1-1 Get Started With Docker

What is Docker in one line?
Docker is a tool that lets you package your application and everything it needs to run (like Python, Node, Nginx, etc.) into a **container**, which can run anywhere.

🧠 **Why does it matter?**
Because without Docker, you might install something on your machine that works fine — but then breaks when your teammate or server tries to run it. Docker stops this from happening by making everything run inside its own tiny world (container).

💡 **Example:**
Imagine you have a Node.js app. You can "dockerize" it and then run it on:

- Your Windows laptop
- A Linux server
- A teammate's Mac
  ...and it works the same way everywhere.

1-2 What is Docker & Why?

🛠️ **What is Docker?**
Docker is a platform for building, shipping, and running containers — isolated environments where your apps live.

🤔 **Why should I care?**
Before Docker:

- You had to manually install software dependencies.
- "It works on my machine" was a real nightmare.
- Setup for a project could take hours.

With Docker:

- You write a **Dockerfile** once and share it — no more "install Node, Mongo, etc."
- Your app is packaged with its environment — no surprises.

- **Faster, repeatable, and cleaner** development.

💡 **Example:**
Let's say you have a Python app that needs `Flask==2.0`.
Without Docker:

- You'd install Python
- Use pip to install Flask
- Maybe create a virtual environment (still risk of version conflicts)

With Docker:

- You write a Dockerfile that says: "Use Python 3.10 and install Flask 2.0"
- That's it. Share the file, and everyone has the same setup.

1-3 Virtual Machine vs. Docker Containers

🔍 **What's a Virtual Machine (VM)?**

- A full **emulation of a computer**, including its own OS.
- Runs on a **hypervisor** like VirtualBox or VMware.
- Needs a lot of **RAM and disk space**.

🐳 **What's a Docker Container?**

- A lightweight, standalone **package** of software.
- Shares the host OS kernel but is still isolated.
- Uses far fewer resources.

⚠️ **Why not just use VMs? What's the problem?**

- Slow boot times (VMs take minutes to start).
- Heavy: Each VM can take gigabytes of space.
- Overhead: Each VM runs a full OS (even if you don't need it).
- Hard to scale: Running 100 VMs? Good luck.

✅ **How Docker solves it:**

- Containers start in **seconds**.
- Much smaller (often under 100MB).
- No full OS inside, just the parts your app needs.
- You can run **dozens of containers** on the same machine where only a few VMs could fit.

📊 **Comparison Table:**

Feature	Virtual Machine	Docker Container
Startup Time	Minutes	Seconds
Size	Gigabytes (includes full OS)	Megabytes (only needed files)
Isolation	Full (separate OS per VM)	Partial (shares host OS kernel)
Performance	Slower (more overhead)	Faster (lightweight)
Use Case	Full OS apps, legacy systems	Modern apps, microservices, fast scaling

💡 **Example:**
Let's say you want to test 5 versions of a Node.js app:

- With VMs: You need 5 OS installations — painful.
- With Docker: You can spin up 5 containers with different versions in seconds.

1-8 Overview of Docker Tools

🧰 Here's a toolbox Docker gives you:

- **Docker Engine**: The core runtime that runs containers.
- **Docker CLI**: The command-line tool (`docker run`, `docker build`, etc.).
- **Docker Hub**: Like GitHub for Docker images — search for official images like `node`, `postgres`, etc.
- **Docker Compose**: Lets you define and run **multi-container** apps (like app + DB + Redis) using a `docker-compose.yml` file.
- **Dockerfile**: A script that defines how to build your app's image (like a recipe).
- **Docker Desktop**: GUI + backend for Docker (mainly for Windows/Mac users).

💡 **Example:**
You're building a blog:

- Your blog app (Node.js) goes in one container.
- Your database (PostgreSQL) goes in another.
- You use Docker Compose to start both with a single command.

1-9 Configuring VS Code & Creating The First Container

👨‍💻 **Step-by-step to run your first container in VS Code:**

1. **Install Docker Desktop**
    - For Windows/Mac — this gives you the Docker engine + GUI.
2. **Install Docker extension in VS Code**
    - Helps you build/run containers easily inside the editor.

Create a Dockerfile in your project:

dockerfile
CopyEdit
```
FROM node:18
WORKDIR /app
COPY . .
RUN npm install
CMD ["node", "index.js"]
```

3. **Build the image:**

bash
CopyEdit
```
docker build -t my-app .
```

4. **Run the container:**

bash
CopyEdit
```
docker run -p 3000:3000 my-app
```

5. Now open `localhost:3000` in your browser — your app is running inside a container.