DevOps Internship Task – Todo List Node.js Application

📁 Project Overview

This project is a DevOps assessment involving the full lifecycle of containerizing a Node.js Todo List app, automating deployments using CI/CD tools, and deploying it on a remote VM using Docker Compose and Kubernetes (bonus). The original repo used for reference is:

Repositories:

Original Repo: https://github.com/Ankit6098/Todo-List-nodejs

My customized and fully implemented version is here:

My Repo: https://github.com/mezozaki12/Todo-List-nodejs_project

What Was Implemented

Part 1: App Customization + Docker + CI

- Repo Cloning**: Cloned the original Node.js Todo app.
- MongoDB Integration**: Connected the app to a cloud MongoDB instance via a .env file.
- Dockerization:
  - Created a Dockerfile for the Node.js app.
  - Used official MongoDB image directly (no custom image)
- CI Pipeline with GitHub Actions:
  - Built Docker image.
  - Pushed to Docker Hub (used public hub for demonstration).

📂 Files Added:

- .github/workflows/docker-image.yml
- Dockerfile

#**********#

Part 2: VM + Ansible Automation

- Launch an aws Ec2 instance t2.micro with red hat
- Ansible Playbook:
  - Installed docker and docker compose.
  - Copy docker compose file to the ec2.
  - Install git on ec2.
  - install k3s instead of full k8s cuz its light weight and better with single ec2.
  - copy the k3s yaml file to the ec2

📂 Files Added:

- ansible/playbook.yml
- ansible/hosts (inventory)

🔐 SSH from local machine used to run the playbook remotely.

#**********#

Part 3: Docker Compose + Auto-Update

- Docker Compose
  - Created a `docker-compose.yml` file to run the app and MongoDB as services.
  - Configured ports, volume mapping, and health checks.

- Health Check: Added health checks inside the Docker compose file to ensure MongoDB and app are running correctly.

- Auto-Update Tool
  - Used **Watchtower** to monitor Docker Hub.
  - Automatically pulled new image when updated.

📂 Files Added:

- `docker-compose.yml` (includes `app`, `mongo`, and `watchtower` services)

#**********#

Bonus (Part 4): Kubernetes + ArgoCD

- Used Kubernetes (K3s) on the same VM.
- Deployed MongoDB and Node.js App via Kubernetes manifests.

📂 Files Added:

- K3s.yml
  - deployment.yaml
  - service.yaml
  - mongo-deployment.yaml

#**********#

📜 Assumptions & Notes

- Used **Docker Hub** instead of a private registry for simplicity.
- Watchtower was selected due to its simplicity and production usage for auto-updating Docker images.

- Kubernetes installation done using **K3s** for lightweight deployment.
- Used **AWS EC2** (Free Tier) for cloud VM provisioning.
- Secrets and sensitive credentials were excluded from Git.

#**********#

🧪 How to Run the App

🐳 Using Docker Compose

bash:
docker-compose up -d

⚙️ Ansible (Run from Local)
bash:
ansible-playbook -i ansible/hosts ansible/playbook.yml

☸️ Kubernetes
bash:
kubectl apply -f

⚡ **Bonus : ArgoCD**

Argo CD is **not implemented** in this task, due to time/resource constraints.
Argo CD could be added later.

✍️ Author

Moaz Zaki Ismail

GitHub/ GitHub Profile