MEAN CRUD Application Deployment with Docker & CI/CD

Project Overview

This project demonstrates the containerization, deployment, and CI/CD setup of a **MEAN stack application** (MongoDB, Express, Angular, Node.js) using Docker, Docker Compose, and GitHub Actions. The application includes separate **frontend** and **backend** directories.

Repository Setup

1. **Create GitHub Repository**
2. Go to GitHub and create a new repository: `crud-dd-task-mean-app`.
3. **Push Project Code**

```
git init
git remote add origin https://github.com/<your-username>/crud-dd-task-mean-app.git
git add .
git commit -m "Initial commit"
git push -u origin main
```

Dockerization

Backend Dockerfile (`backend/Dockerfile`)

```
FROM node:20-alpine
WORKDIR /usr/src/app
COPY package*.json ./
RUN npm install --only=production
COPY . .
EXPOSE 3000
CMD ["node", "server.js"]
```

Frontend Dockerfile (`frontend/Dockerfile`)

```
FROM node:20-alpine as build
WORKDIR /usr/src/app
```

```
COPY package*.json ./
RUN npm install
COPY . .
RUN npm run build --prod

FROM nginx:alpine
COPY --from=build /usr/src/app/dist/<your-angular-app-name> /usr/share/nginx/
html
EXPOSE 80
CMD ["nginx", "-g", "daemon off;"]
```

Docker Compose (`docker-compose.yml`)

```
version: '3.9'

services:
  mongo:
    image: mongo:7
    container_name: mongo
    restart: always
    ports:
      - "27017:27017"
    environment:
      MONGO_INITDB_ROOT_USERNAME: root
      MONGO_INITDB_ROOT_PASSWORD: example

  backend:
    build: ./backend
    container_name: mean-backend
    restart: always
    ports:
      - "3000:3000"
    environment:
      MONGO_URL: mongodb://root:example@mongo:27017
    depends_on:
      - mongo

  frontend:
    build: ./frontend
    container_name: mean-frontend
    restart: always
    ports:
      - "80:80"
    depends_on:
      - backend
```

CI/CD Pipeline (GitHub Actions)

Create file: `.github/workflows/ci-cd.yml`

```
name: CI/CD Pipeline

on:
  push:
    branches:
      - main

jobs:
  build-and-deploy:
    runs-on: ubuntu-latest

    steps:
    - name: Checkout Code
      uses: actions/checkout@v3

    - name: Set up Docker
      uses: docker/setup-buildx-action@v2

    - name: Log in to Docker Hub
      uses: docker/login-action@v2
      with:
        username: ${{ secrets.DOCKER_USERNAME }}
        password: ${{ secrets.DOCKER_PASSWORD }}

    - name: Build & Push Backend Image
      run: |
        docker build -t <dockerhub-username>/mean-backend:latest ./backend
        docker push <dockerhub-username>/mean-backend:latest

    - name: Build & Push Frontend Image
      run: |
        docker build -t <dockerhub-username>/mean-frontend:latest ./frontend
        docker push <dockerhub-username>/mean-frontend:latest

    - name: SSH and Deploy on VM
      uses: appleboy/ssh-action@v0.1.7
      with:
        host: ${{ secrets.VM_HOST }}
        username: ${{ secrets.VM_USER }}
        key: ${{ secrets.VM_SSH_KEY }}
        script: |
          cd ~/mean-app
          docker-compose pull
          docker-compose up -d
```

Secrets Required on GitHub - `DOCKER_USERNAME` → Docker Hub username - `DOCKER_PASSWORD` → Docker Hub password - `VM_HOST` → Public IP of your VM - `VM_USER` → Username (e.g., ubuntu) - `VM_SSH_KEY` → Private SSH key for VM login

---

Deployment Steps

1. Launch **Ubuntu VM** on AWS/Azure.
2. Install Docker & Docker Compose:

   ```
   sudo apt update
   sudo apt install -y docker.io docker-compose
   sudo systemctl enable --now docker
   ```

3. Clone repository:

   ```
   git clone https://github.com/<your-username>/crud-dd-task-mean-app.git
   cd crud-dd-task-mean-app
   ```

4. Run the application:

   ```
   docker-compose up -d
   ```

5. Access frontend via `http://<VM_IP>`.

Nginx Reverse Proxy (Optional)

```
server {
    listen 80;
    server_name _;

    location /api/ {
        proxy_pass http://localhost:3000/;
```

```
        proxy_http_version 1.1;
        proxy_set_header Upgrade $http_upgrade;
        proxy_set_header Connection 'upgrade';
        proxy_set_header Host $host;
        proxy_cache_bypass $http_upgrade;
    }

    location / {
        root /usr/share/nginx/html;
        index index.html;
        try_files $uri /index.html;
    }
}
```

Screenshots (Include in your repository)

1. GitHub Actions workflow run.
2. Docker image build & push logs.
3. Application running in browser.
4. Docker Compose services on VM.
5. Nginx reverse proxy working.

✅This document ensures **containerized MEAN app deployment with CI/CD using GitHub Actions**.