# Cloud Computing Assignment: VirtualBox-Based Microservice Deployment

## Objective

The objective of this assignment is to create and configure multiple Virtual Machines (VMs) using VirtualBox, establish a network between them, and deploy a simple Flask-based microservice-based application across the connected VMs to send data from one VM to another.

## Github link

<https://github.com/AbhilashAgarwalIITJ/VCC_Assignment1_VirtualBox_VM/tree/main>

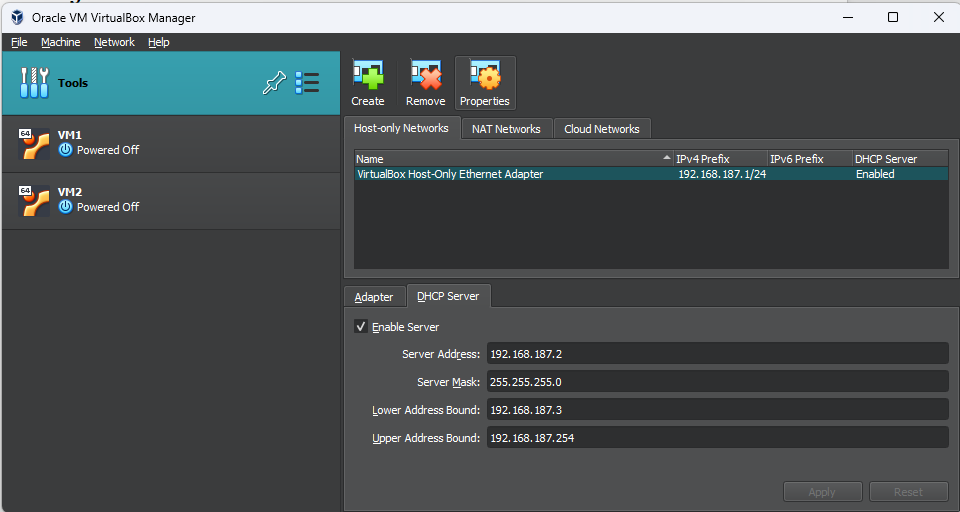
## Video link (google drive)

## Step-by-Step Implementation Guide

### 1. Installing VirtualBox and Creating Multiple VMs

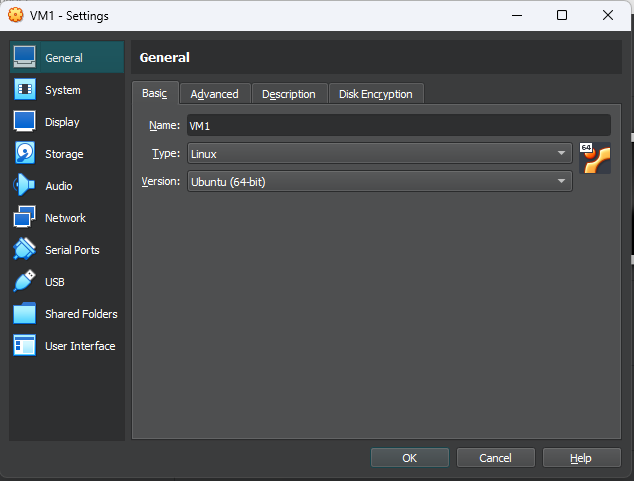
#### Installation of VirtualBox

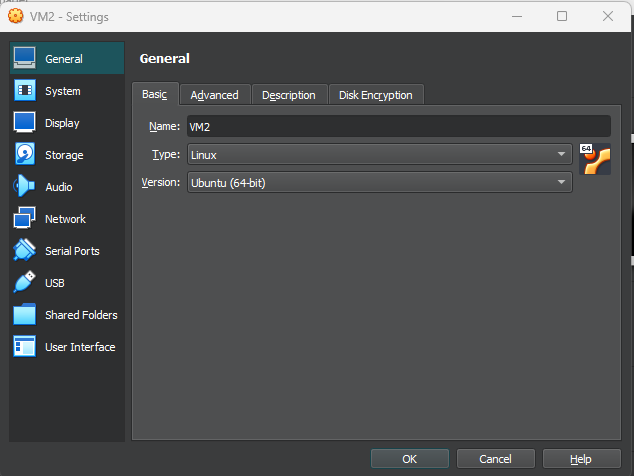
1. Downloaded VirtualBox from the official website: <https://www.virtualbox.org/>.
2. Installed VirtualBox following the installation steps for my Windows 11 operating system.



#### Creating Virtual Machines (VMs)

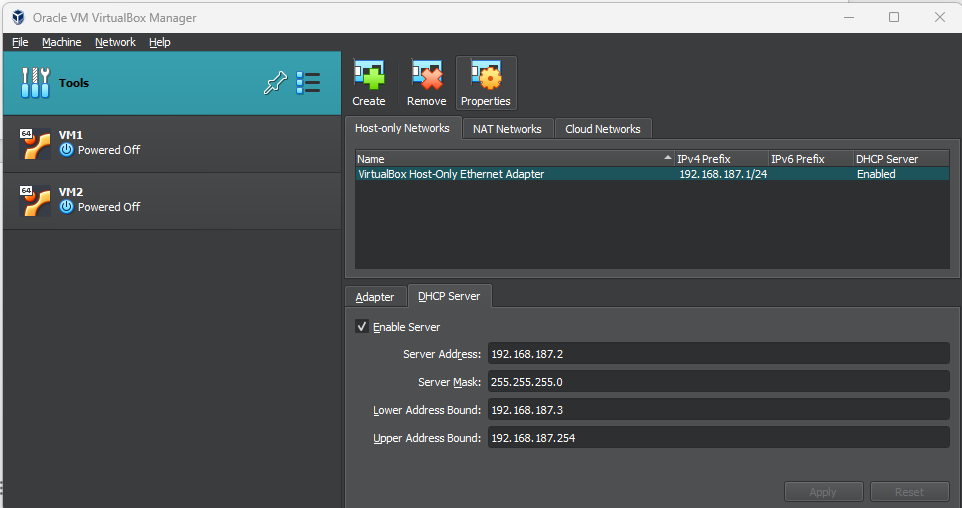
1. Opened VirtualBox and clicked **New** to create a new VM.
2. Created VM (VM1), selected **Linux (lubuntu-24.04.1-desktop-amd64.iso)** as the OS, and allocated 2GB RAM and 25GB storage, 2 core CPU.
3. Repeat the above steps to create a second VM (VM2).





### 2. Configuring Network Settings

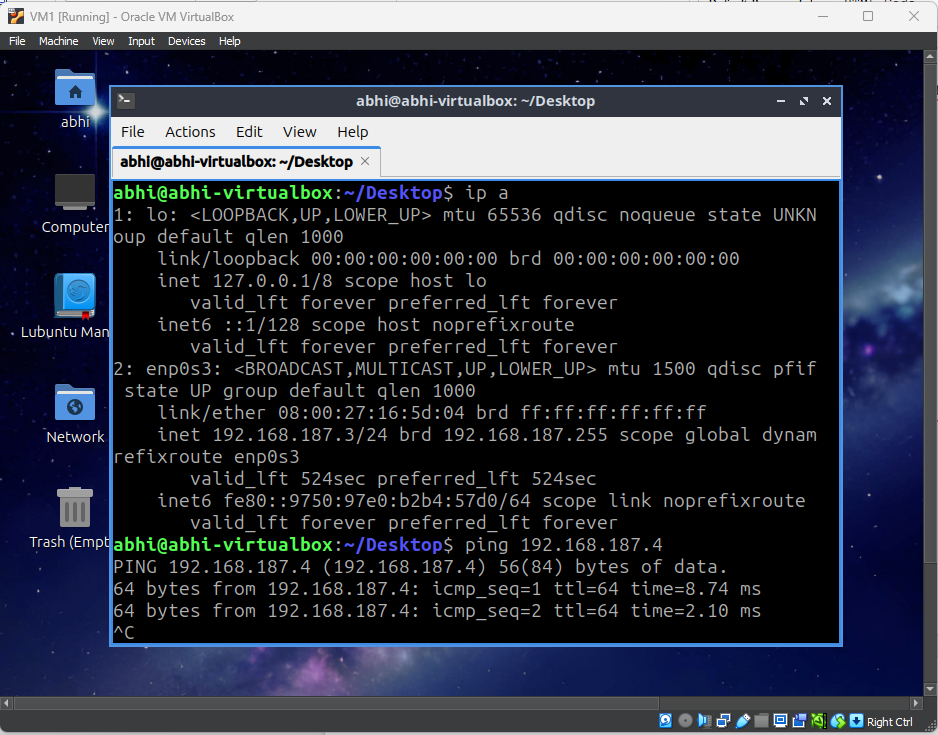
* Configured the **Network Adapter**:
  1. Go to **VirtualBox->Tools > Network**
  2. Selected **Host-only Networks** and clicked on Create to create a new Host-only Networks - **VirtualBox Host-only Ethernet Adaptor**
  3. Clicked on DHCP tab and selected the Enable Server check box

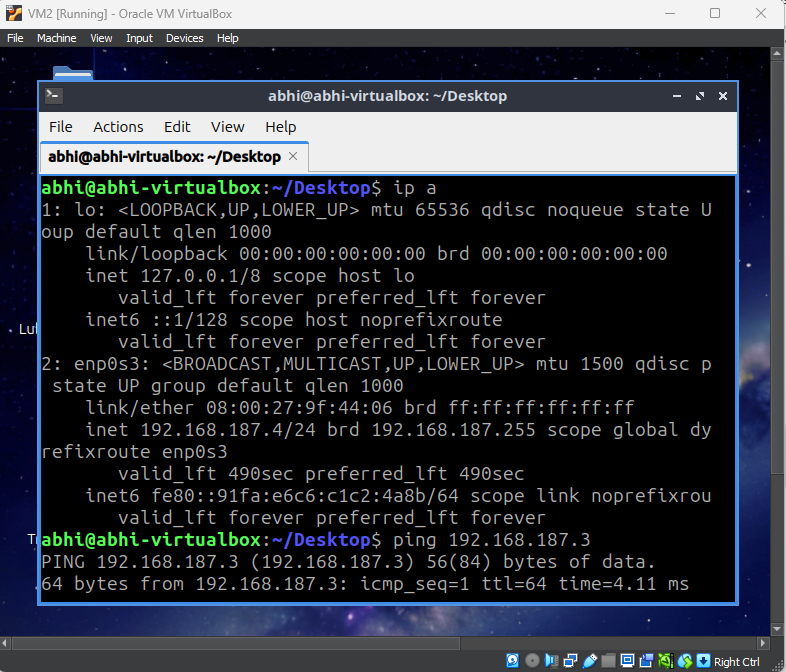


* Repeat steps for both VM1 and VM2
  + Start VM1 and log in.
  + Go to Setting->Network->Adaptor1
  + Select Attached to - > Host-Only Adaptor and pick **VirtualBox Host-only Ethernet Adaptor**
* Got IP address of of VM1 - 192.168.187.3
* Got IP address of of VM2 - 192.168.187.4
* Test connectivity between VMs:

ping 192.168.187.4 # From VM1

ping 192.168.187.3 # From VM2





### 3. Deploying a Simple Flask-Based Microservice Application

#### 3.1. Enable Internet on VM1 and VM2

* + Go to Setting->Network->Adaptor1
  + Select Attached to - >NAT

#### 3.2 Setting up Flask on VM2 (Server)

1. Install Python and Flask:

sudo apt update

sudo apt install python3 python3-pip -y

pip3 install flask

1. Created a flask app - app.py:

*from flask import Flask, jsonify*

*app = Flask(\_\_name\_\_)*

*@app.route('/message', methods=['GET'])*

*def send\_message():*

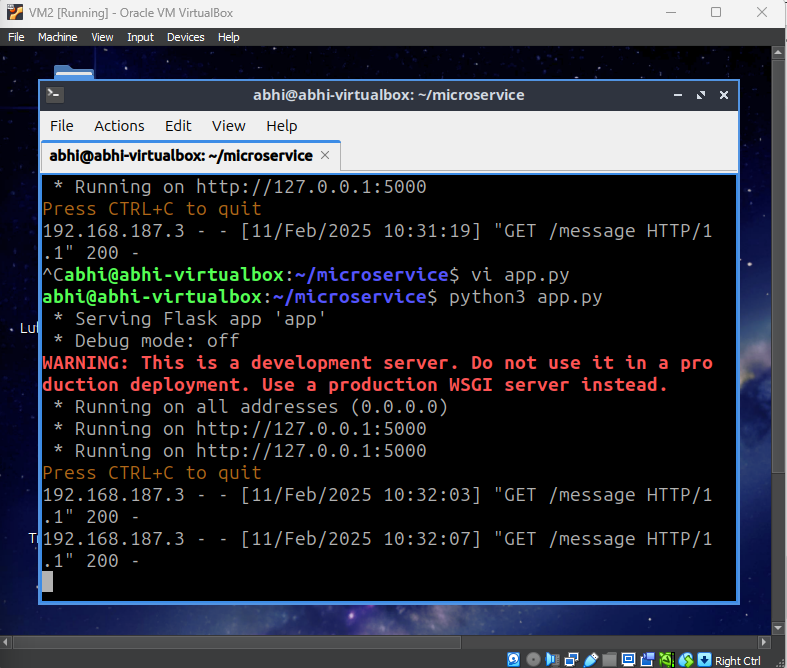
*return jsonify({"message": "Hello from VM2!"})*

*if \_\_name\_\_ == '\_\_main\_\_':*

*app.run(host='0.0.0.0', port=5000)*

1. Go to Setting->Network->Adaptor1
2. Attached to - > Host-Only Adaptor and pick **VirtualBox Host-only Ethernet Adaptor**
3. Run this flask app:

python3 app.py



#### 3.3. Setting up a Client on VM1

1. Install curl and requests:

sudo apt install curl -y

pip3 install requests

1. Created a client script (client.py) to request data from VM2:

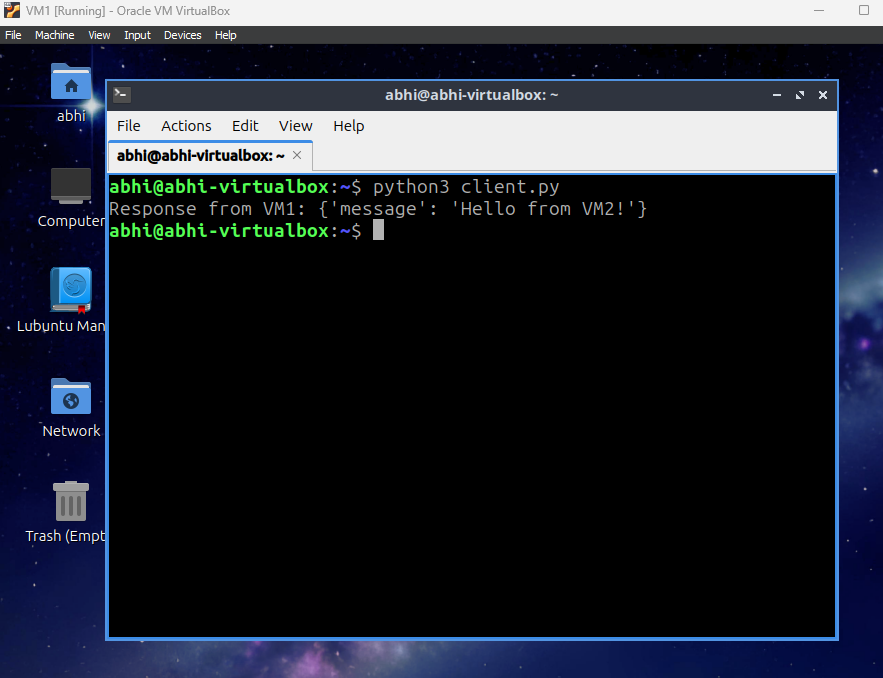
*import requests*

*response = requests.get("http://192.168.187.3:5000/message")*

*print("Response from VM1:", response.json())*

1. Go to Setting->Network->Adaptor1
2. Attached to - > Host-Only Adaptor and pick **VirtualBox Host-only Ethernet Adaptor**
3. Run the client script:

python3 client.py



### 4. Testing and Verification

1. Run the Flask server on VM2.
2. Run the client script on VM1.
3. Client running on VM1 received the message from the server VM2.

### Architecture Diagram

VM1

IP : 192.168.187.3

Python client calling message REST API hosted on VM1

VM2

IP : 192.168.187.4

Flask Server running on 5000 port, exposed message REST API

### 

### 6. Conclusion

This assignment successfully demonstrates the creation and configuration of multiple Virtual Machines using VirtualBox, the establishment of an internal network between the VMs, and the deployment of a microservice-based Flask application. The communication between VMs is tested by sending a message from VM2 to VM1.