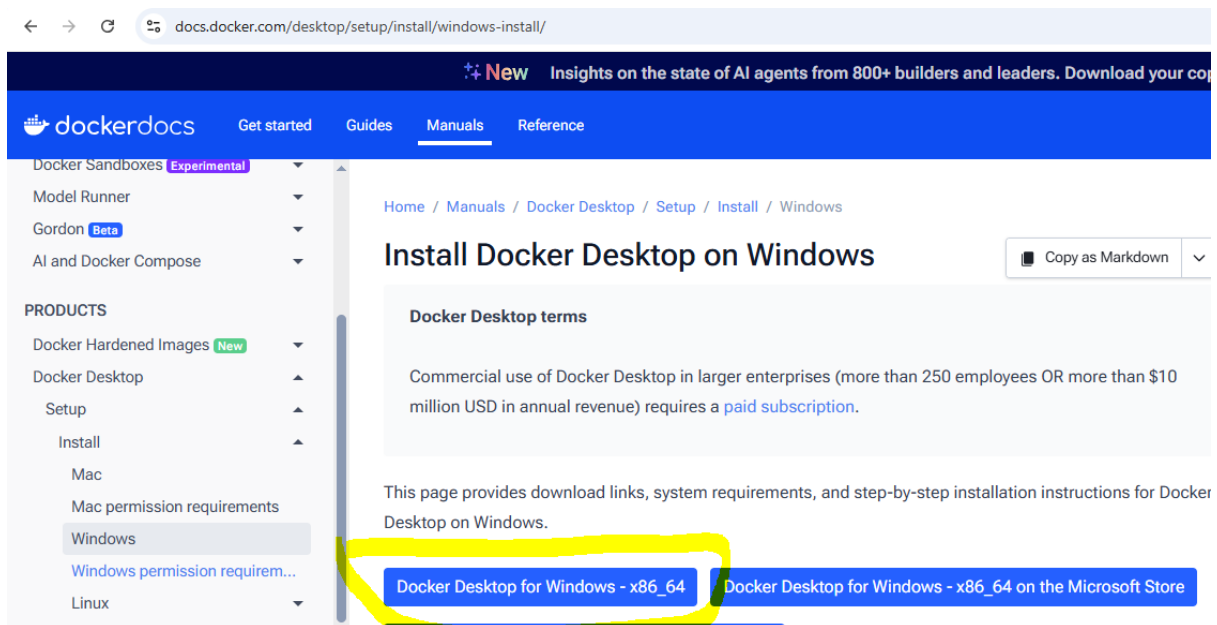


First download docker desktop on your pc:-

Form

<https://docs.docker.com/desktop/setup/install/windows-install/>



The screenshot shows the Docker Docs website at the URL `docs.docker.com/desktop/setup/install/windows-install/`. The page title is "Install Docker Desktop on Windows". The navigation menu includes "Get started", "Guides", "Manuals", and "Reference". The left sidebar lists various Docker products, with "Windows" selected under the "Install" section. The main content area includes a "Copy as Markdown" button, "Docker Desktop terms" (noting that commercial use in larger enterprises requires a paid subscription), and a paragraph stating that the page provides download links, system requirements, and installation instructions. At the bottom, there are two download buttons: "Docker Desktop for Windows - x86_64" (highlighted with a yellow circle) and "Docker Desktop for Windows - x86_64 on the Microsoft Store".

Download and install it .

Now how to crate your react js with vite project image using docker?

1. Vite React runs differently

Vite dev server runs on:

`http://localhost:5173`

So your Docker must expose **5173**, not 3000.

2. Create `Dockerfile` (Development mode)

In your project root:

```
FROM node:20

WORKDIR /app

COPY package.json package-lock.json ./
RUN npm install

COPY . .

EXPOSE 5173

CMD ["npm", "run", "dev", "--", "--host"]
```

As shown below in notepad++ :-

```
F:\falak-web\Dockerfile - Notepad++
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
Dockerfile
1 FROM node:20
2
3 WORKDIR /app
4
5 COPY package.json package-lock.json ./
6 RUN npm install
7
8 COPY . .
9
10 EXPOSE 5173
11
12 CMD ["npm", "run", "dev", "--", "--host"]
```

3. Create .dockerignore

```
node_modules
dist
.git
npm-debug.log
```

Why `--host` is required

```
npm run dev -- --host
```

- Without `--host`, Vite only works inside container
 - With it, you can access from browser
-

□ 4. Build Docker image

```
docker build -t vite-react-app .
```

▶ □ 5. Run container

```
docker run -p 5173:5173 vite-react-app
```

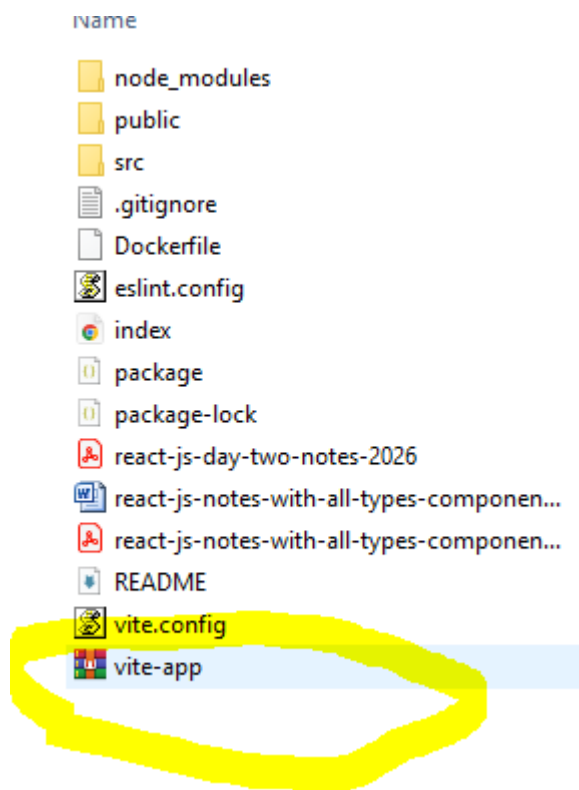
6. Open in browser

<http://localhost:5173>

and save it on locally :-

```
docker save -o vite-app.tar vite-react-app
```

then your project folder you will see vite-app.tar



Production setup (best practice for Vite + React)

In production, you should NOT use:

- `npm run dev`
- Node server running Vite
- large dev images (~400MB+)

Instead you use:

- **Build once** → **Serve static files with Nginx**
-

□ 1. Final Production Dockerfile

```
# Step 1: Build the app
FROM node:20 AS build

WORKDIR /app

COPY package.json package-lock.json ./
RUN npm install

COPY . .
RUN npm run build

# Step 2: Serve using Nginx
FROM nginx:alpine

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

EXPOSE 80

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

As shown below in notepad++

```
F:\falak-web\Dockerfile - Notepad++
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
Dockerfile
1 # Step 1: Build the app
2 FROM node:20 AS build
3
4 WORKDIR /app
5
6 COPY package.json package-lock.json ./
7 RUN npm install
8
9 COPY . .
10 RUN npm run build
11
12 # Step 2: Serve using Nginx
13 FROM nginx:alpine
14
15 COPY --from=build /app/dist /usr/share/nginx/html
16
17 EXPOSE 80
18
19 CMD ["nginx", "-g", "daemon off;"]
```

□ What this setup does

🔗 Build stage

- Uses Node only to build React app
- Output: `/dist` folder

🔗 Production stage

- Uses lightweight Nginx server
 - Serves only static files
-

□ Result (important)

Setup type	Size	Performance
------------	------	-------------

Dev (your current) ~400–500MB □ slower

Production (nginx) ~30–80MB □ fast

□ 2. Build production image

```
docker build -t vite-prod .
```

```
F:\falak-web>docker build -t vite-prod .  
[+] Building 147.1s (14/14) FINISHED  
=> [internal] load build definition from Dockerfile  
=> => transferring dockerfile: 345B
```

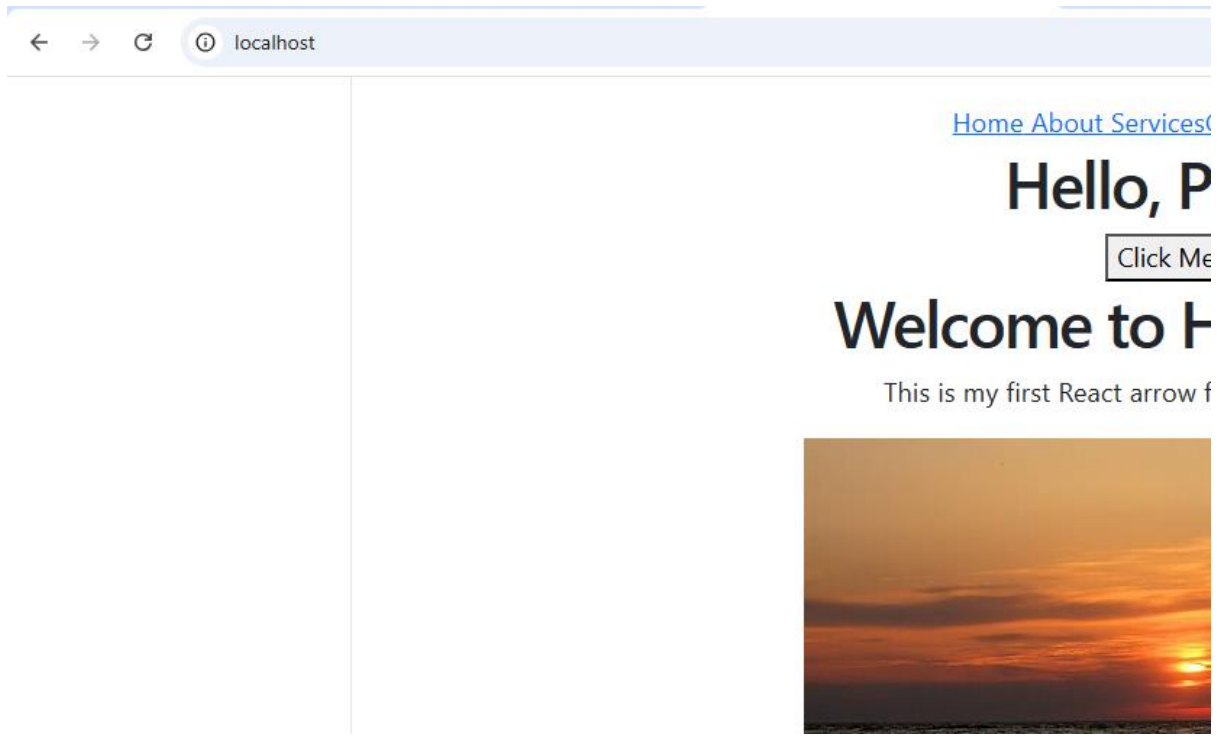
▶ □ 3. Run production container

```
docker run -p 80:80 vite-prod
```

```
F:\falak-web>docker run -p 80:80 vite-prod  
/docker-entrypoint.sh: /docker-entrypoint.d/ is not empty,  
/docker-entrypoint.sh: Looking for shell scripts in /docke  
/docker-entrypoint.sh: Launching /docker-entrypoint.d/10-l  
10-listen-on-ipv6-by-default.sh: info: Getting the checksum  
10-listen-on-ipv6-by-default.sh: info: Enabled listen on IP  
/docker-entrypoint.sh: Sourcing /docker-entrypoint.d/15-lo  
/docker-entrypoint.sh: Launching /docker-entrypoint.d/20-en  
/docker-entrypoint.sh: Launching /docker-entrypoint.d/30-t  
/docker-entrypoint.sh: Configuration complete; ready for st  
2026/04/14 12:17:03 [notice] 1#1: using the "epoll" event m  
2026/04/14 12:17:03 [notice] 1#1: nginx/1.29.8  
2026/04/14 12:17:03 [notice] 1#1: built by gcc 15.2.0 (Alp  
2026/04/14 12:17:03 [notice] 1#1: OS: Linux 6.6.87.2-micro  
2026/04/14 12:17:03 [notice] 1#1: getrlimit(RLIMIT_NOFILE)  
2026/04/14 12:17:03 [notice] 1#1: start worker processes  
2026/04/14 12:17:03 [notice] 1#1: start worker process 30  
2026/04/14 12:17:03 [notice] 1#1: start worker process 31  
2026/04/14 12:17:03 [notice] 1#1: start worker process 32  
2026/04/14 12:17:03 [notice] 1#1: start worker process 33
```

Open:

<http://localhost>



4. Why this is “production setup”

Because in real companies:

- React is NOT run with `npm run dev`
- Vite dev server is NOT exposed
- Only static files are served

They use:

- Nginx

5. Optional upgrades (real-world setup)

- ✓ Add caching headers
 - ✓ Enable gzip compression
 - ✓ Add HTTPS (reverse proxy)
 - ✓ Use Docker Compose with backend
-

Simple explanation

Your app becomes:

React + Vite → build → static files → Nginx → browser

Think like this

Thing	Meaning
Image	Blueprint (template) <input type="checkbox"/>
Container	Running app <input type="checkbox"/>

Example in your case

When you ran:

```
docker run -p 5173:5173 vite-react-app
```

Docker did this:

1. Took your **vite-react-app image**
2. Created a **container from it**
3. Started running your Vite app inside it

How to check container vs image

[?](#) See images:

```
docker images
```

```
C:\Users\User>docker images
IMAGE                ID                DISK USAGE    CONTENT SIZE    EXTRA
react-prod:latest    bdfbf38d5dff      1.85GB        456MB           U
vite-prod:latest     e9b922d4d9a1      93.6MB        26.3MB          U
vite-react-app:latest e6bf4266b1ee      1.92GB        478MB           U
```

See running containers:

docker ps

```
C:\Users\User>docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS        PORTS
efcff2c43cf1   vite-prod     "/docker-entrypoint..." 21 minutes ago Up 21 minutes 0.0.0.0:80->80/tcp,
mystifying_yalow
```

```
C:\Windows\system32\cmd.exe
Microsoft Windows [Version 10.0.19045.6466]
(c) Microsoft Corporation. All rights reserved.

C:\Users\User>docker images
IMAGE          ID              DISK USAGE  CONTENT SIZE  EXTRA
react-prod:latest   bdfbf38d5dff   1.85GB      456MB        U
vite-prod:latest    e9b922d4d9a1   93.6MB      26.3MB        U
vite-react-app:latest e6bf4266b1ee   1.92GB      478MB        U

C:\Users\User>docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS        PORTS
efcff2c43cf1   vite-prod     "/docker-entrypoint..." 21 minutes ago Up 21 minutes 0.0.0.0:80->80/tcp, [::]:80
```