

In VS Code, your folder should look like this:

```
folder_name/
|— server.js
|— package.json
|— Dockerfile
```

Open **VS Code** → **Open Folder** → **folder\_name**

## **Step 1: Create package.json (if not already created)**

In the VS Code terminal:

```
npm init -y
```

It will open `package.json` by default in the terminal

Now install Express:

```
npm install express
```

This creates:

- `package.json`
- `package-lock.json`
- `node_modules/`

check `package.json` again to confirm installation

```
cat package.json
```

## Step 2: Your server.js

General format

```
//define interaction mode
'use strict'; //please be careful and strict.

// 2. Load a library
const someLibrary = require('some-library');

// 3. Create the server
const app = someLibrary();

// 4. Define server behavior
app.whenSomethingHappens((request, response) => {
  response.sendSomething();
});

// 5. Start the server
app.listen(PORT_NUMBER, () => {
  console.log('Server started');
});
```

Our server.js

```
'use strict'; //the rule
const express = require('express'); //the service
// Constants
const PORT = process.env.PORT || 3002;
const HOST = '0.0.0.0';

//the App server
const app = express();
app.get('/welcome', (req, res) => {
  res.send('Welcome to CMPT 353 Tutorials');
});
//Server behaviors
app.get('/', (req, resp) => { console.log(req.originalUrl);
  resp.send('hello world'); });
app.get('/hello', (req, resp) => {
  console.log(req.originalUrl);
  resp.send('hello'); });
//Let's start
app.listen(PORT, HOST, () => {
  console.log("Server listening on port:", PORT);
});
```

## Step 3: Create the Dockerfile

In VS Code:

- Right-click → **New File**
- Name it exactly: **Dockerfile**  
(⚠️ no .txt, no extension)

Learn 4 core building blocks.

**Block 1:** Base Image (FROM)

Every Dockerfile starts with:

FROM something

This answers:

“What already exists?”

Examples:

```
FROM python:3.10
FROM node:18
FROM ubuntu:22.04
```

Think like

“I am standing on someone else’s shoulders.”

**Block 2:** Copy Code (COPY)

COPY source destination

Examples:

```
COPY . /app
COPY hello.py /app
```

Think like

“Put my files inside the container.”

**Block 3:** Install Things (RUN)

RUN some-command

Examples:

```
RUN pip install Flask
```

```
RUN npm install
```

```
RUN apt update && apt install -y curl
```

Think like

“Set up the environment.”

**Block 4:** Start Command (CMD or command:)

Two ways to run

Way 1: Start automatically. When the docker run, it will start the app

```
CMD ["python", "hello.py"]
```

Way 2: For interactive mode. Use the terminal to start the app

```
CMD ["/bin/bash"]
```

```
FROM node:latest
# Create app directory
WORKDIR /app
# Install app dependencies
# Copy app files
COPY *.json ./
COPY server.js .
RUN npm install
#Expose port
EXPOSE 3002
#Start the app
#CMD [ "node", "server.js" ]
#Use terminal to start app
CMD ["/bin/bash"]
```

## **Step 4: Create docker-compose.yml**

In VS Code:

- Right-click → **New File**
- Name it exactly: **docker-compose.yml**

```
version: "3.9"
services:
  web:
    build: .
    container_name: node1
    ports:
      - "3002:3002"
    volumes:
      - ./:/app
    stdin_open: true
    tty: true
```

<b>Compose line</b>	<b>Manual Docker equivalent</b>
build: .	docker build -t node1 .
container_name: node1	docker run --name node1 node1
ports: "8080:8080"	docker run -p 8080:8080
volumes: <code>./:/app</code>  NOTE (SOURCE: DESTINATION)	docker run -v \$(pwd) :/app
stdin_open: true	docker run -i
tty: true	docker run -t

## **Step 5: Run (Compose way)**

From VS Code terminal:

- Make sure the container is not used somewhere else

To delete all the containers (when you're not certain about the name)

```
docker rm -f $(docker ps -aq)
```

To delete a specific container say node\_2

```
docker rm -f node_2
```

- Make sure which mode you have selected on dockerfile
  - If you have selected auto mode: CMD [ "node", "server.js" ]

Then run

```
docker compose up
```

You should see the following (the message you have set on the server.js, i.e., console.log("Server listening on port:", PORT);) on your VS code terminal:

```
Server listening on port: 3002
```

- If you have used terminal mode: CMD ["/bin/bash"]  
Which means you will run by yourself.

So type the following command

```
docker compose up
```

You should see:

```
✓ Container node_2 Running  
0.0s
```

```
Attaching to node_2
```

Now open a new terminal and type the following

```
docker exec -it node_2 bash  
node server.js
```

You should see the following (the message you have set on the server.js, i.e., console.log("Server listening on port:", PORT);) on your VS code terminal:

Server listening on port: 3002

## **Step 6: Test in Browser**

1. http://localhost:3002  
you should see hello world on the browser window  
and URL on the console window
2. http://localhost:3002/welcome  
you should see Welcome to CMPT 353 Tutorials on browser
3. http://localhost:3002/hello  
you should see hello Tutorials on browser

## **Step 6: Close the Compose File**

From VS Code terminal:

```
docker compose down
```