```
////////////////////////////////////////////////////////////////////
//                                                                //
//                         SMART AGRICULTURE                      //
//                                 by                             //
//                           Srinivas Pai A                       //
//                            Suhas Kotian                        //
//                         Sudesh Mohandas Pai                    //
//                         Sudeep Novel Pinto                     //
//                           Vinod Kulkarni                       //
//                                                                //
////////////////////////////////////////////////////////////////////



#include <SPI.h>
#include<Ethernet.h>
#include "DHT.h"


#define DHTPIN 5
#define DHTOUT 6
#define DHTTYPE DHT11
#define PUMP 2
#define LIGHT 3
#define LDR A0
#define RAIN 4
#define SOIL A1
#define SOIL_TH 50//add
#define LIGHT_TH 90
#define HUMID 7
#define DEHUMID 8
#define HEATER 9
#define COOLER 10


DHT dht(DHTPIN, DHTOUT, DHTTYPE);

byte mac[] = { 0xDE, 0xAD, 0xBE, 0xEF, 0xFE, 0xED };
IPAddress ip(192, 168, 1, 177);
EthernetServer server(80);

String HTTP_req = "";
String temperature;
String humidity;
```

```
int TEMPERATURE_TH = 25;
int HUMIDITY_TH = 50;
boolean PUMP_STATUS = 0;
boolean LIGHT_STATUS = 0;
int light, rain, soil;
boolean flag = 0; //ignore
boolean man_pump = 0;
boolean man_light = 0;
boolean override_pump = 0;
boolean override_light = 0;
boolean man_humidity = 1;
boolean man_temperature = 1;
float h, t;
float temp_h, temp_t;
float tt, hh;


void setup()
{
  Ethernet.begin(mac, ip);
  server.begin();
  dht.begin();
  Serial.begin(9600);
  pinMode(PUMP, OUTPUT);
  pinMode(LIGHT, OUTPUT);
  pinMode(LDR, INPUT);
  pinMode(SOIL, INPUT);
  pinMode(RAIN, INPUT);
  pinMode(HEATER, OUTPUT);
  pinMode(COOLER, OUTPUT);
  pinMode(DEHUMID, OUTPUT);
  pinMode(HUMID, OUTPUT);
  digitalWrite(PUMP, LOW);
  digitalWrite(LIGHT, LOW);
  digitalWrite(HEATER, LOW);
  digitalWrite(COOLER, LOW);
  digitalWrite(HUMID, LOW);
  digitalWrite(DEHUMID, LOW);
   h = 60;
   t = 25;
}

void loop()
{
  delay(2000);
```

```
  ErrorCheck();

  Serial.println(h);
  Serial.println(t);
   float dp = t - (100 - h) / 5;

  soil = analogRead(SOIL);
  soil = map(soil, 0, 1023, 0, 100);
  boolean soil_t = soil > SOIL_TH;
  rain = digitalRead(RAIN);
  PUMP_STATUS = soil_t&(rain); //turn on pump only if both are true

  light = analogRead(A0); //reading sensor value
  light = map(light, 0, 1023, 0, 100); //converting it to %
  LIGHT_STATUS = light > LIGHT_TH;
  light = LIGHT_STATUS; //for displaying icon. required because we can
manipulate LIGHT_STATUS

  EthernetClient client = server.available();
  if (client)
  { // got client?
    boolean currentLineIsBlank = true;
    while (client.connected())
     {
      if (client.available())
      { // client data available to read
        char c = client.read(); // read 1 byte (character) from client
        HTTP_req += c;         // save HTTP request character
        // last line of client request is blank and ends with \n
        // respond to client only after last line received
        if (c == '\n' && currentLineIsBlank)
         {
          // send a standard http response header
          client.println("HTTP/1.1 200 OK");
          client.println("Content-Type: text/html");
          client.println("Connection: close");

          client.println();
          // send web page
          client.println("<!DOCTYPE html>");
          client.println("<html>");
          client.println("<head>");

          client.println("<style>");
          //client.println("background: rgba(76,175,80,0.1);");
```

```
        client.println("body{background-image:");

client.println("url('http://i0.wp.com/www.torreslandscapeva.com/wp-conte
nt/uploads/2013/02/Grass-Slider-Background1.png');");
        client.println("background-repeat:repeat-x;
background-attachment:fixed;font-family:'Open Sans', sans-serif;
background-position:bottom left; background-color: #e6f7ff;}");
         client.println(".txt_block{border: 1px solid black;
width:380px; height: 240px; padding-left: 10px;padding-right: 10px;
float: left; margin:20px 20px 20px 20px;}");

client.println(".rainy_weather{background-image:url('http://icons.iconar
chive.com/icons/oxygen-icons.org/oxygen/256/Status-weather-showers-day-i
con.png'); background-position: top right;background-size: 150px auto;
background-repeat: no-repeat; border: 1px solid black; width:380px;
height: 240px; padding-left: 10px;padding-right: 10px; float: left;
margin: 20px 20px 20px 20px;}");

client.println(".sunny_weather{background-image:url('http://icons.iconar
chive.com/icons/icons-land/weather/256/Sunny-icon.png');
background-position: top right;background-size: 150px auto;
background-repeat: no-repeat; border: 1px solid black; width:380px;
height: 240px; padding-left: 10px;padding-right: 10px; float: left;
margin: 20px 20px 20px 20px;}");

client.println(".night{background-image:url('http://icons.iconarchive.co
m/icons/dan-wiersma/solar-system/512/Moon-icon.png');
background-position: top right;background-size: 150px auto;
background-repeat: no-repeat; border: 1px solid black; width:380px;
height: 240px; padding-left: 10px;padding-right: 10px; float: left;
margin: 20px 20px 20px 20px;}");

client.println(".night_rainy{background-image:url('http://icons.iconarch
ive.com/icons/icons-land/weather/256/Night-Rain-icon.png');
background-position: top right;background-size: 150px auto;
background-repeat: no-repeat; border: 1px solid black; width:380px;
height: 240px; padding-left: 10px;padding-right: 10px; float: left;
margin: 20px 20px 20px 20px;}");
        client.println(".button{background-color:#1a8cff;
border:none; color:white; padding: 10px 25px; text-align: center;
text-decoration: none; display: inline-block; font-size: 16px;
cursor:pointer; border-radius: 0px; transition-duration: 0.5s;}");
         client.println(".disabled{opacity: 0.6; cursor:
not-allowed;}");
        client.println(".button1:hover {background-color:#99dfff;}");
```

```
        client.println(".CheckboxStyled{width: 80px; height: 26px;
background: #007a99; margin: 20px auto; position: relative;
border-radius: 50px;}");
        client.println(".CheckboxStyled:after{ content: 'OFF';
color:#000; position: absolute; right: 10px; z-index: 0; font:12px/26px
Arial,sans-serif; font-weight: bold;text-shadow: 1px 1px 0px rgba(255,
255, 255, 0.15);}");
        client.println(".CheckboxStyled:before{ content:'ON';
color:#27ae60; position: absolute; left:10px; z-index: 0;
font:12px/26px Arial,sans-serif; font-weight: bold;}");
        client.println(".CheckboxStyled label{display: block; width:
34px; height: 20px; cursor:pointer; position: absolute; top: 3px; left:
3px; z-index:1; background: #fcfff4; background:
-webkit-linear-gradient(top, #fcfff4 0%, #dfe5d7 40%, #b3bead 100%);
background: linear-gradient(to bottom, #fcfff4 0%, #dfe5d7 40%, #b3bead
100%); border-radius: 50px; transition: all 0.4s ease; box-shadow: 0px
2px 5px 0px rgba(0,0,0,0.3);}");
        client.println(".CheckboxStyled input[type=checkbox]{
visibility:hidden;}.CheckboxStyledinput[type=checkbox]:checked+label{
left: 43px;} {box-sizing: border-box;}");
         client.println("body .ondisplay section{ width: 120px;
height: 70px; background: #e6f7ff; display: inline-block; position:
relative; text-align: center; margin-top: 5px;}");
        client.println("body .ondisplay section:after{content:
attr(title); position: absolute; width: 100%; left:0; bottom: 3px;
font-size: 12px; font-weight: 400;}");
         client.println("</style>");
        client.println("<title>Smart Agriculture</title>");
         client.println("</head>");
         client.println("<body>");
        client.println("<div align=\"center\">");
        client.println("<h1>SMART AGRICULTURE</h1>");
         client.println("</div>");
        client.println("<div class=\"txt_block\">");
         client.println("<p><h3>Click to control water pump and light.
</h3></p>");
         client.println("<form method=\"get\">");
         ProcessCheckbox(client);
         client.println("</form>");
          if (1)
          {
            if (!rain)
            {
              if (light)
              {
```

```
    client.println("<div class=\"night_rainy\">");
   }
   else
   {
    client.println("<div class=\"rainy_weather\">");
   }
 }
 else
 {
   if (light)
   {
    client.println("<div class=\"night\">");
   }
   else
   {
    client.println("<div class=\"sunny_weather\">");
   }
 }
client.println("<h3>Weather Report</h3>");
client.println("Relative Humidity: ");
 client.println(h);
 client.println("%");
 client.println("<Br>");
client.println("Temperature: ");
 client.println(t);
client.println("<sup>O</sup>C<Br>");
 client.println("Dew point: ");
 client.println(dp);
client.println("<sup>O</sup>C<Br>");

 client.println("<p>");
 if (man_temperature)
 {
   if (t > TEMPERATURE_TH)
   {
    client.println("Cooler is ON<Br>");
   }
  else if (t < TEMPERATURE_TH)
   {
    client.println("Heater is ON<Br>");
   }
   else
   {
    client.println("Temperature is Maintained<Br>");
   }
```

```
   }
   else
   {
    client.println("Temperature Control is disabled<Br>");
   }

  if (man_humidity)
   {
     if (h > HUMIDITY_TH)
     {
      client.println("Dehumidifier is ON<Br>");
     }
     else if (h < HUMIDITY_TH)
     {
      client.println("Humidifier is ON<Br>");
     }
     else if (isnan(h))
     {
       client.println("");
     }
     else
     {
      client.println("Humidity is maintained");
     }
   }
   else
   {
    client.println("Humidity control is disabled");
   }
 }
 else
 {
  client.println("Failed to read DHT sensor");
 }
client.println("</p>");
 if (soil_t)
 {
  client.println("Soil moisture is LOW");
 }
 else
 {
  client.println("Soil moisture is normal");
 }

 if (!rain)
```

```
        {

         client.println("<p>It's Raining!</p>");
        }
       client.println("</div>");
       client.println("</body>");
       client.println("</html>");
       Serial.print(HTTP_req);//debug purpose

       HTTP_req = "";    // finished with request, empty string
        break;
      }
     // every line of text received from the client ends with \r\n
     if (c == '\n')
     {
       // last character on line of received text
       // starting new line with next character read
       currentLineIsBlank = true;
     }
     else if (c != '\r')
     {
       // a text character was received from client
       currentLineIsBlank = false;
     }
   } // end if (client.available())
 } // end while (client.connected())
 delay(1);      // give the web browser time to receive the data
 client.stop(); // close the connection
}// end if (client)

if (!override_light)
{
 digitalWrite(LIGHT, LIGHT_STATUS);
}

if (!override_pump)
{
 digitalWrite(PUMP, PUMP_STATUS);
}
if (man_temperature)
{
  if (t > TEMPERATURE_TH)
  {
    digitalWrite(HEATER, LOW);
    digitalWrite(COOLER, HIGH);
```

```
  }
  else if (t < TEMPERATURE_TH)
  {
   digitalWrite(COOLER, LOW);
   digitalWrite(HEATER, HIGH);
  }
  else
  {
   digitalWrite(COOLER, LOW);
   digitalWrite(HEATER, LOW);
  }
}
else
{
 digitalWrite(HEATER, LOW);
 digitalWrite(COOLER, LOW);
}

if (man_humidity)
{
  if (h > HUMIDITY_TH)
  {
   digitalWrite(DEHUMID, HIGH);
   digitalWrite(HUMID, LOW);
  }
  else if (h < HUMIDITY_TH)
  {
   digitalWrite(HUMID, HIGH);
   digitalWrite(DEHUMID, LOW);
  }
  else if (isnan(h))
  {
    //do nothing
  }
  else
  {
   digitalWrite(HUMID, LOW);
   digitalWrite(DEHUMID, LOW);
  }
}
else
{
 digitalWrite(HUMID, LOW);
 digitalWrite(DEHUMID, LOW);
}
```

```
}


// switch PUMP and LIGHT and send back for checkbox
void ProcessCheckbox(EthernetClient cl)
{

  man_pump = HTTP_req.indexOf("PUMP=2") > -1;
  man_light = HTTP_req.indexOf("LIGHT=2") > -1;
  man_humidity = HTTP_req.indexOf("HUMIDITY=2") > -1;
  man_temperature = HTTP_req.indexOf("TEMPERATURE=2") > -1; //checkbox
i.e., to enable or disable text box
  int index = HTTP_req.indexOf("temperature=");//for text box
  int indexa = 0;

 cl.println("<div class=\"ondisplay\">");
 cl.println("<section title=\"Pump Override\">");
 CheckboxStyled(cl);//just a function to simplify things

 if (HTTP_req.indexOf("PUMP_OVERRIDE=2") > -1)
  {
    override_pump = 1;
    PUMP_STATUS = man_pump;//now pump depends only on checkbox
   cl.println("<input type=\"checkbox\" name=\"PUMP_OVERRIDE\"
id=\"PUMP_OVERRIDE\" value=\"2\" \onclick=\"submit();\" checked>");
    Serial.println("override");
  }
 else
  {
    override_pump = 0;
    PUMP_STATUS = PUMP_STATUS;
   cl.println("<input type=\"checkbox\" name=\"PUMP_OVERRIDE\"
id=\"PUMP_OVERRIDE\"value=\"2\" \onclick=\"submit();\">");
    Serial.println("on");//debug purpose
  }
cl.println("<label for=\"PUMP_OVERRIDE\"></label></div></section>");

 cl.println("<section title=\"Pump\">");
 CheckboxStyled(cl);
 if (PUMP_STATUS)
 { // switch PUMP on
   digitalWrite(PUMP, HIGH);
    // checkbox is checked
```

```
   cl.println("<input type=\"checkbox\" name=\"PUMP\" id=\"PUMP\"
value=\"2\" \onclick=\"submit();\" checked>");
  }
  else
  { // switch PUMP off
    digitalWrite(PUMP, LOW);
    // checkbox is unchecked
   cl.println("<input type=\"checkbox\" name=\"PUMP\" id=\"PUMP\"
value=\"2\"\onclick=\"submit();\">");
  }
 cl.println("<label for=\"PUMP\"></label></div></section>");

 cl.println("<section title=\"Light Override\">");
 CheckboxStyled(cl);
 if (HTTP_req.indexOf("LIGHT_OVERRIDE=2") > -1)
  {
    LIGHT_STATUS = man_light;
    override_light = 1;
   cl.println("<input type=\"checkbox\" name=\"LIGHT_OVERRIDE\"
id=\"LIGHT_OVERRIDE\" value=\"2\" \onclick=\"submit();\" checked>");
  }
  else
  {
    LIGHT_STATUS = LIGHT_STATUS;
    override_light = 0;
   cl.println("<input type=\"checkbox\" name=\"LIGHT_OVERRIDE\"
id=\"LIGHT_OVERRIDE\"value=\"2\"\onclick=\"submit();\">");
  }
 cl.println("<label for=\"LIGHT_OVERRIDE\"></label></div></section>");

 cl.println("<section title=\"Light\">");
 CheckboxStyled(cl);
 if (LIGHT_STATUS)
 { // switch LIGHT on
   digitalWrite(LIGHT, HIGH);
   // checkbox is checked
   cl.println("<input type=\"checkbox\" name=\"LIGHT\" id=\"LIGHT\"
value=\"2\" \onclick=\"submit();\" checked>");
  }
  else
  { // switch LIGHT off
    digitalWrite(LIGHT, LOW);
    // checkbox is unchecked
   cl.println("<input type=\"checkbox\" name=\"LIGHT\" id=\"LIGHT\"
value=\"2\"\onclick=\"submit();\">");
```

```
  }
 cl.println("<label for=\"LIGHT\"></label></div></section>");

  cl.println("<section title=\"Control Humidity\">");
  CheckboxStyled(cl);
  if (man_humidity)
  {
   cl.println("<input type=\"checkbox\" name=\"HUMIDITY\"
id=\"HUMIDITY\" value=\"2\" onclick=\"submit();\" checked>");
  }
  else
  {
   cl.println("<input type=\"checkbox\" name=\"HUMIDITY\"
id=\"HUMIDITY\" value=\"2\" onclick=\"submit();\">");
  }
 cl.println("<label for=\"HUMIDITY\"></label></div></section>");

  cl.println("<section title=\"Control Temperature\">");
  CheckboxStyled(cl);
  if (man_temperature)
  {
   cl.println("<input type=\"checkbox\" name=\"TEMPERATURE\"
id=\"TEMPERATURE\" value=\"2\" onclick=\"submit();\" checked>");
  }
  else
  {
   cl.println("<input type=\"checkbox\" name=\"TEMPERATURE\"
id=\"TEMPERATURE\" value=\"2\" onclick=\"submit();\">");
  }
  cl.println("<label
for=\"TEMPERATURE\"></label></div></section></div></div>");


  if (1) //only if DHT sensor is active
  {
   if (HTTP_req.indexOf("temperature=") > -1)//only if text box has
returned something
    {
     indexa = index + 12;
     temperature += HTTP_req[indexa];
     indexa++;
     temperature += HTTP_req[indexa];
     indexa = 0;
     index = 0;
     TEMPERATURE_TH = temperature.toInt();
```

```
  temperature = "";

 }

index = HTTP_req.indexOf("humidity=");
 if (index > -1)
 {
  indexa = index + 9;
  humidity += HTTP_req[indexa];
  indexa++;
  humidity += HTTP_req[indexa];
  indexa = 0;
  index = 0;
  HUMIDITY_TH = humidity.toInt();
  humidity = "";
 }

cl.println("<div class=\"txt_block\">");
cl.println("<p><h3>Humidity and Temperature control</h3></p>");
 cl.println("<p>Temperature in <sup>0</sup>C:     ");
cl.println("<input type=\"text\" name=\"temperature\" value=");
cl.println(TEMPERATURE_TH);
 if (!man_temperature)
 {
  cl.println("disabled");
  cl.println("><Br><Br>");
 }

 else
 {
  cl.println("><Br>");
  cl.println("Current setting for Temperature is <b>");
  cl.println(TEMPERATURE_TH);
  cl.println("<sup>0</sup>C</b><p>");
 }
//cl.println("<Br>");

 cl.println("Relative Humidity in %      :");
cl.println("<input type=\"text\" name=\"humidity\" value=");
cl.println(HUMIDITY_TH);
 if (!man_humidity)
 {
  cl.println("disabled");
  cl.println(">");
 }
```

```
    else
    {
      cl.println("><Br>");
      cl.println("Current setting for humidity is<b> ");
      cl.println(HUMIDITY_TH);
      cl.println("%</b>");
    }
   if (man_humidity || man_temperature)
    {
     Serial.println("button style enable");//debug purpose only
     cl.println("<p><div align=\"center\"><input
type=\"submit\"class=\"button button1\" value=\"Submit\"> </div></p>");
    }
    else
    {
     Serial.println("button style disable");
     cl.println("<p><div align=\"center\"><input
type=\"submit\"class=\"button disabled\" value=\"Submit\"
disabled></div></p>");
    }
   cl.println("</div>");
  }

}


void CheckboxStyled(EthernetClient cl)//function
{
 cl.println("<div class=\"CheckboxStyled\">");
}

void ErrorCheck()
{
  for (int i = 0; i < 10; i++)
  {
    temp_h = dht.readHumidity();//we have to check for NAN
    temp_t = dht.readTemperature();//same here
    if(!isnan(temp_h))
    {
      t=temp_t;
      h=temp_h;
      break;
    }
    delay(2000);
```

```
    }

}
```