Evolution of App Hosting.

From bare-metal to containers

Udhayaprakasha

Udhayaprakasha

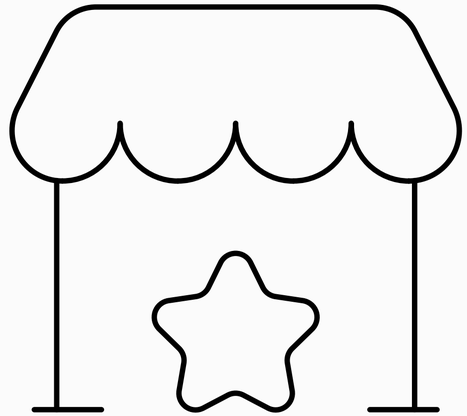
About me

- Full-Stack Engineer | 8+ Years.
- React, Next.js, TypeScript, GraphQL, Java, Vercel, GCP and Terraform.
- Building booking platforms, speaking at FOSS/VueVerse/React events, and working across frontend and infra systems.

🚀 Passionate about creating performant, secure & scalable web apps

What we’ll cover today.

1. Bare-metal hosting
2. Virtual machines
3. Introduction to containers
4. Docker & how it works
5. Docker image optimisation
6. Tools & best practices
7. Q&A



Some groundwork...

- Bare-metal
- Virtual machines
- Containers

Some groundwork...

- **Bare-metal**
- Virtual machines
- Containers

Some groundwork...

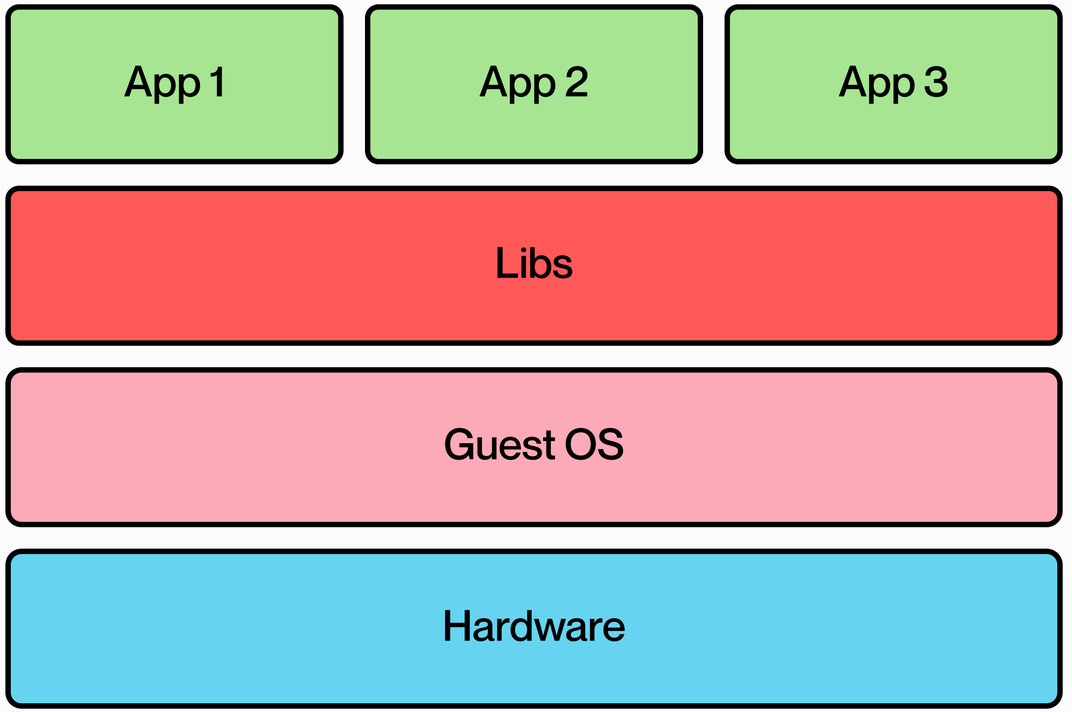
- Bare-metal
- **Virtual machines**
- Containers

Some groundwork...

- Bare-metal
- Virtual machines
- **Containers**

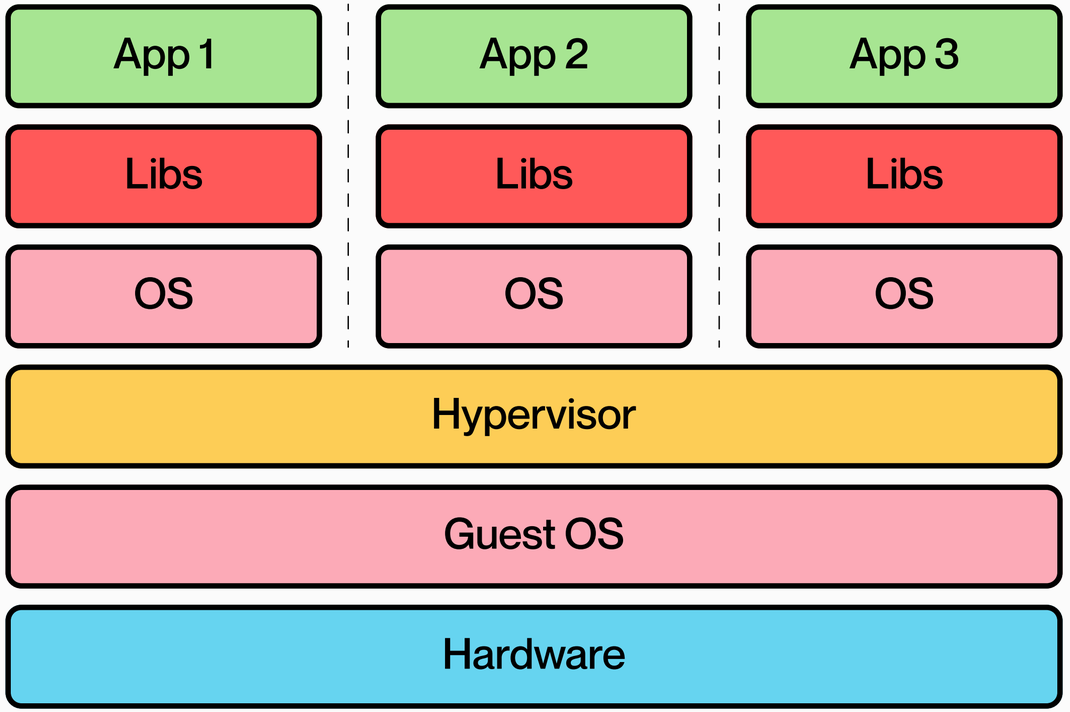
The OG setup.

- Apps deployed directly on hardware
- Shared OS = shared problems (hello, conflicts 👋)
- Manual configs, updates, and chaos
- Scale = buy, wait, plug, pray 😅



Isolation enters the chat.

- Hypervisors split one machine into many
- Each VM runs its own OS – total separation
- Better hardware utilization than bare-metal
- Type 1 = fast & direct | Type 2 = great for devs

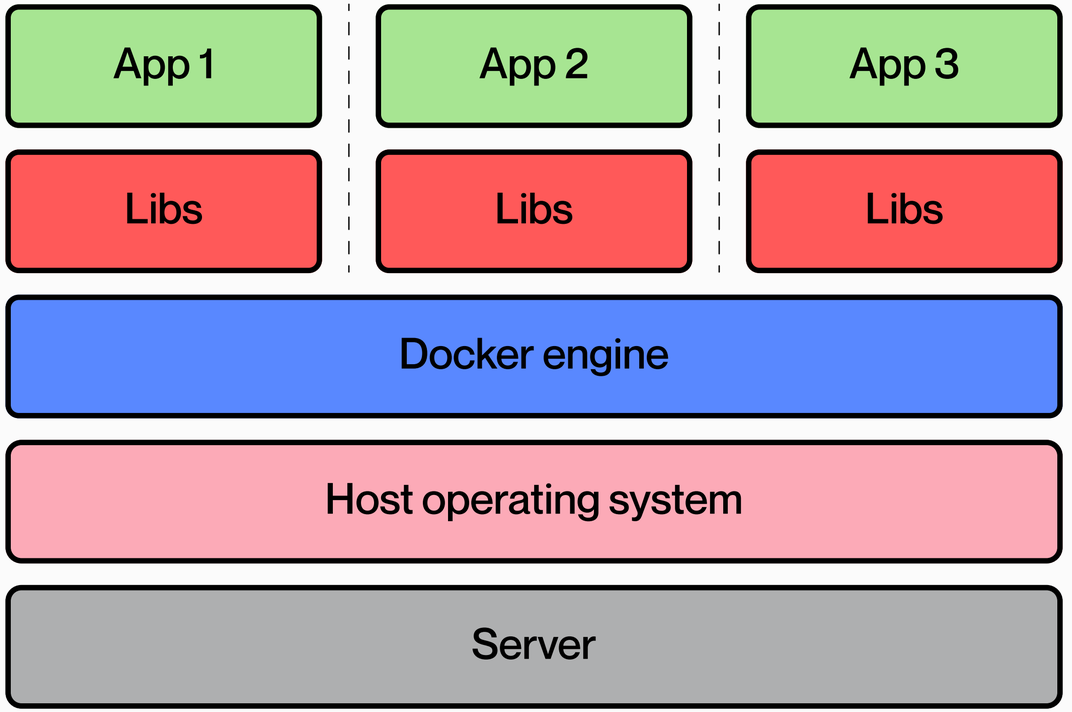


Power comes at a price.

- Every VM runs a full OS = bulky
- Boot time: grab a coffee ☕
- Wastes resources if not tuned
- Scaling still takes effort

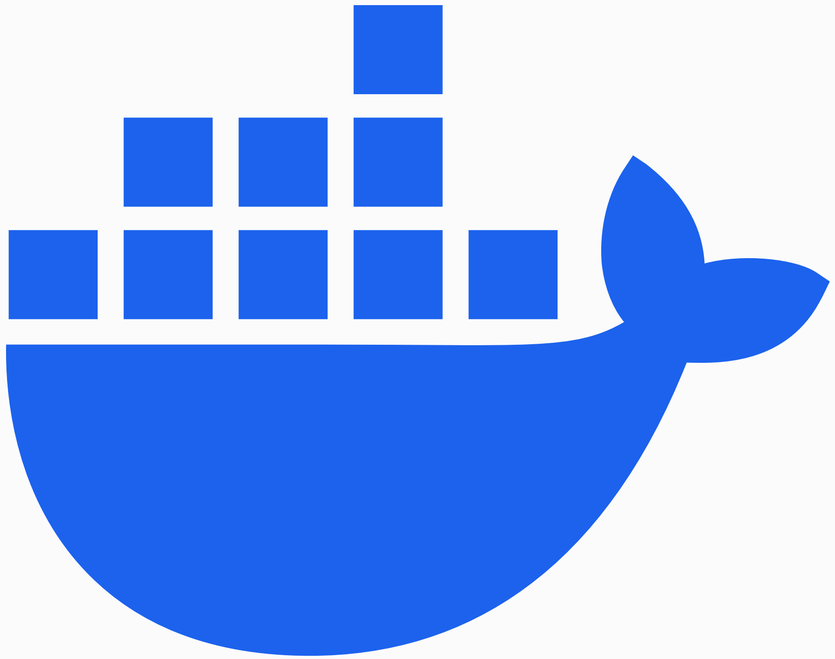
Lightweight, fast, portable.

- Share the host kernel, run in isolation
- Start in seconds, scale like magic 🪄
- Bring your own dependencies
  (no global mess)
- Works the same in dev, test, production



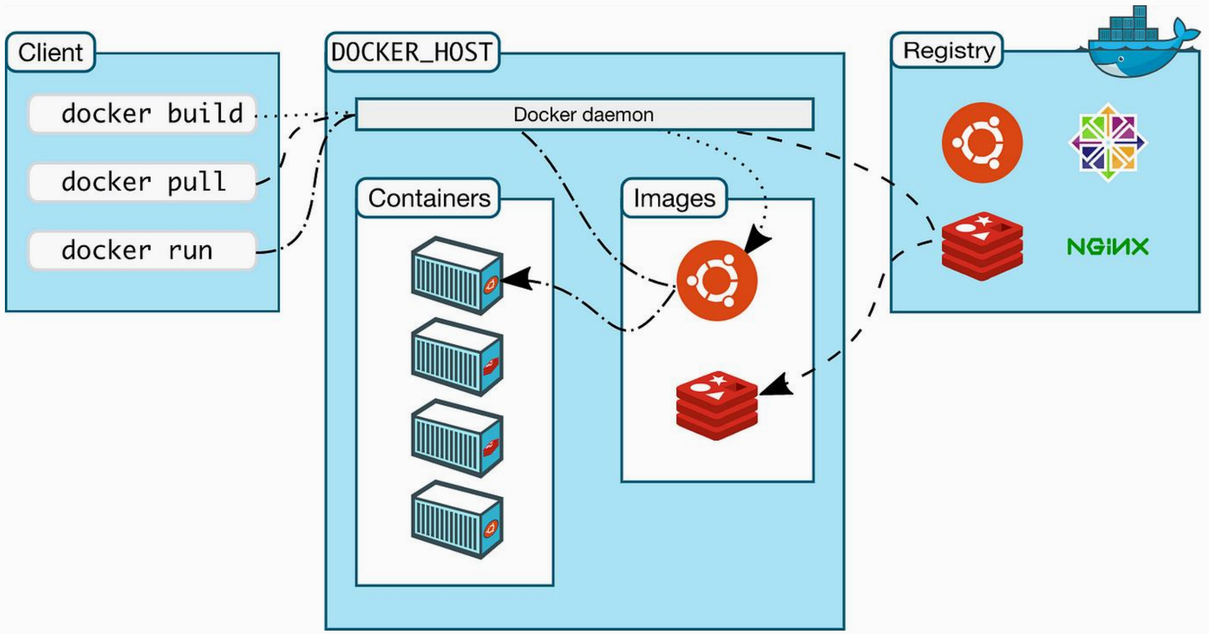
Container magic, simplified.

- Made containers easy & mainstream
- Define apps with dockerfile, run anywhere
- Works the same on laptop & cloud
- Ecosystem: Docker CLI, Compose, Hub, Swarm



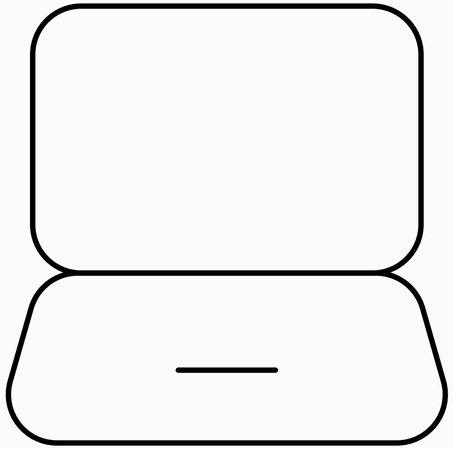
Build, ship, run.

- Docker Engine: builds & runs containers
- Images = versioned app blueprints 📦
- Containers = live, isolated instances
- Registries = app store for images



DEMO

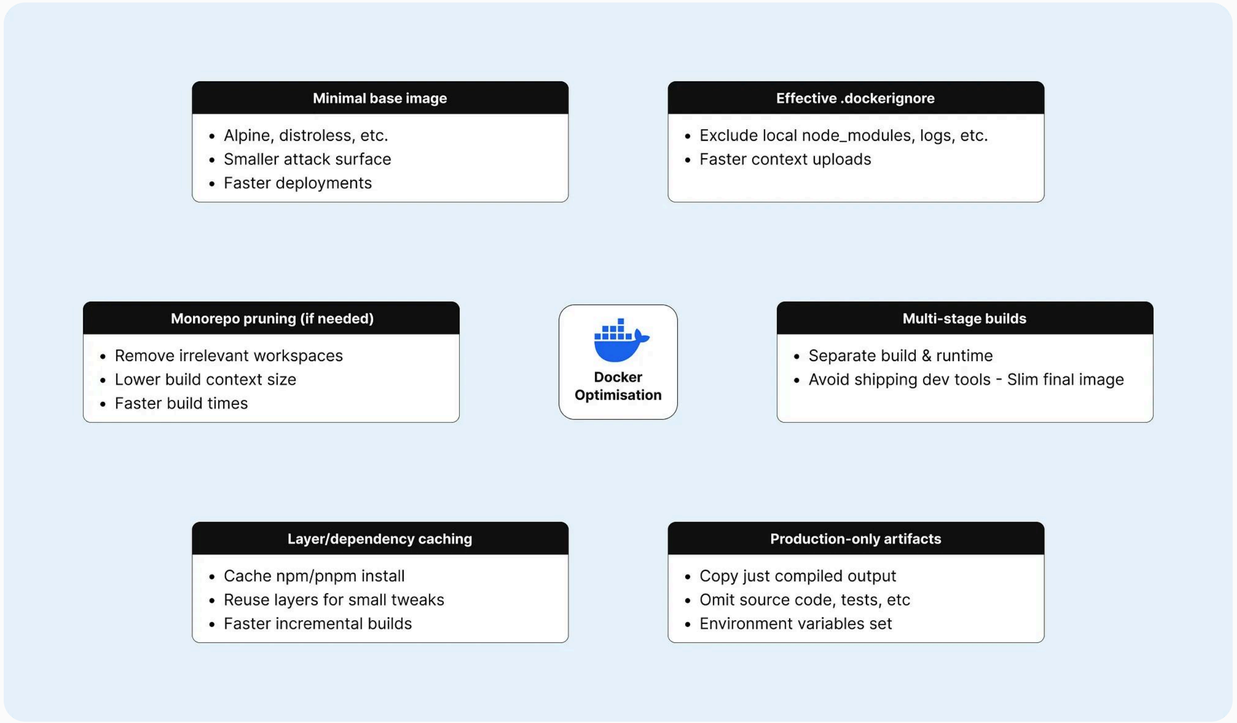
Smooth sailing with Docker basics.



From bloated to blazing.

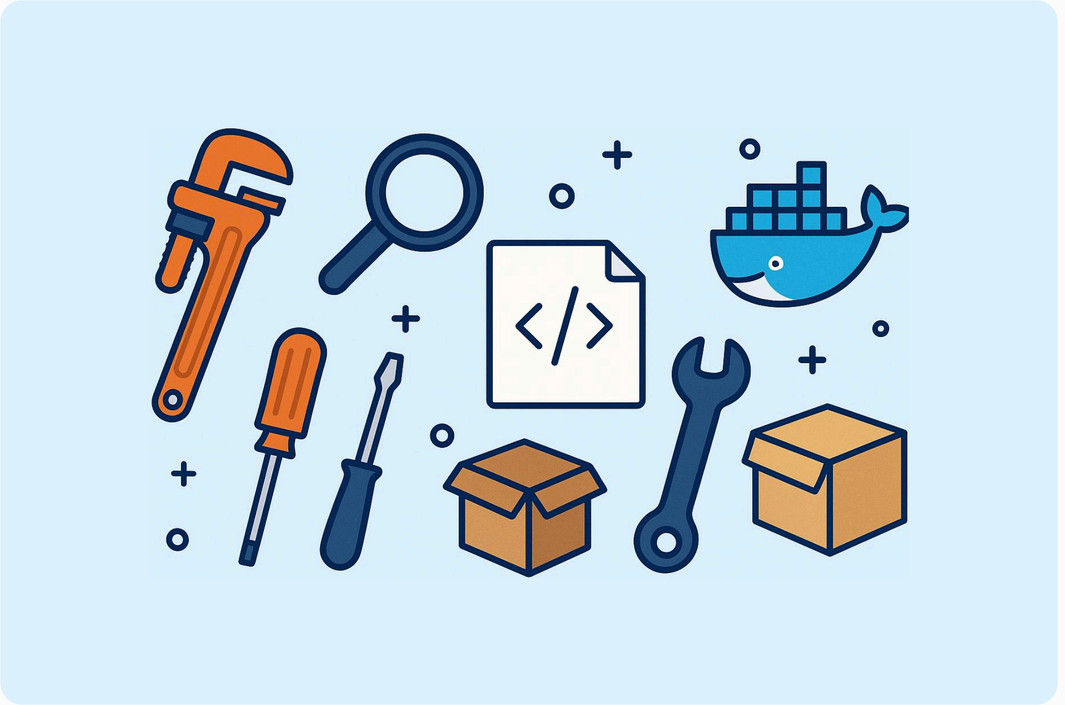
- Multi-stage builds = slim final image
- Use Alpine or minimal base images
- Leverage layered caching to skip redundant rebuilds
- .dockerignore = stop shipping junk

< SCAN ME FOR THE ARTICLE



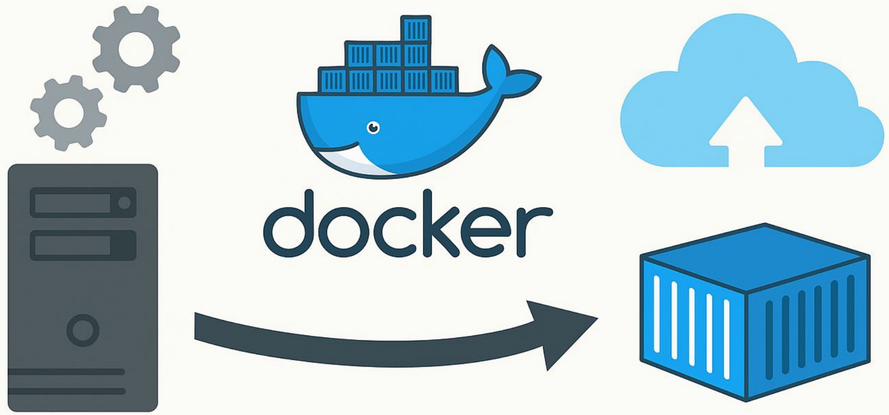
Tools of the trade.

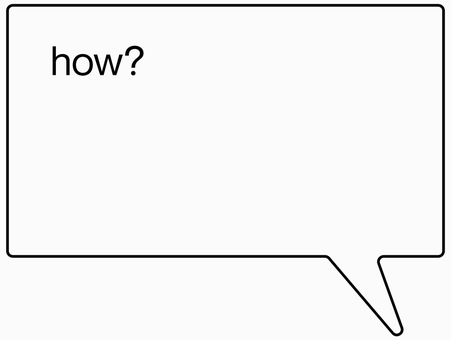
- Dive – explore what's bloating your image
- Hadolint – your dockerfile linter friend
- DockerSlim – auto-trims your images
- Snyk / Trivy – find and fix vulnerabilities

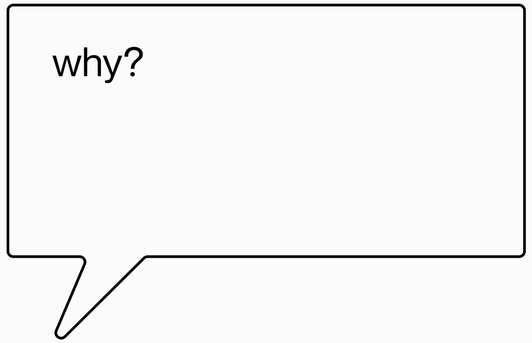


Hosting, evolved.

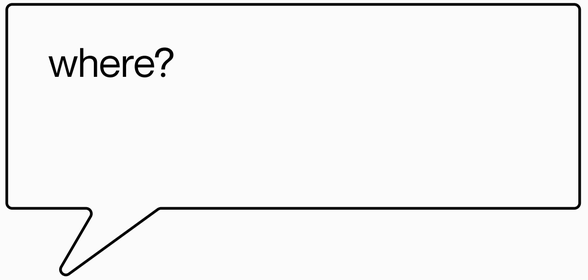
- From bare-metal to containers: we've come far
- Docker is the standard, but optimisation matters
- Small image = fast builds, cheaper deploys
- Continuous iteration is how pros roll







Questions?



Thank You

@Udhayaprakasha

Scan here to download the presentation slide