```
#include "Adafruit_FONA.h"


#define FONA_RX 2
#define FONA_TX 3
#define FONA_RST 4


#include <SoftwareSerial.h>
SoftwareSerial fonaSS = SoftwareSerial(FONA_TX, FONA_RX);
SoftwareSerial *fonaSerial = &fonaSS;



Adafruit_FONA fona = Adafruit_FONA(FONA_RST);

int measurePin = A0;
float Voltage;

float Measured = 0;

float average = 0;
String location;

long time;
int i=0;
long Int1=0;
long lastVal1 = 0;
 float conc;

void status_gsm()
{
  Serial.println("AT");
  fonaSS.println("AT");
  if(fonaSS.find("OK"))
  {
    Serial.println("Connected");
  }
  else
  {
    Serial.println("Connection failed");
    status_gsm();
  }
  delay(100);
}

 void setting_SIM()
{
  Serial.println(("AT+CREG=1")); //  Registration status
  fonaSS.println(("AT+CREG=1"));
  delay(500);
  Serial.println(fonaSS.readString());

  Serial.println("AT+CIPMODE=0"); // Selecting TCPIP application mode. '0' os non-transparent mode
  fonaSS.println("AT+CIPMODE=0");
  delay(300);
  Serial.println(fonaSS.readString());

  Serial.println(("AT+CIPSHUT"));// Close the GPRS PDP context
  fonaSS.println(("AT+CIPSHUT"));
  delay(1000);
```

```
  Serial.println(fonaSS.readString());

  Serial.println(("AT+CIPMUX=0")); // Value can be '0' (single IP)
  fonaSS.println(("AT+CIPMUX=0"));
  delay(500);
  Serial.println(fonaSS.readString());

  Serial.println(("AT+CGATT=1")); // Attach or detach the device to packet domain service.
  fonaSS.println(("AT+CGATT=1"));
  delay(500);
  Serial.println(fonaSS.readString());

  Serial.println(("AT+CSTT=\"WHOLESALE\"")); // Apn, user name and password for the PDP context.
  fonaSS.println(("AT+CSTT=\"WHOLESALE\"")); // +CSTT: "APN","USER","PWD"
  delay(3000);
  fonaSS.println(("AT+CSTT?"));
  delay(3000);
  Serial.println(fonaSS.readString());

  Serial.println("AT+SAPBR=3,1,\"APN\",\"wholesale\"");
  fonaSS.println("AT+SAPBR=3,1,\"APN\",\"wholesale\"");//setting the APN, the second need you fill in your local apn server
  delay(1000);
  Serial.println(fonaSS.readString());


  Serial.println(("AT+CIICR")); // Brings up the GPRS
  fonaSS.println(("AT+CIICR"));
  delay(3000);
  Serial.println(fonaSS.readString());

  Serial.println(("AT+CIFSR"));
  fonaSS.println(("AT+CIFSR")); //request IP
  delay(1000);
  Serial.println(fonaSS.readString());

}

void setup()
{
  fonaSS.begin(4800);
  Serial.begin(115200);
  delay(2000);
  Serial.print("Status:");
  status_gsm();  // Check connection
  setting_SIM(); // Setup procedure

  while (! Serial);
  Serial.println("Initializing FONA");

  fonaSerial->begin(4800);
  if (! fona.begin(*fonaSerial)) {
    Serial.println(F("Finding FONA"));
    while(1);
  }

  Serial.println("Enabling GPS");
  fona.enableGPS(true);

}

void connection()
{
```

```
delay(2000);

 float latitude, longitude,altitude; // Speed_kph, heading can also be measured

 boolean gps_success = fona.getGPS(&latitude, &longitude,&altitude);

 if (gps_success) {

   Serial.print("GPS lat:");
   Serial.println(latitude, 6);
   Serial.print("GPS long:");
   Serial.println(longitude, 6);
   Serial.print("GPS altitude:");
   Serial.println(altitude);

 }

 Serial.println(("AT+CIPSTART=\"TCP\",\"129.118.34.225\",\"80\""));
 fonaSS.println(("AT+CIPSTART=\"TCP\",\"129.118.34.225\",\"80\"")); // Comamnds starts a TCP or UDP connection.r
 delay(8000);
 Serial.println(fonaSS.readString());
 delay(1000);


 Serial.println(("AT+CIPSEND")); //  AT command is used to send the data over the TCP or UDP connection
 fonaSS.println(("AT+CIPSEND"));
 delay(5000);


 Serial.print("GET /LIGHT/index.php?Longitude=" + String(longitude,8) + "&Latitude=" + String(latitude,8) +
"&Intensity="+String(Measured,6));
 fonaSS.print("GET /LIGHT/index.php?Longitude=" + String(longitude,8) + "&Latitude=" + String(latitude,8) +
"&Intensity="+String(Measured,6));
 delay(8000);
 Serial.println(fonaSS.readString());
 delay(1000);

 Serial.print(" HTTP/1.1\r\n");
 fonaSS.print(" HTTP/1.1\r\n");
 Serial.println(fonaSS.readString());
 delay(1000);

 Serial.print("Host: 129.118.34.225\r\n");
 fonaSS.print("Host: 129.118.34.225\r\n");
 Serial.println(fonaSS.readString());
 delay(1000);


 fonaSS.print("Connection: keep-alive");
 fonaSS.print("Connection: close");
 fonaSS.print("\r\n");
 fonaSS.print("\r\n");
 fonaSS.println();
 Serial.println(fonaSS.readString());
 delay(1000);

 fonaSS.println((char)26);
 delay(1000);

  if(fonaSS.find("SEND OK"))
  {
```

```
    delay(1000);
    while (fonaSS.available())
    {
      Serial.println(fonaSS.readString()); //feedback data
    }
    fonaSS.println(("AT+CIPCLOSE"));
  }
  else
  {
    connection();
  }

}



void loop() {


    unsigned long currentTime = millis();

 if (currentTime - lastVal1 < Int1) {
  lastVal1 = currentTime;
  Serial.print("Timer of    ");
  Serial.print(lastVal1 / 1000);
  Serial.println(" Sec");
  //Task 1
 }
 Measured = analogRead(measurePin);



Serial.print(Measured);
  delay(5000);
  connection();
  delay(1000);

}
```