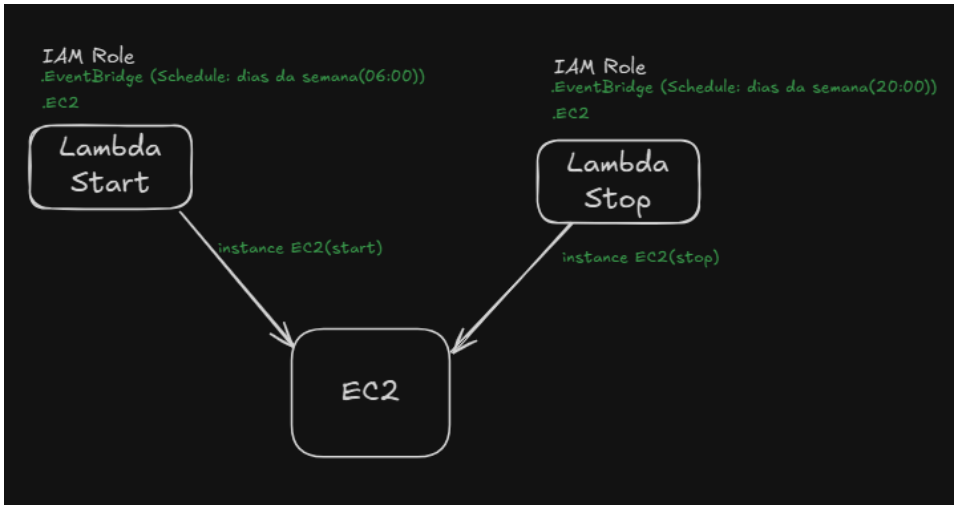
# Start/Stop Maquinas EC2

## Apresentação Projeto

Nesse projeto, iremos automatizar o **desligamento** e a **ligacao** de **maquinas EC2** em certos horarios nos dias da semana, visando a economia de recursos e financeiros!!

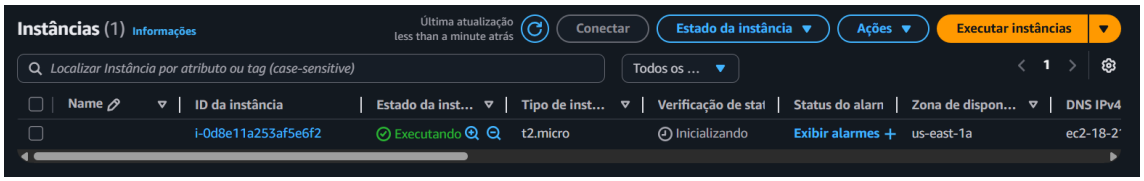
Pois pode ser que o consumo dessas maquinas **EC2** so seja necessario durante o horario comercial e durante a semana, logo precisaremos de **2 scripts**!! Um que realiza esse desligamento e outro que realiza a reativacao das maquinas!!

## Time-Line Projeto



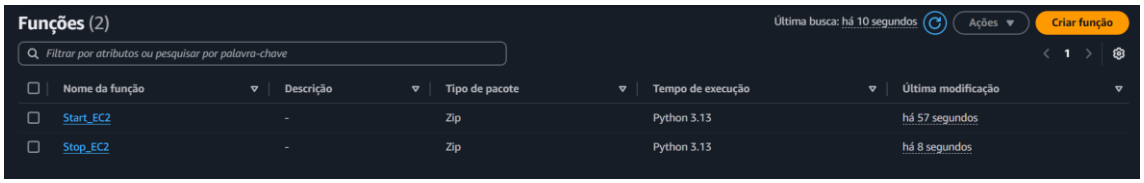

## Criacao maquinas EC2

Primeiramente precisamos criar maquinas **EC2** para que os nossas **funcoes lambda** possam ligar e desligar as maquinas **EC2** dependendo do horario esperado!




## Criando funcoes lambda

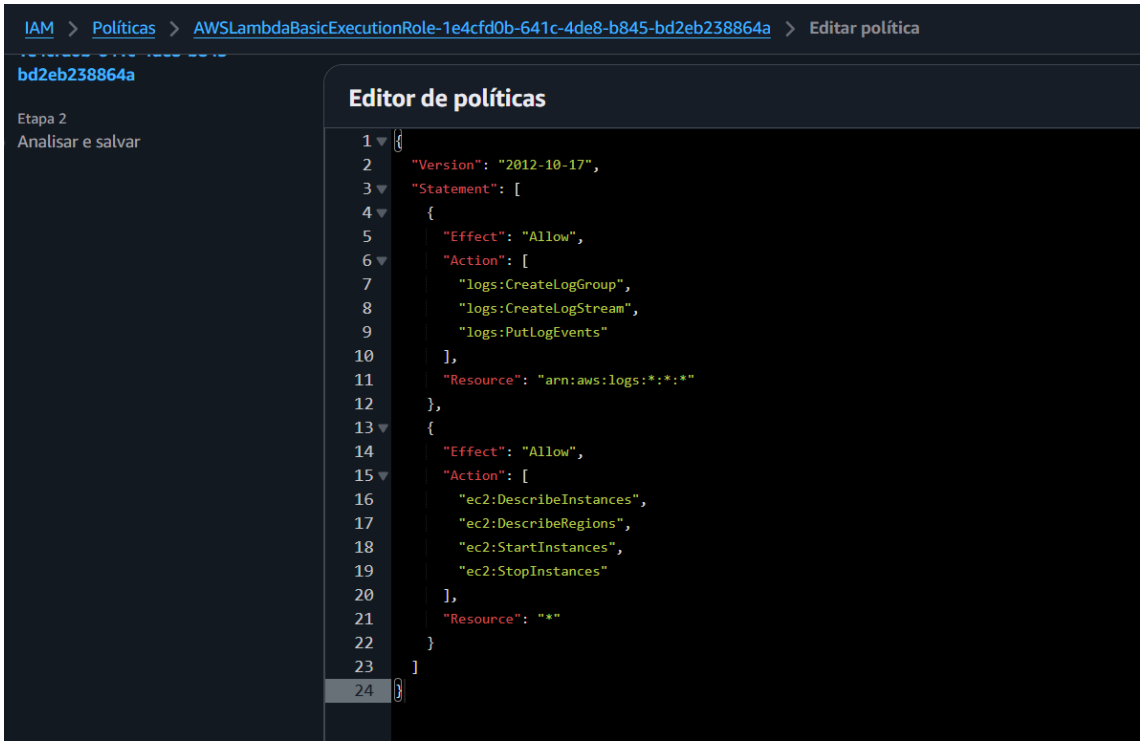
Com as maquinas **EC2** criadas, vamos criar agora as **funcoes lambda**, para podermos automatizar a o ligamento e o desligamento das maquinas **EC2**!!



## Permissao funcoes lambda

Agora precisamos ajustar as regras de permissoes do **IAM** de nossas **funcoes lambda**!!

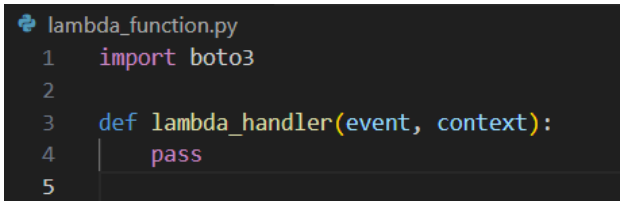
Pois como ja sabemos por padrao a unica permissao que as **funcoes lambda** possui eh referente o servico do **cloudwatch**, e teremos que adicionar as permissoes referente ao servico de **EC2**!! Como podemos ver a seguir:




## Script(Python) - funcao lambda(Stop_EC2)

Vamos agora criar os nossos **scripts python** de nossas **funcoes lambda**, e para comecar vamos criar o **script** referente a **funcao lambda(Stop_EC2)**.

E para iniciar vamos logo importar a biblioteca do **boto3**, que como ja sabemos se trata da blioteca(python) **oficional** da **aws** para integrar diretamernte com os seus servicos!! E claro, a criacao da funcao **lambda_handler(event, context)**!!



Em seguinda, vamos instanciar o objeto **ec2_clien**t alem de listar todas as regioes possiveis de nossa conta aws, atraves pelo metodo **ec2_client.describe_regions()['Regions']**.

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

Veja que foi feito um **List Comprehension**, para que seja retornado uma nova lista alternativa, referente as **regioes** em si, que podemos ter **instancias EC2** criadas na nossa conta **aws**!

Feito isso, vamos criar um **for loop**, para que seja possivel iterar para cada regiao retornado no nosso **List Comprehension**!

```
for region in regions:
    pass
```

E como primeira etapa do nosso **for loop**, vamos utilizar funcao **boto3.resource()** da library do **boto3**, onde buscaremos as nossas **intancias EC2**!! E veja, que definimos a regiao especifica tambem que queremos buscar pelo parametro **region_name**!!

```
for region in regions:
    print("Region:", region)

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

Com as **instancias EC2** retornadas, precisamos agora filtrar apenas as **instacias** que estao com o status **RUNNING**, para que possamos realizar o **STOP**, e para isso utilziamos a prop **'instance-state-name'**!!

```
instances = ec2.instances.filter(
    Filters=[{'Name': 'instance-state-name', 'Values': ['running']}]
)
```

E para finalizar o script, vamos realizar um novo for loop para que de fato realizar o stop dessas instancias EC2

```
for instance in instances:
    instance.stop()
    print('Stopped instance: ', instance.id)
```

Veja que foi stopado atraves do metodo **stop()** pelo objeto **instance**!

## Script(Python) - funcao lambda(Start_EC2)

Vamos agora criar o nosso **scripts python** de nossa **funcao lambda(Start_EC2)** e veja que o script eh praticamente o mesmo!! Com apenas essas alteracoes:

- **Filtro das intancias EC2**: agora filtramos a prop **'instance-state-name'** pelo valor **'stopped'**.
- **Metodo que inicializar a instancia EC2**: agora como a **instancia EC2** esta stopada, precisamos inicializar atraves do metodo **start()** pelo objeto **instance**.

lambda_function.py

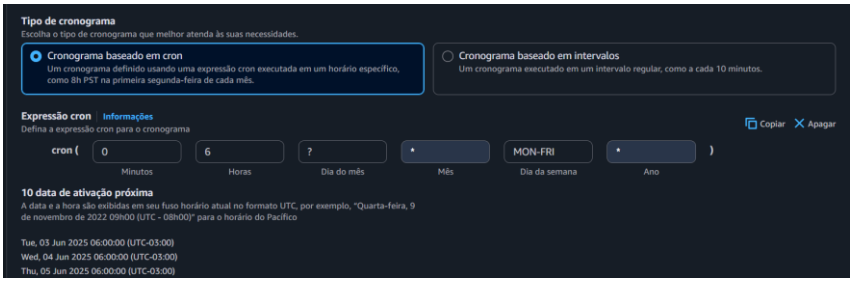
```python
import boto3

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("Region:", region)

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

        instances = ec2.instances.filter(
            Filters=[{'Name': 'instance-state-name', 'Values': ['stopped']}]
        )

        for instance in instances:
            instance.start()
            print('Started instance: ', instance.id)
```

## Configurando Schedule pelo Amazon EventBridge

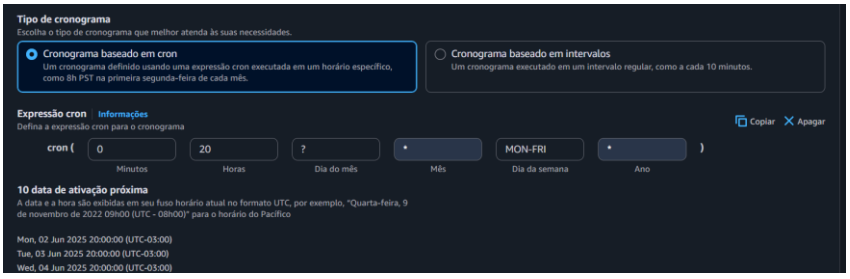
E para finalizar, vamos agora configurar o **schedule(agendamento)** da execucao da nossa **funcao lambda**!! E para isso precisamor ir no servico do **EventBridge**!!

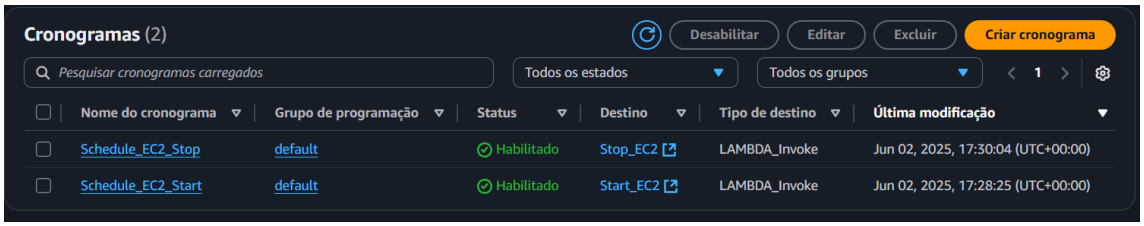
E para a criacao de nossas regras de agendamento, vamos utilziar **expressao cron**!! E como foi citado na **time-line do projeto**, para a **funcao lambda(Start_EC2)** sera feita todos os dias da semana as **06:00:00**! Ja para **funcao lambda(Stop_EC2)** sera feita todos os dias da semana as 20:00:00!

- **Schedule funcao lambda(Start_EC2)**




- **Schedule funcao lambda(Stop_EC2)**






E pronto, com isso criamos todos os **schedules** necessarios para rodar as nossas **funcoes lambdas** referente ao horario correto!