**Assignment 1: Automated Instance Management Using AWS Lambda and Boto3**

**Objective:** In this assignment, you will gain hands-on experience with AWS Lambda and Boto3, Amazon's SDK for Python. You will create a Lambda function that will automatically manage EC2 instances based on their tags.

**Task:** You're tasked to automate the stopping and starting of EC2 instances based on tags. Specifically:

1. Setup:

   - Create two EC2 instances.

   - Tag one of them as `Auto-Stop` and the other as `Auto-Start`.

2. Lambda Function Creation:

   - Set up an AWS Lambda function.

   - Ensure that the Lambda function has the necessary IAM permissions to describe, stop, and start EC2 instances.

3. Coding:

   - Using Boto3 in the Lambda function:

     - Detect all EC2 instances with the `Auto-Stop` tag and stop them.

     - Detect all EC2 instances with the `Auto-Start` tag and start them.

4. Testing:

   - Manually invoke the Lambda function.

   - Confirm that the instance tagged `Auto-Stop` stops and the one tagged `Auto-Start` starts.

**Instructions:**

1. EC2 Setup:

   - Navigate to the EC2 dashboard and create two new t2.micro instances (or any other available free-tier type).

   - Tag the first instance with a key `Action` and value `Auto-Stop`.

   - Tag the second instance with a key `Action` and value `Auto-Start`.

2. Lambda IAM Role:

   - In the IAM dashboard, create a new role for Lambda.

   - Attach the `AmazonEC2FullAccess` policy to this role. (Note: In a real-world scenario, you would want to limit permissions for better security.)

3. Lambda Function:

   - Navigate to the Lambda dashboard and create a new function.

   - Choose Python 3.x as the runtime.

   - Assign the IAM role created in the previous step.

   - Write the Boto3 Python script to:

     1. Initialize a boto3 EC2 client.

     2. Describe instances with `Auto-Stop` and `Auto-Start` tags.

     3. Stop the `Auto-Stop` instances and start the `Auto-Start` instances.

     4. Print instance IDs that were affected for logging purposes.

4. Manual Invocation:

   - After saving your function, manually trigger it.

   - Go to the EC2 dashboard and confirm that the instances' states have changed according to their tags.

Project solution:

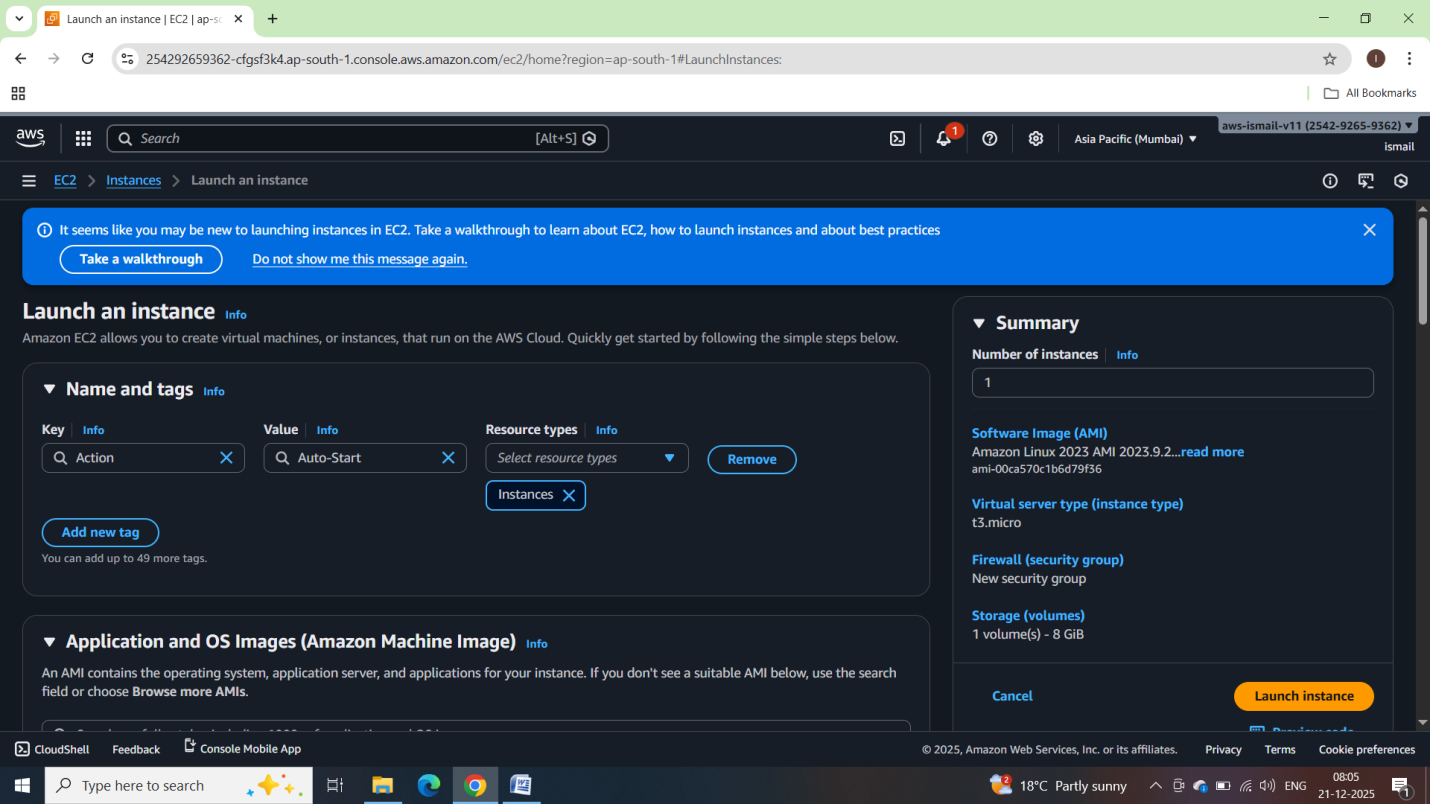
1. EC2 Setup:

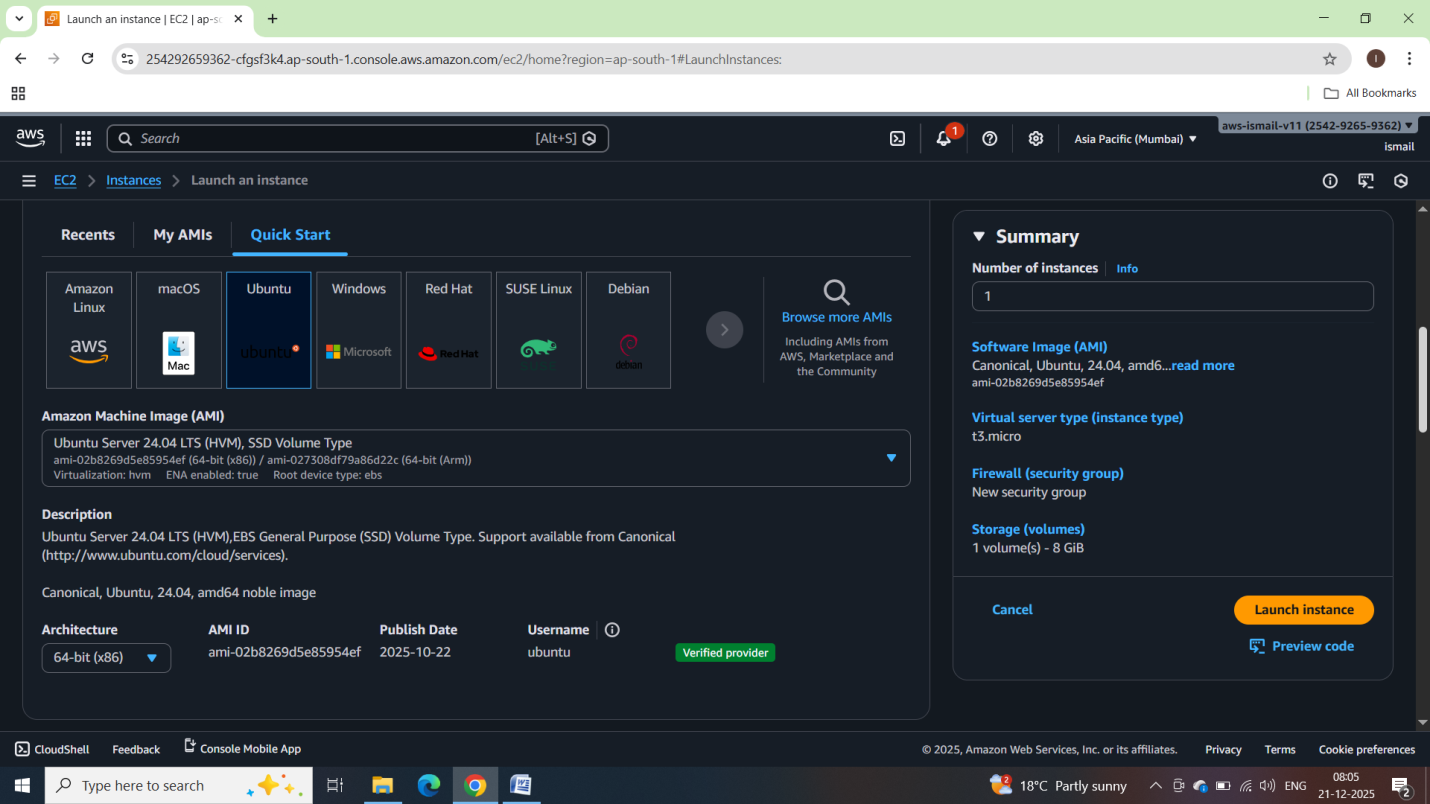
   - Navigate to the EC2 dashboard and create two new t2.micro instances (or any other available free-tier type).

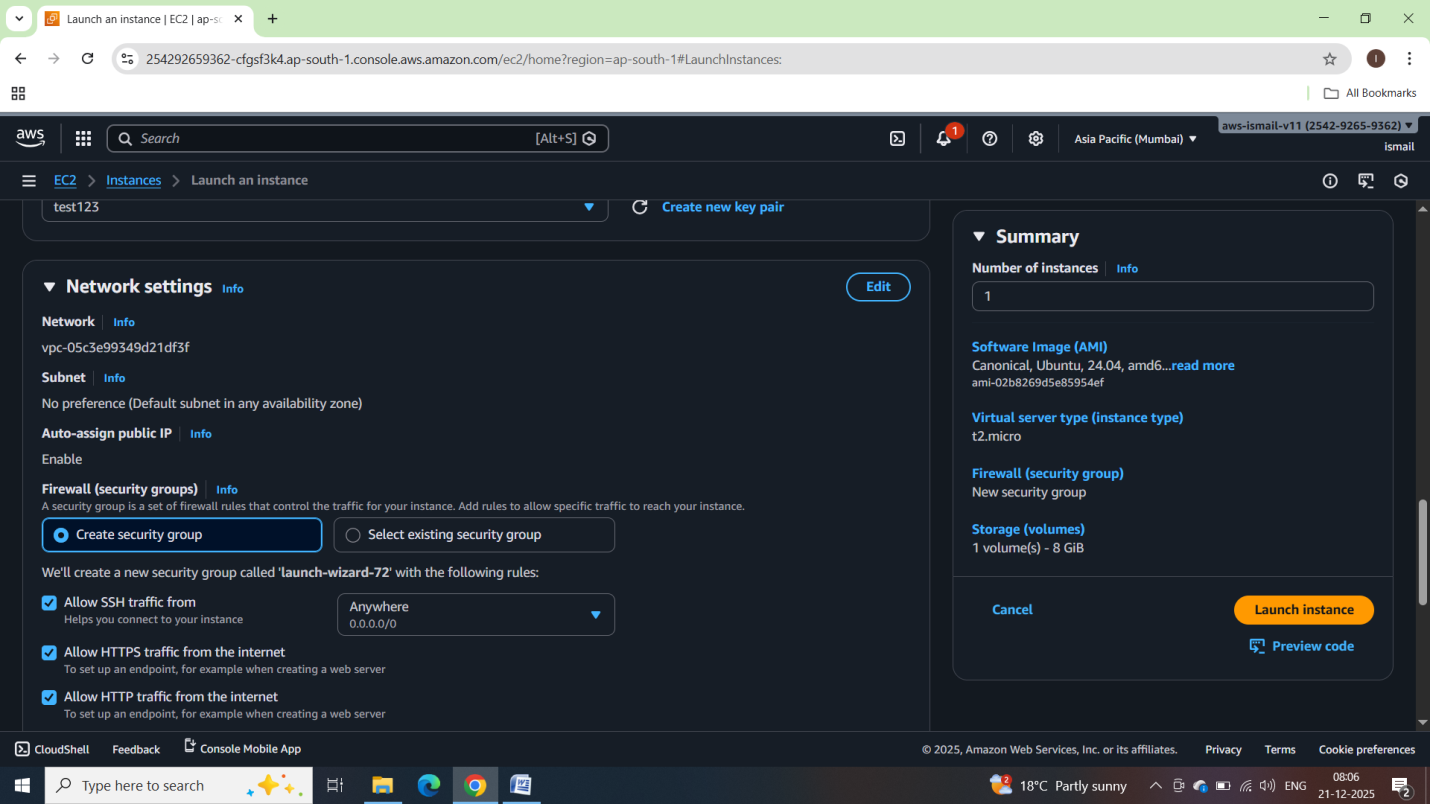
   - Tag the first instance with a key `Action` and value `Auto-Stop`.

   - Tag the second instance with a key `Action` and value `Auto-Start`.

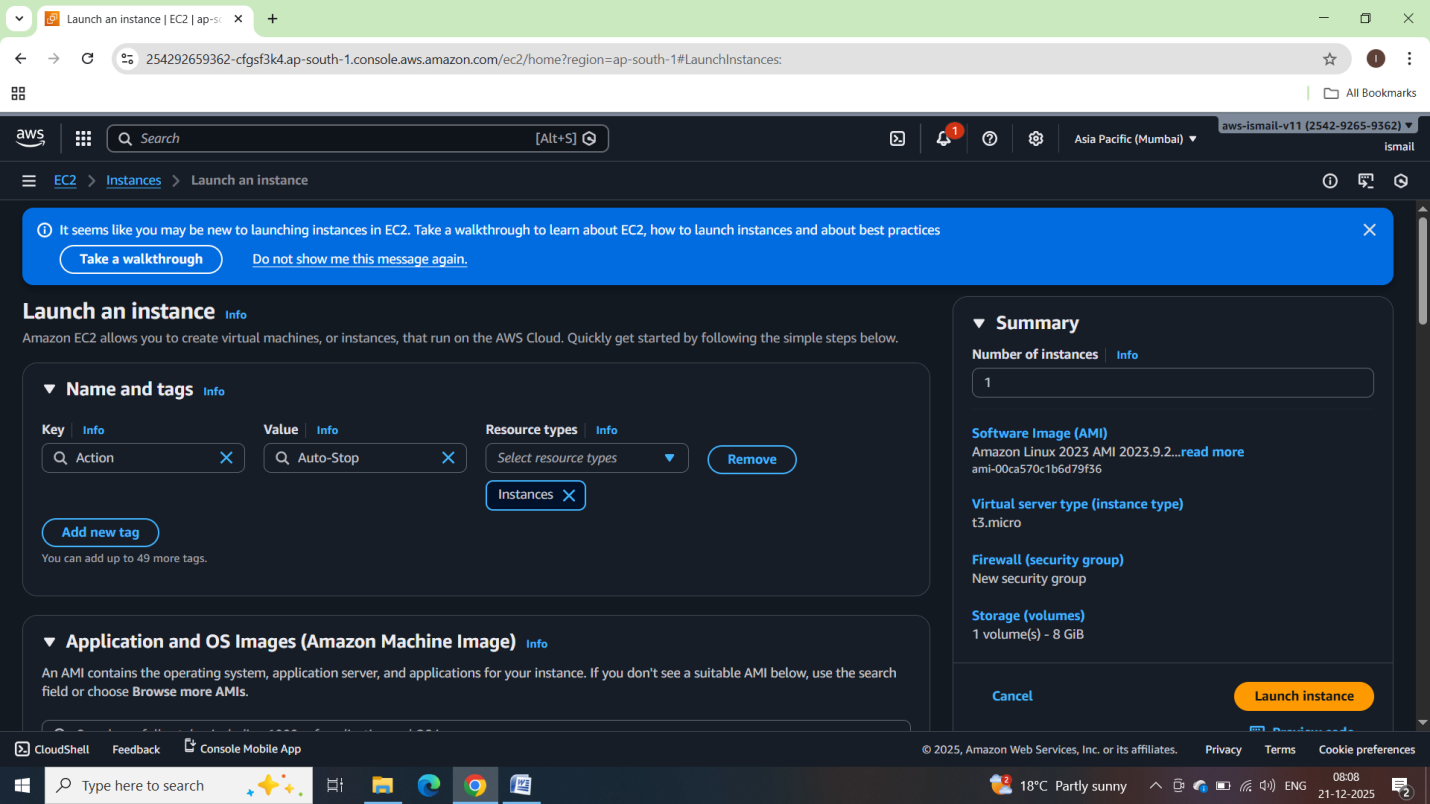
Auto-Start instance:

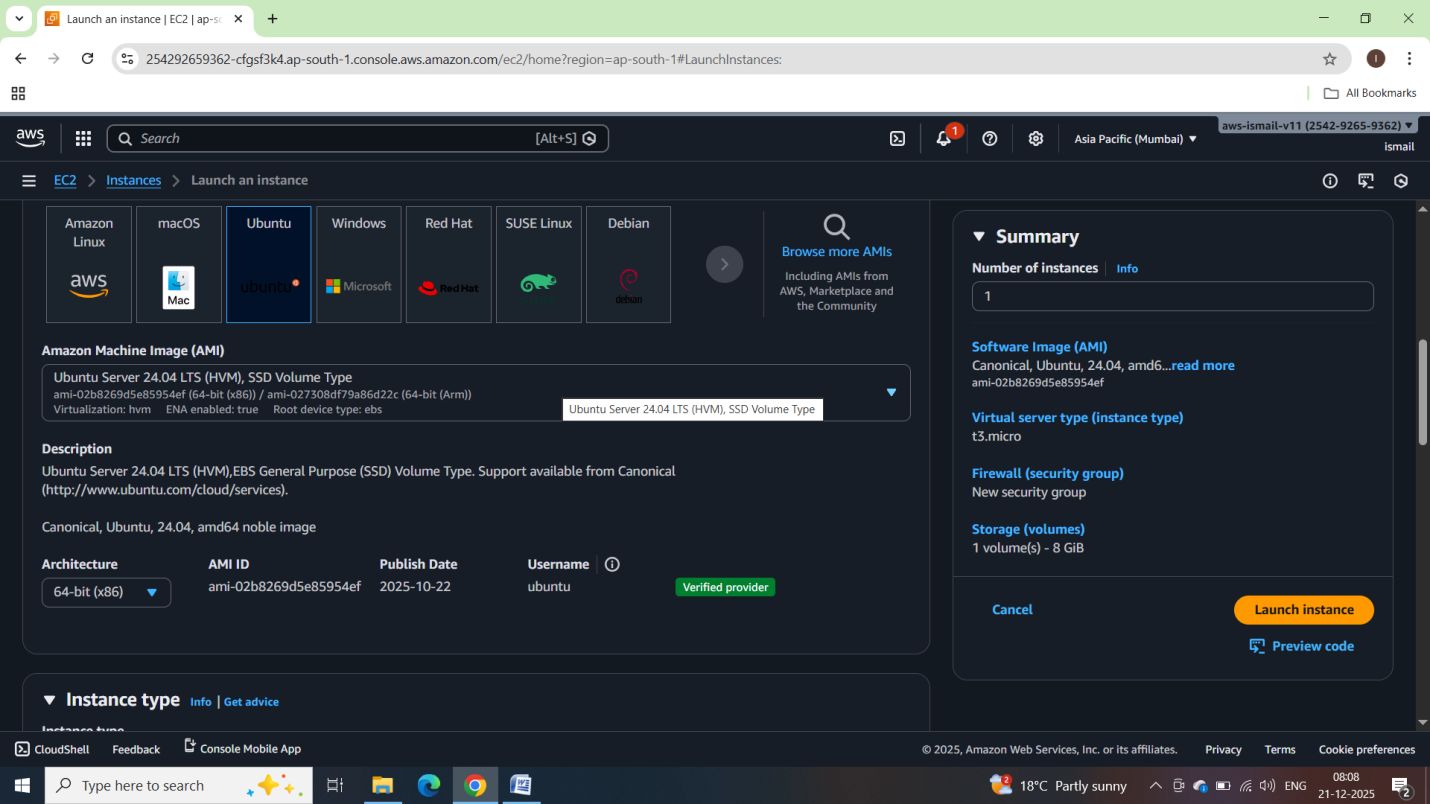


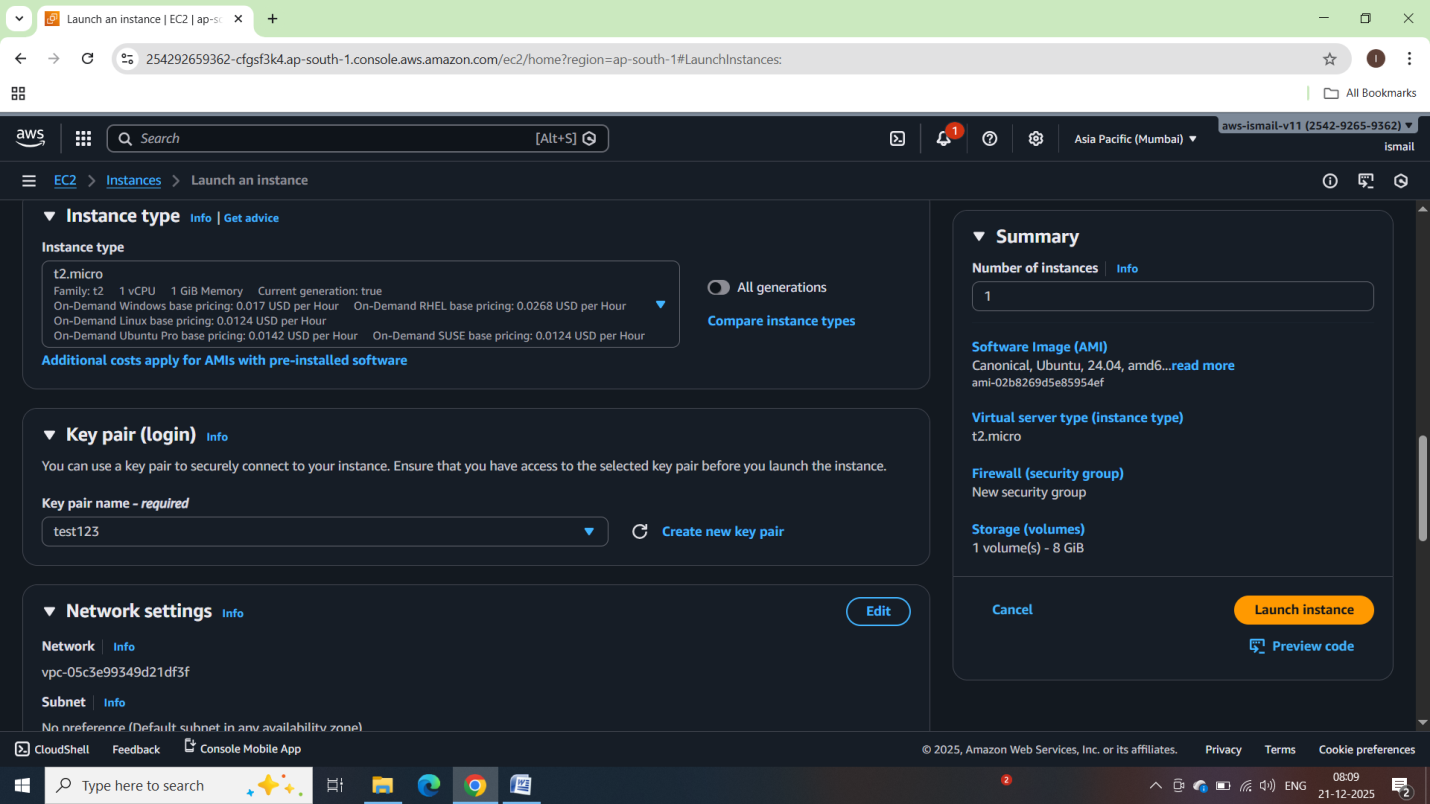


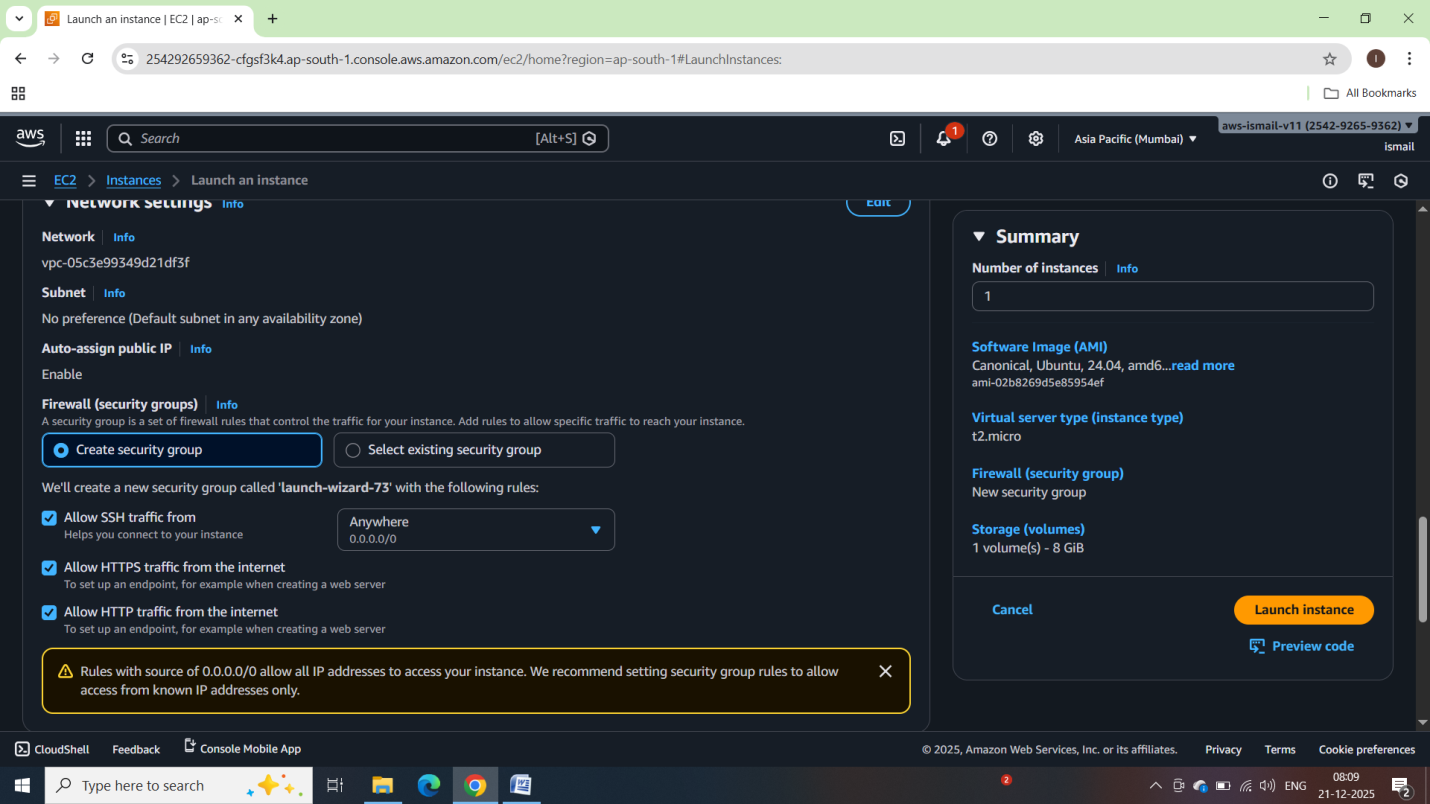


Auto-Stop Instance:

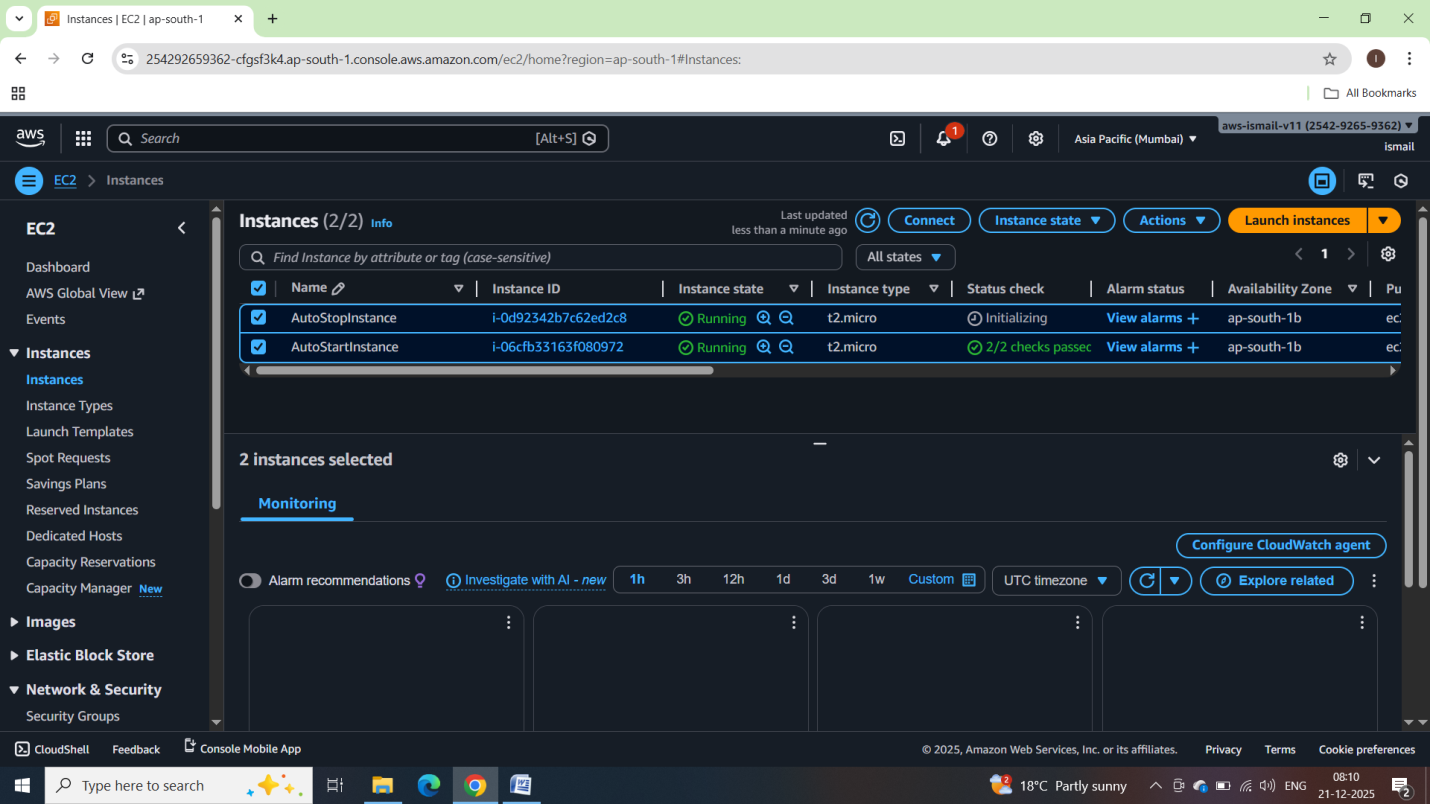








Now Both instances are up and running:

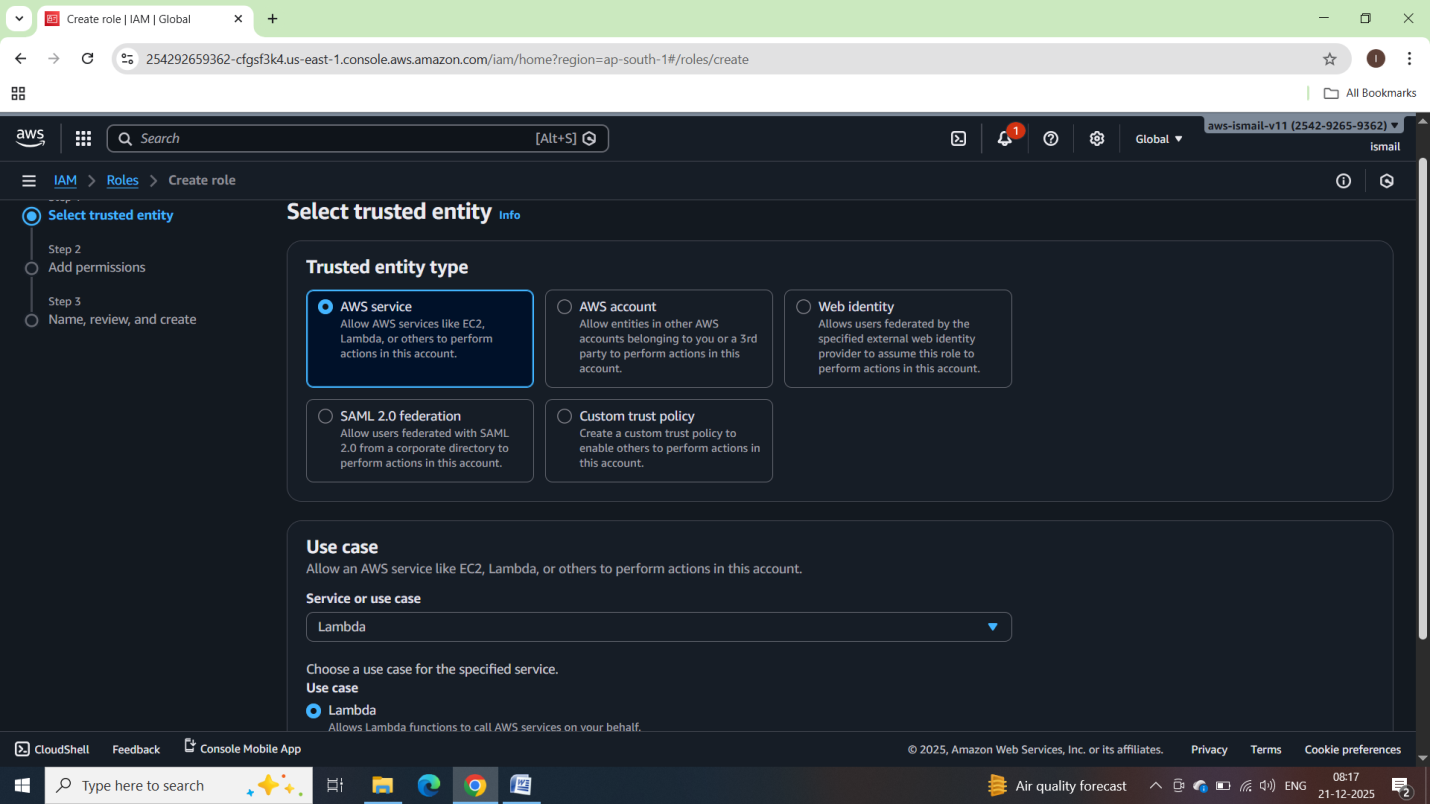


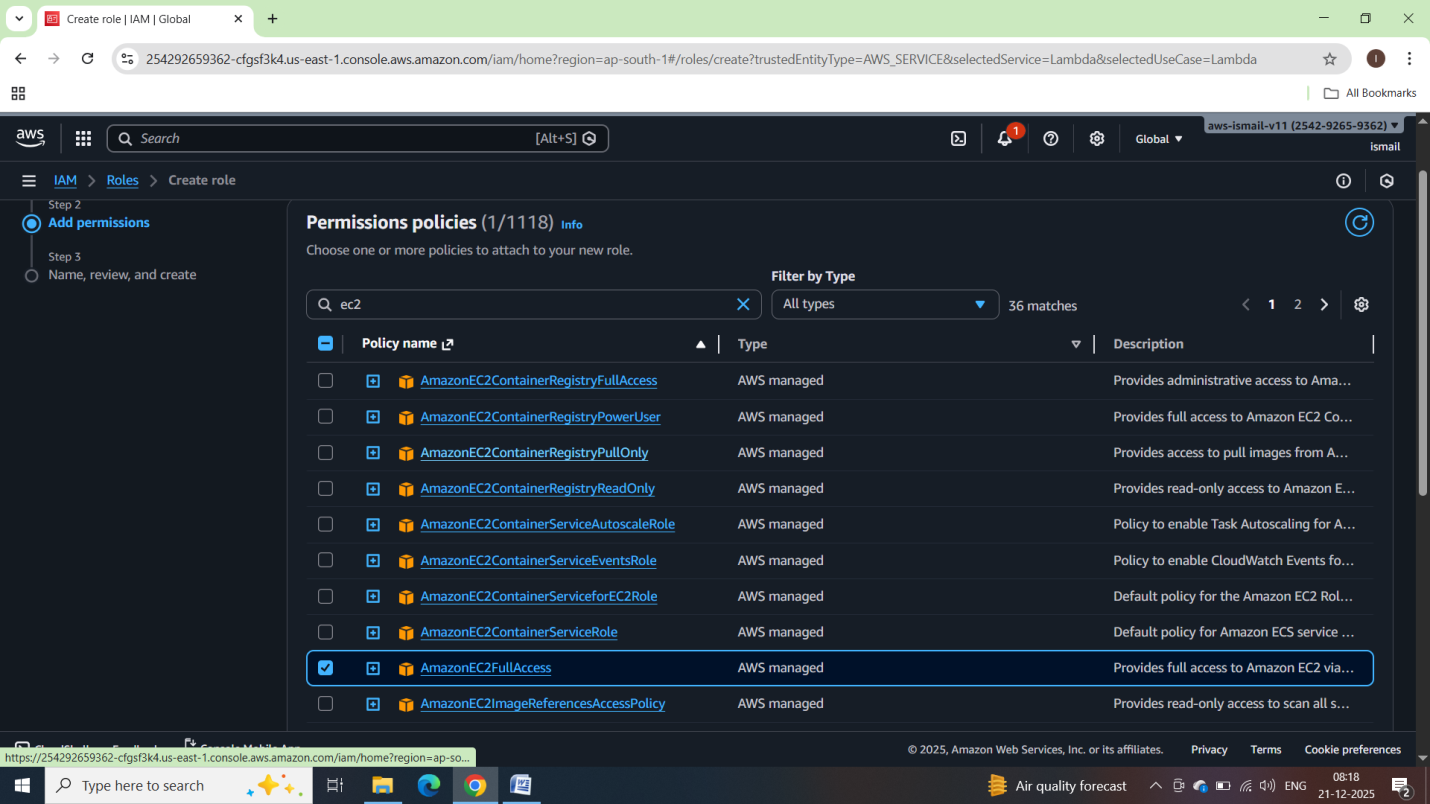
2. Lambda IAM Role:

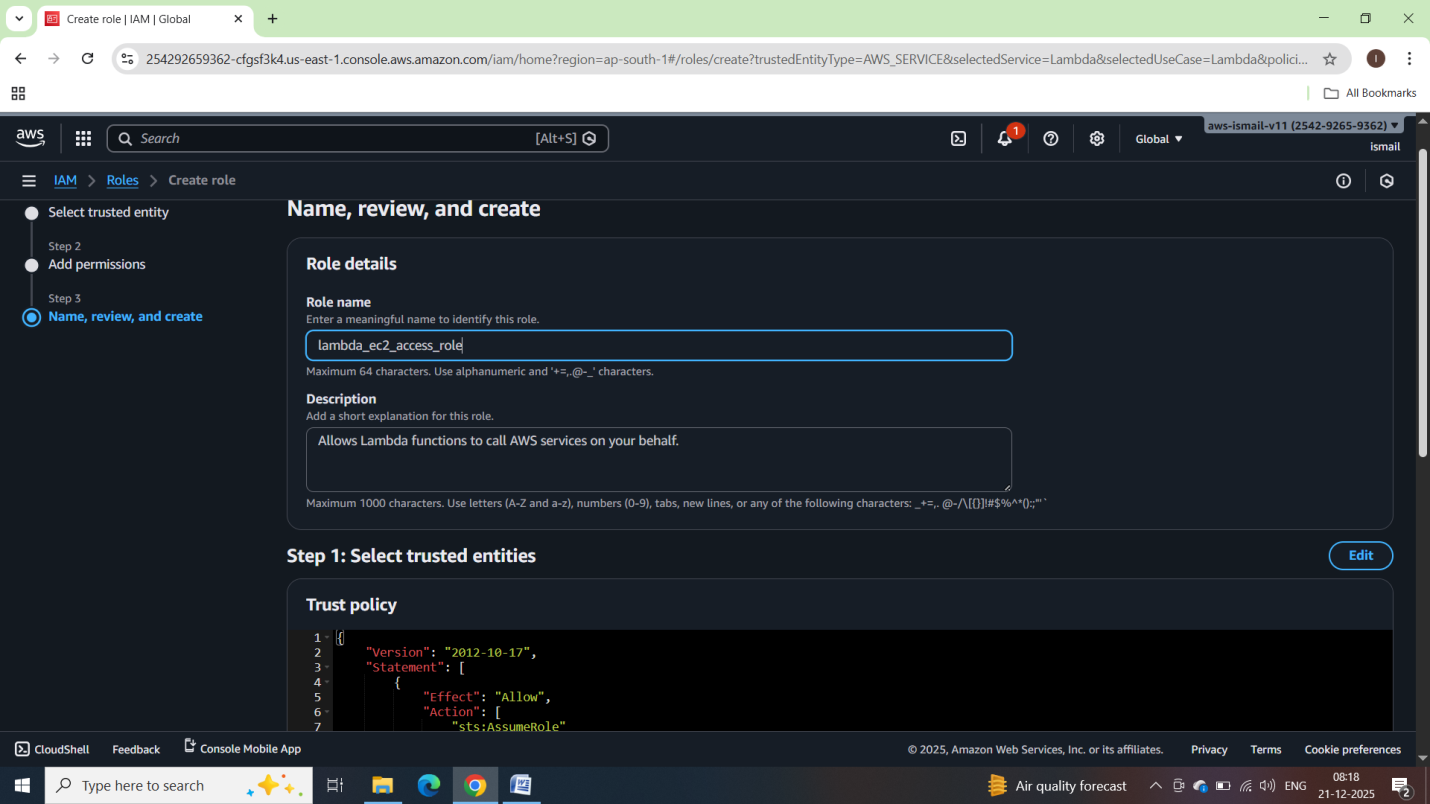
   - In the IAM dashboard, create a new role for Lambda.

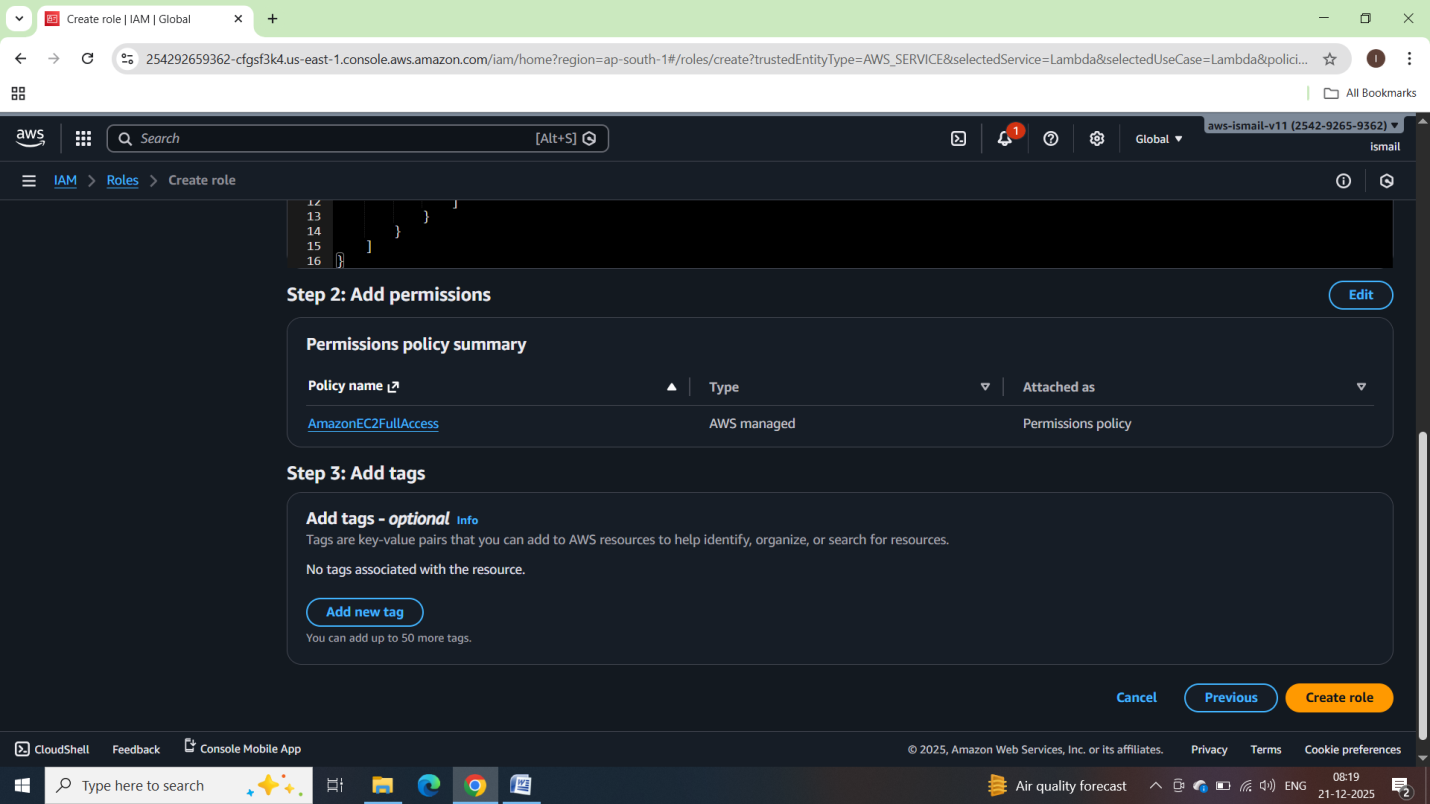
   - Attach the `AmazonEC2FullAccess` policy to this role. (Note: In a real-world scenario, you would want to limit permissions for better security.)

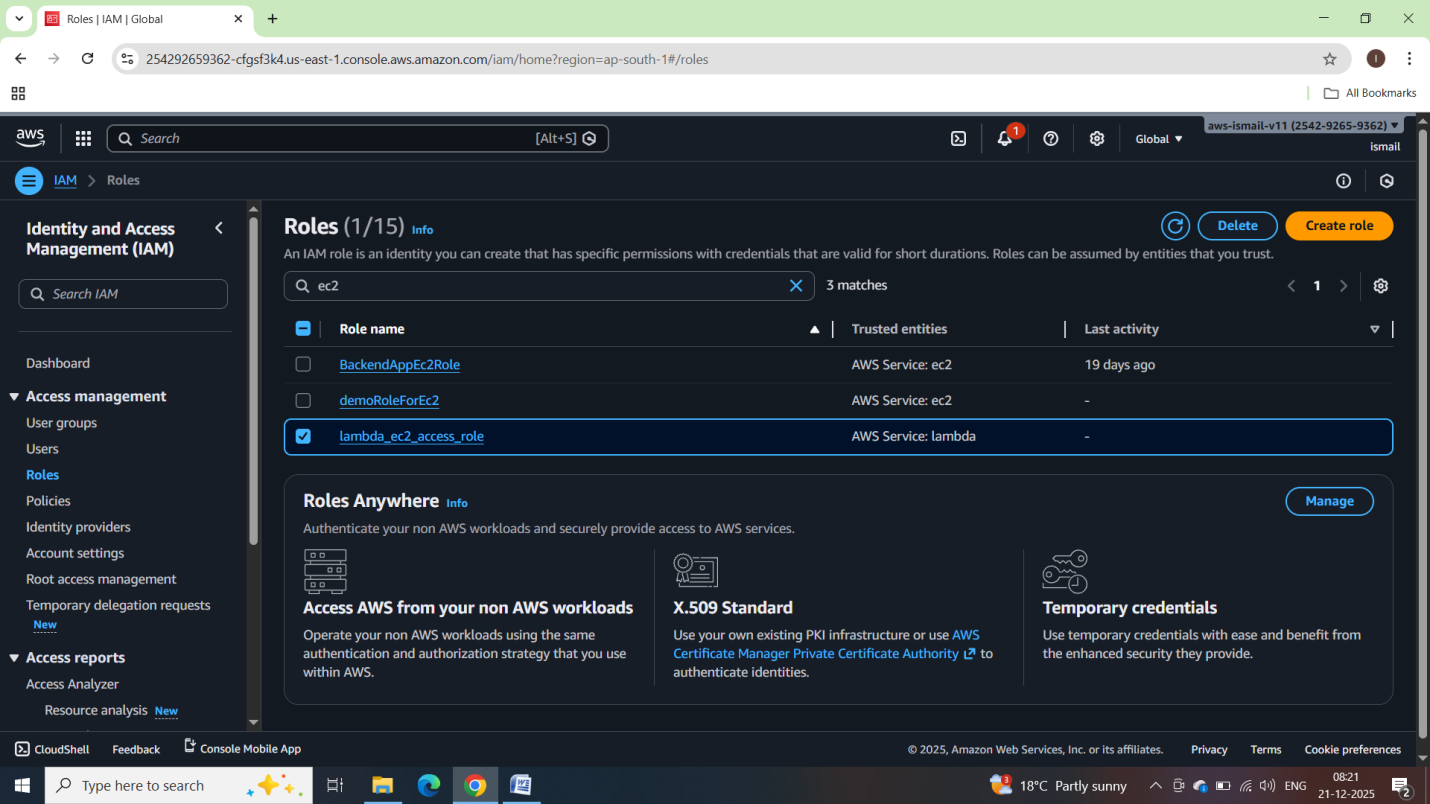
* Setting up IAM permission to describe,stop and start the EC2 Instances











3. Lambda Function:

   - Navigate to the Lambda dashboard and create a new function.

   - Choose Python 3.x as the runtime.

   - Assign the IAM role created in the previous step.

   - Write the Boto3 Python script to:

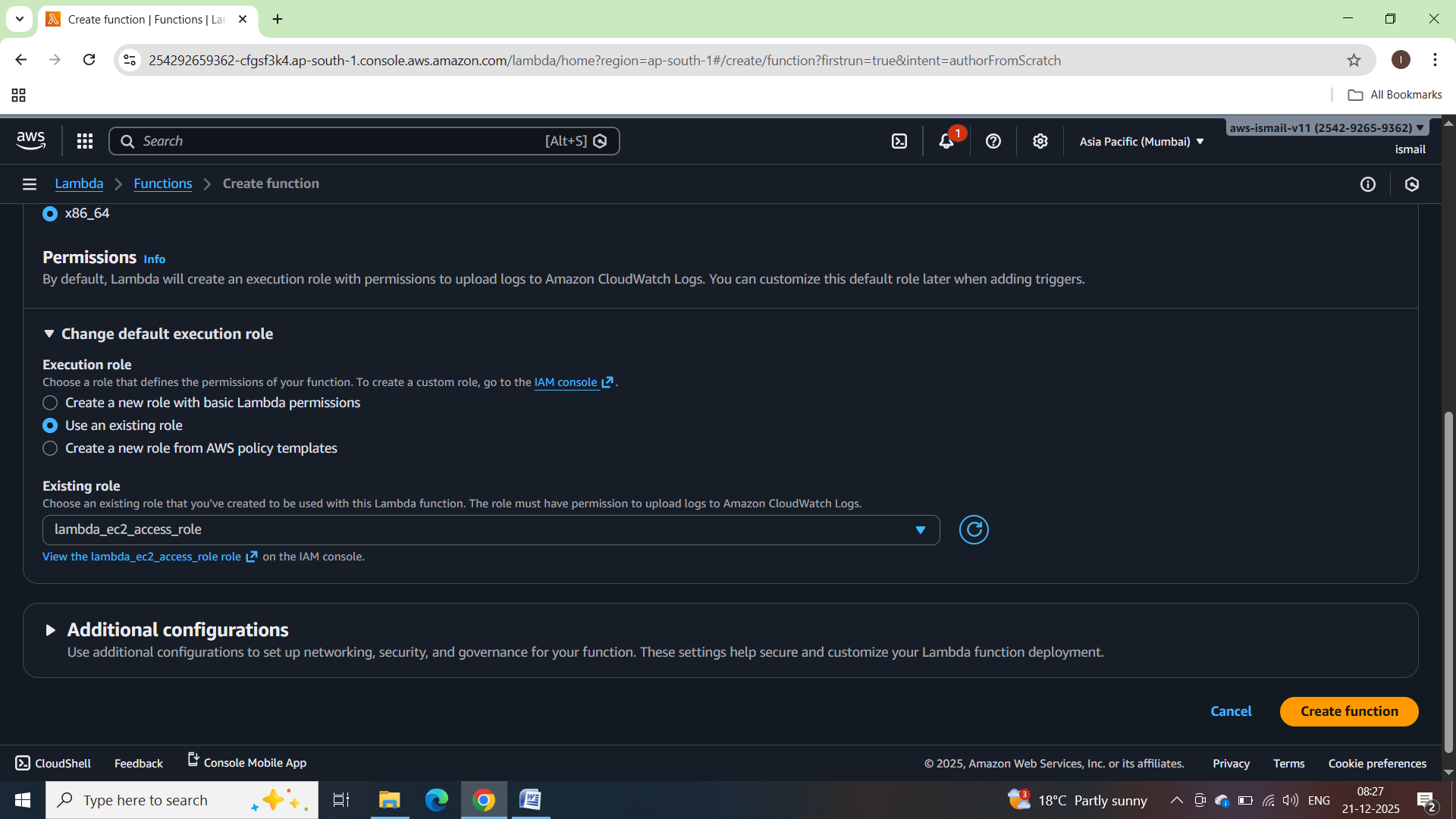
     1. Initialize a boto3 EC2 client.

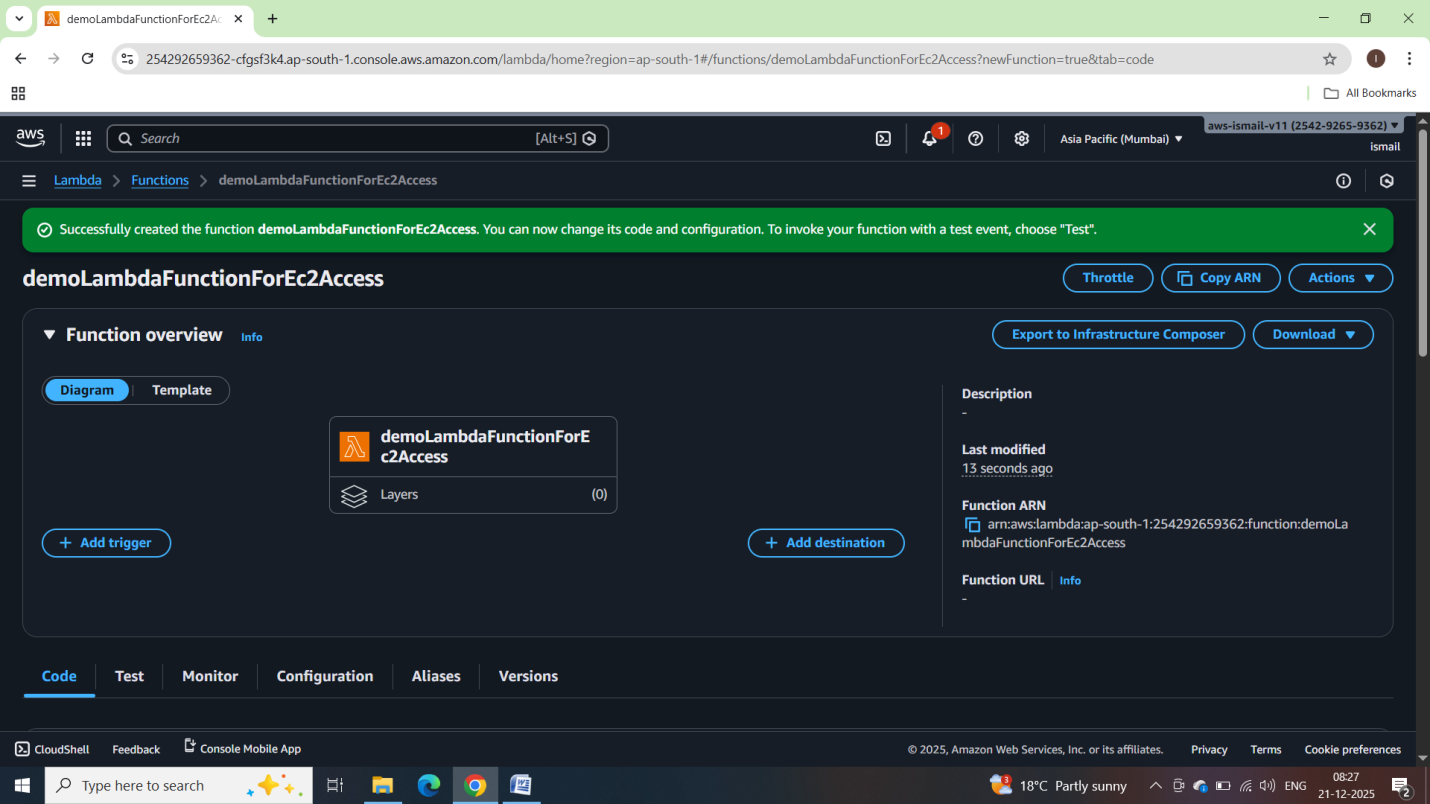
     2. Describe instances with `Auto-Stop` and `Auto-Start` tags.

     3. Stop the `Auto-Stop` instances and start the `Auto-Start` instances.

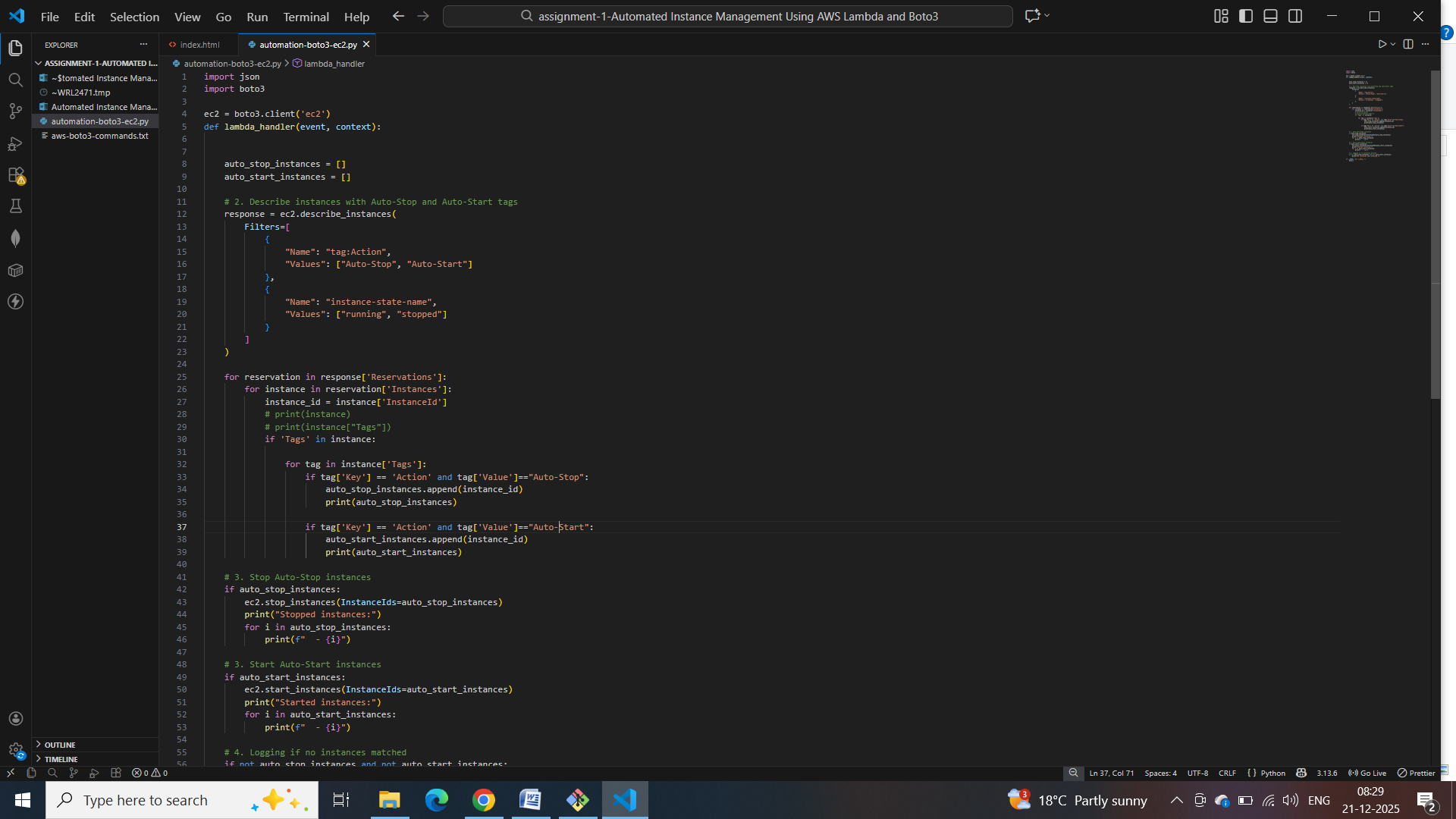
     4. Print instance IDs that were affected for logging purposes.

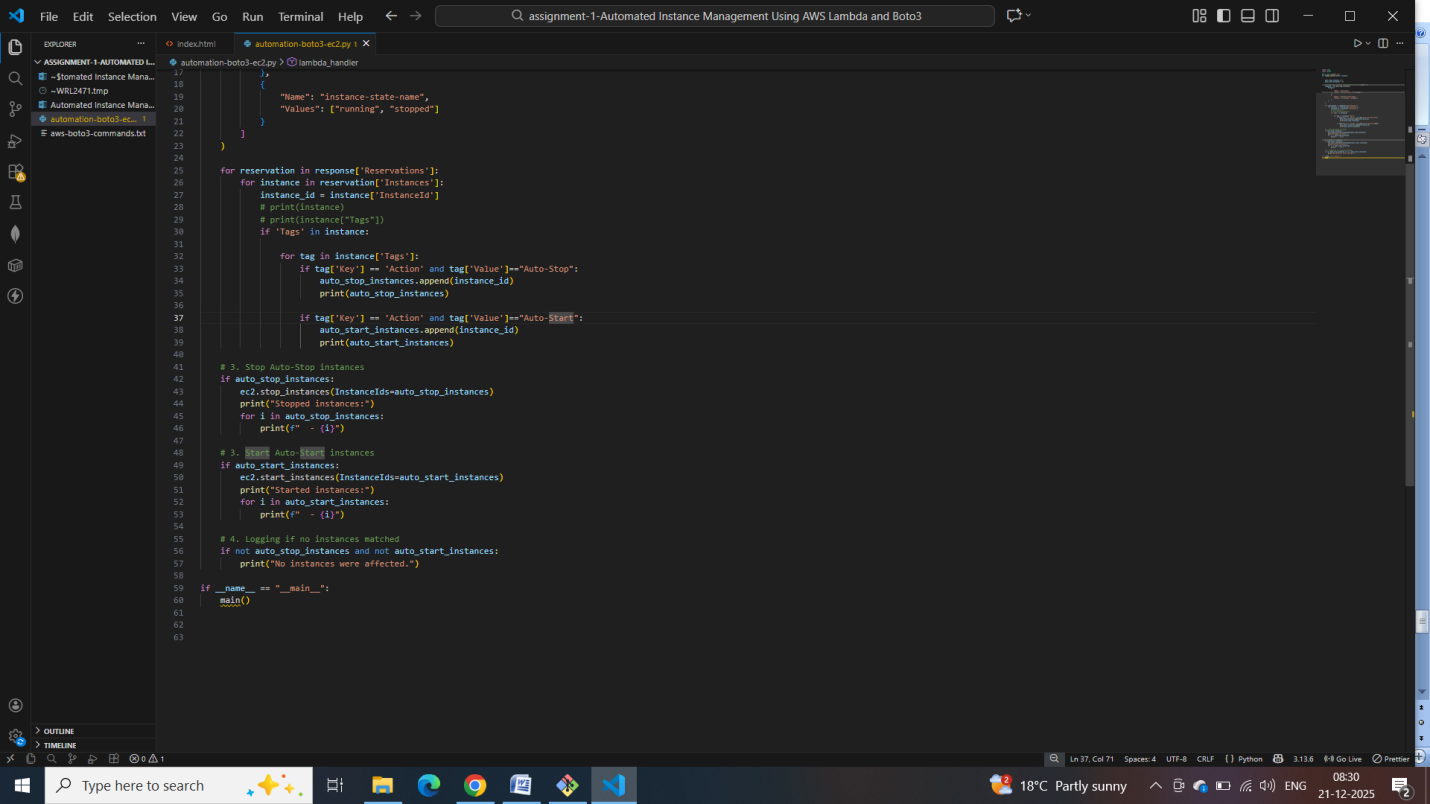






My python script written in my computer





Python boto3 code:

import json

import boto3

ec2 = boto3.client('ec2')

def lambda\_handler(event, context):

    auto\_stop\_instances = []

    auto\_start\_instances = []

    # 2. Describe instances with Auto-Stop and Auto-Start tags

    response = ec2.describe\_instances(

        Filters=[

            {

                "Name": "tag:Action",

                "Values": ["Auto-Stop", "Auto-Start"]

            },

            {

                "Name": "instance-state-name",

                "Values": ["running", "stopped"]

            }

        ]

    )

    for reservation in response['Reservations']:

        for instance in reservation['Instances']:

            instance\_id = instance['InstanceId']

            # print(instance)

            # print(instance["Tags"])

            if 'Tags' in instance:

                for tag in instance['Tags']:

                    if tag['Key'] == 'Action' and tag['Value']=="Auto-Stop":

                        auto\_stop\_instances.append(instance\_id)

                        print(auto\_stop\_instances)

                    if tag['Key'] == 'Action' and tag['Value']=="Auto-Start":

                        auto\_start\_instances.append(instance\_id)

                        print(auto\_start\_instances)

    # 3. Stop Auto-Stop instances

    if auto\_stop\_instances:

        ec2.stop\_instances(InstanceIds=auto\_stop\_instances)

        print("Stopped instances:")

        for i in auto\_stop\_instances:

            print(f"  - {i}")

    # 3. Start Auto-Start instances

    if auto\_start\_instances:

        ec2.start\_instances(InstanceIds=auto\_start\_instances)

        print("Started instances:")

        for i in auto\_start\_instances:

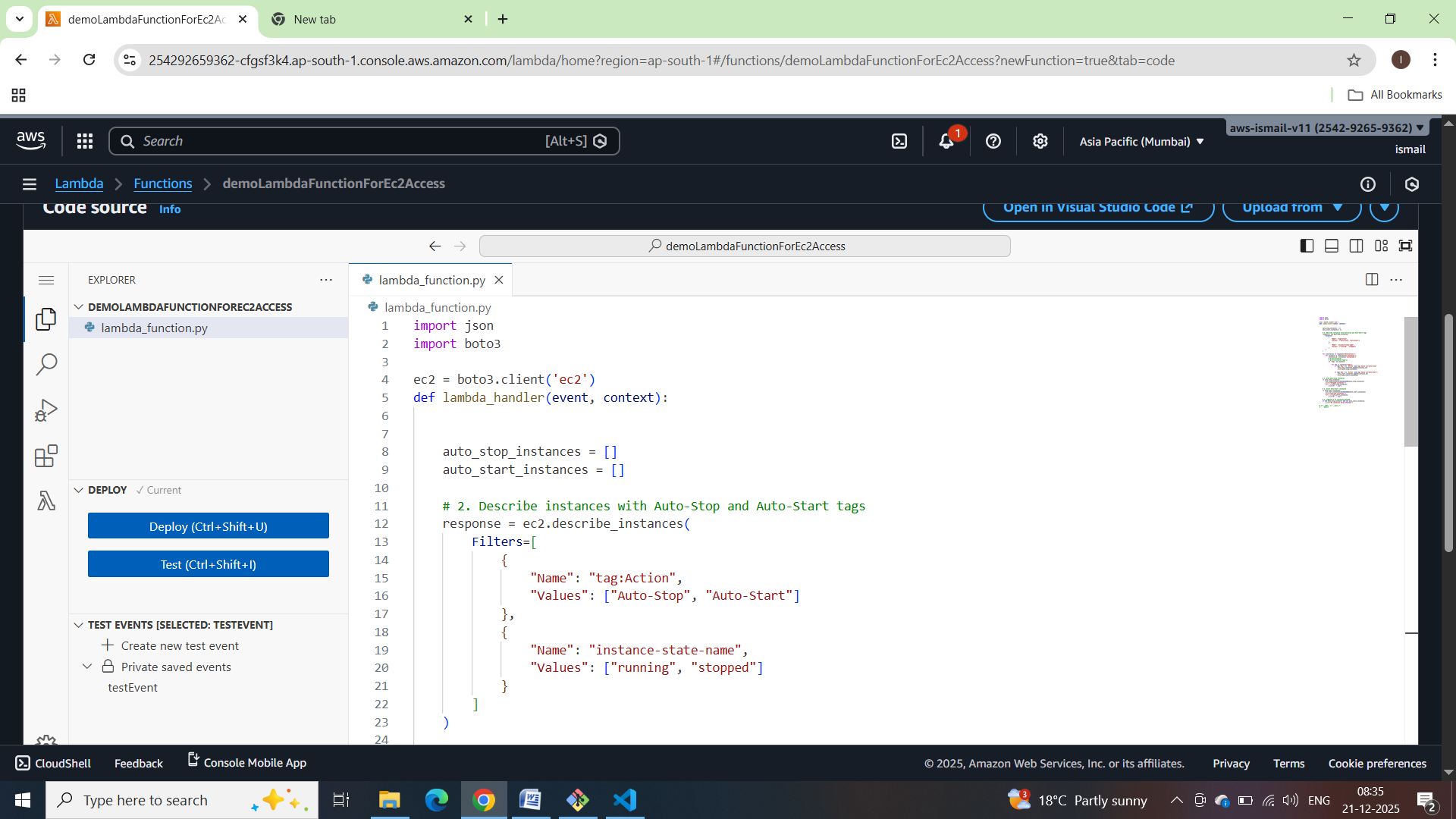
            print(f"  - {i}")

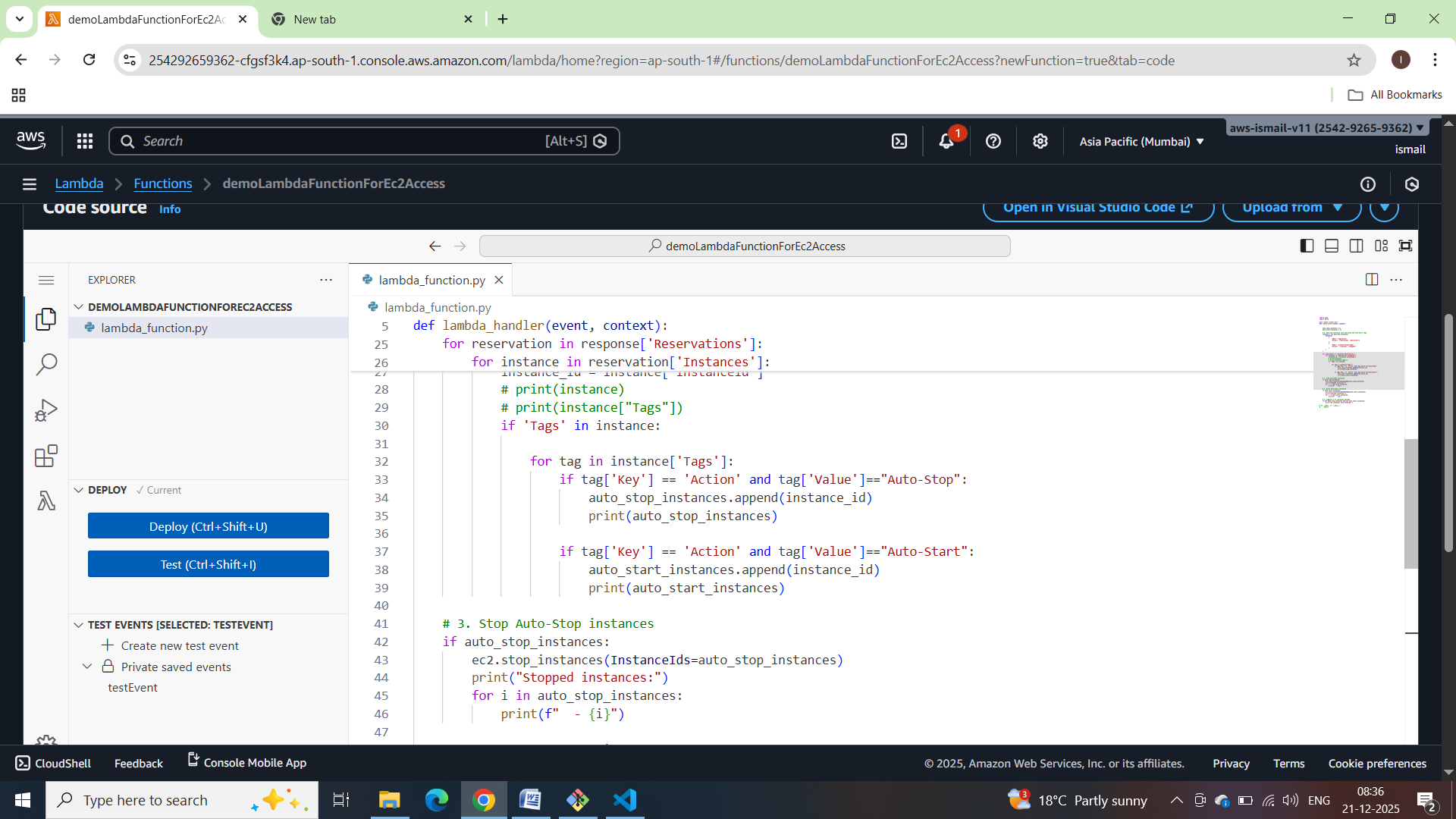
    # 4. Logging if no instances matched

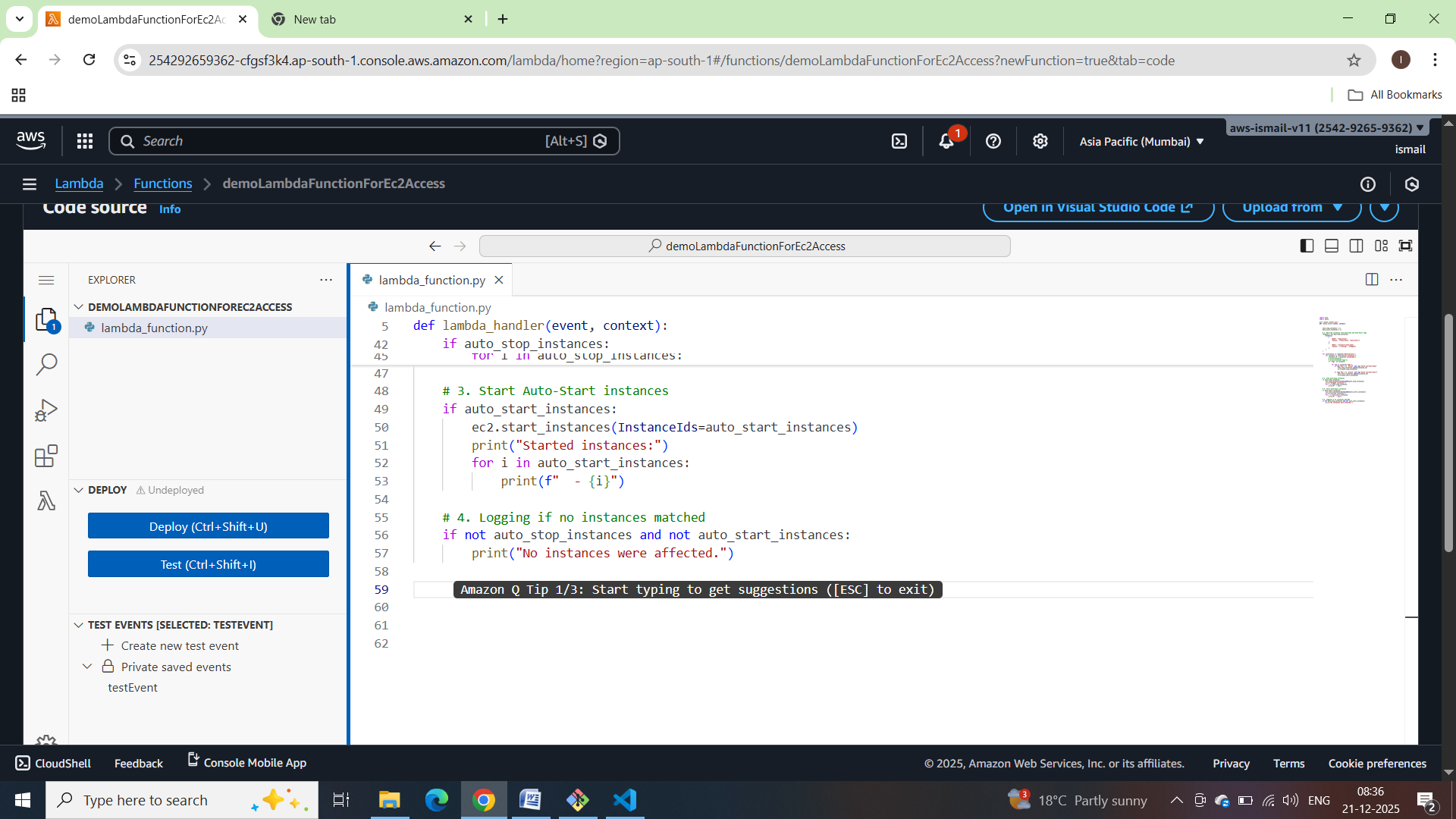
    if not auto\_stop\_instances and not auto\_start\_instances:

        print("No instances were affected.")

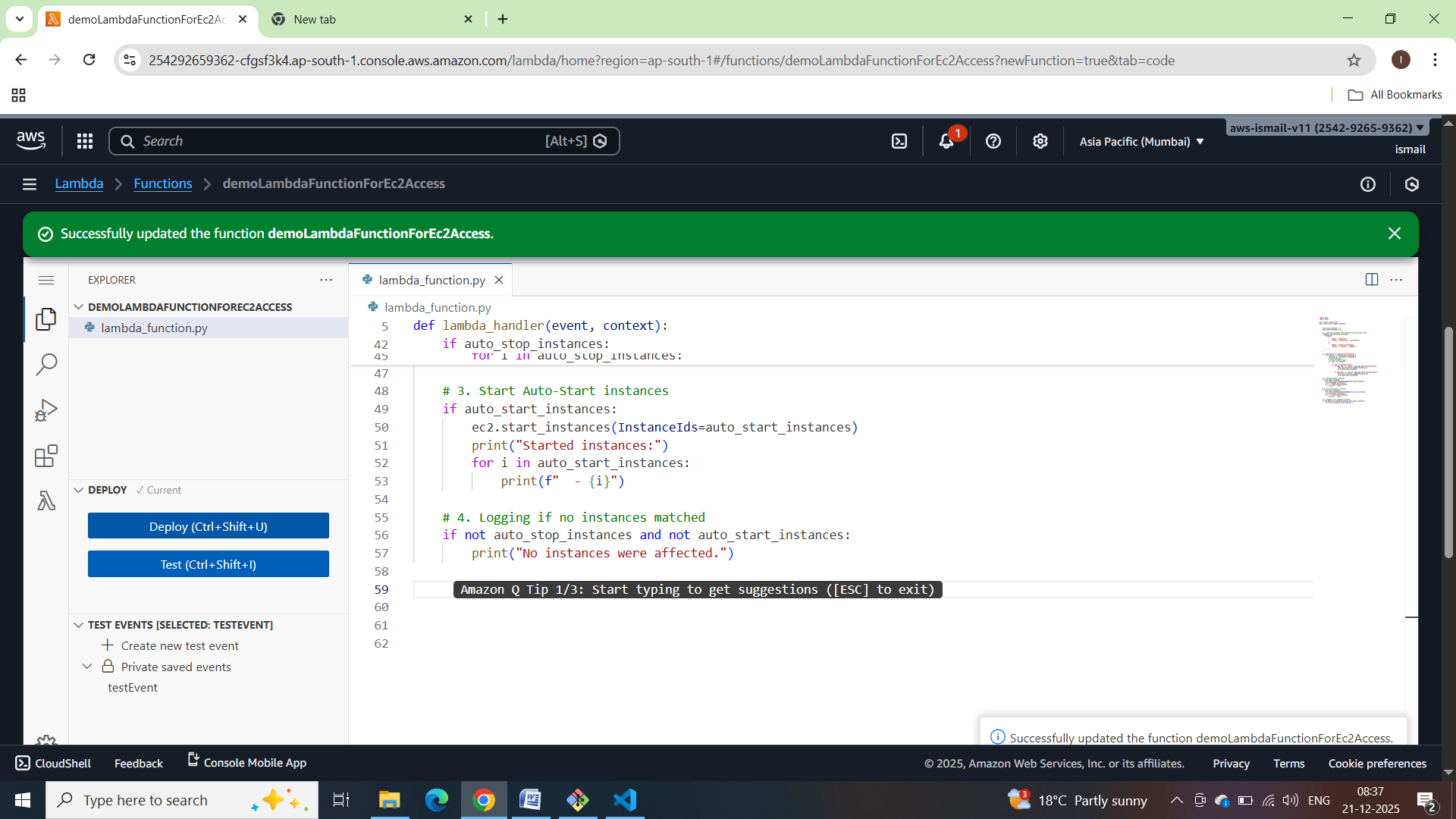
Copying this script to lambda function and deploying into lambada function



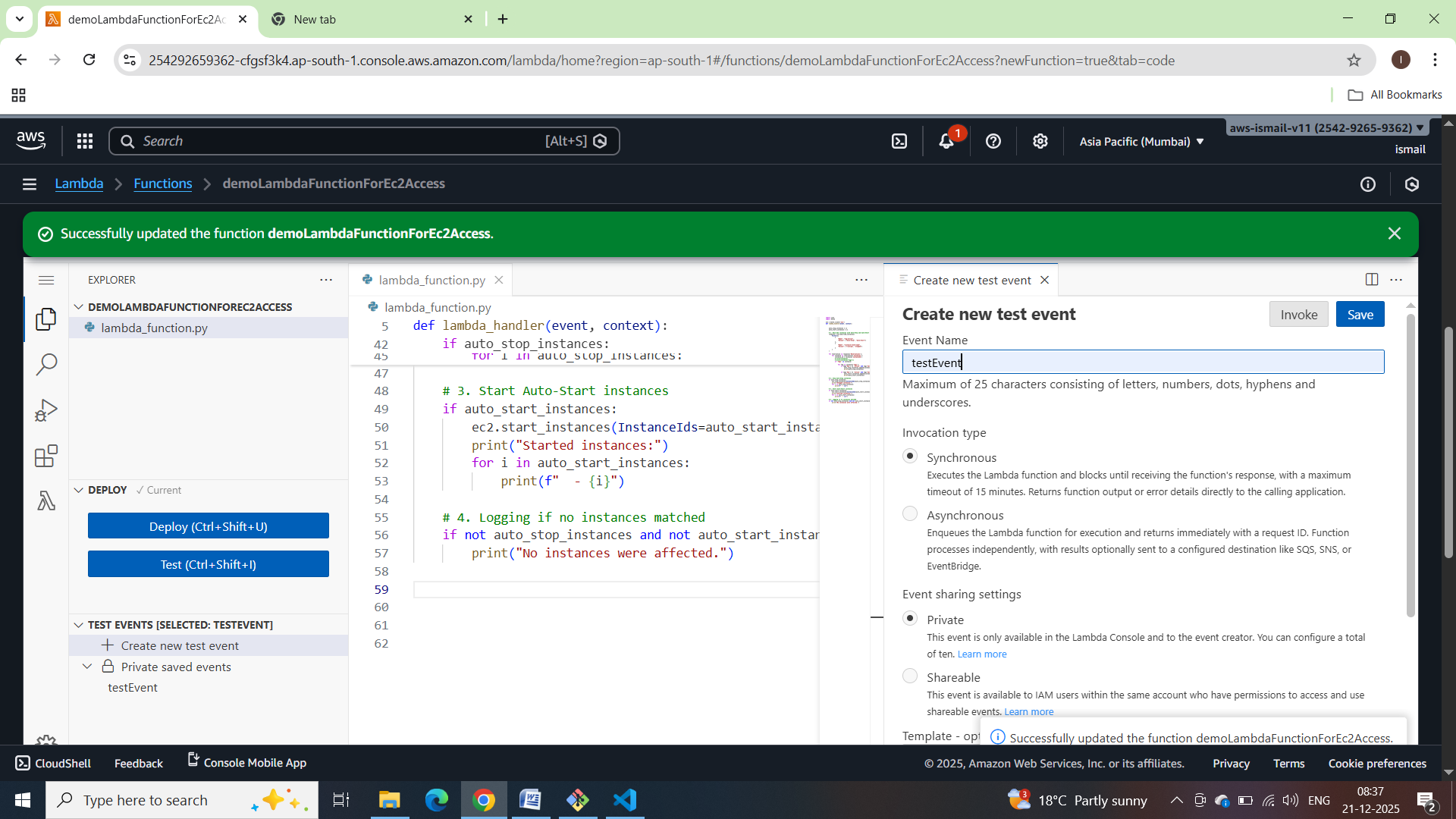




Deploying the lambda function:



Now creating the testEvent:



4. Manual Invocation:

   - After saving your function, manually trigger it.

   - Go to the EC2 dashboard and confirm that the instances' states have changed according to their tags.

Output on running the lambda function

