
* Power: 5V 2A for USB Type-C
* Size: 51.6mmx83mmx13mm

**Power consumption estimate**



**点击图片可查看完整电子表格**

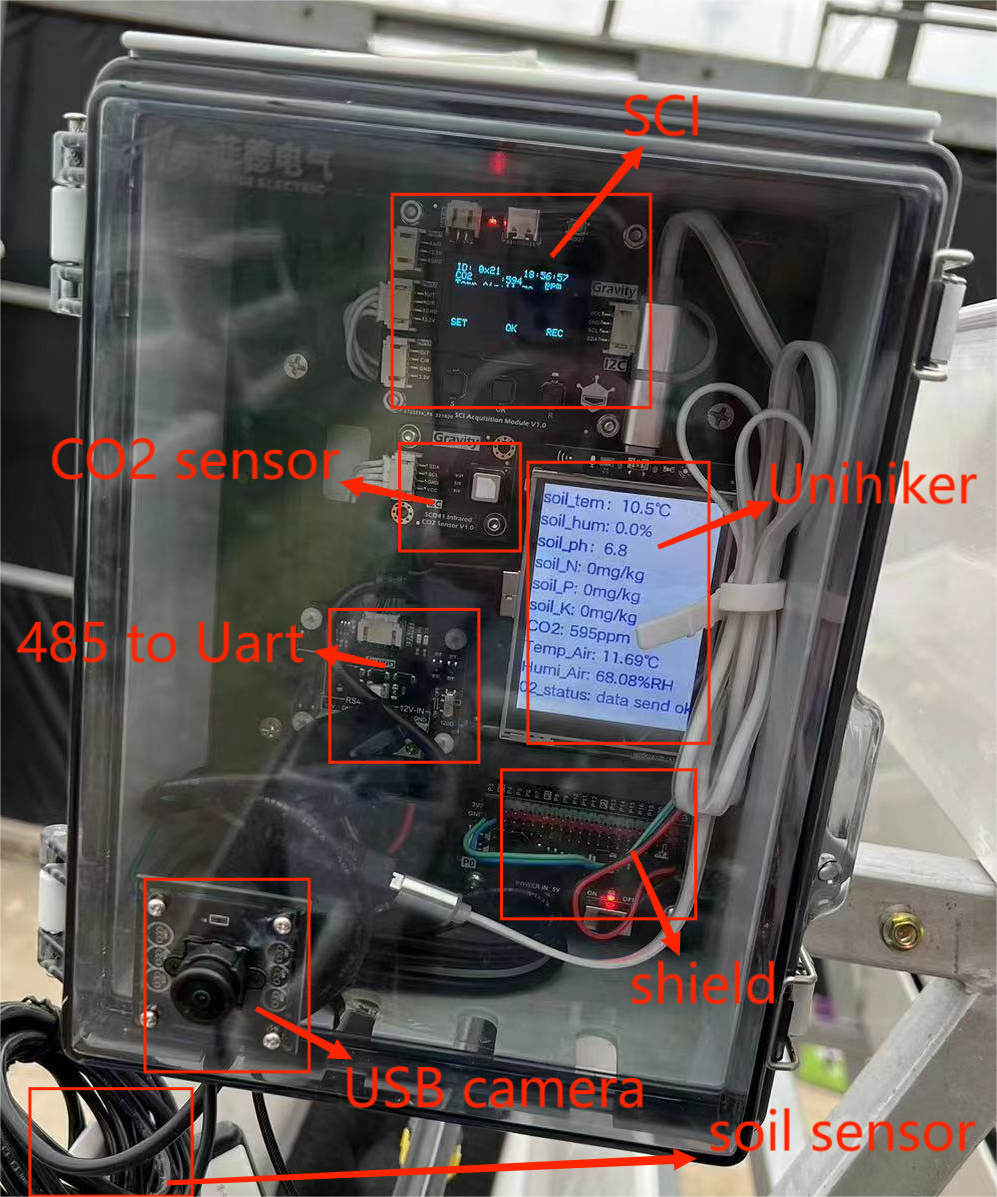
Total power consumption is about 2.3kwh per day, 69kwh per month.

**Data usage estimate**

Assuming data is uploaded every 5 minutes from each node (including image data).  
The data size for each node is 1KB and each image is 2 KB.  
0.3 x 288 times = 86.4 MB

**Connection：**

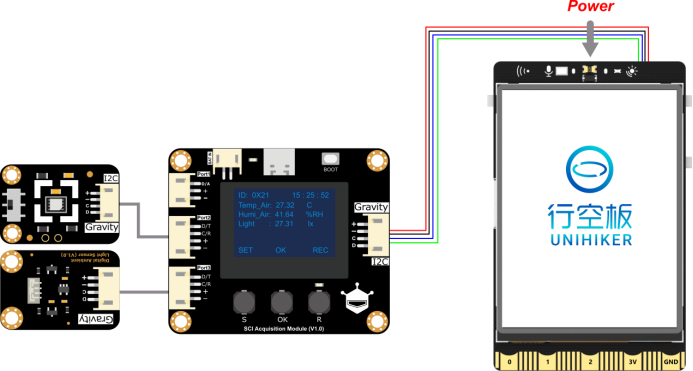
1. **Taking a node as an example, the wiring for nodes 1-5 is the same.**



The connection between the RS485 soil sensor, 485 to UART module (DFR0845), and Uihiker shield(MBT0008) is as follows:

|  |  |  |
| --- | --- | --- |
| **485 to UART module** | **Uihiker shield** | **RS485 soil sensor** |
| UART-TX | P0 |  |
| UART-RX | P3 |  |
| UART - | - |  |
| UART + | + |  |
| RS485-12V |  | VCC |
| RS485-GND |  | GND |
| RS485-A |  | A |
| RS485-B |  | B |

The connection between SCI, sensors (SEN0536, SEN0540, SEN0334) and Uihiker is as follows:



1. The connection between the weather station (SEN0186) and Uihiker shield (MBT0008) is as follows:

|  |  |
| --- | --- |
| **the weather station** | **shield** |
| TX | P0 |
| GND | GND |
| 5V | 5V |

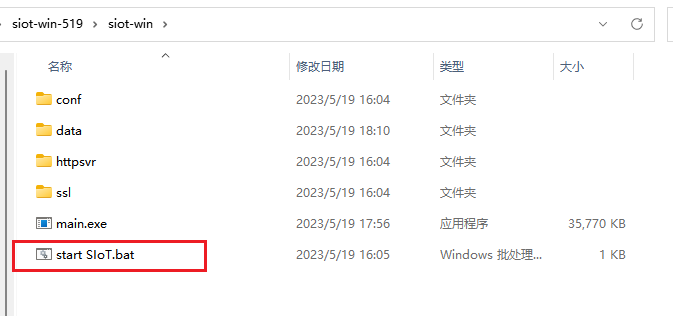
1. **power：**

Use FIT0639 and the power adapter module to supply power to Uihiker and shield (MBT0008).

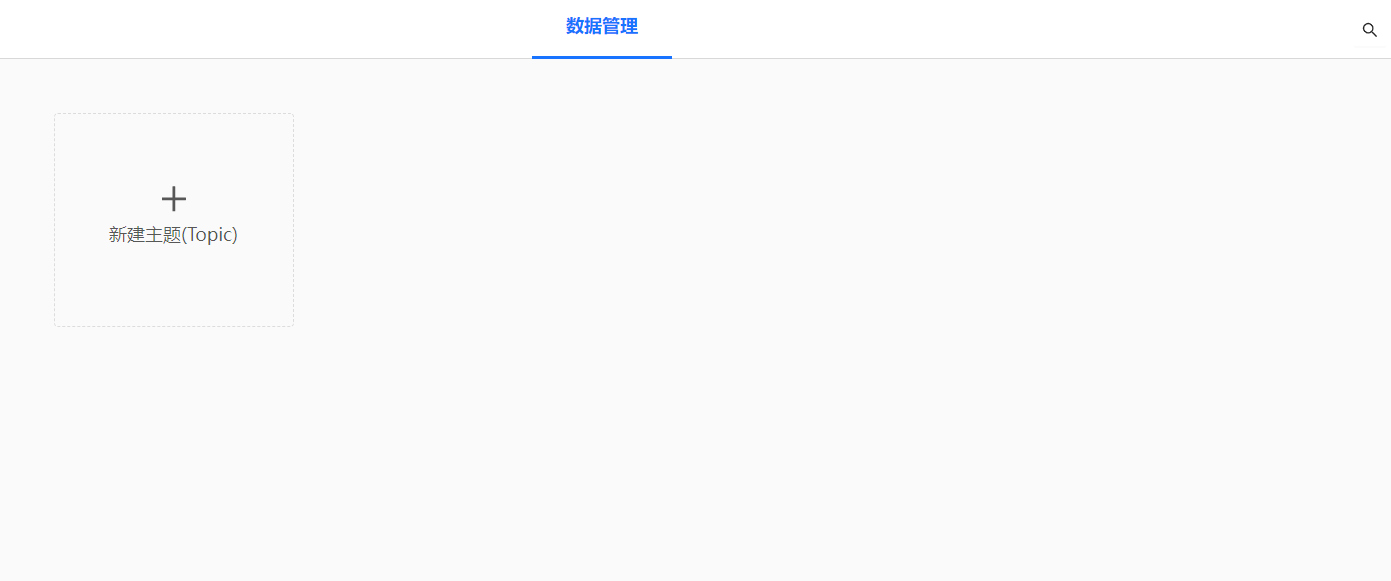
**Build the environment on the LP server:**

**Siot V2**

1. Download the Windows version of SIoT V2 and unzip it. Double-click start SIoT.bat to start SIoT. After startup, a command window will start the server.

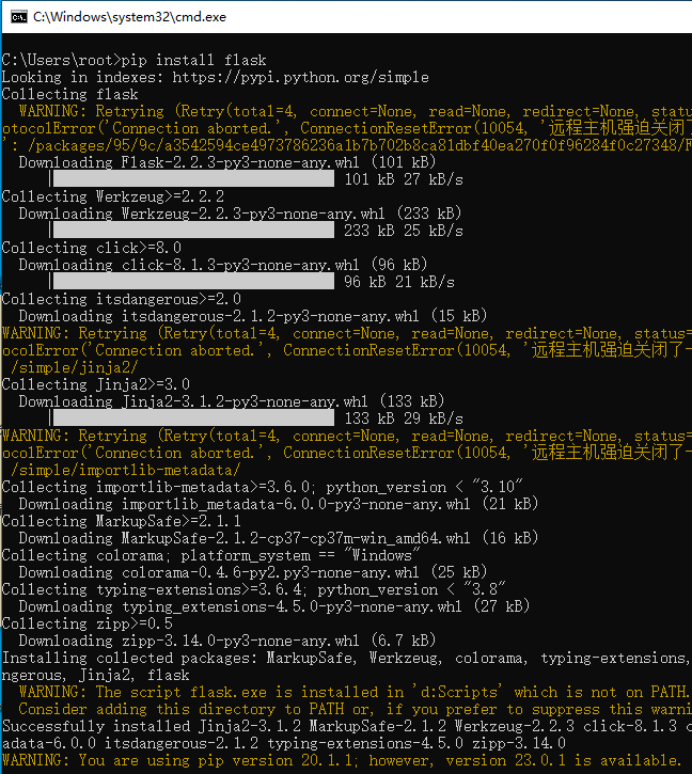


1. Enter 127.0.0.1:8080 in the browser to open the web. The login account is "*siot"* and the password is "*dfrobot"*. After opening, you can create a new topic or view data.



**Flask**

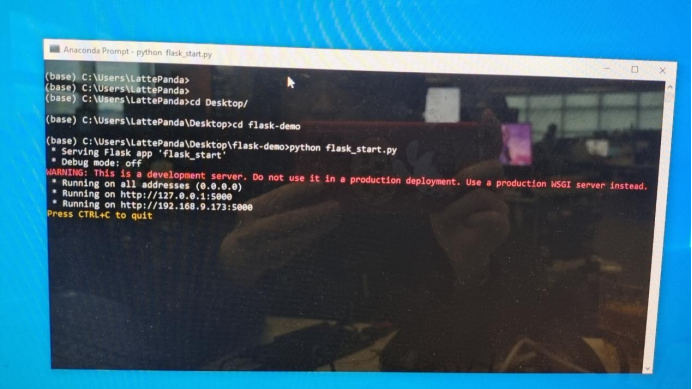
1. Install python3.7: <https://www.python.org/downloads/windows/>
2. Install the flask



1. Download the code (see Section 6), and run flask-start.py

|  |
| --- |
| Bash cd flask-demo |

|  |
| --- |
| Bash python flask-start.py |



1. Enter the URL to access http://127.0.0.1:5000/index (you need to start Siot V2 before you can use this Flask Web). On this web page, you can select to view the sensor data of the node through the button above.



**Configure the router**

Please log in to the router webpage on Lattepanda and set up the fixed IP addresses for Lattepanda and the six Unihikers.

|  |  |
| --- | --- |
| **Device** | **IP** |
| Lattepanda | 10.168.1.100 |
| 1 | 10.168.1.114 |
| 2 | 10.168.1.122 |
| 3 | 10.168.1.115 |
| 4 | 10.168.1.118 |
| 5 | 10.168.1.117 |
| 6 Weather Station | 10.168.1.112 |

**Update Siot V2 through the command line on Unihikers:**

|  |
| --- |
| Bash pip uninstall siot pip install siot |

**Code:**

1. **Unihiker node 1**

（请开启Unihiker开机自启动设置）

|  |
| --- |
| Python # -\*- coding: utf-8 -\*- import time from dfrobot\_rp2040\_sci import \* from pinpong.board import Board, UART import serial  import time import siot import os from unihiker import GUI import requests import base64  import cv2 #Unihiker Initialize Board("").begin()   # SCI Initialize SCI1 = DFRobot\_RP2040\_SCI\_IIC(addr=0x21) while SCI1.begin() != 0:  print("Initialization Sensor Universal Adapter Board failed.")  time.sleep(1) print("Initialization Sensor Universal Adapter Board done.")  #UART P0-RX P3-TX uart1 = UART()  #Initialize UART  uart1.init(baud\_rate = 9600, bits=8, parity=0, stop = 1)   #Unihiker GUI Initialize u\_gui=GUI() #GUI setting soil\_tem\_text=u\_gui.draw\_text(text="soil temperature：NAN",x=0,y=0,font\_size=16, color="#0000FF") soil\_hum\_text=u\_gui.draw\_text (text="soil humidity：NAN",x=0,y=30,font\_size=16, color="#0000FF") soil\_ph\_text=u\_gui.draw\_text(text="soil ph：NAN",x=0,y=60,font\_size=16, color="#0000FF") soil\_N\_text=u\_gui.draw\_text(text="soil N：NAN",x=0,y=90,font\_size=16, color="#0000FF") soil\_P\_text=u\_gui.draw\_text(text="soil P：NAN",x=0,y=120,font\_size=16, color="#0000FF") soil\_K\_text=u\_gui.draw\_text(text="soil K：NAN",x=0,y=150,font\_size=16, color="#0000FF") CO2\_text = u\_gui.draw\_text(text="CO2：NAN",x=0,y=180,font\_size=16, color="#0000FF") status\_text = u\_gui.draw\_text(text="01\_status：NAN",x=0,y=270,font\_size=16, color="#0000FF") air\_tem\_text=u\_gui.draw\_text(text="air temperature：NAN",x=0,y=210,font\_size=16, color="#0000FF") air\_hum\_text=u\_gui.draw\_text(text="air humidity：NAN",x=0,y=240,font\_size=16, color="#0000FF") #lux\_text=u\_gui.draw\_text(text="light lux：NAN",x=0,y=270,font\_size=16, color="#0000FF")  #command sent to soil sensor buf = [0x02, 0x03, 0x00, 0x00, 0x00, 0x0A, 0xC5,0xFE]  #sensor default return command  #['04', '03', '14', '00', 'e7', '00', '00', '00', '00', '00', '28', '00', '00', '00', '00', '00', '00', '00', '00', '00', '00', '25', '80', '25', 'b2']  #Calculating CRC checksum def calc\_crc(string):  #print(string)  #data = bytearray.fromhex(string)  data = ['{:02x}'.format(i) for i in string]  #print(data)  data = " ".join(data)  data = data.replace('0x','')  global data2  data2 = data  print(data2)  data = bytearray.fromhex(data)    crc = 0xFFFF  for pos in data:  crc ^= pos  for i in range(8):  if ((crc & 1) != 0):  crc >>= 1  crc ^= 0xA001  else:  crc >>= 1  return hex(((crc & 0xff) << 8) + (crc >> 8))   def send\_photos():  photos\_path = '/root/photos'  photos\_path\_list = os.listdir(photos\_path)  photos\_path\_list.sort(reverse=False)  photos\_quan = len(photos\_path\_list)    photos\_count = photos\_quan+1  print("count:"+str(photos\_count))  #USB camera setting  cap = cv2.VideoCapture(0)   cap.set(cv2.CAP\_PROP\_FRAME\_WIDTH, 320)   cap.set(cv2.CAP\_PROP\_FRAME\_HEIGHT, 240)   cap.set(cv2.CAP\_PROP\_BUFFERSIZE, 1)     ret, frame = cap.read()  if ret == True:  #save photos on Unihiker  cv2.imwrite(photos\_path+'/Frame'+ str(photos\_count) +'.jpg', frame)  print("save photo!")  cap.release()   with open(photos\_path+'/Frame'+ str(photos\_count) +'.jpg',"rb") as f:   # b64encode是编码，b64decode是解码   data = base64.b64encode(f.read())  src = "data:image/{ext};base64,{data}".format(ext='jpg', data=str(data))  #print(src)  #print(len(src))    # Send base64 buf of the photos to the server  siot.publish\_save(topic="siot/node1/image", data=src)  #siot.publish\_save(topic="siot/image", data=src)  print("photos send ok")  f.close()  status\_text.config(text="04\_status: photo send ok",x=0,y=290)  photos\_count = int(photos\_count) + 1  count = 0 while True:  #refresh GUI  air\_tem\_value=SCI1.get\_value0("Temp\_Air")  air\_tem\_t = "Temp\_Air: "+str(air\_tem\_value)+"℃"  air\_tem\_text.config(text=air\_tem\_t,x=0,y=210)  air\_hum\_value=SCI1.get\_value0("Humi\_Air")  air\_hum\_\_t = "Humi\_Air: "+str(air\_hum\_value)+"%RH"  air\_hum\_text.config(text=air\_hum\_\_t,x=0,y=240)   print("-----------write buf to soil sensor-----------")  uart1.write(buf)  time.sleep(1)  count=0  #"If there is data on the serial port"  while uart1.any()==0:  print("any:"+str(count))  count=count+1  if count>10:  break  time.sleep(0.1)    while uart1.any()>0:  print("while2:"+str(uart1.any()))  #print(uart1.read(uart1.any()))  time.sleep(0.01)  # This sensor's address is 0x02, pls review the wiki for more details.  if uart1.read(1)[0] == 0x02:  print("11")  time.sleep(0.01)   #This sensor's default code is 0x03, pls review the wiki for more details.  if uart1.read(1)[0] == 0x03:   time.sleep(0.01)   print("while2:"+str(uart1.any()))  data = uart1.read(23)   data.insert(0,0x02)  data.insert(1,0x03)  #print(data)  crc = calc\_crc((data))  #print(data[11],data(12))  print("crc="+str(crc))    if crc == '0x0':  print(data2)  data3 = data2.split()  #print(str(data3[10])+str(data3[11]))  #soil data read from soil sensor  soil\_tem = int(str(data3[3])+str(data3[4]),16)/10  soil\_ph = int(str(data3[9])+str(data3[10]),16)/10  soil\_hum = int(str(data3[5])+str(data3[6]),16)/10  soil\_N = int(str(data3[11])+str(data3[12]),16)  soil\_P = int(str(data3[13])+str(data3[14]),16)  soil\_K = int(str(data3[15])+str(data3[16]),16)    #refresh GUI   soil\_ph\_t = "soil\_ph："+ str(soil\_ph)  soil\_ph\_text.config(text= soil\_ph\_t ,x=0,y=60)  soil\_hum\_t = "soil\_hum: "+str(soil\_hum)+"%"  soil\_hum\_text.config(text= soil\_hum\_t ,x=0,y=30)  soil\_tem\_t = "soil\_tem："+str(soil\_tem)+"℃"  soil\_tem\_text.config(text= soil\_tem\_t ,x=0,y=0)  soil\_N\_t = "soil\_N: "+str(soil\_N)+"mg/kg"  soil\_N\_text.config(text= soil\_N\_t ,x=0,y=90)  soil\_P\_t = "soil\_P: "+str(soil\_P)+"mg/kg"  soil\_P\_text.config(text= soil\_P\_t ,x=0,y=120)  soil\_K\_t = "soil\_K: "+str(soil\_K)+"mg/kg"  soil\_K\_text.config(text= soil\_K\_t ,x=0,y=150)   try :  #get wifi's status  my\_variable = requests.get("http://10.1.2.3/wifi/status")  print(my\_variable.text)  status = my\_variable.text.split('"')[11]  print("wifi: "+status)  status\_text.config(text="01\_wifi:"+status,x=0,y=270)    #siot init and connect to the LP server. The server's IP is 10.168.1.100.   #pls note that each unihiker's client\_id is different.   siot.init(client\_id="unihiker01",server="10.168.1.100",port=1883,user="siot",password="dfrobot")  siot.connect()  siot.loop()  #siot subscribe the topics  siot.getsubscribe(topic="siot/node1/soiltemperture")  siot.getsubscribe(topic="siot/node1/soilhum")  siot.getsubscribe(topic="siot/node1/soilpH")  siot.getsubscribe(topic="siot/node1/soilN")  siot.getsubscribe(topic="siot/node1/soilP")  siot.getsubscribe(topic="siot/node1/soilK")  #siot.getsubscribe(topic="siot/node1/二氧化碳")  siot.getsubscribe(topic="siot/node1/airtemper")  siot.getsubscribe(topic="siot/node1/airhum")  #siot.getsubscribe(topic="siot/image")  siot.getsubscribe(topic="siot/node1/image")   siot.publish\_save(topic="siot/node1/soiltemperture", data=soil\_tem)  siot.publish\_save(topic="siot/node1/soilpH", data=soil\_ph)  siot.publish\_save(topic="siot/node1/soilhum", data=soil\_hum)  siot.publish\_save(topic="siot/node1/soilN", data=soil\_N)  siot.publish\_save(topic="siot/node1/soilP", data=soil\_P)  siot.publish\_save(topic="siot/node1/soilK", data=soil\_K)  siot.publish\_save(topic="siot/node1/airtemper", data=air\_tem\_value)  siot.publish\_save(topic="siot/node1/airhum", data=air\_hum\_value)  print("send ok")  siot.stop()    status\_text.config(text="01\_status: data send ok",x=0,y=270)  time.sleep(3600)  except :   print("wifi reconnect！")  status\_text.config(text="wifi reconnect！",x=0,y=270)  #"ssid=dfrobot&password=dfrobot2017" wifi ssid and wifi password  my\_variable = requests.get("http://10.1.2.3/wifi/connect?ssid=dfrobot&password=dfrobot2017")  print(my\_variable.text)  time.sleep(60)  my\_variable = requests.get("http://10.1.2.3/wifi/status")  print(my\_variable.text)  status = my\_variable.text.split('"')[11]  print(status)  status\_text.config(text="01\_wifi:"+status,x=0,y=270) |

2. **Unihiker weatherstation**

|  |
| --- |
| Python # -\*- coding: utf-8 -\*- import time from pinpong.board import Board, UART import siot import os  #Unihiker Initialize Board("UNIHIKER").begin()   #UART P0-RX P3-TX ,pls note that weather station's baud\_rate is 9600 uart1 = UART()  uart1.init(baud\_rate = 9600, bits=8, parity=0, stop = 1)  #siot init and connect to the LP server. The server's IP is 10.168.1.100.  #pls note that each unihiker's client\_id is different.  siot.init(client\_id="weatherstation",server="10.168.1.100",port=1883,user="siot",password="dfrobot") while True:  databuffer = ""  #print(len("c000s000g000t082r000p000h48b10022\*3C"))default buffer  #read uart data  buf = uart1.readline()   if buf is None:  print("recv None")  else:  for i in buf:  #print(i)  a = chr(i)  databuffer = databuffer + a   length = len(databuffer)  print(databuffer)  if length == 38:   print("databuffer:",databuffer)    '''get wind direction'''  try:  WindDirection = int(databuffer[1:4])  except:  WindDirection = 0    if 0 <= WindDirection and WindDirection < 22.5 or 337.5<=WindDirection and WindDirection < 360:  WindDirection\_dir = 'S'  if 22.5 <= WindDirection and WindDirection < 67.5:  WindDirection\_dir = 'SW'  if 67.5 <= WindDirection and WindDirection < 112.5:  WindDirection\_dir = 'W'  if 112.5 <= WindDirection and WindDirection < 157.5:  WindDirection\_dir = 'NW'  if 157.5 <= WindDirection and WindDirection < 202.5:  WindDirection\_dir = 'N'  if 202.5 <= WindDirection and WindDirection < 247.5:  WindDirection\_dir = 'NE'  if 247.5 <= WindDirection and WindDirection < 292.5:  WindDirection\_dir = 'E'  if 292.5 <= WindDirection and WindDirection < 337.5:  WindDirection\_dir = 'SE'  print("WindDirection:" +str(WindDirection) +" degree","WindDirection\_dir:"+WindDirection\_dir)    '''get wind speed'''  # The average wind speed of the previous minute.  try:  WindSpeedAverage = round(0.44704 \* float(databuffer[5:8]),1)  except:  WindSpeedAverage = 0  print("Average Wind Speed (One Minute):" + str(WindSpeedAverage) + "m/s ")  # The maximum wind speed of the previous five minutes.  try:  WindSpeedMax = round(0.44704 \* float(databuffer[9:12]),1)  except:  WindSpeedMax = 0   print("Max Wind Speed (Five Minutes):" + str(WindSpeedMax) + "m/s")     '''get temperture'''  try:  Temperature = round((float(databuffer[13:16]) - 32.00) \* 5.00 / 9.00,2)   except:  Temperature = 0  print("Temperature:" + str(Temperature)+ "℃ ")   # print("Temperature:" + "{:.2f}".format(Temperature)+ "C ")     '''get humidity'''  try:  Humidity = round(float(databuffer[25:27]) ,1)  except:  Humidity = 0  print("Humidity:" + str(Humidity) +"% ")    '''get pressure'''  try:  BarPressure = round(float(databuffer[28:33])/ 10.00,1)  except:  BarPressure = 0  print("BarPressure:" + str(BarPressure) + "hPa")    else:   databuffer = ""   time.sleep(0.5) |

3. **Flask server（Web）**

|  |
| --- |
| Python # -\*- coding: UTF-8 -\*- from flask import Flask,Response,render\_template,request flask\_app = Flask(\_\_name\_\_)  def rec\_route\_funca():  print("b click")  return "rount\_func" def rec\_route\_funcb():  print("b click")  return "b" def rec\_index():  return render\_template("test.html")  @flask\_app.route('/index',methods=['GET','POST']) def route\_index():  return rec\_index() flask\_app.run(host='0.0.0.0', port=5000, threaded=True) |

4. **Python runs on Lattepanda(save pictures, query the online status of nodes)**

|  |
| --- |
| Python # -\*- coding: UTF-8 -\*- import time import siot import os  #all IP addresses of unihikers ip = ['ping 10.168.1.114','ping 10.168.1.117','ping 10.168.1.118','ping 10.168.1.122','ping 10.168.1.115','ping 10.168.1.112']  #when get data from unihikers def on\_message\_callback(client, userdata, msg):  global P1  global N1  global K1    global P5  global N5  global K5    global sendata1  global sendata5  global sendataw  global sendata    global weather\_hum  global weather\_tem   global indoor\_hum  global indoor\_tem    if (msg.topic.find("form")!=-1):  pass  else:  if (msg.topic.find("node1/soilN")!=-1):  N1 = msg.payload.decode()  print(msg.topic)  print(N1)  if (msg.topic.find("node1/soilP")!=-1):  P1 = msg.payload.decode()  print(msg.topic)  print(P1)  if (msg.topic.find("node1/soilK")!=-1):  K1 = msg.payload.decode()  print(msg.topic)  print(K1)    status = ((not (P1 == 0)) and (not (N1 == 0)))    if ((not (status == 0)) and (not (K1 == 0))):  sendata1 = N1 +","+ P1 +","+ K1  print(sendata1)  siot.publish\_save(topic="siot/node1/NPKform", data=sendata1)    if (msg.topic.find("node5/soilN")!=-1):  N5 = msg.payload.decode()  print(msg.topic)  print(N5)  if (msg.topic.find("node5/soilP")!=-1):  P5 = msg.payload.decode()  print(msg.topic)  print(P5)  if (msg.topic.find("node5/soilK")!=-1):  K5 = msg.payload.decode()  print(msg.topic)  print(K5)    status = ((not (P5 == 0)) and (not (N5 == 0)))    if ((not (status == 0)) and (not (K5 == 0))):  sendata5 = N5 +","+ P5 +","+ K5  print(sendata5)  siot.publish\_save(topic="siot/node5/NPKform", data=sendata5)   if (msg.topic.find("weatherstation/temper")!=-1):  weather\_tem = msg.payload.decode()  print((str(msg.topic) + str(weather\_tem)))  if (msg.topic.find("weatherstation/humi")!=-1):  weather\_hum = msg.payload.decode()  print((str(msg.topic) + str(weather\_hum)))   if ((not (weather\_hum == 0)) and (not (weather\_tem == 0))):  sendataw = weather\_hum+ "," + weather\_tem  print(sendataw)  siot.publish\_save(topic="siot/weatherstation/temper\_humi\_form", data=sendataw)    if (msg.topic.find("node1/temper")!=-1):  weather\_tem = msg.payload.decode()  print((str(msg.topic) + str(weather\_tem)))  if (msg.topic.find("node1/humi")!=-1):  weather\_hum = msg.payload.decode()  print((str(msg.topic) + str(weather\_hum)))  if ((not (indoor\_hum == 0)) and (not (indoor\_tem == 0))):  sendata = indoor\_hum+ "," + indoor\_tem  print(sendata)  siot.publish\_save(topic="siot/node1/temper\_humi\_form", data=sendata)    siot.init(client\_id="",server="10.168.1.100",port=1883,user="siot",password="dfrobot") siot.set\_callback(on\_message\_callback) siot.connect() siot.loop() P1 = 0 N1 = 0 K1 = 0  P5 = 0 N5 = 0 K5 = 0  weather\_hum = 0 weather\_tem = 0  indoor\_tem = 0 indoor\_hum = 0  siot.getsubscribe(topic="siot/node1/temper\_humi\_form")  siot.getsubscribe(topic="siot/weatherstation/temper\_humi\_form") siot.getsubscribe(topic="siot/weatherstation/temper") siot.getsubscribe(topic="siot/weatherstation/humi")  siot.getsubscribe(topic="siot/node1/temper") siot.getsubscribe(topic="siot/node1/humi")  siot.getsubscribe(topic="siot/devicestatus")  siot.getsubscribe(topic="siot/node1/NPKform") siot.getsubscribe(topic="siot/node1/soilN") siot.getsubscribe(topic="siot/node1/soilP") siot.getsubscribe(topic="siot/node1/soilK")  siot.getsubscribe(topic="siot/node5/NPKform") siot.getsubscribe(topic="siot/node5/soilN") siot.getsubscribe(topic="siot/node5/soilP") siot.getsubscribe(topic="siot/node5/soilK")  while True:  outdev = 0  online = 0  for i in ip:  result = os.popen(i)  status = result.read()  loc = "unreachable" in status  print(loc)  if loc != True:  loc = status.find('Lost')  loss\_data = status[loc+7]    if loss\_data == '4':  outdev = outdev+1  else:  online = online+1  else:  outdev = outdev+1   output = str(online) + ','+ str(outdevice)  print("online,outdevice:")  print(output)  siot.publish\_save(topic="siot/devicestatus", data=output)  print("send ok")  time.sleep(100) |

Download the full code: https://github.com/polamaxu/AgriculturalSmartSystem

**Summary**

This IoT agriculture monitoring system is designed to effectively detect soil nutrients like temperature, humidity, and PH, as well as nitrogen, phosphorus, and potassium levels in a stable manner. It can also continuously monitor the growth environment for crops both outdoors and the inside greenhouse. By leveraging wireless WIFI, sensor data and images from the greenhouse and outdoor weather station can be uploaded to the cloud in real-time, enabling the provision of real-time analysis of soil nutrients, environmental lighting, and environmental parameters, which are essential in providing accurate data support for agriculture.

**FAQ**

1. The screen of Unihiker keeps showing that the wifi is reconnecting.

If the screen displays a loop of 'wifi connected' followed by 'wifi reconnecting,' please restart the Siot server on the LattePanda. If this does not appear, please check if the router is functioning properly.

1. Unihiker shows the error: runtime error: analog map retrieval time out.

Return to the Unihiker menu page and run the program again. If the problem still persists, you can flash the processor firmware https://www.Unihiker.com/wiki/faq

1. How to remotely monitor the screen of each line board?

Use the software remote desktop connection and fill in the IP to access.

1. How to remotely control and view the programs of each Unihiker?

Use mobaXterm, and select the IP corresponding to Unihiker to open Unihiker's system. The account is "siot", and the password is "dfrobot".

**reference**

1. Unihiker FAQ