Demo Project: Ansible and Docker

Project Description

In this project, we will automate the deployment of a Node.js application using **Ansible** and **Docker** on **AWS EC2** instances. The Ansible playbook will:

- Install Docker and Docker Compose.
- Copy Docker Compose files to the server.
- Start the Docker containers.
- Optionally create a new Linux user to run the application.

Step 1: Create AWS EC2 Instance with Terraform

1. Clone the Terraform repository from this repo: https://gitlab.com/twn-devops-projects/ansible/terraform-learn/-/tree/feature/deploy-to-ec2-default-components?ref_type=heads
2. Initialize Terraform: `terraform init`
3. Apply the Terraform configuration: `terraform apply -auto-approve`

4. Copy the **EC2 Public IP** from the Terraform output.

Step 2: Configure Ansible Inventory

Update the file called `hosts` and add the following content:

```
[docker_server]
<your-public-ip> ansible_ssh_private_key_file=~/.ssh/id_rsa ansible_user=ec2
-user
```

Replace `<your-public-ip>` with your AWS EC2 instance's public IP address.

Step 3: Write Ansible Playbook to Deploy Docker Containers

Create `deploy-docker.yaml`

This playbook installs Docker, Docker Compose, and starts the application containers.

```
---
- name: Install Docker
  hosts: docker_server
  become: yes
  tasks:
  - name: Install Docker
    yum:
      name: docker
      update_cache: yes
      state: present
  - name: Start docker daemon
    systemd:
      name: docker
      state: started
```

```
- name: Install Docker-compose
  hosts: docker_server
  tasks:
  - name: Create docker-compose directory
    file:
      path: ~/.docker/cli-plugins
      state: directory
  - name: Get architecture of remote machine
    shell: uname -m
    register: remote_arch
  - name: Install docker-compose
    get_url:
      url: "https://github.com/docker/compose/releases/latest/download/docker-c
      dest: ~/.docker/cli-plugins/docker-compose
      mode: +x

- name: Add ec2-user to docker group
  hosts: docker_server
  become: yes
  tasks:
  - name: Add ec2-user to docker group
    user:
      name: ec2-user
      groups: docker
      append: yes
  - name: Reconnect to server session
    meta: reset_connection

- name: Start docker containers
  hosts: docker_server
  vars_files:
  - project-vars
  tasks:
  - name: Copy docker compose
    copy:
      src: /mnt/c/Users/eduar/devops_projects2/08-ansible/bootcamp-java-mysql
    dest: /home/ec2-user/docker-compose.yaml
  - name: Docker login
    docker_login:
      username: eduardobautistamaciel
      password: "{{docker_password}}"
  - name: Start containers from compose
    community.docker.docker_compose_v2:
      project_src: /home/ec2-user
```

Step 4: Configure Project Variables

Update Create **project-vars** file with:

```
docker_password: my_dockerhub_password
```

Step 5: Docker Compose File

Review the **docker-compose-full.yaml** example content which has the following:

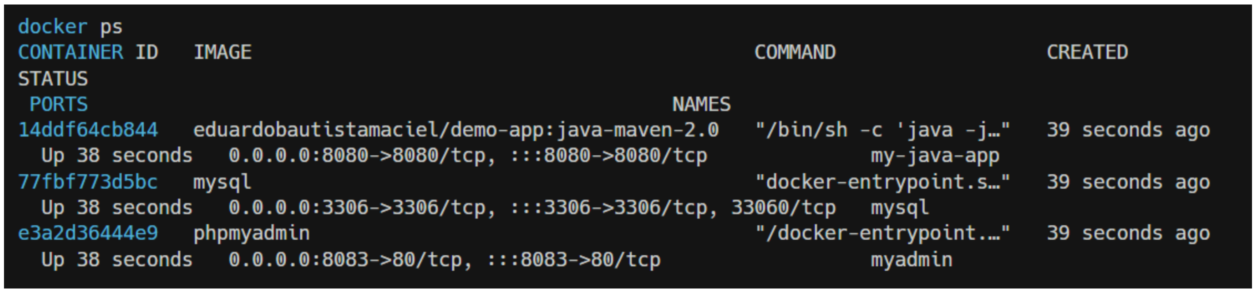
- my-java-app
- mysql
- phpmyadmin (myadmin)

Ref: https://gitlab.com/twn-devops-projects/ansible/bootcamp-java-mysql-project

```
version: '3'
services:
  java-app:
    image: eduardobautistamaciel/demo-app:java-maven-2.0
    environment:
      - DB_USER=user
      - DB_PWD=pass
      - DB_SERVER=mysql
      - DB_NAME=my-app-db
    ports:
    - 8080:8080
    container_name: my-java-app
  mysql:
    image: mysql
    ports:
      - 3306:3306
    environment:
      - MYSQL_ROOT_PASSWORD=my-secret-pw
      - MYSQL_DATABASE=my-app-db
      - MYSQL_USER=user
      - MYSQL_PASSWORD=pass
    volumes:
    - mysql-data:/var/lib/mysql
    container_name: mysql
    # command: --default-authentication-plugin=mysql_native_password
  phpmyadmin:
    image: phpmyadmin
    environment:
      - PMA_HOST=mysql
    ports:
      - 8083:80
    container_name: myadmin
volumes:
  mysql-data:
    driver: local
```

Step 6: Deploy the Application

1. Run the playbook: `ansible-playbook -i hosts deploy-docker.yaml`
2. SSH into the EC2 server: `ssh ec2-user@<your-public-ip>`
3. Verify Docker Containers are running: `docker ps`



```
docker ps
CONTAINER ID   IMAGE                                        COMMAND                  CREATED
STATUS
 PORTS                                                NAMES
14ddf64cb844   eduardobautistamaciel/demo-app:java-maven-2.0   "/bin/sh -c 'java -j…"   39 seconds ago
  Up 38 seconds   0.0.0.0:8080->8080/tcp, :::8080->8080/tcp              my-java-app
77fbf773d5bc   mysql                                        "docker-entrypoint.s…"   39 seconds ago
  Up 38 seconds   0.0.0.0:3306->3306/tcp, :::3306->3306/tcp, 33060/tcp   mysql
e3a2d36444e9   phpmyadmin                                   "/docker-entrypoint.…"   39 seconds ago
  Up 38 seconds   0.0.0.0:8083->80/tcp, :::8083->80/tcp                  myadmin
```

Step 7: Make Playbook Generic

Create **deploy-docker-ec2-new-user.yaml** to execute tasks with a new Linux user:

```
---
- name: Install Docker
  hosts: docker_server
  become: yes
  tasks:
  - name: Install Docker
    yum:
      name: docker
      update_cache: yes
      state: present
  - name: Start docker daemon
    systemd:
      name: docker
      state: started



- name: Create new linux user
  hosts: docker_server
  become: yes
  tasks:
  - name: Create new linux user
    user:
```

```
    name: eduardo
    groups: adm,docker


- name: Install Docker-compose
  hosts: docker_server
  become: yes
  become_user: eduardo
  tasks:
  - name: Create docker-compose directory
    file:
      path: ~/.docker/cli-plugins
      state: directory
  - name: Get architecture of remote machine
    shell: uname -m
    register: remote_arch
  - name: Install docker-compose
    get_url:
      url: "https://github.com/docker/compose/releases/latest/download/docker-co
      dest: ~/.docker/cli-plugins/docker-compose
      mode: +x

- name: Start docker containers
  hosts: docker_server
  become: yes
  become_user: eduardo
  vars_files:
  - project-vars
  tasks:
  - name: Copy docker compose
    copy:
      src: /mnt/c/Users/eduar/devops_projects2/08-ansible/bootcamp-java-mysql-
      dest: /home/eduardo/docker-compose.yaml
  - name: Docker login
    docker_login:
      username: eduardobautistamaciel
      password: "{{docker_password}}"
  - name: Start containers from compose
    community.docker.docker_compose_v2:
      project_src: /home/eduardo
```

Step 8: Clean Up

Destroy the EC2 instance with Terraform:

`terraform destroy --auto-approve`
