Schedule a server start/stop function using Lambda and CloudWatch

Goal: Automate the start/stop of ec2 instance using lambda and CloudWatch services.

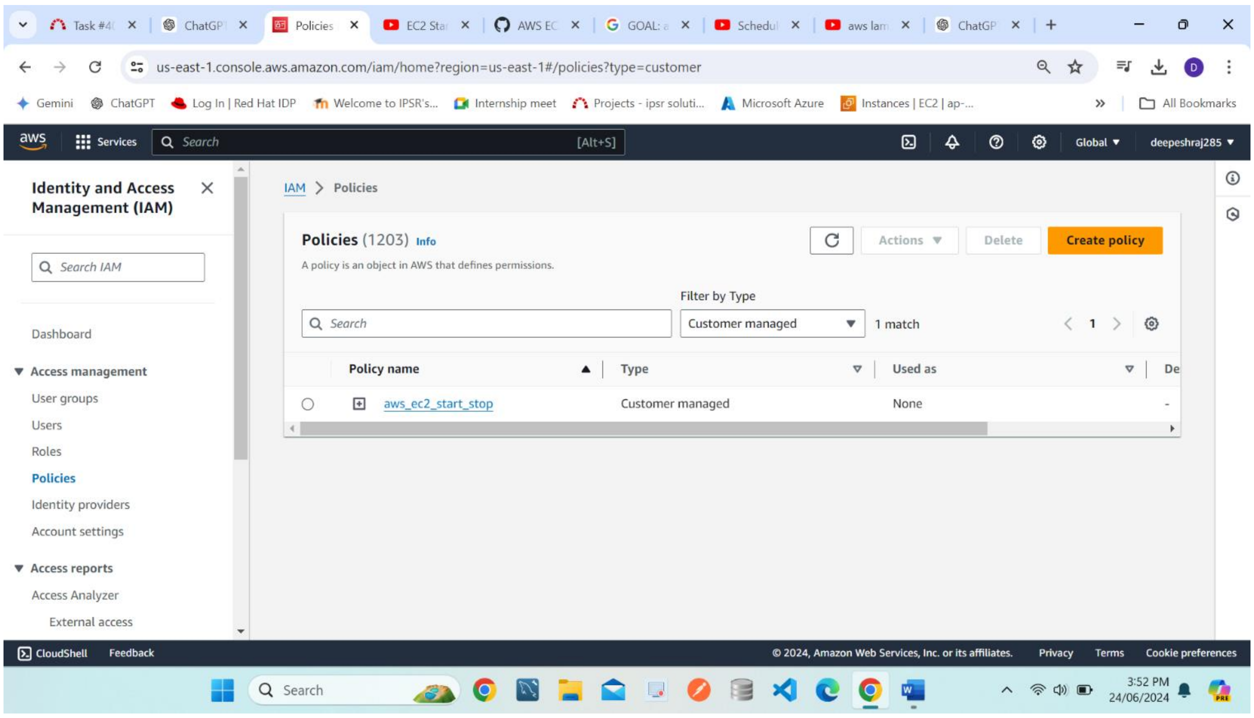
1. Create required IAM policies and Roles
2. Create function under aws-lambda which determines the start/stop process of specific ec2-instance.
3. Test the function created under lambda
4. Create rules inside CloudWatch services which trigger the functions created in lambda {cron}

Steps:

1. Login to the aws account and create an instance in Linux based operating system
2. Go to the IAM Console and create policy
    a. Click on "Policies" from the sidebar.
    b. Click on the "Create policy" button.
    c. Switch to the "JSON" tab and paste the following JSON:

```
{
    "Version": "2012-10-17",
    "Statement": [
      {
        "Effect": "Allow",
        "Action": [
          "ec2:Start*",
          "ec2:Stop*",
          "ec2:DescribeInstanceStatus"
        ],
        "Resource": "*"
      },
      {
        "Sid": "VisualEditor1",
        "Effect": "Allow",
        "Action": [
          "logs:CreateLogStream",
          "logs:CreateLogGroup",
          "logs:PutLogEvents"
        ],
        "Resource": "arn:aws:logs:*:*:*"
      }
    ]
}
```

    d. Name the policy aws_ec2_start_stop and click create.


IAM > Policies

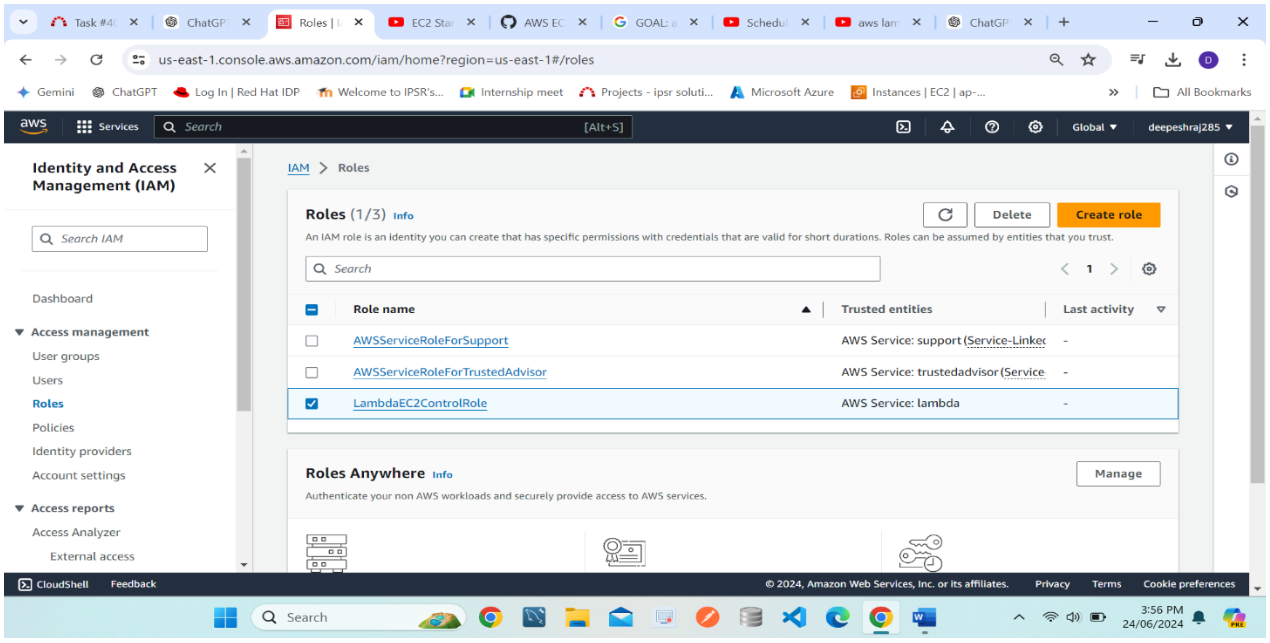
Policies (1203) Info

A policy is an object in AWS that defines permissions.

Filter by Type: Customer managed — 1 match

Policy name	Type	Used as
aws_ec2_start_stop	Customer managed	None

3. Create Role and attach policy
   a. In the IAM console, click on "Roles" from the sidebar.
   b. Click on the "Create role" button.
   c. Choose "AWS service" and select "Lambda".
   d. Click "Next: Permissions".
   e. Search for LambdaEC2ControlPolicy and check the box next to it.
   f. Click on "Next: Tags" and then Click on "Next: Review".
   g. Name the role LambdaEC2ControlRole and Click on "Create role".


IAM > Roles

Roles (1/3) Info

An IAM role is an identity you can create that has specific permissions with credentials that are valid for short durations. Roles can be assumed by entities that you trust.

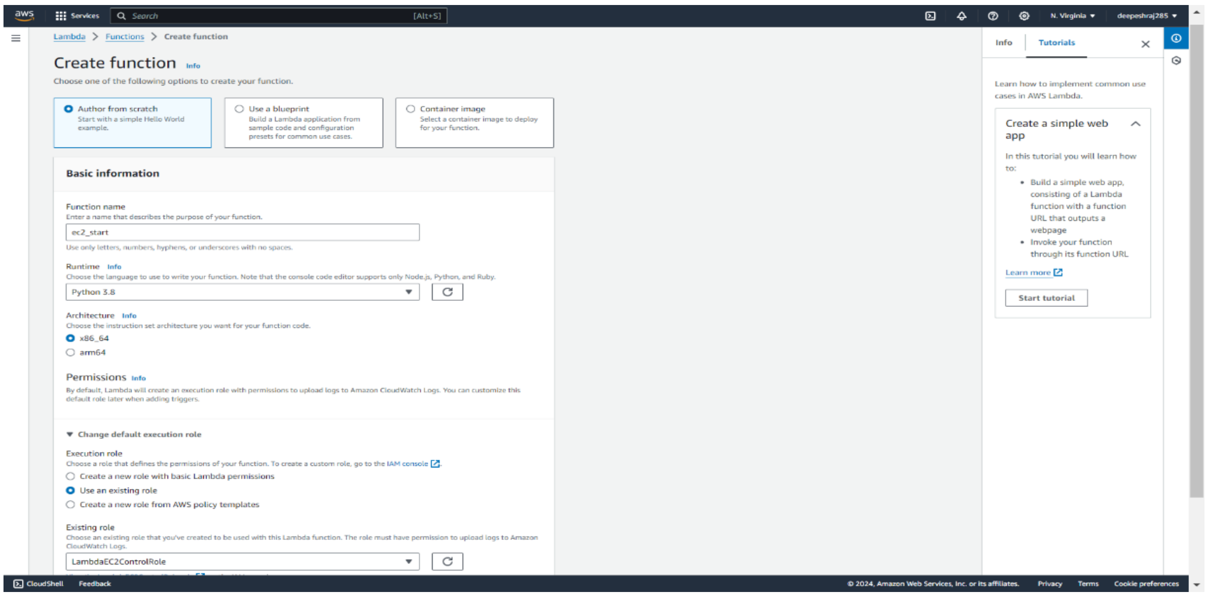
Role name	Trusted entities	Last activity
AWSServiceRoleForSupport	AWS Service: support (Service-Linkec	-
AWSServiceRoleForTrustedAdvisor	AWS Service: trustedadvisor (Service	-
✓ LambdaEC2ControlRole	AWS Service: lambda	-

Roles Anywhere Info

Authenticate your non AWS workloads and securely provide access to AWS services.

4. Create Lambda function

- Open the AWS Management Console and navigate to the Lambda service.
- Click on "Create function". Then choose "Author from scratch".
- Name the function ec2-start.
- Select the runtime as Python 3.x (or any preferred language).
- Under "Permissions", choose "Use an existing role".
- Select the role LambdaEC2ControlRole created earlier.
- Click on "Create function".



Lambda > Functions > Create function

Create function Info

Choose one of the following options to create your function.

[✓ Author from scratch] Start with a simple Hello World example.
Use a blueprint: Build a Lambda application from sample code and configuration presets for common use cases.
Container image: Select a container image to deploy for your function.

Basic information

Function name
Enter a name that describes the purpose of your function.
ec2_start
Use only letters, numbers, hyphens, or underscores with no spaces.

Runtime Info
Choose the language to use to write your function. Note that the console code editor supports only Node.js, Python, and Ruby.
Python 3.8

Architecture Info
Choose the instruction set architecture you want for your function code.
[✓ x86_64] arm64

Permissions Info
By default, Lambda will create an execution role with permissions to upload logs to Amazon CloudWatch Logs. You can customize this default role later when adding triggers.

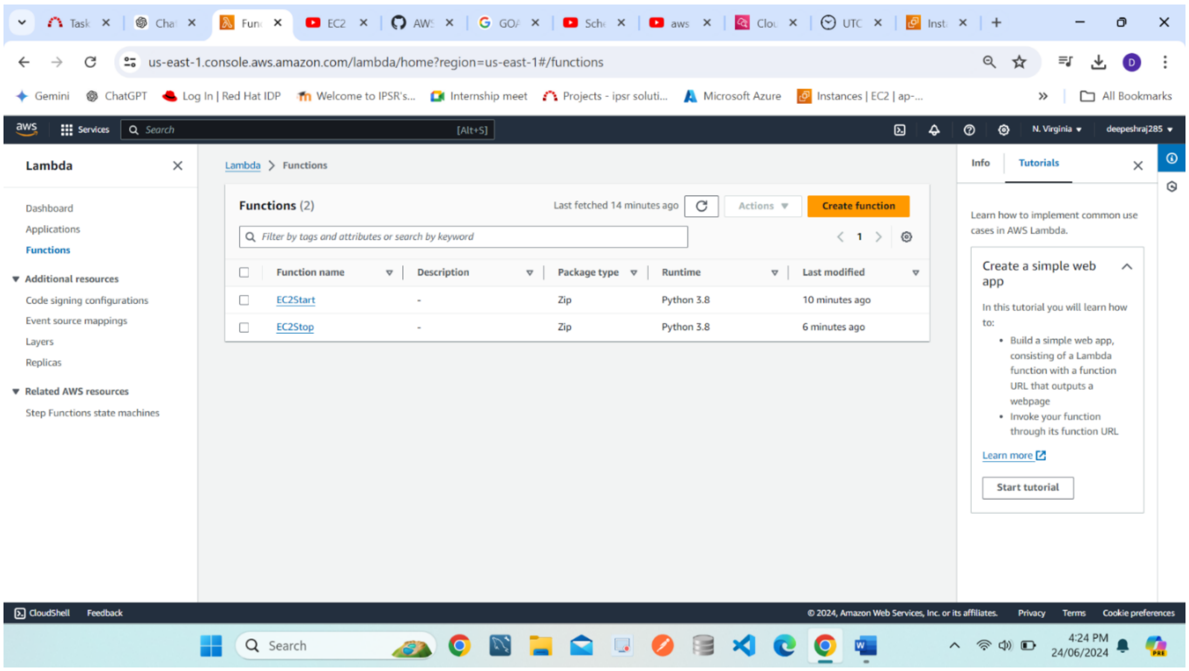
Change default execution role

Execution role
Choose a role that defines the permissions of your function. To create a custom role, go to the IAM console.
Create a new role with basic Lambda permissions
[✓ Use an existing role]
Create a new role from AWS policy templates

Existing role
Choose an existing role that you've created to be used with this Lambda function. The role must have permission to upload logs to Amazon CloudWatch Logs.
LambdaEC2ControlRole

Search
Services
N. Virginia
deepeshraj285
Info
Tutorials
Learn how to implement common use cases in AWS Lambda.
Create a simple web app
In this tutorial you will learn how to:
Build a simple web app, consisting of a Lambda function with a function URL that outputs a webpage
Invoke your function through its function URL
Learn more
Start tutorial
CloudShell
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

- Likewise, Ec2-stop lambda function can be created



us-east-1.console.aws.amazon.com/lambda/home?region=us-east-1#/functions

Lambda > Functions

Functions (2)   Last fetched 14 minutes ago

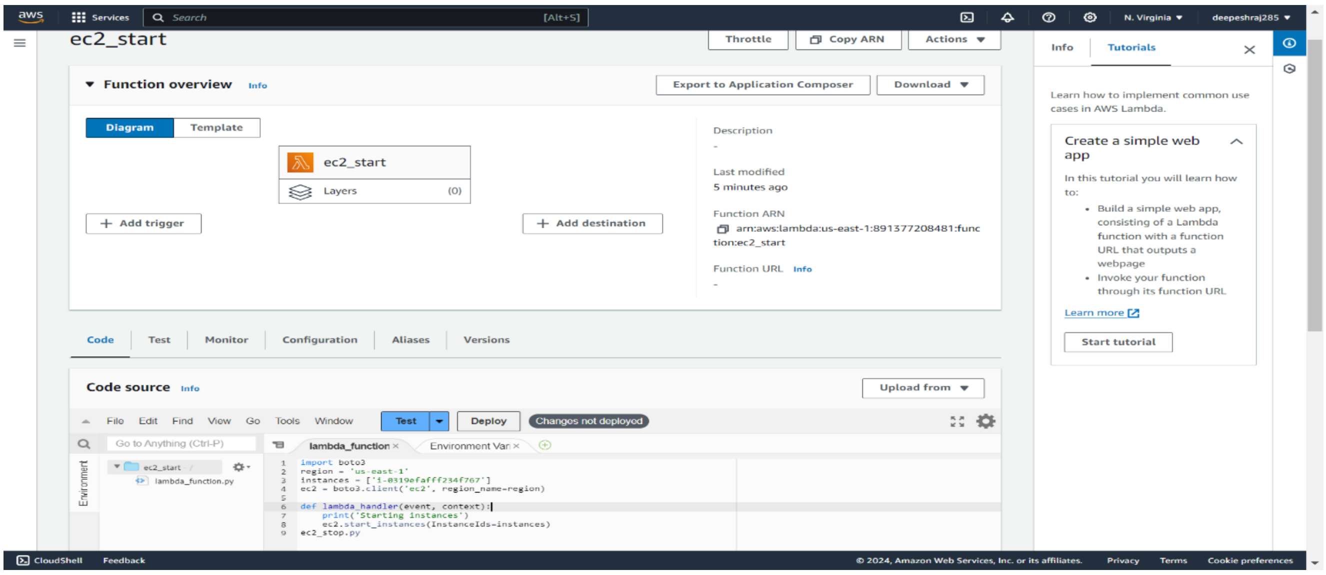
	Function name	Description	Package type	Runtime	Last modified
☐	EC2Start	-	Zip	Python 3.8	10 minutes ago
☐	EC2Stop	-	Zip	Python 3.8	6 minutes ago

5. Edit Function Code for ec2-start function and save it

- In the Lambda console, under the "Function code" section, replace the default code with the following Python code:

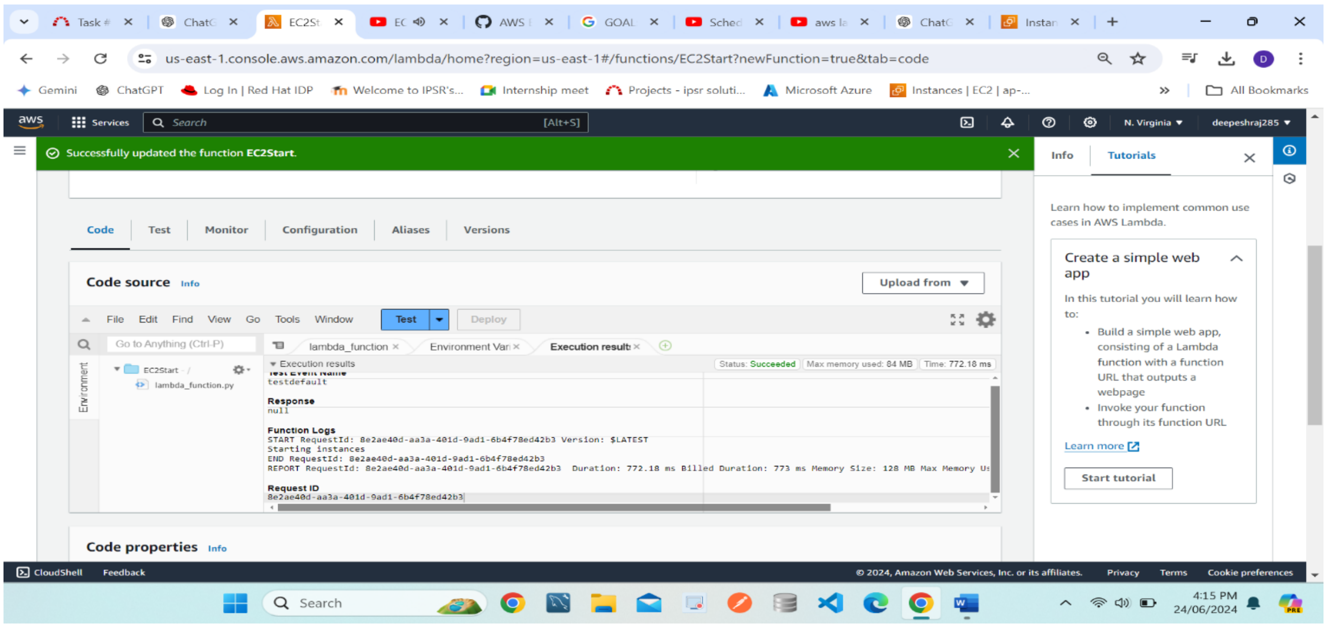
```
import boto3
region = 'us-east-1'
instances = ['i-001e2d04e37ccde3b']
ec2 = boto3.client('ec2', region_name=region)

def lambda_handler(event, context):
    print('Starting instances')
    ec2.start_instances(InstanceIds=instances)
```

- Click on "Deploy" to save the changes.



```
import boto3
region = 'us-east-1'
instances = ['i-0319efafff234f767']
ec2 = boto3.client('ec2', region_name=region)

def lambda_handler(event, context):
    print('Starting instances')
    ec2.start_instances(InstanceIds=instances)
ec2_stop.py
```

- We can test it properly working by clicking deploy and test



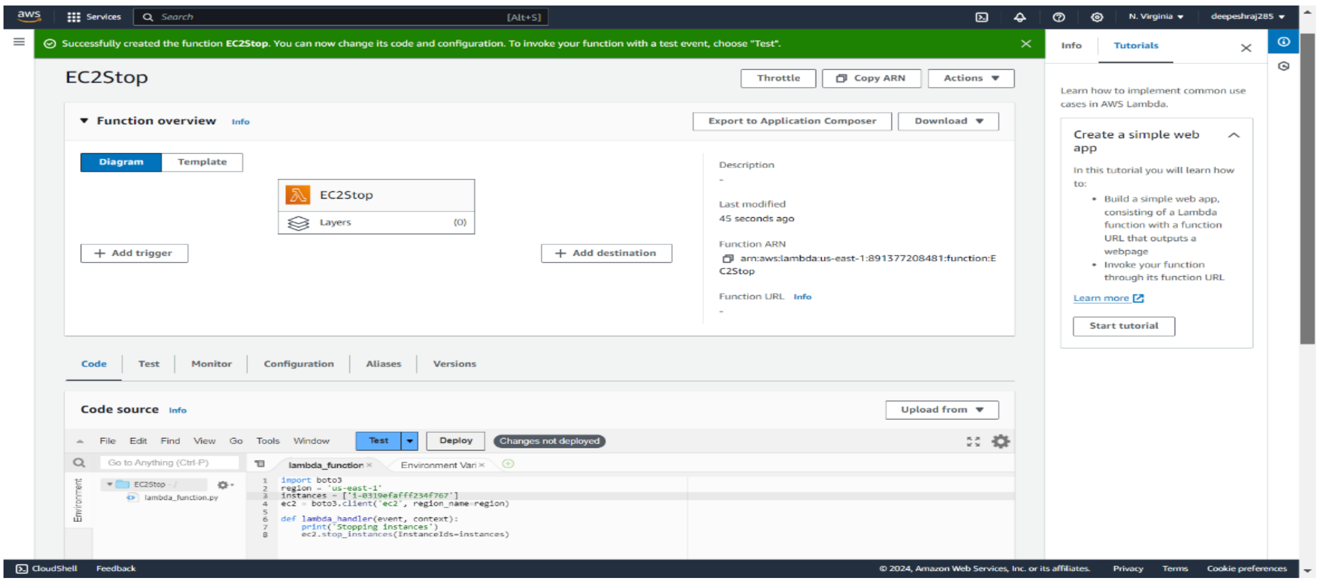
Successfully updated the function EC2Start.

Execution results — Status: Succeeded | Max memory used: 84 MB | Time: 772.18 ms

```
Test Event Name
testdefault

Response
null

Function Logs
START RequestId: 8e2ae40d-aa3a-401d-9ad1-6b4f78ed42b3 Version: $LATEST
Starting instances
END RequestId: 8e2ae40d-aa3a-401d-9ad1-6b4f78ed42b3
REPORT RequestId: 8e2ae40d-aa3a-401d-9ad1-6b4f78ed42b3  Duration: 772.18 ms Billed Duration: 773 ms Memory Size: 128 MB Max Memory Us

Request ID
8e2ae40d-aa3a-401d-9ad1-6b4f78ed42b3
```

6. Repeat the process to create another test event named EC2Stop lambda function and test it if the instance is stopped or not

```
import boto3
region = 'us-east-1'
instances = ['i-001e2d04e37ccde3b']
ec2 = boto3.client('ec2', region_name=region)

def lambda_handler(event, context):
    print('Stopping instances')
    ec2.stop_instances(InstanceIds=instances)
```

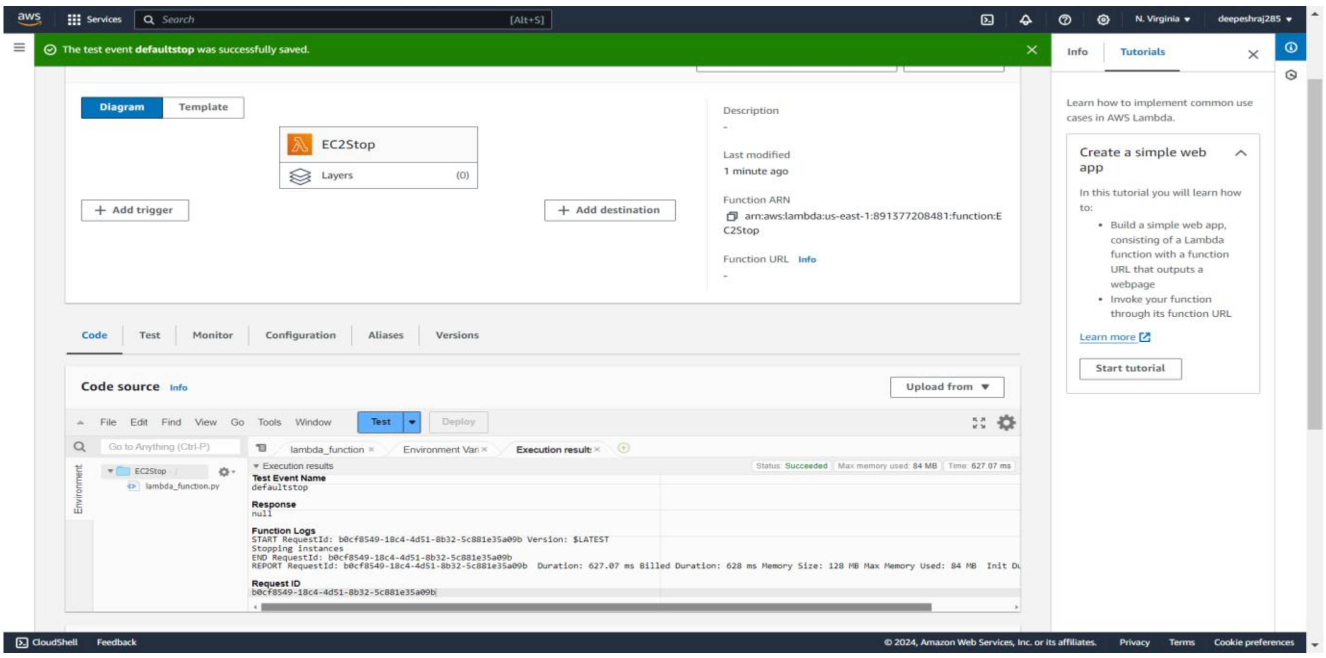


Successfully created the function **EC2Stop**. You can now change its code and configuration. To invoke your function with a test event, choose "Test".

EC2Stop

Description: -
Last modified: 45 seconds ago
Function ARN: arn:aws:lambda:us-east-1:891377208481:function:EC2Stop
Function URL: -

```
import boto3
region = 'us-east-1'
instances = ['i-0319efafff234f767']
ec2 = boto3.client('ec2', region_name=region)

def lambda_handler(event, context):
    print('Stopping instances')
    ec2.stop_instances(InstanceIds=instances)
```



The test event **defaultstop** was successfully saved.

Description: -
Last modified: 1 minute ago
Function ARN: arn:aws:lambda:us-east-1:891377208481:function:EC2Stop
Function URL: -

Execution results — Status: Succeeded | Max memory used: 84 MB | Time: 627.07 ms

Test Event Name
```
defaultstop
```

Response
```
null
```

Function Logs
```
START RequestId: b0cf8549-18c4-4d51-8b32-5c881e35a09b Version: $LATEST
Stopping instances
END RequestId: b0cf8549-18c4-4d51-8b32-5c881e35a09b
REPORT RequestId: b0cf8549-18c4-4d51-8b32-5c881e35a09b  Duration: 627.07 ms Billed Duration: 628 ms Memory Size: 128 MB Max Memory Used: 84 MB  Init Du
```

Request ID
```
b0cf8549-18c4-4d51-8b32-5c881e35a09b
```

7. Create CloudWatch Rule to Start Instance and stop instance

1. Go to CloudWatch Console: -

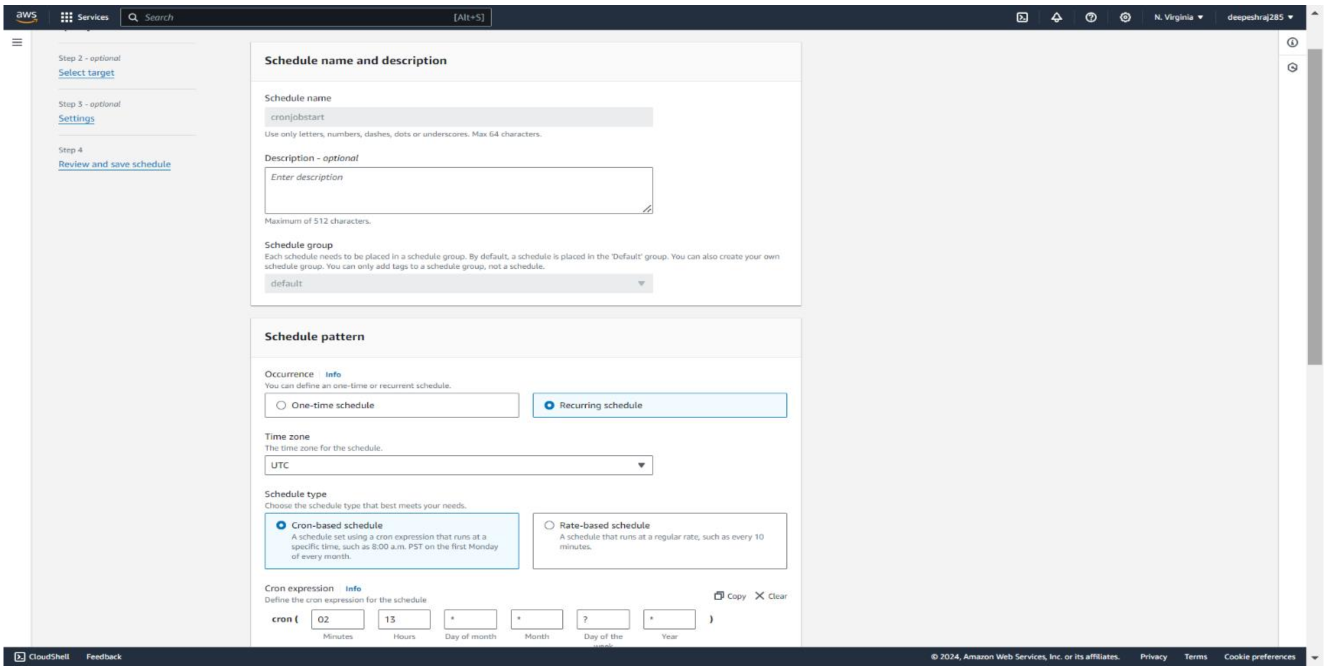
   Open the AWS Management Console and navigate to the CloudWatch service.

2. Create Rule

   Click on "Rules" from the sidebar and Click on the "Create rule" button.

3. Configure Event Source

   - Choose "Event Source" as "EventBridge (CloudWatch Events)".
   - Under "Event Source", choose "Schedule".
   - Configure the schedule using a cron expression. For example, to start the instance at 13.02 every day and stop instance at 13.07 every day (UTC time format)



Step 2 - *optional*
Select target
Step 3 - *optional*
Settings
Step 4
Review and save schedule

Schedule name and description

Schedule name: cronjobstart
Use only letters, numbers, dashes, dots or underscores. Max 64 characters.

Description - *optional*: Enter description
Maximum of 512 characters.

Schedule group
Each schedule needs to be placed in a schedule group. By default, a schedule is placed in the 'Default' group. You can also create your own schedule group. You can only add tags to a schedule group, not a schedule.
default

Schedule pattern

Occurrence
You can define an one-time or recurrent schedule.
One-time schedule [✓ Recurring schedule]

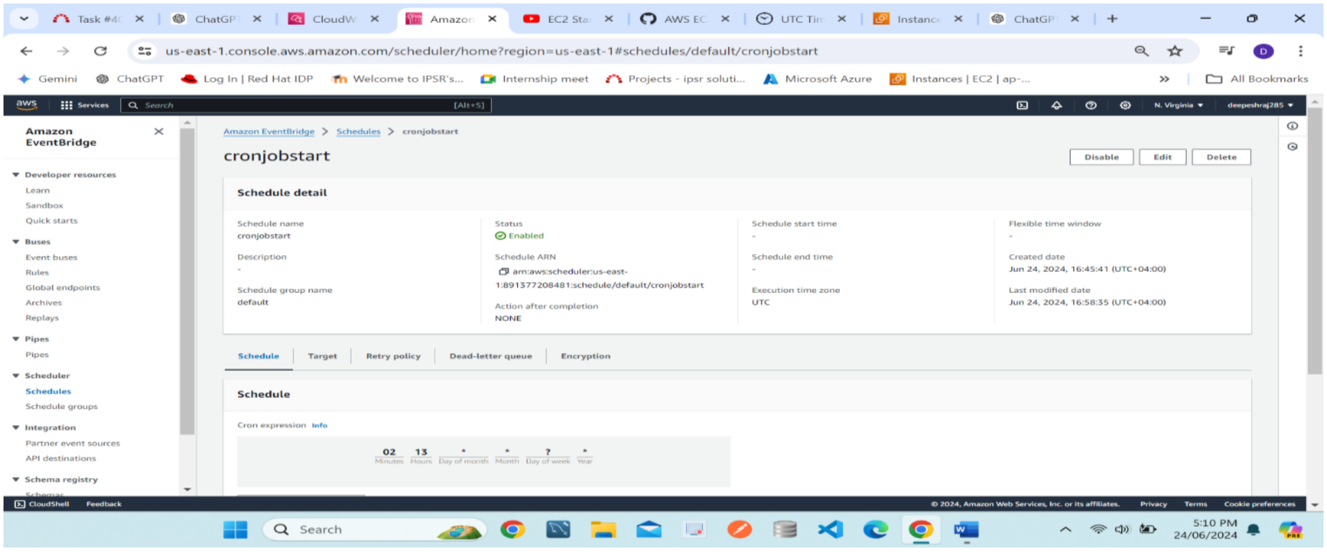
Time zone
The time zone for the schedule.
UTC

Schedule type
Choose the schedule type that best meets your needs.
[✓ Cron-based schedule] A schedule set using a cron expression that runs at a specific time, such as 8:00 a.m. PST on the first Monday of every month.
Rate-based schedule: A schedule that runs at a regular rate, such as every 10 minutes.

Cron expression
Define the cron expression for the schedule

	Minutes	Hours	Day of month	Month	Day of the week	Year	
cron (	02	13	*	*	?	*	)

© 2024, Amazon Web Services, Inc. or its affiliates.



us-east-1.console.aws.amazon.com/scheduler/home?region=us-east-1#schedules/default/cronjobstart

Amazon EventBridge > Schedules > cronjobstart

cronjobstart

Schedule detail

Schedule name: cronjobstart	Status: Enabled	Schedule start time: -	Flexible time window: -
Description: -	Schedule ARN: arn:aws:scheduler:us-east-1:891377208481:schedule/default/cronjobstart	Schedule end time: -	Created date: Jun 24, 2024, 16:45:41 (UTC+04:00)
Schedule group name: default	Action after completion: NONE	Execution time zone: UTC	Last modified date: Jun 24, 2024, 16:58:35 (UTC+04:00)

Schedule

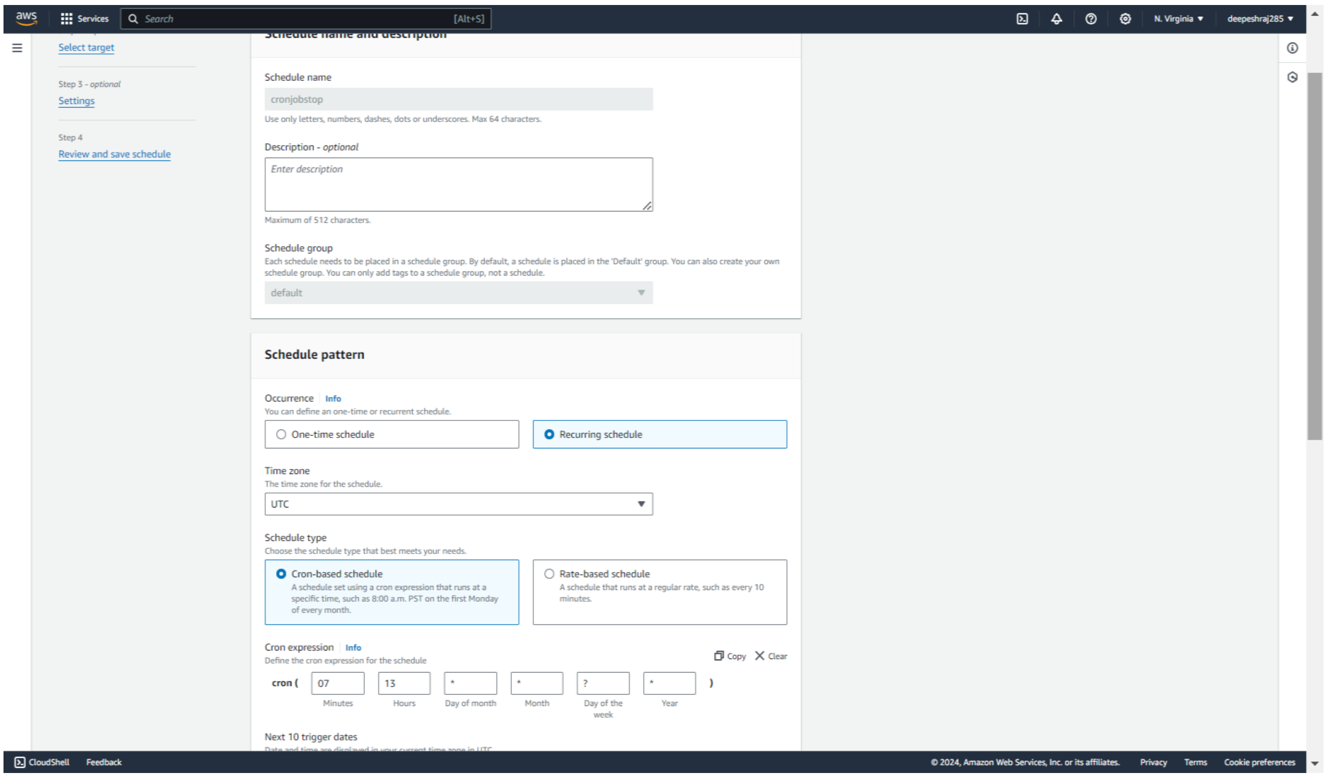
Cron expression

Minutes	Hours	Day of month	Month	Day of week	Year
02	13	*	*	?	*

© 2024, Amazon Web Services, Inc. or its affiliates.

5:10 PM 24/06/2024

Stop instances EventBridge setting



Select target

Step 3 - *optional*
Settings

Step 4
Review and save schedule

Schedule name and description

Schedule name
cronjobstop
Use only letters, numbers, dashes, dots or underscores. Max 64 characters.

Description - *optional*
Enter description
Maximum of 512 characters.

Schedule group
Each schedule needs to be placed in a schedule group. By default, a schedule is placed in the 'Default' group. You can also create your own schedule group. You can only add tags to a schedule group, not a schedule.
default

Schedule pattern

Occurrence | Info
You can define an one-time or recurrent schedule.
One-time schedule [✓ Recurring schedule]

Time zone
The time zone for the schedule.
UTC

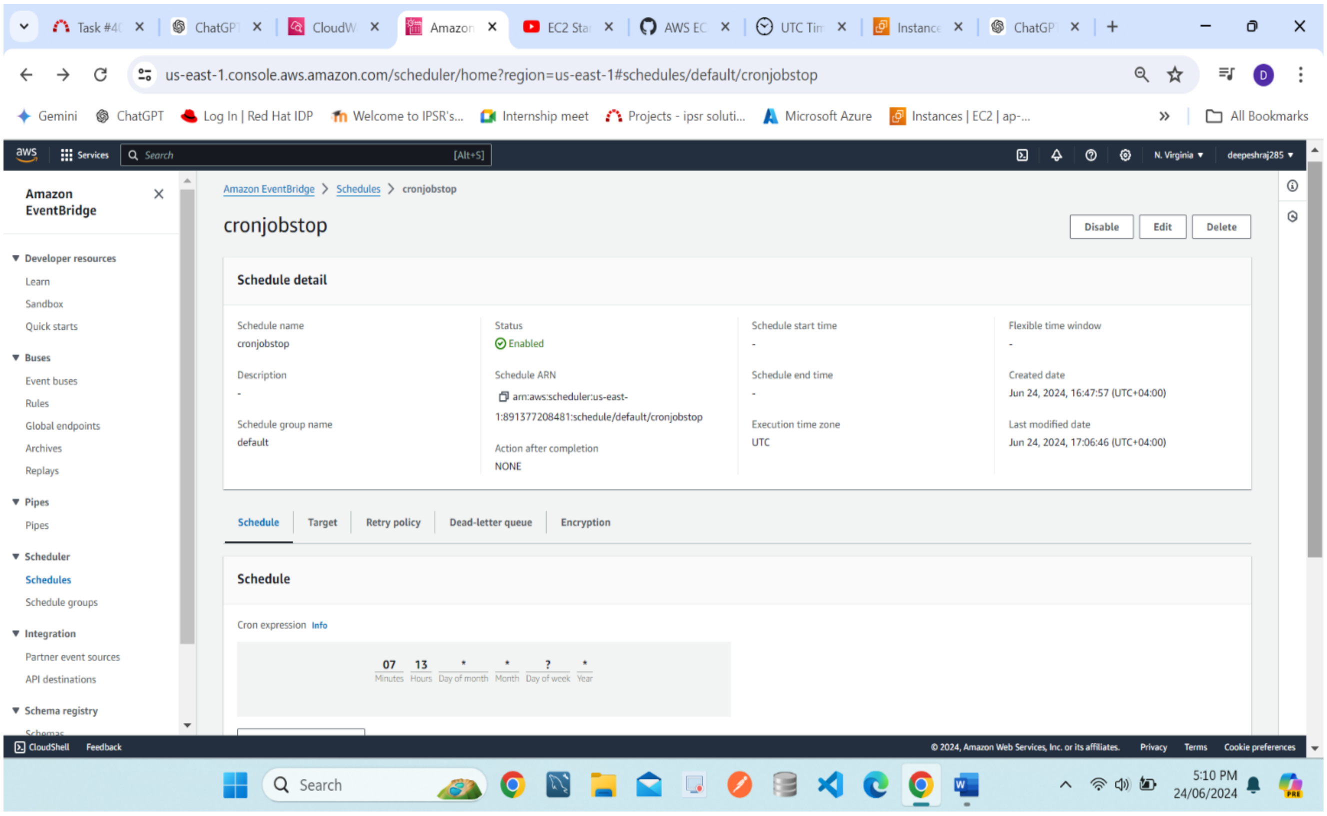
Schedule type
Choose the schedule type that best meets your needs.
[✓ Cron-based schedule — A schedule set using a cron expression that runs at a specific time, such as 8:00 a.m. PST on the first Monday of every month.]
Rate-based schedule — A schedule that runs at a regular rate, such as every 10 minutes.

Cron expression | Info
Define the cron expression for the schedule

cron (	07	13	*	*	?	*)
	Minutes	Hours	Day of month	Month	Day of the week	Year

Next 10 trigger dates

© 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences



us-east-1.console.aws.amazon.com/scheduler/home?region=us-east-1#schedules/default/cronjobstop

Amazon EventBridge > Schedules > cronjobstop

cronjobstop

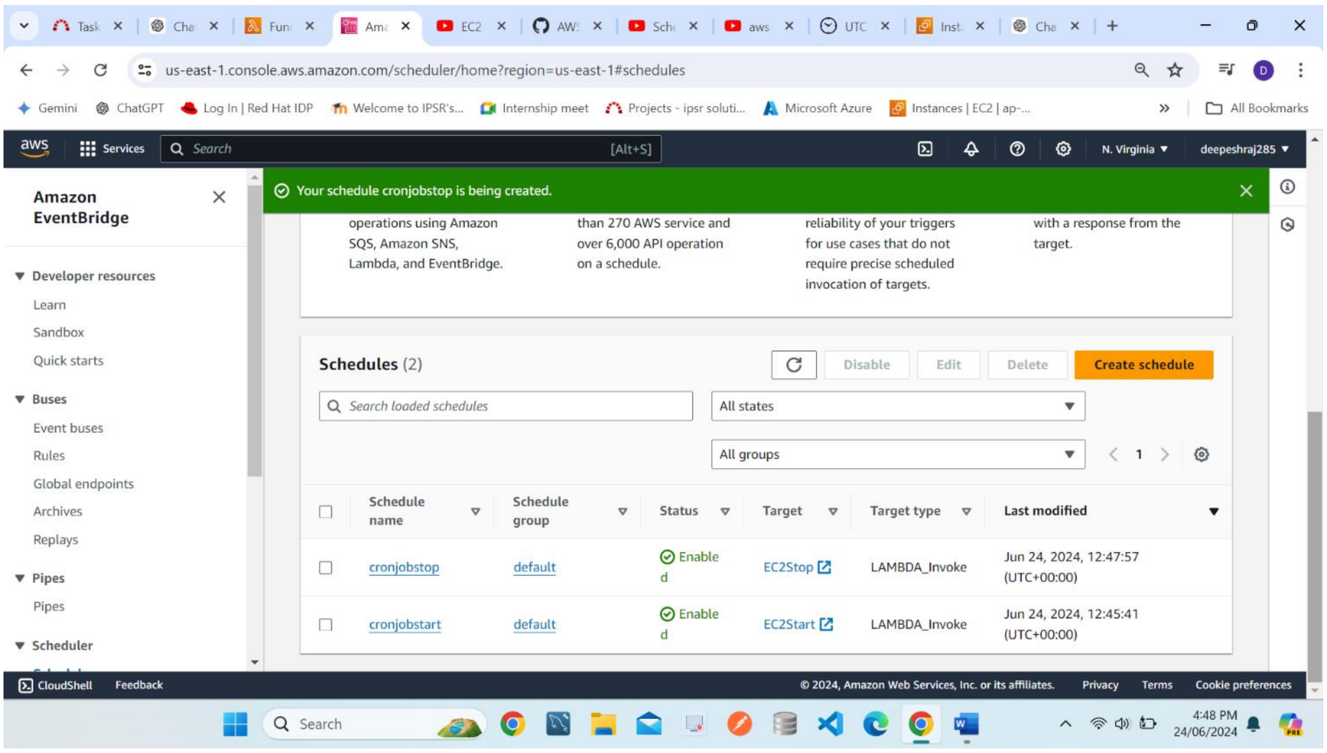
Schedule detail

Schedule name: cronjobstop	Status: Enabled	Schedule start time: -	Flexible time window: -
Description: -	Schedule ARN: arn:aws:scheduler:us-east-1:891377208481:schedule/default/cronjobstop	Schedule end time: -	Created date: Jun 24, 2024, 16:47:57 (UTC+04:00)
Schedule group name: default	Action after completion: NONE	Execution time zone: UTC	Last modified date: Jun 24, 2024, 17:06:46 (UTC+04:00)

Schedule

Cron expression Info

07	13	*	*	?	*
Minutes	Hours	Day of month	Month	Day of week	Year


us-east-1.console.aws.amazon.com/scheduler/home?region=us-east-1#schedules

Amazon EventBridge

Your schedule cronjobstop is being created.

operations using Amazon SQS, Amazon SNS, Lambda, and EventBridge. | than 270 AWS service and over 6,000 API operation on a schedule. | reliability of your triggers for use cases that do not require precise scheduled invocation of targets. | with a response from the target.

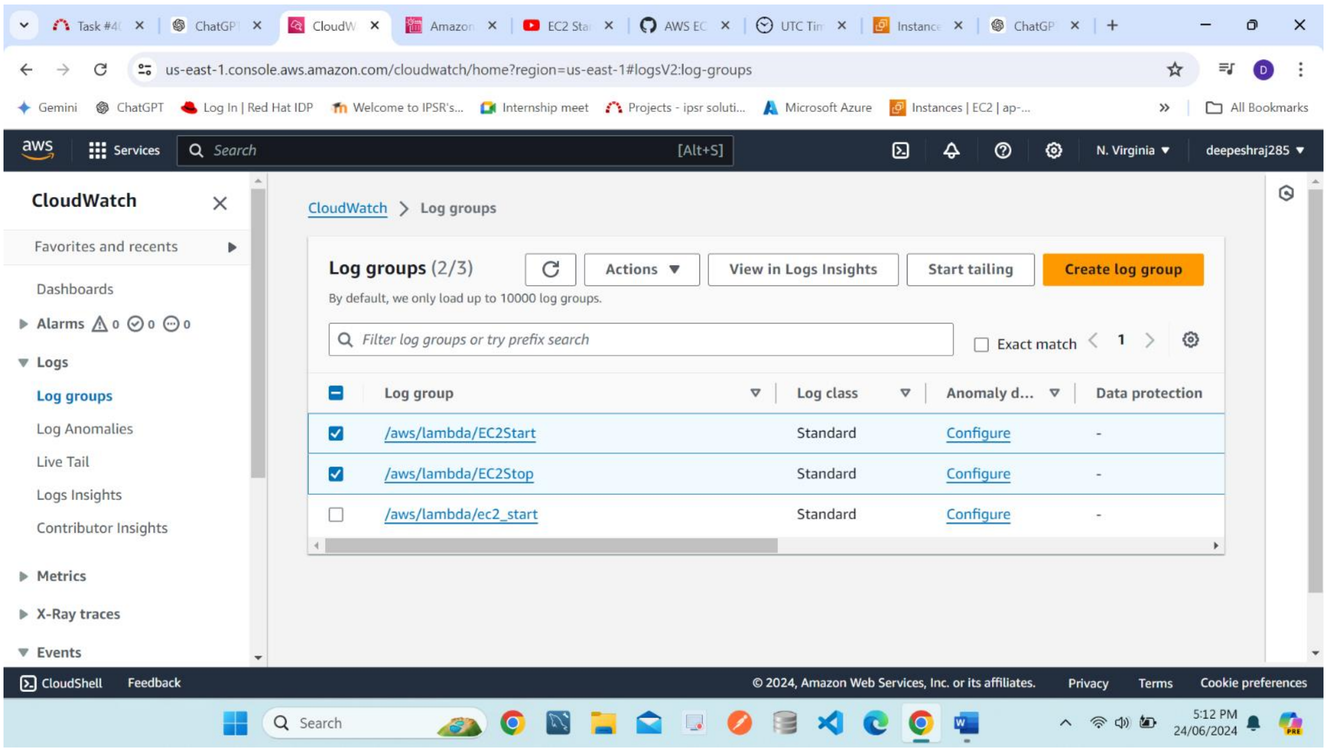
Schedules (2)

Schedule name	Schedule group	Status	Target	Target type	Last modified
cronjobstop	default	Enabled	EC2Stop	LAMBDA_Invoke	Jun 24, 2024, 12:47:57 (UTC+00:00)
cronjobstart	default	Enabled	EC2Start	LAMBDA_Invoke	Jun 24, 2024, 12:45:41 (UTC+00:00)

© 2024, Amazon Web Services, Inc. or its affiliates.

8. Execution results of the Lambda function in the CloudWatch Logs to verify that the instances are being started and stopped as expected.

Log groups


us-east-1.console.aws.amazon.com/cloudwatch/home?region=us-east-1#logsV2:log-groups

CloudWatch > Log groups

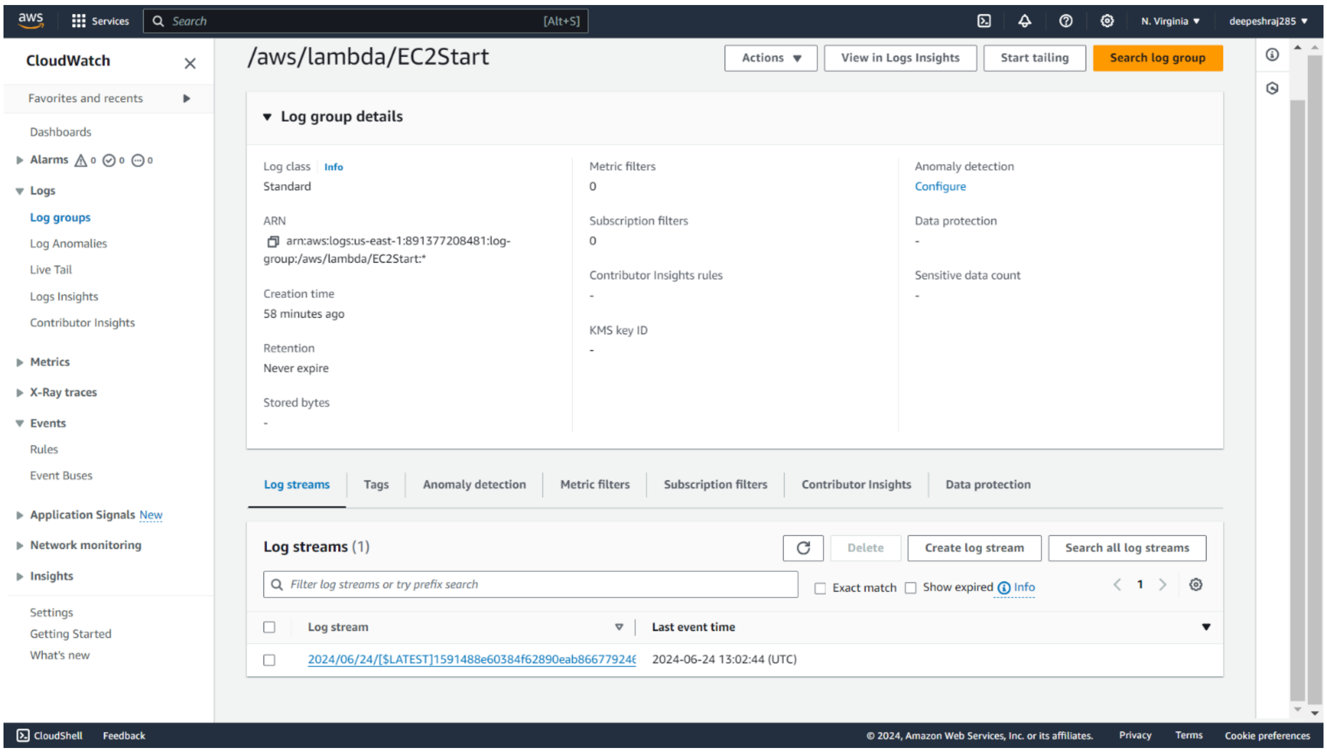
Log groups (2/3)

By default, we only load up to 10000 log groups.

	Log group	Log class	Anomaly d...	Data protection
☑	/aws/lambda/EC2Start	Standard	Configure	-
☑	/aws/lambda/EC2Stop	Standard	Configure	-
☐	/aws/lambda/ec2_start	Standard	Configure	-

© 2024, Amazon Web Services, Inc. or its affiliates.

Log for EC2Start function



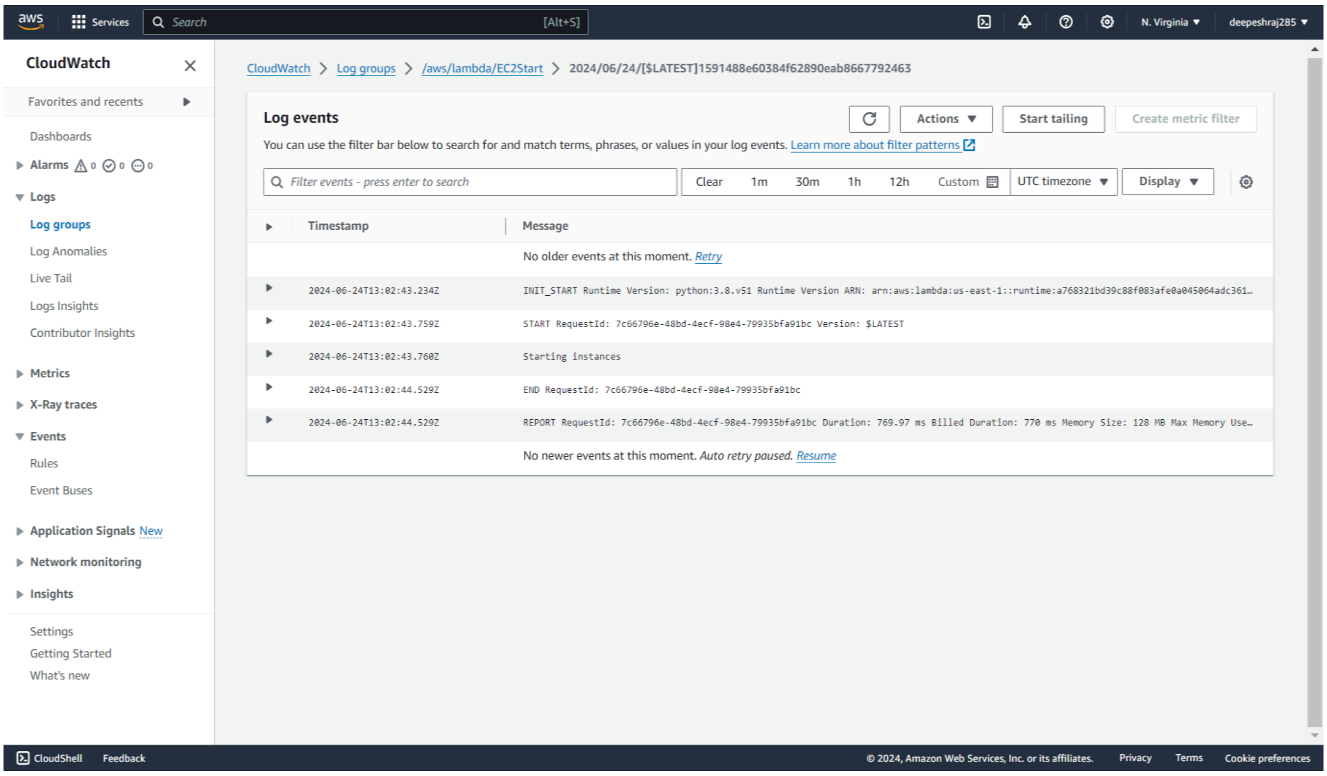
/aws/lambda/EC2Start

Log group details

Log class: Standard
ARN: arn:aws:logs:us-east-1:891377208481:log-group:/aws/lambda/EC2Start:*
Creation time: 58 minutes ago
Retention: Never expire
Stored bytes: -
Metric filters: 0
Subscription filters: 0
Contributor Insights rules: -
KMS key ID: -
Anomaly detection: Configure
Data protection: -
Sensitive data count: -

Log streams (1)

Log stream	Last event time
2024/06/24/[$LATEST]1591488e60384f62890eab86677924€	2024-06-24 13:02:44 (UTC)



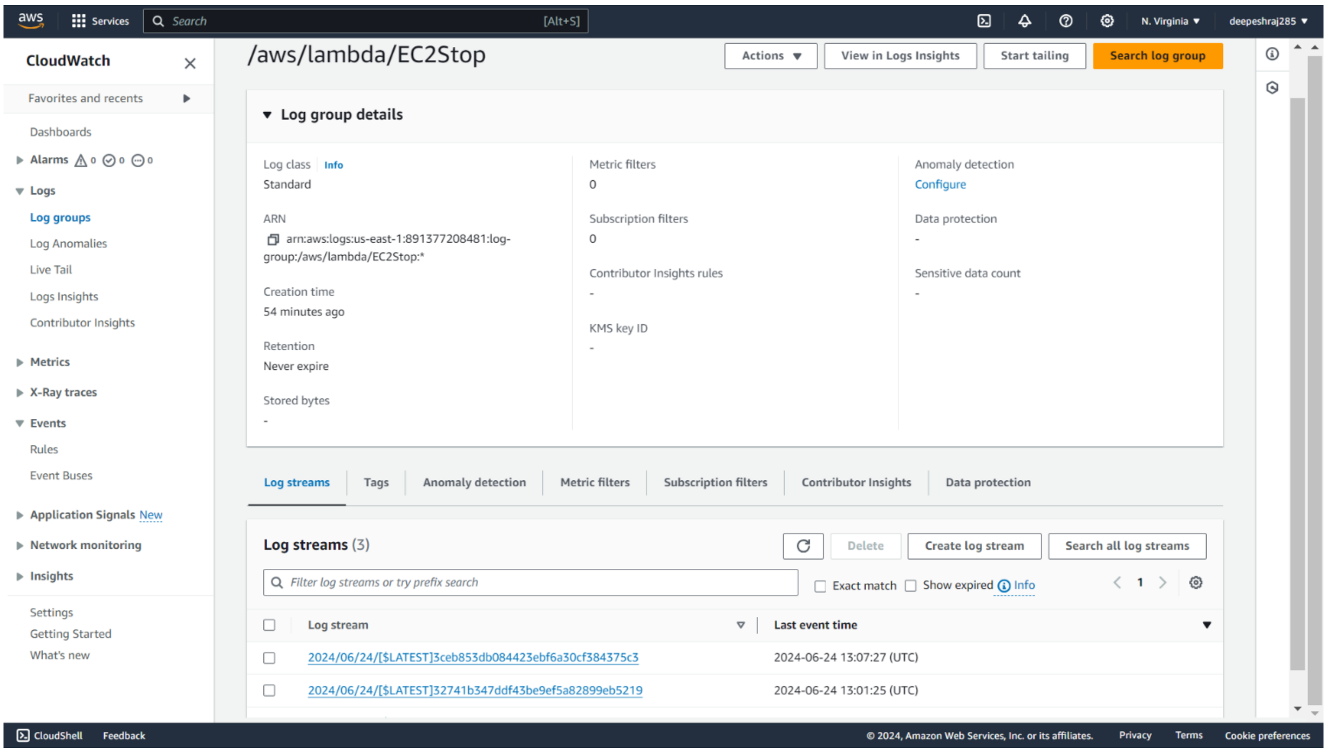
CloudWatch > Log groups > /aws/lambda/EC2Start > 2024/06/24/[$LATEST]1591488e60384f62890eab8667792463

Log events

You can use the filter bar below to search for and match terms, phrases, or values in your log events. Learn more about filter patterns

```
Timestamp                  Message
                           No older events at this moment. Retry
2024-06-24T13:02:43.234Z   INIT_START Runtime Version: python:3.8.v51 Runtime Version ARN: arn:aws:lambda:us-east-1::runtime:a768321bd39c88f083afe0a045064adc361…
2024-06-24T13:02:43.759Z   START RequestId: 7c66796e-48bd-4ecf-98e4-79935bfa91bc Version: $LATEST
2024-06-24T13:02:43.760Z   Starting instances
2024-06-24T13:02:44.529Z   END RequestId: 7c66796e-48bd-4ecf-98e4-79935bfa91bc
2024-06-24T13:02:44.529Z   REPORT RequestId: 7c66796e-48bd-4ecf-98e4-79935bfa91bc Duration: 769.97 ms Billed Duration: 770 ms Memory Size: 128 MB Max Memory Use…
                           No newer events at this moment. Auto retry paused. Resume
```

Log for EC2Stop function

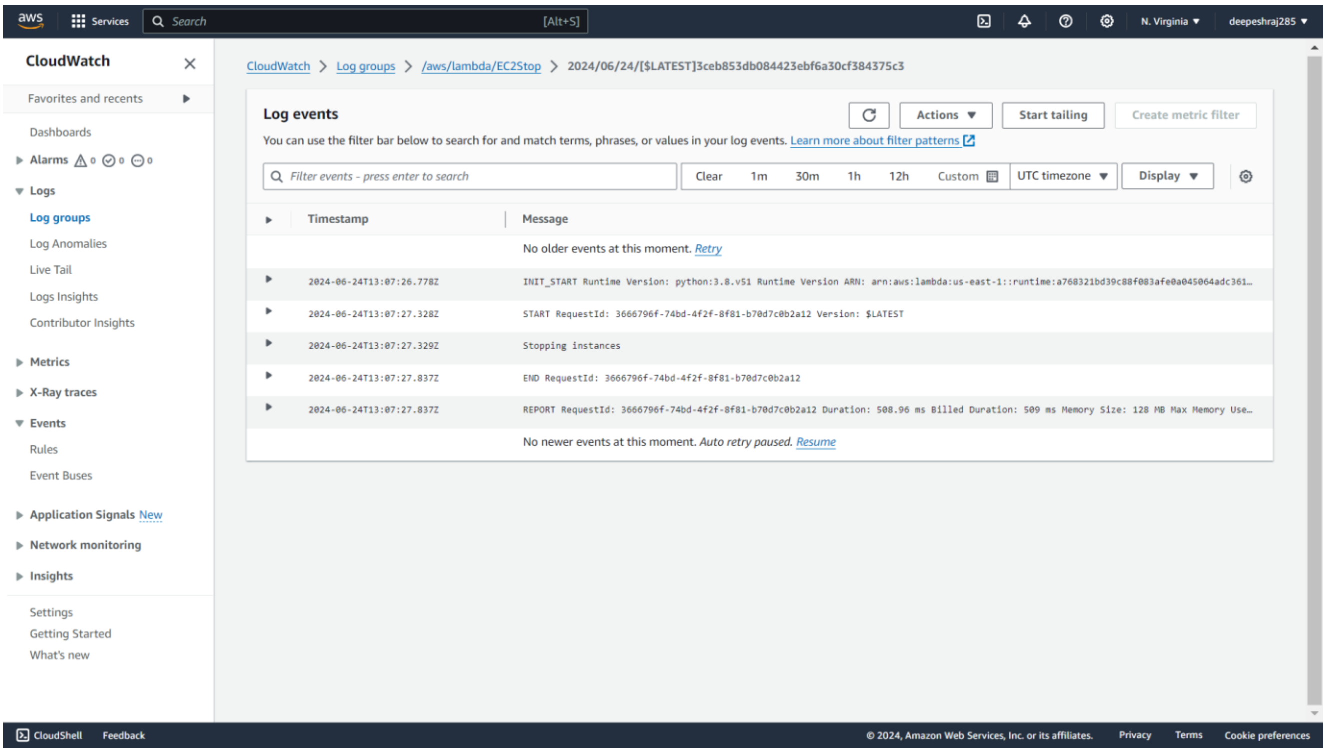

/aws/lambda/EC2Stop

Log group details

Log class: Standard	Metric filters: 0	Anomaly detection: Configure
ARN: arn:aws:logs:us-east-1:891377208481:log-group:/aws/lambda/EC2Stop:*	Subscription filters: 0	Data protection: -
Creation time: 54 minutes ago	Contributor Insights rules: -	Sensitive data count: -
Retention: Never expire	KMS key ID: -	
Stored bytes: -		

Log streams (3)

Log stream	Last event time
2024/06/24/[$LATEST]3ceb853db084423ebf6a30cf384375c3	2024-06-24 13:07:27 (UTC)
2024/06/24/[$LATEST]32741b347ddf43be9ef5a82899eb5219	2024-06-24 13:01:25 (UTC)


CloudWatch > Log groups > /aws/lambda/EC2Stop > 2024/06/24/[$LATEST]3ceb853db084423ebf6a30cf384375c3

Log events

You can use the filter bar below to search for and match terms, phrases, or values in your log events. Learn more about filter patterns

```
Timestamp                  Message
                           No older events at this moment. Retry
2024-06-24T13:07:26.778Z   INIT_START Runtime Version: python:3.8.v51 Runtime Version ARN: arn:aws:lambda:us-east-1::runtime:a768321bd39c88f083afe0a045064adc361…
2024-06-24T13:07:27.328Z   START RequestId: 3666796f-74bd-4f2f-8f81-b70d7c0b2a12 Version: $LATEST
2024-06-24T13:07:27.329Z   Stopping instances
2024-06-24T13:07:27.837Z   END RequestId: 3666796f-74bd-4f2f-8f81-b70d7c0b2a12
2024-06-24T13:07:27.837Z   REPORT RequestId: 3666796f-74bd-4f2f-8f81-b70d7c0b2a12 Duration: 508.96 ms Billed Duration: 509 ms Memory Size: 128 MB Max Memory Use…
                           No newer events at this moment. Auto retry paused. Resume
```