In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from datetime import datetime
import calendar
from pandas.api.types import CategoricalDtype
```

In [4]:
```
customers = pd.read_csv("olist_customers_dataset.csv")
items = pd.read_csv("olist_order_items_dataset.csv")
payments = pd.read_csv("olist_order_payments_dataset.csv")
orders = pd.read_csv("olist_orders_dataset.csv")
products = pd.read_csv("olist_products_dataset.csv")
category_translation = pd.read_csv("product_category_name_translation.csv")
```

In [5]:
```
datasets = [customers, items, payments, orders, products, category_translation]
titles = ["customers", "items", "payments", "orders", "products", "category_translation

#To gather practical information about all datasets
info_df = pd.DataFrame({},)
info_df['dataset']= titles
info_df['cols'] = [', '.join([col for col, null in df.isnull().sum().items() ]) for df
info_df['cols_no']= [df.shape[1] for df in datasets]
info_df['null_no']= [df.isnull().sum().sum() for df in datasets]
info_df['null_cols_no']= [len([col for col, null in df.isnull().sum().items() if null >
info_df['null_cols'] = [', '.join([col for col, null in df.isnull().sum().items() if nu


info_df.style.background_gradient(cmap='coolwarm')
```

Out[5]:

	dataset	cols	cols_no	null_no	null_cols_no	
0	customers	customer_id, customer_unique_id, customer_zip_code_prefix, customer_city, customer_state	5	0	0	
1	items	order_id, order_item_id, product_id, seller_id, shipping_limit_date, price, freight_value	7	0	0	
2	payments	order_id, payment_sequential, payment_type, payment_installments, payment_value	5	0	0	
3	orders	order_id, customer_id, order_status, order_purchase_timestamp, order_approved_at, order_delivered_carrier_date, order_delivered_customer_date, order_estimated_delivery_date	8	4908	3	order_appı order_delivered_carı order_delivered_custor

	dataset	cols	cols_no	null_no	null_cols_no	
4	products	product_id, product_category_name, product_name_lenght, product_description_lenght, product_photos_qty, product_weight_g, product_length_cm, product_height_cm, product_width_cm	9	2448	8	product_catego product_nam product_descriptio product_ph product_v product_le product_he product_v
5	category_translation	product_category_name, product_category_name_english	2	0	0	

In [6]:

```
df = pd.merge(orders,payments, on="order_id")
df = df.merge(customers, on="customer_id")
df = df.merge(items, on="order_id")
df = df.merge(products, on="product_id")
df = df.merge(category_translation, on="product_category_name")

df.dropna(inplace=True)  #Keep the DataFrame with valid entries in the same variable

df.info()
df.isnull().sum().sort_values()  # To get the number of missing value in each row if th
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 113367 entries, 0 to 115877
Data columns (total 31 columns):
 #   Column                         Non-Null Count   Dtype
---  ------                         --------------   -----
 0   order_id                       113367 non-null  object
 1   customer_id                    113367 non-null  object
 2   order_status                   113367 non-null  object
 3   order_purchase_timestamp       113367 non-null  object
 4   order_approved_at              113367 non-null  object
 5   order_delivered_carrier_date   113367 non-null  object
 6   order_delivered_customer_date  113367 non-null  object
 7   order_estimated_delivery_date  113367 non-null  object
 8   payment_sequential             113367 non-null  int64
 9   payment_type                   113367 non-null  object
 10  payment_installments           113367 non-null  int64
 11  payment_value                  113367 non-null  float64
 12  customer_unique_id             113367 non-null  object
 13  customer_zip_code_prefix       113367 non-null  int64
 14  customer_city                  113367 non-null  object
 15  customer_state                 113367 non-null  object
 16  order_item_id                  113367 non-null  int64
 17  product_id                     113367 non-null  object
 18  seller_id                      113367 non-null  object
 19  shipping_limit_date            113367 non-null  object
 20  price                          113367 non-null  float64
 21  freight_value                  113367 non-null  float64
 22  product_category_name          113367 non-null  object
 23  product_name_lenght            113367 non-null  float64
 24  product_description_lenght     113367 non-null  float64
 25  product_photos_qty             113367 non-null  float64
 26  product_weight_g               113367 non-null  float64
 27  product_length_cm              113367 non-null  float64
 28  product_height_cm              113367 non-null  float64
 29  product_width_cm               113367 non-null  float64
```

```
 30  product_category_name_english  113367 non-null  object
dtypes: float64(10), int64(4), object(17)
memory usage: 27.7+ MB
```

Out[6]:

```
order_id                          0
product_height_cm                 0
product_length_cm                 0
product_weight_g                  0
product_photos_qty                0
product_description_lenght        0
product_name_lenght               0
product_category_name             0
freight_value                     0
price                             0
shipping_limit_date               0
seller_id                         0
product_id                        0
order_item_id                     0
product_width_cm                  0
customer_state                    0
customer_zip_code_prefix          0
customer_unique_id                0
payment_value                     0
payment_installments              0
payment_type                      0
payment_sequential                0
order_estimated_delivery_date     0
order_delivered_customer_date     0
order_delivered_carrier_date      0
order_approved_at                 0
order_purchase_timestamp          0
order_status                      0
customer_id                       0
customer_city                     0
product_category_name_english     0
dtype: int64
```

In [7]:

```
df.sample(10)
```

Out[7]:

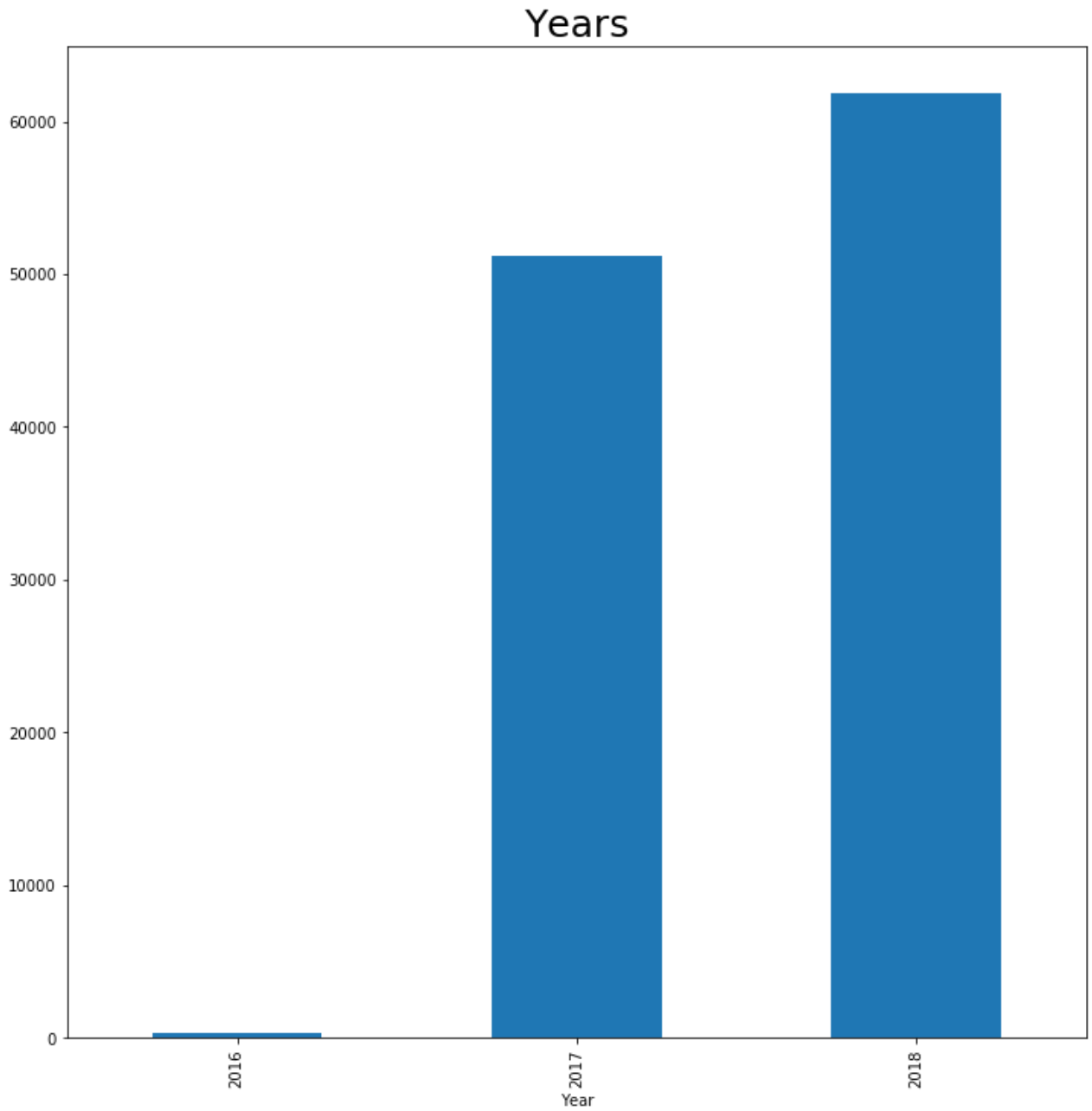
	order_id	customer_id	order_status	order_pur
58514	8bbaf58bf6dba86510e280b874e88425	4e87cf360a8ab537d9af03568d747a1d	delivered	20
52521	8ee32c0f656d4816294802830cd9c2cd	4971dbc8a7c2e77aacbd6fe6064dc45c	delivered	20
32286	2e6e6de992c5567f41aa94497d505e77	e27c13a71b8c090c5df94f96471e7994	delivered	20
114643	c172f96808d5e1df5a287d171592d3c6	13597bcb1cdff68183732f71fdb6dce6	delivered	20
79061	99d77f27c8ce75d51ce78cfe695ca1c3	eb4f40368c958563004db00314d45274	delivered	20
64207	2502fdbda37b0748046e3b6c9d33e7ea	19a81cc55bd0e8ad389c7b787c9c280d	delivered	20
64103	a894b3482b08ff3282f5a9c4485b7978	8bcf893cb7d4dc37ff1dbab13b19aaab	delivered	20
66523	67cb060da41114f8464b736e9e49058e	ea36abf64bc24b2bbe8d2e894728c498	delivered	20

	order_id	customer_id	order_status	order_pur
87855	4f037f852cf9b1a96760775d55221c33	099da9e0fc3ea53e74144c6f8427ae07	delivered	20
19219	a69cfdf2a1c59f8ff05718f69d021e81	a4a6227ac25717d6495f0ac16be85017	delivered	20

10 rows × 31 columns

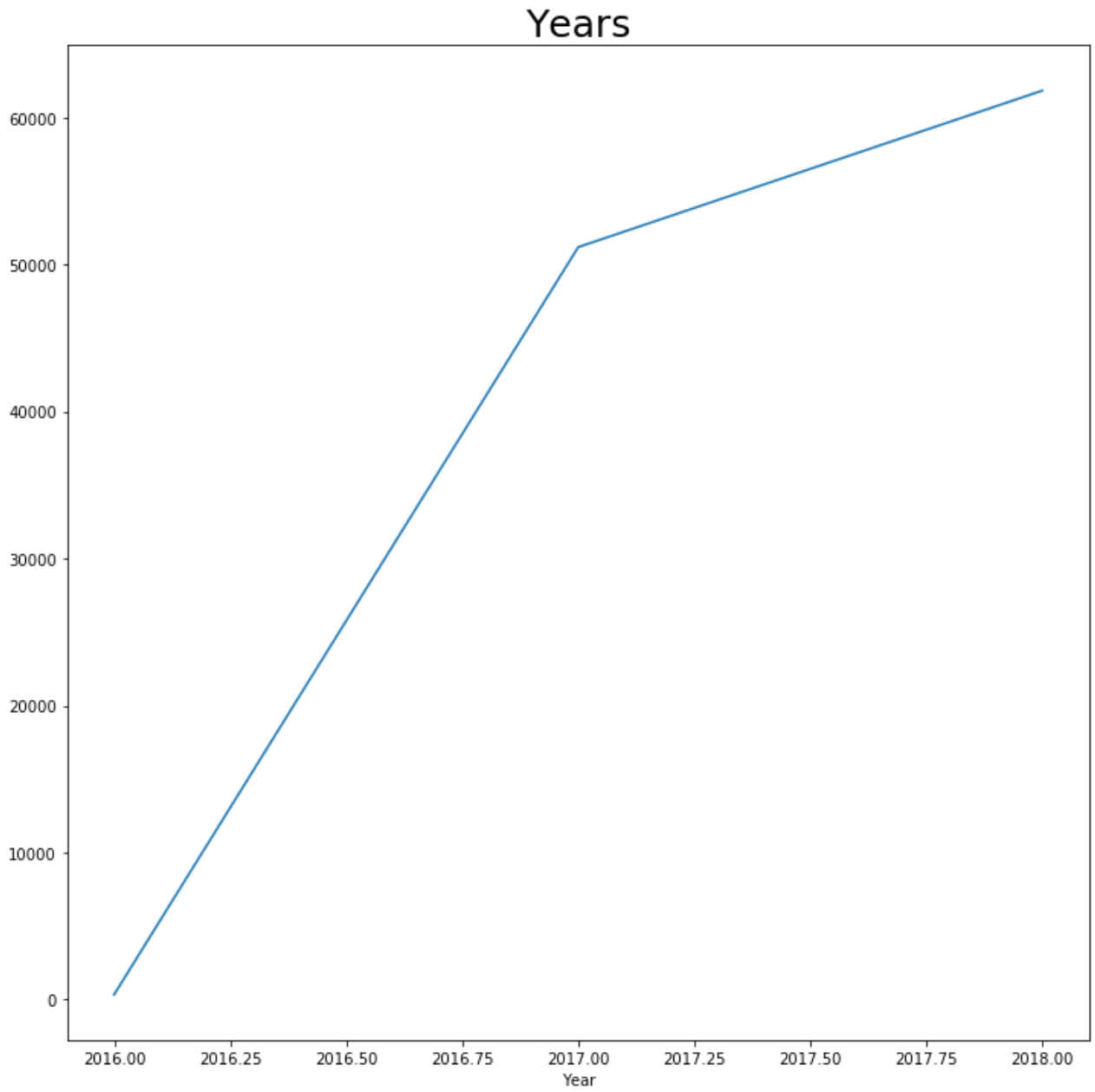
In [8]:
```
df[['order_purchase_timestamp', 'order_delivered_customer_date']]=df[['order_purchase_t

# To extract "year" from date.
df['Year'] = df['order_purchase_timestamp'].dt.year
df.groupby('Year').size().plot(
    kind = 'bar',
    figsize=(12,12),
)
plt.title('Years',fontsize=25)
```

Out[8]: Text(0.5, 1.0, 'Years')



In [9]:
```
df['Year'] = df['order_purchase_timestamp'].dt.year
df.groupby('Year').size().plot(
    kind = 'line',
    figsize=(12,12)
)
plt.title('Years',fontsize=25)
```

Out[9]:
```
Text(0.5, 1.0, 'Years')
```



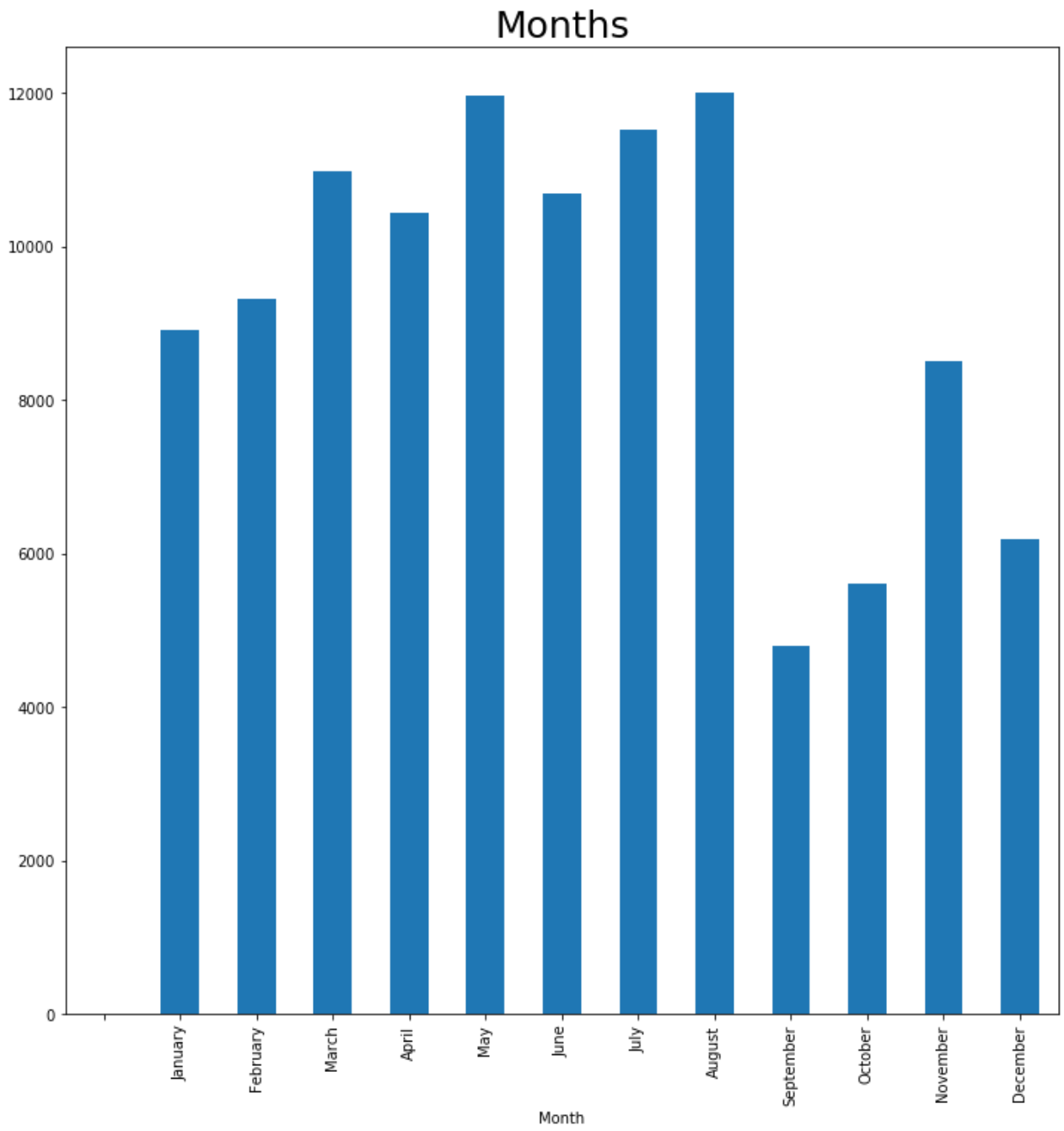
In [10]:
```
df['Year']
```

Out[10]:
```
0         2017
1         2017
2         2017
3         2017
4         2017
          ...
115873    2018
115874    2018
115875    2018
115876    2017
115877    2017
Name: Year, Length: 113367, dtype: int64
```

In [11]:
```
df['Month'] =  pd.Series(pd.Categorical(df['order_purchase_timestamp'].dt.month_name(),
df['Month']
df.groupby('Month').size().plot(
    kind = 'bar',
```

```
    figsize=(12,12)
)
plt.title('Months',fontsize=25)
```

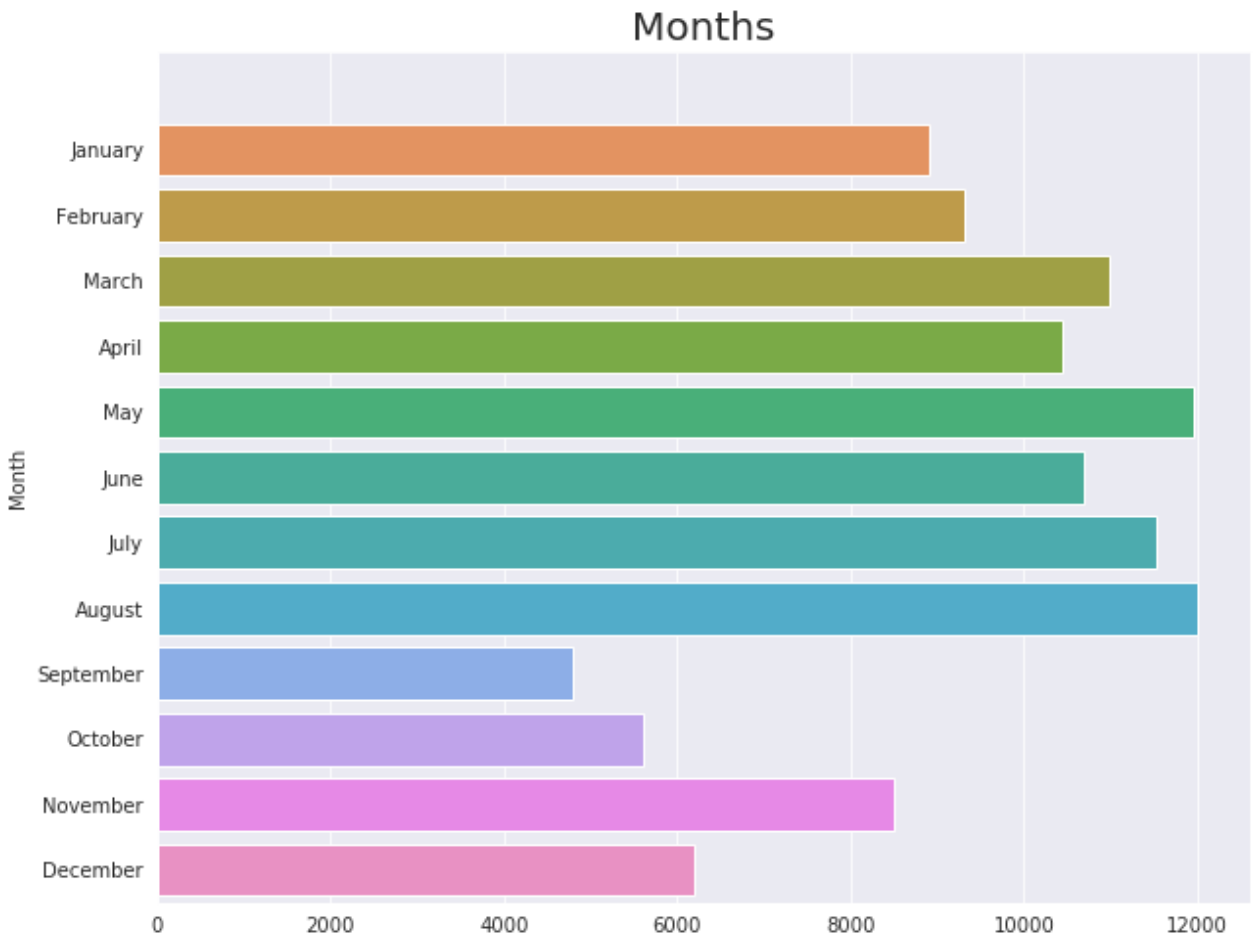
Out[11]: Text(0.5, 1.0, 'Months')



In [12]:

```
month = df.groupby('Month').size().sort_values()
fig=plt.figure(figsize=(10,8))
sns.set_style("darkgrid")
sns.barplot(y=month.index, x=month.values)
plt.title('Months',fontsize=20)
```

Out[12]: Text(0.5, 1.0, 'Months')

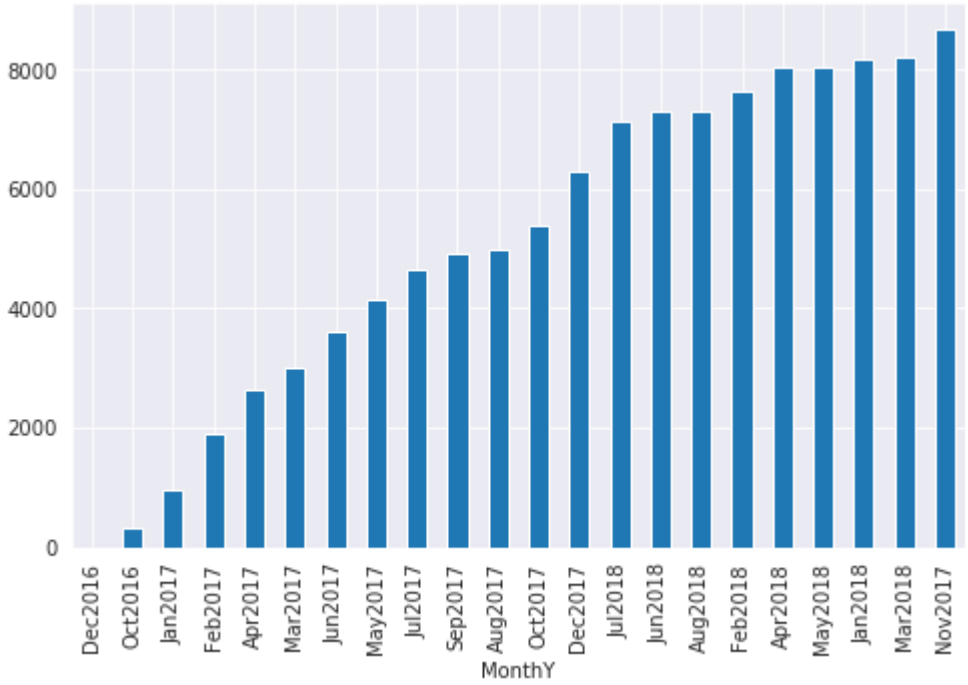


In [13]:
```
df['MonthY'] = df['order_purchase_timestamp'].dt.strftime('%b%Y')
df['MonthY']
```

Out[13]:
```
0         Oct2017
1         Oct2017
2         Oct2017
3         Aug2017
4         Aug2017
           ...
115873    Aug2018
115874    Jul2018
115875    Jul2018
115876    Jan2017
115877    Sep2017
Name: MonthY, Length: 113367, dtype: object
```

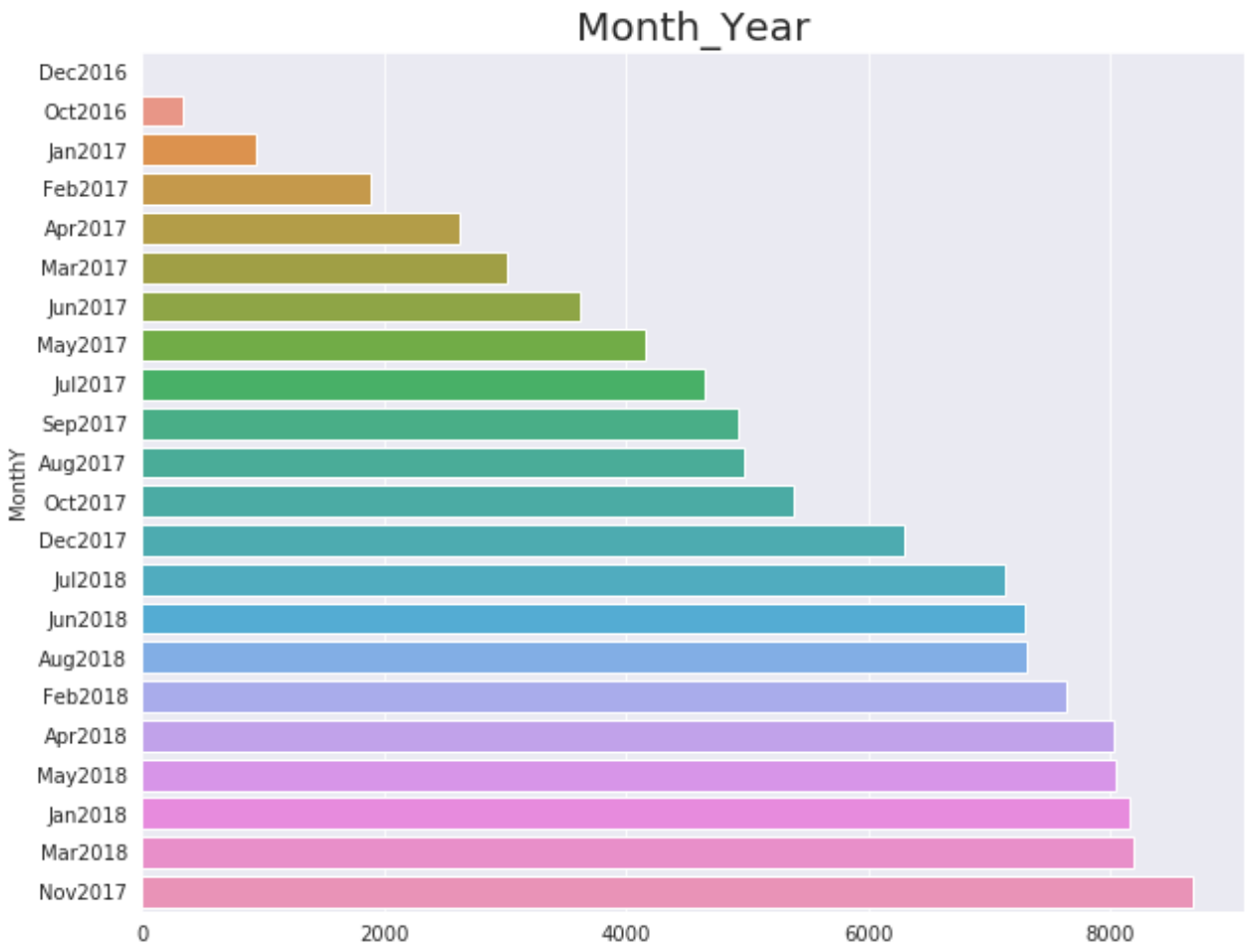
In [14]:
```
df.groupby('MonthY').size().sort_values().plot(
    kind = 'bar',
    figsize=(8,5)
)
```

Out[14]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f59cd068f10>
```



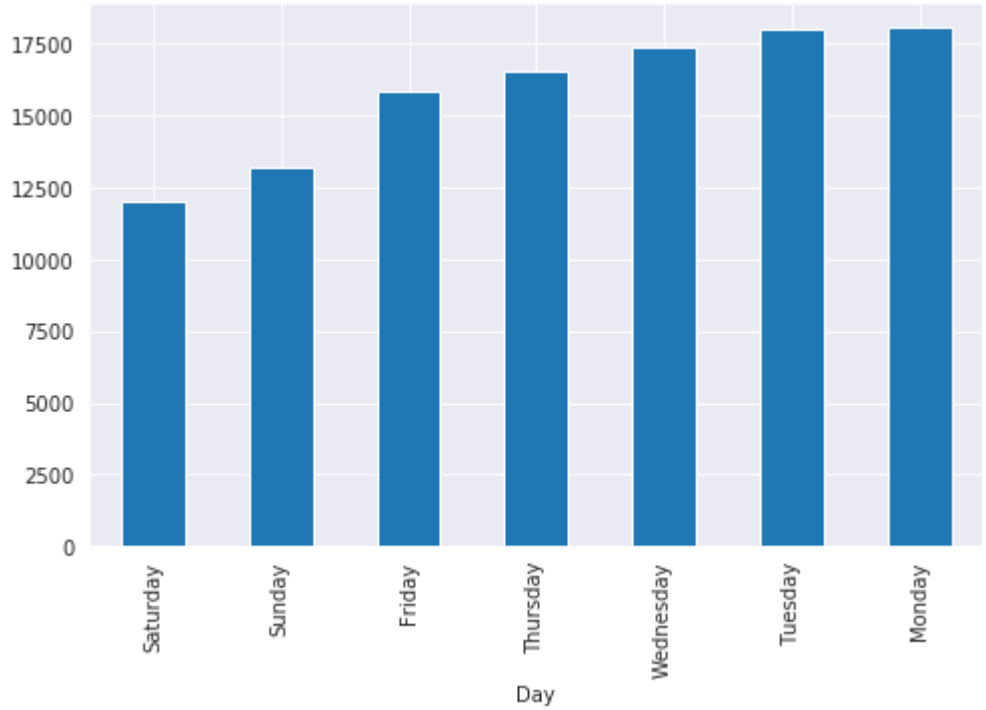
In [15]:
```
Month_Year = df.groupby('MonthY').size().sort_values()

fig=plt.figure(figsize=(10,8))
sns.set_style("darkgrid")
sns.barplot(y=Month_Year.index, x=Month_Year.values)
plt.title('Month_Year',fontsize=20)
```

Out[15]:
```
Text(0.5, 1.0, 'Month_Year')
```



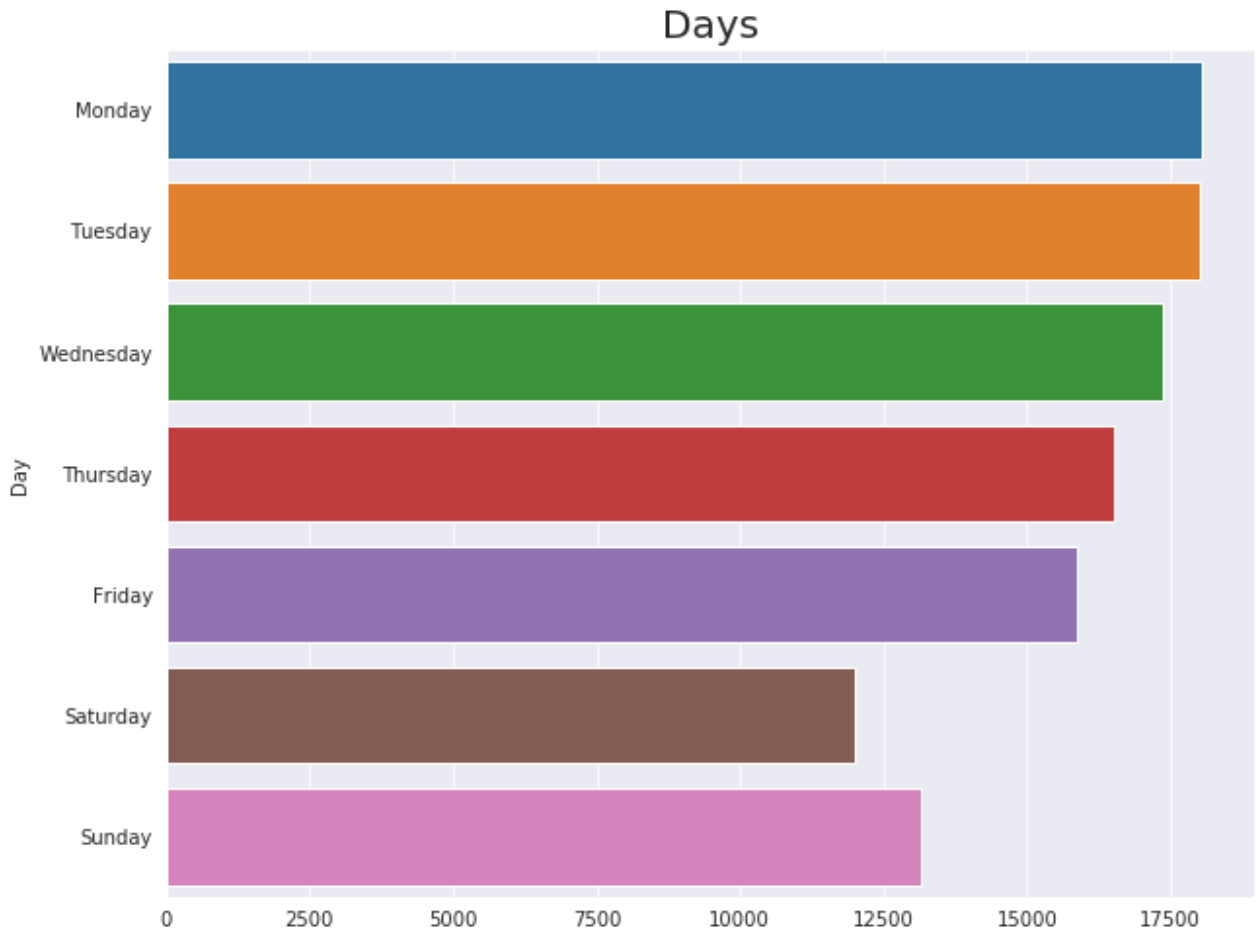
In [16]:
```
df['Day'] = pd.Series(pd.Categorical(df['order_purchase_timestamp'].dt.day_name(), cate
df.groupby('Day').size().sort_values().plot(
    kind = 'bar',
    figsize=(8,5)
)
```

Out[16]: <matplotlib.axes._subplots.AxesSubplot at 0x7f59ccf17a90>



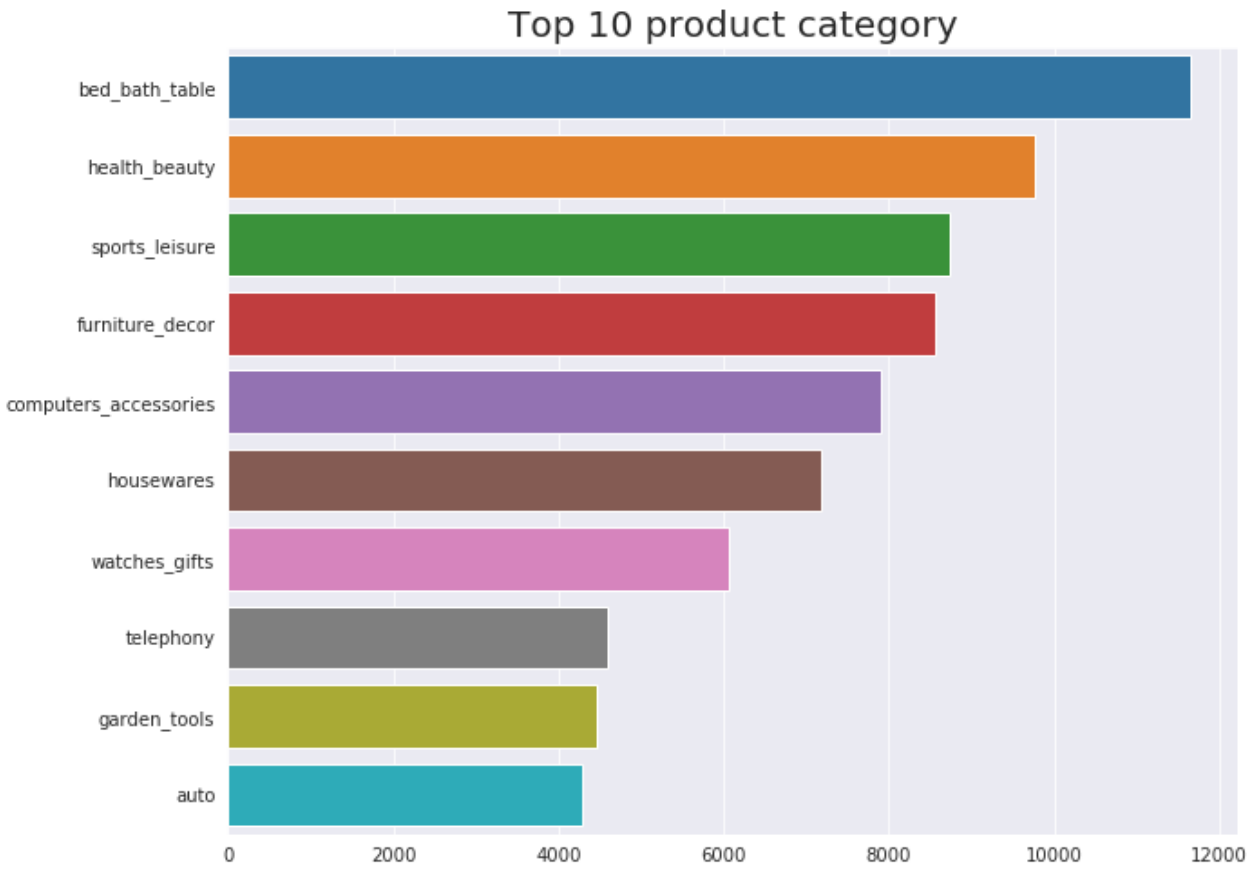
In [18]:
```
day = df.groupby('Day').size().sort_values()

fig=plt.figure(figsize=(10,8))
sns.set_style("darkgrid")
sns.barplot(y=day.index, x=day.values)
plt.title('Days',fontsize=20)
```

Out[18]:
```
Text(0.5, 1.0, 'Days')
```



In [19]:
```
top_10_category = df["product_category_name_english"].value_counts().sort_values(ascend

fig=plt.figure(figsize=(10,8))
sns.set_style("darkgrid")
sns.barplot(y=top_10_category.index, x=top_10_category.values)
plt.title('Top 10 product category',fontsize=20)
```

Out[19]:
```
Text(0.5, 1.0, 'Top 10 product category')
```

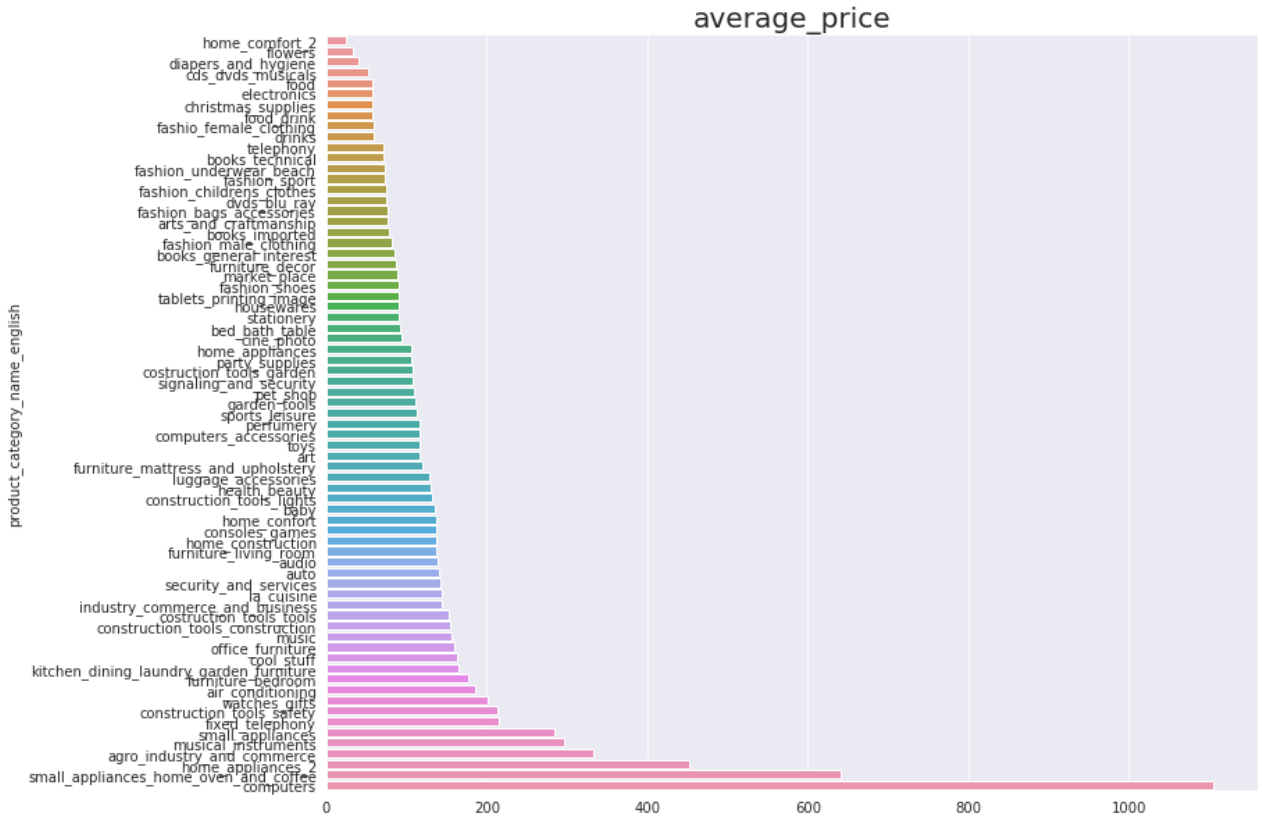


In [20]:
```
top_10_category
```

Out[20]:
```
bed_bath_table           11649
health_beauty             9761
sports_leisure            8731
furniture_decor           8553
computers_accessories     7897
housewares                7172
watches_gifts             6063
telephony                 4601
garden_tools              4463
auto                      4283
Name: product_category_name_english, dtype: int64
```

In [21]:
```
average_price = df.groupby("product_category_name_english")["price"].agg(np.mean).sort_
average_price
fig=plt.figure(figsize=(12,10))
sns.set_style("darkgrid")
sns.barplot(y=average_price.index, x=average_price.values)
plt.title('average_price',fontsize=20)
```

Out[21]:
```
Text(0.5, 1.0, 'average_price')
```



In [22]:
```
average_price.sample(10)
```

Out[22]:
```
product_category_name_english
pet_shop                          109.881561
furniture_decor                    87.191488
signaling_and_security            107.490603
construction_tools_lights         131.019032
la_cuisine                        143.998750
furniture_mattress_and_upholstery 119.779500
home_comfort_2                     24.940968
security_and_services             141.645000
diapers_and_hygiene                40.561892
construction_tools_construction   153.950714
Name: price, dtype: float64
```

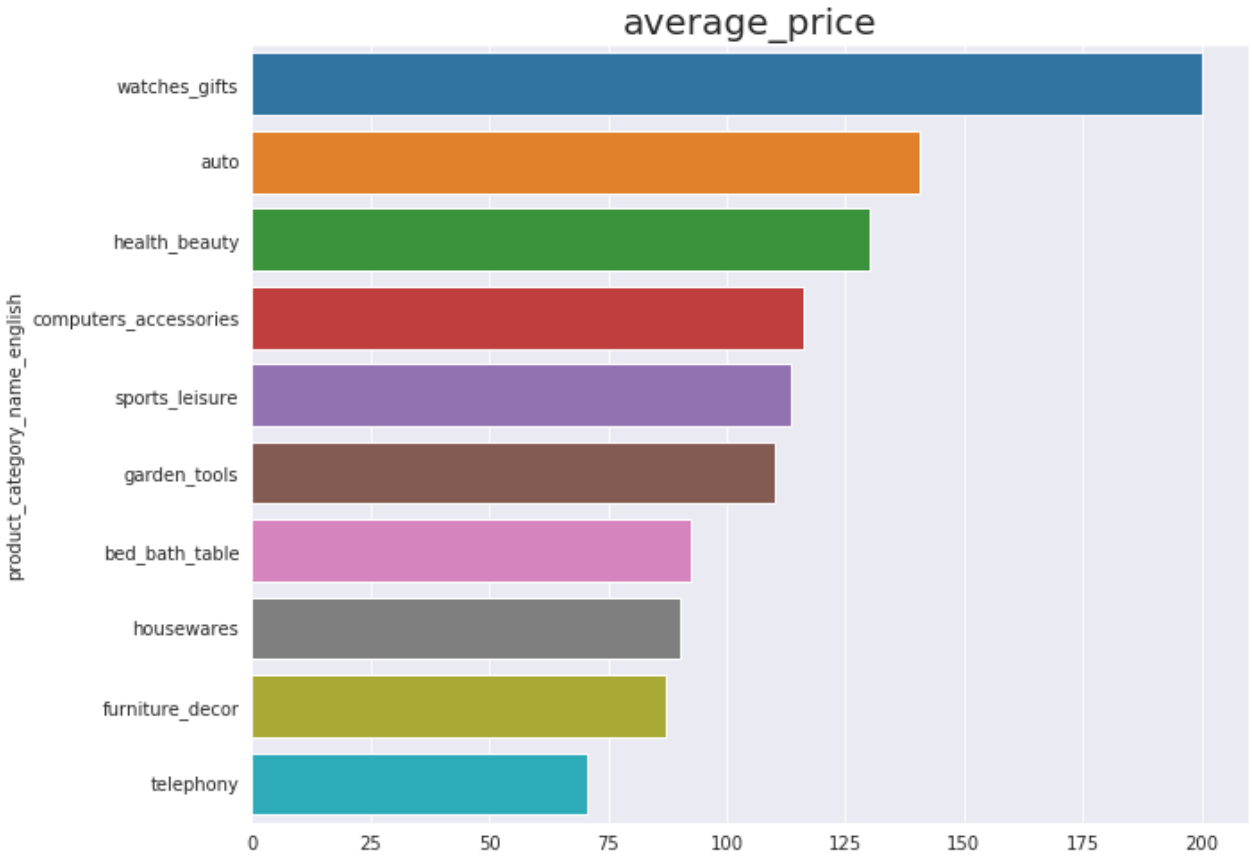
In [23]:
```
top_category = df[df['product_category_name_english'].isin(top_10_category.index)]
price_top_category=round(top_category.groupby("product_category_name_english")["price"]
price_top_category
```

Out[23]:
```
product_category_name_english
watches_gifts            200.09
auto                     140.76
health_beauty            130.25
computers_accessories    116.35
sports_leisure           113.44
garden_tools             110.30
bed_bath_table            92.53
housewares                90.38
furniture_decor           87.19
telephony                 70.73
Name: price, dtype: float64
```

In [24]:
```
fig=plt.figure(figsize=(10,8))
```

```
sns.set_style("darkgrid")
sns.barplot(y=price_top_category.index, x=price_top_category.values)
plt.title('average_price',fontsize=20)
```

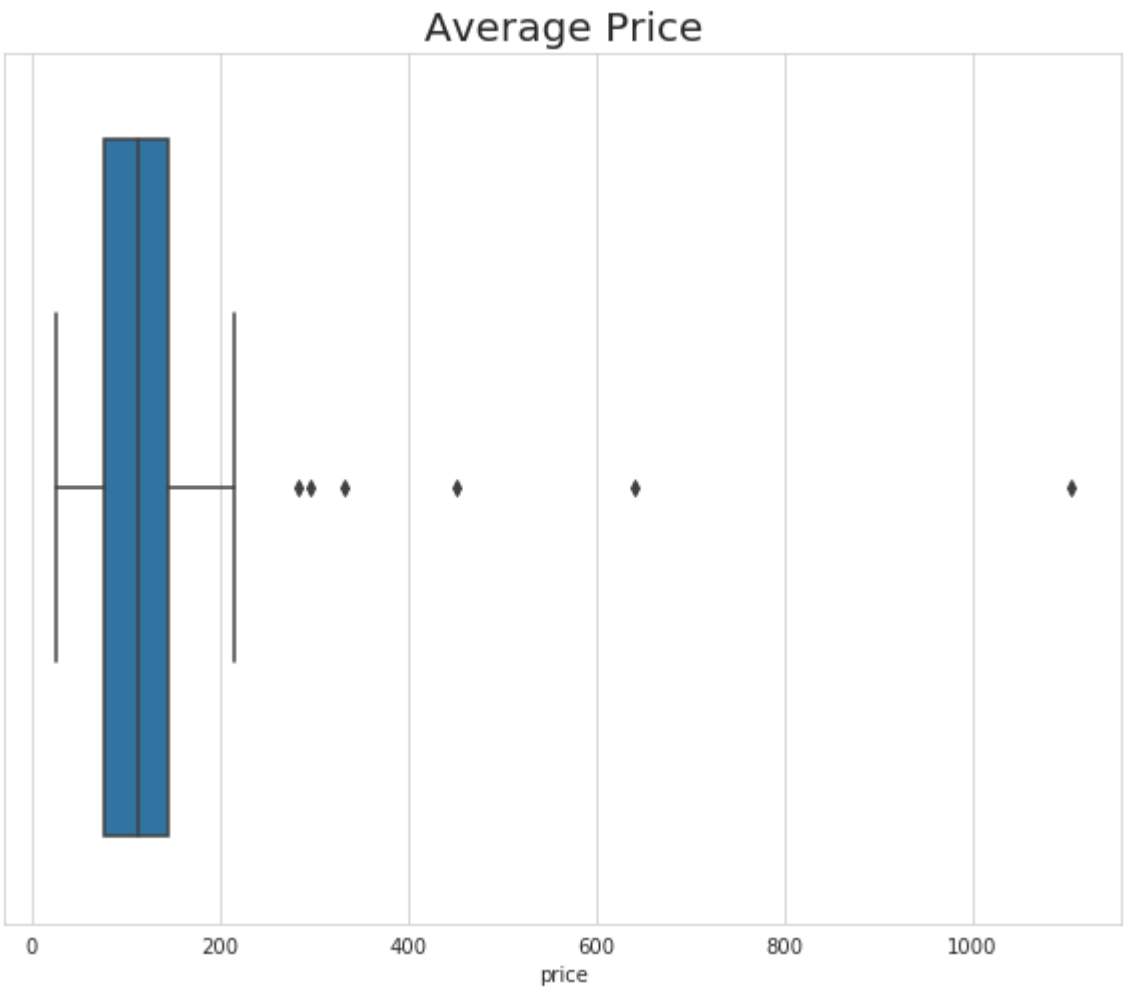
Out[24]: Text(0.5, 1.0, 'average_price')



In [25]:

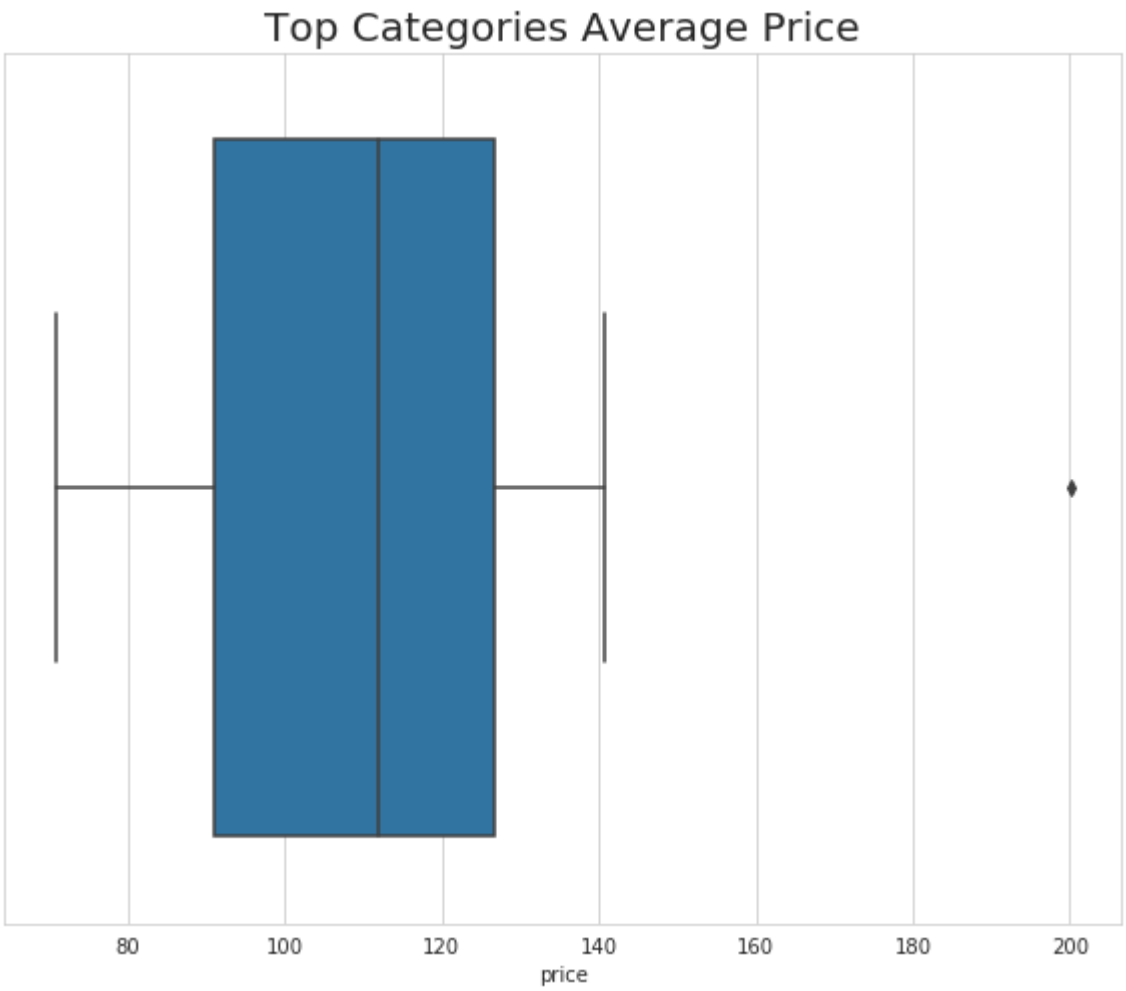
```
fig=plt.figure(figsize=(10,8))
sns.set_style("whitegrid")
sns.boxplot(x=average_price)
plt.title('Average Price',fontsize=20)
```

Out[25]: Text(0.5, 1.0, 'Average Price')



In [26]:
```
fig=plt.figure(figsize=(10,8))
sns.set_style("whitegrid")
sns.boxplot(x=price_top_category)
plt.title('Top Categories Average Price',fontsize=20)
```

Out[26]: Text(0.5, 1.0, 'Top Categories Average Price')



In [27]:
```
df.payment_type.sample(15)
```

Out[27]:
```
56316          boleto
85480     credit_card
80215     credit_card
88162     credit_card
26758     credit_card
33426     credit_card
6291      credit_card
108413         boleto
61546          boleto
24822          boleto
52621          boleto
10823     credit_card
22972          boleto
108786    credit_card
70369     credit_card
Name: payment_type, dtype: object
```

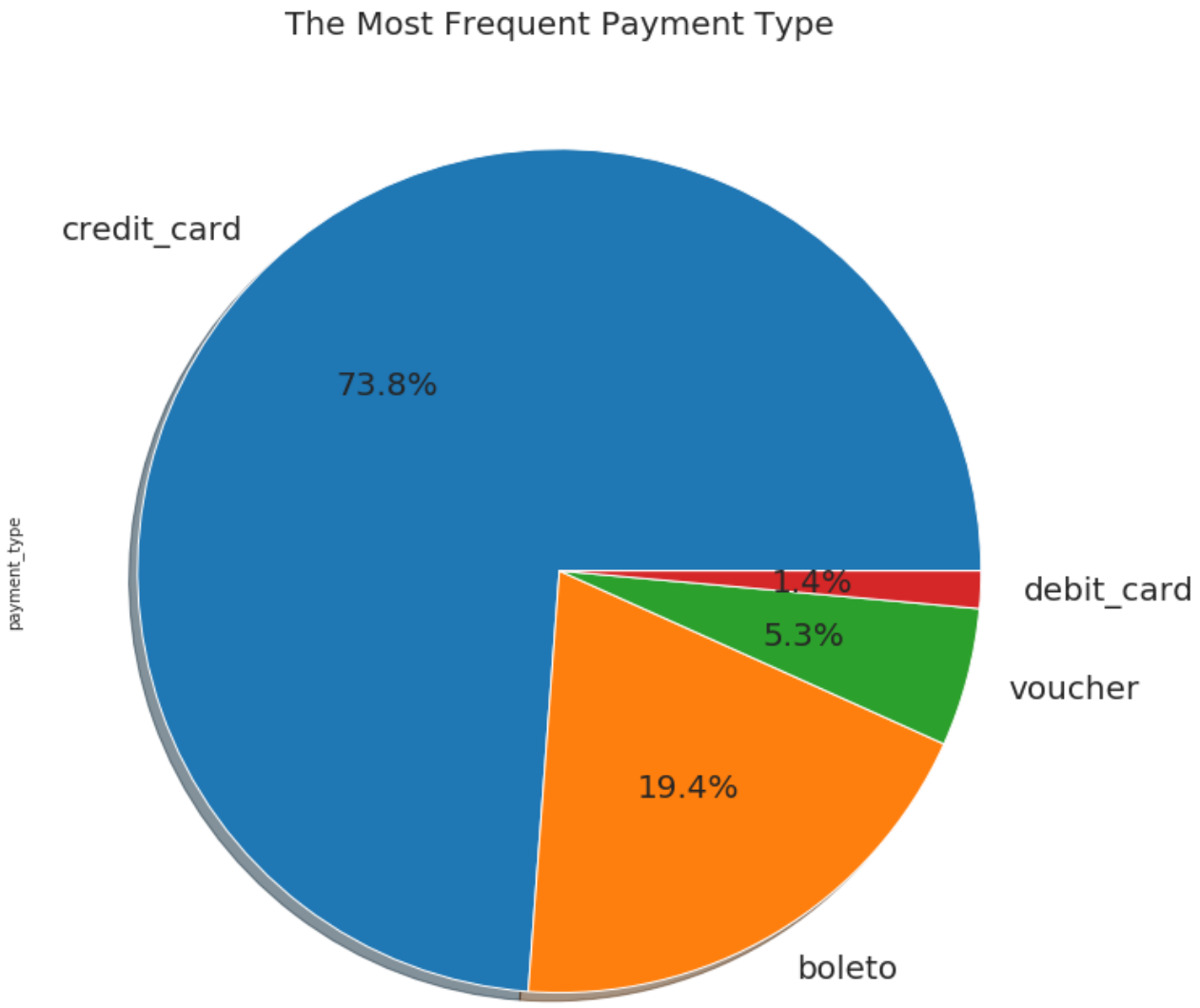
In [28]:
```
df.payment_type.nunique()
```

Out[28]:
```
4
```

In [29]:
```
df.payment_type.unique()
```

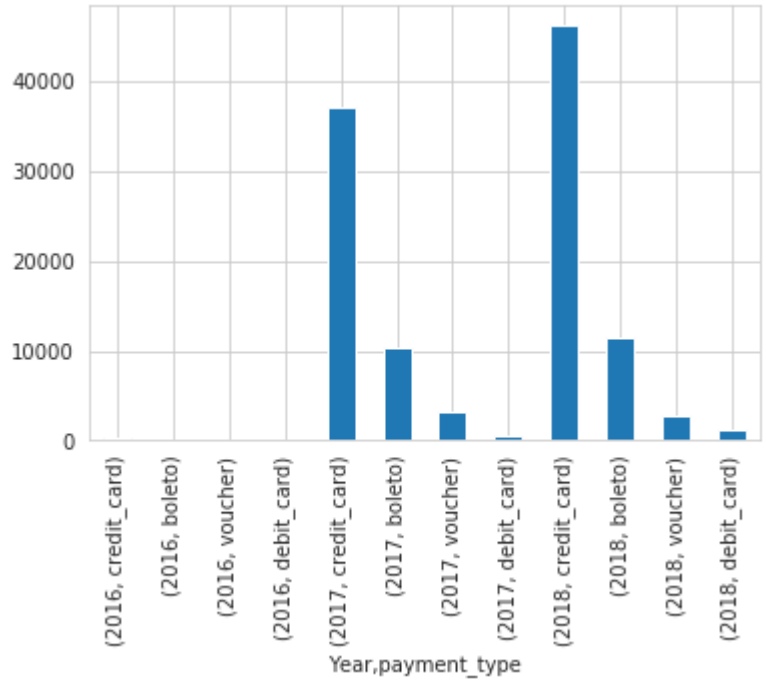
Out[29]:
```
array(['credit_card', 'voucher', 'boleto', 'debit_card'], dtype=object)
```

In [30]:
```
df["payment_type"].value_counts().plot(
    kind="pie",
    autopct="%1.1f%%",
    fontsize=20,
    figsize=(12,12),
    shadow=True

)
plt.title('The Most Frequent Payment Type'.title() , fontsize=20);
```



In [31]:
```
df.groupby("Year")['payment_type'].value_counts().plot(kind='bar')
```

Out[31]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f59ca8d5310>
```



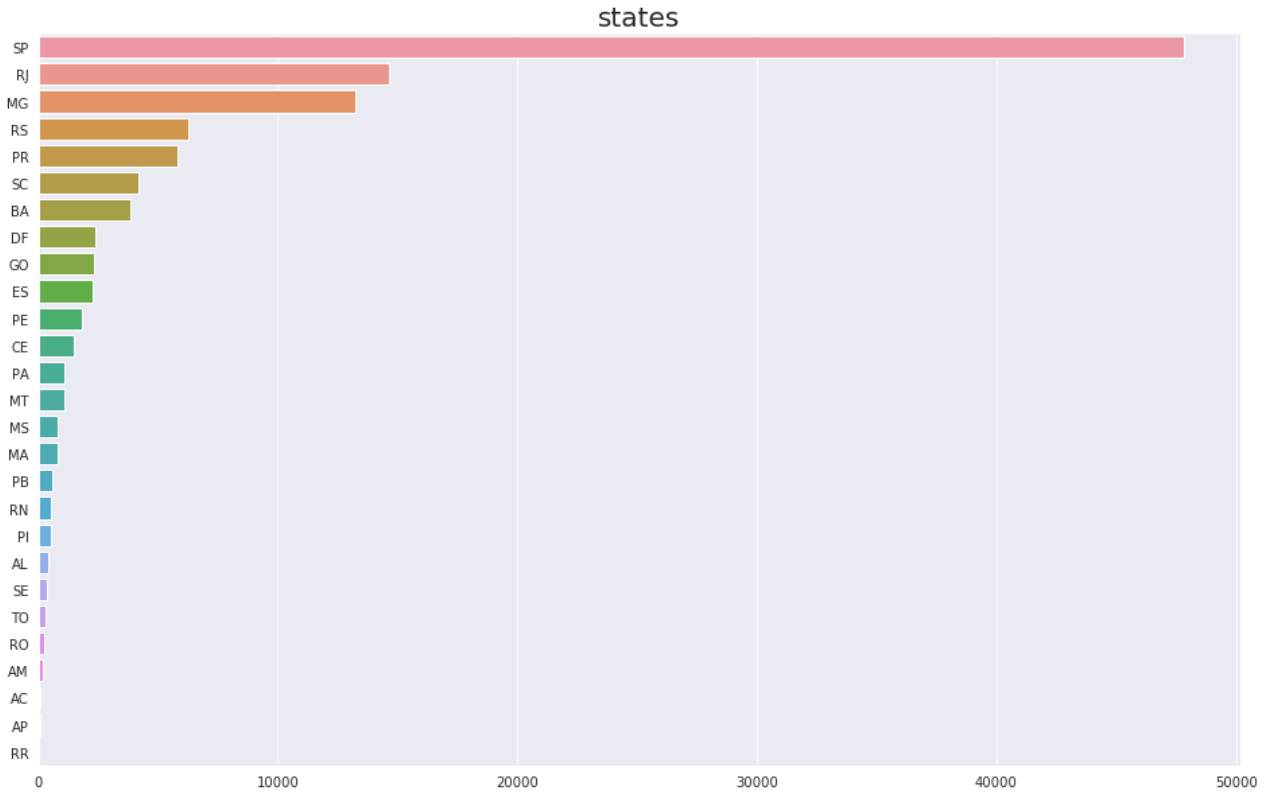
In [32]:
```
top_states = df["customer_state"].value_counts()
top_states
```

Out[32]:
```
SP    47819
RJ    14648
MG    13230
RS     6282
PR     5790
SC     4161
BA     3858
DF     2389
GO     2319
ES     2288
PE     1803
CE     1482
PA     1070
MT     1067
MS      826
MA      808
PB      614
RN      555
PI      546
AL      437
SE      384
TO      330
RO      275
AM      167
AC       92
AP       83
RR       44
Name: customer_state, dtype: int64
```

In [33]:
```
fig=plt.figure(figsize=(16,10))
sns.set_style("darkgrid")
sns.barplot(y=top_states.index, x=top_states.values)
plt.title('states',fontsize=20)
```

```
Text(0.5, 1.0, 'states')
```

Out[33]:



In [ ]:

In [ ]:

In [ ]:

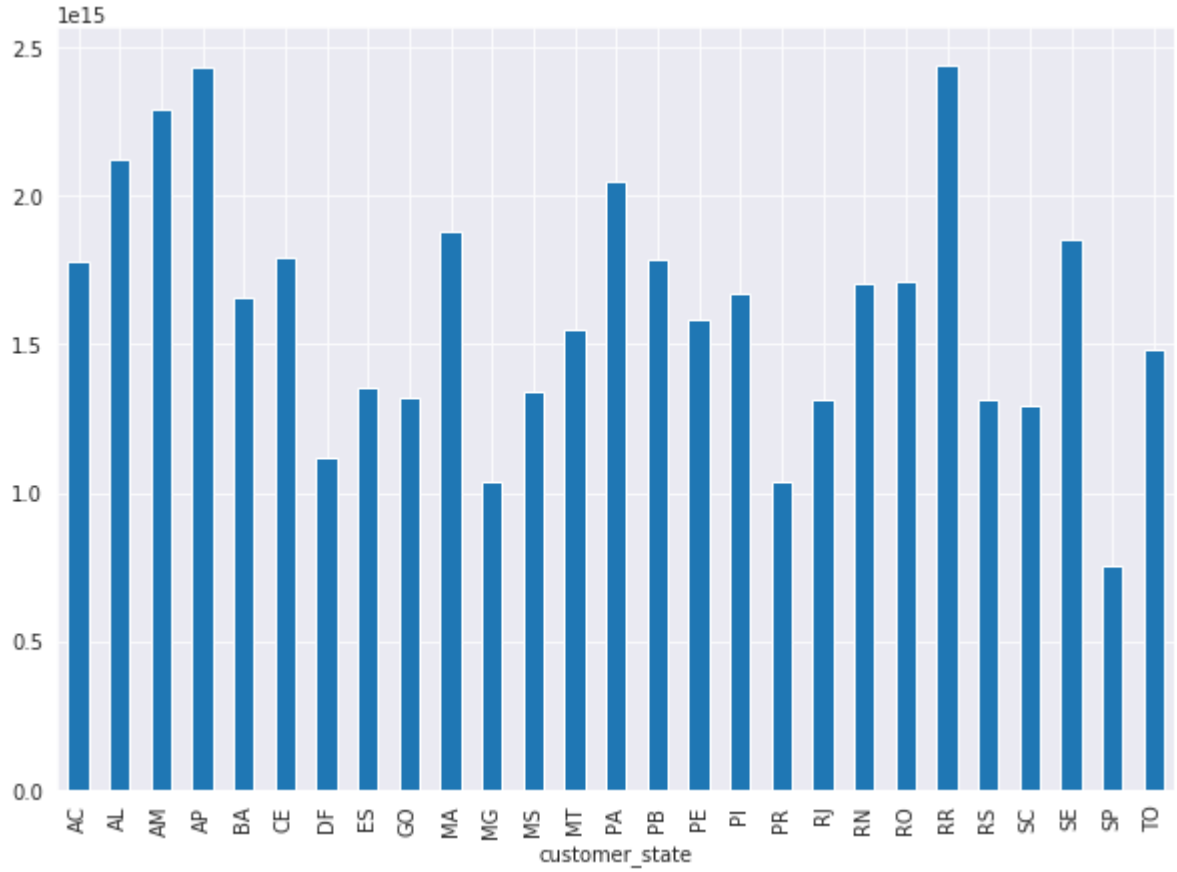
In [ ]:

In [ ]:

In [ ]:

In [34]:
```
df['derivery_time'] = df['order_delivered_customer_date'].astype(int)- df['order_purcha

delivery_per_state = df.groupby("customer_state")['derivery_time'].mean()
delivery_per_state.plot(
    kind='bar',
    figsize=(10,7)
)
```

Out[34]: <matplotlib.axes._subplots.AxesSubplot at 0x7f59ca6ed290>



In [35]:

```
def adding_colour(val):
  if val < 0:
    color = 'red'
  elif val < 1:
    color = 'green'
  else:
    color = 'black'
  return 'color: %s' % color


corr = df.corr().style.applymap(adding_colour) #To apply a function to a data frame we
corr
```

Out[35]:

	payment_sequential	payment_installments	payment_value	customer_zip_code
payment_sequential	1.000000	-0.087636	-0.065092	-0
payment_installments	-0.087636	1.000000	0.274281	0
payment_value	-0.065092	0.274281	1.000000	0
customer_zip_code_prefix	-0.028489	0.057973	0.053122	1
order_item_id	-0.000320	0.074173	0.266621	0
price	0.000263	0.279455	0.736578	0
freight_value	0.008459	0.186467	0.372554	0
product_name_lenght	-0.001524	0.020894	0.004857	0
product_description_lenght	-0.010569	0.036633	0.157190	0

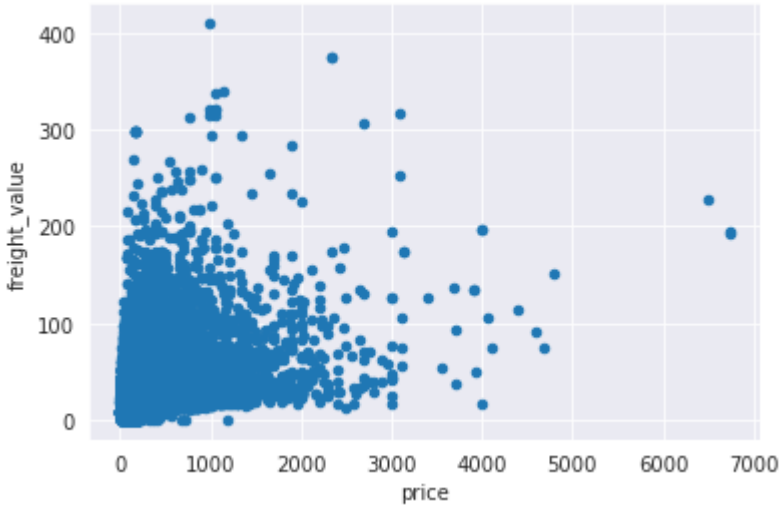
	payment_sequential	payment_installments	payment_value	customer_zip_code
product_photos_qty	-0.008694	-0.000140	0.010524	0
product_weight_g	0.026018	0.179123	0.305568	0
product_length_cm	0.030589	0.116260	0.138097	0
product_height_cm	0.020725	0.120540	0.216583	0
product_width_cm	0.030398	0.136901	0.148412	-0
Year	-0.043490	-0.050385	0.005173	-0
derivery_time	0.003093	0.044075	0.059970	0

In [36]:
```
round(df['freight_value'].corr(df['price']), 2)
```

Out[36]: 0.41

In [37]:
```
df.plot(
    kind = 'scatter',
    x = 'price',
    y = 'freight_value',
)
```

Out[37]: <matplotlib.axes._subplots.AxesSubplot at 0x7f59ca6ed1d0>

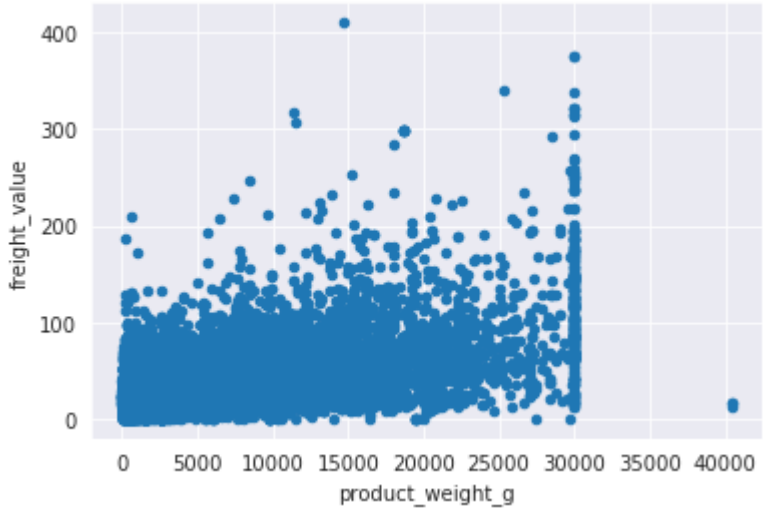


In [38]:
```
round(df['freight_value'].corr(df['product_weight_g']), 2)
```

Out[38]: 0.61

In [39]:
```
df.plot(
    kind = 'scatter',
    x = 'product_weight_g',
    y = 'freight_value',
)
```

Out[39]: <matplotlib.axes._subplots.AxesSubplot at 0x7f59be5a4150>



In [40]:
```
def add_colour(val):
  if val < 0:
    color = 'red'
  else:
    color = 'green'
  return 'color: %s' % color

cov = df.cov().style.applymap(add_colour)
cov
```

Out[40]:

	payment_sequential	payment_installments	payment_value
payment_sequential	0.465288	-0.166055	-11.836070
payment_installments	-0.166055	7.716428	203.107374
payment_value	-11.836070	203.107374	71062.828028
customer_zip_code_prefix	-579.985028	4806.354585	422641.896380
order_item_id	-0.000153	0.144501	49.846437
price	0.032788	141.838425	35876.844129
freight_value	0.090900	8.159839	1564.523976
product_name_lenght	-0.010417	0.581478	12.970756
product_description_lenght	-4.693542	66.248812	27279.739213
product_photos_qty	-0.010203	-0.000669	4.826012
product_weight_g	66.917248	1876.142636	307138.022598
product_length_cm	0.337217	5.219472	594.968179
product_height_cm	0.190121	4.503135	776.466157
product_width_cm	0.243454	4.465100	464.521425
Year	-0.014954	-0.070553	0.695170
derivery_time	1721521023894.498291	99911973611246.218750	13045809455290608.000000

In [41]:
```
round(df['freight_value'].cov(df['price']), 2)
```

Out[41]: 1194.35

In [42]:
```
round(df['freight_value'].cov(df['product_weight_g']), 2)
```

Out[42]: 36383.97

In [ ]:
```

```