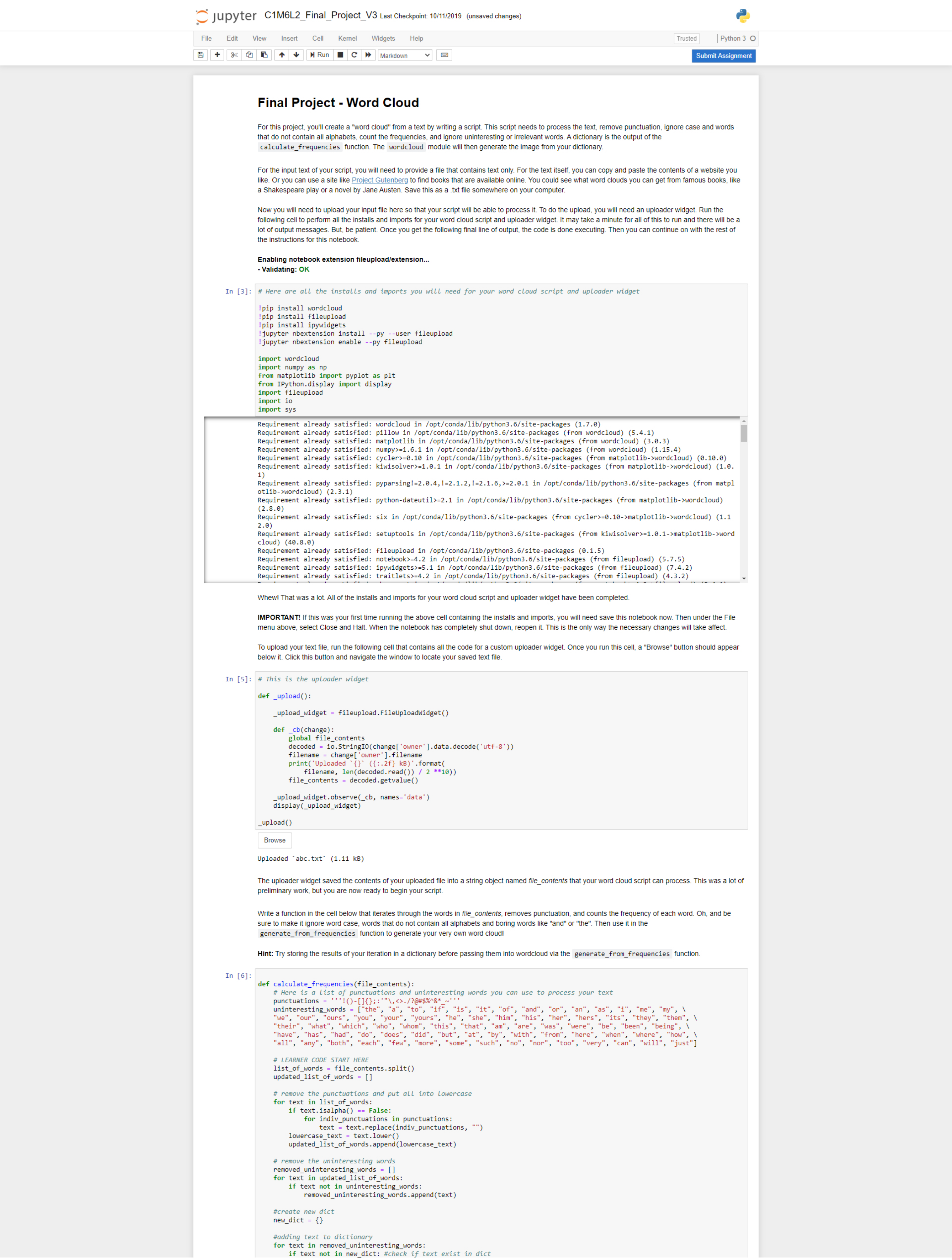


Final Project - Word Cloud

For this project, you'll create a "word cloud" from a text by writing a script. This script needs to process the text, remove punctuation, ignore case and words that do not contain all alphabets, count the frequencies, and ignore uninteresting or irrelevant words. A dictionary is the output of the `calculate_frequencies` function. The `wordcloud` module will then generate the image from your dictionary.

For the input text of your script, you will need to provide a file that contains text only. For the text itself, you can copy and paste the contents of a website you like. Or you can use a site like Project Gutenberg to find books that are available online. You could see what word clouds you can get from famous books, like a Shakespeare play or a novel by Jane Austen. Save this as a .txt file somewhere on your computer.

Now you will need to upload your input file here so that your script will be able to process it. To do the upload, you will need an uploader widget. Run the following cell to perform all the installs and imports for your word cloud script and uploader widget. It may take a minute for all of this to run and there will be a lot of output messages. But, be patient. Once you get the following final line of output, the code is done executing. Then you can continue on with the rest of the instructions for this notebook.

Enabling notebook extension fileupload/extension...
- Validating: OK

In [3]:
```
# Here are all the installs and imports you will need for your word cloud script and uploader widget

!pip install wordcloud
!pip install fileupload
!pip install ipywidgets
!jupyter nbextension install --py --user fileupload
!jupyter nbextension enable --py fileupload

import wordcloud
import numpy as np
from matplotlib import pyplot as plt
from IPython.display import display
import fileupload
import io
import sys
```

```
Requirement already satisfied: wordcloud in /opt/conda/lib/python3.6/site-packages (1.7.0)
Requirement already satisfied: pillow in /opt/conda/lib/python3.6/site-packages (from wordcloud) (5.4.1)
Requirement already satisfied: matplotlib in /opt/conda/lib/python3.6/site-packages (from wordcloud) (3.0.3)
Requirement already satisfied: numpy>=1.6.1 in /opt/conda/lib/python3.6/site-packages (from wordcloud) (1.15.4)
Requirement already satisfied: cycler>=0.10 in /opt/conda/lib/python3.6/site-packages (from matplotlib->wordcloud) (0.10.0)
Requirement already satisfied: kiwisolver>=1.0.1 in /opt/conda/lib/python3.6/site-packages (from matplotlib->wordcloud) (1.0.
1)
Requirement already satisfied: pyparsing!=2.0.4,!=2.1.2,!=2.1.6,>=2.0.1 in /opt/conda/lib/python3.6/site-packages (from matpl
otlib->wordcloud) (2.3.1)
Requirement already satisfied: python-dateutil>=2.1 in /opt/conda/lib/python3.6/site-packages (from matplotlib->wordcloud)
(2.8.0)
Requirement already satisfied: six in /opt/conda/lib/python3.6/site-packages (from cycler>=0.10->matplotlib->wordcloud) (1.1
2.0)
Requirement already satisfied: setuptools in /opt/conda/lib/python3.6/site-packages (from kiwisolver>=1.0.1->matplotlib->word
cloud) (40.8.0)
Requirement already satisfied: fileupload in /opt/conda/lib/python3.6/site-packages (0.1.5)
Requirement already satisfied: notebook>=4.2 in /opt/conda/lib/python3.6/site-packages (from fileupload) (5.7.5)
Requirement already satisfied: ipywidgets>=5.1 in /opt/conda/lib/python3.6/site-packages (from fileupload) (7.4.2)
Requirement already satisfied: traitlets>=4.2 in /opt/conda/lib/python3.6/site-packages (from fileupload) (4.3.2)
```

Whew! That was a lot. All of the installs and imports for your word cloud script and uploader widget have been completed.

IMPORTANT! If this was your first time running the above cell containing the installs and imports, you will need save this notebook now. Then under the File menu above, select Close and Halt. When the notebook has completely shut down, reopen it. This is the only way the necessary changes will take affect.

To upload your text file, run the following cell that contains all the code for a custom uploader widget. Once you run this cell, a "Browse" button should appear below it. Click this button and navigate the window to locate your saved text file.

In [5]:
```
# This is the uploader widget

def _upload():

    _upload_widget = fileupload.FileUploadWidget()

    def _cb(change):
        global file_contents
        decoded = io.StringIO(change['owner'].data.decode('utf-8'))
        filename = change['owner'].filename
        print('Uploaded `{}` ({:.2f} kB)'.format(
            filename, len(decoded.read()) / 2 **10))
        file_contents = decoded.getvalue()

    _upload_widget.observe(_cb, names='data')
    display(_upload_widget)

_upload()
```

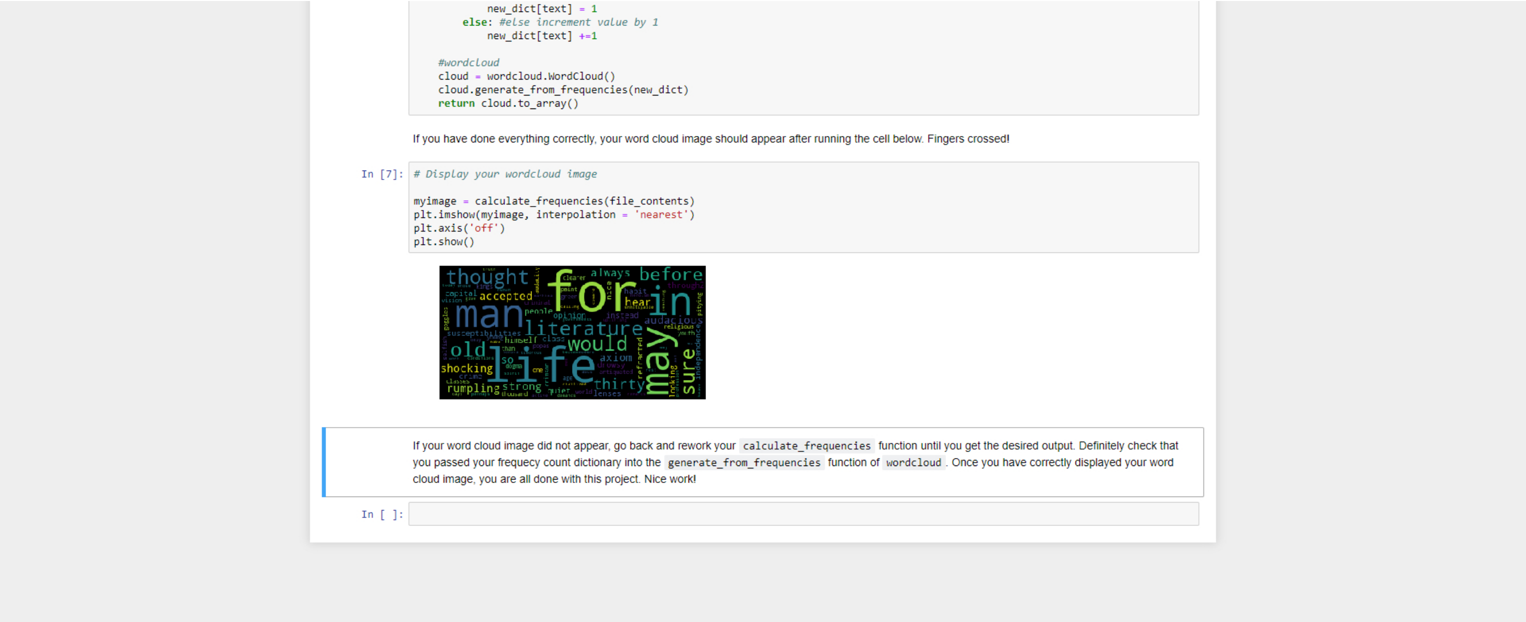
```
Uploaded `abc.txt` (1.11 kB)
```

The uploader widget saved the contents of your uploaded file into a string object named *file_contents* that your word cloud script can process. This was a lot of preliminary work, but you are now ready to begin your script.

Write a function in the cell below that iterates through the words in *file_contents*, removes punctuation, and counts the frequency of each word. Oh, and be sure to make it ignore word case, words that do not contain all alphabets and boring words like "and" or "the". Then use it in the `generate_from_frequencies` function to generate your very own word cloud!

Hint: Try storing the results of your iteration in a dictionary before passing them into wordcloud via the `generate_from_frequencies` function.

In [6]:
```
def calculate_frequencies(file_contents):
    # Here is a list of punctuations and uninteresting words you can use to process your text
    punctuations = '''!()-[]{};:'"\,<>./?@#$%^&*_~'''
    uninteresting_words = ["the", "a", "to", "if", "is", "it", "of", "and", "or", "an", "as", "i", "me", "my", \
    "we", "our", "ours", "you", "your", "yours", "he", "she", "him", "his", "her", "hers", "its", "they", "them", \
    "their", "what", "which", "who", "whom", "this", "that", "am", "are", "was", "were", "be", "been", "being", \
    "have", "has", "had", "do", "does", "did", "but", "at", "by", "with", "from", "here", "when", "where", "how", \
    "all", "any", "both", "each", "few", "more", "some", "such", "no", "nor", "too", "very", "can", "will", "just"]

    # LEARNER CODE START HERE
    list_of_words = file_contents.split()
    updated_list_of_words = []

    # remove the punctuations and put all into lowercase
    for text in list_of_words:
        if text.isalpha() == False:
            for indiv_punctuations in punctuations:
                text = text.replace(indiv_punctuations, "")
        lowercase_text = text.lower()
        updated_list_of_words.append(lowercase_text)

    # remove the uninteresting words
    removed_uninteresting_words = []
    for text in updated_list_of_words:
        if text not in uninteresting_words:
            removed_uninteresting_words.append(text)

    #create new dict
    new_dict = {}

    #adding text to dictionary
    for text in removed_uninteresting_words:
        if text not in new_dict: #check if text exist in dict
```



```
            new_dict[text] = 1
        else: #else increment value by 1
            new_dict[text] +=1

    #wordcloud
    cloud = wordcloud.WordCloud()
    cloud.generate_from_frequencies(new_dict)
    return cloud.to_array()
```

If you have done everything correctly, your word cloud image should appear after running the cell below. Fingers crossed!

In [7]:
```
# Display your wordcloud image

myimage = calculate_frequencies(file_contents)
plt.imshow(myimage, interpolation = 'nearest')
plt.axis('off')
plt.show()
```

If your word cloud image did not appear, go back and rework your `calculate_frequencies` function until you get the desired output. Definitely check that you passed your frequecy count dictionary into the `generate_from_frequencies` function of `wordcloud`. Once you have correctly displayed your word cloud image, you are all done with this project. Nice work!

In [ ]: