CS2032 - Cloud Computing (Ciclo 2024-2)
Event-driven architecture
Semana 14 - Taller 4: SQS - Simple Queue Service

ELABORADO POR: GERALDO COLCHADO

Contenido

Event-driven architecture

1. **Objetivo del taller 4**
2. Ejercicio 1: Evento Nueva Lectura Sensor IoT
3. Ejercicio 2: Ejercicio propuesto
4. Cierre

Event-driven architecture
Objetivo del Taller 4

- Diseño e implementación de una Arquitectura de Solución basada en eventos con el servicio “SNS - Simple Notification Service” y “SQS - Simple Queue Service”.

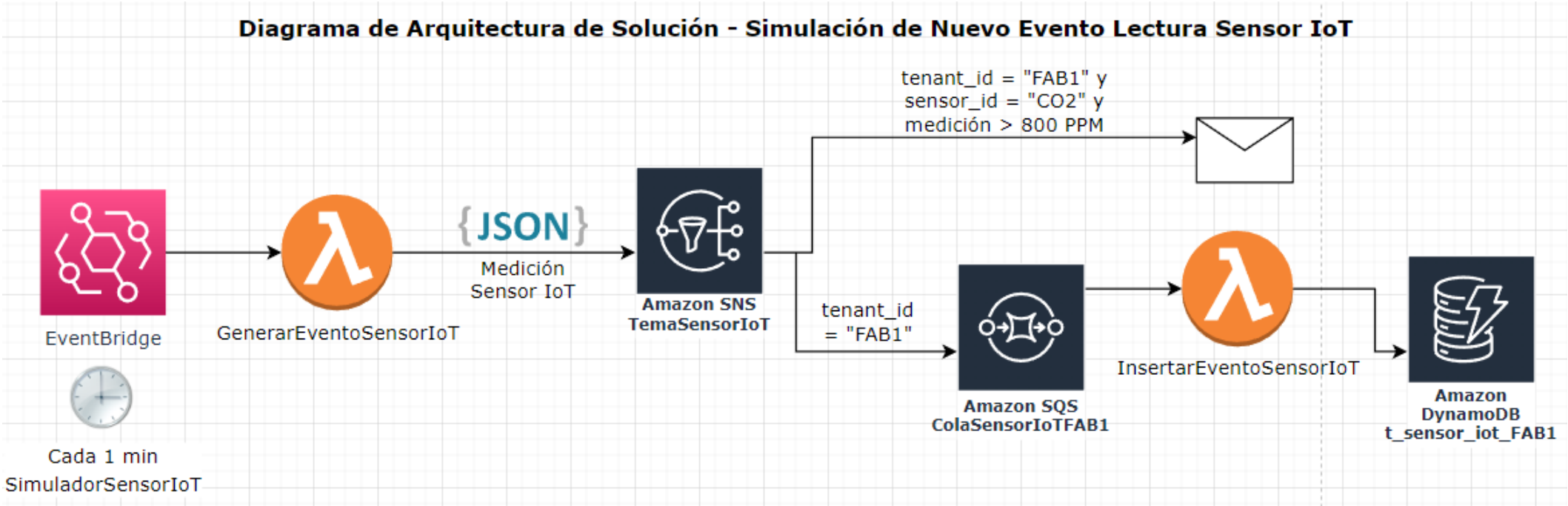
Contenido

Event-driven architecture

1. Objetivo del taller 4
2. **Ejercicio 1: Evento Nueva Lectura Sensor IoT**
3. Ejercicio 2: Ejercicio propuesto
4. Cierre

Event-driven architecture
Ejercicio 1 - Evento Nueva Lectura Sensor IoT

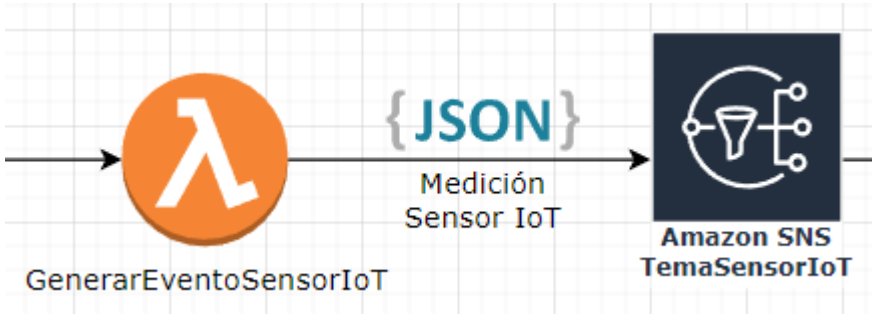
Implemente la siguiente arquitectura para procesar el evento “Nuevo Lectura Sensor IoT”



Event-driven architecture
Ejercicio 1 - Evento Nueva Lectura Sensor IoT

Paso 1: Crear tema SNS “TemaSensorIoT”

Paso 2: Crear lambda “GenerarEventoSensorIoT” con este código fuente (reemplazar amarillo) que simula sensor de CO2 con lectura aleatoria entre 400 y 1000 PPM



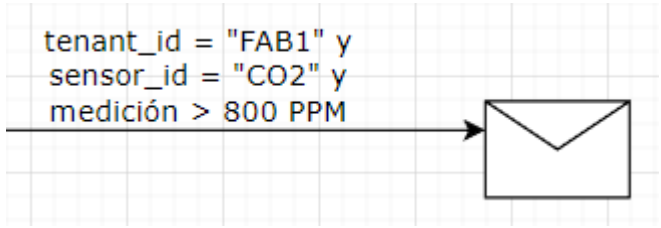

```
import json
import random
from datetime import datetime
import boto3

def lambda_handler(event, context):
    # TODO implement
    tenant_id = "FAB1"
    sensor_id = "CO2"
    now = datetime.now()
    fecha_hora = str(now.date()) + "." + str(now.time())
    medicion = random.randint(400, 1000) # Desde 400 a 1000 PPM (Nivel de CO2)
    unidad_medida = "PPM"

    lectura_sensor = {
        'tenant_id': tenant_id,
        'lectura_id': sensor_id + "." + fecha_hora,
        'lectura_datos': {
            'medicion': medicion,
            'unidad_medida': unidad_medida
        }
    }
    # Publicar en SNS
    sns_client = boto3.client('sns')
    response_sns = sns_client.publish(
        TopicArn = 'arn:aws:sns:us-east-1:447891120606:TemaSensorIoT',
        Subject = 'Nueva Lectura Sensor',
        Message = json.dumps(lectura_sensor),
        MessageAttributes = {
            'tenant_id': {'DataType': 'String', 'StringValue': tenant_id },
            'sensor_id': {'DataType': 'String', 'StringValue': sensor_id },
            'medicion': {'DataType': 'Number', 'StringValue': str(medicion) }
        }
    )
    # TODO implement
    return {
        'statusCode': 200,
        'body': response_sns
    }
```

Event-driven architecture
Ejercicio 1 - Evento Nueva Lectura Sensor IoT

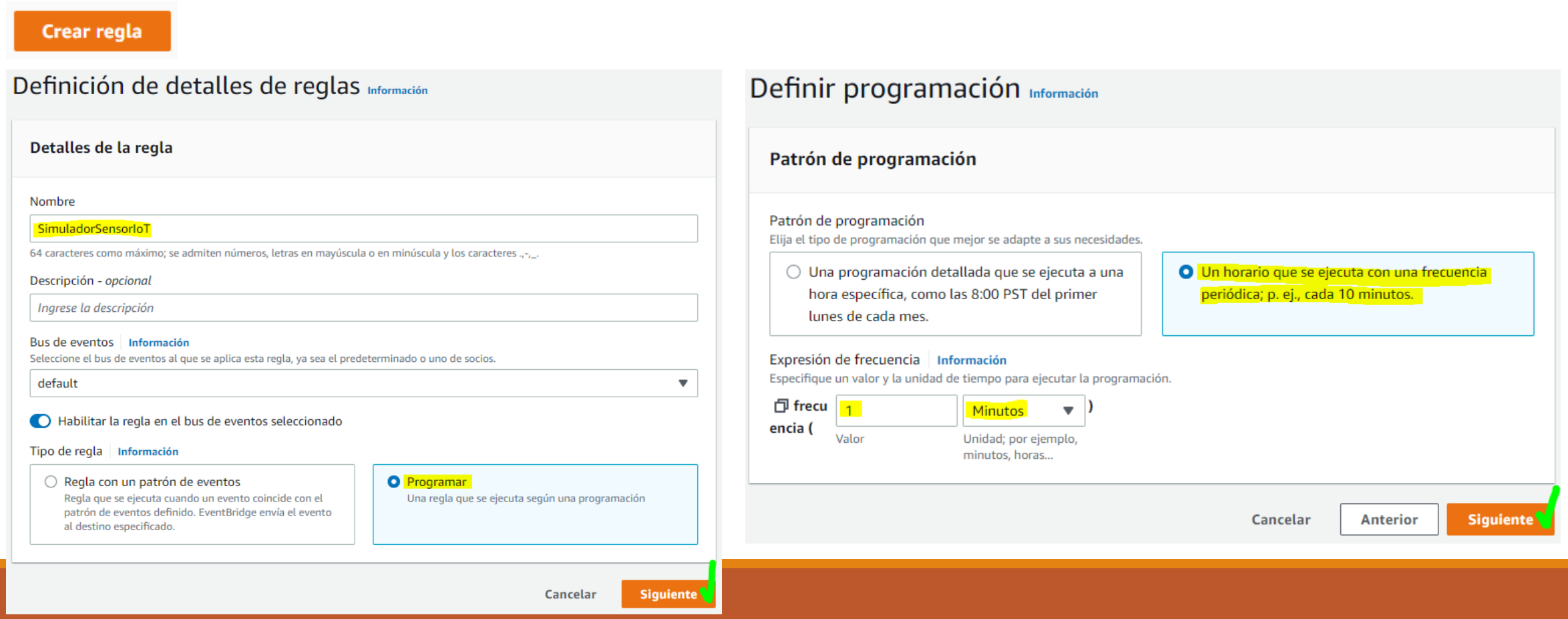
Paso 3: Crear una suscripción de correo electrónico válido al tema SNS “TemaSensorIoT” y colocar la siguiente política de filtro




```
{
 "tenant_id": [
  "FAB1"
 ],
 "sensor_id": [
  "CO2"
 ],
 "medicion": [
  {
   "numeric": [
    ">",
    800
   ]
  }
 ]
}
```

Event-driven architecture
Ejercicio 1 - Evento Nueva Lectura Sensor IoT

Paso 4: Crear regla en EventBridge “SimuladorSensorIoT” que llame al lambda “GenerarEventoSensorIoT” cada 1 minuto



Event-driven architecture
Ejercicio 1 - Evento Nueva Lectura Sensor IoT

Seleccionar destinos

Crear regla

Permisos
Nota: Al usar la consola de EventBridge, EventBridge configurará automáticamente los permisos correspondientes para los destinos seleccionados. Si utiliza CloudFormation, el SDK o la CLI de AWS, debe configurar los permisos adecuados.

Destino 1

Tipos de destino
Seleccione un bus de eventos de EventBridge, un destino de la API de EventBridge (socio de SaaS) u otro servicio de AWS como destino.

- ○ Bus de eventos de EventBridge
- ○ Destino de la API de EventBridge
- ◉ Servicio de AWS

Seleccione un destino | Información
Seleccione los destinos que desee invocar cuando un evento coincida con el patrón de eventos o cuando se active la programación (hay un límite de 5 destinos por regla).

Función Lambda

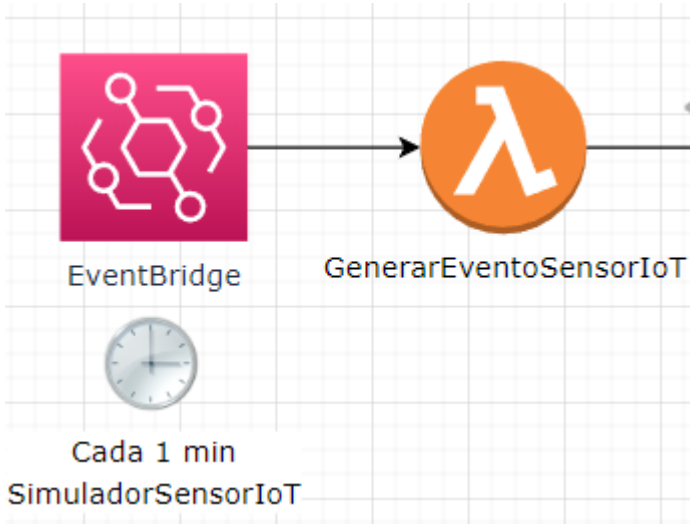
Función

GenerarEventoSensorIoT

▶ Configurar la versión o el alias

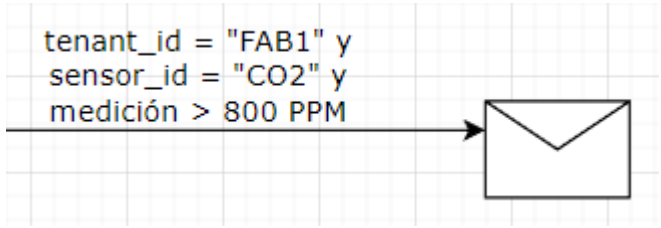
▶ Configuración adicional

Añadir otro destino | Cancelar | Anterior | Siguiente



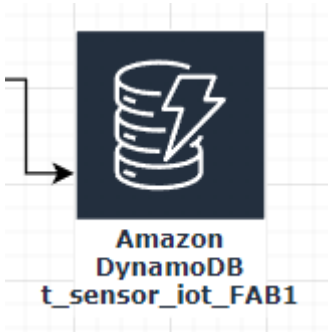
Event-driven architecture
Ejercicio 1 - Evento Nueva Lectura Sensor IoT

Paso 5: Esperar unos 6 minutos y verificar si le ha llegado algún correo notificando este filtro



Event-driven architecture
Ejercicio 1 - Evento Nueva Lectura Sensor IoT

Paso 6: Crear tabla dynamoDB “t_sensor_iot_FAB1”



Nombre ▲	Estado	Clave de partición	Clave de ordenación
t_sensor_iot_FAB1	Creando	tenant_id (S)	lectura_id (S)

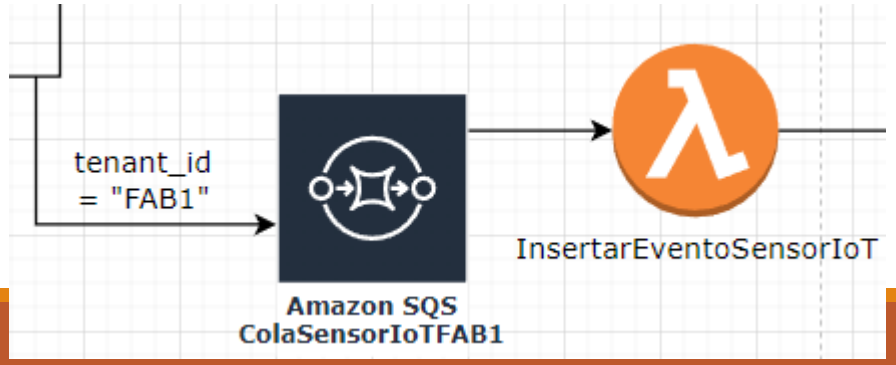
Event-driven architecture
Ejercicio 1 - Evento Nueva Lectura Sensor IoT

Paso 7: Crear lambda “InsertarEventoSensorIoT” con este código fuente

Paso 8: Crear cola SQS “ColaSensorIoTFAB1” que desencadene el lambda “InsertarEventoSensorIoT” y con política de acceso para tema SNS “TemaSensorIoT”

Paso 9: Suscribir la cola SQS “ColaSensorIoTFAB1” al tema SNS “TemaSensorIoT” con política de filtro

```
{
  "tenant_id": [
    "FAB1"
  ]
}
```




```
import json
import boto3

def lambda_handler(event, context):
    # Entrada (json)
    body = json.loads(event['Records'][0]['body'])
    lectura_sensor = json.loads(body['Message'])
    # Proceso
    dynamodb = boto3.resource('dynamodb')
    table = dynamodb.Table('t_sensor_iot_FAB1’)

    response = table.put_item(Item=lectura_sensor)
    # Salida (json)
    return {
        'statusCode': 200,
        'response': response
    }
```

Event-driven architecture
Ejercicio 1 - Evento Nueva Lectura Sensor IoT

Paso 10: Valide que se registren lecturas del sensor de CO2 en la tabla dynamoDB “t_sensor_iot_FAB1”

tenant_id	lectura_id	lectura_datos
FAB1	CO2.2022-10-26.01:45:48.591414	{ "unidad_medida" : { "S" : "PPM" }, "medicion" : { "N" : "807" } }
FAB1	CO2.2022-10-26.01:46:48.060123	{ "unidad_medida" : { "S" : "PPM" }, "medicion" : { "N" : "948" } }
FAB1	CO2.2022-10-26.01:47:48.047735	{ "unidad_medida" : { "S" : "PPM" }, "medicion" : { "N" : "785" } }

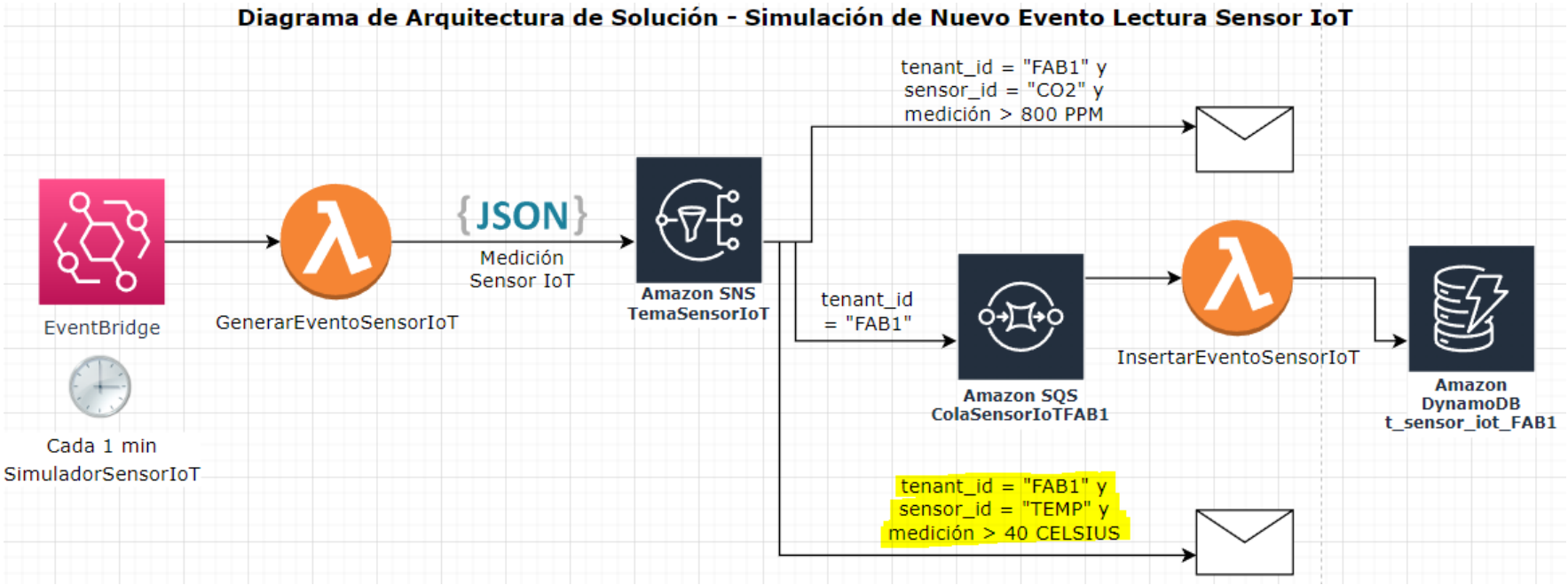
Contenido

Event-driven architecture

1. Objetivo del taller 4
2. Ejercicio 1: Evento Nueva Lectura Sensor IoT
3. **Ejercicio 2: Ejercicio propuesto**
4. Cierre

Event-driven architecture
Ejercicio 2 - Propuesto

- En lambda “GenerarEventoSensorIoT” agregue para tenant_id = “FAB1” un nuevo sensor_id = “TEMP” que genere aleatoriamente lectura entre 10 y 60 °C (grados Celsius) y utilice como unidad_medida = “CELSIUS”
- Agregue una suscripción de otro correo electrónico con el filtro indicado en amarillo
- Verifique que se inserten las lecturas del nuevo sensor en tabla dynamoDB y que llegue el correo cuando se cumpla el filtro
- Muestre evidencia de correo y registros en tabla en padlet indicado por docente



Contenido

Event-driven architecture

1. Objetivo del taller 4
2. Ejercicio 1: Evento Nueva Lectura Sensor IoT
3. Ejercicio 2: Ejercicio propuesto
4. **Cierre**

Cierre:
Event-driven architecture - Qué aprendimos?

- Diseño e implementación de una Arquitectura de Solución basada en eventos con el servicio “SNS - Simple Notification Service” y “SQS - Simple Queue Service”.

Gracias

Elaborado por docente: Geraldo Colchado