In [61]:
```python
# importing required libraries
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
```

In [62]:
```python
# importing or loading the dataset
dataset = pd.read_csv('Ecommerce Customers-1.csv')
dataset.head()
```

Out[62]:

	Email	Address	Avatar	Avg. Session Length	Time on App	Time on Website	Length of Membership	Ye Am Sp
0	mstephenson@fernandez.com	835 Frank Tunnel\nWrightmouth, MI 82180-9605	Violet	34.497268	12.655651	39.577668	4.082621	587.951
1	hduke@hotmail.com	4547 Archer Common\nDiazchester, CA 06566-8576	DarkGreen	31.926272	11.109461	37.268959	2.664034	392.204
2	pallen@yahoo.com	24645 Valerie Unions Suite 582\nCobbborough, D...	Bisque	33.000915	11.330278	37.110597	4.104543	487.547
3	riverarebecca@gmail.com	1414 David Throughway\nPort Jason, OH 22070-1220	SaddleBrown	34.305557	13.717514	36.721283	3.120179	581.852
4	mstephens@davidson-herman.com	14023 Rodriguez Passage\nPort Jacobville, PR 3...	MediumAquaMarine	33.330673	12.795189	37.536653	4.446308	599.406

In [63]:
```python
X= dataset.iloc[:, 3:7].values
y= dataset.iloc[:, 7].values
```

In [64]:
```python
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.40,random_state=101)
```

In [65]:
```python
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
scaler.fit(X_train)
X_train = scaler.transform(X_train)
X_test = scaler.transform(X_test)
```

In [66]:
```python
from sklearn.neighbors import KNeighborsRegressor
clf=KNeighborsRegressor()
clf.fit(X_train,y_train)
```

Out[66]:
```
KNeighborsRegressor()
```

In [67]:
```python
y_pred=clf.predict(X_test)
```

In [68]:
```python
y_pred
```

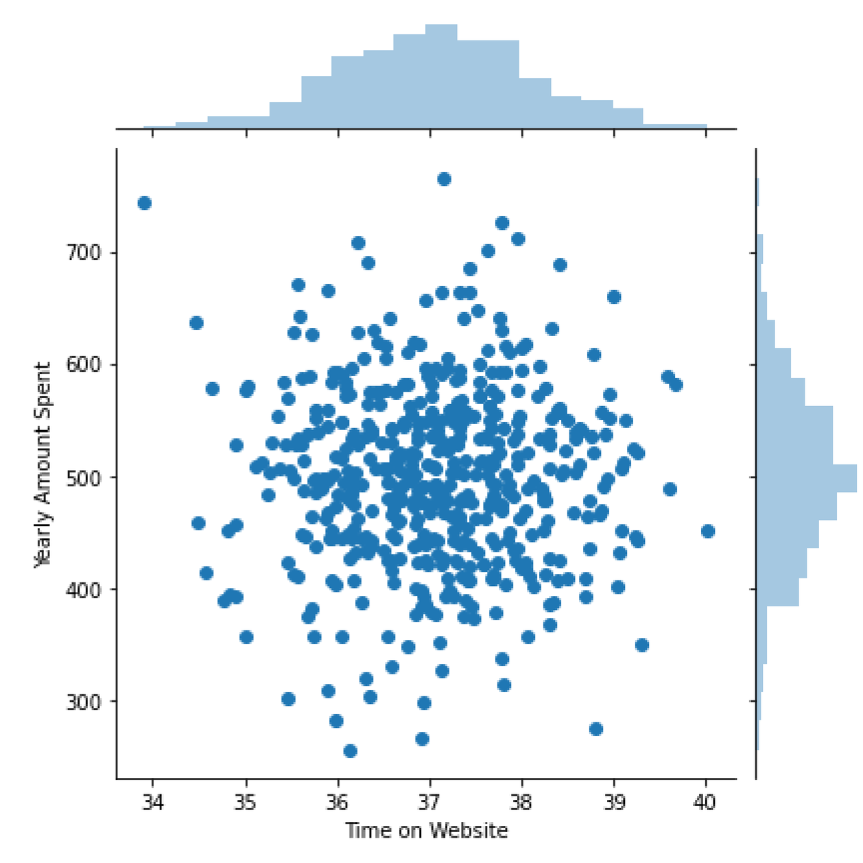
Out[68]:
```
array([463.3731537 , 409.4711749 , 415.66092117, 588.699207  ,
       561.80912834, 563.48649885, 551.67777538, 616.55313974,
       448.38338481, 553.46088116, 363.73978616, 484.44720428,
       560.68605448, 421.9139625 , 660.00122169, 542.12559137,
       642.48356777, 505.22718658, 575.41319641, 544.11798911,
       446.96558827, 550.1634569 , 432.41542133, 478.48846246,
       555.6131385 , 417.3319508 , 525.28924372, 397.69199344,
       532.33573827, 448.66044324, 612.03563187, 604.80679617,
       511.38014422, 566.14320112, 503.64476768, 565.81387955,
       449.30432771, 468.38160791, 460.54847491, 481.5608318 ,
       567.92708684, 445.34101784, 602.4368743 , 506.13412775,
       489.30350008, 498.68631416, 562.19566718, 524.54955629,
       422.83651503, 470.32490275, 502.37039727, 538.44774817,
       343.64762242, 519.64346103, 555.88397297, 364.87284439,
       496.80608022, 407.39107535, 482.52922679, 485.89756729,
       526.01923517, 590.73125448, 630.65126607, 533.45515251,
       497.22757343, 406.64098764, 414.72891876, 492.37645315,
       628.05837262, 515.99754818, 552.34827616, 397.97078564,
       534.81798945, 431.70197852, 492.8710312 , 493.24162283,
       478.58766668, 440.63159143, 542.44770989, 622.84091305,
       547.05314643, 589.98006405, 497.29475843, 562.21820466,
       575.05692632, 486.80831392, 523.18000212, 474.51740316,
       444.64365211, 474.25886893, 491.33897022, 562.11959809,
       471.75411804, 586.91569933, 497.11260625, 567.52030982,
       496.79924069, 482.02504887, 454.37039863, 522.19338158,
       418.26101059, 412.90249241, 521.76982071, 520.17656779,
       492.11587037, 498.82630137, 515.22031032, 454.31275469,
       521.06752834, 472.89741895, 537.53648377, 366.46568984,
       422.16757594, 597.19550525, 568.33947035, 542.67388964,
       548.89493156, 491.23555575, 496.65973652, 382.6507298 ,
       404.89014731, 639.07115525, 505.18370246, 529.20120211,
       572.95467287, 447.46803039, 571.40483682, 495.13380714,
       468.51873681, 440.16187884, 360.82167716, 444.46733286,
       570.61561909, 445.75742675, 451.45869514, 423.3781789 ,
       417.28640285, 495.55213934, 568.7349859 , 431.85550993,
       528.51962658, 520.42284972, 535.22875609, 472.78208842,
       580.09096272, 485.0241122 , 497.588915  , 480.53437843,
       436.51314581, 390.88232783, 436.0616121 , 510.49436735,
       551.83831252, 472.06412467, 543.03801996, 509.85046316,
       471.23956403, 542.419367  , 535.36311349, 448.15472962,
       451.0980979 , 492.58772577, 573.46959904, 565.39631409,
       442.35357601, 539.07892654, 407.21878423, 510.72858481,
       521.06752834, 498.62156661, 450.74629863, 459.73102796,
       463.1287343 , 526.19755058, 544.91080286, 569.11225021,
       552.30398315, 490.34911779, 553.48760049, 517.05889198,
       503.34855452, 518.69963484, 534.78236323, 547.87961684,
       581.51905523, 420.28856154, 522.88945454, 632.4333174 ,
       454.85909013, 495.61939826, 400.9192174 , 439.94836922,
       517.98301862, 471.23956403, 559.03515857, 658.59863455,
       520.94740444, 489.92347501, 496.57153091, 553.99654696])
```

In [69]:
```python
np.savetxt("prediction_KNNREG_Ecommerce.csv",y_pred)
```

In [70]:
```python
import seaborn as sns
```

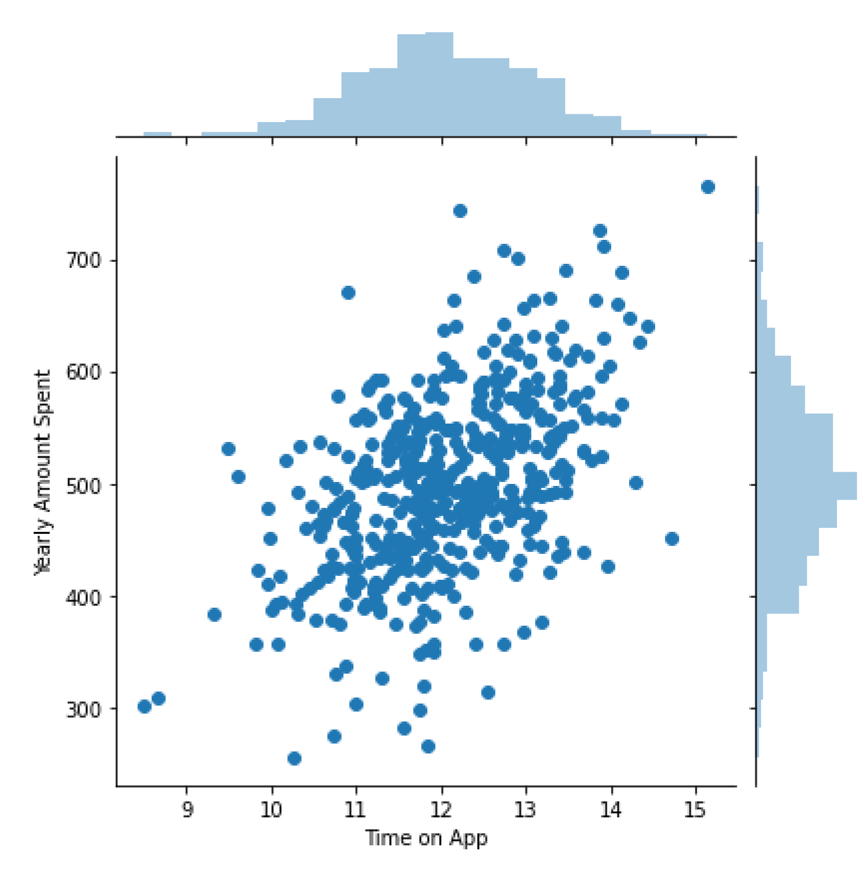
In [71]:
```python
sns.jointplot(x='Time on Website',y='Yearly Amount Spent',data=dataset)
```

Out[71]:
```
<seaborn.axisgrid.JointGrid at 0x1d2a18ef700>
```



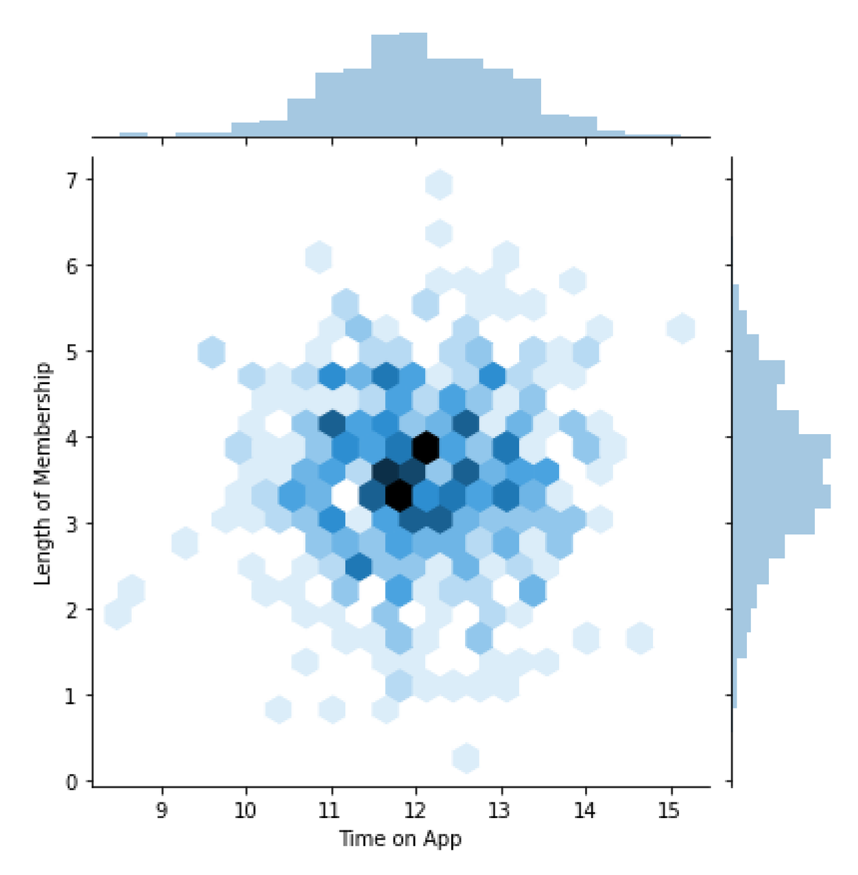
In [72]:
```python
sns.jointplot(x='Time on App',y='Yearly Amount Spent',data=dataset)
```

Out[72]:
```
<seaborn.axisgrid.JointGrid at 0x1d2a1a221c0>
```



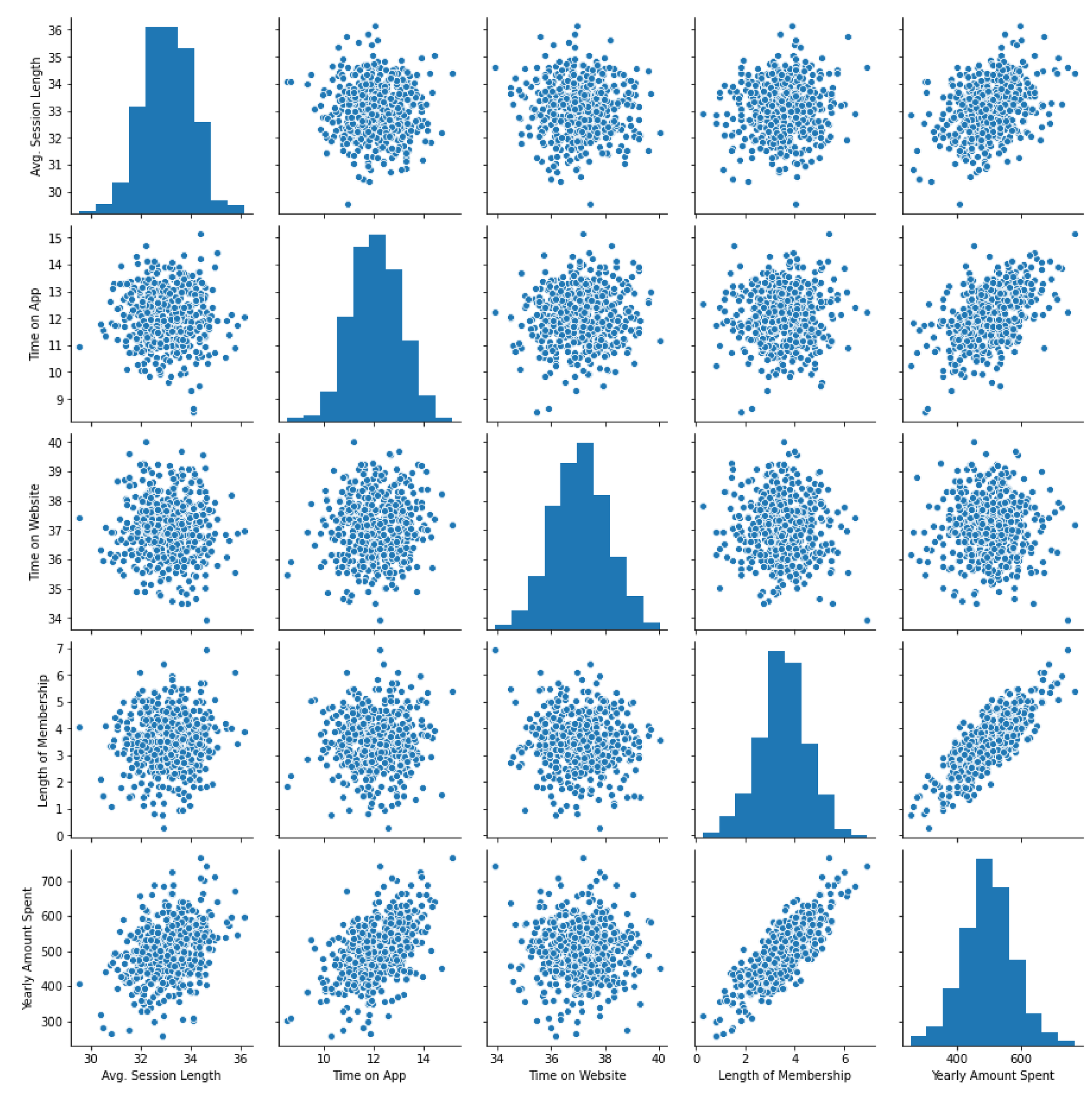
In [73]:
```python
sns.jointplot(x='Time on App',y='Length of Membership',kind='hex',data=dataset)
```

Out[73]:
```
<seaborn.axisgrid.JointGrid at 0x1d2a1ba15e0>
```



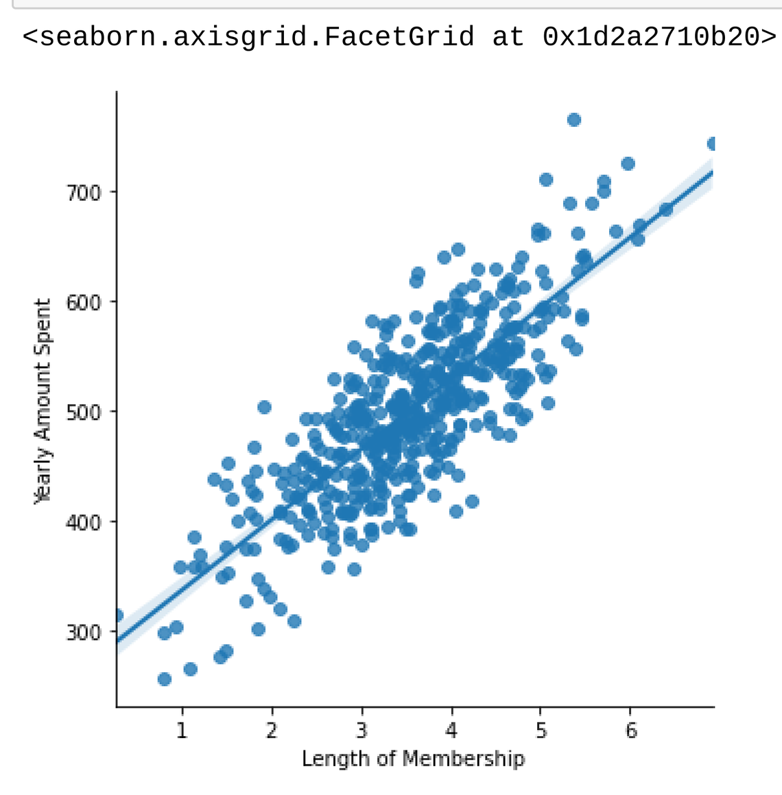
In [74]:
```python
sns.pairplot(dataset)
```

Out[74]:
```
<seaborn.axisgrid.PairGrid at 0x1d2a1cd0d30>
```



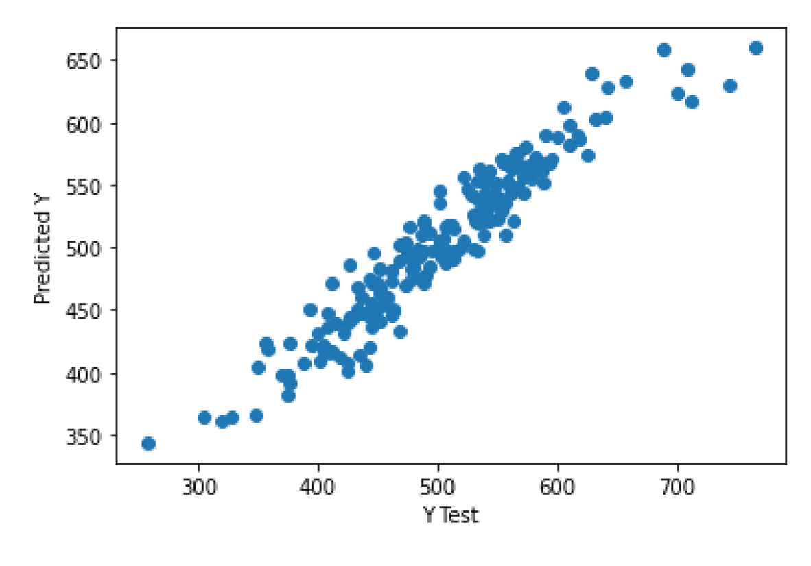
In [75]:
```python
sns.lmplot(x='Length of Membership',y='Yearly Amount Spent',data=dataset)
```

Out[75]:
```
<seaborn.axisgrid.FacetGrid at 0x1d2a2710b20>
```

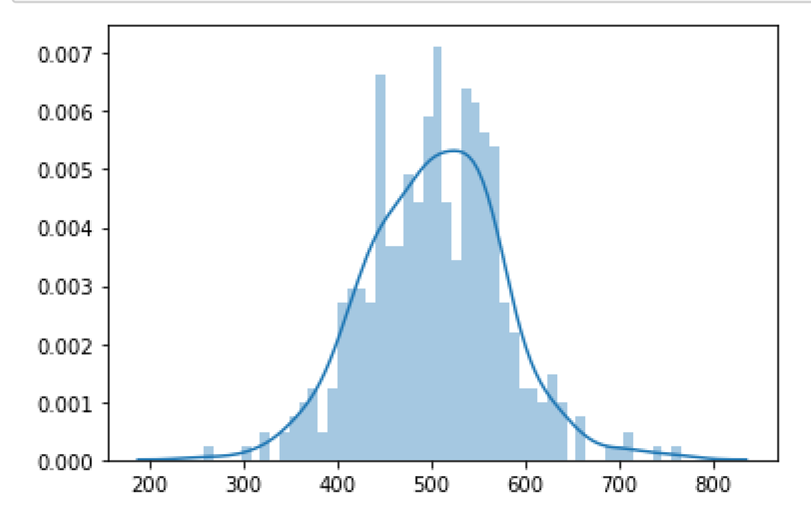


In [76]:
```python
plt.scatter(y_test,y_pred)
plt.xlabel('Y Test')
plt.ylabel('Predicted Y')
```

Out[76]:
```
Text(0, 0.5, 'Predicted Y')
```



In [77]:
```python
sns.distplot((y_test,y_pred),bins=50);
```



In [78]:
```python
# calculate these metrics by hand!
from sklearn import metrics

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

```
MAE: 20.711497272861326
MSE: 770.4194450042638
RMSE: 27.75643069640374
```

In [79]:
```python
clf.score(X_test,y_test)
```

Out[79]:
```
0.8809974473899255
```