In [4]:
```
#  Importing required libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [5]:
```
import warnings
warnings.filterwarnings("ignore")
```

Task 1 :- Understanding Order Amount Distribution

In [7]:
```
# Converting excel file to into python using pandas
order_amt_df = pd.read_csv('order_items_dataset.csv')
order_amt_df
```

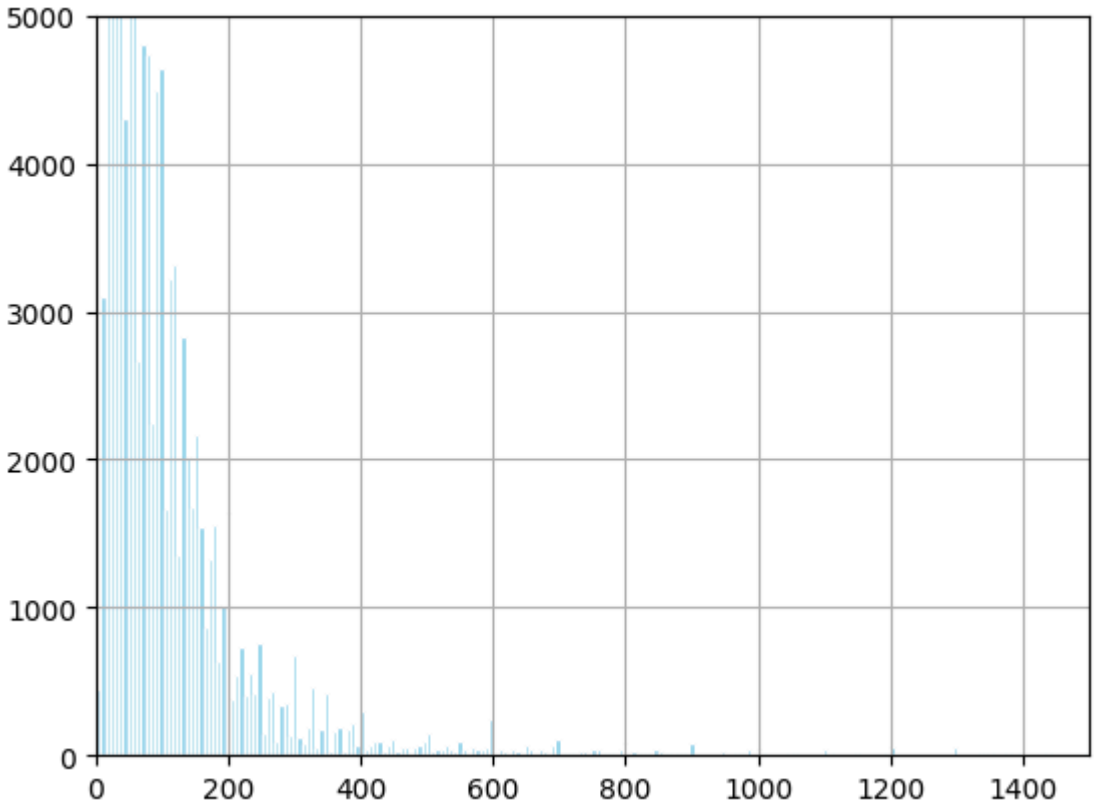
Out[7]:

	order_id	order_item_id	proc
0	00010242fe8c5a6d1ba2dd792cb16214	1	4244733e06e7ecb4970a6e2683
1	00018f77f2f0320c557190d7a144bdd3	1	e5f2d52b802189ee658865ca93
2	000229ec398224ef6ca0657da4fc703e	1	c777355d18b72b67abbeef9df4
3	00024acbcdf0a6daa1e931b038114c75	1	7634da152a4610f1595efa32f1
4	00042b26cf59d7ce69dfabb4e55b4fd9	1	ac6c3623068f30de03045865e4
...	...	...	
112645	fffc94f6ce00a00581880bf54a75a037	1	4aa6014eceb682077f9dc4bffel
112646	fffcd46ef2263f404302a634eb57f7eb	1	32e07fd915822b0765e448c4dd
112647	fffce4705a9662cd70adb13d4a31832d	1	72a30483855e2eafc67aee5dc2
112648	fffe18544ffabc95dfada21779c9644f	1	9c422a519119dcad7575db5af1
112649	fffe41c64501cc87c801fd61db3f6244	1	350688d9dc1e75ff97be326363

112650 rows × 7 columns

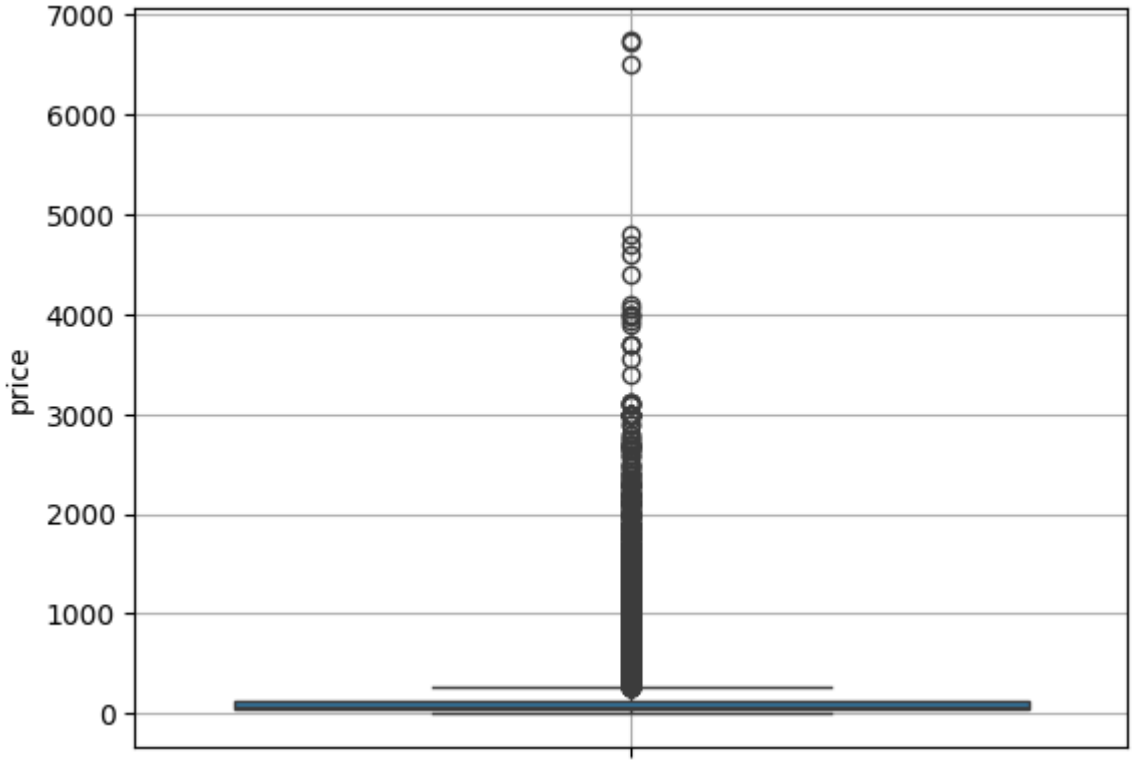
Histogram for order amount

In [9]:
```
plt.hist(order_amt_df['price'],bins = 1000 ,color='skyblue',edgecolor='white')
plt.grid(True)
plt.xlim(0,1500)
plt.ylim(0,5000)
plt.show()
```



Boxplot for identify outliers

In [11]:
```
sns.boxplot(order_amt_df['price'])
plt.grid(True)
plt.show()
```



Removing outliers

In [13]:
```
Q1 = order_amt_df['price'].quantile(0.25)
Q3 = order_amt_df['price'].quantile(0.75)
IQR = Q3-Q1
lower_bound = Q1-1.5*IQR
upper_bound = Q3+1.5*IQR
order_amt_df2 = order_amt_df[(order_amt_df['price'] >=lower_bound) & (order_amt_
order_amt_df2
```

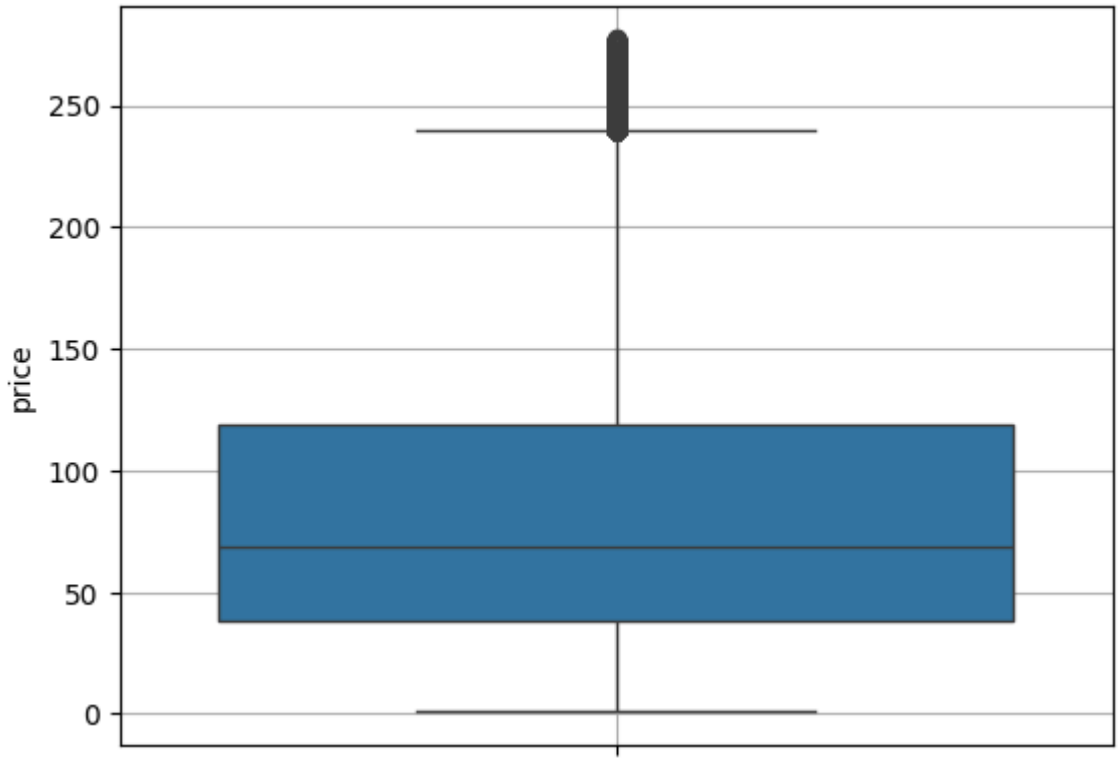
Out[13]:

	order_id	order_item_id	proc
0	00010242fe8c5a6d1ba2dd792cb16214	1	4244733e06e7ecb4970a6e2683
1	00018f77f2f0320c557190d7a144bdd3	1	e5f2d52b802189ee658865ca93
2	000229ec398224ef6ca0657da4fc703e	1	c777355d18b72b67abbeef9df4
3	00024acbcdf0a6daa1e931b038114c75	1	7634da152a4610f1595efa32f1
4	00042b26cf59d7ce69dfabb4e55b4fd9	1	ac6c3623068f30de03045865e4
...	...	...	
112643	fffb9224b6fc7c43ebb0904318b10b5f	4	43423cdffde7fda63d0414ed38
112644	fffbee3b5462987e66fb49b1c5411df2	1	6f0169f259bb0ff432bfff7d82
112647	fffce4705a9662cd70adb13d4a31832d	1	72a30483855e2eafc67aee5dc2
112648	fffe18544ffabc95dfada21779c9644f	1	9c422a519119dcad7575db5af1
112649	fffe41c64501cc87c801fd61db3f6244	1	350688d9dc1e75ff97be326363

104223 rows × 7 columns

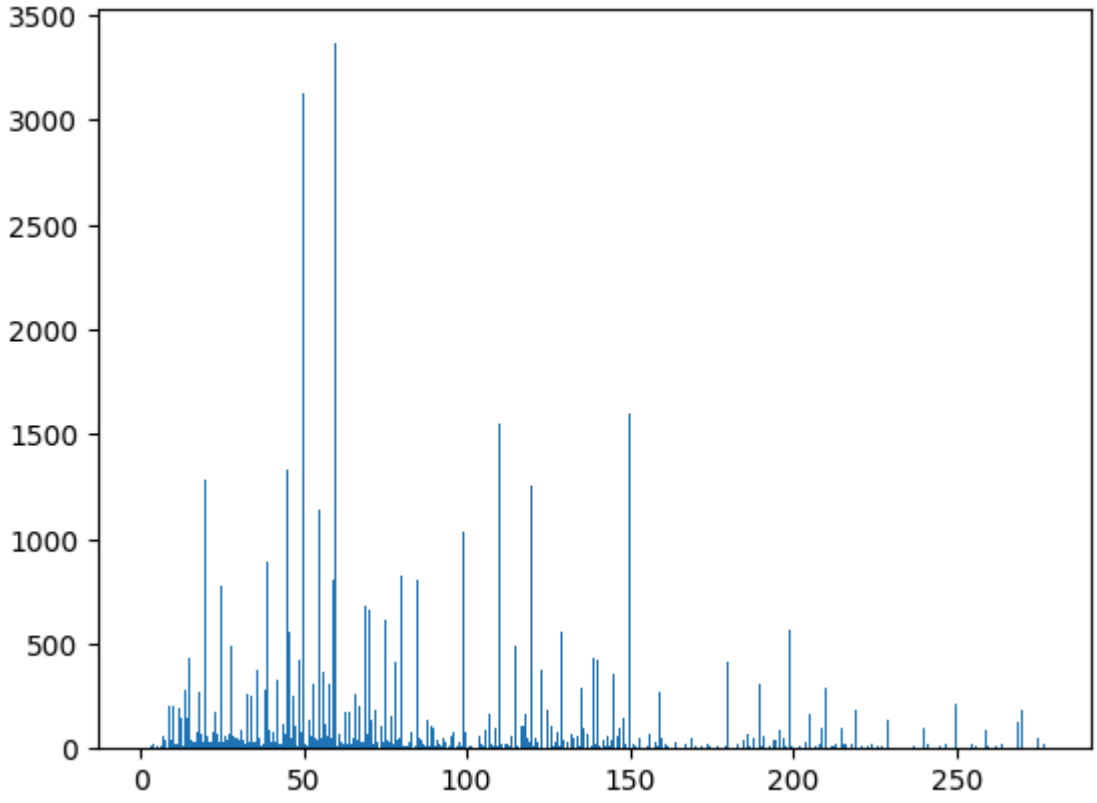
Boxplot after removing outliers

In [15]:
```
sns.boxplot(order_amt_df2['price'])
plt.grid(True)
plt.show()
```



Histogram after removing outliers

In [17]:
```
plt.hist(data = order_amt_df2 , x = 'price', bins=1000)
plt.show()
```



Task 2 :- Analyzing Product Categories

In [19]:
```
# Converting excel file to into python using pandas
product_df = pd.read_csv('products_dataset.csv')
product_df
```

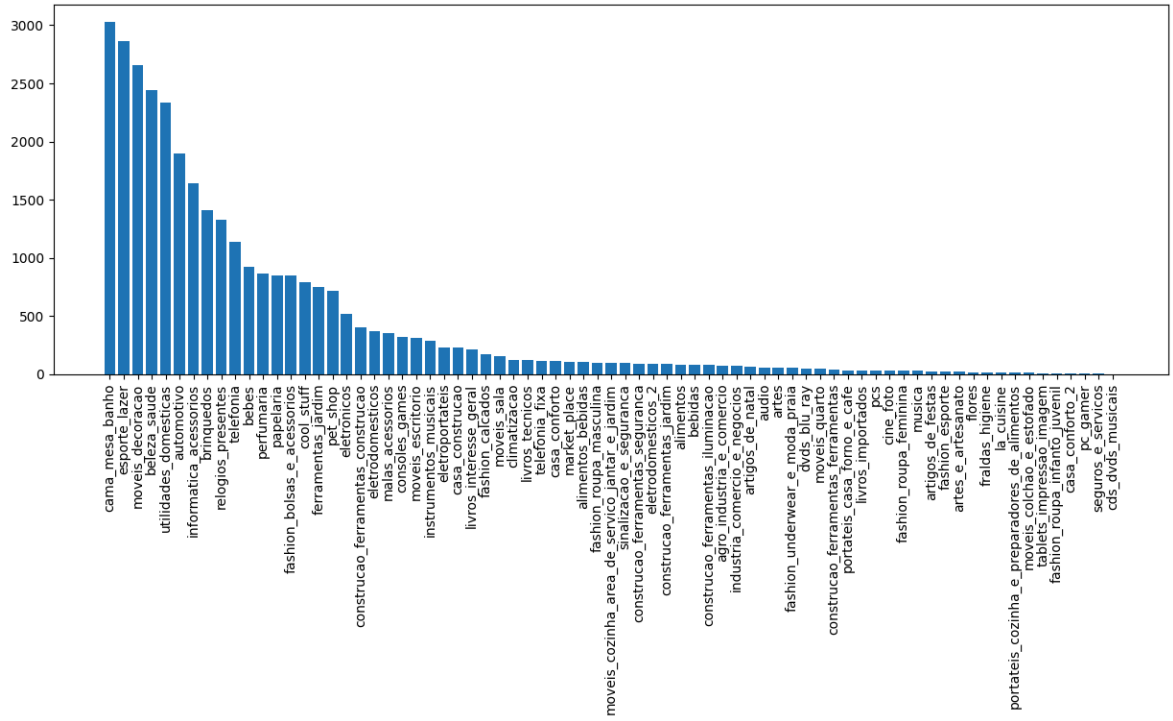
Out[19]:

	product_id	product_category_name	product_na
0	1e9e8ef04dbcff4541ed26657ea517e5	perfumaria	
1	3aa071139cb16b67ca9e5dea641aaa2f	artes	
2	96bd76ec8810374ed1b65e291975717f	esporte_lazer	
3	cef67bcfe19066a932b7673e239eb23d	bebes	
4	9dc1a7de274444849c219cff195d0b71	utilidades_domesticas	
...	...	...	
32946	a0b7d5a992ccda646f2d34e418fff5a0	moveis_decoracao	
32947	bf4538d88321d0fd4412a93c974510e6	construcao_ferramentas_iluminacao	
32948	9a7c6041fa9592d9d9ef6cfe62a71f8c	cama_mesa_banho	
32949	83808703fc0706a22e264b9d75f04a2e	informatica_acessorios	
32950	106392145fca363410d287a815be6de4	cama_mesa_banho	

32951 rows × 9 columns

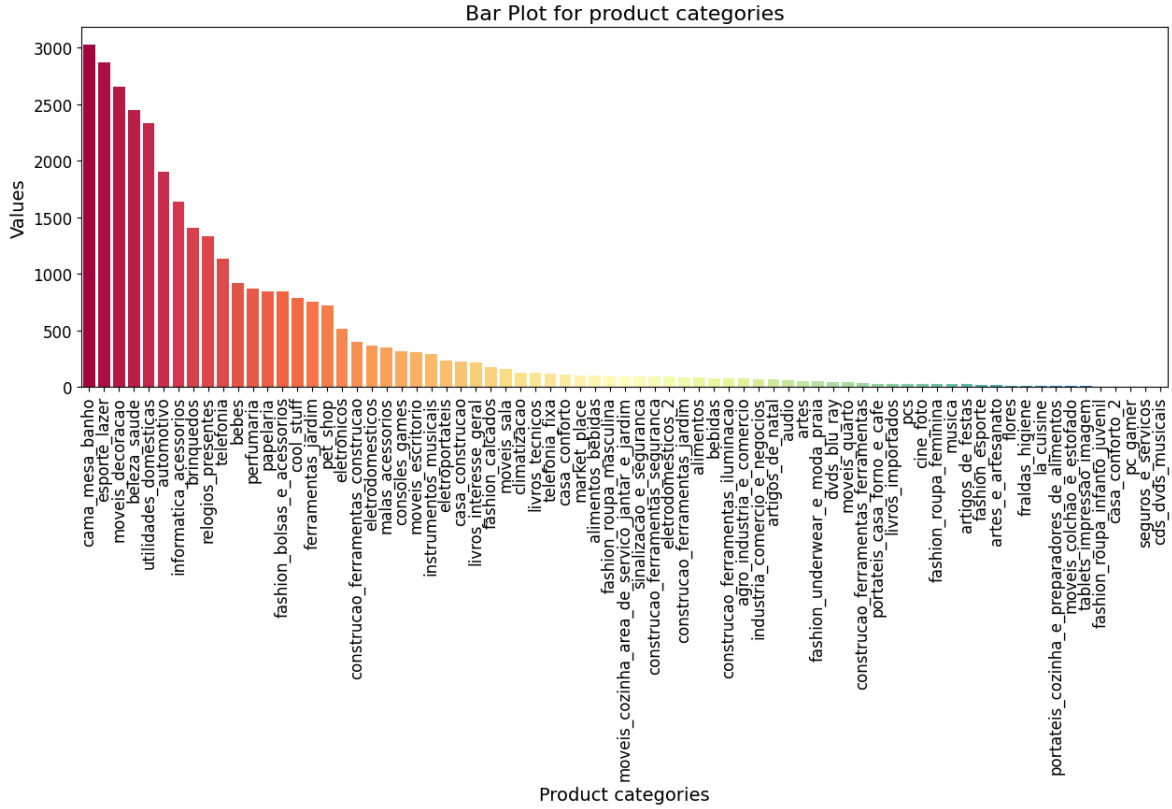
Barchart for product categories

In [21]:
```
plt.figure(figsize=(15,5))
y = product_df['product_category_name'].value_counts()
plt.bar(y.index,y.values)
plt.xticks(rotation = 90)
plt.show()
```



Enhancing Barplot in seaborn for product categories

In [23]:
```
plt.figure(figsize=(15,5))
sns.barplot(x = y.index , y = y.values ,legend = 'auto',palette='Spectral',satur
plt.title('Bar Plot for product categories', fontsize=16)
plt.xlabel('Product categories', fontsize=14)
plt.ylabel('Values', fontsize=14)
plt.xticks(rotation=90, fontsize=12)
plt.yticks(fontsize=12)
plt.show()
```



Task 3 :- Customer Ratings Analysis

In [25]:
```
# Converting excel file to into python using pandas

reviews_df = pd.read_csv('order_reviews_dataset.csv')
reviews_df
```

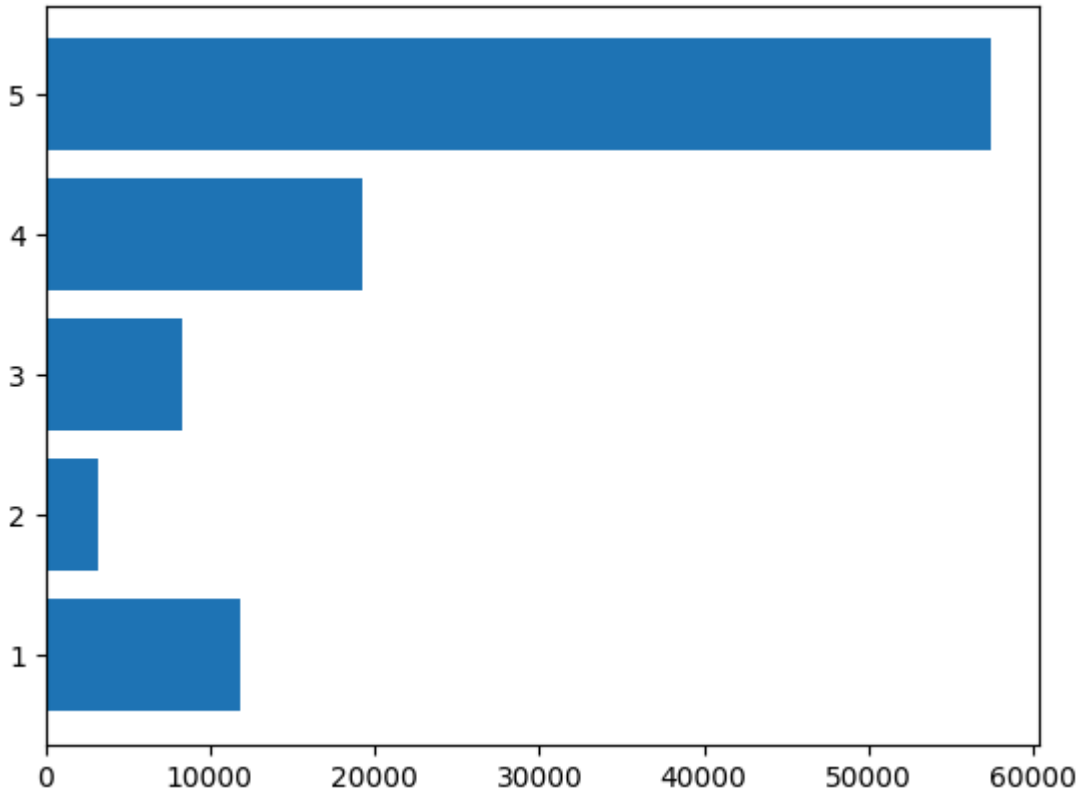
Out[25]:

	review_id	order_id	review_
0	7bc2406110b926393aa56f80a40eba40	73fc7af87114b39712e6da79b0a377eb	
1	80e641a11e56f04c1ad469d5645fdfde	a548910a1c6147796b98fdf73dbeba33	
2	228ce5500dc1d8e020d8d1322874b6f0	f9e4b658b201a9f2ecdecbb34bed034b	
3	e64fb393e7b32834bb789ff8bb30750e	658677c97b385a9be170737859d3511b	
4	f7c4243c7fe1938f181bec41a392bdeb	8e6bfb81e283fa7e4f11123a3fb894f1	
...	...	...	
99995	f3897127253a9592a73be9bdfdf4ed7a	22ec9f0669f784db00fa86d035cf8602	
99996	b3de70c89b1510c4cd3d0649fd302472	55d4004744368f5571d1f590031933e4	
99997	1adeb9d84d72fe4e337617733eb85149	7725825d039fc1f0ceb7635e3f7d9206	
99998	be360f18f5df1e0541061c87021e6d93	f8bd3f2000c28c5342fedeb5e50f2e75	
99999	efe49f1d6f951dd88b51e6ccd4cc548f	90531360ecb1eec2a1fbb265a0db0508	

100000 rows × 7 columns

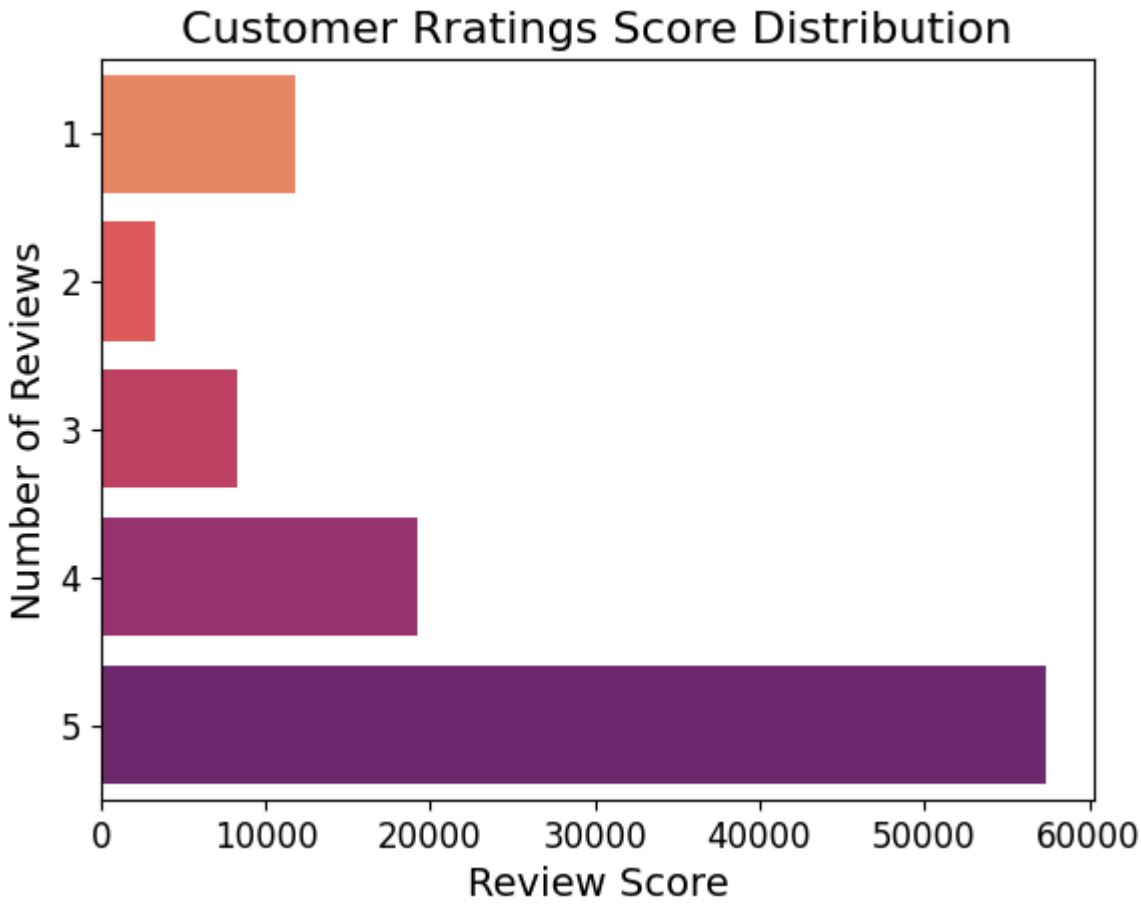
Barchart for customer ratings

In [27]:
```
y = reviews_df['review_score'].value_counts()
plt.barh(y.index,y.values)
plt.show()
```



Enhanced Barplot for customer ratings

In [29]:
```
sns.barplot(y = y.index , x = y.values ,legend='auto',palette='flare',saturation
plt.title('Customer Rratings Score Distribution', fontsize=16)
plt.ylabel('Number of Reviews', fontsize=14)
plt.xlabel('Review Score', fontsize=14)
plt.xticks(fontsize=12)
plt.yticks(fontsize=12)
plt.show()
```



Task 4 :- Statistical Summary of Delivery Times

In [31]:
```
# Converting excel file to into python using pandas

orders_df = pd.read_csv('orders_dataset.csv')
orders_df
```

Out[31]:

	order_id	customer_id	order_st
0	e481f51cbdc54678b7cc49136f2d6af7	9ef432eb6251297304e76186b10a928d	deliv
1	53cdb2fc8bc7dce0b6741e2150273451	b0830fb4747a6c6d20dea0b8c802d7ef	deliv
2	47770eb9100c2d0c44946d9cf07ec65d	41ce2a54c0b03bf3443c3d931a367089	deliv
3	949d5b44dbf5de918fe9c16f97b45f8a	f88197465ea7920adcdbec7375364d82	deliv
4	ad21c59c0840e6cb83a9ceb5573f8159	8ab97904e6daea8866dbdbc4fb7aad2c	deliv
...	...	...	
99436	9c5dedf39a927c1b2549525ed64a053c	39bd1228ee8140590ac3aca26f2dfe00	deliv
99437	63943bddc261676b46f01ca7ac2f7bd8	1fca14ff2861355f6e5f14306ff977a7	deliv
99438	83c1379a015df1e13d02aae0204711ab	1aa71eb042121263aafbe80c1b562c9c	deliv
99439	11c177c8e97725db2631073c19f07b62	b331b74b18dc79bcdf6532d51e1637c1	deliv
99440	66dea50a8b16d9b4dee7af250b4be1a5	edb027a75a1449115f6b43211ae02a24	deliv

99441 rows × 8 columns

In [32]:
```
# converting order_purchase_timestamp column string to date

orders_df['order_purchase_timestamp'] = pd.to_datetime(orders_df['order_purchase

orders_df['purchase_date'] = orders_df['order_purchase_timestamp'].dt.date
```

In [33]:
```
# converting order_delivered_customer_date column string to date

orders_df['order_delivered_customer_date'] = pd.to_datetime(orders_df['order_del

orders_df['delivery_date'] = orders_df['order_delivered_customer_date'].dt.date
```

In [34]:
```
# Calculating total delivery days
# from date of order to date of delivery

orders_df['purchase_date'] = pd.to_datetime(orders_df['purchase_date'])
orders_df['delivery_date'] = pd.to_datetime(orders_df['delivery_date'])

orders_df['date_diff'] = (orders_df['delivery_date'] - orders_df['purchase_date'
```

In [35]:
```
# Calculating mean
mean_1 = orders_df['date_diff'].mean()
```

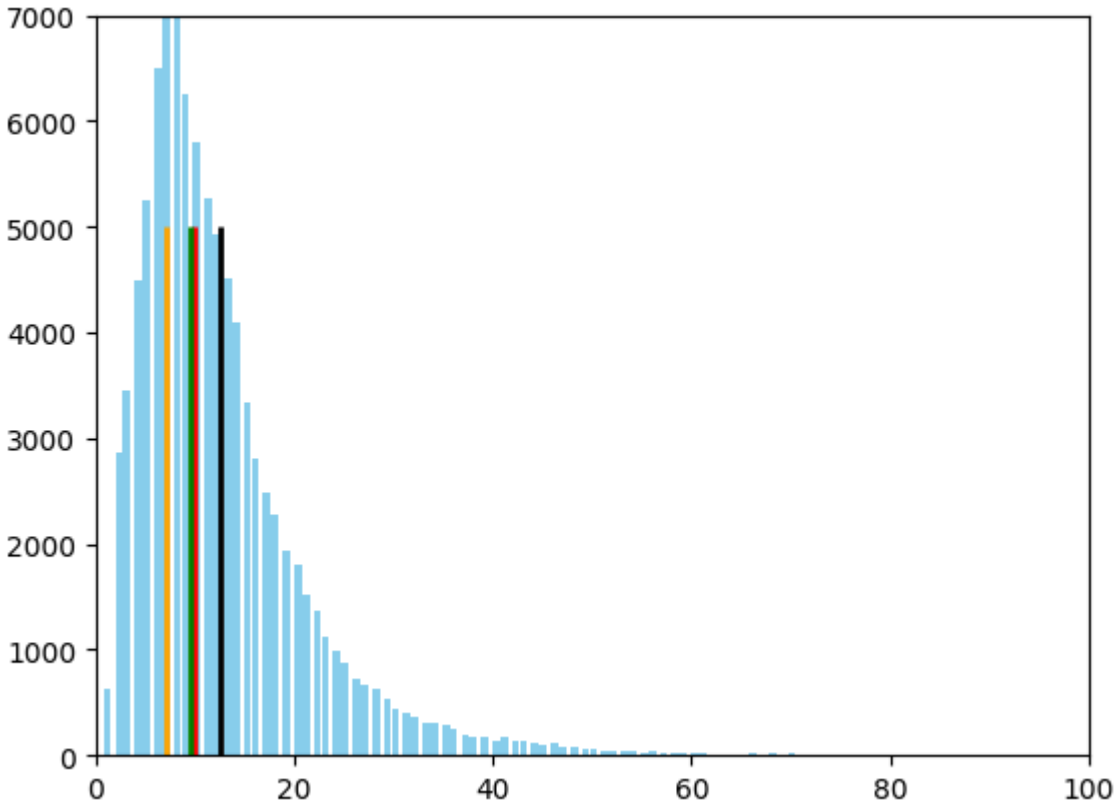
In [36]:
```
# Calculating median
mode_1 = orders_df['date_diff'].median()
```

In [37]:
```
# Calculating meadian_1
median_1 = orders_df['date_diff'].mode()
```

In [38]:
```
# Calculating standard deviation
std_dev = orders_df['date_diff'].std()
```

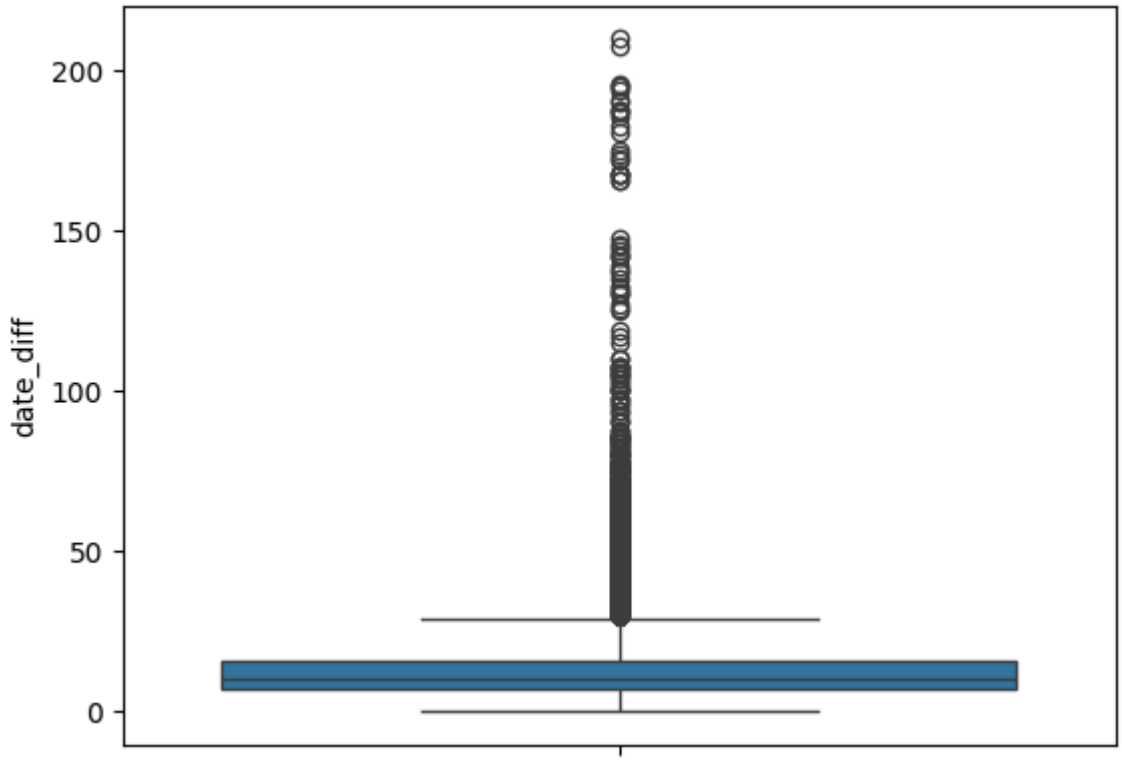
Creating Histogram For Estimate Delivery Days

In [40]:
```
plt.hist(orders_df['date_diff'],bins='auto',color='skyblue',width=0.75)
plt.xlim(0,100)
plt.ylim(0,7000)
plt.vlines(x=mean_1, ymin=0, ymax=5000, color='k', linestyle='-', linewidth=2)
plt.vlines(x=mode_1, ymin=0, ymax=5000, color='r', linestyle='-', linewidth=2)
plt.vlines(x=median_1, ymin=0, ymax=5000, color='orange', linestyle='-', linewid
plt.vlines(x=std_dev, ymin=0, ymax=5000, color='g', linestyle='-', linewidth=2)
plt.show()
```



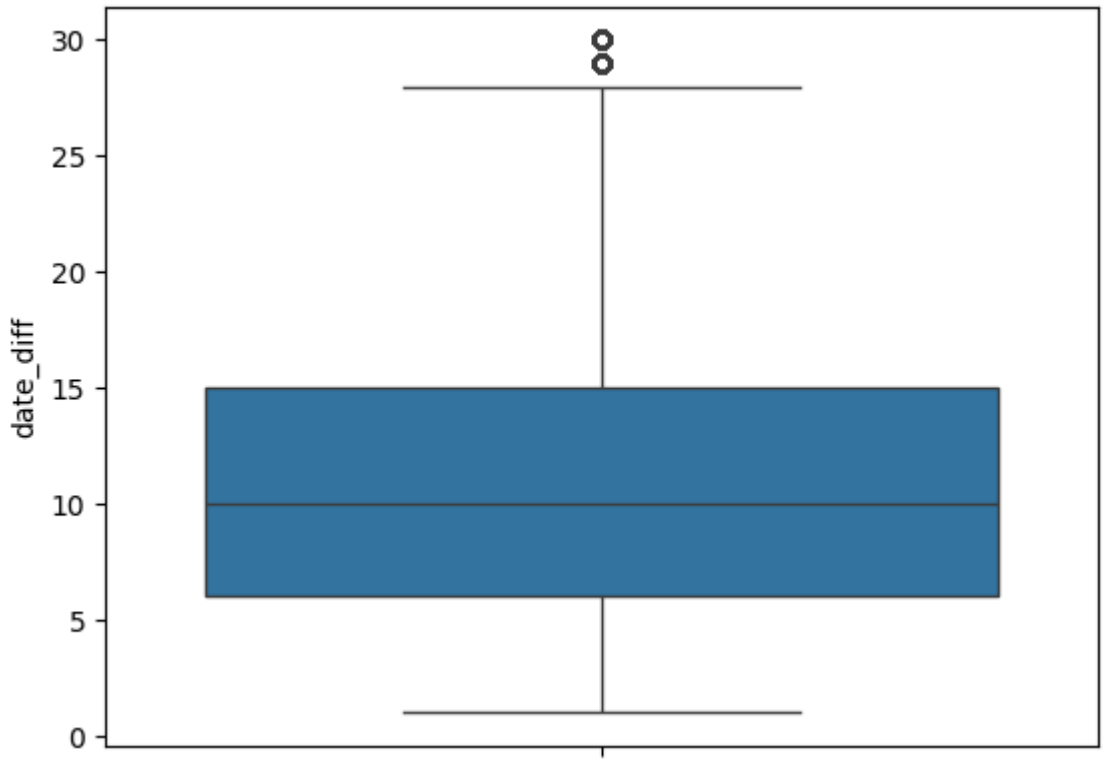
Creating Boxplot For Estimate Delivery Days

In [42]:
```
sns.boxplot(orders_df['date_diff'])
plt.show()
```



In [43]:
```
# Removing outliers

df_time_filtered = orders_df[(orders_df['date_diff'] > 0) & (orders_df['date_dif
sns.boxplot(df_time_filtered['date_diff'])
plt.show()
```



Task 5 :- Relationship Between Order Value and Delivery Time

In [45]:
```
# Converting excel file to into python using pandas

merged_df = pd.merge(order_amt_df, orders_df, on='order_id')
merged_df
```

Out[45]:

	order_id	order_item_id	proc
0	00010242fe8c5a6d1ba2dd792cb16214	1	4244733e06e7ecb4970a6e2683
1	00018f77f2f0320c557190d7a144bdd3	1	e5f2d52b802189ee658865ca93
2	000229ec398224ef6ca0657da4fc703e	1	c777355d18b72b67abbeef9df4
3	00024acbcdf0a6daa1e931b038114c75	1	7634da152a4610f1595efa32f1
4	00042b26cf59d7ce69dfabb4e55b4fd9	1	ac6c3623068f30de03045865e4
...	...	...	
112645	fffc94f6ce00a00581880bf54a75a037	1	4aa6014eceb682077f9dc4bffe
112646	fffcd46ef2263f404302a634eb57f7eb	1	32e07fd915822b0765e448c4dd
112647	fffce4705a9662cd70adb13d4a31832d	1	72a30483855e2eafc67aee5dc2
112648	fffe18544ffabc95dfada21779c9644f	1	9c422a519119dcad7575db5af1
112649	fffe41c64501cc87c801fd61db3f6244	1	350688d9dc1e75ff97be326363

112650 rows × 17 columns

In [46]:
```
# Extracting the order value and delivery time column

merged_df[['price','date_diff']]
```

Out[46]:

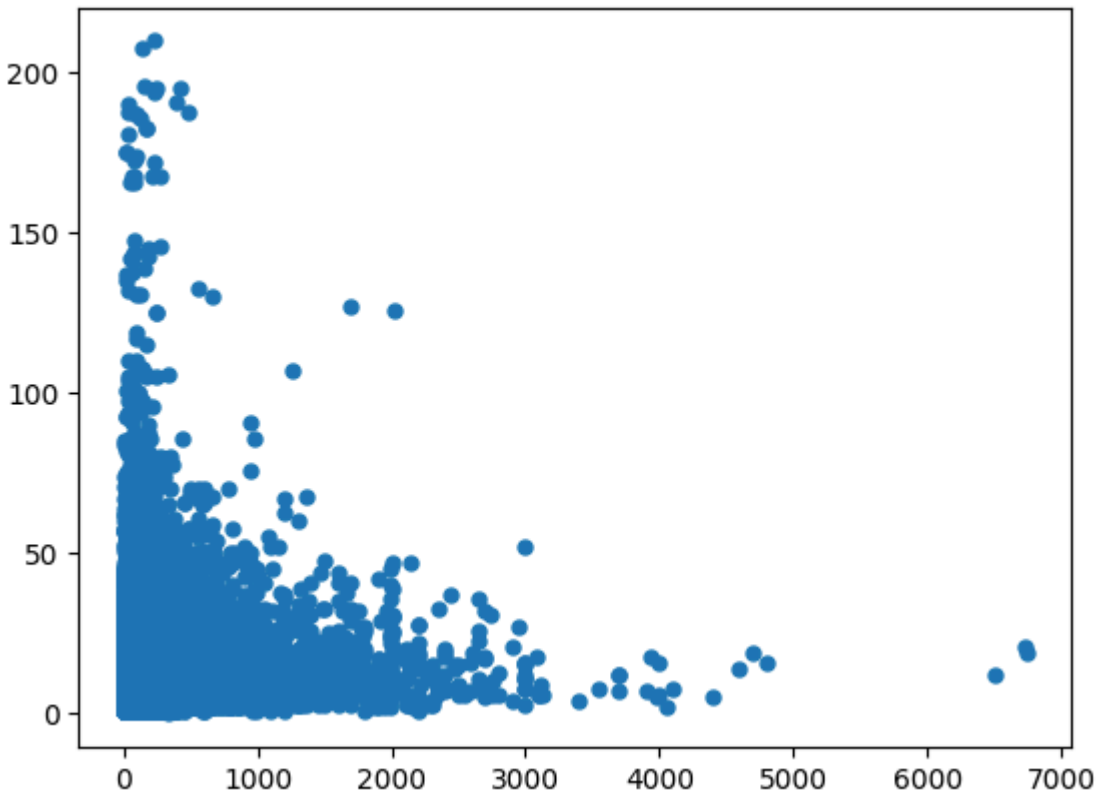
	price	date_diff
0	58.90	7.0
1	239.90	16.0
2	199.00	8.0
3	12.99	6.0
4	199.90	25.0
...	...	...
112645	299.99	17.0
112646	350.00	9.0
112647	99.90	5.0
112648	55.99	2.0
112649	43.00	5.0

112650 rows × 2 columns

Creating scatterplot for relationship between order value vs. delivery time

In [110…

```
plt.scatter(merged_df['price'],merged_df['date_diff'], s=25)
plt.show()
```



In [49]:

```
# Correlation coefficient between price and delivery days
```

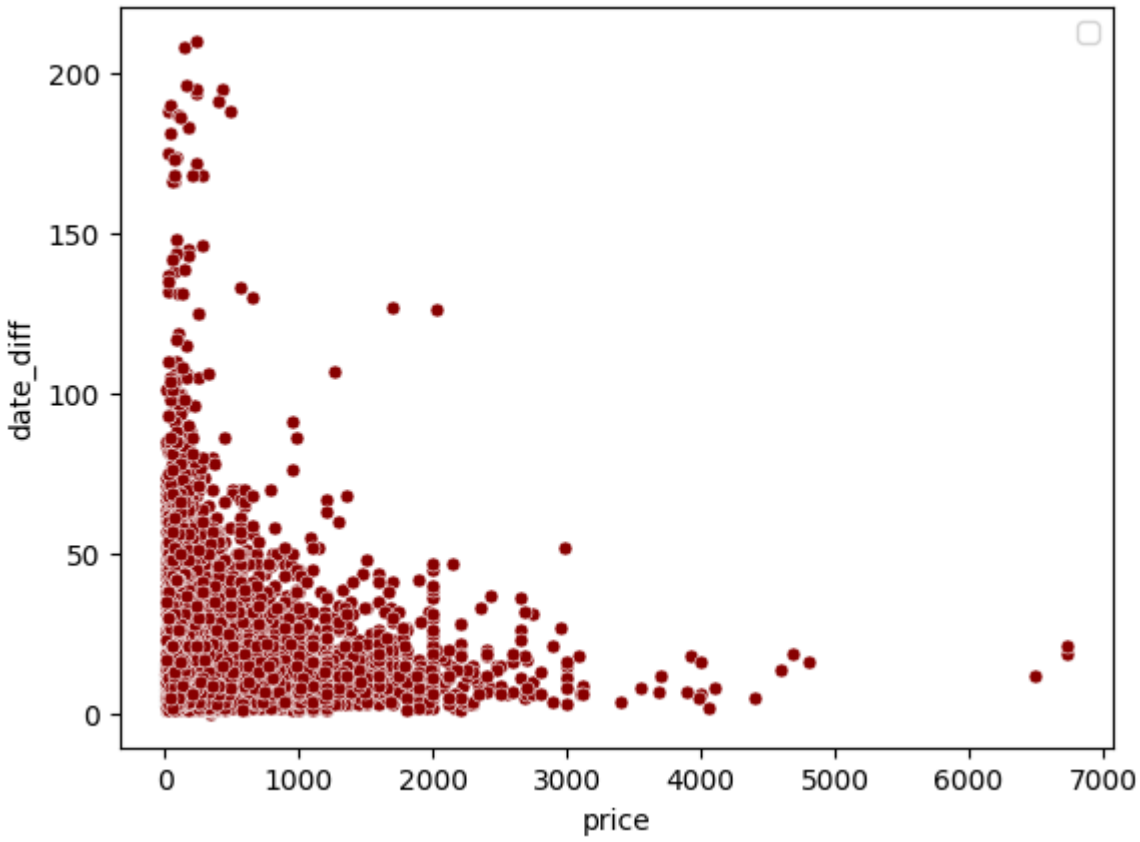
```
corr_1 = merged_df['price'].corr(merged_df['date_diff'])
corr_1
```

Out[49]: 0.06269448164982502

Enhancing scatterplot for relationship between order value vs. delivery time

In [104…

```
sns.scatterplot(data = merged_df , x = 'price' , y = 'date_diff' , s=25, color="
plt.legend()
plt.show()
```



In [ ]: