

VAGRANT

EXERCICE 1

```
Vagrant.configure("2") do |config|  
  config.vm.box = "debian/bookworm64"  
  config.vm.network "private_network", ip: "192.168.1.5", virtualbox__intnet: "intnet"  
  config.vm.provision "shell", path: "install.sh"  
end
```

Figure 1 : Création du VAGRANTFILE

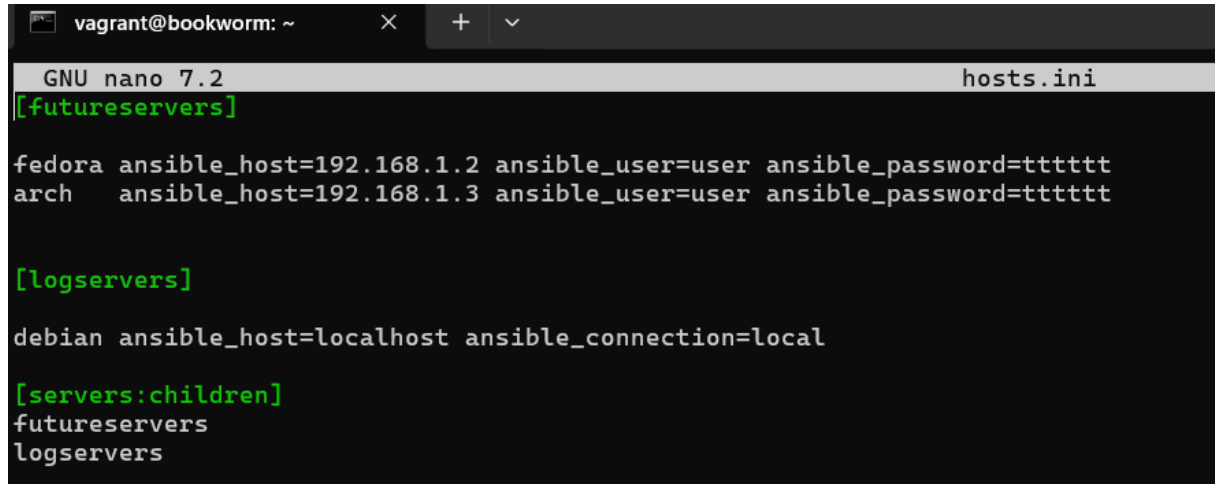
```
C: > Users > infin > OneDrive > Bureau > VAGRANT_EXAMEN > install.sh  
1  #!/bin/bash  
2  
3  apt-get update  
4  apt-get install -y docker.io  
5  
6  usermod -aG docker vagrant
```

Figure 2 : Création du script

- `docker --version`
- `cat /etc/debian_version`

ANSIBLE

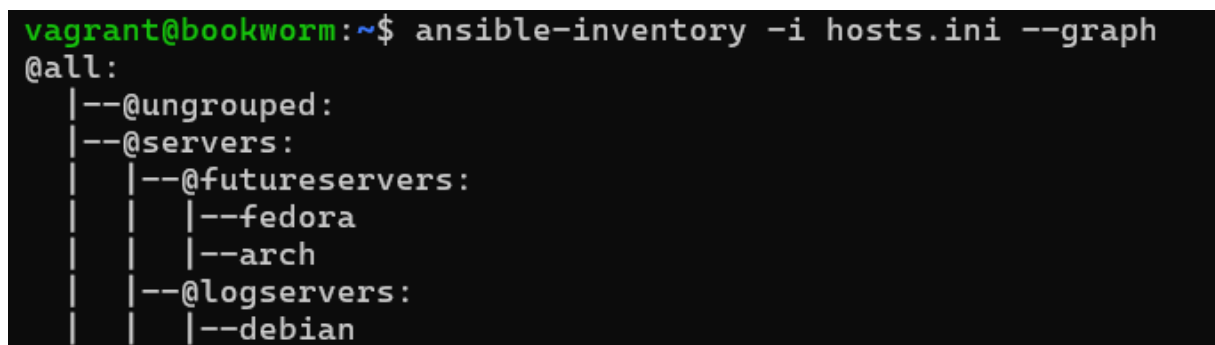
Exercice 2



```
vagrant@bookworm: ~  
GNU nano 7.2 hosts.ini  
[futureservers]  
  
fedora ansible_host=192.168.1.2 ansible_user=user ansible_password=tttttt  
arch   ansible_host=192.168.1.3 ansible_user=user ansible_password=tttttt  
  
[logservers]  
  
debian ansible_host=localhost ansible_connection=local  
  
[servers:children]  
futureservers  
logservers
```

Figure 3 : Création du fichier hosts

Exercice 3



```
vagrant@bookworm:~$ ansible-inventory -i hosts.ini --graph  
@all:  
  |--@ungrouped:  
  |--@servers:  
    |--@futureservers:  
      |--fedora  
      |--arch  
    |--@logservers:  
      |--debian
```

Figure 4 : Preuve des groupes

- `ansible-inventory -i hosts.ini --graph`

Exercice 4

Prérequis :

- `ssh-keygen`

```

GNU nano 7.2 bootstrap.yml
- name: Playbook de bootstrapping
  hosts: servers
  become: yes
  tasks:
    - name: crée user toto
      user:
        name: toto
        create_home: yes
        shell: /bin/bash
        state: present

    - name: Copier clé SSH
      authorized_key:
        user: toto
        state: present
        key: "{{ lookup('file', '/home/vagrant/.ssh/id_rsa.pub') }}"

    - name: Configurer sudo NOPASSWD
      lineinfile:
        path: /etc/sudoers.d/toto
        line: 'toto ALL=(ALL) NOPASSWD: ALL'
        create: yes
        mode: 0440
...

```

Figure 5 : Playbook Bootstrap

- `ansible-playbook -i hosts.ini bootstrap.yml -K`

EXERCICE 5

```

- name: Installation & config
  hosts: servers
  remote_user: toto
  become: yes

  tasks:
    - name: Update debian
      apt:
        update_cache: yes
        when: ansible_os_family == "Debian"

    - name: Update Fedora
      dnf:
        update_cache: yes
        when: ansible_distribution == "Fedora"

    - name: Update Arch Linux
      pacman:
        update_cache: yes
        when: ansible_distribution == "Archlinux"

    - name: Install packages Fedora
      dnf:
        name:
          - docker
          - docker-compose
          - podman
        state: present
        when: ansible_distribution == "Fedora"

    - name: Démarrer Docker
      service:
        name: docker
        state: started
        enabled: yes
        when: ansible_distribution == "Fedora"

```

Figure 6 : Playbook Update

```

- name: Install packages Arch
  pacman:
    name:
      - nginx
      - net-tools
    state: present
    when: ansible_distribution == "Archlinux"

- name: Démarrer Nginx
  service:
    name: nginx
    state: started
    enabled: yes
    when: ansible_distribution == "Archlinux"

- name: Install packages debian
  apt:
    name: net-tools
    state: present
    when: ansible_os_family == "Debian"
...

```

Figure 7 : Playbook Update PARTIE 2

- `ansible fedora -i hosts.ini -u toto -m shell -a "systemctl is-active docker"`
- `ansible arch -i hosts.ini -u toto -m shell -a "systemctl is-active nginx"`
- `ansible logservers -i hosts.ini -u toto -m shell -a "dpkg -l | grep net-tools"`

CONTENEURS

Exercice 6

```

vagrant@bookworm:~$ docker network create net_exam
7842e1fe236ab52d266baf0fadfc453fc46240d3da65d50d1b61b71ca9014d2e
vagrant@bookworm:~$ mkdir exam && cd exam
vagrant@bookworm:~/exam$ echo "Site web" > index.html
vagrant@bookworm:~/exam$ nano Dockerfile

```

Figure 8 : Création réseau, fichier html & dossier

```
FROM debian:bullseye-slim

RUN apt-get update && apt-get install -y curl default-mysql-client python3

RUN useradd -m toto

COPY index.html /home/toto/index.html

USER toto

CMD ["python3", "-m", "http.server", "8080", "-d", "/home/toto"]
```

Figure 9 : Fichier Dockerfile

```
vagrant@bookworm:~/exam$ docker run -d --name db --net net_exam -v vol_sql:/var/lib/mysql -e MYSQL_ROOT_PASSWORD=root mariadb
Unable to find image 'mariadb:latest' locally
latest: Pulling from library/mariadb
a3629ac5b9f4: Pull complete
5ac986756138: Pull complete
87224913ffe8: Pull complete
a57bb736d70f: Pull complete
b90af7f0b0f: Extracting [=====] 329B/329B
9a609955d447: Download complete
b90af7f0b0f: Pull complete
9a609955d447: Pull complete
096c936d41b1: Pull complete
01df660f0d50: Pull complete
Digest: sha256:f54db0cb3ccfe9431aba6d08c65a1763c499789b116b4cb651dd7fcf325965b3
Status: Downloaded newer image for mariadb:latest
820160a62917cac615398640baaa3fbf67fbd6c456cd68771311b1ffbddfc47f
```

Figure 10 : Lance la BDD (VOLUME / RESEAU / NOM)

```
vagrant@bookworm:~/exam$ docker build -t web_img .
Sending build context to Docker daemon 3.072kB
Step 1/6 : FROM debian:bullseye-slim
bullseye-slim: Pulling from library/debian
bd1b97a95a10: Pull complete
Digest: sha256:b32674fb57780ad57d7b0749242d3f585f462f4ec4a60ae0adacd945f9cb9734
Status: Downloaded newer image for debian:bullseye-slim
--> e5e3fc0e0386
Step 2/6 : RUN apt-get update && apt-get install -y curl default-mysql-client python3
--> Running in 5bfd23048eca
Get:1 http://deb.debian.org/debian bullseye InRelease [75.1 kB]
Get:2 http://deb.debian.org/debian-security bullseye-security InRelease [27.2 kB]
Get:3 http://deb.debian.org/debian bullseye-updates InRelease [44.0 kB]
Get:4 http://deb.debian.org/debian bullseye/main amd64 Packages [8066 kB]
Get:5 http://deb.debian.org/debian-security bullseye-security/main amd64 Packages [440 kB]
Get:6 http://deb.debian.org/debian bullseye-updates/main amd64 Packages [18.8 kB]
Fetched 8672 kB in 4s (2418 kB/s)
Reading package lists...
Reading package lists...
Building dependency tree...
```

Figure 11 : Construction de l'image web

```
vagrant@bookworm:~/exam$ docker run -d --name web --net net_exam -p 5555:8080 web_img
034abff28252de1ae9673b4cd9a4c2c151b23c5ff5167afbcc7df94fe5ae4804
```

Figure 12 : Lancement du site web PORT 5555 & Réseau

- -> `curl http://localhost:5555/`
- -> `docker exec web mysql -h db -u root -p -e "SHOW DATABASES;"`
- -> `docker exec web whoami`
- -> `docker inspect -f '{{.Mounts}}' db`