Mini Project: Automating Start/Stop of EC2 Instances Using AWS Lambda

- In this project, I implemented automation for starting and stopping EC2 instances in AWS. The steps involved were:

Steps for it :

1)Created EC2 Instances:

Launched the required EC2 instances that needed to be automated.

2)Configured IAM Role with Policies:

Created an IAM role with the necessary permissions (such as AmazonEC2FullAccess and AWSLambdaBasicExecutionRole) to allow Lambda to manage EC2 instances.

3)Developed Lambda Function:

Wrote a Lambda function (in Python) to start or stop the EC2 instances by using the AWS SDK (boto3).
Attached the IAM role to Lambda so it could execute the required actions.

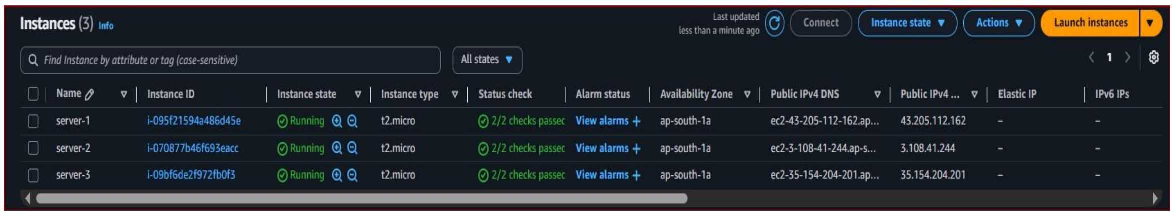
4)Created EventBridge (CloudWatch) Rule:

Set up an EventBridge rule to trigger the Lambda function on a schedule (for example, stop instances at night and start them in the morning).

```
┌──────────────────────┐
│ Amazon EventBridge │
│  (Scheduled Rule)  │
└──────────┬───────────┘
      │ Triggers
      ▼
┌──────────────────────┐
│  AWS Lambda      │
│ (Start/Stop Script) │
└──────────┬───────────┘
      │ Uses IAM Role Permissions
      ▼
┌──────────────────────┐
│   EC2 Instances   │
│ (Start / Stop Ops) │
└──────────────────────┘
```

✅ This setup allows EC2 instances to automatically start and stop based on defined schedules, saving cost and improving operational efficiency.

1. Created Instances :

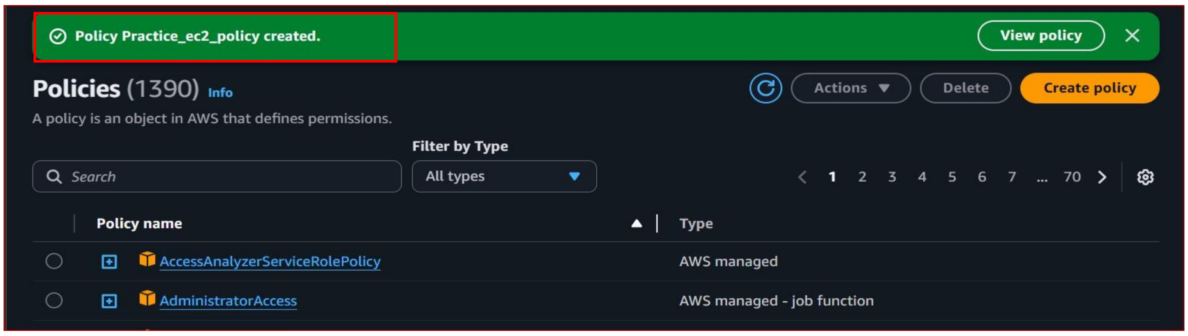


Instances (3) Info

Last updated less than a minute ago

	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	Public IPv4 ...	Elastic IP	IPv6 IPs
	server-1	i-095f21594a486d45e	Running	t2.micro	2/2 checks passec	View alarms +	ap-south-1a	ec2-43-205-112-162.ap...	43.205.112.162	–	–
	server-2	i-070877b46f693eacc	Running	t2.micro	2/2 checks passec	View alarms +	ap-south-1a	ec2-3-108-41-244.ap-s...	3.108.41.244	–	–
	server-3	i-09bf6de2f972fb0f3	Running	t2.micro	2/2 checks passec	View alarms +	ap-south-1a	ec2-35-154-204-201.ap...	35.154.204.201	–	–

2. Creating Policies to IAM Role:



Policy Practice_ec2_policy created.

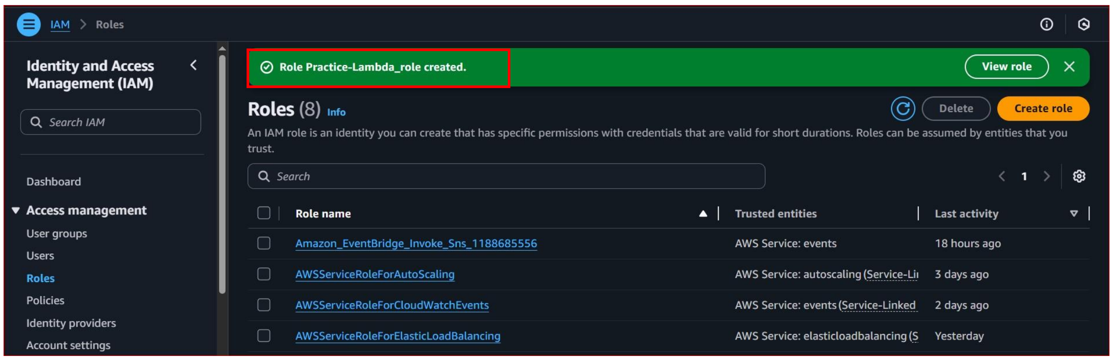
Policies (1390) Info

A policy is an object in AWS that defines permissions.

Filter by Type: All types

Policy name	Type
AccessAnalyzerServiceRolePolicy	AWS managed
AdministratorAccess	AWS managed - job function

3. Created IAM Role :



IAM > Roles

Identity and Access Management (IAM)

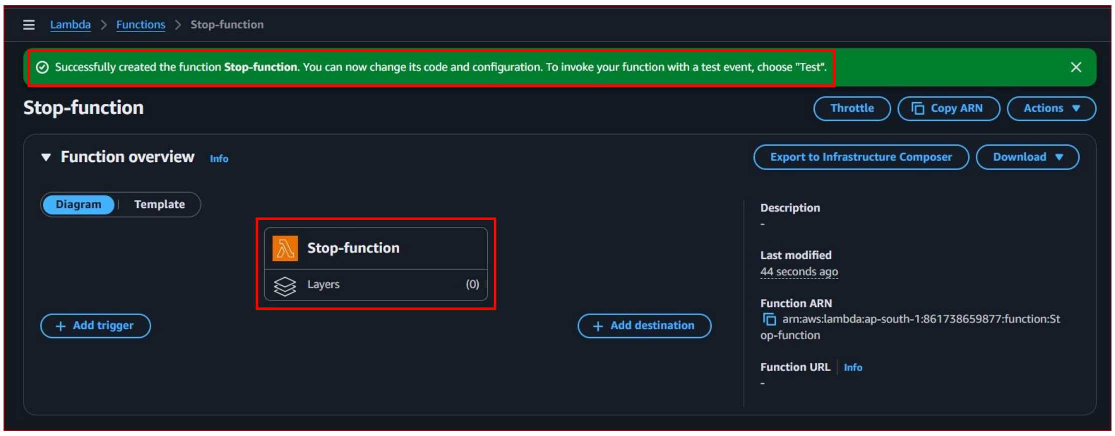
Role Practice-Lambda_role created.

Roles (8) 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
Amazon_EventBridge_Invoke_Sns_1188685556	AWS Service: events	18 hours ago
AWSServiceRoleForAutoScaling	AWS Service: autoscaling (Service-Li	3 days ago
AWSServiceRoleForCloudWatchEvents	AWS Service: events (Service-Linked	2 days ago
AWSServiceRoleForElasticLoadBalancing	AWS Service: elasticloadbalancing (S	Yesterday

4. Created Lambda Function :



Lambda > Functions > Stop-function

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

Stop-function

Function overview

Description
-

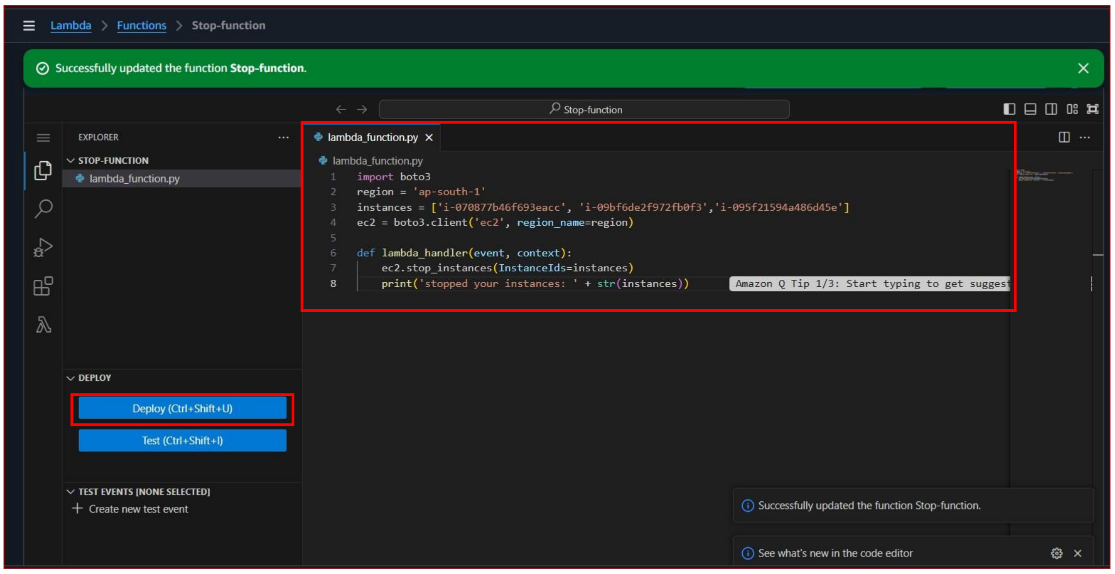
Last modified
44 seconds ago

Function ARN
arn:aws:lambda:ap-south-1:861738659877:function:Stop-function

Function URL
-

5. Write a Python code & Deploy :

- (Write python code including instance ID [instances =['instance_id-1','instance_id-2','instace_id-3']] and Give the name of region ('ap-south-1') and deploy it code)



Lambda > Functions > Stop-function

Successfully updated the function **Stop-function**.

```python
import boto3
region = 'ap-south-1'
instances = ['i-070877b46f693eacc', 'i-09bf6de2f972fb0f3','i-095f21594a486d45e']
ec2 = boto3.client('ec2', region_name=region)

def lambda_handler(event, context):
    ec2.stop_instances(InstanceIds=instances)
    print('stopped your instances: ' + str(instances))
```

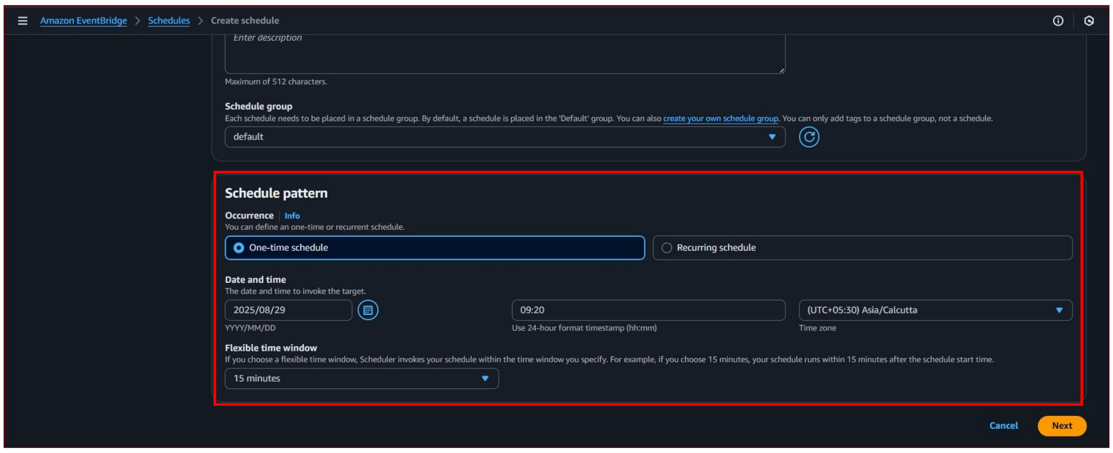
Amazon Q Tip 1/3: Start typing to get suggest

Successfully updated the function Stop-function.

See what's new in the code editor

6.Creating EventBridge (CloudWatch) Rule :

Creating Schedule :



Amazon EventBridge > Schedules > Create schedule

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

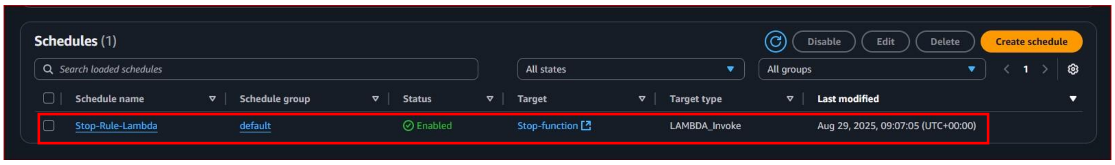
Occurrence | Info
You can define an one-time or recurrent schedule.
[✓ One-time schedule] Recurring schedule

Date and time
The date and time to invoke the target.
2025/08/29 (YYYY/MM/DD)
09:20 (Use 24-hour format timestamp (hh:mm))
(UTC+05:30) Asia/Calcutta (Time zone)

Flexible time window
If you choose a flexible time window, Scheduler invokes your schedule within the time window you specify. For example, if you choose 15 minutes, your schedule runs within 15 minutes after the schedule start time.
15 minutes

Cancel Next

Creted shedule :

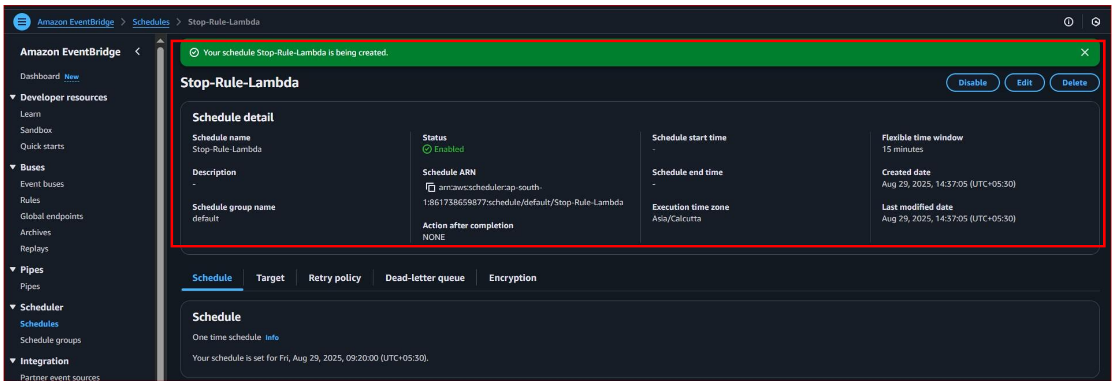


Schedules (1)

Search loaded schedules | All states | All groups

Schedule name	Schedule group	Status	Target	Target type	Last modified
Stop-Rule-Lambda	default	✓ Enabled	Stop-function	LAMBDA_Invoke	Aug 29, 2025, 09:07:05 (UTC+00:00)

Created :



Amazon EventBridge > Schedules > Stop-Rule-Lambda

Your schedule Stop-Rule-Lambda is being created.

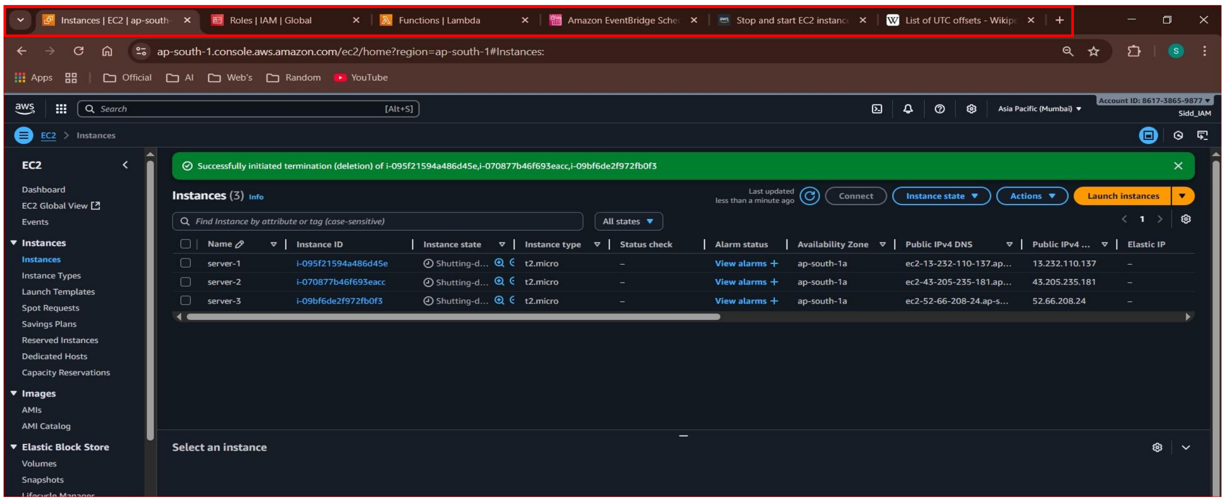
Stop-Rule-Lambda

Schedule detail

Schedule name Stop-Rule-Lambda	**Status** ✓ Enabled	**Schedule start time** -	**Flexible time window** 15 minutes
Description -	**Schedule ARN** arn:aws:scheduler:ap-south-1:861738659877:schedule/default/Stop-Rule-Lambda	**Schedule end time** -	**Created date** Aug 29, 2025, 14:37:05 (UTC+05:30)
Schedule group name default	**Action after completion** NONE	**Execution time zone** Asia/Calcutta	**Last modified date** Aug 29, 2025, 14:37:05 (UTC+05:30)

Schedule
One time schedule Info
Your schedule is set for Fri, Aug 29, 2025, 09:20:00 (UTC+05:30).

Result :

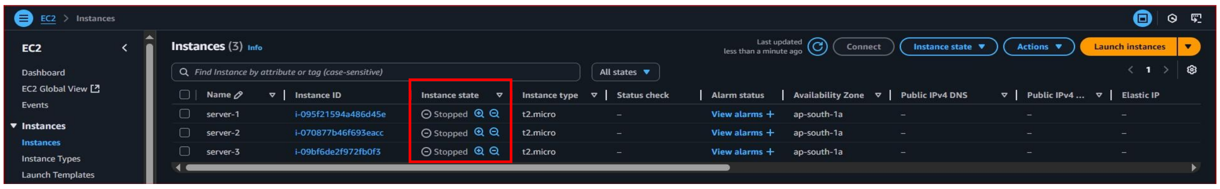


Successfully initiated termination (deletion) of i-095f21594a486d45e,i-070877b46f693eacc,i-09bf6de2f972fb0f3

Instances (3)

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	Public IPv4 ...	Elastic IP
server-1	i-095f21594a486d45e	Shutting-d...	t2.micro	–	View alarms +	ap-south-1a	ec2-13-232-110-137.ap...	13.232.110.137	–
server-2	i-070877b46f693eacc	Shutting-d...	t2.micro	–	View alarms +	ap-south-1a	ec2-43-205-235-181.ap...	43.205.235.181	–
server-3	i-09bf6de2f972fb0f3	Shutting-d...	t2.micro	–	View alarms +	ap-south-1a	ec2-52-66-208-24.ap-s...	52.66.208.24	–

Instances | EC2 | ap-south-
Roles | IAM | Global
Functions | Lambda
Amazon EventBridge Sche
Stop and start EC2 instanc
List of UTC offsets - Wikip
ap-south-1.console.aws.amazon.com/ec2/home?region=ap-south-1#Instances:
Asia Pacific (Mumbai)
Account ID: 8617-3865-9877
Sidd_IAM
Select an instance



Instances (3)

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	Public IPv4 ...	Elastic IP
server-1	i-095f21594a486d45e	Stopped	t2.micro	–	View alarms +	ap-south-1a	–	–	–
server-2	i-070877b46f693eacc	Stopped	t2.micro	–	View alarms +	ap-south-1a	–	–	–
server-3	i-09bf6de2f972fb0f3	Stopped	t2.micro	–	View alarms +	ap-south-1a	–	–	–

By implementing this mini project, I successfully automated the start and stop operations of EC2 instances using AWS services.

-Amazon EventBridge automatically triggers the Lambda function on a predefined schedule.
-AWS Lambda executes the Python code to start or stop the EC2 instances.I
-AM Role ensures that Lambda has the required permissions to perform actions on EC2.
-As a result, EC2 instances are automatically managed (started in the morning and stopped at night, for example) without any manual intervention.

This automation helps in:

-Reducing costs by shutting down unused instances.
-Saving time by removing the need for manual instance management.
-Improving efficiency with a fully serverless, event-driven approach.

Stay Connected :

https://www.linkedin.com/in/siddheshwar-shinde-90b73a353

https://github.com/codexshwar

Thank You .