Commonly used: `rm -rf directory_name`
(`-rdf` is sometimes written but `-rf` is sufficient; the order of flags doesn't matter)

Wildcards in ls Command

- `*` → Matches **any number of characters**.
    - Example: `ls *.txt` → lists all `.txt` files.
- `?` → Matches **exactly one character**.
    - Example: `ls file?.txt` → matches `file1.txt`, `file2.txt`, but not `file10.txt`.
- `[ ]` → Matches **any one character inside brackets**.
    - Example: `ls file[12].txt` → matches `file1.txt` and `file2.txt`.
- `ls *.??` → Lists files with **any name** and **exactly two-character extension**.
    - Matches: `file.sh`, `data.py`
    - Does not match: `file.txt` or `a.c`

1. sudo apt update && sudo apt upgrade

- **Purpose:** Keep your Ubuntu system **up-to-date**.
- **`sudo apt update`** → Updates the list of available software packages and their versions.
- **`sudo apt upgrade`** → Installs the latest versions of installed packages.
- **`sudo`** → Runs the command as **superuser (administrator)**.

Example:

```
sudo apt update
sudo apt upgrade -y
```

- The `-y` automatically confirms all upgrades.

IPv4 – Address Allocation

- **IPv4** is a 32-bit address system for identifying devices on a network, written as four numbers (0–255) separated by dots, e.g., `192.168.1.10`.
- **How it is assigned:**
    1. **Static IP (Manual):** Admin assigns a fixed address to a device.
        - Example: Printer or server often uses static IP.
    2. **Dynamic IP (Automatic):** A DHCP server automatically gives an available IP to a device when it connects.
        - Example: Your laptop or phone usually gets a dynamic IP from your Wi-Fi router.
- **Purpose:** Ensures every device has a unique address to send and receive data.

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.

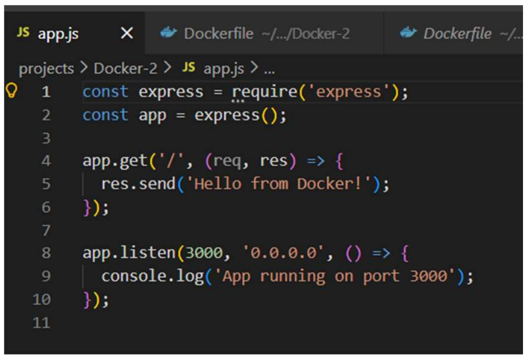
Why Docker is Used

- **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

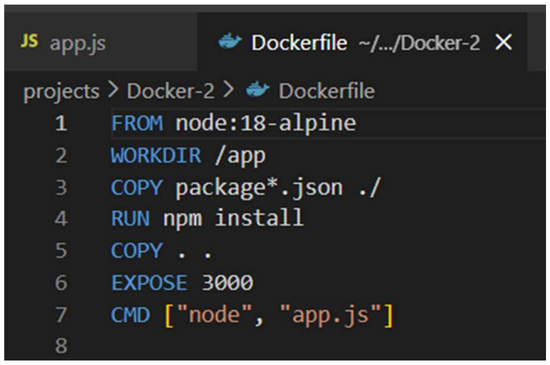
- **Image** – Blueprint of the application
- **Container** – Running instance of an image
- **Dockerfile** – File used to create a Docker image

Eg:docker_sample



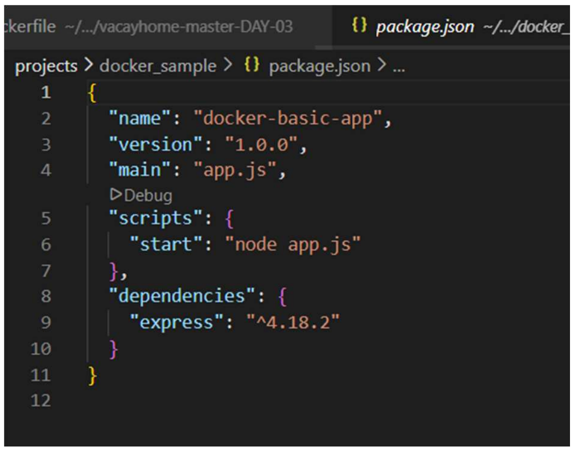
projects > Docker-2 > app.js > ...

```js
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');
});

```



projects > Docker-2 > Dockerfile

```dockerfile
FROM node:18-alpine
WORKDIR /app
COPY package*.json ./
RUN npm install
COPY . .
EXPOSE 3000
CMD ["node", "app.js"]

```



projects > docker_sample > package.json > ...

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

```

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

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.