

TECHNICAL BLUEPRINT · FINAL

MP Ensystems Website Redesign Architecture Document

*The complete technical blueprint — frontend, backend, data, security, hosting,
DevOps, cost, and operations.*

Prepared for: MP Ensystems Advisory Pvt. Ltd.

Prepared by: Architecture/Engineering Lead (Claude)

Date: 11 September 2026

Status: Final — all Conflict Register items confirmed

Region: asia-south1 (Mumbai) · Load target: 1,000 DAU / 50 peak QPS

Table of Contents

1. Document Control & Version History.....	4
2. Executive Summary.....	5
3. Business Context & Objectives.....	6
4. Requirements Recap.....	7
Functional.....	7
Non-Functional.....	7
5. Conflict Register.....	8
6. Architecture Decision Records (ADRs).....	9
7. Recommended Architecture (High-Level).....	11
8. Alternative Platform Evaluation.....	12
9. Database Recommendation.....	13
Migration Note.....	13
10. Component Deep-Dive.....	14
Frontend.....	14
Admin CMS.....	14
Auth.....	14
Content Storage.....	14
Gated Download Flow.....	14
Chart of the Day Automation.....	14
Full-Text Search.....	14
Email Delivery.....	14
Video Hosting.....	14

11. Data Model.....	16
12. Security & Compliance.....	18
13. Observability, Cost & Operations.....	19
Monthly Cost Estimate (at 1,000 DAU / 50 peak QPS).....	19
Scale-to-Zero, Cold Starts, and Max Instances.....	19
Observability.....	19
Backup, Restore, and Disaster Recovery.....	20
14. Risks, Assumptions, and Open Questions.....	21
15. Appendix.....	22
Glossary.....	22
List of Mermaid Figures.....	22
Reference Documents.....	22

1. Document Control & Version History

Version	Date	Author	Change
v0.1-draft	11 Sep 2026	Architecture (Claude)	First draft — reconciled Input 1 (arch-doc.docx), Input 2 (mockups), Input 3 (PRD) with the Google Cloud migration context. 9 open items raised.
v0.2-draft	11 Sep 2026	Architecture (Claude)	Round 2 — all 7 Conflict Register items confirmed by stakeholders; ADR-02/03/05/09 accepted; Cloudflare's role clarified (DNS + firewall only); max-instances behaviour documented.
v1.0 (this document)	11 Sep 2026	Architecture (Claude)	Final — approved, converted to Word with rendered figures.

Field	Value
Owner	MP Ensystems Product/Engineering Lead
Review cadence	Re-review at the end of every architecture-affecting sprint, and mandatorily before any change to region, database engine, or CDN/edge layer.
Status	Approved. One external dependency remains open — the ClimaKosh/GRVC Dashboard API contract (Section 5, Conflict #6) — tracked as a placeholder, not a blocker.

2. Executive Summary

MP Ensystems is moving its website off Squarespace and onto Google Cloud, in India's Mumbai region (asia-south1), because the company's data-residency policy requires it. The new site is a single Next.js application — one codebase serving the public marketing site, a gated-download flow for project reports, and an internal Admin panel — all running in one container on Cloud Run.

Everything public is free to browse with no login. The only gate is a one-time registration form in front of downloadable project reports; the file is never shown on the page, only emailed. Two things — outbound email and embedded videos — are explicit, documented exceptions to the India-only hosting rule, because no compliant alternative exists for those two functions.

THE SINGLE MOST IMPORTANT DECISION IN THIS DOCUMENT

The original architecture assessment (Input 1) planned to store all website content as JSON files directly in Cloud Storage, with no database at all — MongoDB was only for admin login. The PRD's requirements for full-text search, taxonomy filtering, and a real Admin CRUD interface cannot be built well on flat JSON files at scale. This document's central, **now-confirmed** recommendation is a proper database and CMS — **Cloud SQL for PostgreSQL** in asia-south1, managed through **Payload CMS** — replacing the original "no database for content" plan.

3. Business Context & Objectives

MP Ensystems Advisory Pvt. Ltd. runs an advisory-plus-product business model: policy advisory, sectoral assessments, and knowledge-exchange work, delivered alongside its own software products (Platforms: ClimaKosh, EVCI Platform, GRVC Dashboard, and TRIM). The website represents both halves — advisory thought-leadership content, and the Platforms as usable tools — across three confirmed Sectors: Sustainable Power, GRVC (Green Rural Value Chains), and Mobility.

- **Free-access-first:** every public page is browsable with no login and no form.
- **Gated downloads only:** the one exception is project-specific report files, released after a one-time registration form, delivered by email.
- **India data residency:** application data, uploads, logs, and backups stay in India-region infrastructure, with two named, documented exceptions.
- **Advisory + product model:** the site showcases both consulting work and the software Platforms, with a daily-refreshing "Chart of the Day" tying the marketing site to the live Platforms.

4. Requirements Recap

Functional

- Public browsing of all pages, no login, no form.
- Registration-gated download flow for Project Story report files (Name, Email, Organisation, Consent -> email delivery, never inline).
- Full-text search across Project Stories and all Learning & Resources content, powered by per-item Taxonomy tags and Meta-data keywords.
- Admin CMS: full CRUD on all content types, plus registration-record management (view, export, delete).
- Chart of the Day: fully automated daily snapshot integrated with ClimaKosh and GRVC Dashboard - no manual Admin step.

Non-Functional

Category	Requirement
Data residency	India-based servers only (asia-south1/asia-south2) for app, DB, storage, CDN, and Admin auth. Documented exceptions: outbound email (SendGrid / Google Workspace SMTP relay) and video hosting (YouTube/Vimeo).
Performance	LCP (Largest Contentful Paint) < 2.5s. Load target: 1,000 DAU (Daily Active Users), 50 peak QPS (Queries Per Second).
Accessibility	WCAG (Web Content Accessibility Guidelines) 2.1 AA.
Privacy	GDPR (General Data Protection Regulation) and India's DPDP (Digital Personal Data Protection) Act - purpose-limited consent, documented erasure path.
Team skills	Next.js, Node.js/Express.js, MongoDB, React - the build stays inside this stack's learning radius.

5. Conflict Register

Every real tension found between Input 1 (arch-doc), Input 2 (mockups), Input 3 (PRD), and the new "migrate to Google Cloud / GCS container + CDN" business context. All seven items below are confirmed.

#	Topic	Source A	Source B	Resolution (Confirmed)
1	Compute/hosting model	Client's stated preference: "GCS container + CDN."	arch-doc: Next.js needs one container on Cloud Run - cannot be hosted as flat static files.	Confirmed. "GCS container" reads as shorthand for: GCS holds static assets/media/gated files, a container on Cloud Run runs the app, Cloud CDN sits in front of both.
2	CDN / edge layer	PRD: GCS + Cloud CDN pinned to asia-south1, fronted by Cloudflare (already in place).	This brief: Global External HTTPS Load Balancer + Cloud CDN (GCP-native).	Confirmed, with a refinement: Cloudflare is not dropped - it is used only for DNS management and firewall (WAF), with its caching bypassed. Path: Browser -> Cloudflare (DNS + firewall) -> Google Load Balancer -> Cloud CDN -> Cloud Run. See ADR-05.
3	Content persistence model	arch-doc: "Neither approach uses a database [for content]... JSON files... Cloud Storage."	PRD: full-text search, per-item Taxonomy/Meta-data, filterable Admin CRUD with draft/published status.	Confirmed. A proper database (Section 9) plus a CMS layer (Payload CMS, ADR-02) now handles content and metadata. GCS keeps its role for binary media and gated report files.
4	Auth adapter compatibility	arch-doc: NextAuth.js with a MongoDB adapter.	Conflict #3's resolution moves the database to PostgreSQL.	Confirmed. MongoDB adapter replaced by NextAuth.js's @auth/drizzle-adapter, pointed at the same Cloud SQL Postgres instance used for content - one database, not two.
5	CMS framework introduction	arch-doc: hand-built Next.js admin panel, no CMS framework named.	This brief: Payload CMS as "the likely recommendation"; PRD called it only a suggestion.	Accepted. Payload CMS provides the Admin CRUD surface, draft/publish workflow, and access-control scaffolding out of the box. See the Developer Notes section in the Dev/Hosting Guide for how the junior team should sequence learning this.
6	Chart of the Day integration contract	PRD: automated daily pull from ClimaKosh and GRVC Dashboard.	None of the three inputs define the API, auth method, or data format.	Approach accepted. Proceeds with an authenticated HTTPS/REST endpoint assumption, polled daily by a Cloud Function. A named placeholder (endpoint URL, auth method, response shape all marked TBD) is kept in config and code until the ClimaKosh/GRVC team supplies the real contract - an external dependency, not resolvable by this review alone.
7	Region naming	arch-doc: names asia-south1 (Mumbai) specifically.	PRD/business context: allows asia-south1 or asia-south2.	Confirmed. asia-south1 (Mumbai) is the single primary region for app, database, and storage. asia-south2 (Delhi) is reserved only as an optional disaster-recovery target (Section 13).

NOT A CONFLICT - CONFIRMED ALIGNMENT

Auth mechanism: arch-doc's NextAuth.js + server-verified session/JWT approach matches the PRD's earlier confirmed decision (Q4) to use session/JWT-based auth instead of Firebase Authentication. Both documents independently rule out Firebase - no conflict.

6. Architecture Decision Records (ADRs)

One short, self-contained record per major decision, so a future reader can see what was chosen and why.

ADR-01 — Frontend Framework **ACCEPTED**

Context: Need one framework for the public site, download flow, and Admin panel, buildable by a 1-2 year team.

Options: Plain HTML/CSS/JS + Express; plain HTML/CSS/JS + Cloud Functions; Next.js (single app).

Decision: Next.js (App Router). One codebase, one container, server-enforced draft filtering by default.

Consequences: Small learning curve if unfamiliar with Next.js; requires a Node.js server (Cloud Run).

ADR-02 — Admin CMS **ACCEPTED**

Context: PRD requires full CRUD across nine content types, draft/publish toggling, Taxonomy/Meta-data on every item.

Options: Hand-built Next.js admin panel (arch-doc's original plan); Payload CMS; Strapi/Directus.

Decision: Payload CMS - TypeScript-native, installs inside the same Next.js app (v3), native Postgres support via Drizzle, built-in draft/publish and access-control hooks.

Consequences: New dependency for the junior team to learn (sequenced in the Dev Guide's Developer Notes); removes roughly half of the custom admin-panel build work in arch-doc.

ADR-03 — Database **ACCEPTED**

Context: Content, taxonomy, metadata, registrations, and admin users all need a queryable, India-resident store with full-text search.

Options: See Section 9 - Cloud SQL Postgres, Cloud SQL MySQL, AlloyDB, Firestore, SQLite+Litestream, MongoDB Atlas.

Decision: Cloud SQL for PostgreSQL, asia-south1.

Consequences: Replaces the MongoDB Atlas plan from arch-doc; one database serves auth, content, and registrations; native full-text search via tsvector.

ADR-04 — Hosting / Compute **ACCEPTED**

Context: Need a managed, scale-to-zero-capable runtime for one Next.js container in asia-south1.

Options: Cloud Run; Compute Engine VM; App Engine Standard.

Decision: Cloud Run, single service, region-pinned to asia-south1. min-instances: 1 to avoid cold starts on the registration flow.

Consequences: Small always-on cost even at zero traffic, in exchange for consistent registration-form latency.

ADR-05 — CDN / Edge **ACCEPTED**

Context: Need caching and TLS termination in front of Cloud Run and GCS, while keeping Cloudflare's existing DNS management and firewall coverage.

Options: Keep Cloudflare doing everything; drop Cloudflare entirely; split responsibilities.

Decision: Split: Cloudflare stays for DNS + firewall/WAF only (proxy on, caching bypassed). Cloud CDN + Global External HTTPS Load Balancer does all content caching and TLS termination.

Consequences: No loss of Cloudflare's firewall protection; avoids two competing cache layers. Requires a Cloudflare Cache Rule set to "bypass cache" for the site's hostname.

ADR-06 — Authentication **ACCEPTED (ADAPTER CHANGED)**

Context: Admin-only login, no public sign-up, must not depend on a non-India-pinnable service.

Options: Firebase Authentication (ruled out); hand-rolled auth; NextAuth.js (Auth.js).

Decision: NextAuth.js v5, Credentials provider, @auth/drizzle-adapter against Cloud SQL Postgres, httpOnly signed session cookie.

Consequences: Well-tested library handles CSRF protection and session cookies; every write still re-verified server-side inside a Server Action.

ADR-07 — Email Delivery **ACCEPTED**

Context: Gated report files must be emailed, never served inline; India-residency mandate applies except two named exceptions.

Options: SendGrid; Google Workspace SMTP relay; an India-based transactional provider.

Decision: SendGrid or Google Workspace SMTP relay - confirmed documented exception to the residency mandate (PRD Q5).

Consequences: Recorded explicitly in the site's privacy policy as a named, deliberate exception.

ADR-08 — Video Hosting **ACCEPTED**

Context: Learning & Resources videos need a hosting/embedding approach; self-hosting large video in GCS is expensive.

Options: Self-host in GCS + Cloud CDN; YouTube/Vimeo embed.

Decision: YouTube/Vimeo, privacy-enhanced/no-cookie embed mode - confirmed documented exception (PRD Q9).

Consequences: Video delivery sits outside India; mitigated by the privacy-enhanced embed domain.

ADR-09 — Full-Text Search **ACCEPTED**

Context: Search must cover Project Stories and all Learning & Resources content, ranked against title, description, taxonomy, metadata.

Options: MongoDB Atlas Search; Postgres native (tsvector + GIN index, pg_trgm); a dedicated search service.

Decision: Postgres native full-text search - no extra service to run, stays inside the India-region database.

Consequences: Less relevance-tuning power than a dedicated search service; sufficient at this content scale.

7. Recommended Architecture (High-Level)

The whole system at a glance, then a walkthrough of one real request.

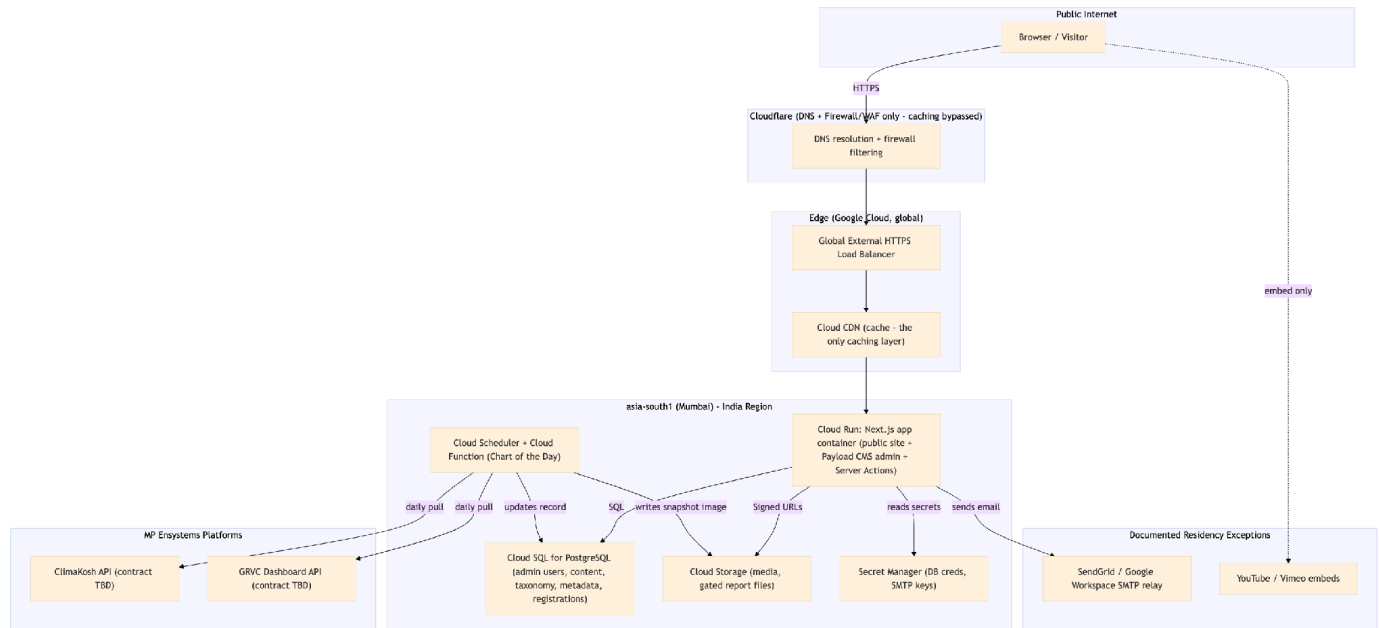


Figure 1. High-Level Container Architecture

Cloudflare only resolves DNS and filters traffic; Cloud CDN is the sole caching layer; one Cloud Run container serves everything dynamic; Postgres holds everything queryable; GCS holds binary files; two named exceptions sit outside the India boundary by explicit agreement.

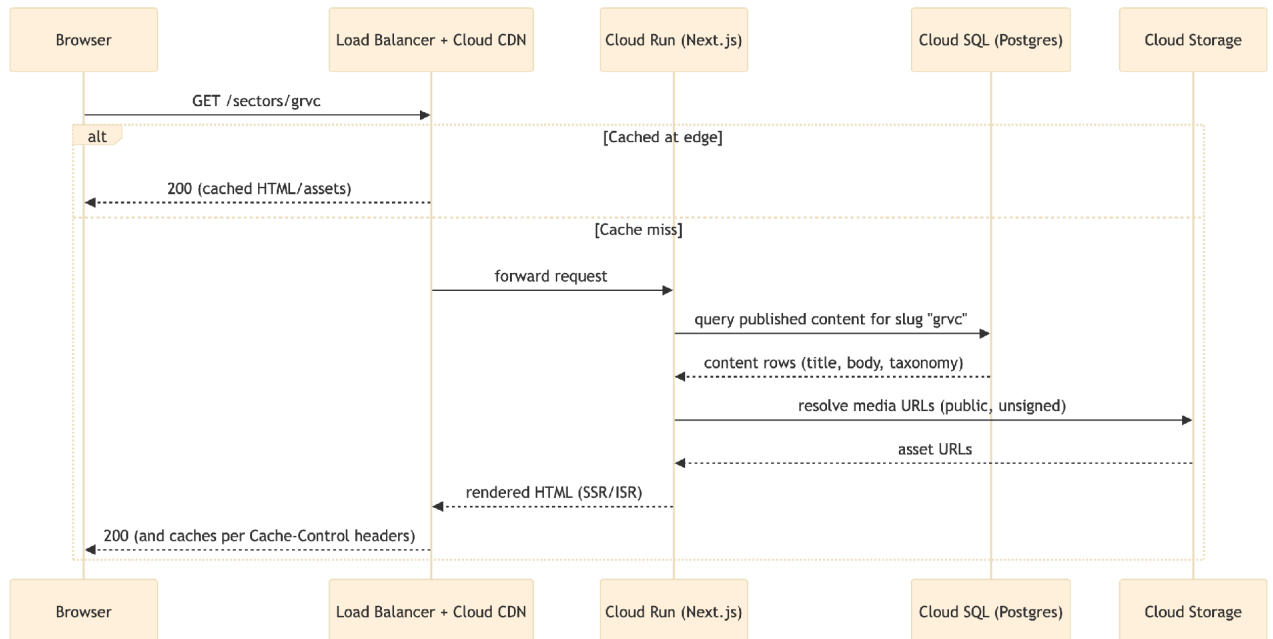


Figure 2. Request Flow: Browser to Response

Most public traffic is served from the CDN cache; only cache misses and dynamic actions (search, registration, admin) reach Cloud Run and the database.

8. Alternative Platform Evaluation

At least three end-to-end alternatives, compared against cost, operational ease, learning curve, scalability, India-residency fit, and existing team-skill fit.

Option	Cost	Ops Ease	Learning Curve	Scalability	India Residency	Team Fit
1. Next.js + Payload CMS on Cloud Run + Cloud SQL Postgres + GCS + Cloud CDN (RECOMMENDED)	Low-Med	Low burden	Low	High	Full fit	High
2. Next.js + Payload CMS on Cloud Run + MongoDB Atlas (Mumbai) + GCS + Cloud CDN	Med	Med - separate vendor	Low	High	Full fit	High
3. Astro (SSG) + lightweight CMS + Cloud Functions + Firestore/GCS	Very low	Low static / weak dynamic	Med	High static, weak dynamic	Fit	Low
4. Next.js on Firebase App Hosting + Firestore	Low	Very low burden	Low	High	FAILS - no hard India-only guarantee	Med

Recommendation: Option 1. It satisfies India residency with the same certainty as Option 2, matches the team's Next.js/React/Node skills exactly like Option 2, and additionally resolves the Section 5 Conflict #3 problem (search and filtered admin listings) with a database designed for exactly that, at a lower steady-state cost than an Atlas M10 tier. Option 2 remains a safe fallback if the team prefers to stay on MongoDB.

9. Database Recommendation

Option	India Region?	Monthly Cost (est.)	Learning Curve	Ops Burden	Search Fit	Payload Support	Verdict
Cloud SQL for PostgreSQL	asia-south1	\$25-55	Low	Low	Strong (tsvector+GIN)	Native	RECOMMENDED
Cloud SQL for MySQL	asia-south1	\$25-55	Low	Low	Weaker (FULLTEXT)	Supported, less common	Viable, not preferred
AlloyDB for PostgreSQL	asia-south1	Higher, no micro tier	Low	Low	Strong + faster analytics	Native	Cost-justified only at ~10x scale
Firestore	asia-south1	Low	Med	Low	Poor - no native FTS	Not supported	Rejected - content is relational
SQLite + Litestream (Cloud Run)	wherever pinned	Near-zero	Low/Med	High - fragile under real traffic	Fair (FTS5)	Not supported	Rejected for production
MongoDB Atlas (Mumbai M0/M10)	Mumbai	M10 ~\$60+	Lowest	Low, separate vendor	Good (Atlas Search, paid add-on)	Native	Solid fallback (Option 2)

Migration Note

The team is not currently running MongoDB in production for content - arch-doc's plan only ever put **admin user accounts** in MongoDB, with all content as JSON files in GCS. So this is not a "migrate an existing content database" project; it is a "stand up Postgres and import the JSON files as rows" project. **Recommended path: write a one-time Node.js migration script that reads every JSON file out of the GCS content bucket and inserts it into the new Postgres schema (Section 11) via Payload's local API, preserving each item's published/draft status.** Admin user accounts move the same way, into the admin_users table, via a one-time seed script - not a live-traffic cutover, since the site has not launched yet. See the Development & Hosting Guide's Developer Notes section for the full step-by-step walkthrough and pitfalls specific to this task.

10. Component Deep-Dive

Frontend

Next.js App Router. Public pages use ISR (Incremental Static Regeneration) - pre-rendered at build time, then silently re-rendered in the background (e.g. every 60 seconds) or on-demand when an Admin publishes a change. The registration form, search, and Admin panel use Server Actions for anything that writes data.

Admin CMS

Payload CMS, mounted inside the same Next.js app at /admin. Collections map directly to the PRD's content types. Every collection gets Payload's built-in draft/publish versioning, so "hide unpublished content from the public" is enforced by the CMS layer itself, not a manually-written filter that a developer could forget.

Auth

NextAuth.js v5, Credentials provider (email + bcrypt-hashed password), Drizzle/Postgres adapter. Admin accounts are added by a one-time seed script or an internal setup route - no public sign-up. Every write-capable Server Action re-checks the session server-side.

Content Storage

Two stores, two jobs: Cloud SQL Postgres holds everything that needs to be queried, filtered, searched, or listed. Cloud Storage holds binary files fetched by URL, not queried - images, video thumbnails, the daily Chart-of-the-Day image, and gated report PDFs.

Gated Download Flow

A visitor clicking Download first hits a Server Action that checks for an "already registered" cookie. If absent, the registration form collects Name, Email, Organisation, Consent, written to the registrations table. The Server Action generates a short-lived GCS signed URL (recommended 48h expiry) and emails it - the file is never served directly on the site.

Chart of the Day Automation

A Cloud Scheduler job triggers a Cloud Function once daily (recommended 06:00 IST). The function calls the ClimaKosh and GRVC Dashboard APIs (contract TBD - Conflict #6), renders the chart image, uploads it to a fixed GCS path, and updates a Postgres row recording the last successful run. On failure, the previous day's image stays live - the homepage never shows a broken image, only a stale one.

Full-Text Search

Every content row carries a generated tsvector column combining title, body, and metadata keywords, indexed with a GIN (Generalized Inverted Index). Search queries rank results with Postgres's ts_rank, filterable by content type and Taxonomy tag.

Email Delivery

SendGrid or Google Workspace SMTP relay (documented exception, Section 5 Conflict #2/#5 treatment). The Server Action that sends the gated-content email logs a delivery status (sent/failed) back onto the registration row.

Video Hosting

YouTube/Vimeo, embedded using their privacy-enhanced ("no-cookie") domains. The Admin CMS stores only the video ID/URL and thumbnail reference - no video file ever touches GCS.

11. Data Model

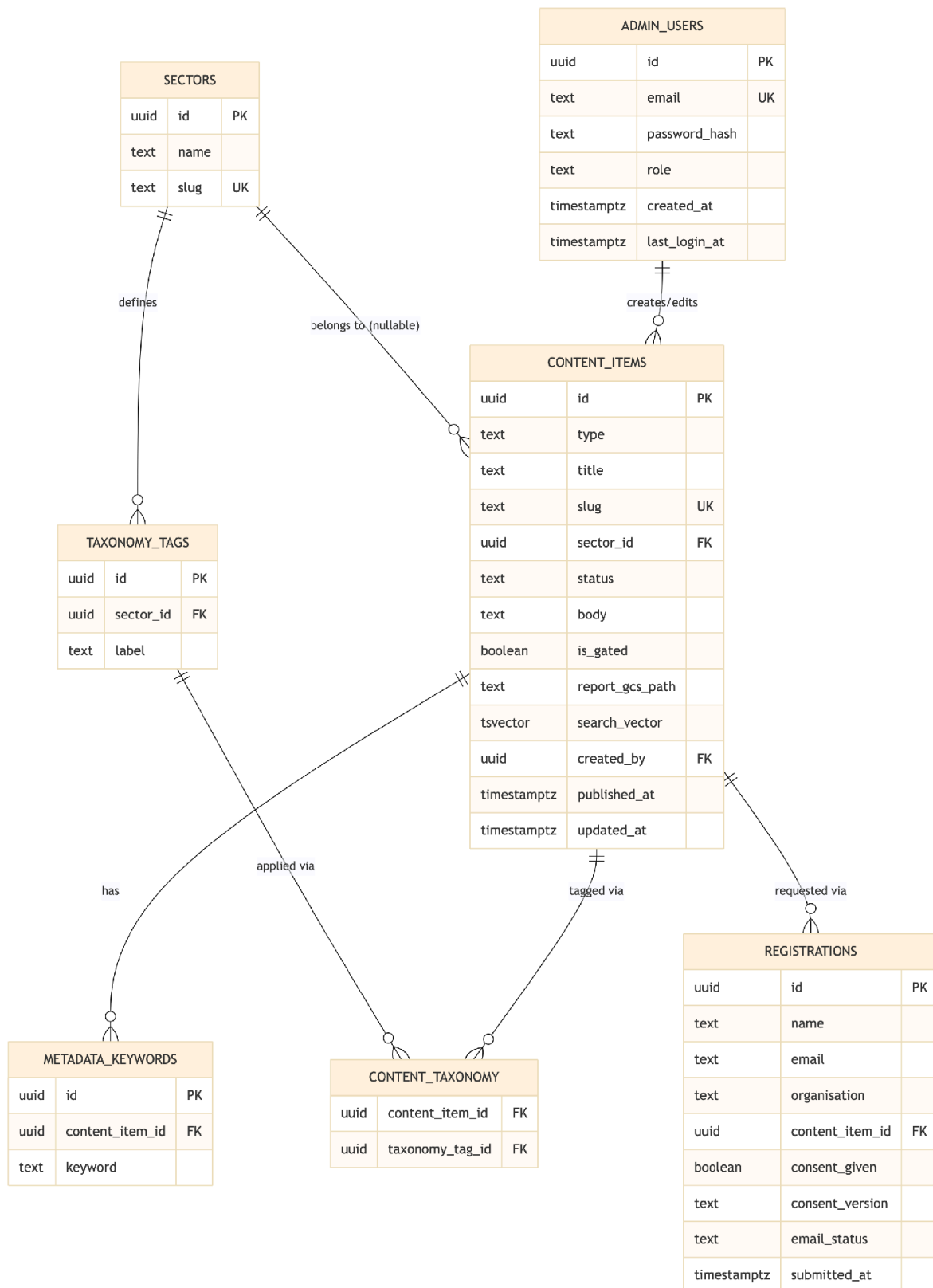


Figure 3. Entity-Relationship Diagram

One relational schema serves admin auth, content, taxonomy/search, and the registration log - the single Postgres database from ADR-03.

Notes: content_items.status is draft or published (Payload's versioning system manages the transition).
content_items.is_gated plus report_gcs_path together drive the download-gate logic in Section 10.
registrations.email_status is pending, sent, or failed, matching the Email Delivery component above.

12. Security & Compliance

- **Consent capture:** the registration form's consent checkbox text and timestamp are stored per-submission (`consent_version`), so the exact wording the user agreed to is always auditable.
- **Erasure path:** Admin's "delete registration record" action performs a hard delete on the registrations row - the documented mechanism for honouring a DPDP/GDPR erasure request.
- **Secret handling:** database credentials, SMTP API keys, and NextAuth's session secret all live in Secret Manager, injected into Cloud Run as environment variables at deploy time - never committed to the repository.
- **Signed URLs:** gated report files are served only via time-limited GCS signed URLs, never a public bucket path.
- **CSRF:** handled by NextAuth.js's built-in protection on all authenticated Server Actions.
- **Rate limiting:** the registration endpoint gets a per-IP rate limit (recommended: 5 submissions per 10 minutes) to blunt automated abuse of the email-sending flow.
- **IAM least privilege:** the Cloud Run service account gets only the specific roles it needs (Cloud SQL Client, Storage Object Admin on specific buckets, Secret Manager Secret Accessor) - not project-level Editor/Owner.
- **Residency statement:** application data, uploads, logs, and backups are India-resident (`asia-south1`). Two named, documented exceptions: outbound transactional email and embedded video - both confirmed by stakeholders and recorded in the site's privacy policy.

13. Observability, Cost & Operations

Monthly Cost Estimate (at 1,000 DAU / 50 peak QPS)

Component	Estimate (US\$/mo)	Notes
Cloud Run	15-40	1 min-instance to avoid cold starts, scales up under load.
Cloud SQL (Postgres, shared-core)	25-55	Includes automated daily backups.
Cloud Storage	3-10	Media + gated reports; low egress since Cloud CDN absorbs repeat requests.
Cloud CDN + Load Balancer	20-35	Fixed LB fee plus cache fill/egress.
Cloud Functions + Scheduler	<2	One run per day, trivial compute.
Secret Manager, Logging, Monitoring	<5	Free tier covers most of this at this traffic level.
SendGrid (email, exception)	0-20	Free tier (100/day) may suffice initially.
Total (estimate)	\$65-165/month	Excludes India GST; reverify with the GCP pricing calculator before committing budget.

Scale-to-Zero, Cold Starts, and Max Instances

Cloud Run can scale to zero, but the registration flow and Admin panel both need consistent latency, so this architecture sets min-instances: 1 (a small, predictable cost) to keep one instance always warm. max-instances is capped (recommended: 10) as a cost safety net against runaway traffic or a misbehaving script.

WHAT HAPPENS WHEN THE MAX-INSTANCES CEILING (10) IS ACTUALLY HIT?

The site does not go down. Two things protect it before Cloud Run's ceiling is even reached:

1. Most public traffic never reaches Cloud Run at all - it is served straight from the Cloud CDN cache (Section 7, Figure 2). Only cache misses, search queries, registrations, and Admin actions reach the container fleet.
2. Each instance can handle multiple requests at once, controlled by `--concurrency` (recommended: 80). The real ceiling on simultaneous work is max-instances x concurrency - with recommended settings, roughly 800 concurrent requests, not just 10.

Only once that combined ceiling is exceeded does Cloud Run start to queue incoming requests, waiting for an existing instance to free up. If the queue wait exceeds the request timeout, the client gets an HTTP 429 (Too Many Requests) or 503 (Service Unavailable) response - a graceful, visible error, not a crash. Everything already running keeps serving normally throughout.

How to bring it back up: raise the ceiling - no redeploy, no downtime:

```
gcloud run services update mpensystems-web-prod \
  --region=asia-south1 --max-instances=20
```

Cloud Run picks up the new ceiling immediately and starts spinning up additional instances to absorb the queued and new requests - typically within seconds. The proactive version of this fix is a Cloud Monitoring alert (Dev Guide Part 13.6) on container instance count approaching the current max, so the team raises the ceiling before visitors see any errors. One downstream effect to watch: more Cloud Run instances means more simultaneous connections to Postgres - check Cloud SQL's connection limit isn't the next bottleneck (Dev Guide Troubleshooting, item 7).

Observability

Cloud Logging captures every request and Server Action error automatically. Cloud Monitoring gets an uptime check against the homepage every 5 minutes, alerting the team by email if it fails from more than one region. A budget alert fires at 50%, 90%, and 100% of a set monthly ceiling.

Backup, Restore, and Disaster Recovery

Cloud SQL automated daily backups, retained 7 days, with point-in-time recovery enabled. GCS buckets use Object Versioning so an accidental overwrite/delete can be rolled back. asia-south2 (Delhi) is reserved as an optional DR target - not provisioned by default, but documented as the fallback region if asia-south1 has a prolonged outage.

14. Risks, Assumptions, and Open Questions

Type	Item	Impact
Assumption (external dependency)	ClimaKosh and GRVC Dashboard expose an authenticated HTTPS/REST API suitable for a daily automated pull. Approach confirmed; exact contract not yet available and carried as a named placeholder (Dev Guide Part 6.2).	High if the real integration is something else entirely; low if it is indeed an HTTPS/REST API.
Confirmed	"GCS container + CDN" means GCS-for-assets + Cloud-Run-for-compute - confirmed by stakeholders.	None.
Resolved	Cloudflare's firewall/DDoS protection is retained - only its caching role is removed.	None - no change to existing firewall coverage.
Resolved	Behaviour at the Cloud Run max-instances ceiling was unclear.	None - explained in Section 13: requests queue and the ceiling can be raised live with no downtime.
Risk	Payload CMS (ADR-02) is a new dependency for a 1-2 year team with no prior Payload experience.	Medium - mitigated by Payload's documentation and the Dev Guide's Developer Notes section; budget extra ramp-up time in the first sprint.
Open question	Exact SLA/notification requirements if a Chart of the Day pull fails for more than one consecutive day.	Low - currently just logs and keeps the stale image; escalation policy undefined.
Open question	Whether Admin CMS needs more than one role (e.g. Editor vs. full Admin) - PRD only defines a single Admin User role.	Low - adding roles later is a small Payload access-control change, not a rearchitecture.

15. Appendix

Glossary

Term	Definition
ADR	Architecture Decision Record - a short document recording one architectural choice and why it was made.
CDN	Content Delivery Network - a network of edge servers that cache and serve content closer to the visitor.
CMS	Content Management System - software for creating, editing, and publishing website content without writing code.
CRUD	Create, Read, Update, Delete - the four basic operations on stored data.
CSRF	Cross-Site Request Forgery - an attack that tricks a logged-in user's browser into submitting an unwanted request.
DAU	Daily Active Users.
DPDP	Digital Personal Data Protection Act - India's 2023 personal-data-protection law.
GCS	Google Cloud Storage - Google Cloud's object storage service.
GDPR	General Data Protection Regulation - the EU's data-protection law.
GIN	Generalized Inverted Index - a Postgres index type well-suited to full-text search.
IAM	Identity and Access Management - controls who/what can access which cloud resources.
ISR	Incremental Static Regeneration - Next.js's pattern for re-generating static pages in the background after publishing.
JWT	JSON Web Token - a signed, compact token format often used for session/auth state.
LCP	Largest Contentful Paint - a Core Web Vitals metric measuring perceived load speed.
QPS	Queries Per Second.
SLA	Service Level Agreement - a commitment to a level of uptime/performance.
SMTP	Simple Mail Transfer Protocol - the standard protocol for sending email.
SSG	Static Site Generation - pre-building pages to plain HTML at build time.
SSR	Server-Side Rendering - generating a page's HTML on the server per-request.
WCAG	Web Content Accessibility Guidelines.

List of Mermaid Figures

Figure	Caption	Appears In
Figure 1	High-Level Container Architecture	This document, Section 7
Figure 2	Request Flow: Browser to Response	This document, Section 7
Figure 3	Entity-Relationship Diagram	This document, Section 11
Figure 4	Gated Download Flow (Implementation Detail)	Development & Hosting Guide, Part 4
Figure 5	Chart of the Day Automation Flow	Development & Hosting Guide, Part 6
Figure 6	CI/CD Pipeline	Development & Hosting Guide, Part 9
Figure 7	Deployment Topology	Development & Hosting Guide, Part 11

All 7 figures' Mermaid source code is in the companion file `MPEnsystems_Mermaid_Diagrams.txt`.

Reference Documents

- Input 1 - `arch-doc.docx` (existing architecture assessment)
- Input 2 - `MPEnsystems_UIUX_Mockups.html` (Figures 7-12, brand palette)
- Input 3 - `MPEnsystems_PRD_UIUX_Final.docx` (confirmed PRD + UI/UX Guidelines)