**DR. VIRENDRA SWARUP INSTITUTE OF COMPUTER STUDIES**

****

**BACHELOR OF COMPUTER APPLICATION**

**BATCH(2023-2026)**

**PROJECT-1**

**TOPIC- SERVERLESS IMAGE PROCESSING**

**SUBMITTED BY:- SUBMITTED TO:-**

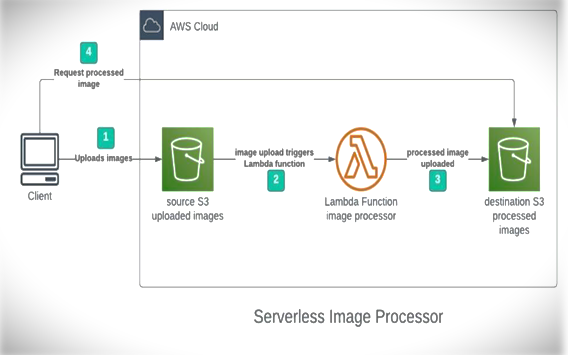
**Aditi Pandey Mr. Abhinav Dwivedi**

**BCA SEMESTER-4**

**Team Name :- RHYTHM**

**Serverless Image Processing**

The Serverless Image Handler solution helps you embed images on your websites and mobile applications to drive user engagement. It uses the [sharp](https://sharp.pixelplumbing.com/en/stable/) Node.js library to provide high-speed image processing without sacrificing image quality. To minimize your costs of image optimization, manipulation, and processing, this solution automates version control and provides flexible storage and compute options for file reprocessing

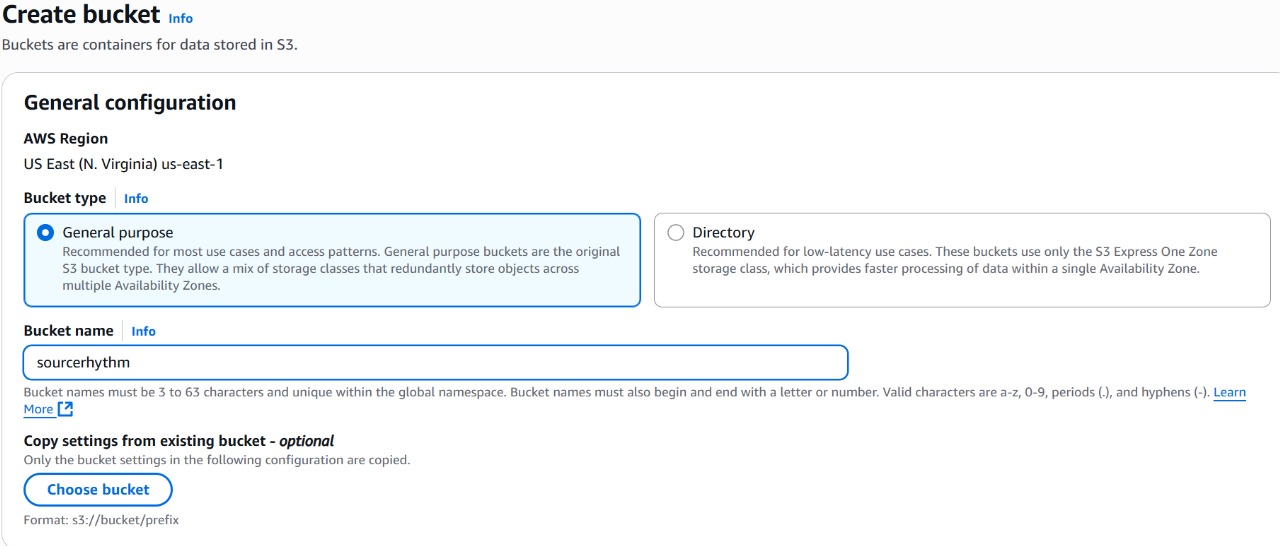


## Step 1 – Creating S3 buckets

### We will use two S3 buckets:

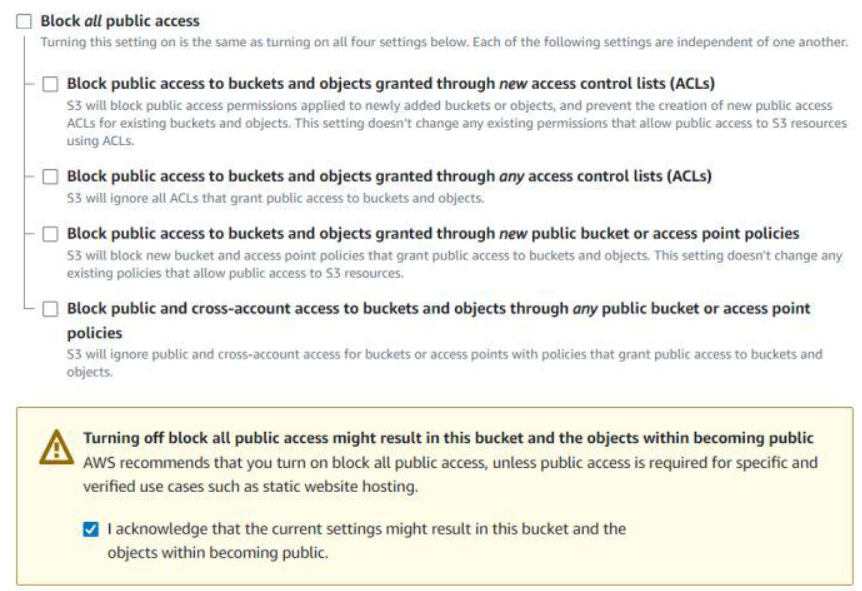
1. source Bucket: For storing uploaded images.

2. destination Bucket: For storing processed images.



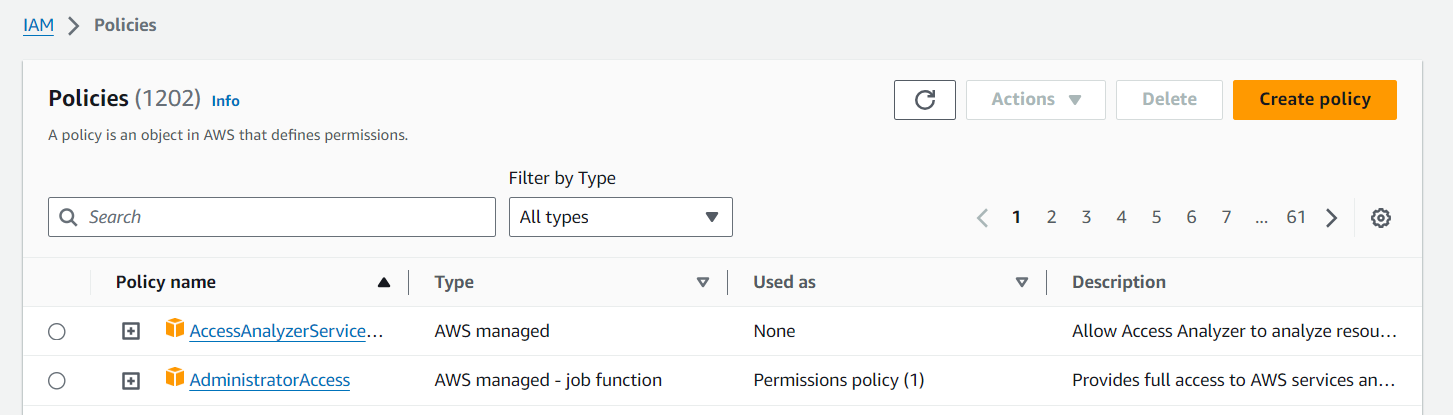
**Step 2 – Configuring S3 bucket policy**

**In ‘Block Public Access settings for this bucket’ section disable “block all public access”.**

****

**Step 3 – Creating police in Iam**

**Go to AWS I am console. Navigate to policies section. Click Create policies in (JSON) and name it “ImageBucketpolicy”. Leave all other settings as default. Create the policy.**

****

**Policy:**

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

"Statement": [

{

"Effect": "Allow",

"Action": [

"logs:PutLogEvents",

"logs:CreateLogGroup",

"logs:CreateLogStream"

],

"Resource": "arn:aws:logs:\*:\*:\*"

},

{

"Effect": "Allow",

"Action": ["s3:GetObject"],

"Resource": "arn:aws:s3:::BUCKET\_NAME/\*"

},

{

"Effect": "Allow",

"Action": ["s3:PutObject"],

"Resource": "arn:aws:s3:::DEST\_BUCKET/\*"

}

]

}

**\*BUCKET\_NAME = SOURSE BUCKET NAME**

**\*DEST\_BUCKET = DESTINATION BUCKET NAME**

**Step 4 –creating role in I am**

**Following Steps are Follows**

\*Go to aws I am console

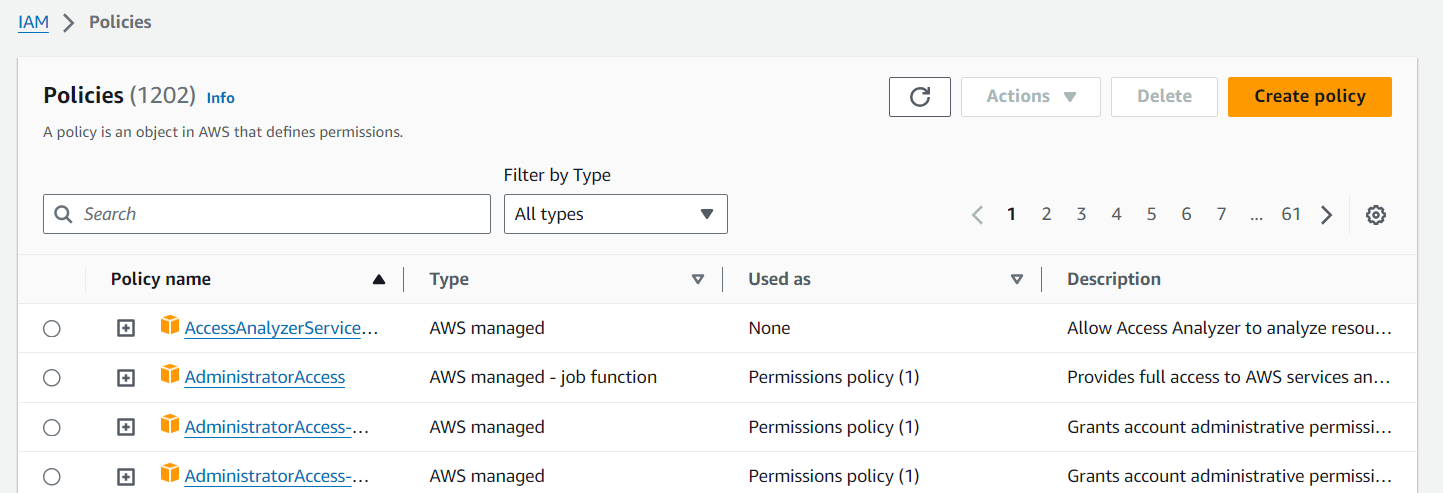
\*Create role

\*name imageresizerlambdarole

\*Use case – Lambda

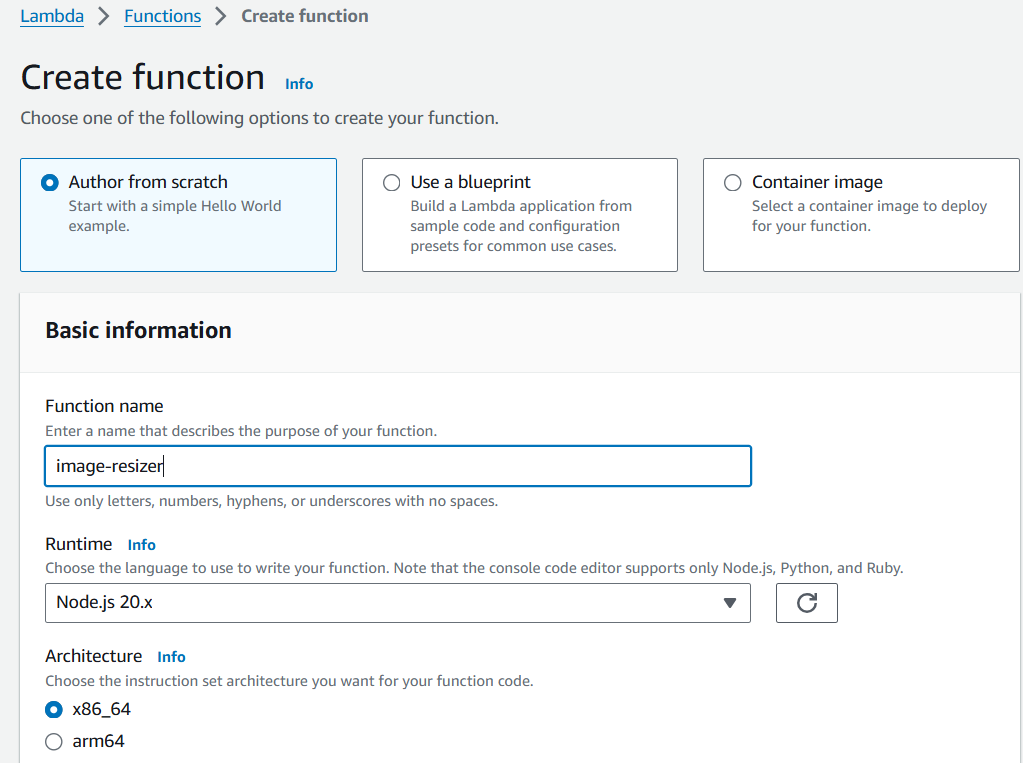
\*Select-ImageBucketPolicy

\*Then create role

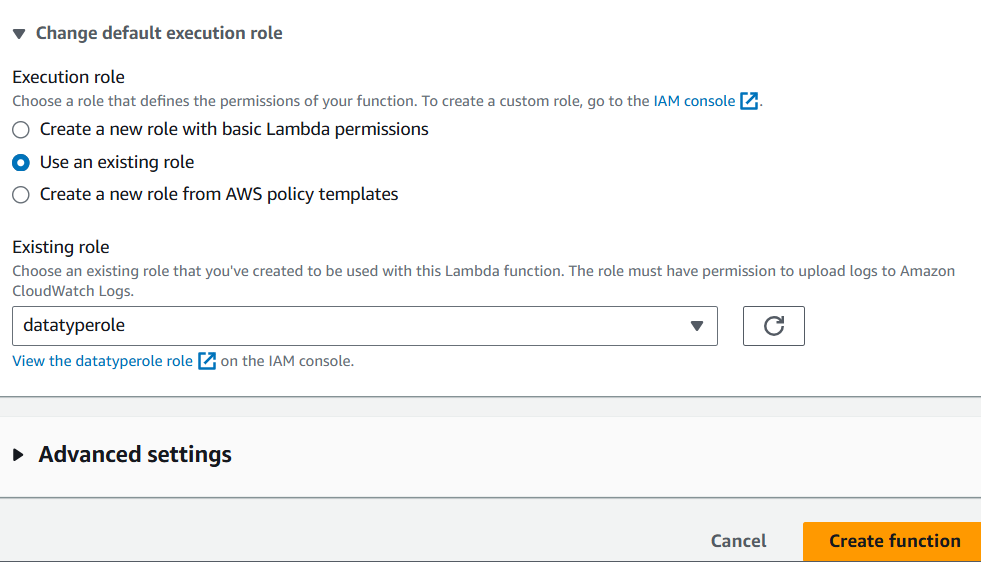
****

**Step 3 – Creating Lambda function**

Go to AWS Lambda console. Navigate to Functions section. Click Create Function and name it “ImageProcessing”. Select runtime as “NodeJS 16.x” and architecture as “x86\_64”. Leave all other settings as default. Create the function.

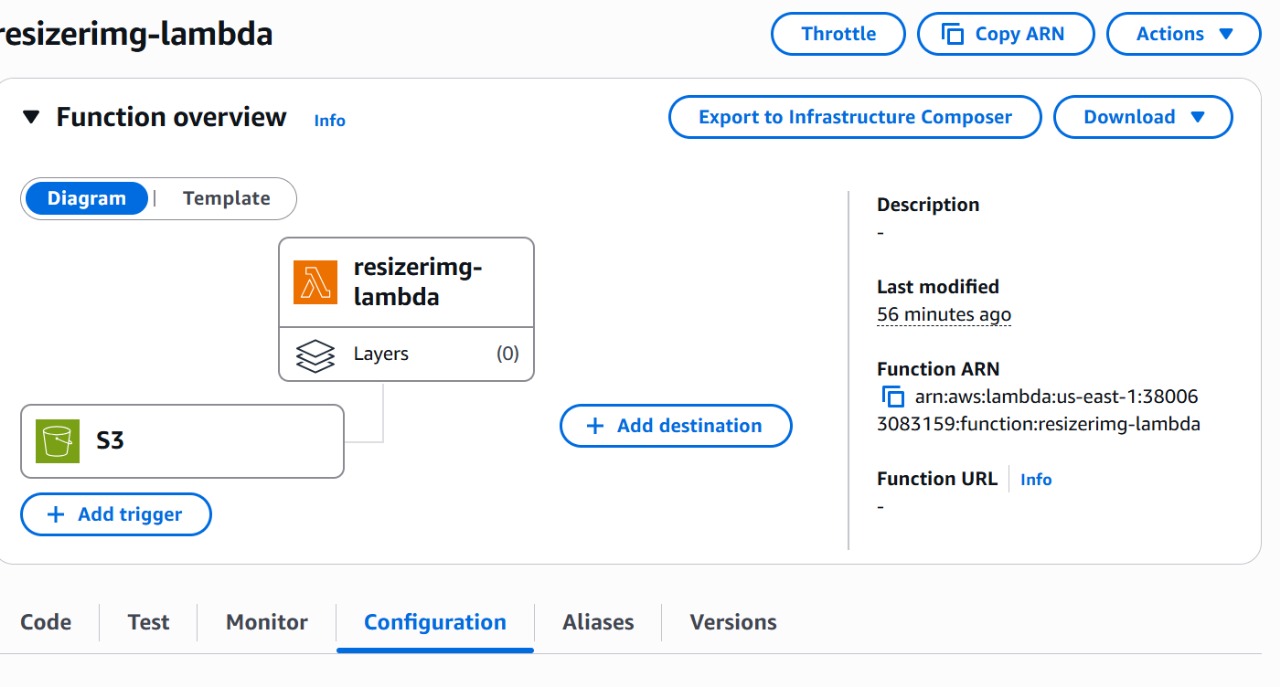


**\*Change default execution role**



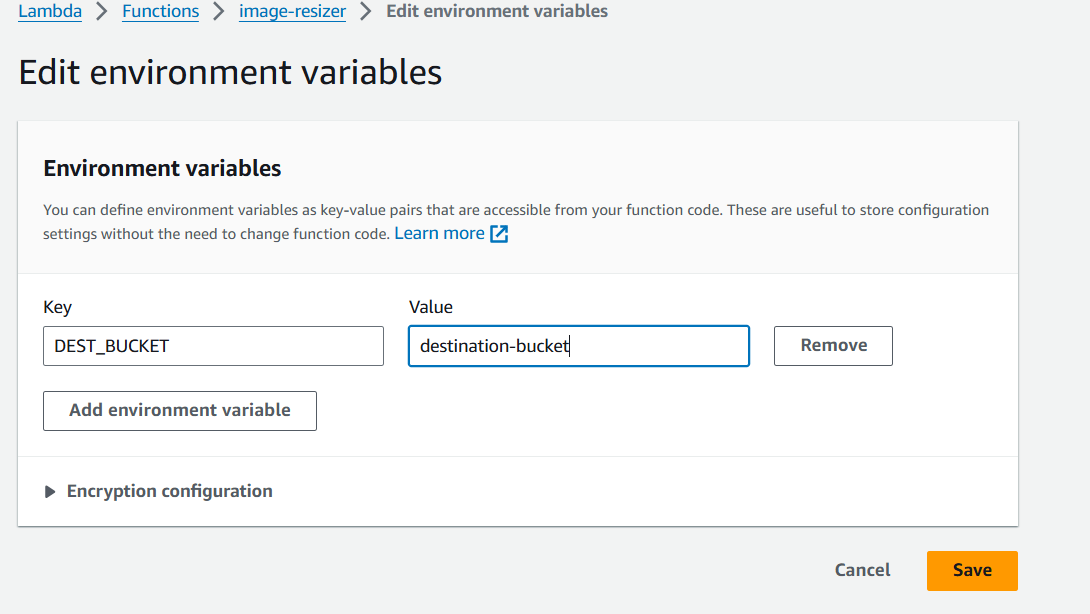
**\*Create function**

**Step 6 -upload zip file in Lambda function**

****

\*Zip file link-https://github.com/OneLightWebDev/image-resizer-lambda

**Step 7** - Edit environment variables



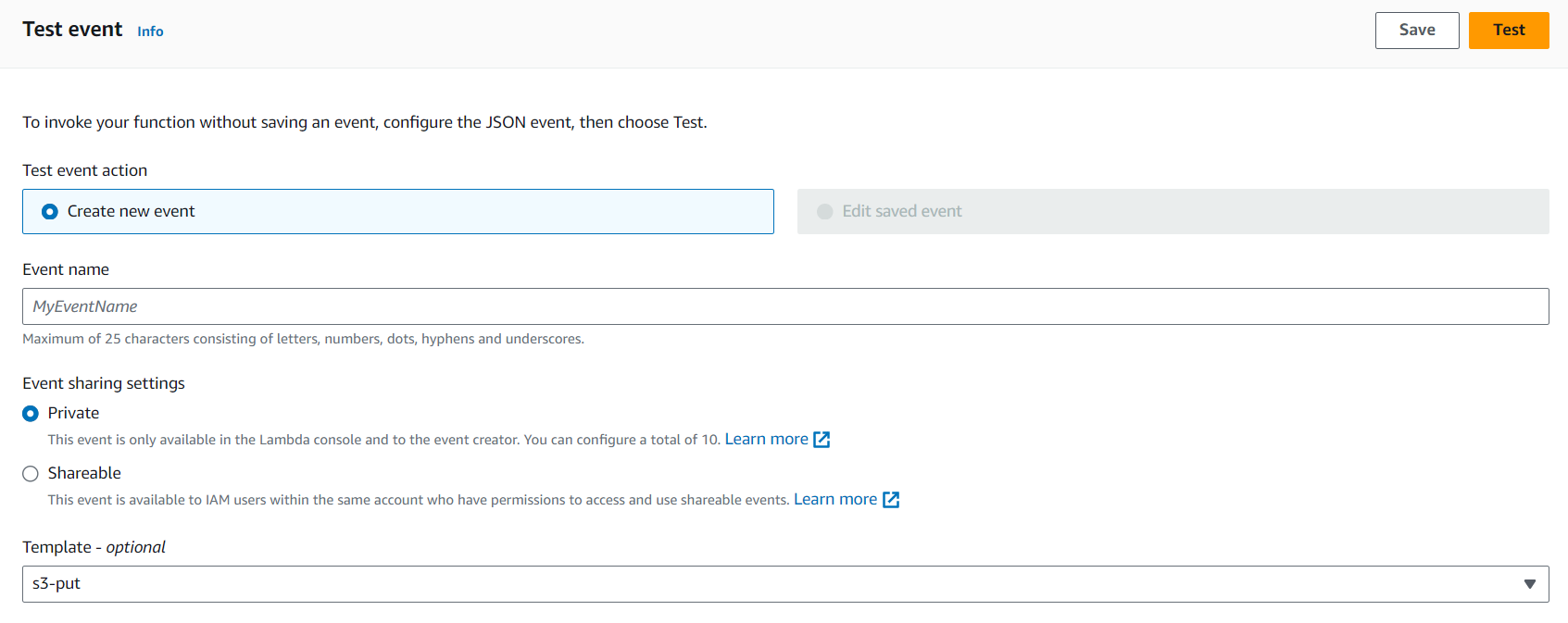
**Step 8 – Test Lambda Function**

\*Go to AWS Lambda console. Navigate to Functions section.

\*open function then will be created

\*open test console

\*template=s3-put



**EVENT JSON**

In event json we can change only 3 value

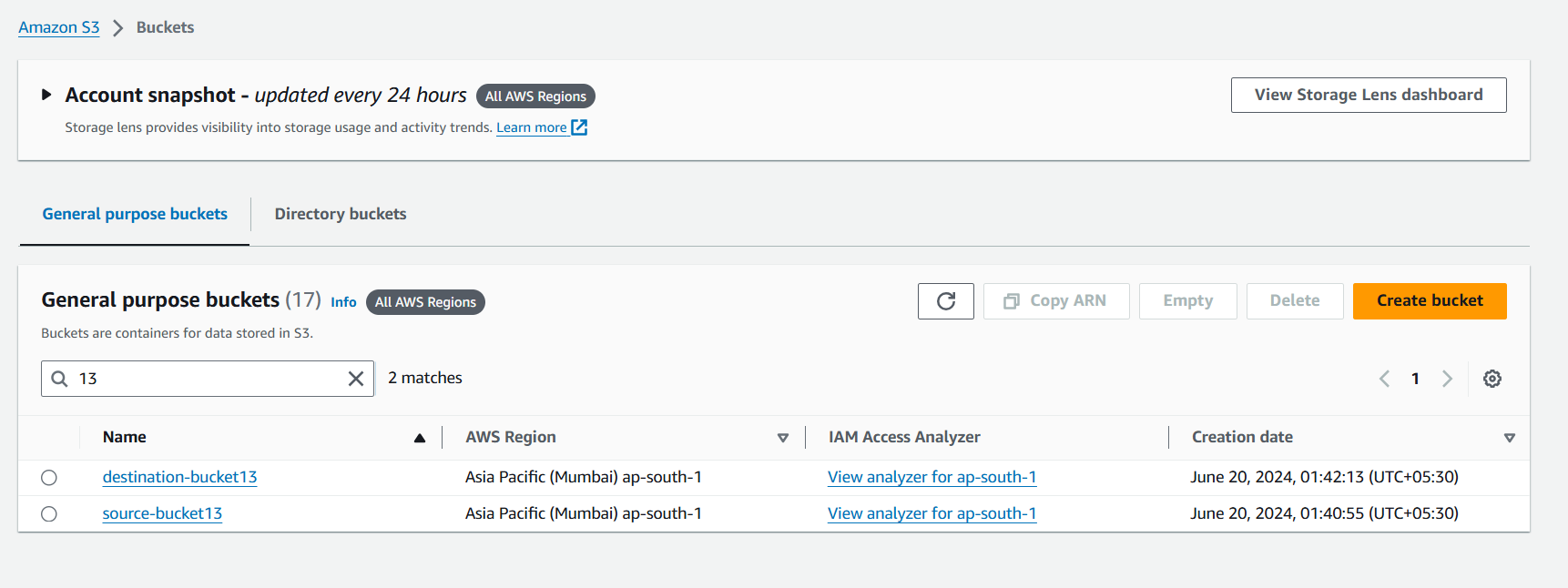
\*name, arn, key

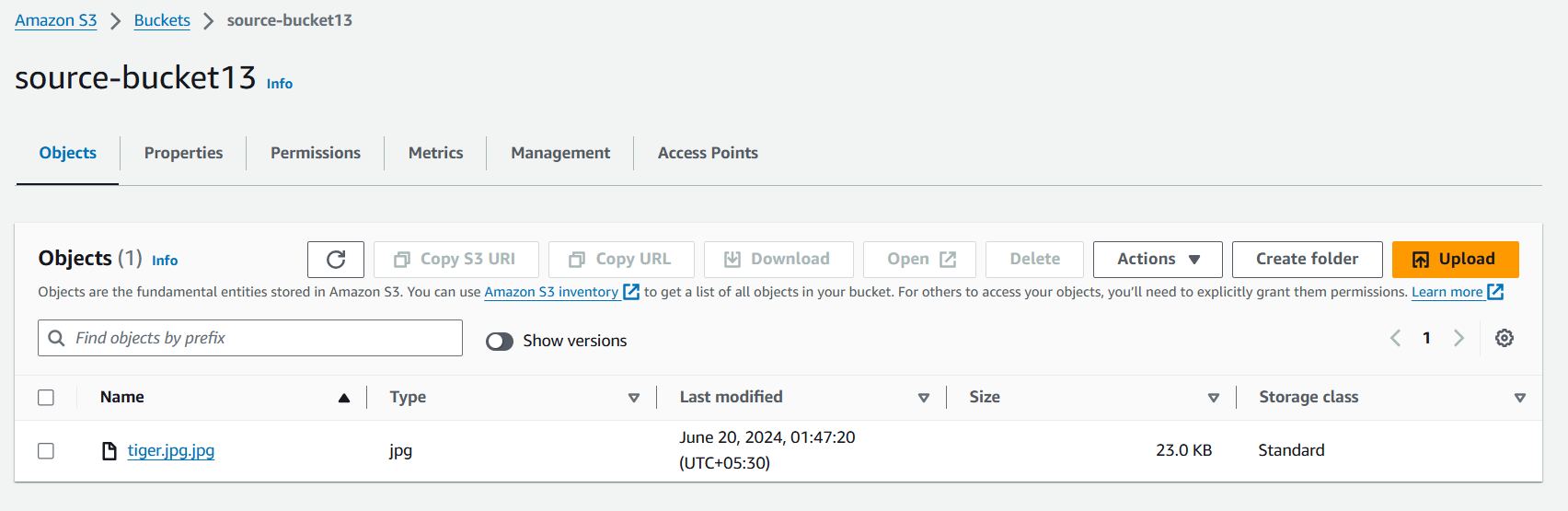


"name":"demo-user-images-bucket",->

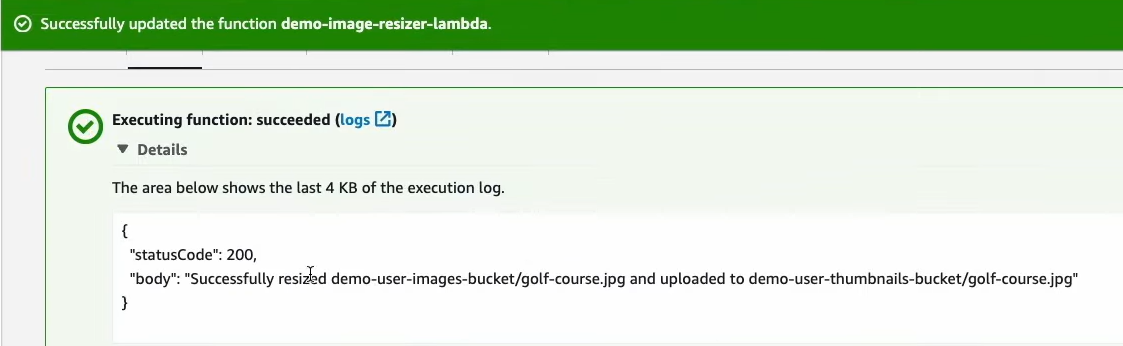
"arn": "arn:aws:s3::: demo-user-images-bucket "->

"key": "golf-course.jpg",->





**Now we can test**



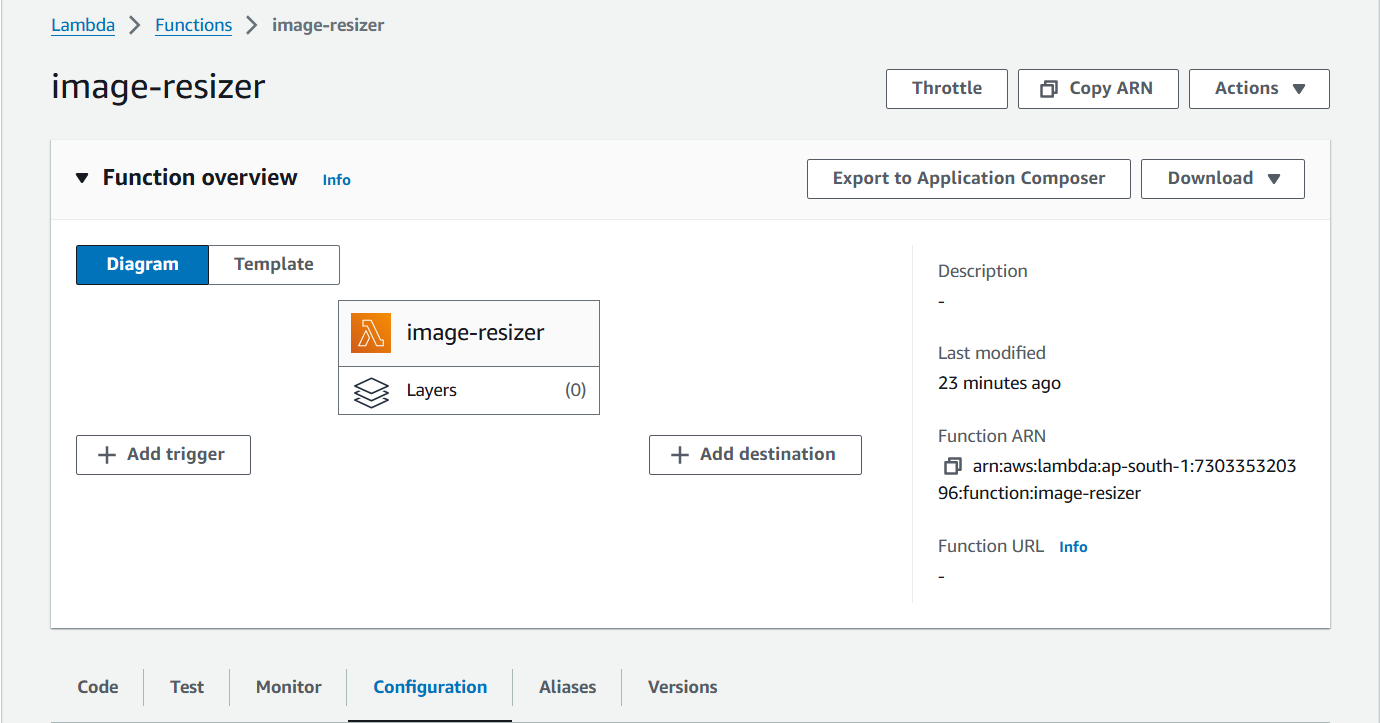
**Step 5 – Creating S3 trigger**

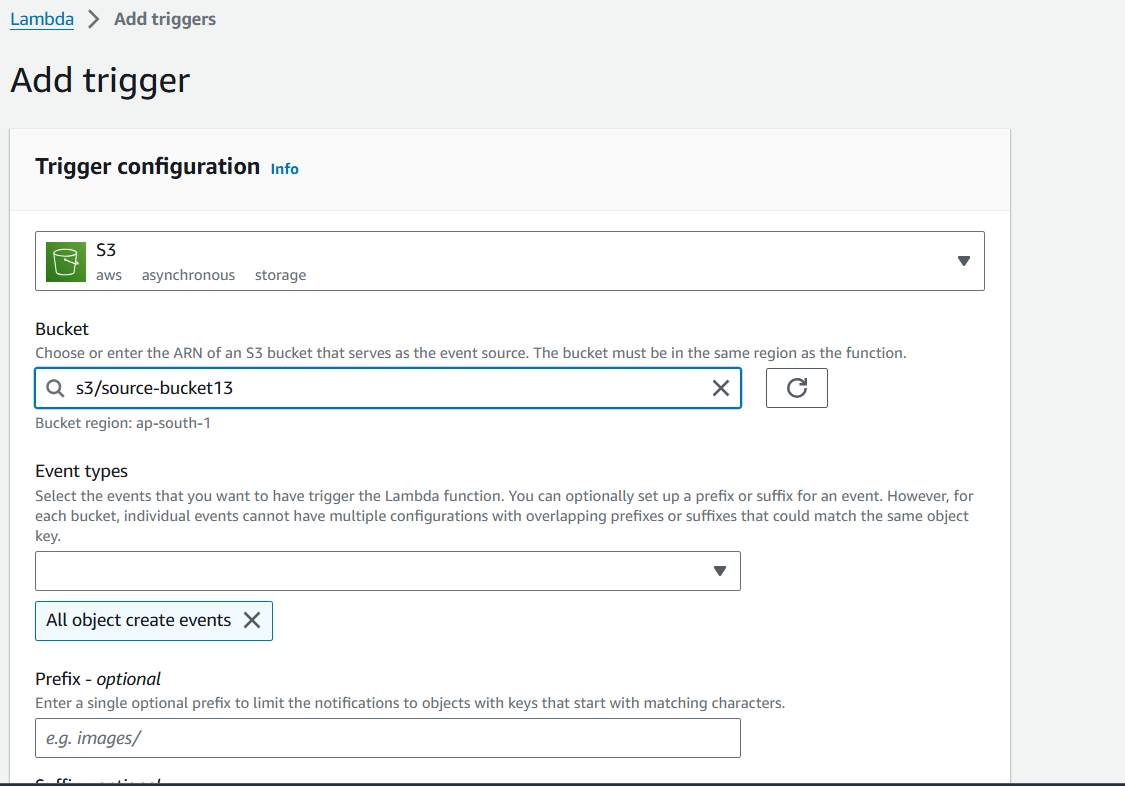
\*Add trigger

\*Select s3

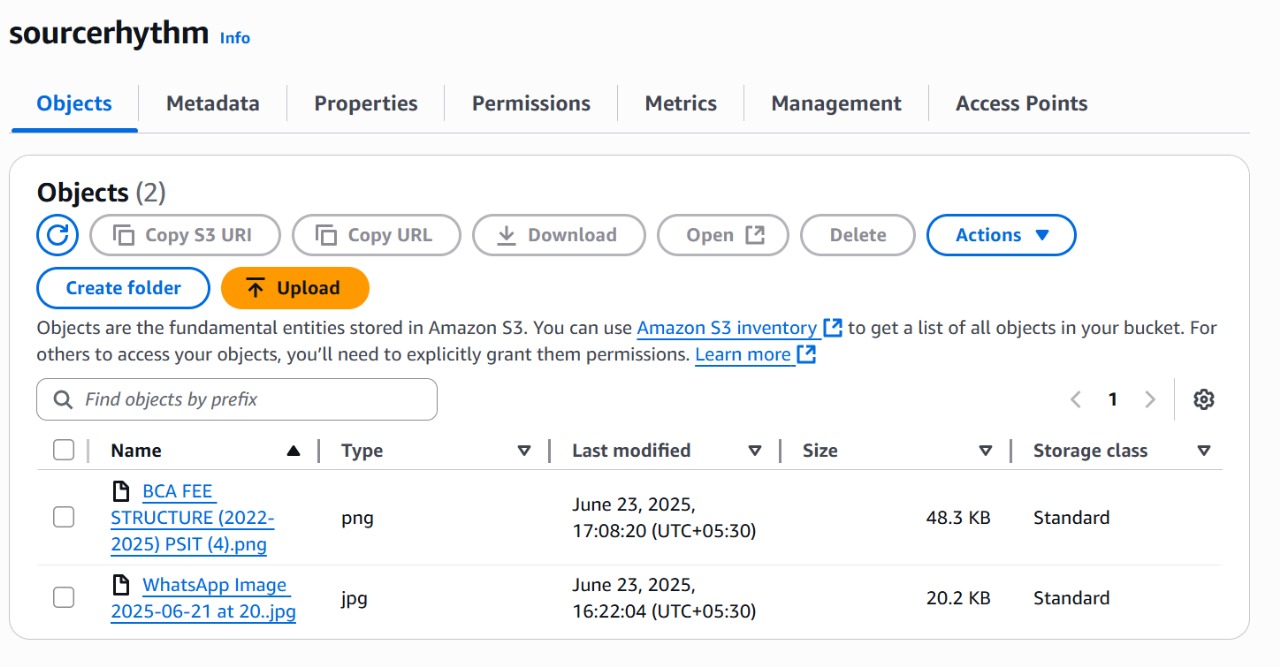
\*choose source Bucket name

\*Now Add

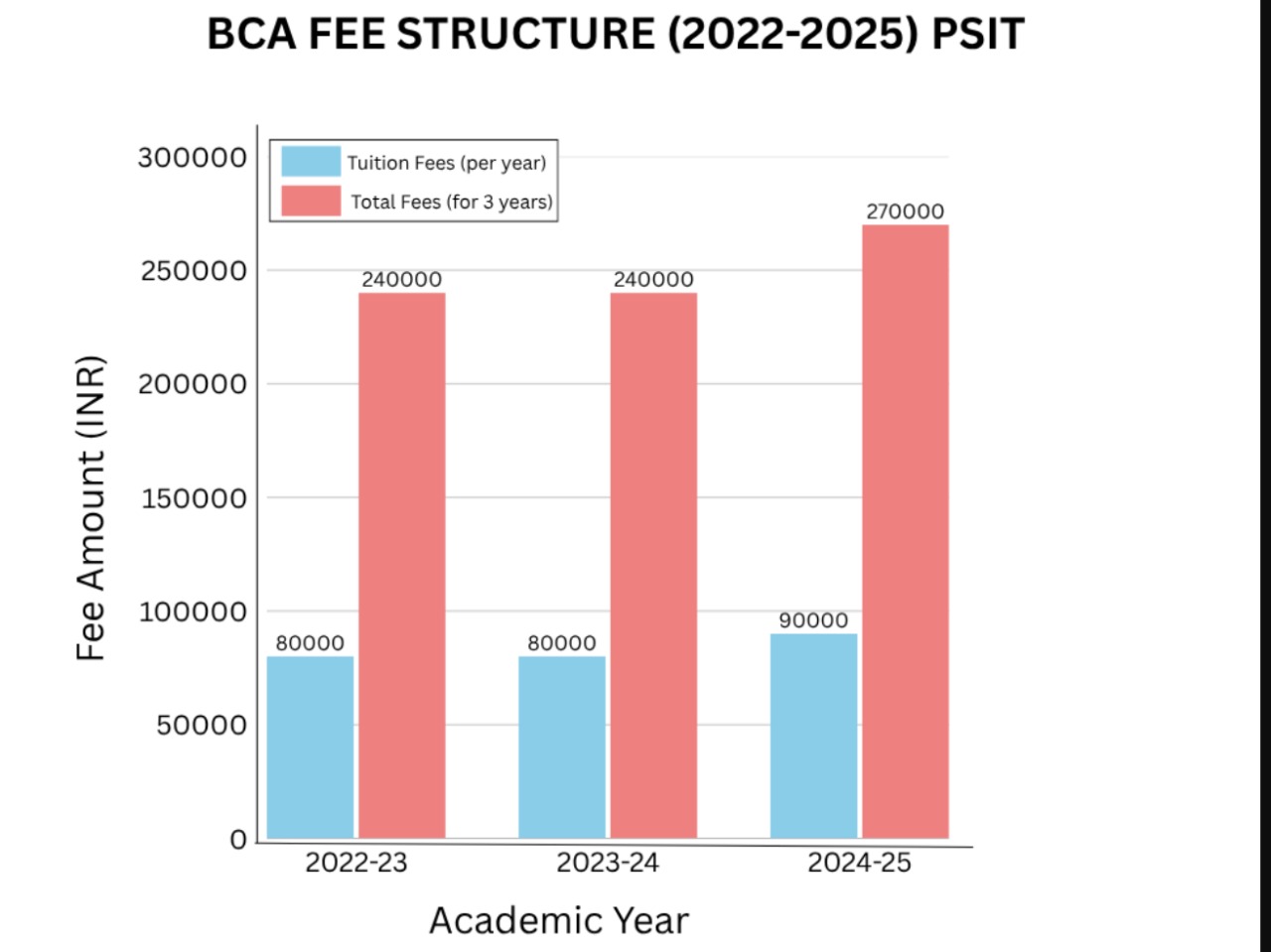




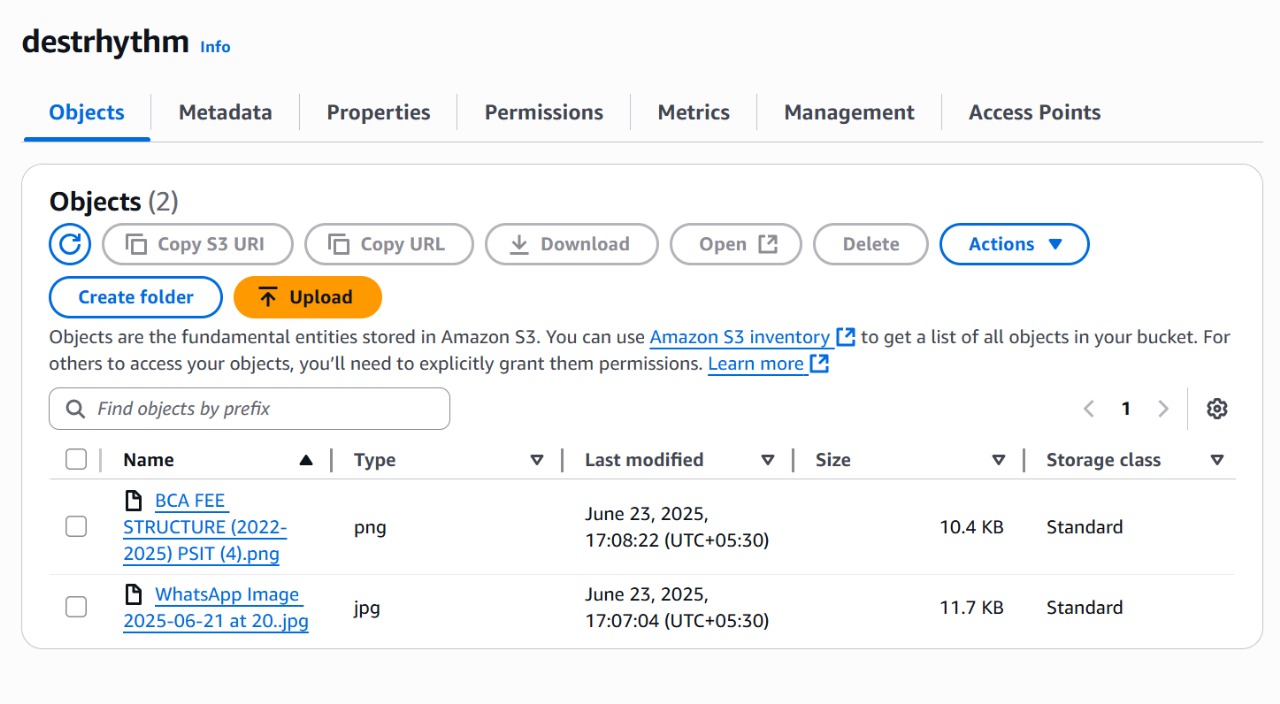
**Upload image in source Bucket**



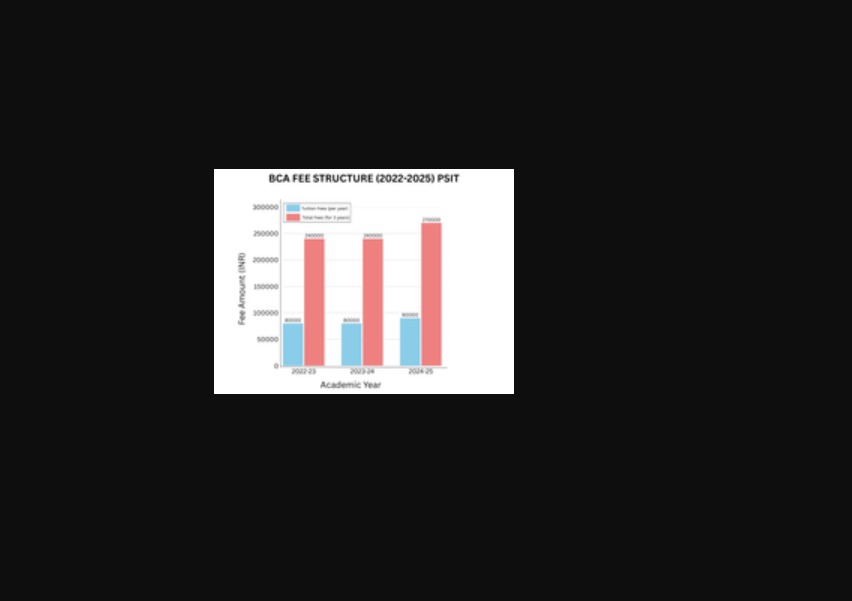
**Original image**



**Destination Bucket**



**Resize Image**



**PROJECT-2**

**TOPIC- DEPLOY A STATIC WEBSITE ON AWS**

**SUBMITTED BY:- SUBMITTED TO:-**

**Aditi Pandey Mr. Abhinav Dwivedi**

**BCA SEMESTER-4**

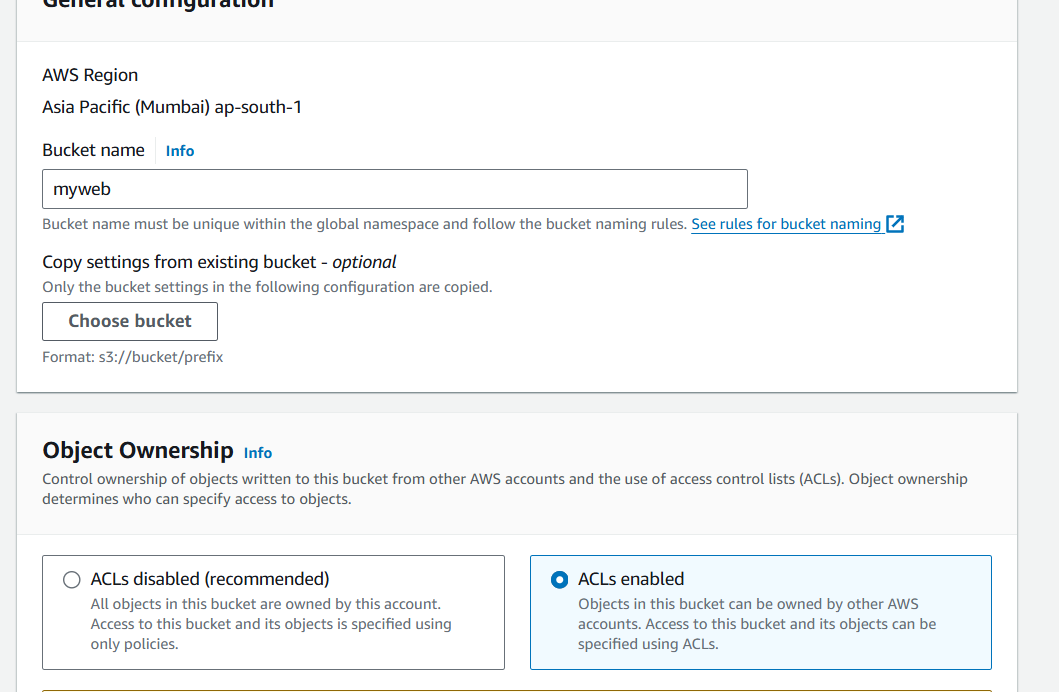
**Topics**

* [Step 1: Create a bucket](https://docs.aws.amazon.com/AmazonS3/latest/userguide/HostingWebsiteOnS3Setup.html#step1-create-bucket-config-as-website)
* [Step 2: Enable static website hosting](https://docs.aws.amazon.com/AmazonS3/latest/userguide/HostingWebsiteOnS3Setup.html#step2-create-bucket-config-as-website)
* [Step 3: Edit Block Public Access settings](https://docs.aws.amazon.com/AmazonS3/latest/userguide/HostingWebsiteOnS3Setup.html#step3-edit-block-public-access)
* [Step 4: Add a bucket policy that makes your bucket content publicly available](https://docs.aws.amazon.com/AmazonS3/latest/userguide/HostingWebsiteOnS3Setup.html#step4-add-bucket-policy-make-content-public)
* [Step 5: Configure an index document](https://docs.aws.amazon.com/AmazonS3/latest/userguide/HostingWebsiteOnS3Setup.html#step5-upload-index-doc)
* [Step 6: Configure an error document](https://docs.aws.amazon.com/AmazonS3/latest/userguide/HostingWebsiteOnS3Setup.html#step6-upload-error-doc)
* [Step 7: Test your website endpoint](https://docs.aws.amazon.com/AmazonS3/latest/userguide/HostingWebsiteOnS3Setup.html#step7-test-web-site)
* [Step 8: Clean up](https://docs.aws.amazon.com/AmazonS3/latest/userguide/HostingWebsiteOnS3Setup.html#getting-started-cleanup-s3-website-overview)

**Step 1: Create a bucket**

The following instructions provide an overview of how to create your buckets for website hosting. For detailed, step-by-step instructions on creating a bucket.

**To create a bucket**

****

1. Sign in to the AWS Management Console and open the Amazon S3 console at <https://console.aws.amazon.com/s3/>.
2. Choose **Create bucket**.
3. Enter the **Bucket name** (for example, **webbucket**).
4. Choose the Region where you want to create the bucket.

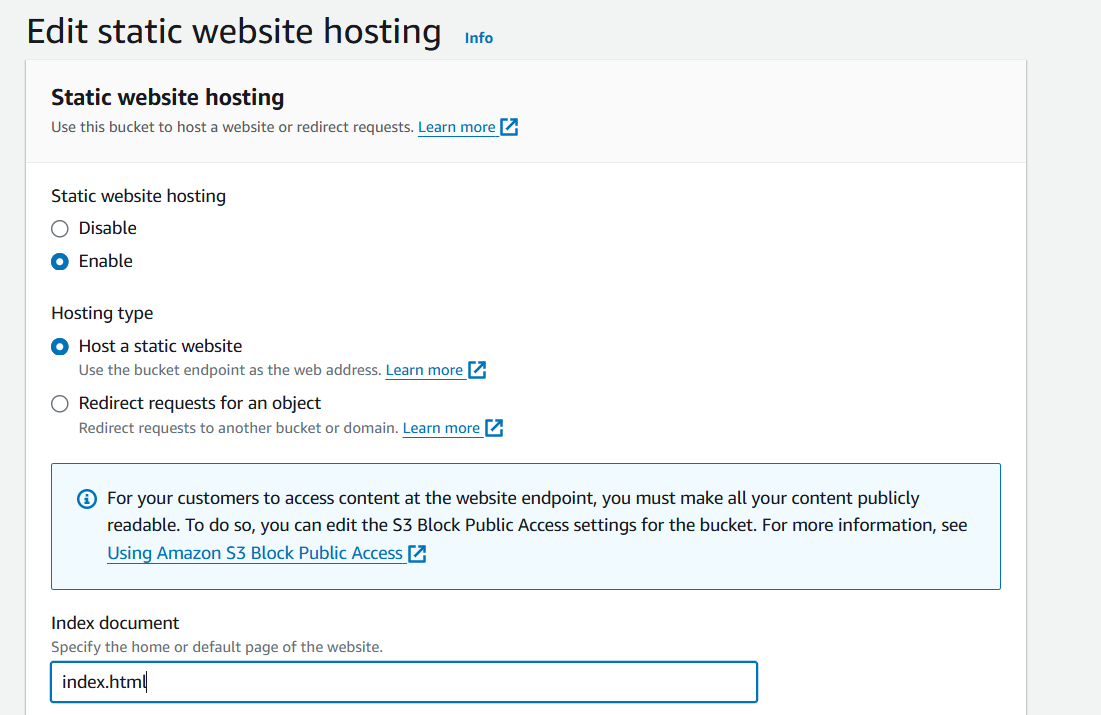
Choose a Region that is geographically close to you to minimize latency and costs, or to address regulatory requirements. The Region that you choose determines your Amazon S3 website endpoint.

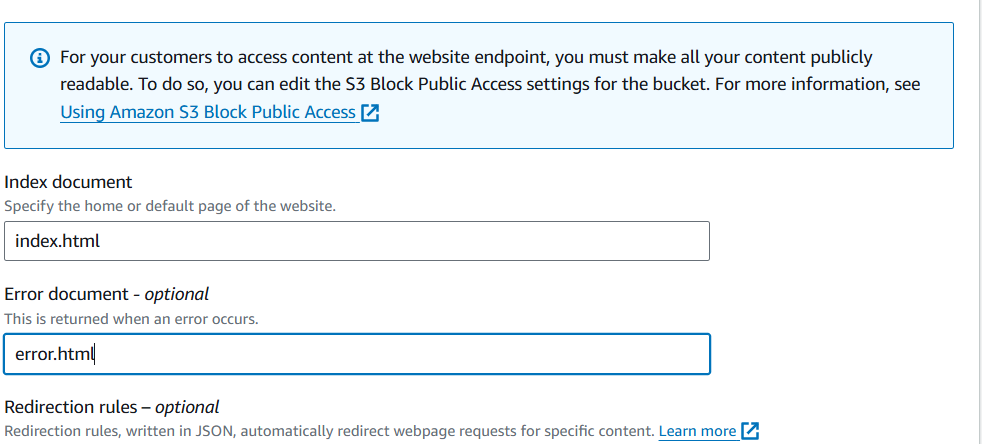
1. To accept the default settings and create the bucket, choose **Create**.

**Step 2: Enable static website hosting**

After you create a bucket, you can enable static website hosting for your bucket.

**To enable static website hosting**

****

****

1. Sign in to the AWS Management Console and open the Amazon S3 console at <https://console.aws.amazon.com/s3/>.
2. In the **Buckets** list, choose the name of the bucket that you want to enable static website hosting for.
3. Choose **Properties**.
4. Under **Static website hosting**, choose **Edit**.
5. Choose **Use this bucket to host a website**.
6. Under **Static website hosting**, choose **Enable**.
7. In **Index document**, enter the file name of the index document, typically index.html.

The index document name is case sensitive and must exactly match the file name of the HTML index document that you plan to upload to your S3 bucket. When you configure a bucket for website hosting, you must specify an index document. Amazon S3 returns this index document when requests are made to the root domain or any of the subfolders.

1. (Optional) If you want to specify advanced redirection rules, in **Redirection rules**, enter JSON to describe the rules.

For example, you can conditionally route requests according to specific object key names or prefixes in the request.

1. Choose **Save changes**.

Amazon S3 enables static website hosting for your bucket. At the bottom of the page, under **Static website hosting**, you see the website endpoint for your bucket.

1. Under **Static website hosting**, note the **Endpoint**.

The **Endpoint** is the Amazon S3 website endpoint for your bucket. After you finish configuring your bucket as a static website, you can use this endpoint to test your website.

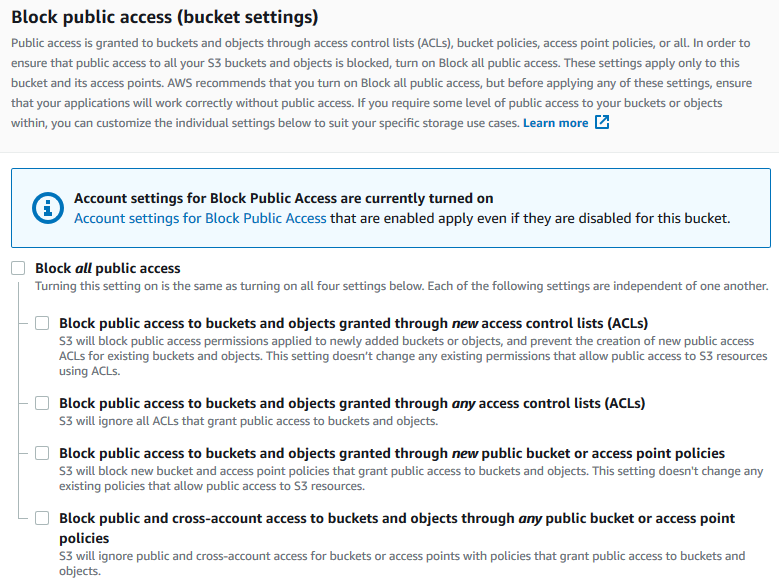
**Step 3: Edit Block Public Access settings**

By default, Amazon S3 blocks public access to your account and buckets. If you want to use a bucket to host a static website, you can use these steps to edit your block public access settings.

1. Open the Amazon S3 console at <https://console.aws.amazon.com/s3/>.
2. Choose the name of the bucket that you have configured as a static website.
3. Choose **Permissions**.
4. Under **Block public access (bucket settings)**, choose **Edit**.
5. Clear **Block *all* public access**, and choose **Save changes**.

**Warning**

Before you complete this step, review [Blocking public access to your Amazon S3 storage](https://docs.aws.amazon.com/AmazonS3/latest/userguide/access-control-block-public-access.html) to ensure you understand and accept the risks involved with allowing public access. When you turn off block public access settings to make your bucket public, anyone on the internet can access your bucket. We recommend that you block all public access to your buckets.



Amazon S3 turns off Block Public Access settings for your bucket. To create a public, static website, you might also have to [edit the Block Public Access settings](https://docs.aws.amazon.com/AmazonS3/latest/user-guide/block-public-access-account.html) for your account before adding a bucket policy.

**Step 4: Add a bucket policy that makes your bucket content publicly available**

After you edit S3 Block Public Access settings, you can add a bucket policy to grant public read access to your bucket. When you grant public read access, anyone on the internet can access your bucket.

**Important**

The following policy is an example only and allows full access to the contents of your bucket.

1. Under **Buckets**, choose the name of your bucket.
2. Choose **Permissions**.
3. Under **Bucket Policy**, choose **Edit**.
4. To grant public read access for your website, copy the following bucket policy, and paste it in the **Bucket policy editor**.
5. {
6. "Version": "2012-10-17",
7. "Statement": [
8. {
9. "Sid": "PublicReadGetObject",
10. "Effect": "Allow",
11. "Principal": "\*",
12. "Action": [
13. "s3:GetObject"
14. ],
15. "Resource": [
16. "arn:aws:s3:::*Bucket-Name*/\*"
17. ]
18. }
19. ]

}

1. Update the Resource to your bucket name.

In the preceding example bucket policy, *Bucket-Name* is a placeholder for the bucket name. To use this bucket policy with your own bucket, you must update this name to match your bucket name.

1. Choose **Save changes**.

A message appears indicating that the bucket policy has been successfully added.

If you see an error that says Policy has invalid resource, confirm that the bucket name in the bucket policy matches your bucket name.

If you get an error message and cannot save the bucket policy, check your account and bucket Block Public Access settings to confirm that you allow public access to the bucket.

**Step 5: Configure an index document**

When you enable static website hosting for your bucket, you enter the name of the index document (for example, **index.html**). After you enable static website hosting for the bucket, you upload an HTML file with this index document name to your bucket.

**To configure the index document**

1. Create an index.html file.

If you don't have an index.html file, you can use the following HTML to create one:

<html xmlns="http://www.w3.org/1999/xhtml" >

<head>

<title>My Website Home Page</title>

</head>

<body>

<h1>Welcome to my website</h1>

<p>Now hosted on Amazon S3!</p>

</body>

</html>

1. Save the index file locally.

The index document file name must exactly match the index document name that you enter in the **Static website hosting** dialog box. The index document name is case sensitive. For example, if you enter index.html for the **Index document** name in the **Static website hosting** dialog box, your index document file name must also be index.html and not Index.html.

1. Sign in to the AWS Management Console and open the Amazon S3 console at <https://console.aws.amazon.com/s3/>.
2. In the **Buckets** list, choose the name of the bucket that you want to use to host a static website.
3. Enable static website hosting for your bucket, and enter the exact name of your index document (for example, index.html).

After enabling static website hosting, proceed to step 6.

1. To upload the index document to your bucket, do one of the following:
   * Drag and drop the index file into the console bucket listing.
   * Choose **Upload**, and follow the prompts to choose and upload the index file.
2. (Optional) Upload other website content to your bucket.

**Step 6: Configure an error document**

When you enable static website hosting for your bucket, you enter the name of the error document (for example, **404.html**). After you enable static website hosting for the bucket, you upload an HTML file with this error document name to your bucket.

**To configure an error document**

1. Create an error document, for example 404.html.
2. Save the error document file locally.

The error document name is case sensitive and must exactly match the name that you enter when you enable static website hosting. For example, if you enter 404.html for the **Error document** name in the **Static website hosting** dialog box, your error document file name must also be 404.html.

1. Sign in to the AWS Management Console and open the Amazon S3 console at <https://console.aws.amazon.com/s3/>.
2. In the **Buckets** list, choose the name of the bucket that you want to use to host a static website.
3. Enable static website hosting for your bucket, and enter the exact name of your error document (for example, 404.html).

After enabling static website hosting, proceed to step 6.

1. To upload the error document to your bucket, do one of the following:
   * Drag and drop the error document file into the console bucket listing.
   * Choose **Upload**, and follow the prompts to choose and upload the index file.

**Step 7: Test your website endpoint**

After you configure static website hosting for your bucket, you can test your website endpoint.

1. Under **Buckets**, choose the name of your bucket.
2. Choose **Properties**.
3. At the bottom of the page, under **Static website hosting**, choose your **Bucket website endpoint**.

Your index document opens in a separate browser window.

Now you have hosted a website on Amazon S3. This website is available at the Amazon S3 website endpoint. However, you might have a domain, such as example.com, that you want to use to serve the content from the website you created.

**Step 8: Clean up**

If you created your static website only as a learning exercise, delete the AWS resources that you allocated so that you no longer accrue charges. After you delete your AWS resources, your website is no longer available.

Project 3

**TOPIC- Integrate Grafana with Linux Server for high CPU utilization and create a graph in Grafana**

**SUBMITTED BY:- SUBMITTED TO:-**

**Aditi Pandey Mr. Abhinav Dwivedi**

**BCA SEMESTER-4**

Project:-3

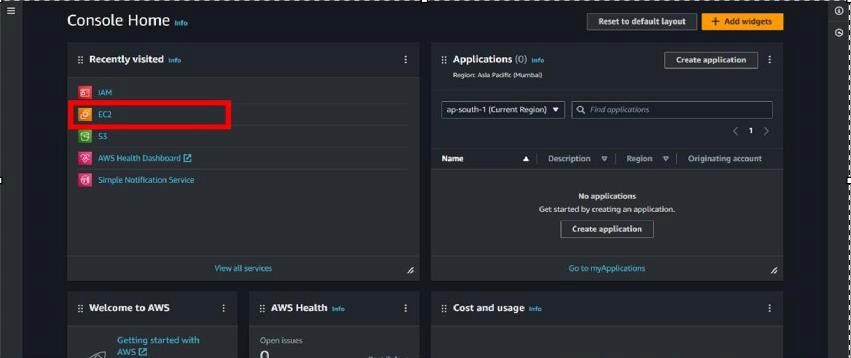
**Integrate Grafana with Linux Server for high cpu utilization and create a graph in Grafana.**

Sign in to AWS Management Console

1. Click on the Open Console button, and you will get redirected to AWS Console in a new browser tab.

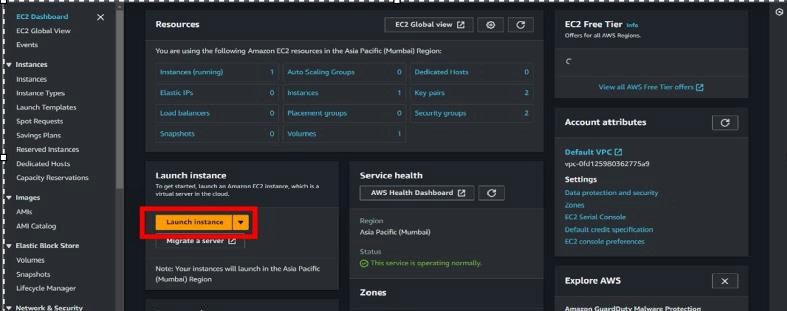
* On the AWS sign-in page, Leave the Account ID as default. Never edit/remove the 12-digit Account ID present in the AWS Console. otherwise, you cannot proceed with the lab.
* Now copy your User Name and Password in the Lab Console to the IAM Username and Password in AWS Console and click on the Sign in button.

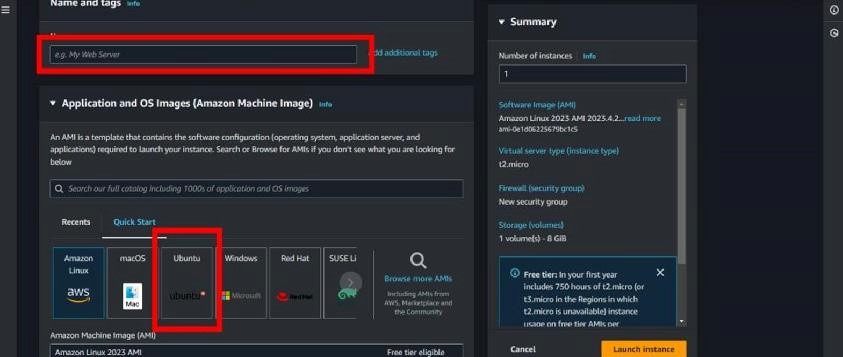
2. Once Signed in to the AWS Management Console, Make the default AWS Region as US East (N. Virginia) us-east-1.



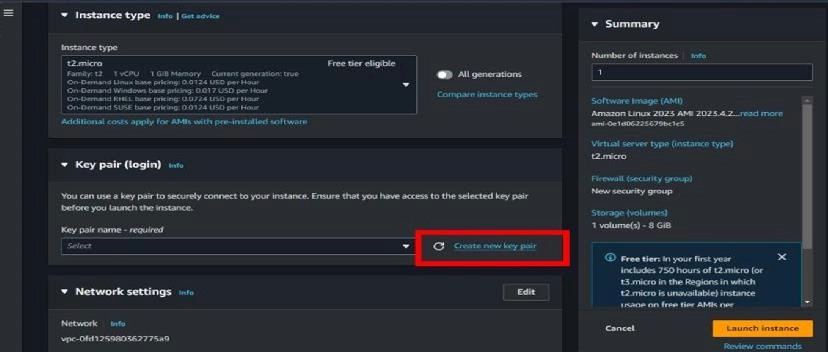
Create an EC2 Instance(ubuntu):

* For creating an EC2 instance follow the following steps as shown in snapshots.

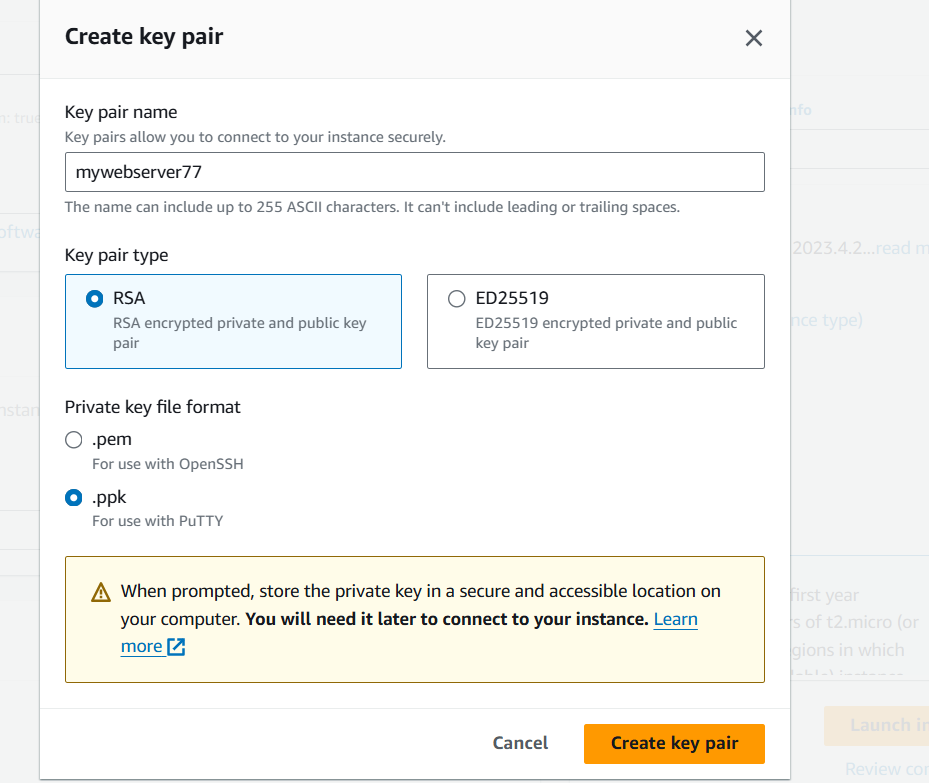




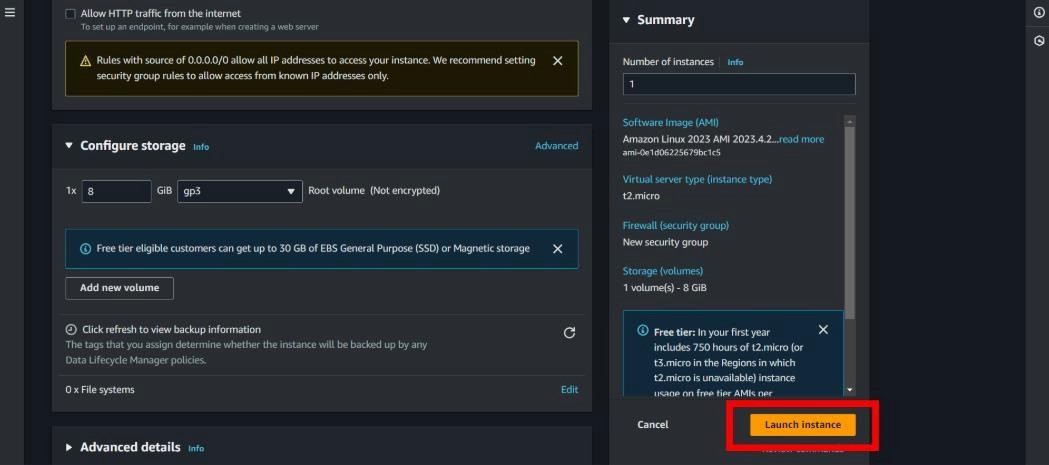
* Provide the EC2 name of your choise and select”**Ubuntu”**as an OS Image.



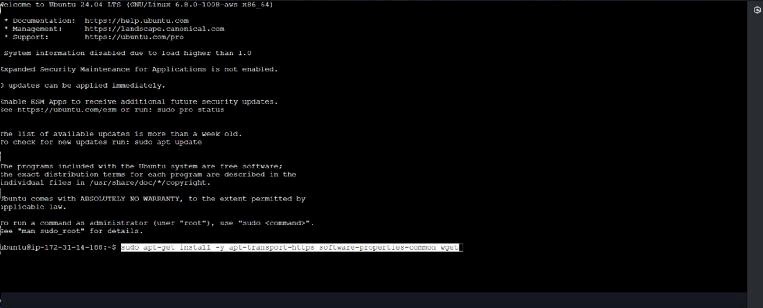
* Create a new key pair.



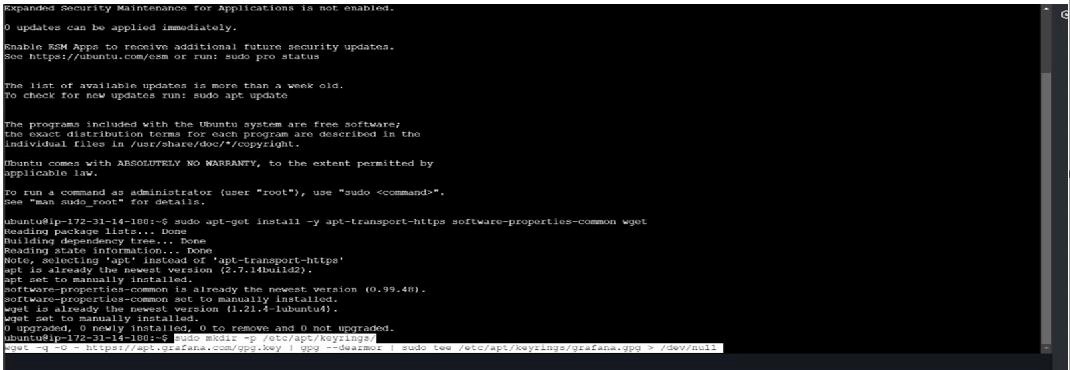
* Scroll down and click on “**LAUNCH INSTANCE** “.



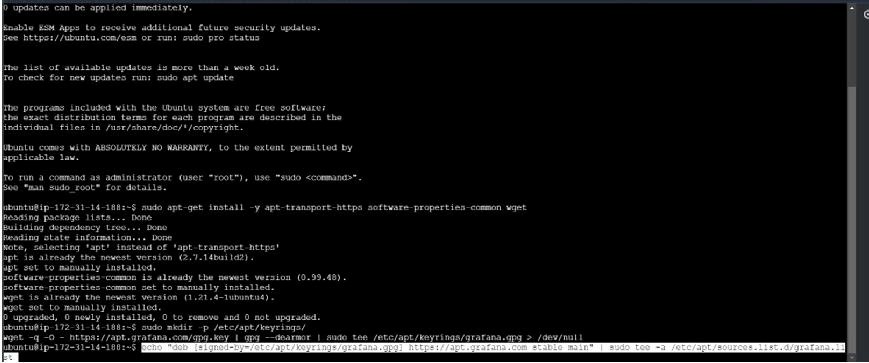
* Then open your instance and connect that instance by putty or on web browser.
* After connecting the instance follow the given command or read Grafana documentation for help.



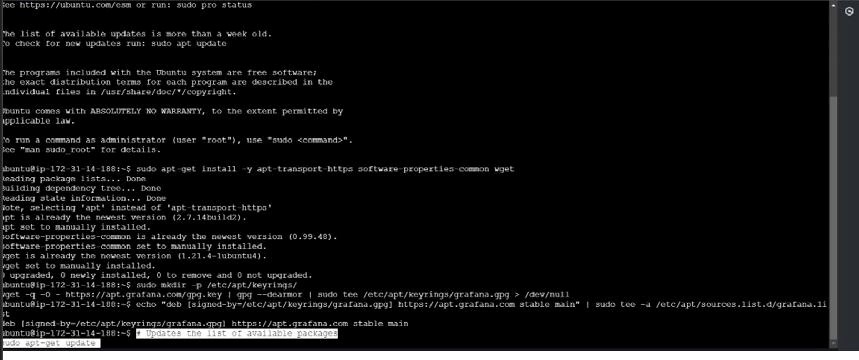
*sudo apt-get install -y apt-transport-https software-properties-common wget*



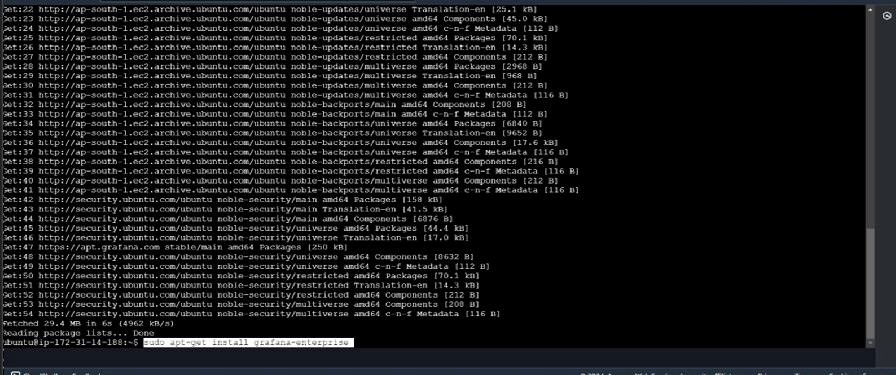
*sudomkdir -p /etc/apt/keyrings/wget -q -O - https://apt.grafana.com/gpg.key | gpg --dearmor | sudo tee /etc/apt/keyrings/grafana.gpg> /dev/null*



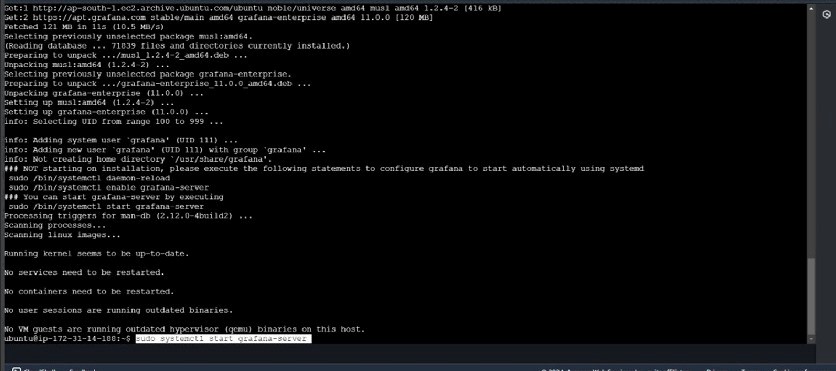
*echo "deb [signed-by=/etc/apt/keyrings/grafana.gpg] https://apt.grafana.com stable main" | sudo tee -a /etc/apt/sources.list.d/grafana.list*



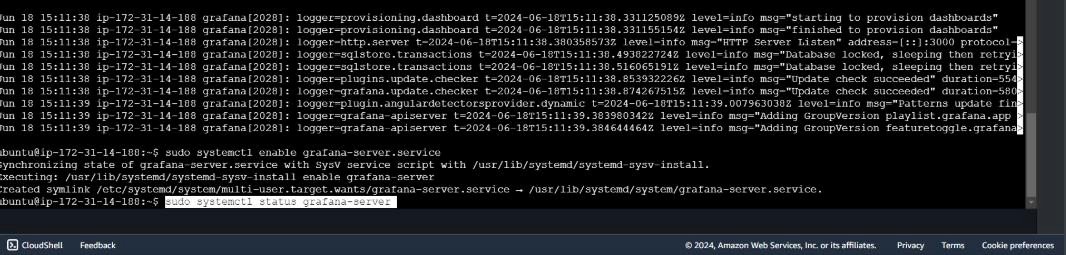
# To updates the list of available packages *sudo apt-get update*



# To installs the latest Enterprise release: *sudo apt-get install grafana-enterprise*

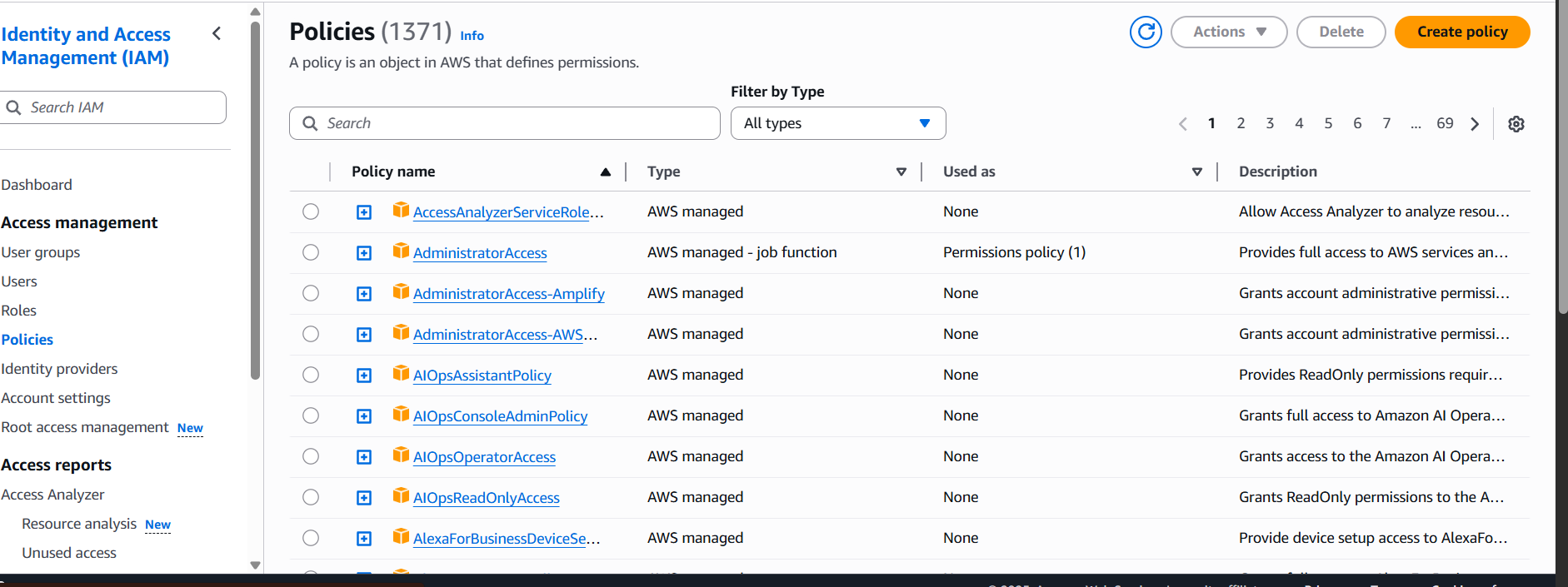


* *sudosystemctl start grafana-server*
* *sudosystemctl enable grafana-server.service*

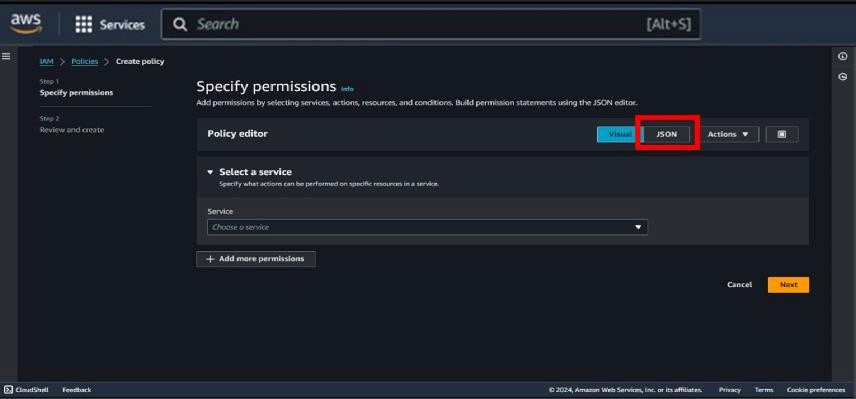


*sudosystemctl status grafana-server.service*

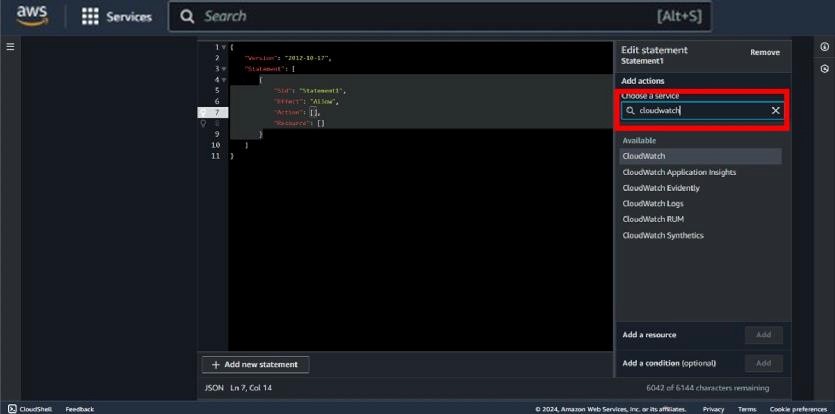




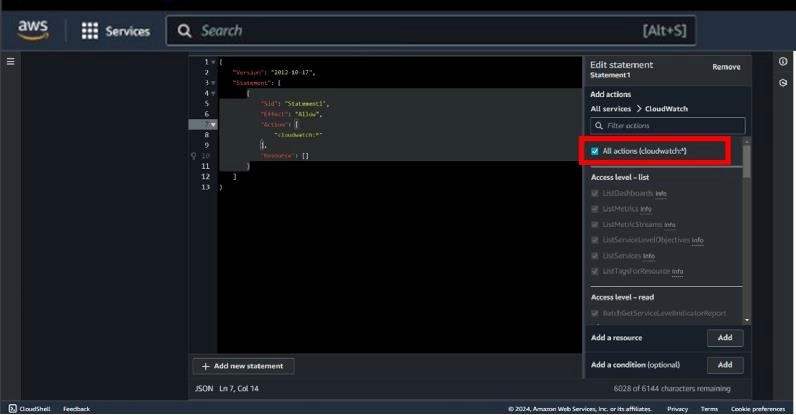
* Then search IAM role and go to policies and click on **Create policies**.



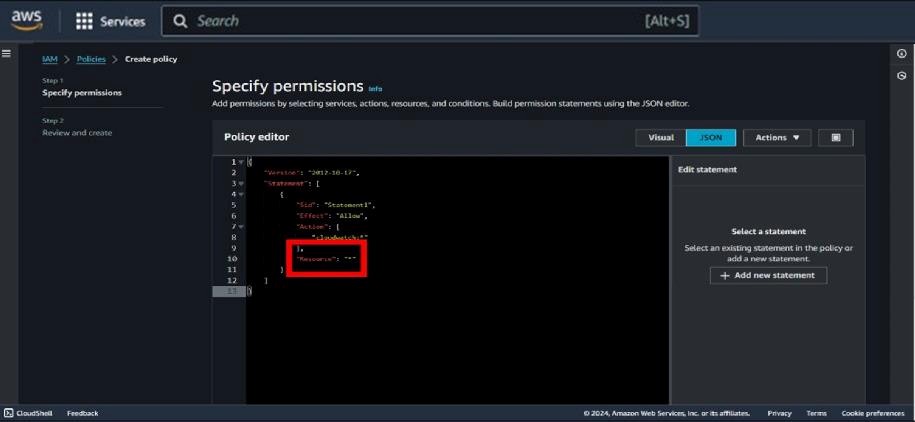
* Then go to **JSON** type.



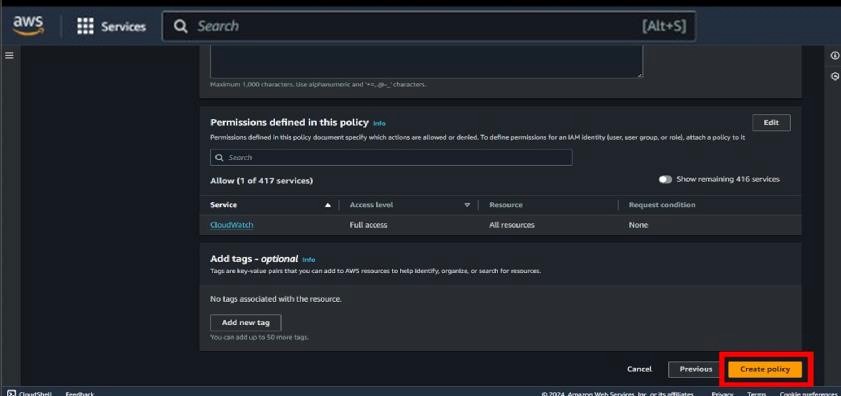
* Click on add action and search for **“Cloudwatch”**.



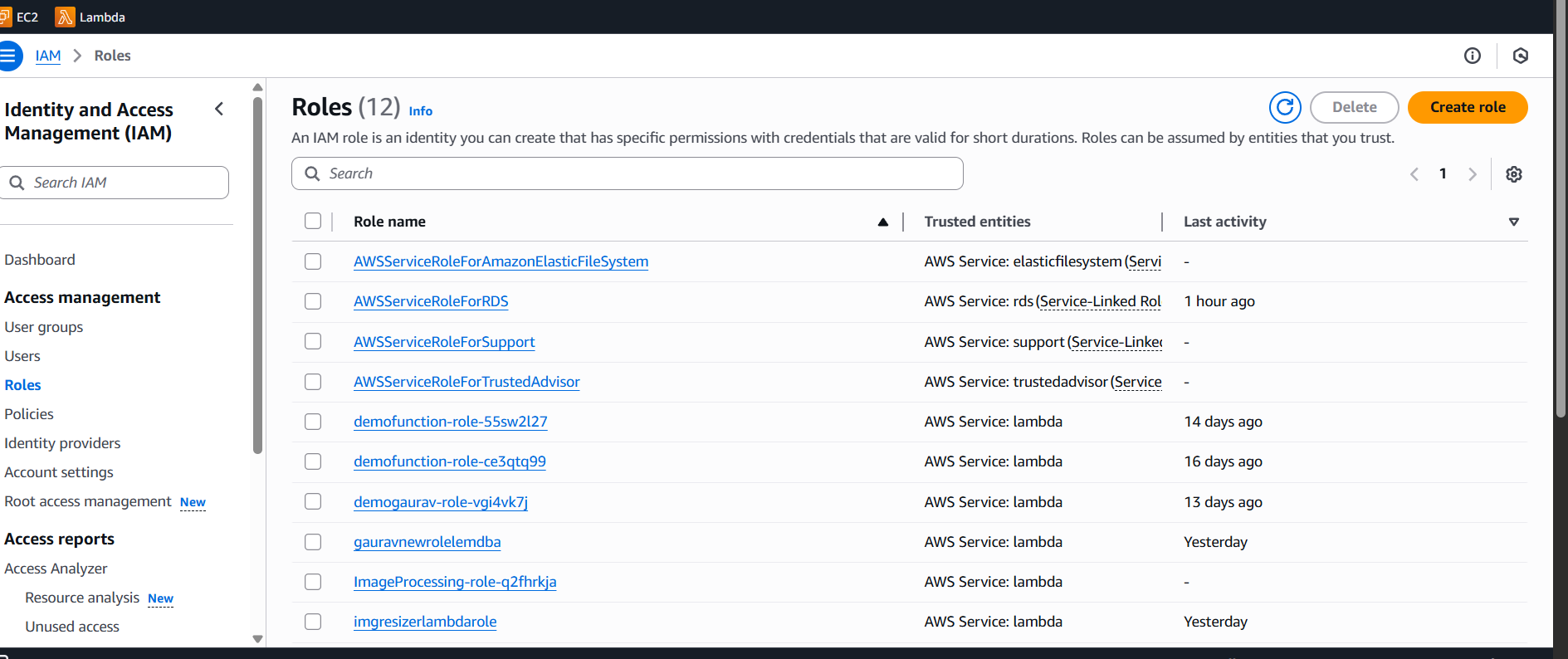
* After selecting cloudwatch select **“All actions “.**



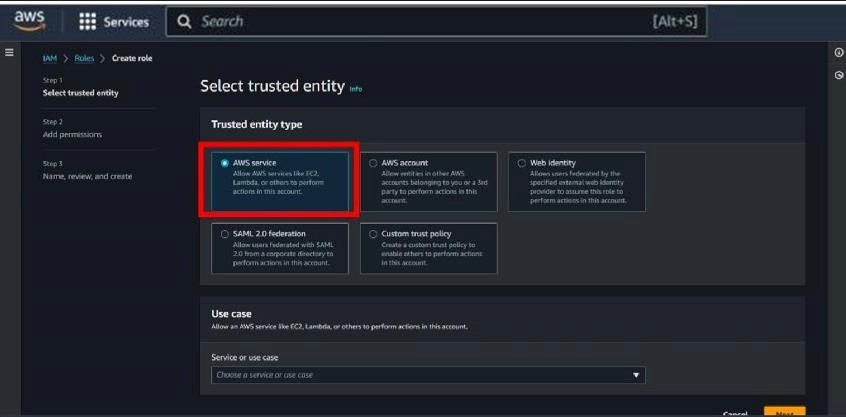
* Then put **“Resource “ : ”\*”**



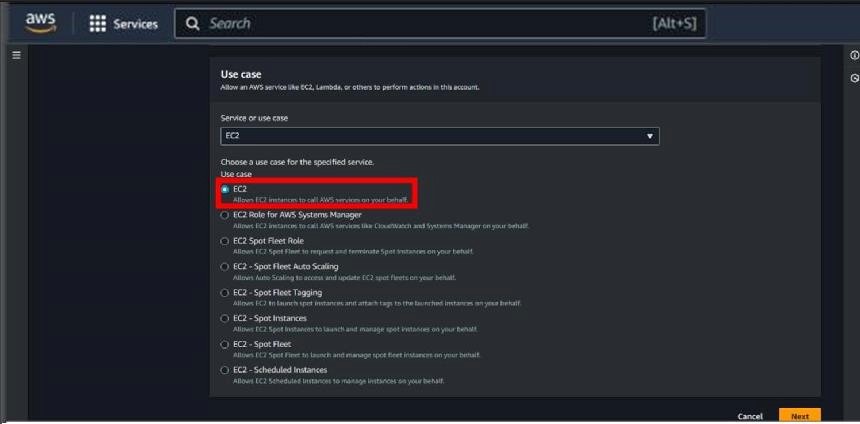
* Scroll down and click on **Create policy.**



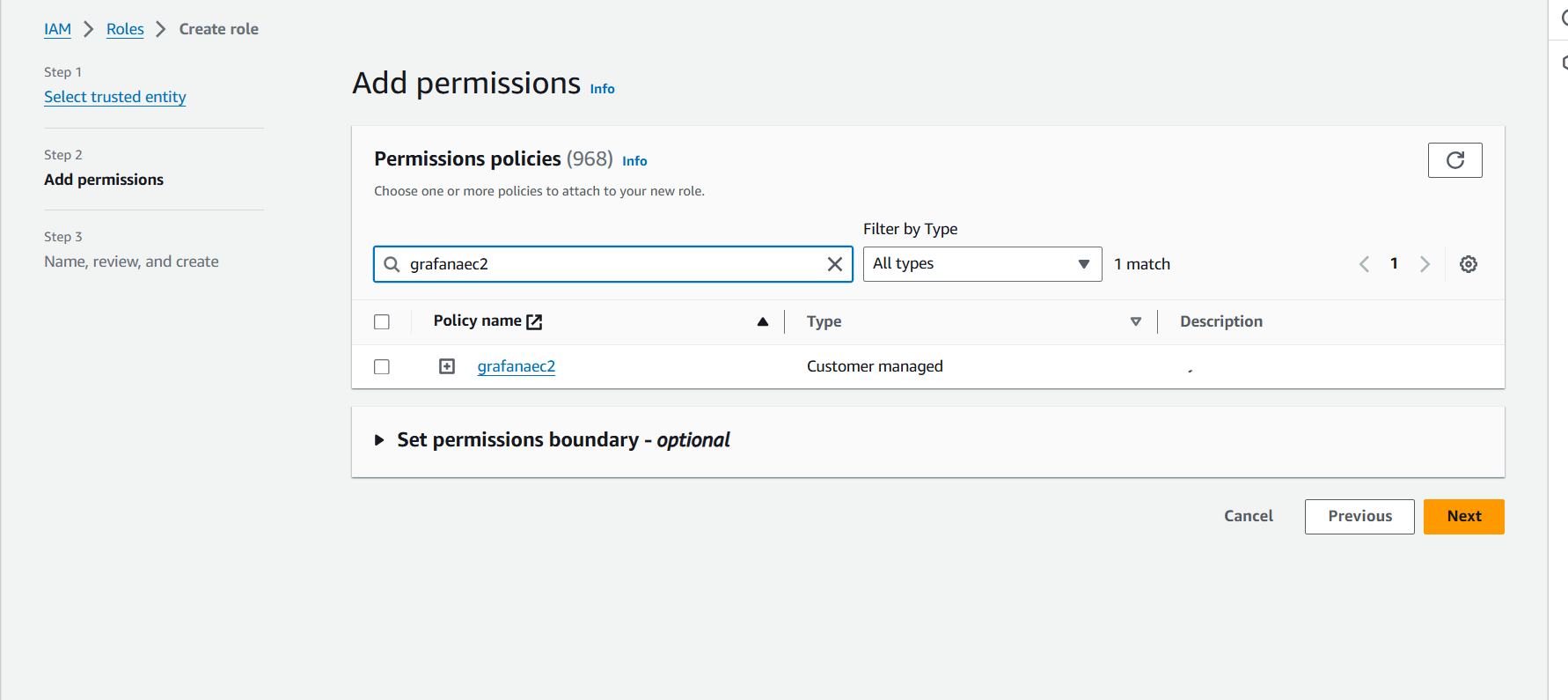
* Then go to roles and click on **Create roles.**



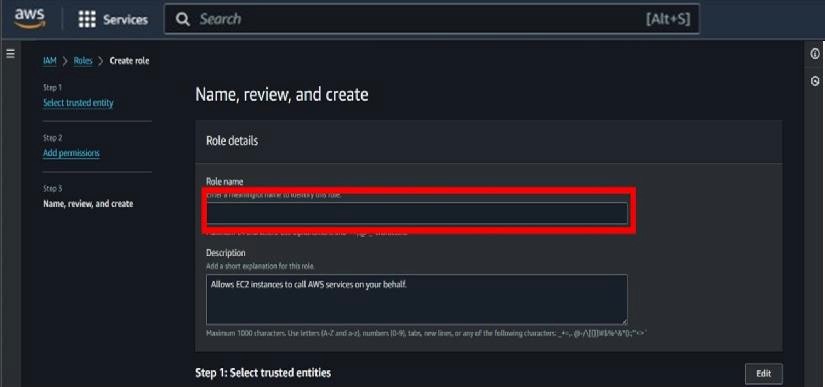
* Then select entity type is **AWS services .**



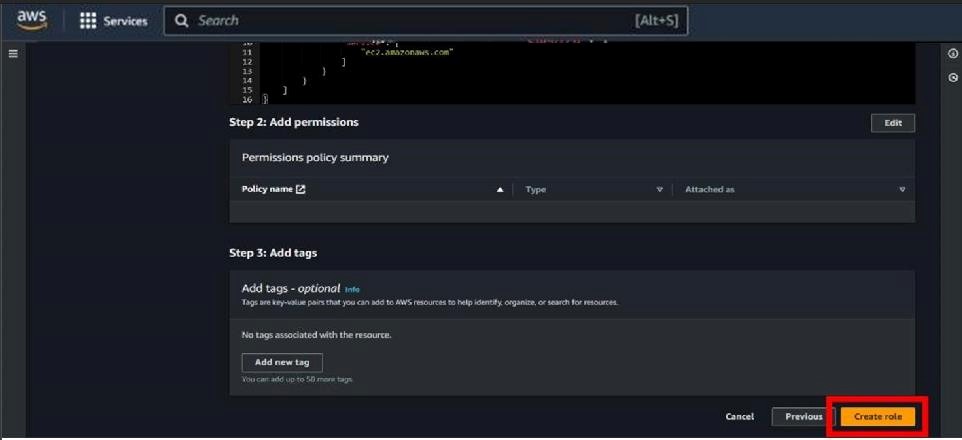
* Now , select use case as **EC2 .**



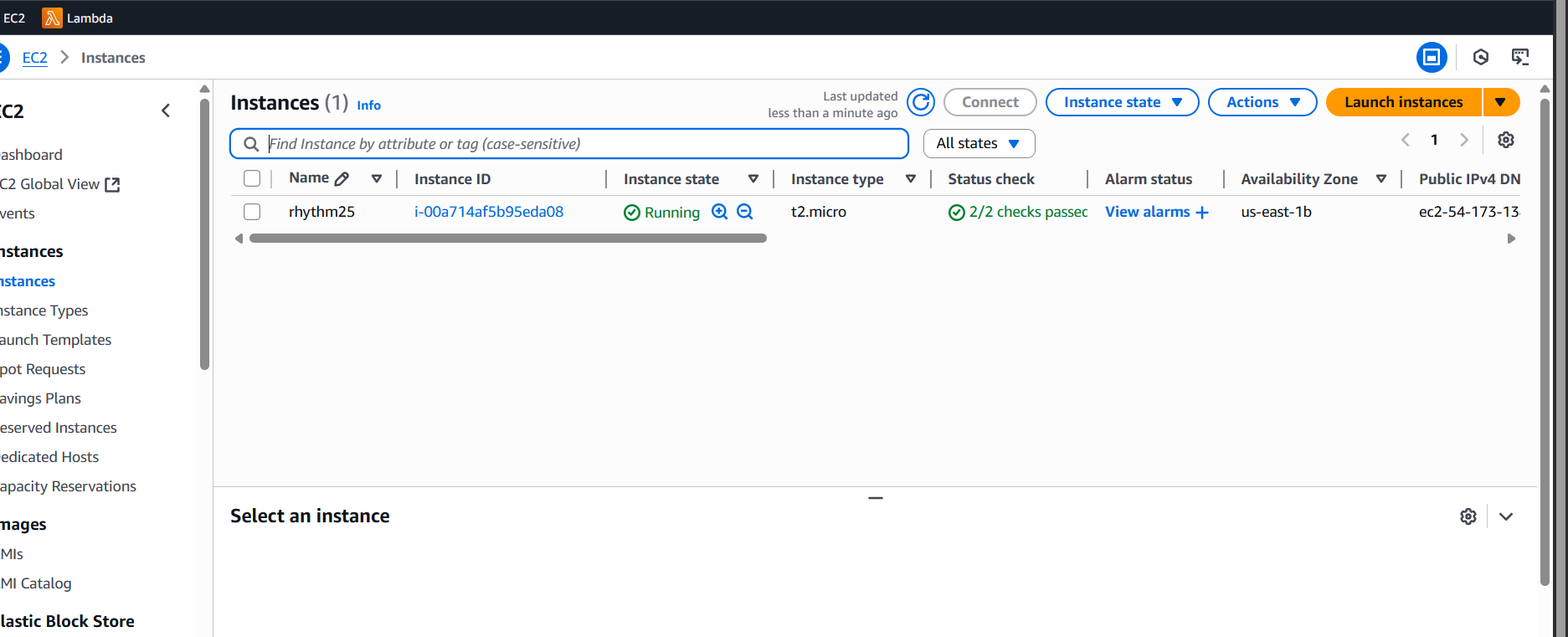
* Select your policy here which you have created previously , then click on **Next**.



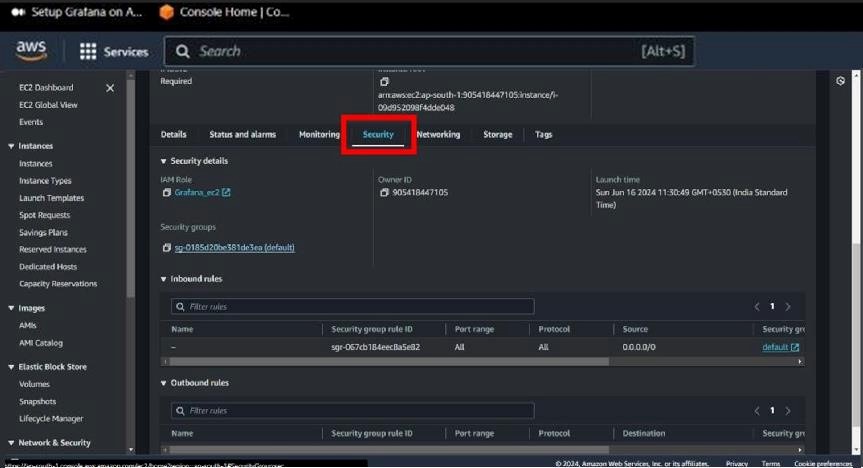
* Then give a name of your choise to the role .



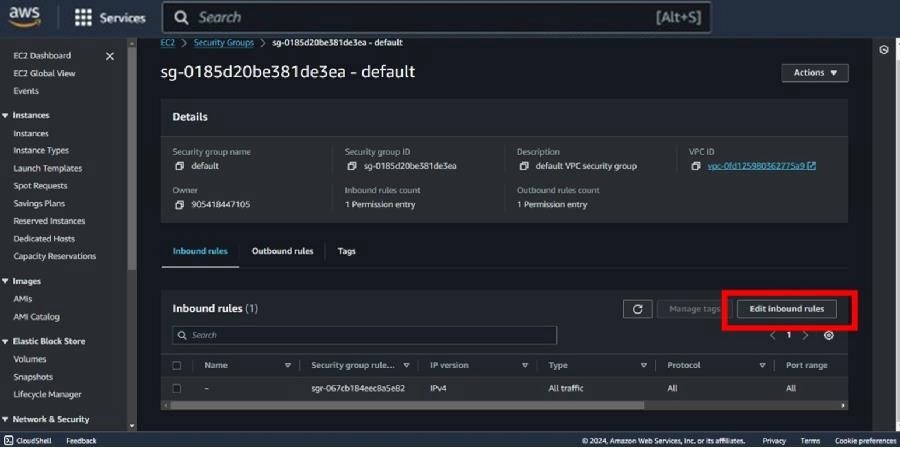
* Then scroll down and click on the **Create role** .

+

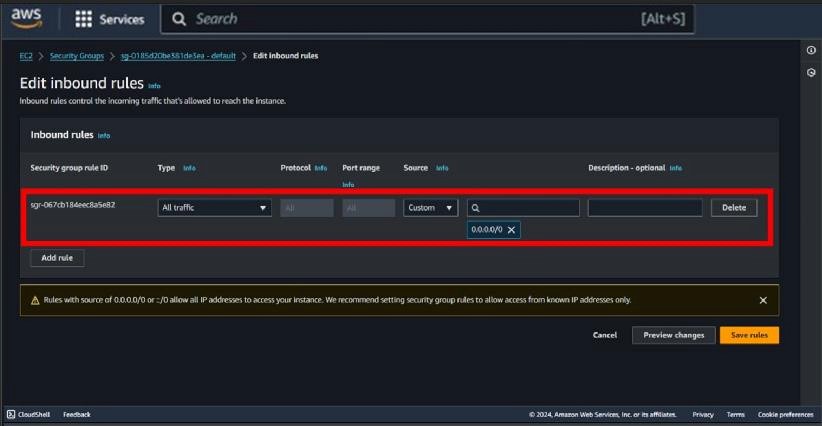
* After creating your role and policy go to instance and open your instance , which you have already created.



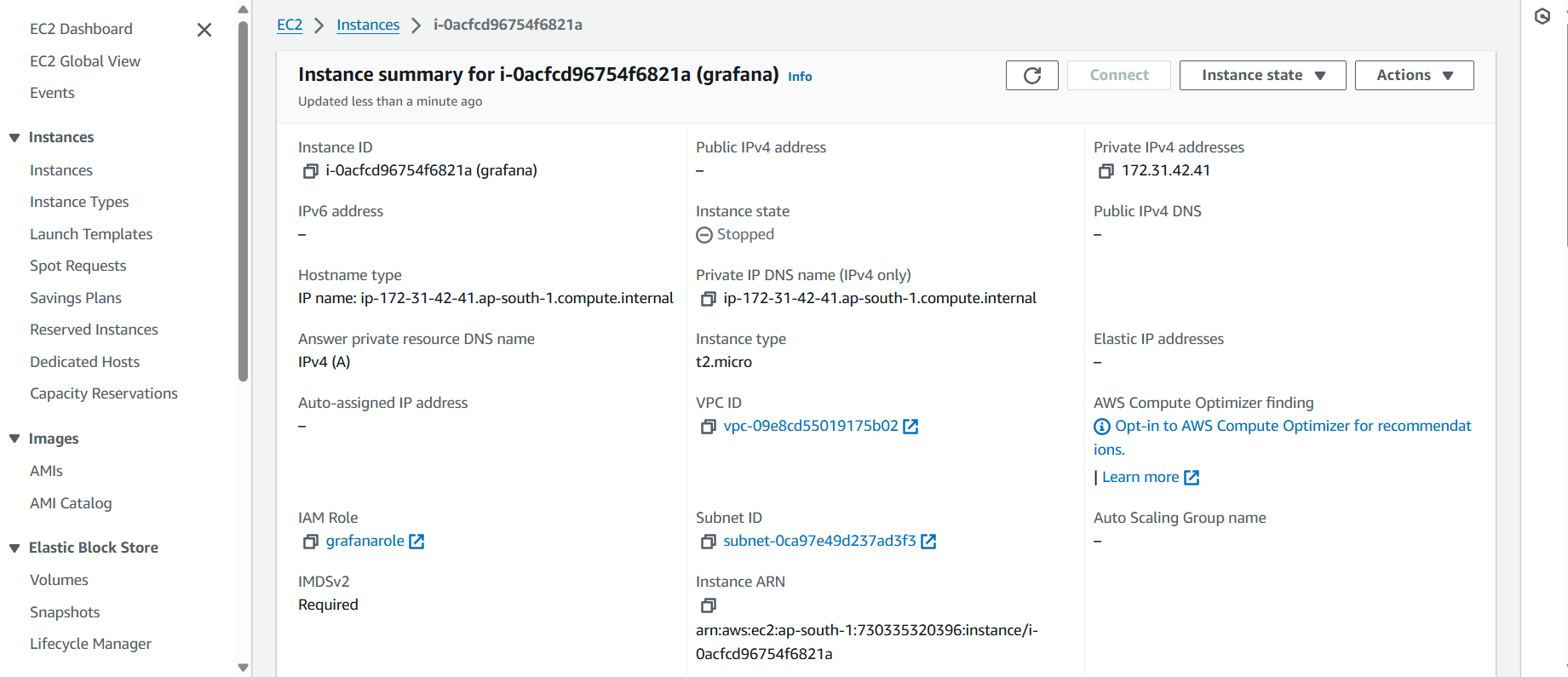
* Scroll down and go to **Security** option.



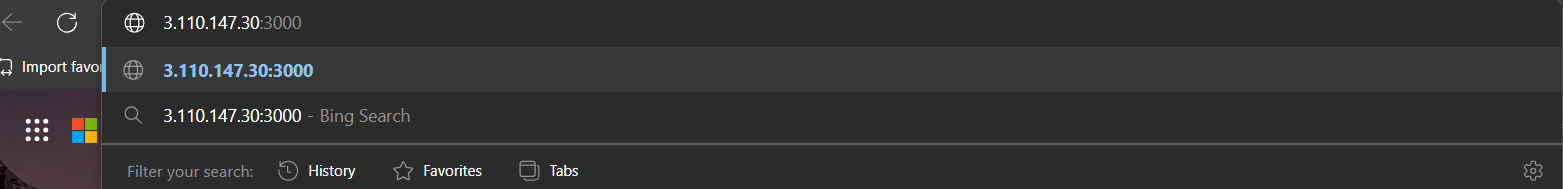
* Then click on **Edit inbound rule** .



* Then modify your rule **select type = All traffic** and **source =0.0.0.0/0**

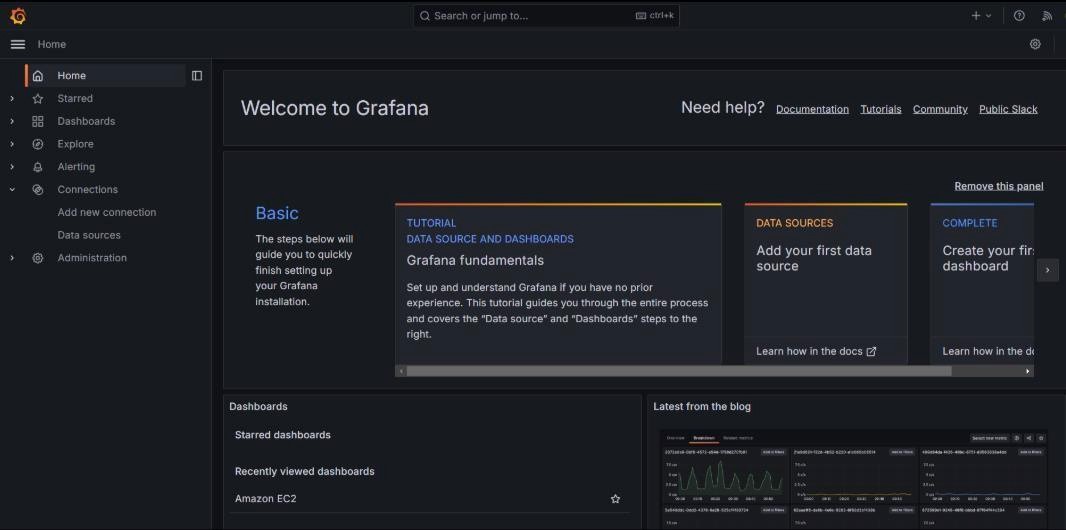


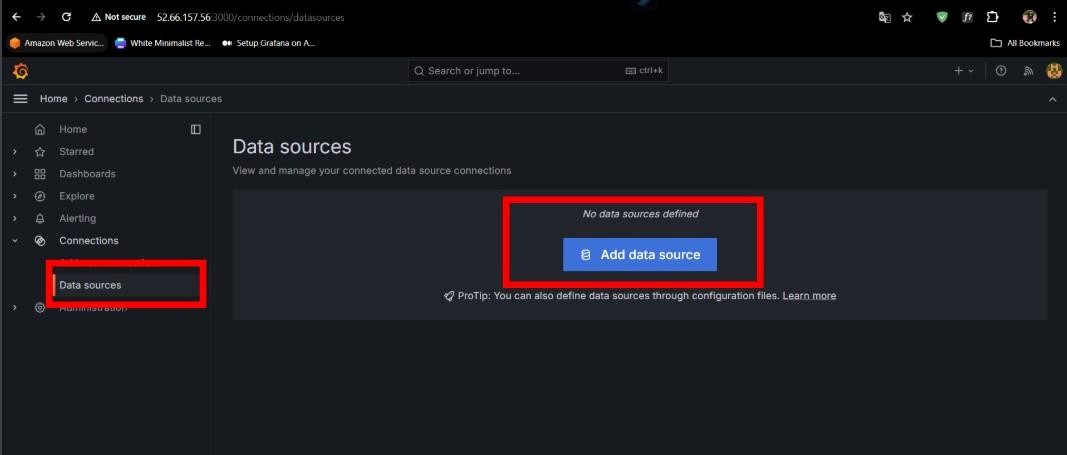
* Then copy your **public IPv4 address** of your instance .

Then paste your copied ip address and type “**:3000”** after ip address and search it .

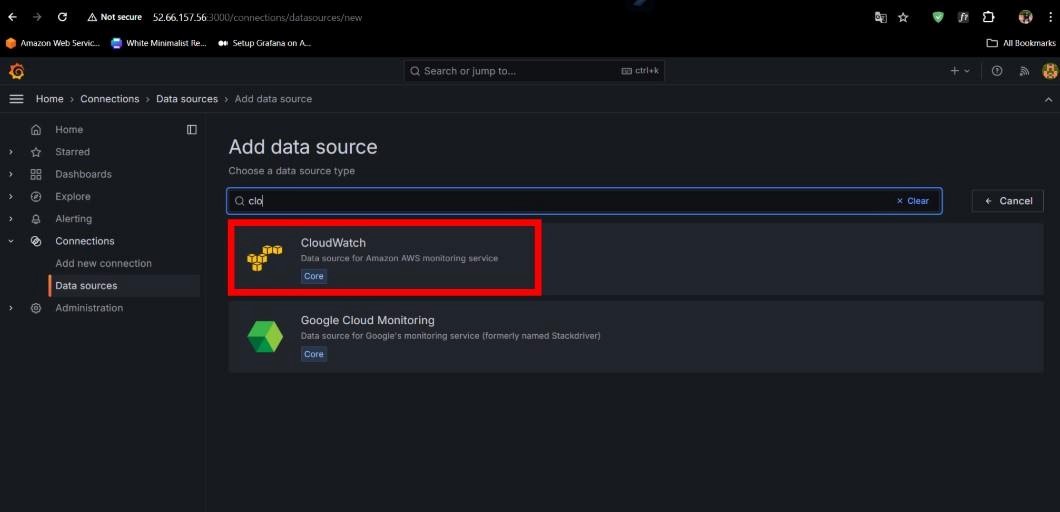


* Then login in Grafana by the help of credientials shown in above snapshot.

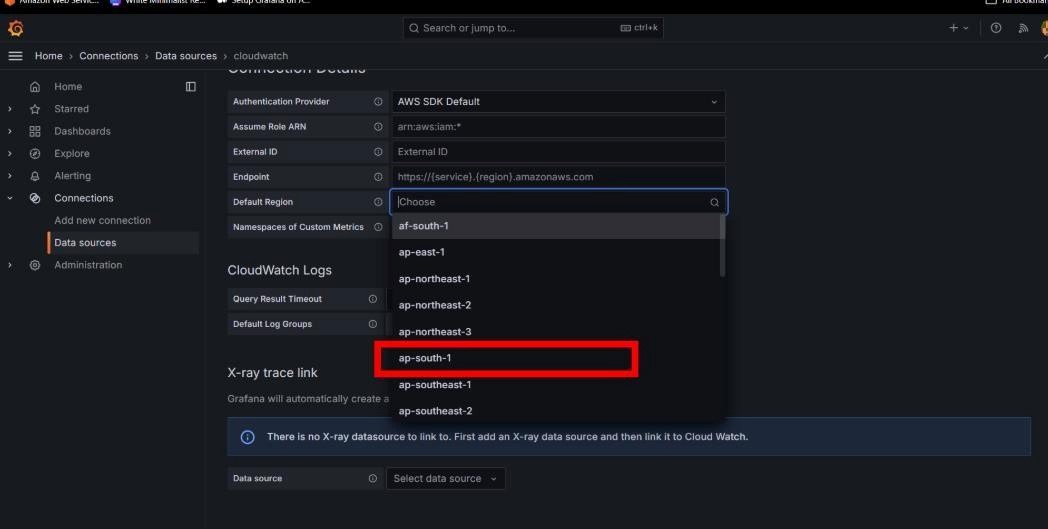




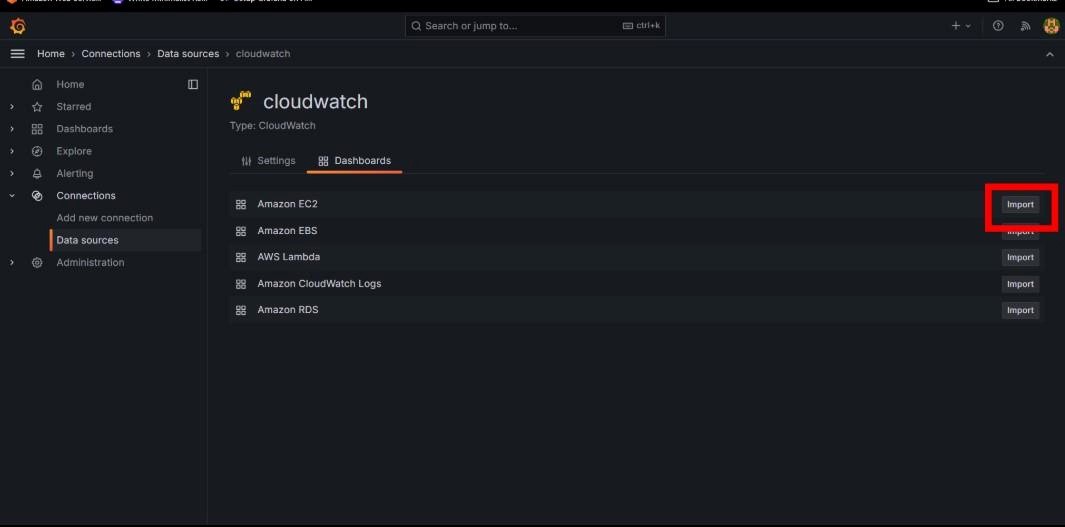
* Then go to **Dashboard** and click on **Adddata source** .



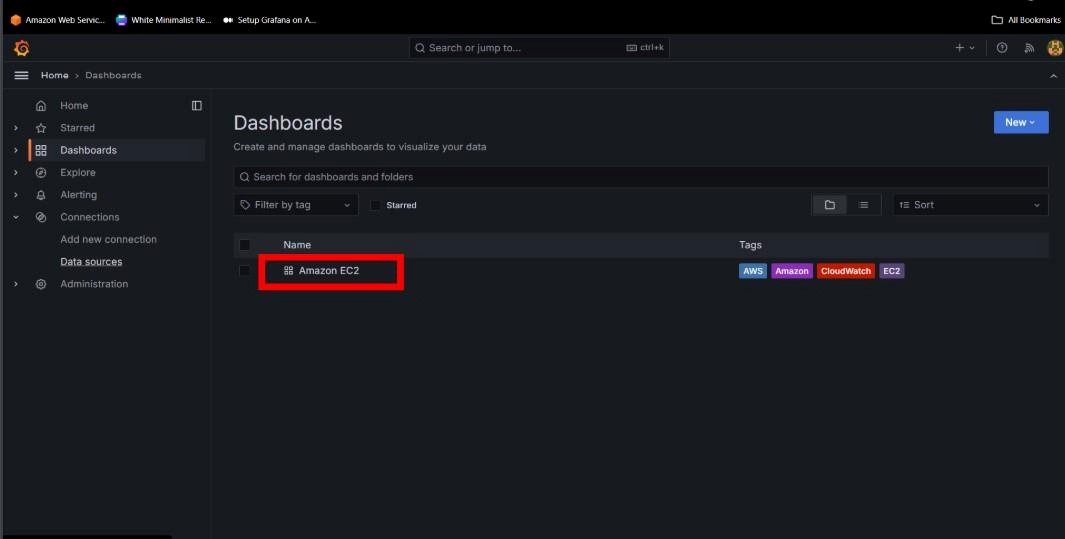
* Now , select “**Cloudwatch**” as an data source .



* Then after selecting cloudwatch select your region .



* Then inside your cloudwatch go to dashboard and **Import EC2**.



* Then go to dashboard and open your **Amazon EC2**.



Thank You Sir