

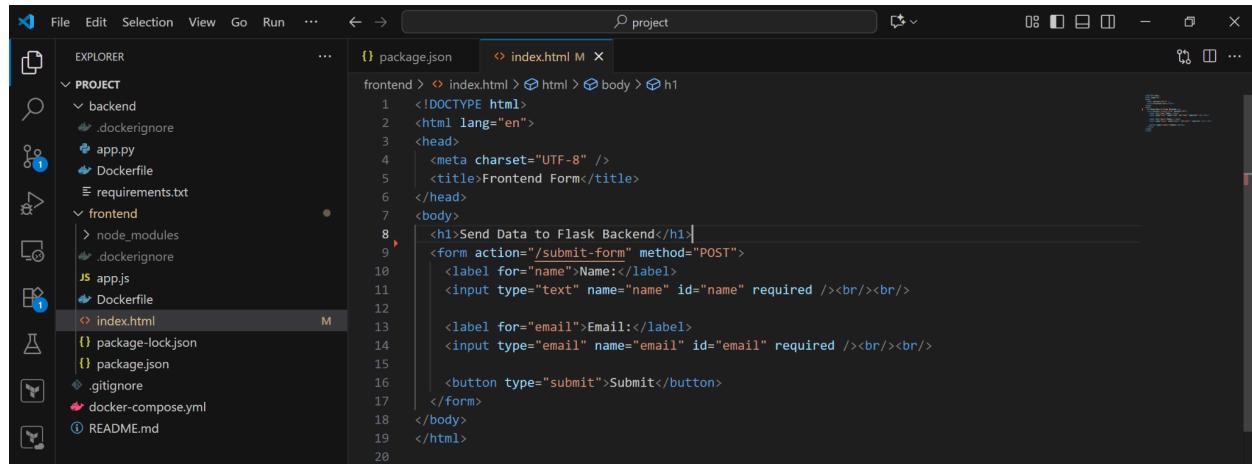
## 1. Frontend (Node.js with Express):

- Create a frontend using Express and Node.js.
- Include a form similar to the one from the Flask Assignment 2.
- Configure the form to send a request to the Flask backend.
- **Backend (Flask):**
- Use the Flask backend to handle the form submission and process the data.
- **Folder Structure:**
- Organize the project with separate folders for the frontend and backend.
- **Docker Configuration:**
- Create a `Dockerfile` for both the frontend and backend.
- Write a `.yaml` file (Docker Compose) to connect both services in the same network.
- Upload both images to docker hub and push your whole code to github and add the `node_modules` and other non required files(`.vscode`) in `.gitignore`

### Frontend (Node.js with Express):

Create a frontend using Express and [Node.js](#).

index.html(form)

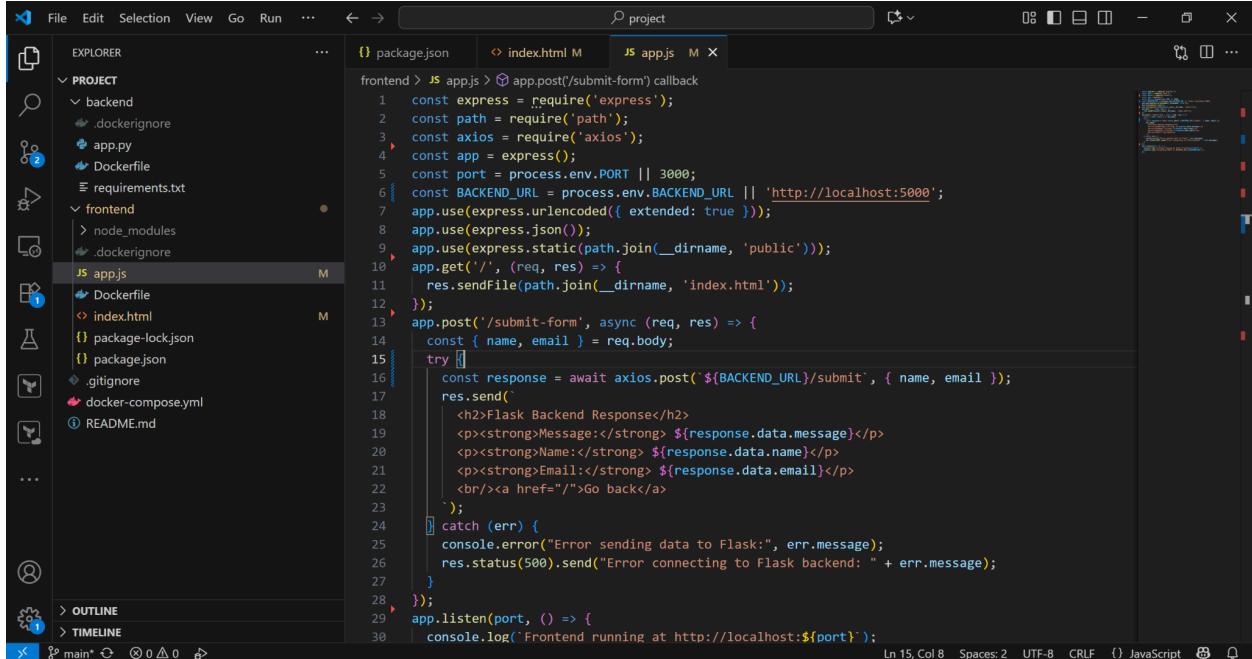


The screenshot shows the Visual Studio Code interface with the following details:

- EXPLORER:** Shows the project structure:
  - PROJECT
  - backend
  - frontend
    - node\_modules
    - dockerignore
    - app.py
    - Dockerfile
    - requirements.txt
- EDITOR:** The `index.html` file is open in the editor tab.

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8" />
<title>Frontend Form</title>
</head>
<body>
<h1>Send Data to Flask Backend</h1>
<form action="/submit-form" method="POST">
<label for="name">Name:</label>
<input type="text" name="name" id="name" required /><br/><br/>
<label for="email">Email:</label>
<input type="email" name="email" id="email" required /><br/><br/>
<button type="submit">Submit</button>
</form>
</body>
</html>
```

## Express app ([app.js](#))



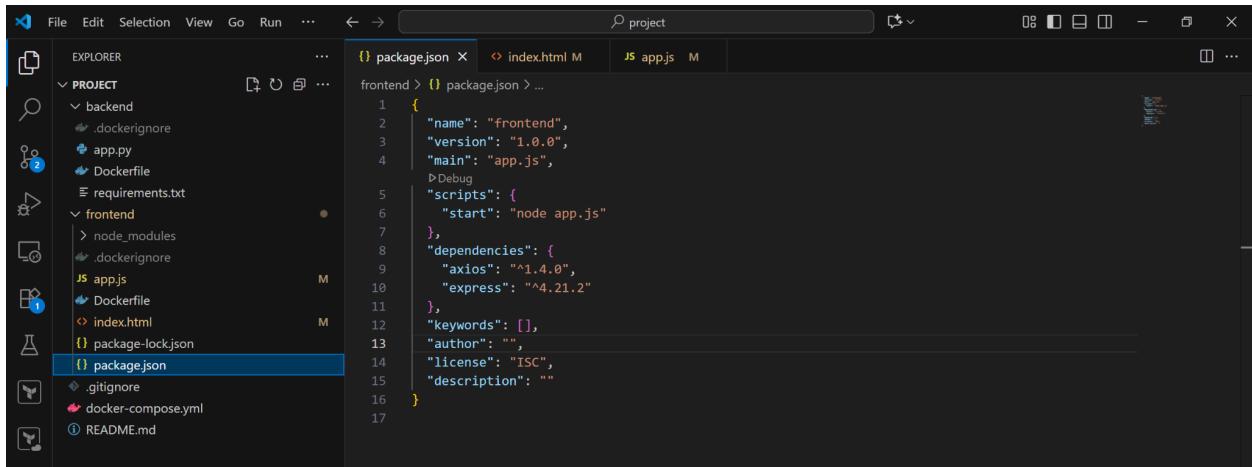
The screenshot shows the VS Code interface with the following details:

- Explorer View:** Shows the project structure with files like `backend`, `frontend`, `requirements.txt`, `Dockerfile`, `index.html`, `package-lock.json`, `package.json`, `.gitignore`, and `README.md`.
- Editor View:** The `app.js` file is open, displaying the following code:

```
const express = require('express');
const path = require('path');
const axios = require('axios');
const app = express();
const port = process.env.PORT || 3000;
const BACKEND_URL = process.env.BACKEND_URL || 'http://localhost:5000';
app.use(express.urlencoded({ extended: true }));
app.use(express.json());
app.use(express.static(path.join(__dirname, 'public')));
app.get('/', (req, res) => {
  res.sendFile(path.join(__dirname, 'index.html'));
});
app.post('/submit-form', async (req, res) => {
  const { name, email } = req.body;
  try {
    const response = await axios.post(`${BACKEND_URL}/submit`, { name, email });
    res.send(`
      <h2>Flask Backend Response</h2>
      <p><strong>Message:</strong> ${response.data.message}</p>
      <p><strong>Name:</strong> ${response.data.name}</p>
      <p><strong>Email:</strong> ${response.data.email}</p>
      <br/><a href="/">Go back</a>
    `);
  } catch (err) {
    console.error("Error sending data to Flask:", err.message);
    res.status(500).send("Error connecting to Flask backend: " + err.message);
  }
});
app.listen(port, () => {
  console.log(`Frontend running at http://localhost:${port}`);
});
```

The status bar at the bottom indicates: Ln 15, Col 8 Spaces: 2 UTF-8 CRLF {} JavaScript

## Package.json



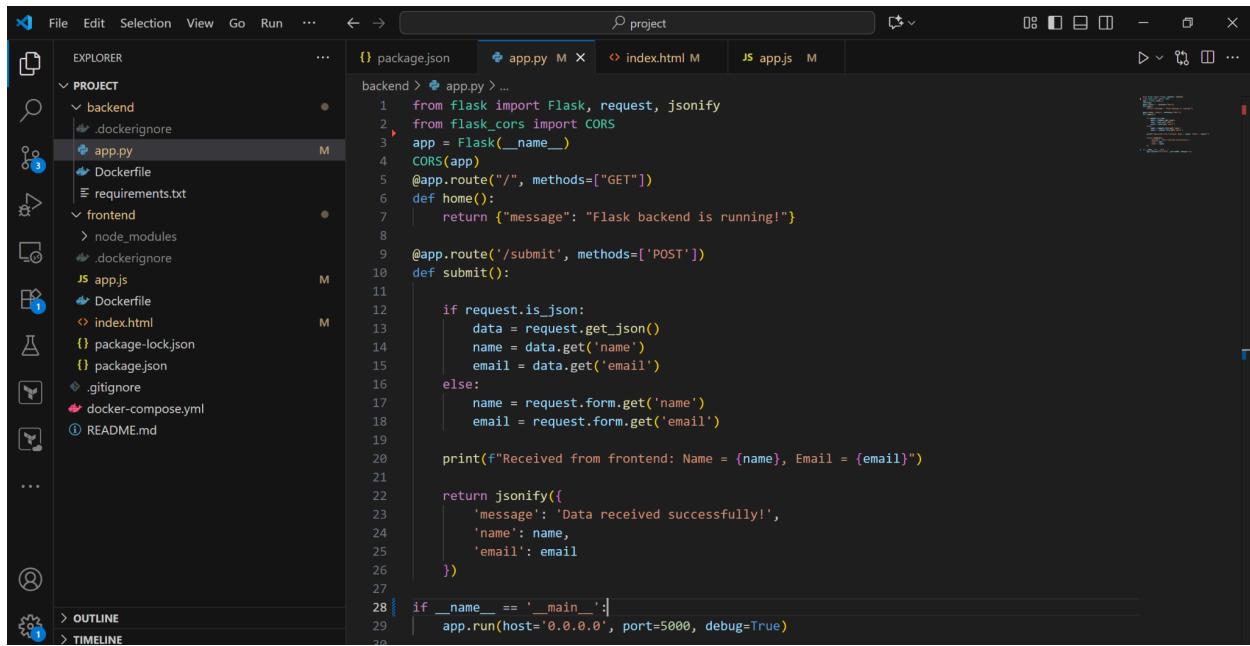
The screenshot shows the VS Code interface with the following details:

- Explorer View:** Shows the project structure with files like `backend`, `frontend`, `requirements.txt`, `Dockerfile`, `index.html`, `package-lock.json`, `.gitignore`, and `README.md`.
- Editor View:** The `package.json` file is open, displaying the following code:

```
{
  "name": "frontend",
  "version": "1.0.0",
  "main": "app.js",
  "scripts": {
    "start": "node app.js"
  },
  "dependencies": {
    "axios": "^1.4.0",
    "express": "^4.21.2"
  },
  "keywords": [],
  "author": "",
  "license": "ISC",
  "description": ""
}
```

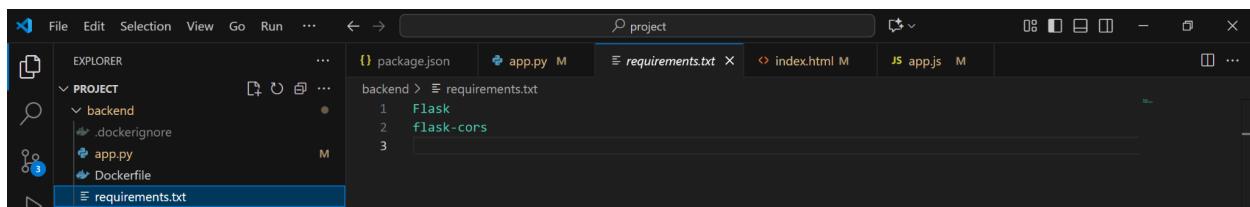
## Backend (Flask):

### app.py



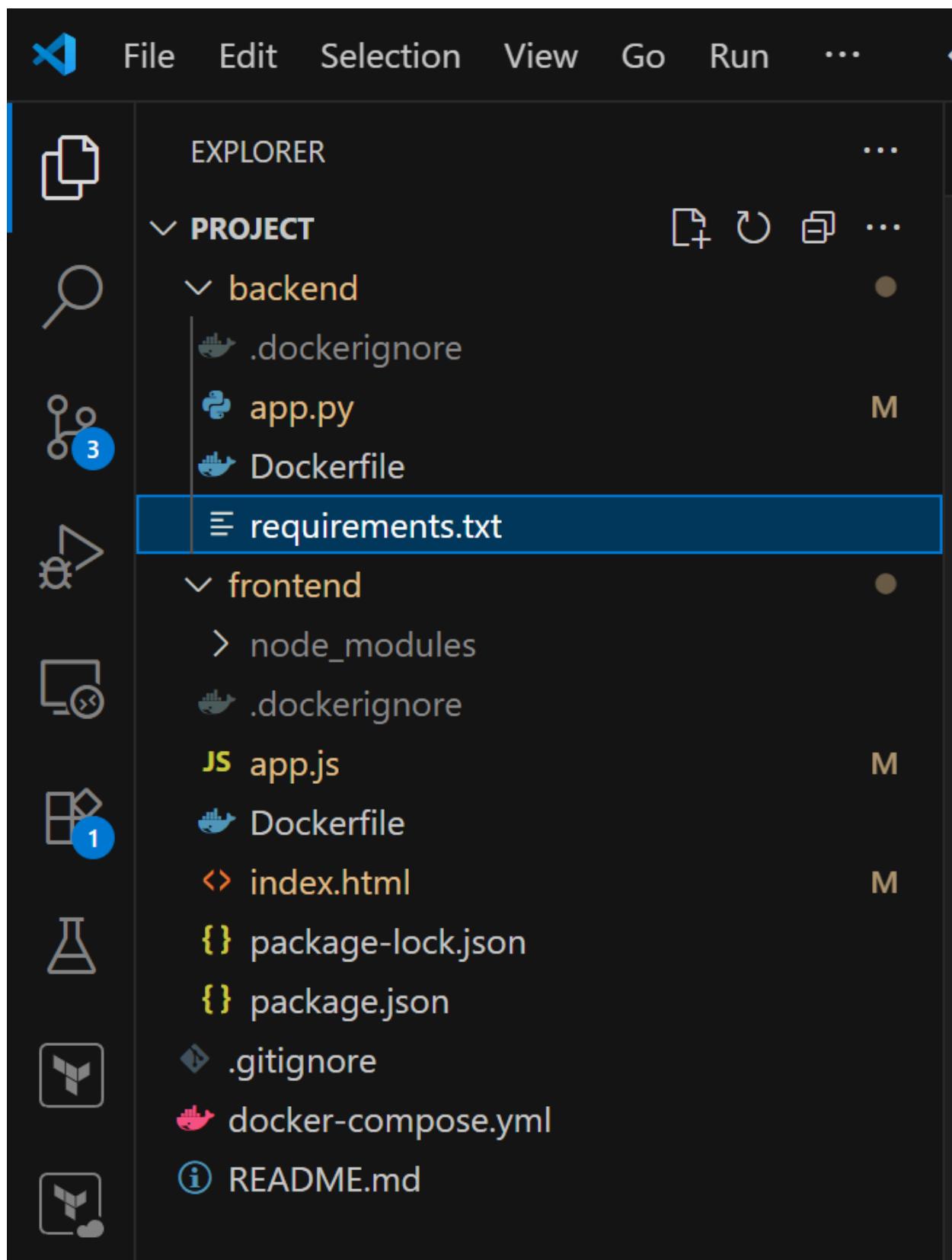
```
File Edit Selection View Go Run ... ← → project ⌂ package.json ⌂ app.py M X ⌂ index.html M JS app.js M ⌂ ... ⌂ EXPLORER PROJECT backend > app.py > ...
1 from flask import Flask, request, jsonify
2 from flask_cors import CORS
3 app = Flask(__name__)
4 CORS(app)
5 @app.route("/", methods=["GET"])
6 def home():
7     return {"message": "Flask backend is running!"}
8
9 @app.route('/submit', methods=['POST'])
10 def submit():
11
12     if request.is_json:
13         data = request.get_json()
14         name = data.get('name')
15         email = data.get('email')
16     else:
17         name = request.form.get('name')
18         email = request.form.get('email')
19
20     print(f"Received from frontend: Name = {name}, Email = {email}")
21
22     return jsonify({
23         'message': 'Data received successfully!',
24         'name': name,
25         'email': email
26     })
27
28 if __name__ == '__main__':
29     app.run(host='0.0.0.0', port=5000, debug=True)
```

### Requirements.txt



```
File Edit Selection View Go Run ... ← → project ⌂ package.json ⌂ app.py M ⌂ requirements.txt M ⌂ index.html M JS app.js M ⌂ ... ⌂ EXPLORER PROJECT backend > requirements.txt
1 Flask
2 flask-cors
3
```

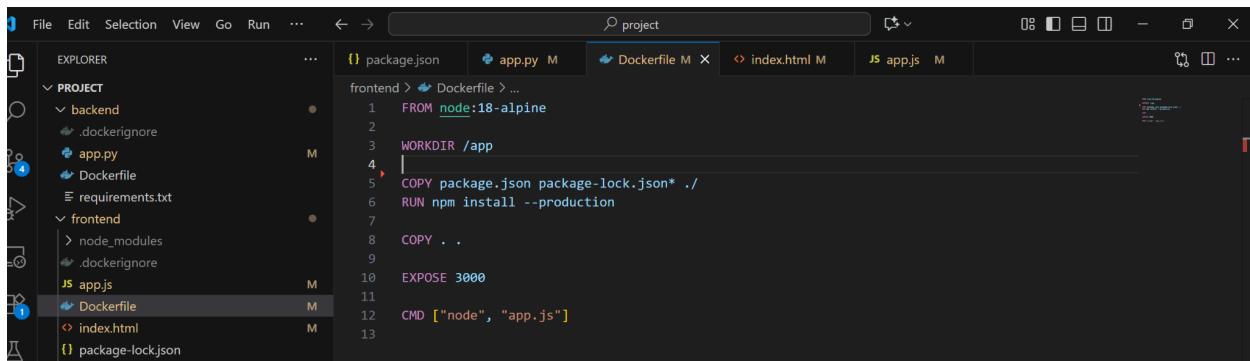
**Folder Structure:**



## Docker Configuration:

### Dockerfiles

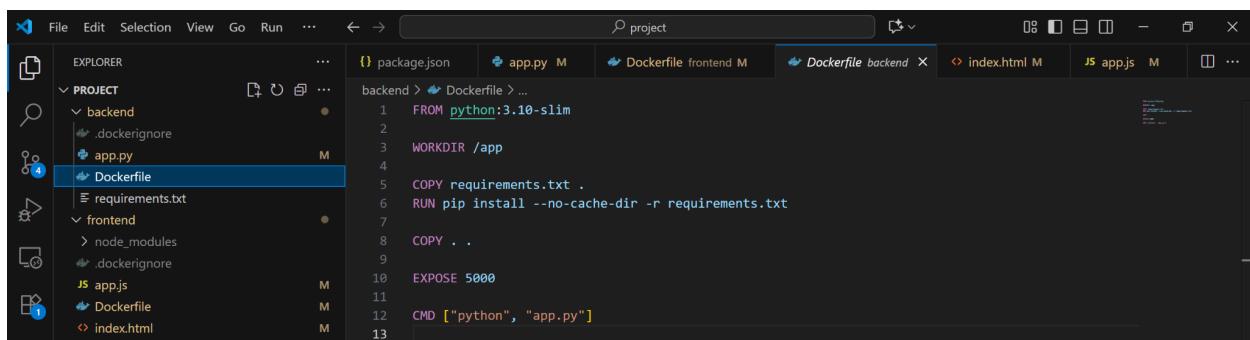
#### frontend/Dockerfile



The screenshot shows the VS Code interface with the Dockerfile for the frontend selected in the tab bar. The code in the editor is as follows:

```
FROM node:18-alpine
WORKDIR /app
COPY package.json package-lock.json* .
RUN npm install --production
COPY .
EXPOSE 3000
CMD ["node", "app.js"]
```

#### backend/Dockerfile

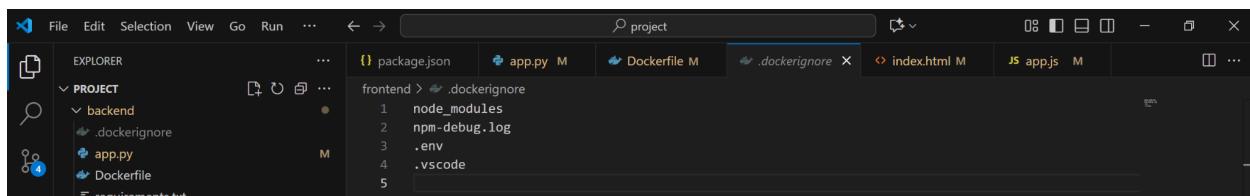


The screenshot shows the VS Code interface with the Dockerfile for the backend selected in the tab bar. The code in the editor is as follows:

```
FROM python:3.10-slim
WORKDIR /app
COPY requirements.txt .
RUN pip install --no-cache-dir -r requirements.txt
COPY .
EXPOSE 5000
CMD ["python", "app.py"]
```

#### .dockerignore

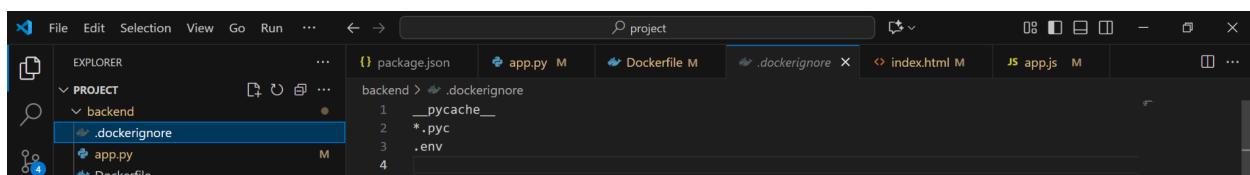
#### frontend/.dockerignore



The screenshot shows the VS Code interface with the .dockerignore file for the frontend selected in the tab bar. The code in the editor is as follows:

```
node_modules
npm-debug.log
.env
.vscode
```

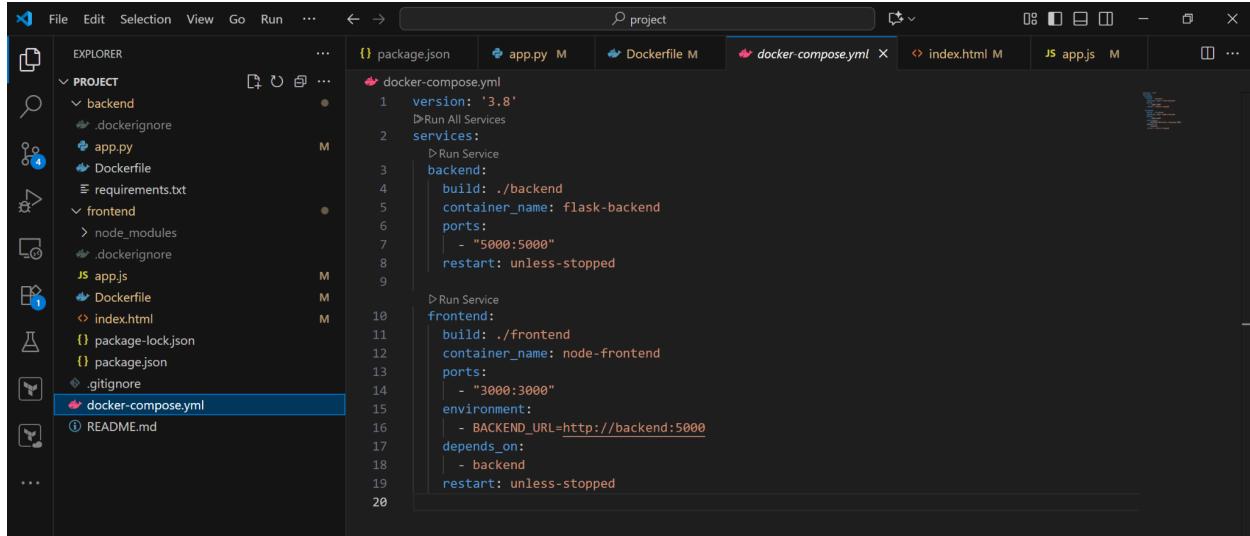
#### backend/.dockerignore



The screenshot shows the VS Code interface with the .dockerignore file for the backend selected in the tab bar. The code in the editor is as follows:

```
__pycache__
*.pyc
.env
```

## Docker-compose.yml



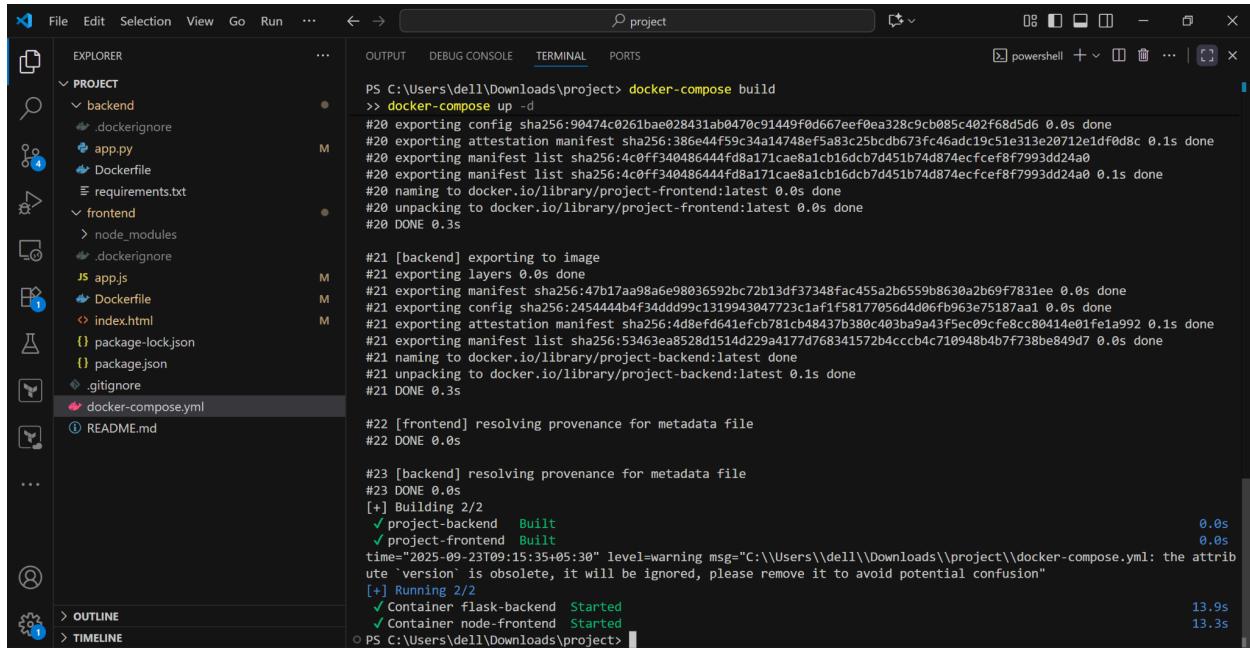
The screenshot shows the VS Code interface with the Docker-compose.yml file selected in the Explorer sidebar. The code editor displays the following configuration:

```
version: '3.8'
services:
  backend:
    build: ./backend
    container_name: flask-backend
    ports:
      - "5000:5000"
    restart: unless-stopped
  frontend:
    build: ./frontend
    container_name: node-frontend
    ports:
      - "3000:3000"
    environment:
      - BACKEND_URL=http://backend:5000
    depends_on:
      - backend
    restart: unless-stopped
```

## Run locally with Docker Compose

docker-compose build

docker-compose up -d



The screenshot shows the VS Code interface with the terminal tab active, displaying the output of running the docker-compose build and up -d commands. The terminal output is as follows:

```
PS C:\Users\dell\Downloads\project> docker-compose build
-> docker-compose up -d
#20 exporting config sha256:90474c0261bae028431ab0470c91449f0d676eef0ea328c9cb885c402f68d5d6 0.0s done
#20 exporting attestation manifest sha256:386e44f59c34a14748ef5a83c25bcd673fc46adc19c51e313e20712e1df0d8c 0.1s done
#20 exporting manifest list sha256:ac0ff340486444fd8a171cae8a1cb16dc7d451b74d874ecfcf8f7993dd24a0
#20 exporting manifest list sha256:ac0ff340486444fd8a171cae8a1cb16dc7d451b74d874ecfcf8f7993dd24a0
#20 naming to docker.io/library/project-frontend:latest 0.0s done
#20 unpacking to docker.io/library/project-frontend:latest 0.0s done
#20 DONE 0.3s

#21 [backend] exporting to image
#21 exporting layers
#21 exporting manifest sha256:47b17aa98a6e98036592bc72b13df37348fac455a2b6559b8630a2b69f7831ee 0.0s done
#21 exporting config sha256:245444b4f34dd99c13199430477231af1f58177956d4d06fb963e75187aa1 0.0s done
#21 exporting attestation manifest sha256:4d8efd641efcb781cb48437b380c403ba9a43f5ec89cf8cc80414e01fe1a992 0.1s done
#21 exporting manifest list sha256:53463ea8528d1514d229a417d768341572b4ccb4c710948b4b7f738be849d7 0.0s done
#21 naming to docker.io/library/project-backend:latest done
#21 unpacking to docker.io/library/project-backend:latest 0.1s done
#21 DONE 0.3s

#22 [frontend] resolving provenance for metadata file
#22 DONE 0.0s

#23 [backend] resolving provenance for metadata file
#23 DONE 0.0s
[+] Building 2/2
  ✓ project-backend  Built
  ✓ project-frontend  Built
time="2025-09-23T09:15:35+05:30" level=warning msg="C:\\\\Users\\\\dell\\\\Downloads\\\\project\\\\docker-compose.yml: the attribute 'version' is obsolete, it will be ignored, please remove it to avoid potential confusion"
[+] Running 2/2
  ✓ Container flask-backend Started
  ✓ Container node-frontend Started
13.9s
13.3s
PS C:\Users\dell\Downloads\project>
```

## Seeing the result in docker desktop

The screenshot shows the Docker Desktop interface for a project named "project". On the left sidebar, there are links for Ask Gordon (BETA), Containers, Images, Volumes, Builds, Models (BETA), MCP Toolkit (BETA), Docker Hub, Docker Scout, and Extensions. The main area displays two containers:

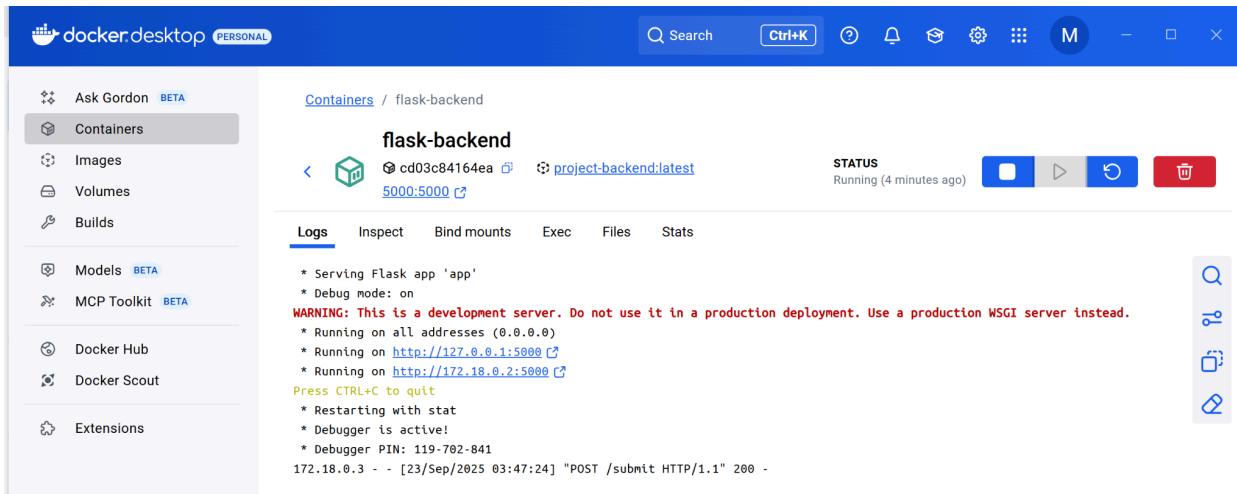
- backend**: Running on port 5000, serving a Flask app. The logs show:
  - \* Serving Flask app 'app'
  - \* Debug mode: on
  - WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.**
  - \* Running on all addresses (0.0.0.0)
  - \* Running on <http://127.0.0.1:5000>
  - \* Running on <http://172.18.0.2:5000>
- frontend**: Running on port 3000, forwarding requests to the backend. The logs show:
  - Error response from daemon: can not get logs from container which is dead or marked for removal
  - \* Serving Flask app 'app'
  - \* Debug mode: on
  - \* Serving Flask app 'app'
  - \* Debug mode: on
  - WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.**
  - \* Running on all addresses (0.0.0.0)
  - \* Running on <http://127.0.0.1:5000>
  - \* Running on <http://172.18.0.2:5000>

At the bottom, the status bar indicates: Engine running, RAM 1.66 GB, CPU 0.00%, Disk: 3.22 GB used (limit 1006.85 GB). A message "New version available" is shown in the bottom right.

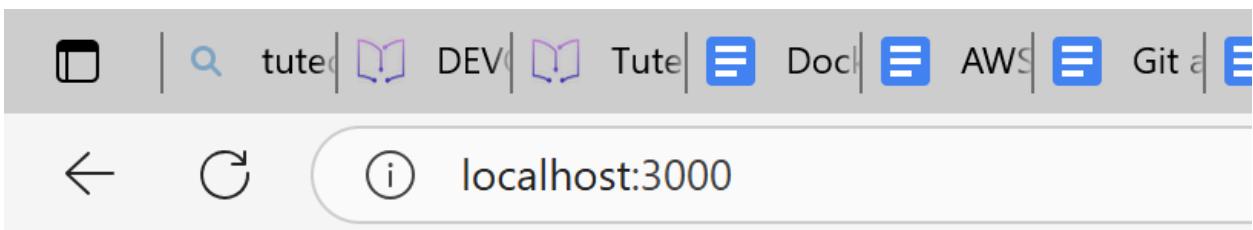
The screenshot shows the Docker Desktop interface for the "node-frontend" container. The sidebar is identical to the previous screenshot. The main area shows the container details:

- node-frontend**: Status: Running (3 minutes ago). The container ID is 44b714babfe8. It is using the image project-frontend:latest and is mapped to port 3000.
- Logs tab: Shows the application output:

```
Frontend running at http://localhost:3000
Forwarding POSTs to BACKEND_URL=http://backend:5000
```
- Inspect, Bind mounts, Exec, Files, and Stats tabs are also visible.



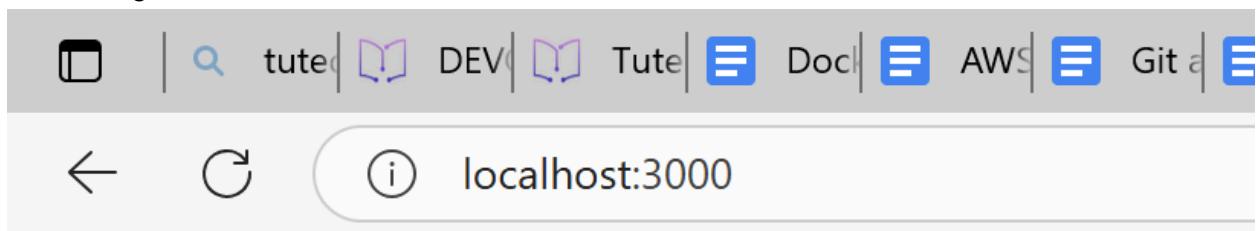
When go to the localhost:3000 this is showing



Name:

Email:

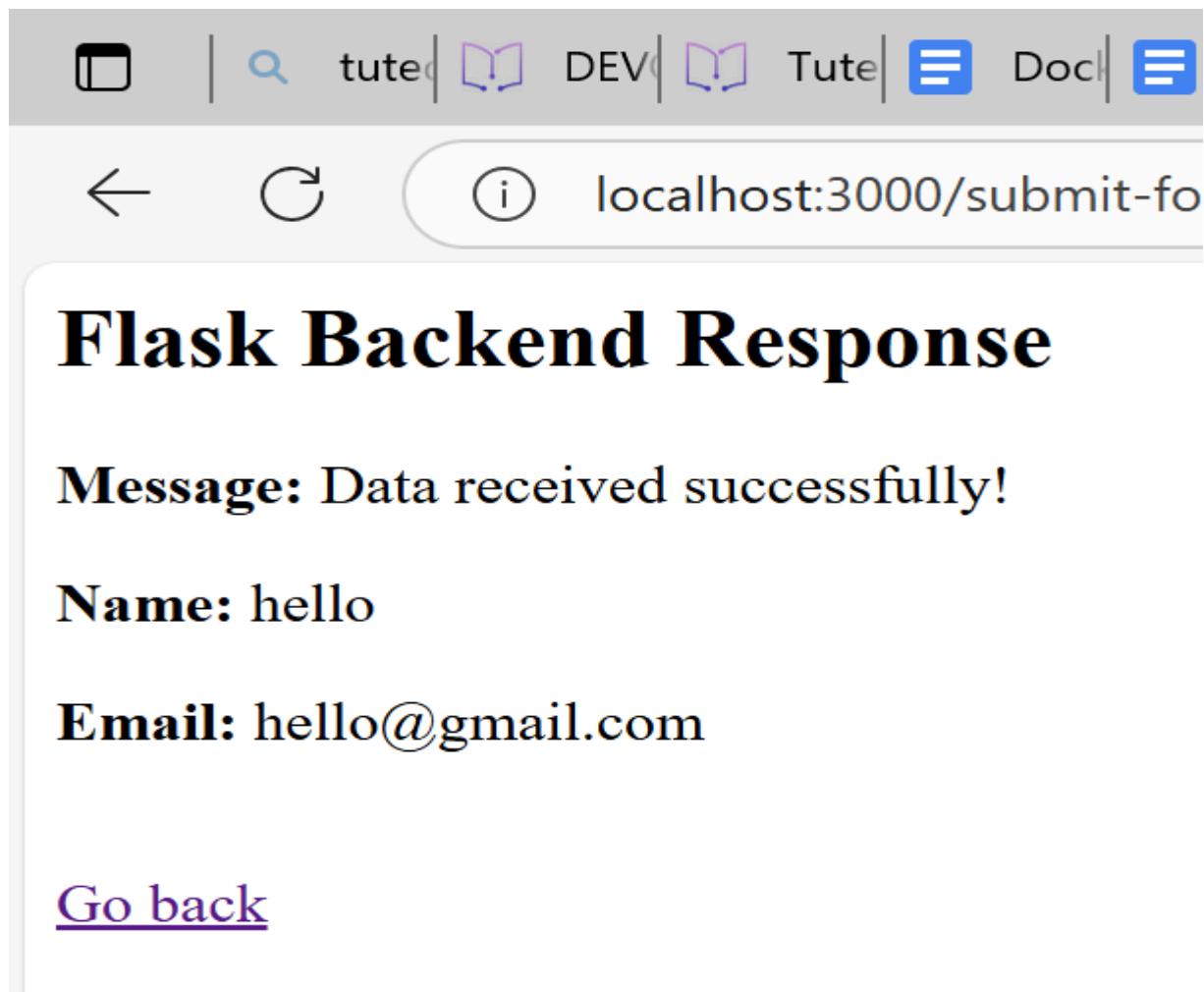
After filling the data in it



Name:

Email:

The Result



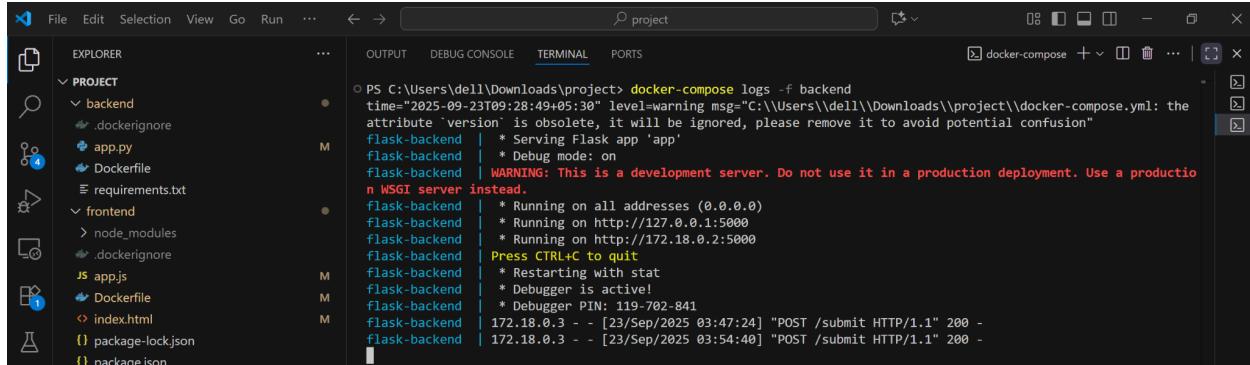
Logs:

docker-compose logs -f frontend

A screenshot of the VS Code interface, specifically the terminal tab. The terminal output shows the following log entries:

```
PS C:\Users\dell\Downloads\project> docker-compose logs -f frontend
time="2025-09-23T09:26:50+05:30" level=warning msg="C:\\\\Users\\\\dell\\\\Downloads\\\\\\project\\\\docker-compose.yml: the
attribute `version` is obsolete, it will be ignored, please remove it to avoid potential confusion"
node-frontend | Frontend running at http://localhost:3000
node-frontend | Forwarding POSTS to BACKEND_URL=http://backend:5000
```

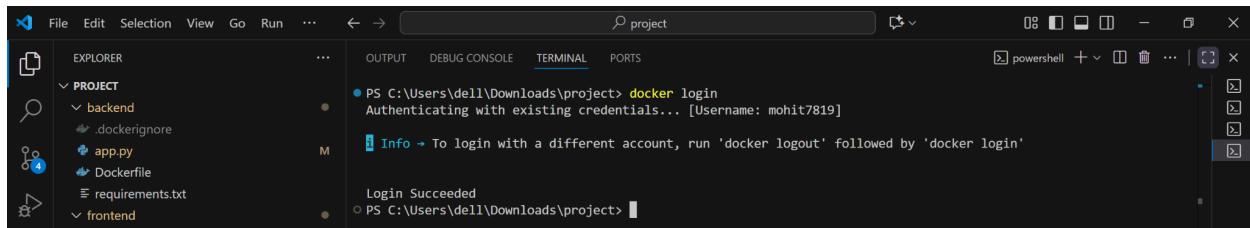
```
docker-compose logs -f backend
```



```
PS C:\Users\dell\Downloads\project> docker-compose logs -f backend
time="2025-09-23T09:28:49+05:30" level=warning msg="C:\Users\dell\Downloads\project\docker-compose.yml: the attribute 'version' is obsolete, it will be ignored, please remove it to avoid potential confusion"
flask-backend | * Serving Flask app 'app'
flask-backend | * Debug mode: on
flask-backend | WARNING: This is a development server. Do not use it in a production deployment. Use a production SQL server instead.
flask-backend | * Running on all addresses (0.0.0.0)
flask-backend | * Running on http://127.0.0.1:5000
flask-backend | * Running on http://172.18.0.2:5000
flask-backend | Press CTRL+C to quit
flask-backend | * Restarting with stat
flask-backend | * Debugger is active!
flask-backend | * Debugger PIN: 119-702-841
flask-backend | 172.18.0.3 - - [23/Sep/2025 03:47:24] "POST /submit HTTP/1.1" 200 -
flask-backend | 172.18.0.3 - - [23/Sep/2025 03:54:40] "POST /submit HTTP/1.1" 200 -
```

Push Docker images to Docker Hub

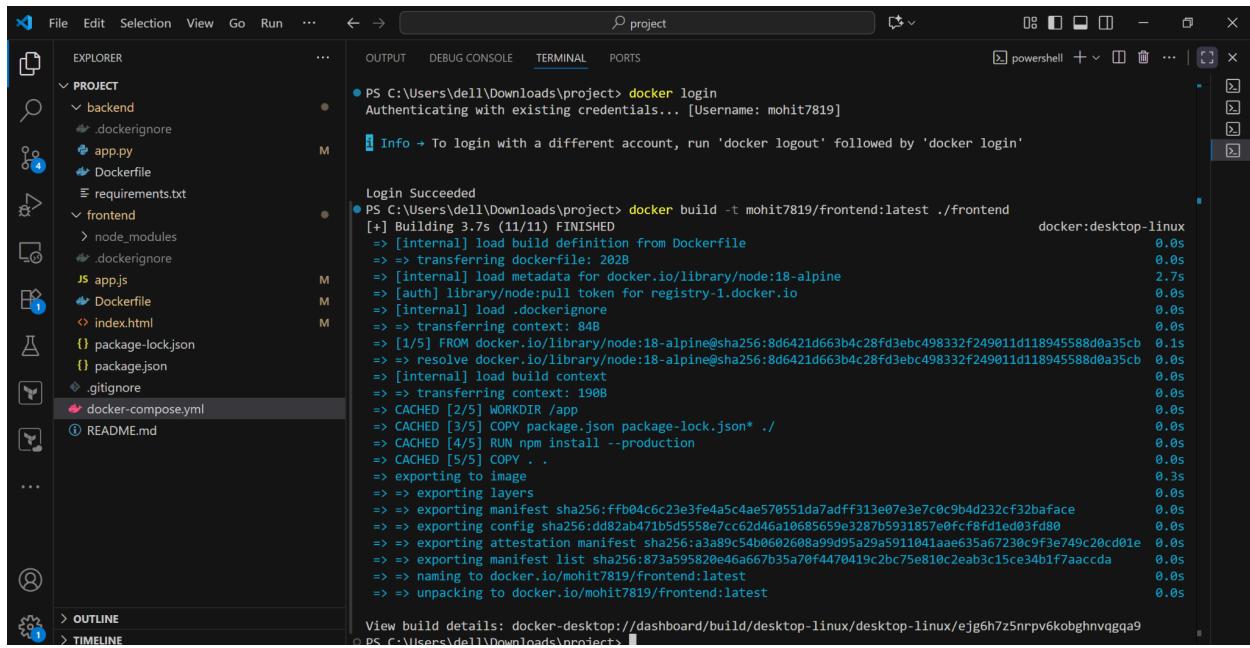
login to docker



```
PS C:\Users\dell\Downloads\project> docker login
Authenticating with existing credentials... [Username: mohit7819]
Info → To login with a different account, run 'docker logout' followed by 'docker login'

Login Succeeded
PS C:\Users\dell\Downloads\project>
```

```
docker build -t <dockerhub-username>/frontend:latest ./frontend (in my case mohit7819)
```



```
PS C:\Users\dell\Downloads\project> docker login
Authenticating with existing credentials... [Username: mohit7819]
Info → To login with a different account, run 'docker logout' followed by 'docker login'

Login Succeeded
PS C:\Users\dell\Downloads\project> docker build -t mohit7819/frontend:latest ./frontend
[+] Building 3.7s (11/11) FINISHED                                            docker:desktop-linux
  => [internal] load build definition from Dockerfile                      0.0s
  => => transferring dockerfile: 202B                                       0.0s
  => [internal] load metadata for docker.io/library/node:18-alpine          2.7s
  => [auth] library/node:pull token for registry-1.docker.io                 0.0s
  => [internal] load .dockerignore                                         0.0s
  => => transferring context: 848                                           0.0s
  => [1/5] FROM docker.io/library/node:18-alpine@sha256:8d6421d663b4c28fd3ebc498332f249011d118945588d0a35cb 0.1s
  => => resolve docker.io/library/node:18-alpine@sha256:8d6421d663b4c28fd3ebc498332f249011d118945588d0a35cb 0.0s
  => [internal] load build context                                         0.0s
  => => transferring context: 1908                                         0.0s
  => CACHED [2/5] WORKDIR /app                                             0.0s
  => CACHED [3/5] COPY package.json package-lock.json* ./                  0.0s
  => CACHED [4/5] RUN npm install --production                           0.0s
  => CACHED [5/5] COPY . .                                               0.0s
  => exporting to image                                                 0.3s
  => => exporting layers                                              0.0s
  => => exporting manifest sha256:ffb04c6c23e3fe4a5c4ae570551da7adff313e07e3e7c0c9b4d232cf32baface 0.0s
  => => exporting config sha256:dd82ab471b5d5558e7cc62d46a10685659e3287b5931857e0fcf8fd1ed03fd80 0.0s
  => => exporting attestation manifest sha256:a3a89c54b602608a99d95a29a5911041aae635a6723bc9f3e749c20cd01e 0.0s
  => => exporting manifest list sha256:873a595820e46a667b35a70f4470419c2bc75e810c2eab3c15ce34b1f7aaccda 0.0s
  => => naming to docker.io/mohit7819/frontend:latest                     0.0s
  => => unpacking to docker.io/mohit7819/frontend:latest                   0.0s
```

docker build -t <dockerhub-username>/backend:latest ./backend (in my case mohit7819)

The screenshot shows the VS Code interface with the terminal tab selected. The command `docker build -t mohit7819/backend:latest ./backend` is being run. The terminal output shows the build process, which takes 3.45 seconds and involves several steps like loading build definitions, transferring files, and resolving dependencies. The Dockerfile and requirements.txt are used to build the image. The final step shows the image being tagged and pushed to the Docker registry.

```
PS C:\Users\dell\Downloads\project> docker build -t mohit7819/backend:latest ./backend
[+] Building 3.4s (11/11) FINISHED
  => [internal] load build definition from Dockerfile
  => => transferring dockerfile: 212B
  => [internal] load metadata for docker.io/library/python:3.10-slim
  => [auth] library/python:pull token for registry-1.docker.io
  => [internal] load .dockerignore
  => => transferring context: 66B
  => [1/5] FROM docker.io/library/python:3.10-slim@sha256:f8081b61393cc16b2339d377bb523a5646cdd28fc6105612b
  => => resolve docker.io/library/python:3.10-slim@sha256:f8081b61393cc16b2339d377bb523a5646cdd28fc6105612b
  => [internal] load build context
  => => transferring context: 125B
  => => CACHED [2/5] WORKDIR /app
  => => CACHED [3/5] COPY requirements.txt .
  => => CACHED [4/5] RUN pip install --no-cache-dir -r requirements.txt
  => => CACHED [5/5] COPY .
  => => exporting to image
  => => exporting layers
  => => exporting manifest sha256:44089092cc70238951cc2f321551d2f541afec926f5eed31138cfb082446eea
  => => exporting config sha256:18821922b5220b13d87ff6c53be6caeaf8e3b966f2153de1f7c58fd3b13cd054
  => => exporting attestation manifest sha256:be40c22dac88ef26ca342ff874e24f12659622bf808926618815c557dd4b7
  => => exporting manifest list sha256:d8a11080581e39b3ee764abdb5cd2ecf561c5e696447258ff4937a118d2d5abf
  => => naming to docker.io/mohit7819/backend:latest
  => => unpacking to docker.io/mohit7819/backend:latest
View build details: docker-desktop://dashboard/build/desktop-linux/desktop-linux/mar6rs1mbvy4idnmc2d1kv1y1
PS C:\Users\dell\Downloads\project>
```

docker push <dockerhub-username>/frontend:latest (in my case mohit7819)

The screenshot shows the VS Code interface with the terminal tab selected. The command `docker push mohit7819/frontend:latest` is being run. The terminal output shows the push process, which takes 2.8895 seconds and involves mounting various files and directories from the host machine to the container. The Dockerfile and app.js are used to build the image. The final step shows the image being tagged and pushed to the Docker registry.

```
PS C:\Users\dell\Downloads\project> docker push mohit7819/frontend:latest
The push refers to repository [docker.io/mohit7819/frontend]
28895bd3454f: Pushed
f18232174bc9: Mounted from mohit7819/project-frontend
dd71dde834b5: Mounted from mohit7819/project-frontend
1e5a4d89ceef: Mounted from mohit7819/project-frontend
35240dbdf0c3: Pushed
264d8afa54a4: Mounted from mohit7819/project-frontend
5a814acf5131b: Mounted from mohit7819/project-frontend
d0e37786d2e8: Mounted from mohit7819/project-frontend
25ff2da83641: Mounted from mohit7819/project-frontend
latest: digest: sha256:873a595820e46a67b35a70f4470419c2bc75e810c2eab3c15ce34b1f7aacda size: 856
PS C:\Users\dell\Downloads\project>
```

docker push <dockerhub-username>/backend:latest (in my case mohit7819)

The screenshot shows the VS Code interface with the terminal tab selected. The command `docker push mohit7819/backend:latest` is being run. The terminal output shows the push process, which takes 2.8612 seconds and involves mounting various files and directories from the host machine to the container. The Dockerfile and app.py are used to build the image. The final step shows the image being tagged and pushed to the Docker registry.

```
PS C:\Users\dell\Downloads\project> docker push mohit7819/backend:latest
The push refers to repository [docker.io/mohit7819/backend]
ce1261c6d567: Mounted from mohit7819/project-backend
d02c566428b0: Mounted from mohit7819/project-backend
6dc114d5e12c: Mounted from mohit7819/project-backend
fceec8b8a90b: Mounted from mohit7819/project-backend
1a9eda09592d: Mounted from mohit7819/project-backend
4b5575efdab2: Mounted from mohit7819/project-backend
c86a4deda990: Mounted from mohit7819/project-backend
b60e89ead25d: Pushed
d60408077bc4: Pushed
latest: digest: sha256:d8a11080581e39b3ee764abdb5cd2ecf561c5e696447258ff4937a118d2d5abf size: 856
PS C:\Users\dell\Downloads\project>
```

In Docker hub we see the repositories

The screenshot shows the Docker Hub interface. The top navigation bar includes links for 'hub', 'Explore', and 'My Hub'. A search bar says 'Search Docker Hub' with a 'CtrlK' button. On the left, a sidebar for 'mohit7819' shows sections for 'Repositories' (selected), 'Collaborations', 'Settings', 'Default privacy', 'Notifications', 'Billing', 'Usage' (expanded), 'Pulls', and 'Storage'. The main area is titled 'Repositories' with the sub-instruction 'All repositories within the mohit7819 namespace.' It features a search bar, a dropdown for 'All content', and a 'Create a repository' button. A table lists four repositories:

Name	Last Pushed	Contains	Visibility	Scout
mohit7819/backend	less than a minute ago	IMAGE	Public	Inactive
mohit7819/frontend	2 minutes ago	IMAGE	Public	Inactive
mohit7819/project-backend	about 4 hours ago	IMAGE	Public	Inactive
mohit7819/project-frontend	about 4 hours ago	IMAGE	Public	Inactive

At the bottom, it says '1-4 of 4'.

Push to GitHub

.gitignore

The screenshot shows the VS Code interface with a dark theme. The Explorer sidebar on the left shows a project structure with files like 'package.json', 'app.py', 'Dockerfile', 'requirements.txt', 'backend', 'frontend', 'index.html', 'app.js', 'Dockerfile', 'index.html', 'package-lock.json', 'package.json', '.gitignore', 'docker-compose.yml', and 'README.md'. The 'REQUIREMENTS' tab is selected. In the center, a code editor window displays the contents of the '.gitignore' file:

```
# Node
.frontend/node_modules
.frontend/npm-debug.log
.frontend/.env

# Python
.backend/__pycache__/
.backend/*.pyc
.backend/.env

# IDEs and OS
.vscode/
.DS_Store
*.log

# Docker
**/.dockerignore
```

Git commands

git init

git add .

git commit -m "Initial commit: frontend + backend + docker setup"

```
git remote add origin https://github.com/Mohit7819/PROJECT
```

```
git branch -M main
```

```
git push -u origin main
```

The screenshot shows a terminal window with the following command history:

```
PS C:\Users\dell\Downloads\project> git remote add origin https://github.com/Mohit7819/PROJECT
error: remote origin already exists.
PS C:\Users\dell\Downloads\project> git branch -M main
PS C:\Users\dell\Downloads\project> git push -u origin main
Enumerating objects: 15, done.
Counting objects: 100% (15/15), done.
Delta compression using up to 4 threads
Compressing objects: 100% (8/8), done.
Writing objects: 100% (8/8), 799 bytes | 266.00 KiB/s, done.
Total 8 (delta 5), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (5/5), completed with 5 local objects.
To github.com:Mohit7819/PROJECT.git
  4e770da..94b39c0  main -> main
branch 'main' set up to track 'origin/main'.
PS C:\Users\dell\Downloads\project> ^C
PS C:\Users\dell\Downloads\project>
```

Open and see the repository in [github.com](https://github.com/Mohit7819/PROJECT)

The screenshot shows a GitHub repository page for 'PROJECT' owned by 'Mohit7819'. The repository is public and has 1 branch and 0 tags. The code tab is selected, showing the following commit history:

Author	Commit Message	Time	Commits
Mohit7819	Initial commit: frontend + backend + docker setup	94b39c0 · 5 minutes ago	2 Commits
backend	Initial commit: frontend + backend + docker setup	5 minutes ago	
frontend	Initial commit: frontend + backend + docker setup	5 minutes ago	
.gitignore	Initial commit with frontend, backend and docker setup	3 hours ago	
README.md	Initial commit with frontend, backend and docker setup	3 hours ago	
docker-compose.yml	Initial commit with frontend, backend and docker setup	3 hours ago	

The repository details section includes:

- About: No description, website, or topics provided.
- Activity: 0 activity.
- Stars: 0 stars.
- Watching: 0 watching.
- Forks: 0 forks.
- Releases: No releases published. [Create a new release](#).
- Packages: No packages published. [Publish your first package](#).