DevOps

[3 Tier Architecture 2](#_Toc194043827)

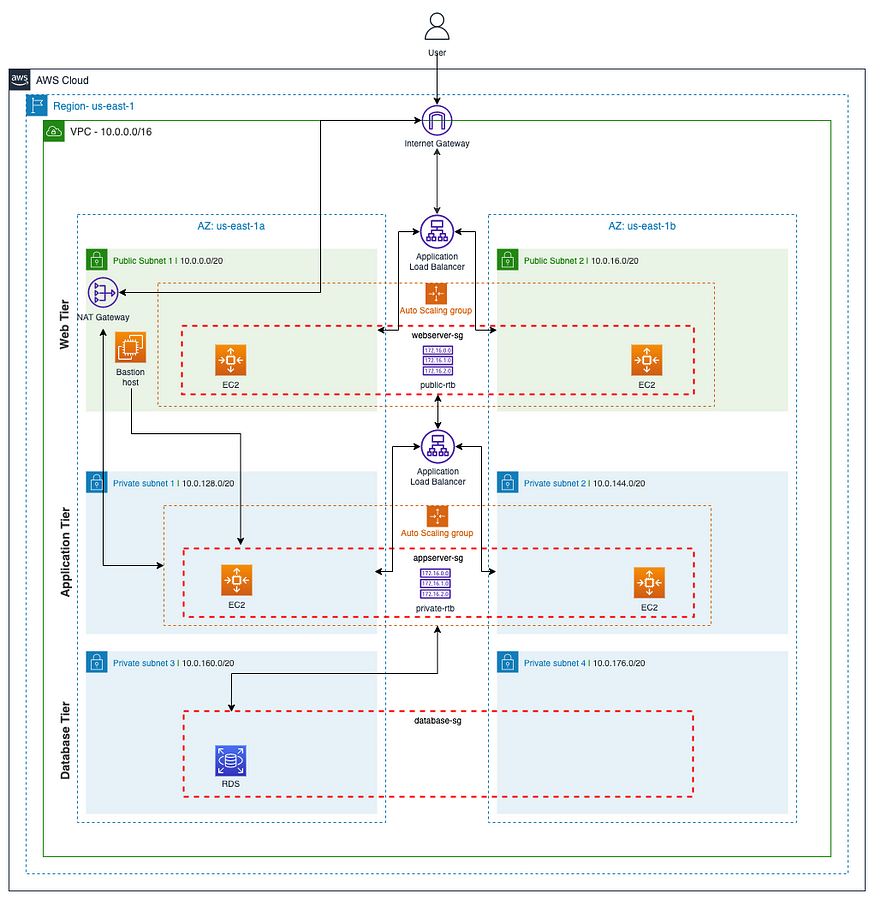
[Overview 2](#_Toc194043828)

[3-Tier Architecture Application Deployment 5](#_Toc194043829)

# 3 Tier Architecture

## Overview

[Building a 3-tier web application architecture with AWS | by Aalok Trivedi | Medium](https://medium.com/@aaloktrivedi/building-a-3-tier-web-application-architecture-with-aws-eb5981613e30)



This architecture diagram represents a highly available, scalable, and secure 3-tier application deployment on AWS, organized into three layers: **Web Tier**, **Application Tier**, and **Database Tier**. The architecture leverages AWS services for networking, load balancing, security, and auto-scaling.

**1. AWS Region and VPC Configuration**

* **Region:** us-east-1
* **VPC CIDR:** 10.0.0.0/16
* The architecture spans two **Availability Zones (AZs)** (us-east-1a and us-east-1b) for high availability.

**2. Network and Security Components**

1. **Internet Gateway:**
   * Connects the VPC to the internet, allowing public access to the web tier.
2. **NAT Gateway:**
   * Enables instances in the private subnets to connect to the internet without exposing themselves directly.
3. **Security Groups (SG):**
   * **Web Server SG (webserver-sg):** Allows HTTP (80), HTTPS (443), and SSH (22) access.
   * **App Server SG (appserver-sg):** Allows traffic from the Web Server SG.
   * **Database SG (database-sg):** Restricts access to the App Server SG only.
   * **Public and Private NACLs:** Implement subnet-level traffic control for security.

**3. Web Tier**

1. **Public Subnets (2)**
   * **Subnet 1:** 10.0.0.0/20 (AZ: us-east-1a)
   * **Subnet 2:** 10.0.16.0/20 (AZ: us-east-1b)
2. **Auto Scaling Group (ASG)**
   * Manages a fleet of EC2 instances for the web application.
   * Distributes traffic through an **Application Load Balancer (ALB)**.
3. **Load Balancer (External)**
   * Accepts incoming traffic from the internet and forwards it to the web servers.
4. **Web Servers (EC2 Instances)**
   * Host NGINX or Apache web servers to serve content.

**4. Application Tier**

1. **Private Subnets (2)**
   * **Subnet 1:** 10.0.128.0/20 (AZ: us-east-1a)
   * **Subnet 2:** 10.0.144.0/20 (AZ: us-east-1b)
2. **Application Load Balancer (Internal)**
   * Balances traffic between multiple application servers.
3. **Auto Scaling Group (ASG)**
   * Dynamically scales the number of application servers based on load and performance metrics.
4. **Application Servers (EC2 Instances)**
   * Host the core business logic and application code.

**5. Database Tier**

1. **Private Subnets (2)**
   * **Subnet 3:** 10.0.160.0/20 (AZ: us-east-1a)
   * **Subnet 4:** 10.0.176.0/20 (AZ: us-east-1b)
2. **RDS MySQL Database**
   * Managed database with multi-AZ deployment for high availability.
   * Restricted to access from the application tier only.

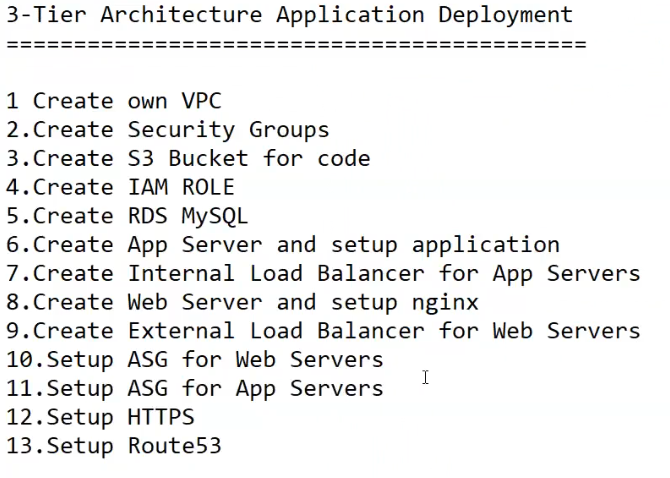
**6. Additional Components**

1. **Bastion Host**
   * Provides secure SSH access to instances in private subnets via the NAT Gateway.
2. **Route 53 (Not Shown)**
   * Provides DNS resolution and domain name management for the web servers.
3. **HTTPS Configuration**
   * Uses **AWS Certificate Manager (ACM)** to provide SSL certificates.
   * Secure communication via HTTPS through the external ALB.

**Benefits of This Architecture:**

1. **High Availability:** Multi-AZ deployment and load balancers ensure uptime.
2. **Scalability:** Auto Scaling Groups at both web and application tiers dynamically manage instance counts.
3. **Security:**
   * Multi-layer security with VPC, subnets, and security groups.
   * Isolated private subnets for application and database layers.
4. **Performance Optimization:**
   * Separate load balancers for web and application layers.
   * NAT Gateway to enable internet access from private subnets.
5. **Cost Optimization:**
   * Use of Auto Scaling reduces costs during low traffic.
   * Efficient use of NAT Gateway and private subnets.

## 3-Tier Architecture Application Deployment



The 3-tier architecture is a well-established model for application deployment that separates the application into three logical and physical tiers: Presentation (Web), Application (App), and Data (Database). Below is a step-by-step guide to deploying a 3-tier architecture using AWS services.

**Step-by-Step Deployment:**

1. **Create Own VPC (Virtual Private Cloud)**
   * Set up a dedicated virtual network for your infrastructure.
   * Define IP ranges, subnets, and network gateways.
2. **Create Security Groups**
   * Set up inbound and outbound rules to control traffic.
   * Different security groups for Web, App, and Database tiers to ensure isolation and security.
3. **Create S3 Bucket for Code**
   * Store application code and configuration files.
   * Enable versioning and configure appropriate IAM policies for access.
4. **Create IAM Role**
   * Define roles with permissions to access S3, EC2, RDS, and other AWS services.
   * Attach the role to instances and services as needed.
5. **Create RDS MySQL**
   * Set up an RDS instance with MySQL as the database engine.
   * Configure multi-AZ deployment for high availability.
   * Enable automatic backups and encryption.
6. **Create App Server and Setup Application**
   * Launch EC2 instances to serve as application servers.
   * Deploy the application code and necessary dependencies.
   * Configure environment variables and settings.
7. **Create Internal Load Balancer for App Servers**
   * Distribute traffic among multiple application servers.
   * Ensure that only internal network traffic is handled by this load balancer.
8. **Create Web Server and Setup Nginx**
   * Set up EC2 instances to serve as web servers.
   * Install and configure **Nginx** for serving web content and reverse proxying to app servers.
9. **Create External Load Balancer for Web Servers**
   * Expose the web tier to the internet using an Application Load Balancer (ALB).
   * Route traffic to web servers and manage SSL termination.
10. **Setup ASG (Auto Scaling Group) for Web Servers**
    * Automatically scale the number of web servers based on traffic and resource usage.
    * Define scaling policies and health checks.
11. **Setup ASG (Auto Scaling Group) for App Servers**
    * Similar to web servers, enable auto-scaling for application servers.
    * Configure policies to ensure high availability and performance.
12. **Setup HTTPS**
    * Implement SSL/TLS for secure communication.
    * Use ACM (AWS Certificate Manager) to manage certificates.
13. **Setup Route 53**
    * Configure DNS to point to the external load balancer.
    * Use domain names for seamless access and routing.

**Benefits of 3-Tier Architecture:**

* **Scalability:** Each tier can scale independently.
* **Security:** Different security groups isolate each layer.
* **Availability:** Auto-scaling and load balancing ensure high availability.
* **Maintainability:** Logical separation makes troubleshooting easier.