In [47]:
```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error,r2_score
from sklearn.linear_model import LinearRegression
```

In [4]:
```
df=pd.read_csv("C://Users//user//Downloads//synthetic_health_data.csv")
df
```

Out[4]:

	Age	BMI	Exercise_Frequency	Diet_Quality	Sleep_Hours	Smoking_Status	Alcohol
0	45.960570	31.996777	5	55.403270	7.300359	0	
1	38.340828	29.623168	6	41.838357	7.012419	1	
2	47.772262	25.298152	5	76.904948	6.028641	1	
3	58.276358	21.765316	2	49.756767	5.802714	1	
4	37.190160	28.491117	2	44.218737	7.912548	0	
...	...	...	...	...	...	...	
995	36.626796	30.350751	1	60.674477	8.470913	0	
996	61.572238	24.867394	3	66.527725	5.355398	1	
997	47.690114	20.590627	4	69.819819	8.641864	0	
998	33.145852	24.184665	6	70.724204	7.941557	0	
999	46.870993	21.275487	0	63.555888	9.038352	0	

1000 rows × 8 columns

In [5]:
```
df.head(5)
```

Out[5]:

	Age	BMI	Exercise_Frequency	Diet_Quality	Sleep_Hours	Smoking_Status	Alcohol_C
0	45.960570	31.996777	5	55.403270	7.300359	0	
1	38.340828	29.623168	6	41.838357	7.012419	1	
2	47.772262	25.298152	5	76.904948	6.028641	1	
3	58.276358	21.765316	2	49.756767	5.802714	1	
4	37.190160	28.491117	2	44.218737	7.912548	0	

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

Out[6]:

	Age	BMI	Exercise_Frequency	Diet_Quality	Sleep_Hours	Smoking_Status	Alcohol
995	36.626796	30.350751	1	60.674477	8.470913	0	
996	61.572238	24.867394	3	66.527725	5.355398	1	
997	47.690114	20.590627	4	69.819819	8.641864	0	
998	33.145852	24.184665	6	70.724204	7.941557	0	
999	46.870993	21.275487	0	63.555888	9.038352	0	

In [7]:
```
df.shape
```

Out[7]:
```
(1000, 8)
```

In [8]:
```
df.columns
```

Out[8]:
```
Index(['Age', 'BMI', 'Exercise_Frequency', 'Diet_Quality', 'Sleep_Hours',
       'Smoking_Status', 'Alcohol_Consumption', 'Health_Score'],
      dtype='object')
```

In [9]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1000 entries, 0 to 999
Data columns (total 8 columns):
 #   Column               Non-Null Count  Dtype  
---  ------               --------------  -----  
 0   Age                  1000 non-null   float64
 1   BMI                  1000 non-null   float64
 2   Exercise_Frequency   1000 non-null   int64  
 3   Diet_Quality         1000 non-null   float64
 4   Sleep_Hours          1000 non-null   float64
 5   Smoking_Status       1000 non-null   int64  
 6   Alcohol_Consumption  1000 non-null   float64
 7   Health_Score         1000 non-null   float64
dtypes: float64(6), int64(2)
memory usage: 62.6 KB
```

In [12]:
```
df.columns=df.columns.str.strip().str.lower()
```

In [14]:
```
df.describe().T
```

Out[14]:

	count	mean	std	min	25%	50%	75%	
age	1000.0	40.231985	11.750591	1.104792	32.228916	40.303607	47.775327	86
bmi	1000.0	25.354181	4.987272	10.298057	21.968792	25.315386	28.644411	40
exercise_frequency	1000.0	2.888000	1.995354	0.000000	1.000000	3.000000	5.000000	6
diet_quality	1000.0	69.952977	14.972061	19.907497	59.945481	69.975151	80.527839	110
sleep_hours	1000.0	6.973135	1.517218	2.431107	5.903351	6.990847	8.054595	11
smoking_status	1000.0	0.499000	0.500249	0.000000	0.000000	0.000000	1.000000	1
alcohol_consumption	1000.0	3.079377	2.084564	-3.592506	1.644111	3.064261	4.489293	11
health_score	1000.0	85.479947	13.633845	29.106017	76.430819	87.498996	99.762644	100

In [17]:
```
df["health_score"].value_counts()
```

Out[17]:
```
health_score
100.000000    242
70.542122       1
68.625766       1
83.950901       1
85.939618       1
             ...
71.200686       1
75.874576       1
91.167406       1
48.158023       1
87.995811       1
Name: count, Length: 759, dtype: int64
```

In [18]:
```
df.duplicated().sum()
```

Out[18]:
```
0
```

In [19]:
```
df.isnull().sum()
```

Out[19]:
```
age                   0
bmi                   0
exercise_frequency    0
diet_quality          0
sleep_hours           0
smoking_status        0
alcohol_consumption   0
health_score          0
dtype: int64
```

In [22]:
```
df["age"]=df["age"].astype(int)
df
```

Out[22]:

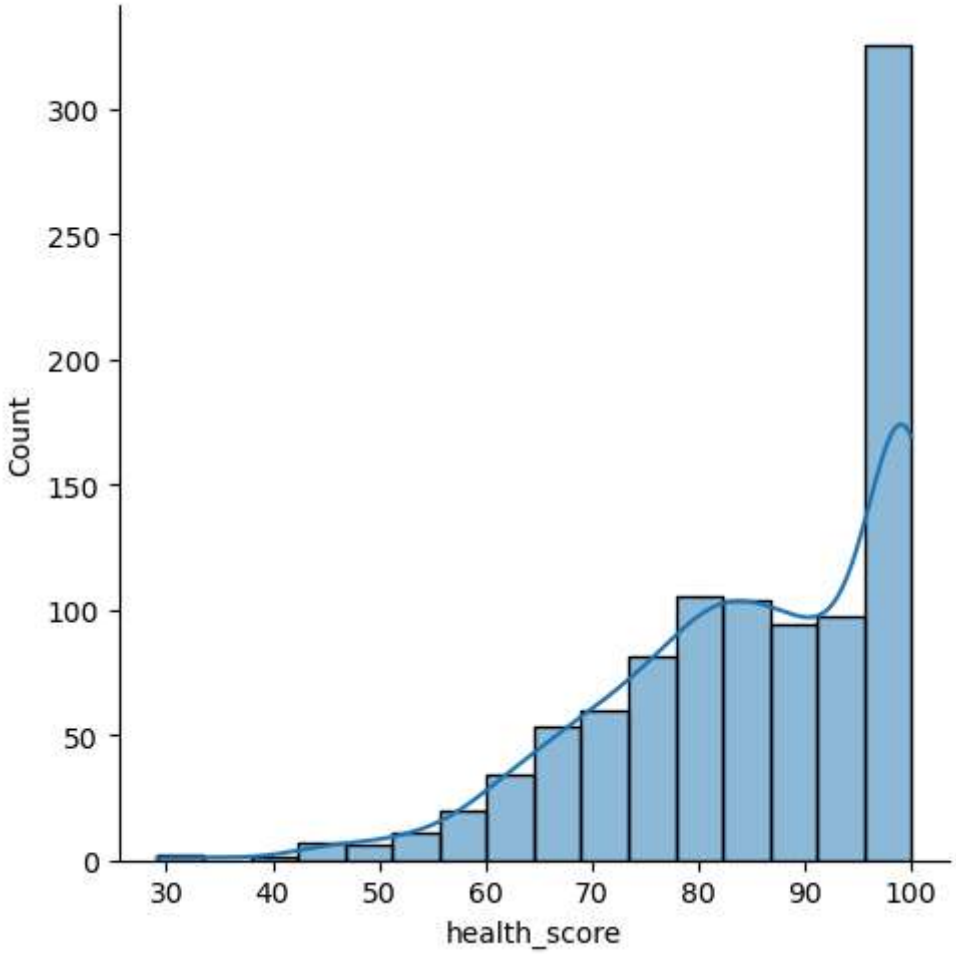
	age	bmi	exercise_frequency	diet_quality	sleep_hours	smoking_status	alcohol_consum
0	45	31.996777	5	55.403270	7.300359	0	2.8
1	38	29.623168	6	41.838357	7.012419	1	7.1
2	47	25.298152	5	76.904948	6.028641	1	4.0
3	58	21.765316	2	49.756767	5.802714	1	3.6
4	37	28.491117	2	44.218737	7.912548	0	2.8
...	...	...	...	...	...	...	
995	36	30.350751	1	60.674477	8.470913	0	3.8
996	61	24.867394	3	66.527725	5.355398	1	5.5
997	47	20.590627	4	69.819819	8.641864	0	8.1
998	33	24.184665	6	70.724204	7.941557	0	3.6
999	46	21.275487	0	63.555888	9.038352	0	4.2

1000 rows × 8 columns

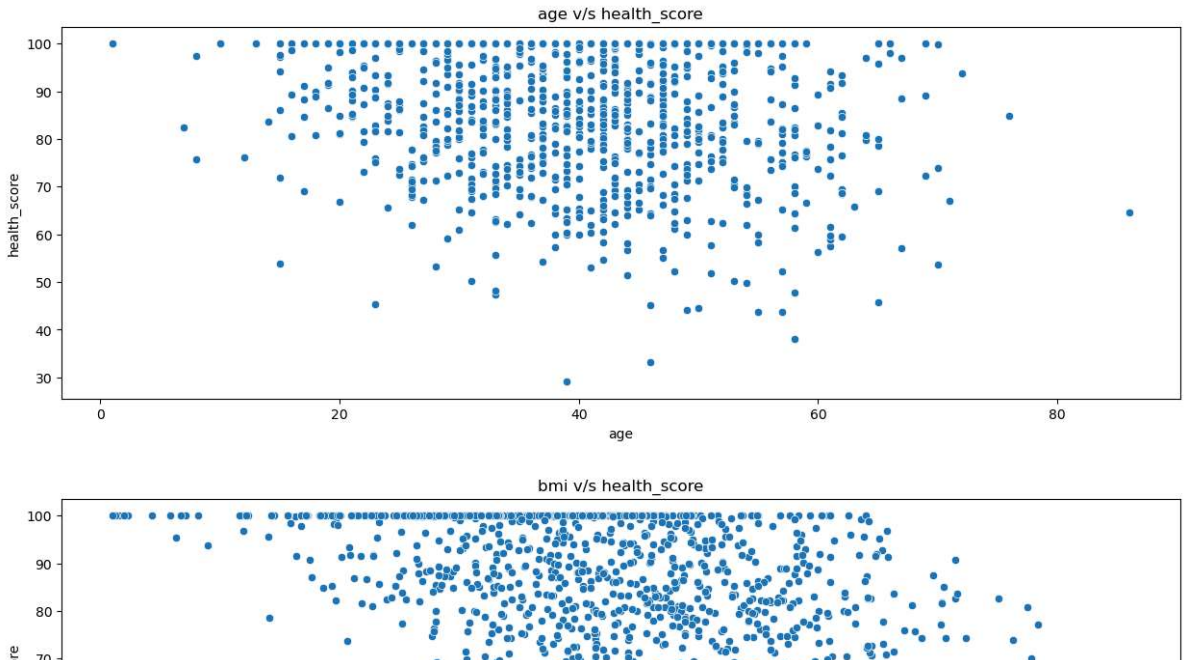
In [27]:
```
sns.displot(x="health_score",data=df,kde=True)
```

```
C:\Users\user\anaconda3\Lib\site-packages\seaborn\axisgrid.py:118: UserWarning:
The figure layout has changed to tight
  self._figure.tight_layout(*args, **kwargs)
```

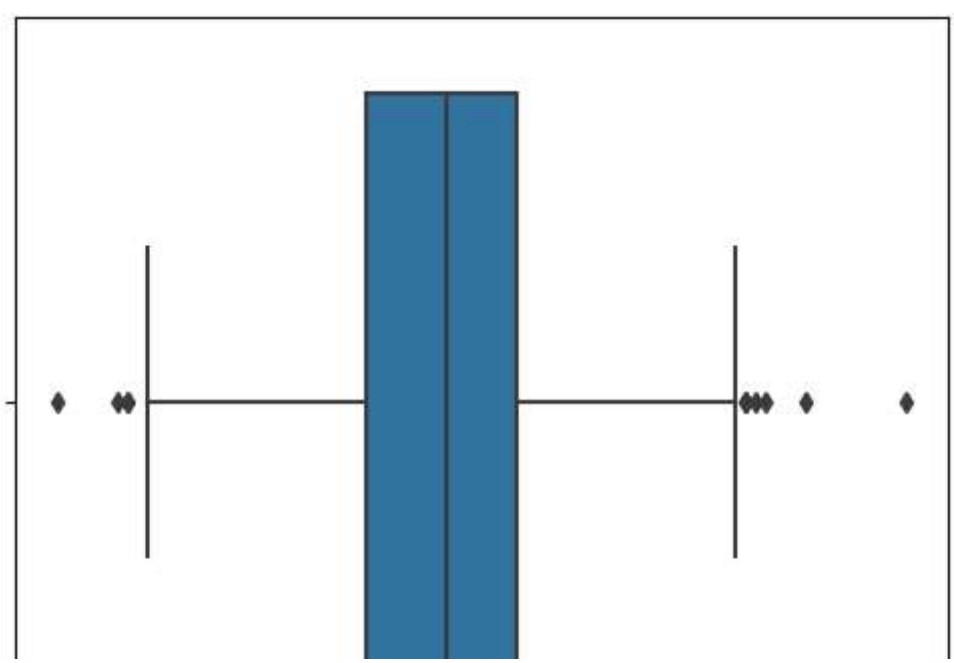
Out[27]:
```
<seaborn.axisgrid.FacetGrid at 0x264c3bef6d0>
```



In [34]:
```
for col in df.columns:
    if col!="health_score":
        plt.figure(figsize=(15,5))
        sns.scatterplot(x=col, y='health_score', data=df)
        plt.title(f"{col} v/s health_score")
        plt.show()
```



In [37]:
```
for col in df.columns:
    plt.figure(figsize=(6,5))
    sns.boxplot(x=col,data=df)
    plt.show
```



In [38]:
```
x=df.drop("health_score",axis=1)
y=df["health_score"]
```

In [41]:
```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2)
```

In [43]:
```
from sklearn.linear_model import LinearRegression
```

In [45]:
```
lr=LinearRegression()
lr.fit(x_train,y_train)
y_pred=lr.predict(x_test)
```

In [48]:
```
mse = mean_squared_error(y_test, y_pred)
print("MSE:", mse)
```

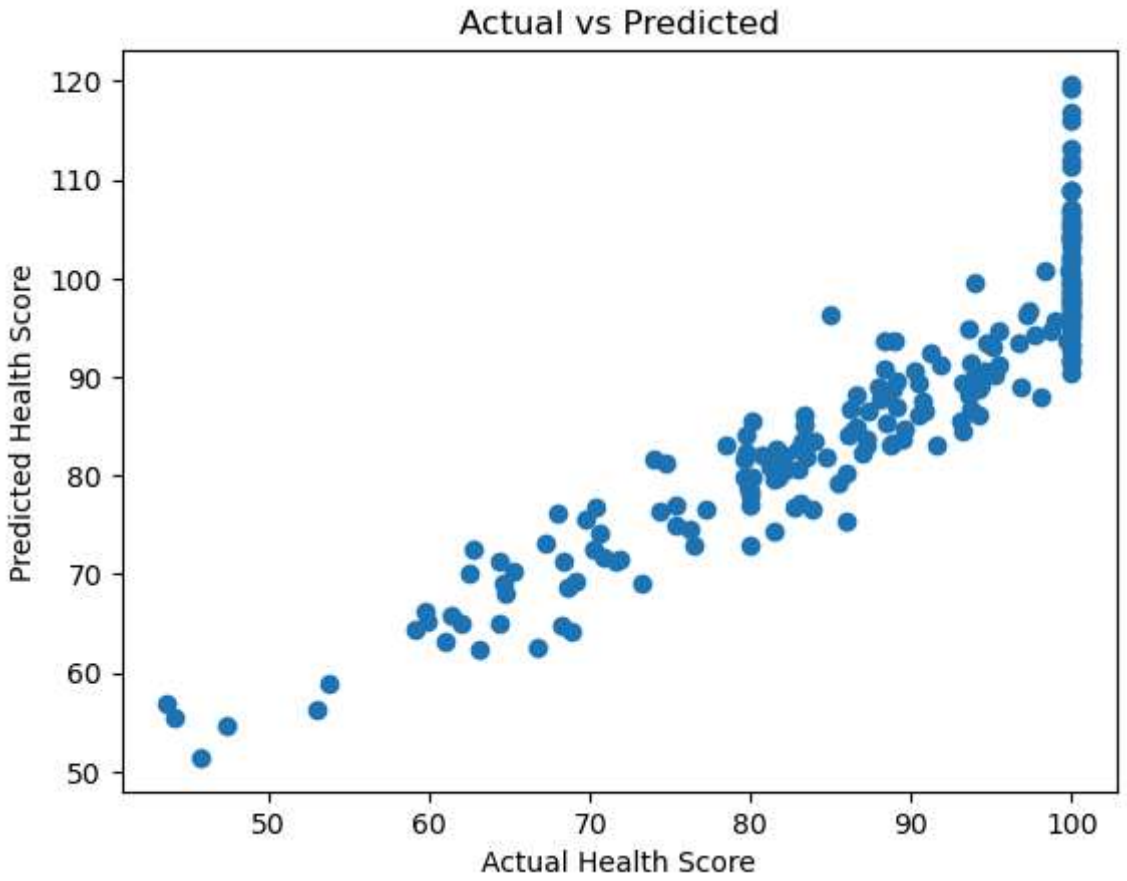
```
MSE: 30.293310721094727
```

In [49]:
```
from sklearn.metrics import r2_score
r2 = r2_score(y_test, y_pred)
print("R2 Score:", r2)
```

```
R2 Score: 0.8376168315066221
```

In [50]:
```
plt.scatter(y_test, y_pred)
plt.xlabel("Actual Health Score")
plt.ylabel("Predicted Health Score")
plt.title("Actual vs Predicted")
```

Out[50]:
```
Text(0.5, 1.0, 'Actual vs Predicted')
```



In [51]:
```
new_data = [[25, 22.0, 4, 3, 7.5, 0, 1]]
prediction = lr.predict(new_data)
print("Predicted Health Score:", prediction[0])
```

```
Predicted Health Score: 58.033320922378

C:\Users\user\anaconda3\Lib\site-packages\sklearn\utils\validation.py:2739: Use
rWarning: X does not have valid feature names, but LinearRegression was fitted
with feature names
  warnings.warn(
```

In [ ]:
```

```