

Integração de Sistemas
Professor Jefferson Chaves
jefferson.chaves@ifpr.edu.br

INTRODUÇÃO À AWS IOT CORE

*Conectar seus dispositivos a nuvem pode ser uma tarefa complexa. Isso porque em cenário tais como esse, é necessário se lidar com **distintos protocolos** e estimar recursos computacionais para sua operação (escala horizontal e vertical). Será necessário considerar questões que envolvem segurança no acesso dos dispositivos e dos dados, além de estruturar uma arquitetura que permita a integração entre os mais diversos dispositivos e o ferramental para processamento, armazenamento e disponibilização dos dados.*

*Nesse sentido, diversos serviços em nuvem foram disponibilizados, tais como IBM Cloud, AWS, Google Cloud, Azure e etc. Dentre essas opções, para aplicações IoT, tem se destacado a **AWS IoT Core**.*

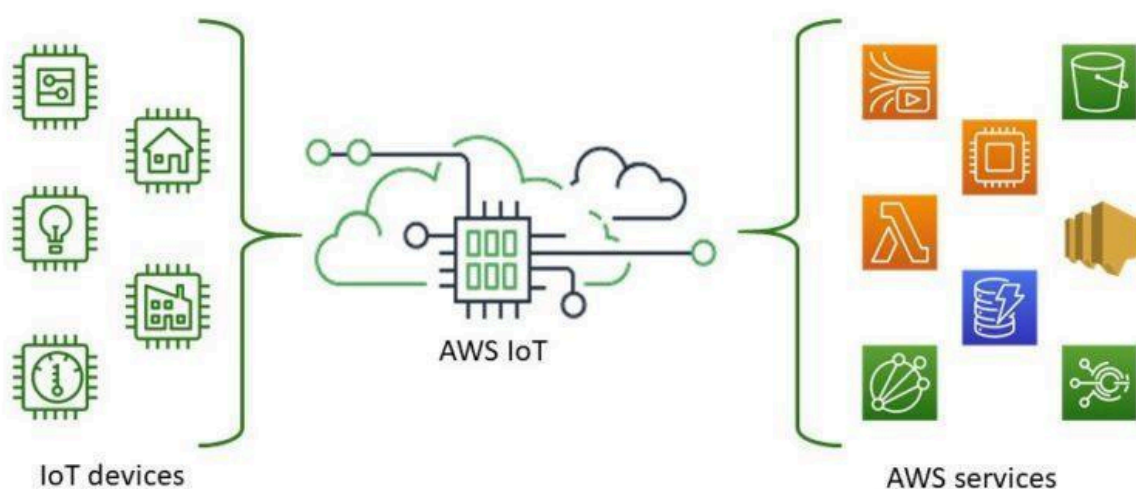
O AWS IoT Core é um serviço de nuvem gerenciado que permite que dispositivos conectados interajam de maneira fácil e segura com aplicativos de nuvem e outros dispositivos.

1. AWS IOT

*A AWS oferece serviços e soluções de Internet das Coisas (IoT) para conectar e gerenciar bilhões de dispositivos. Esses serviços de nuvem conectam seus dispositivos IoT a outros dispositivos e serviços de nuvem AWS. A AWS IoT fornece softwares e certificados que possibilitam de forma segura integrar seus dispositivos em soluções na nuvem IoT da AWS. **Se seus dispositivos podem se conectar à AWS IoT, então a AWS IoT pode conectá-los aos serviços de nuvem que a AWS fornece.***

A AWS IoT permite selecionar as tecnologias mais adequadas e atualizadas para sua solução. Para ajudá-lo a gerenciar e oferecer suporte a seus dispositivos IoT em campo, o AWS IoT Core oferece suporte a estes protocolos: MQTT, MQTT sobre WSS (Websockets Secure), HTTPS e LoRaWAN.

Este texto pretende introduzir os principais conceitos da AWS IoT por meio de guia para realizar a conexão de um ESP8266 à AWS IoT Core e publicar a leitura do sensor no broker MQTT da AWS IoT.



Como exemplo clássico, e de domínio de todos, para demonstração, usaremos o sensor para a leitura dos dados de temperatura de umidade. O ESP8266 se conectará à rede WiFi e publicará os dados do sensor na nuvem AWS IoT. Como toda conexão com a AWS IoT pode ser bidirecional, além de publicar os dados coletados, informações da plataforma serão enviadas ao ESP.

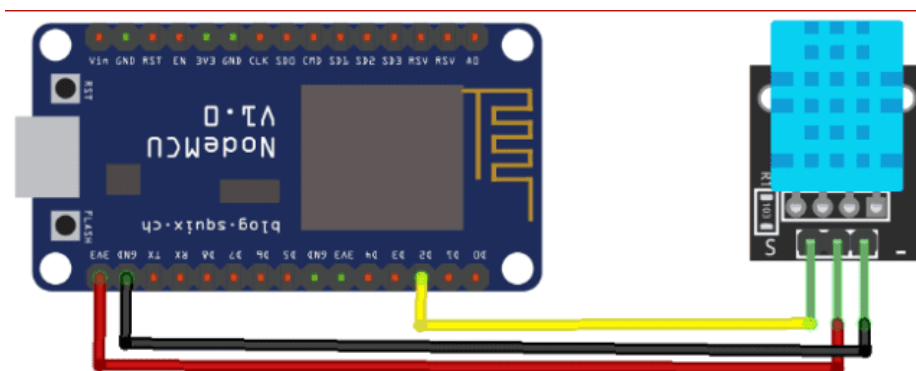
1.1 Seções deste material

- *Inscribendo-se e configurando o Amazon Web Services;*
- *Instalando as bibliotecas necessárias para o Arduino IDE e escrevendo um esboço do Arduino para o projeto;*
- *Criar uma **coisa** é AWS;*
- *Criando política e anexando a coisa;*
- *Gerando certificados;*
- *Modificando o esboço do Arduino de acordo com os dados e credenciais da coisa;*

- Assine e publique dados de e para o AWS Dashboard;

2 . CONFIGURAÇÃO DE HARDWARE

O hardware necessário para este projeto é um módulo Wifi ESP8266. E para a parte do sensor, usaremos o sensor de umidade e temperatura DHT11.



Tela inicial padrão do Home Assistant.

Veja outro exemplo [aqui](#)

Conecte o sensor DHT11 à placa ESP8266 conforme o diagrama de circuito apresentado. Conecte o pino VCC e GND do DHT11 a 3,3 V e GND do ESP8266. Da mesma forma, conecte o pino de sinal do DHT11 ao D2 (GPIO4) do NodeMCU ESP8266.

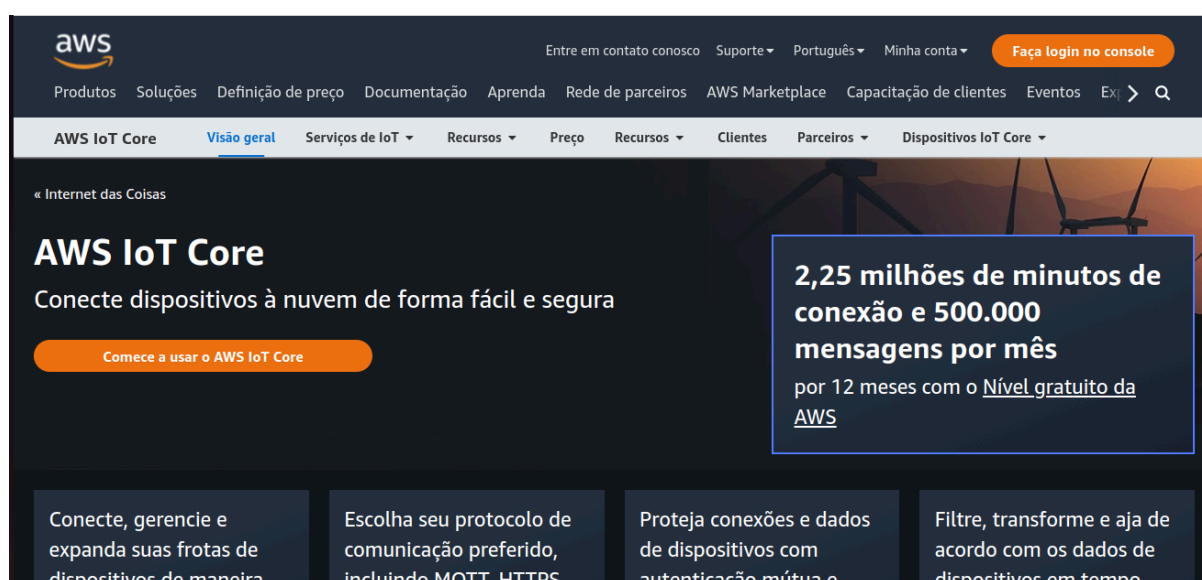


3 . CONCEITOS BÁSICOS DO AMAZON AWS IOT CORE COM ESP8266

Iremos explorar os principais conceitos da plataforma por meio da experimentação. Os passos a seguir servirão de guia para configurar a placa ESP8266 e começar um primeiro projeto com a plataforma.

3.1 Primeiro acesso

Em seu navegador e acesse: <https://aws.amazon.com/pt/iot-core>.



*De forma geral, é necessário configurar uma conta da AWS. Portanto, crie uma conta usando e-mail e senha. A conta também requer as informações **do seu cartão de crédito bancário**. Não haverá cobranças apenas realizando-se a criação da conta (embora seja necessário cuidado com esse tema, afinal trata-se de um serviço pago). Também será solicitado a verificação do número de telefone. Após esses passos, a conta será criada com sucesso.*



Explore os produtos do nível
gratuito com uma nova conta da
AWS.

Para saber mais, acesse aws.amazon.com/free.



Cadastrar-se na AWS

Endereço de e-mail do usuário raiz
Usado para recuperação de contas e algumas funções administrativas

Nome da conta da AWS
Escolha um nome para a sua conta. Você pode mudar o nome nas configurações de sua conta após o cadastro.

Verificar endereço de e-mail

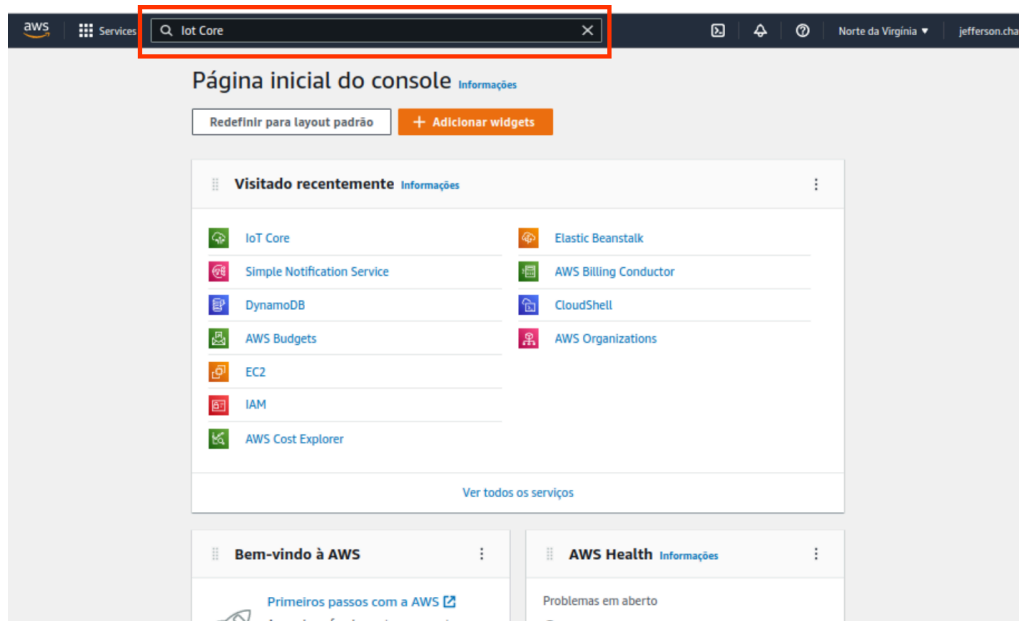
OU

Faça login em uma conta existente da
AWS

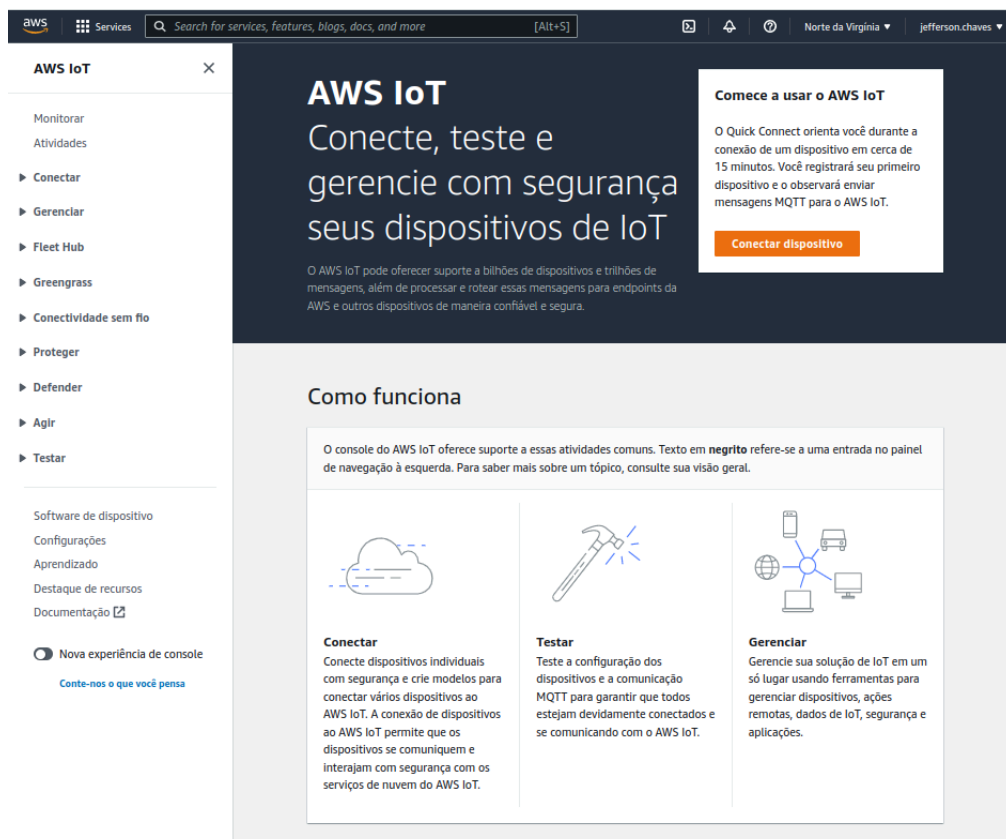
3.2 Painel do AWS IoT Core

Após realizar o cadastro de realizar o login na plataforma (chamada de console), a janela do console de gerenciamento da AWS será aberta. Na guia de pesquisa de serviços na parte superior, escreva 'IoT core' e pressione Enter.

IMPORTANTE: no momento em que esse guia foi elaborado, foi notado em sala que existe uma versão mais atualizada do painel da AWS IoT Core. Assim, pequenas diferenças visuais podem ser encontradas. Você pode ver essa versão do painel desabilitando “**Nova experiência do console**”.



Clique em IoT Core, para acessar a dashboard da AWS IoT.



No lado esquerdo do painel, estão organizados e dispostos os recursos disponíveis. Inicialmente exploraremos dois recursos: **Gerenciar** e **Proteger**.

3.3 Criando uma coisa

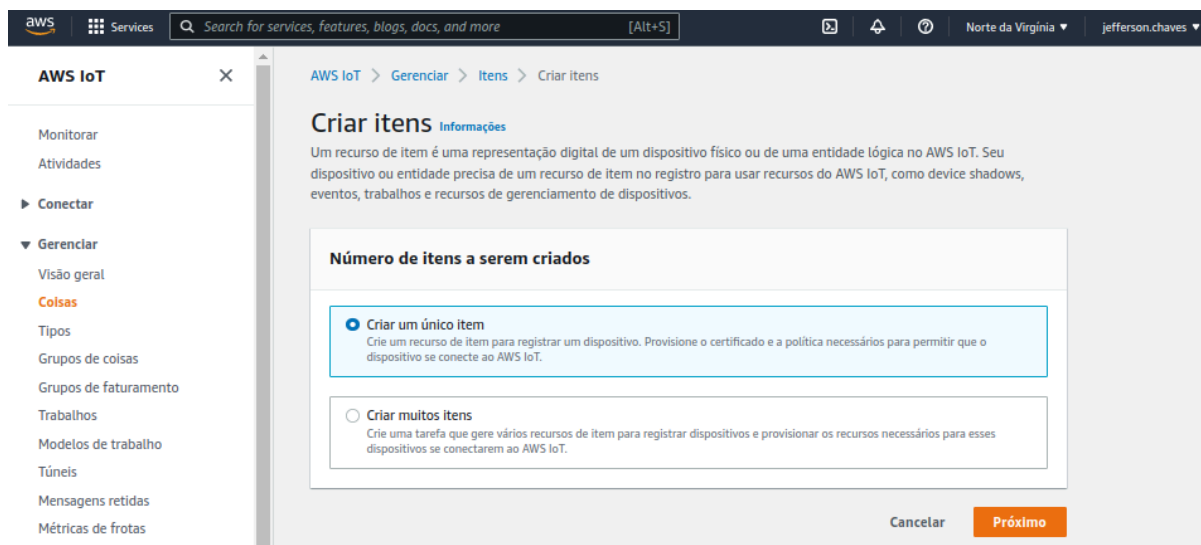
Nesse ponto, será necessário criar uma coisa, que virtualmente irá representar um dispositivo do mundo físico. Para tanto, siga os seguintes passos:

- Especificando propriedades de coisas
- Configurando o certificado do dispositivo
- Anexando políticas ao certificado

Na opção **Gerenciar**, clique em Coisa. Agora precisamos criar uma coisa aqui. Então, clique em Criar coisas aqui.



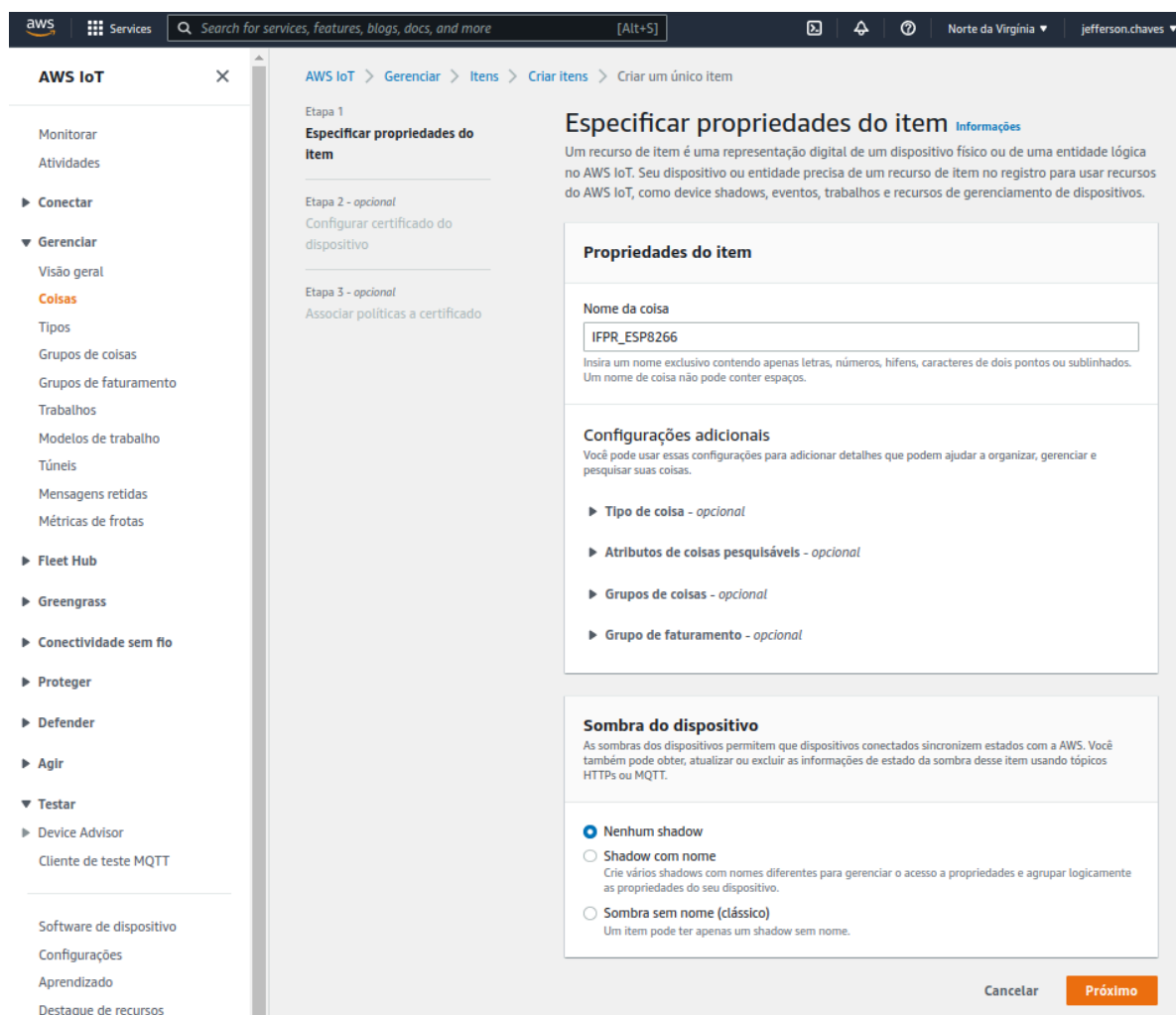
Você pode criar uma única **coisa** ou criar muitas **coisas**. Para este exemplo, selecione criar uma única coisa. Em seguida, clique em Avançar.



The screenshot shows the AWS IoT console interface. On the left is a navigation menu with options like 'Monitorar', 'Atividades', 'Conectar', and 'Gerenciar'. The 'Gerenciar' section is expanded, showing 'Visão geral', 'Coisas' (highlighted), 'Tipos', 'Grupos de coisas', 'Grupos de faturamento', 'Trabalhos', 'Modelos de trabalho', 'Túneis', 'Mensagens retidas', and 'Métricas de frotas'. The main content area is titled 'Criar itens' with a breadcrumb trail 'AWS IoT > Gerenciar > Itens > Criar itens'. Below the title is a description of an item resource. A form titled 'Número de itens a serem criados' contains two radio button options: 'Criar um único item' (selected) and 'Criar muitos itens'. At the bottom right are 'Cancelar' and 'Próximo' buttons.

3.3 Especificar propriedades da coisa

*Neste momento, precisaremos especificar as propriedades da coisa. Primeiro, dê um nome a uma coisa. Você pode nomear qualquer coisa. Por exemplo, vou chamá-lo de **IFPR_ESP8266**.*

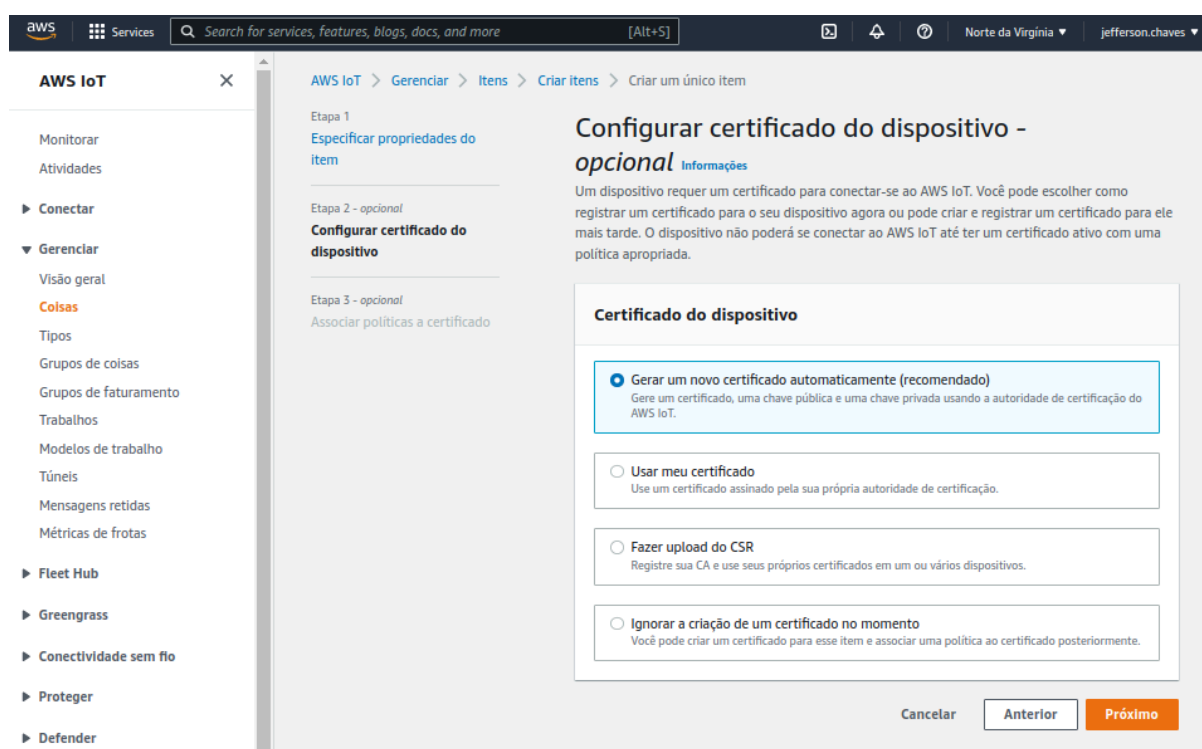


The screenshot shows the AWS IoT console interface. On the left is a navigation menu with categories like 'Monitorar', 'Atividades', 'Conectar', 'Gerenciar', 'Fleet Hub', 'Greengrass', 'Conectividade sem fio', 'Proteger', 'Defender', 'Agir', 'Testar', and 'Device Advisor'. The main content area is titled 'Especificar propriedades do item' (Specify item properties) and includes a breadcrumb trail: 'AWS IoT > Gerenciar > Itens > Criar itens > Criar um único item'. The page is divided into three steps: 'Etapa 1: Especificar propriedades do item' (active), 'Etapa 2 - opcional: Configurar certificado do dispositivo', and 'Etapa 3 - opcional: Associar políticas a certificado'. The 'Especificar propriedades do item' section contains a description of an item as a digital representation of a physical device. Below this are two main configuration sections: 'Propriedades do item' (Item Properties) and 'Sombra do dispositivo' (Device Shadow). The 'Propriedades do item' section has a text input for 'Nome da coisa' (Thing Name) with the value 'IFPR_ESP8266' and a note that names must be unique and cannot contain spaces. The 'Sombra do dispositivo' section has a description of device shadows and three radio button options: 'Nenhum shadow' (selected), 'Shadow com nome' (Create various shadows with different names), and 'Sombra sem nome (clássico)' (One item can have only one shadow without a name). At the bottom right are 'Cancelar' and 'Próximo' buttons.

*Em **Configurações adicionais**, não há necessidade de fazer alterações. Na opção **Sombra do dispositivo**, selecione a primeira opção como “Nenhum shadow”. Em seguida, clique em **Próximo**.*

3.3 Gerar certificado do dispositivo

Agora é necessário realizar a configuração do certificado para o dispositivo. Aqui você pode gerar automaticamente um novo certificado, usar seu próprio certificado, fazer upload do CSR ou pular esta etapa.



The screenshot shows the AWS IoT console interface. On the left is a navigation menu with categories like 'Monitorar', 'Atividades', 'Conectar', 'Gerenciar', 'Fleet Hub', 'Greengrass', 'Conectividade sem fio', 'Proteger', and 'Defender'. The 'Gerenciar' section is expanded, showing 'Visão geral', 'Coisas' (highlighted), 'Tipos', 'Grupos de coisas', 'Grupos de faturamento', 'Trabalhos', 'Modelos de trabalho', 'Túneis', 'Mensagens retidas', and 'Métricas de frotas'. The main content area is titled 'Configurar certificado do dispositivo - opcional' with a sub-link 'Informações'. It explains that a device needs a certificate to connect to AWS IoT and offers three options: 'Gerar um novo certificado automaticamente (recomendado)', 'Usar meu certificado', and 'Fazer upload do CSR'. A fourth option, 'Ignorar a criação de um certificado no momento', is also present. The 'Gerar um novo certificado automaticamente (recomendado)' option is selected. At the bottom right are buttons for 'Cancelar', 'Anterior', and 'Próximo'.

*A recomendação da AWS é gerar um novo certificado automaticamente. Selecione essa opção e em seguida clique em **Próximo**.*

3.3 Criar e anexar política

*Agora precisaremos anexar uma política às **coisas** criadas. Como nenhuma política foi definida ainda, é necessário a criação de política primeiro.*



*Então clique em **Criar política** (você será redirecionado para uma outra página). Aqui dê qualquer nome à política. Por exemplo, darei um nome como “ESP8266_POLICY”. A seção **Documento da política** permite que sejam permitidos ou restringidos acessos aos recursos do nosso broker por quem assinar essa política de segurança. Na coluna de ação da política, digite IoT. Assim, várias opções aparecerão. A partir daqui precisaremos permitir a conexão, publicação, assinatura, e recepção de dados.*

Documento da política Informações

Uma política do AWS IoT contém uma ou mais declarações de política. Cada declaração contém ações, recursos e um efeito que concede ou nega as ações pelos recursos.

Builder JSON

Efeito da política	Ação da política	Recurso da política	
Permitir ▼	iot:Connect ▼	*	Remover
Permitir ▼	iot:Receive ▼	*	Remover
Permitir ▼	iot:Publish ▼	*	Remover
Permitir ▼	iot:Subscribe ▼	*	Remover
Adicionar nova instrução			

Cancelar **Criar**

*Clique em criar para criar a política. Retorne para a opção Criar Coisa. Assim, uma opção política aparecerá. Precisamos anexar as políticas ao certificado. Então **selecione a política recém criada e clique em Criar Item.***

 Services

Search for services, features, blogs, docs, and more [Alt+S]

Norte da Virgínia jefferson.chaves

AWS IoT

Monitorar

Atividades

Conectar

Gerenciar

Visão geral

Coisas

Tipos

Grupos de coisas

Grupos de faturamento

Trabalhos

Modelos de trabalho

Túneis

Mensagens retidas

Métricas de frotas

AWS IoT > Gerenciar > Itens > Criar itens > Criar um único item

Etapa 1

Especificar propriedades do item

Etapa 2 - opcional

Configurar certificado do dispositivo

Etapa 3 - opcional

Associar políticas a certificado

Associar políticas a certificado - *opcional* Informações

As políticas do AWS IoT concedem ou negam acesso a recursos do AWS IoT. Anexar políticas ao certificado do dispositivo aplica esse acesso ao dispositivo.

Políticas (1/1) Atualizar Criar política

Filtrar políticas

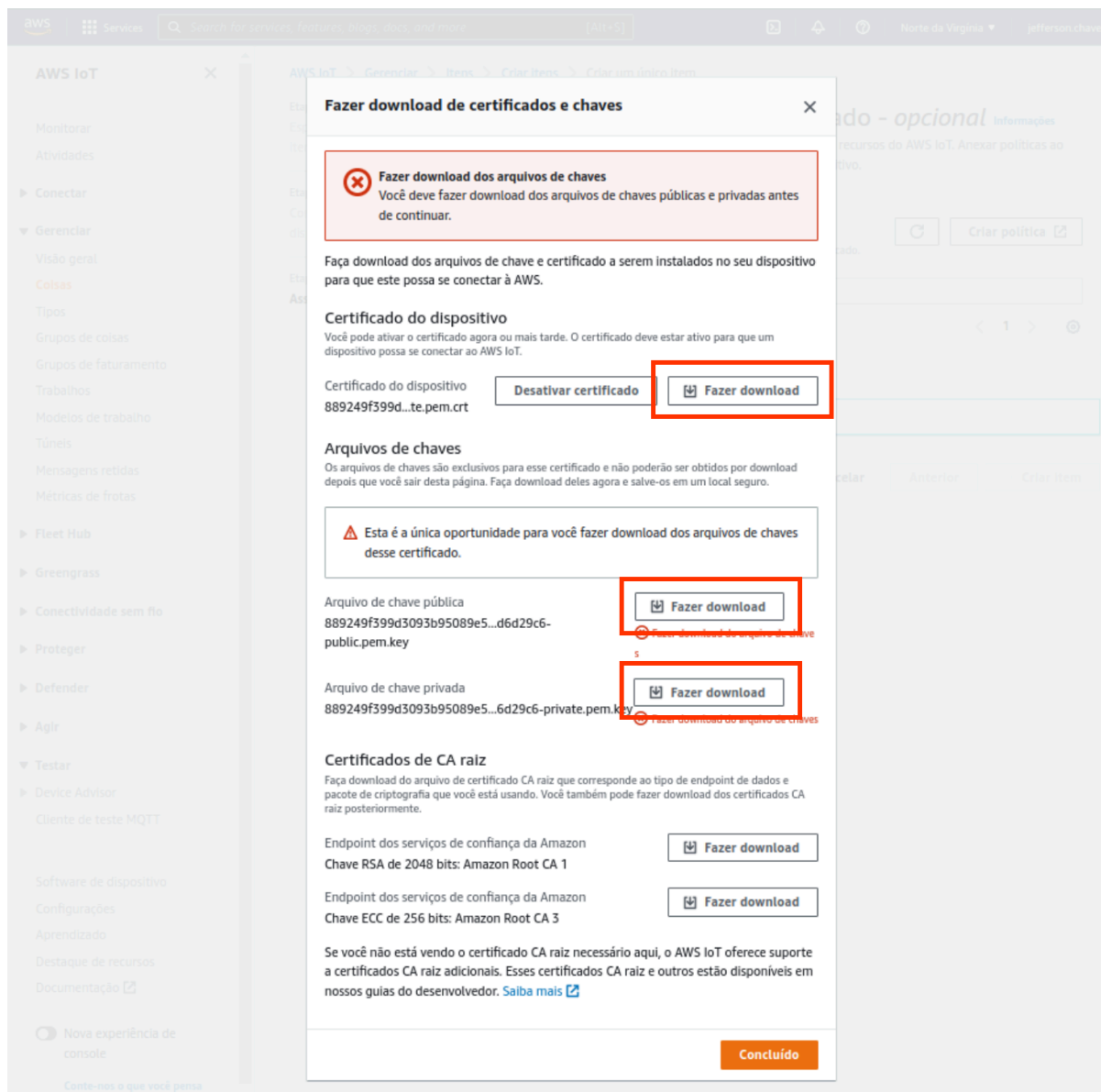
< 1 > ⚙

<input checked="" type="checkbox"/>	Nome
<input checked="" type="checkbox"/>	ESP8266_POLICY

Cancelar Anterior **Criar Item**

3.4 Download de certificados e chaves

Agora precisamos baixar os certificados necessários desta lista.



Fazer download de certificados e chaves

Fazer download dos arquivos de chaves
Você deve fazer download dos arquivos de chaves públicas e privadas antes de continuar.

Faça download dos arquivos de chave e certificado a serem instalados no seu dispositivo para que este possa se conectar à AWS.

Certificado do dispositivo
Você pode ativar o certificado agora ou mais tarde. O certificado deve estar ativo para que um dispositivo possa se conectar ao AWS IoT.

Certificado do dispositivo
889249f399d...te.pem.crt Desativar certificado **Fazer download**

Arquivos de chaves
Os arquivos de chaves são exclusivos para esse certificado e não poderão ser obtidos por download depois que você sair desta página. Faça download deles agora e salve-os em um local seguro.

Esta é a única oportunidade para você fazer download dos arquivos de chaves desse certificado.

Arquivo de chave pública
889249f399d3093b95089e5...d6d29c6-public.pem.key **Fazer download**

Arquivo de chave privada
889249f399d3093b95089e5...6d29c6-private.pem.key **Fazer download**

Certificados de CA raiz
Faça download do arquivo de certificado CA raiz que corresponde ao tipo de endpoint de dados e pacote de criptografia que você está usando. Você também pode fazer download dos certificados CA raiz posteriormente.

Endpoint dos serviços de confiança da Amazon
Chave RSA de 2048 bits: Amazon Root CA 1 **Fazer download**

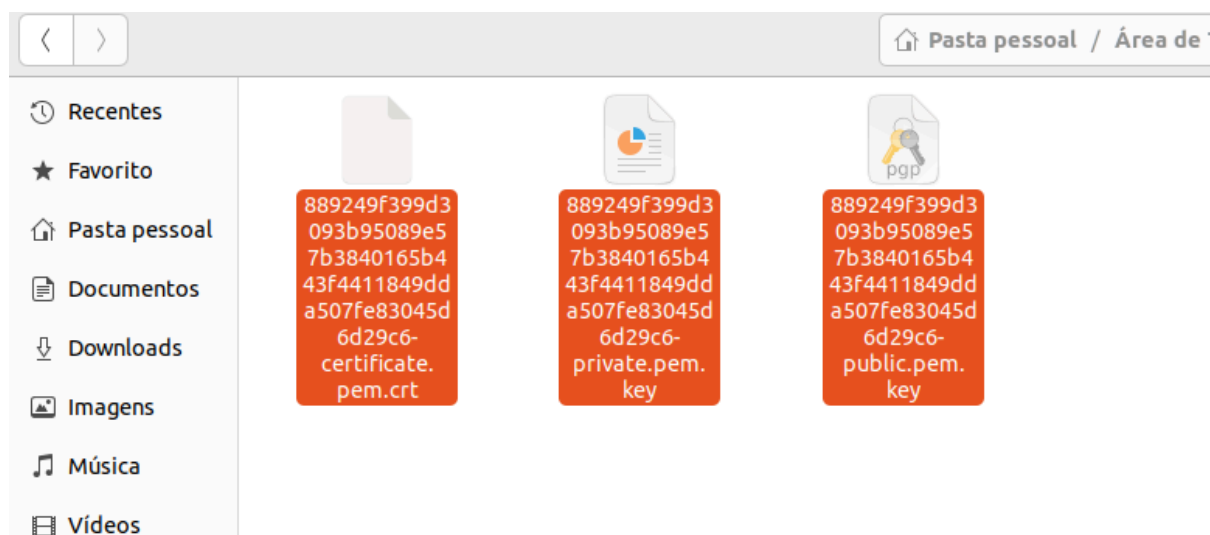
Endpoint dos serviços de confiança da Amazon
Chave ECC de 256 bits: Amazon Root CA 3 **Fazer download**

Se você não está vendo o certificado CA raiz necessário aqui, o AWS IoT oferece suporte a certificados CA raiz adicionais. Esses certificados CA raiz e outros estão disponíveis em nossos guias do desenvolvedor. [Saiba mais](#)

Concluído

Baixe os certificados para dispositivo, a chave pública e a chave privada. Nos certificados de CA raiz, há dois certificados. Para este exemplo, apenas o certificado Root

CAI é necessário, então baixe-o também. Atenção, não será possível baixar alguns desses certificados em outro momento.



Pronto! Sua coisa foi criada, e configurada. Dispositivos usando esses certificados poderão acessar, publicar e receber dados da plataforma AWS IoT.

4 . CONFIGURANDO O PLACA ESP8266 PARA CONEXÃO COM A AWS IOT

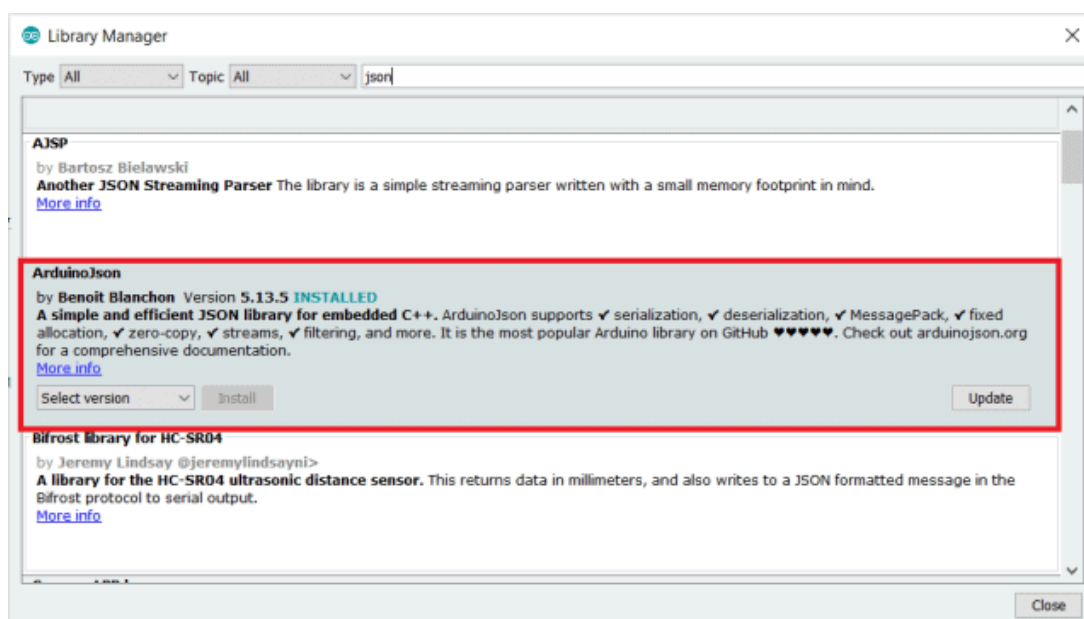
Nos passos seguintes, será realizada a configuração do dispositivo ESP8266 (NodeMCU) para acesso a plataforma AWS IoT utilizando a Arduino IDE em sua versão 1.8.19.

4.1 Instalando as Bibliotecas Arduino Necessárias

Neste passo, serão instaladas as bibliotecas que não são padrão para o ESP8266.

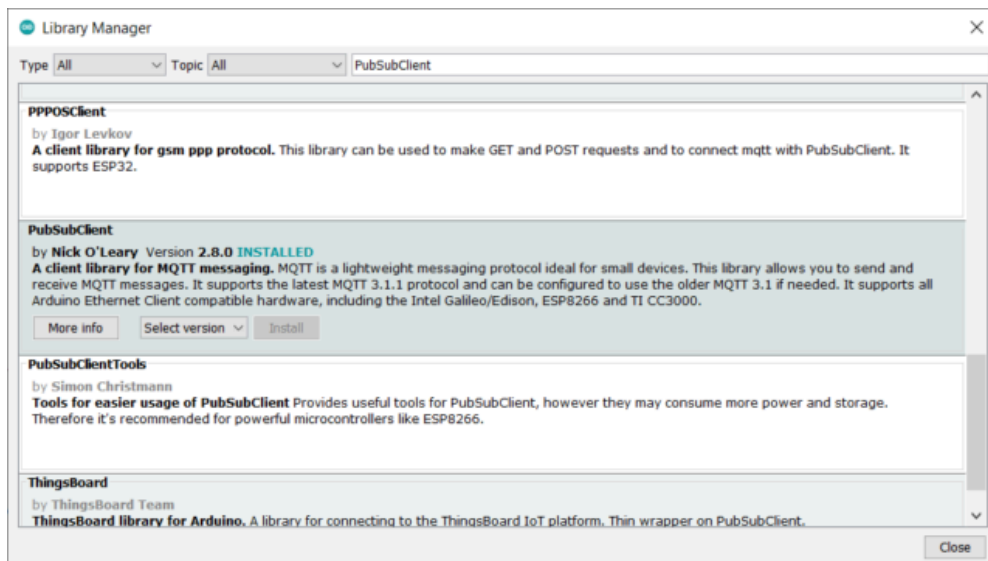
4.1.1. Biblioteca ArduinoJson

*Biblioteca utilizada para formatar objetos json. Acesse o gerenciador de bibliotecas e procure por “JSON” e instale a biblioteca **ArduinoJson**, do autor Benoit Blanchon, em sua versão 5.13.5, conforme mostrado na figura abaixo.*



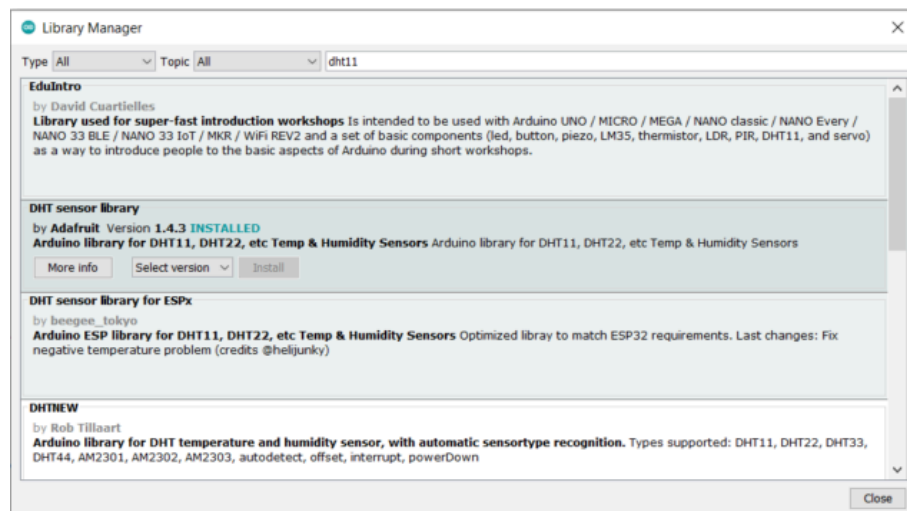
4.1.2. Biblioteca PubSubClient

Biblioteca que implementa um cliente MQTT, além de permitir seu acesso seguro (utilizando TLS). Acesse o gerenciador de bibliotecas e procure por “PubSubClient” e instale a biblioteca de Nick O'Leary, em sua versão 2.8.0.



4.1.3. Biblioteca de Sensores DHT11

Procure por “dht11” e instale a biblioteca *DHT Sensor Library*, da Adafruit, conforme mostrado abaixo.



4.2 Código-fonte/programa para conectar o AWS IoT Core ao ESP8266

O sketch que faz a interface do ESP8266 com o Sensor DHT11 e se conecta ao Amazon AWS IoT Core é escrito no Arduino IDE. O código é dividido em três arquivos para melhor compreensão e organização:

- i. main.ino*
- ii. connection_parameters.h*
- iii. secrets.h*

4.2.1 Arquivo main.ino

Abra um novo sketch no Arduino IDE e cole o código abaixo salvando-o em seguida.

```
#include <ESP8266WiFi.h>
#include <WiFiClientSecure.h>
#include <PubSubClient.h>
#include <ArduinoJson.h>
#include <time.h>
#include "DHT.h"

#include "connection_parameters.h"
#include "secrets.h"

#define DHTPIN 4           // Digital pin connected to the DHT sensor
#define DHTTYPE DHT11     // DHT 11

DHT dht(DHTPIN, DHTTYPE);

float h ;
float t;
unsigned long lastMillis = 0;
unsigned long previousMillis = 0;
//const long interval = 5000;
```

```
WiFiClientSecure secureConnection;

BearSSL::X509List cert(AWS_CERT_CA);
BearSSL::X509List client_cert(client_cert);
BearSSL::PrivateKey key(privkey);

PubSubClient clientMqtt(secureConnection);

time_t now;
time_t nowish = 1510592825;

//***** SETUP *****
void setup()
{
    Serial.begin(115200);

    wifiConnect();
    ntpConnect();
    connectAWS();
    dht.begin();
}

//***** CONEXAO AO WIFI *****
void wifiConnect(){

    WiFi.mode(WIFI_STA);
    WiFi.begin(WIFI_SSID, WIFI_PASSWORD);

    Serial.println(String("CONECTANDO AO SSID: ") + String(WIFI_SSID));

    while (WiFi.status() != WL_CONNECTED) {
        Serial.print(".");
        delay(1000);
    }
}

//***** CONEXAO AO NTP *****
void ntpConnect(void) {
```

```
Serial.println("\nCONFIGURANDO TIMESTAMP POR MEIO DE SNTP");

configTime(TIME_ZONE * 3600, 0 * 3600, "pool.ntp.org",
"time.nist.gov");

now = time(nullptr);

while (now < nowish) {
    delay(500);
    Serial.print(".");
    now = time(nullptr);
}

Serial.println("OBTIDO!");

struct tm timeinfo;
gmtime_r(&now, &timeinfo);
Serial.print("Current time: ");
Serial.print(asctime(&timeinfo));
}

//***** CONEXAO AWS IOT *****
void connectAWS() {

    secureConnection.setTrustAnchors(&cert);
    secureConnection.setClientRSACert(&client_crt, &key);

    clientMqtt.setServer(MQTT_HOST, 8883);
    clientMqtt.setCallback(messageReceived);

    Serial.println("\nCONECTANDO A AWS IOT");

    while (!clientMqtt.connect(THINGNAME)) {
        Serial.print(".");
        delay(1000);
    }

    if (!clientMqtt.connected()) {
        Serial.println("CONEXAO COM A AWS IoT FALHOU!");
        return;
    }
}
```

```
// Subscribe EM UM TÓPICO
clientMqtt.subscribe(AWS_IOT_SUBSCRIBE_TOPIC);

Serial.println("AWS IoT CONECTADA!");
}

void messageReceived(char *topic, byte *payload, unsigned int length) {

    Serial.print("RECEBIDO [");
    Serial.print(topic);
    Serial.print("]: ");

    for (int i = 0; i < length; i++) {
        Serial.print((char)payload[i]);
    }

    Serial.println();
}

void publishMessage() {

    StaticJsonDocument<200> doc;

    doc["time"]          = millis();
    doc["humidity"]      = h;
    doc["temperature"]   = t;

    char jsonData[512];

    serializeJson(doc, jsonData);

    clientMqtt.publish(AWS_IOT_PUBLISH_TOPIC, jsonData);
}

void loop()
{
    h = 80; //dht.readHumidity();
    t = 18; //dht.readTemperature();
```

```
// Verifica se alguma leitura falhou e sai (para tentar novamente).
if (isnan(h) || isnan(t) ) {

    Serial.println("FALHA COM A LEITURA DO SENSOR DHT!");
    return;

}

Serial.print(F("UMIDADE: "));
Serial.print(h);
Serial.print(F("%  TEMPERATURA: "));
Serial.print(t);
Serial.println(F("°C "));

delay(10000); //TEMPO ENTRE UMA PUBLICACAO E OUTRA

now = time(nullptr);

if (!clientMqtt.connected()) {
    connectAWS();
} else {

    clientMqtt.loop();

    if (millis() - lastMillis > 5000) {
        lastMillis = millis();
        publishMessage();
    }
}
}
```

4.2.2 Arquivo connection_parameters.h

```
const char WIFI_SSID[] = "<SSID>";
const char WIFI_PASSWORD[] = "<PASSWORD>";

#define THINGNAME "ESP8266"
#define AWS_IOT_PUBLISH_TOPIC    "ifpr/pub"
#define AWS_IOT_SUBSCRIBE_TOPIC "ifpr/sub"
```

```
int8_t TIME_ZONE = -3; //BR

const char MQTT_HOST[] = "<endpoint-broker-aws>";
```

4.2.2 Arquivo secrets.h

Abra uma nova guia no Arduino IDE, nomeie-a como secrets.h, e cole o seguinte código:

```
#include <pgmspace.h>
// Amazon Root CA 1
static const char AWS_CERT_CA[] PROGMEM = R"EOF(
-----BEGIN CERTIFICATE-----
-----END CERTIFICATE-----
)EOF";

// Copy contents from XXXXXXXX-certificate.pem.crt here ▼
static const char client_cert[] PROGMEM = R"KEY(
-----BEGIN CERTIFICATE-----
-----END CERTIFICATE-----

)KEY";

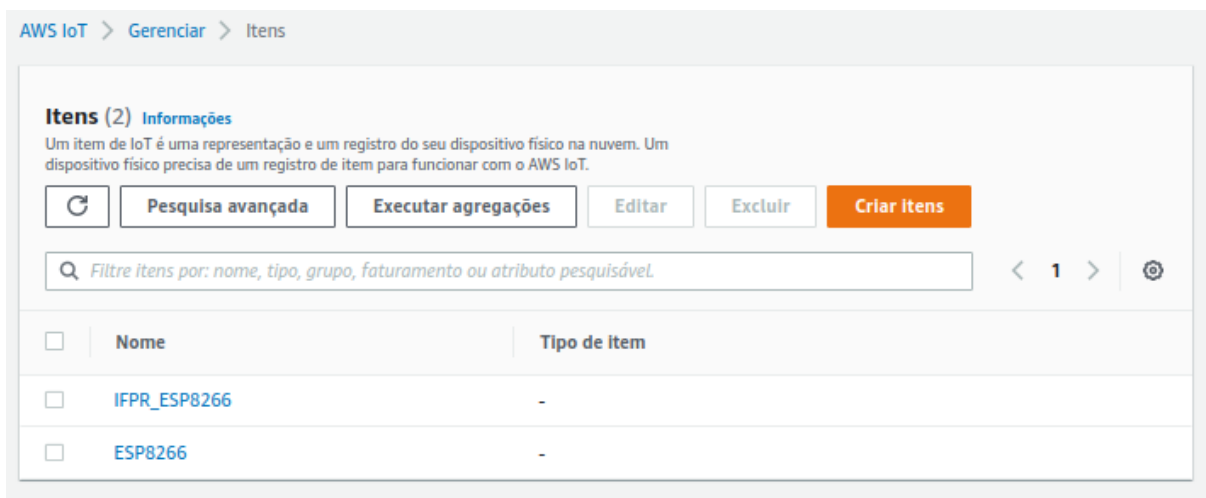
// Copy contents from XXXXXXXX-private.pem.key here ▼
static const char privkey[] PROGMEM = R"KEY(
-----BEGIN RSA PRIVATE KEY-----
-----END RSA PRIVATE KEY-----
)KEY";
```

4.2.3 Ajustando o Sketch de acordo com o ambiente e a coisa criada na AWS

Agora é hora de modificar o arquivo de esboço do Arduino. Vá para a guia secrets.h para realizar os ajustes necessários. Inicialmente ajuste as propriedades WIFI_SSID e WIFI_PASSWORD de acordo com as informações da sua rede local.

```
const char WIFI_SSID[] = "*****";  
const char WIFI_PASSWORD[] = "*****";
```


Será necessário incluir um nome de coisa. Você pode ir para a seção de coisas do Console AWS e copiar o nome da coisa.




AWS IoT > Gerenciar > Itens

Itens (2) Informações

Um item de IoT é uma representação e um registro do seu dispositivo físico na nuvem. Um dispositivo físico precisa de um registro de item para funcionar com o AWS IoT.

 **Pesquisa avançada** **Executar agregações** **Editar** **Excluir** **Criar itens**

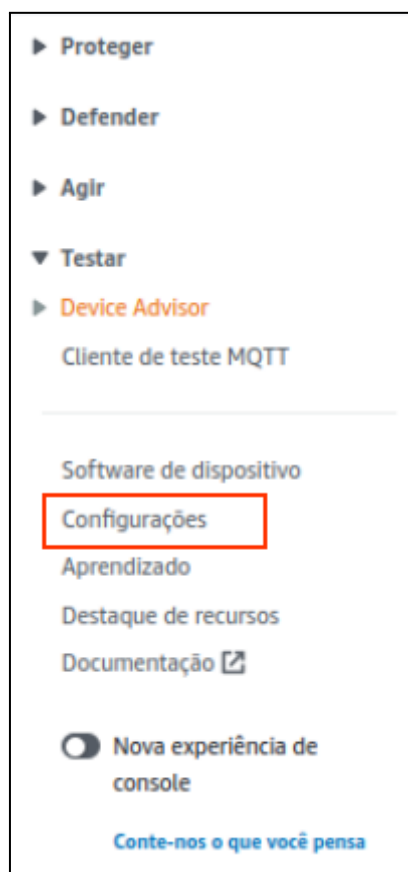
 Filtre itens por: nome, tipo, grupo, faturamento ou atributo pesquisável

<input type="checkbox"/>	Nome	Tipo de item
<input type="checkbox"/>	IFPR_ESP8266	-
<input type="checkbox"/>	ESP8266	-

Ajuste o nome da coisa na linha de código a seguir, de acordo com a linha de código abaixo:

```
#define THINGNAME "IFPR_ESP8266"
```

Será necessário inserir o endpoint da AWS IoT. Para obter o endpoint, acesse o menu de **configurações** do AWS Dashboard.



Clique no ícone de cópia para copiar o endpoint, volte para o Arduino IDE e cole-o na linha a seguir.

```
const char MQTT_HOST[] = "*****.*****.*****.amazonaws.com";
```

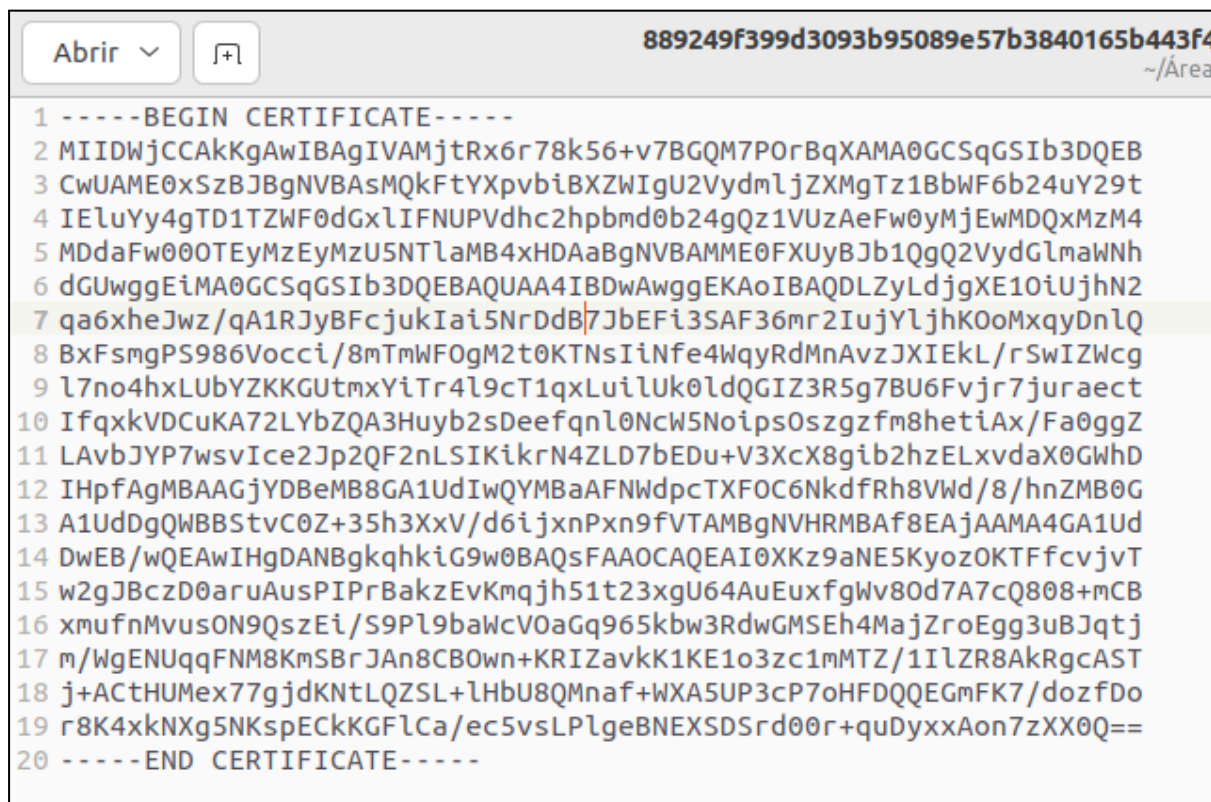
Pronto, agora é necessário configurar os certificados.

4.2.4 Ajuste dos certificados

Neste passo é necessário inserir o **Amazon Root CA1** entre a linha a seguir.

```
// Amazon Root CA 1
static const char AWS_CERT_CA[] PROGMEM = R"EOF(
-----BEGIN CERTIFICATE-----
-----END CERTIFICATE-----
)EOF";
```

Para tanto, abra o certificado **Amazon Root CA1** baixado anteriormente com um editor de texto qualquer.



```
889249f399d3093b95089e57b3840165b443f4
~/Área
Abrir
1 -----BEGIN CERTIFICATE-----
2 MIIDWjCCAKKgAwIBAgIVAMjtRx6r78k56+v7BGQM7P0rBqXAMA0GCSqGSIb3DQEB
3 CwUAME0xSzBjBgNVBAsMQkFtYXpvbiBXZWlU2VydmljZXMgTz1BbWF6b24uY29t
4 IEluYy4gTD1TZWF0dGx1IFNUPVdhc2hpbmd0b24gQz1VUzAeFw0yMjEwMDQxMzY4
5 MDdaFw00OTYyMzE1MjU0NTU0MjU0NTU0MjU0NTU0MjU0NTU0MjU0NTU0MjU0NTU0
6 dGUwggEiMA0GCSqGSIb3DQEBBAQUAA4IBDwAwggEKAoIBAQLDZyLdJgXE10iUjhN2
7 qa6xheJwz/qA1RjyBFcjukIai5NrDdB7JbEFi3SAF36mr2IujYljhK0oMxqyDnLQ
8 BxFsmgPS986Vocci/8mTmWFOgM2t0KTNSiInfe4WqyRdMnAvzJXIEkL/rSwIZWcg
9 l7no4hxLubYZKKGUtmxYiTr4l9cT1qxLuilUk0ldQGIz3R5g7BU6Fvj7juraect
10 IfqxkVDCuKA72LYbZQA3Huyb2sDeefqnL0NcW5Noips0szgzfm8hetiAx/Fa0ggZ
11 LAVbJYP7wsvIce2Jp2QF2nLSIKikrN4ZLD7bEDu+V3XcX8gib2hzELxvdaX0GWhD
12 IHpfAgMBAAGjYDBEMBM8GA1UdIwQYMBaAFNwdpcTXFOC6Nkdfrh8Vwd/8/hnZMB0G
13 A1UdDgQWBBStvc0Z+35h3XxV/d6ijxnPxn9fVTAMBgNVHRMBAf8EAJAAMA4GA1Ud
14 DwEB/wQEAwIHGDANBgkqhkiG9w0BAQsFAAOCAQEAI0XKz9aNE5KyoZ0KTFfcvjt
15 w2gJBczD0aruAusPIPrBakzEvKmqjh51t23xgU64AuEuxfgWv80d7A7cQ808+mCB
16 xmufnMvusON9QszEi/S9PL9baWcV0aGq965kbw3RdwGMSEh4MajZroEgg3uBJqtj
17 m/WgENUqqFNM8KmSBrJAn8CB0wn+KRIZavkK1KE1o3zc1mMTZ/1ILZR8AkRgcAST
18 j+ACTHUMex77gjdKNTLQZSL+lHbU8QMnaf+WXA5UP3cP7oHFDQQEGmFK7/dozfDo
19 r8K4xkNXg5NKspEckKGfLca/ec5vsLP1geBNEXSDSrd00r+quDyxxAon7zXX0Q==
20 -----END CERTIFICATE-----
```

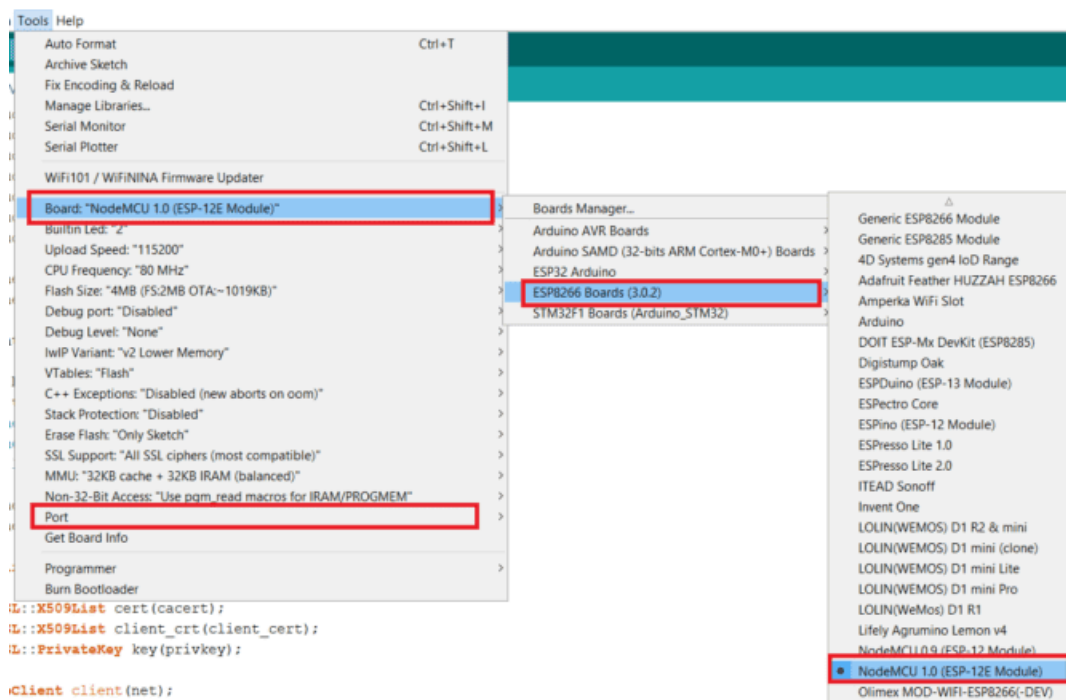
Em seguida, volte para o Arduino IDE e insira o texto copiado entre as tags - - - -
BEGIN CERTIFICATE - - - - e - - - - END CERTIFICATE - - - -. Configure os demais
certificados de acordo com as instruções presentes no Sketch.

4.2.3 Testando a publicação e assinatura de dados

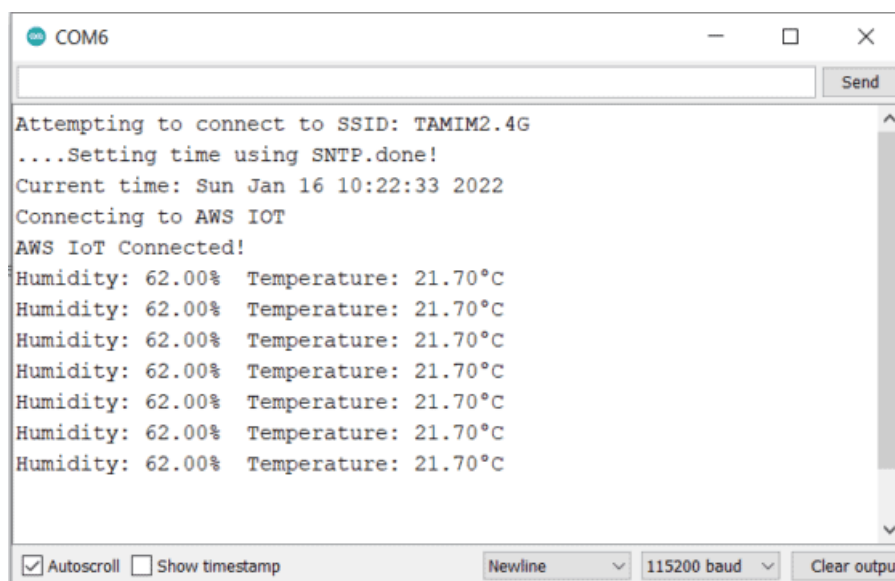
Realizada todas as modificações, conecte o NodeMCU ESP8266 ao seu computador.

Em seguida, vá para as ferramentas e selecione a placa NodeMCU 1.0 (ESP-12E Module).

Selecione também a porta adequada. Em seguida, clique na opção de upload para enviar o código para a placa ESP8266.



Uma vez feito o upload do código, abra o Serial Monitor. O ESP8266 tentará se conectar à rede WiFi. Depois de se conectar à rede WiFi, ele tentará se conectar ao AWS IoT.

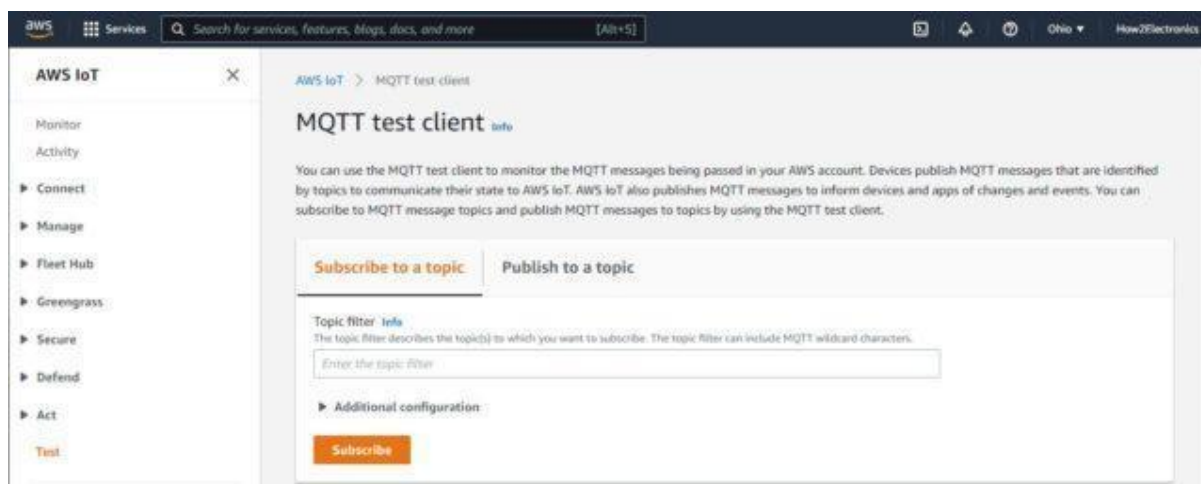


```
COM6
Attempting to connect to SSID: TAMIM2.4G
....Setting time using SNTP.done!
Current time: Sun Jan 16 10:22:33 2022
Connecting to AWS IOT
AWS IoT Connected!
Humidity: 62.00% Temperature: 21.70°C
Humidity: 62.00% Temperature: 21.70°C
Humidity: 62.00% Temperature: 21.70°C
Humidity: 62.00% Temperature: 21.70°C
Humidity: 62.00% Temperature: 21.70°C
Humidity: 62.00% Temperature: 21.70°C
Humidity: 62.00% Temperature: 21.70°C
```

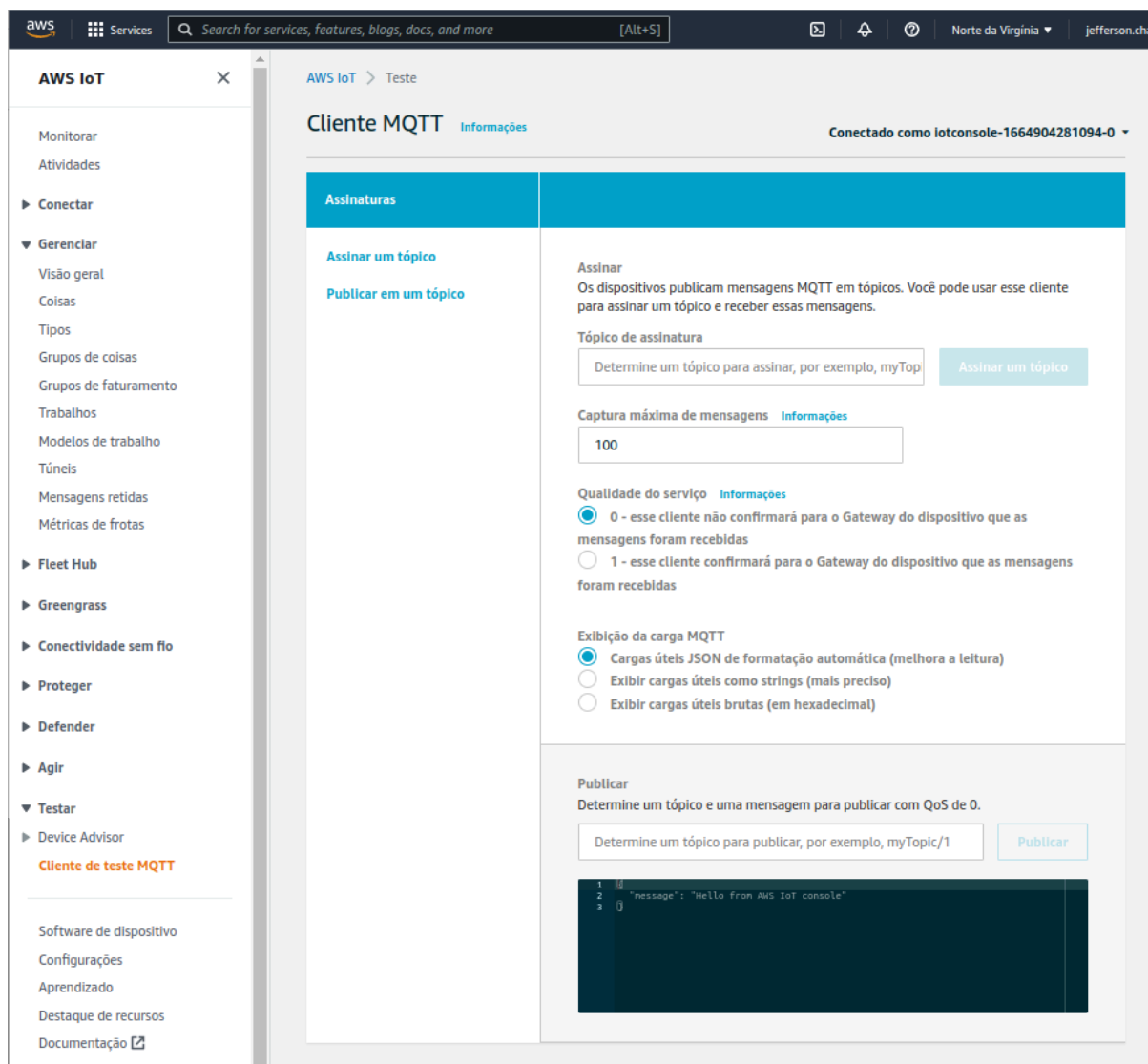
O valor de umidade e temperatura aparecerá no Monitor Serial.

4.2.3 Publicando dados do sensor no AWS Dashboard

*A mesma coisa também deve ser postada no AWS Server. Para verificar isso, vá para a seção **Testar** do AWS IoT. Na seção *Cliente de teste MQTT*, temos uma opção para assinar e publicar.*



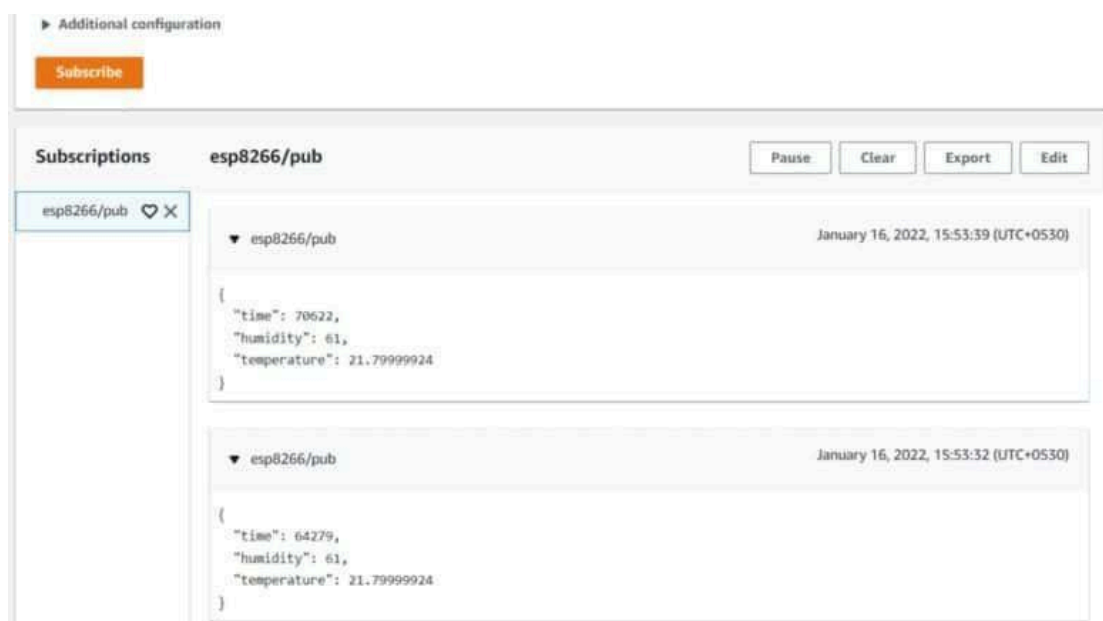
Agora, para ver os dados, você precisa se inscrever no tópico “ifpr/pub” na seção de filtros de tópicos. Na configuração adicional, você pode fazer alterações, se desejar.



The screenshot shows the AWS IoT console interface. On the left is a navigation menu with options like 'Monitorar', 'Atividades', 'Conectar', 'Gerenciar', 'Fleet Hub', 'Greengrass', 'Conectividade sem fio', 'Proteger', 'Defender', 'Agir', 'Testar', 'Device Advisor', and 'Cliente de teste MQTT'. The main area is titled 'Cliente MQTT' and shows a 'Assinaturas' (Subscriptions) section. It includes a form to 'Assinar um tópico' (Subscribe to a topic) with a text input field and a button. Below this, there are settings for 'Captura máxima de mensagens' (Maximum message capture) set to 100, and 'Qualidade do serviço' (Service quality) set to 0. There is also a section for 'Exibição da carga MQTT' (MQTT payload display) with options for JSON, strings, or raw hex. At the bottom, there is a 'Publicar' (Publish) section with a text input field and a button. The console also shows a terminal window with a message: 'Hello from AWS IoT console'.

Em seguida, clique em inscrever-se. Ao clicar no botão de inscrição, imediatamente os dados do ESP8266 serão carregados no AWS IoT.

Você enviou com êxito os dados de temperatura e umidade do sensor DHT11 para a Amazon AWS IoT Core usando o ESP8266.



Os dados são atualizados aqui após um intervalo de um segundo. Podemos receber os dados na AWS IoT Core a partir do ESP8266 por meio do protocolo MQTT. É assim que vemos os dados inscritos.

4.2.3 Lendo dados no Serial Monitor

O objetivo desse passo é publicar em um tópico na AWS IoT e receber essa publicação.

MQTT test client [Info](#)

You can use the MQTT test client to monitor the MQTT messages being passed in your AWS account. Devices publish MQTT messages that are identified by topic. The AWS IoT console also publishes MQTT messages to inform devices and apps of changes and events. You can subscribe to MQTT message topics and publish MQTT messages.

Subscribe to a topic **Publish to a topic**

Topic name
The topic name identifies the message. The message payload will be published to this topic with a Quality of Service (QoS) of 0.

Q esp8266/sub

Message payload

```
{
  "message": "Hello from AWS IoT console"
}
```

► **Additional configuration**

Publish

Para essa interação será necessário publicar no tópico “ifpr/sub” na seção de filtros de tópicos. Configurações adicionais são opcionais. Clique em publicar. Após o ciclo do ESP8266, você verá a mensagem enviada no Serial Monitor.

Subscribe to a topic **Publish to a topic**

Topic name
The topic name identifies the message. The message payload will be published to this topic with a Quality of Service (QoS) of 0.

Q esp8266/sub

Message payload

```
{
  "message": "I love You"
}
```

► **Additional configuration**

Publish

COM6

Send

```
Humidity: 61.00% Temperature: 21.80°C
Received [esp8266/sub]: {
  "message": "HOW2ELECTRONICS.COM"
}
Humidity: 61.00% Temperature: 21.80°C
Humidity: 61.00% Temperature: 21.80°C
Humidity: 61.00% Temperature: 21.80°C
Humidity: 61.00% Temperature: 21.80°C
Humidity: 61.00% Temperature: 21.80°C
Humidity: 61.00% Temperature: 21.80°C
Humidity: 61.00% Temperature: 21.80°C
Received [esp8266/sub]: {
  "message": "I love You"
}
```

☐ Autoscroll ☐ Show timestamp Newline 115200 baud

Para concluir este texto, salienta-se que poderá ser enviado ou recebido dados do Amazon AWS IoT Core usando para um ESP8266 ou vice-versa. Claro que trata-se de um exemplo, e outras placas, tais como ESP32, Arduino e etc., poderão ser utilizadas para realizar comunicação de com a AWS IoT. Além disso, a AWS IoT também disponibiliza SDK's

que realizam essa dinâmica para a conexão de forma análoga. Usando o AWS MQTT, podemos assinar tópicos para leituras de sensores publicados por vários nós de IoT .