To extract the content from the provided Harry potter pdf file.

PyMuPDF (aka "fitz") is the Python bindings for MuPDF. using the start marker and end marker, we iterate through the pdf pages and check for the starting marker and if the starting marker is found, add the page text to text_to_extract variable and break the text extraction if the end_marker has found.

DATE OF BIRTH : 06/15/2001

In [1]:
```
import fitz

def extract_text_between_markers(pdf_path, start_marker, end_marker, output_file_path)
    try:

        pdf_document = fitz.open(pdf_path)

        # Initialize an empty text variable to store extracted text
        text_to_extract = ""

        # To indicate the start marker has found
        found_start = False

        #Iterating through the text and extracting the required content
        for page_number in range(pdf_document.page_count):
            page = pdf_document.load_page(page_number)
            page_text = page.get_text()

            if start_marker in page_text:
                found_start = True

            if found_start:
                text_to_extract += page_text

            if end_marker in page_text:
                break

        #  Removing the content before start and end marker
        start_index = text_to_extract.find(start_marker)
        end_index = text_to_extract.find(end_marker)

        if start_index != -1 and end_index != -1:
            text_to_extract = text_to_extract[start_index + len(start_marker):end_inde

        # Saving to a txt file
        with open(output_file_path, 'w', encoding='utf-8') as output_file:
            output_file.write(text_to_extract)

        print(f"Extracted text between '{start_marker}' and '{end_marker}' and saved t

    except Exception as e:
        print(f"An error occurred: {e}")
    finally:
        pdf_document.close()


pdf_file_path = 'Harry_Potter_(www.ztcprep.com).pdf'
```

```
start_marker = 'P a g e | 15\nHarry Potter and the Half Blood Prince – J.K. Rowling'
end_marker = 'P a g e | 25\nwww.ztcprep.com\nHarry Potter and the Half Blood Prince –
output_file_path = 'file1.txt'

# Extract text between markers
extract_text_between_markers(pdf_file_path, start_marker, end_marker, output_file_path
```

```
Extracted text between 'P a g e | 15
Harry Potter and the Half Blood Prince – J.K. Rowling' and 'P a g e | 25
www.ztcprep.com
Harry Potter and the Half Blood Prince – J.K. Rowling' and saved to 'file1.txt'
```

In [2]:

```
import fitz

def extract_text_between_markers(pdf_path, start_marker, end_marker, output_file_path)
    try:
        # Create a PDF document object
        pdf_document = fitz.open(pdf_path)

        # Initialize an empty text variable to store extracted text
        text_to_extract = ""

        # To indicate the start marker has found
        found_start = False

        # Iterate through the pages and extract the required content
        for page_number in range(pdf_document.page_count):
            page = pdf_document.load_page(page_number)
            page_text = page.get_text()

            if start_marker in page_text:
                found_start = True

            if found_start:
                text_to_extract += page_text

            if end_marker in page_text:
                break

        # Removing the content before start and end marker
        start_index = text_to_extract.find(start_marker)
        end_index = text_to_extract.find(end_marker)

        if start_index != -1 and end_index != -1:
            text_to_extract = text_to_extract[start_index + len(start_marker):end_inde

        # Saving to a txt file
        with open(output_file_path, 'w', encoding='utf-8') as output_file:
            output_file.write(text_to_extract)

        print(f"Extracted text between '{start_marker}' and '{end_marker}' and saved t

    except Exception as e:
        print(f"An error occurred: {e}")
    finally:
        pdf_document.close()


pdf_file_path = 'Harry_Potter_(www.ztcprep.com).pdf'
start_marker = 'P a g e | 101\nHarry Potter and the Half Blood Prince – J.K. Rowling'
```

```
end_marker = 'P a g e | 111\nHarry Potter and the Half Blood Prince – J.K. Rowling'
output_file_path = 'file2.txt'

# Extract text between markers
extract_text_between_markers(pdf_file_path, start_marker, end_marker, output_file_path)
```

```
Extracted text between 'P a g e | 101
Harry Potter and the Half Blood Prince – J.K. Rowling' and 'P a g e | 111
Harry Potter and the Half Blood Prince – J.K. Rowling' and saved to 'file2.txt'
```

MAP_REDUCE using pyenchant

Here we are using enchant and importing the dictionary to verify the words in our text file are legitimate english words. Then convert all words to lowercase and remove the punctuation marks and any other special characters from the text file so that it does'nt create any confusion while checking if words are valid english words.

In [3]:
```
pip install pyenchant
```

```
Requirement already satisfied: pyenchant in c:\users\ashri\anaconda3\lib\site-package
s (3.2.2)Note: you may need to restart the kernel to use updated packages.
```

In [4]:
```
import enchant

# Initialize the English dictionary
d = enchant.Dict("en_US")

# Read the txt file
with open('file1.txt', 'r', encoding='utf-8') as file:
    text = file.read()

# Splitting the text into words
words = text.split()

# Counting the occurrences of each word
word_counts = {}
for word in words:
    word = word.strip(".,!?()[]{}\"'")  # Remove punctuation
    word = word.lower()  # Convert to lowercase
    if word and d.check(word):  # Check if it's a valid English word
        if word in word_counts:
            word_counts[word] += 1
        else:
            word_counts[word] = 1

# Printing the word count
for word, count in word_counts.items():
    print(f'{word}: {count}')
```

bones: 2
head: 1
of: 56
the: 152
department: 1
magical: 2
law: 2
enforcement: 1
we: 7
think: 3
he-who-must-not-be-: 1
named: 3
may: 1
have: 3
murdered: 1
her: 22
in: 33
person: 2
because: 2
she: 15
was: 27
a: 84
very: 7
gifted: 1
witch: 1
and: 54
all: 9
evidence: 1
that: 23
put: 3
up: 11
real: 1
fudge: 14
cleared: 1
his: 18
throat: 1
with: 18
an: 9
effort: 1
it: 13
seemed: 5
stopped: 1
spinning: 2
bowler: 1
hat: 1
murder: 1
said: 29
prime: 30
minister: 29
momentarily: 3
diverted: 1
from: 13
anger: 1
newspapers: 1
just: 7
middle-aged: 1
woman: 4
who: 8
lived: 1
alone: 2

nasty: 1
killing: 1
wasn’t: 4
it’s: 3
had: 15
rather: 6
lot: 1
publicity: 1
police: 1
are: 2
baffled: 1
you: 20
sighed: 1
course: 2
they: 7
he: 21
room: 3
locked: 1
inside: 1
on: 9
other: 6
hand: 3
know: 1
exactly: 1
did: 3
not: 7
gets: 3
us: 1
any: 2
further: 1
toward: 2
catching: 3
him: 11
then: 5
there: 7
maybe: 3
didn’t: 1
hear: 2
about: 3
one: 3
yes: 2
i: 16
happened: 1
around: 6
corner: 2
here: 6
as: 16
matter: 1
fact: 1
papers: 1
field: 1
day: 2
order: 1
minister’s: 3
backyard: 1
if: 5
barely: 3
listening: 1
to: 40
got: 4

swarming: 1
over: 6
place: 2
attacking: 1
people: 2
left: 1
right: 3
center: 1
once: 2
upon: 3
happier: 1
time: 4
this: 9
sentence: 1
would: 1
been: 7
unintelligible: 1
but: 13
wiser: 1
now: 4
thought: 6
guard: 1
prisoners: 1
cautiously: 1
p: 9
g: 9
e: 9
16: 1
harry: 9
potter: 9
half: 9
blood: 9
prince: 9
wearily: 1
anymore: 1
they’ve: 1
deserted: 2
prison: 1
joined: 1
he-who-must-: 1
not-be-named: 1
won’t: 3
pretend: 1
sense: 1
dawning: 1
horror: 1
tell: 1
me: 4
they’re: 2
creatures: 2
drain: 1
hope: 1
happiness: 1
out: 6
breeding: 1
that’s: 4
what’s: 1
causing: 1
sank: 1
weak-kneed: 1

into: 7
nearest: 1
chair: 1
idea: 1
invisible: 1
swooping: 1
through: 5
towns: 1
countryside: 1
spreading: 1
despair: 1
hopelessness: 1
voters: 1
made: 2
feel: 1
quite: 1
faint: 2
see: 1
you’ve: 1
do: 6
something: 1
your: 7
responsibility: 1
dear: 1
can’t: 3
honestly: 1
still: 4
magic: 3
after: 4
sacked: 1
three: 2
days: 1
ago: 1
whole: 2
community: 2
has: 6
screaming: 1
for: 15
my: 5
resignation: 1
fortnight: 2
never: 1
known: 1
them: 6
so: 8
united: 1
term: 1
brave: 1
attempt: 1
at: 14
smile: 4
lost: 1
words: 2
despite: 1
indignation: 1
position: 1
which: 4
placed: 1
felt: 1
shrunken-: 1

looking: 3
man: 6
sitting: 1
opposite: 1
finally: 1
there’s: 2
anything: 1
can: 4
kind: 2
is: 6
nothing: 2
sent: 1
tonight: 1
bring: 1
date: 1
recent: 1
events: 1
introduce: 1
17: 1
successor: 1
he’d: 2
be: 11
by: 4
he’s: 3
busy: 3
moment: 3
much: 1
going: 1
looked: 5
portrait: 3
ugly: 1
little: 2
wearing: 1
long: 6
curly: 1
silver: 1
wig: 1
digging: 1
ear: 1
point: 1
quill: 1
fudge’s: 1
eye: 1
finishing: 1
letter: 1
wish: 1
sounding: 1
bitter: 1
first: 5
writing: 1
twice: 2
past: 1
budge: 1
prepared: 1
persuade: 1
boy: 1
might: 1
well: 5
will: 2
more: 1

subsided: 1
what: 2
clearly: 2
aggrieved: 1
silence: 1
broken: 2
almost: 3
immediately: 2
suddenly: 1
spoke: 1
its: 5
crisp: 1
official: 1
voice: 4
muggles: 2
requesting: 1
meeting: 1
urgent: 1
kindly: 2
respond: 1
distractedly: 1
flinched: 1
flames: 1
grate: 1
turned: 5
emerald: 1
green: 4
again: 2
rose: 1
revealed: 1
second: 5
wizard: 1
their: 3
heart: 1
disgorging: 1
moments: 2
later: 1
onto: 1
antique: 1
rug: 1
feet: 1
moment’s: 1
hesitation: 1
same: 1
watching: 1
new: 3
arrival: 1
straighten: 1
dust: 1
down: 6
black: 4
robes: 2
look: 3
foolish: 1
like: 3
old: 3
lion: 1
18: 1
were: 6
streaks: 1

gray: 1
mane: 1
tawny: 1
hair: 3
bushy: 1
keen: 1
yellowish: 1
eyes: 5
behind: 3
pair: 2
wire-rimmed: 1
spectacles: 1
certain: 1
rangy: 1
loping: 1
grace: 1
even: 1
though: 3
walked: 1
slight: 1
limp: 1
immediate: 1
impression: 1
shrewdness: 1
understood: 1
why: 1
preferred: 1
leader: 1
these: 1
dangerous: 1
times: 1
politely: 1
holding: 2
grasped: 1
briefly: 1
scanning: 1
pulled: 1
wand: 5
under: 5
told: 2
asked: 3
striding: 1
door: 5
tapping: 1
keyhole: 1
heard: 2
lock: 1
click: 1
don’t: 1
mind: 1
remained: 1
shortly: 1
added: 1
pointing: 1
windows: 4
curtains: 3
swept: 1
across: 3
let’s: 1
get: 1

business: 1
need: 1
discuss: 1
drew: 2
himself: 1
fullest: 1
height: 1
replied: 1
am: 2
perfectly: 1
happy: 2
security: 1
already: 6
thank: 1
we’re: 1
cut: 1
poor: 1
lookout: 1
curse: 1
secretary: 1
outer: 1
office: 2
19: 1
getting: 1
rid: 1
you’re: 2
hotly: 1
highly: 2
efficient: 1
work: 2
rest: 2
without: 1
flicker: 1
trained: 1
assigned: 1
wait: 1
declared: 1
decide: 1
works: 1
coldly: 1
say: 2
no: 3
problem: 1
continues: 1
er: 1
lamely: 1
junior: 1
continued: 1
entertaining: 1
public: 1
impersonating: 1
reacted: 1
poorly: 1
performed: 1
addled: 1
brains: 1
could: 2
only: 1
weakly: 1
bit: 1

go: 3
easy: 1
drink: 1
20: 1
team: 1
healers: 1
st: 1
hospital: 1
maladies: 1
injuries: 1
examining: 1
speak: 3
far: 1
attempted: 2
strangle: 1
best: 1
remove: 1
muggle: 2
society: 1
he’ll: 1
anxiously: 1
merely: 3
shrugged: 1
moving: 1
back: 6
fireplace: 1
really: 2
keep: 1
posted: 1
developments: 1
or: 1
least: 1
shall: 2
probably: 1
too: 3
come: 1
personally: 1
case: 2
send: 1
consented: 1
stay: 1
advisory: 1
toothache: 1
rummaging: 1
pocket: 1
mysterious: 1
powder: 1
fire: 2
gazed: 1
hopelessly: 1
fought: 1
suppress: 1
evening: 1
burst: 1
last: 3
heaven’s: 1
sake: 1
wizards: 2
surely: 1
sort: 1

slowly: 1
spot: 1
exchanged: 1
incredulous: 1
manage: 1
trouble: 1
side: 3
two: 2
stepped: 1
bright: 1
vanished: 1
21: 1
spinner’s: 2
end: 2
many: 1
miles: 1
away: 2
chilly: 1
mist: 1
pressed: 1
against: 1
drifted: 1
dirty: 2
river: 4
wound: 1
between: 3
overgrown: 1
rubbish-: 1
strewn: 1
banks: 1
immense: 1
chimney: 2
relic: 1
disused: 1
mill: 2
reared: 1
shadowy: 1
ominous: 1
sound: 1
apart: 2
whisper: 2
water: 1
sign: 1
life: 1
scrawny: 1
fox: 5
slunk: 1
bank: 4
nose: 1
hopefully: 1
some: 2
fish-and-chip: 1
wrappings: 1
tall: 1
grass: 2
pop: 2
slim: 1
hooded: 2
figure: 4
appeared: 1

thin: 1
air: 1
edge: 1
froze: 1
wary: 1
fixed: 1
strange: 1
phenomenon: 1
take: 1
bearings: 1
few: 2
set: 2
off: 1
light: 7
quick: 1
strides: 1
cloak: 3
rustling: 1
louder: 1
another: 3
materialized: 1
22: 1
harsh: 1
cry: 1
startled: 1
crouching: 1
flat: 1
undergrowth: 1
leapt: 1
hiding: 1
flash: 3
yelp: 1
fell: 2
ground: 1
dead: 1
animal: 1
toe: 1
woman’s: 1
dismissively: 1
hood: 3
perhaps: 1
quarry: 1
paused: 1
scrambling: 1
fallen: 1
listen: 2
caught: 3
seized: 1
arm: 3
wrenched: 1
must: 3
listened: 1
decision: 1
leave: 1
gained: 1
top: 1
where: 2
line: 1
railings: 2
separated: 1

narrow: 1
cobbled: 1
street: 3
followed: 3
stood: 2
road: 2
rows: 2
dilapidated: 1
brick: 2
houses: 3
dull: 1
blind: 1
darkness: 2
lives: 1
contempt: 1
dunghill: 1
our: 1
ever: 1
foot: 1
23: 1
slipped: 1
gap: 1
rusty: 1
hurrying: 1
streaming: 2
saw: 1
darting: 1
alley: 1
identical: 1
streetlamps: 1
women: 1
running: 1
patches: 1
deep: 1
pursuer: 2
prey: 1
succeeding: 1
hold: 1
swinging: 1
faced: 1
each: 1
trust: 1
dark: 3
lord: 2
trusts: 1
doesn’t: 1
believe: 1
panted: 1
gleamed: 1
check: 1
indeed: 1
plan: 1
anyone: 1
betrayal: 1
lord’s: 1
snarled: 1
beneath: 1
threateningly: 1
other’s: 1
face: 2

laughed: 1
own: 1
sister: 2
wouldn’t: 2
breathed: 1
note: 1
hysteria: 1
brought: 1
knife: 1
let: 1
sister’s: 1
burned: 1
24: 1
rushed: 1
ahead: 1
rubbing: 1
keeping: 1
distance: 1
moved: 1
deeper: 1
labyrinth: 1
hurried: 1
towering: 1
hover: 1
giant: 1
admonitory: 1
finger: 1
footsteps: 1
echoed: 1
cobbles: 1
passed: 1
boarded: 1
until: 1
reached: 1
house: 1
dim: 1
glimmered: 1
downstairs: 1
knocked: 1
before: 1
cursing: 1
breath: 1
together: 1
waiting: 1
panting: 1
slightly: 1
breathing: 1
smell: 1
carried: 1
night: 1
breeze: 1
seconds: 1
movement: 1
opened: 1
crack: 1
sliver: 1
seen: 1
parted: 1
sallow: 1
threw: 1

```
pale: 1
shine: 1
blonde: 1
gave: 1
drowned: 1
opening: 1
wider: 1
pleasant: 1
strained: 1
```

MAP_REDUCE to find NON-ENGLISH WORDS

Similar to the previous step, we're going to import US dictionary and cross-check all the words in our txt file to map out all the non-english words and count them accordingly.

Here we created a function to check if the word are valid english words. we then split the text file into wordsand call the function to check. Then we initialize a dictionary named non_english_word_counts to count the no.of occurences of that non-english word.

We iterated through each word that we splitted from txt file and passed it through the fucntion to check and count those words that are non-english and print the result.

In [5]:
```python
import enchant
import re

# Initializing the English dictionary
d = enchant.Dict("en_US")

# Function to check if the word is non-English
def is_non_english(word):
    return not d.check(word)

# Reading the txt file
with open('file2.txt', 'r', encoding='utf-8') as file:
    text = file.read()

# splitting the text into words
words = re.findall(r'\b\w+\b', text)

# Count non-English words and their occurrences
non_english_word_counts = {}
for word in words:
    word = word.lower()  # Converting to lowercase
    if is_non_english(word):
        if word in non_english_word_counts:
            non_english_word_counts[word] += 1
        else:
            non_english_word_counts[word] = 1

# Printing the non-English word count
for word, count in non_english_word_counts.items():
    print(f'{word}: {count}')
```

```
ron: 28
www: 16
ztcprep: 16
hermione: 28
arry: 4
eet: 2
mrs: 10
weasley: 11
fleur: 8
delacour: 1
im: 1
seester: 1
gabrielle: 1
rowling: 9
hadn: 2
ze: 1
ard: 1
gringotts: 1
eenglish: 1
zere: 1
isn: 3
tchah: 1
ve: 5
tonks: 6
auror: 1
triwizard: 1
hasn: 2
sirius: 4
azkaban: 1
bellatrix: 2
lestrange: 1
wasn: 2
couldn: 1
lupin: 1
didn: 3
doesn: 1
fred: 3
george: 3
diagon: 1
percy: 1
voldemort: 3
dumbledore: 7
lucius: 1
malfoy: 1
weren: 1
wouldn: 2
countercurses: 1
```

In [ ]: