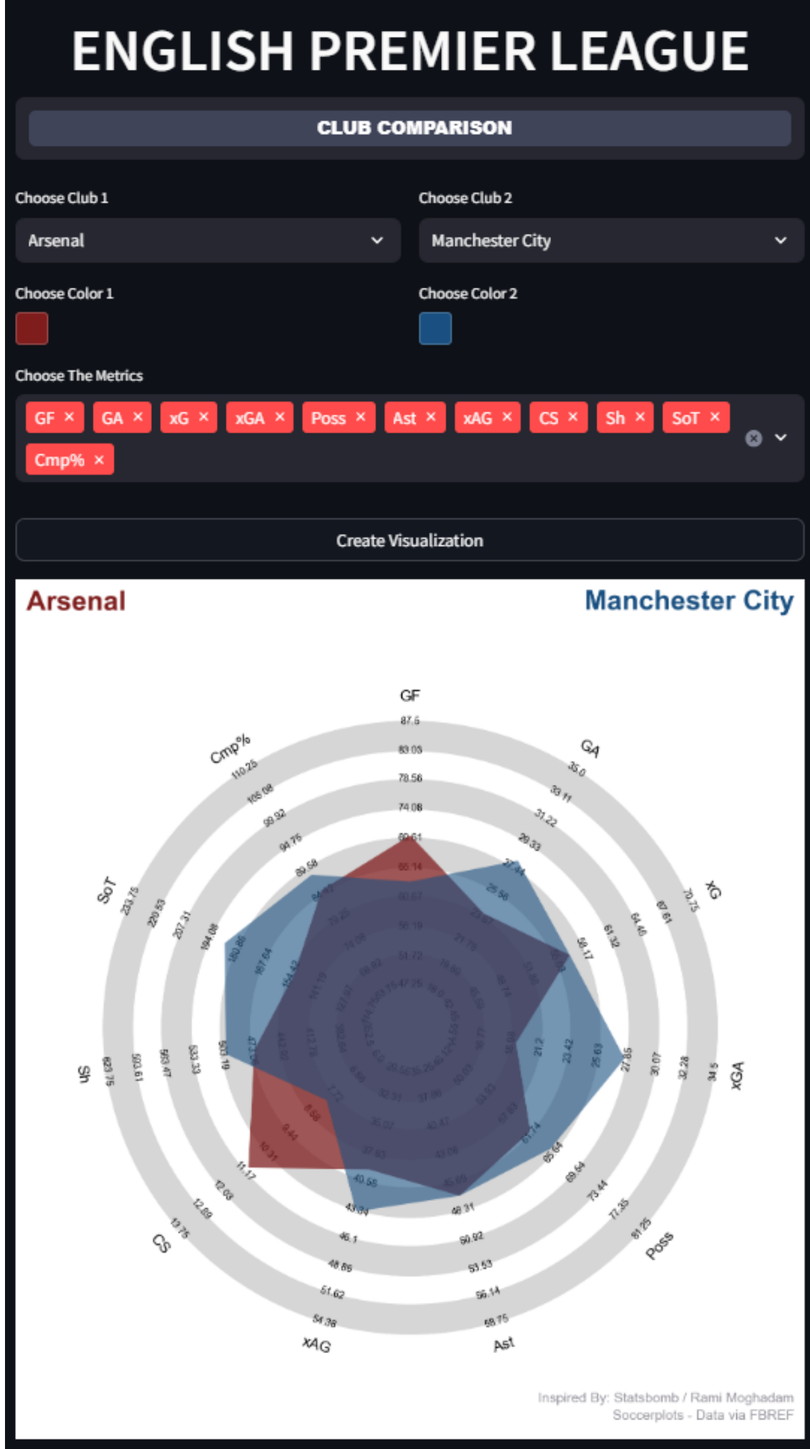
PREMIER LEAGUE CLUB COMPARISONS

UNRAVELING INSIGHTS WITH SOCCERPLOTS RADAR CHARTS
AN INTERACTIVE JOURNEY THROUGH PERFORMANCE ANALYSIS

FBREF Streamlit

PUBLISHED BY : ALFIANSYAH



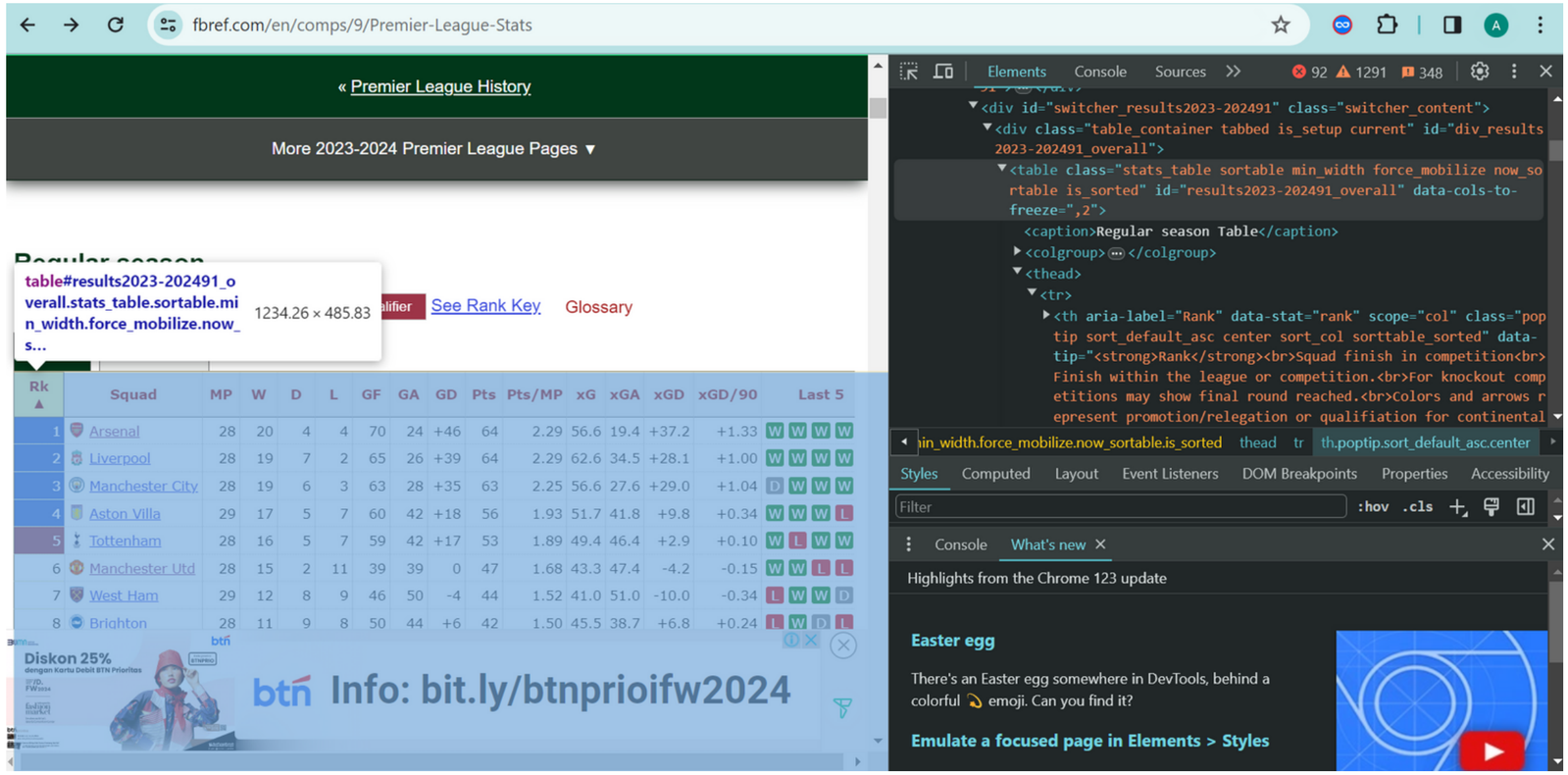
PROJECT FLOW

1	DATA COLLECTION	We will collect Premier League club statistics from FBREF solely by making requests and parsing HTML Table using pandas.
2	DATA PREPROCESSING	We will perform some data filtering.
3	DATA VISUALIZATION	We will create radar charts to compare selected metrics from two different clubs using Soccerplots.
4	BUILD WEB APP	We will build Web App using Streamlit.

DATA COLLECTION

Let's visit the FBREF website to see what data we want to gather, https://fbref.com/en/comps/9/Premier-League-Stats

FBREF primarily stores their data in HTML tables. When we access a specific page on FBREF, the data is typically presented in structured HTML tables, which makes it relatively straightforward to scrape or extract using web scraping techniques.



DATA COLLECTION

So, in this project, we'll use requests and pandas to read HTML from FBref to retrieve the tables and data that we want.

Let's start coding by import requests, setting the URL, and then making a request to that URL.

```
import requests
import warnings
warnings.filterwarnings("ignore")
```

```
url = "https://fbref.com/en/comps/9/Premier-League-Stats"
```

```
data_url = requests.get(url)
```

When we see the response, we notice that there's no useful information available for extraction as the response is still in request-response model format.

```
data_url
```

```
<Response [200]>
```

```
type(data_url)
```

```
requests.models.Response
```

DATA COLLECTION

Now, we need to extract the HTML text from that response. At this point, the data is more readable since it's already in HTML format.

```
html_text = data_url.text
html_text
```

```
'    \n      \n<!DOCTYPE html>\n<html data-version="klecko-" data-root="/home/fb/deploy/www/base" lang="en" class="no-js" >\n<head>\n    <meta charset
="utf-8">\n    <meta http-equiv="x-ua-compatible" content="ie=edge">\n    <meta name="viewport" content="width=device-width, initial-scale=1.0, maximu
m-scale=2.0" />\n    <link rel="dns-prefetch" href="https://cdn.ssref.net/req/202403271" />\n<!-- InMobi Choice. Consent Manager Tag v3.0 (for TCF 2.
2) -->\n<script type="text/javascript" async=true>\n(function() {\n  var host = window.location.hostname;\n  var element = document.createElement(\'sc
```

Next, let's read the HTML text using pandas. As we can see, the data now appears more organized in table format. However, what we obtain here is a list containing all the tables present on the FBREF page, as one FBREF page consists of multiple tables.

```
import pandas as pd
table_data = pd.read_html(html_text)
table_data
```

```
[    Rk            Squad  MP   W  D   L  GF  GA  GD  Pts  Pts/MP    xG   xGA  \
 0    1          Arsenal  28  20  4   4  70  24  46   64    2.29  56.6  19.4
 1    2        Liverpool  28  19  7   2  65  26  39   64    2.29  62.6  34.5
 2    3  Manchester City  28  19  6   3  63  28  35   63    2.25  56.6  27.6
 3    4      Aston Villa  29  17  5   7  60  42  18   56    1.93  51.7  41.8
```

```
type(table_data)
```

```
list
```

DATA COLLECTION

To retrieve a specific table from that list, we can use an index. For example, [0] indicates that we want to access the first table. If we refer to the FBREF website, this table is for Overall Regular season table.

```
table_data[0].head(5)
```

	Rk	Squad	MP	W	D	L	GF	GA	GD	Pts	Pts/MP	xG	xGA	xGD	xGD/90	Last 5	Attendance	Top Team Scorer	Goalkeeper	Notes
0	1	Arsenal	28	20	4	4	70	24	46	64	2.29	56.6	19.4	37.2	1.33	W W W W W	60213	Bukayo Saka - 13	David Raya	NaN
1	2	Liverpool	28	19	7	2	65	26	39	64	2.29	62.6	34.5	28.1	1.00	W W W W D	54519	Mohamed Salah - 15	Alisson	NaN
2	3	Manchester City	28	19	6	3	63	28	35	63	2.25	56.6	27.6	29.0	1.04	D W W W D	52996	Erling Haaland - 18	Ederson	NaN
3	4	Aston Villa	29	17	5	7	60	42	18	56	1.93	51.7	41.8	9.8	0.34	W W W L D	41698	Ollie Watkins - 16	Emiliano Martínez	NaN
4	5	Tottenham	28	16	5	7	59	42	17	53	1.89	49.4	46.4	2.9	0.10	W L W W L	61555	Son Heung-min - 14	Guglielmo Vicario	NaN

Regular season

▲ promoted , ▼ relegated , Cup , Qualifier See Rank Key Glossary

Overall | Home/Away

Rk ▲	Squad	MP	W	D	L	GF	GA	GD	Pts	Pts/MP	xG	xGA	xGD	xGD/90	Last 5	Attendance	Top Team Scorer	Goalkeeper	Notes
1	Arsenal	28	20	4	4	70	24	+46	64	2.29	56.6	19.4	+37.2	+1.33	W W W W W	60,213	Bukayo Saka - 13	David Raya	
2	Liverpool	28	19	7	2	65	26	+39	64	2.29	62.6	34.5	+28.1	+1.00	W W W W D	54,519	Mohamed Salah - 15	Alisson	
3	Manchester City	28	19	6	3	63	28	+35	63	2.25	56.6	27.6	+29.0	+1.04	D W W W D	52,996	Erling Haaland - 18	Ederson	
4	Aston Villa	29	17	5	7	60	42	+18	56	1.93	51.7	41.8	+9.8	+0.34	W W W L D	41,698	Ollie Watkins - 16	Emiliano Martínez	
5	Tottenham	28	16	5	7	59	42	+17	53	1.89	49.4	46.4	+2.9	+0.10	W L W W L	61,555	Son Heung-min - 14	Guglielmo Vicario	

DATA COLLECTION

Now, let's extract specific metrics from that table based on our needs. In this case, we're interested in the following metrics: Goals For (GF), Goals Against (GA), Goal Difference (GD), Expected Goals (xG), Expected Goals Against (XGA), and Expected Goal Difference (XGD).

```
club_data1 = table_data[0].iloc[:, [1,6,7,8,11,12,13]]
club_data1
```

	Squad	GF	GA	GD	xG	xGA	xGD
0	Arsenal	70	24	46	56.6	19.4	37.2
1	Liverpool	65	26	39	62.6	34.5	28.1

Next, let's retrieve another table, which is the third table containing squad standar stats. We'll extract only a few selected metrics: Age, Possession (Poss), Assists (Ast), Yellow Cards (CrdY), Red Cards (CrdR), Expected Assists (xAG), Progressive Carries (PrgC), and Progressive Passes (PrgP). For this table, the headers are not tidy due to multi-level headers. So, we need to clean them up.

```
club_data2 = table_data[2].iloc[:, [0,2,3,9,14,15,18,20,21]]
club_data2
```

	Unnamed: 0_level_0	Unnamed: 2_level_0	Unnamed: 3_level_0	Performance			Expected	Progression	
	Squad	Age	Poss	Ast	CrdY	CrdR	xAG	PrgC	PrgP
0	Arsenal	25.6	61.5	47	43	2	39.4	628	1638
1	Aston Villa	27.6	55.2	42	76	2	38.1	629	1206
2	Bournemouth	26.4	44.9	30	57	2	30.8	523	1004

DATA COLLECTION

Let’s fix the header

```
club_data2.columns = club_data2.columns.map(lambda x: x[1].split(',')[0].replace(" ", ""))
club_data2
```

	Squad	Age	Poss	Ast	CrdY	CrdR	xAG	PrgC	PrgP
0	Arsenal	25.6	61.5	47	43	2	39.4	628	1638
1	Aston Villa	27.6	55.2	42	76	2	38.1	629	1206
2	Bournemouth	26.4	44.9	30	57	2	30.8	523	1004
3	Brentford	27.4	44.6	25	69	2	31.6	359	983

Let's continue by retrieving another table from the Squad Goalkeeping. We will only extract the Clean Sheets (CS) metric from this table.

```
club_data3 = table_data[4].iloc[:, [0,14]]
club_data3.columns = club_data3.columns.map(lambda x: x[1].split(',')[0].replace(" ", ""))
club_data3
```

	Squad	CS
0	Arsenal	11
1	Aston Villa	6
2	Bournemouth	6
3	Brentford	4

DATA COLLECTION

Let's proceed to extract the Total Shots (Sh) and Shots on Target (SoT) metrics from the squad shooting table.

```
club_data4 = table_data[8].iloc[:, [0,4,5]]
club_data4.columns = club_data4.columns.map(lambda x: x[1].split(',')[0].replace(" ", ""))
club_data4
```

	Squad	Sh	SoT
0	Arsenal	470	153
1	Aston Villa	417	152
2	Bournemouth	398	132
3	Brentford	358	120

Last, extract the Pass Accuracy (Cmp%) from the squad passing table.

```
club_data5 = table_data[10].iloc[:, [0,5]]
club_data5.columns = club_data5.columns.map(lambda x: x[1].split(',')[0].replace(" ", ""))
club_data5
```

	Squad	Cmp%
0	Arsenal	85.0
1	Aston Villa	83.5
2	Bournemouth	75.1

DATA COLLECTION

Now, merge all of our data, and the data collection process is complete.

```
club_data = pd.merge(pd.merge(pd.merge(pd.merge(club_data1, club_data2, on='Squad'),
                                       club_data3, on='Squad'), club_data4, on='Squad'), club_data5, on='Squad')
club_data.head(10)
```

	Squad	GF	GA	GD	xG	xGA	xGD	Age	Poss	Ast	CrdY	CrdR	xAG	PrgC	PrgP	CS	Sh	SoT	Cmp%
0	Arsenal	70	24	46	56.6	19.4	37.2	25.6	61.5	47	43	2	39.4	628	1638	11	470	153	85.0
1	Liverpool	65	26	39	62.6	34.5	28.1	27.1	60.1	48	53	5	45.5	658	1463	9	534	181	83.0
2	Manchester City	63	28	35	56.6	27.6	29.0	27.3	65.0	47	47	2	43.5	853	1503	8	499	187	88.2
3	Aston Villa	60	42	18	51.7	41.8	9.8	27.6	55.2	42	76	2	38.1	629	1206	6	417	152	83.5
4	Tottenham	59	42	17	49.4	46.4	2.9	25.9	60.6	47	67	4	42.2	700	1537	6	431	159	85.2
5	Manchester Utd	39	39	0	43.3	47.4	-4.2	27.1	50.4	26	67	1	30.9	555	1118	8	397	128	80.4
6	West Ham	46	50	-4	41.0	51.0	-10.0	28.9	41.4	32	70	3	26.9	434	926	5	344	110	76.4
7	Brighton	50	44	6	45.5	38.7	6.8	26.7	62.3	35	70	3	33.3	637	1402	4	414	160	87.4
8	Wolves	42	44	-2	37.6	47.9	-10.3	27.5	47.9	32	70	3	26.6	530	913	5	327	116	79.9
9	Newcastle Utd	59	48	11	51.1	49.2	2.0	28.0	52.9	38	59	0	33.4	507	1128	8	373	135	81.8

DATA PROCESSING

To create a radar chart, we need to specify the two teams. In this case, we will choose Arsenal and Manchester City.

```
team_1 = "Arsenal"
team_2 = "Manchester City"

selected_club_data = club_data[(club_data['Squad']==team_1) | (club_data['Squad']==team_2)].reset_index()
selected_club_data
```

	index	Squad	GF	GA	GD	xG	xGA	xGD	Age	Poss	Ast	CrdY	CrdR	xAG	PrgC	PrgP	CS	Sh	SoT	Cmp%
0	0	Arsenal	70	24	46	56.6	19.4	37.2	25.6	61.5	47	43	2	39.4	628	1638	11	470	153	85.0
1	2	Manchester City	63	28	35	56.6	27.6	29.0	27.3	65.0	47	47	2	43.5	853	1503	8	499	187	88.2

And then we can also select the metrics to compare. In this case, we'll choose: Goals For (GF), Goals Against (GA), Expected Goals (xG), Expected Goals Against (xGA), Assists (Ast), Expected Assists (xAG), Possession (Poss), Completion Percentage (Cmp%), Clean Sheets (CS), Shots (Sh), and Shots on Target (SoT).

```
selected_metrics = ['GF','GA','xG','xGA','Ast','xAG','Poss','Cmp%','CS','Sh','SoT']
metrics = ['Squad']+selected_metrics

selected_club_data = selected_club_data[metrics]
selected_club_data
```

	Squad	GF	GA	xG	xGA	Ast	xAG	Poss	Cmp%	CS	Sh	SoT
0	Arsenal	70	24	56.6	19.4	47	39.4	61.5	85.0	11	470	153
1	Manchester City	63	28	56.6	27.6	47	43.5	65.0	88.2	8	499	187

DATA PROCESSING

Next, we need to set the minimum and maximum ranges values for our radar chart for each metric. These values will be adjusted by adding and subtracting 25% from each metric's minimum and maximum values.

```
ranges = []
```

```
for x in selected_metrics:
    a = min(selected_club_data[selected_metrics][x])
    a = int(a - (a*.25))

    b = max(selected_club_data[selected_metrics][x])
    b = int(b + (b*.25))

    ranges.append((a,b))

ranges
```

```
[(47, 87),
 (18, 35),
 (42, 70),
 (14, 34),
 (35, 58),
 (29, 54),
 (46, 81),
 (63, 110),
 (6, 13),
 (352, 623),
 (114, 233)]
```

DATA PROCESSING

Next, we need to create a list of metric values for the two teams.

```
for x in range(len(selected_club_data['Squad'])):
    if selected_club_data['Squad'][x] == team_1:
        a_values = selected_club_data.iloc[x].values.tolist()
    if selected_club_data['Squad'][x] == team_2:
        b_values = selected_club_data.iloc[x].values.tolist()
```

```
a_values = a_values[1:]
b_values = b_values[1:]
values = [a_values,b_values]
values
```

```
[[70, 24, 56.6, 19.4, 47, 39.4, 61.5, 85.0, 11, 470, 153],
 [63, 28, 56.6, 27.6, 47, 43.5, 65.0, 88.2, 8, 499, 187]]
```

DATA VISUALIZATION

Now, we can generate the visualization with the soccerplots.radar_chart. We have the flexibility to change the color scheme for the two teams, adjust the font settings, and incorporate an endnote for added context.

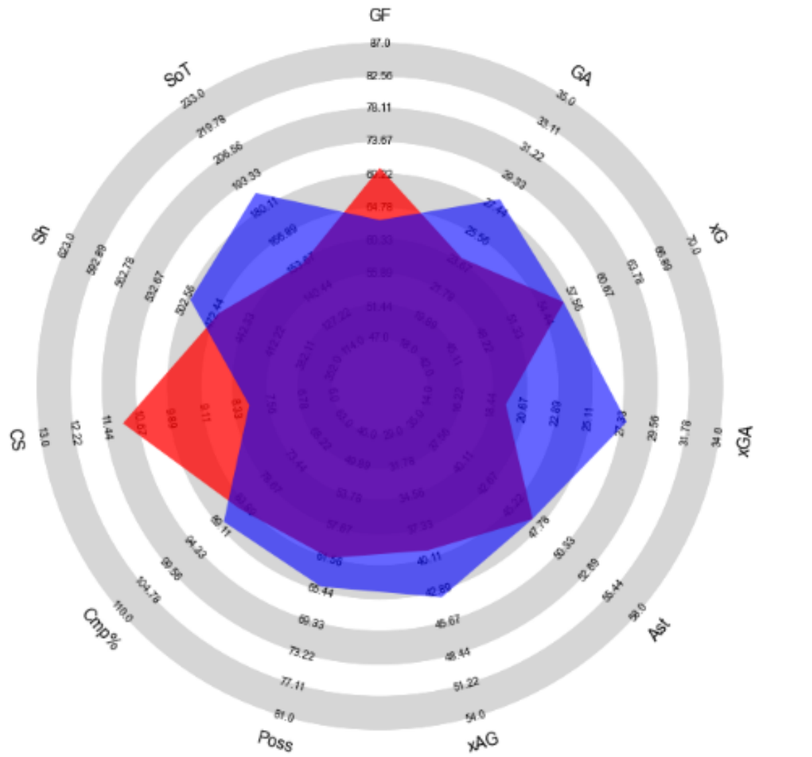
```
color_1 = "red"
color_2 = "blue"


title = dict(
    title_name= team_1,
    title_color = color_1,
    title_name_2=team_2,
    title_color_2 = color_2,
    title_fontsize = 18,
    subtitle_fontsize=15
)

endnote = 'Soccerplots - Data via FBREF'
```

```
from soccerplots.radar_chart import Radar

radar = Radar(fontfamily="Arial")
fig,ax = radar.plot_radar(ranges=ranges,params=selected_metrics,values=values,
                          radar_color=[color_1,color_2],
                          alphas=[.75,.6],title=title,endnote=endnote,
                          compare=True)
```

DATA VISUALIZATION

Results

Arsenal **Manchester City**



Inspired By: Statsbomb / Rami Moghadam
Soccerplots - Data via FBREF

As we can see from the chart, this season Manchester City's overall performance is still better than Arsenal's. Manchester City excels in nearly all metrics, including ball possession, passing accuracy, total shots, shots on target, expected assists, and total assists.

But surprisingly, Arsenal manages to outperform Manchester City with Erling Haaland in terms of goals scored, and Arsenal also has a stronger defensive line with fewer goals conceded and more clean sheet.

So the prediction for tonight's match is that Manchester City will still dominate in ball possession and control the game because the match is also played at the Etihad, but they may encounter difficulty breaking through Arsenal's defense. The victory is likely to be determined by the team that can convert their chances more effectively.

BUILD WEB APP

Now, let’s build the web app in streamlit

```
import streamlit as st
from streamlit_option_menu import option_menu


st.markdown('''
<div style="text-align: center;">
    <span style="font-size:3.3em; font-weight: bold;">
        <strong>ENGLISH PREMIER LEAGUE</strong>
    </span>
</div>
''', unsafe_allow_html=True)


with st.container():
    selected = option_menu(
    menu_title = None,
    options = ['CLUB COMPARISON'],
    icons=['ball'],
    orientation="horizontal",
    styles={
        "nav-link-selected": {"background-color": "#3f4459"},
    }
    )
```

First, let's create the title using st.markdown and an option menu for club comparison using streamlit_option_menu.

We creating the option menu because we will also create another menu for player comparison on the next update .

BUILD WEB APP

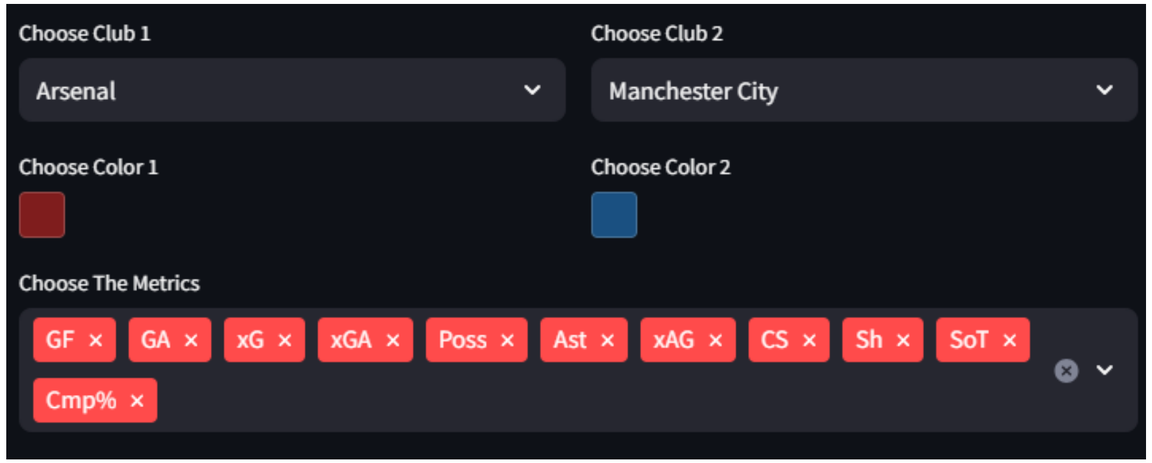
Then, if the user selects 'Club Comparison' it will scrape all the data from FBREF, similar to the previous process.

```
if selected=="CLUB COMPARISON":
    with st.container():
        import requests
        import warnings
        warnings.filterwarnings("ignore")
        url = "https://fbref.com/en/comps/9/Premier-League-Stats"
        data_url = requests.get(url)
        html_text = data_url.text
        import pandas as pd
        table_data = pd.read_html(html_text)
        club_data1 = table_data[0].iloc[:, [1,6,7,8,11,12,13]]
        club_data2 = table_data[2].iloc[:, [0,2,3,9,14,15,18,20,21]]
        club_data2.columns = club_data2.columns.map(lambda x: x[1].split(',')[0].replace(" ", ""))
        club_data3 = table_data[4].iloc[:, [0,14]]
        club_data3.columns = club_data3.columns.map(lambda x: x[1].split(',')[0].replace(" ", ""))
        club_data4 = table_data[8].iloc[:, [0,4,5]]
        club_data4.columns = club_data4.columns.map(lambda x: x[1].split(',')[0].replace(" ", ""))
        club_data5 = table_data[10].iloc[:, [0,5]]
        club_data5.columns = club_data5.columns.map(lambda x: x[1].split(',')[0].replace(" ", ""))
        club_data = pd.merge(pd.merge(pd.merge(pd.merge(club_data1, club_data2, on='Squad'),
                                    club_data3, on='Squad'), club_data4, on='Squad'), club_data5, on='Squad')
        squads_list = club_data['Squad'].tolist()
        metrics_list = club_data.columns[1:].tolist()
```

BUILD WEB APP

Then, we create a dropdown select box using st.selectbox so the user can choose Club 1 and Club 2. We also add a color picker using st.color_picker and a multiselect form using st.multiselect so the user can select several metrics.

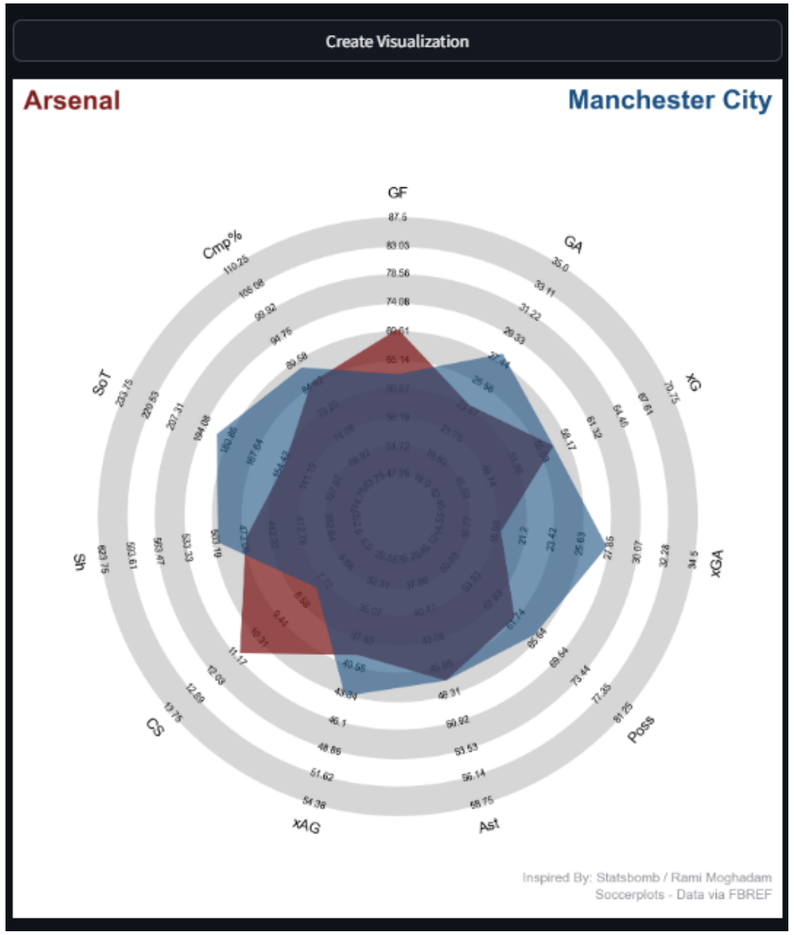
```
        col1,col2 = st.columns([3,3])
        with col1:
            club1 = st.selectbox('Choose Club 1',squads_list)
            color1 = st.color_picker('Choose Color 1', '#3f4459')
        with col2:
            club2 = st.selectbox('Choose Club 2',squads_list)
            color2 = st.color_picker('Choose Color 2', '#3f4459')
    selected_metrics = st.multiselect(
                'Choose The Metrics',metrics_list)
```



BUILD WEB APP

Lastly, we add a 'Create Visualization' button using st.button. After the user presses this button, it will continue the process to create a radar chart using soccerplots. Full code available on my Github.

```
st.markdown(
"""
<style>
.stButton>button {
    width: 100%;
}
</style>
""",
unsafe_allow_html=True,
)

if st.button("Create Visualization"):
    selected_club_data = club_data[(club_data['Squad']==club1) | (club_data['Squad']==club2)].reset_index()
    metrics = ['Squad']+selected_metrics
    selected_club_data = selected_club_data[metrics]
    ranges = []
    a_values = []
    b_values = []
    for x in selected_metrics:
        a = min(selected_club_data[selected_metrics][x])
        a = a - (a*.25)

        b = max(selected_club_data[selected_metrics][x])
        b = b + (b*.25)

        ranges.append((a,b))
    for x in range(len(selected_club_data['Squad'])):
        if selected_club_data['Squad'][x] == club1:
            a_values = selected_club_data.iloc[x].values.tolist()
        if selected_club_data['Squad'][x] == club2:
            b_values = selected_club_data.iloc[x].values.tolist()

    a_values = a_values[1:]
    b_values = b_values[1:]
    values = [a_values,b_values]

    title = dict(
        title_name= club1,
        title_color = color1,
        title_name_2= club2,
        title_color_2 = color2,
        title_fontsize = 18,
        subtitle_fontsize=15
    )
    endnote = 'Soccerplots - Data via FBREF'

    from soccerplots.radar_chart import Radar
    radar = Radar(fontfamily="Arial")
    fig,ax = radar.plot_radar(ranges=ranges,params=selected_metrics,values=values,
                              radar_color=[color1,color2],
                              alphas=[.75,.6],title=title,endnote=endnote,
                              compare=True)
    st.pyplot(fig)
```



Thank you for taking the time to read through!

Feel free to access the web app by clicking here.
If you're interested in exploring the full code, you can find it on my GitHub profile.
Let's connect on LinkedIn as well!