ENGINEERING SCHOOL
Creating the future together

REPORT
DEVOPS

PART 03: ANSIBLE

WRITTEN BY :
ASSELE TSETSE Maria

PROFESSOR LEADER:
CHOUIB Chawki

Acknowledgements

I would like to express my deep gratitude to all the individuals who contributed to the organization and execution of the DevOps courses and practical work. Their sustained efforts, expertise, and dedication have been instrumental in helping me gain understanding.

I warmly thank my instructors for their enlightening teaching, insightful guidance, and availability throughout my learning journey. Their clear explanations, practical demonstrations, and passion for the subject greatly enhanced my understanding and strengthened my skills.

I also want to express my special gratitude to M. CHOUIB Chawki for their exceptional commitment and constant dedication to our learning. Their deep knowledge, invaluable support, and responses to all our questions have been sources of inspiration and motivation throughout this DevOps training. Their tireless efforts to guide us and help us understand complex concepts have had a significant impact on our progress and success. I am sincerely grateful to them for their valuable contribution to our training.

Introduction

In the field of IT and infrastructure management, the automation of recurring tasks has become a priority. Ansible, a tool for automating system administration, application deployment and configuration management tasks, meets this need. This project explores the capabilities of Ansible to simplify and accelerate the deployment and management of a web application on a remote server. Ansible offers a simple and effective approach to automating complex tasks, thanks to playbooks written in YAML, which describe the desired state of the infrastructure. By defining the tasks to be performed and the target servers, Ansible takes care of executing these tasks in a consistent and reproducible way. This approach reduces human error, ensures configuration consistency and accelerates application deployment. This tutorial explores Ansible's various functionalities, from inventory configuration and playbook creation to the deployment of a complete web application. It demonstrates how Ansible simplifies IT infrastructure management and facilitates application deployment, while offering greater visibility and control over the IT environment.

Table of contents

Acknowledgements 2
Introduction 3
I- Global 5
1. Inventories 5
2. Facts 5
II- Playbooks 6
1. First playbook 6
2. Advanced Playbook 6
3. Using roles 7
II- Application deployment 7
1. Front 8
Conclusion 11

I- Global

As a first step, I installed Ansible on my Windows Subsystem for Linux (WSL) environment.

Ajouter la capture si possible

1. Inventories

In my Ansible directory, I've created a directory called inventories. I've added a file called setup.yml, which I've completed with the script provided. The file contains the information needed to connect to the hosts, including the Ansible user and the path to the SSH private key.

```
all:
  vars:
    ansible_user: centos
    ansible_ssh_private_key_file: ~/.ssh/id_rsa
  children:
    prod:
      hosts: maria.asseletsetse.takima.cloud
```

To check that the inventory had been configured correctly, I used Ansible's « ***ping*** » command:

```
massele@mariaVictoire:/mnt/c/Users/assel/Documents/Ansible/inventories$ ansible all -i setup.yml -m ping
maria.asseletsetse.takima.cloud | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python"
    },
    "changed": false,
    "ping": "pong"
}
```

2. Facts

Next, I looked for information about my hosts. These types of variables, which are not user-defined but discovered, are called “facts”.

To obtain the operating system distribution for my servers, I used the « ***setup*** » module :

```
massele@mariaVictoire:/mnt/c/Users/assel/Documents/Ansible/inventories$ ansible all -i setup.yml -m setup -a "filter=ansible_distributio
maria.asseletsetse.takima.cloud | SUCCESS => {
    "ansible_facts": {
        "ansible_distribution": "CentOS",
        "ansible_distribution_file_parsed": true,
        "ansible_distribution_file_path": "/etc/redhat-release",
        "ansible_distribution_file_variety": "RedHat",
        "ansible_distribution_major_version": "7",
        "ansible_distribution_release": "Core",
        "ansible_distribution_version": "7.9",
        "discovered_interpreter_python": "/usr/bin/python"
    },
    "changed": false
}
```

Previously, I had installed the Apache httpd server on my machine. I used the ***yum*** module to manage packages on my servers, simply describing the desired state and letting Ansible take care of the details.

```
massele@mariaVictoire:/mnt/c/Users/assel/Documents/Ansible/inventories$ ansible all -i setup.yml -m yum -a "name=httpd state=absent" --become
maria.asseletsetse.takima.cloud | CHANGED => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python"
    },
    "changed": true,
    "changes": {
        "removed": [
            "httpd"
        ]
    },
    "msg": "",
    "rc": 0,
    "results": [
        "Loaded plugins: fastestmirror\nResolving Dependencies\n--> Running transaction check\n---> Package httpd.x86_64 0:2.4.6-99.el7.centos.1 will be era
sed\n--> Finished Dependency Resolution\n\nDependencies Resolved\n\n================================================================================\n Packa
ge      Arch          Version                    Repository       Size\n================================================================================\
nRemoving:\n httpd        x86_64        2.4.6-99.el7.centos.1         @updates        9.4 M\n\nTransaction Summary\n================================
================================\nRemove  1 Package\n\nInstalled size: 9.4 M\nDownloading packages:\nRunning transaction check\nRunning transaction
test\nTransaction test succeeded\nRunning transaction\n  Erasing    : httpd-2.4.6-99.el7.centos.1.x86_64                          1/1 \n  Verifying  : http
d-2.4.6-99.el7.centos.1.x86_64                          1/1 \n\nRemoved:\n  httpd.x86_64 0:2.4.6-99.el7.centos.1                                         \
n\nComplete!\n"
    ]
}
```

II- Playbooks

1. First playbook

I started by creating a very simple playbook to test the connection to my hosts. I created a file named playbook.yml in my Ansible directory

This playbook contains a task that uses the ***ping*** module to test the connection to all hosts defined in the inventory.

Before running it, I checked its syntax with the ***--syntax-check*** option. Once the verification was done, I ran this playbook :

```
massele@mariaVictoire:/mnt/c/Users/assel/Documents/Ansible$ ansible-playbook -i inventories/setup.yml playbook.yml

PLAY [all] *********************************************************************************************************************************************************

TASK [Test connection] *********************************************************************************************************************************************
ok: [maria.asseletsetse.takima.cloud]

PLAY RECAP *********************************************************************************************************************************************************
maria.asseletsetse.takima.cloud : ok=1    changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
```

2. Advanced Playbook

Then I modified my playbook to install Docker and its dependencies on my servers, following the official documentation.

I re-run my playbook :

```
massele@mariaVictoire:/mnt/c/Users/assel/Documents/Ansible$ ansible-playbook -i inventories/setup.yml playbook.yml

PLAY [all] *********************************************************************************************************************************************************

TASK [Test connection] *********************************************************************************************************************************************
ok: [maria.asseletsetse.takima.cloud]

PLAY RECAP *********************************************************************************************************************************************************
maria.asseletsetse.takima.cloud : ok=1    changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

massele@mariaVictoire:/mnt/c/Users/assel/Documents/Ansible$ ansible-playbook -i inventories/setup.yml playbook.yml

PLAY [all] *********************************************************************************************************************************************************

TASK [Install device-mapper-persistent-data] ***********************************************************************************************************************
changed: [maria.asseletsetse.takima.cloud]

TASK [Install lvm2] ************************************************************************************************************************************************
changed: [maria.asseletsetse.takima.cloud]

TASK [add repo docker] *********************************************************************************************************************************************
changed: [maria.asseletsetse.takima.cloud]

TASK [Install Docker] **********************************************************************************************************************************************
changed: [maria.asseletsetse.takima.cloud]

TASK [Install python3] *********************************************************************************************************************************************
changed: [maria.asseletsetse.takima.cloud]

TASK [Install docker with Python 3] ********************************************************************************************************************************
changed: [maria.asseletsetse.takima.cloud]

TASK [Make sure Docker is running] *********************************************************************************************************************************
changed: [maria.asseletsetse.takima.cloud]

PLAY RECAP *********************************************************************************************************************************************************
maria.asseletsetse.takima.cloud : ok=7    changed=7    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
```

Then I checked the Docker installation :

```
massele@mariaVictoire:/mnt/c/Users/assel/Documents/Ansible$ ansible all -m shell -a "docker --version" -i inventories
maria.asseletsetse.takima.cloud | CHANGED | rc=0 >>
Docker version 26.1.3, build b72abbb
```

3. Using roles

For cleaner, more modular configuration management, I've used Ansible roles. I created a specific role for installing Docker :

```
massele@mariaVictoire:/mnt/c/Users/assel/Documents/Ansible$ ansible-galaxy init roles/docker
- Role roles/docker was created successfully
```

II- Application deployment

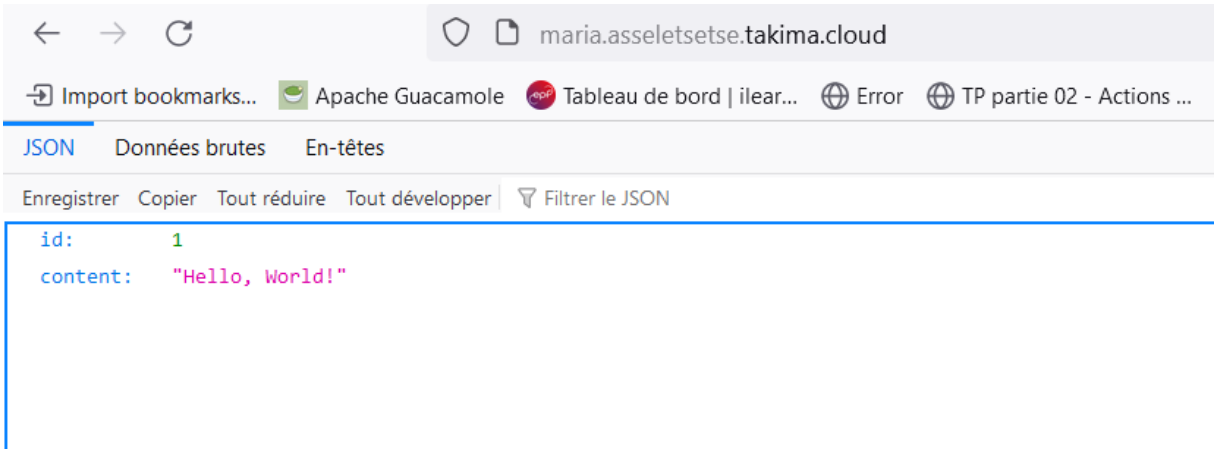
Now it's time to deploy your application on your Ansible-managed server. To do this, I created specific roles for each part of the application and used the Ansible docker_container module to start the application's Docker containers. I gradually wrote the main.yml files for these roles.

app	29/05/2024 11:14	Dossier de fichiers
database	29/05/2024 11:14	Dossier de fichiers
devops-front	29/05/2024 17:22	Dossier de fichiers
docker	29/05/2024 11:04	Dossier de fichiers
network	29/05/2024 11:13	Dossier de fichiers
proxy	29/05/2024 11:14	Dossier de fichiers

To deploy the application, I created a playbook that includes all these roles and then ran the playbook to deploy the application.

```
[centos@ip-10-0-1-215 ~]$ sudo docker ps
CONTAINER ID   IMAGE                                  COMMAND                  CREATED          STATUS          PORTS                  NAMES
461da616bd5c   mariavictoire/docker-compose-backend   "java -jar myapp.jar"    2 minutes ago    Up 2 minutes                           backend-container
49008a027efd   mariavictoire/docker-compose-frontend  "httpd-foreground"       7 minutes ago    Up 7 minutes    0.0.0.0:80->80/tcp     proxy-container
8344a11ceae8   mariavictoire/docker-compose-database  "docker-entrypoint.s…"   27 minutes ago   Up 27 minutes   5432/tcp               database-container
```

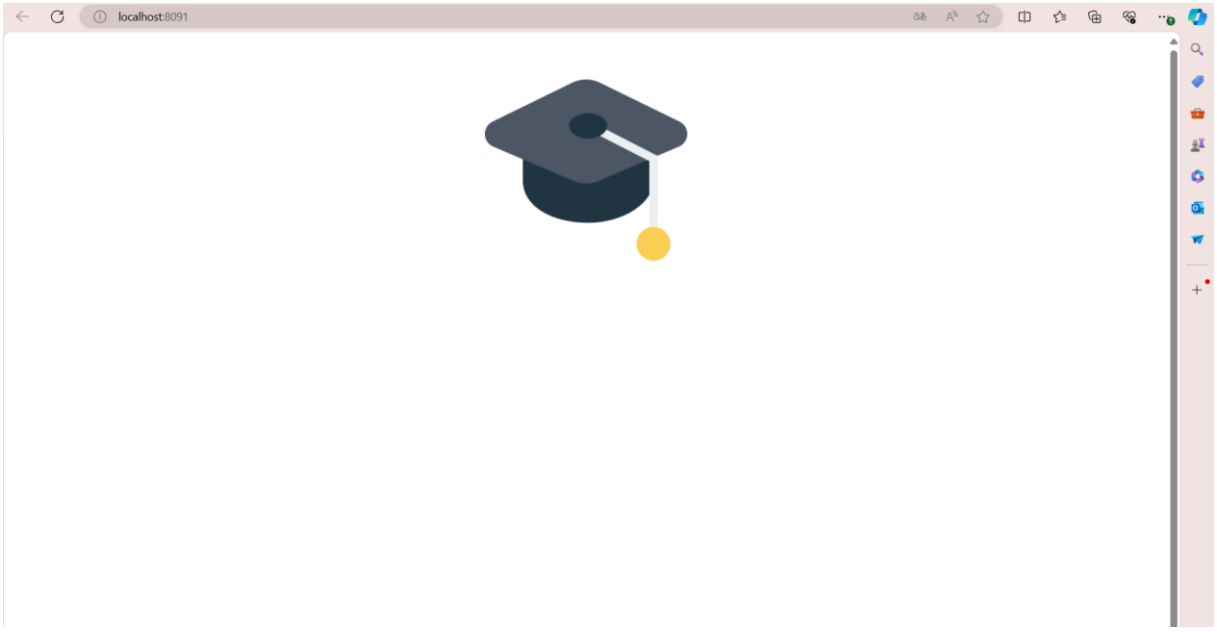
I was able to access my API on my server.



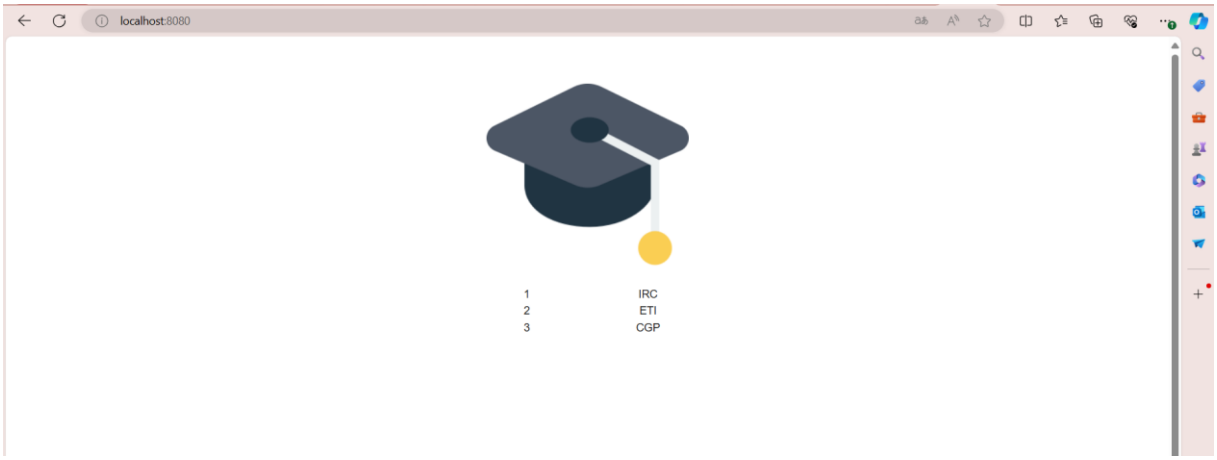
1. Front

After downloading the front-end document, I integrated it into my Git repository directory. I then made the necessary modifications, notably in the `docker-compose.yml` file, where I added the front-end. I also made sure to take into account the ports needed to ensure communication between the various application components.

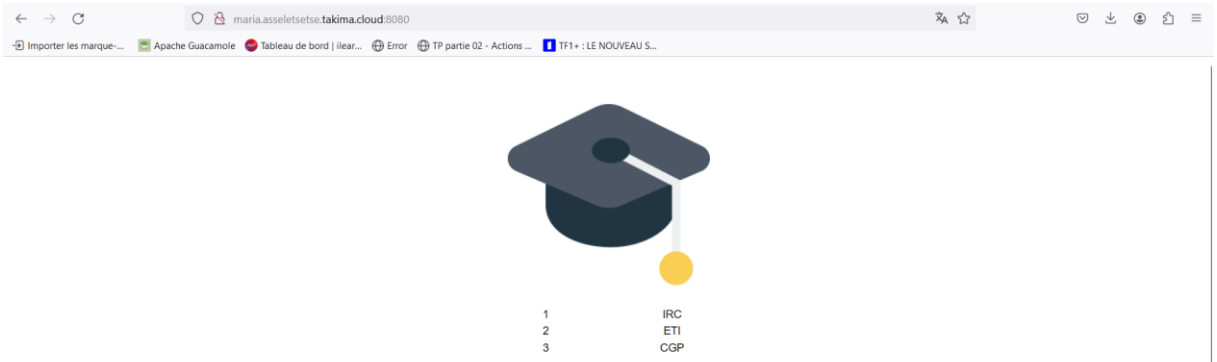
Before deploying on the remote server, I first tested the application locally to make sure everything was working properly. This enabled me to detect and correct any problems before final deployment on the server.



I then linked with my database to visualize the basic data on the front end.



After confirming that everything was running smoothly locally, I set about configuring access to my front-end via Ansible. I set up the necessary configurations to deploy the front-end on the remote server, using the Ansible playbooks I'd previously created.


maria.asseletsetse.takima.cloud:8080
Importer les marque-...
Apache Guacamole
Tableau de bord | ilear...
Error
TP partie 02 - Actions ...
TF1+ : LE NOUVEAU S...
1 IRC
2 ETI
3 CGP

Conclusion

In conclusion, this project has enabled me to better understand the concepts of infrastructure automation with Ansible and to apply them in a real-life scenario. I learned how to configure inventories, create playbooks to install software and deploy a complete application on a remote server. This experience showed me the benefits of automation to efficiently manage IT infrastructures and facilitate application deployment. I think I'm now better equipped to use Ansible in my future projects and take advantage of its features to simplify our management operations.