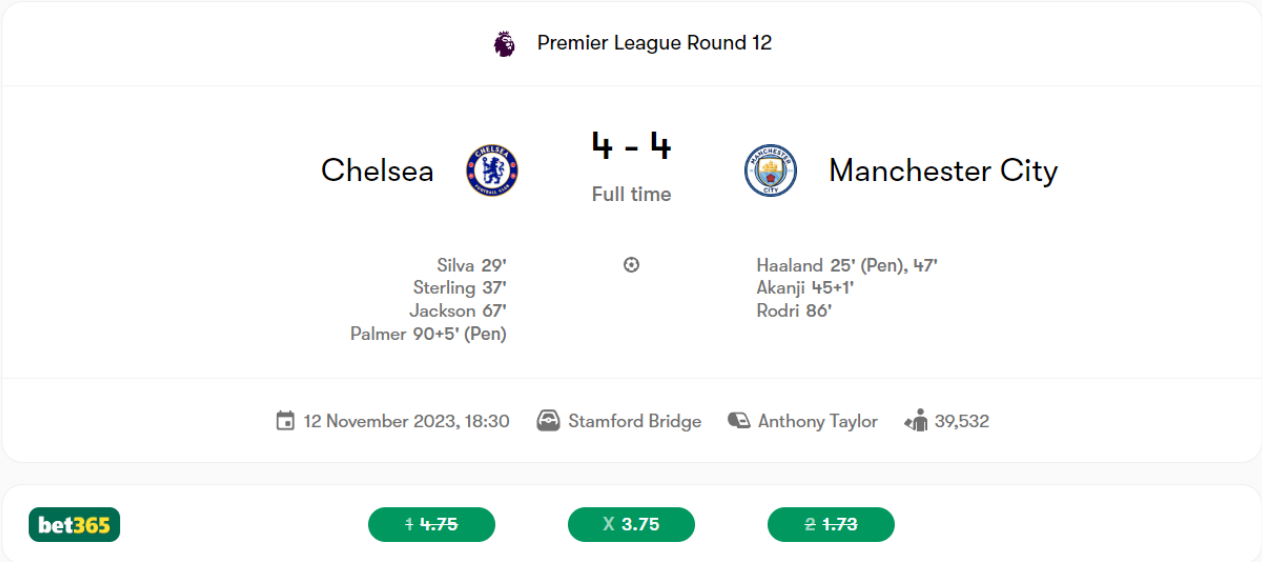
Chelsea - ManCity

Match report

In [ ]:
```
from IPython.display import Image, display
display(Image(filename='img/cover.png', embed=True))
```

Result

In [ ]:
```
display(Image(filename='img/result.png', embed=True))
```



Standings before the game

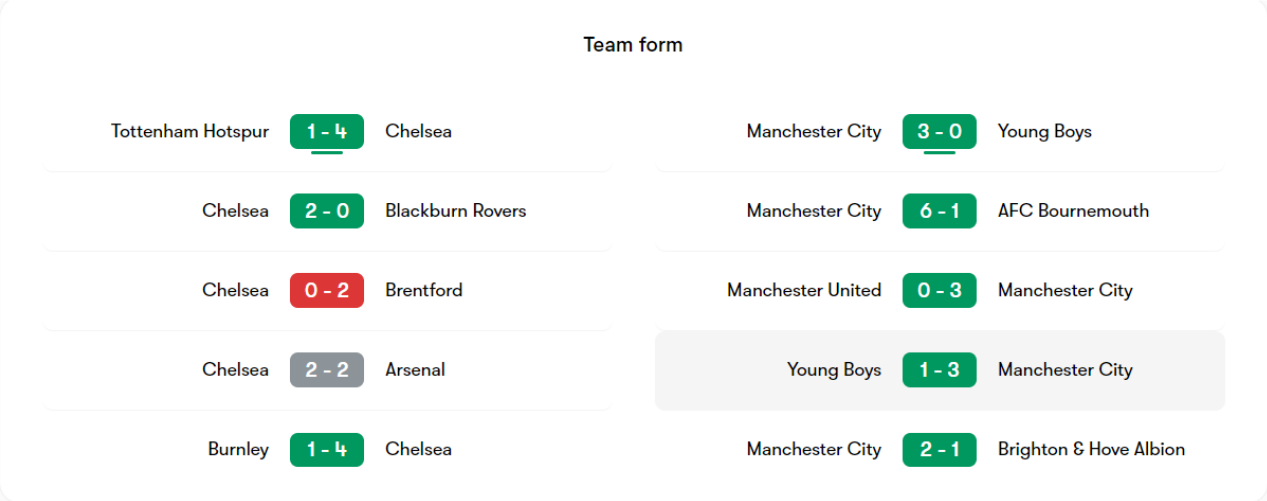
In [ ]:
```
display(Image(filename='img/standings.png', embed=True))
```

Premier League
England

#		PL	GD	PTS
1	Liverpool	21	29	48
2	Man City	20	25	43
3	Arsenal	21	22	43
4	Aston Villa	21	16	43
5	Tottenham	21	13	40
6	West Ham	21	3	35
7	Brighton	21	5	32
8	Man United	21	-5	32
9	Chelsea	21	4	31
10	Newcastle	21	9	29
11	Wolves	21	-1	29
12	Bournemouth	20	-11	25

Team form

In [ ]:
```
display(Image(filename='img/team_form.png', embed=True))
```



In [ ]:
```
%pip install pandas
```

```
Requirement already satisfied: pandas in /usr/local/lib/python3.11/site-packages (1.5.3)
Requirement already satisfied: python-dateutil>=2.8.1 in /Users/ivn/Library/Python/3.11/lib/python/site-packages (from pa
ndas) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.11/site-packages (from pandas) (2023.3.post1)
Requirement already satisfied: numpy>=1.21.0 in /usr/local/lib/python3.11/site-packages (from pandas) (1.25.0)
Requirement already satisfied: six>=1.5 in /Users/ivn/Library/Python/3.11/lib/python/site-packages (from python-dateutil>
=2.8.1->pandas) (1.16.0)

[notice] A new release of pip is available: 23.3.1 -> 23.3.2
[notice] To update, run: python3.11 -m pip install --upgrade pip
Note: you may need to restart the kernel to use updated packages.
```

In [ ]:
```
%pip install soccerbars
```

```
Collecting soccerbars
  Downloading soccerbars-0.2.1-py3-none-any.whl.metadata (10 kB)
Requirement already satisfied: matplotlib<4.0.0,>=3.7.1 in /usr/local/lib/python3.11/site-packages (from soccerbars) (3.
8.2)
Requirement already satisfied: numpy<2.0.0,>=1.24.0 in /usr/local/lib/python3.11/site-packages (from soccerbars) (1.25.0)
Requirement already satisfied: contourpy>=1.0.1 in /usr/local/lib/python3.11/site-packages (from matplotlib<4.0.0,>=3.7.1
->soccerbars) (1.2.0)
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.11/site-packages (from matplotlib<4.0.0,>=3.7.1->so
```

```
ccerbars) (0.12.1)
Requirement already satisfied: fonttools>=4.22.0 in /usr/local/lib/python3.11/site-packages (from matplotlib<4.0.0,>=3.7.
1->soccerbars) (4.47.0)
Requirement already satisfied: kiwisolver>=1.3.1 in /usr/local/lib/python3.11/site-packages (from matplotlib<4.0.0,>=3.7.
1->soccerbars) (1.4.5)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.11/site-packages (from matplotlib<4.0.0,>=3.7.1-
>soccerbars) (23.2)
Requirement already satisfied: pillow>=8 in /usr/local/lib/python3.11/site-packages (from matplotlib<4.0.0,>=3.7.1->socce
rbars) (10.1.0)
Requirement already satisfied: pyparsing>=2.3.1 in /usr/local/lib/python3.11/site-packages (from matplotlib<4.0.0,>=3.7.1
->soccerbars) (3.1.1)
Requirement already satisfied: python-dateutil>=2.7 in /Users/ivn/Library/Python/3.11/lib/python/site-packages (from matp
lotlib<4.0.0,>=3.7.1->soccerbars) (2.8.2)
Requirement already satisfied: six>=1.5 in /Users/ivn/Library/Python/3.11/lib/python/site-packages (from python-dateutil>
=2.7->matplotlib<4.0.0,>=3.7.1->soccerbars) (1.16.0)
Downloading soccerbars-0.2.1-py3-none-any.whl (14 kB)
Installing collected packages: soccerbars
Successfully installed soccerbars-0.2.1

[notice] A new release of pip is available: 23.3.1 -> 23.3.2
[notice] To update, run: python3.11 -m pip install --upgrade pip
Note: you may need to restart the kernel to use updated packages.
```

In [ ]:
```
import pandas as pd
from soccerbars import soccerbar
```

In [ ]:
```
#df_team_forms = pd.read_csv('../data/FBRef/matches_season_2023-2024.csv')

df_team_forms = pd.read_csv('data/FBRef/matches_season_2023-2024.csv')

df_team_forms = df_team_forms.drop(columns=['Link', 'Home Team ID', 'Away Team ID', 'Home xG', 'Away xG', 'Home npxG', 'Away
df_team_forms = df_team_forms.loc[df_team_forms['Date'] < '2023-11-12']
mcity = df_team_forms.loc[df_team_forms['Away Team'].isin(['Manchester City']) | df_team_forms['Home Team'].isin(['Mancheste
mcity
```

Out[ ]:

	Unnamed: 0	Date	Stage	Home Team	Away Team	Home Formation	Away Formation	Home Goals	Away Goals
0	0	2023-08-11	Matchweek 1	Burnley	Manchester City	5-4-1	4-2-3-1	0	3
15	15	2023-08-19	Matchweek 2	Manchester City	Newcastle United	4-2-3-1	4-3-3	1	0
26	26	2023-08-27	Matchweek 3	Sheffield United	Manchester City	3-5-2	4-2-3-1	1	2
35	35	2023-09-02	Matchweek 4	Manchester City	Fulham	4-2-3-1	4-3-3	5	1
42	42	2023-09-16	Matchweek 5	West Ham United	Manchester City	4-2-3-1	4-2-3-1	1	3
49	49	2023-09-23	Matchweek 6	Manchester City	Nottingham Forest	4-2-3-1	5-4-1	2	0
63	63	2023-09-30	Matchweek 7	Wolverhampton Wanderers	Manchester City	3-4-3	4-2-3-1	2	1
79	79	2023-10-08	Matchweek 8	Arsenal	Manchester City	4-3-3	4-3-3	1	0
82	82	2023-10-21	Matchweek 9	Manchester City	Brighton & Hove Albion	3-2-4-1	4-2-3-1	2	1
97	97	2023-10-29	Matchweek 10	Manchester United	Manchester City	4-2-3-1	4-2-3-1	0	3
104	104	2023-11-04	Matchweek 11	Manchester City	Bournemouth	3-4-3	5-4-1	6	1

In [ ]:
```
chelsea = df_team_forms.loc[df_team_forms['Away Team'].isin(['Chelsea']) | df_team_forms['Home Team'].isin(['Chelsea'])]
chelsea
```

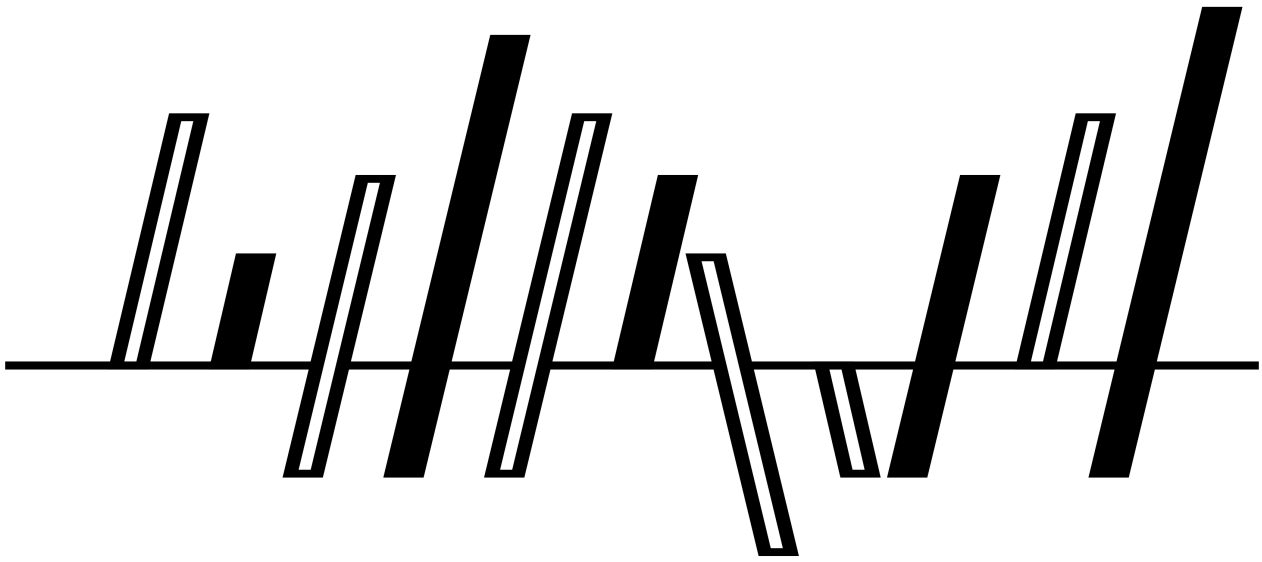
Out[ ]:

	Unnamed: 0	Date	Stage	Home Team	Away Team	Home Formation	Away Formation	Home Goals	Away Goals
8	8	2023-08-13	Matchweek 1	Chelsea	Liverpool	3-4-3	4-3-3	1	1
17	17	2023-08-20	Matchweek 2	West Ham United	Chelsea	4-2-3-1	3-4-3	3	1
19	19	2023-08-25	Matchweek 3	Chelsea	Luton Town	3-4-3	5-3-2	3	0
33	33	2023-09-02	Matchweek 4	Chelsea	Nottingham Forest	3-4-3	3-4-3	0	1
46	46	2023-09-17	Matchweek 5	Bournemouth	Chelsea	4-2-3-1	4-2-3-1	0	0
56	56	2023-09-24	Matchweek 6	Chelsea	Aston Villa	4-2-3-1	4-2-3-1	0	1
68	68	2023-10-02	Matchweek 7	Fulham	Chelsea	4-3-3	4-3-3	0	2
72	72	2023-10-07	Matchweek 8	Burnley	Chelsea	4-3-3	4-3-3	1	4
81	81	2023-10-21	Matchweek 9	Chelsea	Arsenal	4-2-3-1	4-3-3	2	2
91	91	2023-10-28	Matchweek 10	Chelsea	Brentford	4-2-3-1	3-5-2	0	2
109	109	2023-11-06	Matchweek 11	Tottenham Hotspur	Chelsea	4-2-3-1	4-2-3-1	1	4

ManCity results before the match

In [ ]:
```
isaway = []
home_goals = []
away_goals = []
```

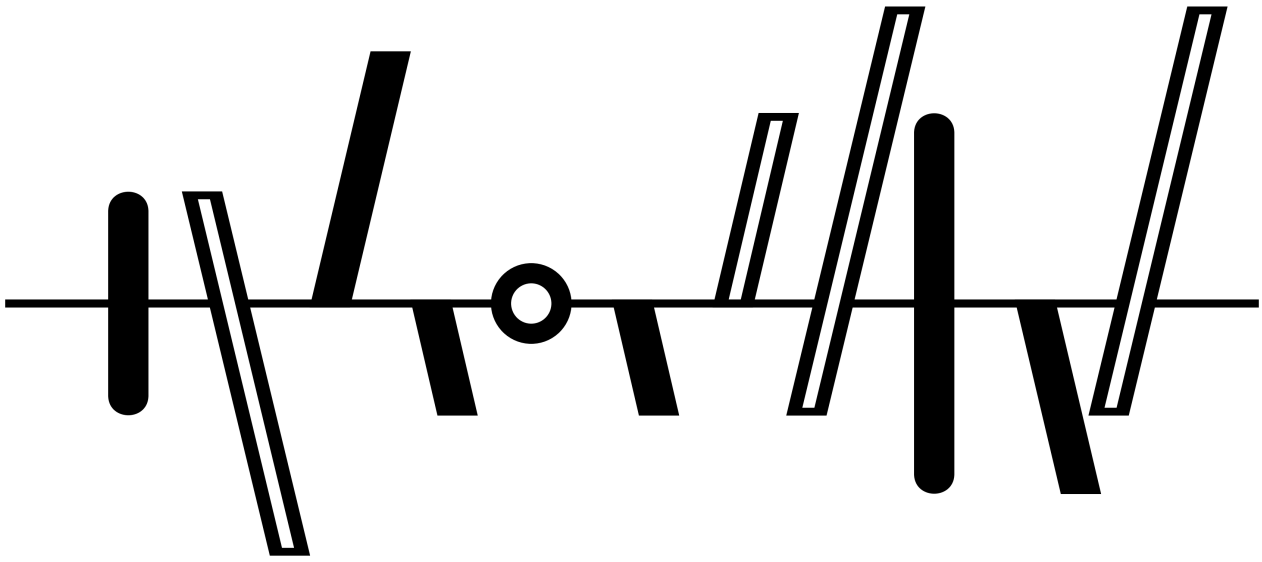
```
for index, row in mcity.iterrows():
    if row['Away Team'] == 'Manchester City':
        isaway.append(True)
    else: isaway.append(False)
    home_goals.append(row['Home Goals'])
    away_goals.append(row['Away Goals'])

soccerbar([
    home_goals,
    away_goals,
    isaway
], outlined=True)
```



Out[ ]: <Axes: >

Chelsea results before the match

In [ ]:
```
isaway = []
home_goals = []
away_goals = []
for index, row in chelsea.iterrows():
    if row['Away Team'] == 'Chelsea':
        isaway.append(True)
    else: isaway.append(False)
    home_goals.append(row['Home Goals'])
    away_goals.append(row['Away Goals'])

soccerbar([
    home_goals,
    away_goals,
    isaway
], outlined=True)
```



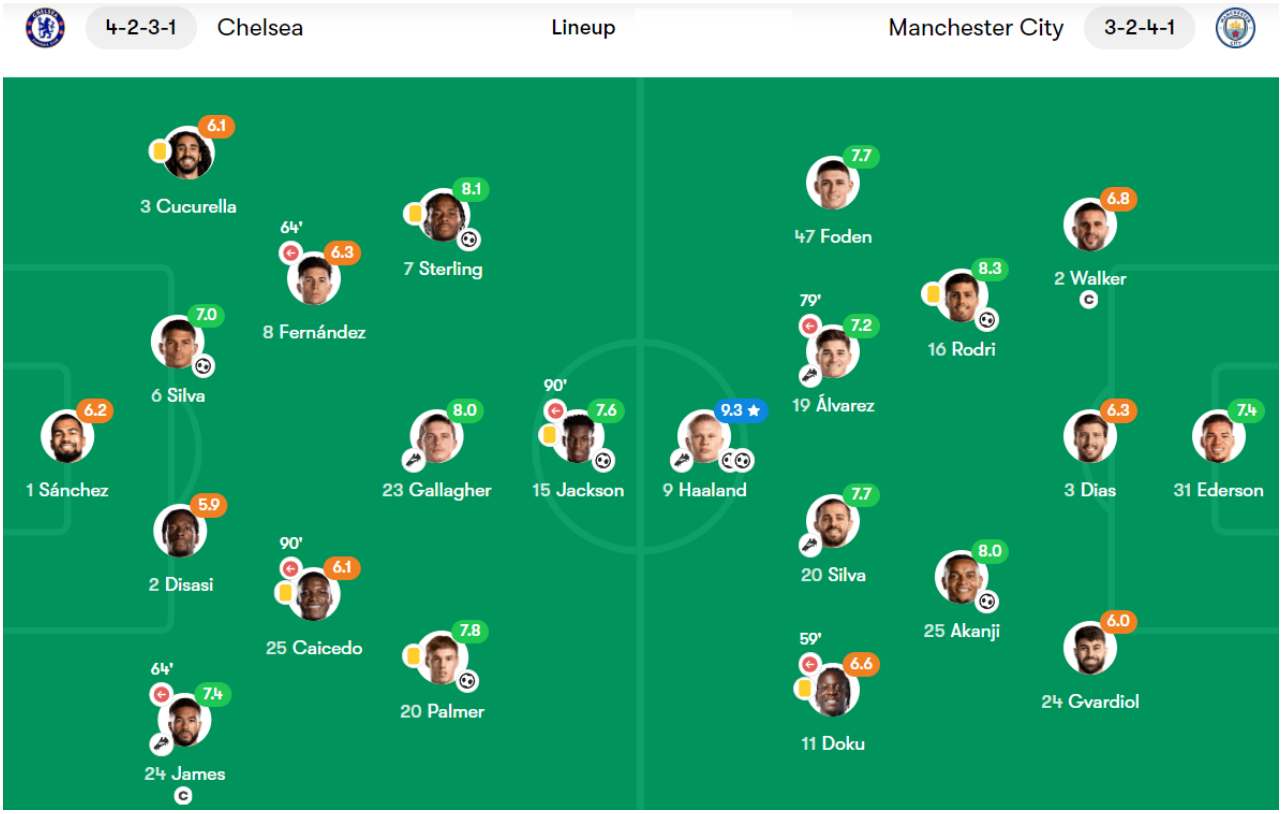
Out[ ]: <Axes: >

In [ ]:
```
display(Image(filename='img/head-to-head.png', embed=True))
```

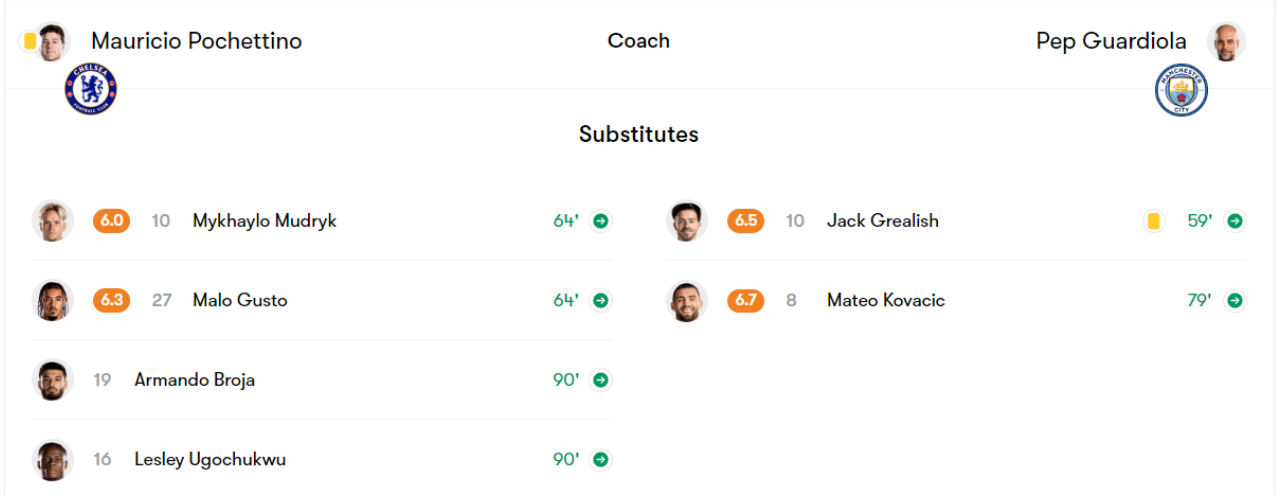
In [ ]:
```
display(Image(filename='img/historical_head-to-head.png', embed=True))
```

Line up

In [ ]:
```
display(Image(filename='img/lineup.png', embed=True))
```



In [ ]:
```
display(Image(filename='img/sabstitutes.png', embed=True))
```



In [ ]:
```
display(Image(filename='img/lineup_worth.png', embed=True))
```

4-2-3-1	3-2-4-1
Lineup worth 444.0 Mio. € (11)	**Lineup worth 794.0 Mio. € (11)**
Robert Sánchez (1)	Ederson (31)
Cucurella (3)	Manuel Akanji (25)
Axel Disasi (2)	Joško Gvardiol (24)
Reece James (24)	Rúben Dias (3)
Thiago Silva (6)	Kyle Walker (2)
Moisés Caicedo (25)	Bernardo Silva (20)
Enzo Fernández (8)	Phil Foden (47)
Conor Gallagher (23)	Rodri (16)
Cole Palmer (20)	Julián Álvarez (19)
Nicolas Jackson (15)	Jeremy Doku (11)
Raheem Sterling (7)	Erling Haaland (9)
Bench worth 194.0 Mio. € (7)	**Bench worth 199.2 Mio. € (6)**
Đorđe Petrović (28)	Scott Carson (33)
Benoît Badiashile (5)	Stefan Ortega Moreno (18)
Malo Gusto (27)	Oscar Bobb (52)
Ian Maatsen (29)	Jack Grealish (10)
Noni Madueke (11)	Mateo Kovačić (8)
Mykhailo Mudryk (10)	Matheus Nunes (27)
Chimuanya Ugochukwu (16)	Kalvin Phillips (4)
Armando Broja (19)	Rico Lewis (82)
Alex Matos (52)	

Average players positions

In [ ]:
```python
from mplsoccer.pitch import Pitch
import pandas as pd

# spadl format assumes pitch size 68 x 105,
# so we use pitch_type=`uefa` for mplsoccer
# https://socceraction.readthedocs.io/en/latest/documentation/spadl/spadl.html

def get_pitch():
  return Pitch(pitch_type='uefa',
               pitch_color='white',
               line_color='#c7d5cc')
```

In [ ]:
```python
import soccerdata as sd
import os
from pathlib import Path

# https://www.whoscored.com/Matches/1729243/Live/England-Premier-League-2023-2024-Chelsea-Manchester-City

data_dir=Path(f"{os.getcwd()}/data/soccerdata_cache")
ws = sd.WhoScored(leagues="ENG-Premier League", seasons="23-24", data_dir=data_dir)

MC_team_id = 167
chelsea_team_id = 15
GAME_ID=1729243
events_atomic_spadl = ws.read_events(match_id=GAME_ID, force_cache=True, output_fmt="atomic-spadl")
```

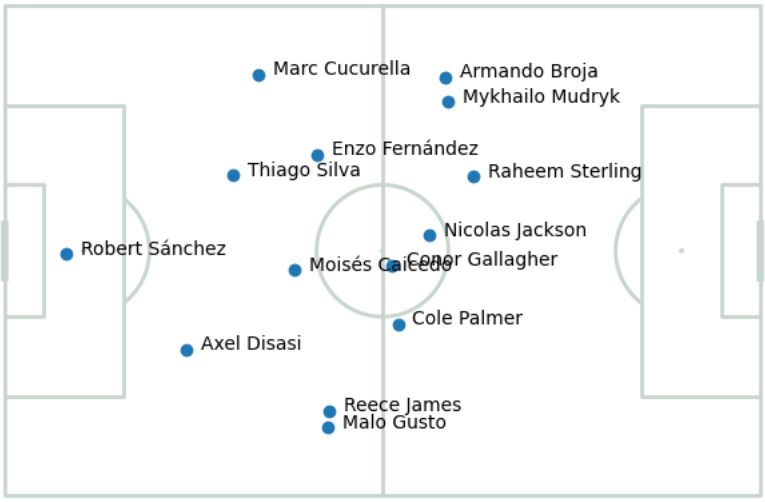
```
[01/30/24 21:19:42] INFO     Saving cached data to                                  _common.p
                             /Users/ivn/Documents/projects/fa_ucu/Manchester-City/data/soccerdata_cac
                             he
[01/30/24 21:20:01] INFO     patching driver executable /Users/ivn/Library/Application  patcher.py
                             Support/undetected_chromedriver/undetected_chromedriver
[01/30/24 21:20:04] INFO     Retrieving game schedule of ENG-Premier League - 2324 from the cache  whoscored.py
                    INFO     [1/1] Retrieving game with id=1729243                  whoscored.py
```

In [ ]:
```python
events_raw = ws.read_events(match_id=GAME_ID, force_cache=True, output_fmt="raw")
events_spadl = ws.read_events(match_id=GAME_ID, force_cache=True, output_fmt="spadl")
events_atomic_spadl = ws.read_events(match_id=GAME_ID, force_cache=True, output_fmt="atomic-spadl")
events = ws.read_events(match_id=GAME_ID, force_cache=True, output_fmt="events")
```

```
[01/30/24 21:21:33] INFO     Retrieving game schedule of ENG-Premier League - 2324 from the cache  whoscored.py
                    INFO     [1/1] Retrieving game with id=1729243                  whoscored.py
                    INFO     Retrieving game schedule of ENG-Premier League - 2324 from the cache  whoscored.py
                    INFO     [1/1] Retrieving game with id=1729243                  whoscored.py
                    INFO     Retrieving game schedule of ENG-Premier League - 2324 from the cache  whoscored.py
                    INFO     [1/1] Retrieving game with id=1729243                  whoscored.py
                    INFO     Retrieving game schedule of ENG-Premier League - 2324 from the cache  whoscored.py
                    INFO     [1/1] Retrieving game with id=1729243                  whoscored.py
```

In [ ]:
```python
# Average positions for all players (including subs!)

def avg_loc_by_team(team_id: int):
    avg_loc = events_atomic_spadl[events_atomic_spadl['team_id'] == team_id].groupby('player').agg({'x': 'mean', 'y': 'mean'
```

```
    pitch = get_pitch()

    fig, ax = pitch.draw()

    sc = pitch.scatter(x=avg_loc['x'].to_list(),
                       y=avg_loc['y'].to_list(),
                       # color=df_shot['color'].to_list(),
                       # marker='football',
                       ax=ax)

    n = avg_loc.index.to_list()
    data = sc.get_offsets()

    # leg = ax.legend(borderpad=1, markerscale=0.5, labelspacing=1.5, loc='upper center', fontsize=15)

    for idx,label in enumerate(n):
        ax.annotate(label, (data[idx][0]+2, data[idx][1]))

MC_team_id = 167
chelsea_team_id = 15

avg_loc_by_team(MC_team_id)
avg_loc_by_team(chelsea_team_id)
```



Match stats

In [ ]:
```
display(Image(filename='img/stats.png', embed=True))
```

Chelsea		Manchester City	Chelsea		Manchester City	Chelsea		Manchester City
13	Fouls	15	21	Tackles	17	1	Offsides	0
3	Corners	3	5	Interceptions	9	8	Goal Kicks	3
7	Crosses	18	3	Aerials Won	4	14	Throw Ins	18
593	Touches	697	20	Clearances	12	52	Long Balls	50

Shots

In [ ]:
```
%pip install pandas
```

```
Requirement already satisfied: pandas in /usr/local/lib/python3.11/site-packages (1.5.3)
Requirement already satisfied: python-dateutil>=2.8.1 in /Users/ivn/Library/Python/3.11/lib/python/site-packages (from pa
ndas) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.11/site-packages (from pandas) (2023.3.post1)
Requirement already satisfied: numpy>=1.21.0 in /usr/local/lib/python3.11/site-packages (from pandas) (1.25.0)
Requirement already satisfied: six>=1.5 in /Users/ivn/Library/Python/3.11/lib/python/site-packages (from python-dateutil>
=2.8.1->pandas) (1.16.0)

[notice] A new release of pip is available: 23.3.1 -> 23.3.2
[notice] To update, run: python3.11 -m pip install --upgrade pip
Note: you may need to restart the kernel to use updated packages.
```

In [ ]:
```
import pandas as pd
import matplotlib.pyplot as plt
from mplsoccer.pitch import Pitch, VerticalPitch
import seaborn as sns
```

In [ ]:
```
df_1 = pd.read_csv('data/12112023/events_1729243.csv')
```

In [ ]:
```
# define dataframe with shots and goals
shots = df_1.loc[(df_1['type'] == 'SavedShot') | (df_1['type'] == 'Goal')].set_index('id')
# define team names
team1, team2 = df_1.team.unique()
```

In [ ]:
```
# check df
shots.tail()
```

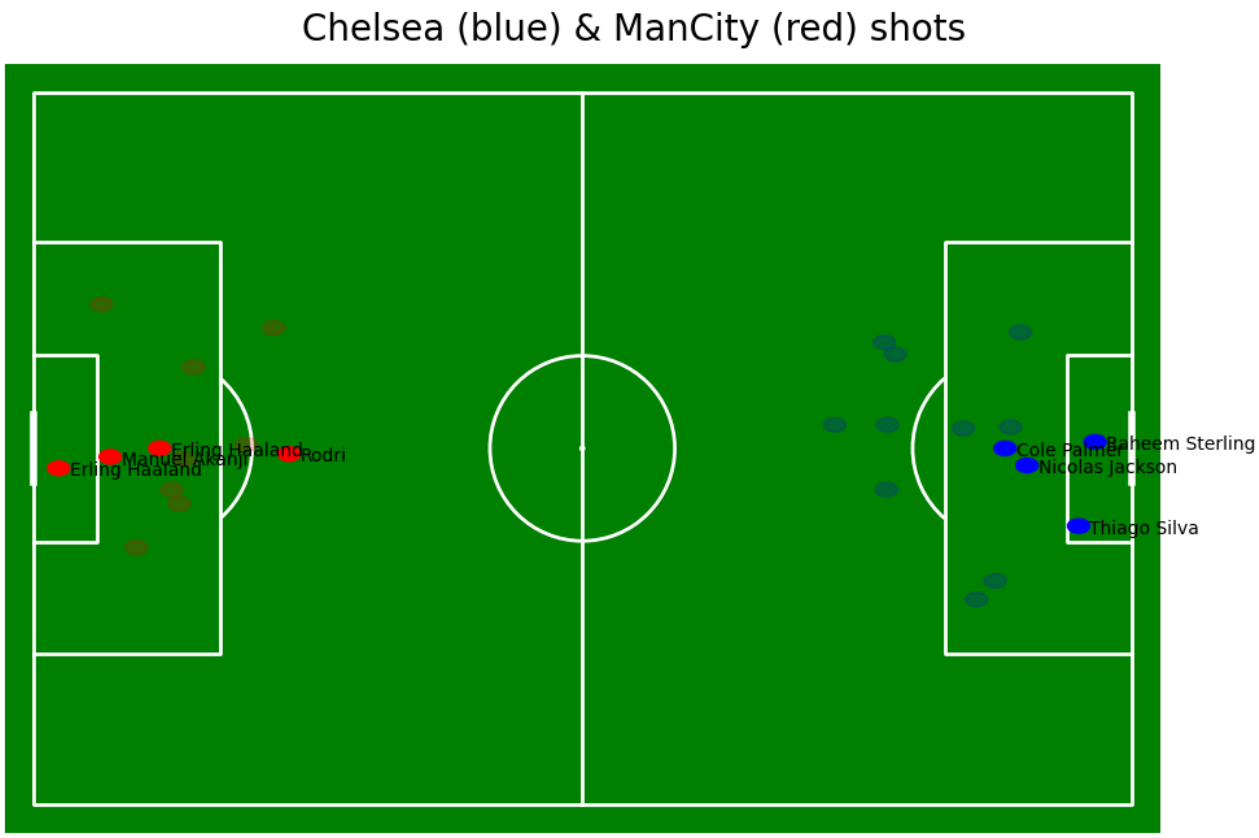
Out[ ]:

id	league	season	game	game_id	period	minute	second	expanded_minute	type	outcome_type	...	goal_mouth_z	b
2617386379	ENG-Premier League	2324	2023-11-12 Chelsea-Manchester City	1729243	SecondHalf	66	22.0	73	SavedShot	Successful	...	1.3	
2617386415	ENG-Premier League	2324	2023-11-12 Chelsea-Manchester City	1729243	SecondHalf	66	25.0	73	Goal	Successful	...	1.9	
2617404351	ENG-Premier League	2324	2023-11-12 Chelsea-Manchester City	1729243	SecondHalf	85	30.0	92	SavedShot	Successful	...	19.0	
2617404379	ENG-Premier League	2324	2023-11-12 Chelsea-Manchester City	1729243	SecondHalf	85	33.0	92	Goal	Successful	...	14.6	
2617410983	ENG-Premier League	2324	2023-11-12 Chelsea-Manchester City	1729243	SecondHalf	94	24.0	101	Goal	Successful	...	22.2	

5 rows × 29 columns

In [ ]:
```
pitch = Pitch(pitch_type='opta', line_color = 'white', pitch_color = 'green')
fig, ax = pitch.draw(figsize=(12,8))

# plot the shots by looping through them
for i,shot in shots.iterrows():
    # get the information
    x=shot['x']
    y=shot['y']
```

```
    goal=shot['type']=='Goal'
    team_name=shot['team']
    # set circlesize
    circleSize=1
    # plot Chelsea
    if (team_name==team1):
        if goal:
            shotCircle=plt.Circle((x,y),circleSize,color="blue")
            plt.text(x+1,y-1,shot['player'])
        else:
            shotCircle=plt.Circle((x,y),circleSize,color="blue")
            shotCircle.set_alpha(.2)
    # plot Man City
    else:
        if goal:
            shotCircle=plt.Circle((100-x,100 - y),circleSize,color="red") #https://mplsoccer.readthedocs.io/en/latest/galler
            plt.text(100-x+1,100 - y - 1, shot['player']) #https://mplsoccer.readthedocs.io/en/latest/gallery/pitch_setup/pl
        else:
            shotCircle=plt.Circle((100-x,100 - y),circleSize,color="red")
            shotCircle.set_alpha(.2)
    ax.add_patch(shotCircle)
    # set title
fig.suptitle("Chelsea (blue) & ManCity (red) shots",  fontsize=20, x=0.5, y=0.95)
fig.set_size_inches(10, 7)
plt.show()
```



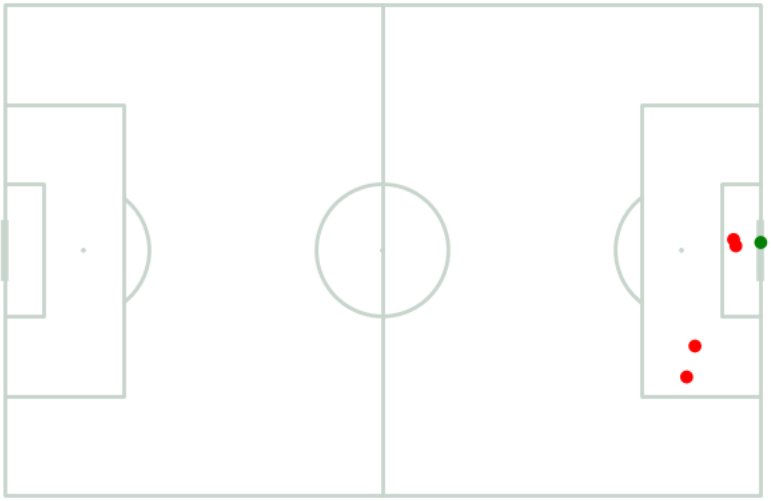
Raheem Sterling goal

In [ ]:

```
from mplsoccer.pitch import Pitch

# points that led to a goal at event #1015 (Sterling, goal #3)
shot = 1015
df_shot = events_atomic_spadl[shot-4:shot+1].copy()
df_shot['color'] = 'red'
df_shot.loc[shot, 'color'] = 'green'

pitch = get_pitch()
fig, ax = pitch.draw()

print(df_shot.loc[shot])
_ = pitch.scatter(x=df_shot['x'].to_list(),
                  y=df_shot['y'].to_list(),
                  color=df_shot['color'].to_list(),
                  ax=ax)
```

```
game_id                    1729243
original_event_id       2617331399
action_id                     1015
period_id                        1
time_seconds                2183.0
team_id                         15
player_id                   97692.0
x                             105.0
y                            35.088
dx                              0.0
dy                              0.0
type_id                          27
```

```
bodypart_id                     5
player              Raheem Sterling
team                        Chelsea
color                         green
Name: 1015, dtype: object
```



In [ ]:
```
%pip install socceraction
```

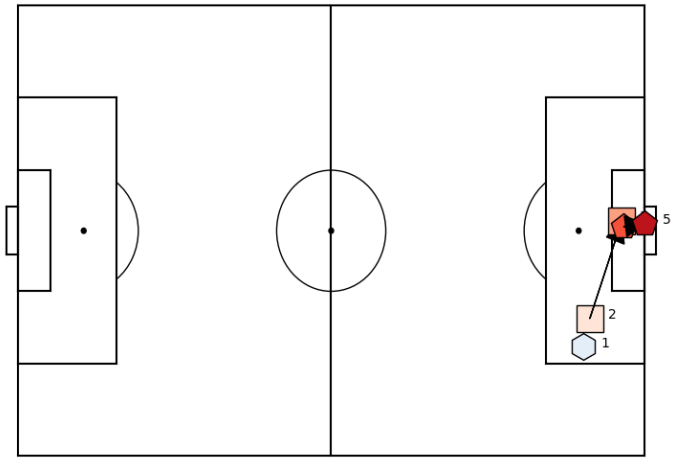
```
Requirement already satisfied: socceraction in /usr/local/lib/python3.11/site-packages (1.4.2)
Requirement already satisfied: lxml<5.0.0,>=4.6.3 in /usr/local/lib/python3.11/site-packages (from socceraction) (4.9.4)
Requirement already satisfied: numpy<2.0.0,>=1.21.2 in /usr/local/lib/python3.11/site-packages (from socceraction) (1.25.
0)
Requirement already satisfied: pandas<2.0.0,>=1.3.3 in /usr/local/lib/python3.11/site-packages (from socceraction) (1.5.
3)
Requirement already satisfied: pandera<0.14.0,>=0.13.4 in /usr/local/lib/python3.11/site-packages (from socceraction) (0.
13.4)
Requirement already satisfied: scikit-learn<2.0.0,>=1.2.1 in /usr/local/lib/python3.11/site-packages (from socceraction)
(1.3.2)
Requirement already satisfied: python-dateutil>=2.8.1 in /Users/ivn/Library/Python/3.11/lib/python/site-packages (from pa
ndas<2.0.0,>=1.3.3->socceraction) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.11/site-packages (from pandas<2.0.0,>=1.3.3->soccer
action) (2023.3.post1)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.11/site-packages (from pandera<0.14.0,>=0.13.4->
socceraction) (23.2)
Requirement already satisfied: pydantic in /usr/local/lib/python3.11/site-packages (from pandera<0.14.0,>=0.13.4->soccera
ction) (2.5.3)
Requirement already satisfied: typing-inspect>=0.6.0 in /usr/local/lib/python3.11/site-packages (from pandera<0.14.0,>=0.
13.4->socceraction) (0.9.0)
Requirement already satisfied: wrapt in /usr/local/lib/python3.11/site-packages (from pandera<0.14.0,>=0.13.4->socceracti
on) (1.16.0)
Requirement already satisfied: scipy>=1.5.0 in /usr/local/lib/python3.11/site-packages (from scikit-learn<2.0.0,>=1.2.1->
socceraction) (1.11.4)
Requirement already satisfied: joblib>=1.1.1 in /usr/local/lib/python3.11/site-packages (from scikit-learn<2.0.0,>=1.2.1-
>socceraction) (1.3.2)
Requirement already satisfied: threadpoolctl>=2.0.0 in /usr/local/lib/python3.11/site-packages (from scikit-learn<2.0.0,>
=1.2.1->socceraction) (3.2.0)
Requirement already satisfied: six>=1.5 in /Users/ivn/Library/Python/3.11/lib/python/site-packages (from python-dateutil>
=2.8.1->pandas<2.0.0,>=1.3.3->socceraction) (1.16.0)
Requirement already satisfied: mypy-extensions>=0.3.0 in /usr/local/lib/python3.11/site-packages (from typing-inspect>=0.
6.0->pandera<0.14.0,>=0.13.4->socceraction) (1.0.0)
Requirement already satisfied: typing-extensions>=3.7.4 in /usr/local/lib/python3.11/site-packages (from typing-inspect>=
0.6.0->pandera<0.14.0,>=0.13.4->socceraction) (4.9.0)
Requirement already satisfied: annotated-types>=0.4.0 in /usr/local/lib/python3.11/site-packages (from pydantic->pandera<
0.14.0,>=0.13.4->socceraction) (0.6.0)
Requirement already satisfied: pydantic-core==2.14.6 in /usr/local/lib/python3.11/site-packages (from pydantic->pandera<
0.14.0,>=0.13.4->socceraction) (2.14.6)

[notice] A new release of pip is available: 23.3.1 -> 23.3.2
[notice] To update, run: python3.11 -m pip install --upgrade pip
Note: you may need to restart the kernel to use updated packages.
```

In [ ]:
```
%pip install matplotsoccer
```

```
Requirement already satisfied: matplotsoccer in /usr/local/lib/python3.11/site-packages (0.0.8)

[notice] A new release of pip is available: 23.3.1 -> 23.3.2
[notice] To update, run: python3.11 -m pip install --upgrade pip
Note: you may need to restart the kernel to use updated packages.
```

In [ ]:
```
# Goal by Sterling
import matplotsoccer
import socceraction


a = df_shot

# Plot the actions
def nice_time(row):
    return f"{row.time_seconds}s"

a["nice_time"] = a.apply(nice_time, axis=1)
a["end_x"] = a["x"] + a["dx"]
a["end_y"] = a["y"] + a["dy"]
a["type"] = a.type_id.apply(lambda x: socceraction.atomic.spadl.config.actiontypes[x])
```

```
labels = a[["nice_time", "type", "player", "team"]]

ax = matplotsoccer.actions(
    location=a[["x", "y", "end_x", "end_y"]],
    action_type=a.type_id,
    team= a.team,
    # result= True,
    label=labels,
    labeltitle=["time", "actiontype", "player", "team"],
    zoom=False,
    figsize=10
)
```



	time	actiontype	player	team
1	2181.0s	bad_touch	Josko Gvardiol	Man City
2	2182.0s	pass	Reece James	Chelsea
3	2182.5s	receival	Raheem Sterling	Chelsea
4	2183.0s	shot	Raheem Sterling	Chelsea
5	2183.0s	goal	Raheem Sterling	Chelsea

Passes

In [ ]:
```
#players names
df_player_name = df_1.groupby(["team", 'player_id', 'player']).id.count().reset_index()
df_player_name = df_player_name.set_index('player_id')
print(df_player_name)
```

```
               team            player   id
player_id
28550.0     Chelsea      Thiago Silva   67
97692.0     Chelsea   Raheem Sterling   74
300426.0    Chelsea       Axel Disasi   88
361330.0    Chelsea       Reece James   56
362151.0    Chelsea    Robert Sánchez   60
363496.0    Chelsea    Marc Cucurella   68
369430.0    Chelsea    Enzo Fernández   59
375621.0    Chelsea   Conor Gallagher   73
393407.0    Chelsea     Armando Broja    6
395692.0    Chelsea       Cole Palmer   60
403850.0    Chelsea        Malo Gusto   34
404616.0    Chelsea  Lesley Ugochukwu    1
409376.0    Chelsea   Mykhailo Mudryk   10
410175.0    Chelsea    Moisés Caicedo   51
426050.0    Chelsea   Nicolas Jackson   28
69778.0    Man City       Kyle Walker   82
93894.0    Man City      Mateo Kovacic   19
113069.0   Man City      Jack Grealish   36
121774.0   Man City            Ederson   59
136741.0   Man City    Bernardo Silva   77
297390.0   Man City     Manuel Akanji   78
303139.0   Man City             Rodri  107
313171.0   Man City         Rúben Dias  108
315227.0   Man City    Erling Haaland   27
331254.0   Man City         Phil Foden   70
365409.0   Man City    Julián Álvarez   54
388098.0   Man City       Jérémy Doku   39
402664.0   Man City    Josko Gvardiol   90
```

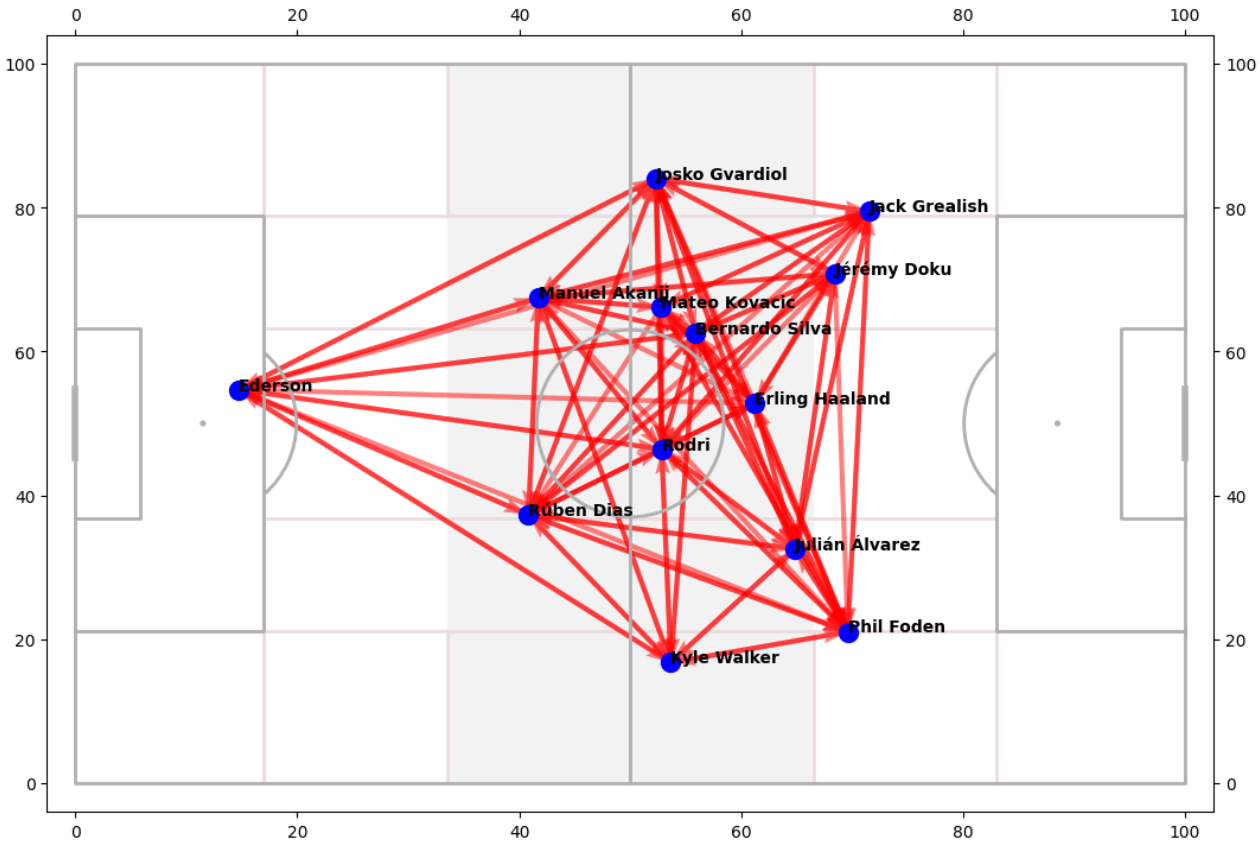
ManCity passing network

[the full match with substitutions]

In [ ]:
```
# define a passer and a pass recipient and add the columns
df_MC_pn_ext = df_1[df_1['team'] == 'Man City']
df_MC_pn_ext['passer'] = df_MC_pn_ext['player_id']
df_MC_pn_ext['recipient'] = df_MC_pn_ext['player_id'].shift(-1)

# define successful passes
MC_successful_passes_ext = df_MC_pn_ext[(df_MC_pn_ext['type'] == 'Pass')&(df_MC_pn_ext['outcome_type'] == 'Successful')]


# define an average player's location and quantity of passes
MC_average_locations_ext = MC_successful_passes_ext.groupby('passer').agg({'x': ['mean'], 'y': ['mean','count']})

MC_average_locations_ext.columns = ['x', 'y', 'count']

MC_average_locations_ext= MC_average_locations_ext.merge(df_player_name[['player']], how='left', left_on='passer', right_inc
```

```
        .rename(columns={'player': 'player_name'})

# count passes
pass_between_MC_ext = MC_successful_passes_ext.groupby(['passer', 'recipient']).id.count().reset_index()
pass_between_MC_ext.rename({'id':'pass_count'}, axis='columns', inplace=True)

pass_between_MC_ext = pass_between_MC_ext.merge(MC_average_locations_ext, left_on ='passer', right_index = True)
pass_between_MC_ext = pass_between_MC_ext.merge(MC_average_locations_ext, left_on ='recipient', right_index = True, suffixes


print(pass_between_MC_ext)

#Put the pass network on the pitch
pitch = Pitch(pitch_type='opta', positional=True, shade_middle=True, positional_color='#eadddd', shade_color='#f2f2f2', axis
fig, ax = plt.subplots(figsize=(13, 8.5))

pitch.draw(ax=ax)


arrows = pitch.arrows(pass_between_MC_ext.x, pass_between_MC_ext.y, pass_between_MC_ext.x_end, pass_between_MC_ext.y_end, ax
                      width =3, headwidth=3, color = 'red', zorder = 1, alpha = 0.5)

nodes = pitch.scatter(MC_average_locations_ext.x, MC_average_locations_ext.y, s =100, color = "blue", linewidth = 2.5, \
                      alpha =1, zorder = 1, ax=ax)

for i, row in MC_average_locations_ext.iterrows():
    pitch.annotate(row.player_name, (MC_average_locations_ext.x[i], MC_average_locations_ext.y[i]), c='black', \
                   weight = "bold", size=10, ax=ax, zorder = 4)
```

```
/var/folders/sc/p5r88fzj46n6sr9l_c6dgt540000gn/T/ipykernel_43692/1285356849.py:3: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-v
iew-versus-a-copy
  df_MC_pn_ext['passer'] = df_MC_pn_ext['player_id']
/var/folders/sc/p5r88fzj46n6sr9l_c6dgt540000gn/T/ipykernel_43692/1285356849.py:4: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-v
iew-versus-a-copy
  df_MC_pn_ext['recipient'] = df_MC_pn_ext['player_id'].shift(-1)
      passer  recipient  pass_count         x          y  count  \
0    69778.0   121774.0           6  53.659091  16.763636     44
15  113069.0   121774.0           1  71.540000  79.460000     15
34  136741.0   121774.0           2  55.897917  62.527083     48
46  297390.0   121774.0           5  41.710526  67.431579     57
57  303139.0   121774.0           3  52.832836  46.425373     67
..       ...        ...         ...        ...        ...    ...
64  303139.0   388098.0           4  52.832836  46.425373     67
75  313171.0   388098.0           2  40.831579  37.307895     76
83  315227.0   388098.0           1  61.230000  52.780000     10
101 365409.0   388098.0           2  64.851515  32.566667     33
119 402664.0   388098.0           6  52.310000  84.065000     60

        player_name  x_end  y_end  count_end player_name_end
0       Kyle Walker  14.69  54.69         40         Ederson
15    Jack Grealish  14.69  54.69         40         Ederson
34   Bernardo Silva  14.69  54.69         40         Ederson
46    Manuel Akanji  14.69  54.69         40         Ederson
57            Rodri  14.69  54.69         40         Ederson
..              ...    ...    ...        ...             ...
64            Rodri  68.44  70.83         10     Jérémy Doku
75       Rúben Dias  68.44  70.83         10     Jérémy Doku
83   Erling Haaland  68.44  70.83         10     Jérémy Doku
101  Julián Álvarez  68.44  70.83         10     Jérémy Doku
119  Josko Gvardiol  68.44  70.83         10     Jérémy Doku

[120 rows x 11 columns]
```



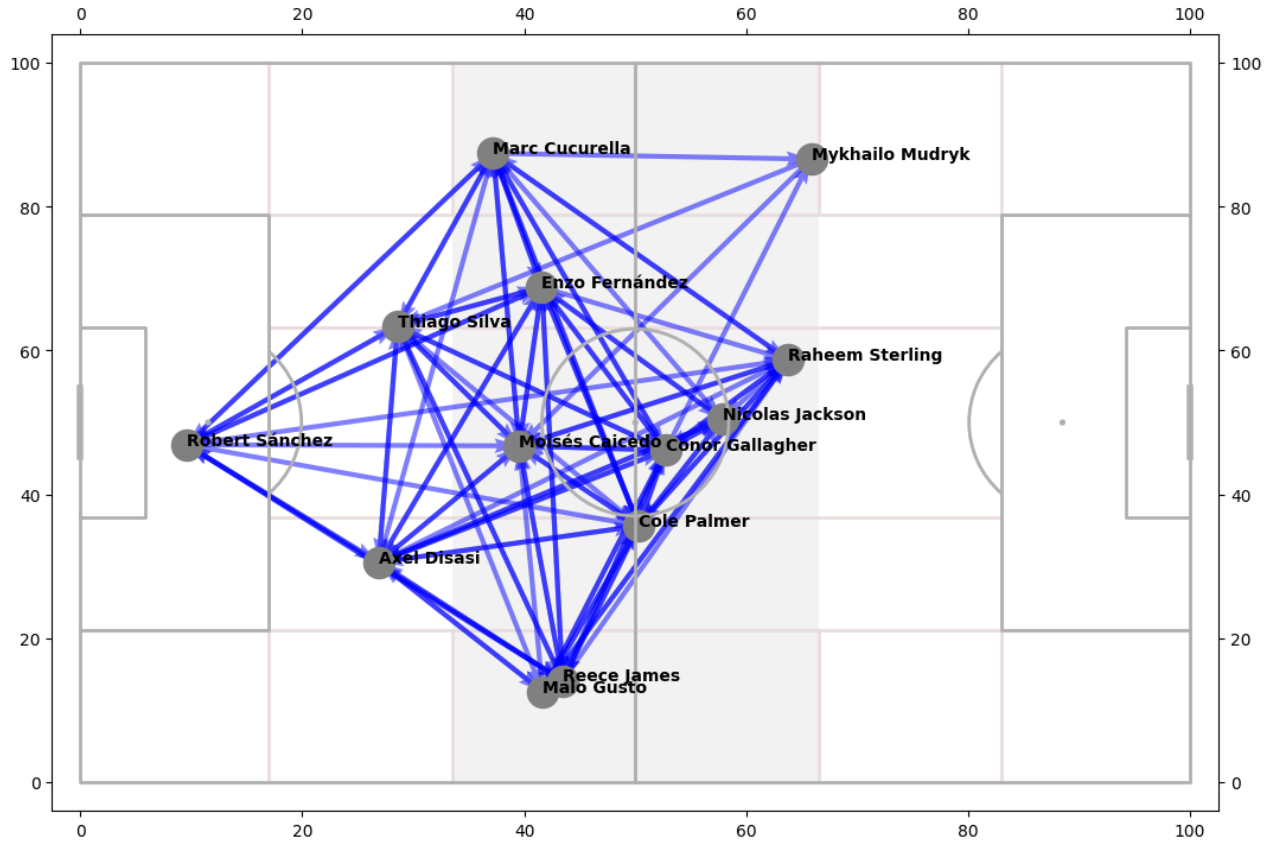
Chelsea passing network

[the full match with substitutions]

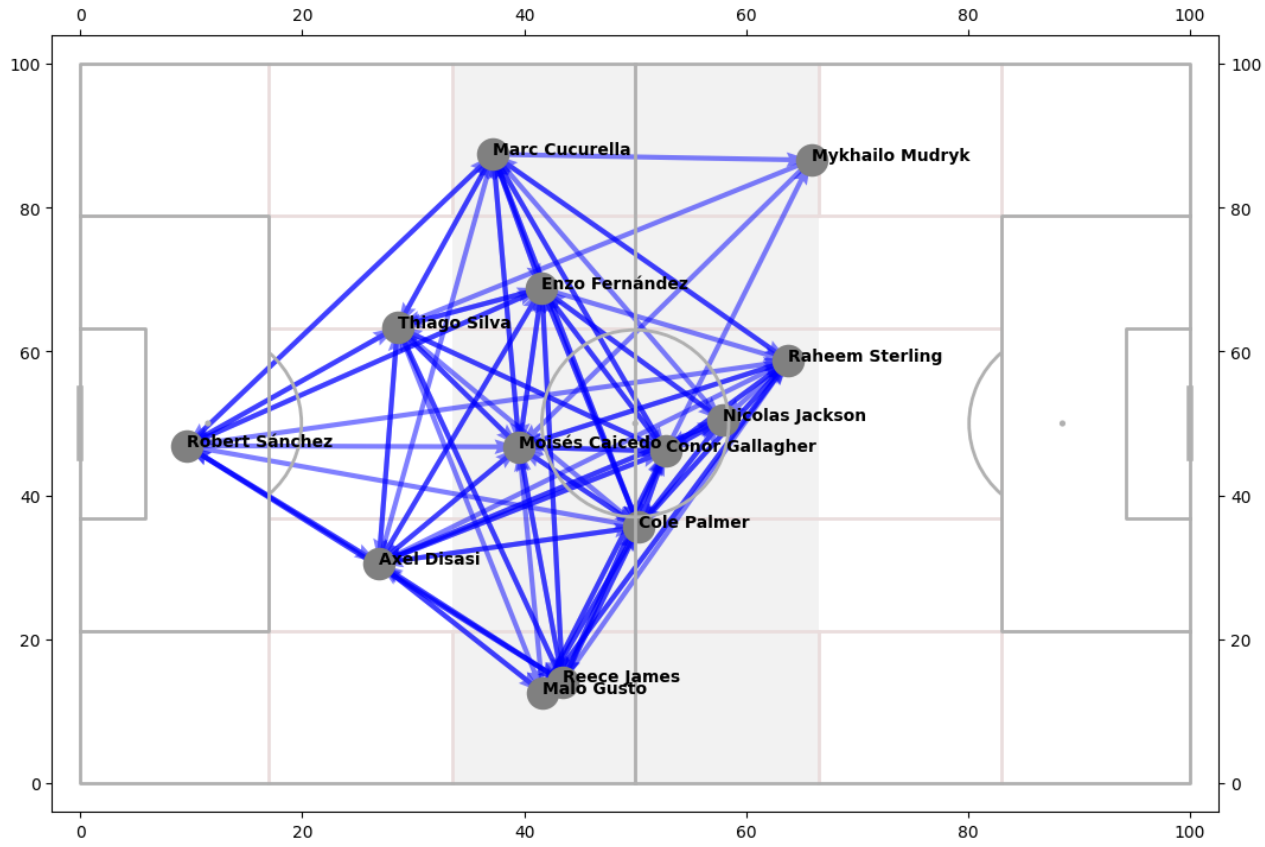
In [ ]:
```
# define a passer and a pass recipient and add the columns
df_Ch_pn_ext = df_1[df_1['team'] == 'Chelsea']
df_Ch_pn_ext['passer'] = df_Ch_pn_ext['player_id']
df_Ch_pn_ext['recipient'] = df_Ch_pn_ext['player_id'].shift(-1)

# define successful passes
Ch_successful_passes_ext = df_Ch_pn_ext[(df_Ch_pn_ext['type'] == 'Pass')&(df_Ch_pn_ext['outcome_type'] == 'Successful')]

# define an average player's location and quantity of passes
Ch_average_locations_ext = Ch_successful_passes_ext.groupby('passer').agg({'x': ['mean'], 'y': ['mean','count']})
Ch_average_locations_ext.columns = ['x', 'y', 'count']

Ch_average_locations_ext= Ch_average_locations_ext.merge(df_player_name[['player']], how='left', left_on='passer', right_ind
        .rename(columns={'player': 'player_name'})

# count passes
pass_between_Ch_ext = Ch_successful_passes_ext.groupby(['passer', 'recipient']).id.count().reset_index()
pass_between_Ch_ext.rename({'id':'pass_count'}, axis='columns', inplace=True)

pass_between_Ch_ext = pass_between_Ch_ext.merge(Ch_average_locations_ext, left_on ='passer', right_index = True)
pass_between_Ch_ext = pass_between_Ch_ext.merge(Ch_average_locations_ext, left_on ='recipient', right_index = True, suffixes
print(pass_between_Ch_ext)

# put the pass network on the pitch
pitch = Pitch(pitch_type='opta', positional=True, shade_middle=True, positional_color='#eadddd', shade_color='#f2f2f2', axis
fig, ax = plt.subplots(figsize=(13, 8.5))

pitch.draw(ax=ax)

arrows = pitch.arrows(pass_between_Ch_ext.x, pass_between_Ch_ext.y, pass_between_Ch_ext.x_end, pass_between_Ch_ext.y_end, ax

nodes = pitch.scatter(Ch_average_locations_ext.x, Ch_average_locations_ext.y, s =300, color = "grey", linewidth = 2.5, alpha

for i, row in Ch_average_locations_ext.iterrows():
    pitch.annotate(row.player_name, (Ch_average_locations_ext.x[i], Ch_average_locations_ext.y[i]), c='black', \
                   weight = "bold", size=10, ax=ax, zorder = 4)
```

```
/var/folders/sc/p5r88fzj46n6sr9l_c6dgt540000gn/T/ipykernel_43692/4097516628.py:3: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-v
iew-versus-a-copy
  df_Ch_pn_ext['passer'] = df_Ch_pn_ext['player_id']
/var/folders/sc/p5r88fzj46n6sr9l_c6dgt540000gn/T/ipykernel_43692/4097516628.py:4: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-v
iew-versus-a-copy
  df_Ch_pn_ext['recipient'] = df_Ch_pn_ext['player_id'].shift(-1)
```

```
      passer  recipient  pass_count          x          y  count  \
0    28550.0    28550.0           1  28.574510  63.405882     51
17  300426.0    28550.0          15  26.877273  30.542424     66
28  361330.0    28550.0           2  43.459375  14.106250     32
36  362151.0    28550.0           6   9.584375  46.896875     32
44  363496.0    28550.0           8  37.200000  87.405405     37
..       ...        ...         ...        ...        ...    ...
74  395692.0    97692.0           4  50.322222  35.629630     27
83  403850.0    97692.0           4  41.656000  12.572000     25
91  410175.0    97692.0           1  39.543333  46.743333     30
50  363496.0   409376.0           2  37.200000  87.405405     37
71  375621.0   409376.0           1  52.740000  46.137500     40

        player_name      x_end      y_end  count_end  player_name_end
0      Thiago Silva  28.574510  63.405882         51     Thiago Silva
17      Axel Disasi  28.574510  63.405882         51     Thiago Silva
28      Reece James  28.574510  63.405882         51     Thiago Silva
36   Robert Sánchez  28.574510  63.405882         51     Thiago Silva
44   Marc Cucurella  28.574510  63.405882         51     Thiago Silva
..              ...        ...        ...        ...              ...
74     Cole Palmer  63.793333  58.693333         15  Raheem Sterling
83      Malo Gusto  63.793333  58.693333         15  Raheem Sterling
91   Moisés Caicedo  63.793333  58.693333         15  Raheem Sterling
50   Marc Cucurella  65.850000  86.600000          2  Mykhailo Mudryk
71  Conor Gallagher  65.850000  86.600000          2  Mykhailo Mudryk

[101 rows x 11 columns]
```



In [ ]:
```python
# put the pass network on the pitch
pitch = Pitch(pitch_type='opta', positional=True, shade_middle=True, positional_color='#eadddd', shade_color='#f2f2f2', axis
fig, ax = plt.subplots(figsize=(13, 8.5))

pitch.draw(ax=ax)

arrows = pitch.arrows(pass_between_Ch_ext.x, pass_between_Ch_ext.y, pass_between_Ch_ext.x_end, pass_between_Ch_ext.y_end, ax

nodes = pitch.scatter(Ch_average_locations_ext.x, Ch_average_locations_ext.y, s =300, color = "grey", linewidth = 2.5, alpha

for i, row in Ch_average_locations_ext.iterrows():
    pitch.annotate(row.player_name, (Ch_average_locations_ext.x[i], Ch_average_locations_ext.y[i]), c='black', \
                   weight = "bold", size=10, ax=ax, zorder = 4)
```

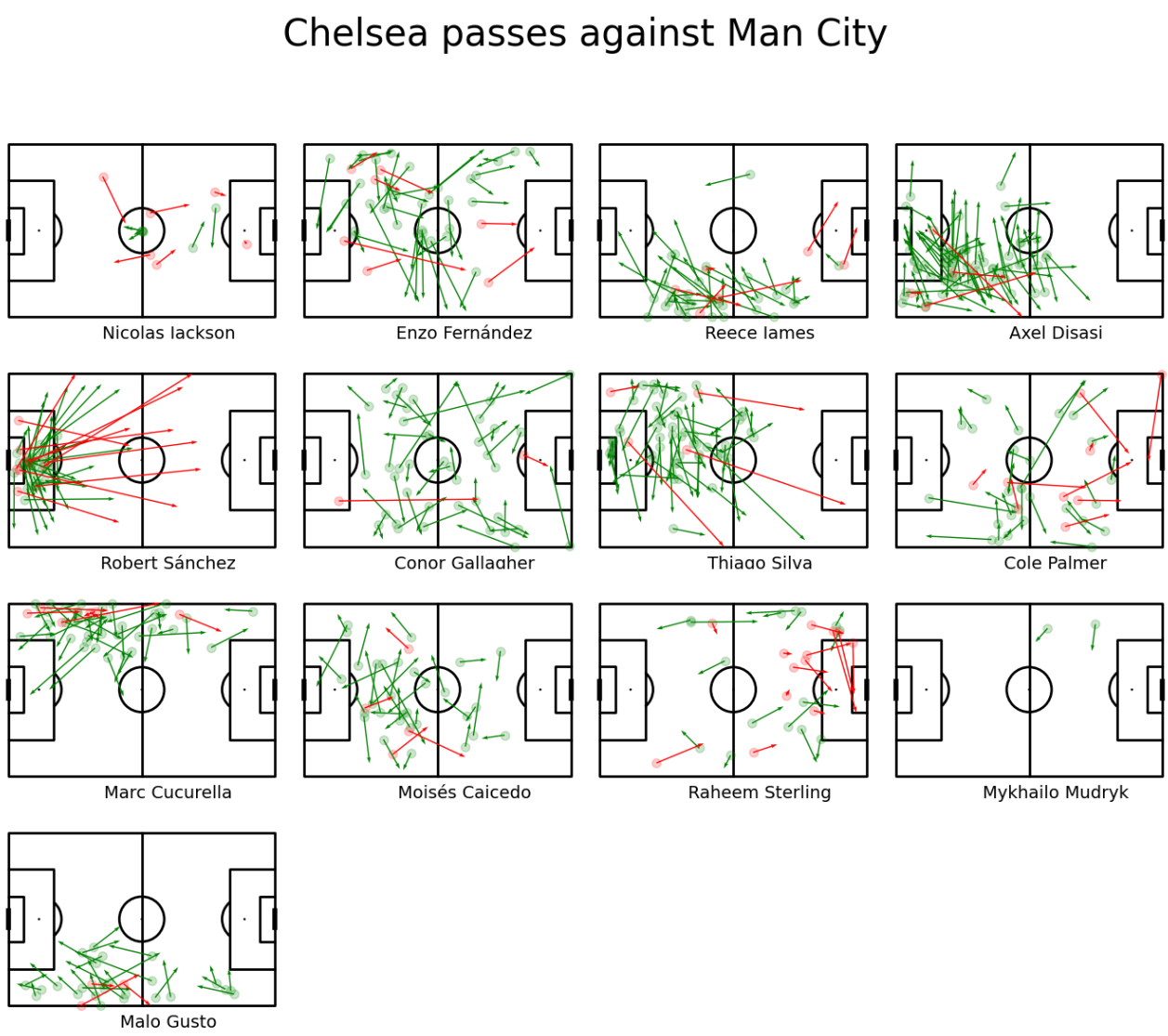


Passes by players

Chelsea players passes

In [ ]:
```
# prepare the dataframe of passes by Chelsea
mask_chelsea = (df_1.type == 'Pass') & (df_1.team == "Chelsea")
df_passes_ch = df_1.loc[mask_chelsea, ['x', 'y', 'end_x', 'end_y', 'player', 'outcome_type']]
#get the list of all players who made a pass
names_ch = df_passes_ch['player'].unique()
```

In [ ]:
```
# draw 4x4 pitches
pitch = Pitch(pitch_type='opta', line_color='black', pad_top=20)
fig, axs = pitch.grid(ncols = 4, nrows = 4, figheight=12,  grid_height=0.85, title_height=0.06, axis=False,
                      endnote_height=0.04, title_space=0.04, endnote_space=0)

# for each player
for name, ax in zip(names_ch, axs['pitch'].flat[:len(names_ch)]):
    #put player name over the plot
    ax.text(60, -10, name,
            ha='center', va='center', fontsize=14)
    #take only successful passes by this player
    player_df_sc_ch = df_passes_ch.loc[(df_passes_ch["player"] == name) & (df_passes_ch["outcome_type"] == "Successful")]
    #take only unsuccessful passes by this player
    player_df_unsc_ch = df_passes_ch.loc[(df_passes_ch["player"] == name) & (df_passes_ch["outcome_type"] == "Unsuccessful")]

    #scatter
    pitch.scatter(player_df_sc_ch.x, player_df_sc_ch.y, alpha = 0.2, s = 50, color = "green", ax=ax)
    pitch.scatter(player_df_unsc_ch.x, player_df_unsc_ch.y, alpha = 0.2, s = 50, color = "red", ax=ax)
    #plot arrow
    pitch.arrows(player_df_sc_ch.x, player_df_sc_ch.y,
            player_df_sc_ch.end_x, player_df_sc_ch.end_y, color = "green", ax=ax, width=1)
    pitch.arrows(player_df_unsc_ch.x, player_df_unsc_ch.y,
            player_df_unsc_ch.end_x, player_df_unsc_ch.end_y, color = "red", ax=ax, width=1)

# We have more than enough pitches - remove them
for ax in axs['pitch'][-1, 16 - len(names_ch)-2:]:
    ax.remove()


# Another way to set title using mplsoccer
axs['title'].text(0.5, 0.5, 'Chelsea passes against Man City', ha='center', va='center', fontsize=30)
plt.show()
```

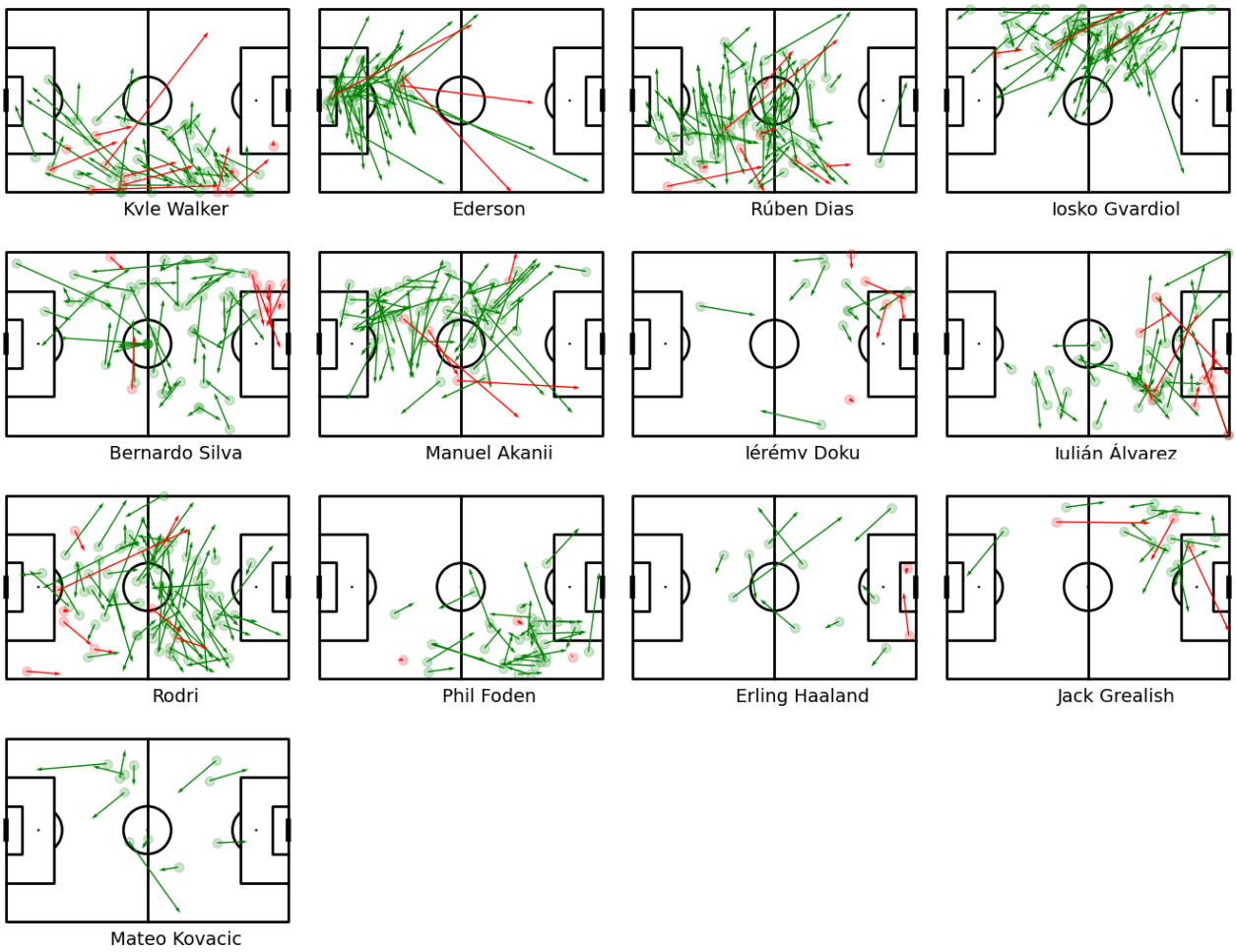


ManCity players passes

In [ ]:
```
# prepare the dataframe of passes by Man City
mask_mc = (df_1.type == 'Pass') & (df_1.team == "Man City")
df_passes_mc = df_1.loc[mask_mc, ['x', 'y', 'end_x', 'end_y', 'player', "outcome_type"]]
# get the list of all players who made a pass
names_mc = df_passes_mc['player'].unique()
```

In [ ]:
```
# draw 4x4 pitches
pitch = Pitch(pitch_type='opta', line_color='black', pad_top=20)
fig, axs = pitch.grid(ncols = 4, nrows = 4, figheight=12,  grid_height=0.85, title_height=0.06, axis=False,
                      endnote_height=0.04, title_space=0.04, endnote_space=0)

# for each player
for name, ax in zip(names_mc, axs['pitch'].flat[:len(names_mc)]):
    # put player name over the plot
    ax.text(60, -10, name,
            ha='center', va='center', fontsize=14)
    #take only successful passes by this player
    player_df_sc_mc = df_passes_mc.loc[(df_passes_mc["player"] == name) & (df_passes_mc["outcome_type"] == "Successful")]
    #take only unsuccessful passes by this player
    player_df_unsc_mc = df_passes_mc.loc[(df_passes_mc["player"] == name) & (df_passes_mc["outcome_type"] == "Unsuccessful")]

    #scatter
    pitch.scatter(player_df_sc_mc.x, player_df_sc_mc.y, alpha = 0.2, s = 50, color = "green", ax=ax)
    pitch.scatter(player_df_unsc_mc.x, player_df_unsc_mc.y, alpha = 0.2, s = 50, color = "red", ax=ax)
    #plot arrow
    pitch.arrows(player_df_sc_mc.x, player_df_sc_mc.y,
            player_df_sc_mc.end_x, player_df_sc_mc.end_y, color = "green", ax=ax, width=1)
    pitch.arrows(player_df_unsc_mc.x, player_df_unsc_mc.y,
            player_df_unsc_mc.end_x, player_df_unsc_mc.end_y, color = "red", ax=ax, width=1)

# We have more than enough pitches - remove them
for ax in axs['pitch'][-1, 16 - len(names_mc)-2:]:
    ax.remove()


# Another way to set title using mplsoccer
axs['title'].text(0.5, 0.5, 'Man City passes against Chelsea', ha='center', va='center', fontsize=30)
plt.show()
```

Man City passes against Chelsea



In [ ]:
```
df_passes_mc_ratio = df_passes_mc.reset_index().groupby(['player', 'outcome_type']).index.count()
```

In [ ]:
```
df_passes_mc_ratio
```

Out[ ]:
```
player          outcome_type
Bernardo Silva  Successful      48
                Unsuccessful     9
Ederson         Successful      40
                Unsuccessful     3
Erling Haaland  Successful      10
                Unsuccessful     2
Jack Grealish   Successful      15
                Unsuccessful     3
Josko Gvardiol  Successful      60
                Unsuccessful     3
Julián Álvarez  Successful      33
                Unsuccessful     9
Jérémy Doku     Successful      10
                Unsuccessful     5
Kyle Walker     Successful      44
                Unsuccessful    11
Manuel Akanji   Successful      57
                Unsuccessful     4
Mateo Kovacic   Successful      11
Phil Foden      Successful      33
                Unsuccessful     3
Rodri           Successful      67
                Unsuccessful     9
Rúben Dias      Successful      76
                Unsuccessful     8
Name: index, dtype: int64
```

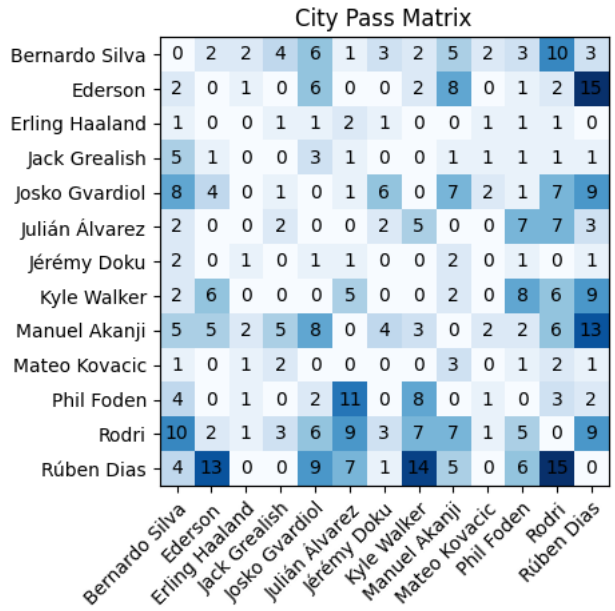
Passes matrix

In [ ]:
```
def get_pass_matrix():
    pass_init = events_atomic_spadl[events_atomic_spadl["type_id"] == 0][
        ["type_id", "player", "original_event_id", "team_id"]
    ]
    pass_receive = events_atomic_spadl[events_atomic_spadl["type_id"] == 23][
        ["type_id", "player", "original_event_id"]
    ]
    # return pass_init
    pass_matrix = pass_init.join(
```

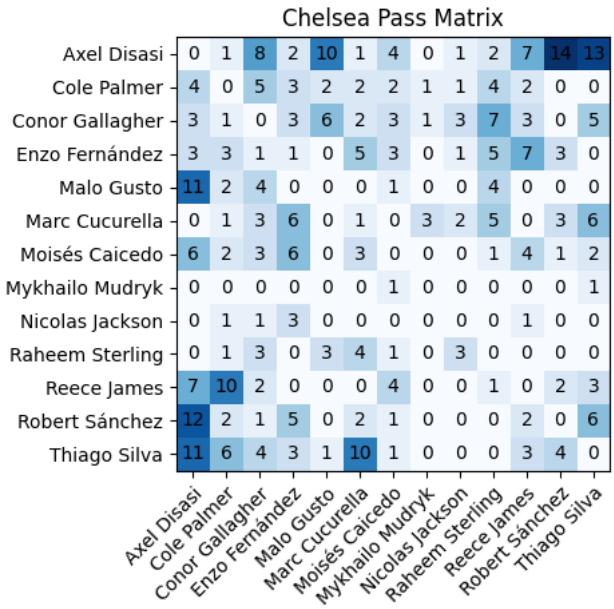
```
        pass_receive.set_index("original_event_id"),
        on="original_event_id",
        rsuffix="_other",
    )[["player", "player_other", "team_id"]]
    pass_matrix["count"] = 1
    pass_matrix = pass_matrix.groupby(["team_id", "player", "player_other"]).agg(
        {"count": "sum"}
    )
    return pass_matrix.reset_index()


pass_matrix = get_pass_matrix()
```

In [ ]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt


def print_pass_matrix(df, team):
    # Create a pivot table to reshape the data for the pass matrix
    pass_matrix = df.pivot_table(
        index="player", columns="player_other", values="count", fill_value=0
    )

    # Get players and pass counts
    players = pass_matrix.columns
    pass_counts = pass_matrix.values

    # Plotting the pass matrix
    fig, ax = plt.subplots()
    im = ax.imshow(pass_counts, cmap="Blues")

    # Set ticks and labels
    ax.set_xticks(np.arange(len(players)))
    ax.set_yticks(np.arange(len(players)))
    ax.set_xticklabels(players)
    ax.set_yticklabels(players)

    # Rotate the tick labels and set their alignment
    plt.setp(ax.get_xticklabels(), rotation=45, ha="right", rotation_mode="anchor")

    # Loop over data dimensions and create text annotations
    for i in range(len(players)):
        for j in range(len(players)):
            text = ax.text(
                j, i, pass_counts[i, j], ha="center", va="center", color="black"
            )

    # Set title and show the plot
    ax.set_title(f"{team} Pass Matrix")
    fig.tight_layout()
    plt.show()
```

In [ ]:
```
print_pass_matrix(pass_matrix[pass_matrix['team_id'] == MC_team_id], 'City')
```



In [ ]:
```
print_pass_matrix(pass_matrix[pass_matrix['team_id'] == chelsea_team_id], 'Chelsea')
```



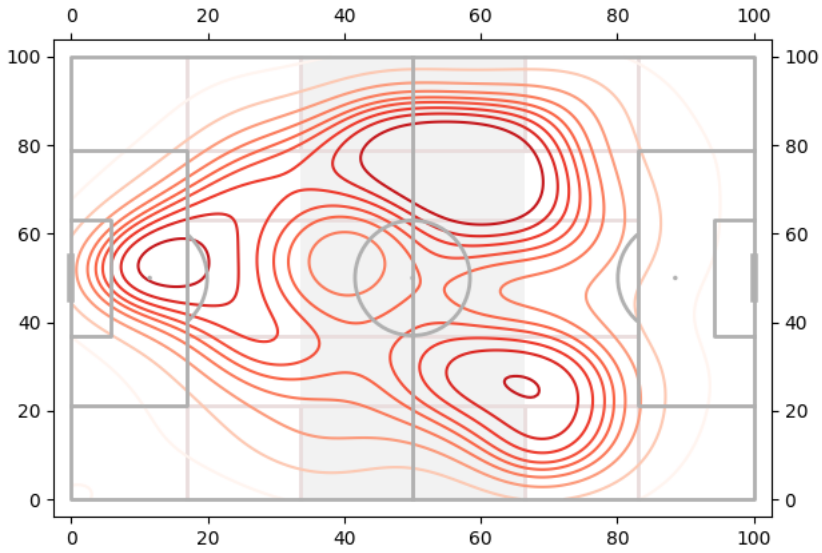
Heatmap

In [ ]:
```
# define dataframes by the teams
df_MC_hm = df_1[(df_1['team'] == 'Man City')&(df_1['outcome_type'] == 'Successful')][['minute', 'second', 'team', 'player','
df_Ch_hm = df_1[(df_1['team'] == 'Chelsea')&(df_1['outcome_type'] == 'Successful')][['minute', 'second', 'team', 'player','t
```

Heatmap for ManCity

In [ ]:
```
# make a pitch with data from OPTA and with a mention of the thirds and draw the heatmap for MC
pitch = Pitch(pitch_type='opta', positional=True, shade_middle=True, positional_color='#eadddd', shade_color='#f2f2f2', axis
fig, ax = pitch.draw()
pitch.kdeplot(df_MC_hm['x'], df_MC_hm['y'], ax=ax, cmap = 'Reds', zorder=1)
```

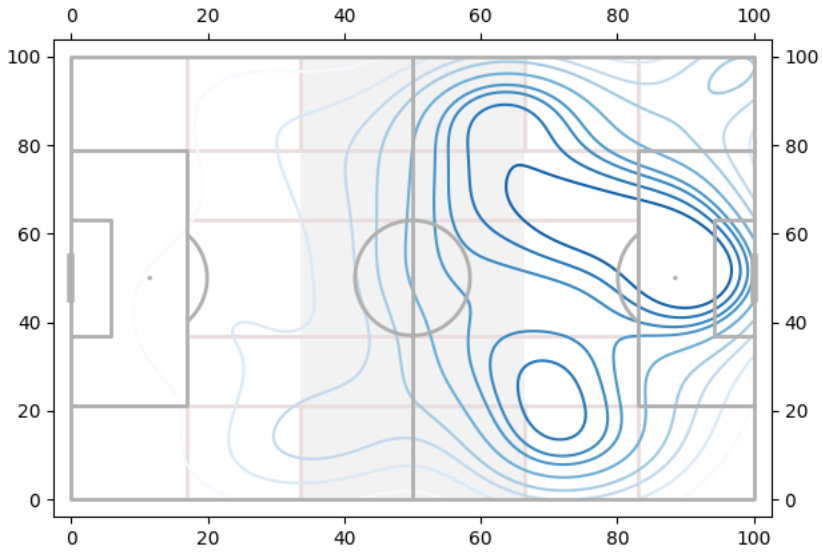
Out[ ]: <Axes: >



Heatmap for Chelsea

In [ ]:
```
# make a pitch with data from OPTA and with a mention of the thirds and draw the heatmap for Chelsea
pitch = Pitch(pitch_type='opta', positional=True, shade_middle=True, positional_color='#eadddd', shade_color='#f2f2f2', axis
fig, ax = pitch.draw()
pitch.kdeplot(100 - df_Ch_hm['x'], 100 - df_Ch_hm['y'], ax=ax, cmap = 'Blues', zorder=1)
```

Out[ ]: <Axes: >



xG Flow Chat

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

In [ ]:
```
df_2 = pd.read_csv('data/12112023/xG_shots.csv')
```

In [ ]:
```
df_2
```

Out[ ]:

	period	minute	second	type	outcome_type	team_id	team	player_id	player	x	y	xG	goal (y/n)
0	FirstHalf	0	25	SavedShot	Successful	15	Chelsea	361330	Reece James	85.9	28.8	0.03	n
1	FirstHalf	11	37	SavedShot	Successful	15	Chelsea	395692	Cole Palmer	87.6	31.4	0.04	n
2	FirstHalf	15	35	SavedShot	Successful	167	Man City	365409	Juli?n ?lvarez	93.8	29.8	0.04	n
3	FirstHalf	16	6	SavedShot	Successful	167	Man City	402664	Josko Gvardiol	86.7	57.8	0.01	n
4	FirstHalf	18	14	SavedShot	Successful	15	Chelsea	375621	Conor Gallagher	77.7	44.2	0.05	n
5	FirstHalf	24	3	Goal	Successful	167	Man City	315227	Erling Haaland	88.5	50.0	0.79	y
6	FirstHalf	26	26	SavedShot	Successful	15	Chelsea	410175	Mois?s Caicedo	77.8	53.3	0.03	n
7	FirstHalf	27	50	SavedShot	Successful	15	Chelsea	361330	Reece James	77.5	64.9	0.06	n
8	FirstHalf	28	24	Goal	Successful	15	Chelsea	28550	Thiago Silva	95.2	39.1	0.07	y
9	FirstHalf	31	40	MissedShot	Successful	167	Man City	315227	Erling Haaland	99.2	58.4	0.07	n
10	FirstHalf	33	51	MissedShot	Successful	167	Man City	331254	Phil Foden	85.6	36.1	0.07	n
11	FirstHalf	36	23	Goal	Successful	15	Chelsea	97692	Raheem Sterling	96.7	50.9	0.43	y
12	FirstHalf	39	48	SavedShot	Successful	15	Chelsea	369430	Enzo Fern?ndez	78.5	63.2	0.04	n
13	FirstHalf	40	41	MissedShot	Successful	15	Chelsea	97692	Raheem Sterling	88.3	69.3	0.02	n
14	FirstHalf	41	33	SavedShot	Successful	167	Man City	315227	Erling Haaland	87.4	55.8	0.31	n
15	FirstHalf	42	36	SavedShot	Successful	167	Man City	331254	Phil Foden	78.1	33.1	0.02	n
16	FirstHalf	43	14	SavedShot	Successful	15	Chelsea	426050	Nicolas Jackson	84.7	52.8	0.09	n
17	FirstHalf	45	27	Goal	Successful	167	Man City	297390	Manuel Akanji	93.0	51.2	0.29	y
18	FirstHalf	51	26	SavedShot	Successful	15	Chelsea	426050	Nicolas Jackson	89.9	66.3	0.04	n
19	SecondHalf	46	31	Goal	Successful	167	Man City	315227	Erling Haaland	97.7	52.8	0.87	y
20	SecondHalf	51	53	SavedShot	Successful	167	Man City	365409	Juli?n ?lvarez	86.3	51.5	0.12	n
21	SecondHalf	57	27	SavedShot	Successful	167	Man City	388098	Jeremy Doku	90.6	63.9	0.09	n
22	SecondHalf	59	42	SavedShot	Successful	15	Chelsea	395692	Cole Palmer	89.0	53.0	0.48	n
23	SecondHalf	60	50	SavedShot	Successful	167	Man City	331254	Phil Foden	85.4	38.6	0.07	n
24	SecondHalf	66	22	SavedShot	Successful	15	Chelsea	375621	Conor Gallagher	73.0	53.3	0.03	n
25	SecondHalf	66	25	Goal	Successful	15	Chelsea	426050	Nicolas Jackson	90.5	47.6	0.55	y
26	SecondHalf	74	53	MissedShot	Successful	15	Chelsea	403850	Malo Gusto	90.5	36.5	0.20	n
27	SecondHalf	80	17	MissedShot	Successful	15	Chelsea	375621	Conor Gallagher	71.0	47.9	0.02	n
28	SecondHalf	85	30	SavedShot	Successful	167	Man City	93894	Mateo Kovacic	80.6	49.5	0.03	n
29	SecondHalf	85	33	Goal	Successful	167	Man City	303139	Rodri	76.7	50.8	0.03	y
30	SecondHalf	94	24	Goal	Successful	15	Chelsea	395692	Cole Palmer	88.5	50.0	0.79	y
31	SecondHalf	98	1	MissedShot	Successful	167	Man City	69778	Kyle Walker	74.9	55.2	0.07	n

In [ ]:
```
a_xG = [0]
h_xG = [0]
```

```
a_min = [0]
h_min = [0]

# Find home and away teams
hteam = df_2['team'].iloc[0]  # first row (Chelsea)
ateam = df_2['team'].iloc[-1] # last row (ManCity)

for x in range(len(df_2['xG'])):
    if df_2['team'][x]==ateam:
        a_xG.append(df_2['xG'][x])
        a_min.append(df_2['minute'][x])
    if df_2['team'][x]==hteam:
        h_xG.append(df_2['xG'][x])
        h_min.append(df_2['minute'][x])
```

In [ ]:
```
a_xG
```

Out[ ]:
```
[0,
 0.04,
 0.01,
 0.79,
 0.07,
 0.07,
 0.31,
 0.02,
 0.29,
 0.87,
 0.12,
 0.09,
 0.07,
 0.03,
 0.03,
 0.07]
```

In [ ]:
```
def nums_cumulative_sum(nums_list):
    return [sum(nums_list[:i+1]) for i in range(len(nums_list))]
a_cumulative = nums_cumulative_sum(a_xG)
h_cumulative = nums_cumulative_sum(h_xG)
```

In [ ]:
```
#a_cumulative
```

In [ ]:
```
#h_cumulative
```

In [ ]:
```
a_total = round(a_cumulative [-1], 2)
h_total = round(h_cumulative [-1], 2)
```

In [ ]:
```
# cumulative Sum xG home team
h_df = df_2[df_2['team'] == hteam]
h_df.sort_values(by='minute', inplace=True)
h_df['h_cum'] = h_df['xG'].cumsum()

# cumulative Sum xG away team
a_df = df_2[df_2['team'] == ateam]
a_df.sort_values(by='minute', inplace=True)
a_df['a_cum'] = a_df['xG'].cumsum()
```

```
/var/folders/sc/p5r88fzj46n6sr9l_c6dgt540000gn/T/ipykernel_43692/2695564966.py:3: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  h_df.sort_values(by='minute', inplace=True)
/var/folders/sc/p5r88fzj46n6sr9l_c6dgt540000gn/T/ipykernel_43692/2695564966.py:4: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  h_df['h_cum'] = h_df['xG'].cumsum()
/var/folders/sc/p5r88fzj46n6sr9l_c6dgt540000gn/T/ipykernel_43692/2695564966.py:8: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  a_df.sort_values(by='minute', inplace=True)
/var/folders/sc/p5r88fzj46n6sr9l_c6dgt540000gn/T/ipykernel_43692/2695564966.py:9: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  a_df['a_cum'] = a_df['xG'].cumsum()
```

In [ ]:
```
h_df
```

Out[ ]:

	period	minute	second	type	outcome_type	team_id	team	player_id	player	x	y	xG	goal (y/n)	h_cum
0	FirstHalf	0	25	SavedShot	Successful	15	Chelsea	361330	Reece James	85.9	28.8	0.03	n	0.03
1	FirstHalf	11	37	SavedShot	Successful	15	Chelsea	395692	Cole Palmer	87.6	31.4	0.04	n	0.07
4	FirstHalf	18	14	SavedShot	Successful	15	Chelsea	375621	Conor Gallagher	77.7	44.2	0.05	n	0.12
6	FirstHalf	26	26	SavedShot	Successful	15	Chelsea	410175	Mois?s Caicedo	77.8	53.3	0.03	n	0.15
7	FirstHalf	27	50	SavedShot	Successful	15	Chelsea	361330	Reece James	77.5	64.9	0.06	n	0.21
8	FirstHalf	28	24	Goal	Successful	15	Chelsea	28550	Thiago Silva	95.2	39.1	0.07	y	0.28

	period	minute	second	type	outcome_type	team_id	team	player_id	player	x	y	xG	goal (y/n)	h_cum
11	FirstHalf	36	23	Goal	Successful	15	Chelsea	97692	Raheem Sterling	96.7	50.9	0.43	y	0.71
12	FirstHalf	39	48	SavedShot	Successful	15	Chelsea	369430	Enzo Fern?ndez	78.5	63.2	0.04	n	0.75
13	FirstHalf	40	41	MissedShot	Successful	15	Chelsea	97692	Raheem Sterling	88.3	69.3	0.02	n	0.77
16	FirstHalf	43	14	SavedShot	Successful	15	Chelsea	426050	Nicolas Jackson	84.7	52.8	0.09	n	0.86
18	FirstHalf	51	26	SavedShot	Successful	15	Chelsea	426050	Nicolas Jackson	89.9	66.3	0.04	n	0.90
22	SecondHalf	59	42	SavedShot	Successful	15	Chelsea	395692	Cole Palmer	89.0	53.0	0.48	n	1.38
24	SecondHalf	66	22	SavedShot	Successful	15	Chelsea	375621	Conor Gallagher	73.0	53.3	0.03	n	1.41
25	SecondHalf	66	25	Goal	Successful	15	Chelsea	426050	Nicolas Jackson	90.5	47.6	0.55	y	1.96
26	SecondHalf	74	53	MissedShot	Successful	15	Chelsea	403850	Malo Gusto	90.5	36.5	0.20	n	2.16
27	SecondHalf	80	17	MissedShot	Successful	15	Chelsea	375621	Conor Gallagher	71.0	47.9	0.02	n	2.18
30	SecondHalf	94	24	Goal	Successful	15	Chelsea	395692	Cole Palmer	88.5	50.0	0.79	y	2.97

In [ ]:
```
h_goal = h_df[h_df['type'].str.contains("Goal")]
h_goal["scorechart"] = h_goal["minute"].astype(str) + "'" + " " +h_goal["player"]
a_goal = a_df[a_df['type'].str.contains("Goal")]
a_goal["scorechart"] = a_goal["minute"].astype(str) + "'" + " " +a_goal["player"]
```

```
/var/folders/sc/p5r88fzj46n6sr9l_c6dgt540000gn/T/ipykernel_43692/1230918128.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  h_goal["scorechart"] = h_goal["minute"].astype(str) + "'" + " " +h_goal["player"]
/var/folders/sc/p5r88fzj46n6sr9l_c6dgt540000gn/T/ipykernel_43692/1230918128.py:4: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  a_goal["scorechart"] = a_goal["minute"].astype(str) + "'" + " " +a_goal["player"]
```

In [ ]:
```
h_goal
```

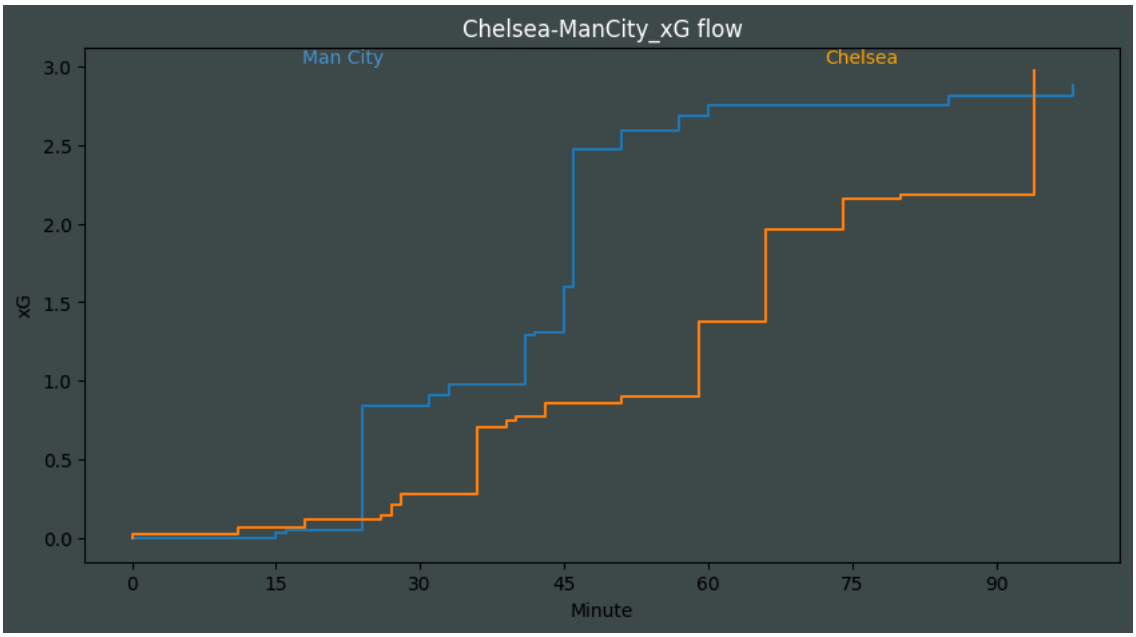
Out[ ]:

	period	minute	second	type	outcome_type	team_id	team	player_id	player	x	y	xG	goal (y/n)	h_cum	scorechart
8	FirstHalf	28	24	Goal	Successful	15	Chelsea	28550	Thiago Silva	95.2	39.1	0.07	y	0.28	28' Thiago Silva
11	FirstHalf	36	23	Goal	Successful	15	Chelsea	97692	Raheem Sterling	96.7	50.9	0.43	y	0.71	36' Raheem Sterling
25	SecondHalf	66	25	Goal	Successful	15	Chelsea	426050	Nicolas Jackson	90.5	47.6	0.55	y	1.96	66' Nicolas Jackson
30	SecondHalf	94	24	Goal	Successful	15	Chelsea	395692	Cole Palmer	88.5	50.0	0.79	y	2.97	94' Cole Palmer

xG flow without goals

In [ ]:
```
fig, ax = plt.subplots(figsize = (10,5))
fig.set_facecolor('#3d4849')
ax.patch.set_facecolor('#3d4849')

plt.xticks([0,15,30,45,60,75,90])
plt.xlabel('Minute')
plt.ylabel('xG')

ax.step(x=a_min, y=a_cumulative, where='post')
ax.step(x=h_min, y=h_cumulative, where='post')


title_text = 'Chelsea-ManCity_xG flow'
ax.set_title(title_text, color='white')

ax.text(0.25, 0.98, 'Man City', color=(72/255, 147/255, 206/255), fontsize=10, ha='center', va='center', transform=ax.transA
ax.text(0.75, 0.98, 'Chelsea', color='orange', fontsize=10, ha='center', va='center', transform=ax.transAxes)
```

Out[ ]:
```
Text(0.75, 0.98, 'Chelsea')
```

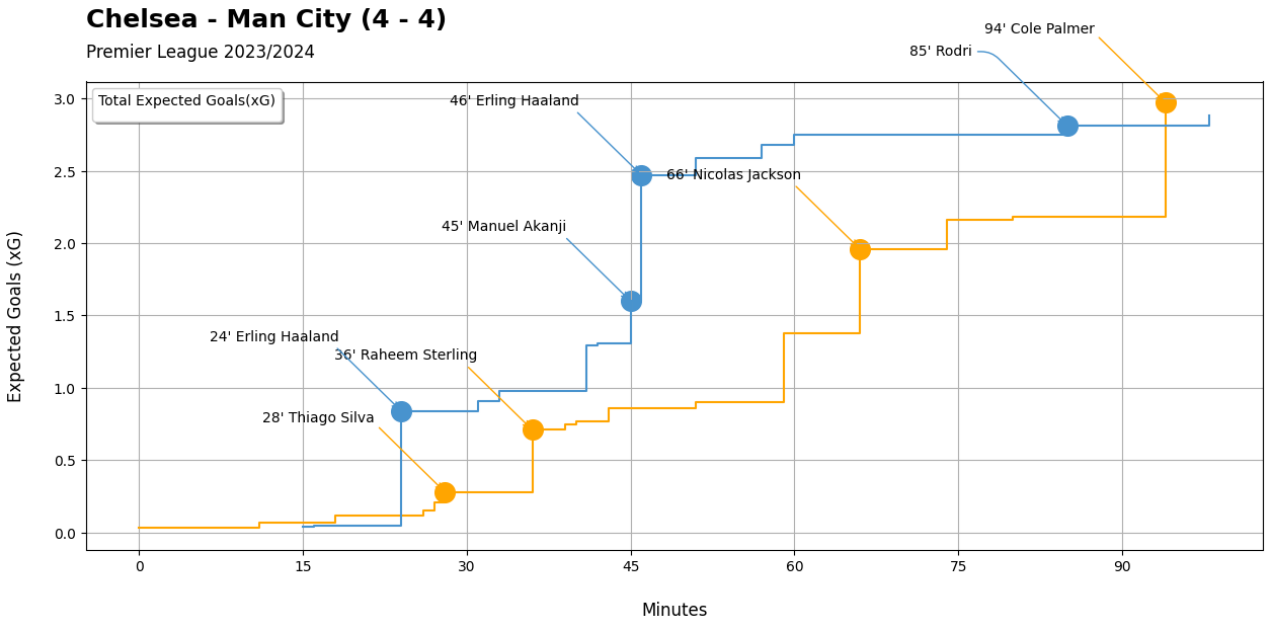


xG flow with goals

In [ ]:

```
fig, ax = plt.subplots(figsize=(15,6))

# Create line plots
ax.step(x = h_df['minute'], y = h_df['h_cum'], where = 'post', color='orange')
ax.step(x = a_df['minute'], y = a_df['a_cum'], where = 'post', color=(72/255, 147/255, 206/255))

# Create scatter plot for highlighting the goal
ax.scatter(x= h_goal['minute'], y = h_goal['h_cum'] , marker= 'o', s= 200, color='orange')
ax.scatter(x= a_goal['minute'], y = a_goal['a_cum'] , marker= 'o', s =200, color=(72/255, 147/255, 206/255))


# Customize our chart
for j, txt in h_goal['scorechart'].items():
    ax.annotate(txt, (h_goal['minute'][j], h_goal['h_cum'][j]), xycoords='data', ha='center',
                xytext=(-90, 50), textcoords='offset points',
                arrowprops=dict(arrowstyle="->", connectionstyle="arc,angleA=0,armA=50,rad=10", color='orange'))

for i, txt in a_goal['scorechart'].items():
    ax.annotate(txt, (a_goal['minute'][i], a_goal['a_cum'][i]), xycoords='data', ha='center',
                xytext=(-90, 50), textcoords='offset points',
                arrowprops=dict(arrowstyle="->", connectionstyle="arc,angleA=0,armA=50,rad=10", color=(72/255, 147/255, 206/


plt.xticks([0,15,30,45,60,75,90])
plt.yticks([0, 0.5, 1, 1.5, 2, 2.5, 3])
plt.grid()

fig.text(s=hteam + " - " + ateam + " (" + '4' + " - " + '4' + ")", x=0.125, y=0.97, fontsize=18,fontweight="bold")
fig.text(s='Premier League 2023/2024', x=0.125, y=0.92, fontsize=12)
legend = ax.legend(title = 'Total Expected Goals(xG)' ,loc='best', shadow=True)
legend._legend_box.align = "left"

plt.ylabel("Expected Goals (xG)", fontsize = 12, labelpad = 20)
plt.xlabel("Minutes", fontsize = 12, labelpad = 20)
plt.show()
```

```
No artists with labels found to put in legend.  Note that artists whose label start with an underscore are ignored when l
egend() is called with no argument.
```



Simulating match result

In [ ]:
```
%pip install statsmodels
```

```
Requirement already satisfied: statsmodels in /usr/local/lib/python3.11/site-packages (0.14.1)
Requirement already satisfied: numpy<2,>=1.18 in /usr/local/lib/python3.11/site-packages (from statsmodels) (1.25.0)
Requirement already satisfied: scipy!=1.9.2,>=1.4 in /usr/local/lib/python3.11/site-packages (from statsmodels) (1.11.4)
Requirement already satisfied: pandas!=2.1.0,>=1.0 in /usr/local/lib/python3.11/site-packages (from statsmodels) (1.5.3)
Requirement already satisfied: patsy>=0.5.4 in /usr/local/lib/python3.11/site-packages (from statsmodels) (0.5.6)
Requirement already satisfied: packaging>=21.3 in /usr/local/lib/python3.11/site-packages (from statsmodels) (23.2)
Requirement already satisfied: python-dateutil>=2.8.1 in /Users/ivn/Library/Python/3.11/lib/python/site-packages (from pa
ndas!=2.1.0,>=1.0->statsmodels) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.11/site-packages (from pandas!=2.1.0,>=1.0->statsmo
dels) (2023.3.post1)
Requirement already satisfied: six in /Users/ivn/Library/Python/3.11/lib/python/site-packages (from patsy>=0.5.4->statsmo
dels) (1.16.0)

[notice] A new release of pip is available: 23.3.1 -> 23.3.2
[notice] To update, run: python3.11 -m pip install --upgrade pip
Note: you may need to restart the kernel to use updated packages.
```

In [ ]:
```
#https://soccermatics.readthedocs.io/en/latest/gallery/lesson5/plot_SimulateMatches.html
import statsmodels.api as sm
import statsmodels.formula.api as smf
#import pandas as pd
#import matplotlib.pyplot as plt
import numpy as np
import seaborn
from scipy.stats import poisson,skellam
```

In [ ]:
```
# prediction before the match
epl = pd.read_csv('data/12112023/E0.csv')
epl = epl[['HomeTeam','AwayTeam','FTHG','FTAG']]
epl = epl.rename(columns={'FTHG': 'HomeGoals', 'FTAG': 'AwayGoals'})
```

In [ ]:
```
goal_model_data = pd.concat([epl[['HomeTeam','AwayTeam','HomeGoals']].assign(home=1).rename(
            columns={'HomeTeam':'team', 'AwayTeam':'opponent','HomeGoals':'goals'}),
           epl[['AwayTeam','HomeTeam','AwayGoals']].assign(home=0).rename(
            columns={'AwayTeam':'team', 'HomeTeam':'opponent','AwayGoals':'goals'})])

poisson_model = smf.glm(formula="goals ~ home + team + opponent", data=goal_model_data,
                        family=sm.families.Poisson()).fit()
poisson_model.summary()
```

Out[ ]:

Generalized Linear Model Regression Results

Dep. Variable:	goals	**No. Observations:**	238
Model:	GLM	**Df Residuals:**	198
Model Family:	Poisson	**Df Model:**	39
Link Function:	Log	**Scale:**	1.0000
Method:	IRLS	**Log-Likelihood:**	-328.03
Date:	Tue, 30 Jan 2024	**Deviance:**	195.65
Time:	21:06:17	**Pearson chi2:**	166.
No. Iterations:	5	**Pseudo R-squ. (CS):**	0.3140
Covariance Type:	nonrobust		

	coef	std err	z	P>|z|	[0.025	0.975]
Intercept	0.1748	0.389	0.450	0.653	-0.587	0.936
team[T.Aston Villa]	0.1510	0.279	0.541	0.588	-0.396	0.698

team[T.Bournemouth]	-0.7570	0.367	-2.061	0.039	-1.477	-0.037
team[T.Brentford]	-0.2635	0.305	-0.864	0.387	-0.861	0.334
team[T.Brighton]	-0.0214	0.288	-0.074	0.941	-0.585	0.542
team[T.Burnley]	-0.9117	0.392	-2.323	0.020	-1.681	-0.142
team[T.Chelsea]	-0.3181	0.321	-0.990	0.322	-0.948	0.312
team[T.Crystal Palace]	-0.7587	0.353	-2.147	0.032	-1.451	-0.066
team[T.Everton]	-0.6307	0.341	-1.849	0.064	-1.299	0.038
team[T.Fulham]	-0.8806	0.378	-2.327	0.020	-1.622	-0.139
team[T.Liverpool]	0.0633	0.285	0.222	0.825	-0.496	0.623
team[T.Luton]	-0.9074	0.378	-2.402	0.016	-1.648	-0.167
team[T.Man City]	0.1198	0.279	0.429	0.668	-0.427	0.667
team[T.Man United]	-0.7168	0.346	-2.074	0.038	-1.394	-0.039
team[T.Newcastle]	0.0462	0.285	0.162	0.871	-0.512	0.605
team[T.Nott'm Forest]	-0.5733	0.339	-1.692	0.091	-1.237	0.091
team[T.Sheffield United]	-0.8170	0.378	-2.163	0.031	-1.557	-0.077
team[T.Tottenham]	-0.0642	0.290	-0.221	0.825	-0.632	0.504
team[T.West Ham]	-0.1320	0.301	-0.438	0.661	-0.723	0.459
team[T.Wolves]	-0.4056	0.324	-1.250	0.211	-1.042	0.230
opponent[T.Aston Villa]	0.4374	0.405	1.080	0.280	-0.356	1.231
opponent[T.Bournemouth]	0.7051	0.377	1.873	0.061	-0.033	1.443
opponent[T.Brentford]	0.4586	0.402	1.142	0.254	-0.329	1.246
opponent[T.Brighton]	0.6030	0.391	1.544	0.123	-0.162	1.369
opponent[T.Burnley]	0.9058	0.370	2.450	0.014	0.181	1.630
opponent[T.Chelsea]	0.1347	0.436	0.309	0.758	-0.721	0.990
opponent[T.Crystal Palace]	0.3510	0.409	0.859	0.391	-0.450	1.152
opponent[T.Everton]	0.3699	0.409	0.904	0.366	-0.432	1.172
opponent[T.Fulham]	0.4674	0.394	1.186	0.236	-0.305	1.240
opponent[T.Liverpool]	-0.2018	0.453	-0.446	0.656	-1.089	0.685
opponent[T.Luton]	0.6155	0.389	1.584	0.113	-0.146	1.377
opponent[T.Man City]	-0.2087	0.481	-0.434	0.664	-1.151	0.734
opponent[T.Man United]	0.4075	0.410	0.994	0.320	-0.396	1.211
opponent[T.Newcastle]	0.0370	0.429	0.086	0.931	-0.803	0.877
opponent[T.Nott'm Forest]	0.3826	0.402	0.951	0.342	-0.406	1.171
opponent[T.Sheffield United]	0.9326	0.369	2.525	0.012	0.209	1.656
opponent[T.Tottenham]	0.4152	0.416	0.997	0.319	-0.401	1.231
opponent[T.West Ham]	0.5698	0.388	1.470	0.142	-0.190	1.330
opponent[T.Wolves]	0.5392	0.392	1.375	0.169	-0.229	1.308
home	0.2146	0.108	1.978	0.048	0.002	0.427

In [ ]:
```
# set teams here
home_team='Chelsea'
away_team='Man City'

# predict for Chelsea vs. Manchester City
home_score_rate=poisson_model.predict(pd.DataFrame(data={'team': home_team, 'opponent': away_team,
                                      'home':1},index=[1]))
away_score_rate=poisson_model.predict(pd.DataFrame(data={'team': away_team, 'opponent': home_team,
                                      'home':0},index=[1]))
print(home_team + ' against ' + away_team + ' expect to score: ' + str(home_score_rate))
print(away_team + ' against ' + home_team + ' expect to score: ' + str(away_score_rate))

# lets just get a result
home_goals=np.random.poisson(home_score_rate)
away_goals=np.random.poisson(away_score_rate)
#print(home_team + ': ' + str(home_goals[0]))
#print(away_team + ': '  + str(away_goals[0]))
```

```
Chelsea against Man City expect to score: 1    0.871664
dtype: float64
Man City against Chelsea expect to score: 1    1.536197
dtype: float64
```

Expected result: Chelsea-ManCity 1-1