DOCKER INTERNALS — QUICK REVISION NOTES

1. What Is a Docker Container?

- A container is just a Linux process.

- Isolated using namespaces.

- Limited using cgroups.

- Uses a filesystem made from image layers.

- Not a VM or OS.

2. Docker Image Structure

- Image = multiple read-only layers.

- Container adds one writable layer on top.

- OverlayFS merges layers.

- Deleting files creates whiteout markers.

3. Container Creation Flow

docker CLI → dockerd → runc → container process (PID 1)

Roles:

- docker CLI: Sends command.

- dockerd: Prepares filesystem, network, config.

- runc: Creates namespaces, cgroups, mounts, and starts PID 1.

4. Namespaces (Isolation)

- PID: Own process list.

- Mount: Own filesystem.

- NET: Own network stack.

- UTS: Own hostname.

- IPC: Own shared memory.

- USER: Own user mapping.

5. Cgroups (Resource Limits)

- Limit CPU, RAM, IO, PIDs, network bandwidth.

6. Networking Model

eth0 (container) <-> veth pair <-> vethXYZ (host) <-> docker0 bridge <-> host NIC

7. PID 1 Behavior

- Must reap zombie processes.

- Ignores some signals.

- If PID 1 exits → container stops.

8. VM vs Container

VM:

- Own OS and kernel.

- Heavy, slower.

Container:

- Shares host kernel.

- Lightweight, fast, starts instantly.

9. Why Containers Are Fast

- No OS boot.

- Copy-on-write layers.

- runc starts a process, not a machine.

- Namespaces + cgroups are lightweight.

- veth networking is virtual and fast.