Recruitment Assistant Documentation

This document provides a detailed explanation of the Recruitment Assistant system, its functionality, and implementation. The system is designed to process candidate resumes, generate embeddings for similarity matching, and retrieve top candidates based on a job description.

# Table of Contents

1. **System Overview**
2. **Requirements**
3. **Architecture**
4. **Implementation Details**
   * AWS Integration
   * Resume Parsing
   * Embedding Generation
   * Candidate Retrieval
5. **Output Example**
6. **Conclusion**

# 1. System Overview

The Recruitment Assistant automates the process of evaluating resumes and ranking candidates based on their suitability for a given job description. It uses:

* **AWS**: For storage and metadata management.
* **OpenSearch**: For vector-based similarity search.
* **Google Gemini API**: For generating embeddings of resume text.

Key functionalities include:

* Uploading and parsing resumes.
* Generating embeddings and storing them in OpenSearch.
* Retrieving top candidates based on job description similarity.

# 2. Requirements

## Libraries

* **boto3**: AWS SDK for Python.
* **PyPDF2**: To parse PDF resumes.
* **FPDF**: To generate professional resumes.
* **OpenSearchPy**: For interacting with OpenSearch.
* **LangChain-Google-GenAI**: To generate embeddings using Google Gemini API.

## AWS Services

* **S3**: Stores resume files.
* **DynamoDB**: Stores metadata about resumes.
* **OpenSearch**: Stores embeddings and retrieves candidates.

## Additional Services

* **Google Gemini API**: Generates embeddings for textual data.

# 3. Architecture

The system comprises the following components:

## Resume Processing:

* + Resumes are uploaded to an S3 bucket.
  + Text is extracted and metadata is parsed.

## Embedding Generation:

* + Extracted text is converted into embeddings using the Google Gemini API.

## Data Storage:

* + Metadata is stored in DynamoDB.
  + Embeddings are indexed in OpenSearch.

## Candidate Retrieval:

* + A KNN search retrieves the top candidates based on job description embeddings.

# 4. Implementation Details

## AWS Integration

* **S3 Bucket**:
  + Stores PDF resumes.
  + Used for input and temporary file processing.
* **DynamoDB**:
  + Stores metadata like name, skills, education, experience, and S3 path.
* **OpenSearch**:
  + Hosts embeddings and performs similarity search.

## Resume Parsing

1. **PDF Parsing**:
   * Resumes are downloaded from S3.
   * PyPDF2 extracts text from the files.
2. **Metadata Extraction**:
   * **Skills**: Identified using predefined keywords.
   * **Experience**: Extracted using regex patterns.
   * **Education**: Extracted using regex for degrees (e.g., BSc, MSc).
3. **Metadata Storage**:
   * Parsed metadata is stored in DynamoDB.

## Embedding Generation

* Uses LangChain-Google-GenAI to convert text into dense vector embeddings.
* Embeddings capture the semantic meaning of the resume text.

## Candidate Retrieval

1. **Query Preparation**:
   * The job description is converted into an embedding.
2. **KNN Search**:
   * OpenSearch performs a similarity search against stored embeddings.
3. **Result Formatting**:
   * Results are ranked and summarized with metadata (skills, experience, education).

# 5. Output Example

**Job Description**

"Looking for a data engineer skilled in Python and AWS with 3+ years of experience."

**Output**

Top Candidates:

1. Candidate 2 Resume

   - Skills: Python, AWS, Spark

   - Experience: 3 years

   - Education: BSc Computer Science

   - Suitability: Strong match

2. Candidate 1 Resume

   - Skills: Python, AWS, Spark

   - Experience: 3 years

   - Education: BSc Computer Science

   - Suitability: Strong match

3. Candidate 8 Resume

   - Skills: Python, AWS, Spark

   - Experience: 3 years

   - Education: BSc Computer Science

   - Suitability: Strong match

4. Candidate 9 Resume

   - Skills: Python, AWS, Spark

   - Experience: 3 years

   - Education: BSc Computer Science

   - Suitability: Strong match

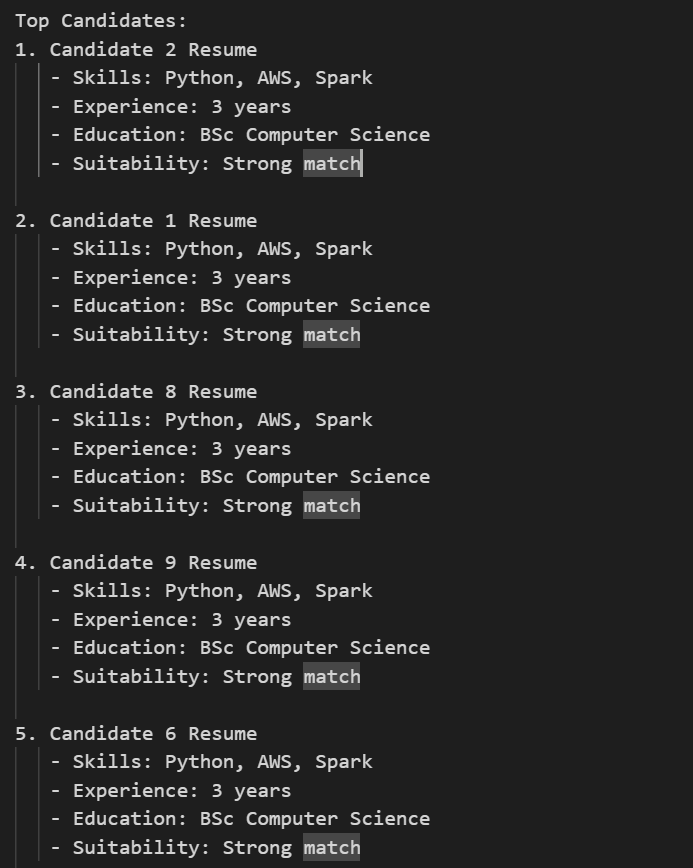
5. Candidate 6 Resume

   - Skills: Python, AWS, Spark

   - Experience: 3 years

   - Education: BSc Computer Science

   - Suitability: Strong match



# 6. Conclusion

The Recruitment Assistant provides a streamlined, automated pipeline for managing and evaluating candidate resumes. By leveraging AWS, OpenSearch, and Google Gemini API, the system ensures efficient and accurate retrieval of the most suitable candidates for any job description.

## Future Enhancements

* Add more advanced NLP techniques for parsing resumes.
* Enable feedback loops for improving ranking accuracy.
* Integrate with ATS (Applicant Tracking Systems) for end-to-end automation.

# 7. Screenshots

