**Problem Statement:** Build an AWS **cost-effective** solution to read S3 bucket files present at /out folder. **Monitor the incoming .txt file** at /out folder and execute application only after getting file at the provided location. Write a script/inline application to count the words present in the .txt file and write it in count.txt file along with the **execution date**. Update count.txt file to **append count** and **store it at /count folder** of S3 bucket.

Also, to perform cost management develop an application to **shut down and start** dev account EC2 instances on 6PM and 9PM every day respectively.

Also, you have been tasked with identifying **14 days (about 2 weeks) older EBS volumes snapshot** and automate the process to delete those on weekly basis.

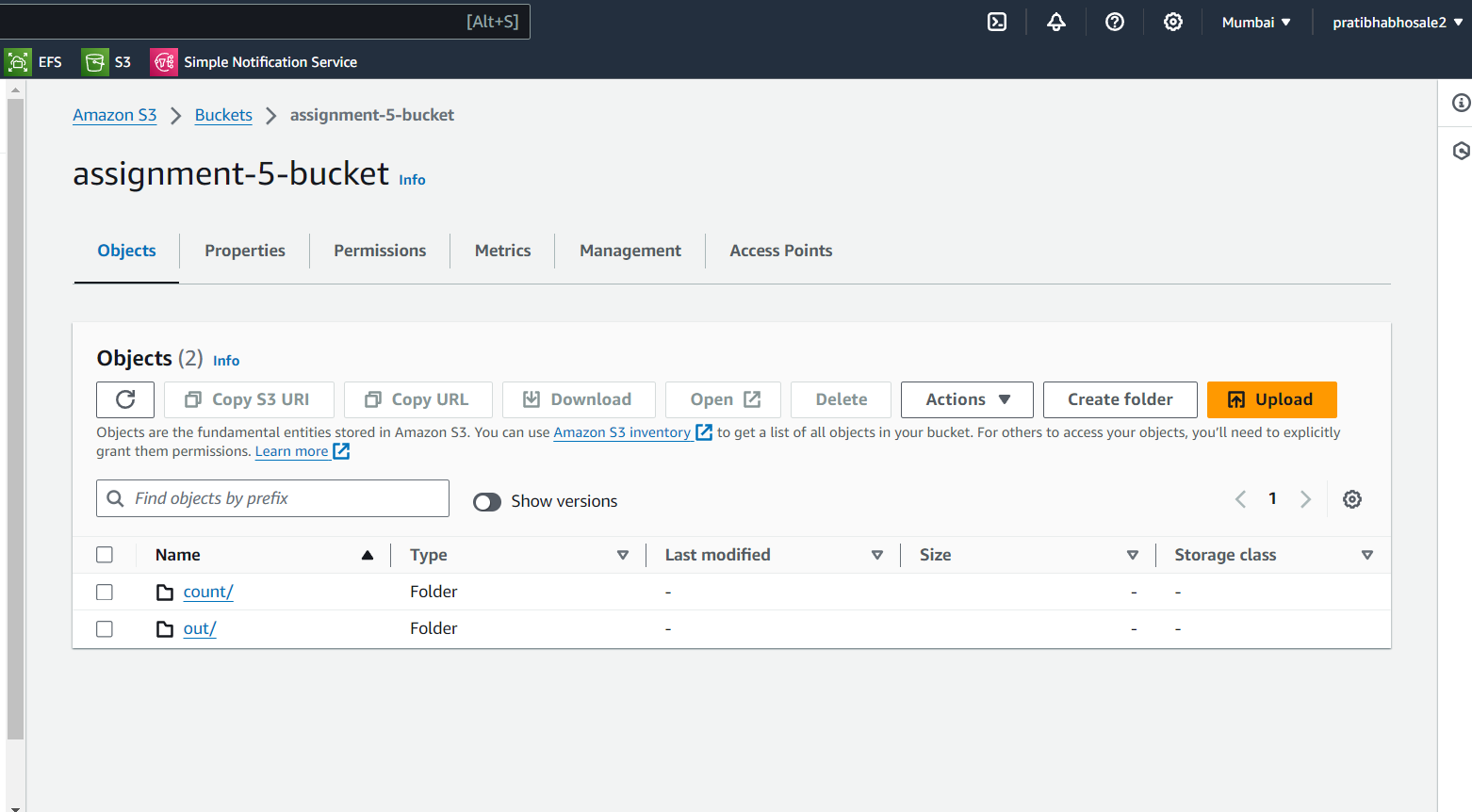
Develop logic in any known coding language like shell, python, NodeJS etc.

**Requirements**

1. **Amazon S3** – For file storage (input and output files).
2. **AWS Lambda** – For a cost-effective, serverless compute platform that can run your application logic only when a .txt file is uploaded.
3. **Amazon CloudWatch Events** – To monitor incoming files and trigger Lambda.
4. **AWS IAM** – For permission management between S3 and Lambda.

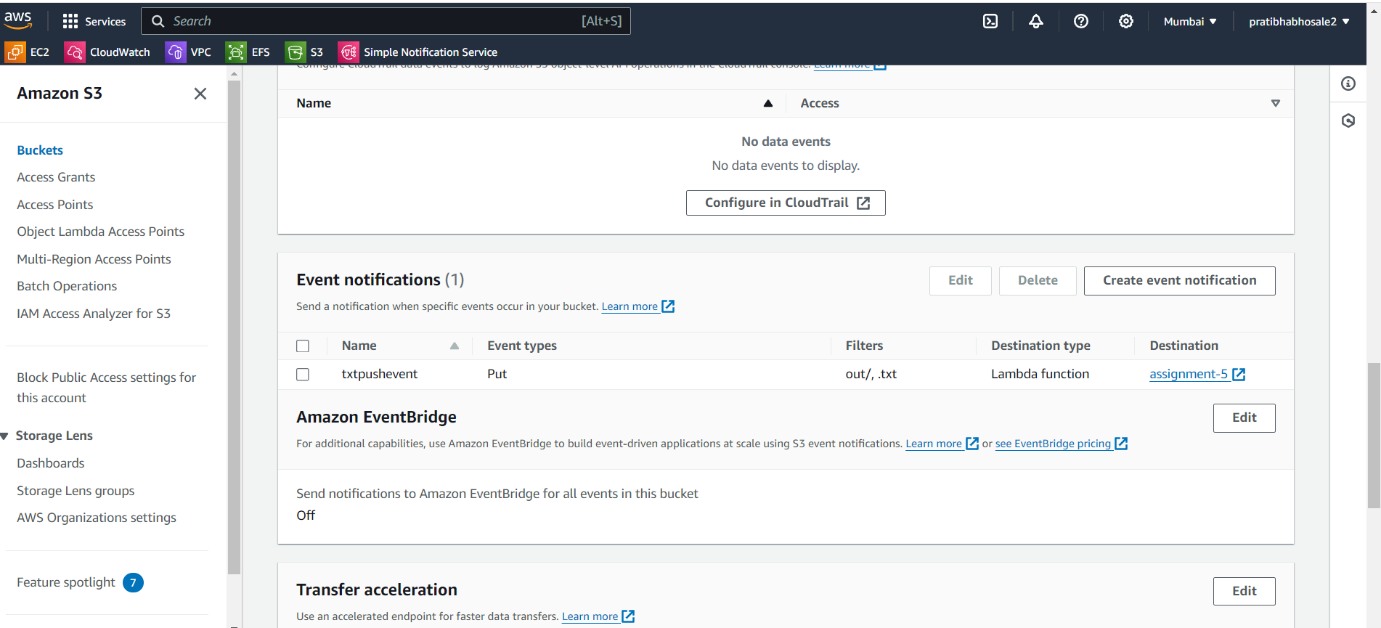
**1. Create an S3 Bucket with Required Folders**

* Create an S3 bucket (e.g., assignment-5-bucket) with two folders:
  + /out folder: The input .txt files will be uploaded here
  + /count folder: This is where the result files (word count and execution date) will be saved



**2. Set Up an S3 Event Trigger for the Lambda Function**

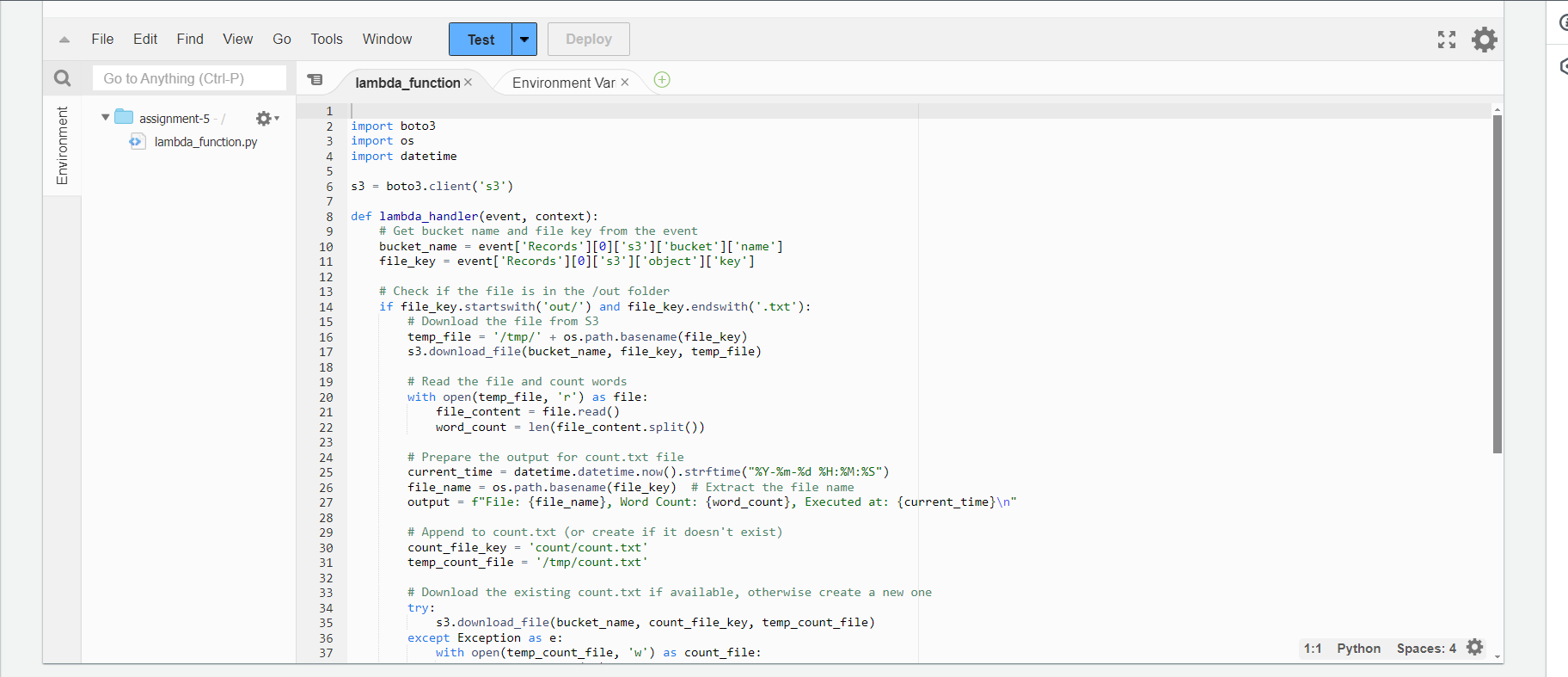
* Configure the S3 bucket to trigger a Lambda function whenever a .txt file is uploaded to the /out folder.
* In the **S3 bucket** settings:
  + Go to **Properties** > **Event Notifications**.
  + Add a notification for PUT events for the /out/\*.txt files.
  + Choose **Lambda Function** as the destination.



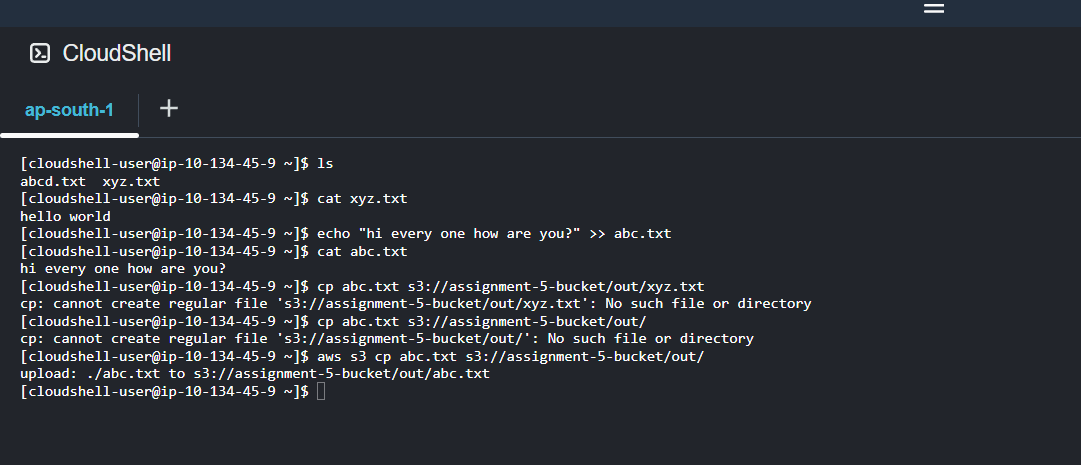
**3. Create the Lambda Function**

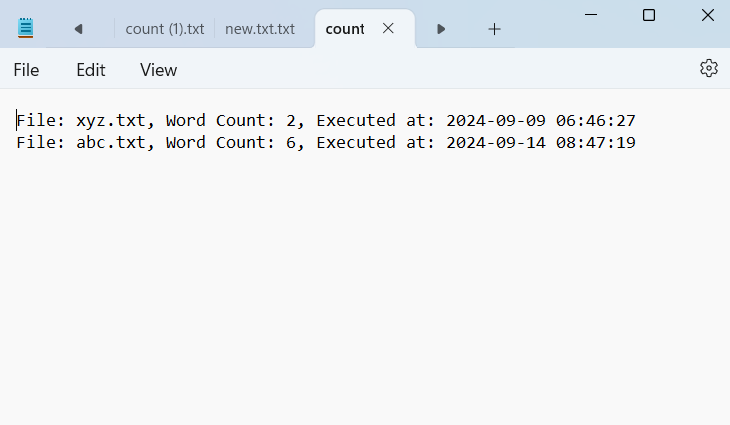
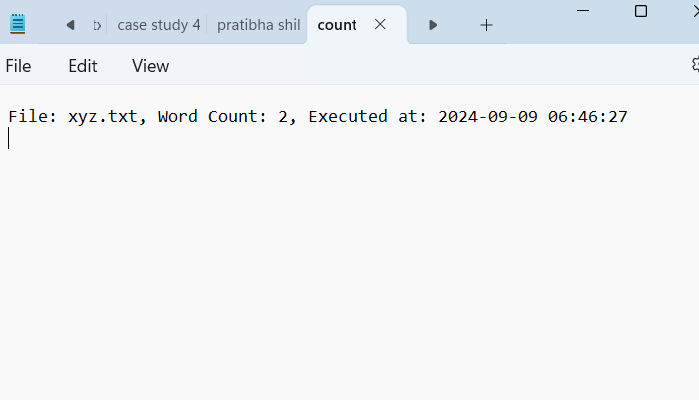
The Lambda function will count the words in the .txt file, write the result into a count.txt file with the execution date, and save it in the /count folder in the S3 bucket.

* Go to **AWS Lambda** and create a new function.
* Choose **Author from Scratch**.
* Select **Python 3.9** for scripting .



Uploaded xyz.txt file in out folder





**Task No-2**

**Step 1: Create the Lambda Functions**

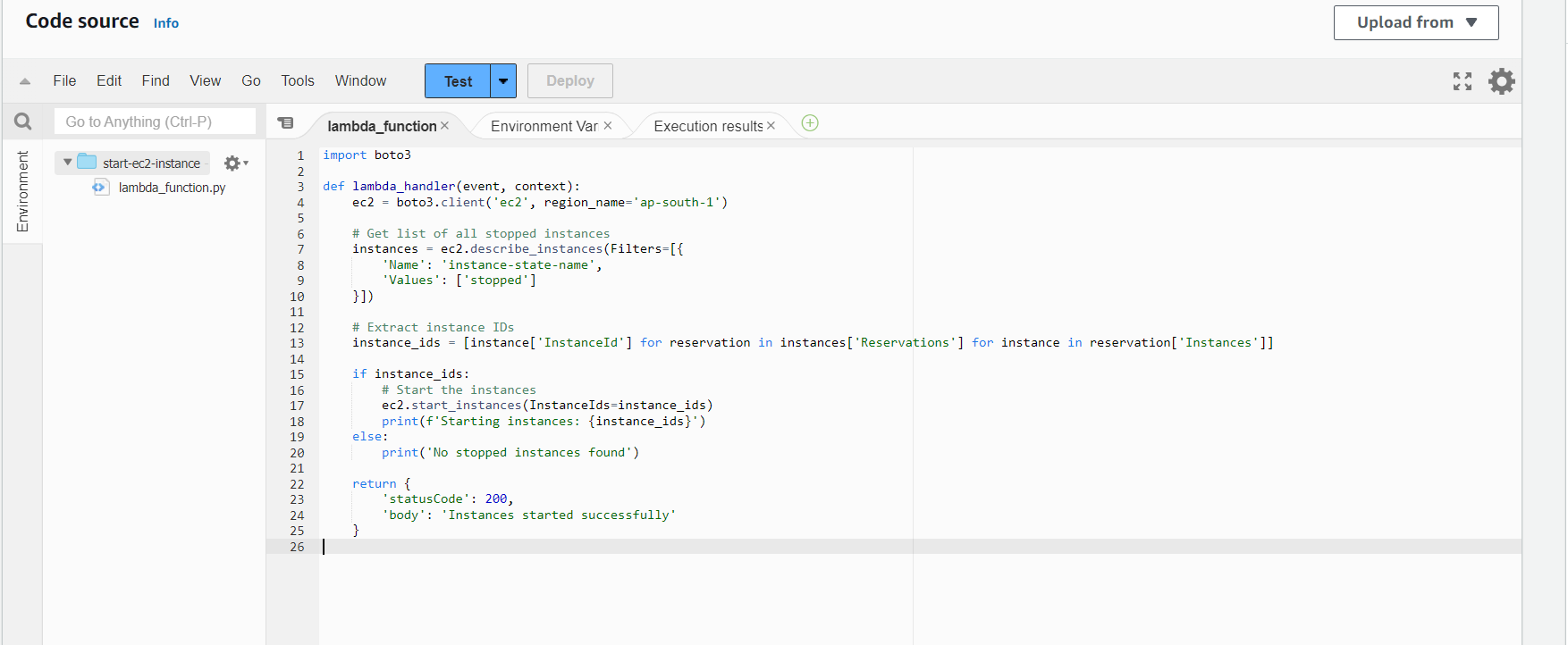
**Step2- create IAM role with policy (start,stop and describe instances)**

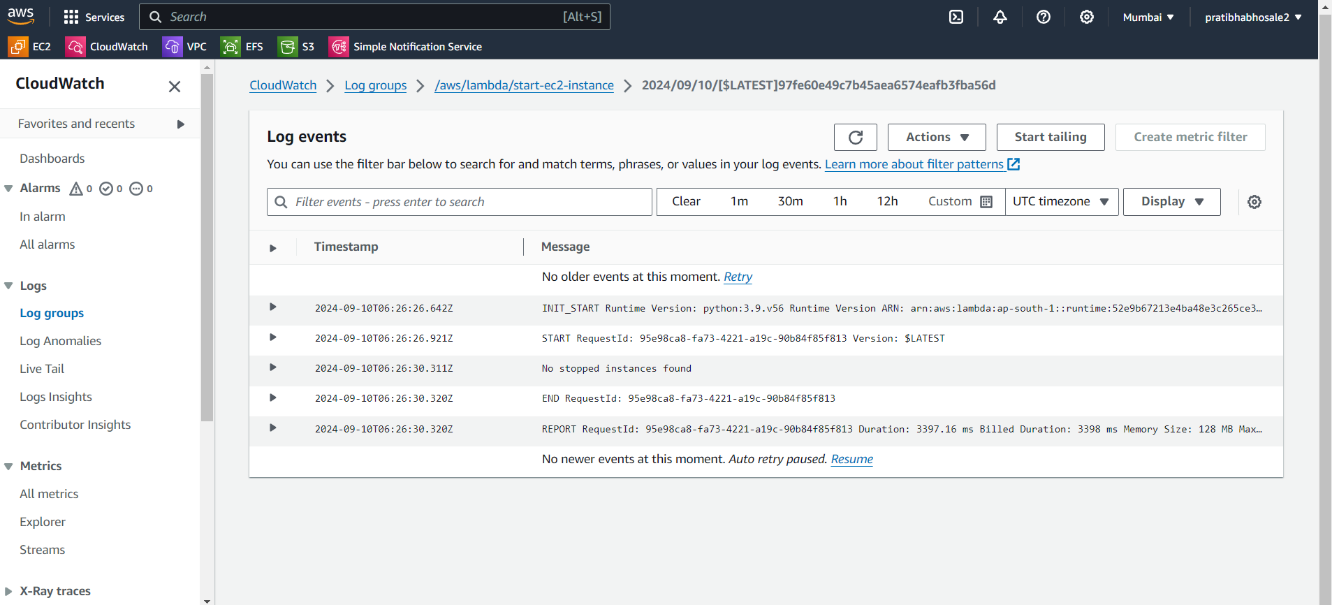
**Step3- attach this role to lambda function**

**Lambda Function to Stop EC2 Instances at 6 PM**

**Go to AWS Lambda and create a new function.**

* **Name: StopEC2Instances**
* **Runtime: Python 3.9**

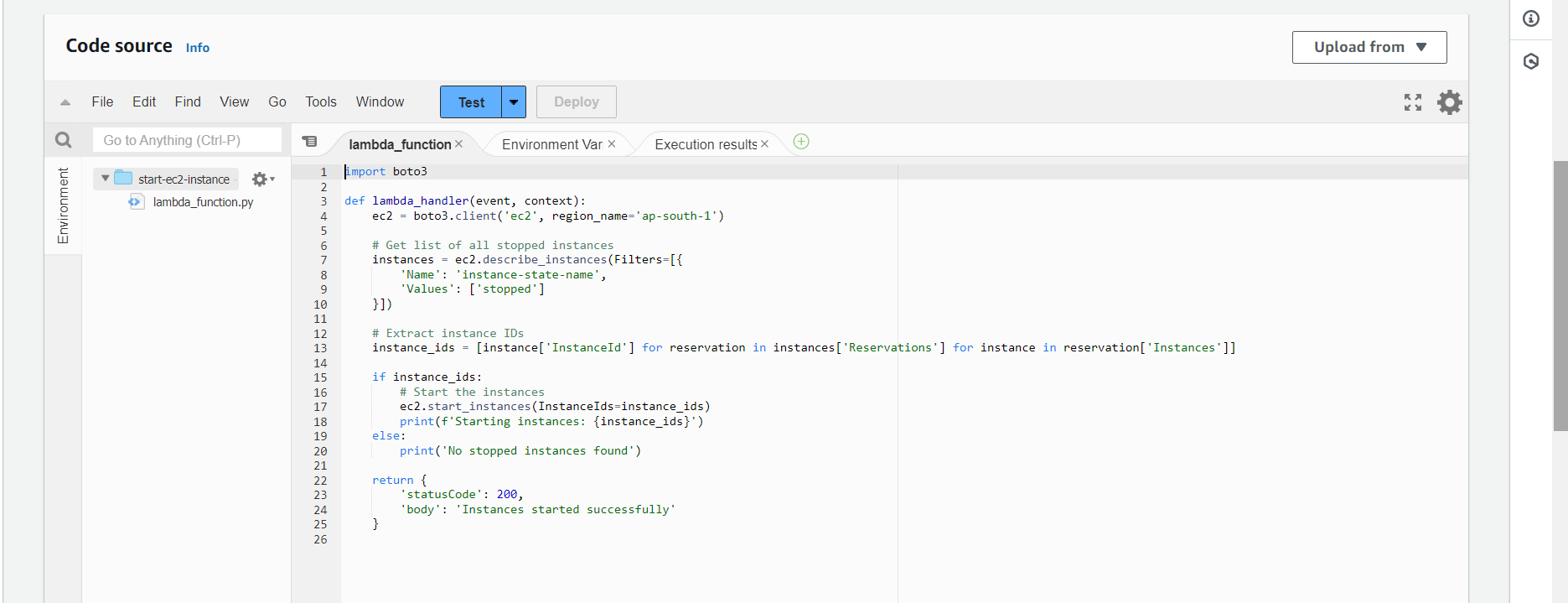
****

****

**Lambda Function to Start EC2 Instances at 9 PM**

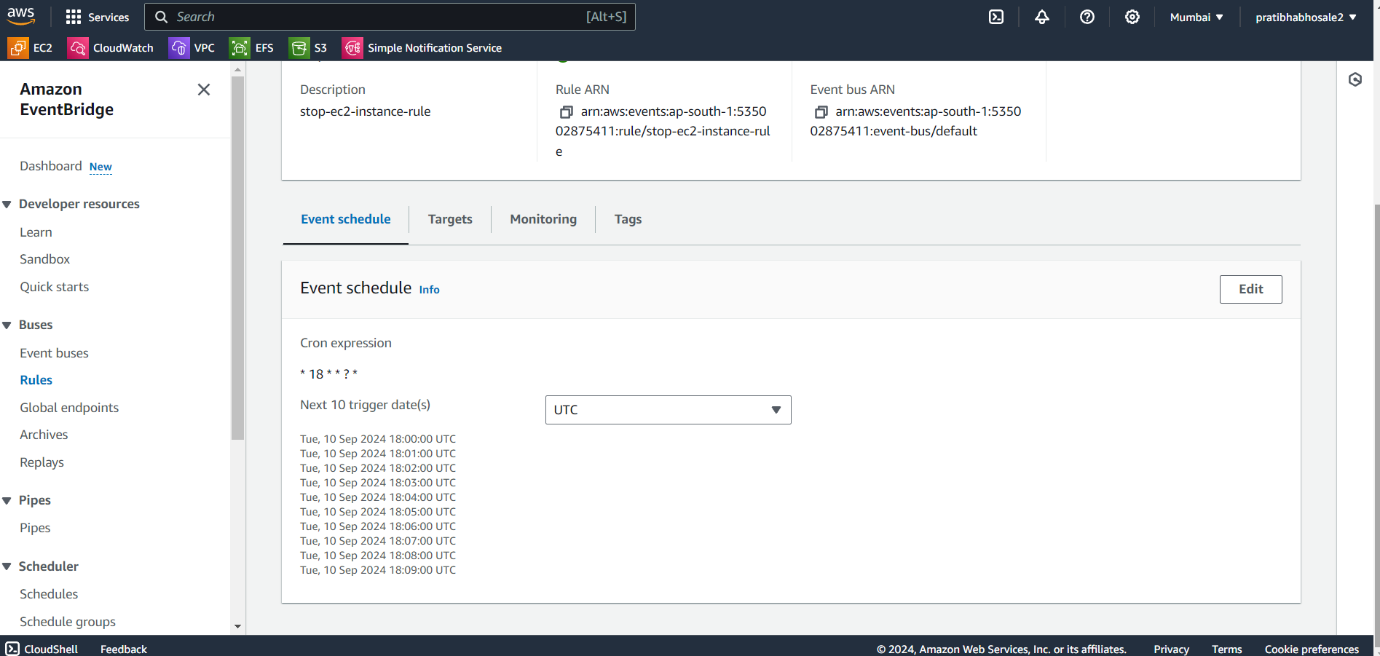
**Create another Lambda function:**

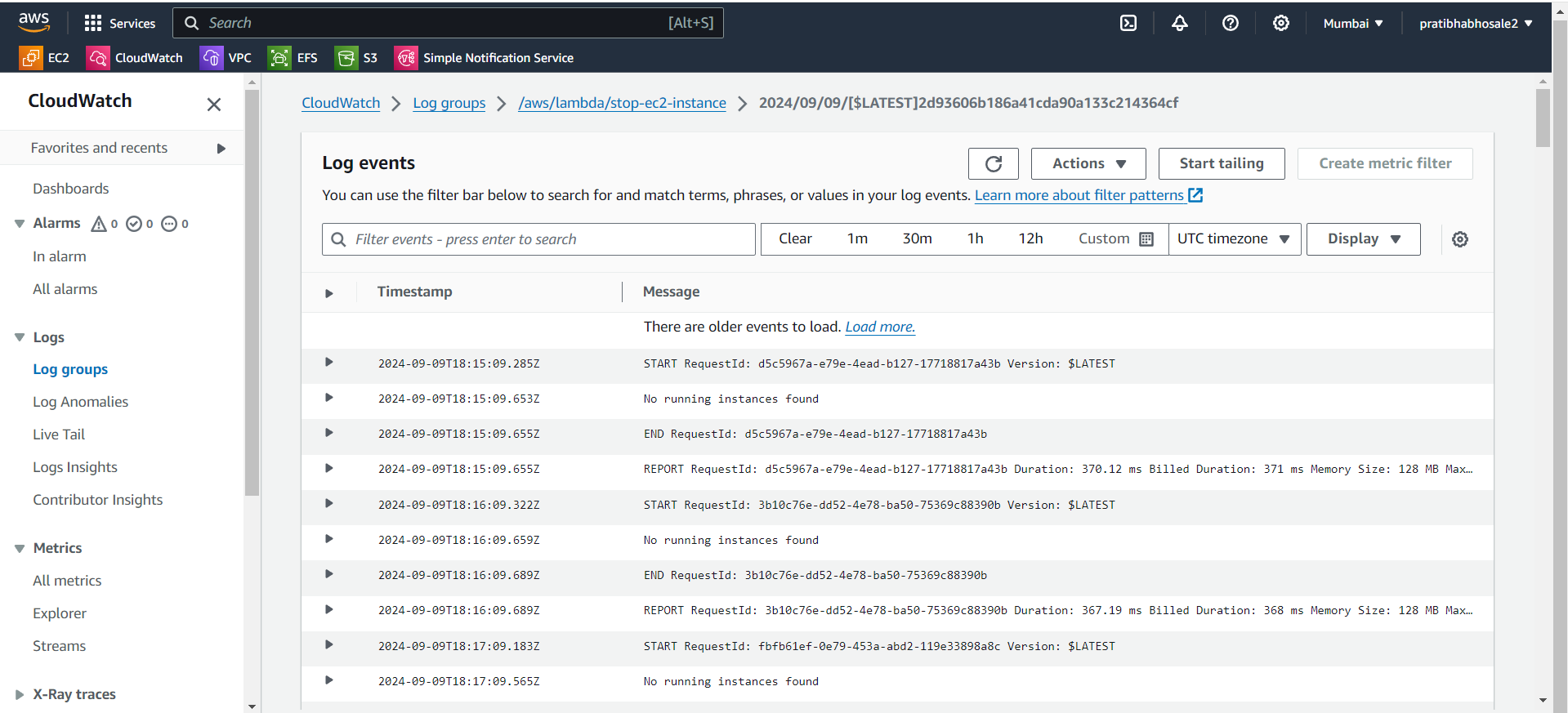
* **Name: StartEC2Instances**
* **Runtime: Python 3.9.**

****

**Set Up CloudWatch EventCreate a CloudWatch Rule to Trigger the Stop Function (6 PM)**

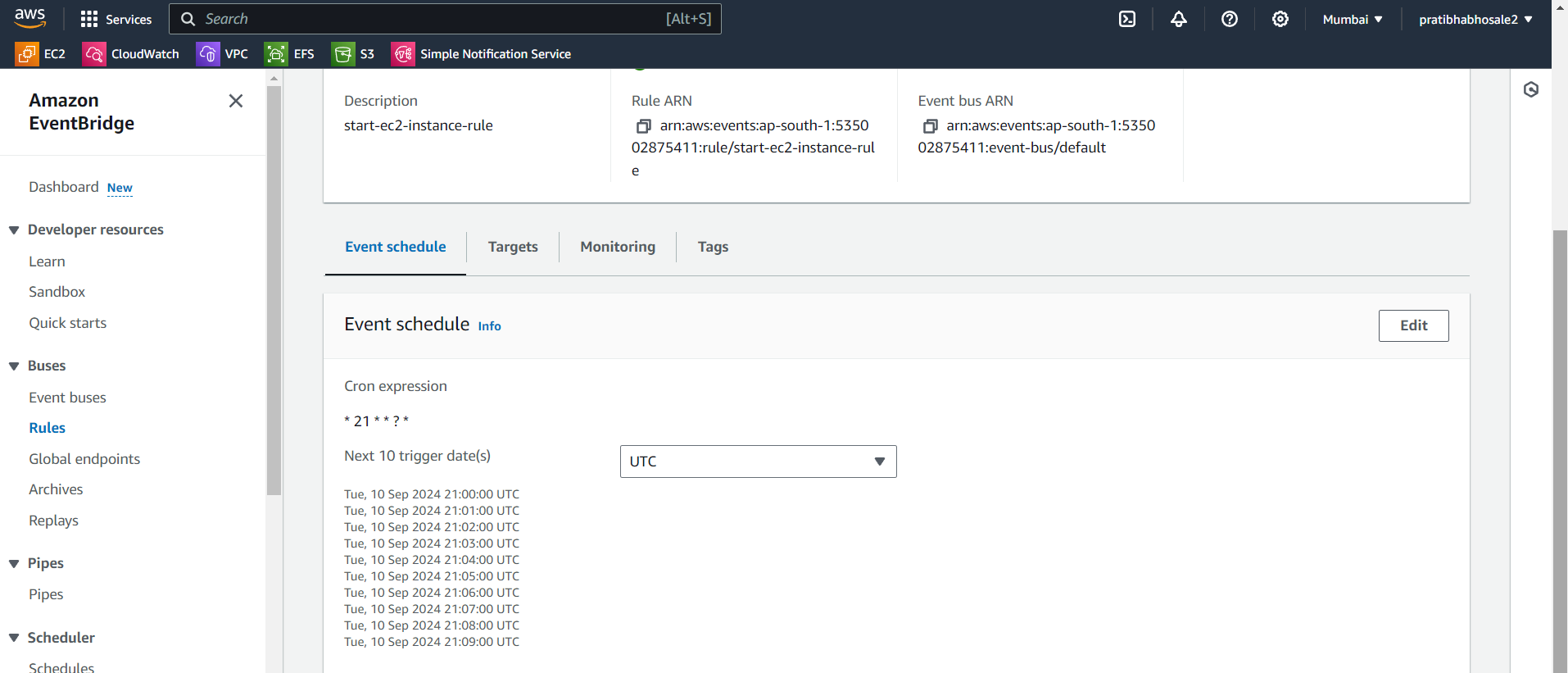
* Go to **Amazon CloudWatch** > **Rules** > **Create Rule**.
* **Event Source**: Choose **Event Schedule** and set it to trigger daily at **6 PM** (using cron expression).
  + Cron expression for 6 PM daily: 0 18 \* \* ? \*
* **Target**: Select the Lambda function StopEC2Instances.

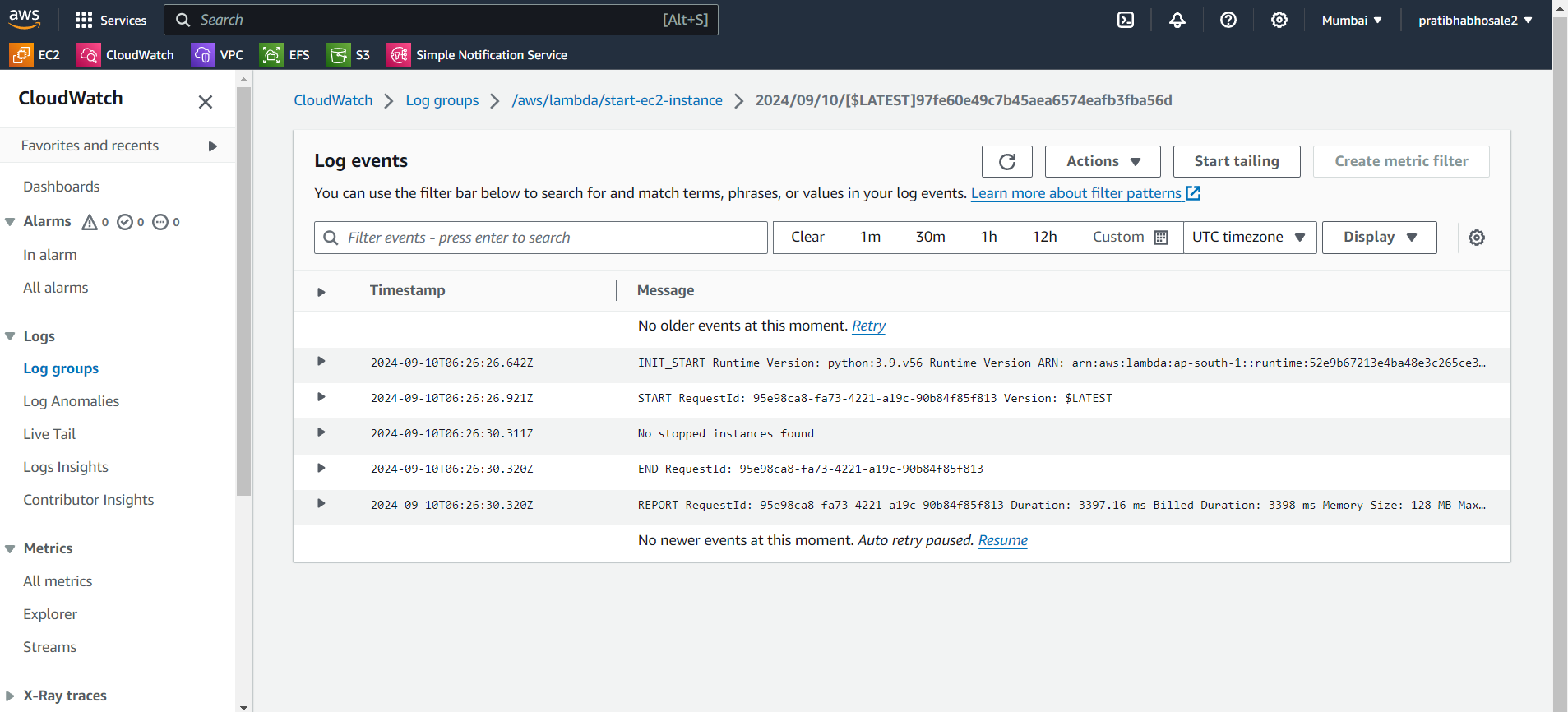




**Create a CloudWatch Rule to Trigger the Start Function (9 PM)**

* Go to **Amazon CloudWatch** > **Rules** > **Create Rule**.
* **Event Source**: Choose **Event Schedule** and set it to trigger daily at **9 PM**.
  + Cron expression for 9 PM daily: 0 21 \* \* ? \*
* **Target**: Select the Lambda function StartEC2Instances.



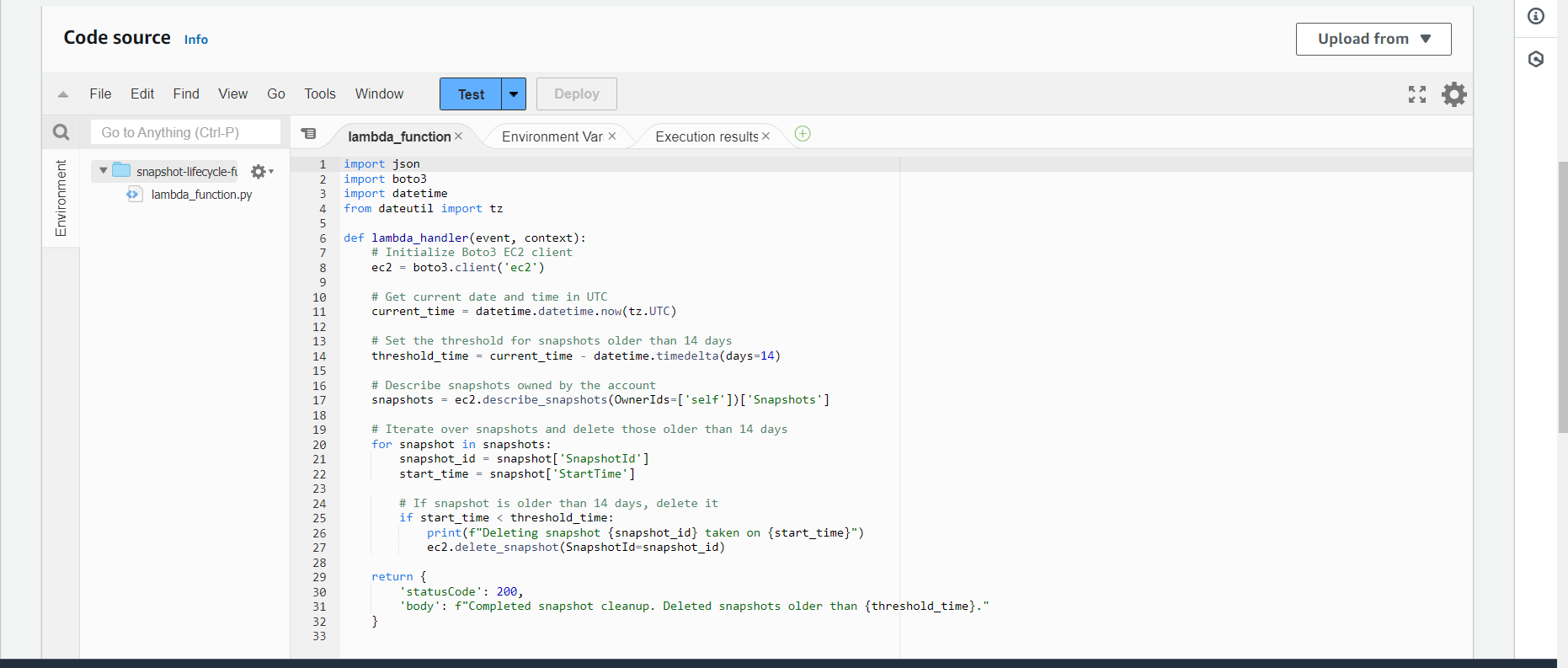
****

Task -3 deletion of snapshots older than 14 days

Step 1: Create the IAM Role for Lambda with permission describe snapshots and delete snapshots.

**Step 2: Create the Lambda Function**

Create a Lambda function using Python



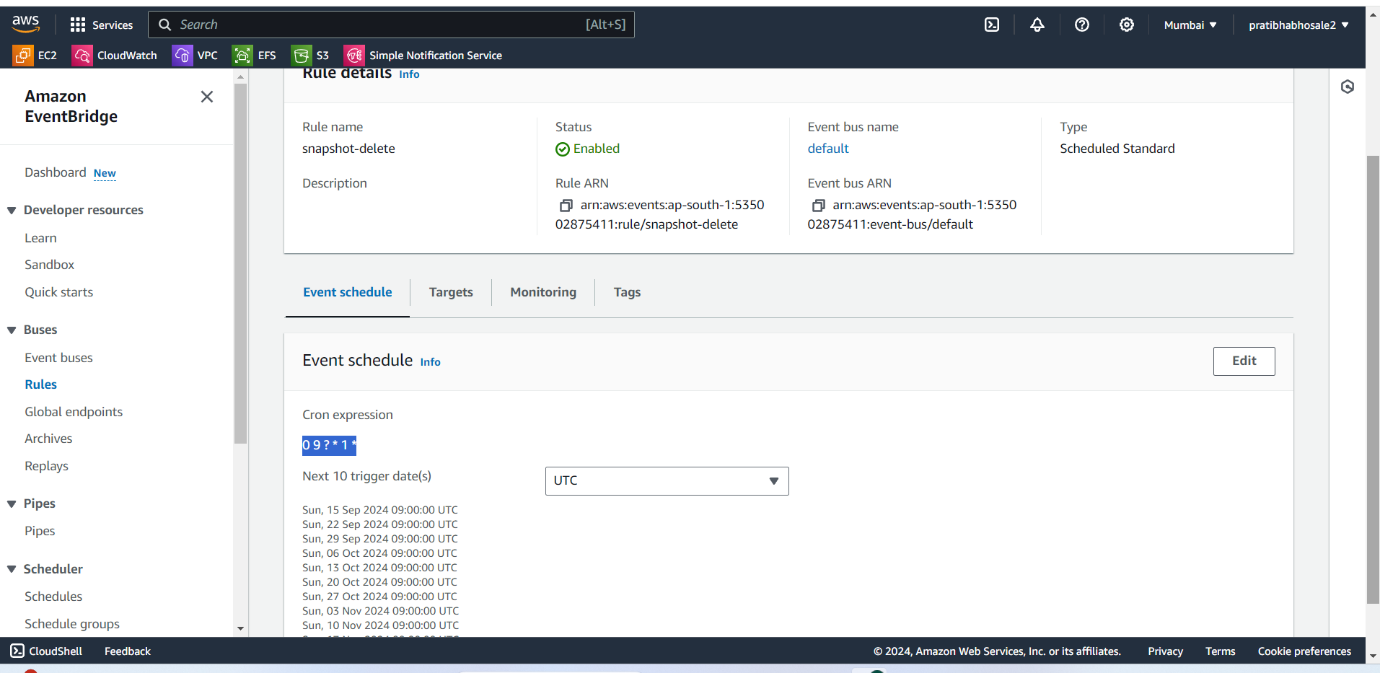
**Step 3: Set Up a CloudWatch Event Rule (EventBridge)**

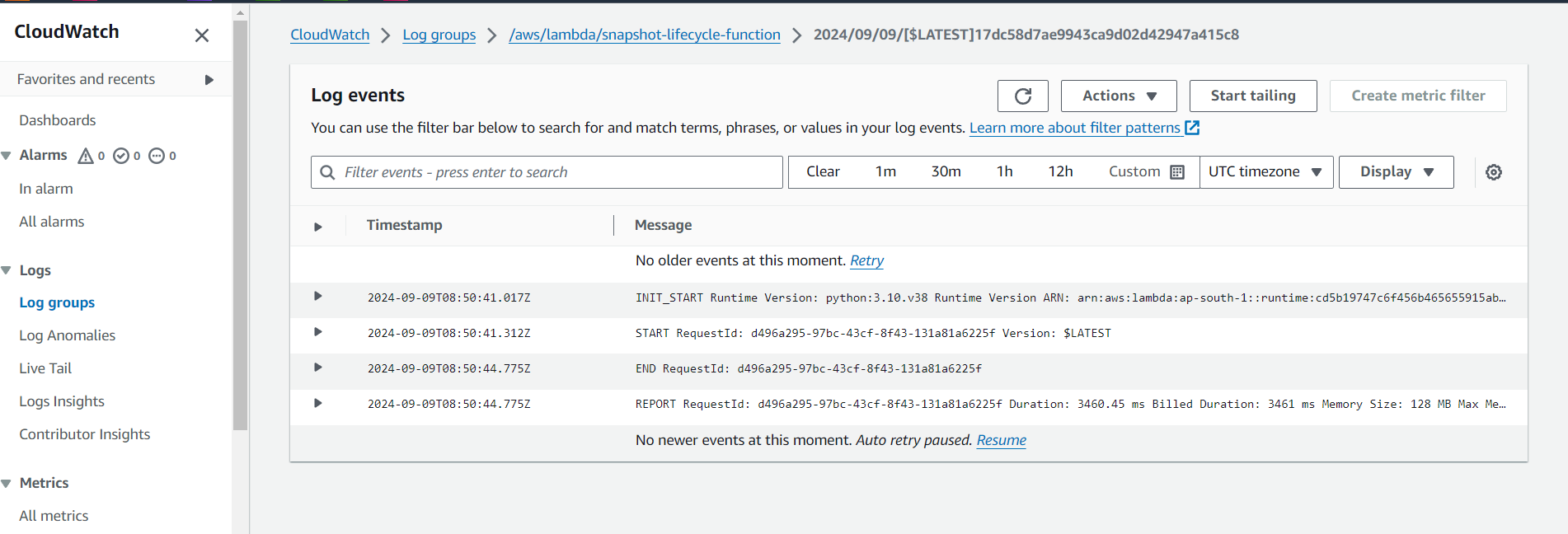
In the **EventBridge** console, create a rule to run the Lambda function on a weekly basis.

Configure the schedule expression like this:

**Cron expression**: 0 9 ? \* 1 \*

This runs the function every sunday at 9 UTC





Github repository for code: https://github.com/Pratibha251093/Case-study-5.git