AI Document Intelligence Platform - System Design

Problem Statement

Multi-tenant platform that ingests documents (batch + real-time), extracts structured information using OCR, layout analysis, NLP, computer vision, and multimodal models, stores results aligned to tenant-defined schemas, and supports semantic search, RAG, and integration exports.

Requirements

Functional Requirements

Requirement	Notes
Process various document types	PDFs, images, emails, contracts, invoices, etc.
Multi-source ingestion	REST API, S3/GCS events; validation and tenant tagging
Dual processing modes	Real-time (dedicated path) vs Batch (standard queue)
Event-driven pipeline	Stateless workers: OCR, layout, table extraction, NLP, embedding
Model routing	Cheap-first with confidence-based escalation to VLM/LLM
Schema-driven extraction	Tenant defines fields/types; output aligned to schema
Results delivery	JSON/CSV export, webhooks, polling API, results bucket
Search and retrieval	Metadata filtering, vector search, hybrid, RAG
Agentic orchestration	Constrained planner for ambiguous cases (Phase 3)
HITL integration	Route low-confidence extractions to human review
Deletion workflow	doc_id-driven cascade purge across all stores

Non-Functional Requirements

Requirement	Target	Notes
Throughput	100M docs/month	~50 docs/sec avg, 100 peak, 1000 burst
Latency (real-time)	P95 < 10 seconds	Dedicated capacity, warm worker pool
Latency (batch)	< 1 hour	Capped workers, backpressure
Availability	Medium for processing, High for retrieval	Durable queues, model fallbacks
Tenant Isolation	Zero data leakage	tenant_id enforced everywhere
Data Residency	EU data in EU region	Regional deployment option
Extraction Accuracy	>95% on golden set	Multi-model consensus for critical fields

Cost per Document	<$0.01 average	Tiered routing, batching, storage lifecycle
Traceability	End-to-end trace	trace_id = doc_id:run_id
Compliance	GDPR, SOC2, HIPAA-ready	Audit logs, deletion proof, data minimization

Key Trade-offs

Tension	Resolution
Latency vs. Cost	Cheap-first: OCR → VLM → LLM (escalate on low confidence)
Accuracy vs. Speed	Confidence thresholds; HITL for critical fields
Isolation vs. Efficiency	Bridge-pool: shared compute, isolated data
Real-time vs. Batch	Physical separation: fast path vs standard queue
Event-driven vs. DAG	Event-driven baseline; agentic planner for complex cases

---

Calculations

Load Estimation

Metric	Formula	Result
Average RPS	100M / month	~50 docs/sec
Peak RPS	2× average	~100 docs/sec
Burst RPS	10× average	~1,000 docs/sec
Peak concurrency	100 docs/sec × 10 sec	~1,000 workers

Storage Estimation (36-month retention)

Storage Type	Formula	Total
Raw documents	100M × 1MB × 36 mo	~3.6 PB
Metadata + extractions (SQL)	100M × 5KB × 36 mo	~18 TB
Embeddings	100M × 6KB × 36 mo	~22 TB

Cost Estimation (Annual)

Category	Annual Cost	Notes
Model inference	~$50-100K	Tiered: OCR (90%), VLM (8%), LLM (2%)
Storage (tiered)	~$400-600K	Lifecycle: hot → warm → cold

Compute (Spot/Reserved)	~$200-300K	Batch on Spot, RT on Reserved
Data transfer + DB	~$100-150K	
Total	**<$1M/year**	Target <$0.01/doc fully loaded

Data Model

Key Identifiers

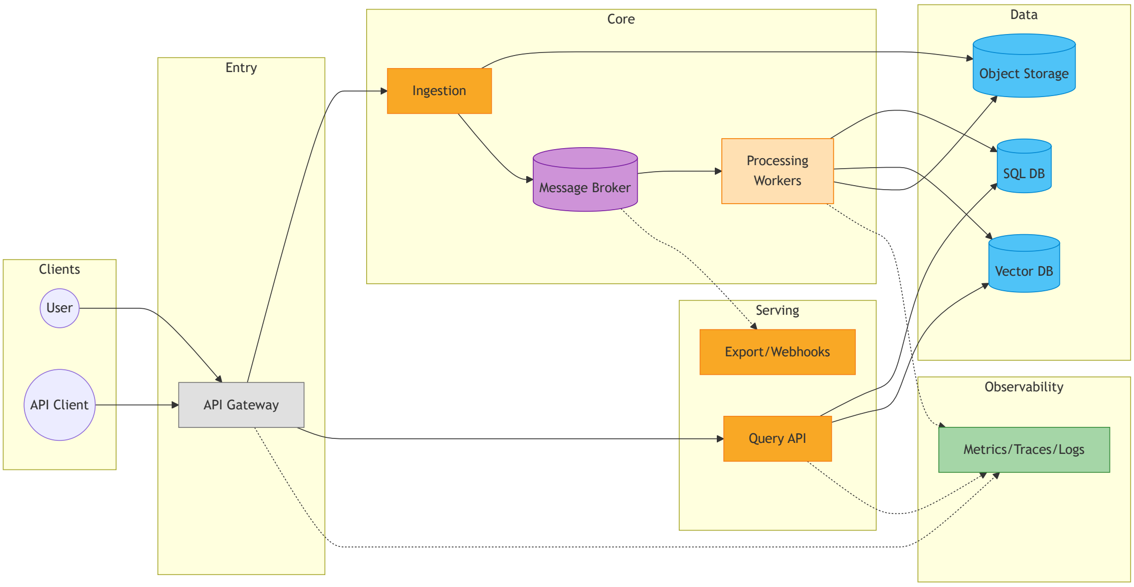
Identifier	Description
`tenant_id`	Partition key; enforced at ingress
`doc_id`	Stable document identity (UUID)
`run_id`	Processing attempt (enables retries)
`trace_id`	`doc_id:run_id` for end-to-end tracing

Core Tables

Table	Key Fields	Purpose
Documents	doc_id, tenant_id, status, doc_type, s3_uri	Document registry
PipelineRuns	run_id, doc_id, status, model_versions	Processing attempts
StepStatus	run_id, step_name, status, duration_ms, error	Per-step tracking
Extractions	doc_id, field_name, value, confidence, bbox	Extracted fields
TenantSchemas	tenant_id, doc_type, field_definitions	Tenant extraction schema

---

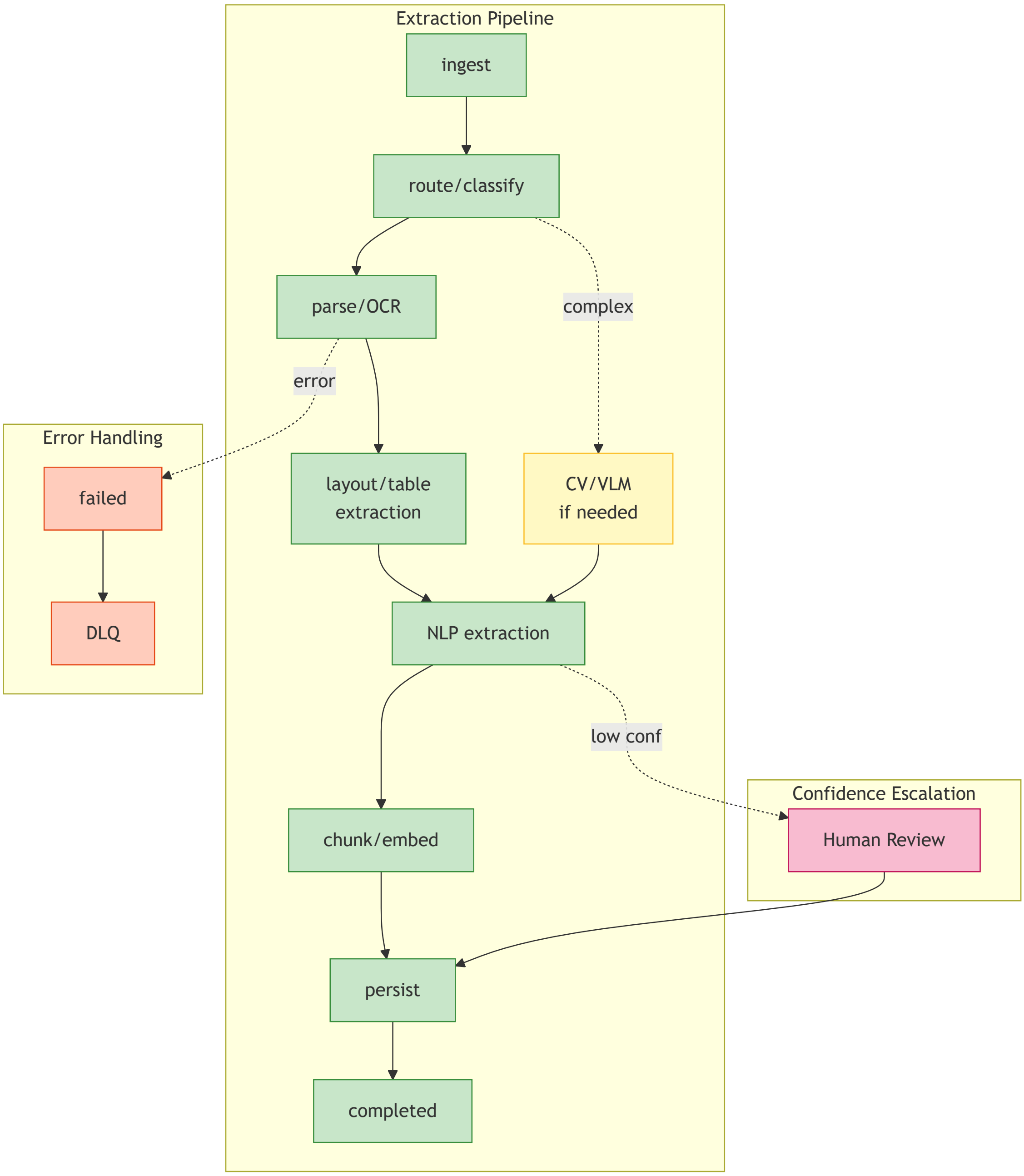
High-Level Design



Components:

- **API Gateway**: Auth, rate limiting, tenant resolution, telemetry injection
- **Ingestion**: Validation, malware scan, tenant tagging, queue routing
- **Message Broker**: Topics per stage; real-time vs batch priority
- **Workers**: Stateless; OCR, layout, table extraction, NLP, CV, embedding
- **Data**: Object storage (raw), SQL (metadata/extractions), Vector DB (embeddings)
- **Query API**: Search, RAG, results retrieval
- **Export**: Webhooks, polling, results bucket
- **Observability**: Cross-cutting metrics, traces, logs (OpenTelemetry)

Processing Pipeline



Processing Approach:

- **Two modes**: Real-time (dedicated queue, warm pool) vs Batch (standard queue, autoscaling)
- **Stateless workers**: Consume → process → write state → publish next
- **Confidence escalation**: Low confidence → try VLM/LLM → still low → HITL
- **Idempotency**: `(doc_id, run_id, step)` prevents duplicates
- **Retry**: Exponential backoff; max retries → DLQ

Client Integration

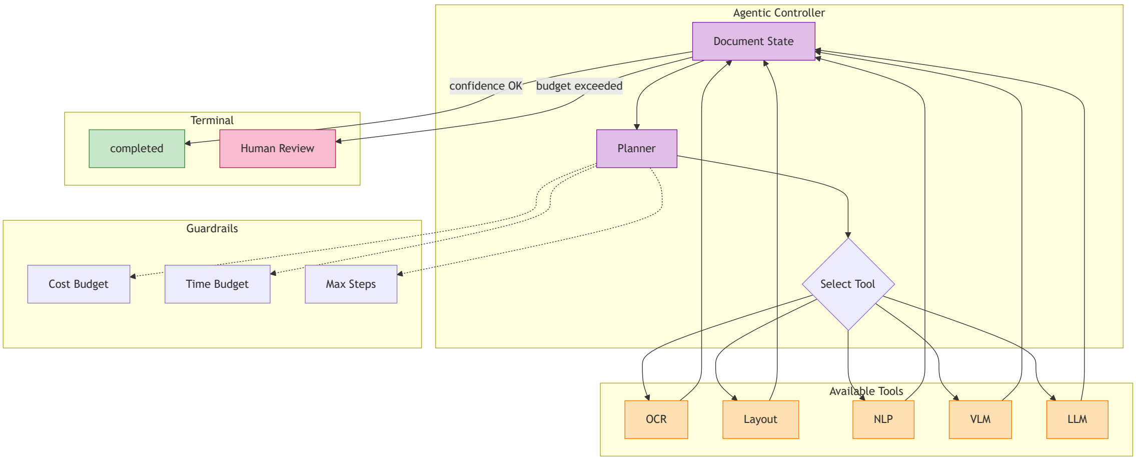
Tenant Onboarding

Artifact	Description
Schema definition	Fields, types, required/optional per doc_type
Document taxonomy	doc_types, labels, classification rules
Delivery config	Webhook URL, polling enabled, results bucket path

Delivery Mechanisms

Method	Use Case
Webhook	Push notification on completion
Polling API	Client polls by doc_id or batch_id
Results bucket	Tenant-scoped S3 path for bulk retrieval
CSV export	Scheduled or on-demand tabular export

Agentic Orchestration (Phase 3)



Agentic Approach:

- **Constrained controller**: Maintains document state, selects next tool, observes results
- **Tool loop**: Can iterate (limited) when confidence is insufficient
- **Guardrails**: Time budget, cost budget, max steps
- **Execution**: Agent publishes commands to worker topics; workers execute and return
- **Fallback**: Budget exceeded or still low confidence → HITL

Observability & Traceability

Identifiers

- `tenant_id` , `doc_id` , `run_id`
- `trace_id = doc_id:run_id` propagated across all services

Telemetry

Type	Coverage
Metrics	Throughput, latency/step, error rate, cost/doc, queue depth
Traces	OpenTelemetry spans across ingestion → processing → persist
Logs	Structured JSON with trace_id correlation

Monitoring & Alerting

Signal	Alert
Error rate spike	> 5% errors in 5 min
Latency degradation	P95 > 15s for real-time
DLQ growth	> 100 messages
Cost anomaly	> 2x expected cost/doc
Model drift	Confidence distribution shift

Lineage

- `model_versions` recorded per run (which model produced which extraction)
- Enables A/B testing and rollback

MLOps Boundary

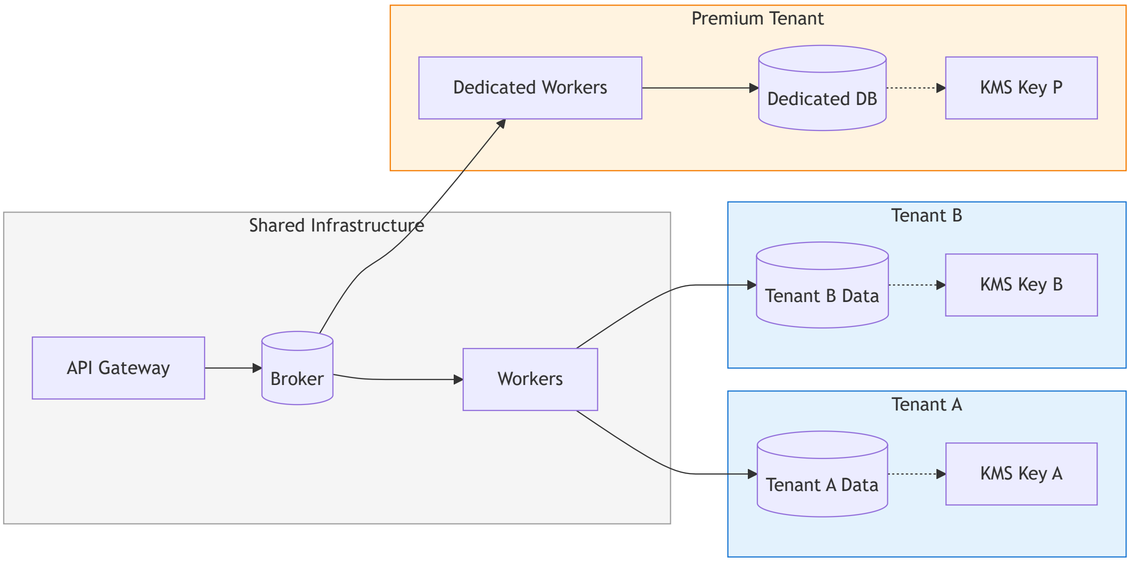
Scope

The IDP platform consumes models from another system - AI/ML platform (model registry, training pipelines, defauld serving configs, etc.). IDP's responsibility:

Responsibility	Implementation
Model versioning	Record `model_version` per step in PipelineRuns
Inference serving	Deploy models via K8s + FastAPI; canary rollouts
Feedback collection	Emit structured traces, HITL corrections, low-confidence samples
Evaluation	Run against golden datasets; block deploy if accuracy drops

---

Tenant Isolation

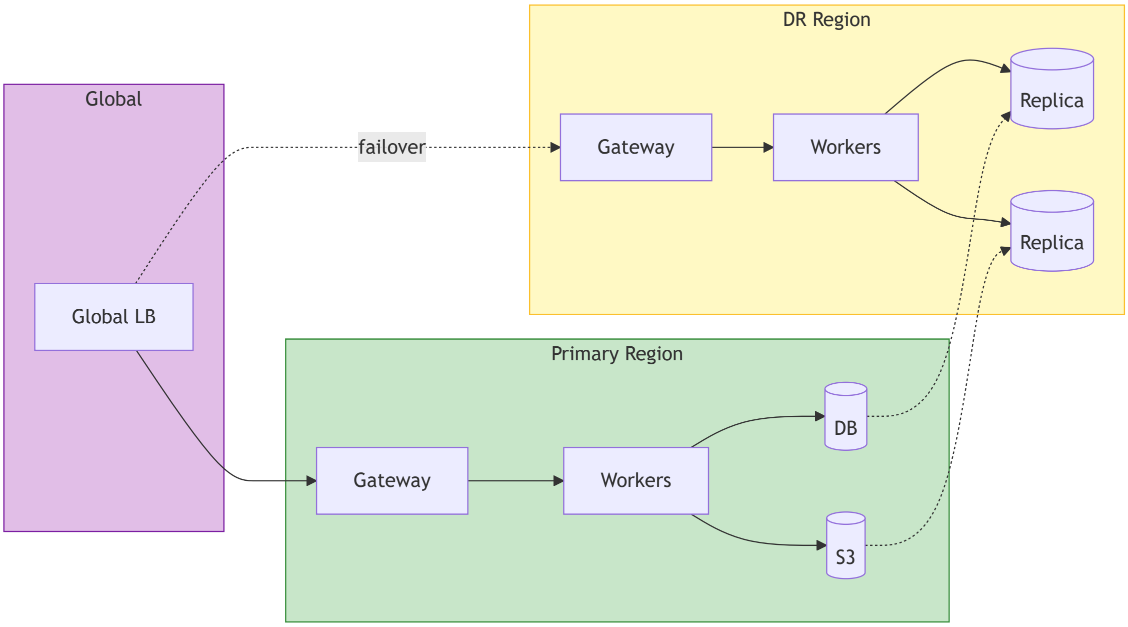


Strategy:

- **Standard**: Shared compute, tenant-partitioned data, per-tenant encryption keys
- **Premium**: Dedicated workers, dedicated database, optional dedicated VPC
- **Enforcement**: tenant_id resolved at gateway, propagated everywhere

---

Disaster Recovery



- **RPO**: < 1 hour (async replication)
- **RTO**: < 4 hours (warm standby)

Retrieval & RAG

Mode	Use Case
Metadata filter	By tenant, doc_type, date
Semantic search	Vector similarity
Keyword search	Exact match, boolean
Hybrid	Vector + BM25 rerank

RAG: Query → retrieve top-k → rerank → generate response → return with citations

---

Iterations

Phase 1: Baseline

- Single region, basic multi-tenancy
- REST API ingestion
- Pipeline: OCR + NLP
- PostgreSQL, basic webhooks

Phase 2: Extended

- Add VLM tier, layout/table extraction
- Real-time fast path (dedicated queue + warm pool)
- Vector DB for semantic search
- HITL queue, golden dataset CI/CD gates
- Client integration (schema, JSON export, webhooks)

Phase 3: Scale & Agentic

- Agentic orchestrator for complex docs
- Premium LLM tier
- Multi-region DR
- Tenant sharding at scale

Evaluation & Quality

Mechanism	Description
Golden datasets	100-500 labeled cases per doc_type
CI/CD gate	Block deploy if accuracy drops
A/B testing	Shadow traffic for model comparison
Runtime metrics	Latency, error rate, cost/doc, correction rate
Feedback loop	HITL corrections → labeling → retraining

Security & Compliance

Requirement	Implementation
Tenant isolation	API key ↔ tenant_id; RLS / sharding
Encryption at rest	Per-tenant KMS; BYOK for premium
Encryption in transit	TLS
Data residency	Regional deployment option
Deletion	doc_id cascade across all stores
Data minimization	Ephemeral processing option
Audit logs	Immutable access logs
Compliance	SOC2, GDPR, HIPAA-ready

Technology Options

Component	Options
Object Storage	S3 / GCS / Azure Blob (lifecycle tiering)
Message Broker	Kafka / Pub/Sub / SQS
Database	PostgreSQL (+ Citus for sharding)
Vector DB	Weaviate / pgvector
Observability	OpenTelemetry + Prometheus/Grafana + Langfuse
Model Serving	K8s + FastAPI + GPU

Other Considerations

Reliability

- Stateless workers; all state in DB
- Idempotency: `(doc_id, run_id, step)` keys
- Retries with backoff → DLQ
- Model fallbacks on failure
- Warm pool for real-time (avoid cold start)

Cost Control

- Cheap-first routing (OCR 90%, VLM 8%, LLM 2%)
- Spot instances for batch
- Storage lifecycle (hot → warm → cold)
- Continuous batching for LLM serving