In [4]:
```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import Lasso
import matplotlib.pyplot as plt
from sklearn.metrics import r2_score
from sklearn.metrics import mean_absolute_error
from sklearn.metrics import mean_squared_error
```

In [5]:
```
from google.colab import files
uploaded = files.upload()
df = pd.read_excel("synthetic_health_data.xlsx")
print(df)
```

Browse...

Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.

```
Saving synthetic_health_data.xlsx to synthetic_health_data.xlsx
     Weight  Height  Age  Gender
0        88    1.70   43       0
1        78    1.69   64       0
2        64    1.57   86       0
3        92    1.67   64       0
4        57    1.66   35       1
..      ...     ...  ...     ...
495      54    1.67   36       1
496      61    1.87   33       0
497      65    1.64   79       0
498      75    1.73   53       1
499      75    1.75   47       0

[500 rows x 4 columns]
```

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

Out[6]: (500, 4)

In [7]:
```
df['BMI'] = df['Weight'] / (df['Height'] ** 2)
```

In [8]:
```
df.isnull().sum()
```

Out[8]:

	0
Weight	0
Height	0
Age	0
Gender	0
BMI	0

dtype: int64

In [9]:
```
x = df.drop("Weight", axis=1).values
y = df["Weight"].values
```

In [14]:
```
from sklearn.preprocessing import StandardScaler

scaler = StandardScaler()
x_scaled = scaler.fit_transform(x)

x_train, x_test, y_train, y_test = train_test_split(x_scaled, y, test_size=0.2
```

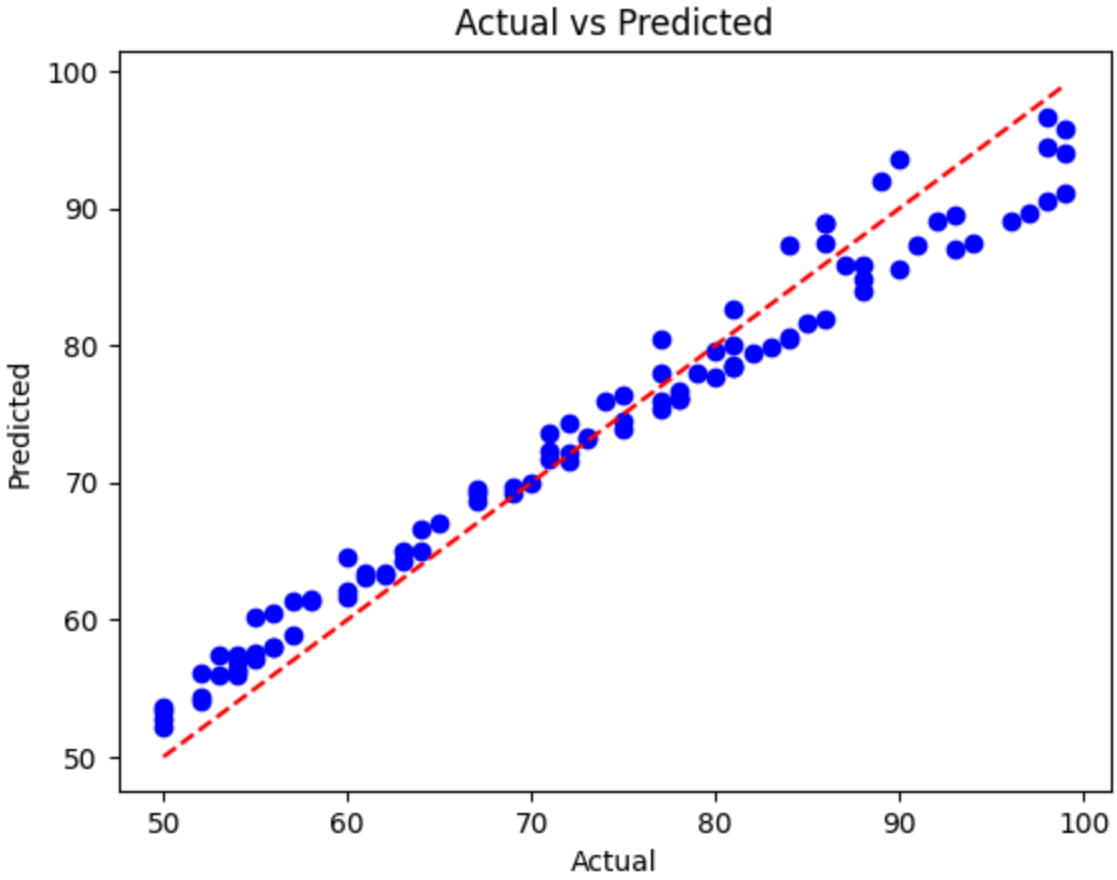
In [15]:
```
model=Lasso()
model.fit(x_train,y_train)
prediction=model.predict(x_test)
print(prediction)
print(y_test)
```

```
[60.47700798 75.36223524 87.49805764 54.41580182 84.01524158 77.66197567
 71.763378   69.22703862 57.57830585 72.19346738 57.47314335 80.51077016
 72.30289974 93.97712262 78.42508348 73.58136854 82.58611552 61.39745007
 88.90073587 61.35351331 76.18255017 62.15704304 57.01353759 52.74860286
 84.87741843 88.90073587 53.60737397 63.3021635  60.21406489 71.52510177
 57.2068147  79.53208127 65.04292761 65.05375628 57.97547698 63.06642388
 67.04684505 73.85282686 78.0290571  64.54206809 70.02666103 80.53846767
 74.46905983 81.56904926 55.92381418 76.00279268 87.34300993 74.29843897
 76.14848321 89.14006888 64.28661229 76.40383871 57.45462588 87.35915594
 53.2853672  79.42158897 96.63807772 73.37744331 69.28598804 95.77754052
 68.59650042 89.00237468 56.29980791 52.13207442 91.96460934 56.09951306
 76.11117051 76.60496007 58.94200157 79.97099199 56.02162214 78.04895117
 79.83483554 73.11621891 78.55069195 89.56100506 85.57654148 69.63402522
 78.4168937  85.83084421 80.55684395 76.00805457 66.64754192 85.837398
 61.52957462 90.5835784  89.64343855 93.55165034 81.88585827 56.42747501
 86.9748046  91.05573798 87.40048315 63.3690267  61.60804469 57.97133493
 63.423829   69.49620392 94.51333234 54.09582188]
[56 77 86 52 88 80 71 67 55 72 54 77 71 99 81 71 81 57 86 58 78 60 54 50
 88 86 50 62 55 72 55 80 63 64 56 61 65 75 79 60 70 84 75 85 53 74 84 72
 78 96 63 75 53 91 50 82 98 73 69 99 67 92 54 50 89 52 78 78 57 81 54 77
 83 73 81 93 90 69 81 87 84 77 64 88 58 98 97 90 86 54 93 99 94 62 60 56
 61 67 98 52]
```

In [16]:
```
r2 = r2_score(y_test, prediction)
print("R² Score:", r2)
```

```
R² Score: 0.9549396070906674
```

In [17]:
```
plt.scatter(y_test, prediction, color='blue')
plt.plot([y_test.min(), y_test.max()], [y_test.min(), y_test.max()], 'r--')
plt.xlabel("Actual")
plt.ylabel("Predicted")
plt.title("Actual vs Predicted")
plt.show()
```


Actual vs Predicted
Predicted
Actual
100
90
80
70
60
50
50
60
70
80
90
100