

**Gift
Abled**



Fostering Inclusion

**Project Submitted by__ Ratna Sai Kumar
Gmail__Ratnasaikumar419@gmail.com**

PROJECT-2

Project Title:- AWS EC2 Instance Automation with Lambda and CloudWatch, SNS.

Project Description:-

- Automating the start and stop of an Amazon EC2 instance using AWS Lambda and CloudWatch Events is a practical approach to optimize costs and ensure the instance runs only when necessary. Here's how you can set up this automation.
- We have a requirement to automate the start and stop of an Amazon EC2 instance on a daily basis. This automation will help us reduce costs and ensure that the EC2 instance is only running during the required hours. We will achieve this by leveraging AWS Lambda for the execution and AWS CloudWatch Events for scheduling.

Project Objectives:-

- Automatically start an Amazon EC2 instance at 9:00 AM daily.
- Automatically stop the same EC2 instance at 6:00 PM daily

Used for services of required is -5

- EC2
- IAM
- LAMBDA
- CLOUDWATCH
- SNS

Creation of EC2 Instance:-

first go to Aws console after that click EC2 instance service

- Dashboard Click on "EC2" under the "Compute" section.
- Click "Launch Instance" to start the instance creation process.
- create the new name Project-2 ec2 launch

Given the quick sort as the amazon Linux or ubantu Iam this is here for amazon Linux.

After that create for key pair like “not recommaned default

Click "Launch Instances" and wait for the instance to initialize.

Successfully for the running initial state.

The screenshot shows the AWS EC2 Instances "Launch an instance" page. A green success bar at the top states "Successfully initiated launch of instance (i-0361182ef10094641)". Below it, a "Next Steps" section lists several options: "Create billing and free tier usage alerts", "Connect to your instance", "Connect an RDS database", and "Create EBS snapshot policy". A search bar and navigation arrows are also present.

The screenshot shows the AWS EC2 Instances page. The left sidebar is expanded to show "Instances" under "Instances". The main area displays a table titled "Instances (1) Info" with one row: "lambda-ec2-aut..." (Instance ID: i-0361182ef10094641, State: Running, Type: t2.micro). A modal window titled "Select an instance" is open over the table. The bottom of the screen shows the Windows taskbar with various pinned icons.

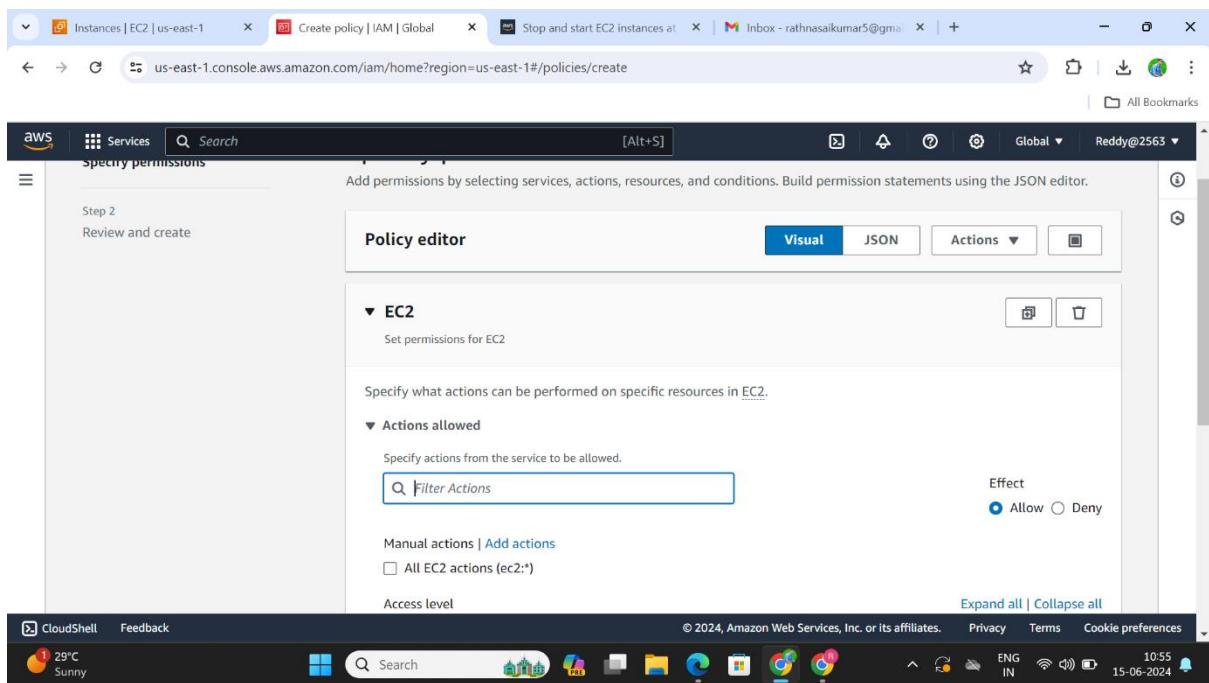
• Create the IAM Policy:-

Select the IAM service.

After that select the policy in the left side policy click

After the policy create is the new name any create

After the EC2 instance is we have to list write any we are click expand all and ec2 start instance select and stop instance will be select automatically running



The screenshot shows the AWS IAM Policies creation page. In the left sidebar, there's a navigation menu with 'Services' selected. The main content area displays a table of actions categorized by service and operation type. Under the 'StartInstances' category, the 'StartInstances' action is checked. Under the 'StopInstances' category, the 'StopInstances' action is also checked. Other actions listed include 'RestoreSnapshotFromRecycleBin', 'RevokeSecurityGroupEgress', 'RunScheduledInstances', 'SendDiagnosticInterrupt', 'UnassignIpv6Addresses', 'UnassignPrivateIpAddress', 'UnassignPrivateNatGatewayAddress', 'UpdateSecurityGroupRuleDescriptionsEgress', 'RestoreSnapshotTier', 'RevokeSecurityGroupIngress', 'RunInstances', 'SendSpotInstanceInterruptions', 'StartNetworkInsightsAccessScopeAnalysis', 'StopInstances', 'TerminateClientVpnConnections', 'UnmonitorInstances', 'UnlockSnapshot', and 'WithdrawByoipCidr'. The table includes columns for 'Action', 'Description', 'Type', and 'Version'.

Start instance and stop instance select the option

Create the policy-automate-ec2

The screenshot shows the AWS IAM Policies creation page. The top navigation bar has 'Services' selected. The main form is titled 'Create policy' and contains fields for 'Name' and 'Description'. The 'Name' field is filled with 'policy-automate-ec2'. The 'Description' field is filled with 'policy-automate-ec2'. Below these fields is a section titled 'Permissions defined in this policy' with a link to 'Edit'. A note states: '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.' A search bar and a button to 'Allow (1 of 417 services)' are shown. At the bottom, there's a note about showing remaining 416 services. The status bar at the bottom indicates it's 11:02, 15-06-2024, and shows system icons.

The screenshot shows the AWS IAM Policies details page for the 'policy-automate-ec2' policy. The policy is customer-managed, created on June 15, 2024, at 11:02 UTC+05:30, and edited on the same day at 11:02 UTC+05:30. The ARN is arn:aws:iam::533267448465:policy/policy-automate-ec2. The 'Permissions' tab is selected, showing a summary of permissions defined in the policy.

Type	Creation time	Edited time	ARN
Customer managed	June 15, 2024, 11:02 (UTC+05:30)	June 15, 2024, 11:02 (UTC+05:30)	arn:aws:iam::533267448465:policy/policy-automate-ec2

Permissions defined in this policy

Policy name	Type	Description
Amazon-EventBridge...	Customer managed	-
AWSLambdaBasicExec...	Customer managed	-
policy-automate-ec2	Customer managed	policy-automate-ec2
StopEc2	Customer managed	-
AdministratorAccess	AWS managed - job function	Provides full access to AWS services an...
PowerUserAccess	AWS managed - job function	Provides full access to AWS services an...
ReadOnlyAccess	AWS managed - job function	Provides read-only access to AWS servi...

The screenshot shows the 'Add permissions' step of creating a new IAM role. The user is choosing policies to attach. The 'policy-automate-ec2' policy is selected and highlighted in blue. Other available policies include Amazon-EventBridge..., AWSLambdaBasicExec..., StopEc2, AdministratorAccess, PowerUserAccess, and ReadOnlyAccess.

After the take create name ‘policy-automate-ec2’

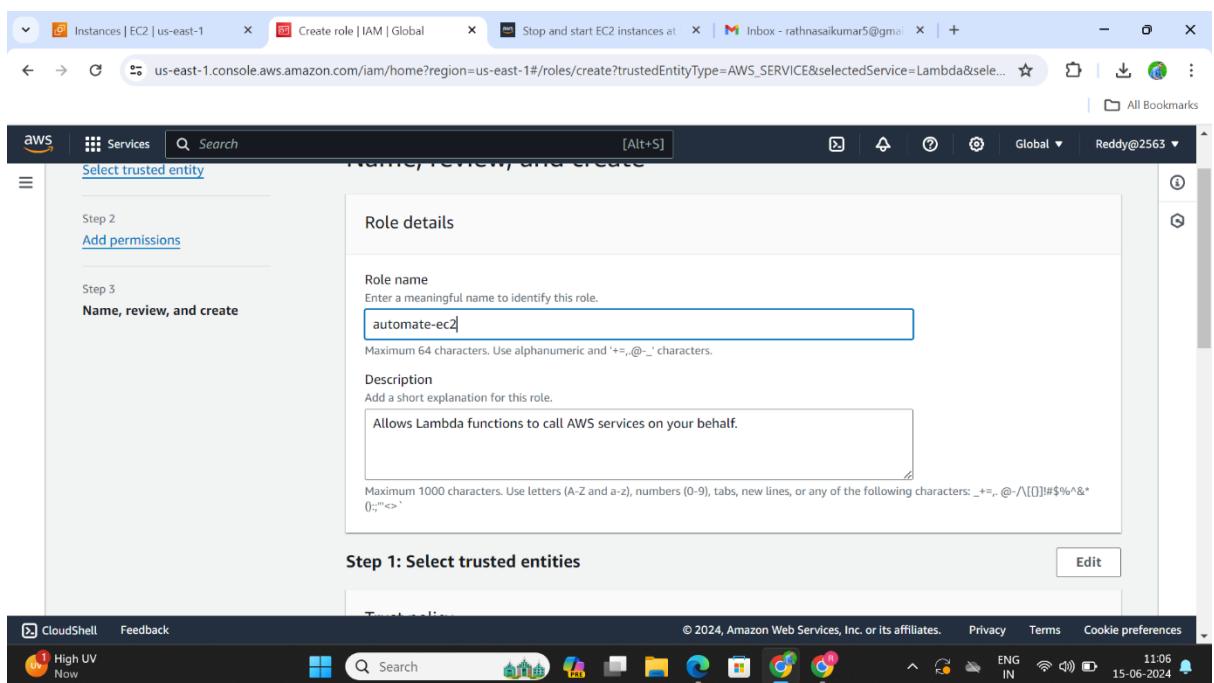
The permission only create the polices

- AWS provides a policy generator and an editor for creating policies. Choose one based on your familiarity and needs

- **Policy Generator:** Use predefined templates or a policy wizard to create policies.
- **JSON Editor:** Write your policy in JSON format for more control and customization.

● IAM ROLE NAME

Give the IAM role name



Create the role name in any name own create the after next

The screenshot shows the AWS IAM console interface for creating a new role. In Step 1: Select trusted entities, a JSON trust policy is displayed:

```

1  {
2    "Version": "2012-10-17",
3    "Statement": [
4      {
6        "Effect": "Allow",
7        "Action": [
8          "sts:AssumeRole"
9        ],
10       "Principal": {
11         "Service": [
12           "lambda.amazonaws.com"
13         ]
14       }
15     ]
16   }

```

In Step 2: Add permissions, a 'Permissions policy summary' section is shown.

The screenshot shows the AWS IAM Roles list page. The 'automate-ec2' role is selected, indicated by a checked checkbox. Other roles listed include:

- Amazon_EventBridge_Scheduler_LAMBDA_248a495407
- automate-ec2
- AWSServiceRoleForElasticLoadBalancing
- AWSServiceRoleForSupport
- AWSServiceRoleForTrustedAdvisor
- Ec2.amazonaws

• LAMDA FUNCTION

Create Function:

- Click "Create function".

- Choose "Author from scratch".
- Enter function name, select runtime (e.g., Python 3.10).

Configure Function:

- Optionally, set IAM role, adjust memory, timeout.

Write Code:

- Paste or write your Lambda function code.

Deploy:

- Click "Deploy" to create your Lambda function.

Test :

- Test function using Lambda console

Monitor using AWS CloudWatch.

The screenshot shows the AWS Lambda home page. The main heading is "AWS Lambda" with the subtext "lets you run code without thinking about servers.". Below this, a paragraph explains the cost model: "You pay only for the compute time that you consume — there is no charge when your code is not running. With Lambda, you can run code for virtually any type of application or backend service, all with zero administration." To the right, there is a "Get started" section with a "Create a function" button. At the bottom, there is a "How it works" section with a "Run" button and a link to "Next: Lambda responds to events". The browser header includes tabs for Instances | EC2 | us-east-1, Roles | IAM | Global, AWS Lambda | Lambda, Stop and start EC2 instances, and Inbox - rathnasai Kumar5.

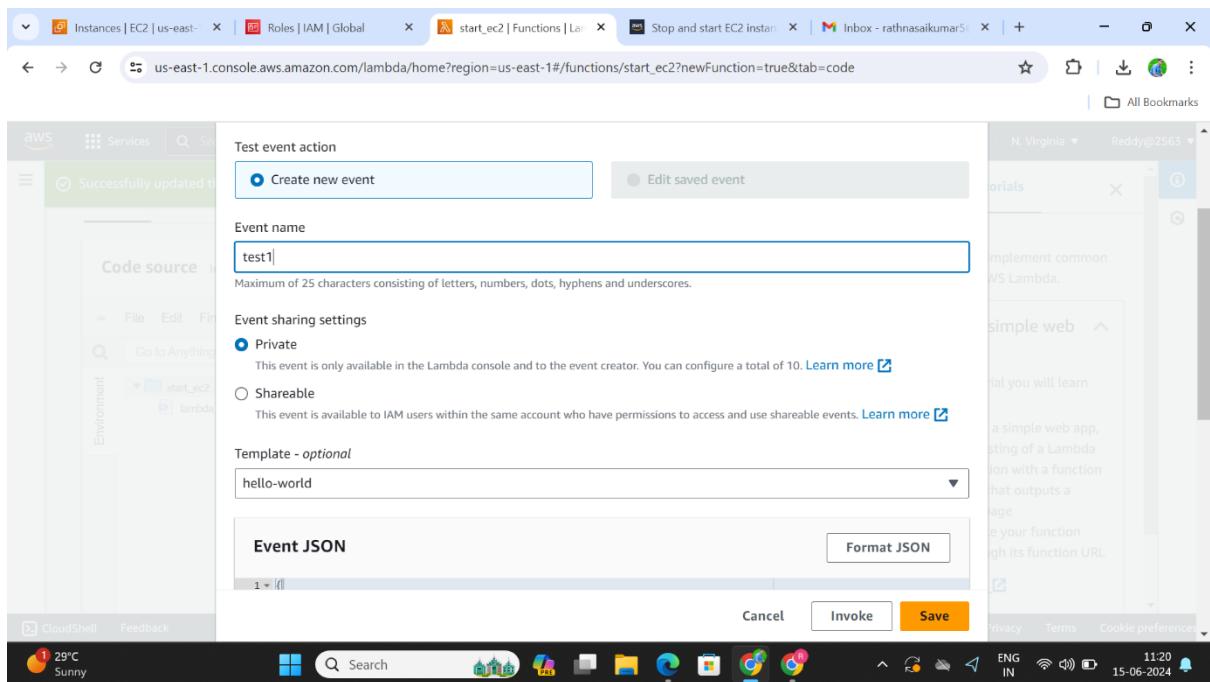
The screenshot shows the "Create function" wizard. It starts with a choice between "Author from scratch", "Use a blueprint", and "Container image". The "Author from scratch" option is selected. The "Basic information" step follows, where the user can enter the function name ("start_ec2") and choose the runtime ("Python 3.10"). On the right, a sidebar titled "Tutorials" provides a guide for creating a simple web app using Lambda. The browser header includes tabs for Instances | EC2 | us-east-1, Roles | IAM | Global, Create function | Function, Stop and start EC2 instances, and Inbox - rathnasai Kumar5.

The screenshot shows the AWS Lambda console interface. A green success message at the top states: "Successfully created the function start_ec2. You can now change its code and configuration. To invoke your function with a test event, choose 'Test'." Below this, the function name "start_ec2" is displayed. The "Function overview" tab is selected, showing a diagram with a single Lambda function block labeled "start_ec2". The "Description" field is empty. The "Last modified" field shows "6 seconds ago". The "Function ARN" field contains "arn:aws:lambda:us-east-1:533267448465:function:start_ec2". On the right side, there is a "Tutorials" sidebar with a section titled "Create a simple web app" and a "Learn more" link.

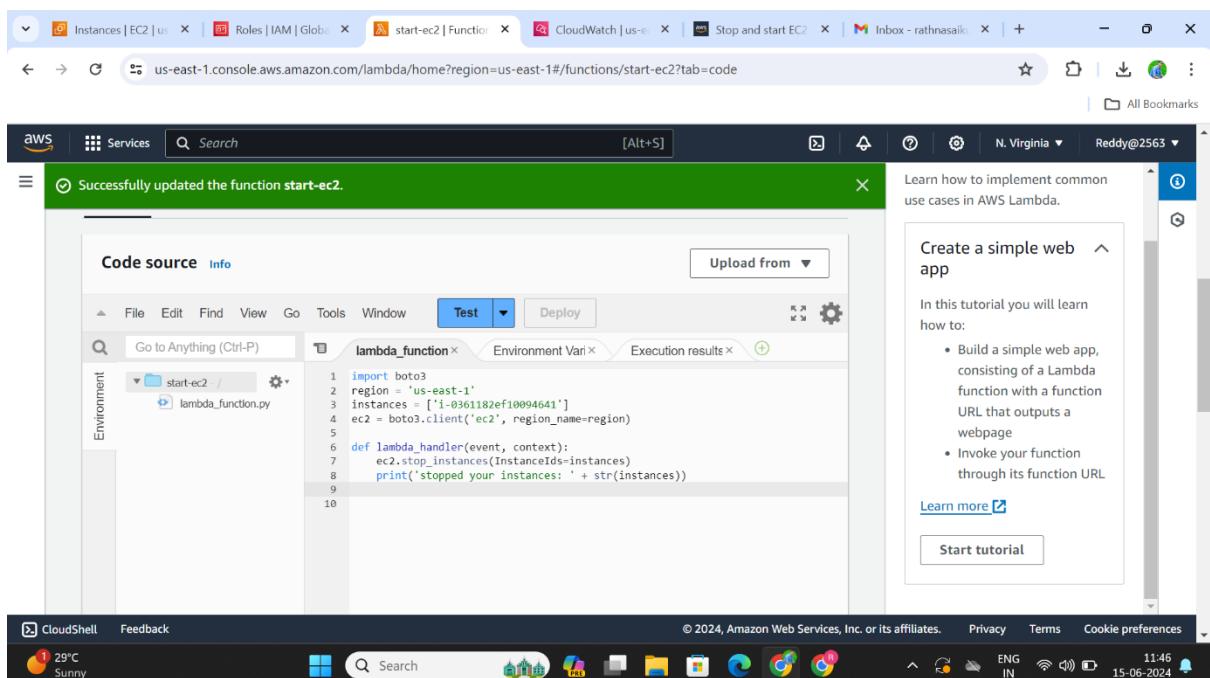
The screenshot shows the AWS Lambda console interface after updating the function. A green success message at the top states: "Successfully updated the function start_ec2." The "Code source" tab is selected, showing the function code in a code editor. The code is as follows:

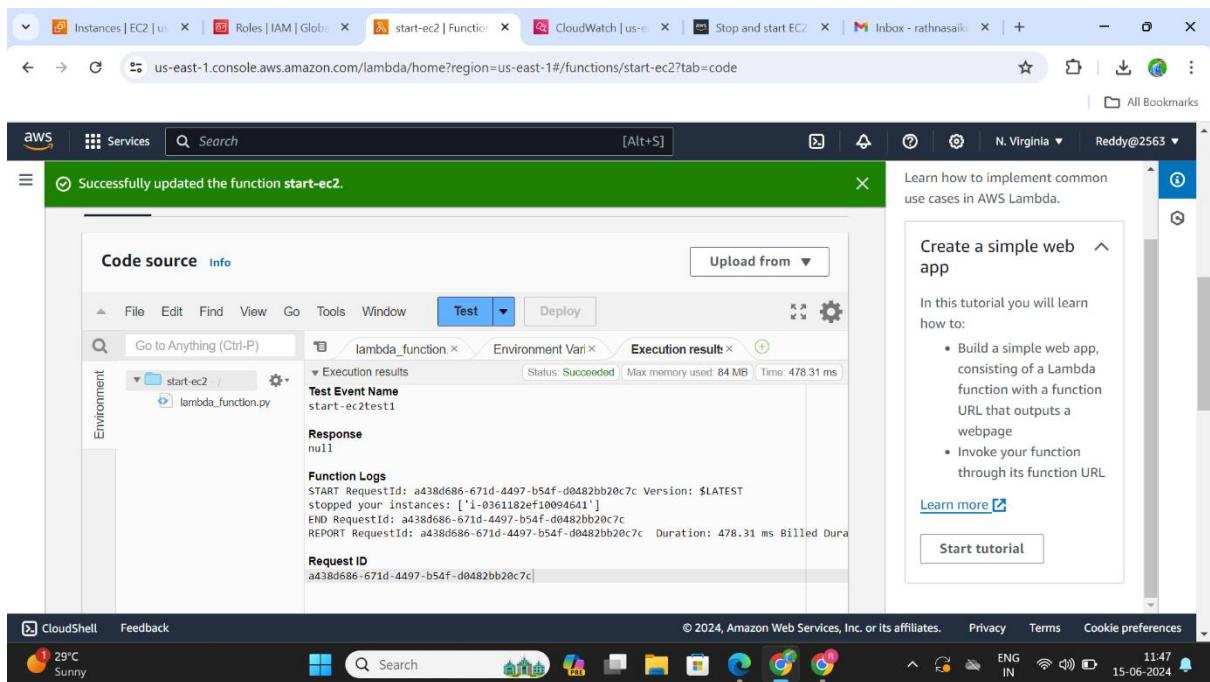
```
1 import boto3
2 region = 'us-east-1'
3 instances = ['i-0361182ef10094641']
4 ec2 = boto3.client('ec2', region_name=region)
5
6 def lambda_handler(event, context):
7     ec2.stop_instances(InstanceIds=instances)
8     print('stopped your instances: ' + str(instances))
```

The "Tutorials" sidebar remains the same as in the previous screenshot.

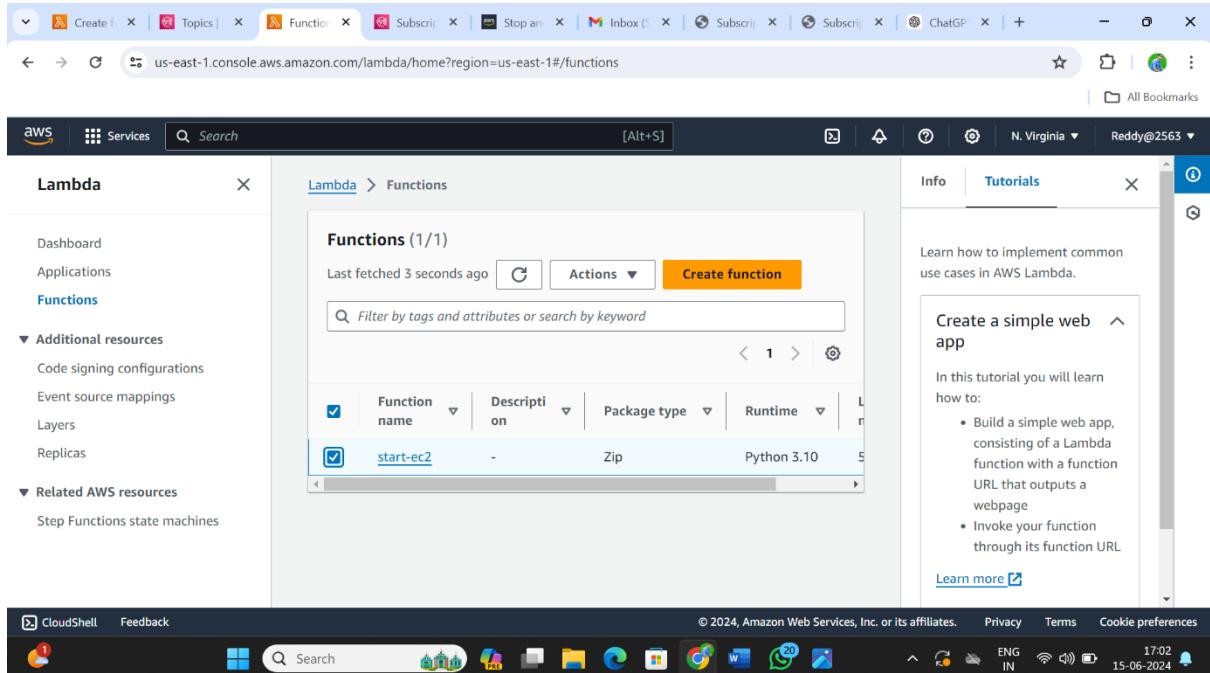


Test1event name create the running execution test code successfully





Execution results in output is working.



• CLOUDWATCH:-

Creation give the cloudwatch dashboard if use for the events and schedules

After the events in rules create the role name ‘automate ec2’

Rule event with an pattern

Next after event service “EC2” select option

After next event type “EC2 instance state -change notification

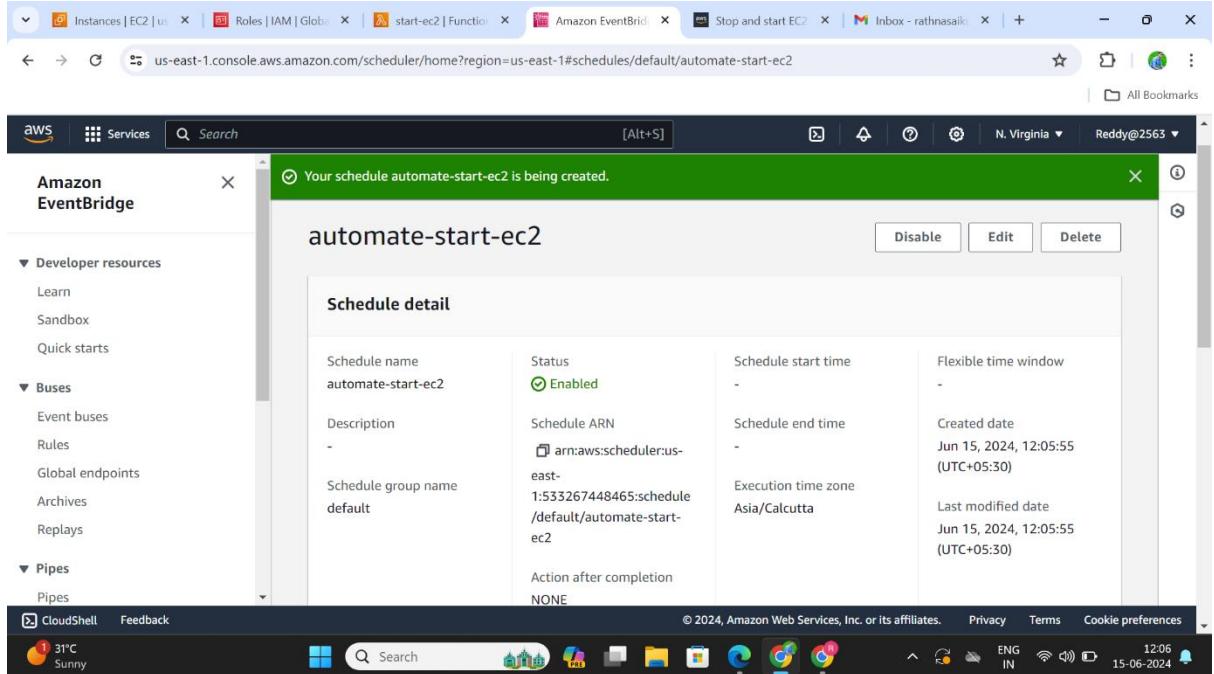
After select the a target “SNS Topic”

The previous create the topic “snstopic_create after the successfully for notification in ec2 instance refresh automatically stopped to running etc.

afternoon 12:30pm start instance and end closed

17:30pm automatically stopped in lambda function.

Target the group in schedule.



The screenshot shows the AWS EventBridge Scheduler console. A modal window is open with the message: "Your schedule automate-start-ec2 is being created." Below the message, there is a brief description: "Lambda and EventBridge operation on a schedule. triggers for use cases that do not require precise scheduled invocation of targets." The main area displays a table titled "Schedules (1/1)" with one row. The row details are:

Schedule name	Schedule group	Status	Target	Target type	Last modified
automate-start-ec2	default	Enabled	start-ec2	LAMBDA_Invoke	Jun 15, 2024, 06:35:55 (UTC+00:00)

At the bottom of the modal, there is a "Copy cron expression" button.

Below the modal, the main interface shows the cron expression for the schedule: "30 12 * * ? *".

Further down, a section titled "Next 10 trigger date" lists the following dates:

- Sat, 15 Jun 2024 12:30:00 (UTC+05:30)
- Sun, 16 Jun 2024 12:30:00 (UTC+05:30)
- Mon, 17 Jun 2024 12:30:00 (UTC+05:30)
- Tue, 18 Jun 2024 12:30:00 (UTC+05:30)
- Wed, 19 Jun 2024 12:30:00 (UTC+05:30)
- Thu, 20 Jun 2024 12:30:00 (UTC+05:30)
- Fri, 21 Jun 2024 12:30:00 (UTC+05:30)
- Sat, 22 Jun 2024 12:30:00 (UTC+05:30)
- Sun, 23 Jun 2024 12:30:00 (UTC+05:30)
- Mon, 24 Jun 2024 12:30:00 (UTC+05:30)

● SNS TOPIC:-

Create service in the amazon SNS

- Click on "Create topic" button.
- Enter a name for your SNS topic (e.g., MyTopic).

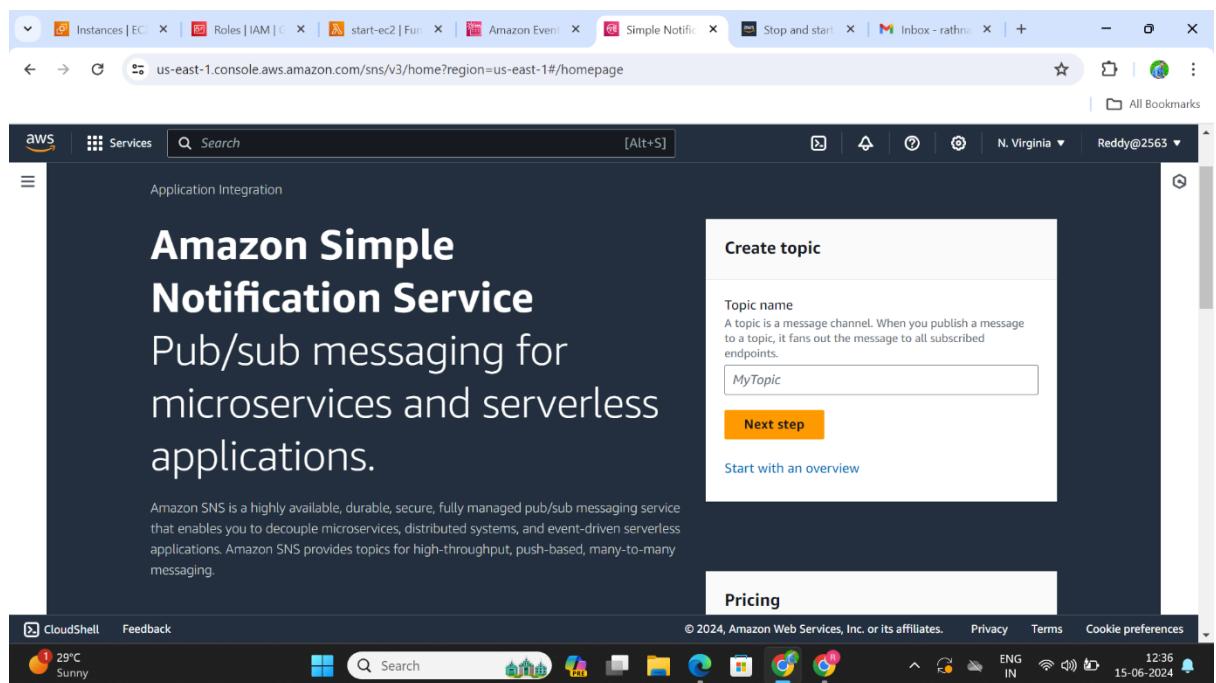
Display name optional: Optionally, provide a display name for your topic.

Optionally, select a KMS key to encrypt messages published to the topic.

Access Policy (Optional): Optionally, define a policy to control access to the topic.

Click on "Create topic" to finalize and create your SNS topic

Once created, you can view details such as ARN (Amazon Resource Name) and manage subscriptions under the "Subscriptions" tab
Go to Gmail verification subscriptions verify after select is email only if will come the notification in mail registered which have the mail if will come notification.



Screenshot of the AWS CloudShell interface showing the creation of an SNS topic named "SNSTopic_create".

The browser address bar shows: us-east-1.console.aws.amazon.com/sns/v3/home?region=us-east-1#/topic/arm:aws:sns:us-east-1:533267448465:SNSTopic_create

The AWS navigation bar includes: Instances | EC2, Roles | IAM, start-ec2 | Functions, Amazon EventBridge, SNSTopic_create, Stop and start, Inbox - rathna, and All Bookmarks.

The main content area displays the "Amazon SNS" service with the "Topics" section selected. A success message states: "Topic SNSTopic_create created successfully. You can create subscriptions and send messages to them from this topic." It includes a "Publish message" button and links to "Edit", "Delete", and "Publish message".

The left sidebar shows navigation options: Dashboard, Topics (selected), Subscriptions, Mobile (Push notifications, Text messaging (SMS), Origination numbers).

Screenshot of the AWS CloudShell interface showing the creation of a new SNS topic.

The browser address bar shows: us-east-1.console.aws.amazon.com/sns/v3/home?region=us-east-1#/create-topic

The AWS navigation bar includes: Instances | EC2, Roles | IAM, start-ec2 | Functions, Amazon EventBridge, Create topic, Stop and start, Inbox - rathna, and All Bookmarks.

The main content area shows the "Create topic" wizard. Under the "Type" section, "Info" is selected. It notes: "Topic type cannot be modified after topic is created". Two options are shown: "FIFO (first-in, first-out)" and "Standard".

- FIFO (first-in, first-out)**
 - Strictly-preserved message ordering
 - Exactly-once message delivery
 - High throughput, up to 300 publishes/second
 - Subscription protocols: SQS
- Standard**
 - Best-effort message ordering
 - At-least once message delivery
 - Highest throughput in publishes/second
 - Subscription protocols: SQS, Lambda, HTTP, SMS, email, mobile application endpoints

Screenshot of the AWS CloudShell interface showing the continuation of the SNS topic creation process.

The browser address bar shows: us-east-1.console.aws.amazon.com/sns/v3/home?region=us-east-1#/create-topic

The AWS navigation bar includes: Instances | EC2, Roles | IAM, start-ec2 | Functions, Amazon EventBridge, Create topic, Stop and start, Inbox - rathna, and All Bookmarks.

The main content area continues the "Create topic" wizard. It shows the "Name" field set to "SNSTopic_create" and the "Display name" field set to "SNS.create".

Information below the fields notes: "Maximum 256 characters. Can include alphanumeric characters, hyphens (-) and underscores (_)." and "To use this topic with SMS subscriptions, enter a display name. Only the first 10 characters are displayed in an SMS message."

Topic ARN
arn:aws:sns:us-east-1:533267448465:SNStopic_create

Protocol
The type of endpoint to subscribe
Email

Endpoint
An email address that can receive notifications from Amazon SNS.
rathanasikumar5@gmail.com

CloudShell Feedback © 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences 29°C Sunny 12:38 15-06-2024

Subscription confirmed is gmail id

Subscription confirmed!
You have successfully subscribed.
Your subscription's ID is:
arn:aws:sns:us-east-1:533267448465:SNStopic_create:2334f0f4-437a-46ec-b13d-ad12e0bb735
If it was not your intention to subscribe, [click here to unsubscribe](#).

CloudShell Feedback © 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences 29°C Sunny 12:39 15-06-2024

The screenshot shows the AWS SNS Subscriptions page. On the left, there's a sidebar with links like Dashboard, Topics, Subscriptions, Mobile (Push notifications, Text messaging (SMS), Origination numbers), and a CloudShell link. The main area has a title "Amazon SNS" and a "Details" section. In the "Details" section, it shows the ARN (arn:aws:sns:us-east-1:533267448465:SNStopic_create:2334f0f4-437a-46ec-b13d-ad12e0bb7a35), Status (Confirmed), Protocol (EMAIL), Endpoint (rathnasaiukumar5@gmail.com), Topic (SNStopic_create), and Subscription Principal (arn:aws:iam::533267448465:root). Below this, there are tabs for "Subscription filter policy" and "Redrive policy (dead-letter queue)". The browser status bar at the bottom shows the date (15-06-2024) and time (12:39).

Amazon SNS

ARN
arn:aws:sns:us-east-1:533267448465:SNStopic_create:2334f0f4-437a-46ec-b13d-ad12e0bb7a35

Status
Confirmed

Protocol
EMAIL

Endpoint
rathnasaiukumar5@gmail.com

Topic
SNStopic_create

Subscription Principal
arn:aws:iam::533267448465:root

Subscription filter policy | Redrive policy (dead-letter queue)

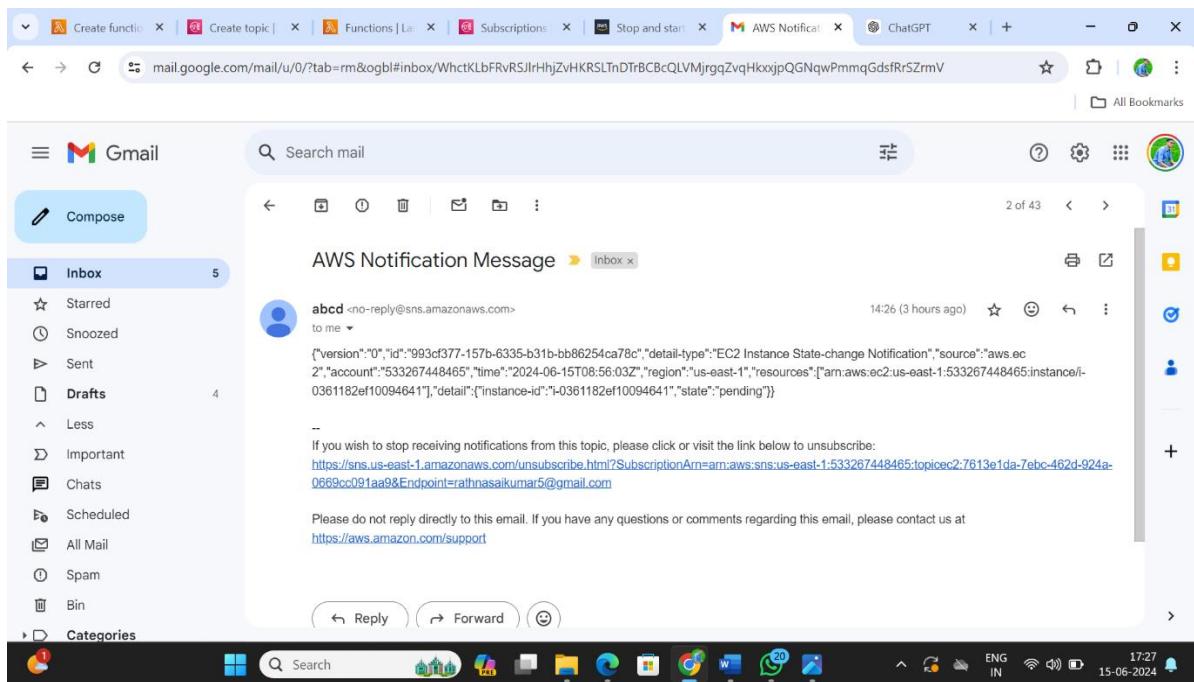
CloudShell | Feedback | © 2024, Amazon Web Services, Inc. or its affiliates. | Privacy | Terms | Cookie preferences | ENG IN | 15-06-2024 | 12:39

The screenshot shows the AWS SNS Subscriptions page with the following details:

ID	Endpoint	Status	Protocol	Topic
2334f0f4-437a-4...	rathnasaikumar5...	Confirmed	EMAIL	SNStopic_create
7613e1da-7ebc-...	rathnasaikumar5...	Confirmed	EMAIL	topicsec2
Deleted	ratnasaikumar41...	Confirmed	EMAIL	simplenotificatio...

Topicsec2 if will came notification Gmail Id ec2 lunch stopped for the start instance in running automatically the if will came to Gmail id notification.

Final if will came notification Gmail ID



Gmail is notification if came the initial state I will changed the start instance the verification will be completed finally.....

Project Done By

R. Sai Kumar

Submitted To

Ajay Sir(GA Foundation)