In [1]:
```
import numpy as np
import pandas as pd
import seaborn as sns
sns.set(style="whitegrid")
import matplotlib.pyplot as plt
from collections import Counter
%matplotlib inline


import os
for dirname,_,filenames in os.walk('\kaggle\input'):
    for filename in filenames:
        print(os.path.join(dirname,filename))
```

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

In [3]:
```
fifa19=pd.read_csv(r"D:\Data Science with AI\FIFA.csv")
```

In [4]:
```
fifa19
```

Out[4]:

	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

In [5]:
```
fifa19.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]:
```
fifa19.info()
```

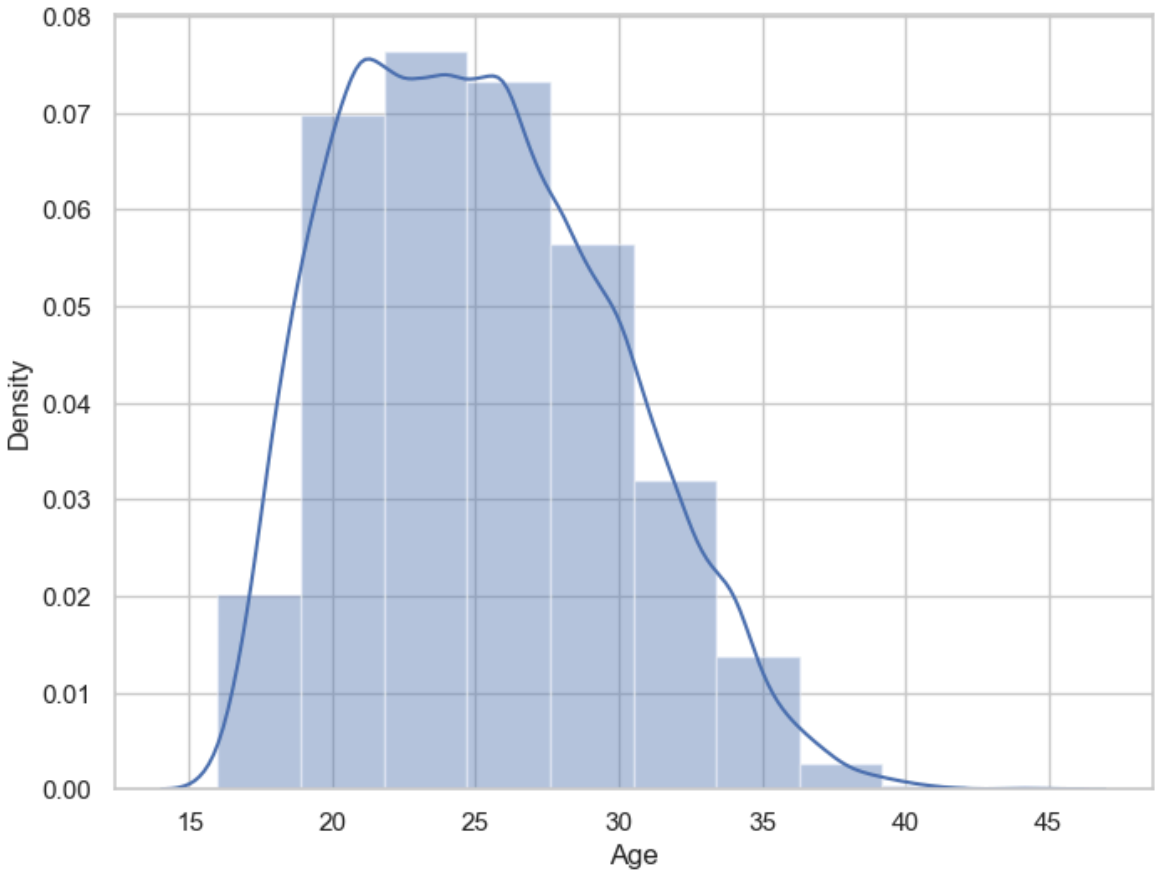
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 18207 entries, 0 to 18206
Data columns (total 89 columns):
 #   Column                    Non-Null Count  Dtype  
---  ------                    --------------  -----  
 0   Unnamed: 0                18207 non-null  int64  
 1   ID                        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
```

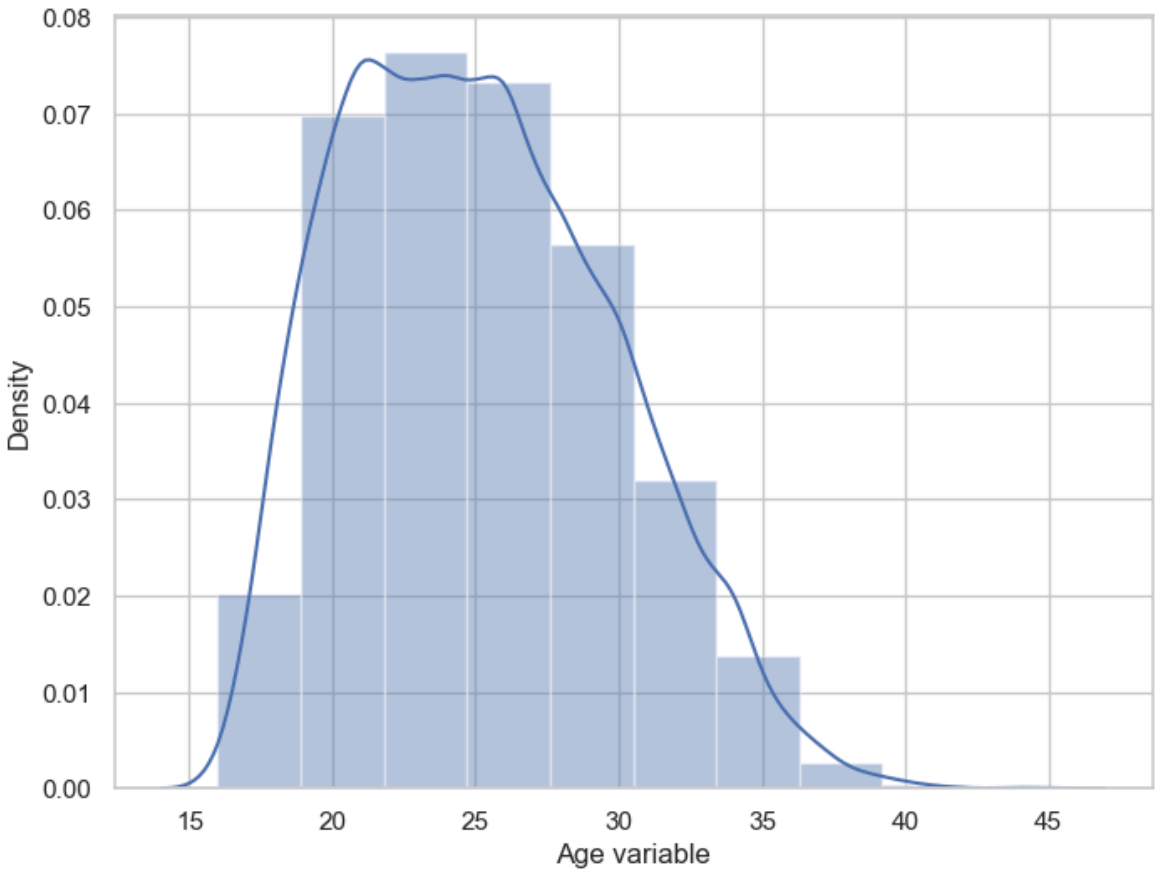
In [7]:
```
fifa19['Body Type'].value_counts()
```

Out[7]:
```
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
```

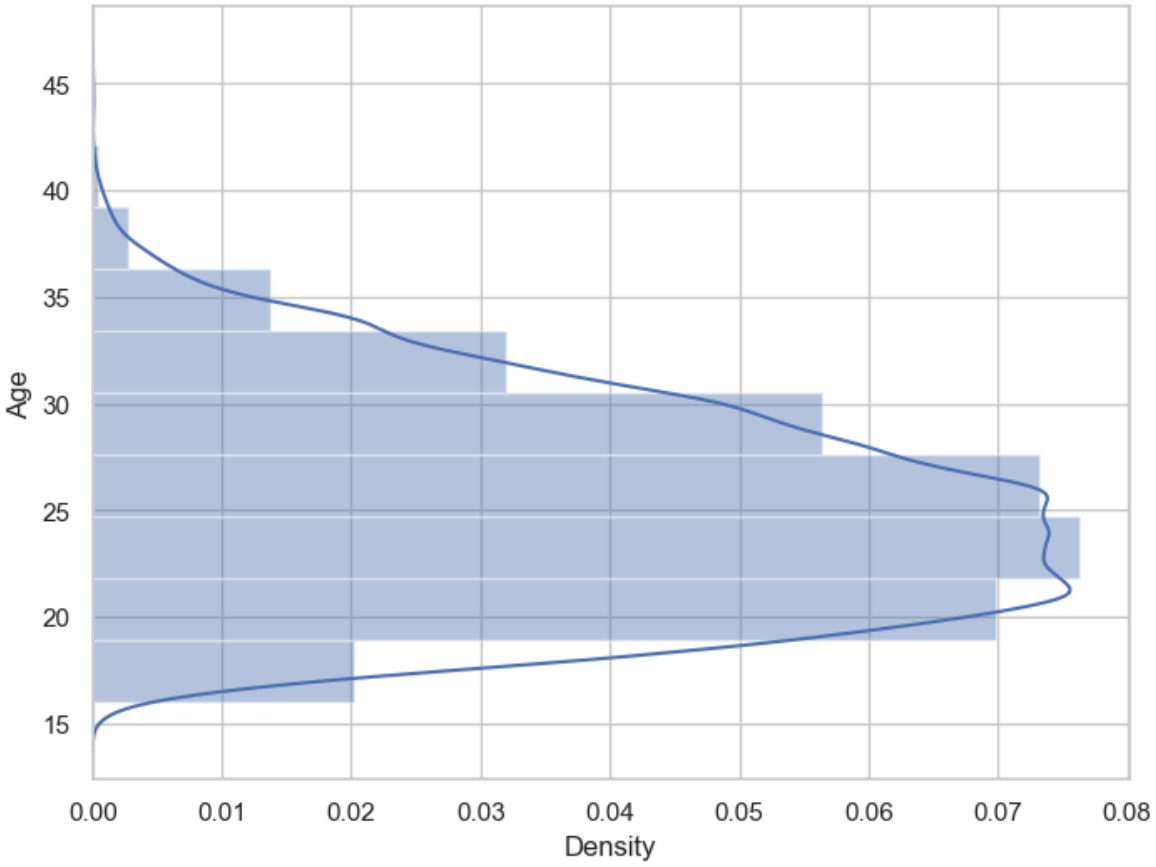
In [8]:
```
f,ax=plt.subplots(figsize=(8,6))
x=fifa19['Age']
ax=sns.distplot(x,bins=10)
plt.show()
```



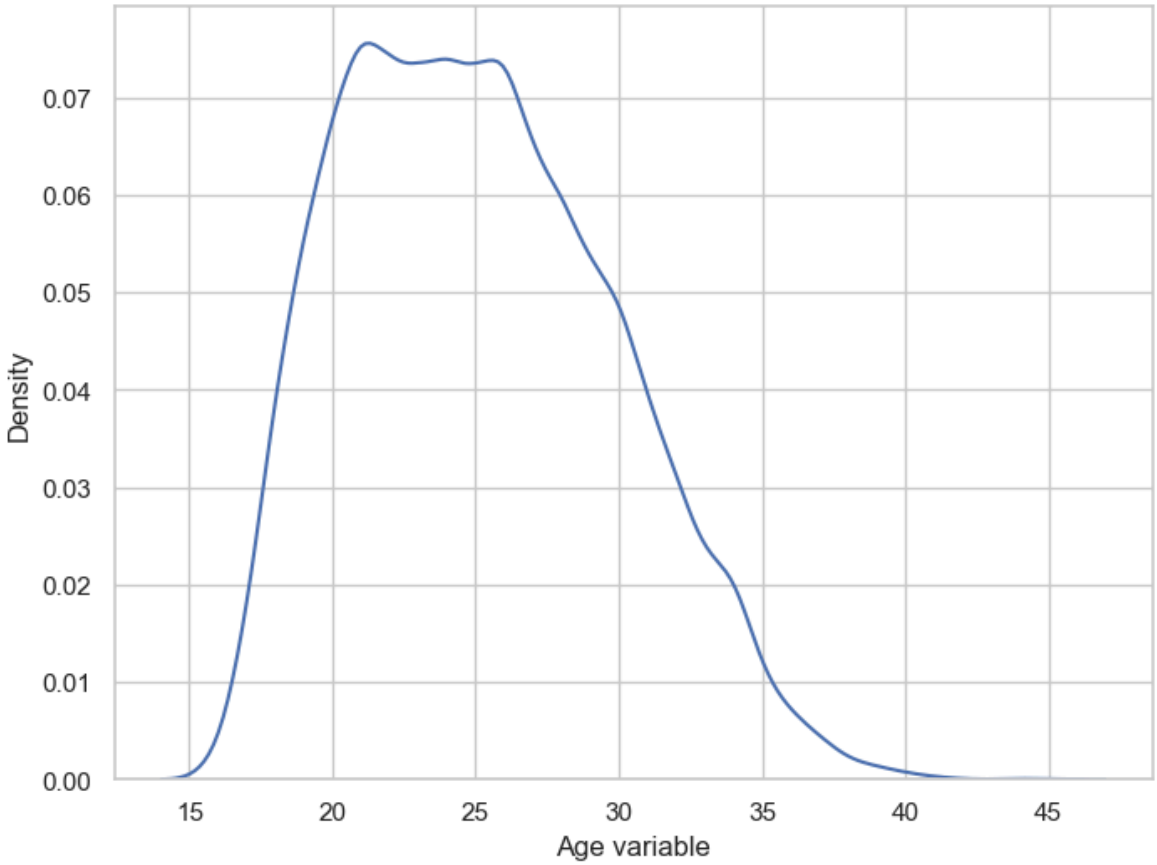
In [9]:
```
f,ax=plt.subplots(figsize=(8,6))
x=fifa19['Age']
x=pd.Series(x,name="Age variable")
ax=sns.distplot(x,bins=10)
plt.show()
```



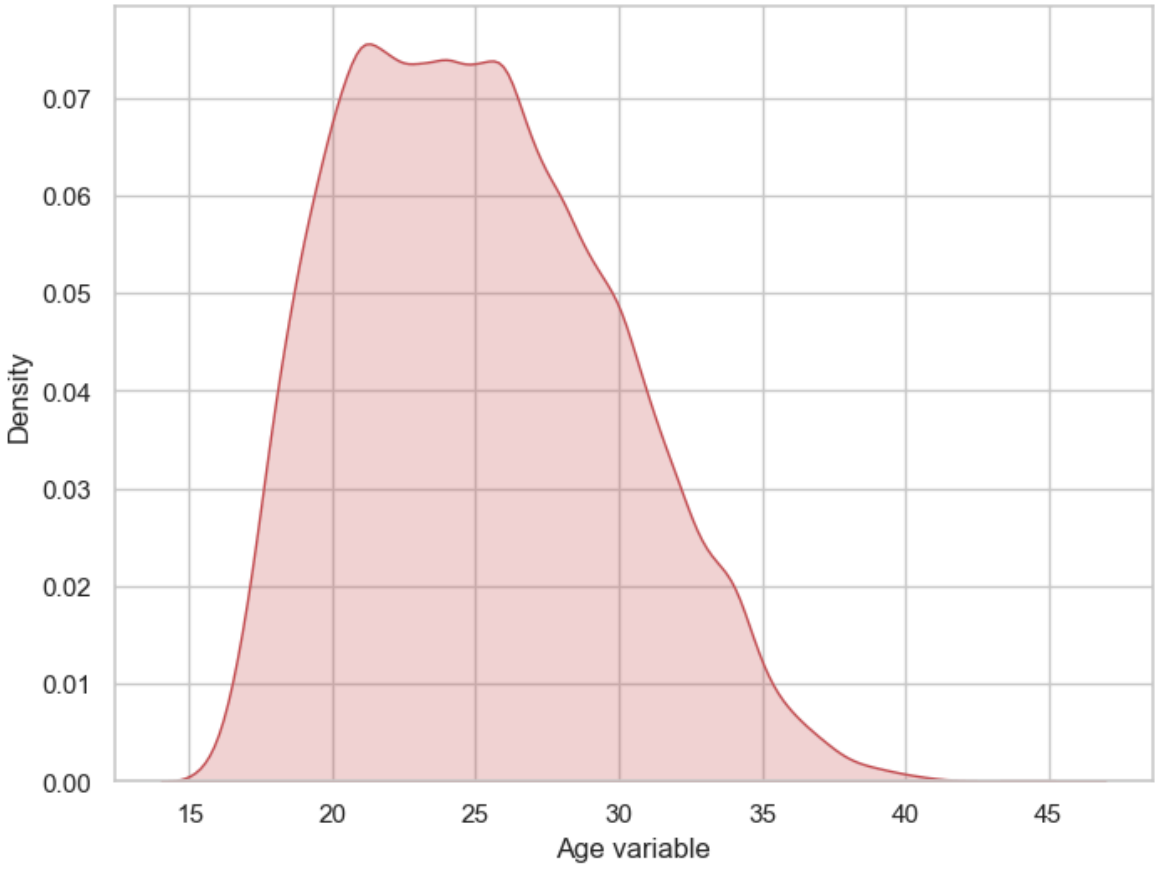
In [10]:
```
f,ax=plt.subplots(figsize=(8,6))
x=fifa19['Age']
ax=sns.distplot(x,bins=10,vertical=True)
plt.show()
```



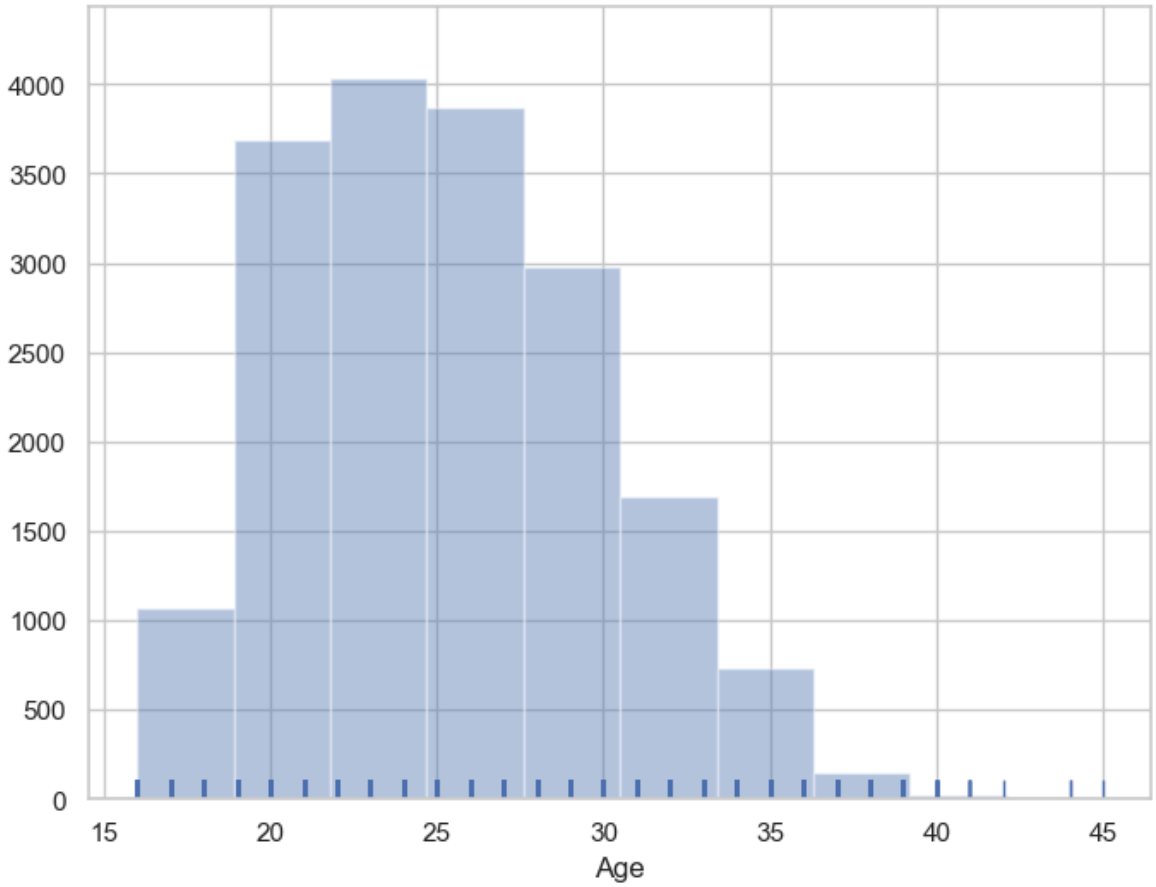
In [11]:
```
f,ax=plt.subplots(figsize=(8,6))
x=fifa19['Age']
x=pd.Series(x,name="Age variable")
ax=sns.kdeplot(x)
plt.show()
```



In [12]:
```
f,ax=plt.subplots(figsize=(8,6))
x=fifa19['Age']
x=pd.Series(x,name="Age variable")
ax=sns.kdeplot(x,shade=True,color='r')
plt.show()
```



In [13]:
```
f,ax=plt.subplots(figsize=(8,6))
x=fifa19['Age']
ax=sns.distplot(x,kde=False,rug=True,bins=10)
plt.show()
```

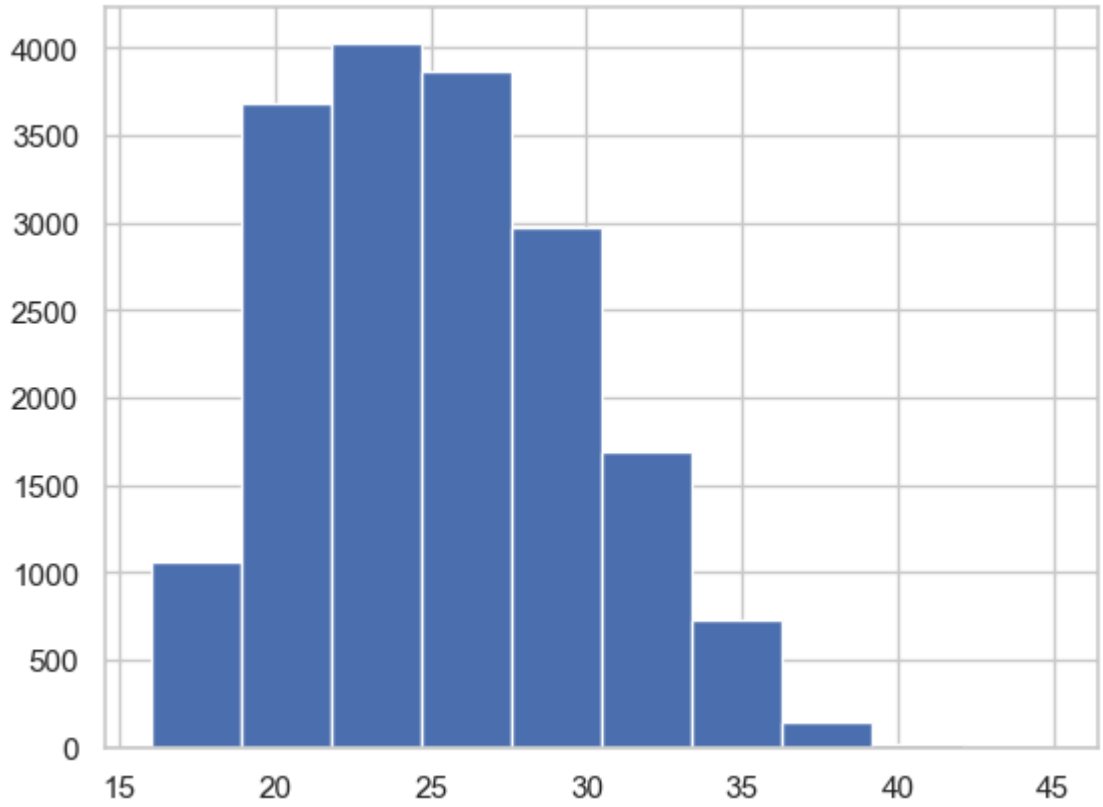


In [14]:
```
sns.hist(fifa19['Age'])
```

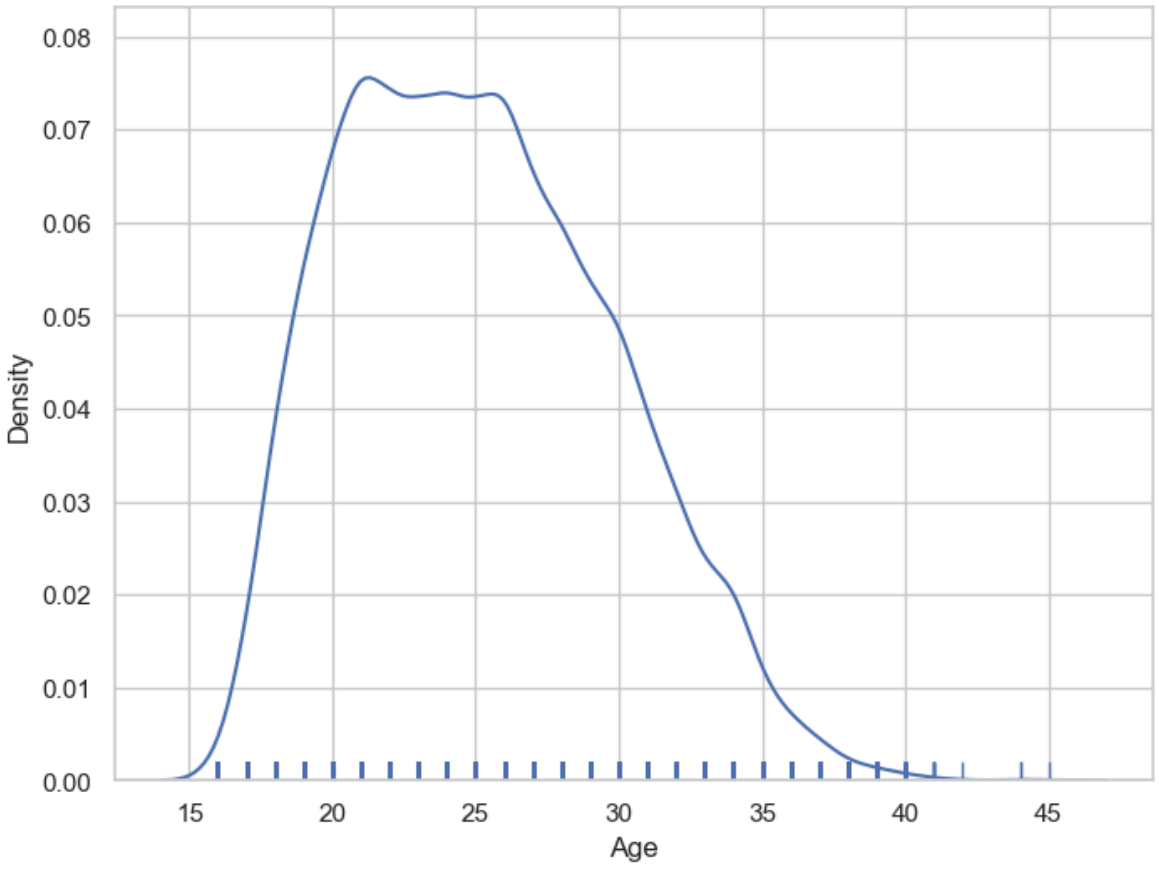
```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
Cell In[14], line 1
----> 1 sns.hist(fifa19['Age'])

AttributeError: module 'seaborn' has no attribute 'hist'
```

In [15]:
```
plt.hist(fifa19['Age'])
```

Out[15]:
```
(array([1.063e+03, 3.687e+03, 4.030e+03, 3.868e+03, 2.977e+03, 1.689e+03,
        7.270e+02, 1.440e+02, 1.900e+01, 3.000e+00]),
 array([16. , 18.9, 21.8, 24.7, 27.6, 30.5, 33.4, 36.3, 39.2, 42.1, 45. ]),
 <BarContainer object of 10 artists>)
```



In [16]:
```
f,ax=plt.subplots(figsize=(8,6))
x=fifa19['Age']
ax=sns.distplot(x,hist=False,rug=True,bins=10)
plt.show()
```



In [17]:
```
fifa19['Preffered Foot'].nunique()
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
File D:\New folder\Lib\site-packages\pandas\core\indexes\base.py:3805, in Index.g
et_loc(self, key)
   3804 try:
-> 3805     return self._engine.get_loc(casted_key)
   3806 except KeyError as err:

File index.pyx:167, in pandas._libs.index.IndexEngine.get_loc()

File index.pyx:196, in pandas._libs.index.IndexEngine.get_loc()

File pandas\\_libs\\hashtable_class_helper.pxi:7081, in pandas._libs.hashtable.Py
ObjectHashTable.get_item()

File pandas\\_libs\\hashtable_class_helper.pxi:7089, in pandas._libs.hashtable.Py
ObjectHashTable.get_item()

KeyError: 'Preffered Foot'

The above exception was the direct cause of the following exception:

KeyError                                  Traceback (most recent call last)
Cell In[17], line 1
----> 1 fifa19[                ].nunique()

File D:\New folder\Lib\site-packages\pandas\core\frame.py:4102, in DataFrame.__ge
titem__(self, key)
   4100 if self.columns.nlevels > 1:
   4101     return self._getitem_multilevel(key)
-> 4102 indexer = self.columns.get_loc(key)
   4103 if is_integer(indexer):
   4104     indexer = [indexer]

File D:\New folder\Lib\site-packages\pandas\core\indexes\base.py:3812, in Index.g
et_loc(self, key)
   3807     if isinstance(casted_key, slice) or (
   3808         isinstance(casted_key, abc.Iterable)
   3809         and any(isinstance(x, slice) for x in casted_key)
   3810     ):
   3811         raise InvalidIndexError(key)
-> 3812     raise KeyError(key) from err
   3813 except TypeError:
   3814     # If we have a listlike key, _check_indexing_error will raise
   3815     #  InvalidIndexError. Otherwise we fall through and re-raise
   3816     #  the TypeError.
   3817     self._check_indexing_error(key)

KeyError: 'Preffered Foot'
```

In [ ]:
```
fifa19['Preferred Foot'].nunique()
```

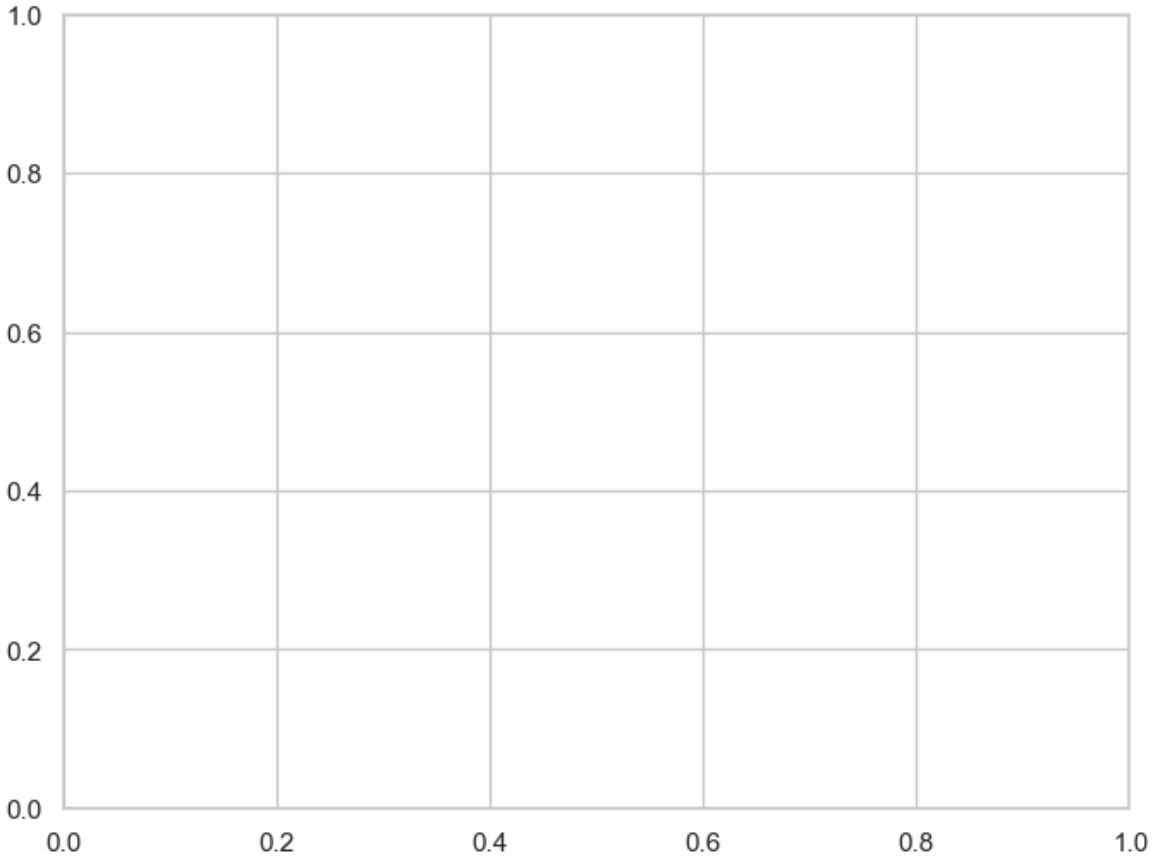
In [ ]:
```
fifa19['Preferred Foot']
```

In [ ]:
```
fifa19['Preferred Foot'].value_counts()
```

In [18]:
```
f,ax=plt.subplots(figsize=(8,6))
sns.countplot(x="preferred Foot",data=fifa19,color="c")
plt.show()
```

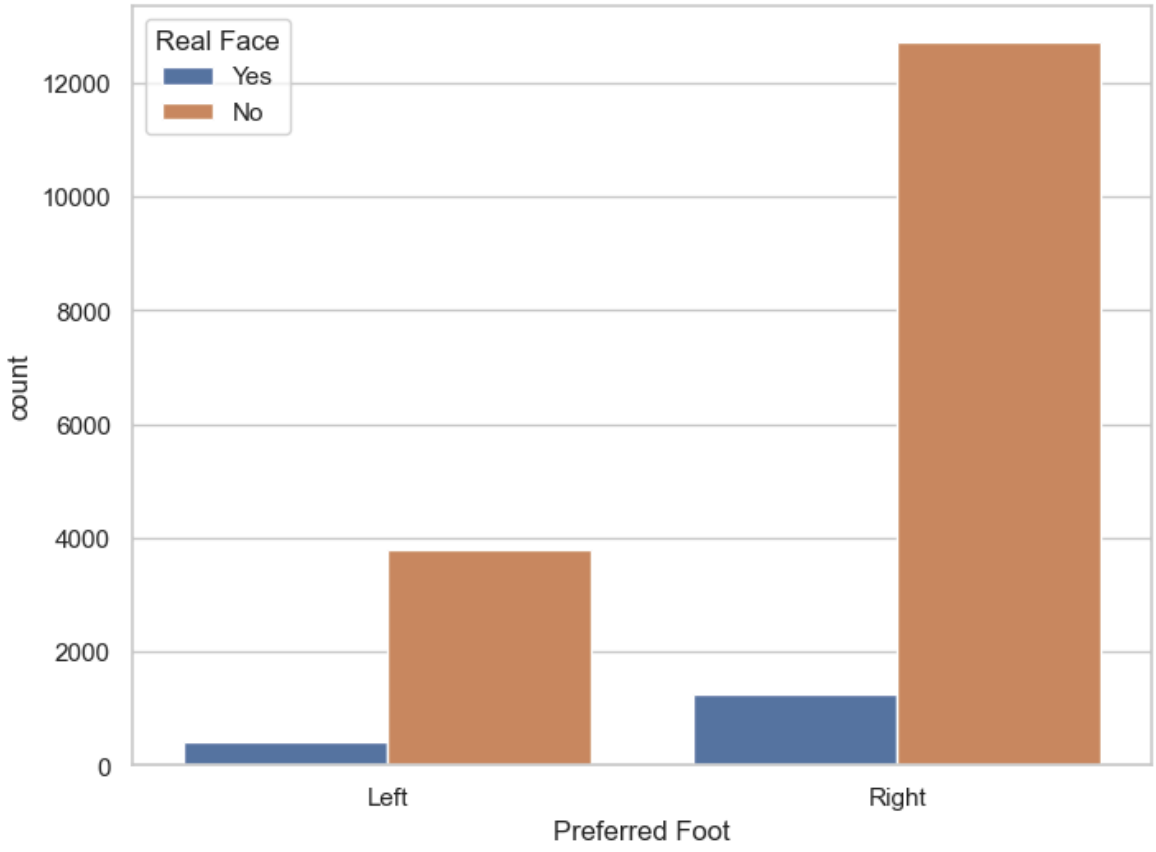
```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
Cell In[18], line 2
      1 f,ax=plt.subplots(figsize=(8,6))
----> 2 sns.countplot(x=                ,data=fifa19,color=    )
      3 plt.show()

File D:\New folder\Lib\site-packages\seaborn\categorical.py:2631, in countplot(da
ta, x, y, hue, order, hue_order, orient, color, palette, saturation, fill, hue_no
rm, stat, width, dodge, gap, log_scale, native_scale, formatter, legend, ax, **kw
args)
   2628 elif x is not None and y is not None:
   2629     raise TypeError("Cannot pass values for both `x` and `y`.")
-> 2631 p = _CategoricalAggPlotter(
   2632     data=data,
   2633     variables=dict(x=x, y=y, hue=hue),
   2634     order=order,
   2635     orient=orient,
   2636     color=color,
   2637     legend=legend,
   2638 )
   2640 if ax is None:
   2641     ax = plt.gca()

File D:\New folder\Lib\site-packages\seaborn\categorical.py:67, in _CategoricalPl
otter.__init__(self, data, variables, order, orient, require_numeric, color, lege
nd)
     56 def __init__(
     57     self,
     58     data=None,
   (...)     64     legend="auto",
     65 ):
---> 67     super().__init__(data=data, variables=variables)
     69     # This method takes care of some bookkeeping that is necessary becaus
e the
     70     # original categorical plots (prior to the 2021 refactor) had some ru
les that
     71     # don't fit exactly into VectorPlotter logic. It may be wise to have
a second
   (...)     76     # default VectorPlotter rules. If we do decide to make orient
part of the
     77     # _base variable assignment, we'll want to figure out how to express
that.
     78     if self.input_format == "wide" and orient in ["h", "y"]:

File D:\New folder\Lib\site-packages\seaborn\_base.py:634, in VectorPlotter.__ini
t__(self, data, variables)
    629 # var_ordered is relevant only for categorical axis variables, and may
    630 # be better handled by an internal axis information object that tracks
    631 # such information and is set up by the scale_* methods. The analogous
    632 # information for numeric axes would be information about log scales.
    633 self._var_ordered = {"x": False, "y": False}  # alt., used DefaultDict
--> 634 self.assign_variables(data, variables)
    636 # TODO Lots of tests assume that these are called to initialize the
    637 # mappings to default values on class initialization. I'd prefer to
    638 # move away from that and only have a mapping when explicitly called.
    639 for var in ["hue", "size", "style"]:

File D:\New folder\Lib\site-packages\seaborn\_base.py:679, in VectorPlotter.assig
n_variables(self, data, variables)
```

```
    674 else:
    675     # When dealing with long-form input, use the newer PlotData
    676     # object (internal but introduced for the objects interface)
    677     # to centralize / standardize data consumption logic.
    678     self.input_format = "long"
--> 679     plot_data = PlotData(data, variables)
    680     frame = plot_data.frame
    681     names = plot_data.names

File D:\New folder\Lib\site-packages\seaborn\_core\data.py:58, in PlotData.__init
__(self, data, variables)
     51 def __init__(
     52     self,
     53     data: DataSource,
     54     variables: dict[str, VariableSpec],
     55 ):
     57     data = handle_data_source(data)
---> 58     frame, names, ids = self._assign_variables(data, variables)
     60     self.frame = frame
     61     self.names = names

File D:\New folder\Lib\site-packages\seaborn\_core\data.py:232, in PlotData._assi
gn_variables(self, data, variables)
    230     else:
    231         err += "An entry with this name does not appear in `data`."
--> 232     raise ValueError(err)
    234 else:
    235
    236     # Otherwise, assume the value somehow represents data
    237
    238     # Ignore empty data structures
    239     if isinstance(val, Sized) and len(val) == 0:

ValueError: Could not interpret value `preferred Foot` for `x`. An entry with thi
s name does not appear in `data`.
```

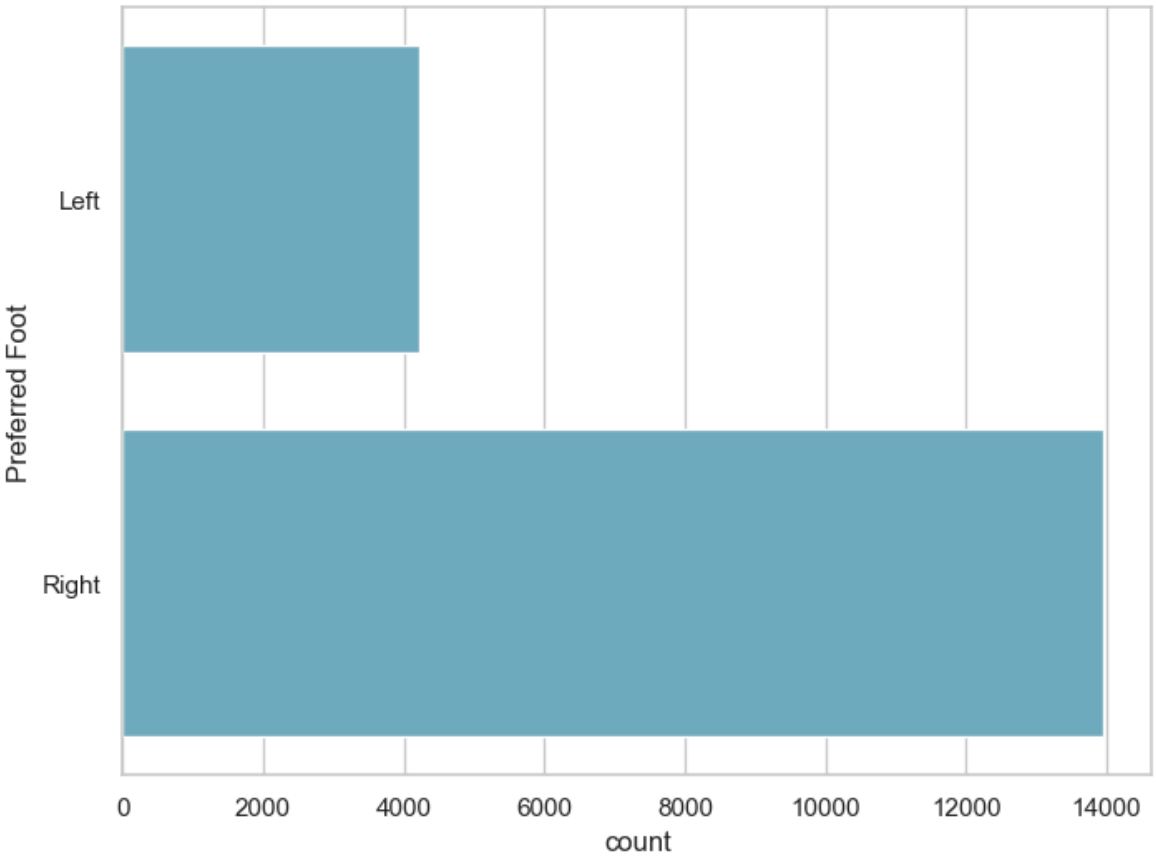


In [ ]:
```
f,ax=plt.subplots(figsize=(8,6))
sns.countplot(data=fifa19,x="Preferred Foot",color="c")
plt.show()
```

In [19]:
```
f,ax=plt.subplots(figsize=(8,6))
sns.countplot(x='Preferred Foot',data=fifa19,hue="Real Face")
plt.show()
```



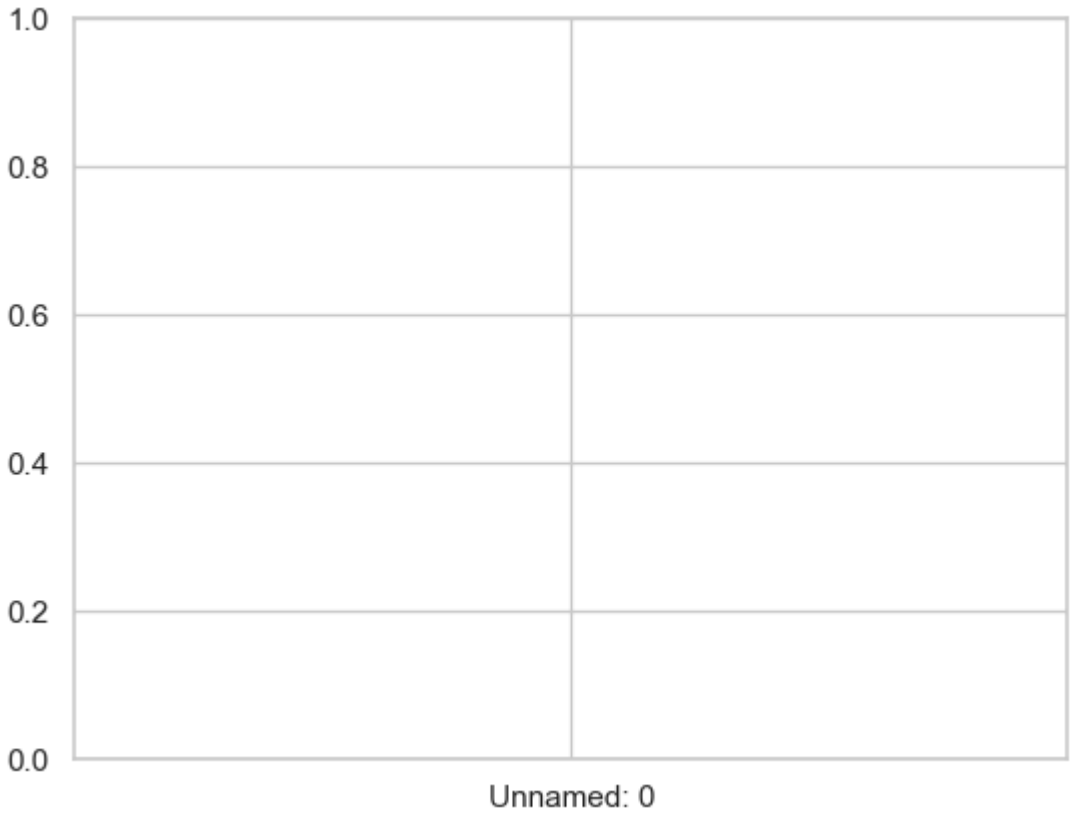
In [20]:
```
f,ax=plt.subplots(figsize=(8,6))
sns.countplot(y="Preferred Foot",data=fifa19,color="c")
plt.show()
```



In [21]:
```
sns.countplot(data=fifa19,vertical=True,color="c")
plt.show()
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
Cell In[21], line 1
----> 1 sns.countplot(data=fifa19,vertical=True,color=    )
      2 plt.show()

File D:\New folder\Lib\site-packages\seaborn\categorical.py:2675, in countplot(da
ta, x, y, hue, order, hue_order, orient, color, palette, saturation, fill, hue_no
rm, stat, width, dodge, gap, log_scale, native_scale, formatter, legend, ax, **kw
args)
   2671     p.plot_data[count_axis] /= len(p.plot_data) / denom
   2673 aggregator = EstimateAggregator("sum", errorbar=None)
-> 2675 p.plot_bars(
   2676     aggregator=aggregator,
   2677     dodge=dodge,
   2678     width=width,
   2679     gap=gap,
   2680     color=color,
   2681     fill=fill,
   2682     capsize=0,
   2683     err_kws={},
   2684     plot_kws=kwargs,
   2685 )
   2687 p._add_axis_labels(ax)
   2688 p._adjust_cat_axis(ax, axis=p.orient)

File D:\New folder\Lib\site-packages\seaborn\categorical.py:1315, in _Categorical
Plotter.plot_bars(self, aggregator, dodge, gap, width, fill, color, capsize, err_
kws, plot_kws)
   1312 else:
   1313     kws.update(color=main_color, edgecolor=main_color, facecolor="none")
-> 1315 bar_func(**{**kws, **plot_kws})
   1317 if aggregator.error_method is not None:
   1318     self.plot_errorbars(
   1319         ax, agg_data, capsize,
   1320         {"color": ".26" if fill else main_color, **err_kws}
   1321     )

File D:\New folder\Lib\site-packages\matplotlib\__init__.py:1521, in _preprocess_
data.<locals>.inner(ax, data, *args, **kwargs)
   1518 @functools.wraps(func)
   1519 def inner(ax, *args, data=None, **kwargs):
   1520     if data is None:
-> 1521         return func(
   1522             ax,
   1523             *map(cbook.sanitize_sequence, args),
   1524             **{k: cbook.sanitize_sequence(v) for k, v in kwargs.items()})
   1526     bound = new_sig.bind(ax, *args, **kwargs)
   1527     auto_label = (bound.arguments.get(label_namer)
   1528                   or bound.kwargs.get(label_namer))

File D:\New folder\Lib\site-packages\matplotlib\axes\_axes.py:2643, in Axes.bar(s
elf, x, height, width, bottom, align, **kwargs)
   2634 for l, b, w, h, c, e, lw, htch, lbl in args:
   2635     r = mpatches.Rectangle(
   2636         xy=(l, b), width=w, height=h,
   2637         facecolor=c,
   (...)   2641         hatch=htch,
   2642         )
-> 2643     r._internal_update(kwargs)
```

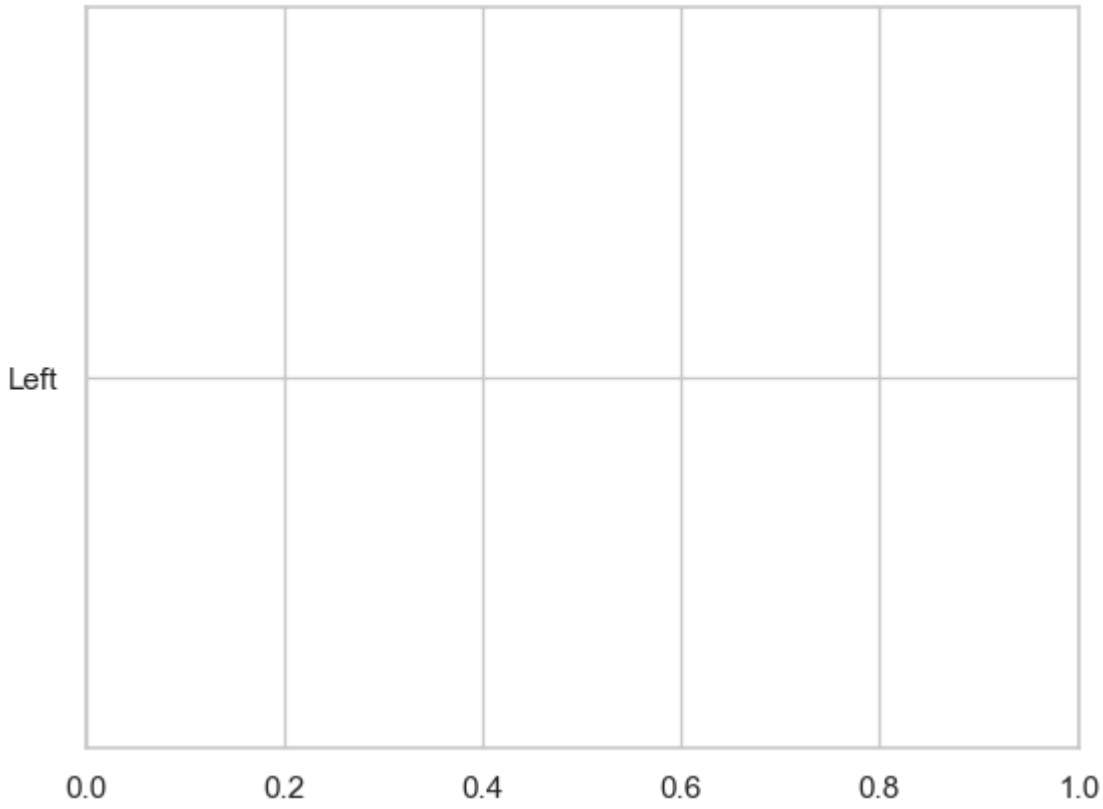
```
   2644     r.get_path()._interpolation_steps = 100
   2645     if orientation == 'vertical':

File D:\New folder\Lib\site-packages\matplotlib\artist.py:1233, in Artist._intern
al_update(self, kwargs)
   1226 def _internal_update(self, kwargs):
   1227     """
   1228     Update artist properties without prenormalizing them, but generating
   1229     errors as if calling `set`.
   1230
   1231     The lack of prenormalization is to maintain backcompatibility.
   1232     """
-> 1233     return self._update_props(
   1234         kwargs,  {cls.__name__}

   1235         {prop_name!r} )

File D:\New folder\Lib\site-packages\matplotlib\artist.py:1206, in Artist._update
_props(self, props, errfmt)
   1204             func = getattr(self, f"set_{k}", None)
   1205             if not callable(func):
-> 1206                 raise AttributeError(
   1207                     errfmt.format(cls=type(self), prop_name=k),
   1208                     name=k)
   1209             ret.append(func(v))
   1210 if ret:

AttributeError: Rectangle.set() got an unexpected keyword argument 'vertical'
```



In [22]:
```
sns.countplot(fifa19['Preferred Foot'],vertical=True)
```

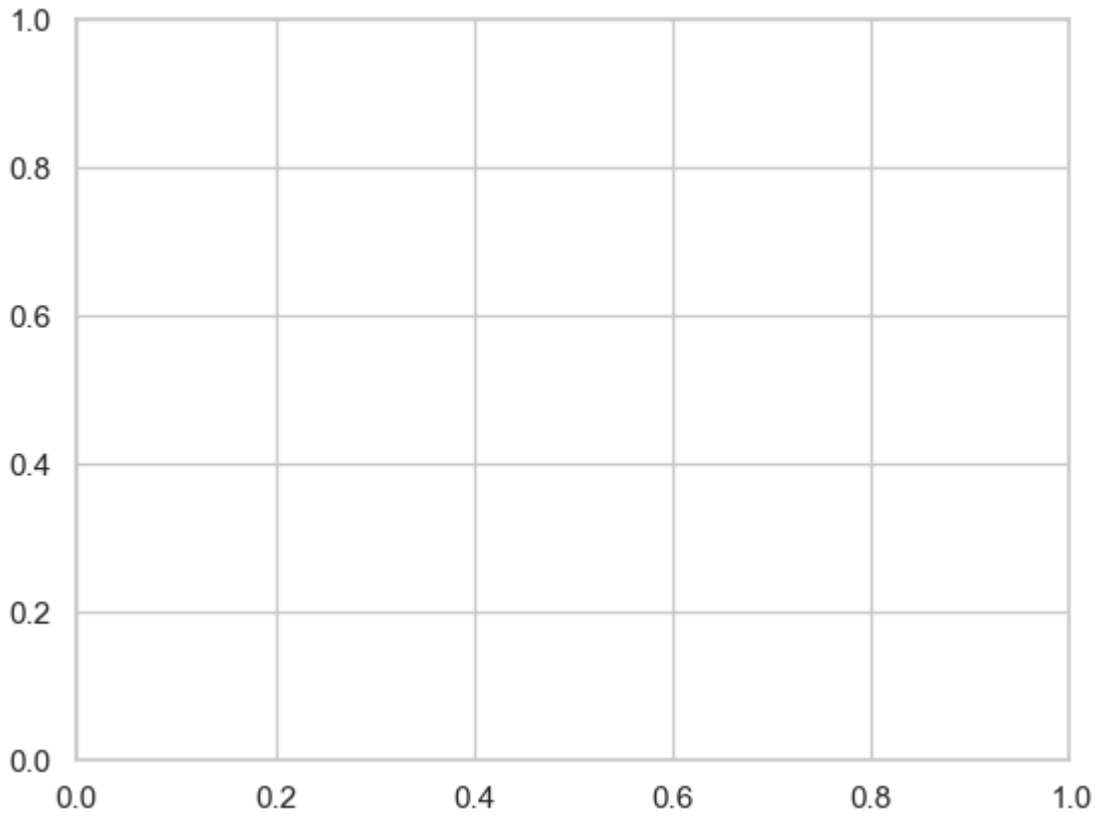
```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
Cell In[22], line 1
----> 1 sns.countplot(fifa19[                ],vertical=True)

File D:\New folder\Lib\site-packages\seaborn\categorical.py:2661, in countplot(da
ta, x, y, hue, order, hue_order, orient, color, palette, saturation, fill, hue_no
rm, stat, width, dodge, gap, log_scale, native_scale, formatter, legend, ax, **kw
args)
   2659 saturation = saturation if fill else 1
   2660 p.map_hue(palette=palette, order=hue_order, norm=hue_norm, saturation=sat
uration)
-> 2661 color = _default_color(ax.bar, hue, color, kwargs, saturation)
   2663 count_axis = {"x": "y", "y": "x"}[p.orient]
   2664 if p.input_format == "wide":

File D:\New folder\Lib\site-packages\seaborn\utils.py:127, in _default_color(meth
od, hue, color, kws, saturation)
    122     scout.remove()
    124 elif method.__name__ == "bar":
    125
    126     # bar() needs masked, not empty data, to generate a patch
--> 127     scout, = method([np.nan], [np.nan], **kws)
    128     color = to_rgb(scout.get_facecolor())
    129     scout.remove()

File D:\New folder\Lib\site-packages\matplotlib\__init__.py:1521, in _preprocess_
data.<locals>.inner(ax, data, *args, **kwargs)
   1518 @functools.wraps(func)
   1519 def inner(ax, *args, data=None, **kwargs):
   1520     if data is None:
-> 1521         return func(
   1522             ax,
   1523             *map(cbook.sanitize_sequence, args),
   1524             **{k: cbook.sanitize_sequence(v) for k, v in kwargs.items()})
   1526     bound = new_sig.bind(ax, *args, **kwargs)
   1527     auto_label = (bound.arguments.get(label_namer)
   1528                   or bound.kwargs.get(label_namer))

File D:\New folder\Lib\site-packages\matplotlib\axes\_axes.py:2643, in Axes.bar(s
elf, x, height, width, bottom, align, **kwargs)
   2634 for l, b, w, h, c, e, lw, htch, lbl in args:
   2635     r = mpatches.Rectangle(
   2636         xy=(l, b), width=w, height=h,
   2637         facecolor=c,
   (...)   2641         hatch=htch,
   2642         )
-> 2643     r._internal_update(kwargs)
   2644     r.get_path()._interpolation_steps = 100
   2645     if orientation == 'vertical':

File D:\New folder\Lib\site-packages\matplotlib\artist.py:1233, in Artist._intern
al_update(self, kwargs)
   1226 def _internal_update(self, kwargs):
   1227     """
   1228     Update artist properties without prenormalizing them, but generating
   1229     errors as if calling `set`.
   1230
   1231     The lack of prenormalization is to maintain backcompatibility.
   1232     """
```

```
-> 1233     return self._update_props(
   1234         kwargs,  {cls.__name__}

   1235         {prop_name!r} )

File D:\New folder\Lib\site-packages\matplotlib\artist.py:1206, in Artist._update
_props(self, props, errfmt)
   1204             func = getattr(self, f"set_{k}", None)
   1205             if not callable(func):
-> 1206                 raise AttributeError(
   1207                     errfmt.format(cls=type(self), prop_name=k),
   1208                     name=k)
   1209             ret.append(func(v))
   1210 if ret:

AttributeError: Rectangle.set() got an unexpected keyword argument 'vertical'
```

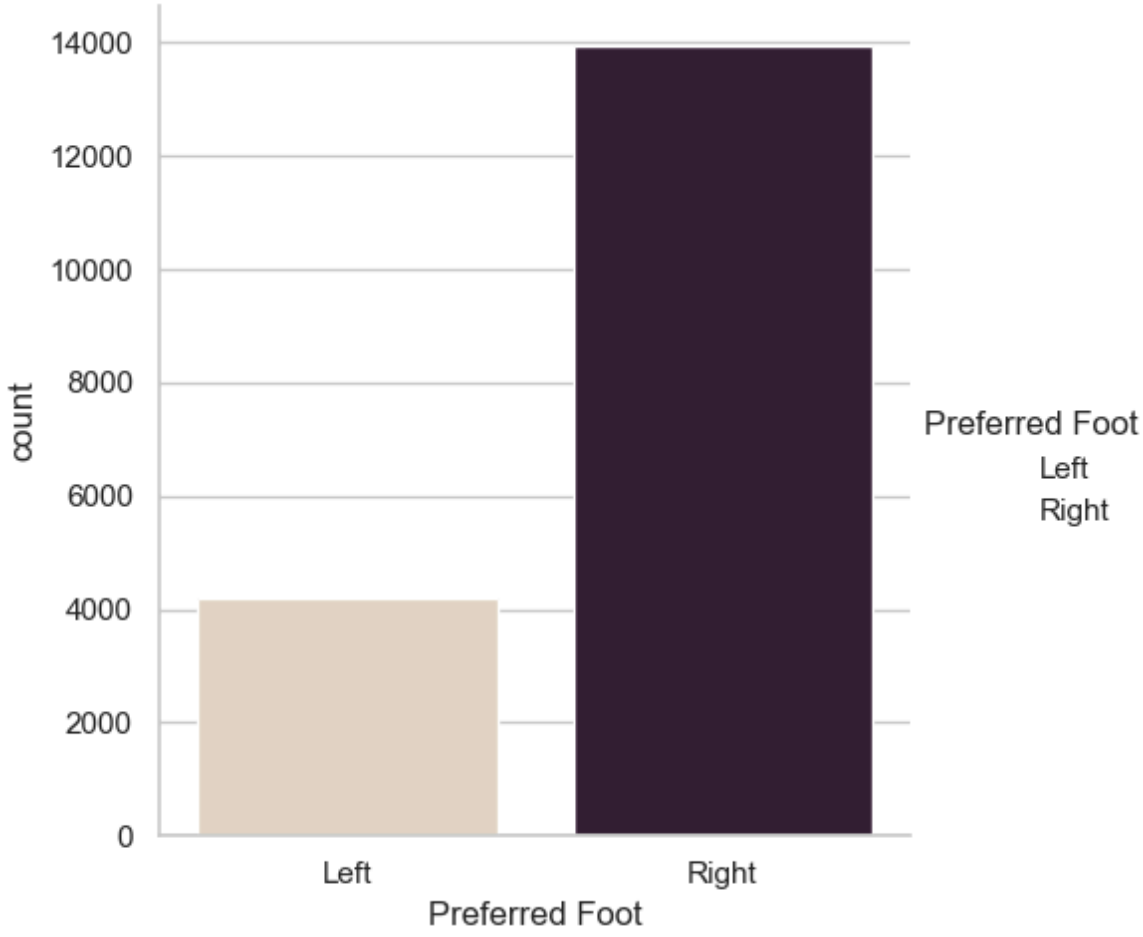


In [23]:
```
sns.distplot(fifa19['Preferred Foot'],vertical=True)
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
Cell In[23], line 1
----> 1 sns.distplot(fifa19[                ],vertical=True)

File D:\New folder\Lib\site-packages\seaborn\distributions.py:2443, in distplot
(a, bins, hist, kde, rug, fit, hist_kws, kde_kws, rug_kws, fit_kws, color, vertic
al, norm_hist, axlabel, label, ax, x)
   2440     a = x
   2442 # Make a a 1-d float array
-> 2443 a = np.asarray(a, float)
   2444 if a.ndim > 1:
   2445     a = a.squeeze()

File D:\New folder\Lib\site-packages\pandas\core\series.py:1031, in Series.__arra
y__(self, dtype, copy)
    981 """
    982 Return the values as a NumPy array.
    983
   (...)   1028       dtype='datetime64[ns]')
   1029 """
   1030 values = self._values
-> 1031 arr = np.asarray(values, dtype=dtype)
   1032 if using_copy_on_write() and astype_is_view(values.dtype, arr.dtype):
   1033     arr = arr.view()

ValueError: could not convert string to float: 'Left'
```



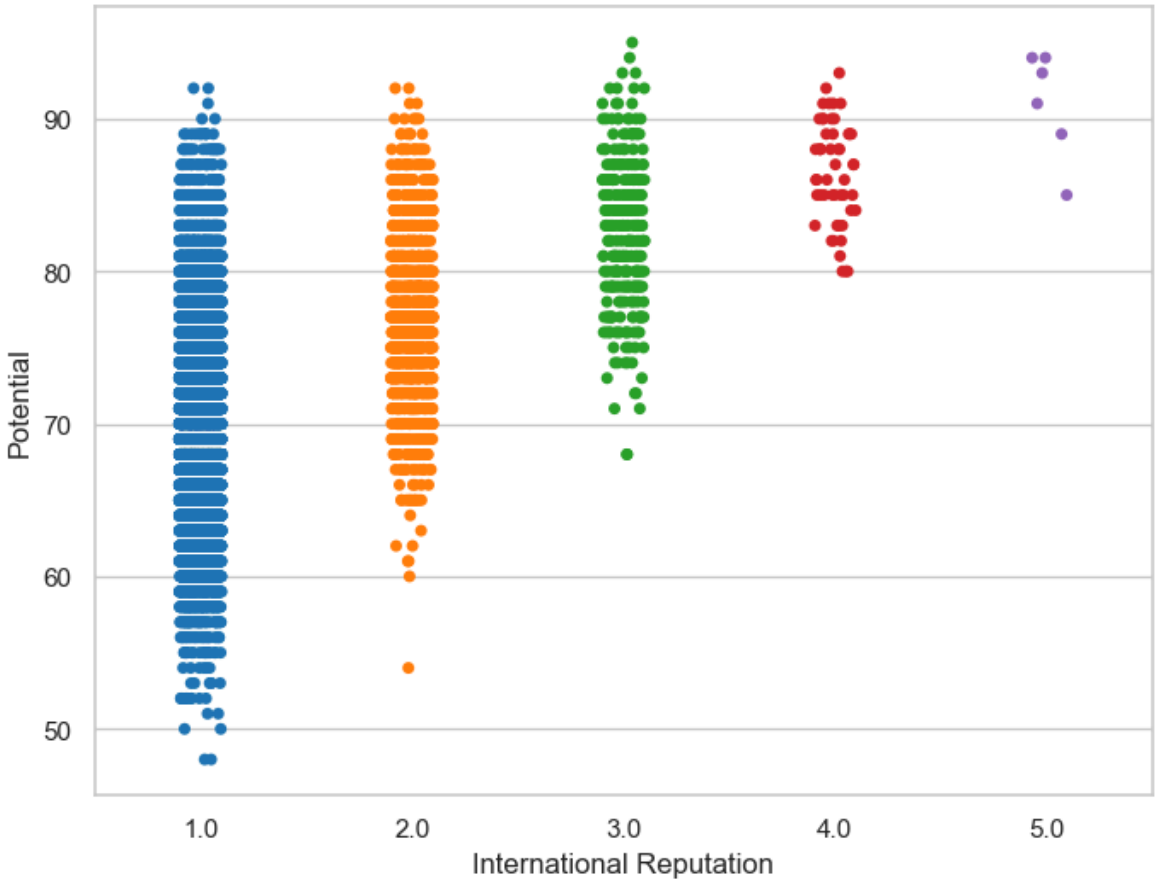
In [24]:
```
g=sns.catplot(x="Preferred Foot",kind="count",palette="ch:.25",data=fifa19)
plt.show()
```



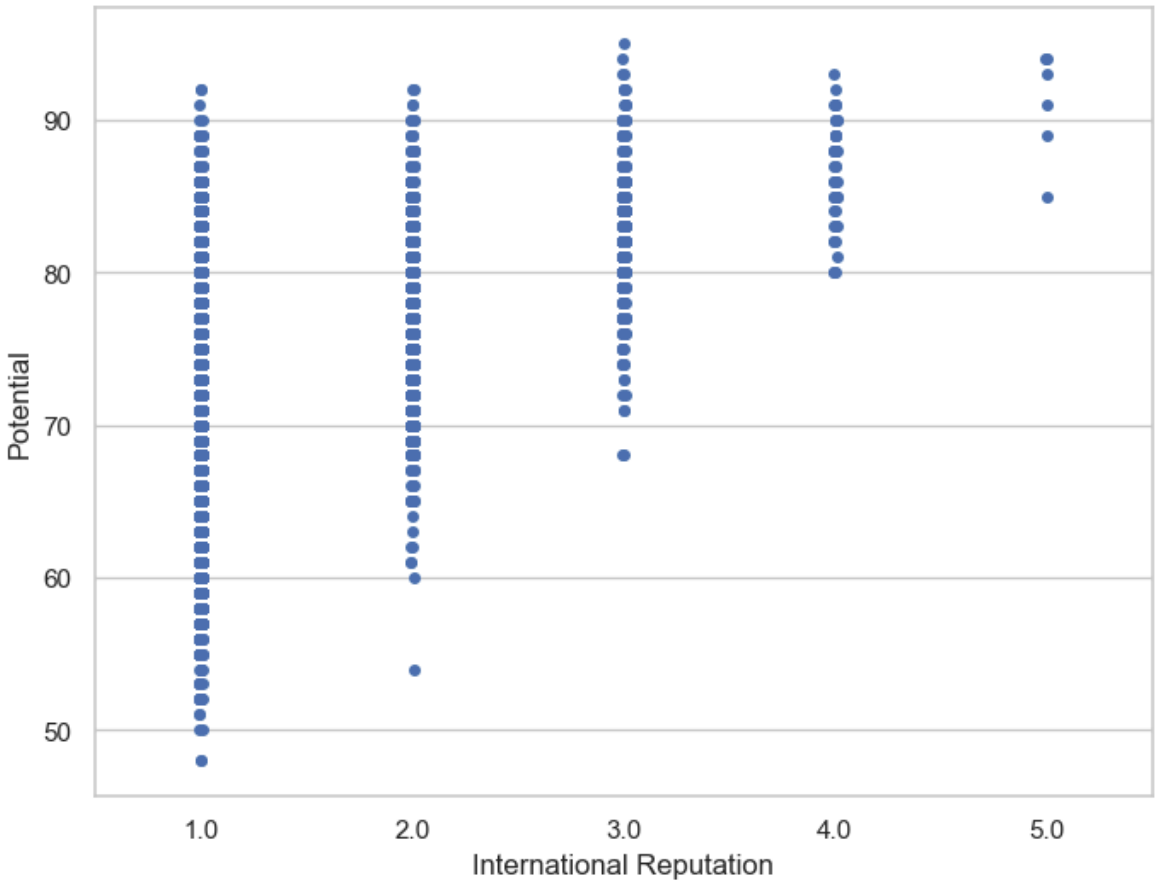
In [25]:
```
fifa19['International Reputation'].value_counts()
```

Out[25]:
```
International Reputation
1.0    16532
2.0     1261
3.0      309
4.0       51
5.0        6
Name: count, dtype: int64
```

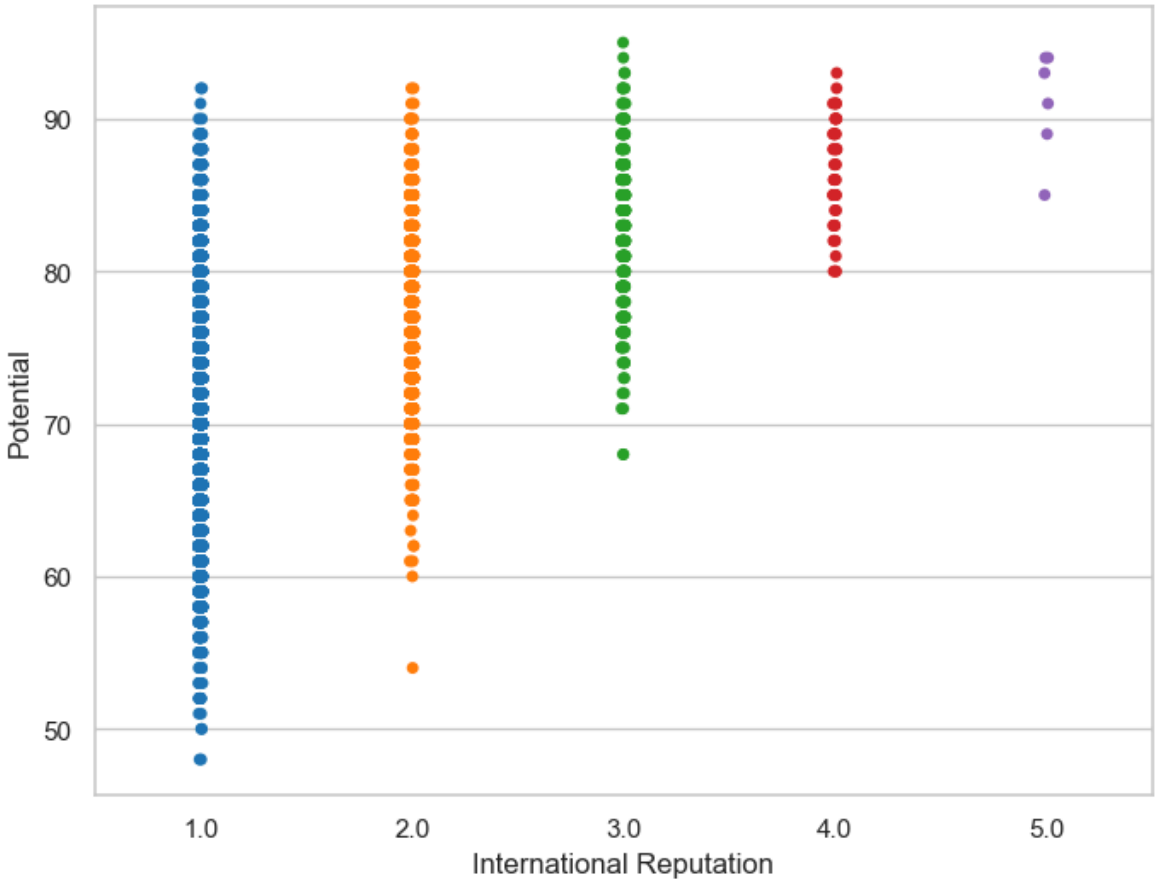
In [26]:
```
f,ax=plt.subplots(figsize=(8,6))
x=sns.stripplot(x="International Reputation",y="Potential",data=fifa19,palette="
plt.show()
```



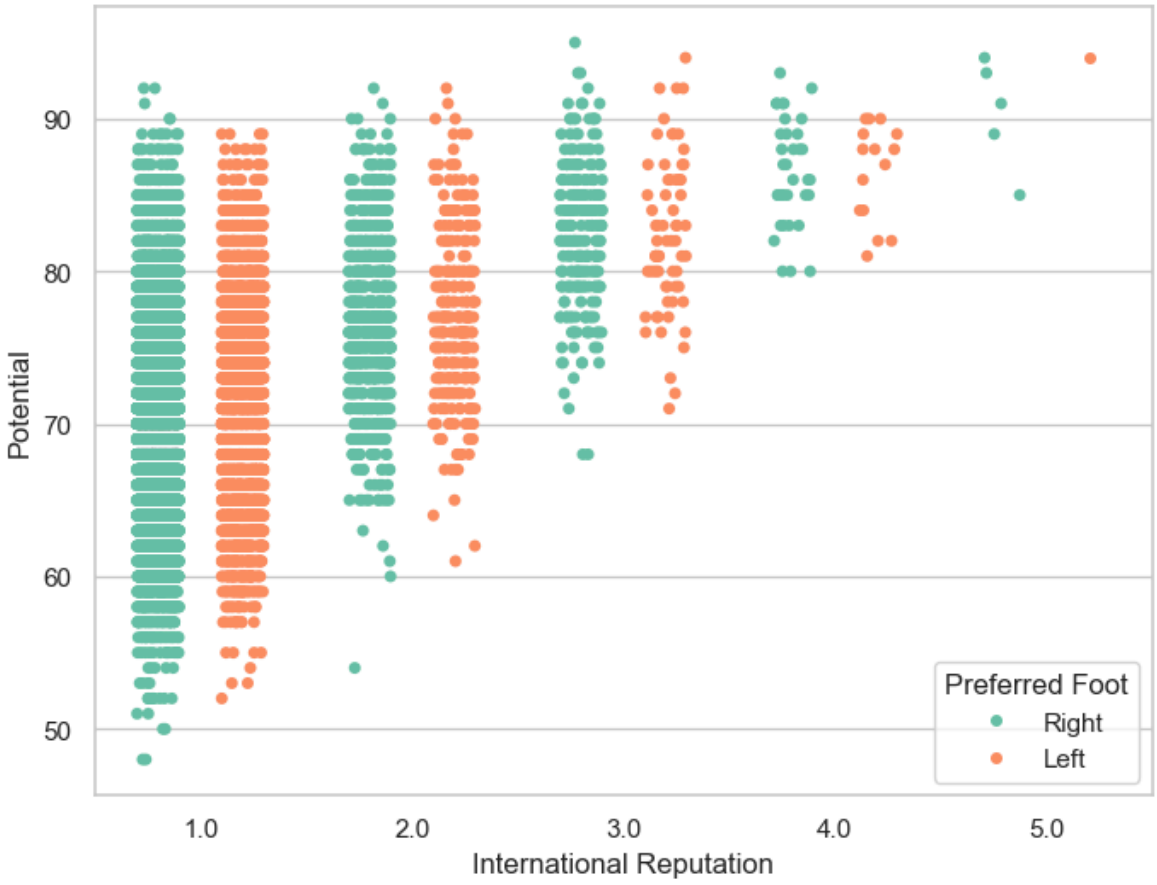
In [27]:
```
f,ax=plt.subplots(figsize=(8,6))
sns.stripplot(x="International Reputation",y="Potential",data=fifa19,jitter=0.01
plt.show()
```



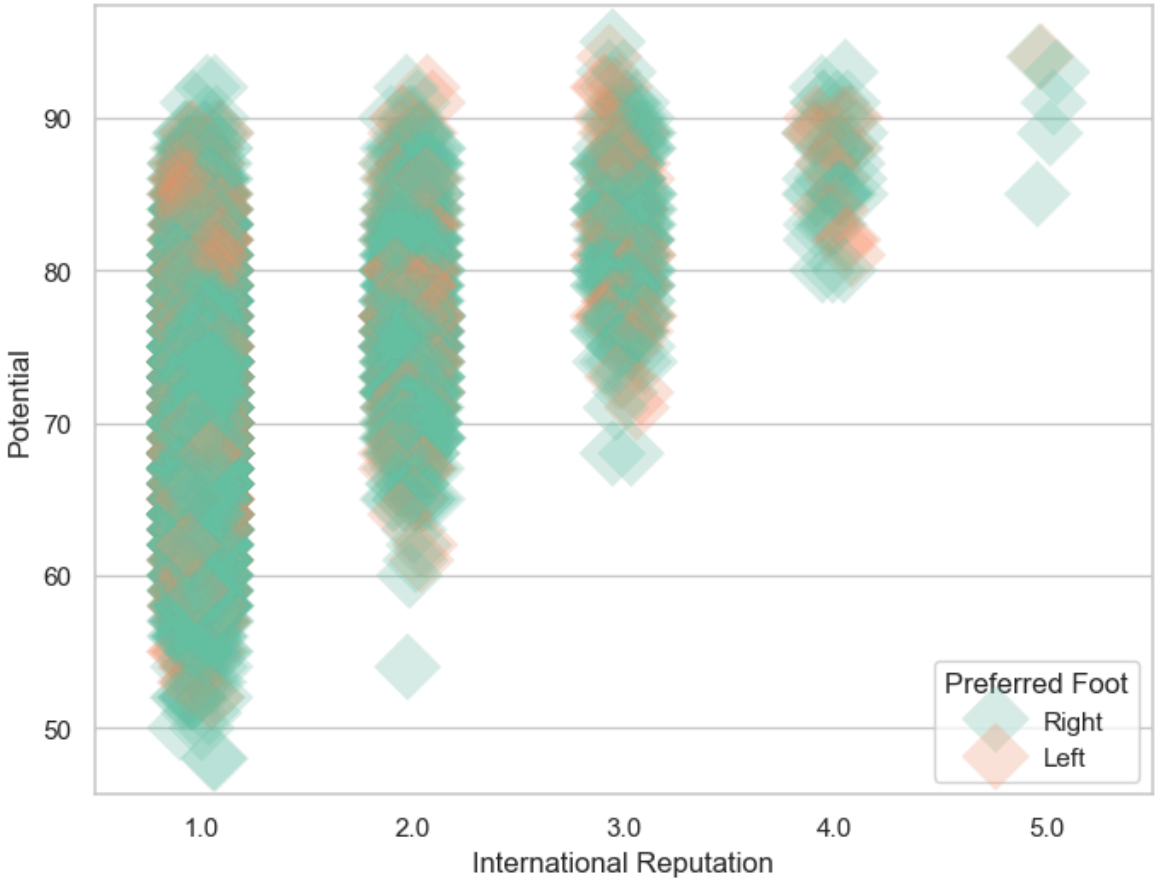
In [28]:
```
f,ax=plt.subplots(figsize=(8,6))
sns.stripplot(x="International Reputation",y="Potential",data=fifa19,jitter=0.01
plt.show()
```



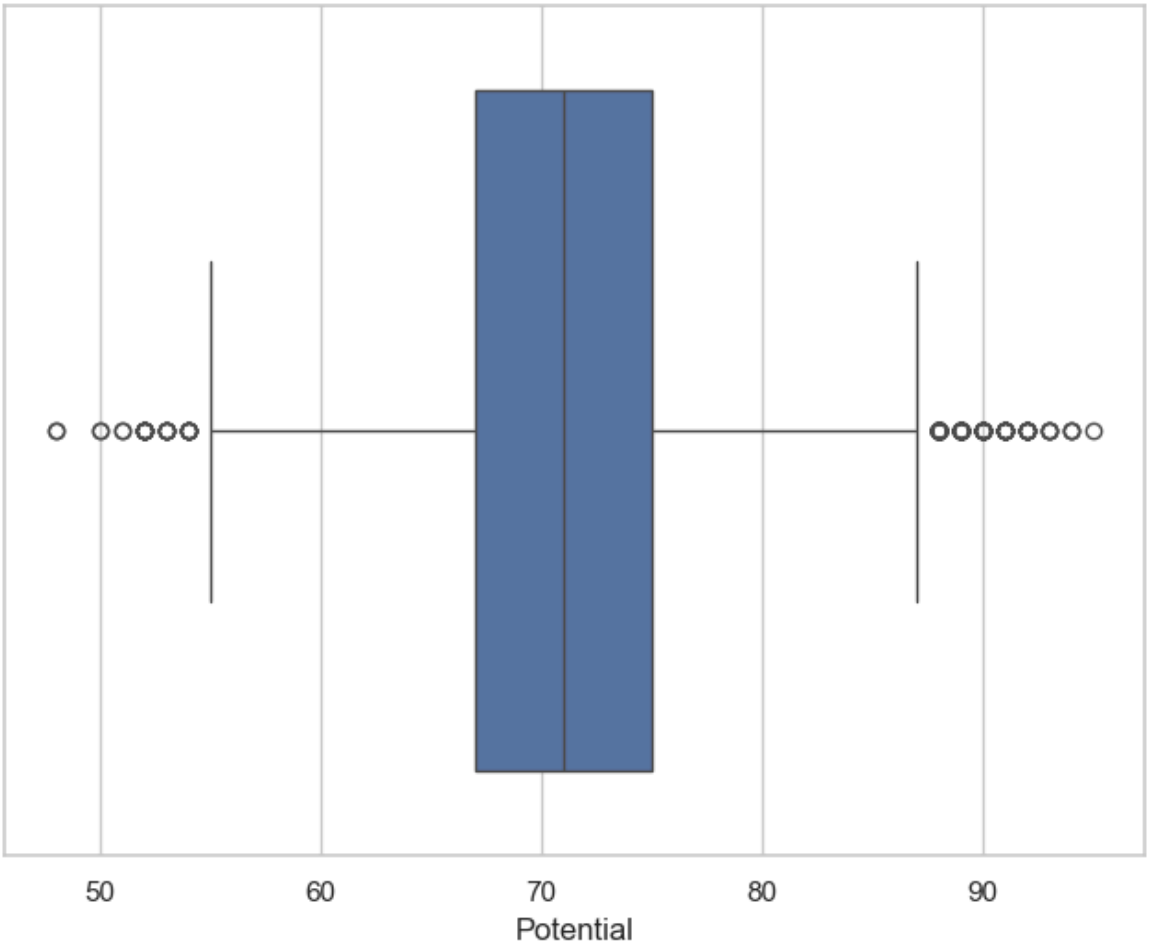
In [29]:
```
f,ax=plt.subplots(figsize=(8,6))
sns.stripplot(x="International Reputation",y="Potential",hue="Preferred Foot",
              data=fifa19,jitter=0.2,palette="Set2",dodge=True)
plt.show()
```



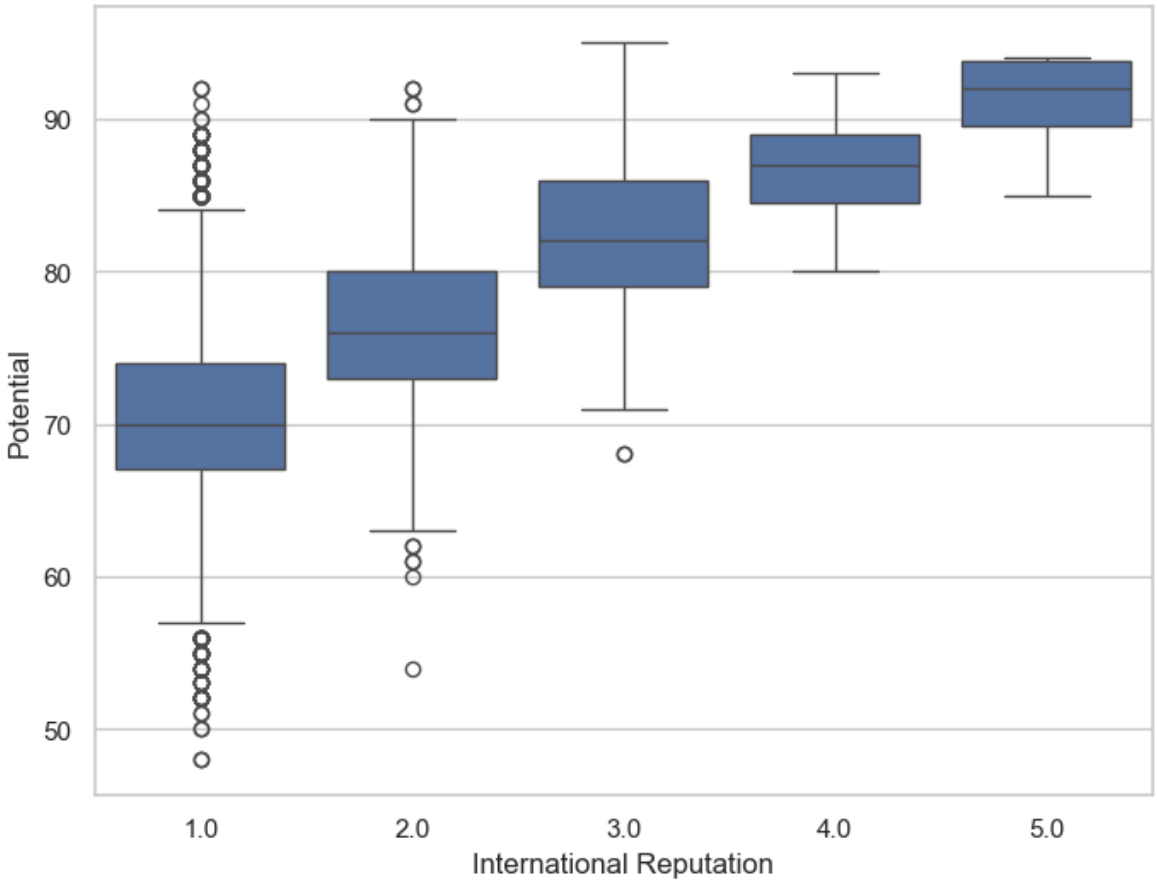
In [30]:
```
f,ax=plt.subplots(figsize=(8,6))
sns.stripplot(x="International Reputation",y="Potential",hue="Preferred Foot",
              data=fifa19,palette="Set2",size=20,marker="D",edgecolor="gray",alp
plt.show()
```



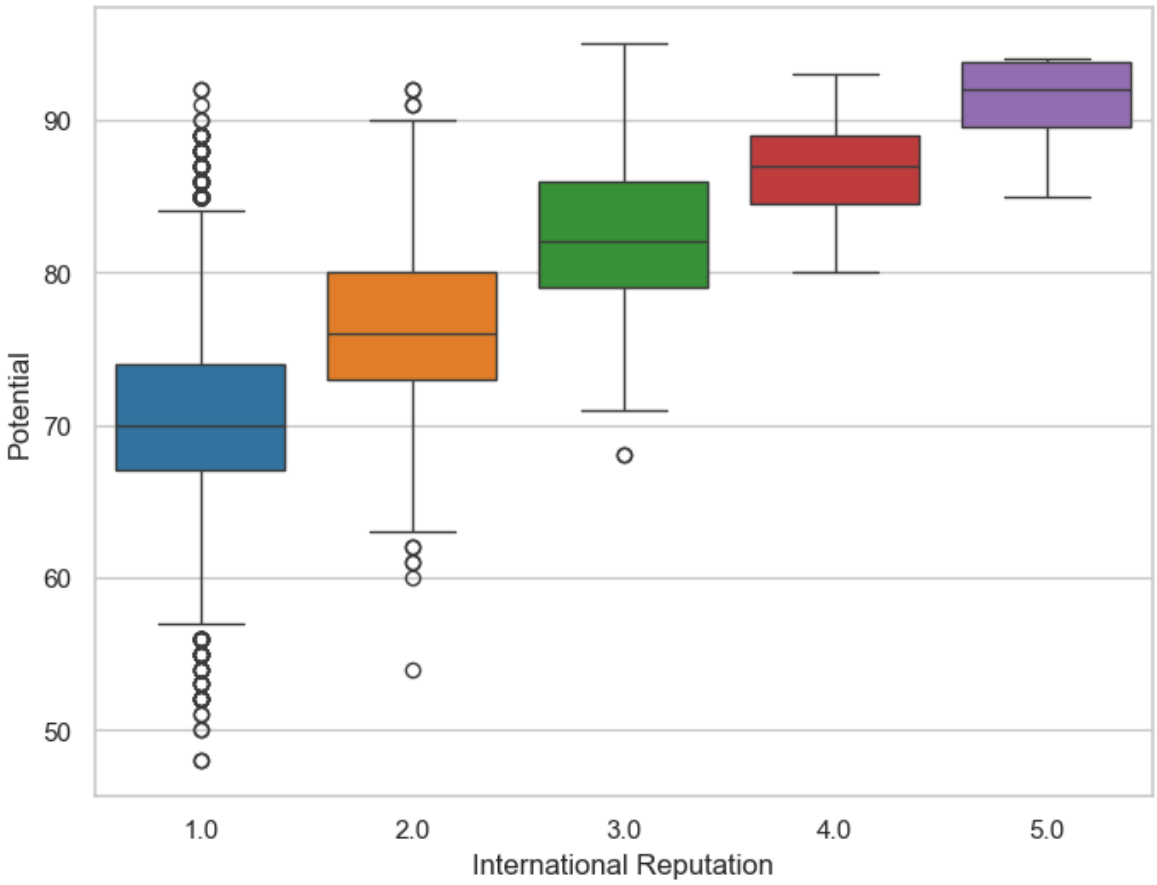
In [31]:
```
f,ax=plt.subplots(figsize=(8,6))
sns.boxplot(x=fifa19['Potential'])
plt.show()
```



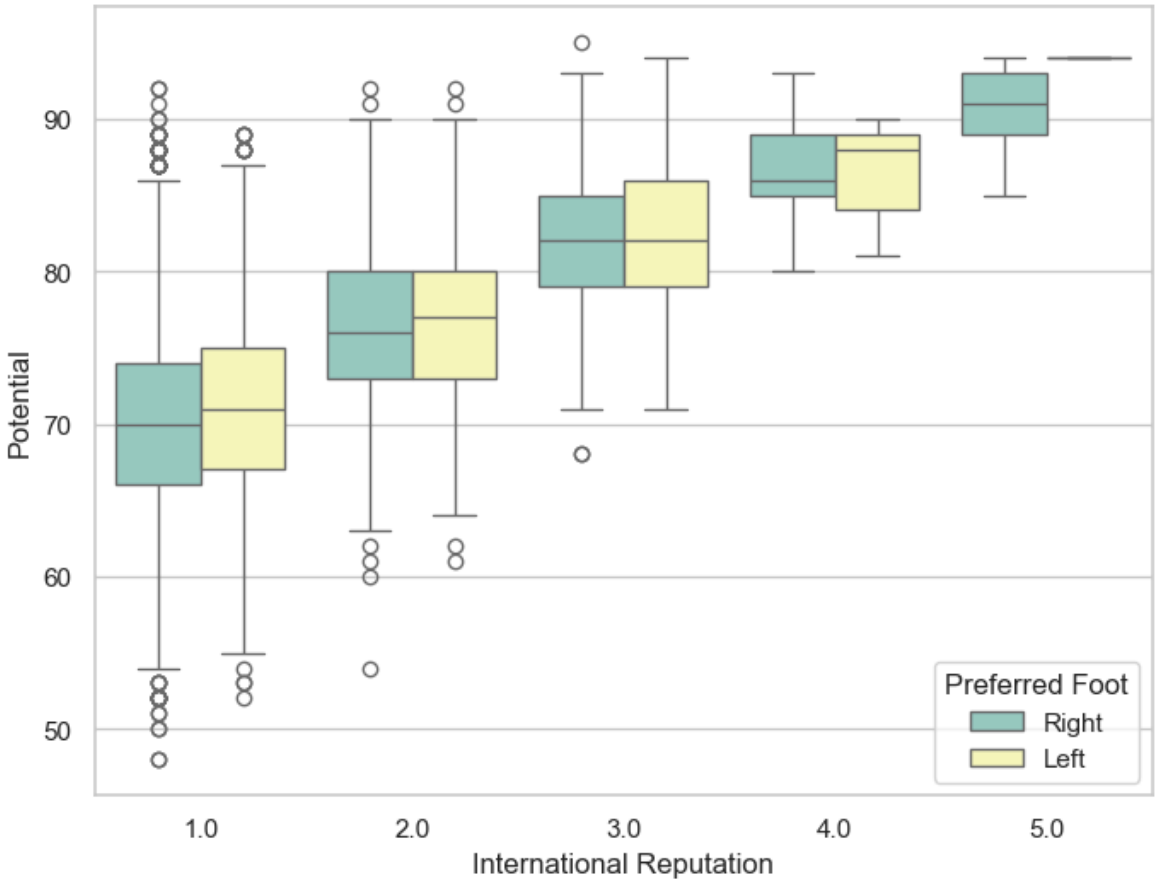
In [32]:
```
f,ax=plt.subplots(figsize=(8,6))
sns.boxplot(x="International Reputation",y="Potential",data=fifa19)
plt.show()
```



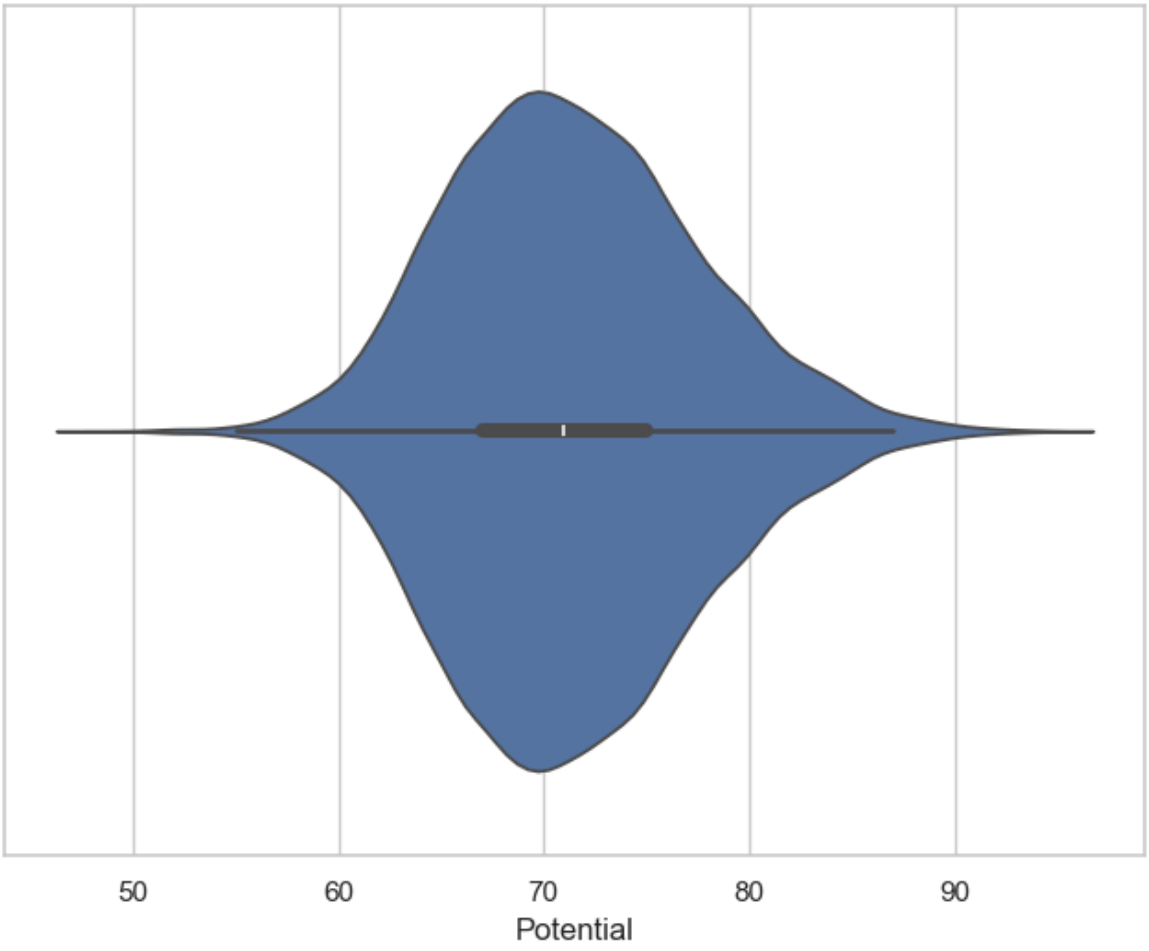
In [33]:
```
f,ax=plt.subplots(figsize=(8,6))
sns.boxplot(x="International Reputation",y="Potential",data=fifa19,palette="tab1
plt.show()
```



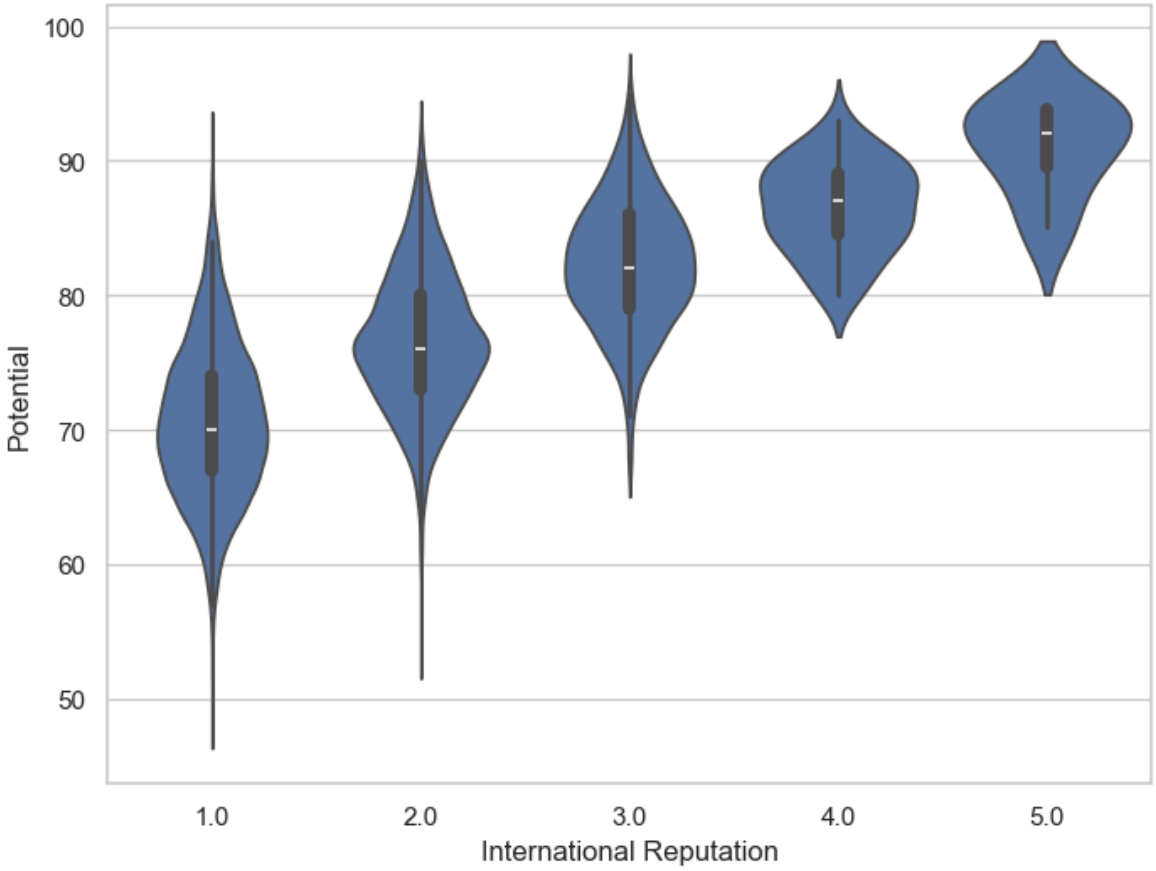
In [34]:
```
f,ax=plt.subplots(figsize=(8,6))
sns.boxplot(x="International Reputation",y="Potential",hue="Preferred Foot",data
plt.show()
```



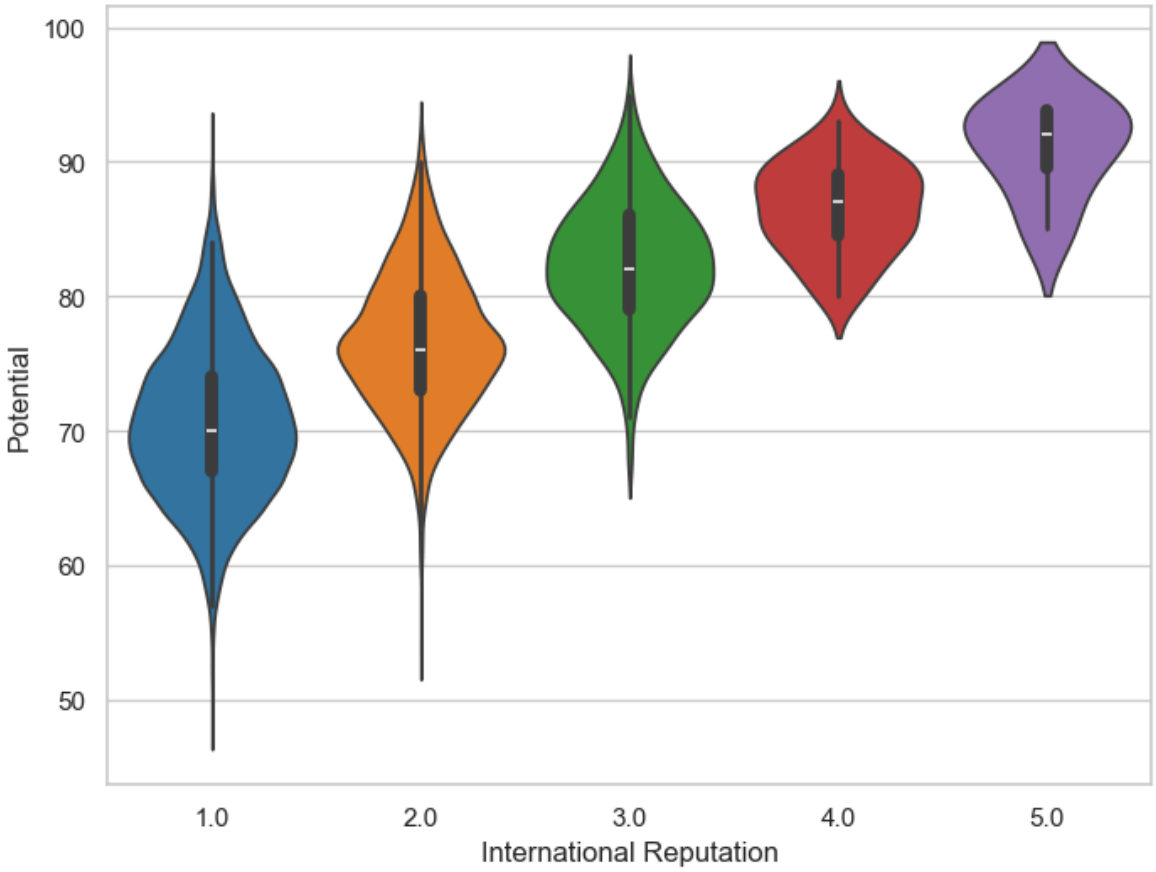
In [35]:
```
f,ax=plt.subplots(figsize=(8,6))
sns.violinplot(x=fifa19["Potential"])
plt.show()
```



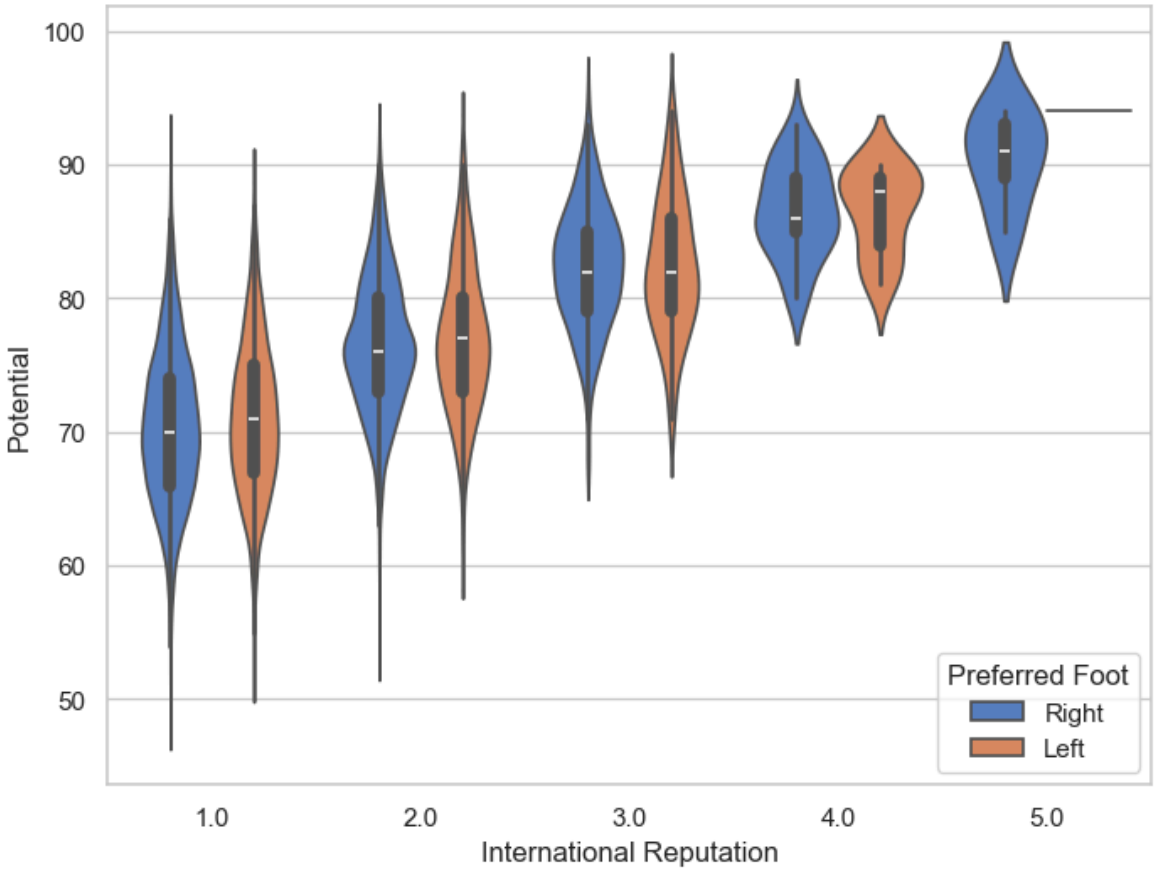
In [36]:
```
f,ax=plt.subplots(figsize=(8,6))
sns.violinplot(x="International Reputation",y="Potential",data=fifa19)
plt.show()
```



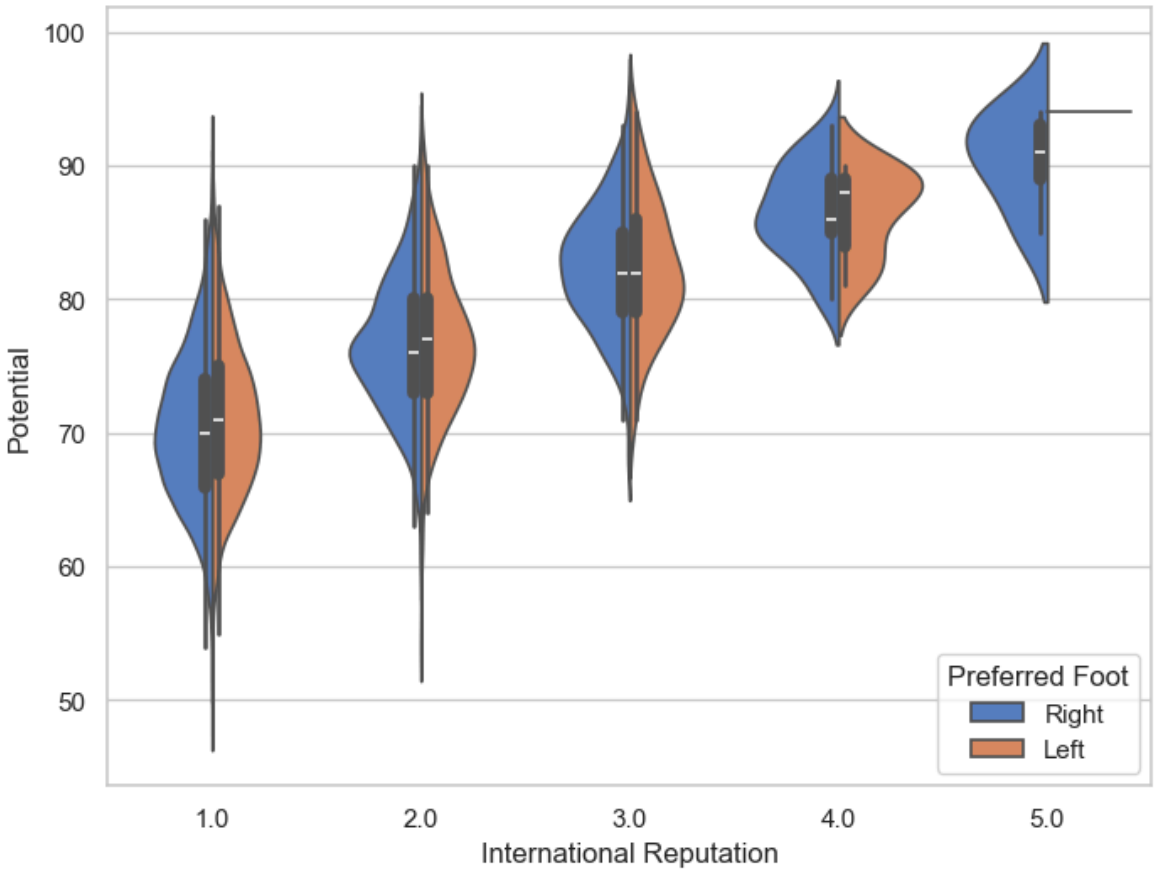
In [37]:
```
f,ax=plt.subplots(figsize=(8,6))
sns.violinplot(x="International Reputation",y="Potential",data=fifa19,palette="t
plt.show()
```



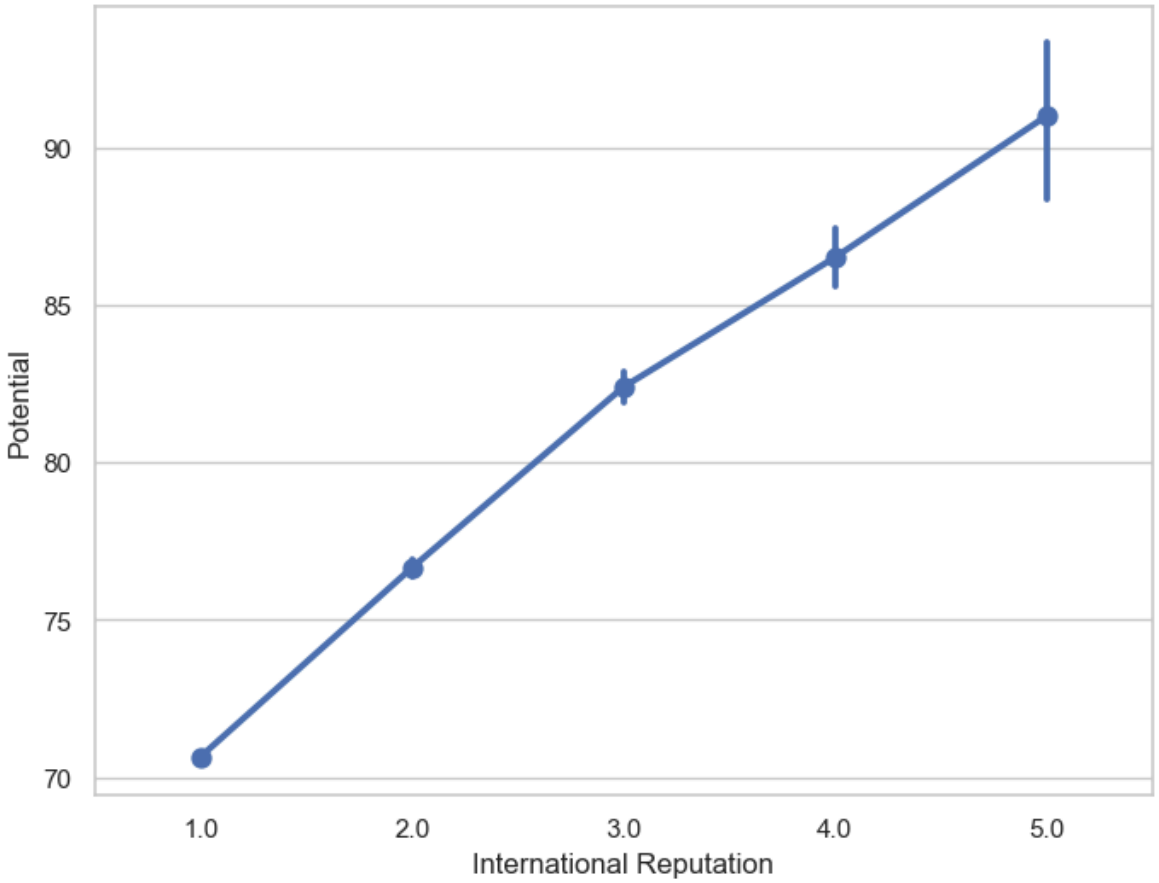
In [38]:
```
f,ax=plt.subplots(figsize=(8,6))
sns.violinplot(x="International Reputation",y="Potential",hue='Preferred Foot',d
plt.show()
```



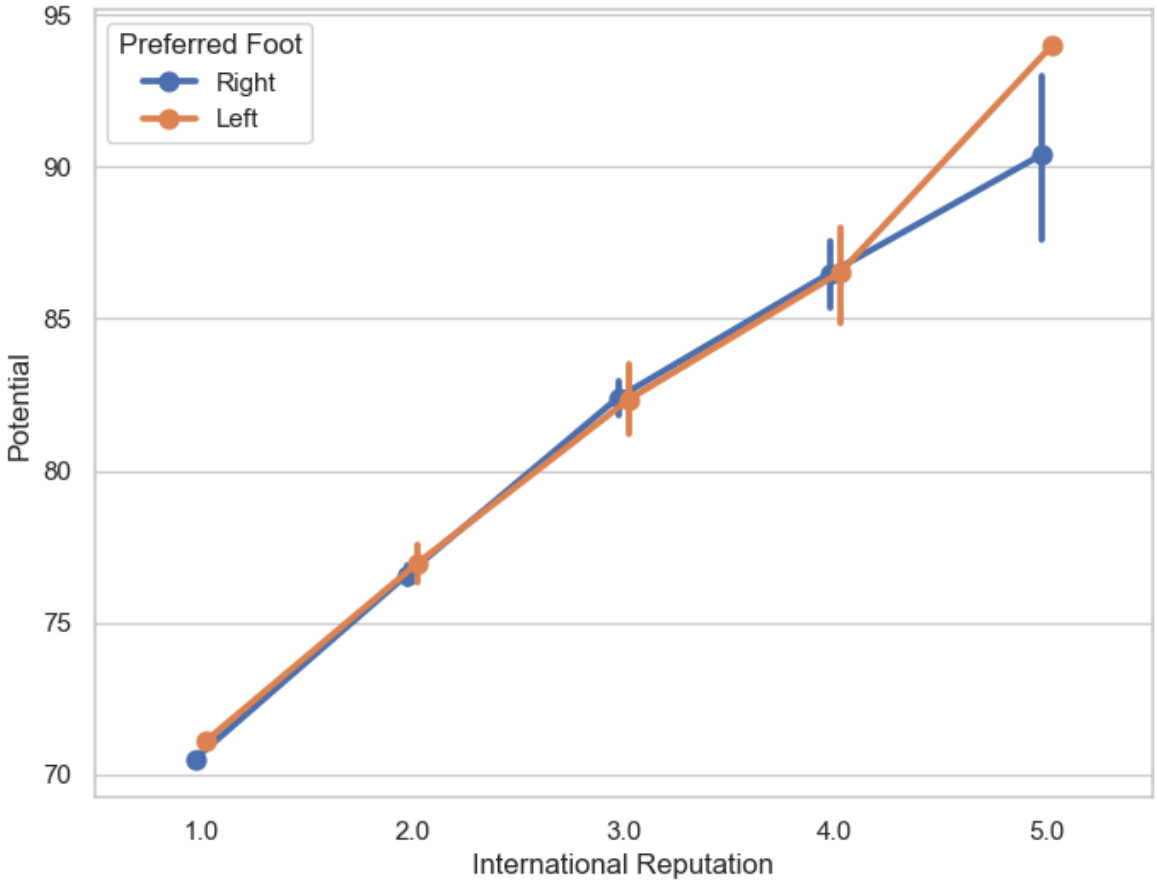
In [39]:
```
f,ax=plt.subplots(figsize=(8,6))
sns.violinplot(x="International Reputation",y="Potential",hue="Preferred Foot",
               data=fifa19,palette="muted",split=True)
plt.show()
```



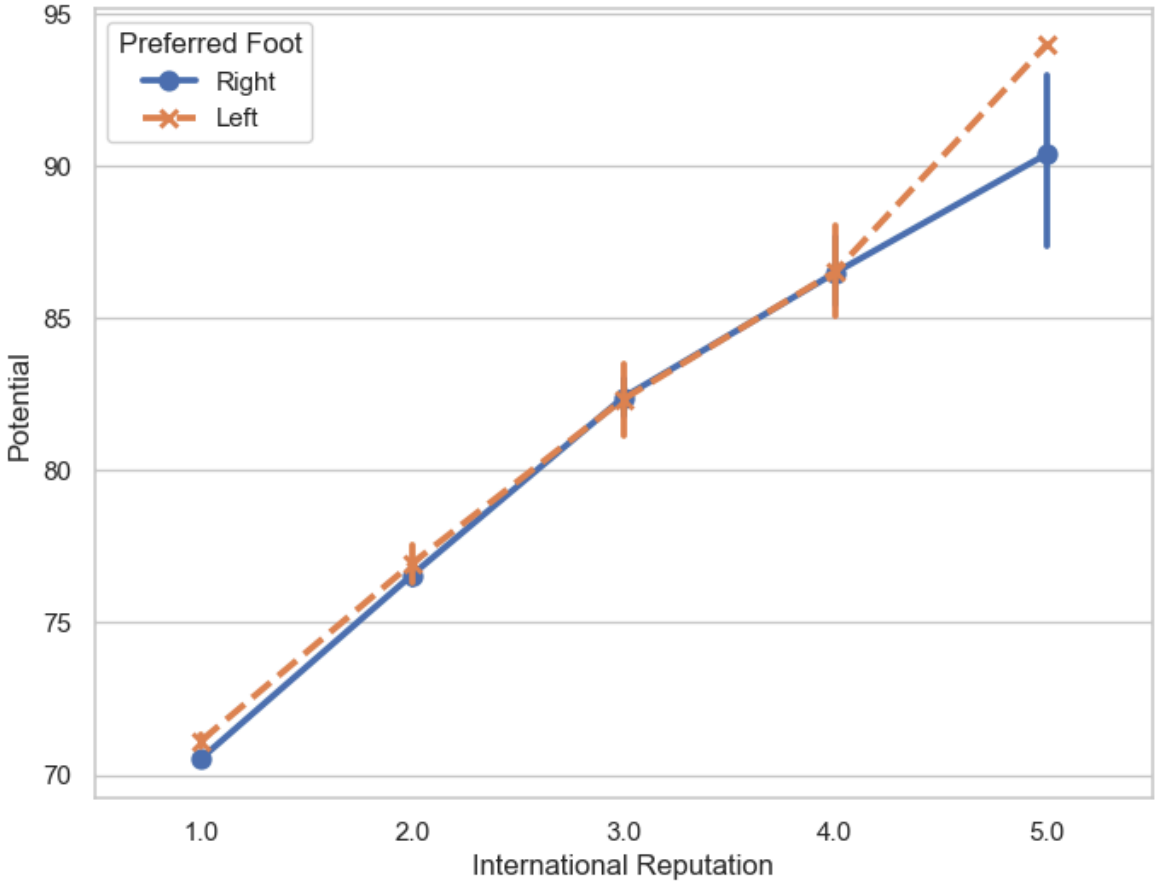
In [40]:
```
f,ax=plt.subplots(figsize=(8,6))
sns.pointplot(x="International Reputation",y="Potential",data=fifa19)
plt.show()
```



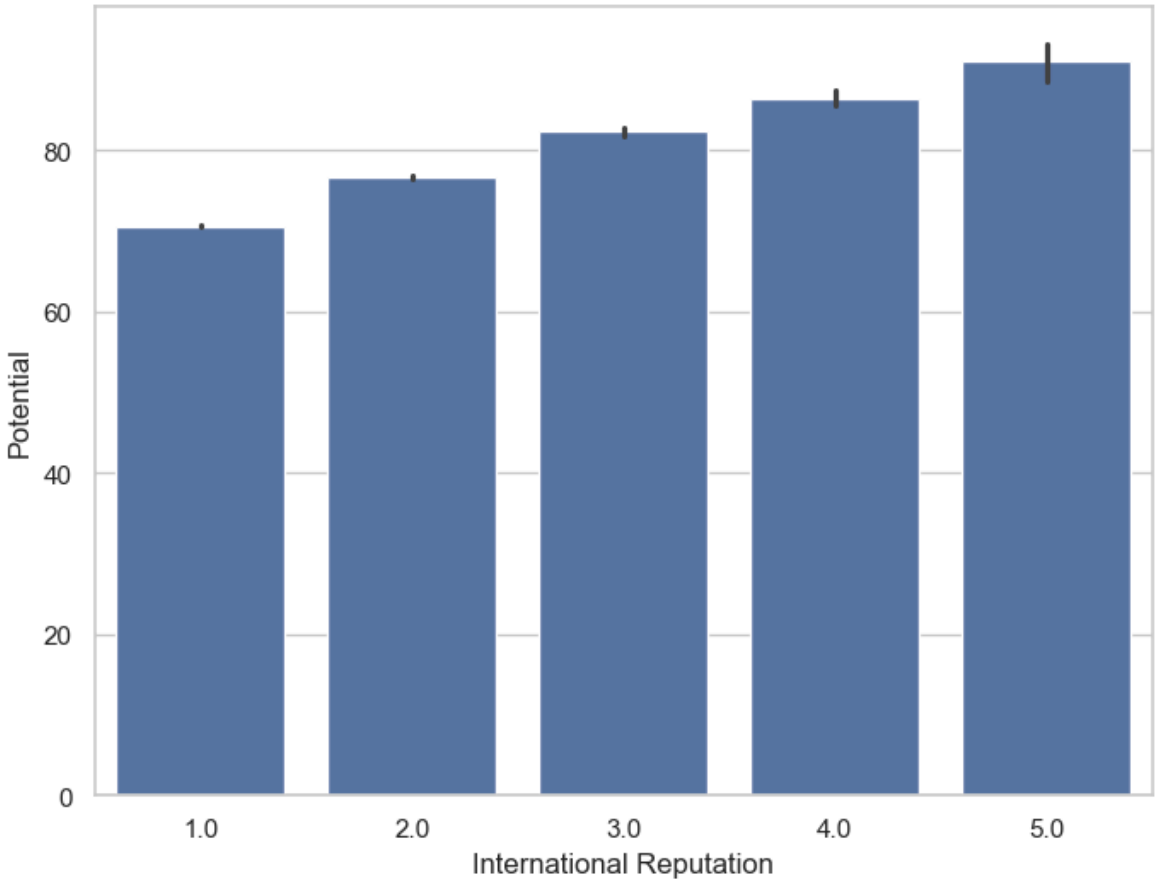
In [41]:
```
f,ax=plt.subplots(figsize=(8,6))
sns.pointplot(x="International Reputation",y="Potential",hue="Preferred Foot",da
plt.show()
```



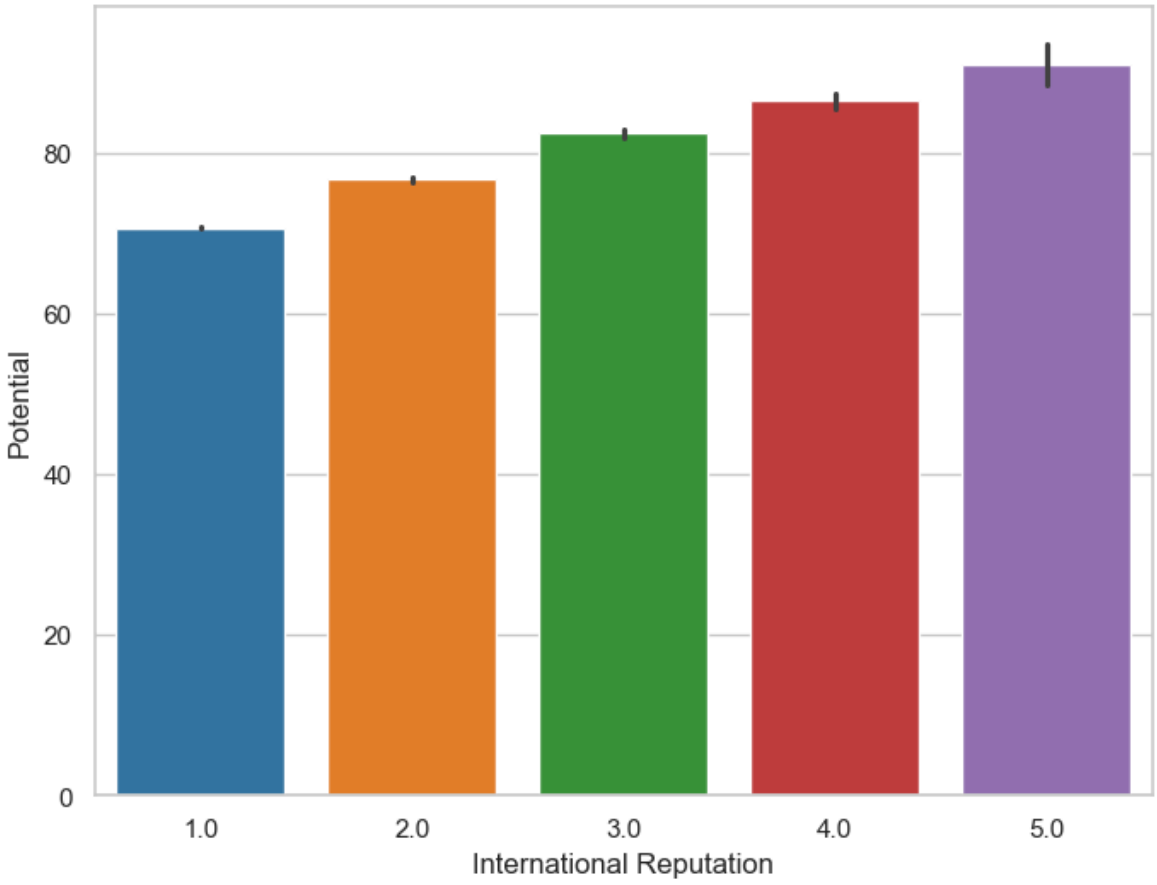
In [42]:
```
f,ax=plt.subplots(figsize=(8,6))
sns.pointplot(x="International Reputation",y="Potential",hue="Preferred Foot",
              data=fifa19,markers=["o","x"],linestyles=["-","--"])
plt.show()
```



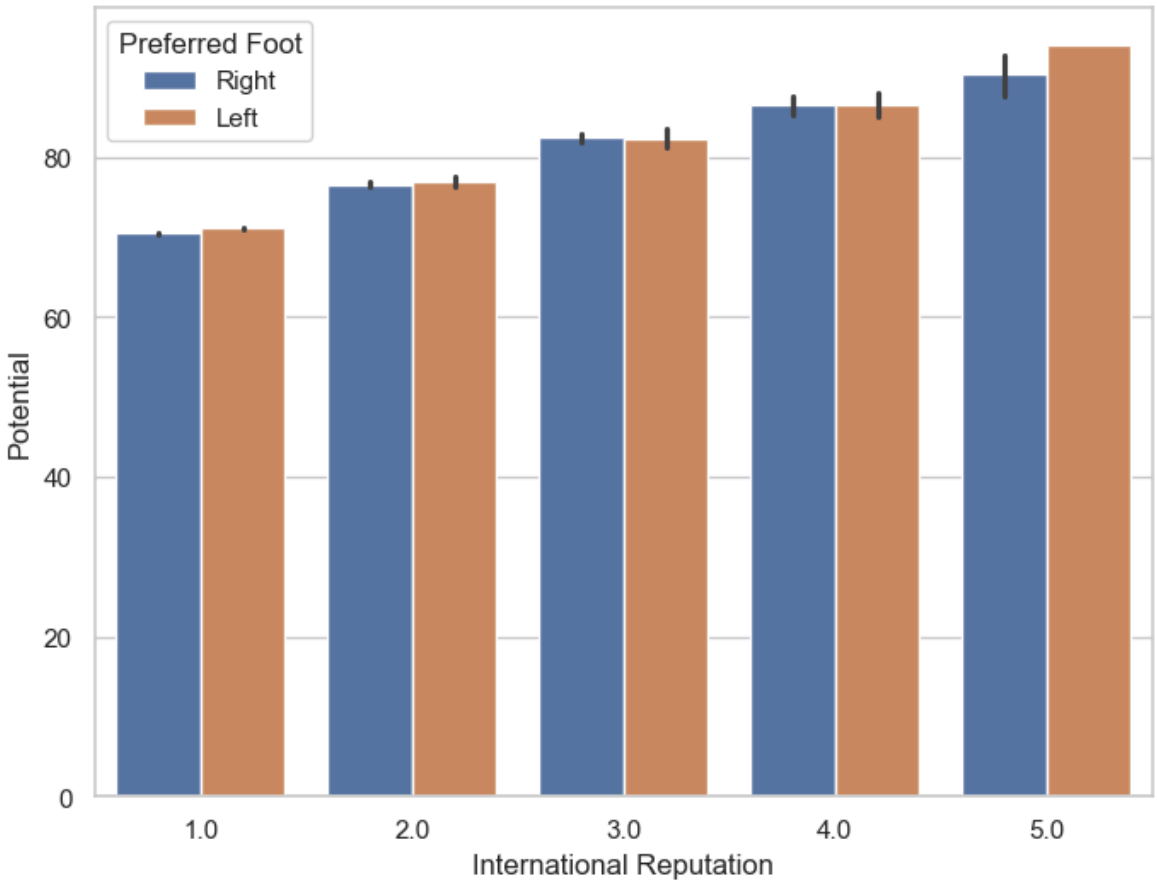
In [43]:
```
f,ax=plt.subplots(figsize=(8,6))
sns.barplot(x="International Reputation",y="Potential",data=fifa19)
plt.show()
```



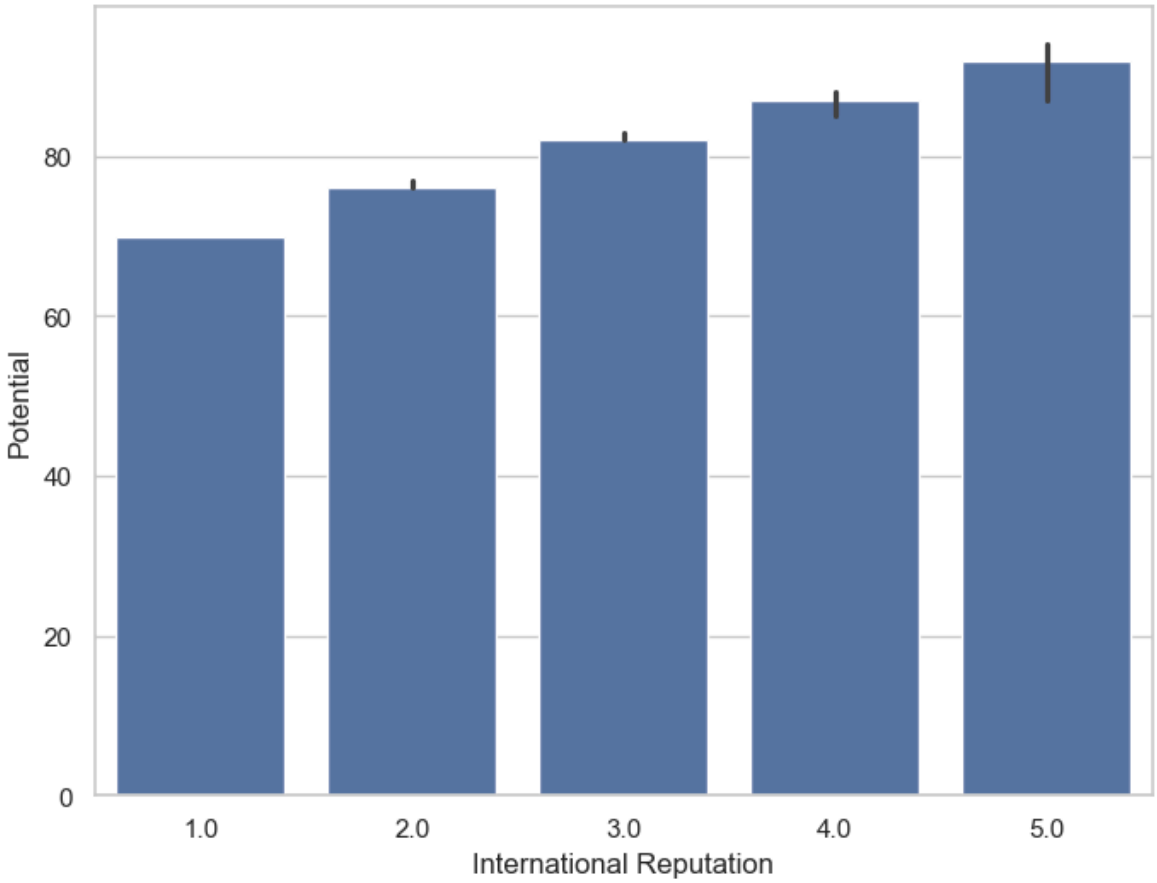
In [44]:
```
f,ax=plt.subplots(figsize=(8,6))
sns.barplot(x="International Reputation",y="Potential",data=fifa19,palette="tab1
plt.show()
```



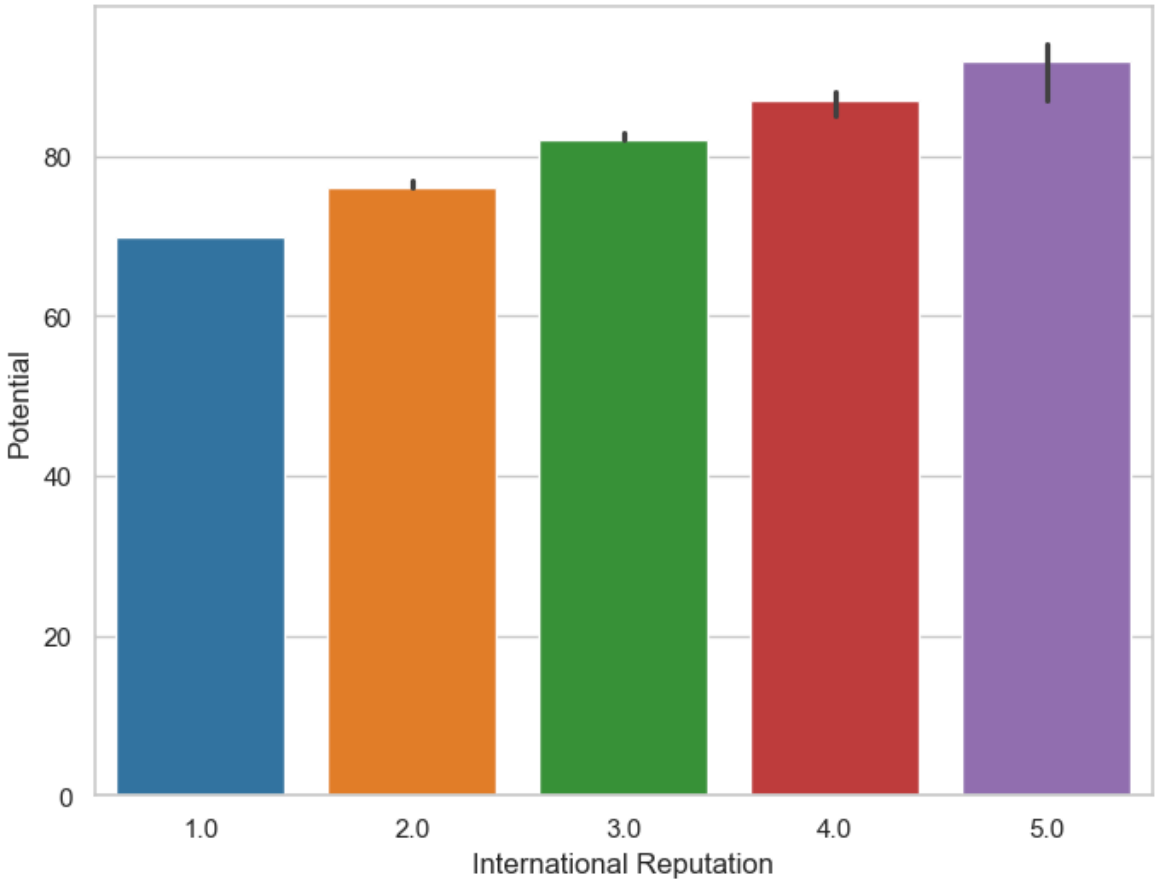
In [45]:
```
f,ax=plt.subplots(figsize=(8,6))
sns.barplot(x="International Reputation",y="Potential",hue="Preferred Foot",data
plt.show()
```



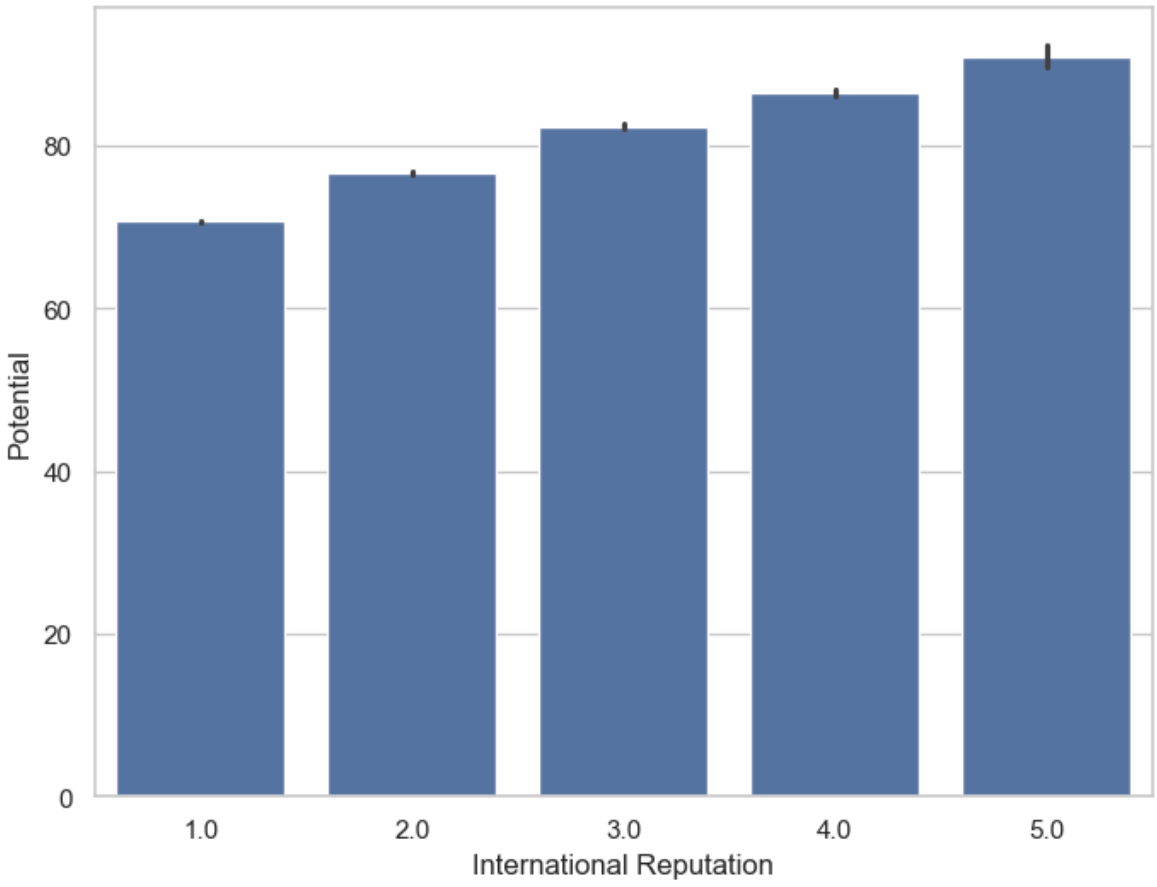
In [46]:
```
from numpy import median
f,ax=plt.subplots(figsize=(8,6))
sns.barplot(x="International Reputation",y="Potential",data=fifa19,estimator=med
plt.show()
```



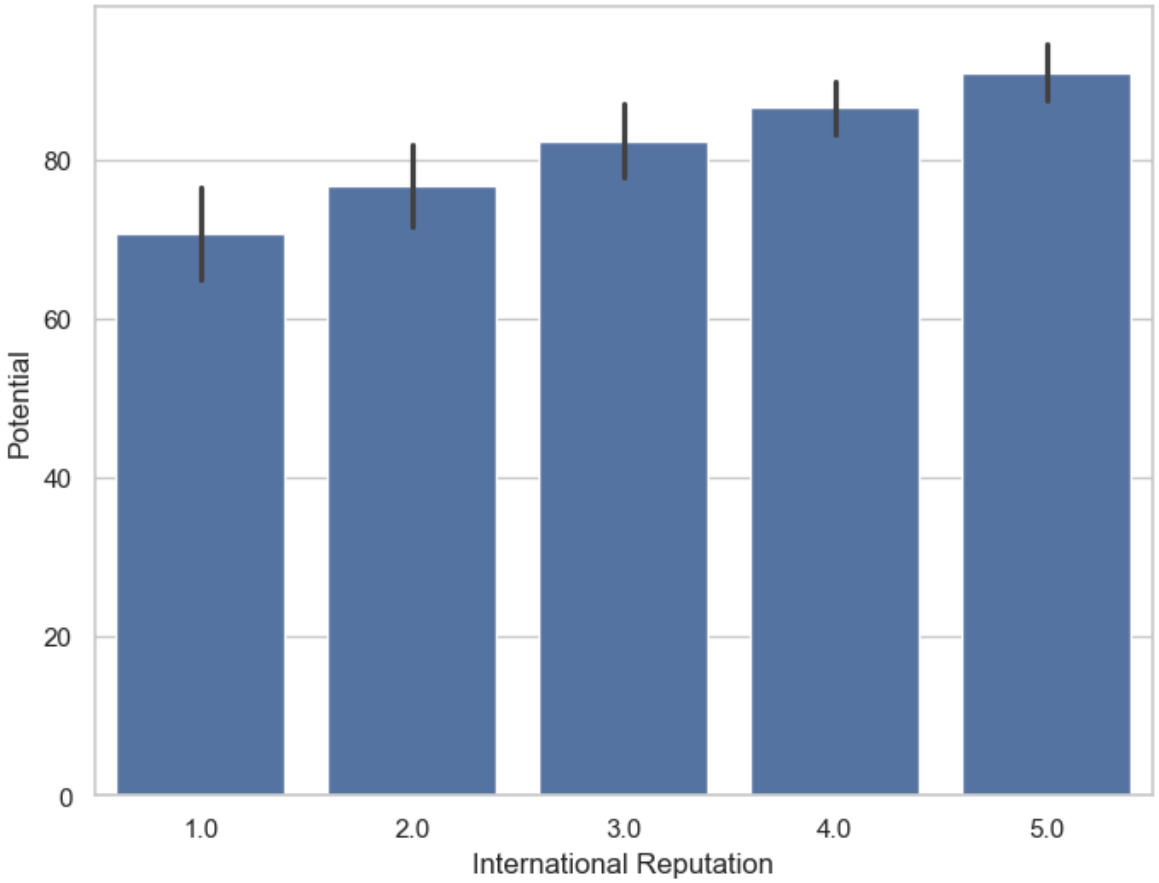
In [47]:
```
from numpy import median
f,ax=plt.subplots(figsize=(8,6))
sns.barplot(x="International Reputation",y="Potential",data=fifa19,estimator=med
plt.show()
```



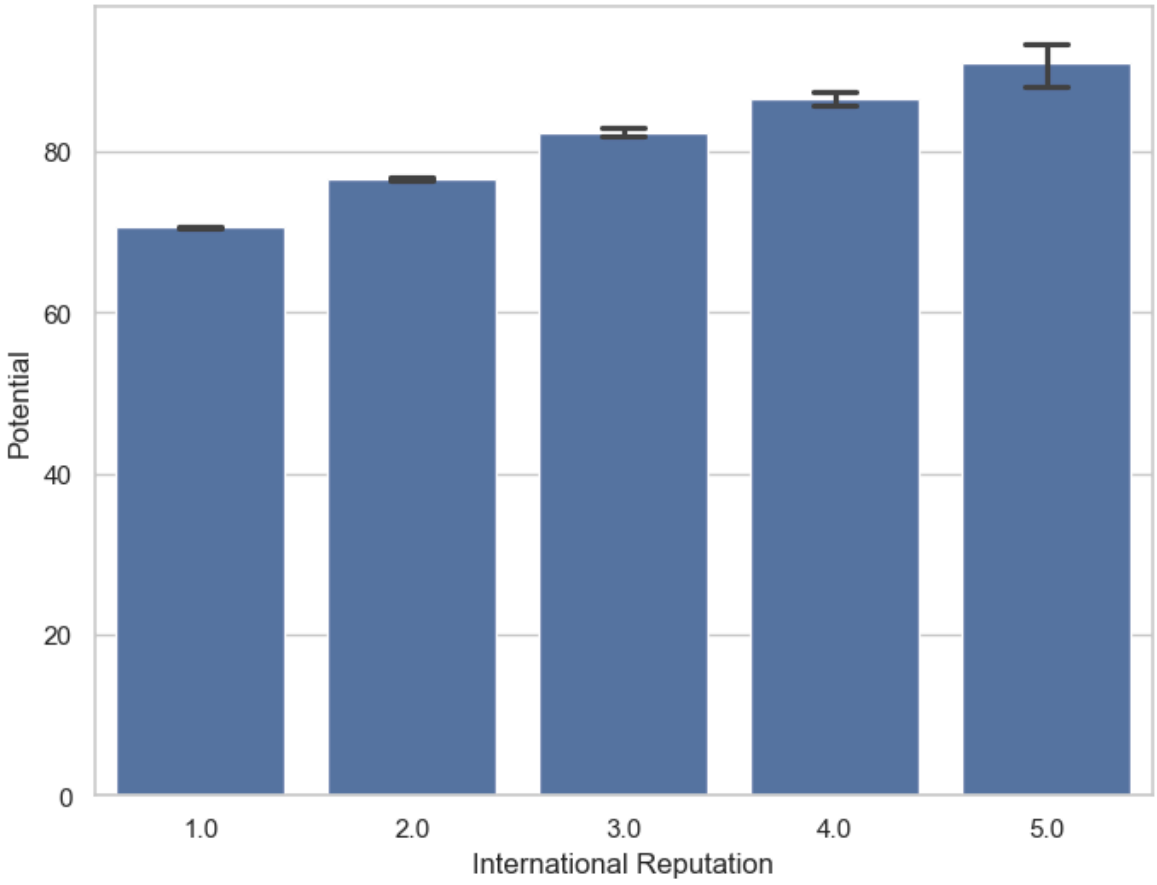
In [48]:
```
f,ax=plt.subplots(figsize=(8,6))
sns.barplot(x="International Reputation",y="Potential",data=fifa19,ci=68)
plt.show()
```



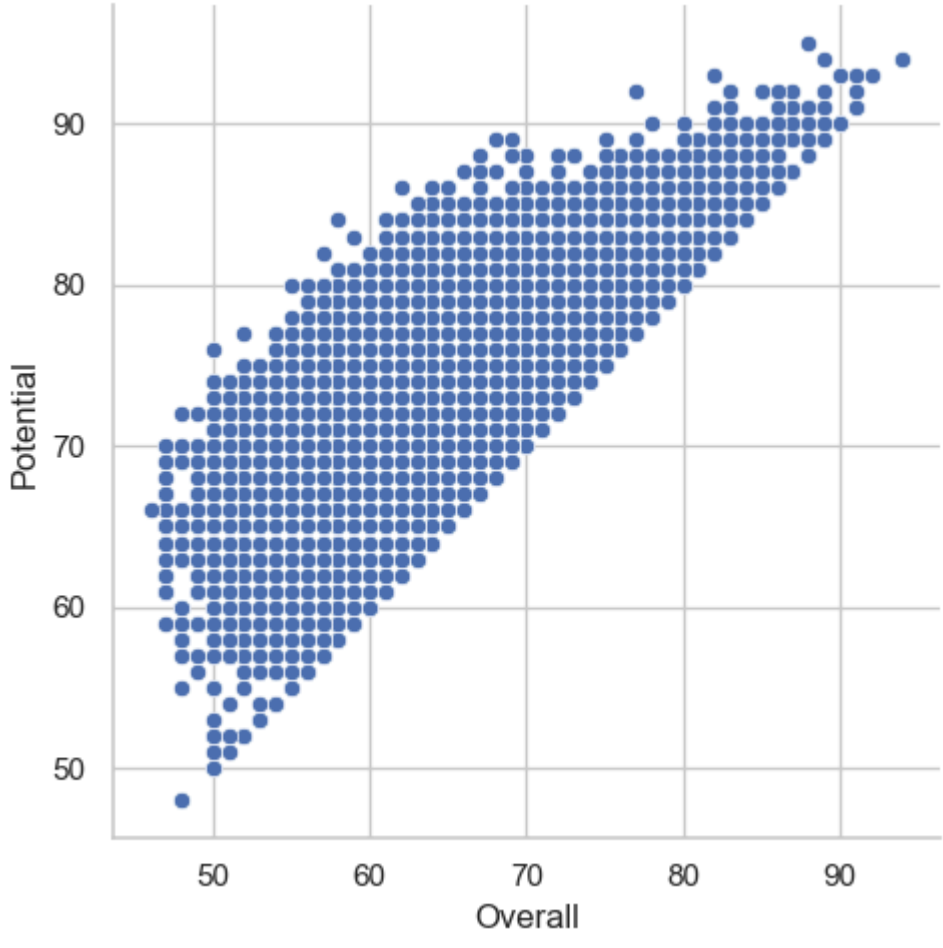
In [49]:
```
f,ax=plt.subplots(figsize=(8,6))
sns.barplot(x="International Reputation",y="Potential",data=fifa19,ci="sd")
plt.show()
```



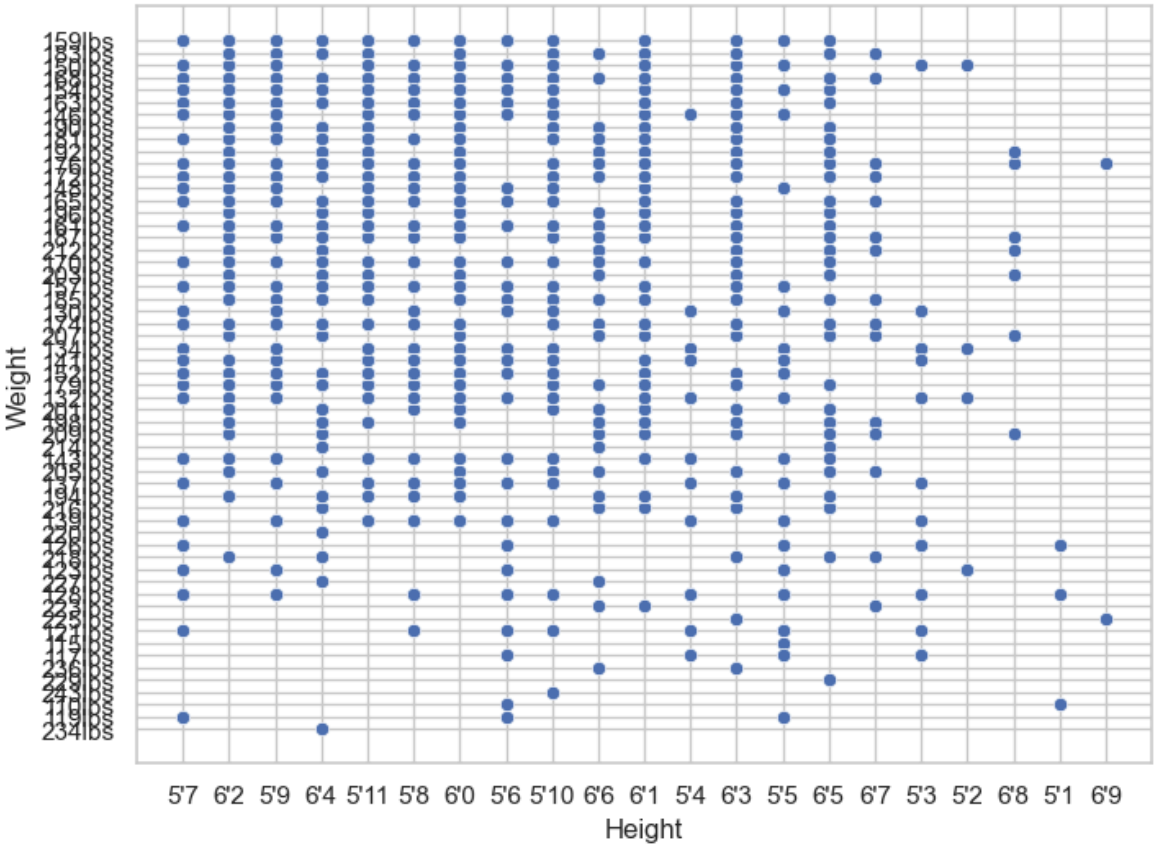
In [50]:
```
f,ax=plt.subplots(figsize=(8,6))
sns.barplot(x="International Reputation",y="Potential",data=fifa19,capsize=0.2)
plt.show()
```



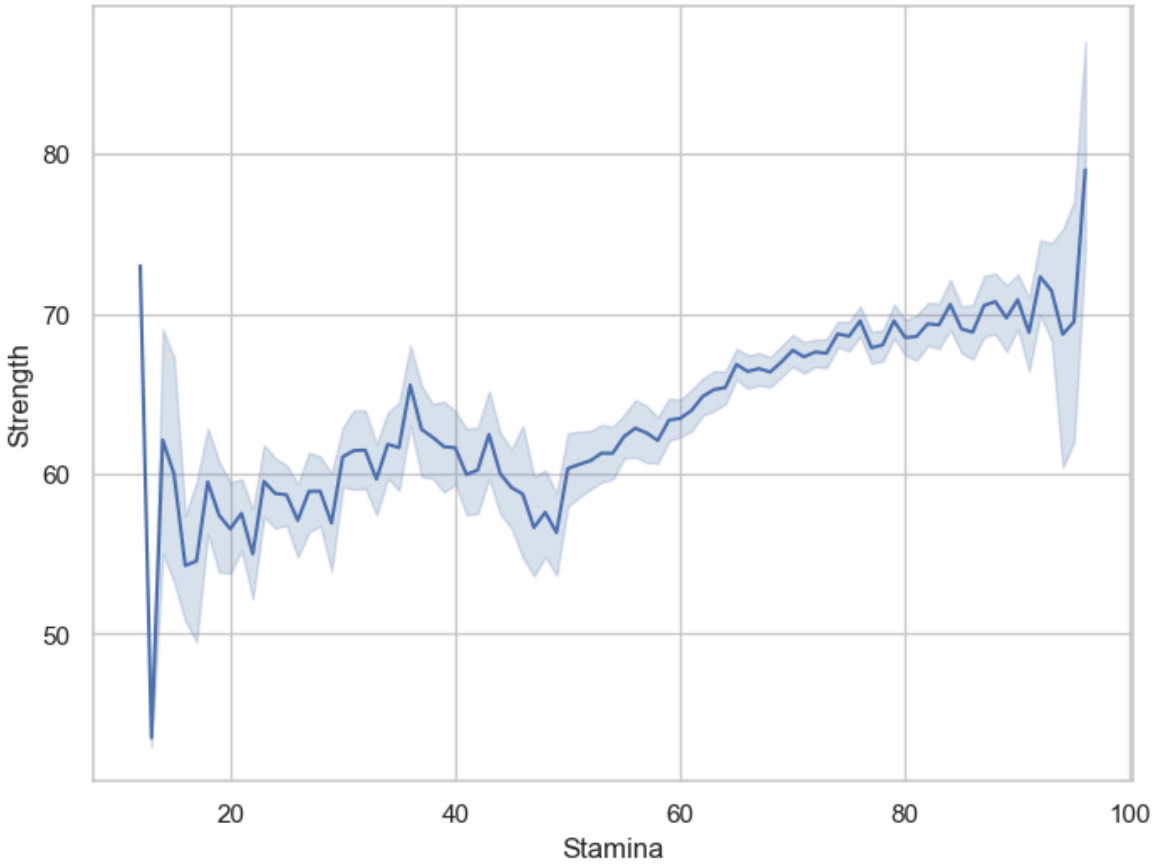
In [51]:
```
g=sns.relplot(x="Overall",y="Potential",data=fifa19)
plt.show()
```



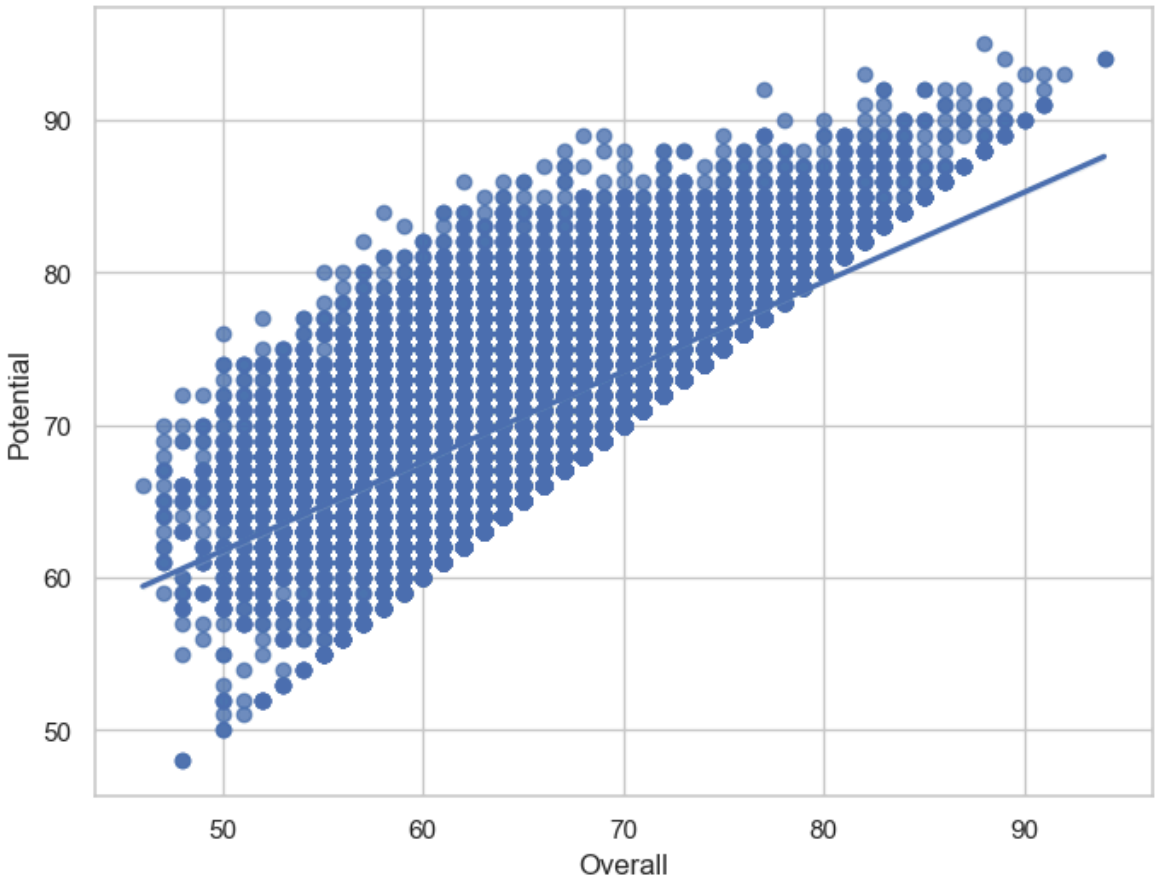
In [52]:
```
f,ax=plt.subplots(figsize=(8,6))
sns.scatterplot(x="Height",y="Weight",data=fifa19)
plt.show()
```



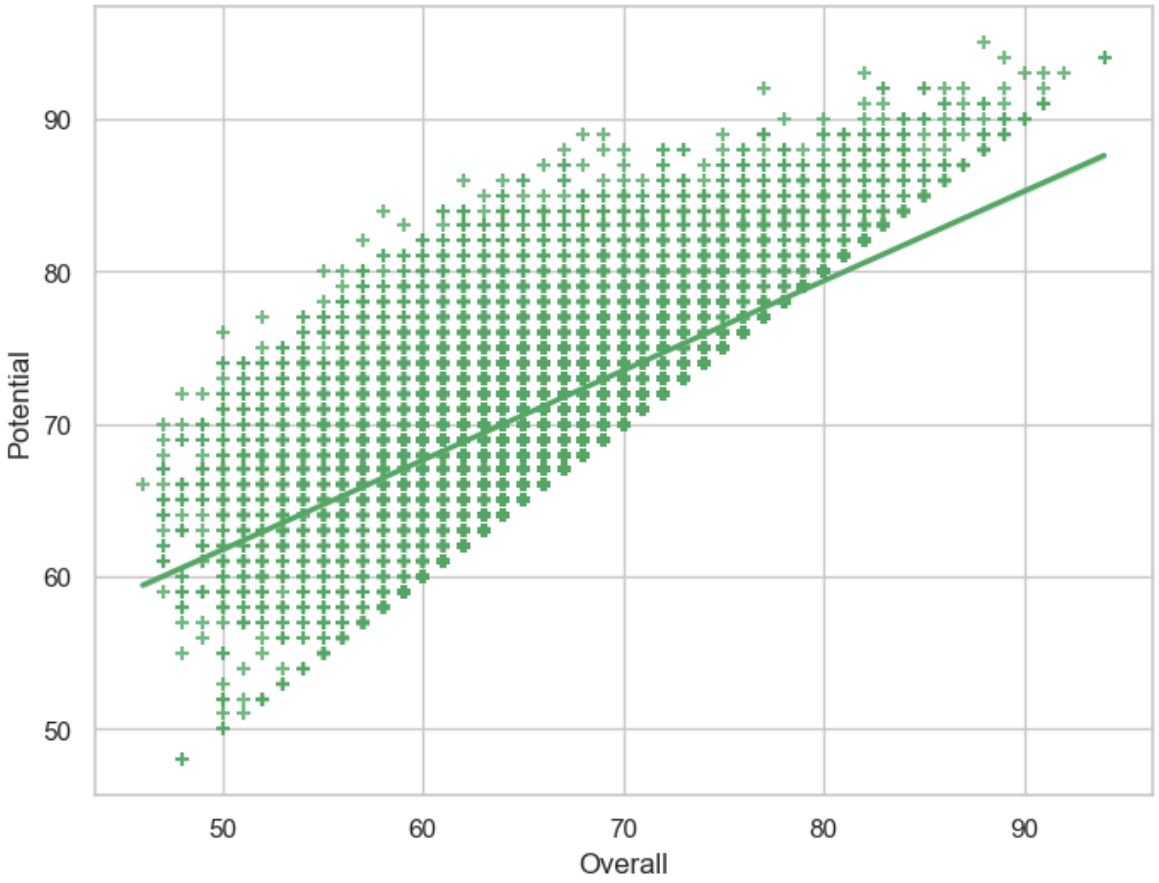
In [53]:
```
f,ax=plt.subplots(figsize=(8,6))
ax=sns.lineplot(x="Stamina",y="Strength",data=fifa19)
plt.show()
```



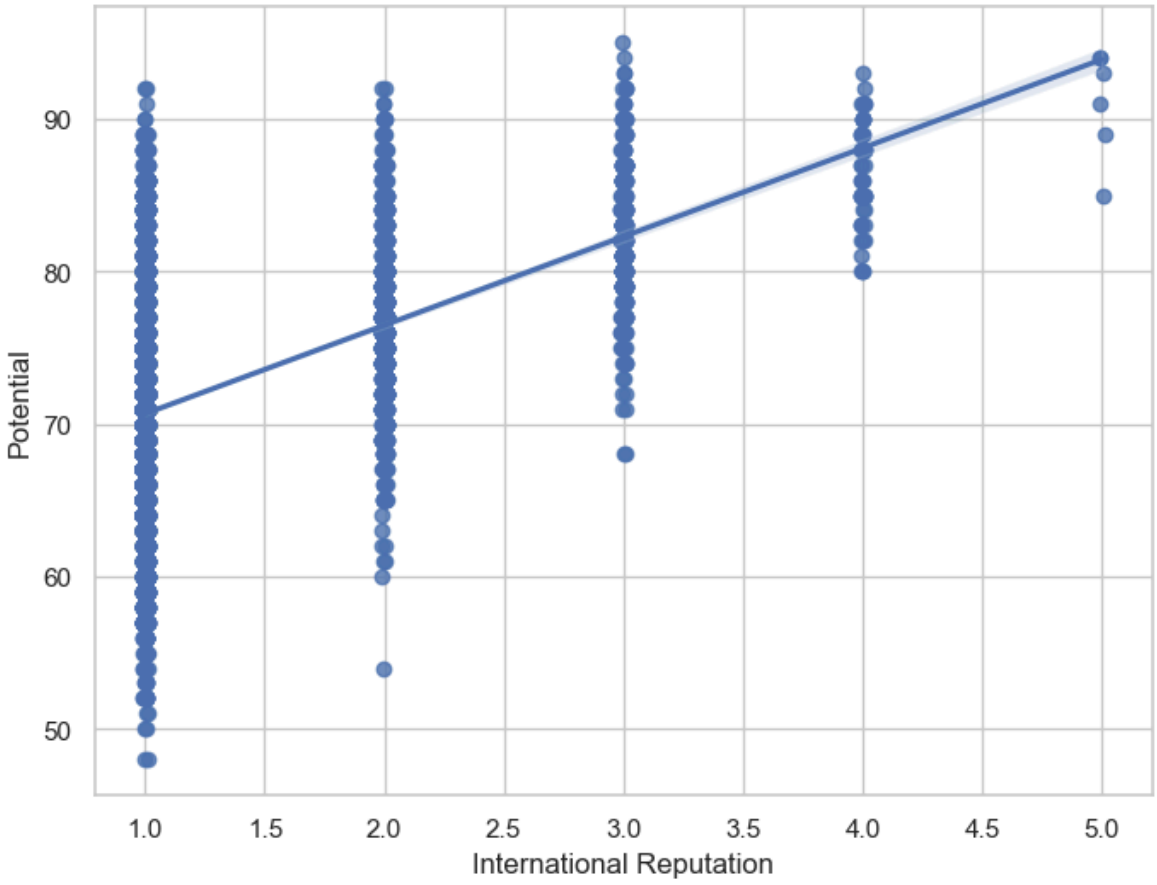
In [54]:
```
f,ax=plt.subplots(figsize=(8,6))
ax=sns.regplot(x="Overall",y="Potential",data=fifa19)
plt.show()
```



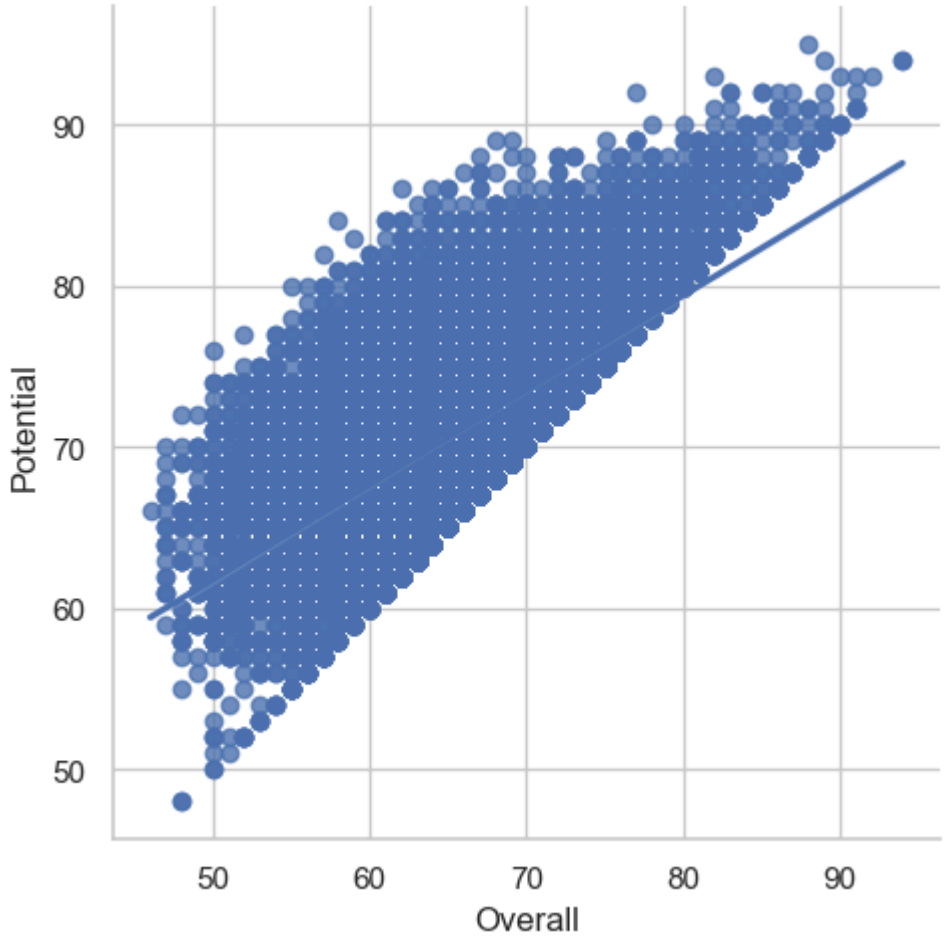
In [55]:
```
f,ax=plt.subplots(figsize=(8,6))
ax=sns.regplot(x="Overall",y="Potential",data=fifa19,color='g',marker="+")
plt.show()
```



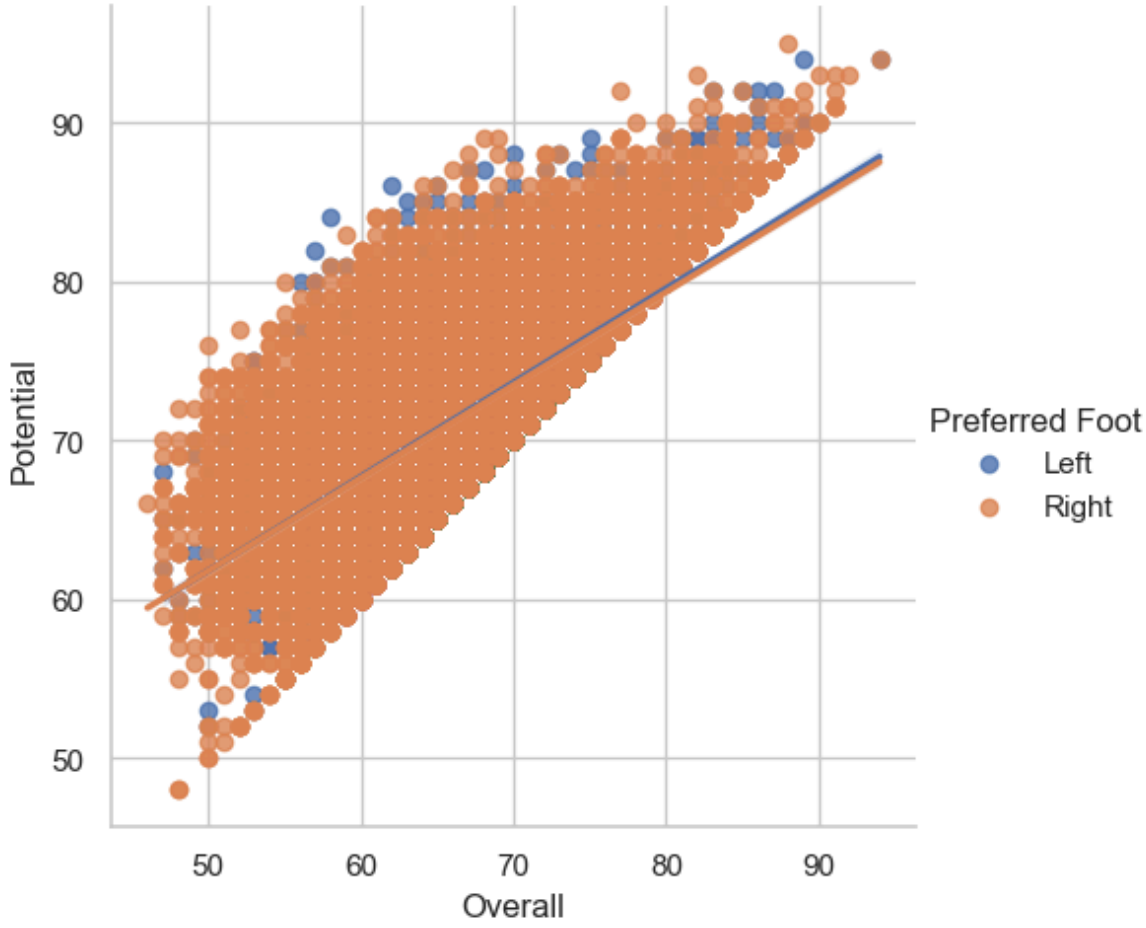
In [56]:
```
f,ax=plt.subplots(figsize=(8,6))
sns.regplot(x="International Reputation",y="Potential",data=fifa19,x_jitter=.01)
plt.show()
```



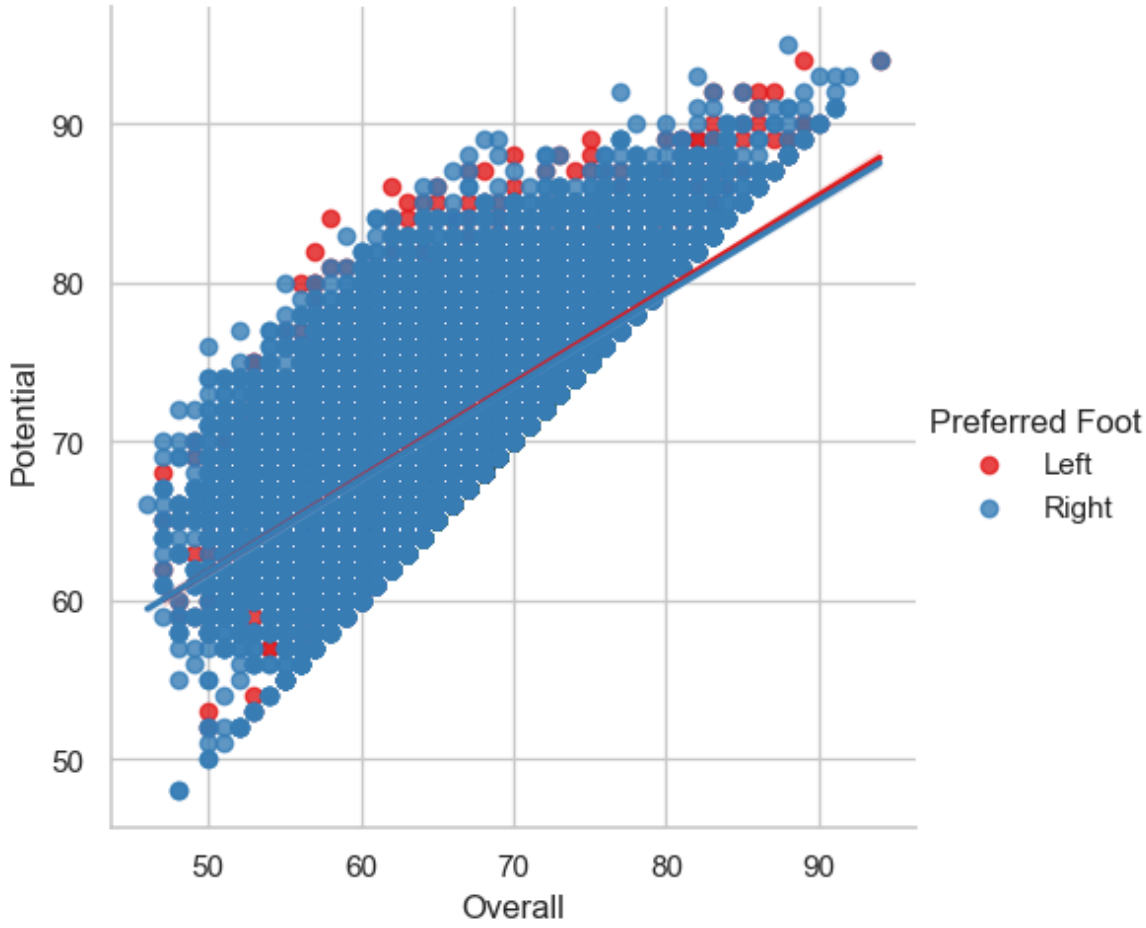
In [57]:
```
g=sns.lmplot(x="Overall",y="Potential",data=fifa19)
plt.show()
```



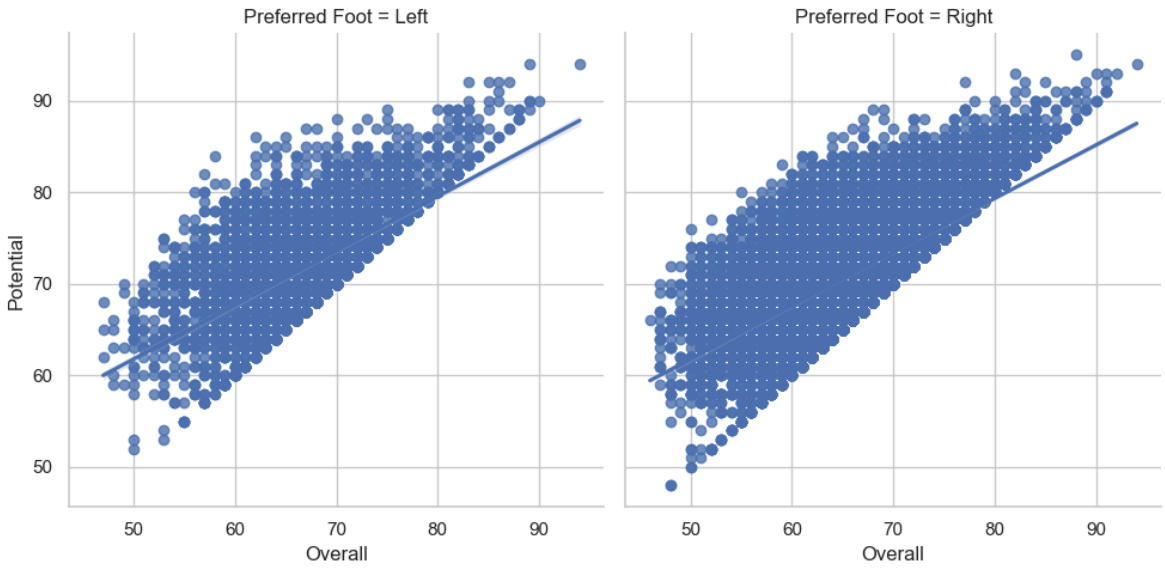
In [58]:
```
g=sns.lmplot(x="Overall",y="Potential",hue="Preferred Foot",data=fifa19)
plt.show()
```



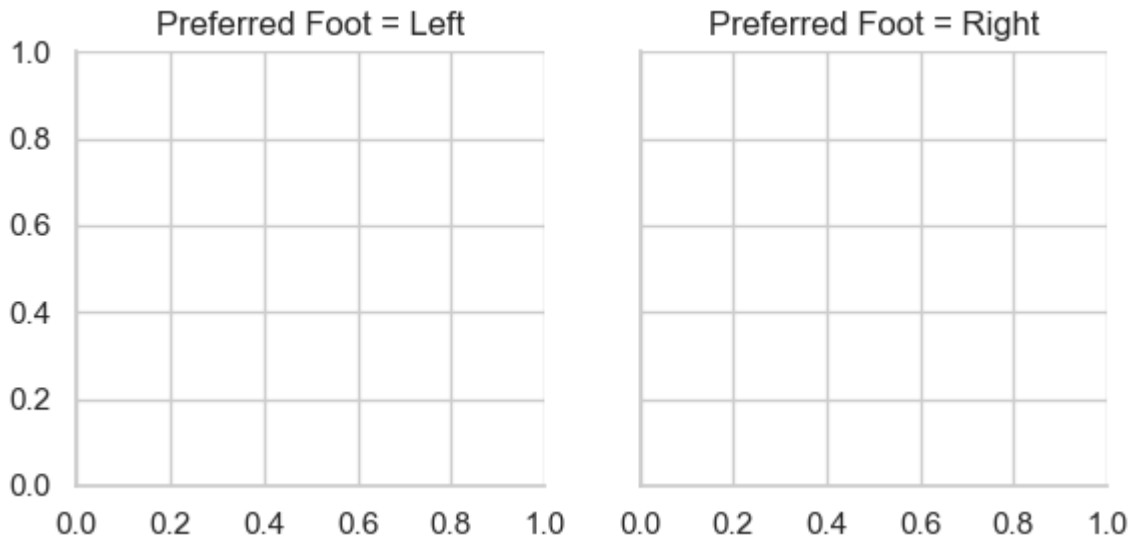
In [59]:
```
g=sns.lmplot(x="Overall",y="Potential",hue="Preferred Foot",data=fifa19,palette=
plt.show()
```



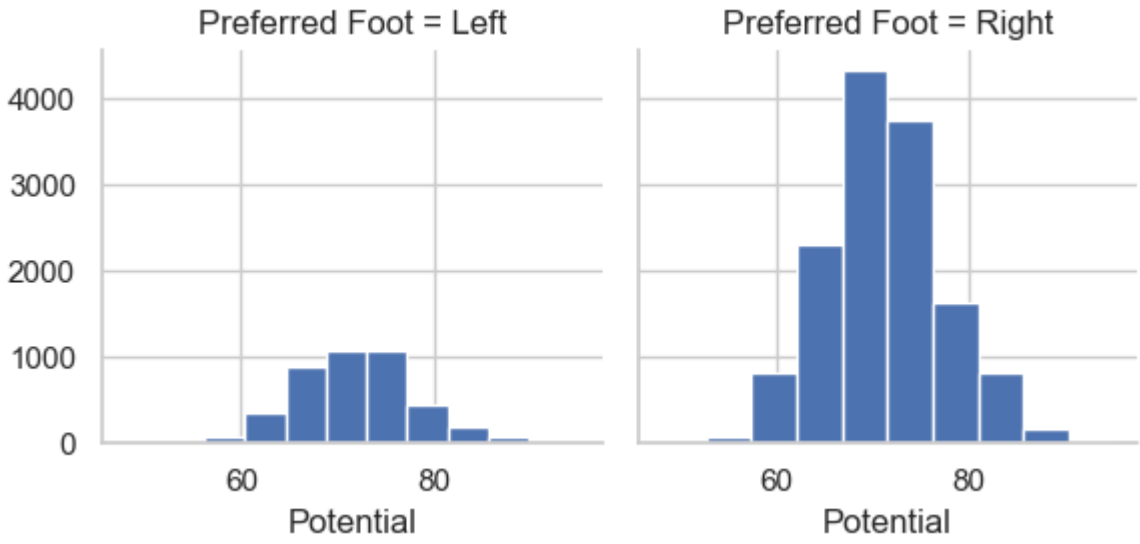
In [60]:
```
g=sns.lmplot(x="Overall",y="Potential",col="Preferred Foot",data=fifa19)
plt.show()
```



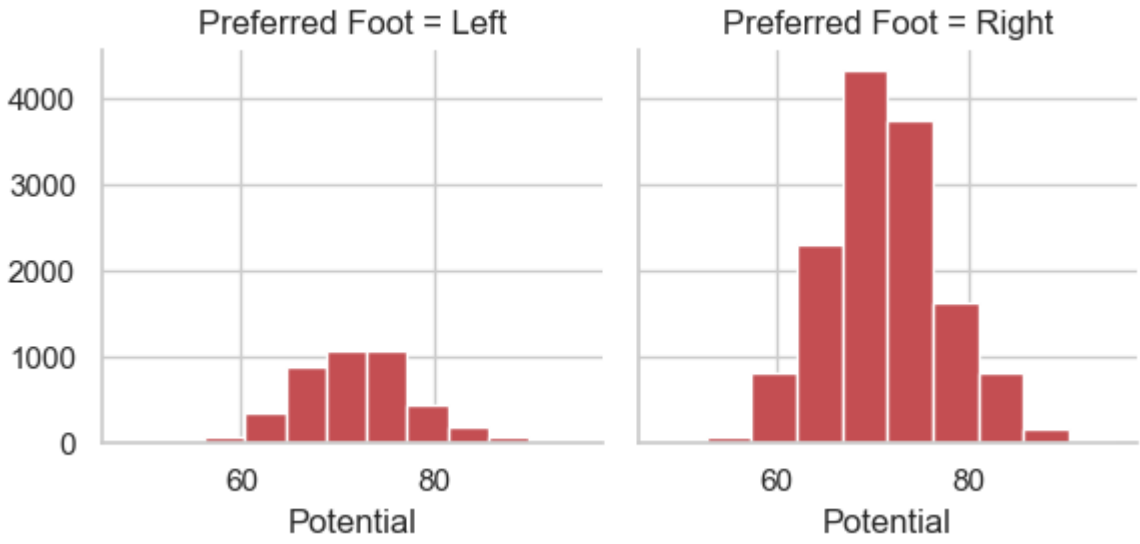
In [61]:
```
g=sns.FacetGrid(fifa19,col="Preferred Foot")
plt.show()
```



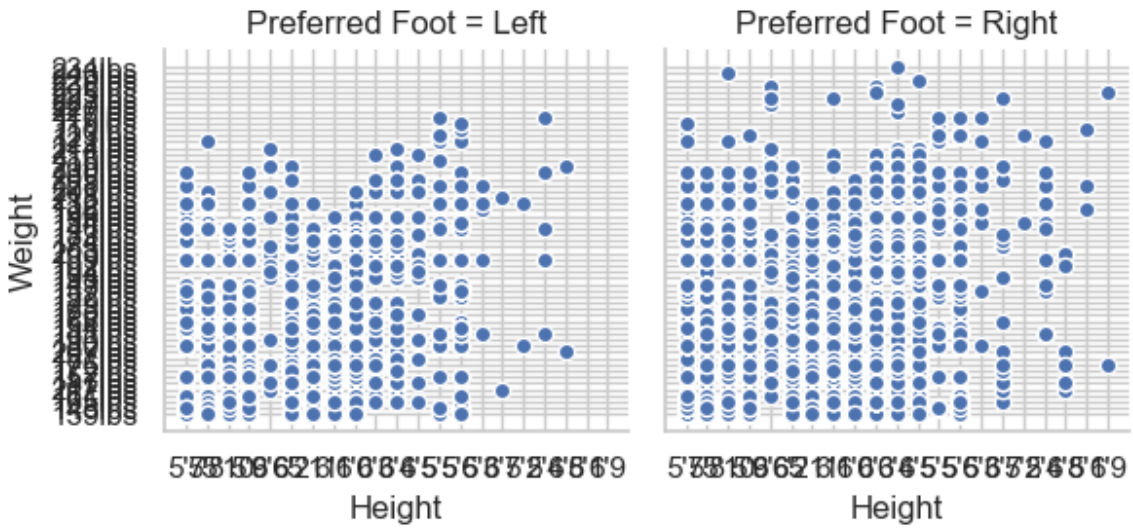
In [62]:
```
g=sns.FacetGrid(fifa19,col="Preferred Foot")
g=g.map(plt.hist,"Potential")
```



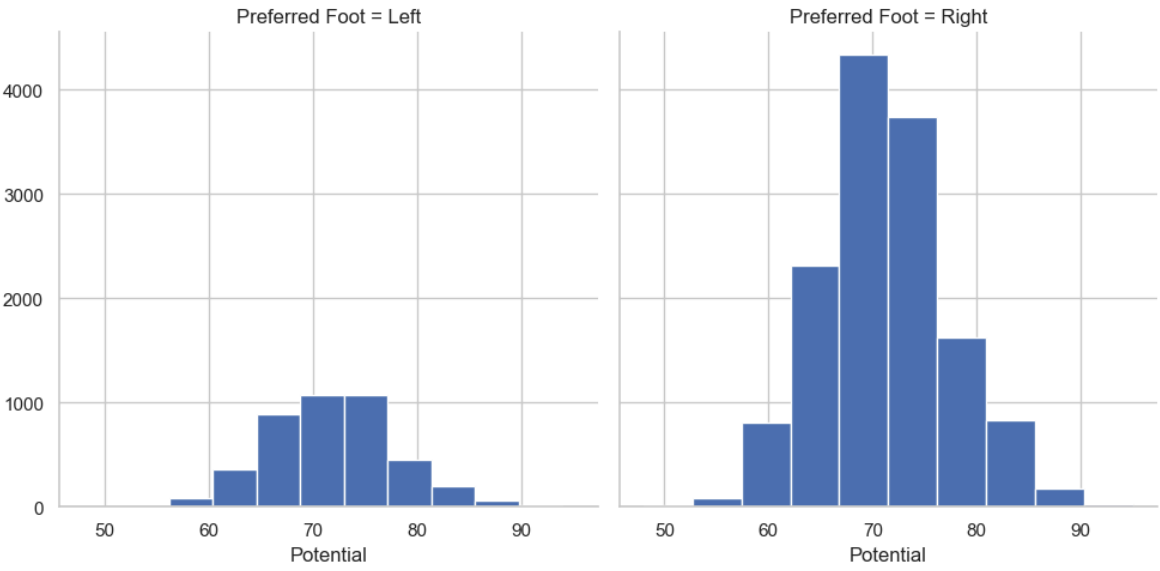
In [63]:
```
g=sns.FacetGrid(fifa19,col="Preferred Foot")
g=g.map(plt.hist,"Potential",bins=10,color="r")
```



In [64]:
```
g=sns.FacetGrid(fifa19,col="Preferred Foot")
g=(g.map(plt.scatter,"Height","Weight",edgecolor="w").add_legend())
```

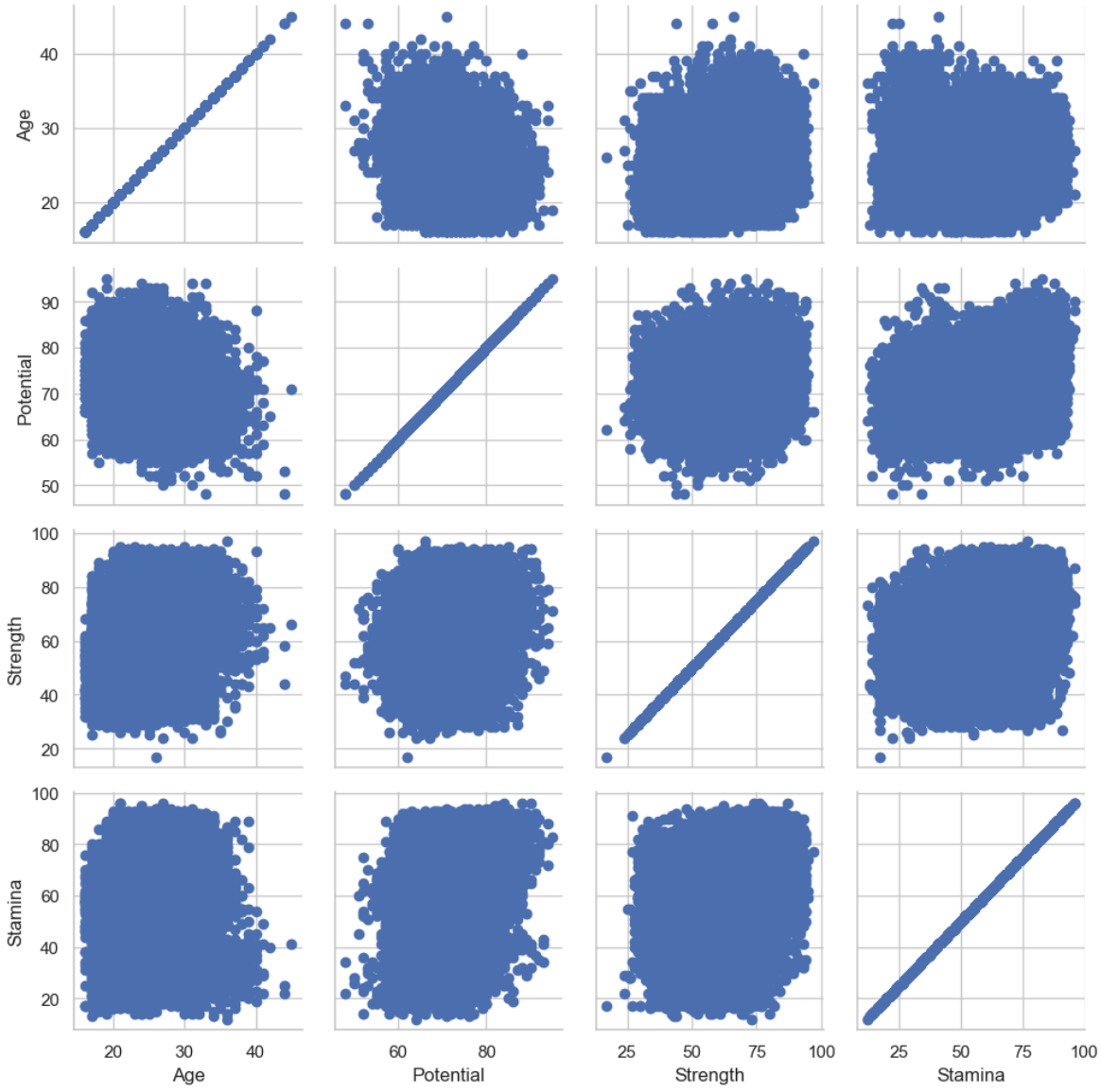


In [65]:
```
g=sns.FacetGrid(fifa19,col="Preferred Foot",height=5,aspect=1)
g=g.map(plt.hist,"Potential")
```



In [66]:
```
fifa19_new=fifa19[['Age','Potential','Strength','Stamina','Preferred Foot']]
```

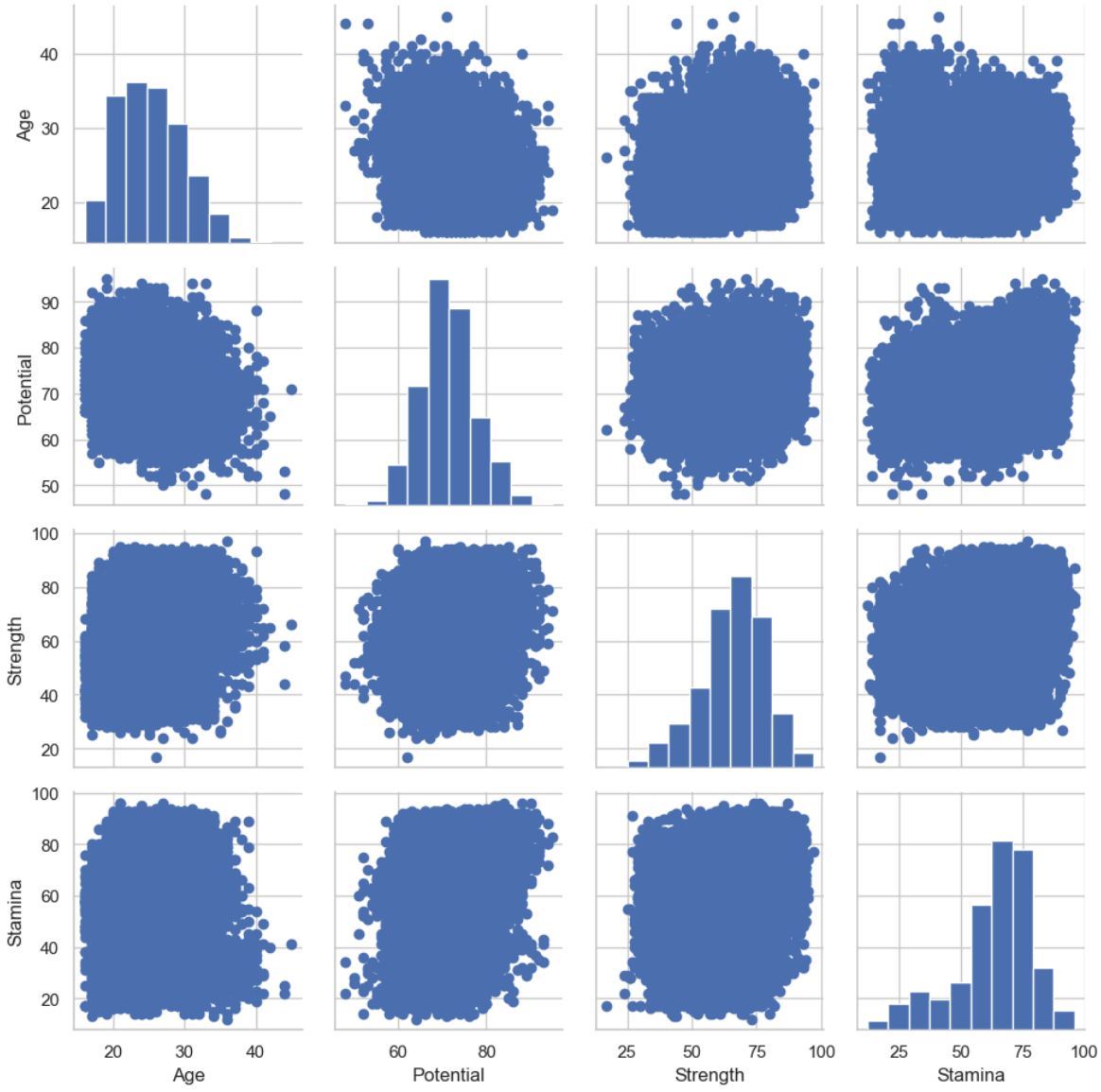
In [67]:
```
g=sns.PairGrid(fifa19_new)
g=g.map(plt.scatter)
```



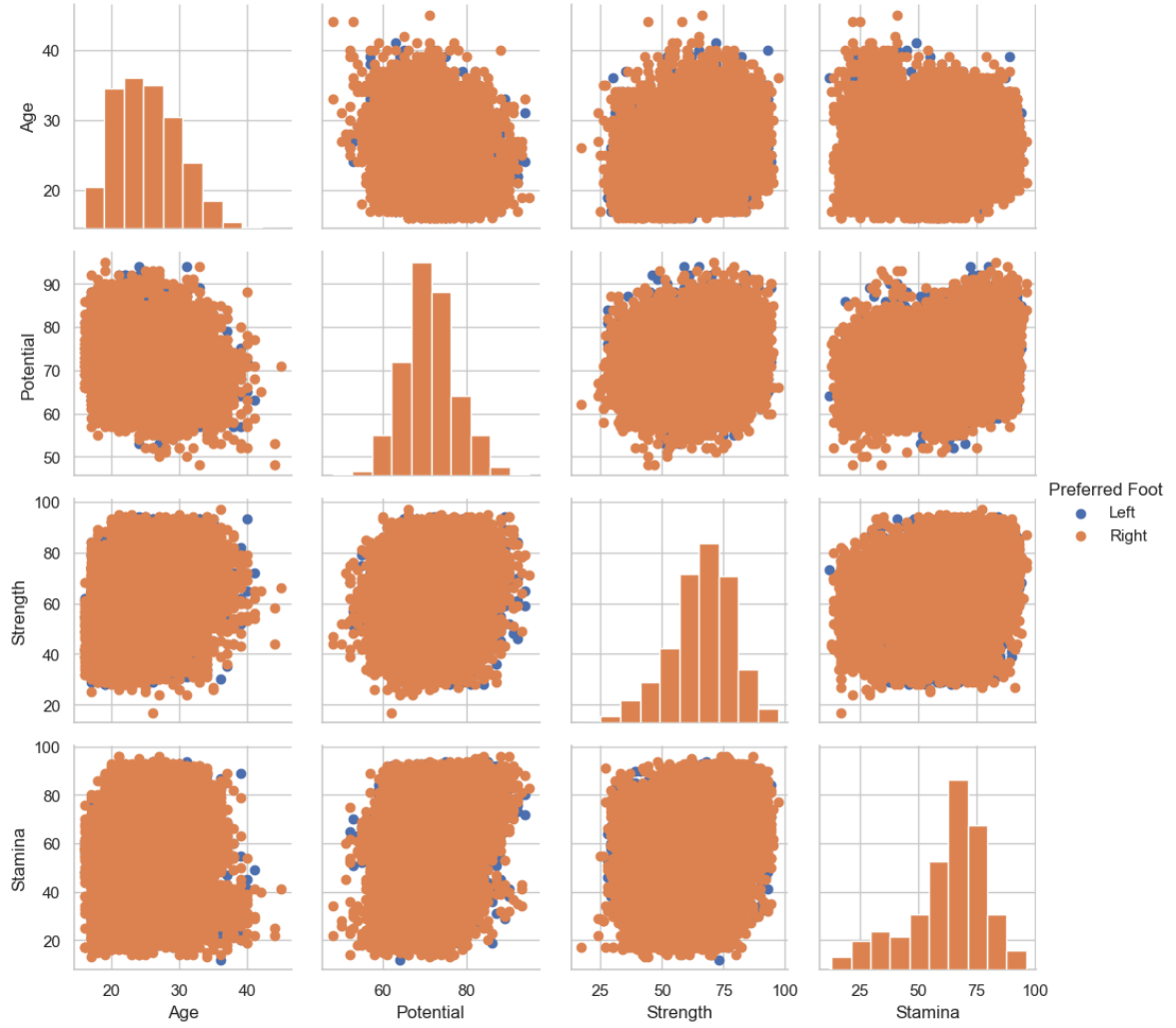
In [68]:
```
g=sns.Pairplot(fifa19_new)
g=g.map_diag(plt.hist)
g=g.map_offdiag(plt.scatter)
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
Cell In[68], line 1
----> 1 g=sns.Pairplot(fifa19_new)
      2 g=g.map_diag(plt.hist)
      3 g=g.map_offdiag(plt.scatter)

AttributeError: module 'seaborn' has no attribute 'Pairplot'
```

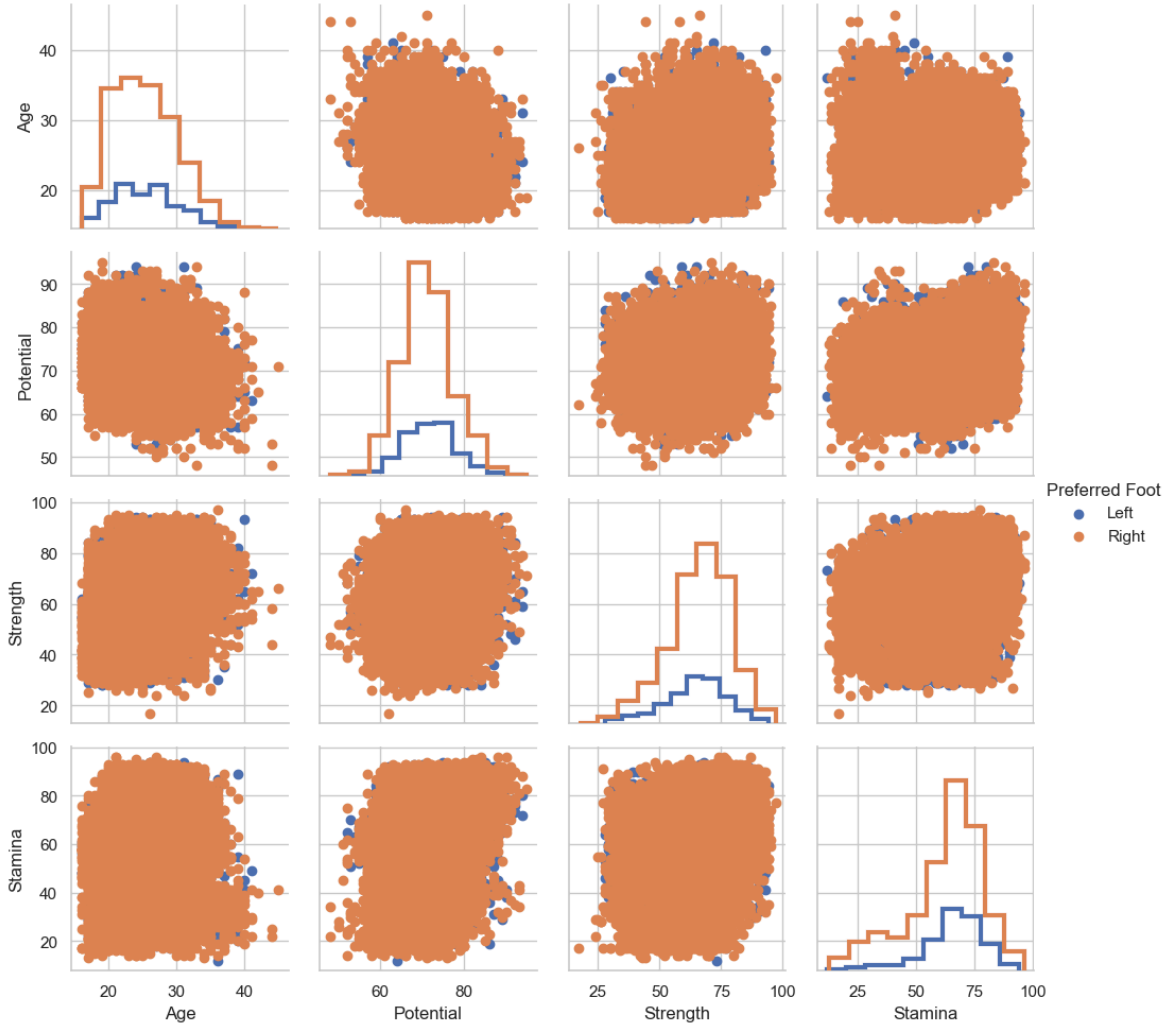
In [69]:
```
g=sns.PairGrid(fifa19_new)
g=g.map_diag(plt.hist)
g=g.map_offdiag(plt.scatter)
```



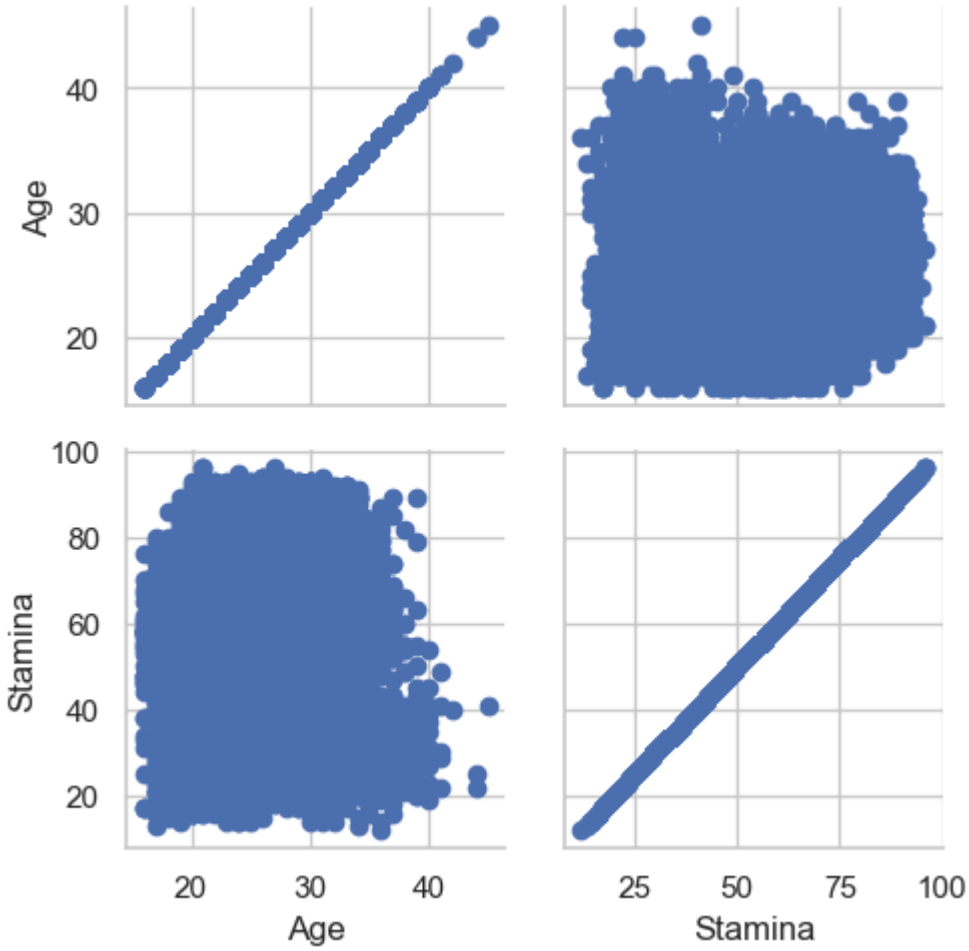
In [70]:
```
g=sns.PairGrid(fifa19_new,hue="Preferred Foot")
g=g.map_diag(plt.hist)
g=g.map_offdiag(plt.scatter)
g=g.add_legend()
```



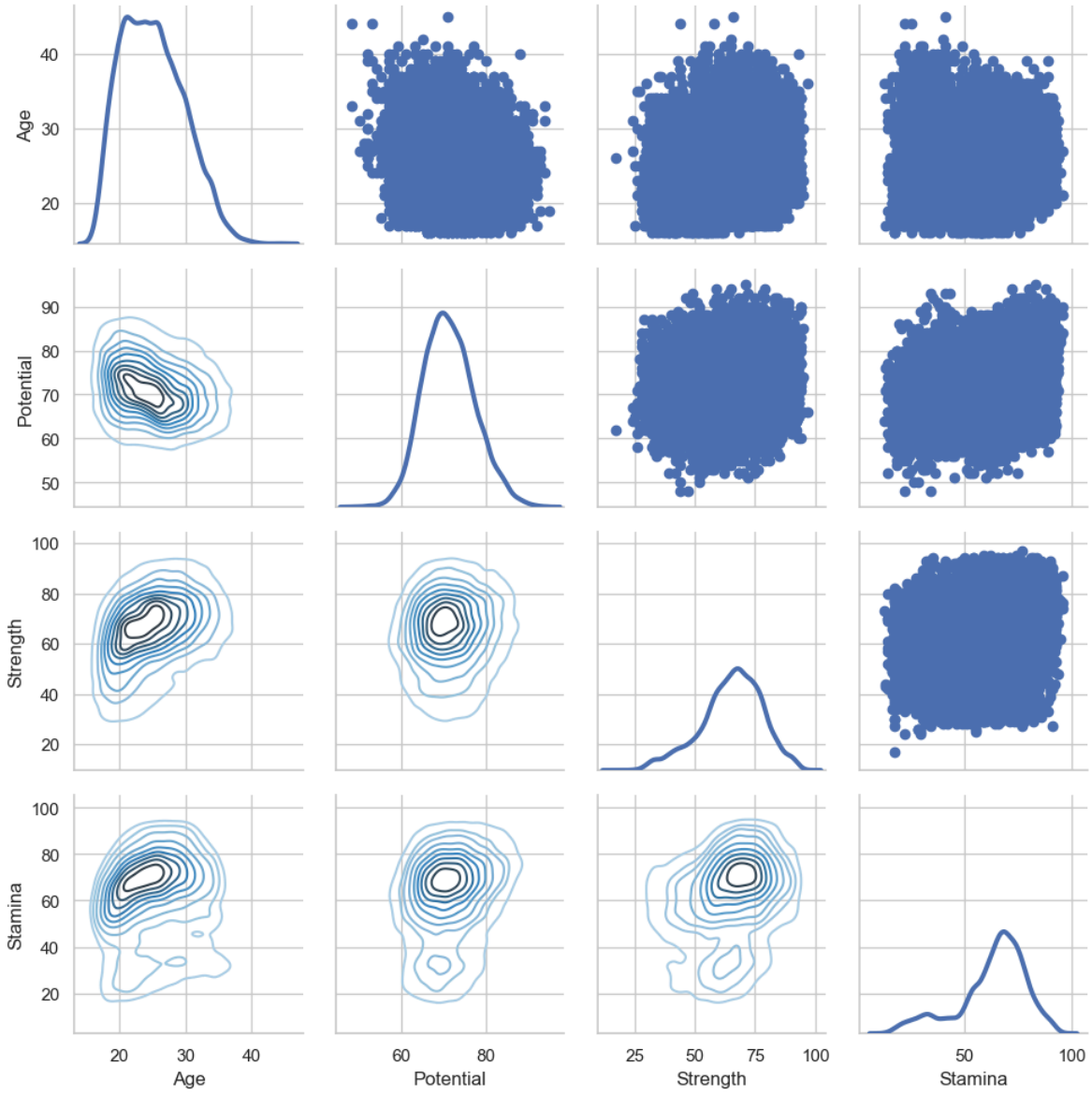
In [71]:
```
g=sns.PairGrid(fifa19_new,hue="Preferred Foot")
g=g.map_diag(plt.hist,histtype="step",linewidth=3)
g=g.map_offdiag(plt.scatter)
g=g.add_legend()
```



In [72]:
```
g=sns.PairGrid(fifa19_new,vars=['Age','Stamina'])
g=g.map(plt.scatter)
```

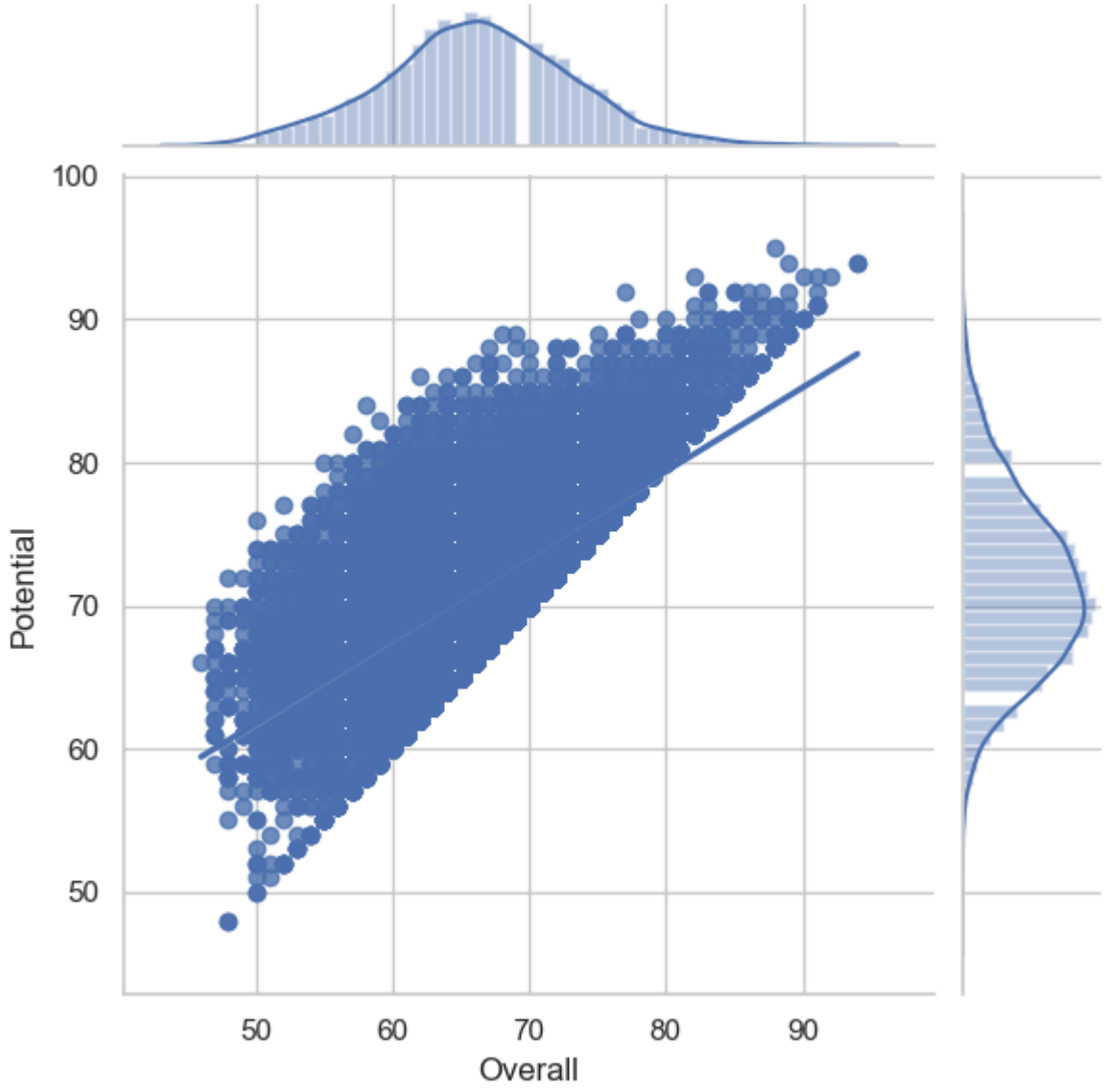


In [73]:
```
g=sns.PairGrid(fifa19_new)
g=g.map_upper(plt.scatter)
g=g.map_lower(sns.kdeplot,cmap="Blues_d")
g=g.map_diag(sns.kdeplot,lw=3,legend=False)
```



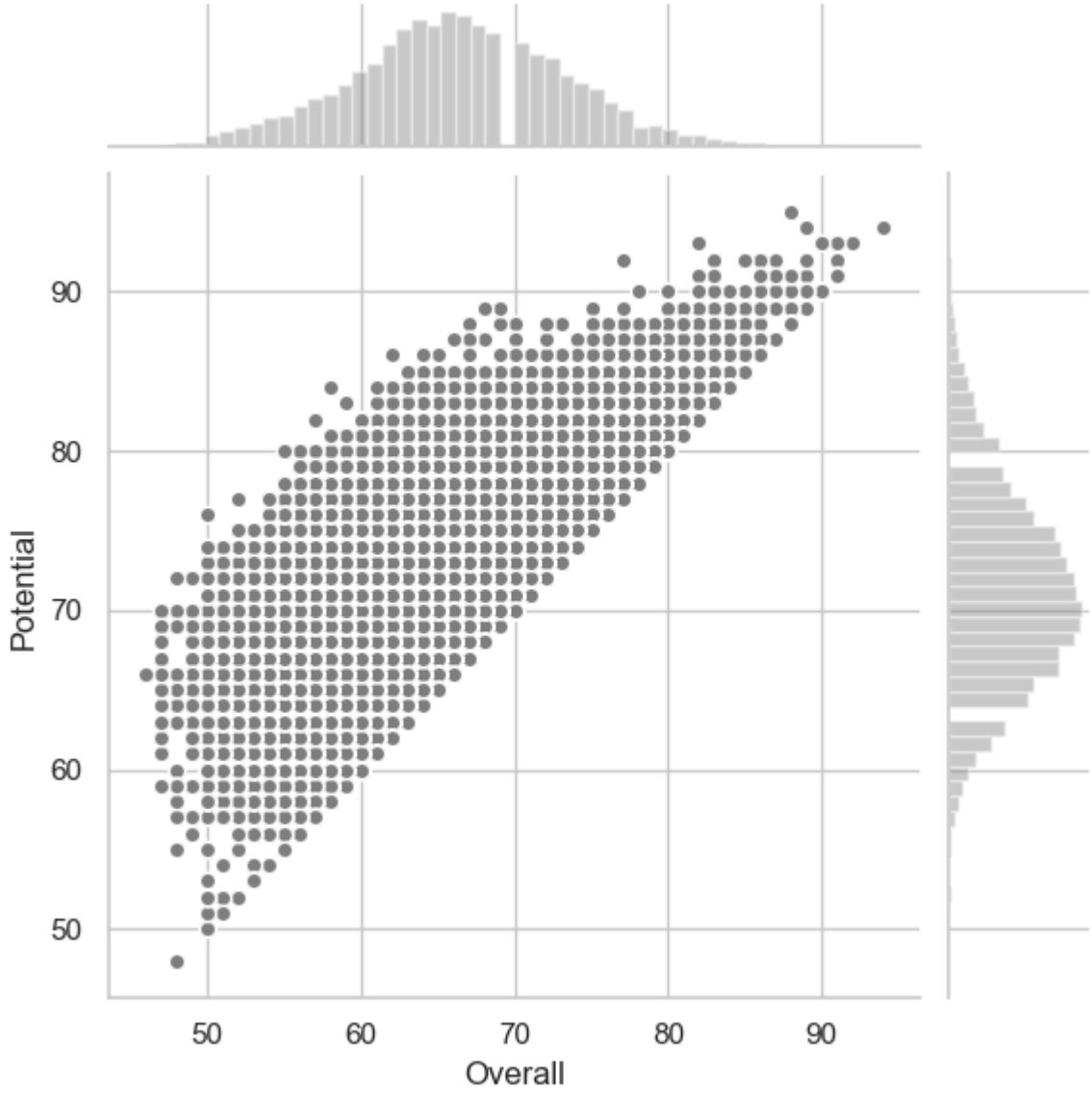
Seaborn Jointgrid() function

In [74]:
```
g=sns.JointGrid(x="Overall",y="Potential",data=fifa19)
g=g.plot(sns.regplot,sns.distplot)
```

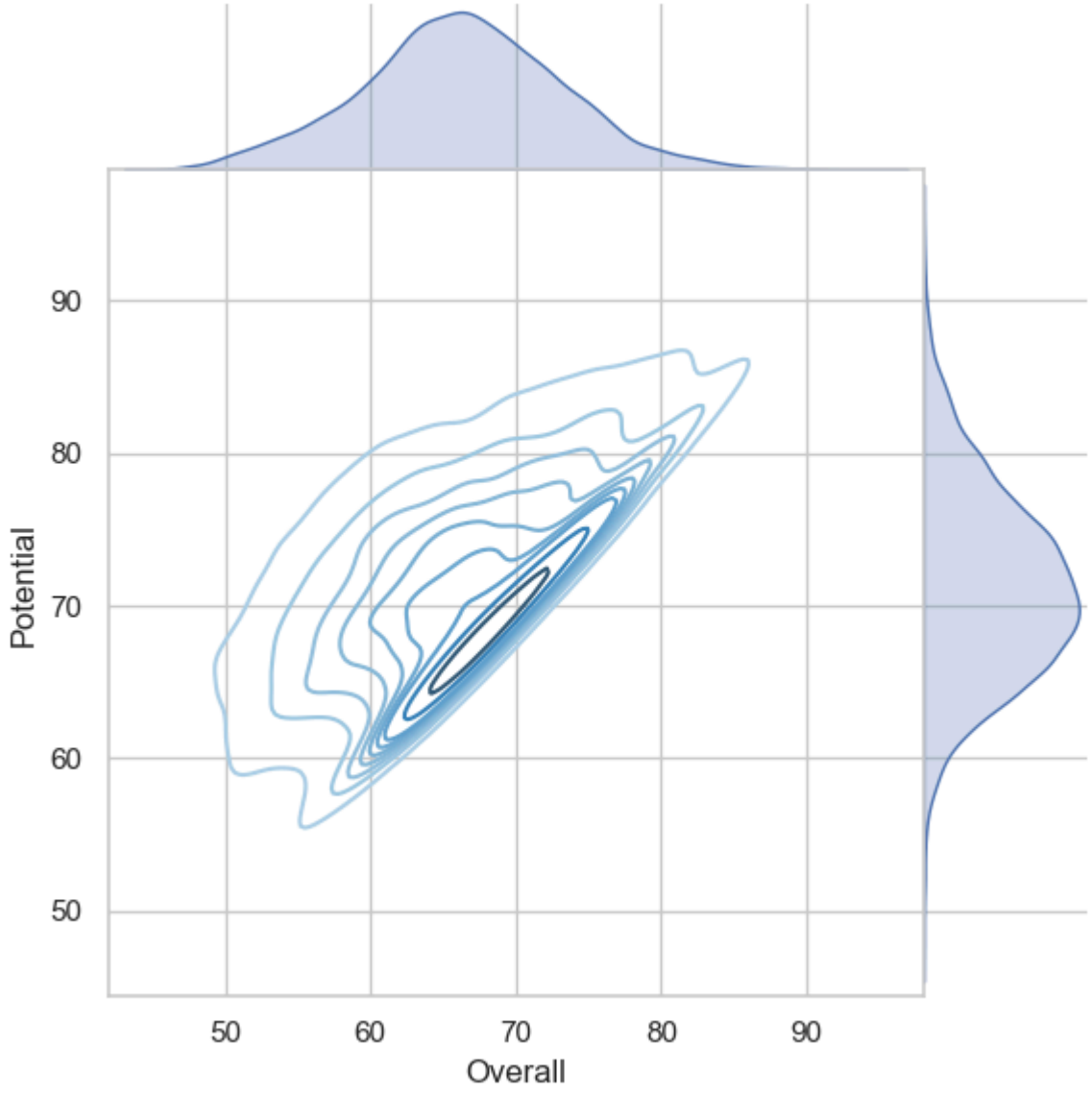


In [75]:
```
import matplotlib.pyplot as plt
```

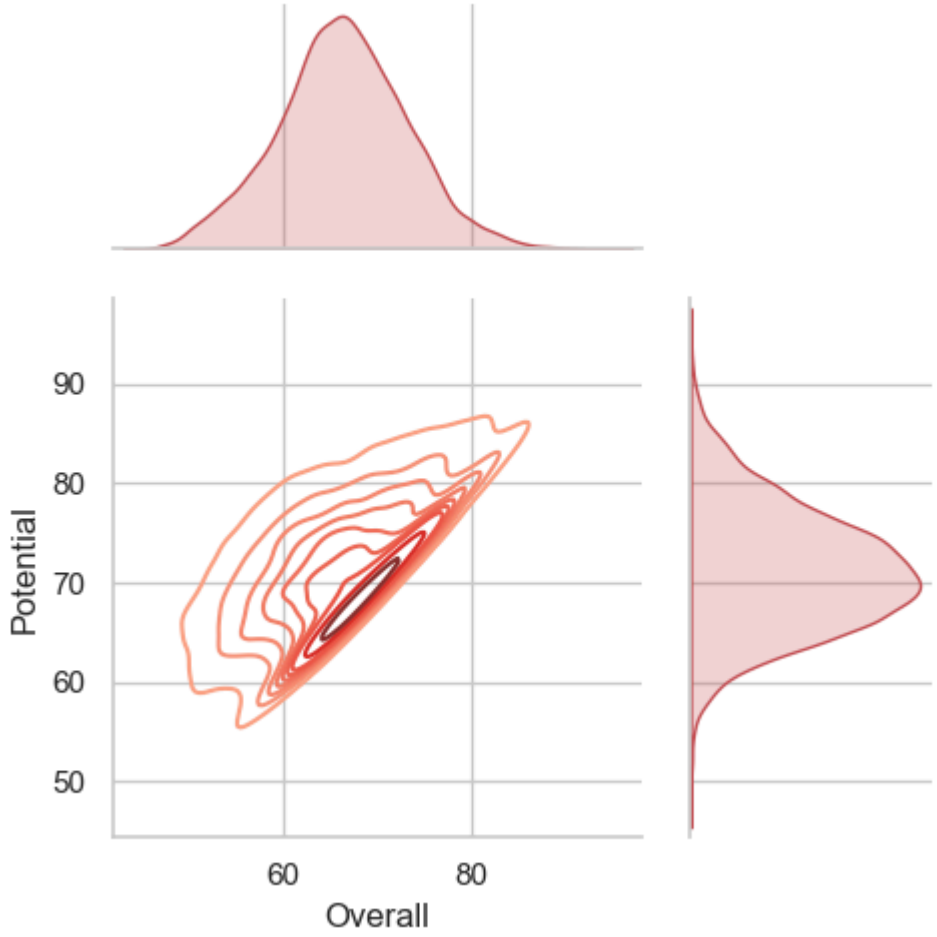
In [76]:
```
g=sns.JointGrid(x="Overall",y="Potential",data=fifa19)
g=g.plot_joint(plt.scatter,color=".5",edgecolor="white")
g=g.plot_marginals(sns.distplot,kde=False,color=".5")
```



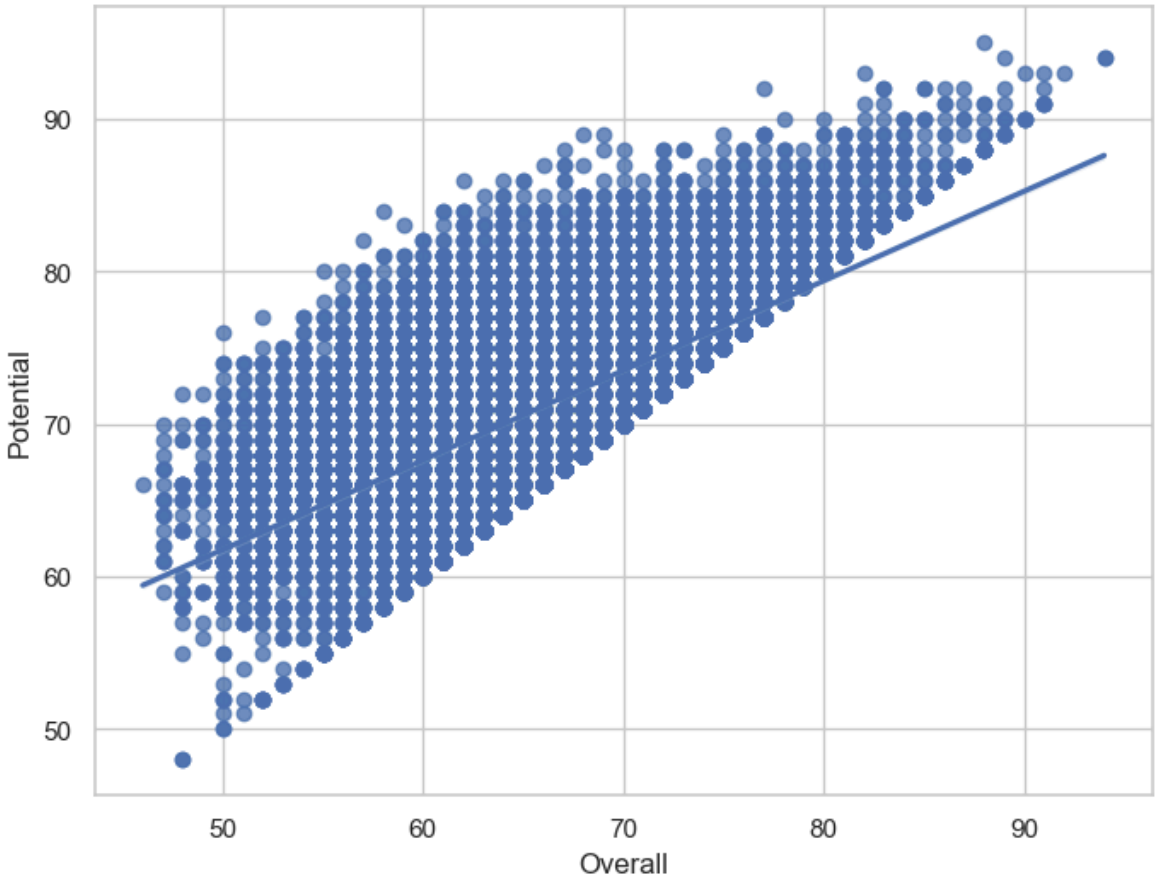
In [77]:
```
g=sns.JointGrid(x="Overall",y="Potential",data=fifa19,space=0)
g=g.plot_joint(sns.kdeplot,cmap="Blues_d")
g=g.plot_marginals(sns.kdeplot,shade=True)
```



In [78]:
```
g=sns.JointGrid(x="Overall",y="Potential",data=fifa19,height=5,ratio=2)
g=g.plot_joint(sns.kdeplot,cmap="Reds_d")
g=g.plot_marginals(sns.kdeplot,color="r",shade=True)
```

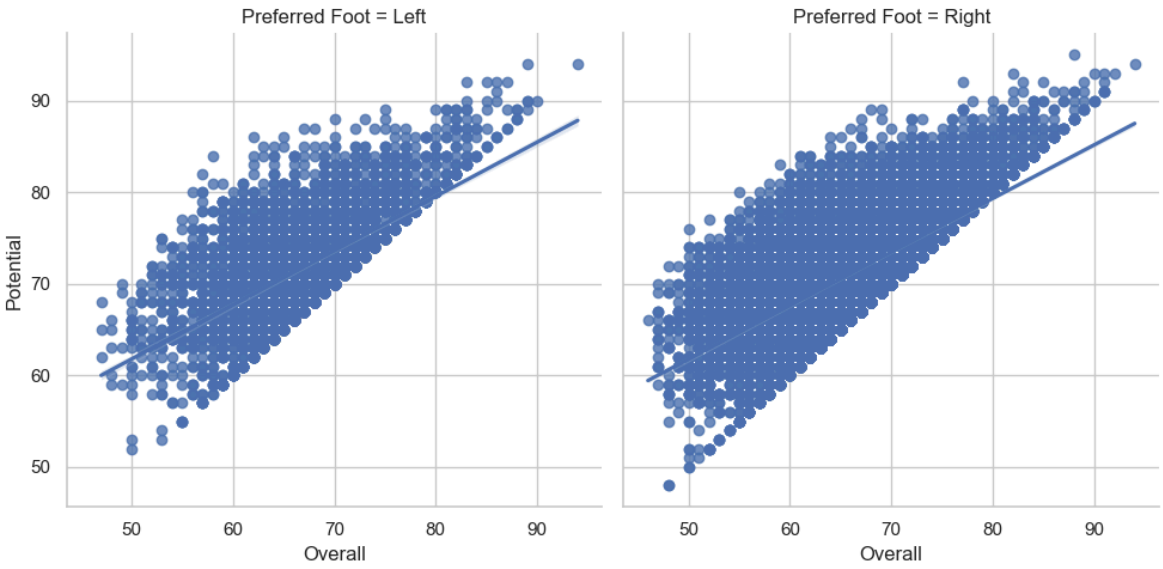


In [79]:
```
f,ax=plt.subplots(figsize=(8,6))
ax=sns.regplot(x="Overall",y="Potential",data=fifa19)
```



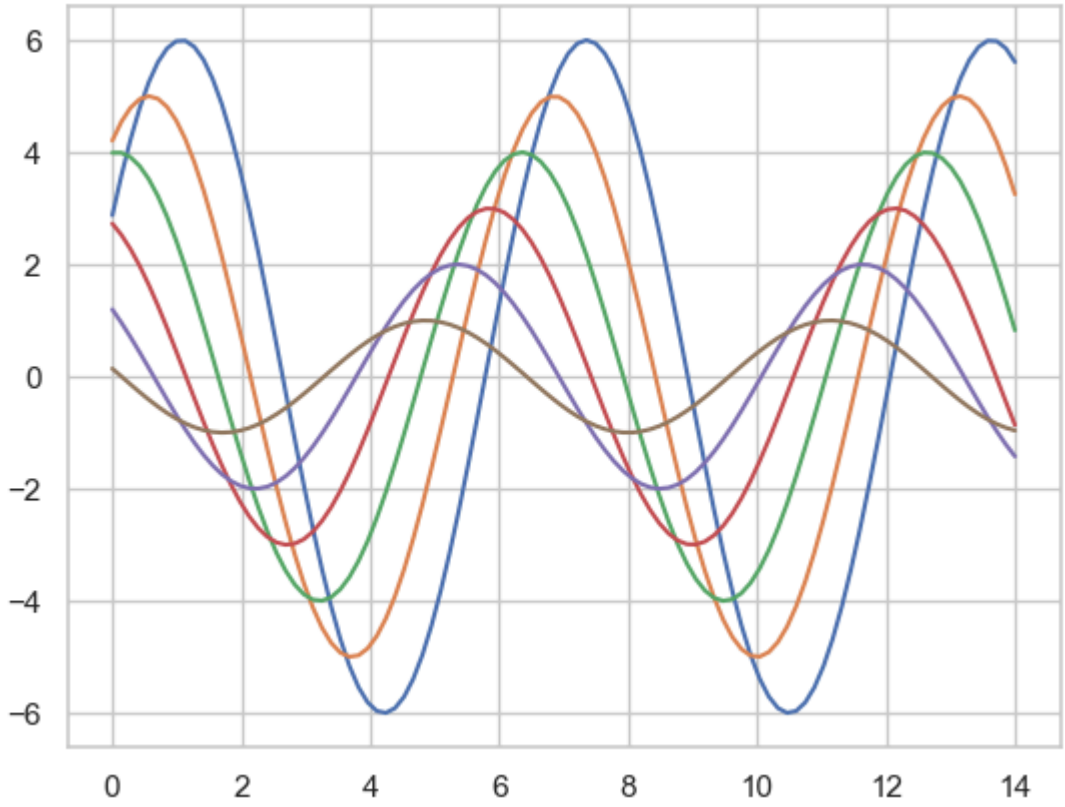
In [80]:
```
sns.lmplot(x="Overall",y="Potential",col="Preferred Foot",data=fifa19,col_wrap=2
```

Out[80]:
```
<seaborn.axisgrid.FacetGrid at 0x2198a37e850>
```

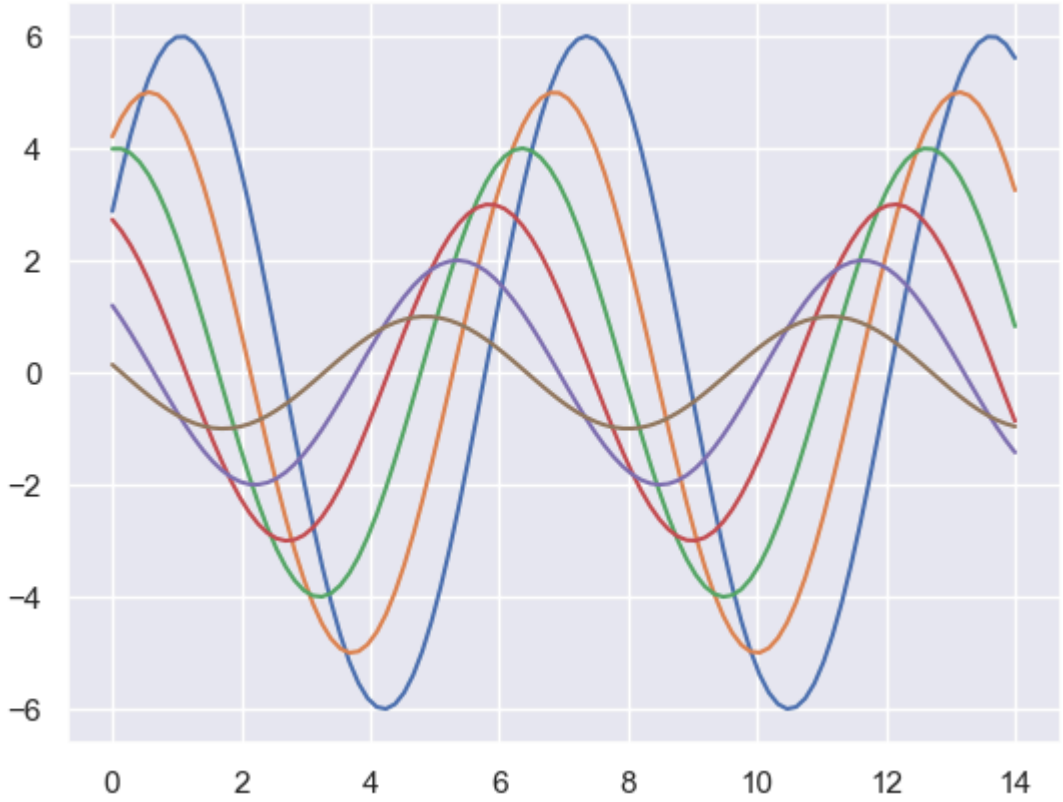


In [81]:
```
def sinplot(flip=1):
    x=np.linspace(0,14,100)
    for i in range(1,7):
        plt.plot(x,np.sin(x+i*.5)*(7-i)*flip)
```

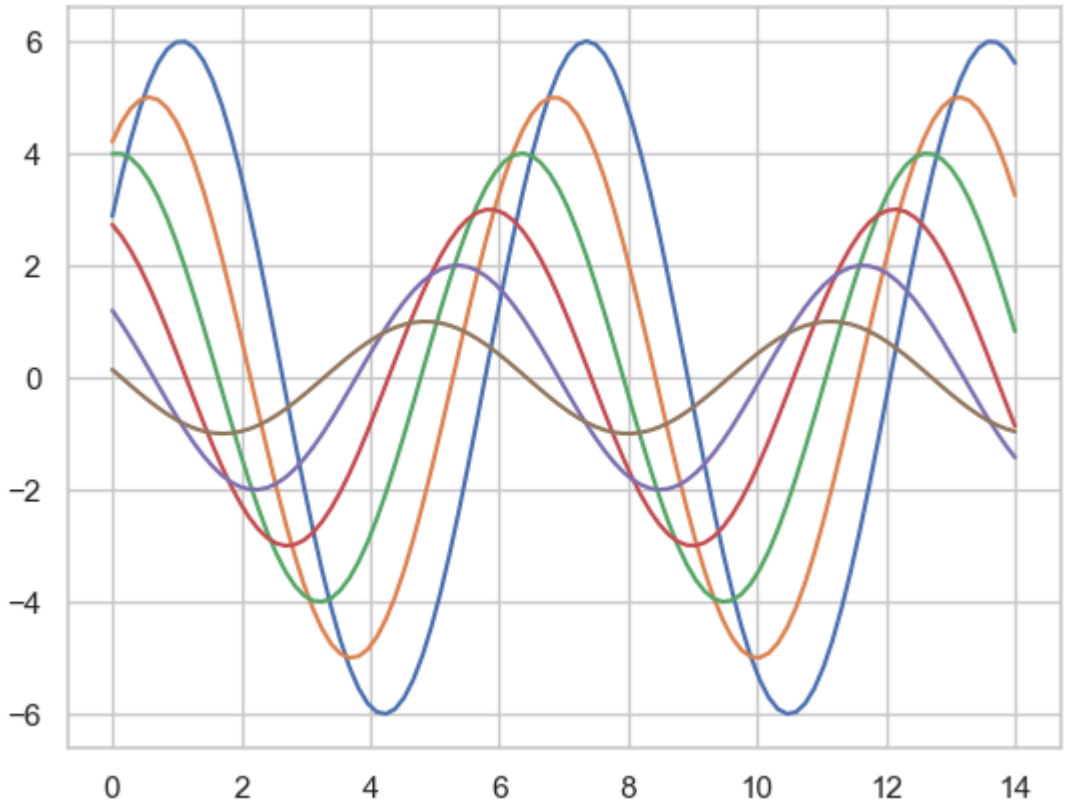
In [82]:
```
sinplot()
```



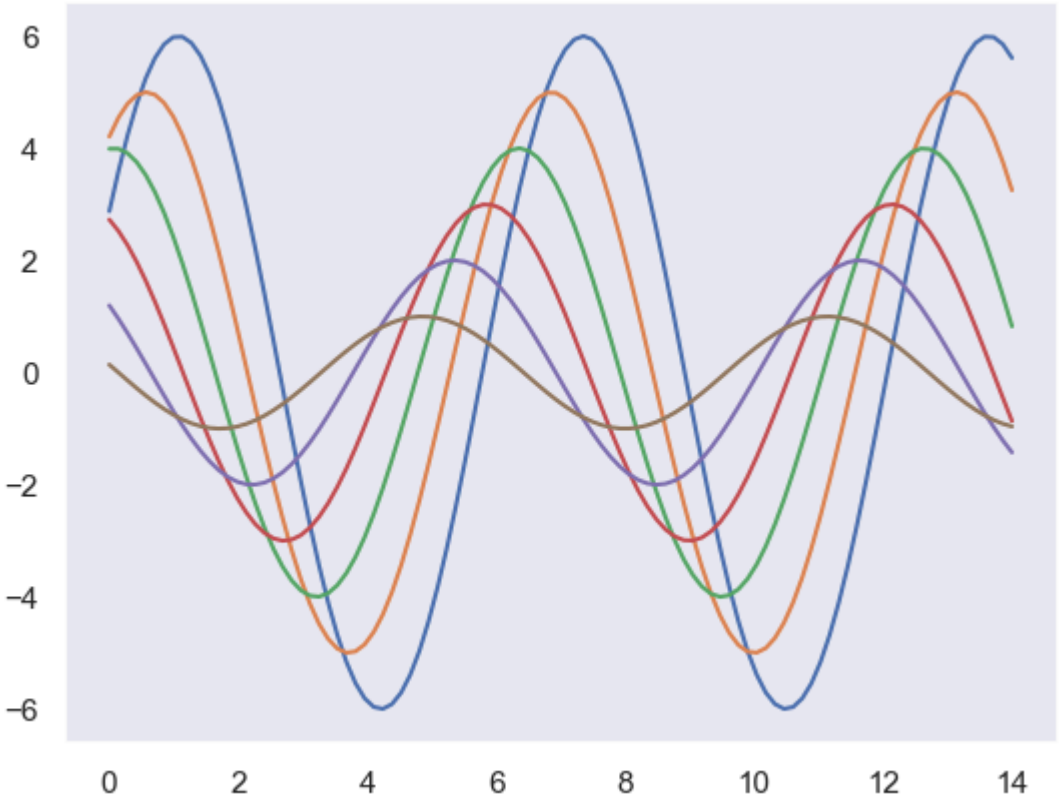
In [83]:
```
sns.set()
sinplot()
```



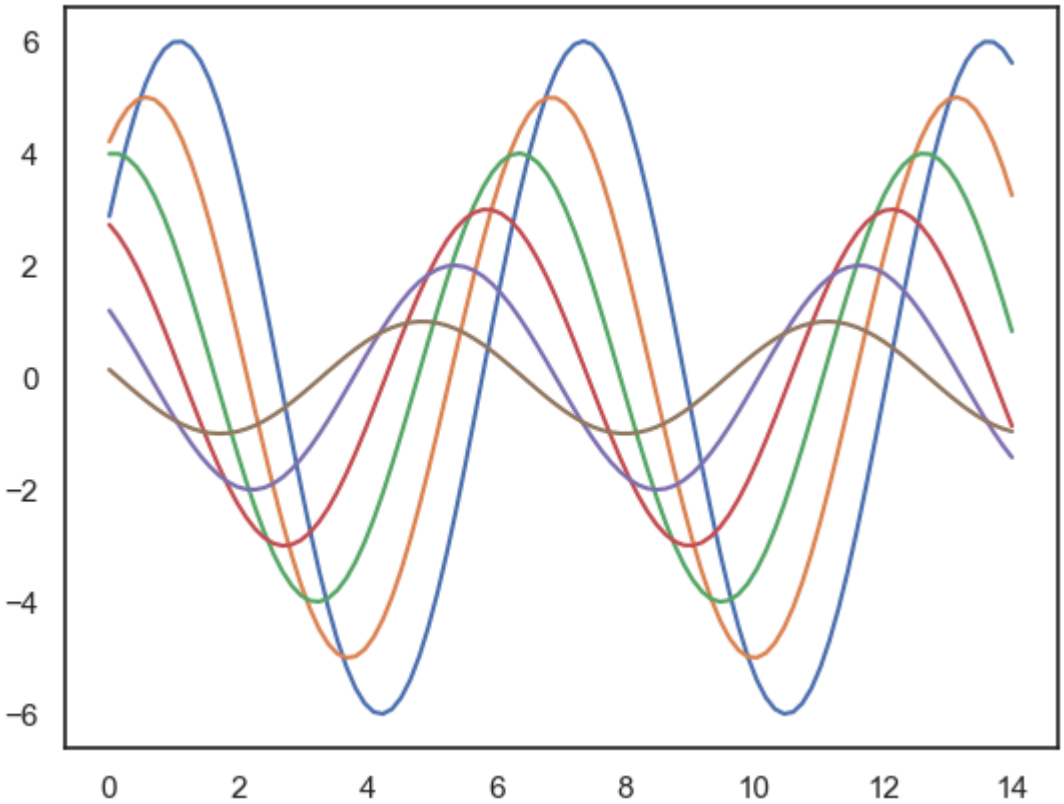
In [84]:
```
sns.set_style("whitegrid")
sinplot()
```



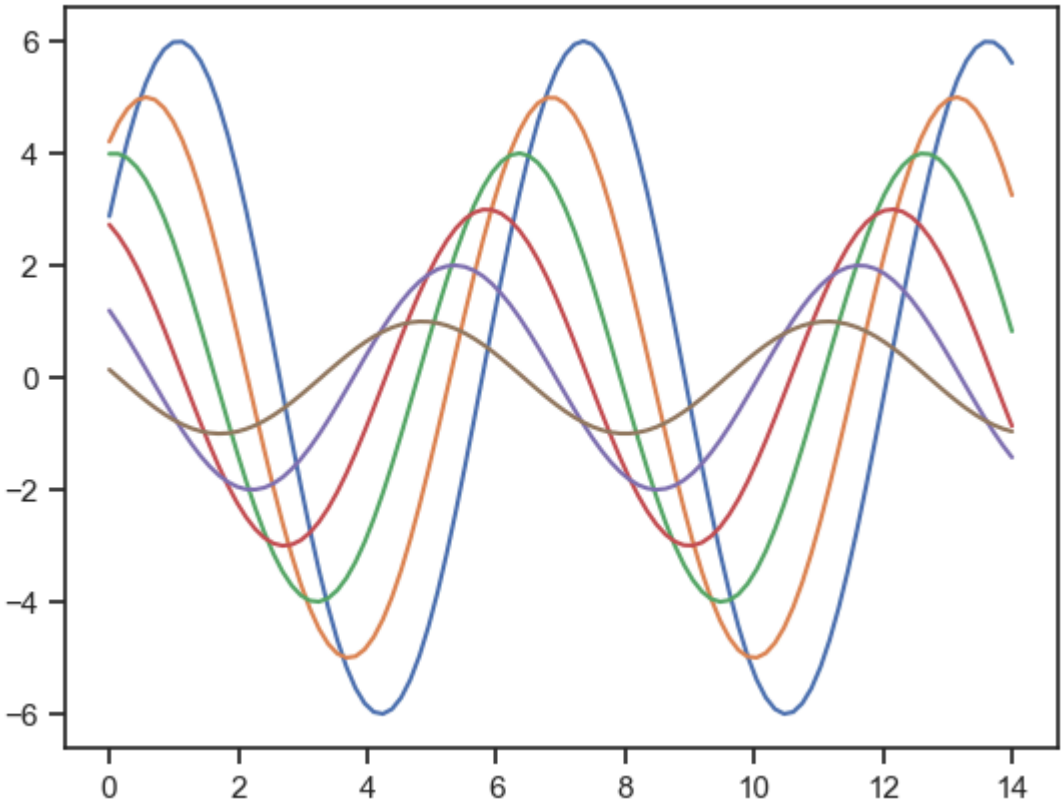
In [85]:
```
sns.set_style("dark")
sinplot()
```



In [86]:
```
sns.set_style("white")
sinplot()
```



In [87]:
```
sns.set_style("ticks")
sinplot()
```



In [ ]: