# pubmed_tool User Manual

`pubmed_tool` is a comprehensive Python package that facilitates scraping records from PubMed based on a keyword and date-range query, uploading of the result data to an SQL database, and visualization of trends in publication over time.

## Requirements

`pubmed_tool` was developed using Python 3.10. Earlier versions have not yet been tested for compatibility in deployment.

`pubmed_tool` was designed to install and otherwise enforce minimum dependency versions. However, this process may fail. In addition to os, logging, sqlite3, and re from the Python Standard Library, `pubmed_tool` requires:

- Requests version 2.31.0 or greater
- Pandas version 1.5.3 or greater
- Numpy version 1.23.1 or greater
- Biopython version 1.81 or greater, with Bio module version 1.6.0 or greater
- Bokeh version 3.3.1 or greater
- Matplotlib version 3.5.2 or greater
- Holoviews version 1.18.1 or greater
- Hvplot version 0.9.0 or greater
- Panels version 1.3.1 or greater

## Installation and Import

`pubmed_tool` can be installed from the github repository using pip:

```
pip install pubmed_tool@git+https://github.com/intro-to-ds-capstone/capstone-project
```

Alternatively, all files within the /pubmed_tool folder found at:

https://github.com/intro-to-ds-capstone/capstone-project

may be copied to a project directory, which will enable further modification of the source files to meet the needs of an individual project.

Standardized import is achieved with:

```
import pubmed_tool
```

`pubmed_tool` may be used in a python script. There is extended functionality for use in interactive environments such as a Jupyter notebook.

## Primary Functions

### Scraper

Scraping PubMed for records based on a keyword query and date range is achievable with a single function: `pubmed_tool.scraper()`

```
pubmed_tool.scraper(keyword, start_date, end_date , email, max_returns = 200000,
      chunksize = None, return_df = False, path = 'publications.csv',
      project_dir = None, overwrite = True)
```

The following minimum fields are required:

> `keyword (string)`: The desired search term in the query
>
> `start_date (string, datetime)`: The start date for the query. May be either a string in 'YYYY/MM/DD' format, or a datetime object. There is some validation of the date performed by `pubmed_tool.validators.date()`
>
> `end_date (string, datetime)`: The end date for the query, which must be chronologically after `start_date`. May be either a string in 'YYYY/MM/DD' format, or a datetime object. There is some validation of the date performed by `pubmed_tool.validators.date()`
>
> `email (string)`: The email for the PubMed query. This is required by PubMed's Entrez system to log access attempts. Some validation is performed by `pubmed_tool.validators.email()`

Additional options are included with default values, that allow for greater customization of functionality.

> `max_returns (integer)`: The maximum number of records to return from a single query. This is beneficial for large queries. The default is `200000`. If set to `None`, the BioPython default of `20` is used.
>
> `chunksize (integer)`: The total number of records to process at one time, if batch-processing is desired. This is beneficial for large queries, which may take significant memory. The default is `None`, which will attempt to process all records at once.
>
> `return_df (Boolean)`: A toggle indicating if it is desired for the function to return the results as a pandas data frame. The default is `False`. If `return_df = True` and `chunksize != None` is not set to None, a validation check will trigger a warning that chunk-processing and returning the data frame are incompatible, and set `return_df = False`.
>
> `path (path)`: The path for saving the output CSV, if this export is desired. May be an absolute or a relative path. The default is `'publications.csv'`. If `path` is either `None` or given as a relative path (rather than an absolute path), the `project_dir`

will be set to the current working directory by validation with `pubmed_tool.validators.path()`, requiring a '.txt' or '.csv' file extension.

`project_dir (path)`: The path of the project directory. The default is `None`. If `path` is either `None` or given as a relative path (rather than an absolute path), `project_dir` will be set to the current working directory by validation with `pubmed_tool.validators.path()`

`overwrite (Boolean)`: A toggle indicating if it is desired to overwrite the file at the target `path`, if `path` already exists. The default is `True`. If `overwrite = False` and the file at `path` already exists, a warning will stop processing in `pubmed_tool.validators.path()`

The `pubmed_tool.scraper()` function is dependent on several subfunctions within the `pubmed_tool.scr` module, which are not likely to be called outside of the main `pubmed_tool.scraper()` function. However, these sub-functions may be useful for debugging or otherwise modifying the functionality to serve an individual project's purpose.

## SQL Upload and Query

Uploading the results of a scrape from `pubmed_tool.scraper()` and querying the results by an author's name is achievable with a single function: `pubmed_tool.sql_full()`

```
pubmed_tool.sql_full(t_df, project_dir = None, db_name = 'publications.db',
             paper_name = 'papers', authors_name = 'authors',
             pairs_name = 'pairs_authorpapers', any_nm = None, first_nm = None,
             last_nm = None, initials_nm = None)
```

Queries are performed with OR comprehension, and the result is returned as a pandas DataFrame. There is no sanitization or other checks of inputs to SQL at this time, and thus this function is vulnerable to facilitating malicious SQL injection.

The following minimum fields are required:

`t_df (path, DataFrame)`: Either the path to a CSV file generated by `pubmed_tool.scraper()` or the output of `pubmed_tool.scraper()` desired for processing, SQL upload, and SQL query.

Additional options are included with default values, that allow for greater customization of functionality.

`project_dir (path)`: The path of the project directory. The default is `None`. If `path` is either `None` or given as a relative path (rather than an absolute path), `project_dir` will be set to the current working directory by validation with `pubmed_tool.validators.path()`

`db_name (string)`: The path of the SQLite database file. May be an absolute or a relative path. The default is `'publications.db'`. If `db_name` is given as a relative

path (rather than an absolute path), the `project_dir` will be set to the current working directory by validation with `pubmed_tool.validators.path()`, requiring a '.db' file extension.

`paper_name (string)`: The name of the table in the database indicated by `db_name` to store paper-specific data. The default is `'papers'`. If this table already exists in the database, it will be deleted and entirely overwritten by the new data.

`authors_name (string)`: The name of the table in the database indicated by `db_name` to store author-specific data. The default is `'authors'`. If this table already exists in the database, it will be deleted and entirely overwritten by the new data.

`pairs_name (string)`: The name of the table in the database indicated by `db_name` to store author-paper pair data. The default is `'pairs_authorpapers'`. If this table already exists in the database, it will be deleted and entirely overwritten by the new data.

`any_nm (string)`: A name to query from the SQL data base, searching for partial matches in any field of author name. The default is `None`, which omits this section of the query.

`first_nm (string)`: A name to query from the SQL data base, searching for partial matches in the first name field of author name only. The default is `None`, which omits this section of the query.

`last_nm (string)`: A name to query from the SQL data base, searching for partial matches in the last name field of author name only. The default is `None`, which omits this section of the query.

`initials_nm (string)`: A name to query from the SQL data base, searching for partial matches in the initials field of author name only. The default is `None`, which omits this section of the query.

The `pubmed_tool.sql_full()` function is dependent on several subfunctions within the `pubmed_tool.sql` module, many of which are not as likely to be called outside of the main `pubmed_tool.sql_full()` function. However, these sub-functions may be useful for debugging or otherwise modifying the functionality to serve an individual project's purpose.

### *SQL Query*

The primary function within the `pubmed_tool.sql` module that may be called outside of the `pubmed_tool.sql_full()` function is `pubmed_tool.sql.query()`:

```
pubmed_tool.sql.query(db_name = 'publications.db', project_dir = None,
              paper_name = 'papers', authors_name = 'authors',
              pairs_name = 'pairs_authorpapers', any_nm = None, last_nm = None,
```

```
                first_nm = None, initials_nm = None)
```

The inputs to this function are the same as those contained in the overall wrapper of `pubmed_tool.sql_full()`. However, this function may be useful when the user wishes to query an existing database.

## Visualization

The visualization of the results of a scrape from `pubmed_tool.scraper()` is achievable with a single function: `pubmed_tool.full_visual()`

To limit dependencies, the visualizer outputs to HTML files. Otherwise, Selenium (Firefox and Geckodriver or Chrome and Chromedriver) and PhantomJS would be required, which are not entirely available as pip installations. Bokeh plots may be saved from their interactive forms in the HTML, which renders locally with all required data embedded.

Testing was performed using Jupyter and Firefox on a standard 14.4 inch 2400 x 1600 resolution display.

```
pubmed_tool.full_visual(t_df, out_path = 'visual.html', project_dir = None,
                  mode = 'html', port = 5007, interactive = False,
                  keyword = None, start_date = None, end_date = None,
                  logo_path = None, primary_color = 'blue',
                  secondary_color = 'grey', accent_color = 'grey')
```

The following minimum fields are required:

> `t_df (path, DataFrame)`: Either the path to a CSV file generated by `pubmed_tool.scraper()` or the output of `pubmed_tool.scraper()` desired for visualization.

Additional options are included with default values, that allow for greater customization of functionality.

> `out_path (path)`: The path for saving the visualizer output HTML, if this export is desired. May be an absolute or a relative path. The default is `'visual.html'`. If `out_path` is either `None` or given as a relative path (rather than an absolute path), the `project_dir` will be set to the current working directory by validation with `pubmed_tool.validators.path()`, requiring an '.html' file extension.
>
> `project_dir (path)`: The path of the project directory. The default is `None`. If `out_path` is either `None` or given as a relative path (rather than an absolute path), `project_dir` will be set to the current working directory by validation with `pubmed_tool.validators.path()`
>
> `mode (string)`: A toggle with options of `'html'`, `'jupyter'`, or `'port'`.

'html': generates the output as an html file for export. If out_path is None, a warning is generated and mode is set to 'port'

'jupyter': returns the output for display in a Jupyter Notebook

'port': opens a port at the port number set by option port for temporary local hosting of the visualization as a webpage

port (integer): The port for use with mode = 'port'. The default is 5007.

chunksize (integer): The total number of records to process at one time, if batch-processing is desired. This is beneficial for large queries, which may take significant memory. The default is None, which will attempt to process all records at once.

interactive (Boolean): A toggle indicating if it is desired for the visualization to be interactive or not. The default is False.

keyword (string): The search term used in the query, if desired to use in naming in the visualization output. The default is None.

start_date (string, datetime): The start date used in the query, if desired to use in naming in the visualization output. The default is None, which will pull the minimum date from the data. May be either a string in 'YYYY/MM/DD' format, or a datetime object.

end_date (string, datetime): The end date used in the query, if desired to use in naming in the visualization output. The default is None, which will pull the maximum date from the data. May be either a string in 'YYYY/MM/DD' format, or a datetime object.

logo_path (path): The path to a desired logo image, with some validation by pubmed_tool.validators.path() with required suffix of '.png', '.jpeg', '.jpg', or '.gif'. The default is None, which omits an image.

primary_color (string): A string containing either a code-recognized color name or a hexadecimal color value for the color used for the trend line in the line plot, box of the boxplot, and distribution in the histogram. Default is 'blue'.

secondary_color (string): A string containing either a code-recognized color name or a hexadecimal color value for the color used for the mean constant line in the line plot, and outliers of the boxplot. Default is 'grey'.

accent_color (string): A string containing either a code-recognized color name or a hexadecimal color value for the color used for the mean 95% CI lines of the line plot. Default is 'grey'.

The pubmed_tool.full_visual() function is dependent on several subfunctions within the pubmed_tool.vis module, many of which are not as likely to be called outside of the main pubmed_tool.full_visual() function. However, these sub-functions may be useful

for debugging or otherwise modifying the functionality to serve an individual project's purpose.

### Other Helper Modules

There are two main helper modules: `pubmed_tool.logs` and `pubmed_tool.validators`:

`pubmed_tool.logs` assists in initialization and set-up of the logging functionality, for descriptive messages in functions. Modification of the functions within this file may be desired to tweak the functionality and output of the logger.

`pubmed_tool.validators` contains helper functions that assist in performance of validating several common inputs shared among the functions of the package. Modification or use of these functions may be desired as standards change or improve, or in the writing of custom functions utilizing the `pubmed_tool` package backbone.