harper

Hyper-efficient composable application platform that ships an integrated database, cache, and message queue in the same executable as your business logic. Deploys across edge or core infrastructure and speaks standard protocols (REST, GraphQL, SQL, WebSockets) out of the box.

Typical Architecture

- Application and data services run in a single system with an embedded ACID - compliant database, cache, and queue
- Down to sub-millisecond, in-process operations — no network hops
- Deployable anywhere, with native geo - distributed clustering for scale and resilience
- No external services like Postgres, Redis, or Kafka needed

Ideal Uses

- Enterprises or scale-ups with large catalogs, global traffic, or interactive workloads (search, personalization, streaming, and a strong desire for low latency
- Teams demanding consistent high-performance and predictable spend across regions

Core Offering

Fused compute + storage + cache + messaging engine deployed to any location. No external middleware, no queuing service, and no ORMs are required—just one highly optimized binary that can be clustered for true edge performance.

HEROKU

Runs applications in isolated Linux containers (“dynos”) backed by AWS. It offers a marketplace of add-ons (Postgres, Redis, Kafka, etc.) to extend functionality over network connections, which slows the system down, and relies on AWS regions for availability.

- Application dyno(s) run your business logic
- Heroku Postgres add-on acts as the database
- Heroku Redis (KV) add-on provides caching
- Each service lives in its own container and communicates over the network, introducing extra latency and operational overhead

- Start-ups or teams that value “git push heroku main” convenience over raw performance
- Burst-y or moderate workloads that can live with dyno sleep/autoscaling latency
- Apps that tolerate an ephemeral file system and additive data-service costs

PaaS for web & worker dynos; build-pipeline, autoscaling, log drains; à-la-carte Postgres, Redis, Kafka, and hundreds of marketplace integrations.

Harper — The Better Enterprise Solution.

Predictable, Cost-Effective Pricing

Heroku's layered pricing—dyno hours, add-ons, and per-feature billing—makes budgeting difficult at scale. Harper offers transparent pricing based on estimated resource requirements per workload, without surprise charges for things like autoscaling, storage, or log drains.

Better Support

Instead of stitching Postgres + Redis + Kafka + CDN add-ons together, Harper bundles ACID storage, real-time cache, durable queueing, and CDN-style data locality in one engine - simplifying architecture and operations.

Unmatched Performance & Flexibility at Massive Scale

Heroku apps share hardware and rely on network calls between dynos and external data stores, introducing latency and stressing budgets at scale. Harper's fused stack eliminates those hops, delivers sub-millisecond in-process reads/writes, and can be deployed across clouds or on-prem for regulatory needs that Heroku's limited region set cannot meet. Harper offers full control over deployment and infrastructure with support for custom runtimes, networking, and system configuration—things Heroku restricts behind layers of abstraction or premium plans.

Full-Stack Solution — No Middleware or Separate Data Systems

Heroku's tiered support depends on your plan and the SLAs of individual add-ons, meaning longer resolution times and more complexity when things break. In contrast, Harper's enterprise-first model includes premium support with every managed deployment and its fused architecture eliminates the need for external services. With fewer moving parts and no finger-pointing between dynos or add-ons, issues are resolved faster and more effectively.

When performance and cost matter, Harper's fused-stack outperforms Heroku's multi-service architecture by orders of magnitude — at a fraction of the price

harper