Laptop Data Analysis

In [1]:

```
import pandas as pd
```

In [2]:

```
data = pd.read_csv('E:/file2/Downloads/laptop_details.csv')
data.head()
```

Out[2]:

	Product	Rating	MRP	Feature
0	Lenovo IdeaPad 3 Core i3 11th Gen - (8 GB/256 ...	4.2	₹36,990	Intel Core i3 Processor (11th Gen)8 GB DDR4 RA...
1	Lenovo IdeaPad 3 Core i3 11th Gen - (8 GB/512 ...	4.2	₹39,990	Intel Core i3 Processor (11th Gen)8 GB DDR4 RA...
2	ASUS VivoBook 15 (2022) Core i3 10th Gen - (8 ...	4.3	₹32,990	Intel Core i3 Processor (10th Gen)8 GB DDR4 RA...
3	HP Pavilion Ryzen 5 Hexa Core AMD R5-5600H - (...	4.4	₹49,990	AMD Ryzen 5 Hexa Core Processor8 GB DDR4 RAM64...
4	ASUS TUF Gaming F15 Core i5 10th Gen - (8 GB/5...	4.4	₹49,990	Intel Core i5 Processor (10th Gen)8 GB DDR4 RA...

In [3]:

```
import re
regex = r'\b(?:AMD|Intel|M1|M2|Qualcomm)[\s\w]+Processor'
fun = lambda x: re.findall(regex, x)
data['processor'] = data['Feature'].apply(fun)
```

In [4]:

```
data['processor']
```

Out[4]:

```
0                 [Intel Core i3 Processor]
1                 [Intel Core i3 Processor]
2                 [Intel Core i3 Processor]
3         [AMD Ryzen 5 Hexa Core Processor]
4                 [Intel Core i5 Processor]
                        ...
715     [Intel Celeron Dual Core Processor]
716       [AMD Ryzen 7 Octa Core Processor]
717       [AMD Ryzen 9 Octa Core Processor]
718               [Intel Core i7 Processor]
719               [Intel Core i5 Processor]
Name: processor, Length: 720, dtype: object
```

In [5]:

```
regex = r'\d+\sGB[\s\w]+RAM'
fun = lambda x : re.findall(regex, x)
data['RAM'] = data['Feature'].apply(fun)
```

In [6]:

```
data['RAM']
```

Out[6]:

```
0          [8 GB DDR4 RAM]
1          [8 GB DDR4 RAM]
2          [8 GB DDR4 RAM]
3          [8 GB DDR4 RAM]
4          [8 GB DDR4 RAM]
                ...
715      [4 GB LPDDR4 RAM]
716        [8 GB DDR4 RAM]
717       [16 GB DDR5 RAM]
718     [16 GB LPDDR5 RAM]
719        [8 GB DDR4 RAM]
Name: RAM, Length: 720, dtype: object
```

In [7]:

```
regex = r'(?:Windows|Chrome|Mac|Dos)[\s\w]+Operating System'
fun = lambda x : re.findall(regex, x, re.IGNORECASE)
data["os"] = data.Feature.apply(fun)
```

In [8]:

```
data.head()
```

Out[8]:

	Product	Rating	MRP	Feature	processor	RAM	os
0	Lenovo IdeaPad 3 Core i3 11th Gen - (8 GB/256 ...	4.2	₹36,990	Intel Core i3 Processor (11th Gen)8 GB DDR4 RA...	[Intel Core i3 Processor]	[8 GB DDR4 RAM]	[Windows 11 Operating System]
1	Lenovo IdeaPad 3 Core i3 11th Gen - (8 GB/512 ...	4.2	₹39,990	Intel Core i3 Processor (11th Gen)8 GB DDR4 RA...	[Intel Core i3 Processor]	[8 GB DDR4 RAM]	[Windows 11 Operating System]
2	ASUS VivoBook 15 (2022) Core i3 10th Gen - (8 ...	4.3	₹32,990	Intel Core i3 Processor (10th Gen)8 GB DDR4 RA...	[Intel Core i3 Processor]	[8 GB DDR4 RAM]	[Windows 11 Operating System]
3	HP Pavilion Ryzen 5 Hexa Core AMD R5-5600H - (...	4.4	₹49,990	AMD Ryzen 5 Hexa Core Processor8 GB DDR4 RAM64...	[AMD Ryzen 5 Hexa Core Processor]	[8 GB DDR4 RAM]	[Windows 10 Operating System]
4	ASUS TUF Gaming F15 Core i5 10th Gen - (8 GB/5...	4.4	₹49,990	Intel Core i5 Processor (10th Gen)8 GB DDR4 RA...	[Intel Core i5 Processor]	[8 GB DDR4 RAM]	[Windows 11 Operating System]

In [12]:

```
#product = r'(?:Lenovo|ASUS|HP|DELL|RedmiBook|realme Book|acer|MSI|APPLE|Infinix)'
#pro = lambda x : re.findall(product,x, re.IGNORECASE)
regex = r'^\w+'

new_data["brand"] = data.Product.apply(lambda x: re.findall(regex, x))
#data['product']=data.Product.apply(pro)
```

In [13]:

```
new_data.head()
```

Out[13]:

	Product	os	RAM	processor	Rating	MRP	brand
0	Lenovo IdeaPad 3 Core i3 11th Gen - (8 GB/256 ...	[Windows 11 Operating System]	[8 GB DDR4 RAM]	[Intel Core i3 Processor]	4.2	₹36,990	[Lenovo]
1	Lenovo IdeaPad 3 Core i3 11th Gen - (8 GB/512 ...	[Windows 11 Operating System]	[8 GB DDR4 RAM]	[Intel Core i3 Processor]	4.2	₹39,990	[Lenovo]
2	ASUS VivoBook 15 (2022) Core i3 10th Gen - (8 ...	[Windows 11 Operating System]	[8 GB DDR4 RAM]	[Intel Core i3 Processor]	4.3	₹32,990	[ASUS]
3	HP Pavilion Ryzen 5 Hexa Core AMD R5-5600H - (...	[Windows 10 Operating System]	[8 GB DDR4 RAM]	[AMD Ryzen 5 Hexa Core Processor]	4.4	₹49,990	[HP]
4	ASUS TUF Gaming F15 Core i5 10th Gen - (8 GB/5...	[Windows 11 Operating System]	[8 GB DDR4 RAM]	[Intel Core i5 Processor]	4.4	₹49,990	[ASUS]

In [11]:

```
new_data =data[['Product','os','RAM','processor','Rating','MRP']]
new_data.head()
```

Out[11]:

	Product	os	RAM	processor	Rating	MRP
0	Lenovo IdeaPad 3 Core i3 11th Gen - (8 GB/256 ...	[Windows 11 Operating System]	[8 GB DDR4 RAM]	[Intel Core i3 Processor]	4.2	₹36,990
1	Lenovo IdeaPad 3 Core i3 11th Gen - (8 GB/512 ...	[Windows 11 Operating System]	[8 GB DDR4 RAM]	[Intel Core i3 Processor]	4.2	₹39,990
2	ASUS VivoBook 15 (2022) Core i3 10th Gen - (8 ...	[Windows 11 Operating System]	[8 GB DDR4 RAM]	[Intel Core i3 Processor]	4.3	₹32,990
3	HP Pavilion Ryzen 5 Hexa Core AMD R5-5600H - (...	[Windows 10 Operating System]	[8 GB DDR4 RAM]	[AMD Ryzen 5 Hexa Core Processor]	4.4	₹49,990
4	ASUS TUF Gaming F15 Core i5 10th Gen - (8 GB/5...	[Windows 11 Operating System]	[8 GB DDR4 RAM]	[Intel Core i5 Processor]	4.4	₹49,990

In [14]:

```
new_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 720 entries, 0 to 719
Data columns (total 7 columns):
 #   Column     Non-Null Count  Dtype  
---  ------     --------------  -----  
 0   Product    720 non-null    object 
 1   os         720 non-null    object 
 2   RAM        720 non-null    object 
 3   processor  720 non-null    object 
 4   Rating     581 non-null    float64
 5   MRP        720 non-null    object 
 6   brand      720 non-null    object 
dtypes: float64(1), object(6)
memory usage: 39.5+ KB
```

In [15]:

```
new_data['MRP'] = new_data['MRP'].str.replace('₹','')
new_data['MRP'] = new_data['MRP'].str.replace(',','')
```

In [16]:

```
new_data['MRP']=new_data['MRP'].astype(int)
new_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 720 entries, 0 to 719
Data columns (total 7 columns):
 #   Column     Non-Null Count  Dtype  
---  ------     --------------  -----  
 0   Product    720 non-null    object 
 1   os         720 non-null    object 
 2   RAM        720 non-null    object 
 3   processor  720 non-null    object 
 4   Rating     581 non-null    float64
 5   MRP        720 non-null    int32  
 6   brand      720 non-null    object 
dtypes: float64(1), int32(1), object(5)
memory usage: 36.7+ KB
```

In [17]:

```
regex = r'[\d]+\s(?:GB|TB)\s(?:HDD|SSD)'
new_data['Storage'] = data['Feature'].apply(lambda x : re.findall(regex, x))
```

In [18]:

```
new_data.processor = new_data.processor.apply(lambda x: ''.join(x))
new_data.RAM = new_data.RAM.apply(lambda x: ''.join(x))
new_data.os = new_data.os.apply(lambda x: ''.join(x))
new_data.Storage = new_data.Storage.apply(lambda x: ''.join(x))
new_data.brand = new_data.brand.apply(lambda x: ''.join(x))
```

In [19]:

```
new_data.head()
```

Out[19]:

	Product	os	RAM	processor	Rating	MRP	brand	Storage
0	Lenovo IdeaPad 3 Core i3 11th Gen - (8 GB/256 ...	Windows 11 Operating System	8 GB DDR4 RAM	Intel Core i3 Processor	4.2	36990	Lenovo	256 GB SSD
1	Lenovo IdeaPad 3 Core i3 11th Gen - (8 GB/512 ...	Windows 11 Operating System	8 GB DDR4 RAM	Intel Core i3 Processor	4.2	39990	Lenovo	512 GB SSD
2	ASUS VivoBook 15 (2022) Core i3 10th Gen - (8 ...	Windows 11 Operating System	8 GB DDR4 RAM	Intel Core i3 Processor	4.3	32990	ASUS	512 GB SSD
3	HP Pavilion Ryzen 5 Hexa Core AMD R5-5600H - (...	Windows 10 Operating System	8 GB DDR4 RAM	AMD Ryzen 5 Hexa Core Processor	4.4	49990	HP	512 GB SSD
4	ASUS TUF Gaming F15 Core i5 10th Gen - (8 GB/5...	Windows 11 Operating System	8 GB DDR4 RAM	Intel Core i5 Processor	4.4	49990	ASUS	512 GB SSD

In [20]:

```
df = new_data[['brand','os','RAM','processor','Rating','MRP']]
df.head()
```

Out[20]:

	brand	os	RAM	processor	Rating	MRP
0	Lenovo	Windows 11 Operating System	8 GB DDR4 RAM	Intel Core i3 Processor	4.2	36990
1	Lenovo	Windows 11 Operating System	8 GB DDR4 RAM	Intel Core i3 Processor	4.2	39990
2	ASUS	Windows 11 Operating System	8 GB DDR4 RAM	Intel Core i3 Processor	4.3	32990
3	HP	Windows 10 Operating System	8 GB DDR4 RAM	AMD Ryzen 5 Hexa Core Processor	4.4	49990
4	ASUS	Windows 11 Operating System	8 GB DDR4 RAM	Intel Core i5 Processor	4.4	49990

In [21]:

```
df['brand'].value_counts()
```

Out[21]:

```
ASUS         251
Lenovo       196
HP            61
DELL          61
acer          35
RedmiBook     31
MSI           30
Infinix       20
APPLE         19
realme         6
ALIENWARE      3
SAMSUNG        2
Ultimus        2
Vaio           1
GIGABYTE       1
Nokia          1
Name: brand, dtype: int64
```

In [22]:

```
df['MRP'].value_counts()
```

Out[22]:

```
39990     61
36990     39
85990     34
38705     30
196990    30
          ..
301990     1
37000      1
42690      1
140990     1
23490      1
Name: MRP, Length: 251, dtype: int64
```

In [23]:

```
print('mean value :',df['MRP'].mean())
print('median value:',df['MRP'].median())
print('std value:',df['MRP'].std())
```

```
mean value : 81605.8375
median value: 59990.0
std value: 61552.95510244973
```

In [24]:

```
print('minimum value:',df['MRP'].min())
print('maximum value:', df['MRP'].max())
```

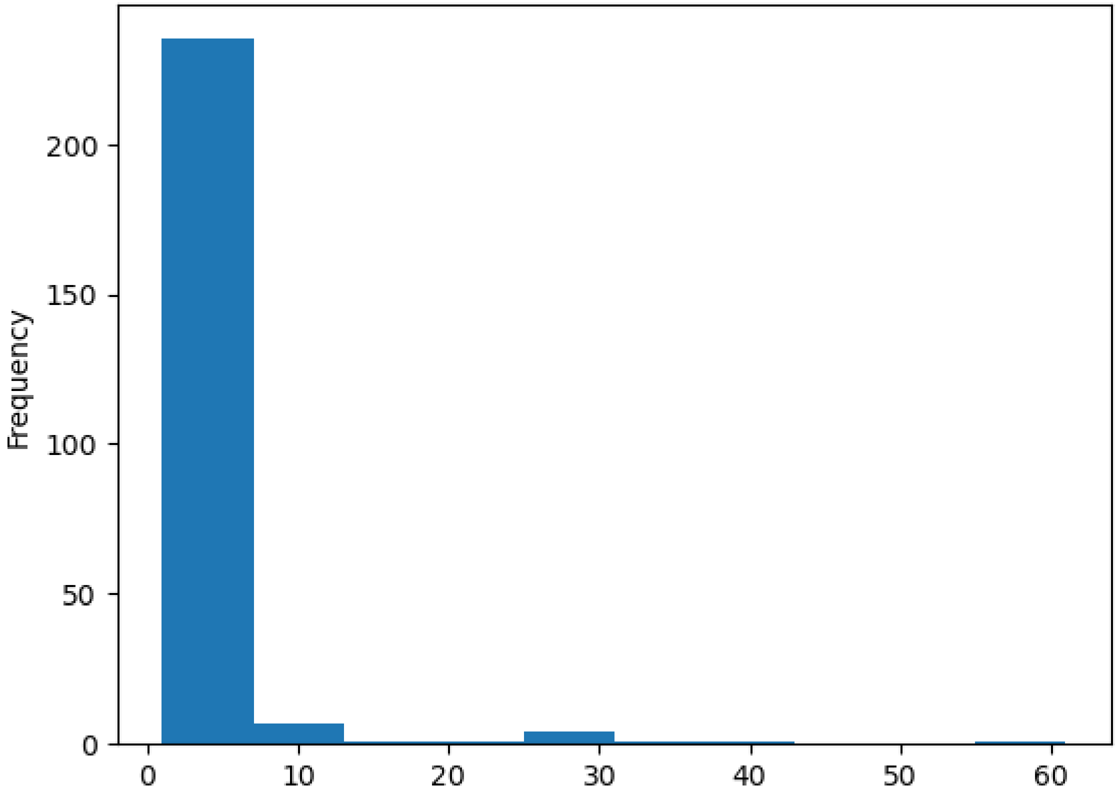
```
minimum value: 14990
maximum value: 434830
```

In [25]:

```
df['MRP'].value_counts().plot(kind='hist')
```

Out[25]:

```
<AxesSubplot:ylabel='Frequency'>
```



In [26]:

```
from sklearn.preprocessing import LabelEncoder
label = LabelEncoder()
```

In [27]:

```
df['brand']=label.fit_transform(df['brand'])
```

```
C:\Users\Safuvan\AppData\Local\Temp\ipykernel_20036\1606693397.py:1: Setti
ngWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-doc
s/stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://
pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-
view-versus-a-copy)
  df['brand']=label.fit_transform(df['brand'])
```

In [28]:

```
df.head()
```

Out[28]:

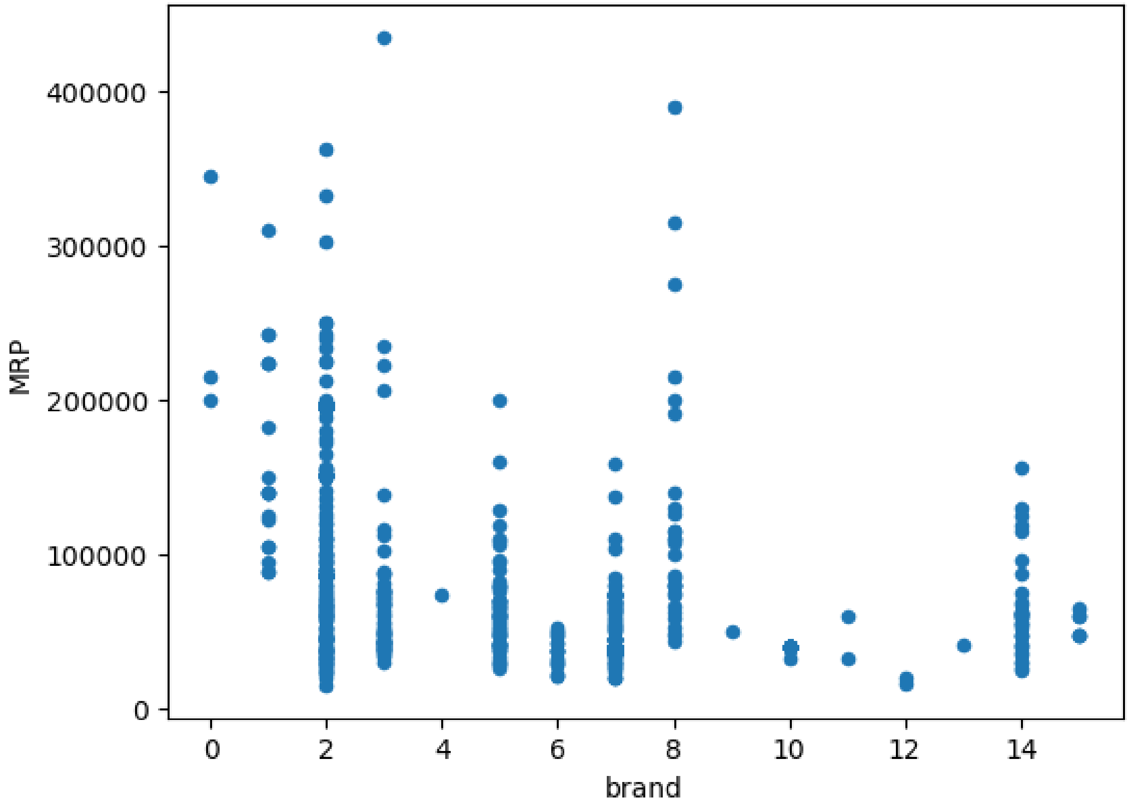
	brand	os	RAM	processor	Rating	MRP
0	7	Windows 11 Operating System	8 GB DDR4 RAM	Intel Core i3 Processor	4.2	36990
1	7	Windows 11 Operating System	8 GB DDR4 RAM	Intel Core i3 Processor	4.2	39990
2	2	Windows 11 Operating System	8 GB DDR4 RAM	Intel Core i3 Processor	4.3	32990
3	5	Windows 10 Operating System	8 GB DDR4 RAM	AMD Ryzen 5 Hexa Core Processor	4.4	49990
4	2	Windows 11 Operating System	8 GB DDR4 RAM	Intel Core i5 Processor	4.4	49990

In [29]:

```
df.plot(kind='scatter',x='brand',y='MRP')
```

Out[29]:

```
<AxesSubplot:xlabel='brand', ylabel='MRP'>
```



In [30]:

```
ram_label = LabelEncoder()
df['RAM']=ram_label.fit_transform(df['RAM'])
```

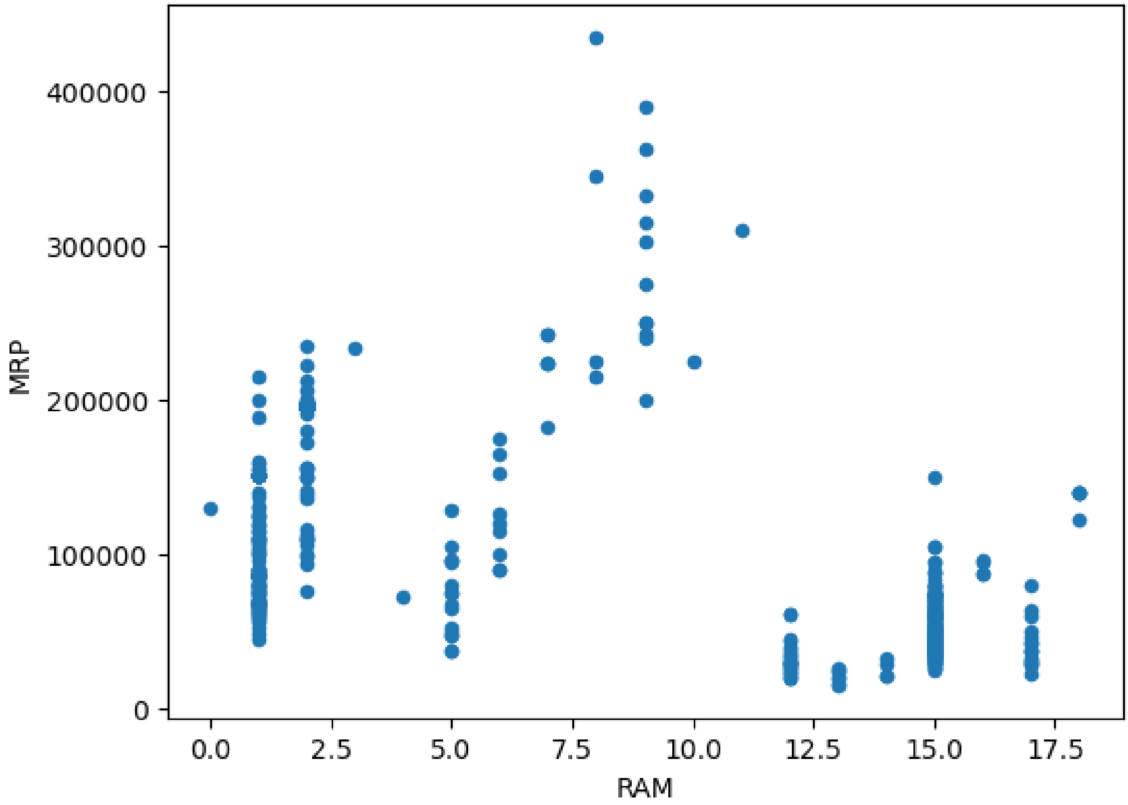
```
C:\Users\Safuvan\AppData\Local\Temp\ipykernel_20036\3658044334.py:2: Setti
ngWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-doc
s/stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://
pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-
view-versus-a-copy)
  df['RAM']=ram_label.fit_transform(df['RAM'])
```

In [31]:

```
df.plot(kind='scatter',x='RAM',y='MRP')
```

Out[31]:

```
<AxesSubplot:xlabel='RAM', ylabel='MRP'>
```

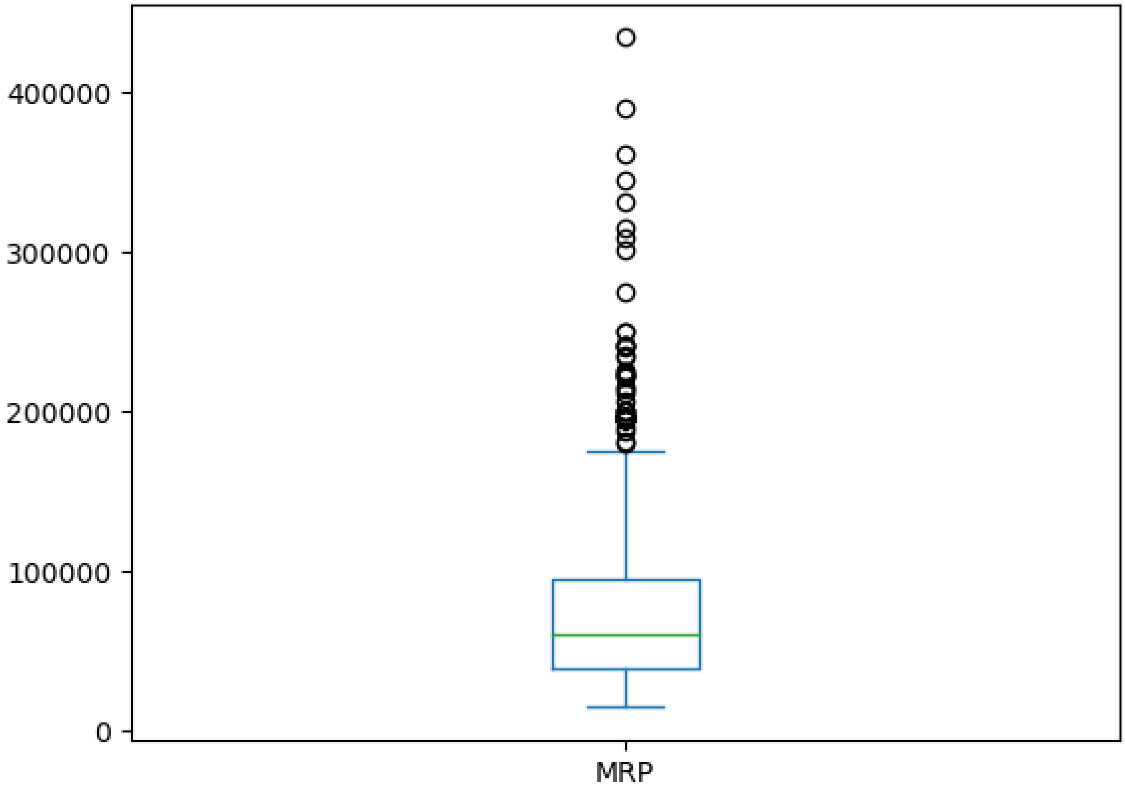


In [32]:

```
df.plot(kind='box',x='processor',y='MRP')
```

Out[32]:

```
<AxesSubplot:>
```



In [33]:

```
df['RAM'].unique()
```

Out[33]:

```
array([15,  1,  2, 13, 12, 18, 17,  7, 14,  5,  6, 11,  4,  9,  0, 16,  8,
       10,  3])
```

In [34]:

```
df.head()
```

Out[34]:

	brand	os	RAM	processor	Rating	MRP
0	7	Windows 11 Operating System	15	Intel Core i3 Processor	4.2	36990
1	7	Windows 11 Operating System	15	Intel Core i3 Processor	4.2	39990
2	2	Windows 11 Operating System	15	Intel Core i3 Processor	4.3	32990
3	5	Windows 10 Operating System	15	AMD Ryzen 5 Hexa Core Processor	4.4	49990
4	2	Windows 11 Operating System	15	Intel Core i5 Processor	4.4	49990

In [35]:

```
processer = LabelEncoder()
df['processor']=processer.fit_transform(df['processor'])
```

```
C:\Users\Safuvan\AppData\Local\Temp\ipykernel_20036\3617156379.py:2: Setti
ngWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-doc
s/stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://
pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-
view-versus-a-copy)
  df['processor']=processer.fit_transform(df['processor'])
```

In [36]:

```
os = LabelEncoder()
df['os']= os.fit_transform(df['os'])
```

```
C:\Users\Safuvan\AppData\Local\Temp\ipykernel_20036\985728333.py:2: Settin
gWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-doc
s/stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://
pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-
view-versus-a-copy)
  df['os']= os.fit_transform(df['os'])
```

In [37]:

```
df.head()
```

Out[37]:

	brand	os	RAM	processor	Rating	MRP
0	7	5	15	14	4.2	36990
1	7	5	15	14	4.2	39990
2	2	5	15	14	4.3	32990
3	5	3	15	7	4.4	49990
4	2	5	15	15	4.4	49990

In [38]:

```
df.isnull().sum()
```

Out[38]:

```
brand          0
os             0
RAM            0
processor      0
Rating       139
MRP            0
dtype: int64
```

In [39]:

```
from sklearn.model_selection import train_test_split

x =df.drop(['MRP','Rating'],axis=1)
```

In [40]:

```
y = df['MRP']
```

In [41]:

```
x.columns
```

Out[41]:

```
Index(['brand', 'os', 'RAM', 'processor'], dtype='object')
```

In [42]:

```
x_train,x_test,y_train,y_test =train_test_split(x,y,test_size=0.2,random_state=42)
```

In [43]:

```
print('x_train :',x_train.shape)
print('x_test :',x_test.shape)
print('y_train :',y_train.shape)
print('y_test :',y_test.shape)
```

```
x_train : (576, 4)
x_test : (144, 4)
y_train : (576,)
y_test : (144,)
```

In [44]:

```
from sklearn.ensemble import RandomForestRegressor

model = RandomForestRegressor()
```

In [45]:

```
model.fit(x_train,y_train)
```

Out[45]:

```
▾ RandomForestRegressor
RandomForestRegressor()
```

In [46]:

```
prediction = model.predict(x_test)
```

In [49]:

```
from sklearn.metrics import r2_score,mean_absolute_error
```

In [50]:

```
print(r2_score(prediction,y_test))
print(mean_absolute_error(prediction,y_test))
```

```
0.9003842855303008
9469.831915380848
```

In [ ]:

```

```