

Node JS Program Execute on Docker

Running with Docker

1. Build the Docker image

Command: docker build -t node-express-app .

2. Run the container

Command: docker run -p 3000:3000 --env-file .env node-express-app

Your app will now be available at:

 http://localhost:3000

Running in Development Mode (with Nodemon)

If you want to enable hot-reloading (e.g., for local development):

1. Modify the Dockerfile (already set up):

```
CMD ["npm", "run", "dev"]
```

2. Rebuild and run again:

Command: docker build -t node-express-app .

Command: docker run -p 3000:3000 --env-file .env node-express-app

Stop the container

Use `Ctrl+C` in the terminal or:

Command: docker ps # find your container ID

Command: docker ps -a # find your hidden container ID

Command: docker stop <id> # stop the container

Useful Scripts without Docker

`npm run dev` — Start server with Nodemon

`npm start` — Start server normally

Docker Process:

Step1: Builds a Docker image

Command: docker build -t node-express-app .

Explanation:

Part	Meaning
docker build	Tells Docker to build a new image using a Dockerfile
-t node-express-app	Tags (names) the image as node-express-app for easy reference
. (dot)	Refers to the current directory — Docker will look for the Dockerfile here

Result: This command builds a Docker image named node-express-app using the instructions in your Dockerfile.

```
PS E:\pavan\learn\Node JS\NodeJSBasicToAdvance\node-with-docker-sample1> docker build -t node-express-app .
[+] Building 17.5s (11/11) FINISHED
=> [internal] load build context from Dockerfile
=> [internal] load metadata from Dockerfile
=> transferring dockerfile: 487B
=> [internal] load metadata for docker.io/library/node:18-alpine
=> [internal] library/node:pulling from registry-1.docker.io
=> [internal] load manifest for registry-1.docker.io
=> [internal] load context: 181B
=> [1/1] FROM docker.io/library/node:18-alpine@sha256:8d6421d663b4c28fd3ebc498332f249011d118945588d0a35cb9bc4b8ca99d9e
=> => resolve docker.io/library/node:18-alpine@sha256:8d6421d663b4c28fd3ebc498332f249011d118945588d0a35cb9bc4b8ca99d9e
=> => sha256:929b0uld7c782404f615cf785488fe4d52b6569f87c73f666ad553a7554f0906 1.72kB / 1.72kB
=> => sha256:ee77cc6cd7c1886ecc802ad6ccedef3a8ec1ea27d1fb96162fb03dd3710839b8da 6.18kB / 6.18kB
=> => sha256:f18232174bc91741fd4f3da96d8501092101a032a93a388b79e99e69c2d5c870 3.6UMB / 3.6UMB
=> => sha256:dd71d0e834b5c783d162992e6fb8991ch2309ae049a0eabc4efea161b2b5a2d9e 40.81MB / 40.81MB
=> => sha256:1e5a4c89cce5c0826e540ab664b6491c96eda01837f430bd47f7fd26782d6e3 1.26MB / 1.26MB
=> => sha256:8d6421d663b4c28fd3ebc498332f249011d118945588d0a35cb9bc4b8ca99d9e 7.67kB / 7.67kB
=> => sha256:25ff2da83641968f65c3a74d889496b1b62ccfaab220b9ea70b80df5a2e0549 446kB / 446kB
=> => sha256:f18232174bc91741fd4f3da96d85011092101a032a93a388b79e99e69c2d5c870 0.1s
=> => extracting sha256:d71d0e834b5c293d162992e6fb8991ch2309ae049a0eabc4efea161b2b5a3d0e
=> => extracting sha256:1e5a4c89cce5c0826c540ab664b6491c96eda01837f430bd47f7fd26782d6e3 0.0s
=> => extracting sha256:25ff2da83641968f65c3a74d880409d6b1b62ccfaab220b9ea70b80df5a2e0549 0.0s
=> [internal] load build context
=> => transferring context: 52.5kB
=> [2/5] WORKDIR /app
=> [3/5] COPY package*.json .
=> [4/5] RUN npm install
=> [5/5] COPY .
=> exporting to image
=> => exporting layers
=> => writing image sha256:536c93cb62ab60a69f515815d664ed4e229b83238828d8dd387277e9c321b4f2
=> => naming to docker.io/library/node-express-app
View build details: docker-decktop://dashboard/build/desktop-linux/desktop-linux/12694zkgwzktumzr55nyq4yeq
What's next:
  View a summary of image vulnerabilities and recommendations → docker scout quickview
PS E:\pavan\learn\Node JS\NodeJSBasicToAdvance\node-with-docker-sample1> |
```

Step 2: Runs container from an image

Command: docker run -p 3000:3000 --env-file .env node-express-app

Part	Meaning
docker run	Runs a new container from an image
-p 3000:3000	Maps port 3000 on your host (left side) to port 3000 in the container (right side)
--env-file .env	Loads environment variables from your local .env file into the container
node-express-app	The image to run — the one you just built with docker build

```
PS E:\pavan\learn\Node JS\NodeJSBasicToAdvance\node-with-docker-sample1> docker run -p 3000:3000 --env-file .env node-express-app
> node-js-and-docker@1.0.0 dev
> nodemon src/server.js

[nodemon] 3.1.10
[nodemon] to restart at any time, enter `rs`
[nodemon] watching path(s): *.*
[nodemon] watching extensions: js,mjs,cjs,json
[nodemon] starting `node src/server.js'
[dotenv@17.2.0] injecting env (0) from .env (tip: ⚙ override existing env vars with { override: true })
Server listening on port { port: '3000' }
```

Result: <http://localhost:3000>

