

## GenAI Assessment Recommendation: RAG Webapp (Concise Approach)

Objective: Build a web-accessible Retrieval-Augmented-Generation (RAG) tool over the SHL product catalog.  
Pipeline (concise):

- 1) Catalog crawl: fetch product pages (title, desc, tags).
- 2) Index: compute text embeddings and store in FAISS/Weaviate.
- 3) Query: embed input, retrieve top-N, rerank with lightweight LLM prompt emphasizing balance between Knowledge and Personality tests.
- 4) API: Flask endpoint /recommend returning JSON list of {name,url,score}. Frontend: simple React page to query API.

Prototype used for submission: rule-based role detection mapping to canonical SHL URLs (fast baseline). Replace rule-based step with embedding retrieval for production.

## Implementation, Evaluation, and Deliverables

Implementation notes: Store product metadata as JSONL. Use OpenAI or local embeddings for vectors. Rerank with LLM prompt that enforces diversity (avoid duplicates) and coverage (mix K and P).

Evaluation: Use labelled train set to compute Recall@K and MRR. Submit predictions CSV: one row per (Query, Assessment\_url).

Deliverables packaged here:

- vipul\_predictions.csv (predictions)
- vipul\_RAG\_approach.pdf (this 2-page doc)
- vipul\_rag\_project.zip (Flask prototype, README, requirements, evaluation placeholder)