
Project Title: AWS 3-Tier Web Application Architecture

Technologies Used: AWS (VPC, EC2, RDS, ELB, Auto Scaling, SNS, CloudWatch), HTTPD, MySQL, WordPress, IAM, Security Groups, etc.

Project Overview:

Designed, deployed, and managed a 3-tier web application architecture on AWS, ensuring high availability, scalability, and security for a WordPress-based application.

Key Responsibilities:

- **VPC Design and Setup**:
  - Created a custom VPC (10.0.0.0/16) with multiple subnets across two availability zones (AZ-a and AZ-b) to ensure high availability and fault tolerance.
  - Configured **Public Subnets** for load balancers and NAT gateways, and **Private Subnets** for application and database servers to isolate critical workloads from the internet.
- **Subnet Configuration**:
  - Configured Public and Private subnets, along with DB-specific subnets to ensure proper routing and segregation of resources.
  - Subnet Design:
    - Public-Subnet-1 (10.0.1.0/24) and Public-Subnet-2 (10.0.2.0/24) for load balancers and NAT gateway.
    - Private-Subnet-1 (10.0.3.0/24) and Private-Subnet-2 (10.0.4.0/24) for application servers.
    - DB Subnets (10.0.5.0/24 & 10.0.6.0/24) across two AZs for database instances.
- **Routing and Connectivity**:
  - Configured **Internet Gateway** (IGW) and **NAT Gateway** for enabling internet connectivity for public and private subnets.

    - Set up custom route tables (Public and Private RTs) to enable communication between the subnets and ensure the proper traffic flow.
- **Security Configuration**:
    - Created and configured **Security Groups** (SG) for Load Balancer, Application Servers, Databases, and Bastion Hosts to manage access control based on specific rules:
        - Load Balancer-SG: Allowed HTTP & HTTPS traffic from any source.
        - App-Server-SG: Allowed HTTP traffic only from the Load Balancer.
        - DB-SG: Restricted database access only from the App Servers.
        - Bastion-SG: Enabled SSH access from trusted IPs for management purposes.
- **IAM Role and EC2 Instance Setup**:
    - Created an **IAM Role (EC2SSMAgent)** with the necessary permissions (**AmazonSSMManagedInstanceCore**) to enable EC2 instances to interact with AWS Systems Manager.
    - Launched EC2 instances in the **Private Subnets**, configured with appropriate IAM roles and application security groups.
    - Installed and configured necessary software (Apache HTTPD, MariaDB, PHP) for hosting the WordPress application on the EC2 instances.
- **WordPress Application Setup**:
    - Installed and configured WordPress on EC2 instances, including setting up the database and adjusting the wp-config.php file to connect to the RDS instance.
    - Configured MariaDB and WordPress, creating a dedicated **WordPress database** and configuring necessary parameters (database name, user credentials, and endpoint) for connection.
- **RDS Database Configuration**:
    - Set up a **multi-AZ RDS instance** for MySQL, configured with backup and replication for high availability and durability.
    - Created a **DB Subnet Group** across two AZs and ensured the proper setup of security groups and network access control.
- **Load Balancer and Auto Scaling**:

    - Configured an **Application Load Balancer (ALB)** to distribute incoming traffic across multiple EC2 instances for better load balancing and fault tolerance.
    - Created a **Target Group** and associated it with the Application Server instances.
    - Set up **Auto Scaling** policies to ensure the application scales dynamically based on the traffic load.
- **Monitoring and Notifications**:
    - Integrated **Amazon CloudWatch** for monitoring the health and performance of EC2 instances, RDS databases, and load balancers.
    - Configured **SNS (Simple Notification Service)** for automatic notifications regarding instance health, load balancer status, and other critical system events.
- **AMI Creation and Application Deployment**:
    - Created a custom **AMI** of the configured Application Server to enable rapid scaling and future deployments of identical environments.
    - Deployed the WordPress application by creating an AMI and associating it with the **Auto Scaling Group** to maintain consistent performance during scaling events.
- **Testing and Validation**:
    - Validated the system by checking the **Load Balancer DNS** to ensure the application was accessible and functioning properly.
    - Conducted load and performance testing to ensure scalability and high availability of the application architecture.

Achievements:

- Successfully deployed a highly available, fault-tolerant 3-tier architecture using AWS services.
- Ensured security and compliance by implementing proper access controls, IAM roles, and security groups.
- Optimized the application for scalability with **Auto Scaling** and **Elastic Load Balancing**, handling fluctuations in traffic seamlessly.

- Enabled seamless application monitoring and alerting using **CloudWatch** and **SNS** for proactive issue resolution.
