Doc Search Reader

Developed by: Rohith Vishnu Kumar

Documented by: Rohith Vishnu Kumar

Contact: rohithvishnuk@gmail.com

Phone: 8247650717 (+91)

This is a Tool which takes documents of (.pdf, .docx, .xlsx, .jpg, .jpeg, .png) formats. It scans the documents (even there are images inside the documents) and extracts the text and allows the user to serach for Keywords (Multiple keywords seperated by commas) at a single time. It no only searches for exact word but also searches for Synonyms related to that keyword. It also have an advanced Search Capability where user can extensively search inside the document with prompt and even ask questions related to the document all in real time. This also provides Spell checking and also have a summarise option where it gives a concise summary of the document and the summarised document is made available for download.

Features:

1. Multi-format Document Support:
   PDF: Extracts text from PDFs, including those with embedded images. DOCX: Reads and processes text from Word documents. XLSX: Extracts text from Excel sheets. Image Files: Utilizes OCR (Optical Character Recognition) to extract text from images in JPG, JPEG, and PNG formats.
2. Text Extraction and Search:
   Keyword Search: Allows users to search for multiple keywords

at once. Synonym Search: Searches for synonyms of the provided keywords to enhance search results. Advanced Search: Enables extensive search within documents using prompts and real-time questions.

3. Summarization:
   Document Summarization: Provides a concise summary of the document. Downloadable Summary: Summarized content is available for download in a user-friendly format**.**

Outline:

The user first goes to root directory (/) if there is previous active session running then it is redirected to the home page else it is redirected to the login page. If the user is new he can go the signup page (Signup Link available on login page). All the data of the field are passed to the */signup* route where entire data is requested by a Flask package request. The password is hashed using *passlib* module and stored in a database. Next it redirects to Login Page if the credentials are correct then it takes to the Multifactor Authentication page. There it sends a mail to the registered mail which is used in signup if the OTP is correct it redirects to the home page. There you will find a upload file button and user can submit his document of accepted formats (.pdf, .docx, .xlsx, .jpg, .jpeg, .png) If user does not upload any document or uploads other than this format an error page is displayed. Upon Successful Uploading of document the extracted text along with text inside images. Entire text is displayed where mistaken spelling are displayed with squiggles If the user clicks on the squiggles it provides suggestions on the right control panel. The user can select desired spelling and it immediately corrects in the text displayed and updates the document while summarizing. This Tool has two types of Searches :

1) Basic Search: This takes single or multiple keywords as input and searches for the keywords and highlights keywords along with their synonyms (if exist). The user can also search by medium sized sentences but this matches as is pattern of the sentence.
2) Advanced Search: This is quite advanced and capable of processing natural language (although Basic search also uses NLP but up to a limited level). User can not only search but also he can ask questions related to the document, It does not go with pattern to pattern matching rather it analyses the document and processes it and answer any type of questions along with their belongings in the exact document. (Company can tweak their sensitivity as per their requirements prompts are provided in advancedsearch.js) Uses google Gemini API

Finally if the user clicks submit button it takes the updated document (spell check) and connects to google Gemini API.

There the document is summarized and the user is redirected to the records page where they can find all their documents which are extracted from the database.

Libraries used:

import shutil ***(used for deleting directories)***

from threading import Timer ***(Used for time delay)***

from reportlab.lib.pagesizes import A4 ***(used for converting text to pdf)***

from reportlab.pdfgen import canvas ***(used for converting text to pdf)***

from flask_mysqldb import MySQL ***(DB)***

import os ,random ***(os for file managing, random for generating otps)***

from flask import Flask, render_template, redirect, request, send_file, session ***(Default flask imports)***

 # from flask_session import Session ***(For managing web session)***

from flask_mail import Mail, Message ***(For sending mails MFA)***

from werkzeug.utils import secure_filename ***(For securing filename to remove harmful characters from the file name (sanitizing the file name))***

import fitz # PyMuPDF ***(For extracting text and images from PDF)***

from docx import Document ***(For extracting text and images from Word doc)***

import openpyxl ***(For extracting text and images from Excel)***

import pytesseract ***(for OCR)***

from googletrans import Translator ***(For Traslator)***

from PIL import Image ***(For image manipulation)***

import io ***(for working with streams of data)***

import re ***(For regular expressions)***

from openpyxl.drawing.image import Image as OpenpyxlImage

import spacy ***(For NLP tasks)***

import nltk ***(For NLP tasks)***

import pkg_resources

from symspellpy import SymSpell, Verbosity ***(For spell check)***

import string ***(For advanced string operations)***

from passlib.hash import pbkdf2_sha256 ***(for hashing password)***

import google.generativeai as genai ***(Google API)***

nltk.download("wordnet")

from nltk.corpus import wordnet

nlp = spacy.load("en_core_web_md") # pip install en_core_web_md

from dotenv import load_dotenv

Scope for V2 :

1. **Includes adding CSS loaders**
2. **Strengthen Server side validation for signup page**
3. **Perfect Preserving of document layout and styles**