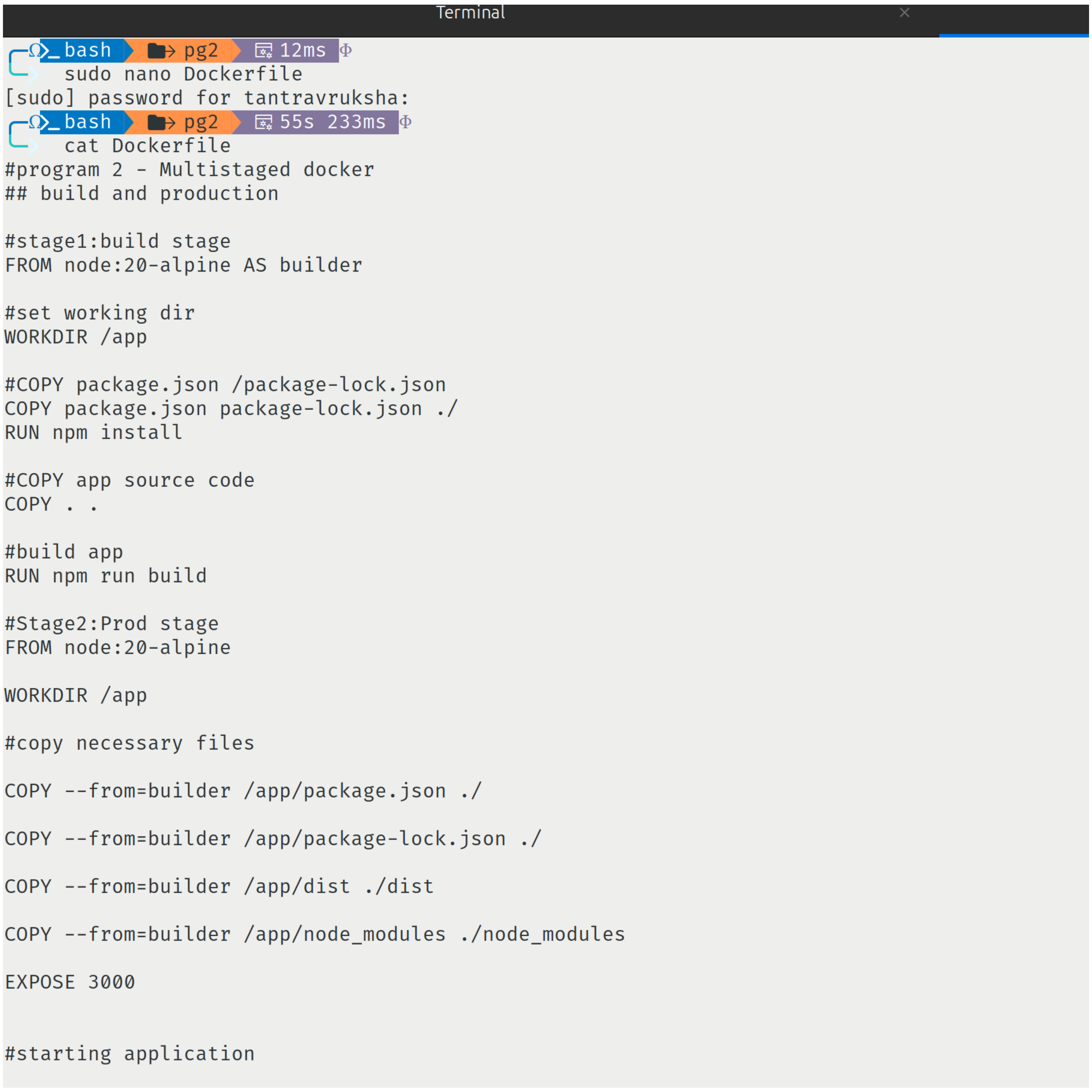
PROGRAM 2

Develop a Multi-Stage Dockerfile for Container Orchestration.

Program structure

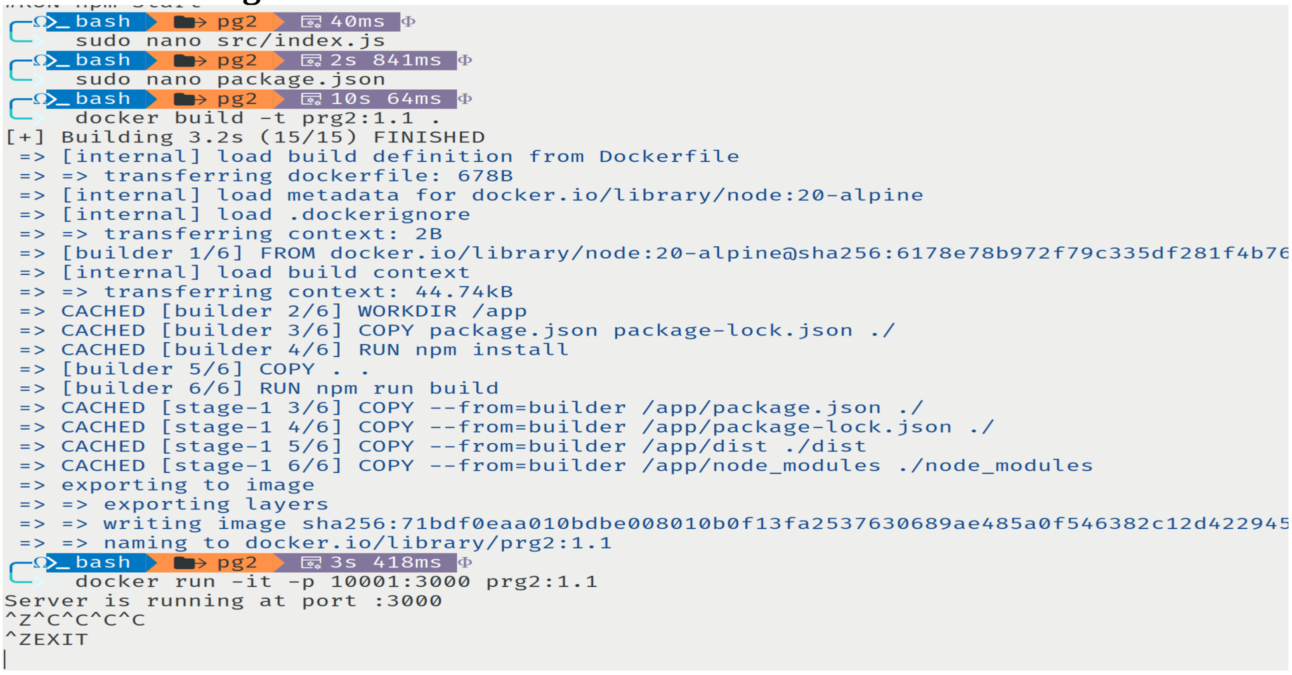
-Dockerfile
-src/index.json
-package.json
-package_lock.json
-node_modules

Step 1 : Create the following files inside a folder (DockerFile and src/index.js)



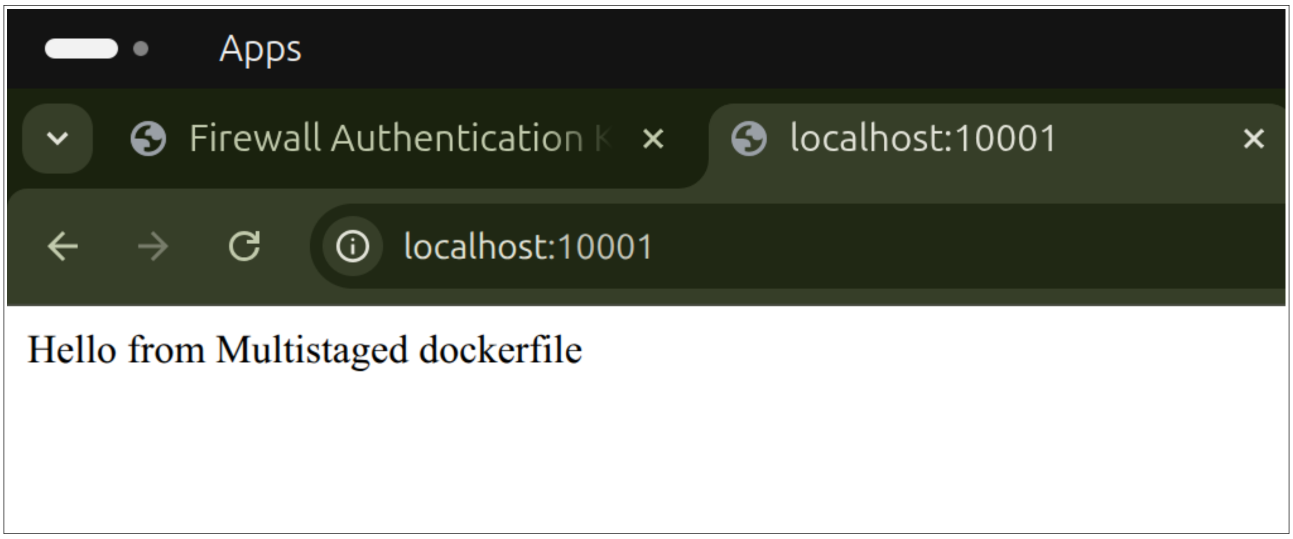
```
Ω>_ bash  pg2  12ms
   sudo nano Dockerfile
[sudo] password for tantravruksha:
Ω>_ bash  pg2  55s 233ms
   cat Dockerfile
#program 2 - Multistaged docker
## build and production

#stage1:build stage
FROM node:20-alpine AS builder

#set working dir
WORKDIR /app

#COPY package.json /package-lock.json
COPY package.json package-lock.json ./
RUN npm install

#COPY app source code
COPY . .

#build app
RUN npm run build

#Stage2:Prod stage
FROM node:20-alpine

WORKDIR /app

#copy necessary files

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 3000


#starting application
```

2.build the image and run the container



```
  bash   pg2   40ms
  sudo nano src/index.js
  bash   pg2   2s 841ms
  sudo nano package.json
  bash   pg2   10s 64ms
  docker build -t prg2:1.1 .
[+] Building 3.2s (15/15) FINISHED
 => [internal] load build definition from Dockerfile
 => => transferring dockerfile: 678B
 => [internal] load metadata for docker.io/library/node:20-alpine
 => [internal] load .dockerignore
 => => transferring context: 2B
 => [builder 1/6] FROM docker.io/library/node:20-alpine@sha256:6178e78b972f79c335df281f4b76
 => [internal] load build context
 => => transferring context: 44.74kB
 => CACHED [builder 2/6] WORKDIR /app
 => CACHED [builder 3/6] COPY package.json package-lock.json ./
 => CACHED [builder 4/6] RUN npm install
 => [builder 5/6] COPY . .
 => [builder 6/6] RUN npm run build
 => CACHED [stage-1 3/6] COPY --from=builder /app/package.json ./
 => CACHED [stage-1 4/6] COPY --from=builder /app/package-lock.json ./
 => CACHED [stage-1 5/6] COPY --from=builder /app/dist ./dist
 => CACHED [stage-1 6/6] COPY --from=builder /app/node_modules ./node_modules
 => exporting to image
 => => exporting layers
 => => writing image sha256:71bdf0eaa010bdbe008010b0f13fa2537630689ae485a0f546382c12d422945
 => => naming to docker.io/library/prg2:1.1
  bash   pg2   3s 418ms
  docker run -it -p 10001:3000 prg2:1.1
Server is running at port :3000
^Z^C^C^C^C
^ZEXIT
```

3.Final output on Browser



Hello from Multistaged dockerfile