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

In [2]:
```
df=pd.read_csv(r"C:\Users\Admin\Downloads\16_Sleep_health_and_lifestyle_dataset
df
```

Out[2]:

	Person ID	Gender	Age	Occupation	Sleep Duration	Quality of Sleep	Physical Activity Level	Stress Level	BMI Category	Blo Pressu
0	1	Male	27	Software Engineer	6.1	6	42	6	Overweight	126/
1	2	Male	28	Doctor	6.2	6	60	8	Normal	125/
2	3	Male	28	Doctor	6.2	6	60	8	Normal	125/
3	4	Male	28	Sales Representative	5.9	4	30	8	Obese	140/
4	5	Male	28	Sales Representative	5.9	4	30	8	Obese	140/
...	...	...	...	...	...	...	...	...	...	
369	370	Female	59	Nurse	8.1	9	75	3	Overweight	140/
370	371	Female	59	Nurse	8.0	9	75	3	Overweight	140/
371	372	Female	59	Nurse	8.1	9	75	3	Overweight	140/
372	373	Female	59	Nurse	8.1	9	75	3	Overweight	140/
373	374	Female	59	Nurse	8.1	9	75	3	Overweight	140/

374 rows × 13 columns

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

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 374 entries, 0 to 373
Data columns (total 13 columns):
 #   Column                   Non-Null Count  Dtype  
---  ------                   --------------  -----  
 0   Person ID                374 non-null    int64  
 1   Gender                   374 non-null    object 
 2   Age                      374 non-null    int64  
 3   Occupation               374 non-null    object 
 4   Sleep Duration           374 non-null    float64
 5   Quality of Sleep         374 non-null    int64  
 6   Physical Activity Level  374 non-null    int64  
 7   Stress Level             374 non-null    int64  
 8   BMI Category             374 non-null    object 
 9   Blood Pressure           374 non-null    object 
 10  Heart Rate               374 non-null    int64  
 11  Daily Steps              374 non-null    int64  
 12  Sleep Disorder           374 non-null    object 
dtypes: float64(1), int64(7), object(5)
memory usage: 38.1+ KB
```

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

Out[4]:

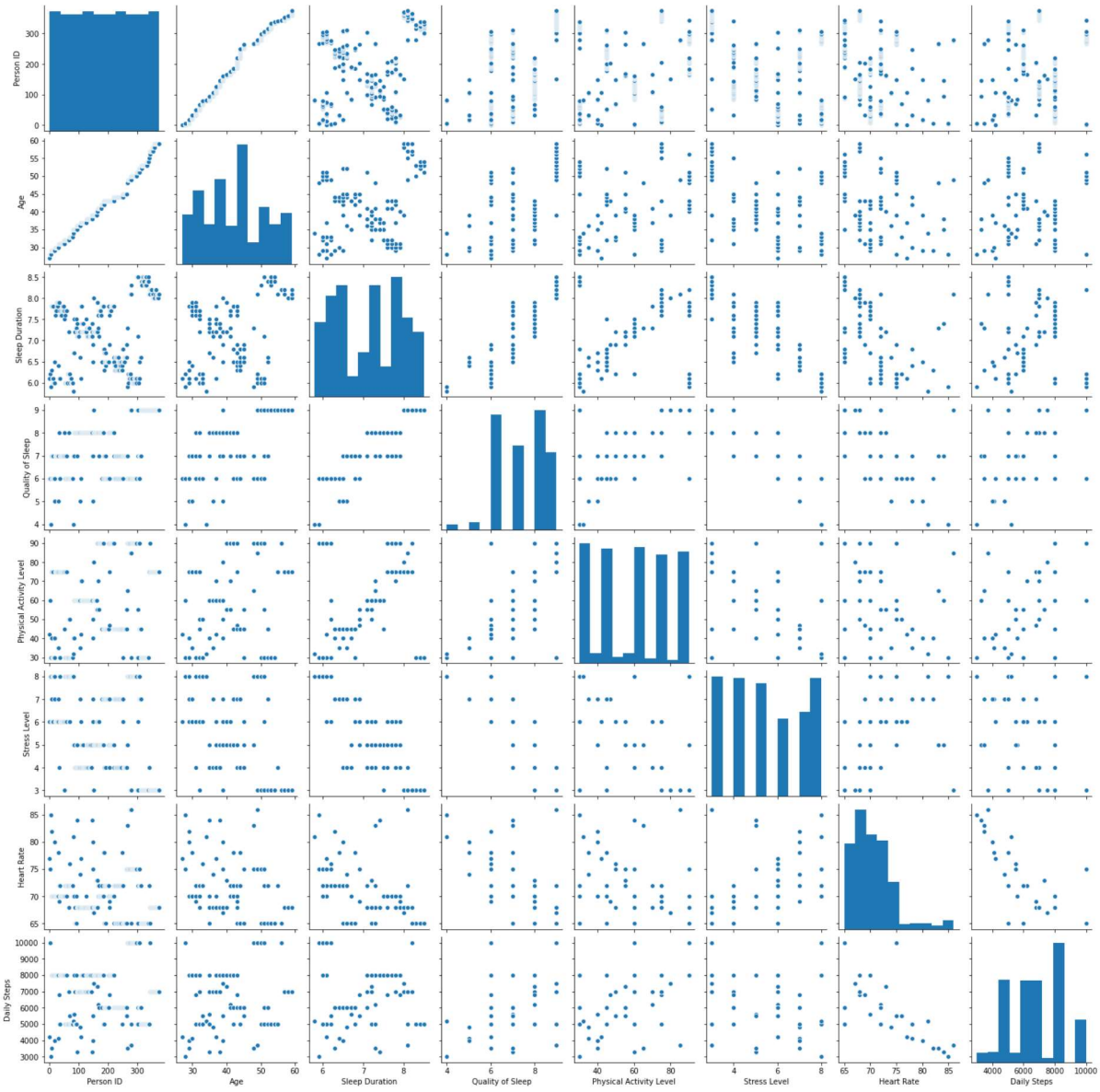
	Person ID	Age	Sleep Duration	Quality of Sleep	Physical Activity Level	Stress Level	Heart Rate	Da
count	374.000000	374.000000	374.000000	374.000000	374.000000	374.000000	374.000000	37
mean	187.500000	42.184492	7.132086	7.312834	59.171123	5.385027	70.165775	681
std	108.108742	8.673133	0.795657	1.196956	20.830804	1.774526	4.135676	161
min	1.000000	27.000000	5.800000	4.000000	30.000000	3.000000	65.000000	300
25%	94.250000	35.250000	6.400000	6.000000	45.000000	4.000000	68.000000	560
50%	187.500000	43.000000	7.200000	7.000000	60.000000	5.000000	70.000000	700
75%	280.750000	50.000000	7.800000	8.000000	75.000000	7.000000	72.000000	800
max	374.000000	59.000000	8.500000	9.000000	90.000000	8.000000	86.000000	1000

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

Out[5]:
```
Index(['Person ID', 'Gender', 'Age', 'Occupation', 'Sleep Duration',
       'Quality of Sleep', 'Physical Activity Level', 'Stress Level',
       'BMI Category', 'Blood Pressure', 'Heart Rate', 'Daily Steps',
       'Sleep Disorder'],
      dtype='object')
```

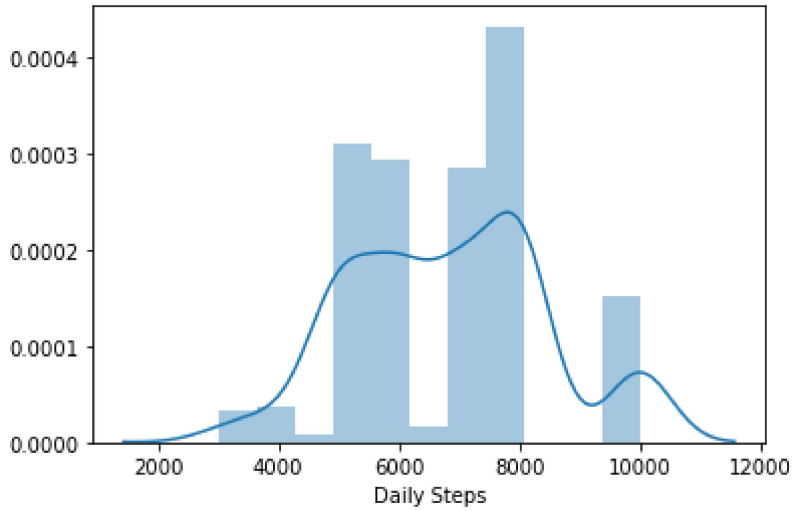
In [6]:
```
sns.pairplot(df)
```

Out[6]:
```
<seaborn.axisgrid.PairGrid at 0x1dbafd54520>
```



In [7]:
```
sns.distplot(df['Daily Steps'])
```

Out[7]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x1dbb1564f40>
```



In [8]:
```
df1=df[[ 'Age','Sleep Duration','Quality of Sleep', 'Physical Activity Level',
df1
```

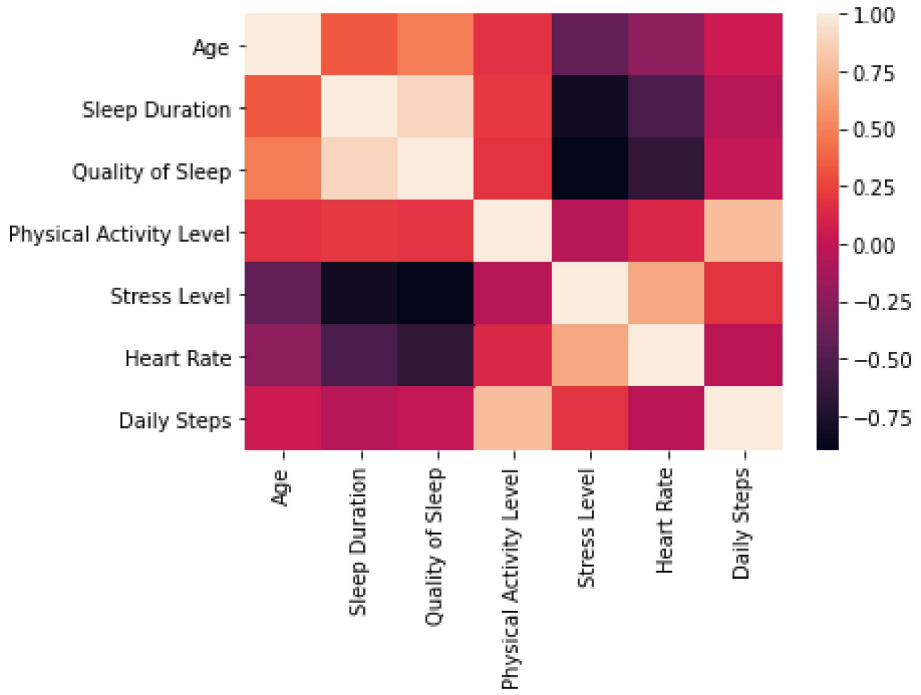
Out[8]:

	Age	Sleep Duration	Quality of Sleep	Physical Activity Level	Stress Level	Heart Rate	Daily Steps
0	27	6.1	6	42	6	77	4200
1	28	6.2	6	60	8	75	10000
2	28	6.2	6	60	8	75	10000
3	28	5.9	4	30	8	85	3000
4	28	5.9	4	30	8	85	3000
...	...	...	...	...	...	...	...
369	59	8.1	9	75	3	68	7000
370	59	8.0	9	75	3	68	7000
371	59	8.1	9	75	3	68	7000
372	59	8.1	9	75	3	68	7000
373	59	8.1	9	75	3	68	7000

374 rows × 7 columns

In [9]:
```
sns.heatmap(df1.corr())
```

Out[9]: <matplotlib.axes._subplots.AxesSubplot at 0x1dbb32efa90>



In [10]:
```
x=df1[['Age','Sleep Duration','Quality of Sleep', 'Physical Activity Level', 'S
y=df[['Daily Steps']]
```

In [11]:
```
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)
```

In [12]:
```
from sklearn.linear_model import LinearRegression
lr= LinearRegression()
lr.fit(x_train,y_train)
```

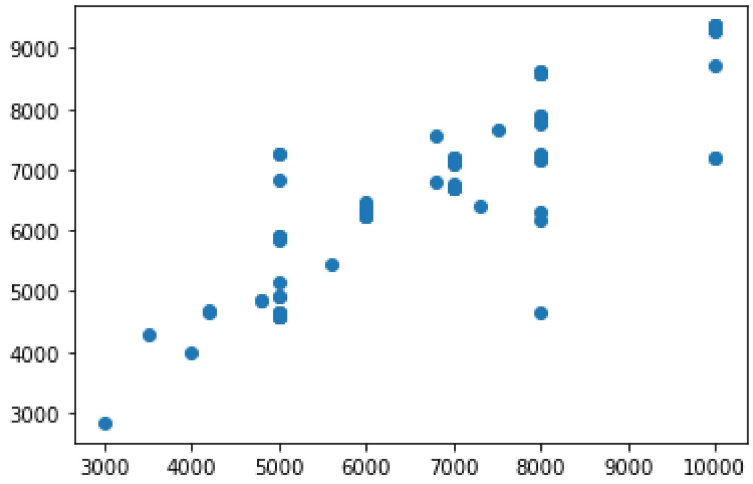
Out[12]: LinearRegression()

In [13]:
```
print(lr.intercept_)
```

```
[15189.72630935]
```

In [14]:
```
prediction= lr.predict(x_test)
plt.scatter(y_test,prediction)
```

Out[14]: <matplotlib.collections.PathCollection at 0x1dbb3de70a0>



In [15]:
```
print(lr.score(x_test,y_test))
```

```
0.7806869187411272
```

In [16]:
```
print(lr.score(x_train,y_train))
```

```
0.811290209197792
```

In [17]:
```
from sklearn.linear_model import Ridge,Lasso
```

In [18]:
```
rr=Ridge(alpha=10)
rr.fit(x_train,y_train)
```

Out[18]: Ridge(alpha=10)

In [19]:
```
rr.score(x_test,y_test)
```

Out[19]: 0.7757902146855578

In [20]:
```
la=Lasso(alpha=10)
la.fit(x_train,y_train)
```

Out[20]: Lasso(alpha=10)

In [21]:
```
la.score(x_test,y_test)
```

Out[21]: 0.7754192134584756

In [22]:
```
from sklearn.linear_model import ElasticNet
en=ElasticNet()
en.fit(x_train,y_train)
```

Out[22]:
```
ElasticNet()
```

In [23]:
```
print(en.intercept_)
```

```
[15404.09861656]
```

In [24]:
```
print(en.predict(x_test))
```

```
[7982.58754127 7227.14574521 8606.36712999 7070.37831617 7231.70352937
 6731.80585701 7245.91483467 7029.19956128 7258.25504848 7682.41016053
 6447.79665551 7984.86643335 4755.77622258 6731.80585701 5612.47126341
 6464.28685289 4998.44019276 4741.56491728 9095.31014209 5273.76783235
 4755.77622258 7041.95570557 4734.72824104 3426.52448464 5114.75950364
 6831.92382024 6250.10887346 6464.28685289 6250.10887346 7970.65512805
 7241.35705051 8763.86130629 6266.59907084 5511.69192171 4767.7086358
 6176.58878603 6480.77705027 5981.04585435 7058.44590295 9081.09883679
 6142.85119786 8015.56793603 6748.29605439 7982.58754127 8620.57843529
 8606.36712999 8017.84682811 6480.77705027 7727.54644945 6729.52696494
 4512.84905533 6466.56574497 4739.2860252  7258.25504848 6325.69308717
 8604.08823792 6478.49815819 6280.81037614 4749.73075667 9095.31014209
 4863.7461111  9064.60863941 7231.70352937 6266.59907084 6420.89963305
 6250.10887346 7241.35705051 6252.38776554 4767.7086358  6280.81037614
 9095.31014209 6729.52696494 6280.81037614 5039.49847696 6280.81037614
 8606.36712999 6870.98542282 9083.37772887 8616.02065114 7984.86643335
 6280.81037614 4840.95719031 4734.72824104 6464.28685289 7257.84724789
 5595.98106603 7260.12613997 6280.81037614 6447.79665551 7058.44590295
 7041.95570557 5624.40367663 4767.7086358  6464.28685289 6731.80585701
 7029.19956128 5114.75950364 4739.2860252  5624.40367663 7257.84724789
 9081.09883679 9081.09883679 6466.56574497 6420.89963305 5273.76783235
 9097.58903417 6280.81037614 7982.58754127 7227.14574521 4755.77622258
 9078.81994471 6731.80585701 9081.09883679]
```

In [25]:
```
print(en.score(x_test,y_test))
```

```
0.7574989049428511
```

Evaluation

In [26]:
```
from sklearn import metrics
```

In [27]:
```
print("Mean Absolute Error",metrics.mean_absolute_error(y_test,prediction))
```

```
Mean Absolute Error 538.1950525376343
```

In [28]:
```
print("Mean Squared Error",metrics.mean_squared_error(y_test,prediction))
```

```
Mean Squared Error 616746.0330451865
```

In [29]:
```
print("Root Mean Squared Error:",np.sqrt(metrics.mean_squared_error(y_test,pred
```

```
Root Mean Squared Error: 785.3317980606582
```

In [ ]:
```

```