 Estácio	Universidade Estácio Campus Santa Cruz - RJ Curso de Desenvolvimento Full Stack Relatório da Missão Prática 5 - Mundo 4
Disciplina:	RPG 0027 - Vamos interligar as coisas com a nuvem
Nome:	Lukas Cauã Oliveira Xavier
Turma:	2023.2

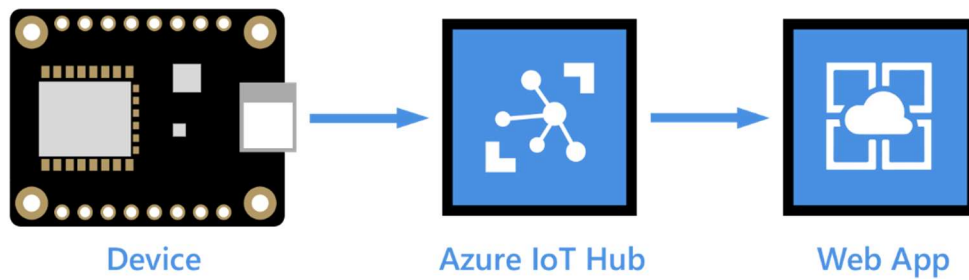
Vamos interligar as coisas com a nuvem

2º Objetivo da Prática:

Nessa atividade revisaremos tudo o que utilizamos nas microatividades anteriores. Além disso, veremos também que os [Aplicativos Lógicos do Azure](#) podem ajudar você a orquestrar fluxos de trabalho em serviços locais e de nuvem, em uma ou mais empresas e em vários protocolos.

- Contextualização

Nesta Missão Prática você aprenderá a visualizar dados em tempo real provenientes de sensores conectados ao seu hub IoT. Isso será realizado por meio da execução de um aplicativo web Node.js em seu computador local. Após a configuração bem-sucedida e execução do aplicativo web local, você terá a opção de hospedá-lo no Serviço de Aplicativo do Azure para facilitar o acesso e a escalabilidade. O fluxo de dados seguirá o caminho delineado na figura abaixo. O dispositivo simulado coletará dados de temperatura e umidade, os quais serão enviados para o Azure IoT Hub e exibidos através do Serviço de Aplicativo do Azure (Web App).



Quando dados de telemetria são recebidos pelo ponto de extremidade do Hub IoT é

possível, além de gerar a visualização em tempo real, configurar um aplicativo lógico

que pode desencadear uma série de ações. Estas incluem o armazenamento dos dados

em um blob no Armazenamento do Azure, o envio de alertas por e-mail em casos de

anomalias nos dados e até mesmo a programação de visitas técnicas em resposta a

falhas relacionadas pelo dispositivo.

As mensagens recebidas pelo seu hub IoT seguem um formato semelhante ao apresentado abaixo, contendo os dados de telemetria no corpo (body) e a propriedade

temperatureAlert nas propriedades do aplicativo (applicationProperties). As propriedades

do sistema não são exibidas.

3º Resultados:

javascript

const wpi = require('wiring-pi');

const Client = require('azure-iot-device').Client; const Message = require('azure-iot-device').Message;

const Protocol = require('azure-iot-device-mqtt').Mqtt; const BME280 = require('bme280-sensor');

const BME280_OPTION = {

```

i2cBusNo: 1,
i2cAddress: BME280.BME280_DEFAULT_I2C_ADDRESS()
};
const connectionString = 'HostName=nome-do-seu-hub.azure
devices.net;DeviceId=dispositivo 001;SharedAccessKey=xxxxxxxxxxxxxx';
const LEDPin = 4;
var sendingMessage = false; var messageId = 0;
var client, sensor;
var blinkLEDDTimeout = null; function getMessage(cb) { messageId++;
sensor.readSensorData()
.then(function (data) { cb(JSON.stringify({ messageId: messageId,
deviceId: 'Raspberry Pi Web Client', temperature: data.temperature_C,
humidity: data.humidity
}), data.temperature_C > 30);
})
.catch(function (err) {
console.error('Failed to read out sensor data: ' + err);
});
}

function sendMessage() {
if (!sendingMessage) { return; }

getMessage(function (content, temperatureAlert) { var message = new
Message(content); message.properties.add('temperatureAlert',
temperatureAlert.toString()); console.log('Sending message: ' + content);
client.sendEvent(message, function (err) { if (err) {
console.error('Failed to send message to Azure IoT Hub');
} else { blinkLED();

```

```
console.log('Message sent to Azure IoT Hub');
}
});
});
}
```

```
function onStart(request, response) {
console.log('Try to invoke method start(' + request.payload + ')');
sendingMessage = true;
```

```
response.send(200, 'Successfully start sending message to cloud',
function (err) {
if (err) {
console.error('[IoT hub Client] Failed sending a method response:\n' +
err.message);
}
});
}
```

```
function onStop(request, response) {
console.log('Try to invoke method stop(' + request.payload + ')');
sendingMessage = false;
```

```
response.send(200, 'Successfully stop sending message to cloud',
function (err) {
if (err) {
console.error('[IoT hub Client] Failed sending a method response:\n' +
err.message);
}
});
}
```

```
function receiveMessageCallback(msg) { blinkLED();
var message = msg.getData().toString('utf-8'); client.complete(msg,
function () { console.log('Receive message: ' + message);
});
}
```

```

function blinkLED() {
if (blinkLEDTIMEOUT) { clearTimeout(blinkLEDTIMEOUT);
}
wpi.digitalWrite(LEDpin, 1); blinkLEDTIMEOUT = setTimeout(function () {
wpi.digitalWrite(LEDpin, 0);
}, 500);
}

wpi.setup('wpi'); wpi.pinMode(LEDpin, wpi.OUTPUT);
sensor = new BME280(BME280_OPTION); sensor.init()
.then(function () { sendingMessage = true;
})
.catch(function (err) { console.error(err.message || err);
});
client = Client.fromConnectionString(connectionString, Protocol); client.open(function
(err) {
if (err) {
console.error('[IoT hub Client] Connect error: ' + err.message); return;
}

client.onDeviceMethod('start', onStart); client.onDeviceMethod('stop', onStop);
client.on('message', receiveMessageCallback); setInterval(sendMessage, 2000);
});

```

3.1ª Imagens:

The screenshot displays the Microsoft Azure portal interface for an IoT Hub named 'fullstack1'. The main section, titled 'Dispositivos', shows a list of devices. A single device, 'Fullstack_dispositivo', is listed with the type 'Dispositivo IoT' and status 'Habilitado'. The left-hand navigation pane includes various management tools such as 'Visão geral', 'Log de atividade', 'IAM (Controle de acesso)', 'Gerenciamento de dispositivo', and 'Dispositivos'.

ID do Dispositivo	Tipo	Status	Última atualização do status	Tipo de autenticação	Mensagens C2D em fila	Marcas
Fullstack_dispositivo	Dispositivo IoT	Habilitado	--	Assinatura de Acesso Compartilhado	0	

Microsoft Azure

Pesquisar recursos, serviços e documentos (G+)

Copilot

202305450556@alunos... EDUCACIONAL

Página inicial >

fullstack1 Hub IoT

Pesquisar

Mover Excluir Atualizar Comentários

Visão geral

Log de atividade

IAM (Controle de acesso)

Marcações

Diagnosticar e resolver problemas

Eventos

Visualizador de recursos

Gerenciamento de dispositivo

Dispositivos

IoT Edge

Configurações + implantações

Atualizações

Consultas

Configurações do hub

Configurações de segurança

Defensor para IoT

Monitoramento

Automação

Ajuda

Fundamentos

Grupo de recursos (mover) : Win.group

Status : Active

Local : East US

Região do serviço : East US

Assinatura (mover) : Azure for Students

Marcações (editar) : Adicionar marcas

Veja mais

Uso Introdução

Nome do host : fullstack1.azure-devices.net

Nível : Free

Limite diário de mensagem... : 8.000

Versão Mínima do TLS : 1.2

Exibição JSON

Mostrar dados para o último: 1 Hora 6 Horas 12 Horas 1 Dia 7 Dias 30 Dias

Uso do Hub IoT

- Mensagens usadas hoje: 294
- Cota de mensagens diárias: 8000
- Dispositivos de IoT: 1

Número de mensagens usadas

Mensagens do dispositivo para a nuvem

Total number of messages used (Máx., fullstack1) : 319

Telemetry messages sent (Soma, fullstack1) : 849

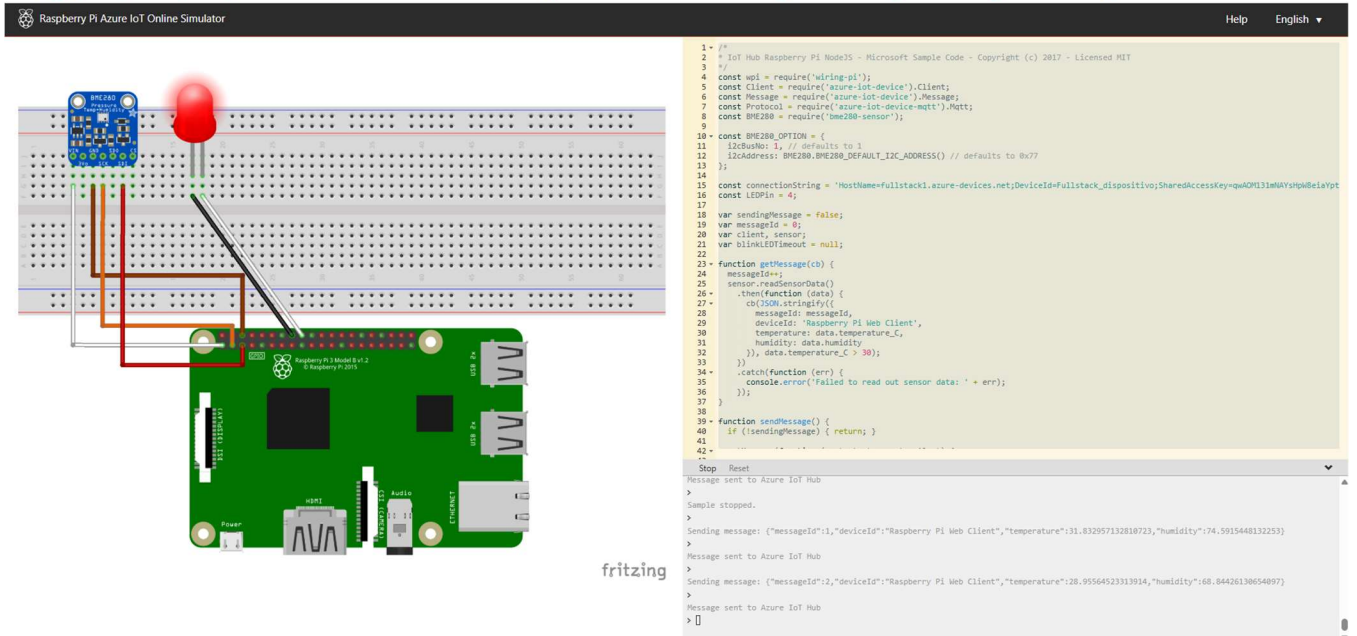
https://portal.azure.com/#

azure-samples.github.io/raspberry-pi-web-simulator/GetStarted

Gmail YouTube Maps Download iObit Fre... Mensagens Indeeds AP APInfo - O ponto d... Instagram Webcam IP Sci-Fi - Página 4... 9:40 Tocando agora... 5:07 Tocando agora... https://bit.ly/3Ozph... 1 Hs Cabeça Desbu... Todos os favoritos

Raspberry Pi Azure IoT Online Simulator

Help English



```

1 //
2 // IoT Hub Raspberry Pi NodeJS - Microsoft Sample Code - Copyright (c) 2017 - Licensed MIT
3
4 const wpi = require('wiring-pi');
5 const client = require('azure-iot-device').Client;
6 const Message = require('azure-iot-device').Message;
7 const Protocol = require('azure-iot-device-mqtt').Mqtt;
8 const BME280 = require('bme280-sensor');
9
10 const BME280_OPTION = {
11   i2cBusNo: 1, // defaults to 1
12   i2cAddress: BME280.BME280_DEFAULT_I2C_ADDRESS() // defaults to 0x77
13 };
14
15 const connectionString = 'HostName=fullstack1.azure-devices.net;DeviceId=fullstack1_dispositivo;SharedAccessKey=qwA0K13lmAYspWBeIaYpT';
16 const LEOPin = 4;
17
18 var sendingMessage = false;
19 var messageId = 0;
20 var client;
21 var blinkLEDFrequency = null;
22
23 function getMessage(cb) {
24   messageId++;
25   sensor.readSensorData()
26     .then(function (data) {
27       cb(JSON.stringify({
28         messageId: messageId,
29         deviceId: 'Raspberry Pi Web Client',
30         temperature: data.temperature_C,
31         humidity: data.humidity,
32       })), data.temperature_C > 30);
33     })
34     .catch(function (err) {
35       console.error('Failed to read out sensor data: ' + err);
36     });
37 }
38
39 function sendMessage() {
40   if (!sendingMessage) { return; }
41 }
42

```

Stop Reset

Message sent to Azure IoT Hub

Sample stopped.

Sending message: ("messageId":1,"deviceId":"Raspberry Pi Web Client","temperature":31.832957132818723,"humidity":74.5915448132253)

Message sent to Azure IoT Hub

Sending message: ("messageId":2,"deviceId":"Raspberry Pi Web Client","temperature":28.95564523313914,"humidity":68.84426130654897)

Message sent to Azure IoT Hub

fritzing

1 device

Fullstack_dispositivo



Temperature & Humidity Real-time Data



4º Conclusão:

Conseguí visualizar dados de sensores em tempo real, tanto localmente quanto no Azure, através de um aplicativo web Node.js. Configurei o IoT Hub e o Serviço de Aplicativo, e aprendi a implantar um aplicativo web na nuvem. Usei a CLI do Azure para configurar os serviços e variáveis de ambiente. O resultado final foi a exibição de dados de temperatura e umidade em um gráfico no navegador, demonstrando o fluxo de dados do dispositivo simulado até a nuvem.

A implantação bem-sucedida me permitiu acessar os dados de qualquer lugar, um aspecto fundamental para aplicações de IoT. Além disso, a exploração dos Aplicativos Lógicos abrindo possibilidades para automatizar tarefas com base nos dados recebidos. Em resumo, a atividade me proporcionou uma experiência completa, desde a configuração local até a implantação na nuvem, consolidando meu conhecimento em IoT e Azure.