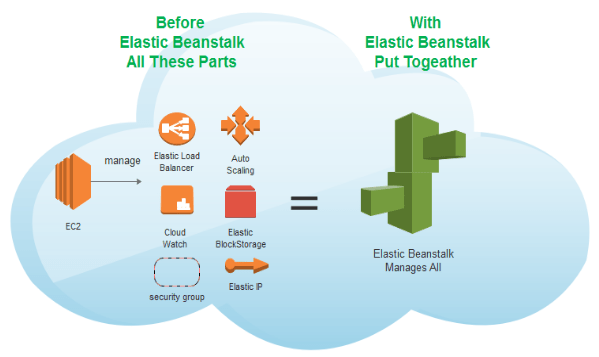
**Introduction to AWS Elastic Beanstalk**

## **Elastic Beanstalk by AWS: What Is It**

* A service from AWS for orchestration.
* Is employed to scale and deliver web applications and services.
* On well-known servers like Apache, Passenger, Nginx, and IIS, support Java, Python, Ruby,.NET, PHP, Node.js, Go, and Docker.
* Deploying your application to AWS in the quickest and simplest manner possible.
* It handles application health monitoring, load balancing, deployment, capacity provisioning, and auto-scaling.
* Managing AWS resources is completely up to you.

## [Feature of Elastic Beanstalk](https://k21academy.com/amazon-web-services/aws-devops/aws-elastic-beanstalk/)

**Also Check:**

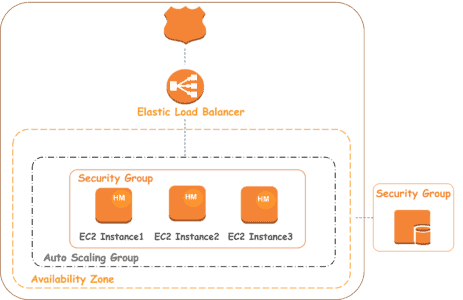
## **Benefits Of Elastic Beanstalk AWS**

* **Fast and simple to deploy:**It is the most **straightforward** and**time-efficient method** of deploying your application on AWS. You simply upload your application using the AWS Management Console, a Git repository, or an**integrated development environment** (IDE) such as Eclipse or Visual Studio, and it handles the deployment details of capacity provisioning, auto-scaling, load balancing, and application health monitoring. Your application will be ready to use in minutes, with no infrastructure or resource configuration required on your behalf.
* **Scalable:**Using simply changeable **Auto Scaling settings,** it automatically scales your application up and down based on its needs. You can, for example, use CPU utilization measurements to initiate Auto Scaling operations. This allows your application to accommodate **spikes in workload or traffic while minimizing costs.**
* **Developer productivity:** Amazon Beanstalk Elastic provisions and **operates the infrastructure and manages the application stack** (platform) for you, so you don’t have to spend the time or develop the expertise. It also keeps the underlying platform running your application up-to-date with the latest patches and updates. So, you can **focus on writing code rather than spending time managing and configuring servers, load balancers, databases, firewalls, and networks**.
* **Complete infrastructure control:** You are**free to select the AWS resources**, such as [**Amazon EC2 instance**](https://k21academy.com/amazon-web-services/aws-devops/aws-ec2-instance/) type, that are optimal for your application. Additionally, it lets you “open the hood” and allow you to have **full control over the AWS resources** powering your application. If you decide you want to take over some (or all) of the elements of your infrastructure, you can do so seamlessly by using Amazon Elastic Beanstalk’s management capabilities.

Also, Read [**Blue-Green Deployment**](https://k21academy.com/amazon-web-services/blue-green-deployment-in-aws/)**.**

## **Key Concepts**

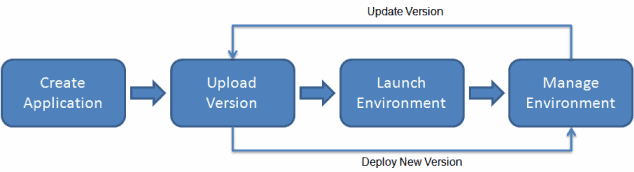
AWS Elastic Beanstalk enables you to manage all of the resources that run your application as environments. Here are some key concepts:



* **Application:** Amazon Elastic Beanstalk application is a logical collection of Elastic Beanstalk **components, including environments, environment configurations, and versions**. In Elastic Beanstalk an application is conceptually similar to a folder.
* **Application version:** An application version of Amazon Elastic Beanstalk refers to a specific, labeled iteration of deployable code for a web application. An application version points to an [Amazon Simple Storage Service (Amazon S3)](https://k21academy.com/amazon-web-services/amazon-s3-bucket-and-storage-classes/) object that contains the deployable code, such as a Python WAR file.
* **Environment:** An environment is a collection of AWS resources running an application version. Each environment runs only one application version at a time, still, **you can able to run the same application version or different application versions in many environments simultaneously.**
* **Environment configuration:**An environment configuration determines a collection of parameters and settings that define how an environment and its associated resources behave. When you update an environment’s configuration settings, **It automatically applies the changes to existing resources or deletes and deploys new resources depending on the changes you made**.
* **Saved configuration:** A saved configuration is a template that you can use as a starting point for creating unique environment configurations. You can create and modify saved configurations and **apply them to environments, using the Elastic Beanstalk console, AWS CLI, EB CLI, or API.**
* **Platform:** A platform is a combination of an **operating system, programming language runtime, application server, web server, and Elastic Beanstalk components.** It provides a variety of platforms on which you can build your applications.

**How Does AWS Elastic Beanstalk Operate?**

You can construct an application using Elastic Beanstalk, upload an application version in the form of an application code bundle (for instance, a Python.war file), and then provide some information about the program. The AWS resources required to run your code are automatically created and configured by Elastic Beanstalk. **You can manage your environment and roll out new application versions once your environment has launched**. The workflow of Elastic Beanstalk is shown in the diagram below.



It supports the DevOps practice name **“rolling deployments.”** When **enabled, your configuration deployments work hand in hand with Auto Scaling** to ensure there are always a defined number of instances available as configuration changes are made. It gives you control as [Amazon EC2 instances](https://k21academy.com/amazon-web-services/aws-solutions-architect/aws-ec2-instance/) are updated.

**Read**this blog where we have discussed the 6 Pillars of [AWS Well-Architected Framework](https://k21academy.com/amazon-web-services/aws-solutions-architect/5-pillars-of-aws-well-architected-framework/) and their design principles.

* 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.

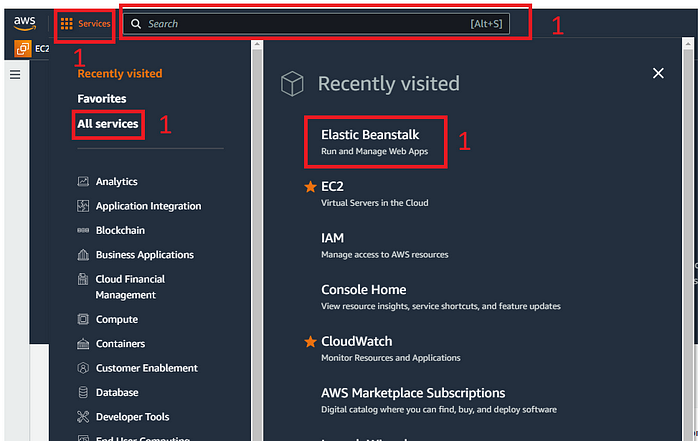
**Integration with Other AWS Services**

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

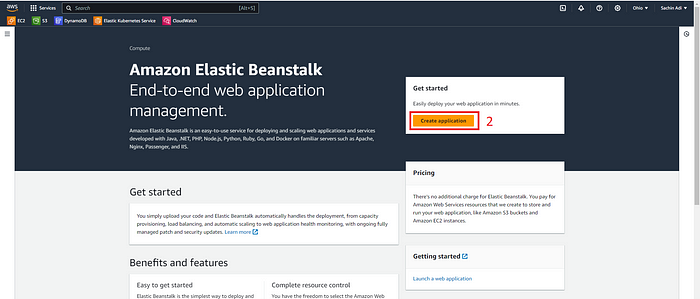
**Demo**

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

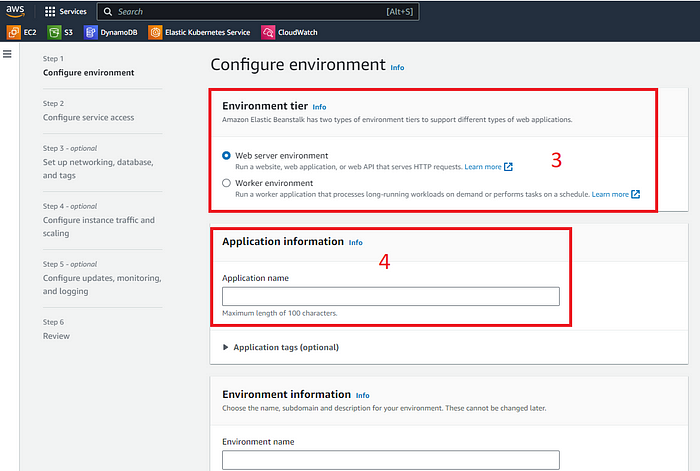
* Navigate to All Services and look for Elastic Beanstalk



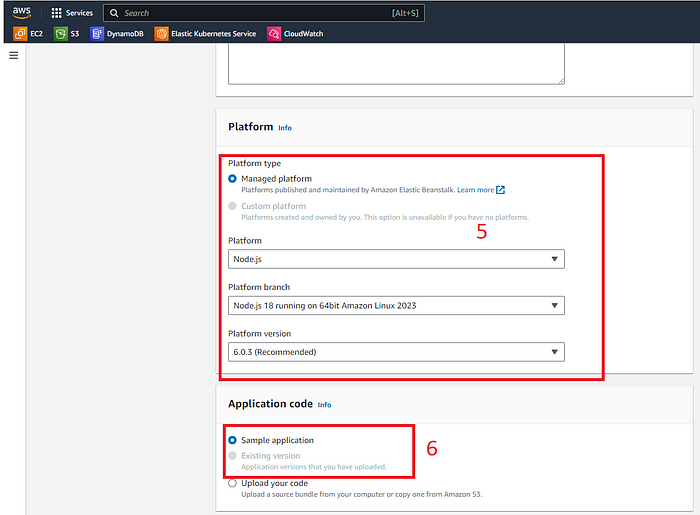
* Click on Create Application



* Select the Web Server Environment, input the Application Name

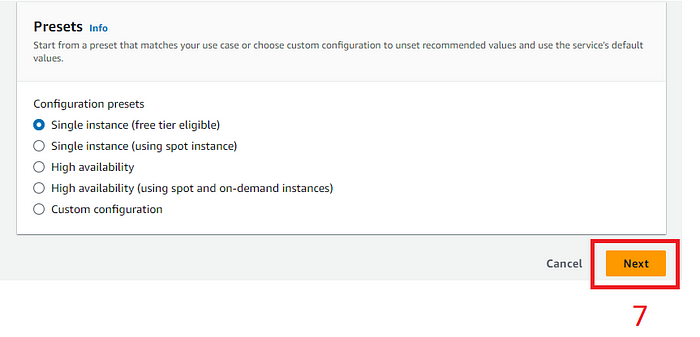


* Enter the platform type.

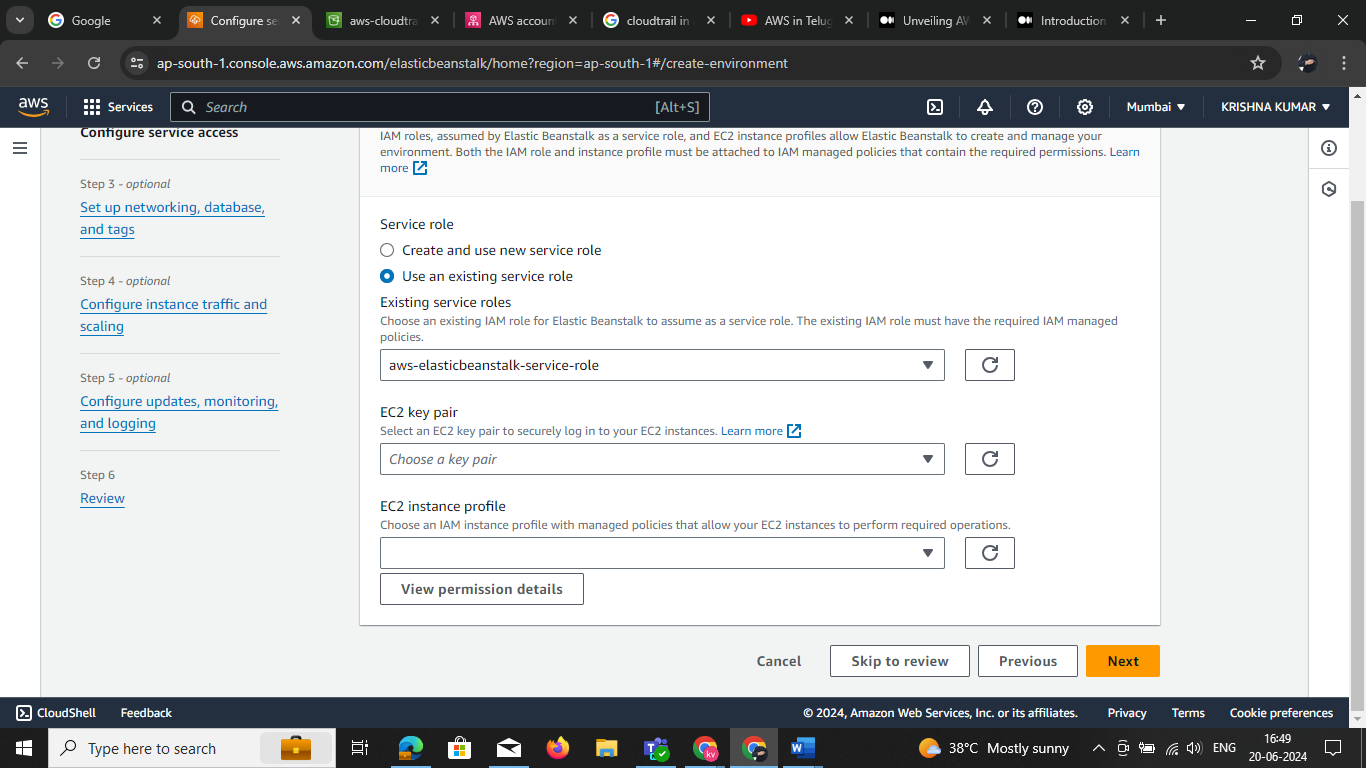


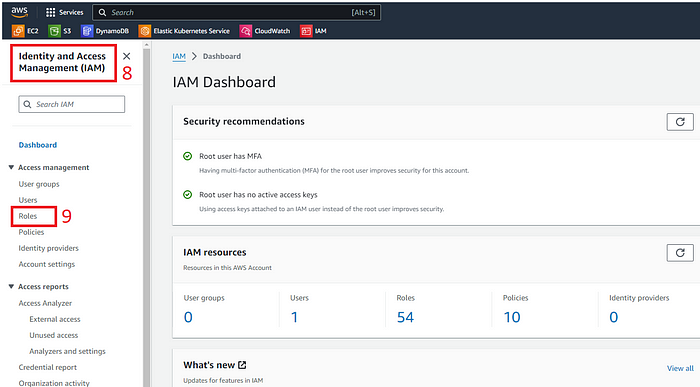
Platform specification

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

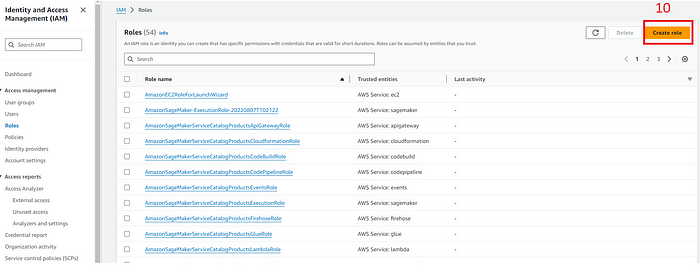


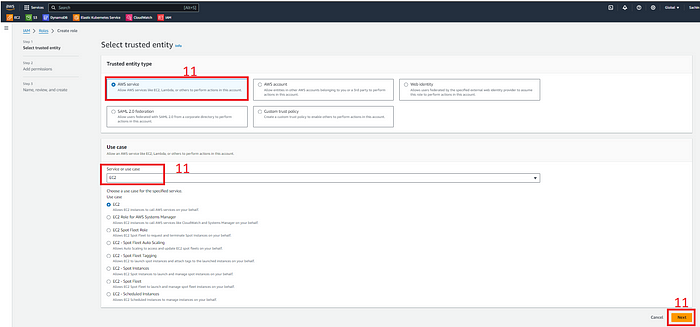
Configuring the presets

* 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



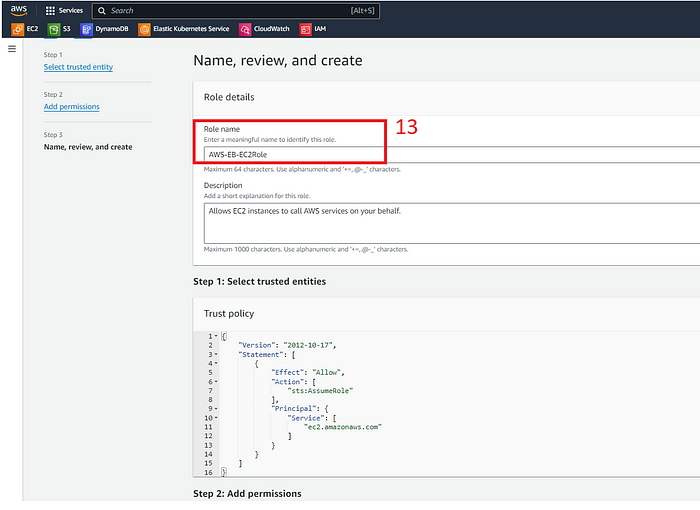
* **Identity and access Management configurations**

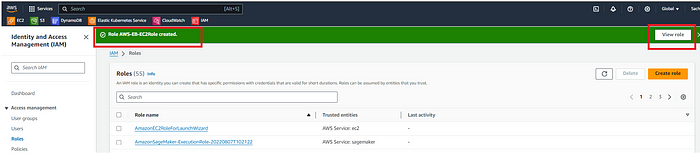




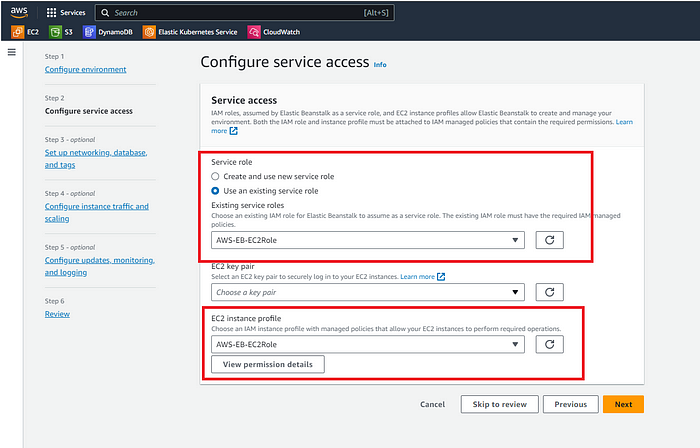
**Role selection on the IAM portal**

Look for the Elastic Beanstalk service and add the AWSElasticBeanstalkMulticontainerDocker, AWSElasticBeanstalkWebTier, and AWSElasticBeanstalkWorkerTier roles. Provide a name to this role and create the role. Your custom role need add the custom polices in the above



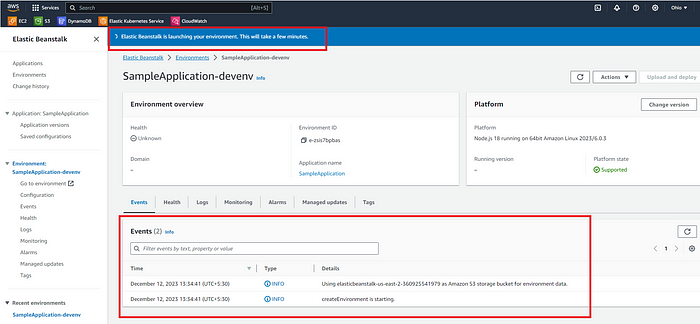


* Back to the console where we were configuring the Elastic Beanstalk, we will need to add this IAM role

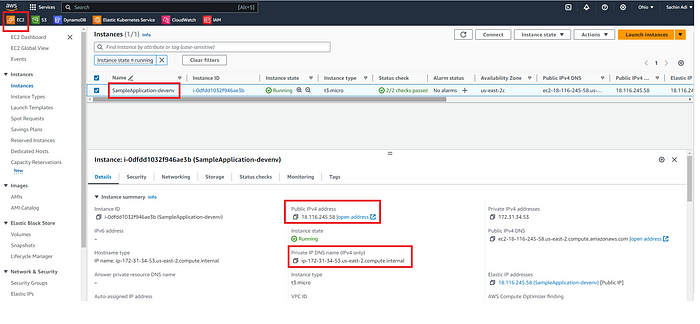


Configuring Service Access

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

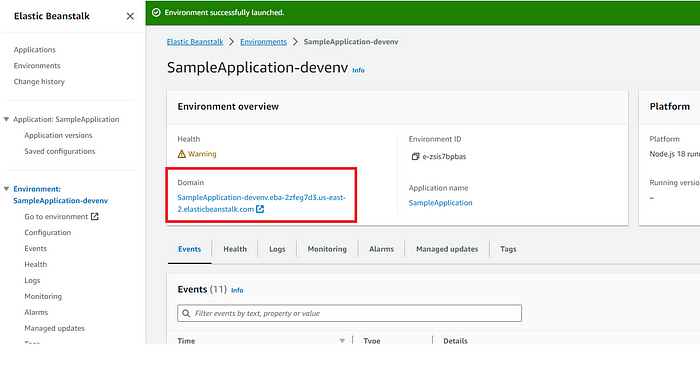


Elastic Beanstalk console

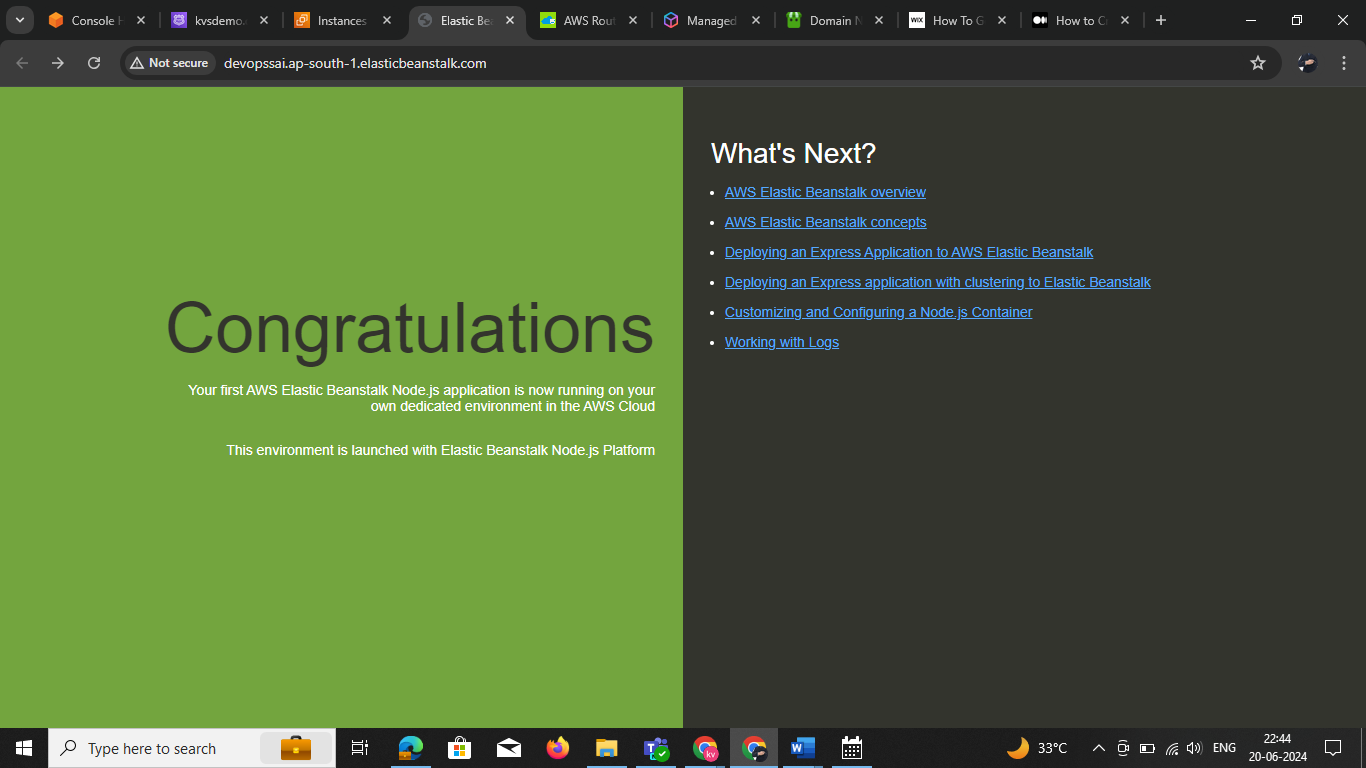


**EC2 console displaying the virtual machine created during process**

* 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.