Solution Architecture Document (SAD)

Version: 1.0
Owner: Architecture & Platform Team
Date: 23 Aug 2025

---

1) Overview & Architectural Drivers

Business drivers: accelerate resume screening; improve match quality; reduce fake resumes; provide explainable Top-20 results and downloadable profile reports.
Technical drivers: AWS-native, secure PII handling, scalable RAG retrieval, fast vector search (FAISS/ Chroma), low-latency APIs (FastAPI), modular services on ECS/EC2, and a React/Next.js UI.
Constraints: US-only data residency, 99.9% availability, cost efficiency, RBAC, auditability.

---

2) High-Level Architecture

Logical components: - **UI Web App (React/Next.js)** – JD intake, filters, results, reports, admin. - **API Gateway / ALB + FastAPI Service (ECS)** – Auth, request routing, orchestration. - **RAG Service (ECS)** – Query builder, retrieval (hybrid), re-ranker, explanation. - **Vector Index (FAISS/Chroma on ECS/EC2)** – k-NN search; shards and snapshots. - **Parser/Embedder Services (ECS + Lambda)** – Ingestion pipeline (parse → normalize → embed). - **Metadata Store (DynamoDB or RDS)** – Resume/JD metadata, statuses, feedback, fraud scores. - **Object Store (S3)** – Landing, Curated, Snapshots buckets with KMS. - **Report Generator (ECS)** – HTML/PDF profile reports with provenance. - **Security & Edge** – Cognito (OIDC), WAF, IAM, KMS, CloudTrail.

Diagrams
- VPC/Network Topology: *diagram_vpc_network_topology.png*
- CI/CD Pipeline: *diagram_cicd_pipeline.png*
- See FRD for ingestion and RAG flow diagrams.

---

3) Deployment Topology (AWS)

- **VPC** with public subnets (ALB), private app subnets (ECS tasks), private data subnets (vector nodes, DB).
- **ALB (HTTPS)** terminates TLS; **AWS WAF** applies managed + custom rules; **Route 53** hosts zone.
- **ECS on EC2** (capacity via Auto Scaling Groups) for stateful vector shards and compute-heavy services.
- **Lambda** for event-driven orchestration on S3 object puts.
- **S3** buckets: `resumes-landing`, `resumes-curated`, `vector-snapshots` (SSE-KMS).
- **DynamoDB** (on-demand) or **RDS PostgreSQL** (Multi-AZ) for system metadata.
- **Cognito** for SSO; **Secrets Manager/SSM** for secrets/params.
- **CloudWatch** for logs/metrics; **OpenTelemetry** traces.

4) Component Design & Responsibilities

4.1 Web App (React/Next.js)

- JD input (text/file), skill chip editor, filter drawer (must-have, years, geo, auth, clearance, pay).
- Results grid (Top-20), evidence tooltips, fraud badge, download/report, comparison up to 5.
- Uses **Cognito Hosted UI** → retrieves JWT → calls FastAPI with bearer token.

4.2 API Service (FastAPI on ECS)

Endpoints (excerpt): - `POST /jd/parse` – extract skills/constraints from JD. - `POST /search` – JD query → Top-K → re-rank → Top-20 (scores + reasons). - `GET /candidate/{id}` – candidate profile (PII-masked). - `POST /report` – generate HTML/PDF profile(s); return S3 pre-signed URL. - Admin: `POST /weights`, `POST /taxonomy`, `GET /audit`. **Cross-cutting:** JWT validation (Cognito JWKs), request quotas, audit logging.

4.3 RAG Service

- **Query Builder:** JD → normalized skills, years per skill, domain, constraints.
- **Retriever:** hybrid search (vector + BM25 keyword) across sub-indexes (skills, experiences, summary).
- **Re-Ranker:** weighted scoring (semantic similarity, exact skill matches, years, recency, domain, geo, auth – fraud penalty).
- **Explainer:** evidence snippets with offsets + missing-skills list.

4.4 Vector Index Layer

- **Choice:** FAISS (HNSW/IVF-PQ) or Chroma (for management features).
- **Sharding:** ≤ 2M vectors per shard; shard by job family (e.g., data, cloud, backend) or hash.
- **Replication:** 2× across AZs for HA; reader replicas for heavy query loads.
- **Snapshots:** Nightly shard dumps to S3; warm-start on deploy.

4.5 Ingestion Pipeline

- **S3 Landing** → S3 event → **Lambda Orchestrator** queues jobs.
- **Parser (ECS)**: Textract/PyPDF/DOCX extraction; entity normalization (skills, titles).
- **Embedder (ECS/EC2)**: batch/text chunk embeddings; store in index service; write curated JSON to S3.
- **Metadata (DynamoDB/RDS)**: ingestion state, hashes, fraud signals.

4.6 Report Generator

- Templated (Jinja/Handlebars) HTML; server-side PDF (headless Chromium).
- Includes match score, breakdown, skills timeline, evidence; outputs to S3 with short-lived pre-signed URLs.

4.7 Fraud Detection

- Signals: date overlap checks, impossible seniority, clone detection via fuzzy hashes, LLM-style verbosity ratio, geo inconsistencies, unverifiable employers.
- Aggregated **Fraud Risk Score** used only for **re-ranking** and UI flagging.

---

5) Data Model & Storage Mapping

- **S3 (raw & curated)**: original files + parsed JSON; KMS, versioning, Object Lock for audit logs.
- **DynamoDB/RDS**: Resume, Experience, Skill, JDQuery, MatchResult, Feedback.
- **Vector store**: Embeddings per resume chunk and per skill/experience chunk; index metadata (version, shard) in DB.
- **Logs/metrics**: CloudWatch; traces in X-Ray/OTel backend.

---

6) Sequence Flows

6.1 JD → Search → Top-20

1. Recruiter logs in via Cognito; UI gets JWT.
2. UI `POST /search` with JD and filters.
3. API validates JWT → calls RAG Service.
4. RAG builds query → vector Top-K (e.g., 200) → keyword retrieval → merge/dedup.
5. Re-ranker scores; Top-20 returned with explanations + fraud flags.
6. API writes search event to audit; UI renders results.

6.2 Ingestion

1. Resume uploaded to S3 (UI/bulk).
2. S3 event triggers Lambda; job persisted to DB.
3. Parser extracts text/fields → Curated S3 + metadata.
4. Embedder computes embeddings → upserts vectors into FAISS/Chroma.
5. Index snapshot scheduled nightly; statuses updated.

6.3 Report Generation

1. UI selects candidates → `POST /report`.
2. Service assembles context & HTML → PDF → stores in S3.
3. API returns pre-signed URL → auto-expires.

---

7) Security Architecture

- **Identity:** Cognito user pool; groups map to RBAC roles (Recruiter, Lead, Admin).
- **Network:** Private subnets for services; S3/DynamoDB via VPC endpoints; NAT for outbound.

- **Edge:** ALB + WAF (managed rules + custom patterns); TLS 1.2+; HSTS.
- **Encryption:** KMS-CMKs for S3/EBS/RDS; TLS in transit.
- **Secrets:** SSM/Secrets Manager; rotation 90d; no long-lived keys.
- **Audit:** CloudTrail + app audit tables; Object Lock for logs.
- **PII Controls:** Mask PII by default in UI exports; DSAR and deletion workflows.

---

8) Observability & SRE

- **Logging:** JSON logs, correlation IDs, request/response sizes.
- **Metrics:** API latency/throughput; vector query time; ingestion queue age; fraud signal rates; report latency.
- **Tracing:** OpenTelemetry across API → RAG → vector shards.
- **Dashboards:** per-service plus business KPIs (Precision@K, Recall@K, nDCG).
- **Alerts:** P95 search > 3s; error rate > 1%; queue age > 10m; shard memory > 80%.

---

9) Availability, Resiliency & DR

- **HA:** Multi-AZ ECS & RDS; vector shards spread across AZs with replicas.
- **Snapshots:** Nightly FAISS/Chroma dumps to S3; DB PITR.
- **Degradation:** If generator fails, still return Top-20; if reports fail, CSV export fallback.
- **DR:** Warm standby in us-east-2; RTO $\leq$ 4h, RPO $\leq$ 24h.

---

10) Performance & Scalability Plan

- **Targets:** `/search` P95 $\leq$ 3s up to 5M resumes; $\leq$ 6s at 20M (roadmap).
- **Scaling:** Horizontal ECS scaling on CPU/latency; shard vector index by job family; cache hot results per JD for 10–30m.
- **Indexing:** IVF-PQ or HNSW for recall vs. latency trade-offs; offline recall tests.

---

11) Technology Choices & Rationale

- **FAISS vs Chroma:** FAISS for performance and control; Chroma if needing simpler ops/metadata.
- **DynamoDB vs RDS:** Start with DynamoDB (on-demand, flexible); RDS if complex joins/analytics.
- **FastAPI:** async support, OpenAPI native; easier performance.
- **ECS on EC2:** control over instance families for memory-heavy vector shards.

---

12) Data Governance & Privacy (Mapping)

- **Classification:** PII vs operational data; tag resources.

- **Retention:** 2 years inactive; configurable per client.
- **Deletion:** Propagate deletes to S3, DB, vector store (tombstones + re-build).
- **Access Reviews:** Quarterly RBAC audits; least-privilege IAM.

---

13) CI/CD & Environments

- **Branches:** trunk-based with feature flags.
- **CI:** GitHub Actions – lint, type-check, unit tests, container build, vulnerability scan.
- **Artifact:** push to **ECR**; SBOM published.
- **IaC:** Terraform/CDK plans; change sets reviewed.
- **CD:** ECS blue/green with ALB; canary 10%; automatic rollback on alarms.
- **Envs:** Dev → Staging → Prod (separate AWS accounts).
- **Diagram:** see *diagram_cicd_pipeline.png*.

---

14) Capacity Planning & Sizing (Initial)

- **Vector shards:** 3 shards × 1 replica (≈ 6 nodes) for 5M vectors (IVF-PQ), c7i.2xlarge (32 GB).
- **API/RAG:** 3–6 tasks, c7g.large; autoscale on CPU 60% or P95 latency.
- **Parser/Embedder:** spot c7g.xlarge for batch; target 120 resumes/min/node.
- **RDS (if used):** db.r6g.large Multi-AZ; or DynamoDB on-demand with auto-scaling.

---

15) Integration Points

- **Cognito** (Auth), **CloudWatch** (obs), **S3** (storage), **DynamoDB/RDS** (metadata), **ECR** (images), **Route53/WAF/ALB** (edge), optional **SQS** for ingestion queues.
- Future: ATS (Greenhouse/Lever) via REST, Slack webhooks for share links.

---

16) Risks & Mitigations (Architecture)

- **Index drift / taxonomy changes:** versioned embeddings; rolling re-index jobs.
- **Stateful vector nodes complexity:** automate snapshots & restore; pre-warm on deploy.
- **Cost spikes from embedding:** schedule windows, batch, and cache embeddings; use Spot for batch.
- **Fraud false positives:** keep as rank penalty; human confirmation required.

---

17) Testing Strategy (Tech-Facing)

- **Contract tests** for APIs; **IR eval** (Precision@K, Recall@K, nDCG).
- **Load tests:** k6 profiles with JD mixes; target P95 $\leq$ 3s.
- **Chaos drills:** kill a shard; confirm degraded yet functional search; restore from snapshot.

- **Security tests:** IAM least privilege, WAF rules, file sanitization.

---

18) Operational Runbooks

- **Deploy & rollback** steps; **index restore**; **PII purge** flows; **hotfix** protocol; **DR failover** checklist.

---

19) Cost Model (Initial Estimate)

- Compute for ECS vector shards/API: baseline $2–4k/month.
- S3 storage + data transfer: $200–600/month at 5M resumes.
- DynamoDB/RDS: $200–500/month.
- Misc (WAF, NAT, CloudWatch, snapshots): $300–800/month.
- **Unit costs target:** $\leq$ $0.05/resume ingest; $\leq$ $0.01/search at 5M scale.

---

20) Acceptance & Handover

- Architecture review sign-off by Product/Engineering/SecOps.
- Successful completion of load, chaos, and security tests.
- Runbooks and dashboards handed to on-call.

Appendix A – Diagrams

- **VPC/Network Topology:** *diagram_vpc_network_topology.png*
- **CI/CD Pipeline:** *diagram_cicd_pipeline.png*
- **(From FRD)** High-Level, Ingestion & Indexing, Matching/RAG flows.