Docker

Docker is a **container platform** used to run applications in isolated environments called **containers**.

A container includes:

- Application code
- Required libraries
- Runtime
- Configuration

So the application runs the same on any system.

Why Docker is Used

- **Portability** – Runs anywhere (local, server, cloud)
- **Lightweight** – Faster and uses less memory than Virtual Machines
- **Consistency** – Same environment in dev, test, and production

Important Terms

- **Image** – Blueprint of the application
- **Container** – Running instance of an image
- **Dockerfile** – File used to create a Docker image

Eg:docker_sample

app.js

```
projects > Docker-2 > app.js > ...
1  const express = require('express');
2  const app = express();
3
4  app.get('/', (req, res) => {
5    res.send('Hello from Docker!');
6  });
7
8  app.listen(3000, '0.0.0.0', () => {
9    console.log('App running on port 3000');
10 });
11
```

Dockerfile

```
projects > Docker-2 > Dockerfile
1  FROM node:18-alpine
2  WORKDIR /app
3  COPY package*.json ./
4  RUN npm install
5  COPY . .
6  EXPOSE 3000
7  CMD ["node", "app.js"]
8
```

package.json

```
projects > docker_sample > {} package.json > ...
1  {
2    "name": "docker-basic-app",
3    "version": "1.0.0",
4    "main": "app.js",
     ▷Debug
5    "scripts": {
6      "start": "node app.js"
7    },
8    "dependencies": {
9      "express": "^4.18.2"
10   }
11 }
12
```

CODE AND EXPLANATION:

Application Code (app.js)

This is a simple Express server that runs on port 3000.

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

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

app.listen(3000, '0.0.0.0', () => {
  console.log('App running on port 3000');
});
```

Explanation:

- `express` → Web framework for Node.js
- `app.get('/')` → Defines route for homepage
- `res.send()` → Sends response to browser
- `app.listen(3000)` → Runs server on port 3000
- `'0.0.0.0'` → Makes app accessible outside container

package.json

This file contains project details and dependencies.

```
{
  "name": "docker-basic-app",
  "version": "1.0.0",
  "main": "app.js",
  "scripts": {
    "start": "node app.js"
  },
  "dependencies": {
    "express": "^4.18.2"
  }
}
```

Explanation:

- `"name"` → Project name
- `"version"` → App version
- `"main"` → Entry file
- `"scripts"` → Start command
- `"dependencies"` → Required packages (Express)

Dockerfile

This file is used to create the Docker image.

```
FROM node:18-alpine
```

```
WORKDIR /app

COPY package*.json ./

RUN npm install

COPY . .

EXPOSE 3000

CMD ["node", "app.js"]
```

Explanation:

- `FROM node:18-alpine` → Uses lightweight Node.js base image
- `WORKDIR /app` → Sets working directory inside container
- `COPY package*.json ./` → Copies dependency files
- `RUN npm install` → Installs dependencies
- `COPY . .` → Copies application code
- `EXPOSE 3000` → Opens port 3000
- `CMD` → Starts the application

Build and Run Commands

Build Docker Image:

```
docker build -t docker-basic-app .
```

Run Container:

```
docker run -p 3000:3000 docker-basic-app
```

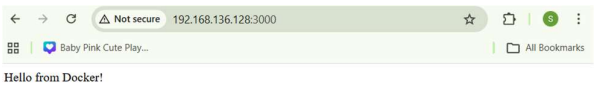
Now open browser:

```
http://localhost:3000
eg:http:// 192.168.136.128:3000
```

You will see:

```
Hello from Docker!
```

Output:



History:

srimathi@ubs2204vm:~/projects/docker_sample$ history

```
    1  whoami
    2  sudo apt update
    3  sudo apt upgrade -y
    4  sudo apt update
    5  sudo apt upgrade -y
    6  sudo apt install -y ca-certificates curl gnupg lsb-release
    7  sudo mkdir -p /etc/apt/keyrings
    8  curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o /etc/apt/keyrings/docker.gpg
    9  sudo mkdir -p /etc/apt/keyrings
   10  curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o /etc/apt/keyrings/docker.gpg
   11  echo "deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.gpg] \
   12  https://download.docker.com/linux/ubuntu \
   13  $(lsb_release -cs) stable" | sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
   14  sudo apt update
   15  sudo apt install -y docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compose-plugin
   16  sudo systemctl status docker
   17  docker --version
   18  mkdir projects
   19  cd projects
   20  pwd
   21  git clone https://github.com/jagadeeshkanna97/docker_sample
   22  ls -a
   23  ls
   24  cd docker_sample
   25  ls
   26  docker build -t docker-sample .
   27  sudo docker build -t docker-sample .
```

```
  28  sudo docker build --network=host -t docker-sample .
  29  sudo docker run -d -p 3000:3000 --name docker_sample docker-sample
  30  sudo docker ps
  31  sudo docker build --no-cache -t docker-sample .
  32  rm -rf ~/.vscode-server
  33  exit
  34  more /etc/os-release
  35  ifconfig
  36  sudo apt install net-tools
  37  ip addr
  38  ifconfig
  39  sudo apt update && sudo apt upgrade
  40  ip a
  41  ping -c 3 google.com
  42  ip a
  43  sudo systemctl status ssh
  44  ping -c 33 google.com
  45  sudo nano /etc/resolv.conf
  46  ip a
  47  nano reboot
  48  sudo reboot
  49  mkdir -p ~/.vscode-server/bin/591199df409fbf59b4b52d5ad4ee0470152a9b31
  50  cd ~/.vscode-server/bin/591199df409fbf59b4b52d5ad4ee0470152a9b31
  51  wget https://update.code.visualstudio.com/commit:COMMIT_ID/server-linux-x64/stable -O
vscode-server.tar.gz
  52  ip a
  53  sudo apt update
  54  sudo apt install openssh-server -y
  55  sudo systemctl status ssh
  56  sudo ufw allow ssh
  57  sudo ufw reload
```

```
58  sudo ufw status
59  sudo ufw allow ssh
60  sudo ufw enable
61  sudo ufw status
62  ip a
63  ssh srimathi@192.168.136.128
64  ip a
65  cd projects
66  cd docker_sample
67  ls
68  docker --version
69  sudo docker build -t docker-sample .
70  sudo docker images
71  sudo docker run -d -p 3000:3000 --name docker_sample_app docker-sample
72  sudo docker ps
73  sudo apt update
74  sudo apt install gh -y
75  sudo apt update
76  sudo apt install gh -y
77  ssh-keygen -t ed25519 -C "srimathit.23cse@kongu.edu"
78  eval "$(ssh-agent -s)"
79  ssh-add ~/.ssh/id_ed25519
80  cat ~/.ssh/id_ed25519.pub
81  ls
82  git init
83  git add .
84  git commit -m "Initial commit"
85  git config --global user.name "SRIMATHI-T"
86  git config --global user.email "srimathit.23cse@kongu.edu"
87  git config --global --list
88  git commit -m "Initial commit"
```

```
 89  git branch -M main
 90  git push -u origin main
 91  git config --global --unset credential.helper
 92  git push -u origin main
 93  git config --global --unset credential.helper
 94  git config --global --unset-all credential.helper
 95  git config --global credential.helper store
 96  git push -u origin main
 97  ssh -T git@github.com
 98  git push -u origin main
 99  git remote -v
100  git remote remove origin
101  git remote -v
102  git remote add origin git@github.com:SRIMATHI-T/devops.git
103  git remote -v
104  git branch -M main
105  git push -u origin main
```

Docker – Build, Push, Pull, Run

A. Build Docker Image

Inside your project folder (where Dockerfile exists):

```
docker build -t vacayhome-nginx .
```

Explanation:

- `build` → create image
- `-t` → tag name
- `.` → current folder

B. Run Docker Container

```
docker run -d -p 8080:80 vacayhome-nginx
```

Explanation:

- `-d` → run in background

- `-p 8080:80` → hostPort:containerPort

Open in browser:

```
http://localhost:8080
```

C. Login to Docker Hub

```
docker login
```

Enter username & password.

D. Tag Image for Docker Hub

```
docker tag vacayhome-nginx srimathi2t/vacayhome:latest
```

Format:

```
docker tag <local-image> <dockerhub-username>/<repo>:tag
```

E. Push to Docker Hub

```
docker push srimathi2t/vacayhome:latest
```

Now image is online.

F.Pull from Docker Hub (Friend System)

```
docker pull srimathi2t/vacayhome:latest
```

G. Run Pulled Image

```
docker run -d -p 8080:80 srimathi2t/vacayhome
```

Git – Push Project to GitHub

A. Initialize Git

```
git init
```

B. Add Files

```
git add .
```

C. Commit

```
git commit -m "Initial commit"
```

D. Connect to GitHub Repo

```
git remote add origin https://github.com/USERNAME/REPO.git
```

E. Push to GitHub

```
git branch -M main
git push -u origin main
```

F. Force Push (Replace Old Files)

```
git push -f origin main
```

Pull Code from GitHub

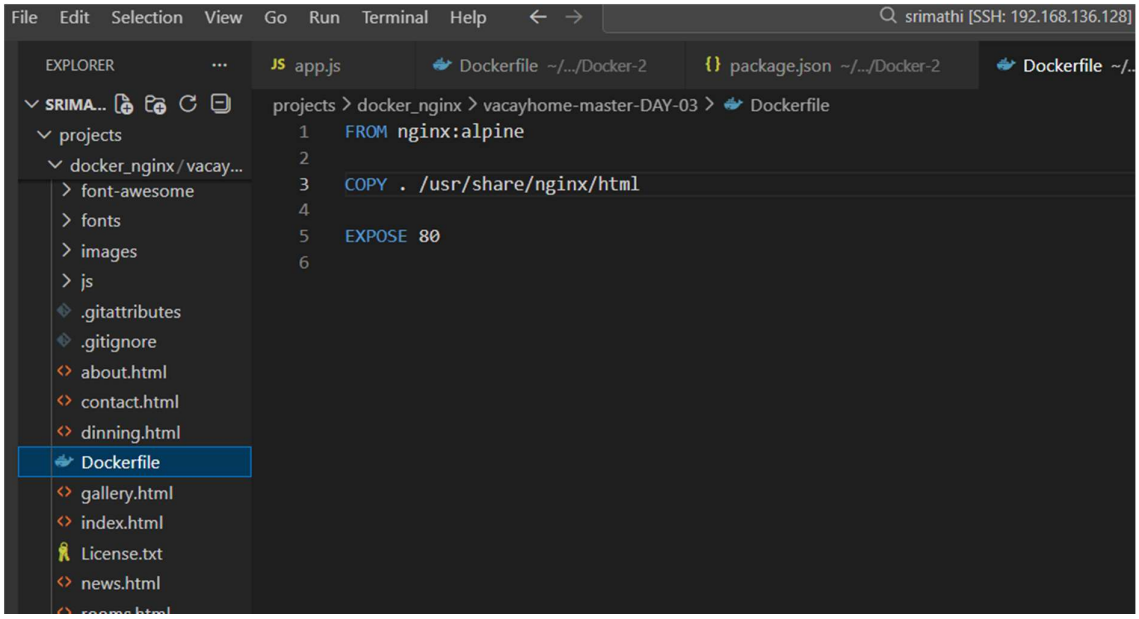
If your friend wants your project:

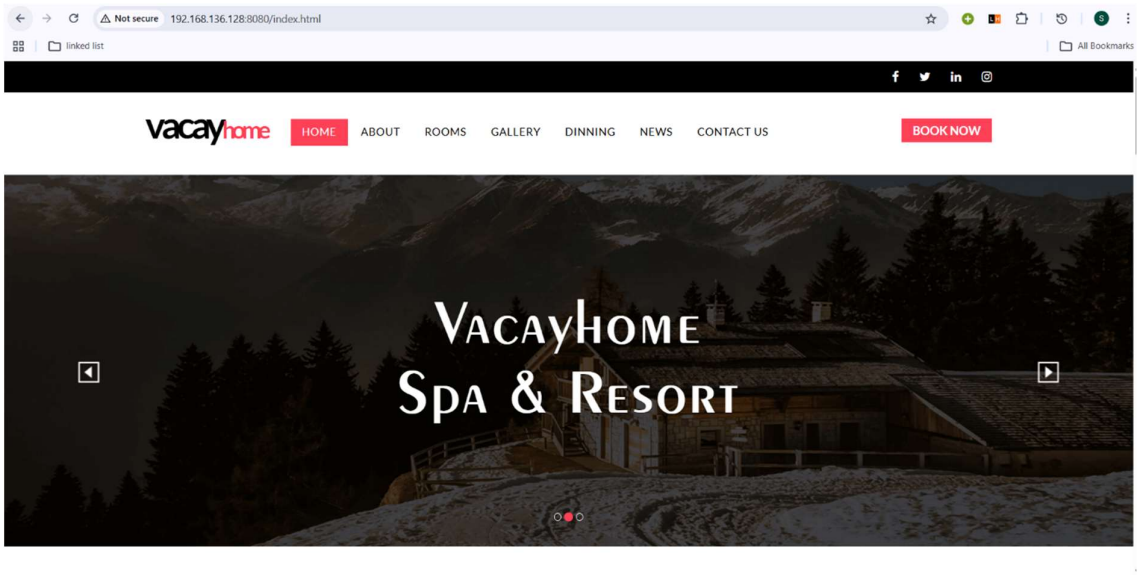
```
git clone https://github.com/USERNAME/REPO.git
```

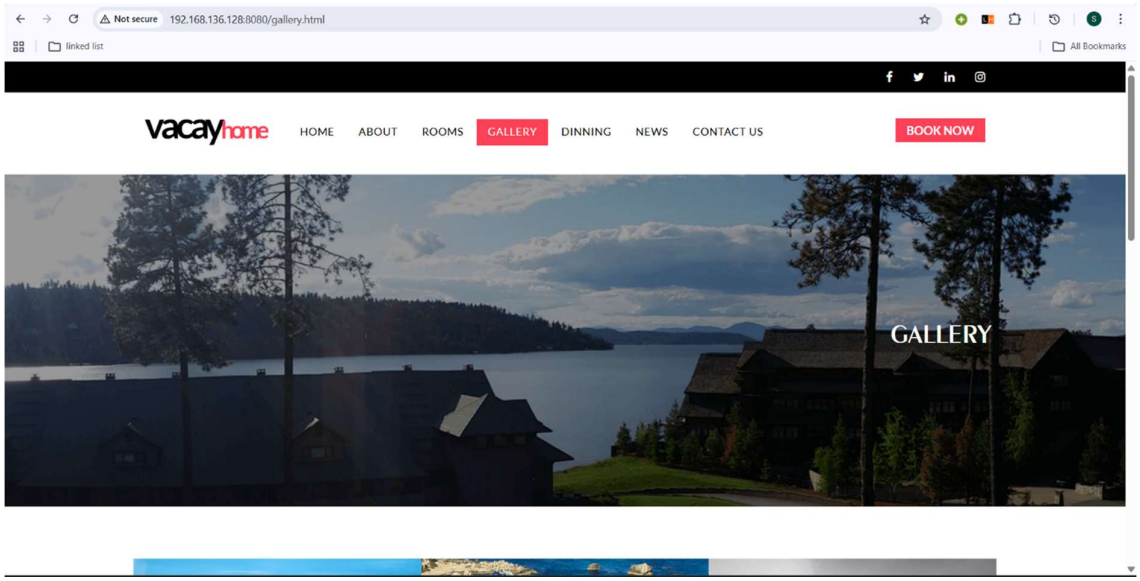
Then:

```
cd REPO
```

eg:nginx:







Ubuntu Docker-push and github-push command:

76 sudo docker login

75 vacayhome

76 docker tag vacayhome-nginx srimathi2t/vacayhome:latest

77 docker push srimathi2t/vacayhome:latest

78 cd ..

79 git init

```
 80  git add .
 81  git commit -m "New vacayhome project"
 82  git remote remove origin https://github.com/SRIMATHI-T/devops.git
 83  git remote add origin https://github.com/SRIMATHI-T/devops.git
 84  git remote -v
 85  git branch -M main
 86  git push -f origin main
 87  git init
 88  git add .
 89  git commit -m "New vacayhome project"
 90  git remote remove origin https://github.com/SRIMATHI-T/DEVOPS.git
 91  git remote add origin https://github.com/SRIMATHI-T/DEVOPS.git
 92  git remote -v
 93  git branch -M main
 94  git push -f origin main
```

Vmware Ubunutu commands:-

```
 33  exit
 34  more /etc/os-release
 35  ifconfig
 36  sudo apt install net-tools
 37  ip addr
 38  ifconfig
 39  sudo apt update && sudo apt upgrade
 40  ip a
 41  ping -c 3 google.com
 42  ip a
 43  sudo systemctl status ssh
 44  ping -c 33 google.com
 45  sudo nano /etc/resolv.conf
 46  ip a
 47  nano reboot
 48  sudo reboot
 49  mkdir -p ~/.vscode-server/bin/591199df409fbf59b4b52d5ad4ee0470152a9b31
 50  cd ~/.vscode-server/bin/591199df409fbf59b4b52d5ad4ee0470152a9b31
 51  wget https://update.code.visualstudio.com/commit:COMMIT_ID/server-linux-x64/stable -O vscode-
server.tar.gz
 52  ip a
 53  sudo apt update
 54  sudo apt install openssh-server -y
 55  sudo systemctl status ssh
 56  sudo ufw allow ssh
 57  sudo ufw reload
 58  sudo ufw status
 59  sudo ufw allow ssh
 60  sudo ufw enable
 61  sudo ufw status
 62  ip a
 63  ssh srimathi@192.168.136.128
 64  ip a
 65  code
 66  docker ps
 67  history
```

System and Network Checks

- `more /etc/os-release` → Check Ubuntu version info.
- `ifconfig` / `ip addr` / `ip a` → Show network interfaces and IP addresses.
- `ping -c 3 google.com` → Test internet connectivity.

Git – Basic Explanation

What is Git?+----------

Git is a **version control system** used to track changes in source code.
It helps developers manage project history and work together.

Main Branch

- The default branch of a repository.
- Usually called **main** (or master).
- Contains stable and production-ready code.

Sub Branch (Feature Branch)

- Created from the main branch.
- Used to develop new features or fix bugs.
- Later merged back into the main branch.

Example:

```
main → feature-login → merge → main
```

Git Commands (From Ubuntu to GitHub Repository)

1 Initialize Git

```
git init
```

Creates a new Git repository in your project folder.

2 Add Files (Insert into Git)

```
git add filename
```

OR

```
git add .
```

Adds files to staging area.

3 Commit Changes

```
git commit -m "Initial commit"
```

Saves changes with a message.

4 Connect to Remote Repository

```
git remote add origin https://github.com/username/repo.git
```

5 Push to GitHub

```
git push -u origin main
```

Uploads code to remote repository.

Append / Insert in Git

When you modify a file:

1. Edit the file
2. Run:
3. `git add .`
4. `git commit -m "Updated file"`
5. `git push`

This appends new changes to Git history.

Branch and History

- Git keeps a complete **history** of changes.
- Each commit has a unique ID (hash).
- You can view history using:
- `git log`

HEAD and Tail (Simple Meaning)

- **HEAD** → Points to the latest commit in the current branch.
- It shows where you are currently working.
- **Tail** → The first (oldest) commit in the history.

Example:

```
Tail ---- Commit1 ---- Commit2 ---- HEAD
```

Git Commands – From Ubuntu to GitHub

1 Create a Project Folder

```
mkdir myproject
cd myproject
```

2 Initialize Git Repository

```
git init
```

This creates a hidden `.git` folder and starts version control.

3 Create Files in Ubuntu

Create empty file

```
touch file.txt
```

Insert text into file (overwrite)

```
echo "Hello World" > file.txt
```

Append text into file

```
echo "Second line" >> file.txt
```

- `>` → Overwrites file
- `>>` → Appends (adds) content
-

4 Add File to Git (Insert into Git)

```
git add file.txt
```

OR add all files:

```
git add .
```

This moves files to the **staging area**.

5 Commit the Changes

```
git commit -m "Added file.txt"
```

This saves changes into Git history.

6 Connect Ubuntu to GitHub Repository

```
git remote add origin https://github.com/username/repository.git
```

7 Push Code to GitHub

```
git push -u origin main
```

Now the file is uploaded from Ubuntu to GitHub.