DESAFIO ANSIBLE E DOCKER (RESOLUÇÃO)

1 - Na máquina do Ansible HOST, vamos precisar instalar um pacote do docker

execute: **python3 -m pip install docker-py**

```
Requirement already satisfied: docker-py in /home/jose/.local/lib/python3.9/site-packages (1.10.6)
Requirement already satisfied: websocket-client>=0.32.0 in /home/jose/.local/lib/python3.9/site-packag
Requirement already satisfied: docker-pycreds>=0.2.1 in /home/jose/.local/lib/python3.9/site-packages
Requirement already satisfied: six>=1.4.0 in /usr/lib/python3/dist-packages (from docker-py) (1.16.0)
```

2 - Com a máquina docker_vm pronta, vamos configurar no hosts do Ansible seu IP e algumas variáveis:

Com usuário **root** execute: **vim /etc/ansible/hosts**

3 - Após o arquivo hosts do Ansible abrir, crie um grupo chamado docker_vm:

```
[docker_vm]
```

4 - Na linha abaixo do grupo docker_vm, configure o host da docker_vm:

```
[docker_vm]

dockervm ansible_host=192.168.154.130
```

O Nome dockervm é uma identificação para esse host, em um ambiente de produção, poderá ser mais fácil identificar de qual instância se trata o determinado endereço IP

5 - Ainda no documento de hosts do Ansible, pule uma linha e crie um outro grupo de variáveis que serão utilizadas pelo grupo docker_vm:

```
[docker_vm:vars]
```

6 - Configure as seguinte variáveis de ambiente:

```
ansible_user=ubuntu
ansible_ssh_private_key_file=/home/jose/.ssh/id_rsa
```

ansible_user=<usuario_da_dockervm>
ansible_ssh_private_key_file=/home/<usuario_da_dockervm>/.ssh/id_rsa

Essas variáveis serão utilizadas pelo Ansible quando ele se conectar na máquina.

Após isso, salve e feche o documento

7 - Com a chave SSH em mãos da Ansible HOST, vamos copiar a mesma para a docker_vm

execute: **ssh-copy-id -i .ssh/id_rsa.pub <usuario_da_dockervm>@<ip_address>**

```
root@ansible:~# ssh-copy-id -i .ssh/id_rsa.pub senai@192.168.15.132
```

8 - Após copiar a chave, acesse a docker_vm, para esta resolução vai ser necessário o gerenciador de bibliotecas pip do Python:

execute: **apt install python3-pip -y**

9 - Após instalar o pip, instale o módulo docker com o pip:

execute: **python3 -m pip install docker**

10 - Agora, para que o Docker possa ser instalado e gerenciado pelo ansible, é necessário passar a localização do python3 para o grupo de variáveis da docker_vm, para isto, ainda na VM do Docker, execute:

which python3

```
root@ubuntu:/home/ubuntu# which python3
/usr/bin/python3
```

OBS: Copie a saída completa após rodar o comando acima…

11 - Volte para a máquina Ansible HOST

12 - Na máquina Ansible HOST, edite novamente, como usuário root, o arquivo hosts do Ansible:

vim /etc/ansible/hosts

13 - No grupo de variáveis da docker_vm, adicione a seguinte linha:

```
ansible_python_interpreter=/usr/bin/python3
```

14 - O seu arquivo hosts do Ansible, no final, deverá ficar assim:

```
[docker_vm]

dockervm ansible_host=192.168.154.130

[docker_vm:vars]
ansible_user=alunoum
ansible_ssh_private_key_file=/home/alunoum/.ssh/id_rsa
ansible_python_interpreter=/usr/bin/python3
```

OBS: após executar os passos 12 e 13, salve e feche o arquivo

15 - Teste a conexão do Ansible com a docker VM:

execute: **ansible docker_vm -m ping**

```
dockervm | SUCCESS => {
    "changed": false,
    "ping": "pong"
}
```

16 - Após o ping ter sido concluído com sucesso, vamos criar um diretório novo para facilitar a organização

execute: **mkdir docker_vm**

17 - Acesse o diretório docker_vm

18 - No diretório da docker_vm crie um arquivo chamado init-playbook.yaml

execute: **vim init-playbook.yaml**

CONFIGURAÇÃO DO INIT-PLAYBOOK.YAML

OBS: UTILIZE O ESPAÇO AO INVÉS DE TAB PARA IDENTAR O SEU CÓDIGO!!!

1 - Nas cinco primeiras linhas configure da seguinte forma:

```
---
- name: NOME = Sibov
  hosts: docker_vm
  become: yes
  tasks:
```

OBS: ALTERE O SIBOV PARA O SEU NOME

2 - Para a primeira task, vamos atualizar a nossa VM, na próxima linha, configure da seguinte forma:

```
  tasks:
    - name: Atualizar lista de pacotes
      apt:
        update_cache: yes
```

3 - Agora, vamos instalar o GIT, na próxima linha, configure da seguinte forma:

```
    - name: Instalando GIT
      apt:
        name: git
        state: present
```

4 - Agora vamos instalar o Docker, na próxima linha, configure da seguinte forma:

```
    - name: Instalando DOCKER
      apt:
        name: docker.io
        state: present
```

5 - Agora vamos clonar o repositório do sitebike novamente, na próxima linha, configure:

```
  - name: Clonando repositorio GIT
    shell: git clone https://github.com/FofuxoSibov/sitebike.git
```

OBS: Existem outras maneiras de se fazer o clone, por questão de tutorial, vamos utilizar a opção acima.

6 - Agora, com o uso do comando sed vamos localizar e editar um determinado texto dentro do nosso index.html, na próxima linha, configure:

```
- name: Alterando o nome do site
  shell: sed -i 's|WILLIAM MORRIS|Danilo Sibov|g' index.html
  args:
    chdir: sitebike
```

OBS: Usamos o chdir como args (argumento) para apontar em qual diretório esse "step/passo" o Ansible deverá executar, neste caso, é dentro do diretório sitebike

OBS: ALTERE O NOME DANILO SIBOV, PARA O SEU NOME

Para explicar o comando acima, vamos imaginar um mágico

Imagine que você tenha diversas palavras escritas em um papel, basta usar um conjunto de palavras para alterar toda a escrita deste papel, para isso, o mágico fala a primeira palavra: sed

Depois, calmamente, o mágico utiliza uma outra palavra " -i ", o que significa que algo vai ser feito no papel

Depois, ele diz: 's|WILLIAM MORRIS|Danilo Sibov|g' index.html

O "s" é de substituição, o "g" é de global, ou seja, tudo o que corresponder há "WILLIAM MORRIS ", vai ser substituído por "Danilo Sibov", e no final "index.html", seria o papel que toda essa mágica vai alterar.

7 - E para finalizar com chave de "our", vamos criar um Dockerfile com Ansible, simples, rápido e prático, portanto, na próxima e última linha, configure:

```
  - name: Criando um Dockerfile
    copy:
      content: |
        FROM httpd:2.4
        COPY . /usr/local/apache2/htdocs/
      dest: ./sitebike/Dockerfile
```

Estamos utilizando a imagem httpd:2.4, que é um Apache por trás, além disso, estamos copiando tudo que está localizado dentro do sitebike para o diretório /usr/local/apache2/htdocs/ que é onde está localizado o index.html padrão do apache e as configurações do mesmo.

8 - Após todas essas configurações acima, salve e feche o arquivo.

“ Professor, fiquei com preguiça de digitar tudo isso, facilita ae pa nois”

Então aqui vai um facilitador, na máquina Ansible host, dentro do diretório docker_vm, execute:

```
cat <<EOF> init-playbook.yaml
---
- name: NOME = <seu_primeiro_nome>
  hosts: docker_vm
  become: yes
  tasks:
    - name: Atualizar lista de pacotes
      apt:
        update_cache: yes

    - name: Instalando GIT
      apt:
        name: git
        state: present

    - name: Instalando DOCKER
      apt:
        name: docker.io
        state: present

    - name: Clonando repositorio GIT
      shell: git clone https://github.com/FofuxoSibov/sitebike.git

    - name: Alterando o nome do site
      shell: sed -i 's|WILLIAM MORRIS|<seu_primeiro_nome>|g' index.html
      args:
        chdir: sitebike

    - name: Criando um Dockerfile
      copy:
        content: |
          FROM httpd:2.4
          COPY . /usr/local/apache2/htdocs/
        dest: ./sitebike/Dockerfile
EOF
```

Não esqueça de alterar os campos <seu_primeiro_nome>

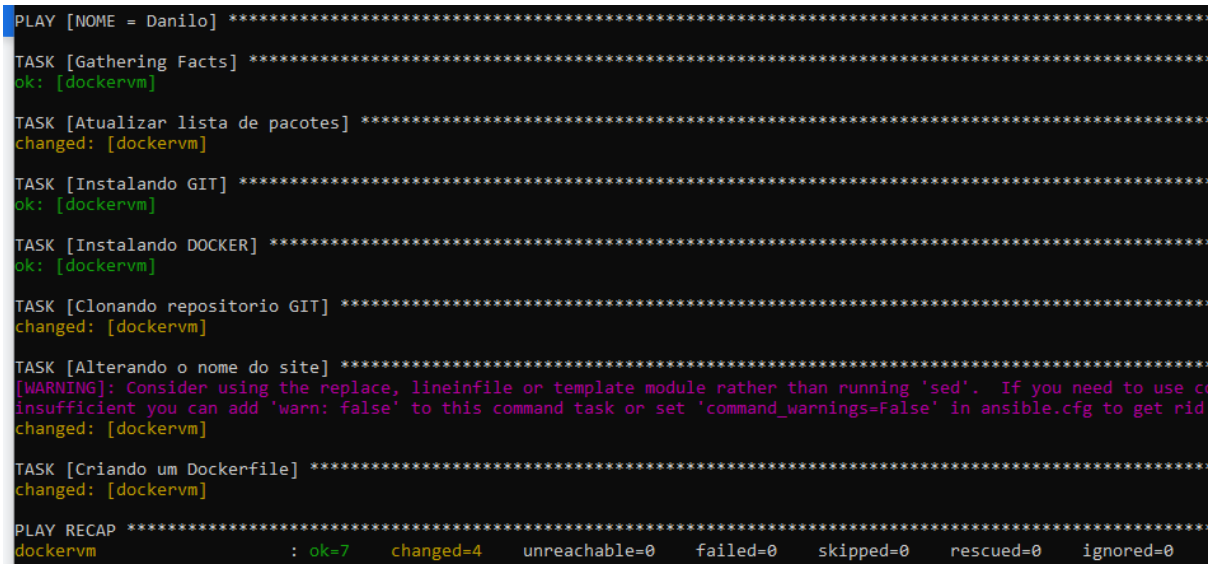
9 - No final de toda essa brincadeira, seu arquivo final ficará parecido com isto:

```yaml
---
- name: NOME = Danilo
  hosts: docker_vm
  become: yes
  tasks:
    - name: Atualizar lista de pacotes
      apt:
        update_cache: yes

    - name: Instalando GIT
      apt:
        name: git
        state: present

    - name: Instalando DOCKER
      apt:
        name: docker.io
        state: present

    - name: Clonando repositorio GIT
      shell: git clone https://github.com/FofuxoSibov/sitebike.git

    - name: Alterando o nome do site
      shell: sed -i 's|WILLIAM MORRIS|Danilo|g' index.html
      args:
        chdir: sitebike

    - name: Criando um Dockerfile
      copy:
        content: |
          FROM httpd:2.4
          COPY . /usr/local/apache2/htdocs/
        dest: ./sitebike/Dockerfile
```

Finalizado configuração do init-playbook.yaml

19 - Agora vamos executar nosso maravilhoso playbook

execute: **ansible-playbook init-playbook.yaml --ask-become-pass**



20 - Se tudo ocorreu bem, como consta na print acima, vamos validar, acesse a docker_vm e execute a série de validações a seguir como usuário root:

- **docker images**
- **docker ps**

```
root@ubuntu:/home/ubuntu# docker ps
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
root@ubuntu:/home/ubuntu#
```

- **git**

```
'git help -a' and 'git help -g' list available subcommands and some
concept guides. See 'git help <command>' or 'git help <concept>'
to read about a specific subcommand or concept.
See 'git help git' for an overview of the system.
root@ubuntu:/home/ubuntu# _
```

- **ls sitebike**

```
root@ubuntu:/home/ubuntu# ls sitebike
contato.html  Dockerfile   img         js              produtos.html  sobre.html
css           favicon.ico  index.html  portfolio.html  README.md
root@ubuntu:/home/ubuntu# _
```

- **cat sitebike/Dockerfile**

```
root@ubuntu:/home/ubuntu# cat sitebike/Dockerfile
FROM httpd:2.4
COPY . /usr/local/apache2/htdocs/
```

- **cat sitebike/index.html | grep “<seu_nome>”**

```
root@ubuntu:/home/ubuntu# cat sitebike/index.html | grep "Danilo"
                                        <cite>Danilo</cite>
                                <cite>Danilo</cite>
root@ubuntu:/home/ubuntu# _
```

21 - Se todas as validações anteriores foram bem-sucedidas, volte para a máquina Ansible host, vamos brincar um pouco com Docker e o seu módulo feito para Ansible

22 - Na Ansible host, dentro do diretório docker_vm, crie um arquivo chamado docker_image-playbook.yaml

execute: vim docker_image-playbook.yaml

23 - Seguindo a mesma lógica de configuração para o arquivo init-playbook.yaml, configure o docker_image-playbook.yaml da seguinte forma:

```yaml
---
- name: Construindo minha imagem docker
  hosts: docker_vm
  become: yes
  tasks:
    - name: Construir imagem Docker
      docker_image:
        name: <seu_nome>
        path: sitebike/

    - name: Iniciar meu container Docker
      shell: docker run -d -p 80:80 --name <seu_nome>-container <nome_da_sua_imagem_do_passo_anterior>
```

Neste outro playbook, nós vamos aproveitar de um dos diversos módulos que o Ansible tem para facilitar integrações, um deles é o docker_image, com ele é possível fazer tag de uma imagem, push para um docker hub por exemplo, construir uma imagem docker, fazer o “download” de uma imagem docker existente e entre outras opções

No primeiro step/passo - Construir imagem Docker, no campo name, definimos qual vai ser o nome da imagem docker a ser construída, no campo path, é a localização do nosso Dockerfile

No próximo step/passo - Iniciar meu container Docker, executamos um módulo **builtIn** (integrado ao Ansible por padrão) chamado shell, que é os comandos do linux por trás, o comando que executamos já é conhecido, é o comando de iniciar/rodar/run um container, o parâmetro -d é para rodar em segundo plano/daemon, o parâmetro -p é para passar a porta do nosso container, neste caso 80:80, o parâmetro --name é para passar o nome do nosso container, e o último campo é qual imagem nós vamos utilizar para subir nosso container

OBS: Não esqueça de substituir os campos entre <seu_nome> e <nome_da_sua_imagem_do_passo_anterior> (Que no caso, vai ser o nome que você escolheu para a imagem ser construída.

Salve e feche o arquivo.

“Professor, estou com preguiça novamente de digitar tudo isso, facilita ae”

Mas é claro:

```
cat <<EOF> docker_image-playbook.yaml
---
- name: Construindo minha imagem docker
  hosts: docker_vm
  become: yes
  tasks:
    - name: Construir imagem Docker
      docker_image:
        name: <seu_nome>
        path: sitebike/

    - name: Iniciar meu container Docker
      shell: docker run -d -p 80:80 --name <seu_nome>-container
<nome_da_sua_imagem_do_passo_anterior>
EOF
```

24 - Após salvar o arquivo ou executar o comando anterior, vamos rodar nosso playbook

execute: **ansible-playbook docker_image-playbook.yaml --ask-become-pass**

```
BECOME password:

PLAY [Construindo minha imagem docker] *****************************************

TASK [Gathering Facts] *********************************************************
ok: [dockervm]

TASK [Construir imagem Docker] *************************************************
[DEPRECATION WARNING]: Param 'path' is deprecated. See the module docs for more information. This feature will be re
Deprecation warnings can be disabled by setting deprecation_warnings=False in ansible.cfg.
[DEPRECATION WARNING]: Please specify build.path instead of path. The path option has been renamed. This feature wil
 Deprecation warnings can be disabled by setting deprecation_warnings=False in ansible.cfg.
[DEPRECATION WARNING]: The value of the "source" option was determined to be "build". Please set the "source" option
community.general 2.0.0. This feature will be removed from community.general in version 2.0.0. Deprecation warnings
 in ansible.cfg.
changed: [dockervm]

TASK [Iniciar meu container Docker] ********************************************
changed: [dockervm]

PLAY RECAP *********************************************************************
dockervm                   : ok=3    changed=2    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
```

25 - Acesse a docker_vm para fazer as seguintes validações com o usuário root:

- **docker images**

```
root@ubuntu:/home/ubuntu# docker images
REPOSITORY   TAG       IMAGE ID       CREATED              SIZE
sibov        latest    c30427c1ca79   About a minute ago   172MB
httpd        2.4       7f6a969e81a5   8 days ago           168MB
```

- **docker ps**

```
root@ubuntu:/home/ubuntu# docker ps
CONTAINER ID   IMAGE     COMMAND              CREATED              STATUS              PORTS
                    NAMES
dcfcb69b09b3   sibov     "httpd-foreground"   About a minute ago   Up About a minute   0.0.0.0:80->8
0/tcp, :::80->80/tcp   sibov-container
```

26 - Agora, se as validações anteriores foram bem sucedidas, agora só correr para o abraço, acesse a página do seu site na WEB

primeiro, pegue o IP do seu servidor, execute: **ip a**

segundo, acesse seu site:

Isso não é magia, é tecnologia!