Extraction Service - PRD

Creator **Michael Clark** Created **Aug 24, 2025, 20:23** Last updated **Aug 24, 2025, 22:19**

Unstructured → Structured Extraction Service

Project Manager	Team members	Date
Michael Clark	Michael Clark	8/24/2025

Summary

We are building a scalable service to convert unstructured documents (PDFs, markdown, white papers) into structured JSONL metadata. The system ingests large batches (up to 100k docs), applies configurable extraction schemas, and delivers normalized outputs with strong reliability, observability, and compliance. Success is measured by throughput, latency, accuracy, and reliability, with delivery in phased rollouts from local filesystem MVP to full multi-tenant GA with connectors and dashboards.

Problem Statement

Organizations have thousands of unstructured documents (PDFs, markdown, white papers, reports) spread across local servers and cloud stores. Manually extracting metadata like authors, publish dates, abstracts, or code snippets is slow, inconsistent, and error-prone. Existing tools either don't scale to **100k documents** or lack observability, retries, and compliance features. We need a reliable service that can **ingest large batches, extract configurable metadata, and produce normalized structured output** while providing full observability and tenant-level isolation.

Objectives

- □ Ingest large batches of unstructured documents (PDF, docx, markdown, etc)
- □ Ingest from filesystem, S3, and Sharepoint
- □ Extract metadata fields specified by job configuration
- □ Support configurable schemas for each job with optional hints (regex, date formats, section markers, etc).
- □ Output normalized JSONL for document extracts
- □ Multi-tenant
- □ Metrics**:** p95, p99, job count per user, job count per tenant, overall job count
- □ Logs**:** Job status, processing time, errors, retries
- □ Throughput knobs: backpressure, rate limits, per tenant quotas
- □ Provide Job Results API for downloading job results.
- □ Secure first: Encryption at rest and over the wire, RBAC, Tenant specific API keys

Must have	□ **Batch reading** of files (max 100k) from any connector (FS, S3, Sharepoint, etc) □ **Job API** for specifying configuration (file source, metadata fields, patterns/hints) □ **Budget estimation pre-checks** (reject or warn if job > threshold cost). □ **Quota enforcement**: max docs/job, max jobs/tenant, daily/weekly limits. □ **Results API** for returning paginated JSONL results, status, and job metrics □ **Cost metrics:** $ per page, per tenant, per connector; exposed in metrics. □ **Tenant API** for creating and configuring new customers □ **Configurable rate limits** (per tenant). □ **Fail-fast policies**: job stops early if budget exceeded. □ API Key Generation □ **Health API** for polling health metrics (p95, p99) □ Per tenant, per job, and overall system. □ **TLS everywhere** (in-transit encryption for APIs, worker comms, storage). □ **Tenant isolation** (per-tenant data separation in storage + metadata). □ **Authentication & authorization** (API tokens or OAuth2; RBAC for multi-tenant use). □ **Secret management** (no creds in env/logs; use Vault/KMS/Secrets Manager). □ **At-rest encryption** for output and job metadata. □ **Access logging** for PII fields (who accessed what, when). □ **Per-job audit logs**: who submitted, config used, files processed, timestamps. □ **Traceability**: every record links back to input doc + job ID. □ **Versioning**: schema version + extractor version logged with each run. □ **Immutable logs**: append-only storage or WORM compliance for audit trails. □ **Metrics:** processed count, error count, retries, throughput, latency (p50, p95, p99). □ **Logs:** structured JSON logs per job with correlation IDs. □ **Tracing:** spans from API → worker → connector → extractor; correlation by job ID. □ **Dashboards & alerts:** prebuilt for error spikes, latency breaches, cost anomalies. □ **Run history:** persisted for trend analysis and debugging (success/fail per job).
Nice to have	□ **Streaming Results API** □ Per Job limit and cost configs □ PII Config per job for automatic redaction □ **Configurable redaction or masking** for sensitive fields. □ **PII tagging in schema/config** (so system knows what needs special handling). □ **Right to delete** — ability to purge outputs/logs per tenant or request. □ **Retention policies** — default expiration for jobs/logs with overrides per tenant. □ **Tenant API** - Configure secure connectors for reading documents □ **Job API** Rich semantic search beyond specified metadata fields.

	□ **Multi-User** - Initial design for single Tenant consuming via API □ **Web application** - Clean login, configure, health, logs, and job create/retrieval
Not in scope	□ OCR for hand written documents or filled forms □ Full text semantic search □ Human review and labeling workflows □ LLM Configuration per tenant/job/user/api key

Project Timeline

Phase 1 – MVP (Sept–Oct 2025)

Deliverables:

- Local filesystem connector
- Job API (submit config, track status, retrieve results)
- Configurable extraction schema with optional hints (regex, date formats, section markers)
- Batch processing (up to 100k docs) with concurrency + retries
- JSONL output + Results API (paginated retrieval)
- Core observability: metrics (p50, p95, p99), structured logs, correlation IDs
- Health API (system + tenant-level metrics)
- Security baseline: TLS everywhere, API keys, tenant isolation

Milestone: MVP runs locally via `docker compose up`; demo on sample set (10k docs).

Phase 2 – Beta (Nov–Dec 2025)

Deliverables:

- S3 connector (LocalStack first, AWS S3 in prod)
- Budget estimation pre-checks + quota enforcement (max docs/job, per-tenant quotas)
- Cost metrics exposed via API (per page, per tenant, per connector)
- Dashboards + alerts for errors, latency breaches, and cost anomalies
- Audit logs (per-job: who submitted, config used, outputs)
- Traceability: correlation from API → worker → connector → extractor
- Fail-fast cost policies
- Secret management integration (Vault/KMS/Secrets Manager)

Milestone: Beta release to internal users with 50k–100k doc runs; validate scale + cost controls.

Phase 3 – GA (Q1 2026)

Deliverables:

- SharePoint connector
- Multi-tenant RBAC + Tenant API (create tenants, configure connectors)
- Immutable audit logs (WORM) + retention policies
- Enhanced PII handling: tagging, redaction/masking, right-to-delete workflows
- Run history persistence for trend analysis + debugging
- Full observability dashboards in production (Grafana/Datadog)
- Load-tested to 100k documents in <24h with p95 <10s/doc

Milestone: General Availability; external tenants onboarded; compliance + audit readiness.

Future / Nice-to-Have (Post-GA)

- Streaming Results API
- Per-job PII configs + automatic redaction

- Rich semantic search (beyond metadata fields)
- Web application for non-API users (job creation, logs, monitoring)

Acceptance Criteria

Functional

- System accepts **job payloads** specifying file source(s), metadata fields, and optional extraction hints.
- System processes **up to 100k documents per job** from filesystem (MVP), S3 (Beta), and SharePoint (GA).
- For each document, system produces **normalized JSONL output** matching the extraction schema.
- **Results API** provides paginated JSONL output, job status, and metrics.
- Jobs are **idempotent**: retries do not create duplicate outputs.
- **Budget pre-checks** and **quota enforcement** are applied before job execution (reject/warn if exceeded).
- **Fail-fast policies** stop a job if budget is exceeded mid-run.
- **Tenant API** allows creation/configuration of new tenants and API keys.

Observability & Reliability

- **Metrics endpoint** exposes:
  - Docs processed, error count, retries, throughput
  - Latency (p50, p95, p99)
  - Per-tenant and system-wide job counts
- **Logs** are structured JSON, correlated by job ID and tenant ID.
- **Tracing** spans API → worker → connector → extractor with correlation IDs.
- **Dashboards and alerts** exist for error spikes, latency breaches, cost anomalies.
- **Run history** is persisted for at least N days (configurable) and retrievable by job ID.

Security & Compliance

- **TLS enforced** for all API traffic and worker/storage comms.
- **At-rest encryption** applied for all job metadata and outputs.
- **Authentication/authorization** enforced: tenant isolation, RBAC, per-tenant API keys.
- **Secrets management**: no credentials in env/logs; KMS/Vault integration.
- **Audit logs** captured per job: who submitted, config used, files processed, timestamps.
- **Traceability**: every output record links back to its input doc + job ID.
- **Schema version + extractor version** logged with every run.
- **Immutable logs** (append-only or WORM) enabled for audit compliance.

Performance & Scale

- Must process **100k documents in <24h**.
- Extraction latency: **p50 <2s, p95 <10s per doc**.
- Accuracy: **≥95% correct field extraction** vs. spot-checked ground truth.
- Cost: **< $0.01 per page** when using LLM extraction.

Out of Scope (explicit rejection tests)

- OCR of scanned/handwritten docs must not be attempted.
- Semantic search or embedding retrieval not provided.
- Human-in-the-loop workflows not supported.