In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from math import sqrt
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error
import seaborn as sns
```

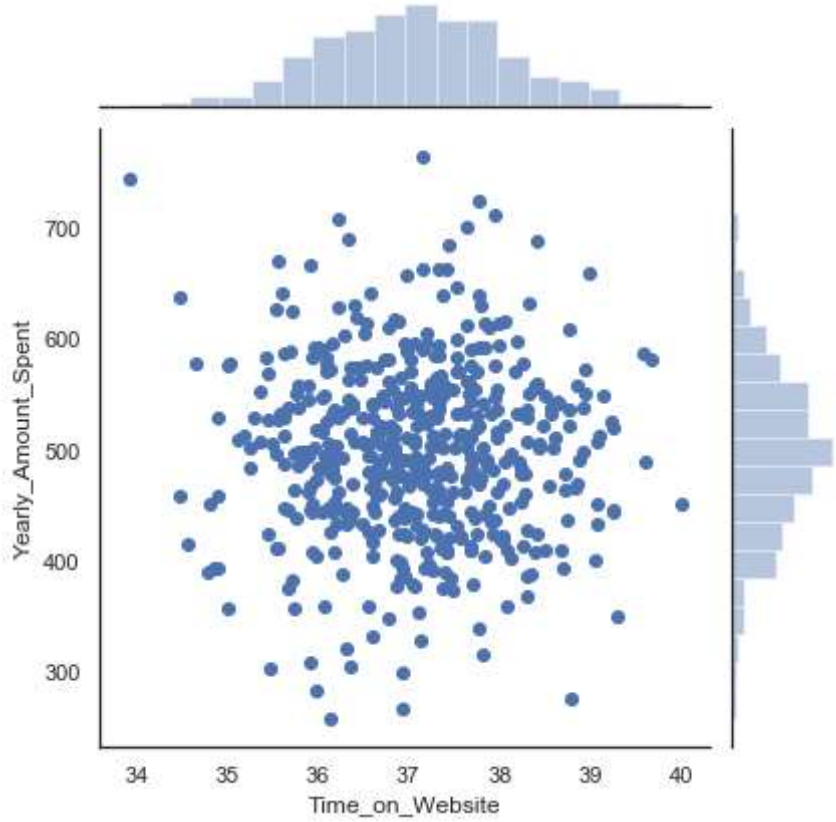
In [2]:
```
df_fyntra = pd.read_csv("FyntraCustomerData.csv")
df_fyntra.head()
```

Out[2]:

	Email	Address	Avatar	Avg_Session_Length	Ti
0	mstephenson@fernandez.com	835 Frank Tunnel\nWrightmouth, MI 82180-9605	Violet	34.497268	
1	hduke@hotmail.com	4547 Archer Common\nDiazchester, CA 06566-8576	DarkGreen	31.926272	
2	pallen@yahoo.com	24645 Valerie Unions Suite 582\nCobbborough, D...	Bisque	33.000915	
3	riverarebecca@gmail.com	1414 David Throughway\nPort Jason, OH 22070-1220	SaddleBrown	34.305557	
4	mstephens@davidson-herman.com	14023 Rodriguez Passage\nPort Jacobville, PR 3...	MediumAquaMarine	33.330673	

In [9]:
```
# 1.Compute --Use seaborn to create a jointplot to compare the Time on Website
and Yearly Amount Spent columns.
df_Time_web_amount = df_fyntra.filter(["Time_on_Website", "Yearly_Amount_Spen
t"])
sns.set(style='white', color_codes=True)
sns.jointplot(x = 'Time_on_Website', y = 'Yearly_Amount_Spent', data=df_Time_w
eb_amount)
```

Out[9]: <seaborn.axisgrid.JointGrid at 0x20c69f935c8>

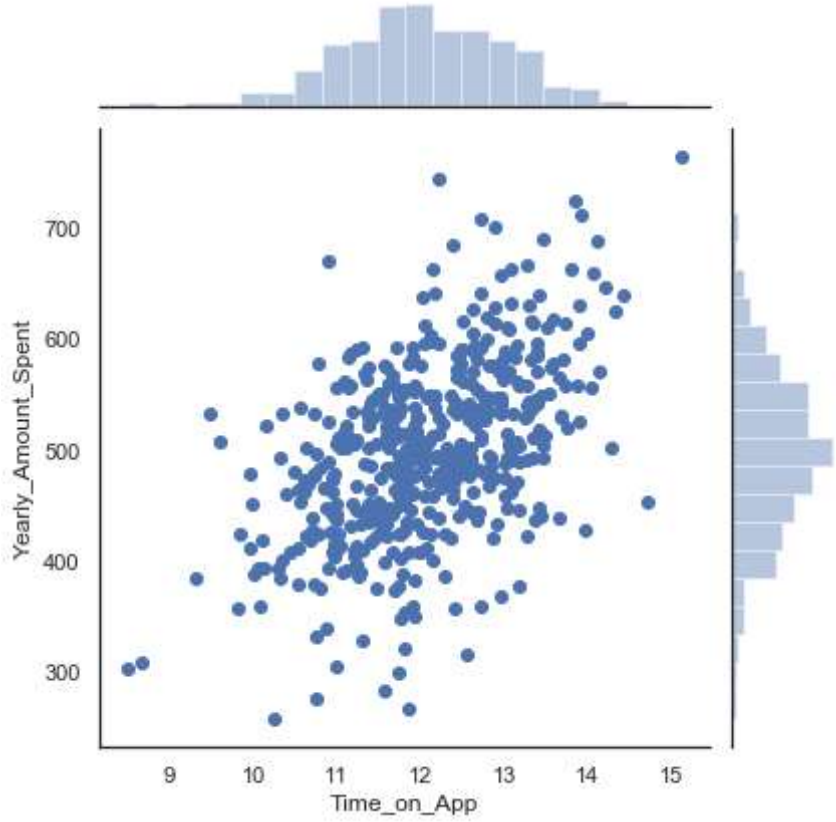


In [12]:
```
# Is there a correlation? - Both seems to have a strong correlation
Coff_Time_Amount = df_Time_web_amount.corr()
sns.heatmap(Coff_Time_Amount)
```

Out[12]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x20c6a5a3e08>
```

In [13]:
```
# 2. Compute –Do the same as above but now with Time on App and Yearly Amount
 Spent.
df_Time_app_amount = df_fyntra.filter(["Time_on_App", "Yearly_Amount_Spent"])

sns.set(style='white', color_codes=True)
sns.jointplot(x="Time_on_App", y="Yearly_Amount_Spent", data=df_Time_app_amoun
t)
```

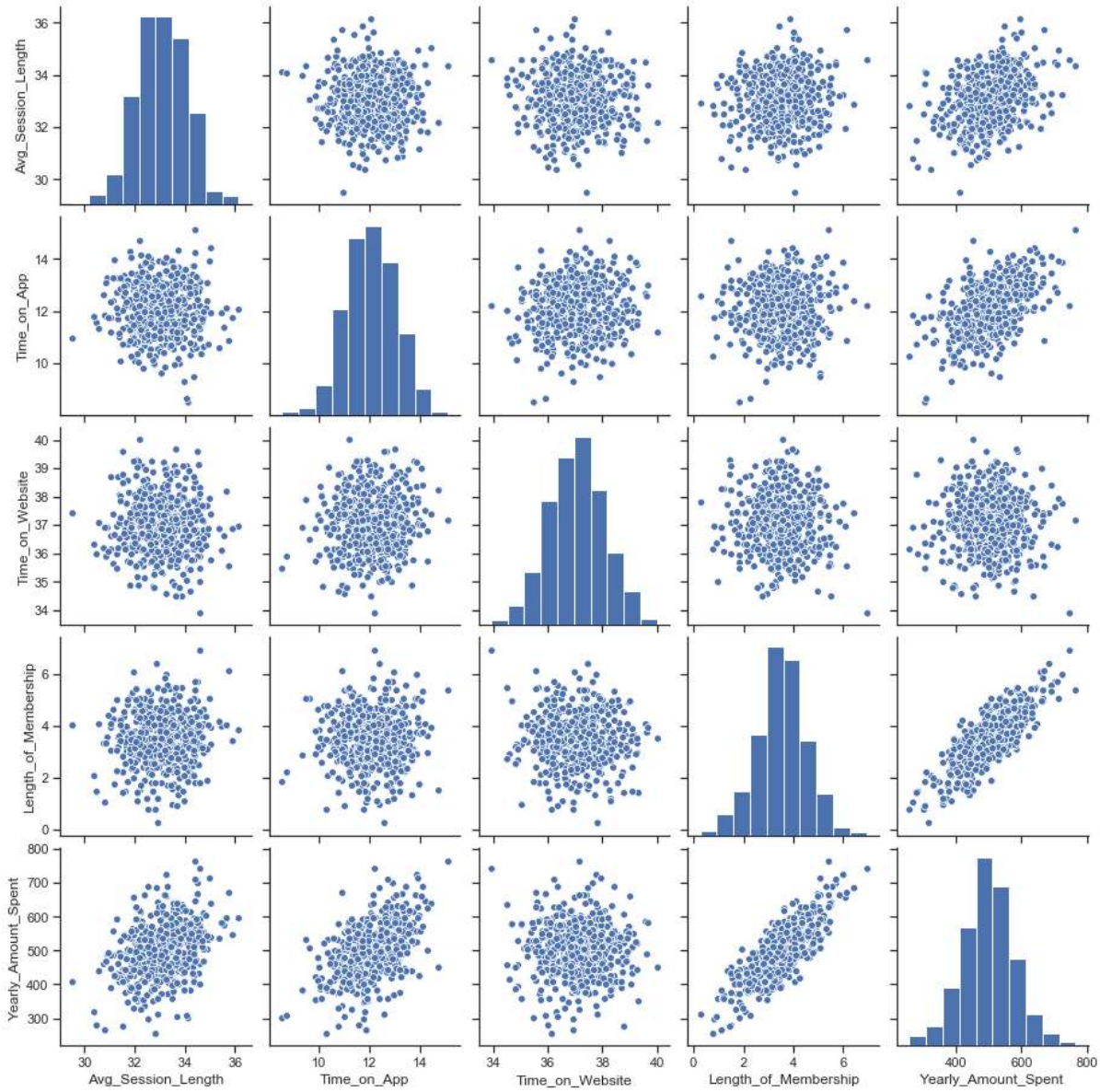
Out[13]: <seaborn.axisgrid.JointGrid at 0x20c6a2cb0c8>



In [15]:
```
# Is this correlation stronger than 1stOne?
coff_Time_app_amount = df_Time_app_amount.corr()
sns.heatmap(coff_Time_app_amount)   #strong correlation
```

Out[15]: <matplotlib.axes._subplots.AxesSubplot at 0x20c6a660308>

In [16]:
```
# 3.Compute --Explore types of relationships across the entire data set using
 pairplot . Based off this plot what looks to be the most correlated feature w
ith Yearly Amount Spent?
sns.set(style='ticks', color_codes=True)
b = sns.pairplot(df_fyntra)
```



In [17]:
```
# Based off this plot what looks to be the most correlated feature with Yearly
Amount Spent?
# The most correlated feature with Yearly_Amount_Spent is Lenght_Of_Membership
```

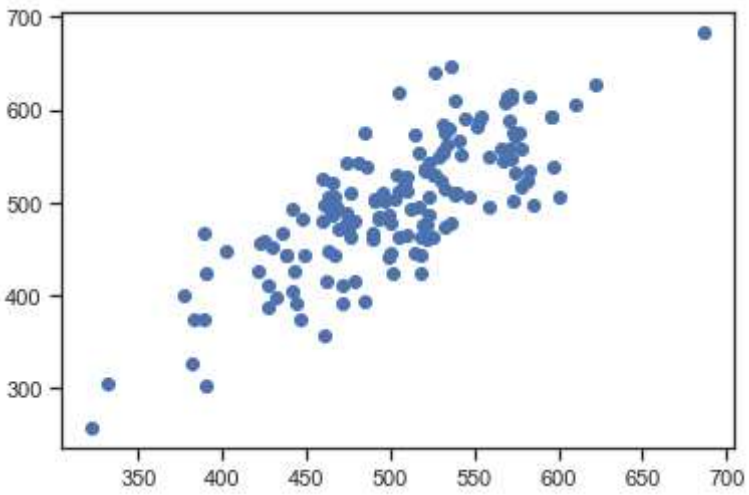
In [20]:
```
# 4.Compute -Create linear model plot of Length of Membership and Yearly Amoun
t Spent.
# Does the data fits well in linear plot?
X = df_fyntra["Length_of_Membership"]
Y = df_fyntra["Yearly_Amount_Spent"]

x_train, x_test, y_train, y_test = train_test_split(X, Y, test_size=0.3, rando
m_state=85)
linear_model = LinearRegression()
linear_model.fit(pd.DataFrame(x_train), y_train)
```

Out[20]:
```
LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None, normalize=Fals
e)
```

In [21]:
```
# 5.What is the use of random_state = 85?
# Random_state is used for initializing random number generators, which willde
cide the splitting of data into train and test indices.
# In case when a random_state is an integer, it is used to seed a new RandomSt
ate object
```

In [27]:
```
# 6.Compute -Predict the data and do a scatter plot. Check if actual and predi
cted data match?
predicted_values = linear_model.predict(pd.DataFrame(x_test))
plt.scatter(predicted_values, y_test)
```

Out[27]:
```
<matplotlib.collections.PathCollection at 0x20c6c1f0908>
```



In [28]:
```
# 7.What is the value of Root Mean Squared Error?
rms = sqrt(mean_squared_error(y_test, predicted_values))
print(rms)
```

```
44.777320711114676
```

In [ ]: