



National Textile University

Department of Computer Science

Subject: Operating System

Submitted to: Sir Nasir

Submitted by: Amina

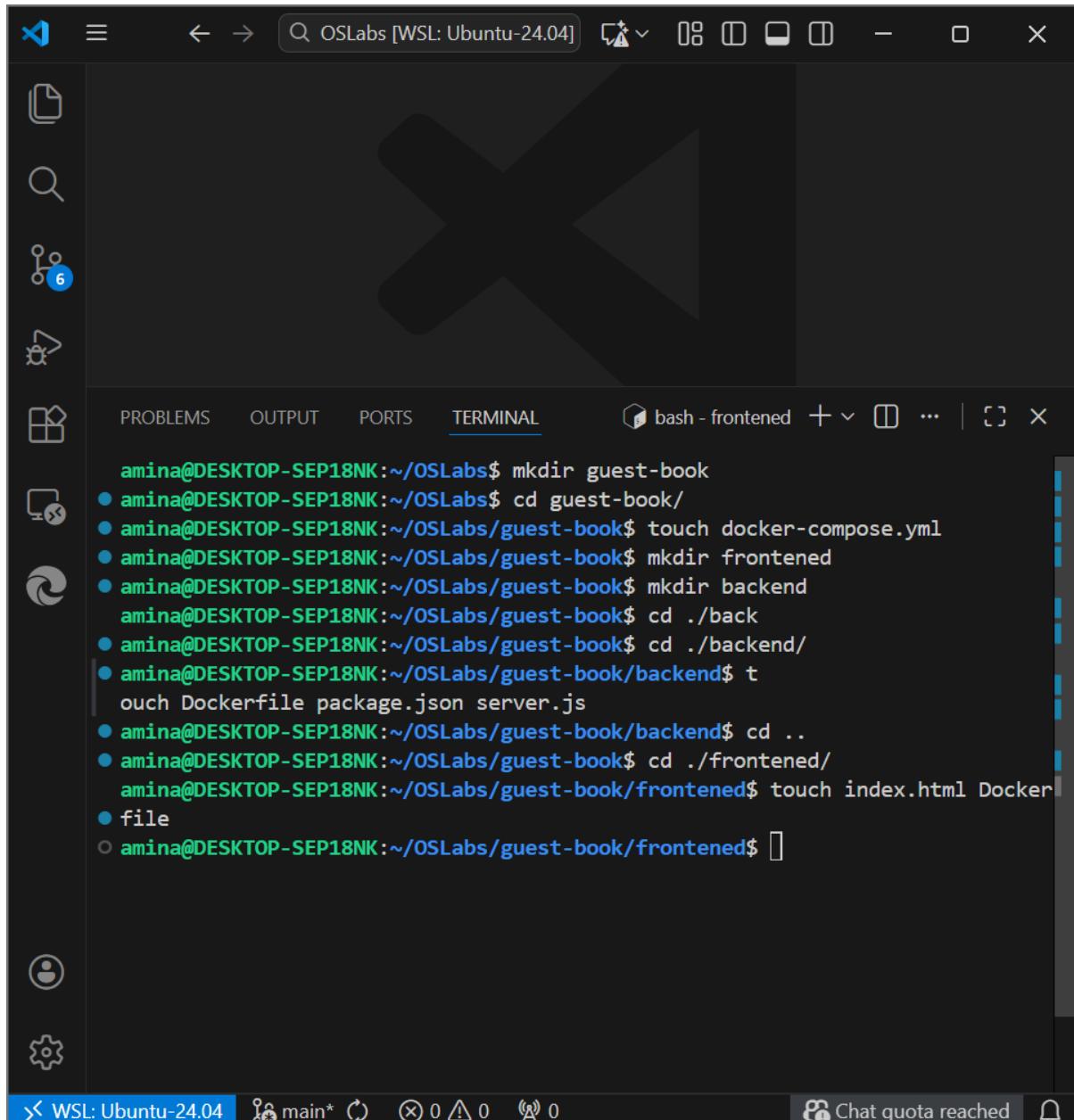
Reg. number: 23-NTU-CS-1136

Docker Lab 03

Semester: 5th

Docker Lab 03

Step 1: Project Structure



The screenshot shows a terminal window in Visual Studio Code (VS Code) running on Windows Subsystem for Linux (WSL). The title bar indicates the session is "OSLabs [WSL: Ubuntu-24.04]". The terminal tab is active, showing a command-line session:

```
amina@DESKTOP-SEP18NK:~/OSLabs$ mkdir guest-book
● amina@DESKTOP-SEP18NK:~/OSLabs$ cd guest-book/
● amina@DESKTOP-SEP18NK:~/OSLabs/guest-book$ touch docker-compose.yml
● amina@DESKTOP-SEP18NK:~/OSLabs/guest-book$ mkdir frontened
● amina@DESKTOP-SEP18NK:~/OSLabs/guest-book$ mkdir backend
    amina@DESKTOP-SEP18NK:~/OSLabs/guest-book$ cd ./back
● amina@DESKTOP-SEP18NK:~/OSLabs/guest-book$ cd ./backend/
● amina@DESKTOP-SEP18NK:~/OSLabs/guest-book/backend$ t
    ouch Dockerfile package.json server.js
● amina@DESKTOP-SEP18NK:~/OSLabs/guest-book/backend$ cd ..
● amina@DESKTOP-SEP18NK:~/OSLabs/guest-book$ cd ./frontened/
    amina@DESKTOP-SEP18NK:~/OSLabs/guest-book/frontened$ touch index.html Docker
● file
○ amina@DESKTOP-SEP18NK:~/OSLabs/guest-book/frontened$
```

The sidebar on the left shows various icons for files, search, and other VS Code features. The bottom status bar shows "WSL: Ubuntu-24.04" and some notification icons.



Codes:

1. Backend

- DockerFile

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

- Package.json

```
{
  "name": "guest-book-backend",
  "version": "1.0.0",
  "type": "module",
  "scripts": {
    "start": "node server.js"
  },
  "dependencies": {
    "express": "^4.18.2",
    "mongoose": "^8.0.0",
    "cors": "^2.8.5"
  }
}
```

- Server.js

```
import express from 'express';
import mongoose from 'mongoose';
import cors from 'cors';
const app = express();
const PORT = 3000;
app.use(cors());
```

```
app.use(express.json());
// MongoDB connection URL
const MONGO_URL = process.env.MONGO_URL ||
'mongodb://mongodb:27017/guestbook';
// Define Mongoose Schema
const messageSchema = new mongoose.Schema({
  name: {
    type: String,
    required: true
  },
  message: {
    type: String,
    required: true
  },
  timestamp: {
    type: Date,
    default: Date.now
  }
});
// Create Mongoose Model
const Message = mongoose.model('Message', messageSchema);
// Connect to MongoDB
async function connectDB() {
try {
  await mongoose.connect(MONGO_URL);
  console.log('Connected to MongoDB with Mongoose');
} catch (error) {
  console.error('MongoDB connection error:', error);
  process.exit(1);
}
// Get all messages
app.get('/api/messages', async (req, res) => {
try {
  const messages = await Message.find()
    .sort({ timestamp:-1 })
    .limit(10);
  res.json(messages);
} catch (error) {
  res.status(500).json({ error: error.message });
}
});
// Post a new message
app.post('/api/messages', async (req, res) => {
try {
  const { name, message } = req.body;
  const newMessage = new Message({
name,
```

```

    message
  });
  await newMessage.save();
  res.status(201).json(newMessage);
} catch (error) {
  res.status(400).json({ error: error.message });
}
});
// Start server
connectDB().then(() => {
app.listen(PORT, () => {
  console.log(`Server running on port ${PORT}`);
});
});

```

2. Frontend

- Dockerfile

```

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

```

- Index.html

```

<!DOCTYPE html>
<html lang="en">

  <head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <title>Guest Book</title>
    <style>
      body {
        font-family: Arial, sans-serif;
        background: linear-gradient(135deg, #667eea 0%, #764ba2 100%);
        min-height: 100vh;
        padding: 20px;
        margin: 0;
      }

      .container {
        max-width: 600px;
        margin: 40px auto;
        background: white;
        padding: 30px;
        border-radius: 10px;
        box-shadow: 0 10px 40px rgba(0, 0, 0, 0.3);
      }
    </style>
  </head>
  <body>
    <div class="container">
      <h1>Guest Book</h1>
      <p>Please leave your guest message below:</p>
      <form>
        <input type="text" placeholder="Name" required>
        <input type="text" placeholder="Email" required>
        <input type="text" placeholder="Message" required>
        <button type="submit">Submit</button>
      </form>
    </div>
  </body>
</html>

```

```
h1 {
  color: #333;
  text-align: center;
}

.subtitle {
  text-align: center;
  color: #666;
  margin-bottom: 30px;
}

.form-group {
  margin-bottom: 15px;
}

input,
textarea {
  width: 100%;
  padding: 10px;
  border: 2px solid #e0e0e0;
  border-radius: 5px;
  font-size: 16px;
  box-sizing: border-box;
}

button {
  background: #667eea;
  color: white;
  border: none;
  padding: 12px 30px;
  font-size: 16px;
  border-radius: 5px;
  cursor: pointer;
  width: 100%;
}

button:hover {
  background: #5568d3;
}

.messages {
  margin-top: 30px;
}

.message {
  background: #f8f9fa;
  padding: 15px;
```

```
        border-radius: 5px;
        margin-bottom: 10px;
    }

    .message-name {
        font-weight: bold;
        color: #667eea;
    }

    .message-time {
        font-size: 12px;
        color: #999;
        margin-top: 5px;
    }
</style>

</head>

<body>
    <div class="container">
        <h1>Guest Book</h1>
        <p class="subtitle">Powered by Docker Compose</p>

        <div class="form-group">
            <input type="text" id="nameInput" placeholder="Your name">
        </div>

        <div class="form-group">
            <textarea id="messageInput" placeholder="Leave a message..." rows="3"></textarea>
        </div>

        <button onclick="addMessage()">Sign Guest Book</button>

        <div class="messages" id="messagesList">
            <p style="text-align: center; color: #999;">Loading messages...</p>
        </div>
    </div>

    <script>
        const API_URL = 'http://localhost:3000/api/messages';

        async function loadMessages() {
            const response = await fetch(API_URL);
            const messages = await response.json();
            const messagesList = document.getElementById('messagesList');

```

```
if (messages.length === 0) {
  messagesList.innerHTML = '<p style="text-align: center; color: #999;">No messages yet. Be the first!</p>';
  return;
}

messagesList.innerHTML = messages.map(msg => `

<div class="message">
  <div class="message-
name">${escapeHtml(msg.name)}</div>
  <div>${escapeHtml(msg.message)}</div>
  <div class="message-time">${new
Date(msg.timestamp).toLocaleString()}</div>
</div>
`).join('`');
}

async function addMessage() {
  const name = document.getElementById('nameInput').value.trim();
  const message =
document.getElementById('messageInput').value.trim();

  if (!name || !message) {
    alert('Please enter both name and message');
    return;
  }

  await fetch(API_URL, {
    method: 'POST',
    headers: { 'Content-Type': 'application/json' },
    body: JSON.stringify({ name, message })
  });

  document.getElementById('nameInput').value = '';
  document.getElementById('messageInput').value = '';
  loadMessages();
}

function escapeHtml(text) {
  const div = document.createElement('div');
  div.textContent = text;
  return div.innerHTML;
}

loadMessages();
</script>
</body>
```

```
</html>
```

3. Docker-compose.yml:

- Create networks to facilitate communication between frontend, backend and database.
- The requirement is that frontend must not be able to call the database container.
- Don't forget to create named volume with the same name you've used in docker-compose.yml

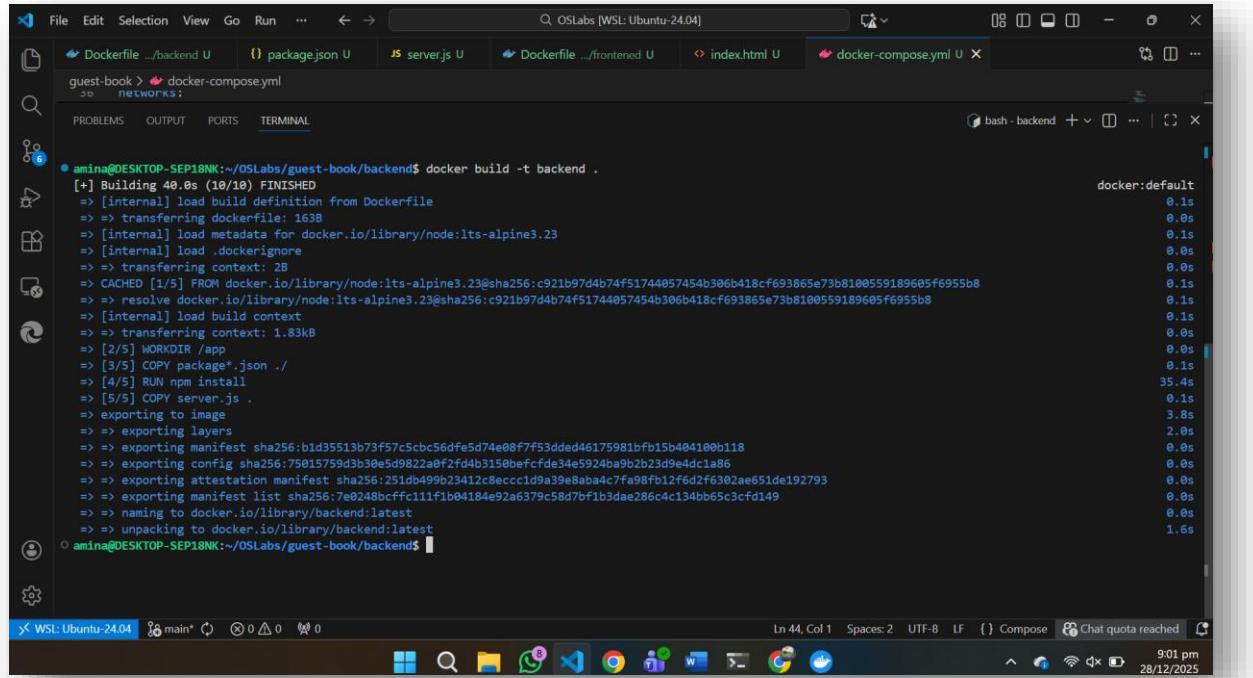
```
services:  
  # MongoDB Database  
  mongodb:  
    image: mongo:7-jammy  
    container_name: guestbook-db  
    networks:  
      - backend_net  
    volumes:  
      - mongo_data:/data/db  
  
  # Backend API  
  api:  
    build: ./backend  
    container_name: guestbook-api  
    environment:  
      - MONGO_URL=mongodb://mongodb:27017  
    ports:  
      - "3000:3000"  
    depends_on:  
      - mongodb  
    networks:  
      - backend_net  
      - frontend_net  
  
  # Frontend  
  web:  
    build: ./frontend  
    container_name: guestbook-web  
    ports:  
      - "8080:80"  
    depends_on:  
      - api  
    networks:  
      - frontend_net  
  
networks:  
  backend_net:  
    driver: bridge  
  frontend_net:  
    driver: bridge
```

```
volumes:  
  mongo_data:
```

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

Commands:

- cd backend
- docker build -t backend .
- cd ..
- cd frontend
- docker build -t frontend .



The screenshot shows a Microsoft Visual Studio Code interface with a terminal window open. The terminal output is as follows:

```
amina@DESKTOP-SEP18NK:~/OSLabs/guest-book/backend$ docker build -t backend .
[+] Building 40.8s (10/18) FINISHED
=> [internal] load build definition from Dockerfile
=> => transferring dockerfile: 163B
=> [internal] load metadata for docker.io/library/node:lts-alpine3.23
=> [internal] load .dockerrcignore
=> => transferring context: 2B
=> CACHED [1/5] FROM docker.io/library/node:lts-alpine3.23@sha256:c921b97d4b74f51744057454b306b418cf693865e73b8100559189605f6955b8
=> => resolve docker.io/library/node:lts-alpine3.23@sha256:c921b97d4b74f51744057454b306b418cf693865e73b8100559189605f6955b8
=> [internal] load build context
=> => transferring context: 1.83kB
=> [2/5] WORKDIR /app
=> [3/5] COPY package*.json .
=> [4/5] RUN npm install
=> [5/5] COPY server.js .
=> exporting to image
=> => exporting layers
=> => exporting manifest sha256:b1d35513b73f57c5cbc56df5d74e08f7f53dded46175981bfb15b404100b118
=> => exporting config sha256:750157593d3b30e5d9822a0f2fd4b3150befcfde34e5924ba9b2b23d9e4dc1a86
=> => exporting attestation manifest sha256:251db499b23412c8ecc11d9a39e8babaa4c7fa98fb12f6d2f6302ae651de192793
=> => exporting manifest list sha256:7e0248bcfffc111f1b04184e92a6379c58d7bf1b3dae286c4c134bb65c3cf149
=> => naming to docker.io/library/backend:latest
=> => unpacking to docker.io/library/backend:latest
```

```
1.6s
● amina@DESKTOP-SEP18NK:~/OSLabs/guest-book/backend$ cd ..
○ amina@DESKTOP-SEP18NK:~/OSLabs/guest-book$ cd f
bash: cd: f: No such file or directory
● amina@DESKTOP-SEP18NK:~/OSLabs/guest-book$ cd ./frontened/
● amina@DESKTOP-SEP18NK:~/OSLabs/guest-book/frontened$ docker build -t frontened .
[+] Building 1.3s (7/7) FINISHED                                            docker:default
=> [internal] load build definition from Dockerfile                      0.1s
=> => transferring dockerfile: 104B                                       0.0s
=> [internal] load metadata for docker.io/library/nginx:trixie           0.1s
=> [internal] load .dockerignore                                         0.1s
=> => transferring context: 2B                                           0.0s
=> [internal] load build context                                         0.0s
=> => transferring context: 3.77kB                                      0.0s
=> [1/2] FROM docker.io/library/nginx@sha256:fb01117203ff38c2          0.3s
=> => resolve docker.io/library/nginx@sha256:fb01117203ff38c2          0.0s
=> [2/2] COPY index.html /usr/share/nginx/html/                           0.1s
=> exporting to image                                                 0.5s
=> => exporting layers                                              0.2s
=> => exporting manifest sha256:af029f765ebbf63b0c159044c976516b5591 0.0s
=> => exporting config sha256:d1d07af36cc90e26607483af534a08c0a2f60c 0.0s
=> => exporting attestation manifest sha256:845d455753a0ecd3d371f67e 0.0s
=> => exporting manifest list sha256:a2a7381e32091feaf5fd8794f6b44b1 0.0s
=> => naming to docker.io/library/frontend:latest                         0.0s
=> => unpacking to docker.io/library/frontend:latest                      0.1s
○ amina@DESKTOP-SEP18NK:~/OSLabs/guest-book/frontened$ █
:Ubuntu-24.04   main* ⌂  ⊗ 0 ⚠ 0  ⓘ 0  LF  { } Compose  Chat quota reached  █
```

Pulling mongo db:

Command:

- Docker pull mongo:7-jammy

```
● amina@DESKTOP-SEP18NK:~/OSLabs/guest-book/frontened$ cd ..
④ amina@DESKTOP-SEP18NK:~/OSLabs/guest-book$ docker pull mongo:7-jammy
Error response from daemon: pull access denied for mongo, repository does no
t exist or may require 'docker login'
● amina@DESKTOP-SEP18NK:~/OSLabs/guest-book$ docker pull mongo:7-jammy
7-jammy: Pulling from library/mongo
a4c2e1a296a6: Pull complete
7e49dc6156b0: Pull complete
d7381d7512ed: Pull complete
487128c0e848: Pull complete
929ce23bedbe: Pull complete
a4867e714827: Pull complete
47c219b2d4f5: Pull complete
d0fc8d2469ec: Pull complete
Digest: sha256:8ddd3db4d2638eb914cce56284e2f0d6daf140bba31679b2af86f7d790a4c
77e
Status: Downloaded newer image for mongo:7-jammy
docker.io/library/mongo:7-jammy
○ amina@DESKTOP-SEP18NK:~/OSLabs/guest-book$
```

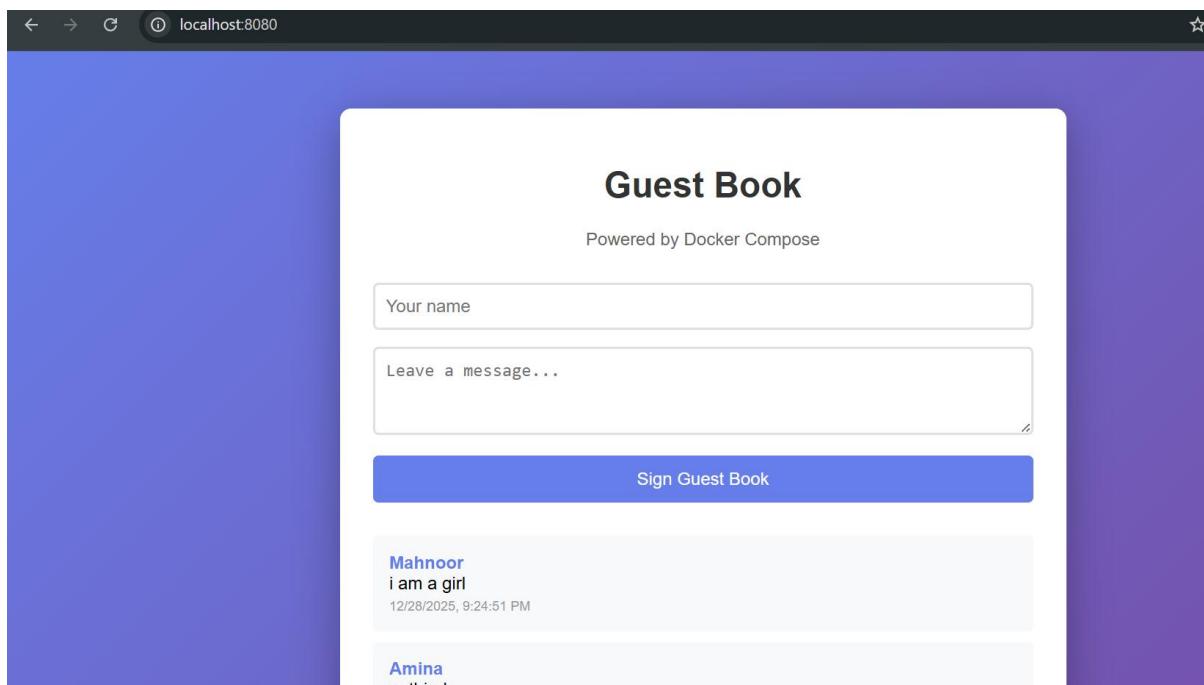
Running the Application

Command:

- docker compose up -d

```
● amina@DESKTOP-SEP18NK:~/OSLabs/guest-book$ ls
backend  docker-compose.yml  frontened
● amina@DESKTOP-SEP18NK:~/OSLabs/guest-book$ docker co
mpose up -d
[+] Building 1.2s (20/20) FINISHED
=> [internal] load local bake definitions      0.0s
=> => reading from stdin 957B                  0.0s
=> [web internal] load build definition from 0.1s
=> => transferring dockerfile: 104B            0.0s
=> [api internal] load build definition from 0.0s
=> => transferring dockerfile: 163B            0.0s
=> [api internal] load metadata for docker.i  0.1s
=> [web internal] load metadata for docker.i  0.1s
=> [api internal] load .dockerignore          0.0s
=> => transferring context: 2B                0.0s
=> [web internal] load .dockerignore          0.1s
=> => transferring context: 2B                0.0s
=> [api 1/5] FROM docker.io/library/node:lts  0.1s
=> => resolve docker.io/library/node:lts-alp  0.1s
=> [api internal] load build context         0.0s
=> => transferring context: 63B              0.0s
=> [web internal] load build context         0.1s
=> => transferring context: 3.77kB          0.0s
=> [web 1/2] FROM docker.io/library/nginx:tr  0.1s
=> => resolve docker.io/library/nginx:trixie  0.1s
```

```
--> CACHED [web 2/2] COPY index.html /usr/share/nginx/html 0.0s
=> [api] exporting to image 0.4s
=> => exporting layers 0.0s
=> => exporting manifest sha256:b484a0915478 0.0s
=> => exporting config sha256:c072a63f5b5415 0.0s
=> => exporting attestation manifest sha256: 0.1s
=> => exporting manifest list sha256:cbf0e6a 0.0s
=> => naming to docker.io/library/guest-book 0.0s
=> => unpacking to docker.io/library/guest-book-b 0.0s
=> [web] exporting to image 0.4s
=> => exporting layers 0.0s
=> => exporting manifest sha256:704c9df55fea 0.0s
=> => exporting config sha256:aaa4b0b9b4bc13 0.0s
=> => exporting attestation manifest sha256: 0.1s
=> => exporting manifest list sha256:778e7db 0.0s
=> => naming to docker.io/library/guest-book 0.0s
=> => unpacking to docker.io/library/guest-book-b 0.0s
=> [web] resolving provenance for metadata f 0.0s
=> [api] resolving provenance for metadata f 0.0s
[+] Running 8/8
✓ guest-book-api          Built          0.0s
✓ guest-book-web           Built          0.0s
✓ Network guest-book_frontend_net Created  0.1s
✓ Network guest-book_backend_net Created  0.1s
✓ Volume guest-book_mongo_data Created  0.0s
✓ Container guestbook-db    Started       1.2s
✓ Container guestbook-api   Starting     2.0s
✓ Container guestbook-web   Starting     2.5s
○ amina@DESKTOP-SEP18NK:~/OSLabs/guest-book$
```



<input type="checkbox"/>	<input checked="" type="radio"/>	tender_wing	5416125fc8ce	my-server	3000:3000	0%	9 days ago				
<input type="checkbox"/>	<input checked="" type="radio"/>	ubuntu_container	c68f1177ae5c	ubuntu:resc	-	0%	16 days ago				
<input type="checkbox"/>			guest-book	-	-	1.76%	4 minutes ago				

Showing 3 items