In [1]:
```
import re#RegEx library used to check if a string contains the specified search
import spacy#SpaCy library for Advance Natural Language Processing(NLP)
from pdfminer.high_level import extract_text as extract_pdf_text#For extracting text from PDF
from docx import Document#For text extraction from DOCX
import json#Returns output in JSON format
```

In [2]:
```
# Load the SpaCy English language model for natural language processing
nlp = spacy.load("en_core_web_sm")
```

In [3]:
```
# Define a function that reads text from PDF
def extract_text_from_pdf(file_path):
    return extract_pdf_text(file_path)

# Define a function that reads text from DOCX
def extract_text_from_docx(file_path):
    doc = Document(file_path)
    return "\n".join([para.text for para in doc.paragraphs])

# Define a function to detremine the file type and extract text
def extract_text(file_path):
    if file_path.endswith(".pdf"):
        return extract_text_from_pdf(file_path)
    elif file_path.endswith(".docx"):
        return extract_text_from_docx(file_path)
    else:
        raise ValueError("Unsupported file format. Kindly upload PDF or DOCX file")#Handles error if the file extens
```

In [4]:
```
# Function to extract name using SpaCy Name Entity Recognition(NER) features
def extract_name(text):
    doc = nlp(text)
    for ent in doc.ents:
        if ent.label_ == "PERSON":
            return ent.text
        return None


# Function to extract e-mail using regex library to find matching pattern as defined
def extract_email(text):
    email_pattern = r'[a-zA-Z0-9._%+-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]+'
```

```
    match = re.search(email_pattern,text)
    return match.group() if match else None

# Function to extract phone using regex library to find matching pattern as defined
def extract_phone(text):
    phone_pattern = r'\+?\d[\d\s\-\(\)]{8,}\d'
    match = re.search(phone_pattern, text)
    return match.group() if match else None

# Function to extract skills using regex library to find matching pattern as defined
def extract_skills(text):
    skills_list = ["python","java","r","sql","machine learning", #List of possible skill set for a Data Scientist
                   "deep learning","pandas","numpy",
                   "keras","pytorch","aws sagemaker",
                   "cloud computing","tensor flow"
                  ]
    skills_found = [skill for skill in skills_list if skill in text.lower()]#Loops through the file and searches for
    return list(set(skills_found))#Returns a list

# Function to extract education using regex library to find matching pattern as defined
def extract_education(text):
    doc = nlp(text)
    education = []#An empty list to store the result of education key-words from the file
    # Common keywords indicating education section
    education_keywords = ["education", "academic", "university",
                          "college", "degree", "bachelor",
                          "master","phd","mba",
                          "b.sc","btech","mtech","certification","certified"
                         ]
    education_pattern = r'(?i)(bachelor|master|phd|diploma)\s*(?:of|in)?\s*([^\n,]+)'

    for sent in doc.sents:
        sent_text = sent.text
        if any(keyword.lower() in sent_text.lower() for keyword in education_keywords):
            match = re.search(education_pattern, sent_text)
            if match:
                degree = match.group(1)
                field = match.group(2).strip()
                # Look for organization (e.g., university) in the same sentence
                org = None
                for ent in sent.ents:
                    if ent.label_ == "ORG":
```

```
                        org = ent.text
                        break
                education.append({
                    "degree": degree,
                    "field": field,
                    "institution": org if org else "Unknown"
                })
    return education
```

In [5]:
```
# Function to parse resume
def parse_resume(file_path):
    text = extract_text(file_path)
    parsed_data = {"name": extract_name(text),
                   "email": extract_email(text),
                   "phone number": extract_phone(text),
                   "skills set": extract_skills(text),
                   "education": extract_education(text)}
    return json.dumps(parsed_data, indent=2)
```

In [6]:
```
if __name__ == "__main__":
    file_path = "C:/Users/Sanayak/Desktop/VICTOR ITINAH INIOBONG Resume.docx"
    print(parse_resume(file_path))
```

```
{
  "name": "VICTOR",
  "email": "viniobong37@gmail.com",
  "phone number": "2349068652505",
  "skills set": [
    "r",
    "pandas",
    "sql",
    "numpy",
    "deep learning",
    "pytorch",
    "machine learning",
    "python"
  ],
  "education": [
    {
      "degree": "Bachelor",
      "field": "Science",
      "institution": "Federal University Lafia"
    }
  ]
}
```


In [ ]: