In [1]:

```
#This is Linear regression predictive analysis to check whether after a session of shopping guidance,
#whether customers order through their mobile app or prefer a website


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

In [2]:

```
%matplotlib inline
```

In [3]:

```
customers=pd.read_csv('Ecommerce Customers')
```

In [4]:

```
customers.head()
```

Out[4]:

	Email	Address	Avatar	Avg. Session Length	Time on App	Time on Website	Length of Membership	Yearly Amount Spent
0	mstephenson@fernandez.com	835 Frank Tunnel\nWrightmouth, MI 82180-9605	Violet	34.497268	12.655651	39.577668	4.082621	587.951054
1	hduke@hotmail.com	4547 Archer Common\nDiazchester, CA 06566-8576	DarkGreen	31.926272	11.109461	37.268959	2.664034	392.204933
2	pallen@yahoo.com	24645 Valerie Unions Suite 582\nCobbborough, D...	Bisque	33.000915	11.330278	37.110597	4.104543	487.547505
3	riverarebecca@gmail.com	1414 David Throughway\nPort Jason, OH 22070-1220	SaddleBrown	34.305557	13.717514	36.721283	3.120179	581.852344
4	mstephens@davidson-herman.com	14023 Rodriguez Passage\nPort Jacobville, PR 3...	MediumAquaMarine	33.330673	12.795189	37.536653	4.446308	599.406092

In [5]:

```
customers.describe()
```

Out[5]:

	Avg. Session Length	Time on App	Time on Website	Length of Membership	Yearly Amount Spent
count	500.000000	500.000000	500.000000	500.000000	500.000000
mean	33.053194	12.052488	37.060445	3.533462	499.314038
std	0.992563	0.994216	1.010489	0.999278	79.314782
min	29.532429	8.508152	33.913847	0.269901	256.670582
25%	32.341822	11.388153	36.349257	2.930450	445.038277
50%	33.082008	11.983231	37.069367	3.533975	498.887875
75%	33.711985	12.753850	37.716432	4.126502	549.313828
max	36.139662	15.126994	40.005182	6.922689	765.518462

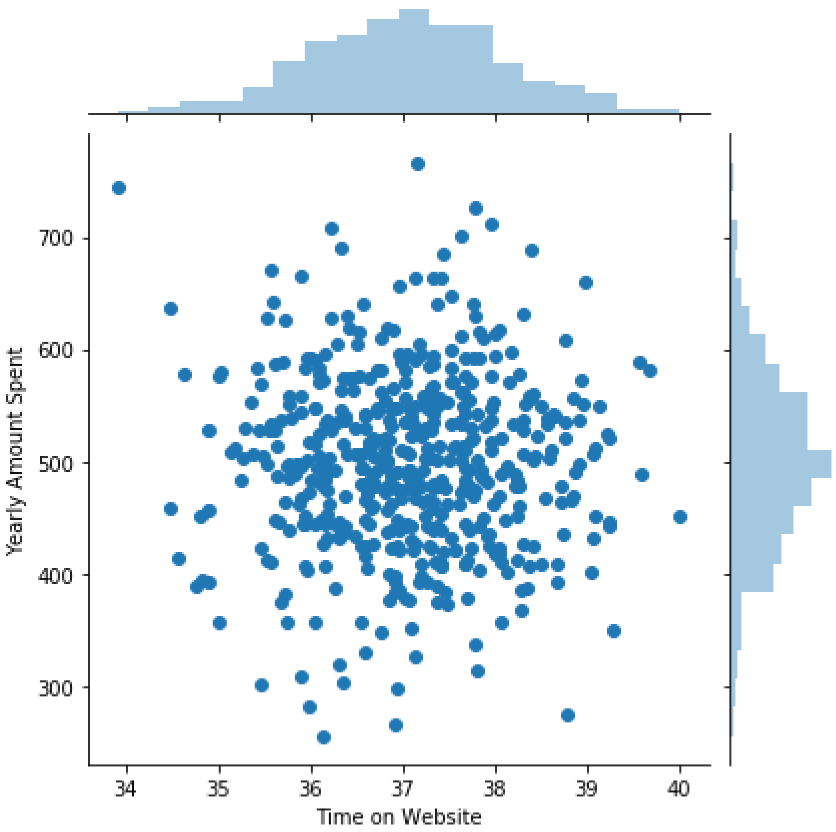
In [6]:

```
customers.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 500 entries, 0 to 499
Data columns (total 8 columns):
 #   Column                Non-Null Count  Dtype  
---  ------                --------------  -----  
 0   Email                 500 non-null    object 
 1   Address               500 non-null    object 
 2   Avatar                500 non-null    object 
 3   Avg. Session Length   500 non-null    float64
 4   Time on App           500 non-null    float64
 5   Time on Website       500 non-null    float64
 6   Length of Membership  500 non-null    float64
 7   Yearly Amount Spent   500 non-null    float64
dtypes: float64(5), object(3)
memory usage: 31.4+ KB
```

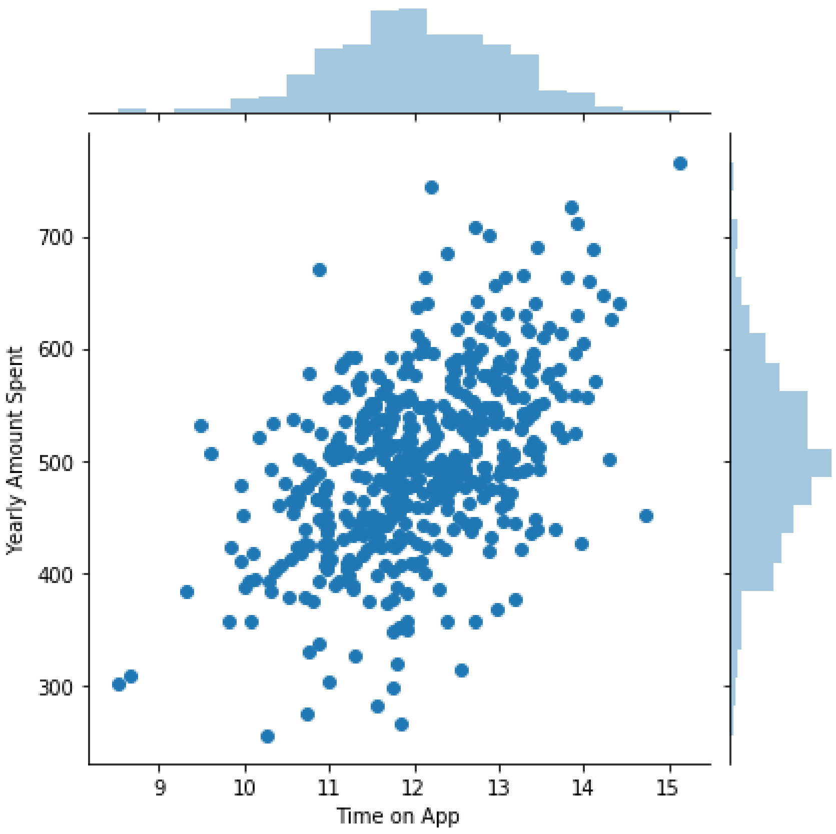
In [8]:
```
#Above functions were used to get a basic know how of the data
#Moving on to some exploratory data analysis

#Using seaborn to create a jointplot to compare the Time on Website and Yearly Amount Spent columns.
sns.jointplot(data=customers,x='Time on Website',y='Yearly Amount Spent')
```

Out[8]:
```
<seaborn.axisgrid.JointGrid at 0x1a404efa248>
```



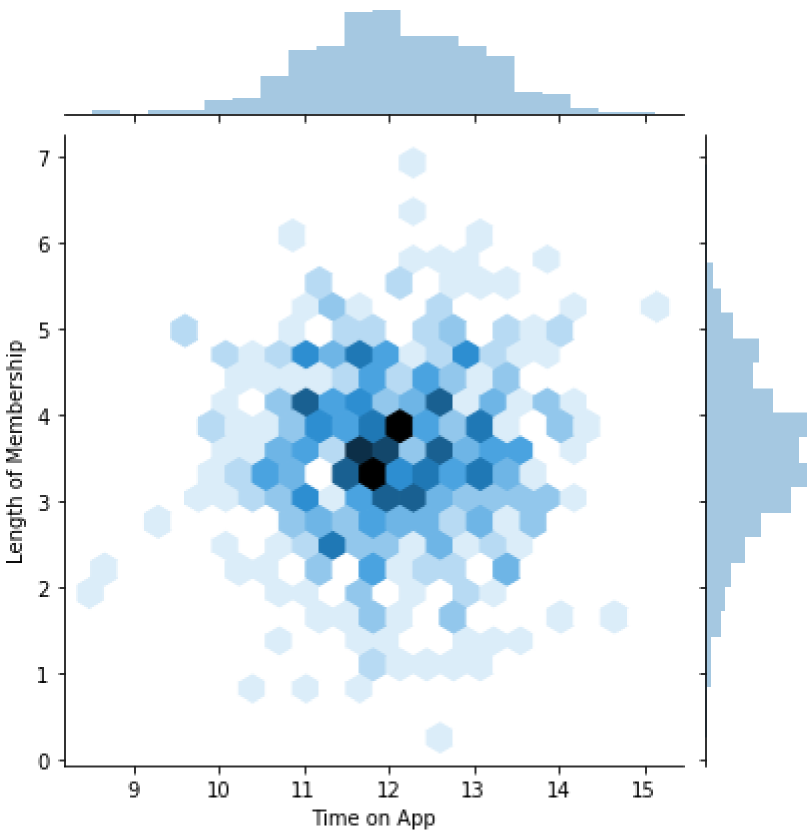
In [9]:
```
sns.jointplot(data=customers,x='Time on App',y='Yearly Amount Spent')
```

Out[9]:
```
<seaborn.axisgrid.JointGrid at 0x1a404ec7208>
```



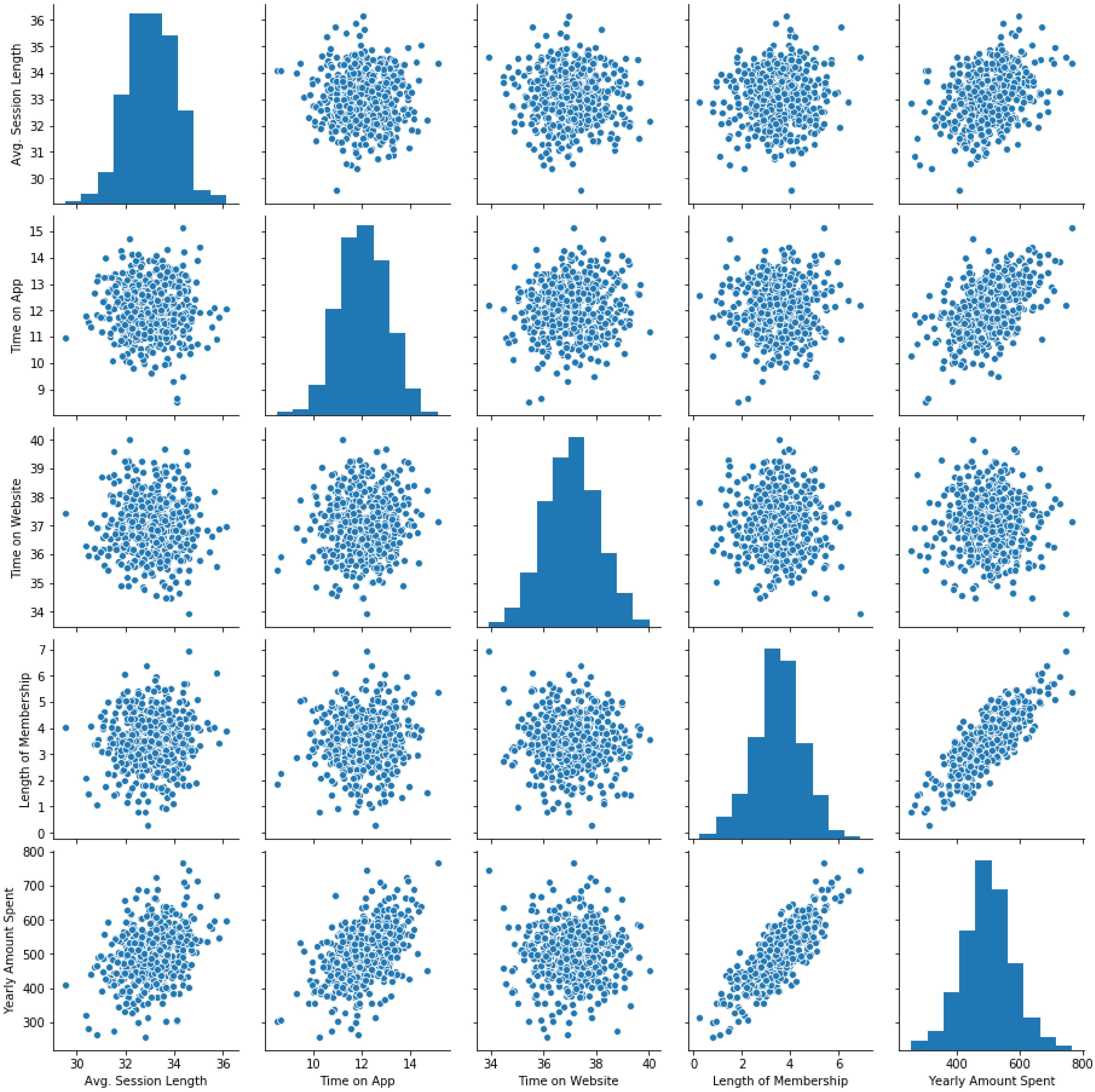
```
In [10]: #Using jointplot to create a 2D hex bin plot comparing Time on App and Length of Membership
         sns.jointplot(data=customers,x='Time on App',y='Length of Membership',kind='hex')
```

```
Out[10]: <seaborn.axisgrid.JointGrid at 0x1a406865f48>
```

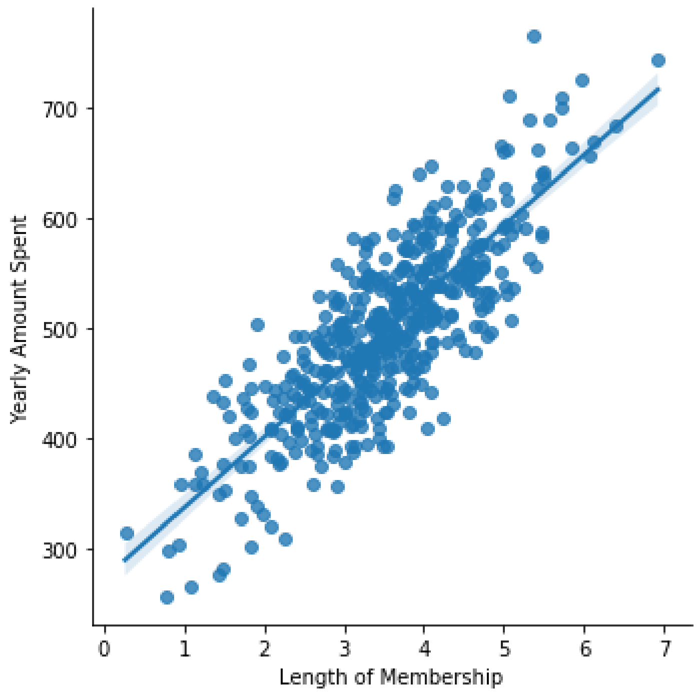


```
In [11]: sns.pairplot(customers)
```

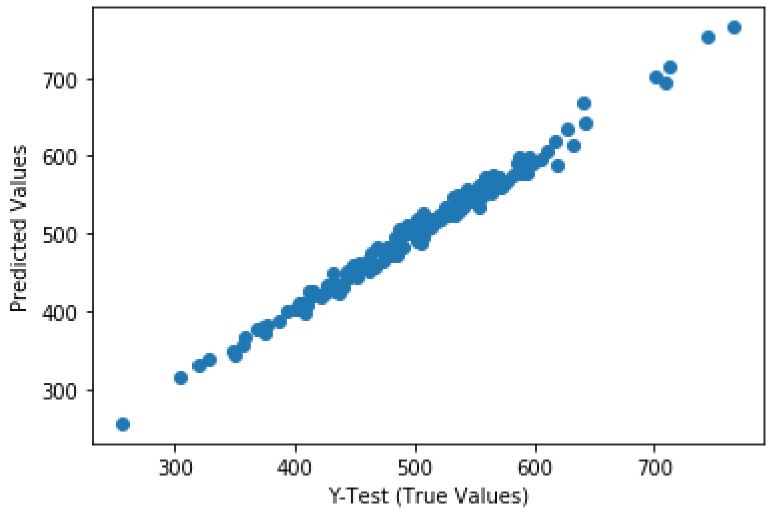
```
Out[11]: <seaborn.axisgrid.PairGrid at 0x1a4071c91c8>
```



```
In [12]: #Based off this plot Length of Membership looks to be the most correlated feature with Yearly Amount Spent
         #lets look at a linear plot
         sns.lmplot(x='Length of Membership',y='Yearly Amount Spent',data=customers)

Out[12]: <seaborn.axisgrid.FacetGrid at 0x1a4092b0d08>
```



```
In [13]: #Now that we've explored the data a bit, let's go ahead and split the data into training and testing sets.
         customers.columns

Out[13]: Index(['Email', 'Address', 'Avatar', 'Avg. Session Length', 'Time on App',
               'Time on Website', 'Length of Membership', 'Yearly Amount Spent'],
              dtype='object')

In [16]: y=customers['Yearly Amount Spent']
         X=customers[[ 'Avg. Session Length', 'Time on App',
                'Time on Website', 'Length of Membership']]

In [17]: from sklearn.model_selection import train_test_split

In [18]: X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=101)

In [19]: from sklearn.linear_model import LinearRegression

In [20]: #Now its time to train our model on our training data!
         lm=LinearRegression()

In [21]: lm.fit(X_train,y_train)

Out[21]: LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None, normalize=False)

In [22]: lm.coef_

Out[22]: array([25.98154972, 38.59015875,  0.19040528, 61.27909654])

In [24]: #Now that we have fit our model, let's evaluate its performance by predicting off the test values!
         predictions=lm.predict(X_test)

In [26]: plt.scatter(y_test,predictions)
         plt.xlabel('Y-Test (True Values)')
         plt.ylabel('Predicted Values')

Out[26]: Text(0, 0.5, 'Predicted Values')
```



In [27]:

```
#EVALUATING Let's evaluate our model performance by calculating the residual sum of squares and the explained variance score (R^2).
from sklearn import metrics
```

In [30]:

```
print('Mean Absolute Error',metrics.mean_absolute_error(y_test,predictions))
print('Mean Squared Error',metrics.mean_squared_error(y_test,predictions))
print('Root Mean Squared Error',np.sqrt(metrics.mean_squared_error(y_test,predictions)))
```

```
Mean Absolute Error 7.228148653430838
Mean Squared Error 79.81305165097461
Root Mean Squared Error 8.933815066978642
```

In [31]:

```
metrics.explained_variance_score(y_test,predictions)  #How much variance our model explains gives about 98% pretty good right
```

Out[31]:

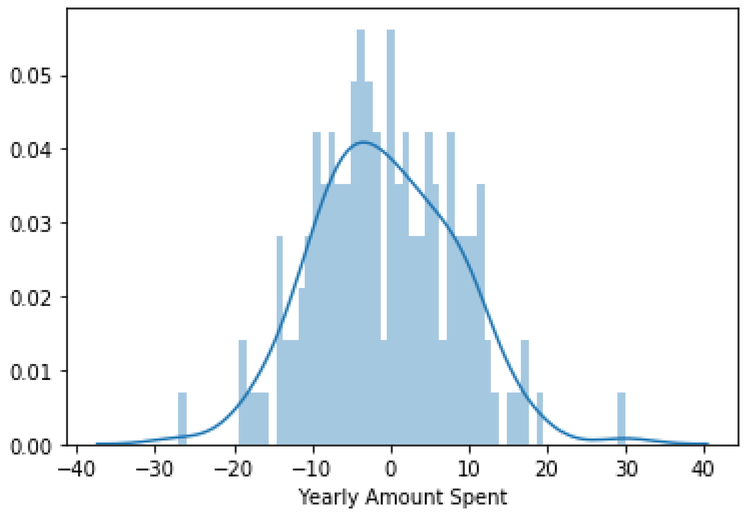
```
0.9890771231889606
```

In [32]:

```
# Have gotten a very good model with a good fit. Let's quickly explore the residuals to make sure everything was okay with our data
sns.distplot((y_test-predictions),bins=60)
```

Out[32]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x1a40a324548>
```



In [33]:

```
#looks pretty normal
```

In [39]:

```
#CONCLUSION:We still want to figure out the answer to the original question, do we focus our efforst on mobile app or website development? Or maybe that doesn't even really matter, and Membership Time is what is really important.
#Let's see if we can interpret the coefficients at all to get an idea.
cf=pd.DataFrame(lm.coef_,X.columns,columns=['Coefficients'])
cf
#this shows us that what happens on 1 unit increase of the first columns element giving an increment of dollars on column 2
```

Out[39]:

	Coefficients
Avg. Session Length	25.981550
Time on App	38.590159
Time on Website	0.190405
Length of Membership	61.279097

In [ ]:

```
# The company should focus more on their mobile app or on their website?
#Can think about it both ways, focus on website more as it needs a lot of betterment,
#or focus on the app more as people find it more suitable to shop through their phones,
#hence a number of factors within the company will be taken into account for that.
```