

Vivekanand Education Society's Institute of Technology, Chembur, Mumbai,

Department of Technology,

Year: 2024-2025(ODD SEM)

Advance DevOps Practical Examination

Name: Sanika Ramakant Mehetre

Div: D15B

Roll No: 34

Date of exam: 24/10/2024

Alternative Case Study: Deploying Mini-project on AWS.

Aim: Deployment of mini-project on AWS using Elastic Beanstalk service.

Theory:

AWS Elastic Beanstalk

AWS Elastic Beanstalk is a fully managed service that simplifies the deployment, scaling, and management of web applications and services. It allows developers to upload their application code, and Elastic Beanstalk automatically handles the deployment—from capacity provisioning, load balancing, and auto-scaling to application health monitoring—without the need for deep infrastructure expertise.

How AWS Elastic Beanstalk Works:

- Application Upload:** The user uploads their application code, either as a ZIP file or a container image (in case of Docker).
- Platform Selection:** The user selects the appropriate platform (e.g., Node.js, Python) for their application.
- Environment Creation:** Elastic Beanstalk creates an environment (a collection of AWS resources) based on the uploaded application, platform, and configurations. This includes EC2 instances, security groups, and a load balancer.

4. **Automated Management:** Elastic Beanstalk automatically handles environment provisioning, scaling, and application deployment. It also monitors resource health and makes adjustments if needed.

Implementation:

1. Set up Elastic Beanstalk:

- Go to AWS Elastic Beanstalk and create a new application.
- Choose Node.js as the platform for your backend.

2. Environment Configuration:

- Upload your backend code (Express server).
- Add environment properties like `NODE_ENV`, `DB_URI`, and `PORT`.

3. Frontend (React):

- Build the React app (`npm run build`).
- Serve the frontend via your backend or deploy it separately using AWS S3 (if needed).

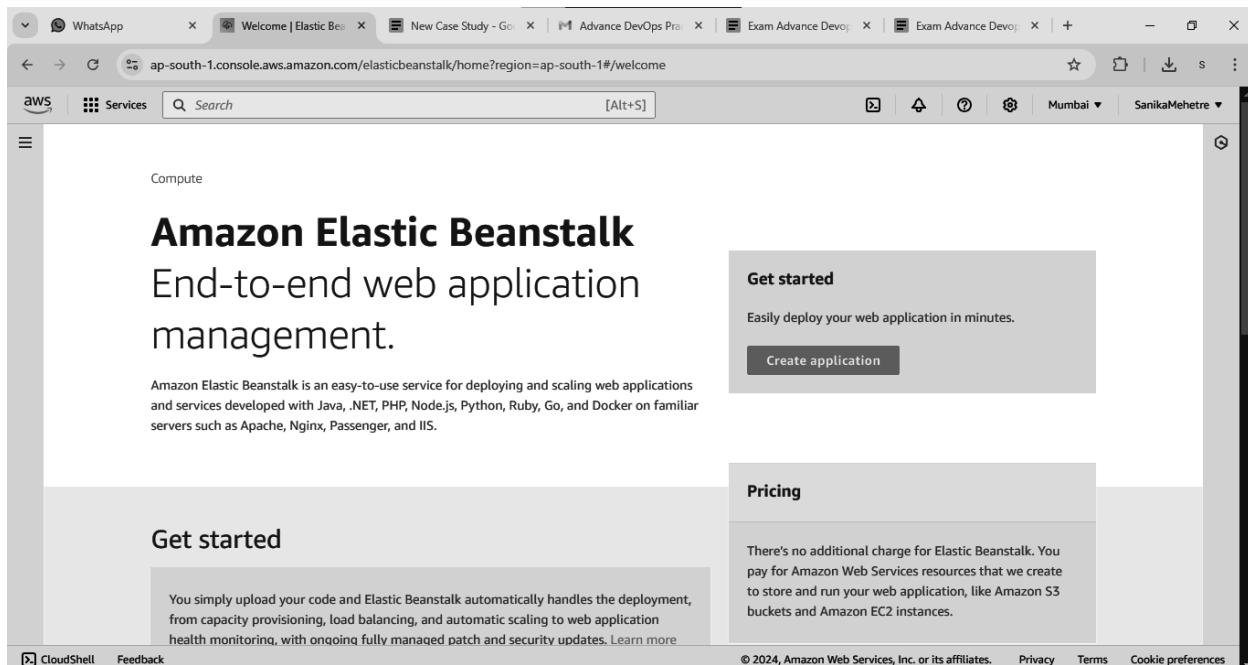
4. Deployment:

- Deploy and monitor through Elastic Beanstalk.

Steps:

Step 1: Set up Elastic Beanstalk

- Open AWS Management Console.
- Navigate to Elastic Beanstalk and click on “Create application”.

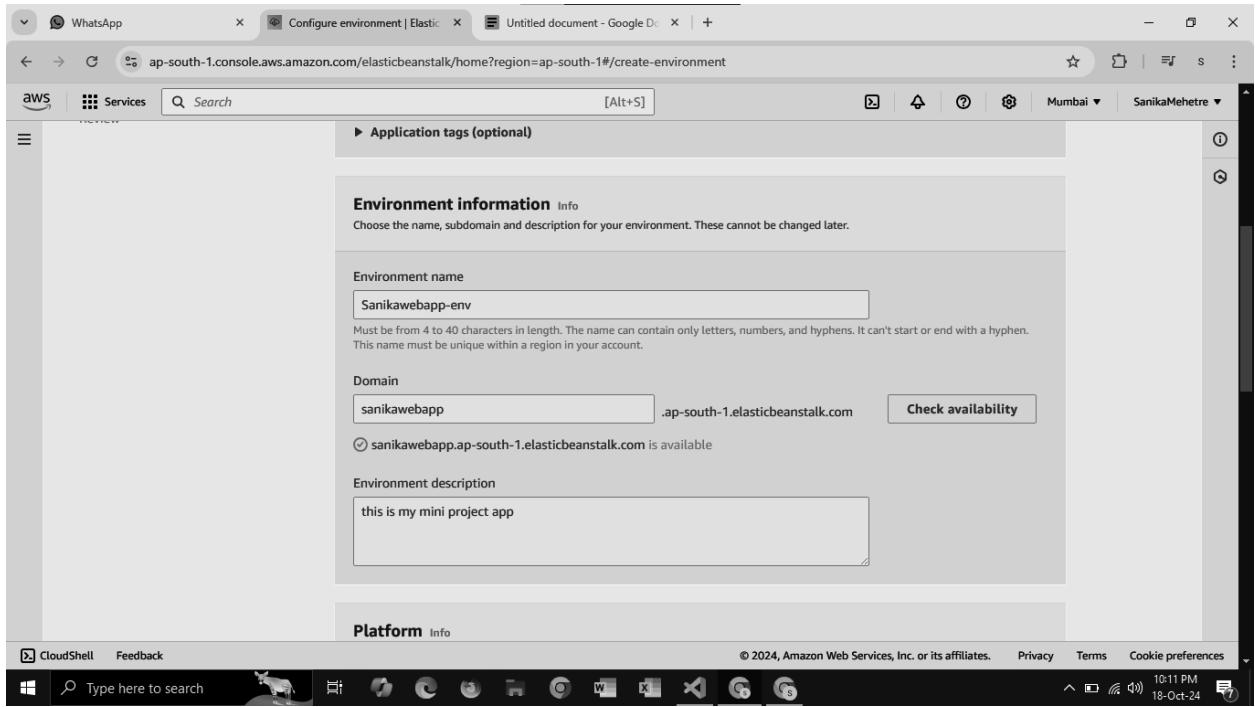


- Then choose the options as shown below and Enter the “Application name” as per your choice.
- Here I’ve named it “sanikawebapp”.

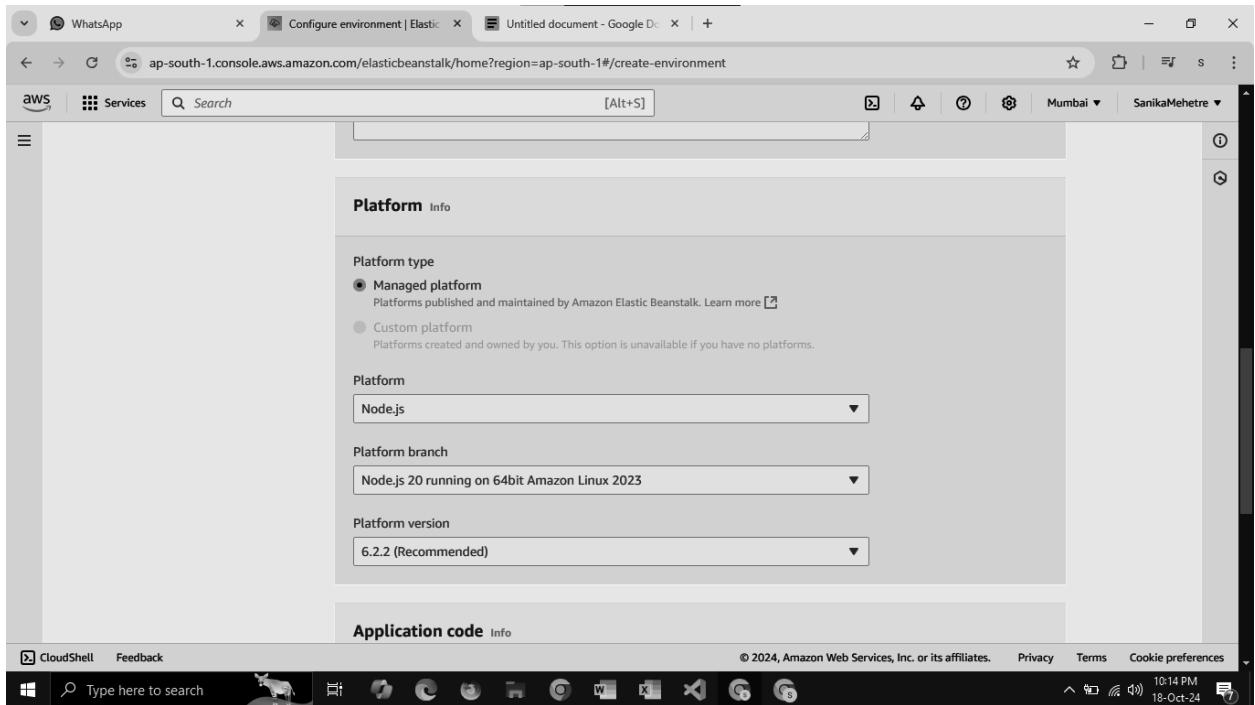
The screenshot shows the 'Configure environment' step in the AWS Elastic Beanstalk console. On the left, a sidebar lists steps: Step 1 (Configure environment), Step 2 (Configure service access), Step 3 (optional: Set up networking, database, and tags), Step 4 (optional: Configure instance traffic and scaling), Step 5 (optional: Configure updates, monitoring, and logging), and Step 6 (Review). The main area is titled 'Configure environment' and contains two sections: 'Environment tier' and 'Application information'. In the 'Environment tier' section, 'Web server environment' is selected. In the 'Application information' section, the 'Application name' field is filled with 'sanikawebapp'. The status bar at the bottom indicates the date and time as '18-Oct-24 10:10 PM'.

This screenshot provides a detailed view of the 'Application information' section. It shows the 'Application name' field containing 'sanikawebapp' and a note that the maximum length is 100 characters. Below this, there is a section titled 'Application tags (optional)' which is currently collapsed.

- The environment name is selected by default.
- You can choose the “domain name” as per your choice and check its availability.
- Here I’ve named it as “sanikawebapp” and add the description if you want.



- Select the “Platform” settings very carefully as it will vary for different projects.
- My project is made using MERN stack so I’ve selected “Node.js”.
- You can select the platform depending on your project.



- Leave the others settings as default.

Application code Info

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.

Version label

Unique name for this version of your application code.

v1

Source code origin. Maximum size 500 MB

Local file

Upload application

File name: **sanikaapi.zip**

File must be less than 500MB max file size

Public S3 URL

Not backed up api

File Home Share View Backup Tools

This PC > Windows (C:) > Users > DELL > Desktop > EEB_Proj > eeb-new > api

Name	Date modified	Type	Size
controllers	18-Oct-24 9:56 PM	File folder	
lib	18-Oct-24 9:56 PM	File folder	
middleware	18-Oct-24 9:56 PM	File folder	
node_modules	18-Oct-24 9:56 PM	File folder	
prisma	18-Oct-24 9:56 PM	File folder	
routes	18-Oct-24 9:56 PM	File folder	
env	19-Oct-24 10:00 PM	File	
.gitignore	18-Oct-24 9:56 PM	File	
app	19-Oct-24 10:00 PM	File	
package	19-Oct-24 10:00 PM	File	
package-lock	19-Oct-24 10:00 PM	File	
Procfile	19-Oct-24 10:00 PM	File	

Archive name and parameters

General Advanced Options Files Backup Time Comment

Archive name: **sanikaapi.zip**

Default Profile: Profiles... Update mode: Add and replace files

Archive format: ZIP RAR RAR4

Compression method: Normal

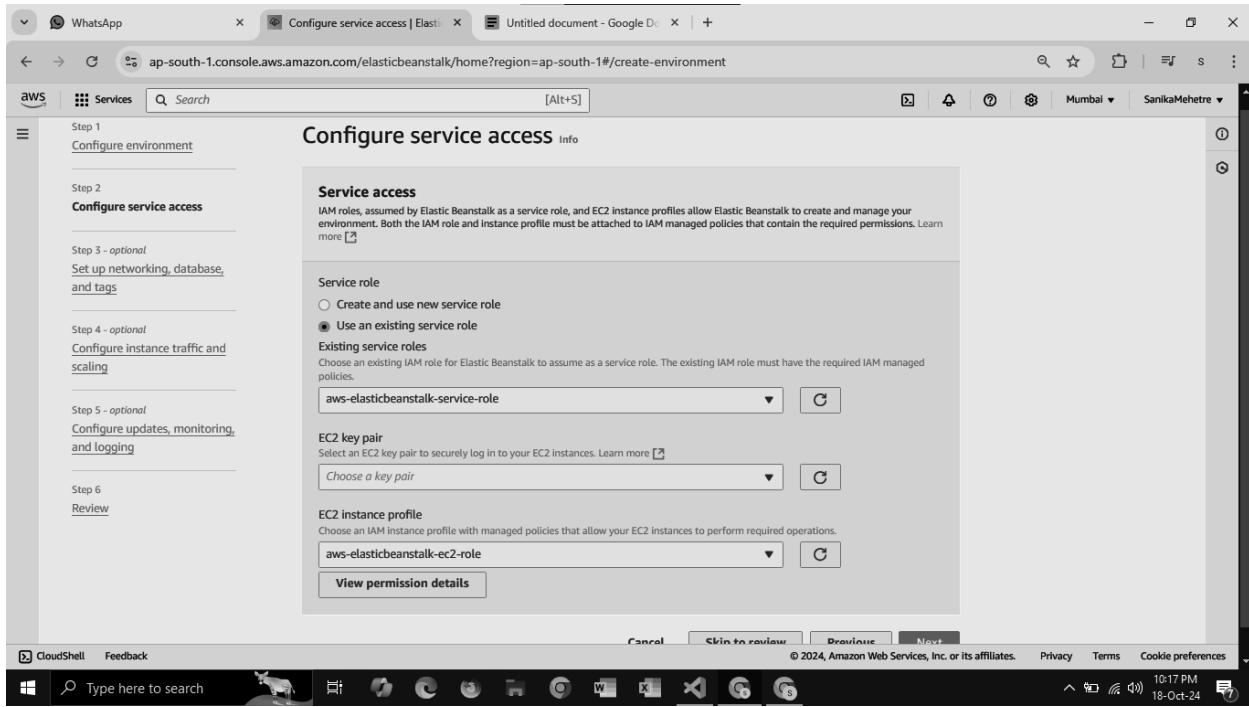
Dictionary size: 32 KB

Split to volumes, size: MB

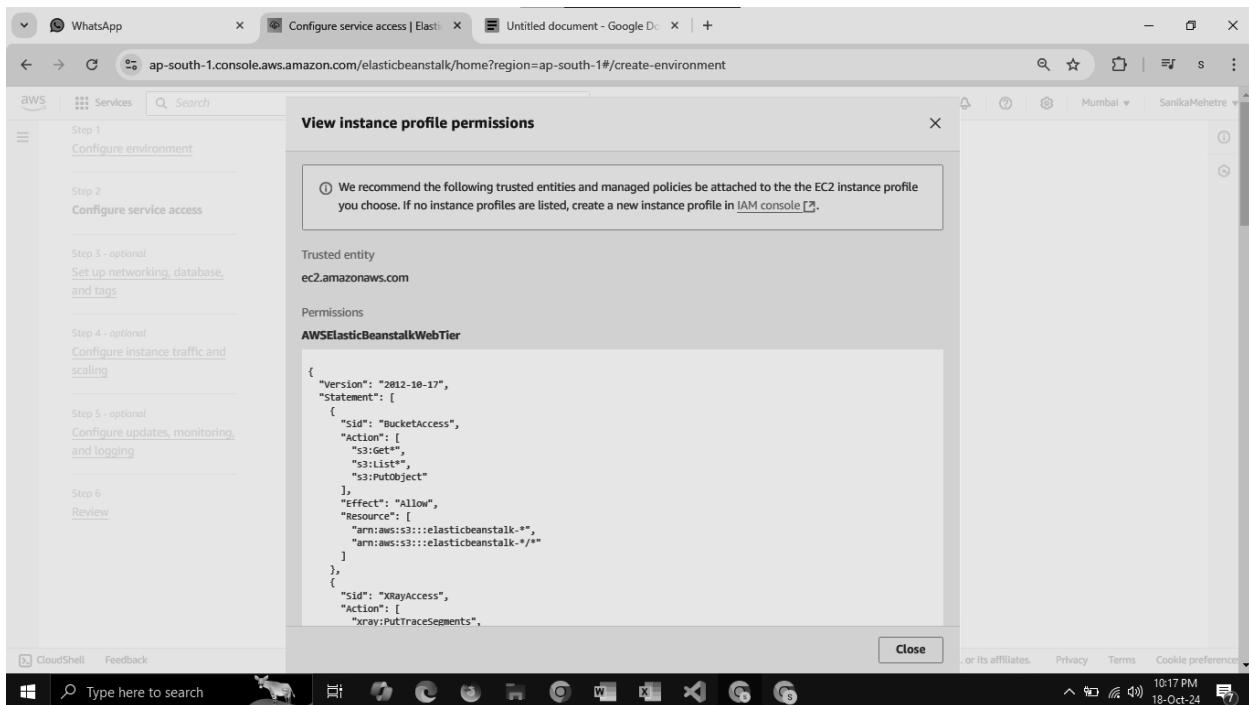
Archiving options:

- Delete files after archiving
- Create SFX archive
- Create solid archive
- Add recovery record
- Test archived files
- Lock archive

OK Cancel Help



- Here we need to create a new EC2 instance profile. Click on "View permission details" and the below page will open.



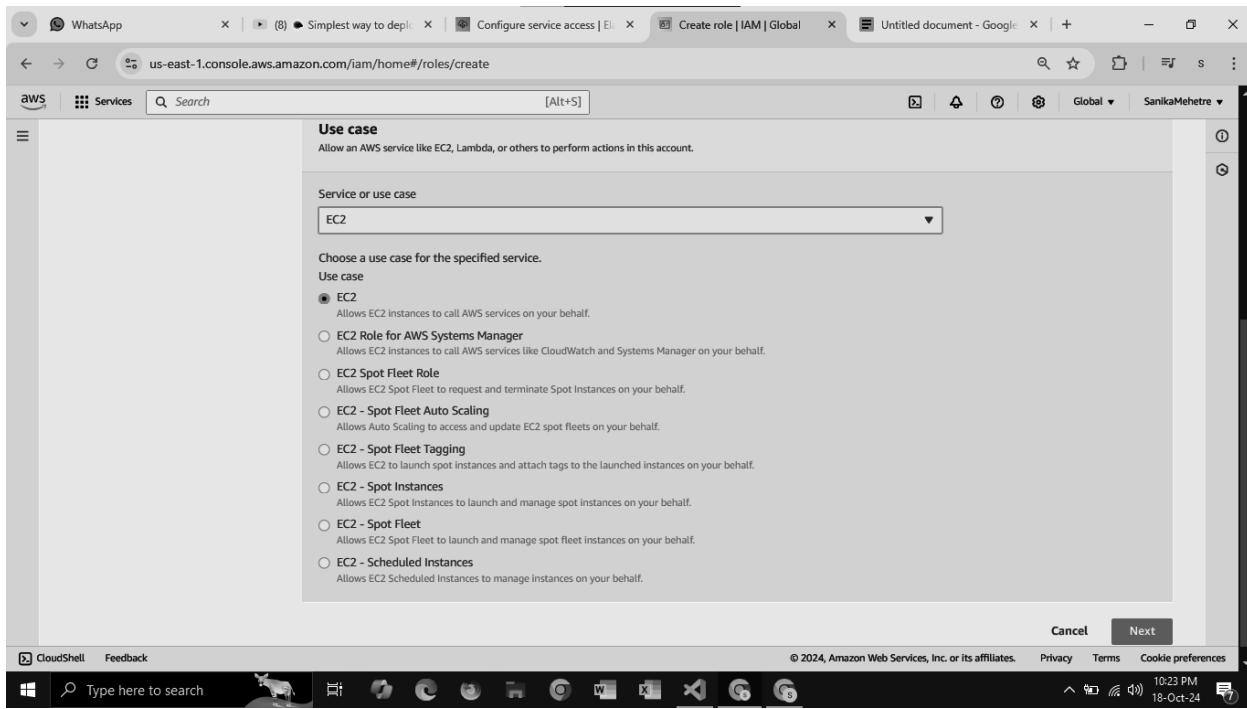
- Click on "IAM Console" and it'll redirect to IAM Console page.

- Select “AWS Service”.

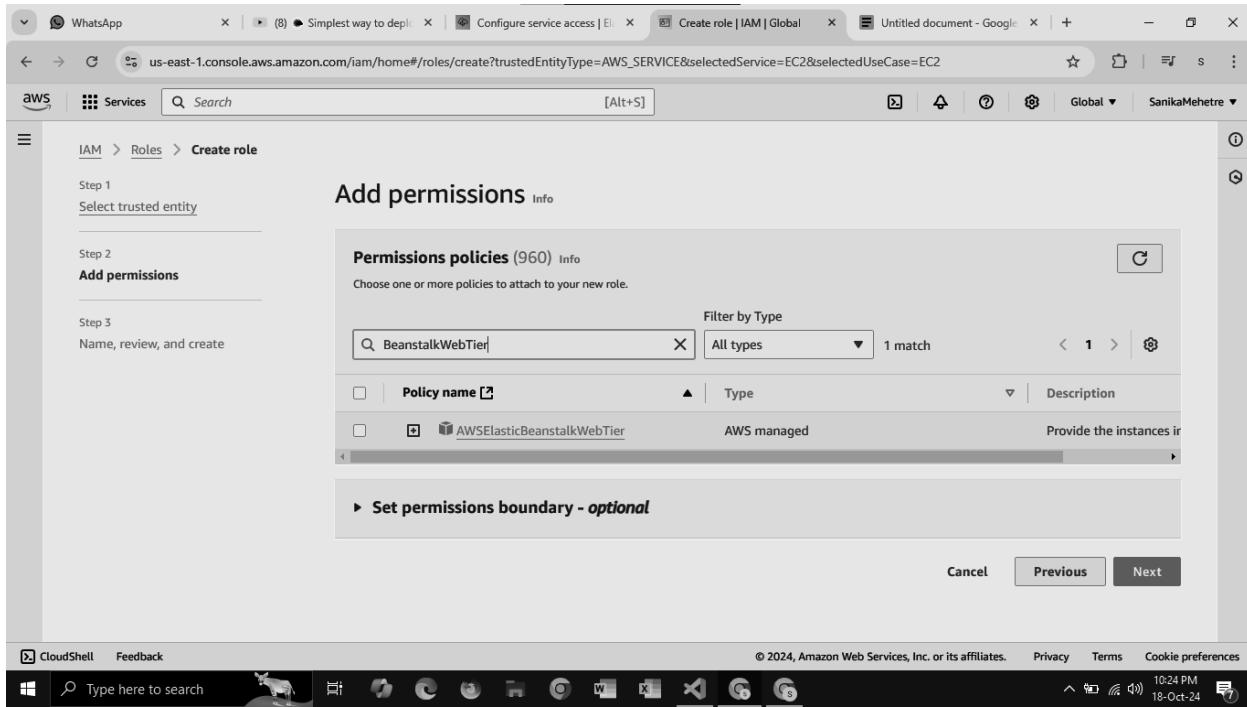
The screenshot shows the AWS IAM 'Create role' wizard. The current step is 'Step 1 Select trusted entity'. On the left, there's a navigation pane with 'Step 1 Select trusted entity', 'Step 2 Add permissions', and 'Step 3 Name, review, and create'. The main area is titled 'Select trusted entity' with a 'Trusted entity type' section. It lists five options: 'AWS service' (selected), 'AWS account', 'Web identity', 'SAML 2.0 federation', and 'Custom trust policy'. Each option has a brief description below it. At the bottom right of the main area, there are 'Previous' and 'Next' buttons, and a note: 'You can always go back to previous steps by clicking the back arrow in the top-left corner of the browser window.'

- Under “Service or Use case” select EC2.

The screenshot shows the AWS IAM 'Create role' wizard. The current step is 'Step 2 Add permissions'. On the left, there's a navigation pane with 'Step 1 Select trusted entity', 'Step 2 Add permissions' (selected), and 'Step 3 Name, review, and create'. The main area has a search bar 'Filter service or use case' and a dropdown menu titled 'Commonly used services'. The 'EC2' service is selected in the dropdown. Other services listed include Lambda, Other services (Amazon EMR Serverless, Amazon OpenSearch Service, Amazon Q Business, Amazon Grafana, Amplify, API Gateway, AppFabric, Application Auto Scaling, Application Discovery Service), and a 'Choose a service or use case' input field. At the bottom right, there are 'Cancel' and 'Next' buttons, and a note: 'You can always go back to previous steps by clicking the back arrow in the top-left corner of the browser window.'



- Click on “Next”.
- Now we have to add permissions to our new role.
 1. AWS>ElasticBeanstalkWebTier



```

    "arn:aws:elasticbeanstalk::*:application/*",
    "arn:aws:elasticbeanstalk::*:environment/*"
}

]

}

AWSElasticBeanstalkWorkerTier

{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "MetricsAccess",
      "Action": [
        "cloudwatch:PutMetricData"
      ],
      "Effect": "Allow",
      "Resource": "*"
    },
    {
      "Sid": "XrayAccess",
      "Action": [
        "xray:PutTraceSegments",
        "xray:PutTelemetryRecords",
        "xray:GetSamplingRules",
        "xray:GetSamplingTargets",
        "xray:GetSamplingStatisticSummaries"
      ],
      "Effect": "Allow",
      "Resource": "*"
    },
    {
      "Sid": "QueueAccess",
      "Action": [
        "SQS:SendMessage",
        "SQS:ReceiveMessage",
        "SQS:DeleteMessage"
      ],
      "Effect": "Allow",
      "Resource": "*"
    }
  ]
}
  
```

CloudShell Feedback

Privacy Terms Cookie preferences 10:24 PM 18-Oct-24

2. AWSElasticBeanstalkWorkerTier

Add permissions

Permissions policies (2/960)

Choose one or more policies to attach to your new role.

Policy name	Type	Description
<input checked="" type="checkbox"/> AWSElasticBeanstalkWorkerTier	AWS managed	Provide the instances for the worker tier

Set permissions boundary - optional

Cancel Previous Next

CloudShell Feedback © 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences 10:25 PM 18-Oct-24

```

    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ECSAccess",
            "Effect": "Allow",
            "Action": [
                "ecs:Poll",
                "ecs:StartTask",
                "ecs:StopTask",
                "ecs:DiscoverPollEndpoint",
                "ecs:StartTelemetrySession",
                "ecs:RegisterContainerInstance",
                "ecs:DeregisterContainerInstance",
                "ecs:DescribeContainerInstances",
                "ecs:Submit"
            ],
            "Resource": "*"
        }
    ]
}

```

3. AWSElasticBeanstalkMulticontainerDocker

Add permissions

Permissions policies (3/960)

Choose one or more policies to attach to your new role.

Policy name	Type	Description
AWSElasticBeanstalkMulticontainerDocker	AWS managed	Provide the instances ir...

Set permissions boundary - optional

Cancel Previous Next

- Click on “Next”.
- Now you’ve to name your role. I have named it “ec2-beanstalk-manuall”.

Name, review, and create

Role details

Role name
Enter a meaningful name to identify this role.
ec2-beanstalk-manuall

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

Description
Add a short explanation for this role.
Allows EC2 instances to call AWS services on your behalf.

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

Step 1: Select trusted entities

Trust policy

```
[{"Version": "2012-10-17", "Statement": [{"Effect": "Allow", "Action": ["sts:AssumeRole"], "Principal": {"Service": ["ec2.amazonaws.com"]}}]}
```

Step 1: Select trusted entities

Trust policy

```
[{"Version": "2012-10-17", "Statement": [{"Effect": "Allow", "Action": ["sts:AssumeRole"], "Principal": {"Service": ["ec2.amazonaws.com"]}}]}
```

Step 2: Add permissions

Permissions policy summary

Policy name	Type	Attached as

- You can see that the permissions we added are now listed here.

Step 2: Add permissions

Policy name	Type	Attached as
AWSElasticBeanstalkMulticontainerDocker	AWS managed	Permissions policy
AWSElasticBeanstalkWebTier	AWS managed	Permissions policy
AWSElasticBeanstalkWorkerTier	AWS managed	Permissions policy

Step 3: Add tags

Add tags - optional Info
Tags are key-value pairs that you can add to AWS resources to help identify, organize, or search for resources.

No tags associated with the resource.

Add new tag

You can add up to 50 more tags.

- Click on “Create Role”.

Step 3: Add tags

Add tags - optional Info
Tags are key-value pairs that you can add to AWS resources to help identify, organize, or search for resources.

No tags associated with the resource.

Add new tag

You can add up to 50 more tags.

Create role

- The role has been successfully created.

The screenshot shows the AWS IAM Roles page. A success message at the top left says "Role ec2-beanstalk-manual created." The main area displays a table of roles with one item: "ec2-beanstalk-manual". The table includes columns for "Role name", "Trusted entities", and "Last activity". Below the table, there are two sections: "Access AWS from your non AWS workloads" and "X.509 Standard". The "Temporary credentials" section is also visible. The left sidebar lists various IAM management options like User groups, Users, Roles, Policies, Identity providers, and Account settings.

- Now navigate back to AWS Management Console.
- Refresh “EC2 Instance Profile” and now you will be able to see that our new role is listed.

The screenshot shows the "Configure service access" step of the "Create environment" wizard. On the left, a sidebar lists steps 3 through 6. Step 3 is "optional" and includes "Set up networking, database, and tags". Step 4 is "optional" and includes "Configure instance traffic and scaling". Step 5 is "optional" and includes "Configure updates, monitoring, and logging". Step 6 is "Review". The main panel describes how IAM roles and EC2 instance profiles work. It shows the "Service role" section where "Use an existing service role" is selected, and the role "aws-elasticbeanstalk-service-role" is chosen. The "EC2 key pair" section shows "aws-elasticbeanstalk-ec2-role" selected. The "View permission details" button is at the bottom. Navigation buttons at the bottom right include "Cancel", "Skip to review", "Previous", and "Next".

- Click on “Next”.
- Select a VPC as suggested.

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*

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.

vpc-00b7080d7ec583dca | (172.31.0.0/16)

vpc-00b7080d7ec583dca | (172.31.0.0/16)

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

- Select the number of subnets as you wish.
- Here I have selected two subnets.

Configure instance traffic and scaling

Step 5 - optional
Configure updates, monitoring, and logging

Step 6
Review

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

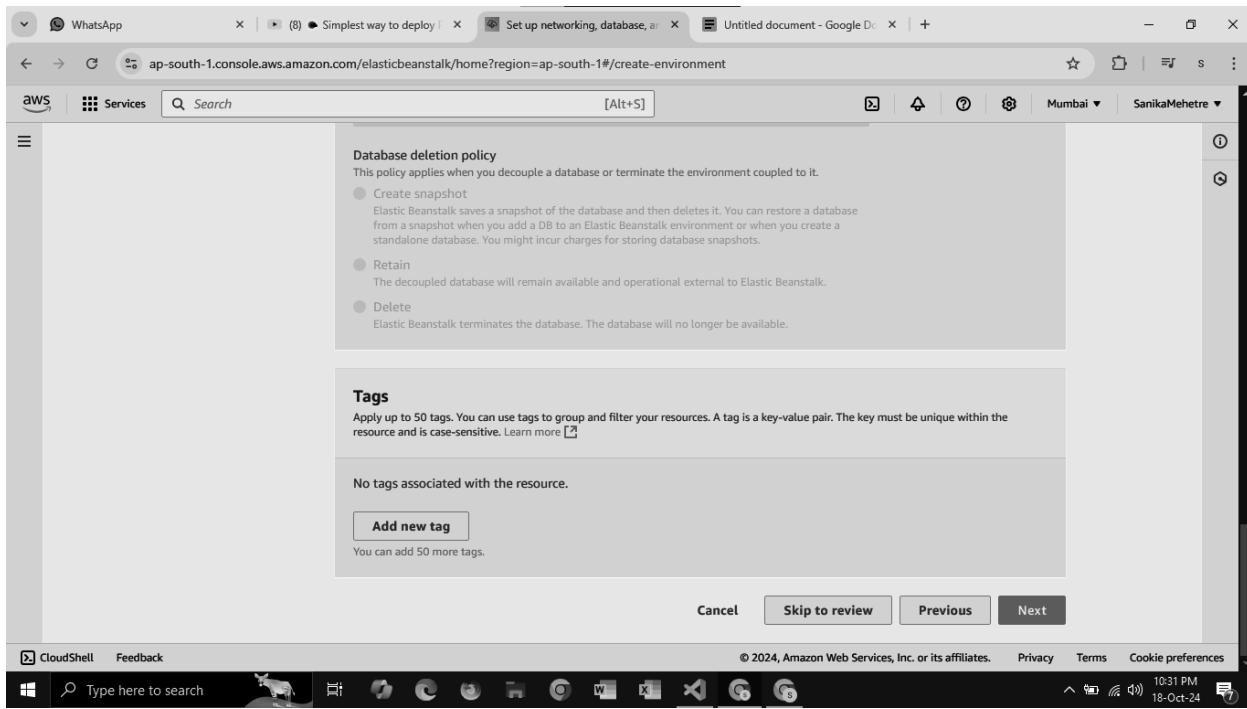
Public IP address

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

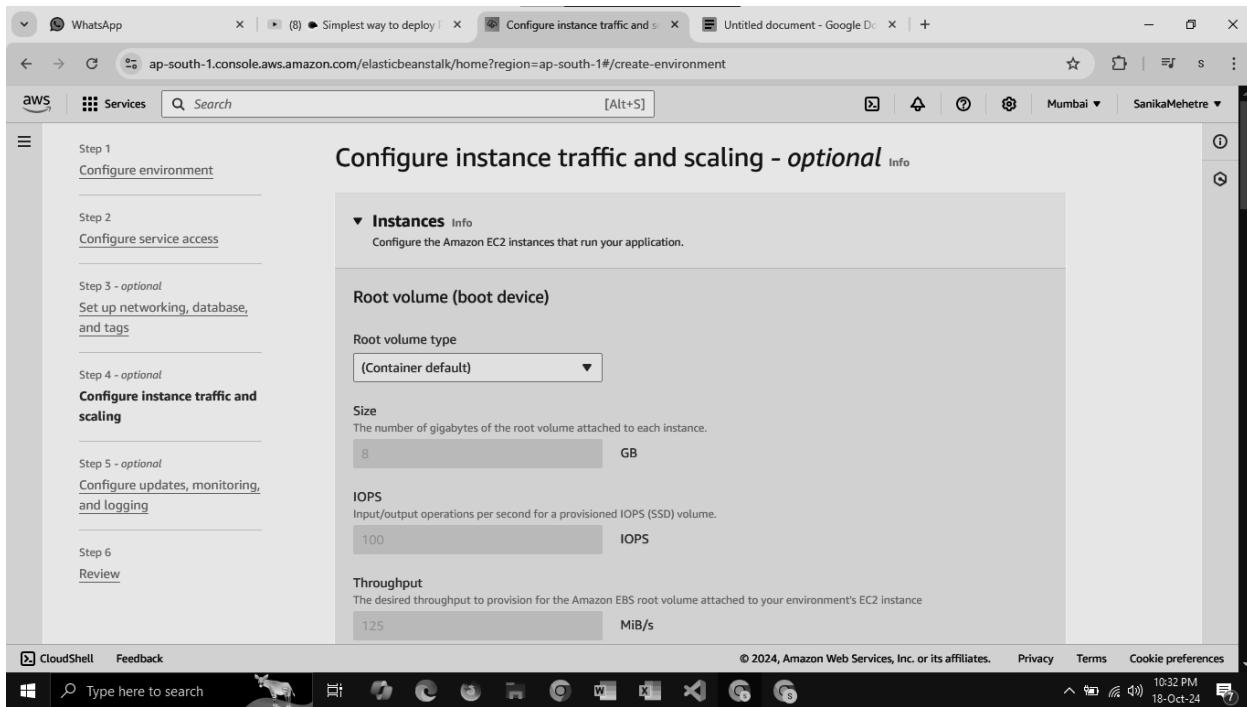
Activated

Instance subnets

	Availability Zone	Subnet	CIDR	Name
<input checked="" type="checkbox"/>	ap-south-1c	subnet-09255a1da...	172.31.16.0/20	
<input checked="" type="checkbox"/>	ap-south-1b	subnet-0d22aa496...	172.31.0.0/20	
<input type="checkbox"/>	ap-south-1a	subnet-0e078e032...	172.31.32.0/20	



- Click on “Next”.
- Keep these settings as shown



The screenshot shows the AWS Elastic Beanstalk configuration interface for creating a new environment. The 'CloudWatch monitoring' section is visible, with the 'Monitoring interval' set to '5 minute'. The 'IMDSv1' section shows that IMDSv1 is deactivated. The 'EC2 security groups' section lists eight security groups, with the 'default' group selected. The interface includes standard browser navigation and search bars at the top.

- Select the “default” security group.

The screenshot shows the AWS EC2 Security Groups list. The 'default' security group is selected, indicated by a checked checkbox next to its name. Other listed groups include 'ep12,13,14', 'launch-wizard-1', 'launch-wizard-2', 'master', 'sanika9', 'sanikaservergroup', and 'worker'. The interface includes standard browser navigation and search bars at the top.

The screenshot shows the 'Capacity' configuration section of the AWS Elastic Beanstalk console. It includes fields for setting the environment type (Single instance), defining the instance range (Min: 1, Max: 1), specifying fleet composition (On-Demand instance selected), and defining maximum spot price (Default). The status bar at the bottom indicates the date and time as 18-Oct-24 at 10:33 PM.

Capacity Info
Configure the compute capacity of your environment and auto scaling settings to optimize the number of instances used.

Auto scaling group

Environment type
Select a single-instance or load-balanced environment. You can develop and test an application in a single-instance environment to save costs and then upgrade to a load-balanced environment when the application is ready for production. Learn more [?]

Single instance

Instances
Min: 1
Max: 1

Fleet composition
Spot instances are launched at the lowest available price. Learn more [?]
 On-Demand instance
 Spot instance

Maximum spot price
The maximum price per instance-hour, in USD, that you're willing to pay for a Spot Instance. Setting a custom price limits your chances to fulfill your target capacity using Spot instances.
 Default

CloudShell Feedback © 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

The screenshot shows the 'Architecture' configuration section of the AWS Elastic Beanstalk console. It includes options for selecting processor architecture (x86_64 selected, arm64 - new available) and choosing instance types (t3.micro and t3.small selected). It also shows the AMI ID (ami-087bd39f4380fb359) and availability zones (Any selected). The status bar at the bottom indicates the date and time as 18-Oct-24 at 10:34 PM.

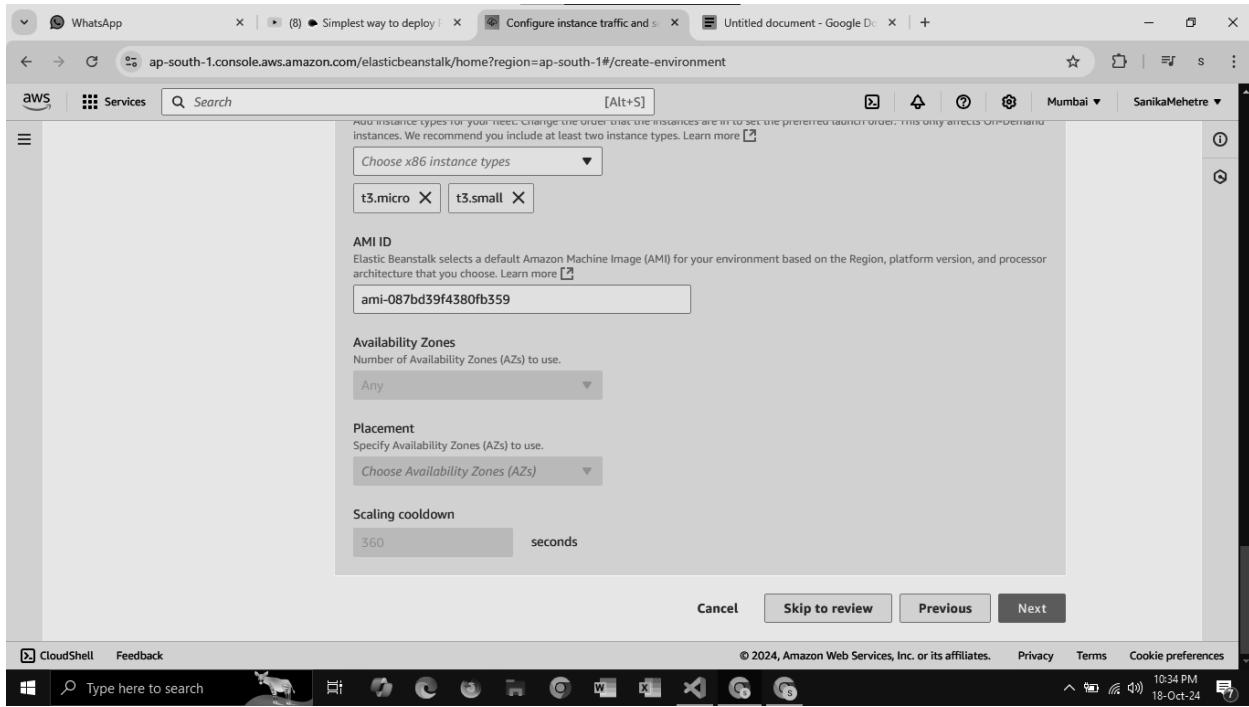
Architecture
The processor architecture determines the instance types that are made available. You can't change this selection after you create the environment. Learn more [?]
 x86_64
 arm64 - new

Instance types
Add instance types for your fleet. Change the order that the instances are in to set the preferred launch order. This only affects On-Demand instances. We recommend you include at least two instance types. Learn more [?]
Choose x86 instance types
 t3.micro t3.small

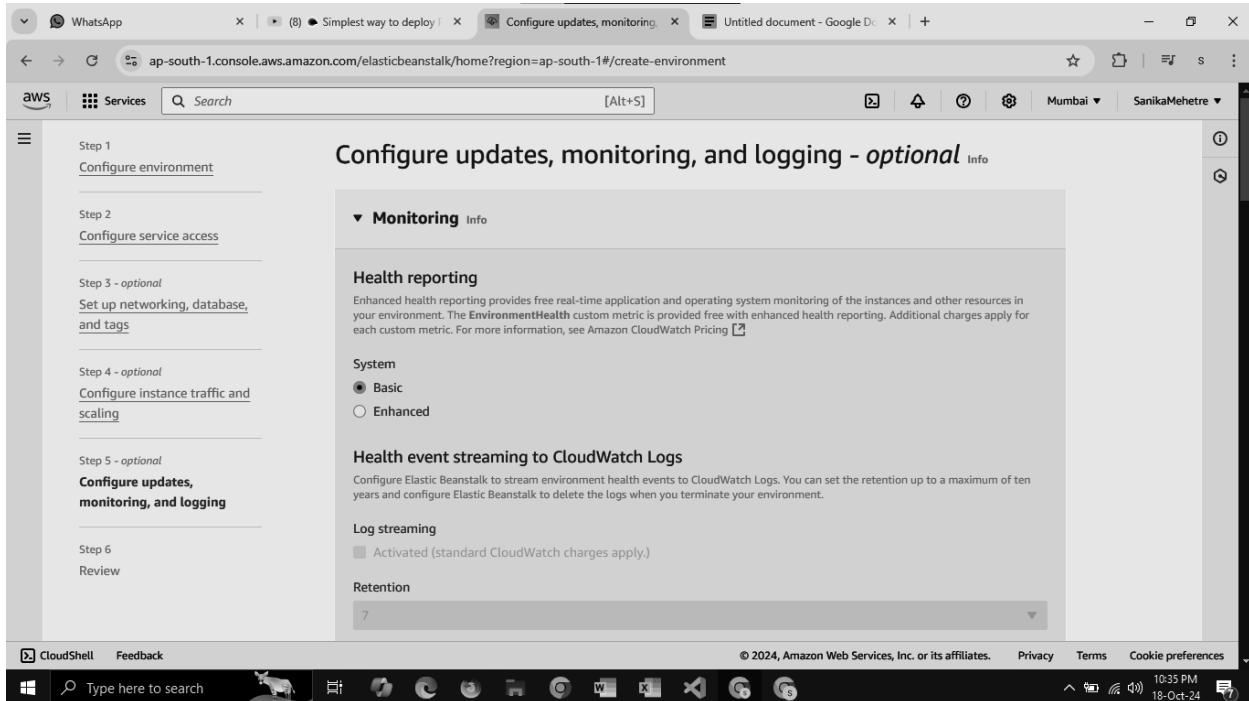
AMI ID
Elastic Beanstalk selects a default Amazon Machine Image (AMI) for your environment based on the Region, platform version, and processor architecture that you choose. Learn more [?]
ami-087bd39f4380fb359

Availability Zones
Number of Availability Zones (AZs) to use.
Any

CloudShell Feedback © 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences



- Click "Next".
- Choose "Basic" Health Reporting.



Deselect the default Activated option.

The screenshot shows the AWS Elastic Beanstalk configuration interface for creating an environment. The 'Managed platform updates' section is expanded, showing the following settings:

- Managed updates**: A checkbox labeled 'Activated' is checked.
- Weekly update window**: Set to 'Monday at 08:06 UTC'.
- Update level**: Set to 'Minor and patch'.
- Instance replacement**: A checkbox labeled 'Activated' is checked.

The 'Email notifications' section is also visible, prompting to enter an email address for notifications.

The screenshot shows the AWS Elastic Beanstalk configuration interface for creating an environment. The 'Rolling updates and deployments' section is expanded, showing the following settings:

- Application deployments**: A note states that Amazon Elastic Beanstalk propagates source code changes and software configuration updates.
- Deployment policy**: Set to 'All at once'.
- Batch size type**: Set to 'Percentage'.
- Deployment batch size**: Set to 100% instances at a time.
- Configuration updates**: A note states that changes to virtual machine settings and VPC configuration trigger rolling updates.
- Rolling update type**: Set to 'Deactivated'.

The interface includes standard browser navigation and search bars at the top and bottom.

The screenshot shows the 'Deployment preferences' section of the AWS Elastic Beanstalk configuration interface. It includes settings for ignoring health checks (set to 'False'), setting a health threshold ('Ok'), and defining a command timeout of 600 seconds. Below this, there's a collapsed section for 'Platform software' and a 'Container options' section which is also collapsed.

Step 2: Environment Configuration

This is an important step

- Add the environment properties here, as they are in the ".env" file of your code.

The screenshot shows the 'Environment properties' section of the AWS Elastic Beanstalk configuration interface. It lists several environment variables with their values:

Name	Value	Action
NODE_ENV	production	Remove
DB_URI	mongodb+srv://komalvsingh1111:Ryfxl	Remove
PORT	8800	Remove
API keys	K0QVcXqzpwcuS7uOstMzUD00Ikij8DAC	Remove

At the bottom, there is a button to 'Add environment property'.

- Now, review all the settings.

The screenshot shows the 'Review' step of the AWS Elastic Beanstalk environment configuration. On the left, a sidebar lists steps: 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), and Step 6 (Review). The main area displays 'Environment information' for a 'Web server environment' named 'sanikawebapp'. It includes fields for Application name ('sanikawebapp'), Environment name ('Sanikawebapp-env'), Application code ('Sample application'), and Platform ('arn:aws:elasticbeanstalk:ap-south-1::platform/Node.js 20 running on 64bit Amazon Linux 2023/6.2.2'). Below this is the 'Step 2: Configure service access' section, which is currently empty. The bottom of the screen shows a Windows taskbar with various pinned icons.

- If everything is correct, click on submit.
- If you wish to make any changes, you can simply edit here itself.

The screenshot shows the 'Configure environment' step of the AWS Elastic Beanstalk setup. The left sidebar shows the same step list as the previous screenshot. The main area contains two sections: 'Lifecycle' and 'Environment properties'. Under 'Lifecycle', settings include Log streaming (Deactivated), Proxy server (nginx), Logs retention (7 days), Rotate logs (Deactivated), Update level (minor), and X-Ray enabled (Deactivated). Under 'Environment properties', a table lists environment variables with their keys and values:

Key	Value
API keys	JWT_SECRET_KEY=zK1fb5UfqXdo8Oq7/2eP4Oph...
DB_URI	mongodb+srv://komalsingh1111:RyfxkX2xmMRy...
NODE_ENV	production
PORT	8800

At the bottom are 'Cancel', 'Previous', and 'Submit' buttons. The bottom of the screen shows a Windows taskbar with pinned icons.

Step 3: Deployment

- Now, the Elastic Beanstalk is launching the environment.

The screenshot shows the AWS Elastic Beanstalk Environment Overview page for the 'Sanikawebapp-env' environment. The main message is 'Elastic Beanstalk is launching your environment. This will take a few minutes.' The left sidebar shows the application and environment details. The right panel has tabs for 'Environment overview' and 'Platform'. Under 'Environment overview', it shows Health (Unknown), Environment ID (e-yrfmqmznm), Domain (sanikawebapp.ap-south-1.elasticbeanstalk.com), and Application name (sanikawebapp). Under 'Platform', it shows Node.js 20 running on 64bit Amazon Linux 2023/6.2.2, Running version -, and Platform state Supported.

- Events will start firing up, during the creation of the environment

The screenshot shows the AWS Elastic Beanstalk Events tab for the 'Sanikawebapp-env' environment. It displays five log entries under the 'Events (5) Info' section. The entries are:

Time	Type	Details
October 18, 2024 22:55:30 (UTC+5:30)	INFO	Waiting for EC2 instances to launch. This may take a few minutes.
October 18, 2024 22:55:14 (UTC+5:30)	INFO	Created EIP: 35.154.165.193
October 18, 2024 22:54:58 (UTC+5:30)	INFO	Created security group named: sg-021507466339b059e
October 18, 2024 22:54:33 (UTC+5:30)	INFO	Using elasticbeanstalk-ap-south-1-009160042094 as Amazon S3 storage bucket for environment data.
October 18, 2024 22:54:32 (UTC+5:30)	INFO	createEnvironment is starting.

- The environment is now successfully launched.

Environment successfully launched.

Sanikawebapp-env Info

Environment overview

Health	Environment ID
Green	e-yrfrmqmznm
Domain	Application name
sanikawebapp.ap-south-1.elasticbeanstalk.com	sanikawebapp

Platform

Platform
Node.js 20 running on 64bit Amazon Linux 2023/6.2.2
Running version
-
Platform state
Supported

Events | Health | Logs | Monitoring | Alarms | Managed updates | Tags

- We can now see our web application in “Applications” and our environment beside it.

Environment successfully launched.

Applications (1) Info

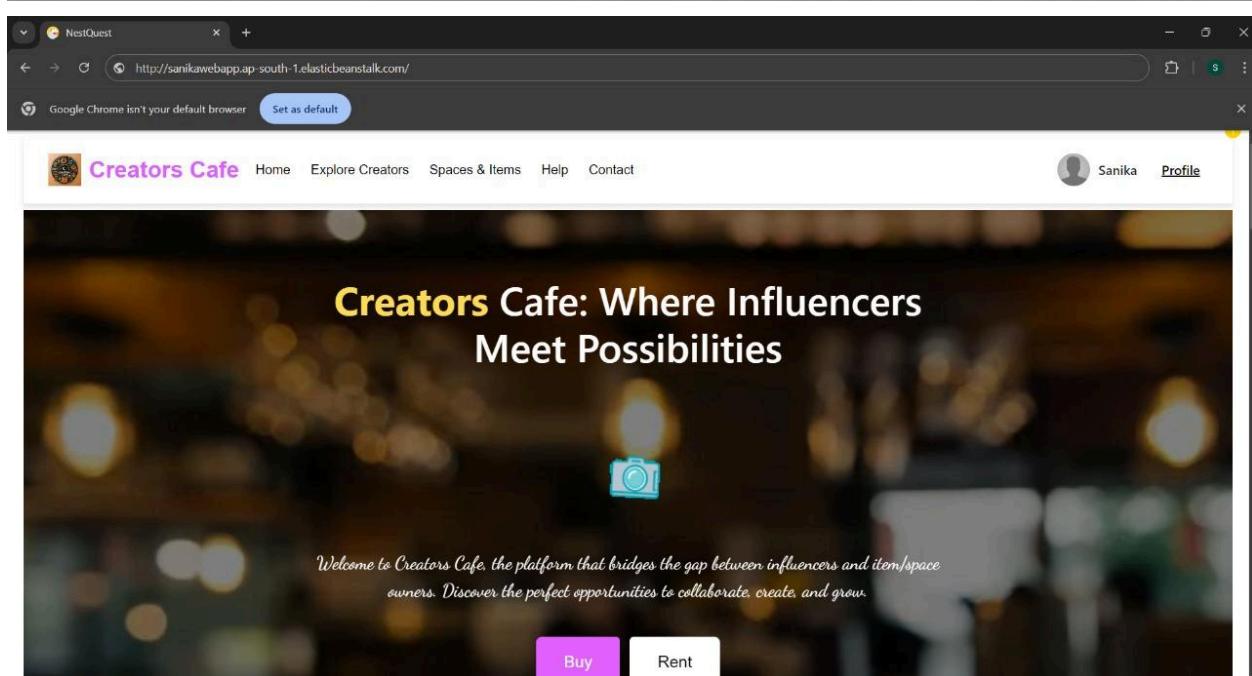
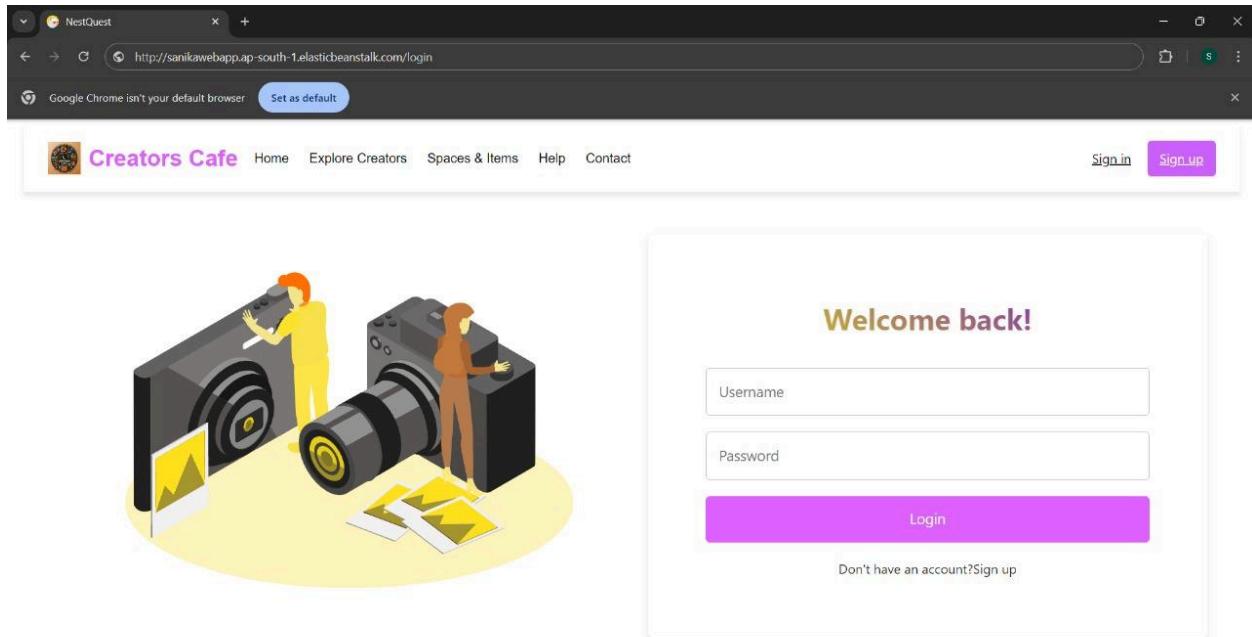
Application name	Environments	Date created	Last modified
sanikawebapp	Sanikawebapp-env	October 18, 2024 22:15:20...	October 18, 2024 22:15:20...

- Click on the link under “Domain”.

The screenshot shows the AWS Elastic Beanstalk console. On the left, a sidebar for the application 'sanikawebapp' and environment 'Sanikawebapp-env' is visible. The main area displays the 'Environment successfully launched.' message and the 'Sanikawebapp-env' info card. The card includes sections for 'Environment overview' (Health: Green, Domain: sanikawebapp.ap-south-1.elasticbeanstalk.com) and 'Platform' (Node.js 20 running on 64bit Amazon Linux 2023/6.2.2). Other tabs like Events, Health, Logs, and Monitoring are also present.

- After much error solving, and deploying the application multiple times it was successful.
- You can see your web application, if the deployment process was successful.

The screenshot shows a web browser displaying the 'Creators Cafe' website. The URL in the address bar is <http://sanikawebapp.ap-south-1.elasticbeanstalk.com/>. The page has a dark background with blurred lights. The main heading is 'Creators Cafe: Where Influencers Meet Possibilities'. Below the heading, there is a welcome message: 'Welcome to Creators Cafe, the platform that bridges the gap between influencers and item/space owners. Discover the perfect opportunities to collaborate, create, and grow.' At the bottom, there are two prominent buttons: 'Buy' and 'Rent'.



The screenshot shows the homepage of the 'Creators Cafe' website. At the top, there's a navigation bar with links for Home, Explore Creators, Spaces & Items, Help, and Contact. A user profile for 'Sanika' is visible on the right. Below the header, a section titled 'Explore Our Studios And Items' displays five studio options with small images, names, reviews, and prices:

- Modern Photography Studio**: 45 reviews, ₹5,500/hr
- Cozy Recording Studio**: 25 reviews, ₹3,700/hr
- Film Production Studio**: 38 reviews, ₹7,500/hr
- Outdoor Garden Studio**: 30 reviews, ₹6,500/hr
- Indoor Podcast Studio**: 40 reviews, ₹4,500/hr

Below the studio cards, there are icons representing various items like cameras, a car, and a sailboat.

- In case of errors, navigate to “Logs” and click on “Request Logs”.
- Here you easily check the errors and solve them.
- After solving the errors, click on “Upload and Deploy”.
- You can now upload the new zip file of your code and update the version.

The screenshot shows the AWS Elastic Beanstalk console. On the left, a sidebar lists 'Applications', 'Environments', and 'Change history'. Under 'Environment: Sanikawebapp-env', it shows 'Go to environment' and other tabs like Configuration, Events, Health, Logs, Monitoring, Alarms, Managed updates, CloudShell, and Feedback. The main area shows the 'Upload and deploy' dialog for the 'Sanikawebapp' environment. The dialog has fields for 'Choose file' (set to 'sanika6api.zip'), 'Version label' (set to 'v9'), and 'Deploy' button. The background shows the environment overview with details like 'Running on 64bit Amazon Linux'.

Conclusion:

After trying different methods for deploying the application,such as AWS CLI, etc. the AWS Elastic Beanstalk service completed the task successfully. I faced numerous errors while doing this case study, such as errors regarding the zip file structure(which needed the entire project structure to be changed and an additional Procfile), errors regarding permissions(for which i created a new role and attached the required policies to that role), errors due to nginx which acts as reverse proxy(502 Bad Gateway error), errors due to the size of type_hash_bucket(for which i added a file in the project directory specifying the size of the hash_bucket and max_bucket), the application also crashed several times due to unavailability of the specific port which was caused by firewall/ antivirus software, i even had to make changes in the node modules which required excessive scanning through code. After making all these changes, the application was successfully deployed on AWS.