Assignment 1: Automated Instance Management Using AWS Lambda and Boto3

Launch EC2 Instances

      1. In the AWS Console, search for EC2 in the search bar and click EC2.
      2. Click Launch instances.

            Name: NikhilAutoStop
            Tag:
            Key: Action
            Value: Auto-Stop

            Name: NikhilAutoStart
            Tag:
            Key: Action
            Value: Auto-Start

            AMI : Ubuntu
            Instance Type : t2.micro
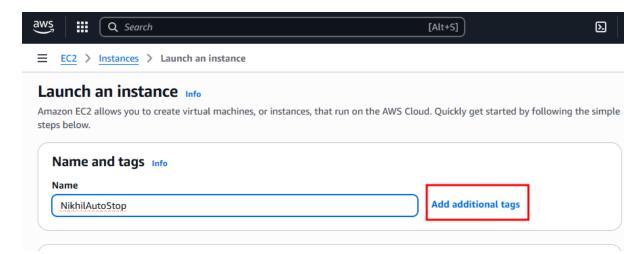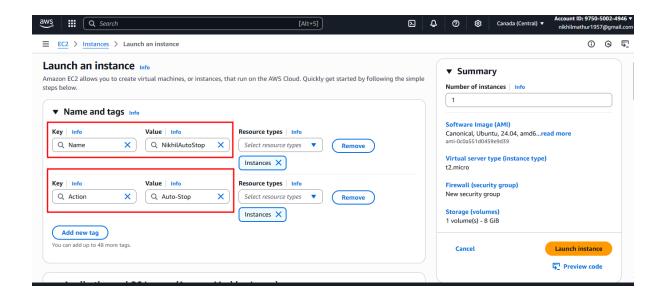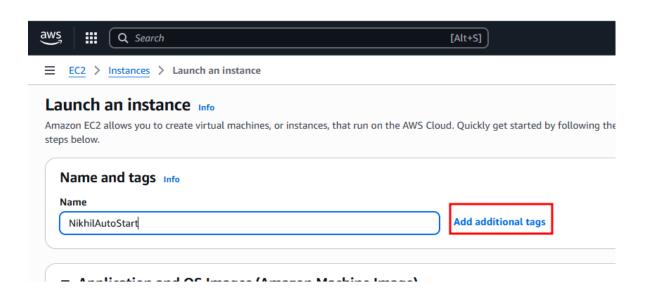   **Key pair (login):**
            Network settings:
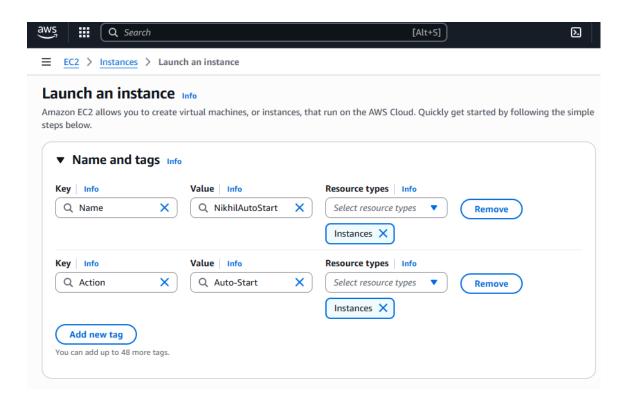   Allow SSH (port 22) from My IP.
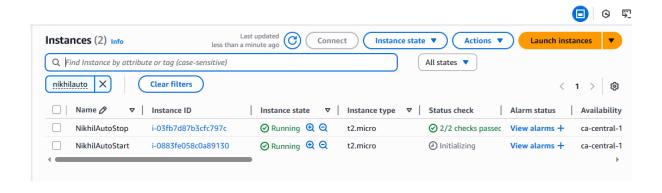   Allow HTTP (port 80)

            Click Launch instance

☰ EC2 ⟩ Instances ⟩ Launch an instance   ⓘ ◉ ⬚

Launch an instance Info

Amazon EC2 allows you to create virtual machines, or instances, that run on the AWS Cloud. Quickly get started by following the simple steps below.

▼ Name and tags Info

Key	Info	Value	Info	Resource types	Info	
🔍 Name	✕	🔍 NikhilAutoStop	✕	Select resource types ▼		Remove
				Instances ✕		

Key	Info	Value	Info	Resource types	Info	
🔍 Action	✕	🔍 Auto-Stop	✕	Select resource types ▼		Remove
				Instances ✕		

Add new tag

You can add up to 48 more tags.

▼ Summary

Number of instances | Info

| 1 |

Software Image (AMI)
Canonical, Ubuntu, 24.04, amd6...read more
ami-0c0a551d0459e9d39

Virtual server type (instance type)
t2.micro

Firewall (security group)
New security group

Storage (volumes)
1 volume(s) - 8 GiB

Cancel   **Launch instance**

🖵 Preview code

☰ EC2 ⟩ Instances ⟩ Launch an instance

Launch an instance Info

Amazon EC2 allows you to create virtual machines, or instances, that run on the AWS Cloud. Quickly get started by following the steps below.

Name and tags Info

Name

| NikhilAutoStart | **Add additional tags**

Launch an instance Info

Amazon EC2 allows you to create virtual machines, or instances, that run on the AWS Cloud. Quickly get started by following the simple steps below.

▼ Name and tags Info

Key Info	Value Info	Resource types Info	
🔍 Name ✕	🔍 NikhilAutoStart ✕	Select resource types ▼	Remove
		Instances ✕	
Key Info	Value Info	Resource types Info	
🔍 Action ✕	🔍 Auto-Start ✕	Select resource types ▼	Remove
		Instances ✕	

Add new tag

You can add up to 48 more tags.

Instances (2) Info

Last updated less than a minute ago ↻ Connect | Instance state ▼ | Actions ▼ | **Launch instances** ▼

🔍 Find Instance by attribute or tag (case-sensitive)   All states ▼

nikhilauto ✕   **Clear filters**   ‹ 1 › ⚙

☐	Name ✎ ▽	Instance ID	Instance state ▽	Instance type ▽	Status check	Alarm status	Availability
☐	NikhilAutoStop	i-03fb7d87b3cfc797c	⊘ Running 🔍 🔍	t2.micro	⊘ 2/2 checks passec	View alarms +	ca-central-1
☐	NikhilAutoStart	i-0883fe058c0a89130	⊘ Running 🔍 🔍	t2.micro	⏱ Initializing	View alarms +	ca-central-1

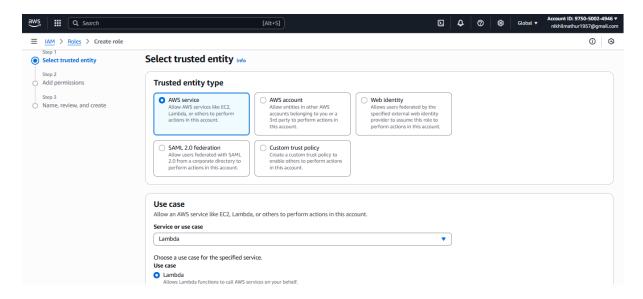**Lambda Function Creation:**

1. **Create an IAM Role for Lambda**
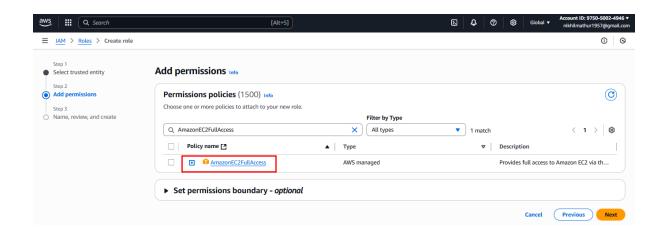   - AWS Console => IAM =>Roles => Create role.
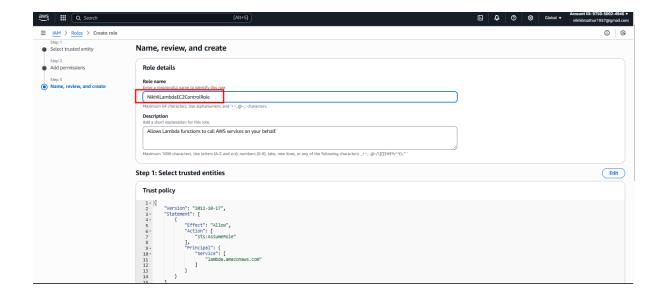   - Trusted Entity type: AWS Services
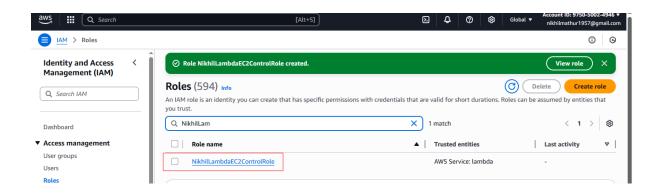   - Use Case: Lambda
   - Click Next

Permissions policies: AmazonEC2FullAccess

Role name: NikhilLambdaEC2ControlRole
Click Create role.

2. Create the Lambda Function
Go to AWS Console => Lambda.
Click Create function.
Select Author from scratch
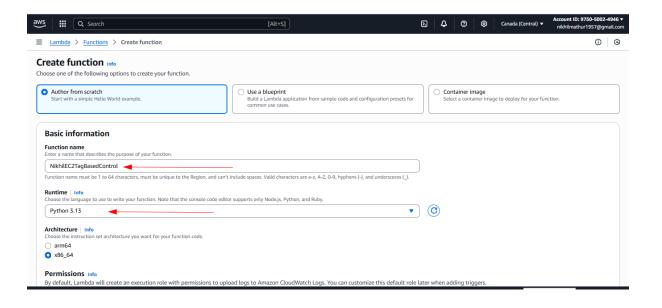Function name: **NikhilEC2TagBasedControl**
Runtime: Python 3.13.
Permissions:
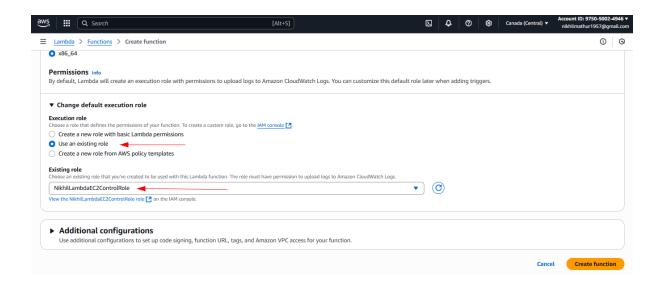Expand Change default execution role.
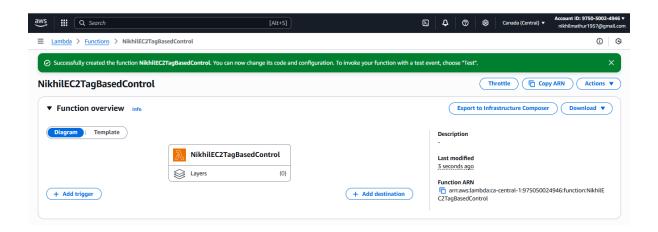Select Use an existing role.
Choose **NikhilLambdaEC2ControlRole** from the dropdown.
Note => I choose the role **prashantb12-role-9p53470y** for permission access to run the code.
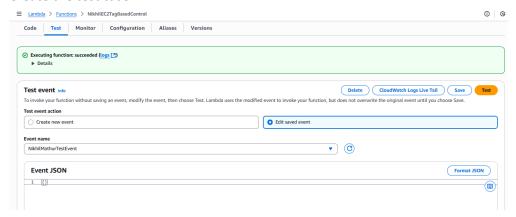**Click Create function.**

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.

`NikhilEC2TagBasedControl`

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.

`Python 3.13`

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.

● 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

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.

`NikhilLambdaEC2ControlRole`

View the NikhilLambdaEC2ControlRole role ↗ on the IAM console.

▶ **Additional configurations**
Use additional configurations to set up code signing, function URL, tags, and Amazon VPC access for your function.

Cancel   **Create function**

✓ Successfully created the function **NikhilEC2TagBasedControl**. You can now change its code and configuration. To invoke your function with a test event, choose "Test".   ✕

NikhilEC2TagBasedControl

Throttle   Copy ARN   Actions ▼

▼ **Function overview** Info

Export to Infrastructure Composer   Download ▼

Diagram | Template

NikhilEC2TagBasedControl
Layers   (0)

+ Add trigger   + Add destination

Description
-

Last modified
3 seconds ago

Function ARN
arn:aws:lambda:ca-central-1:975050024946:function:NikhilEC2TagBasedControl

3. **Add Python Code to Control EC2**
   - **prints instance IDs** with a given tag:
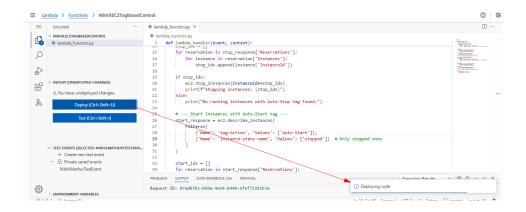   - **Code to print instances with Auto-Stop (testing code)**

```python
import boto3

def lambda_handler(event, context):
    ec2 = boto3.client('ec2')

    # Find instances with Action=Auto-Stop
    response = ec2.describe_instances(
        Filters=[
            {'Name': 'tag:Action', 'Values': ['Auto-Stop']}
        ]
    )

    print(response['Reservations'])

    instances = []
    for reservation in response['Reservations']:
        for instance in reservation['Instances']:
            instances.append(instance['InstanceId'])

    print(f"Found instances with Auto-Stop tag: {instances}")
    return {"instances": instances}
```

   - **Full Code**

```python
import boto3

def lambda_handler(event, context):
    ec2 = boto3.client('ec2')

    # --- Stop instances with Auto-Stop tag ---
    stop_response = ec2.describe_instances(
        Filters=[
            {'Name': 'tag:Action', 'Values': ['Auto-Stop']},
            {'Name': 'instance-state-name', 'Values': ['running']}  # Only running ones
        ]
    )

    stop_ids = []
    for reservation in stop_response['Reservations']:
        for instance in reservation['Instances']:
            stop_ids.append(instance['InstanceId'])

    if stop_ids:
        ec2.stop_instances(InstanceIds=stop_ids)
        print(f"Stopping instances: {stop_ids}")
    else:
        print("No running instances with Auto-Stop tag found.")

    # --- Start instances with Auto-Start tag ---
    start_response = ec2.describe_instances(
        Filters=[
            {'Name': 'tag:Action', 'Values': ['Auto-Start']},
            {'Name': 'instance-state-name', 'Values': ['stopped']}  # Only stopped ones
        ]
    )
```
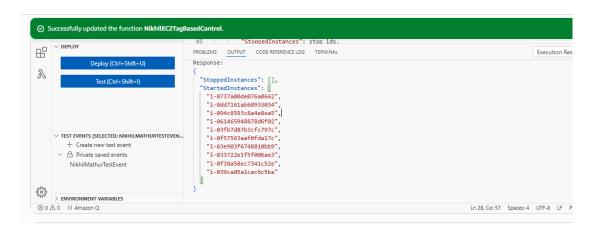
```
30              ]
31          )
32
33          start_ids = []
34          for reservation in start_response['Reservations']:
35              for instance in reservation['Instances']:
36                  start_ids.append(instance['InstanceId'])
37
38          if start_ids:
39              ec2.start_instances(InstanceIds=start_ids)
40              print(f"Starting instances: {start_ids}")
41          else:
42              print("No stopped instances with Auto-Start tag found.")
43
44          return {
45              "StoppedInstances": stop_ids,
46              "StartedInstances": start_ids
47          }
48
```

Create the test case

Click for deploy the code

Click on test for Output

Verification