+ Code + Text

Extracting Data

Choosing the Book: Since my birth month is July (7), I will use Book 7 as per the given instructions.

Extracting `file1.txt` : My birthdate is July 7, so I will extract 10 pages starting from page 7 of Book 7 and save them in `file1.txt` .

Extracting `file2.txt` : My birth year is 2002, which corresponds to page 102. I will extract 10 pages starting from page 102 of Book 7 and save them in `file2.txt` .

```
!pip install pyspellchecker
!pip install PyPDF2
!pip install fpdf
```

```
Requirement already satisfied: pyspellchecker in /usr/local/lib/python3.11/dist-packages (0.8.2)
Requirement already satisfied: PyPDF2 in /usr/local/lib/python3.11/dist-packages (3.0.1)
Requirement already satisfied: fpdf in /usr/local/lib/python3.11/dist-packages (1.7.2)
```

```
from PyPDF2 import PdfReader  # extract text from PDFs
import re  # handle text processing with regular expressions
import pandas as pd  # data manipulation and storage
from collections import Counter  # count word occurrences
from spellchecker import SpellChecker  # detect misspelled or non-English words
from fpdf import FPDF  # create PDF reports
import matplotlib.pyplot as plt  # generate visualizations
```

```
# file paths
PDF_FILE = "/content/Harry_Potter_(www.ztcprep.com).pdf"
OUTPUT_FILE1 = "file1.txt"
OUTPUT_FILE2 = "file2.txt"

# birth details for book and page selection  July 7 2002
BIRTH_MONTH, BIRTH_DATE , BIRTH_YEAR  = 7, 7, 2002
BOOK_ID = 7
START_PAGE1 = BIRTH_DATE   #  10 pages i.e 7-16
START_PAGE2 = 102  # 10 pages i.e 102-111

from PyPDF2 import PdfReader

def extract_pages(pdf_path, start_page, num_pages=10):
    reader = PdfReader(pdf_path)
    return "\n".join(reader.pages[p - 1].extract_text() for p in range(start_page, start_page + num_pages) if p <= len(rea

#extract and save text
with open(OUTPUT_FILE1, "w", encoding="utf-8") as f1, open(OUTPUT_FILE2, "w", encoding="utf-8") as f2:
    f1.write(extract_pages(PDF_FILE, START_PAGE1))
    f2.write(extract_pages(PDF_FILE, START_PAGE2))

print(f"Text extraction complete: {OUTPUT_FILE1}, {OUTPUT_FILE2}")
```

```
Text extraction complete: file1.txt, file2.txt
```

Q1: Write Python code and use MapReduct to count occurrences of each word in the first text file (file.txt). How many times each word is repeated?

```
TEXT_FILE = "/content/file1.txt"
OUTPUT_CSV = "word_count.csv"

def extract_words(text):
    return re.findall(r'\b\w+\b', text.lower())

#read text from file
with open(TEXT_FILE, "r", encoding="utf-8") as file:
```

```
        content = file.read()

    #count word occurrences
    word_freq = Counter(extract_words(content))

    # convert to DataFrame and sort
    df = pd.DataFrame(word_freq.items(), columns=["Word", "Count"]).sort_values(by="Count", ascending=False)

    # saving results to CSV fiiles
    df.to_csv(OUTPUT_CSV, index=False)

    print("\nWord Frequency Analysis from file1.txt:")
    print(df.to_string(index=False))
```

```
          free      1
     perfectly      1
       however      1
     nighttime      1
       mouthed      1
       parking      1
          open      1
         gazed      1
       pointed      1
         broad      1
      swooping      1
       watched      1
        mirror      1
          hadn      1
         floor      1
         ninth      1
         calls      1
       shouted      1
           bit      1
     lunchtime      1
         words      1
        caught      1
           bag      1
      doughnut      1
     clutching      1
           way      1
           tin      1
        single      1
           too      1
         bunch      1
       angrily      1
          eyed      1
         baker      1
         group      1
        gotten      1
        bakery      1
           bun      1
           buy      1
          walk      1
          legs      1
       stretch      1
       reading      1
     grunnings      1
      behavior      1
         shake      1
        huddle      1
         wheel      1
      steering      1
       fingers      1
       drummed      1
       fashion      1
      supposed      1
        getups      1
       clothes      1
        little      1
     strangely      1
       arrived      1
        seemed      1
```

Q2: From the second text file (file2.txt), write Python code and use MapReduct to count how many times non-English words (names, places, spells etc.) were used. List those words and how many times each was repeated.

```
FILE_PATH = "/content/file2.txt"
OUTPUT_FILE = "non_english_words.csv"

#initializing spell checker
spell_checker = SpellChecker()

def get_words(text):
    return re.findall(r'\b\w+\b', text.lower())

#read text from file
with open(FILE_PATH, "r", encoding="utf-8") as file:
    content = file.read()

#extract words and filter non english words
words = get_words(content)
non_english = [word for word in words if word not in spell_checker]

#count occurrences
word_counts = Counter(non_english)

#convert to DataFrame and save
df = pd.DataFrame(word_counts.items(), columns=["Non-English Word", "Count"]).sort_values(by="Count", ascending=False)
df.to_csv(OUTPUT_FILE, index=False)

print("\nIdentified Non-English Words from file2.txt:")
print(df.to_string(index=False))
```

```
Identified Non-English Words from file2.txt:
Non-English Word  Count
          hagrid     29
             ter     23
             yeh     13
             www     10
         ztcprep     10
              ll      7
        gringotts      7
            didn      6
           ernon      5
              ap      3
            stuf      3
              ve      3
           izards      2
              eah      2
            hadn      2
            knuts      2
            albus      2
             wasn      2
           gettin      2
           wouldn      1
               mm      1
              teh      1
               69      1
               64      1
               70      1
             cept      1
        deliverin      1
        everythin      1
           pposed      1
        mentionin      1
               71      1
          guardin      1
          fetchin      1
               66      1
            payin      1
          shouldn      1
           muggle      1
             goin      1
          dumbled      1
             ying      1
            insul      1
               65      1
          speakin      1
               68      1
             aren      1
```

```
meself      1
    ou      1
    67      1
diagon      1
  ther      1
 tryin      1
```