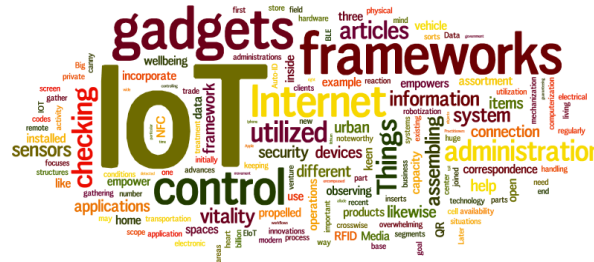


Smart Home Monitoring and Control Using LoRaWAN and Webserver



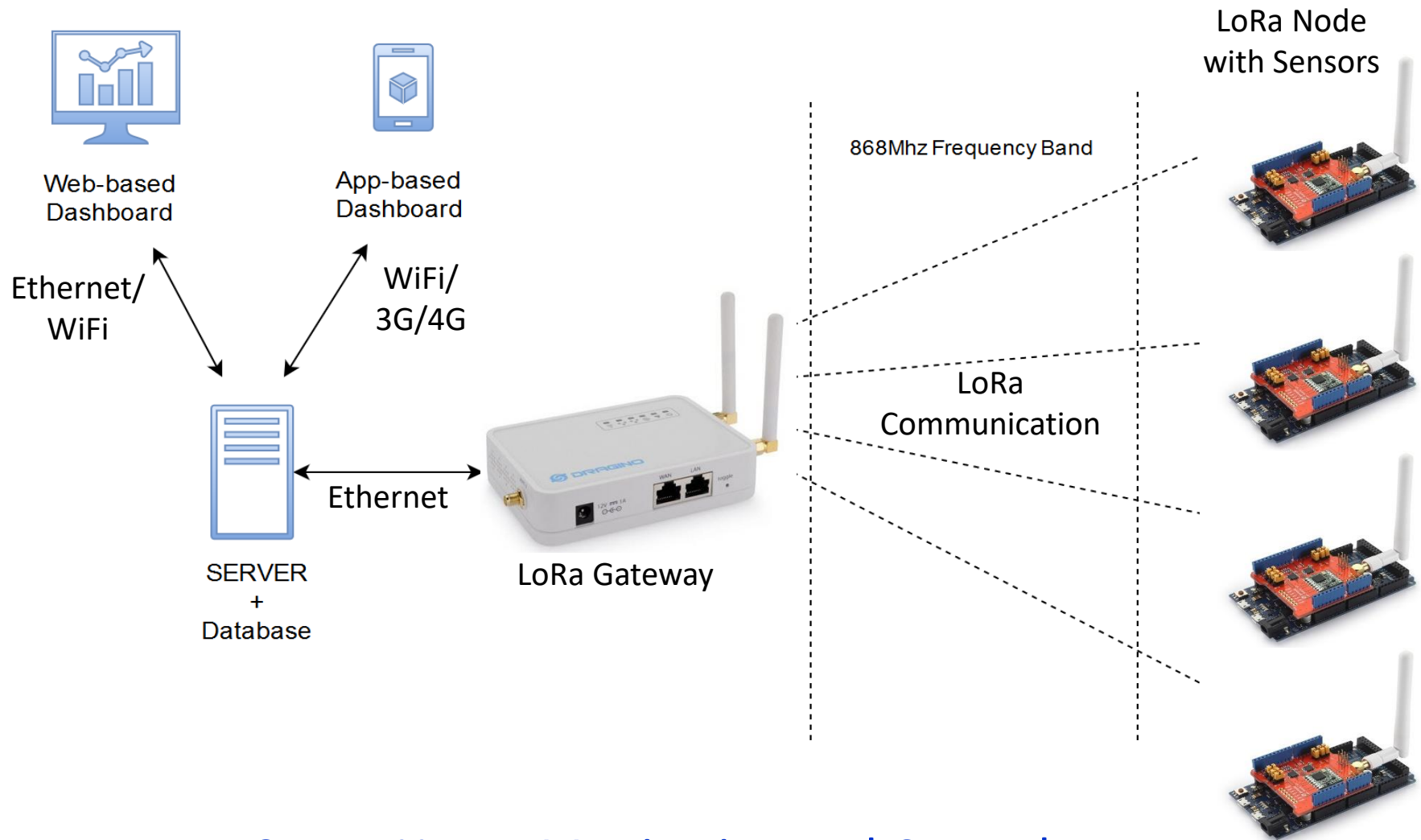
Dr. Manas Khatua

Assistant Professor, Dept. of CSE, IIT Guwahati

E-mail: manaskhatua@iitg.ac.in , URL: <http://manaskhatua.github.io/>

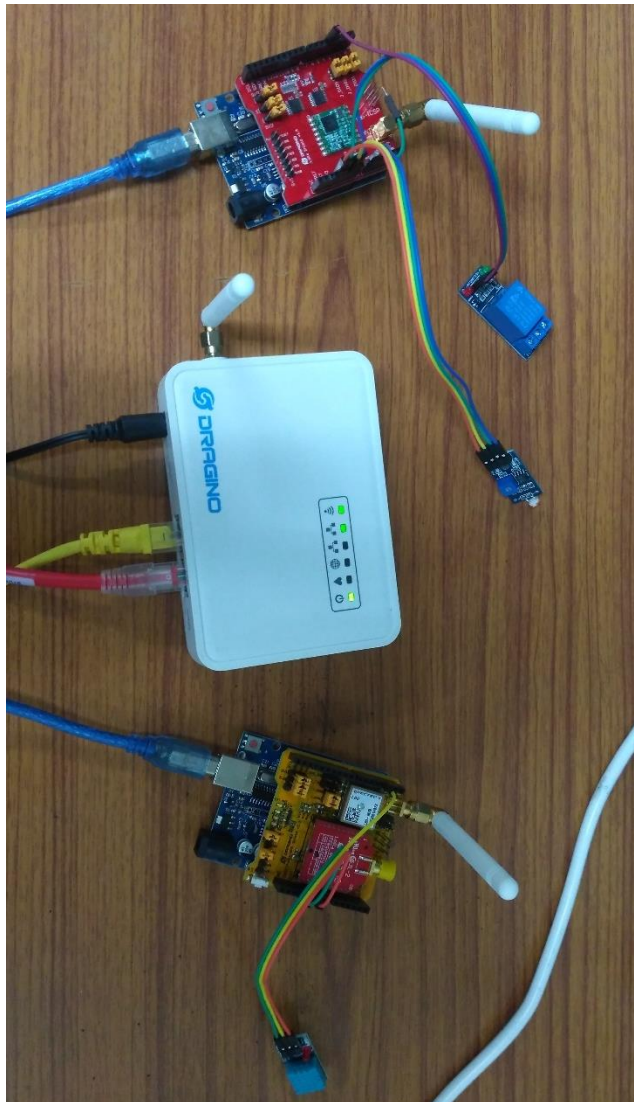
"If you want peace of mind, do not find fault with others. Rather learn to see your own faults." – Sarada Devi

System Diagram

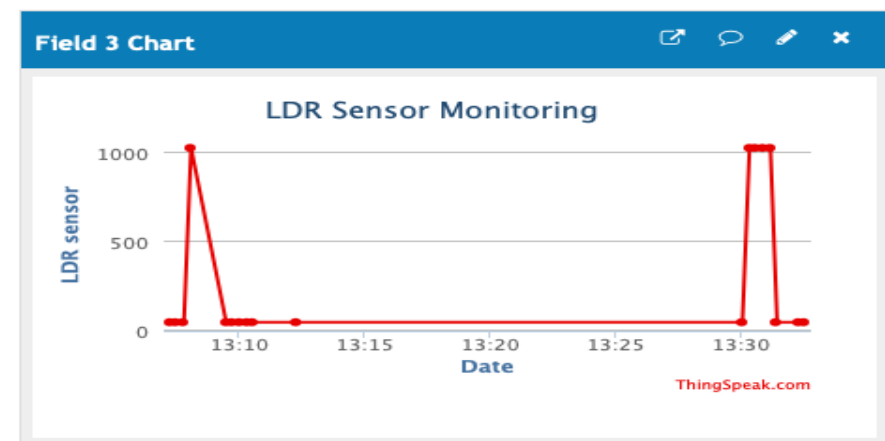
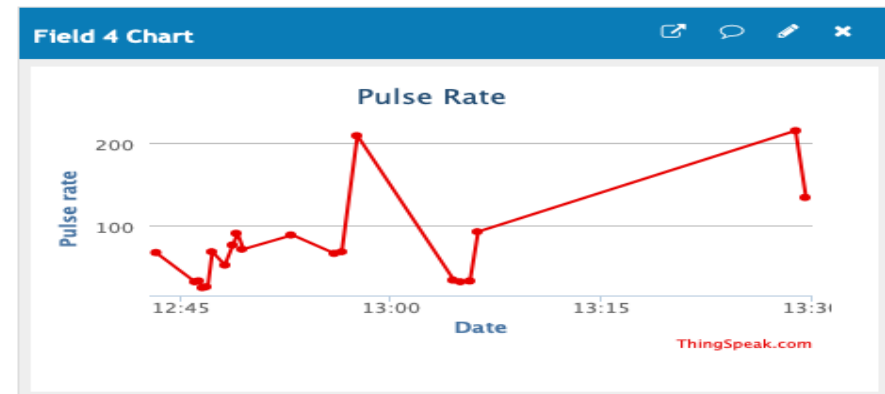


Smart Home Monitoring and Control

Physical Setup



Application Dashboard running in **Web server** accessing from a Laptop/PC/Smartphone



LoRa Gateway / Router Configuration To Connect with IITG Internet

Router Configuration



- This is **LoRA Gateway**
- LoRa sensor nodes will be connected to this Gateway
- Internet connection will be accessible through it.

- Connect IITG LAN to the Gateway WAN port using Ethernet Cable
- Connect your PC/Laptop to the Gateway LAN port using Ethernet Cable
- Login LoRa gateway using given IP (**10.130.1.1**) and user ID (**root**) and password (**dragino**)
- Do the following:
 - Gateway WiFi SSID and Password under “Wireless” tab
 - **SSID**: dragino-1bf050; **Password**: 12345678
 - Time and Date settings
 - You can change admin password
 - [Setup Internet Connection](#) by Network → Internet access
 - Set the **Static IP**, **Subnet Mask**, **Default Gateway**, **DNS Server**, **Alternate DNS Server**
 - Reboot the gateway

Cont...



dragino-1bf050

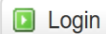
Authorization Required

Please enter your username and password.

Username

root

Password



Login



Reset

dragino-1bf050

Status ▾

Sensor ▾

System ▾

Network ▾

Logout

Status

System

Hostname	dragino-1
Router Model	dragino
Firmware Version	IoT-4.3.6
Build Time	Tue Mar 5 23:54:10 CST 2019
Kernel Version	3.18.45

Internet Access

LAN and DHCP

Access Point

Mesh Network

Firewall

dragino-1bf050

Status ▾

Sensor ▾

System ▾

Network ▾

Logout

Small Enterprise-Campus Network

Internet Access

Access Internet Via

WAN Port ▾

Way to Get IP

Static IP ▾

IP Address

172.16.114.162

Netmask

255.255.248.0 ▾

Gateway

172.16.112.1

DNS Server

202.141.80.9

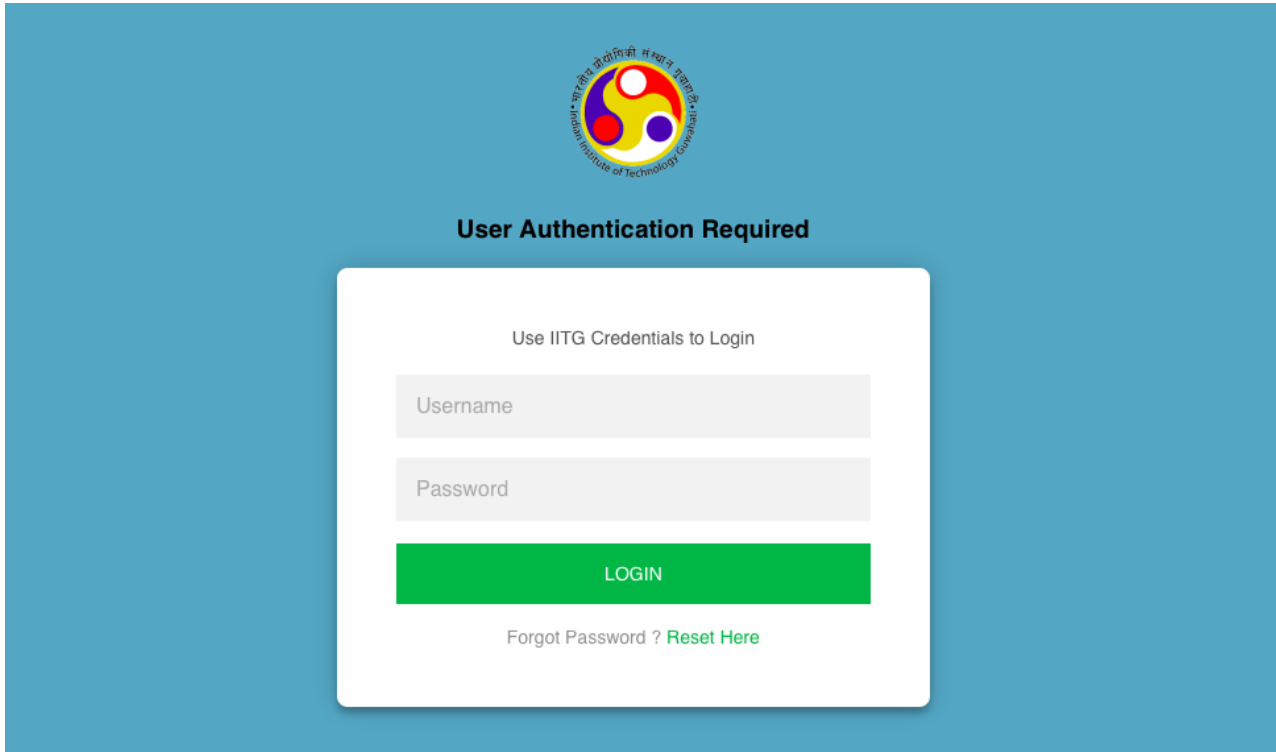
Display Net Connection

8.8.8.8



Continuously Check Net Connection

Connecting with Internet



The image shows a login interface for the Indian Institute of Technology Guwahati (IITG). At the top center is the IITG logo, which is a circular emblem with three interlocking rings in red, yellow, and blue, surrounded by the text 'Indian Institute of Technology Guwahati' in English and Assamese. Below the logo, the text 'User Authentication Required' is displayed in bold. Underneath this, a white rectangular box contains the login form. Inside the box, the text 'Use IITG Credentials to Login' is centered. Below this text are two input fields: 'Username' and 'Password'. Below the password field is a green button with the text 'LOGIN' in white. At the bottom of the box, the text 'Forgot Password ? [Reset Here](#)' is displayed, with 'Reset Here' being a green link.

- You should be able to access Internet in your PC/Laptop using LoRa Gateway

Web Server Configuration to Access Web Service

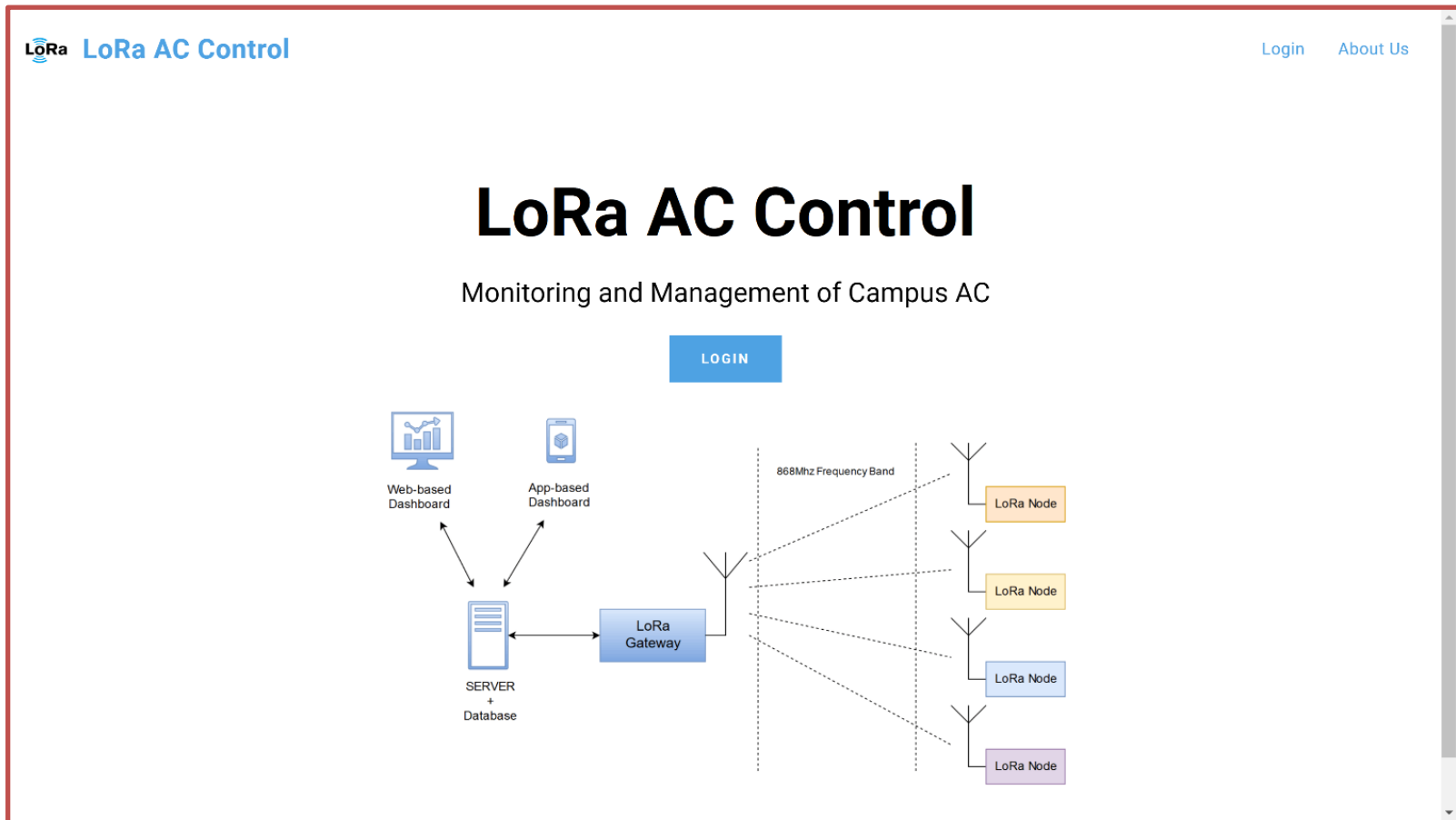
Configure to use Web Server



- The server configuration is done on an Ubuntu 16.04 system. It can be implemented on any OS with a web server and PHP support.
- Run the following commands on Ubuntu Terminal:
 - `sudo apt-get update`
 - `sudo apt-get install apache2 -y`
 - `sudo apt-get install php libapache2-mod-php -y`
 - `sudo rm /var/www/html/index.html`
 - `sudo chown <user>:<user> /var/www/html` ##change <user> to your Ubuntu username
- Server configuration will be required multiple libraries and javascripts.
- Download all files from <https://jayanta525.github.io/server.zip>
- Extract the files in /var/www/html directory with the correct permission.
- Run the commands:
 - `sudo chmod -R 777 /var/www/html`
 - `sudo systemctl restart apache2`
- Check proper connection by going to **localhost** in a web browser.

Cont...

- Check proper connection by going to **localhost** in a web browser.
- Dashboard should be opened as shown below:



Arduino Tool Configuration

Install and Configure Arduino IDE



- Download and **Install** Arduino IDE from <https://www.arduino.cc/en/Main/Software>
- **Download** the installation archive file
- **Extract** the downloaded file
- Make the install file **executable** with “chmod +x install.sh”
- **Run** “sudo ./install.sh”
- Provide root password and ArduinoIDE will be installed.
- To **open** ArduinoIDE, run “arduino” in terminal
- It is suggested to run ArduinoIDE as **root user** with “sudo arduino”.

Cont...

- When the Arduino IDE first opens, this is what you should see:



```
sketch_jul11a | Arduino 1.8.9
1 void setup() {
2   // put your setup code here, to run once:
3 }
4
5
6 void loop() {
7   // put your main code here, to run repeatedly:
8 }
9
```

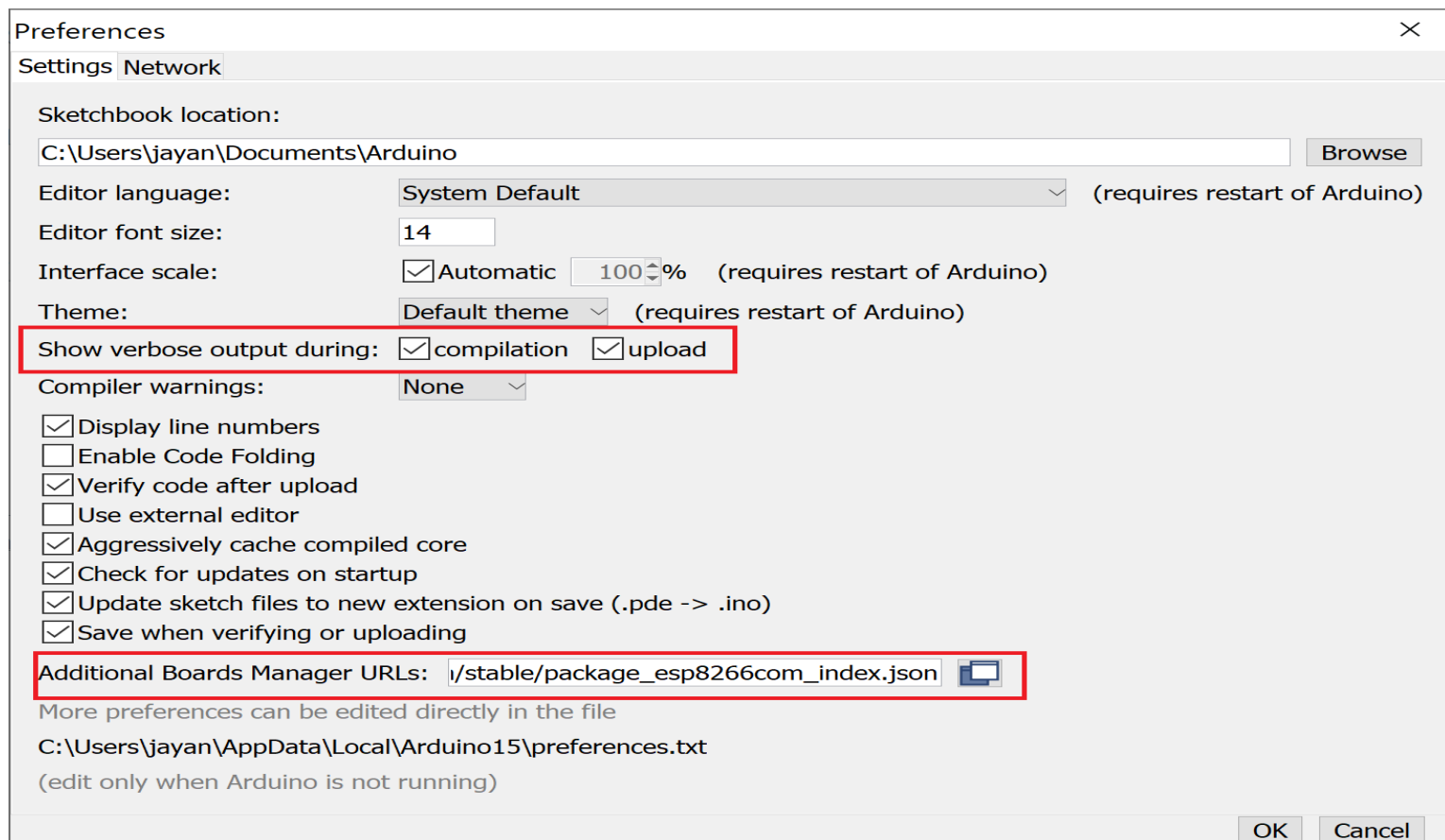
80 MHz, Flash, Disabled, All SSL ciphers (most compatible), 4M (no SPIFFS), v2 Lower Memory, Disabled, None, Only Sketch, 115200 on /dev/cu.SLAB USBtoUART2

- Two functions exist in the programme: **setup ()** and **loop ()**
 - setup():** This function runs once when ESP first boots
 - loop():** This function reads the sensor value and connects to server, and then sends data to server

Install Libraries in IDE

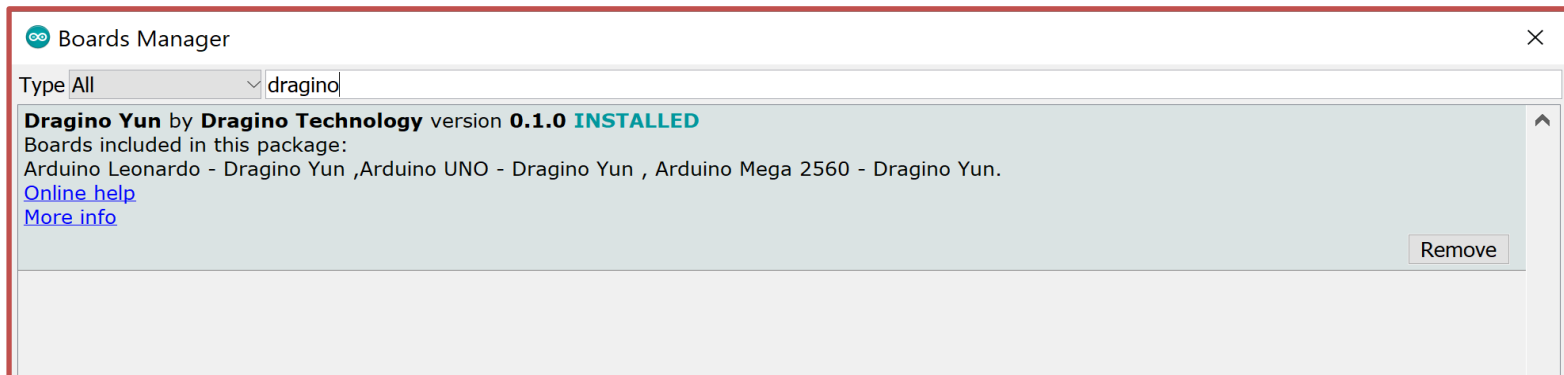
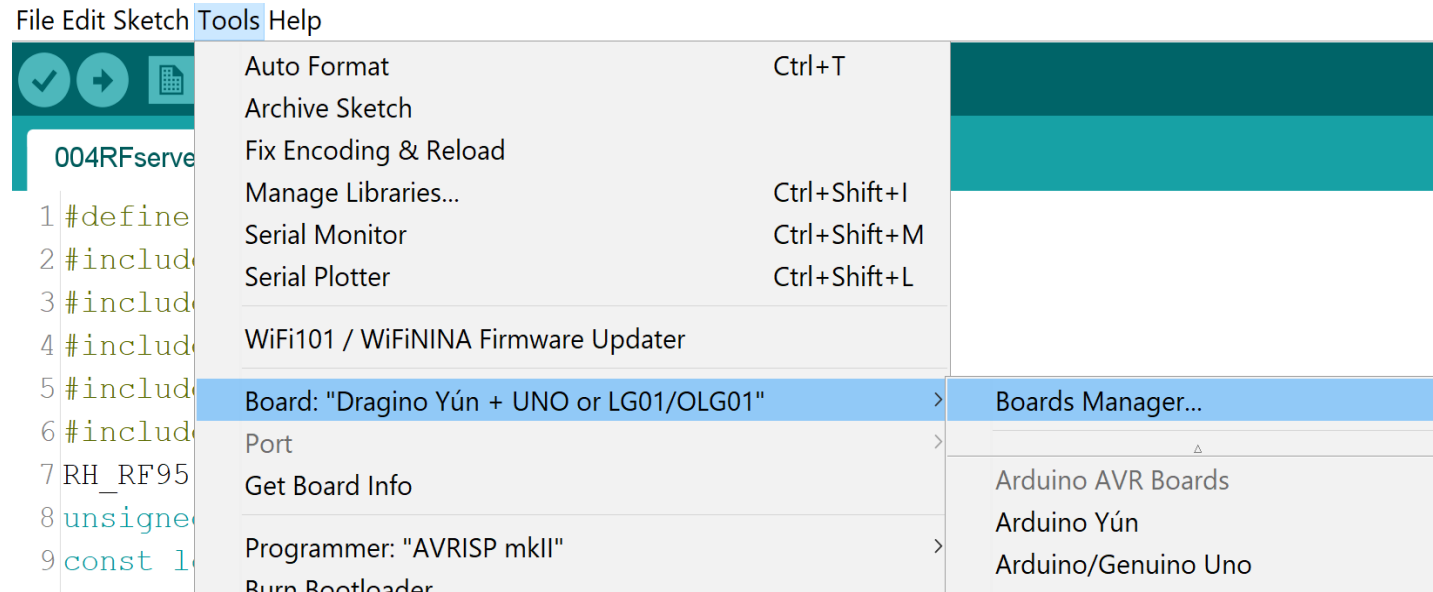


- Go to **File --> Preferences**
- Check Show Verbose output during, **compilation** and **upload**
- Enter the below URL into **Additional Board Manager URLs** field and press the “OK” button
http://www.dragino.com/downloads/downloads/YunShield/package_dragino_yun_test_index.json



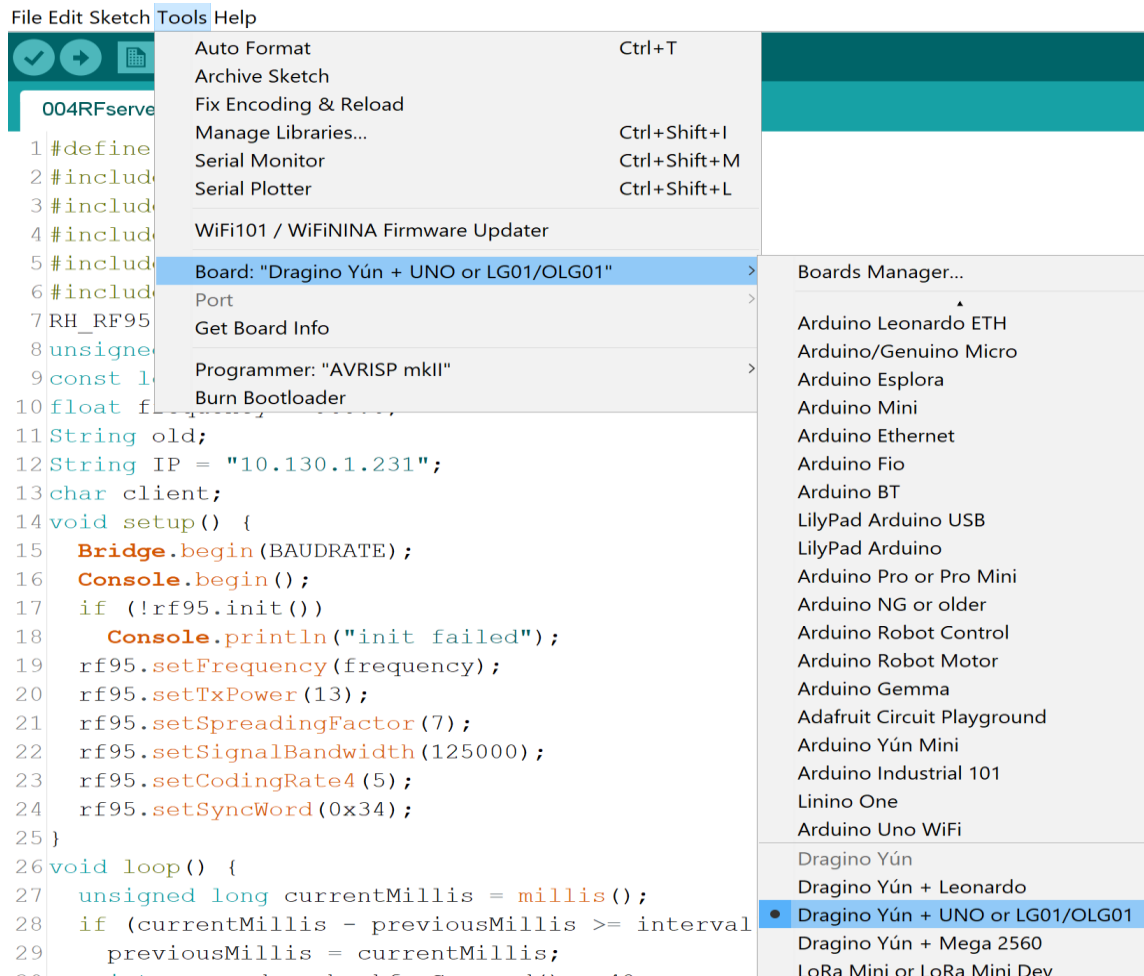
Cont...

- Go to **Tools > Board > Board Manager**
- Search for **Dragino** board and **install** latest version of “**Dragino Yun by Dragino Technology**”



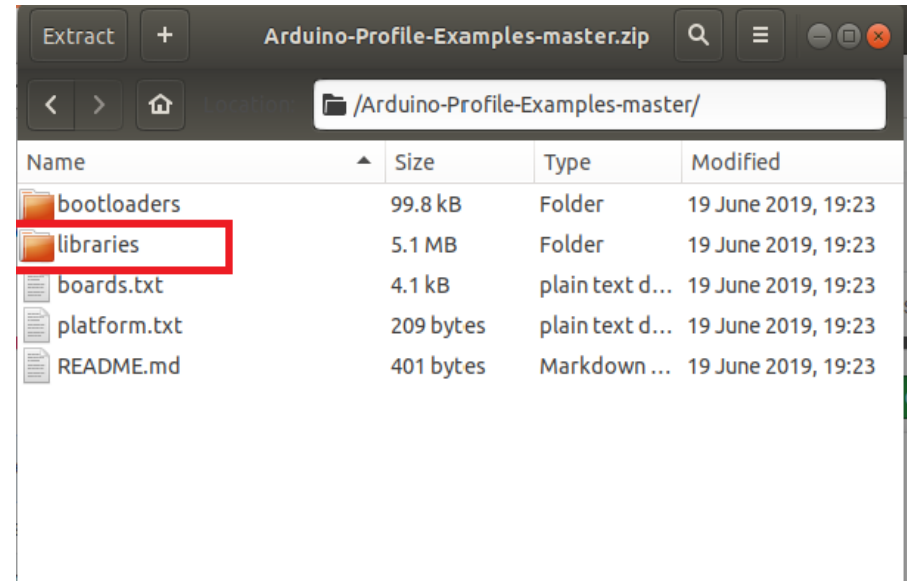
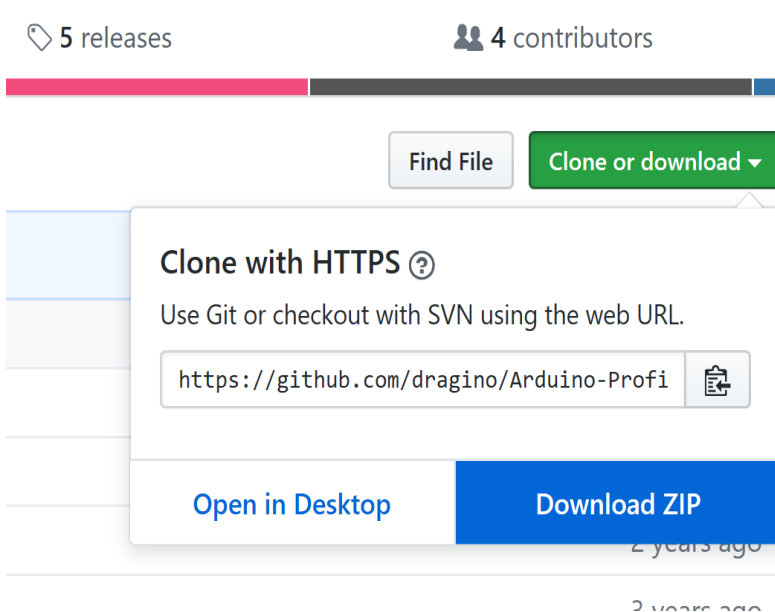
Cont...

- Select the appropriate board after installation
 - Go to **Tools > Board > Dragino Yun + UNO or LG01/OLG01**



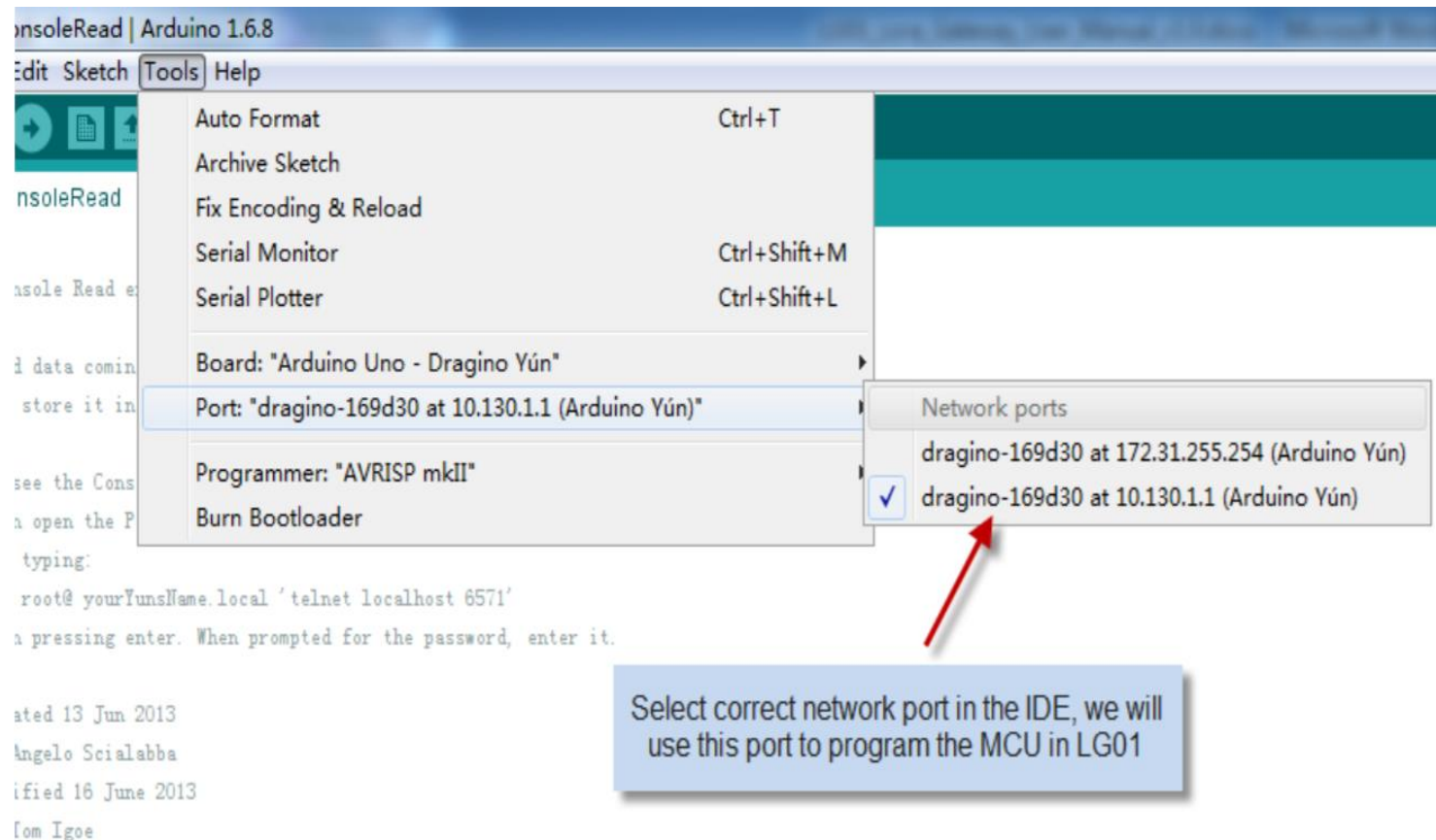
Install Dragino Custom Libraries

- Download libraries from <https://github.com/dragino/Arduino-Profile-Examples>
- Extract and put the “**libraries**” folder only into Arduino folder of the ubuntu system
 - **Run** the File Manager application as root user by the command “**sudo nautilus**”
 - **Paste** the “libraries” folder into **/root/Arduino/libraries**.
- **Restart** ArduinoIDE



Cont...

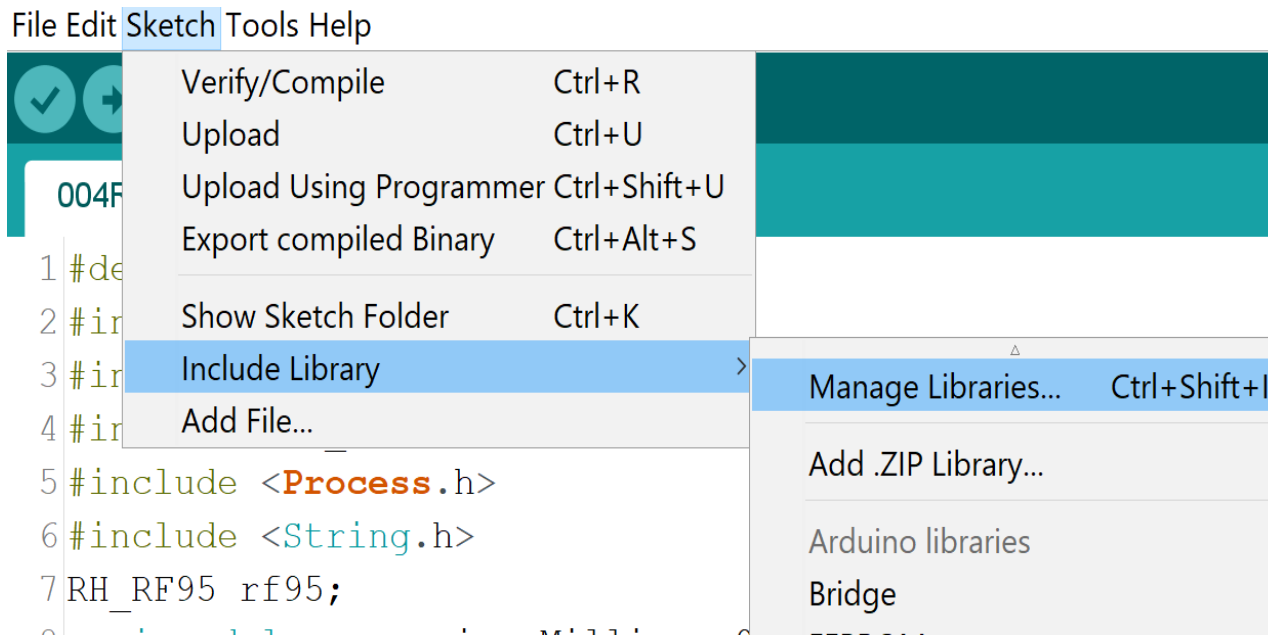
- **Check** whether network port is discovered at **Tools > Port** of ArduinoIDE
- Select network port at “10.130.1.1”



- If the network port doesn't show up, refer “**ALTERNATE UPLOAD**” section.

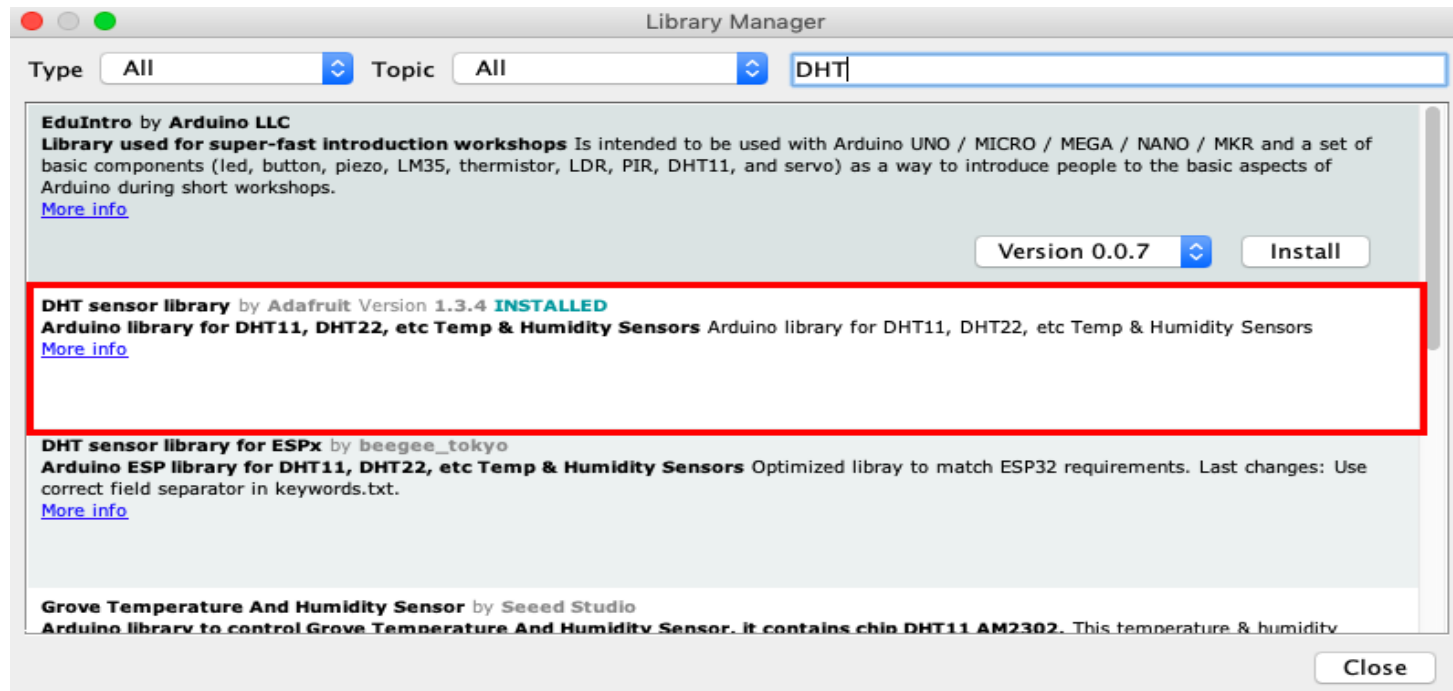
Install Sensor Libraries

- In this demo, we use **few sensors** for which we need to install few corresponding libraries.
- Install Using the **Library Manager**
 - click to **Sketch** menu then **Include Library > Manage Libraries**




Cont...

- Install the following:
 - <https://github.com/adafruit/DHT-sensor-library>
 - https://github.com/adafruit/Adafruit_Sensor
 - <https://github.com/mikalhart/TinyGPS>
 - <https://github.com/dragino/RadioHead>



Cont...

Adafruit TSL2561 by **Adafruit**
Unified sensor driver for Adafruit's TSL2561 breakouts Unified sensor driver for Adafruit's TSL2561 breakouts
[More info](#)

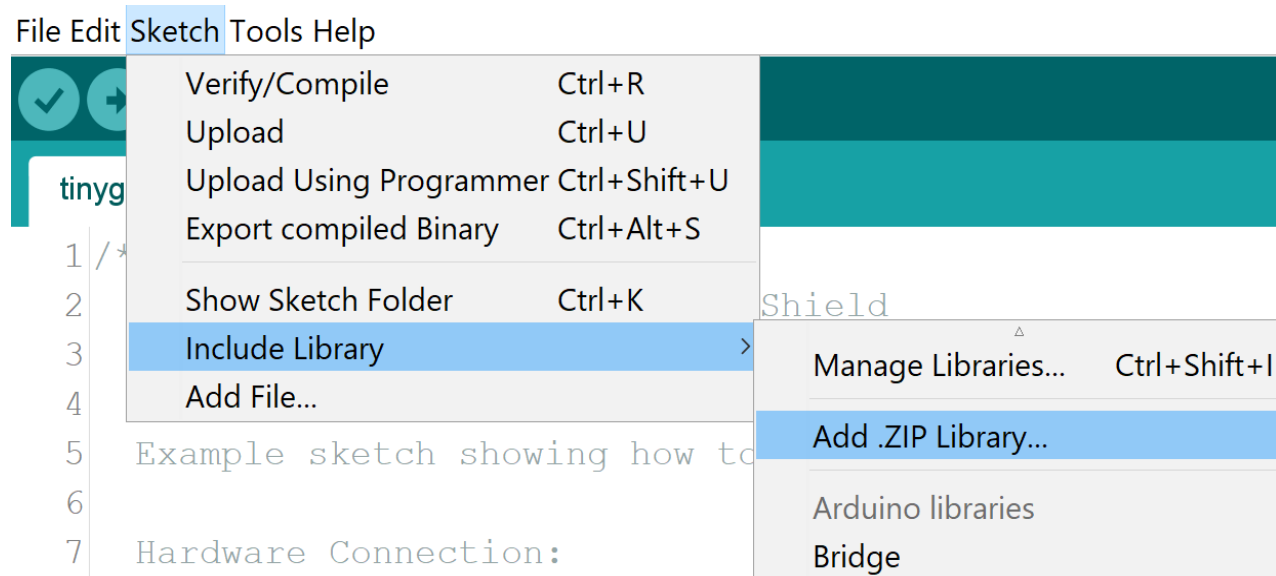
Version 1.0.3  **Install**

Adafruit Unified Sensor by **Adafruit** Version 1.0.3 **INSTALLED**
Required for all Adafruit Unified Sensor based libraries. A unified sensor abstraction layer used by many Adafruit sensor libraries.
[More info](#)

Close

Cont...

- There exist **other methods** for installing libraries
 - **Importing .zip Library**
 - Sketch --> Include Library --> Add .Zip Library

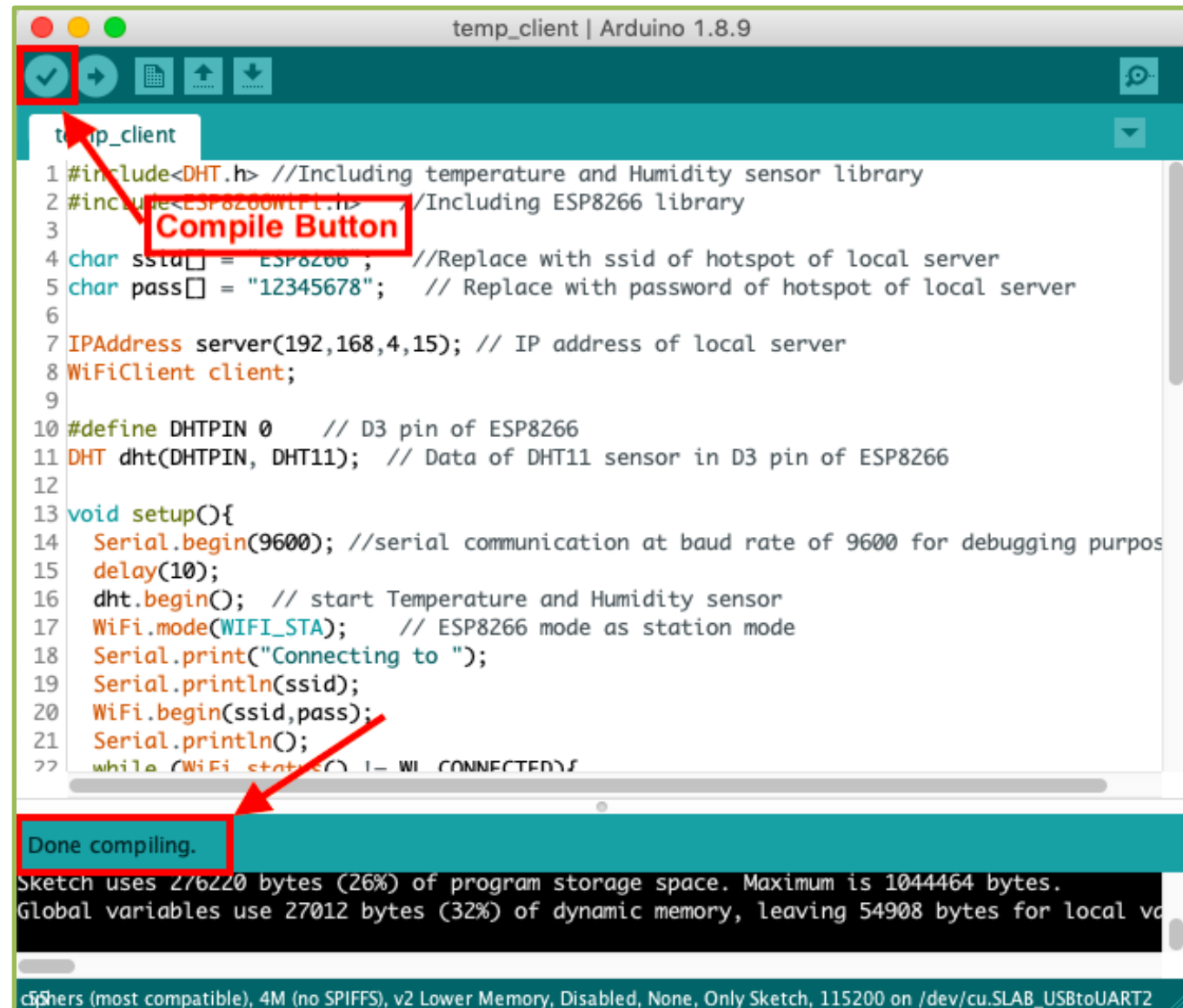


- **Manual Installation of Library**
 - Download the library as .Zip --> extract it
 - Place the files in File --> Preferences --> Sketchbook location
 - Restart Arduino IDE

Code Compilation & Upload Observe Output in Serial Monitor

Code Compilation

Compilation successful
message will appear in
bottom left corner.



```
temp_client | Arduino 1.8.9

temp_client
1 #include<DHT.h> //Including temperature and Humidity sensor library
2 #include<ESP8266WiFi.h> //Including ESP8266 library
3
4 char ssid[] = "ESP8266"; //Replace with ssid of hotspot of local server
5 char pass[] = "12345678"; // Replace with password of hotspot of local server
6
7 IPAddress server(192,168,4,15); // IP address of local server
8 WiFiClient client;
9
10 #define DHTPIN 0 // D3 pin of ESP8266
11 DHT dht(DHTPIN, DHT11); // Data of DHT11 sensor in D3 pin of ESP8266
12
13 void setup(){
14   Serial.begin(9600); //serial communication at baud rate of 9600 for debugging purpos
15   delay(10);
16   dht.begin(); // start Temperature and Humidity sensor
17   WiFi.mode(WIFI_STA); // ESP8266 mode as station mode
18   Serial.print("Connecting to ");
19   Serial.println(ssid);
20   WiFi.begin(ssid,pass);
21   Serial.println();
22   while (WiFi.status() != WL_CONNECTED){
```

Done compiling.

Sketch uses 276220 bytes (26%) of program storage space. Maximum is 1044464 bytes.
Global variables use 27012 bytes (32%) of dynamic memory, leaving 54908 bytes for local variables.

Compilers (most compatible), 4M (no SPIFFS), v2 Lower Memory, Disabled, None, Only Sketch, 115200 on /dev/cu.SLAB_USBtoUART2

Code Compilation + Upload

Done uploading message will appear in bottom left corner.

Note: select appropriate Board and Port to upload code

The screenshot shows the Arduino IDE interface. At the top, the title bar reads "local_server | Arduino 1.8.9". The menu bar includes File, Edit, Tools, Window, Help, and a custom button. Below the menu bar is a toolbar with icons for Check, Run, Upload, Download, and a custom button. A red box highlights the Upload icon (an upward-pointing arrow). Below the toolbar, the code editor displays the following C++ code:

```
#include <ESP8266WiFi.h> //Including ESP8266 library
#include<ESP8266WebServer.h> //Including ESP8266WebServer library for web serv
#include<ThingSpeak.h> //Including ThingSpeak library
//Static IP address of local server
IPAddress IP(192,168,1,15);
//Gateway of the network
IPAddress gateway(192,168,4,1);
//Subnet mask of the network
IPAddress mask(255, 255, 255, 0);

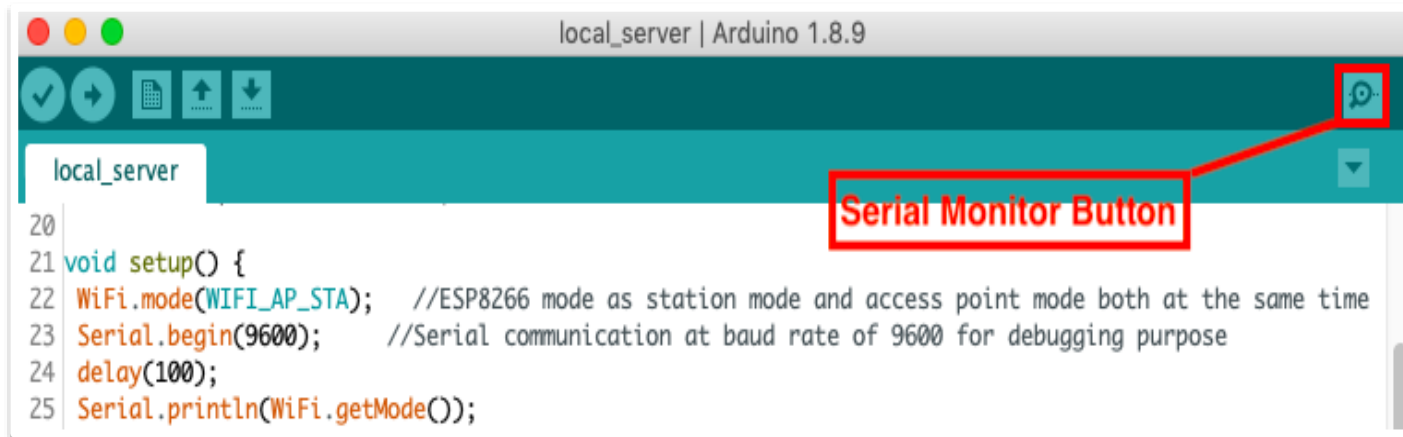
WiFiClient client;
WiFiServer server(80);

unsigned long myChannelNumber = 814887; //Replace with channelID of ThingSpeak
const char * myWriteAPIKey = "EK4LTPHWU4GGE0VP"; //Replace with WriteAPIKey of
const char* softAPssid = "ESP8266"; //SSID of the hotspot of ESP8266 acting
const char* password = "12345678"; //Password of the hotspot of ESP8266 act
const char* wifissid = "Tenda_8060A0"; //Replace with SSID of WIF router provi
const char* pass = "12345678"; //Password of WIFI router providing inte
```

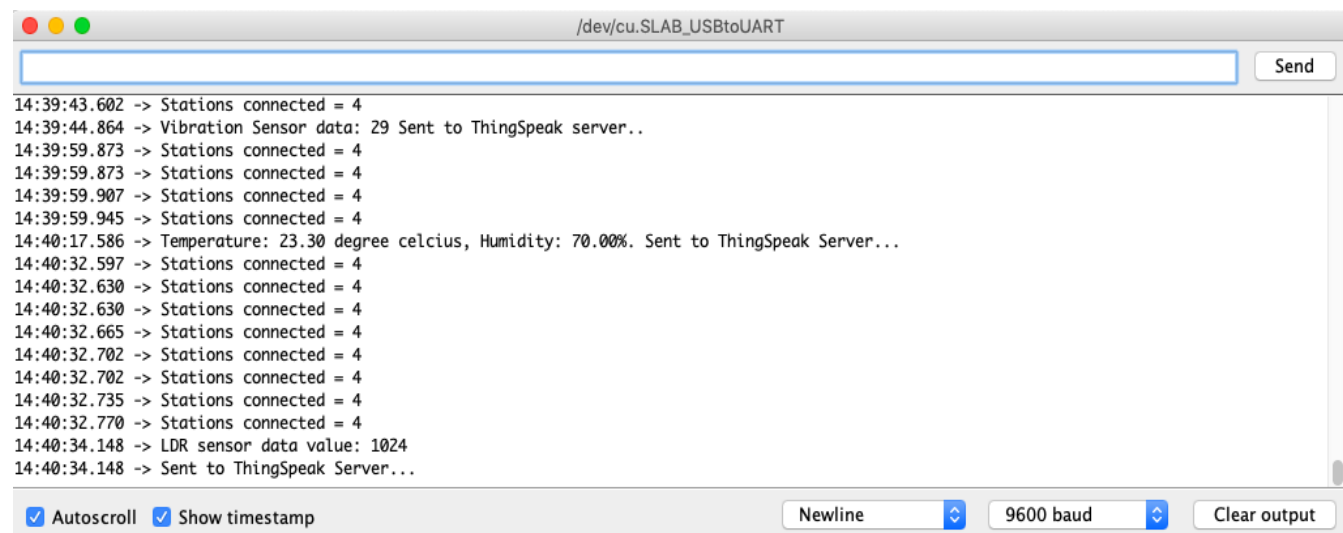
A red box highlights the "Upload Button" label, which is positioned over the Upload icon in the toolbar. Below the code editor, the status bar shows "Done uploading." and "NodeMCU 1.0 (ESP-12E Module) on /dev/cu.SLAB_USBtoUART".

Open Serial Monitor

- First **select the board and port** from which the output will be taken.
- Click the icon of **Serial Monitor** on the top right side of the Arduino IDE.



Serial Monitor of Server



Initial Testing of LoRa Nodes and Gateways

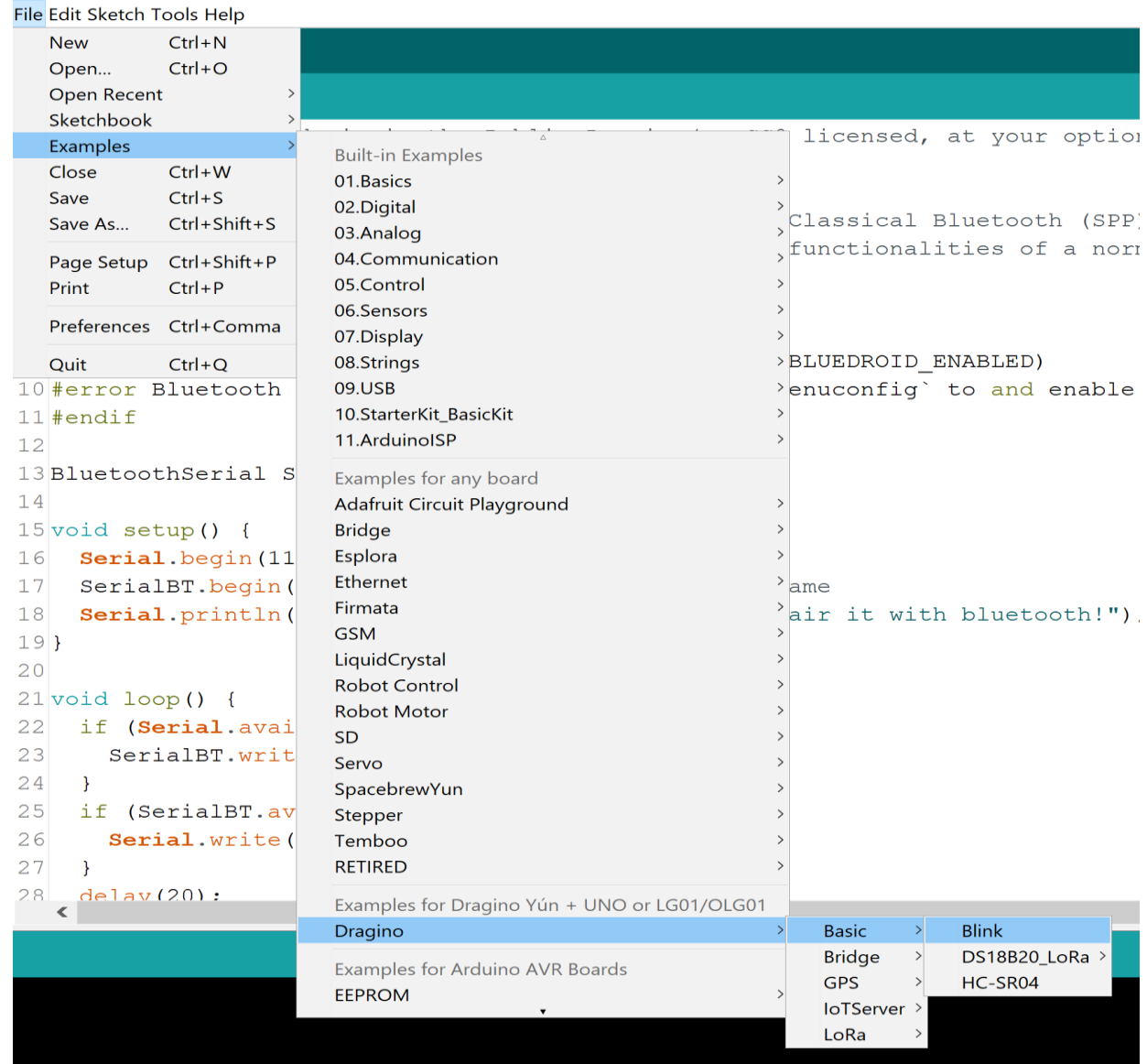
Testing LoRa Gateway

The screenshot shows the Arduino IDE interface. The 'Tools' menu is open, and the 'Board' and 'Port' options are selected. The 'Board' dropdown menu shows 'Dragino Yún + UNO or LG01/OLG01' as the selected option. The 'Port' dropdown menu shows 'dragino-169d30 at 10.130.1.1 (Arduino Yún)' as the selected option. A red arrow points to the selected port. A text box explains: 'Select correct network port in the IDE, we will use this port to program the MCU in LG01'.

- **Open** the Arduino IDE
- **Select** proper **board** and **port**

Cont...

- **Open** test Example from **File -> Examples -> Dragino -> Basic -> Blink**
- **Compile** and **Upload**



Cont...

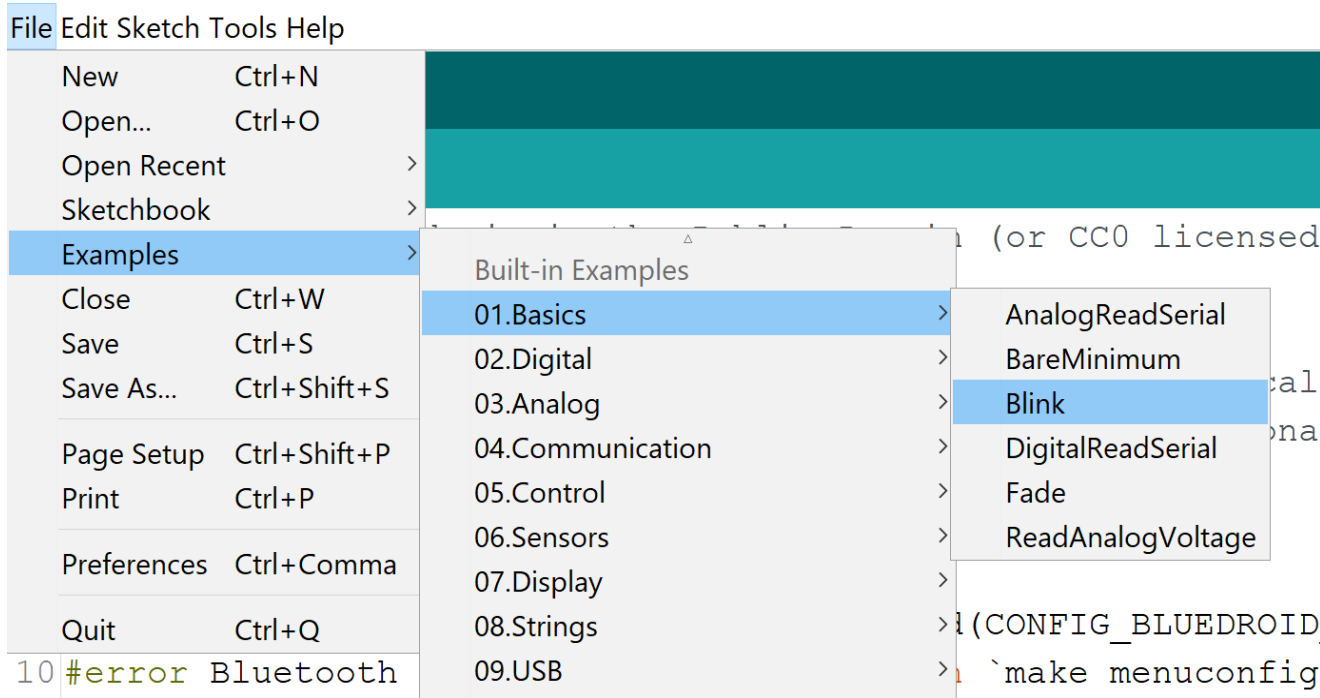
On successful upload, the **heart shaped LED** on the Gateway **will blink** at the interval of 1 second.



Testing Arduino Uno + Lora Shield

- **Connect** the UNO with the LoRa Shield and to the PC
- **Open** Arduino IDE
- **Select** the board as “**Arduino/Genuino Uno**”
- **Select** the respective USB **COM port** (e.g. “COM3”)
- **Open Examples -> Basics -> Blink**
- **Compile and Upload**

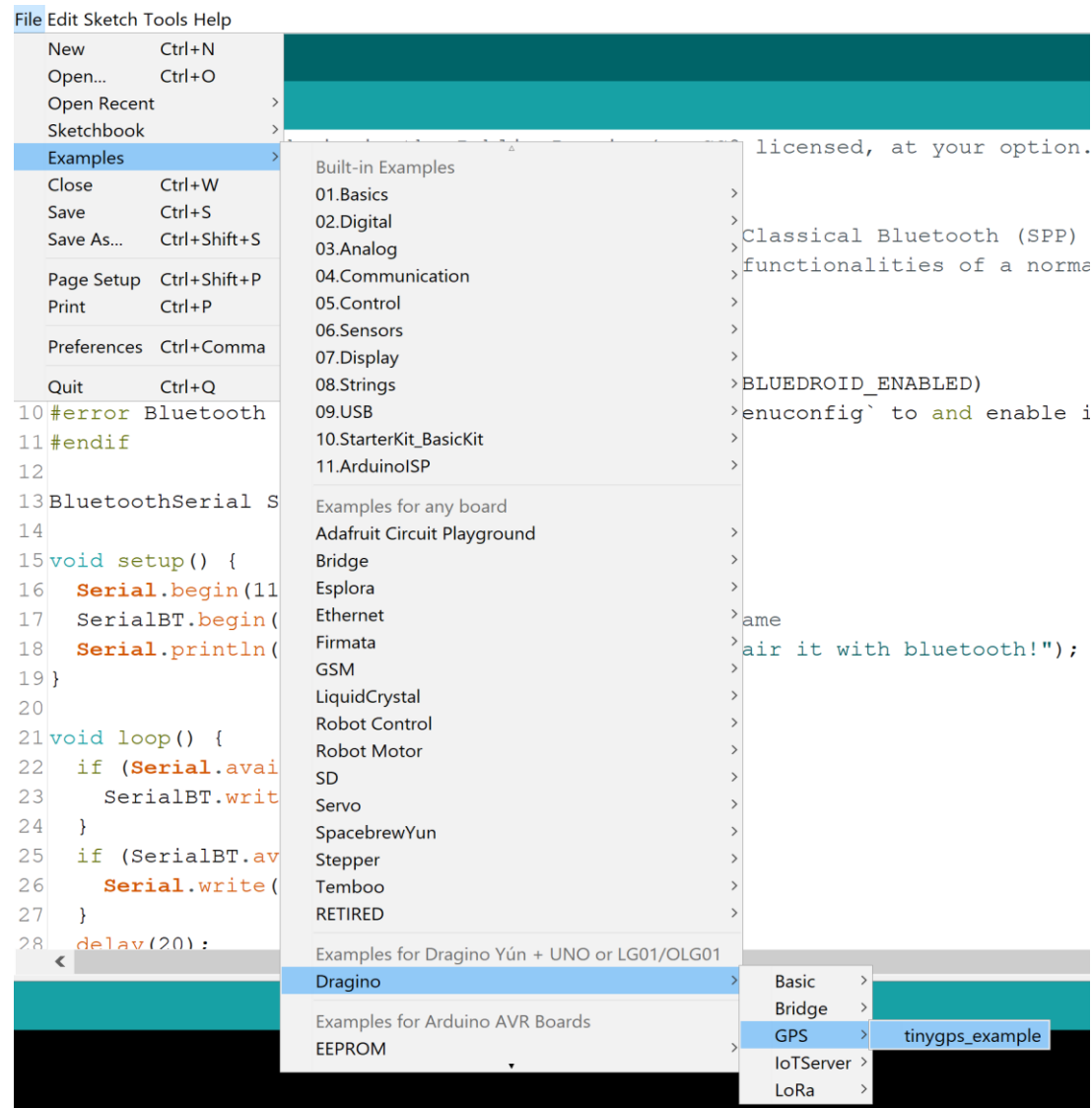
Built-in LED on LoRa shield should blink at an interval of 1 second



Testing Arduino Uno + GPS/Lora Shield



- **Connect** the UNO with the GPS/LoRa Shield and to the PC
- **Open** Arduino IDE
- **Select** the board as “Arduino/Genuino Uno”
- **Select** the respective USB COM port (e.g. “COM3”)
- **Open Examples -> Dragino -> GPS -> tinygps_example**

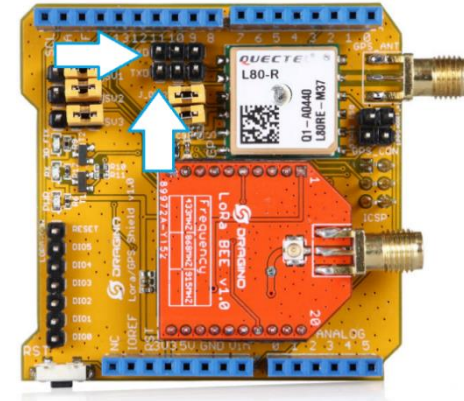


Cont...

- Before uploading the program, **unplug** two jumpers from the board as shown
- **Connect** GPS TX/RX as shown to digital pin 3 and 4 of the Arduino UNO board
 - GPS_RX -> Digital pin 4
 - GPS_TX -> Digital pin 3
- **Compile** and **Upload**
- **Open** Serial Monitor and choose the correct baud rate as written in the program

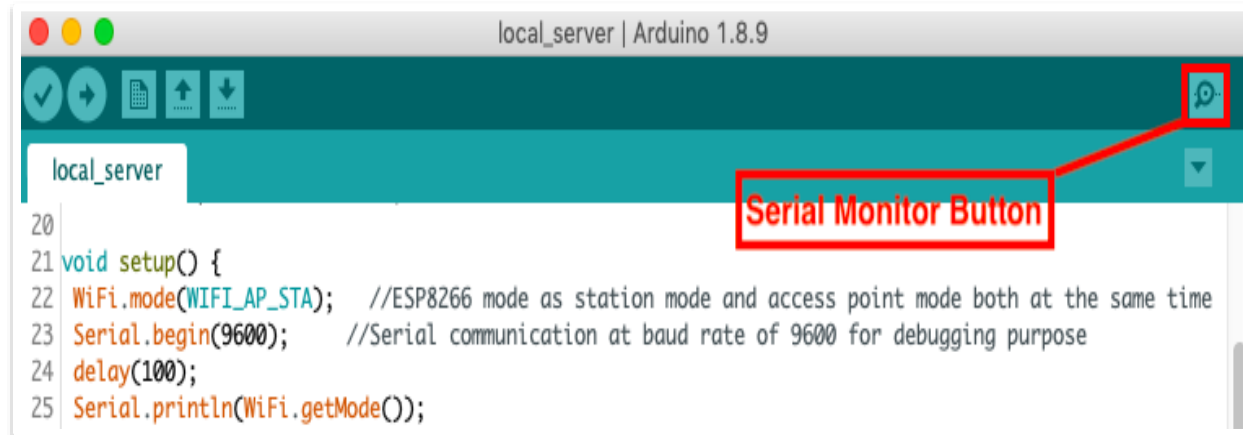
```
14 void setup() {  
15   // initialize serial communication at 9600 bits per second:  
16   Serial.begin(9600);  
17 }
```

Note: GPS takes some time to have a signal fixed, works quicker if placed outdoor environment.



Cont...

- Open the serial monitor
- The Serial Monitor should output GPS location and GPS time



COM9

Minitor Dragino LoRa GPS Shield Status
Testing TinyGPS library v. 13

Sats	HDOP	Latitude (deg)	Longitude (deg)	Fix Age	Date	Time	Date Age	Alt (m)	Course ---	Speed from GPS	Card ----	Distance ---- to London	Course ----	Card ----	Chars RX	Sentences RX	Checksum Fail
****	****	*****	*****	****	*****	*****	****	*****	*****	*****	***	*****	*****	***	63	0	1
9	91	26.127021	91.621002	776	06/22/2019	16:57:40	793	74.90	44.73	0.00	NE	7883	318.82	NW	635	2	1
9	91	26.127021	91.621002	860	06/22/2019	16:57:41	903	74.90	44.73	0.00	NE	7883	318.82	NW	1335	5	1
9	91	26.127021	91.621002	118	06/22/2019	16:57:43	171	74.90	44.73	0.00	NE	7883	318.82	NW	2051	8	1
9	91	26.127021	91.621002	172	06/22/2019	16:57:44	220	74.90	44.73	0.00	NE	7883	318.82	NW	2780	10	1
9	91	26.127021	91.621002	305	06/22/2019	16:57:45	354	74.90	44.73	0.00	NE	7883	318.82	NW	3475	12	1
9	91	26.127021	91.621002	441	06/22/2019	16:57:46	466	74.90	44.73	0.00	NE	7883	318.82	NW	4061	14	1

☒ Autoscroll ☐ Show timestamp

Newline 9600 baud Clear output

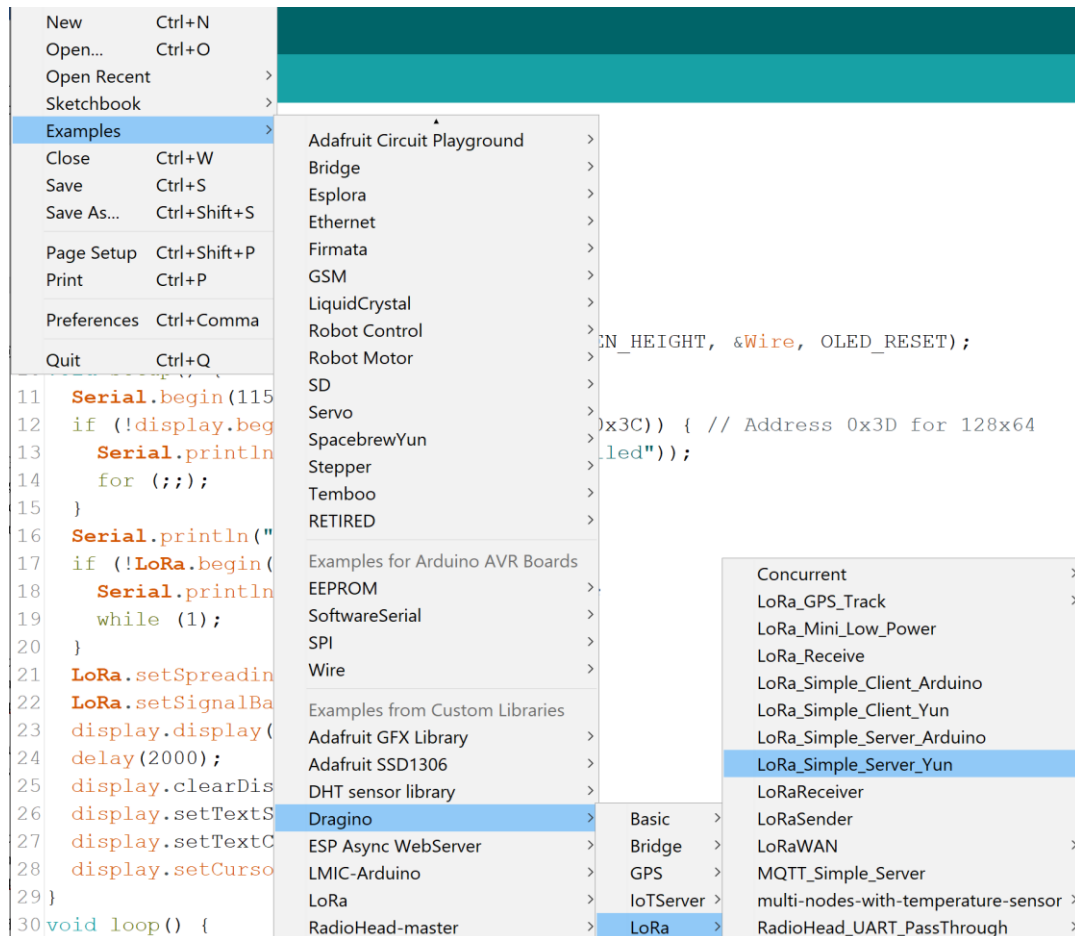
Initial Testing of Communication between LoRa node and LoRa gateway

Gateway configuration

- Power on the Dragino Gateway
- Connect PC with the LAN port of Gateway
- Check whether **PC has IP** lease from Dragino Gateway in the network 10.130.1.0/24

- Open ArduinoIDE
- Select the Gateway board as “Dragino Yun + UNO or LG01/OLG01”
- Select network port as “dragino at 10.130.1.1 (Arduino Yun)”
- Open Examples from **File -> Examples -> Dragino -> Lora -> Lora_Simple_Server_Yun**
- Compile and Upload

Output: The Gateway will now start to listen on the specified frequency and Spreading Factor (SF).



LoRa Node Configuration

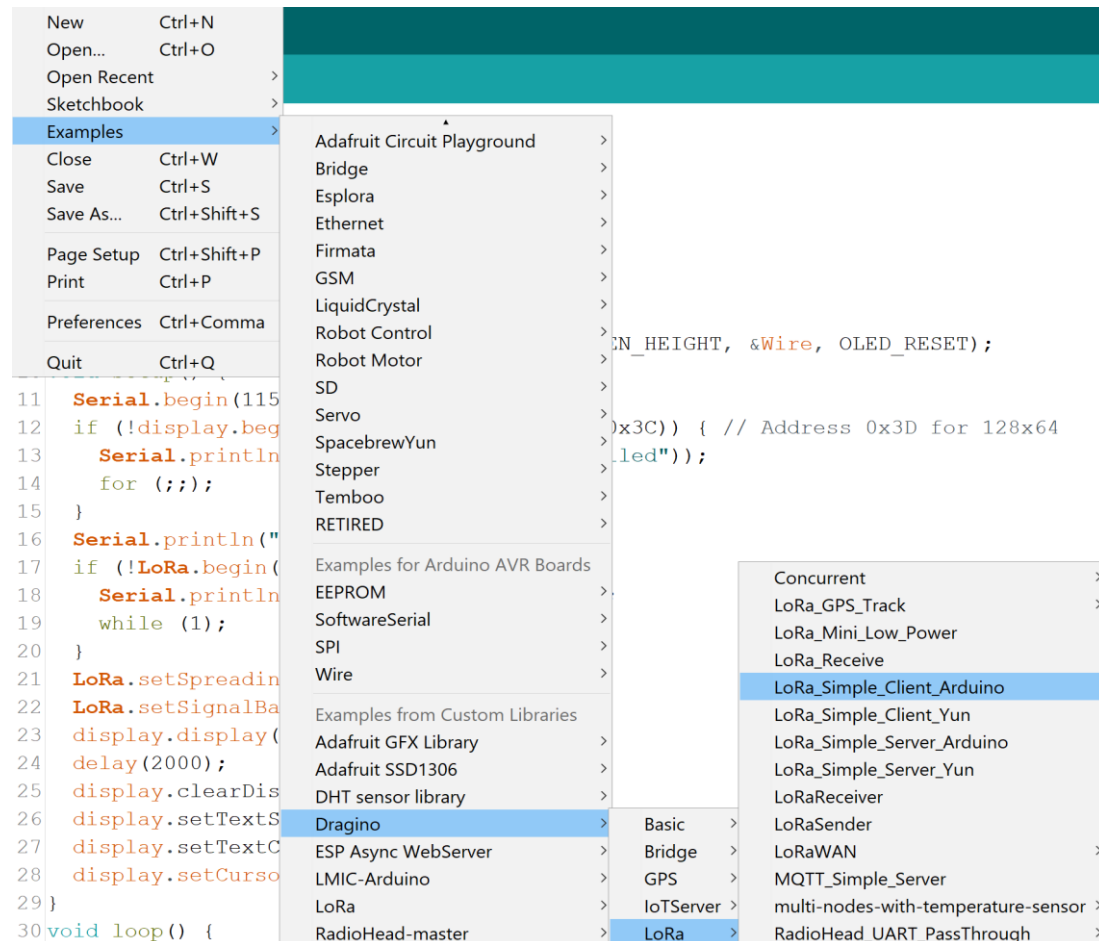
- Connect the UNO with the LoRa Shield, and to the PC using USB cable
- Open Device manager to check the COM port in use for this device
- Open ArduinoIDE and select the board as “Arduino UNO” and the “port”

- Open Examples -> Dragino -> Lora -> Lora_Simple_Client_Arduino

- Compile and Upload

- Open Serial Monitor and choose the correct baud rate as that of the program

Output: The Node will be sending and receiving replies from the Gateway



GPS LoRa Node Configuration

- Connect the UNO with the GPS LoRa Shield, and to the PC using USB cable
- Open Device manager to check the COM port in use for this device
- Open ArduinoIDE and select the board as “Arduino UNO” and the “port”

- Open [Examples -> Dragino -> Lora -> Lora_Simple_Client_Arduino](#)
- Change the device ID display in loop function
- Compile and Upload
- Open Serial Monitor and choose the correct baud rate as that of the program

Output: The Node will be sending and receiving replies from the Gateway

```
74 void loop()
75 {
76     Serial.println("Sending to LoRa Server");
77     // Send a message to LoRa Server
78     uint8_t data[] = "Hello, this is device 2";
79     rf95.send(data, sizeof(data));
80
81     rf95.waitPacketSent();
82     // Now wait for a reply
83     uint8_t buf[RH_RF95_MAX_MESSAGE_LEN];
84     uint8_t len = sizeof(buf);
```

Project Implementation

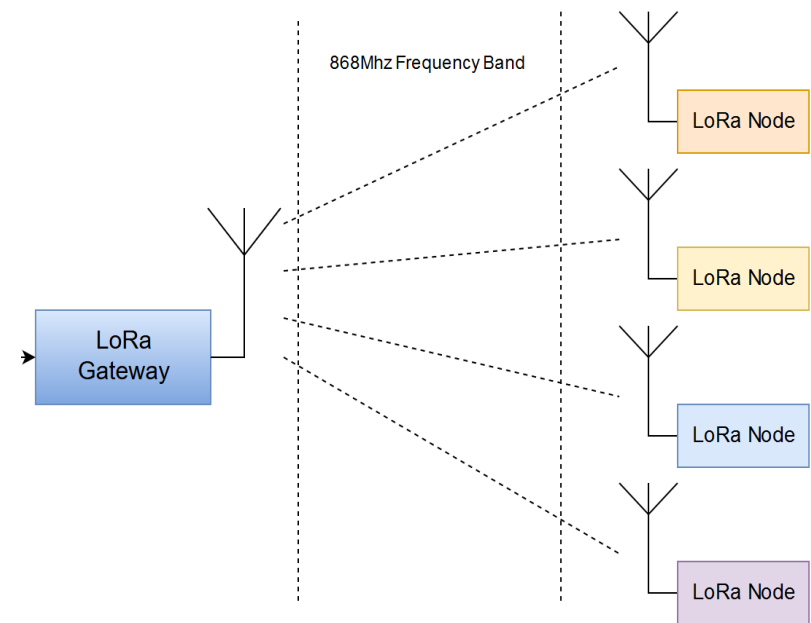
Smart Home Monitoring and Control

IoT Network Configuration

- There are two LoRa nodes and one LoRa gateway
 - LoRa nodes are connected with sensors.

- LoRa1- ESP8266 with **temperature** & **humidity** sensor
- LoRa2- ESP8266 with **Light** sensor

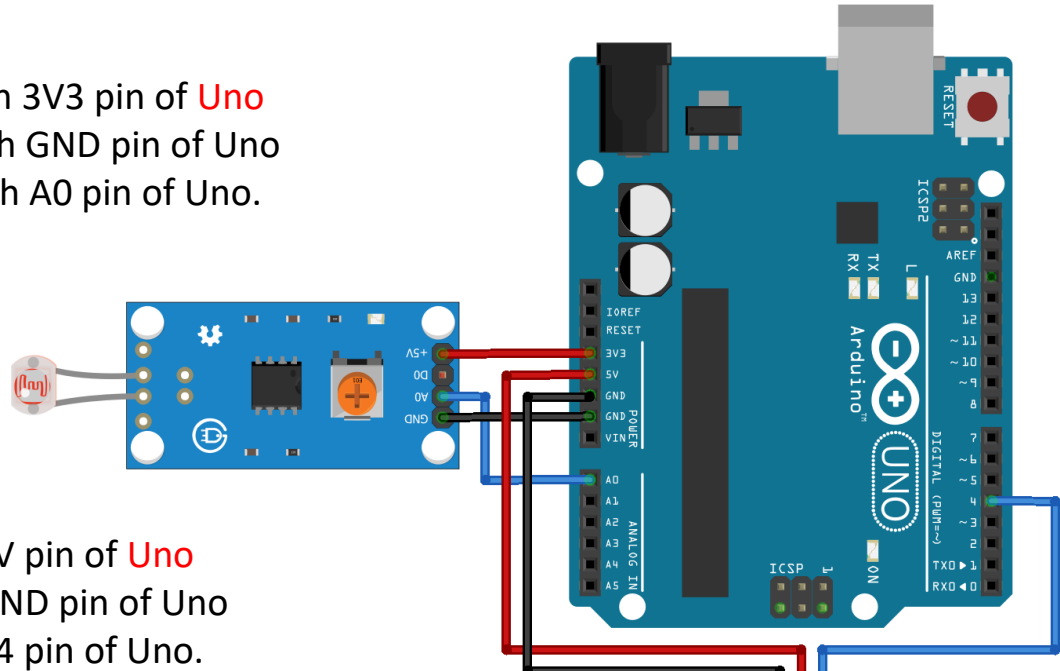
Note: Unique ID for each LoRa node will be needed in programming



Configure LoRa with LDR Sensor & 5V Relay

Arduino Uno with LDR Sensor

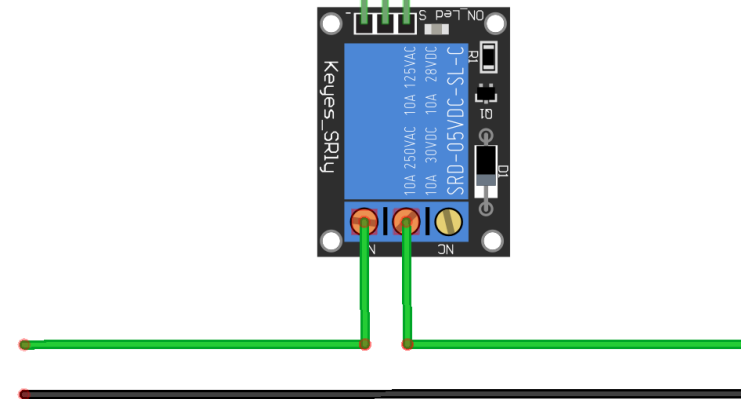
- Connect +5V pin of LDR sensor with 3V3 pin of Uno
- Connect GND pin of LDR sensor with GND pin of Uno
- Connect A0 pin of LDR sensor with A0 pin of Uno.



Arduino Uno with 5V Relay

- Connect VCC pin of Relay with 5V pin of Uno
- Connect GND pin of Relay with GND pin of Uno
- Connect IN pin of Relay with D4 pin of Uno.

- Connect Board to PC
- Open ArduinoIDE
- Select board as "Arduino UNO"
- Select appropriate Port
- Compile and Upload the "client1.ino" Program



client1.ino Programme



```
#include <SPI.h>                                //load required libraries
#include <RH_RF95.h>
int ac = 4;                                     //digital pin 4 to connect relay
int client = 2;                                 //client id
unsigned long previousMillis = 0;
const long interval = 30000;
RH_RF95 rf95;

//duration to send sensor value

void setup()
{
  Serial.begin(9600);                           //Serial connection baud rate
  Serial.println("Start LoRa Client");
  if (!rf95.init())
    Serial.println("init failed");
  rf95.setFrequency(868.0);                     //LoRa frequency
  rf95.setTxPower(13);                          //LoRa TX power
  rf95.setSpreadingFactor(7);                   //Spreading factor
  rf95.setSignalBandwidth(125000);              //Channel Bandwidth
  rf95.setCodingRate4(5);
  rf95.setSyncWord(0x34);                      //LoRa sync word
  pinMode(ac, OUTPUT);                          //digital pin mode
  sendUpdates();
}
```

Cont...

void loop()

```
{
  unsigned long currentMillis = millis();
  if (currentMillis - previousMillis >= interval) {
    previousMillis = currentMillis;
    sendUpdates();
  }
  receiveCommand();
}
```

void receiveCommand()

```
{
  //function to receive commands
  if (rf95.available()) {
    uint8_t buf[RH_RF95_MAX_MESSAGE_LEN];
    uint8_t len = sizeof(buf);
    if (rf95.recv(buf, &len)) {
      Serial.print("Client: ");
      Serial.println(buf[0] - 48);
      if (buf[0] - 48 == client) {
        //verify client id
        Serial.print("Relay Status: ");
        Serial.println(buf[1] - 48);
        String relay = String(buf[1] - 48);
        digitalWrite(ac, relay.toInt()); //setting relay status
        delay(300);
        sendUpdates(); //send back acknowledgement
      }
      else
        Serial.println("Message Not for Me");
    }
  }
}
```

Cont...

void sendUpdates()

```
{  
    //function for sending updates  
    Serial.println("Sending to LoRa Server");  
    String temp = String(client);  
    temp += String(getACstate());  
    temp += String(getSensorData());  
    uint8_t data[temp.length() + 1];  
    temp.getBytes(data, temp.length() + 1);  
    rf95.send(data, temp.length());  
    rf95.waitPacketSent();  
}
```

int getSensorData()

```
{  
    //function for reading sensor  
    int sensorValue = analogRead(A0);  
    return sensorValue;  
}
```

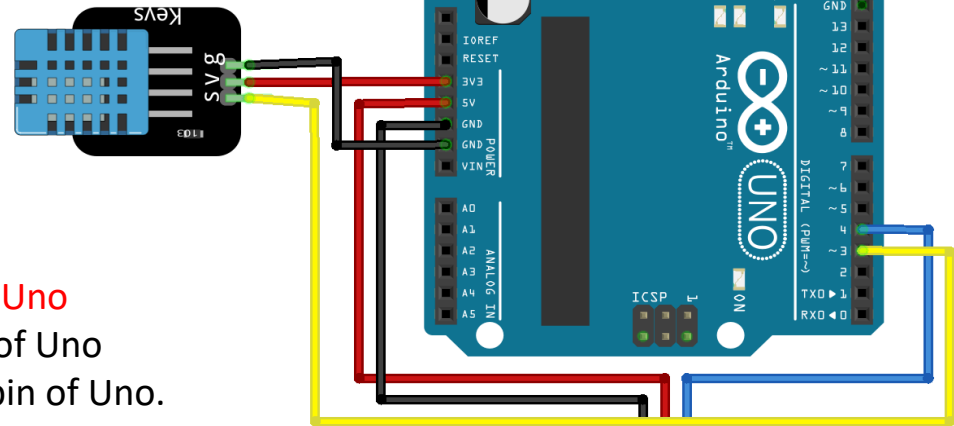
int getACstate()

```
{  
    //function for reading pin status  
    int acState = digitalRead(ac);  
    return acState;  
}
```

Configure GPS/LoRa with DHT11 Sensor & 5V Relay

Arduino Uno with DHT11 Sensor

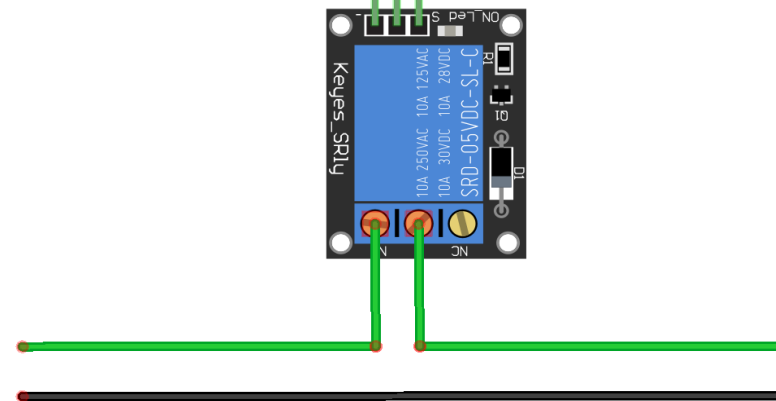
- Connect VCC pin of DHT11 sensor with 3V3 pin of Uno
- Connect DATA OUT pin of DHT11 sensor with D3 pin of Uno
- Connect GND pin of DHT11 sensor with GND pin of Uno.



Arduino Uno with 5V Relay

- Connect VCC pin of Relay with 5V pin of Uno
- Connect GND pin of Relay with GND pin of Uno
- Connect IN pin of Relay with Digital 4 pin of Uno.

- Connect Board to PC
- Open ArduinoIDE
- Select board as "Arduino UNO"
- Select appropriate Port
- Compile and Upload the "client2.ino" Program



client2.ino Programme



```
#include <SPI.h>                                //load required libraries
#include <RH_RF95.h>
#include "DHT.h"                                //DHT library
#define DHTPIN 3                               //DHT to digital pin 3
#define DHTTYPE DHT11                         //DHT11 selection
int ac = 4;                                    //digital pin 4 to connect relay
int client = 3;                                //client id

unsigned long previousMillis = 0;
const long interval = 30000;                  //duration to send updates
RH_RF95 rf95;
DHT dht(DHTPIN, DHTTYPE);

void setup()
{
    Serial.begin(9600);                        //Serial baud rate
    dht.begin();
    Serial.println("Start LoRa Client");
    if (!rf95.init())
        Serial.println("init failed");
    rf95.setFrequency(868.0);                  //LoRa frequency
    rf95.setTxPower(13);                       //TX power
    rf95.setSpreadingFactor(7);                //Spreading Factor
    rf95.setSignalBandwidth(125000);           //Signal Bandwidth
    rf95.setCodingRate4(5);
    rf95.setSyncWord(0x34);
    pinMode(ac, OUTPUT);                       //digital pin mode
    digitalWrite(ac, HIGH);                   //reverse for active low relay
    sendUpdates();
}
```

Cont...



```
void loop()
```

```
{  
  unsigned long currentMillis = millis();  
  if (currentMillis - previousMillis >= interval) {  
    previousMillis = currentMillis;  
    sendUpdates();  
  }  
  receiveCommand();  
}
```

```
void receiveCommand()
```

```
{ //function to receive commands  
  if (rf95.available()) {  
    uint8_t buf[RH_RF95_MAX_MESSAGE_LEN];  
    uint8_t len = sizeof(buf);  
    if (rf95.recv(buf, &len))  
      if (buf[0] - 48 == client) {  
        String relay = String(buf[1] - 48);  
        digitalWrite(ac, !relay.toInt());  
        delay(300);  
        sendUpdates();  
      }  
  }  
}
```

void sendUpdates()

```
{  
    //function to send sensor values  
    String temp = String(client);  
    temp += String(getACstate());  
    temp += String(getSensorData());  
    uint8_t data[temp.length() + 1];  
    temp.getBytes(data, temp.length() + 1);  
    rf95.send(data, temp.length());  
    rf95.waitPacketSent();  
}
```

float getSensorData()

```
{  
    //function to read sensor data  
    float sensorValue = dht.readTemperature();  
    Serial.println(sensorValue);  
    return sensorValue;  
}
```

int getACstate()

```
{  
    //function to read relay state  
    int acState = !digitalRead(ac);  
    return acState;  
}
```


Configure LoRa Gateway



- Connect LoRa gateway to PC
- Open Arduino IDE
- Select board as “Dragino Yun + UNO or LG01/OLG01”
- Select network port at 10.130.1.1
- Update server IP address (as set in server configuration) in the Gateway program
- Compile and Upload the “[gateway.ino](#)” Program

gateway.ino Programme



```
#define BAUDRATE 115200           //Define baud rate of bridge.
#include <Console.h>              //Bridge is the intermediate between
#include <SPI.h>                  //the Linux system and the
#include <RH_RF95.h>              //microcontroller
#include <Process.h>
#include <String.h>              //Include all required libraries
RH_RF95 rf95;
unsigned long previousMillis = 0;
const long interval = 2000;
float frequency = 868.0;         //Listen frequency
String old;
String IP = "10.130.1.229";      //Server IP address
char client;
```

void setup()

```
{
  Bridge.begin(BAUDRATE);
  Console.begin();
  if (!rf95.init())
    Console.println("init failed");
  rf95.setFrequency(frequency);
  rf95.setTxPower(13);           //TX power
  rf95.setSpreadingFactor(7);    //Spreading factor
  rf95.setSignalBandwidth(125000); //Signal Bandwidth
  rf95.setCodingRate4(5);
  rf95.setSyncWord(0x34);
}
```

void loop()

```
{
  unsigned long currentMillis = millis();
  if (currentMillis - previousMillis >= interval) {
    previousMillis = currentMillis;
    int command = checkforCommand() - 48;
    String neww = String(command);
    neww += String(client);
    if (old != neww) //send only if new command arrives
      sendCommand(command);
  }
  receiveUpdates();
}
```

Cont...

```
void sendCommand (int relaystate)
{
    //function to send command to node
    old = String(relaystate);
    old += String(client);
    String temp = String(client);
    temp += String(relaystate);
    uint8_t data[temp.length() + 1];
    temp.getBytes(data, temp.length() + 1);
    rf95.send(data, sizeof(data));
    rf95.waitPacketSent();
    Console.println("Sent a command");
}

void receiveUpdates()
{
    //function to receive data from nodes
    if (rf95.available()) {
        uint8_t buf[RH_RF95_MAX_MESSAGE_LEN];
        uint8_t len = sizeof(buf);
        if (rf95.recv(buf, &len)) {
            String sensor = "";
            for (int i = 2; i < len; i++) {
                sensor += buf[i] - 48;
            }
            uploadData(buf[0] - 48, buf[1] - 48, sensor);
        }
    }
}
```

Cont...

```
void uploadData(int client, int relay, String sensor)
{
    Process p;                //initialize linux process
    String URL = "http://";    //linux command with values
    URL += IP;
    URL += "/receive.php?client=";
    URL += String(client);
    URL += "&access=";
    URL += String(relay);
    URL += "&sensor=";
    URL += String(sensor);
    p.begin("curl");
    p.addParameter(URL);
    p.run();    //execute linux command
}
```

```
char checkforCommand()
{
    //function for checking from server
    char command[2] = {0};
    int count = 0;
    String URL = "http://";
    URL += IP;
    URL += "/new/";
    Process p;                //linux process to fetch command
    p.begin("curl");
    p.addParameter(URL);
    p.run();
    while (p.available() > 0){ //if linux command outputs command
        command[count] = p.read();
        count++;
    }
    client = command[0];
    return command[1];        //return client address and access
}
```

Configure Web Server

send.php

```
=====
<html>
<body>
<?php
    $client = $_GET["client"];
    $access = $_GET["access"];
    $file = fopen('new/index.html', 'w+');
    ftruncate($file, 0);
    $content = $client.$access. PHP_EOL;
    fwrite($file , $content);
    fclose($file );
    die(header("Location: ".$_SERVER["HTTP_REFERER"]));
?>
</body>
</html>
```

receive.php

```
=====
<html>
<body>
<?php
    $client = $_GET["client"];
    $access = $_GET["access"];
    $sensor = $_GET["sensor"];
    $var = $client.'/index.html';
    $file = fopen($var, 'w+');
    ftruncate($file, 0);
    $content = $access.$sensor. PHP_EOL;
    fwrite($file , $content);
    fclose($file );
    die(header("Location: ".$_SERVER["HTTP_REFERER"]));
?>
</body>
</html>
```

Cont...

[index.html](#)

```
=====
<html>
<head></head>
<body onload="init()">
<center>
  <div style="width:450px;border:3px solid black;padding:20px">
    <h1>LoRa AC Control</h1>
    <div id="main">
      <div id="updateMe">
        <div style="width:400px;border: 3px solid black;padding:20px">
          <h2>UNO with LoRa Status:</h2>
          <h3 id="uno2"></h3>
          <h3 id="uno21"></h3>
          <form action="send.php" method="GET" target="hidden-form">
            <input type="hidden" name="client" value="2">
            <button type="submit" onclick="shows();setTimeout(hides,3000)" name="access" value="1" style="padding:10px" id="button1">TURN ON</button>
          </form>
          <div id="stat" style="display:none"></div>
        </div>
        <div style="border: 5px;padding:20px"></div>
        <div style="width:400px;border: 3px solid black;padding:20px">
          <h2>UNO with LoRa/GPS Status:</h2>
          <h3 id="uno3"></h3>
          <h3 id="uno31"></h3>
          <form action="send.php" method="GET" target="hidden-form">
            <input type="hidden" name="client" value="3">
            <button type="submit" onclick="showss();setTimeout(hidess,3000)" name="access" value="1" style="padding:10px" id="button2">TURN ON</button>
          </form>
          <div id="stat2" style="display:none"></div>
        </div>
      </div>
    </div>
  </div>
</center>
</body>
```

Cont...

```
<IFRAME style="display:none" name="hidden-form"></IFRAME>
<script type="text/javascript">
```

```
function refresh() {
    var req = new XMLHttpRequest();
    console.log("Grabbing Value");
    req.onreadystatechange = function () {
        if (req.readyState == 4 && req.status == 200) {
            document.getElementById('uno2').innerText = "Relay Status: "+req.responseText[0];
            document.getElementById('uno21').innerText = "Sensor Value: "+req.responseText[1]+req.responseText[2]+req.responseText[3]+" lux";
        }
    }
    req.open("GET", '2/', true);
    req.send(null);
}

function refresht() {
    var req = new XMLHttpRequest();
    console.log("Grabbing Value");
    req.onreadystatechange = function () {
        if (req.readyState == 4 && req.status == 200) {
            document.getElementById('uno3').innerText = "Relay Status: "+req.responseText[0];
            document.getElementById('uno31').innerText = "Sensor Value: "+req.responseText[1]+req.responseText[2]+". "+req.responseText[4]+req.responseText[5]+"
            \u2103";
        }
    }
    req.open("GET", '3/', true);
    req.send(null);
}
```

Cont...

```
function init()
{
    refresh()
    refresh()
    update2()
    update3()
    var int = self.setInterval(function () {
        refresh()
        refresh()
        update2()
        update3()
    }, 1000);
}

function shows(){
    var x = document.getElementById("stat");
    x.innerHTML = "Sending.";
    if(x.style.display == "none"){
        x.style.display = "block";
    }
    setTimeout(function(){x.innerHTML = "Sending.."},1000);
    setTimeout(function(){x.innerHTML = "Sending..."},2000);
}

function hides(){
    var x = document.getElementById("stat");
    if(x.style.display == "block"){
        x.style.display = "none";
    }
    var x2 = document.getElementById("button1");
    if(x2.innerHTML == "TURN ON"){
        x2.innerHTML = "TURN OFF";
        x2.value = "0";
    }
    else{
        x2.innerHTML = "TURN ON"
        x2.value = "1";
    }
}
```

```
function showss(){
    var x = document.getElementById("stat2");
    x.innerHTML = "Sending.";
    if(x.style.display == "none"){
        x.style.display = "block";
    }
    setTimeout(function(){x.innerHTML = "Sending.."},1000);
    setTimeout(function(){x.innerHTML = "Sending..."},2000);
}

function hidess(){
    var x = document.getElementById("stat2");
    if(x.style.display == "block"){
        x.style.display = "none";
    }
    var x2 = document.getElementById("button2");
    if(x2.innerHTML == "TURN ON"){
        x2.innerHTML = "TURN OFF";
        x2.value = "0";
    }
    else{
        x2.innerHTML = "TURN ON"
        x2.value = "1";
    }
}
```

```
function update2(){
    var req = new XMLHttpRequest();
    var x2 = document.getElementById("button1");
    console.log("Grabbing Value");
    req.onreadystatechange = function () {
        if (req.readyState == 4 && req.status == 200) {
            if (req.responseText[0] == "1"){
                x2.innerHTML = "TURN OFF";
                x2.value = "0";
            }
            else{
                x2.innerHTML = "TURN ON";
                x2.value = "1";
            }
        }
    }
    req.open("GET", '2/', true);
    req.send(null);
}

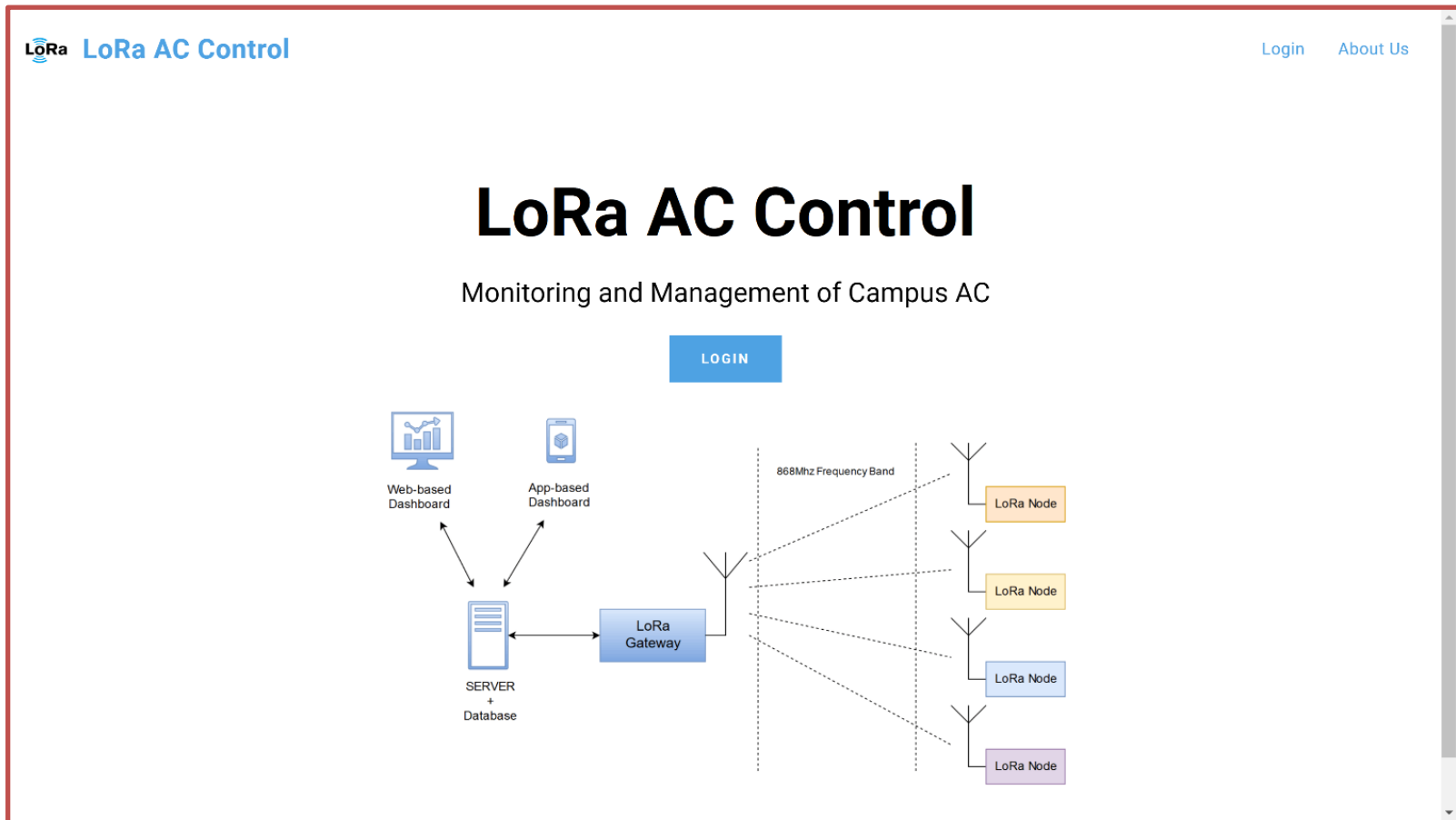
function update3(){
    var req = new XMLHttpRequest();
    var x2 = document.getElementById("button2");
    console.log("Grabbing Value");
    req.onreadystatechange = function () {
        if (req.readyState == 4 && req.status == 200) {
            if (req.responseText[0] == "1"){
                x2.innerHTML = "TURN OFF";
                x2.value = "0";
            }
            else{
                x2.innerHTML = "TURN ON";
                x2.value = "1";}}
        }
    }
    req.open("GET", '3/', true);
    req.send(null);
}

</script>
</html>
```


Observe Outputs

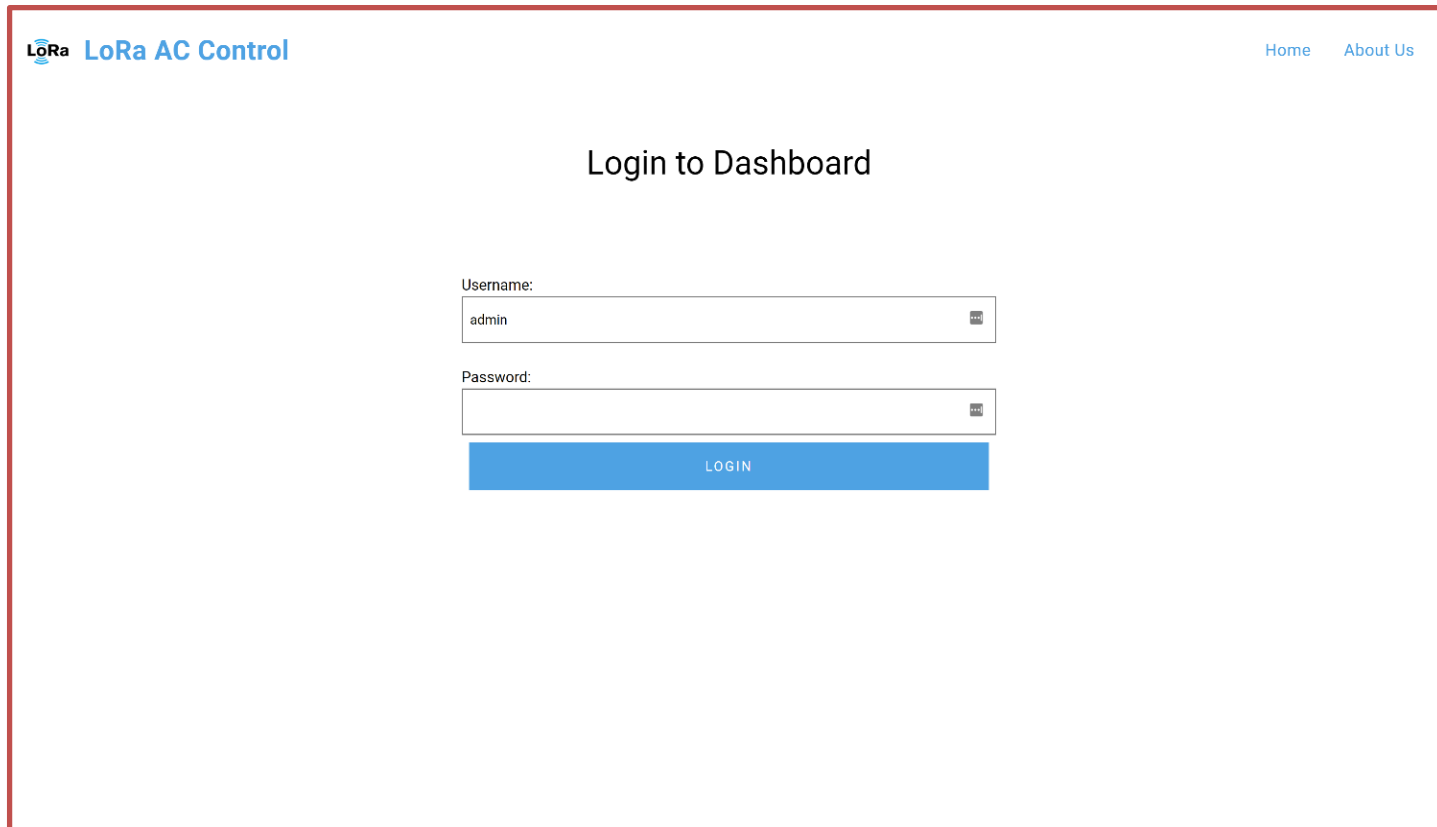
Access & Control from Web

- Check proper connection by going to **localhost** in a web browser.
- Dashboard should be opened as shown below:



Cont...

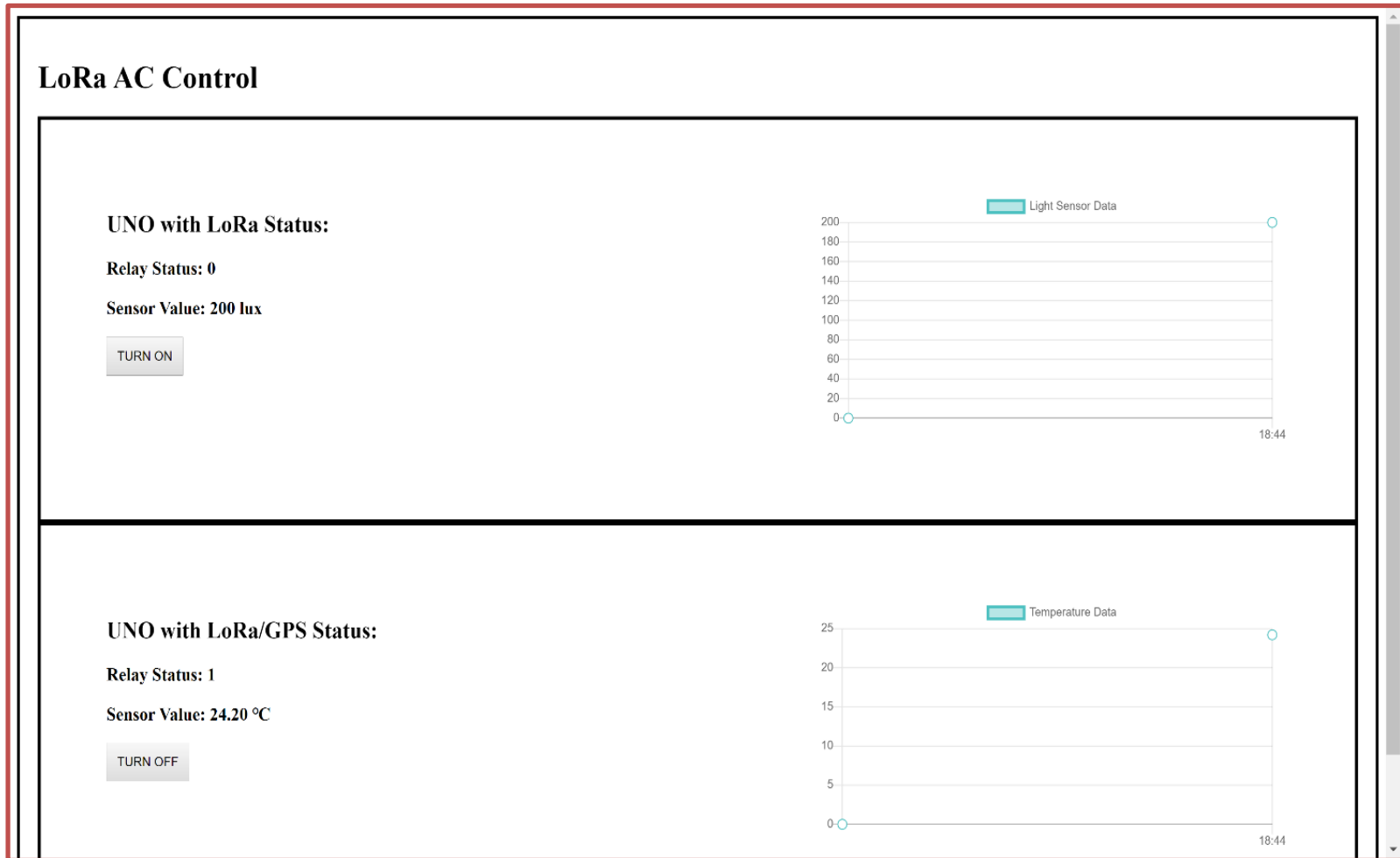
- Login to web service:
 - Username: **admin**; Password: **123**



The screenshot displays the 'LoRa AC Control' web interface. At the top left is the logo and text 'LoRa AC Control'. At the top right are links for 'Home' and 'About Us'. The main heading is 'Login to Dashboard'. Below this, there are two input fields: 'Username:' with the value 'admin' and 'Password:'. A blue 'LOGIN' button is positioned below the password field.

Cont...

- Dashboard of the application:



Thanks!

