Relationship between Six Nations and World Cup Performances

In [1]:
```
#Install relevent packages for scraping
%pip install beautifulsoup4 requests selenium bs4
%pip install --upgrade selenium webdriver-manager
```

```
Requirement already satisfied: beautifulsoup4 in c:\users\gre01\anaconda3\lib\site-pack
ages (4.10.0)
Requirement already satisfied: requests in c:\users\gre01\anaconda3\lib\site-packages
(2.26.0)
Requirement already satisfied: selenium in c:\users\gre01\anaconda3\lib\site-packages
(4.20.0)
Requirement already satisfied: bs4 in c:\users\gre01\anaconda3\lib\site-packages (0.0.
2)
Requirement already satisfied: soupsieve>1.2 in c:\users\gre01\anaconda3\lib\site-packa
ges (from beautifulsoup4) (2.2.1)
Requirement already satisfied: idna<4,>=2.5 in c:\users\gre01\anaconda3\lib\site-packag
es (from requests) (3.2)
Requirement already satisfied: certifi>=2017.4.17 in c:\users\gre01\anaconda3\lib\site-
packages (from requests) (2021.10.8)
Requirement already satisfied: urllib3<1.27,>=1.21.1 in c:\users\gre01\anaconda3\lib\si
te-packages (from requests) (1.26.7)
Requirement already satisfied: charset-normalizer~=2.0.0 in c:\users\gre01\anaconda3\li
b\site-packages (from requests) (2.0.4)
Requirement already satisfied: trio~=0.17 in c:\users\gre01\anaconda3\lib\site-packages
(from selenium) (0.25.0)
Requirement already satisfied: typing_extensions>=4.9.0 in c:\users\gre01\anaconda3\lib
\site-packages (from selenium) (4.11.0)
Requirement already satisfied: trio-websocket~=0.9 in c:\users\gre01\anaconda3\lib\site
-packages (from selenium) (0.11.1)
Requirement already satisfied: exceptiongroup in c:\users\gre01\anaconda3\lib\site-pack
ages (from trio~=0.17->selenium) (1.2.1)
Requirement already satisfied: cffi>=1.14 in c:\users\gre01\anaconda3\lib\site-packages
(from trio~=0.17->selenium) (1.14.6)
Requirement already satisfied: attrs>=23.2.0 in c:\users\gre01\anaconda3\lib\site-packa
ges (from trio~=0.17->selenium) (23.2.0)
Requirement already satisfied: outcome in c:\users\gre01\anaconda3\lib\site-packages (f
rom trio~=0.17->selenium) (1.3.0.post0)
Requirement already satisfied: sniffio>=1.3.0 in c:\users\gre01\anaconda3\lib\site-pack
ages (from trio~=0.17->selenium) (1.3.1)
Requirement already satisfied: sortedcontainers in c:\users\gre01\anaconda3\lib\site-pa
ckages (from trio~=0.17->selenium) (2.4.0)
Note: you may need to restart the kernel to use updated packages.
Requirement already satisfied: pycparser in c:\users\gre01\anaconda3\lib\site-packages
(from cffi>=1.14->trio~=0.17->selenium) (2.20)
Requirement already satisfied: wsproto>=0.14 in c:\users\gre01\anaconda3\lib\site-packa
ges (from trio-websocket~=0.9->selenium) (1.2.0)
Requirement already satisfied: PySocks!=1.5.7,<2.0,>=1.5.6 in c:\users\gre01\anaconda3
\lib\site-packages (from urllib3<1.27,>=1.21.1->requests) (1.7.1)
Requirement already satisfied: h11<1,>=0.9.0 in c:\users\gre01\anaconda3\lib\site-packa
ges (from wsproto>=0.14->trio-websocket~=0.9->selenium) (0.14.0)
Requirement already satisfied: selenium in c:\users\gre01\anaconda3\lib\site-packages
(4.20.0)
Requirement already satisfied: webdriver-manager in c:\users\gre01\anaconda3\lib\site-p
ackages (4.0.1)
Requirement already satisfied: urllib3[socks]<3,>=1.26 in c:\users\gre01\anaconda3\lib
\site-packages (from selenium) (1.26.7)
Requirement already satisfied: trio~=0.17 in c:\users\gre01\anaconda3\lib\site-packages
(from selenium) (0.25.0)
Requirement already satisfied: trio-websocket~=0.9 in c:\users\gre01\anaconda3\lib\site
-packages (from selenium) (0.11.1)
Requirement already satisfied: typing_extensions>=4.9.0 in c:\users\gre01\anaconda3\lib
\site-packages (from selenium) (4.11.0)
Requirement already satisfied: certifi>=2021.10.8 in c:\users\gre01\anaconda3\lib\site-
packages (from selenium) (2021.10.8)
Requirement already satisfied: python-dotenv in c:\users\gre01\anaconda3\lib\site-packa
ges (from webdriver-manager) (1.0.1)
Requirement already satisfied: packaging in c:\users\gre01\anaconda3\lib\site-packages
(from webdriver-manager) (21.0)
Requirement already satisfied: requests in c:\users\gre01\anaconda3\lib\site-packages
(from webdriver-manager) (2.26.0)
Requirement already satisfied: cffi>=1.14 in c:\users\gre01\anaconda3\lib\site-packages
(from trio~=0.17->selenium) (1.14.6)
Requirement already satisfied: outcome in c:\users\gre01\anaconda3\lib\site-packages (f
rom trio~=0.17->selenium) (1.3.0.post0)
```

```
Requirement already satisfied: exceptiongroup in c:\users\gre01\anaconda3\lib\site-pack
ages (from trio~=0.17->selenium) (1.2.1)
Requirement already satisfied: sortedcontainers in c:\users\gre01\anaconda3\lib\site-pa
ckages (from trio~=0.17->selenium) (2.4.0)
Requirement already satisfied: attrs>=23.2.0 in c:\users\gre01\anaconda3\lib\site-packa
ges (from trio~=0.17->selenium) (23.2.0)
Requirement already satisfied: sniffio>=1.3.0 in c:\users\gre01\anaconda3\lib\site-pack
ages (from trio~=0.17->selenium) (1.3.1)
Requirement already satisfied: idna in c:\users\gre01\anaconda3\lib\site-packages (from
trio~=0.17->selenium) (3.2)
Requirement already satisfied: pycparser in c:\users\gre01\anaconda3\lib\site-packages
(from cffi>=1.14->trio~=0.17->selenium) (2.20)
Requirement already satisfied: wsproto>=0.14 in c:\users\gre01\anaconda3\lib\site-packa
ges (from trio-websocket~=0.9->selenium) (1.2.0)
Requirement already satisfied: PySocks!=1.5.7,<2.0,>=1.5.6 in c:\users\gre01\anaconda3
\lib\site-packages (from urllib3[socks]<3,>=1.26->selenium) (1.7.1)
Requirement already satisfied: h11<1,>=0.9.0 in c:\users\gre01\anaconda3\lib\site-packa
ges (from wsproto>=0.14->trio-websocket~=0.9->selenium) (0.14.0)
Requirement already satisfied: pyparsing>=2.0.2 in c:\users\gre01\anaconda3\lib\site-pa
ckages (from packaging->webdriver-manager) (3.0.4)
Requirement already satisfied: charset-normalizer~=2.0.0 in c:\users\gre01\anaconda3\li
b\site-packages (from requests->webdriver-manager) (2.0.4)
Note: you may need to restart the kernel to use updated packages.
```

In [2]:
```
#Install relevent packages for analysis and visualisation
%pip install notebook graphviz pearsonr statsmodels ipywidgets
%pip install plotly --upgrade
```

```
Requirement already satisfied: notebook in c:\users\gre01\anaconda3\lib\site-packages
(6.4.5)
Requirement already satisfied: graphviz in c:\users\gre01\anaconda3\lib\site-packages
(0.20.3)
Requirement already satisfied: pearsonr in c:\users\gre01\anaconda3\lib\site-packages
(0.1.0)
Requirement already satisfied: statsmodels in c:\users\gre01\anaconda3\lib\site-package
s (0.12.2)
Requirement already satisfied: ipywidgets in c:\users\gre01\anaconda3\lib\site-packages
(7.6.5)
Requirement already satisfied: prometheus-client in c:\users\gre01\anaconda3\lib\site-p
ackages (from notebook) (0.11.0)
Requirement already satisfied: tornado>=6.1 in c:\users\gre01\anaconda3\lib\site-packag
es (from notebook) (6.1)
Requirement already satisfied: jinja2 in c:\users\gre01\anaconda3\lib\site-packages (fr
om notebook) (2.11.3)
Requirement already satisfied: traitlets>=4.2.1 in c:\users\gre01\anaconda3\lib\site-pa
ckages (from notebook) (5.1.0)
Requirement already satisfied: nbformat in c:\users\gre01\anaconda3\lib\site-packages
(from notebook) (5.1.3)
Requirement already satisfied: ipykernel in c:\users\gre01\anaconda3\lib\site-packages
(from notebook) (6.4.1)
Requirement already satisfied: Send2Trash>=1.5.0 in c:\users\gre01\anaconda3\lib\site-p
ackages (from notebook) (1.8.0)
Requirement already satisfied: terminado>=0.8.3 in c:\users\gre01\anaconda3\lib\site-pa
ckages (from notebook) (0.9.4)
Requirement already satisfied: jupyter-core>=4.6.1 in c:\users\gre01\anaconda3\lib\site
-packages (from notebook) (4.8.1)
Requirement already satisfied: pyzmq>=17 in c:\users\gre01\anaconda3\lib\site-packages
(from notebook) (22.2.1)
Requirement already satisfied: nbconvert in c:\users\gre01\anaconda3\lib\site-packages
(from notebook) (6.1.0)
Requirement already satisfied: jupyter-client>=5.3.4 in c:\users\gre01\anaconda3\lib\si
te-packages (from notebook) (6.1.12)
Requirement already satisfied: ipython-genutils in c:\users\gre01\anaconda3\lib\site-pa
ckages (from notebook) (0.2.0)
Requirement already satisfied: argon2-cffi in c:\users\gre01\anaconda3\lib\site-package
s (from notebook) (20.1.0)
Requirement already satisfied: numpy>=1.15 in c:\users\gre01\anaconda3\lib\site-package
s (from statsmodels) (1.20.3)
Requirement already satisfied: scipy>=1.1 in c:\users\gre01\anaconda3\lib\site-packages
(from statsmodels) (1.7.1)
Requirement already satisfied: pandas>=0.21 in c:\users\gre01\anaconda3\lib\site-packag
es (from statsmodels) (1.3.4)
Requirement already satisfied: patsy>=0.5 in c:\users\gre01\anaconda3\lib\site-packages
(from statsmodels) (0.5.2)
Requirement already satisfied: jupyterlab-widgets>=1.0.0 in c:\users\gre01\anaconda3\li
b\site-packages (from ipywidgets) (1.0.0)
Requirement already satisfied: widgetsnbextension~=3.5.0 in c:\users\gre01\anaconda3\li
b\site-packages (from ipywidgets) (3.5.1)
Requirement already satisfied: ipython>=4.0.0 in c:\users\gre01\anaconda3\lib\site-pack
ages (from ipywidgets) (7.29.0)
Requirement already satisfied: matplotlib-inline<0.2.0,>=0.1.0 in c:\users\gre01\anacon
da3\lib\site-packages (from ipykernel->notebook) (0.1.2)
Requirement already satisfied: debugpy<2.0,>=1.0.0 in c:\users\gre01\anaconda3\lib\site
-packages (from ipykernel->notebook) (1.4.1)
Requirement already satisfied: setuptools>=18.5 in c:\users\gre01\anaconda3\lib\site-pa
ckages (from ipython>=4.0.0->ipywidgets) (58.0.4)
Requirement already satisfied: backcall in c:\users\gre01\anaconda3\lib\site-packages
(from ipython>=4.0.0->ipywidgets) (0.2.0)
Requirement already satisfied: decorator in c:\users\gre01\anaconda3\lib\site-packages
(from ipython>=4.0.0->ipywidgets) (5.1.0)
Requirement already satisfied: jedi>=0.16 in c:\users\gre01\anaconda3\lib\site-packages
(from ipython>=4.0.0->ipywidgets) (0.18.0)
Requirement already satisfied: pygments in c:\users\gre01\anaconda3\lib\site-packages
(from ipython>=4.0.0->ipywidgets) (2.10.0)
Requirement already satisfied: prompt-toolkit!=3.0.0,!=3.0.1,<3.1.0,>=2.0.0 in c:\users
\gre01\anaconda3\lib\site-packages (from ipython>=4.0.0->ipywidgets) (3.0.20)
Requirement already satisfied: pickleshare in c:\users\gre01\anaconda3\lib\site-package
```

```
s (from ipython>=4.0.0->ipywidgets) (0.7.5)
Requirement already satisfied: colorama in c:\users\gre01\anaconda3\lib\site-packages
(from ipython>=4.0.0->ipywidgets) (0.4.4)
Requirement already satisfied: parso<0.9.0,>=0.8.0 in c:\users\gre01\anaconda3\lib\site
-packages (from jedi>=0.16->ipython>=4.0.0->ipywidgets) (0.8.2)
Requirement already satisfied: python-dateutil>=2.1 in c:\users\gre01\anaconda3\lib\sit
e-packages (from jupyter-client>=5.3.4->notebook) (2.8.2)
Requirement already satisfied: pywin32>=1.0 in c:\users\gre01\anaconda3\lib\site-packag
es (from jupyter-core>=4.6.1->notebook) (228)
Requirement already satisfied: jsonschema!=2.5.0,>=2.4 in c:\users\gre01\anaconda3\lib
\site-packages (from nbformat->notebook) (3.2.0)
Requirement already satisfied: attrs>=17.4.0 in c:\users\gre01\anaconda3\lib\site-packa
ges (from jsonschema!=2.5.0,>=2.4->nbformat->notebook) (23.2.0)
Requirement already satisfied: pyrsistent>=0.14.0 in c:\users\gre01\anaconda3\lib\site-
packages (from jsonschema!=2.5.0,>=2.4->nbformat->notebook) (0.18.0)
Requirement already satisfied: six>=1.11.0 in c:\users\gre01\anaconda3\lib\site-package
s (from jsonschema!=2.5.0,>=2.4->nbformat->notebook) (1.16.0)
Requirement already satisfied: pytz>=2017.3 in c:\users\gre01\anaconda3\lib\site-packag
es (from pandas>=0.21->statsmodels) (2021.3)
Requirement already satisfied: wcwidth in c:\users\gre01\anaconda3\lib\site-packages (f
rom prompt-toolkit!=3.0.0,!=3.0.1,<3.1.0,>=2.0.0->ipython>=4.0.0->ipywidgets) (0.2.5)
Requirement already satisfied: pywinpty>=0.5 in c:\users\gre01\anaconda3\lib\site-packa
ges (from terminado>=0.8.3->notebook) (0.5.7)
Requirement already satisfied: cffi>=1.0.0 in c:\users\gre01\anaconda3\lib\site-package
s (from argon2-cffi->notebook) (1.14.6)
Requirement already satisfied: pycparser in c:\users\gre01\anaconda3\lib\site-packages
(from cffi>=1.0.0->argon2-cffi->notebook) (2.20)
Requirement already satisfied: MarkupSafe>=0.23 in c:\users\gre01\anaconda3\lib\site-pa
ckages (from jinja2->notebook) (1.1.1)
Requirement already satisfied: pandocfilters>=1.4.1 in c:\users\gre01\anaconda3\lib\sit
e-packages (from nbconvert->notebook) (1.4.3)
Requirement already satisfied: testpath in c:\users\gre01\anaconda3\lib\site-packages
(from nbconvert->notebook) (0.5.0)
Requirement already satisfied: jupyterlab-pygments in c:\users\gre01\anaconda3\lib\site
-packages (from nbconvert->notebook) (0.1.2)
Requirement already satisfied: bleach in c:\users\gre01\anaconda3\lib\site-packages (fr
om nbconvert->notebook) (4.0.0)
Requirement already satisfied: mistune<2,>=0.8.1 in c:\users\gre01\anaconda3\lib\site-p
ackages (from nbconvert->notebook) (0.8.4)
Requirement already satisfied: defusedxml in c:\users\gre01\anaconda3\lib\site-packages
(from nbconvert->notebook) (0.7.1)
Requirement already satisfied: nbclient<0.6.0,>=0.5.0 in c:\users\gre01\anaconda3\lib\s
ite-packages (from nbconvert->notebook) (0.5.3)
Requirement already satisfied: entrypoints>=0.2.2 in c:\users\gre01\anaconda3\lib\site-
packages (from nbconvert->notebook) (0.3)
Requirement already satisfied: nest-asyncio in c:\users\gre01\anaconda3\lib\site-packag
es (from nbclient<0.6.0,>=0.5.0->nbconvert->notebook) (1.5.1)
Requirement already satisfied: async-generator in c:\users\gre01\anaconda3\lib\site-pac
kages (from nbclient<0.6.0,>=0.5.0->nbconvert->notebook) (1.10)
Requirement already satisfied: webencodings in c:\users\gre01\anaconda3\lib\site-packag
es (from bleach->nbconvert->notebook) (0.5.1)
Requirement already satisfied: packaging in c:\users\gre01\anaconda3\lib\site-packages
(from bleach->nbconvert->notebook) (21.0)
Requirement already satisfied: pyparsing>=2.0.2 in c:\users\gre01\anaconda3\lib\site-pa
ckages (from packaging->bleach->nbconvert->notebook) (3.0.4)
Note: you may need to restart the kernel to use updated packages.
Requirement already satisfied: plotly in c:\users\gre01\anaconda3\lib\site-packages (5.
22.0)
Requirement already satisfied: tenacity>=6.2.0 in c:\users\gre01\anaconda3\lib\site-pac
kages (from plotly) (8.2.3)
Requirement already satisfied: packaging in c:\users\gre01\anaconda3\lib\site-packages
(from plotly) (21.0)
Requirement already satisfied: pyparsing>=2.0.2 in c:\users\gre01\anaconda3\lib\site-pa
ckages (from packaging->plotly) (3.0.4)
Note: you may need to restart the kernel to use updated packages.
```

World Cup Top 3 Data Collection

In [3]:
```
import requests
from bs4 import BeautifulSoup
import pandas as pd
import numpy as np
```

In [4]:
```
def parse_html_from_url(url):
    #Fetch and parse HTML from a specified URL
    response = requests.get(url)
    response.raise_for_status()
    soup = BeautifulSoup(response.text, 'html.parser')
    return soup
```

In [5]:
```
def extract_data_by_section(soup, section_id):
    #Extract data from a section specified by its ID
    section = soup.find('span', {'id': section_id})
    if section:
        table = section.find_next('table', {'class': 'wikitable'})
        if table:
            return extract_table_data(table)
        else:
            print(f"No table found for {section_id}")
    else:
        print(f"No section found for {section_id}")
    return pd.DataFrame()
```

In [6]:
```
def extract_table_data(table):
    #Extract rows from the specified table
    rows = table.find_all('tr')
    data = []
    for row in rows:
        cols = row.find_all('td')
        if cols:
            year_text = cols[0].text.strip()
            try:
                year = int(year_text)
            except ValueError:
                print(f"Skipping row, invalid year: {year_text}")
                continue
            if year > 2024:
                print(f"Skipping future tournament year: {year}")
                continue
            try:
                champion = cols[2].text.strip()
                runner_up = cols[4].text.strip()
                third = cols[5].text.strip()
                data.append((year_text, champion, runner_up, third))
            except IndexError as e:
                print(f"Error processing row: {row}, error: {e}")
    return pd.DataFrame(data, columns=['Year', 'Champion', 'Runner-up', 'Third'])
```

In [7]:
```
#URL of the Wikipedia page for the Rugby World Cup
url = 'https://en.wikipedia.org/wiki/Rugby_World_Cup'
soup = parse_html_from_url(url)
section_id = 'Tournaments'
World_Cup_Data = extract_data_by_section(soup, section_id)

print("World Cup Data:")
print(World_Cup_Data)
```

```
Skipping future tournament year: 2027
Skipping future tournament year: 2031
World Cup Data:
   Year      Champion    Runner-up         Third
0  1987   New Zealand       France         Wales
1  1991     Australia      England   New Zealand
2  1995  South Africa  New Zealand        France
3  1999     Australia       France  South Africa
4  2003       England    Australia   New Zealand
5  2007  South Africa      England     Argentina
6  2011   New Zealand       France     Australia
7  2015   New Zealand    Australia  South Africa
8  2019  South Africa      England   New Zealand
9  2023  South Africa  New Zealand       England
```

In [8]:
```
World_Cup_Data.to_csv('World_Cup_Data.csv', index=False)
```

World Cup Quarterfinalists Data Collection

In [9]:
```
import time
```

In [10]:
```
#Function to extract top teams from each pool
def extract_top_teams(pool_df):
    top_teams = []
    #Determine the number of rows per pool
    if int(url.split('/')[-1][:4]) <= 1999:
        rows_per_pool = 7
    else:
        rows_per_pool = 8
    #Iterate through each pool
    for pool_start_index in range(0, len(pool_df), rows_per_pool):
        #Get the top two teams from each pool based on points
        top_teams.extend(pool_df.iloc[pool_start_index+2:pool_start_index+4, 0].tolist
())
    return top_teams


world_cup_quarterfinalists = pd.DataFrame(columns=['Year', 'Country'])
```

In [11]:
```
#Iterate through each year
for year in range(1987, 2024, 4):
    try:
        #Construct the URL (page layouts change in the year 2011)
        if year <= 2011:
            url = f'https://www.globalrugbyresults.com/{year}rugbyworldcup.html'
            table_index = 3
        else:
            url = f'https://www.globalrugbyresults.com/{year}RugbyWorldCup.html'
            table_index = 2

        time.sleep(2)

        Rugby_World_Cup_Quarters_df = pd.read_html(url)

        #Extract the relevant DataFrame containing pool stage results
        pool_stage_df = Rugby_World_Cup_Quarters_df[table_index]

        #Extract top teams from each pool
        top_teams = extract_top_teams(pool_stage_df)

        #Add the year to the list of years
        year_list = [year] * len(top_teams)

        year_df = pd.DataFrame({'Year': year_list, 'Country': top_teams})

        #Append the current year's data to world_cup_quarterfinalists
        world_cup_quarterfinalists = world_cup_quarterfinalists.append(year_df, ignore_i
ndex=True)
    except Exception as e:
        print(f"Error processing data for year {year}: {e}")
```

In [12]:
```
print(world_cup_quarterfinalists)
```

```
    Year       Country
0   1987     Australia
1   1987       England
2   1987         Wales
3   1987       Ireland
4   1987   New Zealand
..   ...           ...
77  2023  South Africa
78  2023         Wales
79  2023          Fiji
80  2023       England
81  2023     Argentina

[82 rows x 2 columns]
```

In [13]:
```
world_cup_quarterfinalists.to_csv('world_cup_quarterfinalists.csv', index=False)
```

Six Nations Data Collection

In [14]:
```
from selenium import webdriver
from selenium.webdriver.common.by import By
from selenium.webdriver.chrome.service import Service
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.support import expected_conditions as EC
from webdriver_manager.chrome import ChromeDriverManager
```

In [15]:
```
def fetch_page(url):
    options = webdriver.ChromeOptions()
    options.add_argument('--headless')
    driver = webdriver.Chrome(service=Service(ChromeDriverManager().install()), options=
options)
    driver.get(url)
    return driver
```

In [16]:
```
def scrape_season_data(driver):
    try:
        standings = []
        #Wait for the Javascript driven dynamic content to load
        WebDriverWait(driver, 3).until(EC.presence_of_element_located((By.CSS_SELECTOR,
".ui-table__row")))
        rows = driver.find_elements(By.CSS_SELECTOR, ".ui-table__row")
        #Debug
        #print(f"Found {len(rows)} rows")
        for row in rows:
            rank = row.find_element(By.CSS_SELECTOR, ".table__cell--rank .tableCellRan
k").text.strip()
            team_name = row.find_element(By.CSS_SELECTOR, ".tableCellParticipant__nam
e").text.strip()
            total_points = row.find_element(By.CSS_SELECTOR, ".table__cell--points").tex
t.strip()
            total_playing_points = row.find_element(By.CSS_SELECTOR, ".table__cell--tota
lPoints").text.strip()
            standings.append({'Rank': rank, 'Team': team_name, 'Points': total_points,
'Scored Points': total_playing_points})
            #Debug
            #print(f"Scraped data - Rank: {rank}, Team: {team_name}, Points: {total_poin
ts}")
        return standings
    except Exception as e:
        print(f"Error scraping data: {e}")
        return None
```

In [17]:
```
def compile_data(url):
    main_driver = fetch_page(url)
    #Filter only the links of interest
    links = main_driver.find_elements(By.CSS_SELECTOR, 'a[href*="/six-nations-20"]:not
([href*="standings"])')
    #Debug so I dont have to go through all 24 years each time
    #year_urls = [link.get_attribute('href') for link in links if '2024' in link.text or
'2023' in link.text]
    year_urls = [link.get_attribute('href') for link in links]

    results = []
    for year_url in year_urls:
        print(f"Processing {year_url}")
        driver_year = fetch_page(year_url)
        standings_link = driver_year.find_element(By.CSS_SELECTOR, 'a[href*="standing
s"]')
        standings_url = standings_link.get_attribute('href')

        driver_standings = fetch_page(standings_url)
        teams = scrape_season_data(driver_standings)
        if teams:
            #Extract year from the URL
            year = standings_url.split('/')[-2].split('-')[-1] if 'standings' in standin
gs_url.split('/')[-1] else standings_url.split('/')[-3].split('-')[-1]
            for team in teams:
                results.append([year] + list(team.values()))
        driver_standings.quit()
        driver_year.quit()

    main_driver.quit()
    return pd.DataFrame(results, columns=['Year', 'Rank', 'Team', 'Points', 'Scored Poin
ts'])
```

In [18]:
```
main_url = 'https://www.livesport.com/en/rugby-union/europe/six-nations/archive/'
six_nations_data = compile_data(main_url)
print(six_nations_data)
```

```
Processing https://www.livesport.com/en/rugby-union/europe/six-nations-2024/
Processing https://www.livesport.com/en/rugby-union/europe/six-nations-2023/
Processing https://www.livesport.com/en/rugby-union/europe/six-nations-2022/
Processing https://www.livesport.com/en/rugby-union/europe/six-nations-2021/
Processing https://www.livesport.com/en/rugby-union/europe/six-nations-2020/
Processing https://www.livesport.com/en/rugby-union/europe/six-nations-2019/
Processing https://www.livesport.com/en/rugby-union/europe/six-nations-2018/
Processing https://www.livesport.com/en/rugby-union/europe/six-nations-2017/
Processing https://www.livesport.com/en/rugby-union/europe/six-nations-2016/
Processing https://www.livesport.com/en/rugby-union/europe/six-nations-2015/
Processing https://www.livesport.com/en/rugby-union/europe/six-nations-2014/
Processing https://www.livesport.com/en/rugby-union/europe/six-nations-2013/
Processing https://www.livesport.com/en/rugby-union/europe/six-nations-2012/
Processing https://www.livesport.com/en/rugby-union/europe/six-nations-2011/
Processing https://www.livesport.com/en/rugby-union/europe/six-nations-2010/
Processing https://www.livesport.com/en/rugby-union/europe/six-nations-2009/
Processing https://www.livesport.com/en/rugby-union/europe/six-nations-2008/
Processing https://www.livesport.com/en/rugby-union/europe/six-nations-2007/
Processing https://www.livesport.com/en/rugby-union/europe/six-nations-2006/
Processing https://www.livesport.com/en/rugby-union/europe/six-nations-2005/
Processing https://www.livesport.com/en/rugby-union/europe/six-nations-2004/
Processing https://www.livesport.com/en/rugby-union/europe/six-nations-2003/
Processing https://www.livesport.com/en/rugby-union/europe/six-nations-2002/
Processing https://www.livesport.com/en/rugby-union/europe/six-nations-2001/
Processing https://www.livesport.com/en/rugby-union/europe/six-nations-2000/
     Year Rank      Team Points Scored Points
0    2024   1.   Ireland     20        144:60
1    2024   2.    France     15       128:122
2    2024   3.   England     14       118:123
3    2024   4.  Scotland     12       115:115
4    2024   5.     Italy     11        92:126
..    ...  ...       ...    ...           ...
145  2000   2.    France      6        140:92
146  2000   3.   Ireland      6       168:133
147  2000   4.     Wales      6       111:135
148  2000   5.  Scotland      2        95:145
149  2000   6.     Italy      2       106:228

[150 rows x 5 columns]
```

In [19]:
```
print(six_nations_data.columns)
```

```
Index(['Year', 'Rank', 'Team', 'Points', 'Scored Points'], dtype='object')
```

In [20]:
```
#Expand 'Scored Points' column into two new columns
six_nations_data[['Points Scored', 'Points Conceded']] = six_nations_data['Scored Point
s'].str.split(':', expand=True)

six_nations_data['Points Scored'] = pd.to_numeric(six_nations_data['Points Scored'])
six_nations_data['Points Conceded'] = pd.to_numeric(six_nations_data['Points Conceded'])

six_nations_data['Points Difference'] = six_nations_data['Points Scored'] - six_nations_
data['Points Conceded']

#Remove 'Scored Points' column
six_nations_data = six_nations_data.drop(columns=['Scored Points'])

print(six_nations_data.head())
```

```
   Year Rank      Team Points  Points Scored  Points Conceded  \
0  2024   1.   Ireland     20            144               60
1  2024   2.    France     15            128              122
2  2024   3.   England     14            118              123
3  2024   4.  Scotland     12            115              115
4  2024   5.     Italy     11             92              126

   Points Difference
0                 84
1                  6
2                 -5
3                  0
4                -34
```

In [21]:
```
six_nations_data.to_csv('Six_Nations_data.csv', index=False)
```

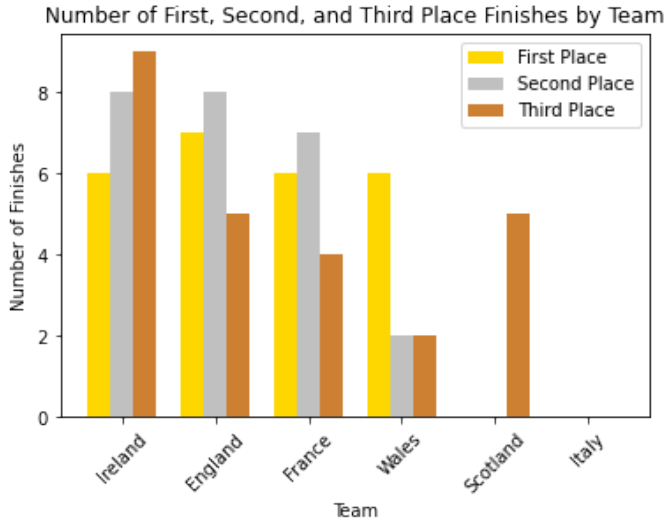
Six Nations Data Analysis

In [22]:
```
import matplotlib.pyplot as plt
import seaborn as sns
```

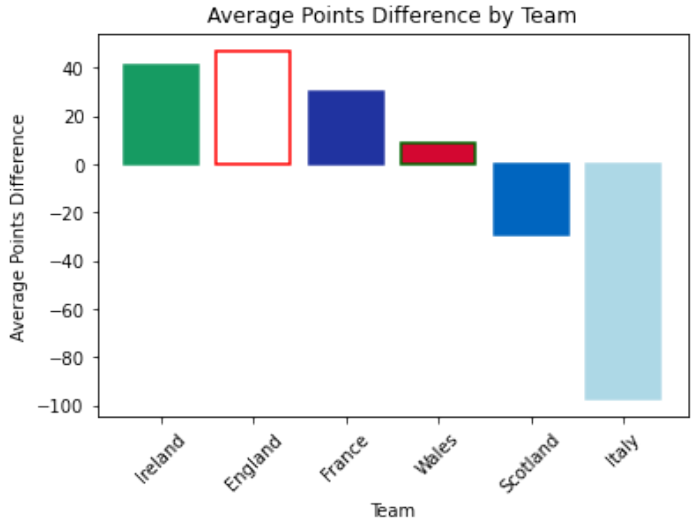
In [23]:
```
#Converting 'Rank' to numeric
six_nations_data['Rank'] = pd.to_numeric(six_nations_data['Rank'], errors='coerce')

#Number of of individual top 3 finishes
first_place_counts = six_nations_data[six_nations_data['Rank'] == 1].groupby('Team').siz
e()
second_place_counts = six_nations_data[six_nations_data['Rank'] == 2].groupby('Team').si
ze()
third_place_counts = six_nations_data[six_nations_data['Rank'] == 3].groupby('Team').siz
e()

#Average rank for each team
average_rank = six_nations_data.groupby('Team')['Rank'].mean()

#Average points difference for each team
average_points_difference = six_nations_data.groupby('Team')['Points Difference'].mean()

first_place_counts, average_rank.sort_values(), average_points_difference.sort_values(as
cending=False)
```

Out[23]:
```
(Team
 England    7
 France     6
 Ireland    6
 Wales      6
 dtype: int64,
 Team
 Ireland     2.32
 England     2.40
 France      2.72
 Wales       3.48
 Scotland    4.44
 Italy       5.64
 Name: Rank, dtype: float64,
 Team
 England     47.24
 Ireland     40.72
 France      29.88
 Wales        8.56
 Scotland   -28.96
 Italy      -97.44
 Name: Points Difference, dtype: float64)
```

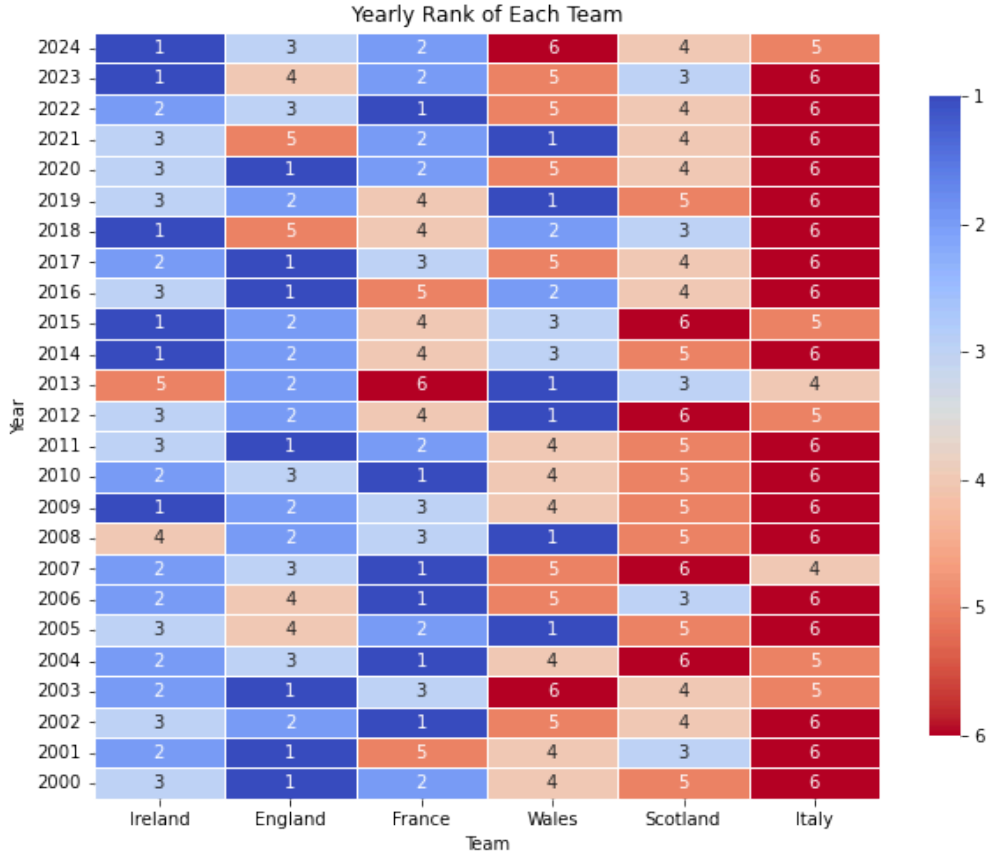
In [24]:
```
all_teams = six_nations_data['Team'].unique()

#Count the first, second, and third place finishes
first_place_counts = six_nations_data[six_nations_data['Rank'] == 1].groupby('Team').siz
e().reindex(all_teams, fill_value=0)
second_place_counts = six_nations_data[six_nations_data['Rank'] == 2].groupby('Team').si
ze().reindex(all_teams, fill_value=0)
third_place_counts = six_nations_data[six_nations_data['Rank'] == 3].groupby('Team').siz
e().reindex(all_teams, fill_value=0)

#Combining all place counts to maintain a consistent order based on overall performance
combined_counts = first_place_counts + second_place_counts + third_place_counts
sorted_teams = combined_counts.sort_values(ascending=False).index

#Sort the first, second, and third place counts based on the sorted order
first_place_counts = first_place_counts.reindex(sorted_teams)
second_place_counts = second_place_counts.reindex(sorted_teams)
third_place_counts = third_place_counts.reindex(sorted_teams)

fig, ax = plt.subplots()
index = range(len(all_teams))
bar_width = 0.25

#Plotting all three sets of bars
bars1 = ax.bar(index, first_place_counts, bar_width, label='First Place', color='gold')
bars2 = ax.bar([p + bar_width for p in index], second_place_counts, bar_width, label='Se
cond Place', color='silver')
bars3 = ax.bar([p + 2 * bar_width for p in index], third_place_counts, bar_width, label
='Third Place', color='#cd7f32')

ax.set_xlabel('Team')
ax.set_ylabel('Number of Finishes')
ax.set_title('Number of First, Second, and Third Place Finishes by Team')
ax.set_xticks([p + bar_width for p in index])
ax.set_xticklabels(sorted_teams, rotation=45)
ax.legend()

plt.show()
```



In [25]:
```
#Define team colors for clarity
team_colours_2 = {
    'Ireland': '#169b62',
    'France': '#2033a0',
    'England': '#FFFFFF',
    'Scotland': '#0065bf',
    'Italy': '#ADD8E6',
    'Wales': '#d30731'
}

#Define border colors
border_colors = {
    'Ireland': '#169b62',
    'France': '#2033a0',
    'England': 'red',
    'Scotland': '#0065bf',
    'Italy': '#ADD8E6',
    'Wales': 'darkgreen'
}

#Average points difference for each team
average_points_difference = six_nations_data.groupby('Team')['Points Difference'].mean()
average_points_difference_sorted = average_points_difference.reindex(sorted_teams)

fig, ax = plt.subplots()
bars = ax.bar(
    average_points_difference_sorted.index,
    average_points_difference_sorted.values,
    color=[team_colours_2[team] for team in average_points_difference_sorted.index],
    edgecolor=[border_colors[team] for team in average_points_difference_sorted.index],
    linewidth=1.5
)

ax.set_title('Average Points Difference by Team')
ax.set_xlabel('Team')
ax.set_ylabel('Average Points Difference')
plt.xticks(rotation=45)
plt.show()
```



In [26]:
```
#Pivot data for heatmap
rank_heatmap_data = six_nations_data.pivot_table(index='Year', columns='Team', values='Rank')
rank_heatmap_data = rank_heatmap_data.reindex(columns=sorted_teams)

#Reverse the order of columns
rank_heatmap_data = rank_heatmap_data.iloc[:, ::-1]

plt.figure(figsize=(10, 8))
sns.heatmap(rank_heatmap_data, annot=True, cmap='coolwarm', fmt=".0f", linewidths=.5)

#Invert y-axis
plt.gca().invert_yaxis()

#Order Ireland to Italy like above
plt.gca().invert_xaxis()

#Order gradient bar
plt.gcf().axes[1].set_ylim(6.5, 0.5)

plt.title('Yearly Rank of Each Team')
plt.xlabel('Team')
plt.ylabel('Year')
plt.show()
```



Investment Analysis

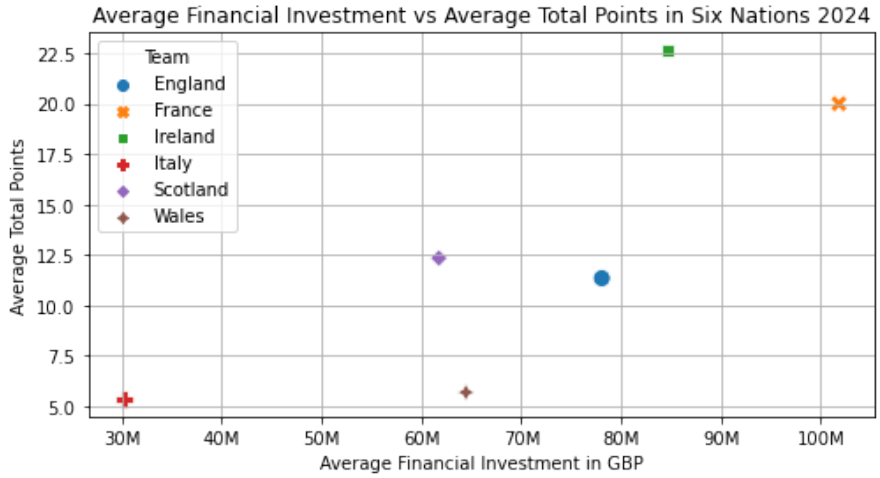
In [27]:
```
investment_data_path = '6_nations_teams_investments.csv'
investment_data = pd.read_csv(investment_data_path)

print(investment_data)
```

```
    Year      Team  Expenditure (£/€)  Expenditure (£)
0   2023   England         99400000.0     9.940000e+07
1   2023    France        142944183.0     1.223639e+08
2   2023   Ireland         93118736.0     7.971203e+07
3   2023     Italy         37496556.9     3.209802e+07
4   2023  Scotland         75382000.0     7.538200e+07
5   2023     Wales         79500000.0     7.950000e+07
6   2022   England         77400000.0     7.740000e+07
7   2022    France        125699609.0     1.076021e+08
8   2022   Ireland        109686828.0     9.389475e+07
9   2022     Italy         36037805.5     3.084929e+07
10  2022  Scotland         61051000.0     6.105100e+07
11  2022     Wales         70100000.0     7.010000e+07
12  2021   England         57400000.0     5.740000e+07
13  2021    France         88150169.0     7.545881e+07
14  2021   Ireland         94093057.0     8.054608e+07
15  2021     Italy         32314997.0     2.766247e+07
16  2021  Scotland         48811000.0     4.881100e+07
17  2021     Wales         43800000.0     4.380000e+07
```

In [28]:
```
#Cleaning data
six_nations_data['Year'] = six_nations_data['Year'].astype(str).str.strip()
investment_data['Year'] = investment_data['Year'].astype(str).str.strip()
six_nations_data['Points'] = pd.to_numeric(six_nations_data['Points'], errors='coerce')

#Filter Six Nations data for the years 2022 to 2024
six_nations_filtered = six_nations_data[six_nations_data['Year'].isin(['2024', '2023',
'2022'])]

average_scores = six_nations_filtered.groupby('Team')['Points'].mean().reset_index()
average_investment = investment_data.groupby('Team')['Expenditure (£)'].mean().reset_ind
ex()

#Merge the averaged scores and investments
merged_investment_data = pd.merge(average_scores, average_investment, on='Team', how='in
ner')
print(merged_investment_data)
```

```
      Team     Points  Expenditure (£)
0   England  11.333333     7.806667e+07
1    France  20.000000     1.018083e+08
2   Ireland  22.666667     8.471762e+07
3     Italy   5.333333     3.020326e+07
4  Scotland  12.333333     6.174800e+07
5     Wales   5.666667     6.446667e+07
```

In [29]:
```
import matplotlib.ticker as ticker

#Plotting the relationship between Investment and Ranking
plt.figure(figsize=(8, 4))
ax = sns.scatterplot(data=merged_investment_data, x='Expenditure (£)', y='Points', hue
='Team', s=100, style='Team')

plt.title('Average Financial Investment vs Average Total Points in Six Nations 2024')
plt.xlabel('Average Financial Investment in GBP')
plt.ylabel('Average Total Points')
plt.grid(True)

#Format the x-axis
ax.xaxis.set_major_formatter(ticker.FuncFormatter(lambda x, pos: '{:,.0f}M'.format(x/1e
6)))

plt.show()
```



In [30]:
```
print(investment_data)
```

	Year	Team	Expenditure (£/€)	Expenditure (£)
0	2023	England	99400000.0	9.940000e+07
1	2023	France	142944183.0	1.223639e+08
2	2023	Ireland	93118736.0	7.971203e+07
3	2023	Italy	37496556.9	3.209802e+07
4	2023	Scotland	75382000.0	7.538200e+07
5	2023	Wales	79500000.0	7.950000e+07
6	2022	England	77400000.0	7.740000e+07
7	2022	France	125699609.0	1.076021e+08
8	2022	Ireland	109686828.0	9.389475e+07
9	2022	Italy	36037805.5	3.084929e+07
10	2022	Scotland	61051000.0	6.105100e+07
11	2022	Wales	70100000.0	7.010000e+07
12	2021	England	57400000.0	5.740000e+07
13	2021	France	88150169.0	7.545881e+07
14	2021	Ireland	94093057.0	8.054608e+07
15	2021	Italy	32314997.0	2.766247e+07
16	2021	Scotland	48811000.0	4.881100e+07
17	2021	Wales	43800000.0	4.380000e+07

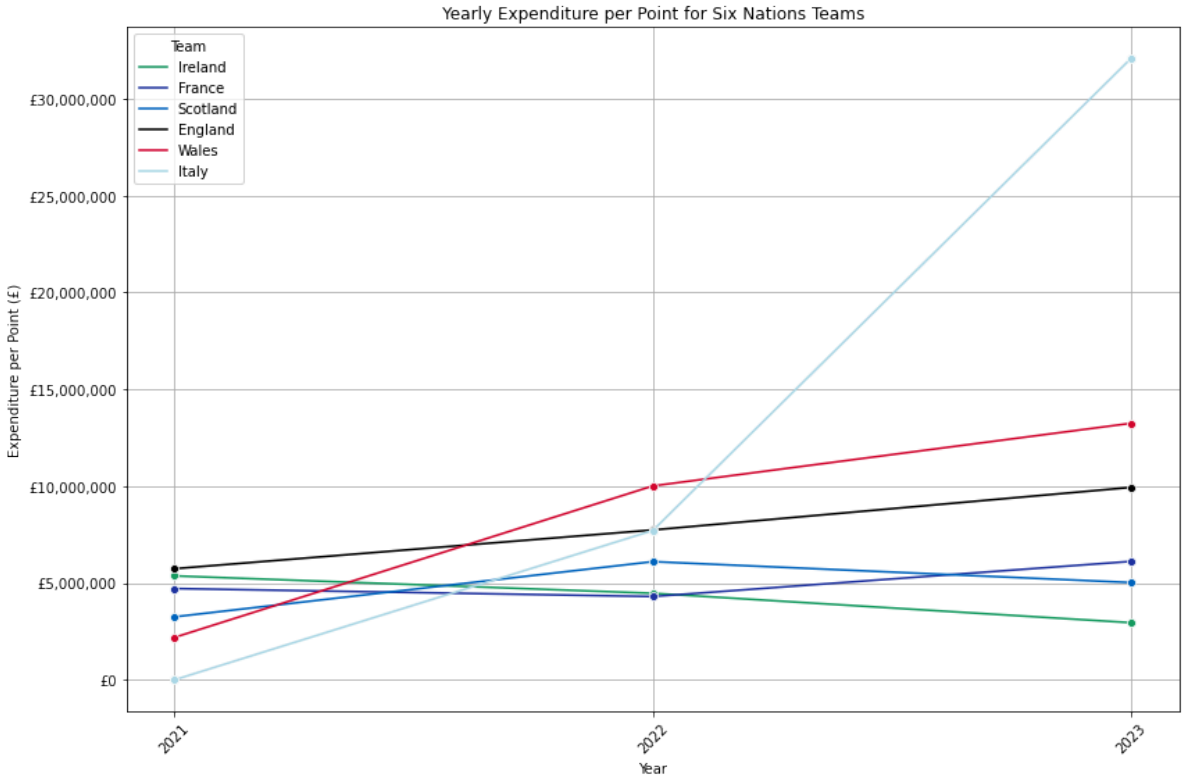
In [31]:
```
merged_data_1 = pd.merge(six_nations_data, investment_data, on=['Year', 'Team'])
print(merged_data_1)
```

```
    Year  Rank      Team  Points  Points Scored  Points Conceded  \
0   2023   1.0   Ireland      27            151               72
1   2023   2.0    France      20            174              115
2   2023   3.0  Scotland      15            118               98
3   2023   4.0   England      10            100              135
4   2023   5.0     Wales       6             84              147
5   2023   6.0     Italy       1             89              149
6   2022   1.0    France      25            141               73
7   2022   2.0   Ireland      21            168               63
8   2022   3.0   England      10            101               96
9   2022   4.0  Scotland      10             92              121
10  2022   5.0     Wales       7             76              104
11  2022   6.0     Italy       4             60              181
12  2021   1.0     Wales      20            164              103
13  2021   2.0    France      16            140              103
14  2021   3.0   Ireland      15            136               88
15  2021   4.0  Scotland      15            138               91
16  2021   5.0   England      10            112              121
17  2021   6.0     Italy       0             55              239

    Points Difference  Expenditure (£/€)  Expenditure (£)
0                  79         93118736.0     7.971203e+07
1                  59        142944183.0     1.223639e+08
2                  20         75382000.0     7.538200e+07
3                 -35         99400000.0     9.940000e+07
4                 -63         79500000.0     7.950000e+07
5                 -60         37496556.9     3.209802e+07
6                  68        125699609.0     1.076021e+08
7                 105        109686828.0     9.389475e+07
8                   5         77400000.0     7.740000e+07
9                 -29         61051000.0     6.105100e+07
10                -28         70100000.0     7.010000e+07
11               -121         36037805.5     3.084929e+07
12                 61         43800000.0     4.380000e+07
13                 37         88150169.0     7.545881e+07
14                 48         94093057.0     8.054608e+07
15                 47         48811000.0     4.881100e+07
16                 -9         57400000.0     5.740000e+07
17               -184         32314997.0     2.766247e+07
```

In [32]:
```
#Clean data
merged_data_1['Points'] = merged_data_1['Points'].astype(int)
merged_data_1['Year'] = merged_data_1['Year'].astype(int)

#Calculate 'Expenditure per Point (£)'
merged_data_1['Expenditure per Point (£)'] = merged_data_1['Expenditure (£)'] / merged_d
ata_1['Points']

#Handling infinite values
merged_data_1['Expenditure per Point (£)'].replace([np.inf, -np.inf], np.nan, inplace=Tr
ue)
merged_data_1['Expenditure per Point (£)'].fillna(0, inplace=True)

print("\nFinal Data with Expenditure per Point:\n")
print(merged_data_1[['Year', 'Team', 'Points', 'Expenditure (£)', 'Expenditure per Point
(£)']])
```

```
Final Data with Expenditure per Point:

    Year      Team  Points  Expenditure (£)  Expenditure per Point (£)
0   2023   Ireland      27     7.971203e+07               2.952298e+06
1   2023    France      20     1.223639e+08               6.118195e+06
2   2023  Scotland      15     7.538200e+07               5.025467e+06
3   2023   England      10     9.940000e+07               9.940000e+06
4   2023     Wales       6     7.950000e+07               1.325000e+07
5   2023     Italy       1     3.209802e+07               3.209802e+07
6   2022    France      25     1.076021e+08               4.304084e+06
7   2022   Ireland      21     9.389475e+07               4.471178e+06
8   2022   England      10     7.740000e+07               7.740000e+06
9   2022  Scotland      10     6.105100e+07               6.105100e+06
10  2022     Wales       7     7.010000e+07               1.001429e+07
11  2022     Italy       4     3.084929e+07               7.712322e+06
12  2021     Wales      20     4.380000e+07               2.190000e+06
13  2021    France      16     7.545881e+07               4.716176e+06
14  2021   Ireland      15     8.054608e+07               5.369738e+06
15  2021  Scotland      15     4.881100e+07               3.254067e+06
16  2021   England      10     5.740000e+07               5.740000e+06
17  2021     Italy       0     2.766247e+07               0.000000e+00
```

In [33]:
```
#Define custom palette
team_colours = {
    'Ireland': '#169b62',
    'France': '#2033a0',
    'England': '#000000',
    'Scotland': '#0065bf',
    'Italy': '#ADD8E6',
    'Wales': '#d30731'
}

plt.figure(figsize=(12, 8))
ax = sns.lineplot(data=merged_data_1, x='Year', y='Expenditure per Point (£)', hue='Tea
m', marker='o', palette=team_colours)

plt.title('Yearly Expenditure per Point for Six Nations Teams')
plt.xlabel('Year')
plt.ylabel('Expenditure per Point (£)')
plt.grid(True)

ax.set_xticks(merged_data_1['Year'].unique())
ax.set_xticklabels(merged_data_1['Year'].unique(), rotation=45)

#Format the y-axis to display currency
formatter = ticker.FuncFormatter(lambda x, pos: f'£{x:,.0f}')
ax.yaxis.set_major_formatter(formatter)

plt.legend(title='Team')
plt.tight_layout()
plt.show()
```

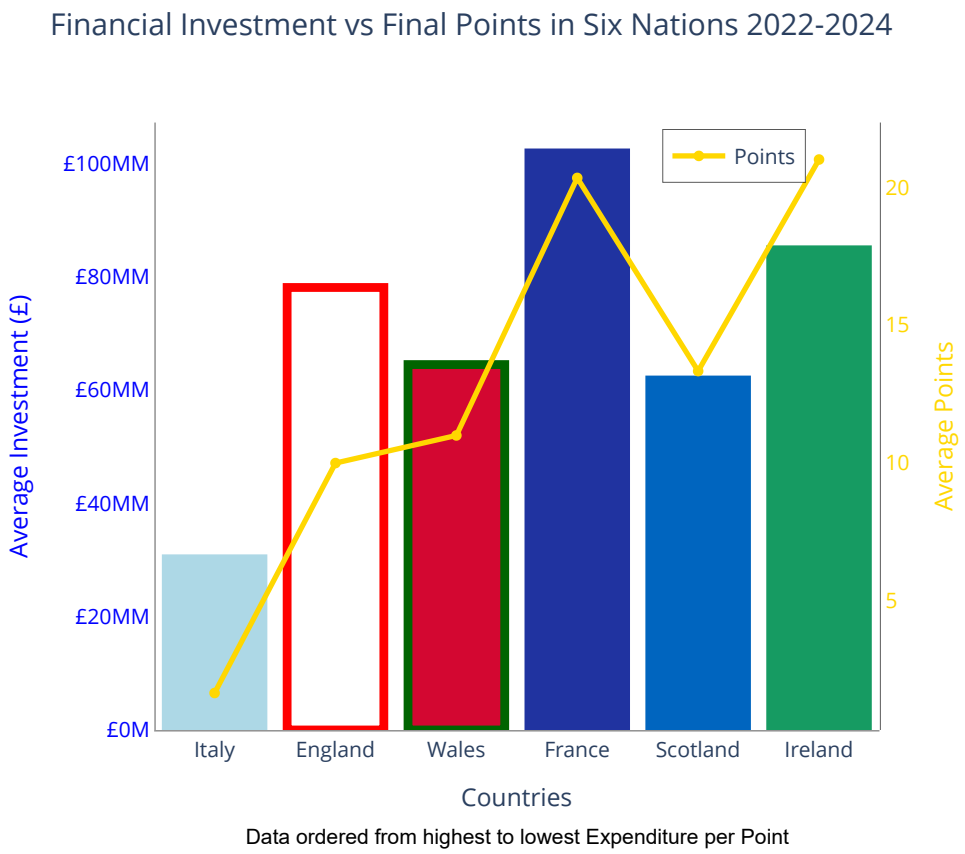


In [34]:
```
from scipy.stats import pearsonr

#Correlation for investment and total points
investment_correlation, _ = pearsonr(merged_data_1['Expenditure (£)'], merged_data_1['Po
ints'])
print("Correlation coefficient between expenditure and points:", investment_correlation)
```

```
Correlation coefficient between expenditure and points: 0.6234089626119186
```

In [35]:
```
import plotly.graph_objects as go
import plotly.io as pio
import plotly.express as px

pio.renderers.default = "notebook"

#Calculate the averages for teams over time
average_data = merged_data_1.groupby('Team').agg({
    'Expenditure (£)': 'mean',
    'Points': 'mean'
}).reset_index()

average_data['Expenditure per Point (£)'] = average_data['Expenditure (£)'] / average_da
ta['Points']

#Sort the data from highest to lowest Expenditure per Point
average_data = average_data.sort_values(by='Expenditure per Point (£)', ascending=False)

#Create the Plotly figure
fig = go.Figure()

#Add bars for Expenditure
fig.add_trace(go.Bar(
    x=average_data['Team'],
    y=average_data['Expenditure (£)'],
    name='Average Expenditure (£)',
    #Apply colours
    marker_color=[team_colours_2[team] for team in average_data['Team']],
    marker_line_color=[border_colors[team] for team in average_data['Team']],
    marker_line_width=5,
    #Add hover pop-up box with key info
    hoverinfo="text",
    hovertemplate=(
        "Team: %{x}<br>" +
        "Average Expenditure: £%{y:,.0f}<br>" +
        "Expenditure per Point: £%{customdata:.2f}<extra></extra>"
    ),
    customdata=average_data['Expenditure per Point (£)'],
    showlegend=False
))

#Add a line for Points with secondary Y-axis
fig.add_trace(go.Scatter(
    x=average_data['Team'],
    y=average_data['Points'],
    name='Points',
    mode='lines+markers',
    line=dict(color='gold', width=3),
    yaxis='y2'
))

#layout of both y-axis
fig.update_layout(
    title='Financial Investment vs Final Points in Six Nations 2022-2024',
    xaxis_title='Countries',
    yaxis=dict(
        title='Average Investment (£)',
        titlefont=dict(color='blue'),
        tickfont=dict(color='blue'),
        tickprefix="£",
        ticksuffix="M",
        position=0
    ),
    yaxis2=dict(
        title='Average Points',
        titlefont=dict(color='gold'),
        tickfont=dict(color='gold'),
        overlaying='y',
        side='right',
```

```
        position=1
    ),
    plot_bgcolor='white',
    legend=dict(x=0.70, y=0.99, bordercolor='black', borderwidth=1)
)

#Remove grid lines for clarity
fig.update_xaxes(showgrid=False, linewidth=1, linecolor='black')
fig.update_yaxes(showgrid=False, linewidth=1, linecolor='black')


#Add a subnote
fig.add_annotation(
    x=0.5,
    y=-0.2,
    xref="paper",
    yref="paper",
    text="Data ordered from highest to lowest Expenditure per Point",
    showarrow=False,
    font=dict(
        family="Arial, sans-serif",
        size=12,
        color="black"
    ),
    align="center"
)

fig.write_html("Interactive Version.html")

fig.show()
```



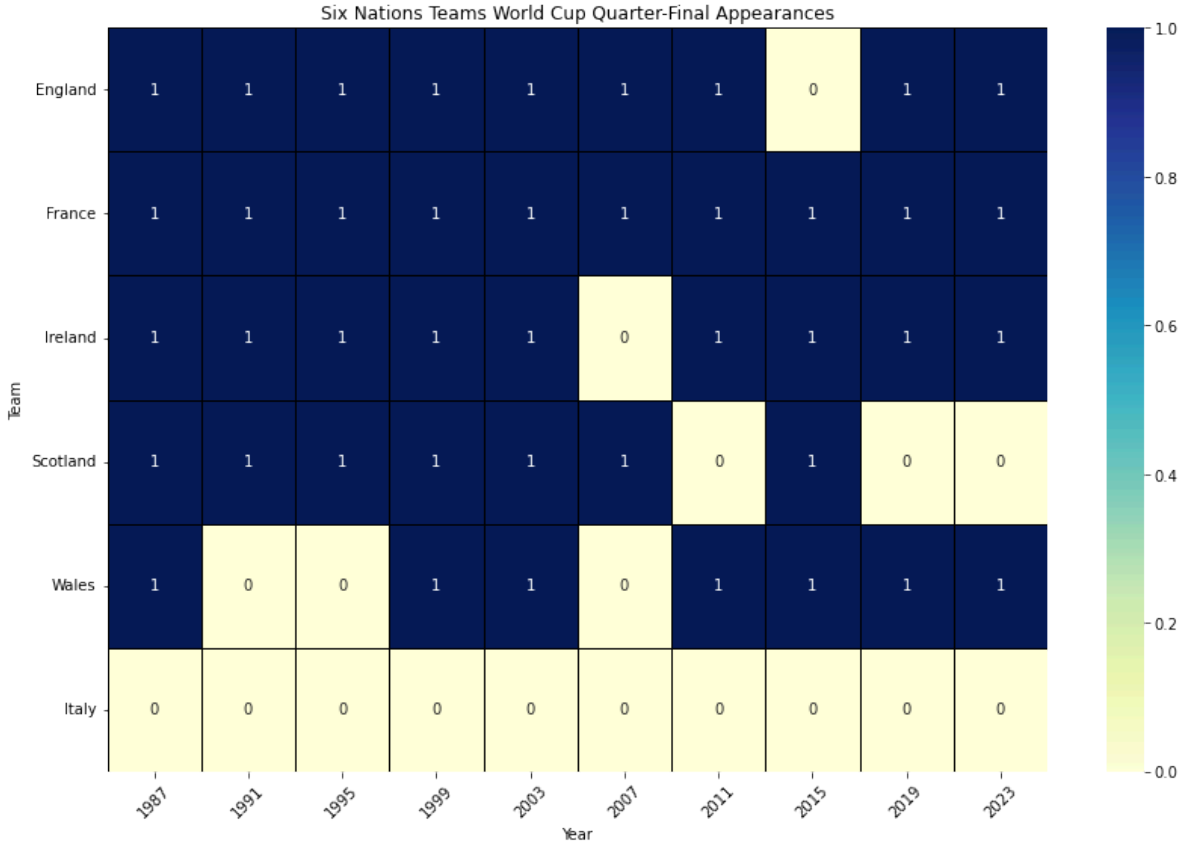
World Cup Quarter Finalists (Top 8) Analysis

In [36]:
```
#Filter the data to include only Six Nations teams
six_nations_teams = ['England', 'Scotland', 'Wales', 'Ireland', 'France', 'Italy']
six_nations_data_teams = world_cup_quarterfinalists[world_cup_quarterfinalists['Countr
y'].isin(six_nations_teams)]

#Pivot the dataFrame to get the count of appearances for each team in each year
heatmap_data_quarters = six_nations_data_teams.pivot_table(index='Country', columns='Yea
r', aggfunc='size', fill_value=0)

#Not all Six Nations teams will have made the quarter finals ever, so append them in for
consistency
for team in six_nations_teams:
    if team not in heatmap_data_quarters.index:
        heatmap_data_quarters.loc[team] = [0] * len(heatmap_data_quarters.columns)

print(heatmap_data_quarters)
```

```
Year      1987  1991  1995  1999  2003  2007  2011  2015  2019  2023
Country
England      1     1     1     1     1     1     1     0     1     1
France       1     1     1     1     1     1     1     1     1     1
Ireland      1     1     1     1     1     0     1     1     1     1
Scotland     1     1     1     1     1     1     0     1     0     0
Wales        1     0     0     1     1     0     1     1     1     1
Italy        0     0     0     0     0     0     0     0     0     0
```

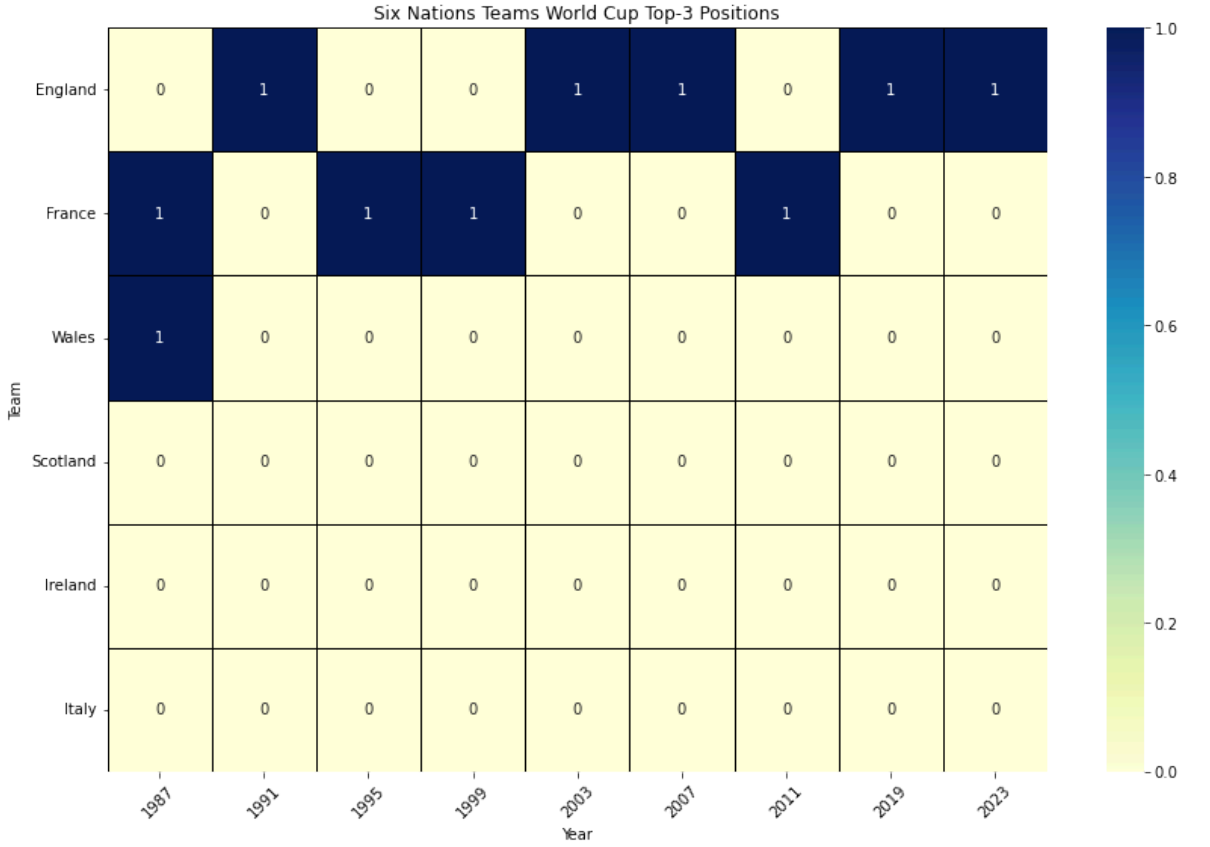
Visualisation of Quarterfinals Apperences of Six Nations Teams

In [37]:
```
#Create the heatmap
plt.figure(figsize=(12, 8))
sns.heatmap(heatmap_data_quarters, cmap='YlGnBu', annot=True, fmt='d', linewidths=0.5, l
inecolor='black')
plt.title('Six Nations Teams World Cup Quarter-Final Appearances')
plt.xlabel('Year')
plt.ylabel('Team')
plt.xticks(rotation=45)
plt.yticks(rotation=0)
plt.tight_layout()
plt.show()
```

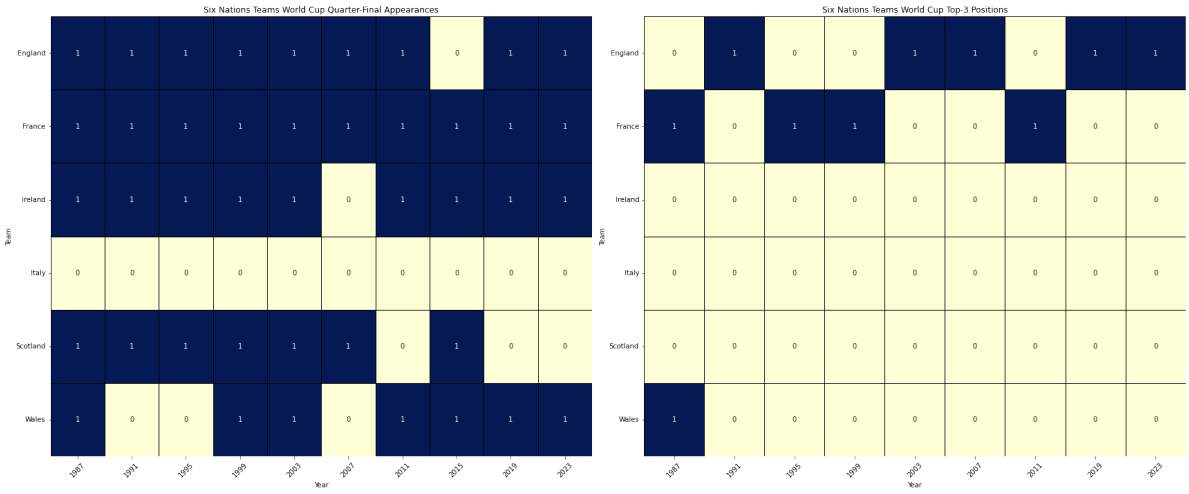


In [38]:
```
#Reshape the dataFrame
long_format = pd.melt(World_Cup_Data, id_vars='Year', value_vars=['Champion', 'Runner-u
p', 'Third'], var_name='Placement', value_name='Country')
filtered_data = long_format[long_format['Country'].isin(six_nations_teams)]

#Pivot table
heatmap_data_top3 = filtered_data.pivot_table(index='Country', columns='Year', aggfunc
='size', fill_value=0)

#Same as above not all teams from the Six Nations will have reached a Top 3 position in
the World Cup
for team in six_nations_teams:
    if team not in heatmap_data_top3.index:
        heatmap_data_top3.loc[team] = [0] * len(heatmap_data_top3.columns)

#Ensure the team order in the heatmap matches quarter finalists
heatmap_data = heatmap_data_top3.loc[six_nations_teams]
```

In [39]:
```
#Create the heatmap
plt.figure(figsize=(12, 8))
sns.heatmap(heatmap_data_top3, cmap='YlGnBu', annot=True, fmt='d', linewidths=0.5, linec
olor='black')
plt.title('Six Nations Teams World Cup Top-3 Positions')
plt.xlabel('Year')
plt.ylabel('Team')
plt.xticks(rotation=45)
plt.yticks(rotation=0)
plt.tight_layout()
plt.show()
```



In [40]:
```
#Sort the indices of the DataFrame
heatmap_data_quarters = heatmap_data_quarters.sort_index()
heatmap_data_top3 = heatmap_data_top3.sort_index()

#Combine plots into one frame
fig, axes = plt.subplots(nrows=1, ncols=2, figsize=(24, 10))

#World Cup Quarter-Final Appearances
sns.heatmap(heatmap_data_quarters, cmap='YlGnBu', annot=True, fmt='d', linewidths=0.5, l
inecolor='black', ax=axes[0], cbar=False)
axes[0].set_title('Six Nations Teams World Cup Quarter-Final Appearances')
axes[0].set_xlabel('Year')
axes[0].set_ylabel('Team')
axes[0].set_xticklabels(axes[0].get_xticklabels(), rotation=45)
axes[0].set_yticklabels(axes[0].get_yticklabels(), rotation=0)

#World Cup Top-3 Appearances
sns.heatmap(heatmap_data_top3, cmap='YlGnBu', annot=True, fmt='d', linewidths=0.5, linec
olor='black', ax=axes[1], cbar=False)
axes[1].set_title('Six Nations Teams World Cup Top-3 Positions')
axes[1].set_xlabel('Year')
axes[1].set_ylabel('Team')
axes[1].set_xticklabels(axes[1].get_xticklabels(), rotation=45)
axes[1].set_yticklabels(axes[1].get_yticklabels(), rotation=0)

plt.tight_layout()
plt.show()
```

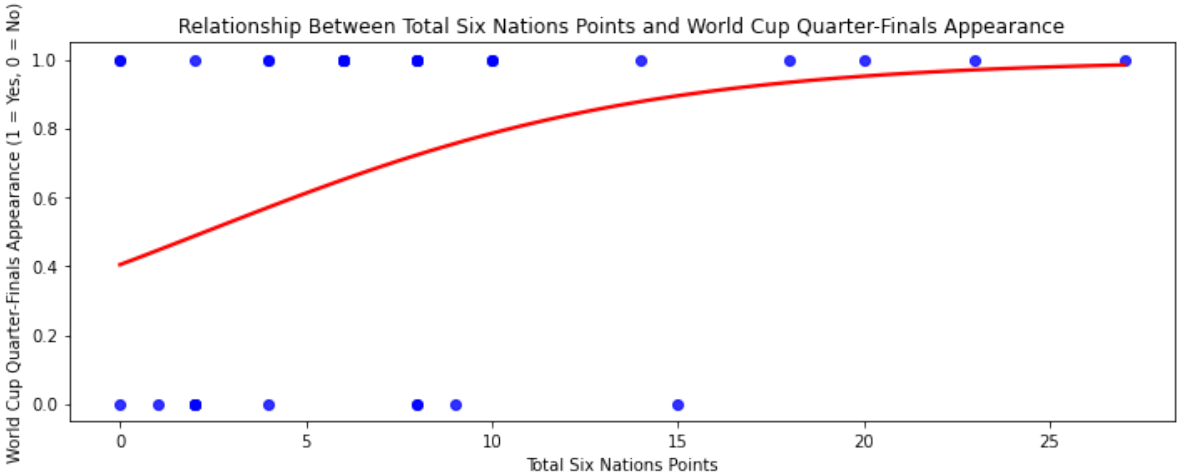


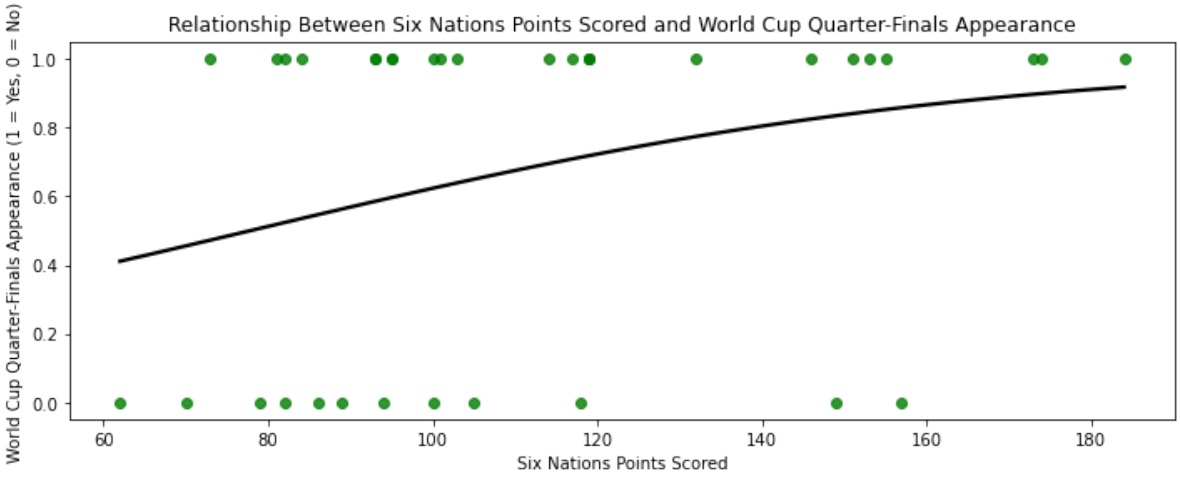
Correlation between Six Nations performance and reaching World Cup Quarterfinals

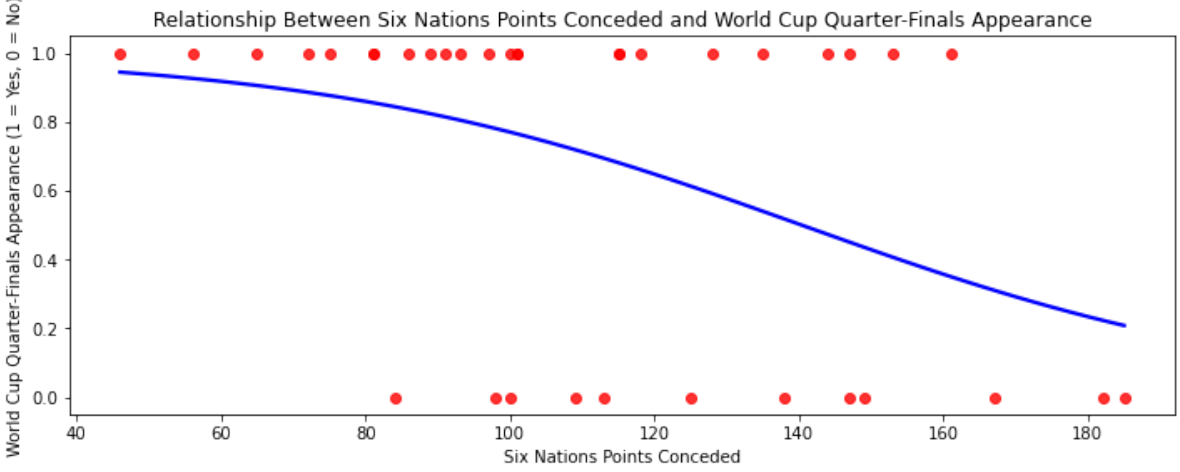
In [41]:
```
#Reshape and Clean Data
heatmap_data_long = heatmap_data_quarters.reset_index().melt(id_vars='Country', var_name
='Year', value_name='QuarterFinalist')
heatmap_data_long['Year'] = heatmap_data_long['Year'].astype(int)

#Rename columns for clarity and consistency
heatmap_data_long.rename(columns={'Country': 'Team'}, inplace=True)

#Clean six_nations_data
six_nations_data['Year'] = six_nations_data['Year'].astype(int)
six_nations_data['Points'] = pd.to_numeric(six_nations_data['Points'])
six_nations_data['Points Scored'] = pd.to_numeric(six_nations_data['Points Scored'])
six_nations_data['Points Conceded'] = pd.to_numeric(six_nations_data['Points Conceded'])

merged_data = pd.merge(six_nations_data, heatmap_data_long, on=['Year', 'Team'])

#Calculate the correlations
correlation_points = merged_data['Points'].corr(merged_data['QuarterFinalist'])
correlation_points_scored = merged_data['Points Scored'].corr(merged_data['QuarterFinali
st'])
correlation_points_conceded = merged_data['Points Conceded'].corr(merged_data['QuarterFi
nalist'])

print(f"The correlation between Six Nations total points and reaching the World Cup quar
ter-finals is: {correlation_points:.3f}")
print(f"The correlation between Six Nations points scored and reaching the World Cup qua
rter-finals is: {correlation_points_scored:.3f}")
print(f"The correlation between Six Nations points conceded and reaching the World Cup q
uarter-finals is: {correlation_points_conceded:.3f}")
```

```
The correlation between Six Nations total points and reaching the World Cup quarter-fin
als is: 0.334
The correlation between Six Nations points scored and reaching the World Cup quarter-fi
nals is: 0.291
The correlation between Six Nations points conceded and reaching the World Cup quarter-
finals is: -0.427
```

In [42]:
```
plt.figure(figsize=(10, 12))

#Subplot 1: Total Points
plt.subplot(3, 1, 1)
sns.regplot(x='Points', y='QuarterFinalist', data=merged_data, logistic=True, ci=None, s
catter_kws={'color': 'blue'}, line_kws={'color': 'red'})
plt.title('Relationship Between Total Six Nations Points and World Cup Quarter-Finals Ap
pearance')
plt.xlabel('Total Six Nations Points')
plt.ylabel('World Cup Quarter-Finals Appearance (1 = Yes, 0 = No)')

#Subplot 2: Points Scored
plt.subplot(3, 1, 2)
sns.regplot(x='Points Scored', y='QuarterFinalist', data=merged_data, logistic=True, ci=
None, scatter_kws={'color': 'green'}, line_kws={'color': 'black'})
plt.title('Relationship Between Six Nations Points Scored and World Cup Quarter-Finals A
ppearance')
plt.xlabel('Six Nations Points Scored')
plt.ylabel('World Cup Quarter-Finals Appearance (1 = Yes, 0 = No)')

#Subplot 3: Points Conceded
plt.subplot(3, 1, 3)
sns.regplot(x='Points Conceded', y='QuarterFinalist', data=merged_data, logistic=True, c
i=None, scatter_kws={'color': 'red'}, line_kws={'color': 'blue'})
plt.title('Relationship Between Six Nations Points Conceded and World Cup Quarter-Finals
Appearance')
plt.xlabel('Six Nations Points Conceded')
plt.ylabel('World Cup Quarter-Finals Appearance (1 = Yes, 0 = No)')

plt.tight_layout()
plt.show()
```


Relationship Between Total Six Nations Points and World Cup Quarter-Finals Appearance
World Cup Quarter-Finals Appearance (1 = Yes, 0 = No)
Total Six Nations Points


Relationship Between Six Nations Points Scored and World Cup Quarter-Finals Appearance
World Cup Quarter-Finals Appearance (1 = Yes, 0 = No)
Six Nations Points Scored

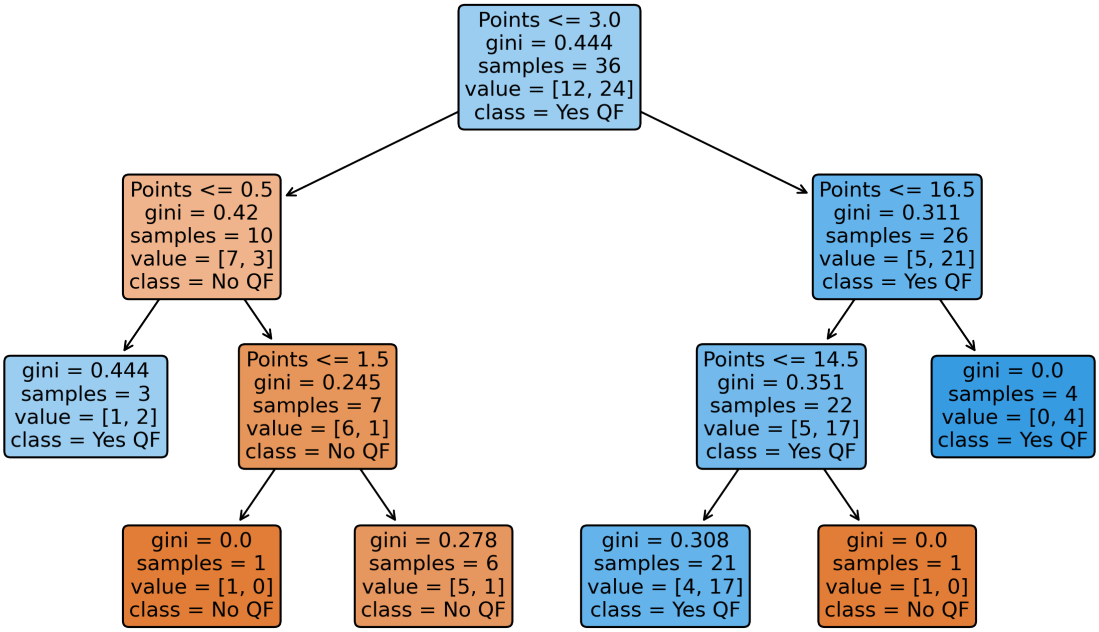

Relationship Between Six Nations Points Conceded and World Cup Quarter-Finals Appearance
World Cup Quarter-Finals Appearance (1 = Yes, 0 = No)
Six Nations Points Conceded

In [43]:
```
import statsmodels.api as sm

#Add a constant to the independent variable
merged_data['intercept'] = 1.0

#Total Points Model
logit_model_points = sm.Logit(merged_data['QuarterFinalist'], merged_data[['intercept',
'Points']])
result_points = logit_model_points.fit()
print("Model Summary for Total Points:")
print(result_points.summary())

#Points Scored Model
logit_model_points_scored = sm.Logit(merged_data['QuarterFinalist'], merged_data[['inter
cept', 'Points Scored']])
result_points_scored = logit_model_points_scored.fit()
print("\nModel Summary for Points Scored:")
print(result_points_scored.summary())

#Points Conceded Model
logit_model_points_conceded = sm.Logit(merged_data['QuarterFinalist'], merged_data[['int
ercept', 'Points Conceded']])
result_points_conceded = logit_model_points_conceded.fit()
print("\nModel Summary for Points Conceded:")
print(result_points_conceded.summary())
```

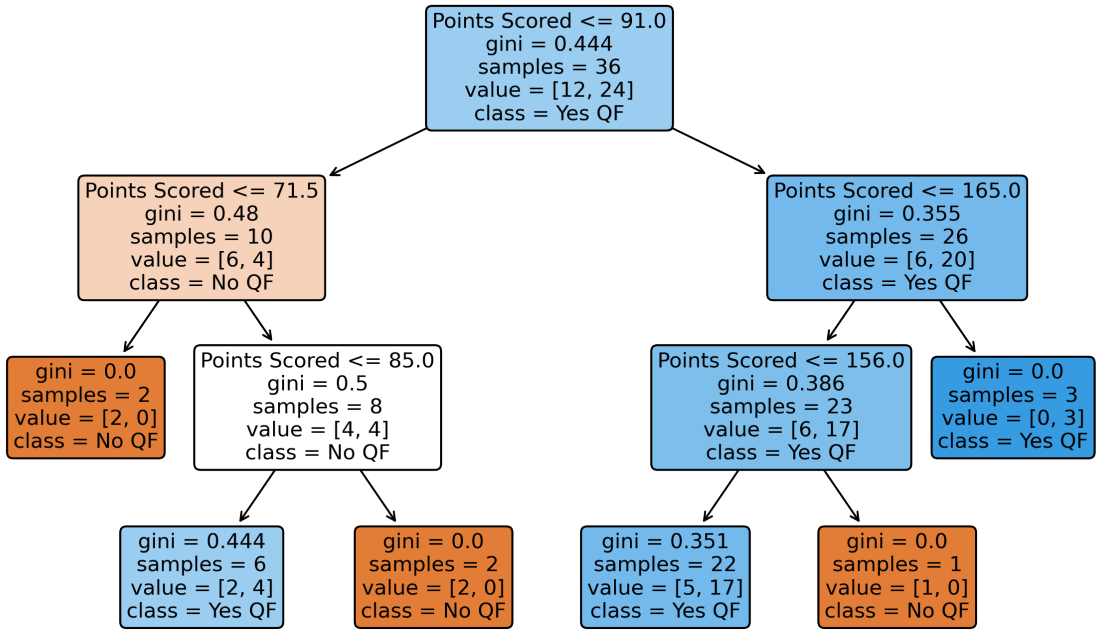
```
Optimization terminated successfully.
         Current function value: 0.566361
         Iterations 6
Model Summary for Total Points:
                           Logit Regression Results
==============================================================================
Dep. Variable:        QuarterFinalist   No. Observations:                   36
Model:                          Logit   Df Residuals:                       34
Method:                           MLE   Df Model:                            1
Date:                Wed, 08 May 2024   Pseudo R-squ.:                  0.1102
Time:                        19:07:22   Log-Likelihood:                -20.389
converged:                       True   LL-Null:                       -22.915
Covariance Type:            nonrobust   LLR p-value:                   0.02461
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
intercept     -0.3878      0.626     -0.619      0.536      -1.615       0.840
Points         0.1688      0.091      1.846      0.065      -0.010       0.348
==============================================================================
Optimization terminated successfully.
         Current function value: 0.590113
         Iterations 6

Model Summary for Points Scored:
                           Logit Regression Results
==============================================================================
Dep. Variable:        QuarterFinalist   No. Observations:                   36
Model:                          Logit   Df Residuals:                       34
Method:                           MLE   Df Model:                            1
Date:                Wed, 08 May 2024   Pseudo R-squ.:                 0.07290
Time:                        19:07:22   Log-Likelihood:                -21.244
converged:                       True   LL-Null:                       -22.915
Covariance Type:            nonrobust   LLR p-value:                   0.06758
==============================================================================
                    coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
intercept        -1.7650      1.465     -1.205      0.228      -4.636       1.106
Points Scored     0.0227      0.014      1.668      0.095      -0.004       0.049
==============================================================================
Optimization terminated successfully.
         Current function value: 0.540494
         Iterations 6

Model Summary for Points Conceded:
                           Logit Regression Results
==============================================================================
Dep. Variable:        QuarterFinalist   No. Observations:                   36
Model:                          Logit   Df Residuals:                       34
Method:                           MLE   Df Model:                            1
Date:                Wed, 08 May 2024   Pseudo R-squ.:                  0.1509
Time:                        19:07:22   Log-Likelihood:                -19.458
converged:                       True   LL-Null:                       -22.915
Covariance Type:            nonrobust   LLR p-value:                  0.008555
==============================================================================
                     coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
intercept           4.2012      1.579      2.660      0.008       1.106       7.297
Points Conceded    -0.0299      0.013     -2.360      0.018      -0.055      -0.005
==============================================================================
```

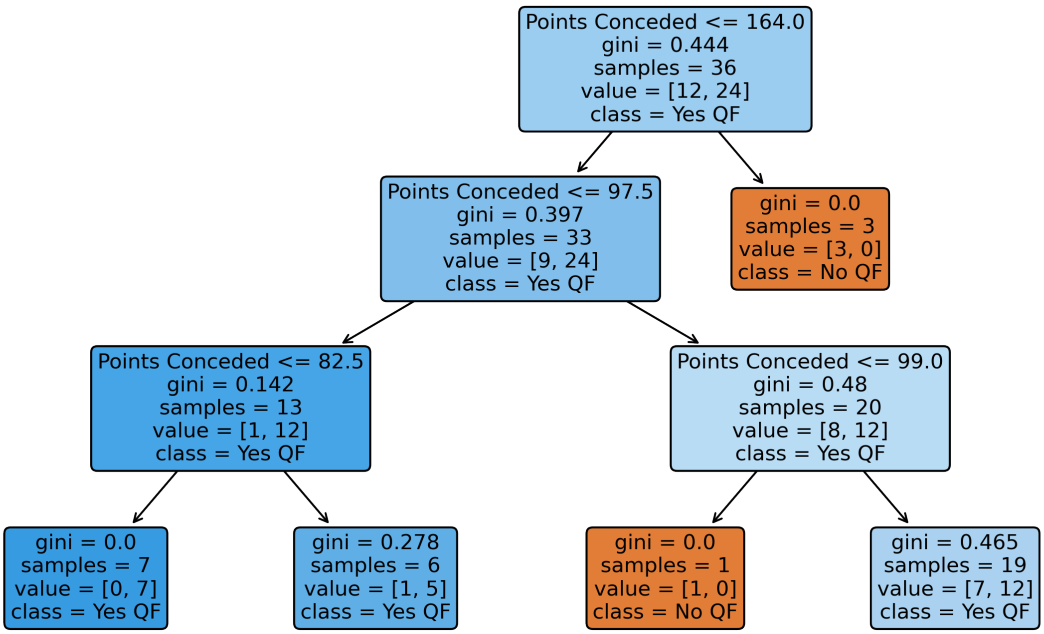
In [44]:
```
from sklearn.tree import DecisionTreeClassifier, plot_tree

features = ['Points', 'Points Scored', 'Points Conceded']
feature_names = [['Points'], ['Points Scored'], ['Points Conceded']]
class_names = ['No QF', 'Yes QF']

# Iterate through each feature and build a separate model
for feature, name in zip(features, feature_names):
    tree_model = DecisionTreeClassifier(max_depth=3)
    tree_model.fit(merged_data[name], merged_data['QuarterFinalist'])

    # Plot the decision tree
    plt.figure(figsize=(10, 6), dpi=300)  # Adjust size and resolution
    tree_plot = plot_tree(
        tree_model,
        filled=True,
        feature_names=name,
        class_names=class_names,
        rounded=True,  # Round the corners of the boxes
        fontsize=10,  # Adjust fontsize for better fit and readability
        label='all'  # Ensure all nodes are labeled
    )
    plt.title(f"Decision Tree for {name[0]}", pad=20)  # Add padding to the title
    plt.show()
```

Decision Tree for Points

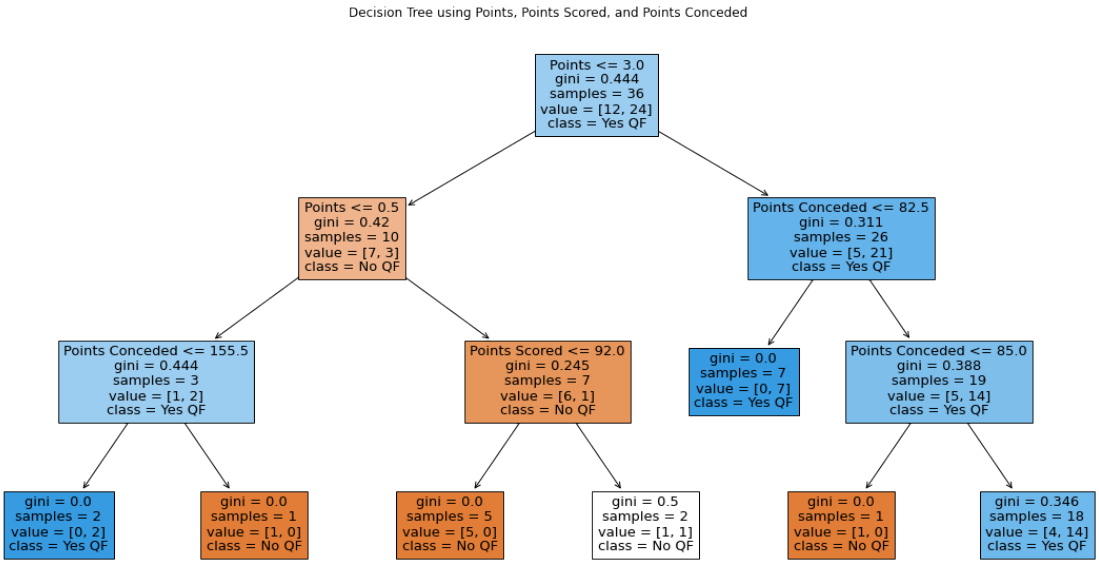


Decision Tree for Points Scored





In [45]:
```
#Create and fit the model using all features
combined_tree_model = DecisionTreeClassifier(max_depth=3)
combined_tree_model.fit(merged_data[features], merged_data['QuarterFinalist'])

#Plot the decision tree for the combined model
plt.figure(figsize=(20,10))
plot_tree(combined_tree_model, filled=True, feature_names=features, class_names=class_na
mes)
plt.title("Decision Tree using Points, Points Scored, and Points Conceded")
plt.show()
```



Average performances in Six Nations to reach World Cup Quarterfinals and Top 3

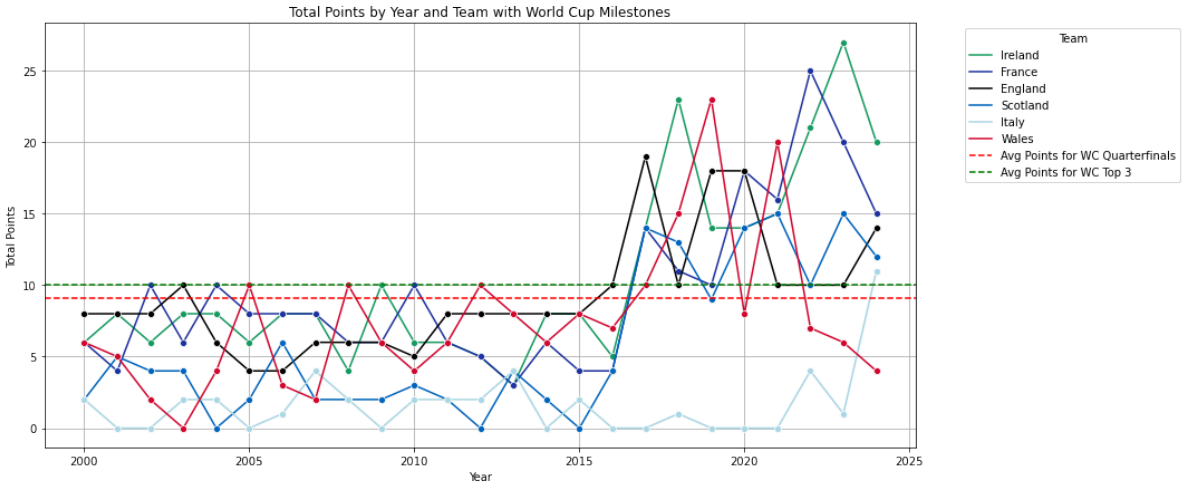
In [46]:
```
#Average Total Points
#Reshape world_cup_data to have a single 'Country' column for teams in top 3
top3_teams = World_Cup_Data.melt(id_vars='Year', value_vars=['Champion', 'Runner-up', 'T
hird'], var_name='Position', value_name='Team')

#Clean data
six_nations_data['Year'] = six_nations_data['Year'].astype(int)
six_nations_data['Team'] = six_nations_data['Team'].str.strip().astype(str)
top3_teams['Year'] = top3_teams['Year'].astype(int)
top3_teams['Team'] = top3_teams['Team'].str.strip().astype(str)
world_cup_quarterfinalists['Year'] = world_cup_quarterfinalists['Year'].astype(int)
world_cup_quarterfinalists['Country'] = world_cup_quarterfinalists['Country'].str.strip
().astype(str)


#Make merged dataframes
top3_merge = pd.merge(six_nations_data, top3_teams, on=['Year', 'Team'])
quarterfinals_merge = pd.merge(six_nations_data, world_cup_quarterfinalists, left_on=['Y
ear', 'Team'], right_on=['Year', 'Country'])

#Calculate averages
average_points_top3 = top3_merge['Points'].mean()
average_points_quarterfinals = quarterfinals_merge['Points'].mean()
print(f"Average Six Nations points for top 3 positions in the World Cup: {average_points
_top3}")
print(f"Average Six Nations points for World Cup quarterfinals qualification: {average_p
oints_quarterfinals}")
```

```
Average Six Nations points for top 3 positions in the World Cup: 10.0
Average Six Nations points for World Cup quarterfinals qualification: 9.083333333333334
```

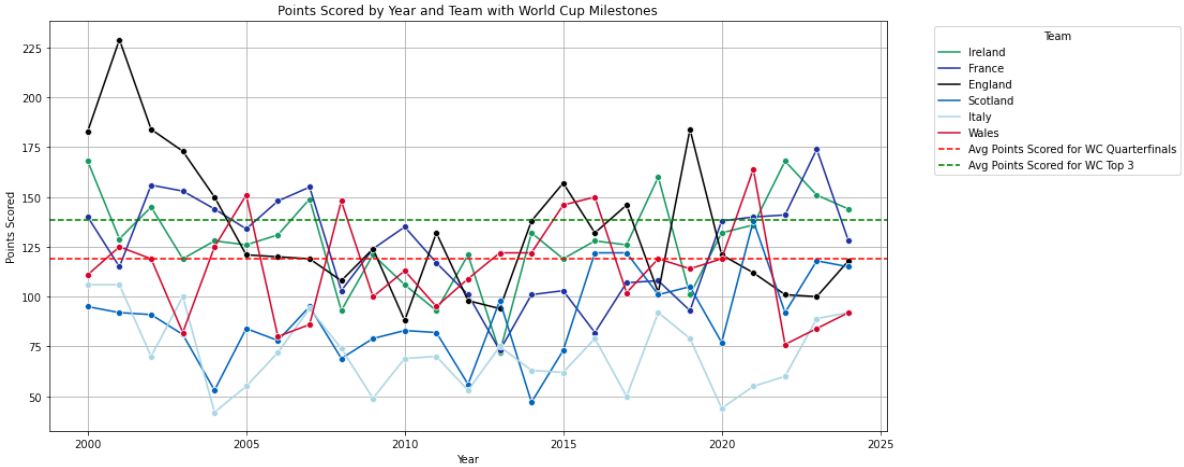
In [47]:
```
#Average Points Scored
#Calculate averages
average_points_scored_top3 = top3_merge['Points Scored'].mean()
average_points_scored_quarterfinals = quarterfinals_merge['Points Scored'].mean()

print(f"Average Six Nations points for top 3 positions in the World Cup: {average_points
_scored_top3}")
print(f"Average Six Nations points for World Cup quarterfinals qualification: {average_p
oints_scored_quarterfinals}")
```

```
Average Six Nations points for top 3 positions in the World Cup: 138.6
Average Six Nations points for World Cup quarterfinals qualification: 119.0
```

In [48]:
```
plt.figure(figsize=(14, 7))
sns.lineplot(x='Year', y='Points', data=six_nations_data, hue='Team', palette=team_colours, marker='o')
plt.axhline(y=average_points_quarterfinals, color='r', linestyle='--', label='Avg Points for WC Quarterfinals')
plt.axhline(y=average_points_top3, color='g', linestyle='--', label='Avg Points for WC Top 3')
plt.title('Total Points by Year and Team with World Cup Milestones')
plt.xlabel('Year')
plt.ylabel('Total Points')
plt.legend(title='Team', bbox_to_anchor=(1.05, 1), loc='upper left')
plt.grid(True)
plt.show()
```

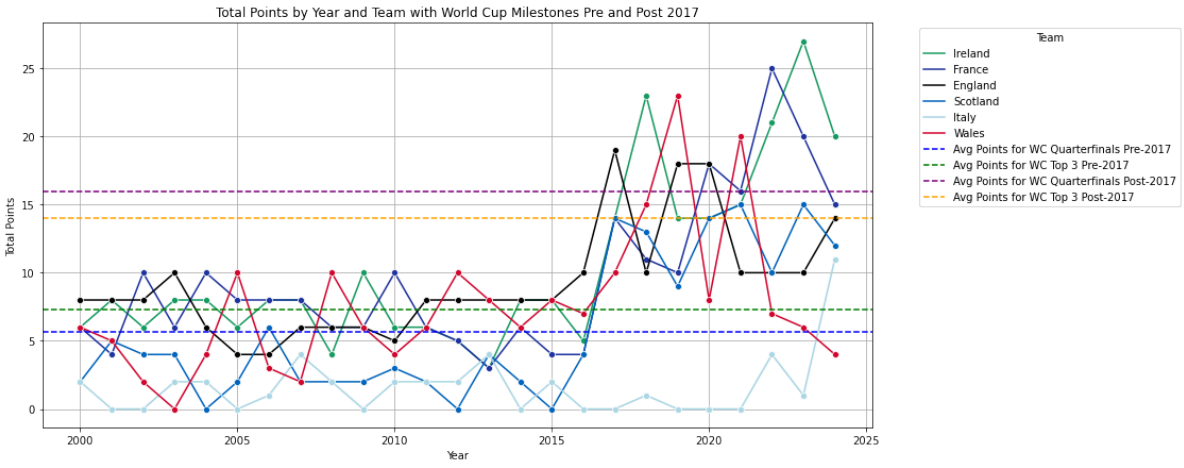


In [49]:
```
plt.figure(figsize=(14, 7))
sns.lineplot(x='Year', y='Points Scored', data=six_nations_data, hue='Team', palette=team_colours, marker='o')
plt.axhline(y=average_points_scored_quarterfinals, color='r', linestyle='--', label='Avg Points Scored for WC Quarterfinals')
plt.axhline(y=average_points_scored_top3, color='g', linestyle='--', label='Avg Points Scored for WC Top 3')
plt.title('Points Scored by Year and Team with World Cup Milestones')
plt.xlabel('Year')
plt.ylabel('Points Scored')
plt.legend(title='Team', bbox_to_anchor=(1.05, 1), loc='upper left')
plt.grid(True)
plt.show()
```



In [50]:
```
#Splitting the data due to Total Points scoring system change
pre_2017_data = six_nations_data[six_nations_data['Year'] < 2017]
post_2017_data = six_nations_data[six_nations_data['Year'] >= 2017]

#Merging with World Cup data
pre_2017_top3_merge = pd.merge(pre_2017_data, top3_teams, on=['Year', 'Team'])
pre_2017_quarterfinals_merge = pd.merge(pre_2017_data, world_cup_quarterfinalists, left_
on=['Year', 'Team'], right_on=['Year', 'Country'])

post_2017_top3_merge = pd.merge(post_2017_data, top3_teams, on=['Year', 'Team'])
post_2017_quarterfinals_merge = pd.merge(post_2017_data, world_cup_quarterfinalists, lef
t_on=['Year', 'Team'], right_on=['Year', 'Country'])

#Calculating averages
#Pre 2017
average_points_top3_pre_2017 = pre_2017_top3_merge['Points'].mean()
average_points_quarterfinals_pre_2017 = pre_2017_quarterfinals_merge['Points'].mean()

#Post 2017
average_points_top3_post_2017 = post_2017_top3_merge['Points'].mean()
average_points_quarterfinals_post_2017 = post_2017_quarterfinals_merge['Points'].mean()
```

In [51]:
```
plt.figure(figsize=(14, 7))
sns.lineplot(x='Year', y='Points', data=six_nations_data, hue='Team', palette=team_colou
rs, marker='o')
plt.axhline(y=average_points_quarterfinals_pre_2017, color='blue', linestyle='--', label
='Avg Points for WC Quarterfinals Pre-2017')
plt.axhline(y=average_points_top3_pre_2017, color='green', linestyle='--', label='Avg Po
ints for WC Top 3 Pre-2017')
plt.axhline(y=average_points_quarterfinals_post_2017, color='purple', linestyle='--', la
bel='Avg Points for WC Quarterfinals Post-2017')
plt.axhline(y=average_points_top3_post_2017, color='orange', linestyle='--', label='Avg
Points for WC Top 3 Post-2017')
plt.title('Total Points by Year and Team with World Cup Milestones Pre and Post 2017')
plt.xlabel('Year')
plt.ylabel('Total Points')
plt.legend(title='Team', bbox_to_anchor=(1.05, 1), loc='upper left')
plt.grid(True)
plt.show()
```



Correlation between Six Nations Winner and reaching World Cup Top 3

In [52]:
```
#Find the winners from each six nations
six_nations_winners = six_nations_data[six_nations_data['Rank'] == 1.0]
print(six_nations_winners)
```

```
     Year  Rank     Team  Points  Points Scored  Points Conceded  \
0    2024   1.0  Ireland      20            144               60
6    2023   1.0  Ireland      27            151               72
12   2022   1.0   France      25            141               73
18   2021   1.0    Wales      20            164              103
24   2020   1.0  England      18            121               77
30   2019   1.0    Wales      23            114               65
36   2018   1.0  Ireland      23            160               82
42   2017   1.0  England      19            146               81
48   2016   1.0  England      10            132               70
54   2015   1.0  Ireland       8            119               56
60   2014   1.0  Ireland       8            132               49
66   2013   1.0    Wales       8            122               66
72   2012   1.0    Wales      10            109               58
78   2011   1.0  England       8            132               81
84   2010   1.0   France      10            135               69
90   2009   1.0  Ireland      10            121               73
96   2008   1.0    Wales      10            148               66
102  2007   1.0   France       8            155               86
108  2006   1.0   France       8            148               85
114  2005   1.0    Wales      10            151               77
120  2004   1.0   France      10            144               60
126  2003   1.0  England      10            173               46
132  2002   1.0   France      10            156               75
138  2001   1.0  England       8            229               80
144  2000   1.0  England       8            183               70

     Points Difference
0                   84
6                   79
12                  68
18                  61
24                  44
30                  49
36                  78
42                  65
48                  62
54                  63
60                  83
66                  56
72                  51
78                  51
84                  66
90                  48
96                  82
102                 69
108                 63
114                 74
120                 84
126                127
132                 81
138                149
144                113
```

In [53]:
```
#Merge Six Nations winners with World Cup top 3 data
winners_top3_merge = pd.merge(six_nations_winners, top3_teams, left_on=['Year', 'Team'],
right_on=['Year', 'Team'])

#Create a binary column indicating if the Six Nations winner was in the top 3 of the Wor
ld Cup
winners_top3_merge['Top_3_Finish'] = 1

print(winners_top3_merge)
```

```
   Year  Rank     Team  Points  Points Scored  Points Conceded  \
0  2003   1.0  England      10            173               46

   Points Difference  Position  Top_3_Finish
0                127  Champion             1
```

In [54]:
```
#Correlation Calculations
#Ensure 'six_nations_winners' is independent
six_nations_winners = six_nations_data[six_nations_data['Rank'] == 1.0].copy()

#To remove the copy of a slice error
six_nations_winners.loc[:, 'Top_3_Finish'] = six_nations_winners.apply(
    lambda row: 1 if (row['Year'], row['Team']) in list(zip(winners_top3_merge['Year'],
winners_top3_merge['Team'])) else 0,
    axis=1
)

#Correlation calc
from scipy.stats import pearsonr
correlation, p_value = pearsonr(six_nations_winners['Points'], six_nations_winners['Top_
3_Finish'])
print("Correlation coefficient:", correlation)
print("P-value:", p_value)
```

```
Correlation coefficient: -0.10305766184139871
P-value: 0.6239809121574545
```

In [55]:
```
six_nations_winners['Non_Top_3_Finish'] = six_nations_winners['Top_3_Finish'].apply(lamb
da x: 1 if x == 0 else 0)

#Add an intercept term
six_nations_winners['intercept'] = 1

#Logistic regression model
logit_model = sm.Logit(six_nations_winners['Non_Top_3_Finish'], six_nations_winners[['in
tercept', 'Points']])

result = logit_model.fit()

print(result.summary())
```

```
Optimization terminated successfully.
         Current function value: 0.161135
         Iterations 9
                           Logit Regression Results
==============================================================================
Dep. Variable:       Non_Top_3_Finish   No. Observations:                   25
Model:                          Logit   Df Residuals:                       23
Method:                           MLE   Df Model:                            1
Date:                Wed, 08 May 2024   Pseudo R-squ.:                0.04054
Time:                        19:07:26   Log-Likelihood:               -4.0284
converged:                       True   LL-Null:                      -4.1986
Covariance Type:            nonrobust   LLR p-value:                   0.5596
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
intercept      1.6967      2.986      0.568      0.570      -4.155       7.548
Points         0.1305      0.279      0.468      0.640      -0.416       0.677
==============================================================================
```