

# AUTOMATE START-STOP EC2 INSTANCES USING LAMBDA

## **AWS PROJECT**

**Presented by:** Shaik Mohammed Shabaz

# PROJECT OVERVIEW

## Objective:

- To automate the start and stop of Amazon EC2 instances using AWS Lambda functions based on specific triggers or schedules.
- The purpose of this project is to implement a cost-effective automation mechanism using AWS Lambda to **automatically start and stop Amazon EC2 instances** at predefined times, eliminating the need for manual intervention and reducing unnecessary AWS usage charges.
- In many organizations or training environments, EC2 instances are used during working hours and remain idle during off-hours, leading to unnecessary costs. Manually managing their lifecycle is inefficient. This project leverages AWS serverless computing (Lambda) to automate the process based on a defined schedule using CloudWatch Events.

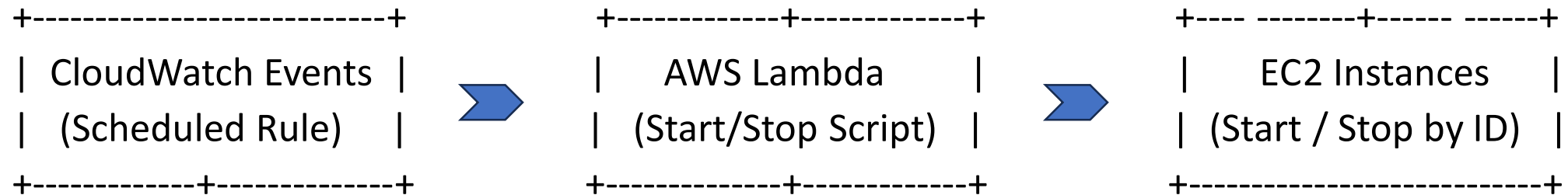
## Goals of the Project:

- Automate the EC2 start and stop process.
- Reduce operational costs by turning off idle resources.
- Eliminate manual efforts and human errors.
- Use serverless (pay-as-you-go) architecture.

# PROJECT ARCHITECTURE

## Components Used :

1. AWS Lambda
2. Amazon EC2
3. IAM Roles, Policies
4. CloudWatch (EventBridge)



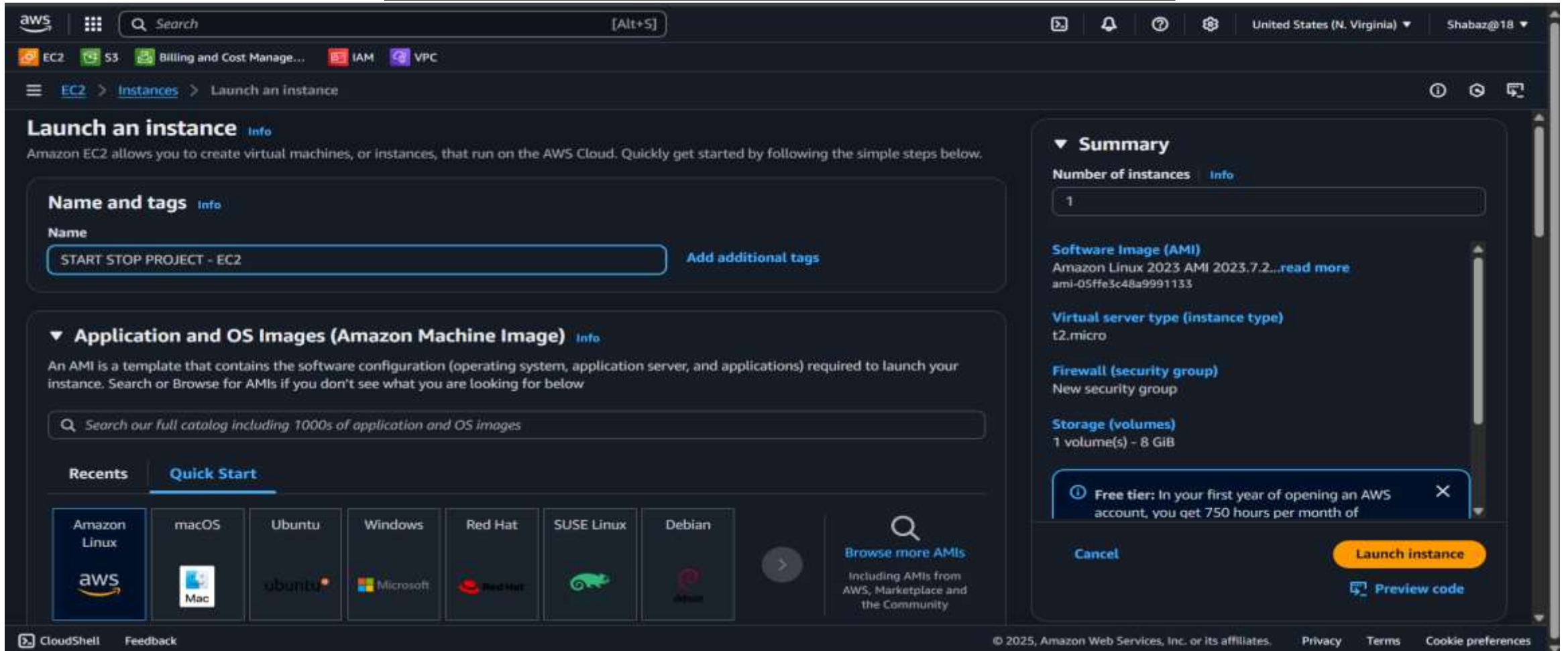
# TOOLS & TECHNOLOGIES USED

- AWS LAMBDA (Serverless Compute)
- AMAZON EC2 (Virtual Machines)
- CLOUDWATCH EVENTS / EVENTBRIDGE (Scheduling)
- IAM (For Permissions)
- PYTHON (Boto3 SDK)

# PROJECT WORKFLOW

- Created EC2 Instance
- Created IAM Role
- Created IAM Policies
- Created Lambda Functions
- Configured CloudWatch Events
- Used Python (Boto3) Code
- Triggered Lambda via CloudWatch Scheduler

# IMPLEMENTATION



>>> LAUNCH AN EC2 INSTANCE

>>> With the name of – Start Stop Project – EC2 (whatever you want)

- With these EC2 Details

The screenshot displays the AWS Management Console interface for an EC2 instance. The left sidebar shows the navigation menu with categories like EC2, Images, and Elastic Block Store. The main content area is titled 'Instance summary for i-005c11e63f93ab74e (START-STOP PROJECT - EC2)' and includes a 'Connect' button, 'Instance state' dropdown, and 'Actions' dropdown. The instance is in a 'Running' state. The summary is organized into four columns: Instance ID, Public IPv4 address, Private IPv4 addresses, and Public DNS. Other details include Hostname type, Answer private resource DNS name, Auto-assigned IP address, IAM Role, IMDSv2, Instance type, VPC ID, Subnet ID, Instance ARN, Elastic IP addresses, AWS Compute Optimizer finding, Auto Scaling Group name, and Managed status.

| Instance ID         | Public IPv4 address                         | Private IPv4 addresses | Public DNS  |
|---------------------|---|------------------------|---|
| i-005c11e63f93ab74e | 54.92.182.58   <a href="#">open address</a> | 172.31.88.129          | ec2-54-92-182-58.compute-1.amazonaws.com   <a href="#">open address</a> |

| Hostname type                          | Instance state | Private IP DNS name (IPv4 only) | Elastic IP addresses |
|--|----------------|---------------------------------|----------------------|
| IP name: ip-172-31-88-129.ec2.internal | Running        | ip-172-31-88-129.ec2.internal   | -                    |

| Answer private resource DNS name | Instance type | AWS Compute Optimizer finding   | Auto Scaling Group name |
|----------------------------------|---------------|---|-------------------------|
| IPv4 (A)                         | t2.micro      | Opt-in to AWS Compute Optimizer for recommendation   <a href="#">Learn more</a> | -                       |

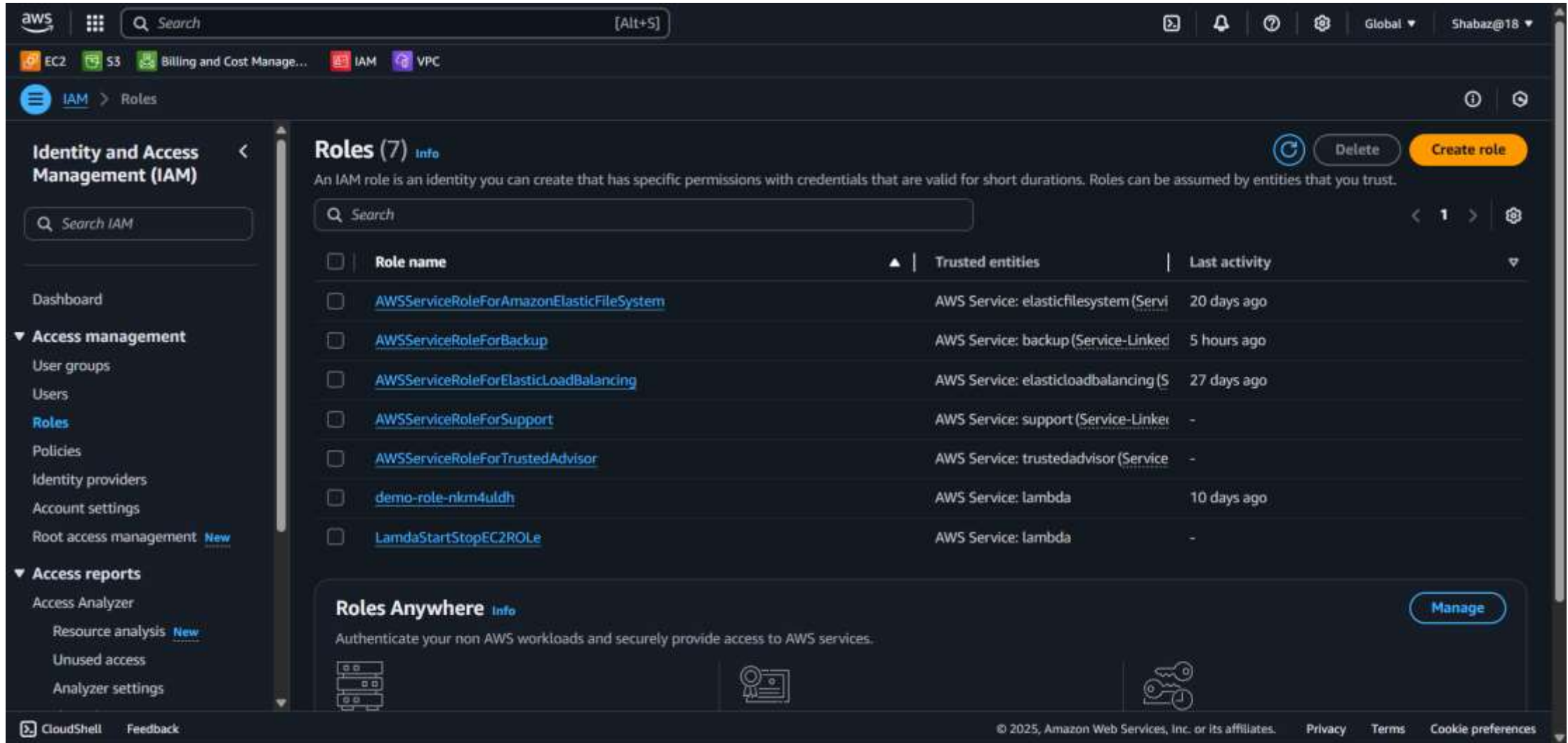
| Auto-assigned IP address | VPC ID                | Instance ARN  | Managed |
|--------------------------|-----------------------|---|---------|
| 54.92.182.58 [Public IP] | vpc-0e315e5772206ab9c | arn:aws:ec2:us-east-1:722599784033:instance/i-005c11e63f93ab74e | false   |

| IAM Role | Subnet ID                |
|----------|--------------------------|
| -        | subnet-04576fc9526a125b4 |

| IMDSv2   |
|----------|
| Required |

# >>> CREATE AN IAM ROLE for LAMBDA

1. Here remember one thing - your Lambda Functions need permission to manage EC2 instances
2. This is why we attach EC2-related policies to the IAM Role assigned to the Lambda.”



The screenshot displays the AWS IAM console interface. The left sidebar shows the navigation menu with 'Identity and Access Management (IAM)' selected. The main content area is titled 'Roles (7)' and includes a search bar and a table of existing roles. The table has columns for 'Role name', 'Trusted entities', and 'Last activity'. The roles listed are:

| Role name  | Trusted entities                                   | Last activity |
|--|--|---------------|
| <a href="#">AWSServiceRoleForAmazonElasticFileSystem</a> | AWS Service: elasticfilesystem (Service-Linked)    | 20 days ago   |
| <a href="#">AWSServiceRoleForBackup</a>                  | AWS Service: backup (Service-Linked)               | 5 hours ago   |
| <a href="#">AWSServiceRoleForElasticLoadBalancing</a>    | AWS Service: elasticloadbalancing (Service-Linked) | 27 days ago   |
| <a href="#">AWSServiceRoleForSupport</a>                 | AWS Service: support (Service-Linked)              | -             |
| <a href="#">AWSServiceRoleForTrustedAdvisor</a>          | AWS Service: trustedadvisor (Service-Linked)       | -             |
| <a href="#">demo-role-nkm4uldh</a>                       | AWS Service: lambda                                | 10 days ago   |
| <a href="#">LambdaStartStopEC2ROLE</a>                   | AWS Service: lambda                                | -             |

At the bottom of the console, there is a 'Roles Anywhere' section with a 'Manage' button. The footer of the console shows the copyright notice: '© 2025, Amazon Web Services, Inc. or its affiliates.' and links for 'Privacy', 'Terms', and 'Cookie preferences'.



- Process to create IAM ROLE --- >>> STEP : 1
  1. Search for IAM in the search bar , Open it
  2. Click on “ROLES” , Then Click on Create role

The screenshot shows the AWS IAM console interface for creating a new role. The top navigation bar includes the AWS logo, a search bar, and user information (Shabaz@18). The breadcrumb trail indicates the path: IAM > Roles > Create role. A progress indicator on the left shows three steps: Step 1 (Select trusted entity, active), Step 2 (Add permissions), and Step 3 (Name, review, and create). The main content area is titled 'Select trusted entity' and contains two sections: 'Trusted entity type' and 'Use case'. The 'Trusted entity type' section has five radio button options: 'AWS service' (selected), 'AWS account', 'Web identity', 'SAML 2.0 federation', and 'Custom trust policy'. Each option has a brief description. The 'Use case' section has a heading and a description, followed by a dropdown menu labeled 'Service or use case' with the placeholder text 'Choose a service or use case'. At the bottom right, there are 'Cancel' and 'Next' buttons. The footer contains 'CloudShell', 'Feedback', and copyright information for Amazon Web Services, Inc. or its affiliates, along with links for 'Privacy', 'Terms', and 'Cookie preferences'.

aws | Search [Alt+S] | Global | Shabaz@18

EC2 S3 Billing and Cost Manage... IAM VPC

IAM > Roles > Create role

Step 1  
● Select trusted entity

Step 2  
○ Add permissions

Step 3  
○ Name, review, and create

### Select trusted entity info

#### Trusted entity type

- ☒ **AWS service**  
Allow AWS services like EC2, Lambda, or others to perform actions in this account.
- ☐ **AWS account**  
Allow entities in other AWS accounts belonging to you or a 3rd party to perform actions in this account.
- ☐ **Web identity**  
Allows users federated by the specified external web identity provider to assume this role to perform actions in this account.
- ☐ **SAML 2.0 federation**  
Allow users federated with SAML 2.0 from a corporate directory to perform actions in this account.
- ☐ **Custom trust policy**  
Create a custom trust policy to enable others to perform actions in this account.

#### Use case

Allow an AWS service like EC2, Lambda, or others to perform actions in this account.

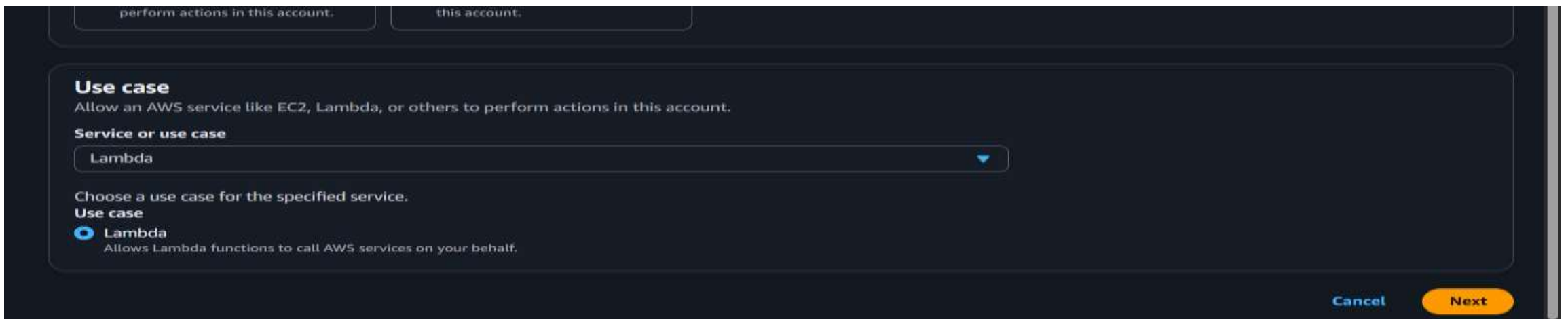
**Service or use case**

Choose a service or use case ▼

Cancel Next

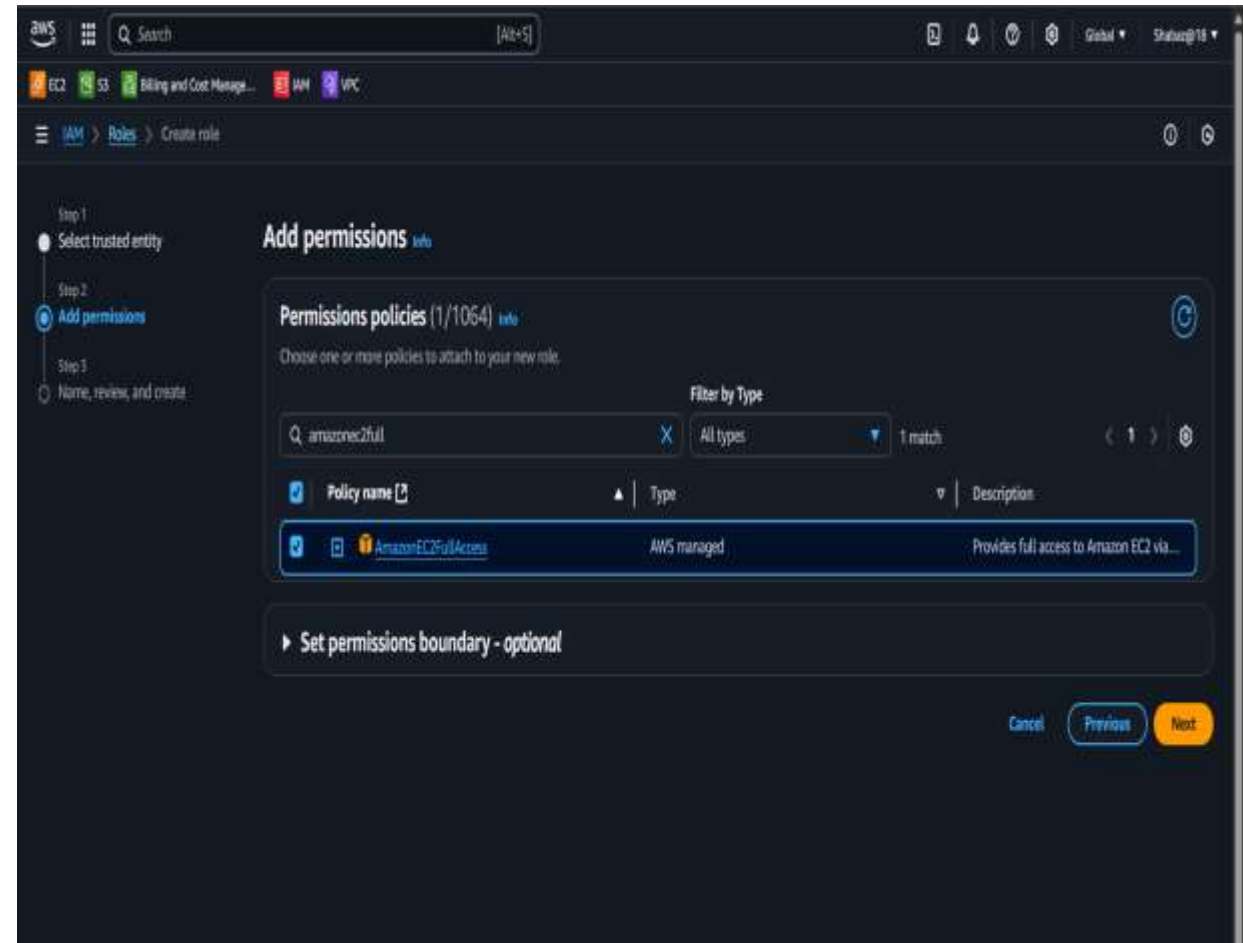
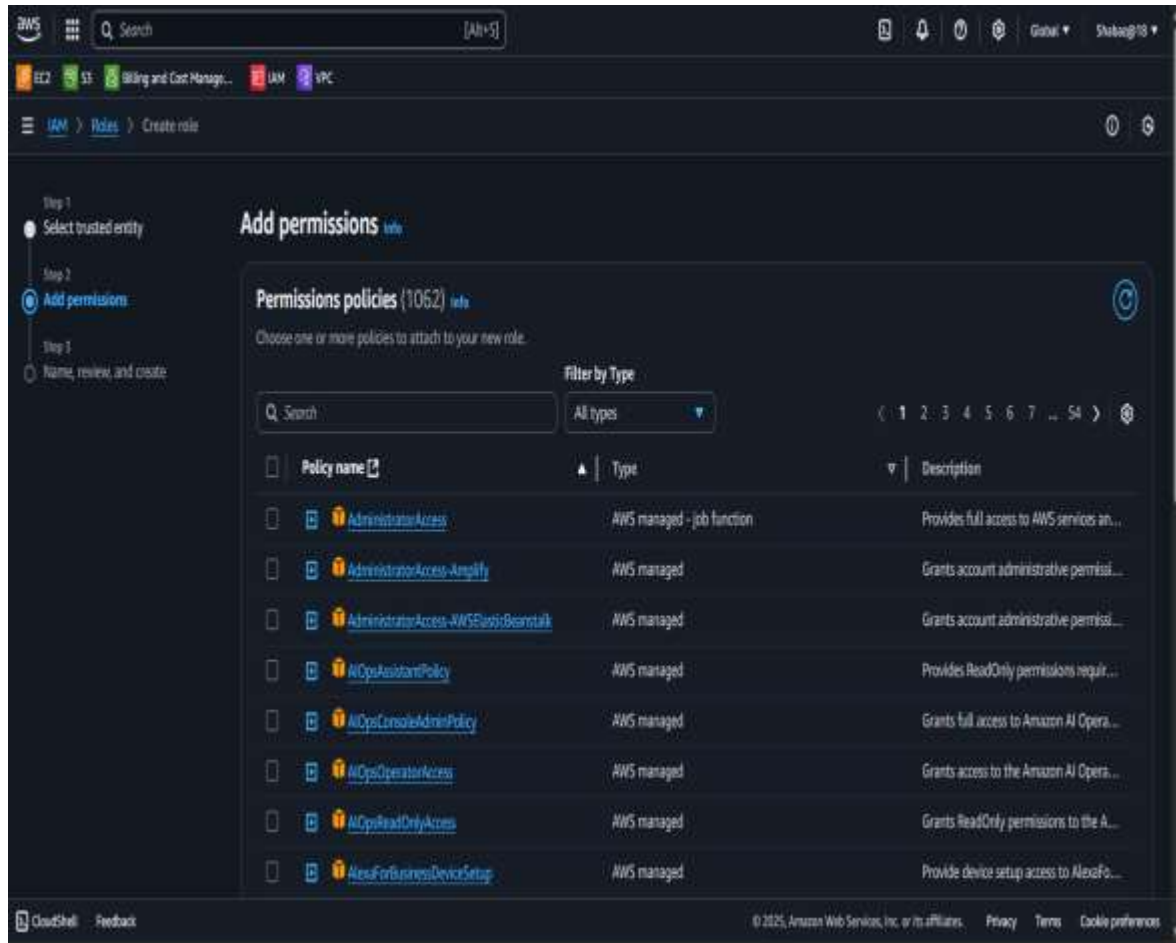
CloudShell Feedback | © 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

- Select AWS Service at Trusted Entity Type
- At USE CASE, search for LAMBDA & Select it & Click on Next



## >>> STEP : 2

- Now we have to add the AmazonEC2FullAccess Permission at ADD PERMISSION section
- Search for AmazonEC2FullAccess & select it, Scroll down & click on next



>>> STEP : 3

- Give ROLE NAME like : LambdaEC2startstopROLE (whatever you want)

The screenshot shows the AWS IAM console interface for creating a new role. The top navigation bar includes the AWS logo, a search bar, and user information (Shabaz@18). The main navigation bar shows links for EC2, S3, Billing and Cost Management, IAM, and VPC. The breadcrumb trail indicates the path: IAM > Roles > Create role. On the left, a sidebar shows the three steps of the process: Step 1 (Select trusted entity), Step 2 (Add permissions), and Step 3 (Name, review, and create), with Step 3 being the active step. The main content area is titled 'Name, review, and create' and contains two sections: 'Role details' and 'Step 1: Select trusted entities'. In the 'Role details' section, the 'Role name' field is filled with 'LambdaEC2StartStopRole' and has a description of 'Allows Lambda functions to call AWS services on your behalf.' The 'Step 1: Select trusted entities' section shows a 'Trust policy' field with a JSON snippet. The footer includes links for CloudShell, Feedback, and copyright information for Amazon Web Services.

aws | Search [Alt+S] | Global | Shabaz@18

EC2 S3 Billing and Cost Manage... IAM VPC

IAM > Roles > Create role

Step 1  
● Select trusted entity

Step 2  
● Add permissions

Step 3  
● **Name, review, and create**

### Name, review, and create

#### Role details

**Role name**  
Enter a meaningful name to identify this role.

LambdaEC2StartStopRole

Maximum 64 characters. Use alphanumeric and '+=, @-\_' characters.

**Description**  
Add a short explanation for this role.

Allows Lambda functions to call AWS services on your behalf.

Maximum 1000 characters. Use letters (A-Z and a-z), numbers (0-9), tabs, new lines, or any of the following characters: \_+=, @-/\[\]!#\$%^&\*()~`'.

#### Step 1: Select trusted entities

**Trust policy**

```
1 {  
2   "Version": "2012-10-17",  
3   "Statement": [  
4     {  
5       "Effect": "Allow",  
6       "Principal": {  
7         "Service": "lambda.amazonaws.com",  
8         "Action": "lambda:InvokeFunction",  
9       }  
10    ]  
11  }
```

CloudShell Feedback

© 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

- Scroll Down and Click on CREATE ROLE

The screenshot displays the AWS IAM console interface for creating a new role. The top navigation bar includes the AWS logo, a search bar, and user information. The breadcrumb trail indicates the path: IAM > Roles > Create role. A code editor at the top shows a JSON snippet for a role definition.

**Step 2: Add permissions** Edit

**Permissions policy summary**

| Policy name <a href="#">[?]</a>     | Type        | Attached as        |
|-------------------------------------|-------------|--------------------|
| <a href="#">AmazonEC2FullAccess</a> | AWS managed | Permissions policy |

**Step 3: Add tags**

**Add tags - optional** [Info](#)

Tags are key-value pairs that you can add to AWS resources to help identify, organize, or search for resources.

No tags associated with the resource.

[Add new tag](#)

You can add up to 50 more tags.

Cancel Previous Create role

CloudShell Feedback © 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

- You can see your created IAM ROLE by clicking on Roles and click on your customized name to see in detail summary

The screenshot displays the AWS IAM console interface. The left sidebar shows the 'Identity and Access Management (IAM)' menu with options like Dashboard, Access management, Roles, Policies, and Access reports. The main content area shows the details for the role 'LamdaStartStopEC2ROLE'. The role's summary includes its creation date (July 10, 2025, 16:36 UTC+05:30), last activity, ARN (arn:aws:iam::722599784033:role/LamdaStartStopEC2ROLE), and maximum session duration (1 hour). Below the summary, there are tabs for Permissions, Trust relationships, Tags, Last Accessed, and Revoke sessions. The 'Permissions' tab is active, showing a list of permissions policies. One policy, 'AmazonEC2FullAccess', is listed with the type 'AWS managed' and is attached to 1 entity. The interface includes a search bar at the top, a navigation menu on the left, and a footer with copyright information and links to Privacy, Terms, and Cookie preferences.

**Identity and Access Management (IAM)**

**LamdaStartStopEC2ROLE** [Info](#) [Delete](#)

Allows Lambda functions to call AWS services on your behalf.

**Summary** [Edit](#)

**Creation date**  
July 10, 2025, 16:36 (UTC+05:30)

**Last activity**  
-

**ARN**  
[arn:aws:iam::722599784033:role/LamdaStartStopEC2ROLE](#)

**Maximum session duration**  
1 hour

**Permissions** | Trust relationships | Tags | Last Accessed | Revoke sessions

**Permissions policies (1)** [Info](#) [Simulate](#) [Remove](#) [Add permissions](#)

You can attach up to 10 managed policies.

**Filter by Type**  
All types

| Policy name                         | Type        | Attached entities |
|-------------------------------------|-------------|-------------------|
| <a href="#">AmazonEC2FullAccess</a> | AWS managed | 1                 |

© 2025, Amazon Web Services, Inc. or its affiliates. [Privacy](#) [Terms](#) [Cookie preferences](#)



## >>> Next , CREATE THE START LAMBDA FUNCTION

- Search for LAMBDA in the search bar, click on create Function
- Give name as StartEC2Function & Select RUNTIME : PYTHON 3.12

The screenshot displays the AWS Lambda 'Create function' interface. At the top, the AWS logo and navigation bar are visible. The main heading is 'Create function' with an 'Info' link. Below this, a prompt says 'Choose one of the following options to create your function.' Three options are presented: 'Author from scratch' (selected with a blue circle), 'Use a blueprint', and 'Container image'. The 'Author from scratch' option includes the text 'Start with a simple Hello World example.' Below these options is the 'Basic information' section. It contains a 'Function name' field with the value 'StartEC2Instances' and a description: 'Enter a name that describes the purpose of your function.' Below the field, a note states: 'Function name must be 1 to 64 characters, must be unique to the Region, and can't include spaces. Valid characters are a-z, A-Z, 0-9, hyphens (-), and underscores (\_).' The 'Runtime' section shows a dropdown menu with 'Python 3.12' selected and a refresh icon. The 'Architecture' section has two radio buttons: 'arm64' and 'x86\_64', with 'x86\_64' selected. The 'Permissions' section is partially visible at the bottom. On the right side, there is a sidebar with 'Info' and 'Tutorials' tabs. The 'Tutorials' tab is active, showing a section titled 'Create a simple web app' with a list of steps: 'Build a simple web app, consisting of a Lambda function with a function URL that outputs a webpage' and 'Invoke your function through its function URL'. A 'Start tutorial' button is at the bottom of this section. The footer of the page includes 'CloudShell', 'Feedback', and copyright information for Amazon Web Services, Inc. or its affiliates, along with links for 'Privacy', 'Terms', and 'Cookie preferences'.

- Scroll Down and come to EXECUTIVE ROLE
- Here Choose – Use an Existing role
- Next, Select your Created IAM ROLE & click on Create Function

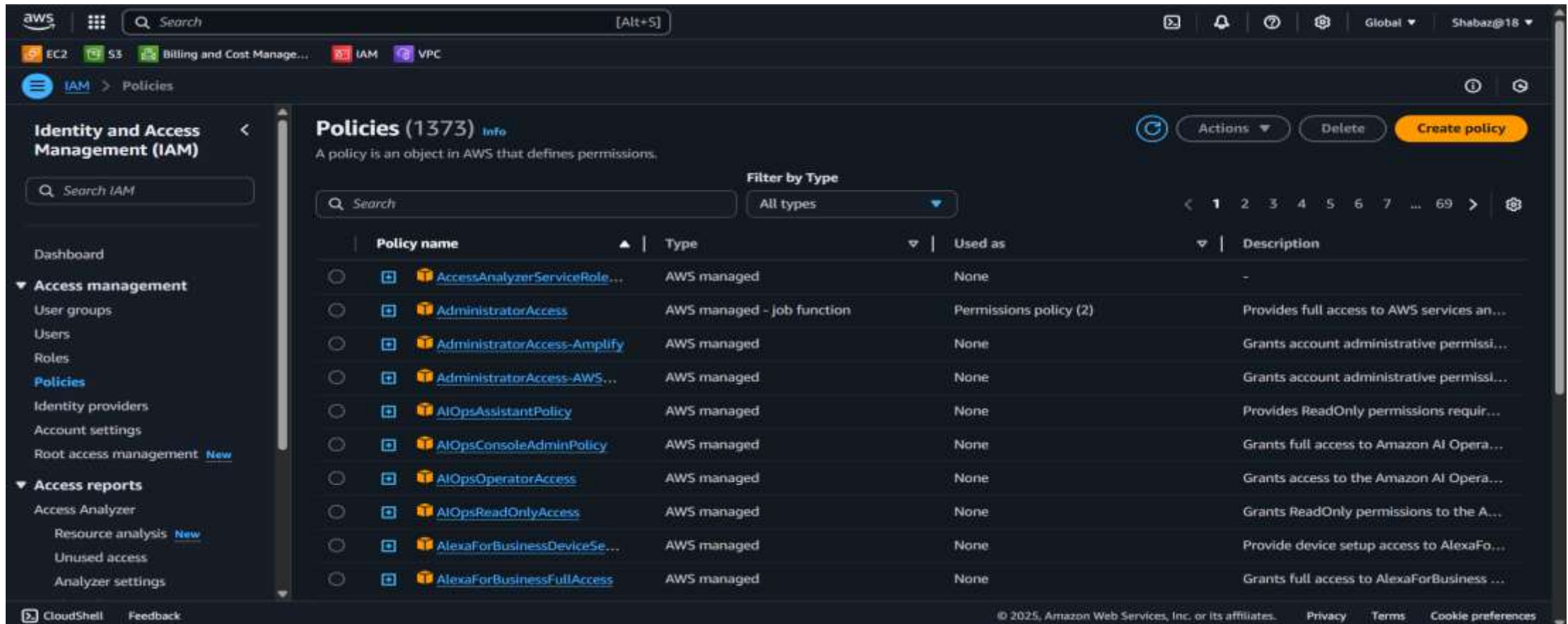
The screenshot shows the AWS Lambda console interface for creating a new function. The breadcrumb navigation indicates the path: Lambda > Functions > Create function. The architecture is set to x86\_64. In the 'Permissions' section, the 'Change default execution role' dropdown is open, showing three options: 'Create a new role with basic Lambda permissions', 'Use an existing role' (which is selected), and 'Create a new role from AWS policy templates'. Below this, the 'Existing role' dropdown is open, displaying a search bar and a list of roles. The role 'LamdaStartStopEC2ROLE' is selected and highlighted with a checkmark. At the bottom right, the 'Create function' button is prominent in orange, next to a 'Cancel' button. On the right sidebar, there is a 'Tutorials' section with a link to 'Create a simple web app' and a 'Start tutorial' button.



NEXT ,

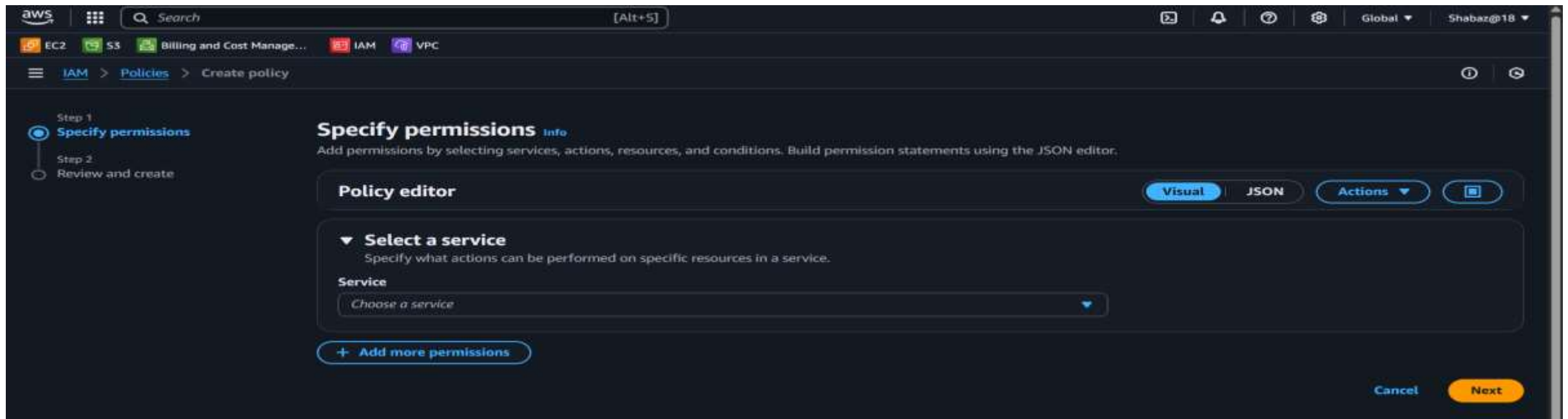
## >>> CREATE POLICIES

1. Go to Search bar & search for IAM
2. Select Policies and Click on Create Policies. Here, at Step 1 : You have to Specify the Permission



The screenshot displays the AWS IAM console interface. The left-hand navigation pane is open, showing the 'Policies' link under the 'Identity and Access Management (IAM)' section. The main content area is titled 'Policies (1373)' and includes a search bar and a 'Filter by Type' dropdown set to 'All types'. A table lists various AWS managed policies, including 'AccessAnalyzerServiceRole...', 'AdministratorAccess', 'AdministratorAccess-Amplify', 'AdministratorAccess-AWS...', 'AIOpsAssistantPolicy', 'AIOpsConsoleAdminPolicy', 'AIOpsOperatorAccess', 'AIOpsReadOnlyAccess', 'AlexaForBusinessDeviceSe...', and 'AlexaForBusinessFullAccess'. Each row shows the policy name, type, whether it is used as a permissions policy, and a brief description. At the top right of the main area, there are buttons for 'Actions', 'Delete', and a prominent orange 'Create policy' button.

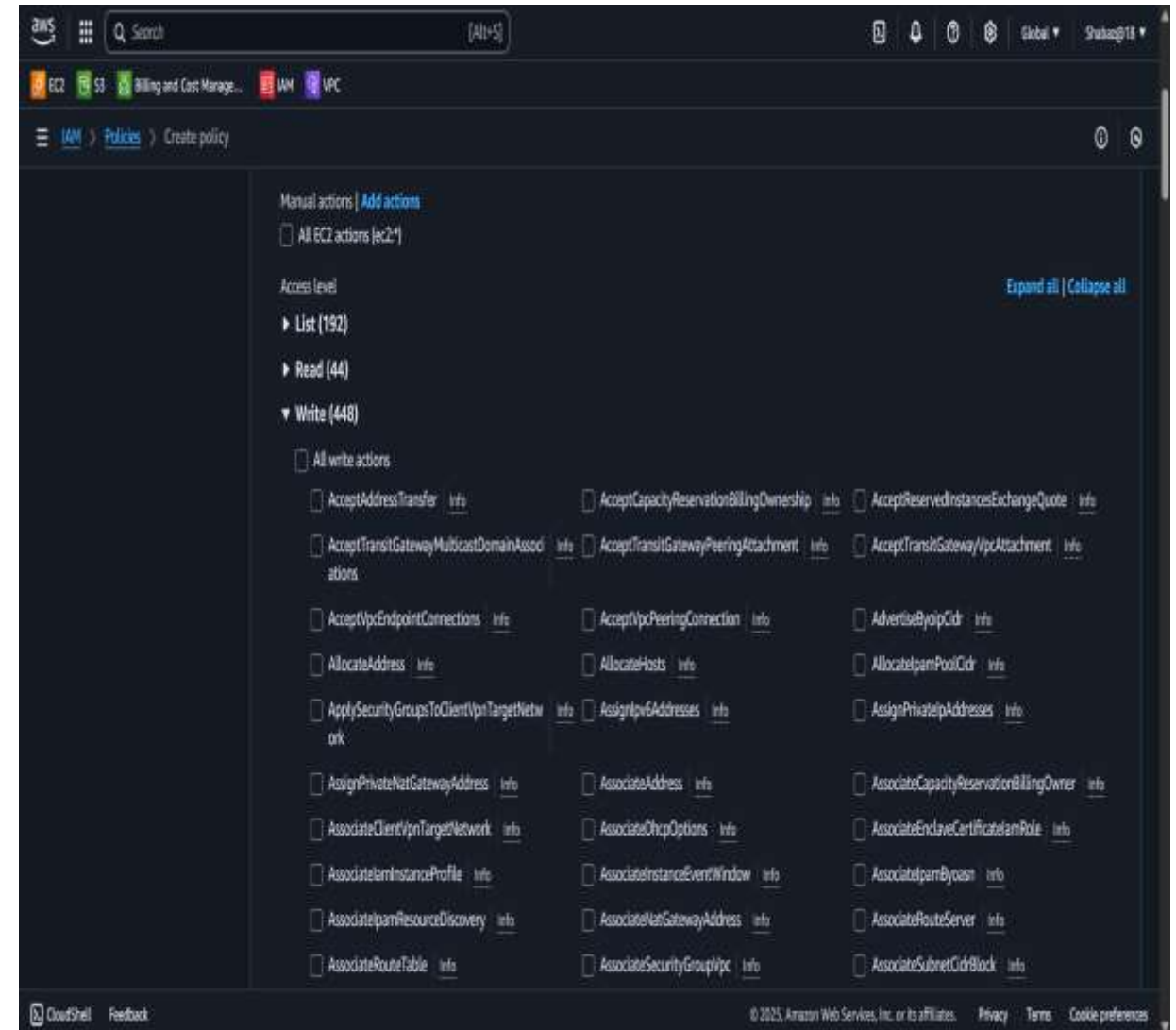
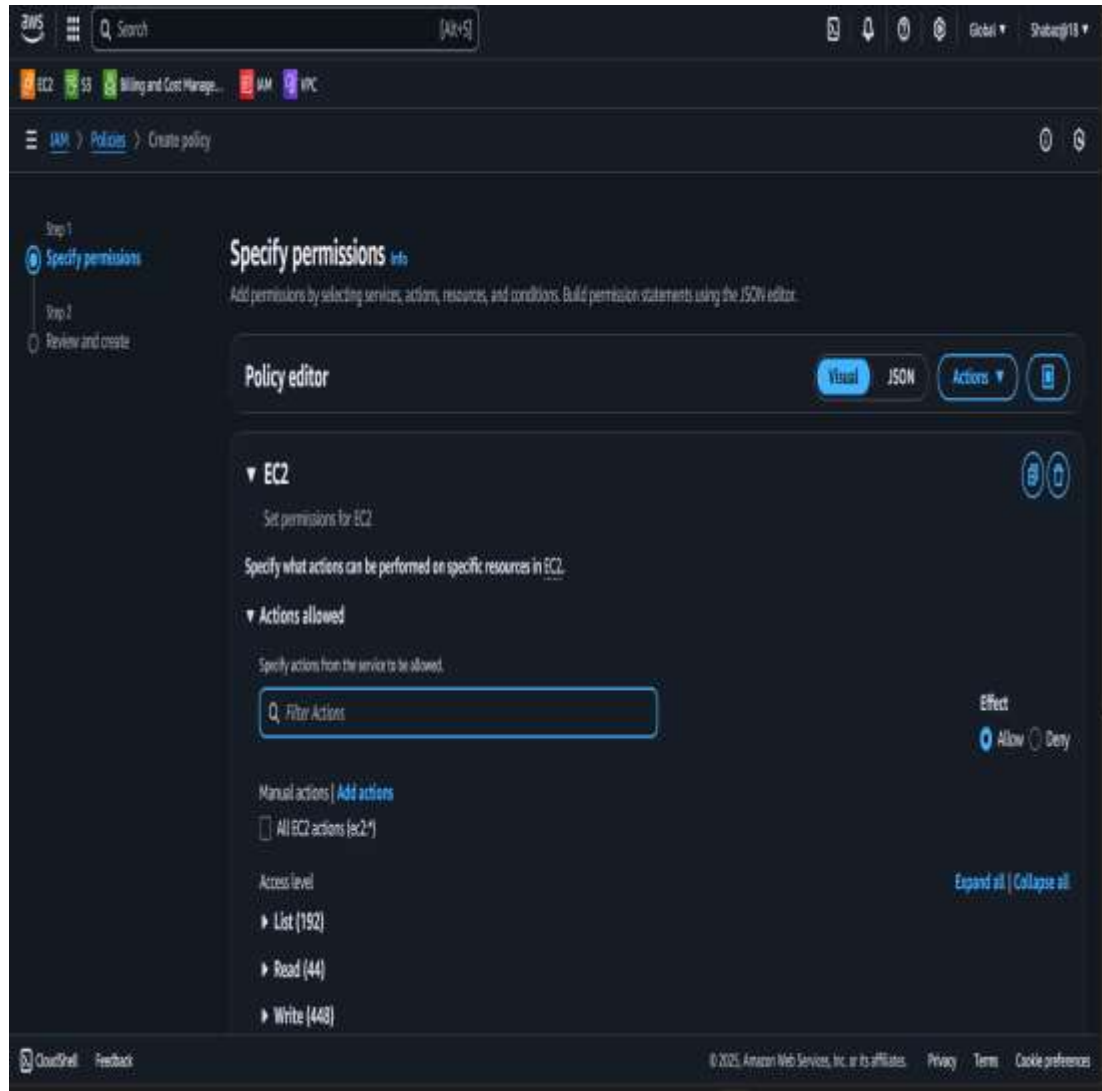
| Policy name                                  | Type                       | Used as                | Description                                |
|--|----------------------------|------------------------|--|
| <a href="#">AccessAnalyzerServiceRole...</a> | AWS managed                | None                   | -  |
| <a href="#">AdministratorAccess</a>          | AWS managed - job function | Permissions policy (2) | Provides full access to AWS services an... |
| <a href="#">AdministratorAccess-Amplify</a>  | AWS managed                | None                   | Grants account administrative permissi...  |
| <a href="#">AdministratorAccess-AWS...</a>   | AWS managed                | None                   | Grants account administrative permissi...  |
| <a href="#">AIOpsAssistantPolicy</a>         | AWS managed                | None                   | Provides ReadOnly permissions requir...    |
| <a href="#">AIOpsConsoleAdminPolicy</a>      | AWS managed                | None                   | Grants full access to Amazon AI Opera...   |
| <a href="#">AIOpsOperatorAccess</a>          | AWS managed                | None                   | Grants access to the Amazon AI Opera...    |
| <a href="#">AIOpsReadOnlyAccess</a>          | AWS managed                | None                   | Grants ReadOnly permissions to the A...    |
| <a href="#">AlexaForBusinessDeviceSe...</a>  | AWS managed                | None                   | Provide device setup access to AlexaFo...  |
| <a href="#">AlexaForBusinessFullAccess</a>   | AWS managed                | None                   | Grants full access to AlexaForBusiness ... |



- Select EC2



- After Selecting EC2 & Click on WRITE -- It will expand



- Here Select StartInstance & Stopinstance

The screenshot shows the AWS IAM console interface for creating a new policy. The breadcrumb navigation indicates the path: IAM > Policies > Create policy. The 'Permissions management' section is expanded, displaying a grid of actions. The actions 'StartInstances' and 'StopInstances' are selected, indicated by blue checkmarks in their respective checkboxes. Other actions listed include 'ResetNetworkInterfaceAttribute', 'RestoreManagedPrefixListVersion', 'RevokeClientVpnIngress', 'RunInstances', 'SendSpotInstanceInterruptions', 'StartNetworkInsightsAnalysis', 'TerminateClientVpnConnections', 'UnassignPrivateIpAddresses', 'UnmonitorInstances', 'WithdrawByoipCidr', 'RestoreAddressToClassic', 'RestoreSnapshotFromRecycleBin', 'RevokeSecurityGroupEgress', 'RunScheduledInstances', 'StartVpcEndpointServicePrivateDnsVerification', 'TerminateInstances', 'UnassignPrivateNatGatewayAddress', 'UpdateSecurityGroupRuleDescriptionsEgress', 'RestoreImageFromRecycleBin', 'RestoreSnapshotTier', 'RevokeSecurityGroupIngress', 'SendDiagnosticInterrupt', 'StartNetworkInsightsAccessScopeAnalysis', 'UnassignIpv6Addresses', 'UnlockSnapshot', and 'UpdateSecurityGroupRuleDescriptionsIngress'. Below the actions list, there are sections for 'Permissions management (16)', 'Tagging (2)', and 'Resources'. The 'Resources' section is currently collapsed, showing a prompt to 'Specify resource ARNs for these actions.' with radio buttons for 'All' and 'Specific'.

aws | Search [Alt+S] | Global | Shabaz@18

EC2 S3 Billing and Cost Manage... IAM VPC

IAM > Policies > Create policy

☐ ResetNetworkInterfaceAttribute [Info](#)

☐ RestoreManagedPrefixListVersion [Info](#)

☐ RevokeClientVpnIngress [Info](#)

☐ RunInstances [Info](#)

☐ SendSpotInstanceInterruptions [Info](#)

☐ StartNetworkInsightsAnalysis [Info](#)

☐ TerminateClientVpnConnections [Info](#)

☐ UnassignPrivateIpAddresses [Info](#)

☐ UnmonitorInstances [Info](#)

☐ WithdrawByoipCidr [Info](#)

☐ RestoreAddressToClassic [Info](#)

☐ RestoreSnapshotFromRecycleBin [Info](#)

☐ RevokeSecurityGroupEgress [Info](#)

☐ RunScheduledInstances [Info](#)

☒ StartInstances [Info](#)

☐ StartVpcEndpointServicePrivateDnsVerification [Info](#)

☐ TerminateInstances [Info](#)

☐ UnassignPrivateNatGatewayAddress [Info](#)

☐ UpdateSecurityGroupRuleDescriptionsEgress [Info](#)

☐ RestoreImageFromRecycleBin [Info](#)

☐ RestoreSnapshotTier [Info](#)

☐ RevokeSecurityGroupIngress [Info](#)

☐ SendDiagnosticInterrupt [Info](#)

☐ StartNetworkInsightsAccessScopeAnalysis [Info](#)

☒ StopInstances [Info](#)

☐ UnassignIpv6Addresses [Info](#)

☐ UnlockSnapshot [Info](#)

☐ UpdateSecurityGroupRuleDescriptionsIngress [Info](#)

► Permissions management (16)

► Tagging (2)

▼ Resources

Specify resource ARNs for these actions.

☐ All

☒ Specific

CloudShell Feedback

© 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

- Scroll Down & click on Add ARN, Give tick mark for both Resources and Click on Add ARN , Do the same thing to second ARN also & Then click on next

The screenshot shows the AWS IAM console interface. The breadcrumb navigation indicates the path: IAM > Policies > Create policy. The 'Specify ARNs' dialog box is open, showing the 'Visual' tab. It includes options for 'Resource in' (This account, Any account, Other account), 'Resource region' (Any region), 'Resource instance' (Any instance), and a 'Resource ARN' field containing 'arn:aws:ec2:\*:722599784033:instance/\*'. The 'Add ARNs' button is highlighted in orange. In the background, the 'Request conditions - optional' section is visible. On the right side of the console, the 'Amazon Q' sidebar is open, displaying the 'Free Tier' badge and a list of sources for job processing services.

aws [Search] [Alt+S] Global ▾ Shabaz@18 ▾

EC2 S3 Billing and Cost Manage... IAM VPC

IAM > Policies > Create policy

UnassignPrivateIPAddresses UnassignPrivateIPAddresses UnassignPrivateIPAddresses UnassignPrivateIPAddresses

**Specify ARNs** ✕

Visual Text

Resource in  
☒ This account ☐ Any account ☐ Other account

Resource region  
\* ☒ Any region

Resource instance  
\* ☒ Any instance

Resource ARN  
arn:aws:ec2:\*:722599784033:instance/\*

Cancel Add ARNs

► Request conditions - optional  
Actions on resources are allowed or denied only when these conditions are met.

Amazon Q Free Tier

How to Add Mult... +

Lambda, or other job processing services.

▼ Sources

Running jobs from the AWS CLI - Amazon EMR

Use ListJobs with an AWS SDK - AWS SDK Code Examples

Ask me anything about AWS

Max 1000 characters

Amazon Q Developer uses generative AI. You may need to verify responses. See the AWS Responsible AI Policy.

CloudShell Feedback

© 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences



- Here, at Step 2 : you have to give Policy name as EC2-StartStop & click on Create Policy

The screenshot shows the AWS IAM console interface during the 'Review and create' step of policy creation. The left sidebar indicates 'Step 2: Review and create' is the current step. The main content area is divided into three sections: 'Policy details', 'Permissions defined in this policy', and 'Add tags - optional'. In the 'Policy details' section, the 'Policy name' field is highlighted with a red box and contains the text 'EC2-StartStop'. Below it, there is a text area for a description. The 'Permissions defined in this policy' section shows a table with one entry for 'EC2' with 'Limited: Write' access level and 'Multiple' resources. The 'Add tags - optional' section shows no tags are currently associated with the resource.

**Policy details**

**Policy name**  
Enter a meaningful name to identify this policy.

EC2-StartStop

Maximum 128 characters. Use alphanumeric and '+=, @-\_' characters.

**Description - optional**  
Add a short explanation for this policy.

Maximum 1,000 characters. Use alphanumeric and '+=, @-\_' characters.

**Permissions defined in this policy**

Permissions defined in this policy document specify which actions are allowed or denied. To define permissions for an IAM identity (user, user group, or role), attach a policy to it

Search

Allow (1 of 445 services) Show remaining 444 services

| Service | Access level   | Resource | Request condition |
|---------|----------------|----------|-------------------|
| EC2     | Limited: Write | Multiple | None              |

**Add tags - optional**

Tags are key-value pairs that you can add to AWS resources to help identify, organize, or search for resources.

No tags associated with the resource.

Add new tag

You can add up to 50 more tags.

Cancel Previous Create policy

- Here, At (IAM > Policies) you can see, your Created Policies by clicking on the created Policy name

The screenshot displays the AWS IAM console interface. The left sidebar shows the navigation menu with 'IAM > Policies > EC2-StartStop' selected. The main content area shows the 'EC2-StartStop' policy details, including its type (Customer managed), creation and edited times (July 10, 2025, 17:05 UTC+05:30), and its ARN (arn:aws:iam:722599784033:policy/EC2-StartStop). Below the details, the 'Permissions' tab is active, showing a table of permissions defined in the policy. The table has columns for Service, Access level, Resource, and Request condition. One permission is listed: EC2 with Limited: Write access level, InstanceID string like [All, region] string like [All] resource, and None request condition. The bottom of the page shows the footer with '© 2025, Amazon Web Services, Inc. or its affiliates.' and links for Privacy, Terms, and Cookie preferences.

**EC2-StartStop** [Info](#) [Edit](#) [Delete](#)

**Policy details**

|                  |                                  |                                  |   |
|------------------|----------------------------------|----------------------------------|---|
| Type             | Creation time                    | Edited time                      | ARN   |
| Customer managed | July 10, 2025, 17:05 (UTC+05:30) | July 10, 2025, 17:05 (UTC+05:30) | arn:aws:iam:722599784033:policy/EC2-StartStop |

[Permissions](#) [Entities attached](#) [Tags](#) [Policy versions \(1\)](#) [Last Accessed](#)

**Permissions defined in this policy** [Info](#) [Edit](#) [Summary](#) [JSON](#)

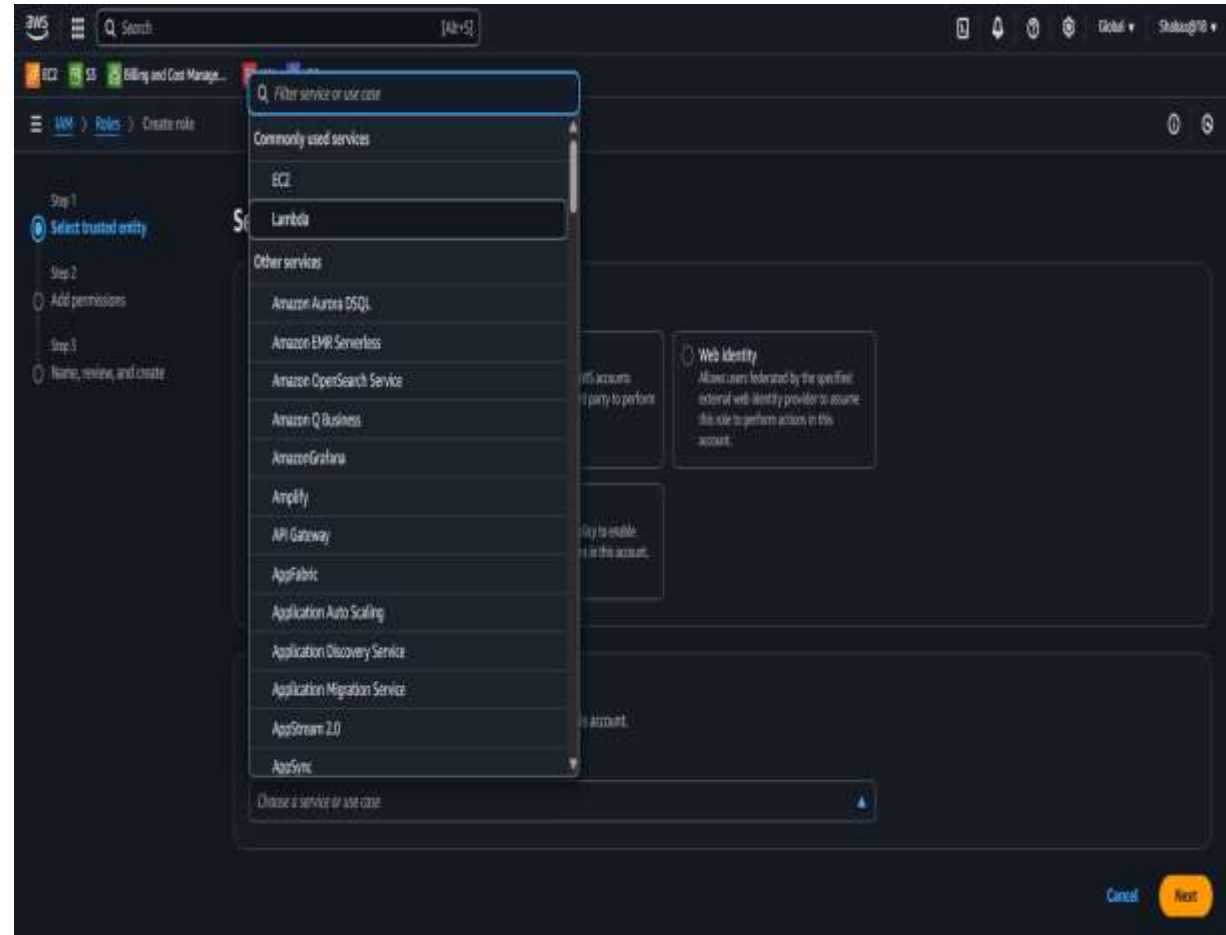
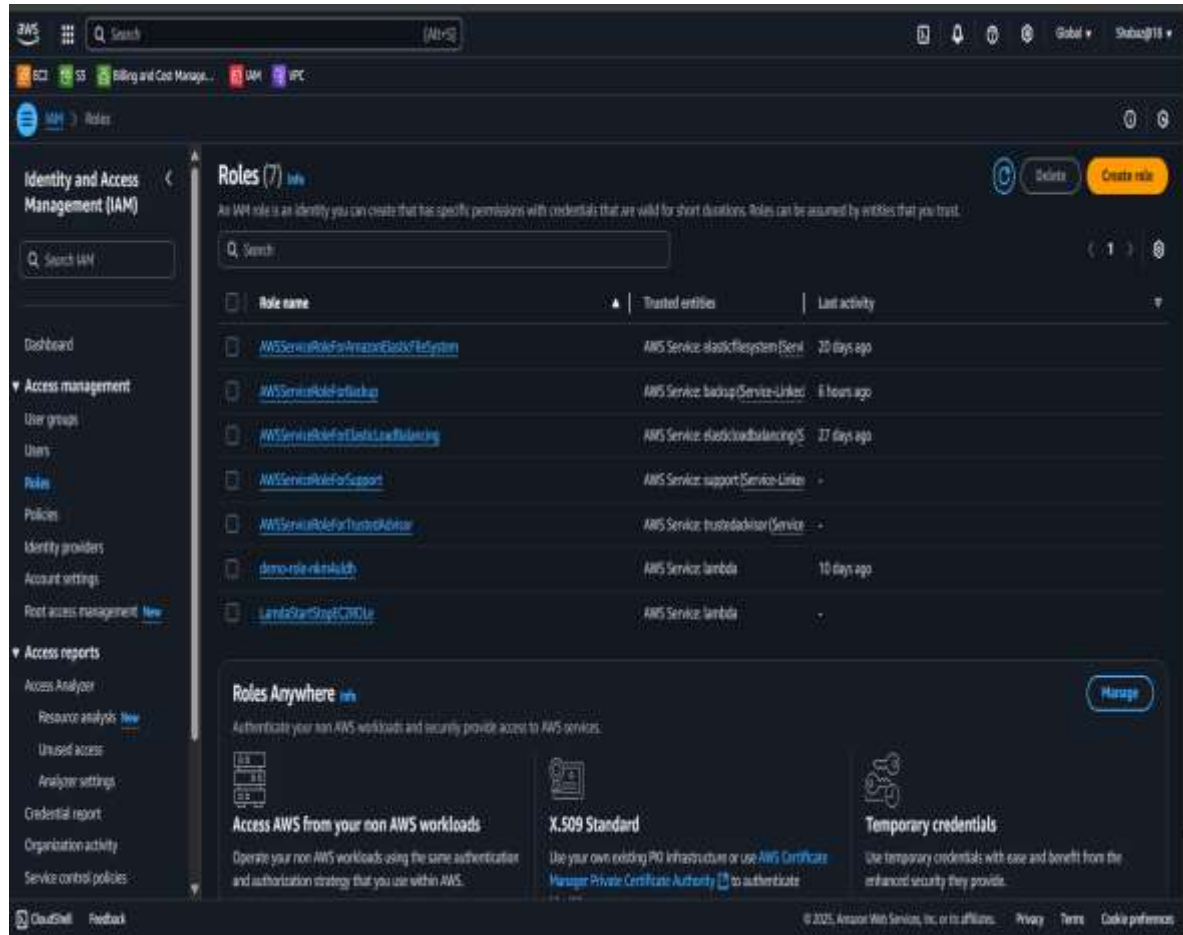
Permissions defined in this policy document specify which actions are allowed or denied. To define permissions for an IAM identity (user, user group, or role), attach a policy to it.

Allow (1 of 445 services) [Show remaining 444 services](#)

| Service             | Access level   | Resource   | Request condition |
|---------------------|----------------|--|-------------------|
| <a href="#">EC2</a> | Limited: Write | InstanceID] string like [All, region] string like [All | None              |

CloudShell Feedback © 2025, Amazon Web Services, Inc. or its affiliates. [Privacy](#) [Terms](#) [Cookie preferences](#)

- Now, we have to Create STOP LAMBDA FUNCTION
- Repeat the same Process again with little bit changes, create an IAM ROLE as it is like :





1. Click on CREATE ROLE, Click on next
2. Select Amazon service , Select Lambda at the USE CASE
3. At ADD permission policies section search for your created policy and select it, then click on Next

The screenshot shows the AWS IAM console interface during the 'Create role' process. The left sidebar indicates the current step is 'Add permissions'. The main content area is titled 'Add permissions' and shows a search for 'EC2-' in the 'Permissions policies' section. A table lists the search results, showing one match: 'EC2-StartStop' with a 'Customer managed' type. At the bottom, there are 'Cancel', 'Previous', and 'Next' buttons.

**Step 1** Select trusted entity

**Step 2** Add permissions

**Step 3** Name, review, and create

### Add permissions Info

Permissions policies (1063) Info

Choose one or more policies to attach to your new role.

Filter by Type

EC2- × All types 1 match

| <input type="checkbox"/> | Policy name <small>↗</small>  | Type             | Description |
|--------------------------|-------------------------------|------------------|-------------|
| <input type="checkbox"/> | <a href="#">EC2-StartStop</a> | Customer managed | -           |

► Set permissions boundary - *optional*

Cancel Previous Next

© 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

- Next, At the ROLE DETAILS Give role name as STOP EC2 and at Description add a short explanation like : Processing Service (like that)

The screenshot shows the AWS IAM console interface for creating a new role. The top navigation bar includes the AWS logo, a search bar, and user information (Shabaz@18). The main navigation pane on the left shows the 'IAM' menu with 'Roles' selected, leading to the 'Create role' page. The page is divided into three steps: 'Step 1: Select trusted entity', 'Step 2: Add permissions', and 'Step 3: Name, review, and create', with the third step being the active one.

**Name, review, and create**

**Role details**

**Role name**  
Enter a meaningful name to identify this role.  
  
Maximum 64 characters. Use alphanumeric and '+,=, @, \_' characters.

**Description**  
Add a short explanation for this role.  
  
Maximum 1000 characters. Use letters (A-Z and a-z), numbers (0-9), tabs, new lines, or any of the following characters: \_+=, @-/[()!#\$%^&\*~'`

**Step 1: Select trusted entities** Edit

**Trust policy**

```
1 {  
2   "Version": "2012-10-17",  
3   "Statement": [  
4     {  
5       "Effect": "Allow",  
6       "Action": [  
7         "sts:AssumeRole"  
8       ],  
9       "Principal": {  
10        "Service": [
```

- The Display will be Appear like below shown figure, Then Click on CREATE ROLE

The screenshot displays the AWS IAM console interface for creating a new role. The top navigation bar includes the AWS logo, a search bar, and user information (Shabaz@18). The main navigation pane on the left shows the 'IAM' section expanded, with 'Roles' selected. The breadcrumb trail indicates the path: IAM > Roles > Create role.

**Step 1: Principal** (partially visible): The JSON configuration for the principal is shown in a code editor. It defines a role for the 'lambda.amazonaws.com' service.

```
9 - "Principal": {
10 -   "Service": [
11 -     "lambda.amazonaws.com"
12 -   ]
13 - }
14 - }
15 - ]
16 - }
```

**Step 2: Add permissions** (partially visible): This section includes a table summarizing the attached permissions policies.

| Policy name                   | Type             | Attached as        |
|-------------------------------|------------------|--------------------|
| <a href="#">EC2-StartStop</a> | Customer managed | Permissions policy |

**Step 3: Add tags** (partially visible): This section provides information about adding tags to the resource.

**Add tags - optional** [Info](#)

Tags are key-value pairs that you can add to AWS resources to help identify, organize, or search for resources.

No tags associated with the resource.

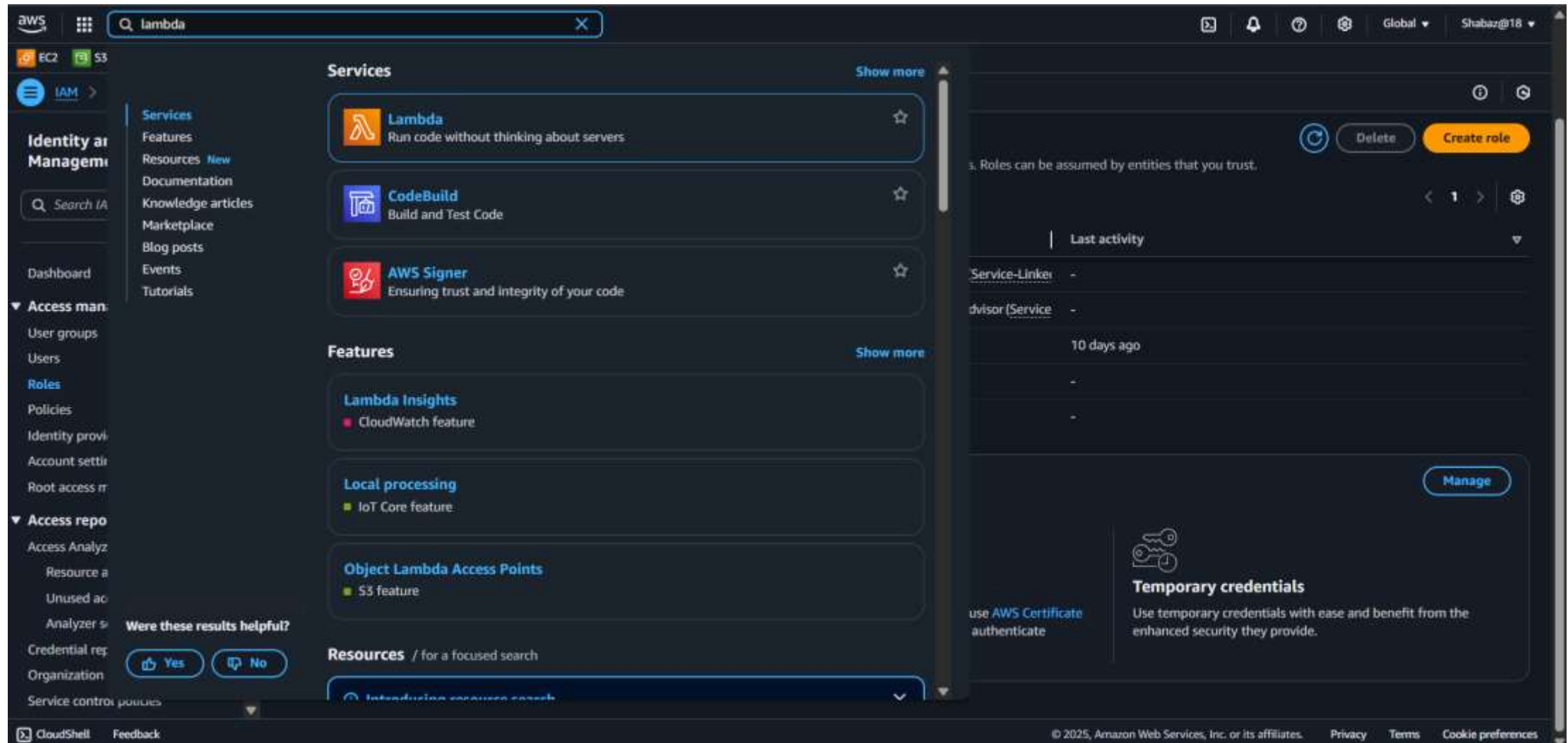
[Add new tag](#)

You can add up to 50 more tags.

At the bottom right, there are three buttons: 'Cancel', 'Previous', and 'Create role' (highlighted in orange).

The footer of the console shows 'CloudShell', 'Feedback', and copyright information for Amazon Web Services, Inc. or its affiliates, along with links for 'Privacy', 'Terms', and 'Cookie preferences'.

- After That Create a LAMBDA Function
- At the search bar, Search Lambda open it



- Here, If you click on DASHBOARD , You can see the how many Functions we created till now
- We have created 1 Already & Now we are going to create another
- Click on CREATE FUNCTION

The screenshot shows the AWS Lambda console dashboard. At the top, there's a navigation bar with the AWS logo, a search bar, and various service icons (EC2, S3, Billing and Cost Management, IAM, VPC). The main content area is divided into several sections:

- Resources for United States (N. Virginia):** This section displays four key metrics:
  - Lambda function(s):** 1
  - Code storage:** 299 byte (0% of 75 GB)
  - Full account concurrency:** 10
  - Unreserved account concurrency:** 10A "Create function" button is located to the right of these metrics.
- Top 10 functions:** This section contains three charts: "Errors", "Invocations", and "Concurrent Executions". Each chart shows a count over time (09:30 to 11:30). All three charts display "No data available. Try adjusting the dashboard time range."
- Account-level metrics:** This section is partially visible at the bottom.

On the left side, there's a sidebar with navigation links for "Lambda", "Dashboard", "Applications", "Functions", and "Additional resources" (Code signing configurations, Event source mappings, Layers, Replicas). Below these are "Related AWS resources" (Step Functions state machines).

On the right side, there's a "Tutorials" section titled "Create a simple web app". It includes a description of the tutorial and a "Start tutorial" button.

- Give Basic information like : Function name as EC2StartStop

The screenshot shows the AWS Lambda 'Create function' page. The 'Author from scratch' option is selected. The 'Function name' field is filled with 'EC2StartStop'. The 'Runtime' is set to 'Node.js 22.x' and 'Architecture' is set to 'x86\_64'. The 'Permissions' section is partially visible at the bottom.

**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.

Function name must be 1 to 64 characters, must be unique to the Region, and can't include spaces. Valid characters are a-z, A-Z, 0-9, hyphens (-), and underscores (\_).

**Runtime** [Info](#)  
Choose the language to use to write your function. Note that the console code editor supports only Node.js, Python, and Ruby.

**Architecture** [Info](#)  
Choose the instruction set architecture you want for your function code.  
☐ arm64  
☒ x86\_64

**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**

**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**

© 2025, Amazon Web Services, Inc. or its affiliates. [Privacy](#) [Terms](#) [Cookie preferences](#)



- Select RUNTIME as PYTHON 3.9

The screenshot shows the AWS Lambda 'Create function' console. The 'Author from scratch' option is selected, and a dropdown menu is open showing the available runtimes. The 'python' search filter is applied, and the list shows 'Python 3.13' as the latest supported runtime, followed by 'Python 3.10', 'Python 3.11', 'Python 3.12', and 'Python 3.9'. The 'Node.js 22.x' runtime is also visible below the Python options. The 'Architecture' section shows 'x86\_64' selected. The 'Permissions' section indicates that a default execution role will be created. On the right, there is a 'Tutorials' section with a link to 'Create a simple web app' and a 'Start tutorial' button.

**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 example code and configuration

☐ **Container image**  
Select a container image to deploy for your function.

Q python

Latest supported

Python 3.13

Other supported

Python 3.10

Python 3.11

Python 3.12

Python 3.9

Node.js 22.x

**Architecture** [Info](#)

Choose the instruction set architecture you want for your function code.

☐ arm64

☒ x86\_64

**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**

**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

© 2025, Amazon Web Services, Inc. or its affiliates. [Privacy](#) [Terms](#) [Cookie preferences](#)

- Select ARCHITECTURE as x86\_64
- Then Click on CREATE FUNCTION

The screenshot shows the AWS Lambda console's 'Create function' page. The 'Basic information' section is expanded, showing the following details:

- Function name:** EC2StartStop. A note states: 'Function name must be 1 to 64 characters, must be unique to the Region, and can't include spaces. Valid characters are a-z, A-Z, 0-9, hyphens (-), and underscores (\_).'.
- Runtime:** Python 3.9. A note states: 'Choose the language to use to write your function. Note that the console code editor supports only Node.js, Python, and Ruby.'
- Architecture:** x86\_64. A note states: 'Choose the instruction set architecture you want for your function code.'
- Permissions:** The default execution role is selected. A note states: '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.'

Below the 'Basic information' section, there are two expandable sections:

- Change default execution role:** A link to change the default execution role.
- Additional configurations:** A link to set up code signing, function URL, tags, and Amazon VPC access for your function.

At the bottom right, there are two buttons: 'Cancel' and 'Create function'.

On the right sidebar, the 'Tutorials' tab is selected, showing a tutorial titled 'Create a simple web app'. The tutorial content includes:

- 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](#)

The footer of the console shows '© 2025, Amazon Web Services, Inc. or its affiliates.' and links for 'Privacy', 'Terms', and 'Cookie preferences'.



# • LAMBDA FUNTION CREATED

The screenshot displays the AWS Lambda console interface. At the top, a green notification banner states: "Successfully created the function EC2StartStop. You can now change its code and configuration. To invoke your function with a test event, choose 'Test'." The breadcrumb navigation shows the path: Lambda > Functions > EC2StartStop. The function name "EC2StartStop" is prominently displayed at the top left of the main content area. To its right are buttons for "Throttle", "Copy ARN", and "Actions". Below the function name, there are tabs for "Diagram" and "Template", and a "Layers" section showing "(0)" layers. A "+ Add trigger" button is located on the left, and a "+ Add destination" button is on the right. The "Description" section on the right shows a hyphen "-". The "Last modified" section shows "13 seconds ago". The "Function ARN" section shows "arn:aws:lambda:us-east-1:722599784033:function:EC2StartStop". The "Function URL" section shows a hyphen "-". At the bottom, there are tabs for "Code", "Test", "Monitor", "Configuration", "Aliases", and "Versions". The "Code" tab is selected, showing a "Code source" section with an "Upload from" button. On the right side, there is a sidebar with "Info" and "Tutorials" tabs. The "Tutorials" tab is active, showing a section titled "Create a simple web app" with a list of steps: "Build a simple web app, consisting of a Lambda function with a function URL that outputs a webpage" and "Invoke your function through its function URL". A "Start tutorial" button is at the bottom of this section. The footer of the console shows the URL "https://console.aws.amazon.com/console/home?region=us-east-1" and the copyright notice "© 2025, Amazon Web Services, Inc. or its affiliates." along with links for "Privacy", "Terms", and "Cookie preferences".

Successfully created the function EC2StartStop. You can now change its code and configuration. To invoke your function with a test event, choose "Test".

## EC2StartStop

Throttle Copy ARN Actions

Function overview Info

Diagram Template

EC2StartStop

Layers (0)

+ Add trigger + Add destination

Export to Infrastructure Composer Download

Description

Last modified 13 seconds ago

Function ARN arn:aws:lambda:us-east-1:722599784033:function:EC2StartStop

Function URL Info

Code Test Monitor Configuration Aliases Versions

Code source Info

Upload from

CloudShell Feedback

© 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

**Create a simple web app**

In this tutorial you will learn how to:

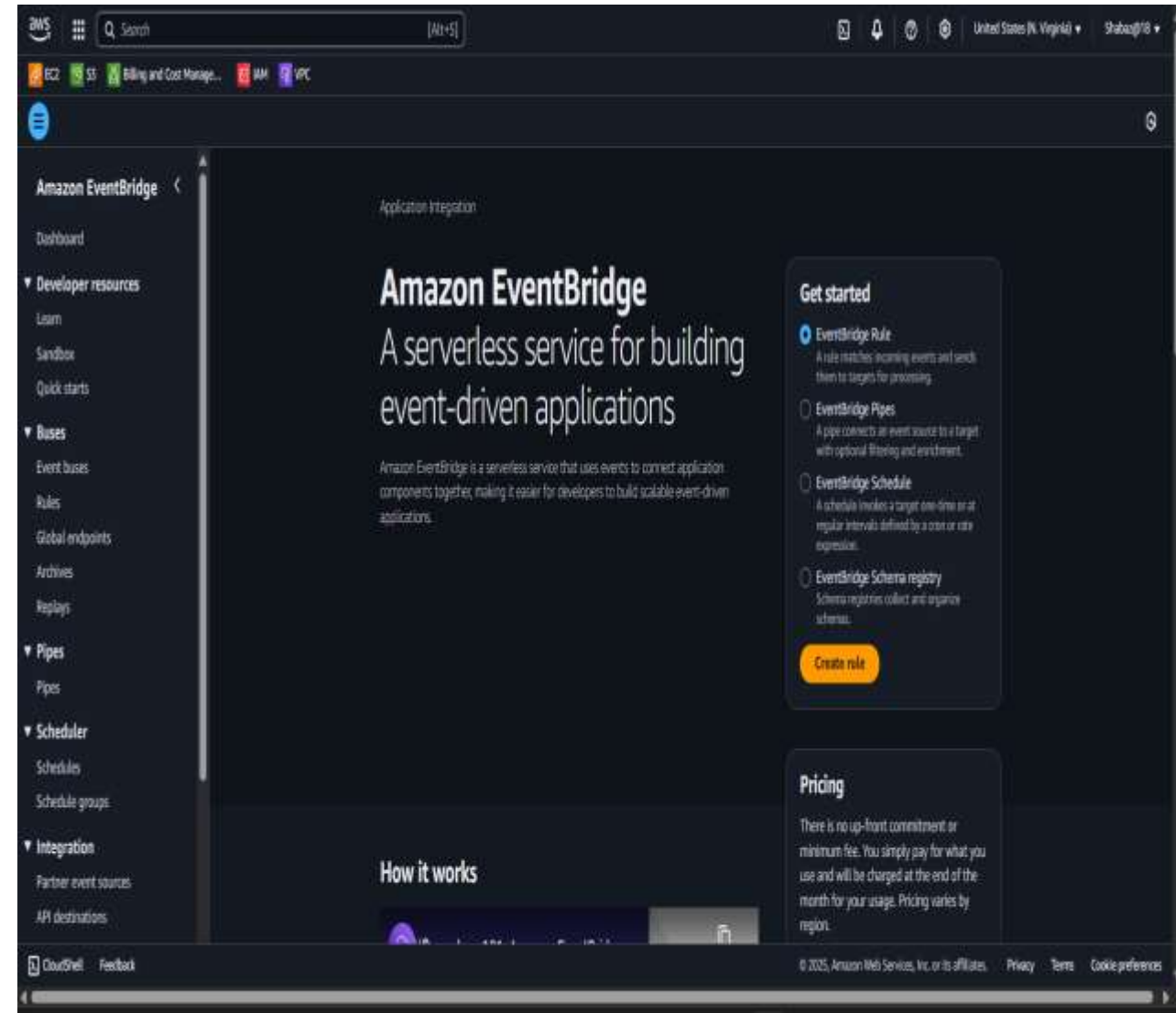
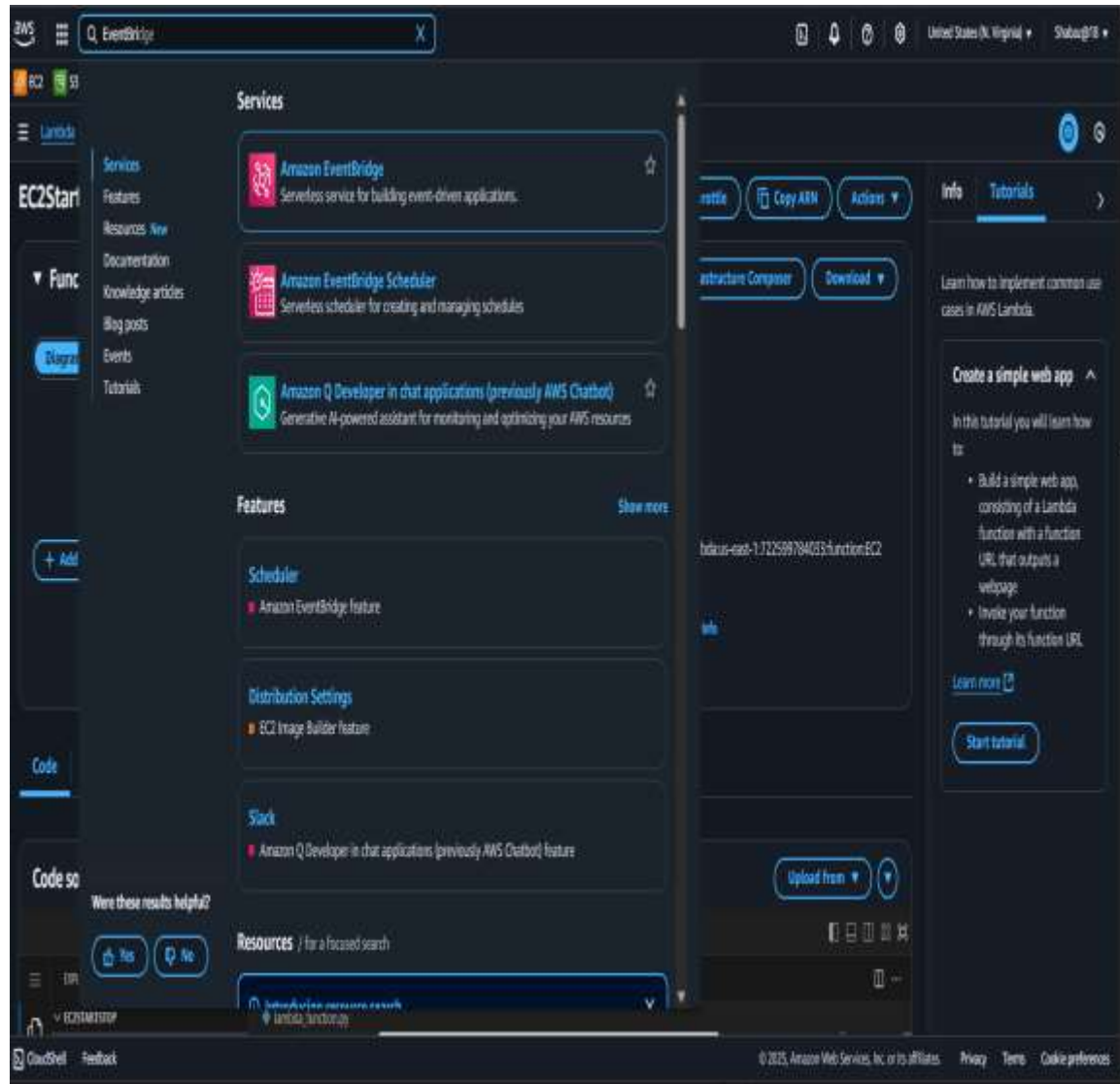
- Build a simple web app, consisting of a Lambda function with a function URL that outputs a webpage
- Invoke your function through its function URL

Learn more

Start tutorial

## >>> NEXT CREATING AMAZON EVENTBRIDGE

- In Search bar, Search for AMAZON EVENTBRIDGE & Open it



- Click on RULE and CREATE NEW RULE

The screenshot displays the AWS Amazon EventBridge console interface. The top navigation bar includes the AWS logo, a search bar, and the user's profile (Shabaz@18) in the United States (N. Virginia) region. The left sidebar shows the navigation menu with categories like Developer resources, Buses, Pipes, Scheduler, and Integration. The main content area is titled 'Rules' and includes a description: 'A rule watches for specific types of events. When a matching event occurs, the event is routed to the targets associated with the rule. A rule can be associated with one or more targets.'

Below the description, there is a 'Select event bus' section with a dropdown menu showing 'default'. The 'Rules' section features a search bar with the placeholder 'Find rules', a status filter set to 'Any status', and a 'Create rule' button. Below this, a table header is visible with columns: Name, Status, Type, ARN, and Description. The table currently displays 'No rules' and a 'Create rule' button.

At the bottom of the console, there is a footer with '© 2025, Amazon Web Services, Inc. or its affiliates.', 'Privacy', 'Terms', and 'Cookie preferences' links.

- Give RULE DETAILS like : NAME as StartEC2Schedule

Step 1

☒ Define rule detail

Step 2

☐ Build event pattern

Step 3

☐ Select target(s)

Step 4 - optional

☐ Configure tags

Step 5

☐ Review and create

Define rule detail

Info

Rule detail

Name

StartEC2Schedule

Maximum of 64 characters consisting of numbers, lower/upper case letters, -, \_.

Description - optional

Enter description

Event bus

Info

Select the event bus this rule applies to, either the default event bus or a custom or partner event bus.

default

▼

☒ Enable the rule on the selected event bus

Rule type

Info

☒ Rule with an event pattern

A rule that runs when an event matches the defined event pattern. EventBridge sends the event to the specified target.

☐ Schedule

A rule that runs on a schedule

Cancel

Next

- At RULE TYPE Select SCHEDULE and Click on Continue in EventBridge Scheduler

Step 2

Define schedule

Step 3

Select target(s)

Step 4 - optional

Configure tags

Step 5

Review and create

### Rule detail

**Name**

StartEC2Schedule

Maximum of 64 characters consisting of numbers, lower/upper case letters, ., -, \_.

**Description - optional**

Enter description

**Event bus** [Info](#)

Select the event bus this rule applies to, either the default event bus or a custom or partner event bus.

default

☒ Enable the rule on the selected event bus

**Rule type** [Info](#)

☐ Rule with an event pattern

A rule that runs when an event matches the defined event pattern. EventBridge sends the event to the specified target.

☒ Schedule

A rule that runs on a schedule

**EventBridge Scheduler - A new AWS scheduling capability!** [New](#)

A new EventBridge scheduling functionality that provides one-time and recurring scheduling functionality independent of Event buses and rules. You can create a schedule to invoke targets such as a Lambda function.

[Learn More](#)

Continue to create rule

Cancel

Continue in EventBridge Scheduler



- After Clicking on (Continue in EventBridge Scheduler) It will appear a new Page
- Give SCHEDULE NAME as StartEC2Schedule

The screenshot shows the AWS EventBridge Schedules console. The breadcrumb navigation is **Amazon EventBridge > Schedules > Create schedule**. On the left, a progress bar shows four steps: **Step 1: Specify schedule detail** (active), **Step 2: Select target**, **Step 3: Settings**, and **Step 4: Review and create schedule**.

The main section is titled **Specify schedule detail** and contains two main panels:

- Schedule name and description**
  - Schedule name:** A text input field containing `StartEC2Schedule`. Below the field, it says: "Use only letters, numbers, dashes, dots or underscores. Max 64 characters."
  - Description - optional:** A text area with the placeholder "Enter description". Below it, it says: "Maximum of 512 characters."
  - Schedule group:** A dropdown menu showing `default`. Below it, it says: "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."
- Schedule pattern**
  - Occurrence:** Two radio buttons: **One-time schedule** (selected) and **Recurring schedule**. Below them, it says: "You can define an one-time or recurring schedule."
  - Date and time:**
    - Date:** A text input field with the placeholder `YYYY/MM/DD` and a calendar icon. Below it, it says: "YYYY/MM/DD".
    - Time:** A text input field with the placeholder `hh:mm` and a clock icon. Below it, it says: "Use 24-hour format timestamp (hh:mm)".
    - Time zone:** A dropdown menu showing `(UTC+05:30) Asia/Calcutta`. Below it, it says: "Time zone".
  - Flexible time window:** A section with the text: "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."

The footer contains links for **CloudShell**, **Feedback**, and copyright information: "© 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences".

- Select Occurrence to RECURRING SCHEDULE at SCHDEULE PATTERN
- And select TIME ZONE as (UTC+00:00) UTC
- Next Select CRON-Based Schedule at SCHEDULE TYPE

The screenshot shows the 'Schedule pattern' configuration interface. The 'Occurrence' dropdown is set to 'Recurring schedule'. The 'Time zone' dropdown is set to '(UTC+00:00) UTC'. The 'Schedule type' dropdown is set to 'Cron-based schedule'. Below these, the 'Cron expression' field is visible with a 'Copy' button and a 'Clear' button. The 'Cron expression' field is currently empty, and the 'Cron' label is visible.

The screenshot shows the 'Schedule pattern' configuration interface. The 'Occurrence' dropdown is set to 'One-time schedule'. The 'Time zone' dropdown is set to '(UTC+00:00) UTC'. The 'Schedule type' dropdown is set to 'Rate-based schedule'. Below these, the 'Cron expression' field is visible with a 'Copy' button and a 'Clear' button. The 'Cron expression' field is currently empty, and the 'Cron' label is visible.

- Set what Schedule you want to - At CRON EXPRESSION
- I have Scheduled at 1:30 Today

aws Search [Alt+S] United States (N. Virginia) Shabaz@18

EC2 S3 Billing and Cost Manage... IAM VPC

Amazon EventBridge > Schedules > Create schedule

**Time zone**  
The time zone for the schedule.  
(UTC+00:00) 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 ( 30 13 \* \* ? \* )  
Minutes Hours Day of month Month Day of the week Year

**Next 10 trigger dates**  
Date and time are displayed in your current time zone in UTC format, e.g. "Wed, Nov 9, 2022 09:00 (UTC - 08:00)" for Pacific time

Thu, 10 Jul 2025 13:30:00 (UTC+00:00)  
Fri, 11 Jul 2025 13:30:00 (UTC+00:00)  
Sat, 12 Jul 2025 13:30:00 (UTC+00:00)  
Sun, 13 Jul 2025 13:30:00 (UTC+00:00)  
Mon, 14 Jul 2025 13:30:00 (UTC+00:00)  
Tue, 15 Jul 2025 13:30:00 (UTC+00:00)  
Wed, 16 Jul 2025 13:30:00 (UTC+00:00)  
Thu, 17 Jul 2025 13:30:00 (UTC+00:00)

CloudShell Feedback © 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences



- Scroll down & Choose a Flexible time windows – What is it ?
- If you choose a flexible time windows, Scheduler invokes your schedule within the time window you specify.
- For Example: If you choose 10 minutes, your schedule runs within 10 minutes after the schedule start time

- THIS STATEMENT CLEARLY MENTIONED AT BOTTOM OF THE FLEXIBLE TIME WINDOWS

The screenshot shows the AWS EventBridge 'Create schedule' page. The 'Flexible time window' dropdown menu is open, displaying the following options: Off, 5 minutes, 10 minutes, 15 minutes, 30 minutes, 1 hour, 2 hours, and 4 hours. The page also includes fields for 'Start date and time - optional' and 'End date and time - optional', both with date and time input boxes. A cron expression field is also visible. The footer of the page contains the text: '© 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences'.

- Scroll Down And Click on Next

The screenshot shows the 'Create schedule' page in the Amazon EventBridge console. The breadcrumb navigation at the top indicates the path: **Amazon EventBridge** > **Schedules** > **Create schedule**. Below the breadcrumb, there is a 'Timeframe' section. At the top of this section, a dropdown menu is set to '5 minutes'. Below this, a blue box contains a 'Daylight saving time' warning: 'Amazon EventBridge Scheduler automatically adjusts your schedule for daylight saving time. When time shifts forward in the Spring, if a cron expression falls on a non-existent date, your schedule invocation is skipped. When time shifts backwards in the Fall, your schedule runs only once and does not repeat its invocation. The following invocations occur normally at the specified date and time.' Below the warning, there are two optional sections: 'Start date and time - optional' and 'End date and time - optional'. Each section has a date input field (placeholder: YYYY/MM/DD) with a calendar icon, and a time input field (placeholder: hh:mm) with a note: 'Use 24-hour format timestamp (hh:mm)'. At the bottom right of the form, there are two buttons: 'Cancel' and 'Next'. The 'Next' button is orange and highlighted. The footer of the page includes 'CloudShell', 'Feedback', and copyright information for Amazon Web Services, Inc. or its affiliates, along with links for 'Privacy', 'Terms', and 'Cookie preferences'.

aws | Search [Alt+S] | United States (N. Virginia) | Shabaz@18

EC2 S3 Billing and Cost Manage... IAM VPC

Amazon EventBridge > Schedules > Create schedule

Timeframe

**Daylight saving time**  
Amazon EventBridge Scheduler automatically adjusts your schedule for daylight saving time. When time shifts forward in the Spring, if a cron expression falls on a non-existent date, your schedule invocation is skipped. When time shifts backwards in the Fall, your schedule runs only once and does not repeat its invocation. The following invocations occur normally at the specified date and time.

**Start date and time - optional**  
The start date and time of the schedule.

YYYY/MM/DD hh:mm  
YYYY/MM/DD Use 24-hour format timestamp (hh:mm)

**End date and time - optional**  
The end date and time of the schedule.

YYYY/MM/DD hh:mm  
YYYY/MM/DD Use 24-hour format timestamp (hh:mm)

Cancel Next

CloudShell Feedback | © 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

- After Clicking Next, You will be Redirected to Previous Page which is Amazon EventBridge - Step 2
- At Target detail Choose Templated targets in that Select AWS LAMBDA

The screenshot displays the AWS Management Console interface for creating a schedule. The top navigation bar includes the AWS logo, a search bar, and various service shortcuts (EC2, S3, Billing and Cost Management, IAM, VPC). The breadcrumb trail indicates the current location: Amazon EventBridge > Schedules > Create schedule. On the left, a sidebar shows the progress of the four-step wizard: Step 2 (Select target), Step 3 - optional (Settings), Step 4 (Review and create schedule), and a final step (Review and create schedule).

The main content area is titled 'Target detail' and includes a 'Target API' section with an 'Info' link. Below this, a message states: 'Select an API that will be invoked as a target for your schedule.' Two tabs are available: 'Templated targets' (selected) and 'All APIs'. A grid of 12 templated targets is shown, each with an icon, service name, and action name. The 'AWS Lambda Invoke' target is highlighted with a blue border and a blue selection circle. At the bottom, the 'Invoke' section shows 'AWS Lambda' as the selected target, and a toggle for 'Universal target definition' is visible.

| Service               | Action                 | Selected |
|-----------------------|------------------------|----------|
| CodeBuild             | StartBuild             | No       |
| CodePipeline          | StartPipelineExecution | No       |
| Amazon ECS            | RunTask                | No       |
| Amazon EventBridge    | PutEvents              | No       |
| Amazon Inspector V1   | StartAssessmentRun     | No       |
| Kinesis Data Firehose | PutRecord              | No       |
| Kinesis Data Streams  | PutRecord              | No       |
| AWS Lambda            | Invoke                 | Yes      |
| Amazon SNS            | Publish                | No       |
| Amazon SQS            | SendMessage            | No       |
| SageMaker             | StartPipelineExecution | No       |
| AWS Step Functions    | StartExecution         | No       |

- Scroll down through INVOKE
- select Start function which we named it as StartEC2instances in Lambda function
- Then Scroll down



- Here You have to Write the Python code of Start EC2 Instance
- Next Click on Deploy to Deploy the code & then Click on Test to Test the code, After that click on enter

aws Search [Alt+S] United States (N. Virginia) Shabaz@18

EC2 S3 Billing and Cost Manage... IAM VPC

Lambda > Functions > StartEC2Instances

Successfully updated the function StartEC2Instances.

Code source Info Upload from

EXPLORER

- STARTEC2INSTANCES
  - lambda\_function.py
- DEPLOY
  - Deploy (Ctrl+Shift+U)
  - Test (Ctrl+Shift+I)
- TEST EVENTS (NONE SELECTED)

lambda\_function.py

```
1 import boto3
2
3 def lambda_handler(event, context):
4     ec2 = boto3.client('ec2')
5     instance_id = 'i-0123456789abcdef0' # Replace with your actual EC2 instance ID
6
7     try:
8         response = ec2.start_instances(InstanceIds=[instance_id])
9         print(f"Starting EC2 instance: {instance_id}")
10        return {
11            'statusCode': 200,
12            'body': f'EC2 instance {instance_id} started successfully!',
13            'details': response['StartingInstances']
14        }
15    except Exception as e:
16        print(f"Error starting EC2 instance: {e}")
17        return {
18            'statusCode': 500,
19            'body': f'Failed to start EC2 instance: {e}'
```

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 © 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

- Same like that you have to Write the Python code of Stop EC2 Instance
- Here are PYTHON CODE for START EC2 INSTANCE & STOP EC2 INSTANCE

### >>> PYTHON CODE to START EC2 INSTANCE

```
import boto3
def lambda_handler(event, context):
    ec2 = boto3.client('ec2')
    instance_id = 'i-0123456789abcdef0' # ↻ Replace with your actual EC2 instance ID
    try:
        response = ec2.start_instances(InstanceIds=[instance_id])
        print(f"Starting EC2 instance: {instance_id}")
        return {
            'statusCode': 200,
            'body': f'EC2 instance {instance_id} started successfully!',
            'details': response['StartingInstances']
        }
    except Exception as e:
        print(f"Error starting EC2 instance: {e}")
        return {
            'statusCode': 500,
            'body': f'Failed to start EC2 instance: {e}'
        }
```



## >>> PYTHON CODE to STOP EC2 INSTANCE

```
import boto3
def lambda_handler(event, context):
    ec2 = boto3.client('ec2')
    instance_id = 'i-0123456789abcdef0' # ↻ Replace with your actual EC2 instance ID
    try:
        response = ec2.stop_instances(InstanceIds=[instance_id])
        print(f"Stopping EC2 instance: {instance_id}")
        return {
            'statusCode': 200,
            'body': f'EC2 instance {instance_id} stopped successfully!',
            'details': response['StoppingInstances']
        }
    except Exception as e:
        print(f"Error stopping EC2 instance: {e}")
        return {
            'statusCode': 500,
            'body': f'Failed to stop EC2 instance: {e}'
        }
```

- Next Open LAMBDA & click on ADD TRIGGERS
- Select EVENTBRIDGE (CloudWatch Events) at Trigger configuration

The screenshot shows the AWS Lambda console interface. At the top, the navigation bar includes the AWS logo, a search bar, and various service icons (EC2, S3, Billing and Cost Management, IAM, VPC). The main header shows 'Lambda > Add triggers'. The 'Add trigger' section is active, displaying a 'Trigger configuration' panel. This panel has a search bar with 'event' entered, showing a dropdown list of triggers. The first result is 'EventBridge (CloudWatch Events)', which is highlighted. Other results include 'SQS' and a list of 'Real-time/streaming data' sources: 'Amazon DocumentDB', 'Apache Kafka', 'DynamoDB', 'Kinesis', and 'MQ'. To the right of the search results are 'Cancel' and 'Add' buttons. On the far right, there's a 'Tutorials' sidebar with the heading 'Create a simple web app' and a list of steps: 'Build a simple web app, consisting of a Lambda function with a function URL that outputs a webpage' and 'Invoke your function through its function URL'. A 'Start tutorial' button is at the bottom of this sidebar. The footer contains 'CloudShell', 'Feedback', and copyright information for Amazon Web Services, Inc. or its affiliates, along with links for 'Privacy', 'Terms', and 'Cookie preferences'.

- Choose Create a New Rule at RULE , Then Give Rule name as StartTrigger
- Next give RULE DESCRIPTION (anything you want) like : Processing Description

The screenshot displays the AWS Lambda console's 'Add trigger' page. The top navigation bar includes the AWS logo, a search bar, and various service icons (EC2, S3, Billing and Cost Management, IAM, VPC). The main header shows 'Lambda > Add triggers'. The 'Add trigger' section is divided into 'Trigger configuration' and 'Rule' settings. In 'Trigger configuration', 'EventBridge (CloudWatch Events)' is selected. Under 'Rule', 'Create a new rule' is chosen. The 'Rule name' field is filled with 'StartTrigger', and the 'Rule description' field contains 'Processing Description'. The 'Rule type' section has 'Schedule expression' selected. On the right, the 'Tutorials' tab is active, showing a 'Create a simple web app' tutorial with a 'Start tutorial' button.

**Add trigger**

**Trigger configuration** [Info](#)

EventBridge (CloudWatch Events)  
aws asynchronous schedule management-tools

**Rule**  
Pick an existing rule, or create a new one.

☒ Create a new rule  
☐ Existing rules

**Rule name**  
Enter a name to uniquely identify your rule.

StartTrigger

**Rule description**  
Provide an optional description for your rule.

Processing Description

**Rule type**  
Trigger your target based on an event pattern, or based on an automated schedule.

☐ Event pattern  
☒ Schedule expression

**Schedule expression**

**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](#)

© 2025, Amazon Web Services, Inc. or its affiliates. [Privacy](#) [Terms](#) [Cookie preferences](#)

- Then choose Rule type as SCHEDULE EXPRESSION
- And enter Schedule Expression like **CRON(30 13 \* \* ? \*)** like you given Before at the Amazon EventBridge > Schedulers
- Then Click on ADD

The screenshot shows the AWS Lambda console's 'Add triggers' page. The 'Existing rules' section is collapsed. The 'Rule name' field is filled with 'StartTrigger'. The 'Rule description' field is filled with 'Processing Description'. Under 'Rule type', the 'Schedule expression' option is selected. The 'Schedule expression' field contains the cron expression 'cron(30 13 \* \* ? \*)'. Below this field, there is a note: 'Self-trigger your target on an automated schedule using [Cron or rate expressions](#). Cron expressions are in UTC. e.g. rate(1 day), cron(0 17 ? \* MON-FRI \*)'. At the bottom right, there are 'Cancel' and 'Add' buttons. The 'Add' button is orange and highlighted. On the right side, there is a sidebar with 'Info' and 'Tutorials' tabs. The 'Tutorials' tab is active, showing a section titled 'Create a simple web app' with a list of steps: 'Build a simple web app, consisting of a Lambda function with a function URL that outputs a webpage' and 'Invoke your function through its function URL'. There is a 'Learn more' link and a 'Start tutorial' button.

aws Search [Alt+S] United States (N. Virginia) Shabaz@18

EC2 S3 Billing and Cost Manage... IAM VPC

Lambda > Add triggers

☐ Existing rules

**Rule name**  
Enter a name to uniquely identify your rule.

StartTrigger

**Rule description**  
Provide an optional description for your rule.

Processing Description

**Rule type**  
Trigger your target based on an event pattern, or based on an automated schedule.

☐ Event pattern

☒ Schedule expression

**Schedule expression**  
Self-trigger your target on an automated schedule using [Cron or rate expressions](#). Cron expressions are in UTC.

cron(30 13 \* \* ? \*)

e.g. rate(1 day), cron(0 17 ? \* MON-FRI \*)

Lambda will add the necessary permissions for Amazon EventBridge (CloudWatch Events) to invoke your Lambda function from this trigger. [Learn more](#) about the Lambda permissions model.

Cancel Add

**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

© 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

- Here You can see the Added trigger to our Start EC2 instance

aws | Search [Alt+S] | United States (N. Virginia) | Shabaz@18

EC2 S3 Billing and Cost Manage... IAM VPC

Lambda > Functions > StartEC2Instances

### StartEC2Instances

Throttle Copy ARN Actions

✓ The trigger StartTrigger was successfully added to function StartEC2Instances. The function is now receiving events from the trigger.

Function overview Info

Diagram Template

StartEC2Instances

Layers (0)

EventBridge (CloudWatch Events)

+ Add trigger

+ Add destination

Export to Infrastructure Composer Download

Description

Last modified 7 minutes ago

Function ARN  
arn:aws:lambda:us-east-1:722599784033:function:StartEC2Instances

Function URL Info

Code Test Monitor Configuration Aliases Versions

CloudShell Feedback

© 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

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



- Same like that, Open another LAMBDA function which is Stop function that we named as EC2StartStop Function
- Click on ADD Trigger

The screenshot displays the AWS Lambda console interface for a function named 'EC2StartStop'. The top navigation bar includes the AWS logo, a search bar, and various service icons (EC2, S3, Billing and Cost Management, IAM, VPC). The breadcrumb trail shows 'Lambda > Functions > EC2StartStop'. The function's name 'EC2StartStop' is prominently displayed at the top left of the main content area, with buttons for 'Throttle', 'Copy ARN', and 'Actions' to its right. Below the name, there are tabs for 'Function overview' (selected) and 'Info'. The 'Function overview' section contains a 'Diagram' tab and a 'Template' tab. A visual representation of the function is shown, including the Lambda icon, the function name 'EC2StartStop', and a 'Layers' section with '(0)' layers. There are buttons for '+ Add trigger' and '+ Add destination'. To the right of the visual representation, there are buttons for 'Export to Infrastructure Composer' and 'Download'. The 'Description' field is empty. The 'Last modified' timestamp is '30 seconds ago'. The 'Function ARN' is 'arn:aws:lambda:us-east-1:722599784033:function:EC2StartStop'. The 'Function URL' is empty. The 'Code' tab is selected at the bottom, showing the 'Code source' section with an 'Upload from' button. The right sidebar contains a 'Tutorials' section with a link to 'Create a simple web app' and a 'Start tutorial' button. The footer includes 'CloudShell', 'Feedback', and copyright information for Amazon Web Services, Inc. or its affiliates.

aws | Search [Alt+S] | United States (N. Virginia) | Shabaz@18

EC2 S3 Billing and Cost Manage... IAM VPC

Lambda > Functions > EC2StartStop

### EC2StartStop

Throttle Copy ARN Actions

Function overview Info

Diagram Template

EC2StartStop

Layers (0)

+ Add trigger + Add destination

Export to Infrastructure Composer Download

Description

Last modified 30 seconds ago

Function ARN arn:aws:lambda:us-east-1:722599784033:function:EC2StartStop

Function URL Info

Code Test Monitor Configuration Aliases Versions

Code source Info

Upload from

CloudShell Feedback

© 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

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



- Select EVENTBRIDGE (CloudWatch Events) at Trigger configuration
- Choose Create a New Rule at RULE , Then Give Rule name as StopTrigger
- Next give RULE DESCRIPTION (anything you want) like : Processing Description

aws Search [Alt+S]

EC2 S3 Billing and Cost Manage... IAM VPC

Lambda > Add triggers

### Trigger configuration [Info](#)

**EventBridge (CloudWatch Events)**  
aws asynchronous schedule management-tools

**Rule**  
Pick an existing rule, or create a new one.

☒ Create a new rule  
☐ Existing rules

**Rule name**  
Enter a name to uniquely identify your rule.

StopTrigger

**Rule description**  
Provide an optional description for your rule.

Processing Description

**Rule type**  
Trigger your target based on an event pattern, or based on an automated schedule.

☐ Event pattern  
☒ Schedule expression

**Schedule expression**  
Self-trigger your target on an automated schedule using [Cron or rate expressions](#). Cron expressions are in UTC.

[CloudShell](#) [Feedback](#)

© 2025, Amazon Web Services, Inc. or its affiliates. [Privacy](#) [Terms](#) [Cookie preferences](#)

**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](#)

- Then choose Rule type as SCHEDULE EXPRESSION
- And enter Schedule Expression like **CRON(0 13 \* \* ? \*)** like you given Before at the Amazon EventBridge > Schedulers
- Then Click on ADD

The screenshot shows the AWS Lambda console's 'Add triggers' page. The 'Existing rules' section is active. The 'Rule name' field is 'StopTrigger'. The 'Rule description' field is 'Processing Description'. The 'Rule type' section has 'Schedule expression' selected. The 'Schedule expression' field contains 'cron(0 13 \* \* ? \*)'. Below this field, there is a note: 'Self-trigger your target on an automated schedule using [Cron or rate expressions](#). Cron expressions are in UTC. e.g. rate(1 day), cron(0 17 ? \* MON-FRI \*)'. At the bottom right, there are 'Cancel' and 'Add' buttons. The 'Add' button is orange and highlighted.

aws | Search [Alt+S] | United States (N. Virginia) | Shabaz@18

EC2 S3 Billing and Cost Manage... IAM VPC

Lambda > Add triggers

☐ Existing rules

**Rule name**  
Enter a name to uniquely identify your rule.

StopTrigger

**Rule description**  
Provide an optional description for your rule.

Processing Description

**Rule type**  
Trigger your target based on an event pattern, or based on an automated schedule.

☐ Event pattern  
☒ Schedule expression

**Schedule expression**  
Self-trigger your target on an automated schedule using [Cron or rate expressions](#). Cron expressions are in UTC.

cron(0 13 \* \* ? \*)  
e.g. rate(1 day), cron(0 17 ? \* MON-FRI \*)

Lambda will add the necessary permissions for Amazon EventBridge (CloudWatch Events) to invoke your Lambda function from this trigger. [Learn more](#) about the Lambda permissions model.

Cancel Add

**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 | © 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

- Here You can see the Added trigger to our Start EC2 instance

The screenshot displays the AWS Lambda console interface for the function **EC2StartStop**. At the top, a green notification bar states: "The trigger StopTrigger was successfully added to function EC2StartStop. The function is now receiving events from the trigger." Below this, the **Function overview** section shows a diagram with the **EC2StartStop** function box connected to an **EventBridge (CloudWatch Events)** box. A **+ Add trigger** button is visible in the EventBridge box. To the right, the **Configuration** tab is active, showing details such as **Description** (empty), **Last modified** (4 minutes ago), **Function ARN** (arn:aws:lambda:us-east-1:722599784033:function:EC2StartStop), and **Function URL** (empty). A **+ Add destination** button is also present. The right sidebar features a **Tutorials** section with a card titled **Create a simple web app**, which includes instructions on building a web app and invoking a Lambda function, along with a **Start tutorial** button. The bottom of the console shows navigation tabs for **Code**, **Test**, **Monitor**, **Configuration** (selected), **Aliases**, and **Versions**. The footer contains links for **CloudShell**, **Feedback**, and copyright information for Amazon Web Services, Inc. or its affiliates.

- **AWS CloudWatch Events** were configured to **automate the timing** of EC2 start and stop actions.

**Two scheduled rules were created:**

- **Start Trigger:** Every day at 1 PM
  - **Stop Trigger:** Every day at 1:30 PM
- 
- These rules are **connected to respective Lambda functions**, enabling **automated EC2 management** without human intervention.

# SUMMARY :

- The **key advantages** of the project:
  - **Automation:** EC2 operations run without manual action
  - **Cost Efficiency:** Idle resources are shut down on schedule
  - **Scalability:** Supports managing multiple EC2 instances
- It also concludes that this solution offers a **simple, serverless, and effective approach** to cloud automation using core AWS services.

# CONCLUSION :

- This project demonstrates how powerful and cost-effective cloud automation can be when the right AWS services are integrated.
- By combining **Lambda**, **CloudWatch**, and **IAM**, we've built a hands-free system that reflects the future of **efficient cloud resource management**.

*“In the cloud, efficiency isn't an option — it's a strategy.”*