

# Secure AI Dev Toolbox — User Guide

Reusable, network-restricted Docker environment with the aidev launcher

Version: 1.3 • Generated: August 30, 2025

## Published image

Use the prebuilt image: [docker.io/pdrittenhouse/secure-ai-dev:1](https://docker.io/pdrittenhouse/secure-ai-dev:1)

## Quick Start (use the published image)

```
mkdir -p "$HOME/.secure-ai-dev"
echo 'SECURE_AI_IMAGE="docker.io/pdrithenhouse/secure-ai-dev:1"' >> "$HOME/.secure-ai-dev/config" # install aidev (from your repo) and ensure allowlist
```

### VS Code devcontainers example:

```
{
  "image": "docker.io/pdrithenhouse/secure-ai-dev:1",
  "workspaceFolder": "/workspaces/app",
  "workspaceMount": "source=${localWorkspaceFolder},target=/workspaces/app,type=bind",
  "runArgs": ["--cap-add=NET_ADMIN","--cap-add=NET_RAW","--add-host=host.docker.internal:host-gateway"],
  "mounts": ["source=${env:HOME}/.secure-ai-dev/security/allowlist,target=/opt/security/allowlist,type=bind,readonly"],
  "remoteUser": "vscode",
  "postStartCommand": "sudo /opt/security/setup-firewall.sh"
}
```

## Build from Source (this repo)

Use these when you're developing the base image itself or want to test local changes before publishing.

### Local single-arch build

```
docker build -t pdrittenhouse/secure-ai-dev:dev .
echo 'SECURE_AI_IMAGE="pdrittenhouse/secure-ai-dev:dev"' >> "$HOME/.secure-ai-dev/config"
aidev rm 2>/dev/null || true
aidev && aidev doctor
```

### Multi-arch build & push (Docker Hub)

```
docker buildx create --use 2>/dev/null || true
docker buildx build --platform linux/amd64,linux/arm64 -t docker.io/pdrittenhouse/secure-ai-dev: 1 -t docker.io/<username>/secure-ai-dev:latest --push $HOME/.secure-ai-dev
```

### Load a specific arch locally (no push)

```
# Apple Silicon
docker buildx build --platform linux/arm64 -t pdrittenhouse/secure-ai-dev:dev --load .
# x86_64
docker buildx build --platform linux/amd64 -t pdrittenhouse/secure-ai-dev:dev --load .
```

# Using with VS Code Dev Containers

Pick one workflow per project: Attach to a running container (no repo config) or use a devcontainer.json that references the published image.

## Option A — Attach to running aidev container

```
# host
cd /your/project
aidev
# VS Code → Command Palette → Dev Containers: Attach to Running Container...
# choose: aidev-<folder>-<hash>
# in container terminal:
aidev doctor --verbose
```

## Option B — Use .devcontainer/devcontainer.json

```
{
  "image": "docker.io/pdrittenhouse/secure-ai-dev:1",
  "workspaceFolder": "/workspaces/app",
  "workspaceMount": "source=${localWorkspaceFolder},target=/workspaces/
app,type=bind",
  "runArgs": ["--cap-add=NET_ADMIN","--cap-add=NET_RAW","--add-
host=host.docker.internal:host-gateway"],  "mounts": ["source=${env:HOME}/.secure-
ai-dev/security/allowlist,target=/opt/security/allowlist,type=bind,readonly"],
  "remoteUser": "vscode",
  "postStartCommand": "sudo /opt/security/setup-firewall.sh",
  "shutdownAction": "stopContainer"
}
```

## Tips & Troubleshooting

- Firewall didn't load? Check the Dev Containers output panel or run `sudo /opt/security/setup-firewall.sh` inside the container.
- Changing image tag? Update `image` in devcontainer.json and choose Dev Containers → Rebuild Container.
- LocalWP/host services on Linux: use `host.docker.internal` (we add the host-gateway alias).
- Per-project allowlist lives at `./.allowed-domains.txt` on host; manage via `aidev domains add|remove|list|test`.

# Allowlists & Security Notes

Your host controls networking via allowlists. Global allowlist is mounted read-only at /opt/security/allowlist. Per-project file is ./allowed-domains.txt.

```
# Example global list
registry.npmjs.org
github.com
api.github.com
objects.githubusercontent.com
codeload.github.com
nodejs.org
api.openai.com
api.anthropic.com
host.docker.internal
# Project list
aidev domains add n8n.example.com
aidev domains test api.openai.com n8n.example.com
aidev reload
```

## `aidev` Reference & Tips

Command	What it does
<code>aidev [start]</code>	Start or reattach a secure container for the current folder.
<code>aidev sh</code>	Open a bash shell inside the container.
<code>aidev stop</code>	Stop the container for this folder.
<code>aidev rm</code>	Remove (force) the container for this folder.
<code>aidev reload</code>	Re-run <code>/opt/security/setup-firewall.sh</code> .
<code>aidev doctor [--verbose]</code>	Diagnostics; with <code>--verbose</code> , per-domain IP checks.
<code>aidev domains add/remove/list/test</code>	Manage per-project allowlist from the host.

# Git Hygiene & Release Flow

Commit the Dockerfile, security scripts, and the aidev launcher. Do not commit operator-specific allowlists.

```
.gitignore:  
/.allowed-domains.txt  
.DS_Store
```

## Release steps:

- 1) Update Dockerfile / security scripts
- 2) Build & push a new tag (e.g., :2)
- 3) Update docs to reference the new tag
- 4) Users: aidev rm && aidev (or Dev Containers → Rebuild)