Project Architecture and Outline: Agentic RAG for SHL Solution Recommendation

1. Introduction (Problem & Solution)

Problem: SHL needs an intelligent system to recommend appropriate assessment solutions for users based query on their specific job role requirements and problem queries. Manually sifting through the catalog of solutions can be time-consuming and may not always yield the optimal choice.

Solution: We will develop an **Agentic Retrieval-Augmented Generation (RAG) system** that leverages a knowledge base of SHL solutions and an intelligent agent to understand user needs and **recommend the most suitable assessments, along with relevant details and links to fact sheets**. This system will improve efficiency, user experience, and the likelihood of successful solution adoption.

2. System Architecture

Core Components:

User Interface: A front-end (Streamlit) where users can input their problem query & select language, job level, and completion time for product recommendation.

Agent: Used an intelligent Reasoning agent (powered by a Large Language Model - GROQ LLM API) **Deepseek-llama-r1-70b** responsible for understanding user intent, deciding which tools to use, and generating the final recommendation.

Data Wrangling: Crawled data from the website using **BeautifulSoup, Scrapy & FireCrawl** and **created a CSV and JSON** file containing all important features -title,description,job_level, languages,assessment_length, downloads, URL,test_type,content_hash for unique ID in the database.

Knowledge Base: A vector database (**ChromaDB**) storing embeddings of SHL solution's information extracted from the provided **JSON & CSV dataset**. Metadata will include solution details, language availability of fact sheets, and links to the PDF fact sheets which will guide our LLM Routing Mechanism to tools.

Agentic Retrieval Mechanism: The initialise_collection and process_collection tool is used by the agent to query the ChromaDB and retrieve relevant SHL solutions based on semantic similarity to the user's query.

Hybrid Search: Integrates and verifies content using **DuckDuckGO search**

PDF Processor: The pdf_processor & search_pdf_content tool used by the agent for process relevant PDF fact sheets (download, extract text, chunk, embed relevant sections) to provide more detailed context if needed.

LLM API: The interface to the chosen Large Language Model Deepseek (e GROQ API) for agent reasoning, decision-making, and response generation.

Final Data Flow:

The user provides a job role and problem query (simulated via code initially). The agent receives the query and analyzes it to understand the user's needs. The agent uses the shl_solution_retrieval tool to query the ChromaDB based on the user's query, retrieving relevant SHL solutions and their metadata. Then at last **LLM will reason and generate final output with feedback Statistics for Augmenting correction in Output.**