In [1]:
```
# Packages loader

import matplotlib.pyplot as plt
import pandas as pd
import seaborn as sns
```

In [2]:
```
# List all the datasets within seaboarn

sns.get_dataset_names()
```

Out[2]:
```
['anagrams',
 'anscombe',
 'attention',
 'brain_networks',
 'car_crashes',
 'diamonds',
 'dots',
 'dowjones',
 'exercise',
 'flights',
 'fmri',
 'geyser',
 'glue',
 'healthexp',
 'iris',
 'mpg',
 'penguins',
 'planets',
 'seaice',
 'taxis',
 'tips',
 'titanic']
```

Diamond Dataset Explained

Carat - weight of a diamond

Cut - the cut quality with five possible values in increasing order: Fair, Good, Very Good, Premium, Ideal

Color - the color of a diamond with color codes from D (the best) to J (the worst)

Clarity - the clarity of a diamond with eight clarity codes

X - length of a diamond (mm)

Y - the height of a diamond (mm)

Z - depth of a diamond (mm)

Depth - total depth percentage calculated as Z / average(X, Y)

Table - the ratio of the height of a diamond to its widest point

Price - diamond price in dollars

In [3]:

```
# Loading diamonds dataset and creaing a sample of diamond dataset

df_diamonds = sns.load_dataset('diamonds')
df_diamonds_sample = df_diamonds.sample(2000)
```

In [4]:

```
# Diamond dataset shape

df_diamonds.shape
```

Out[4]:

```
(53940, 10)
```

In [5]:

```
# Displaying headers of dataset

df_diamonds.head()
```

Out[5]:

	carat	cut	color	clarity	depth	table	price	x	y	z
0	0.23	Ideal	E	SI2	61.5	55.0	326	3.95	3.98	2.43
1	0.21	Premium	E	SI1	59.8	61.0	326	3.89	3.84	2.31
2	0.23	Good	E	VS1	56.9	65.0	327	4.05	4.07	2.31
3	0.29	Premium	I	VS2	62.4	58.0	334	4.20	4.23	2.63
4	0.31	Good	J	SI2	63.3	58.0	335	4.34	4.35	2.75

In [6]:

```
# Learning about the missing information, data-types, non-null row count

df_diamonds.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 53940 entries, 0 to 53939
Data columns (total 10 columns):
 #   Column   Non-Null Count  Dtype
---  ------   --------------  -----
 0   carat    53940 non-null  float64
 1   cut      53940 non-null  category
 2   color    53940 non-null  category
 3   clarity  53940 non-null  category
 4   depth    53940 non-null  float64
 5   table    53940 non-null  float64
 6   price    53940 non-null  int64
 7   x        53940 non-null  float64
 8   y        53940 non-null  float64
 9   z        53940 non-null  float64
dtypes: category(3), float64(6), int64(1)
memory usage: 3.0 MB
```

In [7]:

```
# Calculating statistics for the categorical columns

print("Categorical Statistics For Diamonds:\n")
df_diamonds.describe(include=['category'])
```

```
Categorical Statistics For Diamonds:
```

Out[7]:

	cut	color	clarity
count	53940	53940	53940
unique	5	7	8
top	Ideal	G	SI1
freq	21551	11292	13065

In [8]:

```
# Calculating statistics for all the columns in diamonds dataset

print("All Statistics For Diamonds:\n")
df_diamonds.describe(include='all')
```

```
All Statistics For Diamonds:
```

Out[8]:

	carat	cut	color	clarity	depth	table	price	x	
count	53940.000000	53940	53940	53940	53940.000000	53940.000000	53940.000000	53940.000000	53
unique	NaN	5	7	8	NaN	NaN	NaN	NaN	
top	NaN	Ideal	G	SI1	NaN	NaN	NaN	NaN	
freq	NaN	21551	11292	13065	NaN	NaN	NaN	NaN	
mean	0.797940	NaN	NaN	NaN	61.749405	57.457184	3932.799722	5.731157	
std	0.474011	NaN	NaN	NaN	1.432621	2.234491	3989.439738	1.121761	
min	0.200000	NaN	NaN	NaN	43.000000	43.000000	326.000000	0.000000	
25%	0.400000	NaN	NaN	NaN	61.000000	56.000000	950.000000	4.710000	
50%	0.700000	NaN	NaN	NaN	61.800000	57.000000	2401.000000	5.700000	
75%	1.040000	NaN	NaN	NaN	62.500000	59.000000	5324.250000	6.540000	
max	5.010000	NaN	NaN	NaN	79.000000	95.000000	18823.000000	10.740000	

In [ ]:

```

```

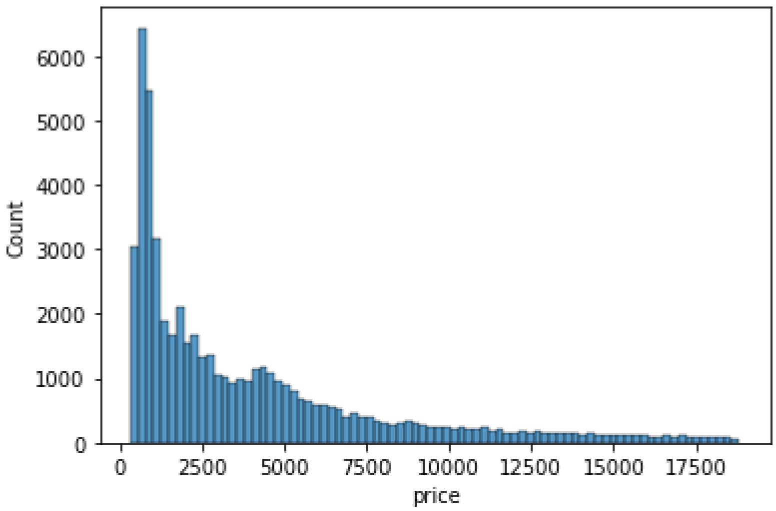
Visualization

In [9]:

```
# Histogram of the entire diamond dataset w.r.t. price

"""
https://seaborn.pydata.org/generated/seaborn.displot.html

seaborn.displot(data=None, *, x=None, y=None, hue=None, row=None, col=None, weights=Non
                log_scale=None, legend=True, palette=None, hue_order=None, hue_norm=Non
                row_order=None, col_order=None, height=5, aspect=1, facet_kws=None, **k
```

```
"""

sns.histplot(x=df_diamonds["price"])
```

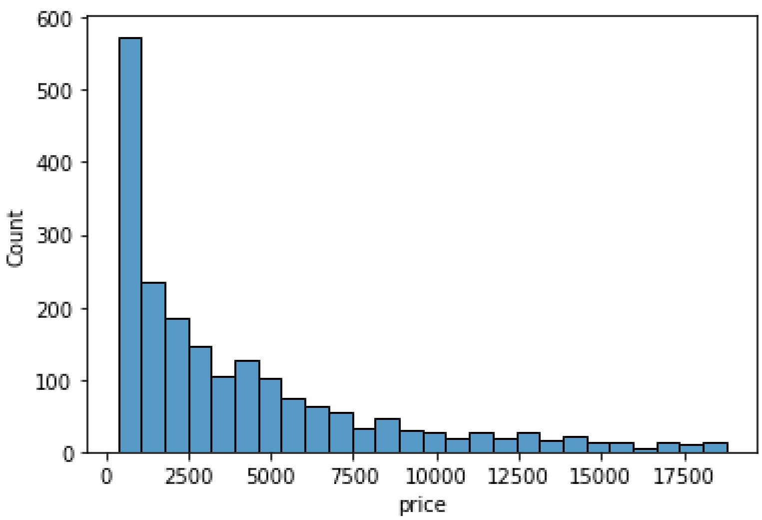
Out[9]: <AxesSubplot:xlabel='price', ylabel='Count'>



In [10]:

```
# Histogram of a sample set w.r.t. price

"""
https://seaborn.pydata.org/generated/seaborn.displot.html

seaborn.displot(data=None, *, x=None, y=None, hue=None, row=None, col=None, weights=Non
                log_scale=None, legend=True, palette=None, hue_order=None, hue_norm=Non
                row_order=None, col_order=None, height=5, aspect=1, facet_kws=None, **k
"""

sns.histplot(x = df_diamonds_sample["price"])
```

Out[10]: <AxesSubplot:xlabel='price', ylabel='Count'>

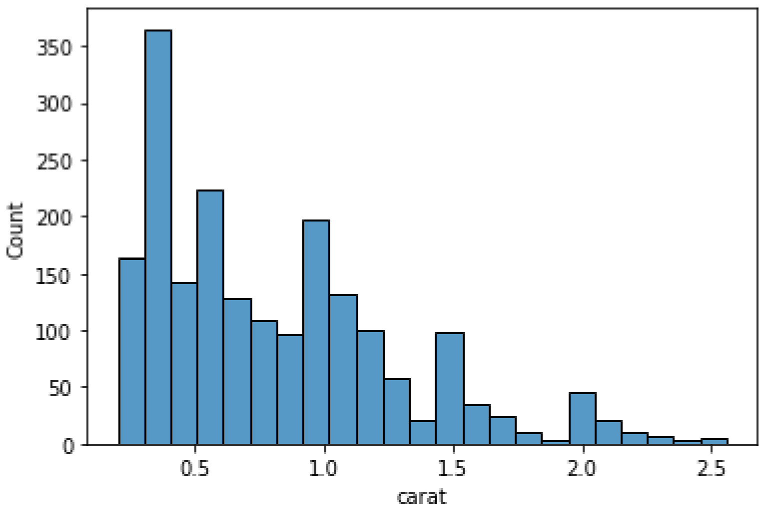


In [11]:

```
# Histplot of a sample set w.r.t. carat

"""
https://seaborn.pydata.org/generated/seaborn.displot.html
```

```
seaborn.displot(data=None, *, x=None, y=None, hue=None, row=None, col=None, weights=Non
                log_scale=None, legend=True, palette=None, hue_order=None, hue_norm=Non
                row_order=None, col_order=None, height=5, aspect=1, facet_kws=None, **k
"""

sns.histplot(df_diamonds_sample["carat"])
```

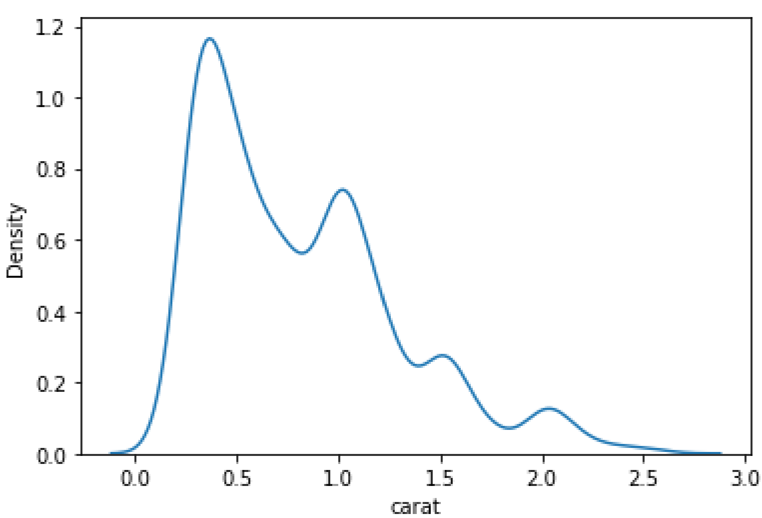
Out[11]: <AxesSubplot:xlabel='carat', ylabel='Count'>



In [12]:

```
# Kernel Density Estimate plot for carat for diamonds sample

"""
https://seaborn.pydata.org/generated/seaborn.kdeplot.html

seaborn.kdeplot(data=None, *, x=None, y=None, hue=None, weights=None, palette=None, hue
                fill=None, multiple='layer', common_norm=True, common_grid=False, cumul
                bw_adjust=1, warn_singular=True, log_scale=None, levels=10, thresh=0.05
                legend=True, cbar=False, cbar_ax=None, cbar_kws=None, ax=None, **kwargs
"""

sns.kdeplot(df_diamonds_sample["carat"])
```

Out[12]: <AxesSubplot:xlabel='carat', ylabel='Density'>

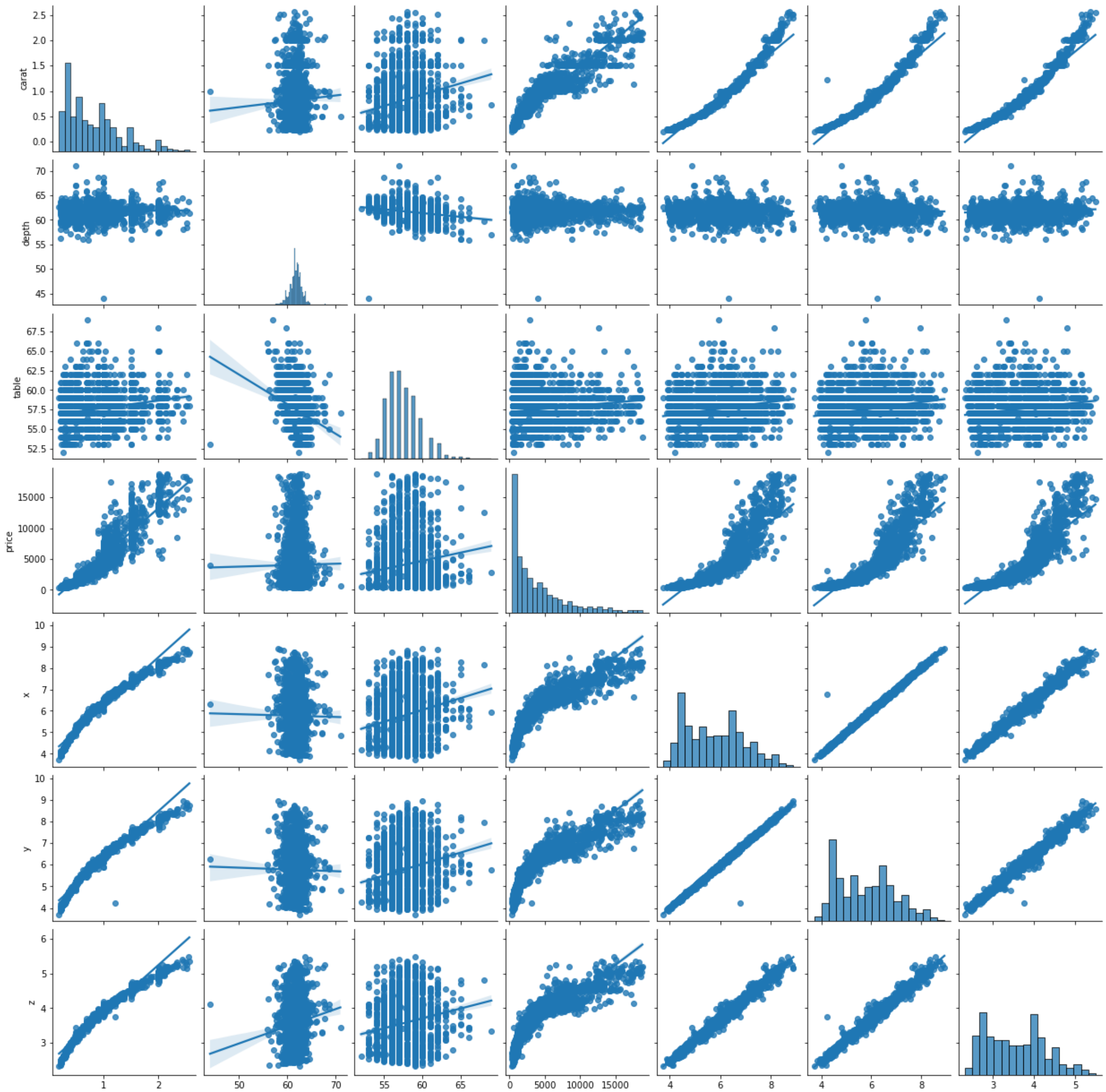


In [13]:

```
# Pairplot of a sample set of diamonds

"""
https://seaborn.pydata.org/generated/seaborn.pairplot.html

seaborn.pairplot(data, *, hue=None, hue_order=None, palette=None, vars=None, x_vars=Non
                 diag_kind='auto', markers=None, height=2.5, aspect=1, corner=False, dr
                 diag_kws=None, grid_kws=None, size=None)
"""

sns.pairplot(df_diamonds_sample, kind="reg")
```

Out[13]:

```
<seaborn.axisgrid.PairGrid at 0x24c06c18040>
```

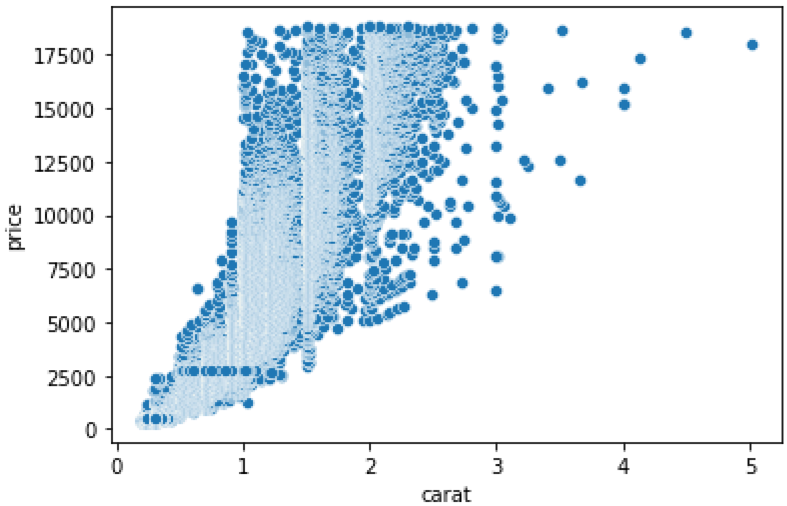


In [14]:

```
# Scatterplot of the entire dataset w.r.t. carat and price

"""
https://seaborn.pydata.org/generated/seaborn.scatterplot.html
```

```
seaborn.scatterplot(data=None, *, x=None, y=None, hue=None, size=None, style=None, pale
                    hue_norm=None, sizes=None, size_order=None, size_norm=None, markers
                    legend='auto', ax=None, **kwargs)
"""

sns.scatterplot(x = df_diamonds["carat"], y = df_diamonds["price"])
```

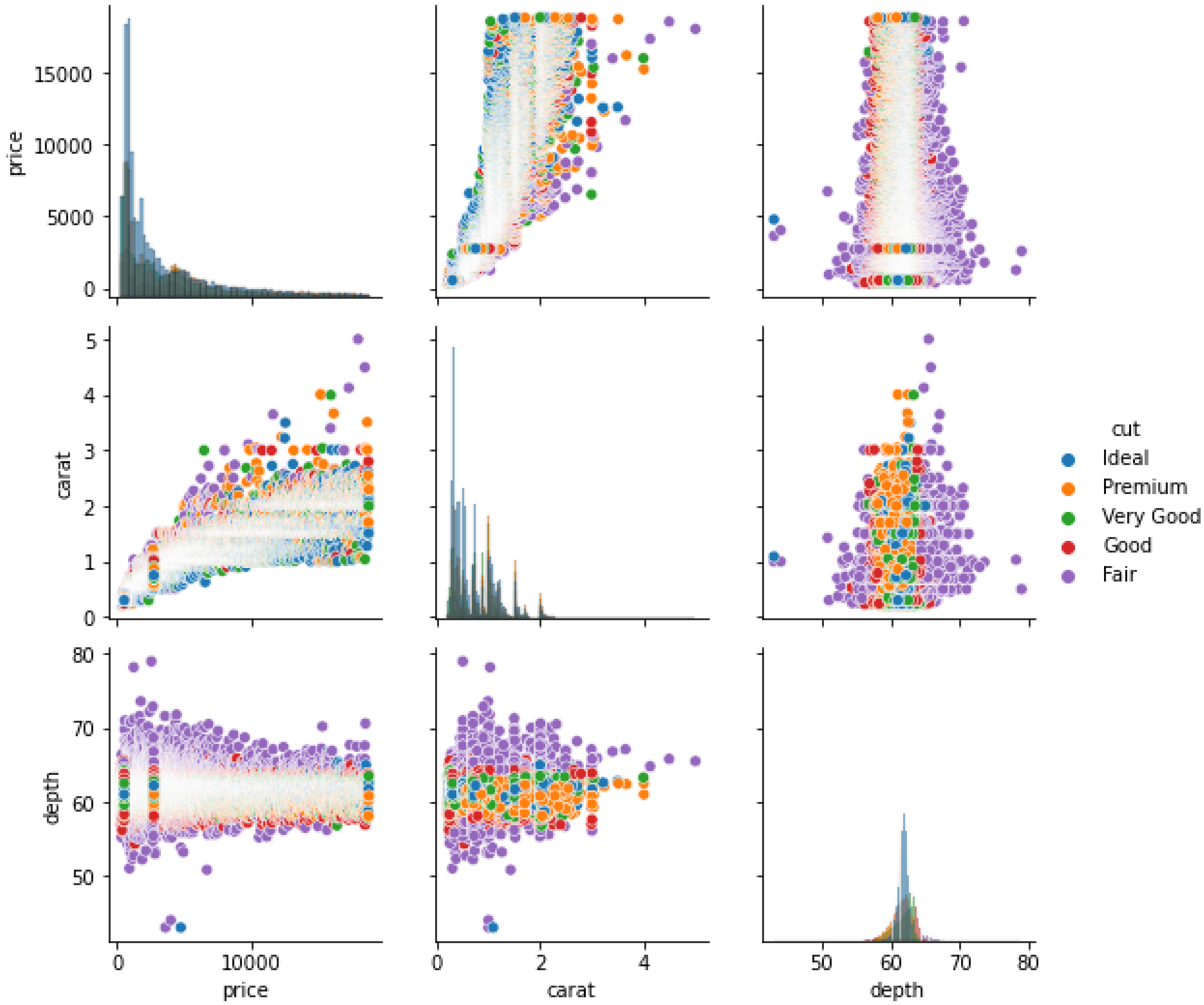
Out[14]: <AxesSubplot:xlabel='carat', ylabel='price'>



In [15]:

```
# Pairplotting by defining hue and considering all columns of diamonds dataframe

"""
https://seaborn.pydata.org/generated/seaborn.PairGrid.html

class seaborn.PairGrid(data, *, hue=None, vars=None, x_vars=None, y_vars=None, hue_orde
                       corner=False, diag_sharey=True, height=2.5, aspect=1, layout_pad
"""

pairplotting_hue = sns.PairGrid(df_diamonds[["price", "carat", "depth", "cut"]], hue="c

pairplotting_hue.map_diag(sns.histplot)
pairplotting_hue.map_offdiag(sns.scatterplot)
pairplotting_hue.add_legend()
```

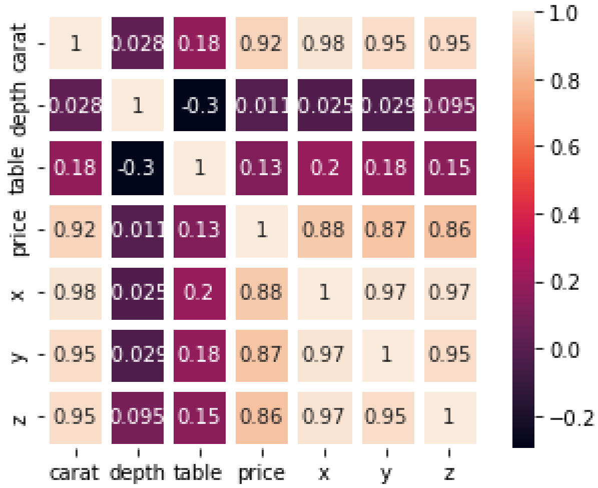
Out[15]: <seaborn.axisgrid.PairGrid at 0x24c09ae5e80>



In [16]:

```
# Correlation between all variables in the diamonds dataframe

"""
https://seaborn.pydata.org/generated/seaborn.heatmap.html

seaborn.heatmap(data, *, vmin=None, vmax=None, cmap=None, center=None, robust=False, an
                linewidths=0, linecolor='white', cbar=True, cbar_kws=None, cbar_ax=None
                yticklabels='auto', mask=None, ax=None, **kwargs)
"""

sns.heatmap(df_diamonds.corr(), square=True, annot=True, linewidths=5)
```

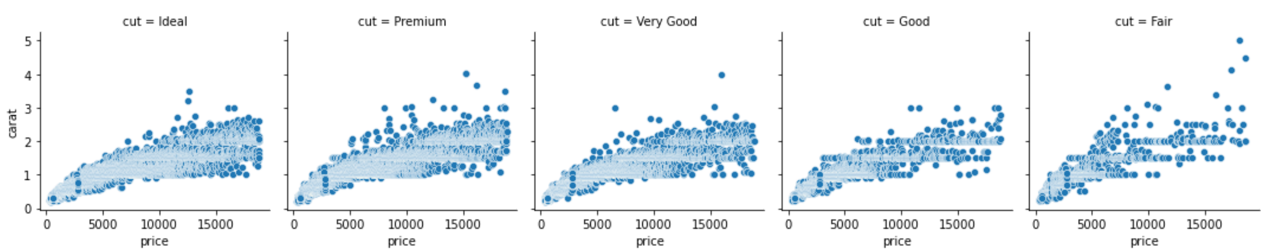
Out[16]: <AxesSubplot:>



In [17]:

```
# Facetgrid with one column as the type of cut

"""
https://seaborn.pydata.org/generated/seaborn.FacetGrid.html

class seaborn.FacetGrid(data, *, row=None, col=None, hue=None, col_wrap=None, sharex=Tr
                        palette=None, row_order=None, col_order=None, hue_order=None, h
                        legend_out=True, despine=True, margin_titles=False, xlim=None,
                        gridspec_kws=None)
"""

facetgrid_one_col = sns.FacetGrid(df_diamonds, col="cut")

facetgrid_one_col.map(sns.scatterplot, "price", "carat")
```

Out[17]: <seaborn.axisgrid.FacetGrid at 0x24c0e00d580>



In [18]:

```
# Facetgrid with a column and a row as color and cut

"""
https://seaborn.pydata.org/generated/seaborn.FacetGrid.html

class seaborn.FacetGrid(data, *, row=None, col=None, hue=None, col_wrap=None, sharex=Tr
                        palette=None, row_order=None, col_order=None, hue_order=None, h
                        legend_out=True, despine=True, margin_titles=False, xlim=None,
                        gridspec_kws=None)
"""

facetgrid_two_col = sns.FacetGrid(df_diamonds, col="color", row="cut")

facetgrid_two_col.map(sns.scatterplot, "price", "carat")
```

Out[18]:

```
<seaborn.axisgrid.FacetGrid at 0x24c0fb4f550>
```

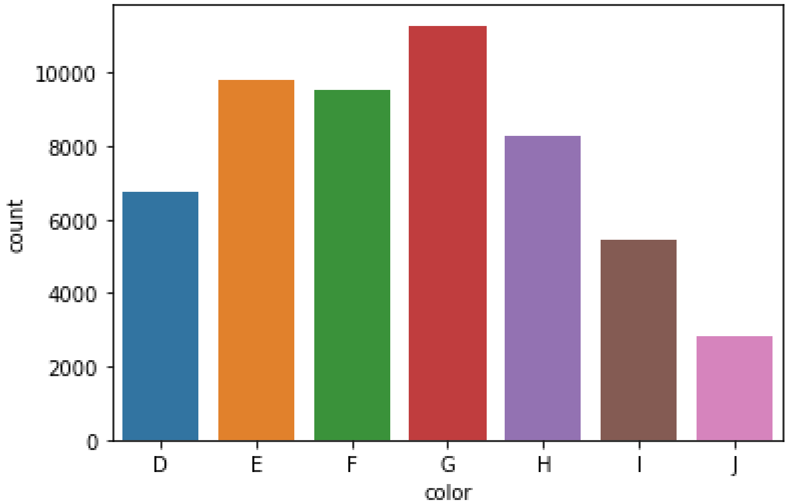
In [19]:

```python
# Countplot of color vs count in the entire diamond dataset

"""
https://seaborn.pydata.org/generated/seaborn.countplot.html

seaborn.countplot(data=None, *, x=None, y=None, hue=None, order=None, hue_order=None, o
                  saturation=0.75, width=0.8, dodge=True, ax=None, **kwargs)
"""

sns.countplot(df_diamonds["color"])
```

```
C:\Users\Lenovo\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pa
ss the following variable as a keyword arg: x. From version 0.12, the only valid positio
nal argument will be `data`, and passing other arguments without an explicit keyword wil
l result in an error or misinterpretation.
  warnings.warn(
```

Out[19]:

```
<AxesSubplot:xlabel='color', ylabel='count'>
```

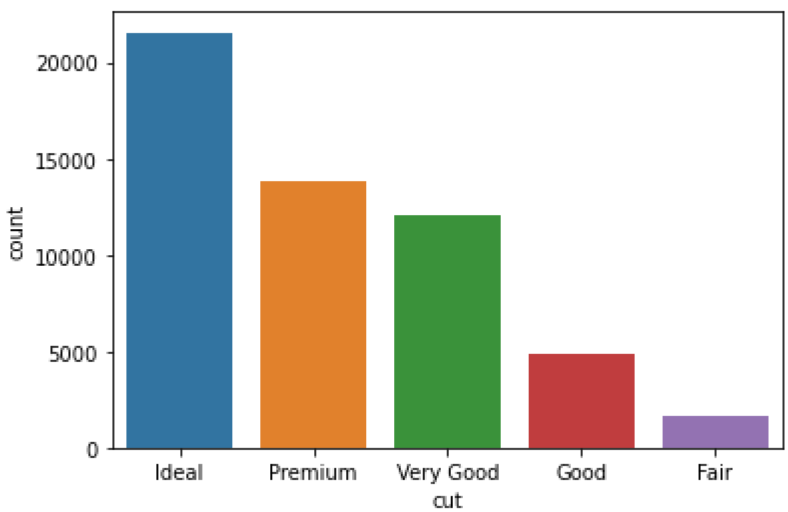


In [20]:

```
# Countplot of cut vs count in the entire diamond dataset

"""
https://seaborn.pydata.org/generated/seaborn.countplot.html

seaborn.countplot(data=None, *, x=None, y=None, hue=None, order=None, hue_order=None, o
                  saturation=0.75, width=0.8, dodge=True, ax=None, **kwargs)
"""

sns.countplot(df_diamonds["cut"])
```

Out[20]:

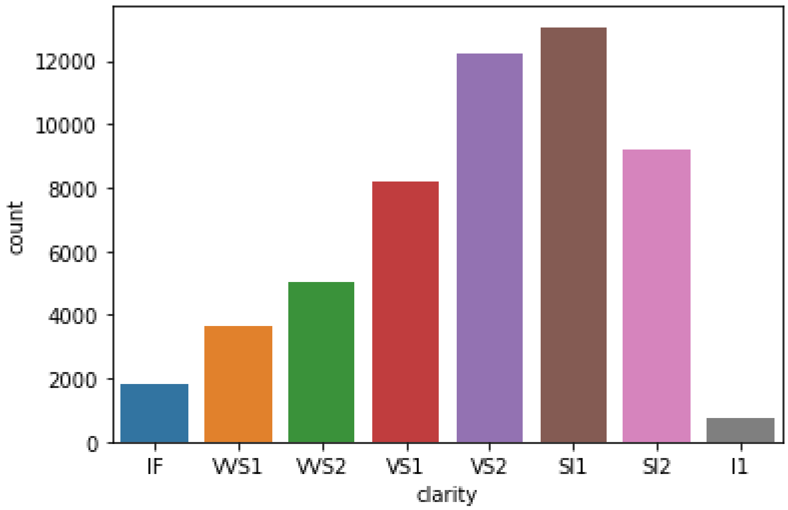
```
<AxesSubplot:xlabel='cut', ylabel='count'>
```



In [21]:

```
# Countplot of clarity vs count in the entire diamond dataset

"""
https://seaborn.pydata.org/generated/seaborn.countplot.html

seaborn.countplot(data=None, *, x=None, y=None, hue=None, order=None, hue_order=None, o
                  saturation=0.75, width=0.8, dodge=True, ax=None, **kwargs)
"""

sns.countplot(df_diamonds["clarity"])
```

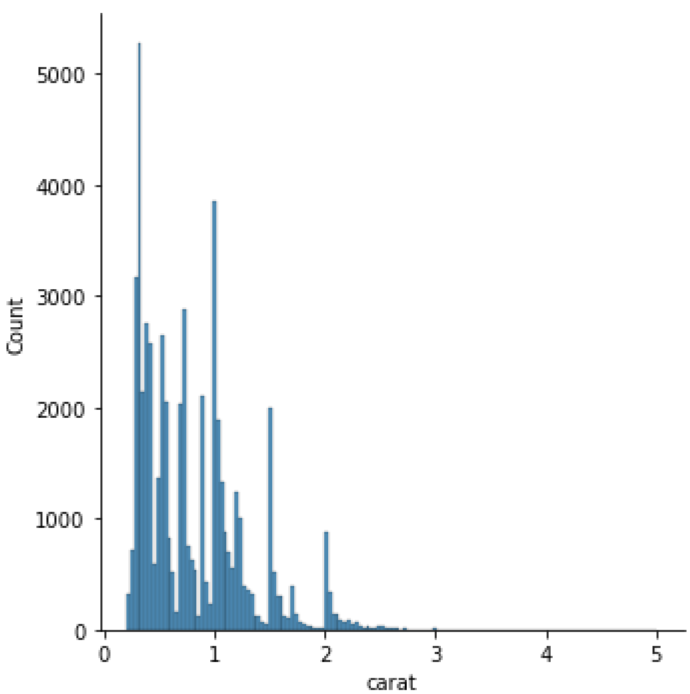
Out[21]: <AxesSubplot:xlabel='clarity', ylabel='count'>



In [22]:

```
# Displot of carat vs count in the entire diamond dataset

"""
https://seaborn.pydata.org/generated/seaborn.distplot.html

seaborn.distplot(a=None, bins=None, hist=True, kde=True, rug=False, fit=None, hist_kws=
                 fit_kws=None, color=None, vertical=False, norm_hist=False, axlabel=Non
"""

sns.displot(df_diamonds, x="carat")
```

Out[22]: <seaborn.axisgrid.FacetGrid at 0x24c1249f700>

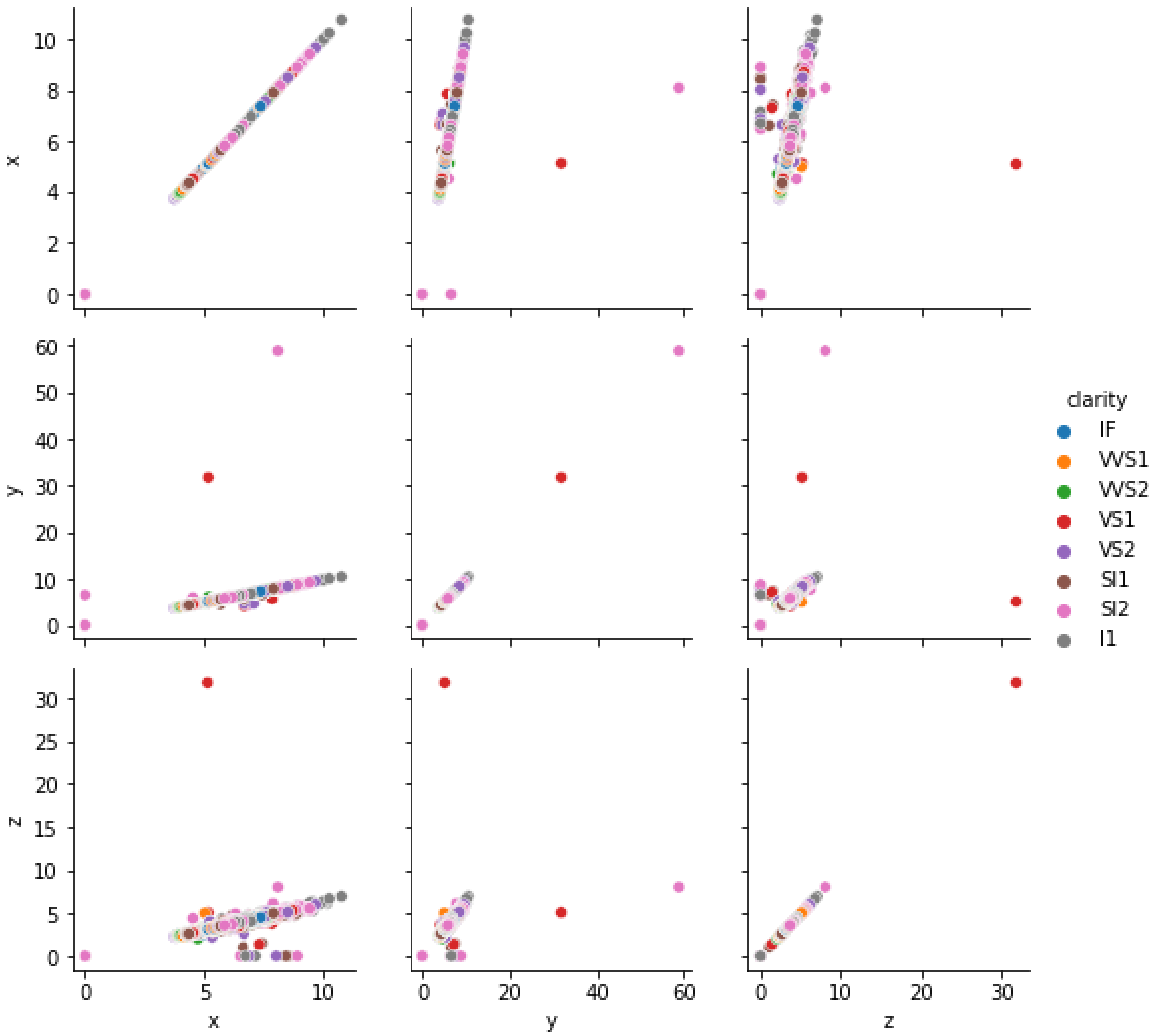


In [23]:

```
# Pairplotting by defining hue as clarity, x, y, z of the diamonds

"""
https://seaborn.pydata.org/generated/seaborn.PairGrid.html
```

```
class seaborn.PairGrid(data, *, hue=None, vars=None, x_vars=None, y_vars=None, hue_orde
                       corner=False, diag_sharey=True, height=2.5, aspect=1, layout_pad
"""

pairplotting_hue = sns.PairGrid(df_diamonds[["x", "y", "z", "clarity"]], hue="clarity")

pairplotting_hue.map(sns.scatterplot)

pairplotting_hue.add_legend()
```

Out[23]: <seaborn.axisgrid.PairGrid at 0x24c126f3a30>

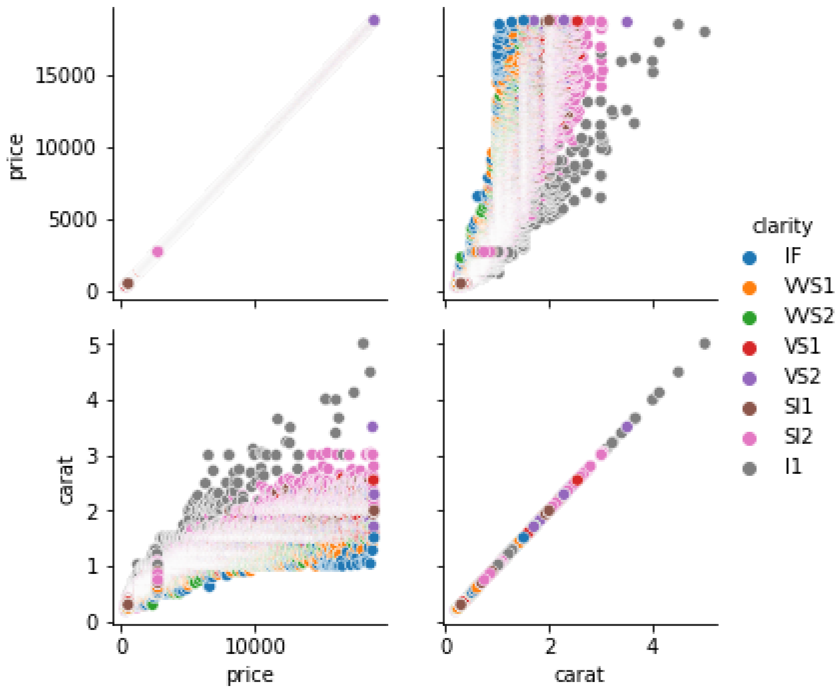


In [24]:

```
# Pairplotting by defining hue as clarity, price, carat of the diamonds

"""
https://seaborn.pydata.org/generated/seaborn.PairGrid.html

class seaborn.PairGrid(data, *, hue=None, vars=None, x_vars=None, y_vars=None, hue_orde
                       corner=False, diag_sharey=True, height=2.5, aspect=1, layout_pad
"""

pairplotting_hue = sns.PairGrid(df_diamonds[["price", "carat", "clarity"]], hue="clarit

pairplotting_hue.map(sns.scatterplot)
```

```
pairplotting_hue.add_legend()
```

Out[24]: <seaborn.axisgrid.PairGrid at 0x24c13dc9eb0>



In [ ]:

```

```

Hypotheses

Hypothesis - The price of diamonds increases with an increase in carat weight.

Data

In [25]:

```
correlation = df_diamonds['carat'].corr(df_diamonds['price'])
correlation
```

Out[25]: 0.9215913011934779

Conclusion

In [26]:

```
if correlation > 0:
    print("Hypothesis : Accepted")
else:
    print("Hypothesis : Rejected")
```

Hypothesis : Accepted

Hypothesis - The cut of a diamond significantly affects its price.

Data

In [27]:
```
cut_price_grouped = df_diamonds.groupby('cut')['price'].mean()
cut_price_grouped
```

Out[27]:
```
cut
Ideal        3457.541970
Premium      4584.257704
Very Good    3981.759891
Good         3928.864452
Fair         4358.757764
Name: price, dtype: float64
```

Conclusion

In [28]:
```
if cut_price_grouped.nunique() > 1:
    print("Hypothesis : Accepted")
else:
    print("Hypothesis : Rejected")
```

```
Hypothesis : Accepted
```

Hypothesis - The clarity of a diamond is related to its price.

Data

In [29]:
```
clarity_price_grouped = df_diamonds.groupby('clarity')['price'].mean()
clarity_price_grouped
```

Out[29]:
```
clarity
IF      2864.839106
VVS1    2523.114637
VVS2    3283.737071
VS1     3839.455391
VS2     3924.989395
SI1     3996.001148
SI2     5063.028606
I1      3924.168691
Name: price, dtype: float64
```

Conclusion

In [30]:
```
if clarity_price_grouped.nunique() > 1:
    print("Hypothesis : Accepted")
else:
    print("Hypothesis : Rejected")
```

```
Hypothesis : Accepted
```

Hypothesis - The color grade of a diamond has an impact on its price.

Data

In [31]:
```
color_price_grouped = df_diamonds.groupby('color')['price'].mean()
color_price_grouped
```

Out[31]:
```
color
D    3169.954096
E    3076.752475
F    3724.886397
G    3999.135671
H    4486.669196
I    5091.874954
J    5323.818020
Name: price, dtype: float64
```

Conclusion

In [32]:
```
if color_price_grouped.nunique() > 1:
    print("Hypothesis : Accepted")
else:
    print("Hypothesis : Rejected")
```

```
Hypothesis : Accepted
```

Hypothesis - There is a relationship between diamond dimensions and carat weight.

Data

In [33]:
```
correlation_x = df_diamonds['carat'].corr(df_diamonds['x'])
correlation_y = df_diamonds['carat'].corr(df_diamonds['y'])
correlation_z = df_diamonds['carat'].corr(df_diamonds['z'])

print("correlation_x with carat = ", correlation_x)
print("correlation_y with carat = ", correlation_y)
print("correlation_z with carat = ", correlation_z)
```

```
correlation_x with carat =  0.9750942267264208
correlation_y with carat =  0.9517221990129818
correlation_z with carat =  0.9533873805614194
```

Conclusion

In [34]:
```
if correlation_x > 0 or correlation_y > 0 or correlation_z > 0:
    print("Hypothesis : Accepted")
else:
    print("Hypothesis : Rejected")
```

```
Hypothesis : Accepted
```

Hypothesis - The distribution of diamond prices varies across different cut categories.

Data

In [35]:

```
cut_price_data = [df_diamonds[df_diamonds['cut'] == cut]['price'] for cut in df_diamond
cut_price_data
```

Out[35]:

```
[0         326
 11        340
 13        344
 16        348
 39        403
          ...
 53925    2756
 53926    2756
 53929    2756
 53935    2757
 53939    2757
 Name: price, Length: 21551, dtype: int64,
 1         326
 3         334
 12        342
 14        345
 15        345
          ...
 53928    2756
 53930    2756
 53931    2756
 53934    2757
 53938    2757
 Name: price, Length: 13791, dtype: int64,
 2         327
 4         335
 10        339
 17        351
 18        351
          ...
 53913    2753
 53914    2753
 53916    2753
 53927    2756
 53936    2757
 Name: price, Length: 4906, dtype: int64,
 5         336
 6         336
 7         337
 9         338
 19        351
          ...
 53921    2755
 53922    2755
 53932    2757
 53933    2757
 53937    2757
 Name: price, Length: 12082, dtype: int64,
 8         337
 91       2757
 97       2759
 123      2762
 124      2762
```

```
         ...
53757    2724
53800    2732
53863    2743
53879    2745
53882    2747
Name: price, Length: 1610, dtype: int64]
```

Conclusion

In [36]:
```
if pd.DataFrame(cut_price_data).var().sum() > 0:
    print("Hypothesis : Accepted")
else:
    print("Hypothesis : Rejected")
```

```
Hypothesis : Rejected
```

Hypothesis - The price per carat differs for different clarity categories.

Data

In [37]:
```
df_diamonds['price_per_carat'] = df_diamonds['price'] / df_diamonds['carat']
clarity_price_per_carat = df_diamonds.groupby('clarity')['price_per_carat'].mean()
clarity_price_per_carat
```

Out[37]:
```
clarity
IF      4259.931736
VVS1    3851.410558
VVS2    4204.166013
VS1     4155.816808
VS2     4080.526787
SI1     3849.078018
SI2     4010.853865
I1      2796.296437
Name: price_per_carat, dtype: float64
```

Conclusion

In [38]:
```
if clarity_price_per_carat.nunique() > 1:
    print("Hypothesis : Accepted")
else:
    print("Hypothesis : Rejected")
```

```
Hypothesis : Accepted
```

Hypothesis - The average price of diamonds has changed over time.

Data & Conclusion

In [39]:
```
if 'date' in df_diamonds.columns:
    df_diamonds['date'] = pd.to_datetime(df_diamonds['date'])
```

```
    time_price_trend = df_diamonds.groupby('date')['price'].mean().pct_change().mean()
    if time_price_trend != 0:
        print("Hypothesis : Accepted")
    else:
        print("Hypothesis : Rejected")
else:
    print("Hypothesis : Dataset does not contain temporal aspect.")
```

```
Hypothesis : Dataset does not contain temporal aspect.
```

In [ ]:

```

```