■ Docker Full Tutorial + Debugging Guide (English + Kannada)

==========

1. Docker Objects

Image:

English → Blueprint for container.

Kannada → Container build ■■■ ■■■■■.

Container:

English → Running instance of image.

Kannada → Image-■ running copy.

Volume:

English → Persistent storage.

Kannada → Data safe ■■■ storage.

Network:

English → Communication between containers.

Kannada → Container-■■ ■■■■■■.

==========

2. Dockerfile (Step by Step)

Example: Node.js App

```
FROM node:16
WORKDIR /app
COPY package*.json ./
RUN npm install
COPY . .
EXPOSE 3000
CMD ["node", "app.js"]
```

English: FROM=base, WORKDIR=set dir, COPY=copy files, RUN=install, EXPOSE=port, CMD=start app

Kannada: FROM=base, WORKDIR=folder, COPY=files ■■■■■■, RUN=install, EXPOSE=port, CMD=start command

==========

3. Multi-Stage Dockerfile

Example: Java JAR build

Stage 1: Build with Maven

Stage 2: Run with JRE (small size)

English → Optimized images.

Kannada → Image size ■■■■■.

==========

4. Docker Volumes & Networks

Volume Example:

docker volume create mydata

docker run -d -v mydata:/app/data nginx

Network Example:

docker network create mynet

docker run -d --network=mynet --name db mysql

docker run -d --network=mynet --name app myapp

English: Use volumes for persistent data, networks for connectivity.

Kannada: Volume = data safe, Network = ■■■■■■.

==========

5. Debugging Containers

Login:

docker exec -it container bash

Logs:

docker logs container

Inspect:

docker inspect container

==========

6. Common Issues & Fixes

Container not starting:

Check logs → Fix CMD/ENTRYPOINT.

Port already in use:

Change port mapping.

CrashLoopBackOff:

Check env vars, dependencies, permissions.

Volume issue:

Check docker volume ls, fix permissions.

Network issue:

Use same network, container name.

Image build failed:

Check Dockerfile, rebuild with --no-cache.

High resource usage:

docker stats, use limits.

Logs too large:

Configure log rotation in /etc/docker/daemon.json.

Docker daemon down:

systemctl start docker

==========

7. Zero Downtime Deployment

Run multiple replicas + reverse proxy (NGINX).

English: Deploy new container, switch traffic, stop old.

Kannada: ■■■■■ deploy ■■■■, proxy switch ■■■■, ■■■■■■ stop.

==========

8. Docker Commands (Build, Run, Start)

---

docker build -t myapp .

docker run -d -p 8080:8080 myapp

docker ps, docker stop, docker start, docker rm, docker rmi

==========

9. Docker Compose Example

---

docker-compose.yml

version: '3'

services:

db:

image: mysql

app:

build: .

ports:

- "8080:8080"

English: Run DB + App together.

Kannada: DB + App ■■■■■■■ start.

==========

10. Interview Q&A;

Q1: Difference image vs container?

English → Image=template, Container=running.

Kannada → Image=■■■■■, Container=copy.

Q2: CMD vs ENTRYPOINT?

CMD=default, ENTRYPOINT=fixed.

Q3: Why multi-stage Dockerfile?

English → Reduce size.

Kannada → Size ■■■■■.

Q4: How debug failing container?

Logs, inspect, exec.

Q5: How to persist data?

Use volumes.

==========

Summary:

Docker = Build (Dockerfile) + Run (Containers) + Persist (Volumes) + Connect (Networks) + Orchestrate (Compose).

Debugging = Logs + Exec + Inspect + Fix configs, ports, permissions.