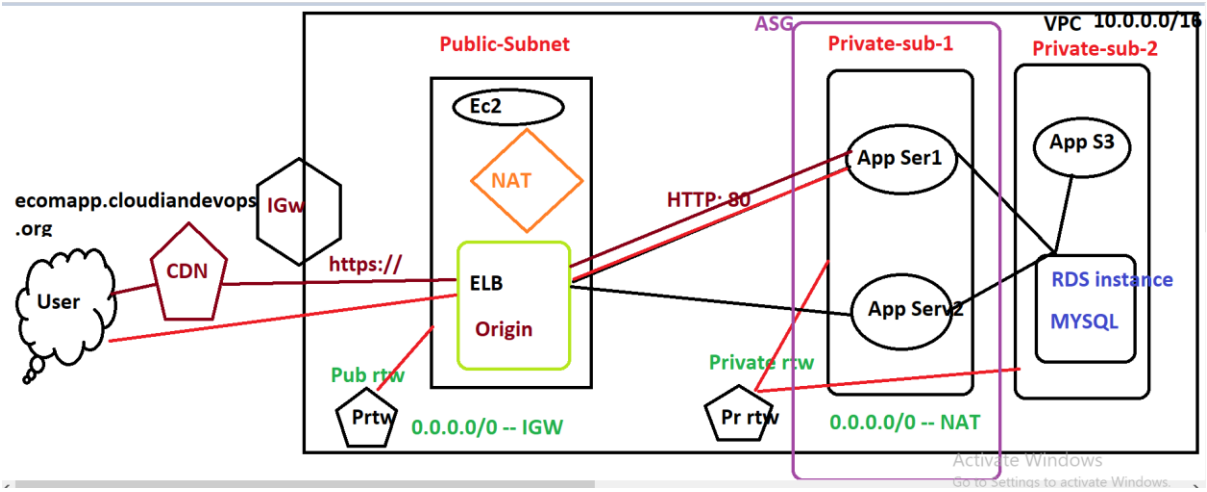
AWS Project

Technology stack: VPC, EC2, ELB, Auto Scaling, SNS, Route 53, AWS Cert Manager, RDS and CloudFront.

Project Overview:

This project involves designing and deploying a highly available, scalable, and secure web application on AWS. The architecture includes core AWS services such as Virtual Private Cloud (VPC) for network isolation, Elastic Compute Cloud (EC2) for hosting the application, Elastic Load Balancer (ELB) for distributing traffic, Auto Scaling for dynamic scalability, Route 53 for DNS management, AWS Certificate Manager (ACM) for secure communication, RDS and Amazon CloudFront for accelerated content delivery. This setup is optimized for performance, fault tolerance, and security, ensuring a reliable and seamless user experience.

Source code repo: https://github.com/AQtar-004/Ecomapp-stack.git



Architecture diagram

Services and Their Roles in the Project

1. Virtual Private Cloud (VPC)

- **Purpose**:
    - Provides a secure and isolated network environment for hosting resources.
    - Enables control over subnets, IP ranges, and routing tables.
- **Implementation**:
    - Create a custom VPC with public and private subnets.

    - Public subnets are for resources like ELB or NAT Gateways.
    - Private subnets host EC2 instances and the RDS database.
- **Key Benefits**:
    - Network isolation.
    - Secure access through private IPs.
    - Fine-grained control over traffic.

2. Elastic Compute Cloud (EC2)

- **Purpose**:
    - Provides virtual servers to host the application backend and frontend.
- **Implementation**:
    - Launch EC2 instances in private subnets to run the backend services.
    - Configure security groups to allow access only from the Load Balancer.
    - Use an Auto Scaling group to handle dynamic traffic spikes.
- **Key Benefits**:
    - Flexible compute capacity.
    - Integration with Auto Scaling for cost optimization.

3. Elastic Load Balancer (ELB)

- **Purpose**:
    - Distributes incoming traffic across multiple EC2 instances for high availability and fault tolerance.
- **Implementation**:
    - Deploy an Application Load Balancer (ALB) to distribute HTTP/HTTPS traffic.
    - Use target groups to route traffic to backend services.
- **Key Benefits**:
    - Redundancy and fault tolerance.
    - Secure SSL termination with AWS Certificate Manager.

4. Auto Scaling

- **Purpose**:
    - Automatically adjusts the number of EC2 instances based on traffic patterns.

- **Implementation**:
    - Define scaling policies to add or remove instances based on CPU utilization or request count.
    - Minimum and maximum instance limits ensure cost control.
- **Key Benefits**:
    - Ensures application availability during peak loads.
    - Optimizes costs by scaling down during low demand.

5. Simple Notification Service (SNS)

- **Purpose**:
    - Sends notifications and alerts to system administrators and users.
- **Implementation**:
    - Configure SNS to send alerts about Auto Scaling events or system failures.
    - Use email or SMS as notification channels.
- **Key Benefits**:
    - Real-time monitoring and alerting.
    - Integration with other AWS services like CloudWatch and Lambda.

6. Route 53

- **Purpose**:
    - Provides domain name resolution and traffic routing to the application.
- **Implementation**:
    - Create a hosted zone for the application domain.
    - Use alias records to point the domain to the ELB.
- **Key Benefits**:
    - High availability and low latency for DNS queries.
    - Seamless routing to backend resources.

7. AWS Certificate Manager (ACM)

- **Purpose**:
    - Simplifies the provisioning and management of SSL/TLS certificates for secure communication.
- **Implementation**:

    - Request a certificate for the domain.
    - Integrate it with the Application Load Balancer to enable HTTPS.
- **Key Benefits**:
    - Free SSL certificates.
    - Automated certificate renewal.

8. Relational Database Service (RDS)

- **Purpose**:
    - Hosts the application's database in a managed and highly available manner.
- **Implementation**:
    - Use Amazon Aurora or MySQL as the database engine.
    - Deploy in a Multi-AZ configuration for high availability.
    - Enable read replicas for performance optimization.
- **Key Benefits**:
    - Automated backups and patching.
    - Scalability and resilience for database workloads.

9. Amazon CloudFront

- **Purpose**:
    - Accelerates the delivery of static and dynamic web content, reducing latency for global users.
    - Acts as an additional layer of security with AWS Web Application Firewall (WAF) integration.
- **Implementation**:
    - Configure CloudFront as a content delivery network (CDN) for static assets such as images, CSS, and JavaScript.
    - Origin is set to the Application Load Balancer (ALB) or an S3 bucket storing static assets.
    - Enable caching and use geo-redundant edge locations to deliver content faster.
- **Key Benefits**:
    - Reduced latency with edge caching.
    - Enhanced security via AWS Shield and WAF.
    - Cost-effective global content delivery.

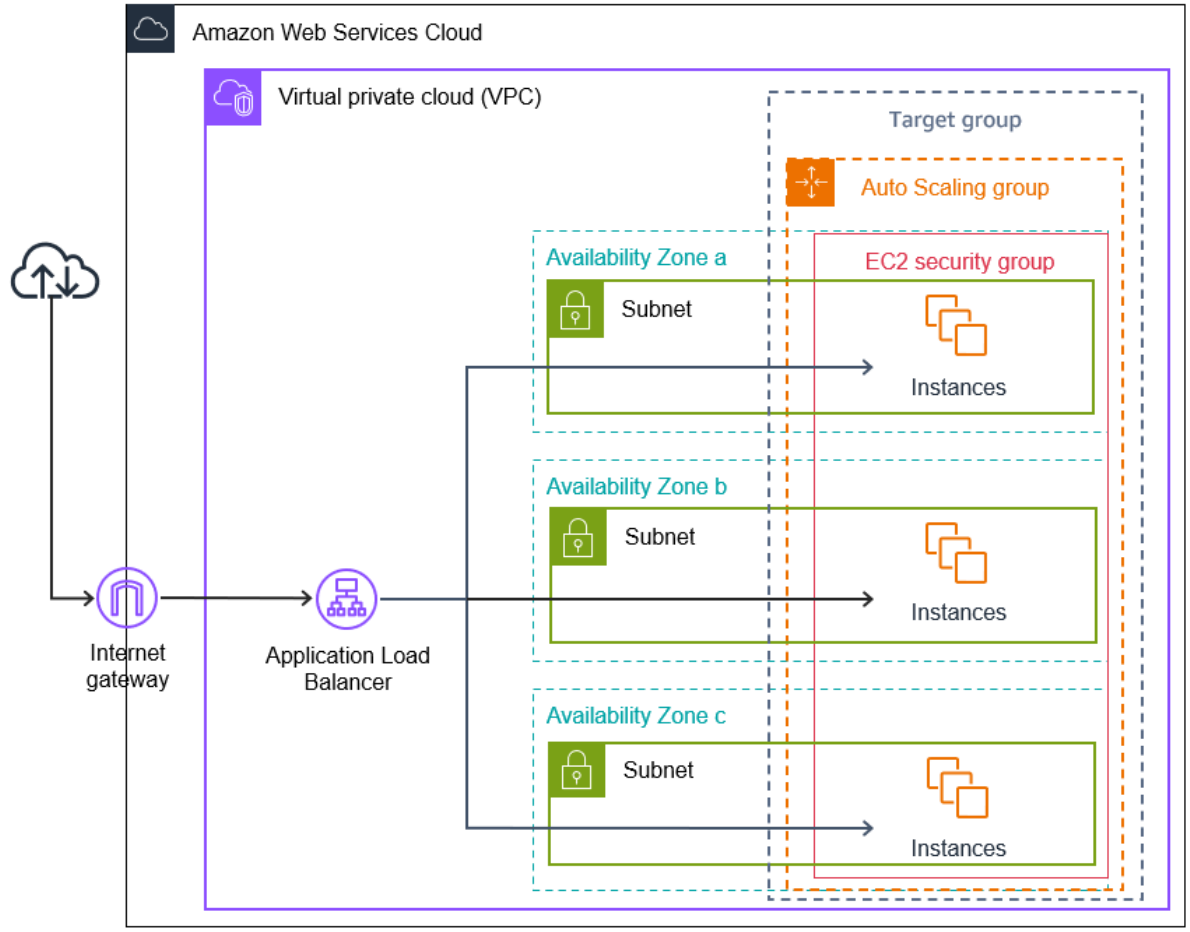
Solution Architecture Overview

1. Users access the e-commerce website through the domain registered in **Route 53**.
2. The request is routed to the **Elastic Load Balancer (ELB)**, which distributes it to EC2 instances.
3. EC2 instances run the application logic in the private subnet.
4. Backend services interact with the database hosted on **RDS**.
5. **Auto Scaling** ensures that the number of EC2 instances adjusts dynamically based on traffic.
6. **ACM** provides SSL certificates for secure communication via HTTPS.
7. **SNS** sends alerts and notifications for critical events.
8. **CloudFront** distributes the static content globally via edge locations, ensuring minimal latency for end users.

Key Advantages of This Setup

- **Scalability**: Auto Scaling and ELB ensure seamless scaling based on demand.
- **High Availability**: Multi-AZ RDS, ELB, and distributed resources minimize downtime.
- **Security**: Secure network design using VPC, SSL/TLS, and private subnets.
- **Automation**: AWS-managed services like RDS and ACM reduce operational overhead.
- **Optimized Costs**: Efficient caching with CloudFront reduces backend load.

Ref: https://docs.aws.amazon.com/autoscaling/ec2/userguide/tutorial-ec2-auto-scaling-load-balancer.html



How CloudFront Works?

Ref blog:
https://docs.aws.amazon.com/AmazonCloudFront/latest/DeveloperGuide/HowCloudFrontWorks.html