IT644 Web Services and SOA

Mini CI/CD Pipeline

Name : Jainil Soni

Student ID : 202412107

(1). Overview :

This project demonstrates a complete mini CI/CD pipeline for a Node.js backend service.
It includes automated build & test, Docker containerization, deployment using GitHub Actions, and real-time monitoring with Prometheus + Grafana.

(2). CI/CD Workflow :

CI Stage (Continuous Integration)

On every push to the **`main`** branch :

1. GitHub Actions checks out the code.
2. Installs Node.js dependencies.
3. Runs unit tests.
4. Builds the Docker image for the backend.
5. Pushes the image to GitHub Container Registry.

CD Stage (Continuous Deployment)

Once the image is pushed :

1. GitHub Actions SSHs into the deployment server.
2. Pulls the latest backend image.
3. Uses Docker Compose to restart the backend, Prometheus, and Grafana services.
4. Confirms successful container startup.

Result : Every code push → automatically built, tested, packaged, and deployed.

(3). Docker & Deployment Setup :

Backend Dockerfile :

```
FROM node:18-alpine
WORKDIR /app

COPY package.json .
RUN npm install --only=production

COPY . .
EXPOSE 3000
```

```
CMD ["node", "index.js"]
```

Docker-compose.yml :

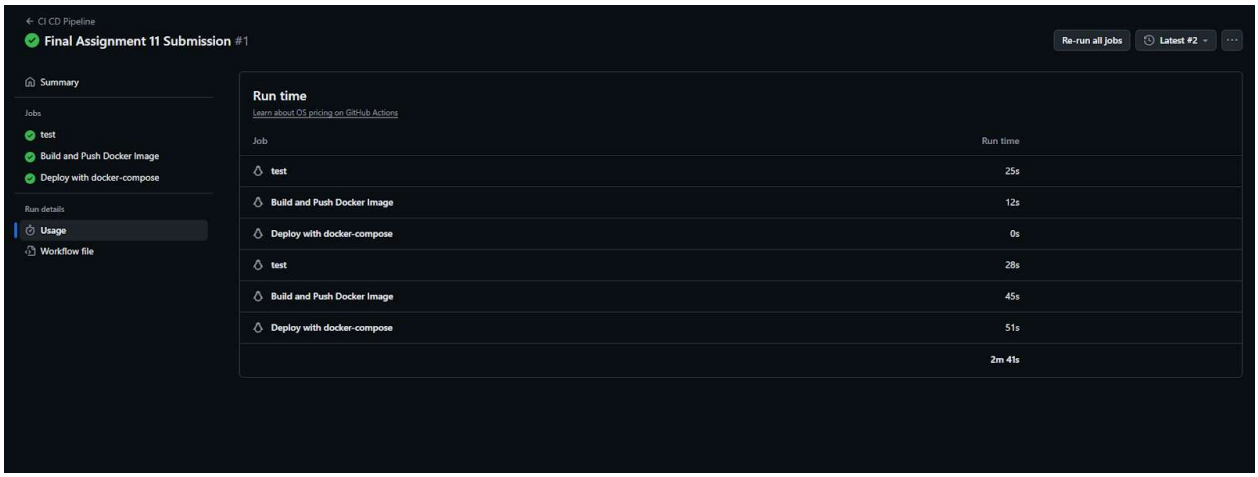
```
services:
  backend:
    build: ./backend
    ports:
      - "3002:3000"
    restart: unless-stopped

  prometheus:
    image: prom/prometheus:latest
    volumes:
      - ./monitoring/prometheus.yml:/etc/prometheus/prometheus.yml:ro
    ports:
      - "9090:9090"
    restart: unless-stopped
    depends_on:
      - backend

  grafana:
    image: grafana/grafana:9.0.0
    environment:
      - GF_SECURITY_ADMIN_PASSWORD=admin
    volumes:
      - ./grafana/provisioning:/etc/grafana/provisioning:ro
      - ./grafana/dashboards:/var/lib/grafana/dashboards:ro
    ports:
      - "3001:3000"
    restart: unless-stopped
    depends_on:
      - prometheus
```

→ Backend runs internally on **3000**, mapped to **3002** on host
→ Prometheus scrapes backend **`/metrics`** endpoint
→ Grafana visualizes Prometheus metrics

Docker Compose Output :

```
[+] Building 4.9s (13/13) FINISHED
 => [internal] load local bake definitions                                                                0.0s
 => => reading from stdin 650B                                                                            0.0s
 => [internal] load build definition from Dockerfile                                                      0.0s
 => => transferring dockerfile: 212B                                                                      0.0s
 => [internal] load metadata for docker.io/library/node:18-alpine                                         2.1s
 => [auth] library/node:pull token for registry-1.docker.io                                               0.0s
 => [internal] load .dockerignore                                                                         0.0s
 => => transferring context: 2B                                                                           0.0s
 => [1/5] FROM docker.io/library/node:18-alpine@sha256:8d6421d663b4c28fd3ebc498332f249011d118945588d0a35cb9bc4b8ca09d9e  0.1s
 => => resolve docker.io/library/node:18-alpine@sha256:8d6421d663b4c28fd3ebc498332f249011d118945588d0a35cb9bc4b8ca09d9e  0.1s
 => [internal] load build context                                                                         0.5s
 => => transferring context: 414.94kB                                                                     0.5s
 => CACHED [2/5] WORKDIR /app                                                                             0.0s
 => CACHED [3/5] COPY package.json .                                                                      0.0s
 => CACHED [4/5] RUN npm install --only=production                                                        0.0s
 => CACHED [5/5] COPY . .                                                                                 0.0s
 => exporting to image                                                                                    1.3s
 => => exporting layers                                                                                   0.0s
 => => exporting manifest sha256:61c1e175dd87bd4045e6c873be65fc6330fc94d72781cea82ce870d0a8d9b7f0         0.0s
 => => exporting config sha256:d33dcbd3dfb8cfa37be70589605c23fb9041b2419c6403f1f6b1175d2ba67740           0.0s
 => => exporting attestation manifest sha256:48d67f3d0b1ffad9fdab107401cd06efdb876a512dc5d643e150493feca046f9  0.1s
 => => exporting manifest list sha256:fcba291aa071f44e9661e19c900bb1088c16b8b1b1536843674eeae32a3cb847    0.0s
 => => naming to docker.io/library/wssoa-lab11-main-backend:latest                                        0.0s
 => => unpacking to docker.io/library/wssoa-lab11-main-backend:latest                                     1.0s
 => resolving provenance for metadata file                                                                0.0s
[+] Running 5/5
 ✓ wssoa-lab11-main-backend                Built                                                          0.0s
 ✓ Network wssoa-lab11_main_default        Created                                                        0.1s
 ✓ Container wssoa-lab11-main-backend-1    Started                                                        1.1s
 ✓ Container wssoa-lab11-main-prometheus-1 Started                                                        1.2s
 ✓ Container wssoa-lab11-main-grafana-1    Started                                                        1.5s
```

GitHub Action CI/CD Successful Run with Test :



(4). Monitoring Setup (Prometheus + Grafana) :

Prometheus

Prometheus pulls metrics from the backend using:

`http://backend:3000/metrics`

Metrics include :

- **`http_requests_total{route="/"}`**
- **`http_requests_total{route="/health"}`**
- **`http_requests_total{route="/metrics"}`**

Grafana Dashboard

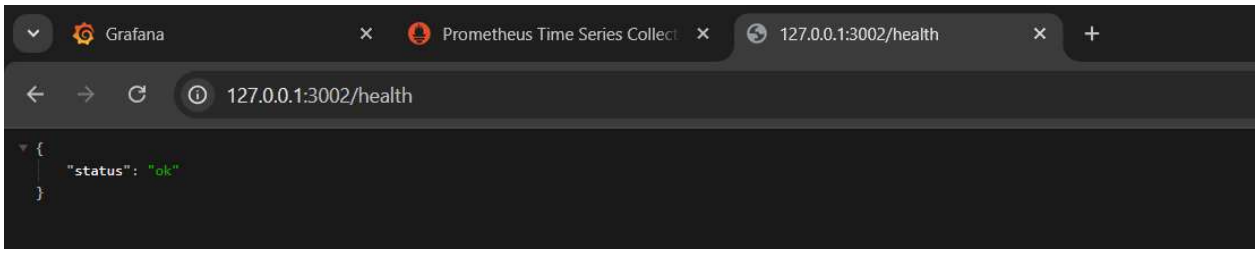
Created a dashboard showing :

- Total HTTP requests per route over time
- Live graphs updating from Prometheus every 15 seconds

Query used :

`sum(http_requests_total) by (route)`

Backend Api Call :



Prometheus Graph :