Steps Followed To Run a 3 Tier Application On AWS

VPC Setup

1. A Custom VPC is created with CIDR range 10.0.0.0/16
2. Then a public subnet and two private subnets that are database and application, are created .
3. Then created an Internet gateway and attached it to the VPC .
4. Then created a public route table ,application route table and database route table .
5. Public route table is associated with the public subnet , application route table associated with application subnet and database route table is associated with the database route table .
6. Then created a NAT gateway on the public subnet , this will help the instances inside the private subnet to connect the internet .
7. Then created a route for the traffic from the private subnet to NAT gateway and a route for the traffic from public subnet to Internet gateway .

Setup The Application To Run

1. Two EC2 instances are on the public subnet , one EC2 instance is used as *Frontend server* for accessing the Application and another EC2 instance is used as *Bastion host* , which helps in accessing the private EC2 instances (i.e. database and application) .
2. Another EC2 instance is created by using database subnet and another EC2 instance is created by using application subnet .

3. Then connected the private EC2 instances (i.e. database and application) through the bastion host and installed docker on both the instances .
4. After this on the database EC2 instance , run the mongodb database using the command :

   docker run -d -p 27017:27017 –name mongo mongo

5. On the application server created a script through which the application is going to run :

   “

   ```
   #!/bin/bash
   # Before running the script
   if docker ps | grep aquila;then
       docker stop aquila
       docker rm aquila
   fi
   # Set environment variables
   export MONGODB_URI=mongodb://10.0.0.48:27017/
   export LANGUAGE=en
   export FIRSTNAME=Admin
   export LASTNAME=User
   export EMAIL=admin@gmail.com
   export PASSWORD=Pranit123
   export APPURL=http://10.0.1.74:3010
   export ADMIN_PREFIX=admin
   export SITENAME=demo

   # Start Docker container with environment variables
   echo "Starting Docker container..."
   docker run -d \
   ```

```
  -p 10.0.1.74:3010:3010/tcp \
  --name aquila \
  -e MONGODB_URI=$MONGODB_URI \
  -e LANGUAGE=$LANGUAGE \
  -e FIRSTNAME=$FIRSTNAME \
  -e LASTNAME=$LASTNAME \
  -e EMAIL=$EMAIL \
  -e PASSWORD=$PASSWORD \
  -e APPURL=$APPURL \
  -e ADMIN_PREFIX=$ADMIN_PREFIX \
  -e SITENAME=$SITENAME \
  aquilacms/aquilacms

# Monitor installation process
echo "Installing.."
sleep 150
echo "RESTARTING THE CONTAINER..."
sudo docker restart aquila
echo "SUCCESSFULLY RESTARTED..."

"
```

6. Even the application successfully ran on the application EC2 instance , but we can not still connect the application .
7. To connect the application successfully we have to create a nginx reverse proxy on the frontend EC2 instance that is present on the public subnet .

```
server {
   listen 80;
   server_name 100.24.34.50;
```

```
    location / {
        proxy_pass http://10.0.1.74:3010;
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For
$proxy_add_x_forwarded_for;
        proxy_set_header X-Forwarded-Proto $scheme;
    }
}
```

8. After this successfully connected the application on the ip :

100.24.34.50:80

Conclusion

Here the application server and database server can not be accessed directly by the users . Only admins can access the application and database server by using the private IP and SSH command on the bastion host .