User Manual

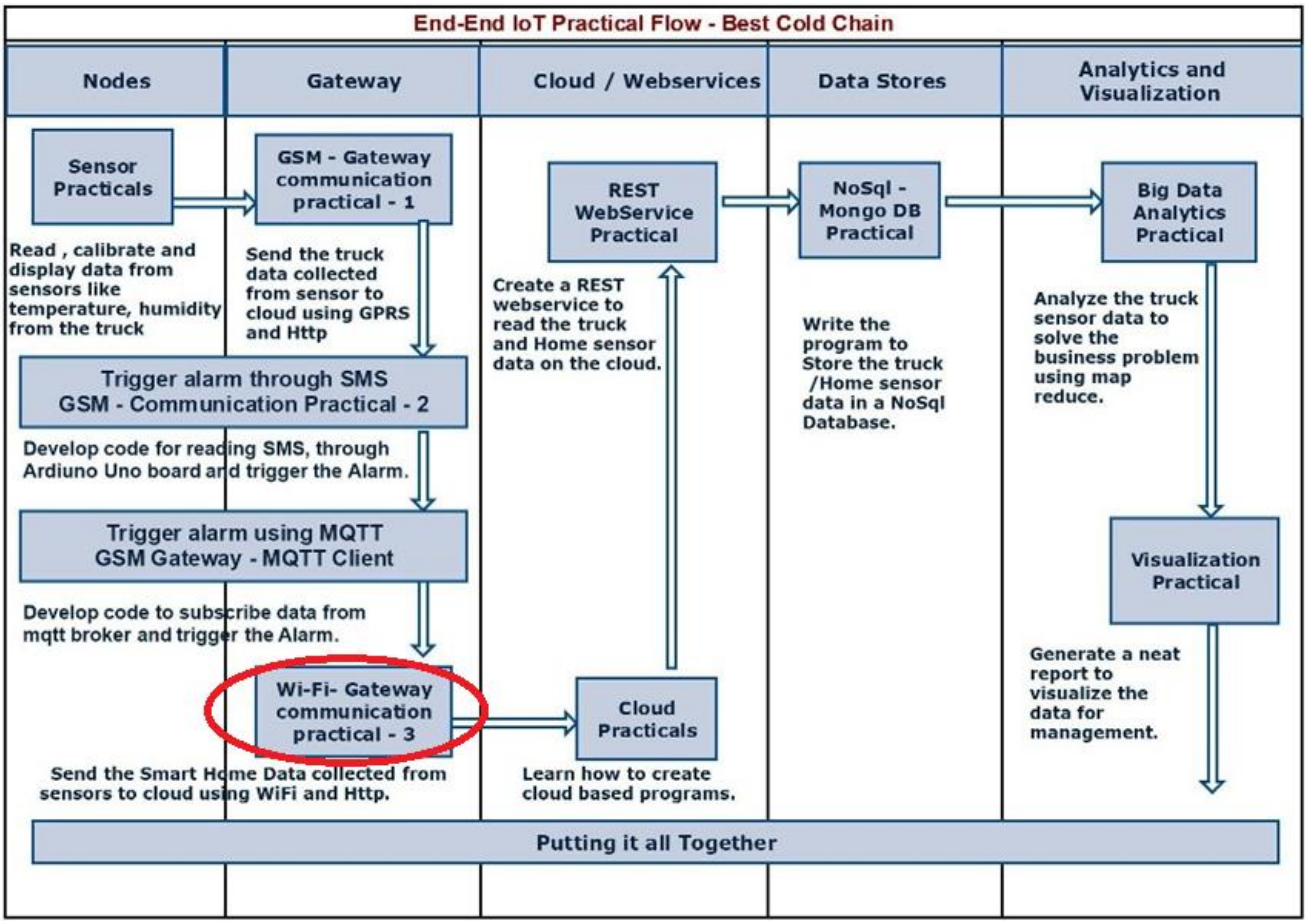
Communication Practical – 3

Send sensor data of Home Automation to cloud using WiFi

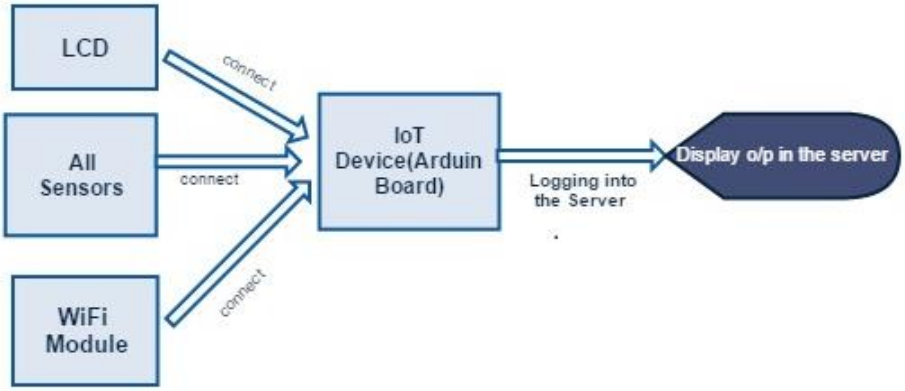
Practical's Objective:

To send sensor data of Best Cold Chain to cloud using WiFi and HTTP

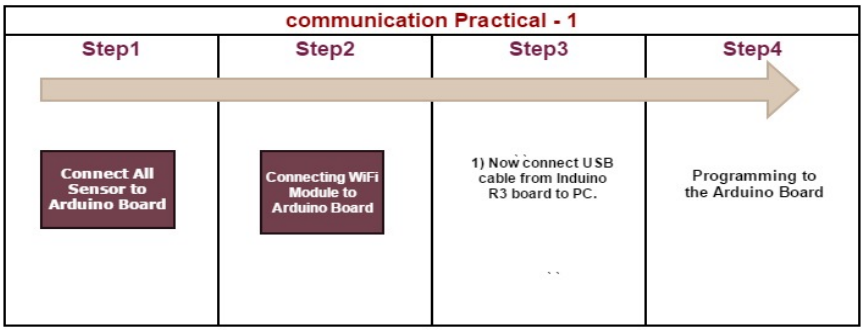
1. **End-End IoT Flow Diagram:**



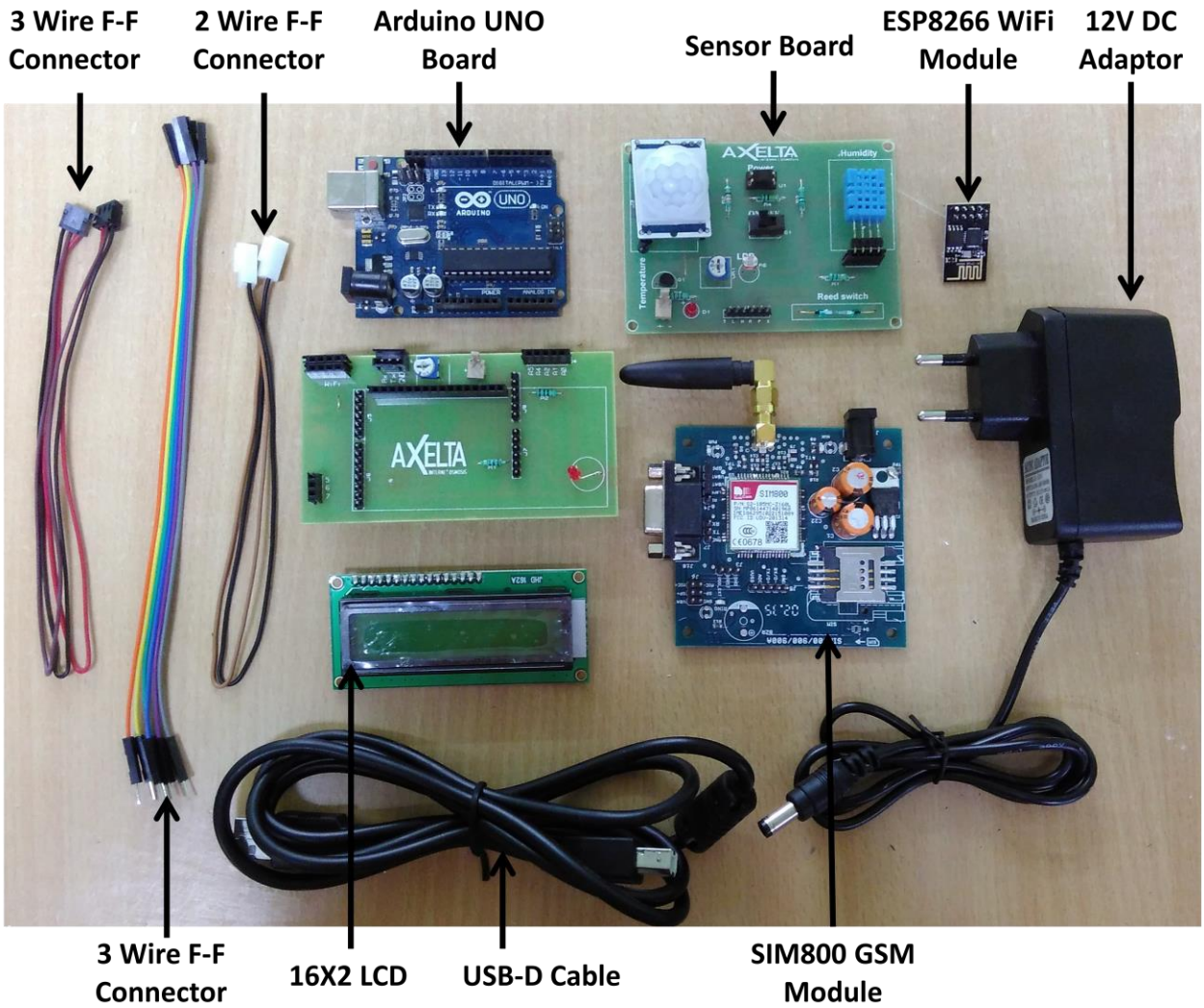
2. Sensors Data Posting Using JSON Hardware Flow Diagram:



3. Sensors Data Posting Using JSON To Do:



4. **Hardware requirements:**

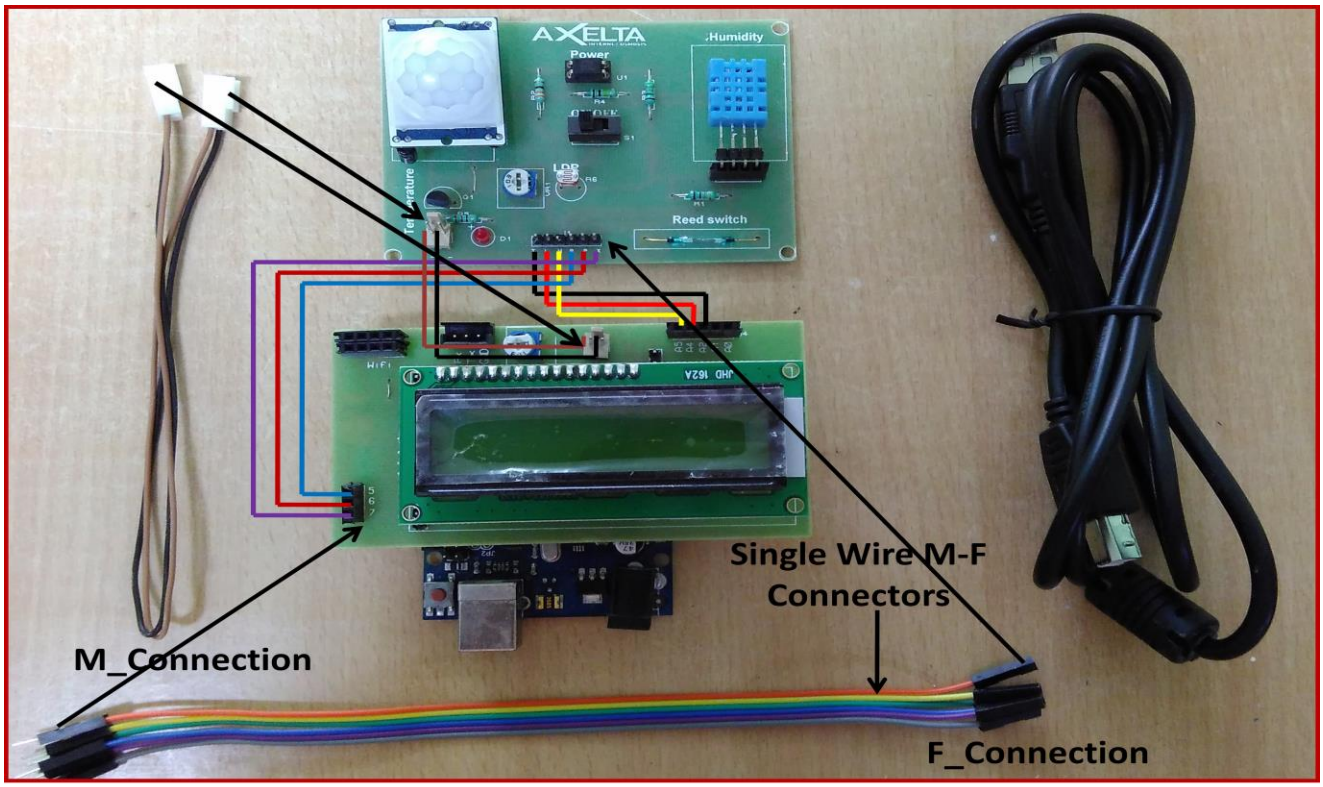


Software Requirement:

- Arduino IDE
- DHT-11 Library

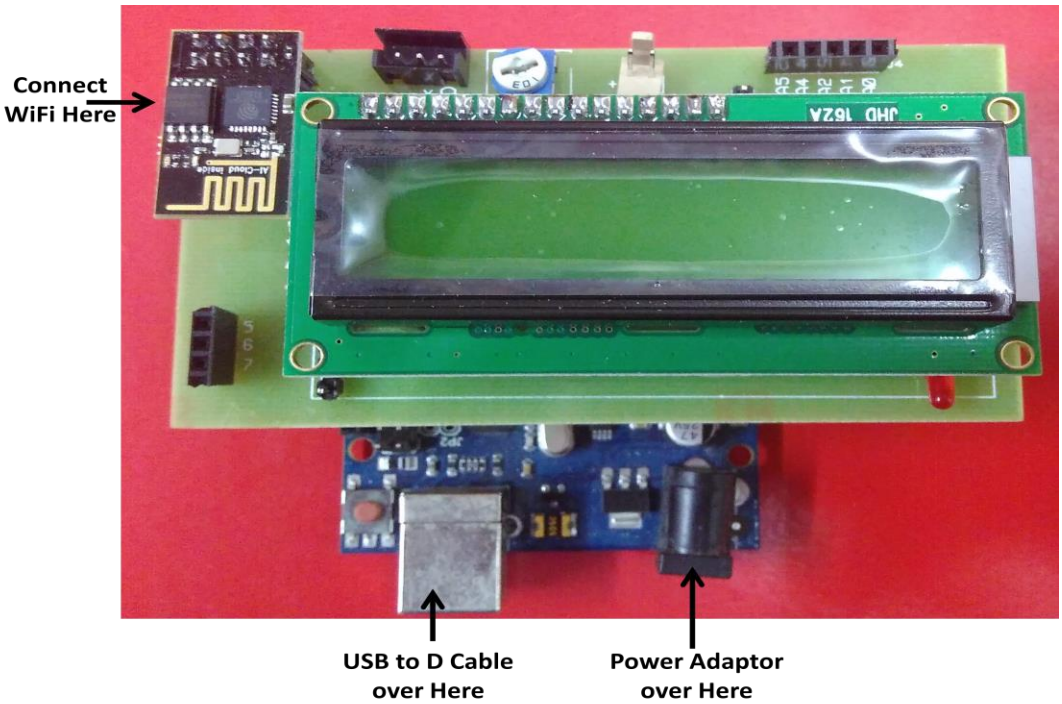
Arduino UNO Board Connections with Sensor Board:

Don't change the sensor connections. They will also remain same.



Note:

1) First remove previous connections of GSM Board (Tx Rx & GND) from the Shield.
2) **Don't connect WiFi Module ESP8266 to the J9 on LCD Shield unless you give 12V Supply to the Arduino Board.**
3) After plugging ESP8266 as shown below connect your PC to Arduino Board via USB Cable.

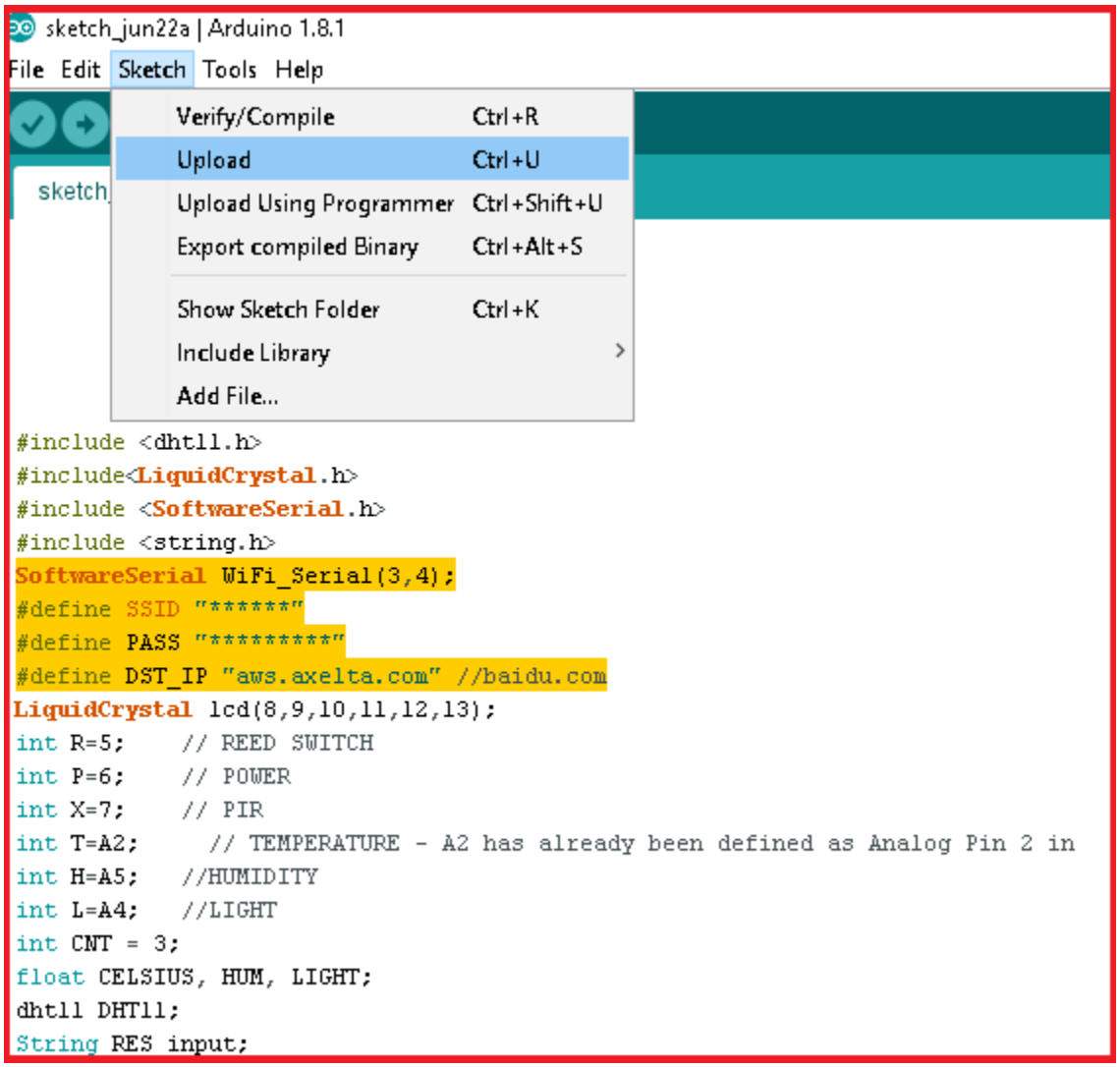


4) Make sure that all the Hardware connections are proper as explained above.

Programming:

5) You can find WiFi code "**Communication practical3_code.txt**" by the link: https://drive.google.com/file/d/0Bz7GE98wyjOxQ0tIT0RZMkw3Y1U/view
6) Go to **File** menu, and Click on **save**, give a File name and click **ok**.

7) Copy, paste & upload the code in Arduino board



If it is uploaded successfully then you can see its output on Serial Monitor. But before that Open the serial monitor & set correct **baud rate as 115200. Correct O/P is shown as below** :

```
PWR STATUS      P:ON    REED STATUS R:OPEN      PIR STATUS X:YES
Geeting Temp    T:      27.86   Getting Hum H: 82.00    Getting Lig L: 69.40
PWR STATUS      P:ON            REED STATUS R:OPEN      PIR STATUS X:NO

Geeting Temp    T:      27.37   Getting Hum H: 82.00    Getting Lig L: 70.87
PWR STATUS      P:ON            REED STATUS R:OPEN      PIR STATUS X:NO

Geeting Temp    T:      27.37   Getting Hum H: 82.00    Getting Lig L: 68.43
PWR STATUS      P:ON            REED STATUS R:OPEN      PIR STATUS X:NO

WIFI NOT PLUGGED.

PLUG IN YOUR WIFI CHIP
```

When you Plug in ESP 8266 (WiFi Chip)
Then only it try for to connect to WiFi Network

```
Trying to Connect to WiFi Network
AT+CWJAP="******","*********"

OK
AT+CWJAP="******","*********"

UNABLE TO CONNECT TO WIFI NETWORK..!
```

Make changes in your program
#define SSID
#define PASS

You will get this Alert on Serial Monitor in case you forget to edit your SSID & PASSWORD in program

```
Geeting Temp     T:       175.46  Getting Hum H: 0.00     Getting Lig L: 72.03
PWR STATUS       P:ON     REED STATUS R:CLOSE     PIR STATUS X:NO

WIFI PLUGGED ON TO THE BOARD..!


NOT CONNECTED TO WIFI NETWORK


Trying to Connect to WiFi Network
AT+CWJAP="Axelta","Axelta140218"


OK

CONNECTED TO WIFI NETWORK..!
Geeting Temp     T:       186.22  Getting Hum H: 0.00     Getting Lig L: 114.92
PWR STATUS       P:ON     REED STATUS R:CLOSE     PIR STATUS X:NO
```

SSID

PASSWORD

This shows successful connection to WiFi N/W

```
COM11 (Arduino/Genuino Uno)

Geeting Temp     T:       202.35  Getting Hum H: 0.00     Getting Lig L: 135.97
PWR STATUS       P:ON     REED STATUS R:CLOSE     PIR STATUS X:NO

WIFI PLUGGED ON TO THE BOARD..!


CONNECTED TO WIFI NETWORK..!


{"device_no":"WKSP-01","client":"Workshop","device_type":"SAM","device_key":"LUJPIOJV7L5NHCWQ1
POST /services/data HTTP/1.1
Host: aws2.axelta.com:80
Accept: */*
Content-Length: 215

Content-Type: application/json



AT+CIPSTART="TCP","aws2.axelta.com",80
CONNECT

OK

AT+CIPSEND=
OK
>

Recv 339 bytes

SEND OK

+IPD,128:HTTP/1.1 200 OK
Date:
<
Autoscroll                                                    Both NL & CR
```

JSON Object

Connected to Server

Data posted successfully on Cloud

Finally Sensor Data is posted to Server; output can be seen by logging to server.

Steps for constructing a simple 'C' program in Arduino.

/*Here we are Sending data to web server using WIFI/GPRS modem and we are interfacing WiFi modem using Uart */

Step1: Create a new sketch

Step2: Include the header file **"LiquidCrystal.h",** to use the functions related to LCD and Include the header file "**dht11.h**", to use the Humidity related function define your SSID and PASSWORD of your WiFi and URL of Cloud.

```
#include <dht11.h>
#include<LiquidCrystal.h>
#include <SoftwareSerial.h>
#define SSID "XXXXXX"
#define PASS "XXXXXXXXXX"
#define DST_IP "aws.axelta.com"
```

Step3: Include the header file **"SoftwareSerial.h",** to use the functions related to UART.

```
#include <SoftwareSerial.h>
```

Step4: Make the global declaration of the UART pins with the microcontroller pins of the Arduino board, i.e., RX, TX with pins 3,4 using "SoftwareSerial".

```
SoftwareSerialWiFi_Serial(3,4);
dht11 DHT11;
```

Step5: Make the global declaration of the LCD pins with the microcontroller pins of the Arduino board, i.e., RS, E, D4, D5, D6, D7, with pins 8, 9, 10, 11, 12, 13 using "LiquidCrystal".

```
LiquidCrystall lcd(8,9,10,11,12,13);
```

Step6: Initialize variables as integers with the pin numbers to which the digital sensors are connected.

```
int R=5;                    // REED SWITCH
int P=6;                    // POWER
int X=7;                    // PIR
```

Step7: Initialize variables as integers with the Analog pin numbers to which the analog sensors are connected.

```
int T= A2;                  // TEMPERATURE - A2 has already been defined as Analog Pin 2 in arduino lib
int H=A4;                   //HUMIDITY
int L=A5;                   //LIGHT
```

Step8: Initialize a global variable "CNT" as integer with a value as 4;

```
int CNT=4;
```

Step9: Initialize strings to store WIFI input & Response.

```
String WiFi_input="";
String RES_input="";
```

Step11: Initialize few global variables **CELSIUS, HUM, LIGHT** as float.

```
float CELSIUS, HUM, LIGHT;
```

Step12: Initialize variables as integers to use them in functions.

```
inti=1;
intStart_chck=0;
```

Step13: Write a **"setup"** function and initialize the UART communication with 9600 baud rate and the LCD as 16 columns with 2 rows with the help of **"lcd.begin"** function and **Serial** with 115200 baud.

```
void setup()
{
    Serial.begin(115200);
    lcd.begin(16,2);
    WiFi_Serial.begin(115200);
    WiFi_Serial.println("AT+UART_DEF=9600,8,1,0,0");
    delay(2000);
    WiFi_Serial.begin(9600);
    WiFi_Serial.println("ATE0");
    delay(200);
    pinMode(P, INPUT);
    pinMode(R, INPUT);
    pinMode(X, INPUT);
}
```

Step14:

14.1 Start a **"WiFi_Check"** function.

14.2 Using **"WiFi_Serial.println"** function, send the command "AT"

14.3 When data available on WiFi serial Read the Response

14.4 Print on LCD

14.5 Now if response matches with "OK" print "WIFI PLUGGED ON TO THE BOARD..!" or print "WIFI NOT PLUGGED..!"

```
void WIFI_Check()
{
    WiFi_Serial.println("AT");
    delay(500);
    if(WiFi_Serial.available())
    {
    if(WiFi_Serial.find("OK"))
    {
    Serial.println("WIFI PLUGGED ON TO THE BOARD..!");
    Serial.println();
    Serial.println();
    WiFi_Serial.println("AT+CWMODE=1");
    delay(500);
    Start_chck=1;
    i=1;
    }
    }
    else
    {
    Serial.println("WIFI NOT PLUGGED..!");
    Serial.println();
    Serial.println("PLUG IN YOUR WIFI CHIP");
    Serial.println();
    Serial.println();
    }
}
```

Step15: WiFi Connection check

15.1 Start a **"connectWiFi"**function

15.2 Using **"WiFi_Serial.println"** function, send the command "AT+CWJAP?" to Check if WiFi connected to Access Point, if not connected WiFI returns "No AP" on serial.

15.3 When find "No AP" Using **"WiFi_Serial.println"** function, send the command "AT+CWJAP" with your SSID and Password of your WiFi to connect to your WiFi Network

15.6 Now if response matches with "WIFI CONNECTED" print "WIFI CONNECTED TO NETWORK" and enter post function if the response doesn't match with "WIFI CONNECTED" then try to connect to AP using same command until it connects to network.

```
void connectWiFi()
    {
    WiFi_Serial.println("AT+CWJAP?");
    delay(5000);
    if(WiFi_Serial.available())
    {
    if(WiFi_Serial.find("No AP"))
    {
    Serial.println("Trying to Connect to WiFi Network");
    String cmd = "AT+CWJAP=\"";
    cmd += SSID;
    cmd += "\",\"";
    cmd += PASS;
    cmd += "\"";
    WiFi_Serial.println(cmd);
    Serial.println(cmd);
    delay(5000);
    if(WiFi_Serial.available())
    {
    String RES_input="";
    while(WiFi_Serial.available()) // read the data into a variable as long as the
    {
    RES_input+= (char)WiFi_Serial.read();
    }
    Serial.println(RES_input);
    if(WiFi_Serial.find("WIFI CONNECTED"))
    {
    Serial.println("CONNECTED TO WIFI NETWORK..!");
    }
    else
    Serial.println("UNABLE TO CONNECT TO WIFI NETWORK..!");
```

```
    }
    }
    else
    {
    lcd.clear();
    lcd.print("CONNECTED TO");
    lcd.setCursor(0,1);
    lcd.print("WIFI NETWORK..!!");
    Serial.println("CONNECTED TO WIFI NETWORK..!");
    Serial.println();
    Serial.println();
    post();
    i=0;
    }
    }
}
```

Step16:

16.1 Start a void **"TEMPERATURE**" function and with the help of "analogRead" function read the value from the Analog variable and store it to an integer variable. This gets the Temperature from the sensor and prints the actual value by converting analog voltage to the temp. Refer to data sheet for temp in manual.

16.2 Divide the obtained value with the resolution of the ADC (i.e., 1023.0) and multiply the result with reference milli volts. (5V = 5000mV) and save it to a float variable.

16.3 Divide the above result by 10 because there will be 1°C change for every 10mV of output and save it to another float variable.

16.4 Set the cursor position by column and row numbers using "lcd.setCursor" function.

16.5 Display the value on the LCD using "lcd.print" function.

```
void TEMPERATURE()
  {
Serial.print("Geeting Temp\t");
lcd.clear();
intvalue_temp=analogRead(T);
floatmillivolts_temp=(value_temp/1023.0)*5000;
CELSIUS=millivolts_temp/10;
lcd.setCursor(0,0);
lcd.print("T:");
```

```
lcd.print(CELSIUS);
Serial.print("T:\t");
Serial.print(CELSIUS); Serial.print("\t");
  }
```

Step17: Write similar functions to read analog Humidity and Light values.

```
void HUMIDITY()
  {
Serial.print("Getting Hum ");
intchk = DHT11.read(H);
    HUM=DHT11.humidity;
lcd.setCursor(9,0);lcd.print("H:");
lcd.print(HUM);
Serial.print("H: ");
Serial.print("HUM"); Serial.print("\t");
   }
void LIG()
  {
Serial.print("Getting Lig ");
intvalue_lig=analogRead(L);
floatmillivolts_lig =(value_lig /1023.0)*5000;
LIGHT=millivolts_lig /10;
lcd.setCursor(0,1);
lcd.print("L:");
lcd.print(LIGHT);
Serial.print("L: ");
Serial.print(LIGHT);Serial.println("\t");
delay(2000);
lcd.clear();
  }
```

Step18:

18.1 Start a void “POWER” function and check the concerned digital pin/variable is LOW.

18.2 If it is LOW, display the status on the LCD with the help of “lcd.print” function and “lcd.setCursor” functions respectively.

18.3 Else, display its status.

```
void POWER()
  {
Serial.print("PWR STATUS\t");
if(digitalRead(P)==LOW)
   {
lcd.setCursor(0,0);
lcd.print("P:ON");
Serial.print("P:ON");Serial.print("\t");
    }
else
    {
lcd.setCursor(0,0);
lcd.print("P:OFF");
Serial.print("P:OFF");Serial.print("\t");
    }
  }
```

Step19: write similar functions to read digital Reed switch status.

```
void REED()
  {
Serial.print("       REED STATUS ");
if(digitalRead(R)==LOW)
    {
lcd.setCursor(6,0);
lcd.print("R:OPEN");
Serial.print("R:OPEN");Serial.print("\t");
    }
else
    {
lcd.setCursor(6,0);
lcd.print("R:CLOSE");
Serial.print("R:CLOSE");Serial.print("\t");
    }
  }
void PIR()
  {
```

```
Serial.print("PIR STATUS ");
if(digitalRead(X)==LOW)
    {
lcd.setCursor(0,1);
lcd.print("X:YES");
Serial.print("X:YES");Serial.print("\t");
    }
else
    {
lcd.setCursor(0,1);
lcd.print("X:NO ");
Serial.print("X:NO");Serial.println("\t");
    }
delay(2000);
  }
```

Step20:

20.1 Start a**"post"** function.

20.1 Open the TCP along with the URL and Port number of the Cloud where the data needs to be posted

20.3 Initialize a String "data" and arrange data in JSON formatwith fields of device key, node number, Temperature, Humidity, Light intensity, Power supply sensing, Reed switch/Door sensing, etc.

20.4 Then Initialize another String with the HTTP post call **header** along with the URL of the cloud

20.5 Using "WiFi_Serial.println" function, send the command **"AT+CIPSEND="**, i.e., data to be sent and wait for">".

20.6 Then Using "WiFi_Serial.print" function, send the Character Count of the total string the Header + Data

20.7 Using "WiFi_Serial.println" function, send the **Header** string first then the **data** String

20.6 Initialize a string "RES_input" and read all the data available on serial from "AT+CIPSEND" response

20.7 Use the **"find_string"** search the response with "200".

20.8 If the string matches with response print "DATA POSTED" else print "Error in Posting"

```
void post()
{
    String data;
    data += "{\"device_no\":\"WKSP-
01\",\"client\":\"Workshop\",\"device_type\":\"SAM\",\"device_key\":\"LUJPIOJV7L5NHCWQ1F4J\",
\"node_no\":\"005\",\"Temp\":\"";
```

```
    data += String(CELSIUS);
    data += "\",\"HUM\":\"";
    data += String(HUM);
    data += "\",\"LDR\":\"";
    data += String("25");
    data += "\",\"POWER\":\"";
    if(digitalRead(P)==LOW)
    {
    data += "ON.";
    }
    else
    {
    data += "OFF";
    }
    data += "\",\"DOOR\":\"";
    if(digitalRead(R)==LOW)
    {
    data += "OPEN.";
    }
    else
    {
     data += "CLOSE";
    }
    data += "\"}";
    String uri="/services/data";
    String port="80";
    String http_req= "POST " + uri + " HTTP/1.1\r\n"+ "Host: " + DST_IP + ":" + port + "\r\n" + "Accept:
*" + "/" + "*\r\n" + "Content-Length: " + data.length() + "\r\n" ;
    String http_req1= "Content-Type: application/json\r\n\r\n" ;
    Serial.println(data);
    int x=(http_req.length());
    int y=(http_req1.length());
    int z=data.length();
    int Total_req_data_Length = (x+y+z);
    Serial.println();
    String cmd = "AT+CIPSTART=\"TCP\",\"";
    cmd += DST_IP;
    cmd += "\",80";
    Serial.println(cmd);
    WiFi_Serial.println(cmd);
    delay(5000);
    if(WiFi_Serial.available())
    {
    String RES_input="";
```

```
while(WiFi_Serial.available()) // read the data into a variable as long as the
{
RES_input+= (char)WiFi_Serial.read();
}
Serial.println(RES_input);
if(WiFi_Serial.find("CONNECT"));
Serial.print("AT+CIPSEND=");
WiFi_Serial.print("AT+CIPSEND=");
WiFi_Serial.println(Total_req_data_Length);
delay(100);
if(WiFi_Serial.available());
RES_input="";
while(WiFi_Serial.available()) // read the data into a variable as long as the
{
RES_input+= (char)WiFi_Serial.read();
}
Serial.println(RES_input);
{
if(WiFi_Serial.find(">"));
{
WiFi_Serial.print(http_req);
WiFi_Serial.print(http_req1);
WiFi_Serial.print(data);
delay(2000);
}
}
if(WiFi_Serial.available())
{
  delay(100);
String RES_input="";
while(WiFi_Serial.available()) // read the data into a variable as long as the
{
RES_input+= (char)WiFi_Serial.read();
}
lcd.clear();
Serial.println(RES_input);
if(WiFi_Serial.find("200"));
{
lcd.clear();
lcd.print("RESPONSE: 200");
lcd.setCursor(0,1);
lcd.print("DATA POSTED");
Serial.println("DATA POSTED");
delay(1000);
```

```
    }
    }
    }
    else
    {
    lcd.clear();
    lcd.print("Error in Posting");
    delay(1000);
    }
}
```

Step21:

21.1 Start a function with the name “loop”. This is the Main loop/function of the whole code/program.

21.2 Call the “lcd.clear” function to erase any old/garbage data on the lcd.

21.3 With the help of “lcd.setCursor” function, set the lcd cursor position to 0th column and 0th row.

21.4 With the help of “lcd.print” function, display some string data – “Axetla Systems”.

21.5 Call the delay function and after some time delay clear the lcd using “lcd.clear” function.

21.6 Start an infinite “while” loop.

21.7 Check if the variable “cnt” value is less than 1.

21.8 Call the function “WIFI_Check”, “connectWiFi” if returns with “OK” then Call“post” sequentially with 2000 milli seconds delay, using “delay” function in milli seconds.

21.9 Call the “lcd.clear” function and display the string “DATA POSTED” using “lcd.print” function.

21.10 If “connectWiFi” returns with no connection then call the functions **“TEMPERATURE”,“HUMIDITY”,“LIG(Light)”,“POWER”** and **“REED”** sequentially

21.11 Make the “cnt” variable value as 4 again.

21.12 Else if the compared “**CNT**” value is not less than 1, then call the functions **“TEMPERATURE”,“HUMIDITY”,“LIG(Light)”,“POWER”** and **“REED”** sequentially with 2000 milli seconds delay.

21.13 Decrease the “**CNT**” value by 1 for every loop.

```
/********MAIN LOOP*********/
void loop()
  {
    lcd.clear();
    lcd.setCursor(0,0);
    lcd.print("Axetla Systems");
    delay(1000);
    lcd.clear();
    while(1)
    {
    if(CNT<1)   //CNT for Time Delay
    {
    if(Start_chck==0)
    {
    WIFI_Check();
    if(i==1)
    {
    connectWiFi();
    }
    }
    else
    {
    CNT=4;
    Start_chck=0;
    }
    }
    TEMPERATURE();
    HUMIDITY();
    LIG();
    POWER();
    REED();
    PIR();
    Serial.println();
    CNT--;
    }
  }
```