AWS Elastic Beanstalk for Application Deployment

A Course Project Report Submitted in partial fulfillment of the course requirements for the award of grades in the subject of

CLOUD BASED AIML SPECIALITY
(22SDCS07A)

by

P. Trisha

(2210030338)

Under the esteemed guidance of

Ms. P. Sree Lakshmi
Assistant Professor,
Department of Computer Science and Engineering

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

K L Deemed to be UNIVERSITY

Aziznagar, Moinabad, Hyderabad,
Telangana, Pincode: 500075

April 2025

K L Deemed to be UNIVERSITY

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

Certificate

This is Certified that the project entitled **"AWS Elastic Beanstalk for Application Deployment"** which is a experimental &/ Simulation work carried out by P.Trisha (2210030338), in partial fulfillment of the course requirements for the award of grades in the subject of **CLOUD BASED AIML SPECIALITY**, during the year **2024-2025**. The project has been approved as it satisfies the academic requirements.

Ms.P.Sree Lakshmi

Course Coordinator

Dr. Arpita Gupta

Head of the Department

Ms. P. Sree Lakshmi

Course Instructor

CONTENTS

1. INTRODUCTION

AWS Elastic Beanstalk is a Platform-as-a-Service (PaaS) offering from Amazon Web Services (AWS) that simplifies application deployment and management. It allows developers to deploy and scale web applications and services without worrying about the underlying infrastructure [1]. Elastic Beanstalk supports multiple programming languages, including Java, Python, Node.js, PHP, Ruby, .NET, and Go, making it a versatile choice for different development needs [4].

With Elastic Beanstalk, developers can upload their code, and AWS automatically handles deployment, capacity provisioning, load balancing, and application health monitoring [2]. This automation reduces operational overhead and allows developers to focus on coding rather than infrastructure management [3].

Additionally, Elastic Beanstalk integrates seamlessly with AWS services like RDS (Relational Database Service), S3 (Simple Storage Service), and CloudWatch for enhanced functionality [7]. One of the key advantages of Elastic Beanstalk is its ability to scale applications automatically based on demand. It provides built-in monitoring and logging features, ensuring high availability and performance optimization [5]. Users have the flexibility to choose between managed and customized environments, enabling greater control over the application stack [6].

2. AWS SERVICES USED AS PART OF THE PROJECT

To implement efficient and scalable application deployment, AWS Elastic Beanstalk has been utilized. It automates infrastructure management, ensuring seamless deployment and scalability. Below are the key AWS services used in the project:

Amazon EC2 (Elastic Compute Cloud)

Amazon EC2 provides the underlying virtual machines (instances) where the application is hosted [2].

- Hosting the application environment managed by Elastic Beanstalk.
- Scaling dynamically based on traffic demands.
- Running application processes efficiently with optimized configurations.

Amazon S3 (Simple Storage Service)

Amazon S3 is used to store application versions, static assets, and logs [1].

- Storing application artifacts before deployment.
- Managing logs generated by Elastic Beanstalk environments.
- Storing static content such as images and configuration files.
-

Amazon RDS (Relational Database Service)

Amazon RDS provides a managed relational database solution [7].

- Storing structured application data.
- Handling database scaling and backups automatically.
- Ensuring high availability and security for database operations.

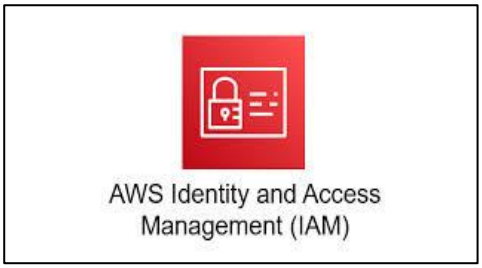
Amazon CloudWatch

Amazon CloudWatch monitors application performance and infrastructure health [4].

- Tracking key application and system metrics.
- Generating alerts based on predefined thresholds.
- Collecting and analysing logs for troubleshooting issues.

AWS IAM (Identity and Access Management)

AWS IAM provides secure access management to AWS resources [5].



- Defining role-based access control for Elastic Beanstalk and associated services.
- Managing permissions for different application components.
- Enforcing security best practices through policy management.

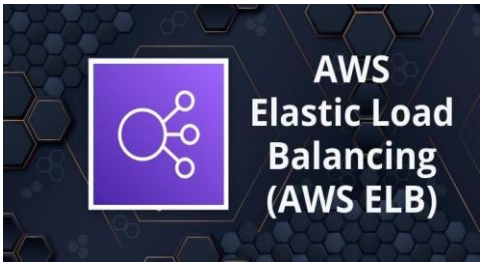
AWS Auto Scaling

AWS Auto Scaling ensures that the application can handle traffic fluctuations [6].

- Automatically adjusting the number of EC2 instances.
- Reducing infrastructure costs during low-traffic periods.
- Maintaining application performance and availability.

AWS Elastic Load Balancing (ELB)

Elastic Load Balancing distributes incoming traffic across multiple EC2 instances [9].



- Ensuring high availability by balancing user requests.
- Preventing individual instances from becoming overloaded.
- Improving fault tolerance and application responsiveness.

AWS Elastic Beanstalk

AWS Elastic Beanstalk is a fully managed service for deploying and scaling web applications [1]. In this project, Elastic Beanstalk is used for:

- Automatically provisioning the required infrastructure (EC2, RDS, Load Balancer, etc.).

- Managing application deployment and environment configurations.
- Monitoring application health and handling auto-scaling.
- Supports multiple languages (Java, .NET, Python, Node.js, etc.).
- Simplifies deployment—just upload code and it handles the rest.
- Allows customization of underlying AWS resources.
- Supports versioning and easy rollback of deployments.
- Includes monitoring and logging via CloudWatch.

3. STEPS INVOLVED IN SOLVING PROJECT PROBLEM STATEMENT

1. **Understanding the Requirements:**
   - Analyze the problem statement and define project objectives.
   - Identify key AWS services needed for deployment [1].
2. **Designing the Architecture:**
   - Plan the infrastructure using AWS services like Elastic Beanstalk, EC2, RDS, S3, and CloudWatch [3].
   - Define scalability, security, and high availability requirements [4].
3. **Setting Up AWS Environment:**
   - Configure IAM roles and policies for secure access [5].
   - Provision resources like EC2 instances, RDS databases, and S3 storage [6].
4. **Developing & Configuring the Application:**
   - Develop or modify the application to ensure compatibility with AWS services [7].
   - Implement CI/CD pipelines using AWS Code Pipeline and Code Deploy [2].
5. **Deploying & Monitoring:**
   - Deploy the application using AWS Elastic Beanstalk or manual deployment on EC2 [1].
   - Monitor performance and security using CloudWatch, Auto Scaling, and ELB [8].
6. **Testing & Optimization:**
   - Conduct load testing and security checks [9].
   - Optimize performance, costs, and scaling strategies [3].
7. **Maintenance & Scaling:**
   - Implement automated scaling and backup strategies [6].
   - Continuously monitor and update the application for improvements [7].

4. STEPWISE SCREENSHOTS WITH BRIEF DESCRIPTION

Step 1: Navigate to IAM

- Go to AWS Console Home.
- Then, go to IAM (Identity and Access Management).

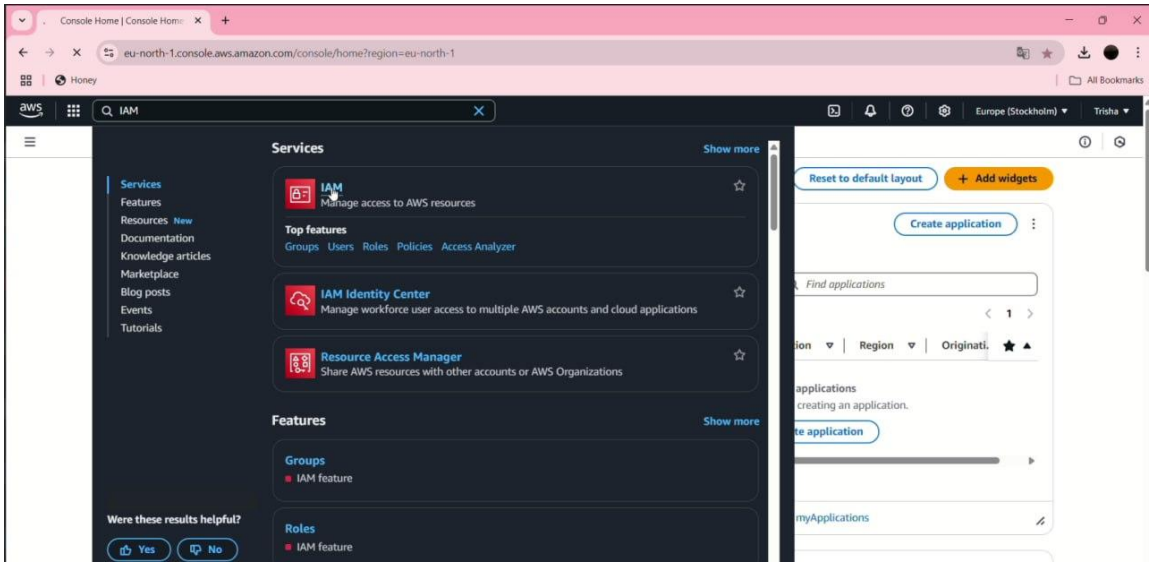


Fig.4.1 Navigate to IAM

Step 2: Create New User

- Enter User Name (e.g., Beanstalk-user).
- Enable Console Access (optional for AWS login).
- Choose IAM User (not Identity Center).

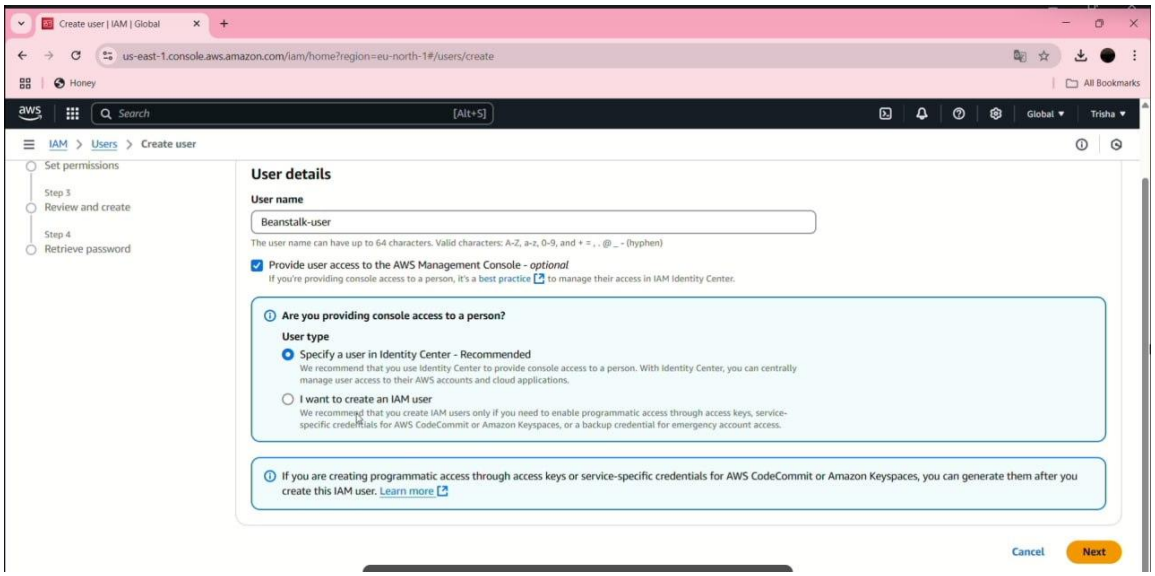


Fig.4.2 Creating a new user

Step 3: Setting Permissions by Attaching Policies Directly to the User

Select Attach Policies Directly (or add to a group).

- Attach policies:
    - AdministratorAccess-AWSElasticBeanstalk
    - AmazonEC2FullAccess
    - AmazonS3FullAccess

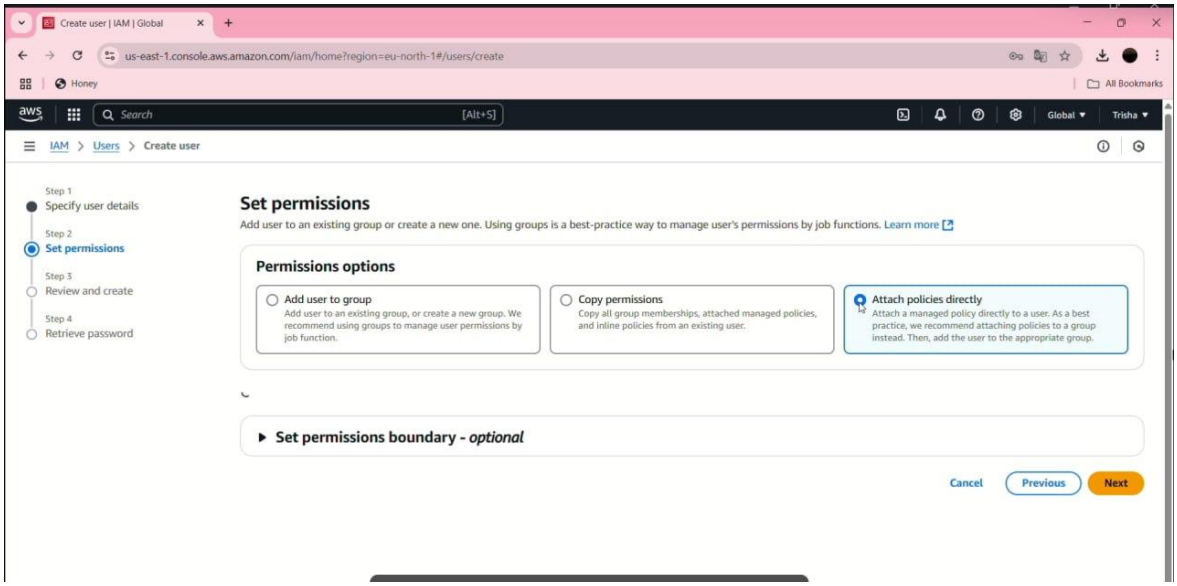


Fig.4.3.1 Setting Permissions by Attaching Policies Directly to the User

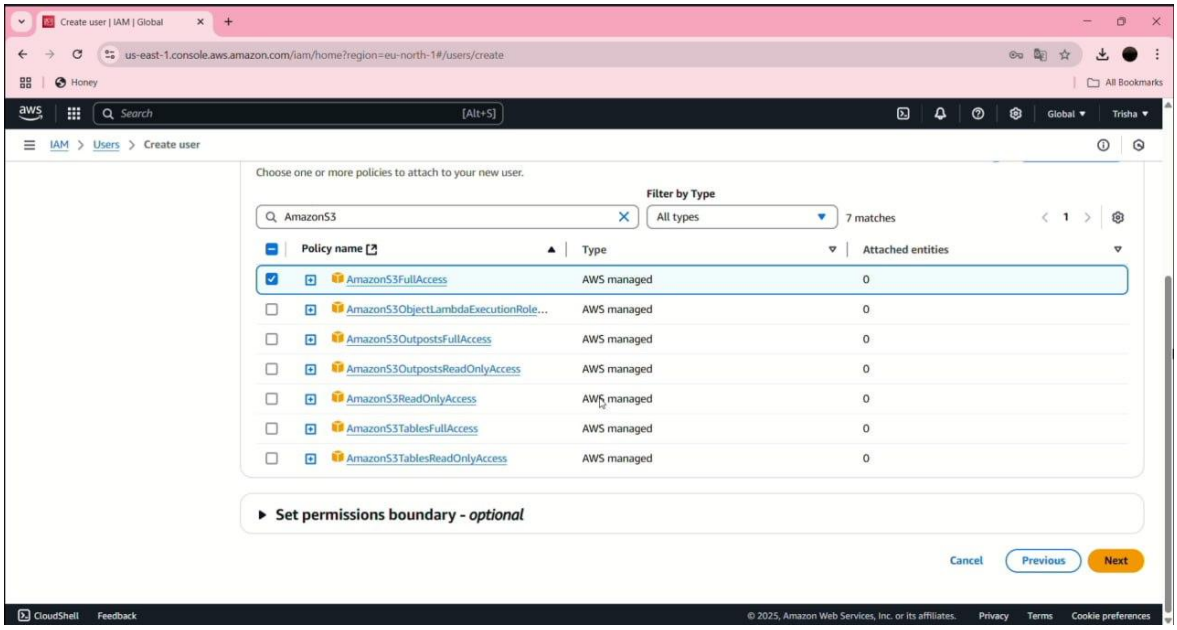


Fig.4.3.2 Attaching the 'AmazonS3FullAccess' Policy to the IAM User in the Set Permissions Step

Step 4: Review & Create

- Verify attached policies.
- User Created Successfully

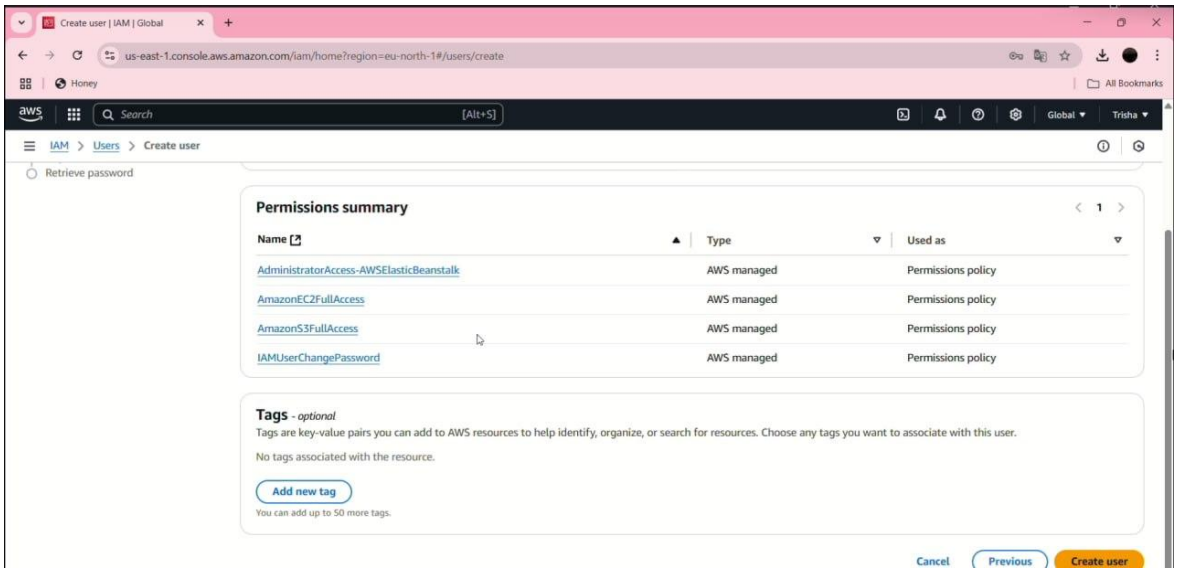


Fig.4.4.1 verifying the policies

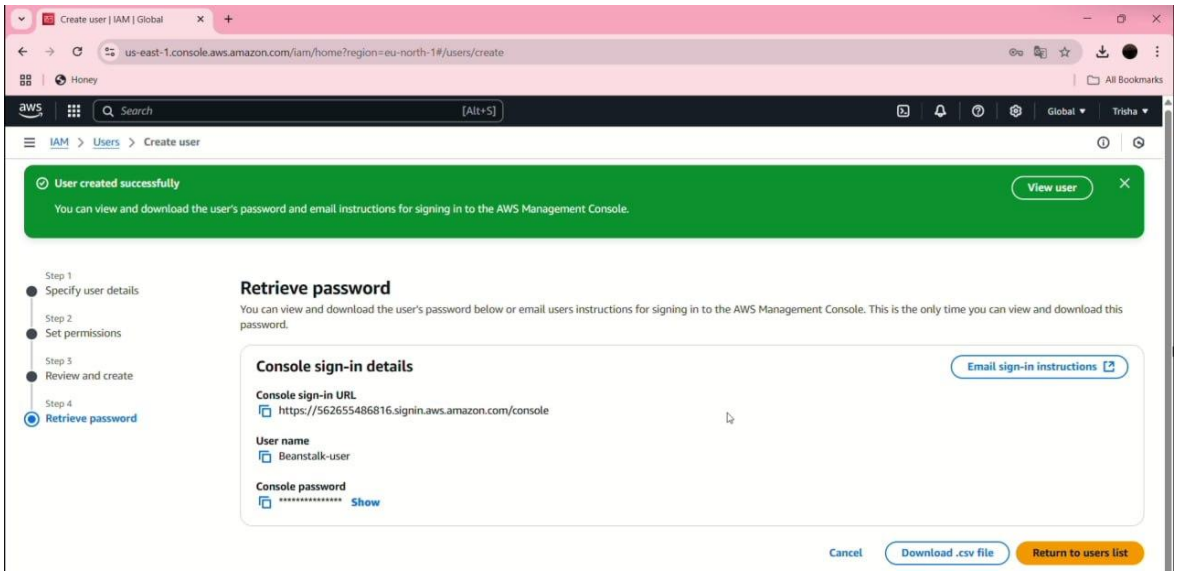


Fig.4.4.2 user created successfully

Step5: Navigate to IAM Roles

- Go to IAM > Roles in the AWS Console.
- Create New Role
- Click Create Role.
- Select AWS Service as the trusted entity.

- Choose Elastic Beanstalk as the use case.

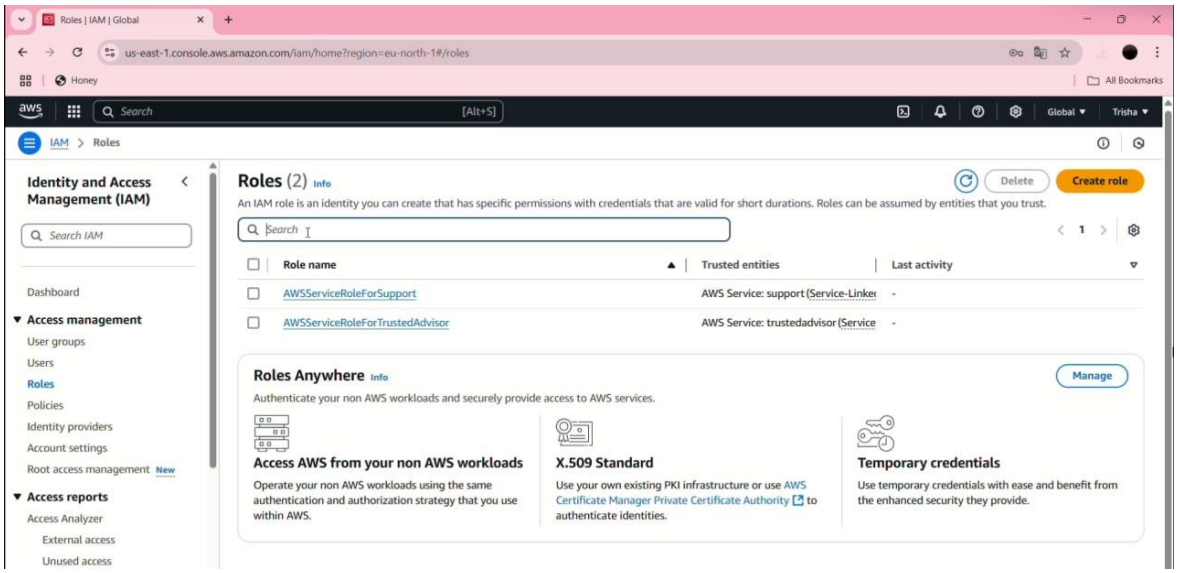


Fig.4.5 Navigate to IAM Roles

Step 6: Set Permissions

Attach AWS managed policies:

- AWSElasticBeanstalkMulticontainerDocker
- AWSElasticBeanstalkWebTier
- AWSElasticBeanstalkWorkerTier

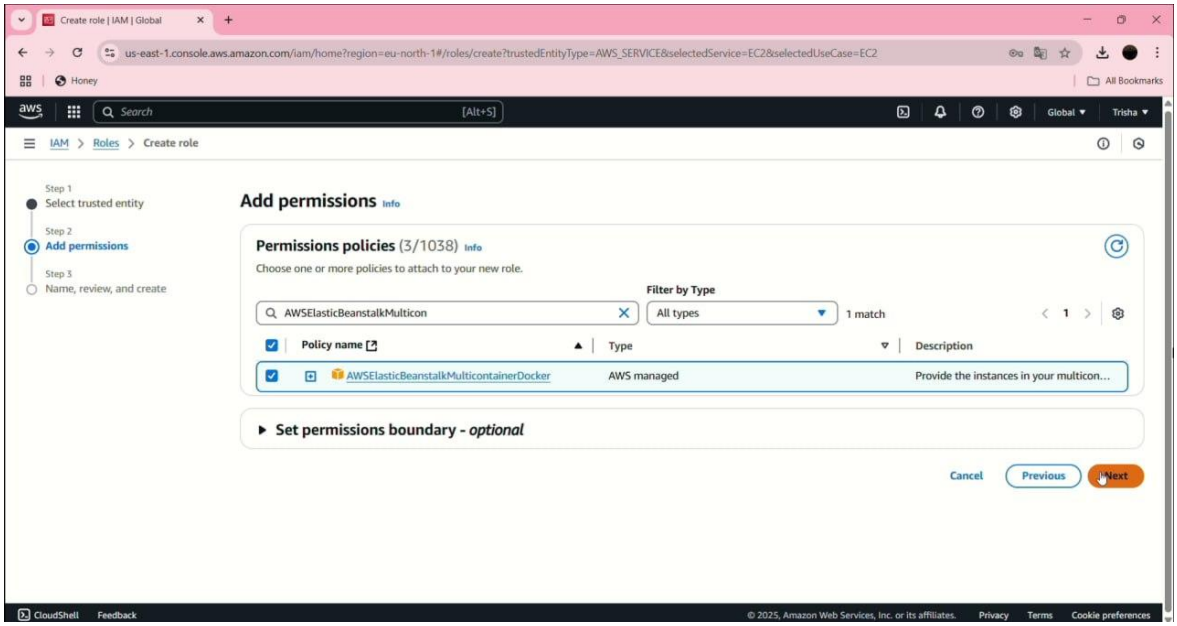


Fig.4.6 Set Permisssions

Step 7: Name & Review

- Enter a Role Name (e.g., aws-elasticbeanstalk-service-role).

- Add a Description (e.g., Allows EC2 instances to call AWS services).
- Create Role
- Confirm settings and click Create Role.

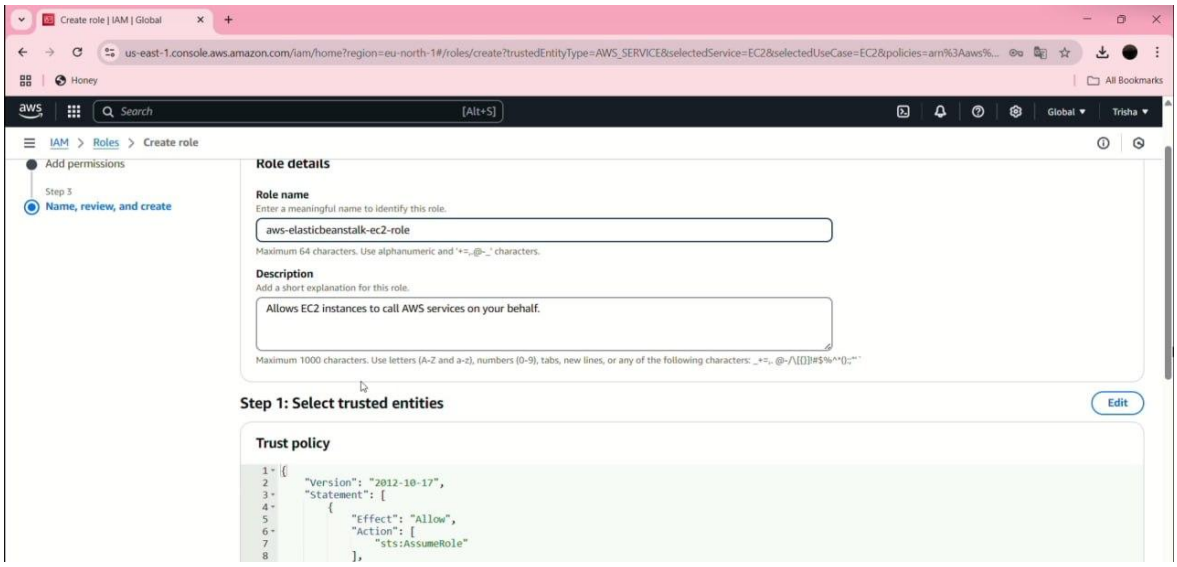


Fig.4.7.1 Name & Review

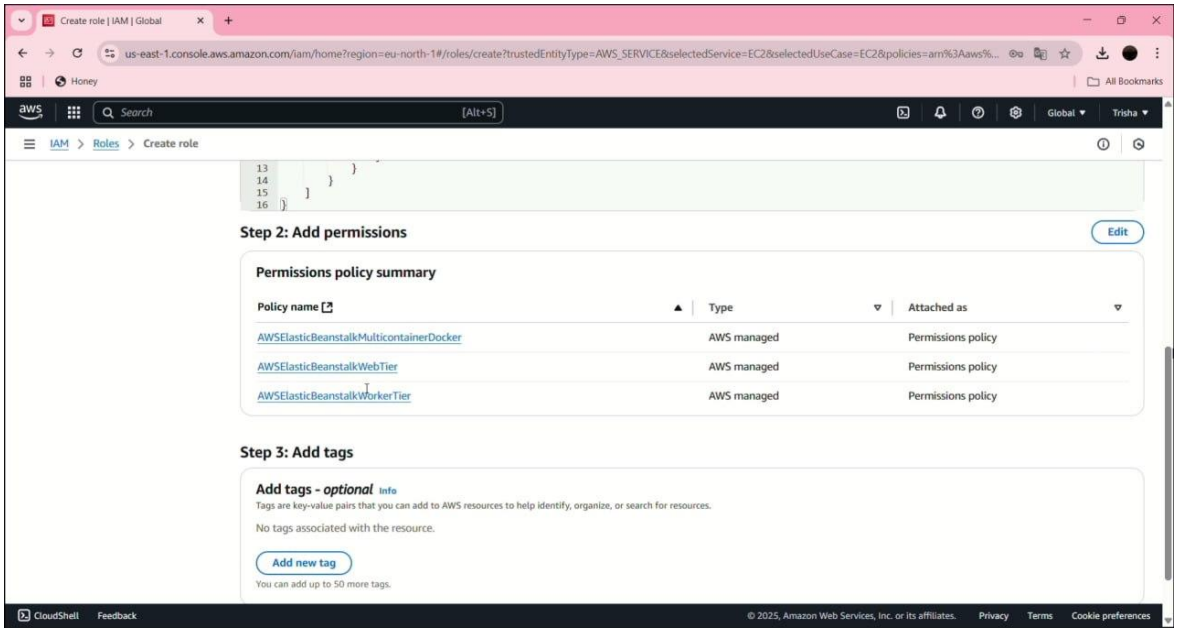


Fig.4.7.2 Verifying the policies

Step 8: Navigate to Elastic Beanstalk

- Go to Services > Elastic Beanstalk.

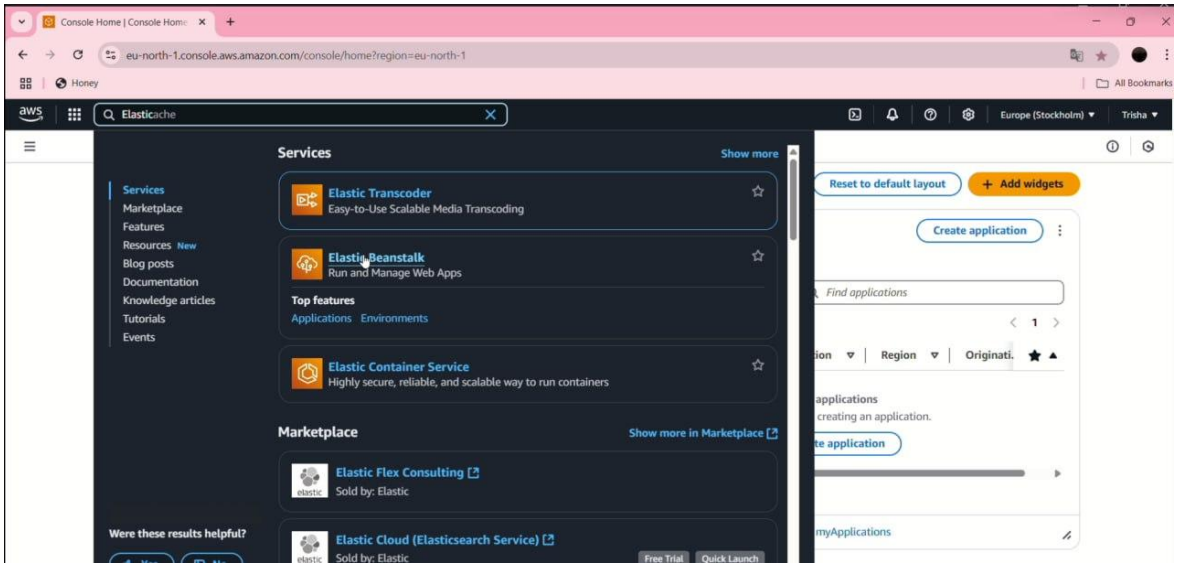


Fig.4.8 Navigate to Elastic Beanstalk

Step 9: Create Application

- Enter an Application Name (e.g., MyFirstApp).

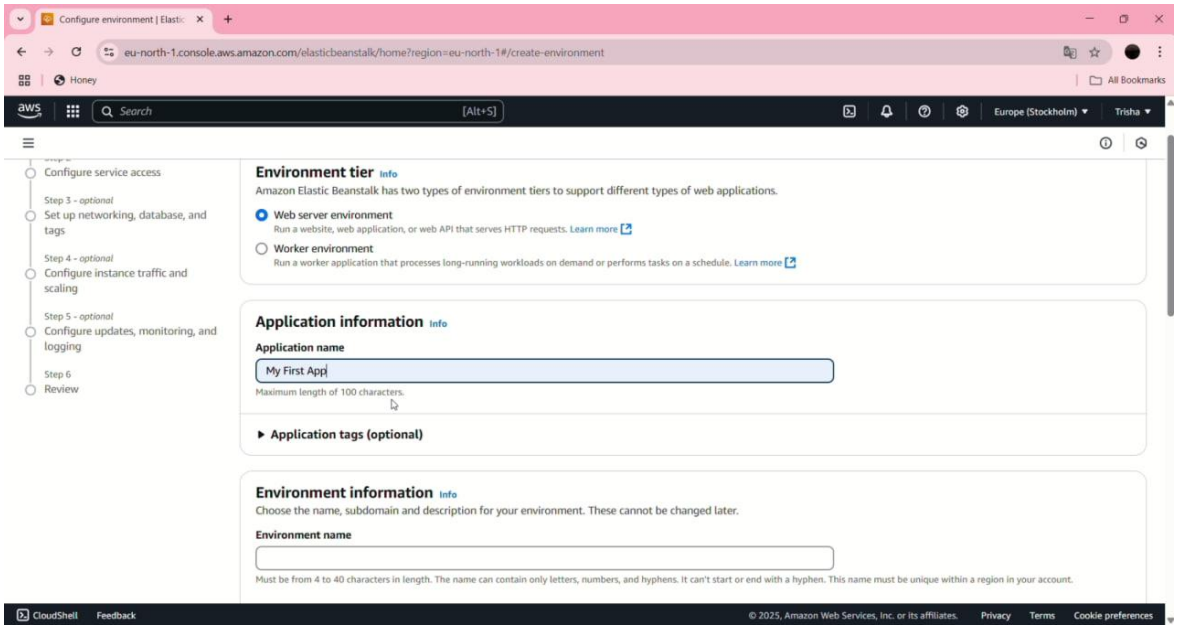


Fig.4.9 Create Application

Step-10: Configure Environment

- Environment Tier: Choose Web Server (HTTP) or Worker (background tasks).
- Platform: Select (e.g., Node.js, Python, Docker).

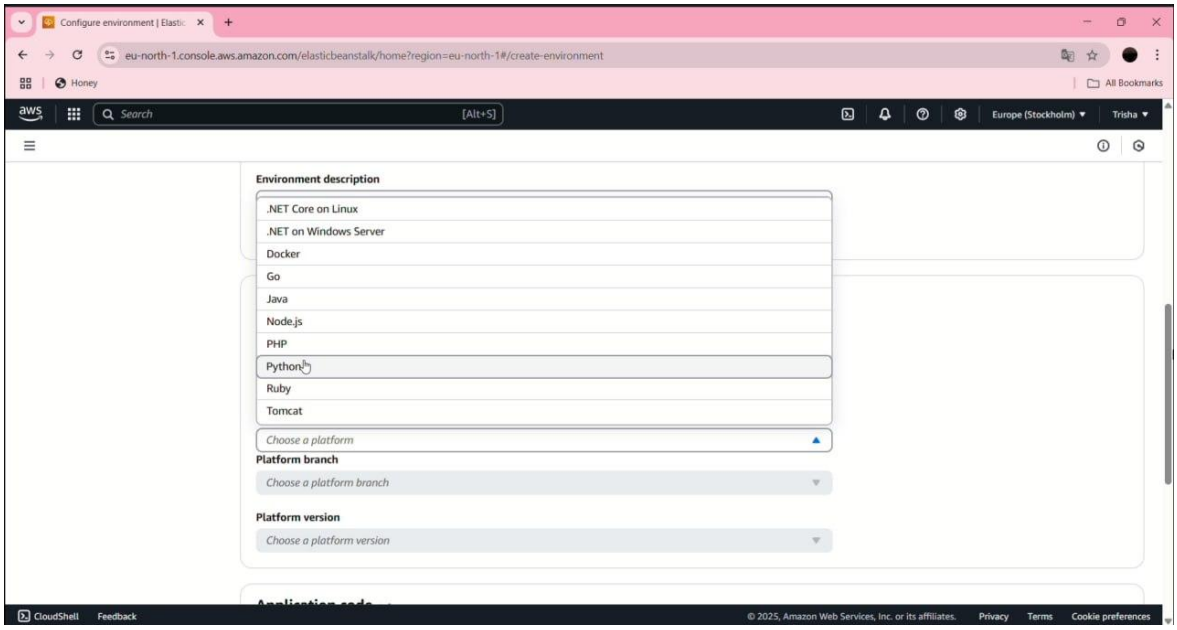


Fig.4.10 Configure Environment

Step-11: Application Code

- Upload a ZIP file or link to S3.

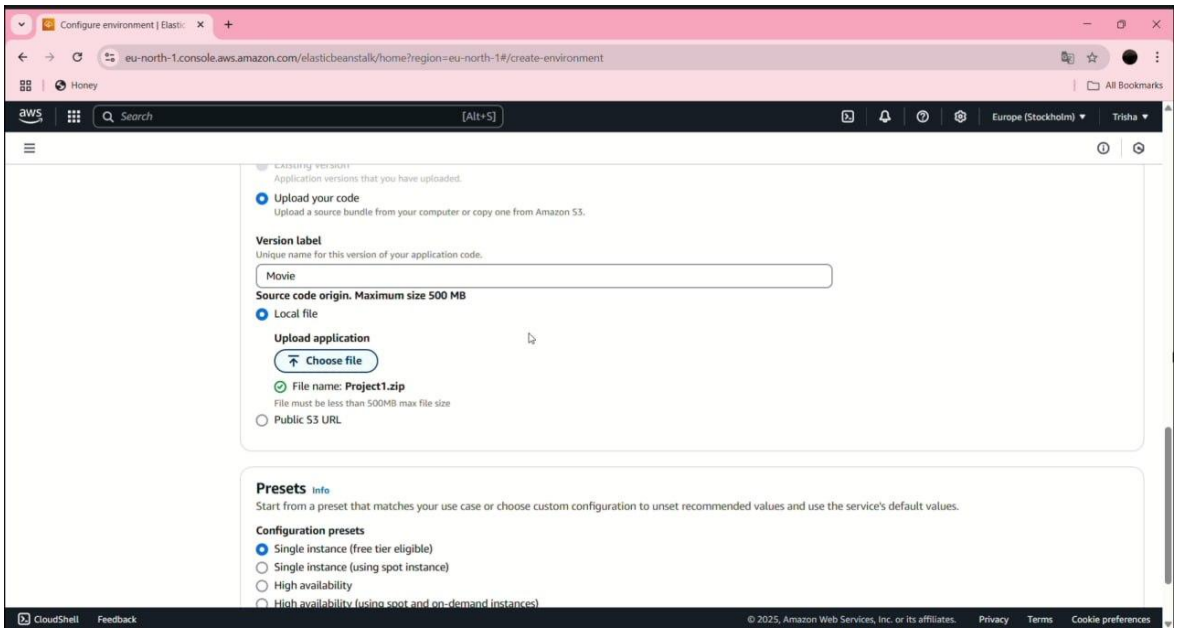


Fig.4.11 Application Code

Step-12: Set IAM Roles

- Service Role: Use existing (aws-elasticbeanstalk-service-role) or create new.
- EC2 Instance Profile: Attach aws-elasticbeanstalk-ec2-role.

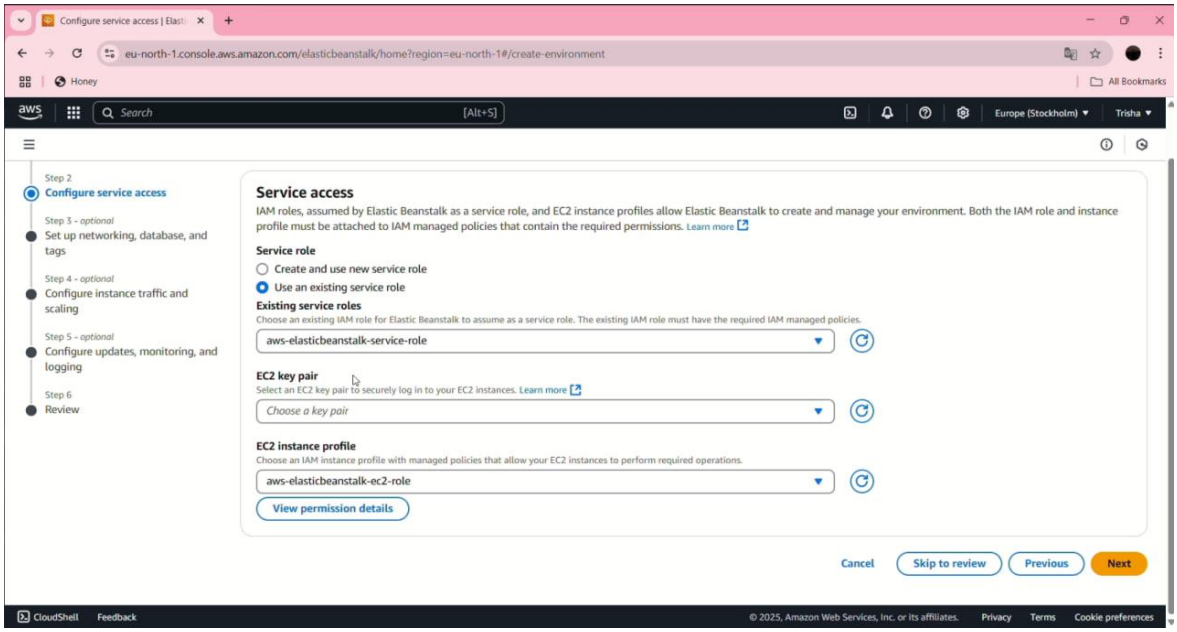


Fig.4.12 set IAM service roles

Step-13: Launch Environment

- Choose a Preset (e.g., Single Instance for testing).
- Click Create Environment to deploy.

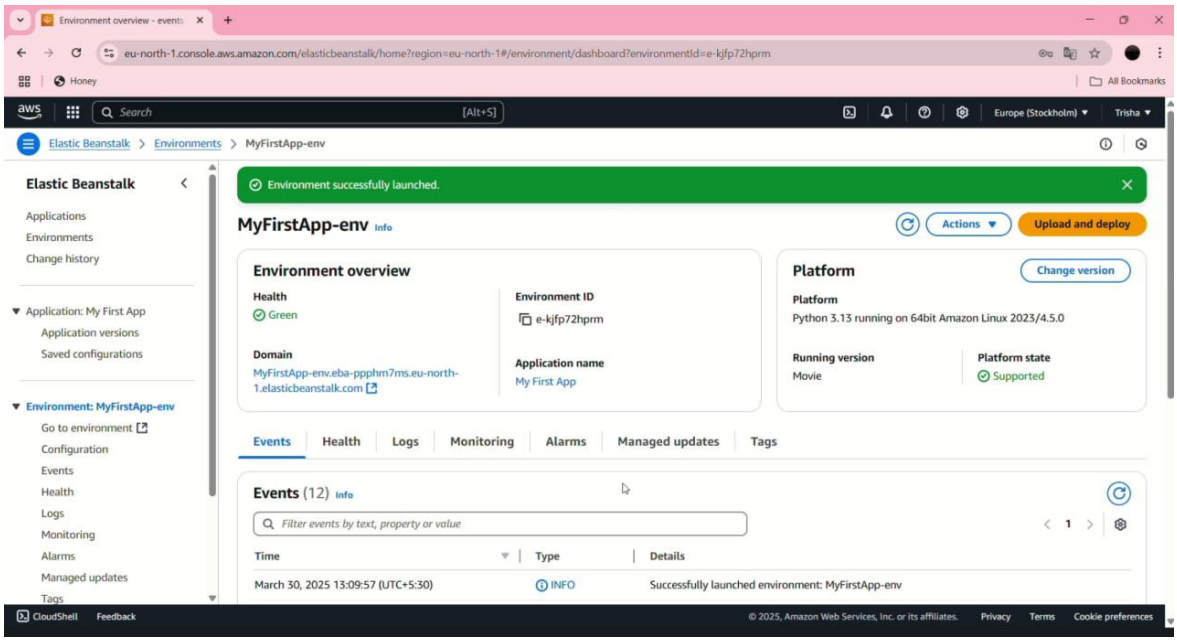


Fig.4.13 Launch Environment

Step-14: Click on link

- If we click on that link the website in the new tab.

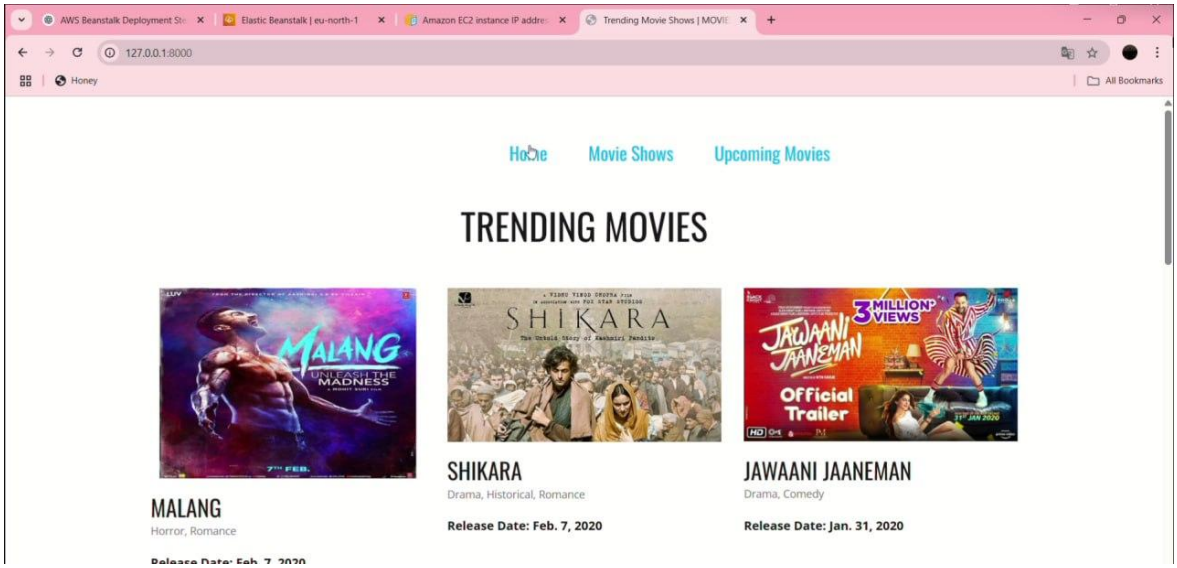


Fig.4.14.1 Movie Booking website output

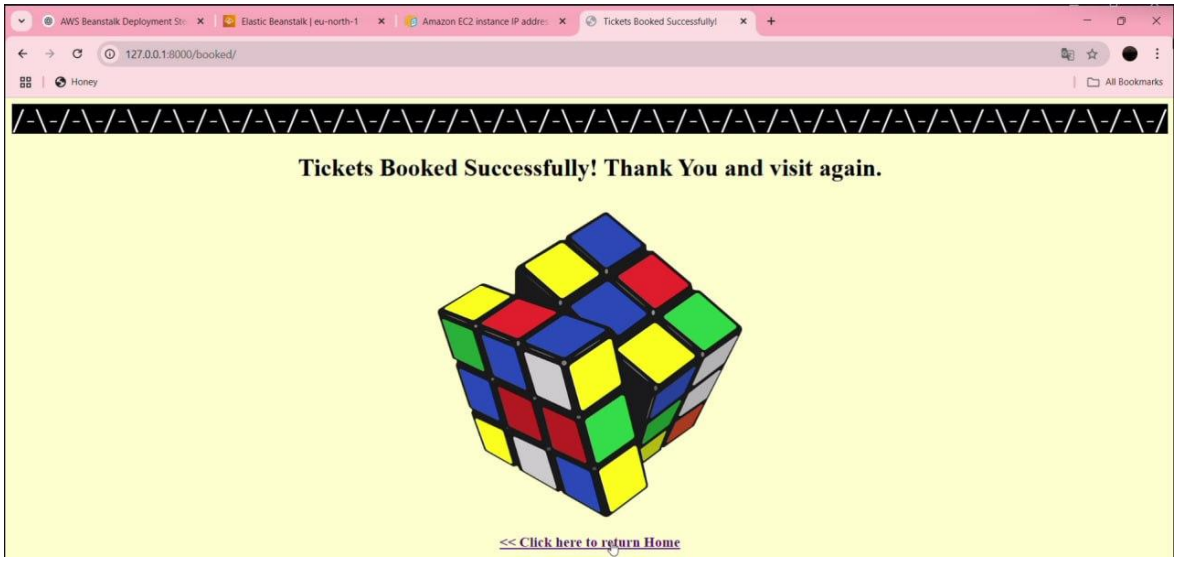


Fig.4.14.2 Ticket is booked successfully

5. LEARNING OUTCOME

Understanding AWS Cloud Services

Gained hands-on experience with AWS Elastic Beanstalk, EC2, RDS, S3, CloudWatch, and other key services [1].

Application Deployment & Management

Learned how to deploy, manage, and scale applications using AWS Elastic Beanstalk [5].

Infrastructure as Code (IaC) & Automation

Understood how to automate resource provisioning using AWS CloudFormation and CI/CD pipelines with Code Pipeline and Code Deploy.

Scalability & High Availability

Learned how to configure Auto Scaling and Elastic Load Balancing (ELB) to handle varying traffic loads [8].

Security & Access Control

Gained knowledge of AWS IAM policies, role-based access control, and security best practices [7].

Monitoring & Performance Optimization

Learned how to track application performance and resource utilization using Amazon CloudWatch.

Cost Optimization & Best Practices

Understood AWS pricing models and optimized resource usage to minimize costs [9].

Problem-Solving & Troubleshooting

Developed skills in identifying and resolving issues related to deployment, scaling, and performance [6].

End-to-End Cloud Project Execution

Gained practical experience in planning, deploying, and maintaining a cloud-based application from start to finish [2].

6. CONCLUSION

The AWS Elastic Beanstalk for Application Deployment project demonstrated how to efficiently deploy, manage, and scale applications in the cloud with minimal manual effort. As a fully managed Platform-as-a-Service (PaaS), AWS Elastic Beanstalk automates infrastructure provisioning, load balancing, and auto-scaling, allowing developers to focus on application logic rather than cloud resource management. By integrating EC2 for computing power, RDS for database management, S3 for storage, CloudWatch for monitoring, IAM for security, and Code Pipeline for CI/CD automation, the deployment process was secure, scalable, and efficient. Elastic Load Balancing (ELB) distributed traffic across multiple instances to ensure high availability, while Auto Scaling dynamically adjusted resources based on demand, optimizing performance and cost. The project also provided hands-on experience in cloud security, performance monitoring, cost optimization, and automation. AWS CloudWatch enabled real-time tracking of application health, IAM enforced role-based access control for enhanced security, and AWS Code Pipeline with Code Deploy streamlined CI/CD workflows to minimize downtime and improve development efficiency. Overall, AWS Elastic Beanstalk proved to be a powerful, scalable, and cost-effective cloud deployment solution, simplifying the process while maintaining flexibility and control for developers and organizations.

7. FUTURE ENHANCEMENTS

Future enhancements for AWS Elastic Beanstalk in application deployment could focus on deeper integration with container orchestration tools like Amazon ECS and EKS, enabling hybrid container and code deployments. Improved support for CI/CD pipelines with native integration to AWS CodePipeline and third-party tools could streamline DevOps workflows. Enhanced observability with AI-driven performance insights and anomaly detection through CloudWatch could further optimize application health monitoring. Additionally, expanding platform support for emerging languages and frameworks, along with greater customization options for environment configuration, would make Elastic Beanstalk even more versatile and developer-friendly for modern application deployment needs.

8. REFERENCES

[1] AWS Elastic Beanstalk Overview: https://aws.amazon.com/elasticbeanstalk/

[2] Overview of Deployment Options on AWS: https://docs.aws.amazon.com/whitepapers/latest/overview-deployment-options/aws-elastic-beanstalk.html

[3] Introduction to DevOps on AWS: AWS Elastic Beanstalk. https://docs.aws.amazon.com/whitepapers/latest/introduction-devops-aws/aws-elasticbeanstalk.html

[4] AWS Elastic Beanstalk Supported Platforms. https://docs.aws.amazon.com/elasticbeanstalk/latest/dg/concepts.platforms.html

[5] AWS Elastic Beanstalk Features. https://docs.aws.amazon.com/elasticbeanstalk/latest/dg/Welcome.html

[6] Managing AWS Resources in Elastic Beanstalk. https://docs.aws.amazon.com/elasticbeanstalk/latest/dg/AWSHowTo.html

[7] AWS Elastic Beanstalk and AWS Service Integration. https://aws.amazon.com/elasticbeanstalk/faqs/

[8] Managing Elastic Beanstalk Environments. https://docs.aws.amazon.com/elasticbeanstalk/latest/dg/environments.html

[9] AWS Elastic Beanstalk FAQs: https://aws.amazon.com/elasticbeanstalk/faqs/