3-Tier Architecture Deployment on AWS using Terraform(from scratch, all in vars)

Here is my Set up how to build 3 tier architecture with Terraform from scratch –

1. Introduction

A **3-Tier Architecture** is a common design pattern in cloud infrastructure that divides the application into three logical layers:

- **Presentation Layer (Web Tier):** Hosts the user interface and handles HTTP/S requests (Nginx).
- **Application Layer (App Tier):** Processes business logic (Tomcat).
- **Database Layer (DB Tier):** Stores and manages application data (MySQL).

In this project, we are deploying a **highly available 3-tier architecture** on AWS using **Terraform** as Infrastructure as Code (IaC).

2. Project Requirements

- **Region:** us-east-1
- **Key Pair:** pavankey
- **Instance OS:** Ubuntu
- **Software Installed:**
    - Web Server: **Nginx**
    - App Server: **Tomcat**
    - DB Server: **MySQL**

3. Design Components

a) VPC (Virtual Private Cloud)

A dedicated network created to host all resources.

- CIDR: 10.0.0.0/16
- Provides isolation and control over networking.

b) Subnets

- **Public Subnet:** For the Web Server (Nginx).
- **Private Subnet (App):** For the Application Server (Tomcat).
- **Private Subnet (DB):** For the Database Server (MySQL).
- Subnets are placed in different **Availability Zones** to ensure high availability.

c) Internet Gateway & NAT Gateway

- **Internet Gateway:** Allows public access to the web server.

- **NAT Gateway:** Allows private instances (App & DB servers) to access the internet securely for updates.

d) Route Tables

- **Public Route Table:** Routes traffic from the public subnet to the Internet Gateway.
- **Private Route Table:** Routes traffic from private subnets to the NAT Gateway for internet access.

e) Security Groups

- **Web SG:** Allows HTTP (80), HTTPS (443), and SSH (22) from anywhere.
- **App SG:** Allows traffic only from Web SG.
- **DB SG:** Allows MySQL (3306) traffic only from App SG.

f) EC2 Instances

- **Web Server (Nginx):** Handles client requests.
- **App Server (Tomcat):** Processes application logic.
- **DB Server (MySQL):** Stores and retrieves data.

Each EC2 instance uses a **user_data script** to automatically install the required software.

4. Workflow

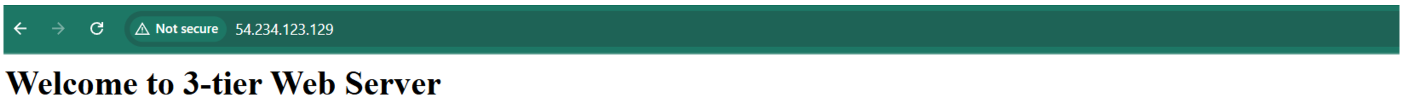
1. **Terraform Init** → Initializes the working directory and downloads AWS provider plugins.
2. **Terraform Plan** → Previews resources to be created.
3. **Terraform Apply** → Provisions the VPC, subnets, gateways, route tables, security groups, and EC2 instances.
4. **Access the Web Server:**
   - Copy the **Public IP / DNS** of the web server.
   - Paste into a browser → Nginx default page should load.
5. **App & DB Communication:**
   - Web Server connects to App Server.
   - App Server connects to DB Server.
6. All scripts are attached in these Notion template so click and open it for guidance

3-tier Architecture script file

SO NOW LETS START CHECKING COMMUNICATION BETWEEN MACHINES & INSTALLATION ARE WORKING OR NOT

SO IT WORKING In my web server I wrote “welcome to 3 tier”

So here is the image of webserver

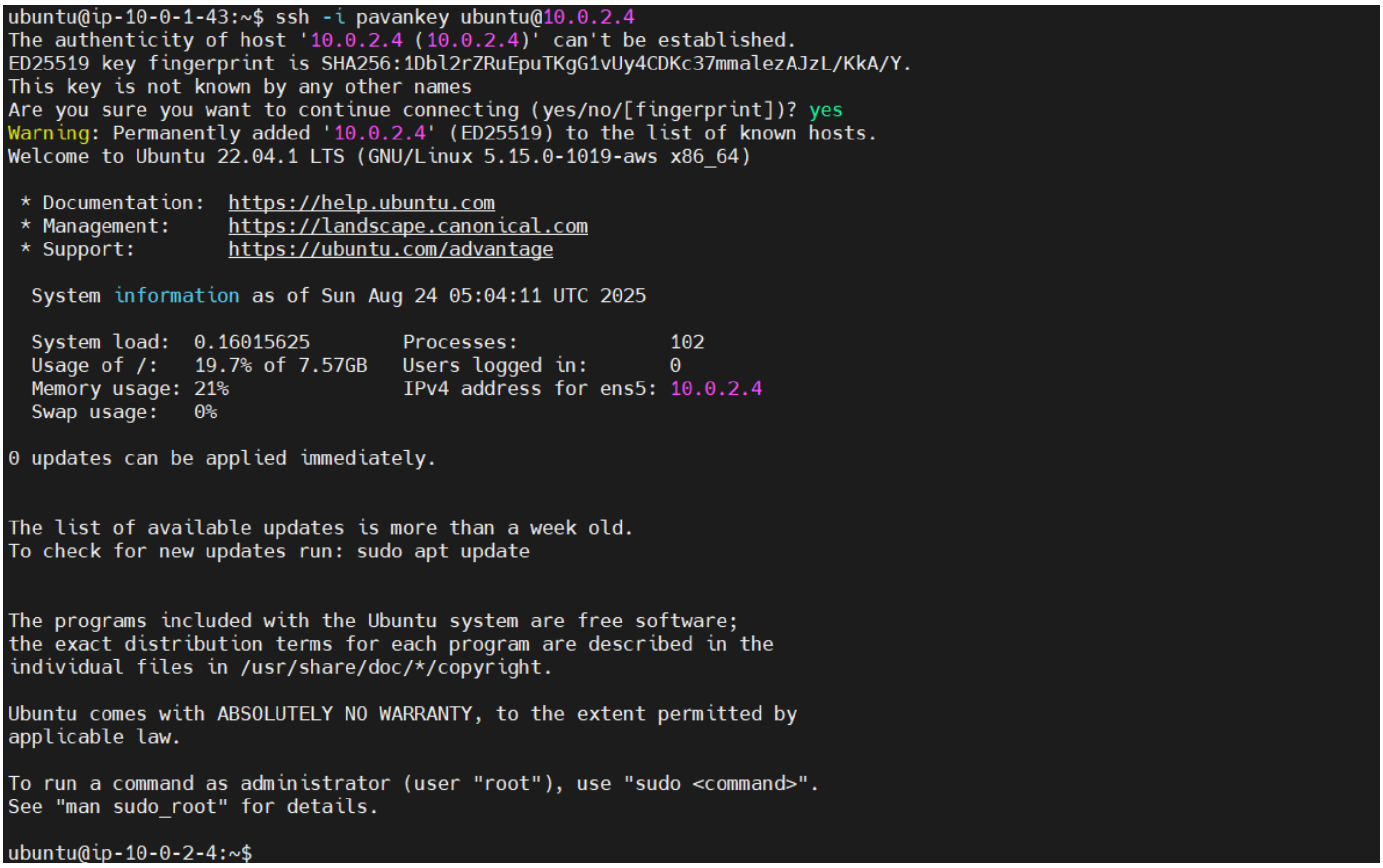


54.234.123.129

Welcome to 3-tier Web Server

Next lets work with app server

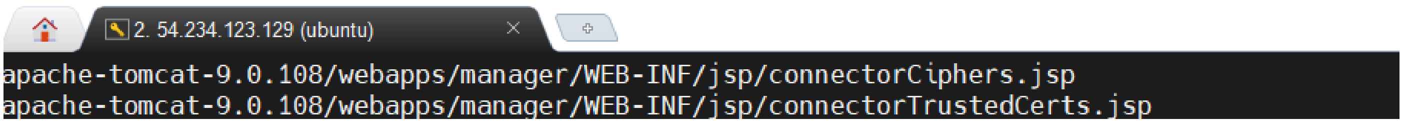
Now I connected with web server to app server via mobaxtrem with key and web ip



```
ubuntu@ip-10-0-1-43:~$ ssh -i pavankey ubuntu@10.0.2.4
The authenticity of host '10.0.2.4 (10.0.2.4)' can't be established.
ED25519 key fingerprint is SHA256:1Dbl2rZRuEpuTKgG1vUy4CDKc37mmalezAJzL/KkA/Y.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '10.0.2.4' (ED25519) to the list of known hosts.
Welcome to Ubuntu 22.04.1 LTS (GNU/Linux 5.15.0-1019-aws x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

  System information as of Sun Aug 24 05:04:11 UTC 2025

  System load:  0.16015625         Processes:             102
  Usage of /:   19.7% of 7.57GB    Users logged in:       0
  Memory usage: 21%                IPv4 address for ens5: 10.0.2.4
  Swap usage:   0%

0 updates can be applied immediately.


The list of available updates is more than a week old.
To check for new updates run: sudo apt update


The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

ubuntu@ip-10-0-2-4:~$
```

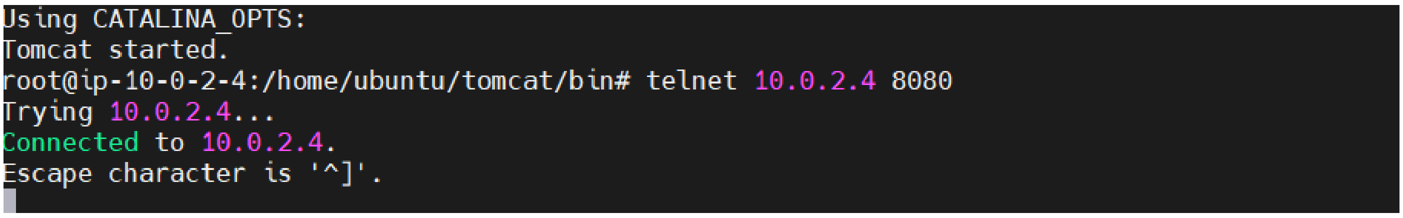
Now lets install tomcat or check tomcat port with our private using " TELNET"

So tomcat installed and connected with my Private machine



2. 54.234.123.129 (ubuntu)

```
apache-tomcat-9.0.108/webapps/manager/WEB-INF/jsp/connectorCiphers.jsp
apache-tomcat-9.0.108/webapps/manager/WEB-INF/jsp/connectorTrustedCerts.jsp
```

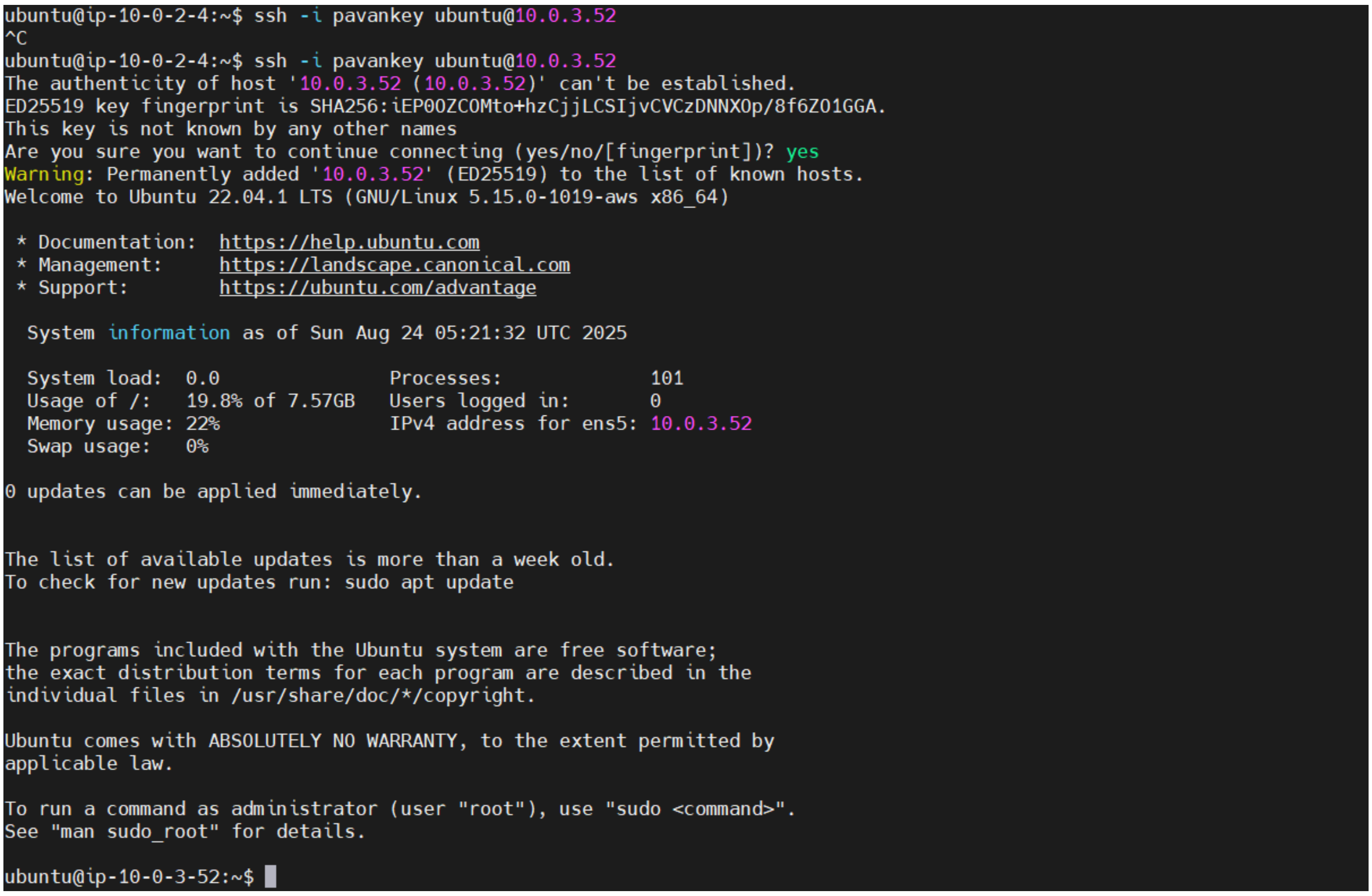


```
Using CATALINA_OPTS:
Tomcat started.
root@ip-10-0-2-4:/home/ubuntu/tomcat/bin# telnet 10.0.2.4 8080
Trying 10.0.2.4...
Connected to 10.0.2.4.
Escape character is '^]'.
```

So Now lets go to my db machine and check SQL installation and port connection

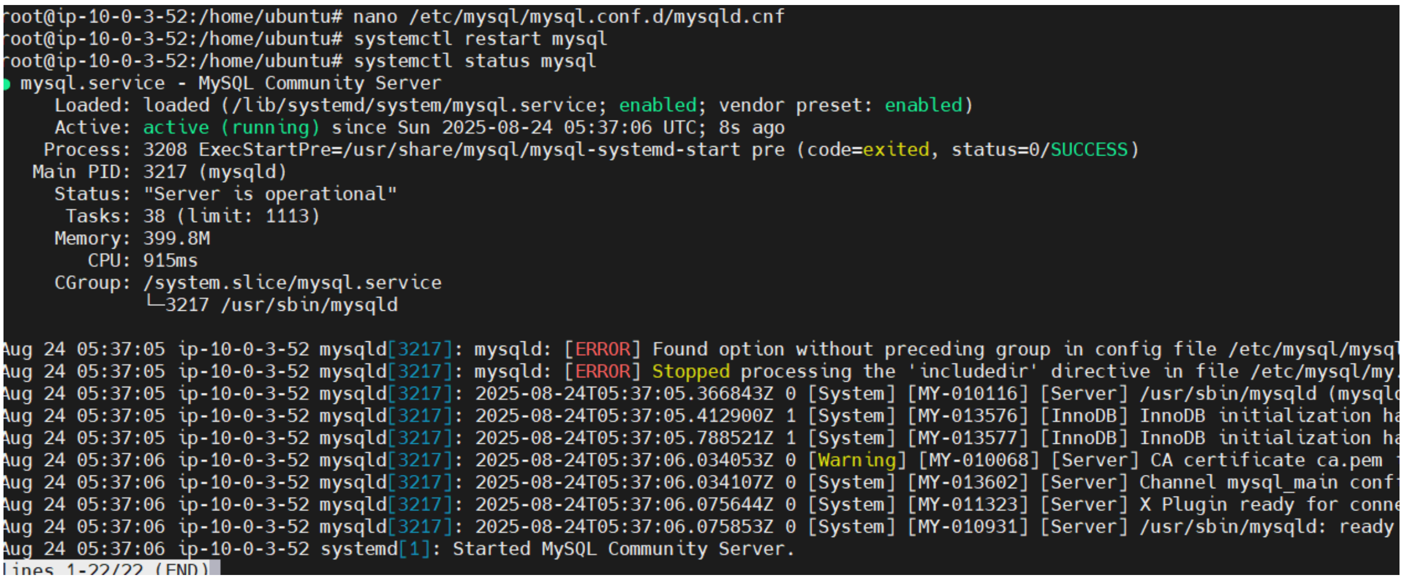
So now lets do ssh in my private for connecting to db machine

Now I connected to db machine from app machine now lets go with sql installation and port connection with machine.



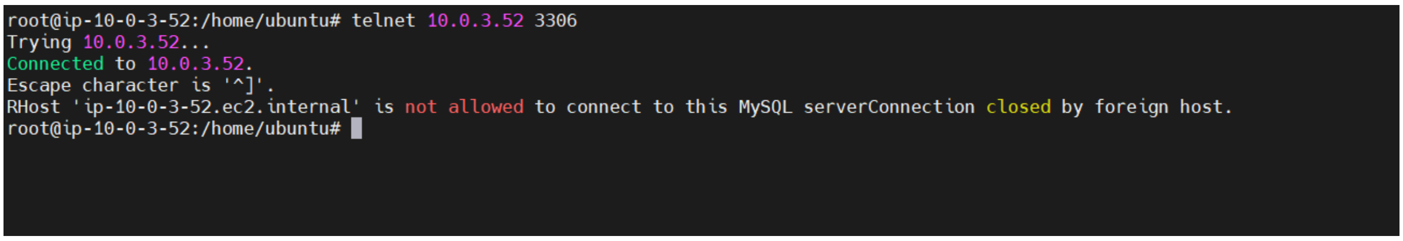
```
ubuntu@ip-10-0-2-4:~$ ssh -i pavankey ubuntu@10.0.3.52
^C
ubuntu@ip-10-0-2-4:~$ ssh -i pavankey ubuntu@10.0.3.52
The authenticity of host '10.0.3.52 (10.0.3.52)' can't be established.
ED25519 key fingerprint is SHA256:iEP0OZCOMto+hzCjjLCSIjvCVCzDNNXOp/8f6ZO1GGA.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '10.0.3.52' (ED25519) to the list of known hosts.
Welcome to Ubuntu 22.04.1 LTS (GNU/Linux 5.15.0-1019-aws x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

  System information as of Sun Aug 24 05:21:32 UTC 2025

  System load:  0.0               Processes:             101
  Usage of /:   19.8% of 7.57GB   Users logged in:       0
  Memory usage: 22%               IPv4 address for ens5: 10.0.3.52
  Swap usage:   0%

0 updates can be applied immediately.


The list of available updates is more than a week old.
To check for new updates run: sudo apt update


The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

ubuntu@ip-10-0-3-52:~$
```

So my MYSQL server started after installation



```
root@ip-10-0-3-52:/home/ubuntu# nano /etc/mysql/mysql.conf.d/mysqld.cnf
root@ip-10-0-3-52:/home/ubuntu# systemctl restart mysql
root@ip-10-0-3-52:/home/ubuntu# systemctl status mysql
● mysql.service - MySQL Community Server
     Loaded: loaded (/lib/systemd/system/mysql.service; enabled; vendor preset: enabled)
     Active: active (running) since Sun 2025-08-24 05:37:06 UTC; 8s ago
    Process: 3208 ExecStartPre=/usr/share/mysql/mysql-systemd-start pre (code=exited, status=0/SUCCESS)
   Main PID: 3217 (mysqld)
     Status: "Server is operational"
      Tasks: 38 (limit: 1113)
     Memory: 399.8M
        CPU: 915ms
     CGroup: /system.slice/mysql.service
             └─3217 /usr/sbin/mysqld

Aug 24 05:37:05 ip-10-0-3-52 mysqld[3217]: mysqld: [ERROR] Found option without preceding group in config file /etc/mysql/mysq
Aug 24 05:37:05 ip-10-0-3-52 mysqld[3217]: mysqld: [ERROR] Stopped processing the 'includedir' directive in file /etc/mysql/my.
Aug 24 05:37:05 ip-10-0-3-52 mysqld[3217]: 2025-08-24T05:37:05.366843Z 0 [System] [MY-010116] [Server] /usr/sbin/mysqld (mysqld
Aug 24 05:37:05 ip-10-0-3-52 mysqld[3217]: 2025-08-24T05:37:05.412900Z 1 [System] [MY-013576] [InnoDB] InnoDB initialization ha
Aug 24 05:37:05 ip-10-0-3-52 mysqld[3217]: 2025-08-24T05:37:05.788521Z 1 [System] [MY-013577] [InnoDB] InnoDB initialization ha
Aug 24 05:37:06 ip-10-0-3-52 mysqld[3217]: 2025-08-24T05:37:06.034053Z 0 [Warning] [MY-010068] [Server] CA certificate ca.pem i
Aug 24 05:37:06 ip-10-0-3-52 mysqld[3217]: 2025-08-24T05:37:06.034107Z 0 [System] [MY-013602] [Server] Channel mysql_main conf
Aug 24 05:37:06 ip-10-0-3-52 mysqld[3217]: 2025-08-24T05:37:06.075644Z 0 [System] [MY-011323] [Server] X Plugin ready for conne
Aug 24 05:37:06 ip-10-0-3-52 mysqld[3217]: 2025-08-24T05:37:06.075853Z 0 [System] [MY-010931] [Server] /usr/sbin/mysqld: ready
Aug 24 05:37:06 ip-10-0-3-52 systemd[1]: Started MySQL Community Server.
lines 1-22/22 (END)
```

So now lets connection, after checking with port number 3306 with machine ip ,it showing connected it end of 3-tier architecture – A/C to me



```
root@ip-10-0-3-52:/home/ubuntu# telnet 10.0.3.52 3306
Trying 10.0.3.52...
Connected to 10.0.3.52.
Escape character is '^]'.
RHost 'ip-10-0-3-52.ec2.internal' is not allowed to connect to this MySQL serverConnection closed by foreign host.
root@ip-10-0-3-52:/home/ubuntu#
```

5. Benefits of Using Terraform

- **Infrastructure as Code (IaC):** Entire setup is codified and version-controlled.
- **Automation:** Reduces manual configuration efforts.
- **Scalability:** Easy to scale by updating variables.
- **Reproducibility:** Same infrastructure can be recreated in any environment.

6. Conclusion

Using **Terraform**, we successfully automated the deployment of a **3-tier architecture** on AWS.

- The **Web Tier** runs Nginx in a public subnet.
- The **App Tier** runs Tomcat in a private subnet.
- The **DB Tier** runs MySQL in a private subnet.

7. Verification & Testing

After Terraform successfully created the infrastructure, the setup was validated through the following steps:

1. **SSH Access Between Machines**
    - Logged into each EC2 instance using the key pair pavankey.
    - Verified that instances were reachable via **private IPs** inside the VPC.
2. **Manual Software Installation Check**
    - Web Server: Verified **Nginx** installation by running systemctl status nginx and accessing it via the public IP in a browser.
    - App Server: Confirmed **Tomcat** installation and checked service availability on its respective port (default **8080**).
    - DB Server: Verified **MySQL** installation, logged into the database with mysql -u root -p, and ensured it was listening on port **3306**.
3. **Port & Connectivity Validation**
    - Used telnet <private-ip> <port> and netstat -tulnp to confirm that ports **80, 8080, and 3306** were open and services were running.

- Ensured that **Security Groups** allowed only the required traffic flow:
    - Web SG → App SG (port 8080).
    - App SG → DB SG (port 3306).

4. **End-to-End Flow**
    - Accessed the Web Server’s public IP via browser → confirmed Nginx response.
    - Verified that the Web Server could connect to the App Server.
    - Checked App Server’s ability to query the MySQL DB instance.