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

In [2]:
```
data = pd.read_csv('D:\documents\weatherHistory.csv' , parse_dates = ['Formatted Date'] , index_col = ['Formatted Date'])
```

In [3]:
```
data.head()
```

Out[3]:

Formatted Date	Summary	Precip Type	Temperature (C)	Apparent Temperature (C)	Humidity	Wind Speed (km/h)	Wind Bearing (degrees)	Visibility (km)	Loud Cover	Pressure (millibars)	Daily Summary
2006-04-01 00:00:00+02:00	Partly Cloudy	rain	9.472222	7.388889	0.89	14.1197	251.0	15.8263	0.0	1015.13	Partly cloudy throughout the day.
2006-04-01 01:00:00+02:00	Partly Cloudy	rain	9.355556	7.227778	0.86	14.2646	259.0	15.8263	0.0	1015.63	Partly cloudy throughout the day.
2006-04-01 02:00:00+02:00	Mostly Cloudy	rain	9.377778	9.377778	0.89	3.9284	204.0	14.9569	0.0	1015.94	Partly cloudy throughout the day.
2006-04-01 03:00:00+02:00	Partly Cloudy	rain	8.288889	5.944444	0.83	14.1036	269.0	15.8263	0.0	1016.41	Partly cloudy throughout the day.
2006-04-01 04:00:00+02:00	Mostly Cloudy	rain	8.755556	6.977778	0.83	11.0446	259.0	15.8263	0.0	1016.51	Partly cloudy throughout the day.

In [4]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 96453 entries, 2006-04-01 00:00:00+02:00 to 2016-09-09 23:00:00+02:00
Data columns (total 11 columns):
 #   Column                    Non-Null Count  Dtype  
---  ------                    --------------  -----  
 0   Summary                   96453 non-null  object 
 1   Precip Type               95936 non-null  object 
 2   Temperature (C)           96453 non-null  float64
 3   Apparent Temperature (C)  96453 non-null  float64
 4   Humidity                  96453 non-null  float64
 5   Wind Speed (km/h)         96453 non-null  float64
 6   Wind Bearing (degrees)    96453 non-null  float64
 7   Visibility (km)           96453 non-null  float64
```

```
 8   Loud Cover                96453 non-null  float64
 9   Pressure (millibars)      96453 non-null  float64
 10  Daily Summary             96453 non-null  object
dtypes: float64(8), object(3)
memory usage: 8.8+ MB
```

In [5]:
```
data.isnull().sum() # there are 517 null columns
```

Out[5]:
```
Summary                       0
Precip Type                 517
Temperature (C)               0
Apparent Temperature (C)      0
Humidity                      0
Wind Speed (km/h)             0
Wind Bearing (degrees)        0
Visibility (km)               0
Loud Cover                    0
Pressure (millibars)          0
Daily Summary                 0
dtype: int64
```

In [6]:
```
new_data = data.dropna() # remove null columns and store it in a new data set
```

In [7]:
```
new_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 95936 entries, 2006-04-01 00:00:00+02:00 to 2016-09-09 23:00:00+02:00
Data columns (total 11 columns):
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
 0   Summary                   95936 non-null  object
 1   Precip Type               95936 non-null  object
 2   Temperature (C)           95936 non-null  float64
 3   Apparent Temperature (C)  95936 non-null  float64
 4   Humidity                  95936 non-null  float64
 5   Wind Speed (km/h)         95936 non-null  float64
 6   Wind Bearing (degrees)    95936 non-null  float64
 7   Visibility (km)           95936 non-null  float64
 8   Loud Cover                95936 non-null  float64
 9   Pressure (millibars)      95936 non-null  float64
 10  Daily Summary             95936 non-null  object
dtypes: float64(8), object(3)
memory usage: 8.8+ MB
```

In [8]:
```
new_data.describe()
```

Out[8]:

	Temperature (C)	Apparent Temperature (C)	Humidity	Wind Speed (km/h)	Wind Bearing (degrees)	Visibility (km)	Loud Cover	Pressure (millibars)

	Temperature (C)	Apparent Temperature (C)	Humidity	Wind Speed (km/h)	Wind Bearing (degrees)	Visibility (km)	Loud Cover	Pressure (millibars)
count	95936.000000	95936.000000	95936.000000	95936.000000	95936.000000	95936.000000	95936.0	95936.000000
mean	11.940976	10.862531	0.734841	10.804936	187.518773	10.362402	0.0	1003.150038
std	9.570671	10.717812	0.195724	6.920727	107.385351	4.173780	0.0	117.276976
min	-21.822222	-27.716667	0.000000	0.000000	0.000000	0.000000	0.0	0.000000
25%	4.604167	2.276389	0.600000	5.796000	116.000000	8.372000	0.0	1011.890000
50%	12.033333	12.033333	0.780000	9.933700	180.000000	10.046400	0.0	1016.420000
75%	18.844444	18.844444	0.890000	14.135800	290.000000	14.812000	0.0	1021.050000
max	39.905556	39.344444	1.000000	63.852600	359.000000	16.100000	0.0	1046.380000

In [9]:
```
new_data.index = pd.to_datetime(new_data.index , utc =True)
```

In [11]:
```
resampled_data = new_data.resample('M').mean() # resample accroading to Month end ('M')
```

In [12]:
```
resampled_data.head()
```

Out[12]:

Formatted Date	Temperature (C)	Apparent Temperature (C)	Humidity	Wind Speed (km/h)	Wind Bearing (degrees)	Visibility (km)	Loud Cover	Pressure (millibars)
2005-12-31 00:00:00+00:00	0.577778	-4.050000	0.890000	17.114300	140.000000	9.982000	0.0	1016.660000
2006-01-31 00:00:00+00:00	-1.677942	-4.173708	0.834610	8.894211	161.018817	7.894064	0.0	1021.204960
2006-02-28 00:00:00+00:00	-0.065394	-2.990716	0.843467	10.957008	197.886905	7.418794	0.0	995.183914
2006-03-31 00:00:00+00:00	4.559274	1.969780	0.778737	14.421488	195.059140	9.602590	0.0	976.436263
2006-04-30 00:00:00+00:00	12.635031	12.098827	0.728625	10.930670	191.877778	10.626760	0.0	1013.493694

In [13]:
```
resampled_data.tail()
```

Out[13]:

Formatted Date	Temperature (C)	Apparent Temperature (C)	Humidity	Wind Speed (km/h)	Wind Bearing (degrees)	Visibility (km)	Loud Cover	Pressure (millibars)
2016-08-31 00:00:00+00:00	21.420296	21.383094	0.674046	9.151378	184.563172	13.948140	0.0	1018.026398
2016-09-30 00:00:00+00:00	18.467924	18.355833	0.688833	6.849029	177.738889	13.723260	0.0	1017.969736
2016-10-31 00:00:00+00:00	10.593141	9.825775	0.827951	11.075846	206.046914	9.208206	0.0	1017.725457
2016-11-30 00:00:00+00:00	5.158800	2.860089	0.848847	10.507636	163.690511	8.725824	0.0	1019.215737
2016-12-31 00:00:00+00:00	1.239158	-2.017272	0.887981	11.024860	179.064603	7.460627	0.0	1019.946339

In [14]:
```
resampled_data['month'] = resampled_data.index.month
```

In [15]:
```
resampled_data['year'] = resampled_data.index.year
```

In [16]:
```
resampled_data.head()
```

Out[16]:

Formatted Date	Temperature (C)	Apparent Temperature (C)	Humidity	Wind Speed (km/h)	Wind Bearing (degrees)	Visibility (km)	Loud Cover	Pressure (millibars)	month	year
2005-12-31 00:00:00+00:00	0.577778	-4.050000	0.890000	17.114300	140.000000	9.982000	0.0	1016.660000	12	2005
2006-01-31 00:00:00+00:00	-1.677942	-4.173708	0.834610	8.894211	161.018817	7.894064	0.0	1021.204960	1	2006
2006-02-28 00:00:00+00:00	-0.065394	-2.990716	0.843467	10.957008	197.886905	7.418794	0.0	995.183914	2	2006
2006-03-31 00:00:00+00:00	4.559274	1.969780	0.778737	14.421488	195.059140	9.602590	0.0	976.436263	3	2006
2006-04-30 00:00:00+00:00	12.635031	12.098827	0.728625	10.930670	191.877778	10.626760	0.0	1013.493694	4	2006

In [17]:
```
resampled_data.index = resampled_data.index.date
```

In [18]:
```
resampled_data = resampled_data[1:] # remove column with year 2005 column
```

In [19]:
```
resampled_data.head()
```

Out[19]:

	Temperature (C)	Apparent Temperature (C)	Humidity	Wind Speed (km/h)	Wind Bearing (degrees)	Visibility (km)	Loud Cover	Pressure (millibars)	month	year
2006-01-31	-1.677942	-4.173708	0.834610	8.894211	161.018817	7.894064	0.0	1021.204960	1	2006
2006-02-28	-0.065394	-2.990716	0.843467	10.957008	197.886905	7.418794	0.0	995.183914	2	2006
2006-03-31	4.559274	1.969780	0.778737	14.421488	195.059140	9.602590	0.0	976.436263	3	2006
2006-04-30	12.635031	12.098827	0.728625	10.930670	191.877778	10.626760	0.0	1013.493694	4	2006
2006-05-31	15.650732	15.539479	0.721801	10.174161	209.310484	11.748066	0.0	1016.629785	5	2006

In [20]:
```
month_to_month_AT = {}
for month in range(1,13):
    month_to_month_AT[month] = list(resampled_data[resampled_data['month'] == month]['Apparent Temperature (C)'].values)
```

In [21]:
```
title = {1:'Jan',2:'Feb',3:'March',4:'April',5:'May',6:'June',7:'July',8:'Aug',9:'Sep',
         10:'Oct',11:'Nov',12:'Dec'}
def plot_AT_or_Humidity(what_for , month_dict):
    for index in range(1,13):
        t = title[index]
        plt.plot(range(2006,2017),month_dict[index])
        plt.title(what_for + ' for ' +t+' Month')
        plt.show()
```

In [22]:
```
month_to_month_Humidity = {}
for month in range(1,13):
    month_to_month_Humidity[month] = list(resampled_data[resampled_data['month'] == month]['Humidity'].values)
```

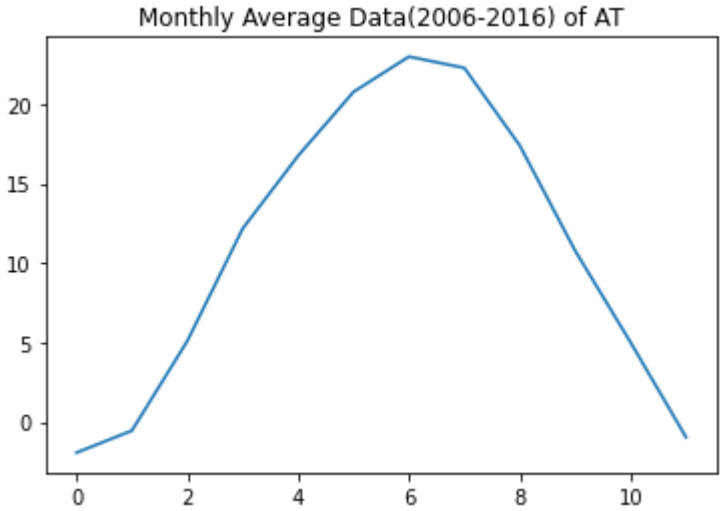
In [23]:

```
def find_avg_difference(month_dict):
    difference = []
    for month in range(1,13):
        difference.append(np.mean(month_dict[month]))
    return difference
```

In [24]:
```
AT_difference_monthly = find_avg_difference(month_to_month_AT)
Humidity_difference_monthly = find_avg_difference(month_to_month_Humidity)
```

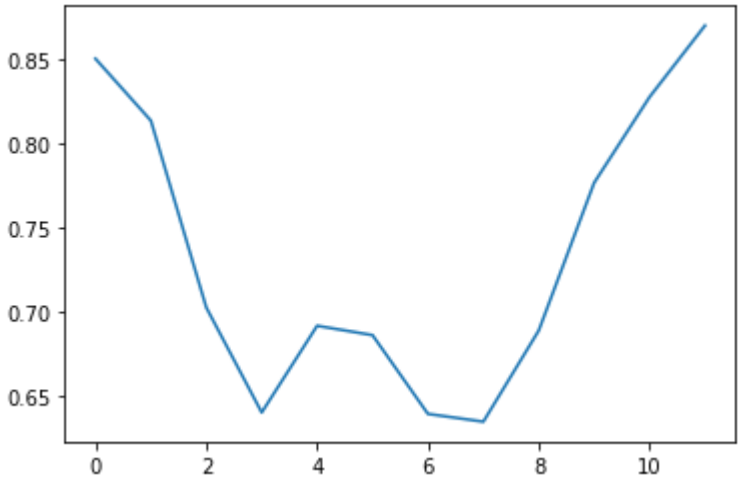
In [25]:
```
plt.plot(AT_difference_monthly)
plt.title('Monthly Average Data(2006-2016) of AT')
```

Out[25]:
```
Text(0.5, 1.0, 'Monthly Average Data(2006-2016) of AT')
```



In [26]:
```
plt.plot(Humidity_difference_monthly)
```

Out[26]:
```
[<matplotlib.lines.Line2D at 0x20b70ec7370>]
```



In [27]:
```
new_data.index = new_data.index.date
```

In [28]:
```
new_data.index = pd.DatetimeIndex(new_data.index)
```

In [29]:
```
pd.options.mode.chained_assignment = None
```

In [30]:
```
new_data['month'] = new_data.index.month
new_data['year'] = new_data.index.year
```

In [31]:
```
def find_average_monthly_AT_or_Humidity(what_for):
    avg_data_tempreature_monthly = {}
    for year in range(2006,2017):
        for month in range(1,13):
            result = list(new_data.loc[(new_data['month'] == month)&(new_data['year']==year) , :][what_for].values)
            if month not in avg_data_tempreature_monthly:
                avg_data_tempreature_monthly[month] = [np.mean(result)]
            else:
                avg_data_tempreature_monthly[month].append(np.mean(result))
    return avg_data_tempreature_monthly
```

In [32]:
```
AT_monthly_average = find_average_monthly_AT_or_Humidity('Apparent Temperature (C)')
Humidity_monthly_average = find_average_monthly_AT_or_Humidity('Humidity')
```

In [33]:
```
AT = pd.DataFrame(AT_monthly_average)
```

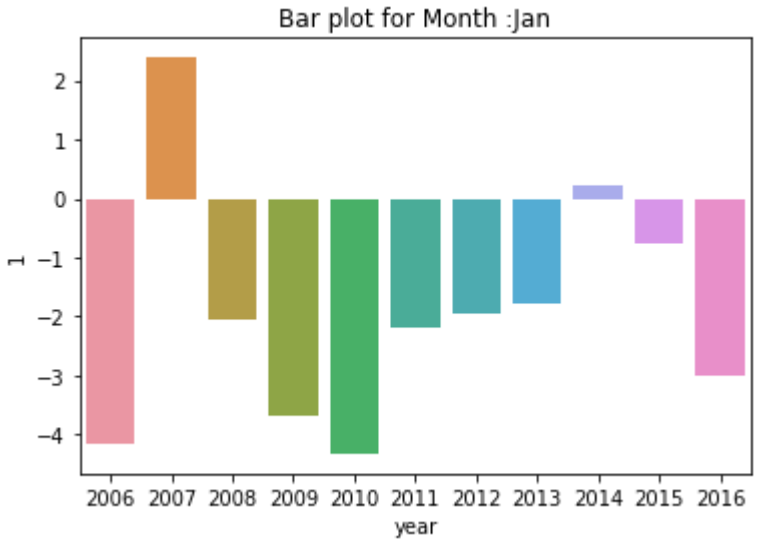
```
AT['year'] = range(2006,2017)
```

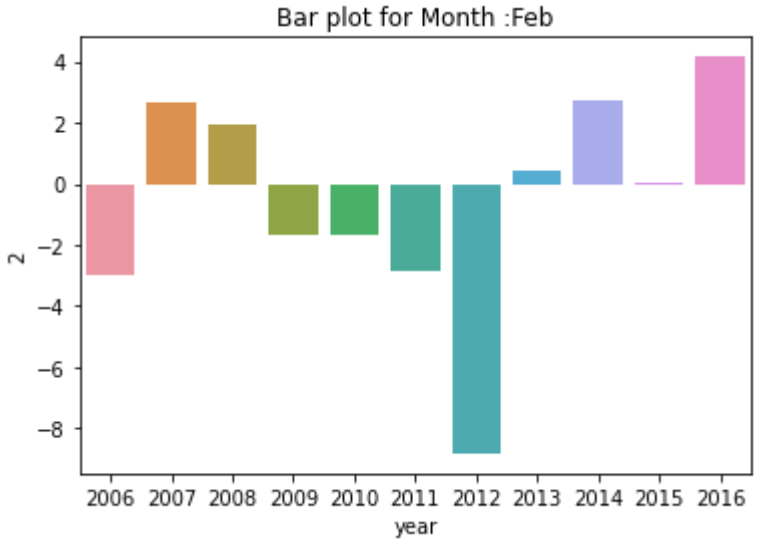
In [34]:

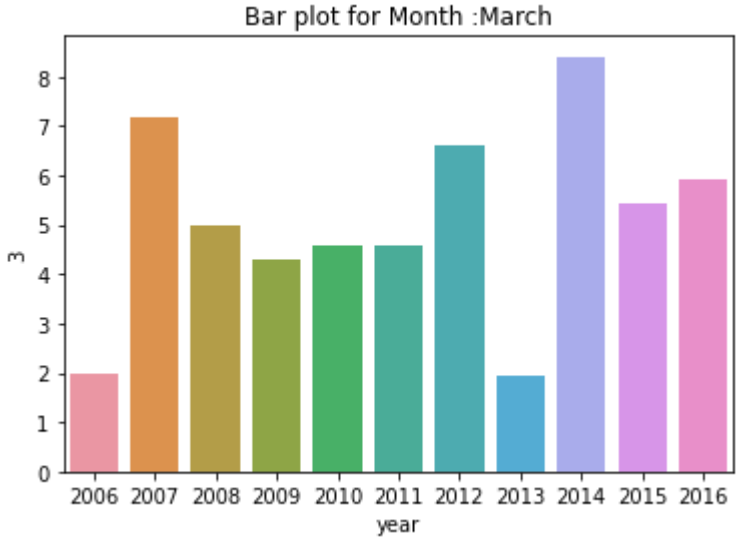
```
H = pd.DataFrame(Humidity_monthly_average)
H['year'] = range(2006,2017)
```

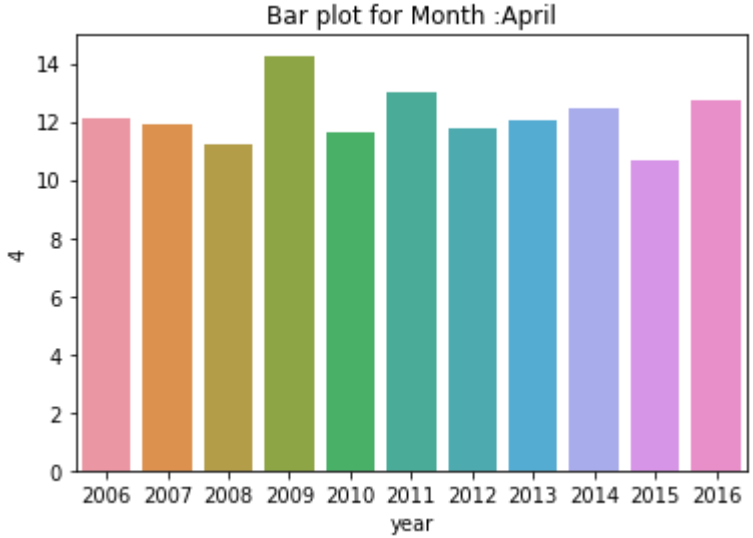
In [35]:

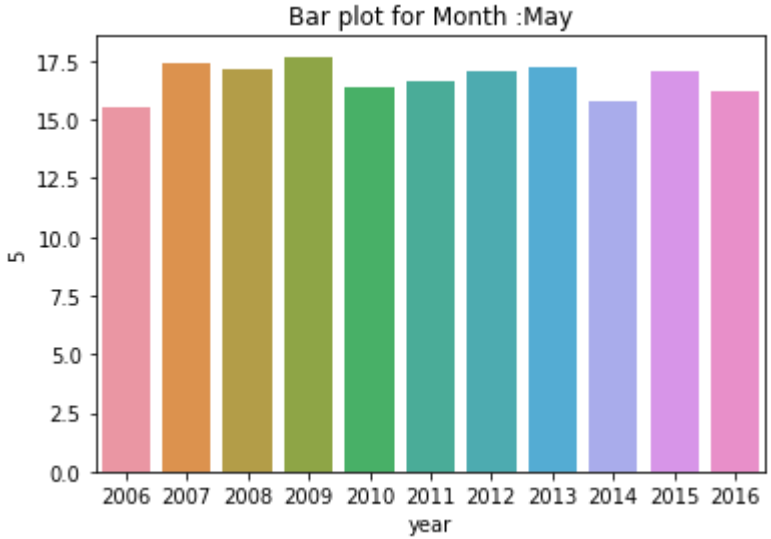
```
for month in range(1,13):
    sns.barplot(x = AT['year'] , y = AT[month])

    plt.title('Bar plot for Month :' + title[month])
    plt.show()
```

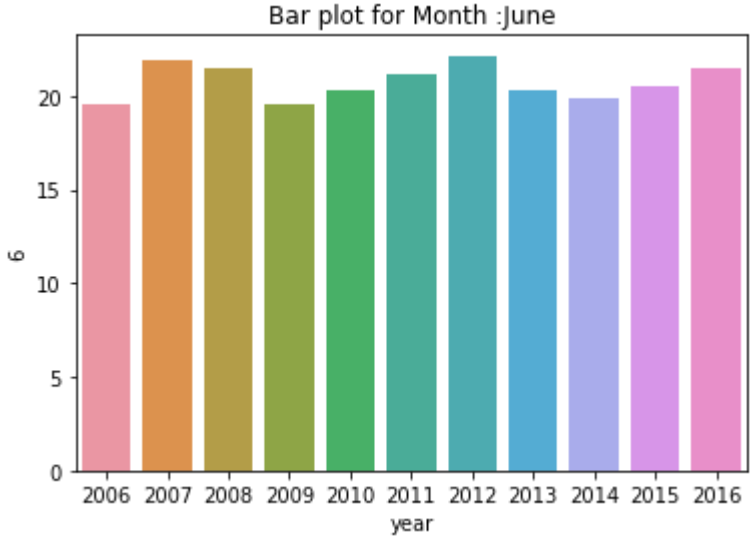


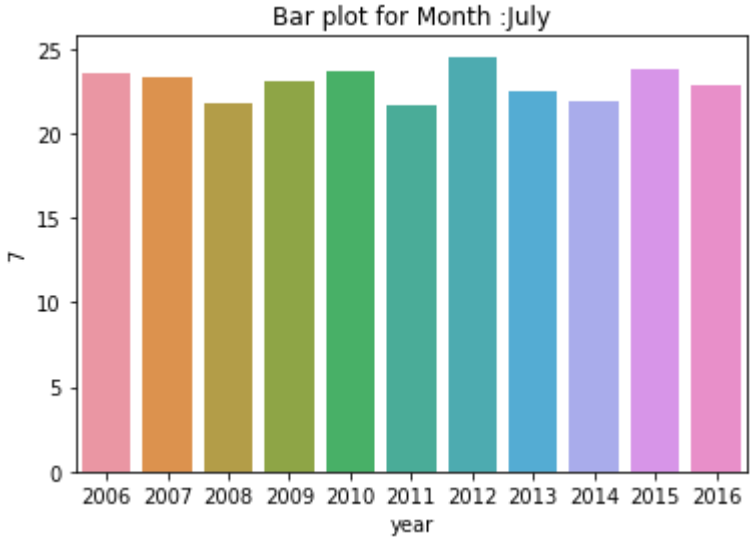


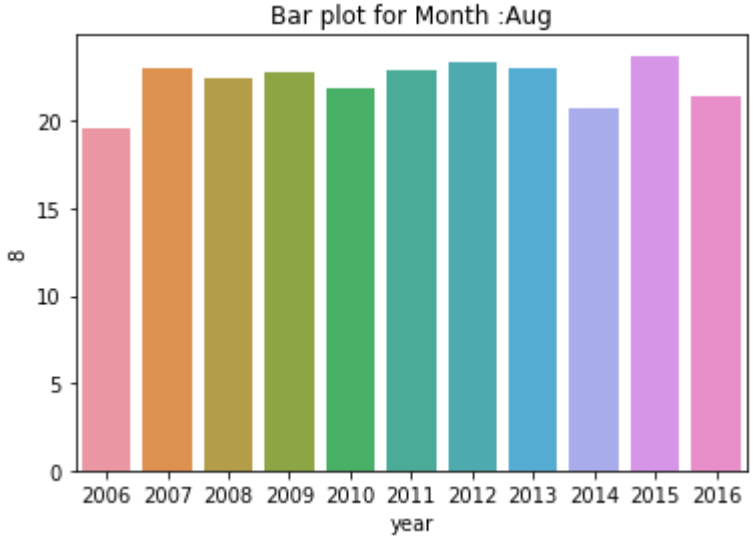

Bar plot for Month :March
3
year
2006 2007 2008 2009 2010 2011 2012 2013 2014 2015 2016
0 1 2 3 4 5 6 7 8

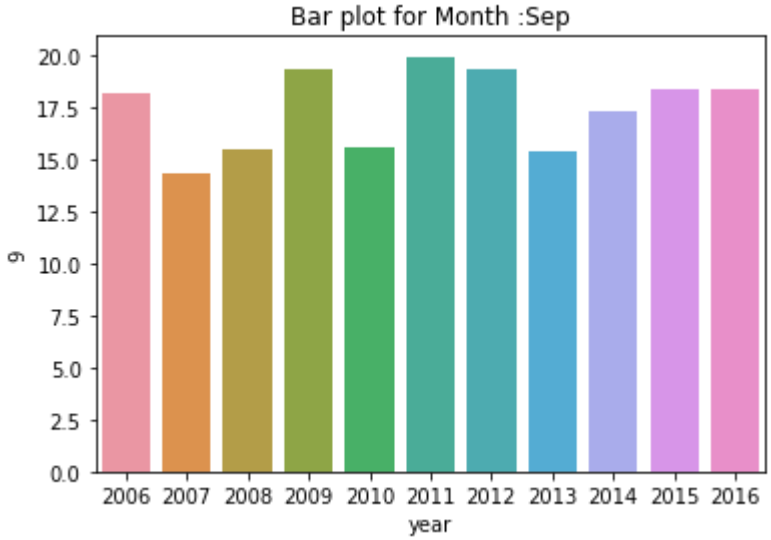

Bar plot for Month :April
4
year
2006 2007 2008 2009 2010 2011 2012 2013 2014 2015 2016
0 2 4 6 8 10 12 14

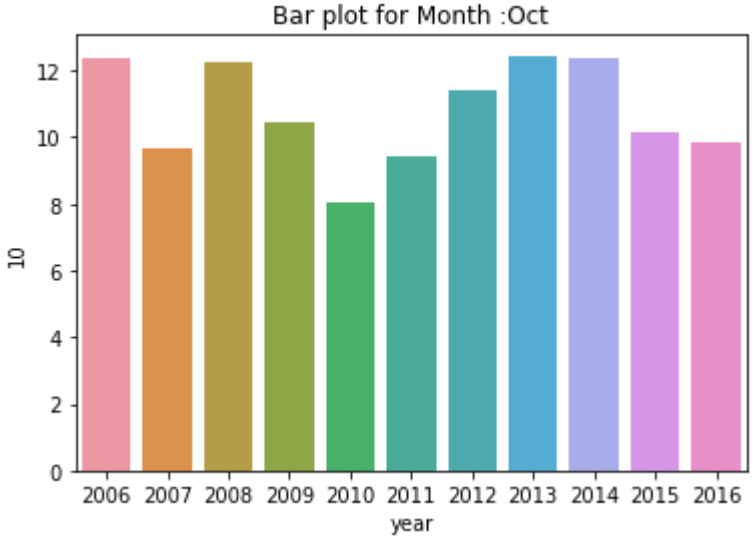

Bar plot for Month :May
17.5
15.0
12.5
10.0
7.5
5.0
2.5
0.0
5
2006 2007 2008 2009 2010 2011 2012 2013 2014 2015 2016
year

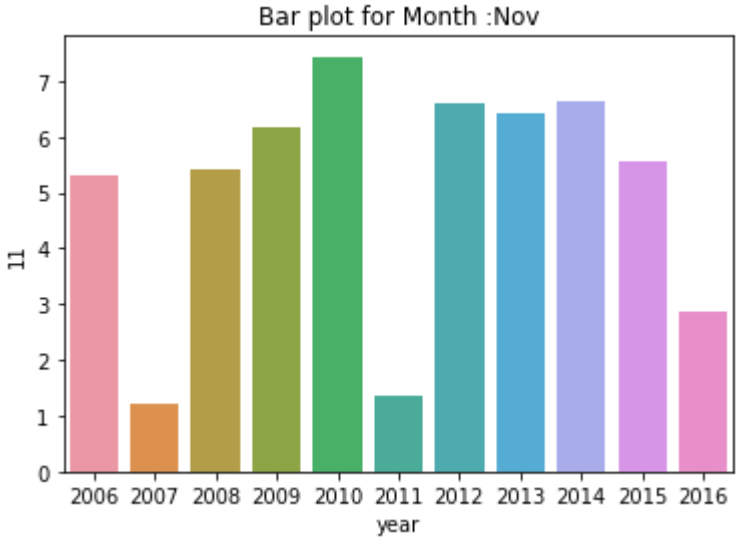

Bar plot for Month :June
20
15
10
5
0
6
2006 2007 2008 2009 2010 2011 2012 2013 2014 2015 2016
year

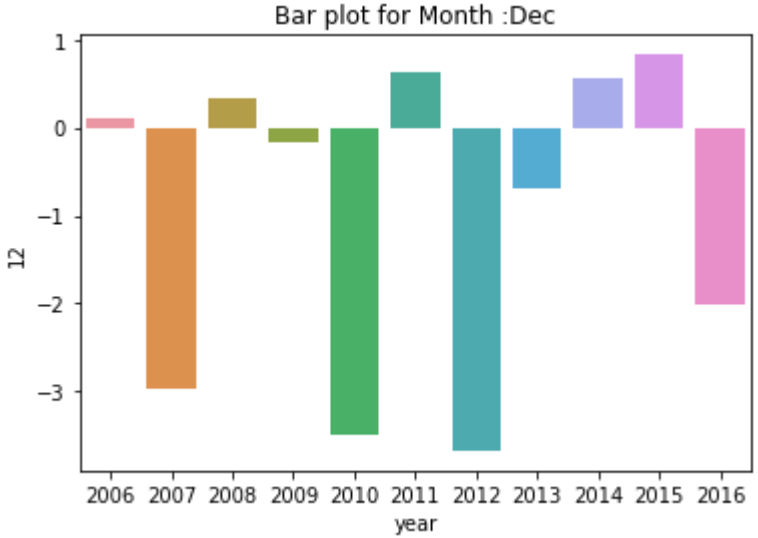

Bar plot for Month :July
25
20
15
10
5
0
7
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
year


Bar plot for Month :Aug
20
15
10
5
0
8
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
year


Bar plot for Month :Sep
9
year
2006 2007 2008 2009 2010 2011 2012 2013 2014 2015 2016
0.0 2.5 5.0 7.5 10.0 12.5 15.0 17.5 20.0

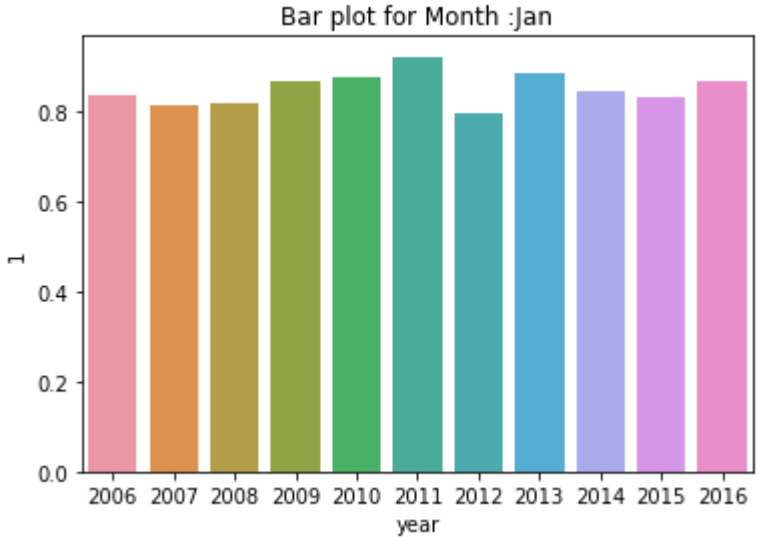

Bar plot for Month :Oct
10
year
2006 2007 2008 2009 2010 2011 2012 2013 2014 2015 2016
0 2 4 6 8 10 12

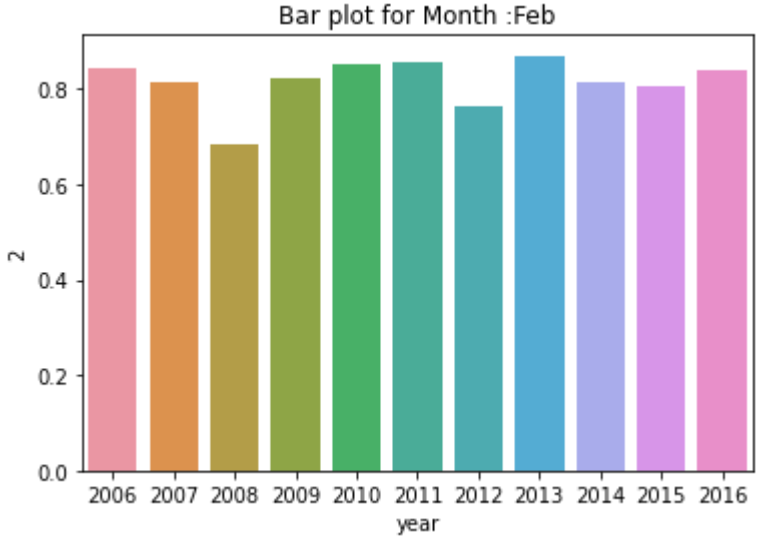


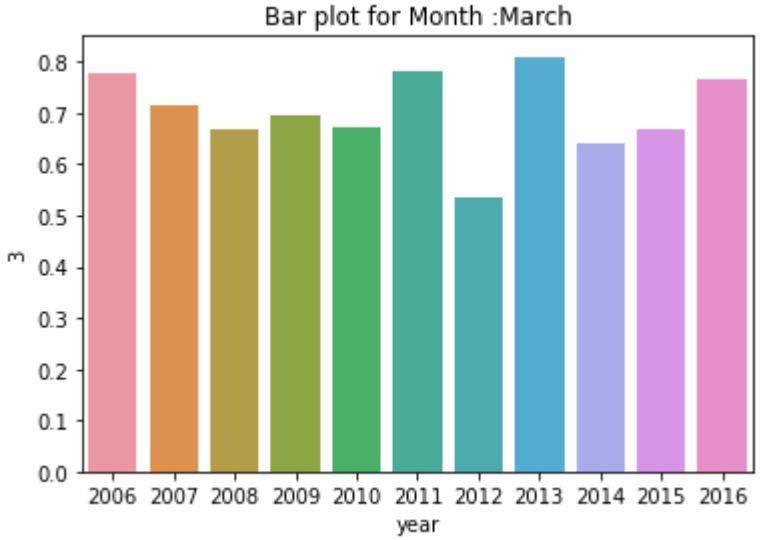


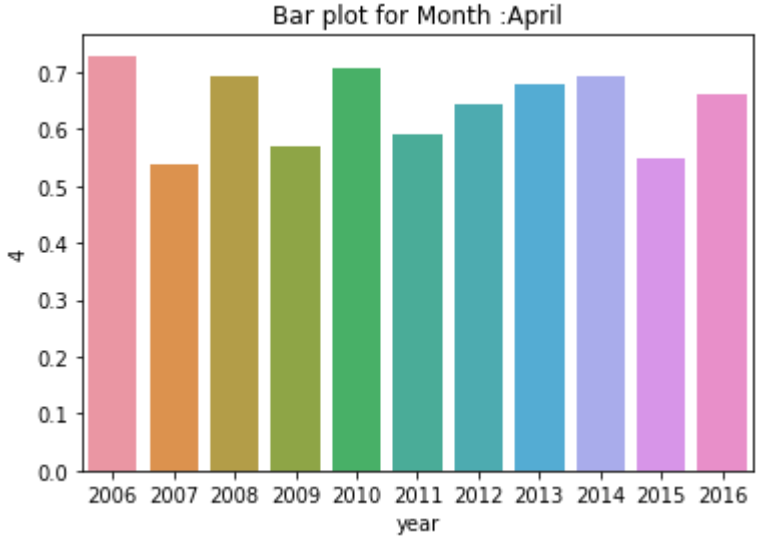
In [36]:

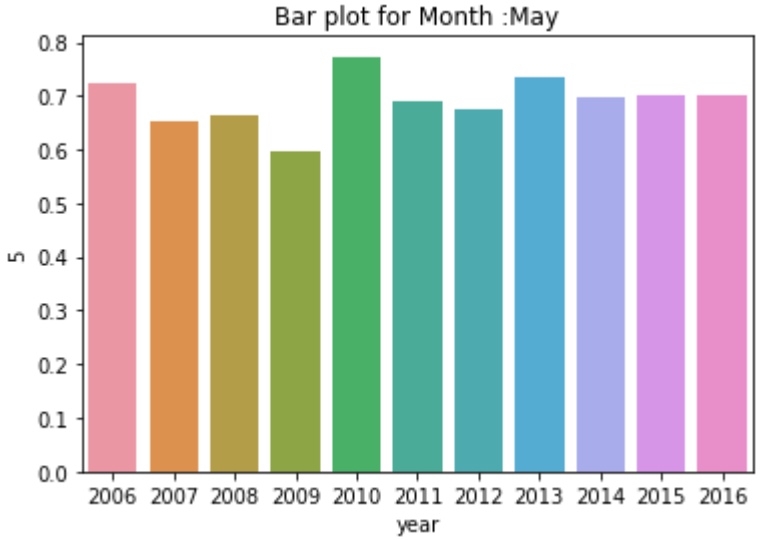
```
for month in range(1,13):
    sns.barplot(x = H['year'] , y = H[month])

    plt.title('Bar plot for Month :' + title[month])
    plt.show()
```

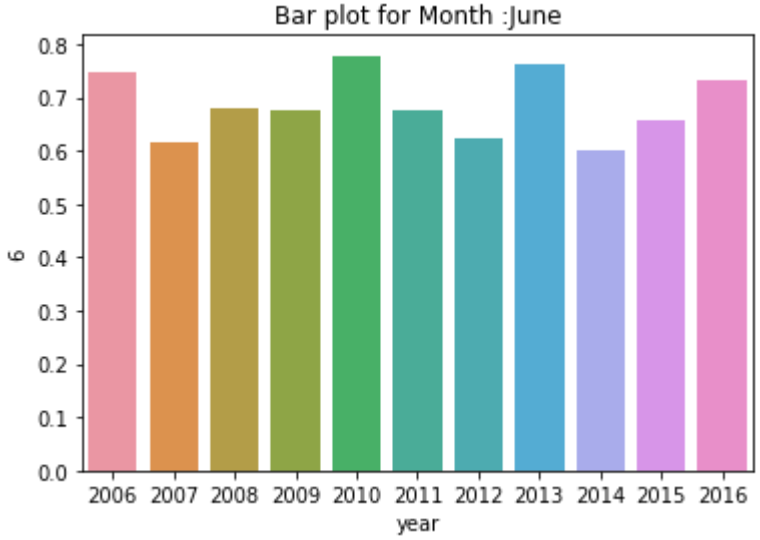

Bar plot for Month :Jan
1
0.0
0.2
0.4
0.6
0.8
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
year

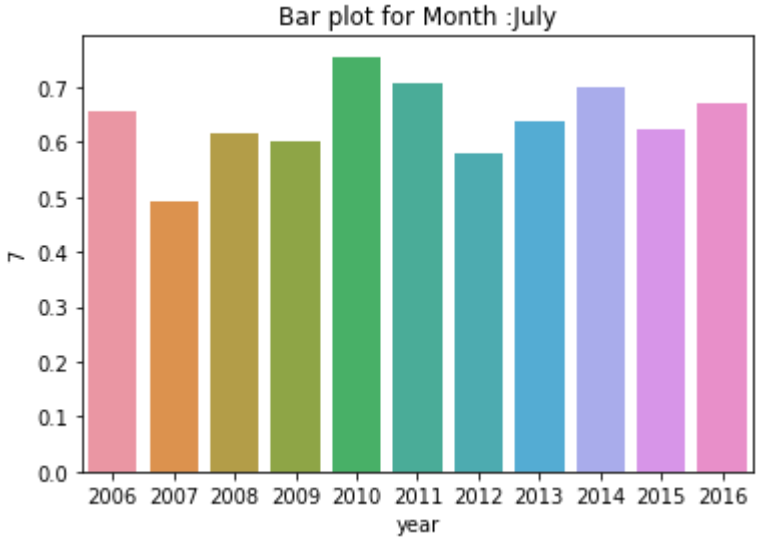

Bar plot for Month :Feb
2
0.0
0.2
0.4
0.6
0.8
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
year

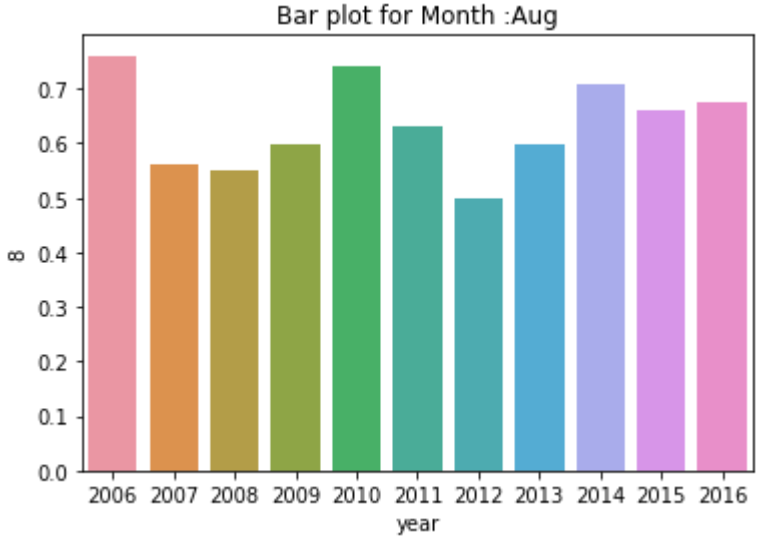

Bar plot for Month :March
3
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
2006 2007 2008 2009 2010 2011 2012 2013 2014 2015 2016
year

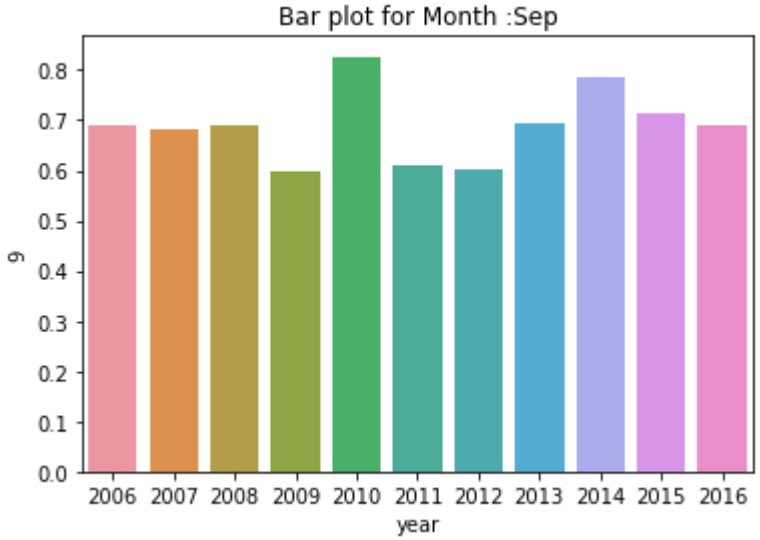

Bar plot for Month :April
4
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
2006 2007 2008 2009 2010 2011 2012 2013 2014 2015 2016
year

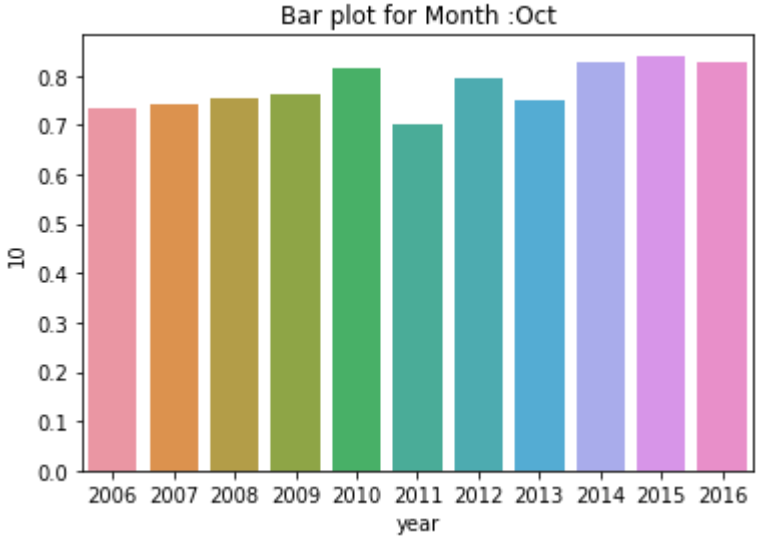

Bar plot for Month :May
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
5
2006 2007 2008 2009 2010 2011 2012 2013 2014 2015 2016
year

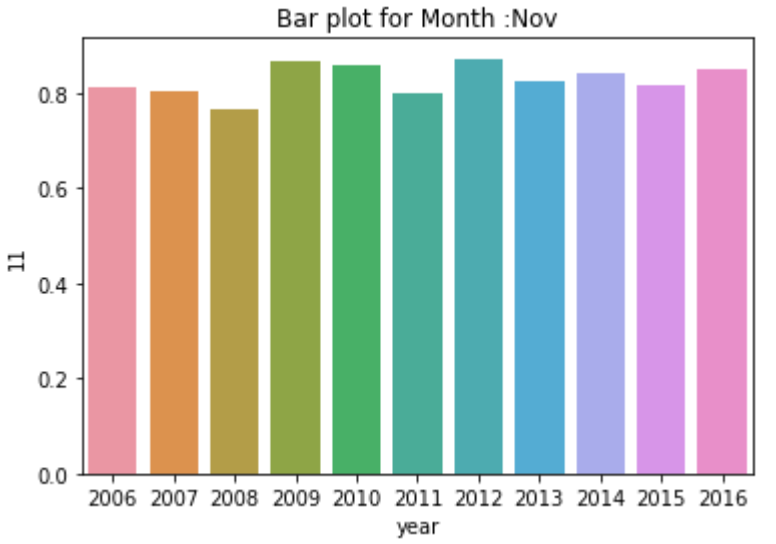

Bar plot for Month :June
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
6
2006 2007 2008 2009 2010 2011 2012 2013 2014 2015 2016
year

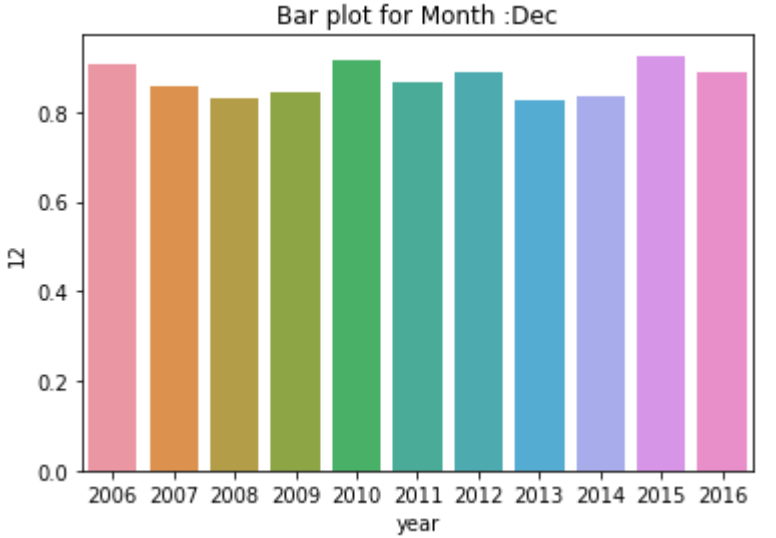

Bar plot for Month :July
7
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
year


Bar plot for Month :Aug
8
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
year


Bar plot for Month :Sep
9
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
year


Bar plot for Month :Oct
10
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
year



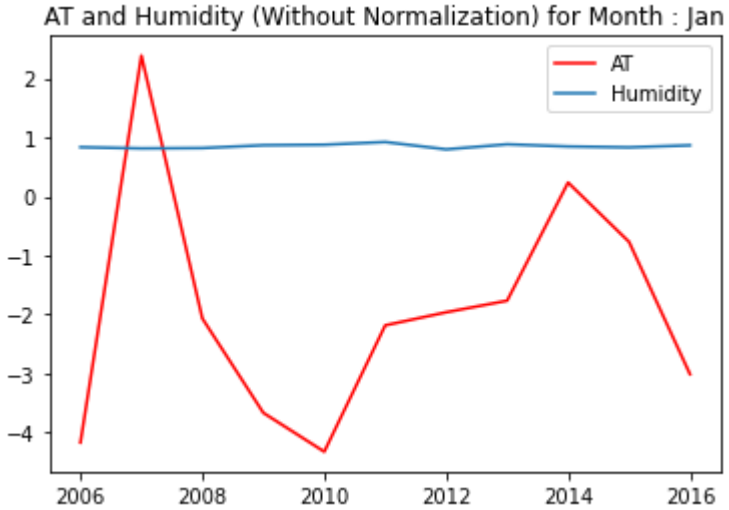


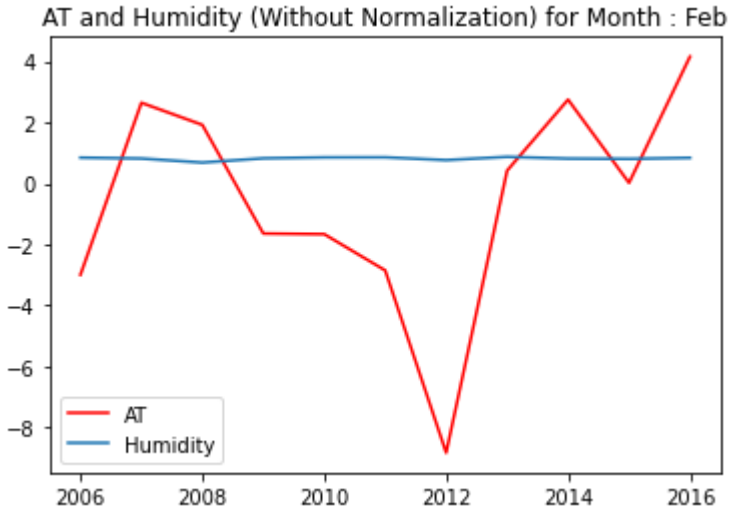
In [37]:

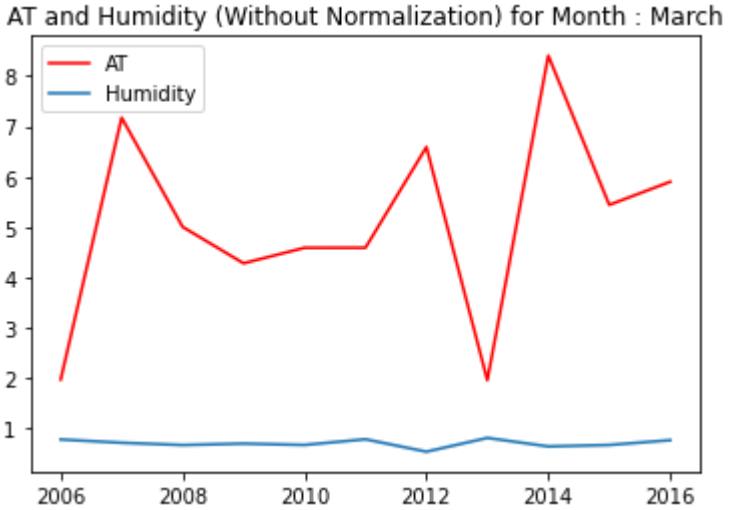
```
def plot_Humidty_and_AT():
    for month in range(1,12):
        plt.plot(range(2006,2017),AT_monthly_average[month] , label = 'AT' , color = 'red')
        plt.plot(range(2006,2017),Humidity_monthly_average[month] , label = 'Humidity')
        plt.legend()
        plt.title('AT and Humidity (Without Normalization) for Month : '+ title[month])
        plt.show()
```

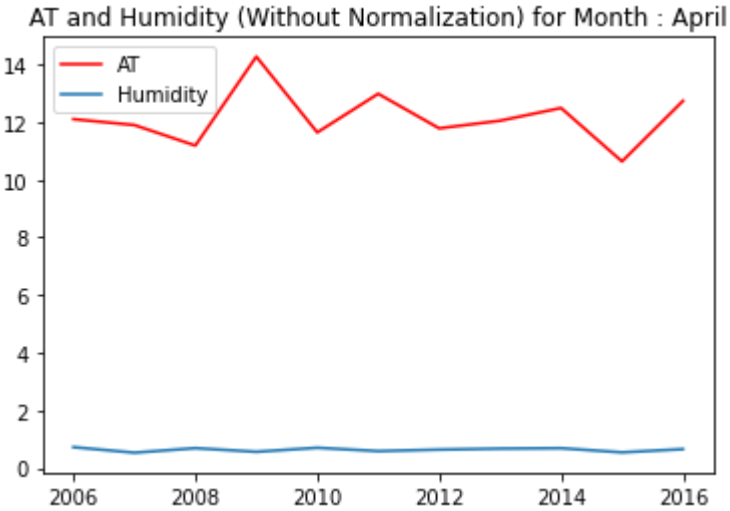
In [38]:

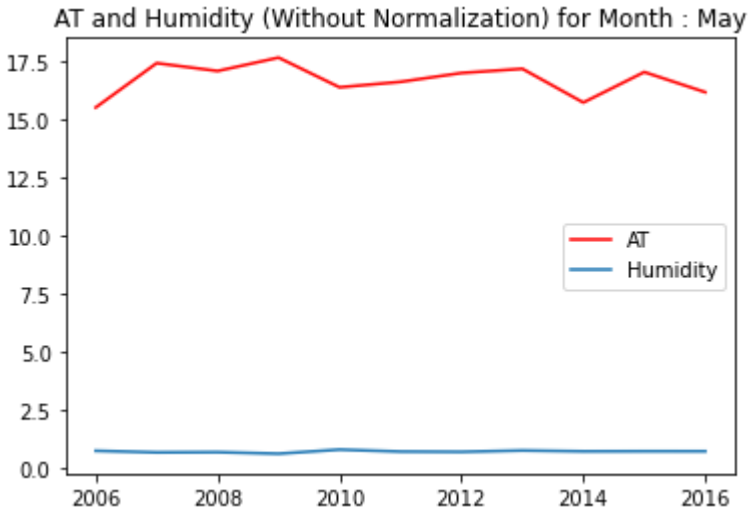
```
plot_Humidty_and_AT()
```

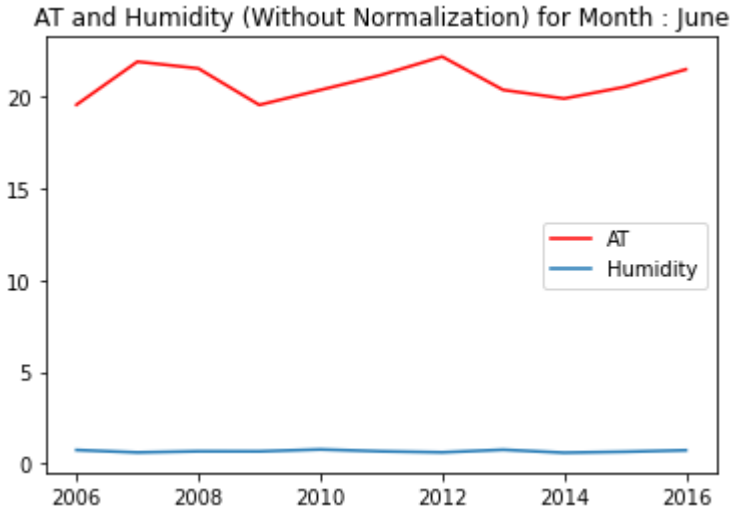

AT and Humidity (Without Normalization) for Month : Jan
AT
Humidity
2
1
0
−1
−2
−3
−4
2006
2008
2010
2012
2014
2016

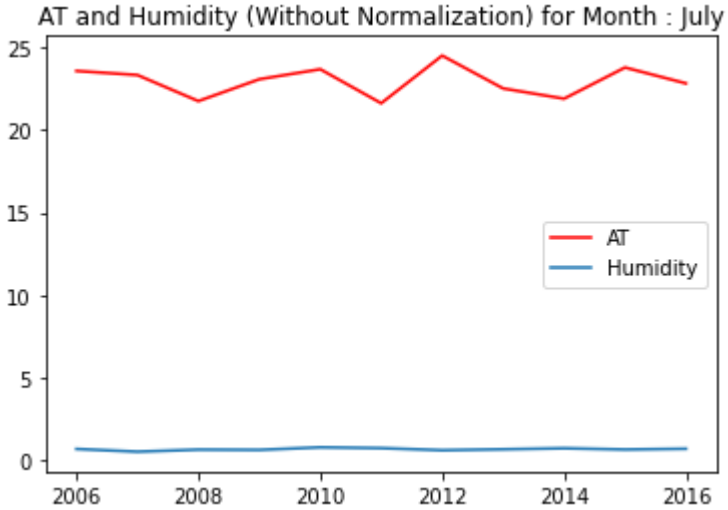

AT and Humidity (Without Normalization) for Month : Feb
AT
Humidity
4
2
0
−2
−4
−6
−8
2006
2008
2010
2012
2014
2016

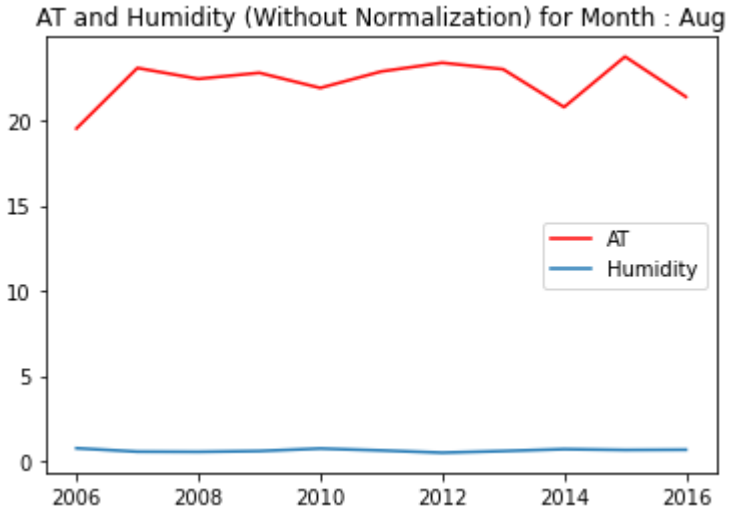

AT and Humidity (Without Normalization) for Month : March
AT
Humidity
8
7
6
5
4
3
2
1
2006
2008
2010
2012
2014
2016

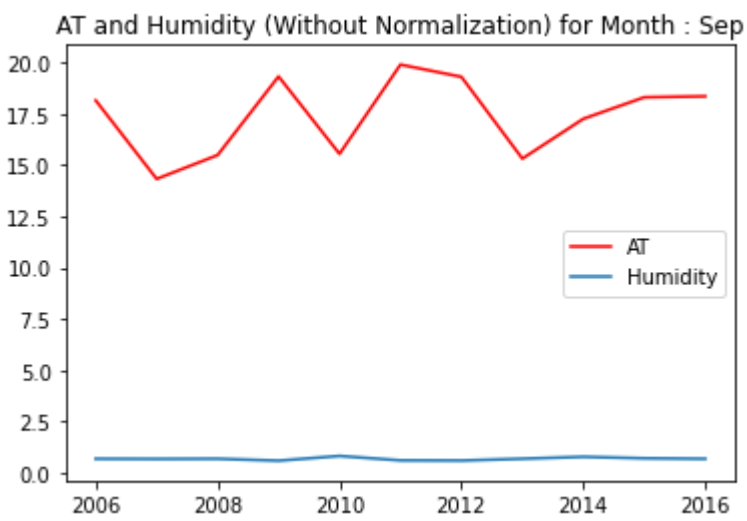

AT and Humidity (Without Normalization) for Month : April
AT
Humidity
14
12
10
8
6
4
2
0
2006
2008
2010
2012
2014
2016

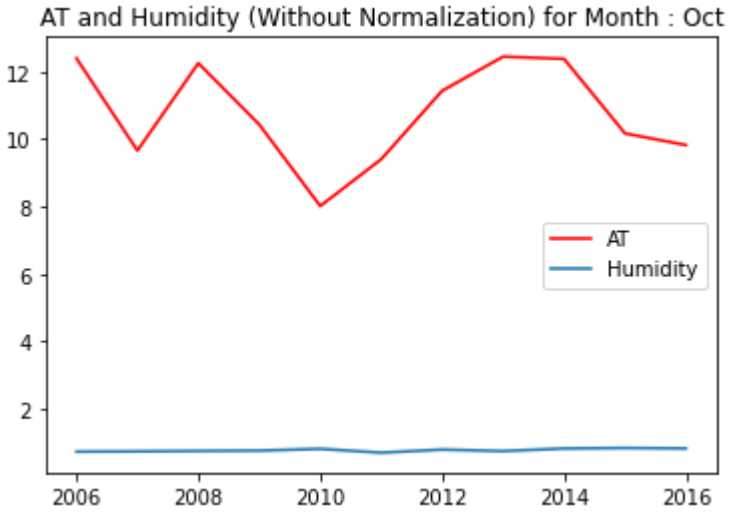

AT and Humidity (Without Normalization) for Month : May
17.5
15.0
12.5
10.0
7.5
5.0
2.5
0.0
2006
2008
2010
2012
2014
2016
AT
Humidity

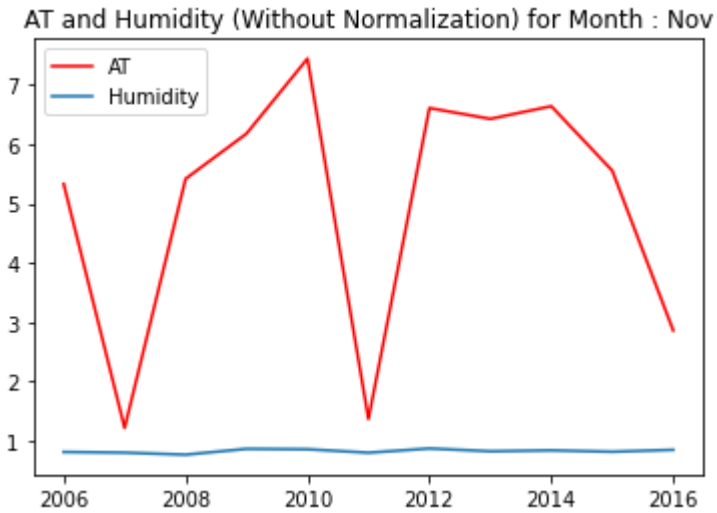

AT and Humidity (Without Normalization) for Month : June
20
15
10
5
0
2006
2008
2010
2012
2014
2016
AT
Humidity


AT and Humidity (Without Normalization) for Month : July
25
20
15
10
5
0
2006
2008
2010
2012
2014
2016
AT
Humidity


AT and Humidity (Without Normalization) for Month : Aug
20
15
10
5
0
2006
2008
2010
2012
2014
2016
AT
Humidity


AT and Humidity (Without Normalization) for Month : Sep
20.0
17.5
15.0
12.5
10.0
7.5
5.0
2.5
0.0
2006
2008
2010
2012
2014
2016
AT
Humidity

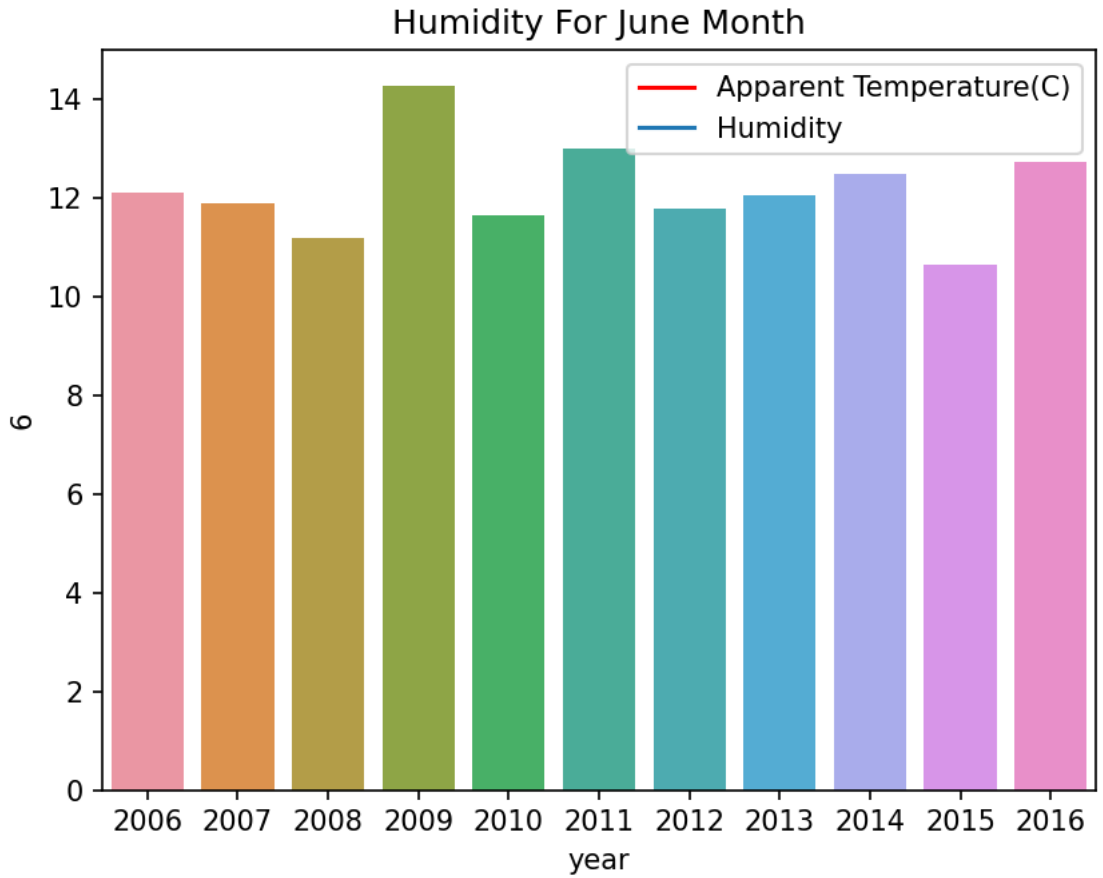

AT and Humidity (Without Normalization) for Month : Oct
12
10
8
6
4
2
2006
2008
2010
2012
2014
2016
AT
Humidity



In [39]:
```
%matplotlib notebook
```

In [40]:
```
sns.barplot(H['year'] , H[4])
plt.title('Humidity For April Month')
plt.show()
```

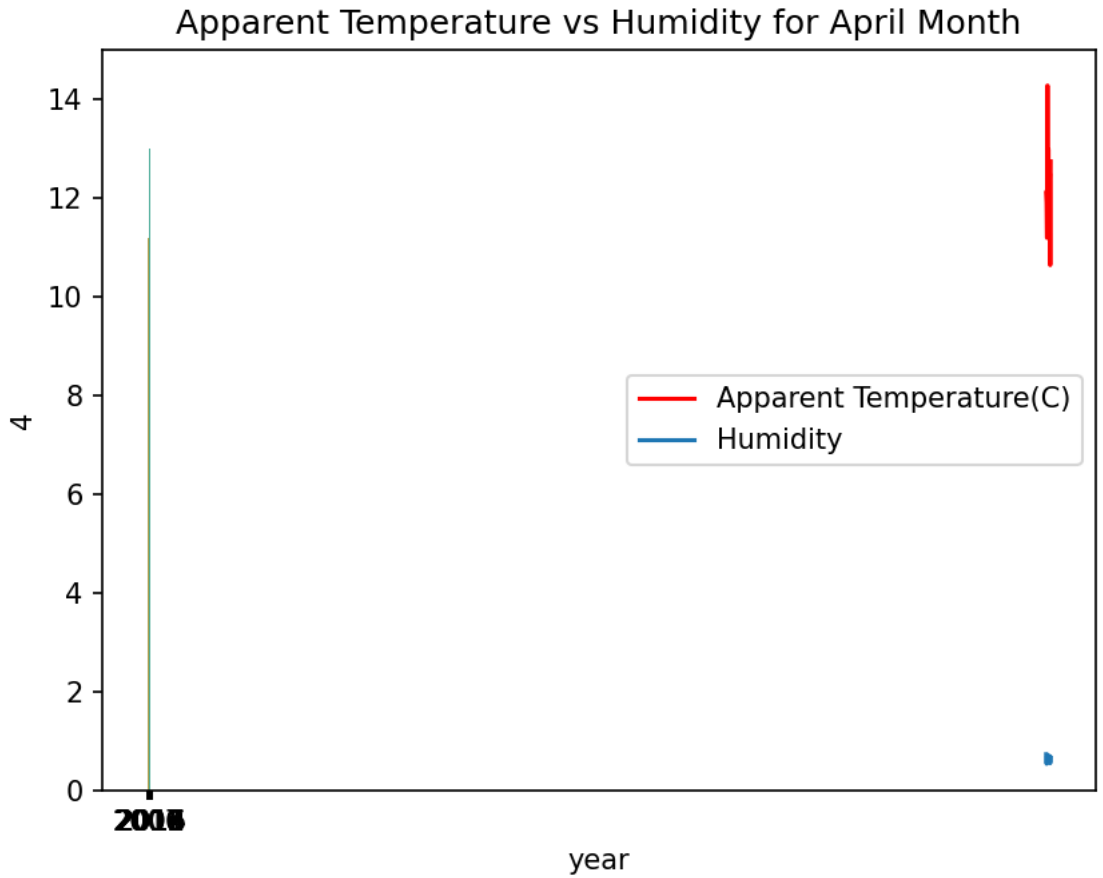
```
c:\users\sandeep\appdata\local\programs\python\python39\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following variab
les as keyword args: x, y. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an expli
cit keyword will result in an error or misinterpretation.
  warnings.warn(
```



In [44]:

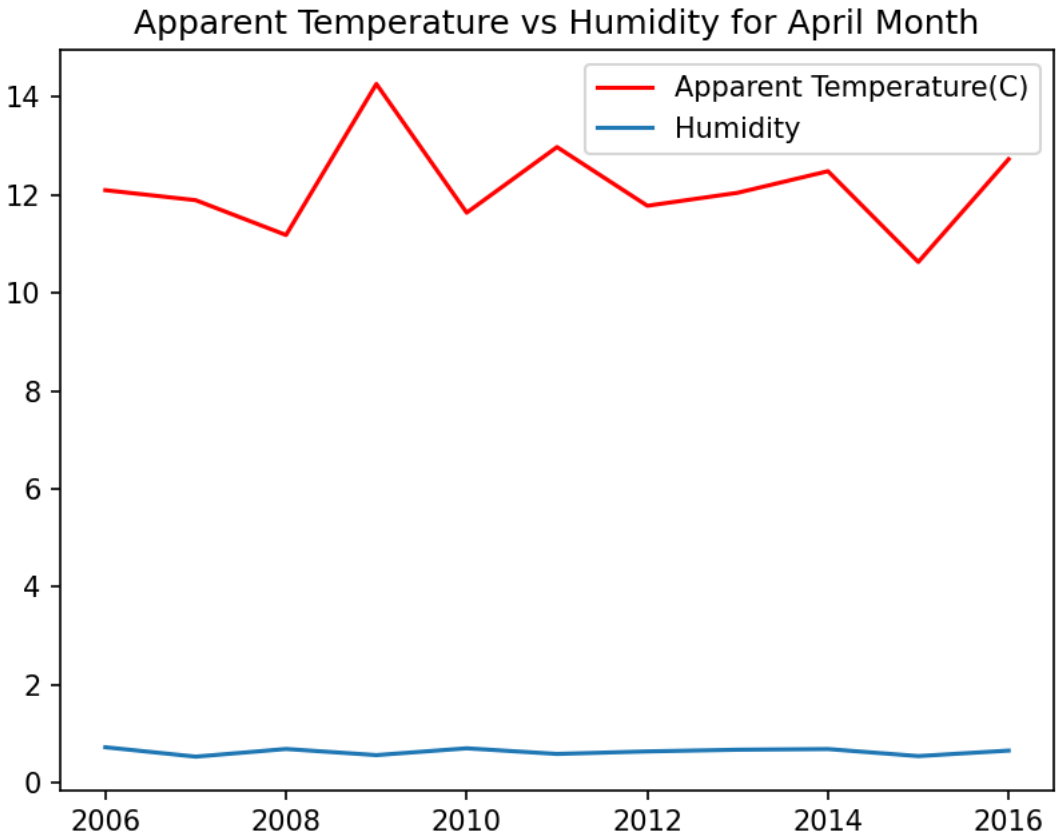
```
sns.barplot(AT['year'] , AT[4])
plt.title('Apparent Temperature For April Month')
plt.show()
```

```
c:\users\sandeep\appdata\local\programs\python\python39\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following variab
les as keyword args: x, y. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an expli
cit keyword will result in an error or misinterpretation.
  warnings.warn(
```



In [46]:

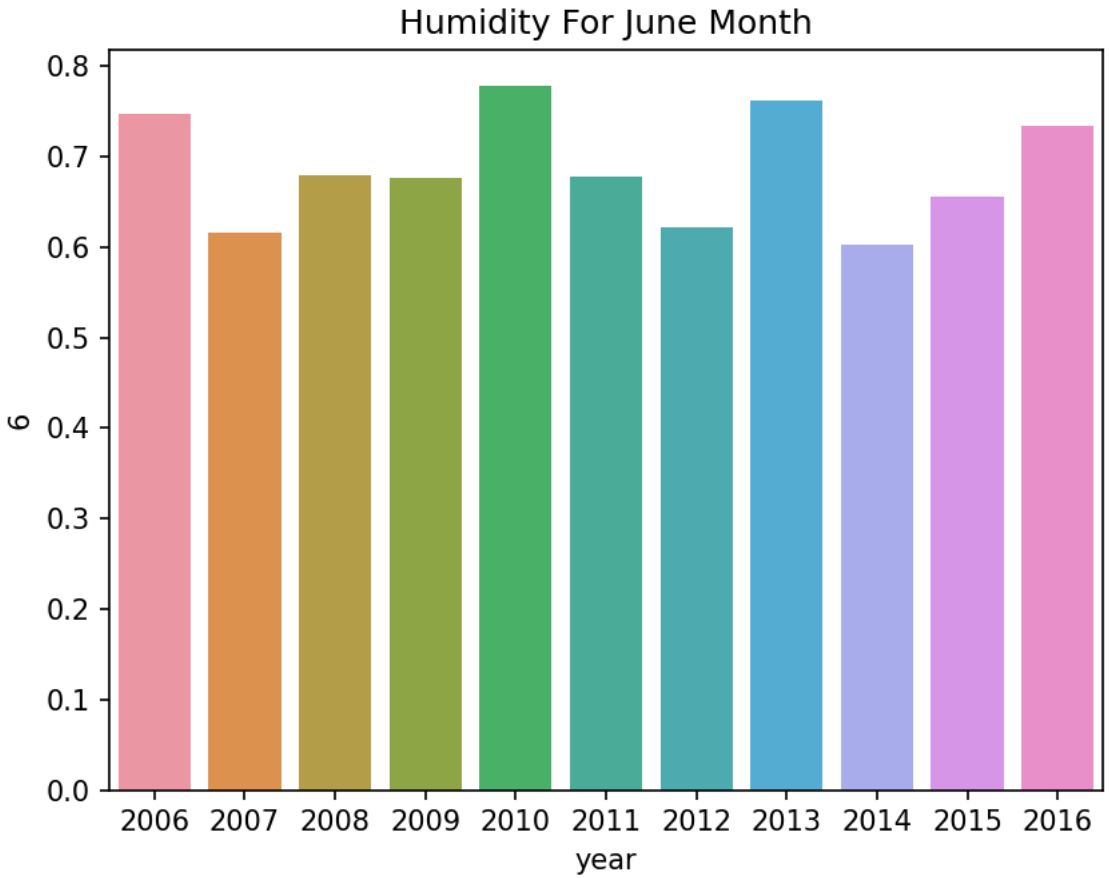
```
plt.plot(range(2006,2017),AT_monthly_average[4] , label = 'Apparent Temperature(C)' , color = 'red')
plt.plot(range(2006,2017),Humidity_monthly_average[4] , label = 'Humidity')
plt.legend()
plt.title('Apparent Temperature vs Humidity for April Month')
plt.show()
```



In [47]:

```
sns.barplot(H['year'] , H[6])
plt.title('Humidity For June Month')
plt.show()
```

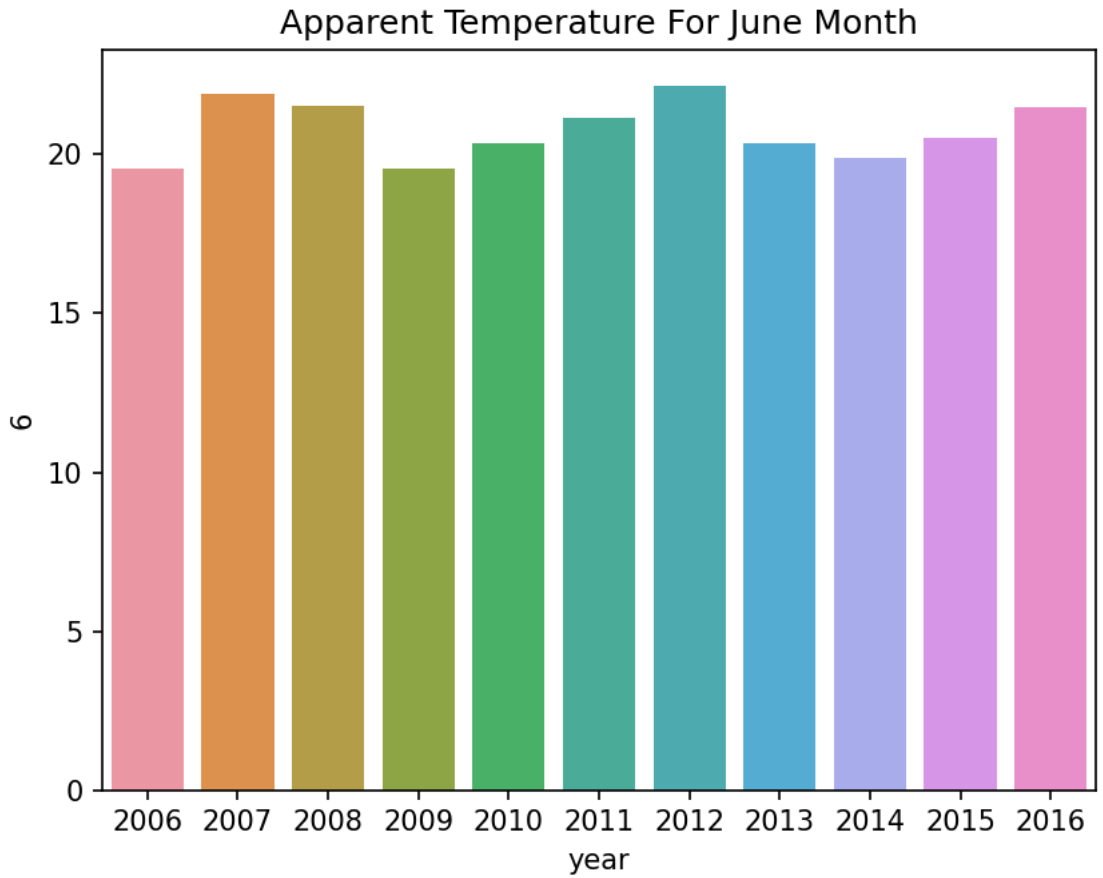
```
c:\users\sandeep\appdata\local\programs\python\python39\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following variab
les as keyword args: x, y. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an expli
cit keyword will result in an error or misinterpretation.
  warnings.warn(
```



In [48]:

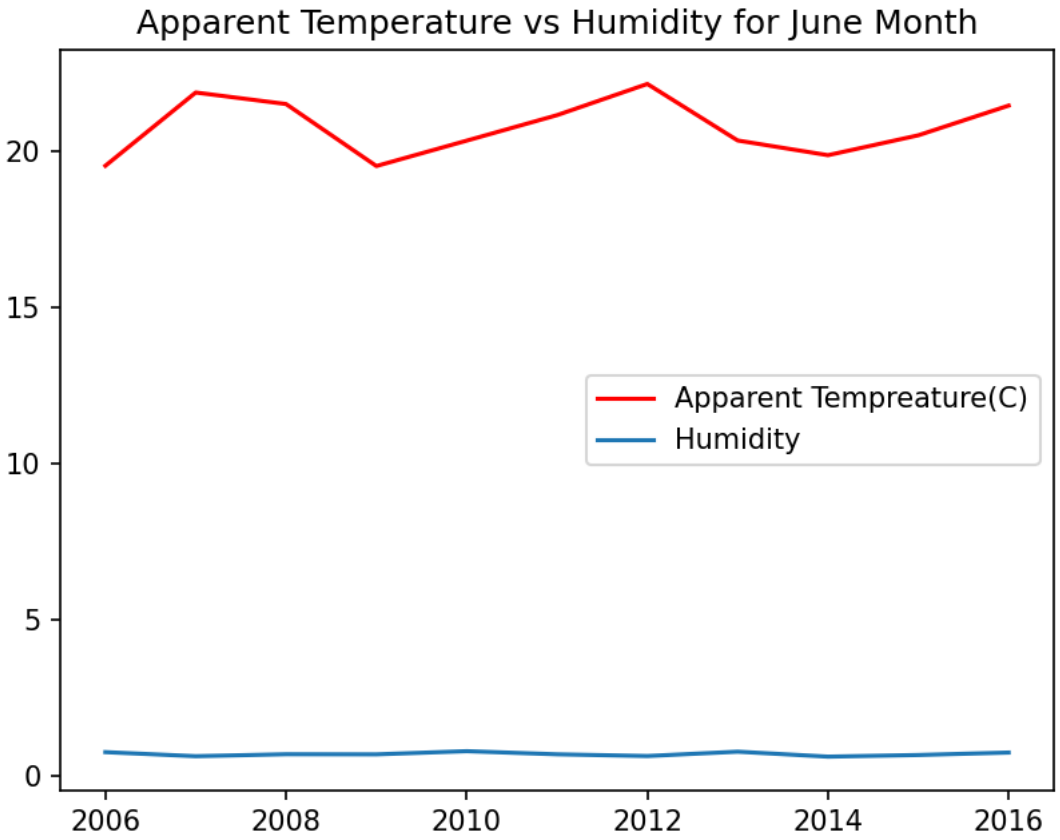
```
sns.barplot(AT['year'] , AT[6])
plt.title('Apparent Temperature For June Month')
plt.show()
```

```
c:\users\sandeep\appdata\local\programs\python\python39\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following variab
les as keyword args: x, y. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an expli
cit keyword will result in an error or misinterpretation.
  warnings.warn(
```



In [49]:

```
plt.plot(range(2006,2017),AT_monthly_average[6] , label = 'Apparent Tempreature(C)' , color = 'red')
plt.plot(range(2006,2017),Humidity_monthly_average[6] , label = 'Humidity')
plt.legend()
plt.title('Apparent Temperature vs Humidity for June Month')
plt.show()
```



In [50]:
```
print("Completed by Sandeep Potdukhe")
```

```
Completed by Sandeep Potdukhe
```

In [ ]: