Modern Docker, production-ready setup:

Here’s a modern, production-ready setup that follows current best practices for Dockerizing a React app.

1. A **multi-stage production Dockerfile** (build with Node, serve with Nginx).
2. An **optional dev Dockerfile** for hot-reload.
3. A tiny **nginx config** tuned for static React.
4. A **.dockerignore** to keep images small.
5. **Build/run commands** and notes about API\_URL handling.
6. **Production Dockerfile (recommended)**

# ----------- STAGE 1: Build the React app -----------

# Use a small, current LTS Node image. Alpine keeps it lean.

FROM node:22-alpine AS builder

# ^ Update this to the latest LTS over time (e.g., node:22-alpine)

# Set a non-root user for security (Node image includes 'node' user)

USER node

# Create and set working dir (owned by 'node' user)

WORKDIR /app

# Copy only dependency manifests first to leverage Docker layer cache

# (If you use pnpm or yarn, copy the corresponding lockfile instead.)

COPY --chown=node:node package.json package-lock.json\* ./

# Install exact deps based on lockfile, no dev scripts yet

# npm ci is faster, reproducible, and fails if lockfile is missing

RUN npm ci

# Now copy the rest of the project (source code, configs, etc.)

COPY --chown=node:node . .

# Build-time API URL (CRA: REACT\_APP\_\*, Vite: VITE\_\*)

# Pass it with: --build-arg APP\_API\_URL=https://api.myapp.com

ARG APP\_API\_URL

# Export framework-specific variable for the build:

# - If Create React App, reference process.env.REACT\_APP\_API\_URL in code

# - If Vite, reference import.meta.env.VITE\_API\_URL in code

ENV REACT\_APP\_API\_URL=$APP\_API\_URL \

VITE\_API\_URL=$APP\_API\_URL \

NODE\_ENV=production

# Build the optimized production bundle

RUN npm run build

# ----------- STAGE 2: Serve with Nginx -----------

FROM nginx:1.27-alpine AS runner

# ^ Keep Nginx up-to-date; 1.27 is current stable at the time of writing

# Copy a minimal, tuned nginx config

COPY ./infra/nginx/default.conf /etc/nginx/conf.d/default.conf

# Copy the compiled static files from the builder stage

# CRA/Vite output is usually in 'build' or 'dist'; adjust if needed.

COPY --from=builder /app/build /usr/share/nginx/html

# If your tool outputs to 'dist', use:

# COPY --from=builder /app/dist /usr/share/nginx/html

# Optional: drop privileges (nginx image already has 'nginx' user)

USER nginx

# Expose HTTP port

EXPOSE 80

# Healthcheck to ensure container is serving content

HEALTHCHECK --interval=30s --timeout=5s --retries=3 CMD wget -qO- http://localhost/ || exit 1

# Start nginx in the foreground

CMD ["nginx", "-g", "daemon off;"]

# ----------------------------------------------------------------------------------------------------------------------------

**2) Optional: Dev Dockerfile (hot reload)**

Use this if you want a containerised dev environment. It mounts your code and runs the dev server.

# Dev container with hot-reload

FROM node:22-alpine

# Create non-root workspace

USER node

WORKDIR /app

# Install deps based on lockfile (copied separately for caching)

COPY --chown=node:node package.json package-lock.json\* ./

RUN npm ci

# Copy the rest of the project (or mount it via volumes in docker-compose)

COPY --chown=node:node . .

ENV NODE\_ENV=development \

CHOKIDAR\_USEPOLLING=true # helpful in Docker on Windows/Mac for file watching

EXPOSE 3000

# For CRA: "npm start", for Vite: "npm run dev"

CMD ["npm", "start"]

# ----------------------------------------------------------------------------------------------------------------------------

* 1. **Minimal Nginx config (save as infra/nginx/default.conf)**

# infra/nginx/default.conf

server {

listen 80;

server\_name \_;

# Path where we copied the React build

root /usr/share/nginx/html;

index index.html;

# Gzip basic static types

gzip on;

gzip\_types text/plain application/javascript application/json text/css text/xml image/svg+xml;

# Cache immutable assets by filename hash (CRA/Vite produce hashed file names)

location ~\* \.(?:js|css|png|jpg|jpeg|gif|svg|webp)$ {

expires 30d;

access\_log off;

try\_files $uri =404;

}

# React Router fallback: send all other routes to index.html

location / {

try\_files $uri /index.html;

}

# Example: if you want to reverse-proxy API requests in prod (optional)

# location /api/ {

# proxy\_pass https://api.myapp.com/;

# proxy\_set\_header Host $host;

# proxy\_set\_header X-Real-IP $remote\_addr;

# }

}

# ----------------------------------------------------------------------------------------------------------------------------

# 4) .dockerignore (highly recommended)

Create a .dockerignore at your project root to keep your image small and builds fast:

# Node/tooling

node\_modules

npm-debug.log

yarn.lock

pnpm-lock.yaml

# Build outputs (we build inside the container)

build

dist

# VCS / IDE

.git

.gitignore

.vscode

.idea

# OS files

.DS\_Store

Thumbs.db

# Tests, coverage, local caches

coverage

\*.local

# ----------------------------------------------------------------------------------------------------------------------------

# 5) Build & run

### Production (static with Nginx)

# Build (inject your API base URL at build time)

docker build \

--build-arg APP\_API\_URL=https://api.myapp.com \

-t my-react-app:prod .

# Run

docker run --rm -p 8080:80 my-react-app:prod

# Open <http://localhost:8080>

# ----------------------------------------------------------------------------------------------------------------------------

Dev (hot reload)

# Build dev image once

docker build -f Dockerfile.dev -t my-react-app:dev .

# Run and mount your source (adjust host path)

docker run --rm -it -p 3000:3000 \

-v "$PWD":/app \

-v /app/node\_modules \

my-react-app:dev

# Open <http://localhost:3000>

# ----------------------------------------------------------------------------------------------------------------------------

## Notes about API\_URL / environment variables

* **Static React builds** (CRA/Vite) **bake env values at build time**.
  + CRA expects variables prefixed with REACT\_APP\_ (e.g., REACT\_APP\_API\_URL).
  + Vite expects variables prefixed with VITE\_ (e.g., VITE\_API\_URL).
  + That’s why the Dockerfile uses ARG APP\_API\_URL → ENV REACT\_APP\_API\_URL and ENV VITE\_API\_URL → then runs npm run build.
* If you need to **change the API URL at runtime** (without rebuilding), use a small runtime config trick (e.g., serve a window.\_env.js that Nginx injects). If you want this, tell me and I’ll add the snippet.