

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

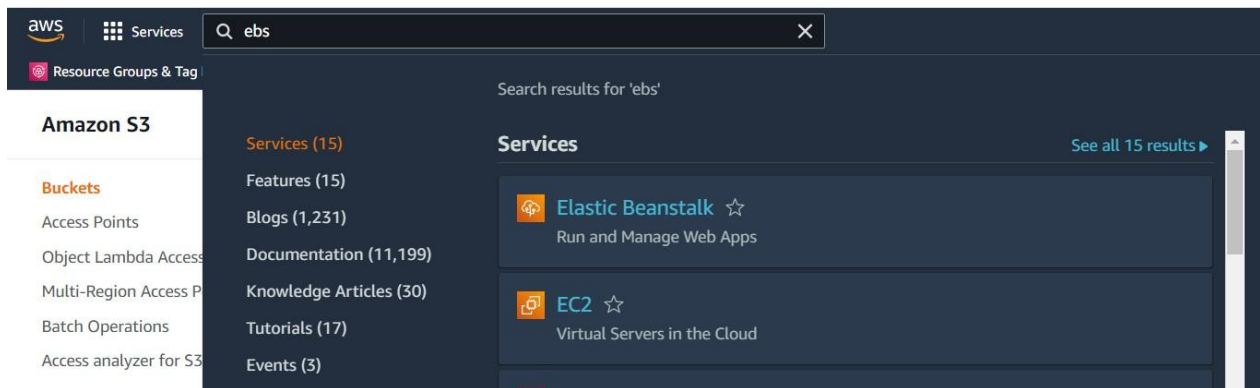
the elements of their infrastructure, you can do so seamlessly by using Elastic Beanstalk's management capabilities.

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.

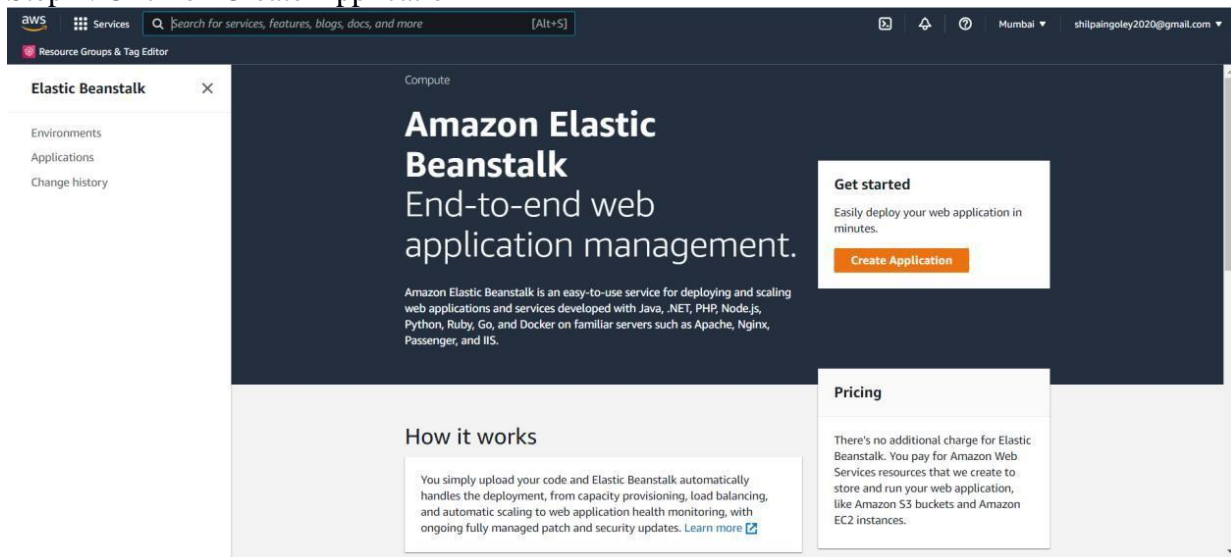
Following are steps to host a website using Elastic Beanstalk:

(Creating Sample Application)

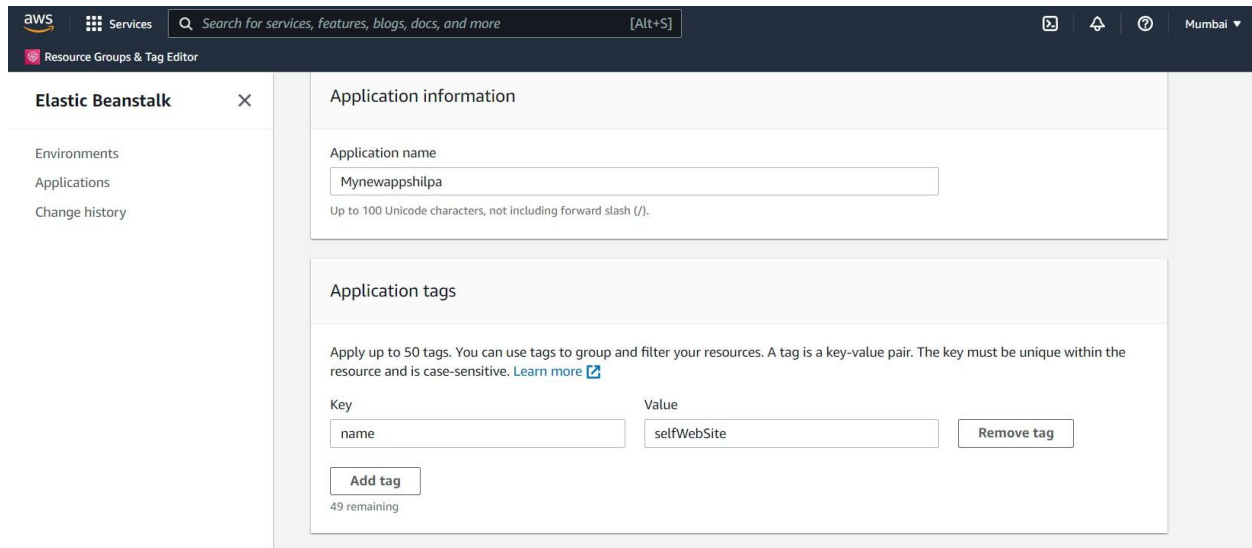
Step1 : Login to AWS console and go to Elastic Beanstalk



Step 2: Click on Create Application



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



aws Services Search for services, features, blogs, docs, and more [Alt+S] Mumbai

Resource Groups & Tag Editor

Elastic Beanstalk ✕

Environments
Applications
Change history

Application information

Application name
Mynewappshilpa
Up to 100 Unicode characters, not including forward slash (/).

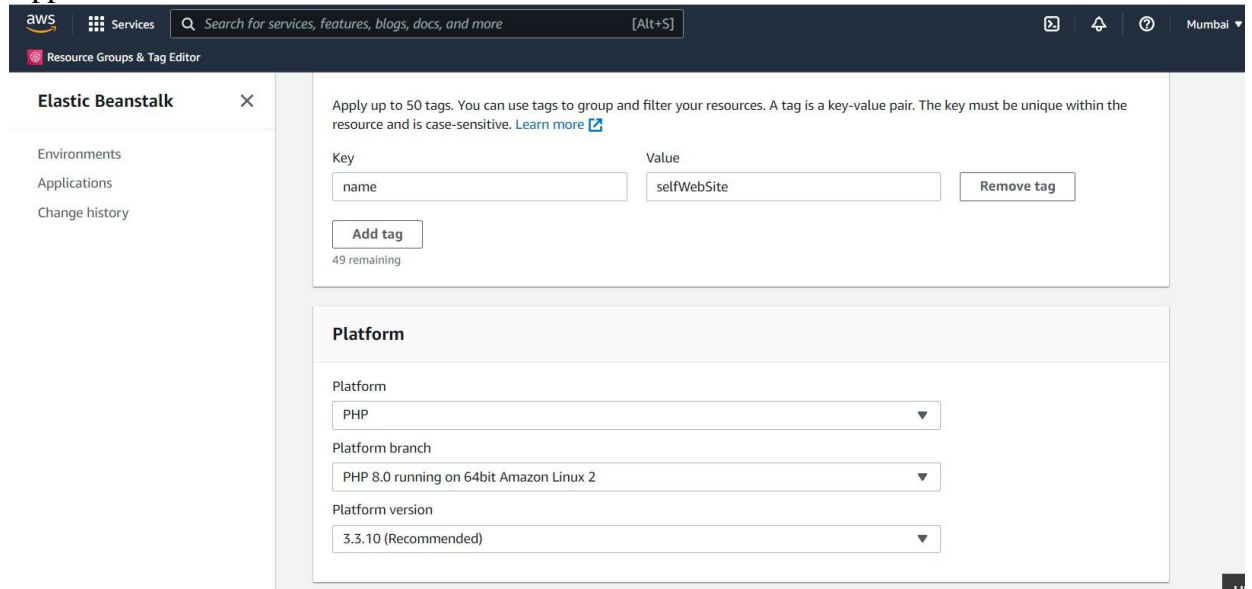
Application tags

Apply up to 50 tags. You can use tags to group and filter your resources. A tag is a key-value pair. The key must be unique within the resource and is case-sensitive. [Learn more](#)

Key	Value	
name	selfWebSite	Remove tag

Add tag
49 remaining

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



aws Services Search for services, features, blogs, docs, and more [Alt+S] Mumbai

Resource Groups & Tag Editor

Elastic Beanstalk ✕

Environments
Applications
Change history

Apply up to 50 tags. You can use tags to group and filter your resources. A tag is a key-value pair. The key must be unique within the resource and is case-sensitive. [Learn more](#)

Key	Value	
name	selfWebSite	Remove tag

Add tag
49 remaining

Platform

Platform
PHP

Platform branch
PHP 8.0 running on 64bit Amazon Linux 2

Platform version
3.3.10 (Recommended)

This will take a few minutes.

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

Application code

☒ Sample application
Get started right away with sample code.

☐ Upload your code
Upload a source bundle from your computer or copy one from Amazon S3.

Cancel

Configure more options

Create application

AWS

Services

Search for services, features, blogs, docs, and more

[Alt+S]

Resource Groups & Tag Editor

Elastic Beanstalk

Environments

Applications

Change history

▼ apptestshilpa

Application versions

Saved configurations

▶ Apptestshilpa-env

Elastic Beanstalk > Environments > Apptestshilpa-env

Creating Apptestshilpa-env

This will take a few minutes. .

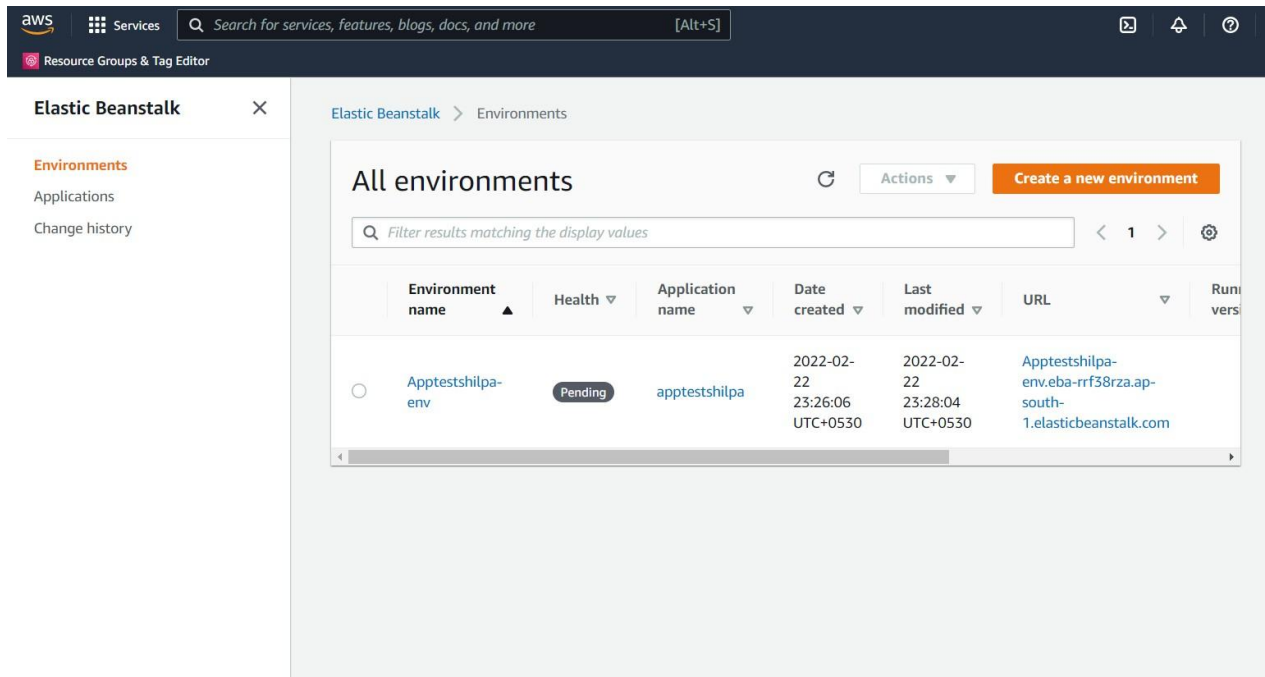
11:26pm Created security group named:
sg-0eb4e480238fb8826

11:26pm Created target group named:
arn:aws:elasticloadbalancing:ap-south-1:580032287469:targetgroup/awseb-AWSEB-ZN1T36EQ9LKB/790b52a1d8640215

11:26pm Using elasticbeanstalk-ap-south-1-580032287469 as Amazon S3 storage bucket for environment data.

11:26pm createEnvironment is starting.

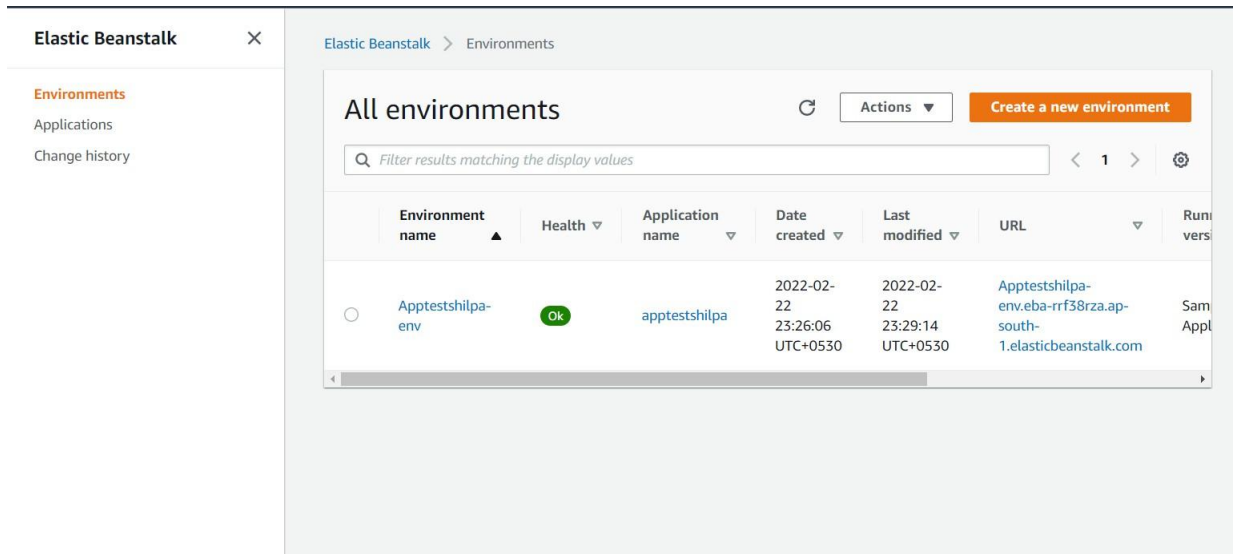
Step 6: Click the URL



The screenshot shows the AWS Elastic Beanstalk console. The left sidebar has 'Elastic Beanstalk' selected. The main area is titled 'All environments'. A table lists the environments:

Environment name	Health	Application name	Date created	Last modified	URL	Run version
Apptestshilpa-env	Pending	apptestshilpa	2022-02-22 23:26:06 UTC+0530	2022-02-22 23:28:04 UTC+0530	Apptestshilpa-env.eba-rrf38rza.ap-south-1.elasticbeanstalk.com	

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



The screenshot shows the AWS Elastic Beanstalk console. The left sidebar has 'Elastic Beanstalk' selected. The main area is titled 'All environments'. A table lists the environments:

Environment name	Health	Application name	Date created	Last modified	URL	Run version
Apptestshilpa-env	Ok	apptestshilpa	2022-02-22 23:26:06 UTC+0530	2022-02-22 23:29:14 UTC+0530	Apptestshilpa-env.eba-rrf38rza.ap-south-1.elasticbeanstalk.com	Sam Appl

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.

aws Search [Alt+S] Europe (Stockholm) Deepak%20Bhatt

EC2 > Key pairs > Create key pair

Create key pair [Info](#)

Key pair
A key pair, consisting of a private key and a public key, is a set of security credentials that you use to prove your identity when connecting to an instance.

Name
Pranjal bhatt
The name can include up to 255 ASCII characters. It can't include leading or trailing spaces.

Key pair type [Info](#)
☒ RSA ☐ ED25519

Private key file format
☐ .pem
For use with OpenSSH
☒ .ppk
For use with PuTTY

Tags - optional

Key	Value - optional	
Environment	Production	Remove

[Add new tag](#)
You can add up to 49 more tags.

[Cancel](#) [Create key pair](#)

Step 2: Create Roles For EC2:

aws

Search

[Alt+S]

Global

Deepak%20Bhatt

IAM

Roles

Create role

Step 1

Select trusted entity

Step 2

Add permissions

Step 3

Name, review, and create

Select trusted entity Info

Trusted entity type

☒ **AWS service**
Allow AWS services like EC2, Lambda, or others to perform actions in this account.

☐ **AWS account**
Allow entities in other AWS accounts belonging to you or a 3rd party to perform actions in this account.

☐ **Web identity**
Allows users federated by the specified external web identity provider to assume this role to perform actions in this account.

☐ **SAML 2.0 federation**
Allow users federated with SAML 2.0 from a corporate directory to perform actions in this account.

☐ **Custom trust policy**
Create a custom trust policy to enable others to perform actions in this account.

Use case

Allow an AWS service like EC2, Lambda, or others to perform actions in this account.

Service or use case

EC2

Choose a use case for the specified service.

Use case

☒ **EC2**
Allows EC2 instances to call AWS services on your behalf.

☐ **EC2 Role for AWS Systems Manager**
Allows EC2 instances to call AWS services like CloudWatch and Systems Manager on your behalf.

Step 1

Select trusted entity

Step 2

Add permissions

Step 3

Name, review, and create

Name, review, and create

Role details

Role name
Enter a meaningful name to identify this role.

Maximum 64 characters. Use alphanumeric and '+=,._@-_' characters.

Description
Add a short explanation for this role.

Maximum 1000 characters. Use letters (A-Z and a-z), numbers (0-9), tabs, new lines, or any of the following characters: _+=,._@-/\[\]\#\\$\%^&*~''

Step 1: Select trusted entities

Edit

Trust policy

```
1 {  
2   "Version": "2012-10-17",  
3   "Statement": [  
4     {  
5       "Effect": "Allow",  
6       "Action": [  
7         "sts:AssumeRole"  
8       ],  
9       "Principal": {  
10        "Service": [  
11          "ec2.amazonaws.com"  
12        ]  
13      }  
14    ]  
15  }  
16 }
```


aws

Search

[Alt+S]

Global

Deepak%20Bhatt

Identity and Access Management (IAM)

Search IAM

Dashboard

Access management

Users groups

Users

Roles

Policies

Identity providers

Account settings

Root access management

Access reports

Access Analyzer

Archive rules

Analizers

Settings

Credential report

Organization activity

Service control policies

Resource control policies

EC2ReadOnlyS3FullAccess

Info

Delete

Allows EC2 instances to call AWS services on your behalf.

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

Simulate

Remove

Add permissions

You can attach up to 10 managed policies.

Search



Filter by Type

All types





< 1 >

	Policy name	Type	Attached entities
<input type="checkbox"/>	AmazonEC2ReadOnlyAccess	AWS managed	1
<input type="checkbox"/>	AmazonS3FullAccess	AWS managed	1

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



[Alt+S]

Europe (Stockholm) ▾Deepak%20Bhatt ▾

≡

⌵

Compute

Amazon Elastic Beanstalk

End-to-end web application management.

Amazon Elastic Beanstalk is an easy-to-use service for deploying and scaling web applications and services developed with Java, .NET, PHP, Node.js, Python, Ruby, Go, and Docker on familiar servers such as Apache, Nginx, Passenger, and IIS.

Get started

Easily deploy your web application in minutes.

Create application

Pricing

There's no additional charge for Elastic Beanstalk. You pay for Amazon Web Services resources that we create to store and run your web application, like Amazon S3 buckets and Amazon EC2 instances.

Get started

You simply upload your code and Elastic Beanstalk automatically handles the deployment, from capacity provisioning, load balancing, and automatic scaling to web application health monitoring, with ongoing fully managed patch and security updates. [Learn more](#)

Getting started

[Launch a web application](#)

Benefits and features

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

aws

Search

[Alt+S]

Europe (Stockholm)

Deepak%20Bhatt

Step 1

Configure environment

Step 2

Configure service access

Step 3 - optional

Set up networking, database, and tags

Step 4 - optional

Configure instance traffic and scaling

Step 5 - optional

Configure updates, monitoring, and logging

Step 6

Review

Configure environment

Environment tier

Amazon Elastic Beanstalk has two types of environment tiers to support different types of web applications.

☒ Web server environment

Run a website, web application, or web API that serves HTTP requests. [Learn more](#)

☐ Worker environment

Run a worker application that processes long-running workloads on demand or performs tasks on a schedule. [Learn more](#)

Application information

Application name

Pranjal-website

Maximum length of 100 characters.

Application tags (optional)

Environment information

Choose the name, subdomain and description for your environment. These cannot be changed later.

Environment name

Pranjal-website-env

Must be from 4 to 40 characters in length. The name can contain only letters, numbers, and hyphens. It can't start or end with a hyphen. This name must be unique within a region in your account.

aws

Search

[Alt+S]

Europe (Stockholm)

Deepak%20Bhatt

Platform

Platform type

☒ Managed platform

Platforms published and maintained by Amazon Elastic Beanstalk. [Learn more](#)

☐ Custom platform

Platforms created and owned by you. This option is unavailable if you have no platforms.

Platform

Node.js

Platform branch

Node.js 22 running on 64bit Amazon Linux 2023

Platform version

6.5.0 (Recommended)

Application code

☒ Sample application

☐ Existing version

Application versions that you have uploaded.

☐ Upload your code

Upload a source bundle from your computer or copy one from Amazon S3.

Step 5 : Configure service access

Step 1

Configure environment

Step 2

Configure service access

Step 3 - optional

Set up networking, database, and tags

Step 4 - optional

Configure instance traffic and scaling

Step 5 - optional

Configure updates, monitoring, and logging

Step 6

Review

Configure service access [Info](#)

Service access

IAM roles, assumed by Elastic Beanstalk as a service role, and EC2 instance profiles allow Elastic Beanstalk to create and manage your environment. Both the IAM role and instance profile must be attached to IAM managed policies that contain the required permissions. [Learn more](#)

Service role

☒ Create and use new service role
☐ Use an existing service role

Service role name

Enter the name for an IAM role that Elastic Beanstalk will create to assume as a service role. Beanstalk will attach the required managed policies to it.

[View permission details](#)

EC2 key pair

Select an EC2 key pair to securely log in to your EC2 instances. [Learn more](#)

EC2 instance profile

Choose an IAM instance profile with managed policies that allow your EC2 instances to perform required operations.

[View permission details](#)

Cancel

[Skip to review](#)

[Previous](#)

[Next](#)

Step 6: Set up networking, database, and tags

Step 1

Configure environment

Step 2

Configure service access

Step 3 - optional

Set up networking, database, and tags

Step 4 - optional

Configure instance traffic and scaling

Step 5 - optional

Configure updates, monitoring, and logging

Step 6

Review

Set up networking, database, and tags - optional [Info](#)

Virtual Private Cloud (VPC)

VPC
Launch your environment in a custom VPC instead of the default VPC. You can create a VPC and subnets in the VPC management console. [Learn more](#)

[Create custom VPC](#)

Instance settings

Choose a subnet in each AZ for the instances that run your application. To avoid exposing your instances to the Internet, run your instances in private subnets and load balancer in public subnets. To run your load balancer and instances in the same public subnets, assign public IP addresses to the instances. [Learn more](#)

Public IP address

Assign a public IP address to the Amazon EC2 instances in your environment.

☒ Activated

Instance subnets

	Availability Zone	Subnet	CIDR	Name
<input type="checkbox"/>	eu-north-1c	subnet-0203ce7f35e1e08a2	172.31.0.0/20	
<input checked="" type="checkbox"/>	eu-north-1b	subnet-06de7eb6822c2c099	172.31.32.0/20	
<input checked="" type="checkbox"/>	eu-north-1a	subnet-0de0815eb98b05eab	172.31.16.0/20	

Step 7- Configure instance traffic and scaling

aws

Search

[Alt+S]

Europe (Stockholm)

Deepak%20Bhatt

Throughput

The desired throughput to provision for the Amazon EBS root volume attached to your environment's EC2 instance

125

MiB/s

Amazon CloudWatch monitoring

The time interval between when metrics are reported from the EC2 instances

Monitoring interval

5 minute

Instance metadata service (IMDS)

Your environment's platform supports both IMDSv1 and IMDSv2. To enforce IMDSv2, deactivate IMDSv1. [Learn more](#)

IMDSv1

With the current setting, the environment enables only IMDSv2.

☒ Deactivated

EC2 security groups

Select security groups to control traffic.

EC2 security groups (2)

Filter security groups

☒

default

sg-0ad0bf41c9940ba07

aws

Search

[Alt+S]

Europe (Stockholm)

Deepak%20Bhatt

Capacity rebalancing

Specifies whether to enable the capacity rebalancing feature for Spot Instances in your Auto Scaling Group. This option is only relevant when EnableSpot is true in the aws:ec2:instances namespace, and there is at least one Spot Instance in your Auto Scaling group.

☐ Turn on capacity rebalancing

Architecture

The processor architecture determines the instance types that are made available. You can't change this selection after you create the environment. [Learn more](#)

☒ x86_64

This architecture uses x86 processors and is compatible with most third-party tools and libraries.

☐ arm64 - new

This architecture uses AWS Graviton2 processors. You might have to recompile some third-party tools and libraries.

Instance types

Add instance types for your environment with your preferred launch order. The order preference only applies to On-Demand Instances and Spot Instances that use the capacity optimized prioritized allocation strategy. We recommend you include at least two instance types. [Learn more](#)

1. t3.micro

2. t3.small

Add instance type

AMI ID

Elastic Beanstalk selects a default Amazon Machine Image (AMI) for your environment based on the Region, platform version, and processor architecture that you choose. [Learn more](#)

ami-0f0cb741d6e4dd1da

Availability Zones

Number of Availability Zones (AZs) to use.

Any

Placement

Specify Availability Zones (AZs) to use.

Step 8 - Configure updates, monitoring, and logging

aws

Search

[Alt+S]

Europe (Stockholm)

Deepak%20Bhatt

Step 1
Configure environment

Step 2
Configure service access

Step 3 - optional
Set up networking, database, and tags

Step 4 - optional
Configure instance traffic and scaling

Step 5 - optional
Configure updates, monitoring, and logging

Step 6
Review

Configure updates, monitoring, and logging - optional

Monitoring

Health reporting

System

Health event streaming to CloudWatch Logs

Log streaming

Retention

Lifecycle

Managed platform updates

Step 9: Review

aws

Search

[Alt+S]

Europe (Stockholm)

Deepak%20Bhatt

Step 1
Configure environment

Step 2
Configure service access

Step 3 - optional
Set up networking, database, and tags

Step 4 - optional
Configure instance traffic and scaling

Step 5 - optional
Configure updates, monitoring, and logging

Step 6
Review

Review

Step 1: Configure environment

Environment information

Step 2: Configure service access

Service access

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

The screenshot shows the AWS Elastic Beanstalk console. At the top, a green banner states "Environment successfully launched." Below this, the "Pranjal-website-env" environment is detailed. The "Environment overview" section shows a "Health" status of "Green". The "Domain" is "Pranjal-website-env.eba-u4wg3cww.eu-north-1.elasticbeanstalk.com". The "Environment ID" is "e-h9wptrkuhg". The "Application name" is "Pranjal-website". The "Platform" section shows "Node.js 22 running on 64bit Amazon Linux 2023/6.5.0" with a "Platform state" of "Supported". Below these sections are tabs for "Events", "Health", "Logs", "Monitoring", "Alarms", "Managed updates", and "Tags". The "Events" tab is active, displaying a list of 11 events. The events include the successful launch of the environment, application availability, EC2 instance addition, health setting to GREEN, instance addition, deployment completion, waiting for EC2 instances, EIP creation, security group creation, and S3 bucket creation.

Environment overview

Health: ✔ Green

Environment ID: e-h9wptrkuhg

Domain: Pranjal-website-env.eba-u4wg3cww.eu-north-1.elasticbeanstalk.com

Application name: Pranjal-website

Platform

Platform: Node.js 22 running on 64bit Amazon Linux 2023/6.5.0

Running version: -

Platform state: ✔ Supported

Events (11)

Time	Type	Details
April 16, 2025 10:13:10 (UTC+5:30)	INFO	Successfully launched environment: Pranjal-website-env
April 16, 2025 10:13:09 (UTC+5:30)	INFO	Application available at Pranjal-website-env.eba-u4wg3cww.eu-north-1.elasticbeanstalk.com.
April 16, 2025 10:13:07 (UTC+5:30)	INFO	Added EC2 instance 'i-0f60801e447444b57' to Auto Scaling Group 'aws-ec2-elasticbeanstalk-aws-ec2-elasticbeanstalk-650yNmqrKer'.
April 16, 2025 10:13:07 (UTC+5:30)	INFO	Environment health has been set to GREEN
April 16, 2025 10:13:07 (UTC+5:30)	INFO	Adding instance 'i-0f60801e447444b57' to your environment.
April 16, 2025 10:12:53 (UTC+5:30)	INFO	Instance deployment completed successfully.
April 16, 2025 10:12:22 (UTC+5:30)	INFO	Waiting for EC2 instances to launch. This may take a few minutes.
April 16, 2025 10:11:51 (UTC+5:30)	INFO	Created EIP: 13.53.34.42
April 16, 2025 10:11:35 (UTC+5:30)	INFO	Created security group named: sg-0903d92965d96e974
April 16, 2025 10:11:07 (UTC+5:30)	INFO	Using elasticbeanstalk-eu-north-1-156041430896 as Amazon S3 storage bucket for environment data.

The screenshot shows a web browser window with the URL "pranjal-website-env.eba-u4wg3cww.eu-north-1.elasticbeanstalk.com". The page has a green header and a dark gray body. The main content area features a large "Congratulations" message in white text, followed by a smaller message: "Your first AWS Elastic Beanstalk Node.js application is now running on your own dedicated environment in the AWS Cloud". Below this, it states "This environment is launched with Elastic Beanstalk Node.js Platform". On the right side, there is a section titled "What's Next?" with a list of links: "AWS Elastic Beanstalk overview", "AWS Elastic Beanstalk concepts", "Deploying an Express Application to AWS Elastic Beanstalk", "Deploying an Express application with clustering to Elastic Beanstalk", "Customizing and Configuring a Node.js Container", and "Working with Logs".

Congratulations

Your first AWS Elastic Beanstalk Node.js application is now running on your own dedicated environment in the AWS Cloud

This environment is launched with Elastic Beanstalk Node.js Platform

What's Next?

- [AWS Elastic Beanstalk overview](#)
- [AWS Elastic Beanstalk concepts](#)
- [Deploying an Express Application to AWS Elastic Beanstalk](#)
- [Deploying an Express application with clustering to Elastic Beanstalk](#)
- [Customizing and Configuring a Node.js Container](#)
- [Working with Logs](#)

aws

Search

[Alt+S]

EC2

Instances

Instances (1) info

Find Instance by attribute or tag (case-sensitive)

All states

Instance state: running

Clear filters

Last updated less than a minute ago

Connect

Instance state

Actions

Launch instances

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	Public IPv4 ...	Elastic IP	IPv6
Pranjal-website...	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		

aws

Search

[Alt+S]

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]

Elastic Beanstalk

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

The image displays two sequential screenshots of the AWS Management Console, illustrating the process of terminating an EC2 instance.

Top Screenshot: The console shows the 'Instances' page with a notification at the top: 'Successfully initiated termination (deletion) of i-0f60801e447444b57'. The instance list contains one entry:

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	Public IPv4 ...	Elastic IP	IF
site-env	i-0f60801e447444b57	Shutting-d...	t3.micro	3/3 checks passed	View alarms +	eu-north-1a	-	-	13.53.34.42	-

Bottom Screenshot: The console shows the same 'Instances' page after the instance has been terminated. The notification at the top reads: 'Successfully initiated termination (deletion) of i-0f60801e447444b57'. The instance list now shows the instance in a 'Terminated' state:

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	Public IPv4 ...	Elastic IP	IF
Pranjal-	i-0f60801e447444b57	Terminated	t3.micro	3/3 checks passed	View alarms +	eu-north-1a	-	-	13.53.34.42	-

B.1 Question 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.