

Hello Friends today I will deploy a Java Sample Application in AWS Elastic Beanstalk. Lets start and get the basic understanding of Elastic Beanstalk and later we will deploy a sample application.

Deploy and scale web applications.

Elastic Beanstalk is a service for deploying and scaling web applications and services. Upload your code and Elastic Beanstalk automatically handles the deployment — from capacity provisioning, load balancing, and auto scaling to application health monitoring.

Elastic Beanstalk allows you to deploy and manage your application in Cloud with agility. You don't have to worry about the underlying infrastructure which is needed to run the application.

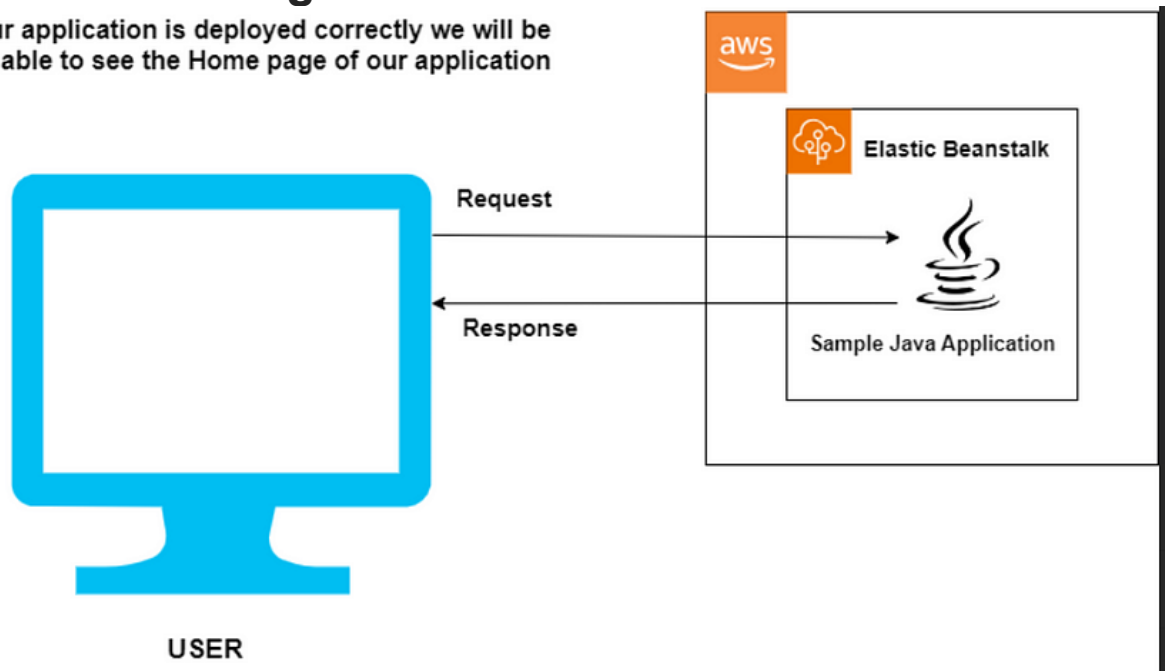
We can deploy the application in any of the following programming languages like Go, Java, .NET, Node.js, PHP, Python, and Ruby.

You can interact with Elastic Beanstalk app with CLI, Elastic Beanstalk console.

Given below is the Architecture Diagram of our Java Application.

Architecture Diagram

If our application is deployed correctly we will be able to see the Home page of our application



To create a Sample Application, follow the steps given below:

Step 1. Open [Elastic Beanstalk console](#).


Step 2. Choose **Create application**

i) Choose **Web Server environment** under Configure Environment

ii) Give name as **Sample-Application**

Environment tier [Info](#)

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

- ☒ **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

Application information [Info](#)

Application name

Maximum length of 100 characters.

► **Application tags (optional)**

iii) Under Platform choose **Java**

Platform [Info](#)

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

Java ▼

Platform branch

Corretto 17 running on 64bit Amazon Linux 2023 ▼

Platform version

4.0.1 (Recommended) ▼

iv) Scroll down and click **Next** button 

Step 3. In **Configure Service Access**

i) Service role: **Create and use new service role**

ii) Ec instance profile : **EC2_Role**

****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.**

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.

aws-elasticbeanstalk-service-role

[View permission details](#)

EC2 key pair

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

Choose a key pair



EC2 instance profile

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

EC2_Role



[View permission details](#)

Click on **Next** 

Step 4 . To set up **Networking, database and tags**


i) VPC : Select Default VPC

ii) Check the Public IP address checkbox .

iii) Check the us-east-1a and us-east-1b subnets checkbox under Instance subnets.

Keep the database and tags settings as default

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

 *Filter instance subnets*

	Availability Zone	Subnet 	CIDR	Name
<input type="checkbox"/>	us-east-1f	subnet-13bfbd1d	172.31.64.0/20	
<input checked="" type="checkbox"/>	us-east-1a	subnet-1b134744	172.31.32.0/20	
<input checked="" type="checkbox"/>	us-east-1b	subnet-3bf0b85d	172.31.0.0/20	

iv) Keep the rest things as default and click on **Next**

Step 5. To Configure instance traffic and scaling :


i) Scroll Down to capacity section and Under Instance Types, remove the existing instance types and select t2.micro from the drop-down.

Capacity rebalancing

Specifies whether to enable the capacity rebalancing feature for Spot Instances when EnableSpot is true in the aws:ec2:instances namespace, and there is no explicit value for the aws:ec2:instances namespace.

☐ Turn on capacity rebalancing

Architecture

The processor architecture determines the instance types that are made available in your environment. [Learn more](#) 

☒ x86_64

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

☐ arm64 - new

This architecture uses AWS Graviton2 processors. You might have to recompile tools and libraries.

Instance types

Add instance types for your fleet. Change the order that the instances are included in the fleet. We recommend you include at least two instance types for Demand instances.

Choose x86 instance types



t2.micro



Step 6. Uncheck the Activated under **Managed platform updates**

Keep the remaining settings as default in this page.

▼ Managed platform updates [Info](#)

Activate managed platform updates to apply platform updates automatically during a weekly update window. Your application stays available during the update process.

Managed updates

☐ Activated

Weekly update window


Tuesday ▼ at 22 ▼ : 21 ▼ UTC

Update level

Minor and patch


Step 7. The review page displays the summary of all your choices.

Step 8. Choose **Submit**, it will take few minutes to deploy the application and once done you will see the Environment successfully launched.

 Environment successfully launched.

[Elastic Beanstalk](#) > [Environments](#) > SampleApplication-env



SampleApplication-env [Info](#)



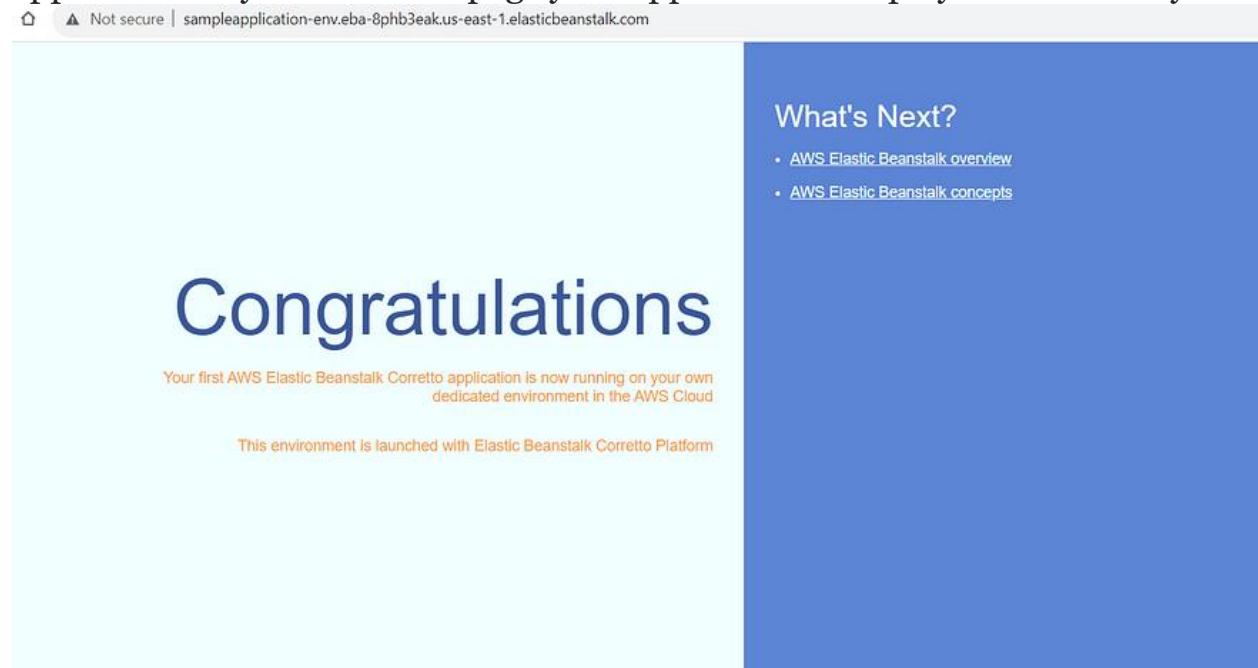
Actions ▼

Upload and deploy

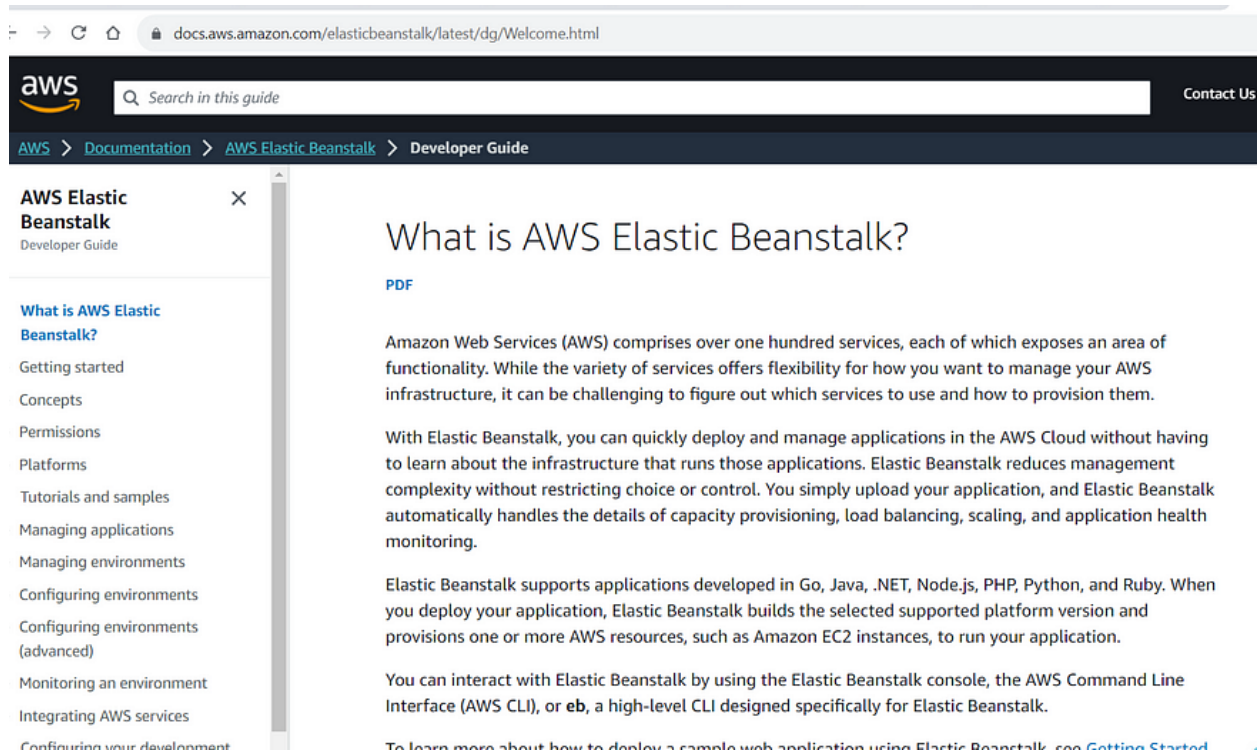
Environment overview

Health  Ok	Environment ID  e-qkw3ungnpp
Domain SampleApplication-env.eba-8phb3eak.us-east-1.elasticbeanstalk.com 🔗	Application name Sample_Application

You will see the Domain , copy it and paste in another browser to test your application. If you the below page your application is deployed successfully.



If you click on the link given the Right panel you will be redirected to AWS Elastic Beanstalk overview page.



Following resources are created when the Sample application is created in Elastic Beanstalk.

- **EC2 instance** — An Amazon EC2 virtual machine configured to run web apps on the platform you choose.
- Each platform runs a different set of software, configuration files, and scripts to support a specific language version, framework, web container, or combination thereof. Most platforms use either Apache or nginx as a reverse proxy that processes web traffic in front of your web app, forwards requests to it, serves static assets, and generates access and error logs.

- **Instance security group** — An Amazon EC2 security group configured to allow incoming traffic on port 80. This resource lets HTTP traffic from the load balancer reach the EC2 instance running your web app. By default, traffic is not allowed on other ports.
- **Amazon S3 bucket** — A storage location for your source code, logs, and other artifacts that are created when you use Elastic Beanstalk.
- **Amazon CloudWatch alarms** — Two CloudWatch alarms that monitor the load on the instances in your environment and are triggered if the load is too high or too low. When an alarm is triggered, your Auto Scaling group scales up or down in response.
- **AWS CloudFormation stack** — Elastic Beanstalk uses AWS CloudFormation to launch the resources in your environment and propagate configuration changes. The resources are defined in a template that you can view in the [AWS CloudFormation console](#).
- **Domain name** — A domain name that routes to your web app in the form `subdomain.region.elasticbeanstalk.com`.