Hack Nairaland

This project is meant to be an exercise in web scraping. Please use gently to avoid overloading the nairaland servers. I'd advise you work with it at night time, when there's less traffic. It also served as an exercise for me in the understanding of html page structure. It can for you too. I have documented some of the more common structures you'll find on the site.

Several demos of this project in action are provided in the accompanying `Hack Nairaland jupyter notebook`. You should start from there.

Functionalities offered by this project

1. Export all comments made by a user to an excel file (other formats can easily be added). You may select how many pages of comments you want.
2. Export all post titles from a section within a range that you specify to an excel file (other formats may be added).
3. Get all unique commenters on a post
4. Get all commenters on a post and their comment frequency
5. Save a post permanently by exporting it to word
6. Pick two users and see their discussion thread (to do)

How to work with this notebook

1. Download or clone the `bitbucket` repo (`git clone https://parousiaic@bitbucket.org/parousiaic/hack-nairaland.git`)
2. Open `cmd.exe` and `cd` (i.e. navigate) into the downloaded folder
3. Issue the command `pipenv install`. This step requires that you have `python` and `pipenv` installed in your system
4. After `pipenv` has finished recreating the project environment, issue the command, `pipenv shell` to activate it.

Creating the environment on your own

1. To complete this step you have to delete both `Pipfile` and `Pipfile.lock`
2. Download or clone the `bitbucket` repo (`git clone https://parousiaic@bitbucket.org/parousiaic/hack-nairaland.git`)
3. Open `cmd.exe` and `cd` (i.e. navigate) into the downloaded folder
4. Issue the command `pipenv install`. This step requires that you have `python` and `pipenv` installed in your system
5. Open `cmd.exe` and `cd` (i.e. navigate) into the folder
6. Issue commands `pipenv install`. Wait for the environment to be created.
7. When environment creation is done, activate it by issuing the command `pipenv shell`

8. After environment activation, issue commands `pipenv install jupyter`, `pipenv install ipykernel`

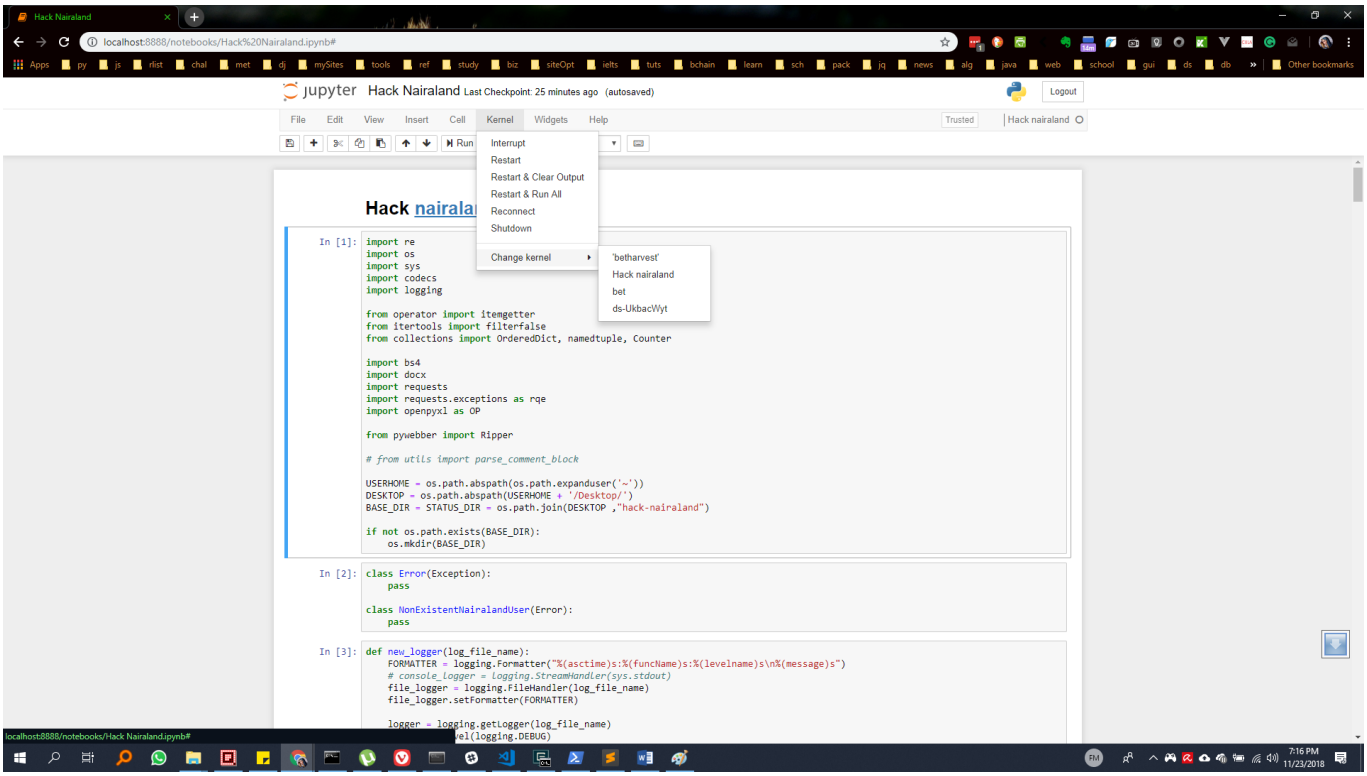
9. To create the custom `ipython` kernel, issue command

   ```
   python -m ipykernel install --user --name other-env --display-name "Hack nairaland"

   python -m ipykernel install --user --display-name "Hack nairaland or whatever name you like"
   ```

10. The `--user other-env` argument, value pair is optional. See here for explanation

11. Now when you reload your `ipython/jupyter` notebooks you should see the newly created kernel listed under kernels. You can see mine in the screenshot below.



Data Structures

Every function has its own data structure. They are documented here for easy understanding of the program logic.

`parse_comment_block`

Returns a namedtuple

```
namedtuple('ParsedComment', ['focus_user_ordered_dict', 'quotes_ordered_dict'])

# Internal structure
('focus_user_ordered_dict', OrderedDict([('moniker', 'comment')]))

('quotes_ordered_dict', OrderedDict([('moniker', 'comment')]))
```

PostCollector()

This class is used to scrap a nairaland post.

`PostCollector.scrap_comments_for_range_of_pages()`

Return type of `types.generator`. `Yields` `OrderedDict()` objects. Where each has structure shown below.

```
OrderedDict(['moniker', parse_comment_block function object])
```

User `comment header <tr>` structure - Post view

```
<tr>
    <td class="bold l pu">
        <a name="68136427"></a>
        <a name="msg68136427"></a>
        <a name="4539839.6"></a>
        <a href="/4539839/agu-aina-dropped-nigerias-final#68136427">Re:Topic.</a>
        by
        <a href="/moniker" class="user" title="Location: Detroit">moniker</a>
        (
        <span class="m or f">m or f</span>
        ):
        <span class="s">
            <b>00:00am</b>
        </span>
    </td>
</tr>
```

User `comment text <tr>` structure - Post view

```
<tr>
    <td id="pb68136427" class="l w pd">
        <div class="narrow">
            <blockquote>
                <a href="/post/68136377">
                    <b>moniker</b>
                </a>
                :
                <br>
                comment
            </blockquote>
            <br>
            <br>
            comment
        </div>
        <p class="s">
            <b id="lpt68136427">n Likes </b>
```

```
            <b id="shb68136427">n Share</b>
        </p>
    </td>
</tr>
```

Unpacking post

```
import textwrap

post = hack.PostCollector('https://www.nairaland.com/4862847/presidency-well-
teach-nursery-school')
for page in list(post.scrap_comments_for_range_of_pages(start=0, stop=2)):
    for moniker, parsed_comment in page.items():
        print(moniker)
        print(parsed_comment.focus_user_ordered_dict)
        for commenter, comment in parsed_comment.quotes_ordered_dict.items():
            print(textwrap.indent(commenter, "    "))
            print(textwrap.indent(comment, "    "))
        print("\n", "*"*100, "\n")
    print("+"*40, " new page ", "+"*40)
```

UserCommentHistory()

User `comment header <tr>` structure - Comment history view

```
<tr>
    <td class="bold l pu">
        <a name=""></a>
        <a name=""></a>
        <a name=""></a>
        <img src="">
        <a href="/section-url">Section name</a>
        /
        <a href="/0000/title">Re: Post title</a>
        by
        <a href="/username" class="user" title="Location:location">username</a>
        (
        <span class="m or f">m or f</span>
        ):
        <span class="s">
            <b>00:00pm</b>
            On
            <b>Oct 19</b>
            ,
            <b>2017</b>
        </span>
    </td>
</tr>
```

User comments text `<tr>` structure - Comment history view

This section has a few other elements if you're logged in

```
<tr>
    <td id="" class="l w pd">
        <div class="narrow">
            <blockquote> <!-- each blockquote represents a quoted comment -->
                <a href="/post/00000000">
                    <b>Moniker</b>
                </a>
                :
                <br> <!-- there'll be as many of these as the number of enter keys
a user presses-->
                Some comment
                <img src="" alt="represents an emoticon" border="0" class="faces">
            </blockquote>
            some comment
        </div>
        <p class="s">
            <b id="lpt61581477">n Likes </b>
            <b id="shb61581477">n Share</b>
        </p>
    </td>
</tr>
```

A supposed anomaly I found in the table

It contains nothing. Just a blank row.

```
<tr>
    <td class="l pu pd">
        <a name="68130046"></a>
    </td>
</tr>
```

Unpacking user comments history

```
import textwrap

for page in
list(UserCommentHistory("preccy69").scrap_comments_for_page_range(start=0,
stop=1)):
    for section, topic_plus_comment in page.items():
        print("\n\n", "*"*40, section, "*"*40)
```

```
        print(topic_plus_comment.topic.upper()) # for differentiation only

        parsed_comment = topic_plus_comment.parsed_comment # a namedtuple instance
        print(parsed_comment.focus_user_ordered_dict)

        quotes = parsed_comment.quotes_ordered_dict
        for username, comment in quotes.items():
            print(" "*8)
            print(textwrap.indent(username, "    "))
            print(textwrap.indent(comment, "    "))
        print("_"*100)
    print("\n\n")
```

TopicCollector()

Section topics `<tr>` structure

```
<tr>
    <td id="top2792995" class="w">
        <a name="2792995"></a>
        <img src="/icons/sticky.gif">
        <b>
            <a href="/2792995/nairaland-says-no-secessionists">Topic title</a>
        </b>
        <a href="topic-full-url/1">(1)</a>
        <a href="topic-full-url/2">(2)</a>
        <a href="topic-full-url/max-page">(max-page)</a>
        <img src="/icons/new.gif">
        <b>*</b>
        <br>
        <span class="s">
            by
            <b> <!--These <b> tags occur a maximum of 7 times-->
                <a href="/username">username</a>
            </b>
            .
            <b>number of comments</b>
            posts &amp;amp;
            <b>number of views</b>
            views.
            <b>00:00am</b>
            On
            <b>Jun 03</b> <!-- Visible for posts older than current date -->
            <b>2015</b> <!-- Visible for posts older than current year -->
            (
            <b>
                <a href="/user_moniker">user_moniker</a>
            </b>
            )
        </span>
```

```
    </td>
</tr>
```

Unpacking Topics

```
import textwrap

for page in
TopicCollector(section='politics').scrap_topics_for_range_of_pages(start=0,
stop=1):
    for topic in list(page):
        print(topic.poster)
        print(textwrap.indent(topic.title, "    "))
        print(textwrap.indent(topic.url, "    "))
        print(textwrap.indent(topic.comments, "    "), " comments")
        print(textwrap.indent(topic.views, "    "), " views")
        print(textwrap.indent(topic.last_commenter, "    "), " commented last")
        print(textwrap.indent(topic.other_meta), "    ")
        print()
```

To do

1. Show list of all sections
2. Fix scroll to top link