Is it Essential? v0.2

May 5, 2020

[ ]:
```
##goalsV0.2:output the hours, whether the business is open, whether it is␣
↪essential in a print statement.
#-getkeyword funcitonality working as well

#if we have the time, v0.3 is going to include some google maps functionality,␣
↪maybe predictive text, and error handling
##dump results into data file
```

[19]:
```
!pip install requests
import json
import requests
import pandas as pd
import warnings
warnings.filterwarnings('ignore')
```

```
Requirement already satisfied: requests in /opt/conda/lib/python3.7/site-
packages (2.12.4)
```

[67]:
```
def naic_lookup(code):
    yr = 2012
    url = f"http://naics.codeforamerica.org/v0/q?year={yr}&code={code}"
    response = requests.get(url)
    data = response.json()
    return data
def geocoding():
    key = "AIzaSyCIZ5647df1-qq4AVq5Jmt9uhLJWSLkmeo"
    address = input("Type in your address. ")
    params = { "key" : f"{key}", "address" : f"{address}"}
    url = "https://maps.googleapis.com/maps/api/geocode/json?"
    response = requests.get(url, params = params)
    one = response.json()
    latlng = (one['results'][0]['geometry']['location'])
    return latlng
def places_lookup(place):
    key = "AIzaSyCIZ5647df1-qq4AVq5Jmt9uhLJWSLkmeo"
    params = { "key" : f"{key}", "input" : f"{place}", "inputtype" :␣
↪"textquery", "language" : "english", "locationbias" : "ipbias"}
```

```
    url = f"https://maps.googleapis.com/maps/api/place/findplacefromtext/json?"
    response = requests.get(url, params = params)
    candidates = response.json()
    values = (candidates['candidates'])
    IDlist = []
    for placeid in values:
        IDlist.append(placeid['place_id'])
    return IDlist
def places_details(place_id):
    key = "AIzaSyCIZ5647df1-qq4AVq5Jmt9uhLJWSLkmeo"
    url = f"https://maps.googleapis.com/maps/api/place/details/json?
 ↪key={key}&place_id={place_id}&fields=name,business_status,formatted_address,opening_hours,n
    response = requests.get(url)
    results = response.json()
```

[65]:
```
address = input("Type in your address.")
geocoding(address)
```

```
      ␣
↪---------------------------------------------------------------------------

       KeyboardInterrupt                         Traceback (most recent call␣
↪last)

       /opt/conda/lib/python3.7/site-packages/ipykernel/kernelbase.py in␣
↪_input_request(self, prompt, ident, parent, password)
       883             try:
   --> 884                 ident, reply = self.session.recv(self.stdin_socket,␣
↪0)
       885             except Exception:


       /opt/conda/lib/python3.7/site-packages/jupyter_client/session.py in␣
↪recv(self, socket, mode, content, copy)
       802         try:
   --> 803             msg_list = socket.recv_multipart(mode, copy=copy)
       804         except zmq.ZMQError as e:


       /opt/conda/lib/python3.7/site-packages/zmq/sugar/socket.py in␣
↪recv_multipart(self, flags, copy, track)
       474         """
   --> 475         parts = [self.recv(flags, copy=copy, track=track)]
       476         # have first part already, only loop while more to receive
```

```
       zmq/backend/cython/socket.pyx in zmq.backend.cython.socket.Socket.recv()

       zmq/backend/cython/socket.pyx in zmq.backend.cython.socket.Socket.recv()

       zmq/backend/cython/socket.pyx in zmq.backend.cython.socket._recv_copy()

       /opt/conda/lib/python3.7/site-packages/zmq/backend/cython/checkrc.pxd in␣
↪zmq.backend.cython.checkrc._check_rc()

       KeyboardInterrupt:


   During handling of the above exception, another exception occurred:


       KeyboardInterrupt                         Traceback (most recent call␣
↪last)

       <ipython-input-65-fbea855b6420> in <module>
   ----> 1 address = input("Type in your address.")
         2 geocoding(address)


       /opt/conda/lib/python3.7/site-packages/ipykernel/kernelbase.py in␣
↪raw_input(self, prompt)
       857             self._parent_ident,
       858             self._parent_header,
   --> 859             password=False,
       860         )
       861


       /opt/conda/lib/python3.7/site-packages/ipykernel/kernelbase.py in␣
↪_input_request(self, prompt, ident, parent, password)
       887             except KeyboardInterrupt:
       888                 # re-raise KeyboardInterrupt, to truncate traceback
   --> 889                 raise KeyboardInterrupt
       890             else:
       891                 break


       KeyboardInterrupt:
```

[66]:
```
geocoding()
place_id = places_lookup()[0]
places_details(place_id)
```

```
Type in your address. 852 Sumner Avenue, Syracuse NY


      ␣
↪---------------------------------------------------------------------------

       TypeError                                 Traceback (most recent call␣
↪last)

       <ipython-input-66-c7af7b083672> in <module>
         1 geocoding()
   ----> 2 place_id = places_lookup()[0]
         3 places_details(place_id)


       TypeError: places_lookup() missing 1 required positional argument:␣
↪'place'
```

[18]:
```
code = input("Enter a NAICS code, up to 6 digits. ")
data = naic_lookup(code)
print(data['title'])
```

```
Enter a NAICS code, up to 6 digits. 8111


      ␣
↪---------------------------------------------------------------------------

       NameError                                 Traceback (most recent call␣
↪last)

       <ipython-input-18-4c8d9313ac57> in <module>
         1 code = input("Enter a NAICS code, up to 6 digits. ")
   ----> 2 data = naic_lookup(code)
         3 print(data['title'])


       NameError: name 'naic_lookup' is not defined
```

[72]:
```
search_select = input("Please type in what you will be searching by: 'keyword',␣
↪'NAIC code', or 'store name'. Type 'dict' to search essential businesses by␣
↪sector. Type 'exit' to quit. ")
```

```
while search_select != 'exit':
    if search_select == "keyword":
        with open ("EssentialBusinesses.txt", 'r') as f:
            search1 = input("Please input a keyword to search our list of␣
↪essential businesses. Type 'quit' to quit. ")
            search = search1.capitalize()
            for line in f:
                if (f"{search}") in line and ("SIC") in line:
                    print("")
                    print(line)
                    keepgoing = input("Is this the industry you were looking␣
↪for? y/n ")
                    if keepgoing == "y":
                        print(f"This industry is considered essential. A *␣
↪denotes that there may be some exceptions.")
                        break
                    else:
                        continue
            break
    elif search_select == "NAIC code":
        code = input("Enter a NAICS code, up to 4 digits. ")
        data = naic_lookup(code)
        industry = (data['title'])
        with open ('EssentialBusinesses.txt', 'r') as e:
            if (f"{code}") in e.read():
                essential = 'yes'
            else:
                essential = 'no'
        print(f"This NAIC code refers to: {industry}." )
        print(f"Essential: {essential}")
        break
    elif search_select == "store name":
        place = input("Please input the name of a store and the nearest one to␣
↪your address  as determined by IP bias will be returned.")
        place_id = places_lookup(place)[0]
        places_details(place_id)
        break
    elif search_select == 'dict':
        with open ('sectors.txt', 'r') as s:
            for line in s.readlines():
                print(line)
        sector = input("Type in one of the above sectors to bring up a list of␣
↪essential businesses in those sectors. ")
        with open ("EssentialBusinesses.txt", 'r') as d:
            for line in d:
                if line.startswith(sector):
                    print("")
```

```
                    print(line)
                    for line in d:
                        print (line)
                        if line.startswith('>'):
                            break

    else:
        print("That input wasn't recognized or the program ran into an␣
 ↪unspecified error. Please try again.")
        break
##could use a try function to find and account for errors, such as any key␣
 ↪errors from the google apis, but we're focused on
#getting this running first and foremost and then expanding functionality
```

```
Please type in what you will be searching by: 'keyword', 'NAIC code', or 'store
name'. Type 'dict' to search essential businesses by sector. Type 'exit' to
quit. store name
Please input the name of a store and the nearest one to your address  as
determined by IP bias will be returned.Target
{'html_attributions': [], 'result': {'business_status': 'OPERATIONAL',
'formatted_address': '1701 S Yale Ave, Tulsa, OK 74112, USA', 'name': 'Target',
'opening_hours': {'open_now': True, 'periods': [{'close': {'day': 0, 'time':
'2100'}, 'open': {'day': 0, 'time': '0800'}}, {'close': {'day': 1, 'time':
'2100'}, 'open': {'day': 1, 'time': '0800'}}, {'close': {'day': 2, 'time':
'2100'}, 'open': {'day': 2, 'time': '0800'}}, {'close': {'day': 3, 'time':
'2100'}, 'open': {'day': 3, 'time': '0800'}}, {'close': {'day': 4, 'time':
'2100'}, 'open': {'day': 4, 'time': '0800'}}, {'close': {'day': 5, 'time':
'2100'}, 'open': {'day': 5, 'time': '0800'}}, {'close': {'day': 6, 'time':
'2100'}, 'open': {'day': 6, 'time': '0800'}}], 'weekday_text': ['Monday: 8:00 AM
- 9:00 PM', 'Tuesday: 8:00 AM - 9:00 PM', 'Wednesday: 8:00 AM - 9:00 PM',
'Thursday: 8:00 AM - 9:00 PM', 'Friday: 8:00 AM - 9:00 PM', 'Saturday: 8:00 AM -
9:00 PM', 'Sunday: 8:00 AM - 9:00 PM']}}, 'status': 'OK'}


      ␣
 ↪---------------------------------------------------------------------------

       TypeError                                 Traceback (most recent call␣
 ↪last)

       <ipython-input-72-6c5401a8d6ca> in <module>
        32         place_id = places_lookup(place)[0]
        33         one = places_details(place_id)
   ---> 34         print(one['result']['0'])
        35         break
        36     elif search_select == 'dict':
```

```
TypeError: 'NoneType' object is not subscriptable
```

[76]:
```
place = input("Please input the name of a store and the nearest one to your␣
 ↪address  as determined by IP bias will be returned.")
place_id = places_lookup(place)[0]
one = places_details(place_id)
print(one.type())
```

```
Please input the name of a store and the nearest one to your address  as
determined by IP bias will be returned.Target
{'html_attributions': [], 'result': {'business_status': 'OPERATIONAL',
'formatted_address': '8061 Brewerton Rd, Cicero, NY 13039, USA', 'name':
'Target', 'opening_hours': {'open_now': True, 'periods': [{'close': {'day': 0,
'time': '2100'}, 'open': {'day': 0, 'time': '0700'}}, {'close': {'day': 1,
'time': '2100'}, 'open': {'day': 1, 'time': '0700'}}, {'close': {'day': 2,
'time': '2100'}, 'open': {'day': 2, 'time': '0700'}}, {'close': {'day': 3,
'time': '2100'}, 'open': {'day': 3, 'time': '0700'}}, {'close': {'day': 4,
'time': '2100'}, 'open': {'day': 4, 'time': '0700'}}, {'close': {'day': 5,
'time': '2100'}, 'open': {'day': 5, 'time': '0700'}}, {'close': {'day': 6,
'time': '2100'}, 'open': {'day': 6, 'time': '0700'}}], 'weekday_text': ['Monday:
7:00 AM - 9:00 PM', 'Tuesday: 7:00 AM - 9:00 PM', 'Wednesday: 7:00 AM - 9:00
PM', 'Thursday: 7:00 AM - 9:00 PM', 'Friday: 7:00 AM - 9:00 PM', 'Saturday: 7:00
AM - 9:00 PM', 'Sunday: 7:00 AM - 9:00 PM']}}, 'status': 'OK'}
```

```
      ␣
 ↪---------------------------------------------------------------------------

       AttributeError                            Traceback (most recent call␣
 ↪last)

       <ipython-input-76-ee90dd8352d9> in <module>
         2 place_id = places_lookup(place)[0]
         3 one = places_details(place_id)
    ----> 4 print(one.type())


       AttributeError: 'NoneType' object has no attribute 'type'
```

[36]:
```
search = input("Please input a keyword to search our list of essential␣
 ↪businesses. Type 'quit' to quit. ")
search1 = search.capitalize()
print(search1)
```

```
Please input a keyword to search our list of essential businesses. Type 'quit'
to quit. auto
```

Auto

[35]:
```
##note: keyword text is not predictive. maybe something to change in another␣
↪version
```

[42]:
```
with open ('sectors.txt', 'r') as s:
    for line in s.readlines():
        print(line)
sector = input("Type in one of the above sectors to bring up a list of␣
↪essential services in those sectors. Type anything else to exit. ")
with open ("EssentialBusinesses.txt", 'r') as d:
    for line in d:
        if line.startswith(sector):
            print("")
            print(line)
            for line in d:
                print (line)
                if line.startswith('>'):
                    break
```

```
Health Care

Infrastructure

Manufacturing

Retail

Services

Media

Financial Institutions

Social Services

Construction

Defense

Sanitation and Safety

Vendors

Recreation

Professional Services
```

```
Type in one of the above sectors to bring up a list of essential services in
those sectors. Type anything else to exit. Professional Services

Professional Services:



        SIC Code 8111 - Legal Services -     Lawyers may continue to perform all
work necessary for any service so long as it is performed remotely. Any in-
person work presence shall be limited to work only in support of essential
businesses or services; however, even work in support of an essential business
or service should be conducted as remotely as possible.

        SIC Code 65 - Real Estate - Real estate services shall be conducted
remotely for all transactions, including but not limited to title searches,
appraisals, permitting, inspections, and the recordation, legal, financial and
other services necessary to complete a transfer of real property; provided,
however, that any services and parts therein may be conducted in-person only to
the extent legally necessary and in accordance with appropriate social
distancing and cleaning/disinfecting protocols; and nothing within this
provision should be construed to allow brokerage and branch offices to remain
open to the general public (i.e. not clients).

>
```

[8]:
```
with open('EssentialBusinesses.txt', 'r') as f:
    for line in f.readlines():
        print(line)
```

```
hi
```

[ ]:
```
##writing naic codes to file
```

[ ]:
```

```