# Software Design Document (SDD)

## Project: Jade Global— AI-Powered Skill Screener

## 1. Introduction

### Purpose

Skill Screener automates the talent acquisition screening process by leveraging Azure OpenAI to semantically compare candidate resumes with job descriptions (JDs), rank candidates based on match scores, and present the results in a structured and intuitive format.

### Scope

This system reduces manual effort, speeds up hiring decisions, and increases the accuracy and consistency of resume screening.

## 2. High-Level Architecture

Components:  
- Frontend: Streamlit web application (User interface)  
- Backend AI Engine: Azure OpenAI GPT model  
- File Processing Layer: PDF and DOCX text extraction  
- Data Processing Layer: Resume–JD semantic similarity scoring  
- Output Layer: Ranked candidates displayed in tabulated format

## 3. Functional Flow (User Inputs and Outputs)

### User Inputs

| Step | User Action | Input Type | Description |
| --- | --- | --- | --- |
| 1 | Upload Job Description (JD) | .txt | JD content used as the base reference |
| 2 | Upload Candidate Resumes | .pdf, .docx | Candidate resumes to evaluate |
| 3 | Trigger Evaluation | Button click | Starts the AI-based matching process |

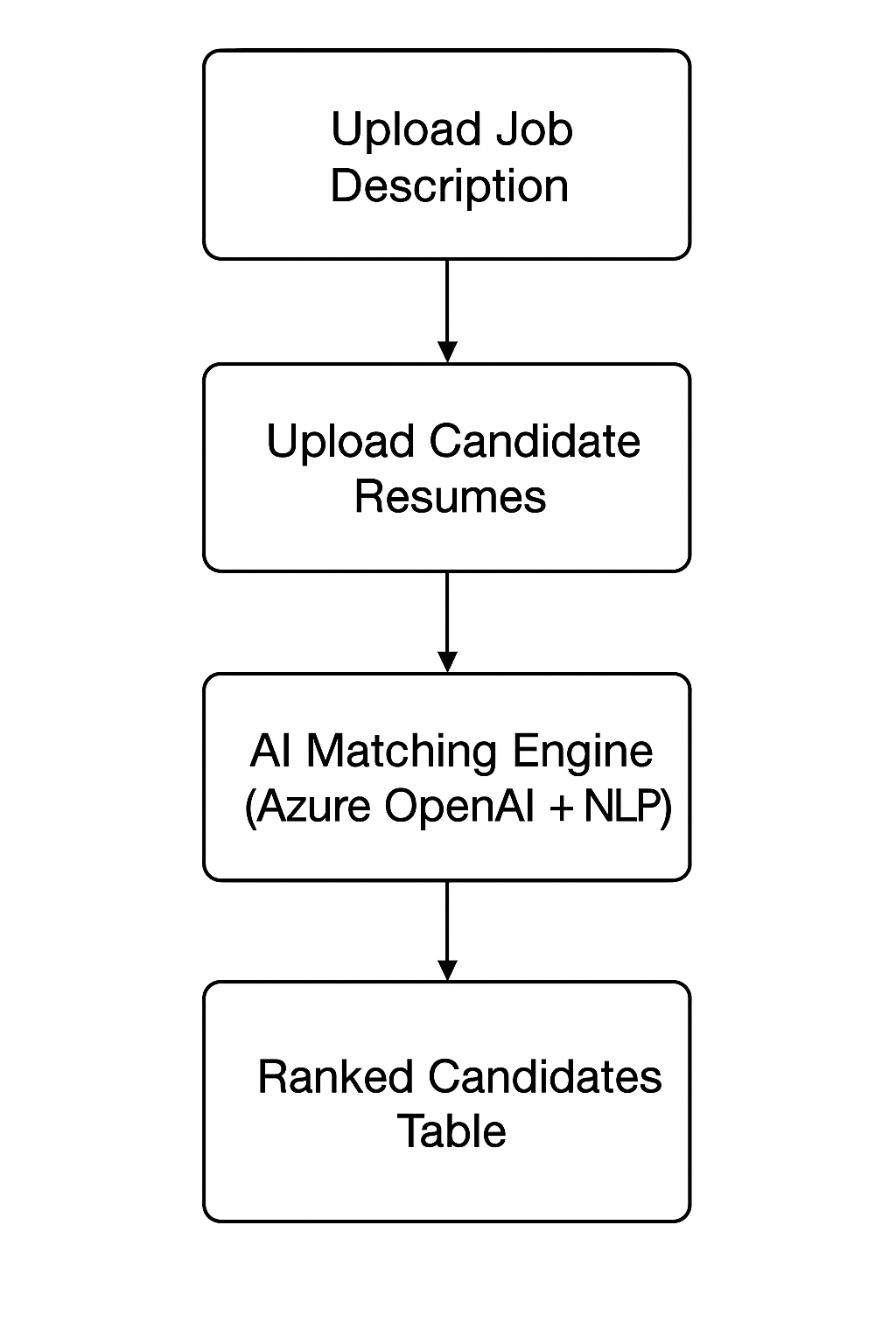
### System Processing

| Stage | Process Component | Description |
| --- | --- | --- |
| A | Text Extraction | Extracts raw text from uploaded JD and resumes using PyPDF2 and python-docx |
| B | Preprocessing | Cleans and normalizes text |
| C | AI Comparison Engine | Sends JD and resume text to Azure OpenAI GPT model to compute semantic similarity |
| D | Match Score Generation | Calculates a numerical score for each resume based on AI output |
| E | Sorting and Ranking | Sorts candidates by descending match score |

### System Outputs

| Step | Output | Description |
| --- | --- | --- |
| 1 | Ranked Candidate Table | Candidate names and match scores in descending order |
| 2 | Match Details (Optional) | Highlighted matching keywords or summary explanation |
| 3 | Downloadable Results | CSV/Excel export of ranked list |

## 4.UI Flow Diagram



## 5. Non-Functional Requirements

- Scalability: Must handle hundreds of resumes at a time  
- Security: Use Azure Key Vault for API keys and secure data transfer (HTTPS)  
- Performance: Provide results in under 30 seconds for up to 50 resumes  
- Compatibility: Supports PDF and DOCX resume formats

## 6. Technology Stack

| Component | Technology |
| --- | --- |
| Frontend | Streamlit |
| AI Engine | Azure OpenAI GPT |
| File Parsing | PyPDF2, python-docx |
| Output Formatting | Pandas |