Name: Phenyo Modise

3D E-Commerce Platform Architecture on AWS

Scenario:

You are part of a startup team launching a next-generation 3D e-commerce web application. This platform will allow users to interact with 3D models of products before purchasing. Millions of users are expected globally, and the experience must be fast, highly available, secure, and cost-efficient. As a Cloud Practitioner, my role is to design the cloud architecture using AWS services to support this 3D application and meet key business and technical requirements.

Requirements :

1. High Availability

- The platform should be available 24/7 with built-in fault tolerance and failover mechanisms.

2. Scalability

- It should handle unpredictable spikes in traffic.

3. Performance

- Users should experience fast interactions with 3D content, quick page loads, and smooth product rendering.

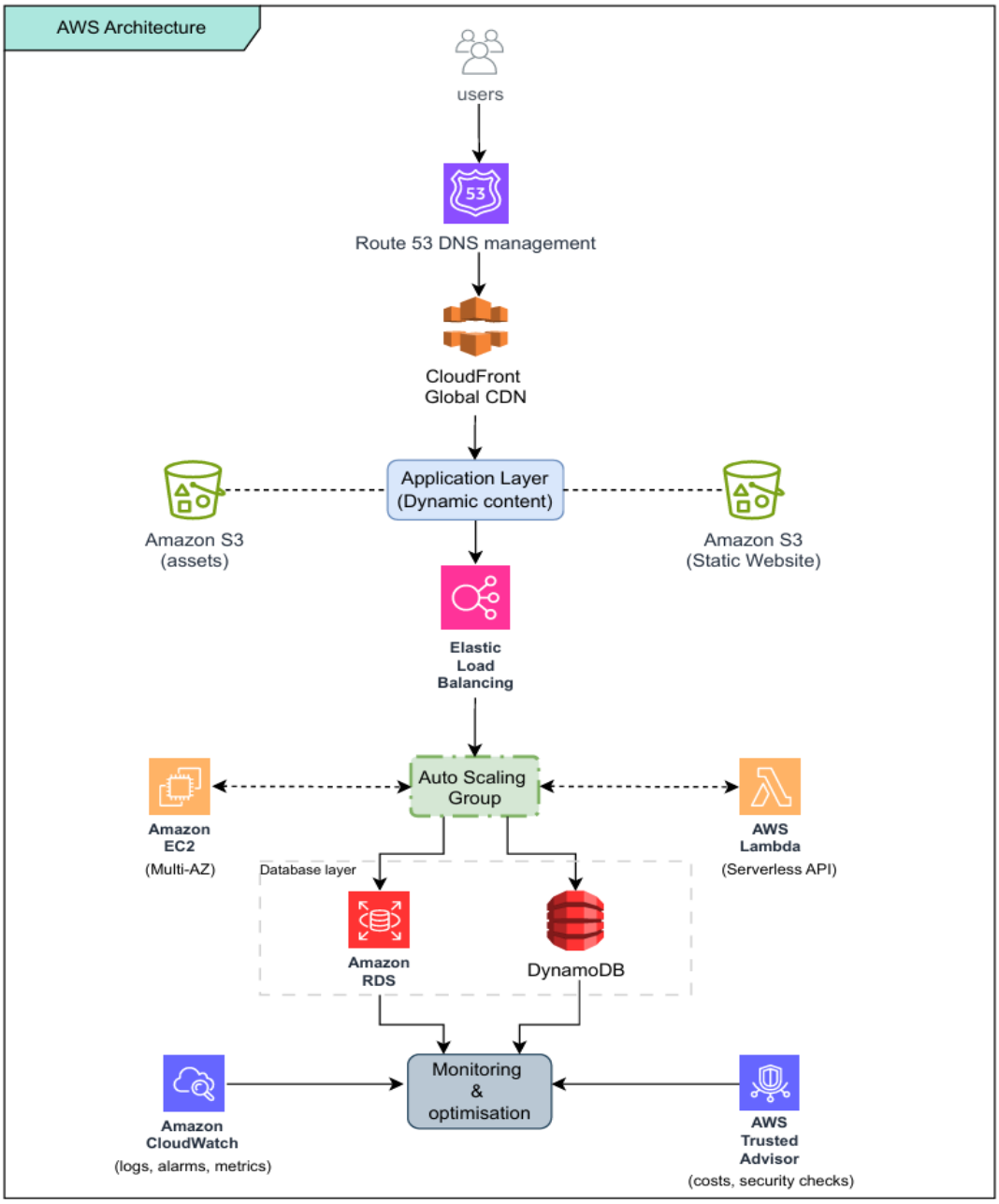
4. Security

- Follow AWS security best practices

5. Cost Optimization

- Avoid over-provisioning. Use auto-scaling, managed services, and monitoring for cost control.

Architecture Design



Why Each AWS Service Was Chosen

- Amazon Route 53: selected for DNS management and intelligent traffic routing. It directs users to the nearest CloudFront edge location and supports health checks and failover routing, improving availability and performance.

- Amazon CloudFront: chosen to deliver 3D assets, images, and static files through a global Content Delivery Network (CDN). Since the platform serves millions of global users interacting with large 3D files, CloudFront reduces latency and improves loading speed by caching content at edge locations.

- Amazon S3: stores 3D models, product images, and static website files. It was chosen because it offers high durability, scalability, cost-efficient storage, and seamless integration with CloudFront.

- Elastic Load Balancer (Application Load Balancer): Elastic Load Balancing distributes incoming traffic across multiple backend servers. This ensures no single server becomes overwhelmed and increases system availability.

- Amazon EC2: provides scalable virtual servers for running backend APIs and business logic. It was selected for flexibility and control over the application environment.

- AWS Lambda: can be used for serverless backend functions. It automatically scales and reduces operational overhead, making it cost-efficient for unpredictable workloads.

- Amazon RDS: stores structured data such as customer accounts, orders, and payments. Multi-AZ deployment ensures failover and high availability.

- Amazon DynamoDB: Amazon DynamoDB handles high-volume, low-latency data such as product catalogues and session information. It automatically scales and delivers millisecond performance.

- Amazon CloudWatch: Amazon CloudWatch monitors application performance, triggers alarms, and tracks system metrics to maintain reliability and cost control.

- AWS Trusted Advisor: AWS Trusted Advisor provides recommendations on cost optimization, security best practices, and performance improvements.

How the Architecture Meets the 5 Requirements

1. High Availability

- Multi-AZ deployment for EC2 and RDS
- Load balancing across instances
- Route 53 health checks and failover
- CloudFront global distribution

If one Availability Zone fails, traffic is redirected automatically, ensuring continuous service.

2. Scalability

- Auto Scaling Groups adjust EC2 capacity automatically
- Lambda scales automatically based on demand
- DynamoDB auto-scaling
- S3 provides unlimited storage

This allows the system to handle unpredictable traffic spikes, such as promotional sales or global campaigns**.**

3. Performance

- CloudFront caches content at edge locations
- DynamoDB provides millisecond response times
- Load balancing reduces backend strain
- S3 delivers high-throughput static content

Users experience fast page loads and smooth 3D rendering globally.

4. Security

- IAM roles enforce least privilege access
- Security Groups control network access

- Data encryption at rest (RDS, S3) and in transit (HTTPS)
- Trusted Advisor security checks
- CloudWatch monitoring for unusual activity

The architecture follows AWS security best practices and protects sensitive customer data.

5. Cost Optimization

- Auto Scaling prevents over-provisioning
- Lambda uses pay-per-execution pricing
- S3 lifecycle policies reduce storage costs
- CloudWatch monitors usage trends
- Trusted Advisor identifies underutilized resources

This ensures the startup only pays for what it uses.

Trade-Offs and Challenges

1. EC2 vs Lambda

- EC2 provides more customization and control but requires management.
- Lambda reduces management overhead but may face cold start latency.

Trade-off: Lambda is more cost-efficient for unpredictable traffic, while EC2 is better for complex backend processing.

2. RDS vs DynamoDB

- RDS supports complex transactions and relational integrity.
- DynamoDB offers extreme scalability and performance but is schema-less.

Trade-off: Using both increases architectural complexity but optimizes performance and scalability.

3. 3D Content Delivery Challenges

- Large 3D files can increase bandwidth usage and costs.
- Requires optimization (compression, caching).
- High global traffic may increase CloudFront data transfer costs.