ANSIBLE

Thomas PLOIX

3-1 Document your inventory and base commands

```
all:
  vars:
    ansible_user: admin
    ansible_ssh_private_key_file: /home/tploix/.ssh/id_rsa_2
  children:
    prod:
      hosts: thomas.ploix.takima.cloud
```

All -> permet de mettre le scope sur tous les hôtes.

Vars -> variables globals qu'on doit indiqué.

ansible user -> le user que va utiliser ansible pour se connecter et pour apporter des modifications à notre serveur distant .
ssh_private_key -> le chemin vers la clée privée ssh qui permet de se connecter au serveur en utilisant le protocole ssh.

Children -> permet de définir des sous groupes (ici que 1)

Prod : petit sous groupe pour l'environnement de production
Host : hote où ansible va se connecter

Command utile :

- ansible all -i inventories/setup.yml -m ping
- ansible-playbook -i inventories/setup.yml playbook.yml
- ansible all -i inventories/setup.yml -m setup -a "filter=ansible_distribution*"

3-2 Document your playbook

- hosts: all On cible tout les hôtes définit dans le scope all

gather_facts: true On va collecter les informations systèmes du server avant d'exécuter les taches

become: true Permet d'exécuter les commandes avec des privilèges élévé (comme root)

roles: On définit les différents l'ordre des roles qui vont être appeler, pour structurer et organiser le playbook. Chaque rôle contient des tâches, des fichiers, des templates, des variables, et d'autres éléments organisés pour une fonctionnalité spécifique.

Document your docker_container tasks configuration.

- docker Permet d'installer docker et ses dépendances

```
---
# Install prerequisites for Docker
- name: Install required packages
  apt:
    name:
      - apt-transport-https
      - ca-certificates
      - curl
      - gnupg
      - lsb-release
      - python3-venv
    state: latest
    update_cache: yes

# Add Docker's official GPG key
- name: Add Docker GPG key
  apt_key:
    url: https://download.docker.com/linux/debian/gpg
    state: present

# Set up the Docker stable repository
- name: Add Docker APT repository
  apt_repository:
    repo: "deb [arch=amd64] https://download.docker.com/linux/debian {{
ansible_facts['distribution_release'] }} stable"
    state: present
    update_cache: yes

# Install Docker
- name: Install Docker
  apt:
    name: docker-ce
    state: present

# Install Python3 and pip3
- name: Install Python3 and pip3
  apt:
    name:
      - python3
      - python3-pip
    state: present

# Create a virtual environment for Python packages
- name: Create a virtual environment for Docker SDK
  command: python3 -m venv /opt/docker_venv
  args:
    creates: /opt/docker_venv  # Only runs if this directory doesn't exist
```

```
# Install Docker SDK for Python in the virtual environment
- name: Install Docker SDK for Python in virtual environment
  command: /opt/docker_venv/bin/pip install docker

# Ensure Docker is running
- name: Make sure Docker is running
  service:
    name: docker
    state: started
  tags: docker
```

- network Créer le network docker

```
- name: Create a Docker network
  docker_network:
    name: my-network
    driver: bridge
    state: present
```

- database Créer et instancie la base de données depuis le docker Hub

```
- name: Install and Launch database
  docker_container:
    name: my-db
    image: tploix/tp-devops-database:latest
    pull: always
    env:
      POSTGRES_DB: db
      POSTGRES_USER: user
      POSTGRES_PASSWORD: pwd
    networks:
      - name: my-network
  tags: database
```

- app Créer et instancie le backend de l'application depuis un conteneur du docker hub

```
- name: Launch application container
  docker_container:
    name: my-api
    image: tploix/tp-devops-simple-api:latest
    pull: always
    state: started
    ports:
      - "8080:8080"
    networks:
      - name: my-network

  tags: app
```

- proxy Permet de reconfigurer le proxy par default de http2 pour rediriger une url vers notre backend

```
- name: Launch Proxy container
  docker_container:
    name: httpd
    image: tploix/tp-devops-proxy-server:latest
    pull: always
    state: started
    networks:
      - name: my-network
  tags: proxy
```

- front Permet de lancer le container qui contient le front de l'application.

```
- name: Launch front container
  docker_container:
    name: front
    image: tploix/tp-devops-front-server:latest
    pull: always
    ports:
      - "80:80"
    state: started
    networks:
      - name: my-network
```