Docker Multi-Service Setup (Intermediate)

Goal:
Learn how to run multiple services (NGINX + App + DB) using Docker Compose.

STEP 1: Project Structure
Make a project folder with this structure:

my-project/
■■■ docker-compose.yml
■■■ .env
■■■ nginx/
■ ■■■ Dockerfile
■ ■■■ default.conf
■■■ app/
■ ■■■ Dockerfile
■ ■■■ index.html
■■■ db/
■■■ Dockerfile

STEP 2: Environment Variables (.env)
Add your config values:
DOMAIN=myproject.local
DB_USER=user
DB_PASS=pass
DB_NAME=mydb

STEP 3: docker-compose.yml
Define services and link them with a custom network. Mount volumes for persistence.

STEP 4: Dockerfiles
Each service has its own Dockerfile. Use Alpine/Debian base, install only what's needed.

Example: NGINX Dockerfile
FROM nginx:alpine
COPY default.conf /etc/nginx/conf.d/

Example: App Dockerfile (static site)
FROM alpine
RUN apk add --no-cache nginx && mkdir -p /var/www/html
COPY index.html /var/www/html/index.html

Example: DB Dockerfile (MariaDB)
FROM mariadb:10.5
ENV MYSQL_ROOT_PASSWORD=$DB_PASS
ENV MYSQL_DATABASE=$DB_NAME
ENV MYSQL_USER=$DB_USER
ENV MYSQL_PASSWORD=$DB_PASS

STEP 5: Reverse Proxy Setup
Configure NGINX to route to your app. Use default.conf to define server blocks.

STEP 6: Volumes and Persistence
Use Docker volumes to persist DB and app data across container restarts.

STEP 7: Build & Run
Run:
docker-compose up --build
Check logs and open http://localhost (or your defined domain).

STEP 8: Cleanup
docker-compose down -v

Done right, this sets you up for the full Inception project: self-contained, layered, clean.
Use this to experiment with networking, volumes, configs, and isolation.