# Báo cáo đồ án môn Mạng Máy Tính Nâng Cao

## Chủ đề: Tìm hiểu Ansible và triển khai thử nghiệm


- 20424041 - Trần Quốc Khánh
- 20424037 - Trần Tuấn Huy
- 20424046 - Hồng Quốc Lâm

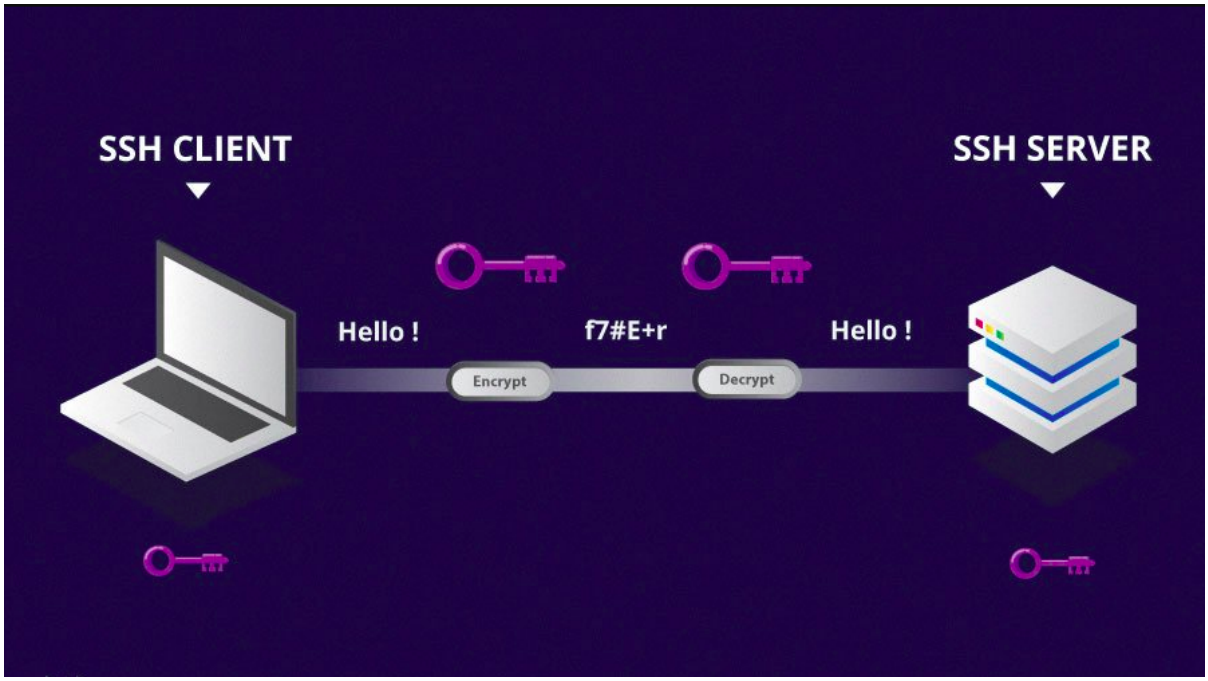



## I. Giao thức SSH

- SSH (secure shell) là một giao thức cho phép 2 máy tính thiết lập kết nối an toàn với nhau qua internet.
- Các thông tin truyền trong SSH sẽ được mã hóa. SSH là một giao thức an toàn, bảo mật.
- SSH cho phép người dùng kết nối, điều khiển máy tính khác từ xa.

Ứng dụng:

- Git
- Điều khiển server




## II. YAML

- Ngôn ngữ dùng để lưu trữ, serialize dữ liệu, tương tự như JSON hay XML
- Được dùng trong các ứng dụng
  - Ansible Playbook
  - GitHub Action
  - BitBucket Pipeline
  - Docker compose

Cú pháp: YAML có 3 kiểu dữ liệu chính

- Key value Pair (Cặp khóa và giá trị)

  Dữ liệu được biểu diễn theo cặp khóa và giá trị, được phân tách với nhau bằng dấu ":" (Lưu ý: luôn phải có khoảng trắng theo sau dấu :)

  ```
  Key: Value
  ```

- Array/List

  Các phần tử trong mảng sẽ được thể hiện bởi dấu gạch ngang "-". Cần có khoảng trắng trước mỗi mục. Các phần tử trong mảng phải có chung indentation và có thứ tự.

  ```
  Key:
  - value1
  - value2
  - value3
  ```

- Dictionary

Cú pháp giống Array nhưng không cần khai báo dấu gạch ngang "-" trước thuộc tính. Các thuộc tính không có thứ tự.

```
Key:
  value1
  value2
  value3
```

# III. Ansible

## 1. Giới thiệu Ansible

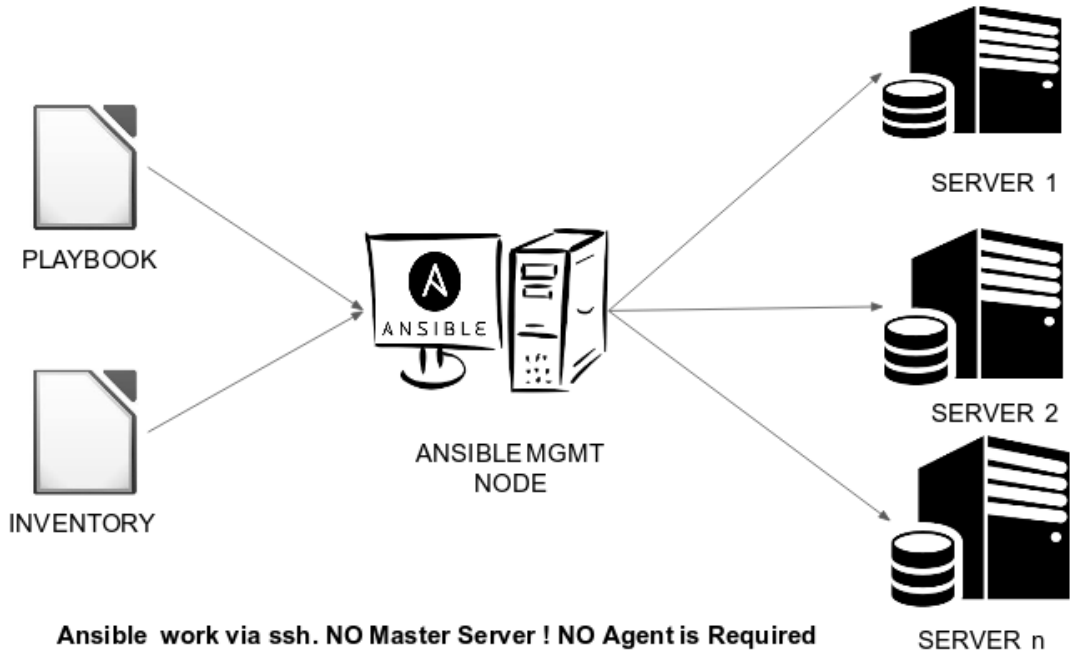
Ansible là 1 công cụ mã nguồn mở của Red Hat. Ansible dùng để tự động hoá quá trình cài đặt cấu hình, triển khai nhiều hệ thống từ xa.

## 2. Các ứng dụng của Ansible

- **Provisioning:** Khởi tạo VM, container hàng loạt trong môi trường cloud dựa trên API (OpenStack, AWS, Google Cloud, Azure...)
- **Configuration Management:** Quản lý cấu hình tập trung các dịch vụ tập trung, không cần phải tốn công chỉnh sửa cấu hình trên từng server.
- **Application Deployment:** Deploy ứng dụng hàng loạt.
- **Security & Compliance:** Quản lý các chính sách về an toàn thông tinmột cách đồng bộ trên nhiều môi trường và sản phẩm khác nhau (deploy policy, cấu hình firewall hàng loạt trên nhiều server...).

## 3. Cách hoạt động của Ansible

Ansible hoạt động bằng cách kết nối (mặc định bằng SSH) máy điều khiển (máy chạy ansible) tới các host (EC2, VPS, docker container, VM, ...) , và đẩy các module (chương trình để thực hiện công việc nào đó) sang máy host để thực thi, sau khi hoàn thành, kết quả được trả về máy chủ và các module bị đẩy sang host sẽ được xóa.




# 4. Ưu và nhược điểm

## a. Ưu điểm

- **Agentless**
  - Ansible sử dụng kiến trúc agentless để giao tiếp với các máy khác mà không cần agent (Không cần cài đặt phần mềm lên các agent, chỉ cần cài đặt tại master). Cơ bản nhất là giao tiếp thông qua giao thức SSH trên Linux, WinRM trên Windows hoặc giao tiếp qua chính API của thiết bị đó cung cấp.
  - Khi sử dụng Ansible thì ta không cần phải cài đặt thêm agent (phần mềm bên ngoài) vào các máy host nên quá trình cấu hình, triển khai phần mềm sẽ diễn ra đơn giản, nhanh chóng và ta không phải lo về sự tương thích giữa agent với các máy host
- **Dễ cài đặt và sử dụng**
  - Cài đặt nhẹ và đơn giản
  - Ansible sử dụng ngôn ngữ YAML cho việc viết các câu lệnh cấu hình, triển khai trong Ansible Playbooks**,** đơn giản và dễ học, dễ hiểu
  - Hệ thống tài liệu phong phú, đầy đủ
  - Cộng đồng hỗ trợ tốt
- **Infrastructure As Code**

  Các thiết lập cấu hình trong Ansible đều được viết dưới dạng code. Mang đến nhiều ưu điểm:

  - Đảm bảo tính nhất quán

    Các quy trình thủ công mặc dù có thể đảm bảo 1 phần nào đó tính nhất quán giữa các hệ thống nhưng yếu tố con người vẫn luôn có thể tạo ra 1 sự sai sót nào đó. Với IaC thì

toàn bộ quá trình sẽ được tự động hóa nên có thể đảm bảo rằng giữa các thiết bị sẽ luôn giống nhau, giúp quá trình debug nếu có xảy ra thì sẽ được giải quyết nhanh chóng

- Tương thích với công cụ version control

  Là đoạn script trong 1 file nên IaC hoàn toàn có thể được lưu trữ trên các công cụ như Git để tiện cho việc quản lý, nắm bắt được những thay đổi cũng như là thuận tiện cho việc chia sẻ giữa các thành viên trong nhóm/công ty

- Tái sử dụng

  Là 1 đoạn script thì ta có thể thực thi nó bao nhiêu lần tùy thích mà không phải lo lắng về việc tinh chỉnh cấu hình hay là phải để đúng người này làm... tiết kiệm một lượng lớn thời gian để giải quyết những vấn đề trọng yếu khác, đẩy nhanh quá trình phát triển phần mềm

- **Modular**

  Ansible sử dụng hệ thống các module, plugin để thực hiện các tác vụ quản lý, triển khai. Với hơn 2000 module (VD: module Azure, Docker, yum...), người dùng có thể thực hiện nhiều tác vụ khác nhau hoặc tự viết module tuỳ chỉnh sao cho phù hợp với từng nhu cầu của mình

- **Free (open-source)**

### b. Nhược điểm

- Hỗ trợ cho Windows chưa tốt

  Ansible quản lý các Host Linux/Unix sử dụng giao thức SSH nhưng với các Host Windows thì Ansible sẽ sử dụng Powershell remoting nên sẽ không có những tính năng bảo mật như SSH nhưng Ansible vẫn có thể giao tiếp được bằng module Python "winrm", cũng bởi vì thế mà mặc định các máy Windows không thể làm Management Node

- Số lượng kết nối lớn

  Đối với mỗi module ở trong Playbook, Ansible sẽ tạo kết nối mới tới các Host, và việc tạo nhiều kết nối có thể dẫn tới việc kết nối thất bại và 1 kết nối thất bại có thể ảnh hưởng tới cả quá trình thực thi script

- Thay đổi host khi Playbook đang chạy

  Một khi Playbook đã chạy thì ta không thể thêm hoặc xóa các Host trong Inventory, điều này có thể gây trở ngại trong 1 số trường hợp nhất định như nếu code của ta tạo 1 danh sách các địa chỉ IP trong cùng lúc mà script đang chạy thì ta không thể nào thêm danh sách đó vào Host được, điều này cũng tương tự cho các Variable

## 5. Các công cụ của Ansible

- Ansible Engine

  Thành phần chính của Ansible, đây là nơi xử lí các lệnh nhận được từ CLI và playbook.

- Ansible Tower, AWX

  Là các giao diện web giúp cho việc sử dụng Ansible trở nên thân thiện với người dùng mới ngoài ra hỗ trợ các tính năng hữu ích như lưu lại những output của các lần chạy của Playbook, cập nhật của các task trong thời gian thực, ... Điểm khác biệt giữa AWX và Tower

đó là AWX là một project mã nguồn mở, cập nhật thường xuyện tuy có thể không ổn định, Ansible Tower là sản phẩm thương mai của Red Hat nên sẽ được tích hợp nhiều tính năng hơn như Git, thông báo Slack, ... nhưng sẽ có tính phí

- Ansible Vault

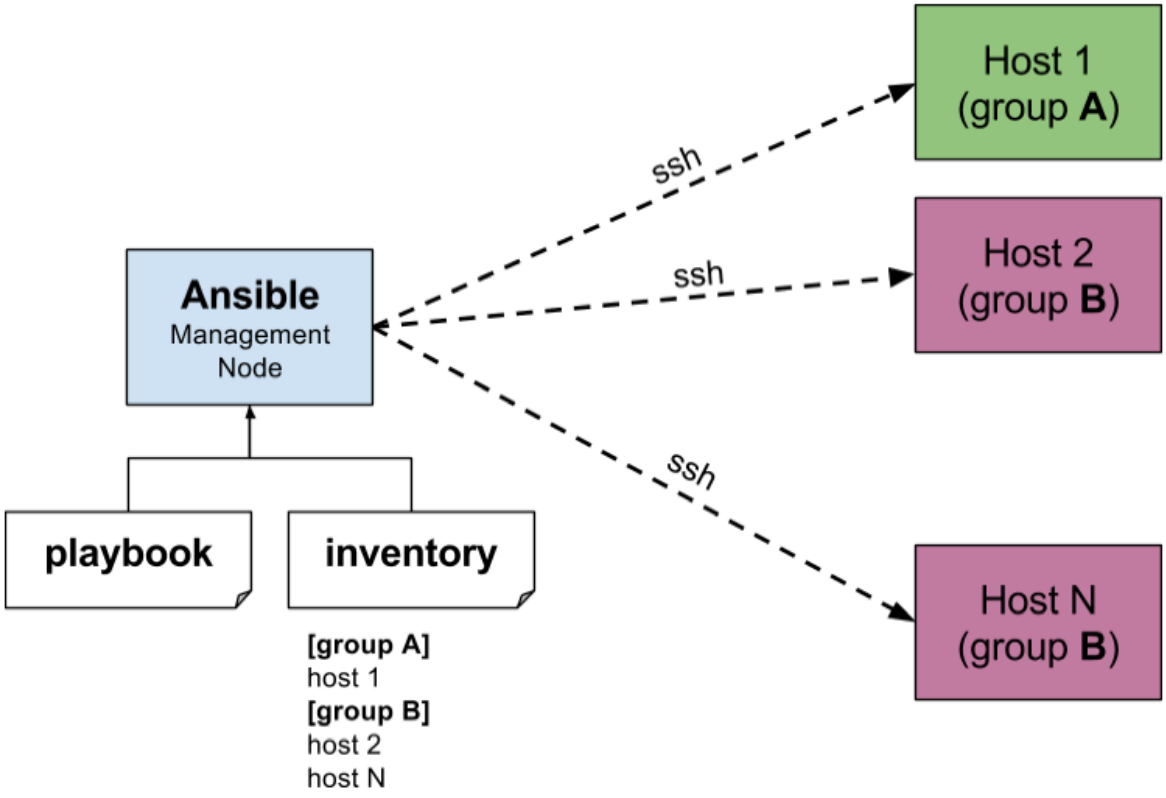
  Ansible Vault là một tính năng bảo mật các thông tin quan trọng như mật khẩu, key. Các thông tin sẽ được mã hóa và lưu vào các file riêng.

  Ta có thể sử dụng Vault bằng lệnh `ansible-vault` hoặc với các flag (`--ask-vault-pass`, `--vault-password-file` , `--vault-id`).

- Ansible Galaxy

  Ansible Galaxy là 1 cổng thông tin cho việc tìm kiếm, tái sử dụng và chia sẻ các nội dung liên quan đến Ansible. Lợi thế lớn nhất của Ansible Galaxy nằm trong ví dụ tải về các Role có thể tái sử dụng cho việc cài đặt các ứng dụng hoặc cấu hình server. Những lượt tải này thích hợp cho Playbook của một người dùng nhất định và có thể góp phần vào việc thúc đẩy tốc độ deploy.

## 6. Các thành phần trong Ansible




- **Controller Machine (Management Node)**: Là máy cài Ansible, chịu trách nhiệm quản lý, điều khiển và gởi task tới các máy con cần quản lý.
- **Managed Node (Host)**: Là máy con được kết nối, nhận và thực thi các module được gửi sang.

- **Inventory**: Là file chứa thông tin các server (host) cần được cấu hình, triển khai và quản lý

```
127.0.0.1

[mail]
127.0.0.2
127.0.0.3

[db]
127.0.0.4

[web]
127.0.0.5

[all_servers:children]
mail
db
web
```

[mail], [db], [web]: Cách khai báo 1 group các server với nhau với [mail] là tên group. Nếu trong playbook bạn muốn thực thi các task liên quan đến group web thì ta sẽ khai báo như sau

```
hosts: web
```

Còn [all_servers:children] là cách khai báo group các group với nhau.

Ngoài ra Ansible còn cung cấp 1 số param để phục vụ cho việc truy cập vào server dễ dàng hơn.

```
[web:vars]
ansible_user=root
ansible_ssh_pass=1
```

Hoặc ta có thể khai báo theo cách này

```
[web]
host1 ansible_host=192.168.100.2 ansible_user=root ansible_ssh_pass=1
```

- **Tasks**: Một block ghi công việc cần thực hiện trong playbook và các thông số liên quan. Ví dụ 1 playbook có thể chứa 2 task là: yum update và yum install vim.
- **Playbook**: Là file chứa các task của Ansible được ghi dưới định dạng YAML. Máy controller sẽ đọc các task trong Playbook và đẩy các lệnh thực thi tương ứng bằng Python xuống các máy con.

  Ví dụ 1 Playbook đơn giản

```
- name: Example Playbook
  hosts: web
  vars:
```

```
    message: "Hi from server"
  tasks:
    - name: Execute script on server
      script: test_script.sh
    - name: Install httpd service
      yum:
        name: httpd
        state: present
    - name: Start web server
      service:
        name: httpd
        state: started
    - name: Send message
      command: echo "Message is {{ message }}"
```

Playbook trong ví dụ trên chứa 1 kịch bản có tên Example Playbook (name: Examle Playbook)

Playbook sẽ thực thi các task trên những server thuộc group [web] (hosts: web)

Playbook sẽ có biến cục bộ message để có thể sử dụng trong các task

Playbook sẽ thực thi tổng cộng 3 tasks, lần lượt là

- Chạy test_script.sh trên server
- Cài đặt dịch vụ httpd
- Chạy dịch vụ httpd
- Gửi message tới host

Các task được liệt kê dạng Array nên việc đổi chỗ các task có thể dẫn tới việc Playbook cho ra kết quả không như mong đợi

Ngoài ra những thuộc tính như script, yum, service là những module mà Playbook trên sử dụng, thì Ansible còn cung cấp hàng trăm module khác phù hợp với nhu cầu của từng người dùng trên website của Ansible

Để chạy một Playbook ta sử dụng lệnh **ansible-playbook**

```
ansible-playbook your_playbook_file_name.yml
```

- **Modules**: Là 1 tập các lệnh thực thi một công việc nào đó được đóng gói sẵn để được dùng trong Playbook. Một số module phổ biến trong Ansible:
  - Các module liên quan tới File như fetch, find, stat, template, ...
  - Các module liên quan tới Command như command, expect, shell, raw, ...
  - ...

- **Role**: Role cho phép người dùng tách 1 file Playbook phức tạp thành nhiều comonent nhỏ để có thể sử dụng lại thay vì gom hết vào 1 chỗ, mỗi component sẽ thực thi 1 tác vụ nhất định

# 7. Ví dụ

Sau đây chúng ta sẽ dùng ansible để cấu hình cho một server chạy một trang web.

Trong phần ví dụ này nhóm sẽ sử dụng server là một docker container, các bạn có thể thay đổi bằng server khác (VPS, VM, ...) tùy ý.

**Bước 1: Cài đặt ansible trên controller machine**

Cách cài đặt chi tiết có thể xem ở trang chủ ansible

Kiểm tra phiên bản ansible bằng lệnh `ansible --version`

```
$ ansible --version
ansible 2.9.9
  config file = /etc/ansible/ansible.cfg
  configured module search path = ['/home/huy/.ansible/plugins/modules', '/usr/share/ansible/plugins/module
  ansible python module location = /usr/lib/python3/dist-packages/ansible
  executable location = /usr/bin/ansible
  python version = 3.8.6 (default, Jan 27 2021, 15:42:20) [GCC 10.2.0]
```

**Bước 2: Cấu hình SSH và khai báo file inventory**

Nếu bạn nào sử dụng một server khác có thể đi thẳng đến phần tạo inventory.

Tạo một Dockerfile với nội dung như sau:

```
# Sử dụng ubuntu phiển bản mới nhất
FROM ubuntu:latest

# Cài đặt các packages cần thiết
RUN apt update
RUN apt -y install curl dirmngr apt-transport-https lsb-release ca-certificates sudo openssh-server

# Tạo một user có username=test và password=test
RUN useradd -rm -d /home/ubuntu -s /bin/bash -g root -G sudo test
RUN su - test
RUN  echo 'test:test' | chpasswd

# Chạy ssh
RUN service ssh start
EXPOSE 22 3000 8080
CMD ["/usr/sbin/sshd","-D"]
```

Build docker images theo Dockerfile

```
$ docker build -t ubuntu_ssh .
```

Kiểm tra docker images vừa tạo

```
$ docker images
REPOSITORY               TAG                 IMAGE ID            CREATED             SIZE
ubuntu_ssh               latest              acb4634eb5c7        3 seconds ago       240MB
```

Tạo và chạy docker container

```
$ docker run -p 22:22 -p 8080:8080 --name ubuntu_ssh_1 ubuntu_ssh
```

Kiểm tra và lấy địa chỉ IP của container

```
$ docker ps
CONTAINER ID        IMAGE               COMMAND               CREATED             STATUS              PORTS
c0520d98d898        a7c17b0056bf        "/usr/sbin/sshd -D"   39 hours ago        Up 3 seconds        0.0.0

$ docker inspect --format '{{ .NetworkSettings.IPAddress }}' ubuntu_ssh_1
172.17.0.2
```

Cuối cùng **tạo file inventory** chứa thông tin kết nối SSH của container và kiểm tra

```
$ mkdir inventory

$ nano inventory/hosts
---GNU nano---
[production]
172.17.0.2 ansible_connection=ssh ansible_user=test ansible_ssh_pass=test ansible_sudo_pass=test
---GNU nano---

$ ansible all -i ./inventory/hosts  --list-hosts
  hosts (1):
    172.17.0.2

$ ansible all -m ping -i ./inventory/hosts
172.17.0.2 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
```

Đối với các thông tin nhạy cảm các bạn có thể đưa vào Ansible Vault.

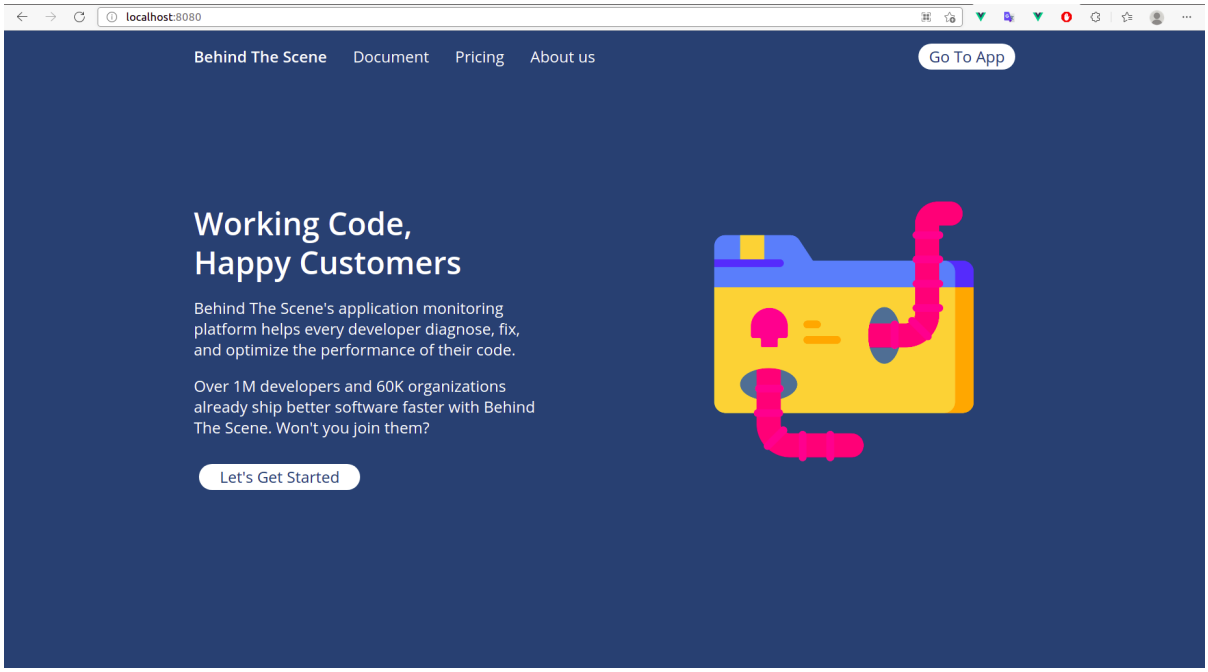
**Bước 3: Tạo các file playbook chứa script cấu hình và thực thi**

Tạo folder `playbooks`, chứa file `setup_and_run.yml`, với nội dung như sau:

```
- hosts:
    - production
  name: Setup and run server
  become: True
  tasks:
    - name: Install the gpg key for nodejs 14
      apt_key:
        url: "https://deb.nodesource.com/gpgkey/nodesource.gpg.key"
        state: present

    - name: Install the nodejs 14 repos
      apt_repository:
        repo: "deb https://deb.nodesource.com/node_14.x {{ ansible_distribution_release }} main"
        state: present
      register: node_repo

    - name: Update apt cache if repo was added.
      become: yes
      apt: update_cache=yes
      when: node_repo.changed
```

```
- name: Install NodeJS
  apt:
    name: nodejs

- name: Check NodeJS version
  command: node -v
  register: node_v

- name: Print NodeJS version
  debug:
    msg: "{{ node_v.stdout }}"

- name: Install yarn
  npm:
    name: yarn
    global: yes

- name: Install git
  apt:
    name: git

- name: Check Git version
  command: git --version
  register: git_v

- name: Print Git version
  debug:
    msg: "{{ git_v.stdout }}"

- name: Git clone
  ignore_errors: yes
  git:
    repo: https://KhanhTranQuoc@bitbucket.org/KhanhQuocTran/bts.git
    dest: bts

- name: Pull Master
  command: chdir=bts/packages/client git pull

- name: Delete package-lock.json
  command: chdir=bts/packages/client rm package-lock.json
  ignore_errors: yes

- name: Install Module
  command: chdir=bts/packages/client yarn

- name: Install pm2
  npm:
    name: pm2
    global: yes

- name: Deploy
  command: chdir=bts/packages/client pm2 --name BTS-Client start yarn -- dev
```

Thực thi playbooks, sau đó có thể truy cập vào http://localhost:8080




## 8. Tổng kết

Ansible là một công cụ đơn giản và rất mạnh mẽ dùng để tự động hóa mọi thứ trong quá trình phát triển, triển khai, quản lí và vận hành phần mềm.

### Tài liệu tham khảo


- https://docs.ansible.com/
- https://www.whizlabs.com/blog/ansible-advantages-and-disadvantages/
- https://blog.risingstack.com/getting-started-with-ansible-infrastructure-automation/
- https://chetna-manku.medium.com/ansible-and-how-nasa-is-using-ansible-ae78838f71ce