DevOps Lab

Program 2- Creating a Multi-Stage Dockerfile

Project Structure:

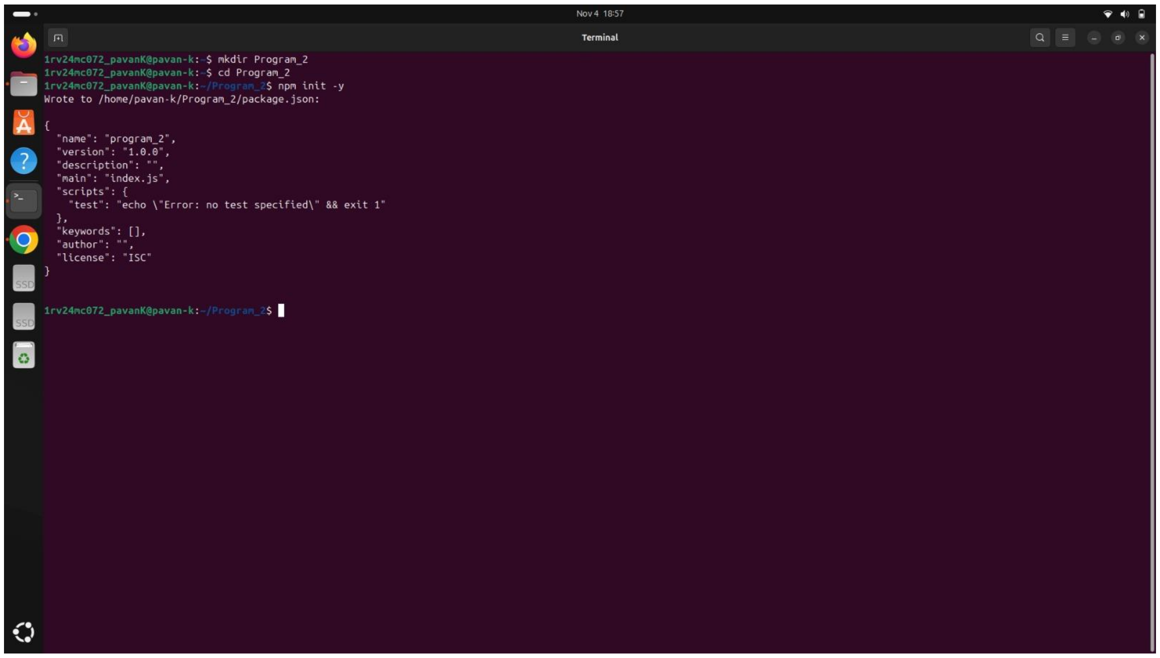
```
program_2
  Dockerfile
  package.json
  package-lock.json
  node_modules/
  src/
    index.js
```

Step 1: Create Project Folder

```
mkdir program_2
cd program_2
```

Step 2: Initialize Node.js Project

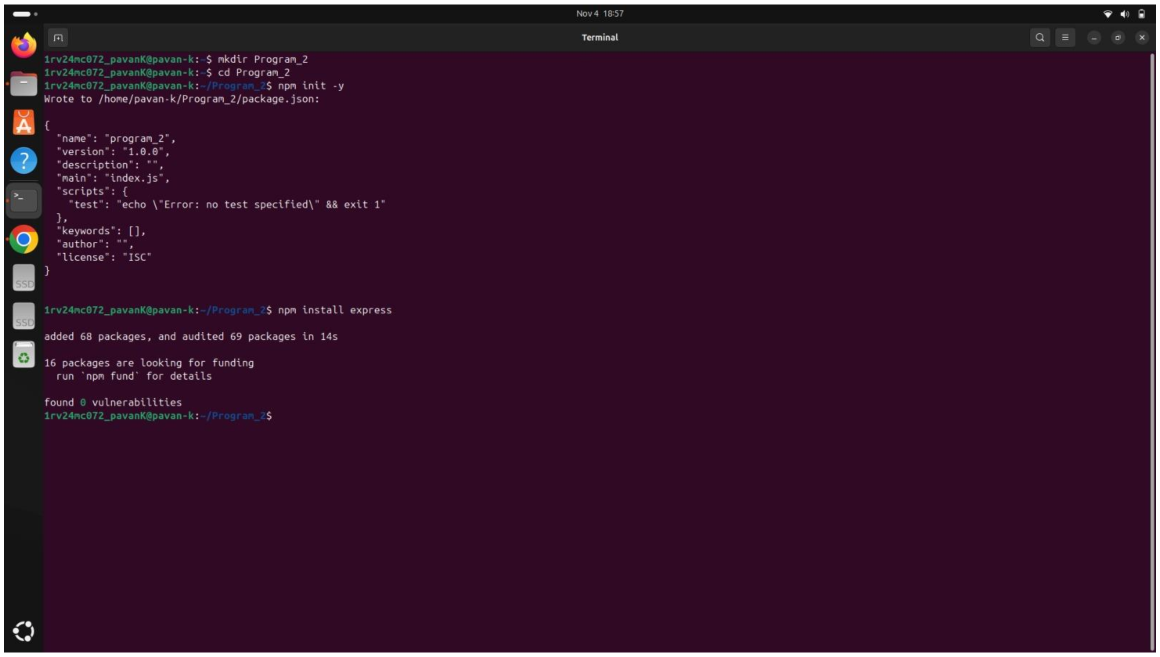
```
npm init -y
```



```
1rv24mc072_pavanK@pavan-k:~$ mkdir Program_2
1rv24mc072_pavanK@pavan-k:~$ cd Program_2
1rv24mc072_pavanK@pavan-k:~/Program_2$ npm init -y
Wrote to /home/pavan-k/Program_2/package.json:

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


1rv24mc072_pavanK@pavan-k:~/Program_2$
```

Step 3: Install Express Framework

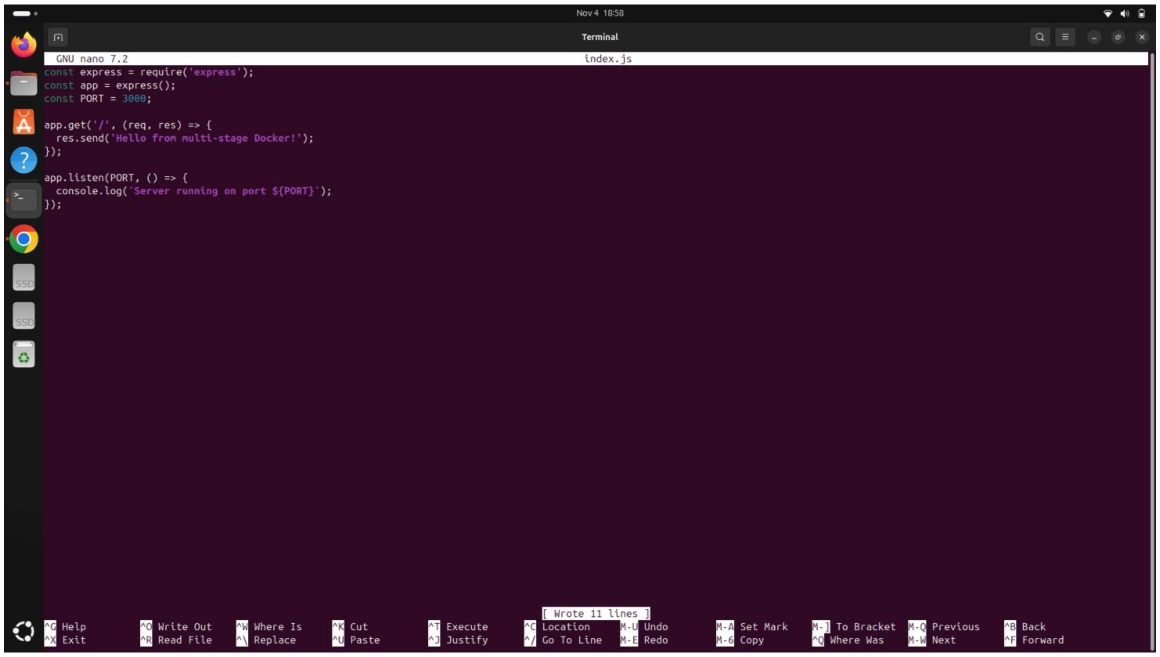
```
npm install express
```



```
1rv24mc072_pavanK@pavan-k:~$ mkdir Program_2
1rv24mc072_pavanK@pavan-k:~$ cd Program_2
1rv24mc072_pavanK@pavan-k:~/Program_2$ npm init -y
Wrote to /home/pavan-k/Program_2/package.json:

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


1rv24mc072_pavanK@pavan-k:~/Program_2$ npm install express

added 68 packages, and audited 69 packages in 14s

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

found 0 vulnerabilities
1rv24mc072_pavanK@pavan-k:~/Program_2$
```

Step 4: Create Source Folder and Application File

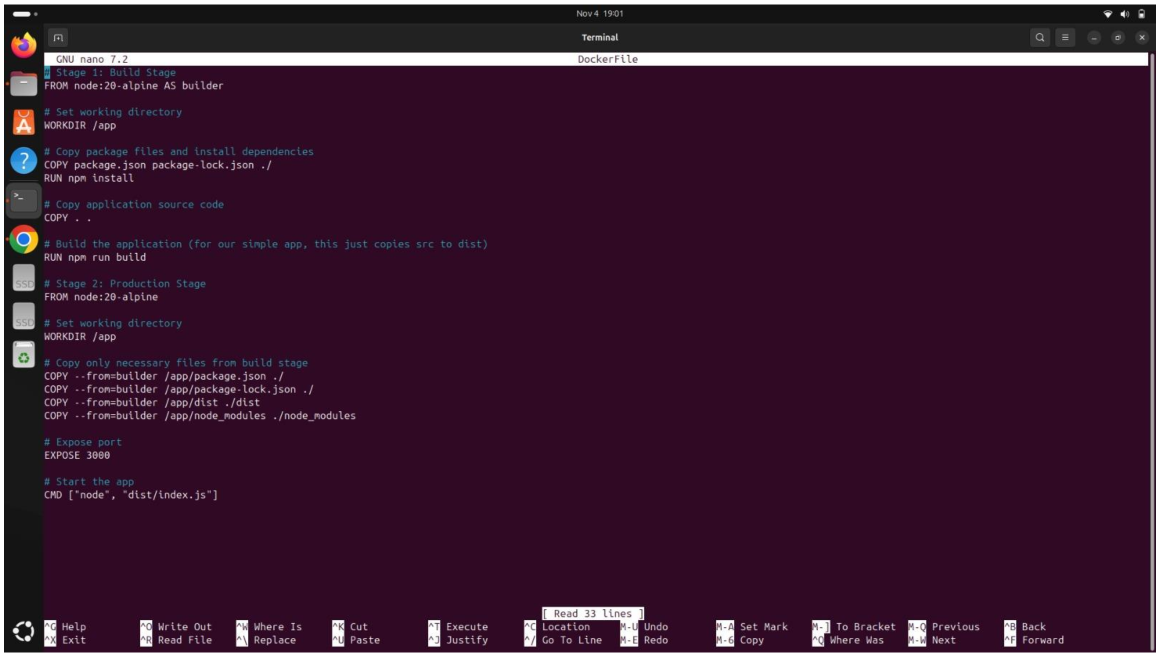
mkdir src
cd src
nano index.js



```
  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      M-A Set Mark  M-] To Bracket  M-Q Previous  ^B Back
^X Exit       ^R Read File  ^\ Replace    ^U Paste      ^J Justify    ^/ Go To Line M-E Redo      M-6 Copy      ^Q Where Was    M-W Next      ^F Forward
```

Step 5: Create the Multi-Stage Dockerfile

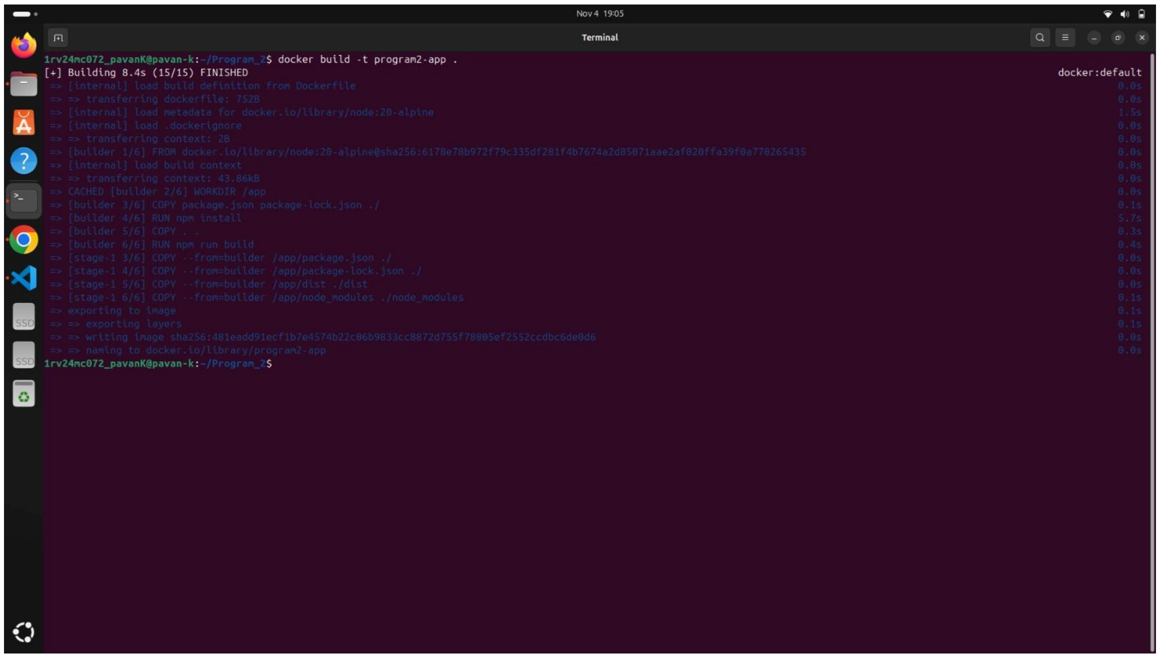
cd ..
nano Dockerfile



```
GNU nano 7.2                                   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 (for our simple app, this just copies src to dist)
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 port
EXPOSE 3000

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

                                           [ Read 33 lines ]
^G Help      ^O Write Out  ^W Where Is   ^K Cut      ^T Execute  ^C Location  M-U Undo   M-A Set Mark  M-] To Bracket  M-Q Previous  ^B Back
^X Exit      ^R Read File  ^\ Replace    ^U Paste    ^J Justify  ^/ Go To Line M-E Redo  M-6 Copy      ^Q Where Was    M-W Next      ^F Forward
```

Step 6: Build the Docker Image

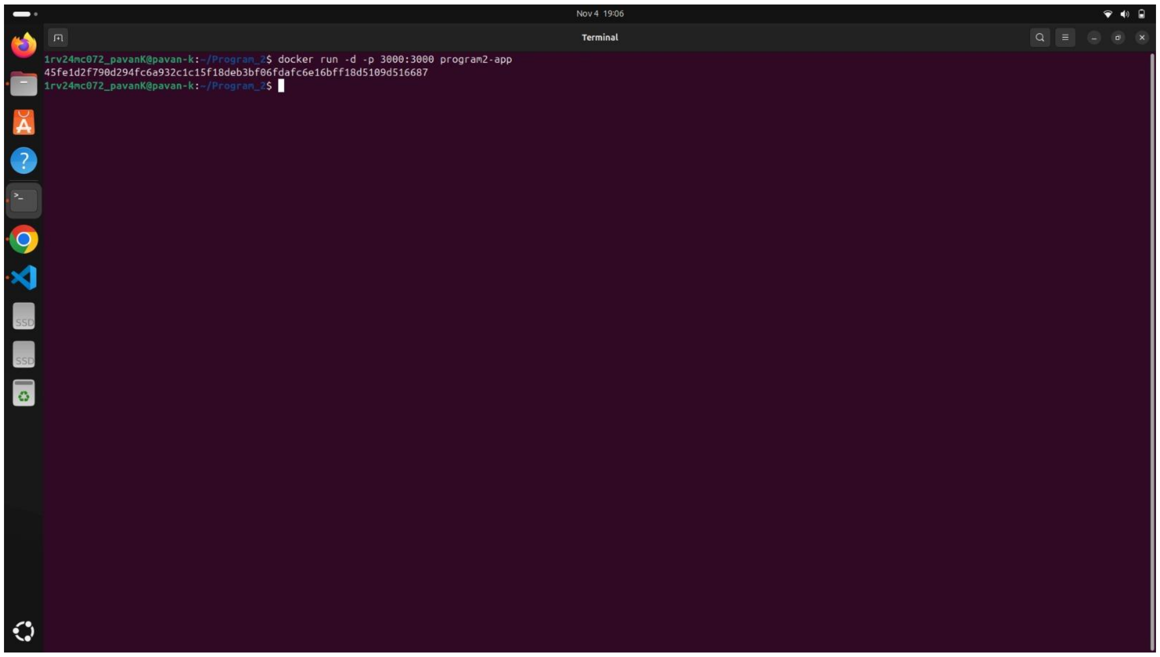
docker build -t program2-app .



```
1rv24mc072_pavanK@pavan-k:~/Program_2$ docker build -t program2-app .
[+] Building 8.4s (15/15) FINISHED                                                                                  docker:default
 => [internal] load build definition from Dockerfile                                                                         0.0s
 => => transferring dockerfile: 752B                                                                                         0.0s
 => [internal] load metadata for docker.io/library/node:20-alpine                                                            1.5s
 => [internal] load .dockerignore                                                                                            0.0s
 => => transferring context: 2B                                                                                              0.0s
 => [builder 1/6] FROM docker.io/library/node:20-alpine@sha256:6178e78b972f79c335df281f4b7674a2d85071aae2af020ffa39f0a770265435 0.0s
 => [internal] load build context                                                                                            0.0s
 => => transferring context: 43.86kB                                                                                         0.0s
 => CACHED [builder 2/6] WORKDIR /app                                                                                        0.0s
 => [builder 3/6] COPY package.json package-lock.json ./                                                                     0.1s
 => [builder 4/6] RUN npm install                                                                                            5.7s
 => [builder 5/6] COPY . .                                                                                                   0.3s
 => [builder 6/6] RUN npm run build                                                                                          0.4s
 => [stage-1 3/6] COPY --from=builder /app/package.json ./                                                                   0.0s
 => [stage-1 4/6] COPY --from=builder /app/package-lock.json ./                                                              0.0s
 => [stage-1 5/6] COPY --from=builder /app/dist ./dist                                                                       0.0s
 => [stage-1 6/6] COPY --from=builder /app/node_modules ./node_modules                                                       0.1s
 => exporting to image                                                                                                       0.1s
 => => exporting layers                                                                                                      0.1s
 => => writing image sha256:481eadd91ecf1b7e4574b22c06b9833cc8872d755f78005ef2552ccdbc6de0d6                                 0.0s
 => => naming to docker.io/library/program2-app                                                                              0.0s
1rv24mc072_pavanK@pavan-k:~/Program_2$
```

Step 7: Run the Docker Container

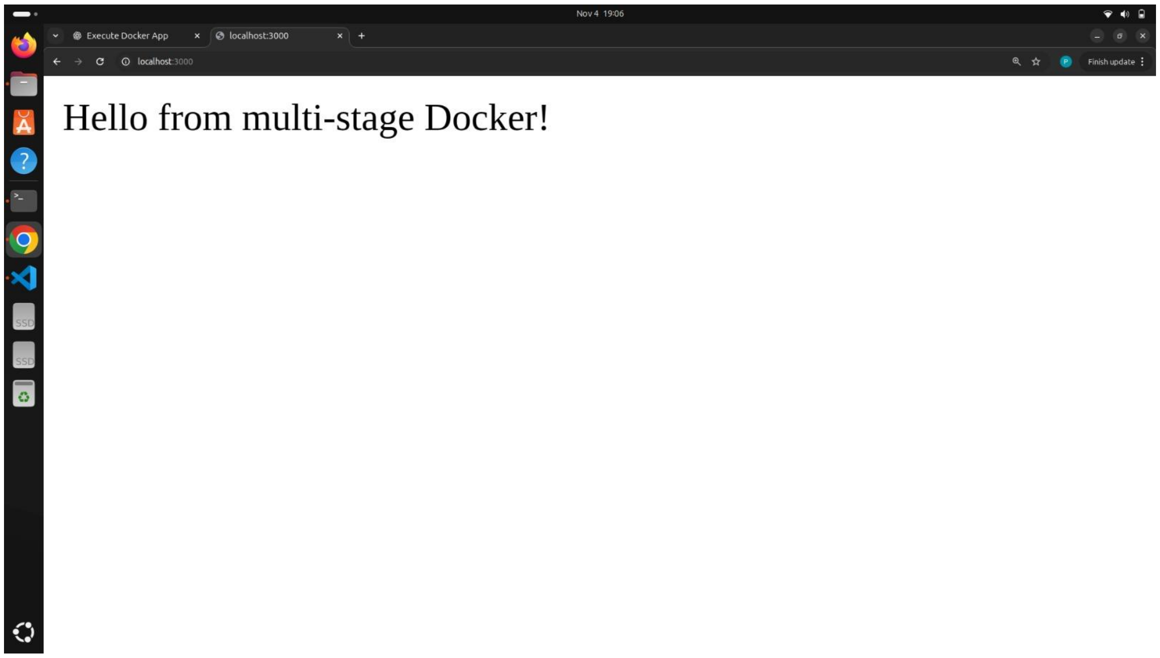
docker run -d -p 3000:3000 program2-app



```
1rv24mc072_pavanK@pavan-k:~/Program_2$ docker run -d -p 3000:3000 program2-app
45fe1d2f790d294fc6a932c1c15f18deb3bf06fdafc6e16bff18d5109d516687
1rv24mc072_pavanK@pavan-k:~/Program_2$ 
```

Step 8: Verify the Application

Open http://localhost:3000



Hello from multi-stage Docker!