In [1]:
```
import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)
import seaborn as sns
sns.set(style="whitegrid")
import matplotlib.pyplot as plt
from collections import Counter
%matplotlib inline
```

In [2]:
```
# ignore warnings
import warnings
warnings.filterwarnings('ignore')
```

In [3]:
```
fifa = pd.read_excel(r"C:\Users\Vansh\Downloads\fifa world cup.xlsx")
fifa
```

Out[3]:

	Unnamed: 0	ID	Name	Age	Phc
0	0	158023	L. Messi	31	https://cdn.sofifa.org/players/4/19/158023.p
1	1	20801	Cristiano Ronaldo	33	https://cdn.sofifa.org/players/4/19/20801.p
2	2	190871	Neymar Jr	26	https://cdn.sofifa.org/players/4/19/190871.p
3	3	193080	De Gea	27	https://cdn.sofifa.org/players/4/19/193080.p
4	4	192985	K. De Bruyne	27	https://cdn.sofifa.org/players/4/19/192985.p
...	...	...	...	...	
18202	18202	238813	J. Lundstram	19	https://cdn.sofifa.org/players/4/19/238813.p
18203	18203	243165	N. Christoffersson	19	https://cdn.sofifa.org/players/4/19/243165.p
18204	18204	241638	B. Worman	16	https://cdn.sofifa.org/players/4/19/241638.p
18205	18205	246268	D. Walker-Rice	17	https://cdn.sofifa.org/players/4/19/246268.p
18206	18206	246269	G. Nugent	16	https://cdn.sofifa.org/players/4/19/246269.p

18207 rows × 89 columns

EXPLORATORY DATA ANALYSIS

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

Out[5]:

	Unnamed: 0	ID	Name	Age	Photo	Natio
0	0	158023	L. Messi	31	https://cdn.sofifa.org/players/4/19/158023.png	Arge
1	1	20801	Cristiano Ronaldo	33	https://cdn.sofifa.org/players/4/19/20801.png	Po
2	2	190871	Neymar Jr	26	https://cdn.sofifa.org/players/4/19/190871.png	
3	3	193080	De Gea	27	https://cdn.sofifa.org/players/4/19/193080.png	
4	4	192985	K. De Bruyne	27	https://cdn.sofifa.org/players/4/19/192985.png	Be

5 rows × 89 columns

In [6]:
```
# view summary of dataset
```

In [7]:
```
fifa.info()
```

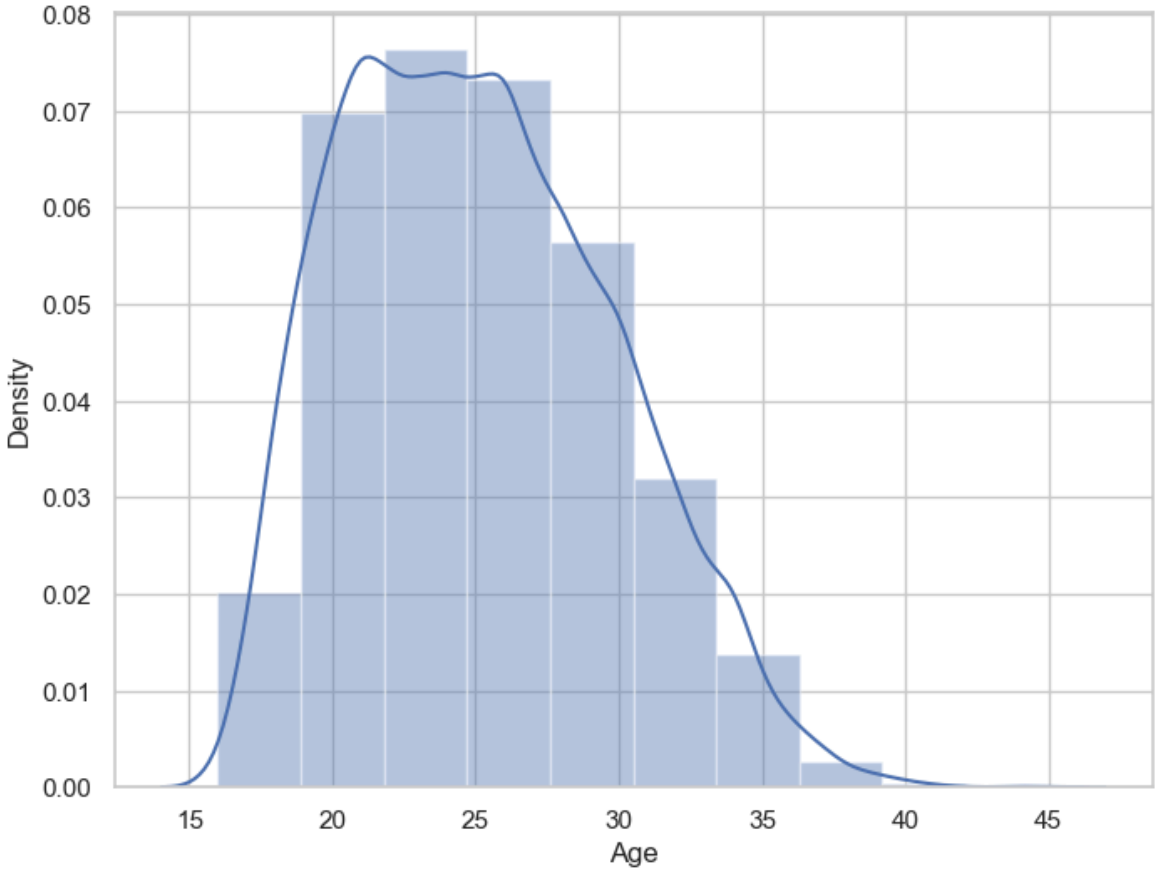
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 18207 entries, 0 to 18206
Data columns (total 89 columns):
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
 0   Unnamed: 0                18207 non-null  int64
 1   ID                        18207 non-null  int64
 2   Name                      18207 non-null  object
 3   Age                       18207 non-null  int64
 4   Photo                     18207 non-null  object
 5   Nationality               18207 non-null  object
 6   Flag                      18207 non-null  object
 7   Overall                   18207 non-null  int64
 8   Potential                 18207 non-null  int64
 9   Club                      17966 non-null  object
 10  Club Logo                 18207 non-null  object
 11  Value                     18207 non-null  object
 12  Wage                      18207 non-null  object
 13  Special                   18207 non-null  int64
 14  Preferred Foot            18159 non-null  object
 15  International Reputation  18159 non-null  float64
 16  Weak Foot                 18159 non-null  float64
 17  Skill Moves               18159 non-null  float64
 18  Work Rate                 18159 non-null  object
 19  Body Type                 18159 non-null  object
 20  Real Face                 18159 non-null  object
 21  Position                  18147 non-null  object
 22  Jersey Number             18147 non-null  float64
 23  Joined                    16654 non-null  object
 24  Loaned From               1264 non-null   object
 25  Contract Valid Until      17918 non-null  object
 26  Height                    18159 non-null  object
 27  Weight                    18159 non-null  object
 28  LS                        16122 non-null  object
 29  ST                        16122 non-null  object
 30  RS                        16122 non-null  object
 31  LW                        16122 non-null  object
 32  LF                        16122 non-null  object
 33  CF                        16122 non-null  object
 34  RF                        16122 non-null  object
 35  RW                        16122 non-null  object
 36  LAM                       16122 non-null  object
 37  CAM                       16122 non-null  object
 38  RAM                       16122 non-null  object
 39  LM                        16122 non-null  object
 40  LCM                       16122 non-null  object
 41  CM                        16122 non-null  object
 42  RCM                       16122 non-null  object
 43  RM                        16122 non-null  object
 44  LWB                       16122 non-null  object
 45  LDM                       16122 non-null  object
 46  CDM                       16122 non-null  object
 47  RDM                       16122 non-null  object
 48  RWB                       16122 non-null  object
 49  LB                        16122 non-null  object
 50  LCB                       16122 non-null  object
 51  CB                        16122 non-null  object
 52  RCB                       16122 non-null  object
 53  RB                        16122 non-null  object
 54  Crossing                  18159 non-null  float64
```

```
 55  Finishing                 18159 non-null  float64
 56  HeadingAccuracy           18159 non-null  float64
 57  ShortPassing              18159 non-null  float64
 58  Volleys                   18159 non-null  float64
 59  Dribbling                 18159 non-null  float64
 60  Curve                     18159 non-null  float64
 61  FKAccuracy                18159 non-null  float64
 62  LongPassing               18159 non-null  float64
 63  BallControl               18159 non-null  float64
 64  Acceleration              18159 non-null  float64
 65  SprintSpeed               18159 non-null  float64
 66  Agility                   18159 non-null  float64
 67  Reactions                 18159 non-null  float64
 68  Balance                   18159 non-null  float64
 69  ShotPower                 18159 non-null  float64
 70  Jumping                   18159 non-null  float64
 71  Stamina                   18159 non-null  float64
 72  Strength                  18159 non-null  float64
 73  LongShots                 18159 non-null  float64
 74  Aggression                18159 non-null  float64
 75  Interceptions             18159 non-null  float64
 76  Positioning               18159 non-null  float64
 77  Vision                    18159 non-null  float64
 78  Penalties                 18159 non-null  float64
 79  Composure                 18159 non-null  float64
 80  Marking                   18159 non-null  float64
 81  StandingTackle            18159 non-null  float64
 82  SlidingTackle             18159 non-null  float64
 83  GKDiving                  18159 non-null  float64
 84  GKHandling                18159 non-null  float64
 85  GKKicking                 18159 non-null  float64
 86  GKPositioning             18159 non-null  float64
 87  GKReflexes                18159 non-null  float64
 88  Release Clause            16643 non-null  object
dtypes: float64(38), int64(6), object(45)
memory usage: 12.4+ MB
```

In [8]:
```
fifa['Body Type'].value_counts()
```

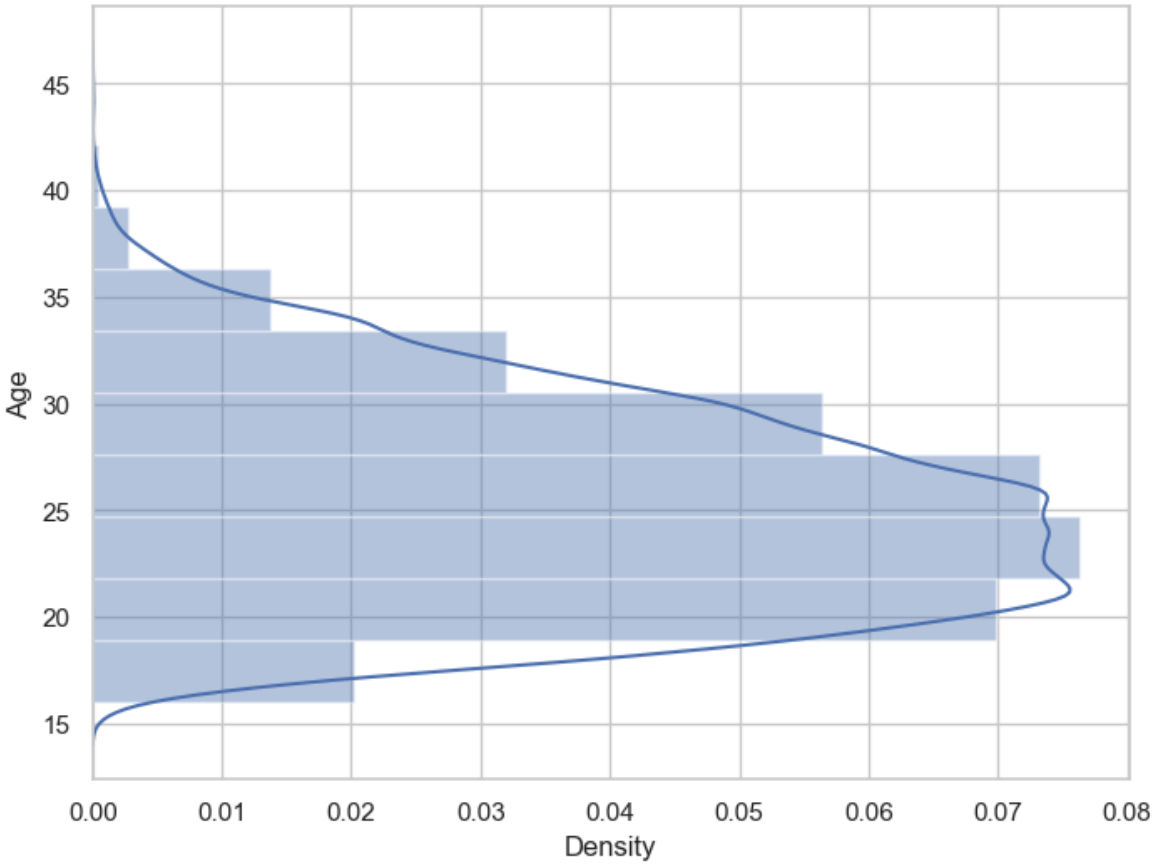
Out[8]:
```
Body Type
Normal                10595
Lean                   6417
Stocky                 1140
Messi                     1
C. Ronaldo                1
Neymar                    1
Courtois                  1
PLAYER_BODY_TYPE_25       1
Shaqiri                   1
Akinfenwa                 1
Name: count, dtype: int64
```

In [9]:
```
f,ax = plt.subplots(figsize=(8,6))
x = fifa['Age']
ax = sns.distplot(x,bins=10)
plt.show()
```



we can plot the distribution on the vertical axis

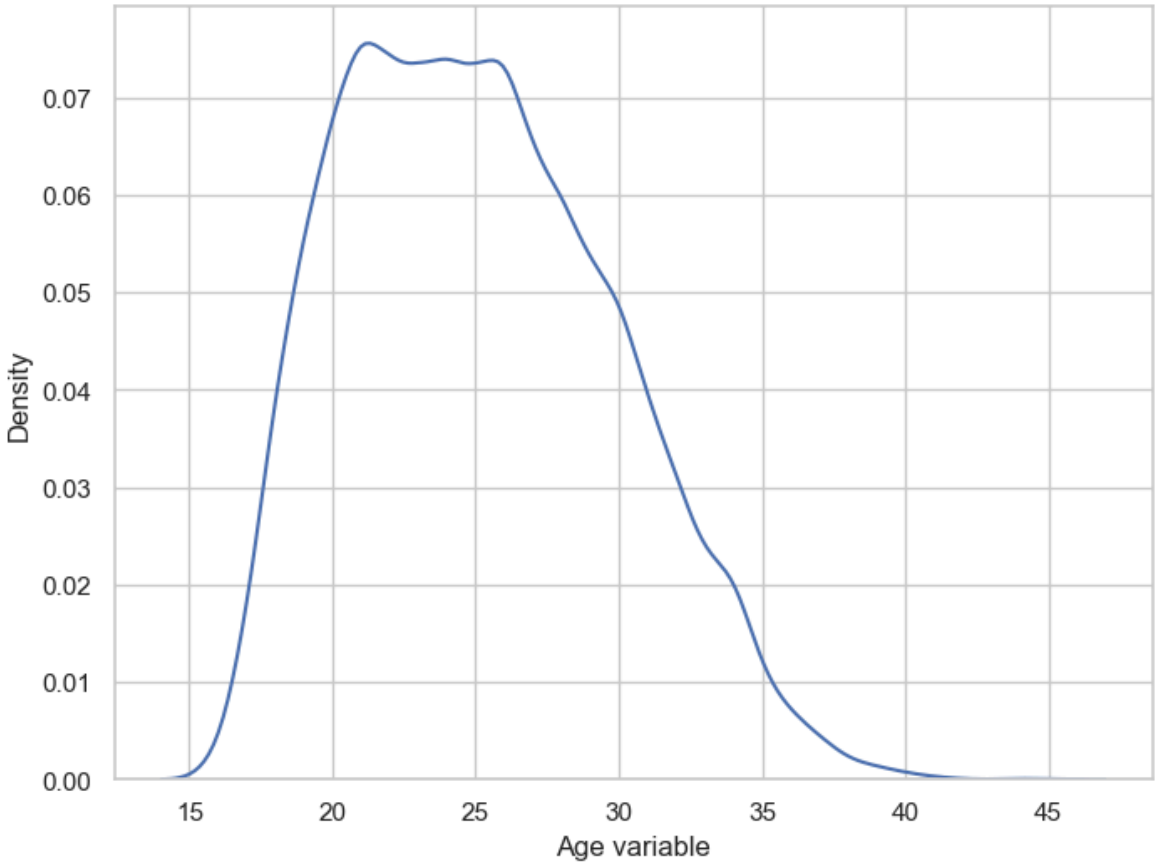
In [11]:
```
f,ax = plt.subplots(figsize=(8,6))
x = fifa['Age']
ax = sns.distplot(x,bins=10 , vertical = True)
plt.show()
```



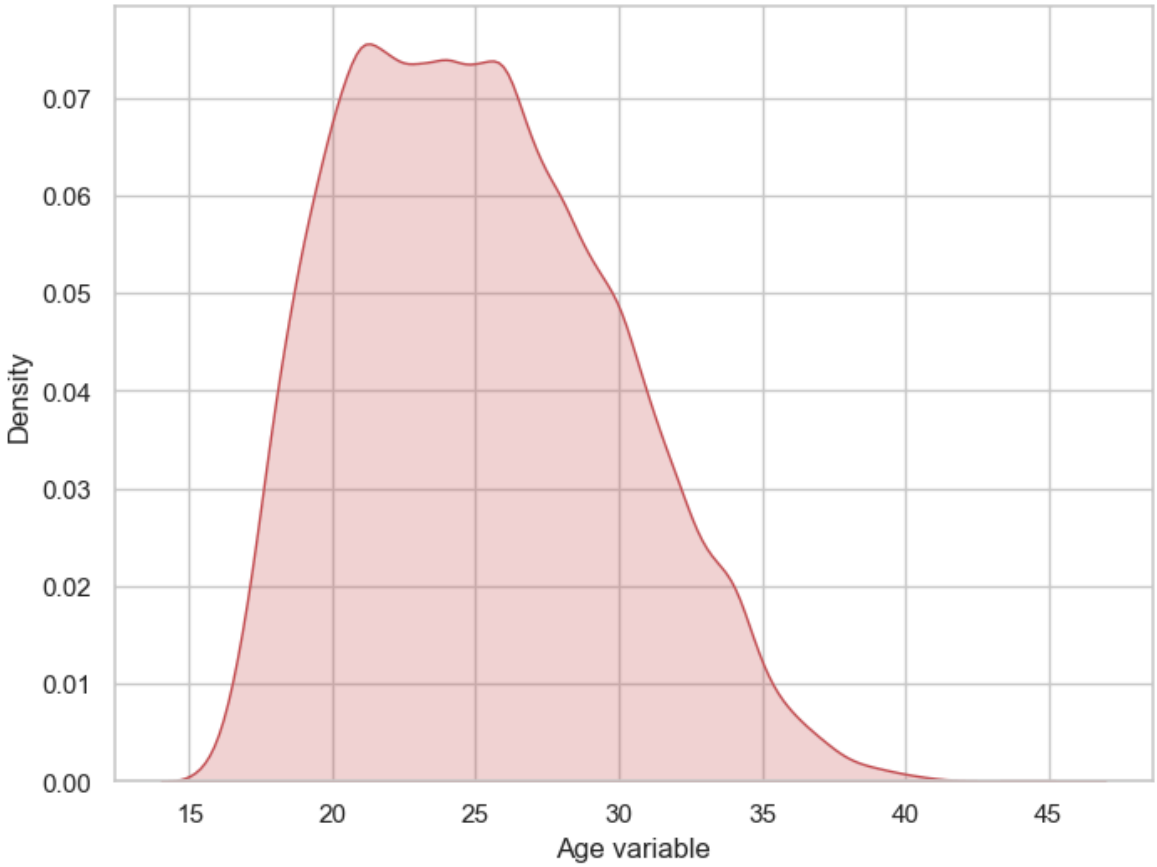
SEABORN KERNAL DENSITY ESTIMATION (KDE) PLOT

In [13]:
```
import pandas as pd
```

In [14]:
```
f,ax = plt.subplots(figsize=(8,6))
x = fifa['Age']
x = pd.Series(x,name ="Age variable")
ax = sns.kdeplot(x)
plt.show()
```

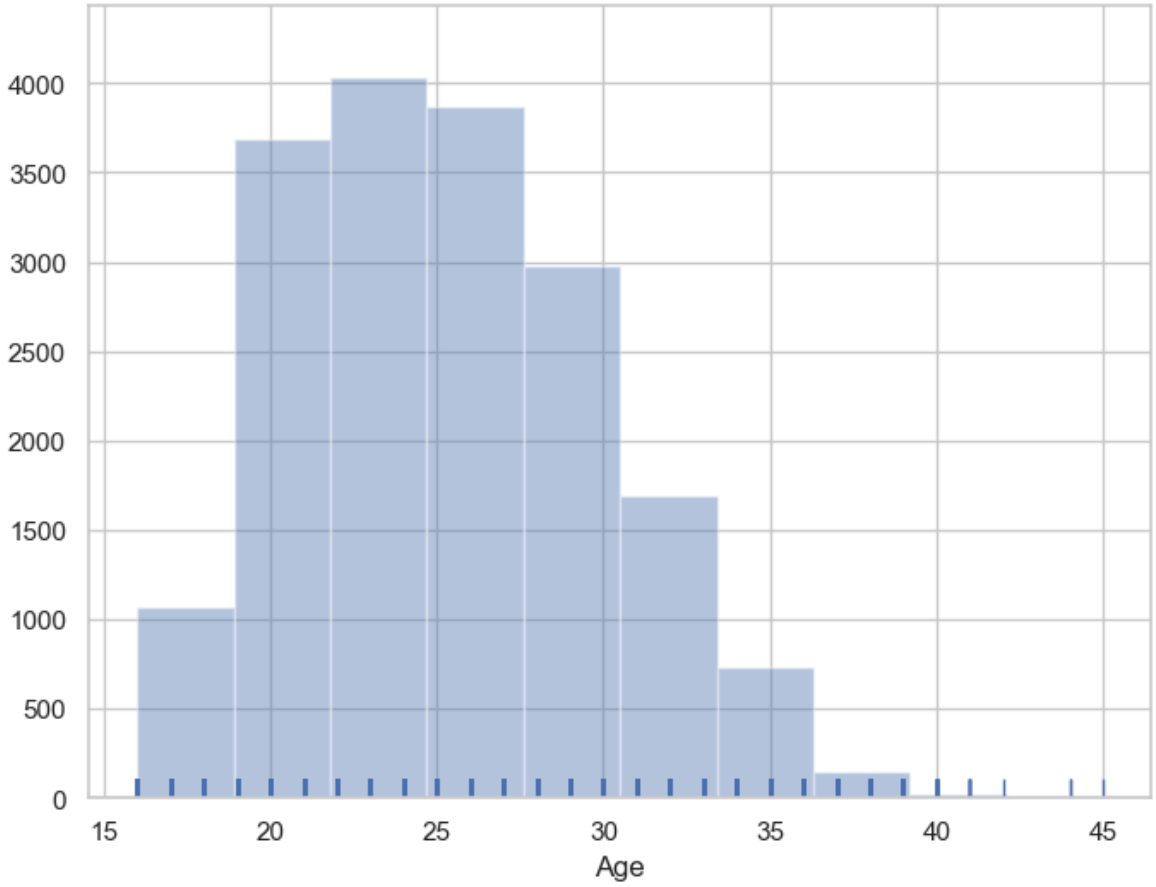


In [15]:
```
f, ax = plt.subplots(figsize=(8,6))
x = fifa['Age']
x = pd.Series(x, name="Age variable")
ax = sns.kdeplot(x, shade=True, color='r')
plt.show()
```

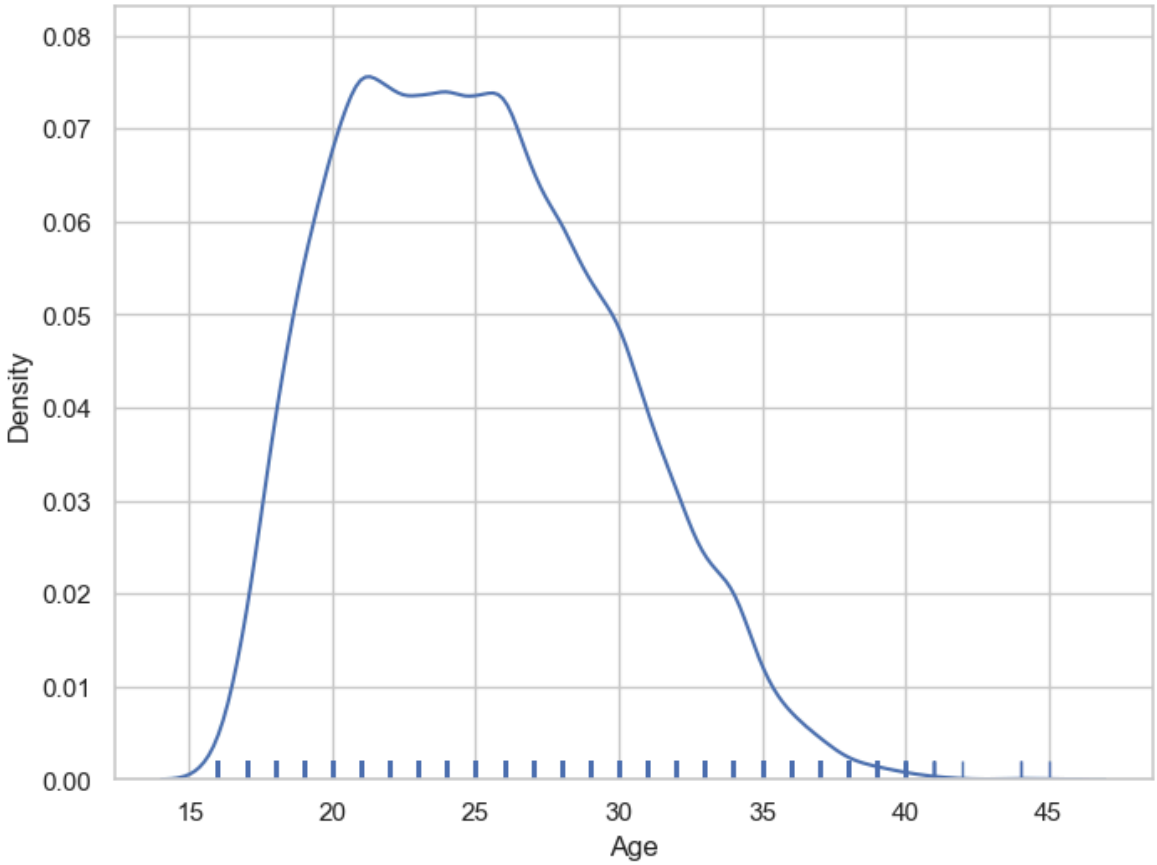


HISTOGRAMS

In [17]:
```
f,ax =  plt.subplots(figsize=(8,6))
x = fifa['Age']
ax = sns.distplot(x,kde= False , rug=True,bins=10)
plt.show()
```



In [18]:
```
f, ax = plt.subplots(figsize=(8,6))
x = fifa['Age']
ax = sns.distplot(x, hist=False, rug=True, bins=10)
plt.show()
```



EXPLORE PREFERRED FOOT VARIABLE

In [20]:
```
fifa['Preferred Foot'].nunique()
```

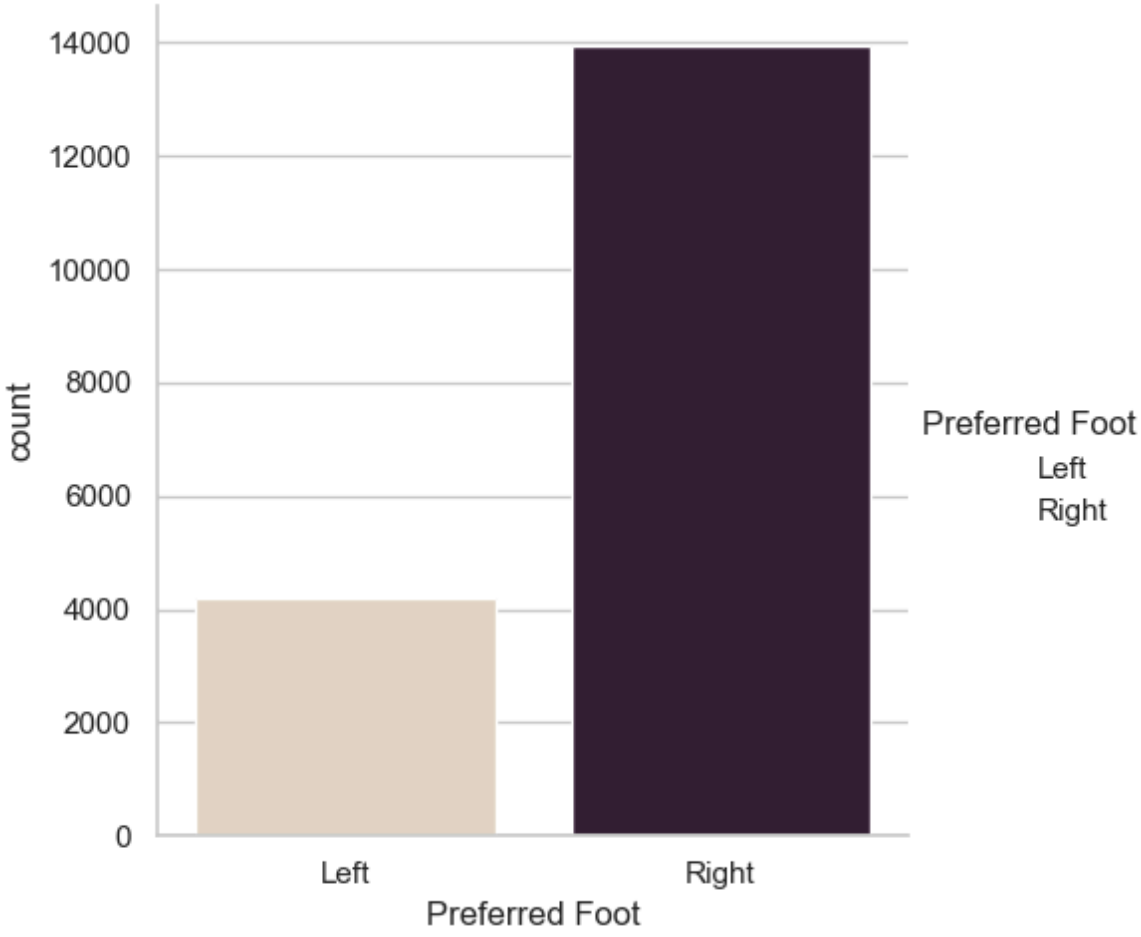
Out[20]:
```
2
```

In [21]:
```
fifa['Preferred Foot'].value_counts()
```

Out[21]:
```
Preferred Foot
Right    13948
Left      4211
Name: count, dtype: int64
```

Visualize distribution of values with Seaborn `countplot()` function

In [23]:
```
g = sns.catplot(x="Preferred Foot", kind="count", palette="ch:.25", data=fifa)
plt.show()
```



Explore `International Reputation` variable

In [25]:
```
fifa['International Reputation'].nunique()
```

Out[25]:
```
5
```

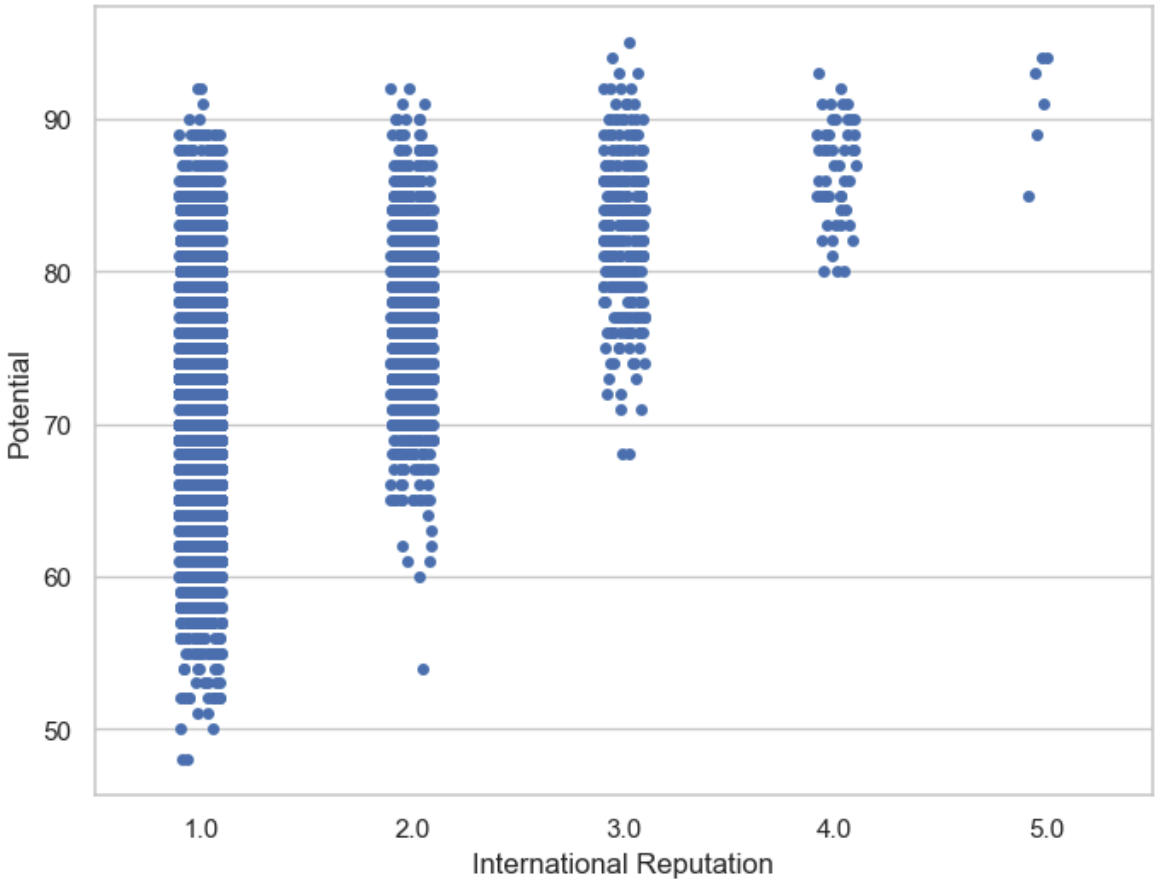
Check the distribution of values in `International Reputation` variable

In [27]:
```
fifa['International Reputation'].value_counts()
```

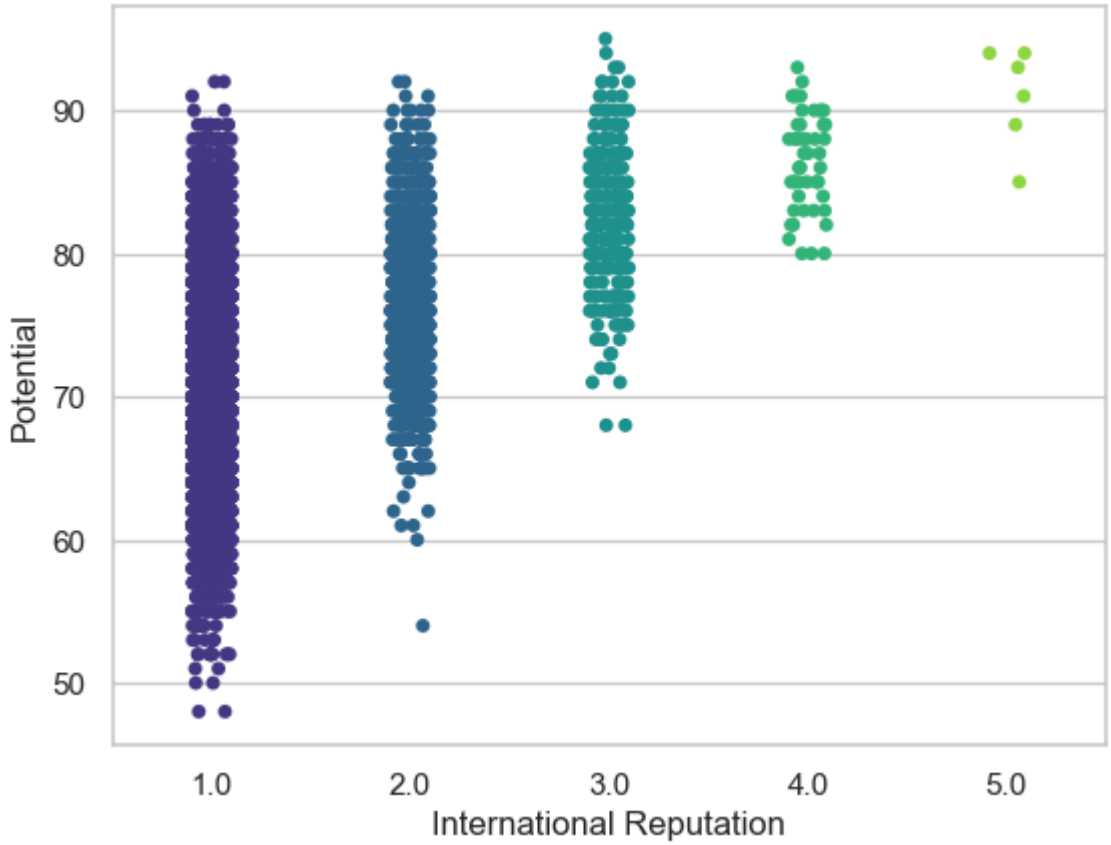
Out[27]:
```
International Reputation
1.0    16532
2.0     1261
3.0      309
4.0       51
5.0        6
Name: count, dtype: int64
```

Seaborn `Stripplot()` function.

In [29]:
```
f, ax = plt.subplots(figsize=(8, 6))
sns.stripplot(x="International Reputation", y="Potential", data=fifa)
plt.show()
```

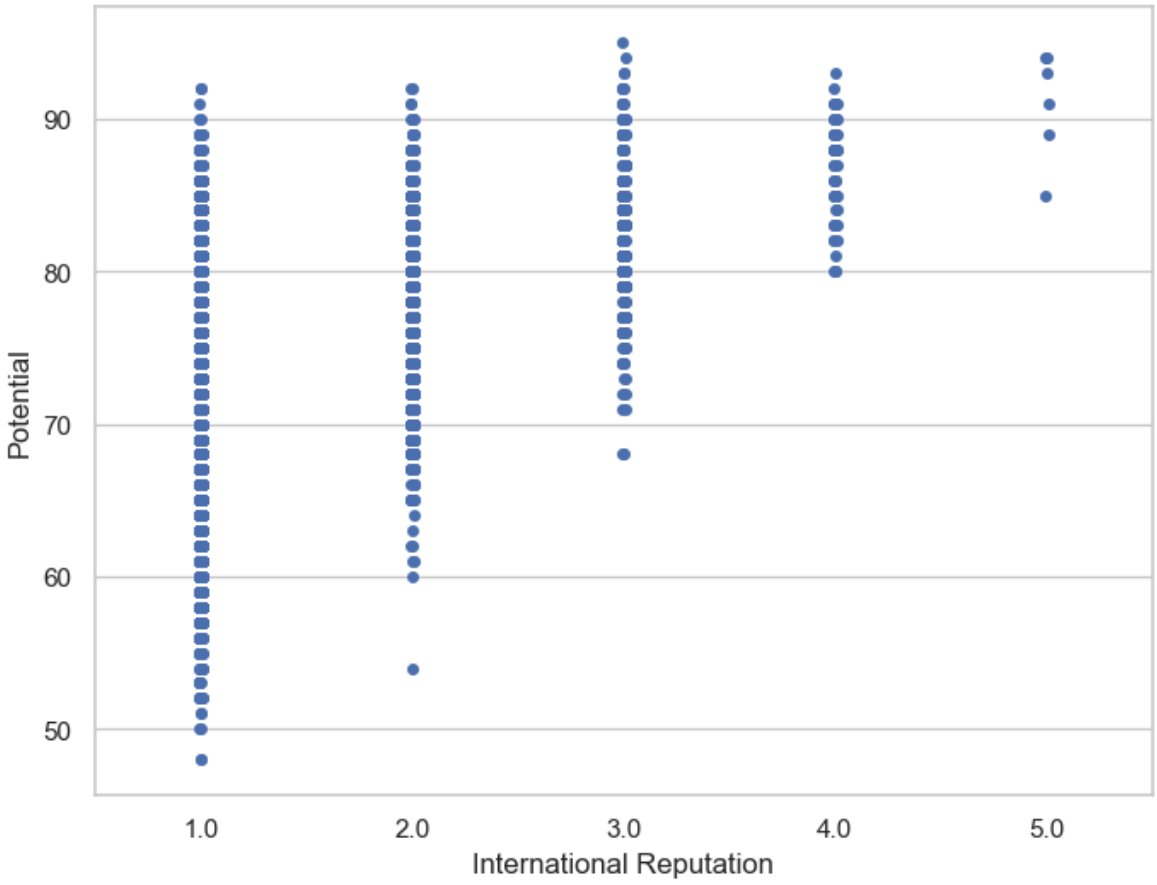


In [30]:
```
sns.stripplot(x="International Reputation", y="Potential", data=fifa, palette='v
plt.show()
```

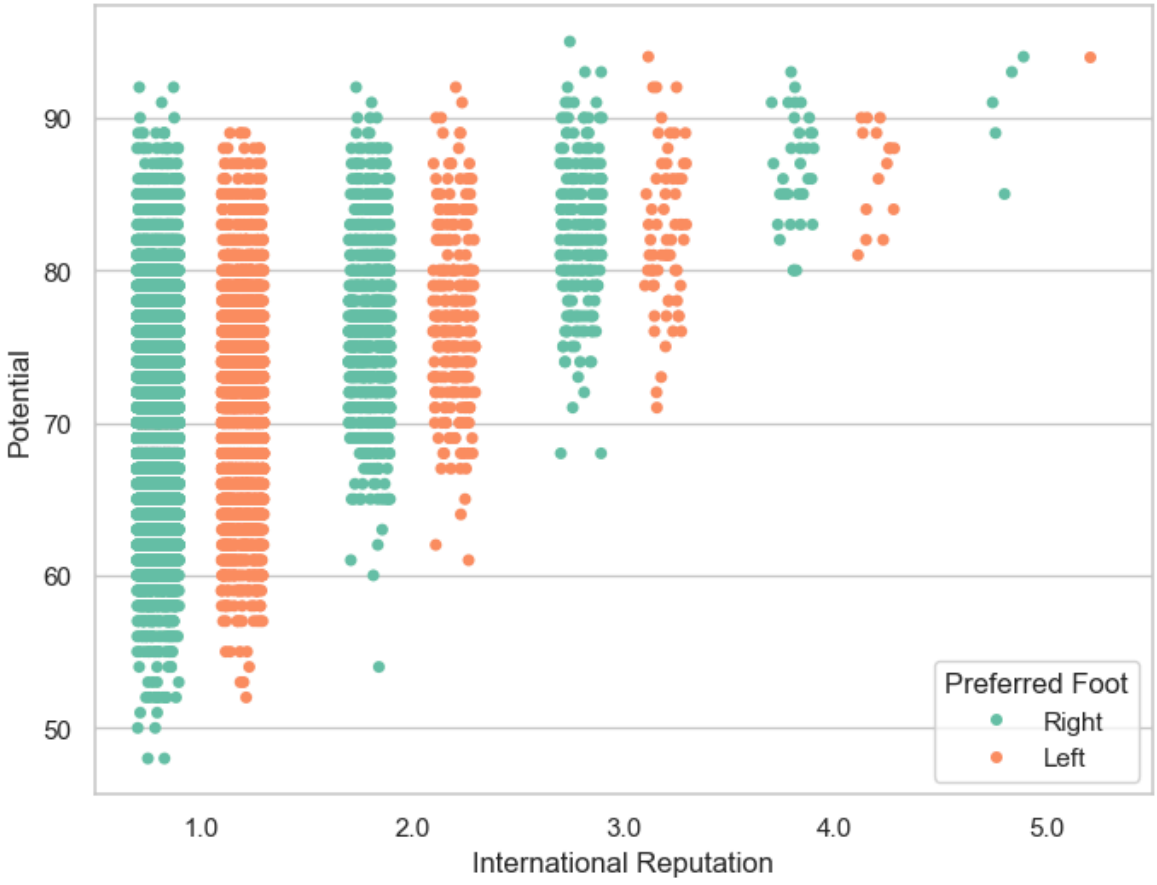


We can add jitter to bring out the distribution of values as follows-

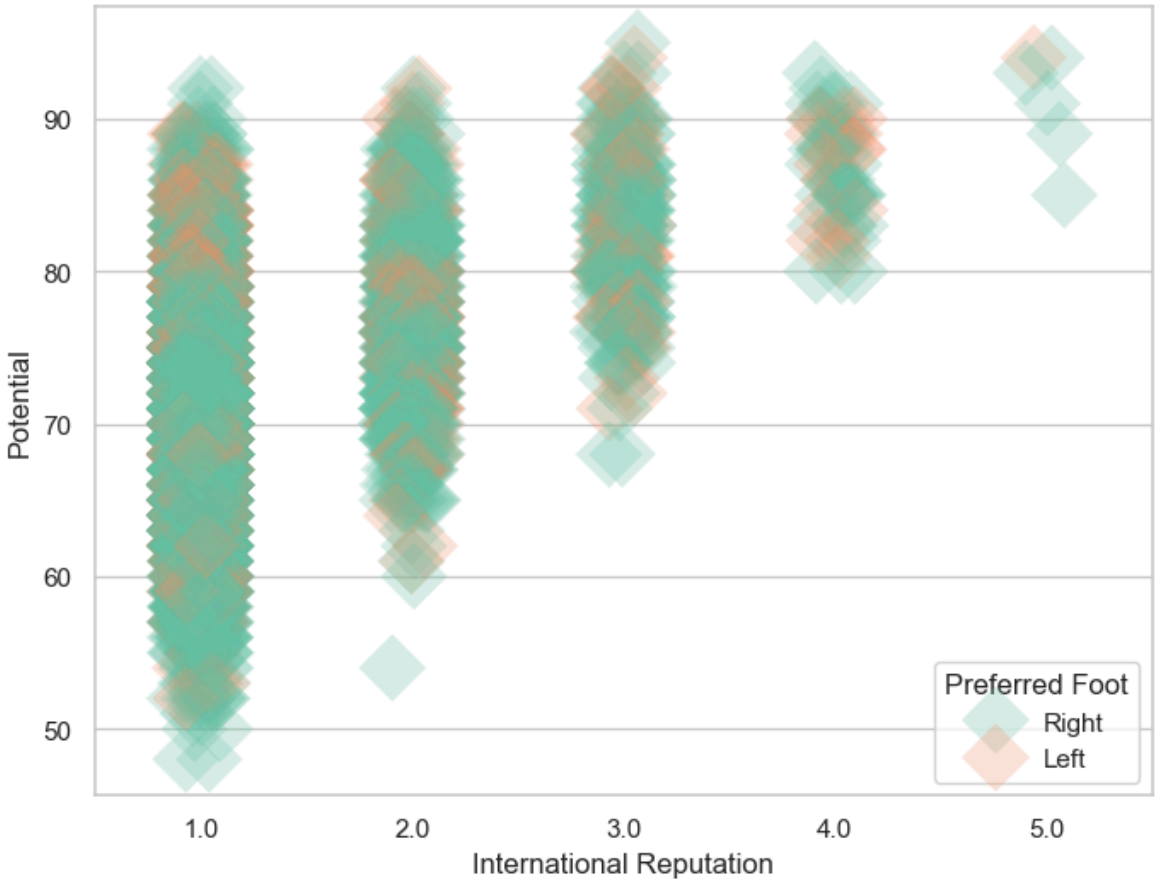
In [32]:
```
f, ax = plt.subplots(figsize=(8, 6))
sns.stripplot(x="International Reputation", y="Potential", data=fifa, jitter=0.0
plt.show()
```



In [33]:
```
f, ax = plt.subplots(figsize=(8, 6))
sns.stripplot(x="International Reputation", y="Potential", hue="Preferred Foot",
                    data=fifa, jitter=0.2, palette="Set2", dodge=True)
plt.show()
```

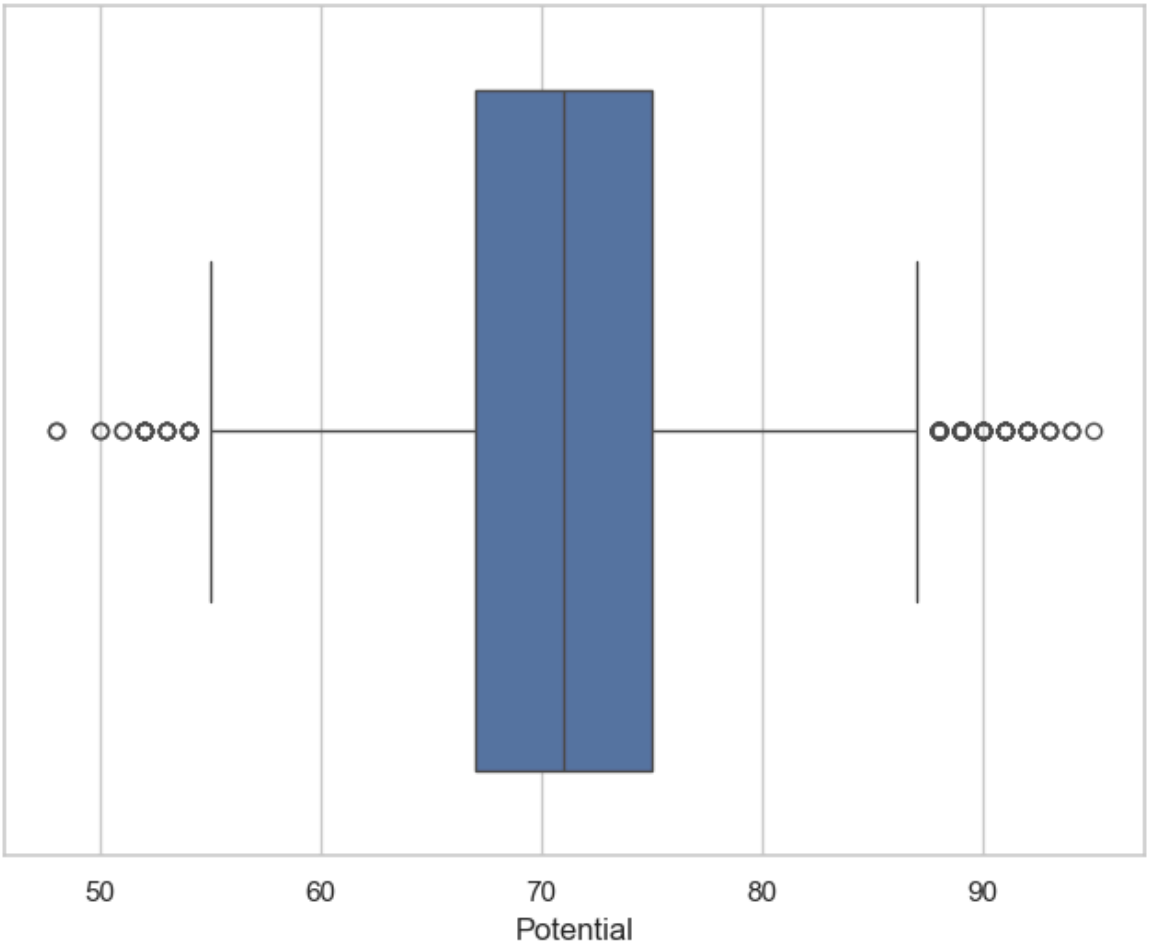


In [34]:
```
f, ax = plt.subplots(figsize=(8, 6))
sns.stripplot(x="International Reputation", y="Potential", hue="Preferred Foot",
plt.show()
```

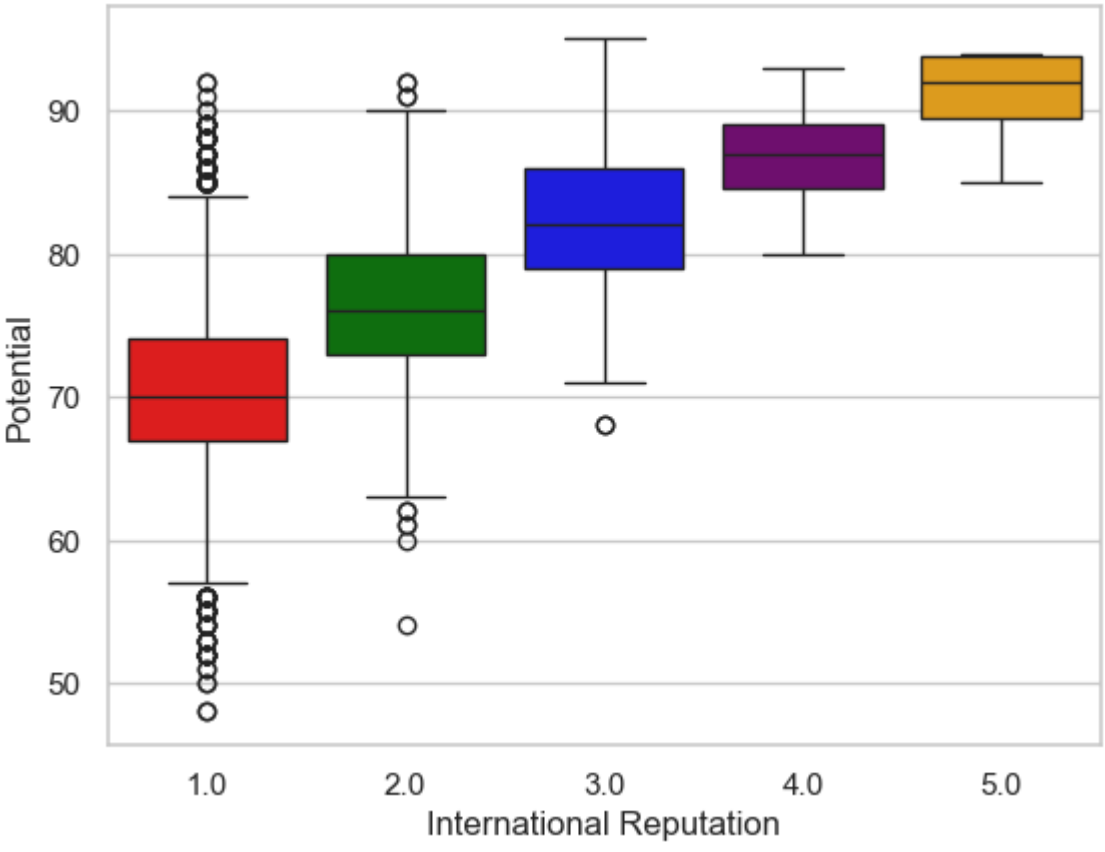


SEABORN BOXPLOT FUNCTION

In [36]:
```
f,ax = plt.subplots(figsize=(8,6))
sns.boxplot(x = fifa["Potential"])
plt.show()
```

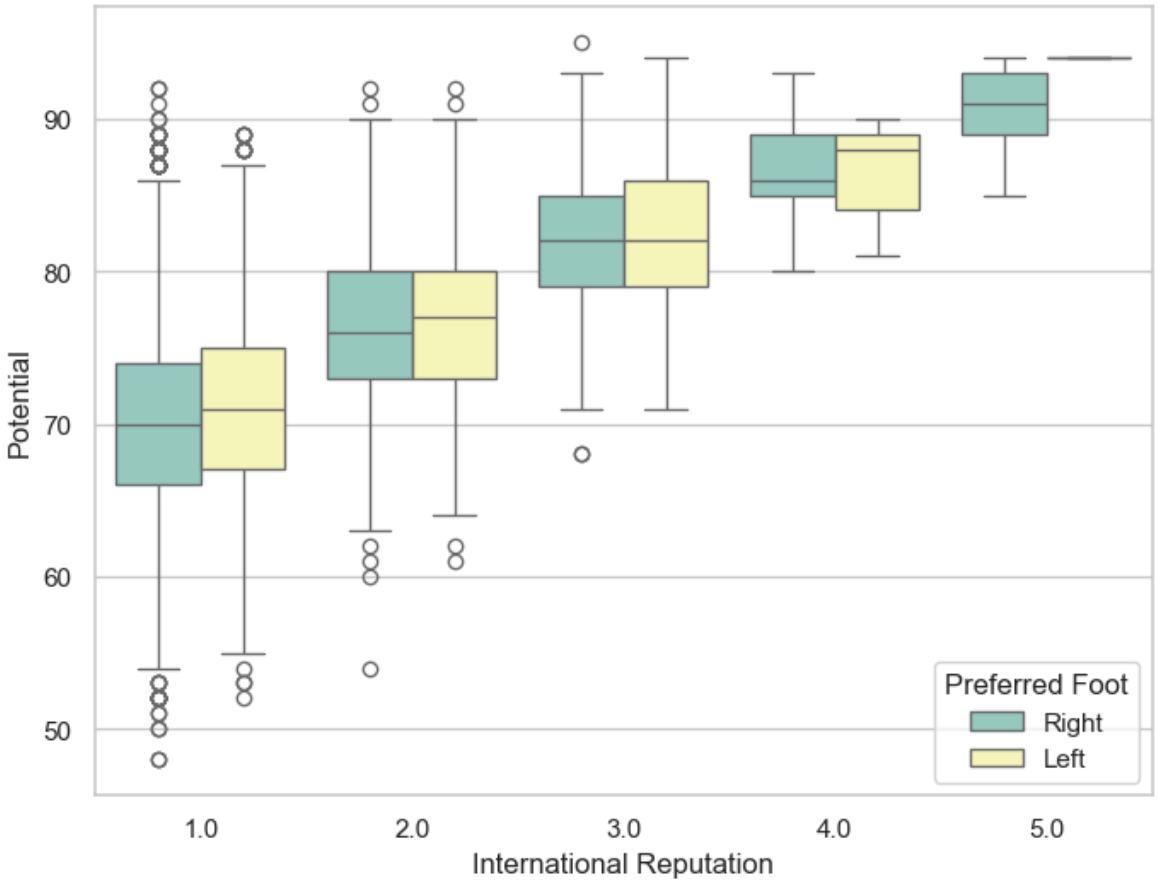


In [37]:
```
sns.boxplot(x="International Reputation", y="Potential", data=fifa, palette=['re
plt.show()
```



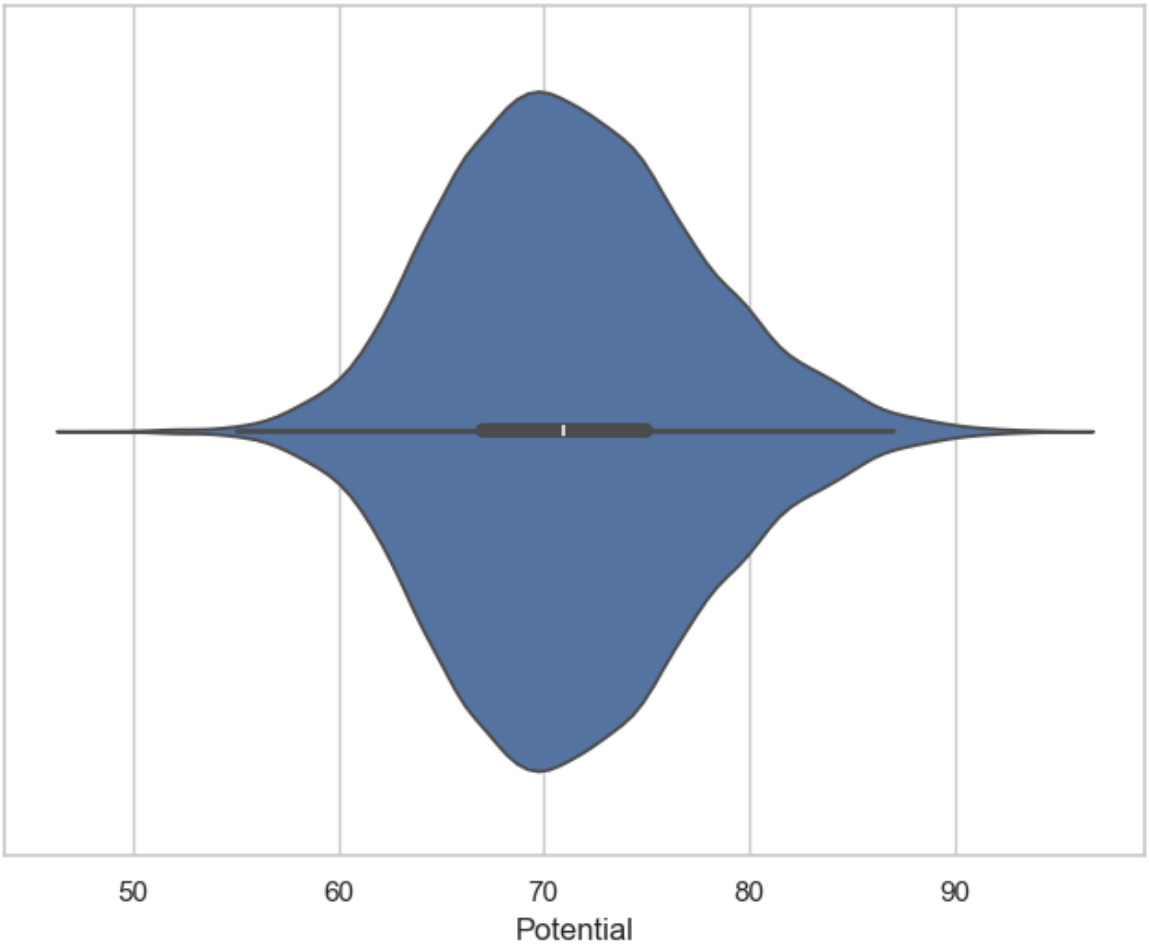
We can draw a boxplot with nested grouping by two categorical variables as follows-

In [39]:
```
f, ax = plt.subplots(figsize=(8, 6))
sns.boxplot(x="International Reputation", y="Potential", hue="Preferred Foot", d
plt.show()
```

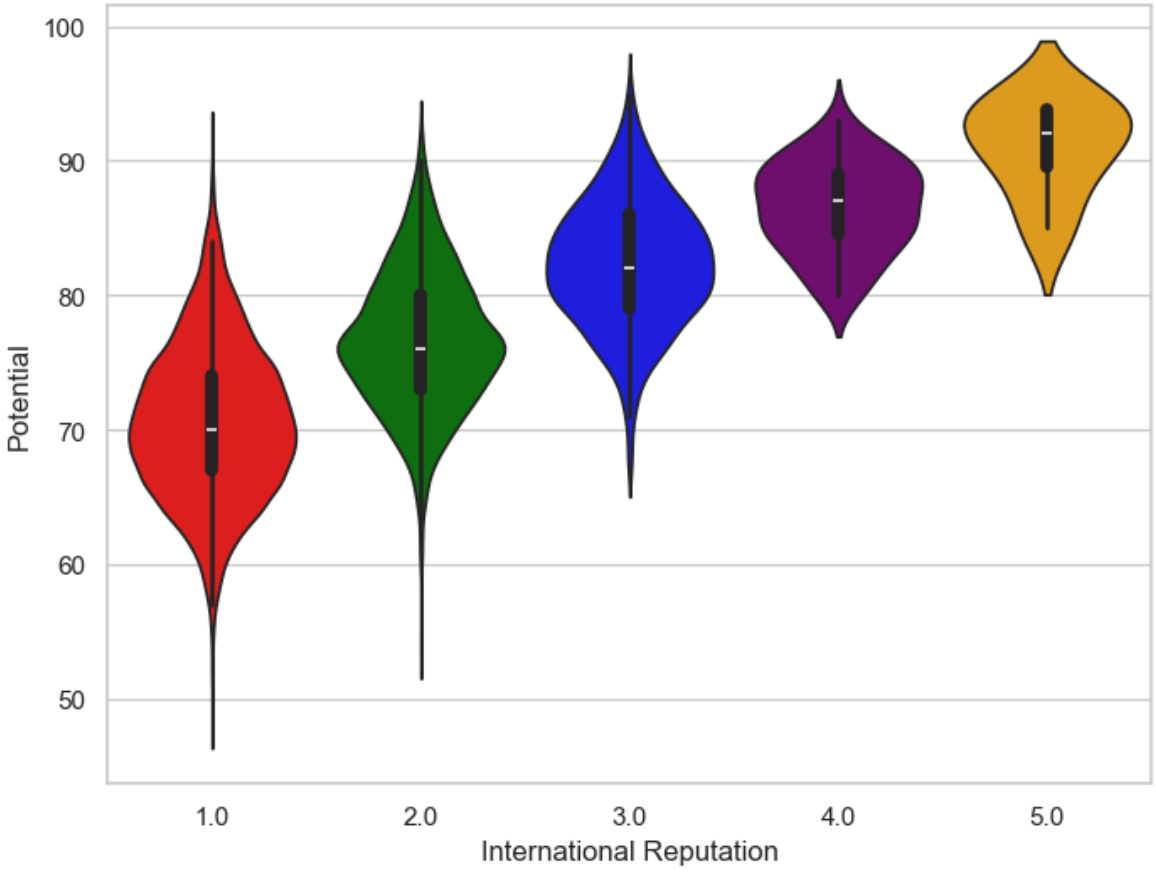


SEABORN violinplot() function

In [41]:
```
f, ax = plt.subplots(figsize=(8, 6))
sns.violinplot(x=fifa["Potential"])
plt.show()
```

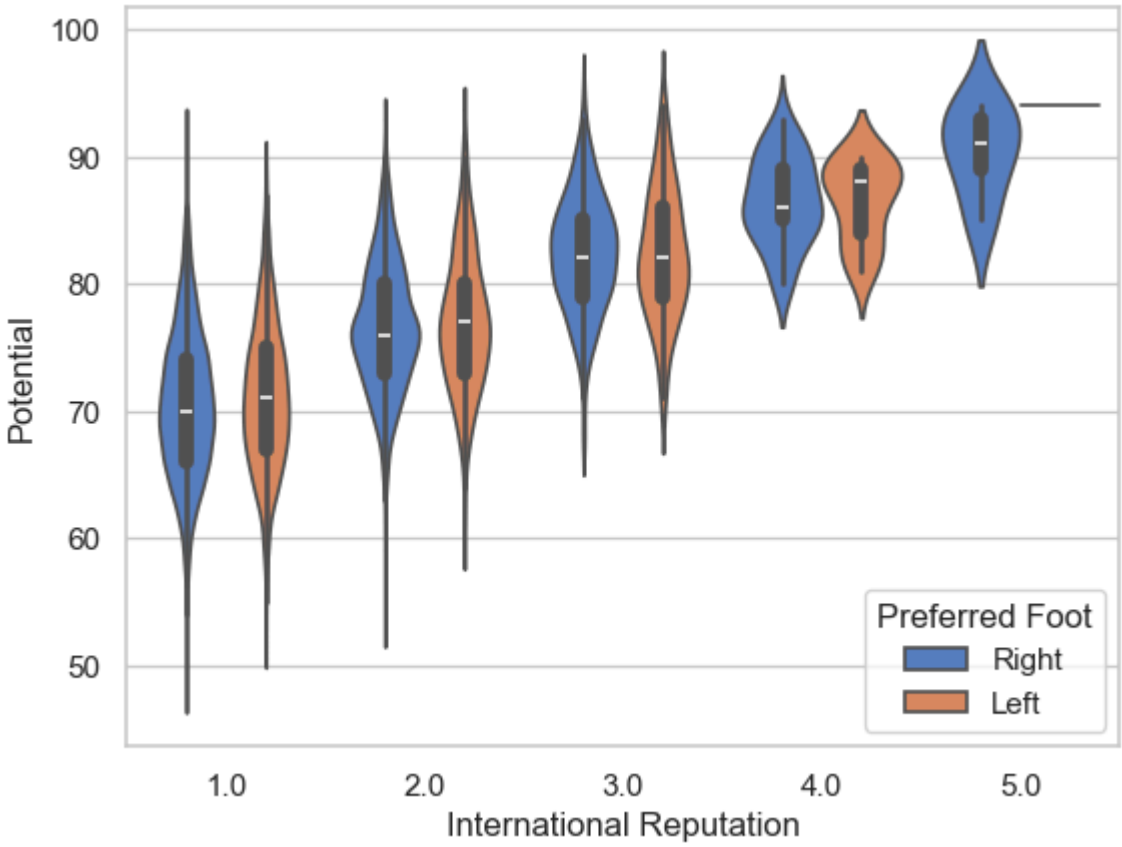


In [42]:
```
f, ax = plt.subplots(figsize=(8, 6))
sns.violinplot(x="International Reputation", y="Potential", data=fifa, palette=[
plt.show()
```

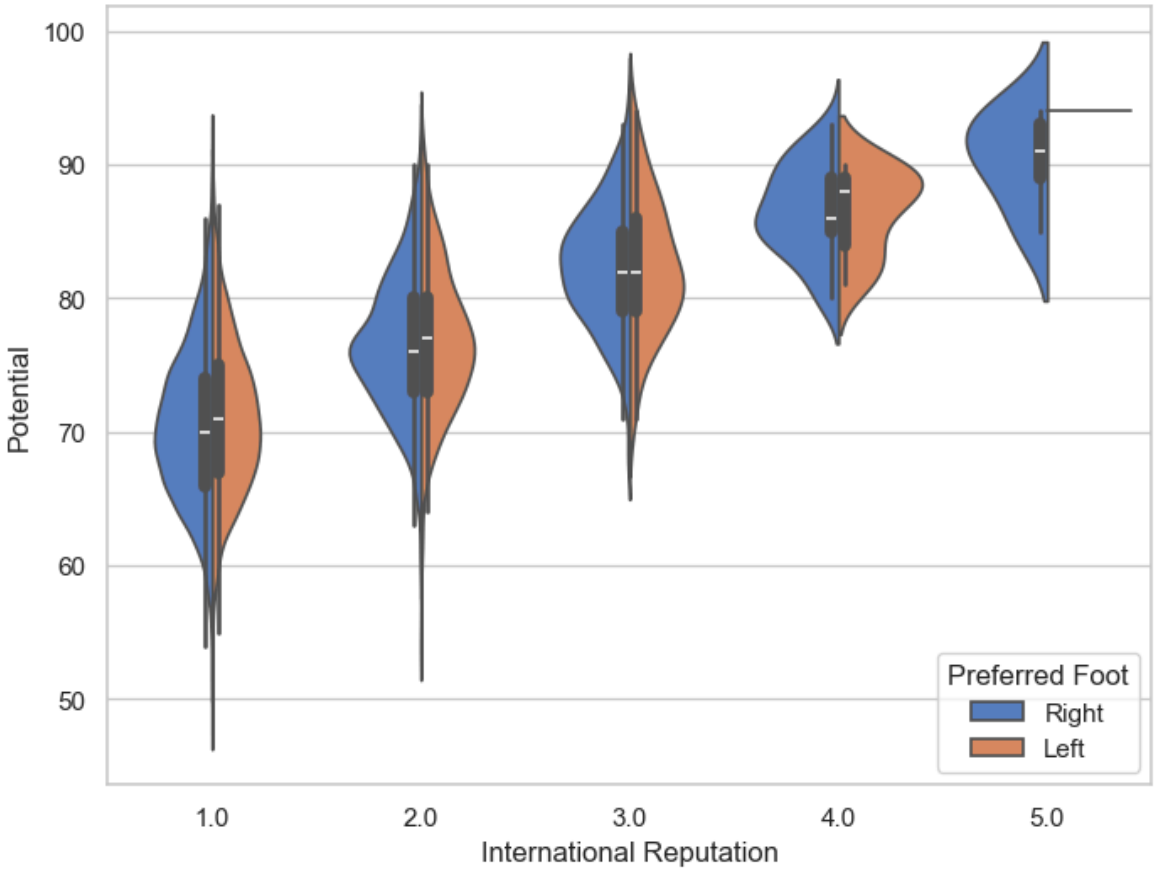


we can draw a violinplot with nested grouping by two categorical variables

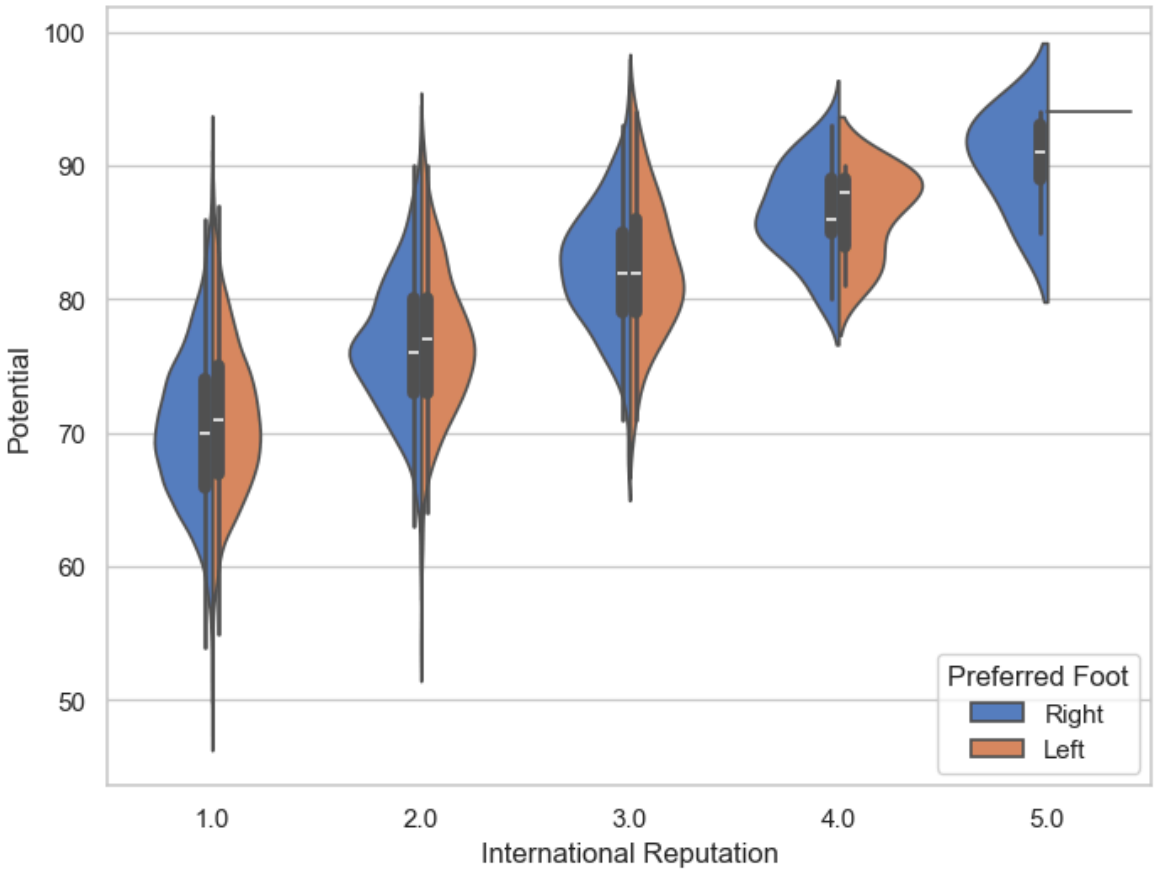
In [44]:
```
sns.violinplot(x="International Reputation", y="Potential", hue="Preferred Foot"
plt.show()
```



In [45]:
```
f, ax = plt.subplots(figsize=(8, 6))
sns.violinplot(x="International Reputation", y="Potential", hue="Preferred Foot"
               data=fifa, palette="muted", split=True)
plt.show()
```

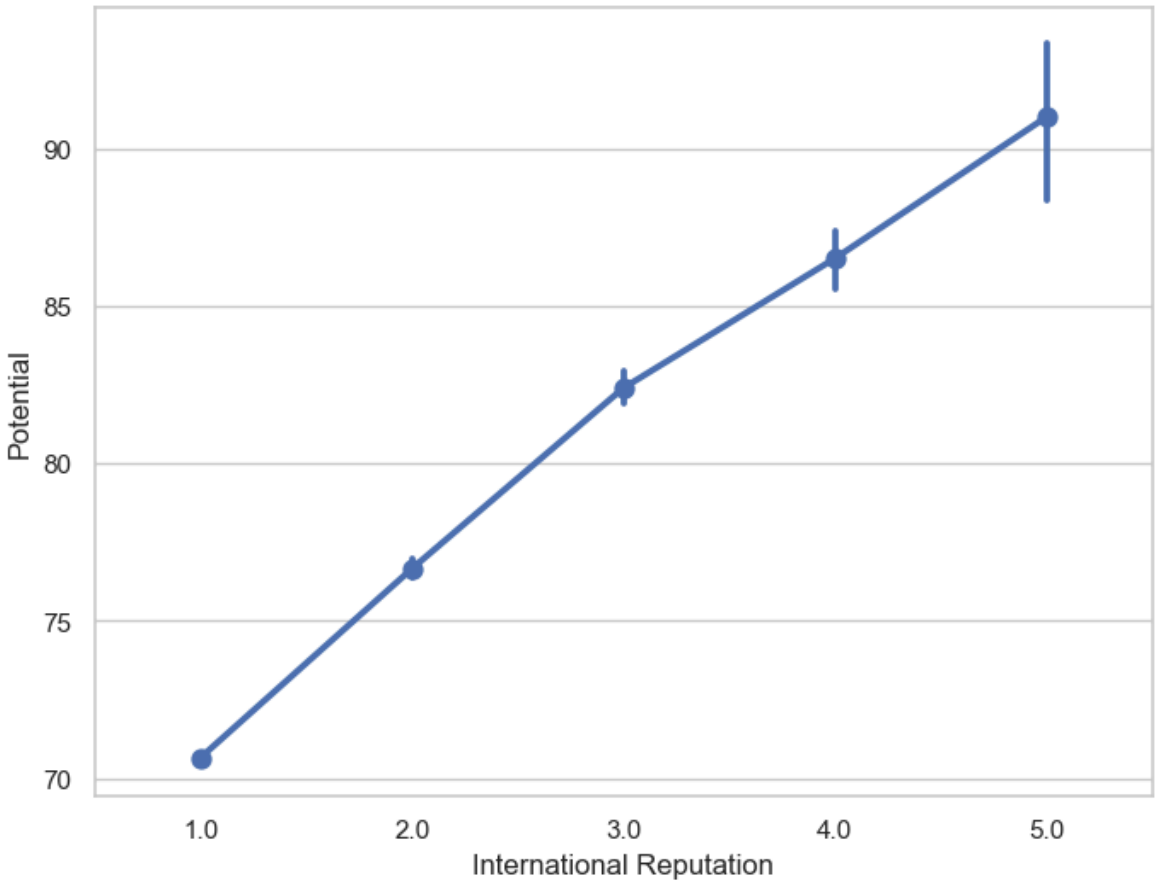


In [46]:
```
f, ax = plt.subplots(figsize=(8, 6))
sns.violinplot(x="International Reputation", y="Potential", hue="Preferred Foot"
               data=fifa, palette="muted", split=True)
plt.show()
```



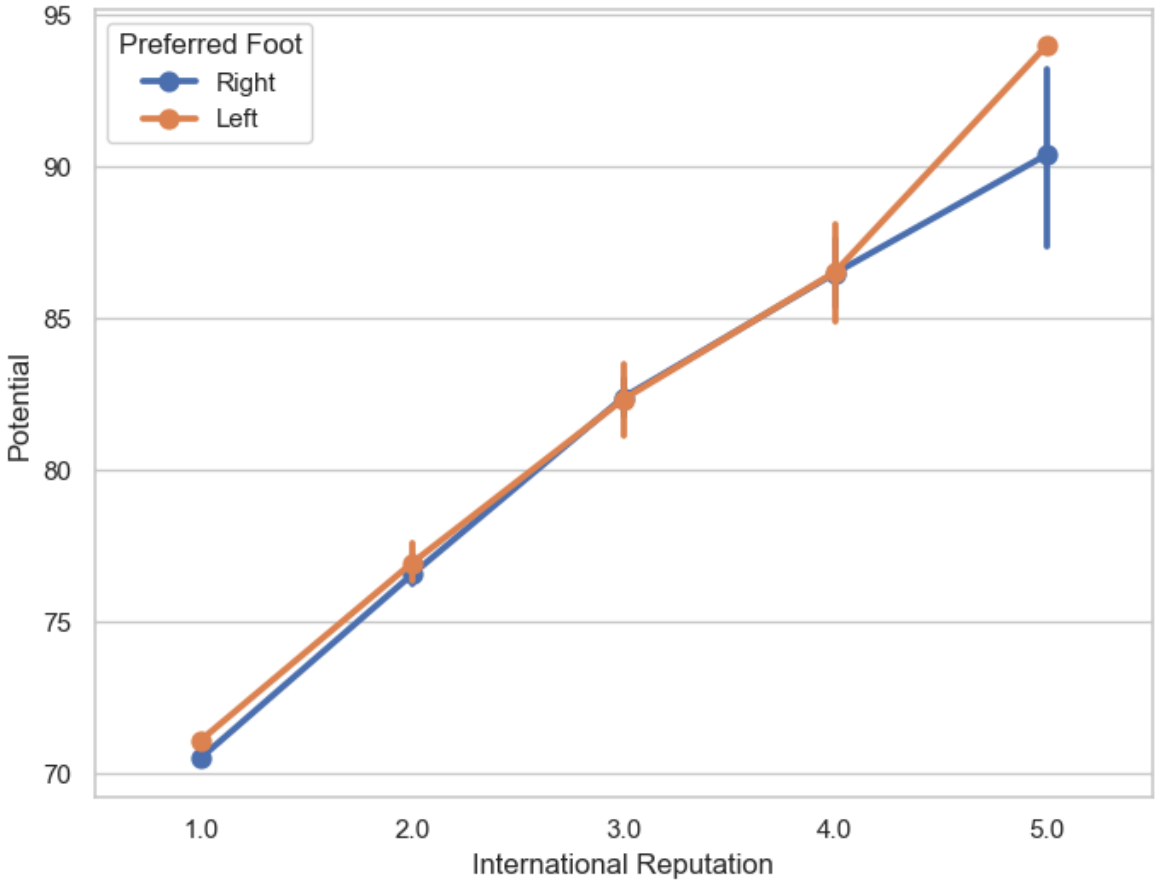
seaborn pointplot() function

In [48]:
```
f,ax= plt.subplots(figsize=(8,6))
sns.pointplot(x="International Reputation",y = "Potential",data= fifa)
plt.show()
```

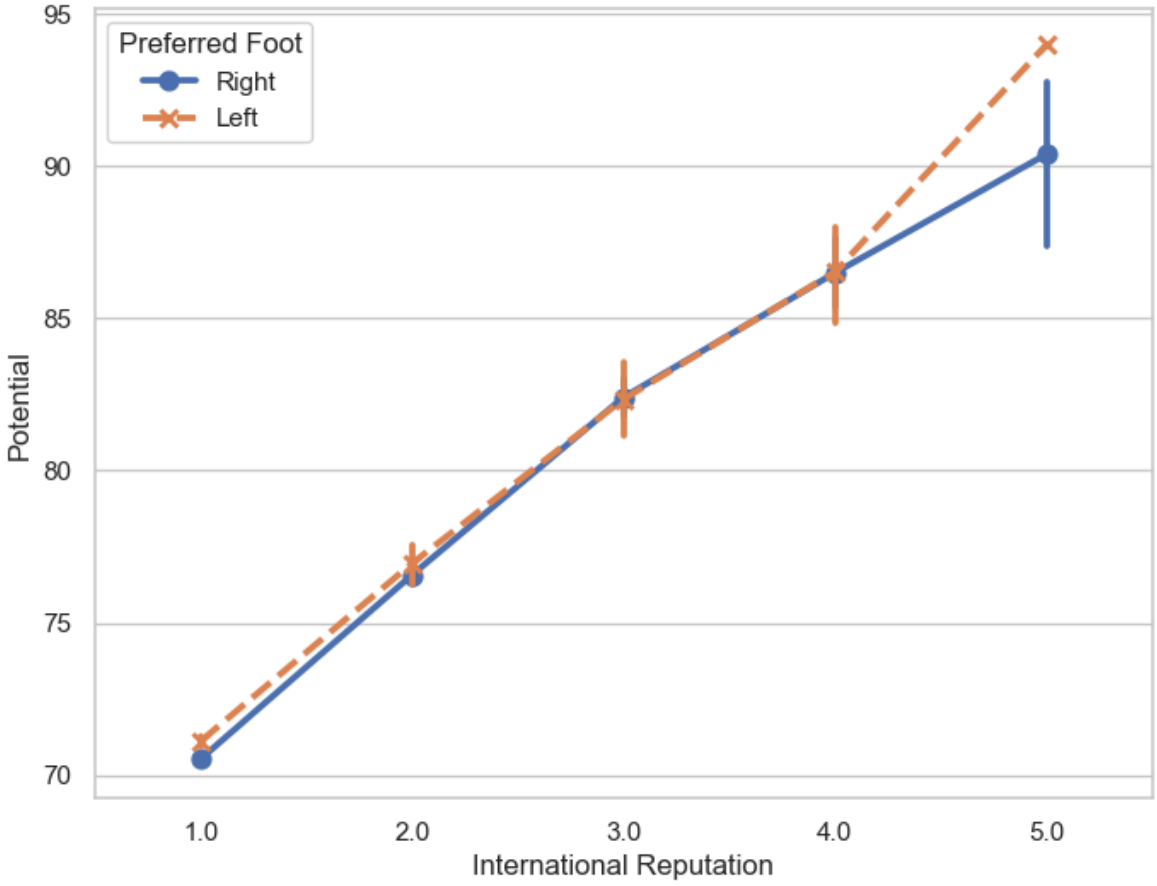


We can draw a set of vertical points with nested grouping by a two variables as follows-

In [50]:
```
f, ax = plt.subplots(figsize=(8, 6))
sns.pointplot(x="International Reputation", y="Potential", hue="Preferred Foot",
plt.show()
```

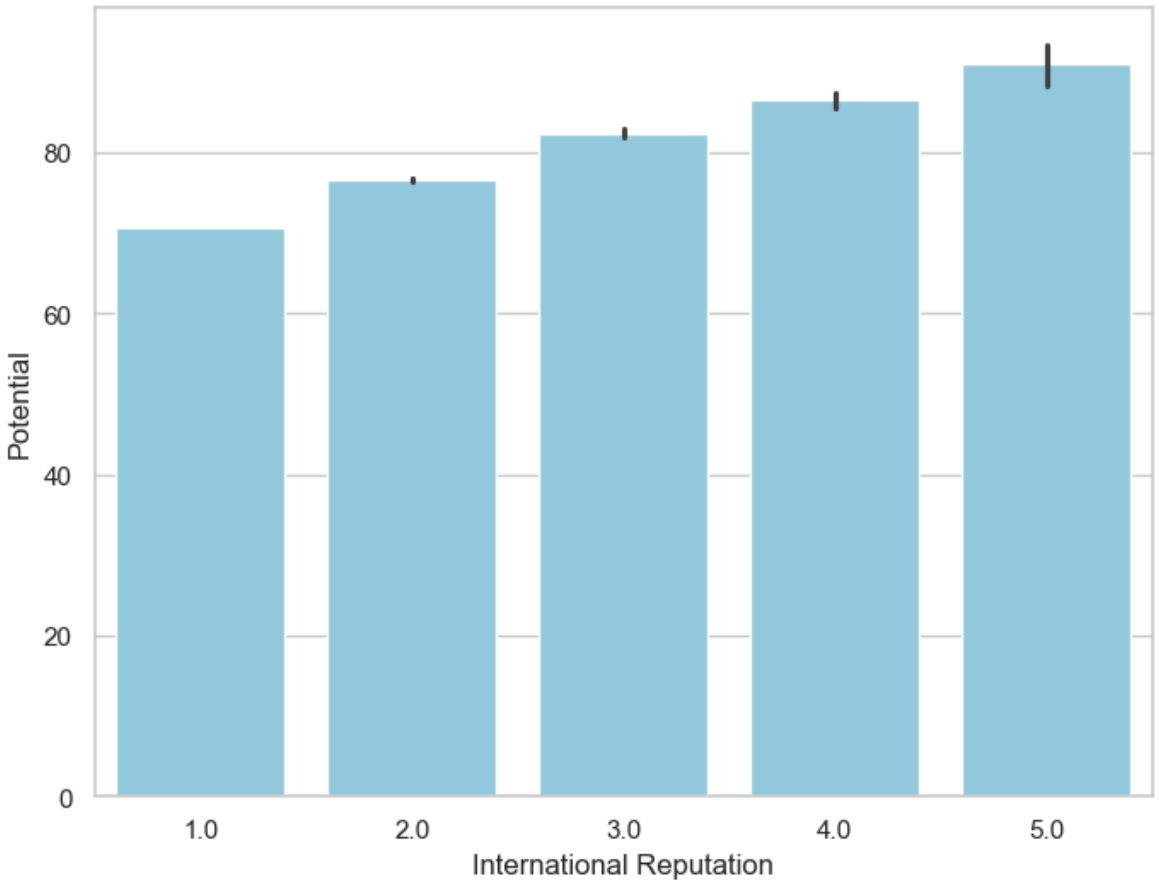


In [51]:
```
f, ax = plt.subplots(figsize=(8, 6))
sns.pointplot(x="International Reputation", y="Potential", hue="Preferred Foot",
              data=fifa, markers=["o", "x"], linestyles=["-", "--"])
plt.show()
```

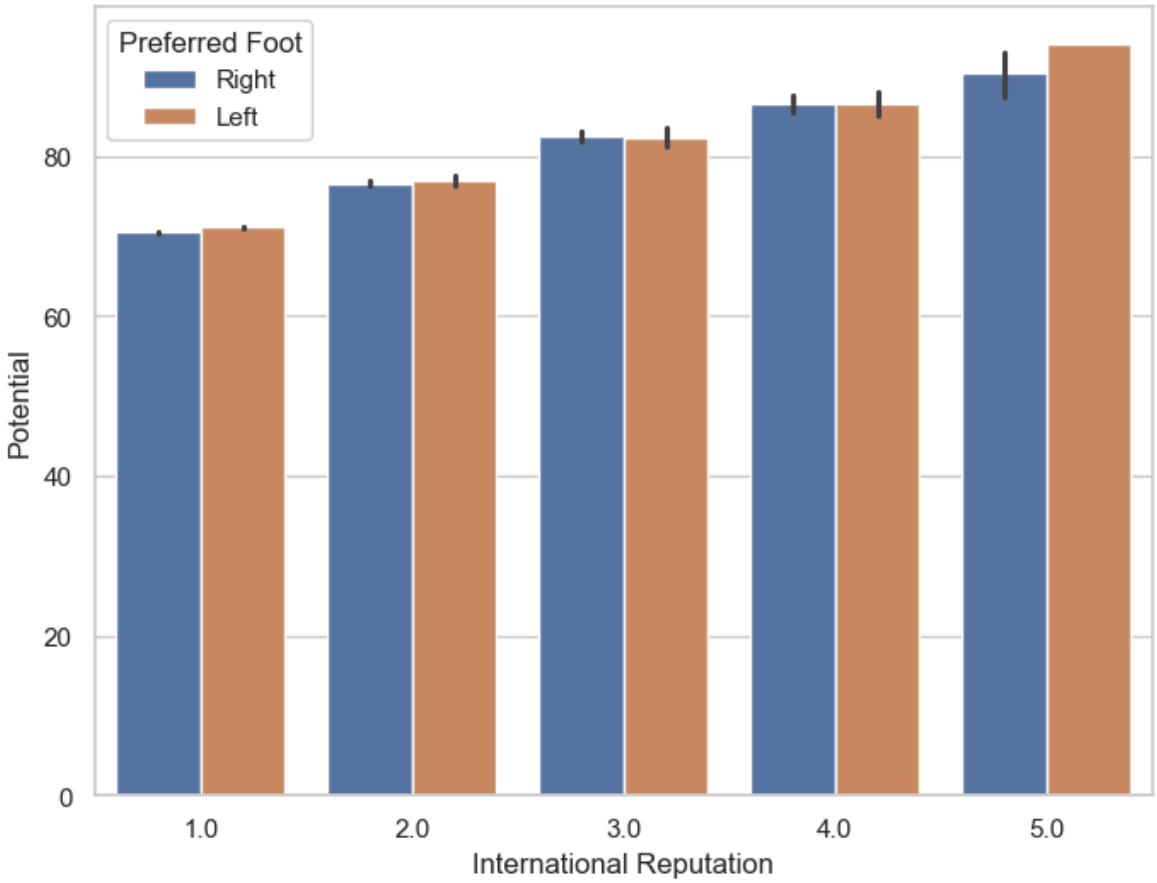


seaborn barplot() function

In [53]:
```
f, ax = plt.subplots(figsize=(8, 6))
sns.barplot(x="International Reputation", y="Potential", data=fifa, color='skybl
plt.show()
```



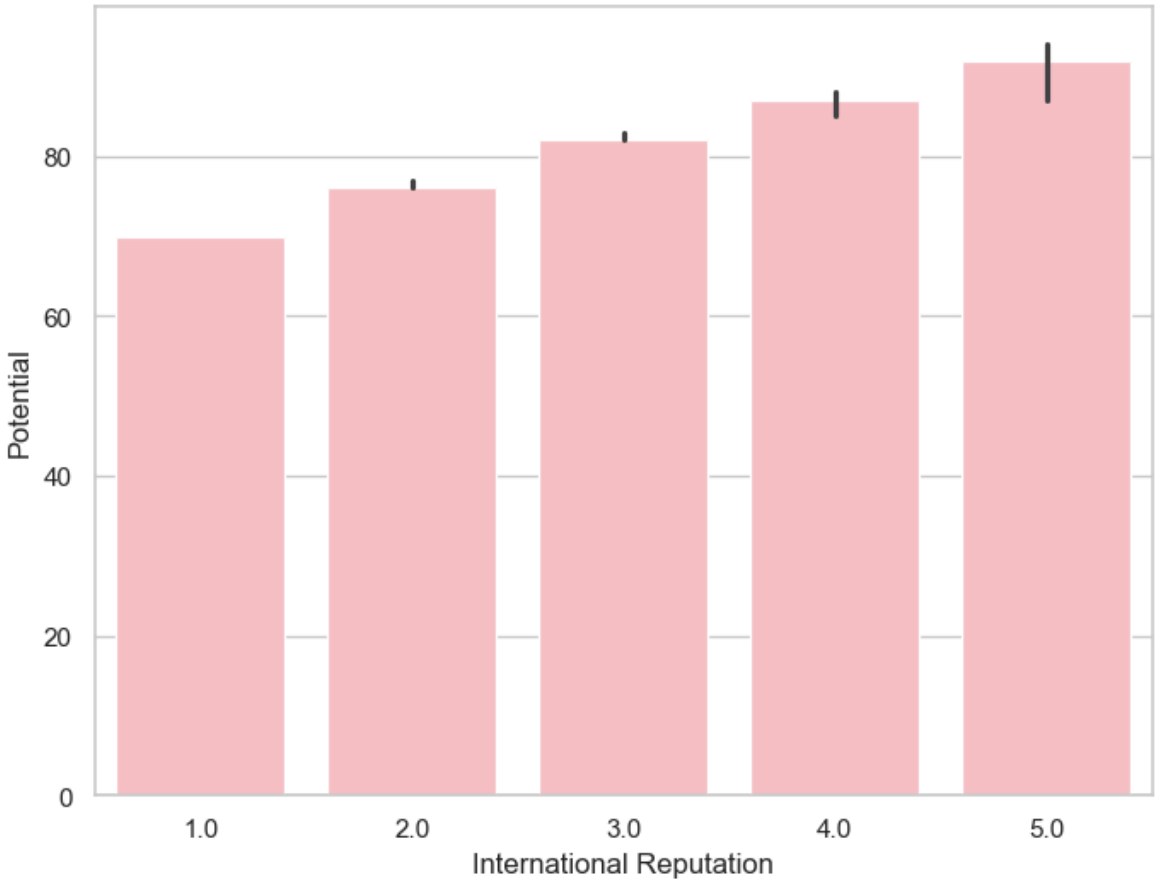
In [54]:
```
f, ax = plt.subplots(figsize=(8, 6))
sns.barplot(x="International Reputation", y="Potential", hue="Preferred Foot", d
plt.show()
```



WE CAN USE MEDIAN AS THE ESTIMATE OF CENTRAL TENDENCY

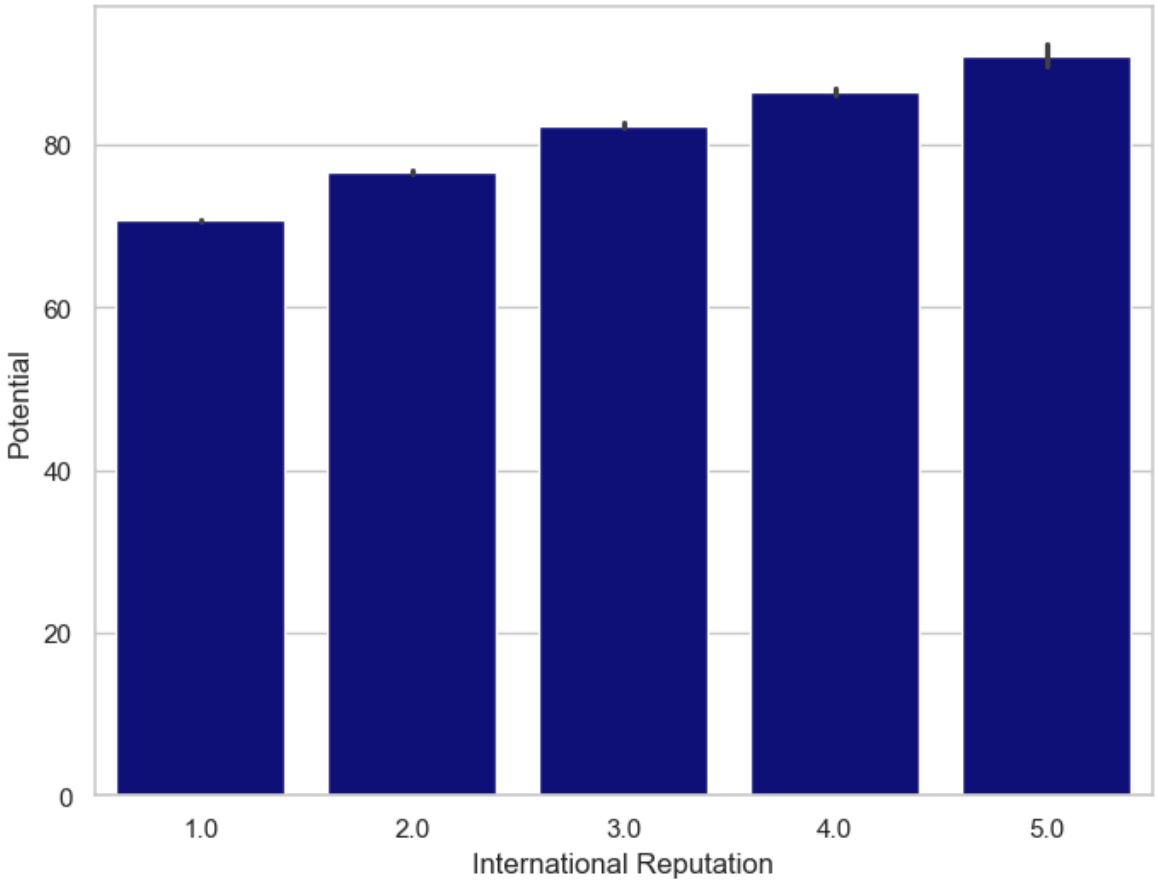
In [56]:
```
from numpy import median
import seaborn as sns
import matplotlib.pyplot as plt

f, ax = plt.subplots(figsize=(8, 6))
sns.barplot(x="International Reputation", y="Potential", data=fifa, estimator=me
plt.show()
```



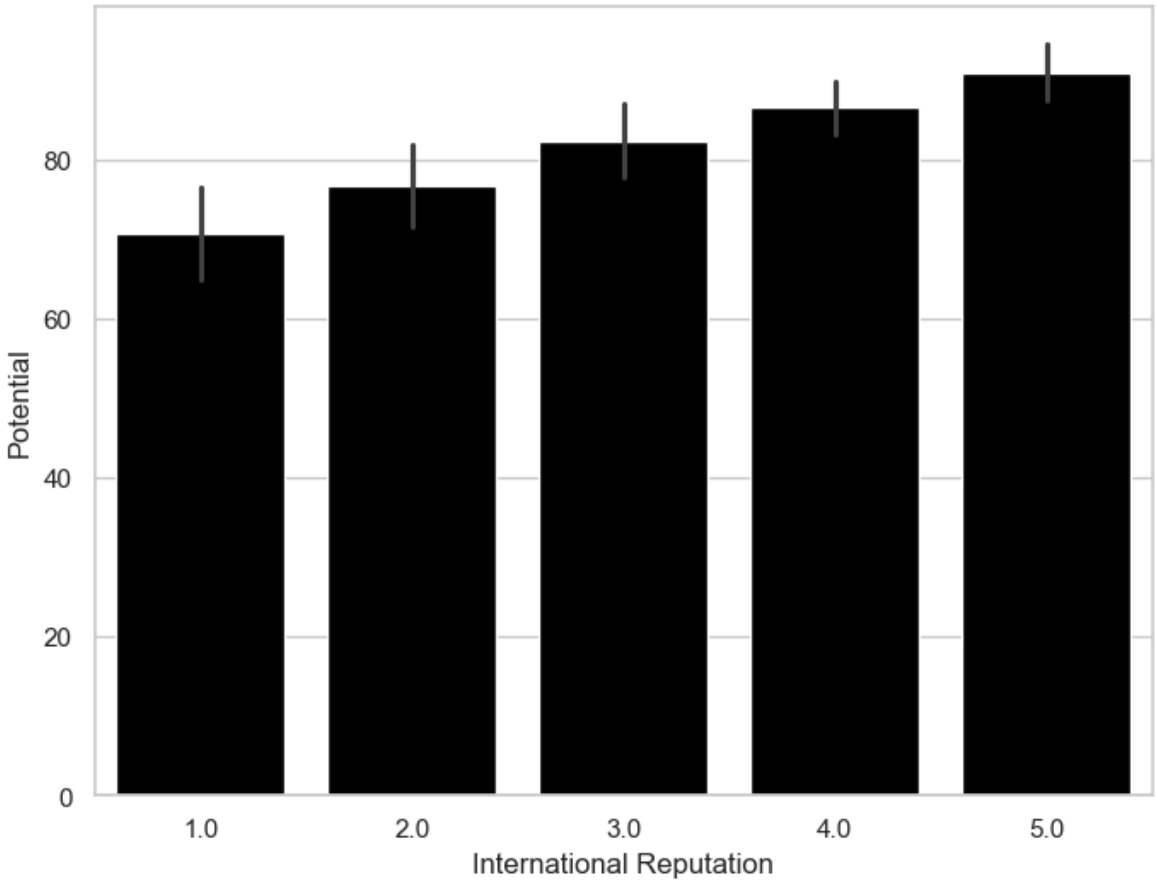
we can show the standard error of the mean with the error bars

In [58]:
```
f, ax = plt.subplots(figsize=(8, 6))
sns.barplot(x="International Reputation", y="Potential", data=fifa, ci=68,color
plt.show()
```



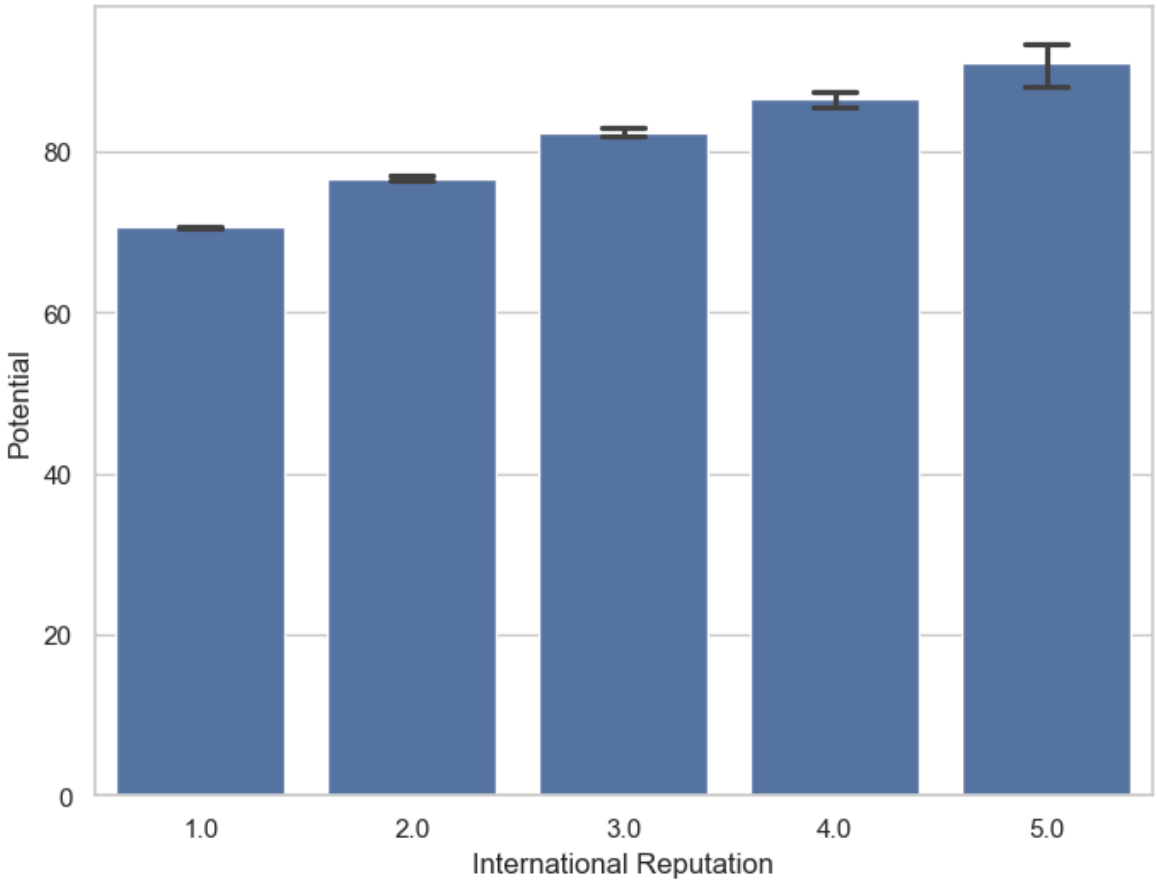
We can show standard deviation of observations instead of a confidence interval as follows-

In [60]:
```
f, ax = plt.subplots(figsize=(8, 6))
sns.barplot(x="International Reputation", y="Potential", data=fifa, ci="sd",colo
plt.show()
```



We can add “caps” to the error bars as follows-

In [62]:
```
f, ax = plt.subplots(figsize=(8, 6))
sns.barplot(x="International Reputation", y="Potential", data=fifa, capsize=0.2)
plt.show()
```

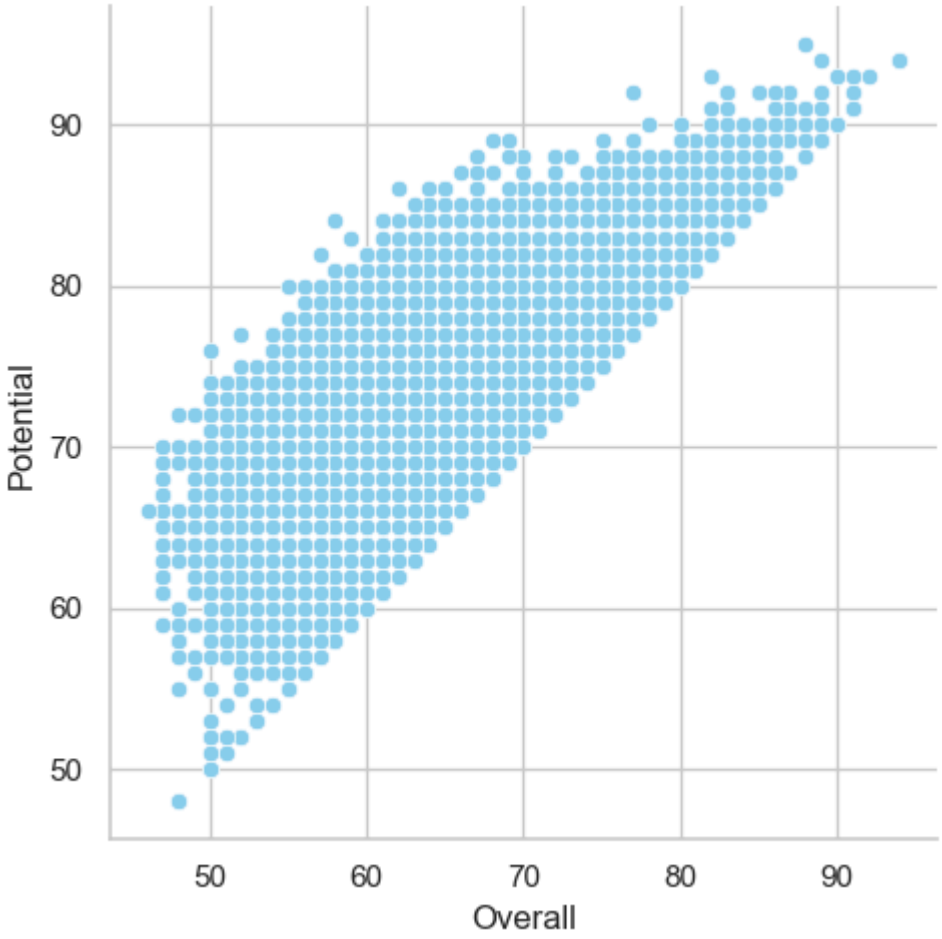


Visualizing statistical relationship with Seaborn `relplot()` function

SEABORN RELPLOT() FUNCTION

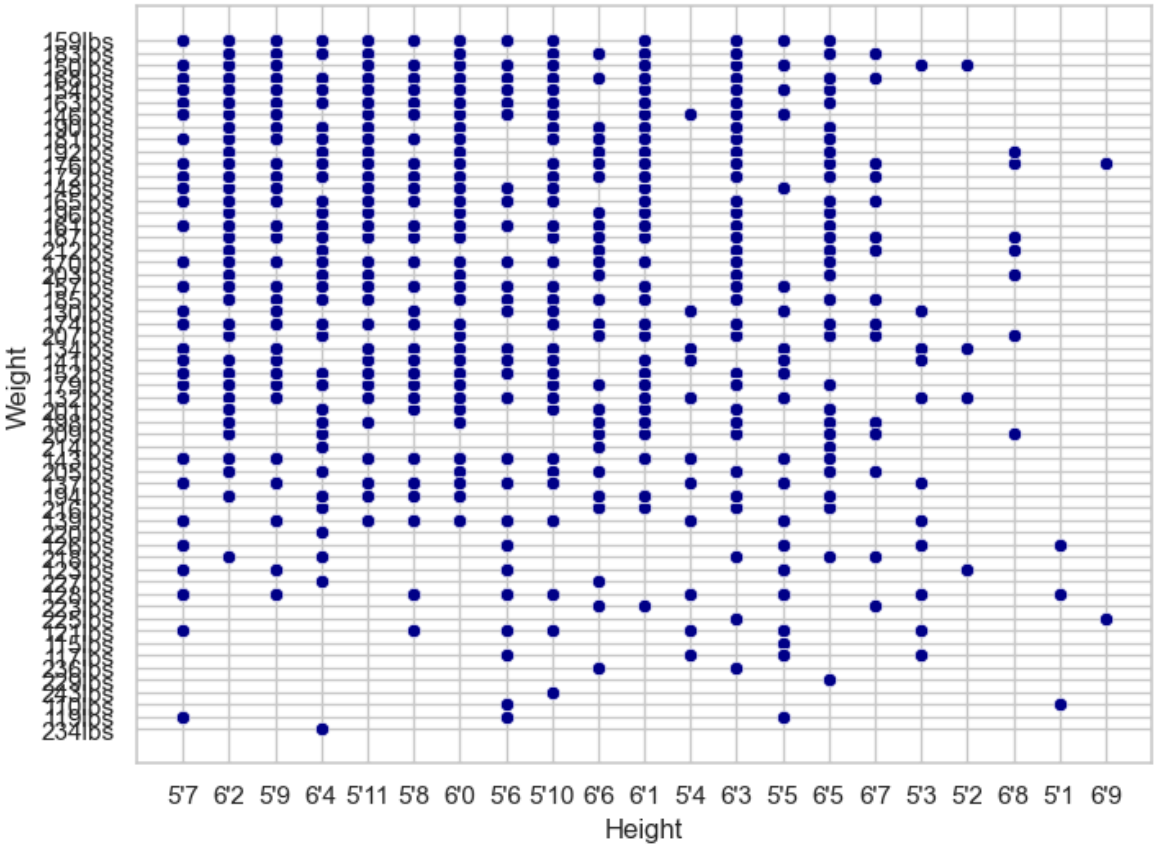
In [65]:
```
# height and weight with seaborn relplot function
```

In [66]:
```
g=sns.relplot(x="Overall",y="Potential",data=fifa,color='skyblue')
plt.show()
```



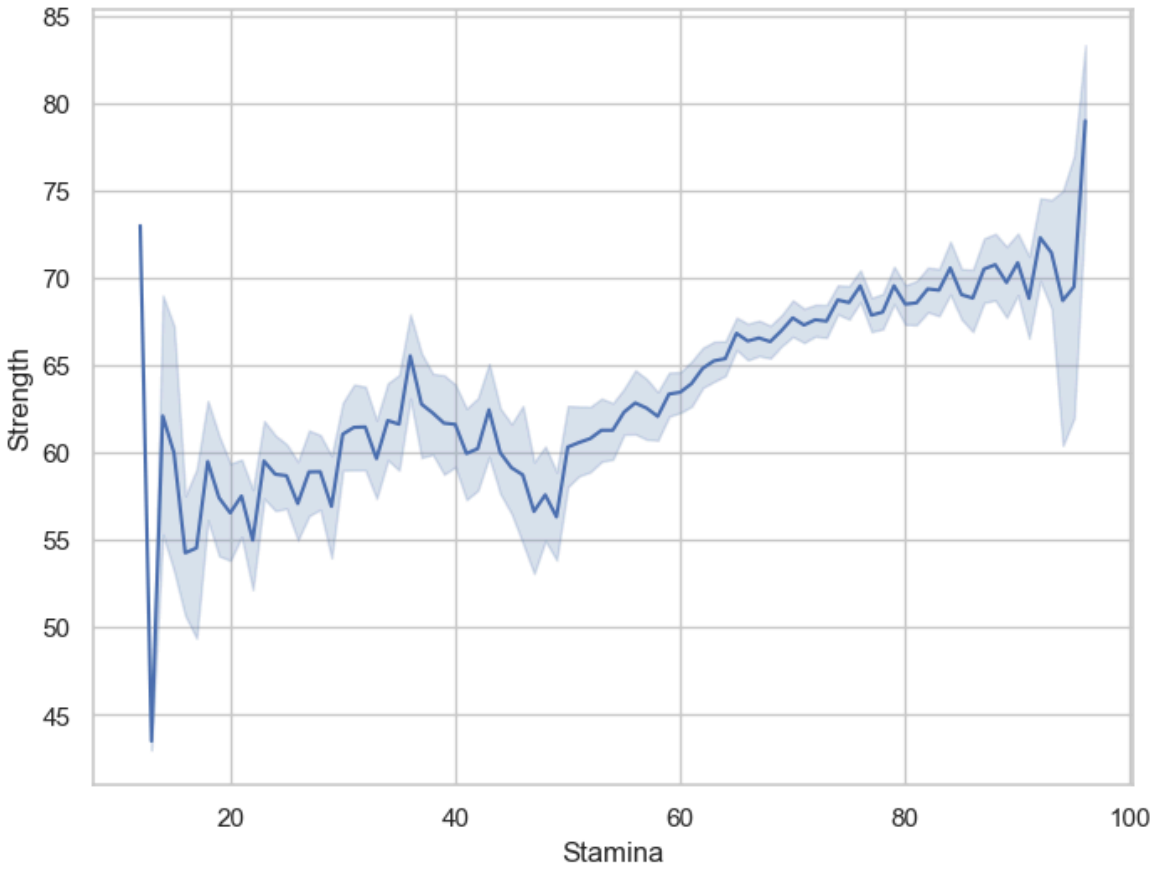
Seaborn `scatterplot()` function

In [68]:
```
f,ax = plt.subplots(figsize=(8,6))
sns.scatterplot(x="Height",y="Weight",data=fifa,color='darkblue')
plt.show()
```



SEABORN LINEPLOT() FUNCTION

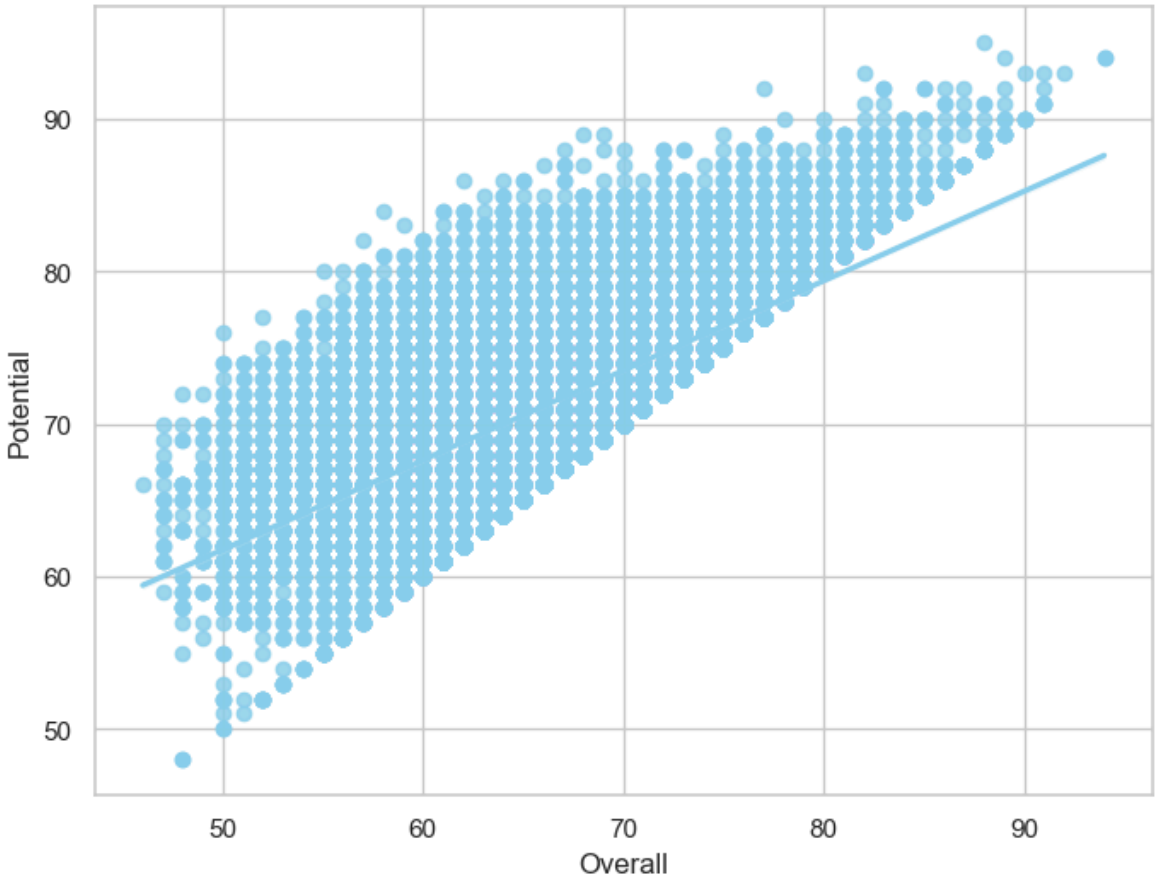
In [70]:
```
f, ax = plt.subplots(figsize=(8, 6))
ax = sns.lineplot(x="Stamina", y="Strength", data=fifa)
plt.show()
```



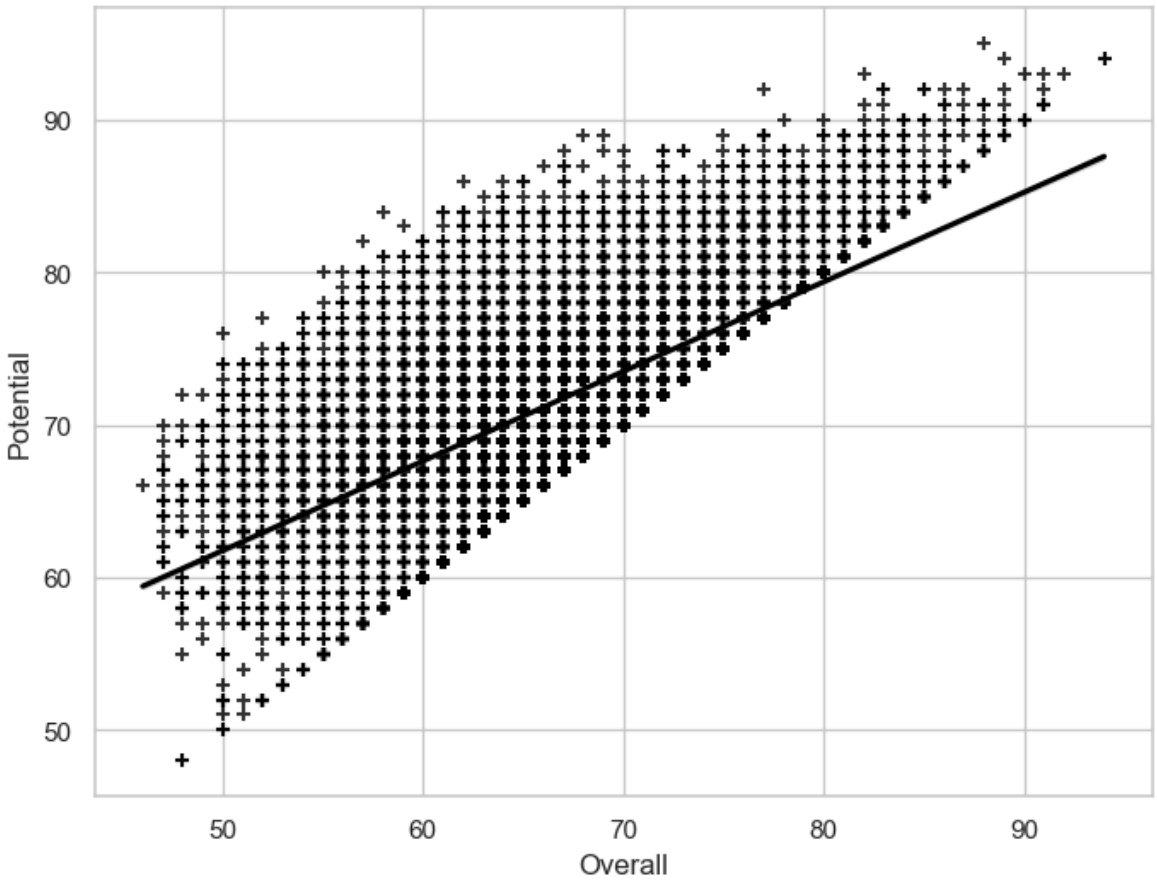
Visualize linear relationship with Seaborn `regplot()` function

SEABORN REGPLOT() FUNCTION

In [73]:
```
f,ax = plt.subplots(figsize=(8,6))
ax = sns.regplot(x="Overall",y="Potential",data=fifa,color='skyblue')
plt.show()
```

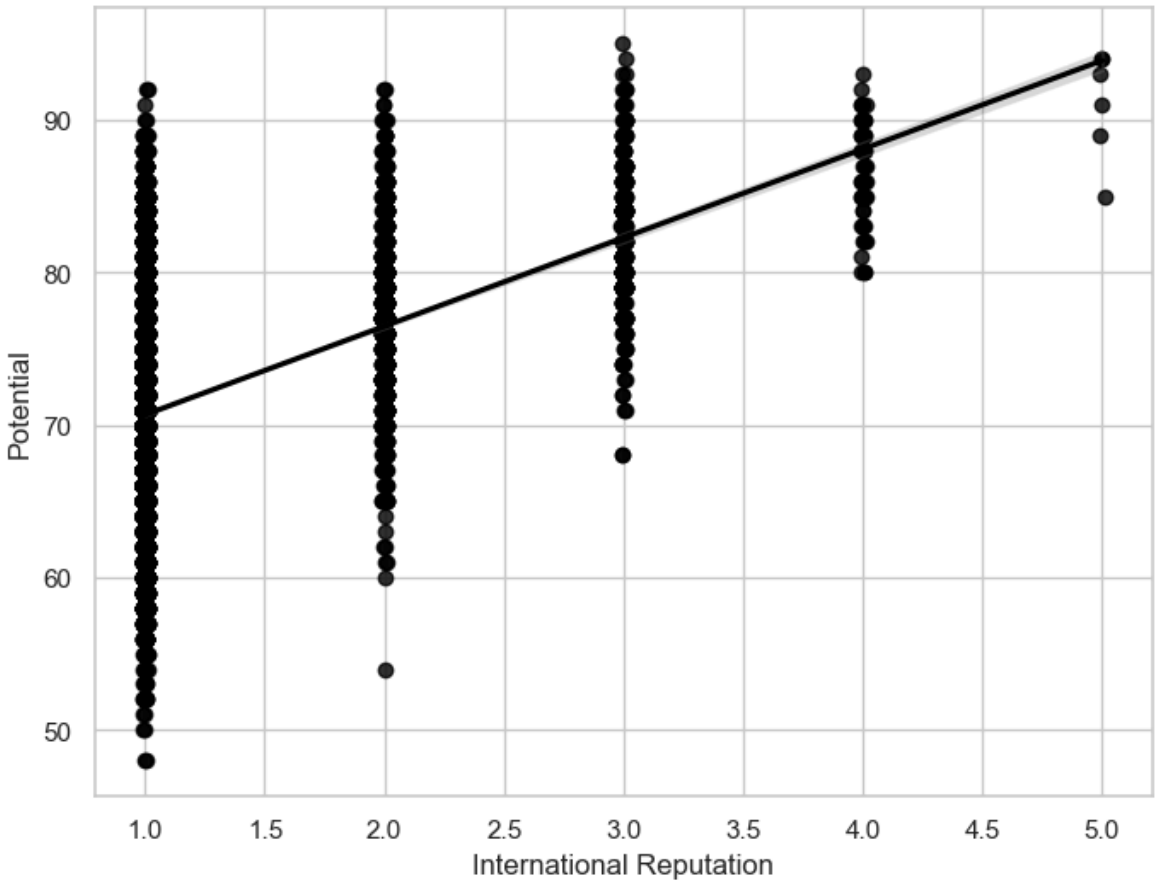


In [74]:
```
f,ax = plt.subplots(figsize=(8,6))
ax = sns.regplot(x="Overall",y="Potential",data=fifa, color= "black", marker="+"
plt.show()
```



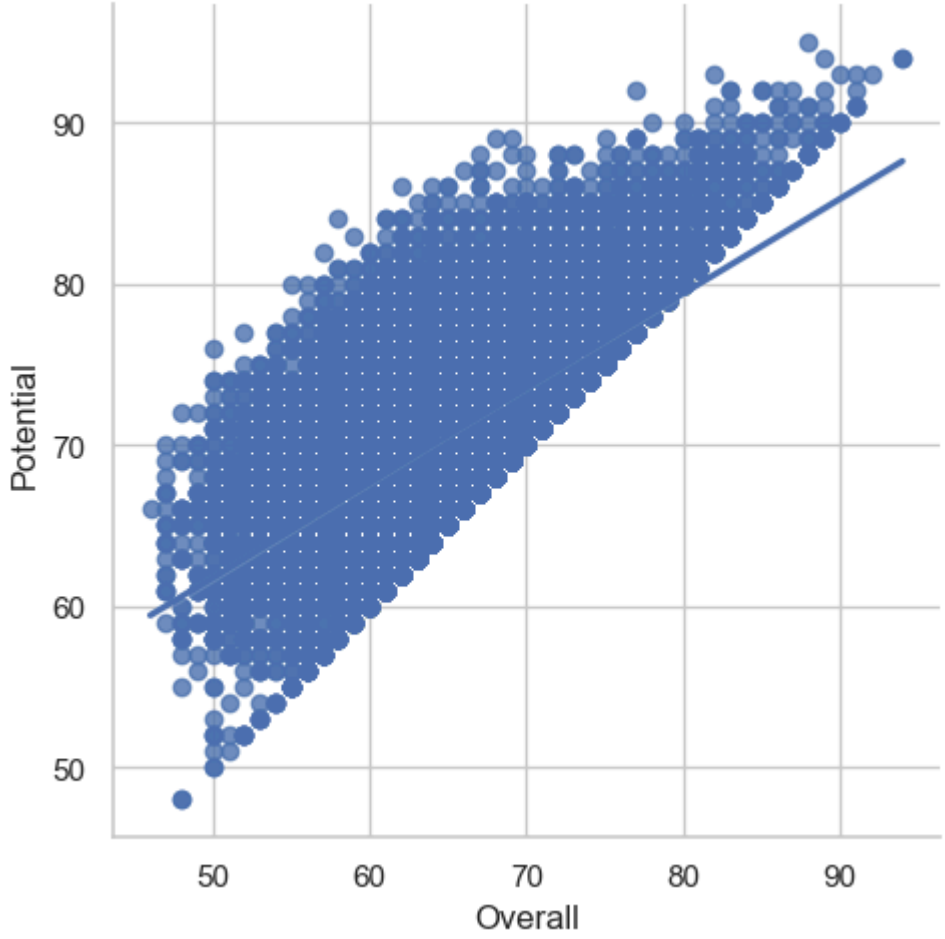
We can plot with a discrete variable and add some jitter :

In [76]:
```
f, ax = plt.subplots(figsize=(8, 6))
sns.regplot(x="International Reputation", y="Potential", data=fifa, x_jitter=.01
plt.show()
```

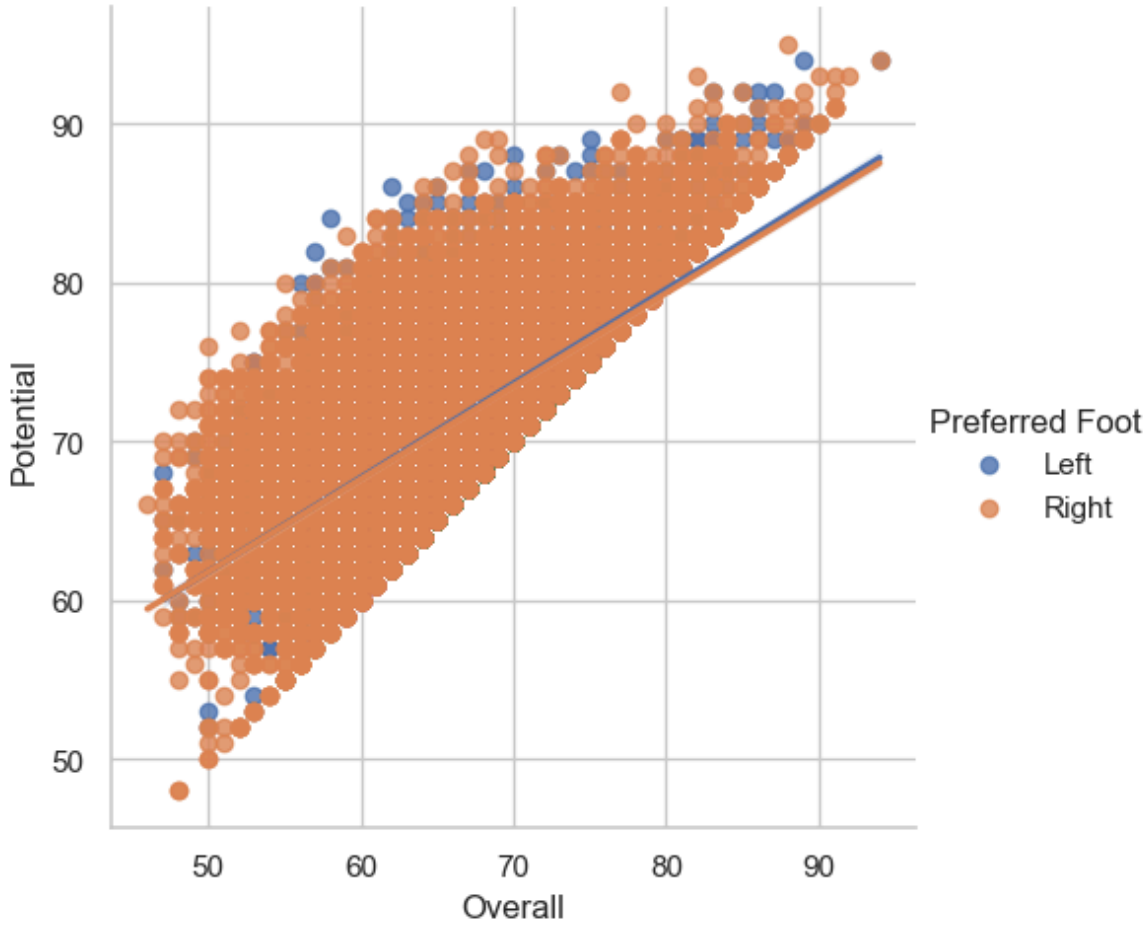


SEABORN LMPLOT() FUNCTION

In [78]:
```
g = sns.lmplot(x="Overall", y = "Potential",data= fifa)
plt.show()
```

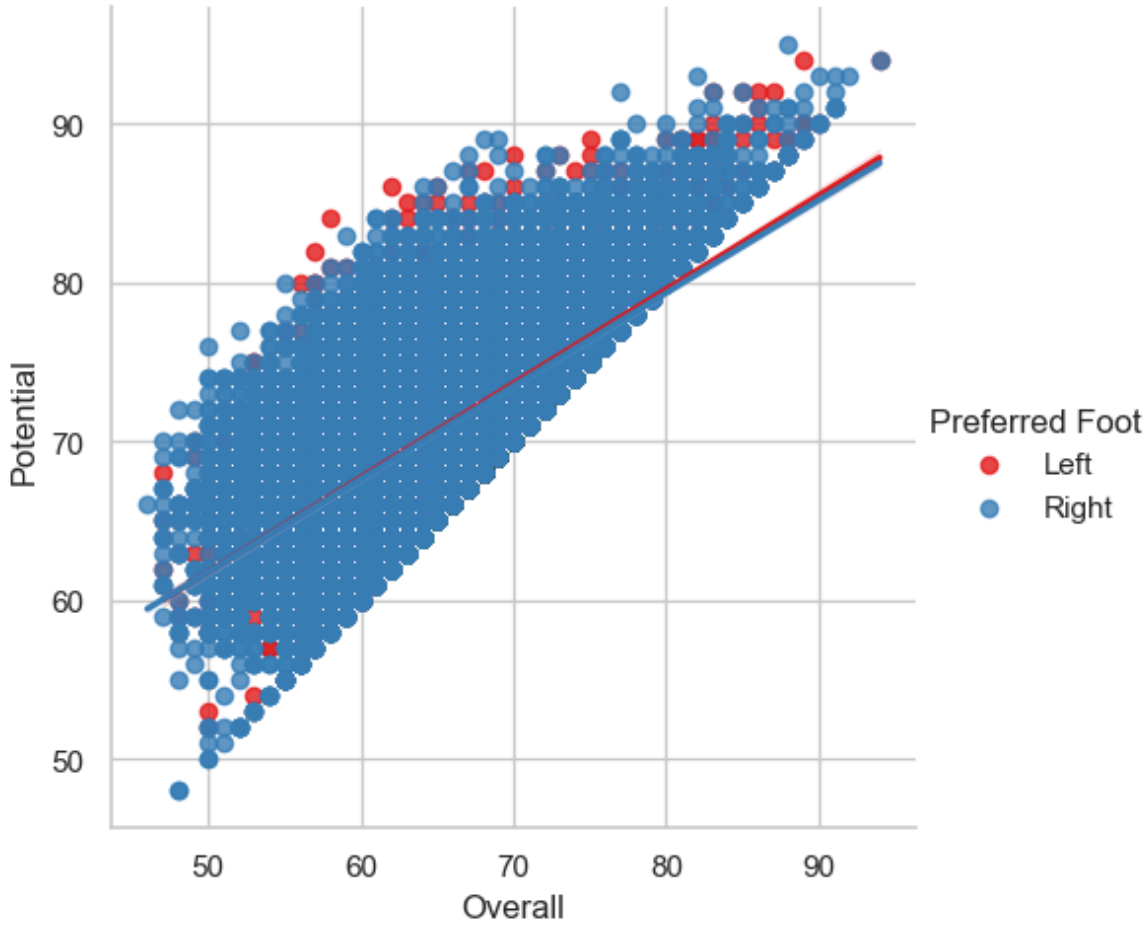


In [79]:
```
g= sns.lmplot(x="Overall", y="Potential", hue="Preferred Foot", data=fifa)
plt.show()
```



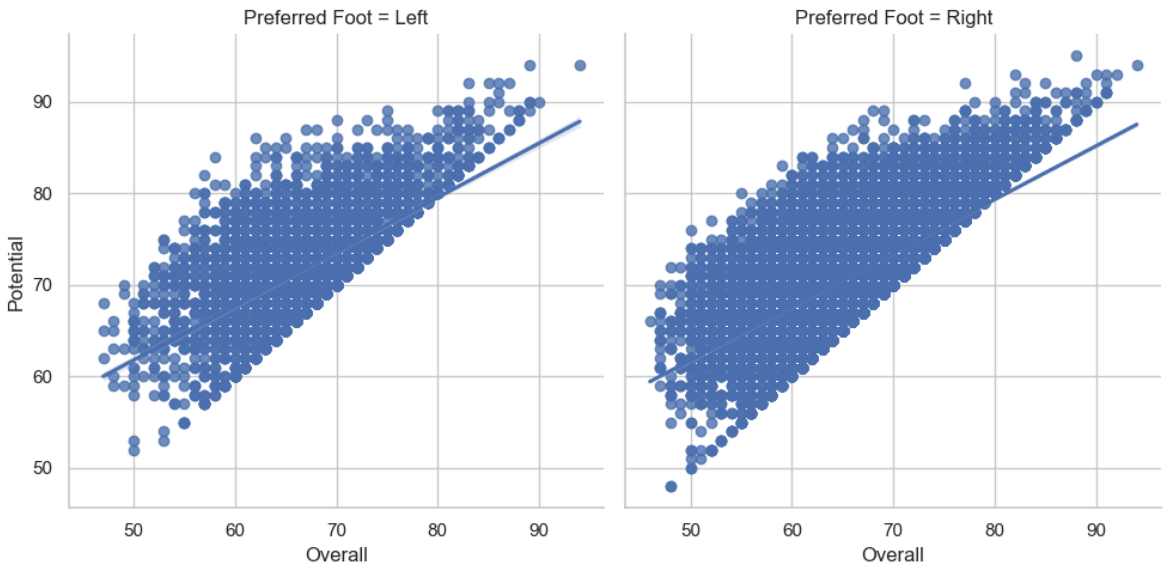
WE CAN USE A DIFFERENT COLOR PALETTE AS :

In [81]:
```
g= sns.lmplot(x="Overall", y="Potential", hue="Preferred Foot", data=fifa, palet
plt.show()
```



WE CAN PLOT THE LEVELS OF THE THIRD VARIABLES ACROSS DIFFERENT VARIABLES COLUMNS :

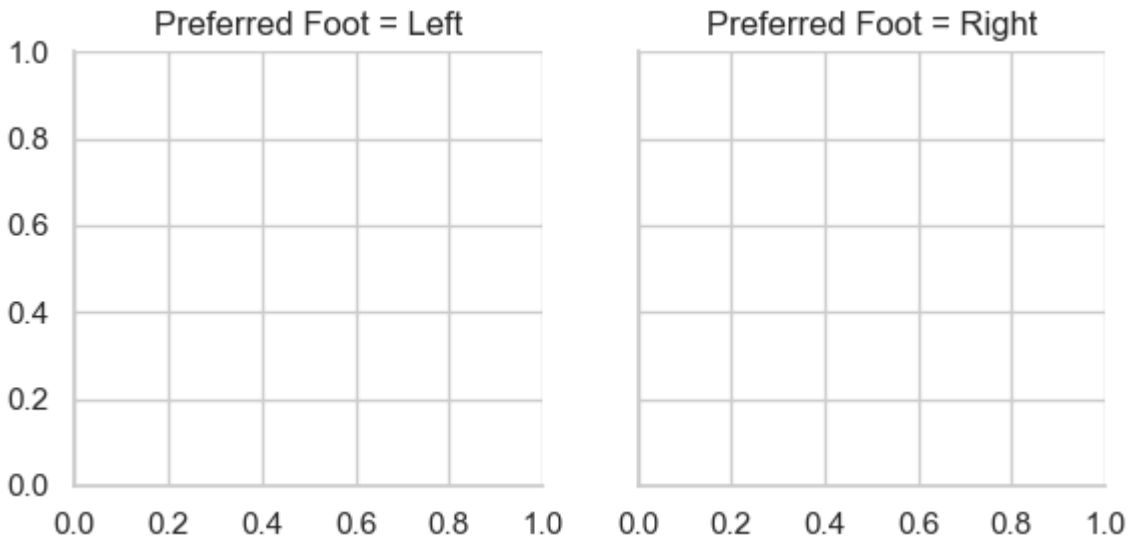
In [83]:
```
g= sns.lmplot(x="Overall", y="Potential", col="Preferred Foot", data=fifa)
plt.show()
```



MULTIPLOT GRIDS

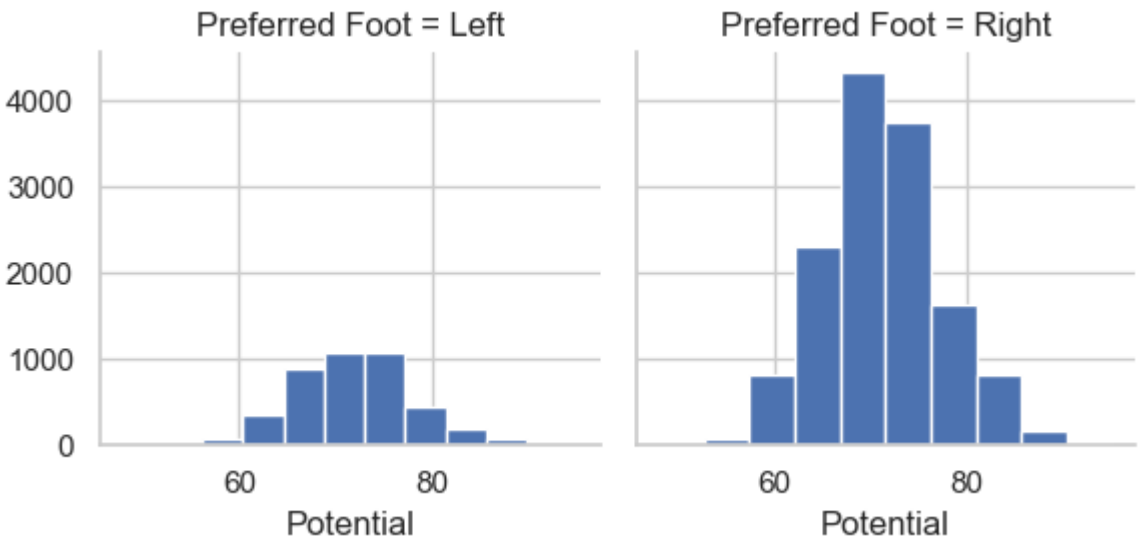
SEABORN FACETGRID() FUNCTION

In [86]:
```
g = sns.FacetGrid(fifa, col="Preferred Foot")
plt.show()
```

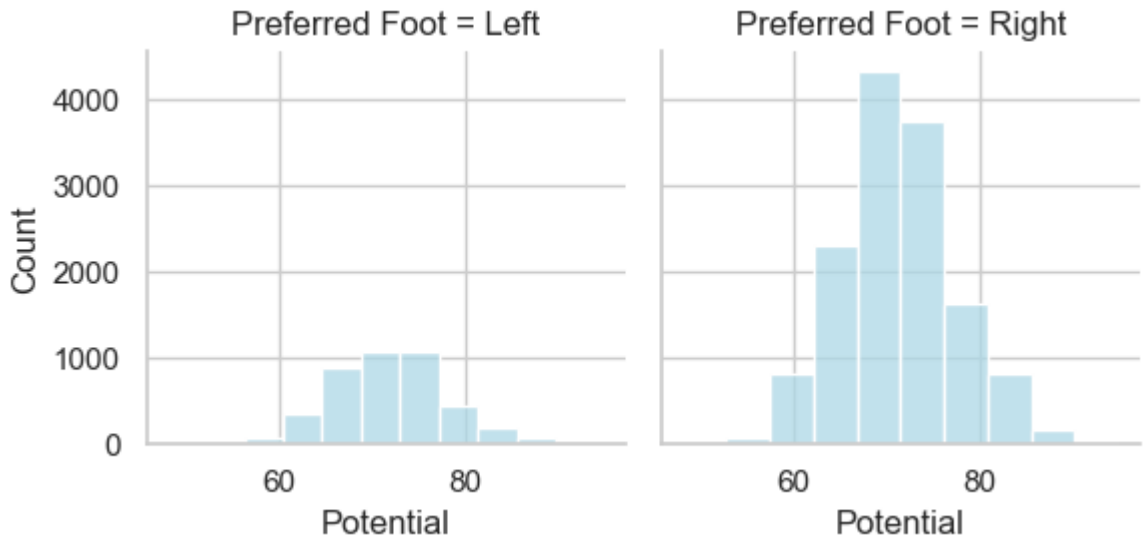


WE CAN DRAW A UNIVARIATE PLOT OF POTENTIAL VARIABLE ON EACH FACET :

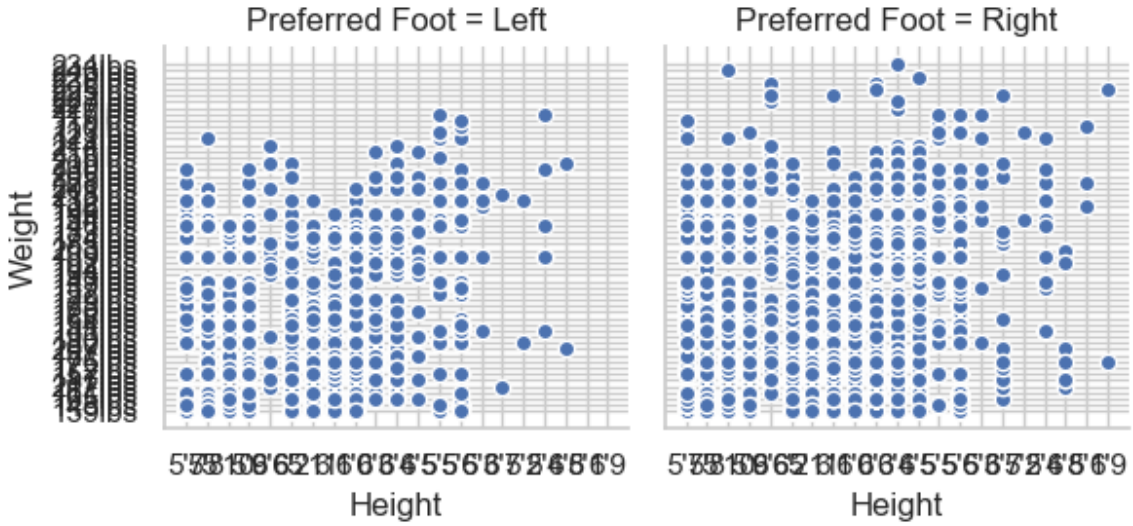
In [88]:
```
g = sns.FacetGrid(fifa, col="Preferred Foot")
g = g.map(plt.hist, "Potential")
plt.show()
```



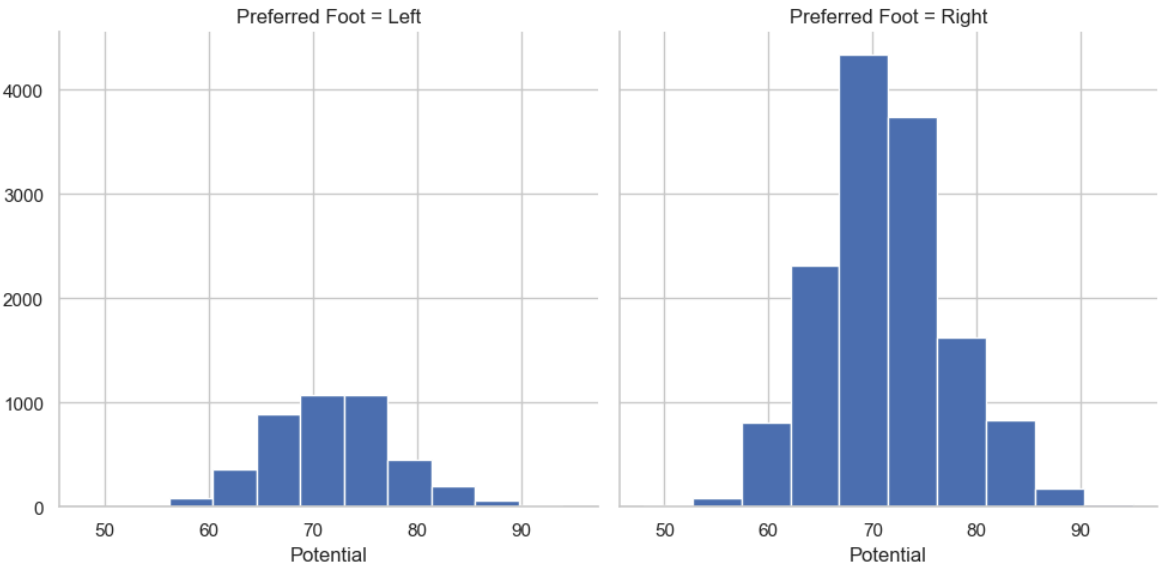
In [89]:
```
g = sns.FacetGrid(fifa, col="Preferred Foot")
g.map(sns.histplot, "Potential", bins=10, color="lightblue")
plt.show()
```



In [90]:
```
g = sns.FacetGrid(fifa, col="Preferred Foot")
g = (g.map(plt.scatter, "Height", "Weight", edgecolor="w").add_legend())
plt.show()
```



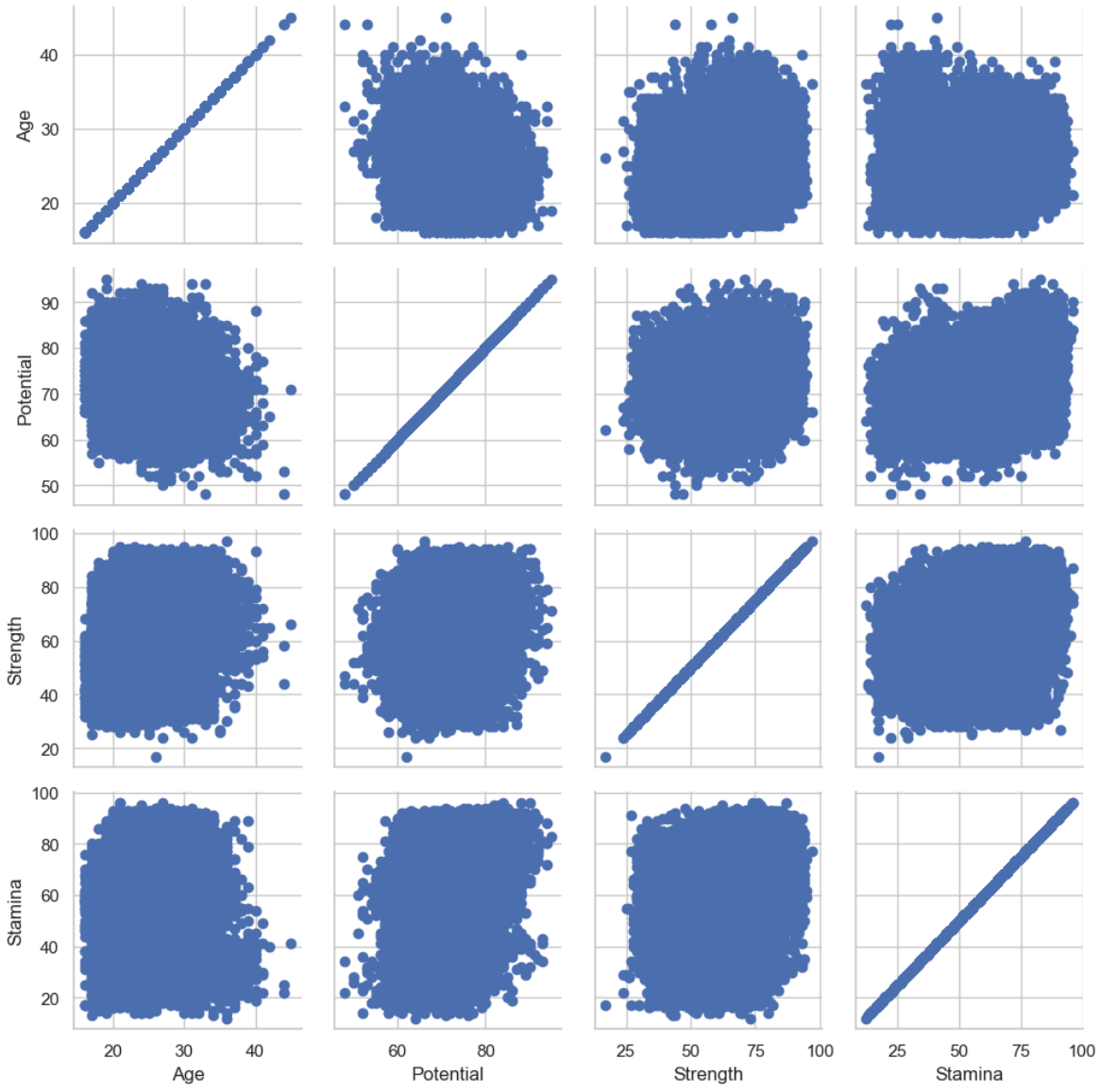
In [91]:
```
g = sns.FacetGrid(fifa, col="Preferred Foot", height=5, aspect=1)
g = g.map(plt.hist, "Potential")
plt.show()
```



SEABORN PAIRGRID() FUNCTION

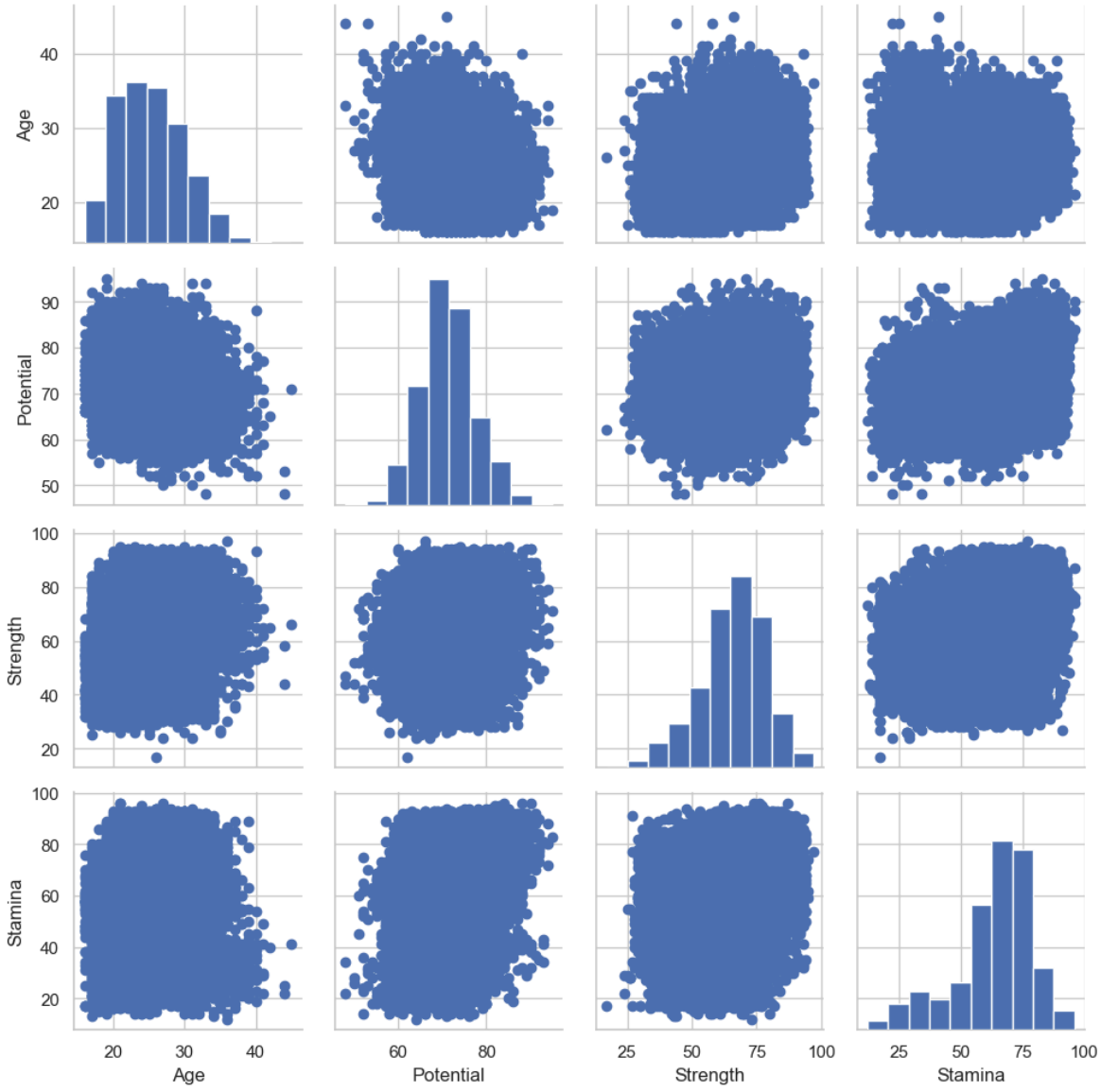
In [93]:
```
fifa_new= fifa[['Age', 'Potential', 'Strength', 'Stamina', 'Preferred Foot']]
plt.show()
```

In [94]:
```
g = sns.PairGrid(fifa_new)
g = g.map(plt.scatter)
plt.show()
```



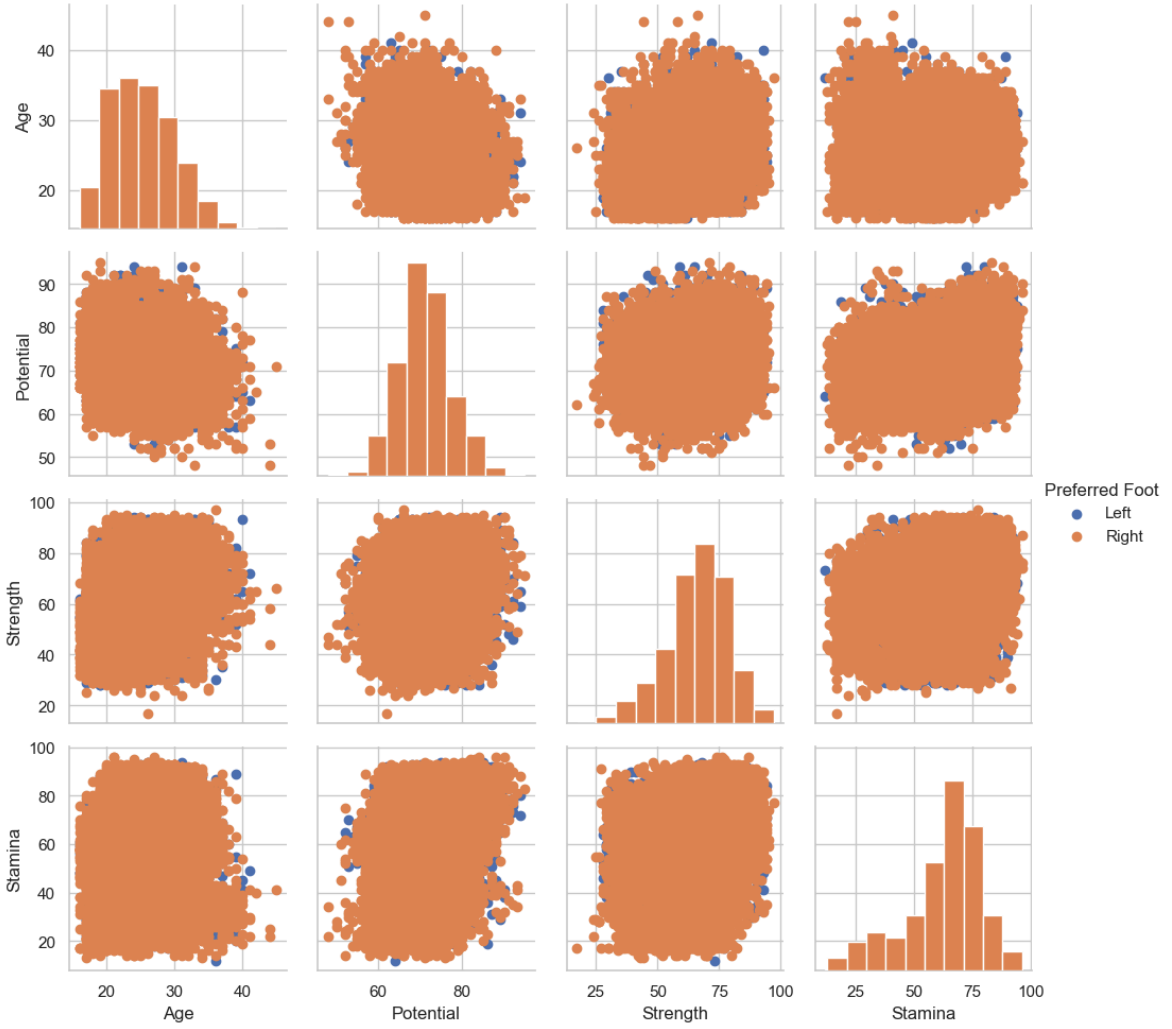
WE CAN SHOW A UNIVARIATE DISTRIBUTION ON THE DIAGONAL AS :

In [96]:
```
g = sns.PairGrid(fifa_new)
g = g.map_diag(plt.hist)
g = g.map_offdiag(plt.scatter)
plt.show()
```



WE CAN COLOR THE POINTS USING THE CATEGORICAL VARIABLE PREFERRED FOOT AS:

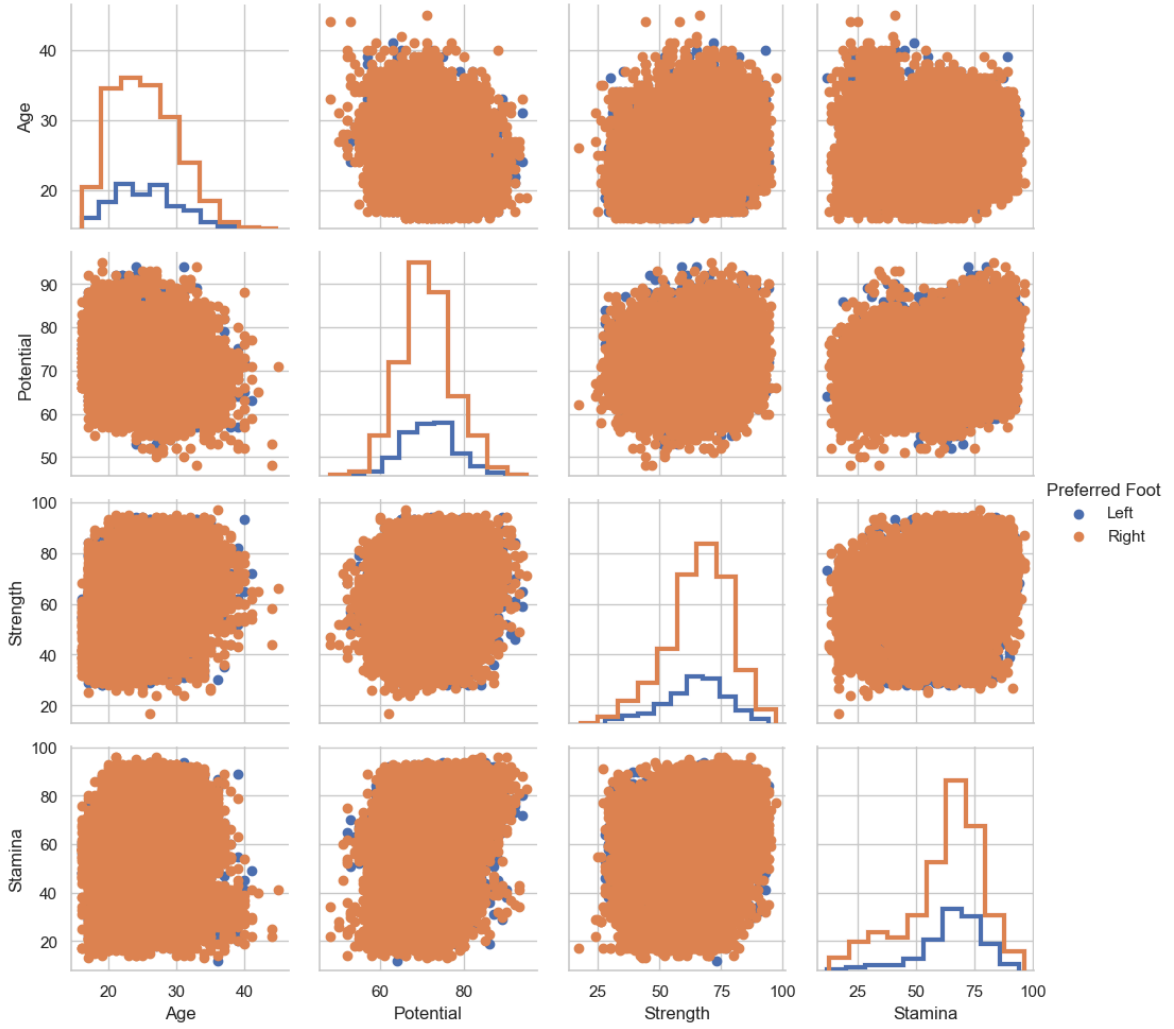
In [98]:
```
g = sns.PairGrid(fifa_new, hue="Preferred Foot")
g = g.map_diag(plt.hist)
g = g.map_offdiag(plt.scatter)
g = g.add_legend()
plt.show()
```



WE CAN USE A DIFFERENT STYLE TO SHOW MULTIPLE HISTOGRAMS

In [100…

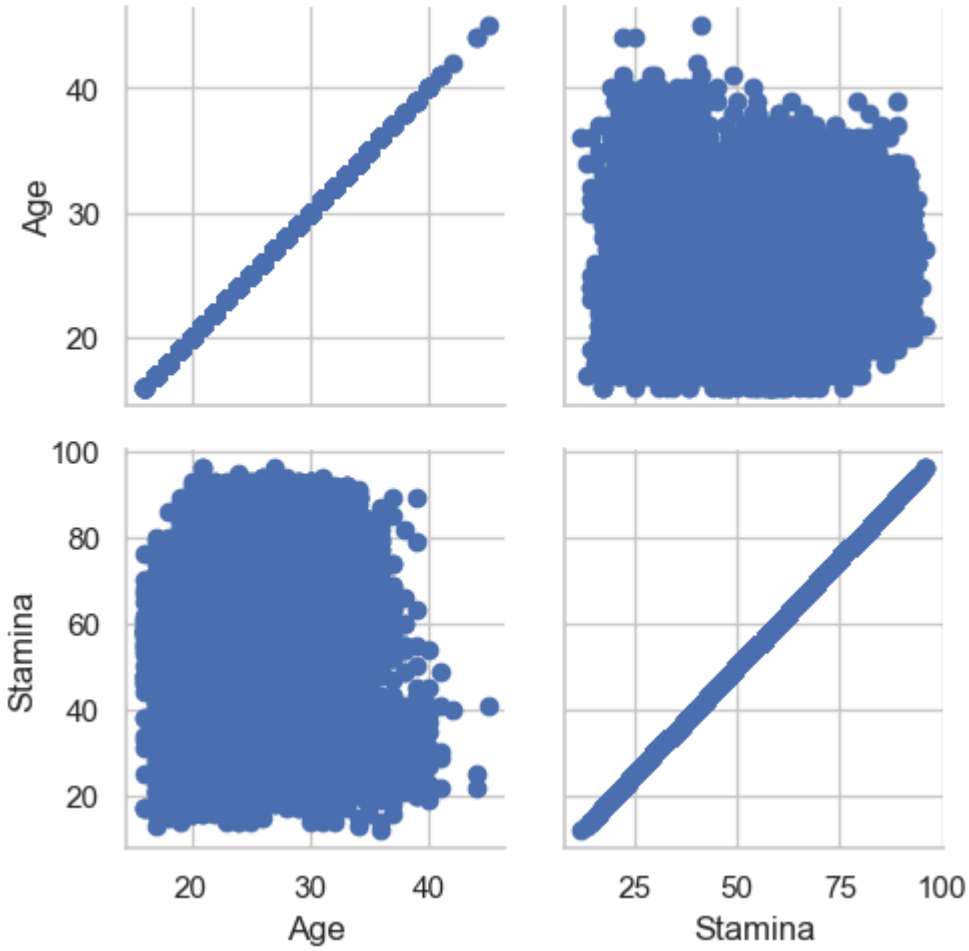
```
g = sns.PairGrid(fifa_new, hue="Preferred Foot")
g = g.map_diag(plt.hist, histtype="step", linewidth=3)
g = g.map_offdiag(plt.scatter)
g = g.add_legend()
plt.show()
```



WE CAN PLOT A SUBSET OF VARIABLES AS

In [102…

```
g = sns.PairGrid(fifa_new, vars=['Age', 'Stamina'])
g = g.map(plt.scatter)
plt.show()
```



We can use Different functions on the upper and lower triangles as follows :

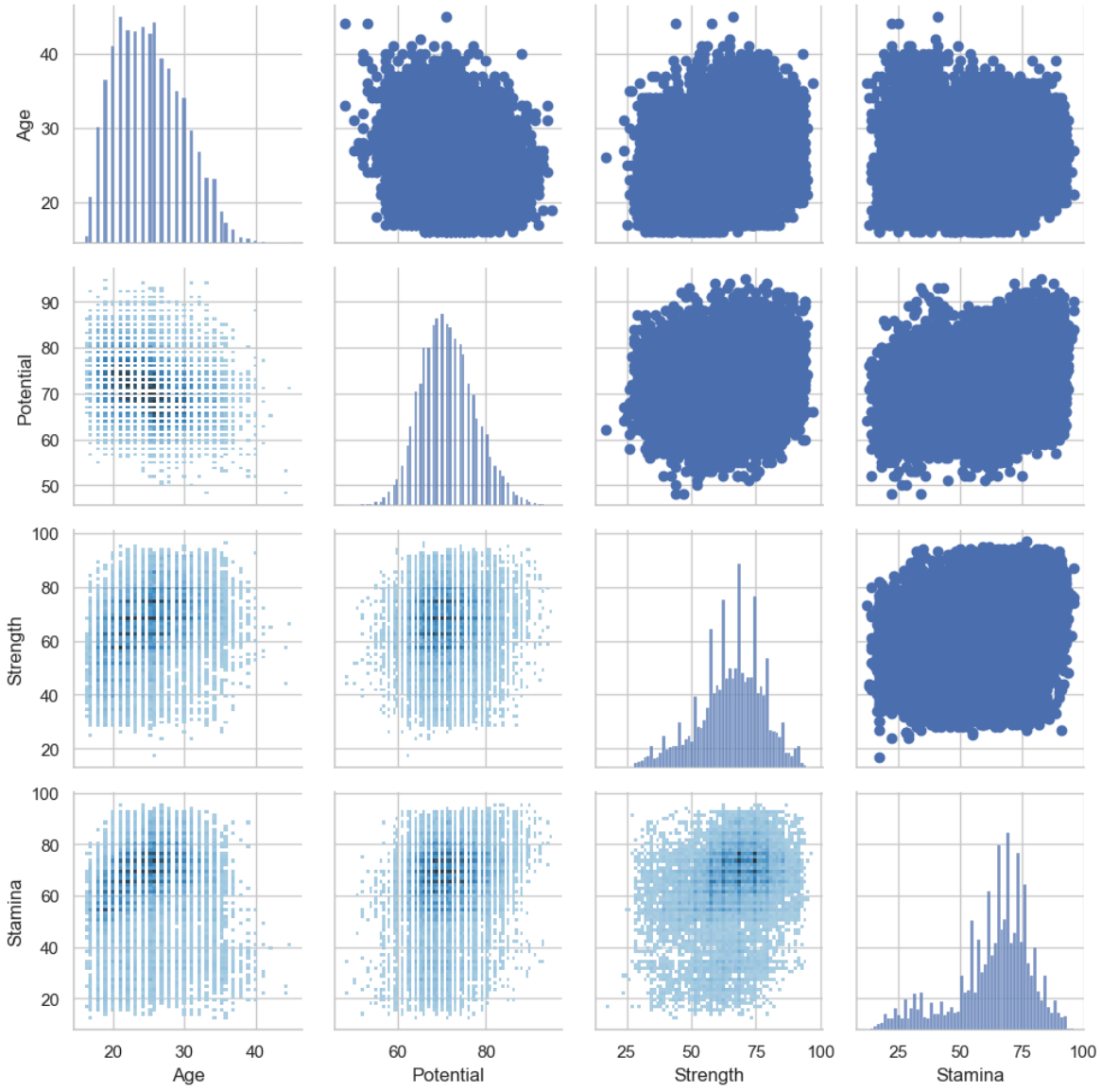
In [104…

```
# Check if your dataset is loaded
print("DataFrame type:", type(fifa_new))
print("DataFrame shape:", fifa_new.shape)
print("Column types:")
print(fifa_new.dtypes)
```

```
DataFrame type: <class 'pandas.core.frame.DataFrame'>
DataFrame shape: (18207, 5)
Column types:
Age                 int64
Potential           int64
Strength          float64
Stamina           float64
Preferred Foot     object
dtype: object
```

In [105…

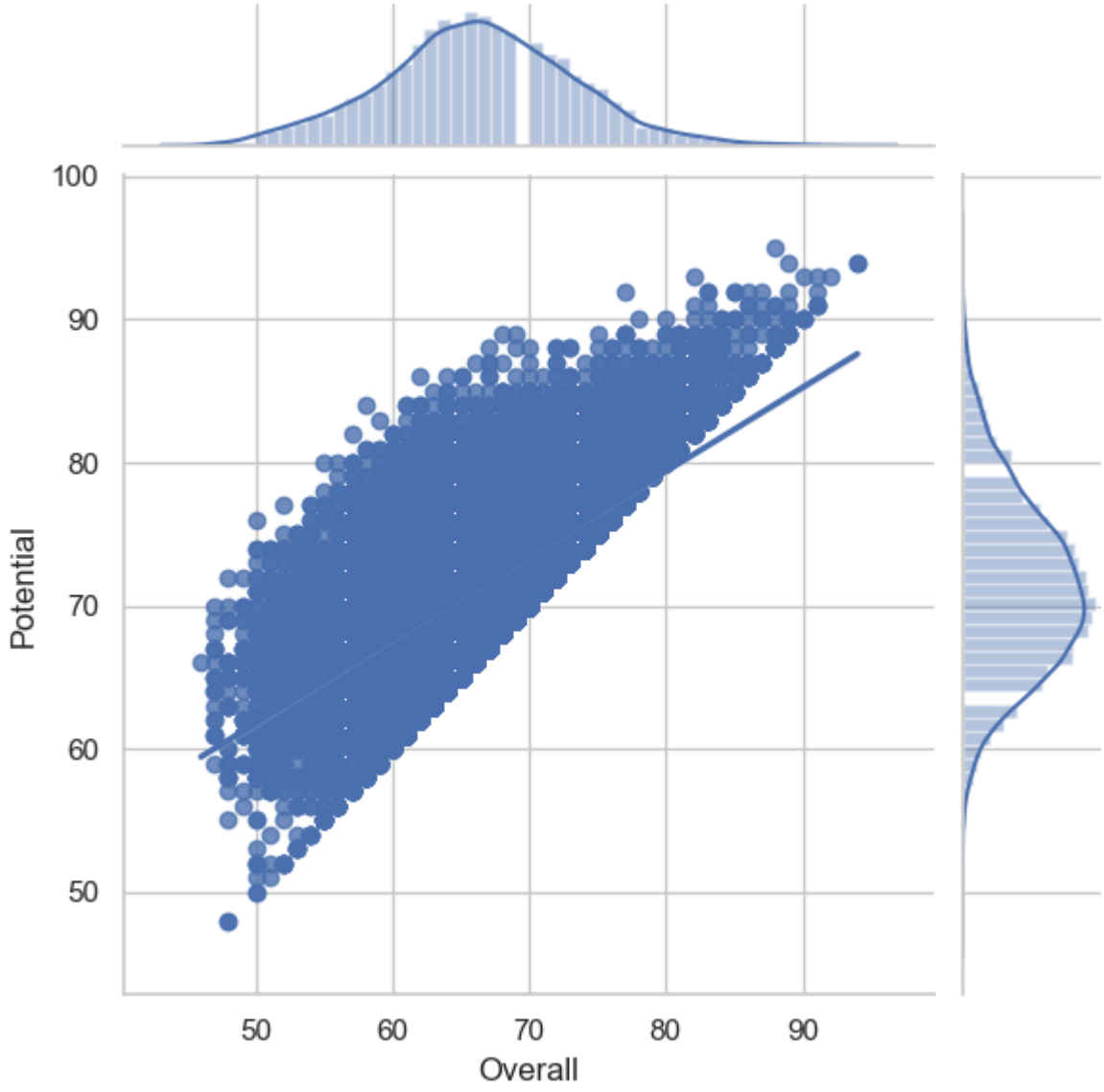
```
g = sns.PairGrid(fifa_new)
g = g.map_upper(plt.scatter)
g = g.map_lower(sns.histplot, cmap="Blues_d")
g = g.map_diag(sns.histplot, lw=3, legend=False)
plt.show()
```



In [ ]:
```
g = sns.PairGrid(fifa_new)
g = g.map_upper(plt.scatter)
g = g.map_lower(sns.kdeplot, cmap="Blues_d")
g = g.map_diag(sns.kdeplot, lw=3, legend=False)
plt.show()
```

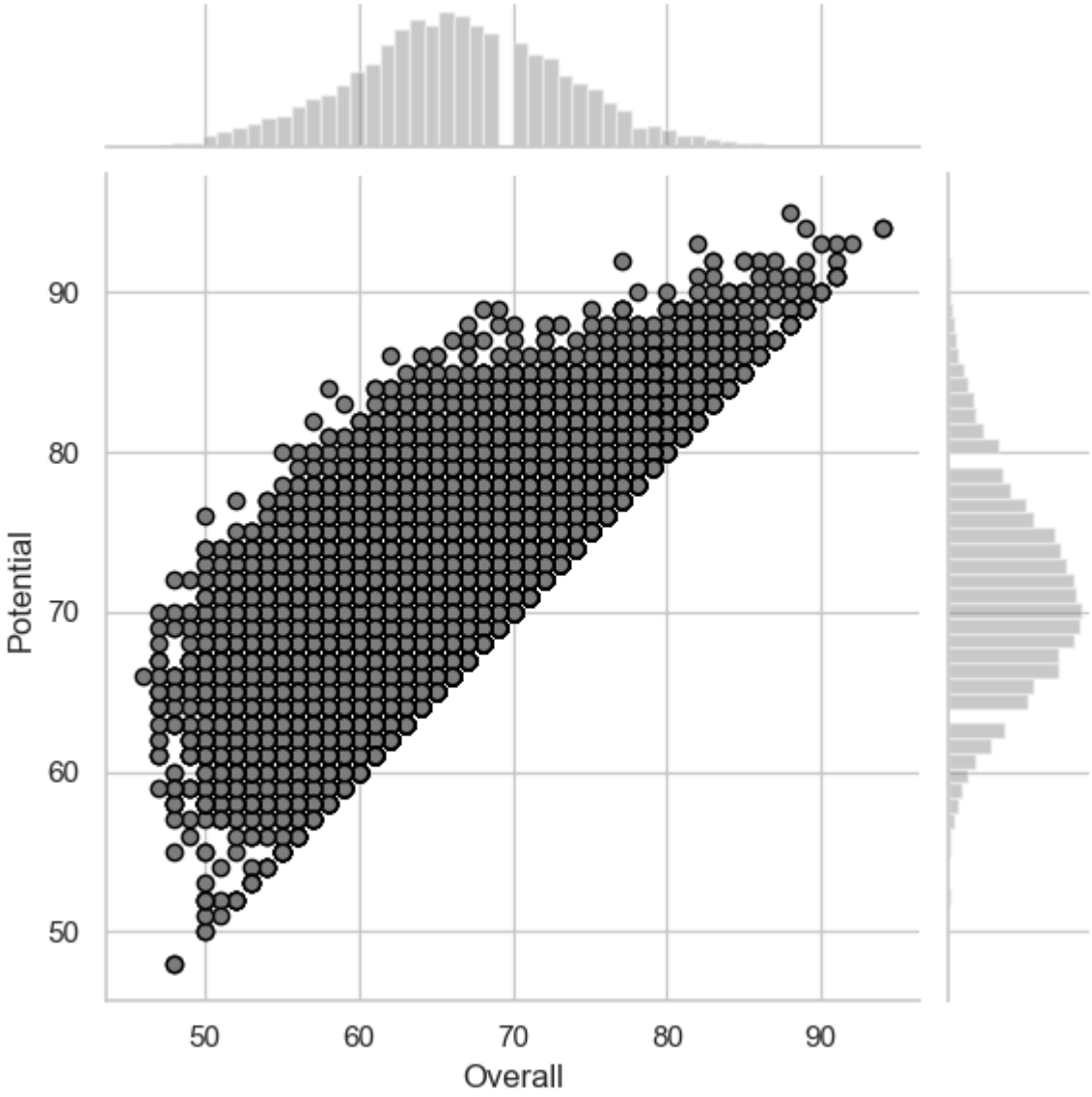
SEABORN JOINTGRID() FUNCTION

In [108…
```
g = sns.JointGrid(x="Overall", y="Potential", data=fifa)
g = g.plot(sns.regplot, sns.distplot)
plt.show()
```



In [109…
```
import matplotlib as pyplot
```

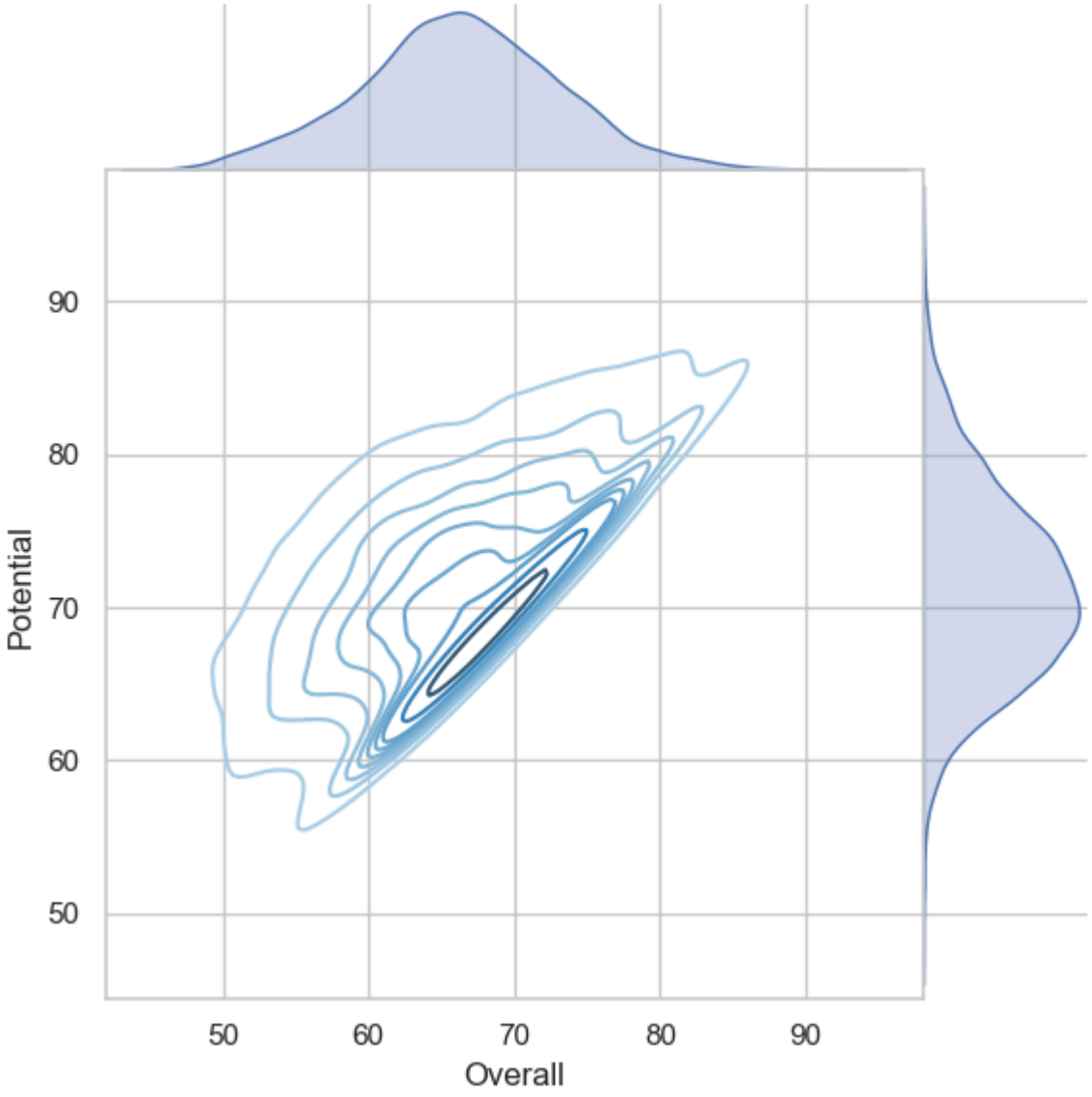
In [110…
```
g = sns.JointGrid(x="Overall", y="Potential", data=fifa)
g = g.plot_joint(plt.scatter, color=".5", edgecolor="Black")
g = g.plot_marginals(sns.distplot, kde=False, color=".5")
plt.show()
```



We can remove the space between the joint and marginal axes as :

In [112…

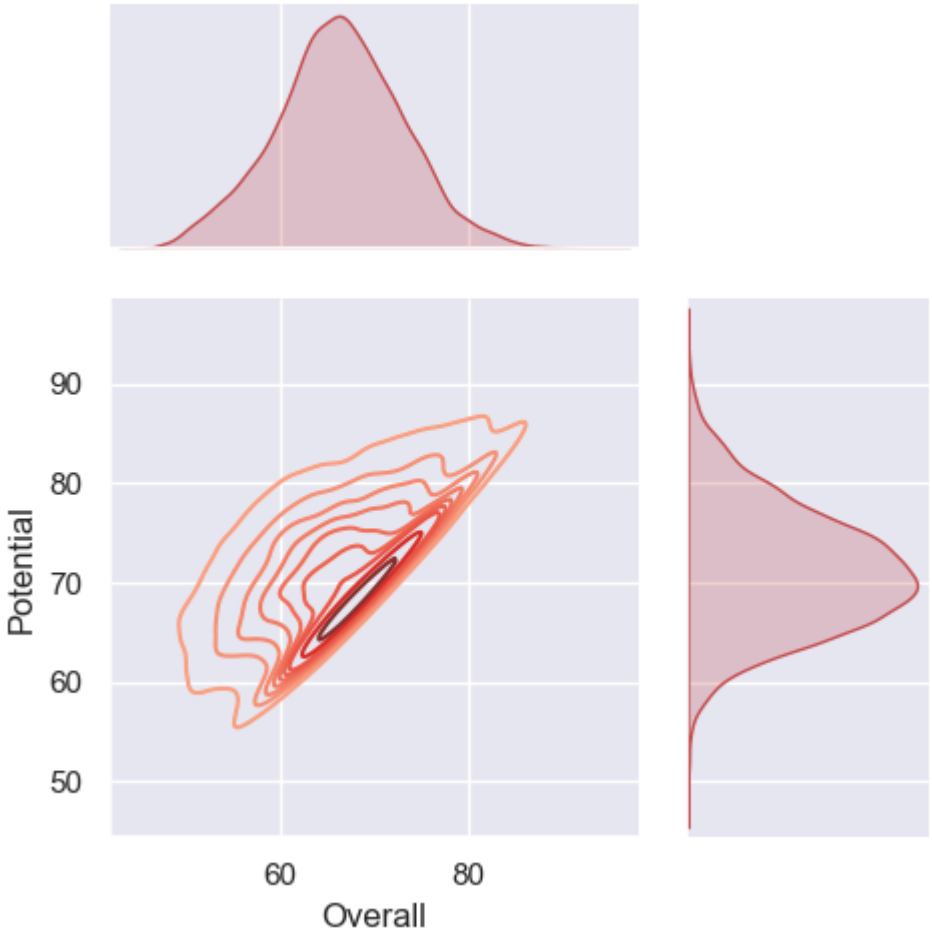
```
g = sns.JointGrid(x="Overall", y="Potential", data=fifa, space=0)
g = g.plot_joint(sns.kdeplot, cmap="Blues_d")
g = g.plot_marginals(sns.kdeplot, shade=True)
plt.show()
```



We can draw a smaller plot with relatively larger marginal axes as :

In [158…

```
g = sns.JointGrid(x="Overall", y="Potential", data=fifa, height=5, ratio=2)
g = g.plot_joint(sns.kdeplot, cmap="Reds_d")
g = g.plot_marginals(sns.kdeplot, color="r", shade=True)
plt.show()
```

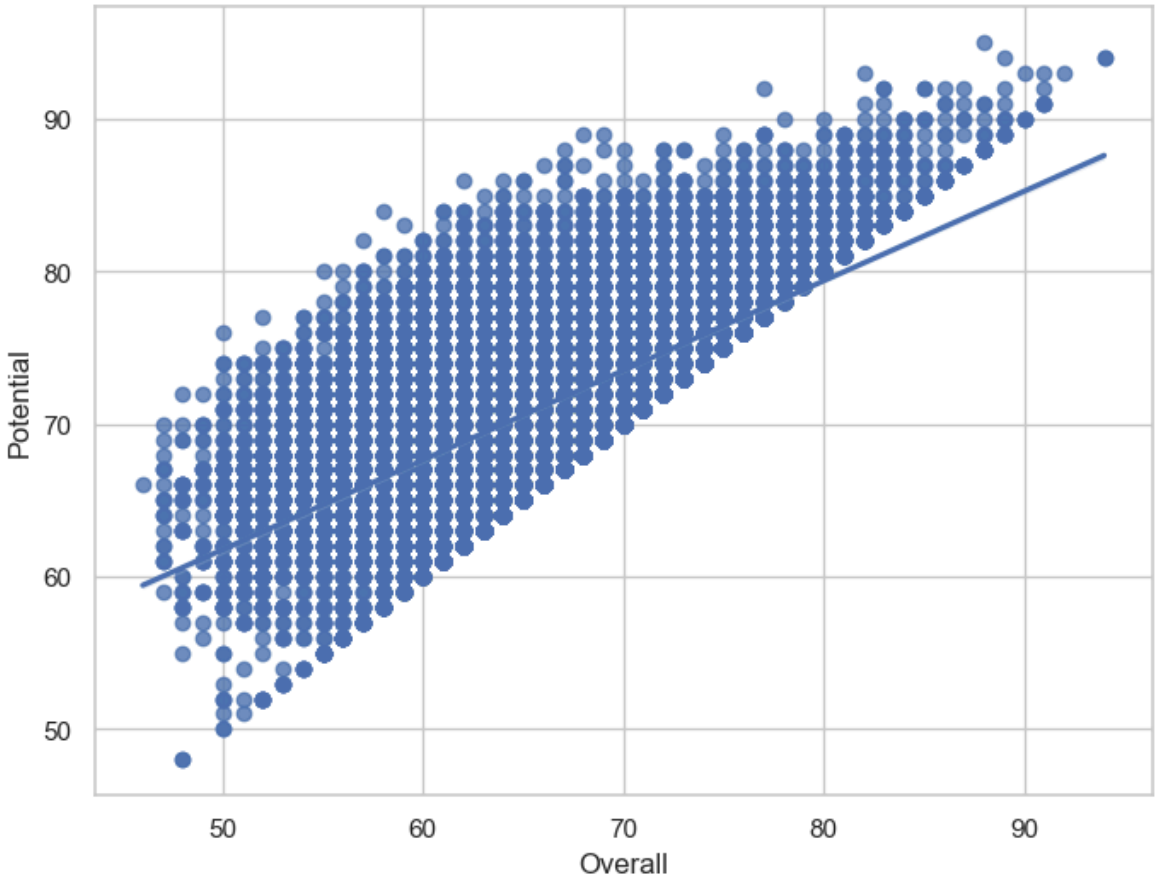


In [159…

```
# controlling the shape and size of the plot
```

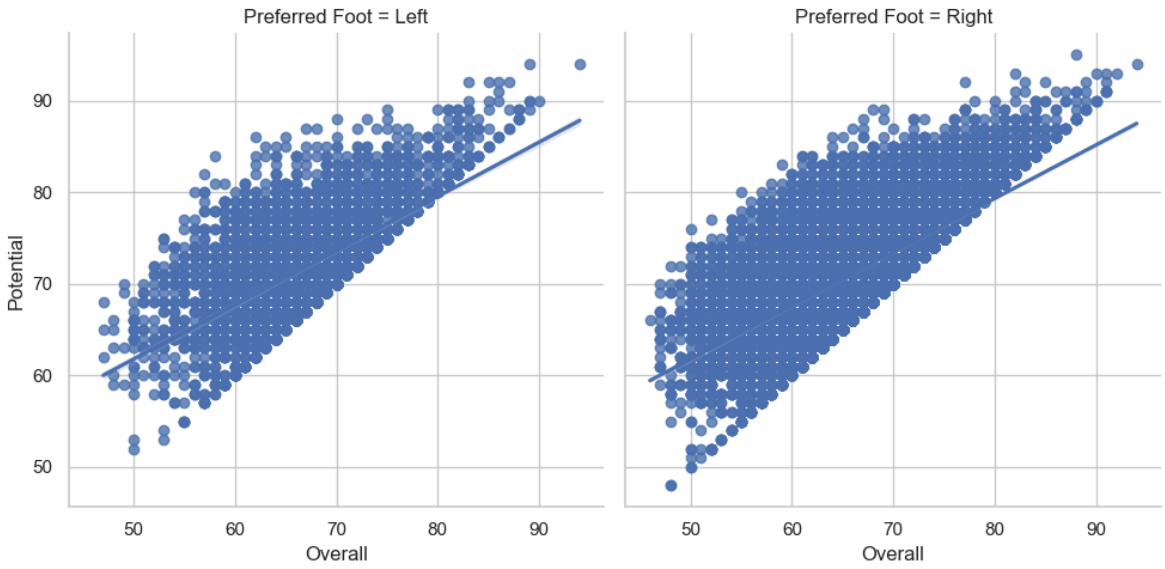
In [116…

```
ax = plt.subplots(figsize=(8, 6))
ax = sns.regplot(x="Overall", y="Potential", data= fifa)
plt.show()
```



In [117…

```
sns.lmplot(x="Overall", y="Potential", col="Preferred Foot", data=fifa, col_wrap
plt.show()
```



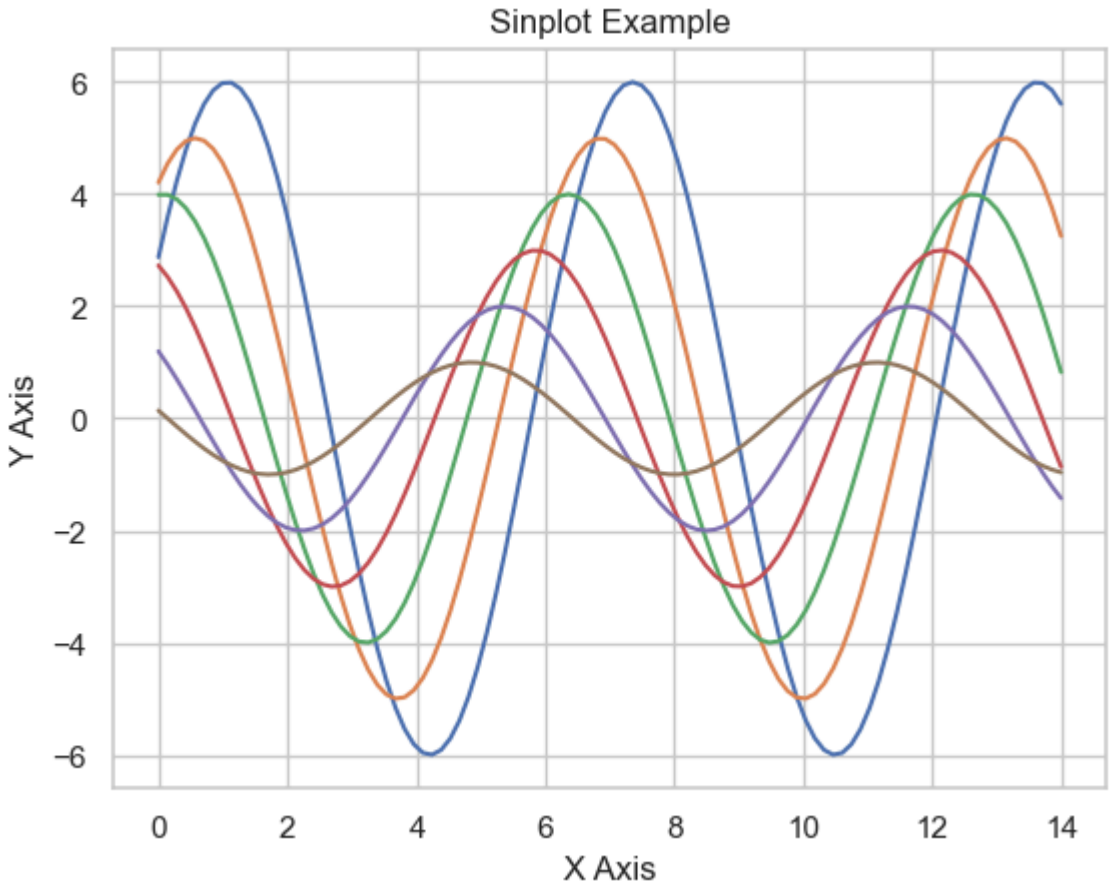
SEABORN FIGURE STYLES

In [145…

```
def sinplot(flip=1):
    x = np.linspace(0, 14, 100)  # Fixed typo here
    for i in range(1, 7):
        plt.plot(x, np.sin(x + i * 0.5) * (7 - i) * flip)  # Proper indentation
    plt.show()
```

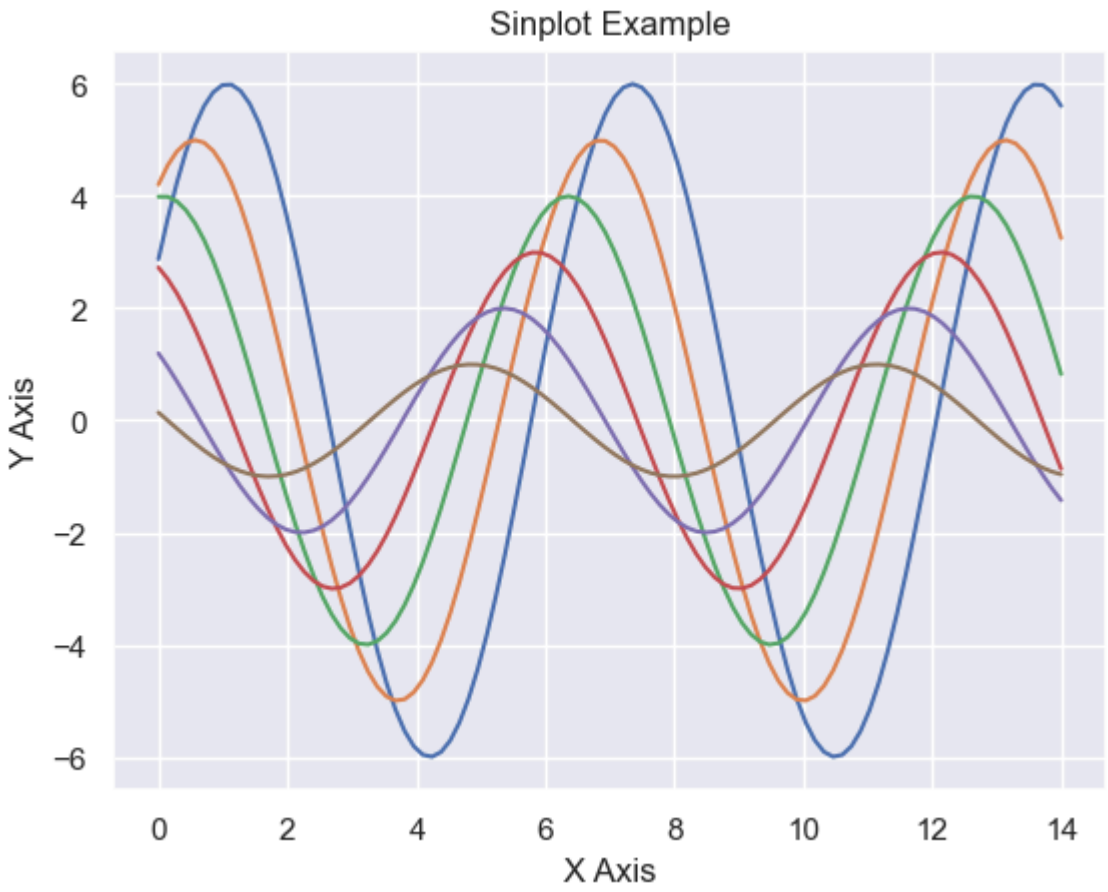
In [153…

```
def sinplot(flip=1):
    x = np.linspace(0, 14, 100)
    for i in range(1, 7):
        plt.plot(x, np.sin(x + i * 0.5) * (7 - i) * flip)
    plt.title("Sinplot Example")
    plt.xlabel("X Axis")
    plt.ylabel("Y Axis")
    plt.grid(True)
    plt.show()
sinplot()
```



In [155…

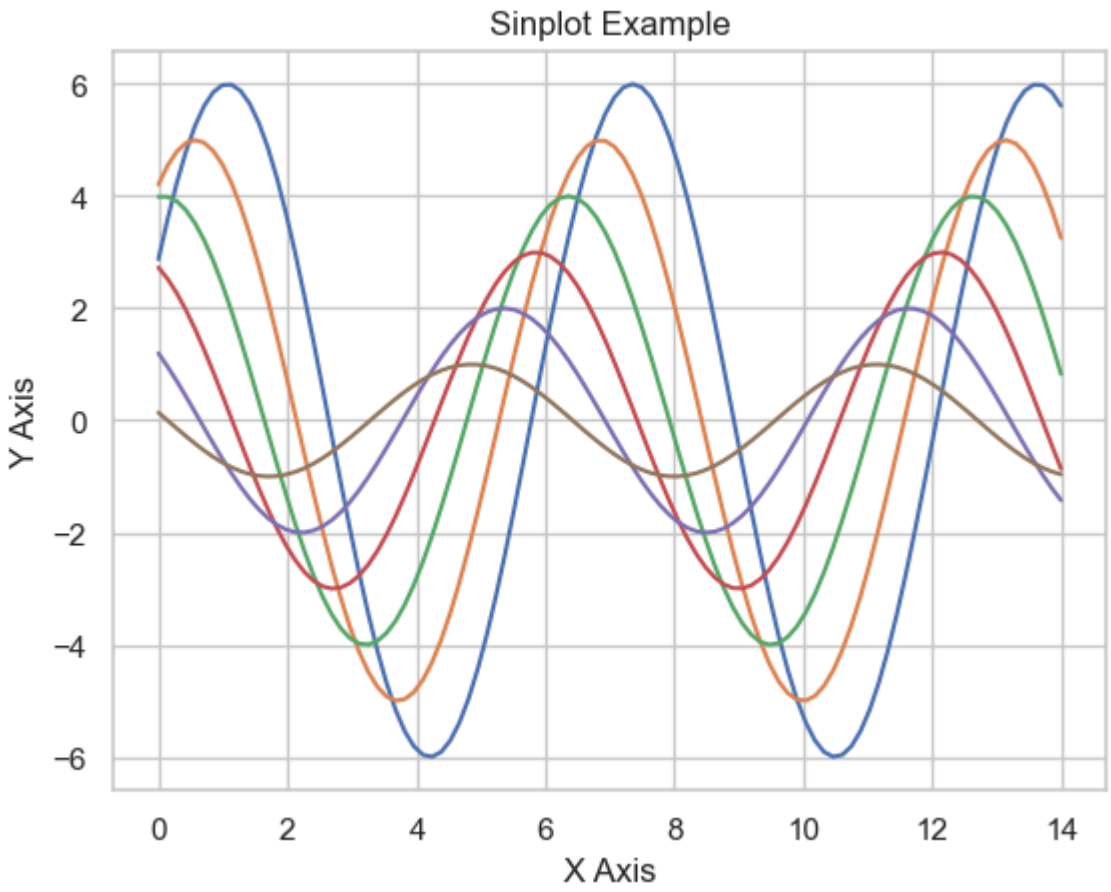
```
sns.set()
sinplot()
```



WE CAN SET DIFFERENT STYLES ALSO :

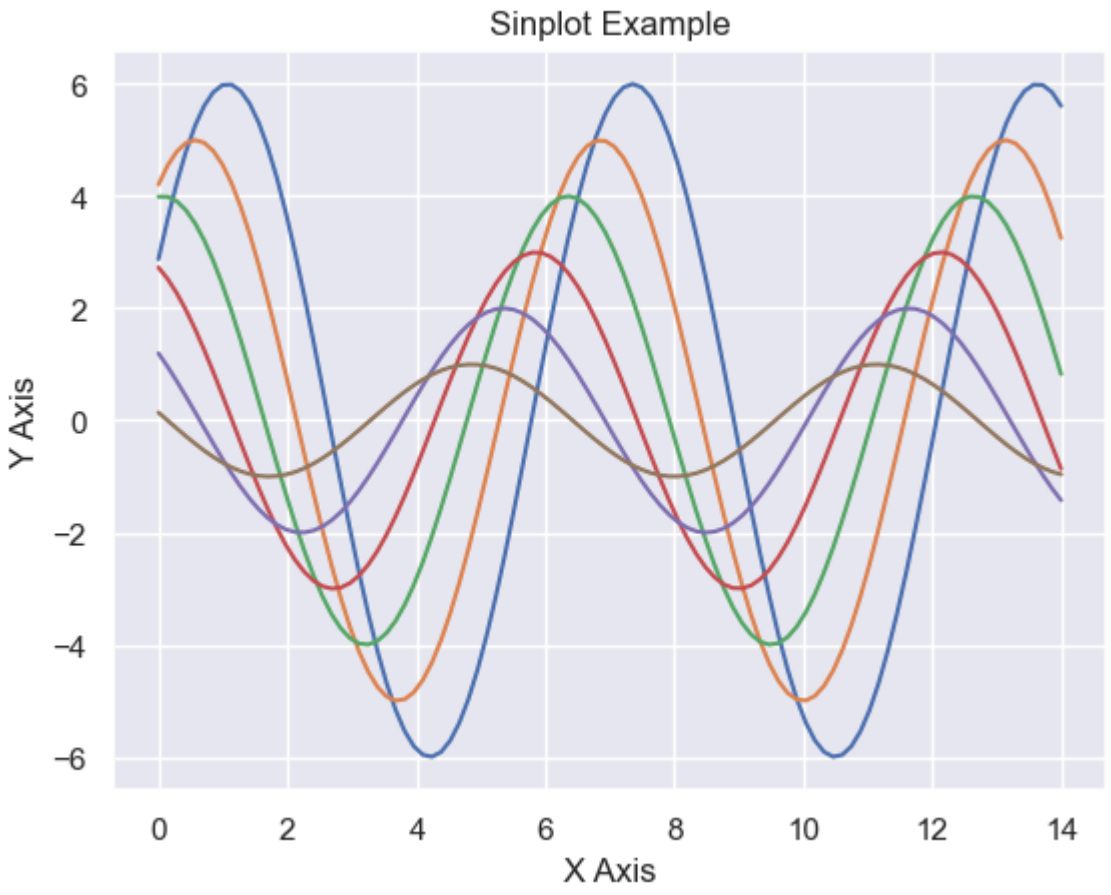
In [162…

```
sns.set_style("whitegrid")
sinplot()
```



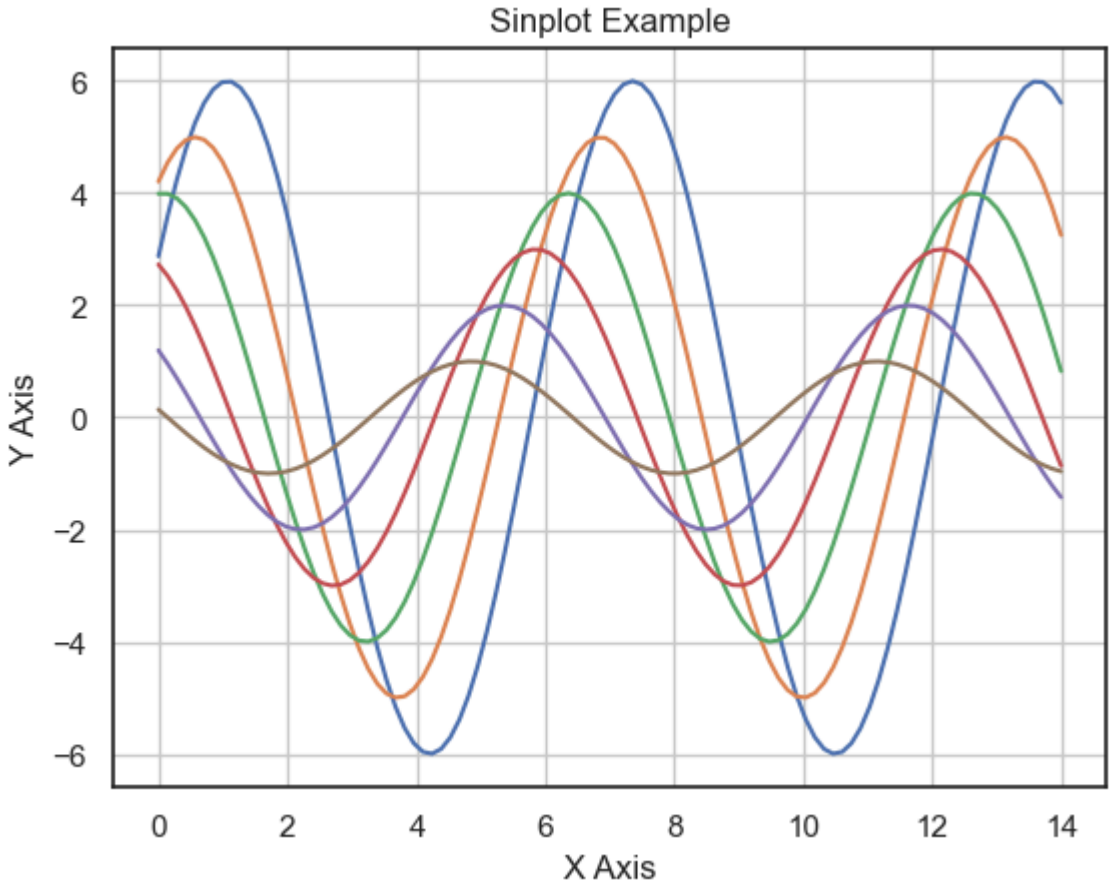
In [168…

```
sns.set_style("dark")
sinplot()
```



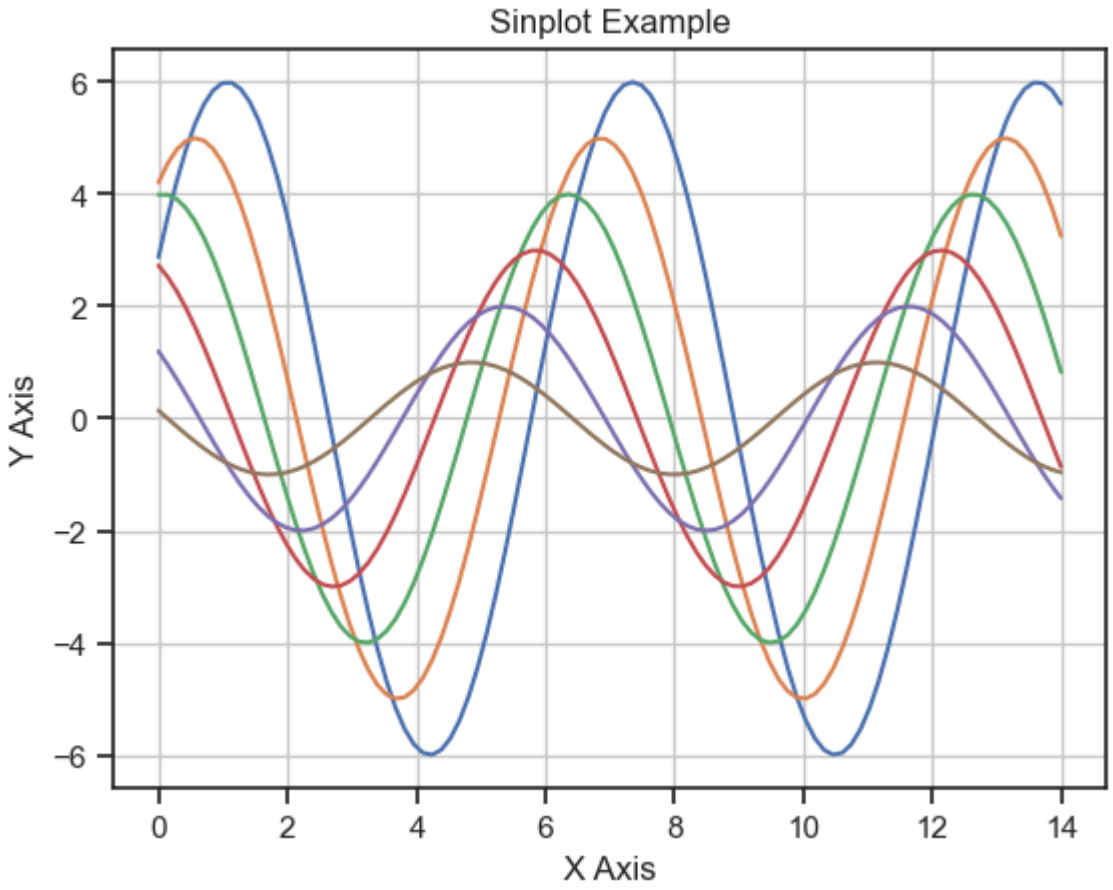
In [172…

```
sns.set_style("white")
sinplot()
```



In [174…

```
sns.set_style("ticks")
sinplot()
```



COMPLETED

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]: