Terna Engineering College

Computer Engineering Department
Program: Sem VI

Course: Cloud Computing Lab (CSL603)

Faculty: PREETI PATIL

Experiment No.4

A.1 Aim:
To study and Implement Platform as a Service using AWS Elastic Beanstalk/ Microsoft Azure App Service.
A.2 Prerequisite:
Understanding of Virtualization,Basics of Networking ,Basics of security and privacy.

A.3 Objective:
To demonstrate the steps to deploy Web applications or Web services written in different languages on AWS Elastic Beanstalk/ Microsoft Azure App Service.

A.3 Outcome: (LO2)
After successful completion of this experiment students will be able to deploy the web application using AWS Elastic Beanstalk.

A.4 Theory:

AWS Elastic Beanstalk is an easy-to-use service for deploying and scaling web applications and services developed with popular programming languages such as Java, .NET, PHP, Node.js, Python and Ruby. You simply upload your application and Elastic Beanstalk automatically handles the deployment details of capacity provisioning, load balancing, auto-scaling and application health monitoring. At the same time, with Elastic Beanstalk, you retain full control over the AWS resources powering your application and can access the underlying resources at any time.

Most existing application containers or platform-as-a-service solutions, while reducing the amount of programming required, significantly diminish developers' flexibility and control. Developers are forced to live with all the decisions pre-determined by the vendor - with little to no opportunity to take back control over various parts of their application's infrastructure. However, with Elastic Beanstalk, you retain full control over the AWS resources powering your application. If you decide you want to take over some (or all) of

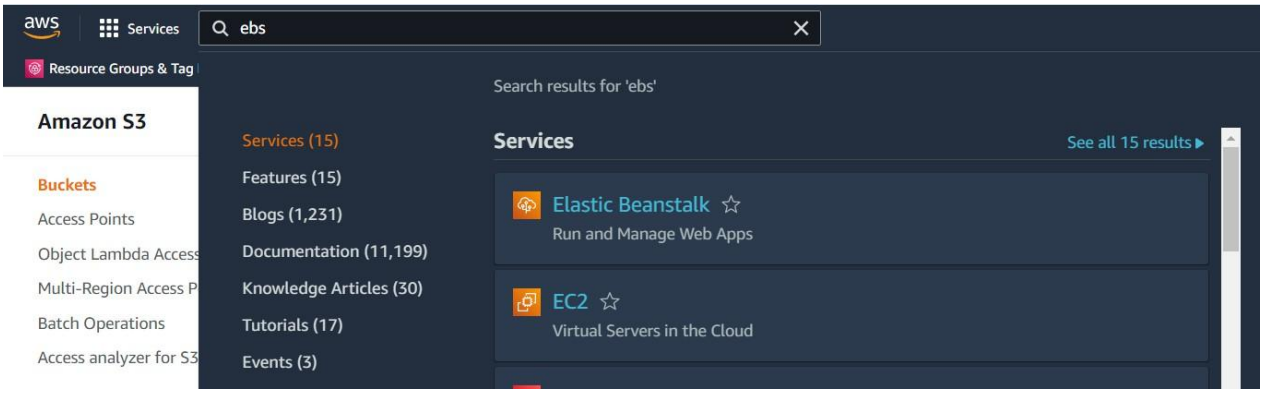
the elements of their infrastructure, you can do so seamlessly by using Elastic Beanstalk's management capabilities.

To ensure easy portability of your application, Elastic Beanstalk is built using familiar application/web servers such as Apache HTTP Server, Apache Tomcat, Nginx, Passenger and IIS 7.5/8.

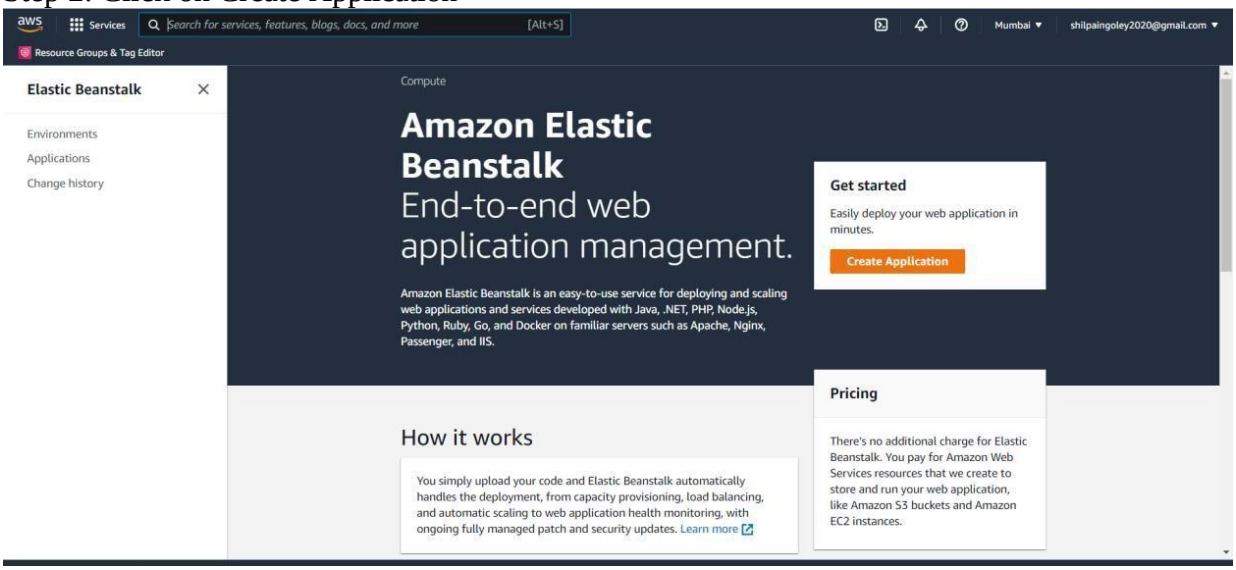
Following are steps to host a website using Elastic Beanstalk:

(Creating Sample Application)

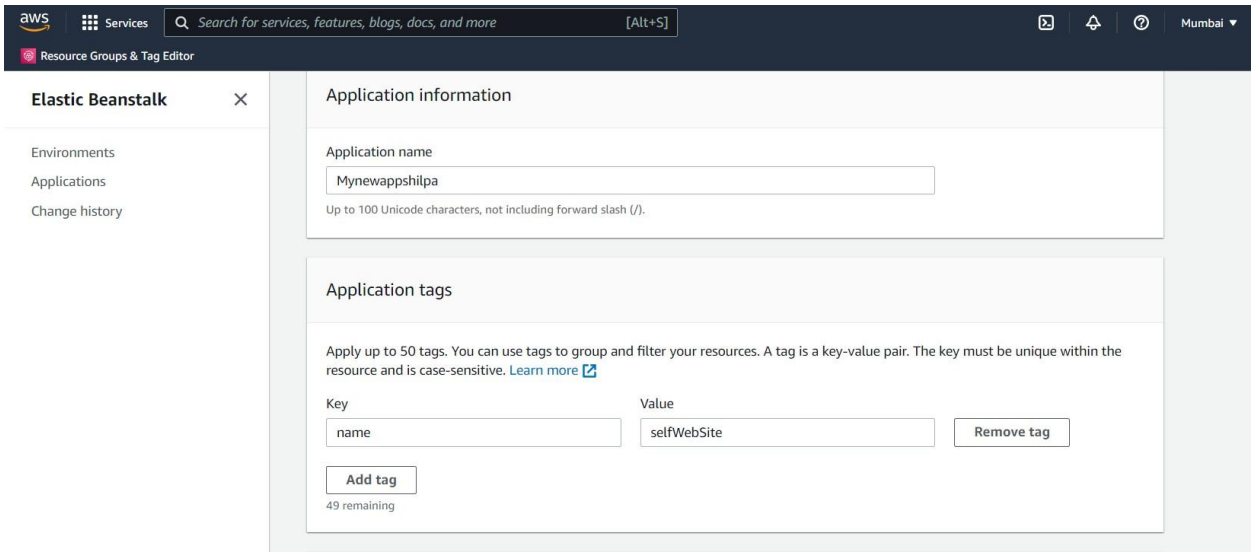
Step1 : Login to AWS console and go to Elastic Beanstalk



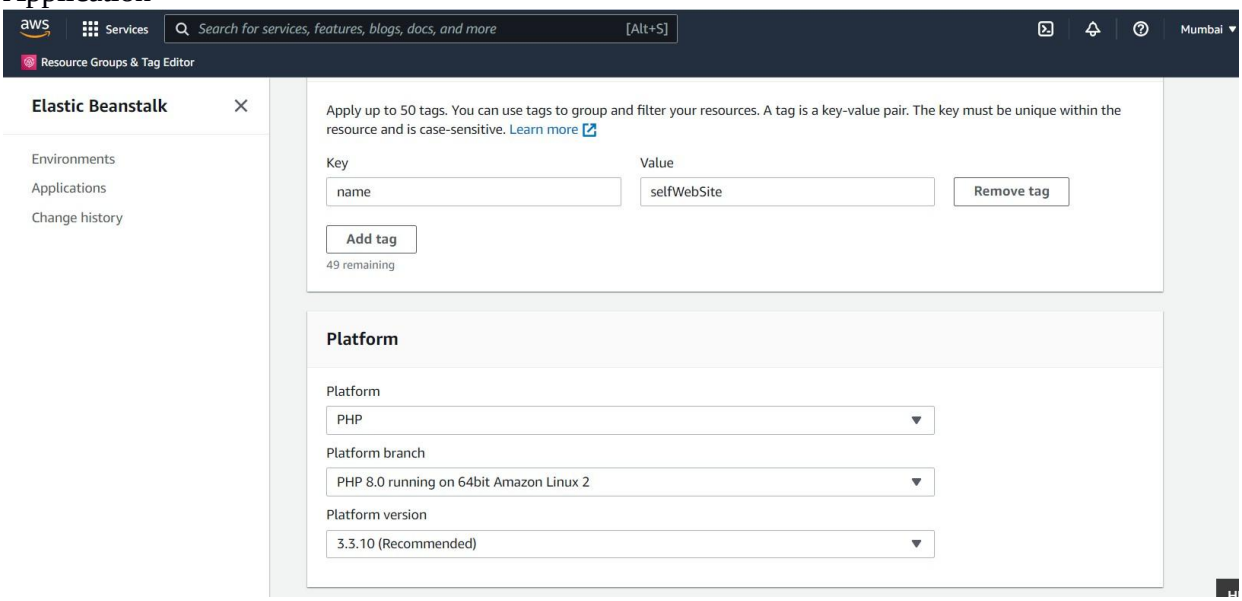
Step 2: Click on Create Application



Step 3: Write Application information: Name, Tag,Platform etc.

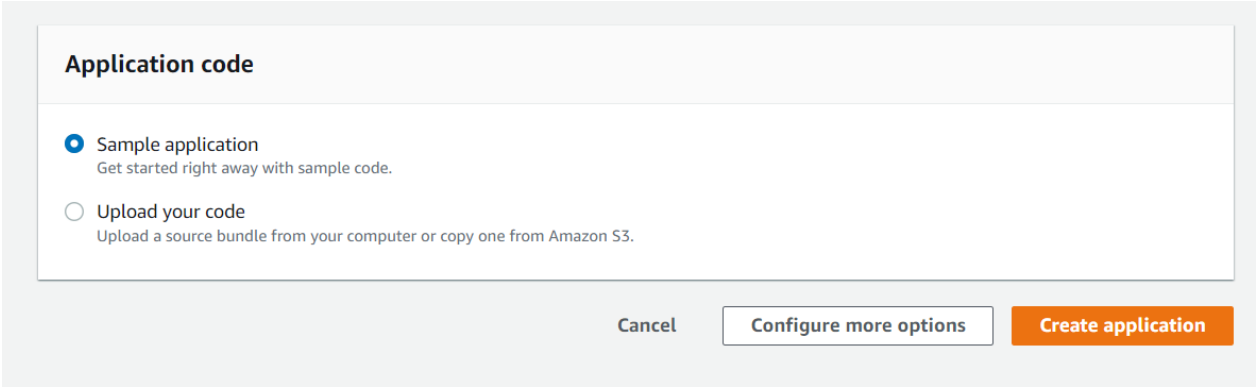


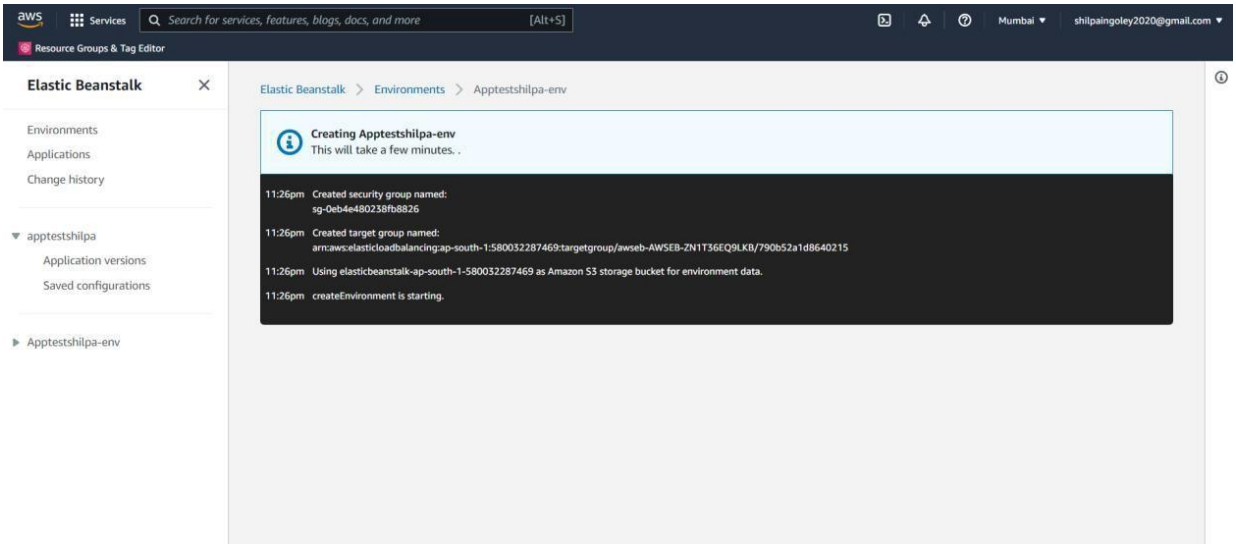
Step 4: In Application Code: select sample application and then Click on button Create Application



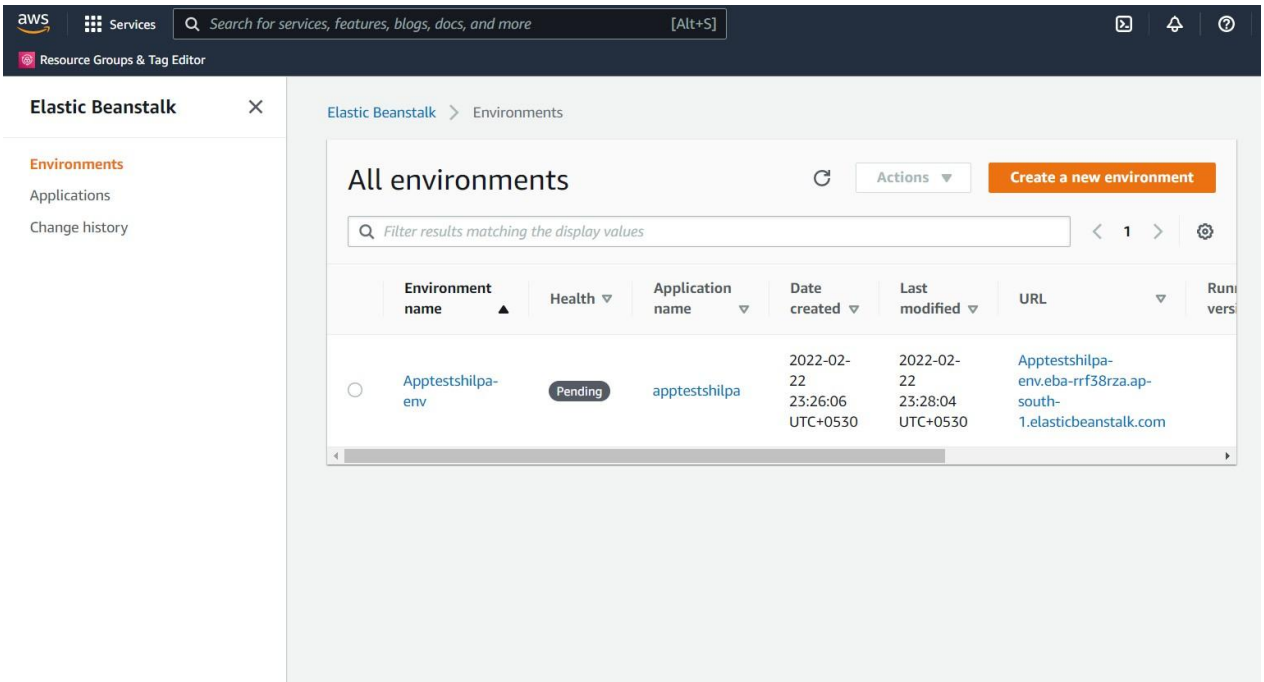
This will take a few minutes.

Step 5: Click on Environments -> Check the health of Environment wait till it becomes ‘OK’

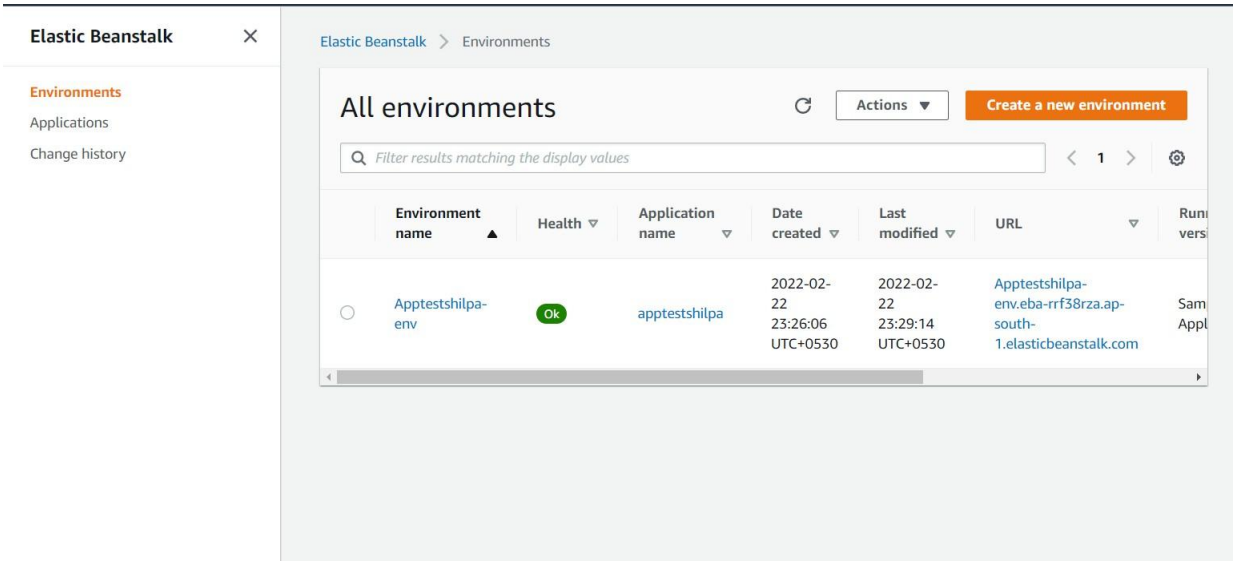




Step 6: Click the URL



To Delete the application and Environment (Select it and in **Action** -Delete/Terminate : give conformation)



Congratulations!

Your AWS Elastic Beanstalk *PHP* application is now running on your own dedicated environment in the AWS Cloud

You are running PHP version 8.0.13

This environment is launched with Elastic Beanstalk PHP Platform

What's Next?

- AWS Elastic Beanstalk overview
- Deploying AWS Elastic Beanstalk Applications in PHP Using Eb and Git
- Using Amazon RDS with PHP
- Customizing the Software on EC2 Instances
- Customizing Environment Resources

AWS SDK for PHP

- AWS SDK for PHP home
- PHP developer center
- AWS SDK for PHP on GitHub

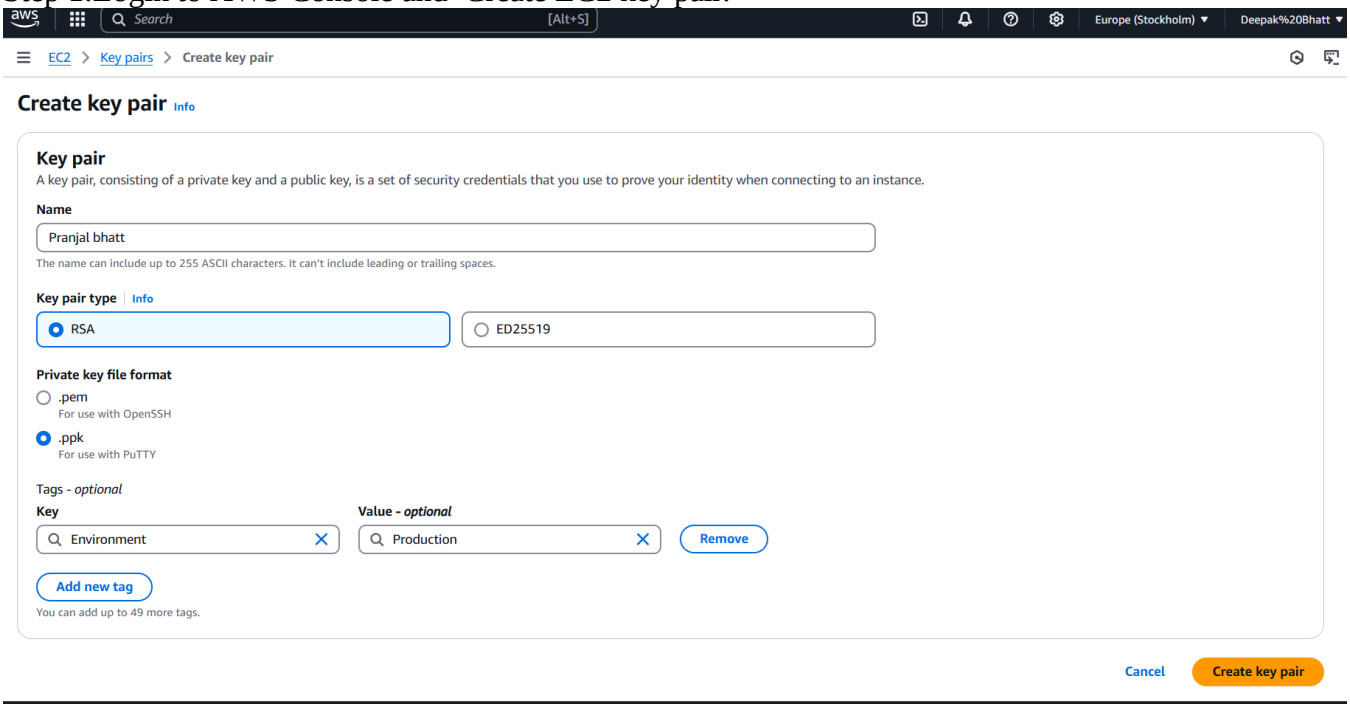
PART B

(PART B: TO BE COMPLETED BY STUDENTS)

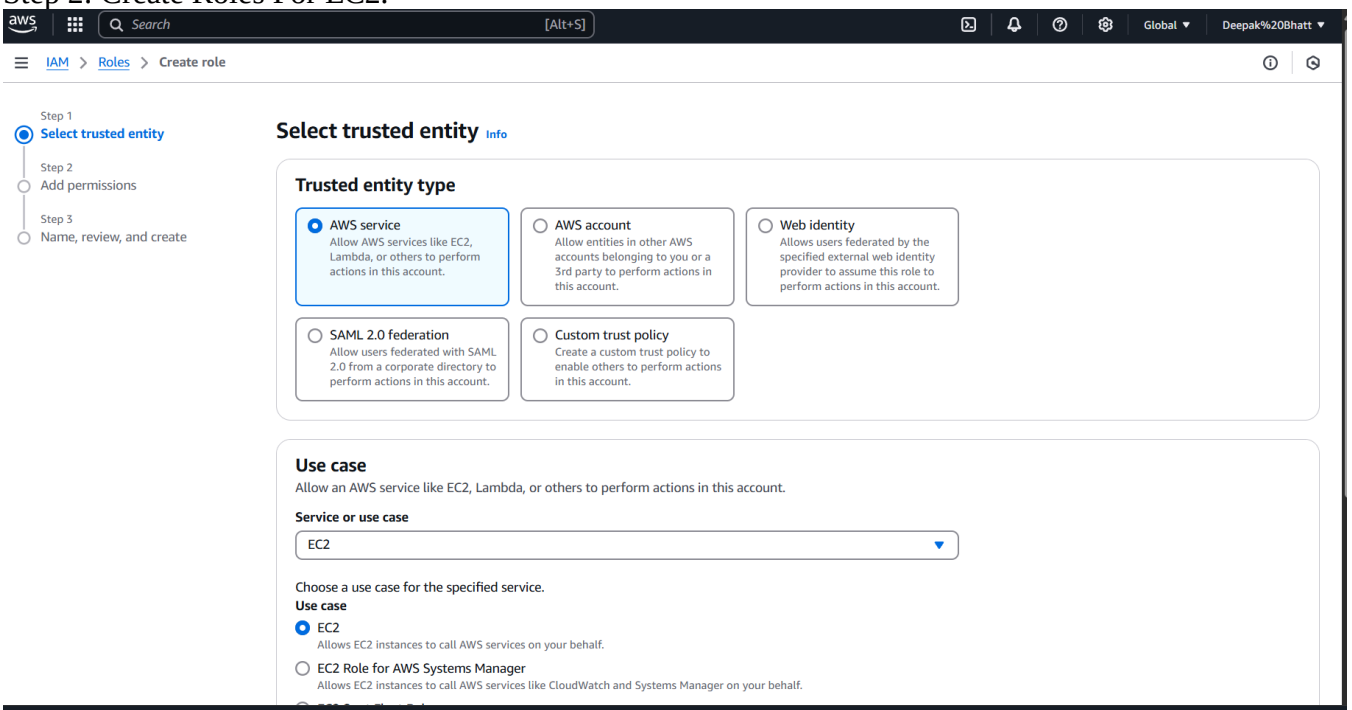
(Students must submit the soft copy as per following segments within two hours of the practical. The soft copy must be uploaded on the ERP or emailed to the concerned lab in charge faculties at the end of the practical in case the there is no ERP access available)

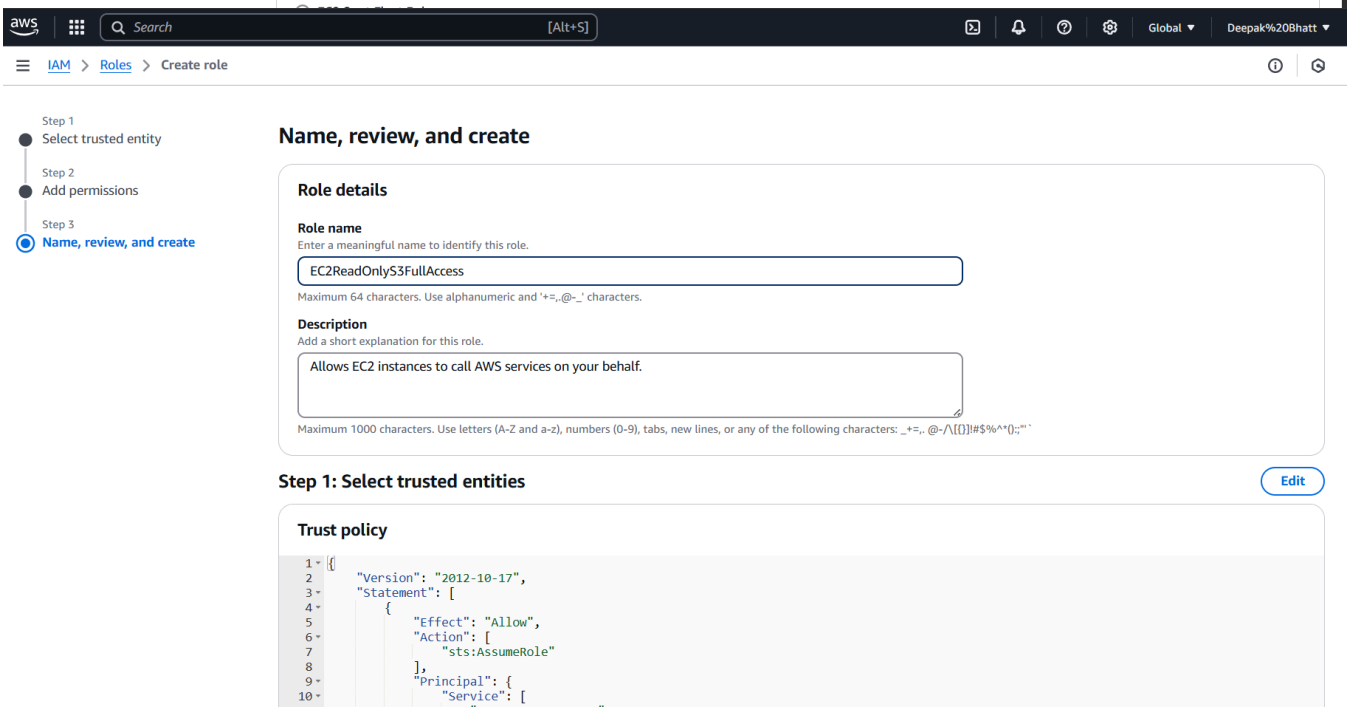
Roll No. B30	Name: Pranjal Bhatt
Class :TE COMPS B	Batch :B2
Date of Experiment:	Date of Submission:
Grade :	

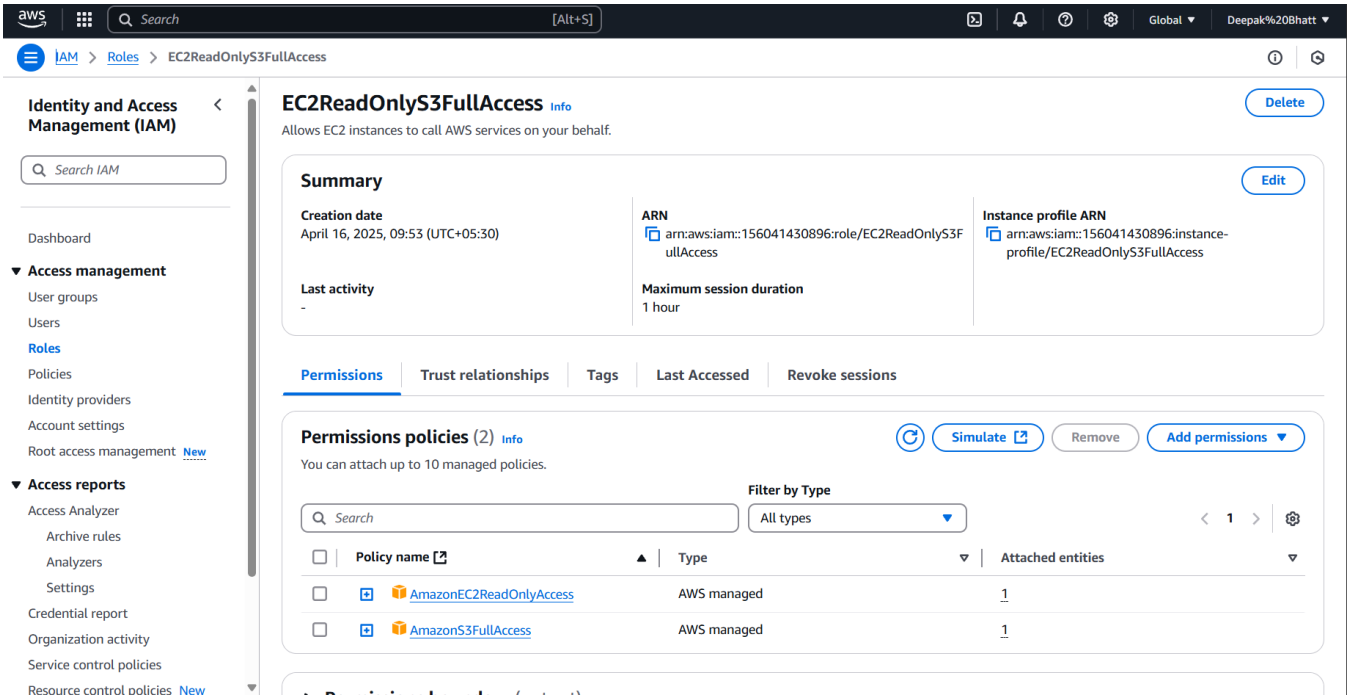
Step 1:Login to AWS Console and Create EC2 key pair.



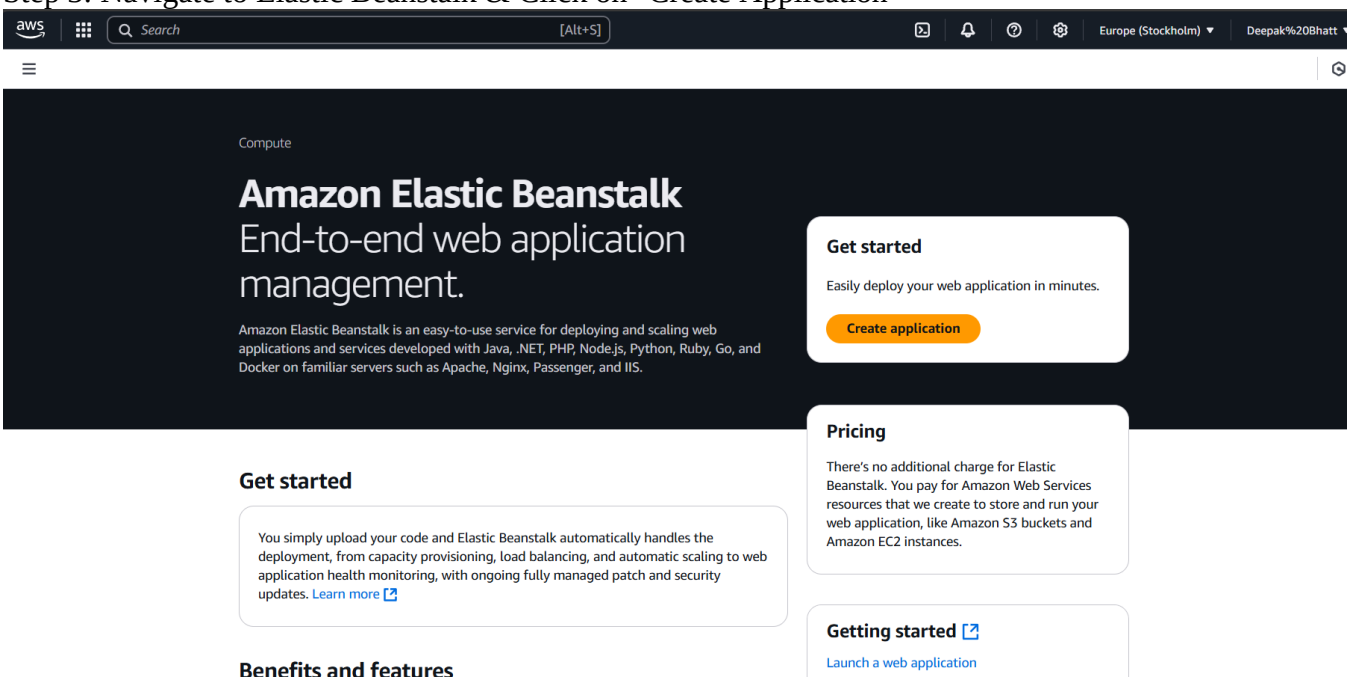
Step 2: Create Roles For EC2:



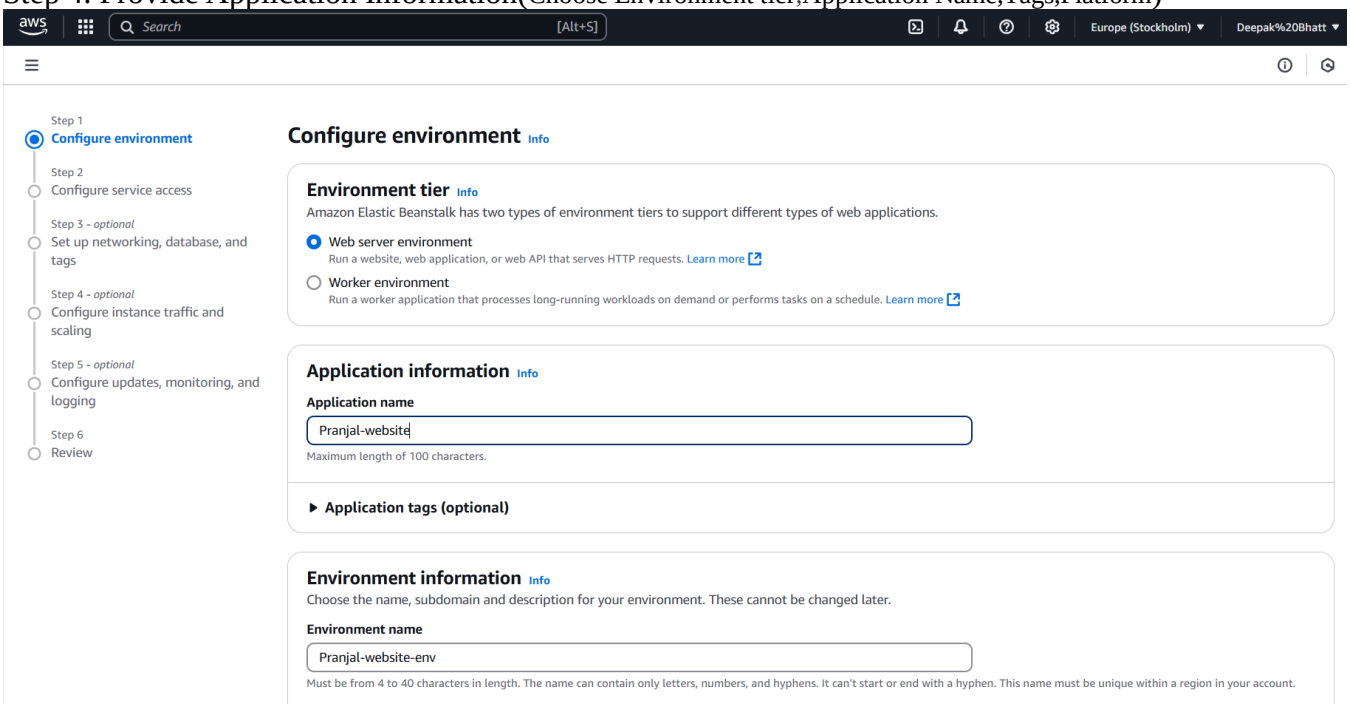


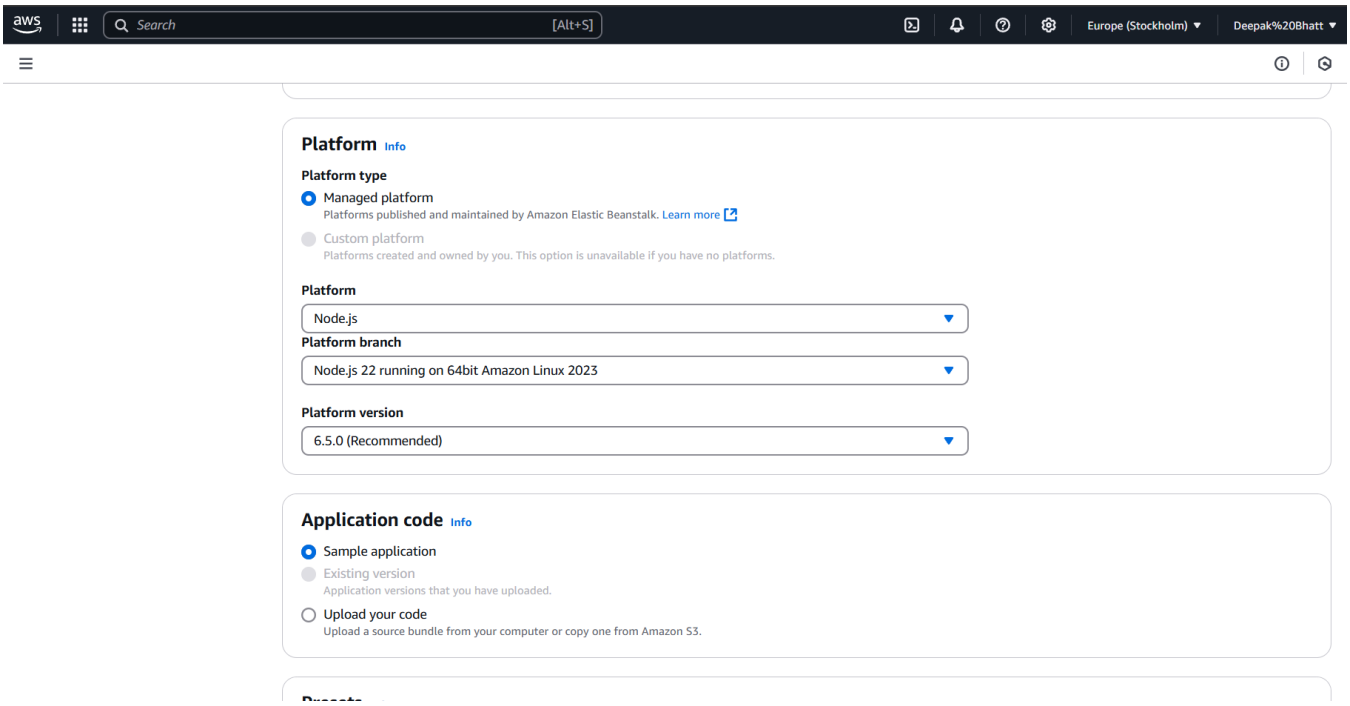

aws
Search
[Alt+S]
Global
Deepak%20Bhatt
IAM > Roles > EC2ReadOnlyS3FullAccess
Identity and Access Management (IAM)
Search IAM
Dashboard
Access management
User groups
Users
Roles
Policies
Identity providers
Account settings
Root access management New
Access reports
Access Analyzer
Archive rules
Analyzers
Settings
Credential report
Organization activity
Service control policies
Resource control policies New
EC2ReadOnlyS3FullAccess Info
Allows EC2 instances to call AWS services on your behalf.
Delete
Summary
Edit
Creation date
April 16, 2025, 09:53 (UTC+05:30)
ARN
arn:aws:iam::156041430896:role/EC2ReadOnlyS3FullAccess
Instance profile ARN
arn:aws:iam::156041430896:instance-profile/EC2ReadOnlyS3FullAccess
Last activity
-
Maximum session duration
1 hour
Permissions
Trust relationships
Tags
Last Accessed
Revoke sessions
Permissions policies (2) Info
You can attach up to 10 managed policies.
Simulate
Remove
Add permissions
Filter by Type
Search
All types
1
Policy name
Type
Attached entities
AmazonEC2ReadOnlyAccess
AWS managed
1
AmazonS3FullAccess
AWS managed
1

Step 3: Navigate to Elastic Beanstalk & Click on "Create Application"

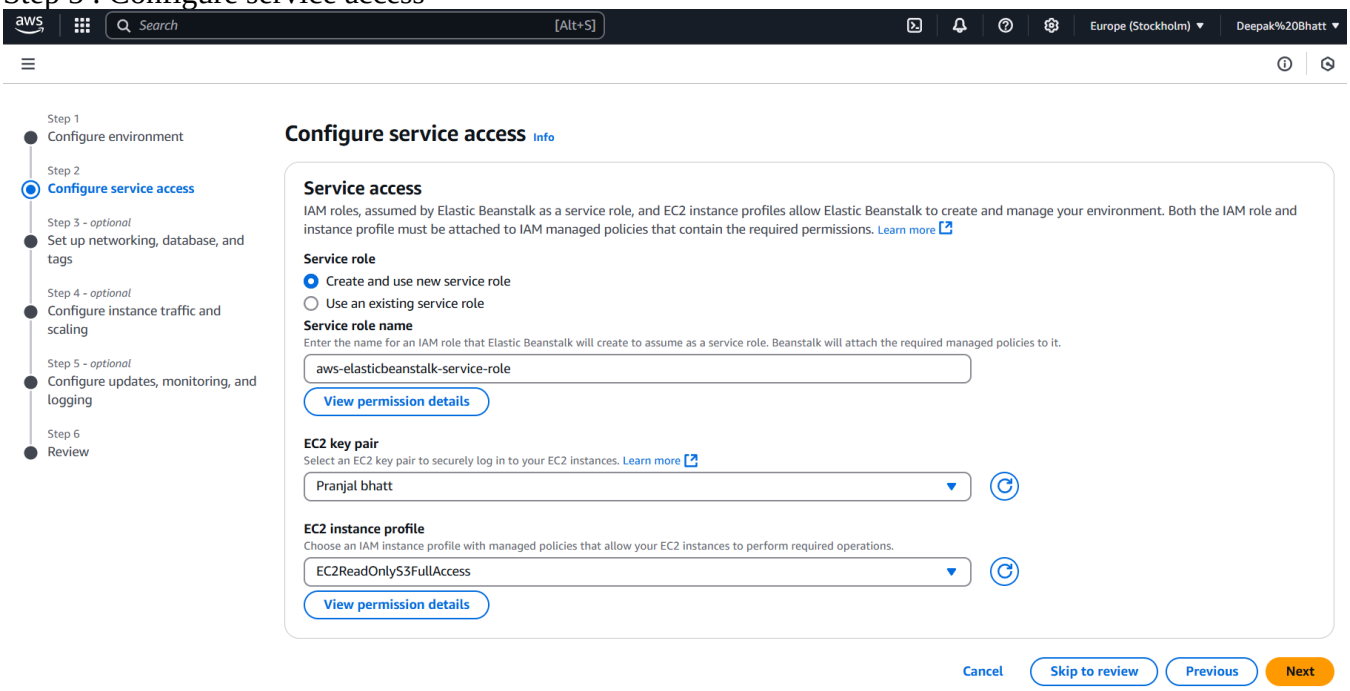


Step 4: Provide Application Information(Choose Environment tier,Application Name,Tags,Platform)

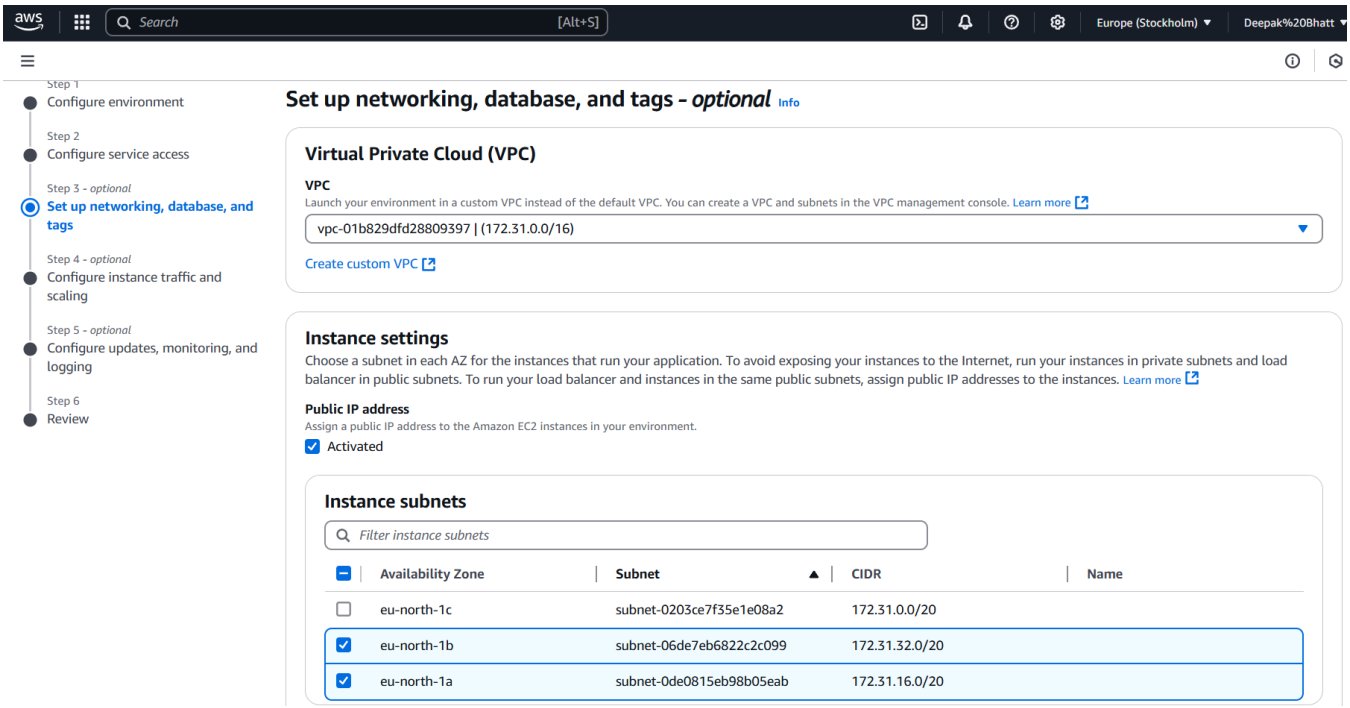




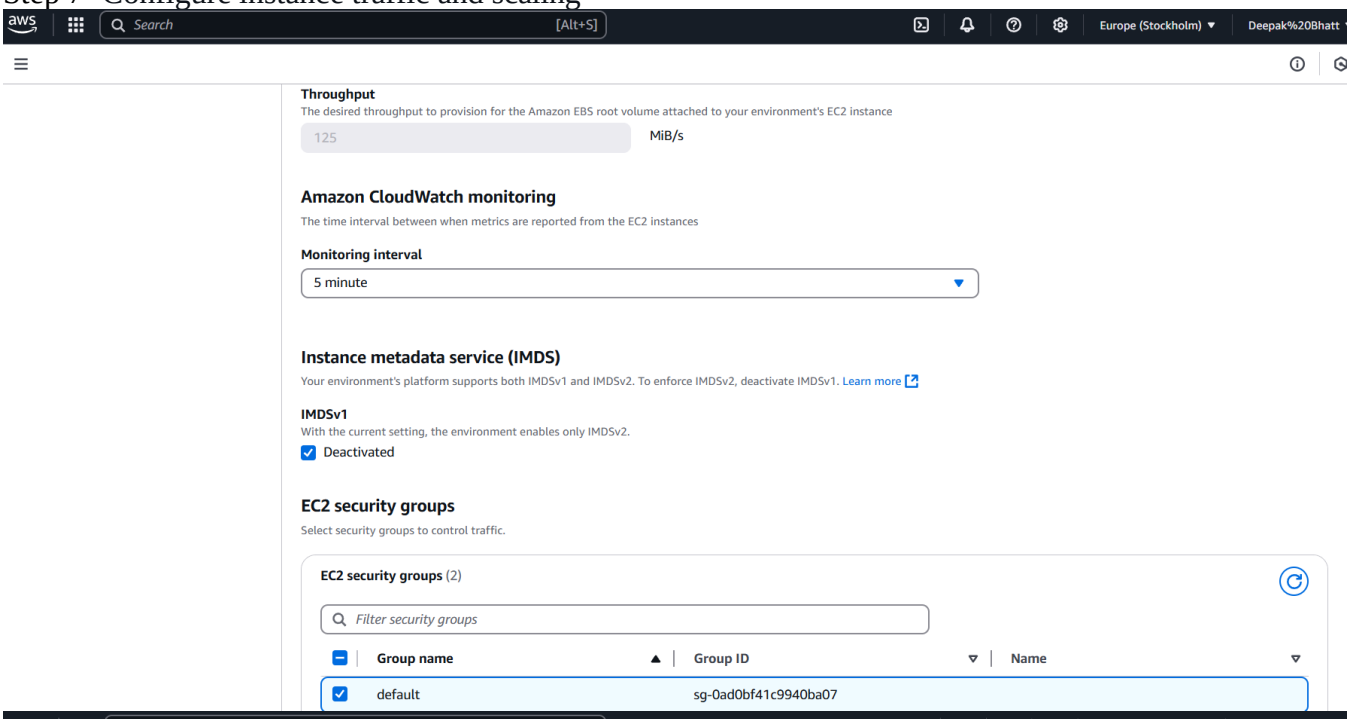
Step 5 : Configure service access

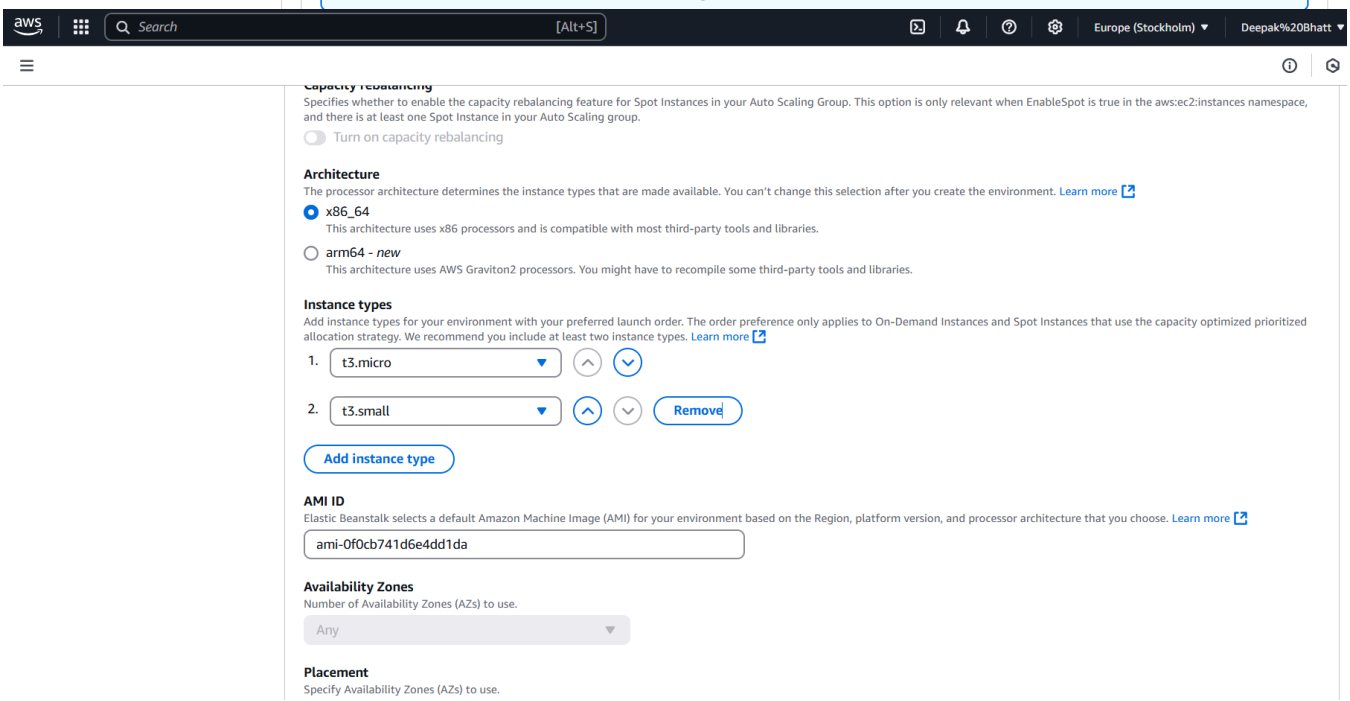


Step 6: Set up networking, database, and tags

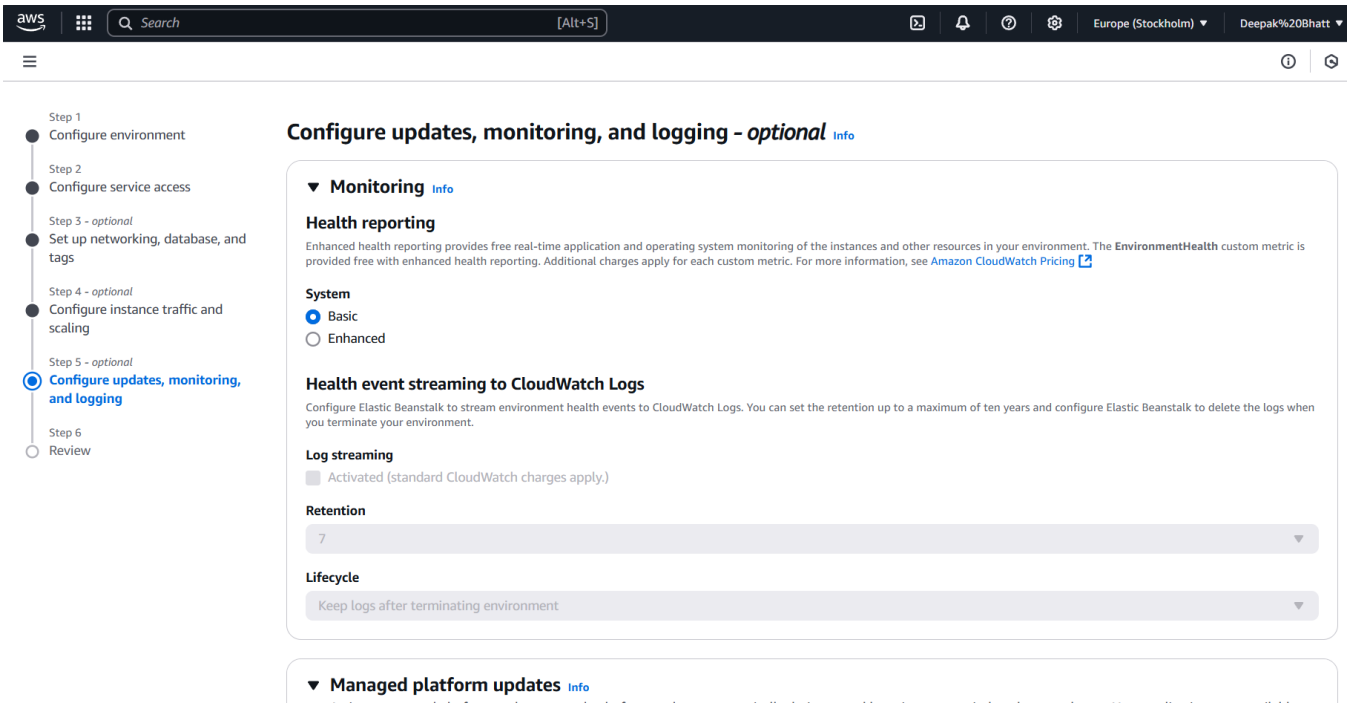


Step 7- Configure instance traffic and scaling

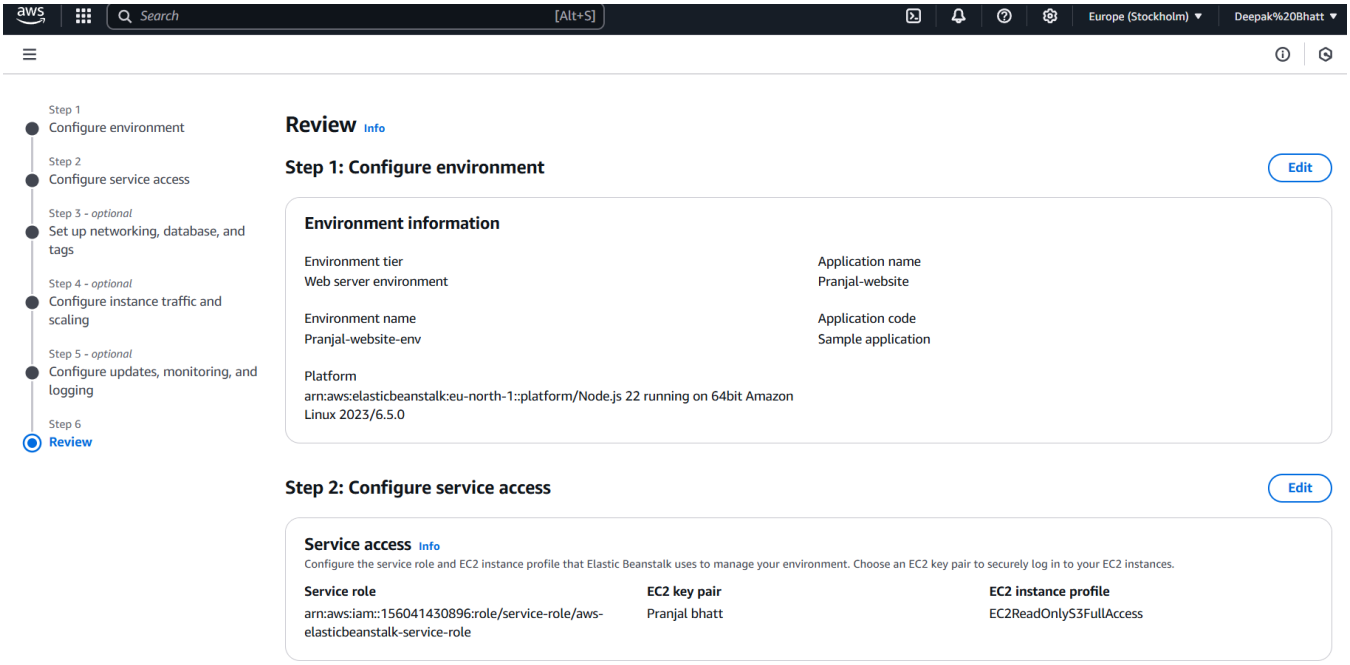




Step 8 - Configure updates, monitoring, and logging



Step 9: Review



0

aws | Search [Alt+S] | Europe (Stockholm) ▼ | Deepak%20H

AWS Console Home

Step 3: Set up networking, database, and tags

Edit

Networking, database, and tags Info

Configure VPC settings, and subnets for your environment's EC2 instances and load balancer. Set up an Amazon RDS database that's integrated with your environment.

Network

VPC	Public IP address	Instance subnets
vpc-01b829dfd28809397	true	subnet-06de7eb6822c2c099,subnet-0de0815eb98b05eab

Tags

Key	Value
No tags There are no tags defined	

Step 4: Configure instance traffic and scaling

Edit

Instance traffic and scaling Info

Customize the capacity and scaling for your environment's instances. Select security groups to control instance traffic.Configure the software that runs on your environment's instances by setting platform-specific options.

Instances

IMDSv1	EC2 Security Groups	
Deactivated	sg-0ad0bf41c9940ba07	

Capacity

Environment type	Fleet composition	On-demand base
Single instance	On-Demand instance	0
On-demand above base	**Capacity rebalancing**	**Scaling cooldown**
0	Deactivated	360
Processor type	**Instance types**	**AMI ID**
x86_64	t3.micro,t3.small	ami-0f0cb741d6e4dd1da

Step 5: Configure updates, monitoring, and logging

Edit

Updates, monitoring, and logging Info

Define when and how Elastic Beanstalk deploys changes to your environment. Manage your application's monitoring and logging settings, instances, and other environment resources.

Monitoring

System	Cloudwatch custom metrics - instance	Cloudwatch custom metrics - environment
basic	—	—
Log streaming	**Retention**	**Lifecycle**
Deactivated	7	false

Updates

Managed updates	Deployment batch size	Deployment batch size type
Deactivated	100	Percentage
Command timeout	**Deployment policy**	**Health threshold**
600	AllAtOnce	Ok
Ignore health check	**Instance replacement**	
false	false	

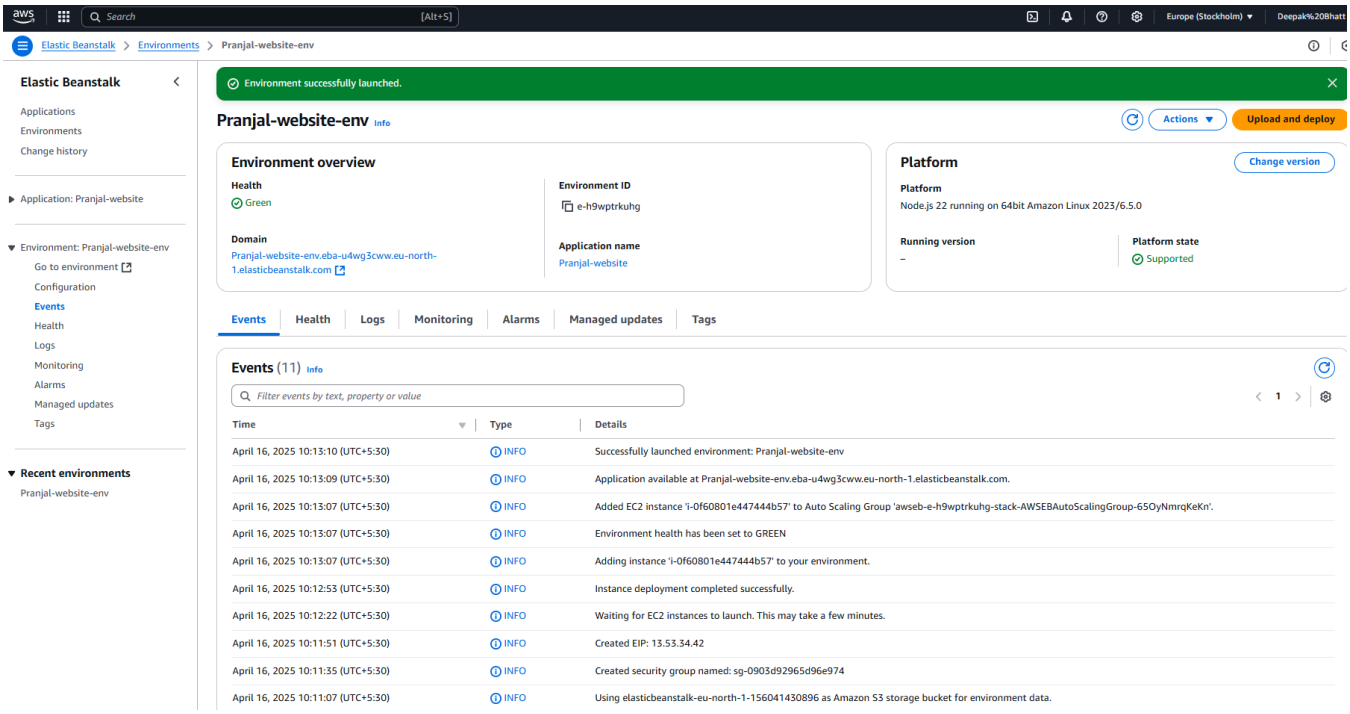
Platform software

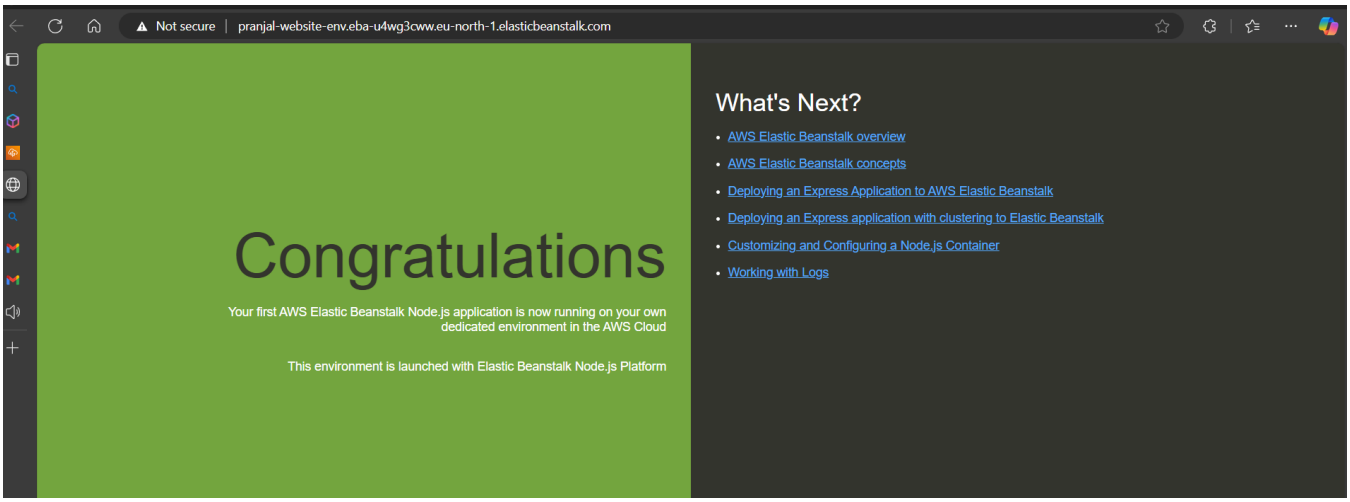
Lifecycle	Log streaming	Proxy server
false	Deactivated	nginx
Logs retention	**Rotate logs**	**Update level**
7	Deactivated	minor
X-Ray enabled		
Deactivated		

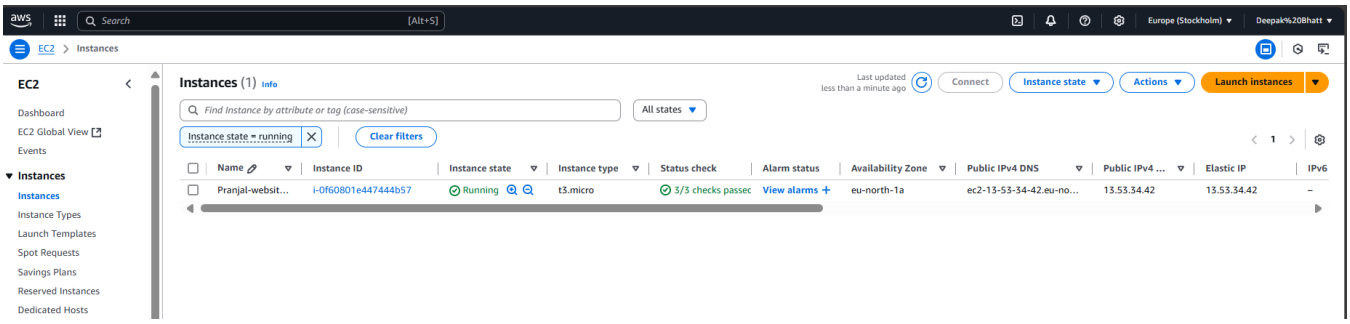
Environment properties

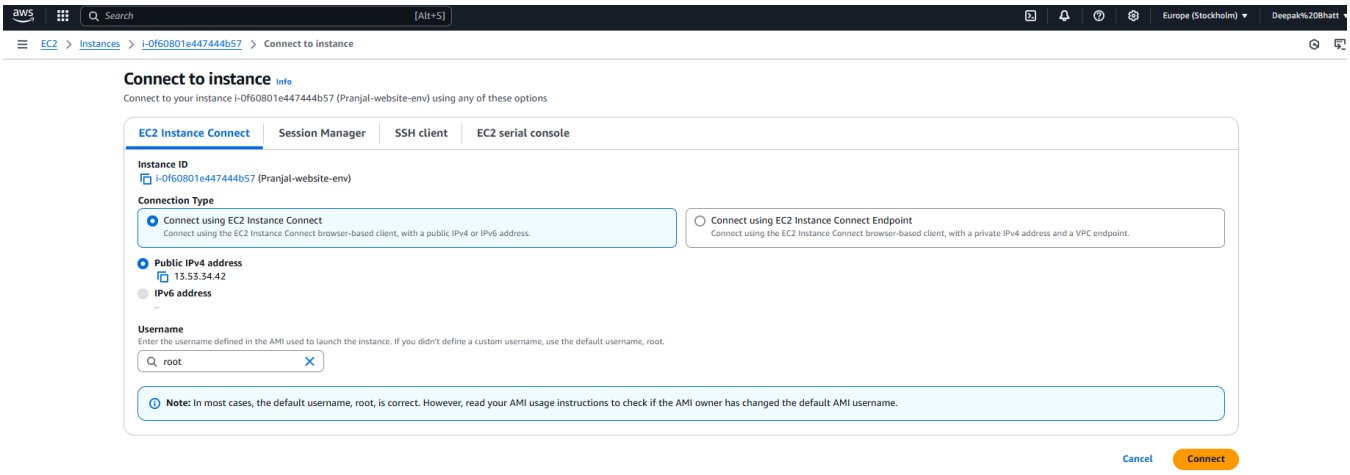
Source	Key	Value
No environment properties There are no environment properties defined		

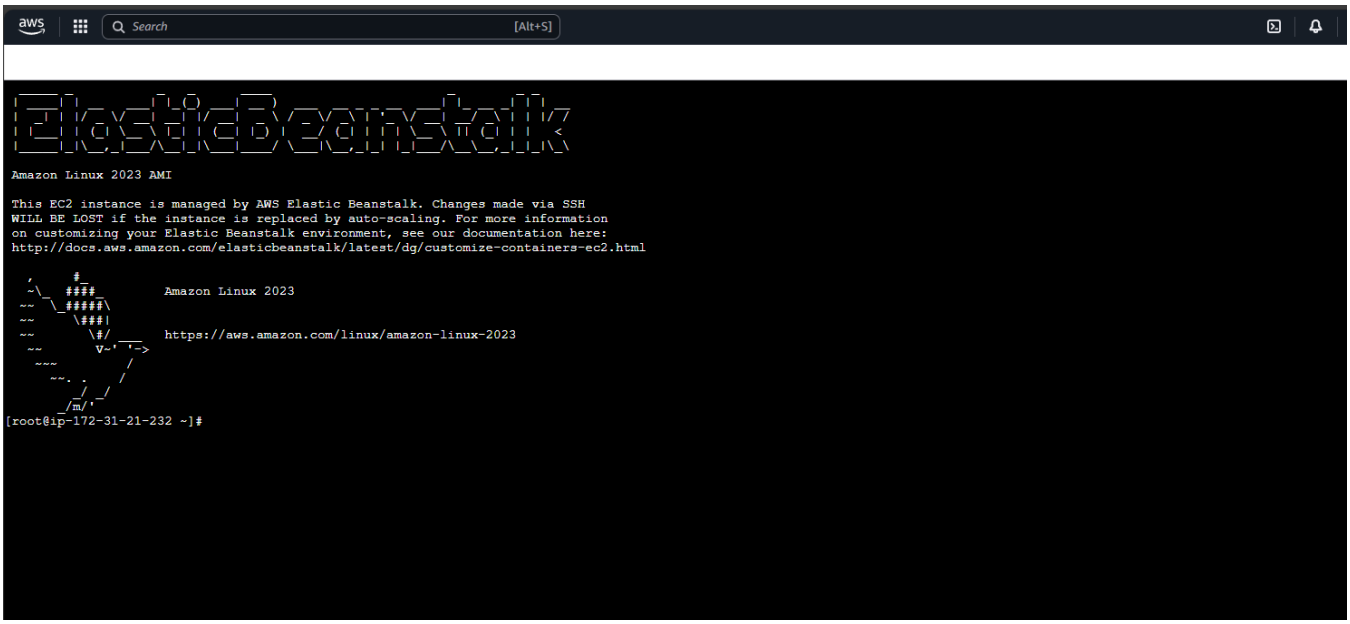
Step 10: Click the URL



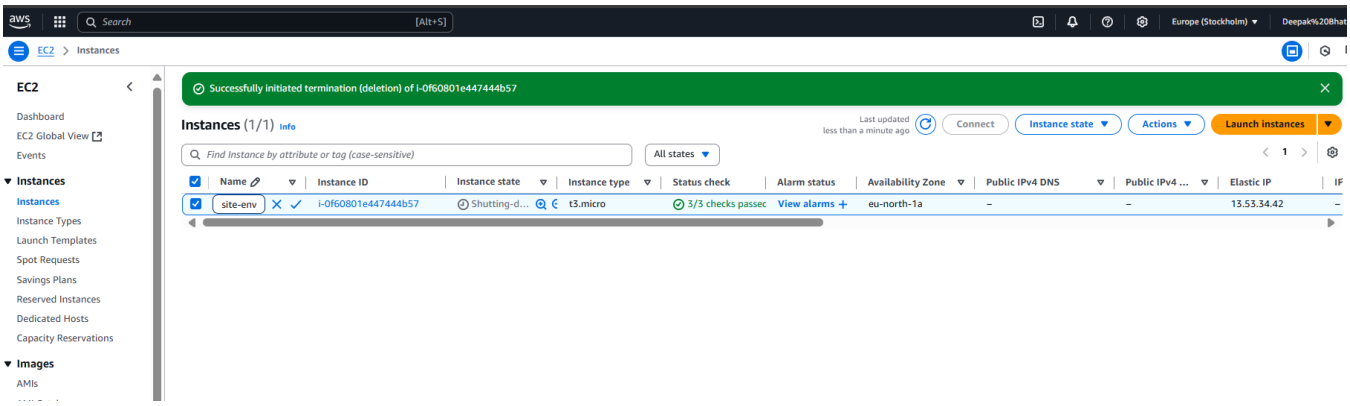


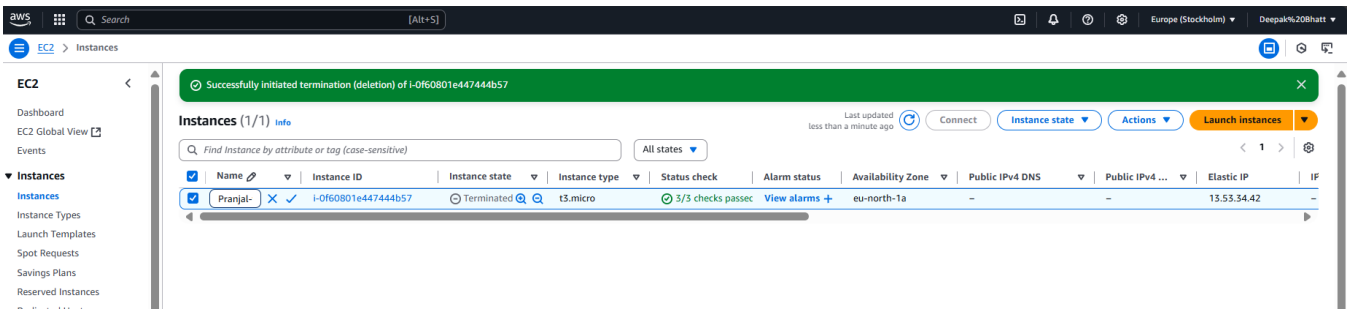

aws
Search
[Alt+S]
Europe (Stockholm)
Deepak%20Bhatt
EC2 > Instances
EC2
Dashboard
EC2 Global View
Events
Instances
Instances
Instance Types
Launch Templates
Spot Requests
Savings Plans
Reserved Instances
Dedicated Hosts
Instances (1) Info
Last updated less than a minute ago
Connect
Instance state
Actions
Launch instances
Find Instance by attribute or tag (case-sensitive)
All states
Instance state = running
Clear filters
Name
Instance ID
Instance state
Instance type
Status check
Alarm status
Availability Zone
Public IPv4 DNS
Public IPv4 ...
Elastic IP
IPv6
Pranjal-websit...
i-0f60801e447444b57
Running
t3.micro
3/3 checks passed
View alarms +
eu-north-1a
ec2-13-53-34-42.eu-no...
13.53.34.42
13.53.34.42
–


aws
Search
[Alt+S]
Europe (Stockholm)
Deepak%20Bhatt
EC2 > Instances > i-0f60801e447444b57 > Connect to instance
Connect to instance Info
Connect to your instance i-0f60801e447444b57 (Pranjal-website-env) using any of these options
EC2 Instance Connect
Session Manager
SSH client
EC2 serial console
Instance ID
i-0f60801e447444b57 (Pranjal-website-env)
Connection Type
Connect using EC2 Instance Connect
Connect using the EC2 Instance Connect browser-based client, with a public IPv4 or IPv6 address.
Connect using EC2 Instance Connect Endpoint
Connect using the EC2 Instance Connect browser-based client, with a private IPv4 address and a VPC endpoint.
Public IPv4 address
13.53.34.42
IPv6 address
–
Username
Enter the username defined in the AMI used to launch the instance. If you didn't define a custom username, use the default username, root.
root
Note: In most cases, the default username, root, is correct. However, read your AMI usage instructions to check if the AMI owner has changed the default AMI username.
Cancel
Connect


aws
Search
[Alt+S]
Amazon Linux 2023 AMI
This EC2 instance is managed by AWS Elastic Beanstalk. Changes made via SSH
WILL BE LOST if the instance is replaced by auto-scaling. For more information
on customizing your Elastic Beanstalk environment, see our documentation here:
http://docs.aws.amazon.com/elasticbeanstalk/latest/dg/customize-containers-ec2.html
Amazon Linux 2023
https://aws.amazon.com/linux/amazon-linux-2023
[root@ip-172-31-21-232 ~]#

Step 11:Terminate the Environment





B.1Question of Curiosity:

Q1: What are the benefits of Paas in cloud computing?

Platform as a Service (PaaS) offers several benefits in cloud computing:

1. **Faster Development & Deployment** – Developers can quickly build, test, and deploy applications without managing infrastructure.
2. **Scalability** – Automatically scales resources based on demand.
3. **Cost-Efficient** – Reduces operational costs by eliminating hardware and infrastructure maintenance.
4. **Built-in Security** – Provides automatic updates, security patches, and compliance features.
5. **Multi-Platform Support** – Supports different programming languages, frameworks, and databases.
6. **Automatic Load Balancing** – Ensures high availability and performance.
7. **Easy Integration** – Seamlessly connects with databases, APIs, and third-party services.
8. **Collaboration & Accessibility** – Enables remote teams to collaborate and access cloud resources from anywhere.

Q2:What is Paas?

Platform as a Service (PaaS) is a cloud computing model where a cloud provider offers a complete development and deployment environment without requiring users to manage the underlying infrastructure.

Key Features of PaaS:

- Provides runtime environments, databases, and development tools.

- Manages servers, networking, and storage for developers.
- Examples: AWS Elastic Beanstalk, Google App Engine, Microsoft Azure App Service.

Use Case:
A developer can deploy a web application on AWS Elastic Beanstalk without worrying about setting up servers, load balancers, or networking.

Q3) what is elastic bean stack ?

AWS Elastic Beanstalk is a Platform as a Service (PaaS) offered by AWS that automates the deployment, scaling, and management of applications.

Key Features:

- Supports multiple languages: Node.js, Python, Java, .NET, PHP, Ruby, Go.
- Automatically provisions EC2 instances, load balancers, databases, and other AWS services.
- Provides a fully managed environment for deploying web applications and APIs.

Allows developers to focus on writing code while AWS handles infrastructure management.

Example Use Case:
A company can deploy a Node.js application on Elastic Beanstalk, and AWS will handle scaling, monitoring, and load balancing automatically.

B.2 Conclusion:

Through this hands-on implementation of Platform as a Service (PaaS) using AWS Elastic Beanstalk, I have gained practical knowledge about deploying, managing, and scaling web applications in a cloud environment. This experiment helped me understand how PaaS simplifies application deployment by handling infrastructure, scaling, and resource management.

I successfully deployed a sample web application, monitored its health, and observed how AWS Elastic Beanstalk automatically manages load balancing, scaling, and provisioning of instances. Additionally, I explored how Elastic Beanstalk integrates with other AWS services like EC2, S3, RDS, and CloudWatch, ensuring seamless cloud operations. The experiment also highlighted how PaaS improves cost efficiency by eliminating the need for manual server management, making application development more efficient and scalable. Overall, this learning experience reinforced my understanding of cloud computing models and how PaaS accelerates deployment while reducing infrastructure complexities.