**Hosting Student Dashboard with Docker, Ansible & Grafana**

This project focuses on deploying a **Student Dashboard website** on an **AWS EC2 instance** using **Docker Compose**, **Ansible**, and **Grafana**.

**🔹 Key Steps We Performed**

1. **Set up an EC2 instance** with Ubuntu 22.04 and necessary security groups.
2. **Installed required packages** (Docker, Docker Compose, and Ansible).
3. **Created project files** for the Student Dashboard website.
4. **Used Docker Compose** to deploy:
   * An **NGINX-based web server** hosting the Student Dashboard.
   * A **Grafana instance** for monitoring.
5. **Automated deployment using Ansible** to simplify container management.
6. **Configured Grafana with AWS CloudWatch** as a data source for monitoring.

**1️launch and Connect to AWS EC2 Instance**

1. Go to **AWS Console** → **EC2** → **Launch Instance**
   * **OS:** Ubuntu 22.04
   * **Instance Type:** t2.micro (Free Tier)
   * **Security Group:** Open **ports 22 (SSH), 80 (HTTP), 3000 (Grafana)**
2. Connect to the instance

**2️ Install Required Packages**

Run these commands in your EC2 instance:

sudo apt update

sudo apt install -y docker.io docker-compose ansible

Enable Docker:

sudo systemctl enable --now docker

sudo usermod -aG docker ubuntu

newgrp docker

**3️Create Project Directory**

mkdir ~/devops-project

cd ~/devops-project

**4️Create Required Files**

**🔹 Create index.html (Student Dashboard Page)**

mkdir website

nano website/index.html

paste your html file here.

**🔹 Create Dockerfile for Website**

nano website/Dockerfile

Paste this:

FROM nginx:latest

COPY index.html /usr/share/nginx/html/index.html

EXPOSE 80

CMD ["nginx", "-g", "daemon off;"]

**🔹 Create docker-compose.yml**

nano docker-compose.yml

Paste this:

version: '3'

services:

web:

build: ./website

container\_name: student\_dashboard

ports:

- "80:80"

restart: always

grafana:

image: grafana/grafana

container\_name: grafana

ports:

- "3000:3000"

environment:

- GF\_SECURITY\_ADMIN\_PASSWORD=admin

- GF\_SECURITY\_ADMIN\_USER=admin

restart: always

**🔹 Create inventory.ini (Ansible Inventory)**

mkdir ansible

nano ansible/inventory.ini

Paste this:

[servers]

localhost ansible\_connection=local

**🔹 Create deploy.yml (Ansible Playbook)**

nano ansible/deploy.yml

Paste this:

---

- name: Deploy Docker Containers & Grafana

hosts: servers

become: yes

tasks:

- name: Install Required Packages

apt:

name:

- docker.io

- docker-compose

state: present

- name: Start Docker Service

service:

name: docker

state: started

enabled: yes

- name: Deploy Application Using Docker Compose

command: docker-compose up -d

args:

chdir: /home/ubuntu/devops-project

**✅ Key Outcomes of Running the Playbook**

1️**Installed Required Packages**

* Ensured that Docker and Docker Compose were installed on the EC2 instance.
* If they were already installed, Ansible ensured they were up to date.

2️**Started and Enabled Docker Service**

* The playbook started the **Docker service** and ensured it would **restart automatically** if the server reboots.

3️ **Deployed the Student Dashboard & Grafana Using Docker Compose**

* Used docker-compose.yml to start the following containers:
  + **NGINX-based website** (student\_dashboard) running on port **80**.
  + **Grafana monitoring tool** running on port **3000**.

4️**Automated Deployment & Eliminated Manual Steps**

* Without Ansible, we would have to manually SSH into the EC2 instance and run several commands (docker-compose up -d, etc.).
* Ansible **automated** these tasks, making deployment **faster, consistent, and error-free**.

5️**Ensured Reproducibility**

* If the instance is restarted or a new instance is created, we can just run:

ansible-playbook -i ansible/inventory.ini ansible/deploy.yml

* This **quickly provisions the environment** without manual reconfiguration.

**5️ Deploy The Project**

Run the Ansible playbook:

cd ~/devops-project

ansible-playbook -i ansible/inventory.ini ansible/deploy.yml

**6️Verify & Access**

**Check Running Containers**

docker ps

You should see:

bash

CopyEdit

CONTAINER ID IMAGE COMMAND PORTS NAMES

xxxxxxx devops-project\_web "/docker-entrypoint.…" 0.0.0.0:80->80/tcp, :::80->80/tcp student\_dashboard

xxxxxxx grafana/grafana "/run.sh" 0.0.0.0:3000->3000/tcp, :::3000->3000/tcp grafana

**Access The Website**

* **Student Dashboard:** http://<EC2-Public-IP>/
* **Grafana Dashboard:** http://<EC2-Public-IP>:3000/
  + **Username:** admin
  + **Password:** admin

**7️Configure Grafana Data Source**

1. **Login to Grafana**
2. **Go to Configuration → Data Sources → Add Data Source**
3. **Select "CloudWatch" → Enter AWS Credentials**
4. **Save & Test**

**8️Restart After Reboot**

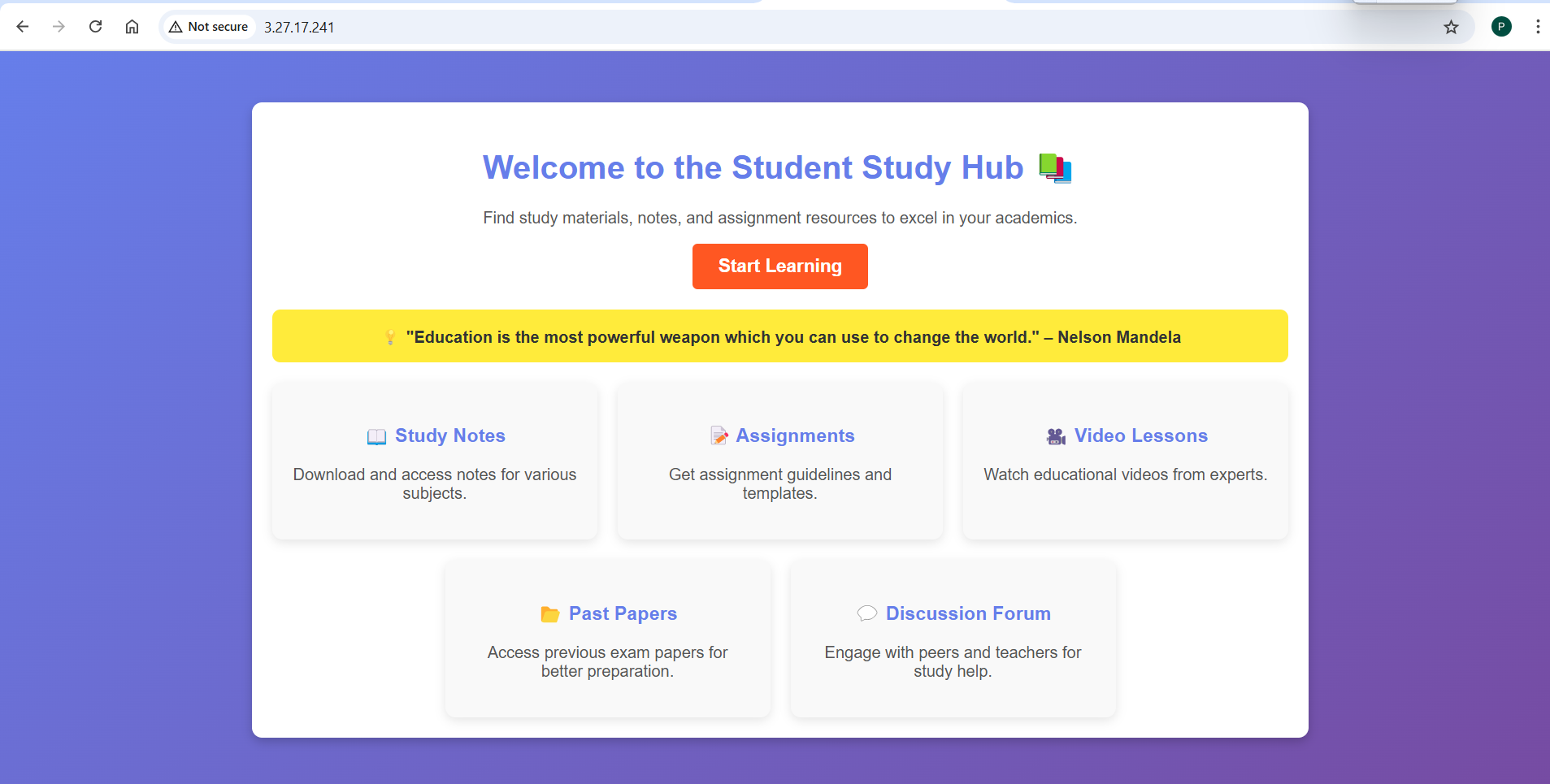
If EC2 restarts, run:

cd ~/devops-project

docker-compose up –d

outputs:

our website dashboard



Grafana enabled with ec2 monitoring:

