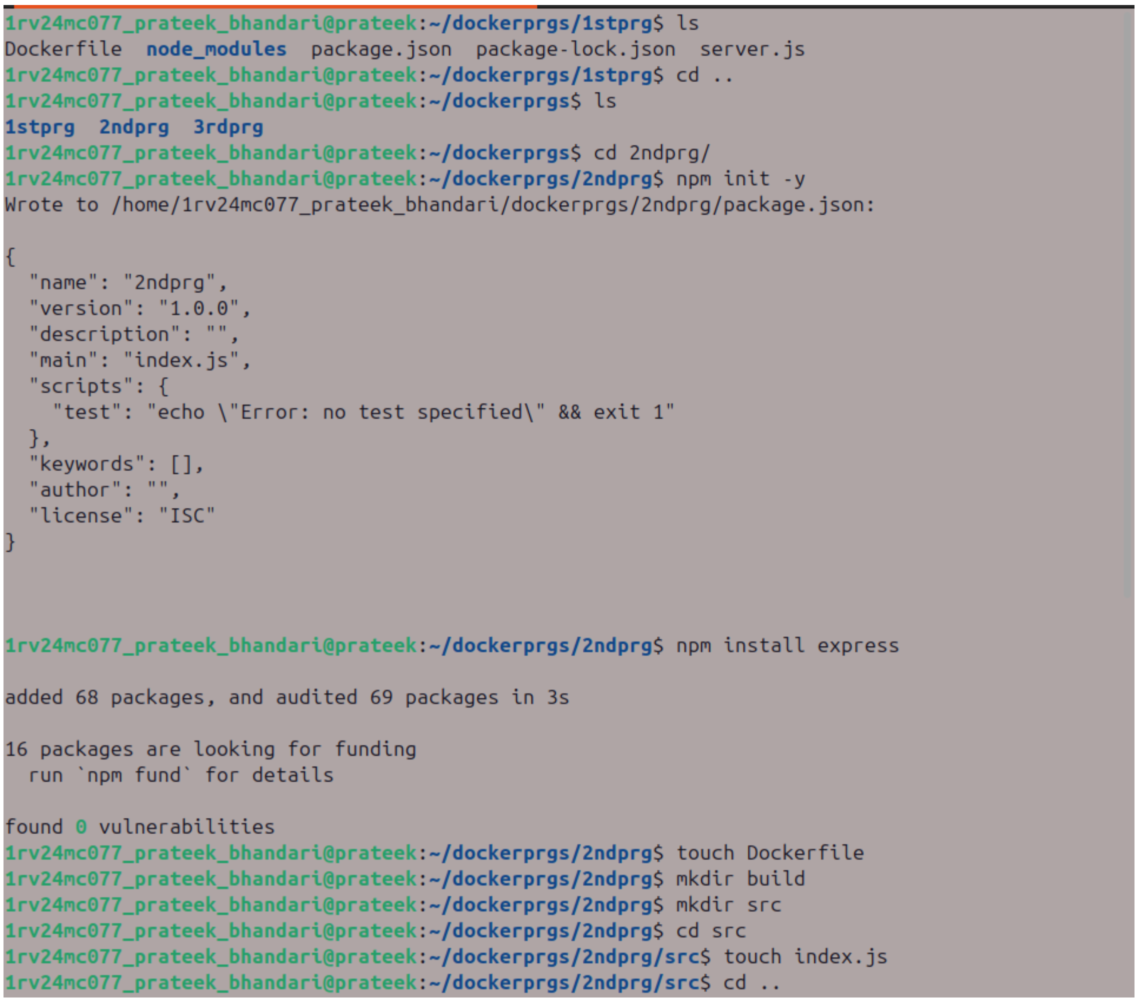
Program 2

step 1: Initialize the project

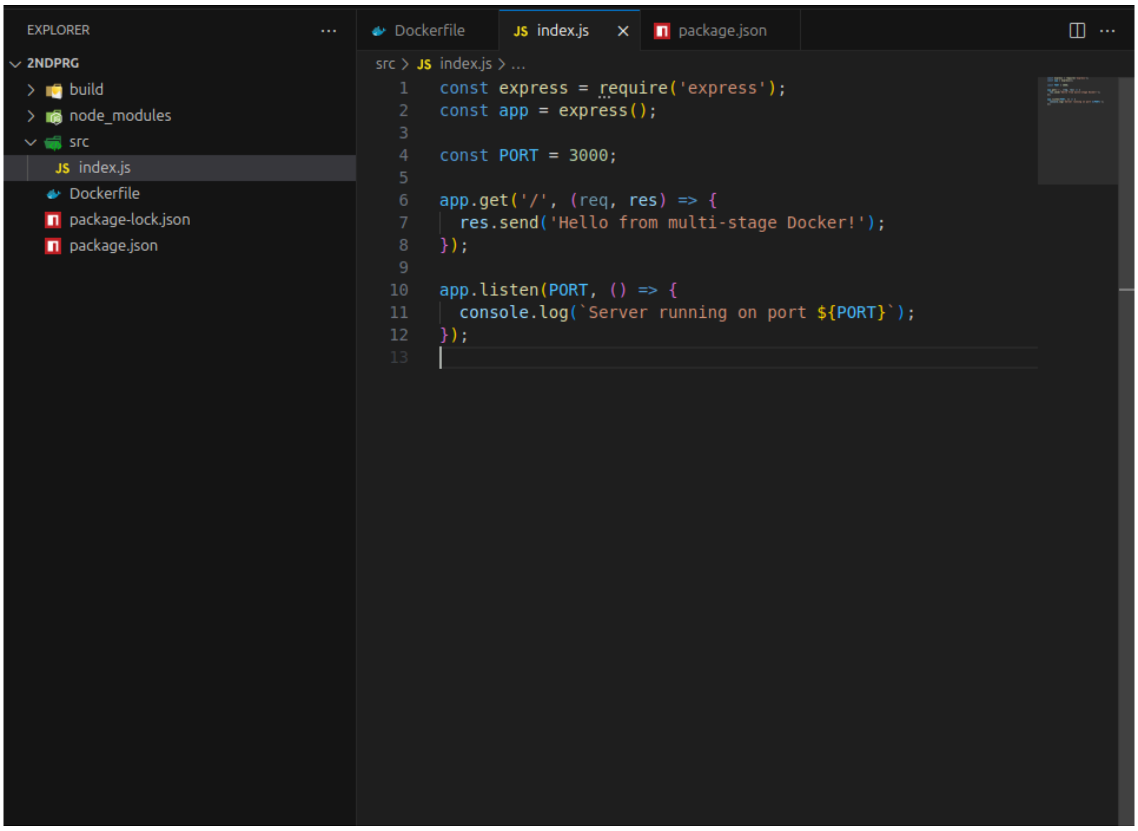
step 2 : Install Express

step 3 : Create a folder named `src` and add `index.js`



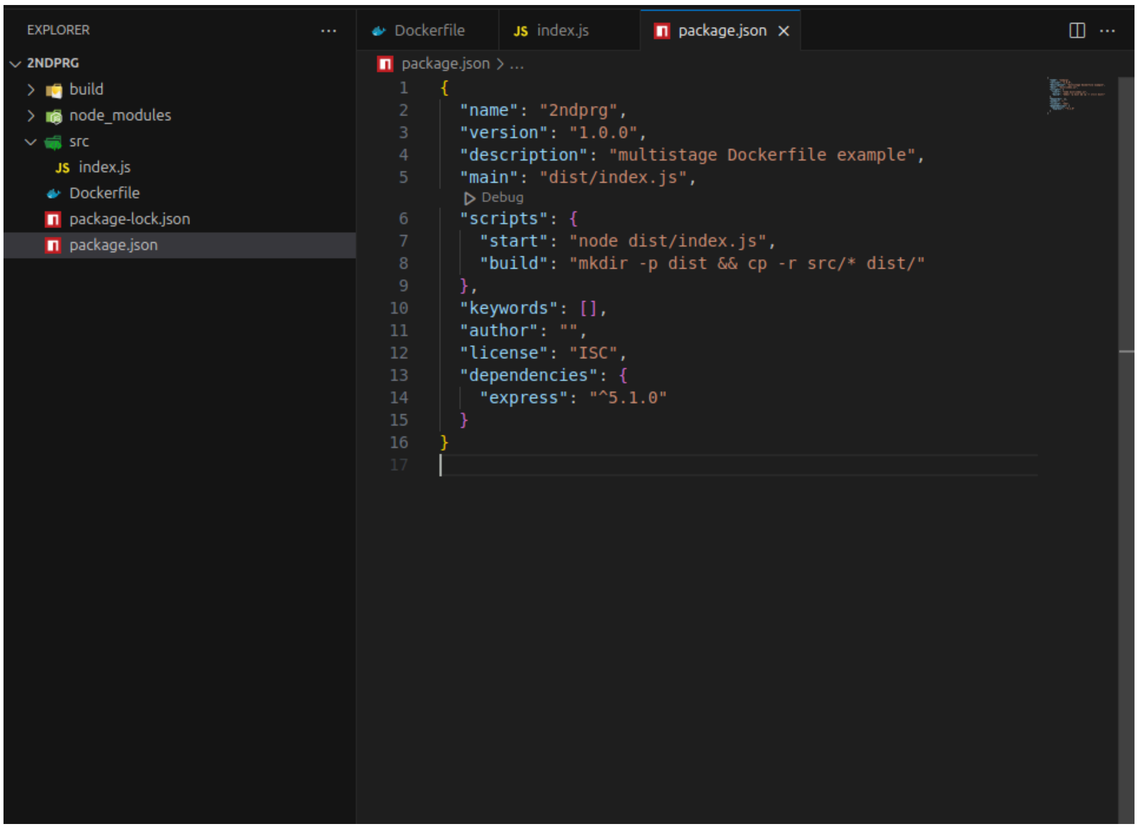
```
1rv24mc077_prateek_bhandari@prateek:~/dockerprgs/1stprg$ ls
Dockerfile  node_modules  package.json  package-lock.json  server.js
1rv24mc077_prateek_bhandari@prateek:~/dockerprgs/1stprg$ cd ..
1rv24mc077_prateek_bhandari@prateek:~/dockerprgs$ ls
1stprg  2ndprg  3rdprg
1rv24mc077_prateek_bhandari@prateek:~/dockerprgs$ cd 2ndprg/
1rv24mc077_prateek_bhandari@prateek:~/dockerprgs/2ndprg$ npm init -y
Wrote to /home/1rv24mc077_prateek_bhandari/dockerprgs/2ndprg/package.json:

{
  "name": "2ndprg",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "keywords": [],
  "author": "",
  "license": "ISC"
}



1rv24mc077_prateek_bhandari@prateek:~/dockerprgs/2ndprg$ npm install express

added 68 packages, and audited 69 packages in 3s

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

found 0 vulnerabilities
1rv24mc077_prateek_bhandari@prateek:~/dockerprgs/2ndprg$ touch Dockerfile
1rv24mc077_prateek_bhandari@prateek:~/dockerprgs/2ndprg$ mkdir build
1rv24mc077_prateek_bhandari@prateek:~/dockerprgs/2ndprg$ mkdir src
1rv24mc077_prateek_bhandari@prateek:~/dockerprgs/2ndprg$ cd src
1rv24mc077_prateek_bhandari@prateek:~/dockerprgs/2ndprg/src$ touch index.js
1rv24mc077_prateek_bhandari@prateek:~/dockerprgs/2ndprg/src$ cd ..
```

step 4 : write index.js



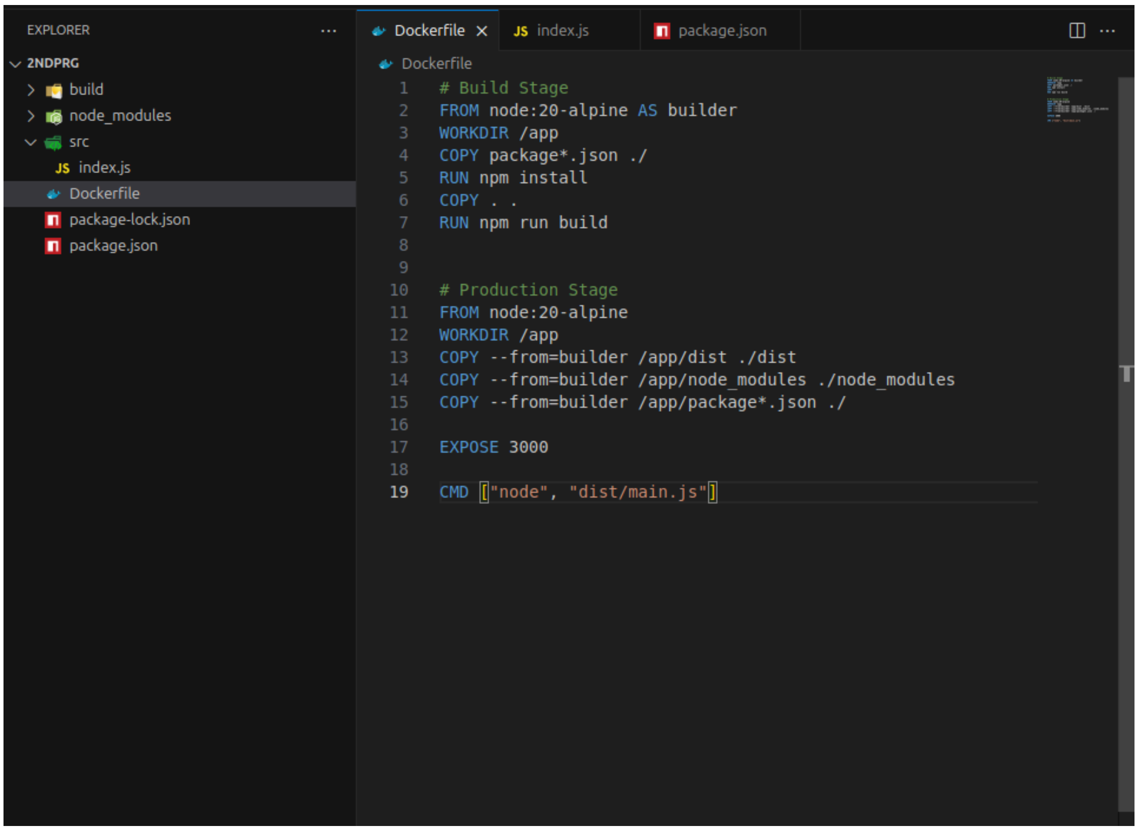
```javascript
const express = require('express');
const app = express();

const PORT = 3000;

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

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

step 5 : Configure package.json



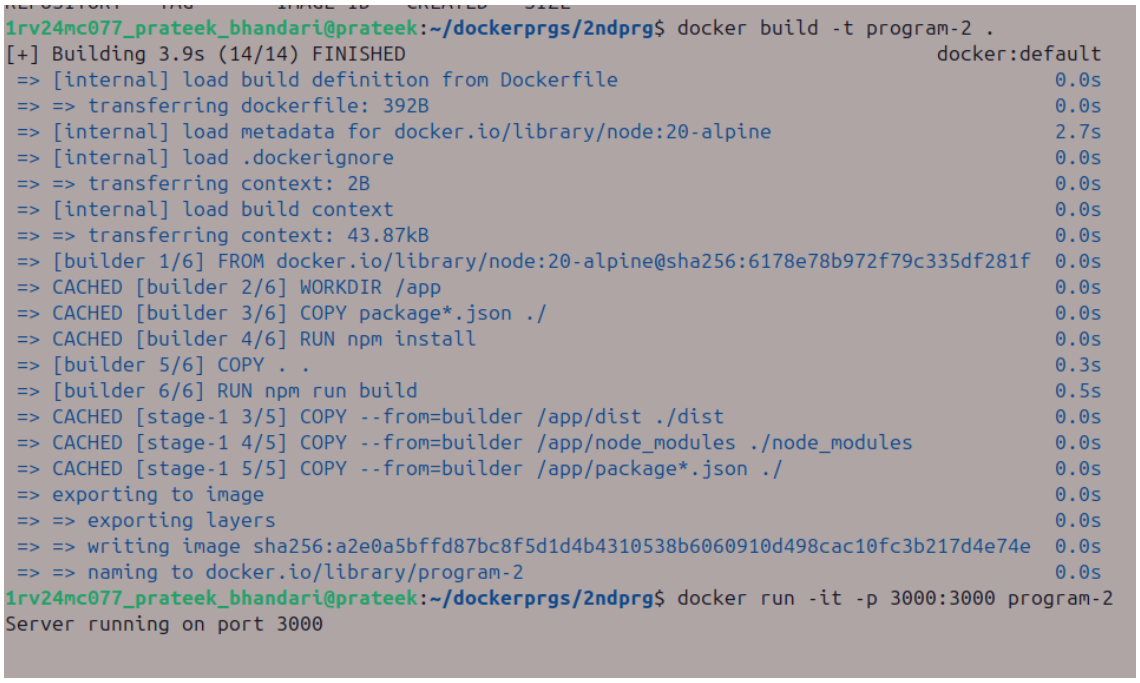
```json
{
  "name": "2ndprg",
  "version": "1.0.0",
  "description": "multistage Dockerfile example",
  "main": "dist/index.js",
  "scripts": {
    "start": "node dist/index.js",
    "build": "mkdir -p dist && cp -r src/* dist/"
  },
  "keywords": [],
  "author": "",
  "license": "ISC",
  "dependencies": {
    "express": "^5.1.0"
  }
}
```

step 6 : Create the Multi-Stage Dockerfile



```
# Build Stage
FROM node:20-alpine AS builder
WORKDIR /app
COPY package*.json ./
RUN npm install
COPY . .
RUN npm run build


# Production Stage
FROM node:20-alpine
WORKDIR /app
COPY --from=builder /app/dist ./dist
COPY --from=builder /app/node_modules ./node_modules
COPY --from=builder /app/package*.json ./

EXPOSE 3000

CMD ["node", "dist/main.js"]
```

step 7 : Build the Docker image



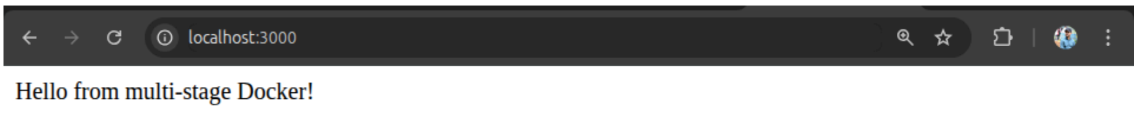
```
1rv24mc077_prateek_bhandari@prateek:~/dockerprgs/2ndprg$ docker build -t program-2 .
[+] Building 3.9s (14/14) FINISHED                                       docker:default
 => [internal] load build definition from Dockerfile                                0.0s
 => => transferring dockerfile: 392B                                               0.0s
 => [internal] load metadata for docker.io/library/node:20-alpine                   2.7s
 => [internal] load .dockerignore                                                   0.0s
 => => transferring context: 2B                                                     0.0s
 => [internal] load build context                                                   0.0s
 => => transferring context: 43.87kB                                                0.0s
 => [builder 1/6] FROM docker.io/library/node:20-alpine@sha256:6178e78b972f79c335df281f  0.0s
 => CACHED [builder 2/6] WORKDIR /app                                               0.0s
 => CACHED [builder 3/6] COPY package*.json ./                                      0.0s
 => CACHED [builder 4/6] RUN npm install                                            0.0s
 => [builder 5/6] COPY . .                                                          0.3s
 => [builder 6/6] RUN npm run build                                                 0.5s
 => CACHED [stage-1 3/5] COPY --from=builder /app/dist ./dist                       0.0s
 => CACHED [stage-1 4/5] COPY --from=builder /app/node_modules ./node_modules       0.0s
 => CACHED [stage-1 5/5] COPY --from=builder /app/package*.json ./                  0.0s
 => exporting to image                                                              0.0s
 => => exporting layers                                                             0.0s
 => => writing image sha256:a2e0a5bffd87bc8f5d1d4b4310538b6060910d498cac10fc3b217d4e74e  0.0s
 => => naming to docker.io/library/program-2                                        0.0s
1rv24mc077_prateek_bhandari@prateek:~/dockerprgs/2ndprg$ docker run -it -p 3000:3000 program-2
Server running on port 3000
```

1. **Stage 1 (builder)**
    - Installs dependencies
    - Copies source files
    - Builds the app (creates `/app/dist`)
2. **Stage 2 (production)**
    - Copies only required files (`dist`, `node_modules`, `package.json`)
    - Removes unnecessary build tools
    - Makes the image small and clean

After this, Docker produces an image named **program-2**.

step 7 :Run the Docker container

step 8 : Test on your browser



```
localhost:3000
Hello from multi-stage Docker!
```