12/15/2023

**CS-4067**

**DevOps Project**

**FARQULEET FARHAT GONDAL**

**20I-0621**

**CS-B**

[**https://github.com/fafnirLore/DevOps-Project**](https://github.com/fafnirLore/DevOps-Project)

**Table of Contents**

[Phase 1 2](#_Toc153550851)

[Appointments-service Dockerfile 2](#_Toc153550852)

[Doctors-service Dockerfile 2](#_Toc153550853)

[Frontend-service Dockerfile 3](#_Toc153550854)

[Docker-compse file 4](#_Toc153550855)

[Phase 2 5](#_Toc153550856)

[Appointments-service Workflow 5](#_Toc153550857)

[Doctors-service Workflow 5](#_Toc153550858)

[Frontend-service Workflow 6](#_Toc153550859)

[Workflow Runs 7](#_Toc153550860)

[Phase 3 7](#_Toc153550861)

[Appointments-service Deployment & Service 7](#_Toc153550862)

[Doctors-service Deployment & Service 9](#_Toc153550863)

[Frontend-service Deployment & Service 10](#_Toc153550864)

[Kubernetes Dashboard 11](#_Toc153550865)

[Phase 4 12](#_Toc153550866)

[Appointments-service Deployment & Service 12](#_Toc153550867)

[Doctors-service Deployment & Service 13](#_Toc153550868)

[Frontend-service Deployment & Service 14](#_Toc153550869)

# Phase 1

## Appointments-service Dockerfile

# Use the official Python image as the base image

FROM python:3.9-slim

# Set the working directory in the container

WORKDIR /app

# Install Flask and pymongo

RUN pip install --no-cache-dir Flask

# Copy the Flask application code to the working directory

COPY . .

# Expose the port that the Flask application will be running on

EXPOSE 7070

# Set the environment variable for the MongoDB URL

# ENV APPOINTMENTS\_DB\_URL="localhost:27017"

# Run the Flask application

CMD ["python", "app.py"]

Image Link: <https://hub.docker.com/repository/docker/farquleet/appointments-service/general>

## Doctors-service Dockerfile

# Use the official Python image as the base image

FROM python:3.9-slim

# Install MongoDB

# RUN apt-get update && apt-get install -y gnupg2 && \

#     wget -qO - https://www.mongodb.org/static/pgp/server-5.0.asc | apt-key add - && \

#     echo "deb [ arch=amd64,arm64 ] https://repo.mongodb.org/apt/ubuntu focal/mongodb-org/5.0 multiverse" | tee /etc/apt/sources.list.d/mongodb-org-5.0.list && \

#     apt-get update && apt-get install -y mongodb-org && \

#     rm -rf /var/lib/apt/lists/\*

# Set the working directory in the container

WORKDIR /app/doctors

# Install Flask and pymongo

RUN pip install --no-cache-dir Flask

# Copy the Flask application code to the working directory

COPY . .

# Expose the port that the Flask application will be running on

EXPOSE 9090

# Set the environment variable for the MongoDB URL

ENV APPOINTMENTS\_DB\_URL="localhost:27017"

# Run the Flask application

CMD ["python", "app.py"]

Image Link: <https://hub.docker.com/repository/docker/farquleet/doctors-service/general>

## Frontend-service Dockerfile

# Use a base image with Node.js pre-installed

FROM node:14

# Set the working directory inside the container

WORKDIR /app

# Copy package.json and package-lock.json to the working directory

COPY package\*.json ./

# Install dependencies

RUN npm install

# Copy the entire application to the working directory

COPY . .

# Expose the port on which your Node.js application listens

EXPOSE 3000

# Define the command to start your Node.js application

CMD [ "npm", "start" ]

Image Link: <https://hub.docker.com/repository/docker/farquleet/frontend-service/general>

## Docker-compse file

version: "3"

services:

  appointments-service:

    image: farquleet/appointments-service

    # build:

    #   context: ./appointments

    #   dockerfile: Dockerfile

    ports:

      - 7070:7070

    # depends\_on:

    #   - mongo

  doctors-service:

    image: farquleet/doctors-service

    # build:

    #   context: ./doctors

    #   dockerfile: Dockerfile

    ports:

      - 9090:9090

    # depends\_on:

    #   - mongo

  frontend:

    image: farquleet/frontend-service

    # build:

    #   context: ./frontend

    #   dockerfile: Dockerfile

    ports:

      - 3000:3000

    depends\_on:

      - appointments-service

      - doctors-service

    environment:

      - APPOINTMENTS\_SERVICE\_URL=172.20.0.3:7070

      - DOCTORS\_SERVICE\_URL=172.20.0.2:9090

  # mongo:

  #   image: mongo

  #   ports:

  #     - 27017:27017

# Phase 2

## Appointments-service Workflow

name: Appointments Service CI/CD PipeLine

on:

  pull\_request:

    paths:

      - 'appointments/\*\*'

  push:

    branches:

      - main

    paths:

      - 'appointments/\*\*'

jobs:

  build-and-push:

    runs-on: ubuntu-latest

    steps:

      - name: Checkout code

        uses: actions/checkout@v3

      - name: Build and push Docker image

        env:

          DOCKER\_USERNAME: ${{ secrets.DOCKER\_USERNAME }}

          DOCKER\_PASSWORD: ${{ secrets.DOCKER\_PASSWORD }}

        run: |

          docker build -t farquleet/appointments-service:${{ github.sha }} ./appointments

          docker login -u $DOCKER\_USERNAME -p $DOCKER\_PASSWORD

          docker push farquleet/appointments-service:${{ github.sha }}

## Doctors-service Workflow

name: Doctors Service CI/CD PipeLine

on:

  pull\_request:

    paths:

      - 'doctors/\*\*'

  push:

    branches:

      - main

    paths:

      - 'doctors/\*\*'

jobs:

  build-and-push:

    runs-on: ubuntu-latest

    steps:

      - name: Checkout code

        uses: actions/checkout@v3

      - name: Build and push Docker image

        env:

          DOCKER\_USERNAME: ${{ secrets.DOCKER\_USERNAME }}

          DOCKER\_PASSWORD: ${{ secrets.DOCKER\_PASSWORD }}

        run: |

          docker build -t farquleet/doctors-service:${{ github.sha }} ./doctors

          docker login -u $DOCKER\_USERNAME -p $DOCKER\_PASSWORD

          docker push farquleet/doctors-service:${{ github.sha }}

## Frontend-service Workflow

name: Frontend Service CI/CD PipeLine

on:

  pull\_request:

    paths:

      - 'frontend/\*\*'

  push:

    branches:

      - main

    paths:

      - 'frontend/\*\*'

jobs:

  build-and-push:

    runs-on: ubuntu-latest

    steps:

      - name: Checkout code

        uses: actions/checkout@v3

      - name: Build and push Docker image

        env:

          DOCKER\_USERNAME: ${{ secrets.DOCKER\_USERNAME }}

          DOCKER\_PASSWORD: ${{ secrets.DOCKER\_PASSWORD }}

        run: |

          docker build -t farquleet/frontend-service:${{ github.sha }} ./frontend

          docker login -u $DOCKER\_USERNAME -p $DOCKER\_PASSWORD

          docker push farquleet/frontend-service:${{ github.sha }}

## Workflow Runs

A screenshot of a computer

Description automatically generated

# Phase 3

## Appointments-service Deployment & Service

apiVersion: apps/v1

kind: Deployment

metadata:

  name: appointments-deployment

spec:

  replicas: 1

  selector:

    matchLabels:

      app: appointments

  template:

    metadata:

      labels:

        app: appointments

    spec:

      containers:

        - name: appointments

          image: farquleet/appointments-service:latest

          ports:

            - containerPort: 7070

---

apiVersion: v1

kind: Service

metadata:

  name: appointments-service

spec:

  selector:

    app: appointments

  ports:

    - protocol: TCP

      port: 7070

      targetPort: 7070

A computer screen shot of a black screen

Description automatically generated

A black screen with a black background

Description automatically generated

## Doctors-service Deployment & Service

apiVersion: apps/v1

kind: Deployment

metadata:

  name: doctors-deployment

spec:

  replicas: 1

  selector:

    matchLabels:

      app: doctors

  template:

    metadata:

      labels:

        app: doctors

    spec:

      containers:

        - name: doctors

          image: farquleet/doctors-service:latest

          ports:

            - containerPort: 9090

---

apiVersion: v1

kind: Service

metadata:

  name: doctors-service

spec:

  selector:

    app: doctors

  ports:

    - protocol: TCP

      port: 9090

      targetPort: 9090

A screen shot of a computer

Description automatically generated

A black screen with a black background

Description automatically generated

## Frontend-service Deployment & Service

apiVersion: apps/v1

kind: Deployment

metadata:

  name: frontend-deployment

spec:

  replicas: 1

  selector:

    matchLabels:

      app: frontend

  template:

    metadata:

      labels:

        app: frontend

    spec:

      containers:

        - name: frontend

          image: farquleet/frontend-service:latest

          ports:

            - containerPort: 3000

          env:

            - name: APPOINTMENTS\_SERVICE\_URL

              value: "http://appointments-service:7070"

            - name: DOCTORS\_SERVICE\_URL

              value: "http://doctors-service:9090"

---

apiVersion: v1

kind: Service

metadata:

  name: frontend-service

spec:

  selector:

    app: frontend

  ports:

    - protocol: TCP

      port: 3000

      targetPort: 3000

## Kubernetes Dashboard

A screenshot of a black screen

Description automatically generated

A screen shot of a computer

Description automatically generated

# Phase 4

## Appointments-service Deployment & Service

apiVersion: apps/v1

kind: Deployment

metadata:

  name: appointments-deployment

spec:

  replicas: 1

  selector:

    matchLabels:

      app: appointments

  template:

    metadata:

      labels:

        app: appointments

    spec:

      containers:

        - name: appointments

          image: farquleet/appointments-service:latest

          ports:

            - containerPort: 7070

          resources:

            limits:

              cpu: 500m

              memory: 512Mi

            requests:

              cpu: 200m

              memory: 256Mi

          readinessProbe:

            httpGet:

              path: /health

              port: 80

            initialDelaySeconds: 10

            periodSeconds: 5

          livenessProbe:

            httpGet:

              path: /health

              port: 80

            initialDelaySeconds: 15

            periodSeconds: 10

---

apiVersion: v1

kind: Service

metadata:

  name: appointments-service

spec:

  selector:

    app: appointments

  ports:

    - protocol: TCP

      port: 7070

      targetPort: 7070

## Doctors-service Deployment & Service

apiVersion: apps/v1

kind: Deployment

metadata:

  name: doctors-deployment

spec:

  replicas: 1

  selector:

    matchLabels:

      app: doctors

  template:

    metadata:

      labels:

        app: doctors

    spec:

      containers:

        - name: doctors

          image: farquleet/doctors-service:latest

          ports:

            - containerPort: 9090

          resources:

            limits:

              cpu: 500m

              memory: 512Mi

            requests:

              cpu: 200m

              memory: 256Mi

          readinessProbe:

            httpGet:

              path: /health

              port: 80

            initialDelaySeconds: 10

            periodSeconds: 5

          livenessProbe:

            httpGet:

              path: /health

              port: 80

            initialDelaySeconds: 15

            periodSeconds: 10

---

apiVersion: v1

kind: Service

metadata:

  name: doctors-service

spec:

  selector:

    app: doctors

  ports:

    - protocol: TCP

      port: 9090

      targetPort: 9090

## Frontend-service Deployment & Service

apiVersion: apps/v1

kind: Deployment

metadata:

  name: frontend-deployment

spec:

  replicas: 1

  selector:

    matchLabels:

      app: frontend

  template:

    metadata:

      labels:

        app: frontend

    spec:

      containers:

        - name: frontend

          image: farquleet/frontend-service:latest

          ports:

            - containerPort: 3000

          env:

            - name: APPOINTMENTS\_SERVICE\_URL

              value: "http://appointments-service:7070"

            - name: DOCTORS\_SERVICE\_URL

              value: "http://doctors-service:9090"

          resources:

            limits:

              cpu: 500m

              memory: 512Mi

            requests:

              cpu: 200m

              memory: 256Mi

          readinessProbe:

            httpGet:

              path: /health

              port: 80

            initialDelaySeconds: 10

            periodSeconds: 5

          livenessProbe:

            httpGet:

              path: /health

              port: 80

            initialDelaySeconds: 15

            periodSeconds: 10

---

apiVersion: v1

kind: Service

metadata:

  name: frontend-service

spec:

  selector:

    app: frontend

  ports:

    - protocol: TCP

      port: 3000

      targetPort: 3000