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

In [2]:
```
df = pd.read_csv("Ecommerce Customers.csv")
```

In [3]:
```
df.head()
```

Out[3]:

	Email	Address	Avatar	Avg. Session Length	Time on App	Time on Website	Length of Membership	
0	mstephenson@fernandez.com	835 Frank Tunnel\nWrightmouth, MI 82180-9605	Violet	34.497268	12.655651	39.577668	4.082621	5
1	hduke@hotmail.com	4547 Archer Common\nDiazchester, CA 06566-8576	DarkGreen	31.926272	11.109461	37.268959	2.664034	3
2	pallen@yahoo.com	24645 Valerie Unions Suite 582\nCobbborough, D...	Bisque	33.000915	11.330278	37.110597	4.104543	4
3	riverarebecca@gmail.com	1414 David Throughway\nPort Jason, OH 22070-1220	SaddleBrown	34.305557	13.717514	36.721283	3.120179	5
4	mstephens@davidson-herman.com	14023 Rodriguez Passage\nPort Jacobville, PR 3...	MediumAquaMarine	33.330673	12.795189	37.536653	4.446308	5

In [4]:
```
df.tail()
```

Out[4]:

	Email	Address	Avatar	Avg. Session Length	Time on App	Time on Website	Length of Membership	Ye Am Sp
495	lewisjessica@craig-evans.com	4483 Jones Motorway Suite 872\nLake Jamiefurt,...	Tan	33.237660	13.566160	36.417985	3.746573	573.847
496	katrina56@gmail.com	172 Owen Divide Suite 497\nWest Richard, CA 19320	PaleVioletRed	34.702529	11.695736	37.190268	3.576526	529.049
497	dale88@hotmail.com	0787 Andrews Ranch Apt. 633\nSouth Chadburgh, ...	Cornsilk	32.646777	11.499409	38.332576	4.958264	551.620
498	cwilson@hotmail.com	680 Jennifer Lodge Apt. 808\nBrendachester, TX...	Teal	33.322501	12.391423	36.840086	2.336485	456.469
499	hannahwilson@davidson.com	49791 Rachel Heights Apt. 898\nEast Drewboroug...	DarkMagenta	33.715981	12.418808	35.771016	2.735160	497.778

In [5]:
```
df.shape
```

Out[5]:
```
(500, 8)
```

In [6]:
```
df.dtypes
```

Out[6]:
```
Email                  object
Address                object
Avatar                 object
Avg. Session Length    float64
Time on App            float64
Time on Website        float64
Length of Membership   float64
Yearly Amount Spent    float64
dtype: object
```

In [7]:
```
df.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]:
```
df.describe()
```

Out[8]:

	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 [9]:
```
df.duplicated().sum()
```

Out[9]:
```
0
```

In [10]:
```
df.isnull().sum()
```

Out[10]:
```
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
```

In [11]:
```
corr = df.corr()
```

In [12]:
```
sns.heatmap(corr,annot=True)
```

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

In [13]:
```
df.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 [14]:
```
def yr_amount_spent(a):
        if a>200 and a<=300:
            group = "200-300"
        elif a>300 and a<=400:
            group = "300-400"
        elif a>400 and a<=500:
            group = "400-500"
        elif a>500 and a<=600:
            group = "500-600"
        elif a>600 and a<=700:
            group = "600-700"
        elif a>700 and a<=800:
            group = "700-800"
        return (group)
```

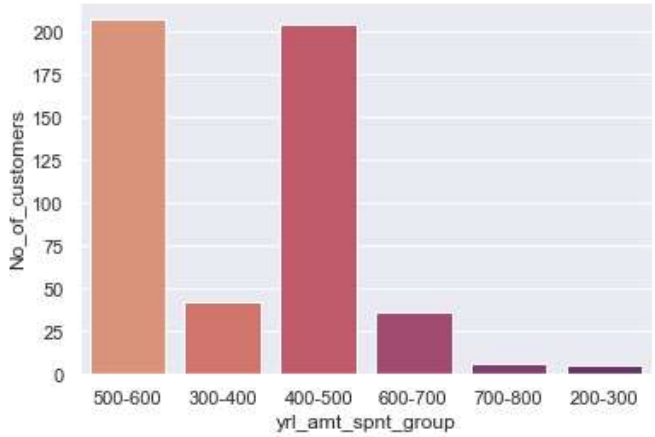
In [15]:
```
df["yrl_amt_spnt_group"] = df["Yearly Amount Spent"].apply(yr_amount_spent)
```

In [16]:
```
df.head()
```

Out[16]:

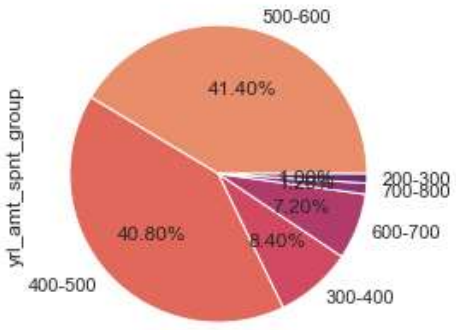
	Email	Address	Avatar	Avg. Session Length	Time on App	Time on Website	Length of Membership	
0	mstephenson@fernandez.com	835 Frank Tunnel\nWrightmouth, MI 82180-9605	Violet	34.497268	12.655651	39.577668	4.082621	5
1	hduke@hotmail.com	4547 Archer Common\nDiazchester, CA 06566-8576	DarkGreen	31.926272	11.109461	37.268959	2.664034	3
2	pallen@yahoo.com	24645 Valerie Unions Suite 582\nCobbborough, D...	Bisque	33.000915	11.330278	37.110597	4.104543	4
3	riverarebecca@gmail.com	1414 David Throughway\nPort Jason, OH 22070-1220	SaddleBrown	34.305557	13.717514	36.721283	3.120179	5
4	mstephens@davidson-herman.com	14023 Rodriguez Passage\nPort Jacobville, PR 3...	MediumAquaMarine	33.330673	12.795189	37.536653	4.446308	5

In [17]:
```
sns.set_theme(style='darkgrid', palette='flare')
sns.countplot(data = df, x = "yrl_amt_spnt_group")
plt.ylabel("No_of_customers")
plt.show()
```



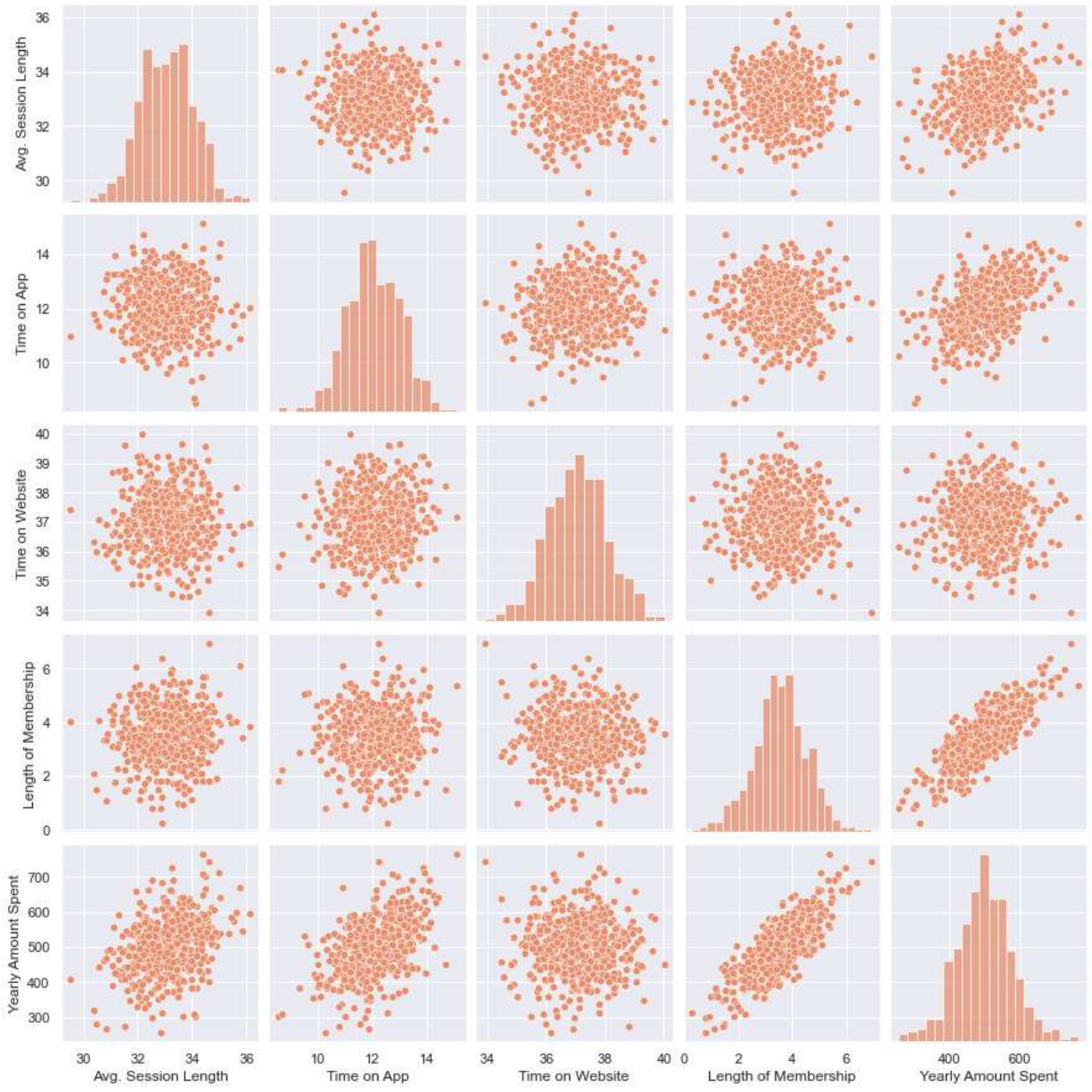
In [18]:
```
sns.set_theme(style='darkgrid', palette='flare')
df["yrl_amt_spnt_group"].value_counts().plot(kind="pie",autopct = "%.2f%%")
```

Out[18]: <AxesSubplot:ylabel='yrl_amt_spnt_group'>



In [19]:
```
sns.pairplot(df)
```

Out[19]:
```
<seaborn.axisgrid.PairGrid at 0x1bacb82c3a0>
```



In [20]:
```
#As we can see their is some sort of Linearity between yearly amount spent and Length of membersh
#Linear regression model to it
```

In [21]:
```
x1 = df.iloc[:,3:7]
x = df.iloc[:,3:7].values
y = df.iloc[:,-2].values
```

In [22]:
```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size = 0.3,random_state = (0))
```

In [23]:
```
from sklearn.linear_model import LinearRegression
lm = LinearRegression()
```

In [24]:
```
lm.fit(x_train,y_train)
```

Out[24]:
```
LinearRegression()
```

In [25]:
```
pred = lm.predict(x_test)
```

In [26]:
```
pred[0:5]
```

Out[26]:
```
array([438.05361824, 489.88569198, 370.69103491, 514.760391  ,
       496.7189217 ])
```

In [27]:
```
y_test[0:5]
```

Out[27]:
```
array([449.07031944, 482.60246733, 374.26967454, 513.15311185,
       502.77107457])
```

In [28]:
```
import sklearn.metrics as metric
metric.mean_absolute_error(y_test,pred)
```

Out[28]:
```
7.85137717086146
```

In [29]:
```
mse = metric.mean_squared_error(y_test,pred)
mse
```

Out[29]:
```
94.55779479273302
```

In [30]:
```
np.sqrt(mse)
```

Out[30]:
```
9.724083236620974
```

In [32]:
```
c = pd.DataFrame(lm.coef_,index=x1.columns)
c
```

Out[32]:

	0
Avg. Session Length	25.767530
Time on App	38.800394
Time on Website	-0.018041
Length of Membership	61.852568

In [ ]:
```
# As we can see that Time on app is more so recommended to invest more on app rather then Website
```