EC2 Backup

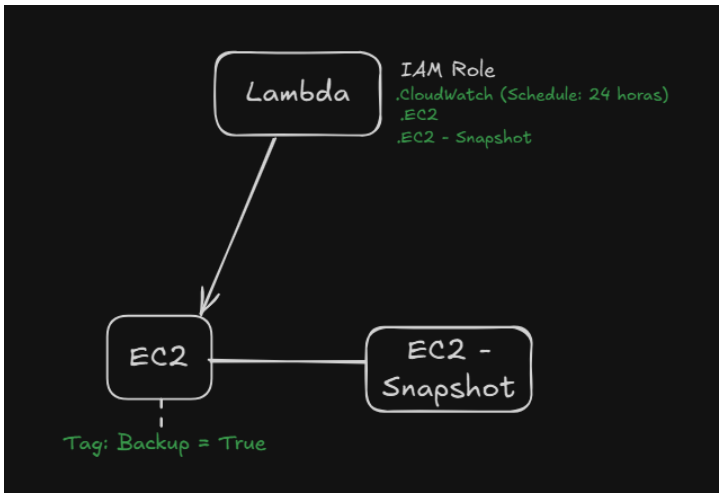
Apresentação Projeto

Nesse projeto, iremos automatizar os **backups** referente as nossas maquinas **EC2**!!

Pois em alguns momentos, precisamos realizar **backups** em nossas maquinas **EC2** para que seja possviel **restaurar-las** quando der algum tipo de problema!!

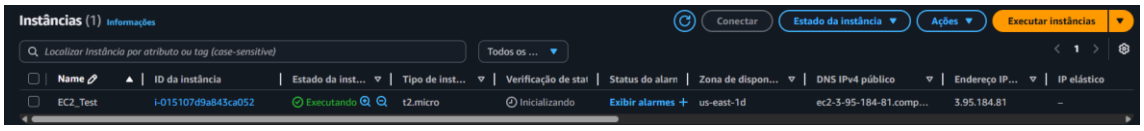
Dito isso, vamos automatizar de forma que a cada **24 horas** seja realziado **backups**, ou conhecido como **snapshot**, em nossas maquinas **EC2**!

Time line Projeto

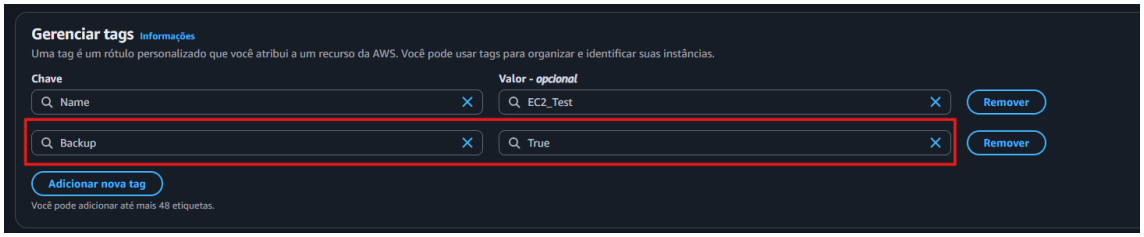


Maquinas EC2

Primeiramente, temos que ter maquinas **EC2** em nosso **ambiente AWS**!! Sabendo disso, foi criado uma maquina **EC2** para teste, como podemos ver!!

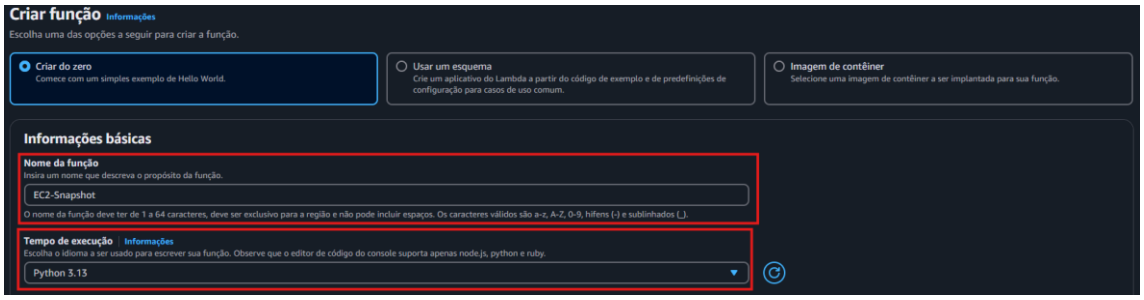


E um detalhe importante, precisamos adicionar uma tag para a nossa maquina **EC2**!! Pois a nossa **automatizacao** so sera feita para essas maquinas **EC2** que tiverem essa **tag**, para que nao seja necessario realizar o **backup** em todas as maquinas, mas sim so nas mais importantes!!



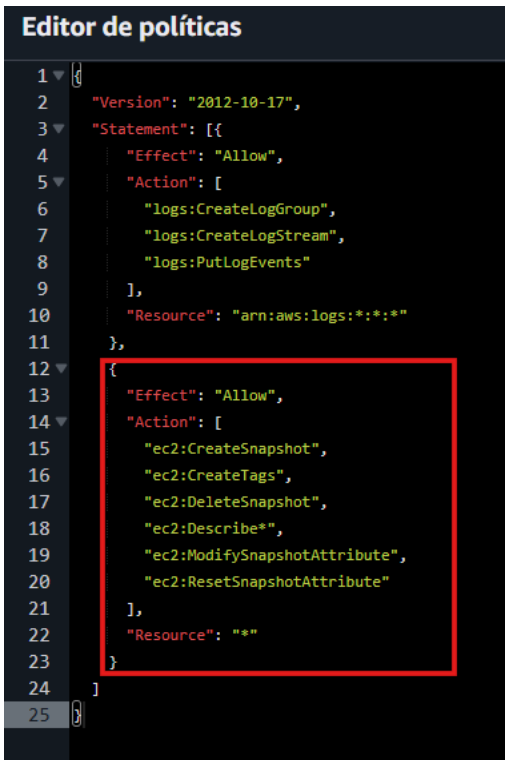
Criando funcao lambda

Vamos agora criar a nossa **função lambda**!! Logo segue a criação de forma simples e direta, lembrando que sera uma **funcao lambda** utilizando **script Python**!!



Após a criação da **função lambda**, como já sabemos, o **IAM Role** padrão dará permissão apenas para o serviço do **CloudWatch**, onde sera necessario para a nossa automatizacao, pois utilziaremos o **job** de **Schedule**, que sera configurado nele para que a nossa **funcao lambda** seja executada a cada **24 horas**!!

Alem do **CloudWatch**, precisaremos editar a política dessa role, adicionando as permissões referente ao serviço do **EC2** e **EC2 Snapshot** também!



Criacao script python

Enfim vamos criar o **script python** da nossa **funcao lambda**!! E Primeiramente vamos importar as biliotecas necessarias, que serao o **boto3** e **datetime**.

```
lambda_function.py
1   from datetime import datetime
2   import boto3
```

Após a importacao, vamos criar a funcao principal lambda_handler(), e de cara já vamos criar o objeto **ec2** pela funcao **boto3.client('ec2')**!

```
5   def lambda_handler(event, context):
6       ec2_client = boto3.client('ec2')
```

Em seguida vamos realizar um **List Comprehension**, para que seja retornado a lista de todas as regioes existentes da **AWS** referente ao servico do **EC2** que estao **habilitadas** na sua **conta**, atraves da funcao **describe_regions()**, e consequentemente iterar nessa lista atraves de um **for loop**.

```
regions = [region['RegionName']] for region in ec2_client.describe_regions()['Regions']
for region in regions:
```

Agora buscar todas as instancias das maquinas **EC2**!!

Dito isso, vamos buscar todas as maquians **EC2** atraves da funcao **resource()** definindo o parametro **region_name**. E depois realizaremos um filtro para trabalharmos apenas com as instancias com a **tag backup = True**.

```
for region in regions:
    print(f'Instances in EC2 Region {region}:')

    ec2 = boto3.resource('ec2', region_name=region)

    instances = ec2.instances.filter(
        Filters=[
            {'Name': 'Backup', 'Values': ['True']}
        ]
    )
```

Agora que temos a lista das instancias da regiao especifica, vamos criar um **for loop** atraves da funcao **all()** do objeto **instances**, para que possamos iterar por cada **instancia**, e de fato realizar o **snapshot** da **instancia especifica**!!

Mas antes de fato realizar o **snapshot** de cada **instancia** da regiao, vamos realizar um print informando a **instancia** e todos os **volume** que serao realizados no **snapshot**!!

```python
timestamp = datetime.utcnow().replace(microsecond=0).isoformat()

for i in instances.all():
    for v in i.volumes.all():
        desc_snapshoot = 'Backup of instance {i.id}, volume {v.id}, created {timestamp}'
        print(desc_snapshoot)
```

Veja que alem do **for loop** das **instancias**, foi realziado tambem um **for loop** de cada **volume** daquela instancia especifica!! Pois o **backup** do **snapshot** eh feito por cada **volume** de uma **instancia do EC2**!!

Dito isso, chegou a hora em realziar o snapshot do volume si!! E para isso utilizaremos a funcao create_snapshot() do objeto **volume (v)** que estamos iterando no **for loop**!

```python
for v in i.volumes.all():
    desc_snapshoot = 'Backup of instance {i.id}, volume {v.id}, created {timestamp}'
    print(desc_snapshoot)

    snapshot = v.create_snapshot(Description=desc_snapshoot)
    print("Created snapshot:", snapshot.id)
```

E pronto!! Nosso **script python** para automatizar o **snapshoot** de todas as **intancias EC2** em todas as **regioes** da nossa conta **AWS** esta feito!!

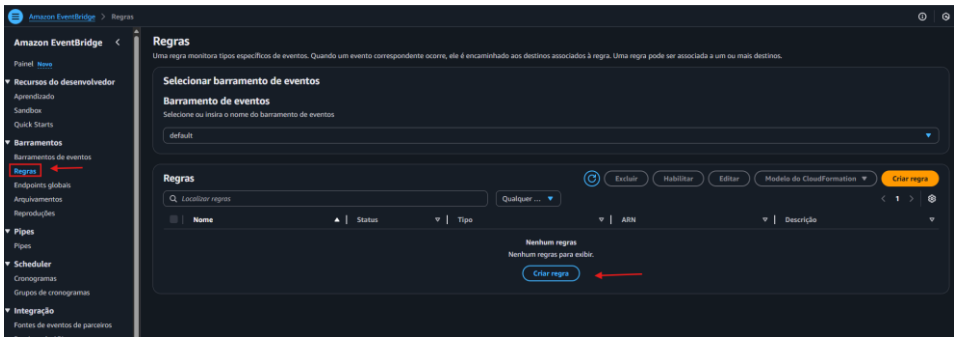
lambda_function.py

```python
from datetime import datetime
import boto3


def lambda_handler(event, context):
    ec2_client = boto3.client('ec2')
    regions = [region['RegionName']] for region in ec2_client.describe_regions()['Regions']

    for region in regions:
        print(f'Instances in EC2 Region {region}:')

        ec2 = boto3.resource('ec2', region_name=region)

        instances = ec2.instances.filter(
            Filters=[
                {'Name': 'Backup', 'Values': ['True']}
            ]
        )

        timestamp = datetime.utcnow().replace(microsecond=0).isoformat()

        for i in instances.all():
            for v in i.volumes.all():
                desc_snapshoot = 'Backup of instance {i.id}, volume {v.id}, created {timestamp}'
                print(desc_snapshoot)

                snapshot = v.create_snapshot(Description=desc_snapshoot)
                print("Created snapshot:", snapshot.id)
```

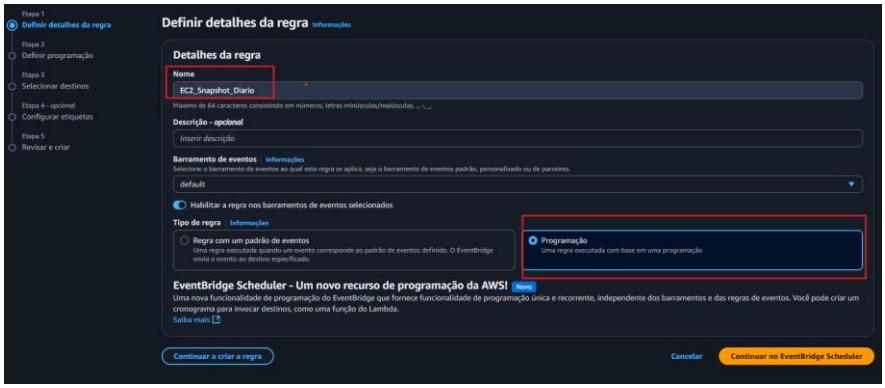
Configurando Schedule pelo Amazon EventBridge

E para finalizar, vamos agora configurar o **schedule** (agendamento) da execucao da nossa **funcao lambda**!! E para isso precisamor ir no servico do **EventBridge**!!

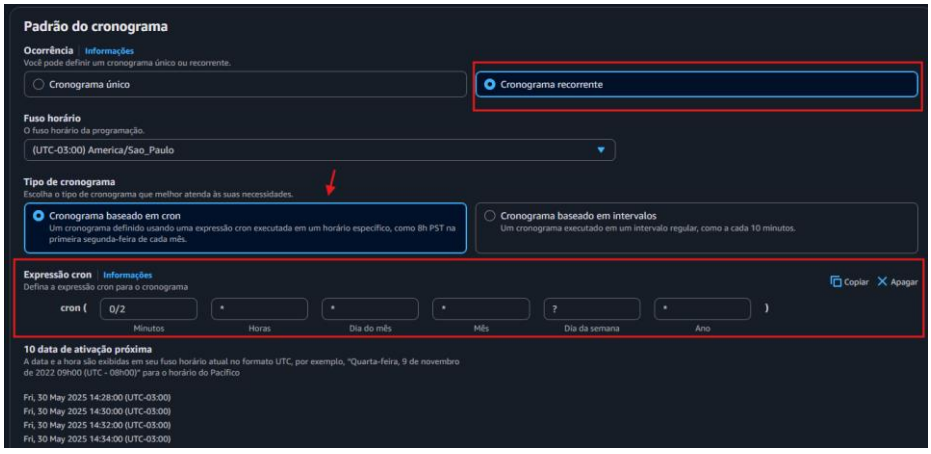
Ao abrir o servico, vamos no menu lateral em barramentos e solicitar para criar uma regra!! Sera nele que iremos criar nossa regra de **schedule**(agendamento) para executar a nossa fuuncao lambda!!



E para criar uma **regra**, devemos seguir uma serie de passos, e na **primeira etapa** basicamente definimos o **nome da regra** e o **tipo da regra**, que nesse caso o tipo sera **programacao**, já que devemos debir um **agendamento** para ser executado essa regra!!

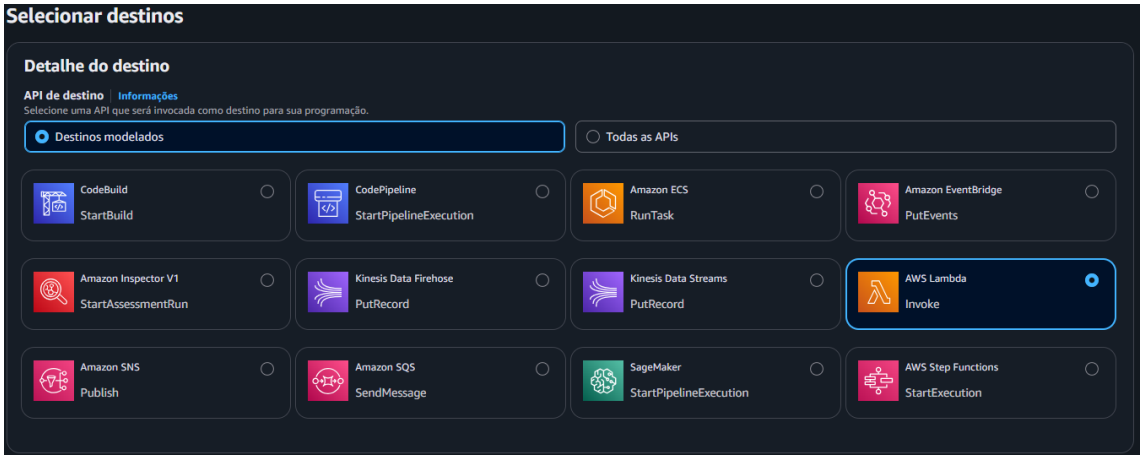


Já na proxima etapa basicamete sera onde definimos nossa expressao de agendamento **schedule**, conhecido como **cron**!!



Veja que para ajudar no nosso teste, foi utilziado uma expressao **cron**, onde sera executado a cada **2 minutos**!! Porem a versao final correta sria a cada **24 horas**!!

Na proxima etapa enfim definimos qual **tipo de servico** que sera executado nesse **schedule**!! Como se trata de uma **funcao lambda**, sera entao selecionado a **funcao lambda** que **criamos anteriormente** para esse **vinculo** do **schedule**!!



E pronto!! Nossa **configuracao do schedule** esta realizada!!

Criacao dos snapshots

E para finalizar, logo em seguida veremos que a execucao da **funcao lambda** sera feita pelo agendamento que realizamos do **schedule**!! Tanto que se formos no servico do **EC2**, e irmos na **area de snapshots** da nossas intancias, veremos que temo um novo **snaphot** realizado pelo nosso **script python** do **lambda**!!