Disciplina: Missão Prática | Nível 5 | Mundo 4
Semestre: 2º - 2024
Aluno: DAVIDSON PEREIRA DE OLIVEIRA
Matrícula: 2022.11.54585-5
Repositório no GIT: dev-davidson/Missao-Pratica-Nivel-5-Mundo-4 (github.com)

Relatório discente de acompanhamento

RPG0027 - Vamos interligar as coisas com a nuvem

EXECUÇÃO:

1. **Foi adicionado um grupo de consumidores ao hub IoT** para fornecer as visualizações independentes do fluxo de eventos. Essas visualizações permitem que aplicativos e serviços do Azure consumam dados do mesmo ponto de extremidade do hub de eventos.

   Nesta etapa, foi configurado um grupo de consumidores no ponto de extremidade integrado do seu **IoT Hub ClimateMonitorHub**, que será utilizado pelo aplicativo web para ler dados.

   O seguinte comando na CLI do Azure serve para adicionar um grupo de consumidores ao ponto de extremidade interno do hub IoT:

```
az iot hub consumer-group create --hub-name ClimateMonitorHub --name YOUR_CONSUMER_GROUP_NAME
```

2. **Obter uma cadeia de conexão de serviço para seu hub IoT** é um passo crucial para conceder permissões necessárias a serviços que precisam ler e gravar nos pontos de extremidade do hub IoT.

O seguinte comando obtém uma cadeia de conexão para o hub IoT, seguindo a política de serviço:

```
az iot hub show-connection-string --hub-name ClimateMonitorHub --policy-name service
```

3. **Configuração das variáveis de ambiente:**

   Foi definido as variáveis de ambiente, utilizado os comandos abaixo na janela de comando, substituindo os espaços reservados pela cadeia de conexão de serviço para o hub IoT e o nome do

```
export IOT_HUB_CONN_STRING="YOUR_IOT_HUB_CONNECTION_STRING"
export EVENT_HUB_CONSUMER_GROUP="YOUR_CONSUMER_GROUP_NAME"
```

   grupo de consumidores criado anteriormente.

4. **Executar o aplicativo localmente:**

   Com todo o ambiente configurado, agora é possível visualizar a coleta de dados através de um navegador.
   Para visualizar os dados do hub IoT, foi criado uma página da web no localhost:

```
http://localhost:3000
```

5. **Criar um plano de Serviço de Aplicativo:**

   Um plano de Serviço de Aplicativo define um conjunto de recursos de computação para a execução de um aplicativo hospedado no Serviço de Aplicativo. Foi criado um novo plano do Serviço de Aplicativo usando o nível gratuito do Windows, utilizando o comando:

```
az appservice plan create --name ClimateMonitorHubPlan --resource-group YOUR_RESOURCE_GROUP --sku FREE
```

6. **Adicionar configurações de aplicativo no Web App:**

   O comando az webapp config appsettings set utilizado para adicionar configurações de aplicativo referentes às variáveis de ambiente que especificam a cadeia de conexão do hub IoT e o grupo de consumidores do hub de eventos:

```
az webapp config appsettings set -n ClimateMonitorHubApp -g YOUR_RESOURCE_GROUP_NAME --settings
EventHubConsumerGroup=YOUR_CONSUMER_GROUP_NAME IotHubConnectionString="YOUR_IOT_HUB_CONNECTION_STRING"
```

7. **Consultar o status do Web App:**

   O comando a seguir foi utilizado para consultar o status do aplicativo web e assegurar-se de que está em execução:

```
az webapp show --name ClimateMonitorHubApp --resource-group YOUR_RESOURCE_GROUP
```

8. **Acessar o Web App hospedado no Azure:**

   Após toda a configuração do Serviço de Aplicativo, pode ser acessado através do endereço:

   https://ClimateMonitorHub.azurewebsites.net/

CÓDIGO PYTHON DO PROJETO CLIMATEMONITORHUB INTEGRADO:

ClimateMonitorHub.py > monitorar_ambiente

```python
# Nome do projeto: ClimateMonitorHub
# Programador: Davidson Oliveira
import asyncio
import random
import uuid
from azure.iot.device.aio import IoTHubDeviceClient
from azure.iot.device import Message

CONEXAO_DISPOSITIVO = "HostName=ClimateMonitorHub.azure-devices.net;DeviceId=Sensor_Climate_001;SharedAccessKey=YourAccessKeyHere"

INTERVALO_ENVIO = 2  # Enviar mensagens a cada 2 segundos
TEMPO_MAX_ENVIO = 30  # Limite de tempo de 30 segundos para envio de mensagens


TEMPERATURA_BASE = 22.0  # Temperatura (graus Celsius)
UMIDADE_BASE = 55.0  # Umidade (%)

LIMITE_ALERTA_TEMPERATURA = 28.0

TEMPLATE_MENSAGEM = '{"temperatura": %.2f, "umidade": %.2f, "alerta": "%s"}'

async def monitorar_ambiente():
    try:

        cliente = IoTHubDeviceClient.create_from_connection_string(CONEXAO_DISPOSITIVO)
        await cliente.connect()

        print("Monitorando ambiente e enviando dados para o ClimateMonitorHub...")

        while True:

            temperatura = TEMPERATURA_BASE + random.uniform(-5, 5)
            umidade = UMIDADE_BASE + random.uniform(-10, 10)
            alerta = "SIM" if temperatura > LIMITE_ALERTA_TEMPERATURA else "NÃO"

            conteudo_mensagem = TEMPLATE_MENSAGEM % (temperatura, umidade, alerta)
            mensagem = Message(conteudo_mensagem)

              mensagem.message_id = str(uuid.uuid4())
              mensagem.content_type = "application/json"
              mensagem.custom_properties["alerta"] = alerta

              print(f"Enviando: {conteudo_mensagem}")
              await cliente.send_message(mensagem)
              await asyncio.sleep(INTERVALO_ENVIO)

    except Exception as e:
        print(f"Erro ao conectar ou enviar mensagem: {e}")
    finally:
        await cliente.shutdown()
        print("Conexão com o ClimateMonitorHub finalizada")

if __name__ == '__main__':
    try:
        asyncio.run(monitorar_ambiente())
    except KeyboardInterrupt:
        print("Monitoramento interrompido pelo usuário.")

```

CONCLUSÃO

O projeto **ClimateMonitorHub** proporcionou uma valiosa oportunidade de aprendizado e prática na integração de dispositivos IoT com serviços de nuvem. A execução deste projeto envolveu a configuração e utilização de várias tecnologias e serviços, incluindo o Azure IoT Hub e o Azure App Service, e forneceu uma compreensão prática de conceitos essenciais no desenvolvimento de soluções IoT.

Principais Aprendizados:

1. **Integração de Dispositivos IoT:**
   - **Conexão de Dispositivos:** Aprendi a conectar um dispositivo simulado ao Azure IoT Hub usando o SDK Python. Isso envolveu o envio de dados de sensores (como temperatura e umidade) para a nuvem e a configuração de mensagens com propriedades personalizadas.

2. **Desenvolvimento de Aplicações Web:**
   - **Criação de Aplicações Web:** Desenvolvi um aplicativo web em Node.js para visualizar dados em tempo real. Este aplicativo consumiu dados do Azure IoT Hub e apresentou essas informações em uma interface gráfica acessível localmente e na nuvem.

3. **Configuração e Gerenciamento de Serviços Azure:**
   - **Configuração de Serviços:** Adquiri experiência prática na configuração de grupos de consumidores, variáveis de ambiente e planos de serviço no Azure. Também aprendi a gerenciar e configurar aplicativos web no Azure App Service, incluindo a configuração de variáveis de ambiente e a publicação do aplicativo.

4. **Práticas de DevOps e Escalabilidade:**
   - **Implantação e Escalabilidade:** A experiência de hospedar o aplicativo web no Azure destacou a importância de práticas de DevOps e escalabilidade para aplicativos em nuvem. Aprendi a implantar e monitorar aplicativos web, garantindo que eles estivessem prontos para operações em larga escala.

Reflexão Final:

Este projeto não apenas reforçou o conhecimento teórico sobre IoT e serviços de nuvem, mas também forneceu uma aplicação prática dessas tecnologias. A experiência adquirida na configuração de dispositivos IoT, desenvolvimento de aplicativos web e gerenciamento de serviços na nuvem será fundamental para futuros projetos e desenvolvimento profissional.