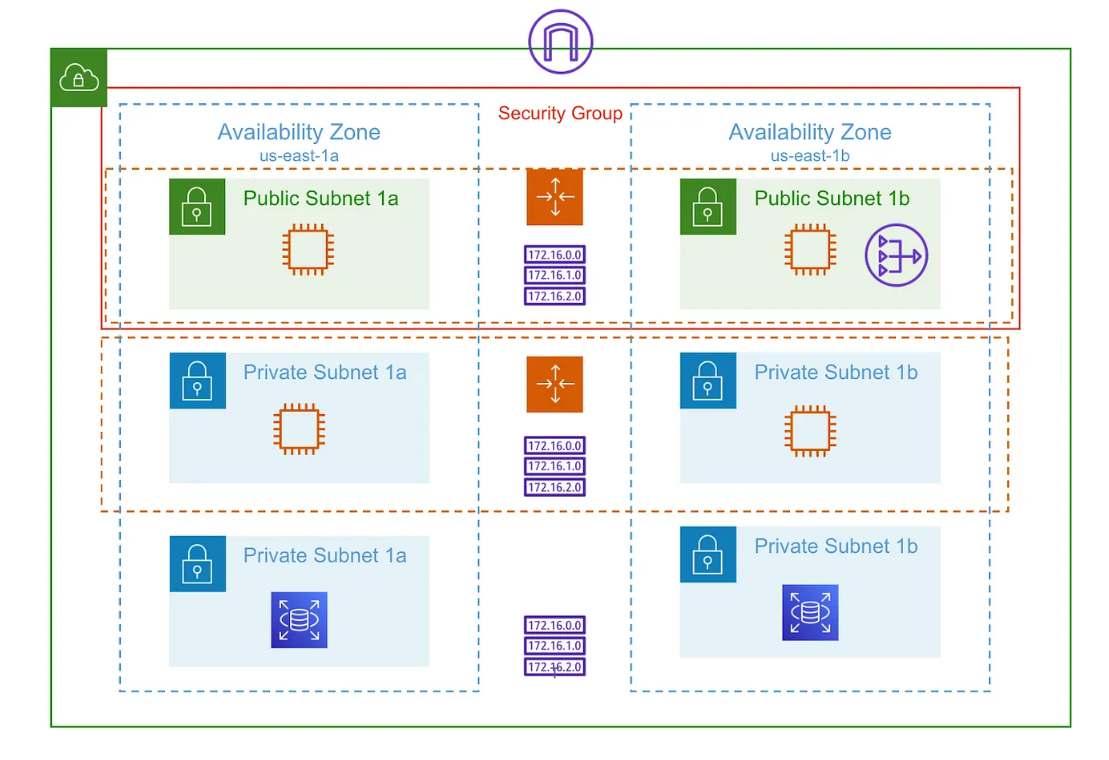
**3-Tier Architecture via AWS CloudFormation**



It’s an architecture that consists of 3 tiers: the web tier, application tier, and database tier.

**web tier** is a top-level tier, and its main purpose is to display and collect information from the user and send its contents to the browser in the form of HTML/JS/CSS. Its main purpose is to display information to and collect information from the user.

**Application tier** is the brains of the application. This tier houses the business logic used to process user inputs. The application tier can also add, delete or modify data in the database tier.

**Database tier** is the data or backend tier of a web application. It is where the information processed by the application is stored and managed.

**1| CloudFormation Template**

You can either create your own template in Visual Studio Code or fork my CloudFormation Template and save it as a 2-ThreeTierArchitecture.yml file on your local computer.

The code(2-ThreeTierArchitecture.yml) will build out the following:

VPC

Route Table

Internet Gateway

Attaches Internet Gateway to the VPC you just created.

2 Public Subnets

Scaling Policy for Public Subnets

Route Table for Public Subnets

Associate the Public Subnets to the Route Table

Security group with HTTP & SSH to open to 0.0.0.0/0

Launch Template with Bootstrap script to install & launch Apache.

Autoscaling Group for Public Subnets

Scaling Policy

4 Private Subnets — two in us-east1a & two in us-east1b

AutoScaling Group for two Private Subnets in the Application tier

**2| Create Your Stack** (Assuming the IAM role is already created if not create using 1-CreateIamRole.yml)

In your AWS Console, navigate to CloudFormation:

Click Create Stack

Select the Template is ready option.

Under Template source, select Upload a template file.

Upload the 2-ThreeTierArchitecture.yml file from the previous step.

Click Next

Next, create your Stack Name

Assign your IAM role.

Stack failure options: Roll back all stack resources.

Leave everything as default and click Next.

Review your template and Click Create stack.

Once you’ve hit Create Stack, the events will take a while to fully create your stack.

**3| Confirm Apache installed on EC2 Instances**

In your AWS Console, navigate to EC2. If you recall from our code, we specified that we wanted to launch four EC2 instances with user data to install Apache Web Server. Two instances in our web tier and two in our application tier.

Grab the Public IPv4 address from the Web tier to verify that Apache was correctly installed on each instance. As you can see Apache Test Page, the installation of Apache worked perfectly!

**4| Create Database Tier for your Architecture.**

save 3-RDSMySql.yml file on your local computer.

In your AWS CloudFormation Console, navigate to Create Stack:

Select the Template is ready option.

Under Template source, select Upload a template file.

Upload the YAML file that you created or attach the .yml file from the previous step.

Click Next

Parameters:

Under Ownership, simply write your name or the Team name deploying the Database Stack

Network Configuration:

VPC: Select the VPC we created with the first stack: ThreeTierArch-Project-cloudVPC.

Private Subnet 1: Put this in the Private subnet 10.0.64.0/20.

Private Subnet 2: Put this in the Private subnet 10.0.80.0/20.

Public Network ACL: Navigate to VPC>Network ACLs and get the Network ACL ID number.

Leave everything else as default and click Next.

Review your template and Click Create stack.

**5| Create a NAT Gateway to route all traffic.**

save 4-NATGateway.yml file on your local computer.

In your AWS CloudFormation Console, navigate to Create Stack:

Select the Template is ready option.

Under Template source, select Upload a template file.

Upload the YAML file that you created or attach the .yml file from the previous step.

Click Next

**Specify stack details:**

Stack name: NATGateway-Project

**Parameters:**

Under Ownership, simply write your name or the Team name deploying the Database Stack

Network Configuration:

VPC: Select the VPC we created with the first stack: ThreeTierArch-Project-cloudVPC.

Public Subnet: Put this in the public subnet 10.0.16.0/20.

Private Route Table: Navigate to VPC>Route tables and get the Route table ID number.

Public Network ACL: Navigate to VPC>Network ACLs and get the Network ACL ID number.

Click Next and deploy your stack.

As you can tell, our final stack was successfully deployed!!

Assign the NATGateway created above to private Route Table (Edit routes add 0.0.0.0/0 ----> NAT Gateway).

Ping the Private App instance from your Public Web instance with the following command:

$ping 10.x.x.x