**Instalacja Virtual Box’a Linuxa (Ubuntu) z Vagrant na Windows 11 – instrukcja krok po kroku**

**1. Zainstaluj VirtualBox**

* Pobierz z: https://www.virtualbox.org/wiki/Downloads
* Zainstaluj z domyślnymi ustawieniami.

**✅ 2. Zainstaluj Vagrant**

* Pobierz z: https://www.vagrantup.com/downloads
* Zainstaluj z domyślnymi ustawieniami.
* Sprawdź w terminalu (PowerShell):

vagrant –version -> 2.4…

**🔹 Krok 2: Utwórz katalog projektu**

1. **Utwórz folder projektu:**

mkdir C:\Users\<mój\_user>\vagrant-lab

1. **Zainicjalizuj projekt:**
2. Otwórz wiersz polecenia cmd lub PowerShell w katalogu projektu
3. Przejdź do katalogu projektu:

cd C:\Users\<mój\_user>\vagrant-lab

1. Zainicjalizuj project:

vagrant init *ubuntu/jammy64*

**🔹 Krok 3: Edytuj plik Vagrantfile**

**Zamień zawartość pliku Vagrantfile na:**

**---------------------------------------------------------**

# Ustawienie wersji konfiguracji Vagranta (wersja 2 to standard dla VirtualBoxa)

Vagrant.configure("2") do |config|

# Określenie obrazu systemu operacyjnego (boxa) - Ubuntu 22.04 LTS

config.vm.box = "ubuntu/jammy64"

# Nazwa hosta maszyny wirtualnej (przydatna w sieci lokalnej lub do identyfikacji)

config.vm.hostname = "shell-lab"

# Konfiguracja sieci prywatnej - maszyna dostaje adres IP z DHCP

# config.vm.network "private\_network", type: "dhcp"

# Konfiguracja dostawcy maszyn wirtualnych - w tym przypadku VirtualBox

config.vm.provider "virtualbox" do |vb|

# Przydzielenie 2048 MB pamięci RAM do maszyny

vb.memory = "2048"

# Przydzielenie 2 rdzeni procesora

vb.cpus = 2

# Wyłączenie domyślnego /vagrant

config.vm.synced\_folder ".", "/vagrant", disabled: true

# Udostępnienie folderu Vagranta do maszyny wirtualnej

# Katalog ../vagrant/lab ma typ vboxsf (VirtualBox Shared Folders) co powoduje, że

# wszystkie pliki mają chmod 777 i należą do użytkownika vagrant i grupy vagrant

# nie można zmienić tych uprawnień ani właściciela przez chmod czy chown

config.vm.synced\_folder "./shared\_files", "/home/vagrant/shared"

end

# Provisioning - automatyczne uruchomienie skryptu shellowego po utworzeniu maszyny

config.vm.provision "shell", path: "provision.sh"

# lub

# config.vm.provision "shell", path: "provision.sh", run: "always"

# Zakończenie bloku konfiguracji

End

-----------------------------------------------------------

**🔹 Krok 4: Stwórz plik provision.sh (prowizjonowanie)**

W tym samym katalogu stwórz plik provision.sh:

--------------------------------------------------------------------------

#!/bin/bash

apt update && apt upgrade -y

# Podstawowe narzędzia

apt install -y curl wget git vim net-tools htop unzip whois

# Bash i narzędzia tekstowe

apt install -y gawk sed grep logrotate

# Ustawienie strefy czasowej na Warszawę

timedatectl set-timezone Europe/Warsaw

# Powitanie

echo "Witaj w laboratorium!" > /home/vagrant/intro.txt

-------------------------------------------------------------

**🔹 Krok 5: Uruchom maszynę wirtualną**

Z wiersza poleceń cmd lub PowerShell, będąc w katalogu projektu wpisz:

vagrant up

To:

* Pobierze Ubuntu (jednorazowo),
* Utworzy maszynę,
* Wykona provisioning (provision.sh).

**🔹 Krok 6: Połącz się z maszyną**

1. **Po zakończeniu vargant up wpisz**:

vagrant ssh

Teraz jesteś wewnątrz Linuxa, gotowy do pracy:

vagrant@shell-lab:~$ cat info.txt

1. **Wychodzenie z sesji ssh:**

exit # wychodzimy z SSH

1. **Połączenie z maszyną bezpośrednio poprzez ssh:**

Wpisz w konsoli cmd lub PowerShell:

ssh -i " C:/Users/<mój\_user>/vagrant-lab/.vagrant/machines/default/virtualbox/private\_key" vagrant@127.0.0.1 -p 2222

📌 Uwaga: użyj pełnej ścieżki do pliku private\_key.

**Podstawowe polecenia Vagrant:**

**1. vagrant up**

▶️ Opis: Uruchamia maszynę wirtualną.

* Jeśli maszyna nie istnieje – tworzy ją, pobiera system (box), ustawia konfigurację i uruchamia provision.sh.
* Jeśli maszyna już istnieje – po prostu ją uruchamia.
* Sprawdza stan maszyny i synchronizuje katalogi.

📌 Typowe użycie: start środowiska pracy.

**2. vagrant halt**

⏹️ Opis: Wyłącza maszynę wirtualną „normalnie”, czyli tak, jakbyśmy kliknęli „Shut Down” w systemie.

* Zatrzymuje ją w sposób kontrolowany.
* Maszyna nadal istnieje i można ją później uruchomić vagrant up.

📌 Typowe użycie: zakończenie pracy na dziś.

**3. vagrant reload**  lub **vagrant reload --provision**

🔁 Opis: Restart maszyny wirtualnej z ponownym wczytaniem konfiguracji z Vagrantfile.

* Używane, gdy zmieniono konfigurację (np. RAM, CPU, sieć) i trzeba ją zastosować.
* Nie wykonuje provision.sh, chyba że użyjesz --provision.

📌 Typowe użycie: gdy zmienisz Vagrantfile i/lub provision.sh chcesz, by maszyna to uwzględniła.

**4. vagrant destroy -f**

💣 Opis: Trwale usuwa maszynę wirtualną bez pytania o potwierdzenie.

* Usuwa ją z VirtualBoxa.
* Czyści katalog .vagrant/ zawierający dane o stanie maszyny.
* Nie rusza plików projektu ani pobranych boxów.

📌 Typowe użycie: gdy chcesz zacząć od nowa lub przetestować nową konfigurację.

**4. vagrant provision**

⚙️ Opis: Wymusza ponowne wykonanie skryptu provisioningowego (np. provision.sh) na już istniejącej i uruchomionej maszynie.

* Nie tworzy nowej maszyny.
* Nie uruchamia maszyny, jeśli jest wyłączona.
* Wykonuje wszystkie zdefiniowane akcje config.vm.provision w Vagrantfile.

📌 Typowe użycie:

* Gdy zmieniłeś provision.sh i chcesz zastosować zmiany bez niszczenia maszyny.
* Gdy chcesz „doładować” nowe pakiety, katalogi lub konfigurację do działającego systemu.
* Gdy testujesz provisioning wielokrotnie.

**Ustawienie VSC do dostępu po SSH:**

**1. Zainstaluj rozszerzenie "Remote - SSH" w VS Code**

1. Otwórz VS Code.
2. Przejdź do Extensions (Ctrl+Shift+X).
3. Wpisz: Remote - SSH → kliknij Install (wydawcą jest Microsoft).

**🔹 2. Sprawdź dane połączenia SSH z maszyną Vagrant**

W katalogu, gdzie masz Vagrantfile, uruchom terminal (np. PowerShell) i wpisz:

vagrant ssh-config

Otrzymasz coś takiego:

Host default

HostName 127.0.0.1

User vagrant

Port 2222

IdentityFile C:/Users/<mój\_user>/vargant-lab/.vagrant/machines/default/virtualbox/private\_key

**🔹 3. Skonfiguruj plik config SSH**

1. Otwórz plik:  
   C:\Users\<mój\_user>\.ssh\config  
   *(jeśli nie istnieje – stwórz go)*
2. Wklej do niego konfigurację:

Host vagrant-vm-<mój\_user>

HostName 127.0.0.1

User vagrant

Port 2222

IdentityFile C:\Users\<mój\_user>\vargant-lab\vagrant\machines\default\virtualbox\private\_key

StrictHostKeyChecking no

UserKnownHostsFile /dev/null

📌 Uwaga: użyj pełnej ścieżki do pliku private\_key.

**🔹 4. Połącz się przez VS Code**

1. Otwórz paletę poleceń: Ctrl+Shift+P
2. Wpisz: Remote-SSH: Connect to Host...
3. Wybierz: vagrant-vm

📌 Pierwsze połączenie może chwilę potrwać – VS Code zainstaluje komponenty na maszynie.

**🔹 5. Otwórz dowolny folder maszyny**

Po połączeniu możesz w VS Code:

* otwierać katalog /home/vagrant lub /etc,
* edytować pliki Linuxowe,
* korzystać z terminala tej maszyny bezpośrednio.

**🔹 6. Połączenie z maszyną bezpośrednio poprzez ssh:**

Po zdefiniowaniu hostów ssh w C:\Users\<mój\_user>\.ssh\config

Możesz logować się do maszyny wprost poprzez ssh, wpisując w konsoli cmd lub PowerShell:

ssh -i " C:/Users/<mój\_user>/vagrant-lab/.vagrant/machines/default/virtualbox/private\_key" vagrant@127.0.0.1 -p 2222

📌 Uwaga: użyj pełnej ścieżki do pliku private\_key.