Project Report

Title: Cloud & DevOps Engineering Projects Report
Name: M Anusuya
Course: Cloud Computing & DevOps
Institute: Besant Technologies, Bengaluru
Duration: Jun 2025 – Present
Role: Cloud & DevOps Trainee

Declaration

I hereby declare that this project report titled **"Cloud & DevOps Engineering Projects"** is my original work carried out during my training period at Besant Technologies. This work has not been submitted previously for any academic qualification. All references and resources used are duly acknowledged.

Acknowledgement

I would like to express my sincere gratitude to Besant Technologies and my trainers for their continuous guidance and support throughout my Cloud & DevOps training. I also thank my peers for their collaboration and encouragement, which helped me successfully complete these projects.

Abstract

This report presents a comprehensive overview of multiple Cloud and DevOps projects implemented using AWS, Azure, Linux, networking concepts, and modern DevOps tools. The projects focus on automating infrastructure provisioning, implementing CI/CD pipelines, deploying scalable cloud-native applications, and improving system reliability and availability. These implementations demonstrate realworld DevOps practices such as Infrastructure as Code (IaC), containerization, orchestration and automation.

Table of Contents

Introduction

Cloud computing and DevOps have become essential for delivering scalable, reliable, and fast software solutions. This project report showcases hands-on implementations that combine cloud infrastructure, automation, and CI/CD practices. The objective is to reduce manual effort, improve deployment speed, and ensure high availability and security of applications.

Tools & Technologies Used

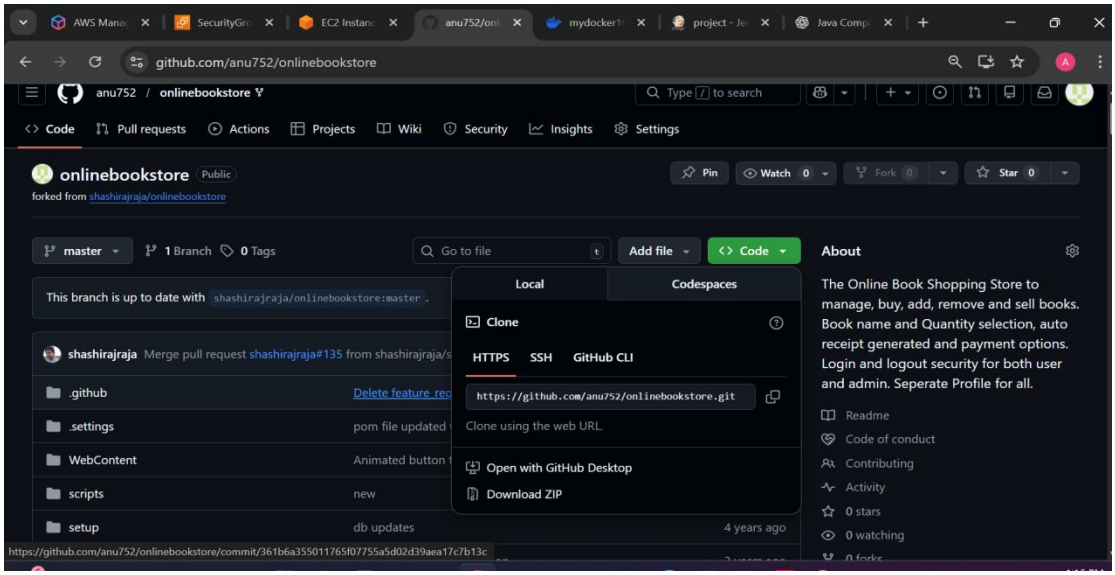
• **Cloud Platforms:** AWS
• **Operating Systems:** Linux (RedHat, Ubuntu)
• **DevOps Tools:** Git, GitHub, Jenkins, Docker, Kubernetes
• **Programming & Scripting:** Python, Shell Scripting
• **Networking:** VPC, Subnets, Routing, Load Balancers, Security Groups

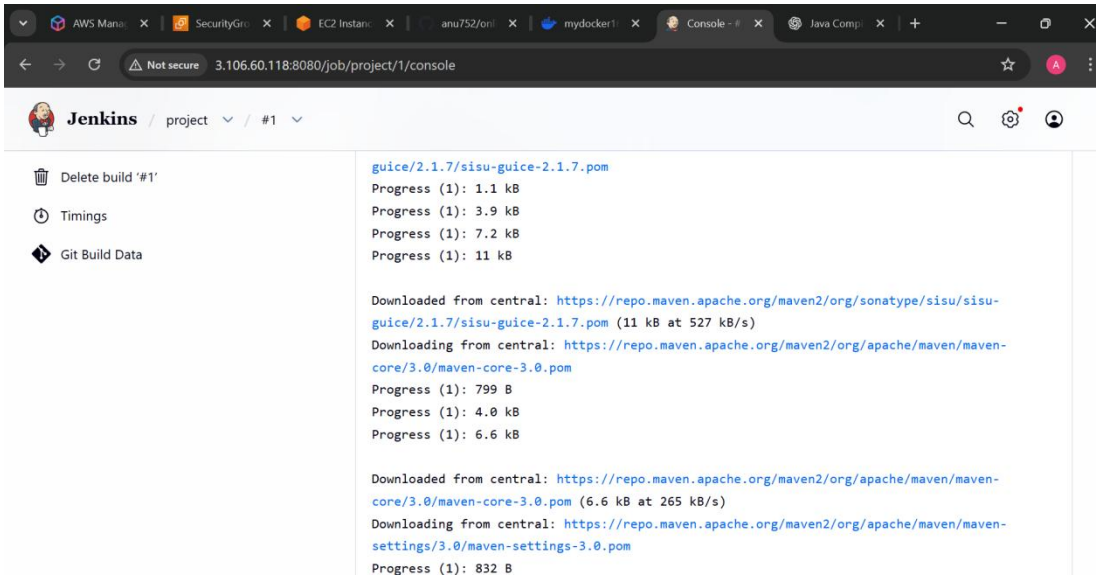
Projects

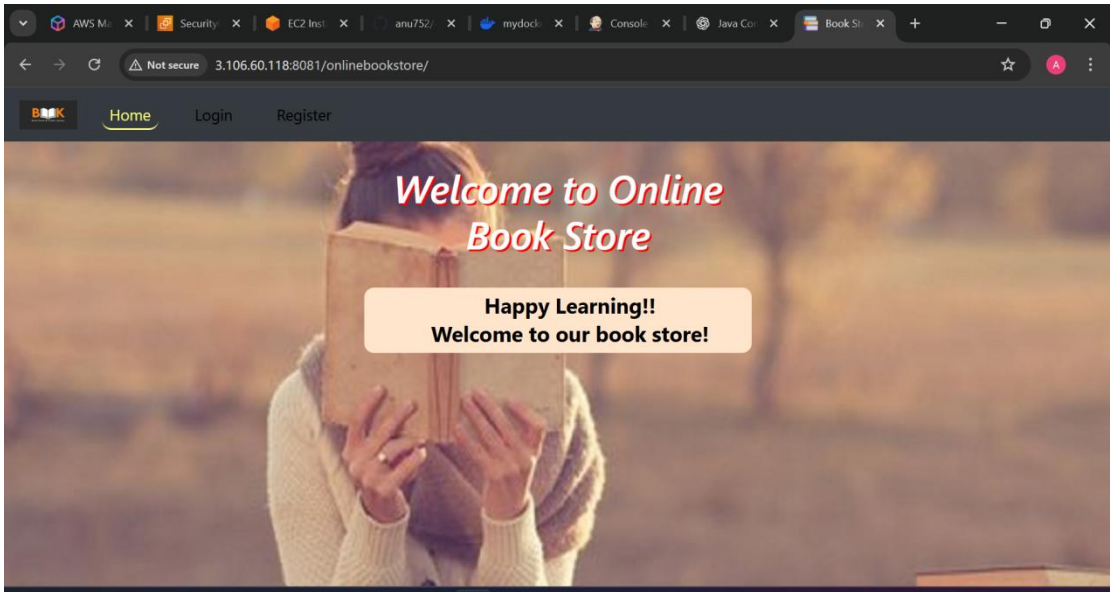
Project – 1 : Automated Cloud-Native CI/CD Pipeline

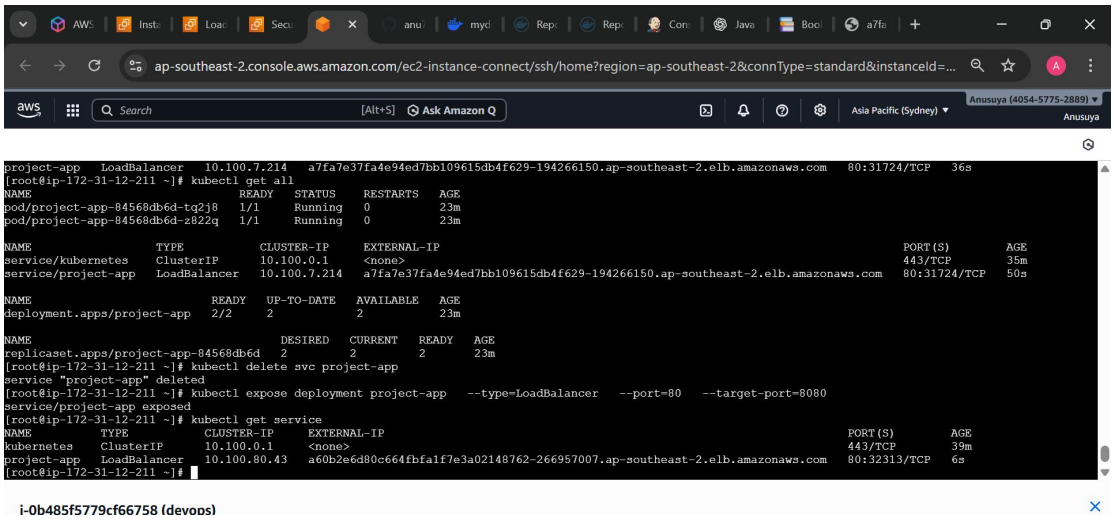
This project focuses on building an automated CI/CD pipeline to deploy cloud-native applications in a consistent and reliable manner. Source code was managed using GitHub, and Jenkins was used to automate build and deployment processes. Docker was used to containerize the application, and Kubernetes handled deployment and scalability on AWS infrastructure.

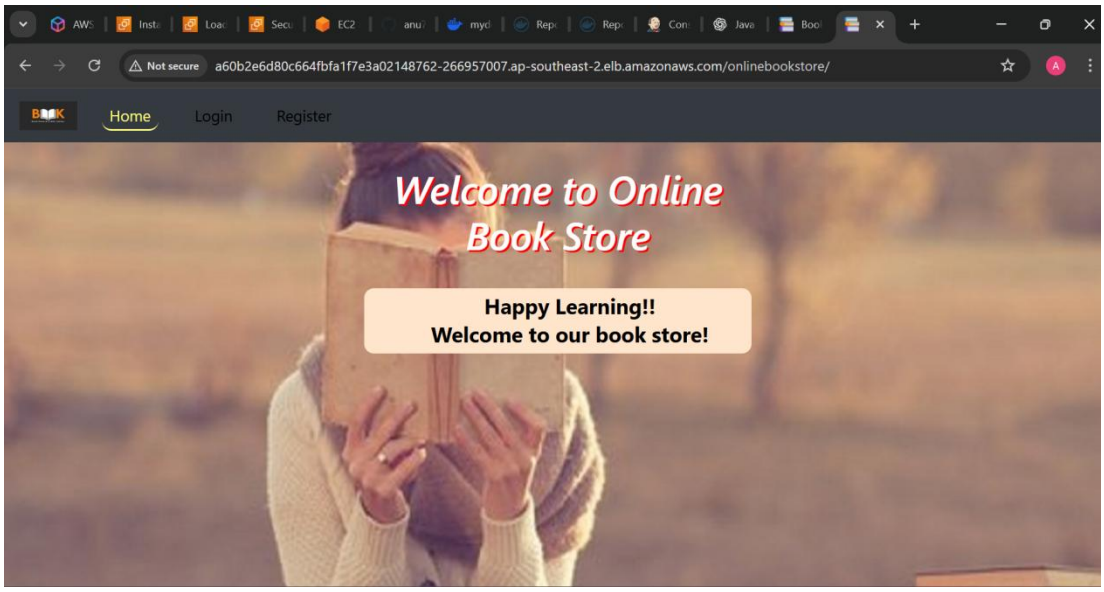
Tools Used: GitHub, Jenkins, Docker, Kubernetes, AWS









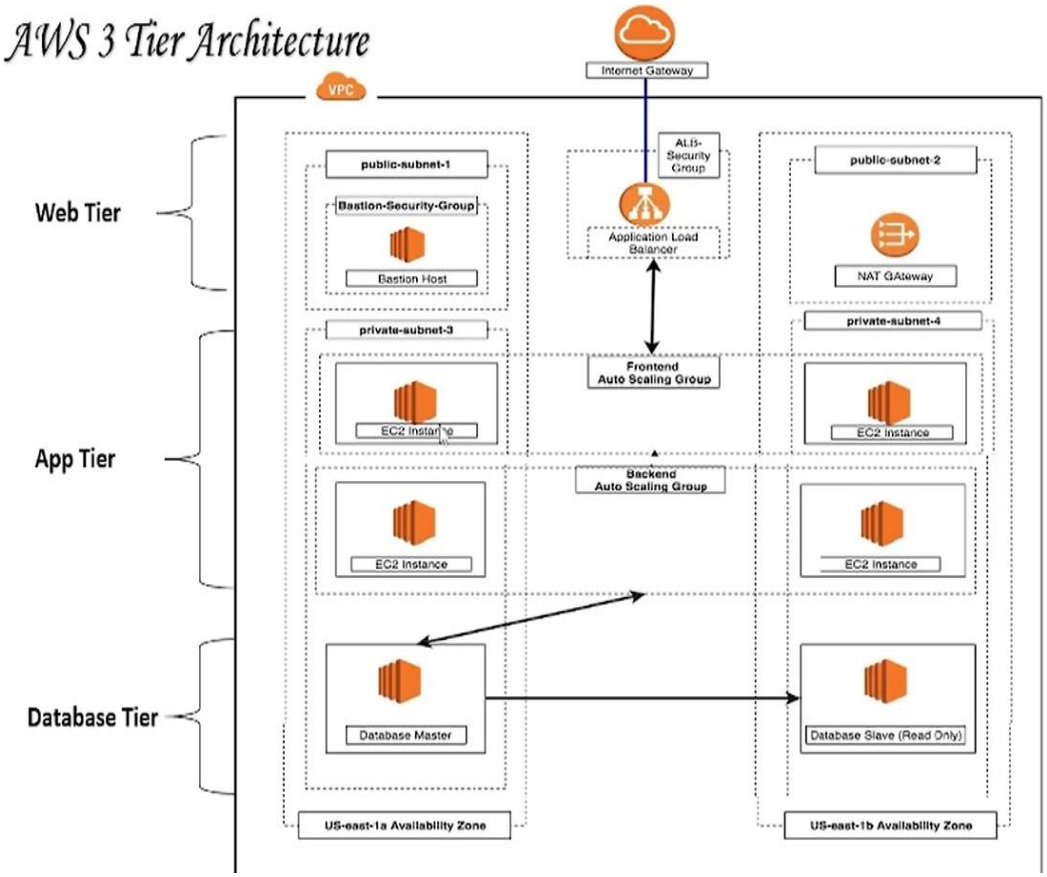


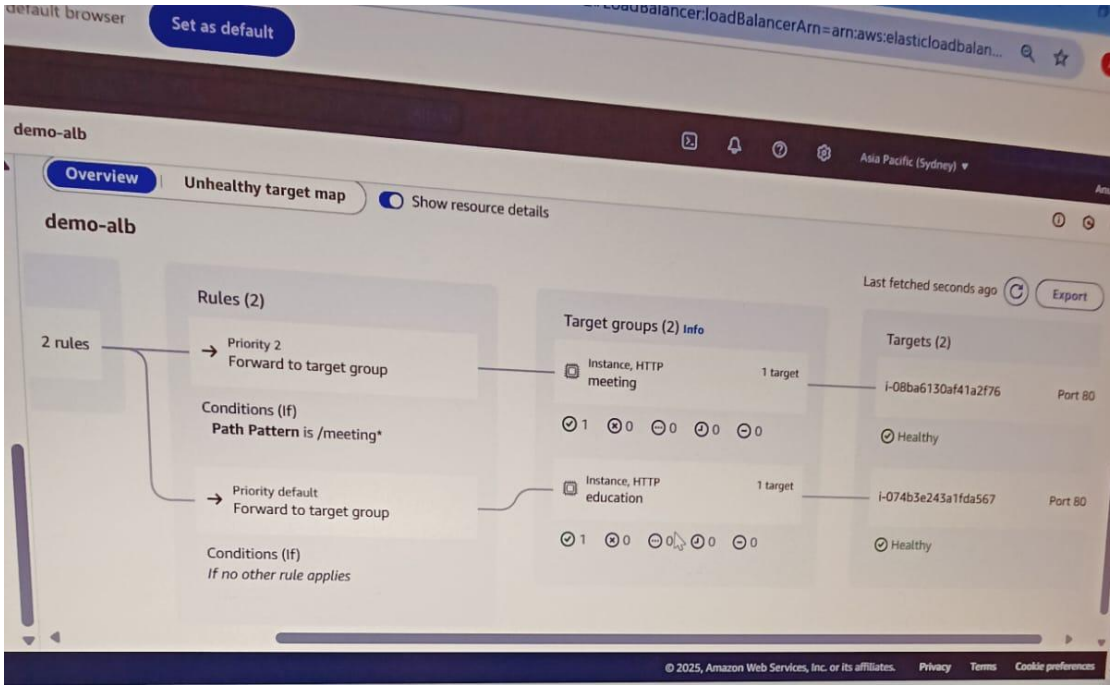
Outcome:
Improved deployment speed, reduced manual effort, and achieved reliable and scalable application delivery.

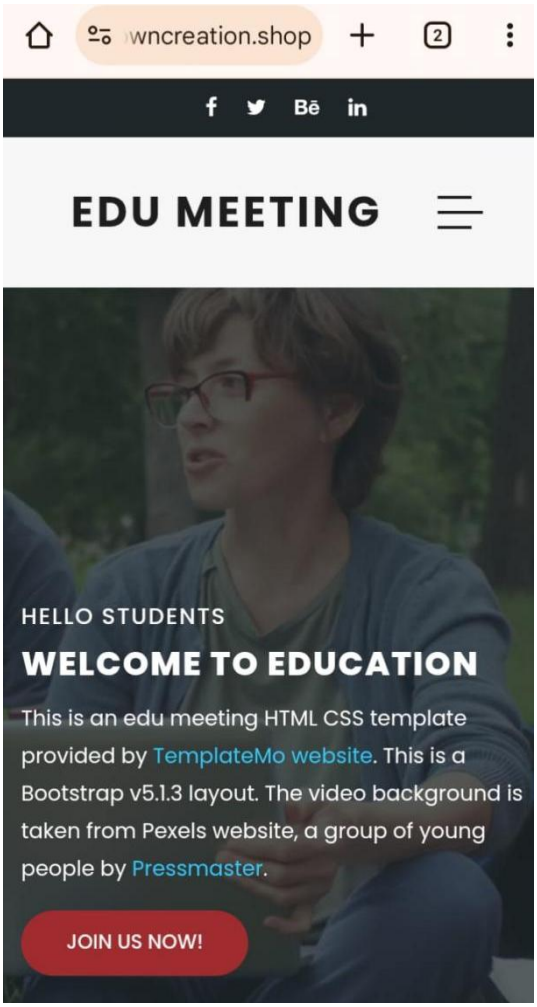
Project – 2: AWS 3-Tier Architecture Implementation

In this project, a secure and scalable 3-tier architecture was designed and implemented on AWS. The architecture separates web, application, and database layers using public and private subnets. Load Balancers and Auto Scaling ensured high availability, while security groups and IAM improved overall security.

Tools Used: AWS EC2, VPC, ALB, Auto Scaling, RDS, IAM, Route53


AWS 3 Tier Architecture
Internet Gateway
VPC
Web Tier
App Tier
Database Tier
public-subnet-1
Bastion-Security-Group
Bastion Host
ALB-Security Group
Application Load Balancer
public-subnet-2
NAT GAteway
private-subnet-3
Frontend Auto Scaling Group
private-subnet-4
EC2 Instance
Backend Auto Scaling Group
EC2 Instance
EC2 Instance
EC2 Instance
Database Master
Database Slave (Read Only)
US-east-1a Availability Zone
US-east-1b Availability Zone


Set as default
demo-alb
Overview
Unhealthy target map
Show resource details
demo-alb
Asia Pacific (Sydney)
Last fetched seconds ago
Export
Rules (2)
2 rules
Priority 2
Forward to target group
Conditions (If)
Path Pattern is /meeting*
Priority default
Forward to target group
Conditions (If)
If no other rule applies
Target groups (2) Info
Instance, HTTP
meeting
1 target
Instance, HTTP
education
1 target
Targets (2)
i-08ba6130af41a2f76
Port 80
Healthy
i-074b3e243a1fda567
Port 80
Healthy
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

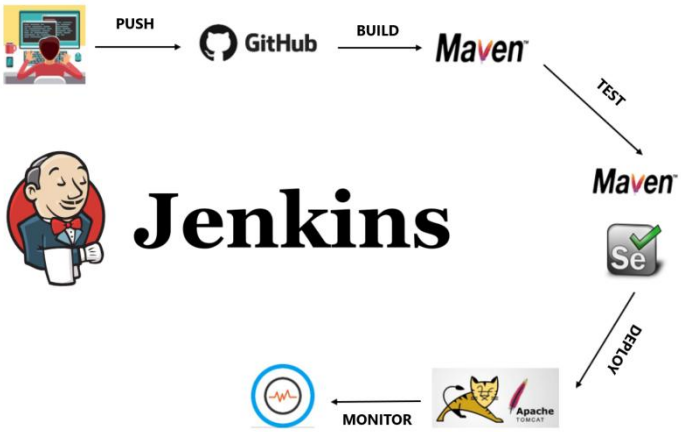


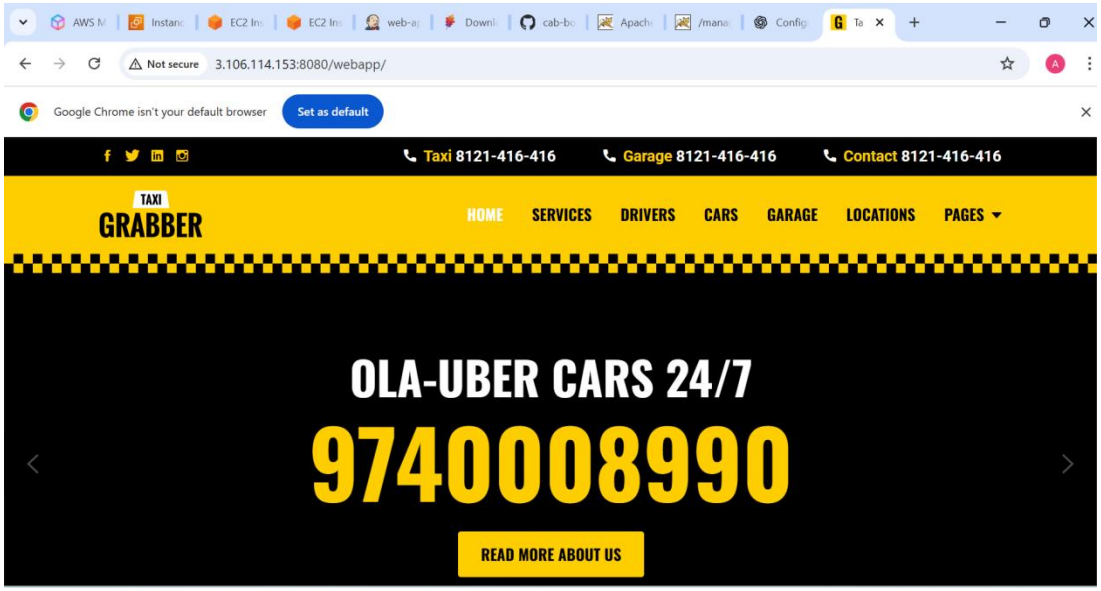
Outcome:

Achieved high availability, improved security, and better performance through layered architecture design.

Project- 3 : Jenkins-Based CI/CD Pipeline for Java Web Application
This project focuses on implementing an automated CI/CD pipeline for a Java-based web application using Jenkins. The pipeline triggers automatically when code is pushed to GitHub, builds and tests the application using Maven and Selenium, and deploys the application to an Apache Tomcat server.

Tools Used: GitHub, Jenkins, Maven, Selenium,Apache Tomcat, Linux



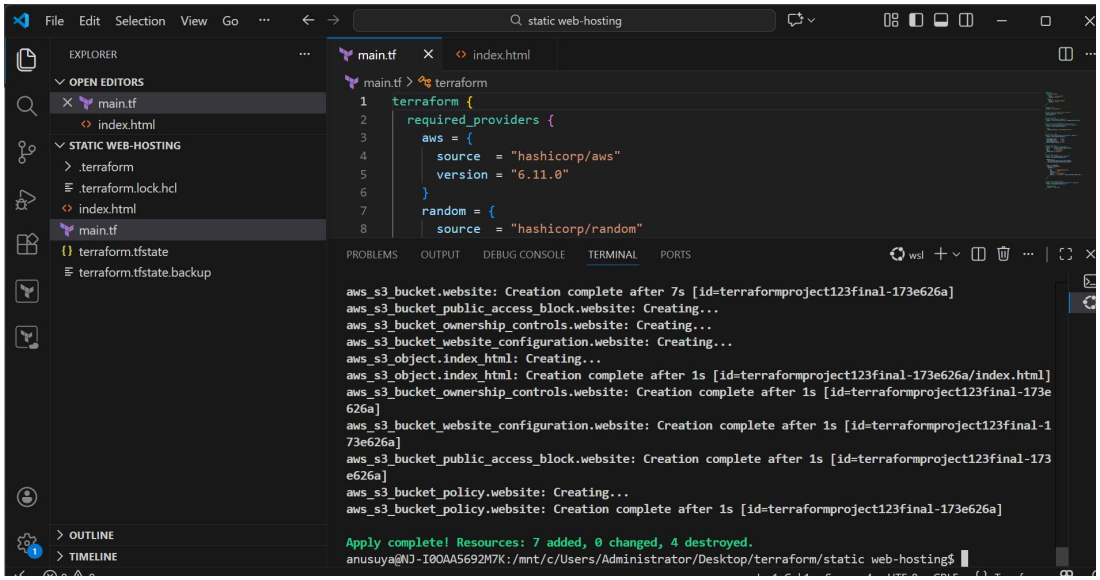


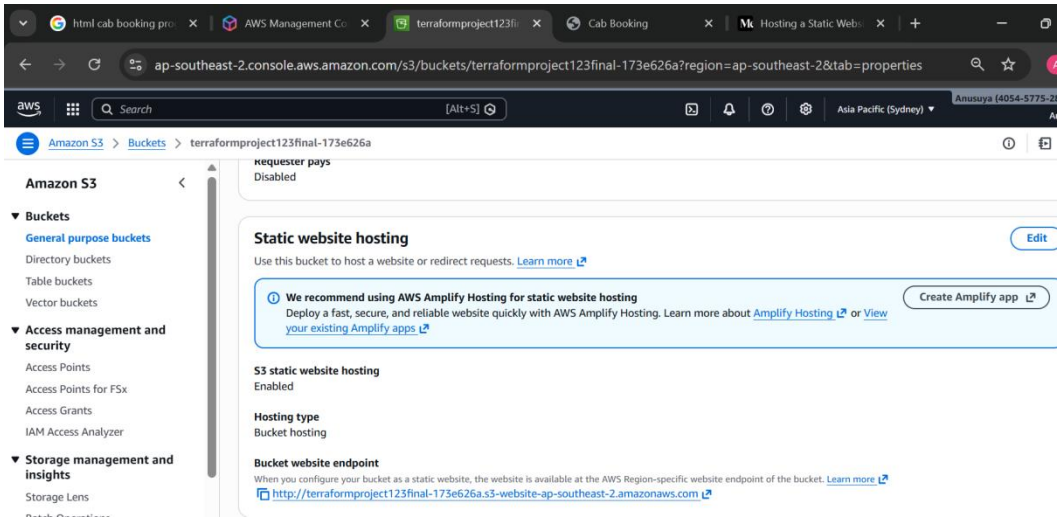
Outcome:
Automated build, test, and deployment process, reduced manual intervention, and improved reliability and delivery speed of the application.

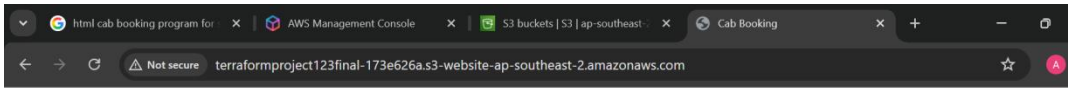
Project – 4: Hosting a Static Website on AWS S3 with Terraform

This project demonstrates the use of Infrastructure as Code, a simple and efficient way to host a basic website using Terraform and Amazon Web Services (AWS) S3. The main aim of this project is to show how to automatically create and deploy a website without much hassle.

Tools Used: AWS S3, Terraform, VS code







Book Your Ride

Pickup Location:
Drop-off Location:
Pickup Date & Time: dd-----yyyy --:-- --
Vehicle Type: Sedan
Book Now

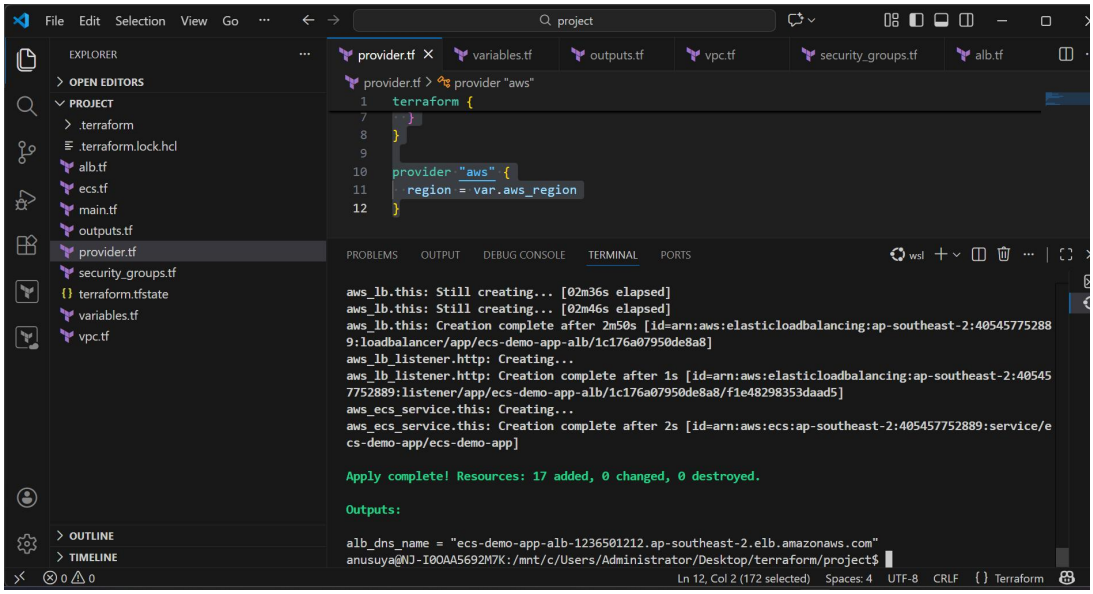
Outcome:

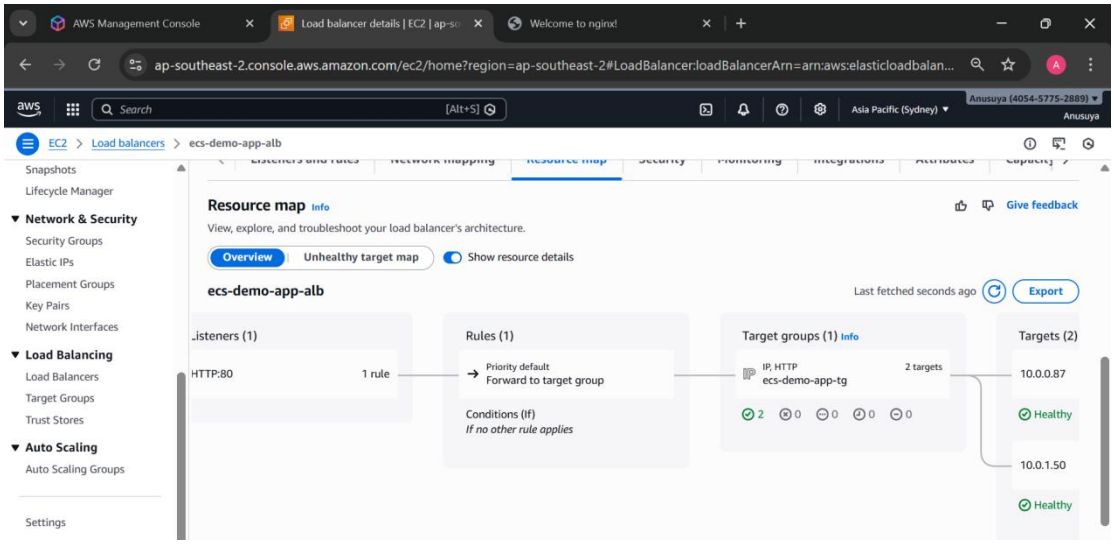
Automatically configured static website hosting, bucket policies, and public access successfully on AWS S3 bucket using Terraform without the need to access the AWS Console.

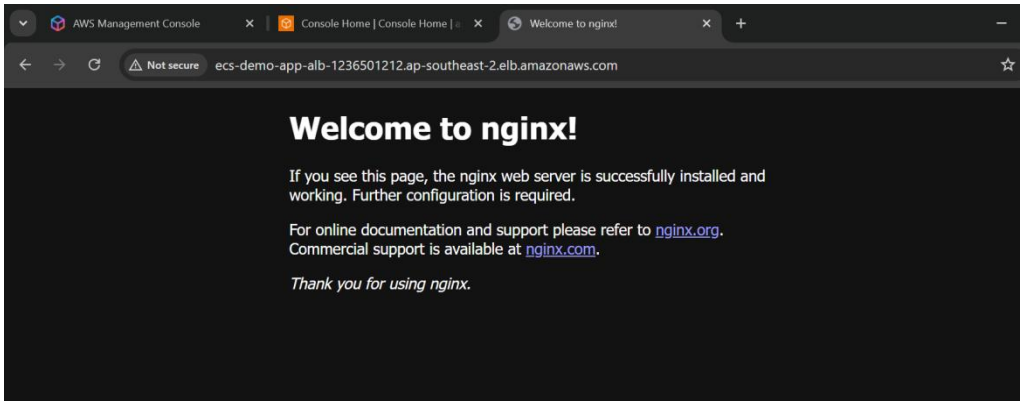
Project 5: Terraform — Deploy Dockerized App on AWS ECS with Load Balancer

This project demonstrates how to deploy a Dockerised app on Amazon ECS. Amazon ECS is the AWS service to run Docker applications on a scalable cluster. It is a container orchestration/ management service similar to Kubernetes

Tools Used: AWS EC2, ELB, IAM, ECS, VPC, Terraform, vs code







Outcome:

This project demonstrated the process of deploying a Dockerized application on Amazon ECS using Terraform. It covered setting up ECS resources, and configuring networking.we can automate the deployment of containerized apps on ECS for scalable and reliable infrastructure on AWS.

Learning Outcomes

- Practical exposure to real-world DevOps workflows
- Strong understanding of cloud infrastructure and automation
- Improved troubleshooting and problem-solving skills
- Hands-on experience with CI/CD, IaC, and monitoring tools

Conclusion

These projects strengthened my understanding of Cloud and DevOps principles and provided hands-on experience in designing, deploying, and automating modern cloud infrastructures. The work aligns with industry standards and prepares me for entry-level Cloud or DevOps roles.