In [1]:
```python
import pandas as pd
import numpy as np
import plotly.express as px
import geopandas as gpd
from geopandas import GeoDataFrame
import seaborn as sns
import matplotlib.pyplot as plt
```

In [2]:
```python
data = pd.read_csv('E://Ginu_StudyMaterials//Sem2//Dissertation//PRP.csv')
```

In [3]:
```python
data
```

Out[3]:

	date_of_sale	address	postal_code	county	price	market_price	VAT_exclusive	property_description	property_size_description	province	month_year	year	mont
0	01/01/2010	5 Braemor Drive, Churchtown, Co.Dublin	NaN	Dublin	343000	No	No	Second-Hand Dwelling house /Apartment	NaN	Leinster	2010-01	2010	
1	03/01/2010	134 Ashewood Walk, Summerhill Lane, Portlaoise	NaN	Laois	185000	No	Yes	New Dwelling house /Apartment	greater than or equal to 38 sq metres and less...	Leinster	2010-03	2010	
2	04/01/2010	1 Meadow Avenue, Dundrum, Dublin 14	NaN	Dublin	438500	No	No	Second-Hand Dwelling house /Apartment	NaN	Leinster	2010-04	2010	
3	04/01/2010	1 The Haven, Mornington	NaN	Meath	400000	No	No	Second-Hand Dwelling house /Apartment	NaN	Leinster	2010-04	2010	
4	04/01/2010	11 Melville Heights, Kilkenny	NaN	Kilkenny	160000	No	No	Second-Hand Dwelling house /Apartment	NaN	Leinster	2010-04	2010	
...	...	...	...	...	...	...	...	...	...	...	...	...	.
516581	28/01/2022	LACKEN, MULTYFARNHAM, MULLINGAR	NaN	Westmeath	305000	No	No	Second-Hand Dwelling house /Apartment	NaN	Leinster	2022-01	2022	
516582	28/01/2022	LARCH HILL, COLMAN, FETHARD	NaN	Tipperary	300000	No	No	Second-Hand Dwelling house /Apartment	NaN	Munster	2022-01	2022	
516583	28/01/2022	SHERRYS WOOD, BELLEWSTOWN, CO MEATH	NaN	Meath	450000	No	No	Second-Hand Dwelling house /Apartment	NaN	Leinster	2022-01	2022	
516584	28/01/2022	ST JUDES, STONEYFORD, KILKENNY	NaN	Kilkenny	242000	No	No	Second-Hand Dwelling house /Apartment	NaN	Leinster	2022-01	2022	
516585	28/01/2022	SYLVAN, DUBLIN ROAD, BRAY	NaN	Wicklow	620000	No	No	Second-Hand Dwelling house /Apartment	NaN	Leinster	2022-01	2022	

516586 rows × 13 columns

In [4]:
```python
counties = data['county'].unique()
counties
```

Out[4]:
```
array(['Dublin', 'Laois', 'Meath', 'Kilkenny', 'Limerick', 'Carlow',
       'Cork', 'Clare', 'Sligo', 'Cavan', 'Tipperary', 'Wicklow',
       'Roscommon', 'Wexford', 'Mayo', 'Donegal', 'Longford', 'Galway',
       'Offaly', 'Kildare', 'Waterford', 'Louth', 'Kerry', 'Westmeath',
       'Monaghan', 'Leitrim'], dtype=object)
```

In [5]:
```python
median_per_county = [data['price'][data['county']==county].median() for county in counties]

median_per_county = np.asarray(median_per_county)
median_per_county
```

Out[5]:
```
array([308370., 139995., 233480., 160000., 150000., 140000., 200000.,
       140000., 110000., 104225., 125000., 275000.,  89000., 150000.,
       110000., 101743.,  84000., 180000., 130000., 250000., 142000.,
       170000., 142500., 138000., 118000.,  87000.])
```

In [6]:
```python
q=[]
for price in median_per_county:
    x = price
    q.append(x)
```

In [7]:
```python
county_price = pd.DataFrame(q)
```

In [8]:
```python
county_price['county'] = counties
```

In [9]:
```python
county_price.rename(columns={0:'price'},inplace=True)
```

In [10]:
```python
from opencage.geocoder import OpenCageGeocode
key = '40d783cbf75143b48b8528d1804a3ccd'  # get api key from:  https://opencagedata.com

geocoder = OpenCageGeocode(key)
```

In [11]:
```python
list_lat = []   # create empty lists

list_long = []

for index, row in county_price.iterrows(): # iterate over rows in dataframe


    City = row['county']
    #State = row['province']
    query = str(City)
    #loc = row['temp_add']
    #query = str(loc)

    results = geocoder.geocode(query)
    lat = results[0]['geometry']['lat']
    long = results[0]['geometry']['lng']

    list_lat.append(lat)
    list_long.append(long)

# create new columns from lists

county_price['lat'] = list_lat

county_price['lon'] = list_long
```

In [12]:
```python
county_price
```

Out[12]:

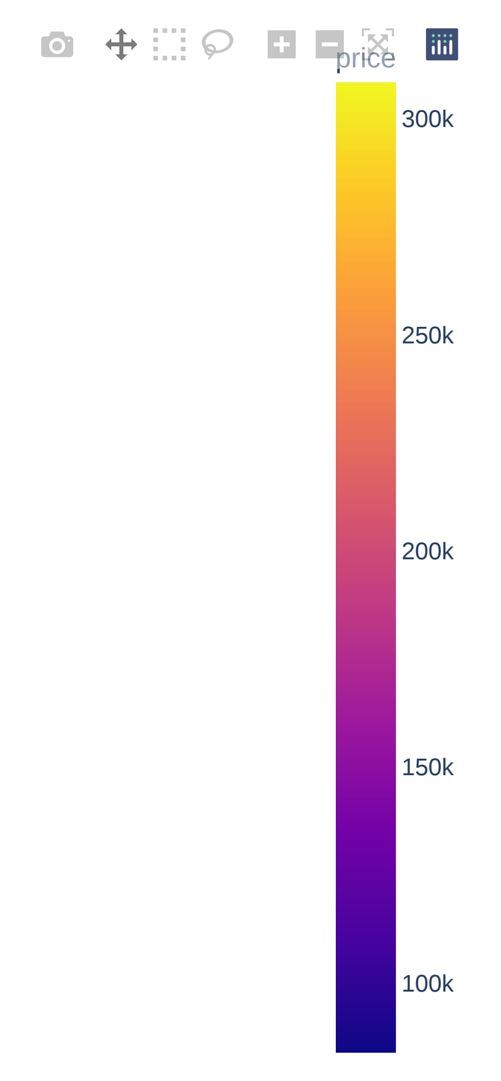
	price	county	lat	lon
0	308370.0	Dublin	53.349764	-6.260273
1	139995.0	Laois	52.998458	-7.398034
2	233480.0	Meath	53.649784	-6.588529
3	160000.0	Kilkenny	52.651022	-7.248495
4	150000.0	Limerick	52.661252	-8.630124
5	140000.0	Carlow	52.690789	-6.825145
6	200000.0	Cork	51.897077	-8.465467
7	140000.0	Clare	52.857258	-8.937436
8	110000.0	Sligo	54.192986	-8.730543
9	104225.0	Cavan	48.672173	-3.345742
10	125000.0	Tipperary	52.684821	-7.898128
11	275000.0	Wicklow	52.958147	-6.381971
12	89000.0	Roscommon	44.332693	-84.616516
13	150000.0	Wexford	52.460187	-6.606516
14	110000.0	Mayo	53.908706	-9.298305
15	101743.0	Donegal	54.920754	-7.952385
16	84000.0	Longford	53.731985	-7.695351
17	180000.0	Galway	53.274412	-9.049060
18	130000.0	Offaly	53.136172	-7.810341
19	250000.0	Kildare	53.154364	-6.818418
20	142000.0	Waterford	52.244963	-7.101788
21	170000.0	Louth	53.906285	-6.532050
22	142500.0	Kerry	52.145334	-9.517401
23	138000.0	Westmeath	53.557790	-7.347856
24	118000.0	Monaghan	54.161066	-6.946365
25	87000.0	Leitrim	54.140162	-8.052478

First method

In [13]:
```python
def df_to_geojson(county_price, properties, lat='lat', lon='lon'):
    # create a new python dict to contain our geojson data, using geojson format
    geojson = {'type':'FeatureCollection', 'features':[]}

    # loop through each row in the dataframe and convert each row to geojson format
    for _, row in county_price.iterrows():
        # create a feature template to fill in
        feature = {'type':'Feature',
                   'properties':{},
                   'geometry':{'type':'Point',
                               'coordinates':[]}}

        # fill in the coordinates
        feature['geometry']['coordinates'] = [row[lon],row[lat]]

        # for each column, get the value and add it as a new feature property
        for prop in properties:
            feature['properties'][prop] = row[prop]

        # add this feature (aka, converted dataframe row) to the list of features inside our dict
        geojson['features'].append(feature)

    return geojson
```

In [14]:
```python
cols = ['price', 'county', 'lat', 'lon']
geojson = df_to_geojson(county_price, cols)
```

In [15]:
```python
#geojson = county_price.election_geojson()
#geojson = df_to_geojson(county_price)

fig = px.choropleth(county_price, geojson=geojson, color="price",
                    locations="county", featureidkey="properties.county",
                    projection="mercator"
                   )
fig.update_geos(fitbounds="locations", visible=False)
fig.update_layout(margin={"r":0,"t":0,"l":0,"b":0})
fig.show()
```



Second method

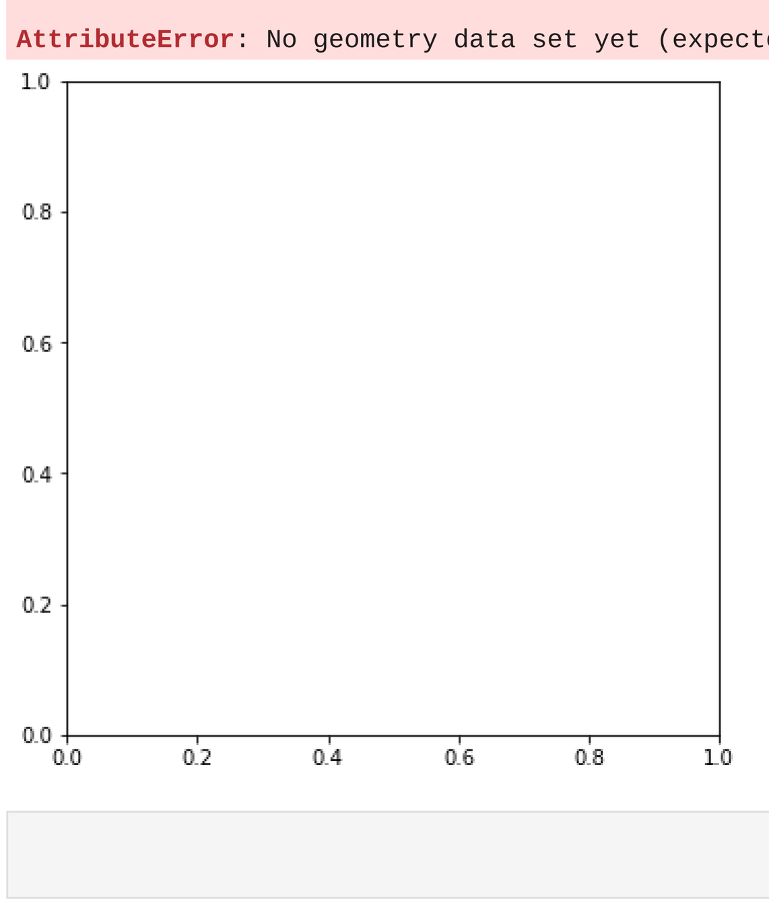
In [16]:
```python
merged = GeoDataFrame(county_price)
merged
```

Out[16]:

	price	county	lat	lon
0	308370.0	Dublin	53.349764	-6.260273
1	139995.0	Laois	52.998458	-7.398034
2	233480.0	Meath	53.649784	-6.588529
3	160000.0	Kilkenny	52.651022	-7.248495
4	150000.0	Limerick	52.661252	-8.630124
5	140000.0	Carlow	52.690789	-6.825145
6	200000.0	Cork	51.897077	-8.465467
7	140000.0	Clare	52.857258	-8.937436
8	110000.0	Sligo	54.192986	-8.730543
9	104225.0	Cavan	48.672173	-3.345742
10	125000.0	Tipperary	52.684821	-7.898128
11	275000.0	Wicklow	52.958147	-6.381971
12	89000.0	Roscommon	44.332693	-84.616516
13	150000.0	Wexford	52.460187	-6.606516
14	110000.0	Mayo	53.908706	-9.298305
15	101743.0	Donegal	54.920754	-7.952385
16	84000.0	Longford	53.731985	-7.695351
17	180000.0	Galway	53.274412	-9.049060
18	130000.0	Offaly	53.136172	-7.810341
19	250000.0	Kildare	53.154364	-6.818418
20	142000.0	Waterford	52.244963	-7.101788
21	170000.0	Louth	53.906285	-6.532050
22	142500.0	Kerry	52.145334	-9.517401
23	138000.0	Westmeath	53.557790	-7.347856
24	118000.0	Monaghan	54.161066	-6.946365
25	87000.0	Leitrim	54.140162	-8.052478

In [17]:
```python
# set a variable that will call whatever column we want to visualise on the map
variable =county_price['county']
# set the range for the choropleth
vmin, vmax = 120, 220
# create figure and axes for Matplotlib
fig, ax = plt.subplots(1, figsize=(10, 6))


# create map
merged.plot(column=variable, cmap='Blues', linewidth=0.8, ax=ax, edgecolor='0.8')

# remove the axis
ax.axis('off')

# add a title
ax.set_title('Price of houses', fontdict={'fontsize': '25', 'fontweight' : '3'})

# create an annotation for the data source
#ax.annotate('Source: Rajasthan Datastore, 2019',xy=(0.1, .08),  xycoords='figure fraction', horizontalalignment='left', verticalalignment='top', f

# Create colorbar as a legend
sm = plt.cm.ScalarMappable(cmap='Blues', norm=plt.Normalize(vmin=vmin, vmax=vmax))

# empty array for the data range
sm._A = []
# add the colorbar to the figure
cbar = fig.colorbar(sm)

fig.savefig('map_export.png', dpi=300)
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
~\AppData\Local\Temp/ipykernel_29612/1250432906.py in <module>
      8
      9 # create map
---> 10 merged.plot(column=variable, cmap='Blues', linewidth=0.8, ax=ax, edgecolor='0.8')
     11
     12 # remove the axis

~\anaconda3\lib\site-packages\geopandas\plotting.py in __call__(self, *args, **kwargs)
    923             kind = kwargs.pop("kind", "geo")
    924             if kind == "geo":
--> 925                 return plot_dataframe(data, *args, **kwargs)
    926             if kind in self._pandas_kinds:
    927                 # Access pandas plots

~\anaconda3\lib\site-packages\geopandas\plotting.py in plot_dataframe(df, column, cmap, color, ax, cax, categorical, legend, scheme, k, vmin, vmax, markersize, figsize, legend_kwds, categories, classification_kwds, missing_kwds, aspect, **style_kwds)
    787
    788     # decompose GeometryCollections
--> 789     geoms, multiindex = _flatten_multi_geoms(df.geometry, prefix="Geom")
    790     values = np.take(values, multiindex, axis=0)
    791     nan_idx = np.take(nan_idx, multiindex, axis=0)

~\anaconda3\lib\site-packages\pandas\core\generic.py in __getattr__(self, name)
   5485         ):
   5486             return self[name]
-> 5487         return object.__getattribute__(self, name)
   5488
   5489     def __setattr__(self, name: str, value) -> None:

~\anaconda3\lib\site-packages\geopandas\geodataframe.py in _get_geometry(self)
    184     def _get_geometry(self):
    185         if self._geometry_column_name not in self:
--> 186             raise AttributeError(
    187                 "No geometry data set yet (expected in"
    188                 " column '%s'.)" % self._geometry_column_name

AttributeError: No geometry data set yet (expected in column 'geometry'.)
```



In [ ]: