Assignment2

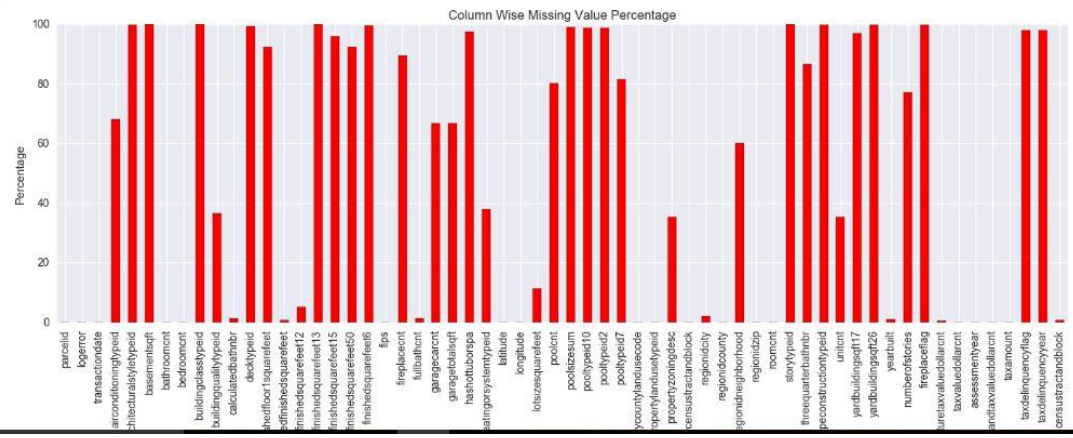
Team7: Snigdha Joshi & Vipra Shah

Assignment2:Team7

Part1: Perform EDA on Zillow Dataset:

Find missing values of merged csv from properties and train csv files.

```
In [193]: %matplotlib inline
          missing_values_df = merged_df.isnull().sum(axis=0)/len(merged_df.index) * 100
          my_plot = missing_values_df.plot(kind='bar',title='Column Wise Missing Value Percentage',figsize=(17, 5), color='red')
          my_plot.set_xlabel('Column')
          my_plot.set_ylabel('Percentage')

Out[193]: <matplotlib.text.Text at 0x2008765a198>
```



Remove columns having 80% blank data

Remove missing value columns

```
In [194]: not_needed = missing_values_df.reset_index()
          not_needed.columns = ['Column', 'Percentange']
          not_needed.ix[not_needed['Percentange']> 80.00]

C:\Users\smits_000\Anaconda3\lib\site-packages\ipykernel\__main__.py:3: DeprecationWarning:
.ix is deprecated. Please use
.loc for label based indexing or
.iloc for positional indexing

See the documentation here:
http://pandas.pydata.org/pandas-docs/stable/indexing.html#deprecate_ix
  app.launch_new_instance()
```

Out[194]:

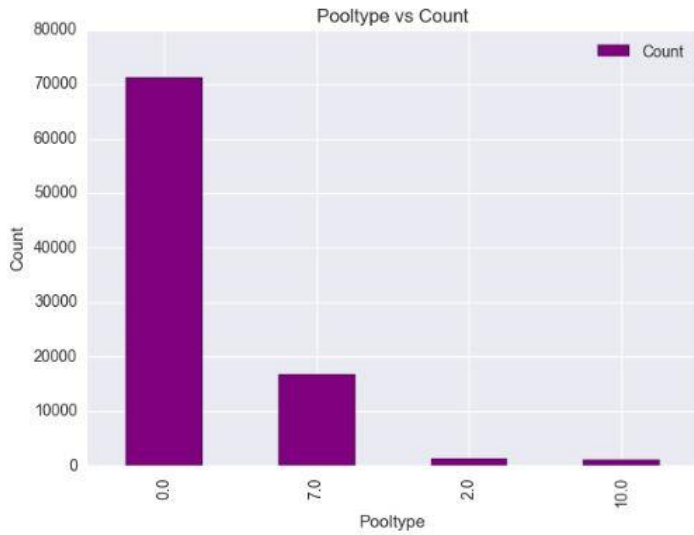
	Column	Percentange
4	architecturalstyletypeid	99.710883
5	basementsqft	99.952368
8	buildingclasstypeid	99.982276
11	decktypeid	99.271116
12	finishedfloor1squarefeet	92.405428

Merge pooltypeid10, pooltypeid2 and pooltypeid7 columns as one column pooltypeid and store respective pooltype for each row

```
In [198]: merged_df['pooltypeid'] = merged_df[['pooltypeid10','pooltypeid2','pooltypeid7']].sum(axis=1)
          merged_df['pooltypeid'].unique()

Out[198]: array([  0.,   7.,  10.,   2.])
```

Plot pooltype and its count

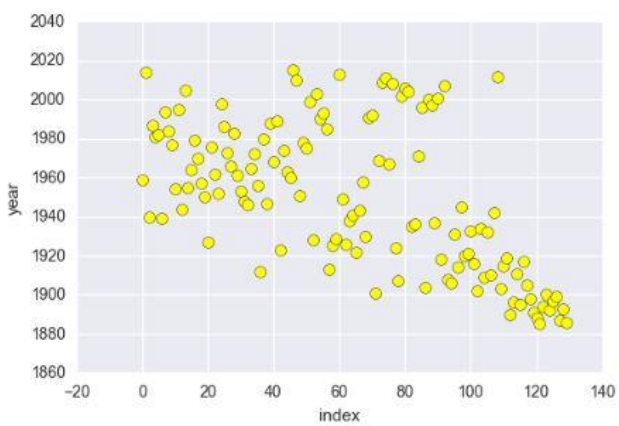
```
In [203]: plot = pool_df.plot(kind='bar',title='Pooltype vs Count',figsize=(7, 5), color='purple')
          plot.set_xlabel('Pooltype')
          plot.set_ylabel('Count')

Out[203]: <matplotlib.text.Text at 0x2008765aef0>
```



Analysis year built distribution

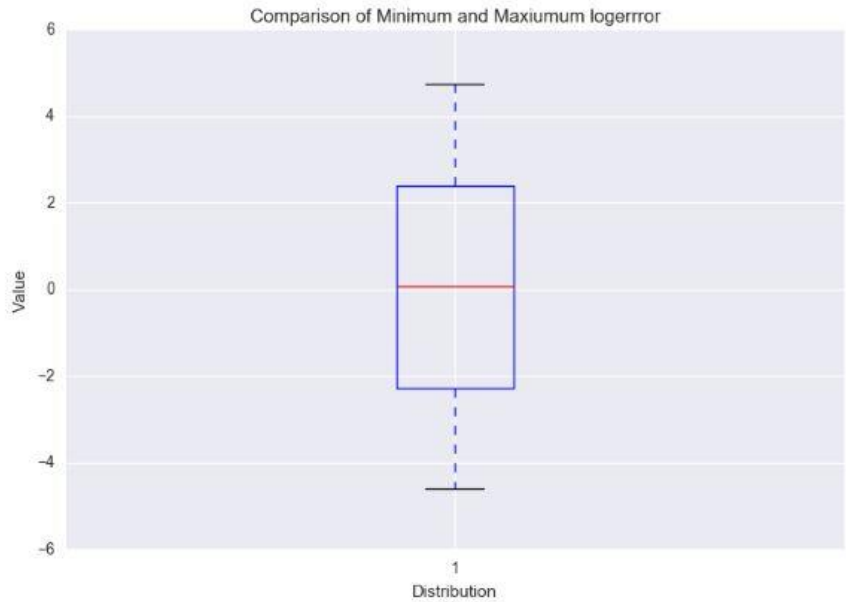
plot yearbuilt distribution as a scatter plot

```
In [209]: yeardf = data_to_csv['yearbuilt'].unique()
          yeardf = pd.DataFrame(yeardf)
          yeardf = yeardf.reset_index()
          yeardf.columns = ['index', 'year']
          yeardf.plot(kind='scatter', x='index', y='year', s=50, color='yellow')

Out[209]: <matplotlib.axes._subplots.AxesSubplot at 0x200916accf8>
```



Comparison of minimum and maximum logerror distibution

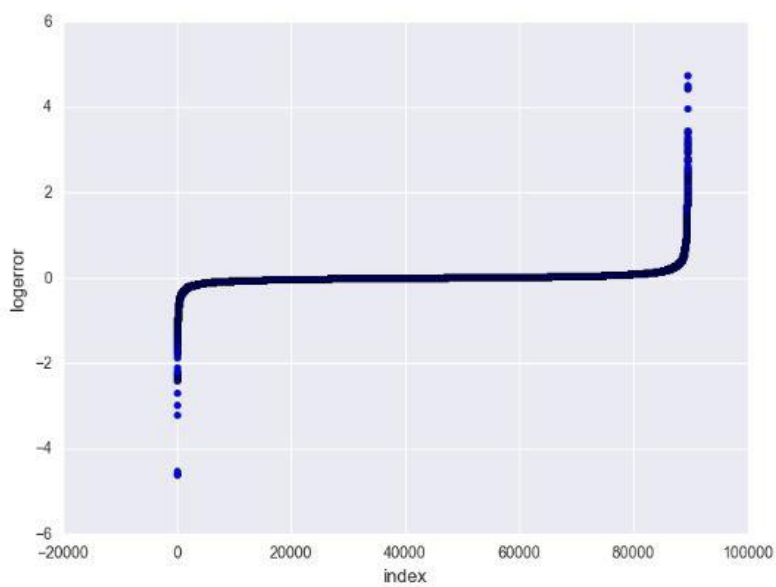
```
In [210]: min_logerror = data_to_csv['logerror'].min()
          max_logerror = data_to_csv['logerror'].max()
          box_plot = [min_logerror, max_logerror]
          fig = plt.figure(1, figsize=(9, 6))
          ax = fig.add_subplot(111)
          bp = ax.boxplot(box_plot)
          ax.set_title('Comparison of Minimum and Maxiumum logerrror')
          ax.set_xlabel('Distribution')
          ax.set_ylabel('Value')
          plt.show()
```



Find outliers for logerror data

Plot logerror distribution, remove outliers and plot its distribution

```
In [211]: plt.figure(figsize=(8,6))
          plt.scatter(range(a.shape[0]), np.sort(a.logerror.values))
          plt.xlabel('index', fontsize= 12)
          plt.ylabel('logerror', fontsize=12)
          plt.show()
```

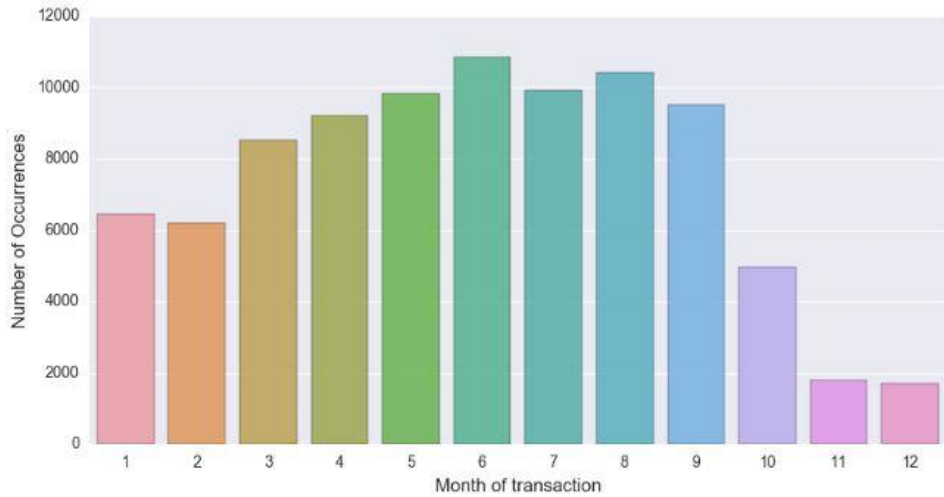


Add a new column as transaction to the clean file and plot its distribution per month

Get number of transactions per month and plot its distribution

```
In [213]: a['transactiondate']= pd.to_datetime(a['transactiondate'])
          a['transaction_month'] = a['transactiondate'].dt.month
          cnt_srs = a['transaction_month'].value_counts()
```

```
In [214]: plt.figure(figsize=(10,5))
          sns.barplot(cnt_srs.index,cnt_srs.values,alpha=0.8)
          plt.xlabel('Month of transaction', fontsize=12)
          plt.ylabel('Number of Occurrences', fontsize=12)
          plt.show()
```



Divide latitude and longitude values by 1,000,000 to get the valid data and plot its distribution

convert latitude and longitude in valid values and plot its distribution

```
In [215]: data_to_csv['latitude'] = data_to_csv['latitude']/1000000

C:\Users\smits_000\Anaconda3\lib\site-packages\ipykernel\__main__.py:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html#indexing-view-versus-copy
  if __name__ == '__main__':
```

```
In [216]: data_to_csv['longitude'] = data_to_csv['longitude']/1000000

C:\Users\smits_000\Anaconda3\lib\site-packages\ipykernel\__main__.py:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html#indexing-view-versus-copy
  if __name__ == '__main__':
```

```
In [217]: plt.figure(figsize=(12,12))
          sns.jointplot(x=data_to_csv.latitude.values, y=data_to_csv.longitude.values, size=10, color='brown')
          plt.ylabel('Longitude', fontsize=12)
          plt.xlabel('Latitude', fontsize=12)
          plt.show()

<matplotlib.figure.Figure at 0x20085420d30>
```

Data Wrangling

Data Wrangling

```
In [204]: a['parcelid'] = a['parcelid'].astype(int)

In [205]: a = a.dropna(subset=['regionidzip'])

In [206]: a = a.dropna(subset=['yearbuilt'])

In [207]: a = a.dropna(subset=['structuretaxvaluedollarcnt'])

In [208]: data_to_csv = a.dropna(subset=['regionidcity'])
```

Data Ingestion and Docker:

Pull the docker image by executing following command:

docker pull vipshah/ads-assginment2:final

```
smit@smit_shah_PC MINGW64 ~/Downloads/ADS_assignment2-master/ADS_assignment2-master/dockerize
$ docker pull vipshah/ads-assignment2:final
final: Pulling from vipshah/ads-assignment2
Digest: sha256:ba2bc4abb2980a6bd7590b593e8ffe3ae354fa49001b1c9abaceb4acaa155812
Status: Image is up to date for vipshah/ads-assignment2:final
```

Create a new container and start new container

```
smit@smit_shah_PC MINGW64 ~/Downloads/ADS_assignment2-master/ADS_assignment2-master/dockerize
$ docker create --name="final_ct" vipshah/ads-assignment2:final
4211f0bd7d07737c0c632079c3a753251127679722300f14ab6a4edc6cb6734d

smit@smit_shah_PC MINGW64 ~/Downloads/ADS_assignment2-master/ADS_assignment2-master/dockerize
$ docker start -i final_ct
done
```

This will upload cleaned data file on Amazon s3 bucket

Upload | Create folder | More | US East (N. Virginia)

Viewing 1 to 2

Name	Last modified	Size	Storage class
Clean.csv	Jul 8, 2017 6:48:19 PM	12.9 MB	Standard
properties_2016.csv	Jul 1, 2017 4:26:38 PM	618.8 MB	Standard

Viewing 1 to 2

Part 2: Upload data on a cloud database(MongoDB Atlas)

```
app = Flask(__name__)

# insert your connection details here

MONGO_URL = 'mongodb://joshisn:<password>@cluster0-shard-00-00-0phxm.mongodb.net:27017,cluster0-shard-00-01-0phxm.mongodb.net:27017,cluster0-shard-00-02-0phxm.mongodb.net:27017/DB?ssl

# connect to the MongoDB server

client = MongoClient(MONGO_URL)
print(client)
# connect to the default database within the server

db = client["DB"]
```

Part3: Create a REST API to serve the data

Hosted flask application on IBM bluemix

Cf push

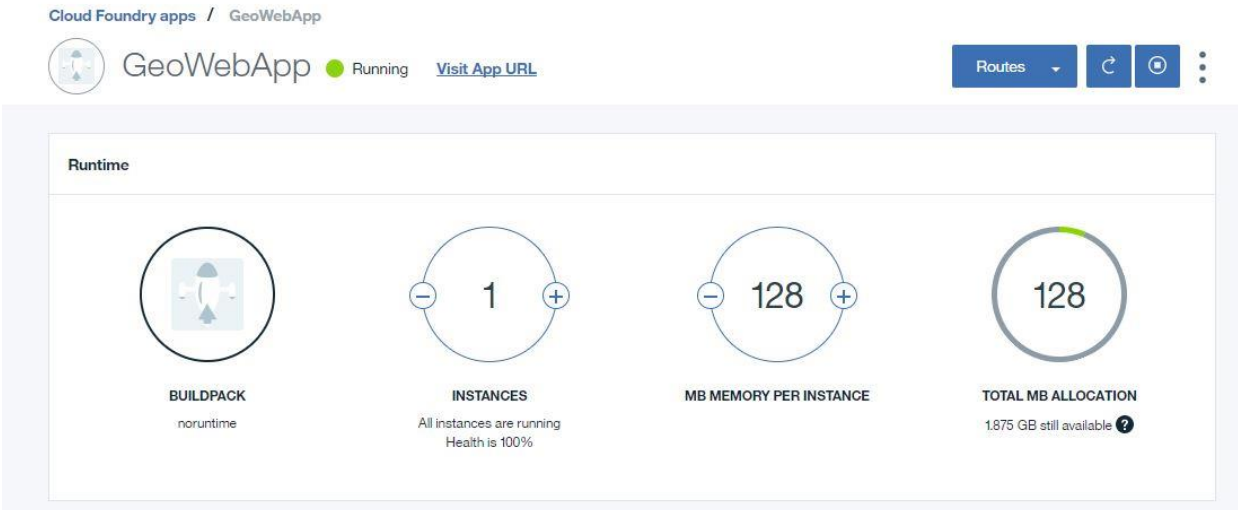
```
Microsoft Windows [Version 10.0.15063]
(c) 2017 Microsoft Corporation. All rights reserved.

C:\Users\Snigdha>cd C:\Users\Snigdha\Documents\ADS\Assignment2\New\ADS_assignment2\web_app\get-started-python

C:\Users\Snigdha\Documents\ADS\Assignment2\New\ADS_assignment2\web_app\get-started-python>cf push
Using manifest file C:\Users\Snigdha\Documents\ADS\Assignment2\New\ADS_assignment2\web_app\get-started-python\ma

Updating app GeoWebApp in org joshi.sn@husky.neu.edu / space ADS_Snigdha as joshi.sn@husky.neu.edu...
OK

Uploading GeoWebApp...
Uploading app files from: C:\Users\Snigdha\Documents\ADS\Assignment2\New\ADS_assignment2\web_app\get-started-pyt
Uploading 3.9K, 9 files
Done uploading
```

```
OK

App GeoWebApp was started using this command `python hello.py`

Showing health and status for app GeoWebApp in org joshi.sn@husky.neu.edu / space ADS_Snigdha as joshi.sn@husky.neu.edu...
OK

requested state: started
instances: 1/1
usage: 128M x 1 instances
urls: geowebapp-strifeful-groundsheet.mybluemix.net
last uploaded: Sat Jul 8 23:58:52 UTC 2017
stack: cflinuxfs2
buildpack: python 1.5.15

     state     since                  cpu    memory          disk           details
#0   running   2017-07-08 08:00:43 PM   0.0%   37.2M of 128M   286.4M of 1G

C:\Users\Snigdha\Documents\ADS\Assignment2\New\ADS_assignment2\web_app\get-started-python>
```

All Apps (2)

Cloud Foundry Apps 128 MB/2 GB Used

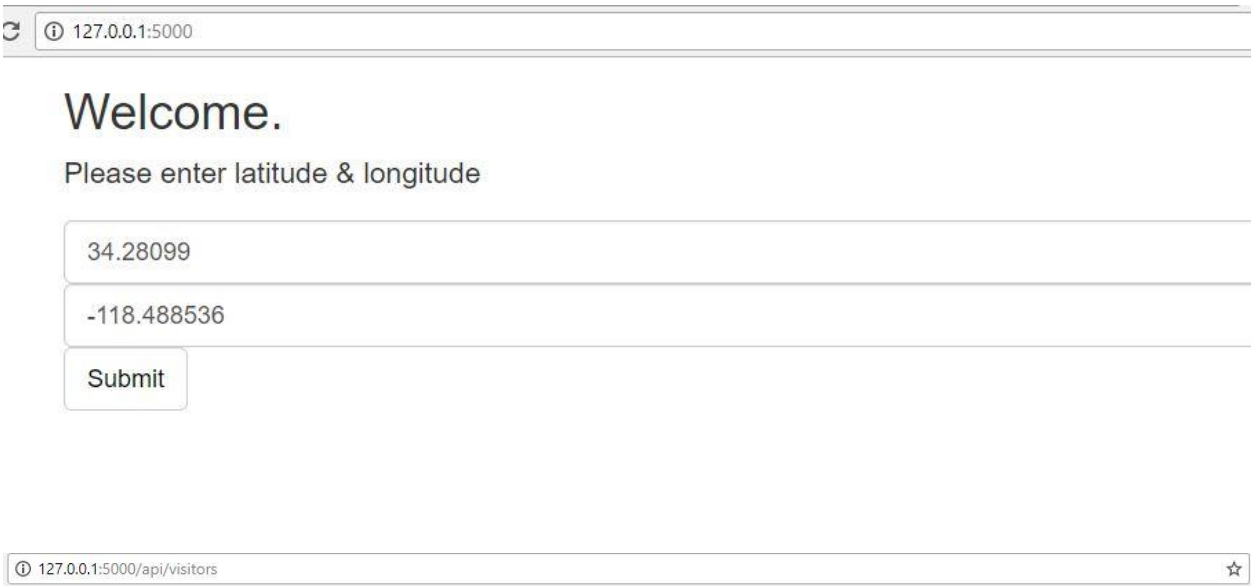
NAME	ROUTE	MEMORY (MB)	INSTANCES	RUNNING	STATE
ads_team7	asign2.mybluemix.net	128	1	0	Stopped
GeoWebApp	geowebapp-strifeful-groundsheet.mybluemi:	128	1	1	Running

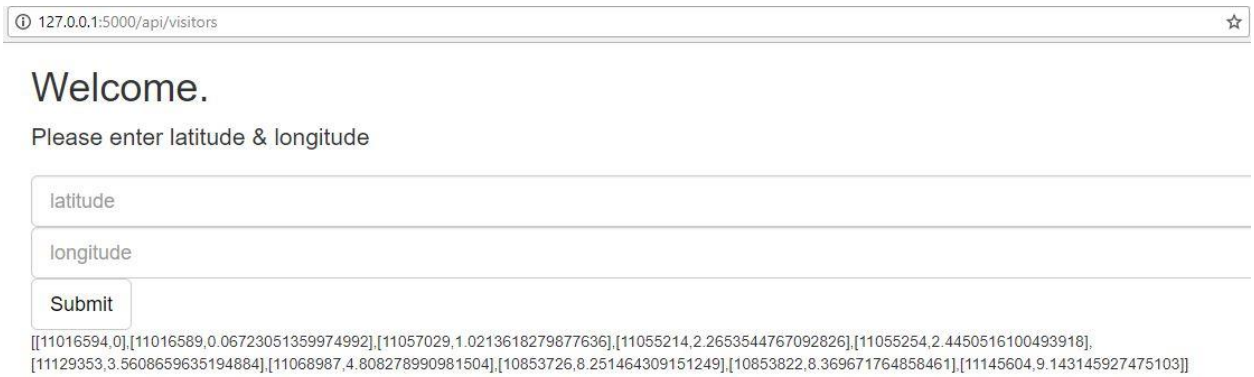


Part4: Geospatial Search

GET and POST method logic to get 10 closest homes from the latitude and longitude values

```
@app.route('/api/visitors', methods=['GET', 'POST'])
def get_visitor():
    logging.info("Inside method")
    logging.info("***************************")
    c = float(request.form['lat'])
    d = float(request.form['lon'])
    lc= c-0.00005
    hc= c+0.00005
    logging.info("lc"+ str(lc))
    ld= d-0.00010
    hd= d+0.00010
    logging.info(str(ld)+"ld")
    collection = db.abc.find({'latitude':  { '$gt' : lc , '$lt' : hc }})
    collectionl = db.abc.find({'longitude':  { '$gt' : ld , '$lt' : hd }})
    p={}
    q={}
    o = (c, d)
    for a in collection:
        n = (a['latitude'], a['longitude'])
        p[a['parcelid']]=vincenty(o, n).miles
    for b in collectionl:
        n= (b['latitude'],b['longitude'])
        q[b['parcelid']] =vincenty(o, n).miles
    z = {**p, **q}
    zl=sorted(z.items(), key=lambda value: value[1])

    logging.info("json"+json.dumps(zl[:10]))
    return render_template('index.html' , pi =json.dumps(zl[:10]))
```


127.0.0.1:5000
Welcome.
Please enter latitude & longitude
34.28099
-118.488536
Submit


127.0.0.1:5000/api/visitors
Welcome.
Please enter latitude & longitude
latitude
longitude
Submit
[[11016594,0],[11016589,0.067230513599974992],[11057029,1.0213618279877636],[11055214,2.2653544767092826],[11055254,2.4450516100493918],
[11129353,3.5608659635194884],[11068987,4.808278990981504],[10853726,8.251464309151249],[10853822,8.369671764858461],[11145604,9.143145927475103]]