In [1]:

```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns
```

In [2]:

```python
retail_sales = pd.read_csv('retail_sales.csv', parse_dates = True, header = 0, index_col = 0)
retail_sales.head()
```

Out[2]:

	Date	Customer ID	Gender	Age	Product Category	Quantity	Price per Unit	Total Amount
Transaction ID								
1	2023-11-24	CUST001	Male	34	Beauty	3	50	150
2	2023-02-27	CUST002	Female	26	Clothing	2	500	1000
3	2023-01-13	CUST003	Male	50	Electronics	1	30	30
4	2023-05-21	CUST004	Male	37	Clothing	1	500	500
5	2023-05-06	CUST005	Male	30	Beauty	2	50	100

Clean the dataset

In [3]:

```python
#check the columns type
retail_sales.dtypes
```

Out[3]:

```
Date                object
Customer ID         object
Gender              object
Age                  int64
Product Category    object
Quantity             int64
Price per Unit       int64
Total Amount         int64
dtype: object
```

In [4]:

```python
#convert the date type from object to datetime
import datetime
retail_sales.Date = pd.to_datetime(retail_sales.Date)
retail_sales.dtypes
```

Out[4]:

```
Date                datetime64[ns]
Customer ID                 object
Gender                      object
Age                          int64
Product Category            object
Quantity                     int64
Price per Unit               int64
Total Amount                 int64
dtype: object
```

In [5]:

```python
#drop the transaction id index
retail_sales.reset_index(drop = True, inplace = True)
retail_sales.head()
```

Out[5]:

	Date	Customer ID	Gender	Age	Product Category	Quantity	Price per Unit	Total Amount
0	2023-11-24	CUST001	Male	34	Beauty	3	50	150
1	2023-02-27	CUST002	Female	26	Clothing	2	500	1000
2	2023-01-13	CUST003	Male	50	Electronics	1	30	30
3	2023-05-21	CUST004	Male	37	Clothing	1	500	500
4	2023-05-06	CUST005	Male	30	Beauty	2	50	100

In [6]:

```python
#check for empty cell
retail_sales.isna().sum()
```

```
Out[6]: Date                0
        Customer ID         0
        Gender              0
        Age                 0
        Product Category    0
        Quantity            0
        Price per Unit      0
        Total Amount        0
        dtype: int64
```

```
In [7]: #set the age range
        gen_z = retail_sales[retail_sales.Age.between(18, 35)]
        millenials = retail_sales[retail_sales.Age.between(36, 50)]
        gen_x = retail_sales[retail_sales.Age > 50]

        gen_z.shape, millenials.shape, gen_x.shape
```

```
Out[7]: ((374, 8), (313, 8), (313, 8))
```

```
In [8]: millenials.Gender.value_counts()
```

```
Out[8]: Female    164
        Male      149
        Name: Gender, dtype: int64
```

Visualize the age group and their purchase

```
In [9]: gen_z['Product Category'].shape
```

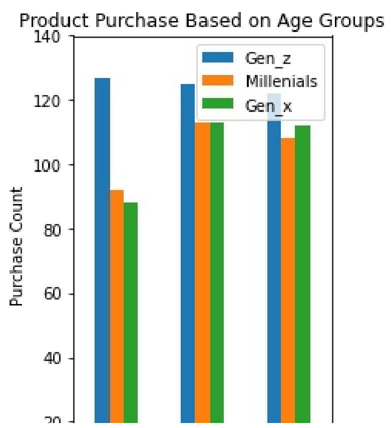
```
Out[9]: (374,)
```

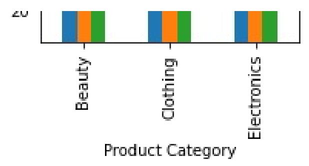
```
In [10]: # categorize the age age groups based on their product purchase
         def agegroup():
             '''
             take a function that returns the age
             insert into a dataframe
             '''

             gez = gen_z['Product Category'].value_counts()
             mil = millenials['Product Category'].value_counts()
             gex = gen_x['Product Category'].value_counts()

             #write a dataframe
             ages = pd.DataFrame({'Gen_z': gez,
                                  'Millenials': mil,
                                  'Gen_x': gex})
             return ages
```

```
In [13]: #plot the data
         agegroup().plot(figsize = (3,5), kind = 'bar', ylim = (10, 140), xlabel = 'Product Category',
                         ylabel = 'Purchase Count', title = 'Product Purchase Based on Age Groups');
```





In [14]:
```
agegroup()
```

Out[14]:

	Gen_z	Millenials	Gen_x
Beauty	127	92	88
Clothing	125	113	113
Electronics	122	108	112

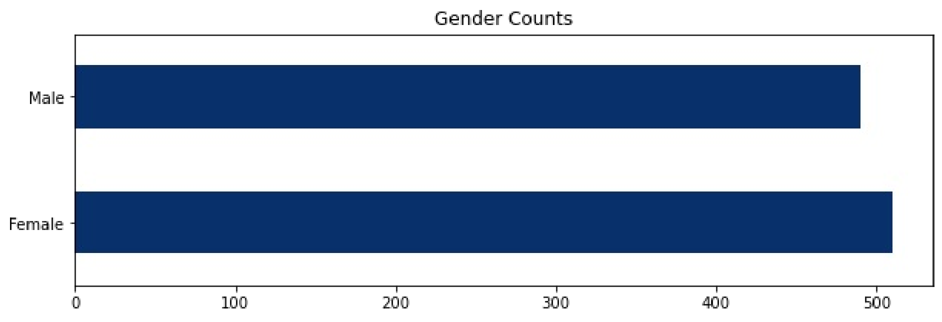
- **Gen_z** purchased the highest product for all the product categories
- **Millenials** purchased more beauty product than electronics and has a tie in the clothing category with gen_x
- **Gen_x** purchase more clothing and electronics than beauty products

Gender counts

In [15]:
```
gen = retail_sales.Gender.value_counts()
gen
```

Out[15]:
```
Female    510
Male      490
Name: Gender, dtype: int64
```

In [17]:
```
gen.plot(kind = 'barh', ylim = (100, 550),
           figsize = (10, 3), colormap = 'Blues_r', title = 'Gender Counts');
```



The company have more female customers

Visualize the product category based on gender purchases

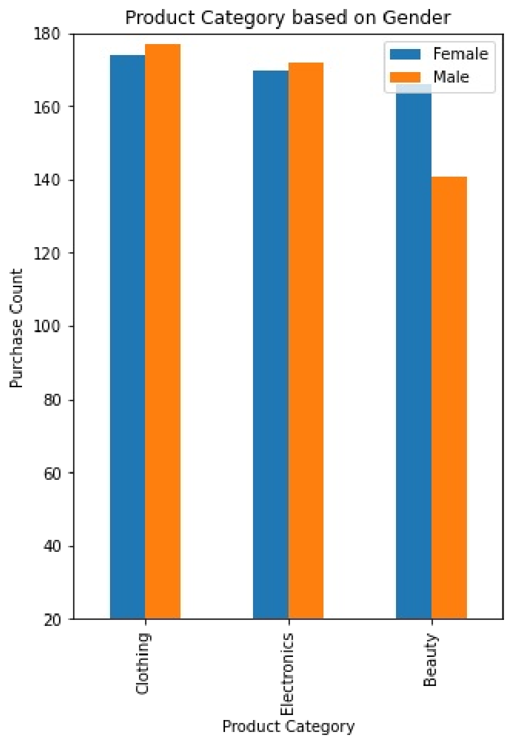
In [18]:
```
# set a target column to show female as 1 and male as 0
def assign_target(retail_sales):
    if retail_sales.Gender == 'Female':
        return 1
    else:
        return 0

retail_sales['Target'] = retail_sales.apply(assign_target, axis = 1)

#set a function that plots the product category by gender
female = retail_sales[retail_sales.Target== 1]
male = retail_sales[retail_sales.Target==0]

def gender():
    '''
    this shows the product purchased based on gender
    '''
    Female = female['Product Category'].value_counts()
    Male = male['Product Category'].value_counts()
    gen = pd.DataFrame({'Female': Female,
                        'Male': Male})
    return gen
```

In [19]:
```
gender().plot(figsize =(5,7), kind = 'bar', xlabel = "Product Category",
             ylabel = 'Purchase Count', title = 'Product Category based on Gender',
             ylim = (20,180));
```



In [20]:
```
gender()
```

Out[20]:

	Female	Male
Clothing	174	177
Electronics	170	172
Beauty	166	141

- **Female** customers bought more products than the male customers
- **Males** customers purchased more Clothing and Electronics than female customers

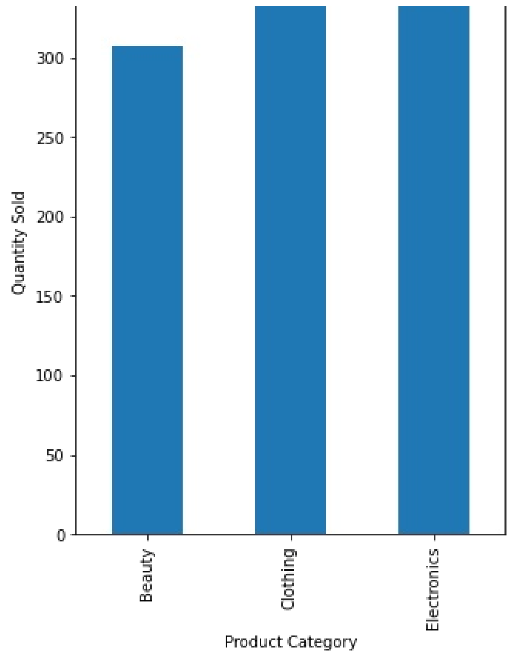
In [21]:
```
retail_sales.head()
```

Out[21]:

	Date	Customer ID	Gender	Age	Product Category	Quantity	Price per Unit	Total Amount	Target
0	2023-11-24	CUST001	Male	34	Beauty	3	50	150	0
1	2023-02-27	CUST002	Female	26	Clothing	2	500	1000	1
2	2023-01-13	CUST003	Male	50	Electronics	1	30	30	0
3	2023-05-21	CUST004	Male	37	Clothing	1	500	500	0
4	2023-05-06	CUST005	Male	30	Beauty	2	50	100	0

Quantity sold based on product category

In [22]:
```
qty_sold = retail_sales[['Product Category', 'Quantity']].groupby('Product Category')['Quantity'].count()
qty_sold.plot(kind =  'bar', figsize = (5,7), title = 'Quantity Sold based on Product Category',
             ylabel = 'Quantity Sold');
```



In [23]:

```
qty_sold
```

Out[23]:

```
Product Category
Beauty         307
Clothing       351
Electronics    342
Name: Quantity, dtype: int64
```

The company sold more clothing than other products

In [24]:

```
retail_sales.head()
```

Out[24]:

	Date	Customer ID	Gender	Age	Product Category	Quantity	Price per Unit	Total Amount	Target
0	2023-11-24	CUST001	Male	34	Beauty	3	50	150	0
1	2023-02-27	CUST002	Female	26	Clothing	2	500	1000	1
2	2023-01-13	CUST003	Male	50	Electronics	1	30	30	0
3	2023-05-21	CUST004	Male	37	Clothing	1	500	500	0
4	2023-05-06	CUST005	Male	30	Beauty	2	50	100	0

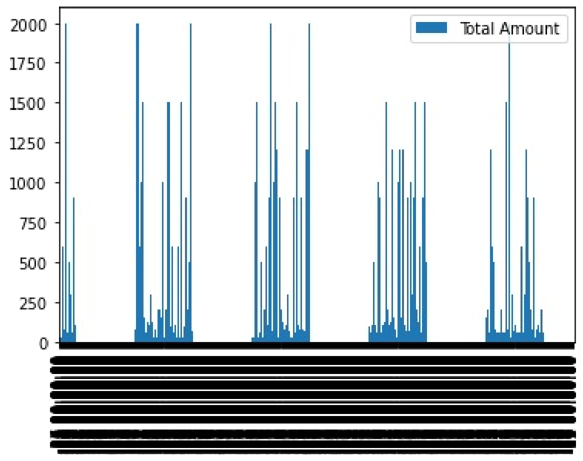
Sales by period

In [25]:

```
# check the year with the highest sales
retail_sales.plot(kind = 'bar', x = 'Date',
                  y = 'Total Amount')
```

Out[25]:

```
<AxesSubplot:xlabel='Date'>
```



the period could not be determined as they are clustered

In [26]:
```python
jan_2023 = retail_sales[retail_sales.Date.between('2023-01-01', '2023-01-31')]
feb_2023 = retail_sales[retail_sales.Date.between('2023-02-01', '2023-02-28')]
mar_2023 = retail_sales[retail_sales.Date.between('2023-03-01', '2023-03-31')]
apr_2023 = retail_sales[retail_sales.Date.between('2023-04-01', '2023-04-30')]
may_2023 = retail_sales[retail_sales.Date.between('2023-05-01', '2023-05-31')]
jun_2023 = retail_sales[retail_sales.Date.between('2023-06-01', '2023-06-30')]
jul_2023 = retail_sales[retail_sales.Date.between('2023-07-01', '2023-07-31')]
aug_2023 = retail_sales[retail_sales.Date.between('2023-08-01', '2023-08-31')]
sep_2023 = retail_sales[retail_sales.Date.between('2023-09-01', '2023-09-30')]
oct_2023 = retail_sales[retail_sales.Date.between('2023-10-01', '2023-10-31')]
nov_2023 = retail_sales[retail_sales.Date.between('2023-11-01', '2023-11-30')]
dec_2023 = retail_sales[retail_sales.Date.between('2023-12-01', '2023-12-31')]
```

In [27]:
```python
def daterange():
    '''
    insert the product category to it and show sales based on period
    '''
    jan_sales = jan_2023['Product Category'].value_counts()
    feb_sales = feb_2023['Product Category'].value_counts()
    mar_sales = mar_2023['Product Category'].value_counts()
    apr_sales = apr_2023['Product Category'].value_counts()
    may_sales = may_2023['Product Category'].value_counts()
    jun_sales = jun_2023['Product Category'].value_counts()
    jul_sales = jul_2023['Product Category'].value_counts()
    aug_sales = aug_2023['Product Category'].value_counts()
    sep_sales = sep_2023['Product Category'].value_counts()
    oct_sales = oct_2023['Product Category'].value_counts()
    nov_sales = nov_2023['Product Category'].value_counts()
    dec_sales = dec_2023['Product Category'].value_counts()

    #return a dataframe
    dates = pd.DataFrame({'January': jan_sales,
                          'February': feb_sales,
                          'March': mar_sales,
                          'April': apr_sales,
                          'May': may_sales,
                          'June': jun_sales,
                          'July': jul_sales,
                          'August': aug_sales,
                          'September': sep_sales,
                          'October': oct_sales,
                          'November': nov_sales,
                          'December': dec_sales})
    return dates
```

In [28]:
```python
daterange()
```

Out[28]:

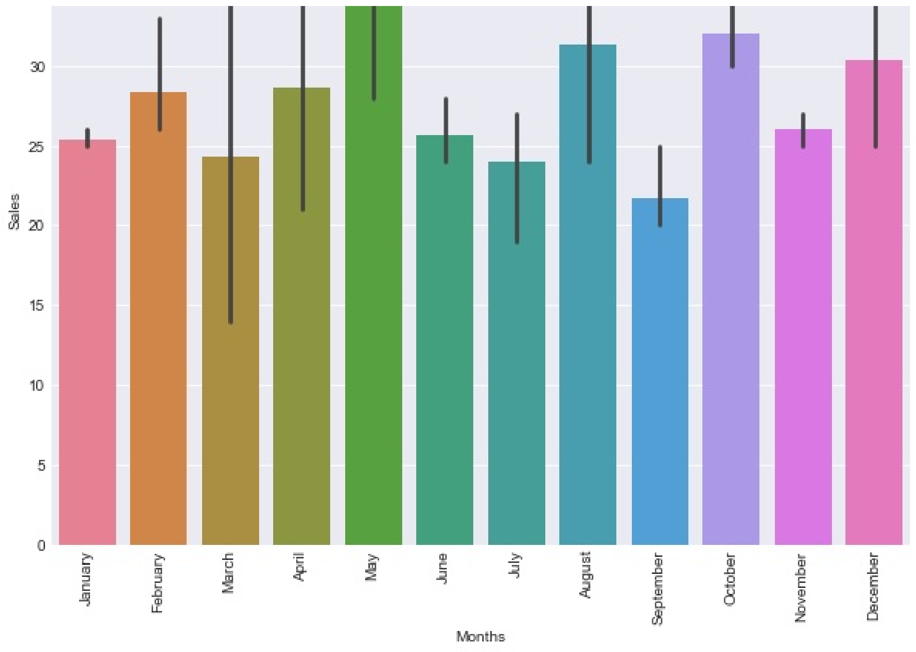
	January	February	March	April	May	June	July	August	September	October	November	December
Beauty	25	26	21	29	28	25	27	24	20	31	25	25
Clothing	26	33	38	36	37	28	19	32	20	30	26	26
Electronics	25	26	14	21	40	24	26	38	25	35	27	40

In [29]:
```python
plt.style.use('seaborn-darkgrid')

fig, ax = plt.subplots(figsize = (10, 8))
sns.barplot(data = daterange(), palette = 'husl')
sns.set_context('poster')
plt.setp(ax.get_xticklabels(), fontsize = 10, rotation = 'vertical')

ax.set(xlabel = 'Months', ylabel = 'Sales',
      title = 'Sales by Month');
```

Sales by Month
40
35



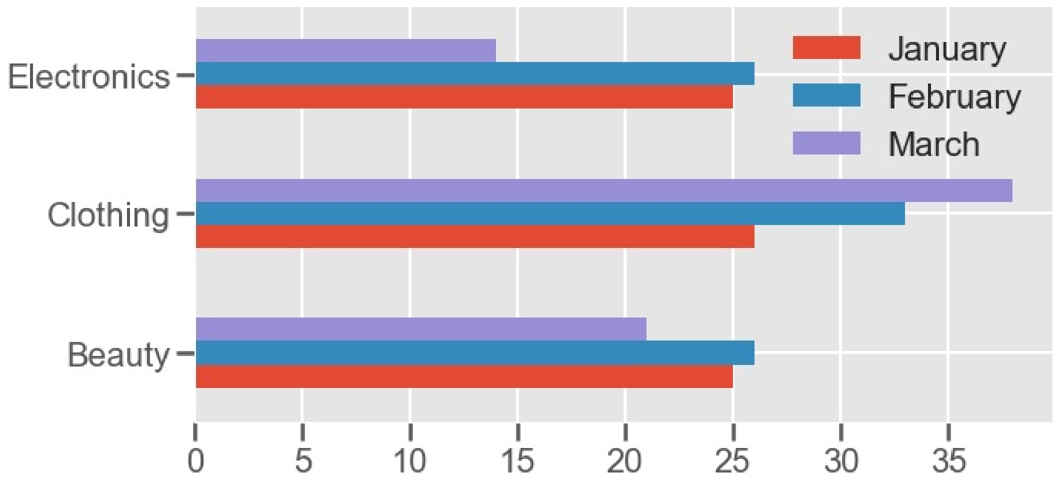
The highest sales was made in May and December

In [30]:
```
daterange()
```

Out[30]:

	January	February	March	April	May	June	July	August	September	October	November	December
Beauty	25	26	21	29	28	25	27	24	20	31	25	25
Clothing	26	33	38	36	37	28	19	32	20	30	26	26
Electronics	25	26	14	21	40	24	26	38	25	35	27	40

In [31]:
```
plt.style.use('ggplot')
daterange()[['January', 'February', 'March']].plot(kind = 'barh', figsize = (10,5));
```



Loading [MathJax]/jax/output/CommonHTML/fonts/TeX/fontdata.js