Linear Regression Project

I have just got a contract work with an Ecommerce company named Shine cosmetics based in New York City that sells cosmetics online but they also have in-store products and cosmethology advice sessions. Customers come in to the store, have meetings with a personal cosmethologist, then they can go home and order either on a mobile app or website for the cosmetics they want.

Shine cosmetics is trying to decide whether to focus their efforts on their mobile app experience or their website. They've hired me on contract to help them figure it out! Let's get started!

Disclaimer: Just follow the steps below to analyze the customer data (it's a fake data, I didn't give you real credit card numbers or emails. This not a financial advice, it's just a dummy dataset to show case my skills).

Imports

Import pandas, numpy, matplotlib,and seaborn. (You'll import sklearn as you need it.)

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

Data Wrangling

I will be making use of the folowing data cleaning steps.

- Load the dataset using Pandas and read a few lines: This is to have an overview of the dataset.
- Get the information about the dataset: This is to check if there are any errors in the data types.
- Get the shape of the dataset: This is to know the number of rows and columns in the dataset.
- Compute general statistics for the dataset
- Check for duplicated rows
- Check for empyty columns

In [2]:
```
# Load the dataset
clients = pd.read_csv('Ecommerce Customers')
```

In [3]:
```
# check the dataset
clients.head()
```

Out[3]:

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

In [4]:
```
# Information about the  dataset
clients.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 [5]:
```
# check if there is null value in the data
clients.isnull().sum()
```

Out[5]:
```
Email                   0
Address                 0
Avatar                  0
Avg. Session Length     0
Time on App             0
Time on Website         0
Length of Membership    0
Yearly Amount Spent     0
dtype: int64
```

There are no null values in the dataset

In [6]:
```
# get the dimension of the dataset
clients.shape
```

Out[6]:
```
(500, 8)
```

The total number of samples(i.e number of customers) present are 500 and there are 8 columns.

In [7]:
```
# perform general statistics for the data
clients.describe()
```

Out[7]:

	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

Data accessment

After accessing the data, the data is ready cleaned

Exploratory Data Analysis (EDA)

in this section i'll be exploring the data with the following steps:

1. compare the time spent on website to the yearly amount spent columns to check if the correlation make sense
2. compare the time spent on mobile app to the yearly amount spent columns to check if the correlation make sense
3. Use jointplot to create a 2D hex bin plot comparing Time on App and Length of Membership.
4. explore relationships across the entire data set
5. Create a linear model plot of Yearly Amount Spent vs. Length of Membership.

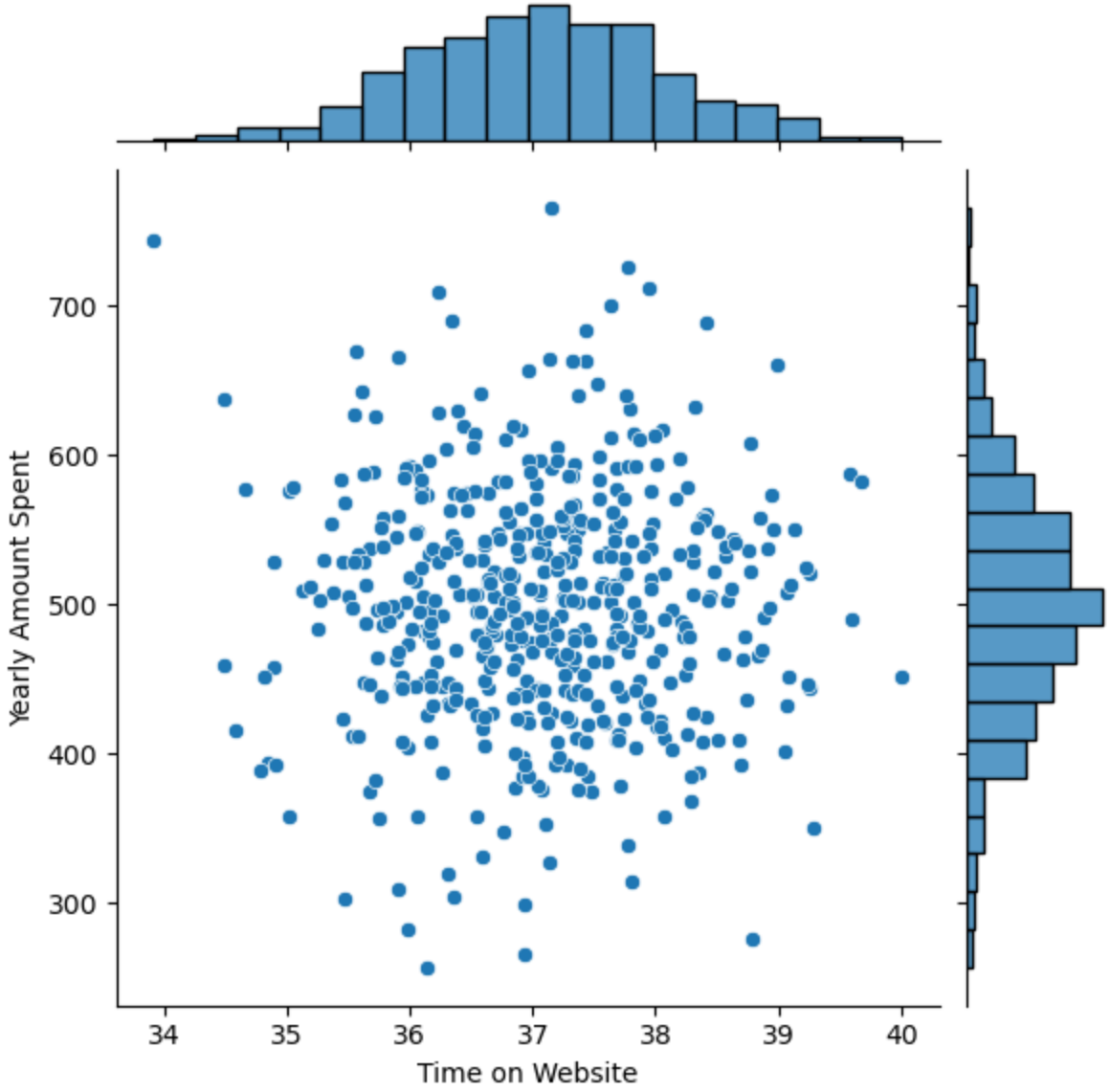
For the rest of the exercise we'll only be using the numerical data of the csv file.

Let's explore the data!

In [8]:
```
# compare the Time on website and the Yearly amount spent columns in order to check if t

sns.jointplot(clients, x='Time on Website', y='Yearly Amount Spent')
```

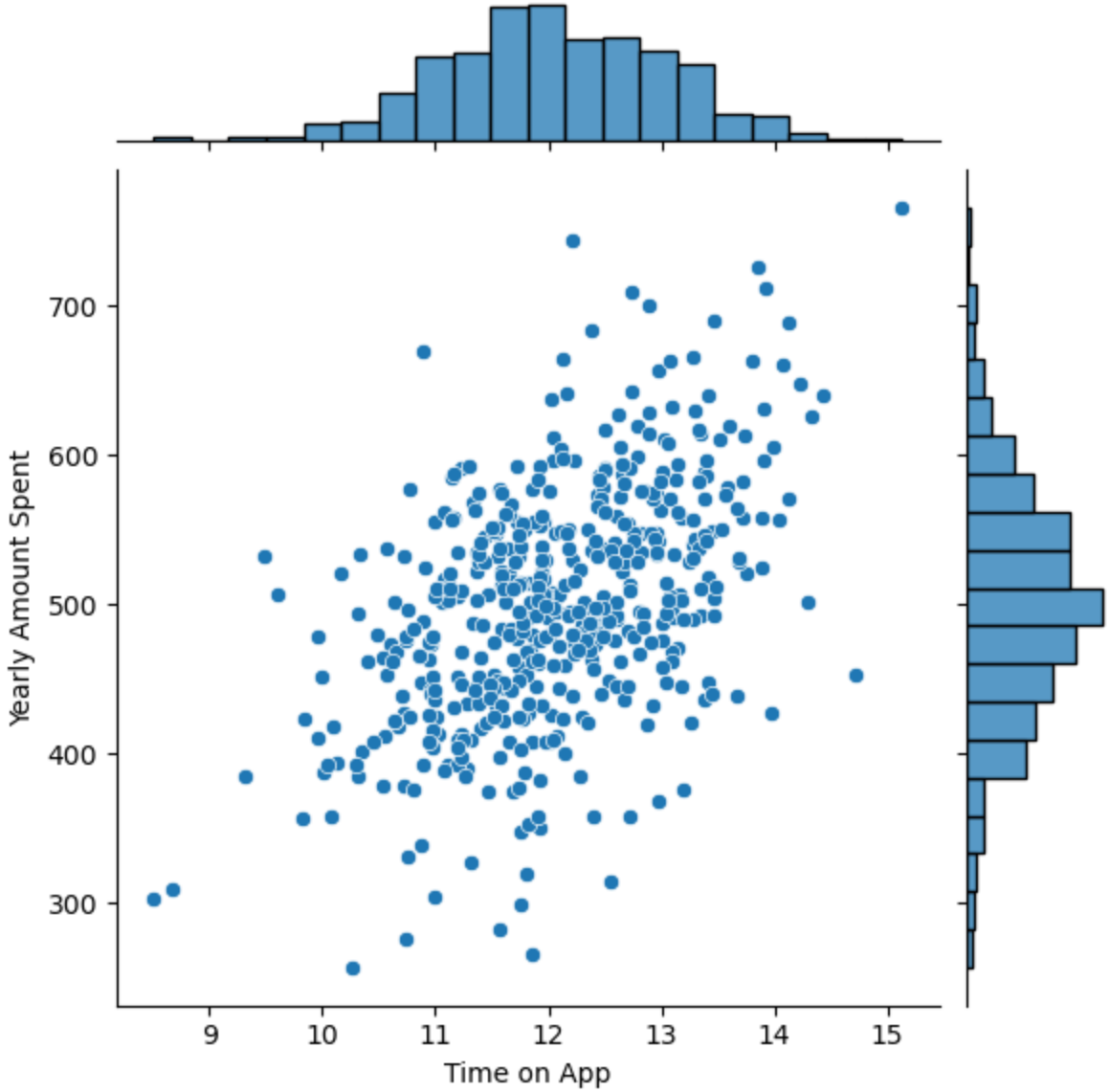
Out[8]:
```
<seaborn.axisgrid.JointGrid at 0x222c3248510>
```



In [9]:
```
# compare the Time on mobile app and the Yearly amount spent columns in order to check i

sns.jointplot(clients, x='Time on App', y='Yearly Amount Spent', palette='tab10')
```

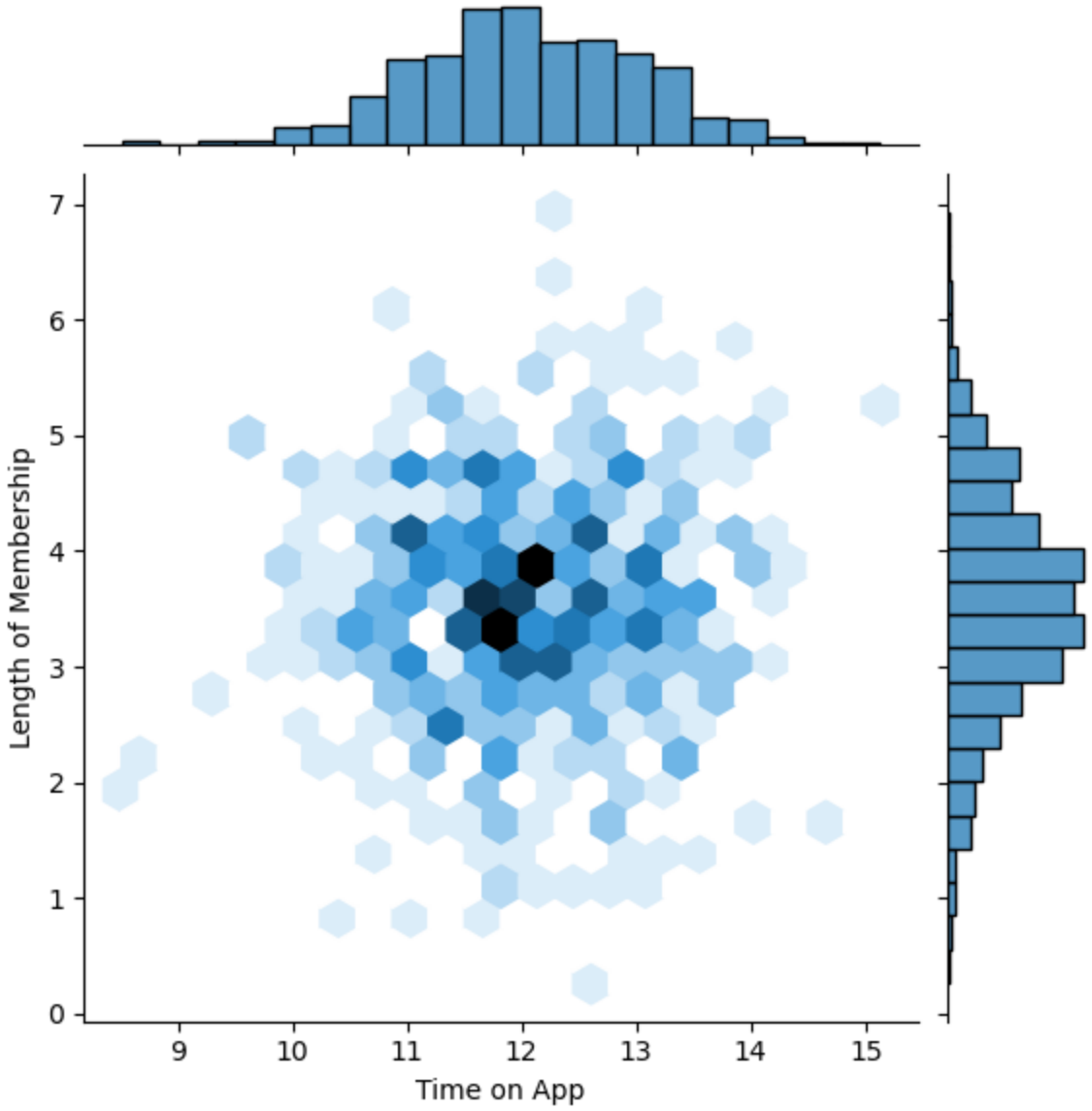
Out[9]:
```
<seaborn.axisgrid.JointGrid at 0x222c33f8d50>
```



The more the time spent on the mobile app the more the money

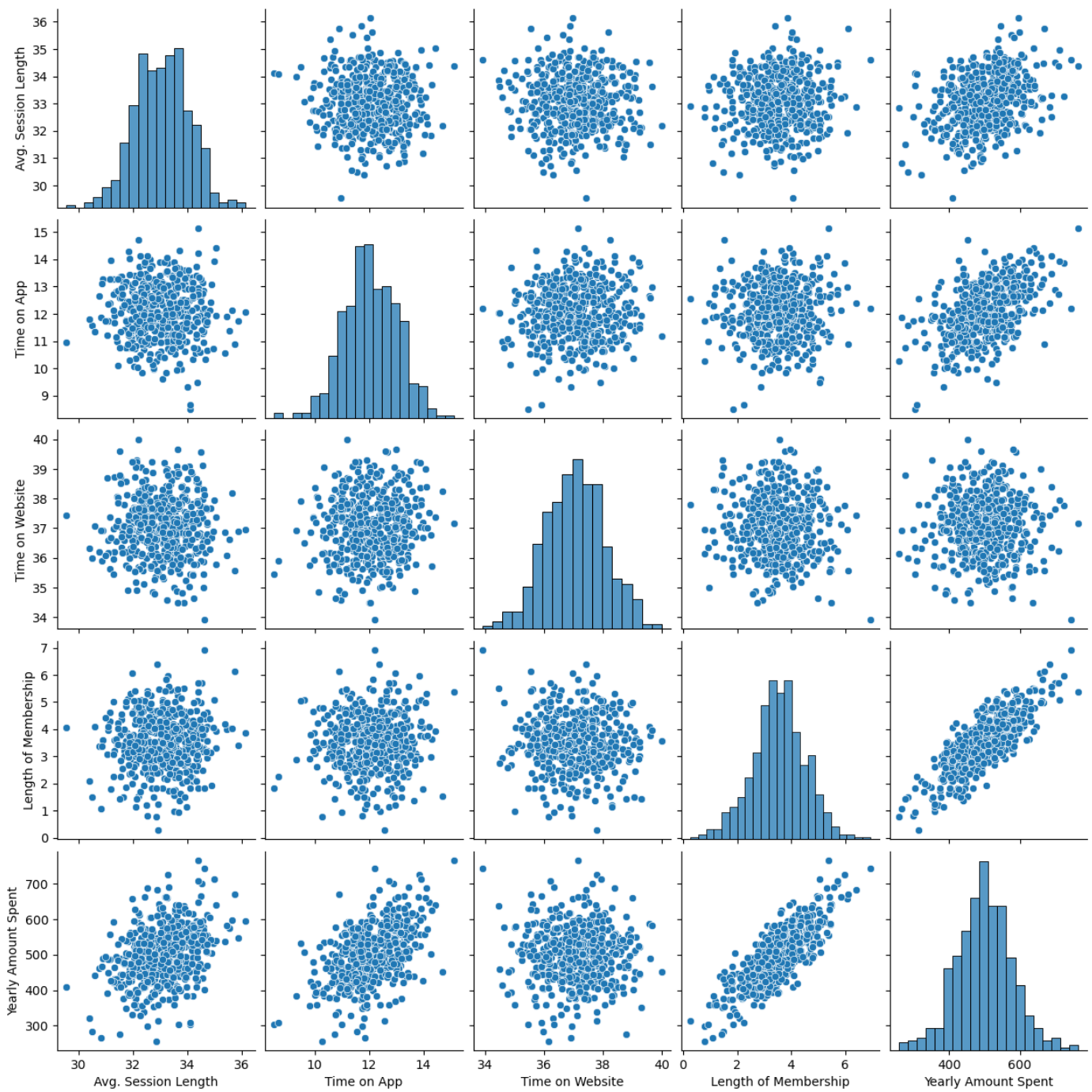
In [10]:
```
# Use jointplot to create a 2D hex bin plot comparing Time on App and Length of Membersh
sns.jointplot(clients, x='Time on App', y='Length of Membership', kind='hex', palette='A
```

Out[10]:
```
<seaborn.axisgrid.JointGrid at 0x222c40604d0>
```



In [11]:
```
# explore relationships across the entire data set
sns.pairplot(clients)
```

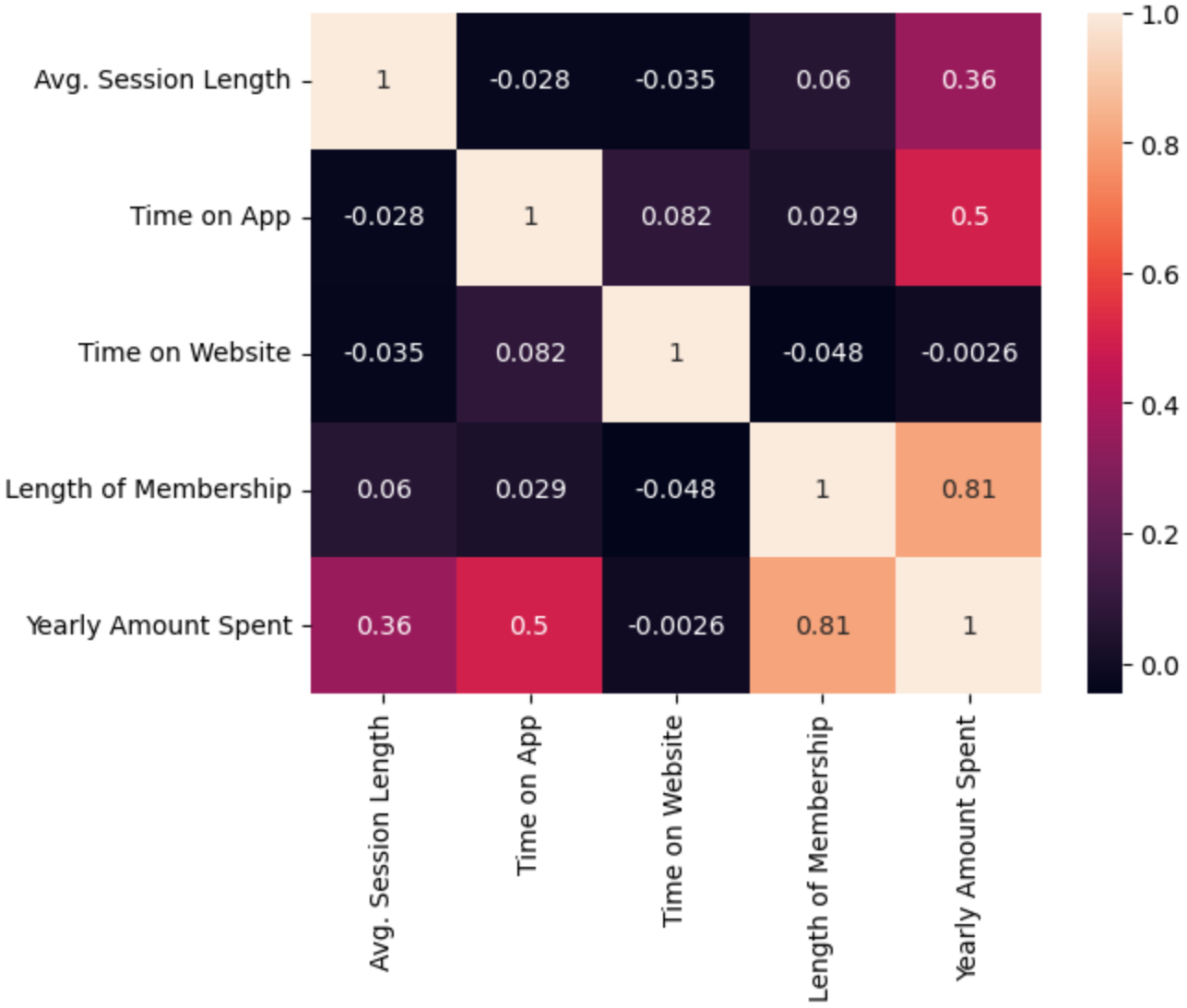
Out[11]:
```
<seaborn.axisgrid.PairGrid at 0x222c40b7f10>
```



In [12]:
```
# Display the strengh of the correllation
sns.heatmap(clients.corr(), annot=True)
```

```
C:\Users\USER\AppData\Local\Temp\ipykernel_13720\3791953507.py:2: FutureWarning: The default value of numeric_only in DataFrame.corr is deprecated. In a future version, it will default to False. Select only valid columns or specify the value of numeric_only to silence this warning.
  sns.heatmap(clients.corr(), annot=True)
```

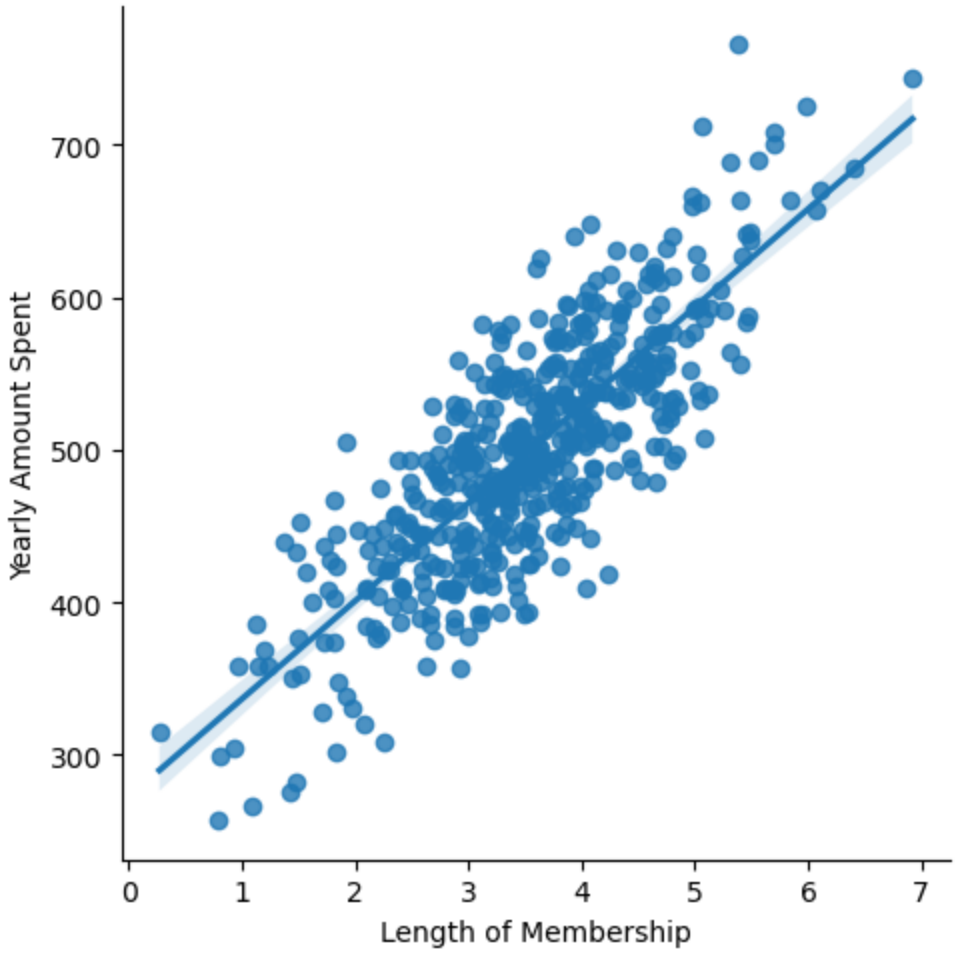
Out[12]:
```
<Axes: >
```



Based off this plot what looks to be the most correlated feature with Yearly Amount Spent is the Lenth of Membership

In [13]:
```
# Create a linear model plot of Yearly Amount Spent vs. Length of Membership.
sns.lmplot(clients, x='Length of Membership', y='Yearly Amount Spent')
```

Out[13]:
```
<seaborn.axisgrid.FacetGrid at 0x222c5e9d550>
```



Training and Testing Data

Now that the data as been explored, I'll go ahead and split the data into training and testing sets.

- Set a variable X equal to the numerical features of the clients and
- variable y equal to the "Yearly Amount Spent" column.

In [14]:
```
clients.columns
```

Out[14]:
```
Index(['Email', 'Address', 'Avatar', 'Avg. Session Length', 'Time on App',
       'Time on Website', 'Length of Membership', 'Yearly Amount Spent'],
      dtype='object')
```

In [15]:
```
x = clients[['Avg. Session Length', 'Time on App',
       'Time on Website', 'Length of Membership']]

y = clients['Yearly Amount Spent']
```

Use model_selection.train_test_split to split the data into training and testing sets. Set test_size=0.3 and random_state=0

In [16]:
```
from sklearn.model_selection import train_test_split
```

In [17]:
```
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.3, random_state=0)
```

Training the Model

Now its time to train the model on our training data!

Import LinearRegression from sklearn.linear_model

In [18]:
```
from sklearn.linear_model import LinearRegression
```

In [19]:
```
# create instance of LinearRegression as lm
lm = LinearRegression()
```

In [20]:
```
# Train/fit lm on the training data.
lm.fit(x_train, y_train)
```

Out[20]:
```
▼ LinearRegression
LinearRegression()
```

In [21]:
```
# Print out the coefficients of the model

coefficients = pd.DataFrame(lm.coef_, x.columns, columns=['Coefficients'])
coefficients
```

Out[21]:

	Coefficients
Avg. Session Length	25.767530
Time on App	38.800394
Time on Website	-0.018041
Length of Membership	61.852568

Predicting Test Data

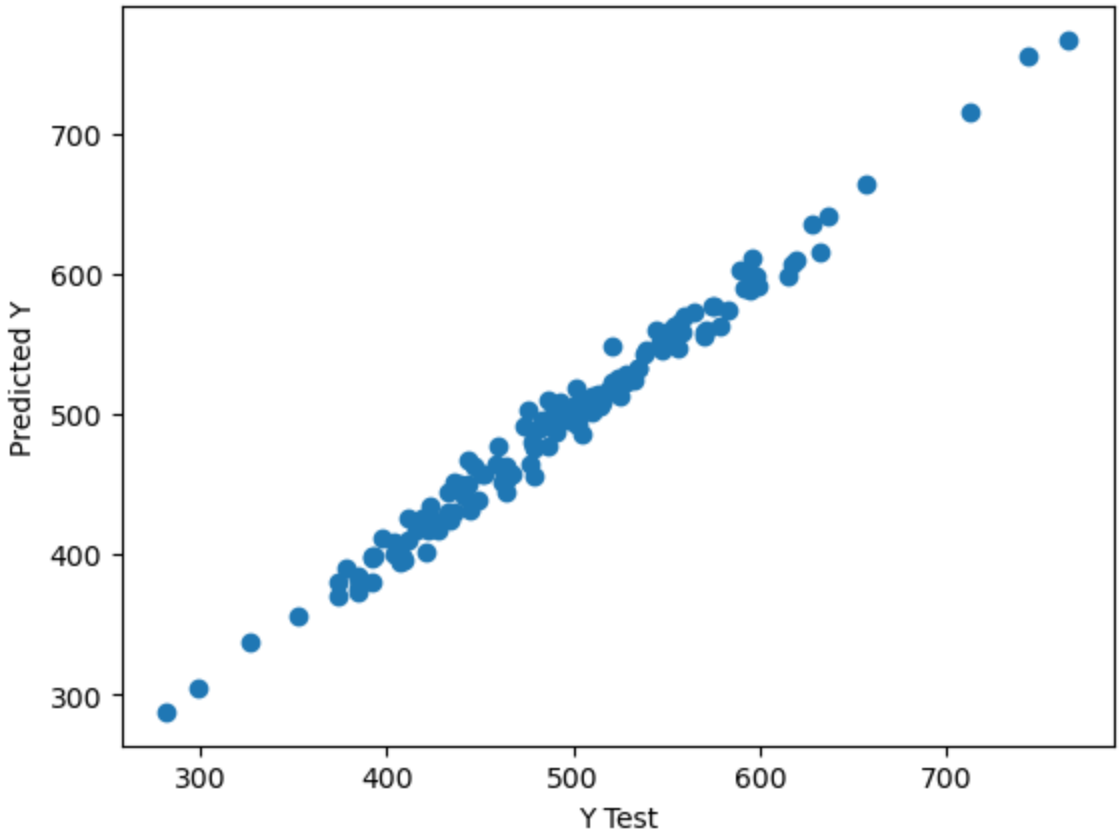
Now that the model has been trained/fit, I'll evaluate its performance by predicting off the test values!

In [22]:
```
# predict off the x_test set of the data
predictions = lm.predict(x_test)
```

Here I'll create a scatterplot of the real test values VS the predicted values.

In [23]:
```
plt.scatter(y_test, predictions)
plt.xlabel("Y Test")
plt.ylabel("Predicted Y")
```

Out[23]:
```
Text(0, 0.5, 'Predicted Y')
```



Evaluating the Model

In this section, I'll evaluate the model performance by calculating the residual sum of squares and the explained variance score (R^2).

** I'll Calculate the followings:

- Mean Absolute Error (MAE),
- Mean Squared Error (MSE), and the
- Root Mean Squared Error (RMSE).

In [24]:
```
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: 7.851377170861467
MSE: 94.55779479273308
RMSE: 9.724083236620976
```

In [25]:
```
metrics.explained_variance_score(y_test,predictions)
```

Out[25]:
```
0.984939275525299
```

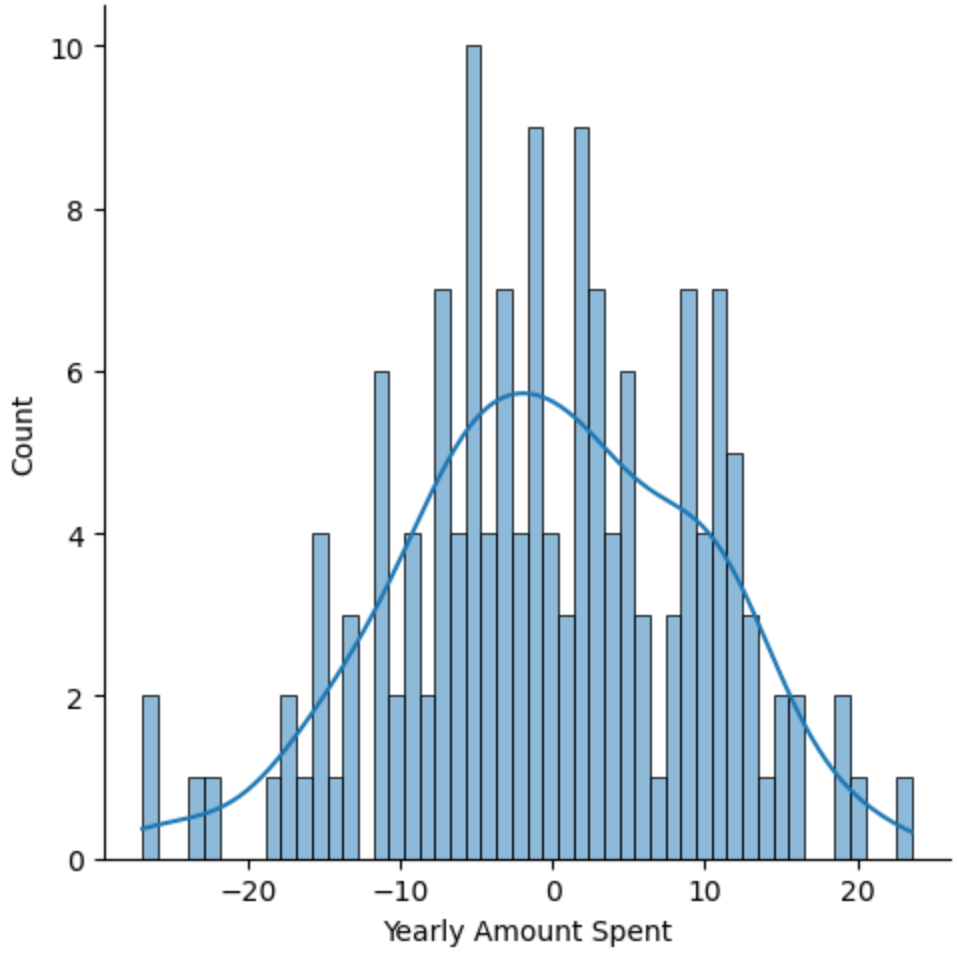
Residuals

In this section, I'll explore the residuals to make sure everything was okay with our data.

I'll Plot a histogram of the residuals and make sure it looks normally distributed. Use either seaborn distplot,

In [26]:
```
sns.displot((y_test - predictions), bins=50, kde=True)
```

Out[26]:
```
<seaborn.axisgrid.FacetGrid at 0x222bdfe3fd0>
```



Conclusion and Recommendation

still want to figure out the answer to the original question, should Shine cosmetic focus their efforts on their mobile app experience or their website? Or maybe that doesn't even really matter, and Membership Time is what is really important.

I will Recreate the dataframe below to interpret the coefficient.

In [27]:
```
coefficients
```

Out[27]:

	Coefficients
Avg. Session Length	25.767530
Time on App	38.800394
Time on Website	-0.018041
Length of Membership	61.852568

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**.

focus more on their mobile app or on their website?

- 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.
- The company should give discount to best partronizing customers in their subsequent partronage

In [ ]:

In [ ]: