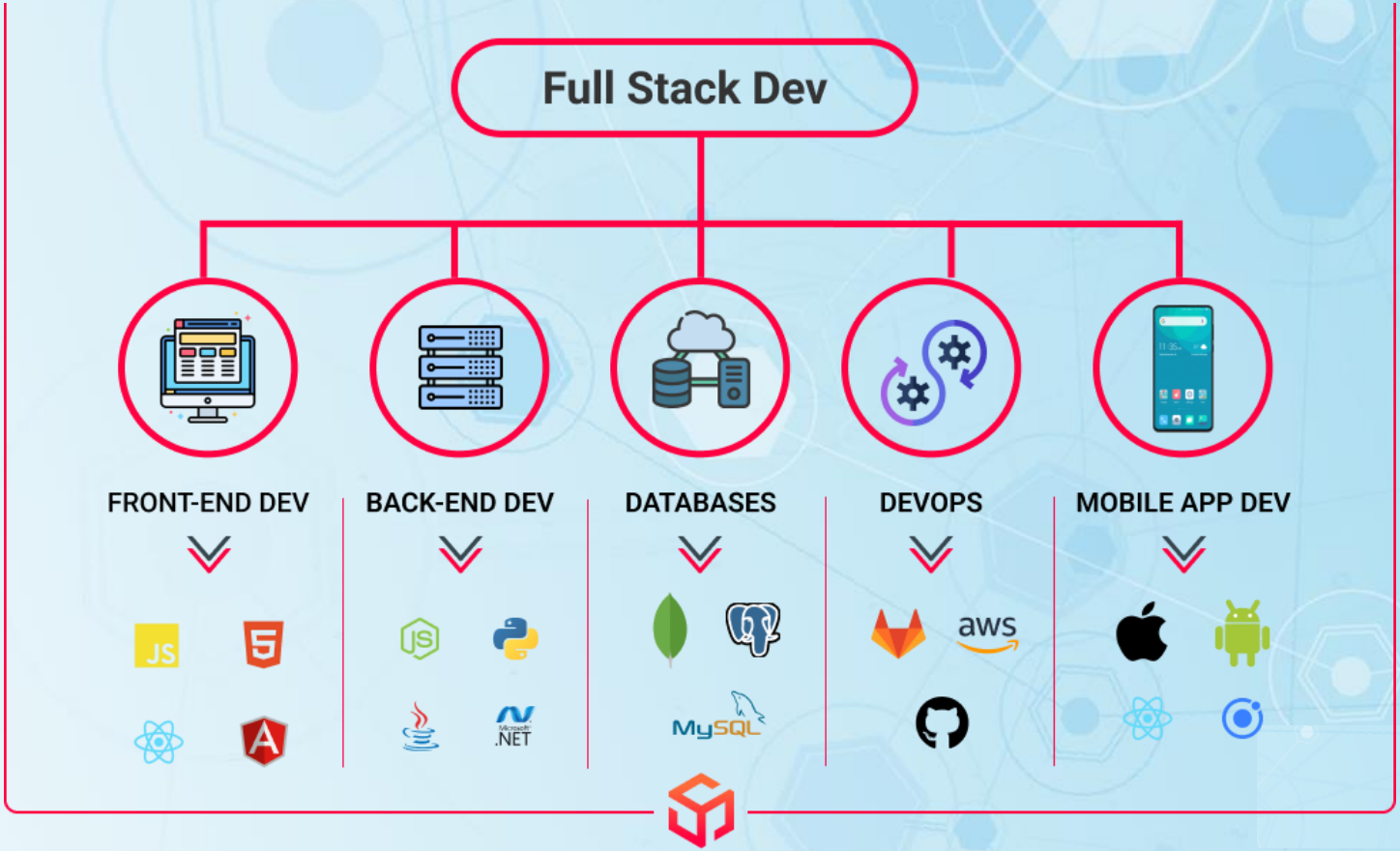
MC4266 – Full Stack Web Development



Ex:11

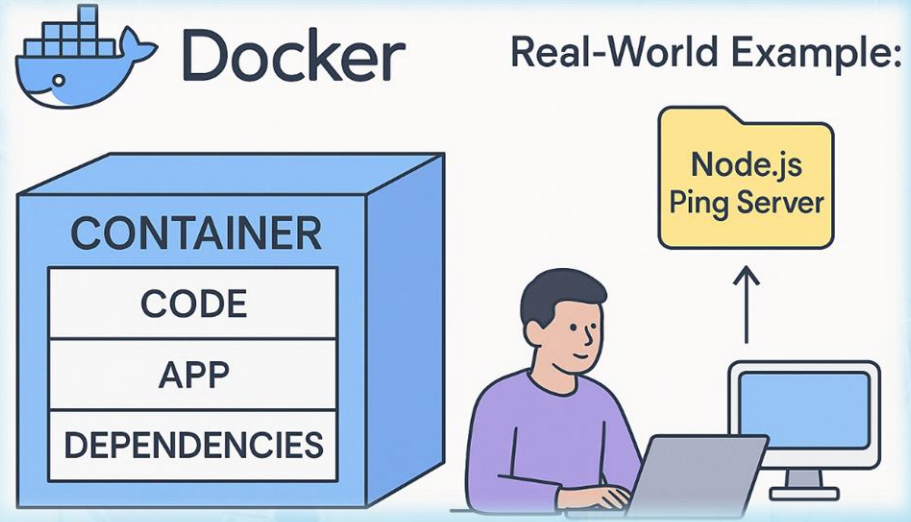
Create a docker container that will deploy a NodeJS ping server using the NodeJS image.

Ex:11

Docker

Docker is a **platform** for developing, shipping, and running applications in **containers**. A **container** packages your code with everything it needs to run: system tools, libraries, settings—everything—so it behaves the same **regardless of where it's run** (your laptop, a server, the cloud, etc.).

Think of Docker as a **lightweight virtual machine**, but faster and more efficient. Unlike a VM, Docker shares the host's OS kernel but keeps the app isolated.



Ex:11

Real-World Example: Node.js Ping Server

💻 **Developer Scenario**

- ❑ You're a developer building a small backend API (like the Node.js ping server you asked about). You want to:
- ❑ Share your project with teammates
- ❑ Run it on different environments (laptop, CI/CD, production)

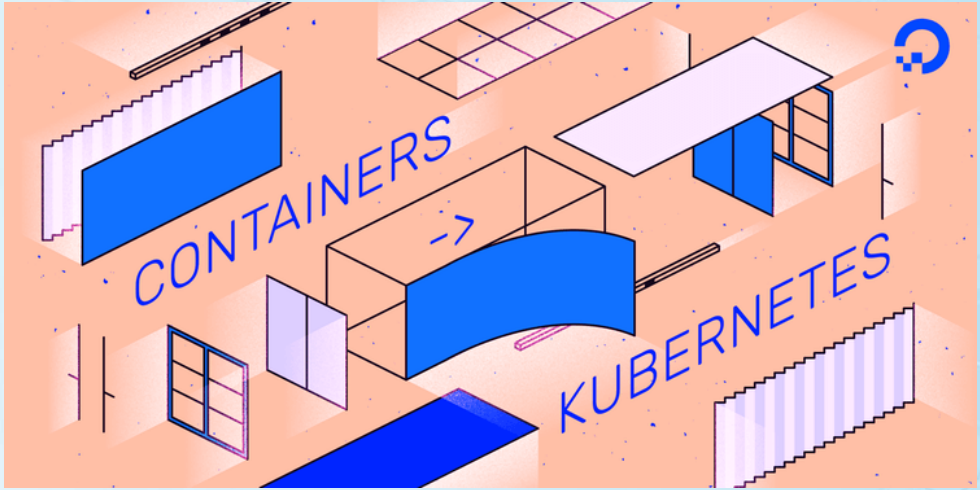
❌ **Without Docker**

- ❑ Every developer needs to manually install Node.js, set up the environment, install dependencies.
- ❑ It might work on your machine but break on someone else's due to version mismatch.

Ex:11

1. Download Docker and Install

https://docs.docker.com/desktop/setup/install/windows-install/



Docker, the starting point is typically creating an image for your application, which you can then run in a container. The image includes your application code, libraries, configuration files, environment variables, and runtime.

Ex:11

2. Create a NodeJS application

3. Create a ex_11_server.js

```
const http = require('http');
const port = 3000;

const server = http.createServer((req, res) => {
  res.writeHead(200, { 'Content-Type': 'text/plain' });
  res.end('ping\n');
});

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

Ex:11

4. Create a Dockerfile:

Create a file named Dockerfile without file extension in the same directory as server.js with the following content:

```
# Use the official Node.js image
FROM node:18
# Create app directory
WORKDIR /app
# Copy package.json and install dependencies
COPY package*.json ./
RUN npm install
# Copy the rest of the app code
COPY . .
# Expose the port
EXPOSE 3000
# Start the server
CMD ["node", "server.js"]
```

Ex:11

5. Build the Docker image in the terminal:

```
D:\FSWD_Lab\Ex_11>docker build -t ping-server .
```

6. Run the Docker container:

```
D:\FSWD_Lab\Ex_11>docker  docker run -d -p 3000:3000 ping-server
```

Ex:11

Output:

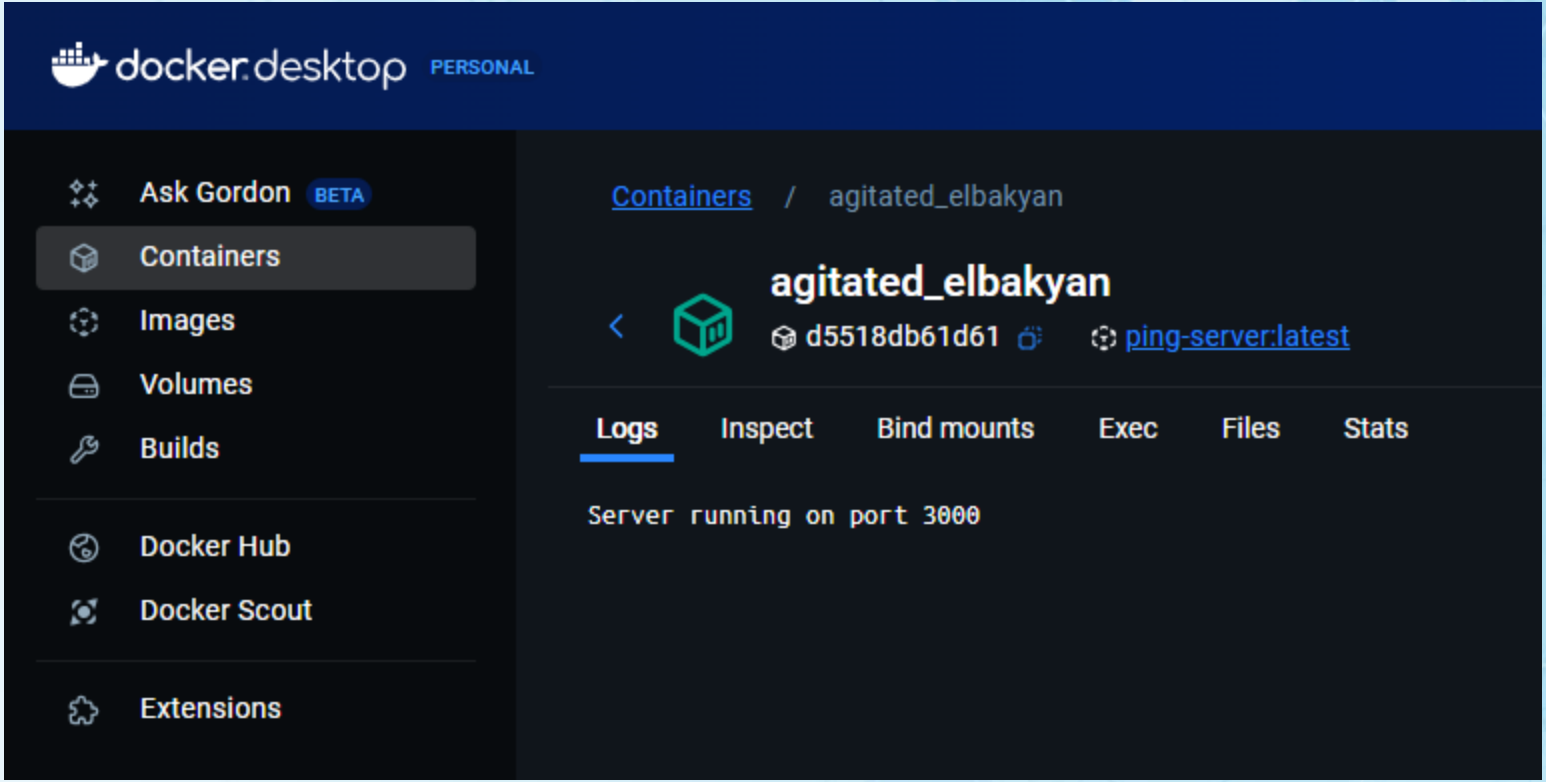
```
D:\FSWD_Lab\Ex_11>    docker build -t ping-server .
[+] Building 152.0s (11/11) FINISHED
 => [internal] load build definition from Dockerfile
 => => transferring dockerfile: 325B
 => [internal] load metadata for docker.io/library/node:18
 => [auth] library/node:pull token for registry-1.docker.io
 => [internal] load .dockerignore
 => => transferring context: 2B
 => [1/5] FROM docker.io/library/node:18@sha256:c649e5805e1801dbd653d4dd1dd4ca48d069531618a8cd400439cab2deee23b0
 => => resolve docker.io/library/node:18@sha256:c649e5805e1801dbd653d4dd1dd4ca48d069531618a8cd400439cab2deee23b0
 => => sha256:c649e5805e1801dbd653d4dd1dd4ca48d069531618a8cd400439cab2deee23b0 6.41kB / 6.41kB
 => => transferring context: 424B
 => [2/5] WORKDIR /app
 => [3/5] COPY package*.json ./
 => [4/5] RUN npm install
 => [5/5] COPY . .
 => exporting to image
 => => exporting layers
 => => writing image sha256:6178a6a41a23f4061e9ae6abc357401ae3b117b4825bece72b1afc5757816190
 => => naming to docker.io/library/ping-server

View build details: docker-desktop://dashboard/build/desktop-linux/desktop-linux/zhatkgx62cuwuei1emyh924uq

D:\FSWD_Lab\Ex_11>
```

Click the link to see the output executed on the docker desktop

Ex:11 Output:


docker desktop PERSONAL
Ask Gordon BETA
Containers
Images
Volumes
Builds
Docker Hub
Docker Scout
Extensions
Containers / agitated_elbakyan
agitated_elbakyan
d5518db61d61
ping-server:latest
Logs
Inspect
Bind mounts
Exec
Files
Stats
Server running on port 3000