To do:

* Create function string documentation
* Add os package to dependencies
* Update requirements.txt once PDF packages are finalized
* Generate program functional diagram

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Welcome to Outlook Analyzer!

This document assumes that you are interested in learning more about the technical aspects of this Python package. The guide also assumes you have a basic understanding of Python programming.

# Package Dependencies

Outlook Analyzer relies on several packages. Its current version relies on the packages below:

* [**pywin32 (“win32com.client”)**](https://pypi.org/project/pywin32/)

This package allows the program access to standard Windows APIs, enabling feature like email data extraction from the Windows Outlook desktop client.

* [**tqdm**](https://pypi.org/project/tqdm/)

This package provides the program’s progress bars as displayed when it is extracting email and task information.

* [**pandas**](https://pandas.pydata.org/)

This package provides data tables and frame used for file reading, metric organization, and display.

* [**tabulate**](https://pypi.org/project/tabulate/)

This package enables the program to easily print tables as outputs.

* [**matplotlib (“matplotlib.pyplot”)**](https://matplotlib.org/)

This package enables the program to generate analysis plot derived from extracted Outlook information.

* [**dataframe\_image**](https://pypi.org/project/dataframe-image/)

This package enables the program to generate image files from generated data frames.

* [**re**](https://docs.python.org/3/library/re.html)

This package simplifies the process of identifying character patterns during word cloud generation, specifically when removing links from email text bodies.

* [**time**](https://docs.python.org/3/library/time.html)

This package provides time-related functions and formatting.

* [**datetime**](https://docs.python.org/3/library/datetime.html)

This package provides classes by which the program can store and manipulate time-related data.

* [**dateutil**](https://pypi.org/project/python-dateutil/)(“dateutil.relativedelta”)

This package extends the standard datetime package so that the program can represent relative time intervals.

* [**sys**](https://docs.python.org/3/library/sys.html)

This package enables the program to identify and work with Python interpreter-specific variables like function names.

# Function Descriptions

## append\_to\_error\_list(function\_name, error\_text)

## extract\_outlook\_information(max\_email\_number\_to\_extract\_input,date\_start\_input,date\_end\_input)

## word\_cloud\_extract(messages)

## unread\_senders\_data\_gen(unread\_senders\_raw\_list,unread\_senders\_unique\_dict,sender\_data\_file)

## generate\_unread\_senders\_viz()

## generic\_email\_data\_gen(messages\_list, email\_data\_file)

## category\_data\_gen(category\_list,categories\_data\_file)

## generate\_count\_viz(counting\_dict, date\_start\_str, date\_end\_str)

## generate\_categories\_viz()

## cleanup(inp)

## generate\_generic\_viz(flagged\_counter\_int,email\_data\_file,email\_image\_file,title)

## word\_cloud\_content\_clean()

## word\_cloud\_generate()

## word\_cloud\_display()

## unique (list1)

## main()

# Understanding Script Variable Naming Standards

* Camel case is used when required for MS object model variables
* UPPERCASE is used for global variables
* lower\_case with an underscore between words for variables used in the script and adding the data type to the end of the variables (dict, string, int, etc.)