🛠️ Elegance E-commerce — Setup & Deployment Guide

This guide walks you through how to run and deploy the **Elegance** e-commerce app — from development to production in the cloud.

🧰 Prerequisites

Before you begin, ensure you have:

- Docker & Docker Compose
- Node.js and npm
- Terraform
- Ansible
- AWS CLI configured (`aws configure`)
- GitHub account & repository access
- GitHub Personal Access Token (for CI/CD if needed)
- Public & private SSH keys for Ansible

🚀 1. Clone the Repository

git clone https://github.com/AdhamGamal/Ecommerce-DevOps-Project.git
cd Ecommerce-DevOps-Project

📦 2. Local Development

🖥️ Frontend

```
cd frontend
npm install
npm run dev
```

Runs on `http://localhost:5173`

⚙️ Backend

```
cd backend
npm install
npm run dev
```

> Runs on `http://localhost:8000`

🐳 3. Docker Containerization

🏗️ Build Docker Images

```
# From project root
docker build -t elegance-frontend ./frontend
docker build -t elegance-backend ./backend
```

☁️ 4. Provision Infrastructure with Terraform

🔐 Set Up AWS Credentials

```
export AWS_ACCESS_KEY_ID=your_key
export AWS_SECRET_ACCESS_KEY=your_secret
```

🌍 Initialize and Apply Terraform

```
terraform init
terraform apply
```

> This provisions EC2 instances, security groups, and other cloud resources.

🤖 5. Configure and Deploy with Ansible

🔧 Setup Ansible Inventory

Update `inventory.ini` with your EC2 instance public IPs:

```
[frontend]
your.frontend.server.ip

[backend]
your.backend1.ip
your.backend2.ip
```

🚀 Deploy with Ansible Playbook

```
ansible-playbook -i inventory.ini playbook.yml
```

🔄 6. CI/CD with GitHub Actions

GitHub Actions will:

- Build Docker images
- Push to Docker Hub
- SSH into EC2 and deploy using Ansible
- Monitor deployment with status checks

Ensure:

- GitHub Secrets are set for:
  - `AWS_ACCESS_KEY_ID`, `AWS_SECRET_ACCESS_KEY`
  - `DOCKER_USERNAME`, `DOCKER_PASSWORD`
  - `SSH_PRIVATE_KEY`, `SSH_HOST`, `SSH_USER`

📊 7. Monitoring

✅ Access Prometheus and Grafana

After deployment:

- **Prometheus**: `http://your-server-ip:9090`
- **Grafana**: `http://your-server-ip:3000`

Use Grafana default credentials:

- Username: `admin`
- Password: `admin`

✅ 8. Final Checklist

- Frontend running at your domain/IP
- Backend API accessible via reverse proxy
- Docker containers running on EC2
- Monitoring accessible via Grafana
- CI/CD pipeline successfully builds and deploys

🤝 Contribution

If you're working as a team, use Git feature branches and open pull requests. Ensure each PR triggers GitHub Actions workflows and passes checks before merging.