



National Textile University

Department of Computer Science

Subject:
Operating System

Submitted to:

Sir Nasir

Submitted by:

Shanza Batool

Reg. number:

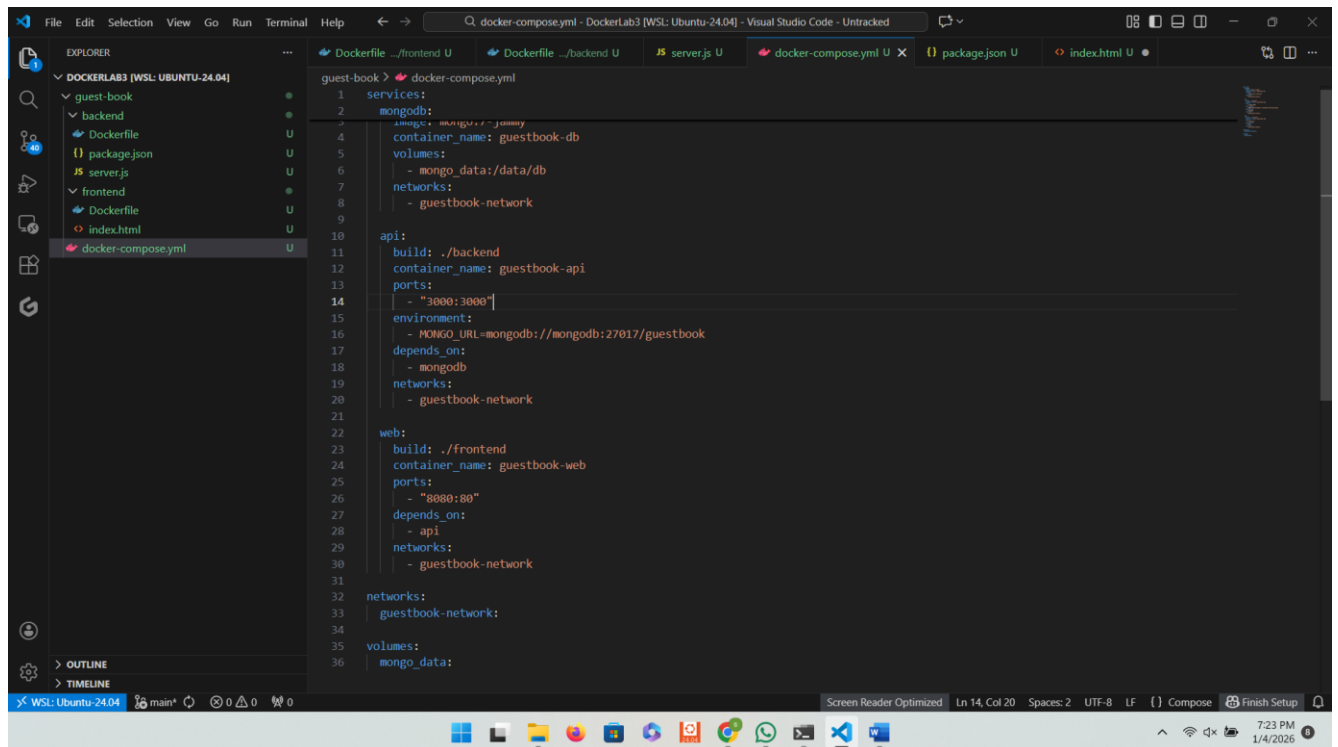
23-NTU-CS-FL-1209

Semester:
5th- A

Docker LAB_3

Task_01: Create networks to facilitate communication between frontend, backend and database.

Code:

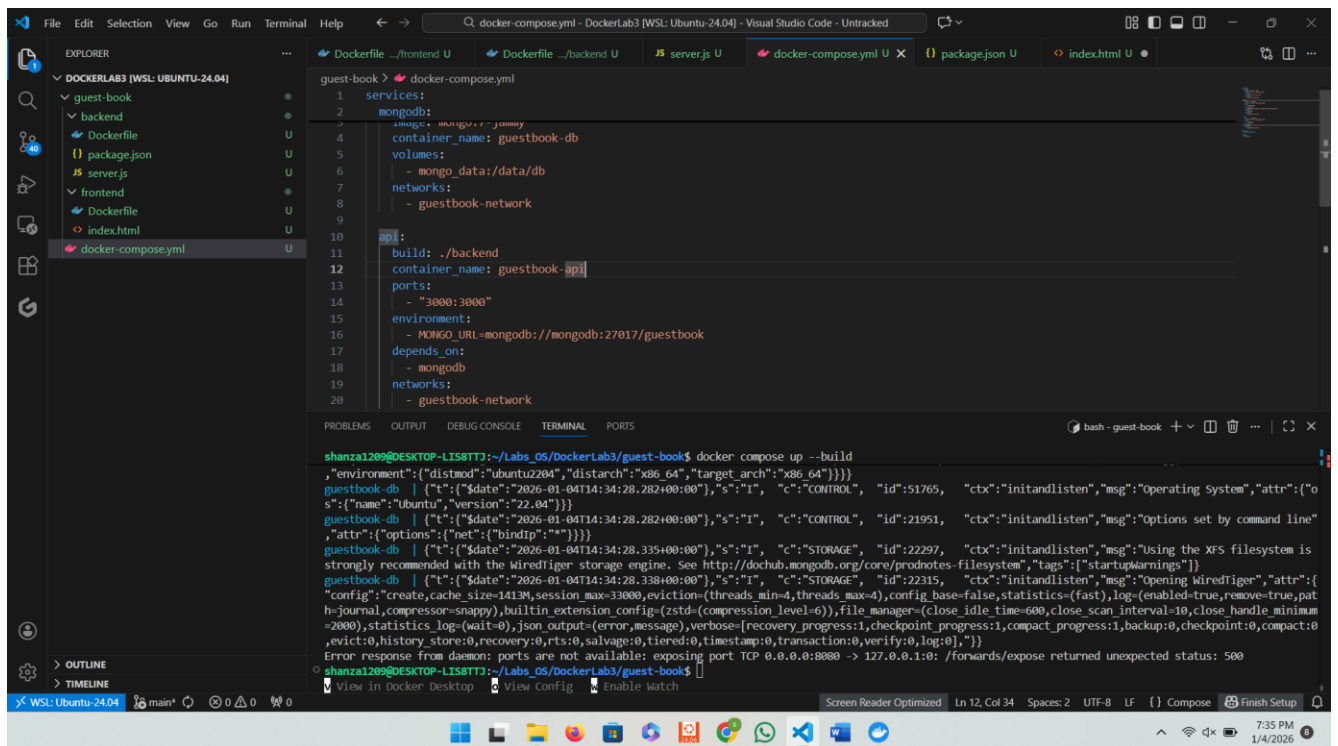


The screenshot shows the Visual Studio Code editor with a Docker Compose file named `docker-compose.yml` open. The file is located in the `guest-book` directory. The Explorer sidebar on the left shows the project structure, including `backend`, `frontend`, `server.js`, `package.json`, `index.html`, and `docker-compose.yml`. The main editor displays the following YAML content:

```
1 services:
2   mongodb:
3     image: mongo:3.6-jessie
4     container_name: guestbook-db
5     volumes:
6       - mongo_data:/data/db
7     networks:
8       - guestbook-network
9
10  api:
11    build: ./backend
12    container_name: guestbook-api
13    ports:
14      - "3000:3000"
15    environment:
16      - MONGO_URL=mongodb://mongodb:27017/guestbook
17    depends_on:
18      - mongodb
19    networks:
20      - guestbook-network
21
22  web:
23    build: ./frontend
24    container_name: guestbook-web
25    ports:
26      - "8080:80"
27    depends_on:
28      - api
29    networks:
30      - guestbook-network
31
32  networks:
33    guestbook-network:
34
35  volumes:
36    mongo_data:
```

The status bar at the bottom indicates the file is in the `main` branch, with 0 changes and 0 errors. The system tray shows the date and time as 7:23 PM on 1/4/2026.

Task_02: Running of “docker compose up –build”

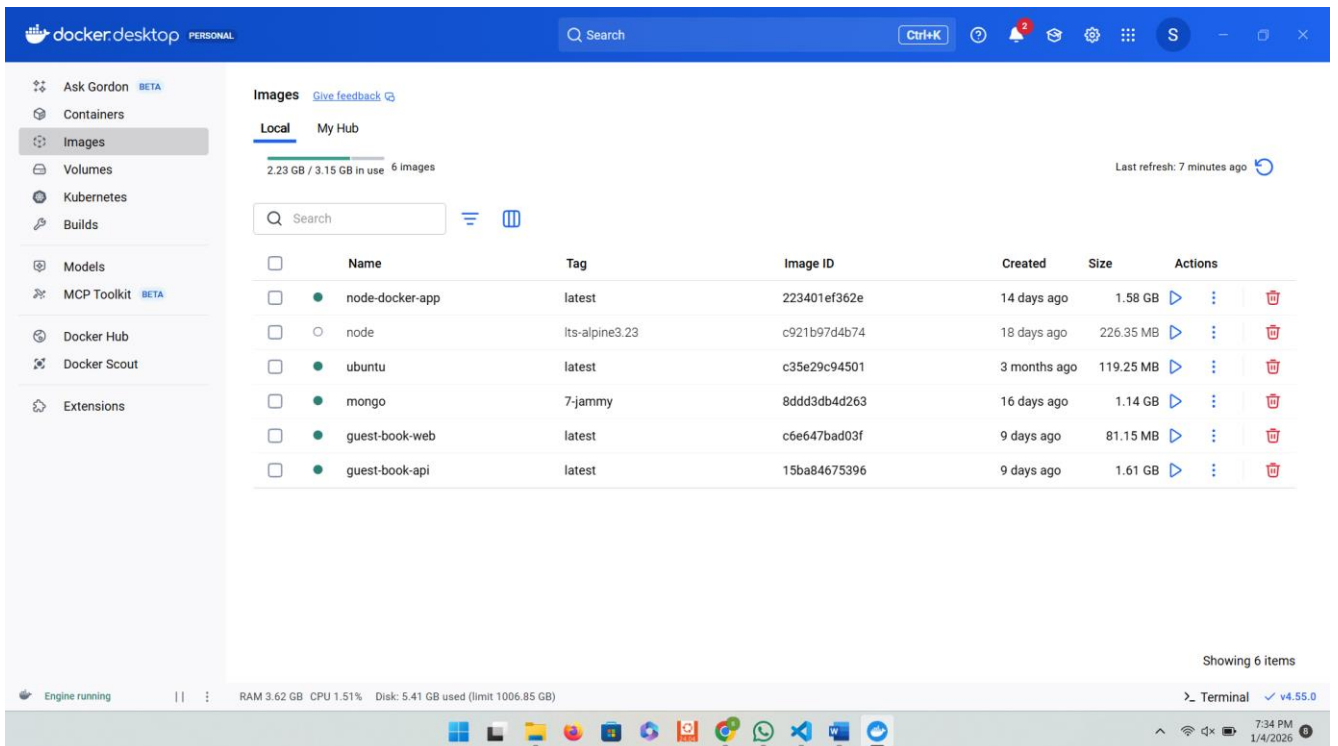


The screenshot shows the Visual Studio Code editor with a Docker Compose file named `docker-compose.yml` open. The file defines two services: `mongodb` and `api`. The `mongodb` service uses the `mongo:3.6` image and mounts a volume `mongo_data`. The `api` service uses the `node:14` image and depends on the `mongodb` service. The terminal output shows the command `docker compose up --build` being executed, which successfully builds and starts the containers. The output also shows the Docker Compose file content and the status of the containers.

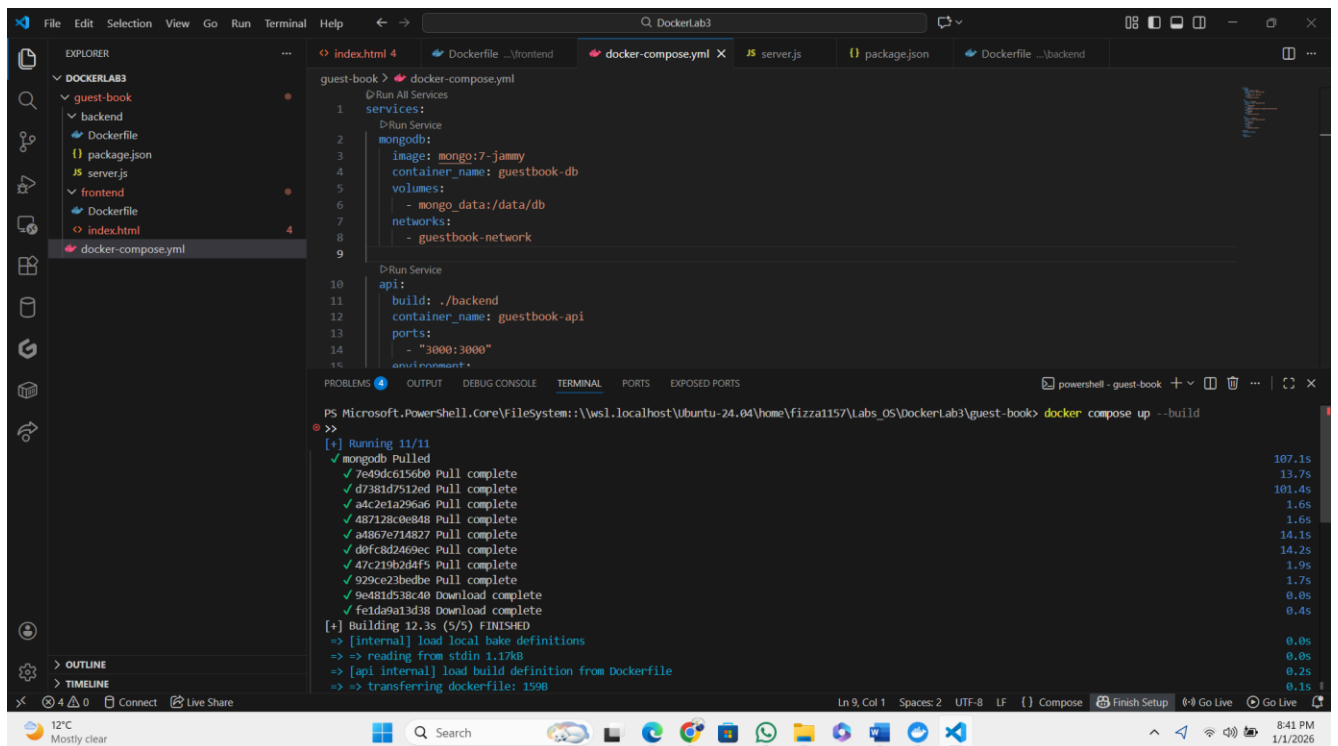
```
1 services:
2   mongodb:
3     image: mongo:3.6
4     container_name: guestbook-db
5     volumes:
6       - mongo_data:/data/db
7     networks:
8       - guestbook-network
9
10  api:
11    build: ./backend
12    container_name: guestbook-api
13    ports:
14      - "3000:3000"
15    environment:
16      - MONGO_URL=mongodb://mongodb:27017/guestbook
17    depends_on:
18      - mongodb
19    networks:
20      - guestbook-network
```

```
shanza1209@DESKTOP-LIS8TTJ:~/Labs_OS/DockerLab3/guest-book$ docker compose up --build
,"environment":{"distmod":"ubuntu2204","distarch":"x86_64","target_arch":"x86_64"}}}}
guestbook-db | {"t":{"$date":"2026-01-04T14:34:28.282+00:00"},"s":"I", "c":"CONTROL", "id":51765, "ctx":"initandlisten","msg":"Operating System","attr":{"o
s":{"name":"Ubuntu","version":"22.04"}}}}
guestbook-db | {"t":{"$date":"2026-01-04T14:34:28.282+00:00"},"s":"I", "c":"CONTROL", "id":21951, "ctx":"initandlisten","msg":"Options set by command line
","attr":{"options":{"net":{"bindip":""}}}}
guestbook-db | {"t":{"$date":"2026-01-04T14:34:28.335+00:00"},"s":"I", "c":"STORAGE", "id":22297, "ctx":"initandlisten","msg":"Using the XFS filesystem is
strongly recommended with the wiredtiger storage engine. See http://dochub.mongodb.org/core/prodnotes-filesystem","tags":["startupwarnings"]}
guestbook-db | {"t":{"$date":"2026-01-04T14:34:28.338+00:00"},"s":"I", "c":"STORAGE", "id":22315, "ctx":"initandlisten","msg":"Opening WiredTiger","attr":{"
config":"create,cache_size=1413M,session_max=33000,eviction-(threads_min=4,threads_max=4),config_base=false,statistics=(fast),log=(enabled=true,remove=true,pat
h=journal,compressor=snappy),builtin_extension_config=(zstd-(compression_level=6)),file_manager=(close_idle_time=600,close_scan_interval=10,close_handle_minimum
=2000),statistics_log=(wait=0),json_output=(error,message),verbose=[recovery_progress:1,checkpoint_progress:1,compact_progress:1,backup:0,checkpoint:0,compact:0
,evict:0,history_store:0,recovery:0,rtts:0,salvage:0,tiered:0,timestamp:0,transaction:0,verify:0,log:0],"}}
Error response from daemon: ports are not available: exposing port TCP 0.0.0.0:8080 -> 127.0.0.1:80 /forwards/expose returned unexpected status: 500
shanza1209@DESKTOP-LIS8TTJ:~/Labs_OS/DockerLab3/guest-book$
```

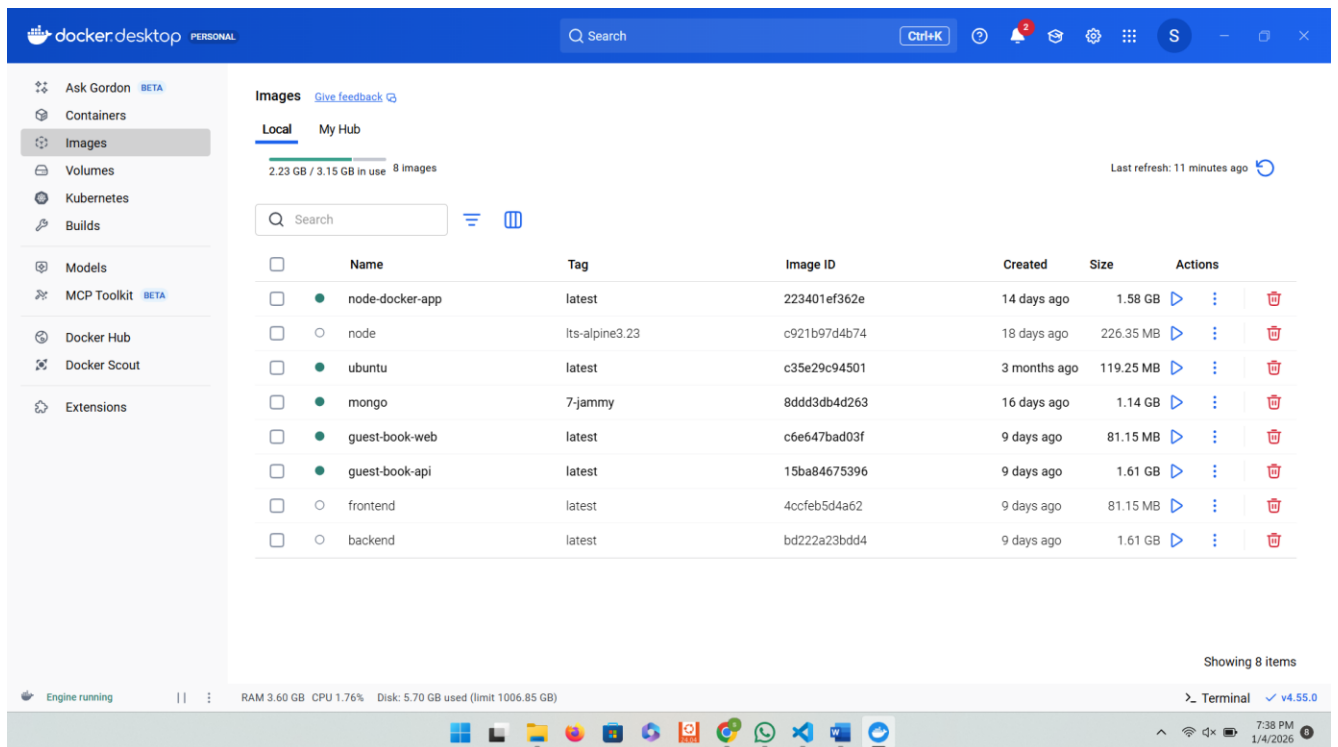
Task_03: Don't forget to create named volume with the same name you've used in dockercompose.yml



Task_04: Screenshot of whole project file

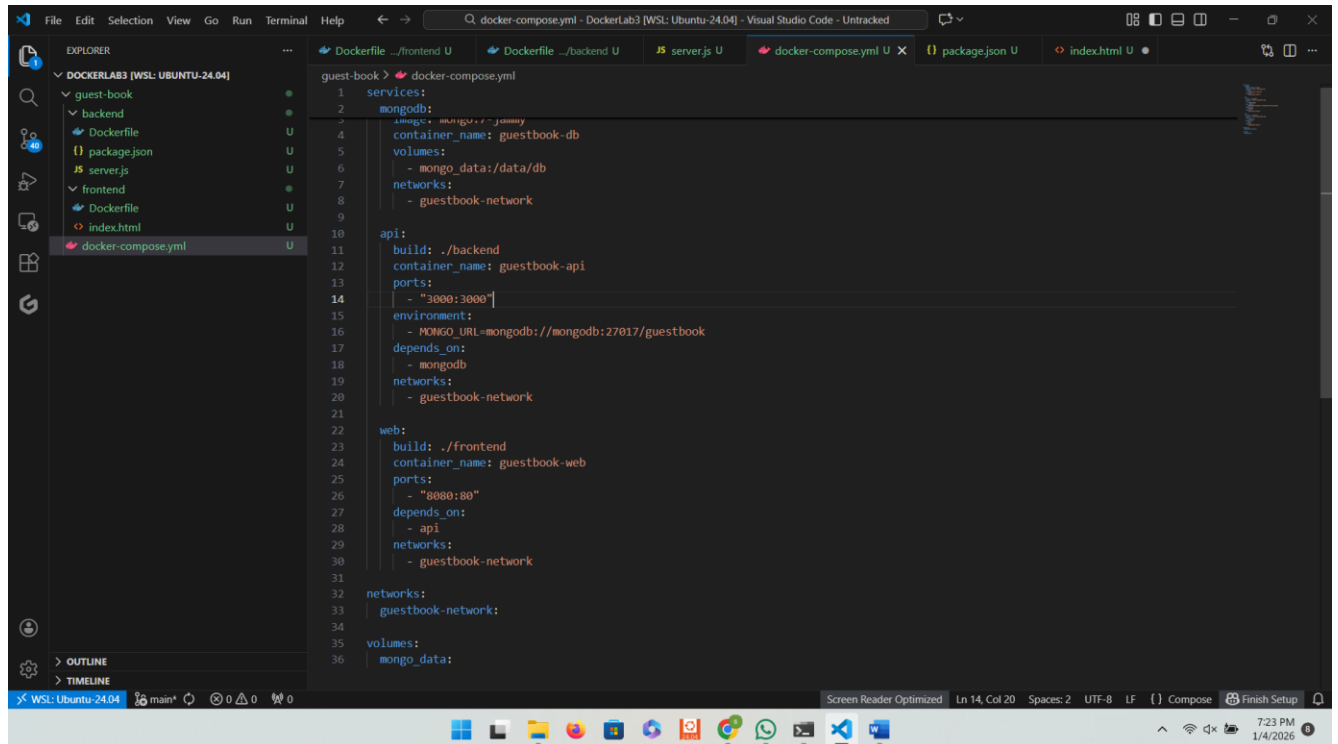


Task_05: Output of both image building commands e.g "docker build -t frontend."



Task_06: Copy contents of docker-compose and both Docker files in the pdf.

Code of docker- compose file:

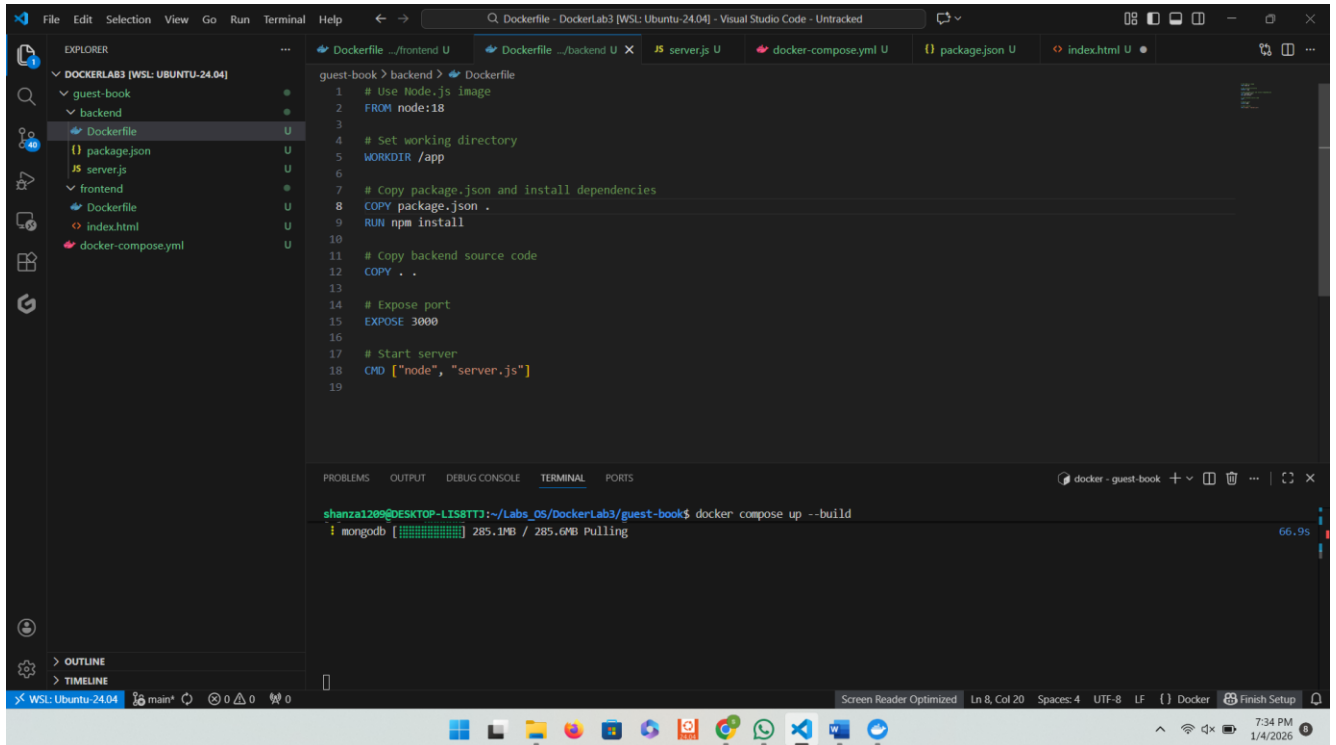


The screenshot shows the Visual Studio Code editor with a Docker Compose file open. The Explorer panel on the left shows the project structure for 'DOCKERLAB3 [WSL: UBUNTU-24.04]'. The file explorer includes 'guest-book', 'backend', 'Dockerfile', 'package.json', 'server.js', 'frontend', 'Dockerfile', 'index.html', and 'docker-compose.yml'. The main editor displays the 'docker-compose.yml' file with the following content:

```
1 services:
2   mongodb:
3     image: mongo:5.0-jammy
4     container_name: guestbook-db
5     volumes:
6       - mongo_data:/data/db
7     networks:
8       - guestbook-network
9
10  api:
11    build: ./backend
12    container_name: guestbook-api
13    ports:
14      - "3000:3000"
15    environment:
16      - MONGO_URL=mongodb://mongodb:27017/guestbook
17    depends_on:
18      - mongodb
19    networks:
20      - guestbook-network
21
22  web:
23    build: ./frontend
24    container_name: guestbook-web
25    ports:
26      - "8080:80"
27    depends_on:
28      - api
29    networks:
30      - guestbook-network
31
32  networks:
33    guestbook-network:
34
35  volumes:
36    mongo_data:
```

The status bar at the bottom indicates the file is 'Ln 14, Col 20' and the encoding is 'UTF-8'. The system tray shows the time as 7:23 PM on 1/4/2026.

Code of docker file of backend:

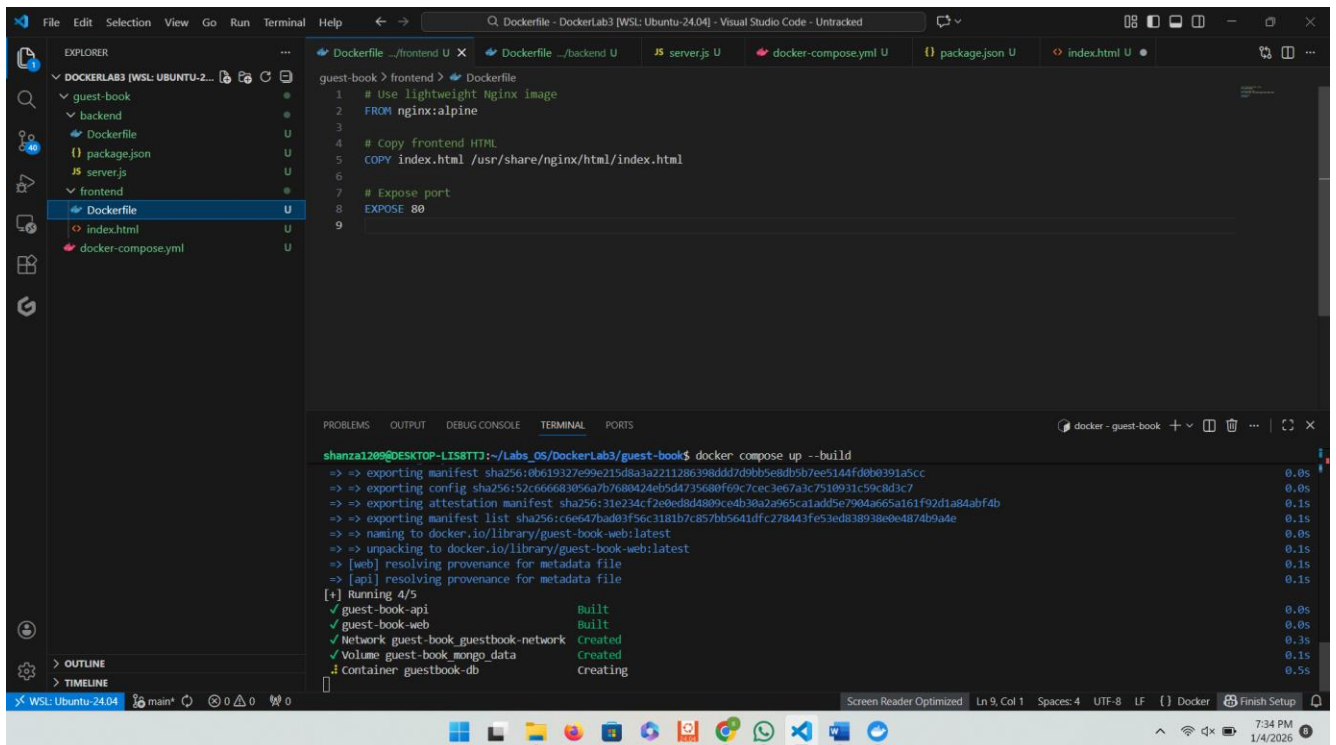


```
1 # Use Node.js image
2 FROM node:18
3
4 # Set working directory
5 WORKDIR /app
6
7 # Copy package.json and install dependencies
8 COPY package.json .
9 RUN npm install
10
11 # Copy backend source code
12 COPY . .
13
14 # Expose port
15 EXPOSE 3000
16
17 # Start server
18 CMD ["node", "server.js"]
19
```

shanzai209@DESKTOP-LIS8TTJ:~/Labs_05/DockerLab3/guest-book\$ docker compose up --build

mongo [██████████] 285.1MB / 285.6MB Pulling 66.9s

Code of docker file of frontend:



```
1 # Use lightweight Nginx image
2 FROM nginx:alpine
3
4 # Copy frontend HTML
5 COPY index.html /usr/share/nginx/html/index.html
6
7 # Expose port
8 EXPOSE 80
9
```

shanzai209@DESKTOP-LIS8TTJ:~/Labs_05/DockerLab3/guest-book\$ docker compose up --build

>=> exporting manifest sha256:0b619327e99e215d8a3a2211286398dd7d9bb5e8db5b7ee5144f0b0391a5cc 0.0s

>=> exporting config sha256:52c666683056a7b7680424eb5d4735680f9c7cec3e67a3c7510931c59c8d3c7 0.0s

>=> exporting attestation manifest sha256:31e234cf2e0ed8d4809cedb30a2a965ca1add5e7904a665a161f92d1a84abf4b 0.1s

>=> exporting manifest list sha256:c6e647bad03f56c3181b7c857bb5641dfc278443fe3ed38938e0e4874b9a4e 0.1s

>=> naming to docker.io/library/guest-book-web:latest 0.0s

>=> unpacking to docker.io/library/guest-book-web:latest 0.1s

> [web] resolving provenance for metadata file 0.1s

> [api] resolving provenance for metadata file 0.1s

[+] Running 4/5

✓ guest-book-api Built 0.0s

✓ guest-book-web Built 0.0s

✓ Network guest-book_guestbook-network Created 0.3s

✓ Volume guest-book_mongo_data Created 0.1s

! Container guestbook-db Creating 0.5s