ICT & Infra S3 Automation & Orchestration, week 6

Class:	I3-CB01
Student number:	4642295, 4216709, 4961854
Student name:	Ryan Smith, Edris Rahimi, Heiko Morales

Introduction

Running multiple EC2 instances can be expensive. By using a Lambda function and Amazon EventBridge you can shut down EC2 instances used for development environment.

This is **group** assignment.

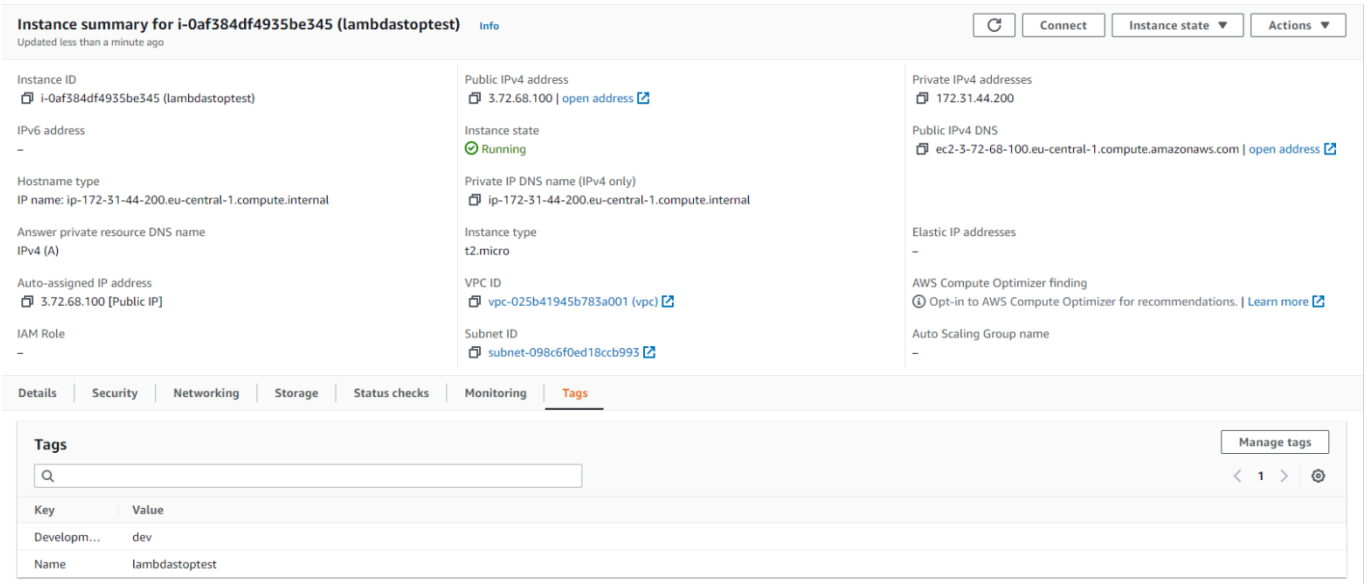
Assignment 1. Automatically shut down EC2 instances

Difficulty: ★★★★☆

To save costs, some EC2 instances can be shut down and started up on a schedule. Follow the tutorial at AWS to implement a Lambda function to control an EC2 instance state every night. Use cron expressions to specify a schedule. Extend the implementation with the following rules:

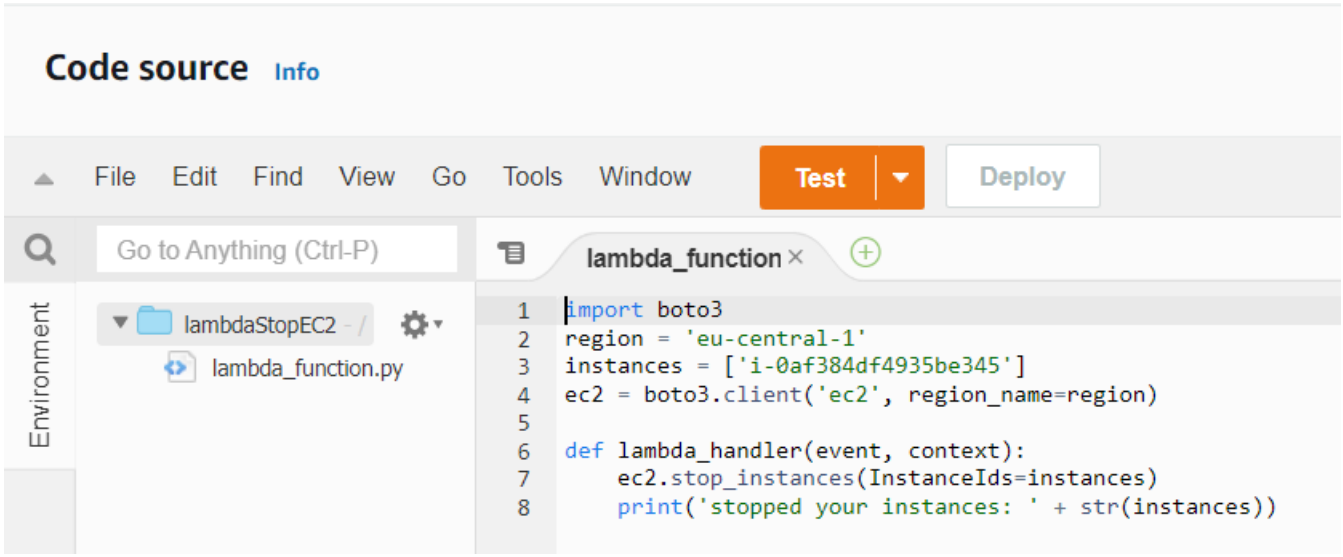
- Every EC2 instance must have an additional tag that identifies importance of an instance. A possible tag can be[1]: *Development*, *Testing*, *Staging*, *Production*. Add necessary tag that identifies the purpose of an EC2 instance.
- An EC2 instance with *Development* or *Testing* tag, must be:
  - Shut down daily at 23:00.
  - It must not be started automatically.

I began by creating an EC2 instance and tagging it accordingly

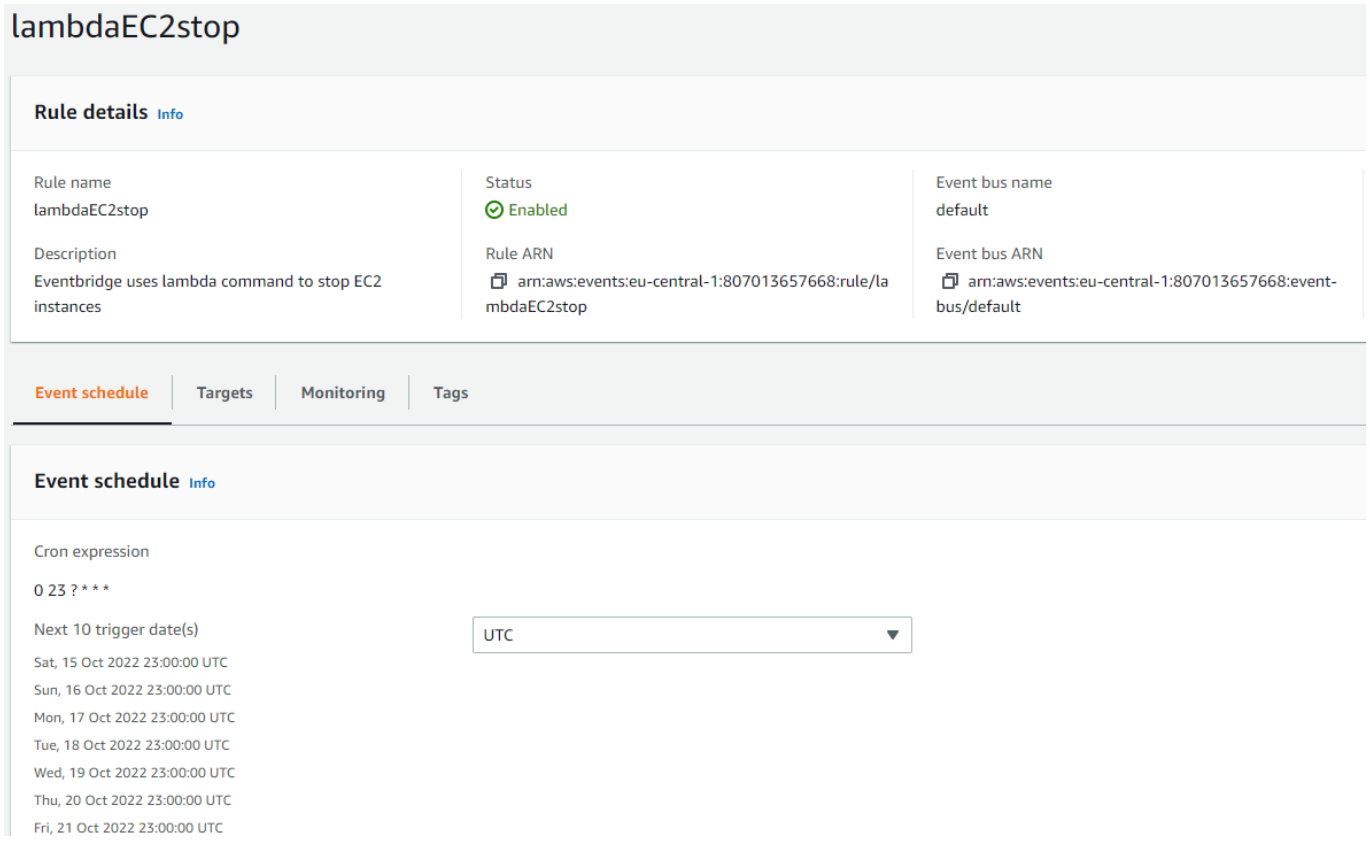


I then created a lambda function that stopped specified EC2 instances

[1] *Differences Between Dev, Staging, Preprod...: https://www.flagship.io/test-environment/*

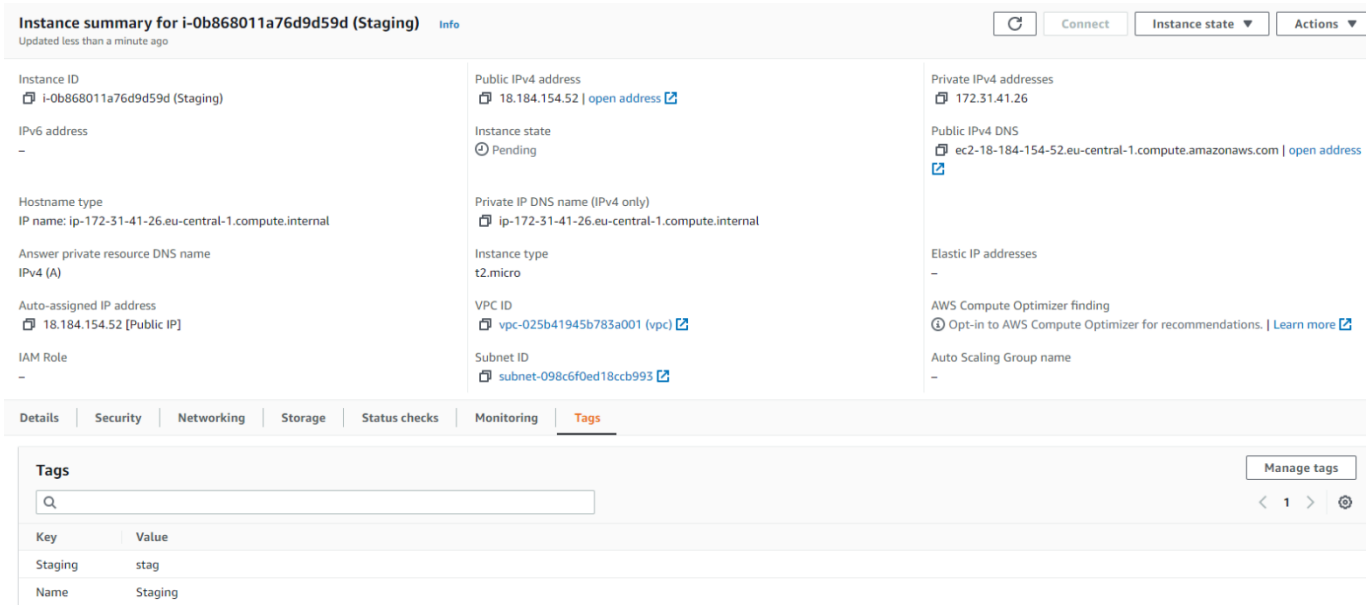


After that I created a rule in eventbridge that would use the lambda function everyday at a certain time. I specified this time using a CRON expression

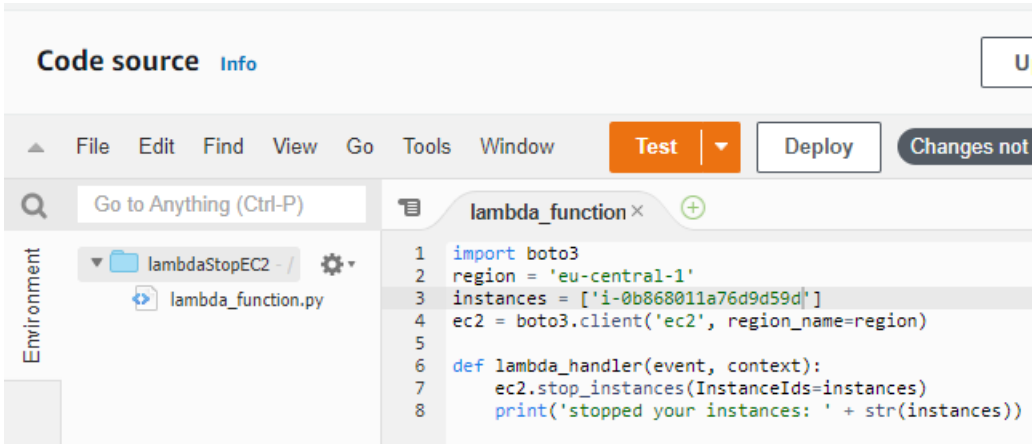


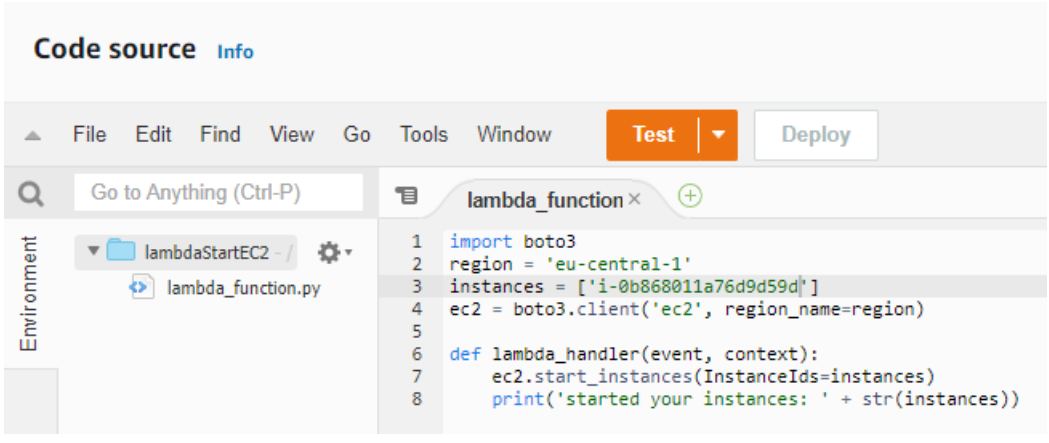
- An EC2 instance with *Staging* tag, must be:
  - Shut down daily at 23:00.
  - It should be started automatically at 9:00 on working days.
  - On a weekend it must not be started automatically.

I started this task the same as the last, by creating an EC2 instance

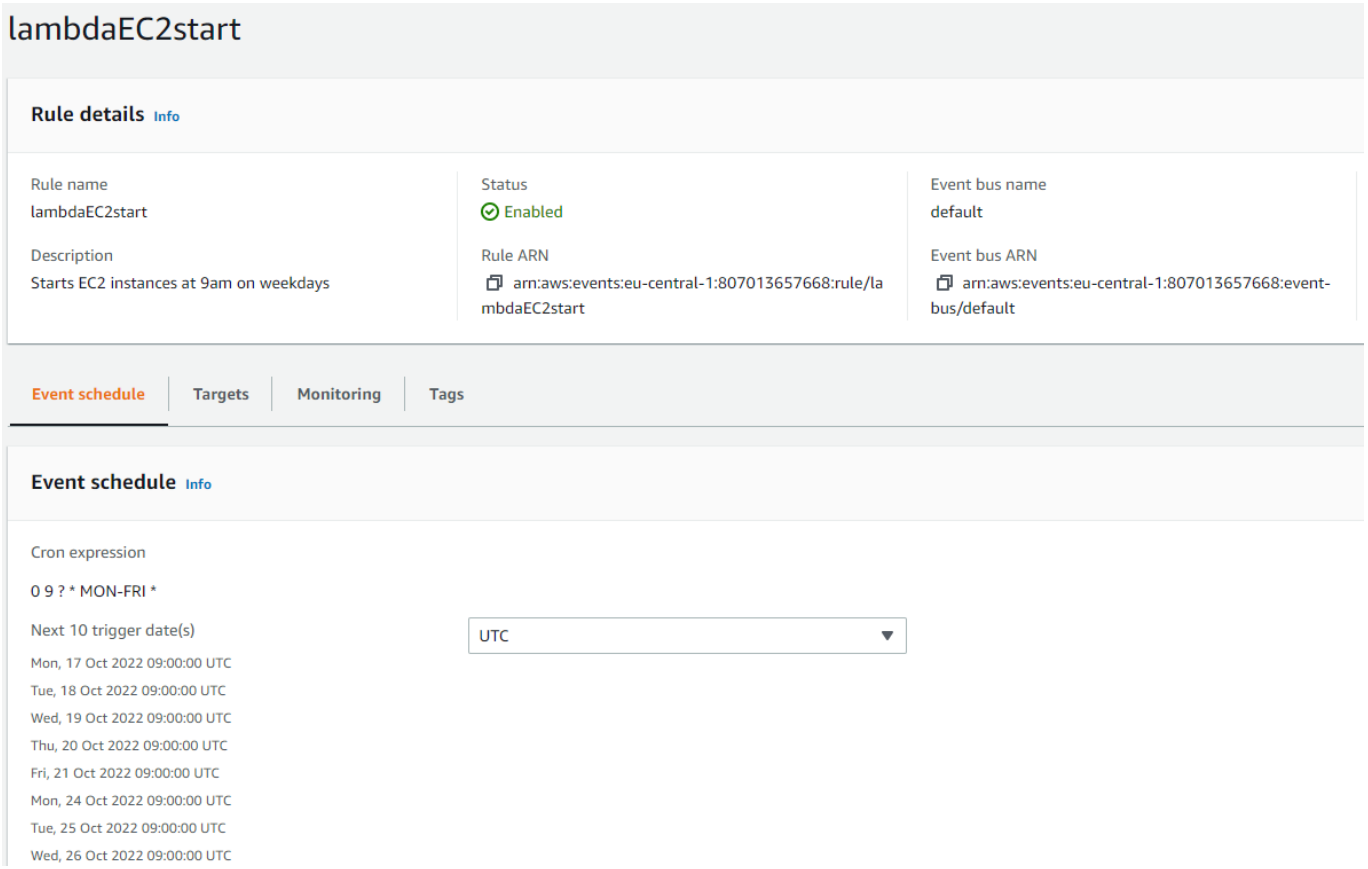


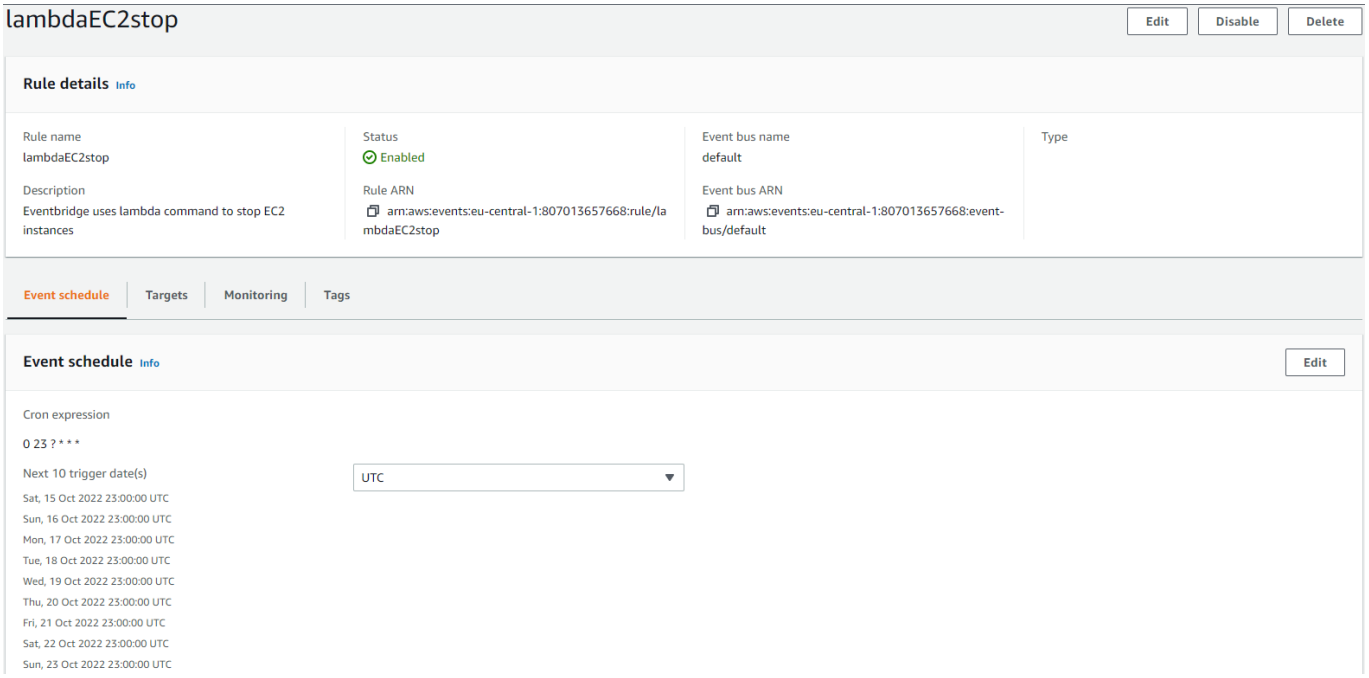
For this task I created another lambda function that started EC2 functions. I altered my previous stop function with the new instance id





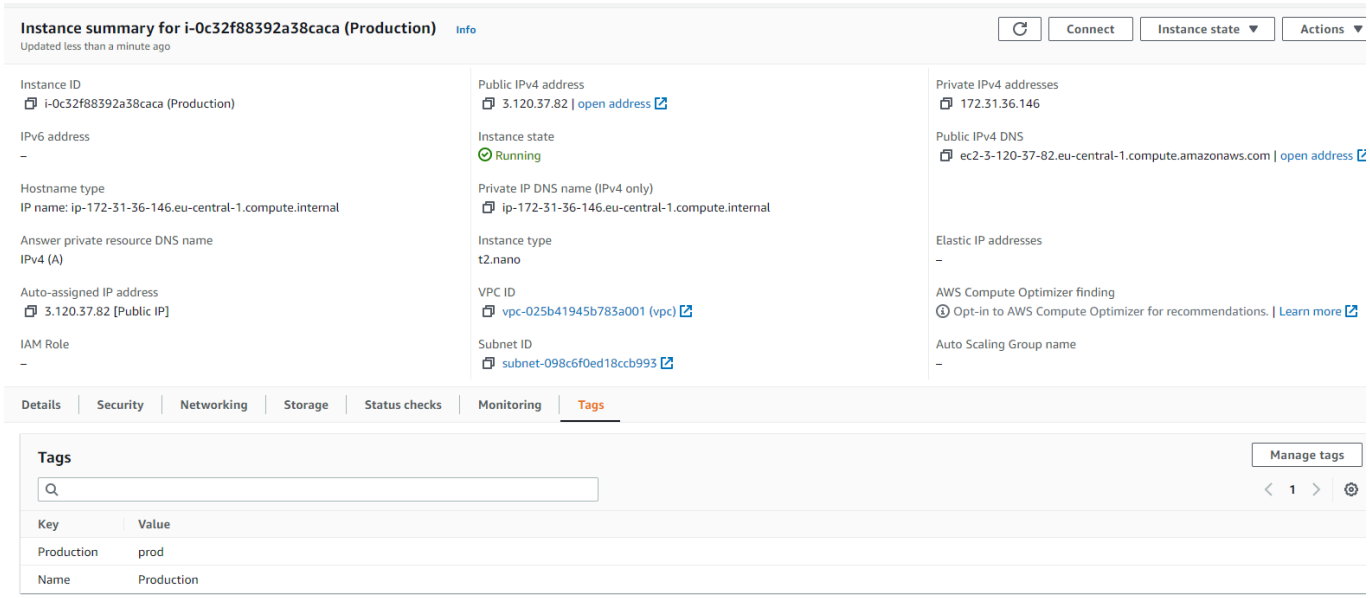
I then created a new eventbridge rule for the new lambda function and set the time accordingly. I altered thye previous rule to trigger at a different time

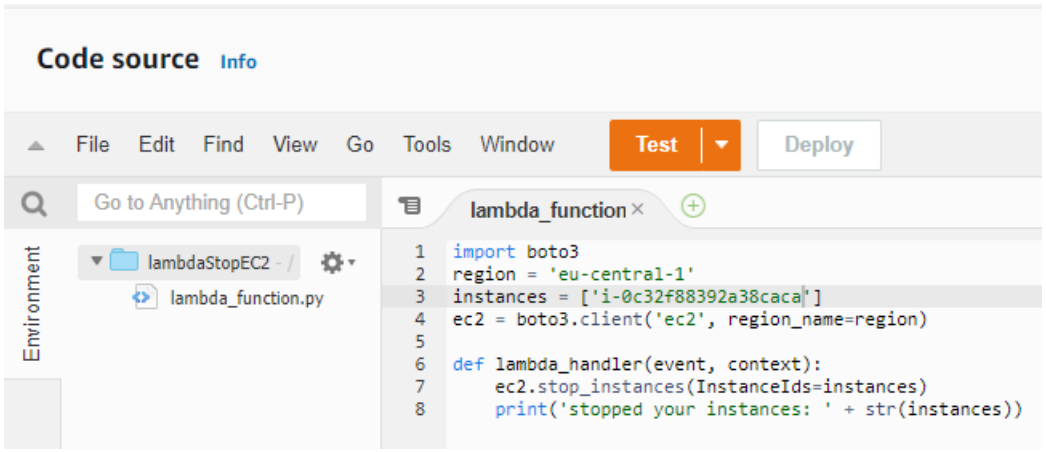


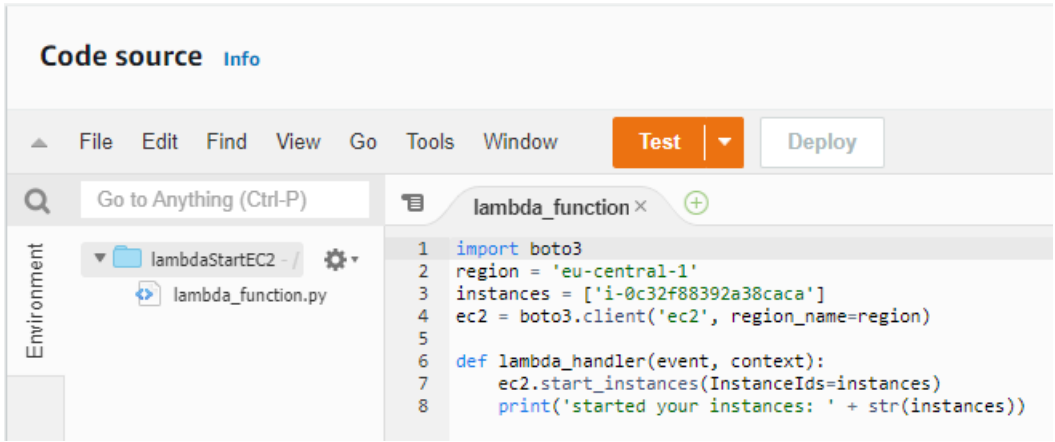


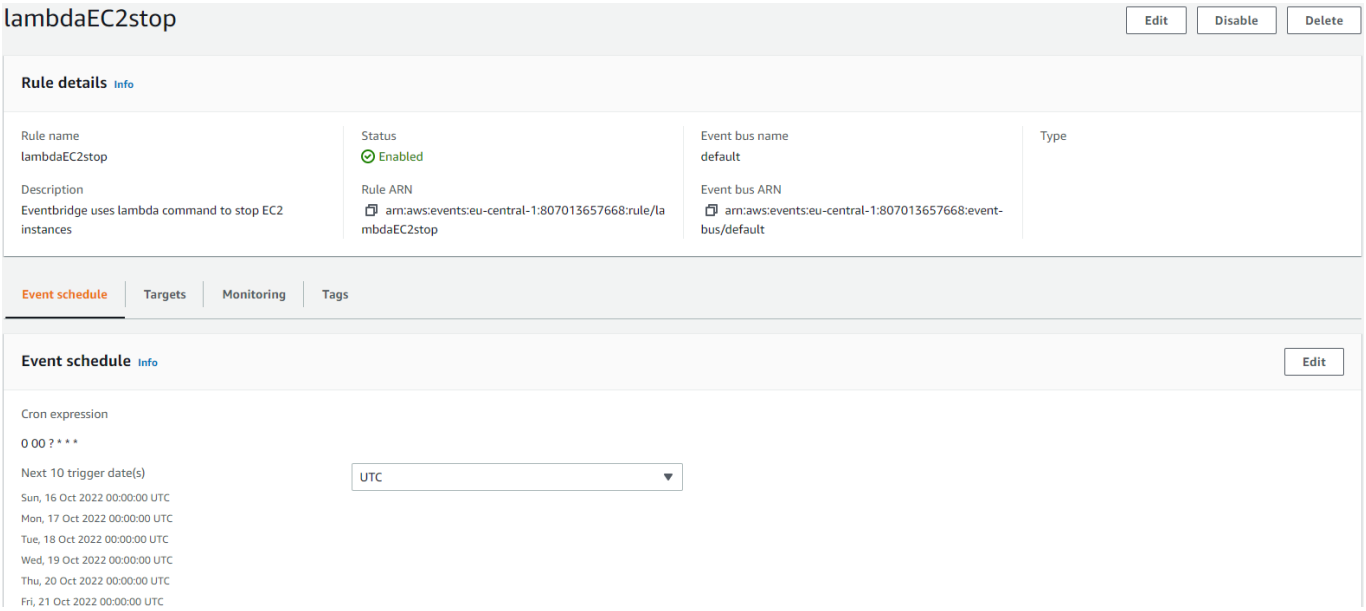
- An EC2 instance with *Production* tag and type of *t2.small* or smaller, must be:
    - Shut down daily at 24:00.
    - It should be started automatically at 8:00 daily.

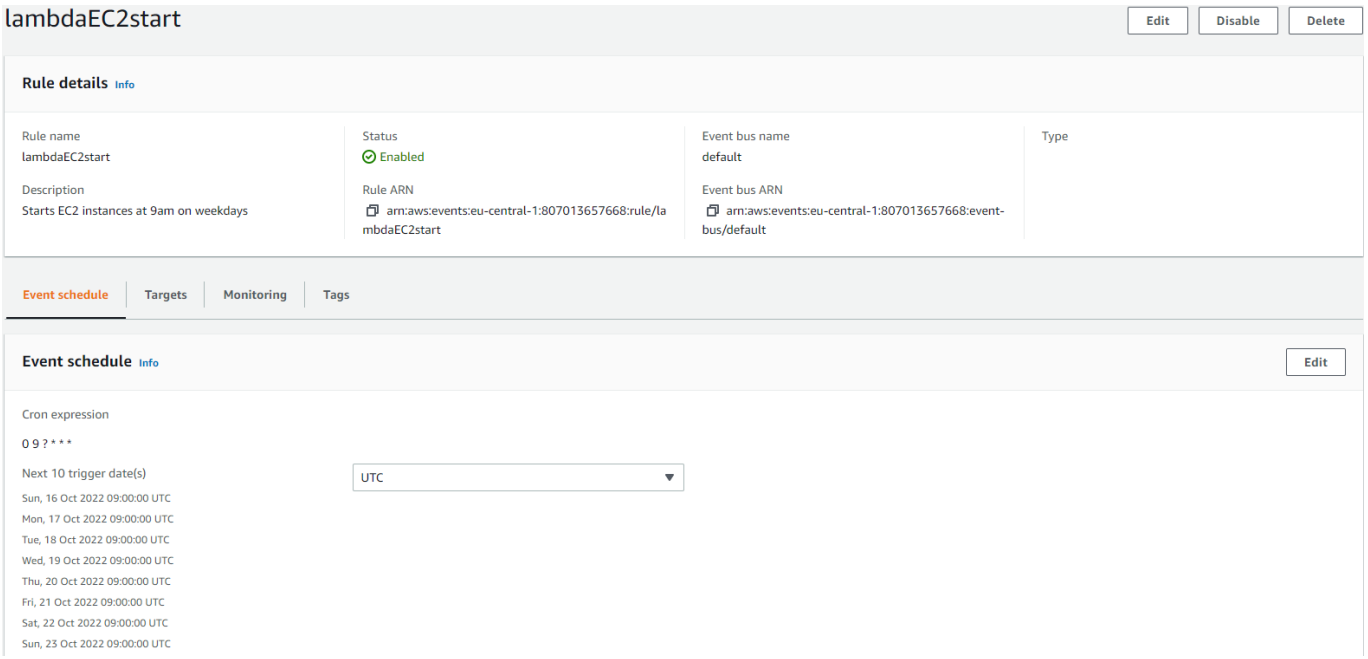
For this task I did the same steps as the last task, only with different times and tags. I also had to choose a different instance size, one that was smaller than I was used to


Instance summary for i-0c32f88392a38caca (Production) Info
Updated less than a minute ago
Connect
Instance state
Actions
Instance ID
i-0c32f88392a38caca (Production)
Public IPv4 address
3.120.37.82 | open address
Private IPv4 addresses
172.31.36.146
IPv6 address
–
Instance state
Running
Public IPv4 DNS
ec2-3-120-37-82.eu-central-1.compute.amazonaws.com | open address
Hostname type
IP name: ip-172-31-36-146.eu-central-1.compute.internal
Private IP DNS name (IPv4 only)
ip-172-31-36-146.eu-central-1.compute.internal
Answer private resource DNS name
IPv4 (A)
Instance type
t2.nano
Elastic IP addresses
–
Auto-assigned IP address
3.120.37.82 [Public IP]
VPC ID
vpc-025b41945b783a001 (vpc)
AWS Compute Optimizer finding
Opt-in to AWS Compute Optimizer for recommendations. | Learn more
IAM Role
–
Subnet ID
subnet-098c6f0ed18ccb993
Auto Scaling Group name
–
Details
Security
Networking
Storage
Status checks
Monitoring
Tags
Tags
Manage tags
Key
Value
Production
prod
Name
Production


Code source Info
File
Edit
Find
View
Go
Tools
Window
Test
Deploy
Go to Anything (Ctrl-P)
lambda_function
Environment
lambdaStopEC2
lambda_function.py
import boto3
region = 'eu-central-1'
instances = ['i-0c32f88392a38caca']
ec2 = boto3.client('ec2', region_name=region)
def lambda_handler(event, context):
ec2.stop_instances(InstanceIds=instances)
print('stopped your instances: ' + str(instances))

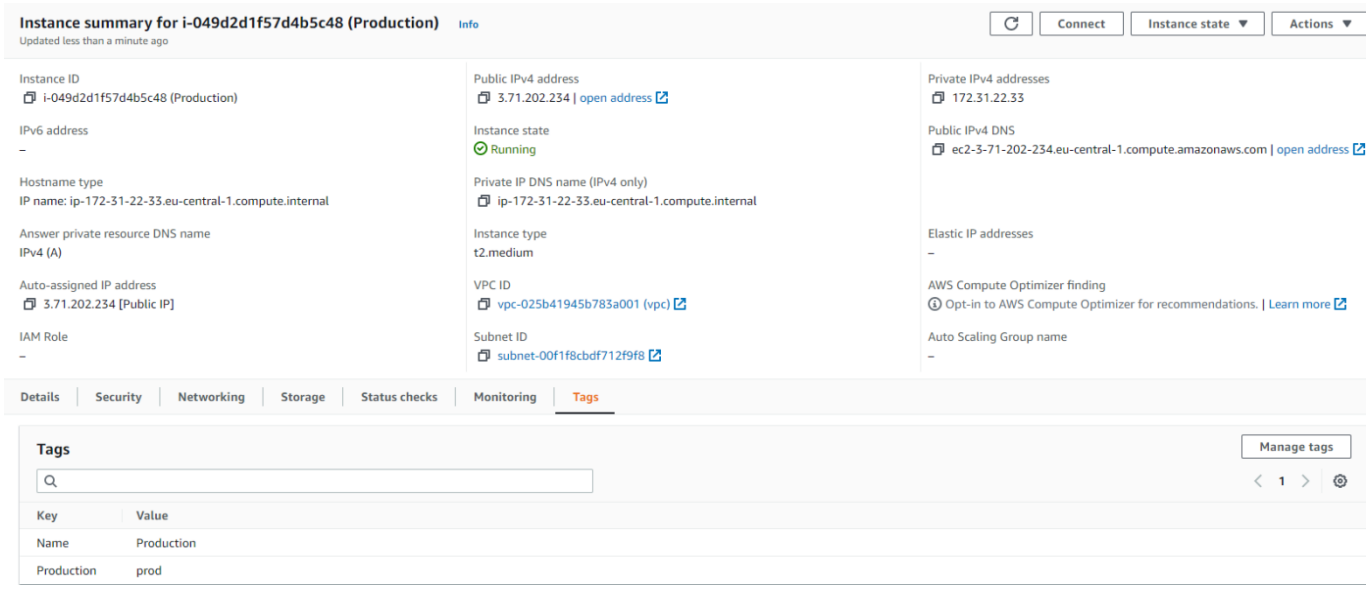

Code source Info
File
Edit
Find
View
Go
Tools
Window
Test
Deploy
Go to Anything (Ctrl-P)
lambda_function
Environment
lambdaStartEC2
lambda_function.py
import boto3
region = 'eu-central-1'
instances = ['i-0c32f88392a38caca']
ec2 = boto3.client('ec2', region_name=region)
def lambda_handler(event, context):
ec2.start_instances(InstanceIds=instances)
print('started your instances: ' + str(instances))

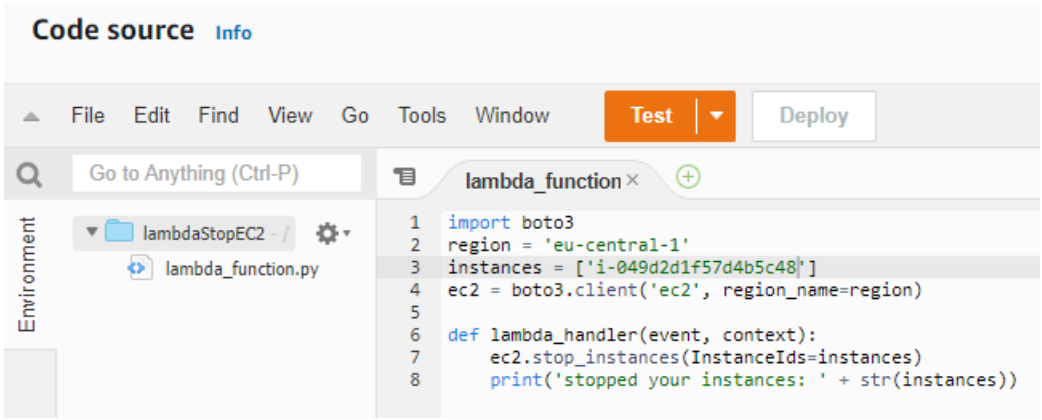


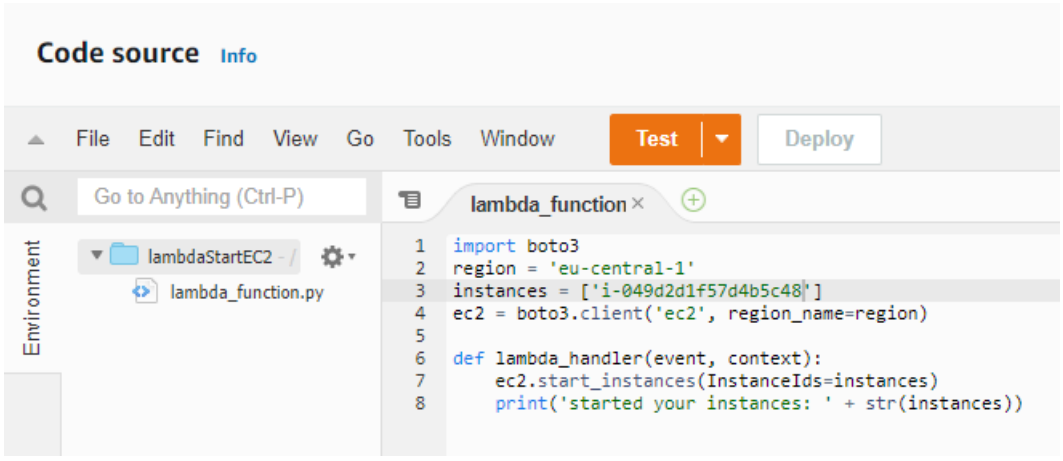


- An EC2 instance with *Production* tag and type of *t2.medium* or bigger, must be:
    - Shut down daily at 24:00.
    - It should be started automatically at 8:00 on working days.
    - On a weekend it must not be started automatically.

For this task I did the same steps as the last task, only with different times and tags. I also had to choose a different instance size, one that was larger than I was used to


Instance summary for i-049d2d1f57d4b5c48 (Production) Info
Updated less than a minute ago
Connect
Instance state
Actions
Instance ID
i-049d2d1f57d4b5c48 (Production)
Public IPv4 address
3.71.202.234 | open address
Private IPv4 addresses
172.31.22.33
IPv6 address
–
Instance state
Running
Public IPv4 DNS
ec2-3-71-202-234.eu-central-1.compute.amazonaws.com | open address
Hostname type
IP name: ip-172-31-22-33.eu-central-1.compute.internal
Private IP DNS name (IPv4 only)
ip-172-31-22-33.eu-central-1.compute.internal
Answer private resource DNS name
IPv4 (A)
Instance type
t2.medium
Elastic IP addresses
–
Auto-assigned IP address
3.71.202.234 [Public IP]
VPC ID
vpc-025b41945b783a001 (vpc)
AWS Compute Optimizer finding
Opt-in to AWS Compute Optimizer for recommendations. | Learn more
IAM Role
–
Subnet ID
subnet-00f1f8cbdf712f9f8
Auto Scaling Group name
–
Details
Security
Networking
Storage
Status checks
Monitoring
Tags
Tags
Manage tags
Key
Value
Name
Production
Production
prod


Code source Info
File
Edit
Find
View
Go
Tools
Window
Test
Deploy
Go to Anything (Ctrl-P)
lambda_function
Environment
lambdaStopEC2
lambda_function.py
import boto3
region = 'eu-central-1'
instances = ['i-049d2d1f57d4b5c48']
ec2 = boto3.client('ec2', region_name=region)
def lambda_handler(event, context):
ec2.stop_instances(InstanceIds=instances)
print('stopped your instances: ' + str(instances))


Code source Info
File
Edit
Find
View
Go
Tools
Window
Test
Deploy
Go to Anything (Ctrl-P)
lambda_function
Environment
lambdaStartEC2
lambda_function.py
import boto3
region = 'eu-central-1'
instances = ['i-049d2d1f57d4b5c48']
ec2 = boto3.client('ec2', region_name=region)
def lambda_handler(event, context):
ec2.start_instances(InstanceIds=instances)
print('started your instances: ' + str(instances))

lambdaEC2stop

Rule details Info

Rule name
lambdaEC2stop

Status
Enabled

Event bus name
default

Description
Eventbridge uses lambda command to stop EC2 instances

Rule ARN
arn:aws:events:eu-central-1:807013657668:rule/lambdaEC2stop

Event bus ARN
arn:aws:events:eu-central-1:807013657668:event-bus/default

Event schedule | Targets | Monitoring | Tags

Event schedule Info

Cron expression

0 00 ? * * *

Next 10 trigger date(s)

UTC

Sun, 16 Oct 2022 00:00:00 UTC
Mon, 17 Oct 2022 00:00:00 UTC
Tue, 18 Oct 2022 00:00:00 UTC
Wed, 19 Oct 2022 00:00:00 UTC
Thu, 20 Oct 2022 00:00:00 UTC
Fri, 21 Oct 2022 00:00:00 UTC
Sat, 22 Oct 2022 00:00:00 UTC
Sun, 23 Oct 2022 00:00:00 UTC
Mon, 24 Oct 2022 00:00:00 UTC
Tue, 25 Oct 2022 00:00:00 UTC

lambdaEC2start

Rule details Info

Rule name
lambdaEC2start

Status
Enabled

Event bus name
default

Description
Starts EC2 instances at 9am on weekdays

Rule ARN
arn:aws:events:eu-central-1:807013657668:rule/lambdaEC2start

Event bus ARN
arn:aws:events:eu-central-1:807013657668:event-bus/default

Event schedule | Targets | Monitoring | Tags

Event schedule Info

Cron expression

0 8 ? * MON-FRI *

Next 10 trigger date(s)

UTC

Mon, 17 Oct 2022 08:00:00 UTC
Tue, 18 Oct 2022 08:00:00 UTC
Wed, 19 Oct 2022 08:00:00 UTC
Thu, 20 Oct 2022 08:00:00 UTC
Fri, 21 Oct 2022 08:00:00 UTC
Mon, 24 Oct 2022 08:00:00 UTC
Tue, 25 Oct 2022 08:00:00 UTC
Wed, 26 Oct 2022 08:00:00 UTC
Thu, 27 Oct 2022 08:00:00 UTC
Fri, 28 Oct 2022 08:00:00 UTC

Provide screenshots (evidence) for your solution. Always explain your evidence! As a prof, we expect at least:

- All the CRON expressions used in the solution
- Lambda function(s) code that implements the requirements