Secure AI Dev Toolbox — User Guide

Reusable, network-restricted Docker environment with the aidev launcher

Version: 1.6 • Generated: September 19, 2025

Published image: docker.io/pdrittenhouse/secure-ai-dev:1

Quick Start (use the published image)

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

Use with VS Code — Attach to the running container

```
# host
cd /path/to/project
aidev

# VS Code
Press F1 (or Shift+Cmd+P / Ctrl+Shift+P) → Dev Containers: Attach to Running Container…
Pick: aidev-<folder>-<hash>
File → Open Folder… → /workspaces/app
```

If you accidentally open /home/vscode and don’t see your repo, use File → Open Folder… → /workspaces/app.

Alternative: Use 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"
```

Install Claude Code inside the container

The extension is a UI; install the Claude runtime in the same place VS Code runs (the dev container).

```
# In the attached container window
1) Extensions → Install "Claude Code" in Dev Container
2) F1 → Claude Code: Install ...
3) Verify: which claude   &&   claude --version
4) API key (container):
   export ANTHROPIC_API_KEY=sk-ant-...
   echo 'export ANTHROPIC_API_KEY=sk-ant-...' >> ~/.zshrc
```

Ensure `api.anthropic.com` is allow-listed on the host, then reload firewall:

```
# host
aidev domains add api.anthropic.com
aidev reload
aidev doctor --verbose
```

Troubleshooting

```
EACCES permission denied → run: npm config set prefix ~/.local
“Oh My Zsh can’t be loaded from: bash” → don’t source ~/.zshrc in bash
Fix ownership if needed:

  sudo chown -R "$(id -u)":"$(id -g)" ~/.npm ~/.config || true
  npm install -g @anthropic-ai/claude-code
```

Install & Use Claude Code inside the container

Install the extension *in the Dev Container*, then install the CLI into a per-user npm prefix.

Zsh

```
npm config set prefix ~/.local
npm install -g @anthropic-ai/claude-code
```

Bash

```
npm config set prefix ~/.local
source ~/.bashrc
hash -r
npm install -g @anthropic-ai/claude-code
```

API key & allowlist

```
# In container
export ANTHROPIC_API_KEY=sk-ant-...
# Persist (zsh):  echo 'export ANTHROPIC_API_KEY=sk-ant-...' >> ~/.zshrc
# Persist (bash): echo 'export ANTHROPIC_API_KEY=sk-ant-...' >> ~/.bashrc

# On host
aidev domains add api.anthropic.com
aidev reload
aidev doctor --verbose
```

aidev commands reference

Command	What it does
aidev [start]	Start or reattach a secure container for this folder.
aidev sh	Open a bash shell inside the container.
aidev stop	Stop the container.
aidev rm	Remove the container.
aidev reload	Re-apply firewall rules.
aidev doctor [--verbose]	Diagnostics; with --verbose, per-domain ACCEPT checks.
aidev domains add/remove/list/test	Manage per-project allowlist from the host.

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 login
docker buildx build --platform linux/amd64,linux/arm64   -t docker.io/pdrittenhouse/secure-ai-de
docker buildx imagetools inspect docker.io/pdrittenhouse/secure-ai-dev:1
```

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 .
```

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
```

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)
```