Design and Implementation of an Auto-Scalable Web Application on AWS Using Load Balancer, CloudWatch and SNS

Introduction

This project focuses on designing and implementing an auto-scalable web application infrastructure using Amazon Web Services. The system dynamically adjusts computing resources based on real-time demand, ensuring high availability, performance efficiency, and cost optimization. By integrating load balancing, monitoring, and notification services, the architecture demonstrates a practical cloud solution capable of handling variable traffic loads without manual intervention.

Problem Statement

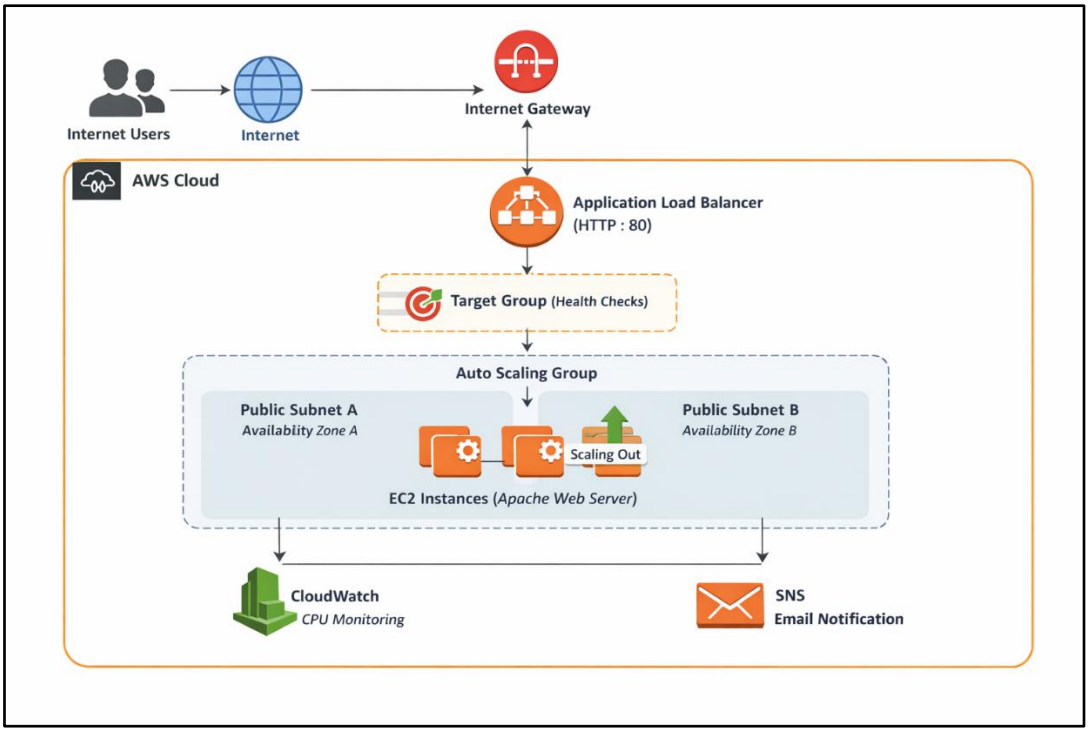
Traditional web application infrastructures struggle to efficiently manage sudden increases or decreases in user traffic. Over-provisioning leads to unnecessary costs, while under-provisioning causes performance degradation and downtime. The problem addressed in this project is to design a cloud-based infrastructure that can automatically scale resources in response to workload changes while maintaining availability, reliability, and minimal operational overhead.

Objective

- Deploy a web application on EC2
- Distribute traffic using an Application Load Balancer
- Automatically scale EC2 instances based on CPU usage
- Monitor system performance using CloudWatch
- Send real-time notifications through SNS

Architecture Overview

The architecture consists of an internet-facing Application Load Balancer that distributes incoming HTTP requests across multiple EC2 instances. These instances are managed by an Auto Scaling Group, which launches or terminates instances based on CPU utilization. CloudWatch continuously monitors performance metrics and triggers scaling actions, while SNS sends notifications during significant scaling events. This design ensures fault tolerance, scalability, and consistent application availability.

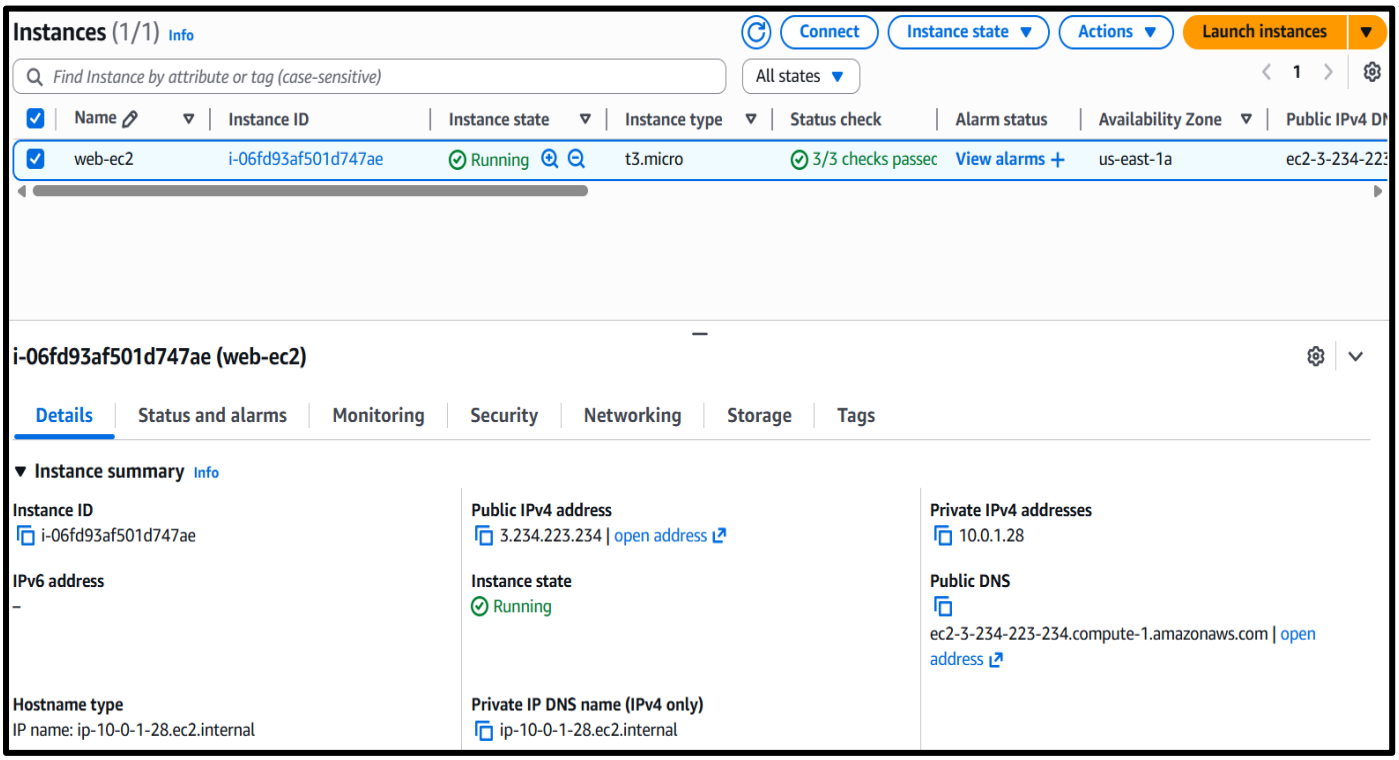


AWS Services Used

The following Amazon Web Services were used in the implementation of this project:

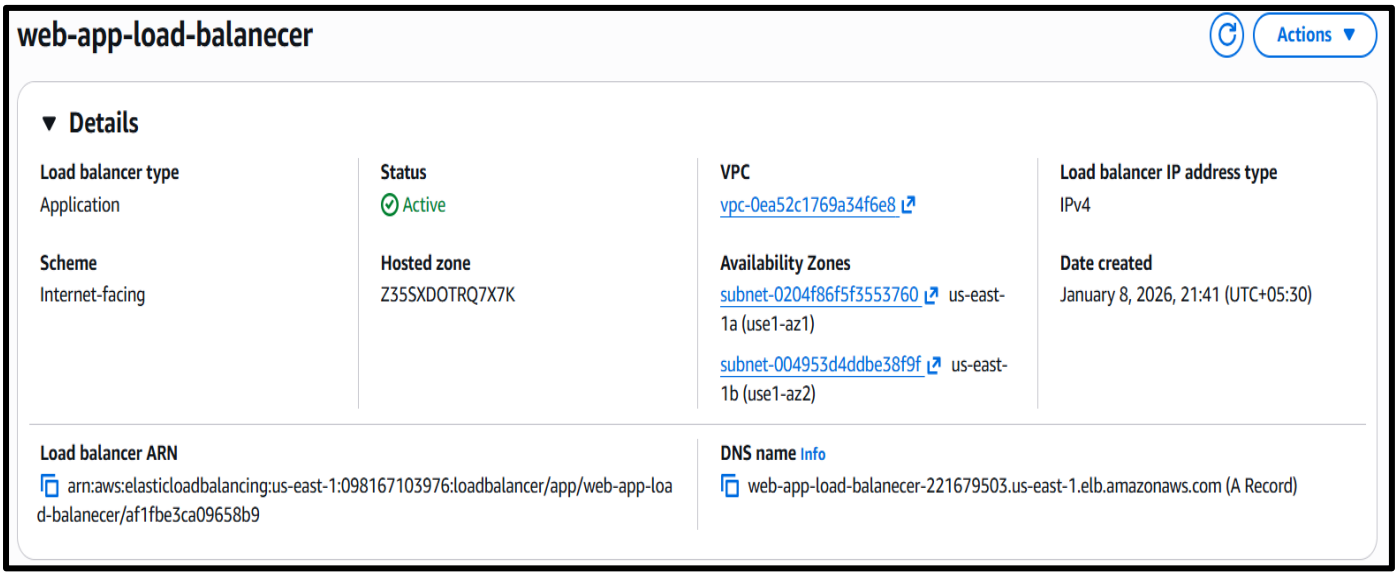
1. Amazon EC2 (Elastic Compute Cloud)

Amazon EC2 provides scalable virtual servers to host the web application. EC2 instances run the Apache web server and dynamically scale based on workload requirements. These instances are launched automatically using a predefined launch template.



2. Application Load Balancer (ALB)

The Application Load Balancer distributes incoming HTTP traffic across multiple EC2 instances. It improves application availability and fault tolerance by ensuring that traffic is routed only to healthy instances.



3. Auto Scaling Group (ASG)

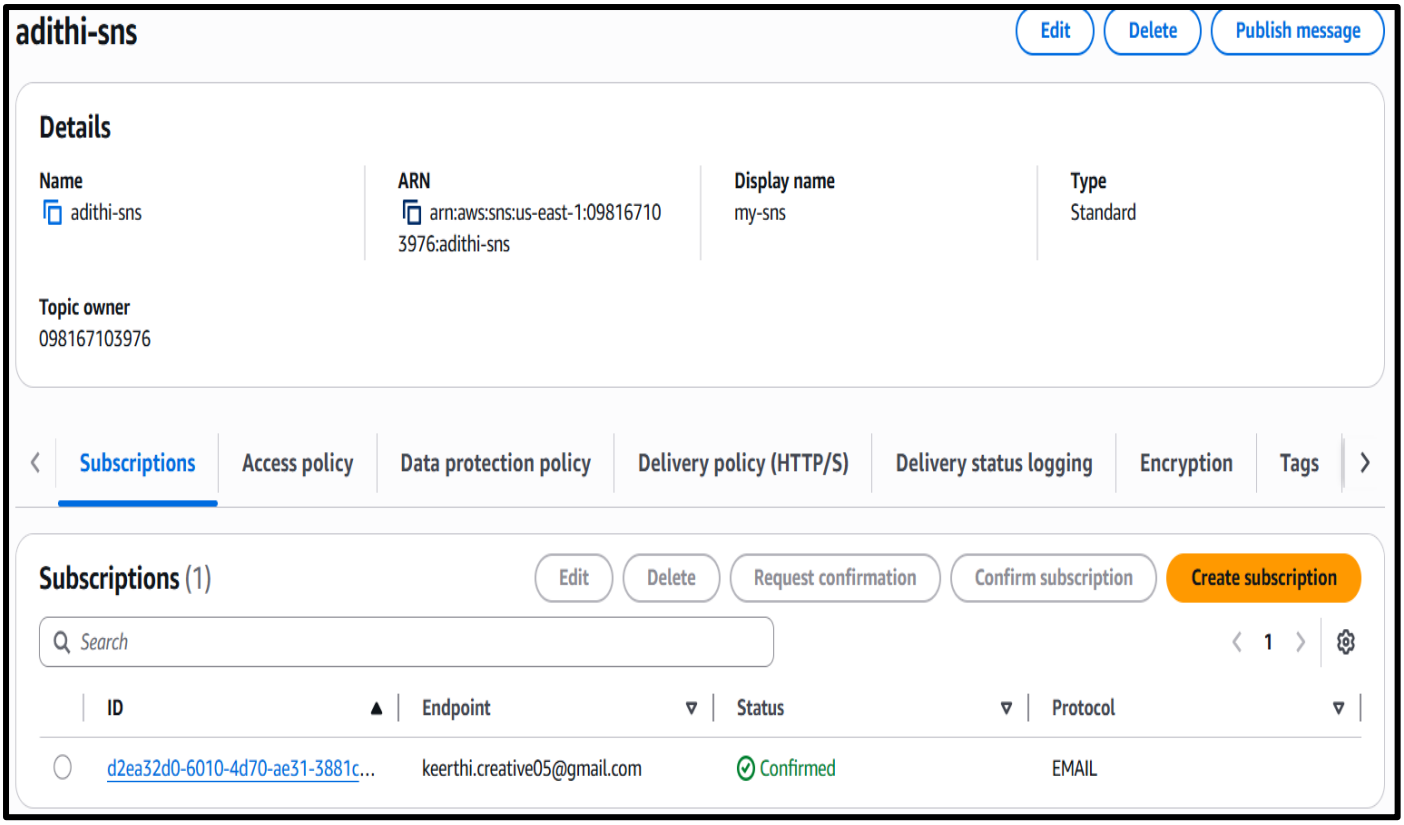
Auto Scaling Groups automatically manage the number of EC2 instances. Based on CPU utilization metrics, the ASG launches new instances during high demand and terminates instances when demand decreases, ensuring optimal resource utilization.

4. Amazon CloudWatch

Amazon CloudWatch monitors system performance metrics such as CPU utilization. CloudWatch alarms are configured to trigger scaling actions when predefined thresholds are crossed.

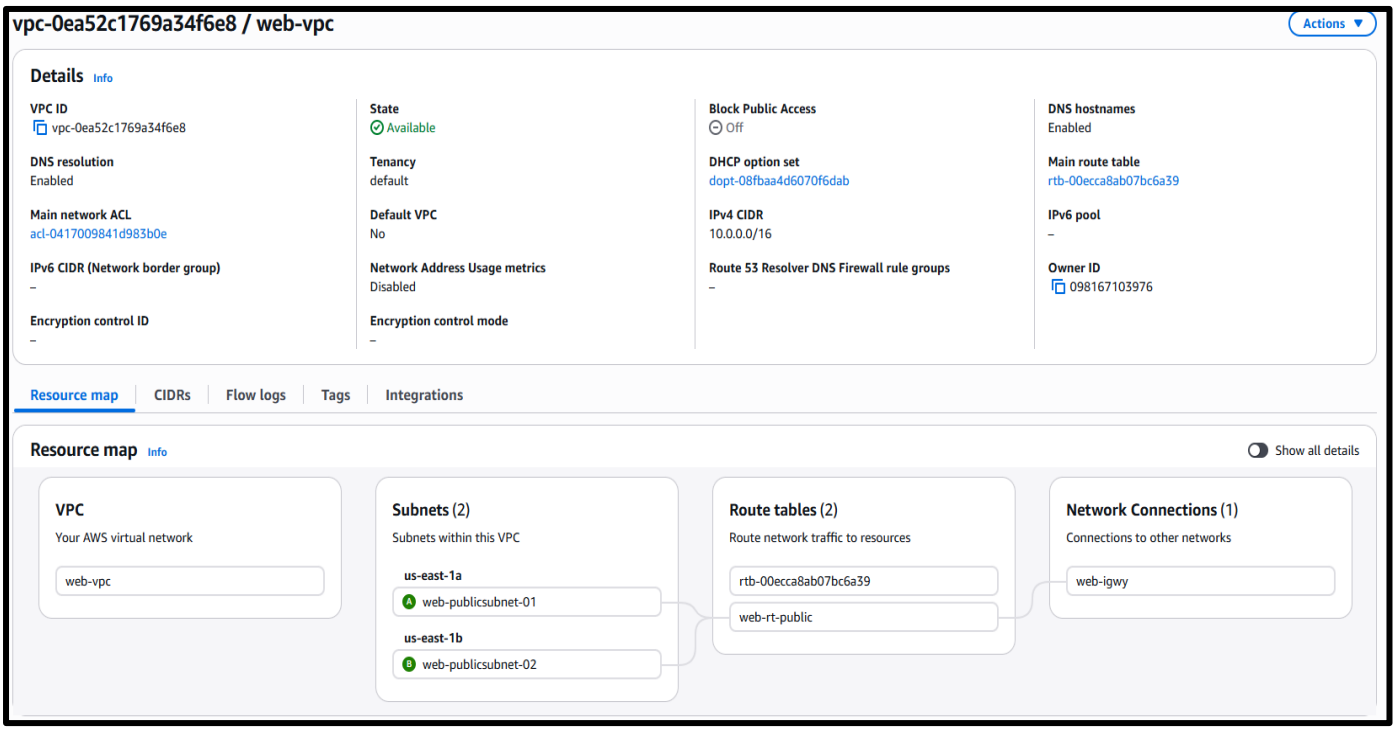
5. Amazon SNS (Simple Notification Service)

Amazon SNS is used to send real-time email notifications when CloudWatch alarms change state. This helps administrators stay informed about scaling events and system performance.



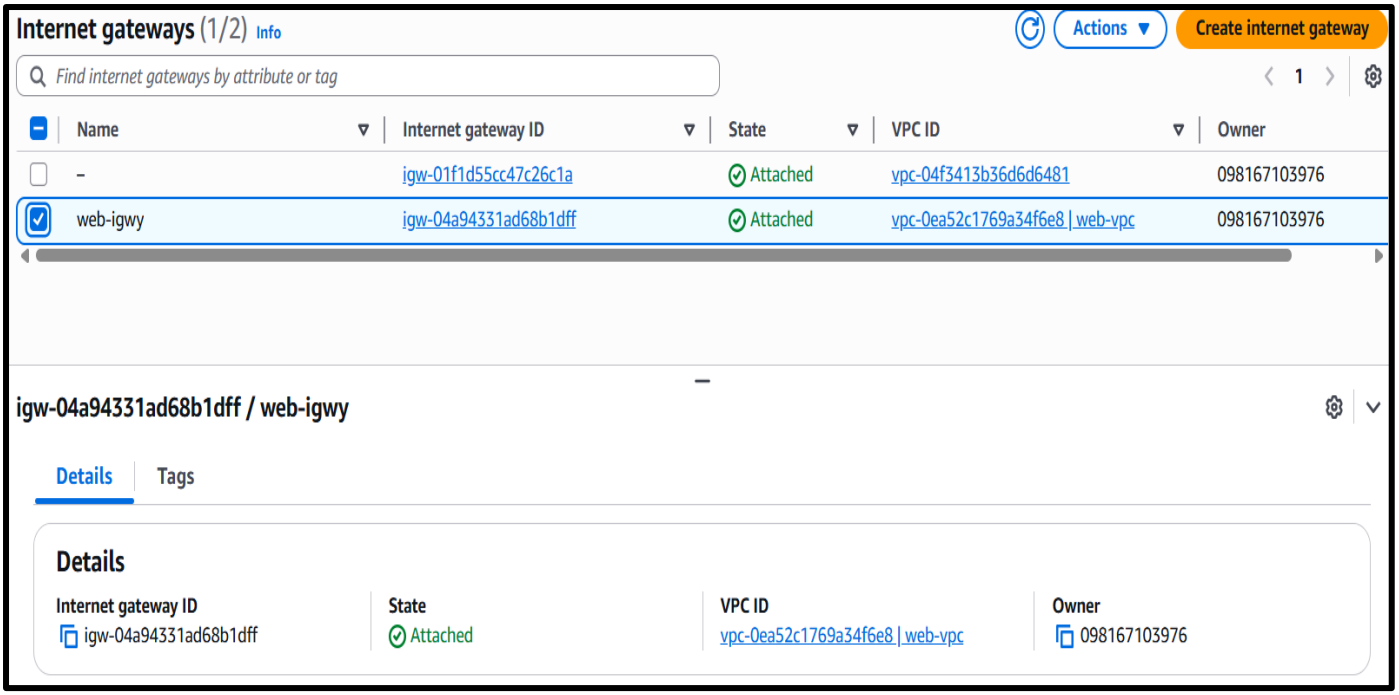
6. Amazon VPC (Virtual Private Cloud)

Amazon VPC provides the networking foundation for the project. It enables the creation of subnets, route tables, and internet gateways, allowing secure communication between AWS resources and external users.



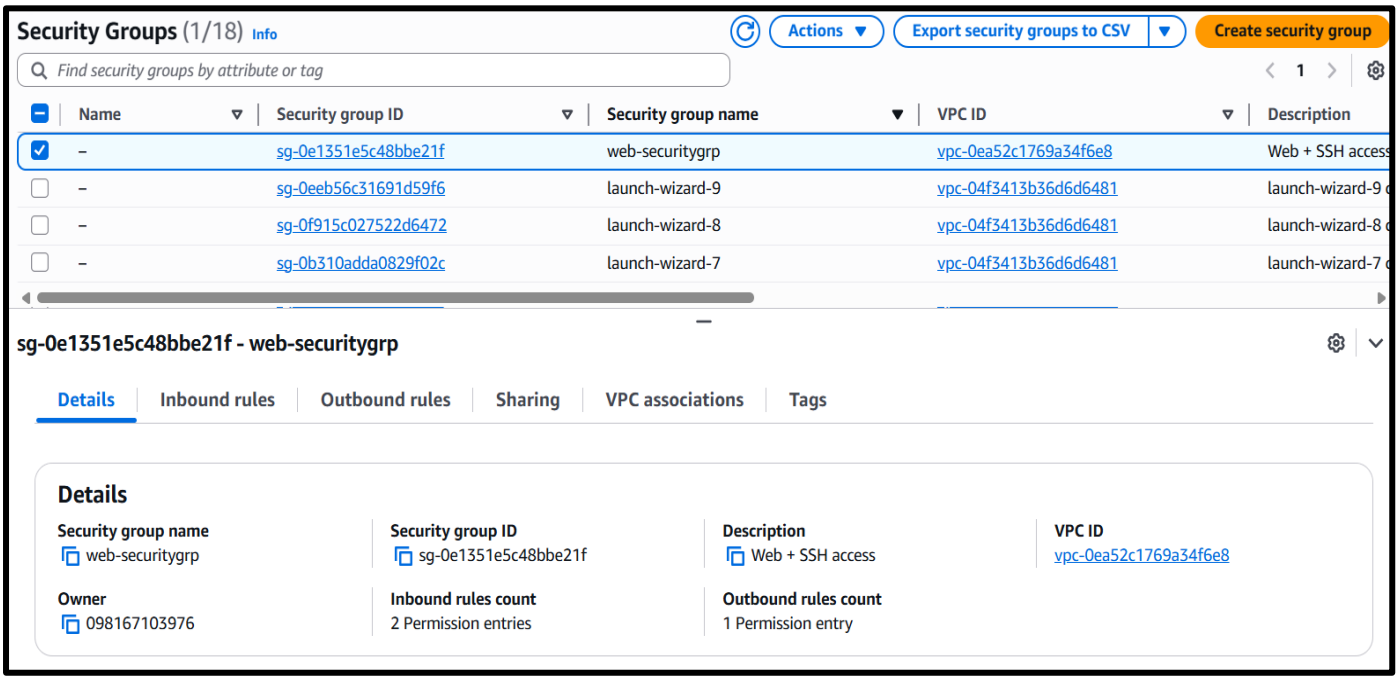
7. Internet Gateway

The Internet Gateway enables internet access for resources deployed within the VPC, allowing users to access the application through the Application Load Balancer.



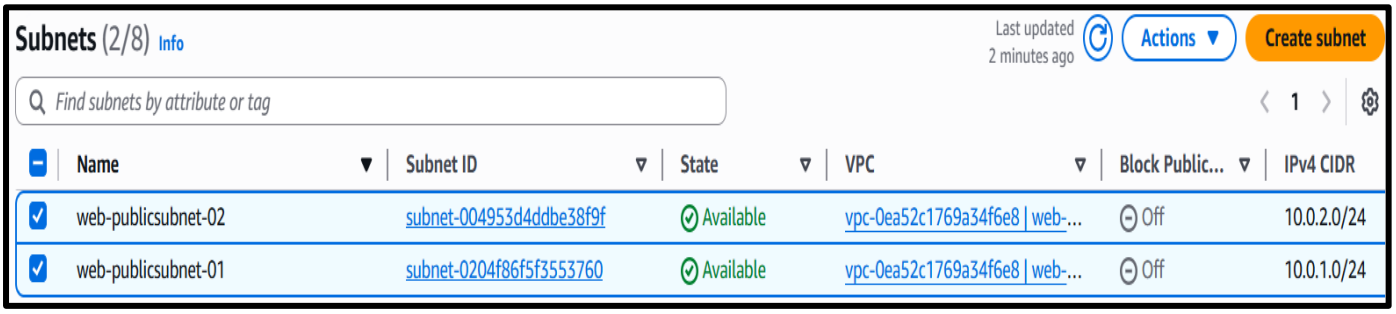
8. AWS Security Groups

Security Groups act as virtual firewalls that control inbound and outbound traffic to EC2 instances and the load balancer. They ensure that only authorized traffic is allowed.



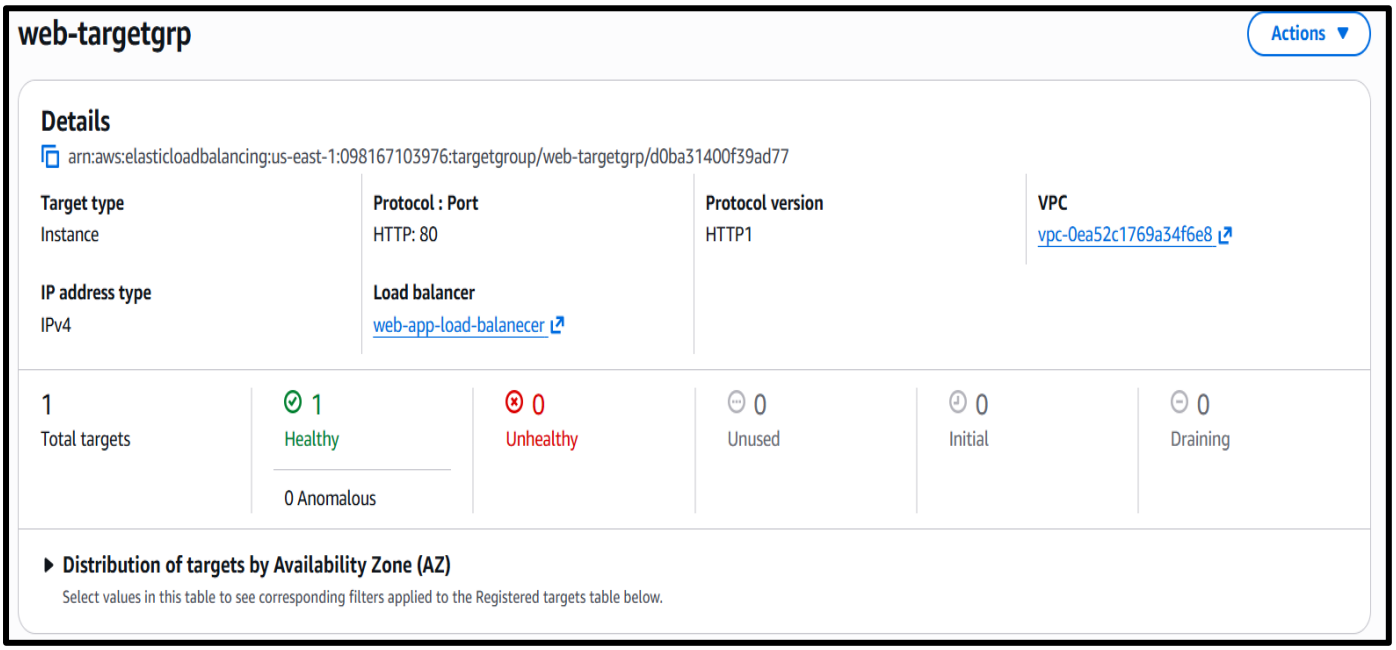
9. Subnets

Subnets are used to logically divide the VPC network into smaller segments. In this project, multiple public subnets are created across different Availability Zones to ensure high availability and fault tolerance. These subnets host the Application Load Balancer and EC2 instances managed by the Auto Scaling Group, allowing traffic to be distributed across zones in case of failures.



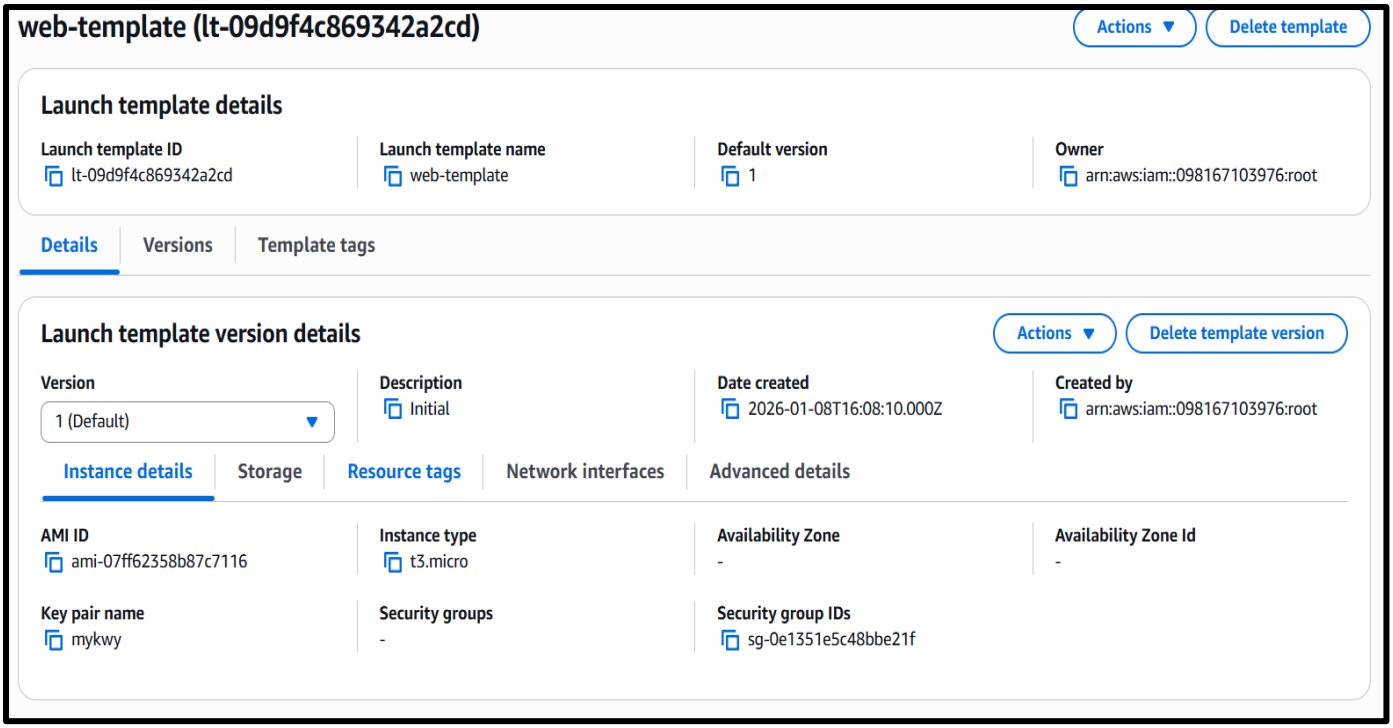
10. Target Groups

Target Groups are used by the Application Load Balancer to route incoming traffic to registered EC2 instances. The target group continuously performs health checks on the instances to ensure only healthy resources receive traffic. If an instance becomes unhealthy, it is automatically removed from traffic routing, improving reliability and application uptime.



11. Launch Template

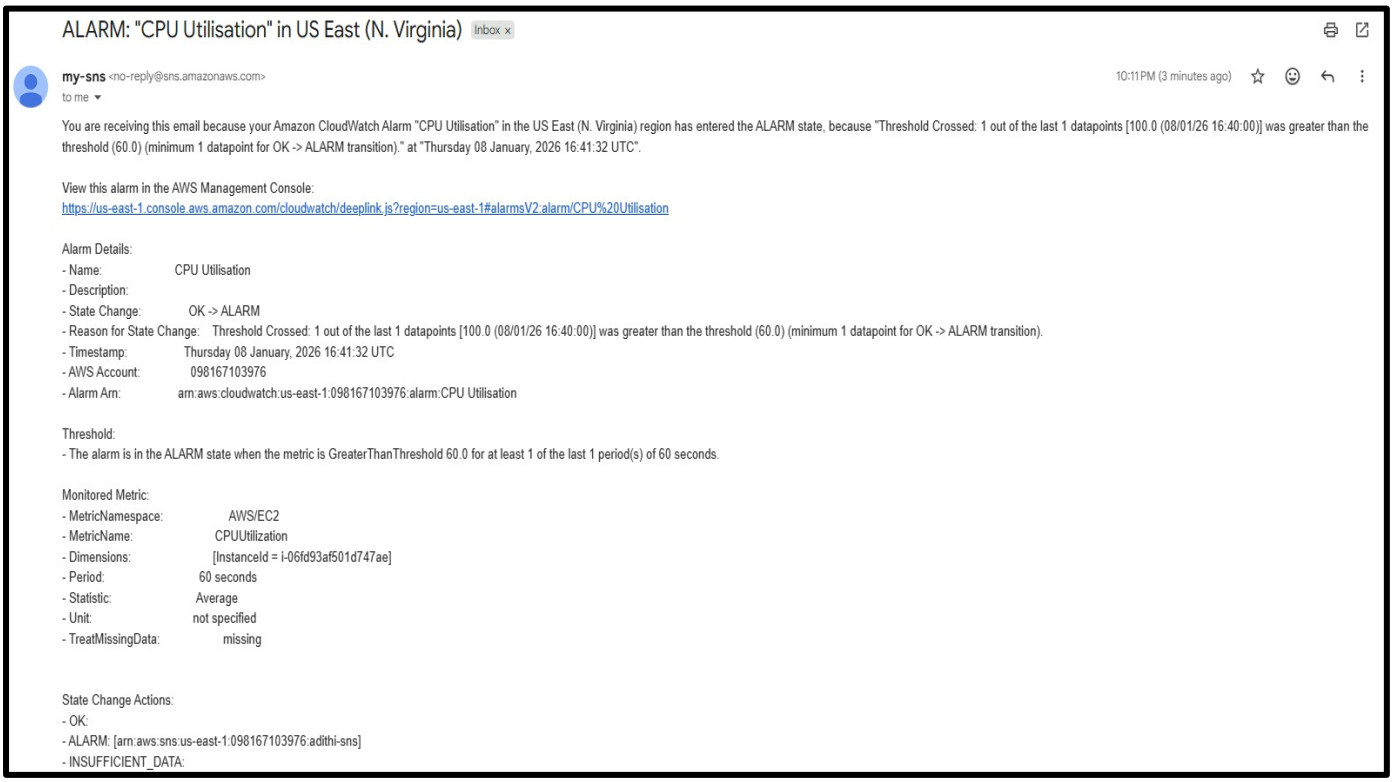
A Launch Template defines the configuration required to launch EC2 instances automatically. It includes the Amazon Machine Image (AMI), instance type, security group, key pair, and user data script for software installation. The Auto Scaling Group uses this launch template to create consistent and identical EC2 instances during scaling events, ensuring uniform application behaviour.



Security Design

Security is implemented using a layered approach. Security groups act as virtual firewalls, allowing only essential inbound traffic such as HTTP (port 80) for web access and SSH (port 22) for administrative purposes. All outbound traffic is permitted to allow system updates and service communication. Access control is minimized to reduce the attack surface, and resources are automatically managed to avoid prolonged exposure of unused instances.

STRESS TESTING AND VALIDATION



To test auto scaling:

- Connected to EC2 instance via SSH
- Executed stress command to increase CPU utilization
- Observed CPU crossing threshold
- CloudWatch alarm triggered
- Auto Scaling launched a new EC2 instance
- SNS email notification received

This confirms successful dynamic scaling.

Conclusion

The project successfully demonstrates the deployment of an auto-scaling web infrastructure in the cloud. By leveraging AWS services such as EC2, Application Load Balancer, Auto Scaling, CloudWatch, and SNS, the system efficiently adapts to changing workloads while maintaining high availability and cost efficiency. This implementation provides a strong foundation for understanding scalable cloud architectures and real-world infrastructure automation.

By -
R. Adithi
Aspiring DevOps Engineer
Batch - 06