**CV Management System Documentation**  
**Version:** 1.0  
**Date:** February 22, 2025

### ****Table of Contents****

1. Introduction
2. System Overview
3. Features
4. System Architecture
   * 4.1 Flutter Frontend
   * 4.2 Firestore Backend
   * 4.3 **Python Integration for AI-Powered Text Extraction**
5. Detailed Functionalities
   * 5.1 Data Searching and Filtering
   * 5.2 CV Status Management
   * 5.3 **CV Upload & AI-Driven Text Extraction Process**
   * 5.4 Charts & Data Visualization
6. Conclusion

## ****1. Introduction****

The CV Management System is a desktop application built with Flutter and powered by Firestore, targeting Windows platforms. It is designed to manage and streamline the handling of CVs (resumes) by providing advanced filtering options, status management (archiving/assigning), and insightful data visualization through multiple charts. The application leverages **Python for AI-powered text extraction**, which processes uploaded CV files and integrates the extracted JSON data into Firestore via Dart.

## ****2. System Overview****

The application assists HR professionals or recruitment teams by:

* Allowing advanced search filters (languages, education, certifications, and skills).
* Enabling easy management of CVs (marking as archived or assigned).
* **Automating AI-driven data extraction from CV files.**
* Displaying statistical and analytical charts to better understand candidate data.

## ****3. Features****

### ****Advanced Filtering:****

Search CVs by various parameters, including languages, education background, certifications, and skills.

### ****CV Status Management:****

Archive CVs or mark them as “assigned” to track progress and manage candidate pipelines.

### ****AI-Powered CV Upload and Text Extraction:****

* Upload CV files (PDF, Word, Images, TXT, and more).
* Python extracts text content and processes it using **Google Generative AI** to extract structured data and headers.
* The extracted data is formatted as JSON and uploaded to Firestore via Dart.

### ****Data Visualization:****

Multiple charts, such as:

* Certifications Chart
* Education Timeline Chart
* Languages Chart
* Project Chart

## ****4. System Architecture****

### ****4.1 Flutter Frontend****

* **Platform:** Windows
* **UI Framework:** Flutter  
  The Flutter framework is used to create a responsive and intuitive user interface. It handles all interactions related to filtering, viewing charts, and status updates for the CVs.

### ****4.2 Firestore Backend****

* **Database:** Cloud Firestore  
  Firestore is employed as the backend database, ensuring real-time synchronization of data across the application. It stores detailed CV information along with metadata such as status flags (archived, assigned).

### ****4.3 Python Integration for AI-Powered Text Extraction****

#### **Text Extraction Pipeline:**

A **Python script** is responsible for:

* Accepting an uploaded CV file (PDF, Word, Image, TXT, etc.).
* Extracting textual content using **OCR and NLP techniques** if necessary.
* Sending the extracted text to **Google Generative AI** for data structuring.
* Receiving a **JSON response** with categorized headers (e.g., Name, Experience, Education, Skills).
* Uploading the structured JSON data to Firestore via Dart integration.

#### **Integration with Dart & Firestore:**

* The extracted data is processed in Dart.
* Firestore stores the structured data, making it **searchable and analyzable** within the application.

## ****5. Detailed Functionalities****

### ****5.1 Data Searching and Filtering****

#### **Search Parameters:**

The application allows users to filter CVs based on:

* Languages known by the candidate.
* Educational background.
* Certifications obtained.
* Specific skills.

#### **Implementation:**

The filtering mechanism interacts with Firestore’s querying capabilities, returning a dynamic list of CVs based on user-specified criteria.

### ****5.2 CV Status Management****

#### **Status Flags:**

Users can:

* **Archive CVs:** Mark CVs that are no longer actively considered.
* **Assign CVs:** Indicate that a CV has been processed or is assigned to a recruiter.

#### **Purpose:**

This allows for better organization and tracking within the recruitment workflow.

### ****5.3 CV Upload & AI-Driven Text Extraction Process****

#### **Uploading a CV:**

* Users select the CV file from their local machine.
* The selected file is passed to the integrated **Python AI pipeline**.

#### **AI-Powered Text Extraction:**

1. **Python script processes the file** to extract raw textual content.
2. Extracted text is **sent to Google Generative AI** for structured data extraction.
3. AI returns a **JSON object** with categorized sections such as:
   * Candidate Name
   * Contact Information
   * Work Experience
   * Education
   * Skills & Certifications
4. JSON data is **uploaded to Firestore** via Dart, ensuring the CV data is searchable and analyzable.

### ****5.4 Charts & Data Visualization****

#### **Types of Charts:**

* **Languages Chart:** Highlights the language proficiencies of the candidate pool.
* **Categories Chart**: Highlights the categories and how many times they exist in the CVs

#### **Implementation:**

Charts are implemented using Flutter’s charting libraries, fetching data in real time from Firestore.

## ****6. Conclusion****

The CV Management System leverages modern technologies—**Flutter, Firestore, and AI-powered Python processing**—to streamline CV processing, enable advanced filtering, and provide insightful visual analytics. This documentation provides a comprehensive overview of the system’s features, architecture, and operational guidelines, ensuring a smooth setup and deployment experience.