**Assignments(Karthik) 09-09-2023**

1. **How to use memory as a metric for auto scaling groups.**

By default CloudWatch will not track the Memory Utilization for that we need to install CloudWatch Agent. It is used to track the memory Utilization of ec2 instance and can be used as a metric for Auto Scalling Group.

Installation of CloudWatch Agent:

Step1: create a IAM role for ec2 instancea and name it as CWAgentRole and attach AmazonEc2RoleforSSM and CloudWatchAgentServerPloicy to the role.

Step2: Create a Ec2 instance and attach the IAM role and in userdata write shell script - yum update –y and in security groups allow ssh and https ports and allow anywhere to allow cloudwatch can communicate with the cloudwatch agent

Step3: login to the ec2 machine and install the CloudWatch Agent

Command: sudo yum install amazon-cloudwatch-agent –y

Step4: move to the folder where cloudwatch agent was installed and lauch the configuration file

**Sudo cd /opt/aws/amazon-cloudwatch-agent/bin/ amazon-cloudwatch-agent-config-wizard** 🡪 to launch the agent and choose the required configures and give path as /var/log/messages and name of metrics for cloudwatch i.e., ec2logsforcwagentinstance.

Step5: move inside the folder where cloudwatch agent installed i.e., Sudo cd /opt/aws/amazon-cloudwatch-agent/bin/ and check config.jason file has created or not.

Step6: start the cloudwatch agent

Command to start agent:

**sudo /opt/aws/amazon-cloudwatch-agent/bin/amazon-cloudwatch-agent-ctl -a fetch-config –m\ ec2 -s -c file:/opt/aws/amazon-cloudwatch-agent/bin/config.json**

**step7:** Goto CloudWatch Console 🡪 metrics 🡪all metrics 🡪 hete CWAgent has created as a custome namespace.

Reference: [Learn how to install AWS CloudWatch Agent on an EC2 instance - YouTube](https://www.youtube.com/watch?v=U7X3ehGZYwQ&ab_channel=SkillCurb)

1. **At least 3 use cases of AWS Lambda.**
2. We can use AWS Lambda for creating and deleting a Ec2 instances at a certain period of time.

Example: consider Banking sector, at working hours it requires a 10 instances to smooth the service, so lambda creates a required instance at that particular time without manual intervention and we set a cron job for this to delete the instance once the work has done.

1. Deleting the old snapshots which are not required anymore. For creating a lambda function it checks the snapshots periodically and deletes if It older than 7 days.
2. **Where do you configure log rotation in linux**

In /etc/logrotate.conf

Example of configuration file: sudo vi /etc/logrotate.conf

/var/log/auth.log {

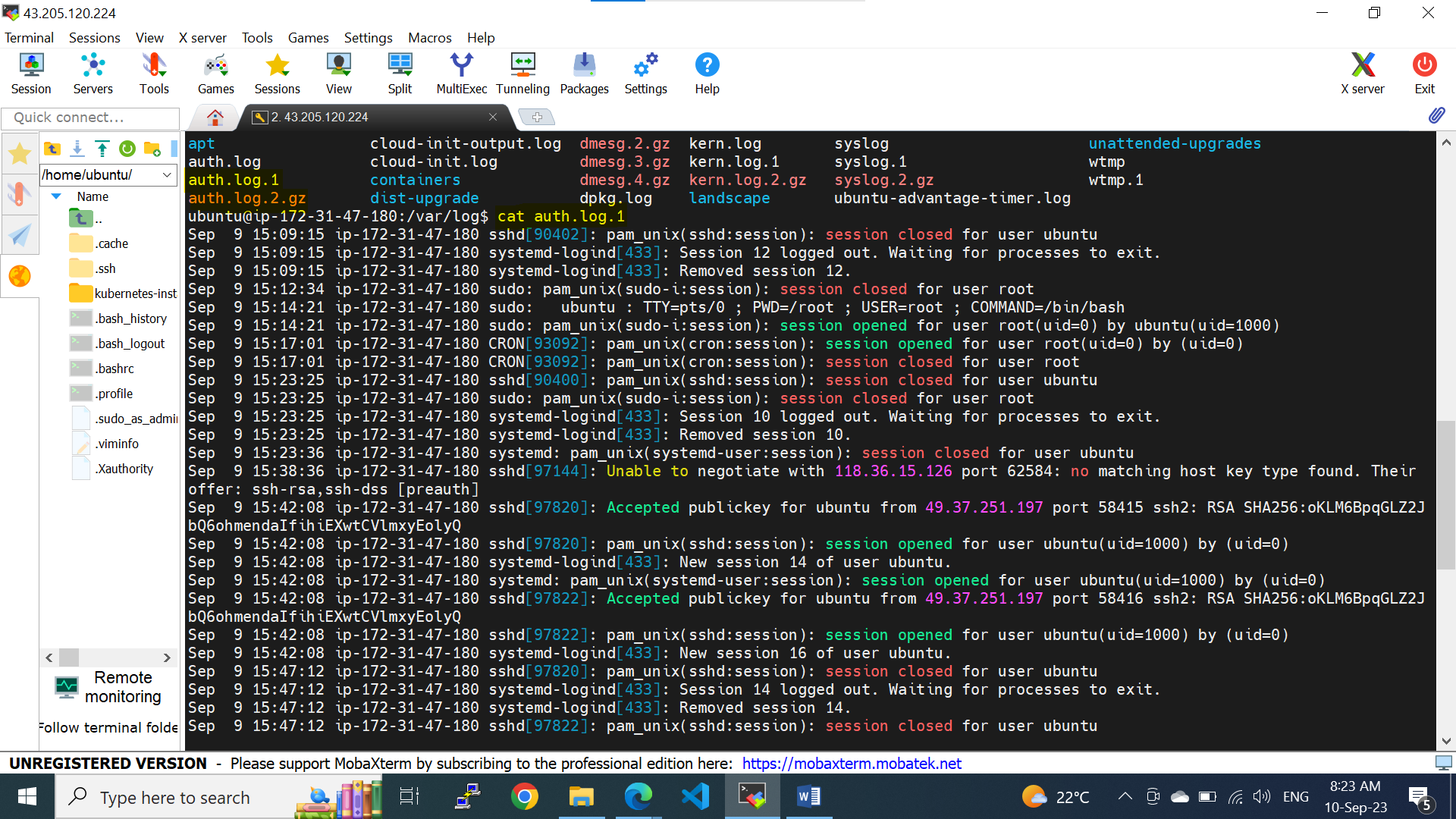
daily

rotate 2

compress

create

Command: sudo logrotate –f /etc/logrotate.conf 🡪 to apply the logrotate manually.



1. **How do you copy s3 bucket from one region to other region? or from one aws account to another aws account.**

**Step1:** Create a Bucket in both Source and Destination Accounts.

**Step2:** Upload some objects into the Source bucket.

**Step 3:** Create a IAM policy in the Destination Account and name it as s3-iam-policy

**#IAM policy in destination account to copy s3 objects from source**

{

"Version": "2012-10-17",

"Statement": [

{

"Effect": "Allow",

"Action": [

"s3:ListBucket",

"s3:GetObject"

],

"Resource": [

"arn:aws:s3:::source-bucket",

"arn:aws:s3:::source-bucket/\*"

]

},

{

"Effect": "Allow",

"Action": [

"s3:ListBucket",

"s3:PutObject",

"s3:PutObjectAcl"

],

"Resource": [

"arn:aws:s3:::destination-bucket",

"arn:aws:s3:::destination-bucket/\*"

]

}

]

}

**Step4:** Create a User in the Destination Account and name it as test-user with programmatic access and attach the created policy i.e., s3-iam-policy

**Step5:** copy the arn of the test-user and create a Bucket policy into the Source Account and paste it at target-accountuserid/role and also edit the source arns in Resources in bucket policy.

**#Source Account bucket policy**

{

"Version": "2012-10-17",

"Statement": [

{

"Sid": "DelegateS3Access",

"Effect": "Allow",

"Principal": {

"AWS": "arn:aws:iam::target-accountuserid/role"

},

"Action": [

"s3:GetObject",

"s3:PutObject",

"s3:ListBucket"

],

"Resource": [

"arn:aws:s3:::source-bucket-arn/\*",

"arn:aws:s3:::source-bucket-arn"

]

}

]

}

Step6: install aws cli in local machine or linux machine and configure the **test-user** from Destination Account into the machine by using a command: aws configure

Provide the access key, secret access key and region

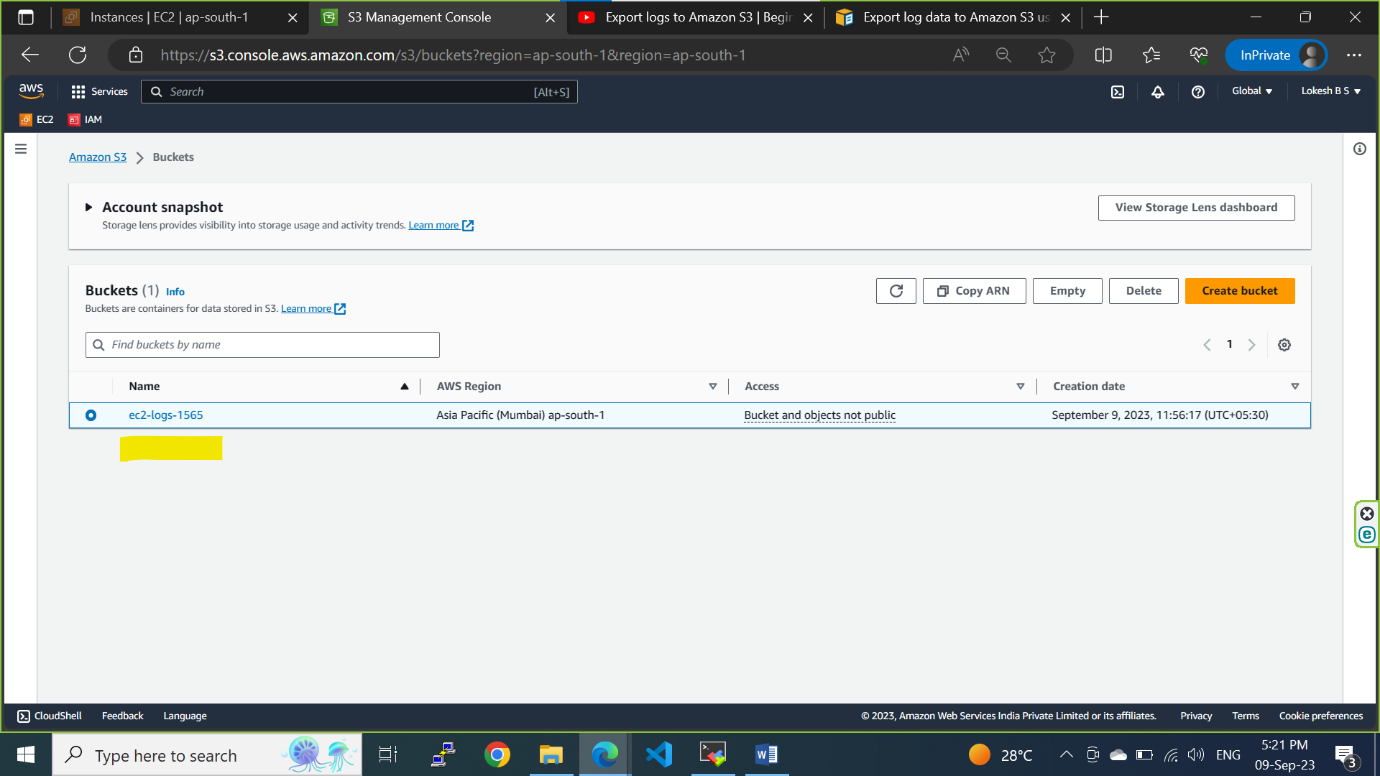
Aws cli command to copy objects from source account to destination account:

**aws s3 sync s3://<source\_bucket\_name> s3://<destination\_bucket\_name>** 🡪if both accounts are in same region

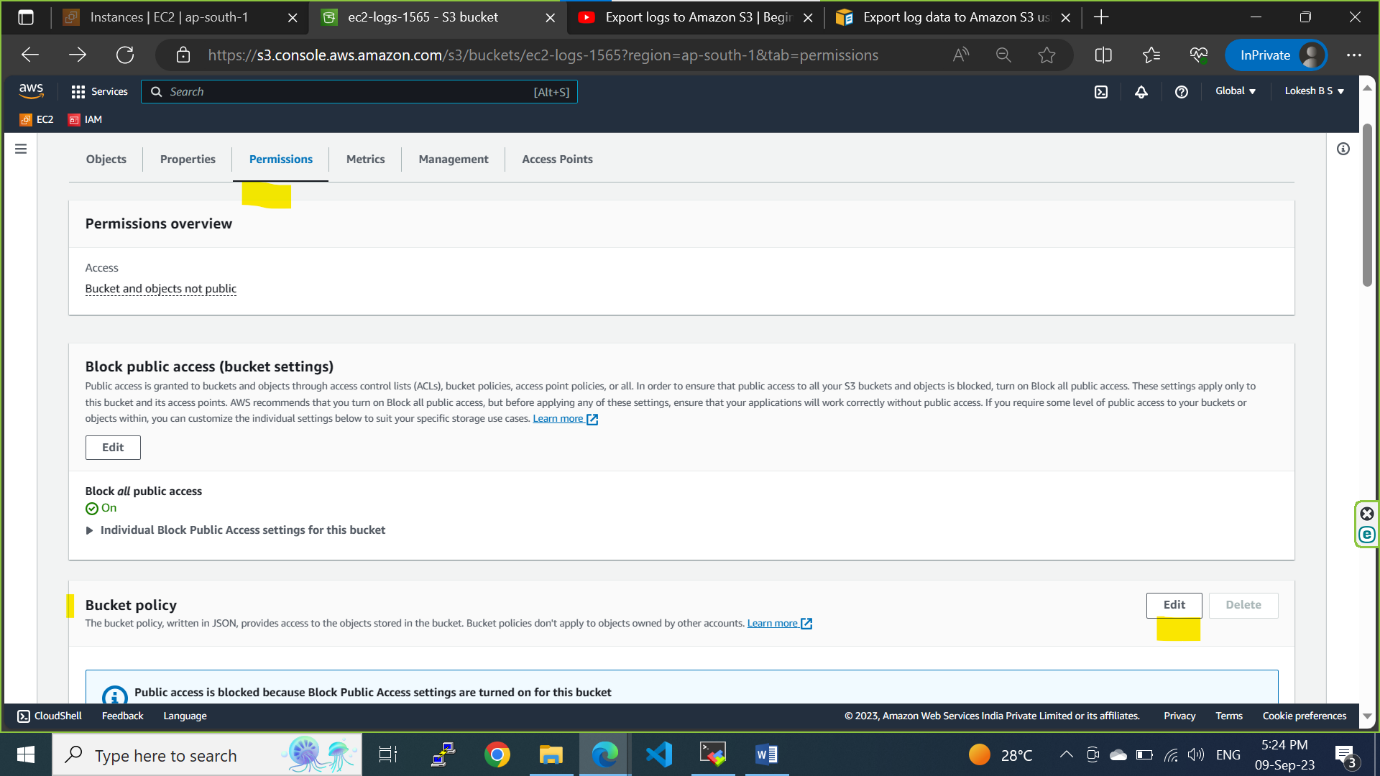
**aws s3 sync s3://<source\_bucket\_name> s3://<destination\_bucket\_name> --source-region <source\_region> --region <destination\_region>** 🡪 if both accounts are in different regions**.**

1. **How can I export logs from cloudwatch to s3 after 7days using Lambda function?**

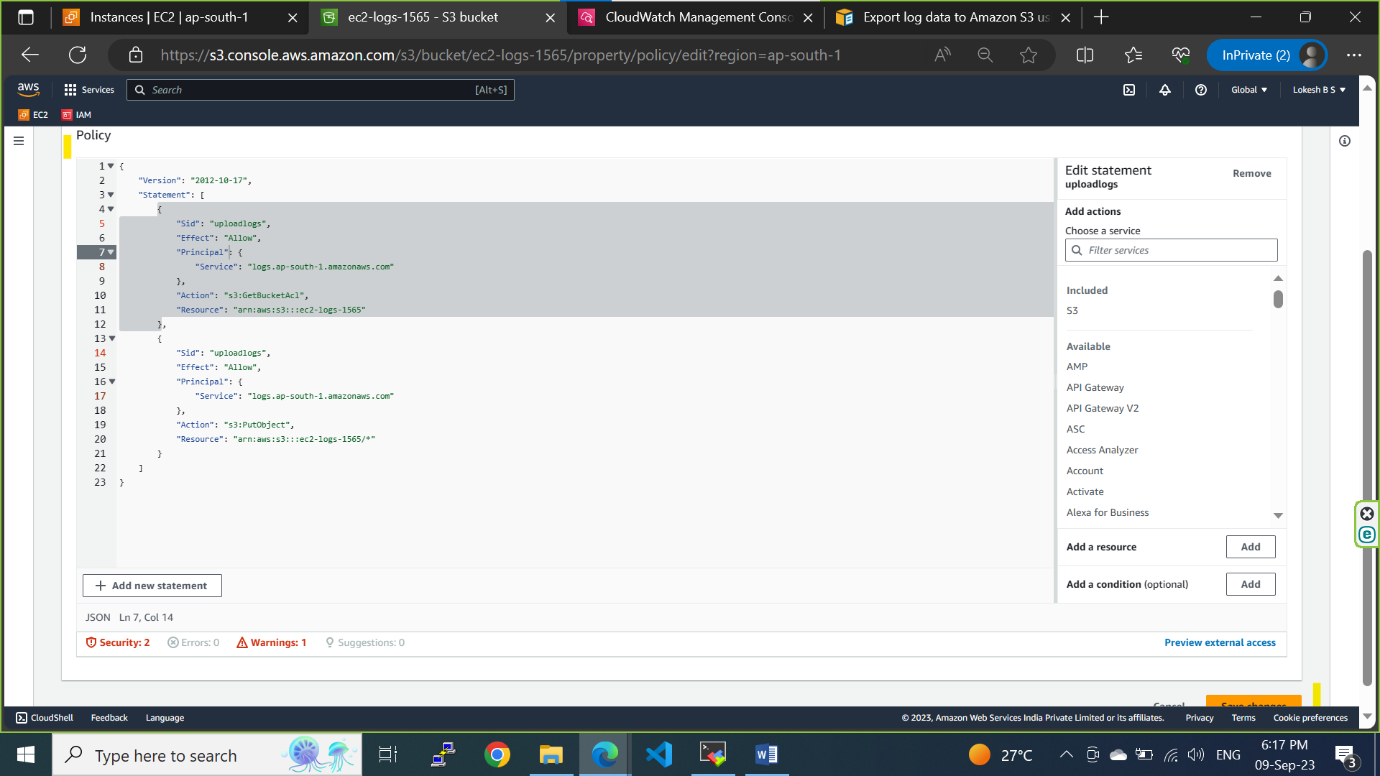
**Step1: Create a bucket in s3 to store the logs**

****

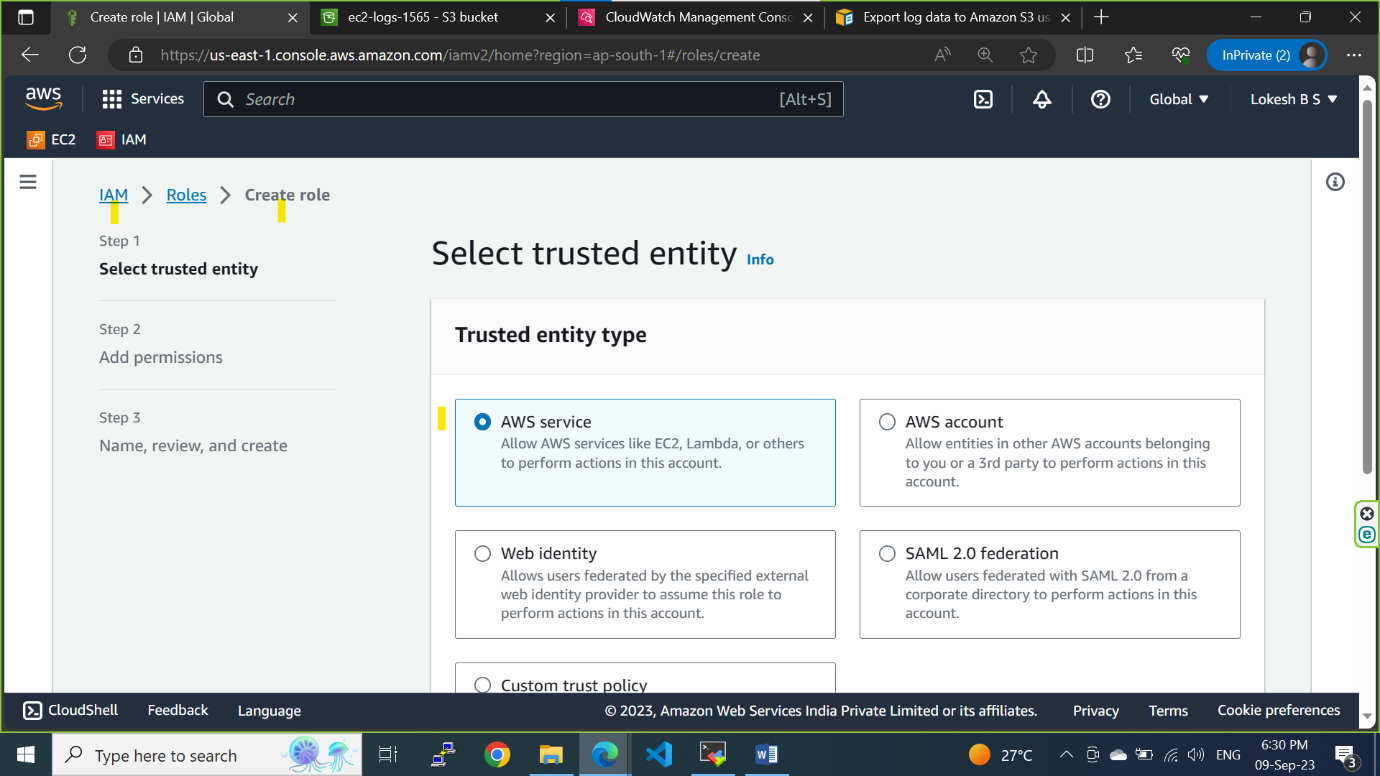
**Step2: click on permission tab and click on edit policy to write a policy to give Bucket access to CloudWatch**

****

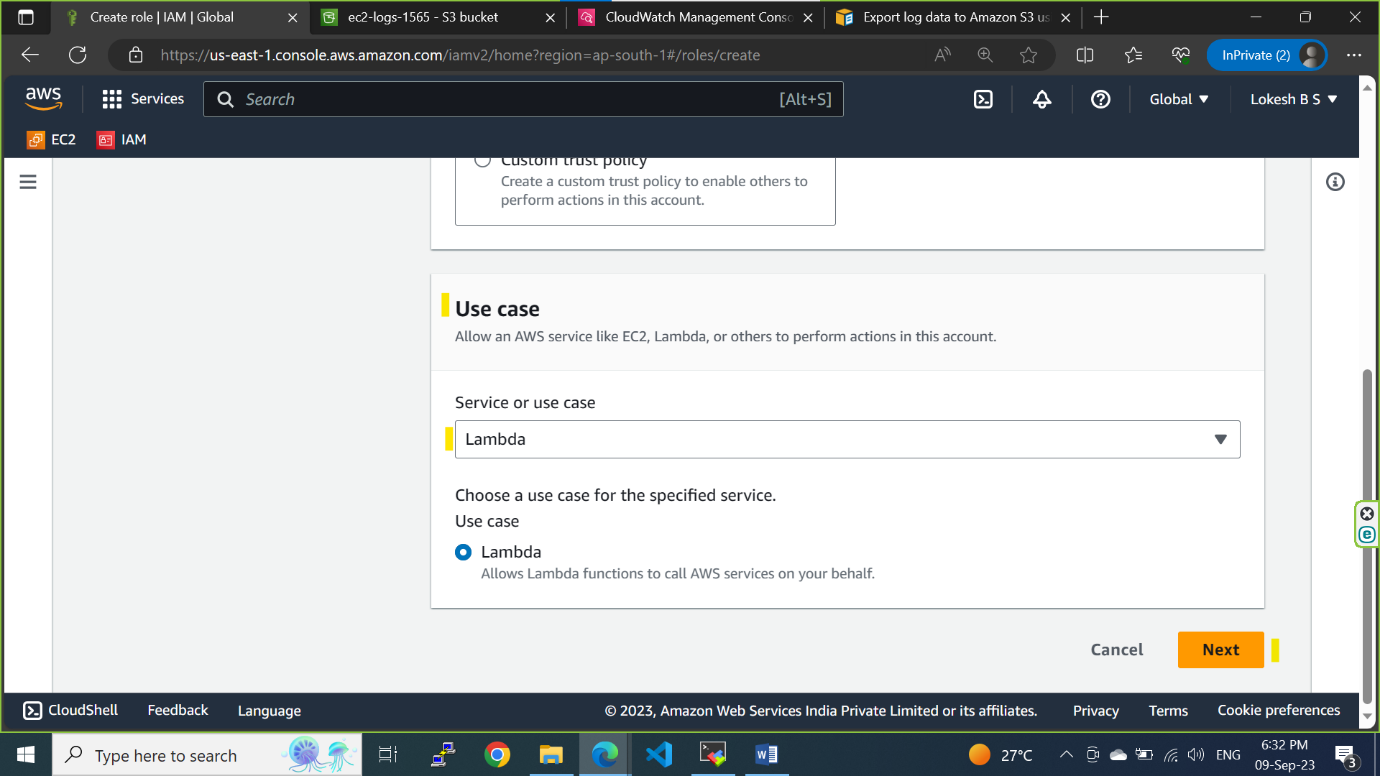
Github repo for Bucket Policy : [PythonHacks/AWS\_Export\_Cloudwatchlogs\_to\_s3\_bucket/export\_cloudwatchlogs\_to\_s3.py at main · RekhuGopal/PythonHacks · GitHub](https://github.com/RekhuGopal/PythonHacks/blob/main/AWS_Export_Cloudwatchlogs_to_s3_bucket/export_cloudwatchlogs_to_s3.py)

****

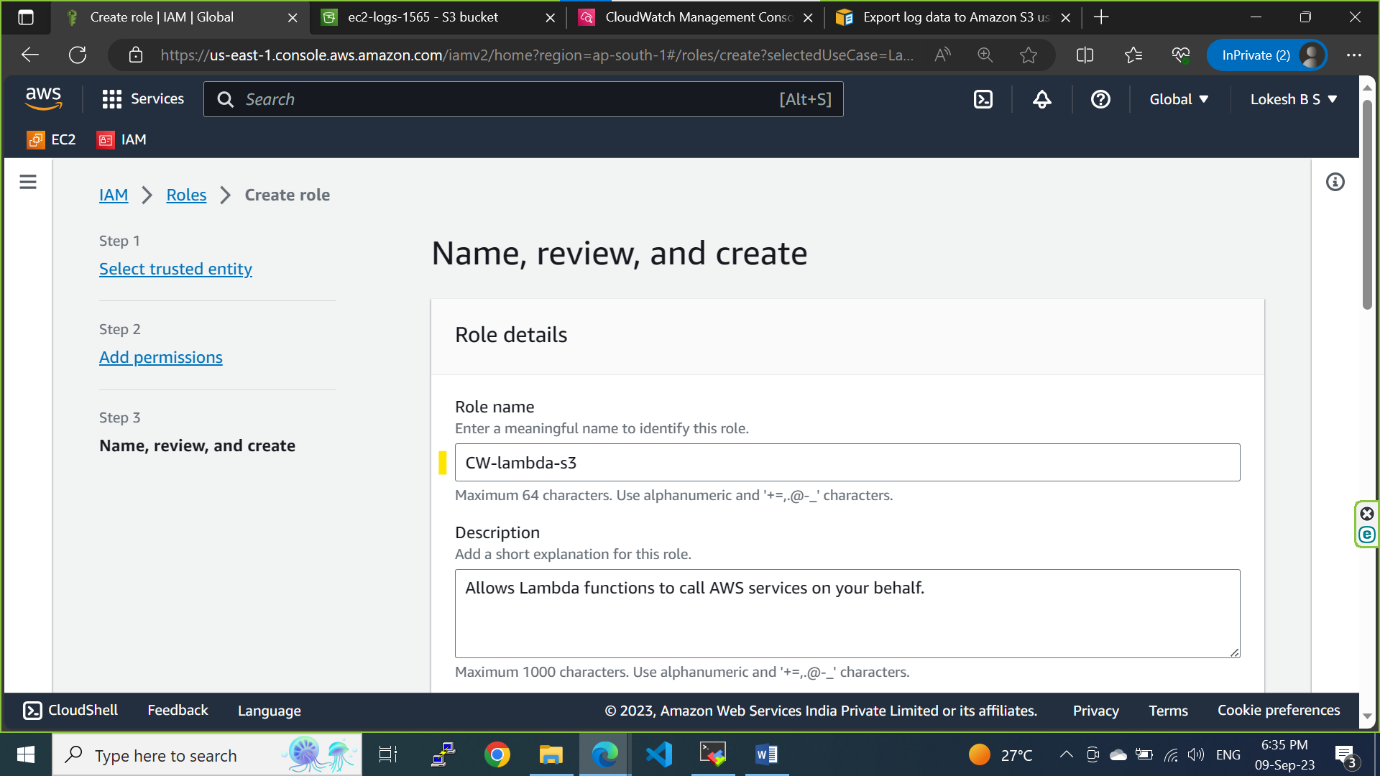
**Step3: Create a IAM role for Lambda to give the access to CloudWatch and s3**

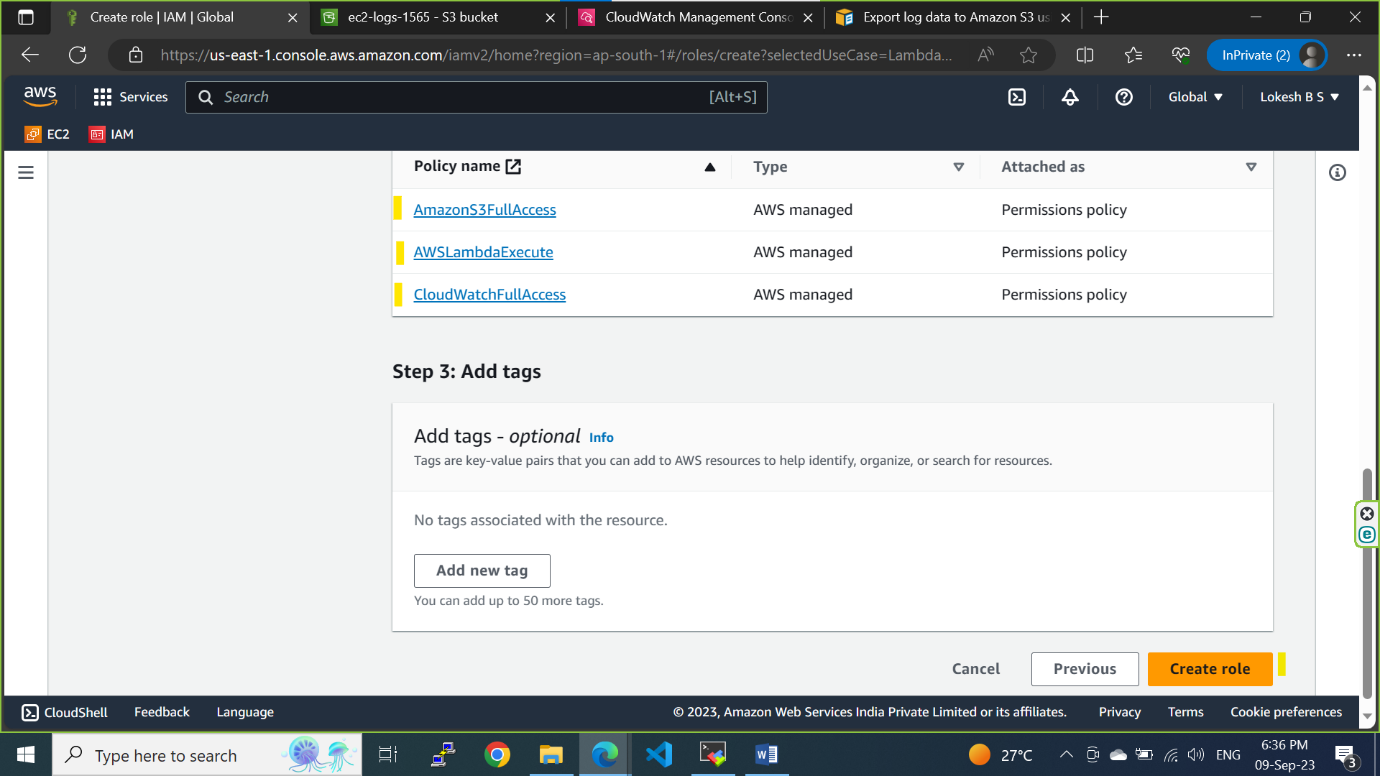
****

**Choose a Lambda for a service**

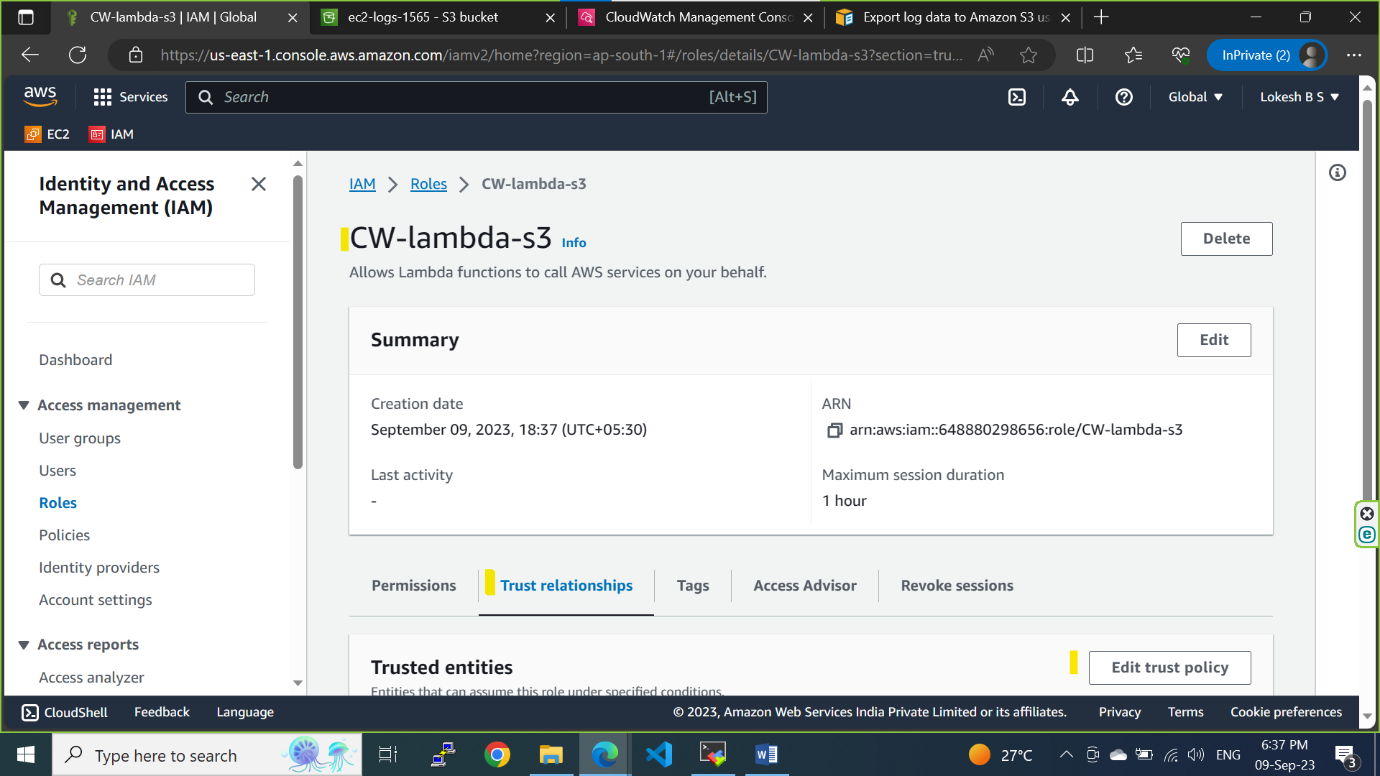
****

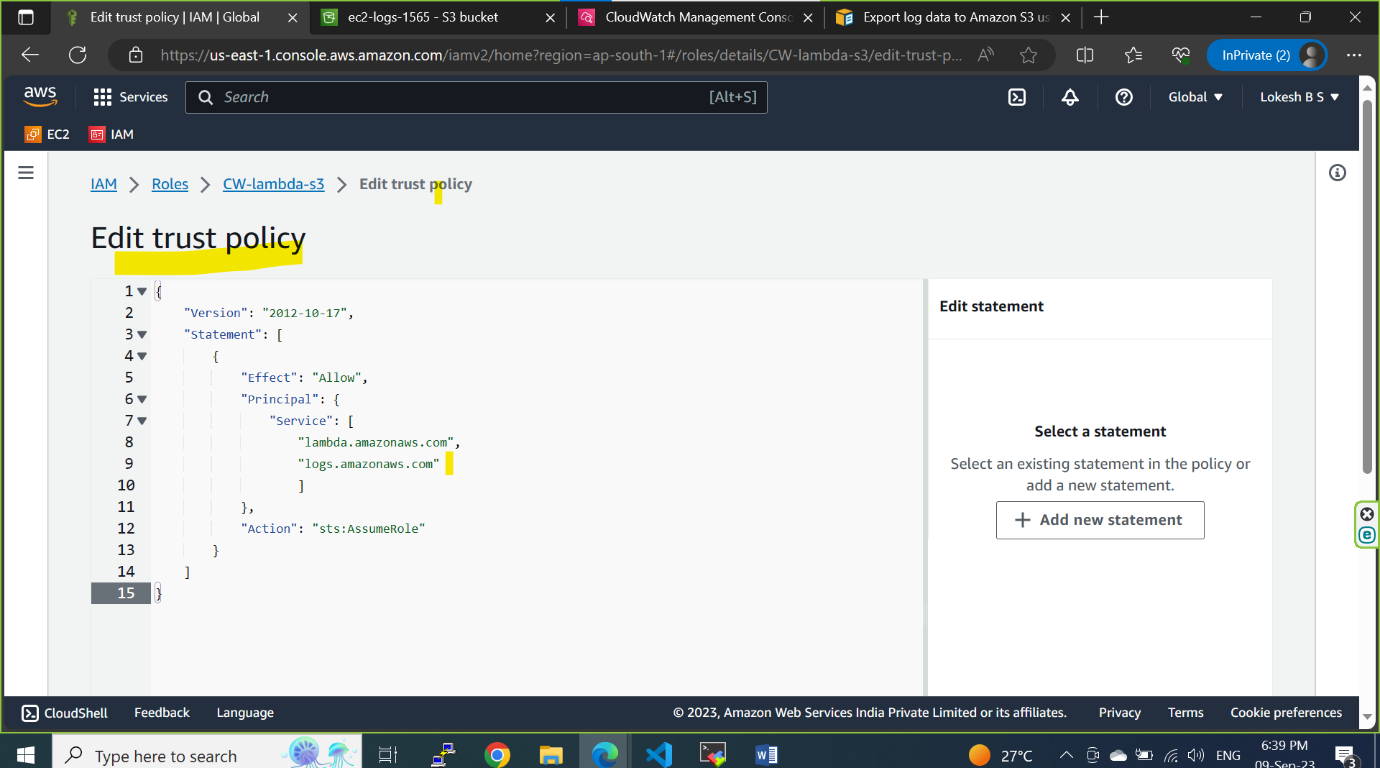
**Give 3 roles and select 3 policies**

****

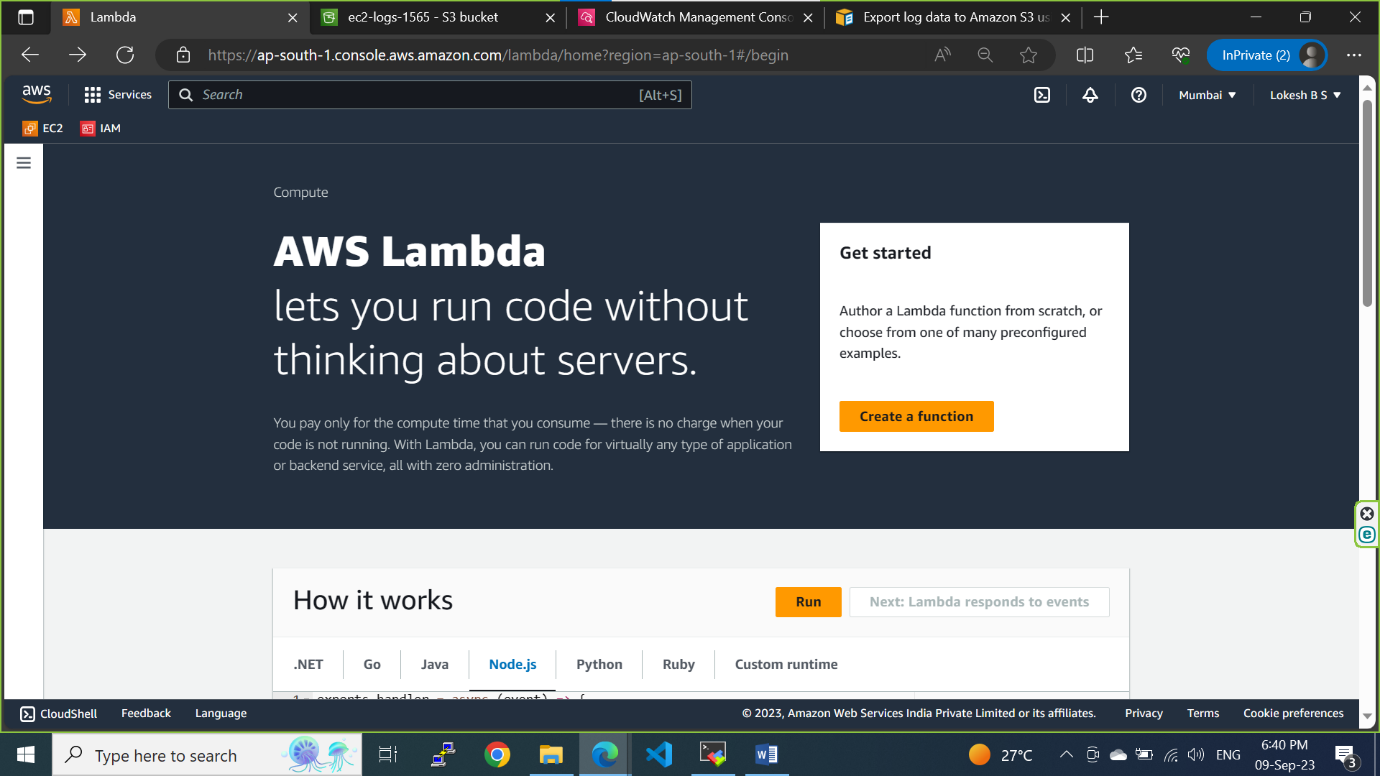
****

**Goto trusted policy and click on edit**

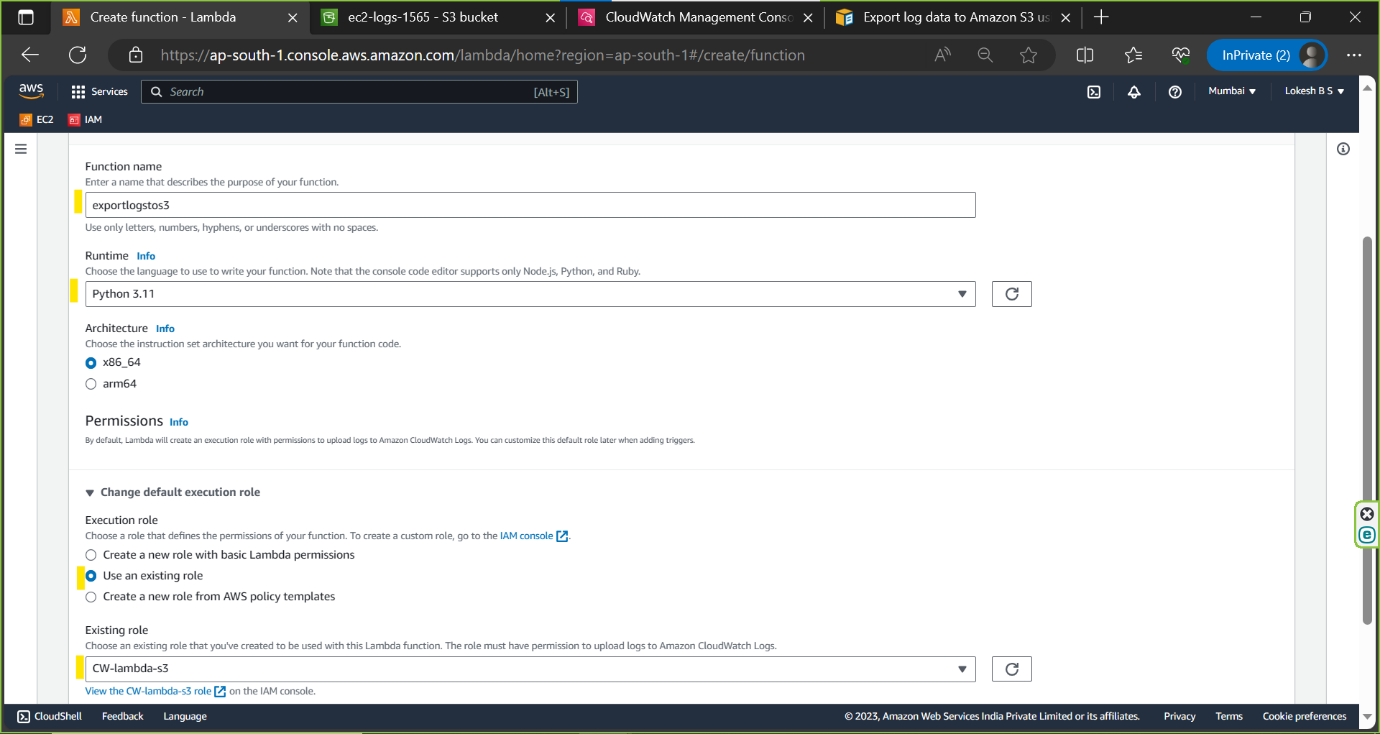
****

****

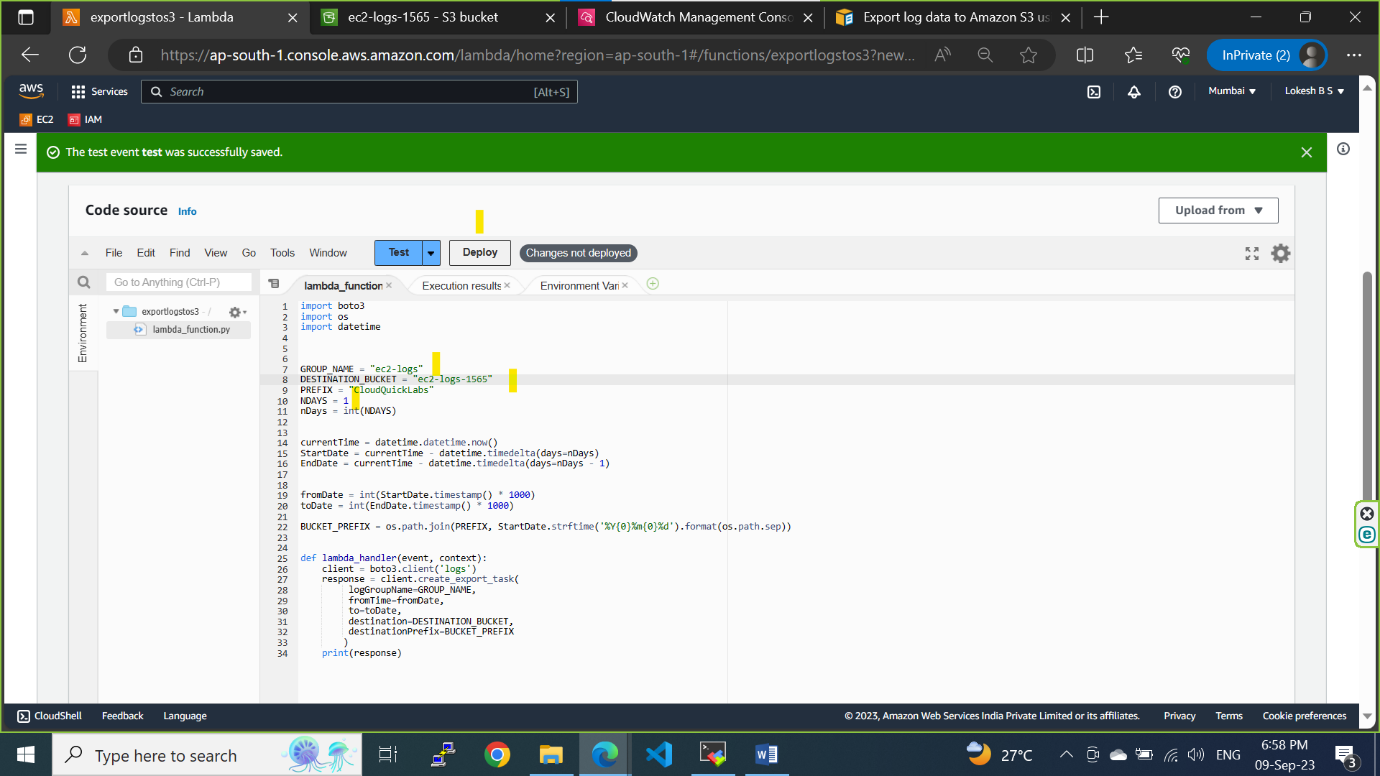
**Step4: go to AWS Lambda console and click on Create a function**

****

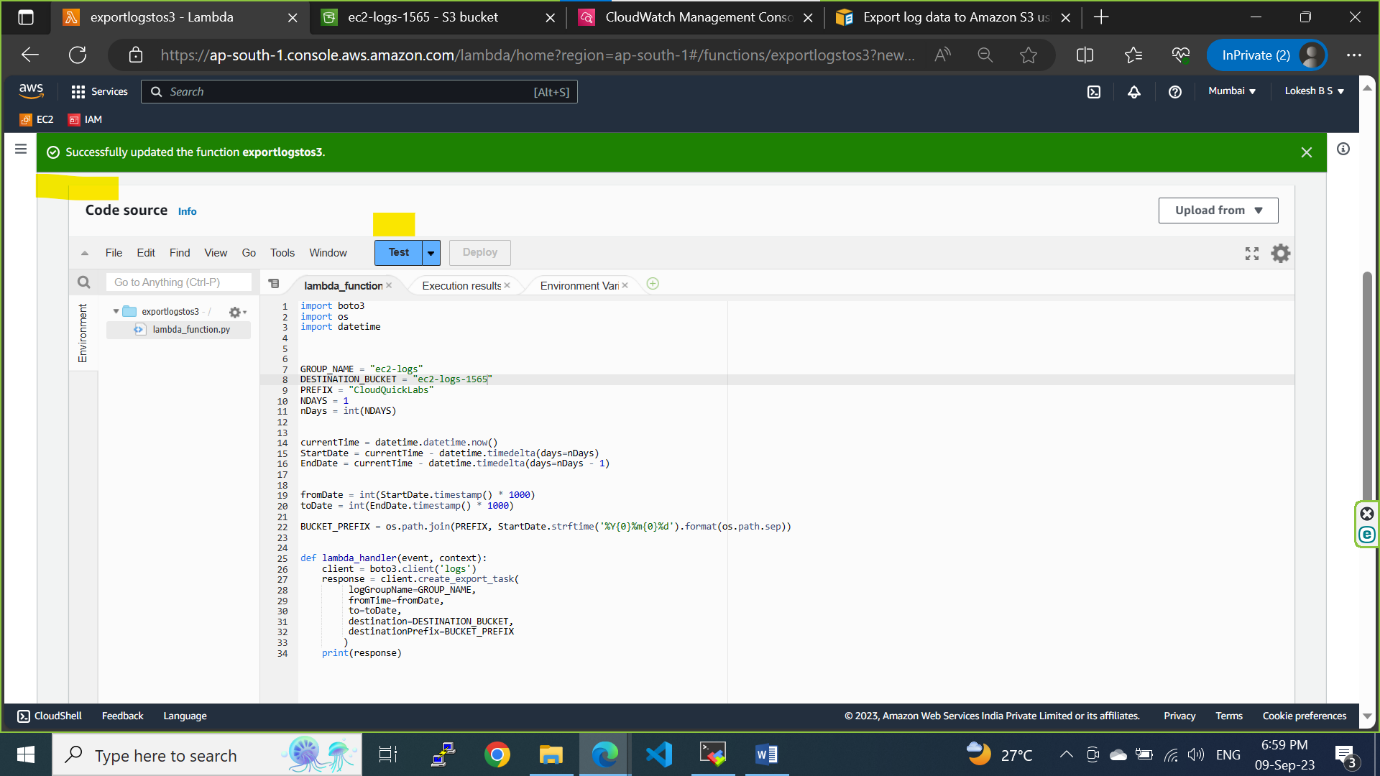
**Give the function details and use an existing role.**

****

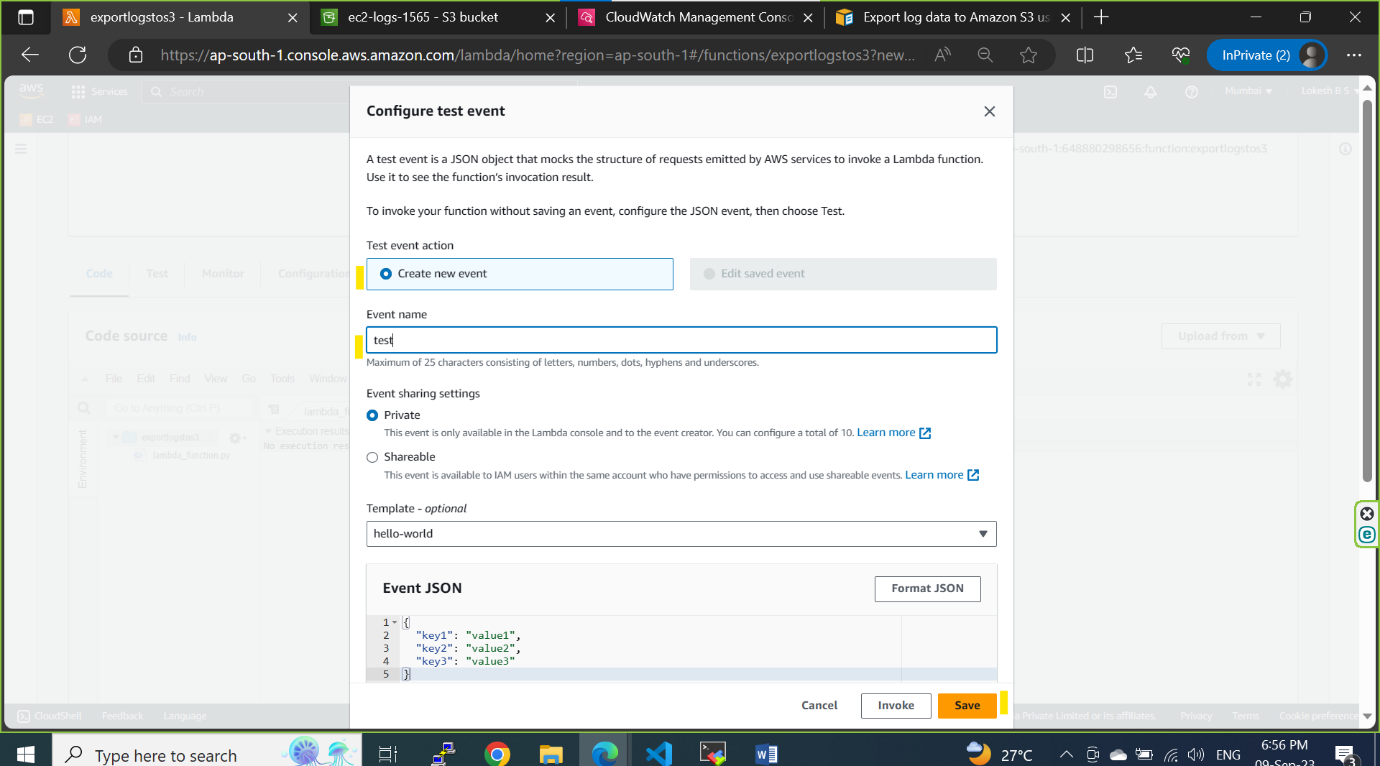
**Step5: write a python code to automate the logs export to s3 from CloudWatch**

****

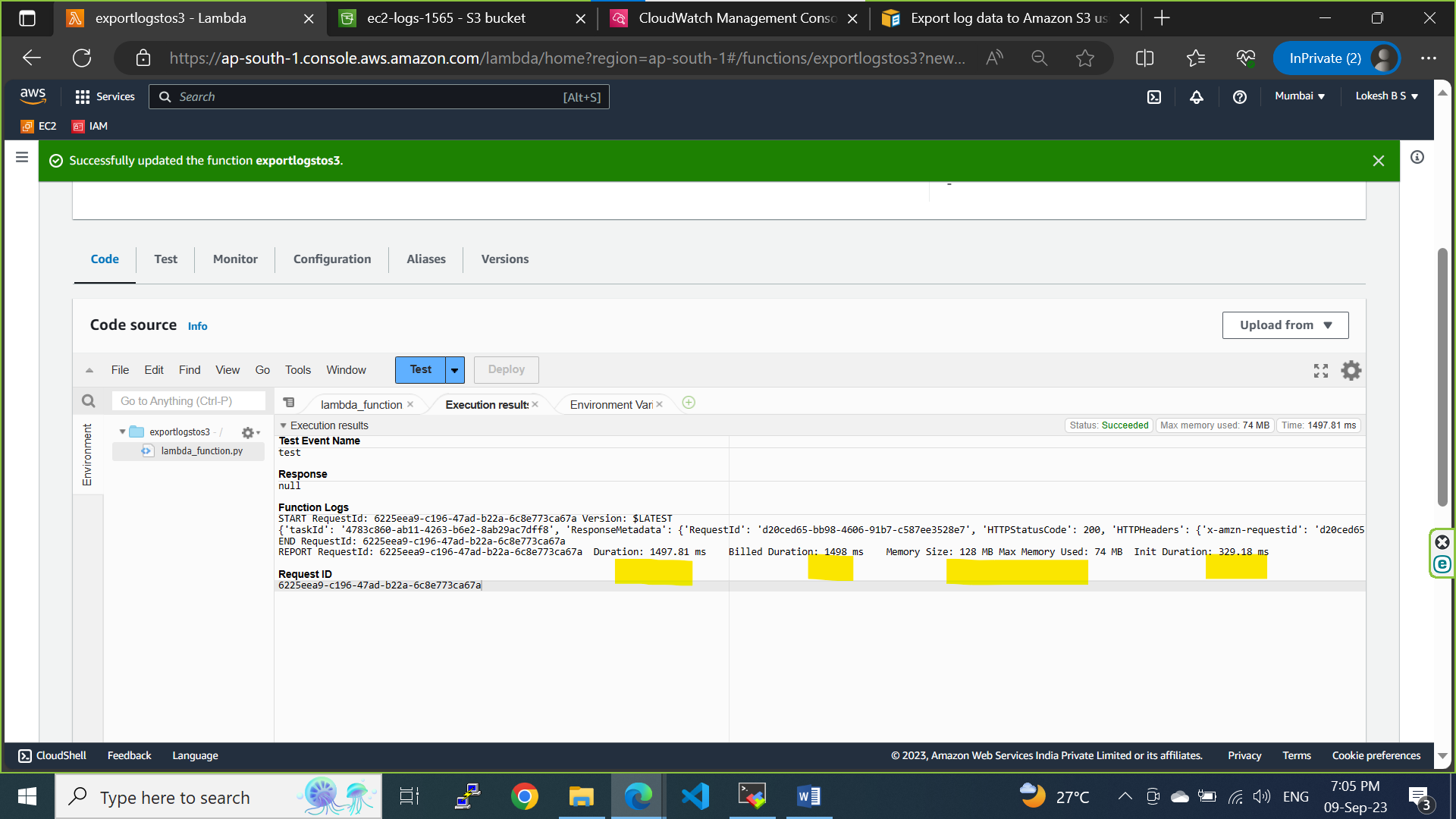
**Once code is deployed successfully click on test and create a event to start the lambda execution.**

****

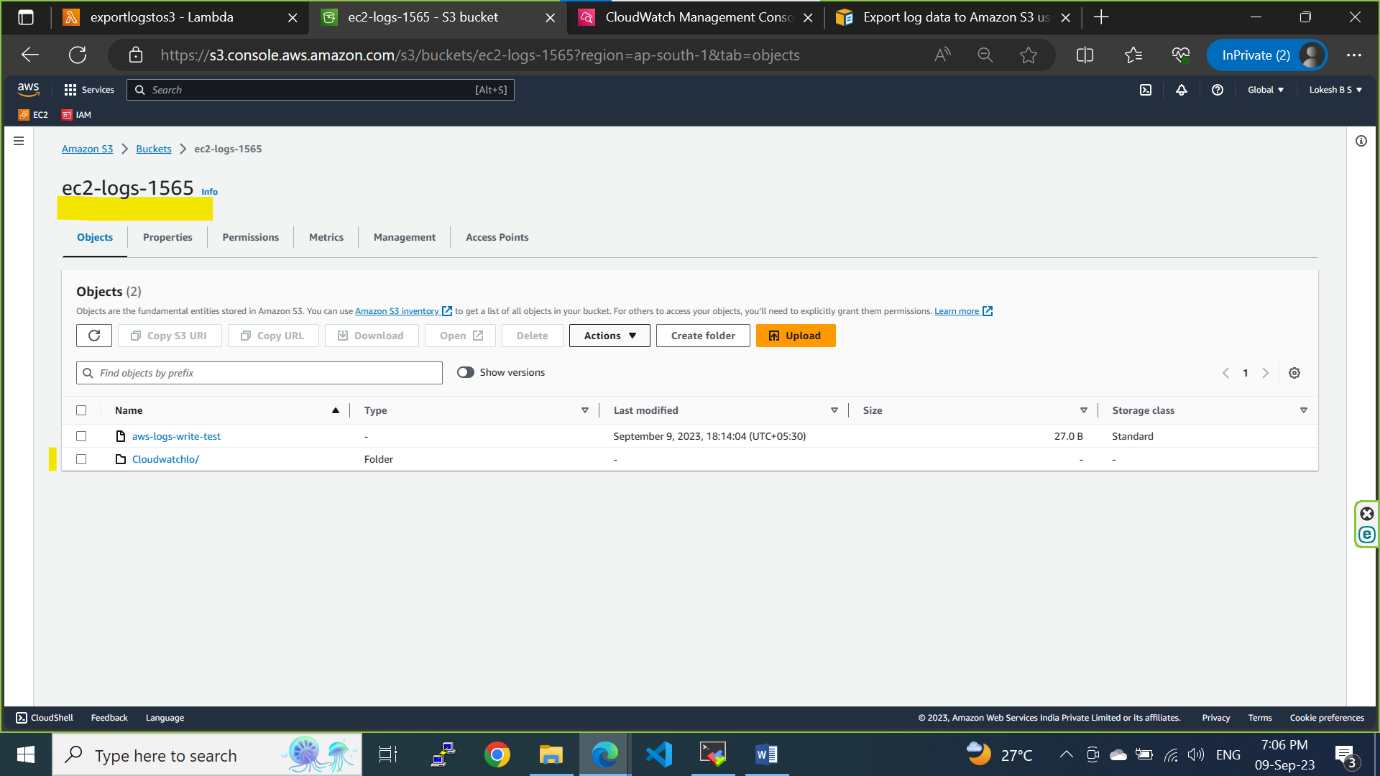
**Create a event and save**

****

**Execution completed successfully**

****

**Logs exported from CloudWatch successfully**

****

**ASSIGNMENT – 11-09-2023**

**Mount configMap and Secret as volume into the pod’s file system:**

**Step1: create a configMap using manifest file and apply.**

apiVersion: v1

kind: ConfigMap

metadata:

name: config-dev

data:

user: devops

role: qa

os: ubuntu

**Step2: Create a secret using manifest file and apply.**

apiVersion: v1

kind: Secret

metadata:

name: dev-password

data:

password: test

**Step3: create a pod by mounting configMap and secret as volume.**

apiVersion: v1

kind: Pod

metadata:

name: nginx

spec:

volumes:

- name: config-volume

configMap:

name: config-dev

- name: secret-volume

secret:

secretName: dev-password

containers:

- name: nginx

image: nginx

ports:

- containerPort: 80

volumeMounts:

- name: config-volume

mountPath: /etc/config

- name: secret-volume

mountPath: /etc/secret

command to check configMap and secret is mounted as volume.

kubectl exec –it nginx – bash

cd /etc/config 🡪 ls

cd /etc/secret 🡪 ls