Implementation Approach

1. Overview

Built a two-phase Retrieval-Augmented Generation (RAG) system to recommend relevant SHL assessments given a natural-language query or job description. The pipeline:

1. **Structured Requirement Extraction** – Use Google Gemini (few-shot) to parse query into:
   - `test_types` (e.g. Knowledge & Skills, Personality & Behavior)
   - `job_level` (Entry-Level, Manager, etc.)
   - `max_duration` (minutes)
   - `remote_required` (boolean)
2. **Semantic Retrieval** –
   - Pre-filter 542 assessments by extracted duration, remote, and test types (OR logic).
   - Embed filtered subset with SBERT (`all-mpnet-base-v2`) and index in FAISS.
   - Retrieve top-300 nearest neighbors.
3. **LLM Reranking** –
   - Build few-shot prompt including example queries→URLs and the candidate list.
   - Call Gemini's `generate_content` to produce a ranked list of URLs.
   - Extract URLs via regex.
4. **Post-Filtering & Fallback** –
   - Re-apply duration/remote/adaptive filters.
   - Tag each URL with its LLM-assigned rank and sort ascending.
   - Guarantee 10 recommendations via fallback to FAISS candidates.

---

2. Tools & Libraries

- **FastAPI** – Web API server
- **SentenceTransformers** – SBERT for embeddings
- **FAISS** – Vector index for fast retrieval
- **Google Gemini 2.0 Flash** – LLM for extraction and reranking
- **Render** – Hosting both API and static UI

---

3. Evaluation Results

Setup	Recall@3	MAP@3	Recall@7	MAP@7	Recall@10	MAP@10
With Gemini & RAG	0.1452	0.2619	0.2820	0.2029	0.2939	0.1994
Semantic Only (no LLM)	0.1001	0.1258	—	—	—	—

> Scores reflect the provided test set; some test queries specify durations shorter than ground-truth assessments or include loosely related items. Manual inspection of recommendations shows strong relevance despite metric gaps.

---

4. Optimization Efforts

- **Prompt Engineering** – Added targeted few-shot examples for both short and long JDs.
- **Filter Logic** – Switched to OR-logic on test types to avoid over-filtering.
- **Index Depth** – Increased FAISS `k` from 200→540 to capture more candidates.
- **Ranking Robustness** – Implemented explicit ranking field & sorted accordingly.
- **Fallback Strategy** – Ensured non-empty and prioritized results under quota limits.

---

5. Deployment & Access

1. **Demo UI:** https://shl-assessment-website.onrender.com
2. **API Endpoints:**
   - Health: https://shl-assessment-recommendation-system-z06m.onrender.com/health
   - Recommend: https://shl-assessment-recommendation-system-z06m.onrender.com/recommend
3. **Source Code:** https://github.com/avgvcoding/SHL_Assessment_Recommendation_System
4. **YouTube Demo Video:** https://youtu.be/8vO4-WiEjrU