Scalable Web Application Deployment on AWS

Project Report

Name: SAMAR DHAWAN

Abstract

This project demonstrates the deployment of a secure and scalable website using Amazon Web Services (AWS). The architecture includes EC2 instances in an Auto Scaling Group behind an Elastic Load Balancer for high availability. Route 53 manages DNS, and SSL ensures secure communication. CloudWatch monitors resources and sends alerts via AWS Simple Email Service (SES), providing reliability, scalability, and security with minimal manual effort.

Introduction & Objective

Introduction

In today's digital era, deploying websites on cloud platforms ensures better scalability, availability, and cost-effectiveness compared to traditional hosting. Amazon Web Services (AWS) provides a range of cloud-based solutions to build highly reliable and secure infrastructures for hosting websites. This project focuses on implementing a scalable and secure website using AWS services such as EC2, Auto Scaling, Elastic Load Balancer, Route 53, and CloudWatch. The goal is to ensure the website can handle varying traffic loads while maintaining high performance and uptime.

Objective

The main objectives of this project are:

- To deploy a **secure and scalable website** using AWS infrastructure.
- To configure **Auto Scaling** and **Load Balancing** for handling traffic fluctuations.
- To implement **DNS management** using Route 53 and enable **SSL encryption** for secure communication.
- To set up **monitoring and alerting** using CloudWatch and SES.
- To demonstrate a **cost-effective, reliable, and automated cloud hosting solution**.

Architecture Diagram

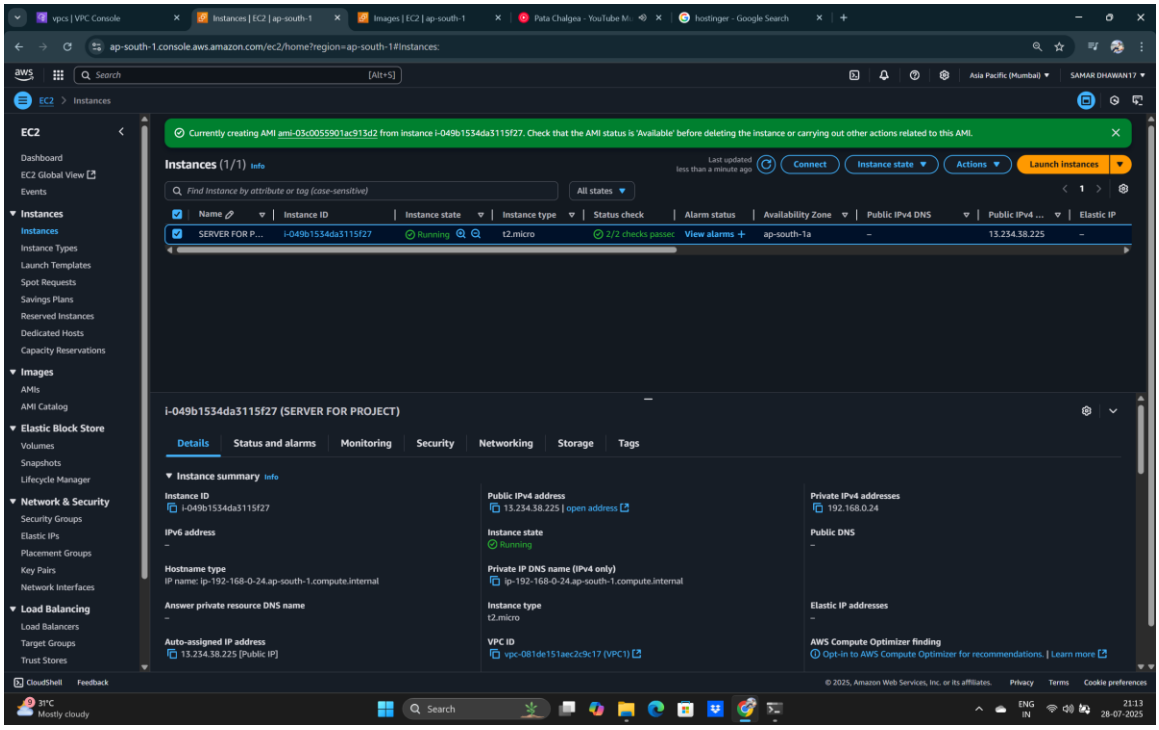
AWS Services Used

Service	Purpose
VPC (Virtual Private Cloud)	Creates an isolated network environment to securely host AWS resources.
EC2 (Elastic Compute Cloud)	Hosts the website on virtual servers with customizable compute capacity.
Auto Scaling	Automatically adjusts the number of EC2 instances based on traffic load.
Elastic Load Balancer (ELB)	Distributes incoming traffic across EC2 instances for high availability.
Route 53	Provides DNS routing and domain management; directs users to the website.
CloudWatch	Monitors system performance; sends alerts based on defined metrics.
SES (Simple Email Service)	Sends email notifications for alerts like instance health or scaling events.

---

implementation Steps

Step 1: Launch EC2 Instance



Step 1: Launch EC2 Instance and Deploy HTML Code

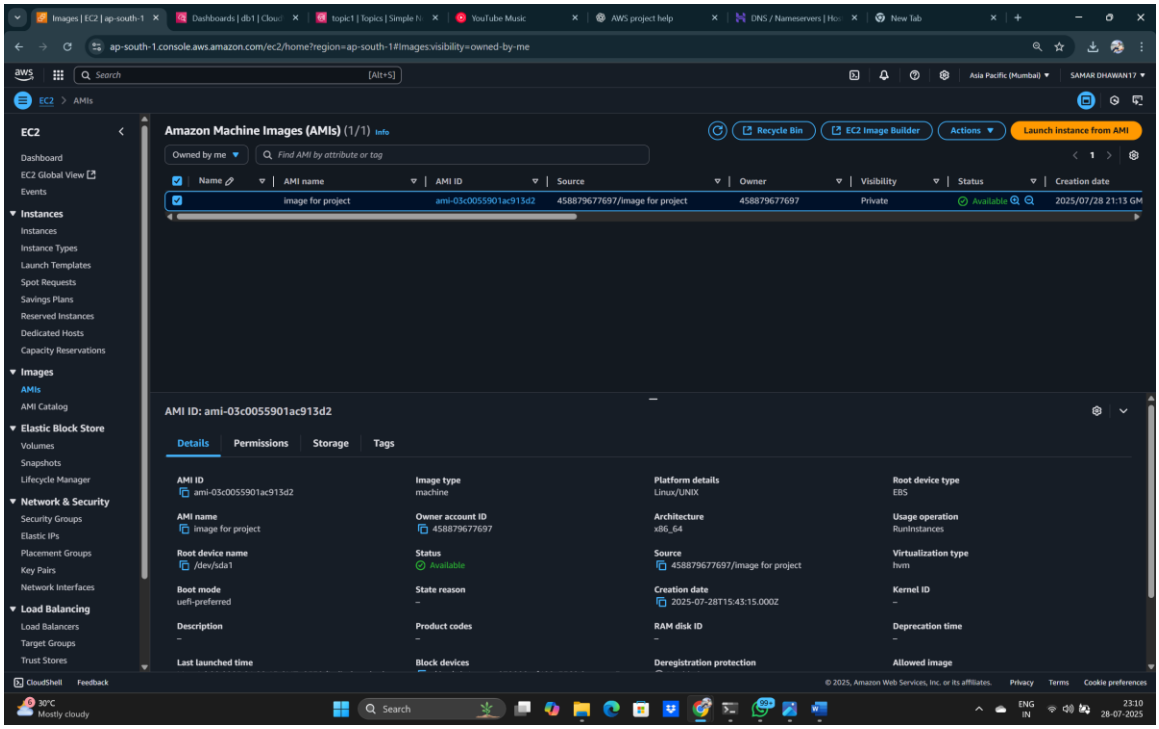
Launch an EC2 instance (Amazon Linux 2 or Ubuntu).

SSH into it and install a web server (like Apache or Nginx).

Deploy your HTML code in /var/www/html.

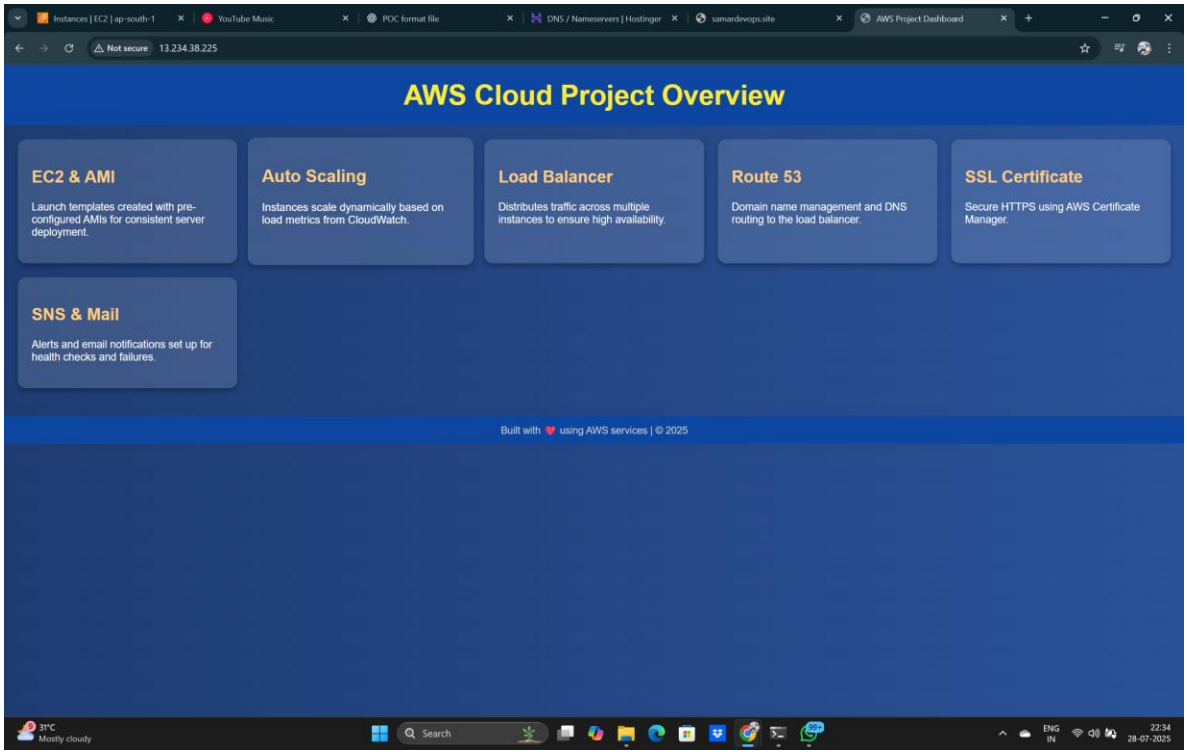
[I have taken ubuntu and apache web server and taken a simple aws project html code]

Step 2: Create an AMI (Image)



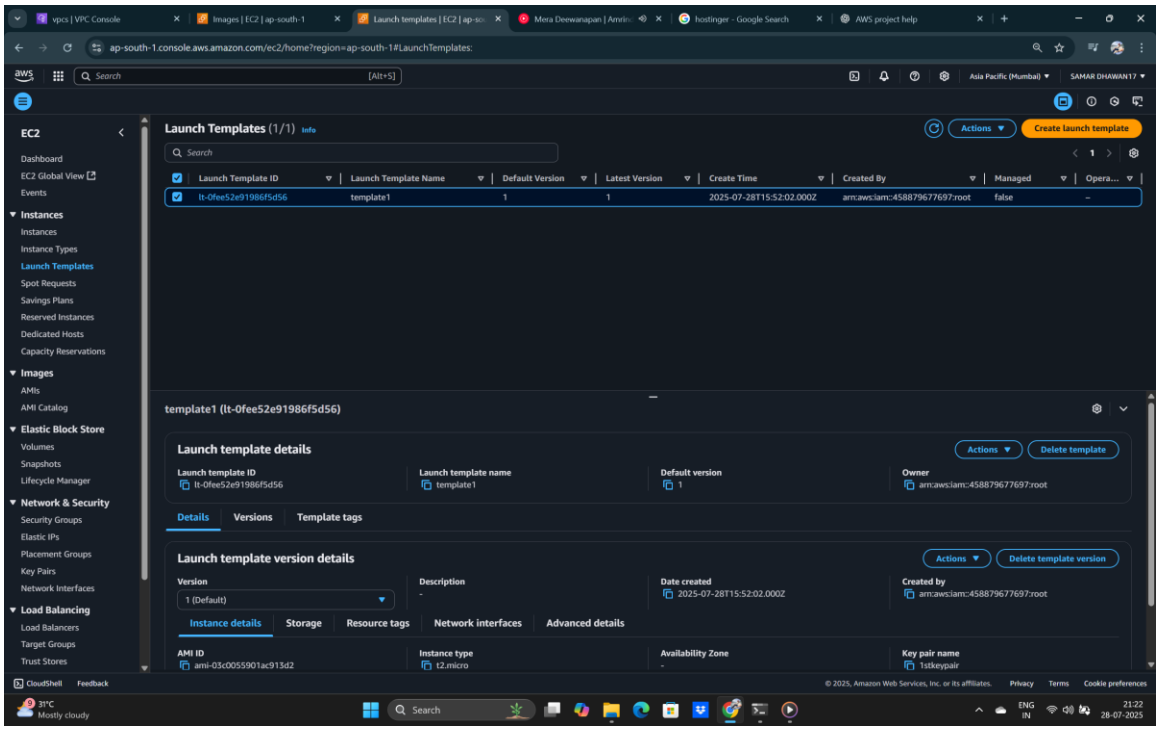
After configuring the EC2 instance, create an **AMI** (Amazon Machine Image) of it.

- Go to EC2 → Instances → Select → Actions → Image → Create Image.

- 

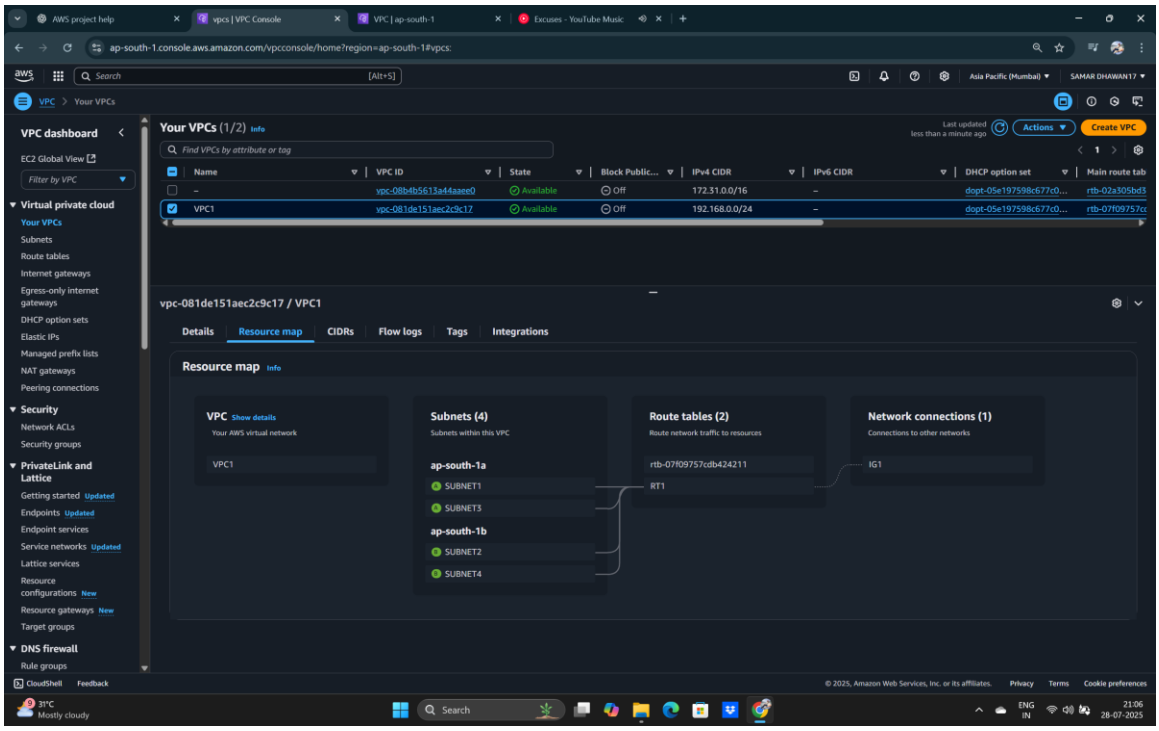
THIS IS HOW WEBSITE LOOK AFTER DEPLOYMENT

Step 3: Create a Launch Template



- **Use the created AMI to define a Launch Template.**
    - **Go to EC2 → Launch Templates → Create.**
    - **Select your AMI, instance type, key pair, security group, and other config.**

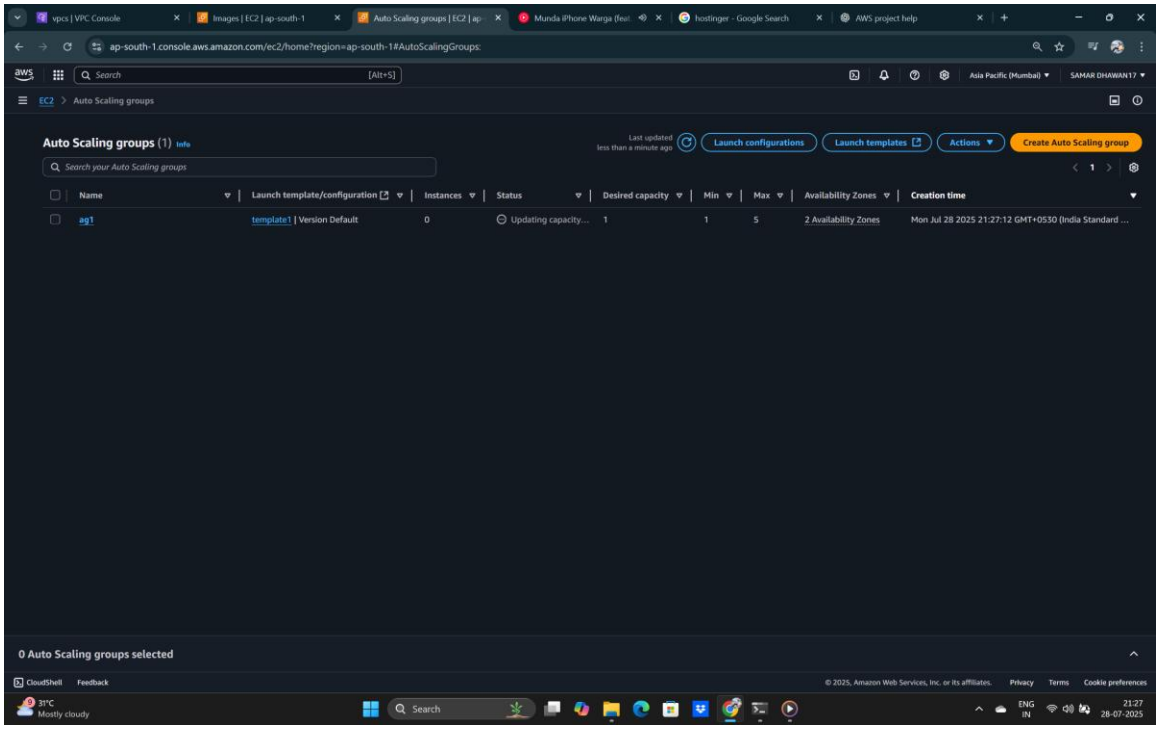
Step 4: Set Up Networking



- **VPC** (1)
- **Subnets** (4 - for HA across AZs)
- **Internet Gateway** (IGW)
- **Route Table (RT)** with IGW attached
- Associate subnets with the RT
- Enable auto-assign public IP in public subnets

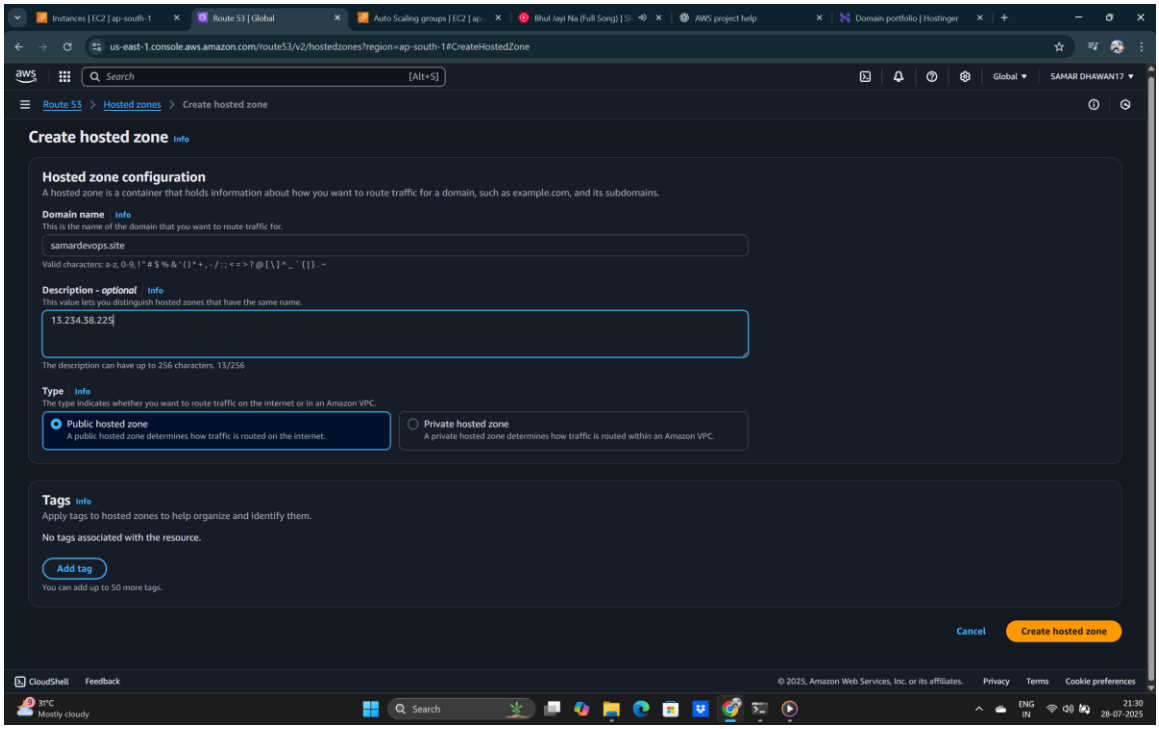
Step 5: Create Auto Scaling Group

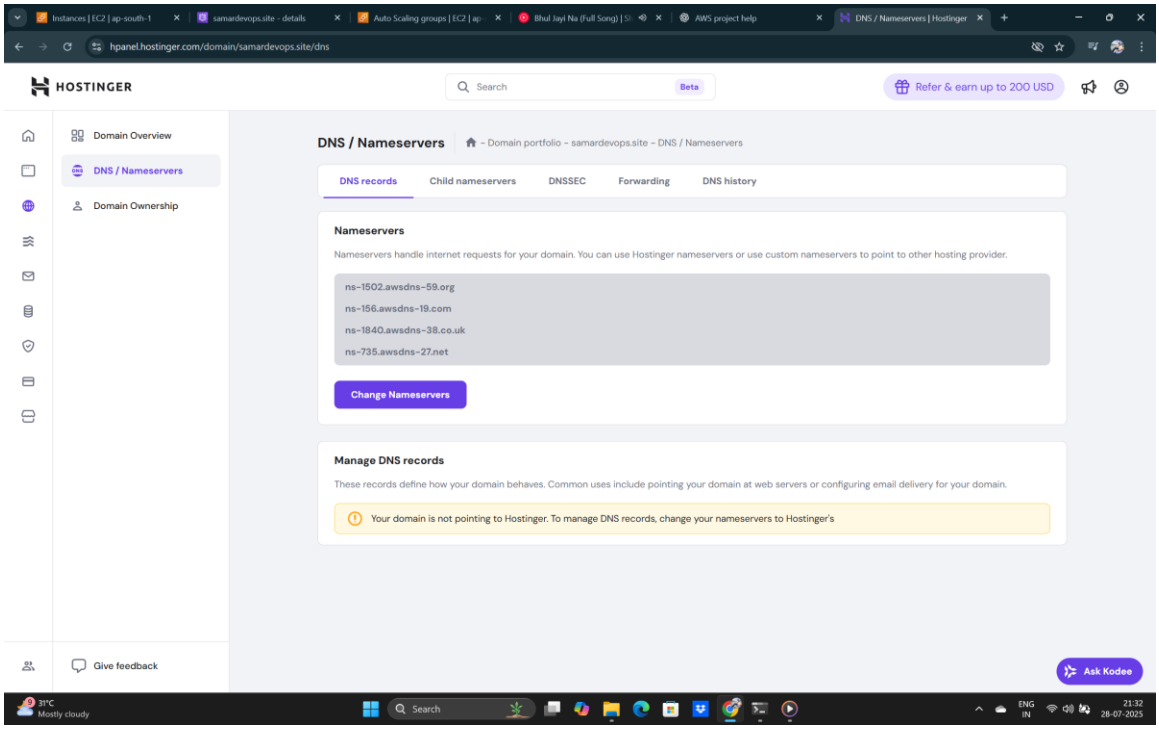
[

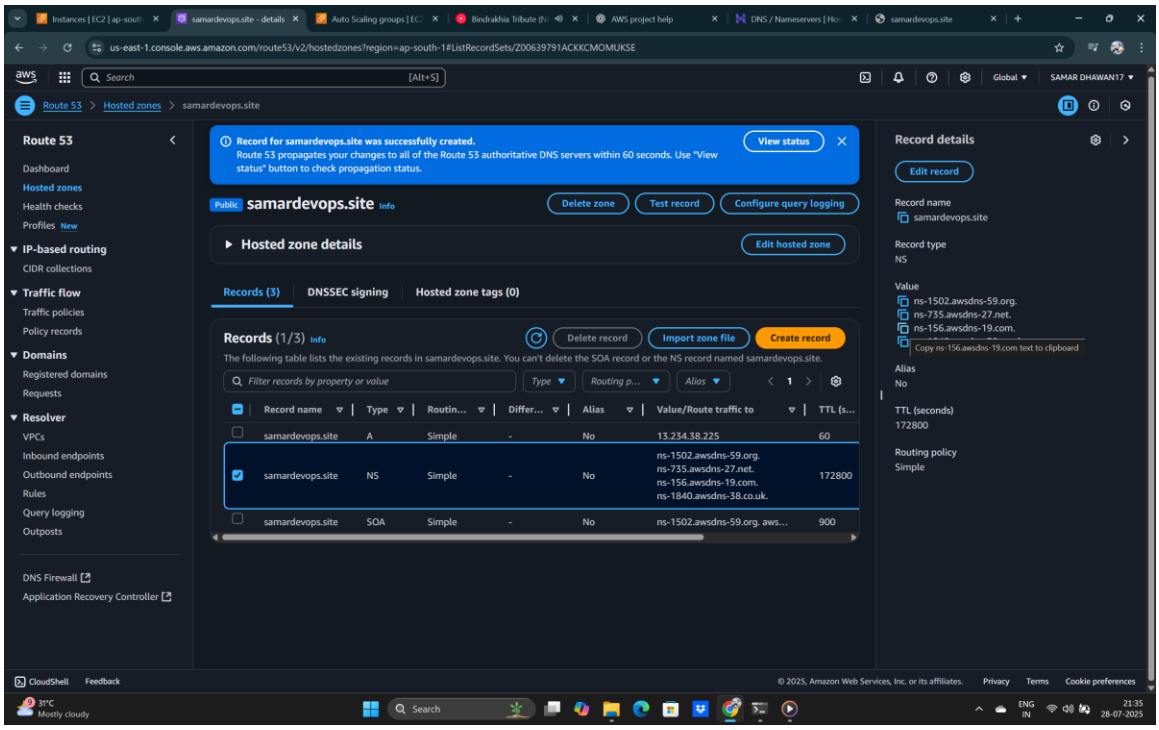


- Use the Launch Template.
- Set minimum, maximum, and desired capacity.
- Attach the 4 subnets.
- Enable health checks (ELB/EC2).
- Attach a Load Balancer.

Step 6: Configure Domain with Route 53

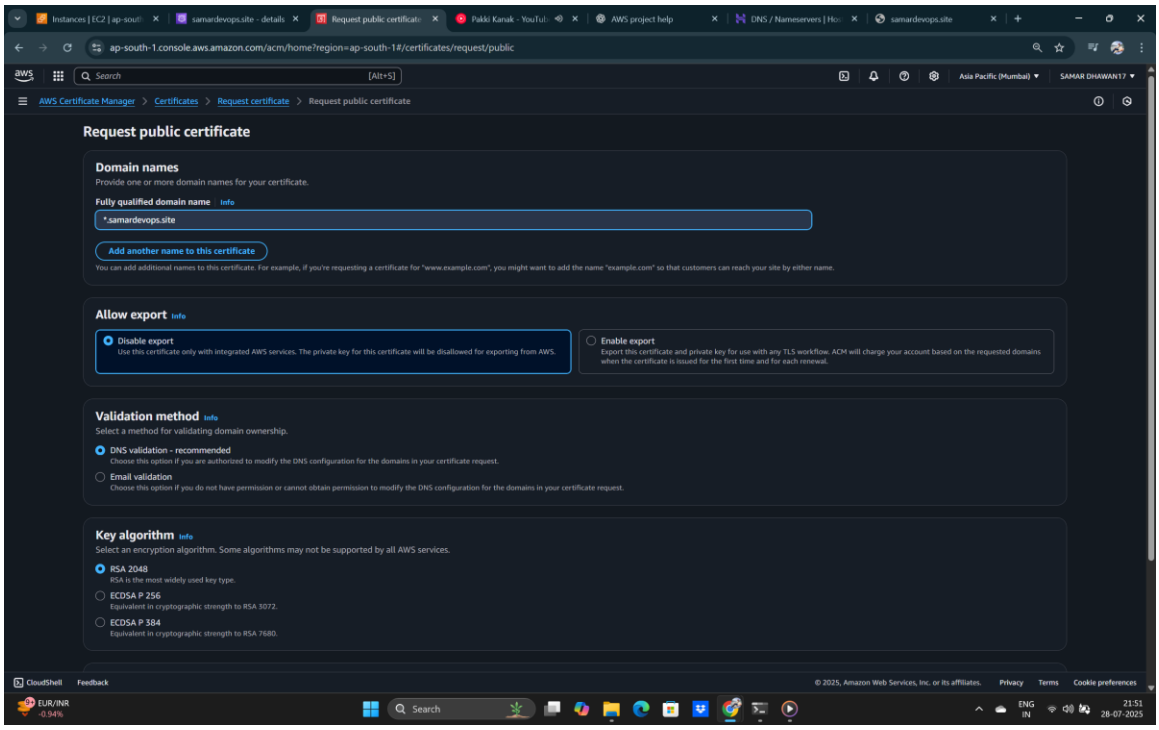


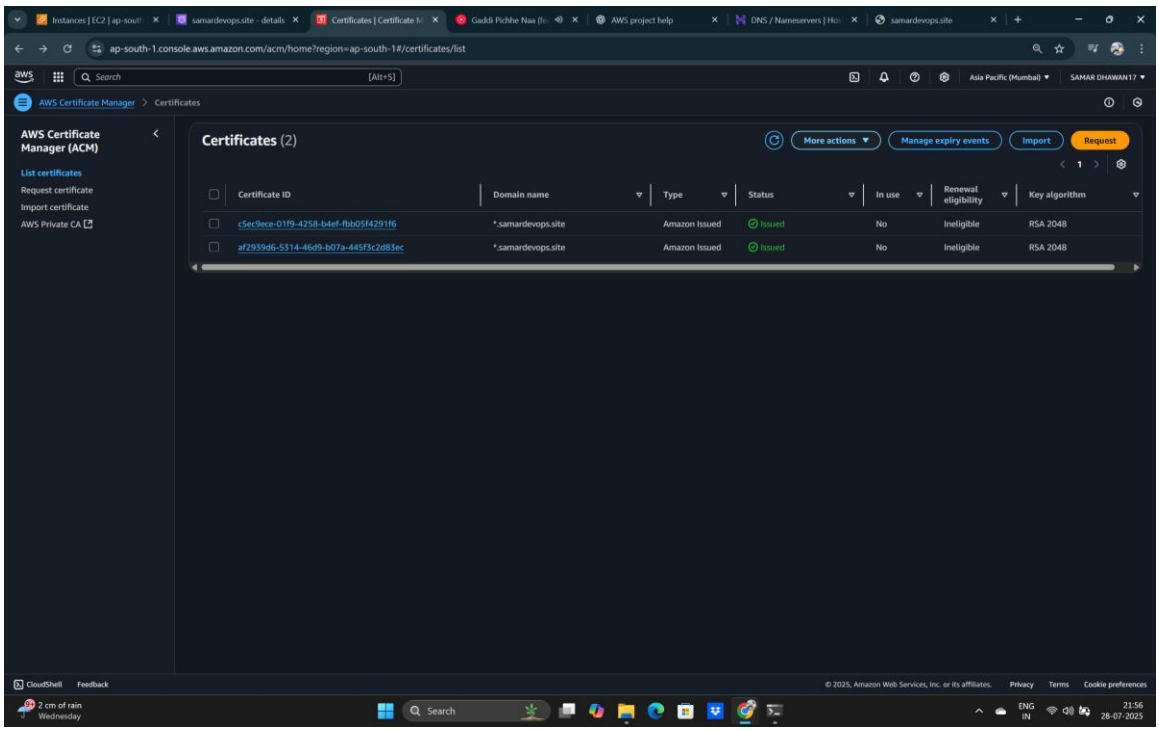




- Register a domain or use an existing one in **Route 53**.
- Create a **Hosted Zone**.
- Add an **A record** pointing to the **Load Balancer DNS**.

Step 8: Add SSL with AWS Certificate Manager

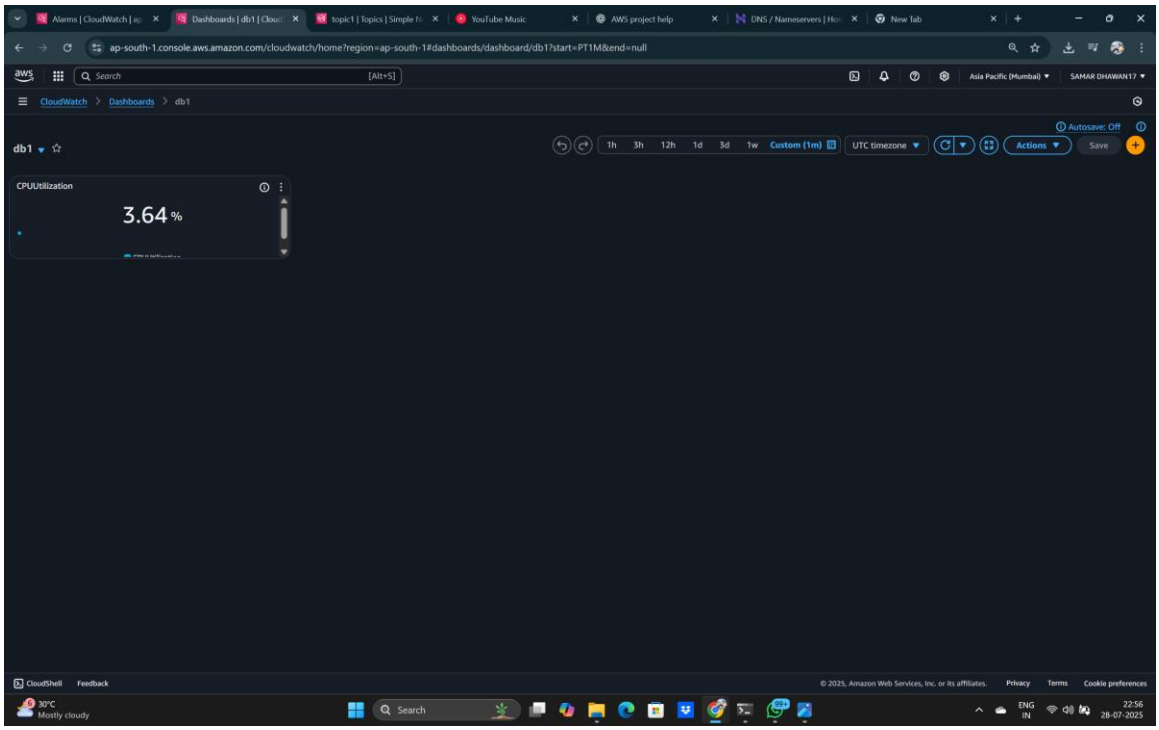


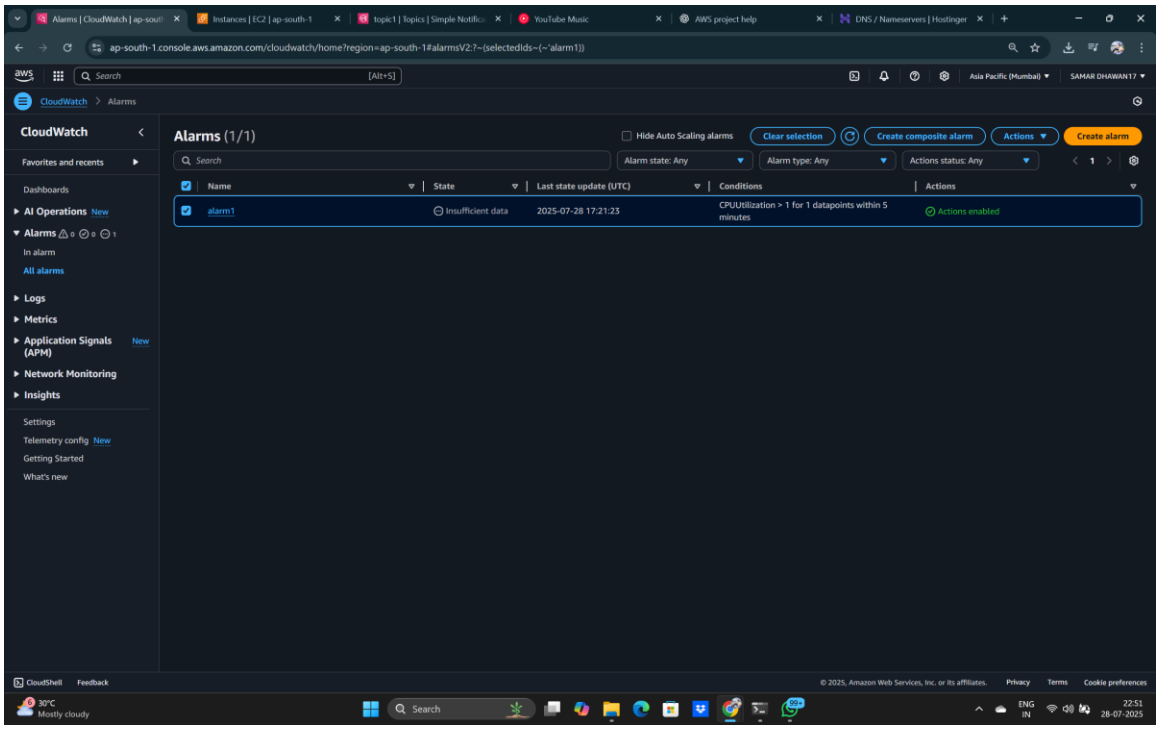


- Go to **Certificate Manager**.

- Request a public certificate for your domain.
- Validate via DNS (Route 53 makes it easy).
- Attach the certificate to your **Load Balancer** (HTTPS listener)

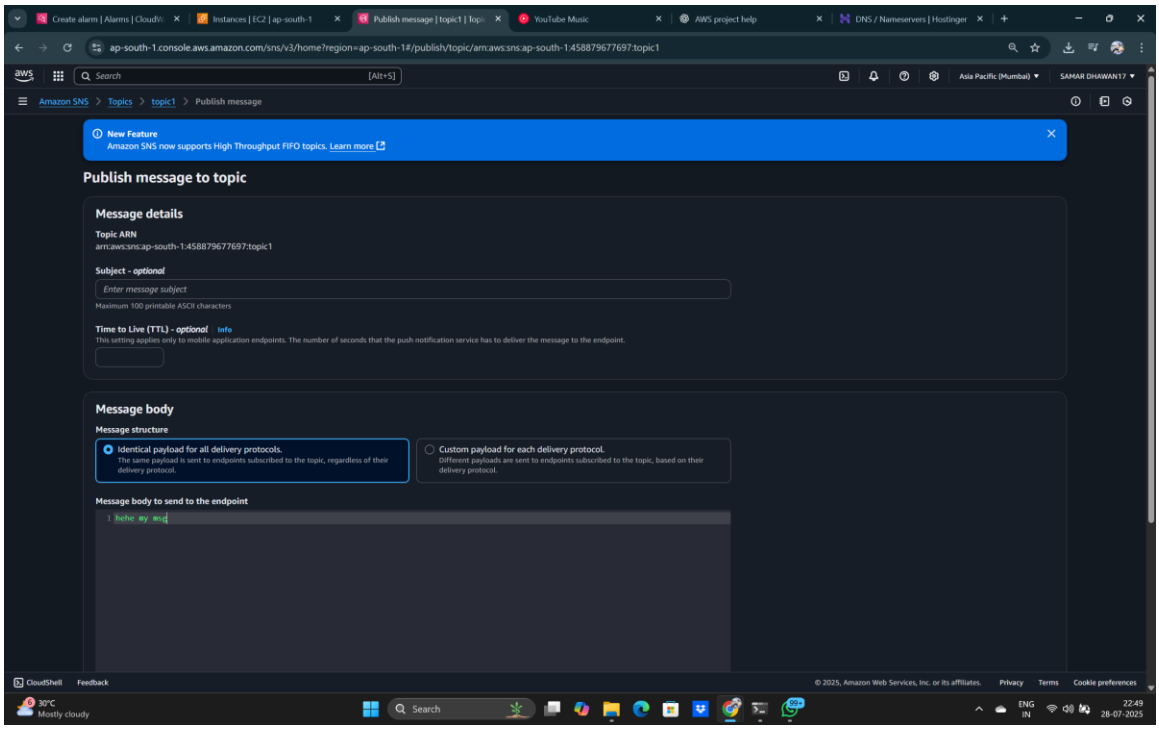
Step 9: Set Up CloudWatch





- Create alarms for metrics like CPU usage.
- Connect alarms to Auto Scaling policies (scale out/in).
- Set thresholds (e.g., CPU > 80%).

Step 10: Configure SNS & Email Alerts



- Create an SNS topic (e.g., cpu-alerts).
- Subscribe your email to it.
- Attach the topic to your CloudWatch alarms for notifications

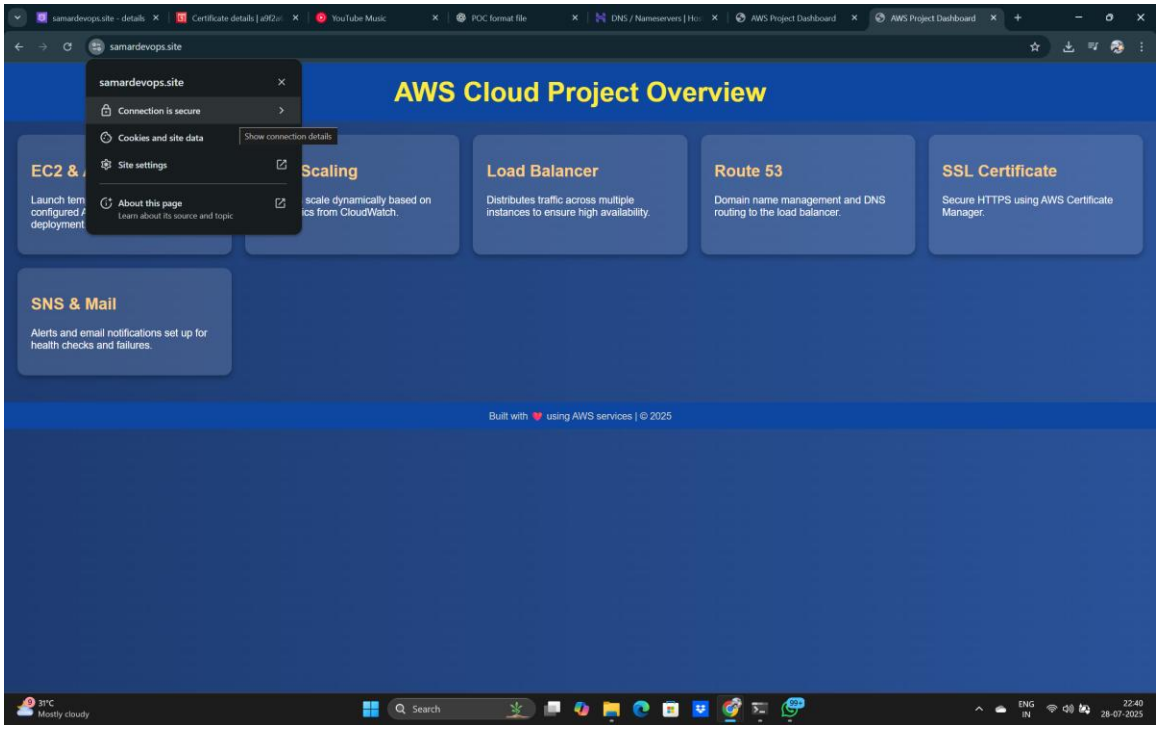
Project Conclusion

This project successfully demonstrates the design and deployment of a **highly available, scalable, and secure web infrastructure** on AWS. By integrating key services such as **VPC, EC2, Auto Scaling, Elastic Load Balancer, Route 53, CloudWatch**, and **SES**, the architecture ensures:

- **High Availability** through multiple subnets across Availability Zones and Load Balancer distribution.
- **Scalability** with Auto Scaling automatically adjusting resources based on traffic.
- **Security & Isolation** using VPC with proper subnet segmentation and routing.
- **Performance Monitoring** via CloudWatch for proactive alerting and system health tracking.
- **Efficient DNS & SSL Management** using Route 53 and Certificate Manager.

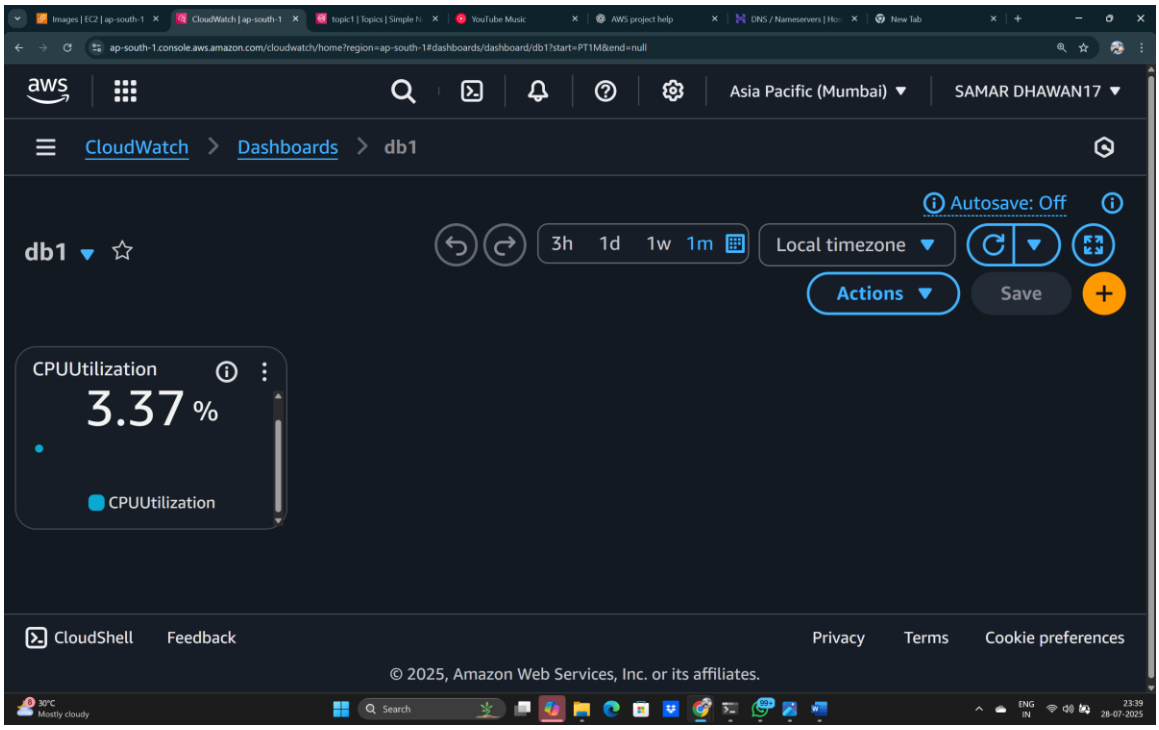
- **Automated Notifications** with SES for operational alerts and system events.

Overall, this cloud-based architecture provides a **robust, fault-tolerant solution** suitable for hosting modern web applications with minimal manual intervention and maximum reliability.

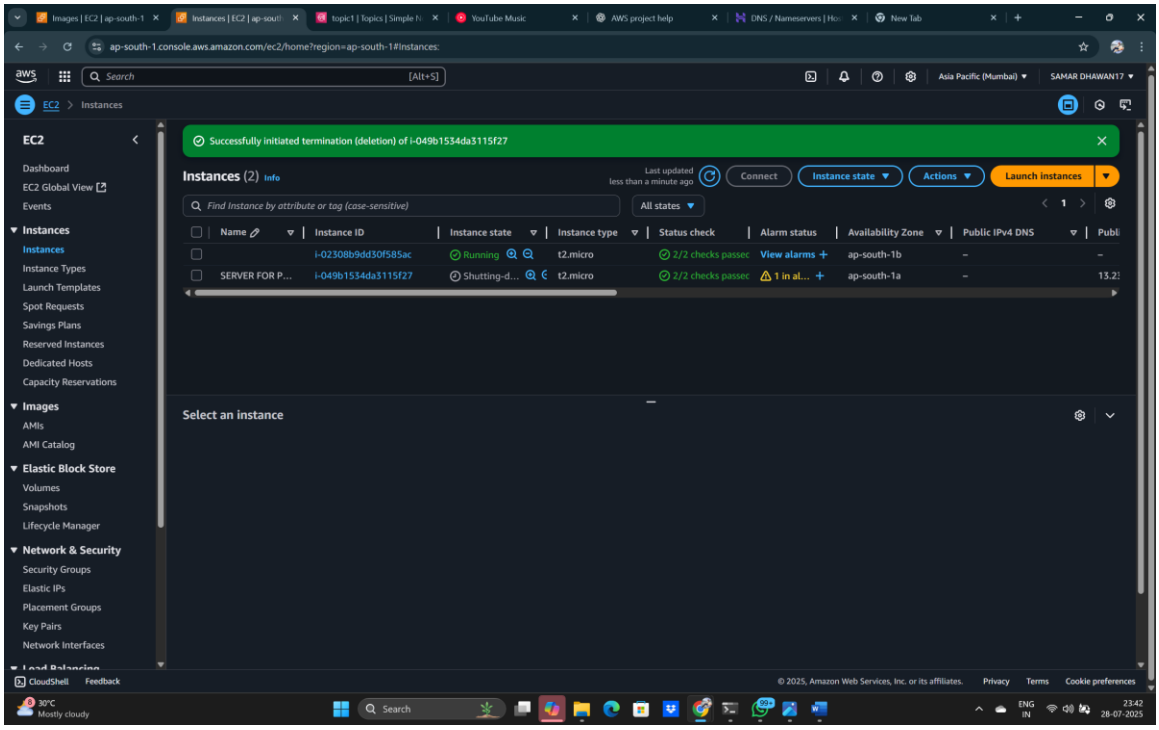


The website is now secure

SNS SERVICE IS WORKING AND ACTIVE



CLOUD WATCH IS CONTINOUSLY MONITORING CPU UTILIZATION



Auto scalling automatically adjusts the capacity of your AWS resources to handle fluctuating application loads, ensuring optimal performance and cost-efficiency