0.Architecture Summary

📘 Architecture Summary

Source:

💻 Overall Architecture Overview

A highly available and scalable **three-tier web application architecture** built on AWS.

🌐 1. Network / Load Balancing Layer

Category	Requirements	Configuration	Implemented Tasks
VPC / Subnets	Ensure high availability and network segmentation.	Two **Public Subnets** (for ALB) and two **Private Subnets** (for EC2/DB), distributed across multiple AZs.	Created VPC, subnets, Internet Gateway, NAT Gateway, and route tables.
Load Balancing	Distribute traffic across EC2 instances and ensure redundancy.	**Application Load Balancer (ALB)** deployed in the Public Subnets.	Created ALB, created Target Group, registered EC2 instances, configured HTTPS listener and HTTP→HTTPS redirect.
DNS / SSL	Domain-based access and HTTPS termination.	Managed domain via **Route 53**, issued SSL certificate via **ACM**.	Registered domain, created Hosted Zone, issued ACM certificate, created ALB A-record.
Monitoring	Monitor service health and availability.	CloudWatch monitoring for ALB metrics.	Configured alarms for **HealthyHostCount**, **5XXError**, **ConnectionErrorCount**.

💻 2. Application / Web Layer

Category	Requirements	Configuration	Implemented Tasks
Web / AP Servers	Provide runtime environment for the web application and ensure redundancy.	Two **EC2 instances (Amazon Linux 2023)** placed in Private Subnets.	Provisioned EC2, installed essential packages (git, PHP-FPM, etc.).
Middleware	Provide runtime for the PHP (Laravel) application.	**Nginx** (web server) and **PHP-FPM** (PHP runtime) integrated.	Configured Nginx reverse proxy, PHP-FPM settings, and verified integration ("Hello GEMY!").
Logging / Monitoring	Detect application anomalies and support troubleshooting.	CloudWatch monitoring for EC2 metrics & ALB access logs stored in S3.	Configured CPUUtilization (>80%) alarm, created S3 bucket for ALB logs, set lifecycle rules.

💾 3. Database Layer

Category	Requirements	Configuration	Implemented Tasks
DB Service	Ensure high availability and reduce operational overhead with a managed service.	**RDS MySQL Multi-AZ**, placed in Private Subnets.	Created RDS instance, configured Parameter Group, installed DB client (mariadb105), verified DB connectivity.
Monitoring	Detect performance bottlenecks and connection saturation.	CloudWatch monitoring for RDS metrics.	Set alarms on **DatabaseConnections**, **CPUUtilization** (per DBInstanceIdentifier).