



NATIONAL TEXTILE

UNIVERSITY

DEPARTMENT OF COMPUTER SCIENCE

SUBMITTED BY:

Ayema

23-NTU-CS-1142

SECTION SE: 5th(A)

Lab - 14

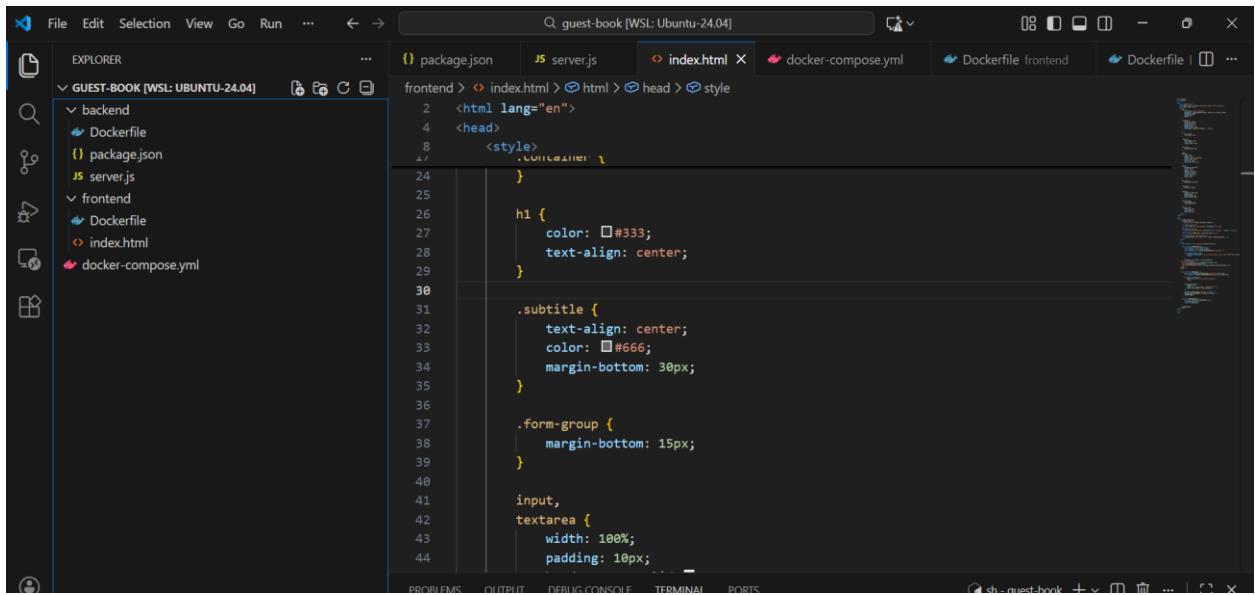
SUBMITTED TO:

Sir Nasir Mehmood

SUBMISSION DATE:

12-28-2025

1. Screenshot of Whole project structure:



2. Output of both image building commands e.g "docker build -t frontend ."

The screenshot shows the VS Code terminal tab titled "zsh - backend" with the following command history and output:

- Entered: `cd backend`
- Output:
 - at 16:07:03
 - [+] Building 6.4s (10/10) FINISHED
 - => [internal] load build definition from Dockerfile
 - => => transferring dockerfile: 170B
 - => [internal] load metadata for docker.io/library/node:lts-alpine3.23
 - => [internal] load .dockerignore
 - => => transferring context: 2B
- Output on the right side (labeled "docker:default"):
 - 0.1s
 - 0.0s
 - 0.3s
 - 0.0s
 - 0.0s

The screenshot shows a terminal window titled "Operating-system-1 [WSL: Ubuntu-24.04]". The terminal is displaying the output of a "docker build" command for a frontend application. The Dockerfile in the background specifies a base image of "nginx:trixie" and copies the "index.html" file from the local directory to the container's "/usr/share/nginx/html" directory. The build process is shown in progress, with various stages like "Building 3.2s" and "Transferring dockerfile: 103B" being completed. The terminal also shows the user pulling the "mongo:7-jammy" image.

```
1142-docker3 > guest-book > frontend > Dockerfile
1 FROM nginx:trixie
2 COPY index.html /usr/share/nginx/html/
3 EXPOSE 80

$ docker build -t frontend .
ayena@DESKTOP-1VWCHS6:~/1142-lab3/Operating-system-1/1142-lab3/Operating-system/Operating-system-1/1142-docker3/guest-book/frontend
● $ docker build -t frontend .
[+] Building 3.2s (7/7) FINISHED
-> [internal] load build definition from Dockerfile
-> => transferring dockerfile: 103B
-> [internal] load metadata for docker.io/library/nginx:trixie
-> [internal] load .dockerignore
-> => transferring context: 2B
-> [internal] load build context
-> => transferring context: 3.77kB
-> [1/2] FROM docker.io/library/nginx:trixie@sha256:fb01117203ff38c2f9af91db1a7409459182a37c87ced5cb442d1d8fcc66d19
-> => resolve docker.io/library/nginx:trixie@sha256:fb01117203ff38c2f9af91db1a7409459182a37c87ced5cb442d1d8fcc66d19
-> [2/2] COPY index.html /usr/share/nginx/html/
-> exporting to image
-> => exporting layers
-> => exporting manifest sha256:fb976433d24ff1ae90cc17d263500cb3bfbe97afaf9a958dfce6bd5df3c
-> => exporting config sha256:ef586718cc758511a3fd30ae60c9677a59bb21ef9a9352ef8712f3aa36b
-> => exporting attestation manifest sha256:1970ca22b9452735af8c6495abfab70aa3d218dd9d8423bb2ad96fc5d50b5a36
-> => exporting manifest list sha256:6243e08bd16d661aa62b030d2096b3302ef12edc6d956118bd8fd9f87d8b70
-> => naming to docker.io/library/frontend:latest
-> => unpacking to docker.io/library/frontend:latest
● $ docker pull mongo:7-jammy
ayena@DESKTOP-1VWCHS6:~/1142-lab3/Operating-system-1/1142-lab3/Operating-system/Operating-system-1/1142-docker3/guest-book/frontend
● $ docker pull mongo:7-jammy

Ln 3, Col 10 Spaces: 4 UTF-8 LF {} Docker Finish Setup
```

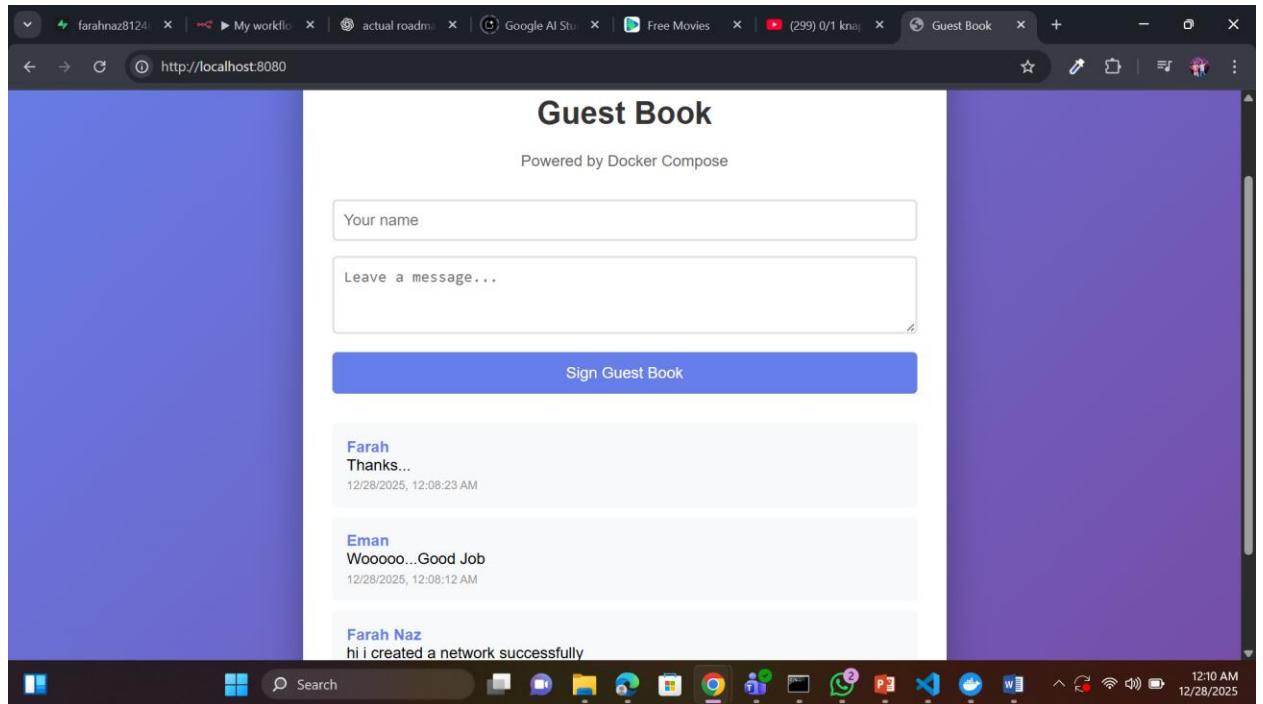
This screenshot is identical to the one above, showing the same terminal session and Docker build output. It displays the build of a "frontend" Docker image from a "Dockerfile" located in the "Operating-system-1" directory. The build process is shown in progress, and the terminal also shows the user pulling the "mongo:7-jammy" image.

```
1142-docker3 > guest-book > frontend > Dockerfile
1 FROM nginx:trixie
2 COPY index.html /usr/share/nginx/html/
3 EXPOSE 80

$ docker build -t frontend .
ayena@DESKTOP-1VWCHS6:~/1142-lab3/Operating-system-1/1142-lab3/Operating-system/Operating-system-1/1142-docker3/guest-book/frontend
● $ docker build -t frontend .
[+] Building 3.2s (7/7) FINISHED
-> [internal] load build definition from Dockerfile
-> => transferring dockerfile: 103B
-> [internal] load metadata for docker.io/library/nginx:trixie
-> [internal] load .dockerignore
-> => transferring context: 2B
-> [internal] load build context
-> => transferring context: 3.77kB
-> [1/2] FROM docker.io/library/nginx:trixie@sha256:fb01117203ff38c2f9af91db1a7409459182a37c87ced5cb442d1d8fcc66d19
-> => resolve docker.io/library/nginx:trixie@sha256:fb01117203ff38c2f9af91db1a7409459182a37c87ced5cb442d1d8fcc66d19
-> [2/2] COPY index.html /usr/share/nginx/html/
-> exporting to image
-> => exporting layers
-> => exporting manifest sha256:fb976433d24ff1ae90cc17d263500cb3bfbe97afaf9a958dfce6bd5df3c
-> => exporting config sha256:ef586718cc758511a3fd30ae60c9677a59bb21ef9a9352ef8712f3aa36b
-> => exporting attestation manifest sha256:1970ca22b9452735af8c6495abfab70aa3d218dd9d8423bb2ad96fc5d50b5a36
-> => exporting manifest list sha256:6243e08bd16d661aa62b030d2096b3302ef12edc6d956118bd8fd9f87d8b70
-> => naming to docker.io/library/frontend:latest
-> => unpacking to docker.io/library/frontend:latest
● $ docker pull mongo:7-jammy
ayena@DESKTOP-1VWCHS6:~/1142-lab3/Operating-system-1/1142-lab3/Operating-system/Operating-system-1/1142-docker3/guest-book/frontend
● $ docker pull mongo:7-jammy

Ln 3, Col 10 Spaces: 4 UTF-8 LF {} Docker Finish Setup
```

3. Webpage at localhost:8080:



4. Copy contents of docker-compose and both Dockerfiles in the pdf.

Docker compose file:

```
services:  
  # MongoDB Database  
  mongodb:  
    image: mongo:7-jammy  
    container_name: guestbook-db  
    networks:  
      - guestbook-network  
    volumes:  
      - mongo_data:/data/db  
  
  # Backend API  
  api:  
    build: ./backend  
    # image: <image_name>  
    container_name: guestbook-api  
    environment:  
      - MONGO_URL=mongodb://mongodb:27017  
    ports:  
      - "3000:3000"  
    depends_on:  
      - mongodb
```

```
networks:
- guestbook-network
# Frontend
web:
  build: ./frontend
  # image: <image_name>
  container_name: guestbook-web
  ports:
  - "8080:80"
  depends_on:
  - api
  networks:
  - guestbook-network
networks:
  guestbook-network:
volumes:
  mongo_data:
```

Docker file of backend:

```
FROM node:lts-alpine3.23
WORKDIR /app
COPY package*.json .
RUN npm install
COPY server.js .
EXPOSE 3000
CMD [ "npm", "start" ]
```

Docker file of frontend:

```
FROM nginx:trixie
COPY index.html /usr/share/nginx/html/
EXPOSE 80
```