JOURNEY การทำงานกับข้อมูลชุดดังกล่าว

1. กำหนดคำสั่งที่เกี่ยวข้องกับการทำงาน โดย import คำสั่ง เช่น pandas, seaborn, numpy, matplotlib เป็นต้น

Mini-Project >> ทิศทางราคาค่าไฟฟ้าและปัจจัยที่เกี่ยวข้อง

In [1]:
```
import pandas as pd
import seaborn as sns
sns.set()
import numpy as np
import matplotlib as mpl
%matplotlib inline
import matplotlib.pyplot as plt
```

In [2]:
```
ft = pd.read_csv('Ft.csv', header= 0,
                         encoding= 'unicode_escape')
ft.info()
ft
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 21 entries, 0 to 20
Data columns (total 13 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   Year    21 non-null     int64
 1   Jan     21 non-null     float64
 2   Feb     21 non-null     float64
```

2. ดึงข้อมูล dataset ค่า Ft จากไฟล์ชื่อ “Ft.csv” เพื่อทำการวิเคราะห์ ซึ่งเป็นรายละเอียดค่า Ft (Unit: Strang/Kwh) ตั้งแต่ปี 2002-2022 แต่เมื่อเขียน code เพื่ออ่านข้อมูล เกิด UnicodeDecodeError ตามภาพข้างต้น

```
UnicodeDecodeError: 'utf-8' codec can't decode byte 0xa0 in position 86: invalid start byte
```

จึงแก้ไขด้วยการกำหนด encoding= 'unicode_escape'

In [2]:
```
ft = pd.read_csv('Ft.csv', header= 0,
                         encoding= 'unicode_escape')
ft.info()
ft
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 21 entries, 0 to 20
Data columns (total 13 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   Year    21 non-null     int64
 1   Jan     21 non-null     float64
 2   Feb     21 non-null     float64
 3   Mar     21 non-null     float64
 4   Apr     21 non-null     float64
 5   May     21 non-null     object
 6   June    21 non-null     object
 7   July    21 non-null     object
 8   Aug     21 non-null     object
 9   Sep     21 non-null     object
 10  Oct     21 non-null     object
 11  Nov     21 non-null     object
 12  Dec     21 non-null     object
dtypes: float64(4), int64(1), object(8)
memory usage: 2.3+ KB
```

3. ทำการ Cleansing data และจัดระเบียบตารางข้อมูลด้วยการ Melt Column รวมทั้ง กำหนดค่าแสดงผลที่จำเป็น โดยเลือก Index Column : Year และ Value : Jan - Dec

```
In [5]: ft.melt( id_vars=['Year'],
                 value_vars=['Jan','Feb','Mar','Apr','May','June','July','Aug','Sep','Oct','Nov','Dec'] )
```

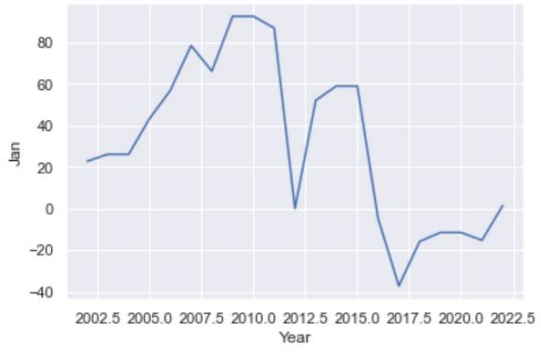
Out[5]:

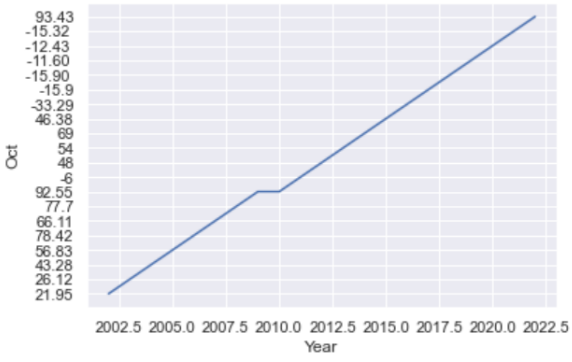
	Year	variable	value
0	2022	Jan	1.39
1	2021	Jan	-15.32
2	2020	Jan	-11.6
3	2019	Jan	-11.6
4	2018	Jan	-15.9
...	...	...	...
247	2006	Dec	78.42
248	2005	Dec	56.83
249	2004	Dec	43.28
250	2003	Dec	26.12
251	2002	Dec	21.95

4. นำข้อมูลที่เหลือมา Plot กราฟเส้น โดยเลือกค่าเดือน ม.ค.(Jan) และ ต.ค. (Oct) ซึ่งเป็นตัวแทนข้อมูลในช่วงต้นปีและปลายปีมา Plot เนื่องจาก ค่า Ft มีแนวโน้มเท่ากันทุกๆ 4 เดือนในทุกปี จากตัวอย่าง raw data ปี 2022 ข้างต้น

	Year	Jan	Feb	Mar	Apr	May	June	July	Aug	Sep	Oct	Nov	Dec
0	2022	1.39	1.39	1.39	1.39	24.77	24.77	24.77	24.77	93.43	93.43	93.43	93.43

จะทำให้ได้กราฟเทียบเดือน ม.ค. และ ต.ค. ดังนี้





5. เพิ่มเติมการอ่านข้อมูล dataset จากไฟล์ชื่อ “การใช้ไฟฟ้าของทั้งประเทศ (จำแนกตามสาขา)”

การใช้ไฟฟ้าของทั้งประเทศ (จำแนกตาม Sector)

```
In [8]: Electricity_Usage = pd.read_csv('การใช้ไฟฟ้าของทั้งประเทศ (จำแนกตามสาขา).csv')
        Electricity_Usage
```

Out[8]:

	Year	Month	Sector	Quantity	UNIT
0	2002	January	Residential	1524.085177	GWh
1	2002	February	Residential	1597.466131	GWh
2	2002	March	Residential	1842.061202	GWh
3	2002	April	Residential	2048.720372	GWh
4	2002	May	Residential	1989.872169	GWh
...	...	...	...	...	...
1724	2022	July	Industrial	7635.906273	GWh
1725	2022	July	Government & Non-Profit	18.779860	GWh
1726	2022	July	Agriculture	29.637428	GWh
1727	2022	July	Other	364.262411	GWh

6. นำข้อมูลการใช้ไฟฟ้ามา Groupby ค่า 'Year', 'Month', 'Sector' กับ 'Quantity' โดยแสดงผลในรูปแบบค่าเฉลี่ย

```
In [9]: df_groupby = Electricity_Usage.groupby(['Year','Month','Sector'])[['Quantity']].agg('mean')
        df_groupby
```

Out[9]:

			Quantity
Year	Month	Sector	
2002	April	Agriculture	23.081002
		Business	1660.139789
		Free of Charge	85.951362
		Government & Non-Profit	286.972177
		Industrial	4233.868312
...	...	...	...
2022	May	Free of Charge	331.038929
		Government & Non-Profit	18.367340
		Industrial	7836.707937
		Other	363.044752
		Residential	5011.733738

7. ทำการจัดรูปแบบตารางใหม่ด้วยการใช้ Pivot Table

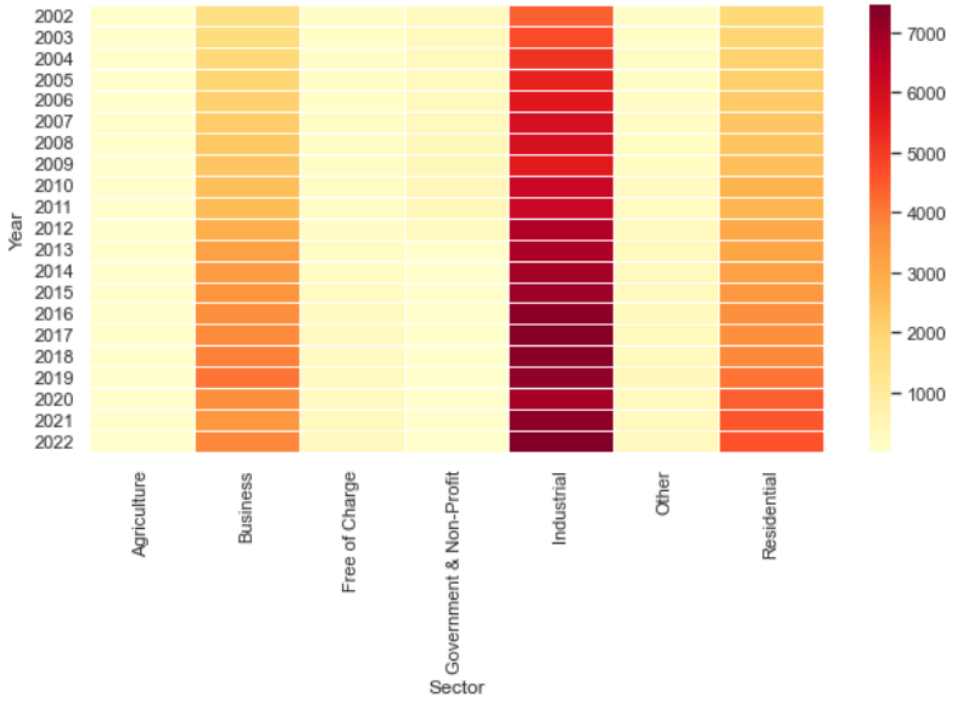
```
In [10]: df_1 = Electricity_Usage.pivot_table( index='Year', columns='Sector', values='Quantity', aggfunc='mean' )
         df_1
```

Out[10]:

Sector	Agriculture	Business	Free of Charge	Government & Non-Profit	Industrial	Other	Residential
Year							
2002	16.010886	1605.629233	79.940684	302.931799	4413.729848	92.413081	1830.264535
2003	18.989929	1723.659296	90.322828	308.425335	4728.925628	101.144001	1944.128831
2004	20.896843	1840.833651	98.674014	317.872740	5137.637334	130.984266	2044.860240
2005	20.779787	1928.788726	105.329786	320.706281	5447.116180	157.123404	2123.491356
2006	20.001473	2068.199752	128.001803	331.588211	5696.810946	174.723133	2237.281910

8. และทำการ Plot Heatmap เพื่อแสดงความสัมพันธ์ระหว่าง Year และ 7 Sectors เช่น ภาคเกษตรกรรม (Agriculture), ภาคธุรกิจ (Business), ภาคพิเศษ (Free of Charge), หน่วยงานภาครัฐและ Non-Profit (Government & Non-Profit), ภาคอุตสาหกรรม (Industrial), ที่พักอาศัย (Residential) และอื่นๆ (Other) เป็นต้น

```
In [11]: sns.set(rc={'figure.figsize':(10,5)})
         ax = sns.heatmap(df_1, linewidths=0.5,cmap="YlOrRd")
```



9. Merge และ Aggregate dataset “Ft” และ “การใช้ไฟฟ้าของทั้งประเทศ (จำแนกตามสาขา)” ด้วย Column Year เพื่อหาความสัมพันธ์ของทั้ง 2 datasets

```
In [12]: df_merge = pd.merge( ft, df_1, left_on='Year', right_on='Year', how='left' )
         df_merge
```

Out[12]:

	Year	Jan	Feb	Mar	Apr	May	June	July	Aug	Sep	Oct	Nov	Dec	Agriculture	Business	Free of Charge	Government & Non-Profit	Ind
0	2022	1.39	1.39	1.39	1.39	24.77	24.77	24.77	24.77	93.43	93.43	93.43	93.43	35.868516	3806.570659	328.220005	18.039581	7448.3
1	2021	-15.32	-15.32	-15.32	-15.32	-15.32	-15.32	-15.32	-15.32	-15.32	-15.32	-15.32	-15.32	33.162915	3460.739000	316.148993	16.712601	7202.2
2	2020	-11.60	-11.60	-11.60	-11.60	-11.60	-11.60	-11.60	-11.60	-12.43	-12.43	-12.43	-12.43	34.737636	3662.484869	298.828108	16.987290	6846.4
3	2019	-11.60	-11.60	-11.60	-11.60	-11.6	-11.60	-11.60	-11.60	-11.60	-11.60	-11.60	-11.60	38.985509	4094.026985	284.153418	17.581790	7175.3
4	2018	-15.90	-15.90	-15.90	-15.90	-15.90	-15.90	-15.90	-15.90	-15.90	-15.90	-15.90	-15.90	30.440770	3896.966927	271.242715	17.025103	7319.0
5	2017	-37.29	-37.29	-37.29	-37.29	-24.77	-24.77	-24.77	-24.77	-15.90	-15.9	-15.90	-15.90	24.852801	3758.335222	261.232127	16.486802	7314.3
6	2016	-4.80	-4.80	-4.80	-4.80	-33.29	-33.29	-33.29	-33.29	-33.29	-33.29	-33.29	-33.29	22.276942	3719.914985	246.887949	16.765389	7239.8
7	2015	58.96	58.96	58.96	58.96	49.61	49.61	49.61	49.61	46.38	46.38	46.38	46.38	32.223202	3538.806859	228.616534	14.916272	6998.6
8	2014	59.00	59.00	59.00	59.00	69	69	69	69	69	69	69	69	34.528433	3335.489500	215.193333	12.653881	6885.3
9	2013	52.04	52.04	52.04	52.04	46.92	46.92	46.92	46.92	54	54	54	54	29.468391	3239.457702	198.236785	12.404053	6765.6

```
In [13]: df_merge['Sum Sectors'] = (df_merge['Agriculture'] + df_merge['Business'] + df_merge['Free of Charge'] + df_merge['Government & N
         df_merge
```

10. จากนั้นทำการ Drop Column Sectors ย่อย หลังจากทำการ Sum Sectors เป็นค่ารวมเรียบร้อยแล้ว

```
In [14]: df_merge2 = df_merge.drop( columns=['Agriculture','Business','Free of Charge','Government & Non-Profit', 'Industrial'
         df_merge2
```

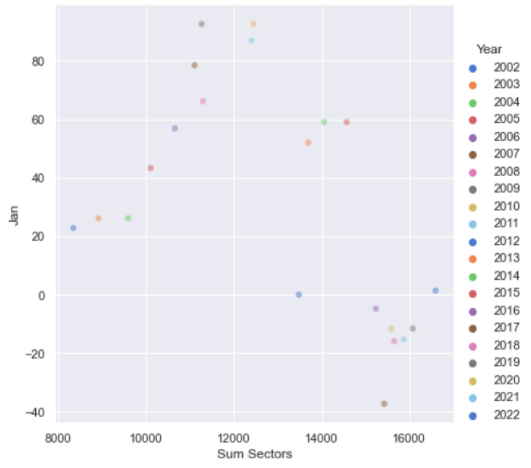
Out[14]:

	Year	Jan	Feb	Mar	Apr	May	June	July	Aug	Sep	Oct	Nov	Dec	Sum Sectors
0	2022	1.39	1.39	1.39	1.39	24.77	24.77	24.77	24.77	93.43	93.43	93.43	93.43	16598.320157
1	2021	-15.32	-15.32	-15.32	-15.32	-15.32	-15.32	-15.32	-15.32	-15.32	-15.32	-15.32	-15.32	15872.337996
2	2020	-11.60	-11.60	-11.60	-11.60	-11.60	-11.60	-11.60	-11.60	-12.43	-12.43	-12.43	-12.43	15587.206788
3	2019	-11.60	-11.60	-11.60	-11.60	-11.6	-11.60	-11.60	-11.60	-11.60	-11.60	-11.60	-11.60	16080.036243
4	2018	-15.90	-15.90	-15.90	-15.90	-15.90	-15.90	-15.90	-15.90	-15.90	-15.90	-15.90	-15.90	15652.654930
5	2017	-37.29	-37.29	-37.29	-37.29	-24.77	-24.77	-24.77	-24.77	-15.90	-15.9	-15.90	-15.90	15427.004441
6	2016	-4.80	-4.80	-4.80	-4.80	-33.29	-33.29	-33.29	-33.29	-33.29	-33.29	-33.29	-33.29	15237.285878
7	2015	58.96	58.96	58.96	58.96	49.61	49.61	49.61	49.61	46.38	46.38	46.38	46.38	14569.426127
8	2014	59.00	59.00	59.00	59.00	69	69	69	69	69	69	69	69	14057.115486
9	2013	52.04	52.04	52.04	52.04	46.92	46.92	46.92	46.92	54	54	54	54	13695.092895
10	2012	0.00	0.00	0.00	0.00	0	30	30	30	48	48	48	48	13481.581735
11	2011	86.88	86.88	86.88	86.88	95.81	95.81	-6	-6	-6	-6	-6	-6	12404.624093
12	2010	92.55	92.55	92.55	92.55	92.55	92.55	92.55	92.55	92.55	92.55	92.55	92.55	12441.760383
13	2009	92.55	92.55	92.55	92.55	92.55	92.55	92.55	92.55	92.55	92.55	92.55	92.55	11265.069391
14	2008	66.11	68.86	68.86	68.86	68.86	62.85	62.85	62.85	62.85	77.7	77.7	77.7	11293.357243
15	2007	78.42	73.42	73.42	73.42	73.42	68.42	68.42	68.42	68.42	66.11	66.11	66.11	11104.868967

และเริ่มทำการ Plot กราฟ เทียบเดือน ม.ค. และ ต.ค. เช่นเดียวกับกราฟแรกที่ทำผ่านมา เพื่อหาความสัมพันธ์ของค่า Ft และ Sum Sectors ในแต่ละปี (Year)

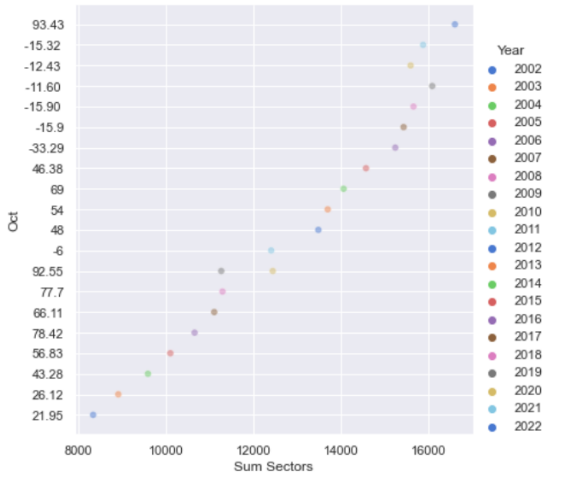
```
In [15]: sns.relplot(x="Sum Sectors", y="Jan", hue="Year",
                     sizes=(10, 100), alpha=.5, palette="muted",
                     height=6, data=df_merge2)
```

Out[15]: <seaborn.axisgrid.FacetGrid at 0x25830ebef40>



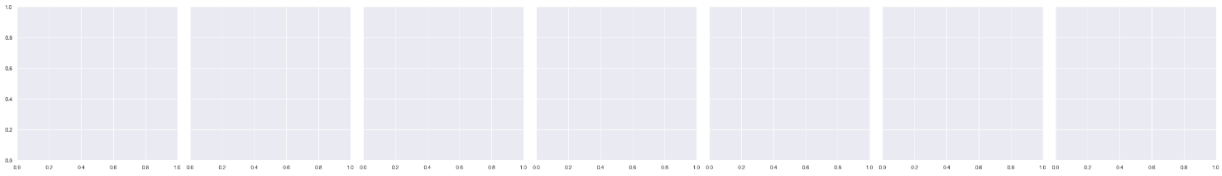

```
In [16]: sns.relplot(x="Sum Sectors", y="Oct", hue="Year",
                     sizes=(10, 100), alpha=.5, palette="muted",
                     height=6, data=df_merge2)
```

Out[16]: <seaborn.axisgrid.FacetGrid at 0x25830faac70>



11. ทำการดึงข้อมูล datasets เข้ามาเพิ่ม จากไฟล์ชื่อ “Power Generation by Type of Fuel.csv” และ “Import of Electricity.csv” ซึ่งนำมาจัดระเบียบข้อมูลด้วยการ Fillna(0), drop, sort_values และการเลือก row ด้วยการใช้ .loc เป็นต้น โดยระหว่างทำการ Plot กราฟก็จะเจอปัญหา Error ด้านต่างๆ เช่น การเลือกใช้กราฟที่เหมาะสมในการแสดงผลข้อมูล เนื่องจากกราฟหลายประเภทที่นำมาแสดงผลไม่เหมาะสม อ่านยาก ตัวแปรไม่แสดงผล และเกิดการทับซ้อนของข้อมูลในการแสดงผล เป็นต้น ดังนี้

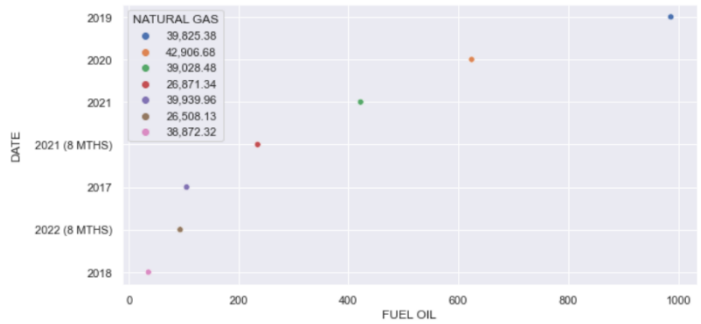
```
~\anaconda3\lib\site-packages\seaborn\regression.py in update_datalim(data, x, y, ax, **kws)
    628
    629     def update_datalim(data, x, y, ax, **kws):
--> 630         xys = np.asarray(data[[x, y]]).astype(float)
    631         ax.update_datalim(xys, updatey=False)
    632         ax.autoscale_view(scaley=False)

ValueError: could not convert string to float: '39,939.96'
```

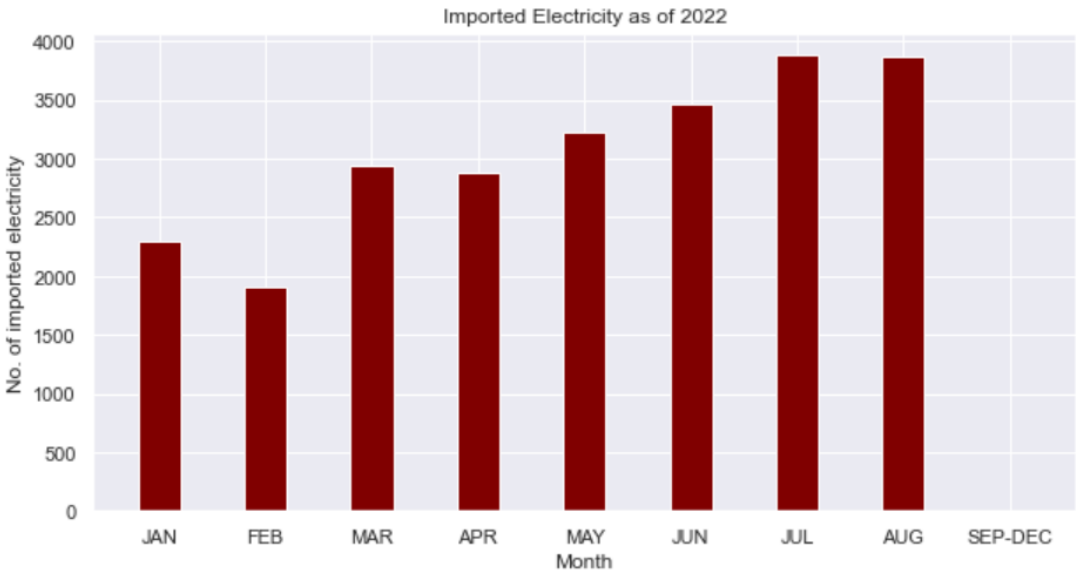


```
    380         with np.errstate(all="ignore"):
--> 381             result = func(self.values, **kwargs)
    382
    383         return self._split_op_result(result)

~\anaconda3\lib\site-packages\pandas\core\ops\array_ops.py in comparison_op(left, right, op)
    282
    283     elif is_object_dtype(lvalues.dtype) or isinstance(rvalues, str):
--> 284         res_values = comp_method_OBJECT_ARRAY(op, lvalues, rvalues)
    285
    286     else:

~\anaconda3\lib\site-packages\pandas\core\ops\array_ops.py in comp_method_OBJECT_ARRAY(op, x, y)
     71         result = libops.vec_compare(x.ravel(), y.ravel(), op)
     72     else:
---> 73         result = libops.scalar_compare(x.ravel(), y, op)
     74     return result.reshape(x.shape)
     75

~\anaconda3\lib\site-packages\pandas\_libs\ops.pyx in pandas._libs.ops.scalar_compare()

TypeError: '<' not supported between instances of 'str' and 'int'
```

ก่อนที่จะทำการแก้ไขด้วยการลอง Plot กราฟรูปแบบใหม่ๆ ให้ได้ความสัมพันธ์ที่สอดคล้องและเข้าใจง่ายยิ่งขึ้นไป

```
In [22]: sns.scatterplot(x=fuel4['FUEL OIL'], y=fuel4['DATE'], hue=fuel4['NATURAL GAS'])
Out[22]: <AxesSubplot:xlabel='FUEL OIL', ylabel='DATE'>
```




Imported Electricity as of 2022
No. of imported electricity
4000
3500
3000
2500
2000
1500
1000
500
0
JAN
FEB
MAR
APR
MAY
JUN
JUL
AUG
SEP-DEC
Month