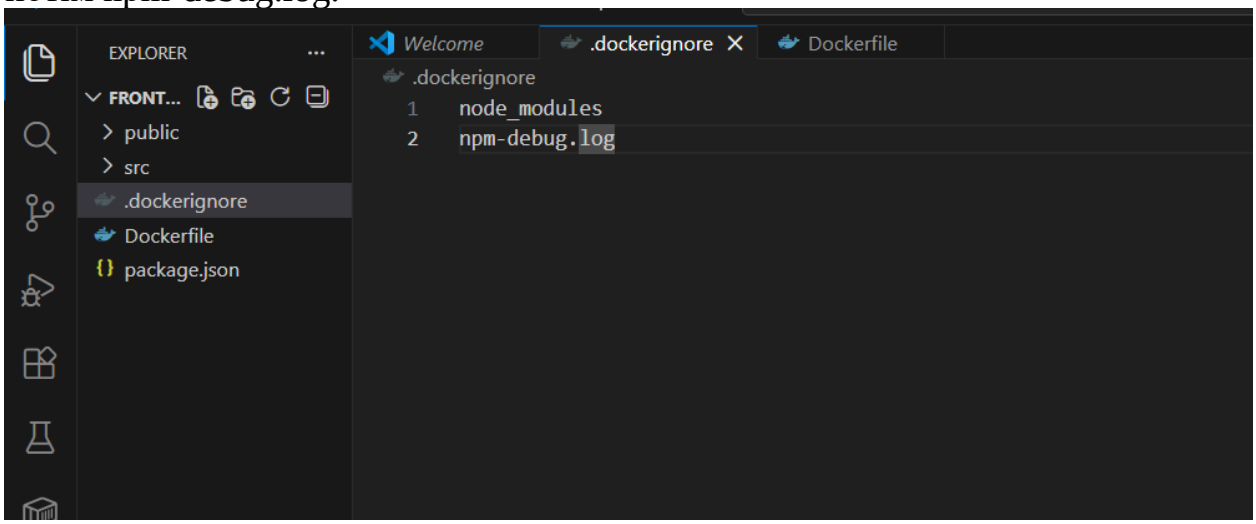
**Лабораторна робота 2**
**Тема: Створення Docker контейнеру для frontend додатку**

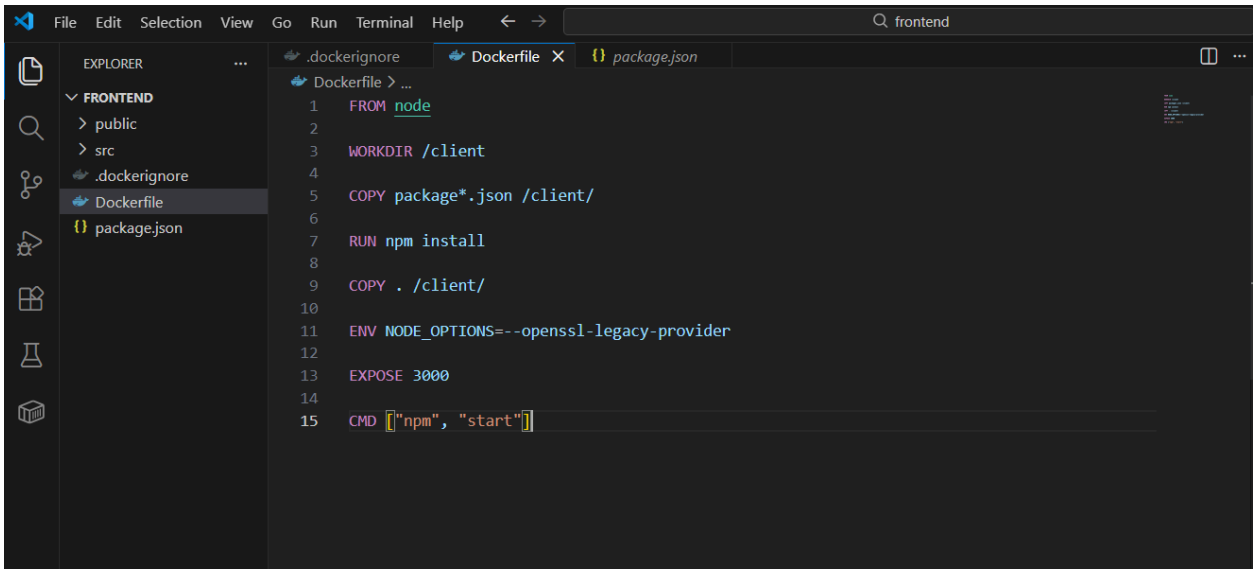
Перше, що я зроблю на своєму робочому столі, це створю нову папку під назвою frontend. А потім я перейду до файлів вправ і знову перегляну Resources, щоб знайти папку frontend.

Це точно той самий процес, який був у 1 лабораторній роботі, коли створювали Dockerfile та файл dockerignore. Отже, давайте створимо ці два файли.

Перший, який ми створимо, це dockerignore. Зробимо це. Отже, .dockerignore. А потім Dockerfile. Добре, додамо два рядки з node_modules і потім npm-debug.log.




Dockerfile трохи відрізняється, але більшість інструкцій будуть однаковими.

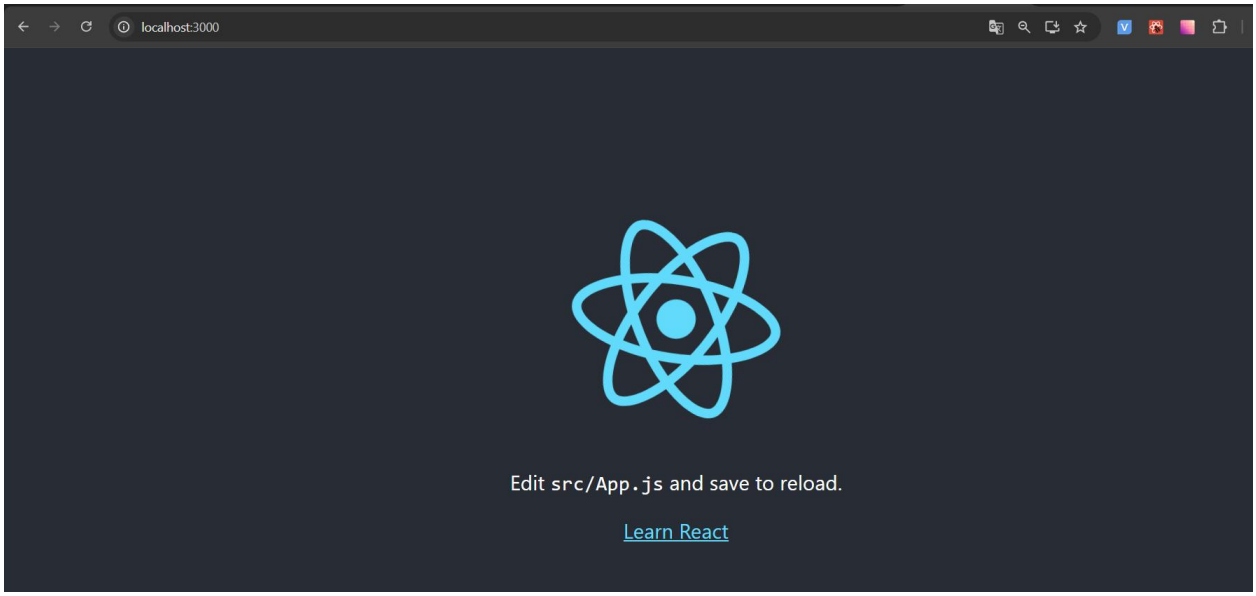



Далі створимо образ:

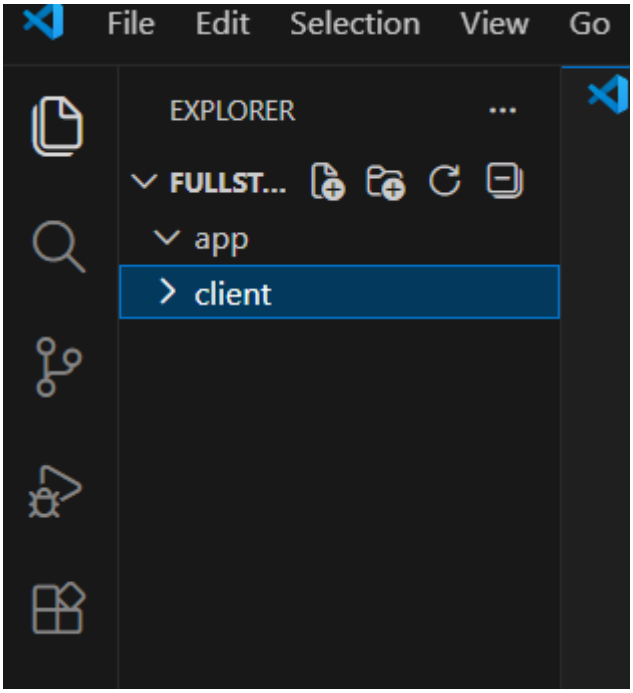
**docker build -t cyberfan/frontend .**

Потім створимо контейнер з цього образу:

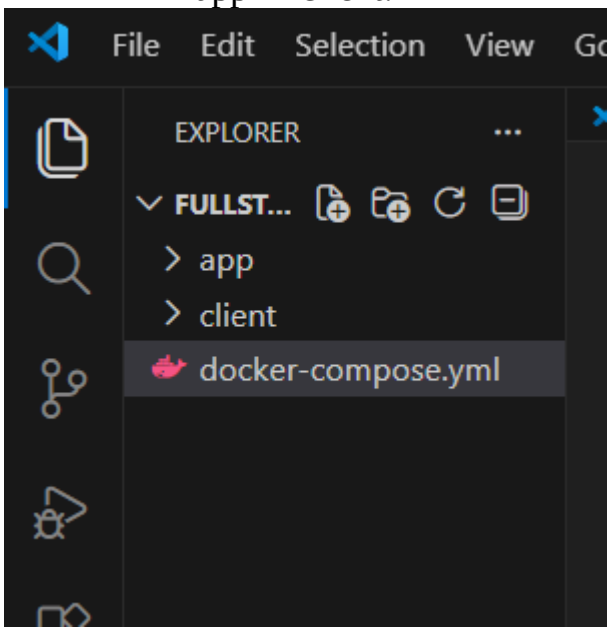
**docker run -p 3000:3000 cyberfan/frontend**



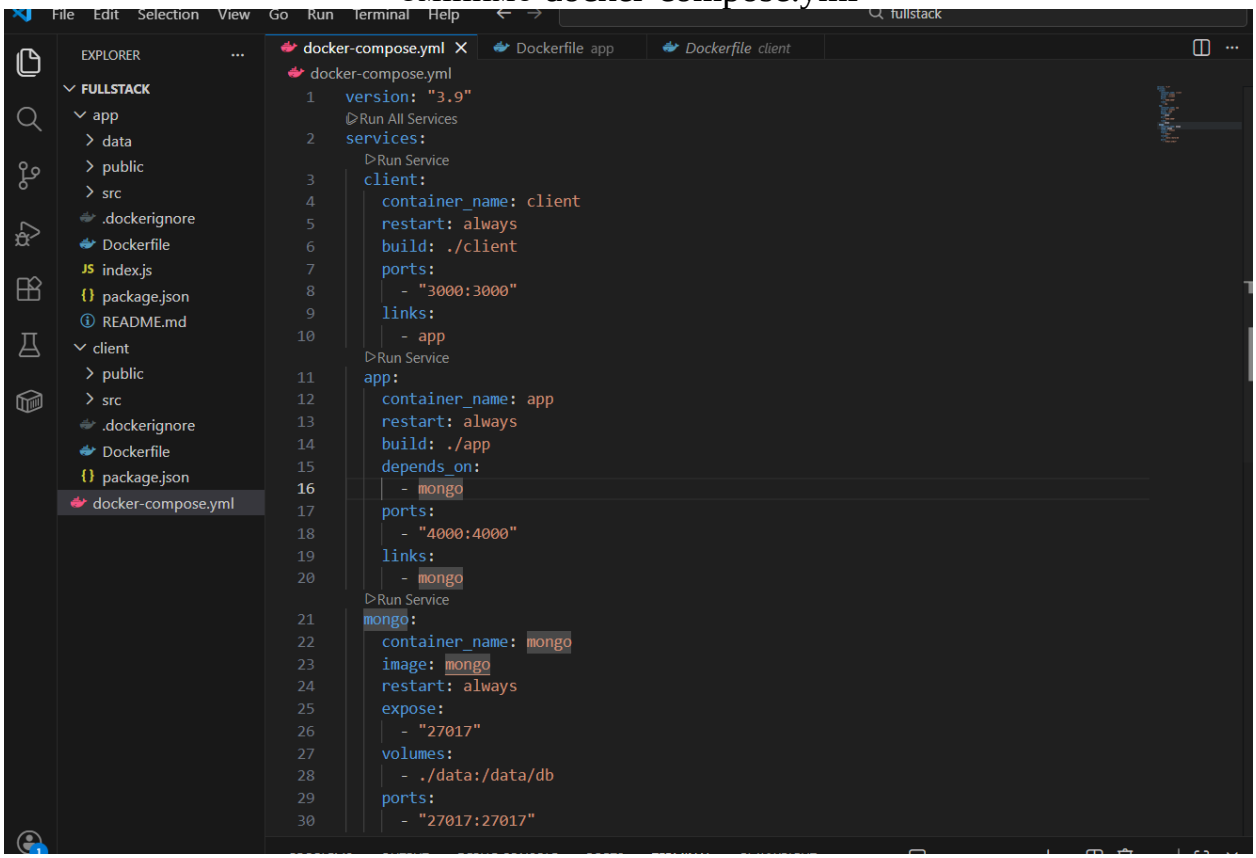

Наступний крок створення контейнерів для повноцінного fullstack додатку. Створимо каталог fullstack. В середині цього каталогу створимо ще два як на малюнку:



В client треба скопіювати весь код з frontend а в app з бекенд окрім docker-compose.yml. Файл docker-compose.yml треба скопіювати на верхню дерикторію поруч з каталогами app та client.




Змінимо docker-compose.yml




Внесемо деякі зміни в Dockerfile в каталозі app

```
FROM node

WORKDIR /app
```

```
COPY package*.json ./

RUN npm install

COPY . .

EXPOSE 4000
CMD    ["npm", "start"]
```

Запустимо команду:

**docker-compose build**

Далі запустимо команду, яка піднімає контейнер MongoDB, описаний у вашому docker-compose.yml, у фоновому режимі. Якщо у файлі є інші сервіси (наприклад, backend, frontend), вони не будуть запущені — тільки mongo.

**docker-compose up -d mongo**

Потім бекенд:

**docker-compose up -d app**

І нарешті фронтенд

**docker-compose up -d client**

Перевірте що всі три контейнера запушені

**docker ps**