DevOps Lab Documentation

Program 2:

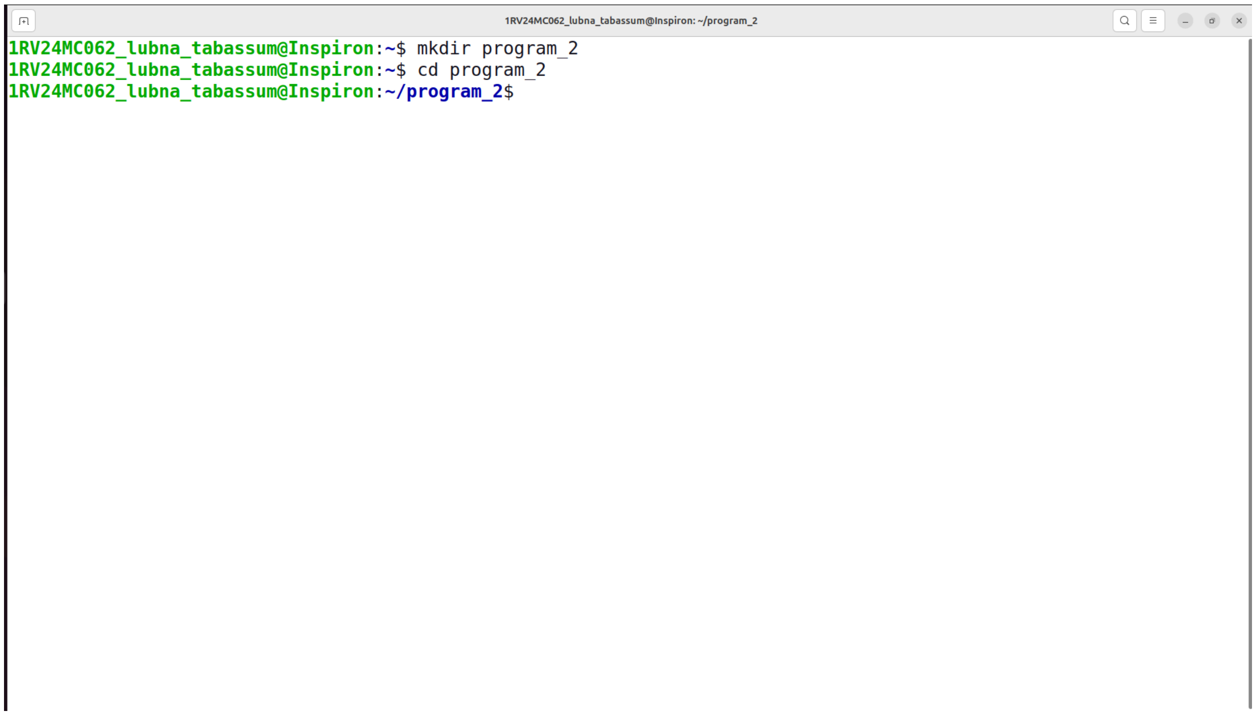
Develop a Multi-Stage Dockerfile for Container Orchestration

Project Structure:

program_2/
 —>Dockerfile
 —>package.json
 —>package-lock.json
 —>node_modules
 —>src/
        —>index.js

Procedure:

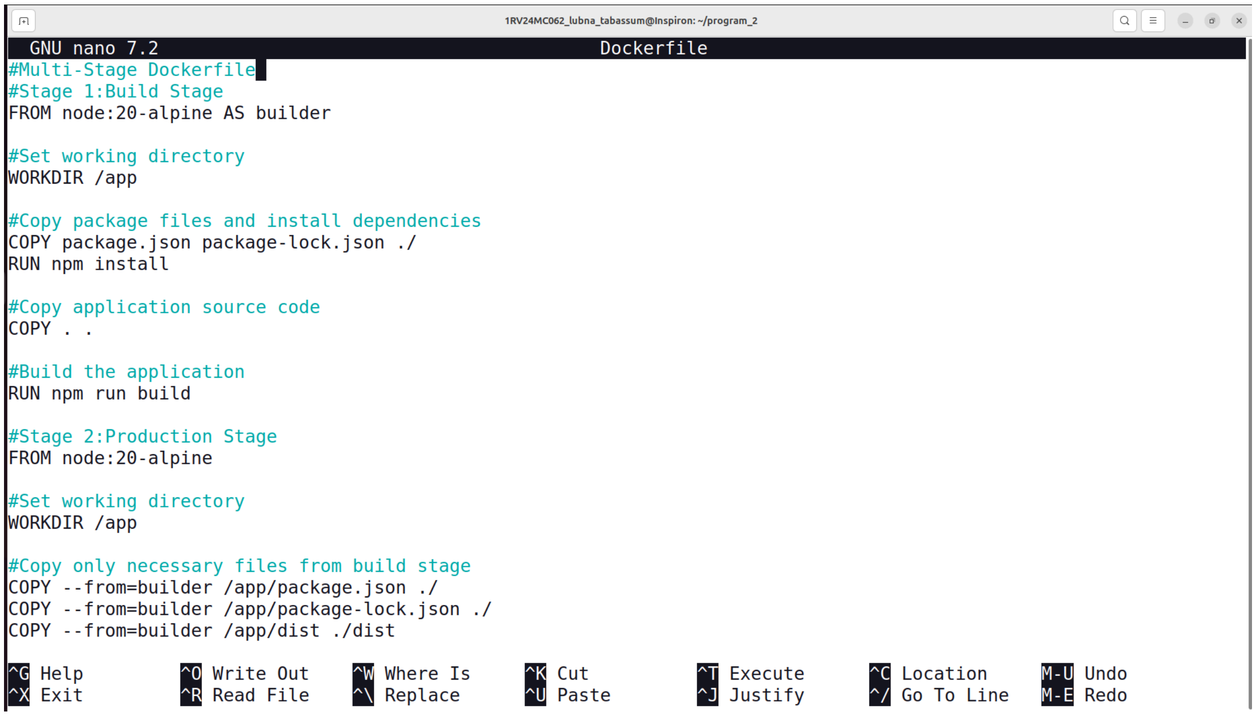
Create a folder (ex: program_2



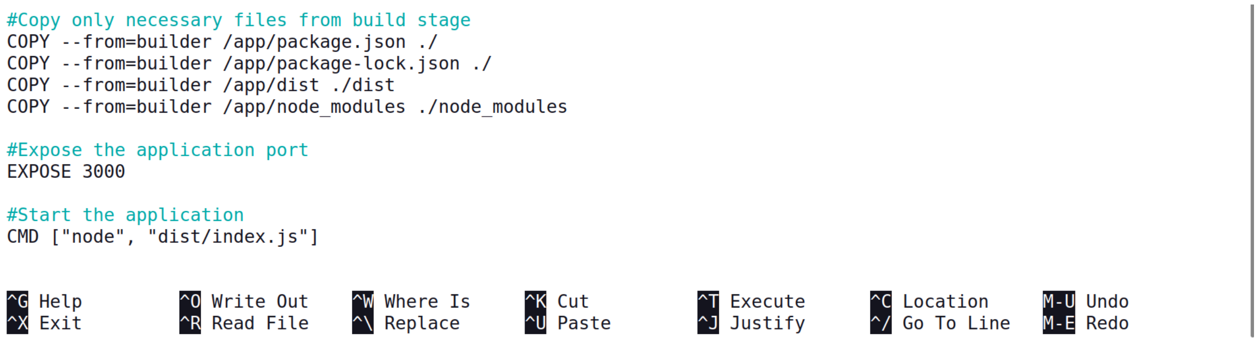
```
1RV24MC062_lubna_tabassum@Inspiron:~$ mkdir program_2
1RV24MC062_lubna_tabassum@Inspiron:~$ cd program_2
1RV24MC062_lubna_tabassum@Inspiron:~/program_2$
```

Step 1: Create a **Dockerfile** without any extensions and add the following content

sudo nano Dockerfile



```
1RV24MC062_lubna_tabassum@Inspiron: ~/program_2
  GNU nano 7.2                                        Dockerfile
#Multi-Stage 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

^G Help          ^O Write Out     ^W Where Is      ^K Cut           ^T Execute       ^C Location      M-U Undo
^X Exit          ^R Read File     ^\ Replace       ^U Paste         ^J Justify       ^/ Go To Line    M-E Redo
```

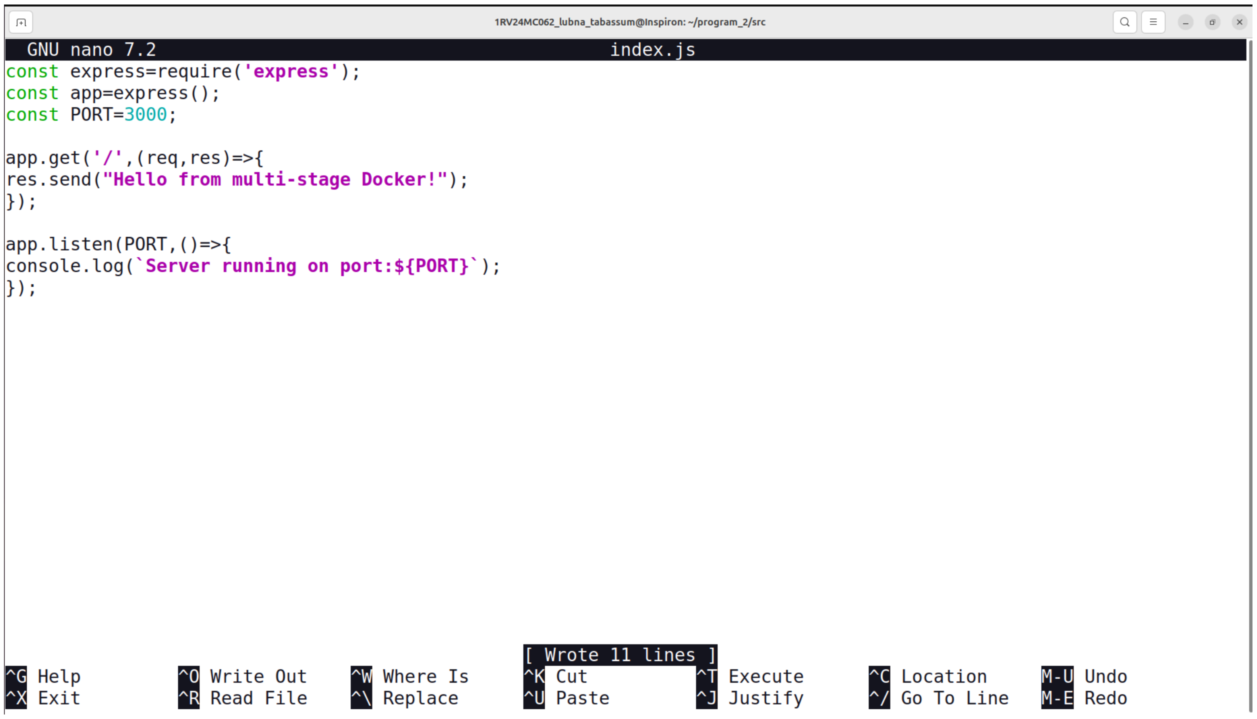


```
#Copy only necessary files from build stage
COPY --from=builder /app/package.json ./
COPY --from=builder /app/package-lock.json ./
COPY --from=builder /app/dist ./dist
COPY --from=builder /app/node_modules ./node_modules

#Expose the application port
EXPOSE 3000

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


^G Help          ^O Write Out     ^W Where Is      ^K Cut           ^T Execute       ^C Location      M-U Undo
^X Exit          ^R Read File     ^\ Replace       ^U Paste         ^J Justify       ^/ Go To Line    M-E Redo
```

Step 2: Create a index file- index.js with following code

cd src

nano index.js


1RV24MC062_lubna_tabassum@Inspiron: ~/program_2/src

```
 GNU nano 7.2                                         index.js
const express=require('express');
const app=express();
const PORT=3000;

app.get('/',(req,res)=>{
res.send("Hello from multi-stage Docker!");
});

app.listen(PORT,()=>{
console.log(`Server running on port:${PORT}`);
});

                                                  [ Wrote 11 lines ]
^G Help        ^O Write Out   ^W Where Is    ^K Cut         ^T Execute     ^C Location    M-U Undo
^X Exit        ^R Read File   ^\ Replace     ^U Paste       ^J Justify     ^/ Go To Line  M-E Redo
```

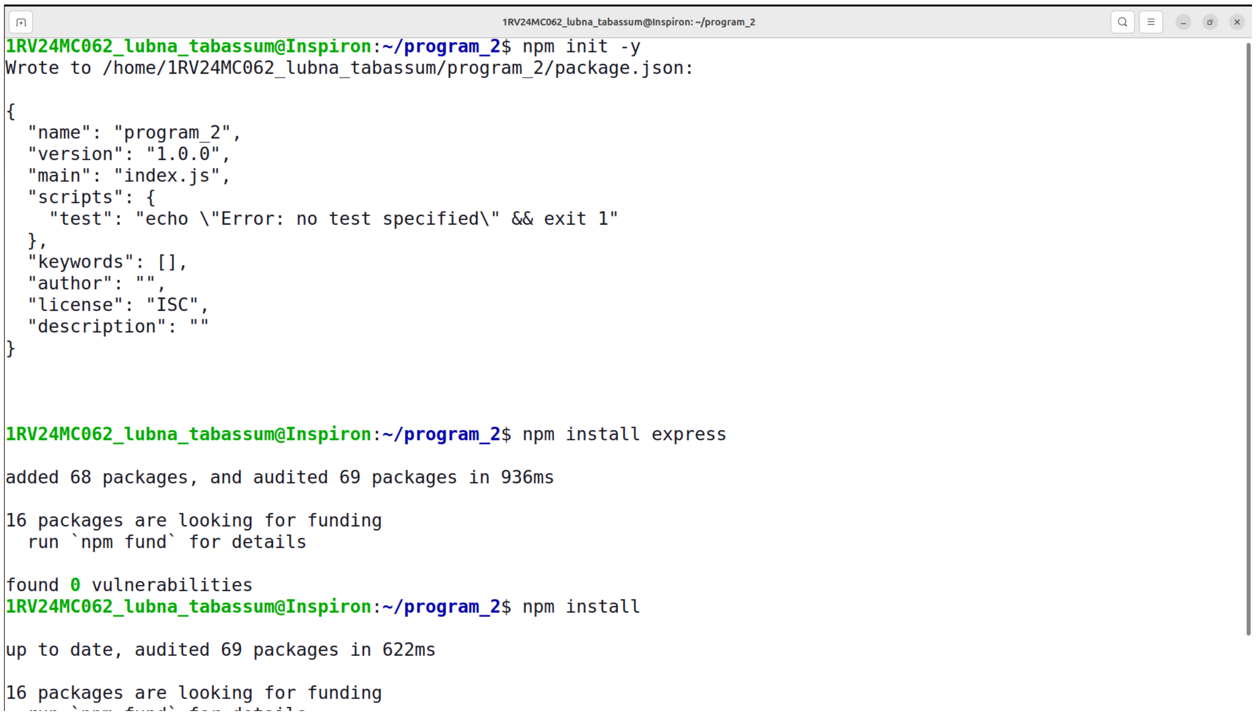
Step 3: run the following commands

npm init -y

(node_modules will be automatically created

npm install express

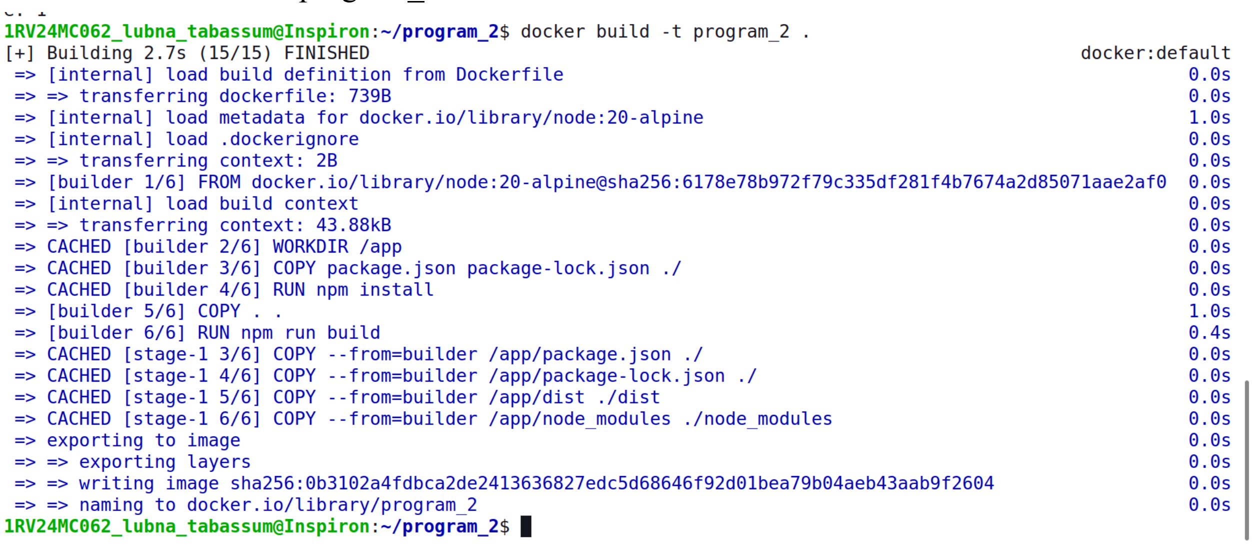
npm install



```
1RV24MC062_lubna_tabassum@Inspiron:~/program_2$ npm init -y
Wrote to /home/1RV24MC062_lubna_tabassum/program_2/package.json:

{
  "name": "program_2",
  "version": "1.0.0",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "keywords": [],
  "author": "",
  "license": "ISC",
  "description": ""
}



1RV24MC062_lubna_tabassum@Inspiron:~/program_2$ npm install express

added 68 packages, and audited 69 packages in 936ms

16 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities
1RV24MC062_lubna_tabassum@Inspiron:~/program_2$ npm install

up to date, audited 69 packages in 622ms

16 packages are looking for funding
```

Step 4: Execute the Docker Build command

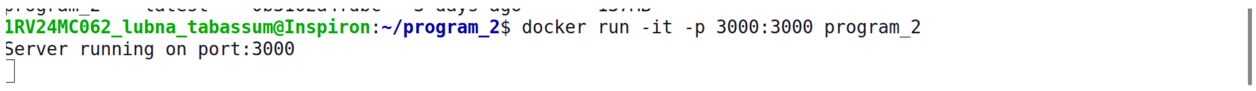
docker build -t program_2 .



```
1RV24MC062_lubna_tabassum@Inspiron:~/program_2$ docker build -t program_2 .
[+] Building 2.7s (15/15) FINISHED                                                        docker:default
 => [internal] load build definition from Dockerfile                                               0.0s
 => => transferring dockerfile: 739B                                                               0.0s
 => [internal] load metadata for docker.io/library/node:20-alpine                                  1.0s
 => [internal] load .dockerignore                                                                  0.0s
 => => transferring context: 2B                                                                    0.0s
 => [builder 1/6] FROM docker.io/library/node:20-alpine@sha256:6178e78b972f79c335df281f4b7674a2d85071aae2af0  0.0s
 => [internal] load build context                                                                  0.0s
 => => transferring context: 43.88kB                                                               0.0s
 => CACHED [builder 2/6] WORKDIR /app                                                              0.0s
 => CACHED [builder 3/6] COPY package.json package-lock.json ./                                    0.0s
 => CACHED [builder 4/6] RUN npm install                                                           0.0s
 => [builder 5/6] COPY . .                                                                         1.0s
 => [builder 6/6] RUN npm run build                                                                0.4s
 => CACHED [stage-1 3/6] COPY --from=builder /app/package.json ./                                  0.0s
 => CACHED [stage-1 4/6] COPY --from=builder /app/package-lock.json ./                             0.0s
 => CACHED [stage-1 5/6] COPY --from=builder /app/dist ./dist                                      0.0s
 => CACHED [stage-1 6/6] COPY --from=builder /app/node_modules ./node_modules                      0.0s
 => exporting to image                                                                             0.0s
 => => exporting layers                                                                            0.0s
 => => writing image sha256:0b3102a4fdbca2de2413636827edc5d68646f92d01bea79b04aeb43aab9f2604       0.0s
 => => naming to docker.io/library/program_2                                                       0.0s
1RV24MC062_lubna_tabassum@Inspiron:~/program_2$ 
```

Step 5: Run the docker command and specify the port number

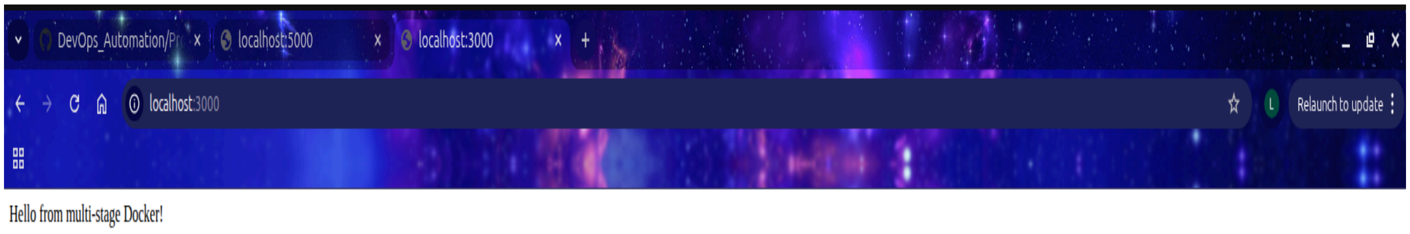
docker run -d -p 3000:3000 program_1



```
1RV24MC062_lubna_tabassum@Inspiron:~/program_2$ docker run -it -p 3000:3000 program_2
Server running on port:3000
```

Step 6: Verify localhost details on browser

http://localhost:3000



Hello from multi-stage Docker!