TINA'S Company that sells clothing online but also have in-store style and clothing advice sessions,customers come into the store,have sessions/meetings with personal stylist,then they can go home and order either on a mobile app or website for the clothes they want,the company seeks my advice to help them decide whether to focus their efforts on their mobile experience or their website.

In [1]:

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

In [2]:

```
# Getting the data
Customers=pd.read_csv(r'C:\Users\chumj\Downloads\E_Customers.csv')
```

In [3]:

```
Customers.head()
```

Out[3]:

	Email	Address	Avatar	Avg Session Length	Time on App	Time on Website	Length of Membership	Yearly Amount Spent
0	mstephenson@fernandez.com	835 Frank TunnelWrightmouth, MI 82180-9605	Violet	34.497268	12.655651	39.577668	4.082621	587.951054
1	hduke@hotmail.com	4547 Archer CommonDiazchester, CA 06566-8576	DarkGreen	31.926272	11.109461	37.268959	2.664034	392.204933
2	pallen@yahoo.com	24645 Valerie Unions Suite 582Cobbborough, DC ...	Bisque	33.000915	11.330278	37.110597	4.104543	487.547505
3	riverarebecca@gmail.com	1414 David ThroughwayPort Jason, OH 22070-1220	SaddleBrown	34.305557	13.717514	36.721283	3.120179	581.852344
4	mstephens@davidson-herman.com	14023 Rodriguez PassagePort Jacobville, PR 372...	MediumAquaMarine	33.330673	12.795189	37.536653	4.446308	599.406092

In [34]:

```
Customers.corr()['Yearly Amount Spent'].sort_values(ascending=False)
```

Out[34]:

```
Yearly Amount Spent     1.000000
Length of Membership    0.809084
Time on App             0.499328
Avg Session Length      0.355088
Time on Website        -0.002641
Name: Yearly Amount Spent, dtype: float64
```

In [4]:

```
Customers.describe()
```

Out[4]:

	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

	Avg Session Length	Time on App	Time on Website	Length of Membership	Yearly Amount Spent
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 [5]:

```
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 [6]:

```
Customers.columns
```

Out[6]:

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

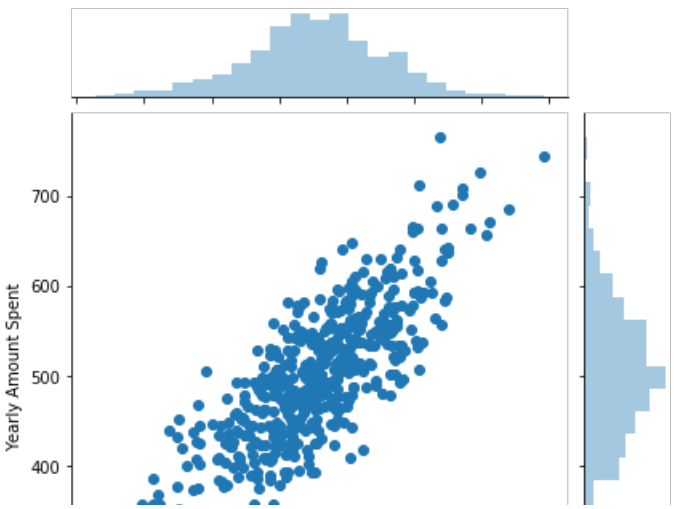
Exploratory Data Analysis Using seaborn to create a jointplot to compare the Time on Website,Time on App with Yearly Amount Spent columns. correlation make sense?

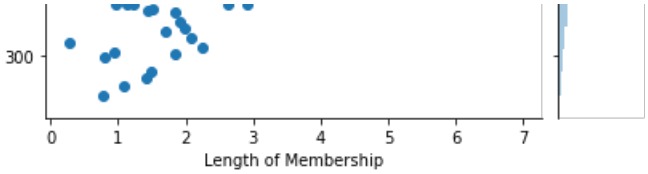
In [35]:

```
sns.jointplot(x='Length of Membership',y='Yearly Amount Spent',data=Customers)
```

Out[35]:

```
<seaborn.axisgrid.JointGrid at 0x1a8d404e988>
```



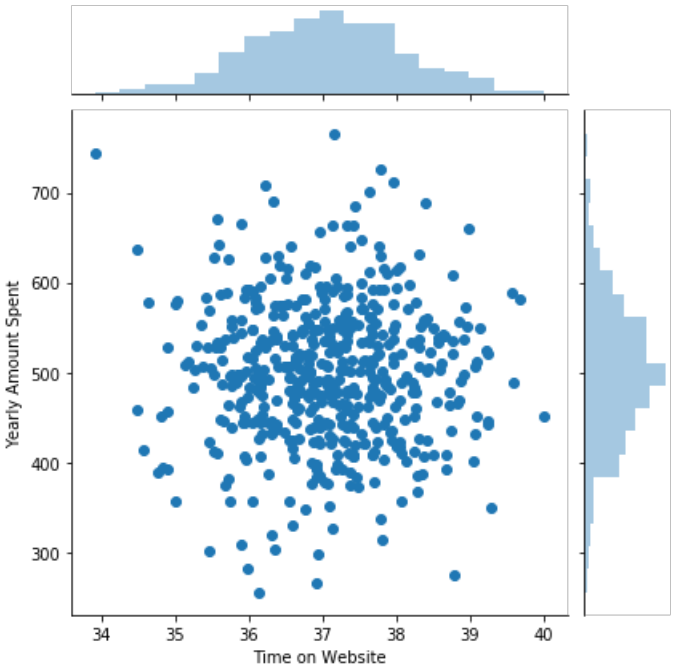


In [7]:

```
# More time on site, more money spent.
sns.jointplot(x='Time on Website',y='Yearly Amount Spent',data=Customers)
```

Out[7]:

```
<seaborn.axisgrid.JointGrid at 0x1a8cd2f5648>
```

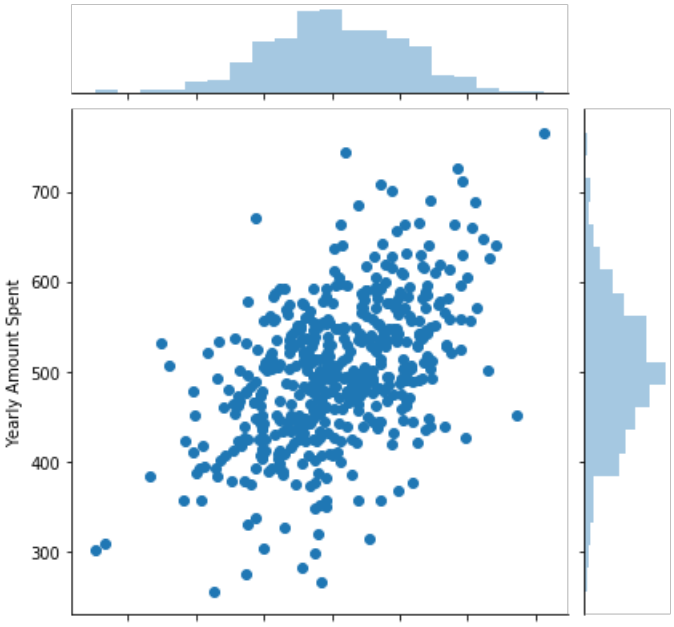


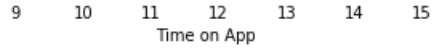
In [8]:

```
sns.jointplot(x='Time on App',y='Yearly Amount Spent',data=Customers)
```

Out[8]:

```
<seaborn.axisgrid.JointGrid at 0x1a8cdaee348>
```





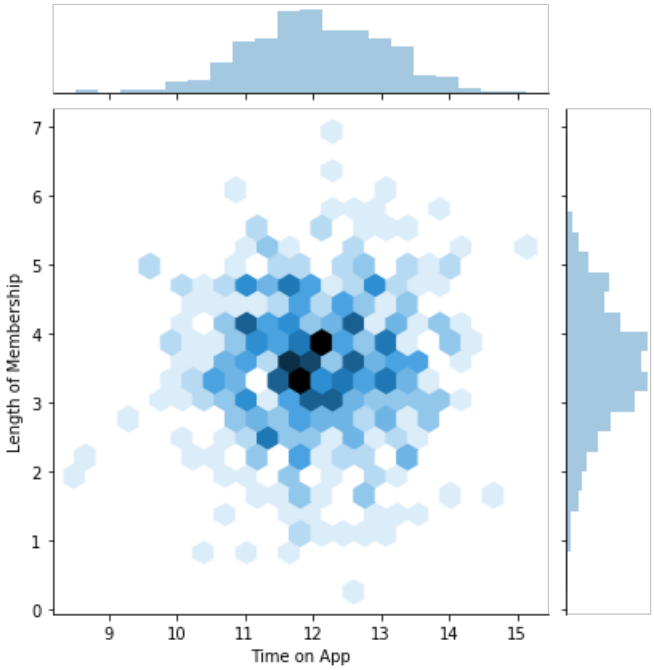
Use jointplot to create a 2D hex bin plot comparing Time on App,Time on Website with Length of Membership.

In [9]:

```
sns.jointplot(x='Time on App',y='Length of Membership',kind='hex',data=Customers)
```

Out[9]:

```
<seaborn.axisgrid.JointGrid at 0x1a8cdd71c48>
```

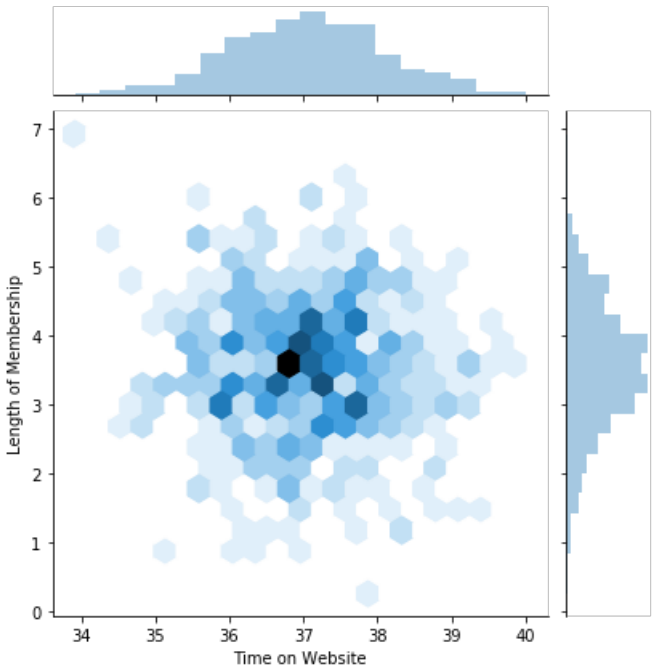


In [10]:

```
sns.jointplot(x='Time on Website',y='Length of Membership',kind='hex',data=Customers)
```

Out[10]:

```
<seaborn.axisgrid.JointGrid at 0x1a8cdeaba88>
```



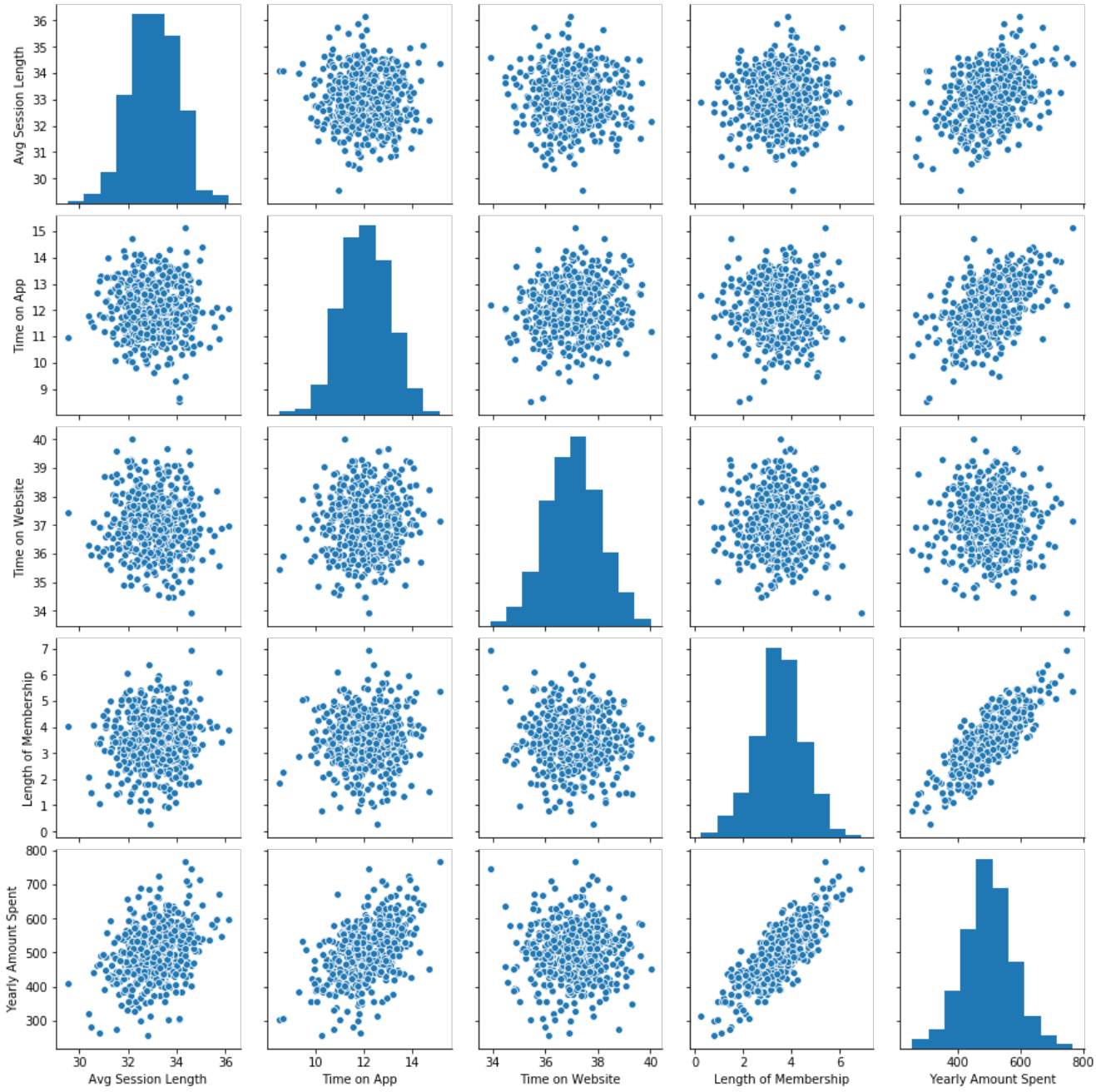
Exploring these types of relationships across the entire data set, Using pairplot.

In [11]:

```
sns.pairplot(Customers)
```

Out[11]:

```
<seaborn.axisgrid.PairGrid at 0x1a8ce0848c8>
```



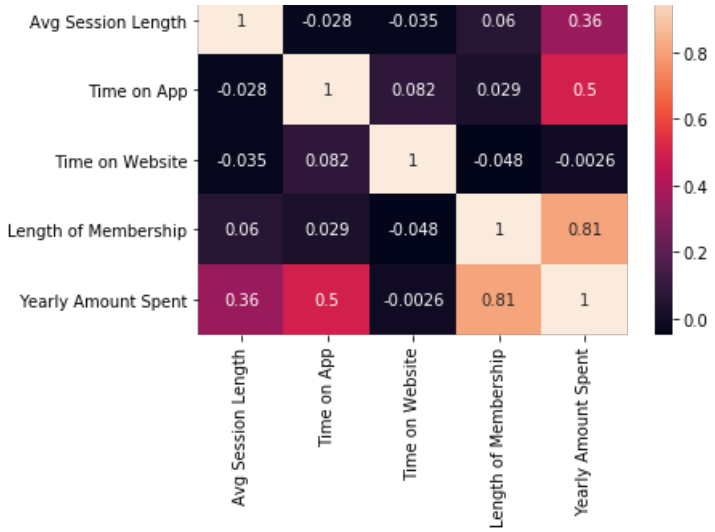
Length of Membership seems to be having some corrolation with Yearly Amount Spent than any other features. Let,s continue with further proves using Heatmap and clustermap.

In [12]:

```
sns.heatmap(Customers.corr(),annot=True)
```

Out[12]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x1a8d01e8948>
```

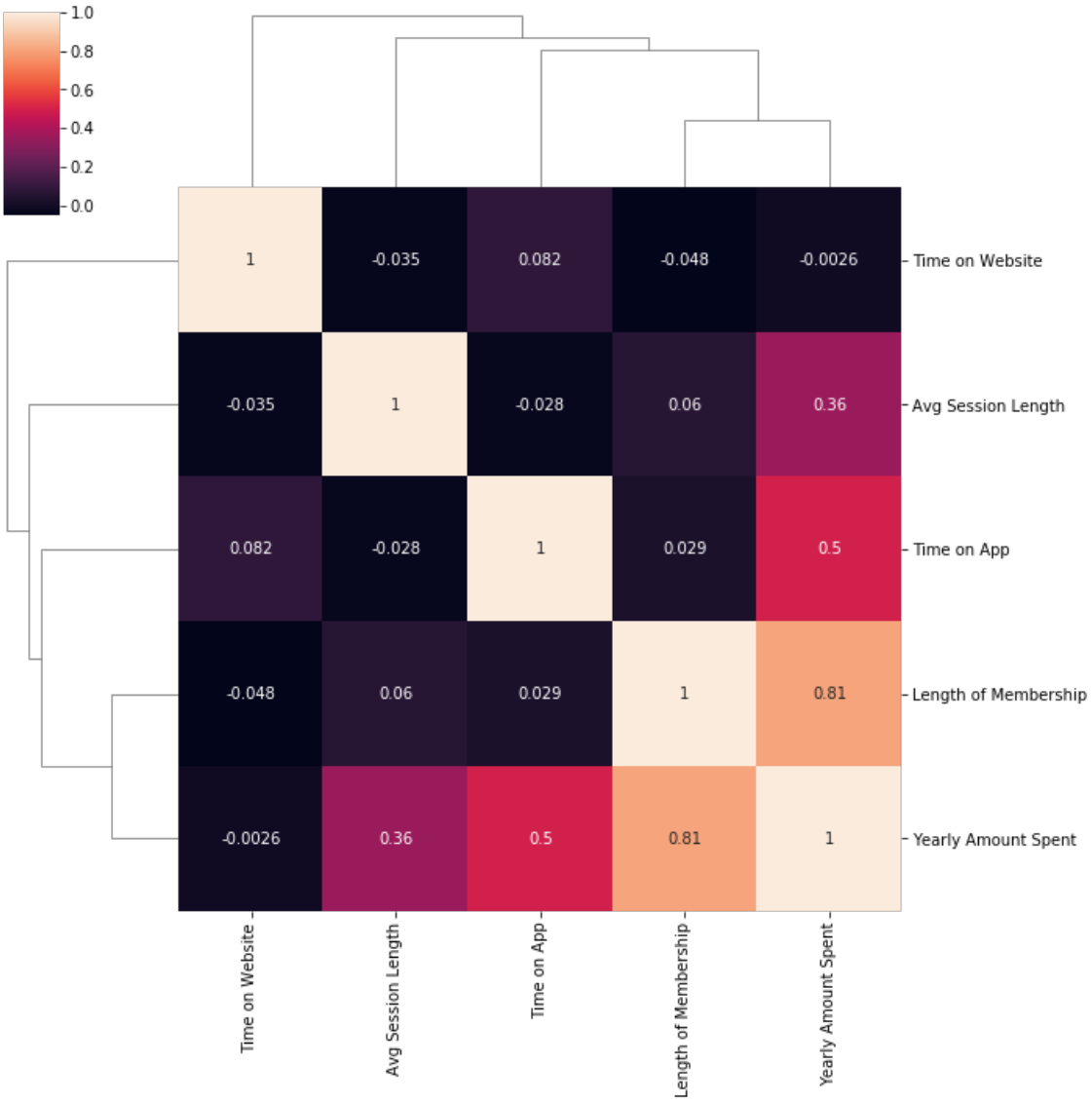


In [13]:

```
sns.clustermap(Customers.corr(),annot=True)
```

Out[13]:

```
<seaborn.matrix.ClusterGrid at 0x1a8d025bf88>
```



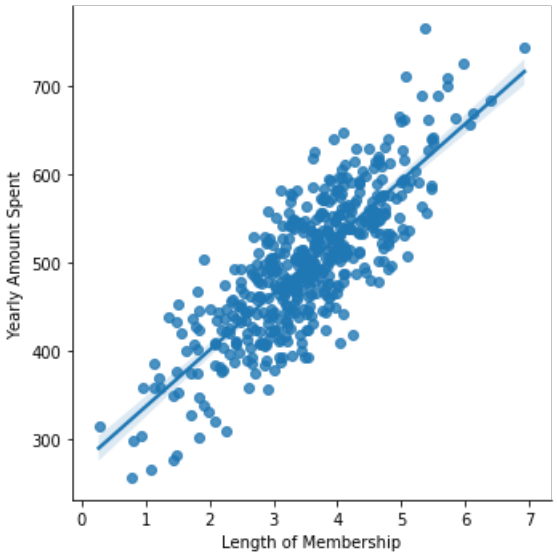
Creating a linear model plot (using seaborn's lmplot) of Yearly Amount Spent vs. Length of Membership.

In [14]:

```
sns.lmplot(x='Length of Membership',y='Yearly Amount Spent',data=Customers)
```

Out[14]:

```
<seaborn.axisgrid.FacetGrid at 0x1a8d0631308>
```



Training and Testing Data.

After exploring the data a bit, let's go ahead and split the data into training and testing sets. **Set a variable X equal to the numerical features of the customers** Set y equal to the "Yearly Amount Spent" column. **

In [16]:

```
X = Customers[['Avg Session Length', 'Time on App','Time on Website', 'Length of Membership']]
y = Customers['Yearly Amount Spent']
```

Using model_selection.train_test_split from sklearn to split the data into training and testing sets. Set test_size=0.3 and random_state=101

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)
```

Training the Model

Now its time to train our model on our training data! **Import LinearRegression from sklearn.linear_model**

In [19]:

```
from sklearn.linear_model import LinearRegression
```

In [20]:

```
lm = LinearRegression()
```

In [21]:

```
lm.fit(X_train,y_train)
```

Out[21]:

Out[21]:

```
LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None, normalize=False)
```

In [22]:

```
 #The coefficients
print('Coefficients: \n', lm.coef_)
```

```
Coefficients:
 [25.98154972 38.59015875  0.19040528 61.27909654]
```

Predicting Test Data

Now that we have fit our model, let's evaluate its performance by predicting off the test values! **Using lm.predict() to predict off the X_test set of the data.**

In [23]:

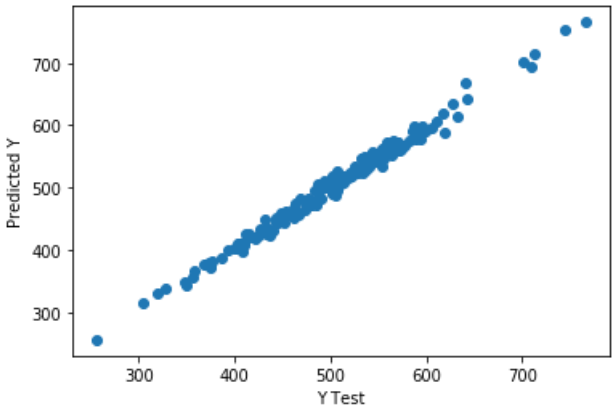
```
predictions = lm.predict( X_test)
```

In [24]:

```
plt.scatter(y_test,predictions)
plt.xlabel('Y Test')
plt.ylabel('Predicted Y')
```

Out[24]:

```
Text(0, 0.5, 'Predicted Y')
```



Evaluating the Model

Evaluating the model performance by calculating the residual sum of squares and the explained variance score (R^2). ** Calculate the Mean Absolute Error, Mean Squared Error, and the Root Mean Squared Error.

In [25]:

```
# calculate these metrics by hand!
from sklearn import metrics

print('MAE:', metrics.mean_absolute_error(y_test, predictions))
print('MSE:', metrics.mean_squared_error(y_test, predictions))
print('RMSE:', np.sqrt(metrics.mean_squared_error(y_test, predictions)))
```

```
MAE: 7.228148653430835
MSE: 79.81305165097459
RMSE: 8.93381506697864
```

In [31]:

```
y_test.mean()
```

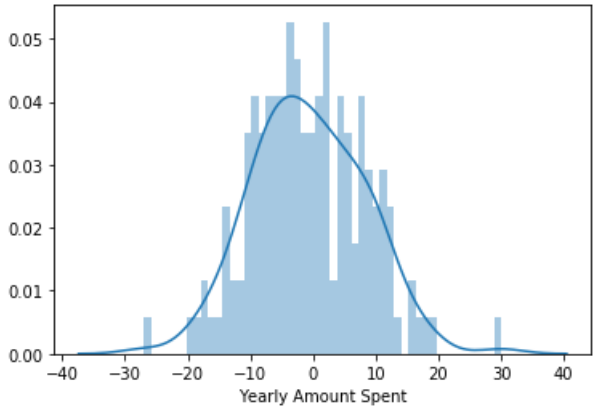
Out[31]:

```
500.6928557155851
```

Residuals

In [32]:

```
sns.distplot((y_test-predictions),bins=50);
#residuals look normally distributed,
```



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 interpret the coefficients at all to get an idea.

In [33]:

```
coeffecients = pd.DataFrame(lm.coef_,X.columns)
coeffecients.columns = ['Coeffecient']
coeffecients
```

Out[33]:

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

Interpreting the coefficients:

Holding all other features fixed, a 1 unit increase in Avg. Session Length is associated with an increase of 25.98 total dollars spent. Holding all other features fixed, a 1 unit increase in Time on App is associated with an increase of 38.59 total dollars spent. Holding all other features fixed, a 1 unit increase in Time on Website is associated with an increase of 0.19 total dollars spent. Holding all other features fixed, a 1 unit increase in Length of Membership is associated with an increase of 61.27 total dollars spent.

Is the company going to focus more on their mobile app or on their website?

This is tricky, there are two ways to think about this: Develop the Website to catch up to the performance of the mobile app, or develop the app more since that is what is working better. This sort of answer really depends on the other factors going on at the company, I would probably want to explore the relationship between Length of Membership and the App or the Website before coming to a conclusion!