Words Wallet

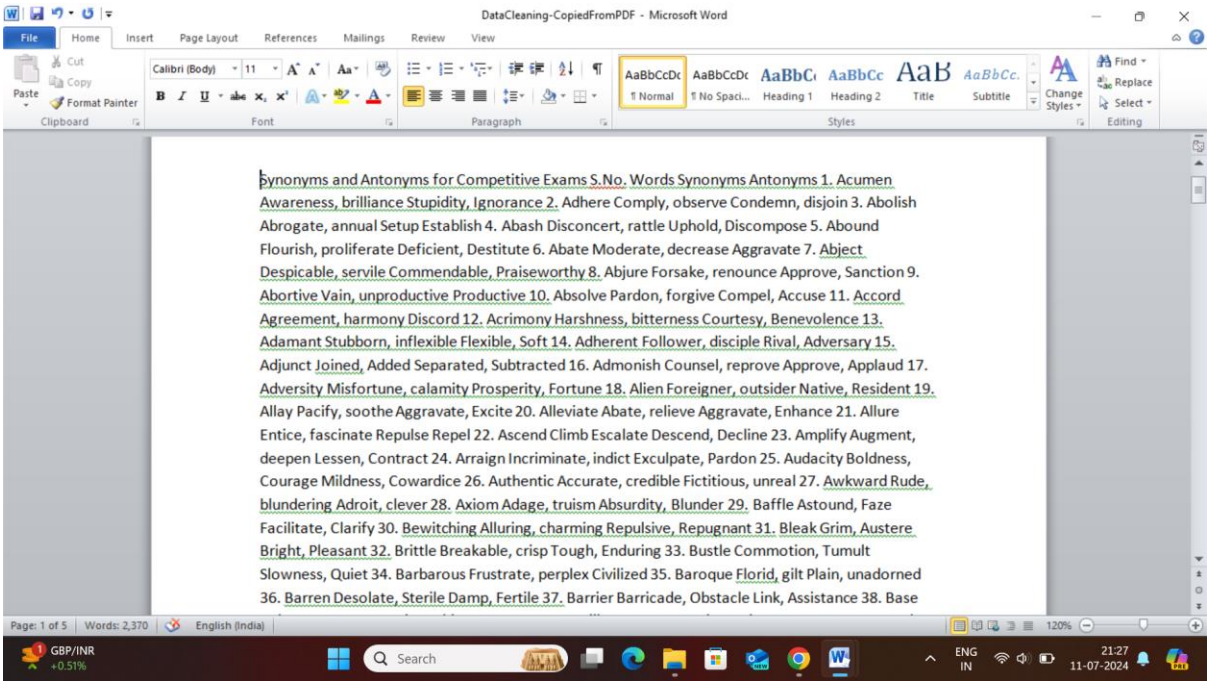
- ***Creating a Dataset of Words***

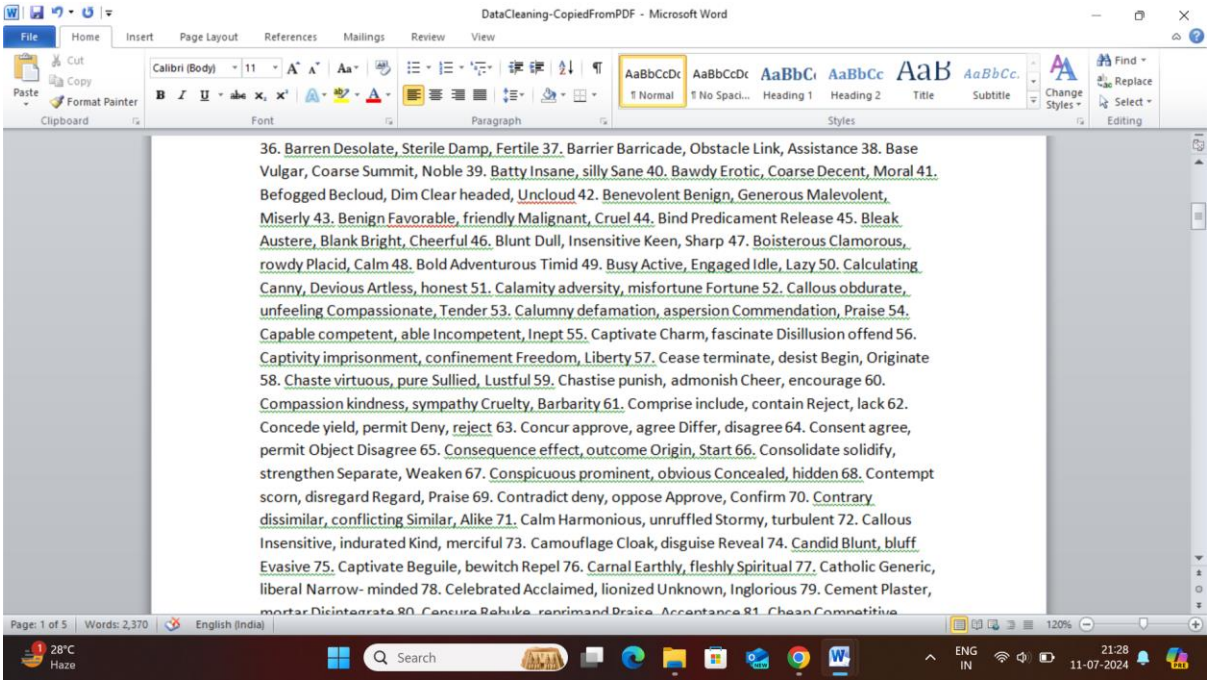
Date: 11th Jul, 2024
Team Size: 1

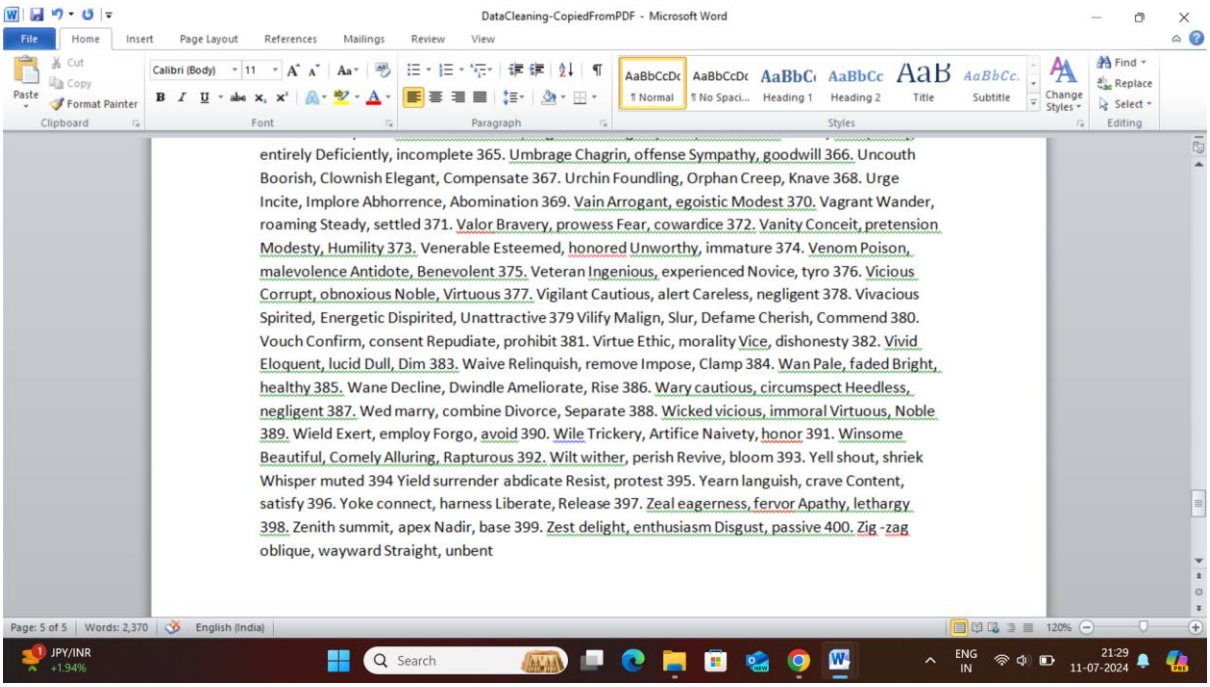
This project was done to create a dataset(.csv file) of words, for competitive exams, and their synonyms that can later be used for mini projects like online vocabulary builder, etc.

The data cleaning process followed for this project is as follows:

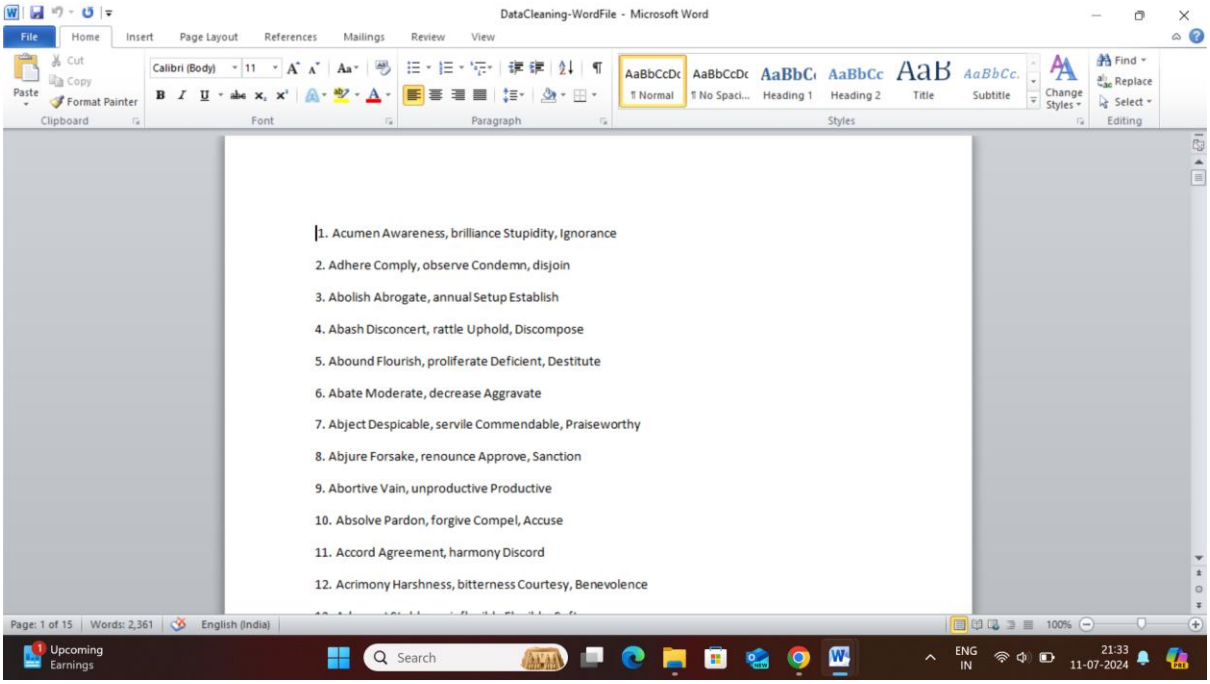
1. A list of words, along with their synonyms and antonyms, was downloaded from internet as a PDF file.
2. The content of the file was manually copied from PDF to a word file using the following commands –
    a. Ctrl + A to Select the entire text (from PDF)
    b. Ctrl + C to copy the selected text (from PDF)
    c. Ctrl + V to paste the copied text (from Word)
3. Some screenshots of the word file with raw text are as follows

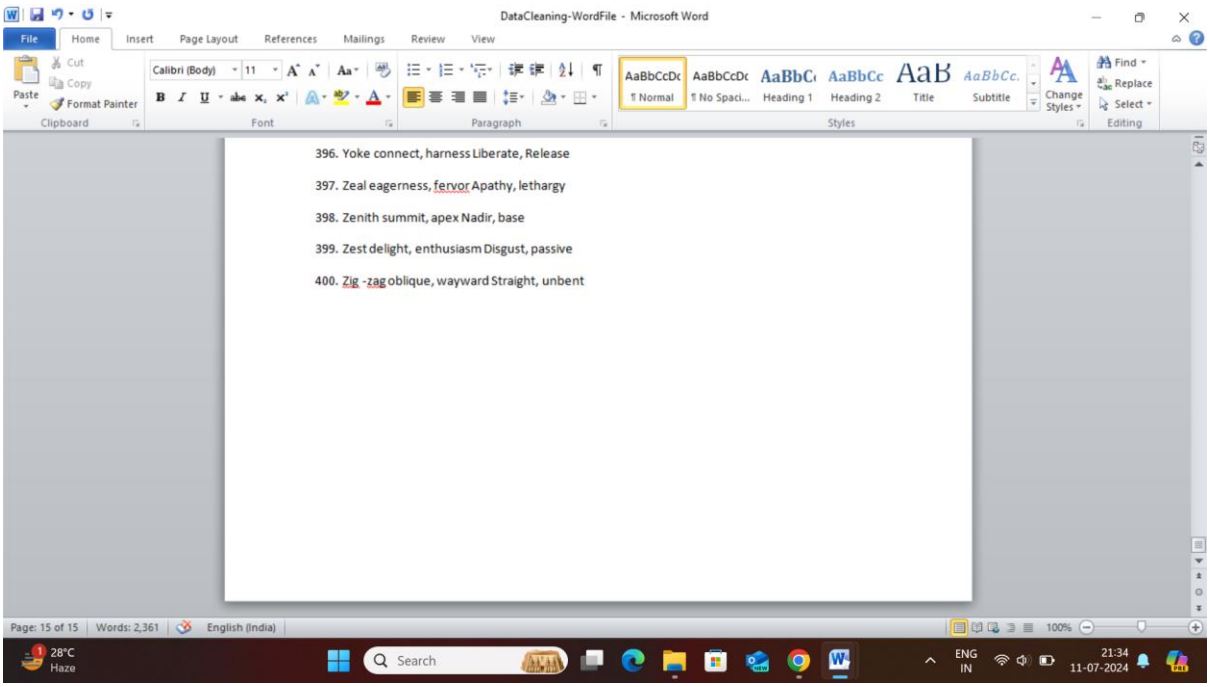




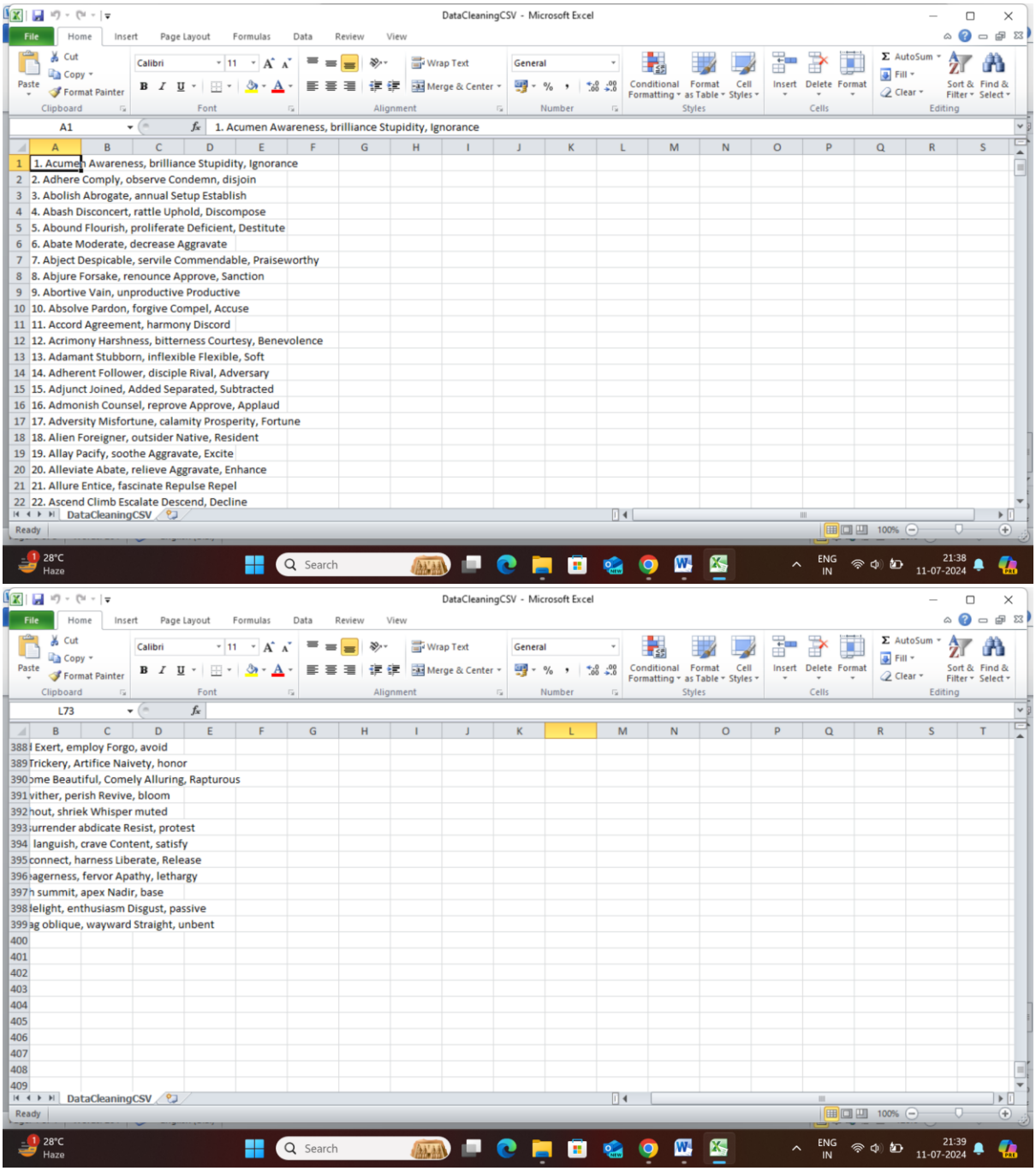


4. This raw text was manually edited to separate the sets of each word and its synonyms and antonyms along with the s.no. Commas were inserted separating the actual word, the set of synonyms and the set of antonyms. Some of the screenshots of the edited file are as follows:





5. This edited and properly arranged data was copied to a spreadsheet in Excel and saved as a .csv file. Some screenshots of the .csv file.



6. Data in this file was cleaned in many ways using various python functions
    a. The file was opened and read as a pandas dataframe

```
import pandas as pd
import numpy as np
```

```
words = pd.read_csv('/content/SynonymsAntonyms.csv',names=['Words'])
```

```
words.head(2)
```

	Words
0	1. Acumen Awareness, brilliance Stupidity, Ig...
1	2. Adhere Comply, observe Condemn, disjoin

```
def separater(row):
  for i in str(row).split(','):
      return i
```

b. The words were separated using the str.split() function.

```
words['Words'].apply(lambda row: str(row).split(' '))
```

```
0      [, 1., Acumen, Awareness,, brilliance, Stupidi...
1      [2., Adhere, Comply,, observe, Condemn,, disjoin]
2      [3., Abolish, Abrogate,, annual, Setup, Establ...
3      [4., Abash, Disconcert,, rattle, Uphold,, Disc...
4      [5., Abound, Flourish,, proliferate, Deficient...
                             ...
394    [396., Yoke, connect,, harness, Liberate,, Rel...
395    [397., Zeal, eagerness,, fervor, Apathy,, leth...
396          [398., Zenith, summit,, apex, Nadir,, base]
397    [399., Zest, delight,, enthusiasm, Disgust,, p...
398    [400., Zig, -zag, oblique,, wayward, Straight,...
Name: Words, Length: 399, dtype: object
```

c. Three lists for words, their synonyms and antonyms were created and the separated words, stored in series, were copied to these lists. And these lists were assigned as columns to a new dataframe.

87]

```
words = []
synonyms = []
antonyms = []
for i in split_words:
  words.append(i[1])
  synonyms.append(i[2] + i[3])
  antonyms.append(i[-1] + i[-2])
```

92]

```
new_df = pd.DataFrame()
new_df['Words'] = words
new_df['Synonyms'] = synonyms
new_df['Antonyms'] = antonyms
```

93]

```
new_df
```

	Words	Synonyms	Antonyms
0	1.	AcumenAwareness,	IgnoranceStupidity,
1	Adhere	Comply,observe	disjoinCondemn,

d. The data in **Synonyms** column in the dataframe had two synonyms as a single string. Similarly, the data in **Antonyms** column had two synonyms as a single string.
The Synonyms column data was separated using the str.split() function. The separated data was stored in two separated lists, Synonym1 and Synonym2.
Two new columns were created in the dataframe for Synonym1 and Synnonym2 and the data in the two lists was copied.
The very first value in the dataframe was a number, so it was manually assigned the actual word with the help of domain knowledge.

```
new_df['Synonym1'][0] = 'Accumen'
new_df['Synonym2'][0] = 'Awareness'
new_df
```

	Words	Synonyms	Antonyms	Synonym1	Synonym2
0	Acumen	AcumenAwareness,	IgnoranceStupidity,	Accumen	Awareness
1	Adhere	Comply,observe	disjoinCondemn,	Comply	observe
2	Abolish	Abrogate,annual	EstablishSetup	Abrogate	annual
3	Abash	Disconcert,rattle	DiscomposeUphold,	Disconcert	rattle
4	Abound	Flourish,proliferate	DestituteDeficient,	Flourish	proliferate
...	...	...	...	...	...
394	Yoke	connect,harness	ReleaseLiberate,	connect	harness
395	Zeal	eagerness,fervor	lethargyApathy,	eagerness	fervor
396	Zenith	summit,apex	baseNadir,	summit	apex

396	Zenith	summit,apex	baseNadir,	summit	apex
397	Zest	delight,enthusiasm	passiveDisgust,	delight	enthusiasm
398	Zig	-zagoblique,	unbentStraight,	-zagoblique	

399 rows × 5 columns

```
20]
synonyms = new_df['Synonyms'].apply(lambda row: str(row).split(','))
```

```
58]
Synonym1 = []
Synonym2 = []

for i in synonyms:
  Synonym1.append(i[0])
  Synonym2.append(i[-1])
```

```
59]
new_df['Synonym1'] = Synonym1
new_df['Synonym2'] = Synonym2
new_df
```

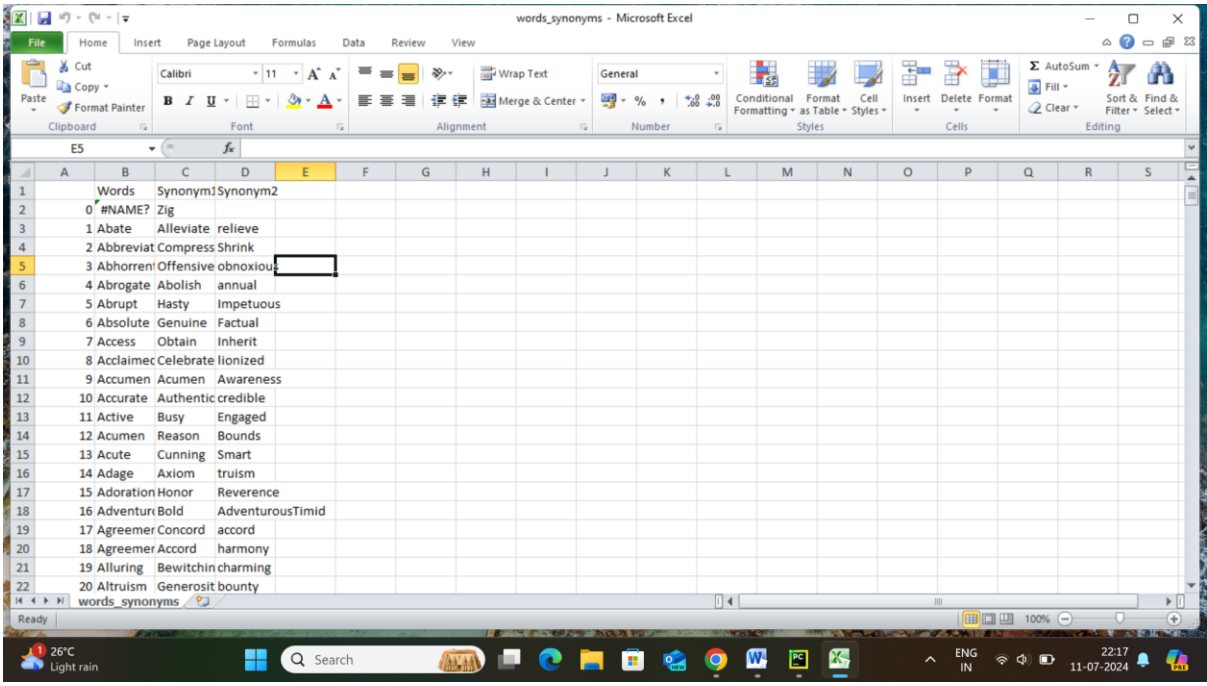
e. The unwanted columns were dropped using the df.drop() method. Only three columns were retained – Word, Synonym1 and Synonym2 in the dataframe.

0	Acumen	Awareness	Accumen
1	Adhere	observe	Comply
2	Abolish	annual	Abrogate
3	Abash	rattle	Disconcert
4	Abound	proliferate	Flourish
...	...	...	...
394	Yoke	harness	connect
395	Zeal	fervor	eagerness
396	Zenith	apex	summit

f. Thif final dataframe was saved a .csv file using the df.to_csv() method.

```
words_syno.to_csv('words_synonyms.csv')
```

g. Some of the screenshots of the newly created csv file are as follows.



h. This file was later used as a dataset in creating an API for synonyms for competitive exams. I plan to extend this dataset by including antonyms in the future.

Note:

The steps listed above are not exhaustive. Only the most important functions have been mentioned.

Project Done and
Report Created by

Mrs Taranum Begum

Machine Learning Enthusiast