ALLIED BizTech SOLUTIONS

FIELD GUIDE · 2026

# Rebuild Algolia yourself.
# *The plan we follow.*

For teams paying $500 to $5,000+ a month on Algolia who want to own their index, autocomplete, facets and ranking, and stop paying per record and per search.

---

- → Is it worth it, at your size
- → What to rebuild and what to drop
- → The architecture, in three layers
- → Migration and go-live checklists
- → The index model
- → A week-by-week build plan
- → Relevance, typos, index sync
- → Where teams get stuck

# Algolia bills per record and search. *Meilisearch bills neither.*

Algolia bills on three axes: indexed records, search operations and tier features. Each grows with your catalog and traffic. Meilisearch is open source; a rebuild replaces all three with the server that hosts it.

| WORKLOAD | ALGOLIA A YEAR | REBUILT: YEAR ONE / AFTER | SAVED IN YEAR ONE |
|---|---|---|---|
| **100K records** · 200K searches a month | ~$3,000 | ~$12,200 / $600 | Stay on Algolia |
| **500K records** · 1M searches a month | ~$12,000 | ~$13,000 / $1,400 | Pays back in 14 months |
| **1M records** · 2M searches a month | ~$24,000 | ~$13,400 / $1,800 | ~$10,600 |
| **3M records** · 5M a month · Premium | ~$48,000+ | ~$21,800 / $3,500 | ~$26,200 |
| **10M records** · 20M a month · Enterprise | ~$120,000+ | ~$26,300 / $8,000 | ~$93,700 |

Algolia list prices, May 2026. "Rebuilt" runs Meilisearch on a Hetzner or DigitalOcean VPS or AWS EC2, sized for the workload, plus our fixed build price: $11,600 below 3M records, $18,300 (the top of the $14,900–$18,300 scale tier) from 3M. If your own team builds it, put your engineering cost in that place.

THE RULE OF THUMB

At 100K records, stay on Algolia: the build does not pay back. At 500K it pays back in 14 months. **At 1M records, the rebuild pays back in 7 months, then ~$22K a year stays with you: ~$55K saved over three years.** From 3M records, year one alone saves more than $25K.

## Who you need

- **A senior backend engineer** at home with search engines, your source database, webhooks and Linux servers. Our team ships this in 3–4 weeks; one engineer working between other priorities should plan for longer.
- **Your product or merchandising lead**, two hours a week: top queries, ranking rules, the pilot locale, the go-live call.
- **An owner after launch**: upgrades, backups, re-indexing, the monthly relevance review, and on-call.

# Rebuild the 14 features your users trigger. *Drop the rest.*

Algolia's dashboard ships hundreds of features. Your search experience uses about 14. Those are the project, at parity your users can feel; the long tail is where rebuilds stall.

### REBUILD ONE FOR ONE

- **Instant search and autocomplete**: sub-50ms responses, the same UX
- **Typo tolerance**, with a threshold per attribute
- **Faceted search and filters**: price ranges, categories, multi-select facets, filter chips
- **Custom ranking**: popularity, recency, boost in-stock, demote unavailable
- **Synonyms and alternative spellings**, per attribute
- **Geo-search**: radius, polygon, sort by distance
- **Multi-index federation**: products, content and categories in one query
- **Multi-language tokenisation** for the 1–5 languages you serve
- **Click and conversion tracking**, in your own database

### REBUILD, SIMPLER

- **Personalisation**: per-user ranking from click and conversion history, rule-based rather than Insights ML scoring
- **A/B testing**: a harness on your own index, or GrowthBook or Statsig
- **Semantic search**: vector embeddings via OpenAI, Voyage or Cohere, billed to you directly
- **Synonyms and rules admin**: CLI and JSON config in Git, not a web console

### DROP ON PURPOSE

- **Recommend**: keep Algolia for that slice, or build on embeddings later
- **Visual and image search**: a dedicated tool such as Syte
- **DocSearch and add-ons nobody uses**
- **Multi-region replication**: two instances and a sync cover most needs

### ADD WHAT HOSTED SEARCH HIDES

- Query access from any service in your VPC, AI agents included
- Per-query cost reporting; no record-count tier jumps

# Three layers. *Same UX your users know.*

Your front-end, most likely React or Vue InstantSearch, keeps working: Meilisearch ships an InstantSearch-compatible adapter, so the components do not change. The index, the indexer and the runtime move to your infrastructure.

LAYER 1 · INDEX

### Meilisearch

- Open source, MIT licensed
- One binary, ~250 MB RAM at 1M records
- Typo tolerance, facets, filters, ranking
- Multi-language tokenisation
- A $40-a-month VPS or your AWS

LAYER 2 · INDEXER

### The sync pipeline

- Postgres, Shopify, Contentful or custom, into Meilisearch
- Webhook-driven incremental updates
- An idempotent full-rebuild job
- Schema changes with reindex orchestration
- Health checks and rollback

LAYER 3 · UI

### Front-end and analytics

- InstantSearch React or Vue, no rewrite
- Or a custom autocomplete and faceted UI
- Search analytics in your database
- A ranking A/B harness
- Latency monitoring per query

## Decisions to make in week one

| DECISION | OUR DEFAULT | WHY |
|---|---|---|
| Where it runs | A Hetzner or DigitalOcean VPS, or your AWS | p95 of ~25–40ms at 1M records on a $40 VPS |
| Engine | Meilisearch; OpenSearch or Typesense past 10M records and 50M searches a month | Same architecture, sized to your shape |
| Front-end | The InstantSearch adapter | A config swap, not a rewrite |
| Sync | Webhooks for changes, a full rebuild on a schedule | Fresh results, and a clean index every time |
| Regions | One; 2–3 regional instances and a router for global users | About $20 a month per extra region |
| Fallback | Postgres full-text search | Search stays up in a catastrophic failure |

Licence: Meilisearch is MIT licensed, so every version released stays free whatever the company does. If you ever need to move, Typesense and OpenSearch take the same architecture with a different binary.

# Your database stays the truth. *The index is a copy you can rebuild.*

Model these before the first import. Use the source record's primary key as the document id, the same value Algolia held as `objectID`, so every sync is an upsert and any hit traces back to its row.

| OBJECT | KEY FIELDS |
|---|---|
| `documents` (per index) | id (source primary key), title, description, brand, category path, price, in_stock, popularity, updated_at, locale, `_geo` |
| `index settings` · Git | searchable attributes in priority order, filterable and sortable attributes, ranking rules, typo tolerance per attribute, stop words, distinct attribute |
| `synonyms` · Git | locale, words, type (two-way, one-way), source (imported from Algolia, added by merchandising) |
| `query_rules` | id, pattern, action (pin, hide, boost, filter), objects, active_from, active_to, owner |
| `sync_events` | id, entity, entity_id, op (upsert, delete), version, created_at, processed_at, attempts, error |
| `sync_jobs` | id, kind (incremental, full), index_uid, started_at, finished_at, docs_upserted, docs_deleted, status |
| `search_events` | id, query, index, filters, hits_count, latency_ms, session_id, user_id, created_at |
| `click_events` | search_event_id, document_id, position, converted, value, created_at |
| `parity_cases` | query, filters, Algolia top 10, Meilisearch top 10, overlap, status, checked_at |

TWO RULES THAT SAVE A REBUILD

**Build a full index under a new name, then swap it in**: users never see a half-built index, and rollback is one swap back. **Log every search with its latency, hits and clicks**: zero-result queries tell you which synonyms to add, and the log proves the rebuild matches what Algolia did.

# Four weeks, *pilot included.*

This is the order we build in. Each week ends with something your product lead can search, and the last week is a pilot on a single locale or category before the full cutover.

| WEEK | BUILD | DONE WHEN |
|---|---|---|
| 1 | **Export and translate.** Export your Algolia indexes, settings, ranking, synonyms, rules and analytics history through its API. Meilisearch up on your VPS or AWS; settings translated and kept in Git. The parity test written: your top 100 production queries against both engines. | Document counts match Algolia; the parity test runs |
| 2 | **The indexer.** The full-rebuild job from your source, webhook-driven incremental updates through a retrying queue, deletes and out-of-stock handled, reindex on schema change, health checks. | A change in your source shows in search within seconds |
| 3 | **Front-end and relevance.** InstantSearch pointed at the adapter, or the custom calls adapted. Ranking, typo and synonym tuning until parity is green. Search analytics, latency monitoring, the Postgres fallback, a load test. | The parity test passes on every one of the 100 queries |
| 4 | **Pilot and cutover.** One locale or category on the new search for a week, a daily fix list, then every locale switched over, with the old config one deploy away. | The pilot's conversion and zero-result rate hold or improve |

AFTER CUTOVER

Keep Algolia **live but idle until your renewal date**, then cancel. Rollback stays one config change away while the team settles in, and nobody is in a hurry.

# Search is judged on the first three results. *Prove them.*

Standing up Meilisearch takes an afternoon. The real work is relevance that matches what your users already trust, typo tolerance that helps rather than guesses, and an index that never shows a product you no longer sell.

1. **Prove parity on real queries.** Take your top 100 production queries from Algolia's analytics, run each against both engines and compare the top results. Fix the gaps one by one; cut over only when every query is green.
2. **Translate ranking in order.** Algolia's ranking criteria and custom ranking map to Meilisearch's ordered ranking rules plus sort attributes such as popularity and recency. Put business rules (boost in-stock, demote unavailable) into the data as attributes, so they rank the same everywhere.
3. **Tune typos per attribute.** Typo tolerance helps on product names and hurts on SKUs, model numbers and short brand names. Switch it off on those attributes and words, and set the minimum word length for one and two typos.
4. **Treat synonyms as content.** Import Algolia's synonyms, then add the ones your zero-result log asks for, per language, with stop words for each of the 1–5 languages you serve.
5. **Keep the index in sync.** Send every change through a queue with retries and idempotent upserts. Deletes are where indexes drift: index sold-out and removed items as changes too, and run the full rebuild on a schedule into a fresh index, then swap.

## Measure what users feel

Track p95 latency, the zero-result rate, click-through on the top three results and sync lag, week by week. Those four numbers prove parity after cutover and show where relevance needs work.

## Plan for the bad day

Snapshots on a schedule, a restore you have tested, graceful restarts, and Postgres full-text search as the fallback. Run that way, a properly deployed VPS matches Algolia's 99.95% SLA.

# Move it, launch it, *run it.*

## Migration

- ☐ Export indexes, settings, ranking, synonyms, rules and analytics history
- ☐ Keep Algolia's `objectID` as the document id
- ☐ Map searchable, filterable and sortable attributes
- ☐ Turn Algolia's sort replicas into sort settings
- ☐ Move synonyms and stop words, per language
- ☐ Recreate query rules your merchandisers still use
- ☐ Run the parity test on your top 100 queries
- ☐ Check document counts per index and locale

## Go-live

- ☐ InstantSearch on the adapter in every site and app
- ☐ Search-only keys in the browser; the admin key server-side
- ☐ Webhooks live, with an alert on sync lag
- ☐ Snapshots proven by a restore
- ☐ Alerts on latency, errors and zero-result spikes
- ☐ Rollback ready: the Algolia config one deploy away

## Running it

- ☐ **Always on:** health checks, latency and sync-lag alerts, security patches
- ☐ **Weekly:** top and zero-result queries to the product lead, new synonyms
- ☐ **Monthly:** Meilisearch upgrades through staging first, a ranking and re-tuning review
- ☐ **Quarterly:** cost review, the server bill against the Algolia bill you replaced
- ☐ **Yearly:** a restore drill, a key rotation and an access review

### Who owns it

Name one person before launch. Search sits in front of every sale: someone has to answer when results go stale at 2 a.m. or a catalog import breaks the ranking.

# Five ways a rebuild *stalls.*

### 1. Switching to another hosted vendor

Typesense Cloud is cheaper than Algolia, with the same hosted lock-in. Klevu and Searchspring wrap Elasticsearch at comparable prices, sometimes higher at scale. Coveo runs 2–3× Algolia's price at the same volume. Each swaps one wrapper for another, plus a migration.

### 2. Postgres full-text search for customers

It is free and fine for low-traffic admin tools. In front of customers, typo tolerance is poor, faceting is slow and autocomplete lags, and your conversion rate drops with them.

### 3. Self-hosting Elasticsearch

JVM tuning, mapping evolution, cluster upgrades and snapshot and restore make it complex to run. Most teams underestimate the ongoing cost by 5×. Meilisearch sidesteps most of that work.

### 4. Rebuilding everything

Recommend, Insights ML scoring and visual search are separate products. Ship the 14 features in daily use first; add the semantic or recommendations tier when the numbers ask for it.

### 5. Cutting over before parity is green

The top 100 queries carry most of your revenue. Cut over while some still differ and your users notice before you do.

### And one that is not a mistake: negotiating

Vendr and Tropic typically negotiate Algolia annual contracts down 12–22%. Do it if the rebuild is a year away; it captures about an eighth of what self-hosting does.

RATHER HAVE IT BUILT?

# We build this exact plan. *Fixed price.*

The same scope, architecture and four weeks, delivered by a senior team that has shipped 250+ products.

---

**$11,600 fixed**: a senior engineer, a coder and a half-time delivery manager for 3½ weeks; $14,900–$18,300 for 3M+ records on a multi-shard cluster; agreed before we start

---

**Live in 3–4 weeks**, with a one-week pilot on a single locale or category first

---

**Your front-end keeps working**: InstantSearch on the adapter; ranking, synonyms and filters migrated

---

**90-day warranty**, and the code is in your GitHub from day one

---

**We can run it** for $1,200 a year: Meilisearch updates, index re-tuning and monitoring

**alliedbiz.tech/rebuild/algolia-alternative**

Allied BizTech · 15+ years · ISO 9001:2015 · 250+ products shipped. Algolia is a trademark of its owner, named here for identification only; this guide is independent and not endorsed by Algolia. Prices as of May 2026.