

5.Create a lambda function to start and stop ec2 instance at regular intervals

Step 1: Create an IAM Role for Lambda

- Log in to the AWS Management Console
- Navigate to the IAM dashboard
- Click "Roles" > "Create role"
- Select "AWS service" > "Lambda"
- Click "Next: Permissions"
- Attach the "json" policy by stoping and starting the ec2 instances or choose alternate way also
- Click "Next: Review"
- Name the role (e.g., "startstoppec21")
- Click "Create role"

Step 2: Create a Lambda Function

- Navigate to the Lambda dashboard
- Click "Create function"
- Select "Author from scratch"
- Name the function (e.g., "startstopinstance")
- Choose "Python 3.9" as the runtime
- Click "Create function"

Step 3: Configure Lambda Function Triggers

- Click "Add trigger"
 - Select "EventBridge (CloudWatch Events)"
 - Choose "Schedule" as the event source
 - Set the schedule to your desired interval (e.g., every 1 hour)
 - Click "Add"
 - here I created rate without using cron timing. Also create cron timing.
- When to use Rate:

1. Simple scheduling needs (e.g., every 5 minutes, daily).
2. Ease of understanding and maintenance is crucial.
3. You don't need precise control over scheduling.

When to use Cron:

1. Complex scheduling needs (e.g., specific times, intervals, and exclusions).
2. Precise control over scheduling is crucial.
3. You're comfortable with cron syntax and need the flexibility it provides.

Step 4: Write Lambda Function Code from Aws documentation

Code: To stop the ec2 instances

```
import boto3
region = 'ap-south-1'
instances = ['i-02b6c6ec82ddf2265']
ec2 = boto3.client('ec2', region_name=region)
```

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

```
print('stopped your instances: ' + str(instances))
```

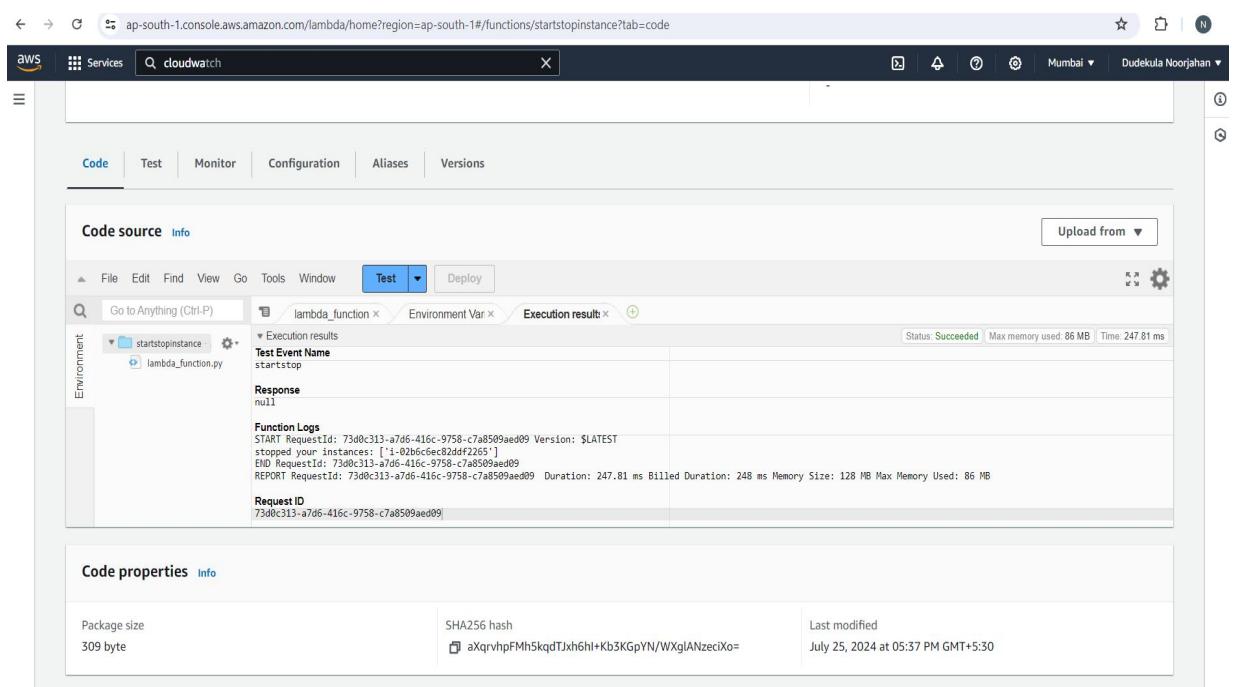
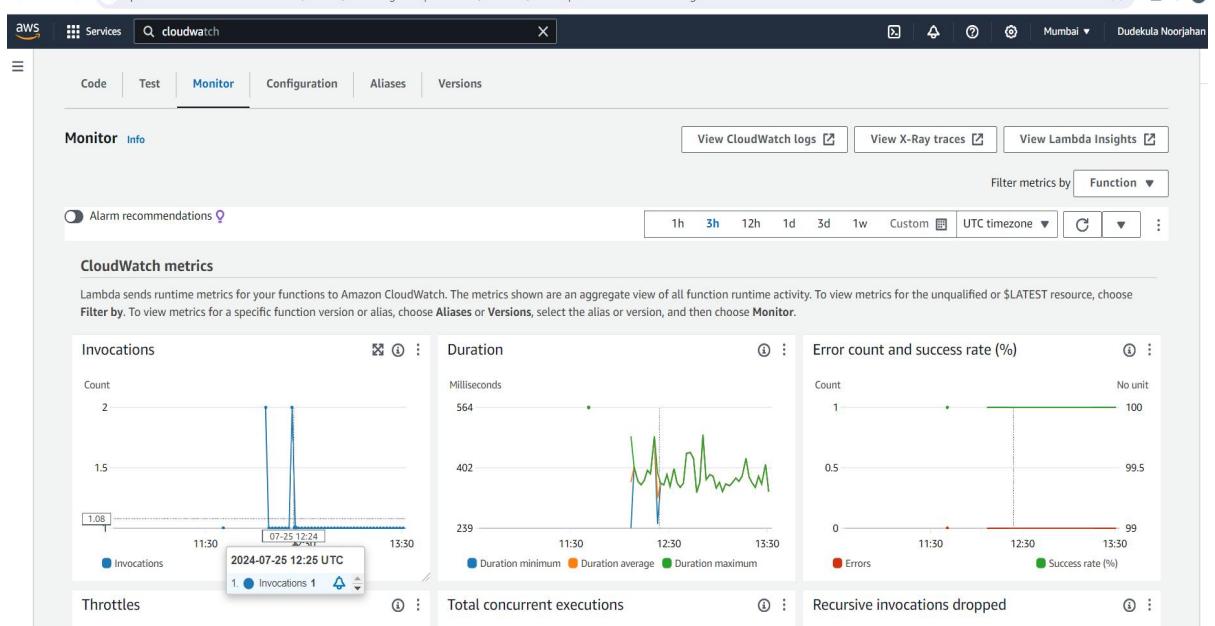
Code: To start the ec2 instances

```
import boto3
region = 'ap-south-1'
instances = ['i-02b6c6ec82ddf2265']
ec2 = boto3.client('ec2', region_name=region)
```

```
def lambda_handler(event, context):
    ec2.start_instances(InstanceIds=instances)
    Print('started your instances.'+str(instances))
```

The screenshot shows the AWS EC2 Instances page. On the left, there's a navigation sidebar with options like EC2 Dashboard, EC2 Global View, Events, Instances (selected), Instance Types, Launch Templates, Spot Requests, Savings Plans, Reserved Instances, Dedicated Hosts, Capacity Reservations, Images (AMIs, AMI Catalog), and Elastic Block Store (Volumes, Snapshots, Lifecycle Manager). The main area displays a table for 'Instances (1/1)'. The table has columns for Name, Instance ID, Instance state, Instance type, Status check, Alarm status, Availability Zone, and Public IPv4 DNS. One row is selected for 'instance1' (i-02b6c6ec82ddf2265), which is currently 'Running'. Below the table, a detailed view for 'i-02b6c6ec82ddf2265 (instance1)' is shown with tabs for Details, Status and alarms, Monitoring, Security, Networking, Storage, and Tags. Under the Details tab, it shows the Instance ID, Public IPv4 address (172.31.194.25), Instance state (Running), Private IP DNS name (ip-172-31-194-25.ap-south-1.compute.internal), and Instance type (t2.micro).

The screenshot shows the AWS Lambda function configuration page for 'startstopinstance'. At the top, it shows the Function ARN: arn:aws:lambda:ap-south-1:730335216686:function:startstopinstance and the Function URL: http://. The 'Code source' tab is selected, showing the code editor with the file 'lambda_function.py' containing the provided Python code. The 'Test' tab is also visible at the bottom.



Step 6: create for the Opposite Action to start or stop our ec2instances

- Create another CloudWatch event with the opposite schedule and action

That's it! Your Lambda function should now start and stop your EC2 instance at regular intervals.