# Algorithm: Retrieval-Augmented Generation for Career Guidance (RAG)

## 1. Initialization

### 1.1 Define Knowledge Bases:

1.1.1 KB1: O\*NET Database with Comprehensive occupation titles and detailed descriptions

1.1.2 KB2: LinkedIn Job Scraping Module- Current job postings scraped from the Arbeitnow API based on relevant job titles.

### 1.2 Initialize System Components:

1.2.1 Input Module: Collects user inputs including resumes, psychometric profiles, and academic history.

1.2.2 Keyword Extraction Module: Identifies essential skills and keywords from user data using Natural Language Processing (NLP) techniques.

1.2.3 Retriever Module: Fetches relevant job descriptions from KB1 based on extracted keywords.

1.2.4 Similarity Search Module: Measures semantic similarity between user skills and job descriptions.

1.2.4 Ranking Module: Orders job matches based on relevance scores derived from similarity metrics.

1.2.5 Scraping Module: Retrieves job postings from the Arbeitnow API for top-ranked job titles.

1.2.6 Generator Module: Creates personalized career guidance using aggregated information.

1.2.7 Visualization Module: Displays skill match percentages and skill gap analysis.

1.2.8 Unsupervised Clustering Module: Performs topic modeling on O\*NET occupation descriptions to uncover underlying themes.

1.2.9 Output Module: Presents final recommendations and job listings to the user.

### 1.3 Set Parameters:

1.3.1 Set Number of top relevant jobs to identify

1.3.2 Set Criteria for keyword relevance and match scores.

1.3.3 Set Parameters for LinkedIn searches, such as job titles.

1.3.4 Set LDA paramteters for topic modeling (e.g., number of topics, iterations).

### 1.4 Load Necessary Resources and Models:

1.4.1 Load spaCy For skill extraction.

1.4.2 Load SentenceTransformers for generating semantic embeddings.

1.4.2 Load LDAModel for unsupervised clustering on O\*NET data.

## 2. User Input Collection and Processing

### 2.1 Collect User Inputs:

2.1.1 User uploads a resume in PDF or DOCX format.

2.1.2 User inputs such as personality traits, strengths, and work preferences.

2.1.3 User inputs details including degree, institution, graduation year, and certifications.

2.1.4 User inputs work and location preferences

### 2.2 Extract and Preprocess Information:

2.2.1 Extract text from the uploaded resume.

2.2.2 Identify and extract relevant skills from the resume text.

2.2.3 Consolidate psychometric inputs into a structured format.

2.2.4 Aggregate academic details into a comprehensive summary.

### 2.3 Store Processed User Profile:

2.3.1 Combine extracted data (skills, academic history, psychometric profile) into a structured format for further processing.

## 3. Keyword Extraction and Retrieval

### 3.1 Extract Keywords and Skills:

3.1.1 Utilize the Keyword Extraction Module to identify and extract key skills, competencies, and relevant keywords from the user's resume and profile.

### 3.2 Retrieve Relevant Job Descriptions (Retriever Module):

3.2.1 Extract skills and keywords.

3.2.2 Query KB1 (O\*NET Database) to fetch job descriptions that align with the extracted keywords.

3.2.3 Return a subset of job descriptions from KB1 relevant to the user's profile.

## 4. Similarity Measurement and Ranking

### 4.1 Compute Similarity Scores:

4.1.1 Use the Similarity Search Module to generate semantic embeddings for user skills using the SentenceTransformer model.

4.1.2 Compare these embeddings with precomputed embeddings of job titles from KB1 using cosine similarity.

4.1.3 Return similarity scores indicating how closely each job description matches the user's skills.

### 4.2 Assign Relevance Scores:

4.2.1 Score each job based on the similarity metrics to determine its relevance to the user's profile.

### 4.3 Rank Jobs by Relevance:

4.3.1 Order all jobs in KB1 based on their relevance scores in descending order.

### 4.4 Select Top\_N\_Jobs:

4.4.1 Choose the top N most relevant jobs for further processing.

## 5. Job Scraping from LinkedIn (KB2)

### 5.1 Configure Scraping Parameters:

5.1.1 Set search criteria based on the selected top job titles, including location preferences and job type.

### 5.2 Execute Scraping:

5.2.1 Collect current job postings from LinkedIn by searching the top job titles.

### 5.3 Aggregate Job Postings:

5.3.1 Compile details such as job title, company name, location, job description, application links.

## 6. Generation of Personalized Career Guidance

### 6.1 Prepare Input for Generator Module:

6.1.1 Combine user data (skills, academic history, psychometric profile), top job matches, and scraped job postings.

### 6.2 Invoke Generator Module:

6.2.1 Feed the consolidated input as a prompt to the generator module.

### 6.3 Integrate Job Postings:

6.3.1 Embed relevant job listings with direct application links into the guidance.

## 7. Visualization and Analysis

### 7.1 Skill Match Visualization:

7.1.1 Display skill match percentages for recommended jobs using Plotly bar charts.

### 7.2 Skill Gap Analysis:

7.2.1 Identify and visualize missing skills required for top job matches, suggesting areas for development.

### 7.3 Unsupervised Clustering with LDA:

7.3.1 Apply Latent Dirichlet Allocation (LDA) to uncover topics within O\*NET occupation descriptions.

## 8. Termination and Feedback

### 8.1 Finalize Recommendations:

8.1.1 Present the personalized career guidance and relevant job postings to the user in a structured format.

### 8.2 End of Algorithm:

8.2.1 Conclude the RAG process, ensuring all user recommendations are delivered and stored.