

Missão Prática | Nível 5 | Mundo 4

Objetivo

Este projeto esta dividido em duas fases que são :

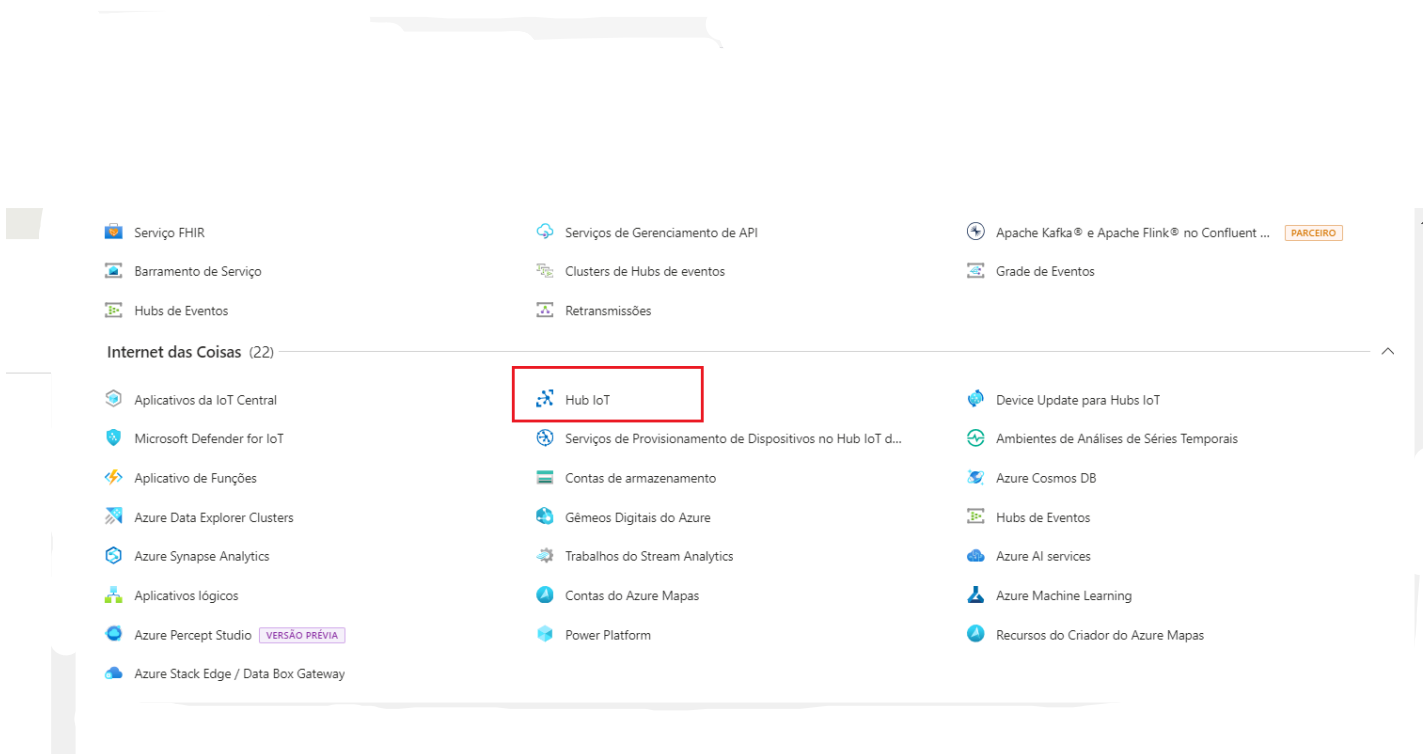
- Fase 1, consiste em um sistema para simulação, coleta e visualização de dados de dispositivos IoT. O projeto inclui um código Python que simula um sensor IoT, um servidor Node.js para receber e transmitir os dados, e uma interface web para visualizar esses dados em tempo real.
- Fase 2 , tem o objetivo de migrar a aplicação local, entenda-se o servidor Node.js, para a nuvem, utilizando os serviços do Azure para hospedar a aplicação e gerenciar a infraestrutura que continuará a receber dados do emulador do sensor de temperatura local utilizado na fase 1

Fase 1: Configuração Inicial e Simulação utilizando IoT

Configuração do Azure IoT Hub

1. Criar um Azure IoT Hub:

- Acesse o portal do Azure e crie um novo IoT Hub.
- Anote o nome do IoT Hub e a chave de conexão.



2. Registrar um dispositivo no IoT Hub:

- No IoT Hub, registre um novo dispositivo e anote a string de conexão do dispositivo.

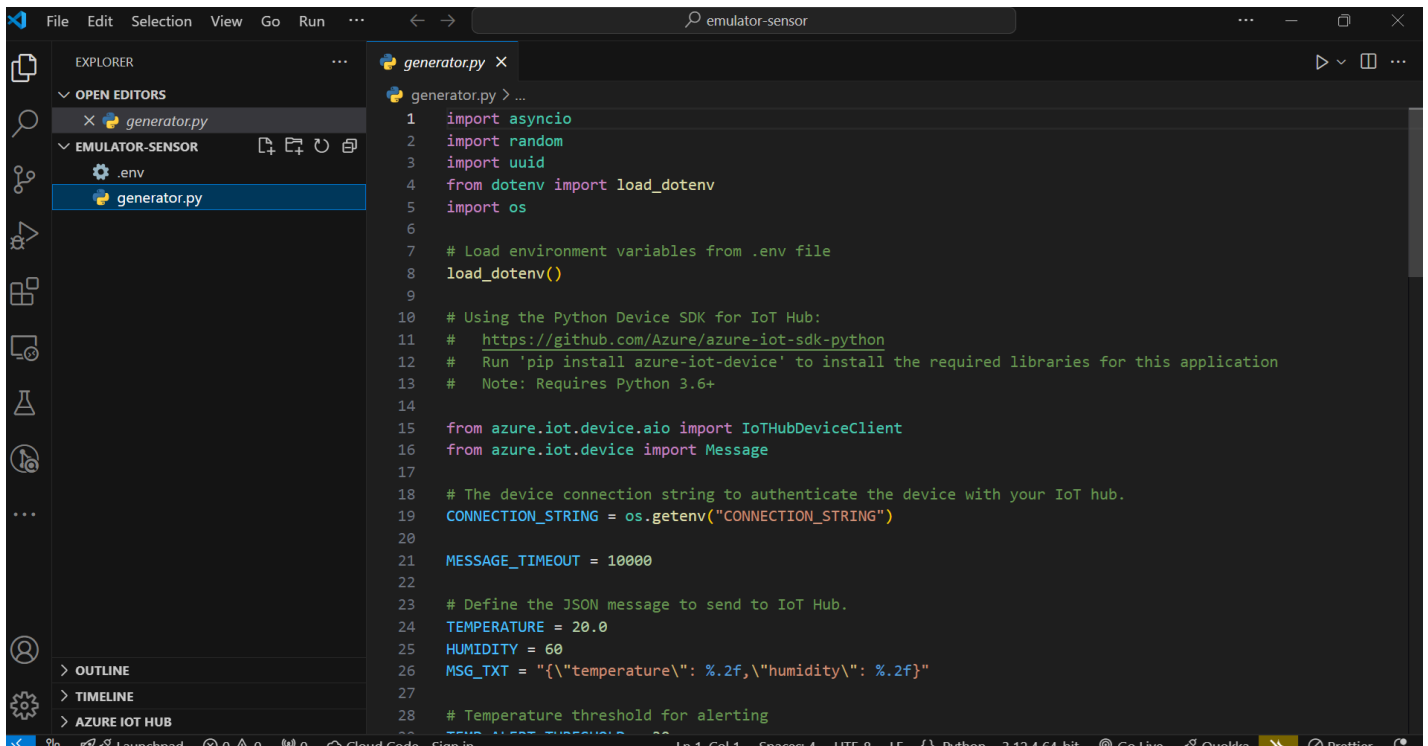
3. Adicionar um grupo de consumidores:

- Adicione um grupo de consumidores ao IoT Hub para permitir a leitura de eventos.

Configuração do Código Python (Simulador de Sensor)

Objetivo

O código Python simula um sensor IoT que envia dados de temperatura e umidade para o Azure IoT Hub.



Configuração

1. Instalar Dependências:

- Instalação da biblioteca azure-iot-device e python-dotenv:

```
pip install azure-iot-device python-dotenv
```

2. Criação de um arquivo .env no diretório do seu script Python com a string de conexão do

```

emulator-sensor > .env
1 CONNECTION_STRING= <YOUR CONNECT STRING>
2

```

dispositivo:

obtenha sua connect string pelo cloud shell

```
az iot hub show-connection-string --hub-name <MyHubName> --policy-name service
```

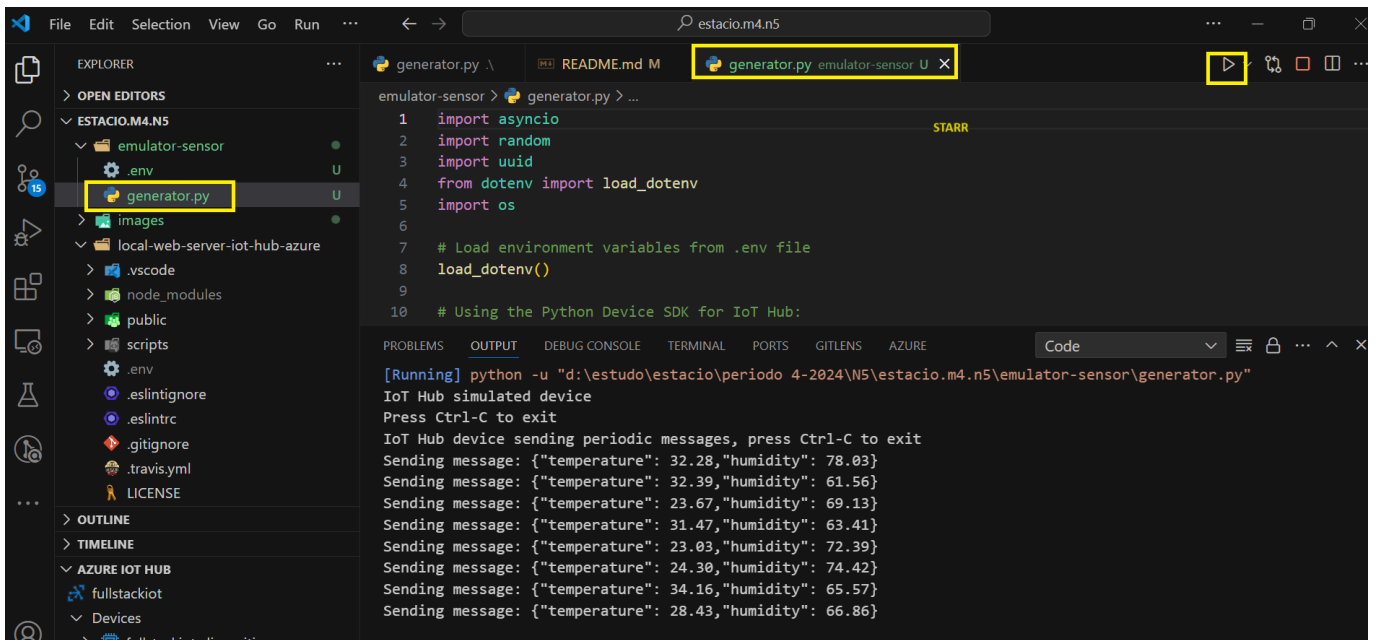
e cole em

```
CONNECTION_STRING= <MyconnectString>
```

3. Executar o Script Python (simulador do sensor de temperatura):

```
python sensor.py
```

npm start ou pelo vscode ou apenas dando um double click no arquivo python



Configuração do Servidor Node.js

Objetivo

O servidor Node.js recebe os dados do IoT Hub e os transmite via WebSocket para a interface web.

Configuração

1. Instalar Dependências:

- Instale as dependências necessárias:

```
npm install express http ws dotenv @azure/event-hubs
```

2. Criar um arquivo .env no diretório do servidor com a string de conexão do IoT Hub e o grupo de consumidores:

```
IotHubConnectionString= YOUR_IOT_HUB_NAME  
EventHubConsumerGroup= YOUR_CONSUMER_GROUP_NAME
```

3. Servidor Node.js (server.js):

Este código configura um servidor web que serve arquivos estáticos e redireciona todas as requisições para a raiz. Ele também configura um WebSocket para transmitir dados recebidos vindos de 'Azure IoT Hub' para todos o browse em tempo real.

4. Executar o Servidor (o aplicativo web local):

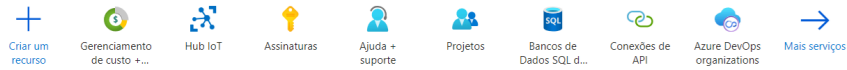
Interface Web

1. Objetivo:

- A interface web se conecta ao servidor via WebSocket e exibe os dados de telemetria em tempo real.
- Execute o arquivo

- o Preencha os detalhes necessários, como Nome do Aplicativo, Assinatura, Grupo de Recursos, Plano de Serviço de Aplicativo, entre outros

Serviços do Azure



Recursos

Recente Favorito

Nome	Tipo	Última visualização
fullstackiot-m4-web	Serviço de Aplicativo	8 horas atrás
Azure for Students	Assinatura	11 horas atrás
fullstackiot	Hub IoT	2 dias atrás
ASP-estaciomissao5-Bec5	Plano do Serviço de Aplicativo	5 dias atrás
estacio_missao_n5	Grupo de recursos	6 dias atrás
estacioMissaoPratica	Máquina virtual	uma semana atrás
estacio_m4_missao_pratica	Grupo de recursos	uma semana atrás
estacioMissaoPratica-vnet	Rede virtual	uma semana atrás

Ver todos

- o teste execução



Seu aplicativo web está sendo executado e aguardando pelo seu conteúdo

Seu aplicativo web está ativo, mas ainda não temos seu conteúdo. Se você já implantou, pode levar até 5 minutos para que seu conteúdo seja mostrado, então volte logo.



Suporte a Node.js, Java, .NET e muito mais

Ainda não foi implantado?
Use o centro de implantação para publicar o código ou configurar a implantação contínua.

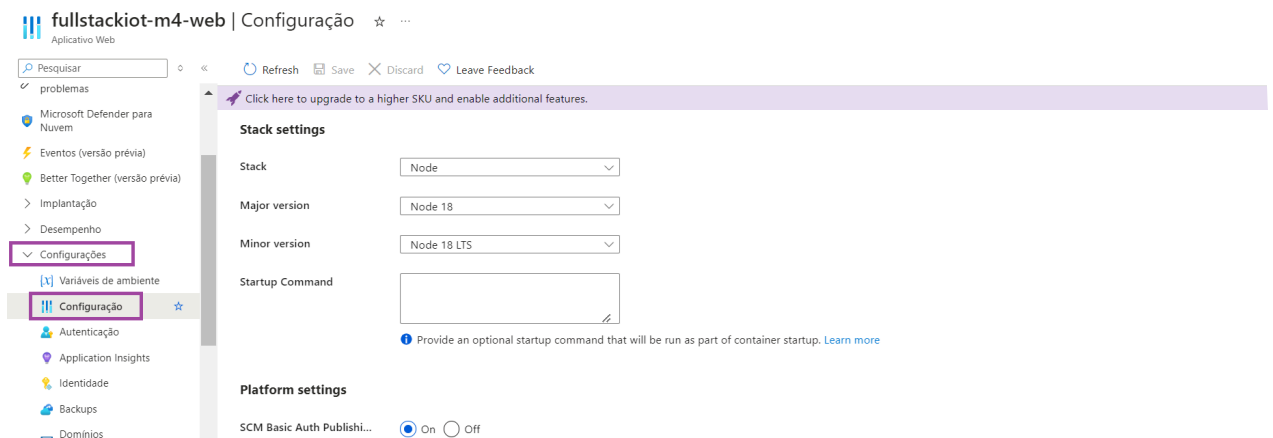
Centro de implantação

Iniciando um novo site?
Siga nosso guia de início rápido para preparar um aplicativo da web rapidamente.

Início rápido

2. Configuração Básica:

- o Após a criação, vá para a página de "Configurações" do seu aplicativo web.
- o Em "Configurações Gerais", configure o ambiente de execução (runtime stack) conforme necessário (por exemplo, Node.js, .NET, etc.).



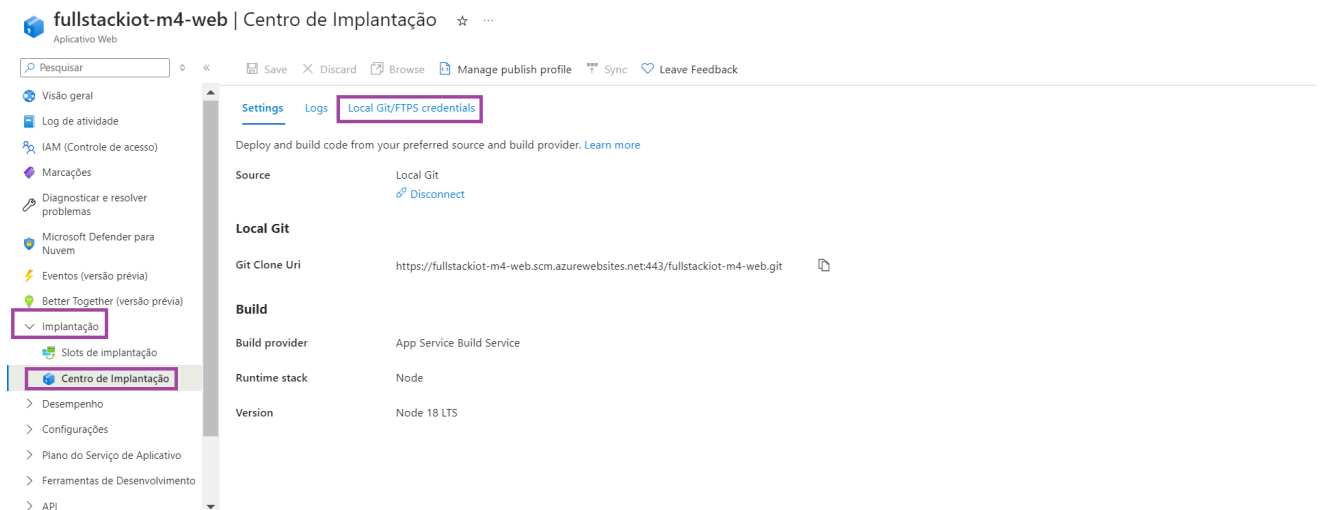
3. Habilitar Autenticação SCM:

- No menu lateral, selecione "Configurações" e depois "Geral".
- Ative "SCM Basic Auth Publishing" e "FTP Basic Auth Publishing".

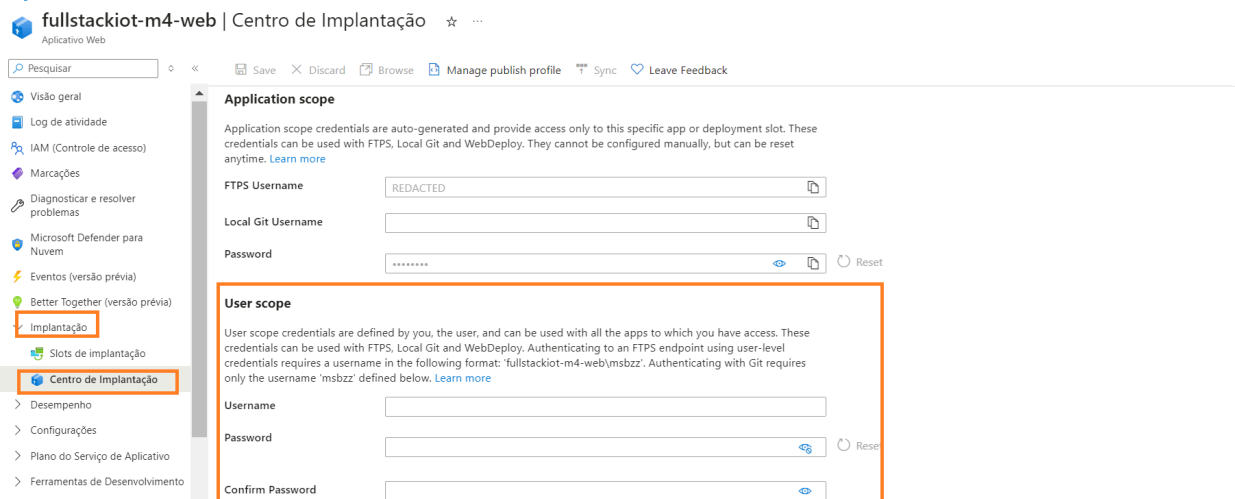
2. Configuração do Repositório Git Local:

1. Obtenha as Credenciais do Git:

- Navegue até "Centro de Implantação" no menu lateral.



- Selecione "Local Git/FTPS credentials" para obter o URL do repositório Git, o Nome de Usuário e a Senha.



- o Anote essas informações, pois serão usadas para autenticação.

2. Configurar o Repositório Localmente:

- o No terminal (PowerShell, Bash, etc.), navegue até o diretório do seu projeto.
- o Configure o repositório remoto com as credenciais obtidas:

```
git remote remove azure
```

```
git remote add azure https://<username>:<password>@<seu_app>.scm.azurewebsites.net:443/<seu_app>.git
```

3. Realizar atualização para o repositório git Azure:

- No terminal faça o 'git clone' apenas do projeto 'Servidor Node.js'

obs1: para isso vc precisará ter o projeto separado em um repositório

- o após clonar o projeto, entre na pasta e configure o arquivo .env não esqueendo de remover a referencia do arquivo local ".gitignore"
- o realize o 'git add .'
- o realize o 'git commit -m "SUA DESCRIÇÃO"
- o realize o 'git push -u azure master'

se quiser pode conferir o push no repositório git do azure

- o acesse o meu lateral ferramentas de seu web app e acesse o link 'ir'

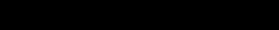


- o em Azure App Service selecione o menu 'Bash'

Build	20240522.2
Site up time	00.01:57:02
Site folder	/home
Temp folder	/tmp/

- App Settings
- Deployments
- Source control info
- Files
- Current Docker logs (Download as zip)

- Deployment Logs
- Site wwwroot

- ```

DEBUG CONSOLE | AZURE APP SERVICE ON LINUX

Documentation: http://aka.ms/webapp-linux
Kudu Version : 20240522.2
Commit : 6ea5b2dd86b0954053a7e59b1638a4c8f096802f

kudu_ssh_user@fullstackiot-m4-web_kudu_81e05024:/ $
```

- ```
kudu ssh user@fullstackiot-m4-web kudu 81e05024:~$ ls
ASP.NET  DeploymentLogStream  LogFiles  site  u0a03ebd205f1344a9a84bf
kudu ssh user@fullstackiot-m4-web kudu 81e05024:~$ cd site
kudu ssh user@fullstackiot-m4-web kudu 81e05024:~/site$ ls
build  config  deployments  diagnostics  locks  repository  wwwroot
kudu ssh user@fullstackiot-m4-web kudu 81e05024:~/site$ cd wwwroot
kudu ssh user@fullstackiot-m4-web kudu 81e05024:~/site/wwwroot$ ls
LICENSE  hostingstart.html  node_modules.tar.gz  package-lock.json  public  server.js
README.md  node_modules  oryx-manifest.toml  package.json  scripts
kudu ssh user@fullstackiot-m4-web kudu 81e05024:~/site/wwwroot$
```


3. Verificar a Implantação:

2. Testar a Aplicação:

- o Ative o emulador local em python do sensor de umidade

npm start ou pelo vscode ou apenas dando um double click no arquivo python

```
Sending message: {"temperature": 28.77, "humidity": 74.42}
Sending message: {"temperature": 22.44, "humidity": 63.57}
Sending message: {"temperature": 27.33, "humidity": 63.31}
Sending message: {"temperature": 21.42, "humidity": 73.80}
Sending message: {"temperature": 27.70, "humidity": 78.50}
Sending message: {"temperature": 20.84, "humidity": 79.21}
Sending message: {"temperature": 34.43, "humidity": 61.95}
Sending message: {"temperature": 28.69, "humidity": 66.19}
Sending message: {"temperature": 20.65, "humidity": 72.37}
Sending message: {"temperature": 26.57, "humidity": 76.89}
Sending message: {"temperature": 22.01, "humidity": 69.78}
Sending message: {"temperature": 28.11, "humidity": 70.53}
Sending message: {"temperature": 30.85, "humidity": 60.03}
Sending message: {"temperature": 26.74, "humidity": 69.28}
Sending message: {"temperature": 26.15, "humidity": 62.12}
Sending message: {"temperature": 24.53, "humidity": 60.12}
Sending message: {"temperature": 24.82, "humidity": 70.90}
Sending message: {"temperature": 32.76, "humidity": 67.59}
Sending message: {"temperature": 20.38, "humidity": 73.17}
Sending message: {"temperature": 22.03, "humidity": 65.28}
Sending message: {"temperature": 22.99, "humidity": 67.71}
Sending message: {"temperature": 33.11, "humidity": 74.08}
Sending message: {"temperature": 28.17, "humidity": 65.82}
```

- o Acesse a URL do seu aplicativo web para verificar se está funcionando conforme esperado.

1 device

fullstackiot_dispositivo

Temperature & Humidity Real-time Data

