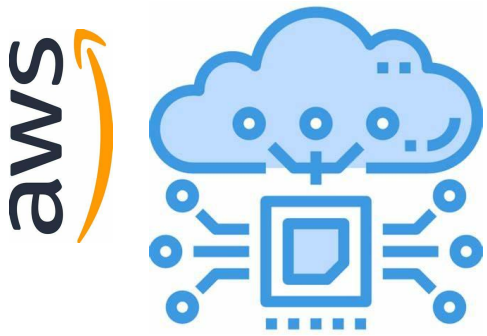


Cloud Computing Tutorials

Date: 15-03-2023

International Conference on Computing and Communication Systems (I3CS)-2023,
NEHU Shillong

Dr. Ferdous Ahmed Barbhuiya
Associate Professor
Department of CSE
IIIT Guwahati

&

Mehbub Alam
Doctoral Student
Department of CSE
IIIT Guwahati

Cloud Computing Experiments

Platform: Amazon AWS

1. Account Sign up in Amazon AWS
    a. It requires a valid credit card to validate the payment method
2. Get the account in service

Topic to be covered:

1. EC2 (Elastic Compute Cloud)
2. S3 (Simple Storage Service) Bucket
3. Amazon AutoScaling
4. Elastic BeanStalk
5. RDS

EC2

Elastic Compute Cloud

Amazon Elastic Compute Cloud (Amazon EC2) is a web service that provides resizable compute capacity in the cloud. It is designed to make web-scale cloud computing easier for developers. With Amazon EC2, you can provision and scale compute capacity in minutes and have complete control over your instances, including root access and the ability to install software.

Amazon EC2 offers a wide selection of instance types optimized to fit different use cases, such as compute-optimized, memory-optimized, and storage-optimized instances. You can choose from a variety of operating systems, including Amazon Linux, Ubuntu, Windows, and more.

To create an Amazon EC2 instance, follow these steps:

1. Sign in to the AWS Management Console
2. Go to services
3. Click on “Compute”
4. Select “EC2”
5. Click on “Launch an instance”
6. Under Name and tags, Give the Name of the Instance to be created.
    a. For eg, i3cs-aws
7. Select Amazon Machine Image as Amazon Linux.
8. Instance Type - t2.micro

9. Click on “Create new key pair” to create a new key pair.
    a. Give key pair name. Eg, i3cs2-key
    b. Key pair type - RSA
    c. Select .ppk for windows or .pem for linux users.
    d. Click on “create key pair”.
    e. Keep the .ppk or .pem file safely.
10. For Network configuration
    a. Create security group and select “Allow HTTPS traffic from Internet” and “Allow HTTP traffic from Internet”.
    b. If you already have created security group with proper ports open, select the pre-existing security group.
11. Keep “Configure Storage” as it is.
12. Add tags to the instances for easier management and organization.
13. Then Click on “Launch Instance”.
14. Review the instance details and launch the instance
15. Click on Services -> Then EC2 -> Then “Instances(Running)” to view all running instances.
16. Click on instance id to check details.

Security Group Creation-

:Configure security group settings to control inbound and outbound traffic.

1. Goto Services -> Compute -> EC2
2. On the left panel, Under “Network & Security,” Click on Security Groups.
3. On the top right corner, Click on security groups.
4. Fill in the basic details.
5. For Inbound/Outbound Rules depending on the type of software.
    a. For Eg, For software using MySQL, the Inbound and Outbound rules should contain MySQL/Aurora.
6. Click on “Create Security Group.”

S3 *Simple Storage Service*

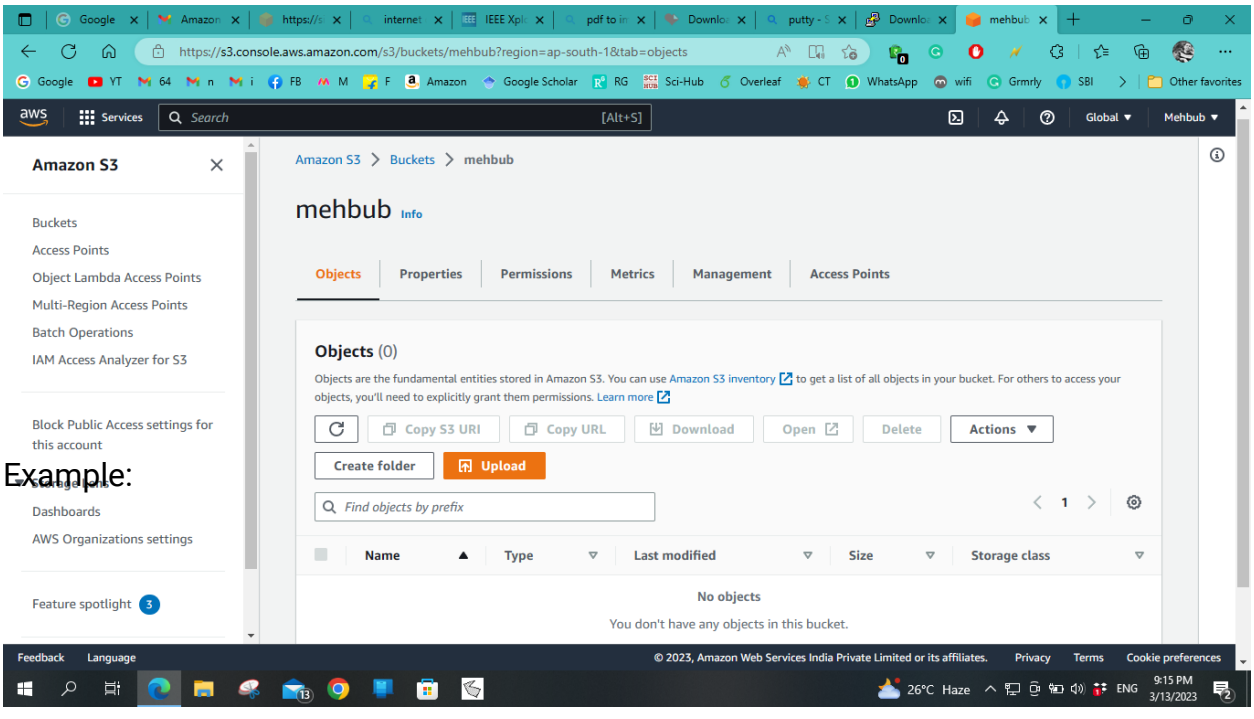
Amazon S3 (Simple Storage Service) is an object storage service provided by Amazon Web Services (AWS) that offers industry-leading scalability, availability, and durability. It is designed to store and retrieve any amount of data, at any time, from anywhere on the web. S3 is a cost-effective storage service with a simple web-based interface that can be used to store and retrieve any amount of data from anywhere in the world.

To create an S3 bucket, follow these steps:

1. Log in to the AWS Management Console
2. Services -> Storage -> S3
3. Click on the "Create Bucket" button
4. Provide a unique name for your bucket
5. Select the region where you want your bucket to be created
6. Set the properties and permissions for your bucket
7. Click on Bucket Name to enter the bucket to upload and access files.

Once you have created an S3 bucket, you can use it to store and retrieve data. You can upload and download files from your S3 bucket using either the AWS Management Console or the AWS SDKs.

Example:



Here is an example of how to use the AWS SDK for Python (Boto3) to upload a file to an S3 bucket:

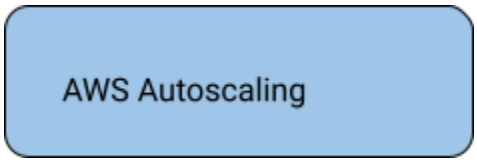
```
import boto3

# Create an S3 client
s3 = boto3.client('s3')

# Define the name of the S3 bucket and the filename
bucket_name = 'my-bucket-name'
filename = 'my-file.txt'

# Upload the file to the S3 bucket
s3.upload_file(filename, bucket_name, filename)
```

This code snippet will upload a file named my-file.txt to an S3 bucket named my-bucket-name. You can download files from an S3 bucket in a similar way using the download_file() method.



AWS Autoscaling Group is a service that allows you to automatically adjust the number of EC2 instances in your Amazon Web Services (AWS) environment based on changing demand. The Autoscaling Group automatically increases or decreases the number of instances in response to changes in demand or based on pre-defined conditions, ensuring that you always have the right number of instances running to handle the current workload.

Here is an example of how Autoscaling Group can be used:

Suppose you have a web application that typically receives a lot of traffic during peak hours, but not as much during off-peak hours. Instead of having a fixed number of instances running all the time, you can create an Autoscaling Group that automatically adds new instances during peak hours, and removes them during off-peak hours. This way, you only pay for the instances that you need, and your application can handle the changing demand without any manual intervention.

To create an Autoscaling Group in AWS, you will need to follow these steps:

1. Create a Launch Configuration: A Launch Configuration is a blueprint for your EC2 instances. It specifies the type of instance, the AMI to use, and any other configuration options you want to set. You can create a Launch Configuration using the AWS Management Console, the AWS CLI, or the AWS SDKs.

2. Create an Autoscaling Group: Once you have a Launch Configuration, you can create an Autoscaling Group. You will need to specify the minimum and maximum number of instances you want to run, as well as other parameters such as the scaling policies and the health checks.

3. Set up scaling policies: Autoscaling Group allows you to set up policies that determine when and how the group should scale up or down. You can set up policies based on a variety of metrics, such as CPU utilization or network traffic.

4. Test your Autoscaling Group: Once you have created your Autoscaling Group, you can test it by generating load on your application and observing how the Autoscaling Group responds.

To scale up or scale down an Autoscaling Group, you can use the following procedures:

1. Scale up: When you want to add more instances to your Autoscaling Group, you can increase the desired capacity of the group. The Autoscaling Group will automatically launch new instances based on your Launch Configuration and scaling policies.

2. Scale down: When you want to remove instances from your Autoscaling Group, you can decrease the desired capacity of the group. The Autoscaling Group will terminate instances based on your scaling policies.

Overall, Autoscaling Group is a powerful tool that allows you to automate the management of your EC2 instances, making it easier to handle changes in demand and ensuring that you always have the right number of instances running to handle your workload.

Implementation steps:

1. Manual Scaling up and down:

1. Go to Services -> Compute -> EC2
2. In EC2 Console on the left panel under “instances” find “Launch Templates”
3. Click on create launch templates
4. Insert Launch template name (eg.- i3csTemp)
5. Insert Template version description(eg.-v1)
6. Select OS: Amazon AWS
7. Select instance type as t2.micro
8. Key pair: use a previously created key pair or create a new pair of keys
9. Keep network configuration as default
10. Click on Advanced details: On user data box type the following shell script

```
#!/bin/bash

apt-get update

apt-get install nginx -y

service nginx start

echo "this is $(hostname)">/var/www/html/index.html
```

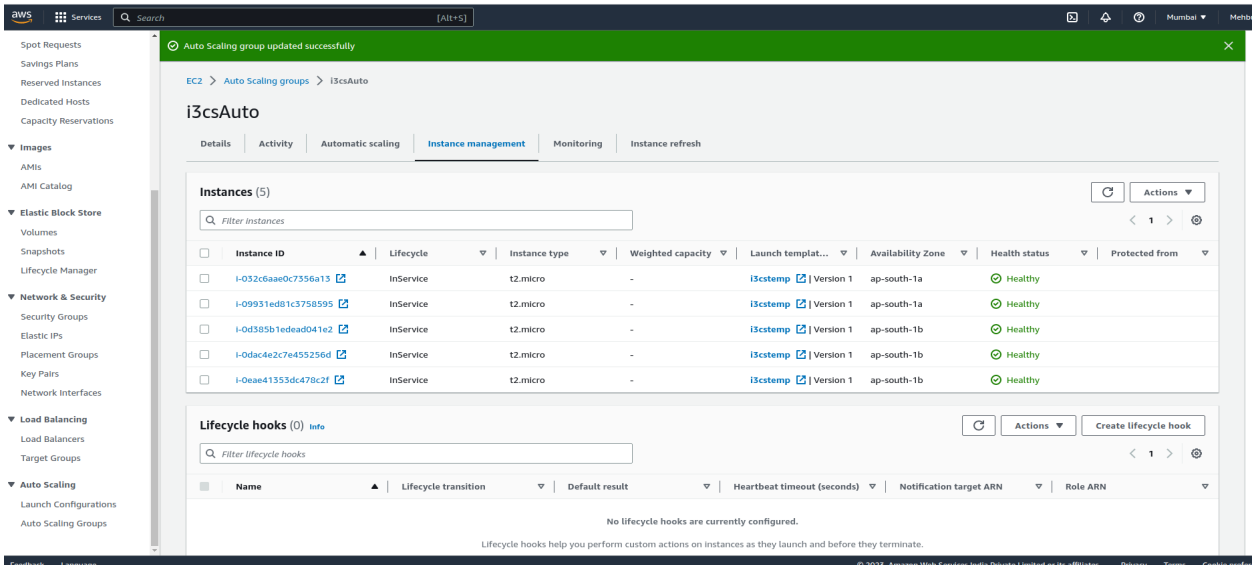
11. Click on the launch template and complete it.
12. Now, you can view the newly created template

Now, for autoscaling facility: eg. minimum 1 server and maximum 10 servers

1. Click on auto scaling on bottom left panel
2. Select auto scaling group
3. Create auto scaling group
4. Insert an auto scaling group name : i3csauto
5. In launch template: the one server will run on this templete which we have recently created i.e., i3csTemp
6. Next

7. Select VPC and available zones: your server runs on which available zones, let us select all
8. For instance type requirement keep it default
9. Next
10. Select No Load balancer
11. Health check grace period =300 sec i.e 5 min
12. Next
13. Desire capacity=1
14. Minimum capacity=1
15. Maximum capacity=100
16. Next
17. Next
18. Create autoscalling group
19. Click on autoscalling id to view the details
20. Go to instance management
    a. You can see the instance status as pending and after sometime(may require reload) it will change to InService.
    b. Check in activity: it should display “sucess”
21. Go to instance management again
22. See one running instance i.e., i3csTemp
23. Go back to EC2 dashboard
24. From running instances select the instance which is currently running
25. Now delete the instance.
    a. Go to instances
    b. Click on the instance state
    c. Click terminate
26. Reload and clear the filter
27. There should be no running instances
28. Reload
29. Automatically one instance should go on running mode
30. Go to the autoscaling group
31. Edit
32. Desire=5
33. Click on autoscaling id
34. Go to instance management

35. Voila!! 5 new instances!



All this scaling is manual, as if we know when traffic will be high and low.

Delete autoscaling group

All running instances will be terminated.

2. Automatic scaling:

Now 0 to 100 will be scaled up and down automatically. When the load is high, servers will be more and vice versa.

1. Go to Services -> Compute -> EC2
2. In EC2 Console on the left panel under “instances” find “Launch Templates”
3. Click on create launch templates
4. Insert Launch template name (eg.- i3csTemp)
5. Insert Template version description(eg.-v1)
6. Select OS: Amazon AWS
7. Select instance type as t2.micro
8. Key pair: use a previously created key pair or create a new pair of keys
9. **In network configuration: Select create security group**
    a. **Give a name: i3cs_ssh**

b. Description: allow i3cs ssh and webserver port

10. Rule 1: type:=ssh
11. Source: anywhere
12. Rule 2: type =HTTP
13. Source: anywhere
14. Click on Advanced details: On user data box type the following shell script

```
#!/bin/bash

apt-get update

apt-get install nginx -y

service nginx start

echo "this is $(hostname)">/var/www/html/index.html
```

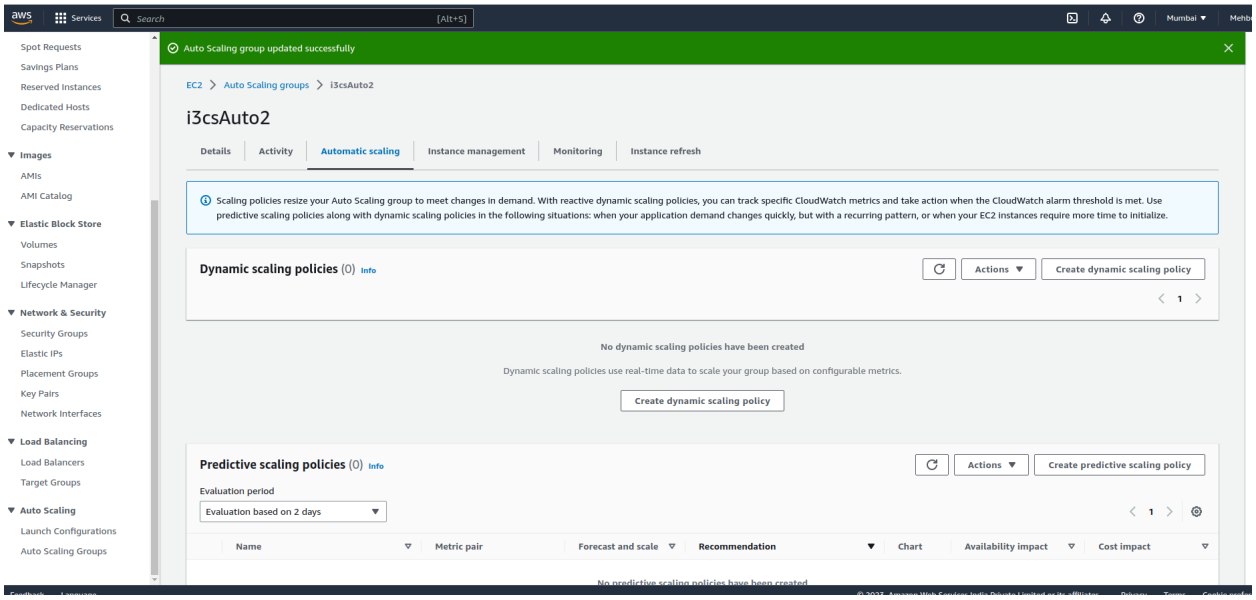
13. Click on the launch template and complete it.
14. Now, you can view the newly created template

Again repeat:

36. Click on auto-scaling on the bottom left panel
37. Select auto-scaling group
38. Create auto-scaling group
39. Insert an auto-scaling group name : **i3csauto2**
40. In the launch template: the one server will run on this template which we have recently created i.e., **i3csTemp2**
41. Next
42. Select VPC and available zones: your server runs on which available zones, let us select all.
43. For instance type requirement keep as default
44. Next
45. Select No Load balancer
46. Health check grace period =300 sec i.e 5 min
47. Next
48. Desire capacity=1
49. Minimum capacity=1

50. Maximum capacity=20
51. Next
52. Next
53. Create autoscaling group
54. Go to the EC2 dashboard
55. Check instances
56. Check there must be 5 instances in the running state

Now go to the autoscaling gr.oup



1. Scheduled actions: you know when your server will get more traffic. For example, on the day of results at a university website, the website of the election commission on the vote counting day, or on sale day in an e-commerce site.

Schedule: date time

Recurrence: every 5 minute

Time zone: select yours

2. Predictive scaling policies: prediction on previous data

Turn on scaling

3. **Dynamic scaling policies: dynamically scale**
    a. **Select simple scaling policy**
    b. **Give a name**
    c. **Set CloudWatch alarm: above 5% of one cpu , increase one more server**
    d. **You may add one**
    e. **You may add 10% of your servers**
    f. **Set to 7 if cpu utilization is 90%**
4. **For Step scaling, multiple conditions can be given**
5. **Target tracing scaling policy**
    a. **Average CPU utilization**
    b. **Target value 40%,**
    c. **After 100 seconds, scale up**
6. **Create**
7. **This is the procedure how we automatically scale up.**

Click on the running instance

Select

Click on connect: a new tab will open with the command editor

Type: top

To see the processor ids and their load, using the stress command, we can increase or decrease the CPU load.

To give load on instances, we can give multiple yum commands to install and check the instance dashboard.

How to autoscale using a python script?

Python code: Available at this link

Pre-requisite: Visual studio code or any other editor

Install Python in Visual studio code,

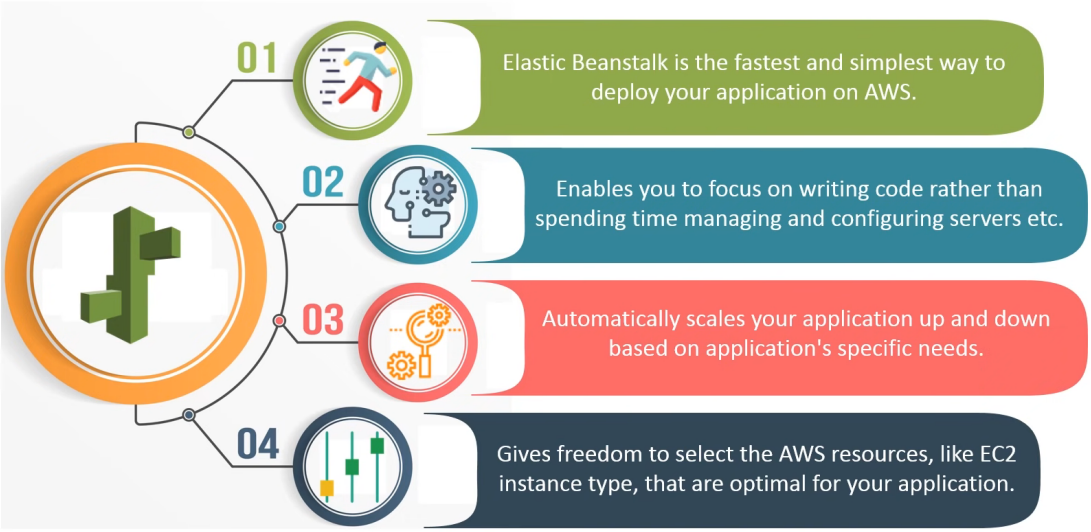
Run the code.

Note: you need to change the security key to your own key pairs, name of the bucket, name of the instance, Key name, launch configuration name, Image ID, security group id, etc

AWS Elastic Beanstalk

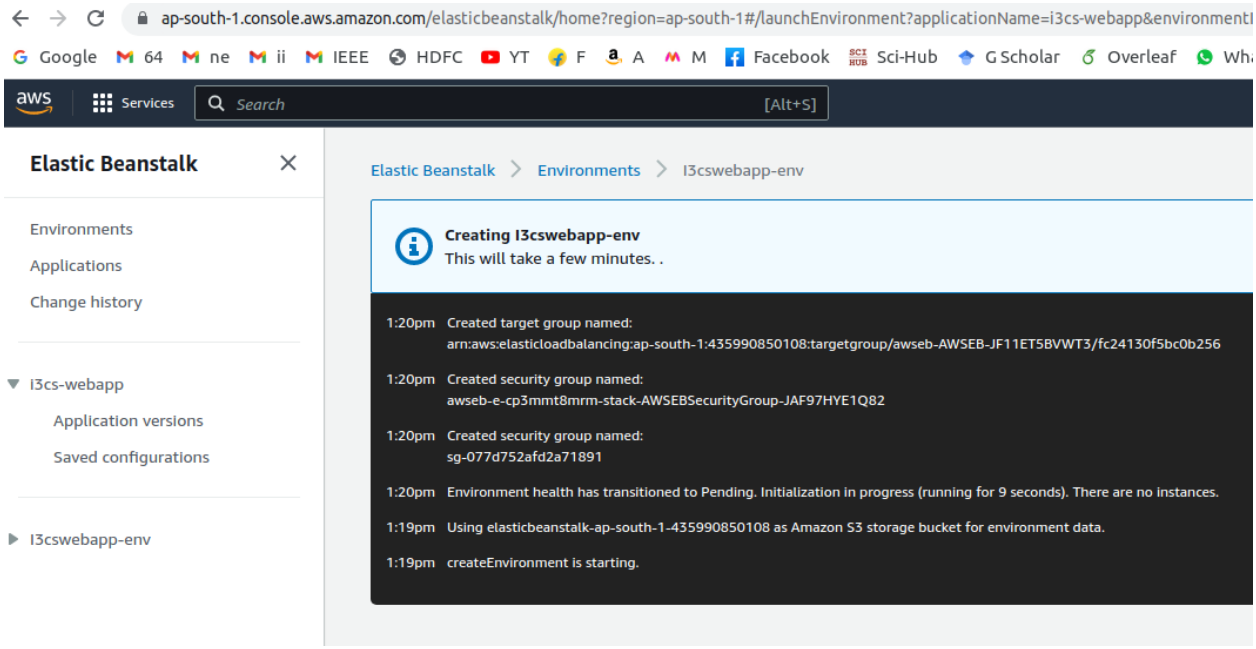
AWS Elastic Beanstalk is a fully-managed platform that allows developers to easily deploy and manage web applications. It supports several programming languages, such as Java, Python, Ruby, Node.js, PHP, .NET, and Go, and can be used to deploy web applications written in these languages.

Features of AWS Elastic Beanstalk



In summary, AWS Elastic Beanstalk provides a simple and efficient way to deploy and manage web applications. By following a few simple steps, you can quickly create an Elastic Beanstalk environment and deploy your application without worrying about the underlying infrastructure.

1. In services, search for elastic beanstalk
2. Click on Create
3. Give an Application name
4. Select a platform, for example: tomcat
5. Application code: let's keep it a sample from amazon AWS
6. Click on create application



It might take some time

7. Go to Configuration on the left panel
8. You can modify the following resources as required
    a. Software
    b. Instances
    c. Capacity:
        i. Single instance
        ii. Load balancer (autoscaling groups)
    d. Load balancer
    e. Rolling updates and deployments

f. Security
g. Monitoring
h. Network

9. In “logs”: we have the option to see full logs or recent 100 logs-it gives a log file to download in your system
10. Similarly, check health.
11. Click on the environment
12. Click on the application name
13. Click on the environment again from the list shown
14. From the left panel, click on “Go to environment” under l3cswebapp-env (the environment name)
15. Done! Your web application is launched!

Warning: Delete the auto-scaling groups you have created and terminate all the running instances!!

Thank you

For queries:
ferdous@iiitg.ac.in
mehbub@iiitg.ac.in