Program - 2

Develop a Multi-Stage Dockerfile for Container Orchestration

Description

The objective of this problem is to develop a multi stage dockerfile for container orchestration. This approach helps optimize image size, improve security, and streamline the build and deployment process.

Project Structure

program2/

→ Dockerfile

→ package.json

→ package-lock.json

→ src/ index.js

→ node_modules

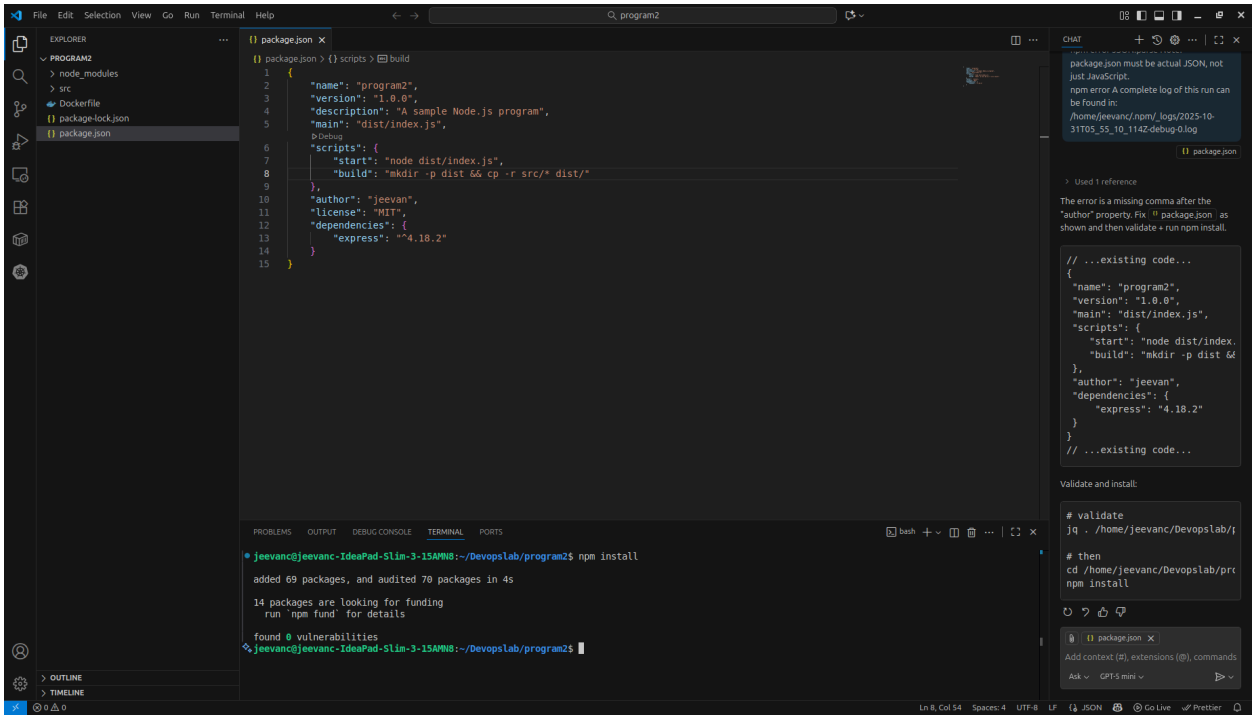
Step 1: Create the Dockerfile

```
jeevanc@jeevanc-IdeaPad-Slim-3-15AMN8:~/Devopslab/program2$ cat Dockerfile
#Stage 1: Build stage
FROM node:20-alpine AS builder

#Set Working directory
WORKDIR /app

#Copy package files and install dependencies
COPY package.json package-lock.json ./
RUN npm install

#Copy application source code
COPY . .

#Build the application
RUN npm run build

#Stage 2: Production stage
FROM node:20-alpine

#Set Working directory
WORKDIR /app

#Copy only necessary files from build stage
COPY --from=builder /app/package.json ./
COPY --from=builder /app/package-lock.json ./
COPY --from=builder /app/dist ./dist
COPY --from=builder /app/node_modules ./node_modules

#Expose the application port
EXPOSE 3000

#Start the application
CMD ["node", "dist/index.js"]
```

Step 2: Make a folder -> src and create a file -> index.js

```
jeevanc@jeevanc-IdeaPad-Slim-3-15AMN8:~/Devopslab/program2$ cat src/index.js
const express = require('express');

const app = express();

const PORT = 3000;

app.get('/', (req,res) => {
        res.send("Hello");
});

app.listen(PORT, () =>{
        console.log("Server is running");
});
```

Step 3: Create a file -> package.json



Step 4: Install Node modules with the following command

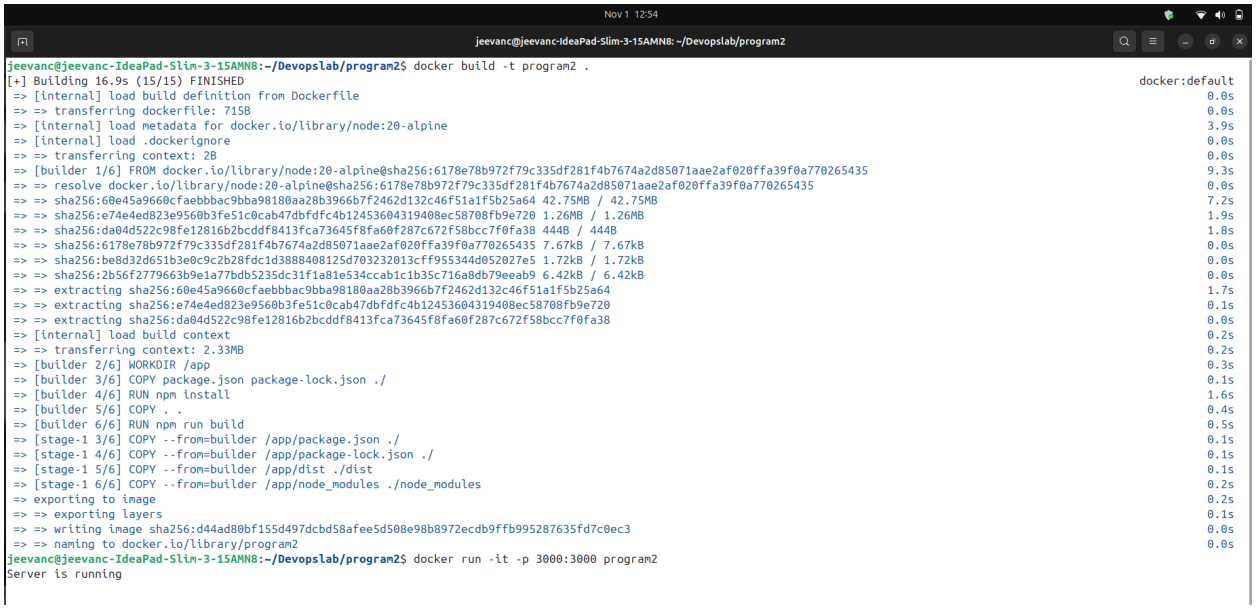
npm install

Step 5: Execute the Docker Build Command

docker build -t program2

Step 6: Run the Docker Run command specifying the port number

docker run -it -p 3000:3000 program2



Step 7: Test the container by verifying the localhost details on web server

Step 8: Stop the Docker container, remove the container

docker container ls -a

docker container stop <container-number>

docker rm <container-number>

docker rmi <image-id>