Program 1: Build a docker container from a custom dockerfile

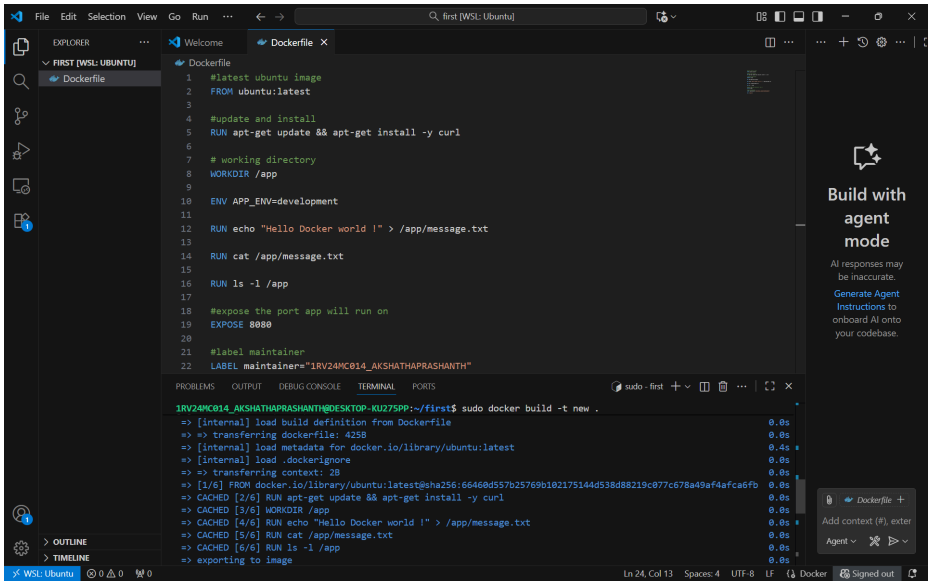
=> In the terminal,

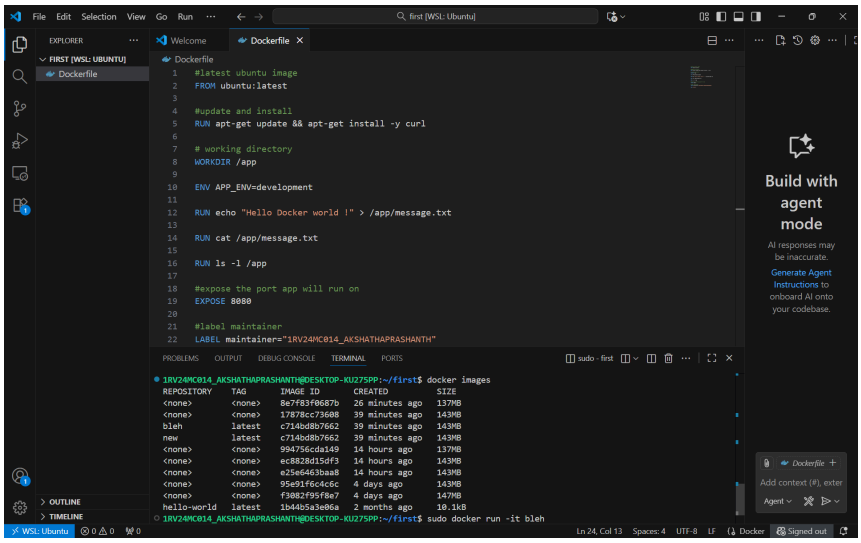
- Mkdir first
- Cd first
- Code .

In VS code, create a Dockerfile

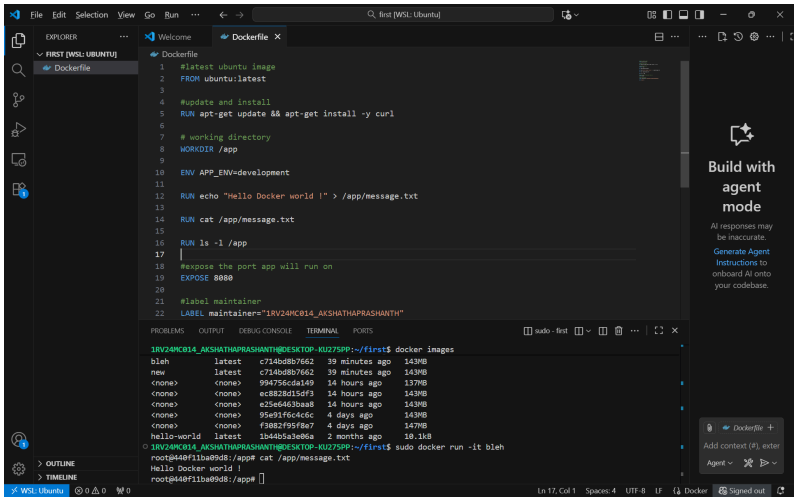
- FROM ubuntu:latest
- RUN apt-get update && apt-get install -y curl
- WORKDIR /app
- ENV APP_ENV=development
- RUN echo “hello docker world: > /app/message.txt
- RUN cat /app/message.txt
- COPY ./app
- ENV APP_ENV=development
- EXPOSE 8080
- CMD [“bash”]
- Label maintainer=”1RV24MC014_AKSHATHAPRASHANTH

RUN:
docker build -t new .
docker images





sudo docker run -it bleh
In root: cat /app/message.txt



Program 2: Develop a multistage Dockerfile for container orchestration

=> In the terminal,

- Mkdir second
- cd second
- code .

In VS code, create a folder src, in that create index.js

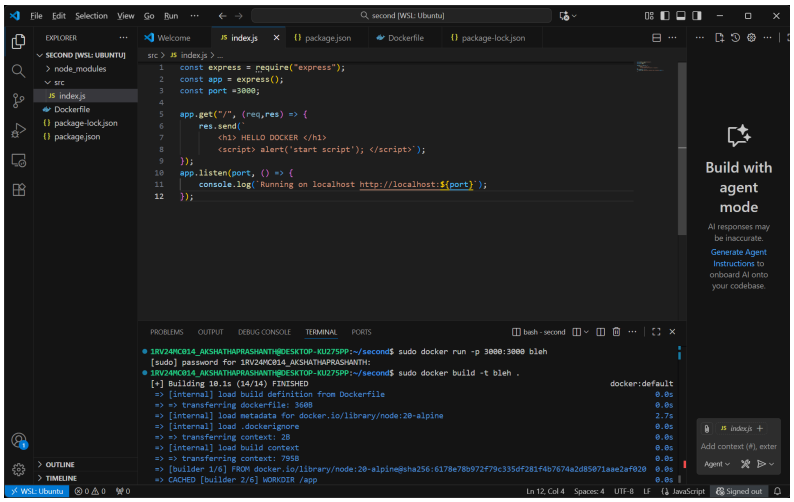
```
const express= require('express');
const app = express();
const port =3000;

app.get('/', (req,res) => {
    res.send(`
            <script> alert ('start script') ; </script>
             <h1> Hello Docker </h1> `);
});

app.listen(port, () => {
        console.log(`server running on localhost http://localhost:${port}`);
});
```

In the terminal, install npm, it will also create the package.json file

- $npm init -y
- $npm install express



In the package.json file, under "scripts", add

```
"scripts" :{
    "build": "echo 'HI' && mkdir -p dist && cp src/index.js dist/",
```

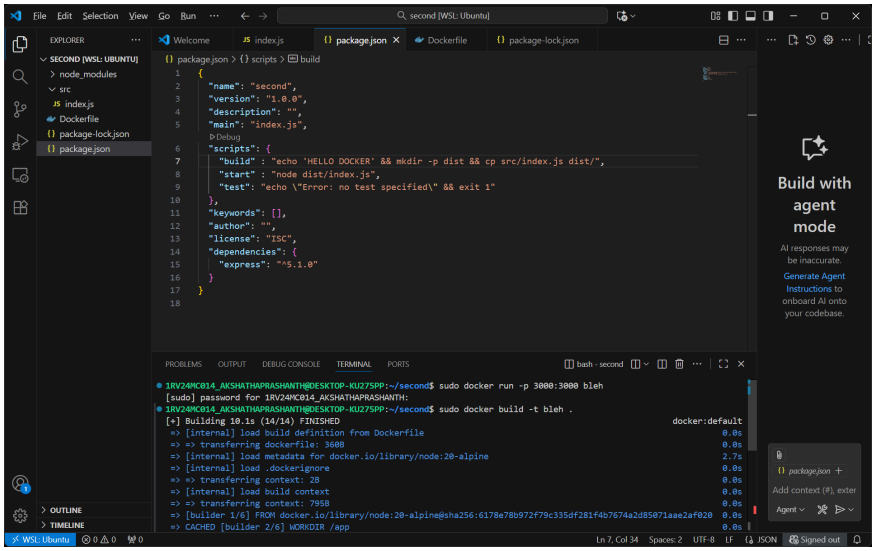
“start”: “node dist/index.js”,
}

Create a Dockerfile

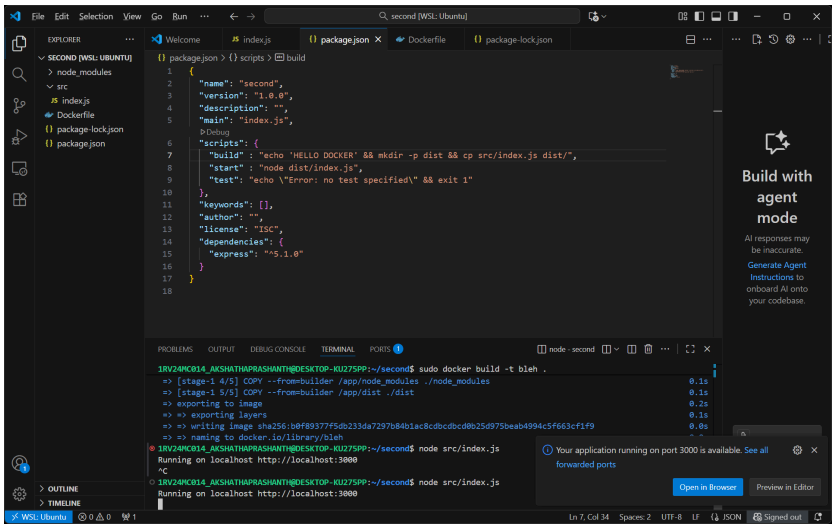
- FROM node:20-alpine AS builder
- WORKDIR /app
- COPY package.json ./
- RUN npm install
- COPY src ./src
- RUN npm run build
  #stage2
- FROM node:20-alpine
- WORKDIR /app
- COPY –from=builder /app/package.json ./
- COPY –from=builder /app/node_modules ./node_modules
- COPY –from=builder /app/dist ./dist
- EXPOSE 3000
- CMD [“npm” , “start”]

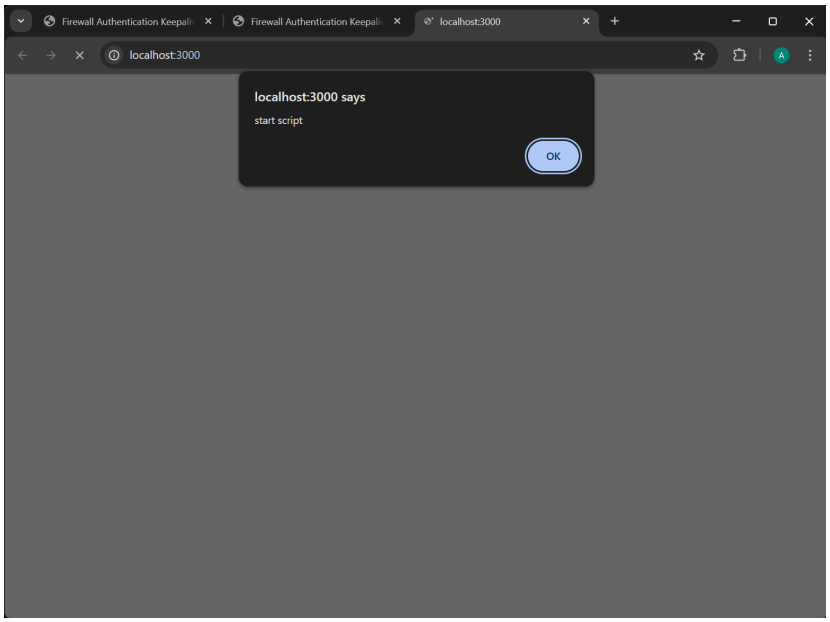
In the terminal, run

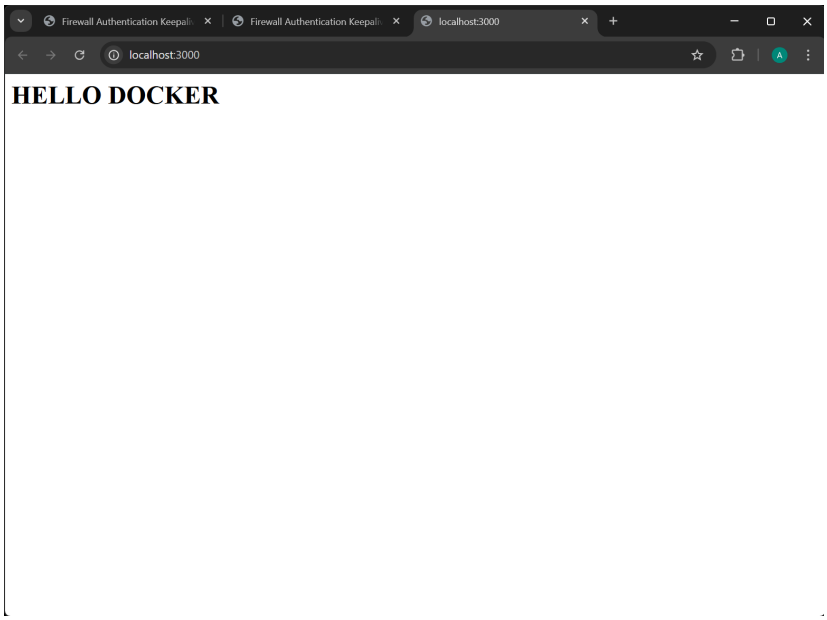
- Sudo docker run -p 3000:3000 bleh
- sudo docker build -t bleh



Next, run node src/index.js


File Edit Selection View Go Run
second [WSL: Ubuntu]
EXPLORER
SECOND [WSL: UBUNTU]
node_modules
src
index.js
Dockerfile
package-lock.json
package.json
Welcome
index.js
package.json
Dockerfile
package-lock.json
{
"name": "second",
"version": "1.0.0",
"description": "",
"main": "index.js",
Debug
"scripts": {
"build" : "echo 'HELLO DOCKER' && mkdir -p dist && cp src/index.js dist/",
"start" : "node dist/index.js",
"test": "echo \"Error: no test specified\" && exit 1"
},
"keywords": [],
"author": "",
"license": "ISC",
"dependencies": {
"express": "^5.1.0"
}
}
Build with agent mode
AI responses may be inaccurate.
Generate Agent Instructions to onboard AI onto your codebase.
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
node - second
1RV24MC014_AKSHATHAPRASHANTH@DESKTOP-KU275PP:~/second$ sudo docker build -t bleh .
=> [stage-1 4/5] COPY --from=builder /app/node_modules ./node_modules 0.1s
=> [stage-1 5/5] COPY --from=builder /app/dist ./dist 0.1s
=> exporting to image 0.2s
=> => exporting layers 0.1s
=> => writing image sha256:b0f89377f5db233da7297b84b1ac8cdbcdbcd0b25d975beab4994c5f663cf1f9 0.0s
=> => naming to docker.io/library/bleh 0.0s
1RV24MC014_AKSHATHAPRASHANTH@DESKTOP-KU275PP:~/second$ node src/index.js
Running on localhost http://localhost:3000
^C
1RV24MC014_AKSHATHAPRASHANTH@DESKTOP-KU275PP:~/second$ node src/index.js
Running on localhost http://localhost:3000
Your application running on port 3000 is available. See all forwarded ports
Open in Browser
Preview in Editor
OUTLINE
TIMELINE
WSL: Ubuntu
Ln 7, Col 34 Spaces: 2 UTF-8 LF JSON Signed out


Firewall Authentication Keepali
Firewall Authentication Keepali
localhost:3000
localhost:3000
localhost:3000 says
start script
OK


Firewall Authentication Keepali
Firewall Authentication Keepali
localhost:3000
localhost:3000
HELLO DOCKER