

Ship & operate · Runbooks

Runbook: Rollback

🕒 3 min read · Updated 27 September 2026 · Verified against [d2f2e41](#) on 2026-09-27

Rolling production back: the website and the API functions together on Vercel, the API's standby on Render, one context or the whole API off the functions, the dataset, the app.

<https://docs.gurbanisoul.com/process/runbooks/rollback/>

Since 1.3.10 the website and the API are **one Vercel deployment**: the API runs as Python functions inside the web project ([ADR-0011](#)), and `gateway/routes.json` sends production's `/api` to them. So rolling the website back rolls the API back with it, in one step. The single API on Render keeps deploying every release as the standby (`deploy-api` and `verify-api` in `deploy-production.yml`) until it is retired ([moving production to functions](#), step 4).

The website and the API (Vercel) — the usual case

The pipeline rolls back by itself if the promote or the public smoke after it fails; a failure before promote (checkout, setup) leaves the previous deployment serving and rolls nothing back. An automatic rollback opens an issue naming the run and the deployment it went back to.

The rollback target is the deployment `gurbanisoul.com` resolves to when `deploy-web` starts (`scripts/ci/vercel_api.py live`, i.e. `GET /v13/deployments/gurbanisoul.com`), printed in that job's log as `previous production deployment`. It is never “the newest READY production deployment”: a build that was deployed unaliased and then failed its smoke is READY and `target=production` too, but was never promoted. If the lookup fails, or no READY deployment serves the domain, `deploy-web` stops before building and nothing is deployed.

Manually: `vercel rollback <previous-deployment-url> --yes` with that logged URL, or Vercel → Deployments → the deployment that last served the domain → **Instant Rollback**. Not sure which one that was? `VERCEL_TOKEN=... VERCEL_ORG_ID=... python3 scripts/ci/vercel_api.py live gurbanisoul.com` prints the one serving it now.

Then confirm: `https://gurbanisoul.com/api/health` returns `ok: true` and its commit is the rolled-back one, and the site's footer shows the same version. Close the issue the pipeline opened, and fix forward through a normal PR.

Taking the API off the functions

When the functions themselves are the problem in every recent deployment, move the traffic, not the deployment. Both are routing changes in `gateway/routes.json`, regenerated (`python3 tools/gen_gateway.py`) and released through the normal pipeline — they are not instant; `vercel rollback` is.

- **One context** (say `search`): remove it from `production.services`. The catch-all function `all` answers its routes again on the next release.
- **The whole API**: set `production.api_platform` back to `render` with its `api` origin. `/api` is then proxied to the Render standby. This is available until Render is retired.

The Render standby

If `verify-api` fails, the pipeline stops **before** the website is built and opens an incident; production is untouched, because production's `/api` is not served by Render. Roll the standby back so it stays a usable fallback: `Render` → the service → `Events` → the previous successful deploy → **Rollback**, then confirm its `/api/health` returns `ok: true` and the rolled-back commit.

Data (the dataset pin)

The database is pinned by `dataset.lock.json`. To revert a dataset, revert the lock bump (`git revert <commit>`): CI and the deploy install the previous object, sha256-verified, and nothing in `sggs-data` is ever deleted, so every previously pinned object stays available. Release through the normal pipeline.

iOS

In `Algorithmos-AI/gurbani-soul-ios` (its `ios-hotfix` runbook): `TestFlight` → expire the bad build; the App Store has no binary rollback, so pause the phased release and ship a fix.