Extraction Service – Scaling

Creator **Michael Clark** Created **Aug 24, 2025, 22:57** Last updated **Aug 24, 2025, 22:57**

Scaling Strategy

Target Scale

- Must process up to **100k documents per job**.
- Expected output: potentially **millions of JSONL records**, sharded to keep files ≤250 MB compressed.
- SLA: complete within **24 hours**, with p95 <10 s/document.

Concurrency & Fan-Out

- **Step Functions + SQS** handle fan-out of work.
- **Lambda workers** scale horizontally with SQS concurrency.
- Default concurrency: **1,000 workers/job**, tunable per tenant.
- Each worker processes ~10–20 docs/minute; system can process **~20k docs/hour** under nominal load.

Backpressure

- **SQS queue depth** is primary signal.
- If queue age > SLA thresholds → throttle enqueuing or autoscale workers up.
- **Worker memory/CPU alarms** trigger downscale/backoff.
- **Per-tenant partitions** ensure one tenant cannot starve others.

Rate Limits & Quotas

- **Per-tenant quotas**:
  - Max **100k docs/job**
  - Max **N concurrent jobs/tenant** (configurable, e.g., 3–5)
  - Daily/weekly caps enforced in **DynamoDB**.
- **API Gateway usage plans** enforce request rate & burst limits.
- **Step Functions throttling** prevents runaway concurrency across tenants.

Cost Caps & Controls

- **Pre-flight estimator**: estimates token/pages → $ based on config (LLM or not). Reject or warn if projected > budgetCap.
- **Mid-run monitor**: Lambda checks cumulative spend (docs × tokens × cost/page). Cancels job if > budgetCap (fail-fast).
- **Per-tenant budgets**: AWS Budgets alarms → EventBridge → Slack/email.
- **Metrics exposure**: $ per page, per tenant, per connector.

Observability for Scale

- **CloudWatch metrics**: processedDocs, errorCount, retries, throughput, p50/p95/p99, SQS backlog, cost spend.
- **Dashboards**: throughput per tenant, cost per tenant, SLA heatmaps.
- **Alarms**: queue age, error spikes, latency breaches, cost overruns.