resume

```
import json
import os
import io
from langchain_google_genai import ChatGoogleGenerativeAI
import time

from File_reader import extract_text_from_pdf
from File_reader import extract_text_from_ppt
from File_reader import extract_text_from_docx
from resume_analyser import analyze_resume
from resume_analyser import extract_and_remove_contact_info
from file_select import select_file
from Excel_data import add_data_to_excel_bef
from main_NLP import analyze_resume_nltk
#from ex1 import analyze_resume

api_key = os.getenv("GEMINI_API_KEY")
print(api_key)
script_dir=os.path.dirname(__file__)
llm = ChatGoogleGenerativeAI(model='gemini-1.5-flash', google_api_key=api_key)

start_time = time.time()
folder_path=os.path.join(script_dir,"Resumes")
files=select_file(folder_path)
files=files.split(',')
file_list=[]
for name in files:
    fl=os.path.join(folder_path, name)
    file_list.append(fl)

filepath = os.path.join(script_dir, 'Job_description.txt')
filepath="C:\\foxtrot\\Resume analysis\\support files\\Job_Description.txt"
try:
    with open(filepath, 'r', encoding='utf-8') as txt: # Using utf-8 encoding is generally recommended
        jd_content = txt.read()
except FileNotFoundError:
    print(f"Error: File not found at {filepath}")
except Exception as e:
    print(f"An error occurred: {e} the file: {filepath}")

for file in file_list:
    filename = os.path.basename(file)
    filename = os.path.splitext(filename)[0]
    if file.endswith(".pdf"):
        try:
            with open(file, 'rb') as pdf_file: # Open the file in binary read mode.
                pdf_bytes = pdf_file.read()
                pdf_bytes_io = io.BytesIO(pdf_bytes) # Create a BytesIO object.
                res_txt = extract_text_from_pdf(pdf_bytes_io)
        except FileNotFoundError:
            print(f"Error: File not found at {file}")
        except Exception as e:
            print(f"An unexpected error occurred: {e} the file: {file}")
    elif file.endswith(".pptx"):
        try:
            with open(file, 'rb') as ppt_file: # Open the file in binary read mode.
                ppt_bytes = ppt_file.read()
                ppt_bytes_io = io.BytesIO(ppt_bytes) # Create a BytesIO object.
                res_txt = extract_text_from_ppt(ppt_bytes_io)

        except FileNotFoundError:
            print(f"Error: File not found at {file}")
        except Exception as e:
            print(f"An unexpected error occurred: {e} the file: {file}")
    elif file.endswith(".docx") or file.endswith(".doc"):
        try:
            with open(file, 'rb') as doc_file: # Open the file in binary read mode.
                doc_bytes = doc_file.read()
                doc_bytes_io = io.BytesIO(doc_bytes) # Create a BytesIO object.
                res_txt = extract_text_from_docx(doc_bytes_io,file)

        except FileNotFoundError:
            print(f"Error: File not found at {file}")
        except Exception as e:
            print(f"An unexpected error occurred: {e} the file: {file}")

    #res=analyze_resume(res_txt,jd_content,filename)

    #print(json.dumps(res,indent=4))

    #print(analyze_resume_nltk(res_txt, jd_content, filename))

    mnumb,email,res_txt=extract_and_remove_contact_info(res_txt)
    mnumb='Contact Number is not present' if mnumb[0]=='N' else mnumb[0]
    email='Email ID is not present' if email == 'None' else email[0]



    try:
        res=analyze_resume(res_txt,jd_content,api_key)
        #print(res)
        remarks= [res['strengths'], res['areas_for_improvement'], res['matching_skills'],
res['missing_skills'],
                      res['summary'], res['match_score'], res['recommendation']]

        jtitl=res['Job title']
        cname=res['Candidate Name']
        exp=res['Experience']
        pri_skills=res['Primary Skills']
        op="C:\\foxtrot\\Resume analysis\\Result.xlsx"
        add_data_to_excel_bef(op,filename,jtitl,cname,email,mnumb,exp, pri_skills,remarks)

    except Exception as e:
        print(f"An unexpected error occurred: {e} the file: {file}")


end_time=time.time()
print(print(f"the amount of execution time it takes for the complete resumne analysis:
{end_time-start_time}"))
```