Complete Documentation of the Process

The complete process of deploying **Node.js microservice** using **three Ubuntu VirtualBox VMs** with **Docker, Nginx, and MySQL**.

**Architecture**

* **VM1: App Server 1** → Runs Node.js microservice (Docker)
* **VM2: App Server 2** → Runs Node.js microservice (Docker)
* **VM3: Database & Load Balancer** → Runs **MySQL** and **Nginx** (Docker)

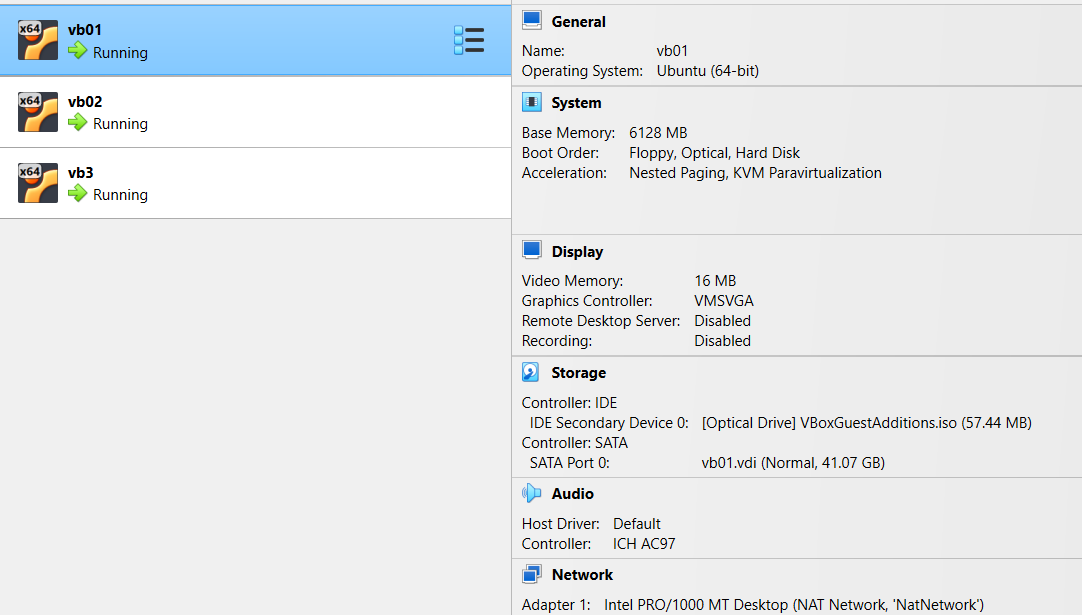
Download and installed the below mentioned VirtualBox.

VirtualBox-7.1.6-167084-Win

Created 3 VirtualBox VMs of Ubuntu 24.04 with the following specifications.

**Three VMs**

* Vb01: **App Server 1** (Node.js service)
* Vb02: **App Server 2** (Node.js service)
* Vb03: **Database & Load Balancer** (MySQL + Nginx)



3.created a NAT network and used the same for all 3 VMs.

A screenshot of a computer

AI-generated content may be incorrect.

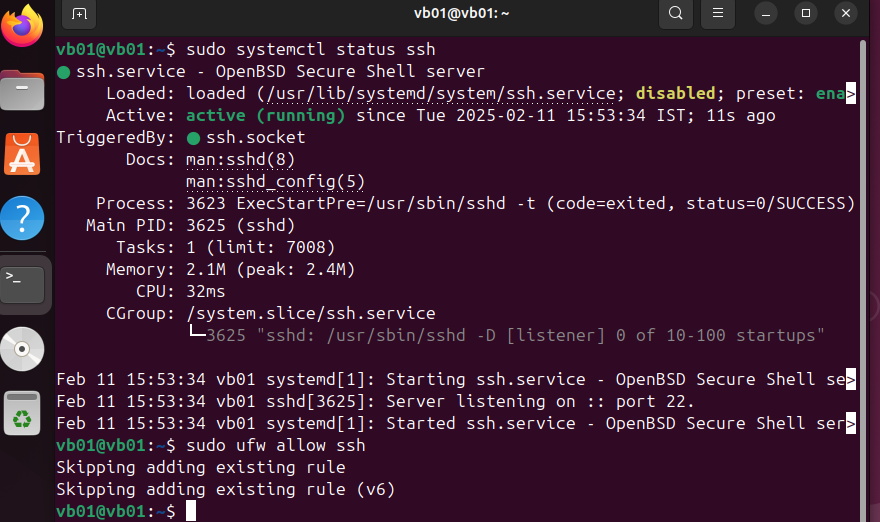
4.Install the SSH network protocol in all servers.

A screenshot of a computer

AI-generated content may be incorrect.

Sudo apt update

Sudo apt install openssh-server



Sudo systemctl status ssh

Sudo ufw allow ssh

**Update and Install Docker on All VMs**

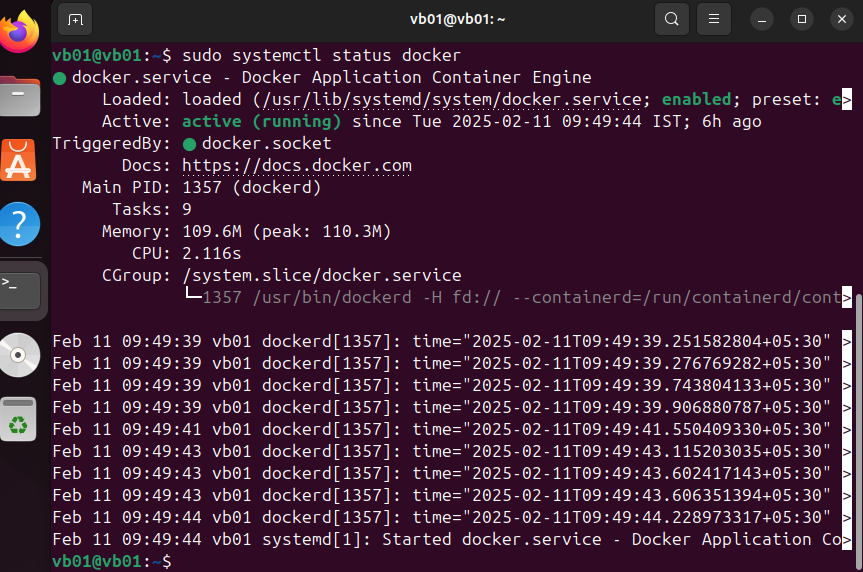
Run on all VMs:

sudo apt update && sudo apt upgrade -y

sudo apt install -y docker.io

sudo systemctl enable docker

sudo systemctl start docker



**Deploy MySQL Database (vb03)**

**1. Create a MySQL Docker Container**

Run on **vb03**:

docker run -d --name mysql-db \

-e MYSQL\_ROOT\_PASSWORD=root \

-e MYSQL\_DATABASE=microservice\_db \

-e MYSQL\_USER=microservice\_user \

-e MYSQL\_PASSWORD=microservice\_pass \

-p 3306:3306 \

mysql:latest

**2. Verify MySQL is Running**

Run:

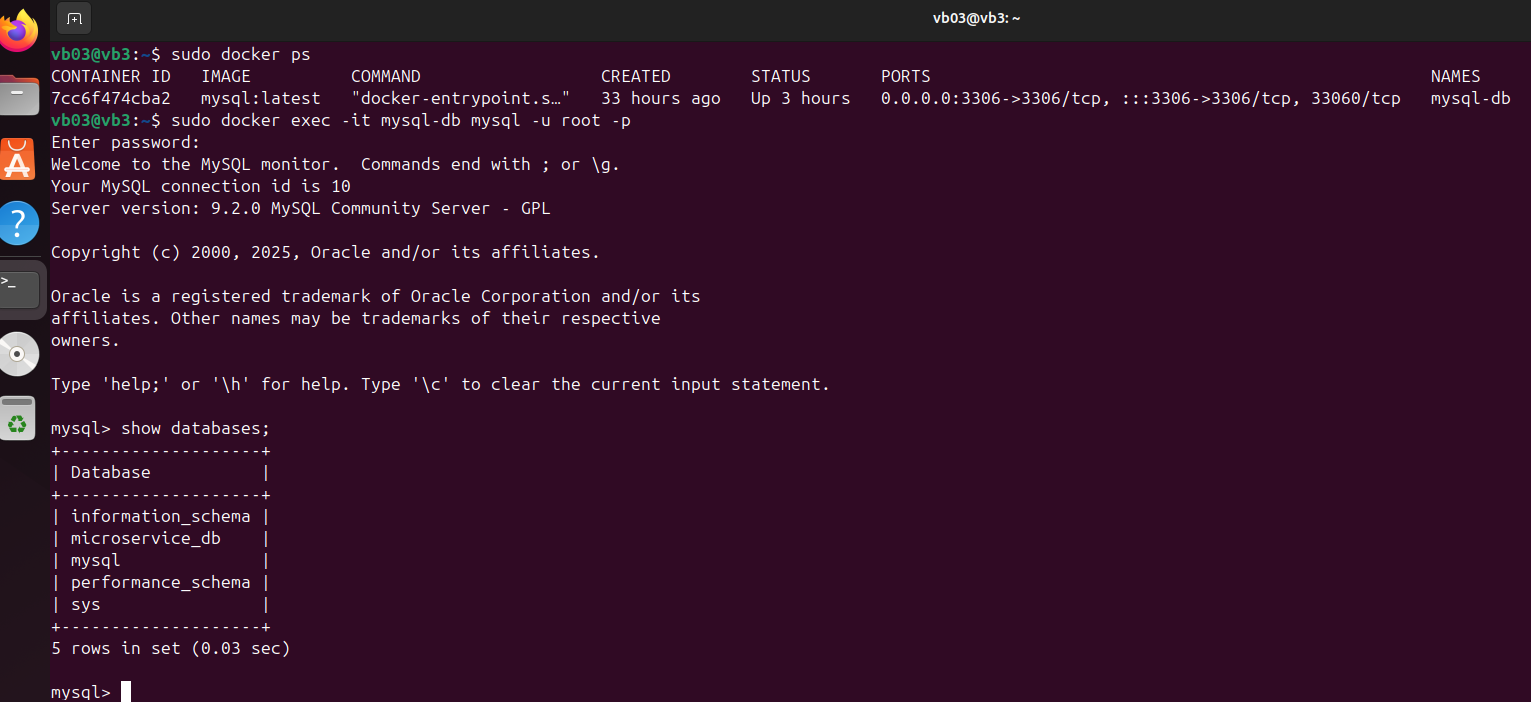
docker ps

Connect to MySQL:

docker exec -it mysql-db mysql -u root -p

Enter password (rootpassword) and check database:

SHOW DATABASES;



**Deploy Node.js Microservice (vb01 & vb02)**

**1. Install Node.js and MySQL Client**

Run on **vb01** and **vb02**:

sudo apt install -y mysql-client

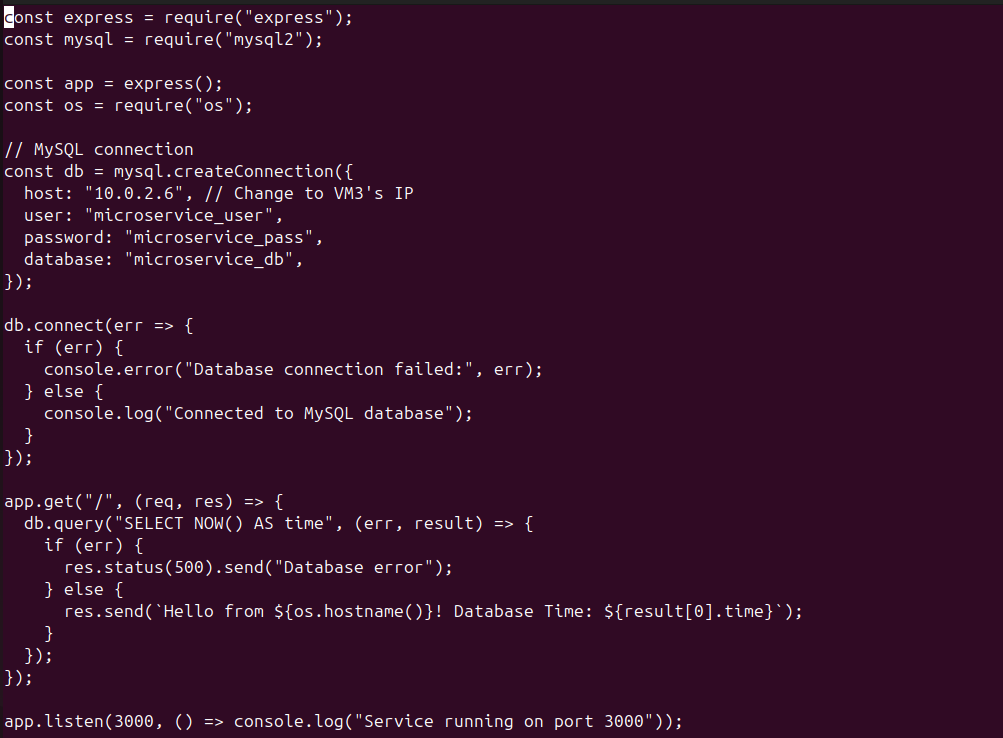
**2. Create Node.js App**

Run these commands on both **vb01** and **vb02**:

mkdir -p ~/microservice && cd ~/microservice

Create app.js:

Sudo nano app.js



**3. Create package.json**

Sudo nano package.json

A screenshot of a computer

AI-generated content may be incorrect.

**4. Create Dockerfile**

Create Dockerfile:

Sudo nano Dockerfile

A purple rectangular object with black border

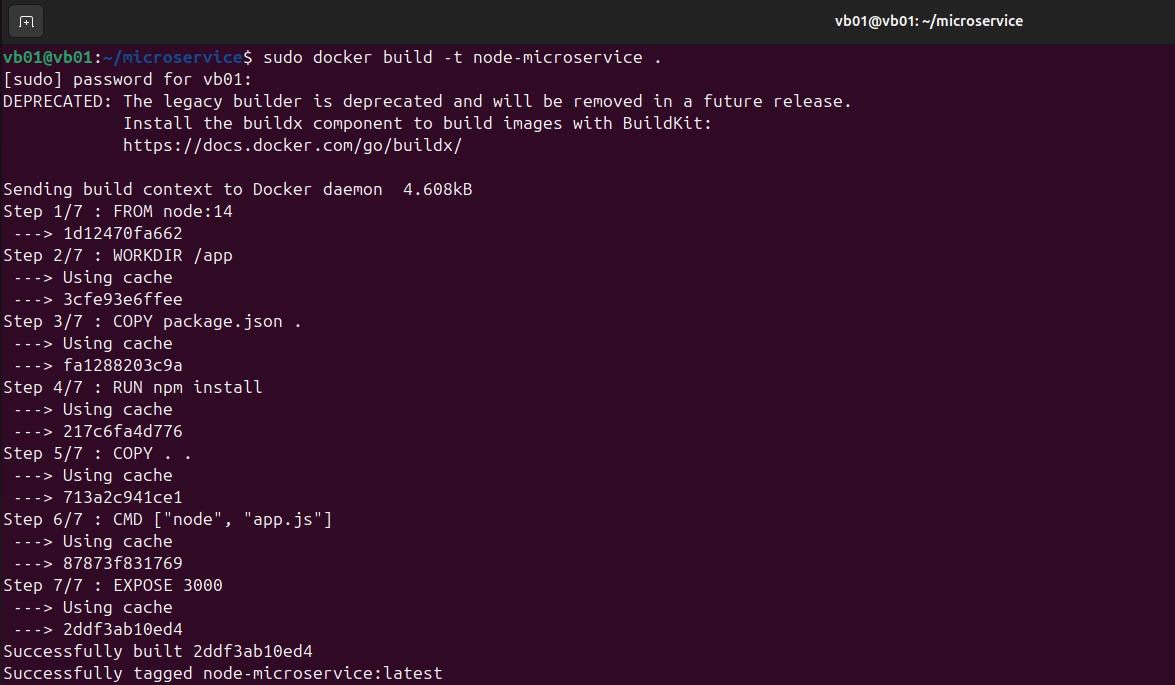
AI-generated content may be incorrect.

**5. Build & Run Docker Container**

Run on **vb01** and **vb02**:

docker build -t node-microservice .

docker run -d --name microservice -p 3000:3000 node-microservice

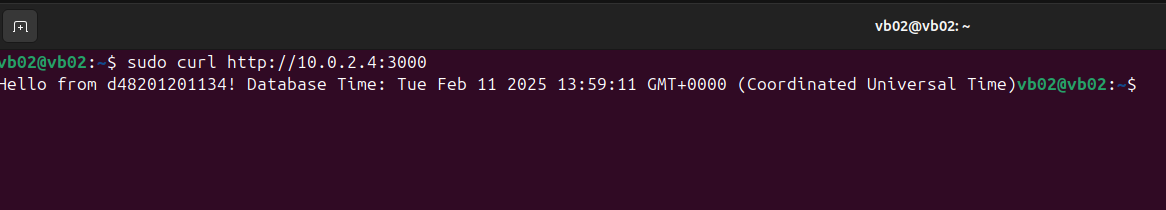


**6. Verify the Service**

ip a

Test the service from another VM:

curl <http://10.0.2.4:3000>



curl <http://10.0.2.5:3000>

A screenshot of a computer

AI-generated content may be incorrect.

**Set Up Nginx Load Balancer (vb03)**

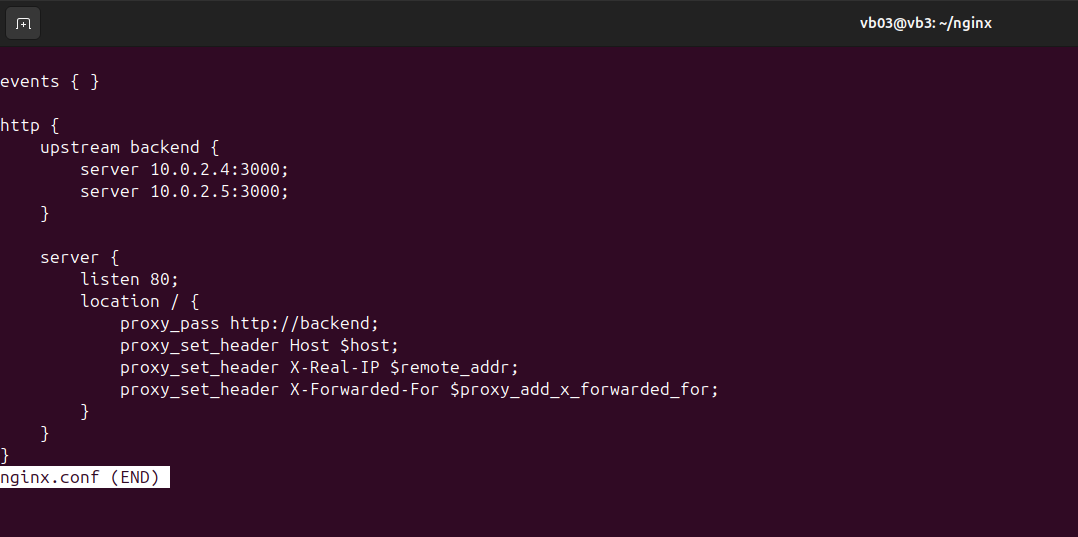
**1. Create Nginx Configuration File**

On **VM3**, create a directory for Nginx:

mkdir -p ~/nginx && cd ~/nginx

Create nginx.conf:

Sudo nano nginx.conf



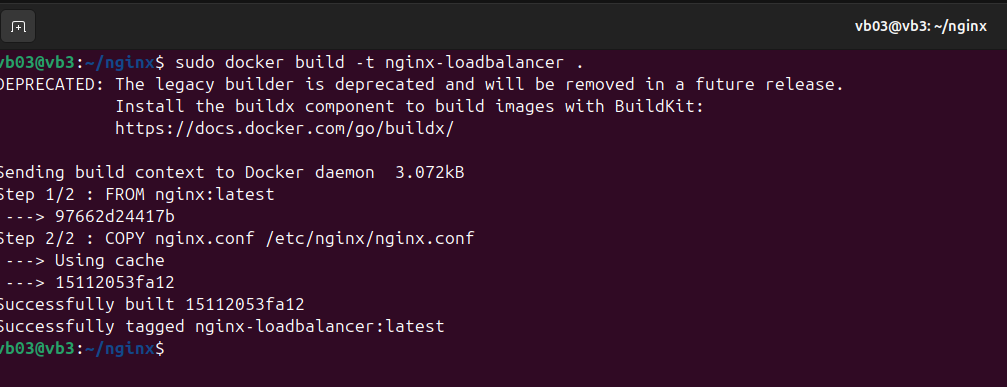
**2. Create Dockerfile for Nginx**

A black and purple rectangle

AI-generated content may be incorrect. **3. Build & Run Nginx in Docker**

docker build -t nginx-loadbalancer .

docker run -d --name nginx-lb -p 80:80 nginx-loadbalancer



**Step 5: Test the Load Balancer**

Find **vb03’s IP**:

ip a

Run:

curl http://10.0.2.6

