# EXPERIMENT 1

**Name: Madhav Madan**

**SapID: 500105699**

**Roll No: R2142220265**

**Lab Exercise: Introduction to Vagrant and Vagrantfile**

This exercise will guide them through setting up a virtual environment using Vagrant, configuring the environment via a Vagrantfile, and managing the virtual machines (VMs) with basic Vagrant commands.

## Objective:

- Learn how to set up and configure virtual environments using Vagrant.

- Understand the structure and components of a Vagrantfile.

- Gain hands-on experience in managing virtual machines using Vagrant commands.

## Prerequisites:

- Basic knowledge of virtualization concepts.

- Familiarity with command-line interfaces.

- Installation of Vagrant and VirtualBox (or any other supported provider) on your local machine.
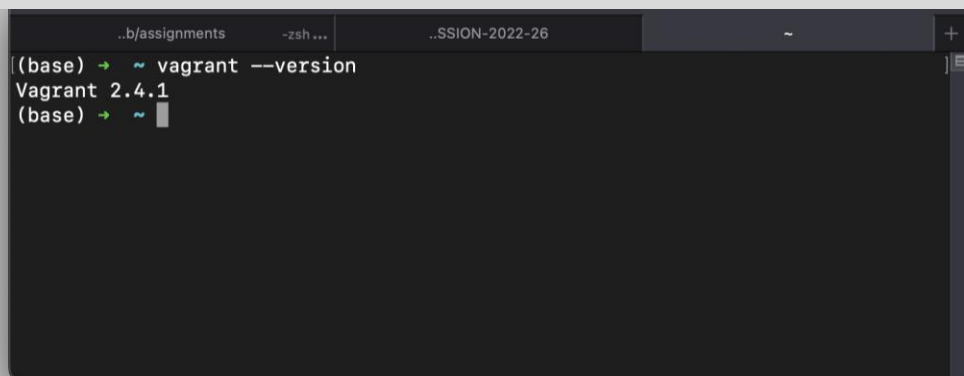
**Step-by-Step Exercise:**

**1. Setting Up the Environment:**

**Install Vagrant:**

- Download and install Vagrant from the official website.

- Ensure you have VirtualBox installed as it is a commonly used provider with Vagrant.

- Verify Installation:

- Open a terminal or command prompt.

- Run the following commands to verify the installation:

```
vagrant --version
```

## 2. Creating a New Vagrant Project:
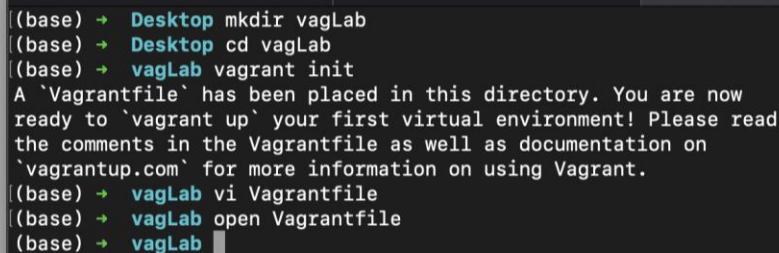
- Create a Project Directory:

- In your terminal, create a new directory for your Vagrant project and navigate into it:

```
mkdir vagrant_lab
cd vagrant_lab
```

## Initialize Vagrant:

- Run the following command to initialize a new Vagrantfile in your project directory:

```
vagrant init
```



This command will generate a Vagrantfile in the current directory.

## 3. Understanding the Vagrantfile:

- Open the Vagrantfile:

- Open the Vagrantfile in a text editor of your choice.

- The Vagrantfile is a Ruby-based configuration file used to define the virtual environment.

- Basic Vagrantfile Configuration:

- Modify the Vagrantfile to configure a basic virtual machine. For example:

```
Vagrant.configure("2") do |config|
  config.vm.box = "ubuntu/bionic64"  # Specifies the base box to use (Ubuntu 18.04)
  config.vm.network "private_network", type: "dhcp"  # Configures a private network
  config.vm.provider "virtualbox" do |vb|
    vb.memory = "1024"  # Allocates 1GB of RAM to the VM
  end
end
```
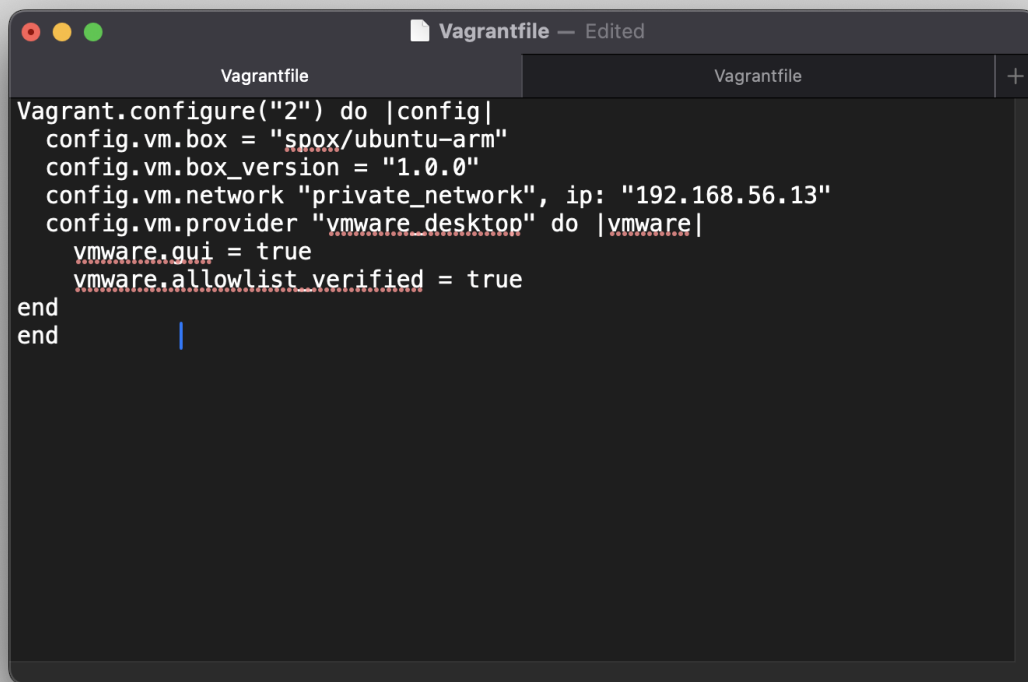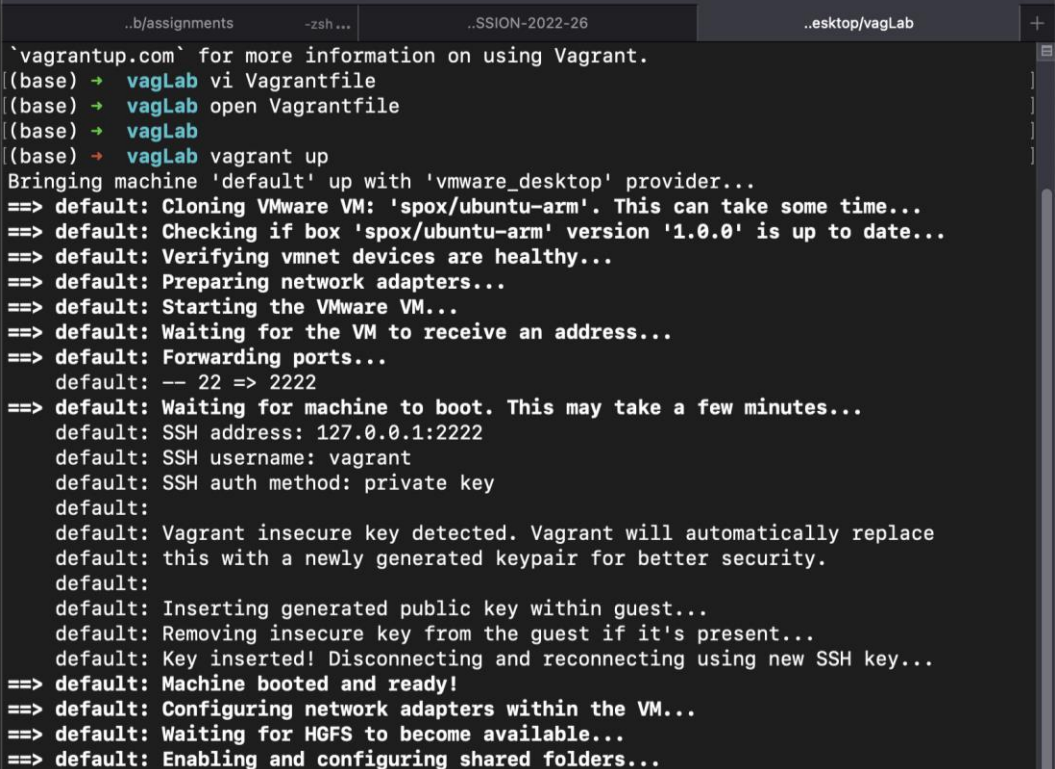
**4. Launching and Managing the VM:**

Start the VM:

In the terminal, start the VM using the following command:

```
vagrant up
```



Vagrant will download the specified box (if not already downloaded) and launch the VM.

- SSH into the VM:

- Connect to the running VM using SSH:

```
vagrant ssh
```

- This command will log you into the VM's shell.

- Exploring the VM:

- Inside the VM, explore the filesystem, install packages, and run commands to understand the environment.

- Stop the VM:

- Exit the SSH session by typing exit.

- Stop the VM with the following command:

```
vagrant halt
[vagrant@vagrant:~$ exit
logout
(base) → vagLab vagrant halt
==> default: Attempting graceful shutdown of VM...
==> default: Stopping the VMware VM...
(base) → vagLab
```

Destroy the VM (optional):

To remove the VM completely, use the following command:

```
vagrant destroy
```

```
vagrant@vagrant:~$ exit
logout
(base) → vagLab vagrant destroy
    default: Are you sure you want to destroy the 'default' VM? [y/N] y
==> default: Stopping the VMware VM...
==> default: Deleting the VM...
(base) → vagLab
```

This will remove all traces of the VM, including any data stored on it.

Explore the benefits of using Vagrant for development and testing environments.