

Roteiro Nº 06	Acesso Remoto a Dados de Sensores com ESP32 e Ngrok <i>(Monitoramento de Temperatura e Umidade com Servidor Web Local e Túnel Seguro)</i>	
Curso Engenharia Elétrica	Disciplina ELT 1119 – Redes e Aplicações IoT	Professor Carlos Alberto Vasconcelos Bezerra
Nome do Estudante		Data

1. Objetivos da Aprendizagem

Demonstrar como acessar remotamente um servidor web embarcado no **ESP32**, que exibe dados de **temperatura e umidade do sensor DHT11**, usando a ferramenta **ngrok** para criar um túnel público, mesmo em uma rede privada criada por um **roteador de celular**.

2. Material Necessário

- Placa de desenvolvimento ESP32-DevKitC-V4;
- Computador com ambiente de desenvolvimento Thonny instalado e ngrok instalado e autenticado;
- Celular com modo roteador Wi-Fi;
- Cabo USB para comunicação entre o ESP32 e o computador;
- Sensor de humidade e temperatura DHT11;

3. Fundamentação Teórica

O desenvolvimento de aplicações embarcadas conectadas à internet — como é o caso de sistemas baseados no ESP32 — frequentemente exige que essas aplicações possam ser acessadas de forma remota, seja para monitoramento de sensores, controle de atuadores ou coleta de dados em tempo real.

Entretanto, dispositivos embarcados como o ESP32 normalmente operam em redes locais (LANs), sem um endereço IP público. Isso significa que, por padrão, eles não são acessíveis a partir da internet, a menos que se configurem roteadores com redirecionamento de portas (port forwarding) ou se contrate serviços de hospedagem em nuvem — o que pode ser complexo ou inviável em ambientes educacionais e de prototipagem. O ngrok é uma ferramenta que permite expor servidores locais à internet de forma segura e prática. Ele cria um túnel criptografado entre o dispositivo local e um servidor público operado pela plataforma ngrok, fornecendo uma URL pública temporária, que redireciona as requisições para o servidor local do usuário.

Dessa forma, qualquer dispositivo, em qualquer lugar do mundo, pode acessar a aplicação local como se ela estivesse hospedada em um servidor web real, mesmo que o ESP32 esteja conectado apenas a uma rede doméstica ou ao roteador de um celular.

Nesta aula prática, o ESP32 atua como servidor web local, exibindo valores lidos do sensor de temperatura e umidade DHT11. Através do ngrok, é possível criar um túnel reverso para esse servidor, permitindo que usuários de fora da rede local acessem os dados em tempo real, mesmo que estejam em outra cidade ou rede. Essa estratégia proporciona uma experiência completa de Internet das Coisas (IoT), na qual os dados captados localmente podem ser disponibilizados globalmente, sem a necessidade de infraestrutura complexa.

Funcionamento resumido do ngrok:

1. O ngrok é executado em um computador conectado à mesma rede que o ESP32.
2. O ngrok cria um túnel entre o computador e um servidor na nuvem.
3. O servidor do ngrok fornece uma URL pública.
4. Qualquer dispositivo pode acessar essa URL para visualizar a página hospedada no ESP32.

4. Procedimento Experimental

4.1 Montagem do circuito:

Monte o circuito, alimentando o DHT11 com 3,3V e conectando sua saída de dados ao pino 15 do ESP32.

4.2 Configuração do Ambiente

- Abra o Thonny IDE e conecte-se ao ESP32.
- Digite o código abaixo no ambiente Thonny.

```
import network
import socket
import time
from machine import Pin
import dht

# === Configurações do Wi-Fi ===
SSID = "Nome_do_Hotspot"
PASSWORD = "Senha"

# === Inicializa sensor DHT11 no pino 15 ===
sensor = dht.DHT11(Pin(15))

# === Inicializa Wi-Fi ===
sta = network.WLAN(network.STA_IF)
sta.active(True)
sta.connect(SSID, PASSWORD)

while not sta.isconnected():
    pass

print("Conectado à rede!")
print("Endereço IP:", sta.ifconfig()[0])

# === Função HTML com os dados atualizados ===
def webpage(temp, hum):
    html = f"""<!DOCTYPE html>
<html>
<head>
<title>DHT11 - Temperatura e Umidade</title>
<meta http-equiv="refresh" content="5">
<meta name="ngrok-skip-browser-warning" content="true">
<style>
.container {{
    width: 30%;
    margin: auto;
    padding: 20px;
    border: 2px solid #000;
    border-radius: 10px;
    background-color: #f0f0f0;
    text-align: center;
    """
```

```
    font-family: Arial, sans-serif;
  }}
  h2 {{
    margin-bottom: 20px;
  }}
  p {{
    font-size: 1.2em;
  }}
</style>
</head>
<body>
  <div class="container">
    <h2>Leitura do DHT11</h2>
    <p>Temperatura: <strong>{temp} &deg;C</strong></p>
    <p>Umidade: <strong>{hum} %</strong></p>
  </div>
</body>
</html>""
  return html
```

```
# === Criação do servidor socket ===
s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.bind(("", 80))
s.listen(1)
print("Servidor iniciado!")
```

```
while True:
```

```
  try:
```

```
    conn, addr = s.accept()
    print("Conexão de", addr)
```

```
    # Lê dados do DHT11
```

```
    sensor.measure()
```

```
    temp = sensor.temperature()
```

```
    hum = sensor.humidity()
```

```
    # Gera a resposta HTTP completa de uma vez
```

```
    response = (
```

```
        "HTTP/1.1 200 OK\r\n"
```

```
        "Content-Type: text/html\r\n"
```

```
        "Connection: close\r\n"
```

```
        "ngrok-skip-browser-warning: true\r\n"
```

```
        "\r\n"
```

```
        + webpage(temp, hum)
```

```
    )
```

```
    # Recebe a requisição (limpa o buffer)
```

```
    conn.recv(1024)
```

```
    # Envia toda a resposta de uma vez como bytes
```

```
    conn.sendall(response.encode("utf-8"))
```

```
conn.close()
```

```
except OSError as e:  
    print("Erro:", e)  
    conn.close()
```

5. Cliente ngrok – Instalado no Notebook

5.1 Conectar o notebook à rede do celular

- Ative o roteador Wi-Fi (hotspot) no celular.
- Conecte o notebook a essa rede Wi-Fi.

5.2 Conectar o ESP32 à mesma rede

- No código MicroPython do ESP32, defina o SSID e a senha do roteador do celular.
SSID = "Nome_do_Hotspot"
PASSWORD = "Senha_do_Hotspot"
- O ESP32 receberá um IP da rede local, algo como 192.168.43.x.

5.3 Instalar e configurar o ngrok no notebook

- Baixe em: <https://ngrok.com/download>
- Extraia e abra o terminal (cmd) na pasta do ngrok.exe.
- Configure seu authtoken (copiado do painel da sua conta): *Abrir uma conta no NGROK*
Digite ngrok config add-authtoken SEU_AUTHTOKEN

5.4 Rodar o túnel ngrok para o ESP32

- No terminal (CMD) do notebook, execute:
ngrok http 192.168.43.x:80
Substitua 192.168.43.x pelo IP real do ESP32.

5.5 Acessar a URL gerada

- O ngrok mostrará algo como:
Forwarding https://abc123.ngrok.io -> http://192.168.43.x:80
- Copie a URL pública e abra em qualquer navegador (inclusive de fora da rede).

5.6 Celulares na mesma rede

Acessam o IP local do ESP32 diretamente (ex: <http://192.168.43.50>).

5.7 Clientes remotos

Acessam o link do ngrok (ex: <https://abc123.ngrok.io>).

6. Exercícios Propostos:

6.1 – Interpretação da Arquitetura

Explique com suas palavras o papel de cada um dos seguintes elementos na topologia da rede utilizada nesta aula:

- a) ESP32
- b) Celular atuando como roteador
- c) Notebook com ngrok
- d) Clientes conectados localmente
- e) Cliente remoto via internet

Dica: destaque como ocorre a comunicação entre os dispositivos e qual a função do ngrok na estrutura.

6.2 – Análise de Código

Observe o trecho de código abaixo, responsável por montar a resposta HTTP enviada pelo ESP32:

```
response = (  
    "HTTP/1.1 200 OK\\r\\n"  
    "Content-Type: text/html\\r\\n"  
    "Connection: close\\r\\n"  
    "ngrok-skip-browser-warning: true\\r\\n"  
    "\\r\\n"  
    + webpage(temp, hum)  
)  
conn.sendall(response.encode("utf-8"))
```

- a) Por que é necessário enviar toda a resposta com sendall() em vez de usar send() múltiplas vezes?
- b) Qual é a finalidade do cabeçalho ngrok-skip-browser-warning?

6.3 – Aplicação e Reflexão

Imagine que o ESP32 estivesse em uma rede sem acesso ao ngrok (por exemplo, sem notebook disponível). Quais seriam duas possíveis alternativas técnicas para permitir o acesso remoto ao servidor hospedado no ESP32? Quais as vantagens e desvantagens de cada uma?

Bibliografia

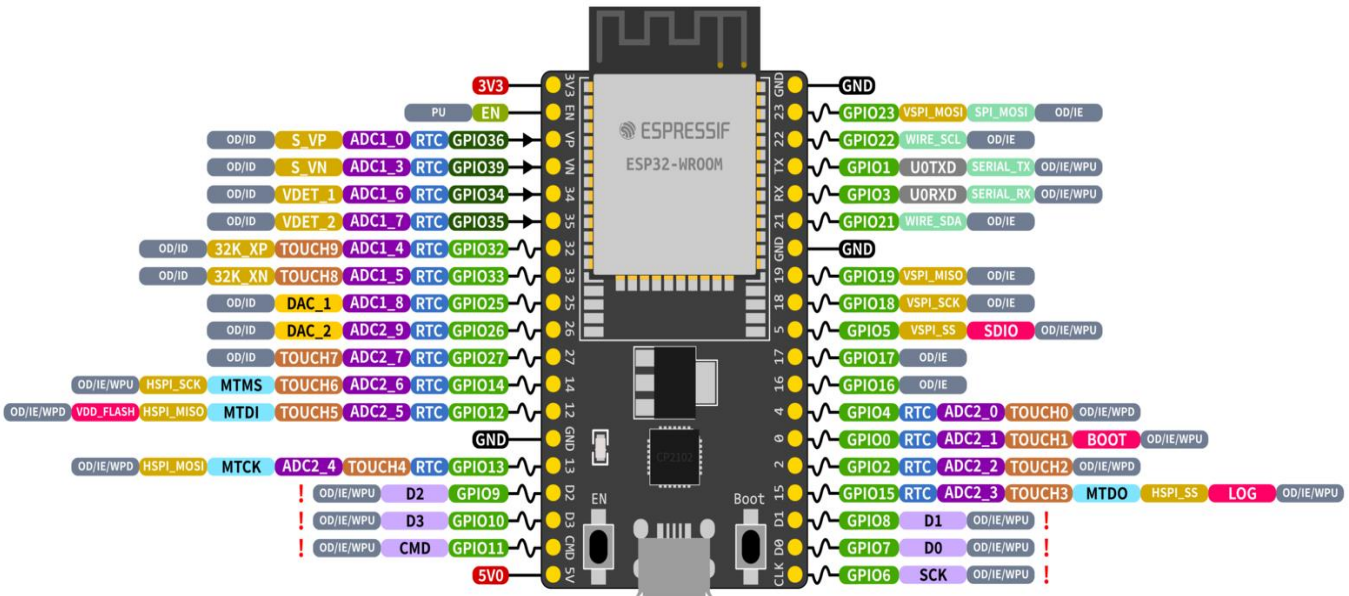
ASCHER, D.; LUTZ, M. Aprendendo Python. Porto Alegre: Bookman, 2007.

MENEZES, N. N. C. Introdução à programação com Python. São Paulo: Ed. Novatec, 2014.

OLIVEIRA, S. Internet das coisas com ESP8266, Arduino e Raspberry Pi. São Paulo: Ed. Novatec, 2017.

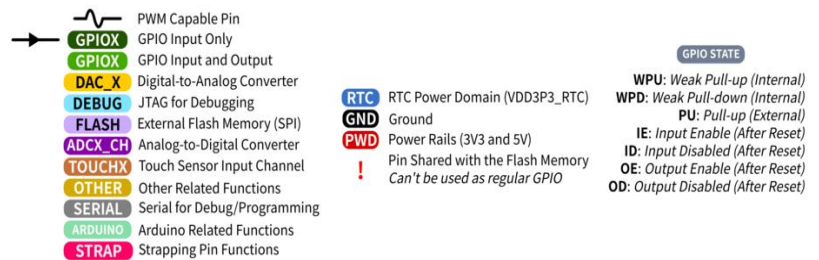
OLIVEIRA, C. L. V. IoT com Micropython e NodeMCU. São Paulo: Ed. Novatec, 2022.

ESP32-DevKitC



ESP32 Specs

32-bit Xtensa® dual-core @240MHz
Wi-Fi IEEE 802.11 b/g/n 2.4GHz
Bluetooth 4.2 BR/EDR and BLE
520 KB SRAM (16 KB for cache)
448 KB ROM
34 GPIOs, 4x SPI, 3x UART, 2x I2C,
2x I2S, RMT, LED PWM, 1 host SD/eMMC/SDIO,
1 slave SDIO/SPI, TWAI®, 12-bit ADC, Ethernet



```
ca: Prompt de Comando - ngrok http 192.168.100.133:80
ngrok (Ctrl+C to quit)
In London? Let's hang out. https://ngrok.com/info/kubecon-2025-ngrok-user-meetup

Session Status      online
Account             Carlos Alberto Vasconcelos Bezerra (Plan: Free)
Update              update available (version 3.22.0, Ctrl-U to update)
Version             3.20.0
Region              South America (sa)
Latency             53ms
Web Interface       http://127.0.0.1:4040
Forwarding           https://585e-2804-d59-8007-b400-60e9-14e7-53a8-b4e1.ngrok-free.app -> http://192.168.100.133:80

Connections          ttl    opn    rt1    rt5    p50    p90
                    53     0      0.22   0.11   0.36   0.42

HTTP Requests
-----
21:41:21.551 -09 GET /favicon.ico          200 OK
21:41:20.384 -09 GET /                    200 OK
21:41:14.546 -09 GET /favicon.ico          200 OK
21:41:13.418 -09 GET /                    200 OK
21:41:08.404 -09 GET /favicon.ico          200 OK
21:41:07.275 -09 GET /                    200 OK
21:41:02.257 -09 GET /favicon.ico          200 OK
21:41:01.155 -09 GET /                    200 OK
21:40:56.220 -09 GET /favicon.ico          200 OK
21:40:55.090 -09 GET /                    200 OK
```