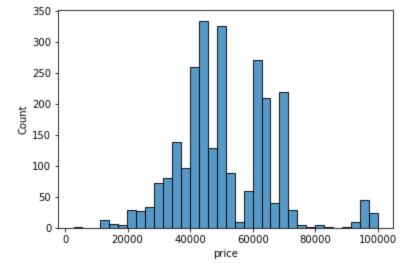
```
In [20]:
          import numpy as np
          import pandas as pd
          import seaborn as sns
          from sklearn.tree import DecisionTreeRegressor
          from sklearn.model selection import train test split
          from sklearn.metrics import r2 score
          import matplotlib.pyplot as plt
          #df = pd.read csv("C:\Code\D214\move.csv")
          df = pd.read_csv("C:\\Code\\D214\\rent_apartment_moscow.csv")
In [2]:
          #viewing size of dataset
          print('Length of dataset: ', len(df.index))
         print('Number of columns: ', len(df.columns))
         Length of dataset: 4241
         Number of columns:
In [3]:
          #viewing price distribution
          sns.histplot(df['price'])
         <AxesSubplot:xlabel='price', ylabel='Count'>
Out[3]:
           1200
           1000
            800
            600
            400
            200
                                      3
                                             4
                                                    5
                                                           6
                                                          le6
                                    price
In [4]:
          #remove price outliers
          df = df[df['price'] < 100000]</pre>
In [5]:
          #viewing price distribution again
          sns.histplot(df['price'])
         <AxesSubplot:xlabel='price', ylabel='Count'>
Out[5]:
```



In [8]:

In [9]:

x = df['total_area']
y = df['price']

x = np.array(x)

```
In [6]:
            #creating variable for sale price
            price = df['price']
            price.describe()
                        2569.000000
          count
Out[6]:
          mean
                       50987.761386
                       15154.275901
          std
          min
                        2700.000000
          25%
                       40000.000000
          50%
                       50000.000000
          75%
                       60000.000000
                       99999.000000
          max
          Name: price, dtype: float64
In [7]:
            #viewing correlation
            cor = df.corr()
            sns.heatmap(cor, annot = True)
           <AxesSubplot:>
Out[7]:
                                                                            -1.0
           Unnamed: 0 - 1 0.11 0.046-0.05-0.15 0.0240.0250.0530.0710.071
                                 0.083-0.16-0.0480.0340.026 0.3 0.26 0.071
                                                                            - 0.8
               minutes -0.0460.083 1 0.0140.0190.00680.01-0.0270.0190.036
                                                                            - 0.6
            fee percent -0.05-0.160.014 1 0.006060056.00960.17-0.28-0.28
                 views -0.15-0.0480.0190.006 1 0.0079.0058002-20.0390.046
                                                                            - 0.4
                 storey -0.0240.0340.006280056.007
                                                1 0.91 0.0220.001-0.009
                                                                            - 0.2
                storeys -0.0250.026-0.010.00906.005 0.91 1
              total area -0.053 0.3 -0.027-0.170.00210.0220.023
                                                                             0.0
             living_area -0.071 0.26 0.019-0.28 0.039.00130.012 0.75
           kitchen_area -0.0710.0710.036-0.280.04@.009300290.56 0.89
                         Unnamed: 0
                                            views
                                                     storeys
                                                               living_area
                                                                    dtchen_area
                                                          total_area
                                  minutes
                                       ee percent
```

```
y = np.array(y)
In [11]:
          x = x.reshape(-1,1)
          y = y.reshape(-1,1)
In [10]:
          print(x)
         [50 38 55 ... 50 64 64]
In [12]:
          x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.2)
In [13]:
          model = DecisionTreeRegressor()
          model = model.fit(x train,y train)
In [14]:
          predict = model.predict(x test)
In [15]:
          r2_score(y_test, predict)
         0.1439700610069341
Out[15]:
In [27]:
          sns.residplot(x = y_test, y = predict)
         <AxesSubplot:>
Out[27]:
          40000
          30000
          20000
          10000
         -10000
                       20000
                                40000
                                                 80000
                                         60000
                                                          100000
```

In []: