VIRTUAL MACHINE SETUP AND MICROSERVICE DEPLOYMENT

Using Oracle VirtualBox and FastAPI

Prepared By

Anshul Kumar

Course: CSL 7510 VCC

Indian Institute of Technology, Jodhpur

Date: February 1, 2026

1. Introduction

This document provides a comprehensive guide for creating and configuring multiple Virtual Machines (VMs) using Oracle VirtualBox, establishing network connectivity between them, and deploying a microservice-based application across the connected VMs.

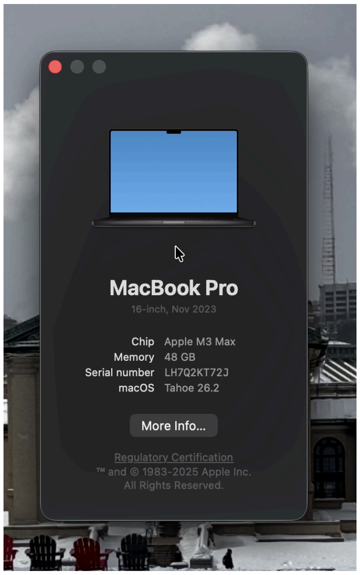
The project demonstrates a simple three-tier microservice architecture where requests flow through an API Gateway, get validated by a Validator service, and receive a response from a Greeter service. This showcases fundamental concepts of distributed systems and inter-service communication.

1.1 Objectives

- Install and configure Oracle VirtualBox on the host system
- Create three Ubuntu Server virtual machines
- Configure NAT Network for inter-VM communication
- Deploy and test a distributed microservice application

1.2 Host System Specifications

Component	Specification
Device	MacBook Pro 16-inch (Nov 2023)
Processor	Apple M3 Max
Memory	48 GB
Operating System	macOS Tahoe 26.2
VirtualBox Version	7.2.6 r172322

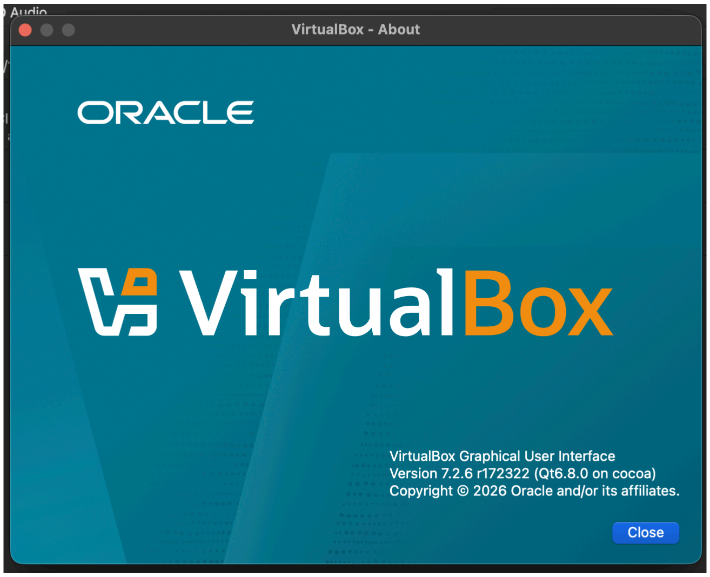


MacBook Pro

16-inch, Nov 2023

Chip Apple M3 Max
Memory 48 GB
Serial number LH7Q2KT72J
macOS Tahoe 26.2

Regulatory Certification
™ and © 1983-2025 Apple Inc.
All Rights Reserved.

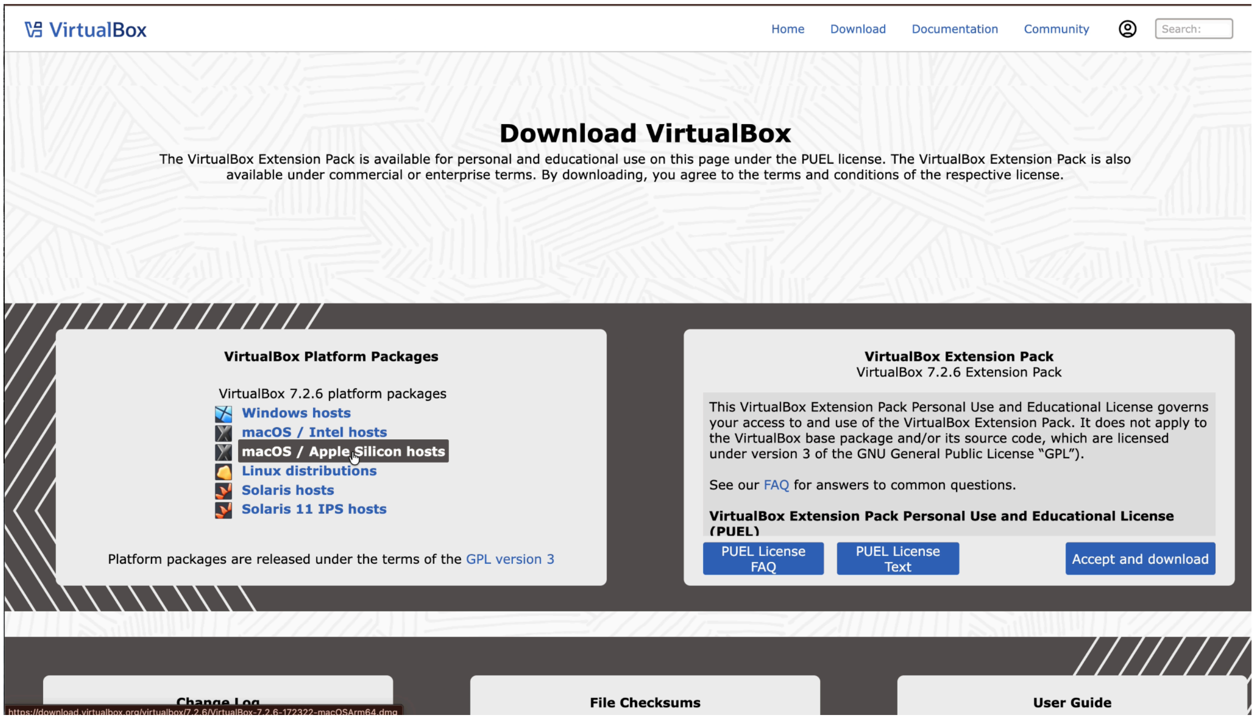


VirtualBox Graphical User Interface
Version 7.2.6 r172322 (Qt6.8.0 on cocoa)
Copyright © 2026 Oracle and/or its affiliates.

2. VirtualBox Installation

2.1 Download and Installation Steps

1. Navigate to the official VirtualBox website: https://www.virtualbox.org/wiki/Downloads
2. Select the appropriate platform package (macOS hosts for Apple Silicon)
3. Download the installer (approximately 233 MB)
4. Open the downloaded DMG file and run the installer
5. Follow the installation wizard with default settings
6. Grant necessary permissions in System Preferences > Security & Privacy
7. Launch VirtualBox to verify successful installation



Download VirtualBox

The VirtualBox Extension Pack is available for personal and educational use on this page under the PUEL license. The VirtualBox Extension Pack is also available under commercial or enterprise terms. By downloading, you agree to the terms and conditions of the respective license.

VirtualBox Platform Packages

VirtualBox 7.2.6 platform packages

- Windows hosts
- macOS / Intel hosts
- macOS / Apple Silicon hosts
- Linux distributions
- Solaris hosts
- Solaris 11 IPS hosts

Platform packages are released under the terms of the GPL version 3

VirtualBox Extension Pack
VirtualBox 7.2.6 Extension Pack

This VirtualBox Extension Pack Personal Use and Educational License governs your access to and use of the VirtualBox Extension Pack. It does not apply to the VirtualBox base package and/or its source code, which are licensed under version 3 of the GNU General Public License "GPL").

See our FAQ for answers to common questions.

VirtualBox Extension Pack Personal Use and Educational License (PUEL)

https://download.virtualbox.org/virtualbox/7.2.6/VirtualBox-7.2.6-172322-macOSArm64.dmg

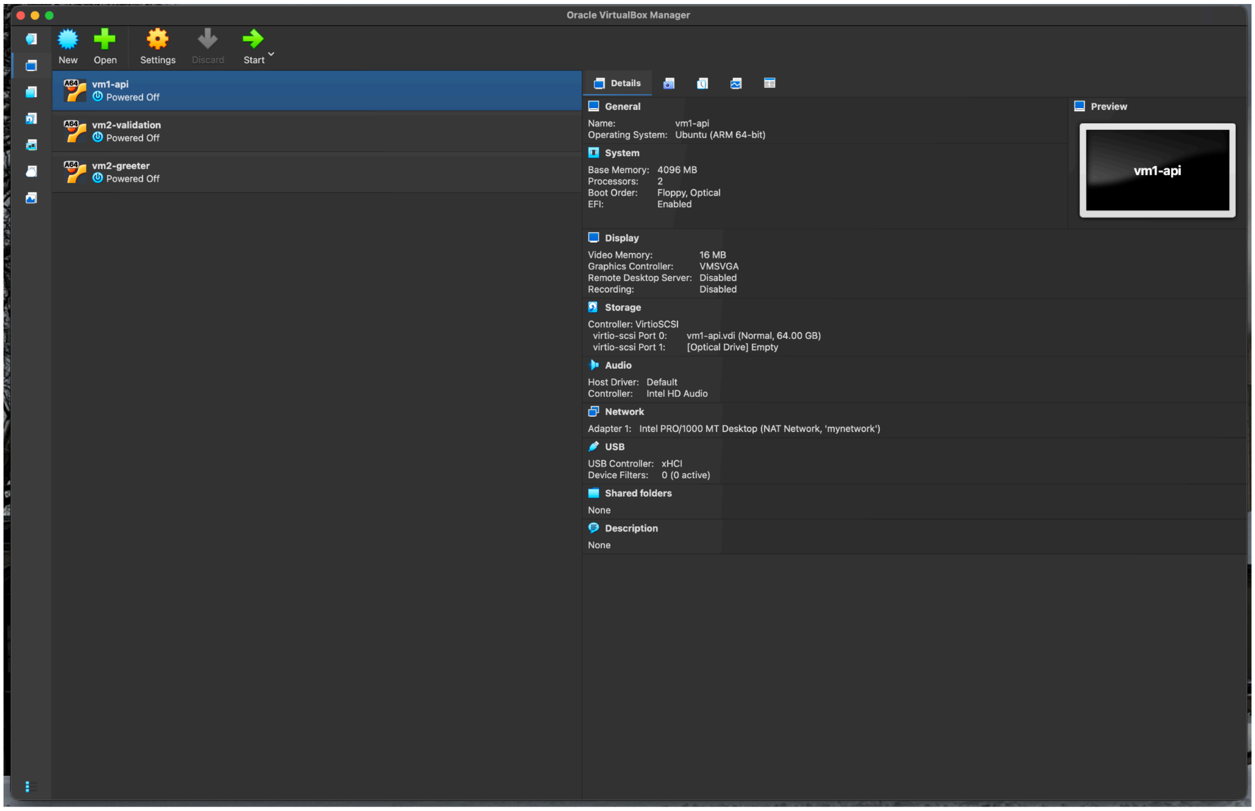
3. Virtual Machine Creation

3.1 VM Specifications

Three Ubuntu Server 24.04.3 LTS virtual machines were created with the following specifications:

VM Name	Hostname	Role	Port
VM1-API-Server	vm1-api	API Gateway	8001

VM2-Validator	vm2-validator	Validator Service	8002
VM3-Greeter	vm2-greeter	Greeter Service	8003



vm1-api
Powered Off

vm2-validation
Powered Off

vm2-greeter
Powered Off

General
Name: vm1-api
Operating System: Ubuntu (ARM 64-bit)

System
Base Memory: 4096 MB
Processors: 2
Boot Order: Floppy, Optical
EFI: Enabled

Display
Video Memory: 16 MB
Graphics Controller: VMSVGA
Remote Desktop Server: Disabled
Recording: Disabled

Storage
Controller: VirtioSCSI
virtio-scsi Port 0: vm1-api.vdi (Normal, 64.00 GB)
virtio-scsi Port 1: [Optical Drive] Empty

Audio
Host Driver: Default
Controller: Intel HD Audio

Network
Adapter 1: Intel PRO/1000 MT Desktop (NAT Network, 'mynetwork')

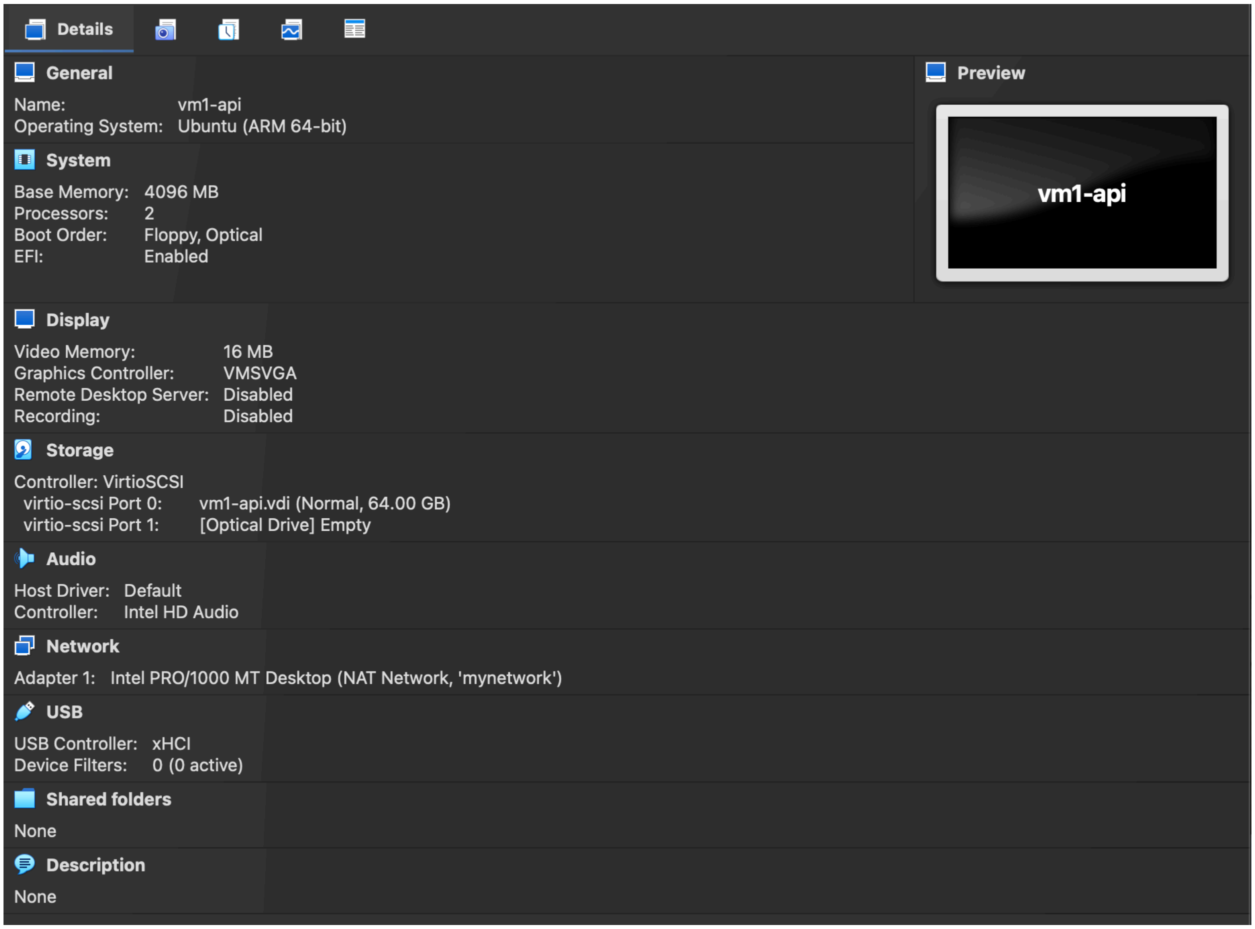
USB
USB Controller: xHCI
Device Filters: 0 (0 active)

Shared folders
None

Description
None

3.2 Hardware Configuration (Per VM)

Resource	Allocation
RAM	4096 MB (4 GB)
Processors	2 CPU Cores
Storage	64 GB (Dynamically Allocated VDI)
Network	NAT Network



General
Name: vm1-api
Operating System: Ubuntu (ARM 64-bit)

System
Base Memory: 4096 MB
Processors: 2
Boot Order: Floppy, Optical
EFI: Enabled

Display
Video Memory: 16 MB
Graphics Controller: VMSVGA
Remote Desktop Server: Disabled
Recording: Disabled

Storage
Controller: VirtioSCSI
virtio-scsi Port 0: vm1-api.vdi (Normal, 64.00 GB)
virtio-scsi Port 1: [Optical Drive] Empty

Audio
Host Driver: Default
Controller: Intel HD Audio

Network
Adapter 1: Intel PRO/1000 MT Desktop (NAT Network, 'mynetwork')

USB
USB Controller: xHCI
Device Filters: 0 (0 active)

Shared folders
None

Description
None

3.3 Ubuntu Server Installation Process

8. Click 'New' in VirtualBox Manager and enter VM name
9. Select Ubuntu Server 24.04.3 ISO image, Type: Linux, Version: Ubuntu (64-bit)
10. Enable 'Skip Unattended Installation' option
11. Allocate RAM (4096 MB) and CPU cores (2)
12. Create virtual hard disk (64 GB dynamically allocated)
13. Start VM and complete Ubuntu Server installation
14. Create user account (username: vmuser) and install OpenSSH server
15. Repeat process for all three VMs

Video Link : Assignmnet 1 VCC.mov

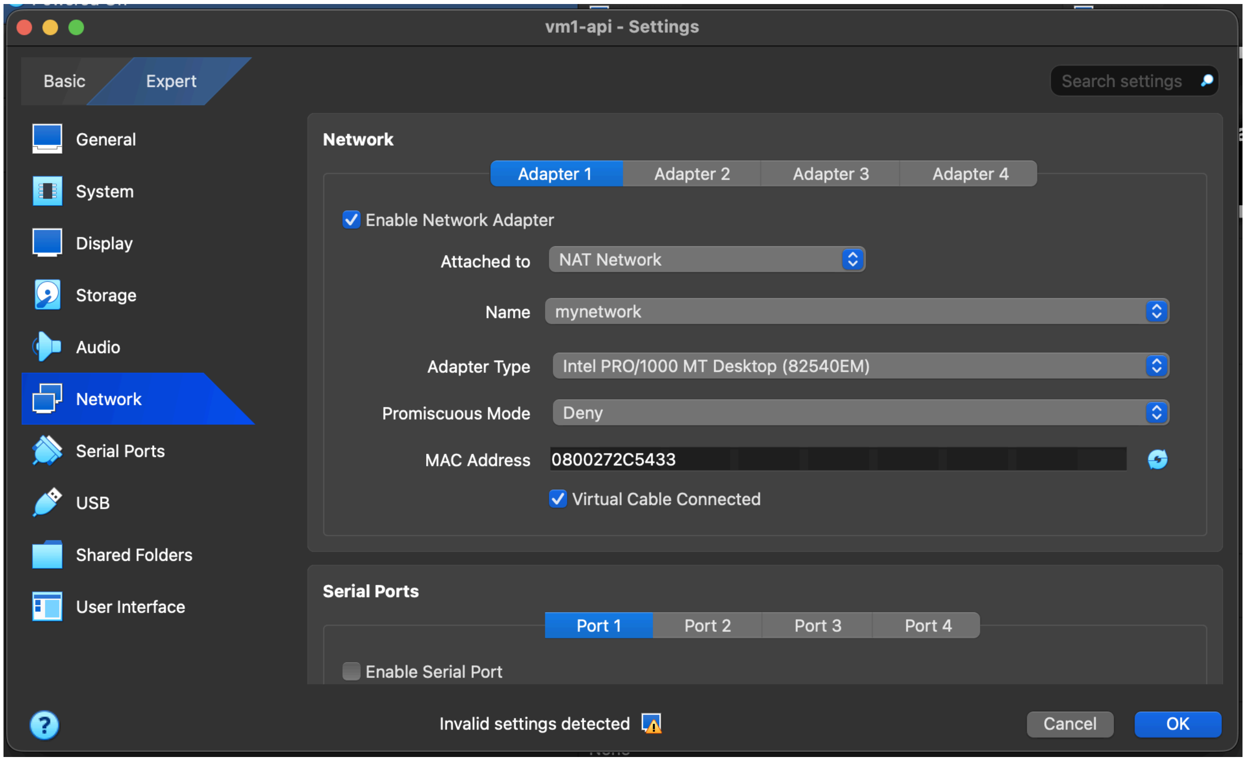
4. Network Configuration

4.1 Network Challenge

The default NAT configuration in VirtualBox isolates each VM in separate networks, preventing inter-VM communication required for microservices. To solve this, a NAT Network was created to enable VM communication while maintaining internet access.

4.2 NAT Network Setup

16. Navigate to File > Tools > Network Manager > NAT Networks tab
17. Click 'Create' to add a new NAT Network
18. Configure: Name: mynetwork, IPv4 Prefix: 10.0.2.0/24, DHCP: Enabled
19. For each VM: Right-click > Settings > Network
20. Change 'Attached to' from NAT to NAT Network and select 'mynetwork'



vm1-api - Settings

Network

☑ Enable Network Adapter

Attached to: NAT Network

Name: mynetwork

Adapter Type: Intel PRO/1000 MT Desktop (82540EM)

Promiscuous Mode: Deny

MAC Address: 0800272C5433

☑ Virtual Cable Connected

Serial Ports

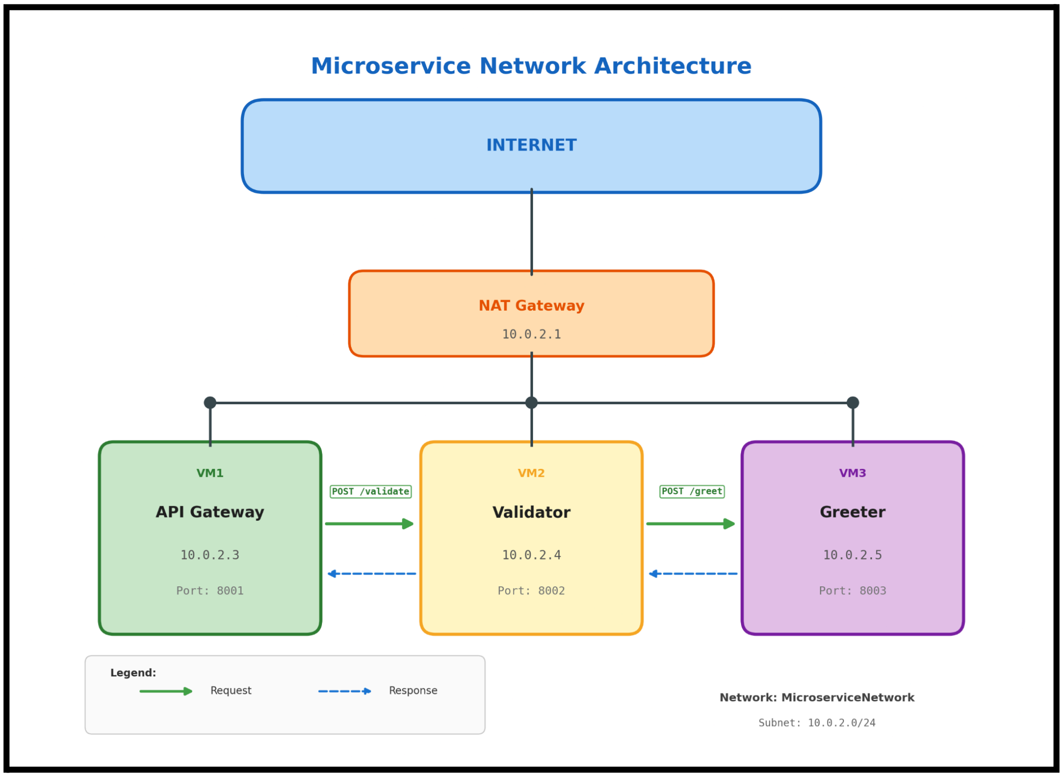
□ Enable Serial Port

Invalid settings detected

4.3 IP Address Assignment

Virtual Machine	IP Address	Port
VM1 (API Gateway)	10.0.2.3	8001
VM2 (Validator)	10.0.2.4	8002
VM3 (Greeter)	10.0.2.5	8003

4.4 Network Architecture Diagram



5. Microservice Application Deployment

5.1 Dependencies Installation

Run the following commands on each VM to install required packages:

```
sudo apt update
Sudo apt install uvicorn
sudo apt install python3 python3-pip -y
pip3 install fastapi uvicorn requests
```

5.2 Service Code

VM1 - API Gateway (gateway.py):

```
from fastapi import FastAPI
import requests
app = FastAPI()
VM2_URL = "http://10.0.2.4:8002"
@app.post("/ping")
def ping(data: dict):
    response = requests.post(f"{VM2_URL}/validate", json=data)
```

VM2 - Validator (validator.py): Validates input and forwards to VM3

VM3 - Greeter (greeter.py): Returns formatted greeting response

(Full source code available in GitHub repository - see Section 7)

5.3 Running the Services

VM	Command
VM1	uvicorn gateway:app --host 0.0.0.0 --port 8001
VM2	uvicorn validator:app --host 0.0.0.0 --port 8002
VM3	uvicorn greeter:app --host 0.0.0.0 --port 8003

6. Testing and Verification

6.1 Success Test

```
curl -X POST http://10.0.2.3:8001/ping \
  -H "Content-Type: application/json" \
  -d '{"name": "Anshul"}'
```

Expected Result: Success message with greeting from VM3

6.2 Failure Test (Validation Error)

```
curl -X POST http://10.0.2.3:8001/ping \
  -H "Content-Type: application/json" \
  -d '{"name": ""}'
```

Expected Result: Validation error from VM2

7. Project Deliverables

Deliverable	Location / Link
Documentation	This document
Architecture Diagram	Section 4.4 of this document
Source Code Repository	https://github.com/anshulk-cmu/assignmnet1-vcc
Video Demonstration	🎬 Assignmnet 1 VCC.mov

8. Conclusion

This project successfully demonstrated the creation and configuration of multiple virtual machines using Oracle VirtualBox, establishment of network connectivity through NAT Network configuration, and deployment of a distributed microservice application using FastAPI and Python.

The three-tier architecture (API Gateway > Validator > Greeter) showcases fundamental concepts of microservice communication, request routing, and distributed systems. The implementation provides a foundation for understanding more complex containerized and orchestrated deployments.

8.1 Key Learnings

- VirtualBox NAT Network enables inter-VM communication while maintaining internet access
- FastAPI provides an efficient framework for building RESTful microservices
- Proper JSON handling is essential for inter-service communication
- Service discovery and IP configuration are critical in distributed systems