THE GUINNESS PLATFORM · OPERATOR COMPANION

Infrastructure, Explained — what runs, where, and how to operate it.

Where *"How It Works"* explained what the platform is and why you are not locked in, this document explains the machine room: what is actually running, where it runs, and how a change becomes live. Written for the people who will operate the platform — no supplier jargon, no internal references.

● Verified against the running system · June 2026

IN ONE PARAGRAPH

The entire Guinness suite — a front-door shell plus ten applications and an in-house data engine — runs on **one dedicated server that Guinness owns.** On that server, an open-source control panel (**Coolify**) builds each app from an on-box **code vault** (Gitea) and runs it in its own container; a single **routing layer** (Traefik) terminates HTTPS and routes each Guinness web address to the right app. There is **no external dependency at runtime** — code, build and serving all live on this one machine — which is exactly what makes a clean handover to Guinness possible.

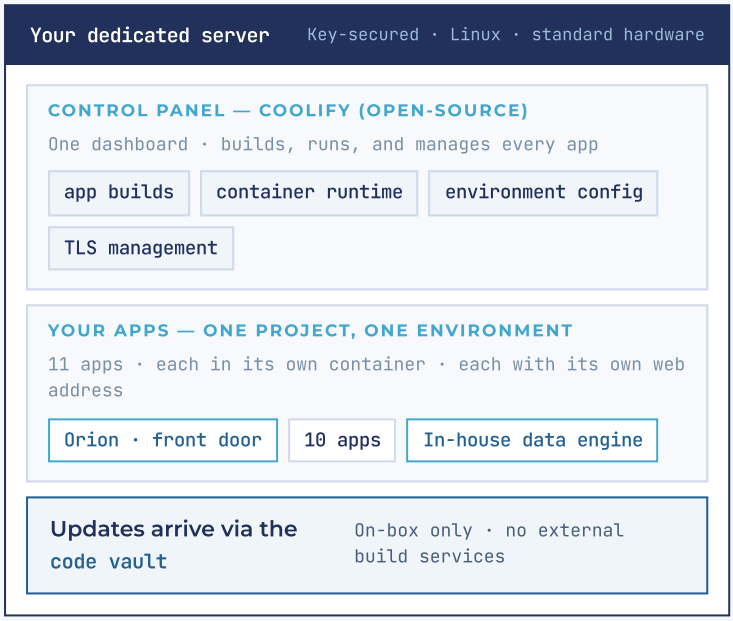
1	11	3	0
DEDICATED SERVER	GUINNESS APPS LIVE	OPEN-SOURCE LAYERS	EXTERNAL DEPS AT RUNTIME

01 THE BIG PICTURE

One owned machine — three layers, one front door

Hetzner + Coolify — one server, a self-hosted control panel

One **Hetzner Cloud** server in Germany runs the whole suite. Nothing on it is proprietary to the supplier: a stock Linux operating system running standard containers. On it, **Coolify** — an open-source, self-hosted platform-as-a-service — orchestrates everything via Docker: it builds each app from the on-box code vault and runs it as a container, managing settings, domains, and TLS. Coolify is to this deployment what a cloud platform is to a cloud-hosted app — but **self-hosted, so it belongs to Guinness.**



THE LAYERED SERVER — CONTROL PANEL, YOUR APPS, AND THE ON-BOX DEPLOY PATH. NO EXTERNAL BUILD SERVICE; NO OUTSIDE SYSTEM NEEDED TO DEPLOY.

VERIFIED JUNE 2026

- ✓ **All 11 apps up** — zero stopped containers
- ✓ **Plenty of headroom** — disk and memory mostly free
- ✓ Control panel and routing layer **healthy**
- ✓ Each app runs **independently** — one failure doesn't affect others

WHY SELF-HOSTED MATTERS

- ! **Coolify is open-source and standard.** If Guinness ever wishes to change supplier, the control panel moves with the box — there is no proprietary lock-in at the infrastructure layer.
- ! **Moving provider** is a routine server migration, not a rebuild.

DAY-TO-DAY OPERATIONS

- **One box, one view.** If you can reach the control panel, you can see and operate the entire estate.
- **Restart or redeploy an app** from the control panel without touching any other app.
- **Logs** are available per-app directly in the control panel.

Gitea & the ownership model — Guinness controls what deploys

All application source code and history lives on a **self-hosted code vault** (Gitea) running on the box — the blueprints, kept on-site. The model is **two lines per app:** a *supplier line* (a read-only mirror of what the supplier ships) and a *Guinness line* (what actually deploys). Supplier updates arrive on the mirror for visibility and are applied to the Guinness line deliberately — a **conscious decision, not an automatic event.**

VERIFIED JUNE 2026

- ✓ **11 app repositories** in the Guinness-owned code vault
- ✓ **Two-line pattern confirmed** across all supplier-originated apps
- ✓ The in-house data engine has **one line only** — born in the code vault, no supplier origin
- ✓ Regular sync running cleanly — **zero conflicts** across all repos
- ✓ What you read in the Guinness line is **exactly what is running** — no hidden artefacts

WHAT THIS MEANS FOR GUINNESS

- **You control the deploy switch.** The supplier can propose updates; only the Guinness line deploys — and adopting an update is a conscious choice.
- **To freeze production** (e.g. before a client demo): set the pause flag in the code vault; clear it to resume.
- **The Guinness line is the audit trail** — a complete history of every change that ever went live.
- **Handover** = remove one credential. The code vault and all its history stays exactly where it is, on the box.

The Orion shell & the apps behind it

guinnessgi.x-trillion.com is the front door — an **Orion shell** presenting a sidebar of apps, each loaded independently from its own Guinness web address. Eleven app containers run on the box, each deployed with its own address and certificate, so rebuilding or restarting one never affects any of the others or the shell itself.

App	What it is
Orion (shell)	The front door — the sidebar interface that hosts all apps.
Athena	Portfolio cockpit — NAV, yield, duration, holdings, P&L, and compliance checks.
Lexa	Sovereign credit research — deep country reports, served via API.
Iris	Interactive investment guide — the GGI AI Hub for staff and clients.
Jess	Narrated video briefings — market and fund commentary, with its own access control.
Jules	Compliance and brand audit — deck and factsheet review, slide generation.
Maia	Execution airlock — the bridge between Athena and order execution.
Iona	Wealthy Nations Bond Fund voice briefings — audio-first fund commentary.
Georgia	Sovereign ESG Scorer — live World Bank data, password-gated.
Brian	The in-shell help agent — context-aware guidance across all apps.
Data engine	Guinness's **in-house data engine** — sovereign data, Haver feeds, and the path to full in-house analytics (see §5).

The in-house data engine — independence running, not promised

The **first genuinely Guinness-owned service** — a lean engine that sits between the apps and their data. It **serves sovereign fundamentals natively** from licence-free public data (World Bank), and **Guinness's own Haver subscription** — a feed Guinness funds directly. For anything it cannot yet serve, it falls transparently to the interim engine, stamping every answer with its source. This is the **independence model in motion**: two data circuits already in-house, the rest available to bring across on Guinness's own timeline.

Your apps
ask the engine for data

• IN-HOUSE — WORLD BANK + HAVER (OWNED)

Sovereign fundamentals and macro time series are served directly from licence-free public data and Guinness's own Haver subscription. **No external analytics in the path. Grows over time.**

○ INTERIM — SUPPLIER FALLBACK (TRANSPARENT)

Anything the engine cannot yet serve natively is routed to the interim engine transparently — and stamped with its source so it is always clear which circuit answered. **Shrinks as more moves in-house.**

BORN IN THE CODE VAULT — NO SUPPLIER ORIGIN. THE EXTERNAL CONNECTION IS A SINGLE CONFIGURATION VALUE; REPOINT IT TO CHANGE THE FALLBACK WITH NOTHING ELSE TOUCHED.

● VERIFIED LIVE · IN-HOUSE PATH CONFIRMED

- ✓ **World Bank sovereign data live** — confirmed on real 2024 figures for multiple countries
- ✓ **Haver macro time series live** — fed from Guinness's own Haver subscription
- ✓ Every answer stamped with **its source** — you can always see which circuit responded
- ✓ Engine running on the box — **minimal footprint, zero restarts**

HONEST CURRENT STATE

- ! **In-house today:** sovereign macro data (World Bank) and macro time series (Haver). Both genuinely live on real numbers.
- ! **Coming in-house on Guinness's timeline:** bond analytics, pricing, and ratings — each moves across as the relevant licence or engine lands. No platform rebuild needed.

This is the "not locked in" story running: the supply of each data circuit is a configuration choice, not a code change.

THE OWNERSHIP MODEL IN PRACTICE

- **Born in the code vault.** This engine has no supplier origin — it is Guinness's own service, on Guinness's own box.
- **Re-sourcing, not recomputing.** When an in-house circuit is ready, it answers from the new source with the same number — books and reports are never affected.

06 ACCESS & SECURITY

Three access planes — each locked to its audience

Access is layered by audience. **Staff and clients** reach the apps through a single email one-time-code login — no per-app passwords to remember. **Operators** (Tom & Will) access the control panel and code vault through their own authenticated dashboards. The **server itself** is locked to authorised keys only — no password login, no guessable entry point.

Users

Email + one-time passcode, shared across all apps. A single sign-in gives access to every app in the suite. No per-app passwords.

Services

Apps communicate with each other and with data engines using signed tokens — no secrets visible to the browser, no internal addresses exposed.

Infrastructure

Key-only access — password login to the server is disabled. The control panel and code vault are behind their own HTTPS logins.

WHO LOGS IN WHERE

Who	Where	What they can do
All staff	Front door — `guinnessgi.x-trillion.com`	The apps — one email one-time-code sign-in; all most users ever need.
Ops (Tom & Will)	Control panel — `your Guinness control panel address`	App status, logs, restart, redeploy, environment settings.
Developers (Will)	Code vault — `your Guinness code vault address`	Source code, history, branch management, and when new versions go live.
Machines only	In-house data engine	No login — apps call it directly over the internal network; not exposed to browsers.

ACCESS CONFIRMED JUNE 2026

- ✓ All app logins operating via email one-time-code — **no passwords stored in the apps**
- ✓ Control panel and code vault: **login-gated with their own credentials**
- ✓ Machine-to-machine endpoints: **bearer-token gated** — a 401 is returned to anything without a valid token
- ✓ **Personal accounts for Tom & Will** ready once work email addresses are confirmed — each with a forced password change on first sign-in

Three switches — not three rebuilds

The platform was built with handover as a design goal, not an afterthought. The steps are **credential actions**, not migrations: nothing moves, nothing is rebuilt — access and control shift. **What Guinness already owns** is concrete and running today.

WHAT GUINNESS OWNS — CONCRETE, NOT ASPIRATIONAL

The dedicated server | Coolify · the control panel | Gitea · the code vault + all history | In-house data engine · Guinness-born code

What runs in production is, by construction, exactly the Guinness line in the code vault — no hidden build artefact, no supplier-side dependency.

1
Dedicated server project

Move the server into a project named for Guinness; grant Guinness's own account access to just that project. Clean billing and access boundary — the server is unambiguously yours.

2
Detach supplier sync

Remove one credential or set the pause flag — the code vault becomes a standalone source of truth. The supplier's systems are never exposed; the Guinness line keeps deploying exactly as before.

3
In-house data engine absorbs

Already serving sovereign data and Haver feeds in-house. Each additional data circuit moves across as the relevant licence or engine lands — on Guinness's timeline, not the supplier's.

THE BOTTOM LINE · VERIFIED JUNE 2026

- ✓ **No external dependency at runtime** — code, build and serving all live on one owned machine.
- ✓ Handover is a **credential action, not a migration** — the server, control panel, code vault and data engine are already Guinness's.
- ✓ The first two in-house data circuits (World Bank sovereign data · Haver macro series) are **live and verified**; the rest move in-house on Guinness's timeline.
- ✓ Personal operator accounts for Tom & Will **ready to activate** once email addresses are confirmed.

Prepared for Guinness Global Investors. This document describes the Guinness Platform infrastructure as verified against the running system in June 2026. It is intended for Tom and Will as the operator companion to the solution overview. Figures reflect a point-in-time snapshot and will evolve as the platform grows. The application source code, the code vault, and the server itself are Guinness-owned assets — not supplier infrastructure.