Roll NO. A018

**Cloud Computing Practical No: 02 Writeup:**
**1.Platform as a service**

Platform as a Service (PaaS) is a cloud computing service model that provides a platform allowing customers to develop, run, and manage applications without dealing with the complexity of building and maintaining the underlying infrastructure. PaaS sits between Infrastructure as a Service (IaaS) and Software as a Service (SaaS) in the cloud computing service stack.

**Characteristics of PaaS:**

Development Frameworks: PaaS provides a set of tools, services, and development frameworks that allow developers to build, test, and deploy applications more efficiently.

Middleware Services: PaaS often includes middleware services such as databases, messaging systems, and caching services. These services abstract the underlying infrastructure, making it easier for developers to focus on application development.

Automated Deployment and Scaling: PaaS platforms automate the deployment and scaling of applications. Developers can easily scale their applications up or down based on demand without having to manage the underlying infrastructure.

Integrated Development Tools: PaaS offers integrated development tools, including code repositories, version control, and collaboration tools, facilitating collaboration among development teams.

Multi-Tenancy: PaaS platforms are designed to support multiple users or tenants, allowing developers to share development tools and resources in a secure and efficient manner.

**Advantages of PaaS:**

Faster Development: PaaS accelerates the development process by providing ready-to-use components and services.

Cost-Efficiency: Users can avoid the costs and complexities associated with maintaining and managing infrastructure.

Scalability: PaaS platforms offer automatic scaling, ensuring applications can handle varying workloads.

**Disadvantages of PaaS:**

Less Control: Developers have less control over the underlying infrastructure compared to IaaS.

Dependency on Provider: Users are dependent on the PaaS provider for updates, security patches, and overall platform stability.

**2.Amazon Elastic Beanstalk**

Amazon Elastic Beanstalk is a web infrastructure management service. It handles deployment and scaling for web applications and services.Elastic Beanstalk can

automatically manage setup, configuration, scaling and provisioning for other AWS services. AWS Elastic Beanstalk is an AWS-managed service for web applications. Elastic Beanstalk is a pre-configured EC2 server that can directly take up your application code and environment configurations and use it to automatically provision and deploy the required resources within AWS to run the web application. Unlike EC2 which is Infrastructure as a service, Elastic Beanstalk is a Platform As A Service (PAAS) as it allows users to directly use a pre-configured server for their application. Of course, you can deploy applications without ever having to use elastic beanstalk but that would mean having to choose the appropriate service from the vast array of services offered by AWS, manually provisioning these AWS resources, and stitching them up together to form a complete web application. Elastic Beanstalk abstracts the underlying configuration work and allows you as a user to focus on more pressing matters.

### 3.)Components of Amazon ElasticBeanStalk

- **Application:** Elastic Beanstalk directly takes in our project code. So Elastic Beanstalk application is named the same as your project home directory.
- **Application Environments:** Users may want their application to run on different environments like DEV, UAT, and PROD. You can create and configure different environments to run applications on different stages.
- **Environment Health:** One of the most lucrative features of running applications on AWS or most of the other cloud platforms is automated health checks. AWS runs automatic health checks on all EC-2 deployments (Elastic Beanstalk is a managed EC-2 service) which can be monitored from the AWS console. For example, in the case of web applications AWS will regularly, as scheduled by the developers, ping the application to check if the response is status code 200 and if the application is running as expected. Health check responses:
- **Red:** The application failed all health tests.
- **Yellow:** The application failed some of the health tests.
- **Grey:** The application is updating.
- **Green:** The application passed the health check successfully.
- **Isolated:** All environments within a single application are isolated from each other (independent of each others' running states). Needless to say, two different applications are also isolated.
- **Scalability:** Using Auto-Scaling within Elastic Beanstalk makes the application dynamically scalable.
- **Elastic Load Balancing:** All the web requests to the application are not directly relayed to application instances. They first hit the Elastic Load Balancer (ELB), which, as the name suggests, balances the load across all the application instances.
- **Language support:** Elastic Beanstalk supports the applications developed with Java, .NET, PHP, Node.js, Python, Ruby, Go, and Docker on familiar servers such as Apache, Nginx, Passenger, and IIS.
- **Pricing:** There is no extra charge for using Elastic Beanstalk. Users are only required to pay for the services and resources provisioned by Elastic Beanstalk Service.
- **Automatic Provisioning:** Elastic Beanstalk takes away the burden of choosing the right services and configuring their security groups to work together.
- **Impossible to Outgrow:** AWS claims that since Elastic Beanstalk uses the Auto Scaling feature it can, in theory, handle any amount of internet traffic.

**4.)IAM:**

IAM stands for Identity and Access Management. In the context of Amazon Web Services (AWS), IAM refers to the service that allows you to manage access to AWS resources securely. IAM enables you to control who (authentication) can do what (authorization) in your AWS environment.

Here are key aspects of AWS Identity and Access Management (IAM):

**Users and Groups:**

Users: Represent individuals or entities that interact with AWS services. Each user has a unique set of security credentials.

Groups: Users can be organized into groups, and permissions can be assigned to groups, making it easier to manage access.

**Roles:**IAM roles define a set of permissions for making AWS service requests. Roles are not associated with a specific user or group but can be assumed by users, applications, or services when needed.

**Policies:**IAM policies are JSON documents that define permissions. They can be attached to users, groups, or roles, specifying what actions are allowed or denied on what resources.

**Access Keys:**IAM provides access keys (access key ID and secret access key) for programmatic access to AWS services. These keys are often used by developers and applications.

**Multi-Factor Authentication (MFA):**IAM supports MFA, an additional layer of security that requires users to provide a second form of authentication (such as a code from a virtual or hardware MFA device) in addition to their password.

**Identity Federation:**IAM allows you to integrate with external identity providers, such as Active Directory or social identity providers, to grant temporary access to AWS resources.

**Resource-Level Permissions:**IAM policies can define permissions not only at the service level but also at the resource level. This allows fine-grained control over access to specific AWS resources.

**IAM Roles for EC2 Instances:**IAM roles can be assigned to EC2 instances, allowing applications running on those instances to securely access AWS resources without embedding credentials in the code.

**Policy Conditions:**IAM policies can include conditions that must be met for the policy to be in effect. Conditions can be based on factors such as the time of day, the source IP address, or the use of MFA.

**Implement paas using elastic beanstalk for the following.**

**1. Server**

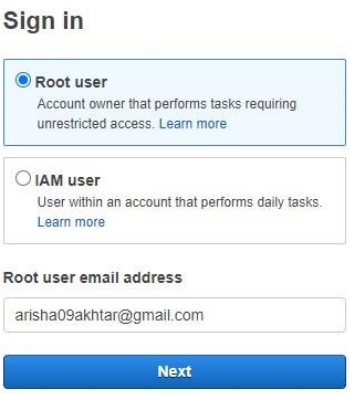
**2. Java**

**3. Python**

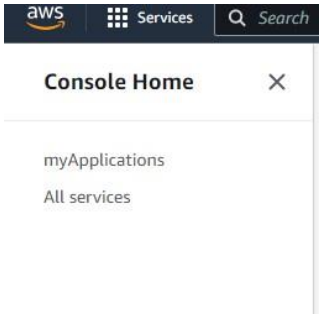
## 4. Node.js

## For Server

1) Sign In to your aws acc

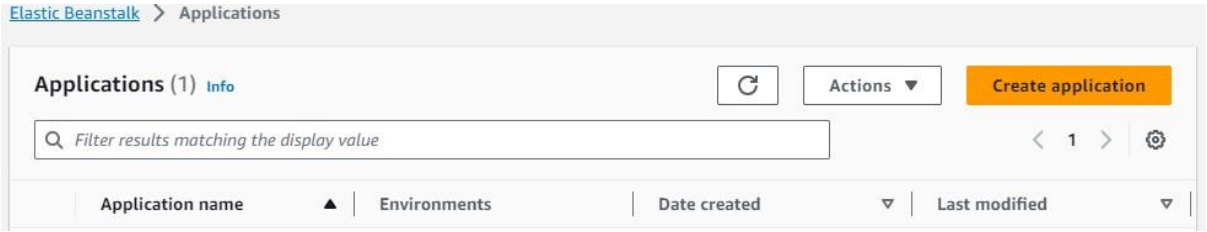



2.Select all services

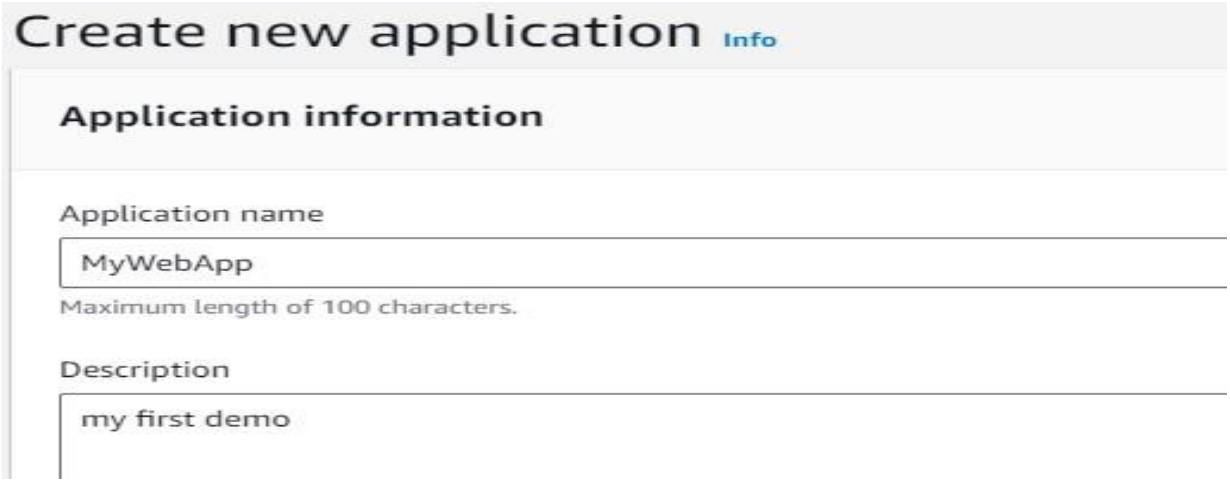



3.Select Elastic Beanstalk

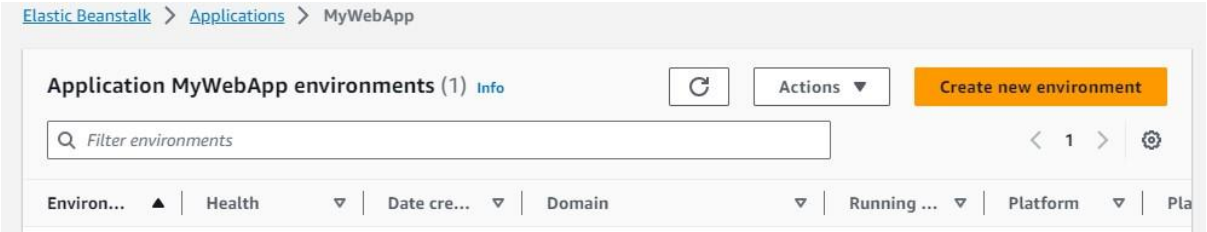

4.Click on Create Application



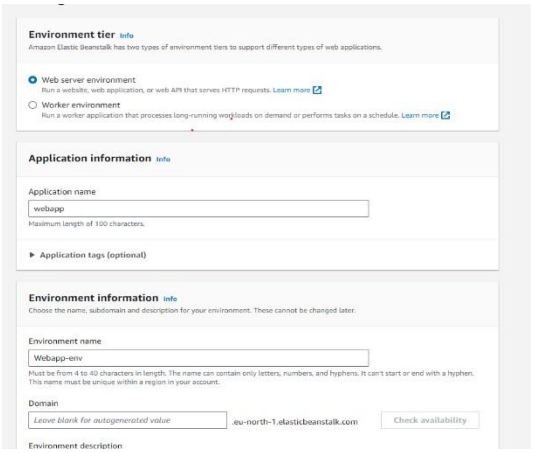

5.Enter application name and description and then click on create

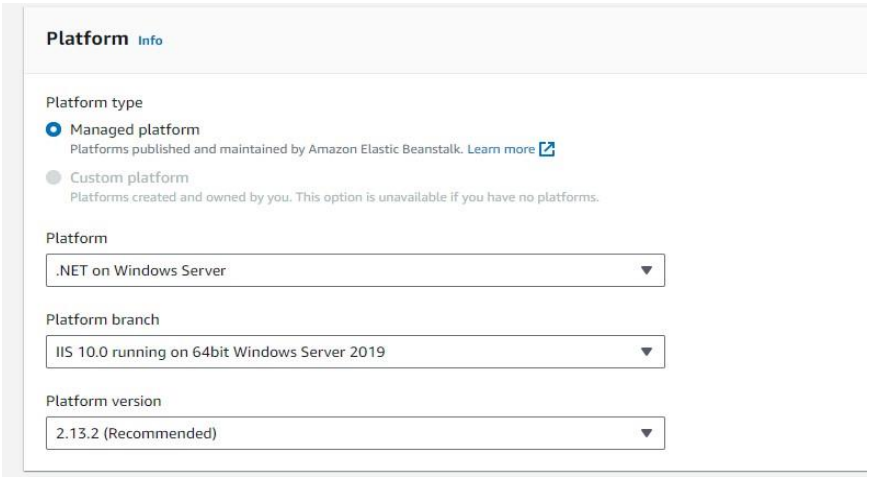


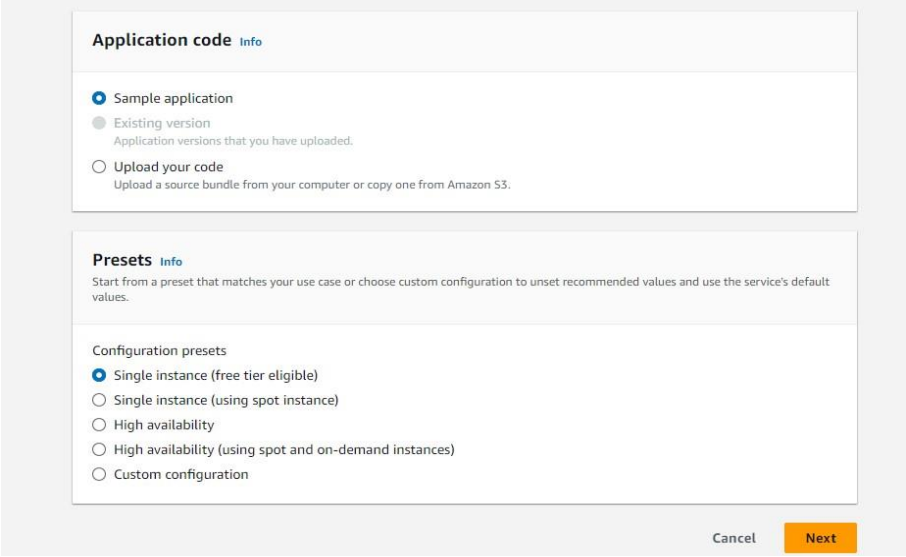

6. Now create environment



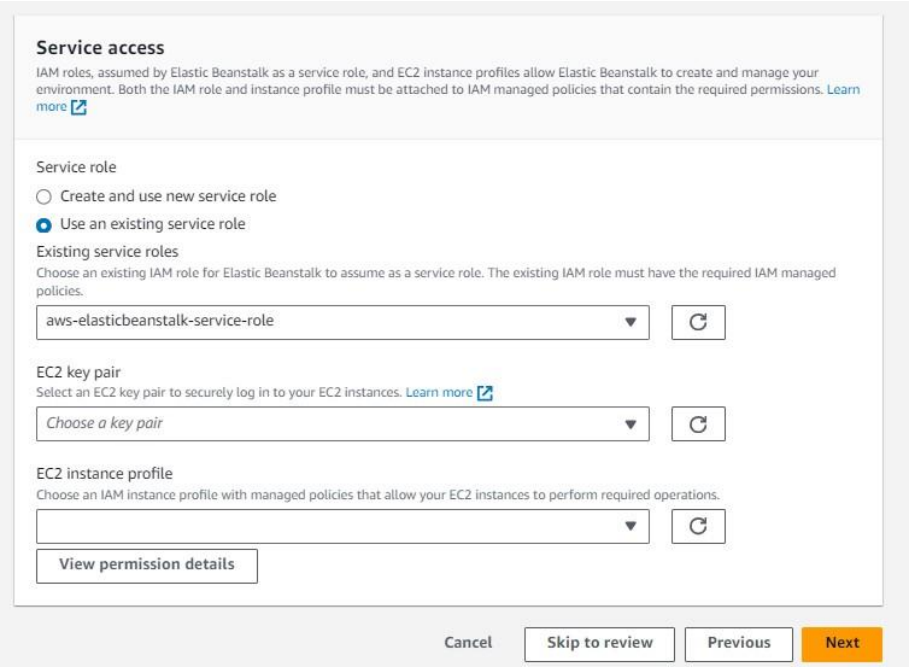

7.For Configure environment everything will be default except platform since we are doing for ser ver we will choose .Net on Windows server Platform branch and platform version will be default.Click on next.



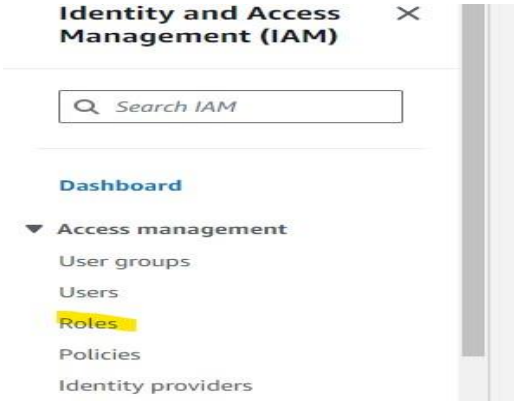



8.For Service access you have to create a role since it is by default over here you can create your own role .

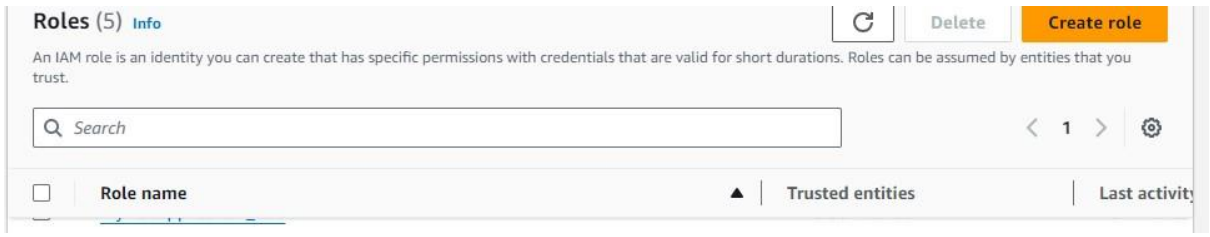



9.To create role go to services IAM

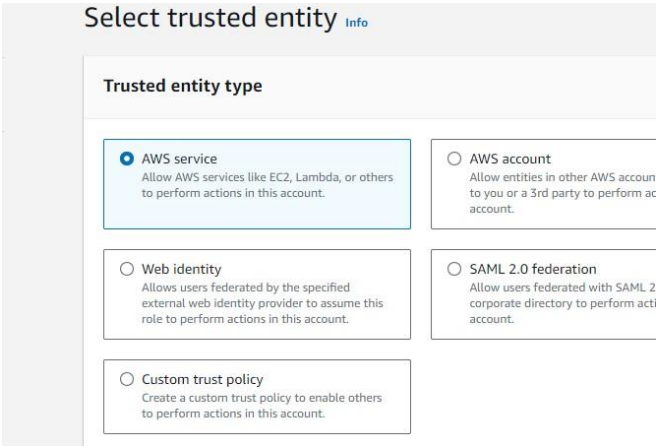

10.Inside IAM dashboard go to roles

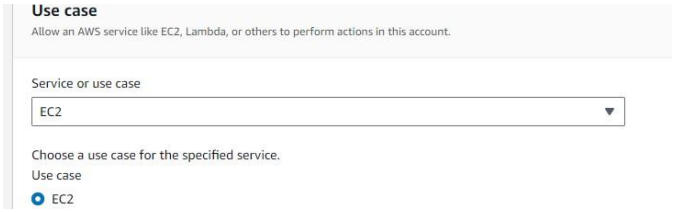



11.Create role

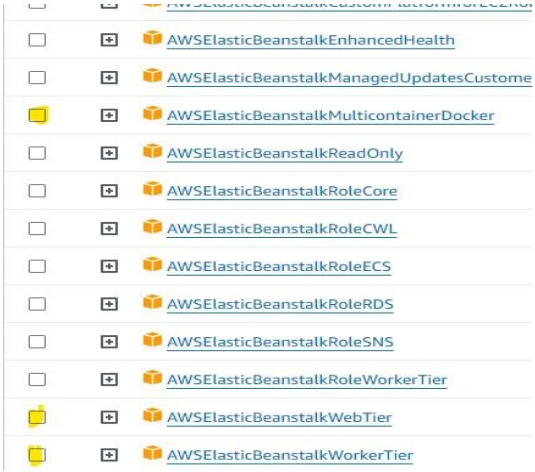



12.It will ask for entity select AWS Service and for use case select EC2 and click on next

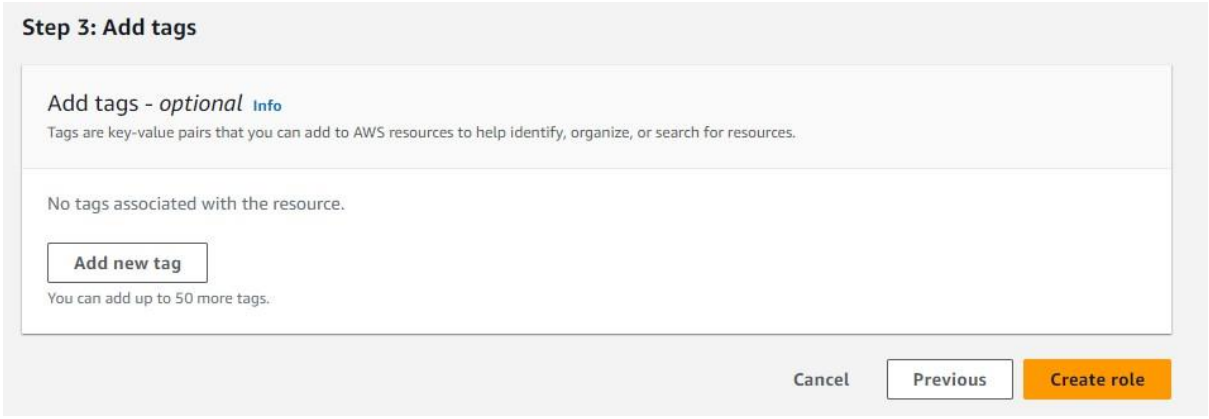


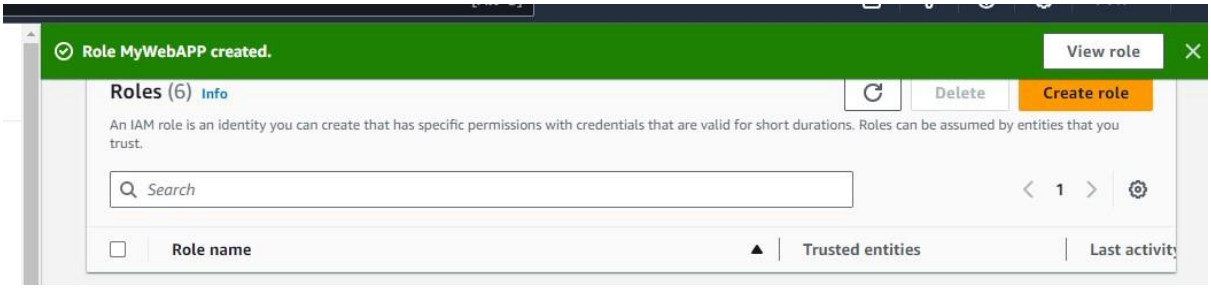

13.Search elasticbeanstalk on top select this three opt and click on next



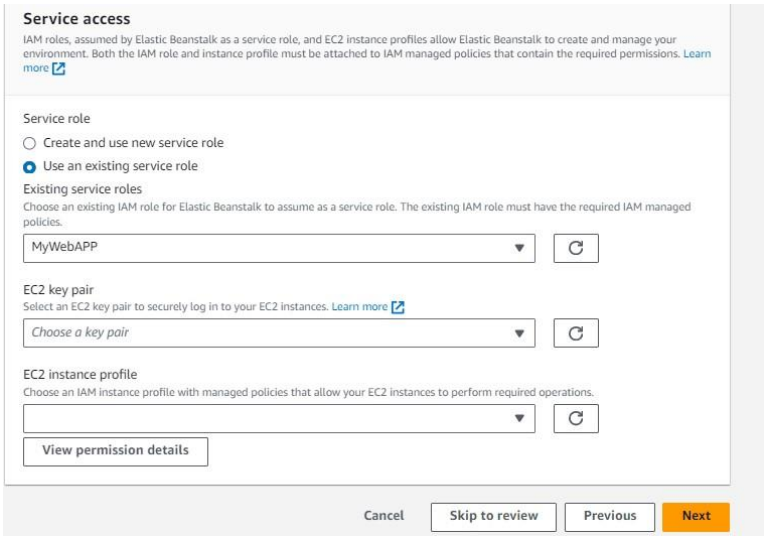

14.Give role name and click on create role



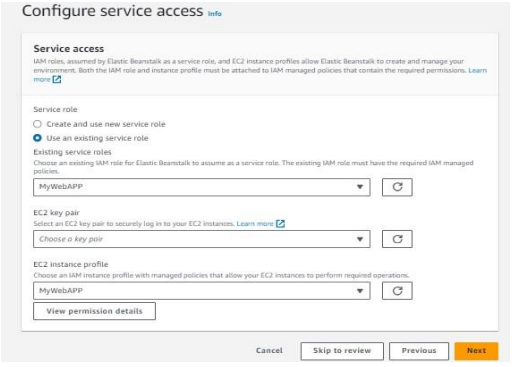

15.Role will get created




16.Now you can again go to your service access that is your 8 step now inside existing service roles refresh it and click on dropdown you can see over there your role name which you have created above automatically comes there click on that

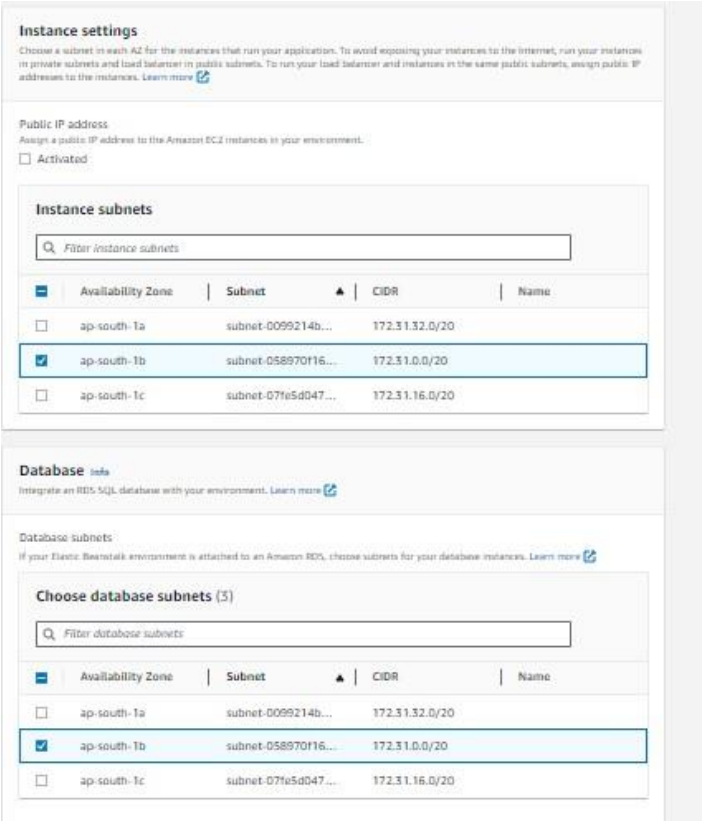



17.For instance profile click on dropbutton your role name will come select that then enter next

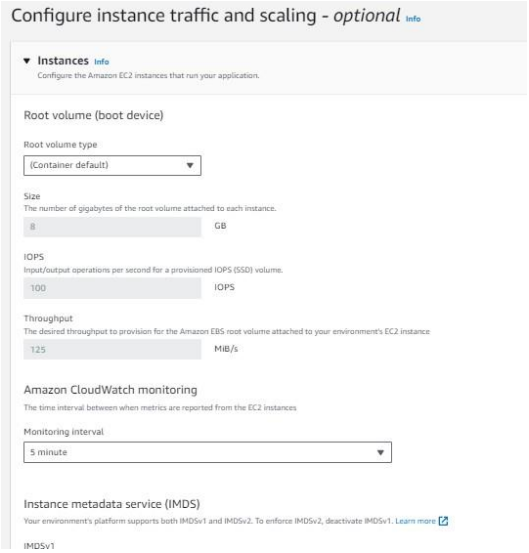



18.After clicking on next it will ask for VPC click on dropdowm and select which is default not down the ip address of this

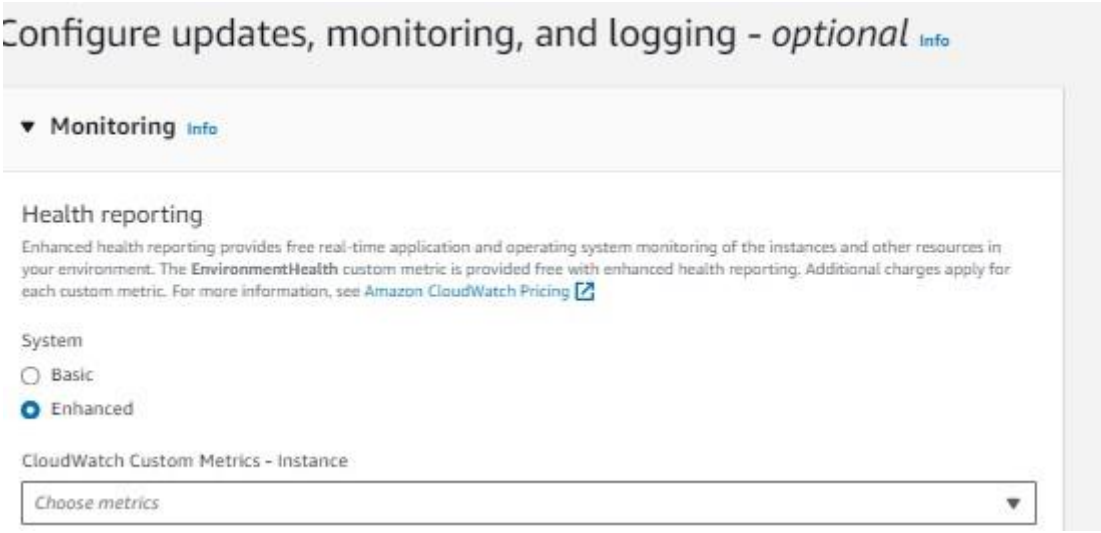

19.The ip address you have selected above same over here select that ip address and click on next



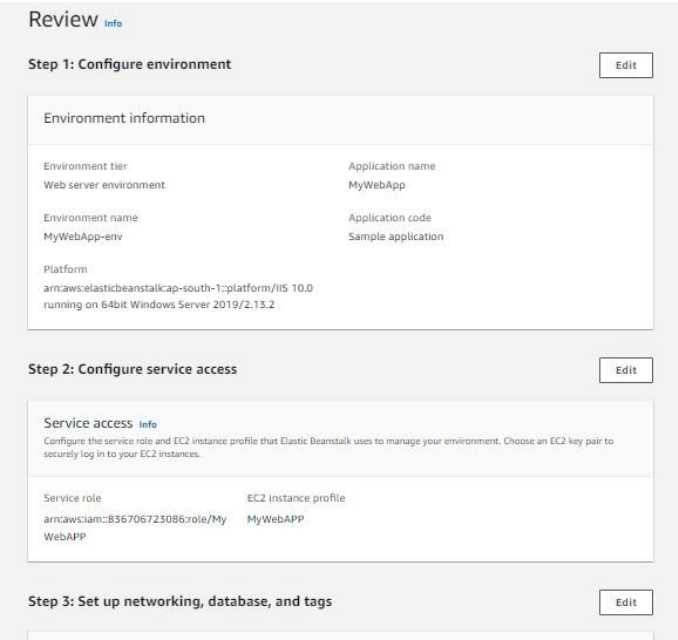

20.for configure instance everything will be by default enter next



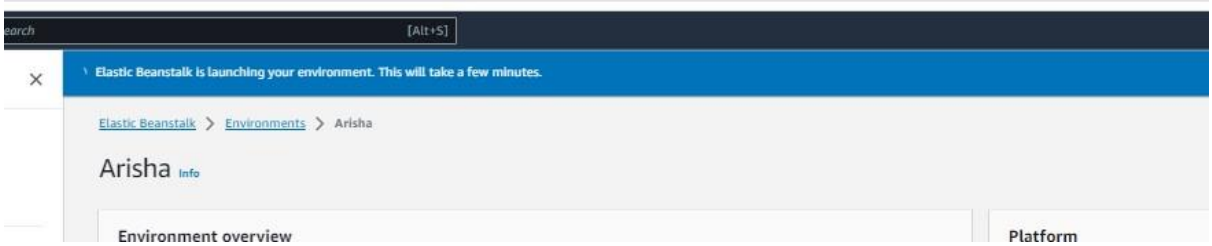

21.For configure update also everthing will be default click to next



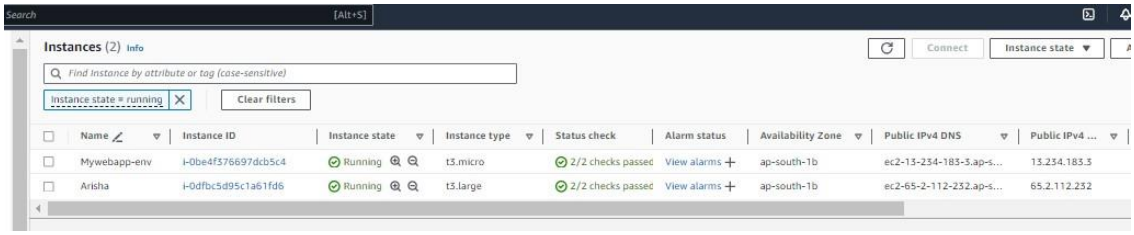

22.Review will display click on submit



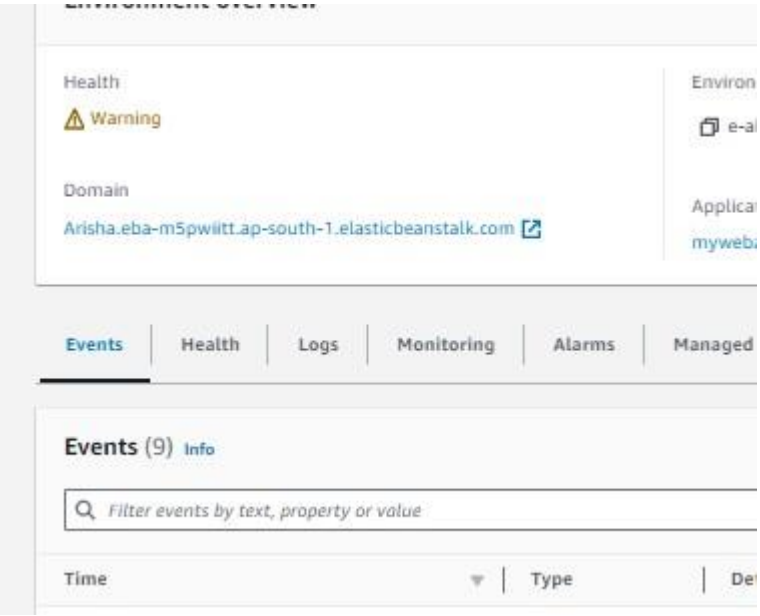

23.You can see it is getting launch




24.Now you can go to instance from my service you can see The instance is running.



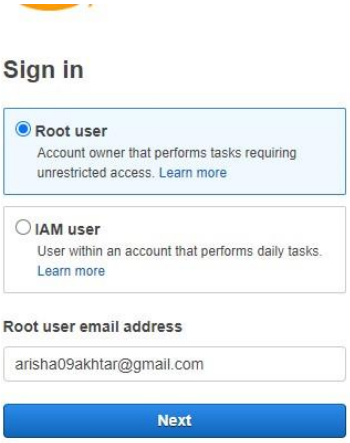

25.Click on domain



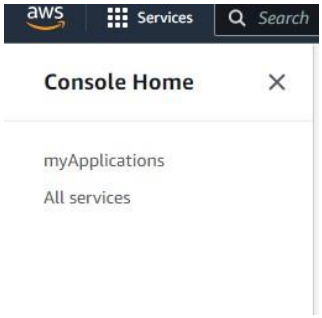

26. It will get open into another brower now Elastic Beanstalk Server is running on your own dedicated environment.


## For Java

1.)Sign In to your aws acc

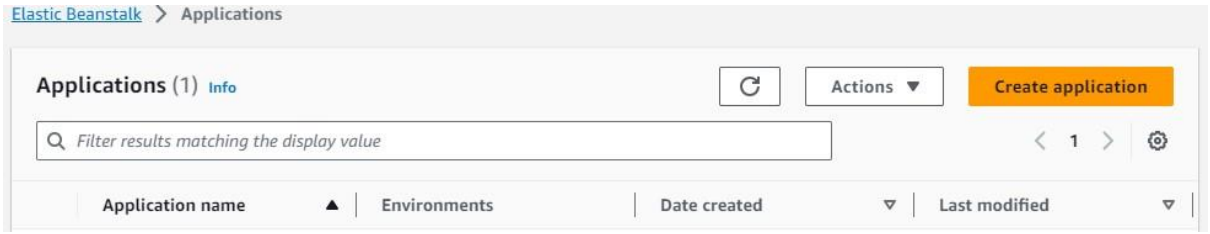



2.)Select all services

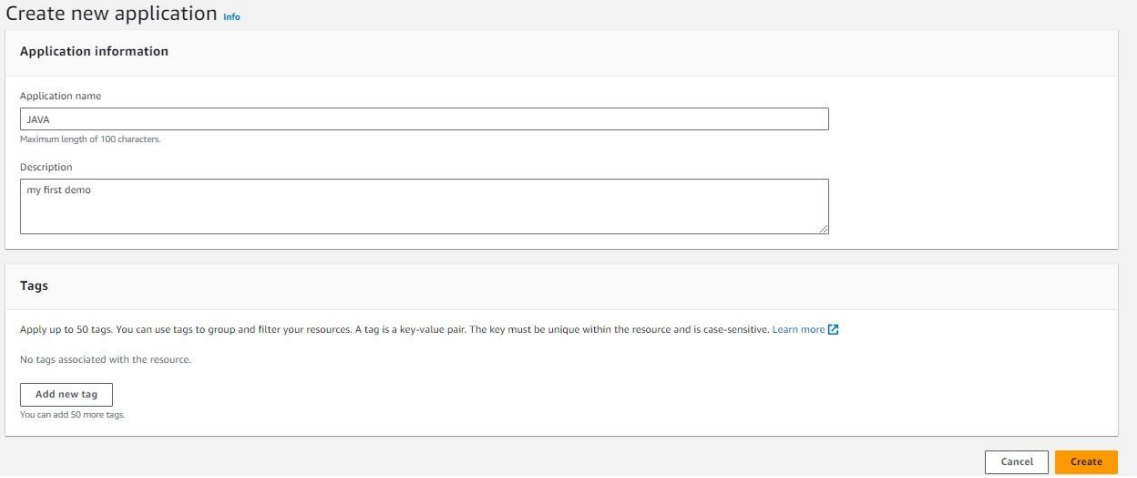



3.)Select Elastic Beanstalk

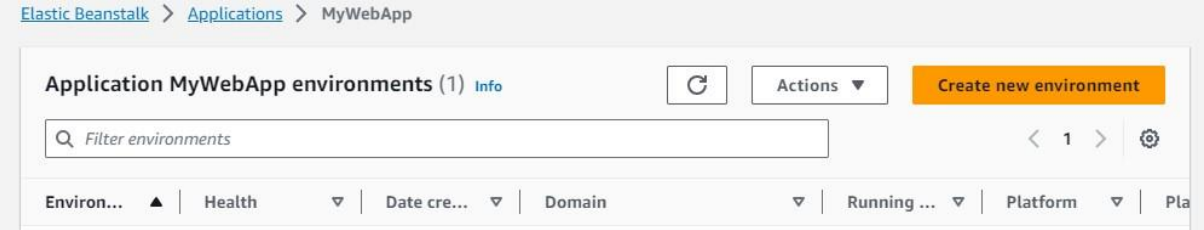

4.) Click on Create Application



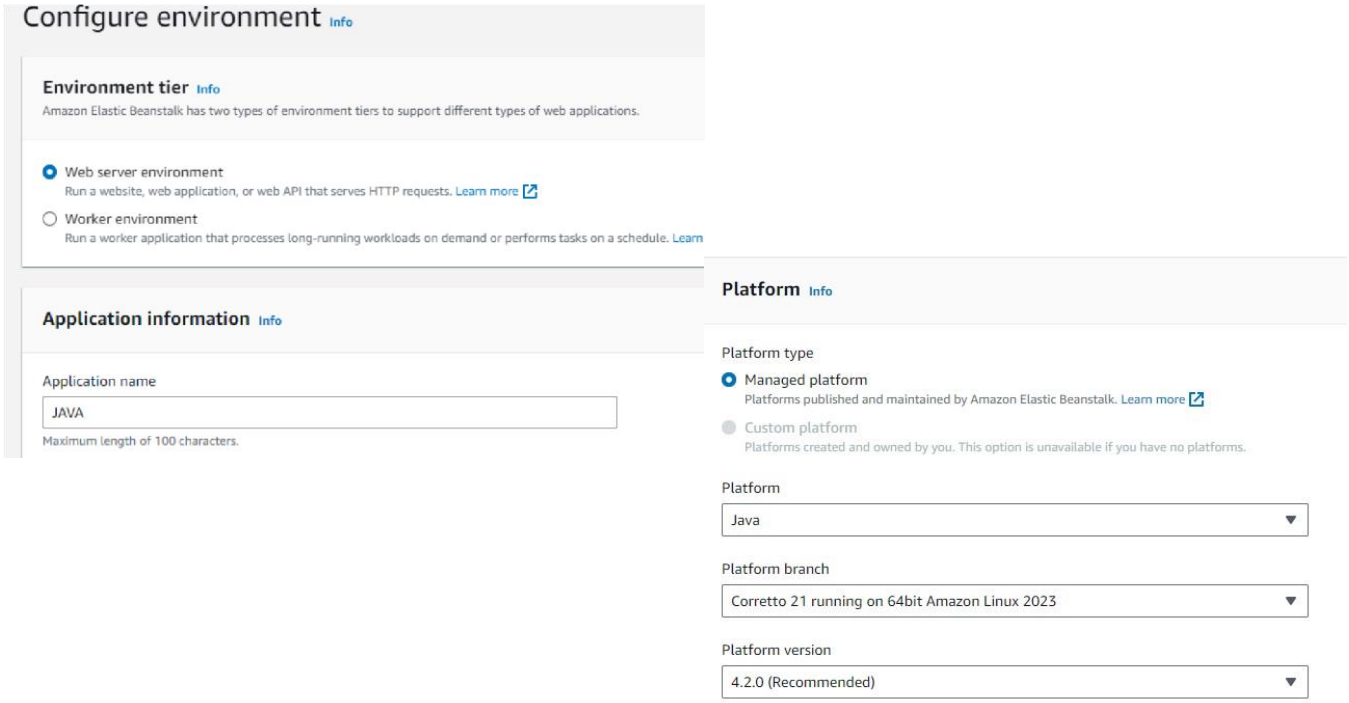

5.) Enter application name and description and then click on create



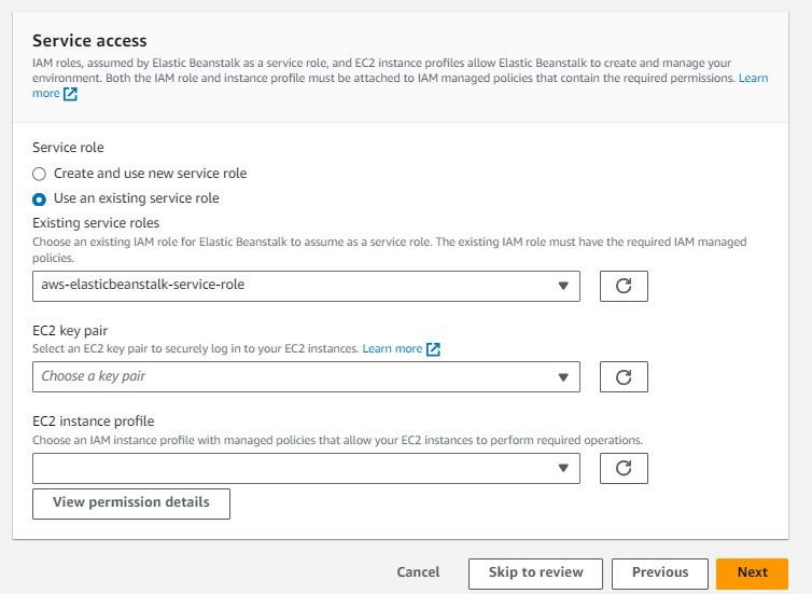

6. Now create environment




7.For Configure environment everything will be default except platform since we are doing for Java we will choose .Net on Windows server Platform branch and platform version will be default.Click on next.

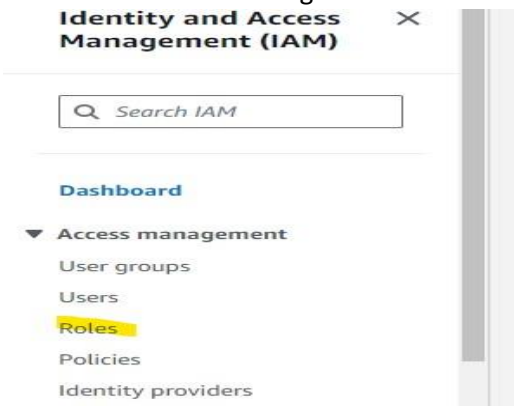



8.) For Service access you have to create a role since it is by default over here you can create your own role .

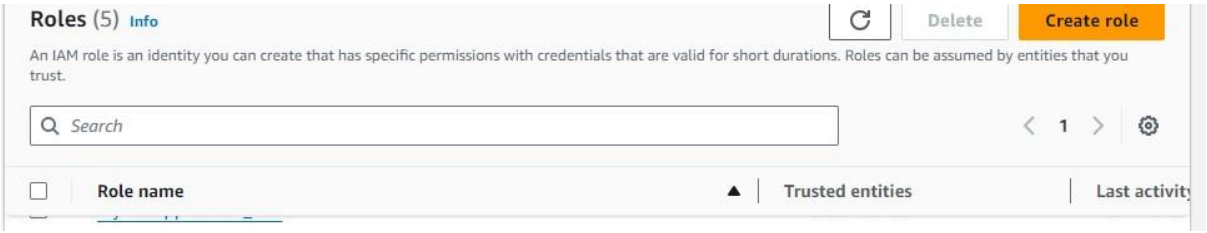



9.)To create role go to services IAM

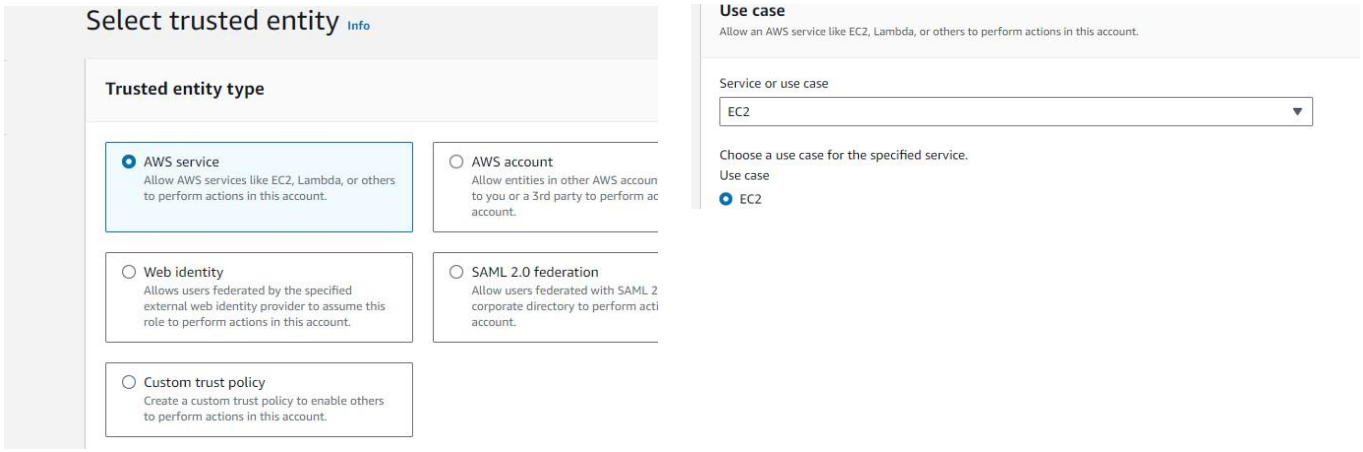

10.Inside IAM dashboard go to roles



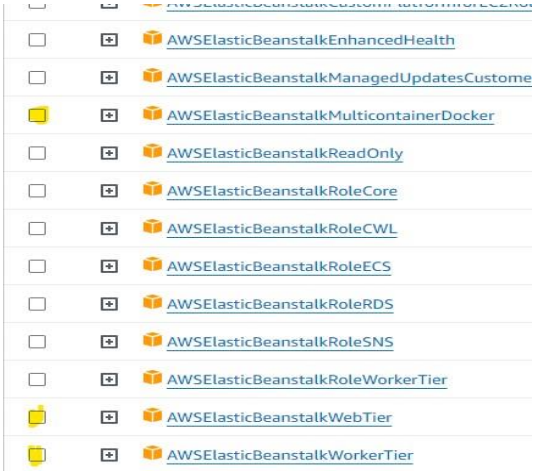

11.Create role



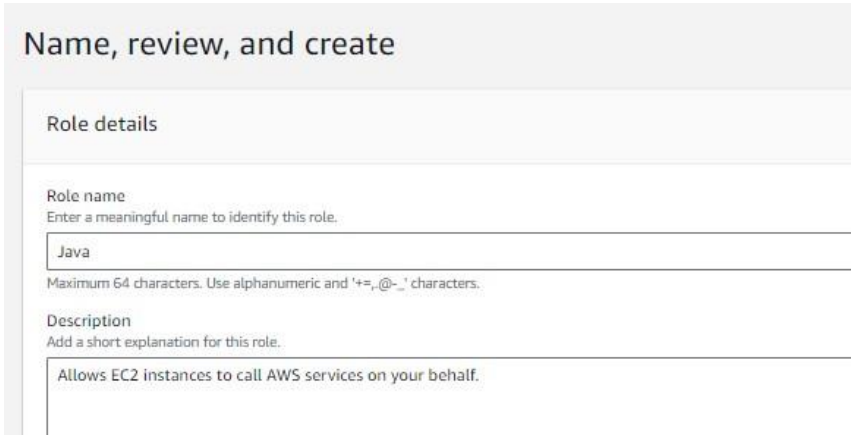

12.It will ask for entity select AWS Service and for use case select EC2 and click on next



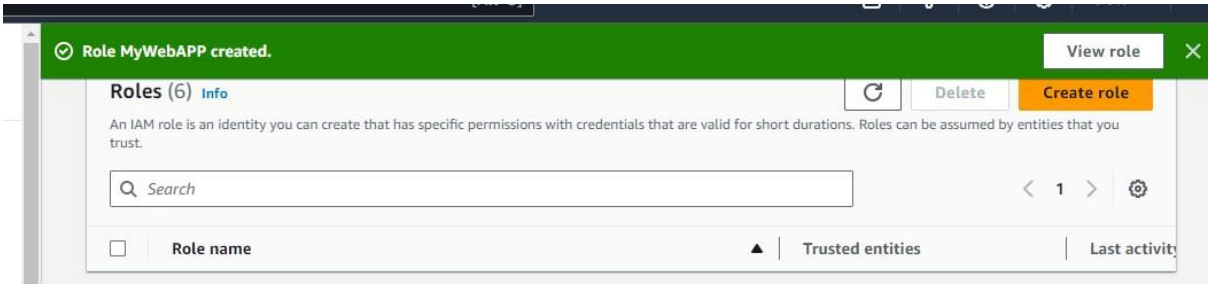

13.Search elasticbeanstalk on top select this three opt and click on next



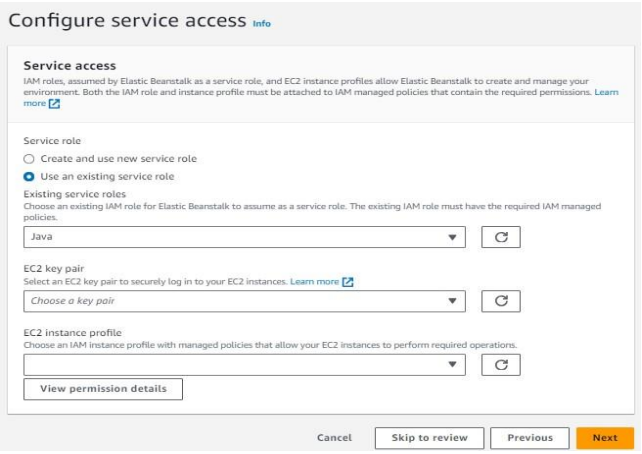

14.Give role name and click on create role



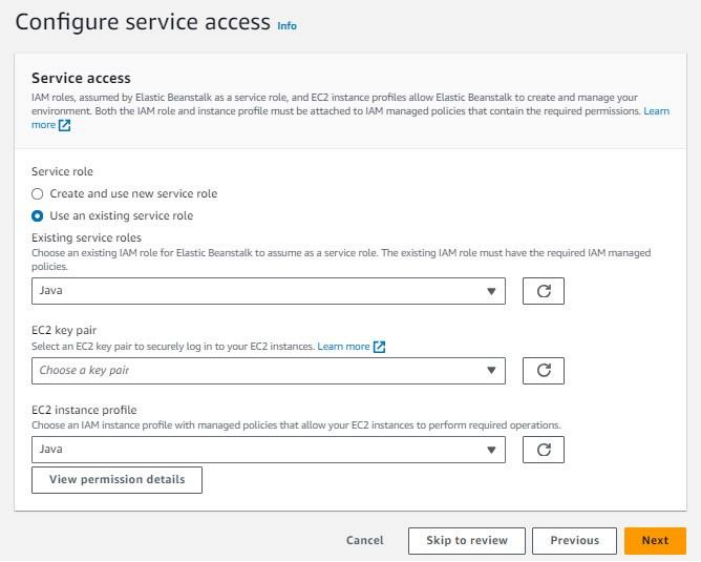

15.Role will get created



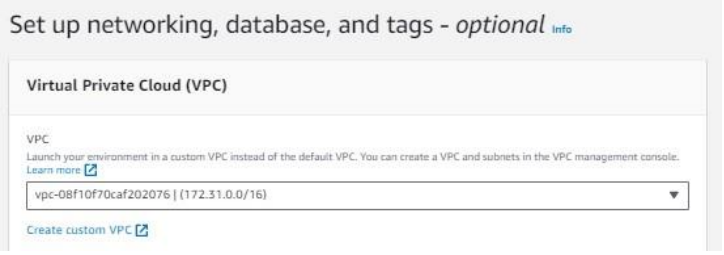

16.Now you can again go to your service access that is your 8 step now inside existing service roles refresh it and click on dropdown you can see over there your role name which you have created above automatically comes there click on that



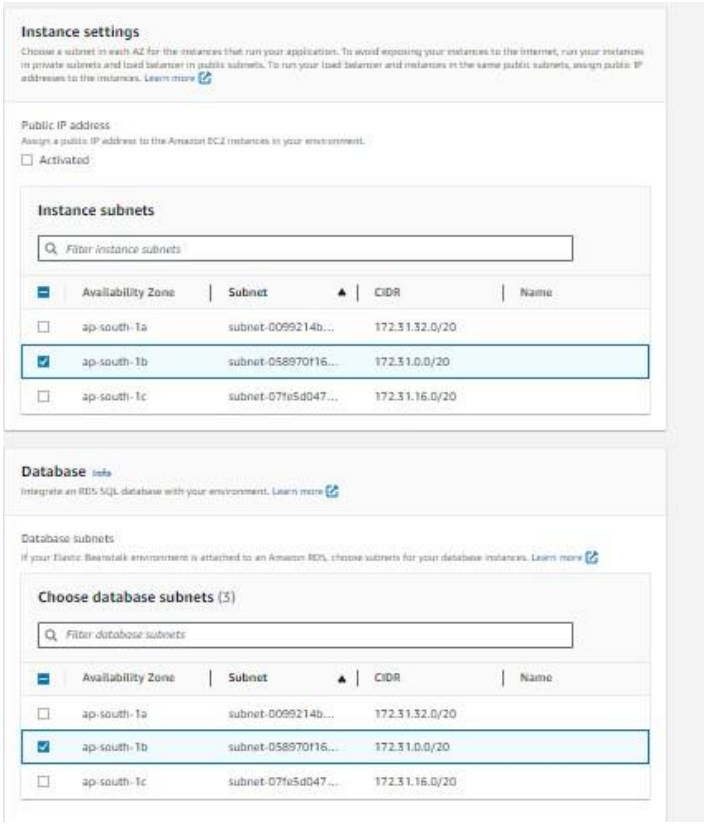

17.For instance profile click on dropbutton your role name will come select that then enter next



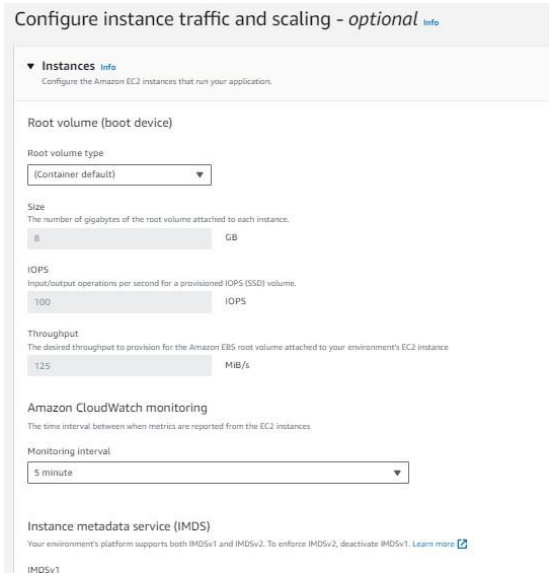

18.After clicking on next it will ask for VPC click on dropdowm and select which is default not down the ip address of this



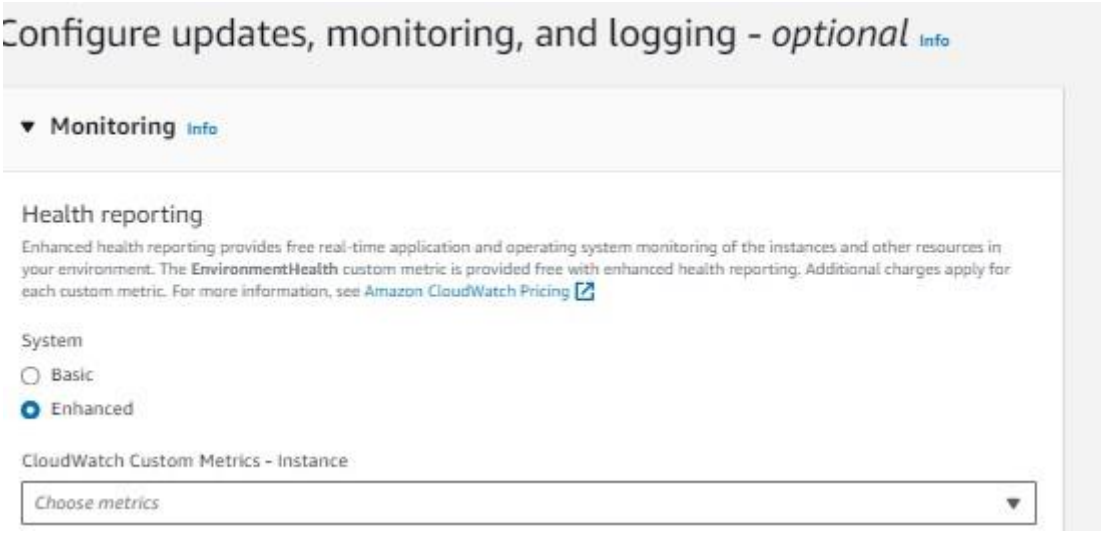

19.The ip address you have selected above same over here select that ip address and click on next



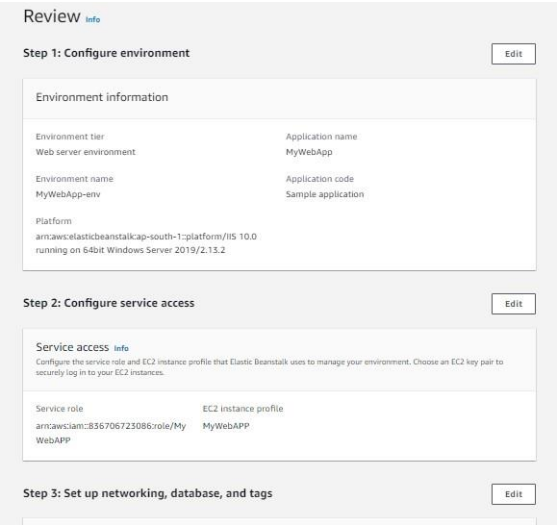

20.for configure instance everything will be by default enter next



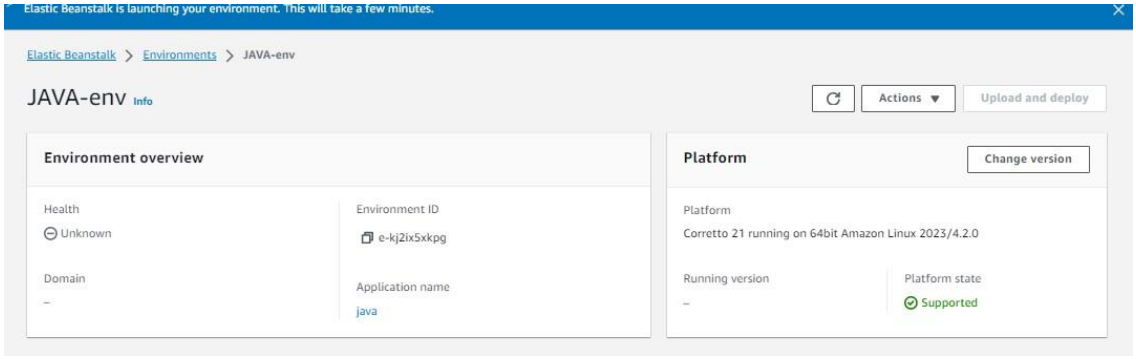

21.For configure update also everthing will be default click to next



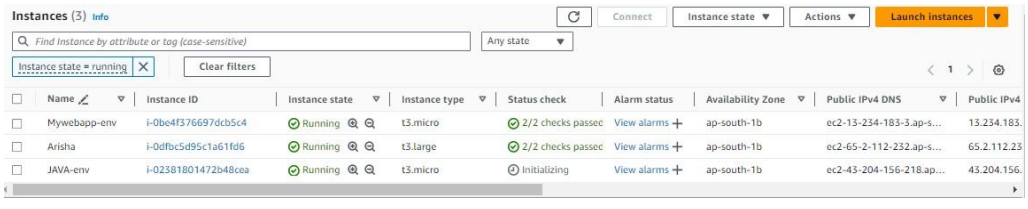

22.Review will display click on submit



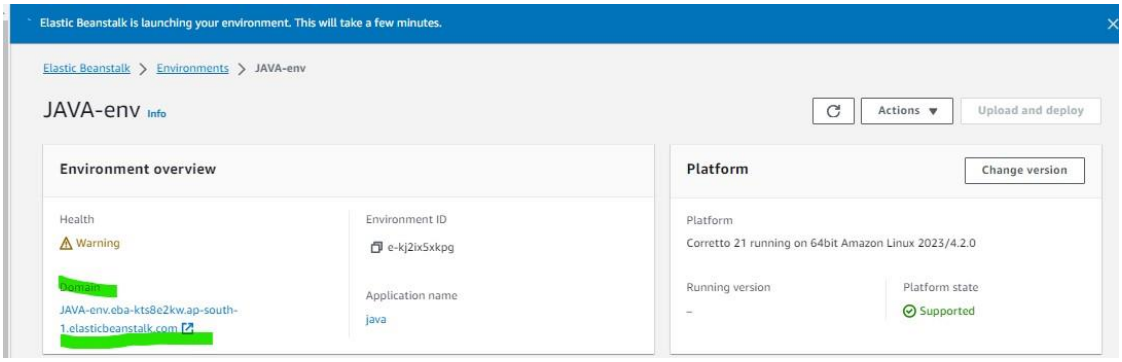

23.You can see it is getting launch



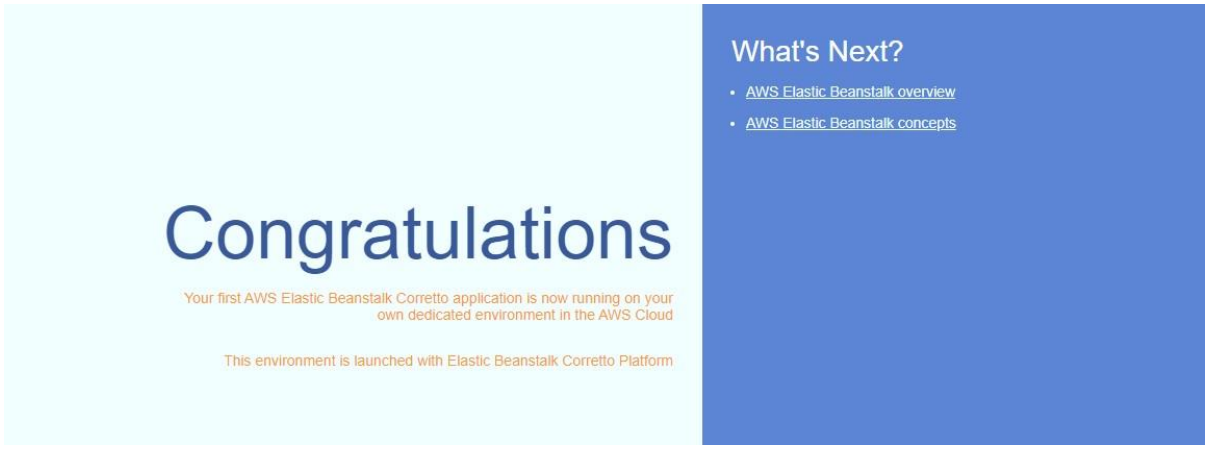

24.Now you can go to EC2 from my service you can see The instance is running.



25.Click on domain

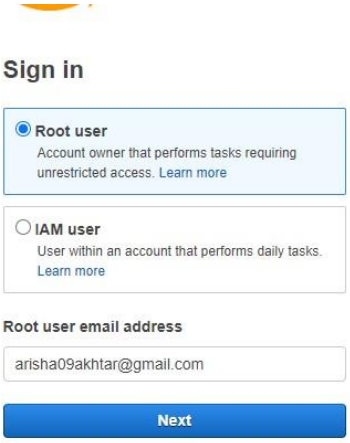



26. It will get open into another brower now Elastic Beanstalk Server is running on your own dedicated environment.

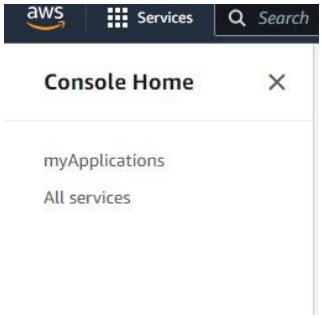



## For Python

1.)Sign In to your aws acc

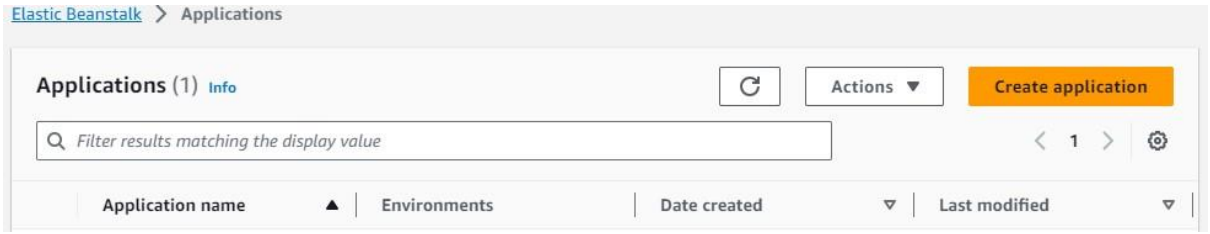



2.)Select all services

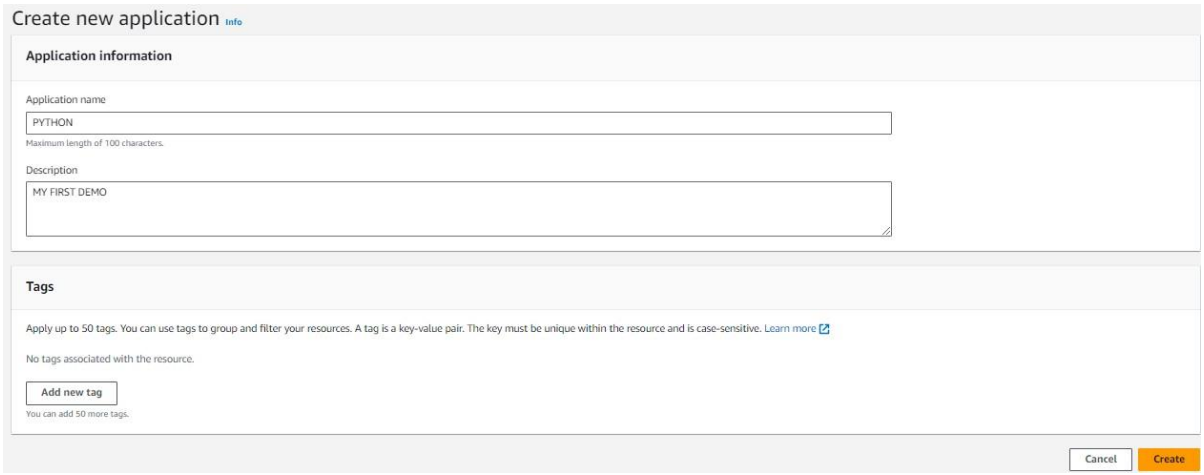



3.)Select Elastic Beanstalk

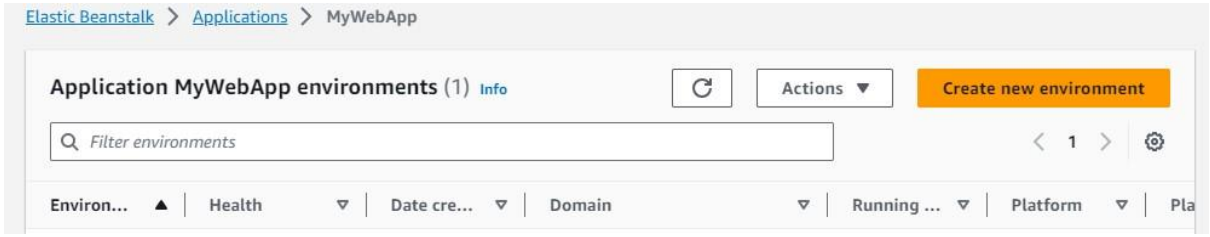

4.) Click on Create Application



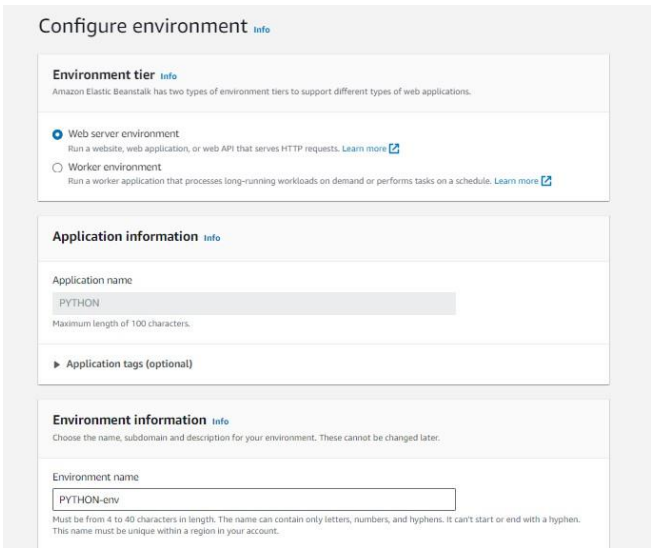

5.) Enter application name and description and then click on create

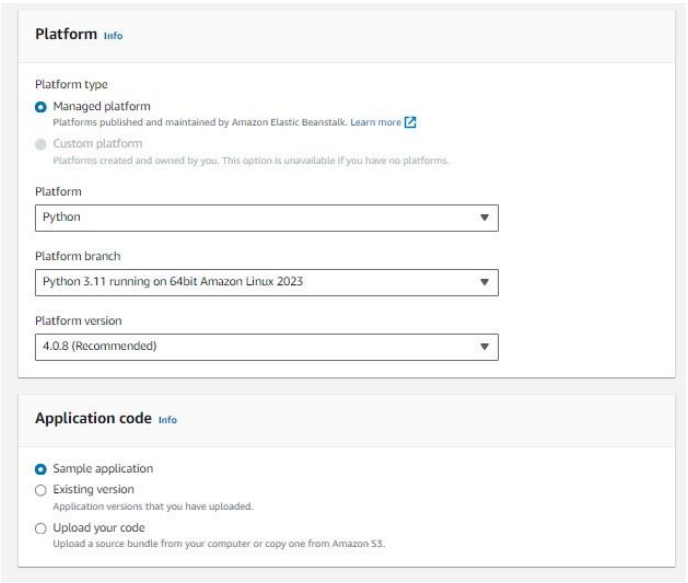



6. Now create environment

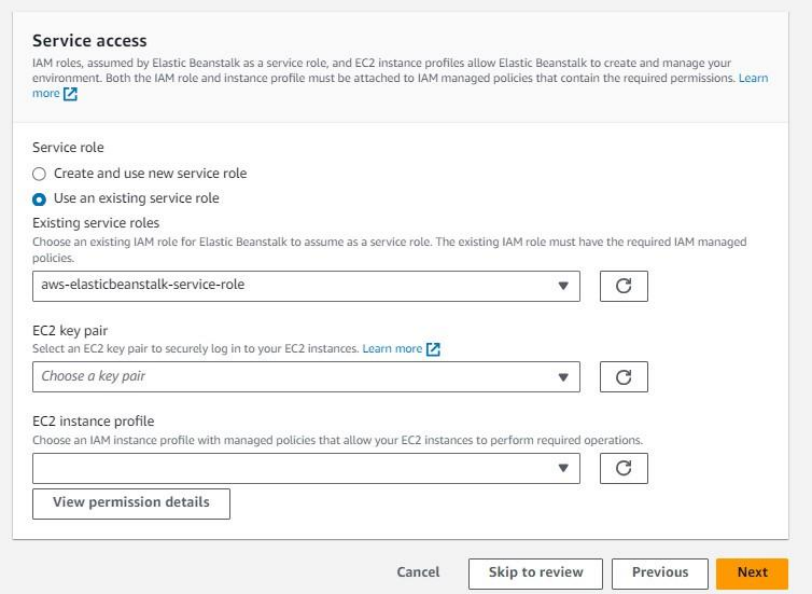



7.For Configure environment everything will be default except platform since we are doing for python we will choose python Platform branch and platform version will be default.Click on next.



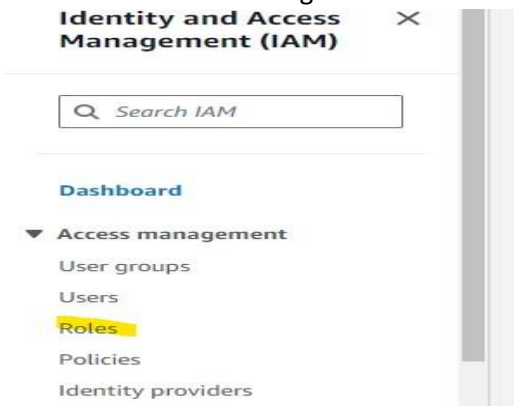




8.) For Service access you have to create a role since it is by default over here you can create your own role .

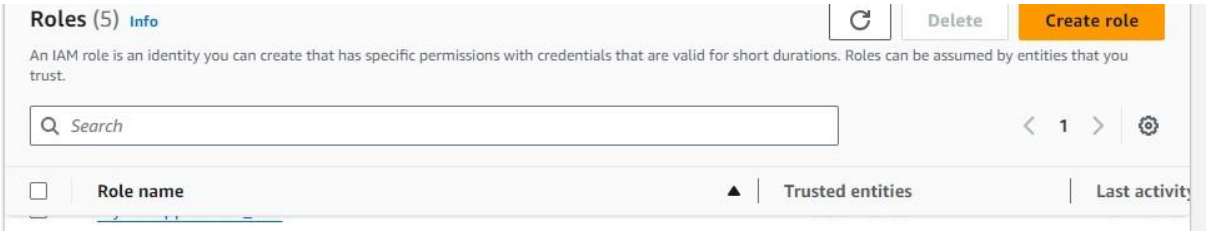



9.)To create role go to services IAM

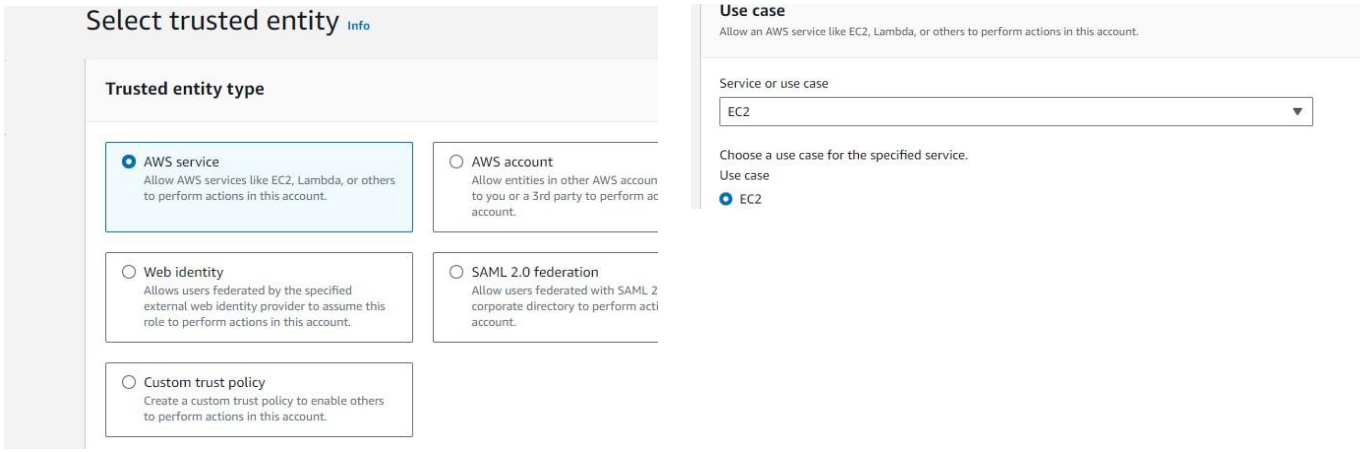

10.Inside IAM dashboard go to roles



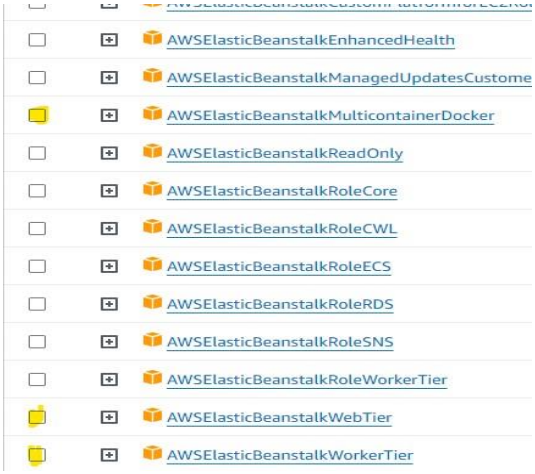

11.Create role




12.It will ask for entity select AWS Service and for use case select EC2 and click on next

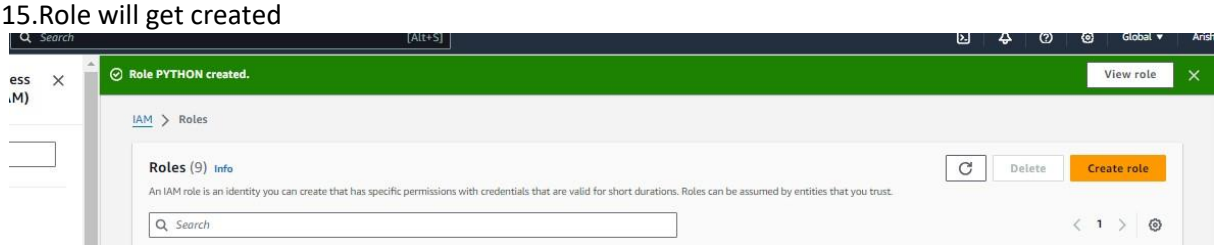



13.Search elasticbeanstalk on top select this three opt and click on next

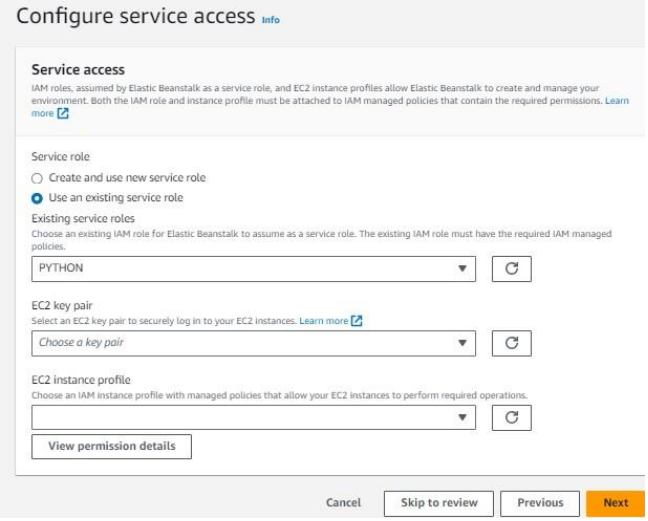



14.Give role name and click on create role

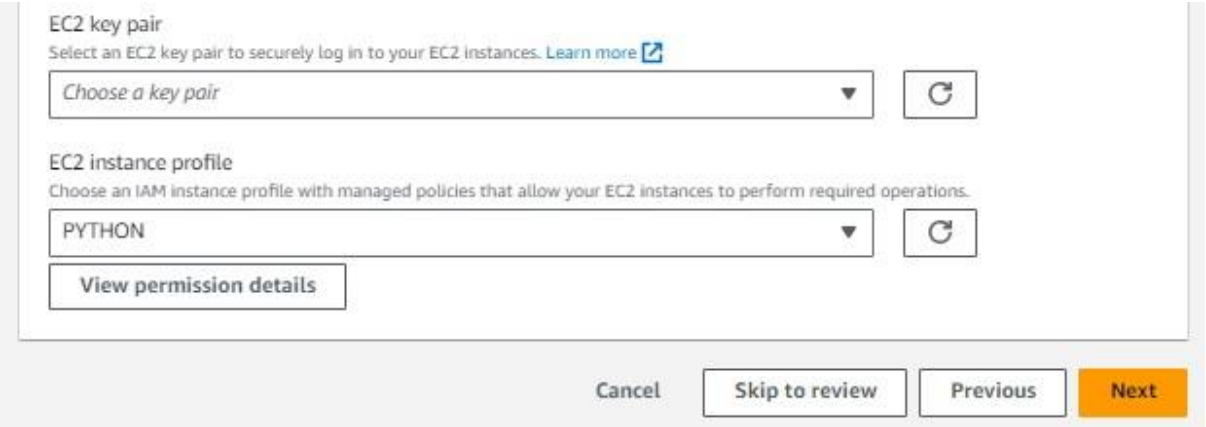

15.Role will get created




16.Now you can again go to your service access that is your 8 step now inside existing service roles refresh it and click on dropdown you can see over there your role name which you have created above automatically comes there click on that

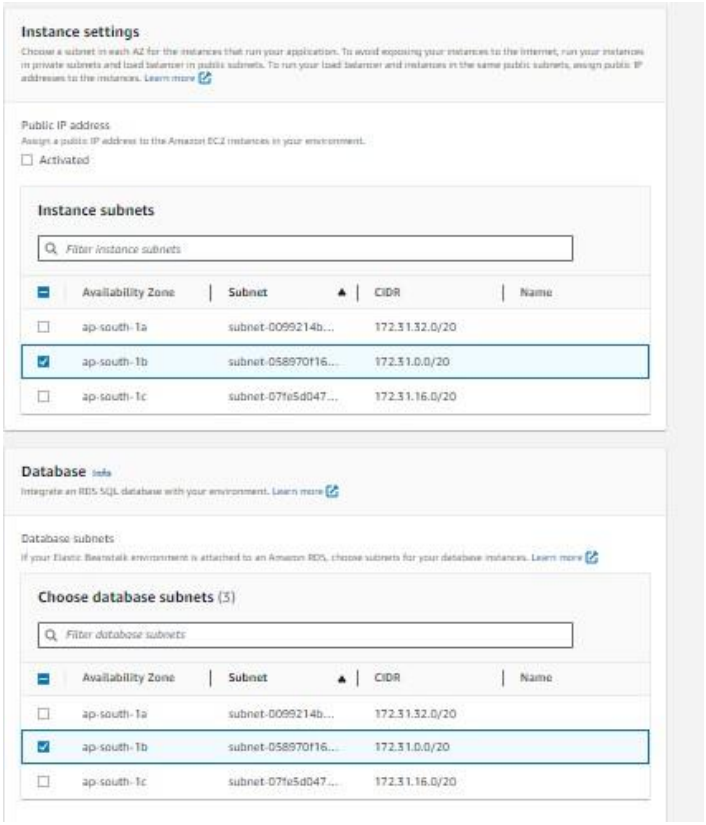



17.For instance profile click on dropbutton your role name will come select that then enter next

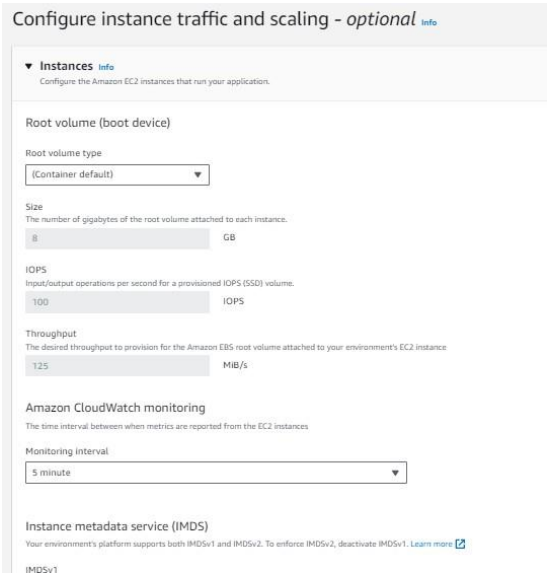



18.After clicking on next it will ask for VPC click on dropdowm and select which is default not down the ip address of this

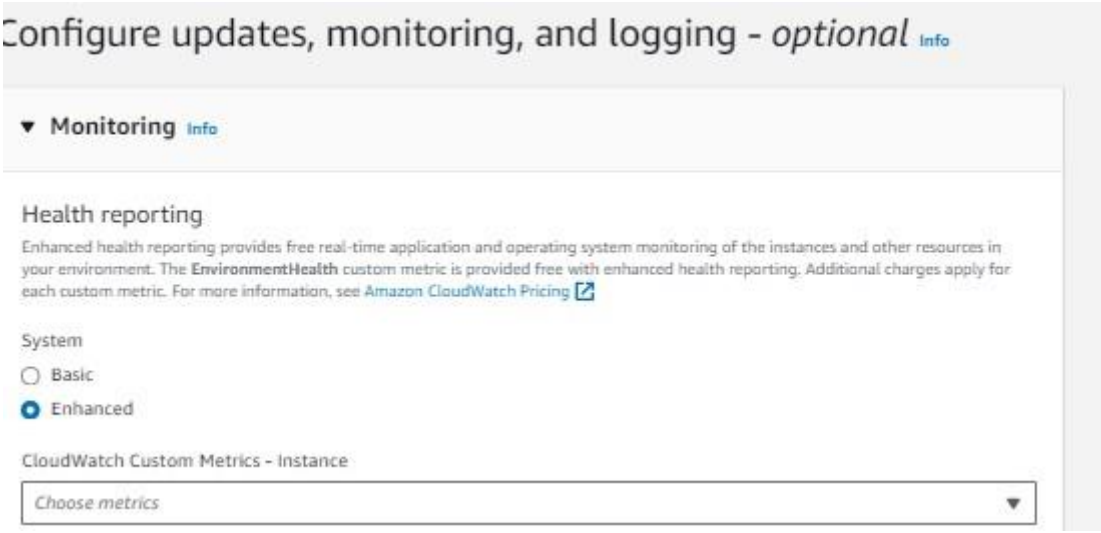

19.The ip address you have selected above same over here select that ip address and click on next



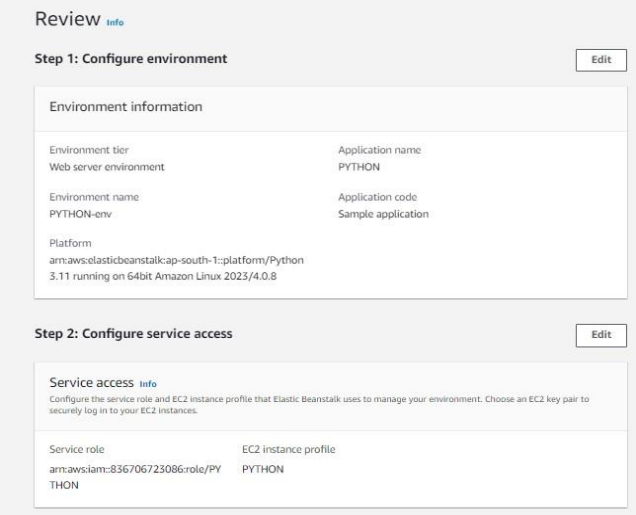

20.for configure instance everything will be by default enter next



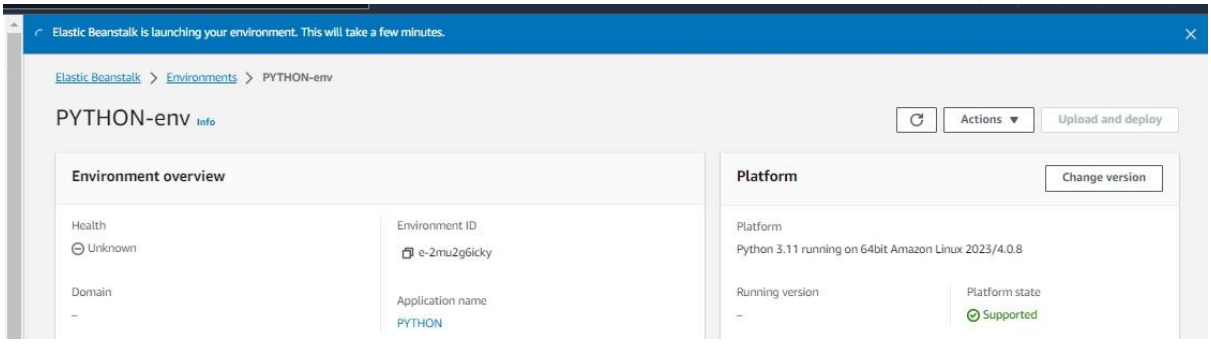

21.For configure update also everthing will be default click to next



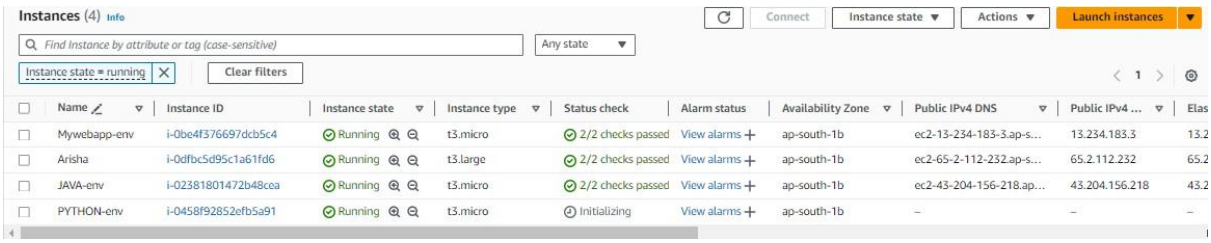

22.Review will display click on submit



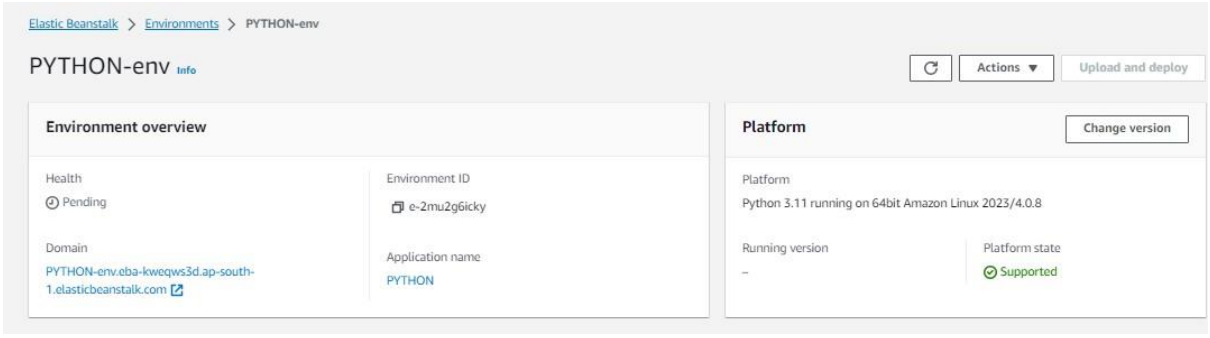

23.You can see it is getting launch



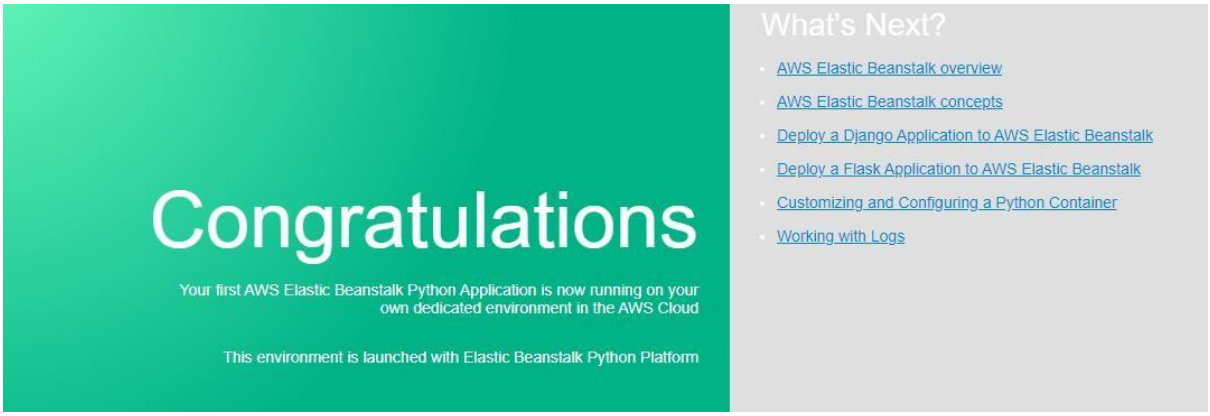

24.Now you can go to EC2 from my service you can see The instance is running.



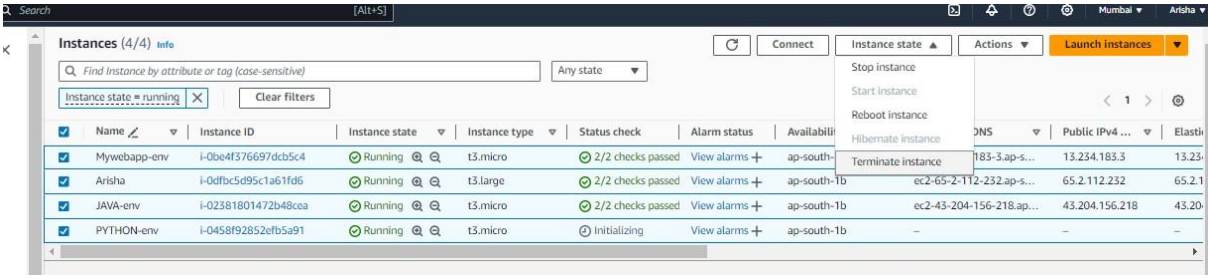

25.Click on domain




26. It will get open into another brower now Elastic Beanstalk Server is running on your own dedicated environment.

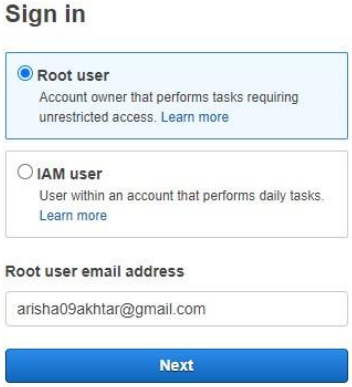



27.Last step don't forget to terminate the instances that have been created

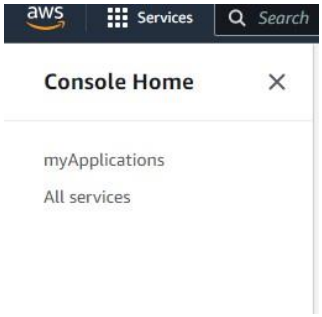



## For Tomcat:

1.)Sign In to your aws acc

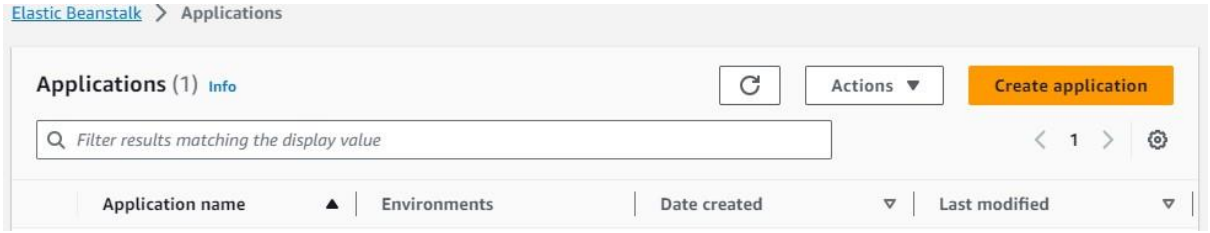


2.)Select all services

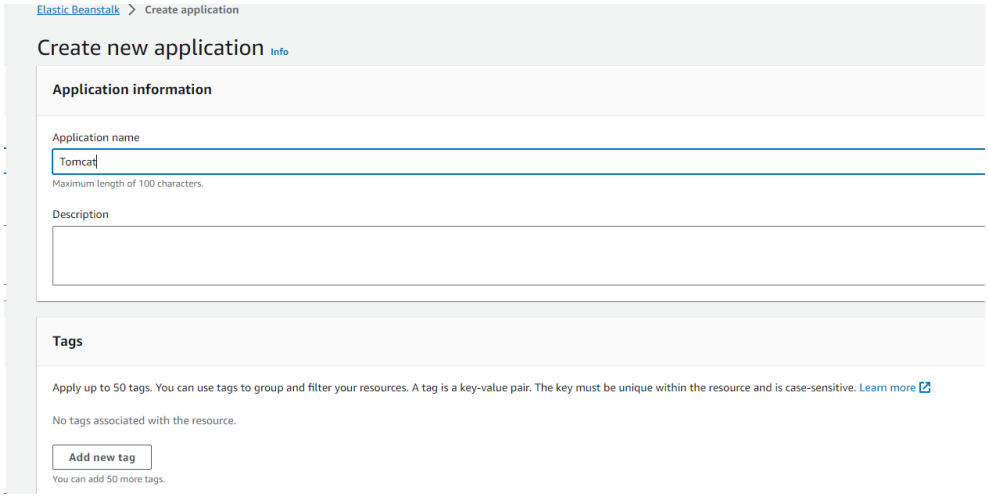



3.)Select Elastic Beanstalk

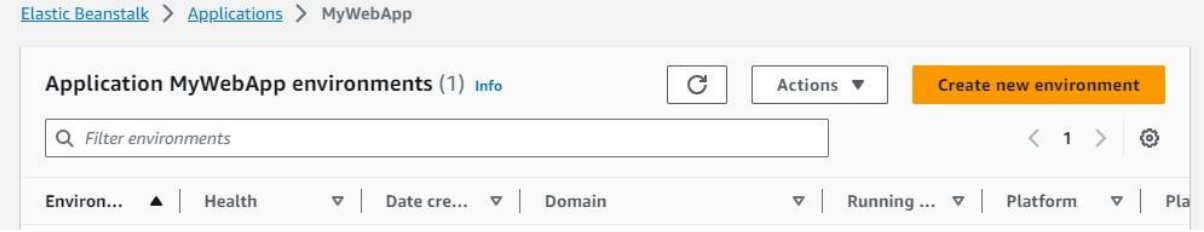

6.) Click on Create Application



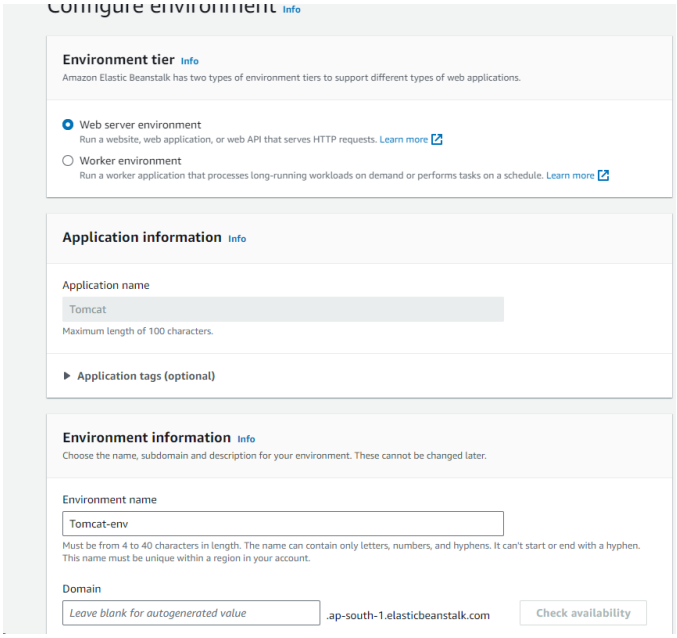

7.) Enter application name and description and then click on create

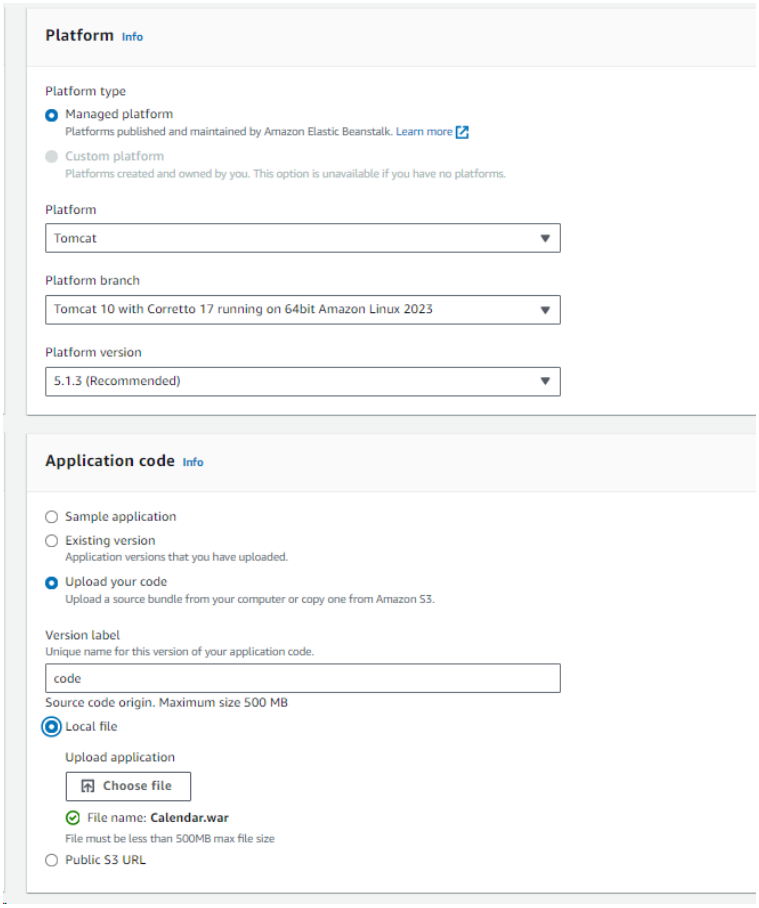


8. Now create environment

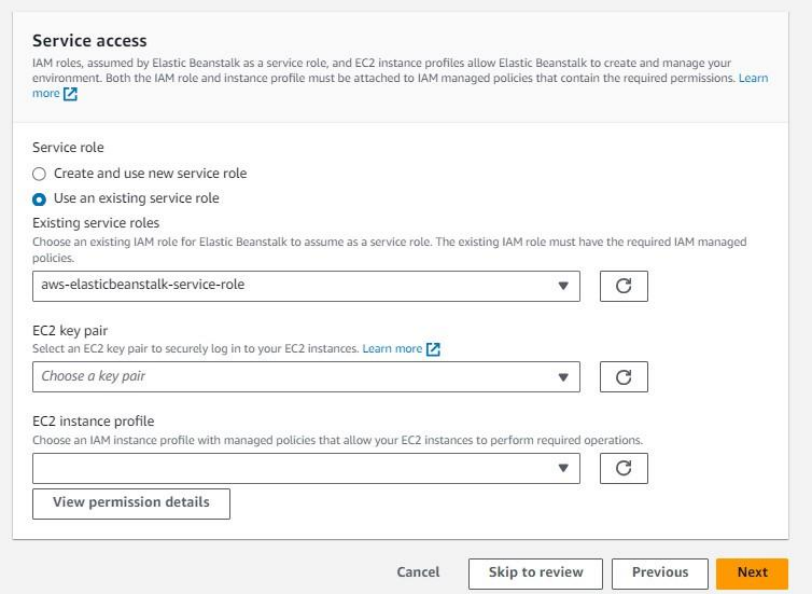



7.For Configure environment everything will be default except platform since we are doing for tomcat we will choose tomcat Platform branch and platform version will be default.Click on next.



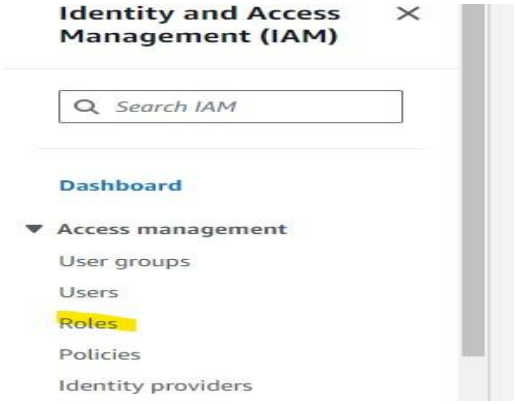



8.) For Service access you have to create a role since it is by default over here you can create your own role .

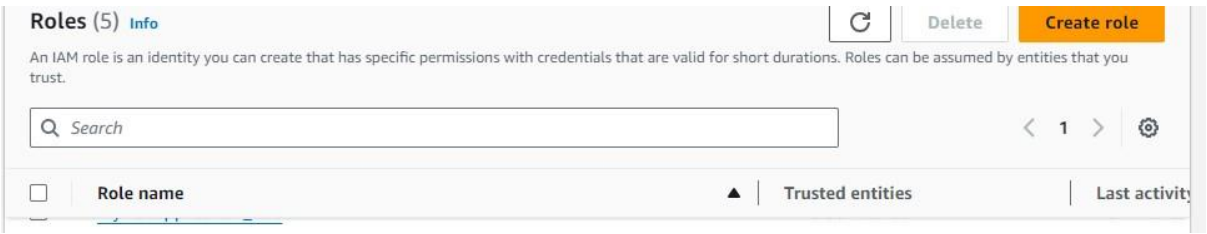



9.)To create role go to services IAM

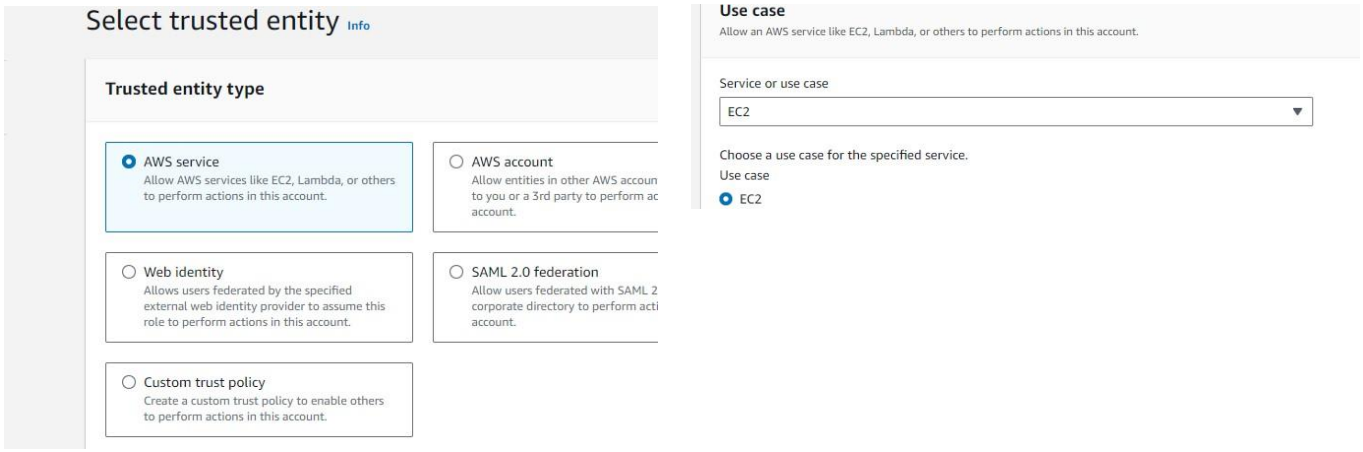

10.Inside IAM dashboard go to roles



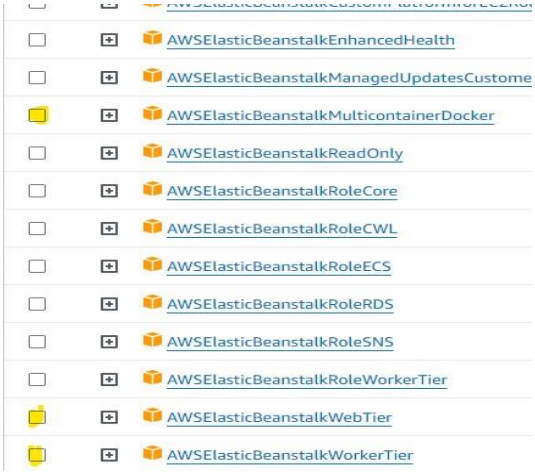

11.Create role




12.It will ask for entity select AWS Service and for use case select EC2 and click on next

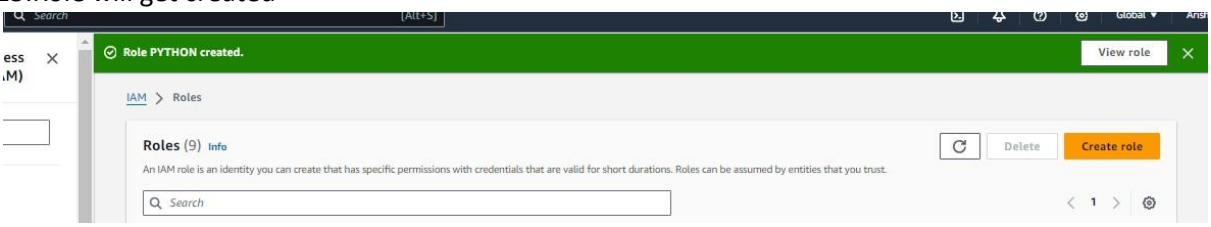



13.Search elasticbeanstalk on top select this three opt and click on next



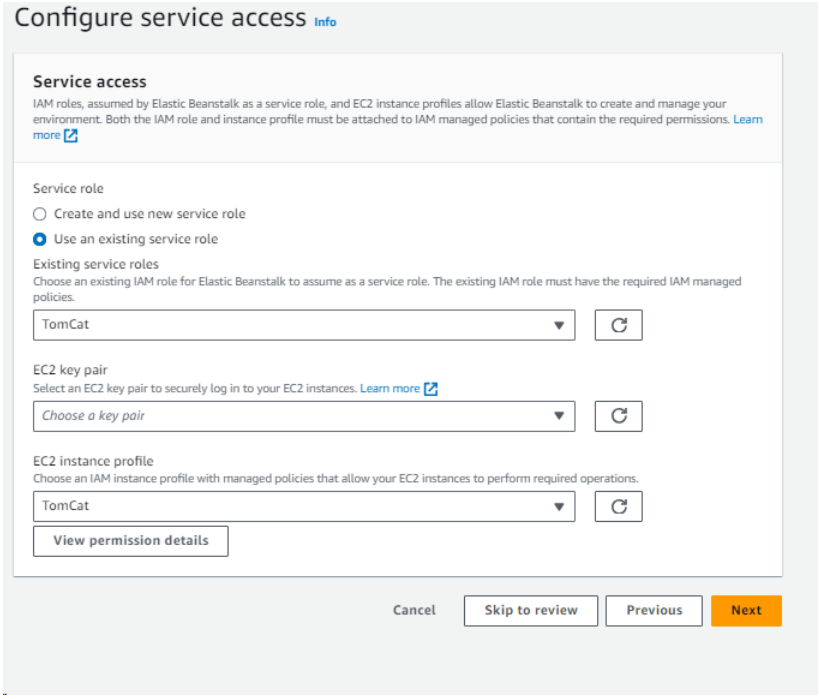

14.Give role name and click on create role


15.Role will get created



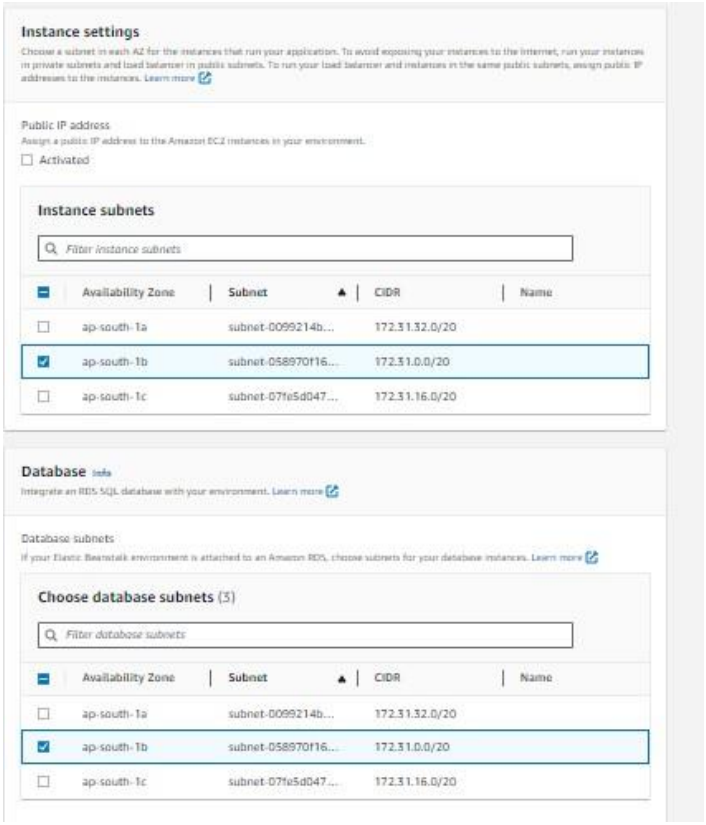

16.Now you can again go to your service access that is your 8 step now inside existing service roles refresh it and click on dropdown you can see over there your role name which you have created above automatically comes there click on that

17.For instance profile click on dropbutton your role name will come select that then enter next

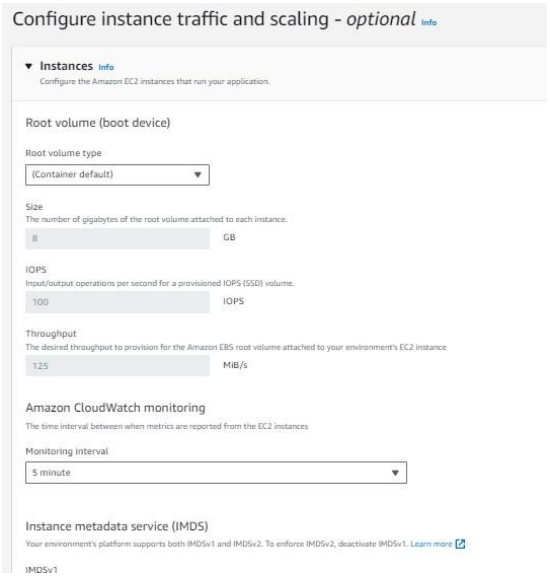



18.After clicking on next it will ask for VPC click on dropdowm and select which is default not down the ip address of this

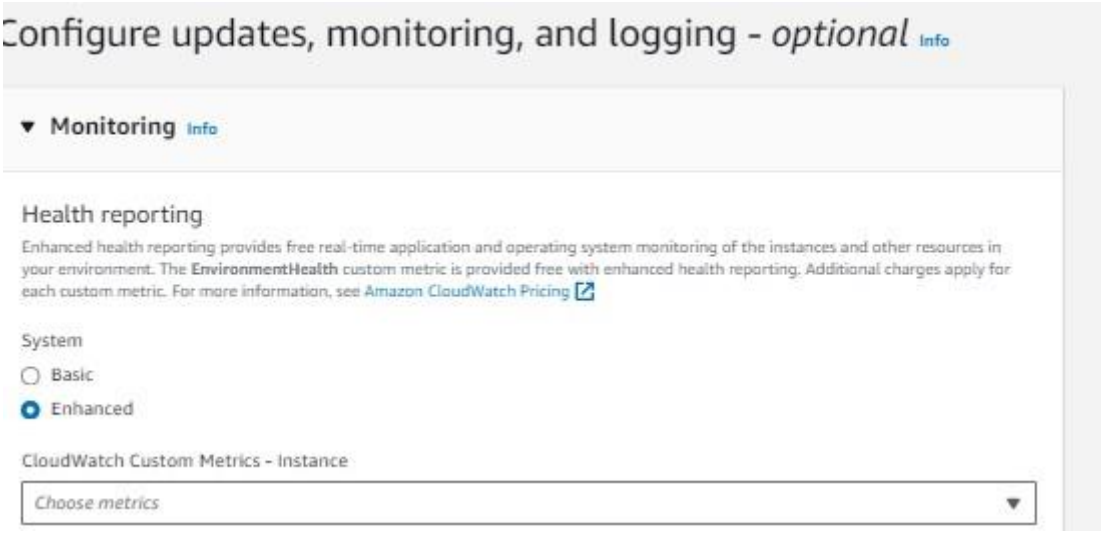

19.The ip address you have selected above same over here select that ip address and click on next



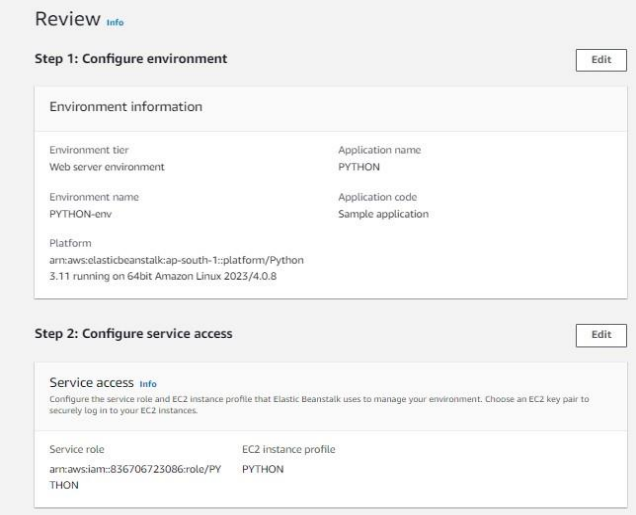

20.for configure instance everything will be by default enter next




21.For configure update also everthing will be default click to next




22.Review will display click on submit




23. It will get open into another brower now Elastic Beanstalk Server is running on your own dedicated environment.