Architecture Review — Critical Questions

Event Finder — Architecture Review: Critical Questions

Strategy & Scope

1) What is the real target scale: dozens, hundreds, or thousands of events, and how frequently does the catalog refresh?

2) Who owns the ingestion sources and what legal/usage constraints apply to each source?

3) Are events local-only or multi-region? If multi-region, do we need locale/region awareness (time zones, languages, date formats)?

4) What is the expected user journey: quick lookup or browsing/discovery? Any mobile-first constraints?

Data Quality & Governance

5) How do we handle duplicates across sources, conflicting fields (e.g., venue names), and missing data (time, location)?

6) Do we need canonicalization rules (e.g., normalize dates, locations, and organizer names)?

7) Should we track data lineage to know which source produced each field?

Search & Relevance

8) What are the matching expectations (exact match, stemming, synonyms, typo tolerance)?

9) Do we need ranking signals beyond text match (recency, popularity, distance, organizer reputation)?

10) Should we support structured filters (date range, location radius, price, category) in the near term?

Experience & Content

11) Do we need result cards with fielded data or is the narrative response the primary format? Both?

12) Should the UI suggest related queries or clarifications when zero results are found?

13) Accessibility targets (keyboard navigation, screen readers, contrast)?

Operations & Reliability

14) What uptime/latency SLOs matter (if any)? Is this strictly best-effort or user-facing with expectations?

15) Do we require persistence across restarts now, or is ephemeral acceptable until phase 2?

16) What’s the update mechanism for ingestion: manual trigger, scheduled job, or webhook-driven?

17) Do we need observability basics now (request logs, ingestion metrics, error tracking)?

Security & Compliance

18) Will this be exposed publicly? If yes, when do we add TLS, rate limiting, and abuse controls?

19) Any PII or sensitive fields planned now or later (e.g., RSVP emails)?

20) What is the threat model for data poisoning from untrusted sources?

Roadmap & Governance

21) Who prioritizes the roadmap (search quality vs. ingestion coverage vs. UI polish)?

22) What are success metrics (conversion to event page, time to first useful result, query success rate)?

23) Do we need an admin view for curation, audit logs, and content takedowns?

Event Finder — High-Level Architecture (Revised)

Event Finder — High-Level Architecture (Revised)

1) Overview & Objectives

A pragmatic event discovery service that ingests event listings from approved sources, normalizes them, provides fast search, and returns clear summaries in a simple web UI. Priorities: correctness, clarity, and low operational overhead.

2) Capabilities

• Ingestion: Collect and normalize events from configured sources (HTML, feeds, files).

• Catalog: Maintain a clean, deduplicated, structured store of events.

• Search & Relevance: Text search with sensible ordering (recency-first), optional structured filters.

• Response Generation: Human-friendly narratives and optional card-style results.

• Presentation: Minimal, responsive web interface with a search box and results.

3) Context (Conceptual Flow)

Sources → Ingestion → Normalization/Deduplication → Catalog → Search & Relevance → Response → Web UI

4) Logical Components

• Source Connectors: One per source type (HTML/feeds/files); encapsulate parsing and error handling.

• Normalizer/Deduper: Standardizes fields (name, date/time, location), resolves duplicates via heuristics.

• Catalog Service: Owns the event model; supports upsert, list, and read for search.

• Search Service: Provides keyword search and optional filters (date range, location, category).

• Response Service: Formats results as narratives and/or structured cards.

• Web UI: Search input, loading state, results; zero-login, mobile-friendly.

• Admin (Future): Manual review, source health, re-ingestion triggers.

5) Data Model (Business-Level)

Event

- Core: name, date, time, location (venue, city/region), description

- Optional (future): category/tags, organizer, price, URL, image, geo-coordinates, source id
- Metadata: source, ingestion timestamp, normalization flags

6) Key Cross-Cutting Concerns

• Data Quality: Normalization rules, duplicate detection, missing-field fallbacks.

• Relevance: Recency bias by default; optional distance/categorization in phase 2.

• Observability: Basic logs for ingestion and search; error summaries.

• Security & Legal: Use only approved sources; do not expose admin/debug endpoints publicly.

• Privacy: Avoid PII unless explicitly required and governed.

• Accessibility: Basic a11y (labels, semantic markup, keyboard navigation).

7) Interaction Scenarios

Search (Happy Path)

1. User enters a query in the UI.
2. UI calls Search Service with query and optional filters.
3. Search returns candidate events from the Catalog.
4. Response Service formats results as a short narrative (+ optional cards).
5. UI renders results and suggests refinements if needed.

Ingestion/Refresh

1. Admin or schedule initiates ingestion for configured sources.
2. Connectors fetch data; Normalizer/Deduper standardizes and merges.
3. Catalog is updated with upserts; Search index reflects updates.
4. Logs and summaries indicate freshness and failures.

8) Non-Functional Goals (Target)

• Simplicity: Minimal moving parts; single small host acceptable.

• Performance: Snappy results for small-to-medium catalogs (sub-second typical).

• Resilience: Graceful behavior with missing or malformed source data.

• Portability: Able to run locally and in a small cloud instance.

• Extensibility: Pluggable connectors, fields, and filters.

9) Operational View

• Environments: Local dev, optional small cloud.

• Configuration: Source list, refresh cadence, feature flags (filters on/off).

• Deployment: App server with static UI; no public debug endpoints.

• Monitoring: Request logs, ingestion counts, error rates; simple dashboards later.

• Runbooks: “Search is slow” and “Source is failing” quick checks.

10) Risks & Mitigations

• Source Volatility: Formats drift — isolate parsers and add validation.

• Data Drift/Noise: Duplicates, inconsistent fields — normalization & merge heuristics.

• Exposure Risk: Public debug or open endpoints — disable debug, add basic protections before public launch.

• Relevance Gaps: Exact-match brittleness — plan synonyms/typo tolerance roadmap.

• Scalability Ceiling: Growth beyond a single host — plan step-up (persistent store, external search).

11) Roadmap

Phase 0 (Now): MVP with keyword search, narrative responses, seed data, minimal UI.

Phase 1: Persistent catalog, basic filters (date range, city/region), admin trigger for refresh.

Phase 2: Relevance tuning (recency, synonyms), multi-source connectors, basic metrics dashboard.

Phase 3: Geo-awareness, richer cards (images, URLs), rate limiting, TLS, and basic auth for admin.

12) Success Metrics

• Query Success Rate (non-empty useful results)

• Time to First Result (p50/p90)

• Data Freshness (time since last successful ingestion)

• Zero-Result Recovery (suggestions clicked)

• Operational Health (ingestion error rate)