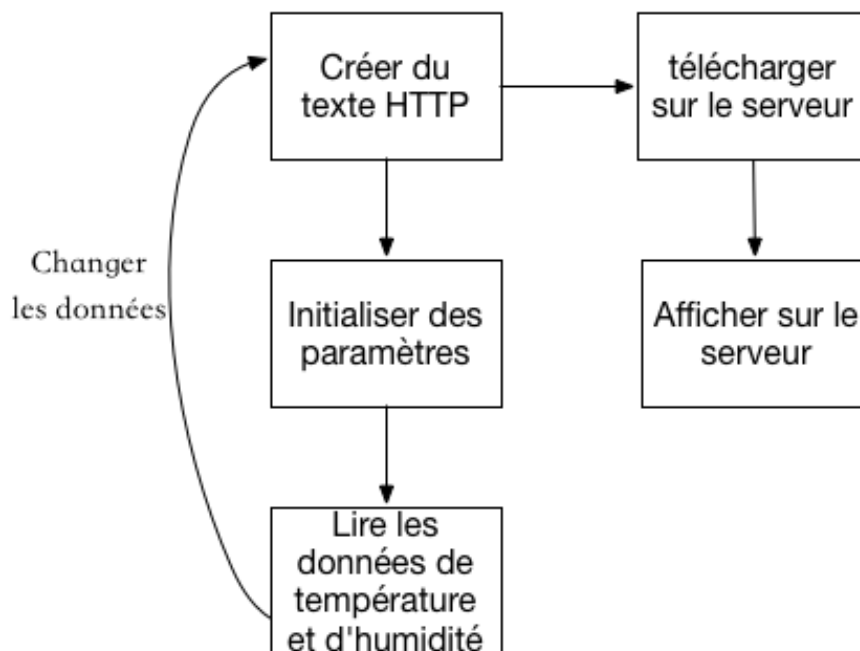


Rapport de Mini projet

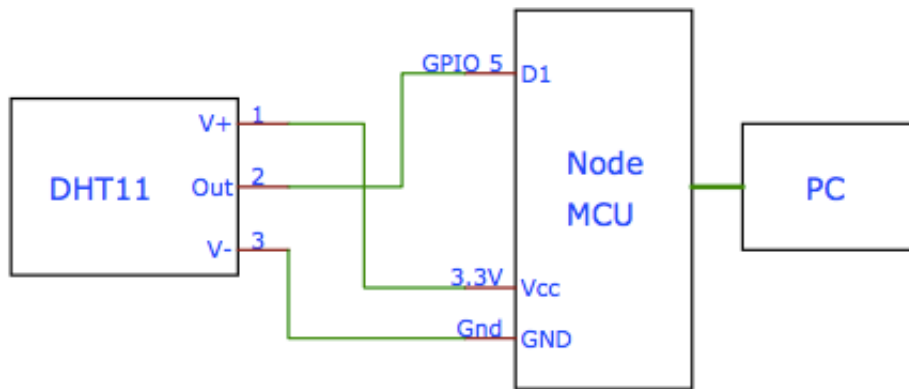
- **Les technologies .**

- **WIFI:** Un réseau Wi-Fi permet de relier par ondes radio plusieurs appareils informatiques (ordinateur, routeur, smartphone, modem Internet, etc.) au sein d'un réseau informatique afin de permettre la transmission de données entre eux.
 - **Web asynchrone:** Dans le Web asynchrone, il est possible de fournir des modifications de présentation spontanées à l'utilisateur à mesure que l'état d'un système dynamique change, sans que l'utilisateur ait besoin d'interagir avec l'interface.
-

- **Schéma solution**



- **Schéma électronique**





• Installer et lancer solution

◦ Créer une page Web

- Demo

Station Météo

 Temperature 27.00 °C  Humidity 28.00 %

History,---1---,---2---,---3---,---4---,---5---,---6---,---7---

Temp:;26.00,26.00,26.00,26.00,26.00,27.00,27.00

Humi:;21.00,21.00,21.00,23.00,24.00,24.00,28.00

- Code

```
const char index_html[] PROGMEM = R"rawliteral(
<!DOCTYPE HTML><html>
<head>
<meta charset="utf-8">
<meta name="viewport" content="width=device-width, initial-scale=1"
>
<link rel="stylesheet" href="https://use.fontawesome.com/releases/v
5.7.2/css/all.css" integrity="sha384-fnmOCqbTlWIlj8LyTjo7mOUStjsKC4
pOpQbqyi7RrhN7udi9RwhKkMHpvLbHG9Sr" crossorigin="anonymous">
<style>
html {
font-family: Arial;
display: inline-block;
margin: 0px auto;
```

```

    text-align: center;
}
h2 { font-size: 3.0rem; }
p { font-size: 3.0rem; }
.units { font-size: 1.2rem; }
.dht-labels{
    font-size: 1.5rem;
    vertical-align:middle;
    padding-bottom: 15px;
}
</style>
</head>
<body>
<h2>Station Météo</h2>
<p>
<i class="fas fa-thermometer" style="color:#c00000;"></i>
<span class="dht-labels">Temperature</span>
<span id="temperature">%TEMPERATURE%</span>
<sup class="units">&deg;C</sup>
<i class="fas fa-water" style="color:#00add6;"></i>
<span class="dht-labels">Humidity</span>
<span id="humidity">%HUMIDITY%</span>
<sup class="units">%</sup>
</p>
<p id="demo3">History</p>
<p id="demo">Temperature</p>
<p id="demo2">Humidity</p>
</body>
<script>
var i = 0;
var de = [ "History" ];
var tem = [ "Temp:" ];
var hum = [ "Humi:" ];
setInterval(function ( ) {
var xhttp = new XMLHttpRequest();
xhttp.onreadystatechange = function() {
if (this.readyState == 4 && this.status == 200) {
    document.getElementById("temperature").innerHTML = this.responseText;
    tem.push(this.responseText)
var x=document.getElementById("demo");
x.innerHTML=tem;
    i=i+1;
    de.push("----"+i+"----")
var y=document.getElementById("demo3");
y.innerHTML=de;
}
};

```

```

xhttp.open("GET", "/temperature", true);
xhttp.send();
}, 20000 ) ;
setInterval(function ( ) {
var xhttp = new XMLHttpRequest();
xhttp.onreadystatechange = function() {
if (this.readyState == 4 && this.status == 200) {
    document.getElementById("humidity").innerHTML = this.responseText
;
        hum.push(this.responseText)
var x=document.getElementById("demo2");
x.innerHTML=hum;
}
};
xhttp.open("GET", "/humidity", true);
xhttp.send();
}, 20000 ) ;
</script>
</html>
)rawliteral";

```

Tout le texte HTML avec les styles inclus est stocké dans la variable `index_html`. À l'intérieur de la balise `</body>`, nous ajoutons le contenu Web. La balise est nécessaire pour charger les icônes du site Web fontawesome.

Cette section précédente est responsable de la mise à jour de la température de manière asynchrone. Le même processus est répété pour les relevés d'humidité.

• Processor

```

String processor(const String& var){
if(var == "TEMPERATURE"){
return String(t);
}else if(var == "HUMIDITY"){
return String(h);
}
return String();
}

```

La fonction remplacera les espaces réservés dans le texte HTML par les valeurs réelles de température et d'humidité.

• Initialiser

- Initialiser le capteur DHT.

```
dht.begin();
```

- Configurer le réseau.

```
WiFi.begin(ssid, password);
Serial.println("Connecting to WiFi");
while (WiFi.status() != WL_CONNECTED) {
  delay(1000);
  Serial.println(".");
}

Serial.println(WiFi.localIP());

server.on("/", HTTP_GET, [](AsyncWebServerRequest *request){
  request->send_P(200, "text/html", index_html, processor);
});
server.on("/temperature", HTTP_GET, [](AsyncWebServerRequest *request){
  request->send_P(200, "text/plain", String(t).c_str());
});
server.on("/humidity", HTTP_GET, [](AsyncWebServerRequest *request){
  request->send_P(200, "text/plain", String(h).c_str());
});
server.begin();
```

Ces codes se connectent au réseau local et impriment l'adresse IP ESP8266.

Lorsque nous faisons une demande sur l'URL racine, nous envoyons le texte HTML stocké sur la variable `index_html`.

Lorsque nous recevons la demande sur l'URL / temperature(ou Humidité), nous avons juste besoin d'envoyer la valeur de température(ou Humidité) mise à jour.

Enfin, démarrer le serveur.

- **Main Loop**

```
unsigned long currentMillis = millis();
if (currentMillis - previousMillis >= interval) {
  previousMillis = currentMillis;
  float newT = dht.readTemperature();
  if (isnan(newT)) {
    Serial.println("Failed to read from DHT sensor!");
  }
  else {
    t = newT;
    Serial.println(t);
  }
  float newH = dht.readHumidity();
  if (isnan(newH)) {
    Serial.println("Failed to read from DHT sensor!");
  }
  else {
    h = newH;
    Serial.println(h);
  }
}
```

Obtenir de nouvelles lectures de température(ou Humidité) du capteur toutes les 20 secondes. Mettre à jour les variables de température et d'humidité.

- **Usages potentiels**

- Détecter la température et l'humidité dans la chambre.
- Détecter la température et l'humidité en laboratoire.
- Connecter l'humidificateur et le chauffage pour un réglage automatique.