***Log in to VM4 through VM2 :***

Ssh [sysadmin@10.0.4.4](mailto:sysadmin@10.0.4.4)

You will get log in to vm4

***### \*\*Step-by-Step Guide to Setting Up MySQL on a VM\*\*:***

**#### \*\*Step 1: SSH into Your Private Database VM\*\***

1. \*\*SSH from your Public VM\*\* into the \*\*Private VM (VM3)\*\* that will host MySQL.

```bash

ssh <username>@<private\_vm\_private\_ip>

```

**#### \*\*Step 2: Install MySQL on the Private VM\*\***

2. \*\*Update the Package List\*\*:

- Ensure your system’s package list is up to date.

```bash

sudo apt update

```

**3. \*\*Install MySQL Server\*\*:**

- Install MySQL using the following command:

```bash

sudo apt install mysql-server -y

```

**4**. **\*\*Start MySQL Service\*\*:**

- Once the installation is complete, start MySQL service.

```bash

sudo systemctl start mysql

**5. \*\*Secure MySQL Installation\*\*:**

- Run the MySQL security script to set up a root password, remove test databases, and more.

```bash

sudo mysql\_secure\_installation

```

- You’ll be prompted to set up a \*\*root password\*\* and make other security decisions (follow the prompts).

#### \*\*Step 3: Create a Database and User for Django\*\*

**6. \*\*Log into MySQL\*\*:**

- Access the MySQL shell.

```bash

sudo mysql -u root -p

```

- Enter the \*\*root password\*\* you set earlier.

**7. \*\*Create a Database for Your Django Project\*\*:**

```sql

CREATE DATABASE my\_django\_db;

```

**8. \*\*Create a New MySQL User\*\*:**

- Create a user and grant them privileges on the newly created database.

```sql

CREATE USER 'django\_user'@'%' IDENTIFIED BY 'Himanshu@2001';

GRANT ALL PRIVILEGES ON my\_django\_db.\* TO 'django\_user'@'%';

FLUSH PRIVILEGES;

**```**

**'%': The % wildcard represents "any host," meaning this user can connect to the MySQL server from any IP address or host.**

\*\*Step 4: Configure MySQL for Remote Access (Optional)\*\*

Since you’re running MySQL on a private VM, it’s secure, but you might still want to allow access only from trusted sources.

**9. \*\*Edit MySQL Configuration\*\*:**

- Open the MySQL configuration file.

```bash

**sudo vim /etc/mysql/mysql.conf.d/mysqld.cnf**

```

- Find the line that starts with `bind-address` and change it from `127.0.0.1` to `0.0.0.0` so MySQL listens on all interfaces:

```bash

**bind-address = 0.0.0.0**

```

- Save and exit the file.

Without specifying 0.0.0.0, by default, Django's development server listens only on localhost (127.0.0.1), which means the web application would only be accessible from the same machine and not from other machines in the network.

By specifying 0.0.0.0, you're saying, "**accept connections from any IP address to this machine," which makes the application accessible over the network or from other VMs.**

10. \*\*Restart MySQL\*\*:

```bash

sudo systemctl restart mysql

```

#### \*\*Step 5: Allow MySQL Traffic in NSG (Network Security Group)\*\*

11. \*\*Allow Access to MySQL from Your Backend VM\*\*:

- In Azure, ensure that your \*\*NSG\*\* (Network Security Group) allows the \*\*backend VM (VM2)\*\* to communicate with the \*\*MySQL port (3306)\*\* in the private subnet.

***……………………………………………………………………………………………………………………………………………………***

### \*\*Summary\*\*:

- \*\*Step 1\*\*: SSH into the \*\*private VM\*\*.

- \*\*Step 2\*\*: Install MySQL and secure it.

- \*\*Step 3\*\*: Create a database and user for your Django project.

- \*\*Step 4\*\*: Allow remote access and configure network settings (optional).

- \*\*Step 5\*\*: Allow traffic through \*\*NSG\*\*.

- \*\*Step 6\*\*: Update \*\*Django\*\* to use MySQL.

This setup ensures that your Django project is connected to MySQL on your \*\*private VM\*\*

***Connection between django and mysql:***

- \*\*VM2 (Public)\*\*: 13.70.65.137 (This will be the public-facing server)

- \*\*VM3 (Private)\*\*: 10.0.3.4 (Django application server)

- \*\*VM4 (Private)\*\*: 10.0.4.4 (MySQL database server)

### Next Steps:

1. \*\*Connect Django on VM3 to MySQL on VM4\*\*:

Since both VM3 and VM4 are private, they can communicate with each other using their private IPs. Ensure that your Django settings (`settings.py` file) in VM3 point to MySQL in VM4.

In `settings.py`, the database configuration should look like this:

```python

DATABASES = {

'default': {

'ENGINE': 'django.db.backends.mysql',

'NAME': 'my\_django\_db', # Replace with your database name

'USER': 'django\_user', # Replace with your database username

'PASSWORD': 'Himanshu@2001', # Replace with your database password

'HOST': '10.0.4.4', # Private IP of VM4

'PORT': '3306',

}

}

```

2. \*\*Accessing Django from VM2\*\*:

Since VM2 is public, you can use it as a jump host to access VM3. To access the Django application, you can SSH into \*\*VM2\*\* and then SSH into \*\*VM3\*\* using its private IP.

- SSH into VM2:

```bash

ssh <username>@13.70.65.137

```

- From VM2, SSH into VM3:

```bash

ssh <username>@10.0.3.4

```

3. \*\*Expose Django to the Internet (Optional)\*\*:

If you want to access the Django application directly from the public internet (via VM2), you could forward traffic from VM2 to VM3 using SSH tunneling or configure a reverse proxy on VM2 (e.g., using Nginx).