AI for Science – Operations Companion Guide (CLI‑First on Hetzner + Plesk)

Date: 2025-09-28

# Scope & What’s New

This companion guide complements the main “Deployment & Operations Guide”. It focuses on CLI‑first operations, day‑2 runbooks, troubleshooting, and safe maintenance. It assumes the app is deployed at:  
- UI → https://squeezecost.com/ai-for-science/  
- API → https://api.squeezecost.com/ai-for-science

Key additions here:  
- CLI workflow (no Portainer/Stacks management), advanced deploy.sh, and release flow  
- Health checks (local/public), smoke/load quick tests  
- Base vs override compose rationale, and flattening into docker-compose.final.yml  
- Plesk proxy patterns, slash-loop fixes, Next.js basePath notes  
- Safe cleanup (images/cache), backup/restore (volumes), disaster recovery  
- Portainer admin reset + IP allowlist + SSH tunnel  
- Optional systemd unit to auto-run compose on boot

# CLI‑First Workflow (no registry, loopback ports)

Folder: /opt/ai-for-science  
Compose: docker-compose.yml (base) + docker-compose.override.yml (loopback ports)

Start / redeploy (keeps volumes):  
 docker compose -f docker-compose.yml -f docker-compose.override.yml up -d --build

Stop:  
 docker compose -f docker-compose.yml -f docker-compose.override.yml down

Status & logs:  
 docker compose -f docker-compose.yml -f docker-compose.override.yml ps  
 docker compose -f docker-compose.yml -f docker-compose.override.yml logs -f backend  
 docker compose -f docker-compose.yml -f docker-compose.override.yml logs -f frontend

# Advanced deploy.sh (usage recap)

File: /opt/ai-for-science/deploy.sh (make executable: chmod +x deploy.sh)

Common examples:  
 ./deploy.sh up --build # start/redeploy all (build if needed)  
 ./deploy.sh build --no-cache frontend # force rebuild of frontend image  
 ./deploy.sh up frontend # restart only frontend  
 ./deploy.sh logs -f backend # follow backend logs  
 ./deploy.sh check all # health checks (local & public)  
 ./deploy.sh prune images # prune unused images  
Global -f (custom compose files):  
 ./deploy.sh -f docker-compose.yml -f docker-compose.override.yml up --build  
Environment variables:  
 DOMAIN=squeezecost.com BASE\_PATH=/ai-for-science  
 API\_PUBLIC=https://api.squeezecost.com/ai-for-science  
 FRONT\_PUBLIC=https://squeezecost.com/ai-for-science

# Base vs Override compose (and how to flatten)

Why two files:  
- Base is portable and safe (no public ports)  
- Override adds host-specific loopback ports for Plesk proxy

Flatten to one file (for sharing/debugging):  
 cd /opt/ai-for-science  
 docker compose -f docker-compose.yml -f docker-compose.override.yml config > docker-compose.final.yml

Run with the flattened file:  
 docker compose -f docker-compose.final.yml up -d

# Health checks & quick tests

Local (bypass Plesk):  
 curl -sS http://127.0.0.1:8001/api/v1/healthz  
 curl -I http://127.0.0.1:3001/ai-for-science/

Public (through Plesk TLS):  
 curl -sS https://api.squeezecost.com/ai-for-science/api/v1/healthz  
 curl -I https://squeezecost.com/ai-for-science/

Sample API call (reverse text demo):  
 curl -sS -X POST https://api.squeezecost.com/ai-for-science/api/v1/llm/generate -H 'Content-Type: application/json' -d '{"prompt":"hello world"}'

# Plesk nginx proxy rules (patterns to remember)

UI under subpath on main site:  
 location = /ai-for-science { return 301 /ai-for-science/; }  
 location ^~ /ai-for-science/ {  
 proxy\_pass http://127.0.0.1:3001;  
 proxy\_http\_version 1.1;  
 proxy\_set\_header Host $host;  
 proxy\_set\_header X-Forwarded-Proto $scheme;  
 proxy\_set\_header X-Forwarded-For $proxy\_add\_x\_forwarded\_for;  
 }

API under path prefix on subdomain:  
 location ^~ /ai-for-science/ {  
 proxy\_pass http://127.0.0.1:8001;  
 proxy\_http\_version 1.1;  
 proxy\_set\_header Host $host;  
 proxy\_set\_header X-Forwarded-Proto $scheme;  
 proxy\_set\_header X-Forwarded-For $proxy\_add\_x\_forwarded\_for;  
 }

# Next.js basePath & trailing-slash notes

- Build-time: NEXT\_PUBLIC\_\* envs are baked into the client bundle.  
- Ensure docker-compose.yml passes build args to frontend:  
 args:  
 BASE\_PATH: /ai-for-science  
 NEXT\_PUBLIC\_API\_URL: https://api.${DOMAIN}/ai-for-science  
- next.config.ts:  
 basePath: process.env.NEXT\_PUBLIC\_BASE\_PATH || "/ai-for-science"  
 trailingSlash: true # optional, stops slash loops  
- Symptom: 404 at /ai-for-science → likely not built with basePath → rebuild frontend --no-cache.  
- Symptom: 301 ↔ 308 loop → add exact Nginx redirect OR set trailingSlash: true and rebuild.

# Cleanup / Disk usage

What’s safe to prune:  
- Dangling images (no tag): docker image prune -f  
- All images not used by any container: docker image prune -a -f  
- Build cache: docker builder prune -f  
- Check disk usage: docker system df

Portainer UI also provides “Prune” for images/build cache from the browser.

# Backups & restore (volumes)

List project volumes:  
 docker volume ls | grep ai-for-science

Backup SQLite volume:  
 VOL=ai-for-science\_backend-data  
 docker run --rm -v $VOL:/v -v $(pwd):/b busybox sh -c 'cd /v && tar czf /b/backend-data.tgz .'

Restore:  
 docker run --rm -v $VOL:/v -v $(pwd):/b busybox sh -c 'cd /v && tar xzf /b/backend-data.tgz'

# Disaster recovery (quick checklist)

1) DNS → verify A records for squeezecost.com and api.squeezecost.com  
2) TLS in Plesk → issue/renew Let's Encrypt  
3) Recreate project:  
 git clone or restore /opt/ai-for-science  
4) Docker/Compose installed and running  
5) Restore volumes from backups (if needed)  
6) Start stack:  
 docker compose -f docker-compose.yml -f docker-compose.override.yml up -d --build  
7) Apply Plesk nginx proxy rules (UI+API as documented)  
8) Health checks: local & public

# Portainer (optional): login reset + IP allowlist + SSH tunnel

Reset admin password:  
 DATA=$(docker inspect plesk-portainer --format '{{range .Mounts}}{{if eq .Destination "/data"}}{{if .Name}}{{.Name}}{{else}}{{.Source}}{{end}}{{end}}{{end}}')  
 docker stop plesk-portainer  
 docker pull portainer/helper-reset-password  
 docker run --rm -v "$DATA":/data portainer/helper-reset-password  
 docker start plesk-portainer

Limit access by IP (Plesk → Additional Nginx directives on portainer.squeezecost.com):  
 allow 1.2.3.4;  
 deny all;  
 # Do NOT add a custom location for /.well-known/acme-challenge/ (Plesk already has one).

SSH tunnel instead of exposing:  
 ssh -N -L 9000:127.0.0.1:9000 root@<SERVER\_IP>  
 # then open http://localhost:9000

# systemd (optional): auto-redeploy on boot

File: /etc/systemd/system/ai-for-science.service

[Unit]  
Description=ai-for-science compose stack  
After=network-online.target docker.service  
Wants=network-online.target

[Service]  
Type=oneshot  
WorkingDirectory=/opt/ai-for-science  
RemainAfterExit=yes  
Environment=COMPOSE\_CMD=docker compose -f docker-compose.yml -f docker-compose.override.yml  
ExecStart=/bin/bash -lc "$COMPOSE\_CMD up -d"  
ExecStop=/bin/bash -lc "$COMPOSE\_CMD down"  
TimeoutStartSec=300

[Install]  
WantedBy=multi-user.target

Enable & test:  
 sudo systemctl daemon-reload  
 sudo systemctl enable ai-for-science  
 sudo systemctl start ai-for-science  
 sudo systemctl status ai-for-science

# Release process (suggested)

1) Code changes → commit to Git  
2) Backend only:  
 ./deploy.sh build backend  
 ./deploy.sh up backend  
3) Frontend (env/basePath changed):  
 ./deploy.sh build --no-cache frontend  
 ./deploy.sh up frontend  
4) Post‑deploy checks:  
 ./deploy.sh check all  
5) Prune unused images periodically:  
 ./deploy.sh prune images

# Where to store these docs

Keep guides under version control:  
- /opt/ai-for-science/docs/deployment-guide.docx  
- /opt/ai-for-science/docs/ops-companion.docx