I developed this program using a development environment that consisted of the following components.

1. MacBook Pro running MacOS Mojave 10.14.6
2. iMac running MacOS Mojave 10.14.6
3. Python 3.8.2
4. IDLE 3.8.2
5. Shared drive between the MacBook and iMac

The process for to do this consists of a few steps:

1. Search the real estate listings for sale in Harding township (Zillow does this all the time.. right?)
2. Compare this list of real estate for sale to the list of Historic properties.
3. Create the list of historic properties and post it to my website.
4. Send an email to the members of the Harding Historical Preservation Committee.

Sounds easy ... right? Well, let's go through it!

1. Search the real estate listings for sale in Harding township.

Finding a reliable and inexpensive source for the properties for sale in Harding sounds somewhat easy but it turns out it's not. There are many sources of free information about a specific property for SOLD properties but only multiple listing service (MLS) data has information about properties for sale. I found out that there is not one but many MLS sources that are owned by many different organizations. After much searching, I found https://rapidapi.com/apidojo/api/realtor. It's well supported and the price is free for a low volume of API calls. You need to sign up here https://rapidapi.com. I picked the free option (<= 500 API calls/month) since I'll be making one (1) API call/week in production. Here's the code fragment for retrieving those records and writing the JSON data to a data file.

```
#
# Make API calls to RapidAPI to find properties for sale in Harding.
#
#url = "https://realtor.p.rapidapi.com/locations/auto-complete" # used to search for specific properties
url = "https://realtor.p.rapidapi.com/properties/list-for-sale"
headers = {
    'x-rapidapi-host': "realtor.p.rapidapi.com",
    'x-rapidapi-key': "229e80e323msh0697f51cb0c7a00p1d0eacjsn18e02be76e4a"
    }

print( "** Starting ...'" )
```

```
fo = open( '/Users/Clay/Documents/Python/HistoricProperties/HardingListings.json', 'w+' )


print( "** Retrieving remote Real Estate data  ...'" )
# retrieve 1 chunk of records.
querystring =
{"sort":"relevance","radius":"1","city":"Harding","offset":"0","limit":"200","state_code":"NJ"}
response = requests.request("GET", url, headers=headers, params=querystring)
if response.status_code  != 200:
    print( "Abnormal Status Code from requests.request(GET): ", response.status_code )
    fo.close()
    sys.exit( "Processing Terminated" )

hdata = json.dumps( response.text )

print( "** Writing MLS data to file [HardingListings.json] ...'" )
json.dump( response.text, fo )

#debugging stuff
#pretty_json = json.loads( response.text )
#print (json.dumps(pretty_json, indent=2))

fo.close()
```

For the sake of efficiency, I combined steps 2 & 3 above. After I opened the excel file with the list of Historic properties, I compared the addresses to the address in the JSON data. Once a had a match, I constructed strings that would be send to my WordPress site using post requests.
Here's the code for performing all of that. Please note that I attempted to use several different substring methods available in Python (e.g. "in", "startswith()", others) instead of the "find()" method. The find() method was the most reliable method to use although other methods seemed to work in small test programs and in the Python shell. I suspect that there are some memory leaks in these methods.

```
for i in range(sheet.nrows):
    house_addr = sheet.cell_value(i, 2)
    # print( house_addr )
    for property in json_string[ 'listings' ]:
        # print( house_addr, property[ 'address' ] + " URL-> " +  property[ 'rdc_web_url' ] )
        s = property[ 'address' ]
        if ( s.find( house_addr ) != -1 ):
            print( "Found: " + house_addr + " URL: " + property[ 'rdc_web_url' ] )
            wp_tmp_content = wp_part1 + wp_part2 + '"' + property[ 'rdc_web_url' ] + '">' +
house_addr + wp_part3
            wp_content = wp_content + wp_tmp_content
            num_houses += 1

print ("" )
```

```
print ( "*** Total number of historic properties for # Give the location of the XL file with
Harding Historical Houses
print( "** Opening Excel file with Historic properties ..." )
loc = ("/Users/Clay/Documents/Python/HistoricProperties/HistoricHardingHouses.xlsx")

# Open Historic Workbook
wb = xlrd.open_workbook(loc)
sheet = wb.sheet_by_index(0)
sheet.cell_value(0, 0)

print ("" )
print ("====== Summary of Historic Property for Sale in Harding Township ======" )
print( "                    ", time.strftime("%c") )
print ("" )
# Extracting number of rows in Spreadsheet
# print(sheet.nrows)

# Mostly Artifacts now..
# Example HTML so that a link can be sent about the property to the Post
# wp_content = '</span><strong><span style="text-decoration: underline;"> ' \
# '<a
href="https://www.realtor.com/realestateandhomes-detail/607-Spring-Valley-Rd_Morristow
n_NJ_07960_M63809-33099">' \
# '607 Spring Valley Rd</a></span></strong></span></em></p>'

wp_part1 = '</span><strong><span style="text-decoration: underline;"> '
wp_part2 = '<a href='
wp_part3 = '</a></span></strong></span></em></p>'

wp_content = "<h4><strong>Please click on the address below for more
information.</strong></h4>"
num_houses = 0

print( "** Search for historic properties in MLS listing data ..." )
# Search for historic house in listings data
sale =", num_houses, "."  )

if ( num_houses == 0 ): # nothing more to do!!
    print( "No more processing required..." )
    exit()

print( "** Sending Data to WordPress POST ..." )
# deal with Current date and time using isoformat:" )
now = _datetime.datetime.now()
wp_date = now.isoformat()
wp_title = 'Historic Property For Sale in Harding on ' + time.strftime("%c")
wp_post = {
    'date':    wp_date,                    # '2020-04-07T07:00:00', YYYY-MM-DD
```

```
    'title':   wp_title,
    'content': wp_content,
    'status':  'publish',
    'categories' : [19], # Historic Category
    'slug' :   'historic'
}

wp_status = requests.post( wp_url + '/posts', headers=wp_header, json=wp_post )
print( "** WordPress POST returned status: ", wp_status )

# Readable display format... if needed
#print( json.dumps( json_object, indent=2) )
```

Final step 3. Sending the email to the Historic committee using my Hotmail/outlook account. There are several methods to send email from Python: Hotmail, gmail, different SMTP servers and ports, I chose this account because I was not concerned about security. If you are using your gmail account to send an email to a distribution list, then you will need to configure your gmail account to authorize communications from 3rd Party email addresses. I didn't want this because I am trying to keep my personal gmail account free of undesirable messages.

```
# Now send email to the members of the HPC using my hotmail address ...

print( "** Sending email to HPC ... " )

hotmail_user = 'YOURNAME@hotmail.com'
hotmail_password = 'YOUR_PASSWORD'
# Email addresses
to = ['EXAMPLE_TO@gmail.com', 'EXAMPLE2_TO@hotmail.com']

sent_from = hotmail_user
subject = 'List of Historic Properties for sale in Harding ' + time.strftime("%c")
body = "Hello Everyone, \n\nThis is the weekly list of Historic Properties for sale in
Harding.\n " \
"Please see this page for more information https://clayurl.com/category/historic/" \
"\n\nThanks! \nClay"

email_text = """\
From: %s
To: %s
Subject: %s

%s
```

```
""" % (sent_from, ", ".join(to), subject, body)

try:
# This MS SMTP server & port work well 19 April 2020
    server = smtplib.SMTP( 'smtp-mail.outlook.com', 587 )
    server.ehlo()
    server.starttls()
    server.ehlo()
    server.login(hotmail_user, hotmail_password )
    server.sendmail(sent_from, to, email_text)
    server.close()

    print ( '*** Email sent successfully ...' )
except:
    print ( 'Error sending Email to HPC ...something went wrong...' )

print( "** Processing finished Normally ..." )
```

Finally, I used cron on my iMac to setup a “cron job” to automatically execute this program once a week to refresh the list of historic properties.
Here’s the cron table.

```
Clays-iMac:~ root# crontab -l
# m h dom mon dow command
#hourly statistics calculation
0 2 * * 7  /Library/Frameworks/Python.framework/Versions/3.8/bin/python3 /Users/Clay/Documents/Python/HistoricProperties/HPC.py
>>/Users/Clay/Documents/Python/cron.txt 2>1
Clays-iMac:~ root#
```

Summary

This was a wonderful experience and I learned Python on a basic level. Since I spent many years as a C and C++ programmer, much of the Python language is very similar to C and C++. When I needed a function or method, there were similar constructs in Python. However, I do appreciate the strength of Python especially in terms of string functions, regular expressions, pattern matching, and documentation.