AWS 3-Tier Web Application

Project Name: NatureNest

Project Description:

NatureNest is an AWS 3-tier web application aimed at delivering a smooth online experience for discovering and purchasing nature-based products. The application architecture is divided into three main layers: the Web Tier (Presentation Layer), Application Tier (Logic Layer), and Database Tier (Data Layer). The objective is to build a scalable, secure, and resilient platform leveraging AWS services.

Objectives:

1. Set up a VPC, which will connect the Web, Application, and Database Tiers. Create two public subnets for the Web Tier and two private subnets each for the Application and Database Tiers.
2. Design the Web Tier to handle incoming requests and deliver static content to users using an Auto Scaling Group, with public subnets allowing traffic from the internet to the EC2 instances.
3. Build the Application Tier to process business logic like transactions, payment processing, and database interactions.
4. Establish the Database Tier to store customer details, product data, and transaction logs using a free-tier MySQL RDS instance.
5. Conduct connectivity tests by accessing the Web Tier via its public IPv4 address, then SSH into the Web Tier EC2 to ping the Application Tier, and finally test connectivity from the Application Tier to the Database Tier.

Part 1: VPC, Subnets, and NAT Gateway Configuration

1. Go to **VPC** > **Create VPC** > Select **VPC and more options**. Use the IPv4 CIDR block 10.0.0.0/16 and disable the IPv6 CIDR block. Keep **Tenancy** as **Default**.
2. Choose **2** Availability Zones: select ap-south-1a for the first and ap-south-1b for the second.
3. Enter the CIDR blocks for both public and private subnets.
4. Select **1 NAT Gateway per AZ** and choose **None** for VPC endpoints. Enable both **DNS Hostnames** and **DNS Resolution**, then click **Create VPC**.
5. After creation, click **View VPC** to confirm we now have a VPC, 2 public subnets, 4 private subnets, and 2 NAT gateways.
6. Go to **VPC** > **Subnets** > Select each public subnet, click **Actions**, and edit the settings to enable auto-assignment of public IPv4 addresses.
7. Ensure the **Internet Gateway** is attached to the VPC.
8. Confirm both **NAT Gateways** are active.
9. Navigate to the **Route Tables** and confirm that public subnets are associated with public route tables and private subnets with private route tables.

Part 2: Web Tier Setup

1. Go to **EC2** then **Launch Templates** then Click **Create Launch Template**. Name the template and enable the **Auto Scaling guidance** option.
2. Choose **Amazon Linux 2 AMI** and the **t2.micro** instance type.
3. Use an existing key pair for SSH access.
4. Create a new security group, assign it to the VPC, and add rules for **ICMP**, **HTTP**, and **SSH**.
5. Enable auto-assigning of public IPs under advanced network settings.
6. Add the following script to the **User Data** section:

   ```
   #!/bin/bash
   yum update -y
   yum install -y httpd
   systemctl start httpd
   systemctl enable httpd
   echo "<center><h1> AWS 3-Tier Web Application - NatureNest</h1></center>" >
   /var/www/html/index.html
   ```

7. Create the template and then move to **Auto Scaling Groups**. Set up an Auto Scaling group with the template, select the public subnets, and proceed.
8. Attach an **Application Load Balancer**, selecting the **Internet-facing** scheme.
9. Confirm the **Target Group** is created, and health checks are enabled. Proceed with default settings.
10. Set the desired capacity for the Auto Scaling group and finalize the setup.
11. Validate the creation of the Load Balancer and ensure two EC2 instances are running in the Web Tier.

Part 3: Application Tier Setup

1. Create a new launch template for the Application Tier, enabling **Auto Scaling guidance**.
2. Choose **Amazon Linux 2 AMI** and **t2.micro** instance type.
3. Reuse the existing key pair.
4. Create a security group, assigning the VPC, and add rules for **ICMP** and **SSH** (with Web Tier security group as the source).
5. Create the template and set up an **Auto Scaling Group** using the private subnets for the Application Tier.
6. Attach an **Internal Application Load Balancer**.
7. Let AWS automatically create a **Target Group** and keep default settings for health checks.
8. Complete the Auto Scaling setup and confirm the desired capacity and scaling policies.

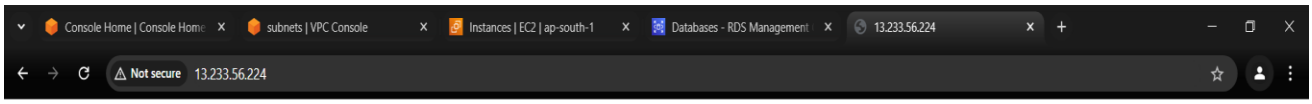
Part 4: Database Tier Setup

1. Go to **RDS** > **Subnet Groups** and create a new DB subnet group, selecting the VPC and the private subnets used for the Database Tier.
2. Under **RDS** > **Create Database**, choose the **MySQL Engine** and **Free-tier Template**.
3. Enter a DB instance name and master credentials, selecting **db.t2.micro** for instance type.

4. Do not connect the DB to an EC2 resource; instead, select the VPC and DB subnet group created earlier.
5. Set **Public access** to **No** to restrict access to the DB from the internet.
6. Create a new VPC security group and set inbound rules to allow traffic only from the Application Tier's security group.
7. Verify and finalize the creation of the RDS instance.

Part 5: Testing and Verification

1. Access the public IP of a Web Tier EC2 instance in your browser to check if the website is live.



AWS 3-Tier Web Application-NatureNest

2. SSH into the Web Tier EC2 and ping a private IPv4 (10.0.3.72) address of an Application Tier EC2 instance.

```
[ec2-user@ip-10-0-1-108 ~]$ ping 10.0.3.72
PING 10.0.3.72 (10.0.3.72) 56(84) bytes of data.
64 bytes from 10.0.3.72: icmp_seq=1 ttl=255 time=0.557 ms
64 bytes from 10.0.3.72: icmp_seq=2 ttl=255 time=0.495 ms
64 bytes from 10.0.3.72: icmp_seq=3 ttl=255 time=0.497 ms
64 bytes from 10.0.3.72: icmp_seq=4 ttl=255 time=0.490 ms
64 bytes from 10.0.3.72: icmp_seq=5 ttl=255 time=5.26 ms
64 bytes from 10.0.3.72: icmp_seq=6 ttl=255 time=0.535 ms
64 bytes from 10.0.3.72: icmp_seq=7 ttl=255 time=0.526 ms
64 bytes from 10.0.3.72: icmp_seq=8 ttl=255 time=0.491 ms
64 bytes from 10.0.3.72: icmp_seq=9 ttl=255 time=0.506 ms
64 bytes from 10.0.3.72: icmp_seq=10 ttl=255 time=0.570 ms
64 bytes from 10.0.3.72: icmp_seq=11 ttl=255 time=0.548 ms
64 bytes from 10.0.3.72: icmp_seq=12 ttl=255 time=0.495 ms
64 bytes from 10.0.3.72: icmp_seq=13 ttl=255 time=0.503 ms
64 bytes from 10.0.3.72: icmp_seq=14 ttl=255 time=0.463 ms
64 bytes from 10.0.3.72: icmp_seq=15 ttl=255 time=0.588 ms
64 bytes from 10.0.3.72: icmp_seq=16 ttl=255 time=0.664 ms
64 bytes from 10.0.3.72: icmp_seq=17 ttl=255 time=0.569 ms
^C
--- 10.0.3.72 ping statistics ---
17 packets transmitted, 17 received, 0% packet loss, time 16355ms
rtt min/avg/max/mdev = 0.463/0.809/5.262/1.114 ms
[ec2-user@ip-10-0-1-108 ~]$
```

3. Install MySQL on the Application Tier EC2 instance and use the RDS endpoint to connect to the MySQL database.

   mysql -h <RDS-Endpoint> -P 3306 -u admin -p

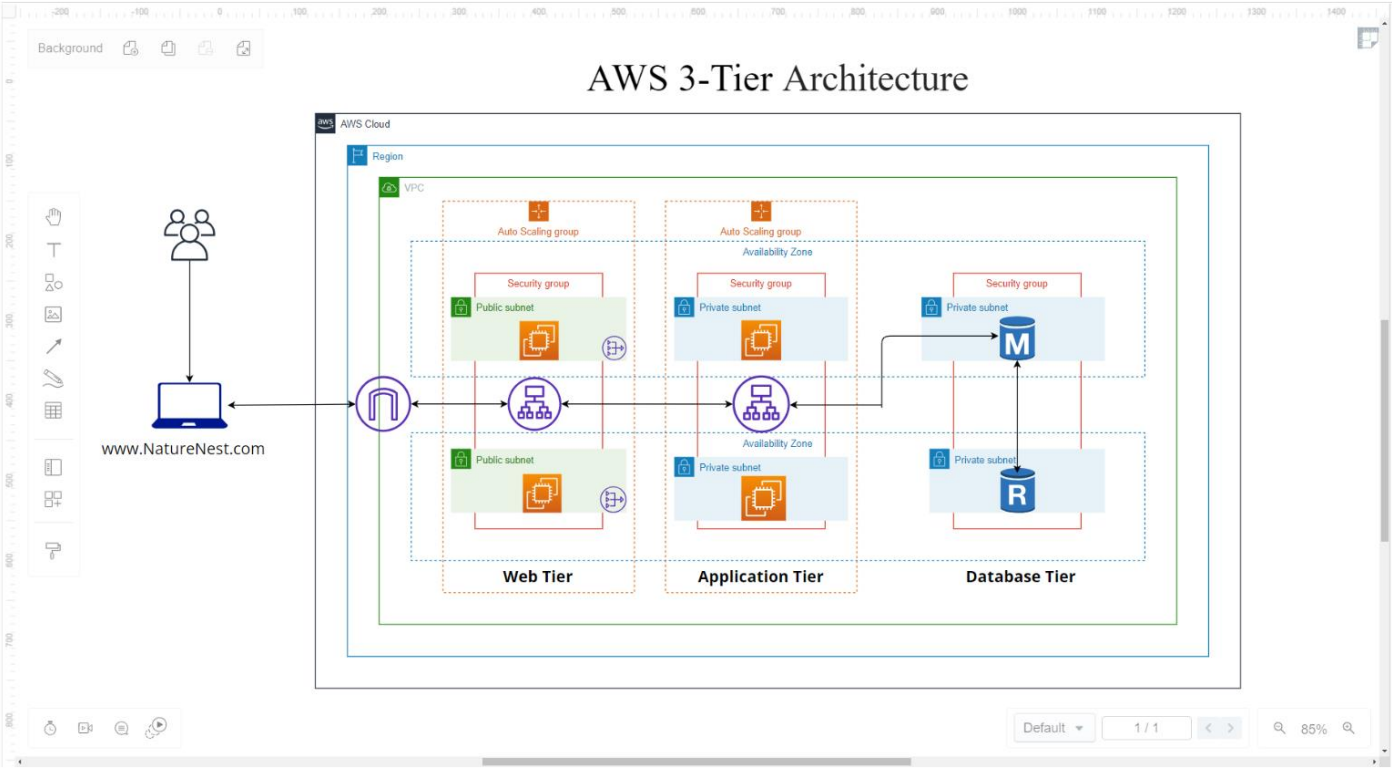
```
[ec2-user@ip-10-0-1-108 ~]$ mysql -h database-1.c7o8seu6yf2w.ap-south-1.rds.amazonaws.com -P 3306 -u admin -p
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MySQL connection id is 55
Server version: 8.0.35 Source distribution

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MySQL [(none)]>
```

AWS 3-Tier Architecture



Conclusion:

The **NatureNest** project successfully demonstrated the implementation of an AWS 3-tier architecture, delivering a scalable, resilient, and secure system. Connectivity tests confirmed smooth interactions between the Web, Application, and Database tiers, showcasing the effectiveness of the design.