# SkillMatch AI 系統技術手冊

文件版本： 1.0 (SpaCy 整合版)

日期： 2025年11月15日

## 1. 緒論

### 1.1 文件目的

本文件旨在為 SkillMatch AI 應用程式提供詳盡的技術說明與部署指南。內容涵蓋系統架構、核心演算法、部署配置檔案的設定、以及關鍵的套件依賴性管理。本手冊是未來維護、交接或系統遷移的依據。

### 1.2 系統概述

SkillMatch AI 是一個基於 Streamlit 開發的智慧職涯分析工具。它透過自然語言處理 (NLP) 技術，從使用者輸入的履歷文本中提取關鍵技能，將其與預定義的職位資料庫進行向量化比對，從而提供職位匹配度分析、技能雷達圖，以及詳細的技能差距 (Skill Gap) 報告。

## 2. 系統架構與核心元件

本系統採用「混合NLP模型」架構，結合了 SpaCy 的**提取能力**與 Scikit-learn 的**編碼能力**。

### 2.1 核心技術棧

| **元件** | **技術** | **目的** |
| --- | --- | --- |
| Web 框架 | Streamlit | 提供使用者互動介面 (UI) |
| 技能提取 | SpaCy (PhraseMatcher) | 從原始文本中解析、正規化並提取技能 |
| 技能編碼 | Scikit-learn (CountVectorizer) | 將提取的技能列表轉換為用於比對的向量 |
| 相似度計算 | Scikit-learn (cosine\_similarity) | 計算使用者向量與職位向量的相似度 |
| 資料庫 | Python Dictionaries | 儲存 SKILLS\_DB, JOBS\_DB, COURSES\_DB |
| 視覺化 | Plotly | 繪製技能雷達圖 |

### 2.2 AI 核心邏輯詳解

本系統的 AI 流程分為兩個獨立但銜接的階段：

1. **階段一：技能提取 (Extraction) - SpaCy**
   * **任務：** 處理非結構化的 resume\_text。
   * **機制：** 使用 spacy.load("en\_core\_web\_sm") 載入模型，並透過 PhraseMatcher 進行高效率的詞組比對。
   * **優勢：** 取代了傳統的 NORMALIZATION\_MAP。PhraseMatcher 能同時處理同義詞 (如 "sklearn", "sci-kit learn" 皆指向 scikit-learn) 與多詞詞組 (如 "adobe creative suite")。
   * **輸出：** 一個標準化的技能集合 (Set)，例如 {'python', 'react', 'scikit-learn'}。
2. **階段二：技能編碼 (Encoding) - CountVectorizer**
   * **任務：** 將 SpaCy 提取的技能集合轉換為可供 cosine\_similarity 計算的向量。
   * **機制：** CountVectorizer 在**系統啟動時**便已使用 SKILLS\_DB 作為詞彙表，並預先計算好了所有 job\_vectors。
   * **橋接：** 為了比對，我們必須將 SpaCy 的輸出 (Set) 轉回一個 CountVectorizer 能理解的字串 (" ".join(user\_found\_skills))，然後呼叫 vectorizer.transform()。
   * **輸出：** 一個與 job\_vectors 維度一致的稀疏向量 (Sparse Vector)。

## 3. 部署指南 (Streamlit Community Cloud)

將此應用程式部署到 Streamlit Community Cloud 需要精確的環境配置。

### 3.1 先決條件

* 一個 GitHub 帳戶。
* 一個 Streamlit Community Cloud 帳戶。
* 一個包含 app.py、requirements.txt 和 runtime.txt 的 GitHub 儲存庫。

### 3.2 關鍵設定檔案

部署的成敗 100% 取決於 requirements.txt 和 runtime.txt 這兩個檔案的內容。它們必須位於儲存庫的**根目錄**。

#### 3.2.1 runtime.txt (Python 版本)

此檔案用於強制 Streamlit Cloud 使用特定的 Python 版本。這是解決套件不相容問題的**第一步**。

Plaintext

python-3.11

* **理由：** Python 3.11 是目前（2025年）支援 spacy, pandas, 和 scikit-learn 預編譯二進位檔案最穩定的版本。使用 3.13+ 會導致 spacy 安裝失敗；使用 3.8- 可能會導致其他套件版本過舊。

#### 3.2.2 requirements.txt (套件依賴)

此檔案是部署中最關鍵、也最容易出錯的部分。**順序**和**特定版本**至關重要。

Plaintext

# 1. 強制指定 NumPy 版本, 解決二進位衝突  
numpy==1.26.4  
  
# 2. 核心應用程式與資料科學套件  
streamlit  
pandas  
scikit-learn  
plotly  
  
# 3. SpaCy 核心與模型  
# 必須指定特定版本, 並使用 URL 安裝模型  
spacy==3.7.4  
https://github.com/explosion/spacy-models/releases/download/en\_core\_web\_sm-3.7.1/en\_core\_web\_sm-3.7.1-py3-none-any.whl

## 4. 關鍵注意事項：套件版本管理

在維護此應用程式時，必須深刻理解為何 requirements.txt 必須如此設定。

### 4.1 錯誤根源：Numpy 二進位不相容

* **症狀：** 部署成功，但執行 import spacy 時應用程式崩潰，顯示 ValueError: numpy.dtype size changed, may indicate binary incompatibility。
* **原因：** pandas, scikit-learn, 和 spacy (及其底層 thinc) 都是依賴 numpy 的 C 語言編譯的。如果 pip 在安裝過程中，為它們安裝了**不同**的 numpy 二進位版本，它們在記憶體中載入時就會發生衝突。
* **解決方案：** 如 requirements.txt 所示，我們在**檔案最頂端**強制 numpy==1.26.4。這會讓 pip 在安裝 pandas 和 spacy 時，自動尋找與 numpy 1.26.4 相容的**單一**版本，從而消除衝突。

### 4.2 錯誤根源：SpaCy 模型的安裝

* **症狀：** 部署在安裝依賴時失敗，顯示 ERROR: No matching distribution found for en\_core\_web\_sm。
* **原因：** en\_core\_web\_sm **不是一個 PyPI 套件**，pip install en\_core\_web\_sm 永遠會失敗。
* **解決方案：** SpaCy 模型必須透過其**完整的 .whl 檔案 URL** 來安裝。spacy==3.7.4 和 en\_core\_web\_sm-3.7.1 是已知相容的組合。

### 4.3 錯誤根源：Pandas 編譯卡住

* **症狀：** 部署日誌顯示正在安裝 pandas-xxx.tar.gz，然後卡住或RAM超時。
* **原因：** 未指定 Python 版本 (或版本太新/舊)，導致 pip 找不到預編譯的 .whl 檔案，因而試圖下載原始碼 (.tar.gz) 並在 1GB RAM 的伺服器上**即時編譯**，這幾乎註定失敗。
* **解決方案：** 使用 runtime.txt 指定 python-3.11，確保 pip 能找到 pandas-xxx-cp311-manylinux.whl 這種預編譯檔案。

## 5. 維護與擴充

* **新增技能：**
  1. 在 SKILLS\_BY\_CATEGORY 字典中的相應列表添加標準化的小寫技能。
  2. SKILLS\_DB 會自動更新。
  3. （重要）vectorizer 會自動將新技能納入詞彙表。
* **新增同義詞 (Alias)：**
  + 在 app.py 的 3.2. SpaCy (Extractor Role) 區塊，找到 patterns 字典。
  + 為相應的技能（如 patterns["scikit-learn"]）添加新的 nlp.make\_doc("new alias")。
* **新增職位：**
  + 在 JOBS\_DB 字典中新增條目，確保其技能字串中的所有技能都已在 SKILLS\_DB 中定義。