Environment Orchestration with Vagrant

1. What is Vagrant?

Vagrant is an open-source tool used to **build and maintain portable virtual development environments**.

Think of it as a "Wrapper" or a "Remote Control" for Virtualization. Instead of manually clicking buttons in VirtualBox to create a VM, you write a single script (the **Vagrantfile**), and Vagrant builds the machine for you automatically.

The Key Difference:

- **VirtualBox/VMWare:** The "Provider" (The software that runs the virtual hardware).
- **Vagrant:** The "Manager" (The tool that tells the provider exactly how to build and configure the machine).

2. Importance of Vagrant

Why do we bother with Vagrant when we have Docker?

1. **Isolation:** It isolates dependencies.You can run PHP 5.6 for one project and PHP 8.3 for another on the same laptop without conflicts.
2. **Team Consistency:** You can share your Vagrantfile with a teammate. When they run vagrant up, they get an **identical** copy of your server.
3. **Cross-Platform:** It works the same on Windows, Mac, and Linux.
4. **Disposability:** If you mess up the server configuration, you don't have to fix it. You just "destroy" it and "up" it again in seconds.

3. Setup & Installation

To use Vagrant, you need two things: **Vagrant** (the manager) and a **Provider** (the engine).

A. Setup with VirtualBox

1. Install **VirtualBox**.
2. Install **Vagrant**.
3. Initialize a project: vagrant init ubuntu/focal64 (This creates a basic Vagrantfile).
4. Start the machine: vagrant up.

B. Setup with Docker

Vagrant can also manage Docker containers instead of heavy Virtual Machines. This is much faster and uses less RAM.

- In your Vagrantfile, you simply change the provider to config.vm.provider "docker".

4. Writing the Vagrantfile

The **Vagrantfile** is a Ruby-based configuration file that defines the "DNA" of your server.

Key Features and Components:

- **Boxes:** These are pre-packaged "base images" (like an ISO). Example: ubuntu/bionic64.
- **Networking:**
  - *Forwarded Ports:* Map your laptop's port 8080 to the VM's port 80.
  - *Private Network:* Give the VM a specific IP address (e.g., 192.168.33.10).
- **Synced Folders:** Automatically share a folder on your laptop with a folder inside the VM. You edit code on your laptop; it updates live on the server.

5. Provisioning: Automating the Setup

Provisioning is the most powerful part of Vagrant. It allows you to automatically install software (Nginx, MySQL, etc.) the moment the machine is created.

A. Shell Provisioning (Basic)

You can write a script directly in the Vagrantfile:

Ruby
```
Vagrant.configure("2") do |config|
  config.vm.box = "ubuntu/focal64"

  # The Provisioner
  config.vm.provision "shell", inline: <<-SHELL
    sudo apt-get update
    sudo apt-get install -y nginx
  SHELL
end
```

B. File Provisioning

You can tell Vagrant to move a file (like your custom Nginx config) from your laptop into the VM:

Ruby
```
config.vm.provision "file", source: "./nginx.conf", destination: "/etc/nginx/nginx.conf"
```

6. Common Vagrant Commands (The Cheat Sheet)

Command	Purpose

vagrant init	Creates a new Vagrantfile in the current folder.
vagrant up	Creates and starts the virtual machine.
vagrant ssh	Logs you into the terminal of the VM.
vagrant halt	Shuts down the VM (keeps the data).
vagrant destroy	Deletes the VM entirely (wipes the data).
vagrant reload --provision	Restarts the VM and re-runs the setup scripts.