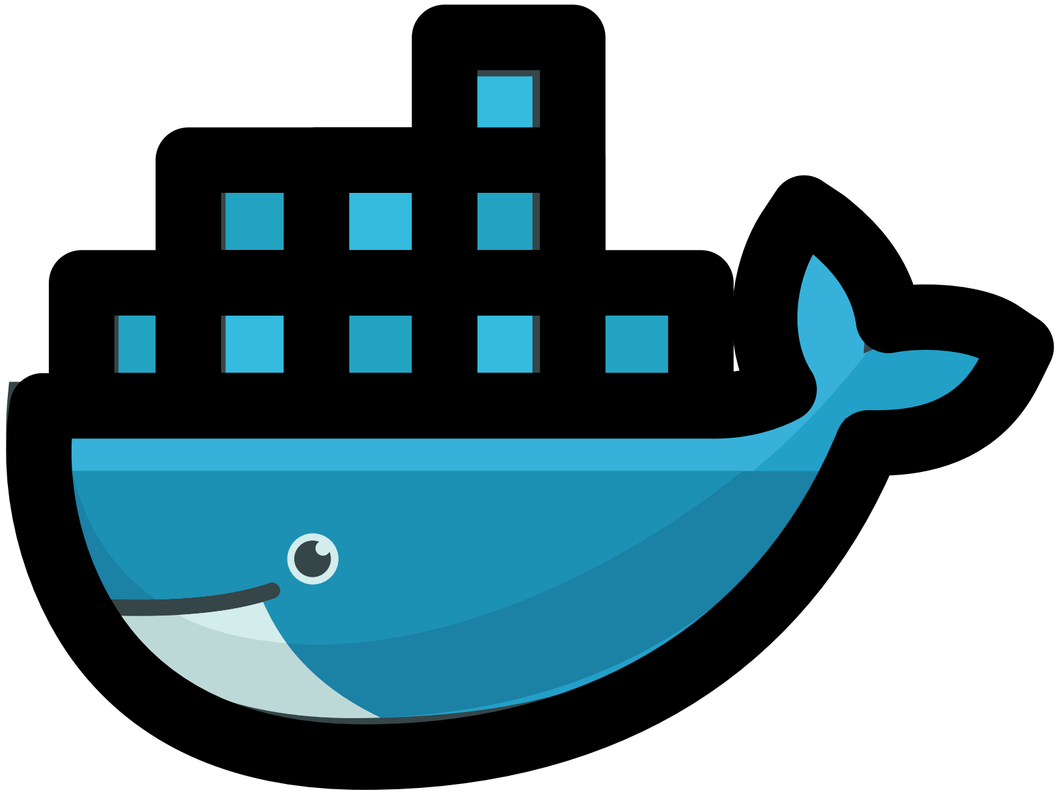
DOCKER ESSENTIALS



- **Class:** 22MMT - Linux operating system and applications
- **Teacher:** Le Ha Minh
- **Group:** 01
- **Students:**
  - 22127154 - Nguyen Gia Huy
  - 22127210 - Pham Anh Khoi
  - 22127420 - Nguyen Ha Nam Tran

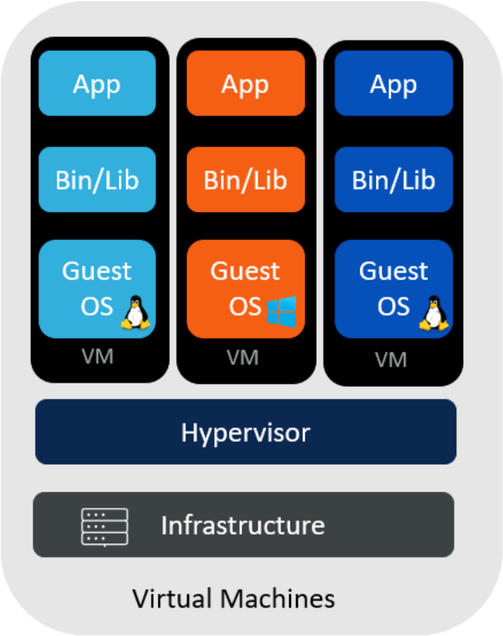
TABLE OF CONTENT

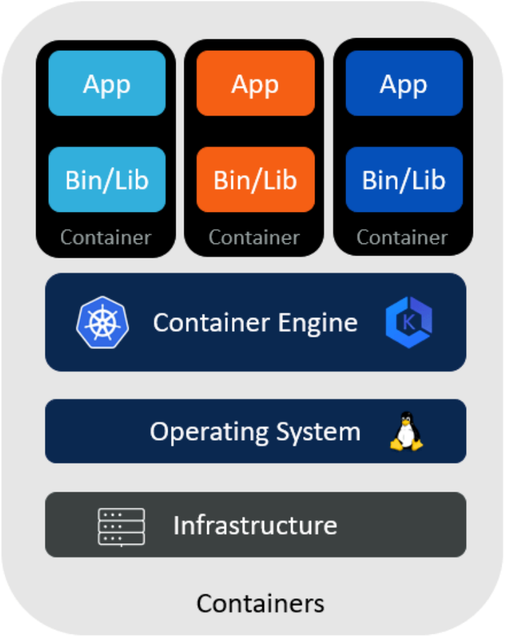
- Foundations
- Orchestration
- Demo

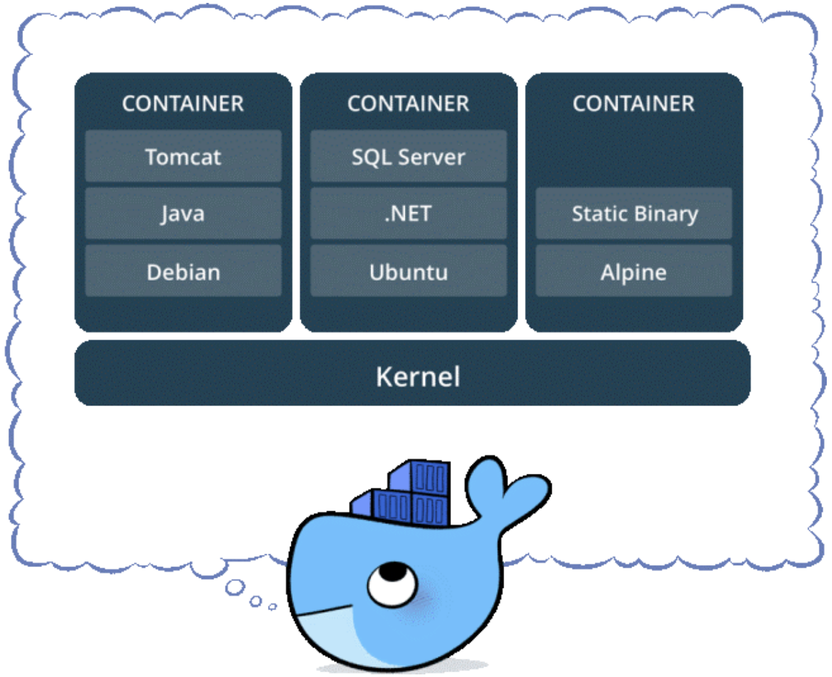
Introduce to Docker

What is Docker?

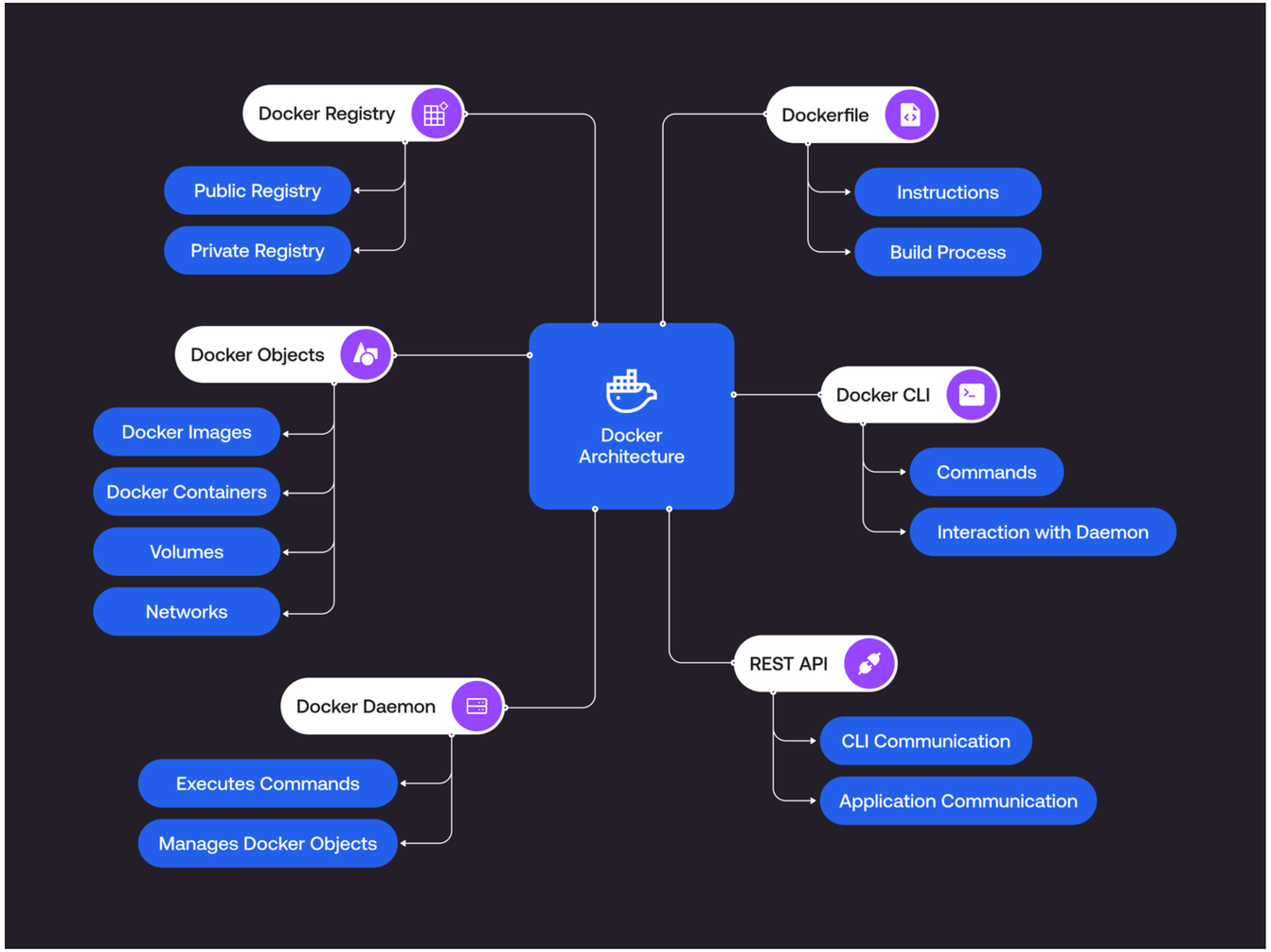
- Docker is a platform to build, ship, and run applications inside containers.
- Containers include everything needed to run the app: code, dependencies, runtime.

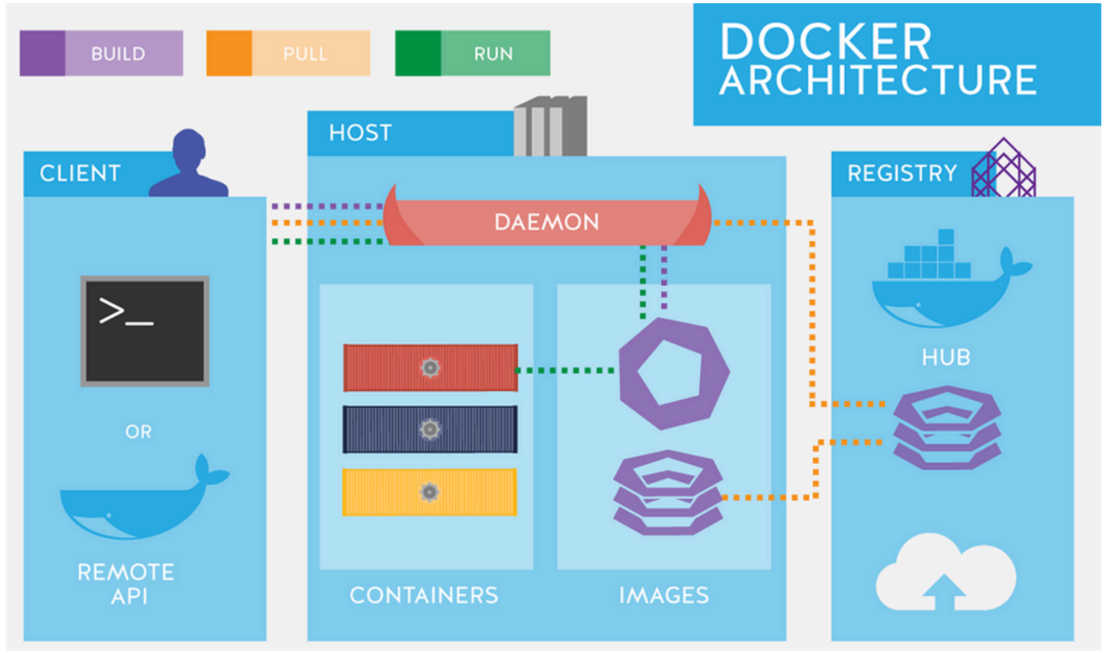






Docker Architecture



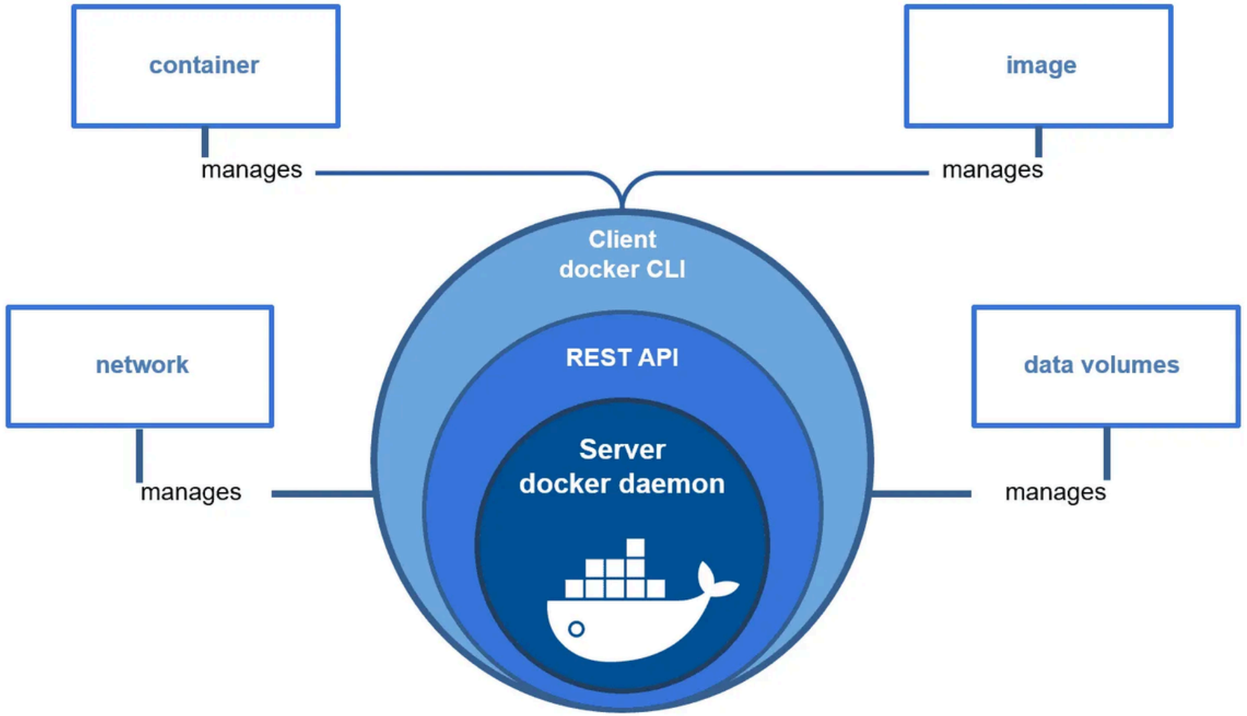


Docker Engine

The core client-server technology that runs and manages containers.

- **Docker Daemon**: Manages containers, images, networks, and volumes
- **Docker API**: Interface for tools and Docker Client to interact with the Daemon.
- **Docker CLI**: Command-line tool for sending commands

.



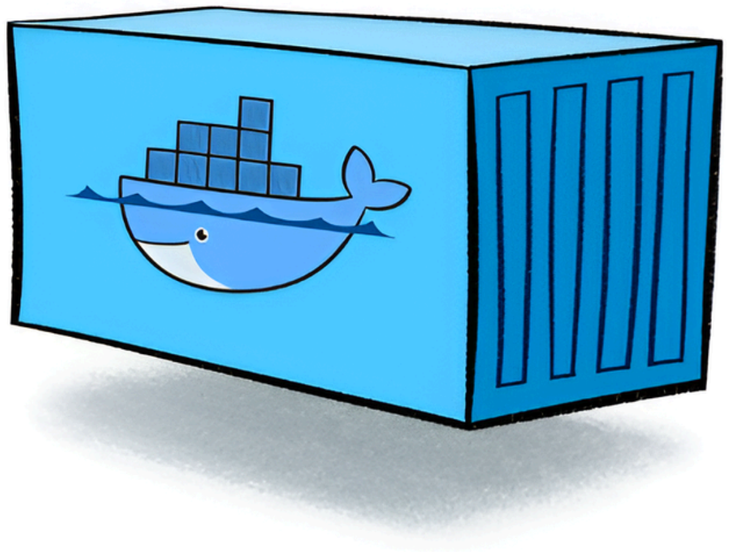
Docker Container

Runnable instance of an image containing an app and its dependencies.

Components:

- **Application**: Core code or service (e.g., web server, database).
- **Dependencies**: Libraries, runtime, and configs for consistent execution.
- **Isolation**: Runs in a sandbox, sharing the host OS kernel.

.



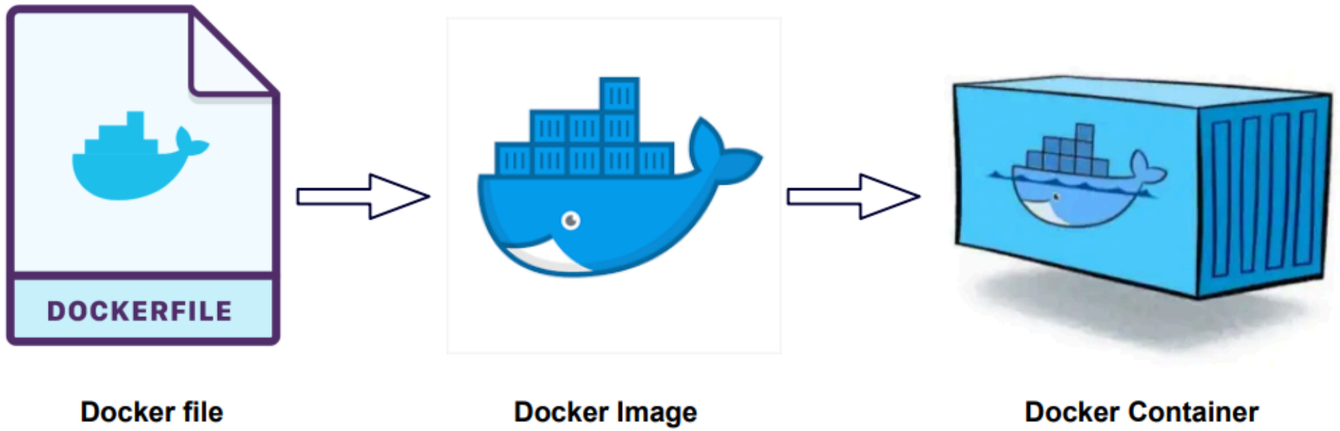
DockerFile & Image

Read-only template for creating **Docker Containers**.

How to create:

- **Dockerfile**: Script defining image build steps.
- Contents: Includes OS, app code, and libraries.

.



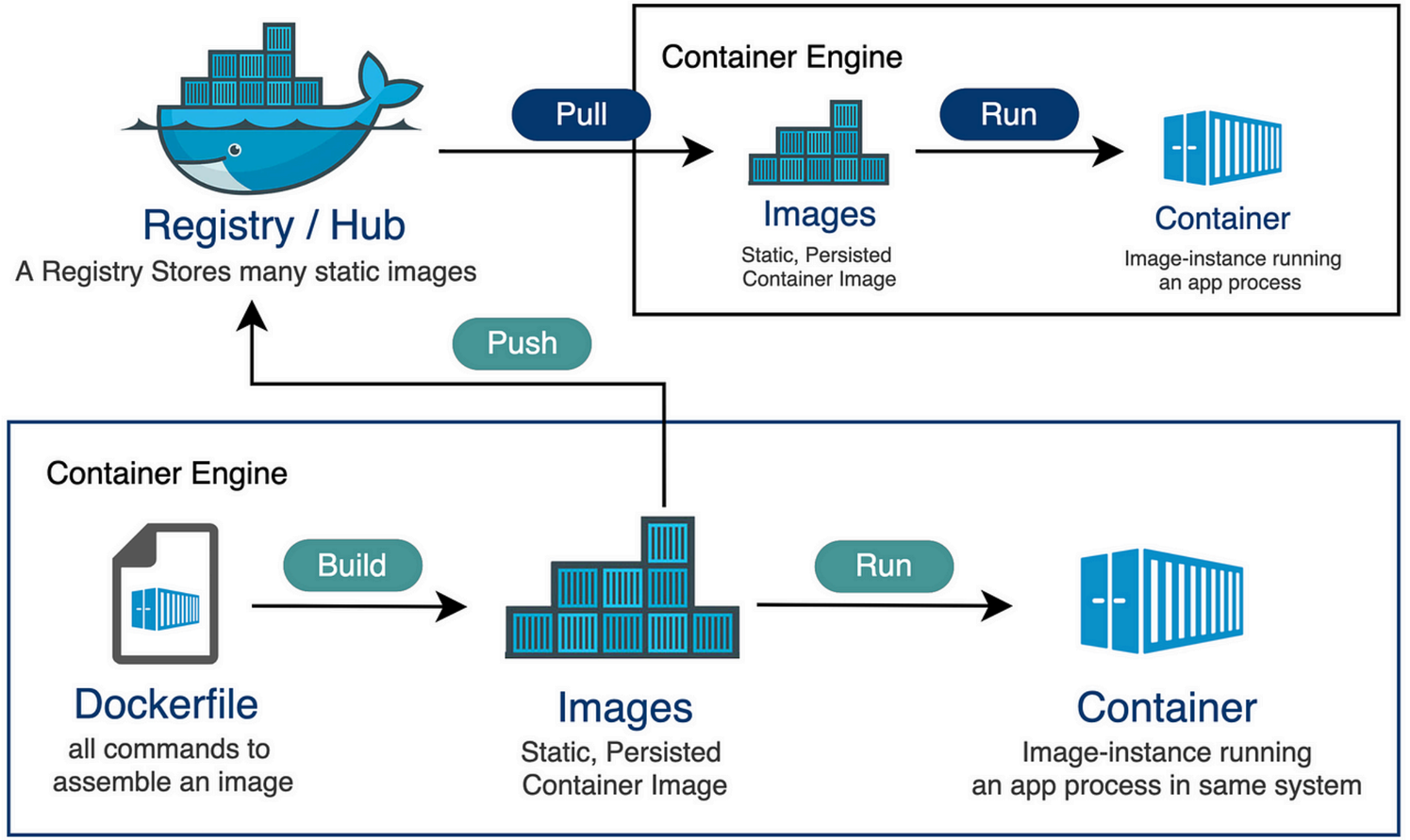
portfolio > Dockerfile > ...

You, 2 hours ago | 3 authors (namtran1205 and others)

```dockerfile
# Multi-stage build for Next.js application
FROM node:22-alpine AS base

# Install dependencies only when needed
FROM base AS deps
RUN apk add --no-cache libc6-compat
WORKDIR /app

# Copy package files
COPY package.json package-lock.json* ./
RUN npm ci

# Rebuild the source code only when needed
FROM base AS builder
WORKDIR /app
COPY --from=deps /app/node_modules ./node_modules
COPY . .

# Next.js collects completely anonymous telemetry data
ENV NEXT_TELEMETRY_DISABLED=1

# Build the application
RUN npm run build
```

Docker Registry

Storage and distribution system for Docker images.

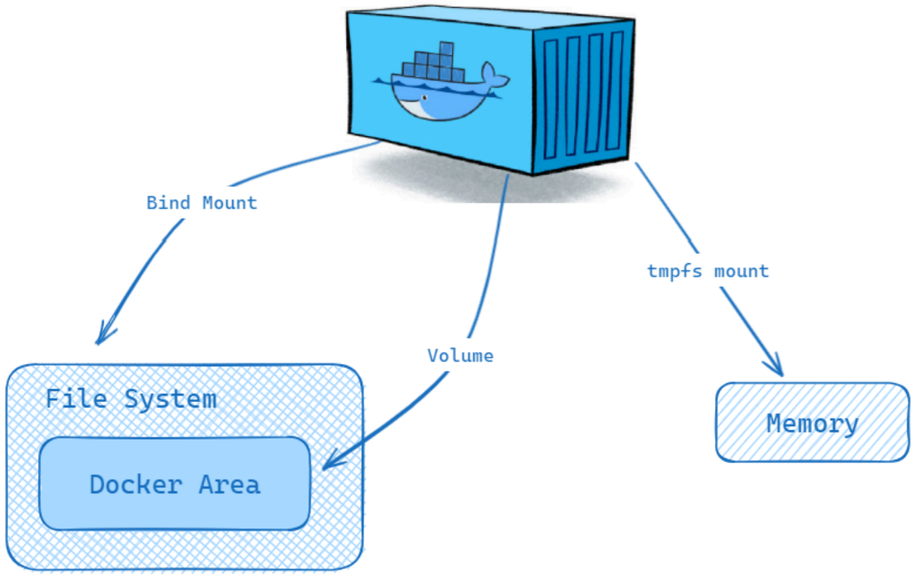


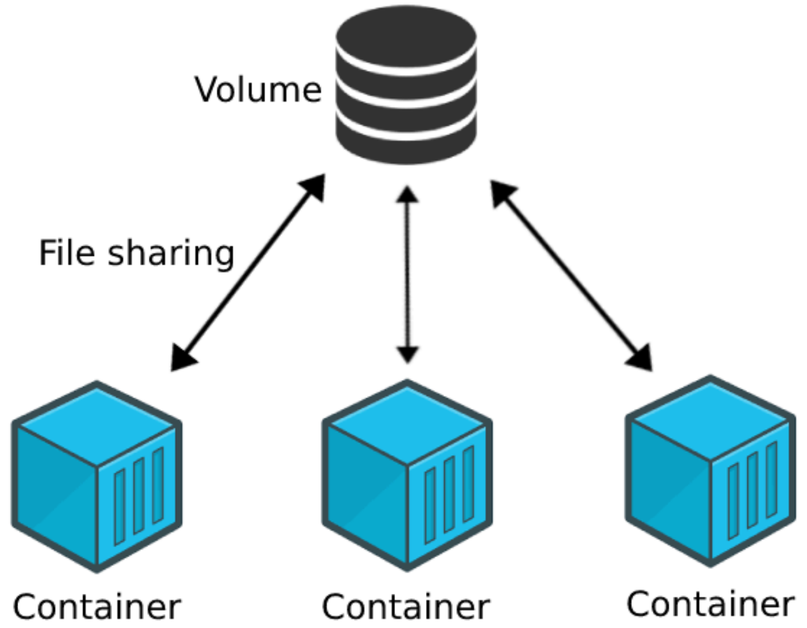
Docker Volumes

Purpose: Preserves data when containers stop or are deleted.

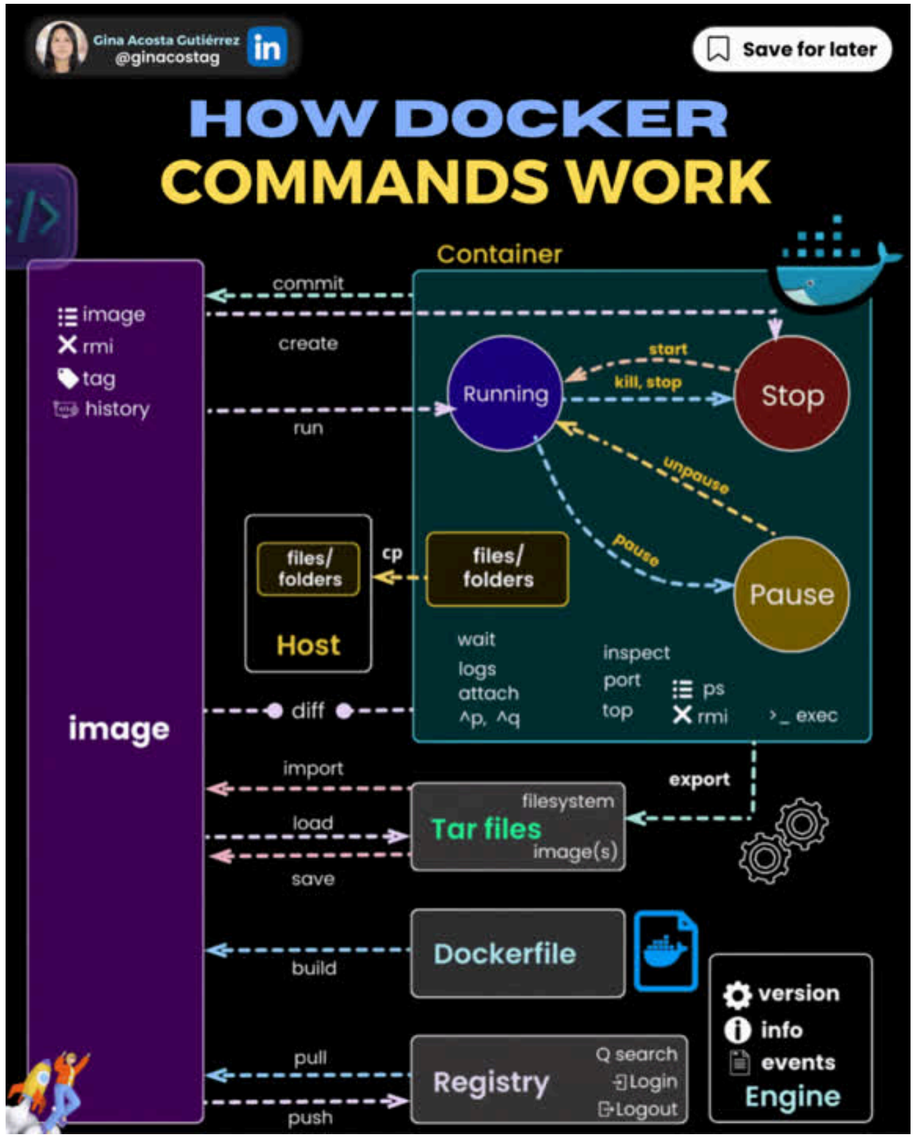
Types:

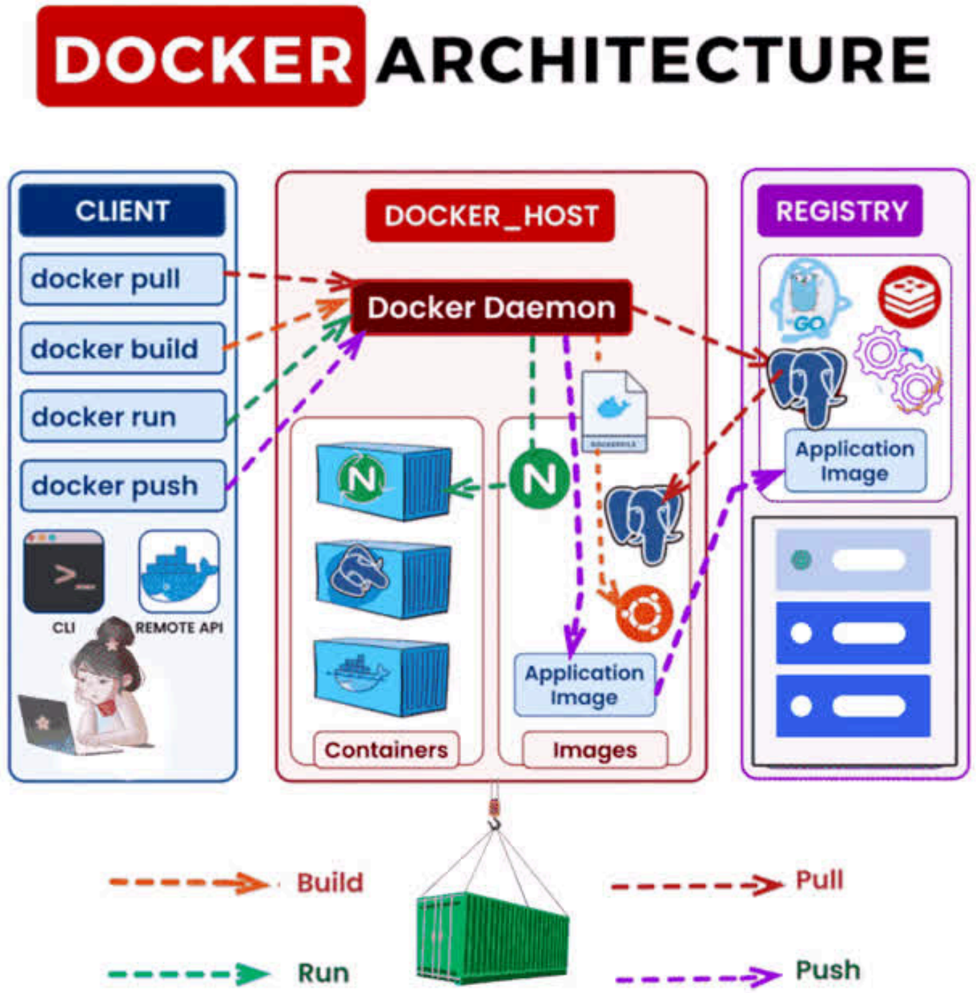
- Bind Mounts: Links container to host filesystem.
- Named Volumes: Docker-managed, reusable storage.
- Anonymous Volumes: Temporary, Docker-managed storage.



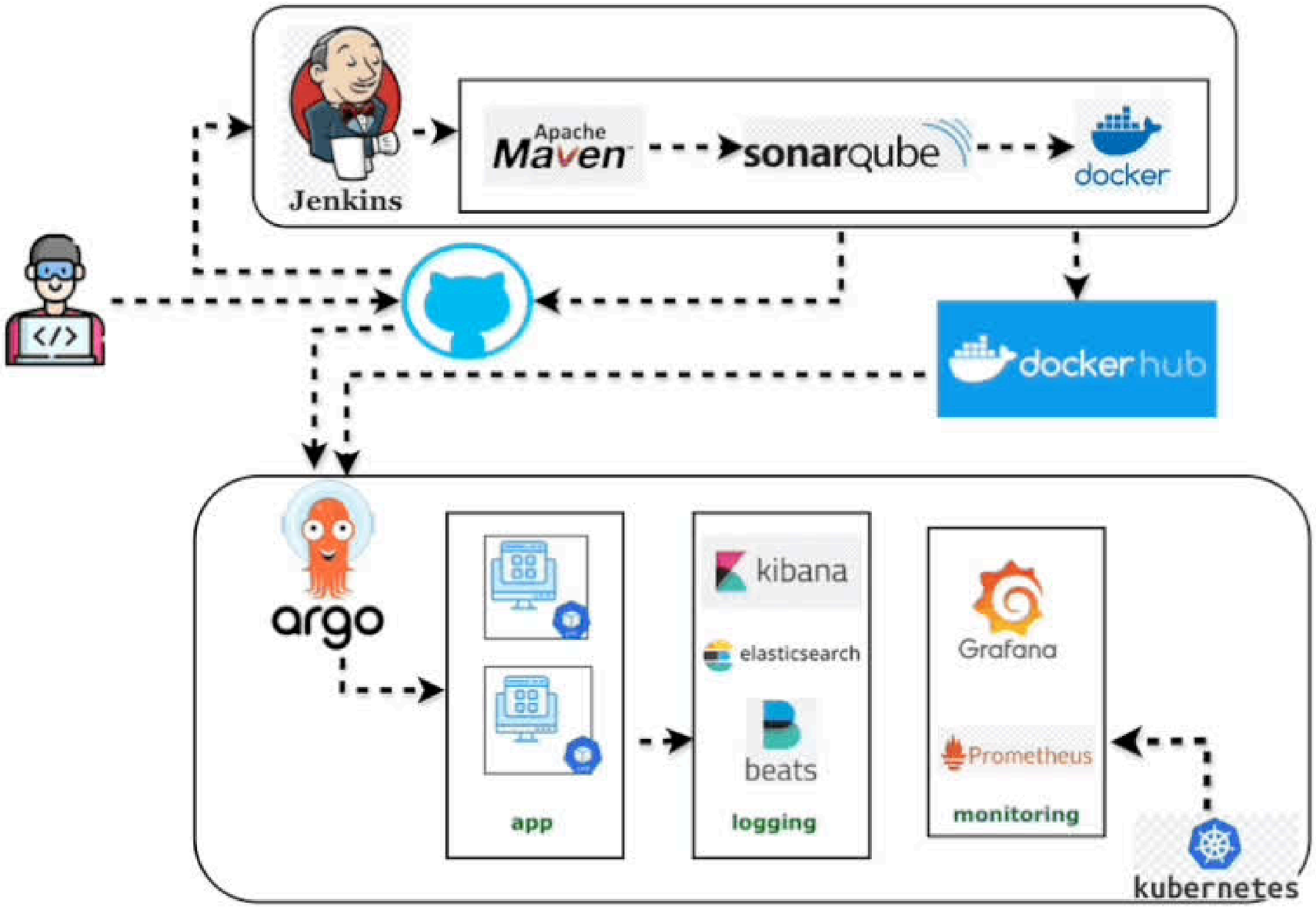


Docker Commands

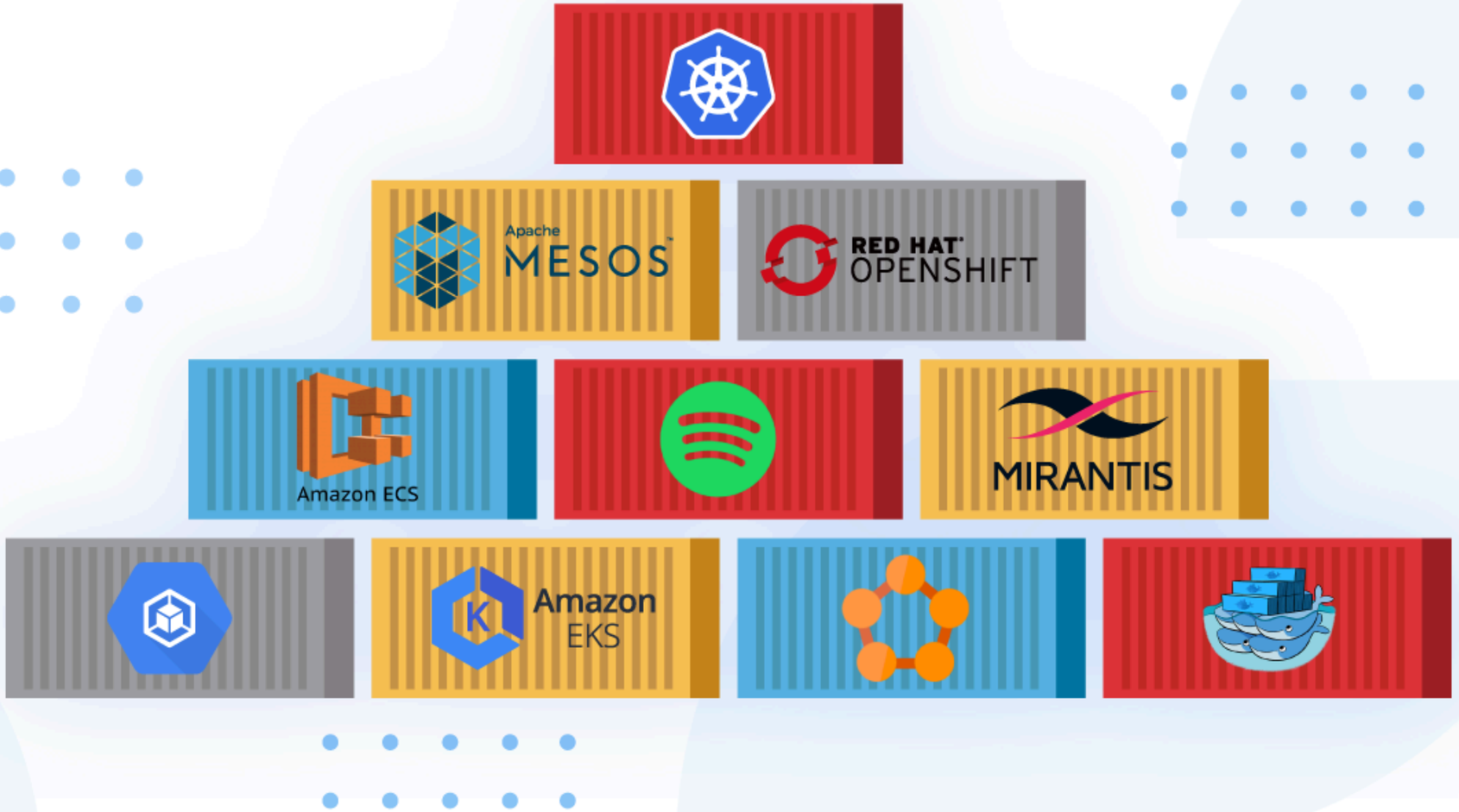




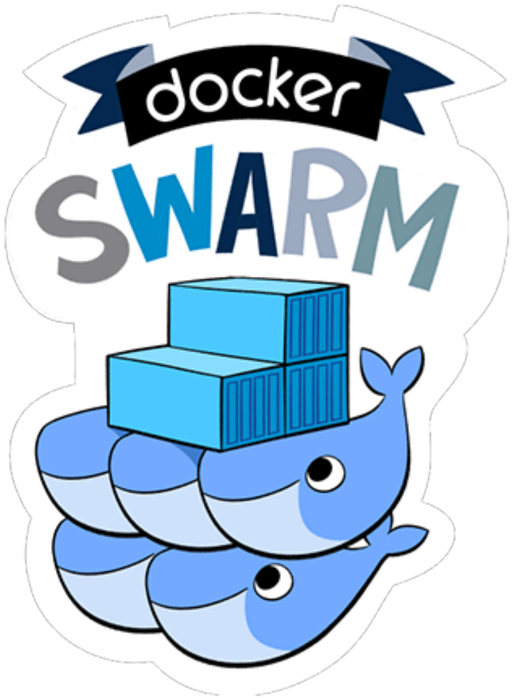
Docker in CI/CD



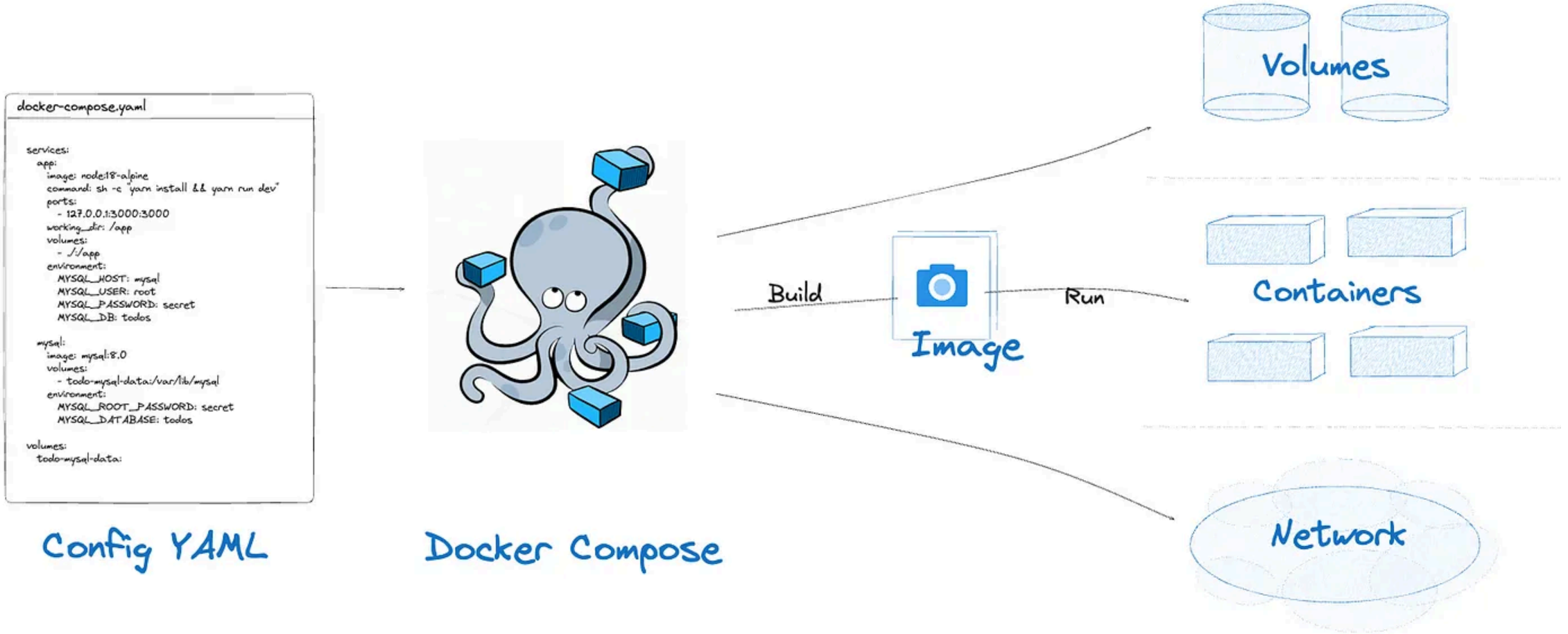
Container Orchestration



Evolution: Compose → Swarm → Kubernetes



Docker Compose Architecture



Compose Deep Dive – File Structure

```yaml
services:
  # MongoDB Container
  ▷Run Service
  mongodb: ···

  # Redis Container
  ▷Run Service
  redis: ···

  # Backend API
  ▷Run Service
  backend: ···

  # Frontend Application
  ▷Run Service
  frontend: ···

# Networks
networks:
  quiz-network:
    driver: bridge
    name: docker-quiz-network

# Volumes (only used when MongoDB/Redis containers are active)
volumes:
  mongodb_data:
    driver: local
    name: docker-quiz-mongodb-data
  redis_data:
    driver: local
    name: docker-quiz-redis-data
```

```yaml
services:
  # MongoDB Container
  ▷Run Service
  mongodb:
    image: mongo:latest
    container_name: docker-quiz-mongodb
    restart: unless-stopped
    profiles: ["local", "local-dev"]
    environment:
      MONGO_INITDB_ROOT_USERNAME: ${MONGO_ROOT_USER:-admin}
      MONGO_INITDB_ROOT_PASSWORD: ${MONGO_ROOT_PASSWORD:-password123}
      MONGO_INITDB_DATABASE: ${MONGO_INIT_DB:-docker_quiz_game}
      # Pass environment for init script
      ENVIRONMENT: ${ENVIRONMENT:-production}
    ports:
      - "${MONGO_PORT:-27017}:27017"
    volumes:
      - mongodb_data:/data/db
      - ./mongo-init.js:/docker-entrypoint-initdb.d/mongo-init.js:ro
    networks:
      - quiz-network
    healthcheck:
      test: ["CMD", "mongosh", "--eval", "db.adminCommand('ping')"]
      interval: 30s
      timeout: 10s
      retries: 3
      start_period: 40s

  # Redis Container
  ▷Run Service
  redis: ···
```

Compose Deep Dive – File Structure

```yaml
services:
  # Backend API
  backend:
    build:
      context: ./backend
      dockerfile: Dockerfile
    container_name: docker-quiz-backend
    restart: unless-stopped
    environment: ···
    ports:
      - "${BACKEND_PORT:-8000}:8000"
    networks:
      - quiz-network
    # Conditional dependencies based on profile
    depends_on:
      mongodb:
        condition: service_healthy
        required: false
      redis:
        condition: service_healthy
        required: false
    profiles: ···
    healthcheck:
      test: ["CMD", "curl", "-f", "http://localhost:8000/health"]
      interval: 30s
      timeout: 10s
      retries: 5
      start_period: 60s
    deploy: ···
```

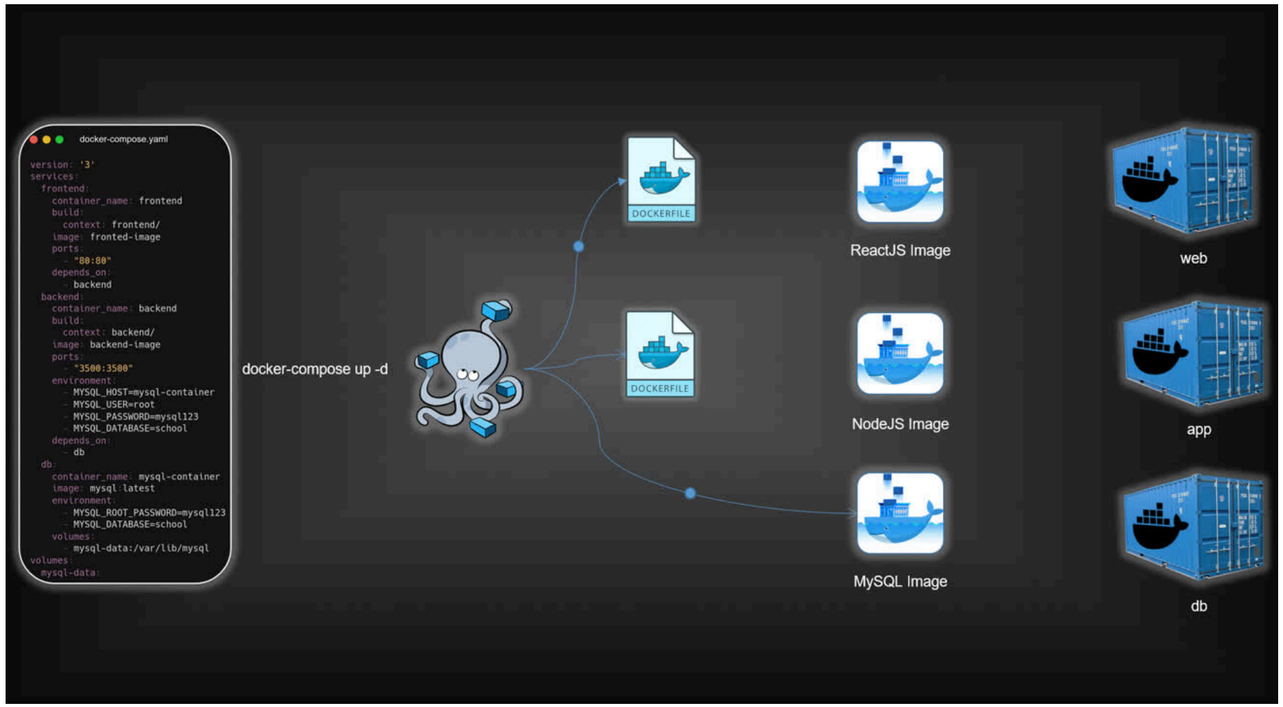
docker-compose.override.yml

```yaml
services:
  # Backend API (using pre-built image)
  backend:
    image: hzeroxium/docker-quiz-game-backend:latest
    container_name: docker-quiz-backend-prod
    restart: unless-stopped
    environment: ···
    ports:
      - "${BACKEND_PORT:-8000}:8000"
    networks:
      - quiz-network
    profiles:
      - cloud
    healthcheck:
      test: ["CMD", "curl", "-f", "http://localhost:8000/health"]
      interval: 30s
      timeout: 15s
      retries: 5
      start_period: 60s
    deploy:
      resources:
        limits:
          memory: 256M
          cpus: "0.5"
        reservations:
          memory: 128M
          cpus: "0.25"
    logging:
      driver: json-file
      options:
        max-size: "10m"
        max-file: "3"
```

docker-compose.yml

Compose Commands

```
docker-compose up -d --build --profile <PROFILE_NAME>
# Start containers in detached mode, rebuild images, using the specified profile.

docker-compose up <SERVICE_NAME>
# Start only the specified service and its dependencies in the foreground.

docker-compose down -v
# Stop and remove all containers, networks, and **volumes** created by `up`.

docker-compose stop <SERVICE_NAME>
# Stop the specified running service container without removing it.

docker-compose build <SERVICE_NAME>
# Build or rebuild the image for the specified service.

docker-compose ps
# List all running (and stopped) containers managed by this Compose project.

docker-compose logs
# Display logs from all services defined in the `docker-compose.yml`.

docker-compose pull
# Pull the latest images from the Docker registry for all services.
```

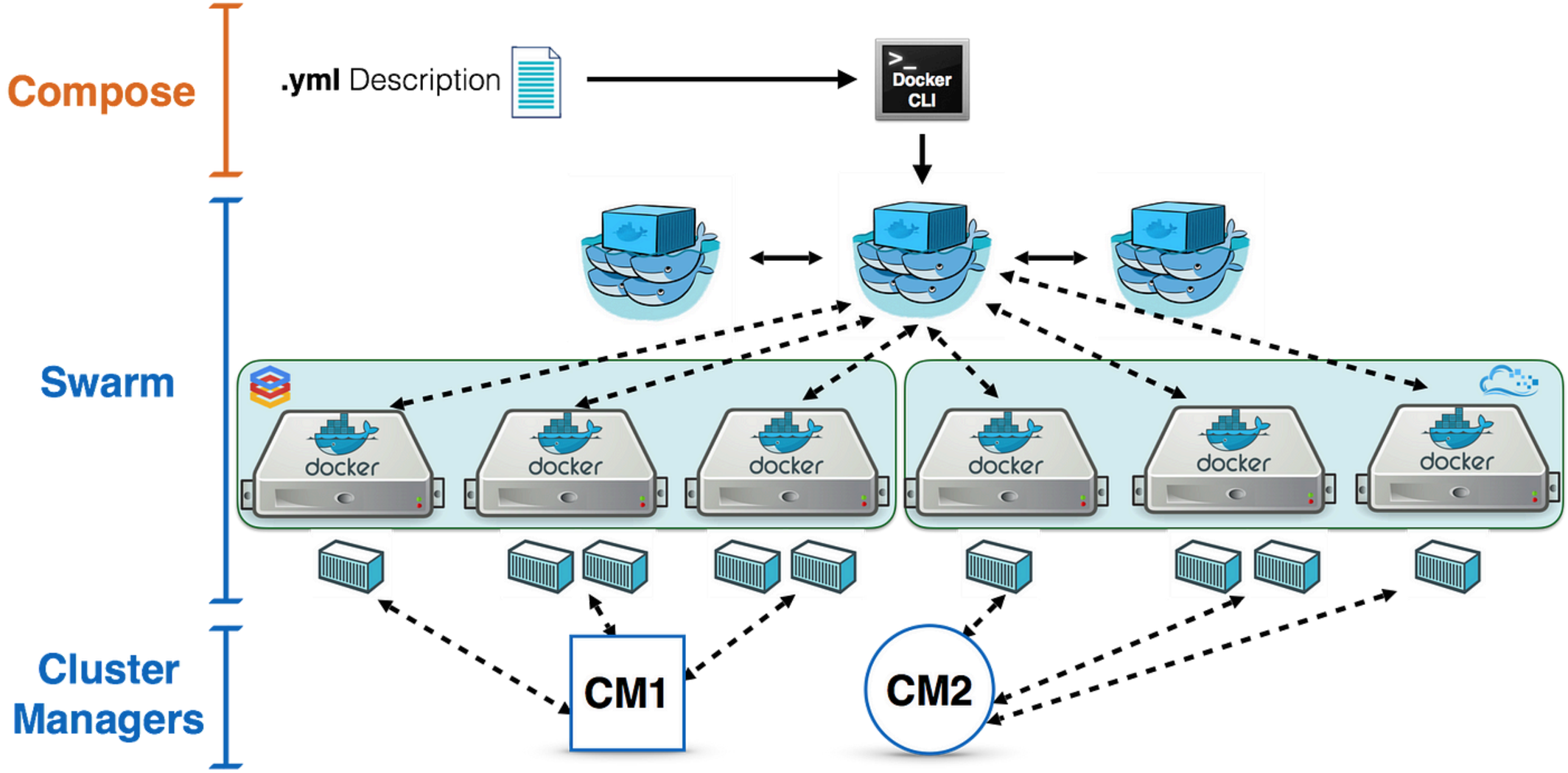


Compose Best Practices

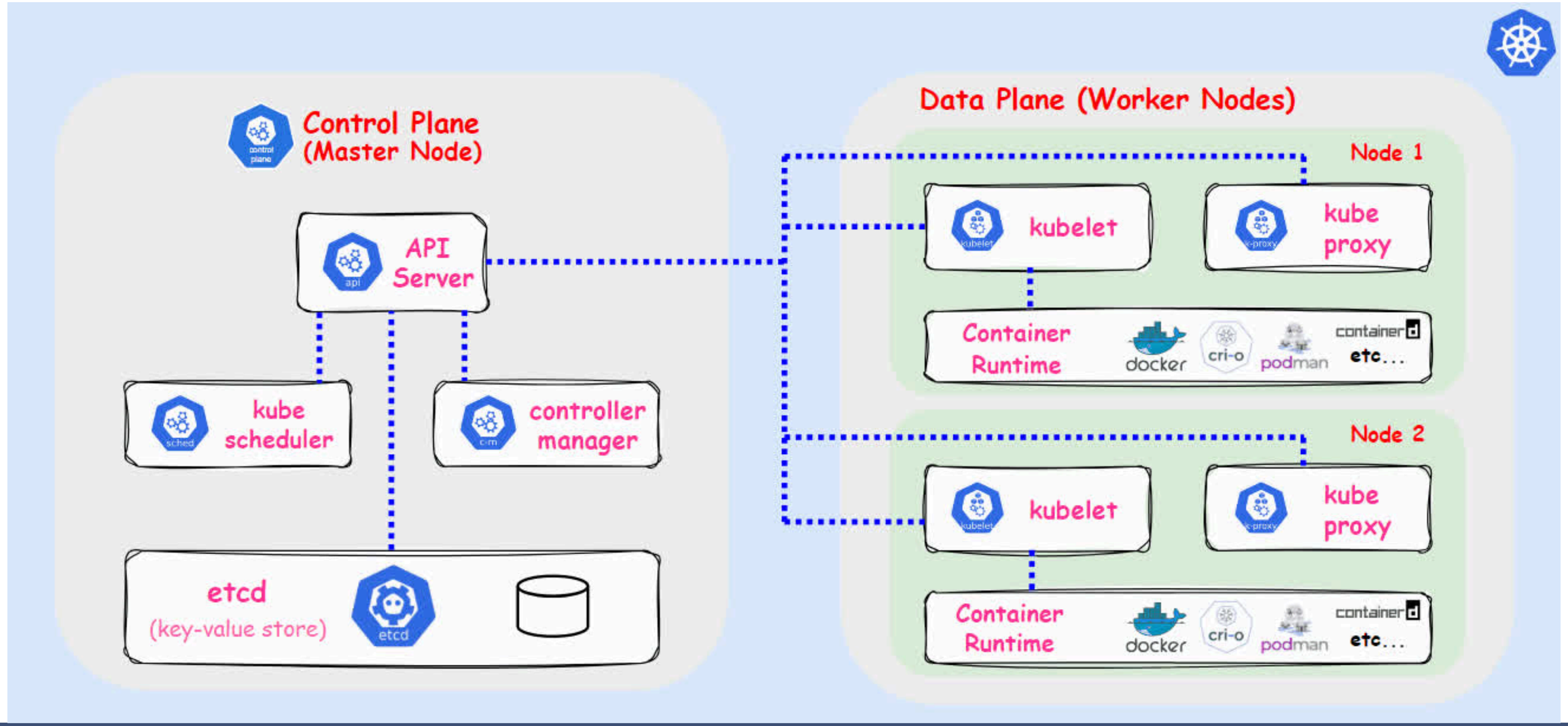
Best Practices:

- Fix the image version to avoid discrepancies
- Set depends_on to only start the order
- Use healthcheck to ensure service is ready
- Keep the file short, only declare services that are really needed
- ...

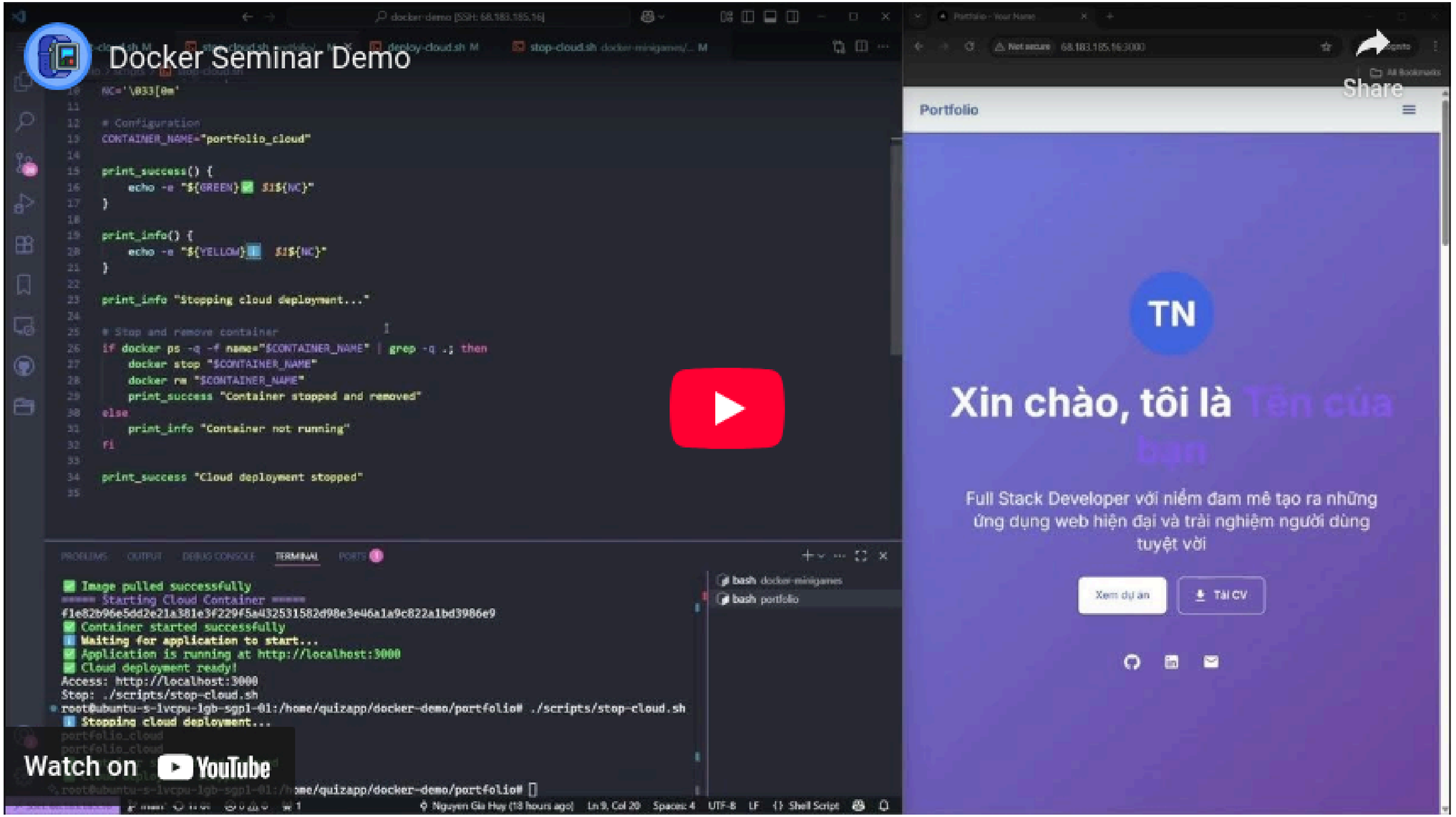
Docker Swarm Overview



Kubernetes Overview



Demo


Docker Seminar Demo
Watch on YouTube
Portfolio
TN
Xin chào, tôi là Tên của bạn
Full Stack Developer với niềm đam mê tạo ra những ứng dụng web hiện đại và trải nghiệm người dùng tuyệt vời
Xem dự án
Tải CV

THANK YOU

For your attention