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


1. Read the file

In [2]:
```
#1. بارگذاری داده‌ها
df = pd.read_excel('cleardata.xlsx')
```

In [3]:
```
df = df.rename(columns={'managhe_shahrdari': 'region'})
```

2.Exploring the data

In [4]:
```
df.describe()
```

Out[4]:

	region	masahat	price	age
count	209989.000000	3.318600e+05	3.318590e+05	331860.000000
mean	7.743810	2.204266e+03	3.435901e+04	6.901371
std	5.441857	9.347086e+05	2.255549e+05	8.301445
min	1.000000	1.000000e+00	0.000000e+00	0.000000
25%	4.000000	6.681000e+01	1.362840e+04	1.000000
50%	6.000000	8.450000e+01	2.320000e+04	4.000000
75%	11.000000	1.078000e+02	3.838384e+04	10.000000
max	22.000000	5.300000e+08	7.100000e+07	1309.000000

In [5]:
```
df.head()
```

Out[5]:

	region	masahat	price	age	eskelet	date	Ostan	Shahrestan
0	NaN	60.00	5000.00	10	felezi	1395/01/01	Tehran	Pakdasht
1	14.0	70.63	35962.06	1	botoni	1395/01/01	Tehran	Tehran
2	NaN	75.00	8000.00	1	botoni	1395/01/01	Kurdistan	turpentine
3	2.0	196.16	173327.90	20	felezi	1395/01/01	Tehran	Tehran
4	NaN	118.00	11101.69	0	botoni	1395/01/01	Alborz	Karaj

In [6]:
```
# محاسبه کوانتایل‌های ۲۰ و ۸۰ درصد
q_20 = df['price'].quantile(0.2)
q_80 = df['price'].quantile(0.8)

# فیلترینگ بر اساس کوانتایل‌ها
df = df[(df['price'] >= q_20) & (df['price'] <= q_80)]
```

In [7]:
```
# price / 10,000
df['price'] = df['price']%10000
df.head(2)
```

Out[7]:

	region	masahat	price	age	eskelet	date	Ostan	Shahrestan
1	14.0	70.63	5962.06	1	botoni	1395/01/01	Tehran	Tehran
5	NaN	145.00	5172.41	5	botoni	1395/01/01	Alborz	Karaj

Double-click to zoom back out

In [8]:
```
# قیمت برحسب میلیون تومان
df['price'] = np.log(df['price'])
df.head(5)
```

```
/home/anjel/.local/lib/python3.11/site-packages/pandas/core/arraylike.py:399: RuntimeWarning: divide by zero encountered in log
  result = getattr(ufunc, method)(*inputs, **kwargs)
```

Out[8]:

	region	masahat	price	age	eskelet	date	Ostan	Shahrestan
1	14.0	70.63	8.693171	1	botoni	1395/01/01	Tehran	Tehran
5	NaN	145.00	8.551094	5	botoni	1395/01/01	Alborz	Karaj
6	1.0	87.00	8.407996	1	botoni	1395/01/01	Tehran	Tehran
7	NaN	107.00	7.673004	5	botoni	1395/01/01	Alborz	Karaj
10	NaN	58.00	7.788953	0	botoni	1395/01/02	Alborz	Karaj

In [ ]:
```
```

In [9]:
```
df['price'].isin([-np.inf]).sum()
```

Out[9]: 3292

In [10]:
```
df = df[~df['price'].isin([-np.inf])]
```

In [11]:
```
#check -inf value is clear
df['price'].isin([-np.inf]).sum()
```

Out[11]: 0

In [12]:
```
#بررسی عدد ۰ و منفی در ستون ها
count_zero_or_negative1 = (df['masahat'] <= 0).sum()
print(count_zero_or_negative1, 'عدد منفی در مساحت')

count_zero_or_negative2 = (df['price'] <= 0).sum()
print(count_zero_or_negative2, ' : عدد منفی در یک متر مربع')

count_zero_or_negative3 = (df['age'] < 0).sum()
print(count_zero_or_negative3, ' : عدد منفی در سن بنا')
```

```
0 عدد منفی در مساحت
2  : عدد منفی در یک متر مربع
0  : عدد منفی در سن بنا
```

In [13]:
```
# value <0 is drop
# حذف سطرهایی که مقدار یکی از ستون‌ها (masahat، gheymat_1_metr_moraba، age
df = df[(df['masahat'] > 0) & (df['price'] > 0) &
```

Double-click to zoom back out

In [14]:
```
#بررسی عدد ۰ و منفی در ستون ها
count_zero_or_negative1 = (df['masahat'] <= 0).sum()
print(count_zero_or_negative1, 'عدد منفی در  مساحت')

count_zero_or_negative2 = (df['price'] <= 0).sum()
print(count_zero_or_negative2, ' قیمت :')

count_zero_or_negative3 = (df['age'] < 0).sum()
print(count_zero_or_negative3, ' : عدد منفی در سن بنا')
```

```
0 عدد منفی در  مساحت
0  قیمت :
0  : عدد منفی در سن بنا
```

In [15]:
```
df.isnull().sum()
```

Out[15]:
```
region        57320
masahat           0
price             0
age               0
eskelet           0
date              0
Ostan             0
Shahrestan        0
dtype: int64
```

In [16]:
```
df.fillna(0, inplace=True)
df.isnull().sum()
```

Out[16]:
```
region        0
masahat       0
price         0
age           0
eskelet       0
date          0
Ostan         0
Shahrestan    0
dtype: int64
```

In [17]:
```
df.count()
```

Out[17]:
```
region        168676
masahat       168676
price         168676
age           168676
eskelet       168676
date          168676
Ostan         168676
Shahrestan    168676
dtype: int64
```

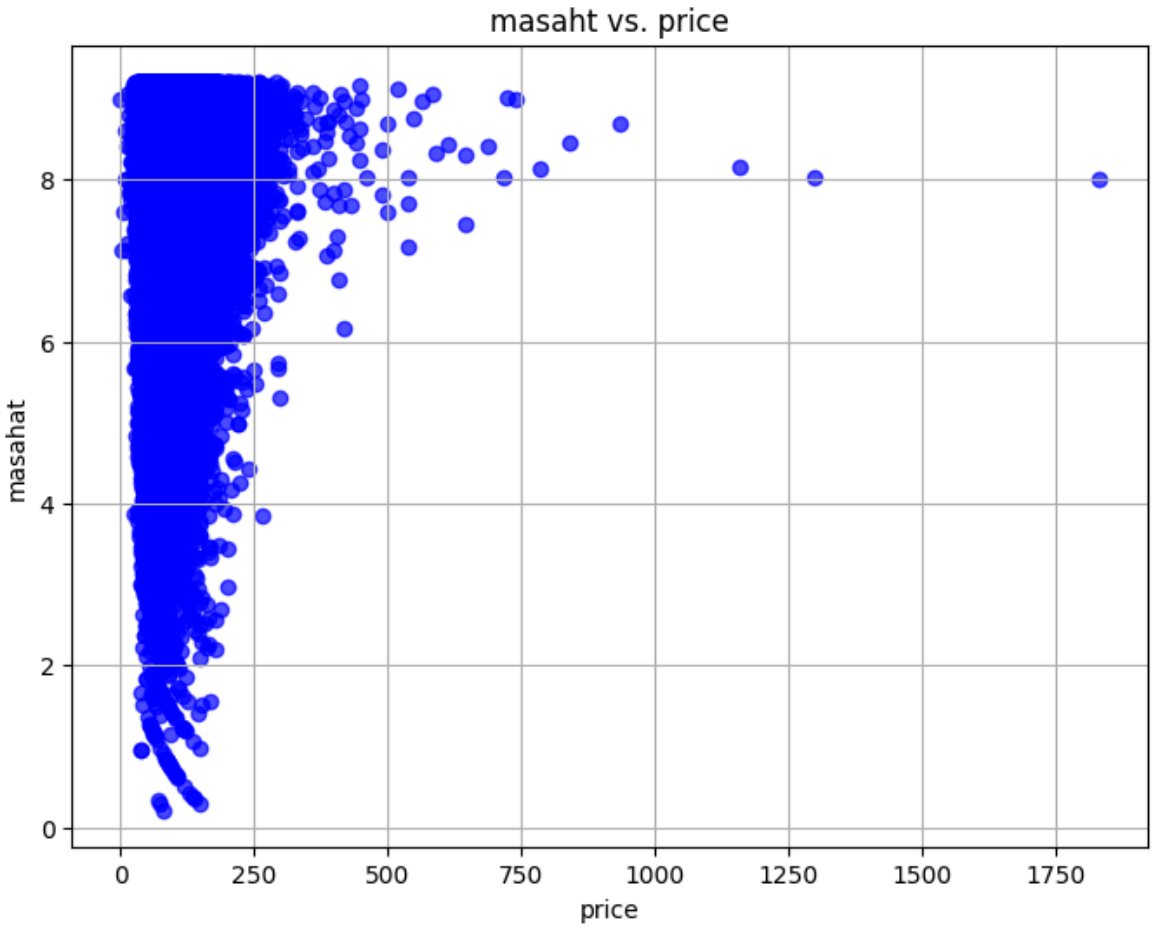
In [18]:
```
df.head(2)
```

Out[18]:

	region	masahat	price	age	eskelet	date	Ostan	Shahrestan
1	14.0	70.63	8.693171	1	botoni	1395/01/01	Tehran	Tehran
5	0.0	145.00	8.551094	5	botoni	1395/01/01	A	

Double-click to zoom back out

In [19]:
```
# رسم نمودار scatter برای دو ویژگی SepalLength و SepalWidth
plt.figure(figsize=(8, 6))
plt.scatter(df['masahat'], df['price'], color='blue', alpha=0.7)
plt.title('masaht vs. price')
plt.xlabel('price')
plt.ylabel('masahat')
plt.grid(True)
plt.show()
```



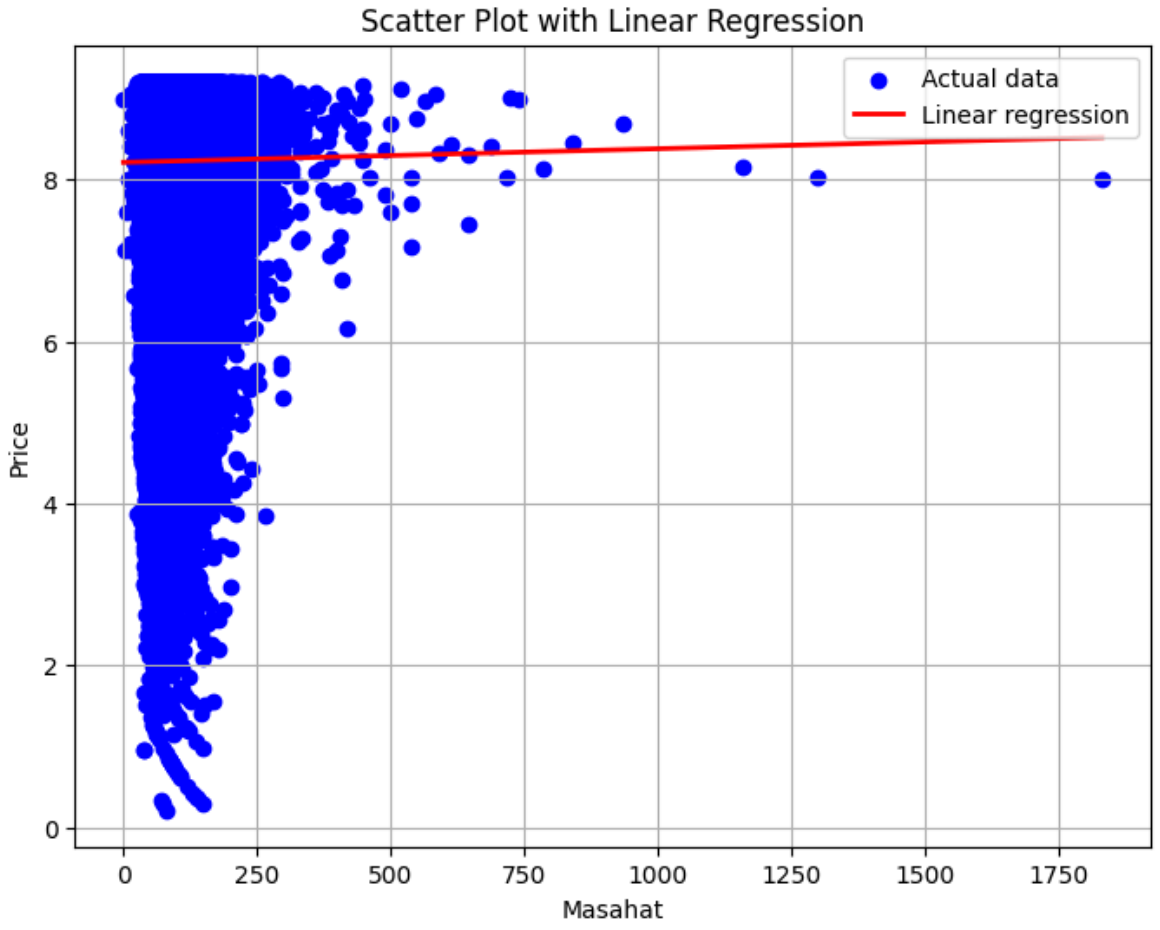
در نمودار بالا شاهد پراکندگی هستیم

In [20]:
```
from sklearn.linear_model import LinearRegression

# ایجاد مدل رگرسیون خطی
model = LinearRegression()
model.fit(df[['masahat']], df['price'])

# پیش‌بینی قیمت بر اساس مساحت ساختمان
masahat_range = np.linspace(min(df['masahat']), max(df['masahat']), 100).
price_pred = model.predict(masahat_range)

# رسم نمودار scatter برای داده‌های واقعی و خط پیش‌بینی
plt.figure(figsize=(8, 6))
plt.scatter(df['masahat'], df['price'], color='blue', label='Actual data'
```

```
plt.plot(masahat_range, price_pred, color='red', linewidth=2, label='Line
plt.title('Scatter Plot with Linear Regression')
plt.xlabel('Masahat')
plt.ylabel('Price')
plt.legend()
plt.grid(True)
plt.show()
```

Double-click to zoom back out

```
/home/anjel/.local/lib/python3.11/site-packages/sklearn/base.py:493: UserW
arning: X does not have valid feature names, but LinearRegression was fitt
ed with feature names
  warnings.warn(
```

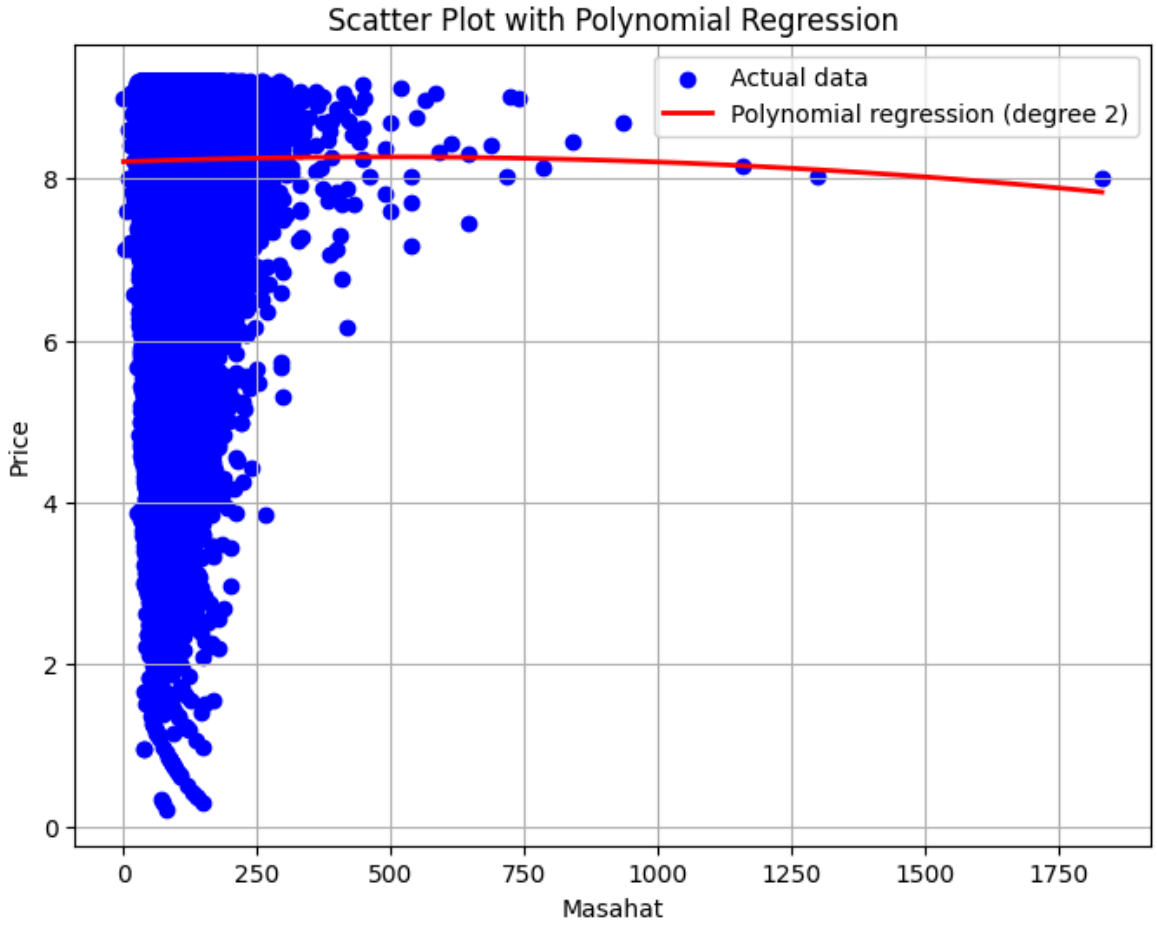


In [21]:
```
# استفاده از رگرسیون چندجمله‌ای درجه دوم
from sklearn.preprocessing import PolynomialFeatures
from sklearn.linear_model import LinearRegression
from sklearn.pipeline import make_pipeline

degree = 2
model = make_pipeline(PolynomialFeatures(degree), LinearRegression())
model.fit(df[['masahat']], df['price'])

# پیش‌بینی قیمت بر اساس مساحت ساختمان
masahat_range = np.linspace(min(df['masahat']), max(df['masahat']), 100).
price_pred = model.predict(masahat_range)

# رسم نمودار scatter برای داده‌های واقعی و خط پیش‌بینی
plt.figure(figsize=(8, 6))
plt.scatter(df['masahat'], df['price'], color='blue', label='Actual data'
plt.plot(masahat_range, price_pred, color='red', linewidth=2, label=f'Pol
plt.title('Scatter Plot with Polynomial Regression')
plt.xlabel('Masahat')
```

```
plt.ylabel('Price')
plt.legend()
plt.grid(True)
plt.show()
```

Double-click to zoom back out

```
/home/anjel/.local/lib/python3.11/site-packages/sklearn/base.py:493: UserW
arning: X does not have valid feature names, but PolynomialFeatures was fi
tted with feature names
  warnings.warn(
```



Here's the translated version of the text you provided:

Sample DataFrame Creation:

First, create a simple DataFrame with two columns, `masahat` (area) and `price`:

Polynomial Regression:

Using `make_pipeline` from Scikit-learn, create a linear pipeline using `PolynomialFeatures` and `LinearRegression` that automatically generates polynomial features from the input and applies a linear regression model to it.

Price Prediction:

Using the trained model, predict prices for different areas and store them in `price_pred`.

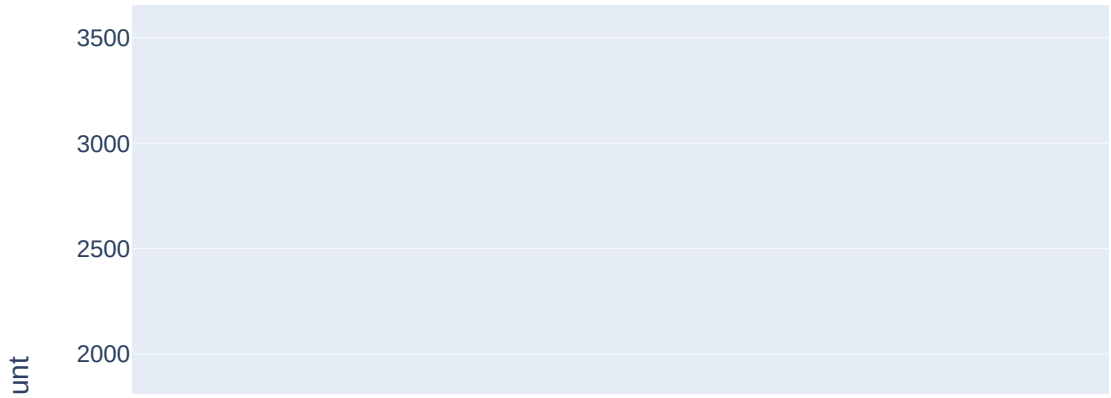
Plotting:

Plot the actual data points and the prediction line. Display the scatter plot using matplotlib.

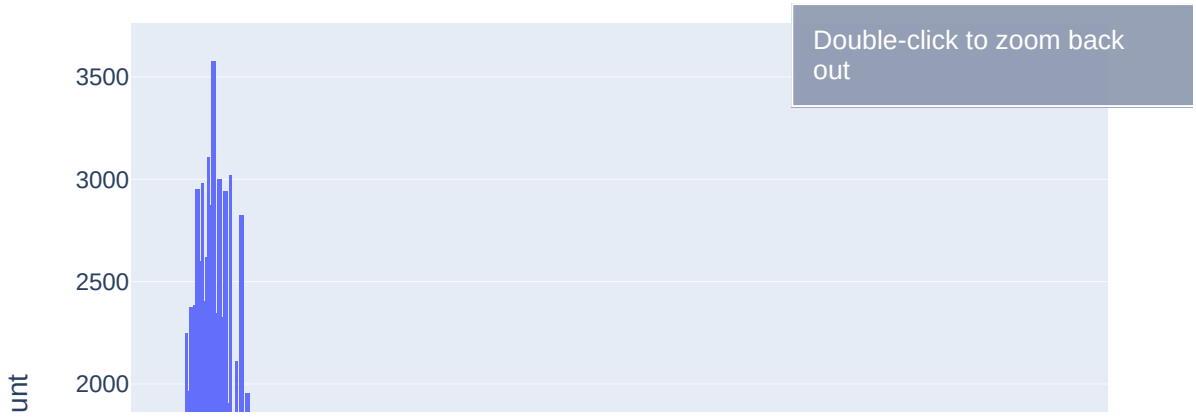
This translation captures the essence of each section: creating a DataFrame, performing polynomial regression with `make_pipeline`, predicting prices based on the model, and visualizing the results using a scatter plot.

df.shape

In [23]:
```
px.histogram(df, x='price')
```

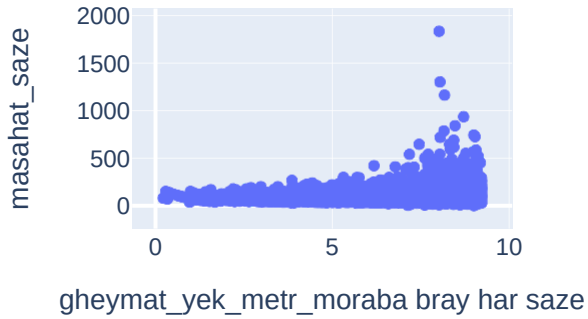


In [24]:
```
px.histogram(df, x='masahat')
```



In [25]:
```
px.histogram(df, x='age')
```

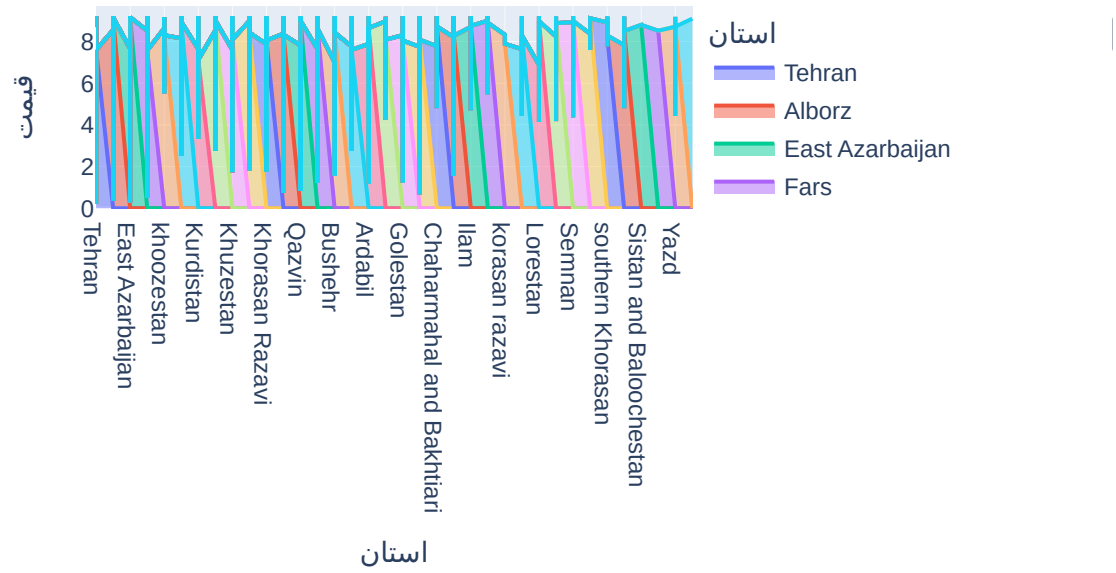
In [26]:
```
px.scatter(df, x='price', y='masahat',
           labels={"price": "gheymat_yek_metr_moraba bray har saze",
                   "masahat": "masahat_saze"},
           width=350, height=250)
```



شاهد پراکندگی زیادی هستیم با توجه به حجم بالای داده ها

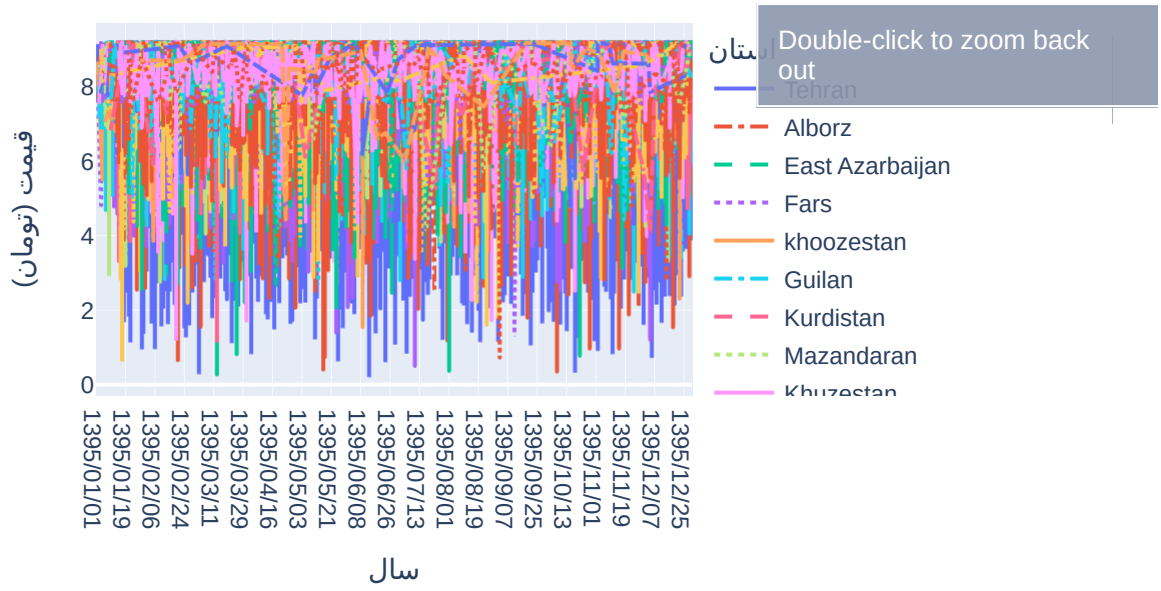
In [27]:
```
labels = {
    'Ostan': 'استان',
    'price': 'قیمت'
}
```

```
# رسم نمودار area با استفاده از plotly.express
fig = px.area(df, x='Ostan', y='price', color='Ostan', line_group='Ostan'

# نمایش نمودار
fig.show()
```

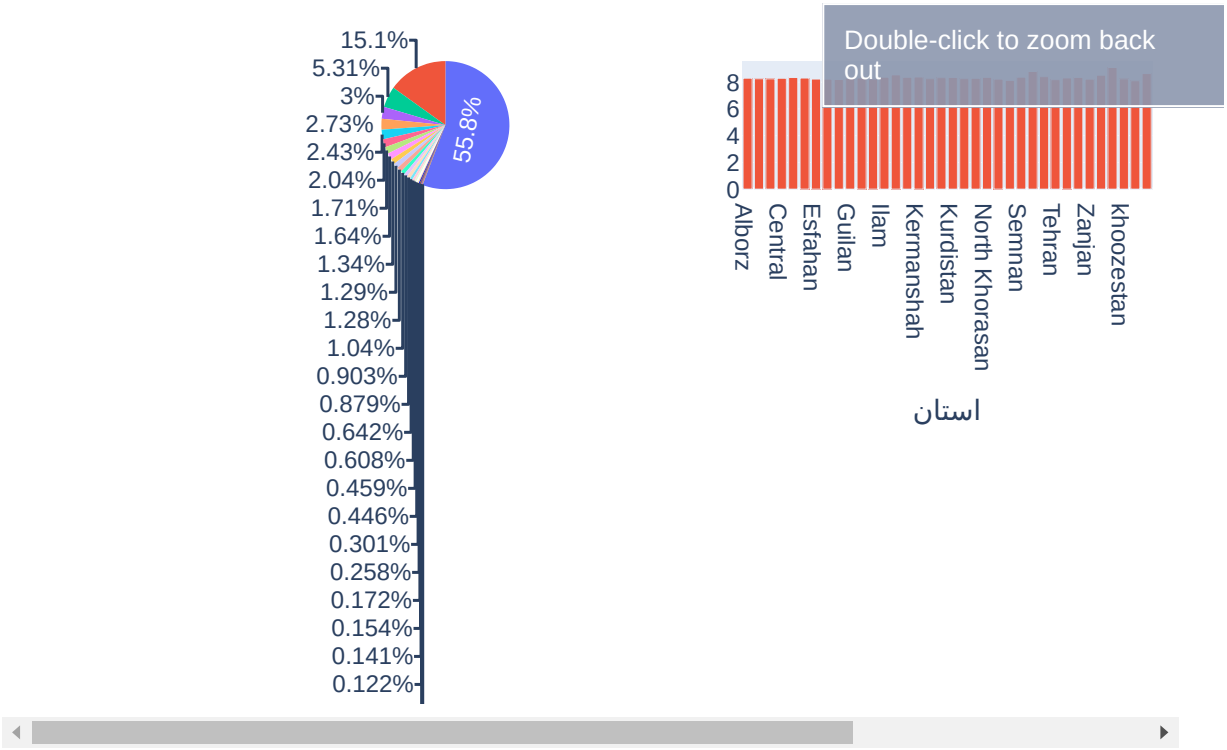


In [28]:

```
from itertools import cycle

labels = {
    'Ostan': 'استان',
    'price': 'قیمت',
    'Year': 'سال'
}

# رسم نمودار خطی با استفاده از plotly.express
fig = px.line(df, x='date', y='price', color='Ostan', width=600, height=3

# تنظیم استایل خطهای نمودار
styles = cycle([None, 'dashdot', 'dash', 'dot'])
for ostan in df['Ostan'].unique():
    fig.update_traces(selector=dict(name=ostan), line=dict(dash=next(styl

fig.update_yaxes(title_text='قیمت (تومان)')
fig.update_xaxes(title_text='سال')

# نمایش نمودار
fig.show()
```

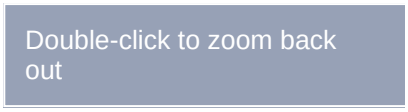


In [29]:
```
import plotly.graph_objects as go
from plotly.subplots import make_subplots
# محاسبه تعداد تکرار هر استان
ostan_counts = df['Ostan'].value_counts()
ostan_labels = ostan_counts.index
ostan_values = ostan_counts.values

# محاسبه میانگین قیمت‌ها برای هر استان
ostan_avg_price = df.groupby('Ostan')['price'].mean()

# ایجاد نمودارهای ترکیبی
fig = make_subplots(rows=1, cols=2, specs=[[{"type": "pie"}, {"type": "xy

# اضافه کردن نمودار Pie
fig.add_trace(go.Pie(values=ostan_values, labels=ostan_labels), row=1, co

# اضافه کردن نمودار Bar
fig.add_trace(go.Bar(x=ostan_avg_price.index, y=ostan_avg_price.values),

# تنظیم محور X برای نمودار Bar
fig.update_xaxes(title_text="استان", row=1, col=2)

# تنظیمات نهایی و نمایش نمودار
fig.update_layout(width=800, height=400, showlegend=True)
fig.show()
```

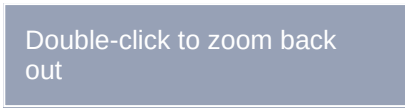


In [30]:
```
data1 = {
    'Shahr': ['Tehran', 'Mashhad', 'Isfahan', 'Tabriz', 'Shiraz', 'Karaj'
    'price': [7.53, 8.6, 9.3, 7.77, 8.89, 7.730, 9.63, 9.75, 8.24, 8.33],
    'Latitude': [35.6892, 36.2605, 32.6539, 38.0800, 29.5926, 35.8400, 34
    'Longitude': [51.3890, 59.6168, 51.6660, 46.2919, 52.5836, 50.9916, 5
}
data1 = pd.DataFrame(data1)

# تعریف برچسب‌ها برای نمودار
labels = {
    'Shahr': 'شهر',
    'price': 'قیمت'
}

# رسم نمودار ژئوگرافیک با استفاده از plotly.express
fig = px.scatter_geo(data1, lat='Latitude', lon='Longitude', hover_name='
                     title='نقشه جغرافیایی قیمت‌های شهرهای ایران', labels=
                     projection='mercator', width=800, height=600)

# تنظیمات نهایی و نمایش نمودار
fig.update_geos(showland=True, landcolor="lightgray", showcountries=True,
fig.show()
```

نقشه جغرافیایی قیمت‌های شهرهای ایران



In [31]:
```
# تعریف برچسب‌ها برای نمودار
df_labels = {
    'price': 'قیمت فروش (تومان)',
    'date': 'تاریخ',
    'masahat': 'مساحت (متر مربع)'
}

# رسم نمودار خطی با استفاده از plotly.express
fig = px.line(df, x='date', y='price', title='قیمت فروش میانه', labels=df

# افزودن حاشیه‌ها به نمودار
fig.update_layout(margin=dict(t=30))

# نمایش نمودار
fig.show()
```


Double-click to zoom back out