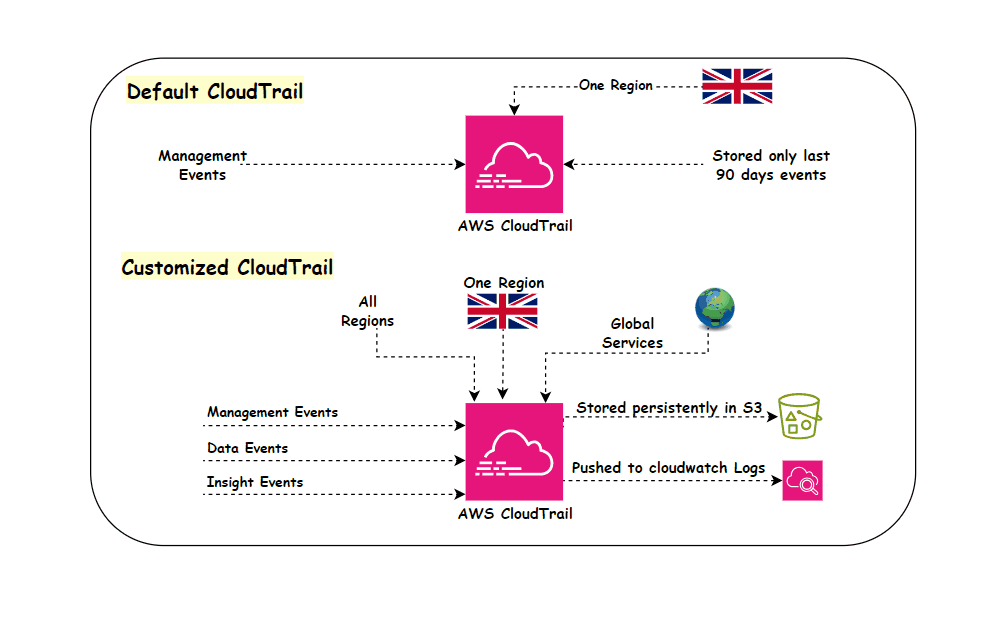
# AWS CloudTrail

# Introduction

AWS CloudTrail is like a detailed journal for your AWS account. It keeps track of everything that happens, like starting or stopping an instance, changing security settings, or creating and deleting buckets in S3.

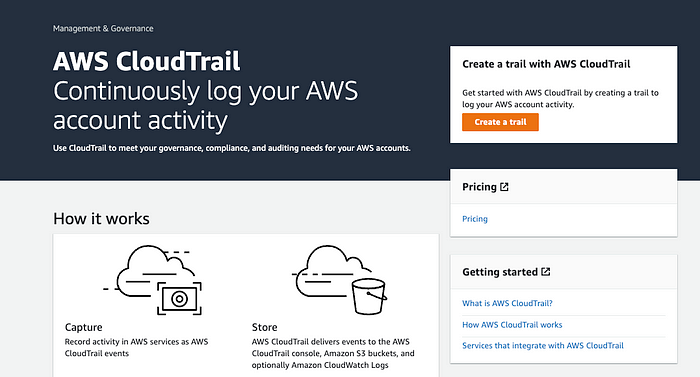
It’s like a detective that watches what every User, Role, or Service in your AWS account is doing. These records are called CloudTrail events and are super handy for finding out if something went wrong or just keeping a record of what’s happening.

**Note:** in CloudTrail Event history default enable in all region and it show the all activities in AWS account, delete logs is not possible in CloudTrail event

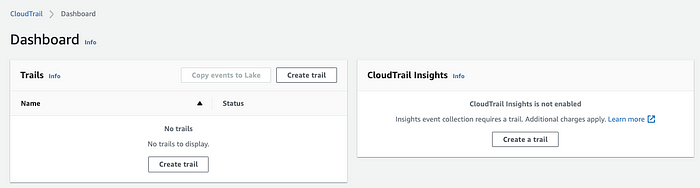


**Getting Started**

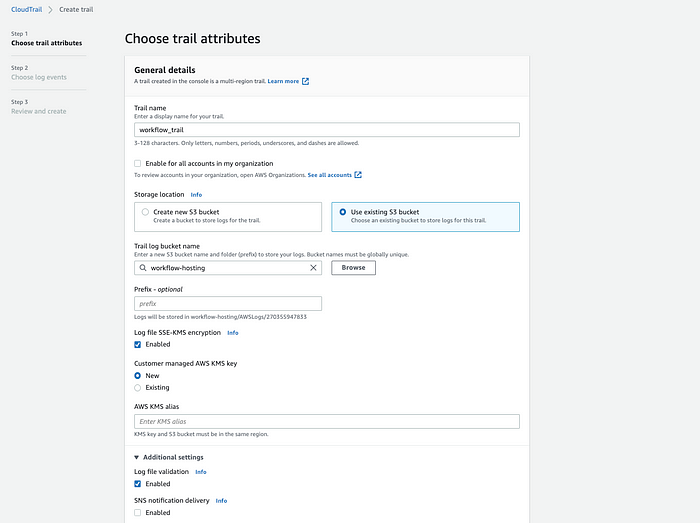
* In your AWS Management Console, search and click on AWS CloudTrail.



* Create a New Trail by clicking on Create Trail.



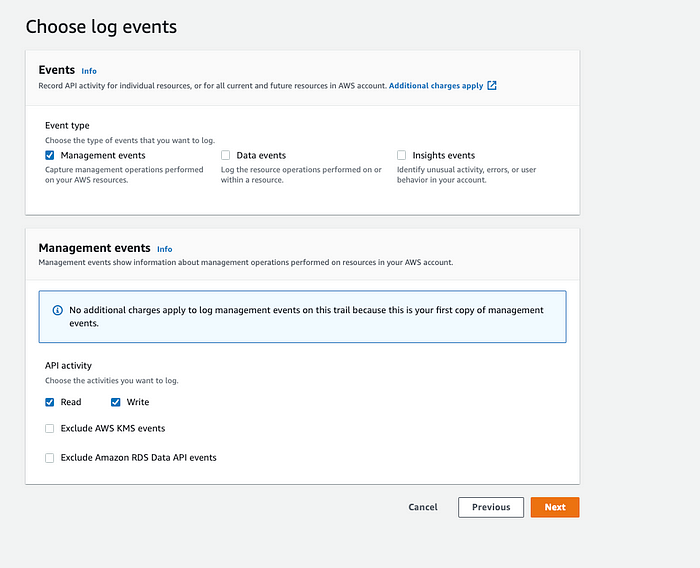
* Choose your Trail attributes. Enter your Trail name and storage location (select an existing S3 bucket or create a new S3 bucket). Enable your log file encryption with your file validation. This will ensure all aws resources are encrypted.



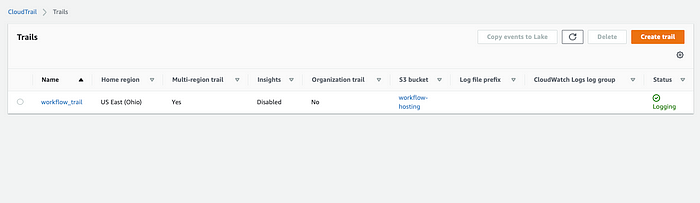
* When you’re done configuring your Trail attributes, click on **Next.**
* Next, choose your log events. In AWS CloudTrail, there are three types of events. Management events, Data events, and Insights events.

Management events are free and can be viewed in the event history tab for 90 days. Data events are not free to the user and cannot be viewed in the event history tab. Insights events let you identify unusual activity, errors, or user behavior in your account.

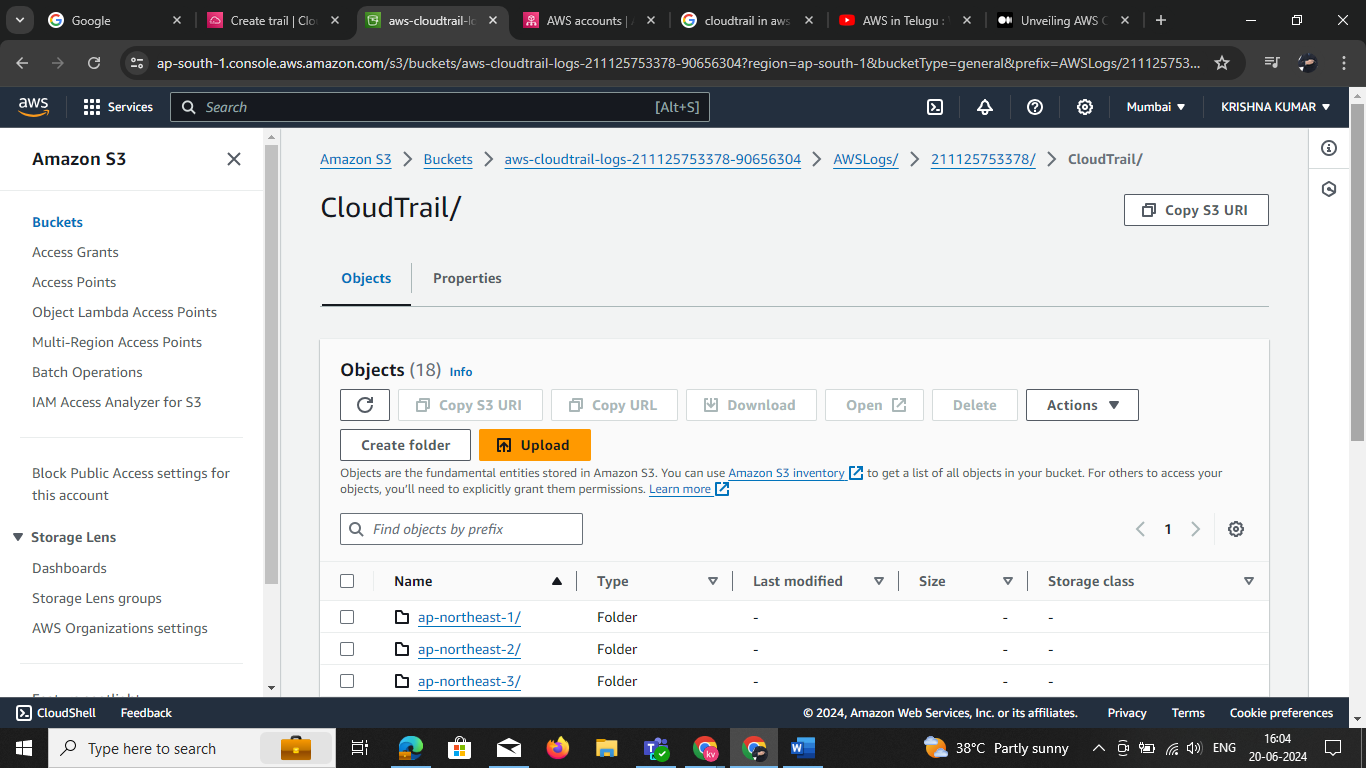
Only Management events are free for your workloads. Data and Insights events will incur costs. In this tutorial, we’ll be using **Management Events.**



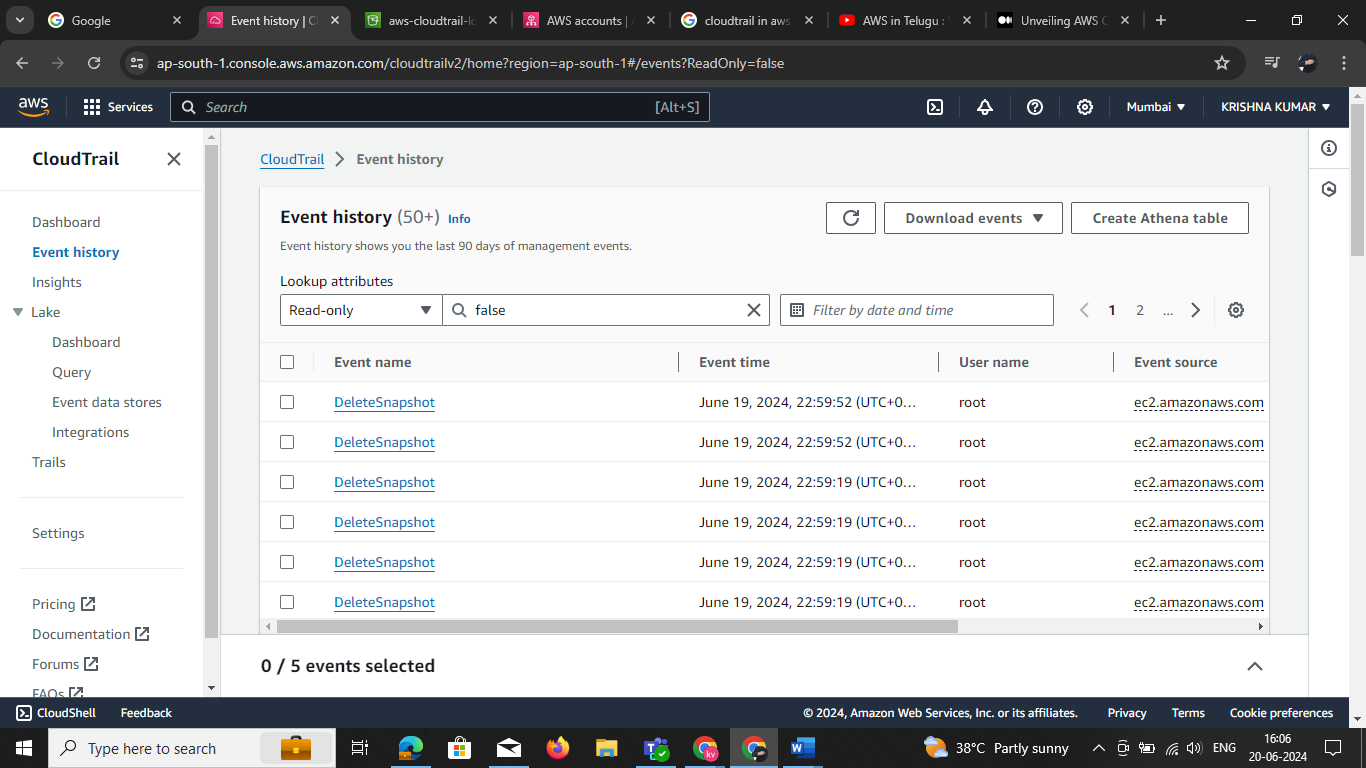
* When you’re done configuring log events, click on **Next**, you’ll see the overview and general details of your configuration, and click on **Create Trail.**
* In your Trails dashboard, you’ll see the Trail you just created.



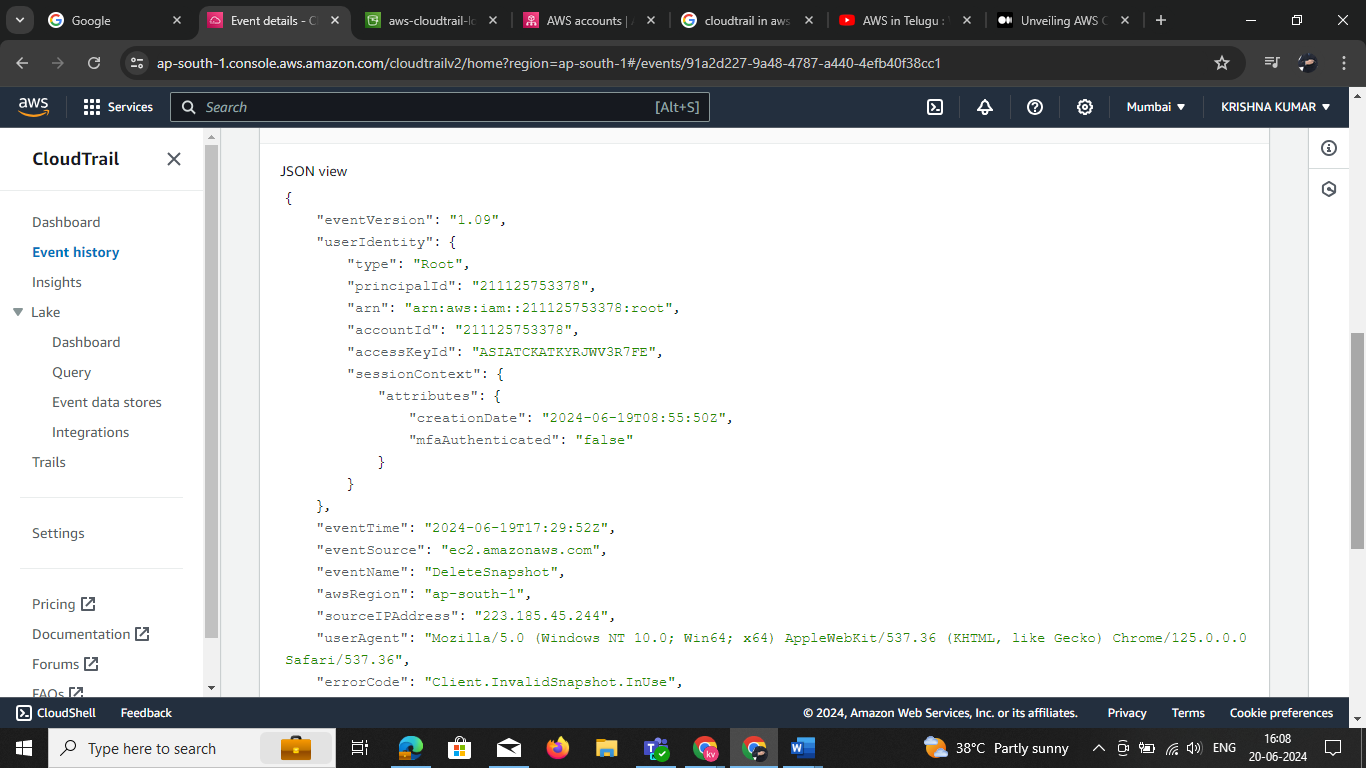
* Integrate other AWS resources with your trail to see how it works and see different log events. For example, in my S3 bucket, I’ll upload a new file into my S3 bucket. Once I’m done uploading the file, I’ll automatically see the events in my CloudTrail.



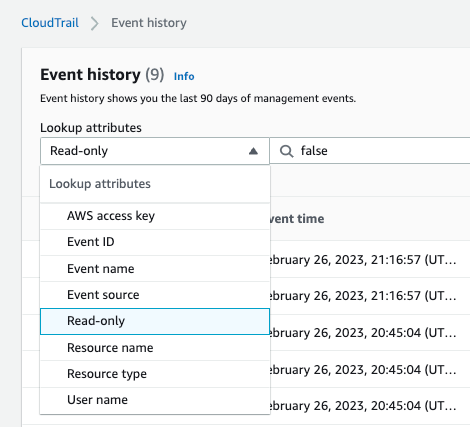
* In your CloudTrail event history, you’ll see all your events and logs from your S3 bucket.



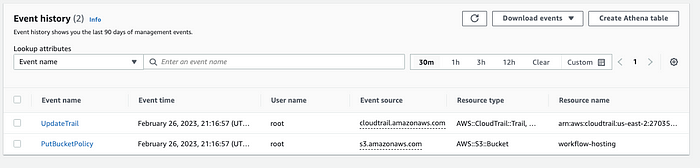
* You’ll see your event records and referenced resources when you click on them.



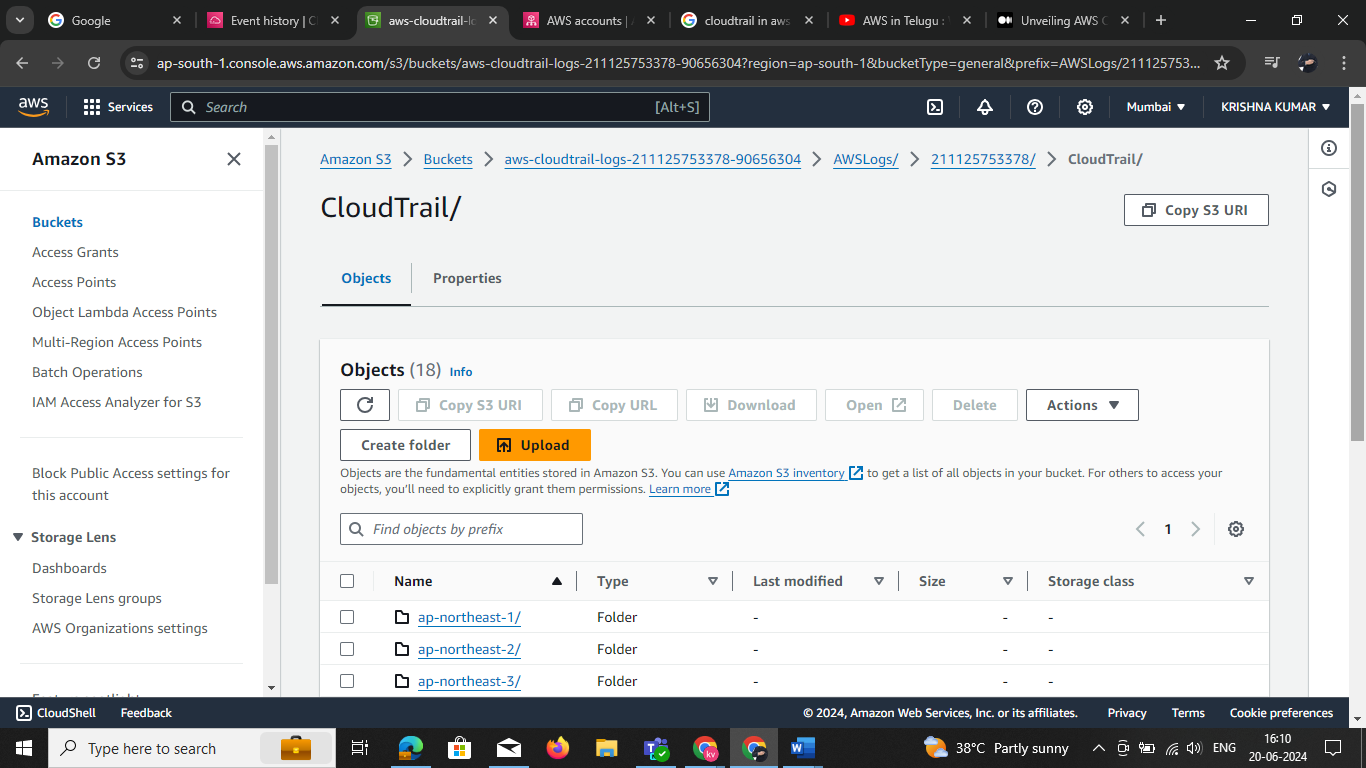
* You can also filter your event history based on AWS access key, Event ID, Event Name, Event Source, Resource name, and user type.



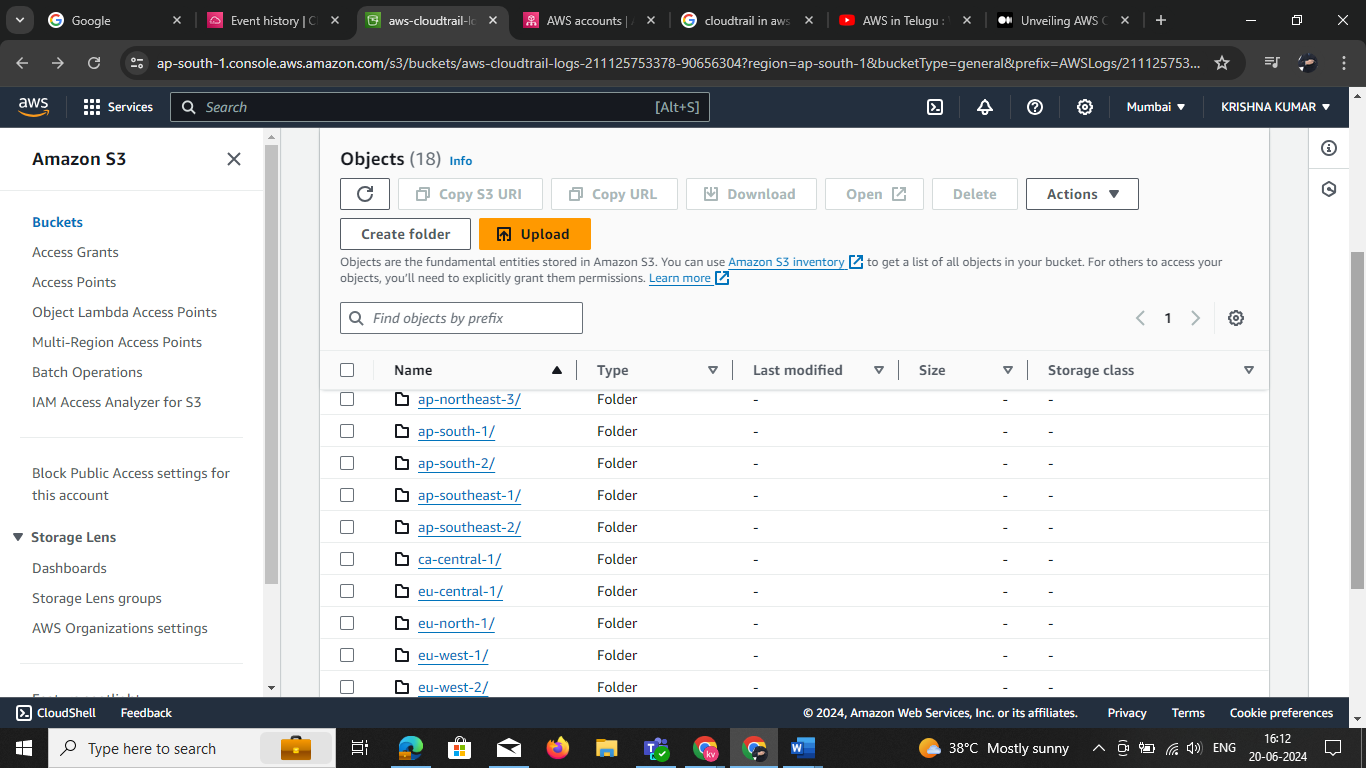
* You’ll see the PUT event history in your Event Name, the S3 bucket we updated earlier.



* In your AWS S3 storage bucket, you’ll see your CloudTrail log events in the AWS logs folder.



* When you click on Cloud Trail, you can see the logs from each AWS Region.



**Introduction to AWS Elastic Beanstalk**

You don’t need to understand the technology behind the applications to swiftly launch and maintain them in the AWS Cloud, thanks to Elastic Beanstalk. It relieves developers of the burden of worrying about the supporting infrastructure so they may concentrate on building code.

Elastic Beanstalk simplifies management without sacrificing control or choice. Elastic Beanstalk takes care of the capacity provisioning, load balancing, scaling, and application health monitoring; all you have to do is upload your application.

Applications written in Go, Java, .NET, Node.js, PHP, Python, and Ruby can be supported by Elastic Beanstalk. Elastic Beanstalk builds the chosen supported platform version and sets up one or more AWS resources, like Amazon EC2 instances, to run your application when you deploy it. The Elastic Beanstalk terminal, the AWS Command Line Interface (AWS CLI), or eb, a high-level CLI created especially for Elastic Beanstalk, can all be used to communicate with the platform.

Here are some key features and concepts related to Elastic Beanstalk:

**Platform as a Service (PaaS)**

As a PaaS product, Elastic Beanstalk gives developers a platform to launch and operate their apps while abstracting the underlying infrastructure. The underlying servers, networking, and other infrastructure components don’t need to be managed by developers; they may concentrate solely on writing their application code.

**Supported Platforms**

Programming languages and platforms supported by Elastic Beanstalk include but are not limited to Java,.NET, PHP, Node.js, Python, Ruby, and Go. Additionally, it supports different application containers and web servers.

**Easy Deployment**

Elastic Beanstalk application deployment is simple. AWS Code Pipeline is one continuous integration and deployment technology that you may integrate with, or use the AWS Management Console and AWS Command Line Interface (CLI).

**Auto Scaling**

Your application can be automatically scaled by Elastic Beanstalk based on demand. To adapt to variations in demand, you can set up auto-scaling rules that will vary the number of instances executing your application.

**Load Balancing**

Load balancing is integrated into Elastic Beanstalk and allows you to split up incoming traffic among several instances of your application. This enhances your application’s failure tolerance and availability.

**Managed Updates**

You can quickly update your application to a new version with Elastic Beanstalk. It updates without causing any downtime by utilizing strategies like rolling updates to keep your application running.

**Logging and Monitoring**

For logging and monitoring, Elastic Beanstalk connects with AWS services such as Amazon CloudWatch. You may monitor logs, create alarms, and learn more about how well your program is operating.

**Environment Configuration**

With Elastic Beanstalk, you may set up many environments (such development, testing, and production) for your application. It is possible for every environment to have unique configurations, resources, and settings.

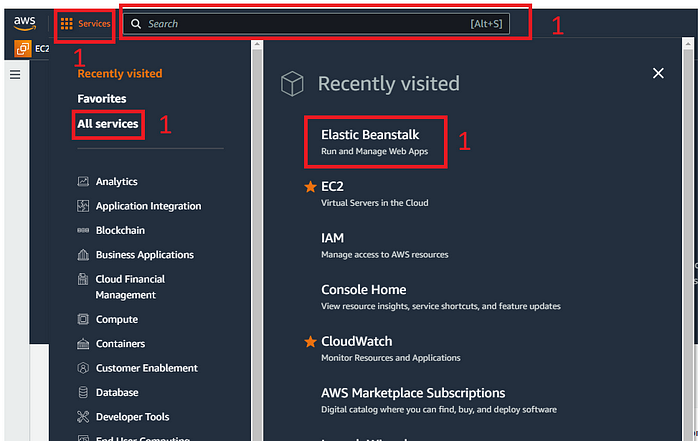
**Integration with Other AWS Services**

Numerous AWS services, including Amazon RDS for databases, Amazon S3 for storage, and Amazon VPC for networking, are integrated with Elastic Beanstalk.

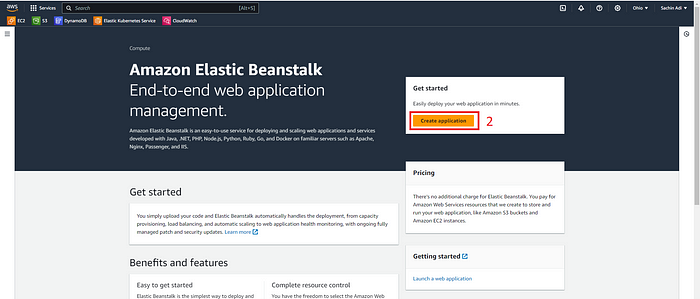
**Demo**

Let’s quickly check out a deployment that can show us how Elastic Beanstalk works.

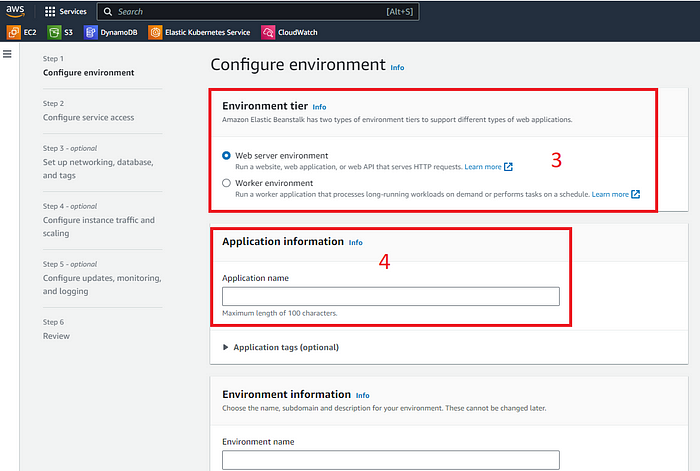
* Navigate to All Services and look for Elastic Beanstalk



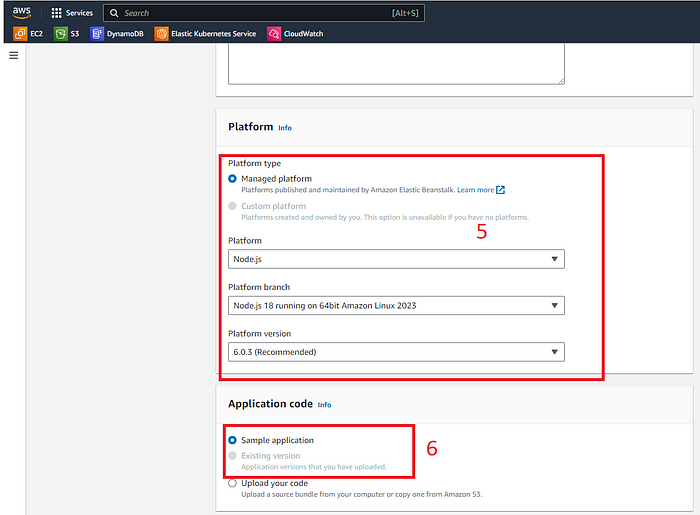
* Click on Create Application



* Select the Web Server Environment, input the Application Name

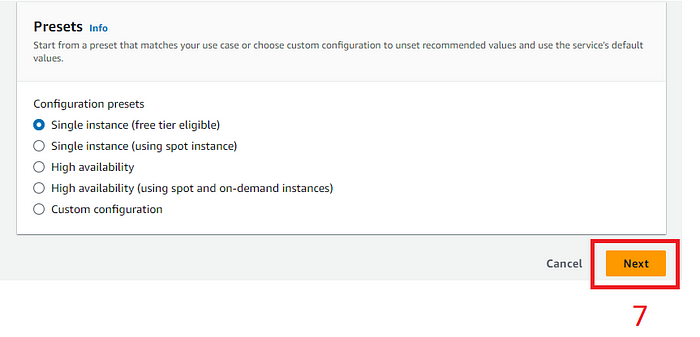


* Enter the platform type.

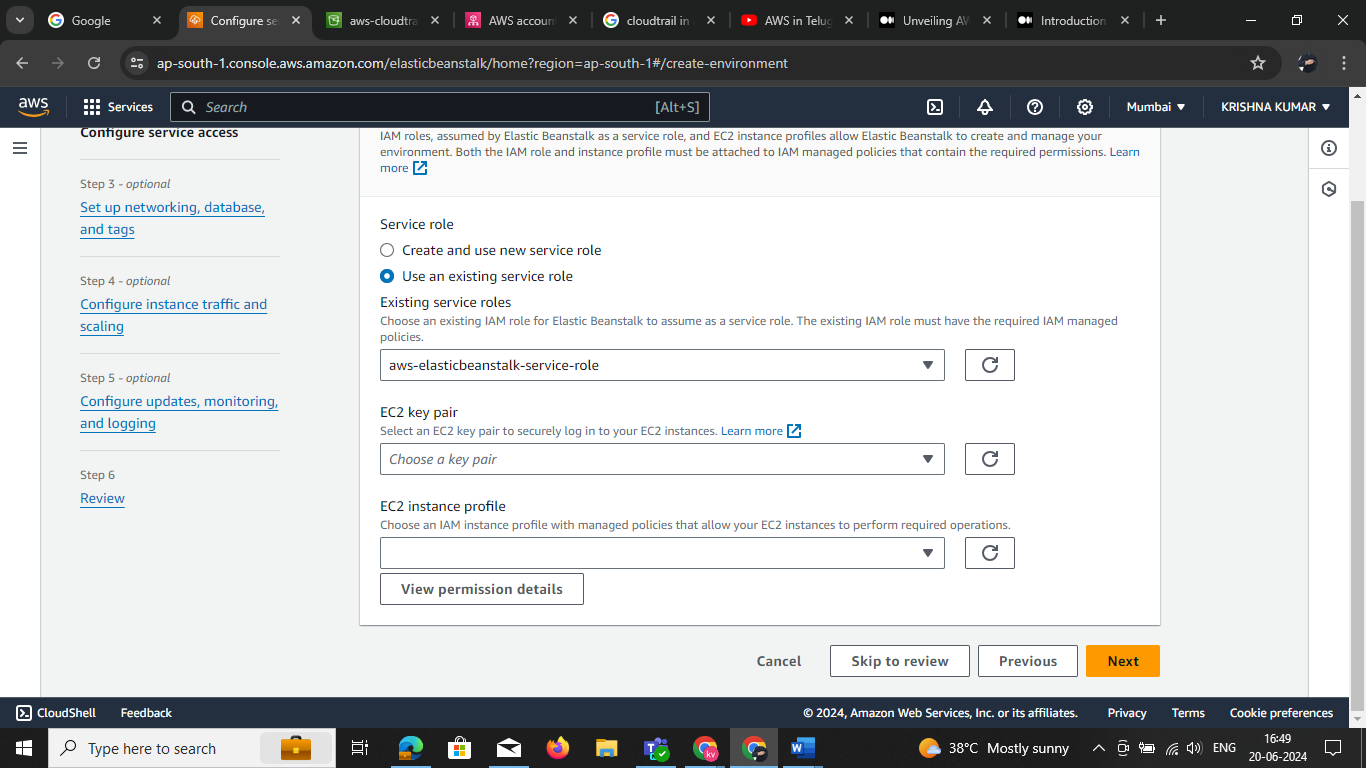


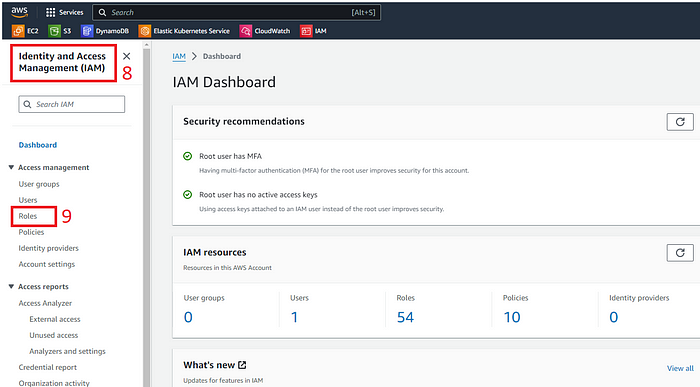
Platform specification

* Since we are just doing a test, I would keep it to a single instance

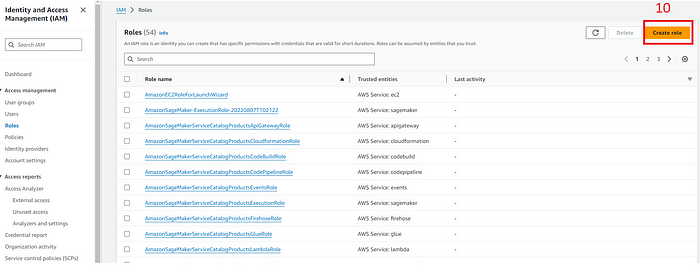


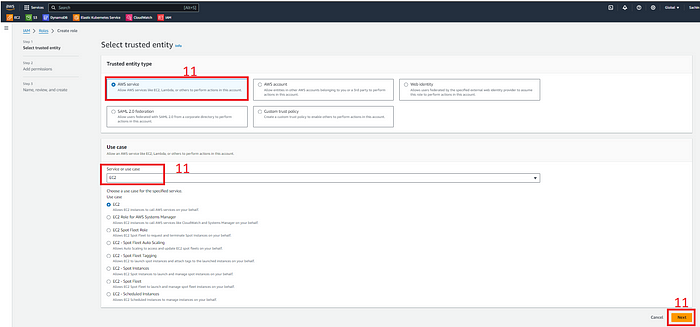
Configuring the presets

Now before we go any further, we would need an IAM role to work out with this project. So lets start creating this role. Create new role



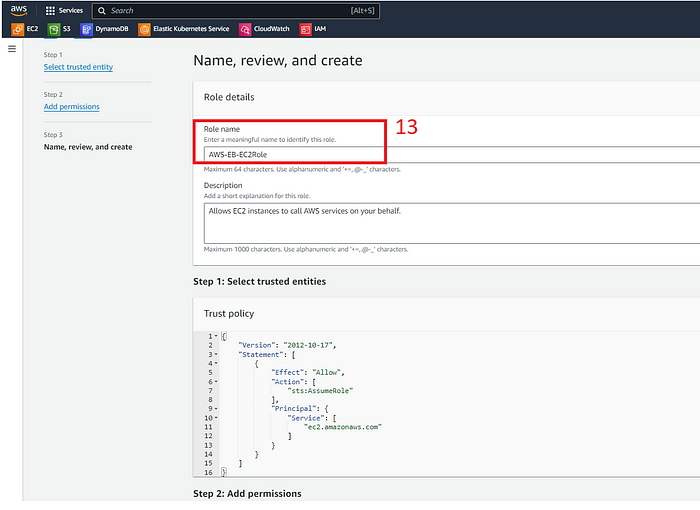
Identity and access Management configurations

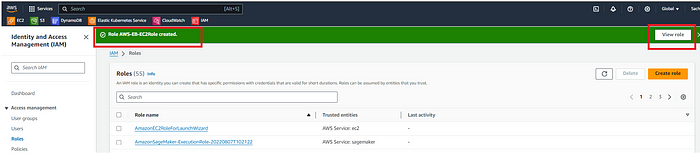




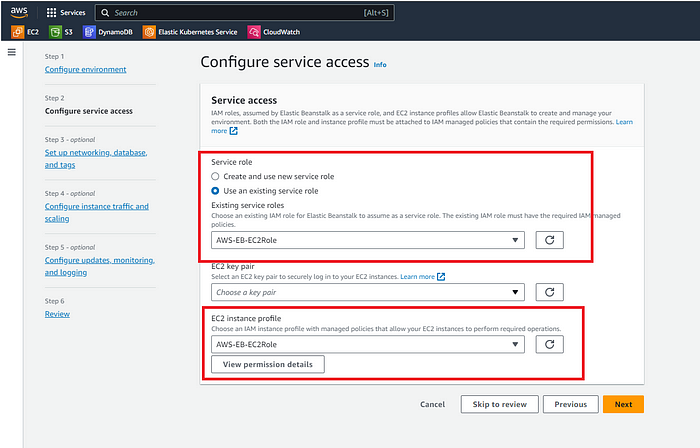
Role selection on the IAM portal

Look for the Elastic Beanstalk service and add the AWSElasticBeanstalkMulticontainerDocker, AWSElasticBeanstalkWebTier, and AWSElasticBeanstalkWorkerTier roles. Provide a name to this role and create the role.



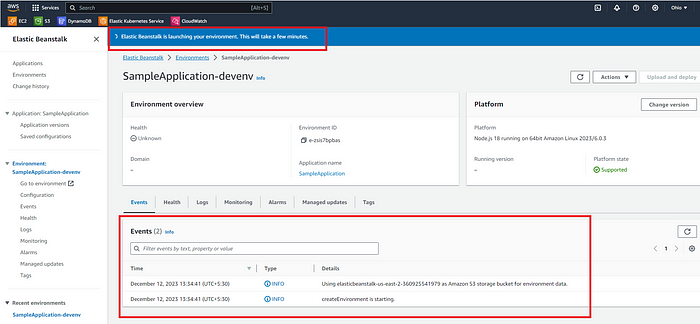


Back to the console where we were configuring the Elastic Beanstalk, we will need to add this IAM role

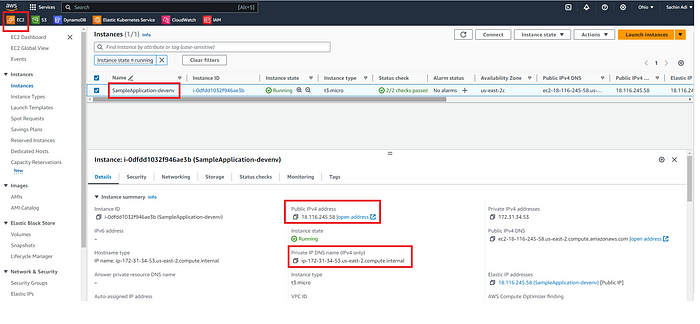


Configuring Service Access

Once you select all this and click on the Next button, the environment starts building up your architecture automatically.

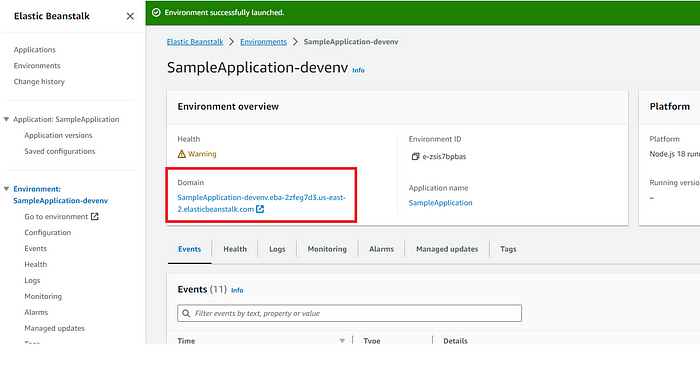


Elastic Beanstalk console

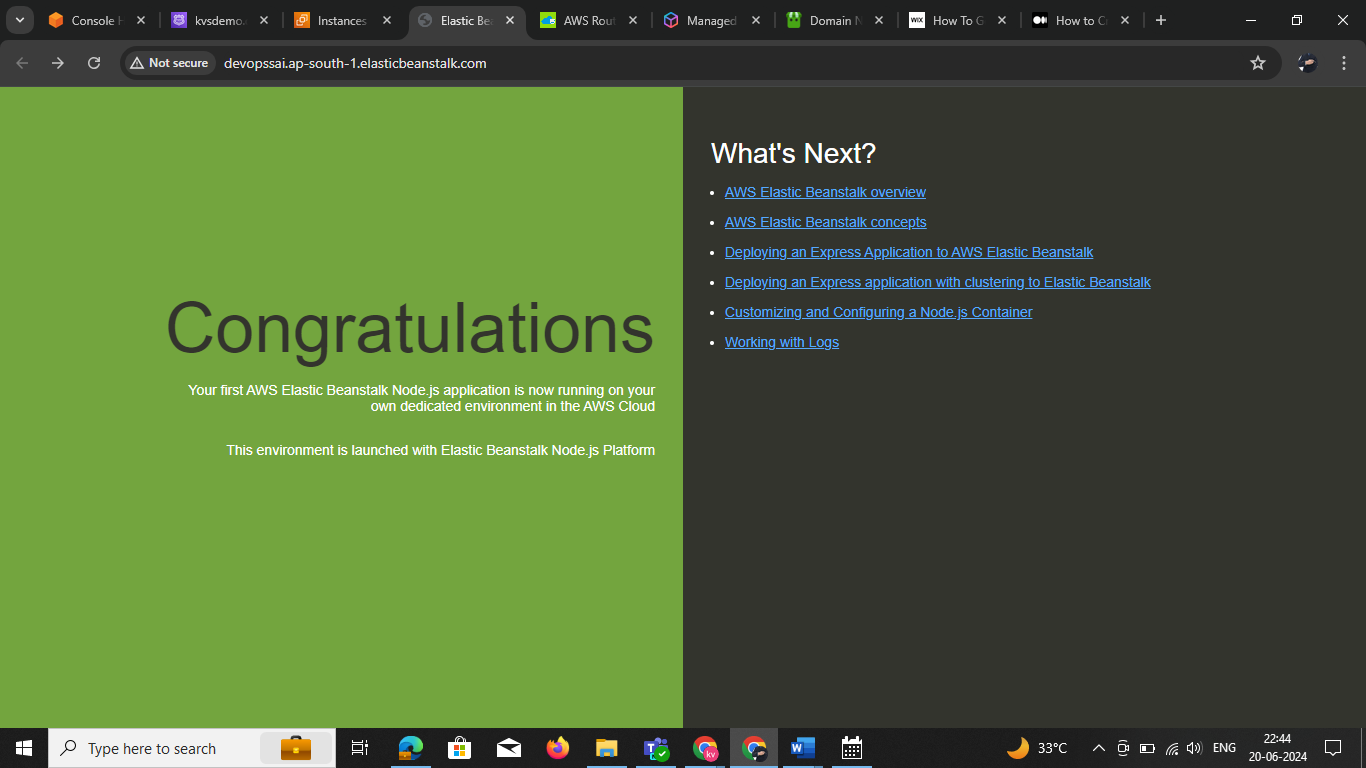


EC2 console displaying the virtual machine created during process

Look at the Domain that is shown on a successful creation of the environment:



Let’s access this and see if we can see a sample application page loading up:



Sample Application as displayed on the browser

Boom!! the deployment looked successful. We could see the app load up on the browser. Now there is one last step after all this hard work, to ensure that there is no charge on your account, terminate the resource you created.