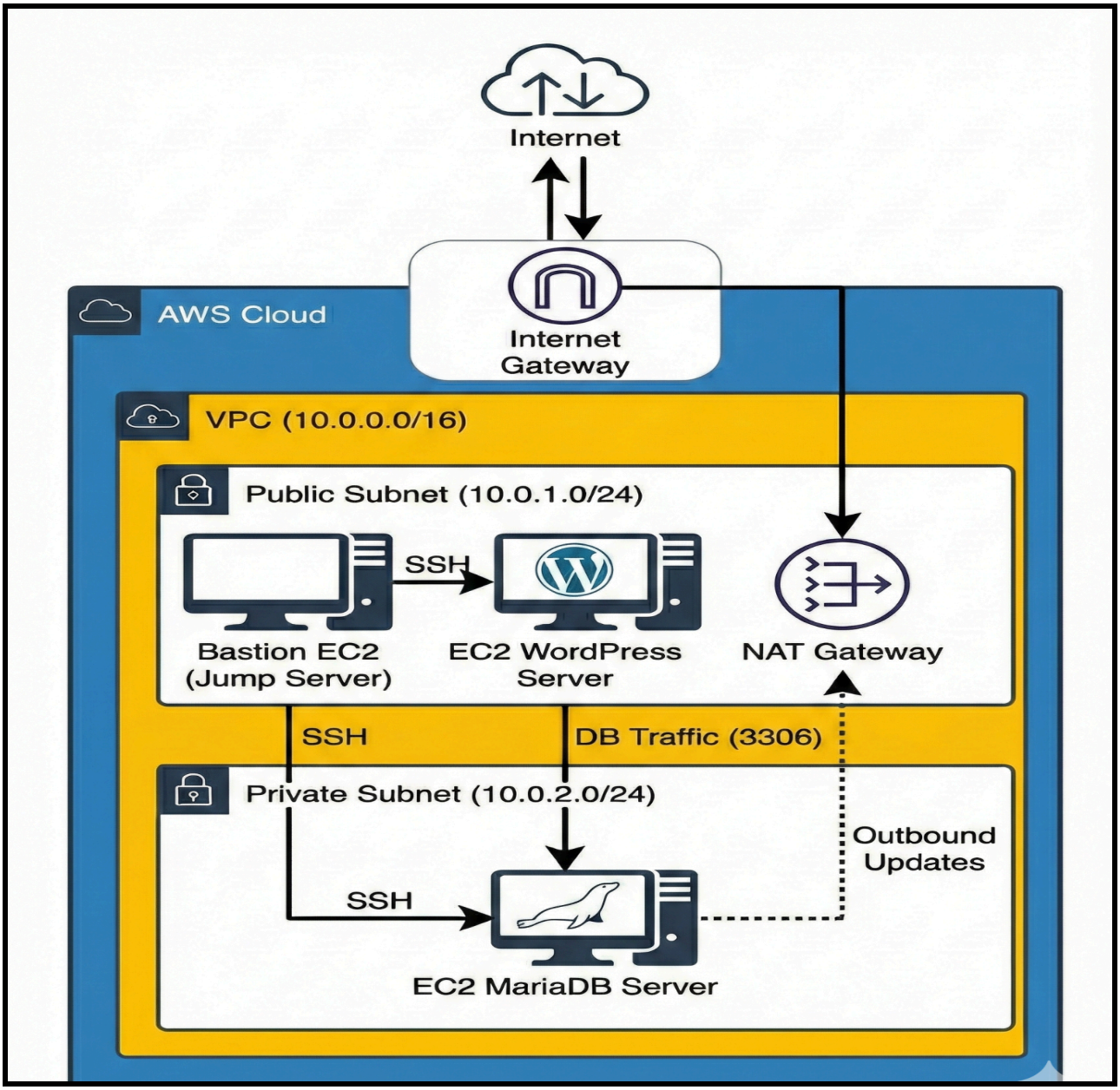
AWS Multi-Tier Architecture for WordPress with Bastion Host and NAT Gateway



1. Objective

The goal of this project is to deploy a secure, scalable WordPress application on AWS using a multi-tier architectural approach. By isolating the database in a private subnet and utilizing a Jump Server (Bastion Host), we ensure that sensitive infrastructure is protected from direct public internet exposure.

Key Features

- **Jump Server: Secure SSH entry point for administrative tasks.**
- **Three-Tier Logic: Web tier in public subnets, Data tier in private subnets.**
- **NAT Gateway: Enables private instances to download updates without being reachable from the internet.**
- **Enhanced Security: Granular Security**
- **Group rules to restrict traffic flow**

2. Architecture Overview

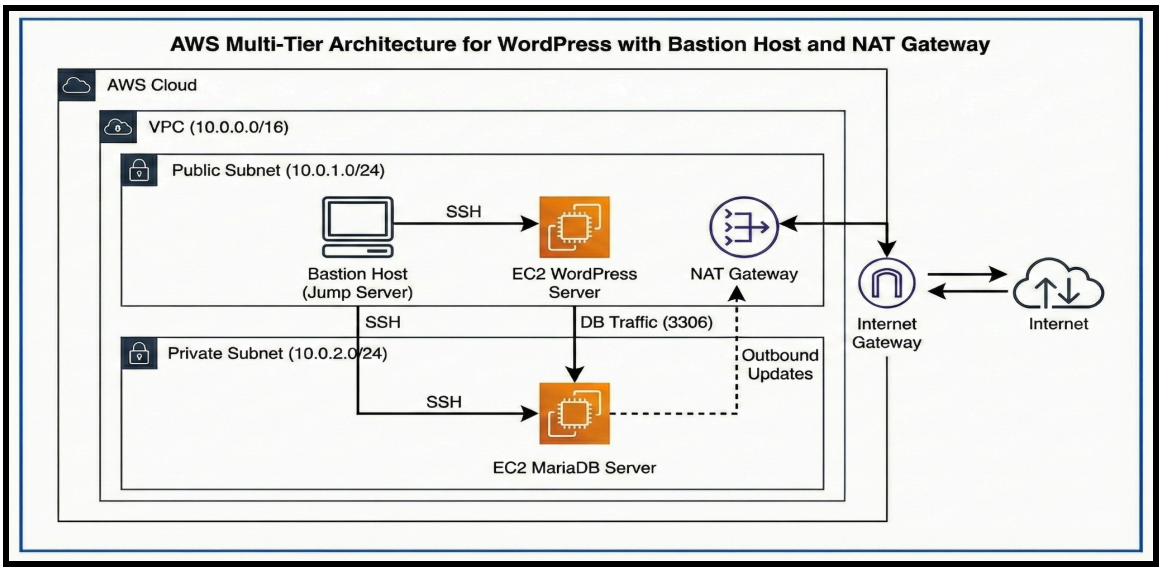
Network Flow

Internet <-> Internet Gateway <-> VPC
Public Subnet (10.0.1.0/24)

- **Jump Host: SSH entry point.**
- **WordPress EC2: Running Apache + PHP.**
- **NAT Gateway: Attached to an Elastic IP for outbound private traffic.**

Private Subnet (10.0.2.0/24)

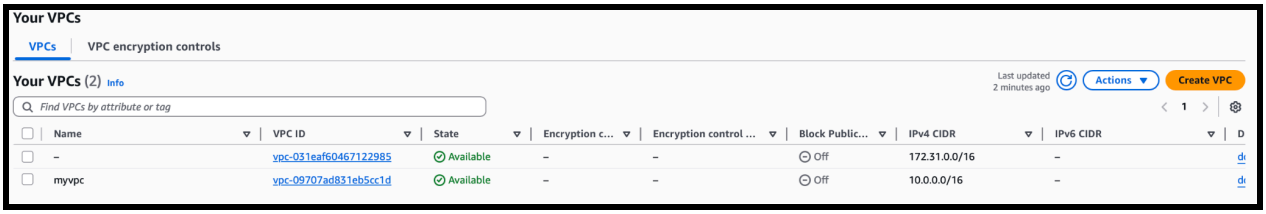
- **MariaDB EC2: No public IP; accessible only via Jump Host and Web Server.**



3. Infrastructure Setup

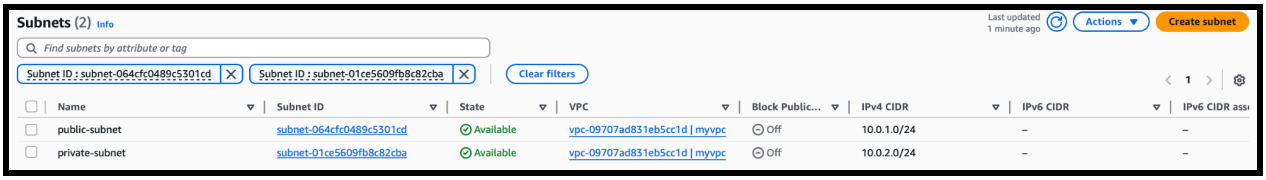
STEP 1: VPC Creation

1. **Navigate to the VPC Dashboard.**
2. **Select Create VPC.**
3. **Name: WP-VPC**
4. **IPv4 CIDR: 10.0.0.0/16**



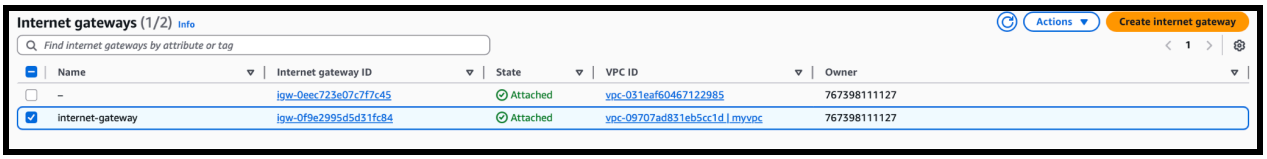
STEP 2: Subnet Creation

- **Public Subnet:**
  - **Name: Public-Subnet**
  - **AZ: ap-south-1a**
  - **CIDR: 10.0.1.0/24**
- **Private Subnet:**
  - **Name: Private-Subnet**
  - **AZ: ap-south-1a**
  - **CIDR: 10.0.2.0/24**



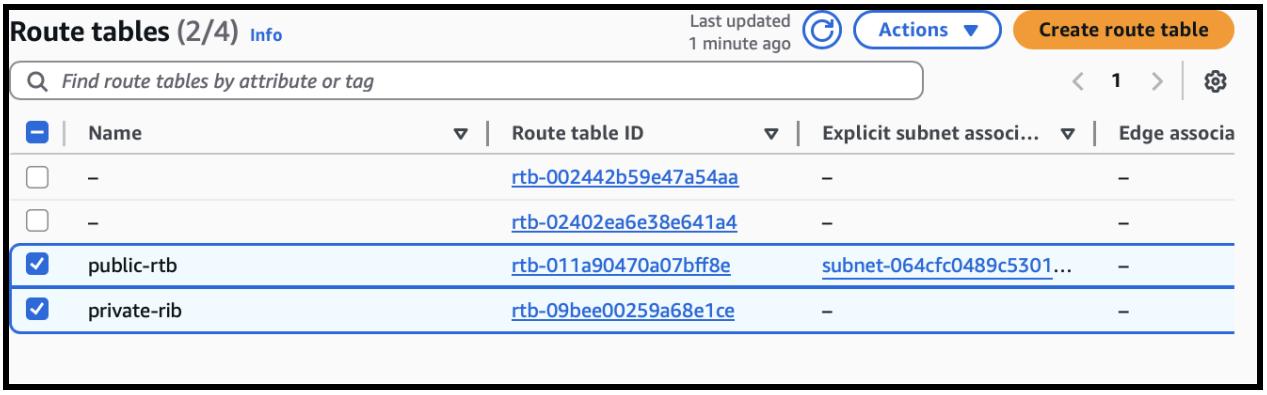
STEP 3: Internet Gateway (IGW)

1. Create IGW named WP-IGW.
2. Attach WP-IGW to WP-VPC.

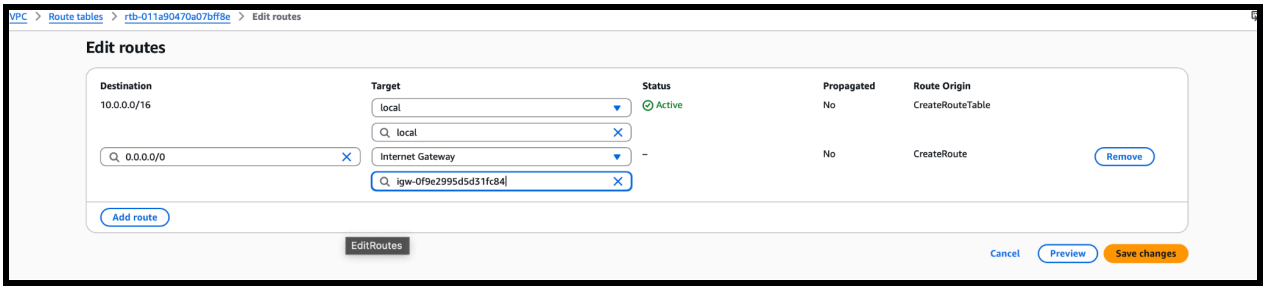


STEP 4: Route Table Configuration

- Public-RT:
  - Routes: 0.0.0.0/0 -> WP-IGW
  - Association: Public-Subnet
- Private-RT:
  - Routes: 10.0.0.0/16 -> local (Initial state)
  - Association: Private-Subnet



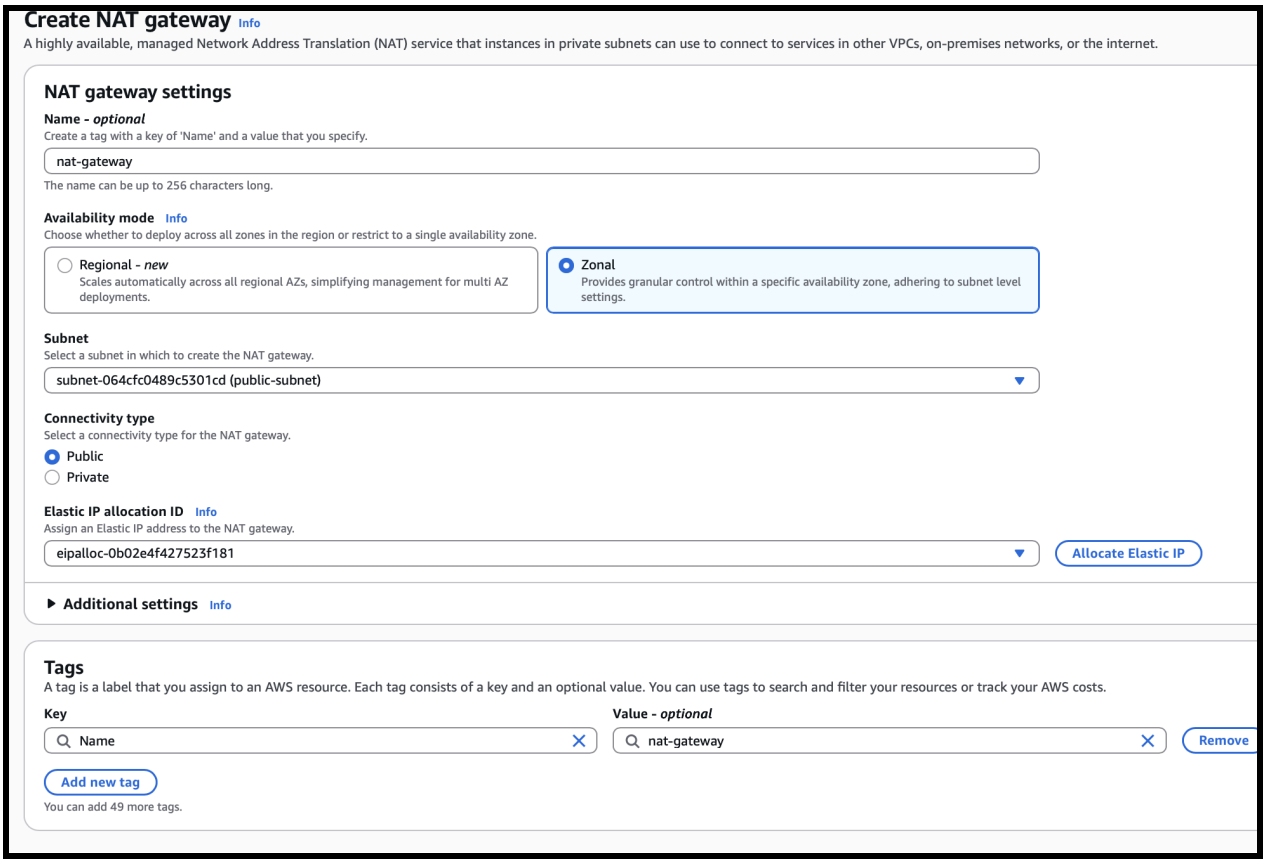
UPDATE PUBLIC-RT ROUTES :



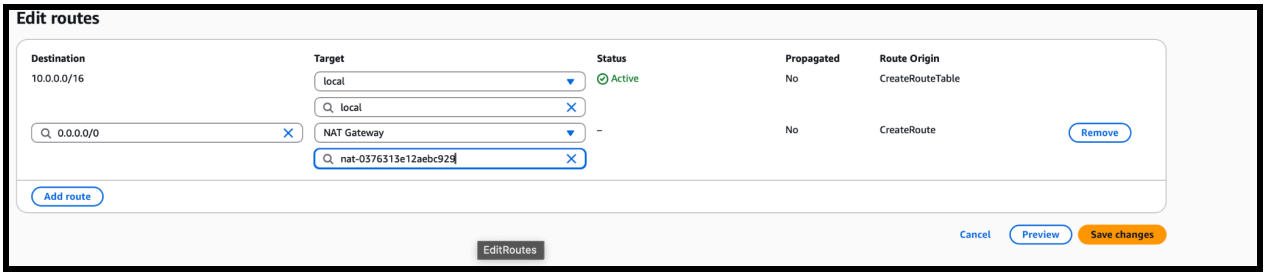
STEP 5: NAT Gateway Setup

1. Elastic IP: Allocate a new EIP.
2. NAT Gateway: Name: nat-gateway
   - Subnet: Public-Subnet

- **EIP: Select the allocated EIP.**



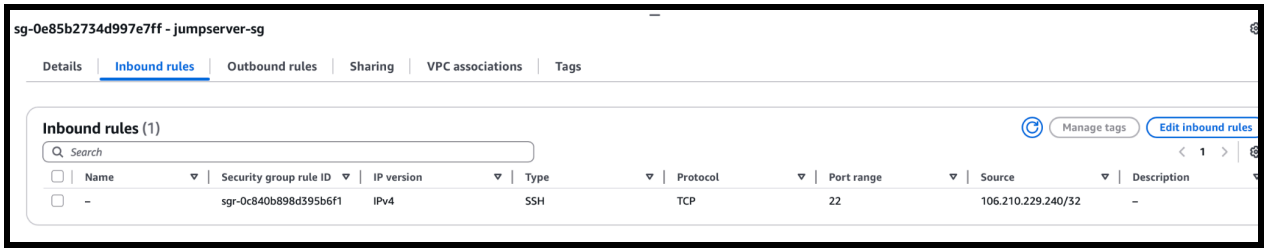
3. **Update Private-RT: Add route 0.0.0.0/0 -> NAT Gateway (nat-gateway).**

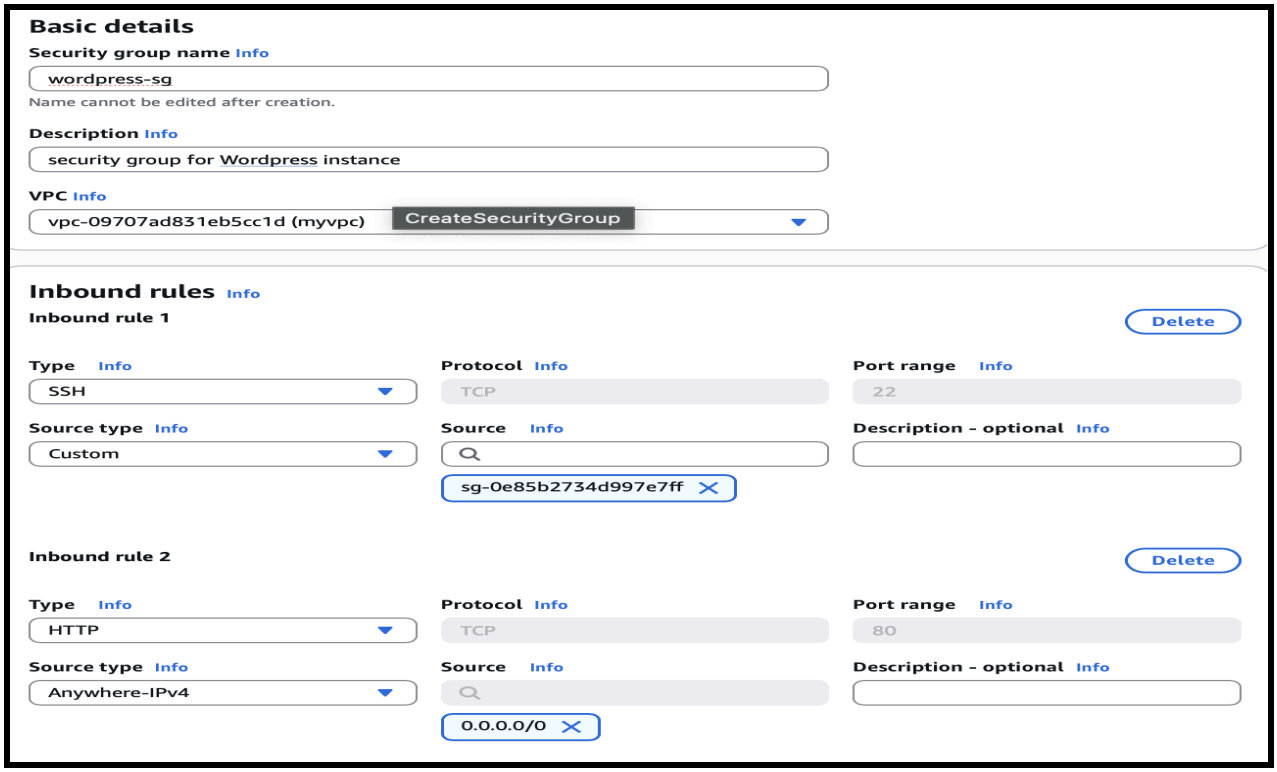


4. Security & Compute

STEP6: Security Group Configuration

Security Group	Inbound Rules	Source
Jump-SG	SSH (22)	Your Public IP (/32)
WP-SG	SSH (22) HTTP (80)	Jump-SG 0.0.0.0/0
DB-SG	SSH (22) MySQL (3306)	Jump-SG WP-SG





Basic details

Security group name Info

mariadb-sg

Name cannot be edited after creation.

Description Info

to access mariadb privately

VPC Info

vpc-09707ad831eb5cc1d (myvpc)

Inbound rules Info

Inbound rule 1 Delete

CreateSecurityGroup

Type Info	Port range Info
MYSQL/Aurora	TCP / 3306

Source type Info	Source Info	Description - optional Info
Custom	sg-0d27faf74ac16c1d0	

Inbound rule 2 Delete

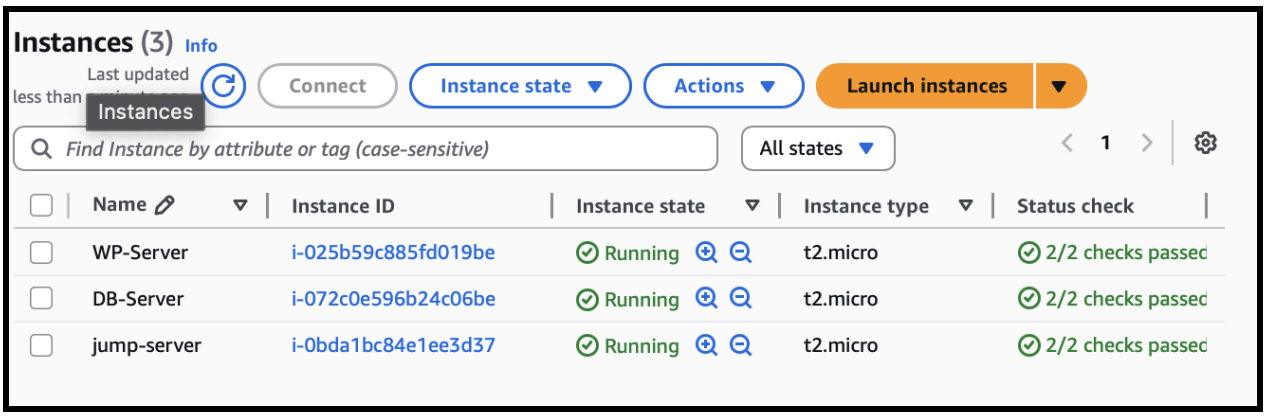
Type Info	Protocol Info	Port range Info
SSH	TCP / 22	

Source type Info	Source Info	Description - optional Info
Custom	sg-0e85b2734d997e7ff	

Step7: EC2 Instance Launch

General Specs: Amazon Linux 2023, t2.micro, wp-key.pem.

1. **Jump Host: Public Subnet, Auto-assign Public IP: Enabled, SG: Jump-SG.**
2. **WordPress Server: Public Subnet, Auto-assign Public IP: Enabled, SG: WP-SG.**
3. **MariaDB Server: Private Subnet, Auto-assign Public IP: Disabled, SG: DB-SG.**



5. Deployment & Configuration

STEP 8: SSH Access Flow

Securely hop from your machine to the private infrastructure:

Set permissions
#chmod 400 wp-key.pem
Connect to Jump Host
#ssh -i wp-key.pem ec2-user@JUMP_PUBLIC_IP

```
mayurnikam@MAYURs-MacBook-Air Downloads % chmod 400 "key-2.pem"
mayurnikam@MAYURs-MacBook-Air Downloads % ssh -i "key-2.pem" ec2-user
@54.160.93.52
   ,     #_
   ~\_  ####_        Amazon Linux 2023
  ~~  \_#####\
  ~~     \###|
  ~~       \#/ ___   https://aws.amazon.com/linux/amazon-linux-2023
   ~~       V~' '->
    ~~~         /
      ~~._.   _/
         _/ _/
       _/m/'
Last login: Wed Jan 21 16:10:28 2026 from 106.210.229.240
[ec2-user@ip-10-0-1-121 ~]$
```

From Jump Host to WordPress

ssh -i wp-key.pem ec2-user@WP_PRIVATE_IP

```
[ec2-user@ip-10-0-1-121 ~]$ chmod 400 key-2.pem
[ec2-user@ip-10-0-1-121 ~]$ ssh -i key-2.pem ec2-user@10.0.1.177
   ,     #_
   ~\_  ####_        Amazon Linux 2023
  ~~  \_#####\
  ~~     \###|
  ~~       \#/ ___   https://aws.amazon.com/linux/amazon-linux-2023
   ~~       V~' '->
    ~~~         /
      ~~._.   _/
         _/ _/
       _/m/'
[ec2-user@ip-10-0-1-177 ~]$
```

3. From Jump Host to MariaDB

ssh -i wp-key.pem ec2-user@DB_PRIVATE_IP

```
Downloads — ec2-user@ip-10-0-1-177:~ — ssh -i key-2.pem ec2-user@54.160.93.52 — 65×36
[ec2-user@ip-10-0-1-121 ~]$ ssh -i key-2.pem ec2-user@10.0.1.177
   ,     #_
   ~\_  ####_        Amazon Linux 2023
  ~~  \_#####\
  ~~     \###|
  ~~       \#/ ___   https://aws.amazon.com/linux/amazon-linux-20
23
   ~~       V~' '->
    ~~~         /
      ~~._.   _/
         _/ _/
       _/m/'
Last login: Wed Jan 21 16:14:56 2026 from 10.0.1.121
[ec2-user@ip-10-0-1-177 ~]$
```

STEP 9: MariaDB Installation (Private Subnet)

1. **Install Mariadb:** **#sudo dnf update -y && sudo dnf install mariadb105-server -y**
2. **Service:** **#sudo systemctl start mariadb && sudo systemctl enable mariadb**
3. **Secure:** **#sudo mariadb-secure-installation**
4. **Database Setup:**

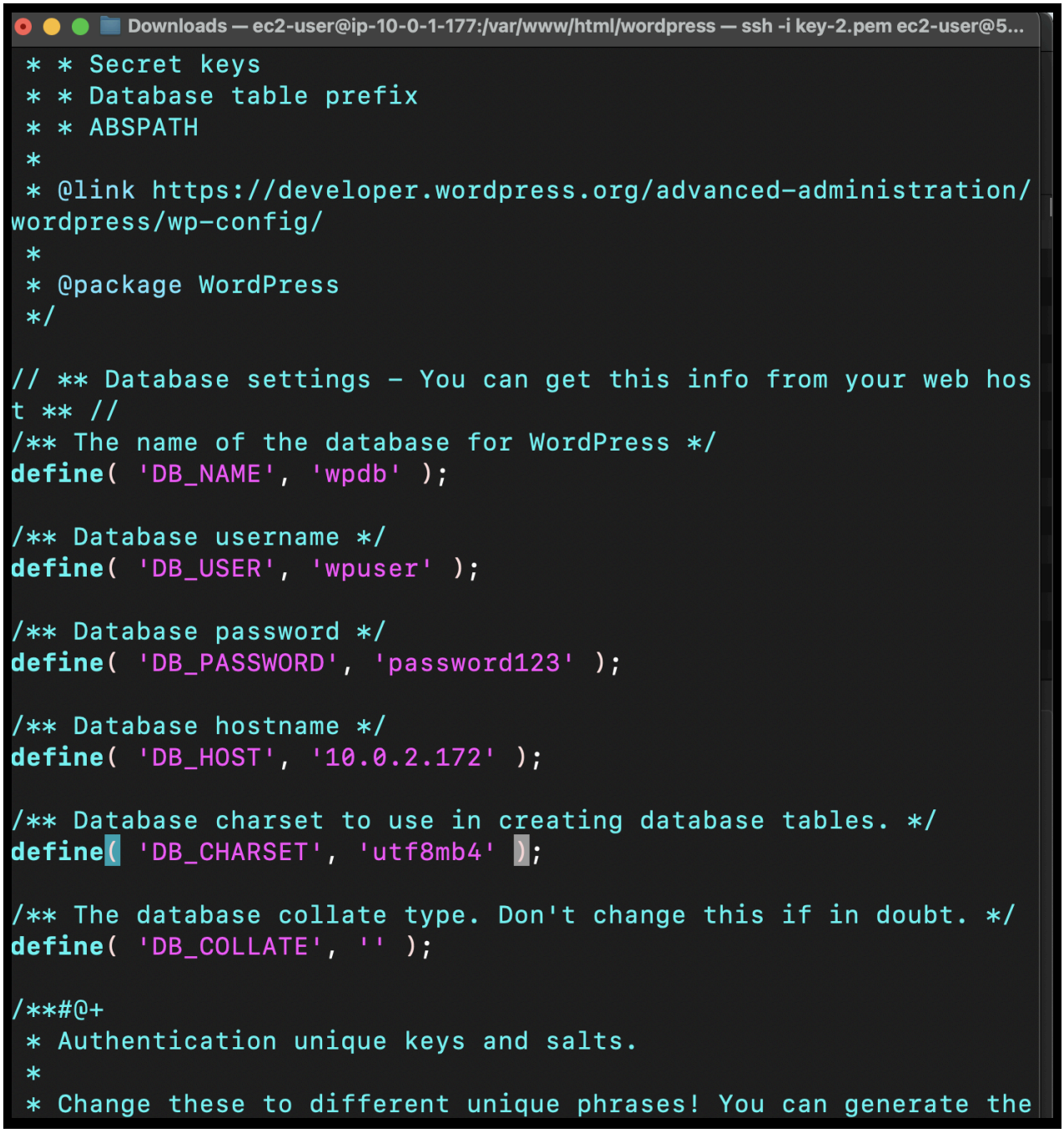
CREATE DATABASE wpdb;
CREATE USER 'wpuser'@'%' IDENTIFIED BY 'password123';
GRANT ALL PRIVILEGES ON wpdb.* TO 'wpuser'@'%';
FLUSH PRIVILEGES;

```
[ec2-user@ip-10-0-1-177 ~]$ sudo mariadb
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 13
Server version: 10.5.29-MariaDB MariaDB Server

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

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

MariaDB [(none)]> CREATE DATABASE wpdb;
Query OK, 1 row affected (0.000 sec)

MariaDB [(none)]> CREATE USER 'wpuser'@'%' IDENTIFIED BY 'password123';
Query OK, 0 rows affected (0.002 sec)

MariaDB [(none)]> GRANT ALL PRIVILEGES ON wpdb.* TO 'wpuser'@'%';
Query OK, 0 rows affected (0.001 sec)

MariaDB [(none)]> FLUSH PRIVILEGES;
Query OK, 0 rows affected (0.001 sec)

MariaDB [(none)]>
```

STEP 10: WordPress Installation

1. **Packages:** sudo dnf install httpd php php-mysqlnd wget unzip -y
2. **Download:**
   #cd /var/www/html
   #sudo wget https://wordpress.org/latest.zip
   #sudo unzip latest.zip
   #sudo chown -R apache:apache wordpress

3. **Config:**
   #cd wordpress
   #sudo cp wp-config-sample.php wp-config.php
   #sudo vi wp-config.php
   # Update DB_NAME, DB_USER, DB_PASSWORD, and DB_HOST (MariaDB Private IP)


Downloads — ec2-user@ip-10-0-1-177:/var/www/html/wordpress — ssh -i key-2.pem ec2-user@5...
* * Secret keys
* * Database table prefix
* * ABSPATH
*
* @link https://developer.wordpress.org/advanced-administration/wordpress/wp-config/
*
* @package WordPress
*/
// ** Database settings - You can get this info from your web host ** //
/** The name of the database for WordPress */
define('DB_NAME', 'wpdb');
/** Database username */
define('DB_USER', 'wpuser');
/** Database password */
define('DB_PASSWORD', 'password123');
/** Database hostname */
define('DB_HOST', '10.0.2.172');
/** Database charset to use in creating database tables. */
define('DB_CHARSET', 'utf8mb4');
/** The database collate type. Don't change this if in doubt. */
define('DB_COLLATE', '');
/**#@+
* Authentication unique keys and salts.
*
* Change these to different unique phrases! You can generate the

Phase 12: Final Access