CBP RAG + Embeddings – Critical Bottlenecks & Engineering Playbook (No Code)

# 0) Scope & Assumptions

Purpose: Keep LLM minimal (already deployed). Maximize investment in retrieval quality and compliance.

- Infra constraints: On‑prem / FedRAMP; ABAC required; SIEM integrated; no external SaaS unless cleared.

- KPI targets: Recall@10 ≥ target, p95 latency budget ≤ target, operator acceptance ≥ target.

- Resource tiers: GPU (low/medium/high) and Storage (low/medium/high) referenced below. No exact SKUs.

## Resource Tiers (Descriptive Only)

GPU compute: low (prototype/QA), medium (pilot), high (production with re‑rank).

Storage: low (≤10M vectors), medium (10–100M), high (≥100M+ with snapshots).

# 1) Data Intake & Chunking (critical bottleneck)

Purpose: Turn messy artifacts into stable, retrievable chunks with provenance and predictable recall.

Primary packages/services: PyMuPDF (fitz), Unstructured.

GPU/Storage: GPU=low; Storage grows with chunk count (low→high).

## Work Breakdown

1. Parse sources: PDFs (PyMuPDF), DOCX/HTML/MD (Unstructured), transcripts (pre‑parsed STT).

2. Normalize: strip boilerplate, deduplicate headers/footers, fix encodings.

3. Segment: sentence‑aware splits for text; turn/time‑based splits for transcripts; heading‑based splits for Confluence/JIRA.

4. Tag metadata: chunk\_id, doc\_id, source, timestamps, classification, project\_id/sprint\_id.

5. Validate: sampling checks—token length, empty chunks, duplicate ratios.

6. Persist: write chunks + metadata to RDBMS, push text to embedding queue.

## Decision Points

- Chunk size (512 vs 1,000 tokens): larger boosts recall but increases duplication risk.

- Transcript segmentation (by speaker vs fixed window): choose by downstream query style.

- Boilerplate removal thresholds: aggressive removal risks losing context; tune with eval set.

# 2) NER Extraction (JIRA, Confluence, MS Teams, Scrum notes)

Purpose: Promote task/action entities for retrieval and cross‑linking across collaboration systems.

Primary packages/services: spaCy, HuggingFace Transformers.

GPU/Storage: GPU=low–medium (batch NER on transcripts); Storage=low (entity tables), medium if embedding entities.

## Work Breakdown

A. Rule‑first: EntityRuler for TASK\_ID/PROJECT/ACTION\_ITEM patterns → fast, deterministic.

B. Model‑first: HF NER for PERSON/DATE/LOC → higher recall on noisy notes.

Merge spans, resolve overlaps, link to chunk\_id, persist to RDBMS; optional: push entity spans to embedding queue.

## Decision Points

- Rule vs model precedence: rules for IDs; model for natural entities; tie‑break with confidence.

- Entity embedding on/off: enable when queries target ACTION\_ITEM/DECISION specifically.

# 3) Text Embeddings (critical bottleneck)

Purpose: Encode chunks/entities into vectors optimized for IR recall and stability.

Primary packages/services: sentence-transformers, transformers.

GPU/Storage: GPU=medium (batch encoding); Storage scales with vectors (low→high).

## Work Breakdown

1. Select encoder family (keep same for corpus and queries).

2. Configure batching and normalization (L2 norm).

3. Version every vector with embedder\_version and date.

4. Encode batch → persist vectors + metadata; retry on failures.

5. Run offline Recall@k on validation set before exposing to production.

## Decision Points

- E5 vs Instructor: E5 is solid general baseline; Instructor allows task prompts but adds complexity.

- L2 normalization on/off: usually on for cosine similarity in ANN.

- Dimension reduction (PCA) for storage: only if metrics hold.

# 5) Vector Store (critical bottleneck)

Purpose: ANN retrieval with policy-aware metadata filtering, scalable by collection.

Primary packages/services: pgvector OR Qdrant (choose one).

GPU/Storage: GPU=none (unless GPU re‑rank is colocated), Storage=low→high by corpus size and snapshots.

## Work Breakdown

1. Choose store: pgvector (SQL‑centric) or Qdrant (service‑centric).

2. Define collections: text\_chunks, entities, code; define metadata schema (classification, project\_id, sprint\_id).

3. Create ANN indexes (HNSW for recall; IVF‑like if store supports for scale).

4. Ingest vectors in batches with backpressure; validate kNN results on a probe set.

5. Implement ABAC pre‑filters (WHERE clauses or filter objects) before scoring.

6. Set up replication and snapshot policy per storage tier.

## Decision Points

- One big collection vs many: separate per modality (text/entity/code) to isolate behavior.

- Metadata partitioning: pre‑partition by project/sensitivity for predictable latency.

- Snapshot cadence: low (monthly), medium (weekly), high (daily).

# 8) Query Pipeline (critical bottleneck)

Purpose: Maximize precision under latency budget while enforcing ABAC before scoring.

Primary packages/services: Haystack OR LlamaIndex (pick one).

GPU/Storage: GPU=medium–high if using cross‑encoder re‑rank; Storage unaffected.

## Work Breakdown

1. Encode query with same encoder family as corpus.

2. Apply ABAC filters → restrict candidate space (project\_id, classification).

3. ANN search (k=50–100).

4. Optional cross‑encoder re‑rank to k=10–20.

5. Deduplicate near duplicates; cap per source.

6. Assemble prompt context (IDs + snippets) and pass to LLM.

## Decision Points

- Re‑rank on/off: turn on when precision matters more than raw speed; batch on GPU if QPS is high.

- k values: k\_ann 50–100, k\_rerank 10–20; tune to latency budget.

- Context cap: 2–4 chunks; more hurts grounding via dilution.

# 9) LLM Generation (already deployed)

Purpose: Ground outputs; keep brief. Enforce context window, include citations/IDs, log decisions.

# 10) Governance, PII, and Audit

Purpose: Prevent exposure; prove compliance. Pre‑index masking; ABAC at retrieval; traces to SIEM.

# 11) API Layer

Purpose: Serve RAG to downstream tools. FastAPI/gRPC; mTLS + JWT/Kerberos; rate‑limit via Kong/Istio.

# 12) Security

Purpose: Harden endpoints and data paths. Encrypt at rest; TLS in transit; quarterly red‑team; zero‑trust.

# 16) Evaluation & Monitoring

Purpose: Detect drift/regressions. Offline Recall@k/MRR; online accept/reject; trigger re‑chunk or re‑embed as needed.

# 17) Minimal Build Stack (Defaults)

Purpose: Deployable v1 with minimal entropy.

- Intake: PyMuPDF (fitz), Unstructured

- NER: spaCy (+ EntityRuler)

- Embeddings: sentence-transformers (e5)

- Vector: pgvector OR Qdrant

- Orchestration: Haystack OR LlamaIndex

- API: FastAPI

- Security/Governance: OPA + OpenTelemetry + SIEM