TALLER IoT

Nombre: Luis Eduardo Cahuana Lopez **ID:** 000324966

Elabore un proceso de distribución en la recepción de una entidad EC2 (Máquina virtual en la Nube)

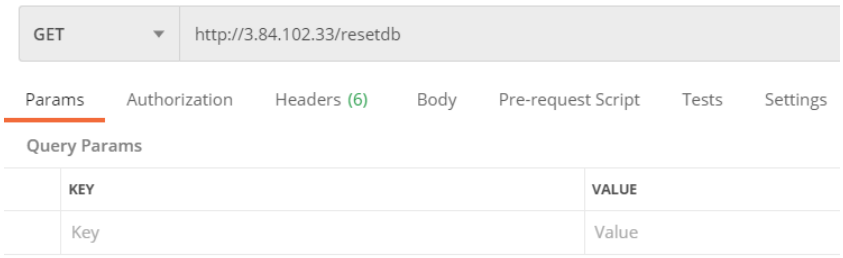
Programa su sensor NodeMCU para escribir en la máquina virtual en AWS por el protocolo HTTP

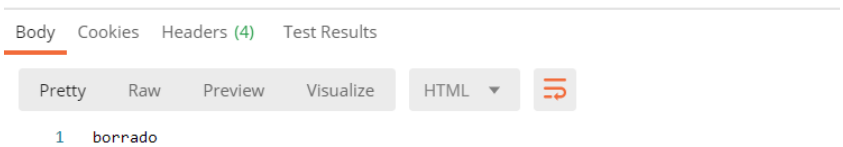
Se modificó el código realizado para el parcial y se le atribuyó el envío de datos mediante el protocolo HTTP y conexión a wifi.

```
void Armar_Trama()
{
  Trama = "";
  Serial.print ("Trama creada: ");
  Trama = String("id="+String(id)+"; temperatura="+temperatura+"; humedad="+humedad+"; latitud="+latitud +"; longitud="+longitud);
  Serial.println(Trama);
}
```

```
void Enviar()
{
 String PostData = "";
 Serial.println("datos para enviar");
 PostData = Trama; //String("id="+String(id)+"; temperatura="+String(temperatura,7)+"; longitud="+String(longitud,7)+"; latitud="+String(latitud,7));
 Serial.println(PostData);
 if ( client.connect(server,80))
 {
  Serial.println("conectado");
  client.print("POST /datos HTTP/1.1\n");
  // poner la direccion IP del servidor
  client.print("Host: 3.84.102.33
  \n");
  client.println("User-Agent: Arduino/1.0");
  client.println("Connection: close");
  client.println("Content-Type: application/x-www-form-urlencoded;");
  client.print("Content-Length: ");
  client.println(PostData.length());
  client.println();
  client.println(PostData);
 }
 else
 {
  Serial.println("error de conexion");
 }
}
```

Almacene cada registro en una base de datos





Almacenamiento de datos:

```
ubuntu@ip-172-31-41-232:~/ejemplo01$ sudo python3.6 programa01.py
 * Serving Flask app "programa01" (lazy loading)
 * Environment: production
   WARNING: This is a development server. Do not use it in a production deployment.
   Use a production WSGI server instead.
 * Debug mode: on
 * Running on http://0.0.0.0:80/ (Press CTRL+C to quit)
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 278-290-458
CombinedMultiDict([ImmutableMultiDict([]), ImmutableMultiDict([('id', '1001; temperatura=23.0729218; humedad=81.4941406; latitud=6.2446399; longitud=-75.5864029')])])
201.232.195.133 - - [27/Aug/2020 01:46:49] "POST /datos HTTP/1.1" 201 -
CombinedMultiDict([ImmutableMultiDict([]), ImmutableMultiDict([('id', '1001; temperatura=23.0719147; humedad=81.5979004; latitud=6.2446365; longitud=-75.5864029')])])
201.232.195.133 - - [27/Aug/2020 01:46:52] "POST /datos HTTP/1.1" 201 -
201.232.195.133 - - [27/Aug/2020 01:46:52] "GET /resetdb HTTP/1.1" 201 -
CombinedMultiDict([ImmutableMultiDict([]), ImmutableMultiDict([('id', '1001; temperatura=23.0714111; humedad=81.6027832; latitud=6.2446332; longitud=-75.5864029')])])
201.232.195.133 - - [27/Aug/2020 01:46:55] "POST /datos HTTP/1.1" 201 -
CombinedMultiDict([ImmutableMultiDict([]), ImmutableMultiDict([('id', '1001; temperatura=23.0729218; humedad=81.5979004; latitud=6.2446299; longitud=-75.5864029')])])
201.232.195.133 - - [27/Aug/2020 01:46:58] "POST /datos HTTP/1.1" 201 -
CombinedMultiDict([ImmutableMultiDict([]), ImmutableMultiDict([('id', '1001; temperatura=23.0699005; humedad=81.4941406; latitud=6.2446284; longitud=-75.5864029')])])
201.232.195.133 - - [27/Aug/2020 01:47:01] "POST /datos HTTP/1.1" 201 -
CombinedMultiDict([ImmutableMultiDict([]), ImmutableMultiDict([('id', '1001; temperatura=23.0714111; humedad=81.3598633; latitud=6.2446251; longitud=-75.5863953')])])
201.232.195.133 - - [27/Aug/2020 01:47:05] "POST /datos HTTP/1.1" 201 -
CombinedMultiDict([ImmutableMultiDict([]), ImmutableMultiDict([('id', '1001; temperatura=23.0693970; humedad=81.2988281; latitud=6.2446218; longitud=-75.5863953')])])
201.232.195.133 - - [27/Aug/2020 01:47:07] "POST /datos HTTP/1.1" 201 -
CombinedMultiDict([ImmutableMultiDict([]), ImmutableMultiDict([('id', '1001; temperatura=23.0688934; humedad=81.1950684; latitud=6.2446165; longitud=-75.5863953')])])
201.232.195.133 - - [27/Aug/2020 01:47:10] "POST /datos HTTP/1.1" 201 -
CombinedMultiDict([ImmutableMultiDict([]), ImmutableMultiDict([('id', '1001; temperatura=23.0653687; humedad=81.0974121; latitud=6.2446132; longitud=-75.5863953')])])
201.232.195.133 - - [27/Aug/2020 01:47:13] "POST /datos HTTP/1.1" 201 -
CombinedMultiDict([ImmutableMultiDict([]), ImmutableMultiDict([('id', '1001; temperatura=23.0683899; humedad=80.9997559; latitud=6.2446117; longitud=-75.5864029')])])
201.232.195.133 - - [27/Aug/2020 01:47:16] "POST /datos HTTP/1.1" 201 -
CombinedMultiDict([ImmutableMultiDict([]), ImmutableMultiDict([('id', '1001; temperatura=23.0699005; humedad=80.8959961; latitud=6.2446084; longitud=-75.5864029')])])
201.232.195.133 - - [27/Aug/2020 01:47:20] "POST /datos HTTP/1.1" 201 -
CombinedMultiDict([ImmutableMultiDict([]), ImmutableMultiDict([('id', '1001; temperatura=23.0663757; humedad=80.8862305; latitud=6.2446084; longitud=-75.5864029')])])
201.232.195.133 - - [27/Aug/2020 01:47:22] "POST /datos HTTP/1.1" 201 -
CombinedMultiDict([ImmutableMultiDict([]), ImmutableMultiDict([('id', '1001; temperatura=23.0673828; humedad=80.7983398; latitud=6.2446065; longitud=-75.5864029')])])
201.232.195.133 - - [27/Aug/2020 01:47:26] "POST /datos HTTP/1.1" 201 -
CombinedMultiDict([ImmutableMultiDict([]), ImmutableMultiDict([('id', '1001; temperatura=23.0673828; humedad=80.8325195; latitud=6.2446051; longitud=-75.5864029')])])
201.232.195.133 - - [27/Aug/2020 01:47:28] "POST /datos HTTP/1.1" 201 -
CombinedMultiDict([ImmutableMultiDict([]), ImmutableMultiDict([('id', '1001; temperatura=23.0663757; humedad=80.8959961; latitud=6.2446032; longitud=-75.5863953')])])
201.232.195.133 - - [27/Aug/2020 01:47:31] "POST /datos HTTP/1.1" 201 -
CombinedMultiDict([ImmutableMultiDict([]), ImmutableMultiDict([('id', '1001; temperatura=23.0658722; humedad=80.8959961; latitud=6.2446017; longitud=-75.5864029')])])
201.232.195.133 - - [27/Aug/2020 01:47:34] "POST /datos HTTP/1.1" 201 -
CombinedMultiDict([ImmutableMultiDict([]), ImmutableMultiDict([('id', '1001; temperatura=23.0688934; humedad=80.8959961; latitud=6.2446017; longitud=-75.5864029')])])
201.232.195.133 - - [27/Aug/2020 01:47:38] "POST /datos HTTP/1.1" 201 -
CombinedMultiDict([ImmutableMultiDict([]), ImmutableMultiDict([('id', '1001; temperatura=23.0683899; humedad=80.8959961; latitud=6.2446017; longitud=-75.5864029')])])
```

- ***Desarrolle una subpágina web que grafique el historial del sensor empleando la librería DASH***

```python
import dash
import dash_core_components as dcc
from dash.dependencies import Input, Output
import dash_html_components as html
import sqlite3

db = "mibasededatos.db"
con = sqlite3.connect(db)
cur = con.cursor()
cur.execute("SELECT * FROM registro")
temperatura = []
humedad = []
timestamp = []
for fila in cur.execute("SELECT * FROM registro"):
  temperatura.append(fila[2])
  humedad.append(fila[3])
  timestamp.append(fila[1])

app = dash.Dash()

app.layout = html.Div(children=[
    html.H1(children='Dash Tutorials'),
    dcc.Graph(id='example',
        figure={
            'data': [
                {'x': timestamp, 'y': temperatura, 'type': 'bar', 'name': 'Temperatura sensor 1'},
                {'x': timestamp, 'y': humedad, 'type': 'bar', 'name': 'Humedad sensor 1'},
            ],
            'layout': {
                'title': 'Basic Dash Example'
            }
        })
])


if __name__ == '__main__':
    app.run_server(debug=True, host='0.0.0.0', port=80 )
```

```
ubuntu@ip-172-31-41-232:~/ejemplo01$ sudo python3.6 programa02.py
Dash is running on http://0.0.0.0:80/

 Warning: This is a development server. Do not use app.run_server
 in production, use a production WSGI server like gunicorn instead.

 * Serving Flask app "programa02" (lazy loading)
 * Environment: production
   WARNING: This is a development server. Do not use it in a production deployment.
   Use a production WSGI server instead.
 * Debug mode: on
```

Muestra de la página: