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

In [2]:
```
df=pd.read_csv(r"C:\Users\user\Downloads\16_Sleep_health_and_lifestyle_dataset.
df.fillna(0,inplace=True)
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.head()
```

Out[3]:

	Person ID	Gender	Age	Occupation	Sleep Duration	Quality of Sleep	Physical Activity Level	Stress Level	BMI Category	Blood Pressure
0	1	Male	27	Software Engineer	6.1	6	42	6	Overweight	126/83
1	2	Male	28	Doctor	6.2	6	60	8	Normal	125/80
2	3	Male	28	Doctor	6.2	6	60	8	Normal	125/80
3	4	Male	28	Sales Representative	5.9	4	30	8	Obese	140/90
4	5	Male	28	Sales Representative	5.9	4	30	8	Obese	140/90

In [4]:
```
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 [5]:
```
import seaborn as sns
```

In [6]:
```
df.describe()
```

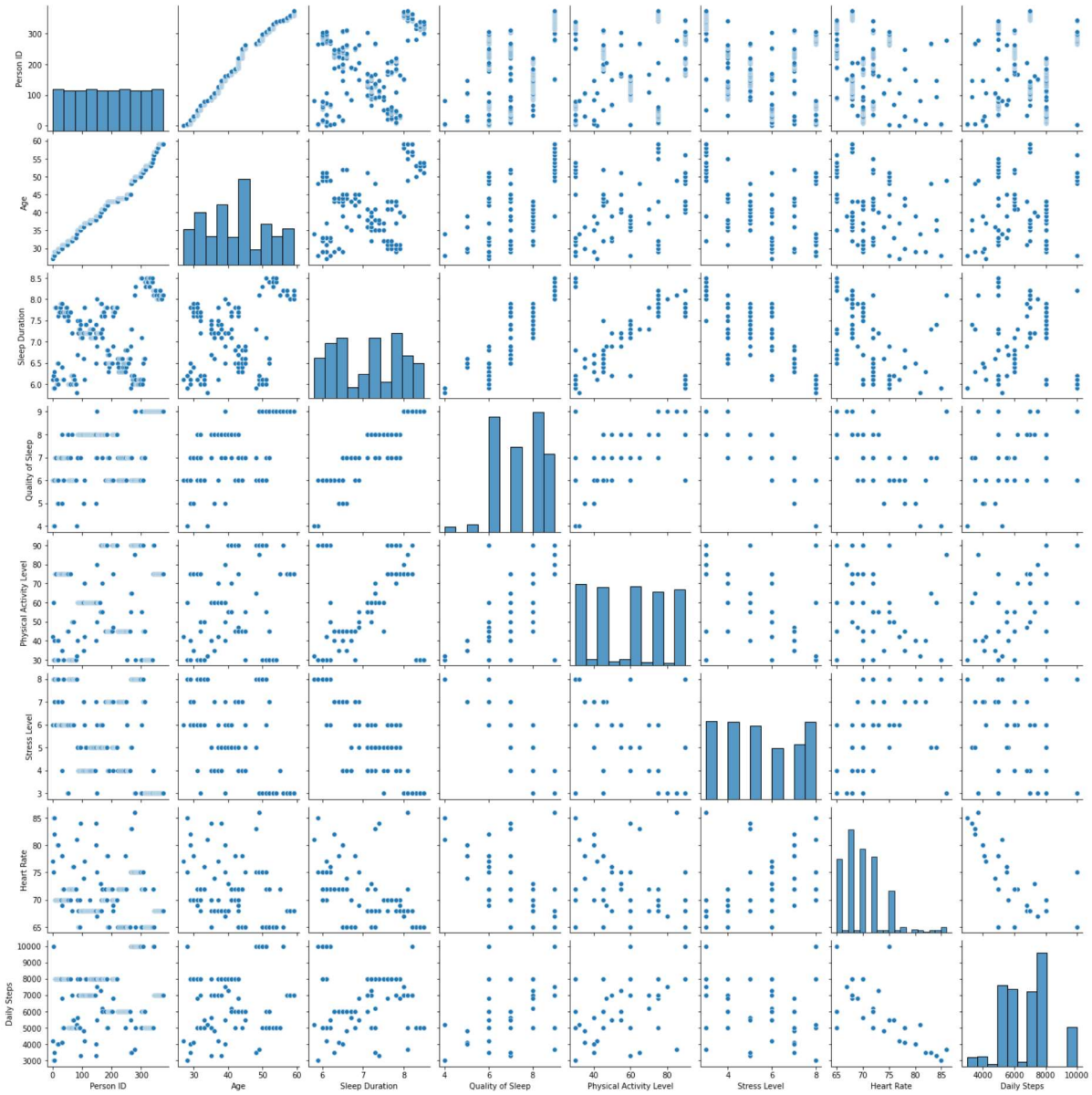
Out[6]:

	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

Type *Markdown* and LaTeX: α^2

In [7]:
```
sns.pairplot(df)
```

Out[7]:
```
<seaborn.axisgrid.PairGrid at 0x17c8218e0a0>
```

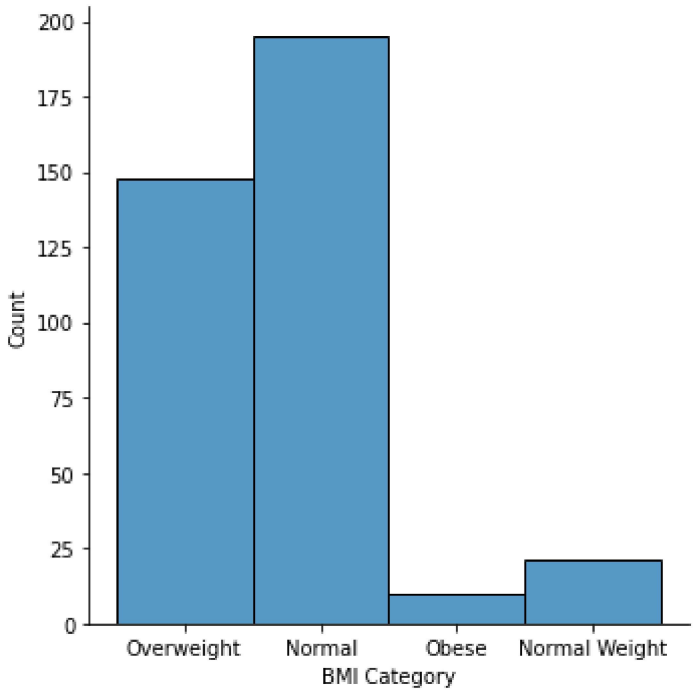


In [8]:
```
df1=df.drop(['Stress Level'],axis=1)
df1
df1=df1.drop(df1.index[1537:])
df1.isna().sum()
```

Out[8]:
```
Person ID                  0
Gender                     0
Age                        0
Occupation                 0
Sleep Duration             0
Quality of Sleep           0
Physical Activity Level    0
BMI Category               0
Blood Pressure             0
Heart Rate                 0
Daily Steps                0
Sleep Disorder             0
dtype: int64
```

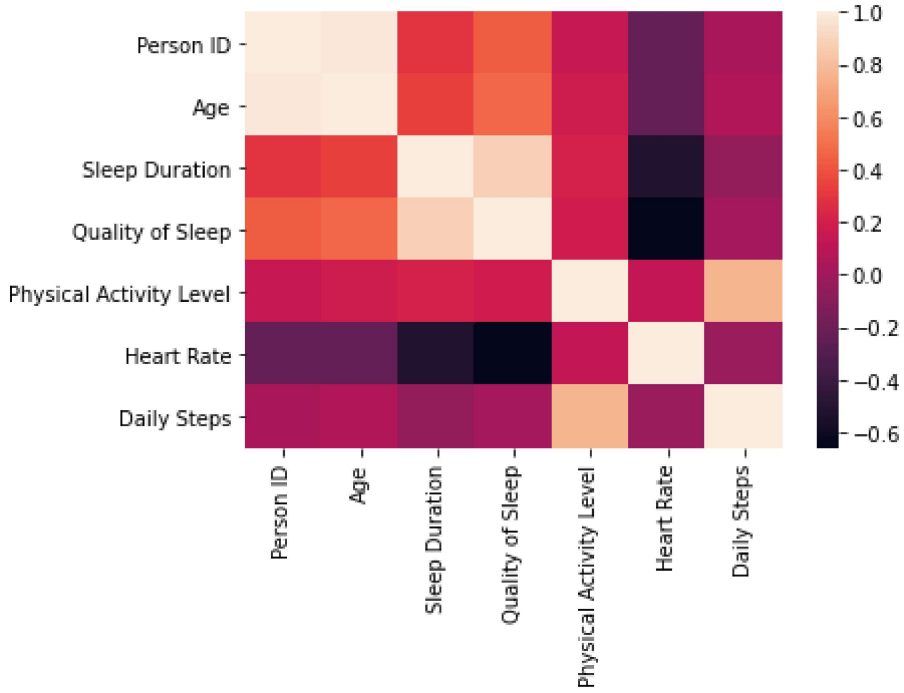
In [9]:
```
sns.displot(df['BMI Category'])
```

Out[9]:
```
<seaborn.axisgrid.FacetGrid at 0x17c85ea8340>
```



In [10]:
```
sns.heatmap(df1.corr())
```

Out[10]: <AxesSubplot:>



In [11]:
```
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
```

In [12]:
```
df1.isna().sum()
```

Out[12]:
```
Person ID                  0
Gender                     0
Age                        0
Occupation                 0
Sleep Duration             0
Quality of Sleep           0
Physical Activity Level    0
BMI Category               0
Blood Pressure             0
Heart Rate                 0
Daily Steps                0
Sleep Disorder             0
dtype: int64
```

In [13]:
```
y=df1['Age']
x=df1.drop(['Gender','BMI Category','Sleep Disorder','Occupation','Blood Pressu
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.3)
print(x_train)
```

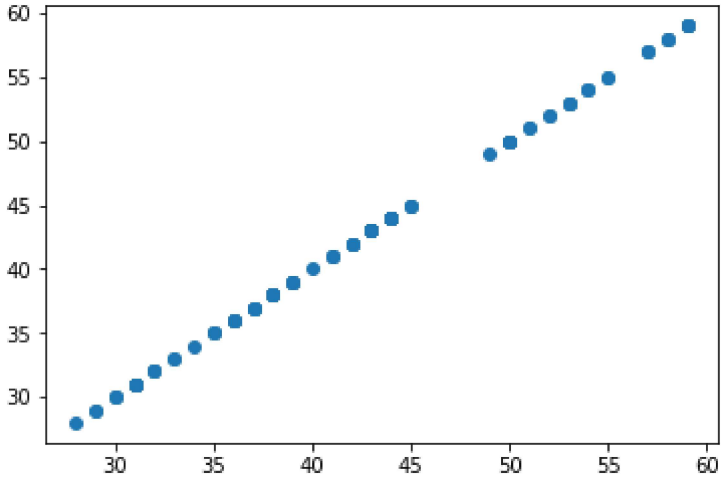
```
     Person ID  Age  Sleep Duration  Quality of Sleep  \
359        360   59             8.1                 9
14          15   29             6.0                 6
320        321   53             8.5                 9
193        194   43             6.5                 6
237        238   44             6.5                 7
..         ...  ...             ...               ...
89          90   35             7.3                 8
281        282   50             6.1                 6
92          93   35             7.5                 8
7            8   29             7.8                 7
141        142   38             7.1                 8

     Physical Activity Level  Heart Rate  Daily Steps
359                       75          68         7000
14                        30          70         8000
320                       30          65         5000
193                       45          72         6000
237                       45          65         6000
..                       ...         ...          ...
89                        60          65         5000
281                       90          75        10000
92                        60          70         8000
7                         75          70         8000
141                       60          68         8000

[261 rows x 7 columns]
```

In [14]:
```
model=LinearRegression()
model.fit(x_train,y_train)
model.intercept_
```

Out[14]:
```
4.710898338089464e-12
```

In [15]:
```
prediction=model.predict(x_test)
plt.scatter(y_test,prediction)
```

Out[15]:
```
<matplotlib.collections.PathCollection at 0x17c88696c40>
```



In [16]:
```
model.score(x_test,y_test)
```

Out[16]:
```
1.0
```

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]:
```
0.9999789567073385
```

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]:
```
0.9780638231683756
```

In [22]:
```
from sklearn.linear_model import ElasticNet
en=ElasticNet()
en.fit(x_train,y_train)
print(en.coef_)
print(en.intercept_)
print(en.predict(x_test))
print(en.score(x_test,y_test))
from sklearn import metrics
print("Mean Absolute Error:",metrics.mean_absolute_error(y_test,prediction))
print("Mean Squared Error:",metrics.mean_squared_error(y_test,prediction))
print("Root Mean Squared Error:",np.sqrt(metrics.mean_squared_error(y_test,prec
```

```
[ 4.24139585e-02  4.60870002e-01  0.00000000e+00  0.00000000e+00
  7.78282840e-03 -0.00000000e+00 -2.48127842e-05]
14.51323515335801
[39.24496833 42.00624006 44.87531816 51.9129809  37.80857728 50.47876579
 30.88197342 41.37571423 56.08014736 56.62584528 50.5635937  37.07562235
 41.62081044 28.77264258 33.78078386 32.00016159 45.53634032 50.60600766
 44.74807629 31.61832607 51.82815299 46.39653713 32.92758513 42.13348194
 38.29994361 31.30611301 52.62123293 52.4515771  46.48136504 43.8630667
 34.71901937 51.44860329 42.67547586 36.73631068 42.1758959  45.044974
 31.09404321 33.13965492 43.8667707  56.66825924 35.66890455 54.78389334
 53.84555419 44.07884049 31.22128509 27.83112547 45.5539415  50.89362772
 41.94252093 43.56987299 38.02064707 36.90596652 35.6392313  46.22688129
 49.92738433 28.68781466 57.42602695 55.99531945 39.58427999 31.57591211
 53.6334844  44.70566233 54.74147939 42.71788982 29.86972196 43.65470091
 29.57282425 34.37881929 35.76647318 52.53640502 39.32979624 37.16045027
 31.70326388 35.59681734 42.88754565 57.85016654 44.12125445 39.45703812
 44.79049024 37.87580402 34.51733914 57.59568279 45.34187171 50.0546262
 45.25704379 41.20605839 56.79550112 43.82435674 35.75373247 40.85604309
 57.93499446 37.89340519 31.17887113 37.03320839 44.62083441 40.06601876
 29.827308   50.18186808 30.66990363 57.68051071 55.95290549 38.21511569
 31.0092153  31.26369905 52.9605446  45.12980191 43.61228695 49.04237474
 43.90918466 38.44478666 53.80314023 42.50582003 38.99048457]
0.9939920318787495
Mean Absolute Error: 3.7482701308548294e-13
Mean Squared Error: 2.009212250272817e-25
Root Mean Squared Error: 4.482423730832257e-13
```