

```

import requests
from urllib.parse import quote
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
import matplotlib.colors as mcolors
import matplotlib.gridspec as gridspec

api_key = "RGAPI-b211c0ba-cf9f-4cc3-84cd-185eba5196bc"

def get_puuid():
    print("Entrez le pseudo du joueur :")
    gametag = input()
    print(f"Entrez le # du joueur")
    tagline = input()
    gametag_encode = quote(gametag)
    api_url = f"https://europe.api.riotgames.com/riot/account/v1/accounts/by-riot-id/{gametag_encode}/{tagline}?api_key={api_key}"
    rep = requests.get(api_url)
    info_player = rep.json()
    puuid = info_player['puuid']
    return puuid

puuid = get_puuid()

def get_matches(puuid, nb_matches:int):
    api_url_matches = f"https://europe.api.riotgames.com/lol/match/v5/matches/by-puuid/{puuid}/ids?queue=420&start=0&count={nb_matches}&api_key={api_key}"
    rep = requests.get(api_url_matches)
    match_player = rep.json()
    return match_player

def get_data(match_id):
    url_get_data = f"https://europe.api.riotgames.com/lol/match/v5/matches/{match_id}?api_key={api_key}"
    rep = requests.get(url_get_data)
    if rep.status_code == 200:
        all_data = rep.json()
        return all_data['info']['participants']
    else :
        return print ('Problème dans la récupération de données')

a_garder = ['assists', 'champExperience', 'champLevel', 'championName', 'deaths', 'firstBloodKill', 'firstBloodAssist', 'firstTowerKill',
data_final = []
list_match_ids = get_matches(puuid=puuid, nb_matches=50)

for match_id in list_match_ids:
    data_match = get_data(match_id)
    df = pd.DataFrame(data_match)
    data_avocat_de_jaid = df[df['puuid'] == puuid].copy()
    data_avocat_de_jaid.loc[:, 'match_id'] = match_id
    data_final.append(data_avocat_de_jaid)

final_df = pd.concat(data_final)
final_df.set_index('match_id', inplace = True)

df_minus = final_df[a_garder]

#dernier_matches = get_matches(puuid, 5)
#dernier_match_id = dernier_matches[0]

#url_data_matches = f"https://europe.api.riotgames.com/lol/match/v5/matches/{dernier_match_id}?api_key={api_key}"
#rep = requests.get(url_data_matches)
#data_match_player = rep.json()

#participants = data_match_player['metadata']['participants']

#puuid_random = 'NtFeqA9M10sivyVOzDRjdyJpPBd0JJpA07xehBCnIsYEHynLtMXKJ3QGoSjMtq5gVtmbrrod4RSMgg'
#api_url_puuid = f"https://europe.api.riotgames.com/riot/account/v1/accounts/by-puuid/{puuid}?api_key={api_key}"
#rep = requests.get(api_url_puuid)
#info_compte = rep.json()
#pseudo = info_compte['gameName']

#participants_game = []

#for i in participants:
    #api_url_puuid = f"https://europe.api.riotgames.com/riot/account/v1/accounts/by-puuid/{i}?api_key={api_key}"
    #rep = requests.get(api_url_puuid)
    #info_compte = rep.json()
    #participants_game.append(info_compte['gameName'])

```

↗ Entrez le pseudo du joueur :
 avocat de jaïd
 Entrez le # du joueur
 mid

```
champion_counts = df_minus['championName'].value_counts()
```

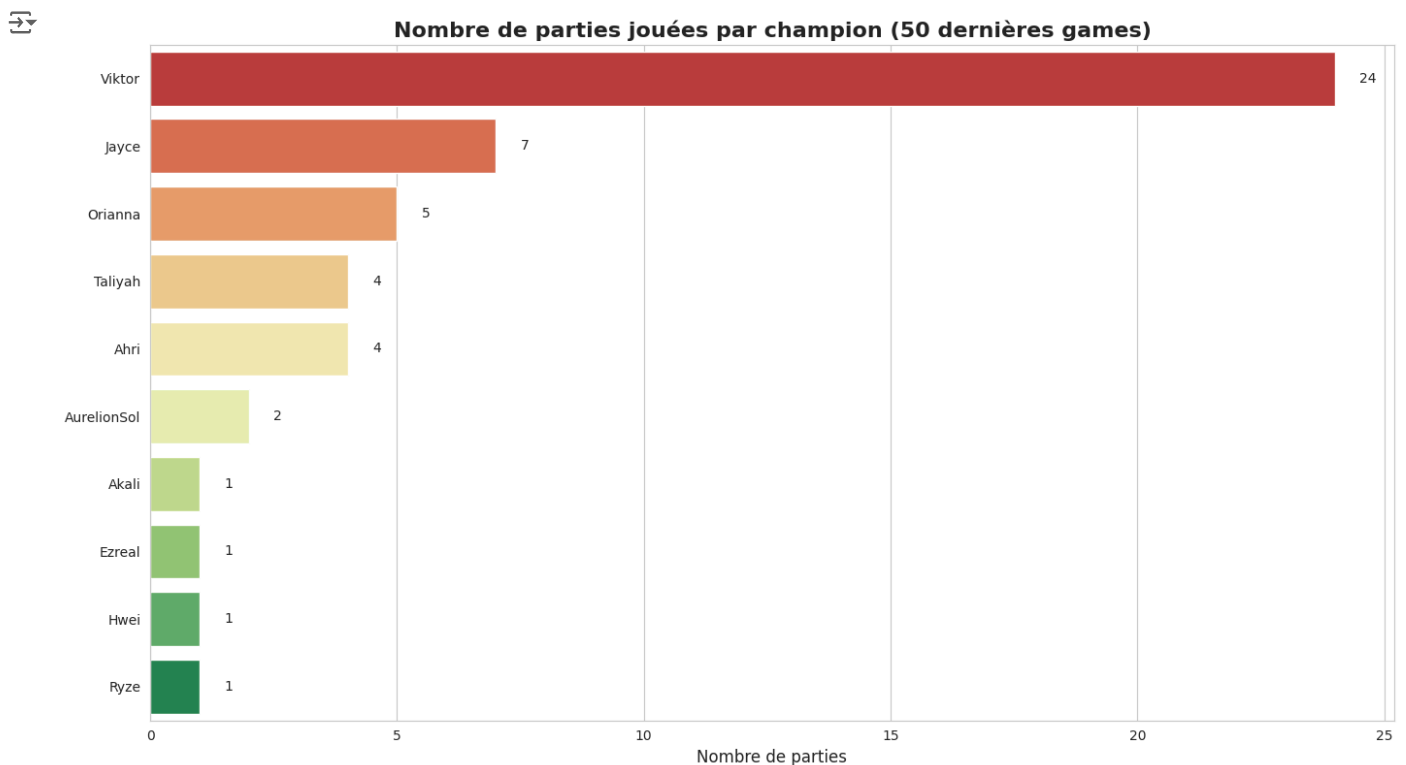
```
plt.figure(figsize=(14, 8))
sns.set_style("whitegrid")
```

```
palette = sns.color_palette("RdYlGn", len(champion_counts))
sns.barplot(y=champion_counts.index, x=champion_counts.values, palette=palette, hue = champion_counts.index, legend=False)
```

```
plt.title('Nombre de parties jouées par champion (50 dernières games)', fontsize=16, weight='bold')
plt.ylabel('', fontsize=12)
plt.xlabel('Nombre de parties', fontsize=12)
```

```
for i, value in enumerate(champion_counts.values):
    plt.text(value + 0.5, i, str(value), va='center', ha='left', fontsize=10)
```

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




```
df_minus['kda'] = (df_minus['kills'] + df_minus['assists']) / df_minus['deaths'].replace(0, np.nan)
df_minus['kda'] = df_minus['kda'].fillna(df_minus['kills'] + df_minus['assists'])
avg_kda_per_champion = df_minus.groupby('championName')['kda'].mean().sort_values(ascending=False)
```

```
overall_avg_kda = df_minus['kda'].mean()
```

```
num_champions = len(avg_kda_per_champion)
colors = plt.cm.Red(np.linspace(0.9, 0.3, num_champions))
```

```
plt.figure(figsize=(12, 6))
sns.barplot(x=avg_kda_per_champion.index, y=avg_kda_per_champion.values, palette=colors)
plt.axhline(overall_avg_kda, color='blue', linestyle='--', label='KDA moyen global')
plt.title('KDA moyen par champion vs KDA moyen global')
plt.xlabel('')
plt.ylabel('KDA moyen')
plt.xticks()
plt.legend()
plt.tight_layout()
plt.show()
```

 <ipython-input-13-08025c4b991e>:1: 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

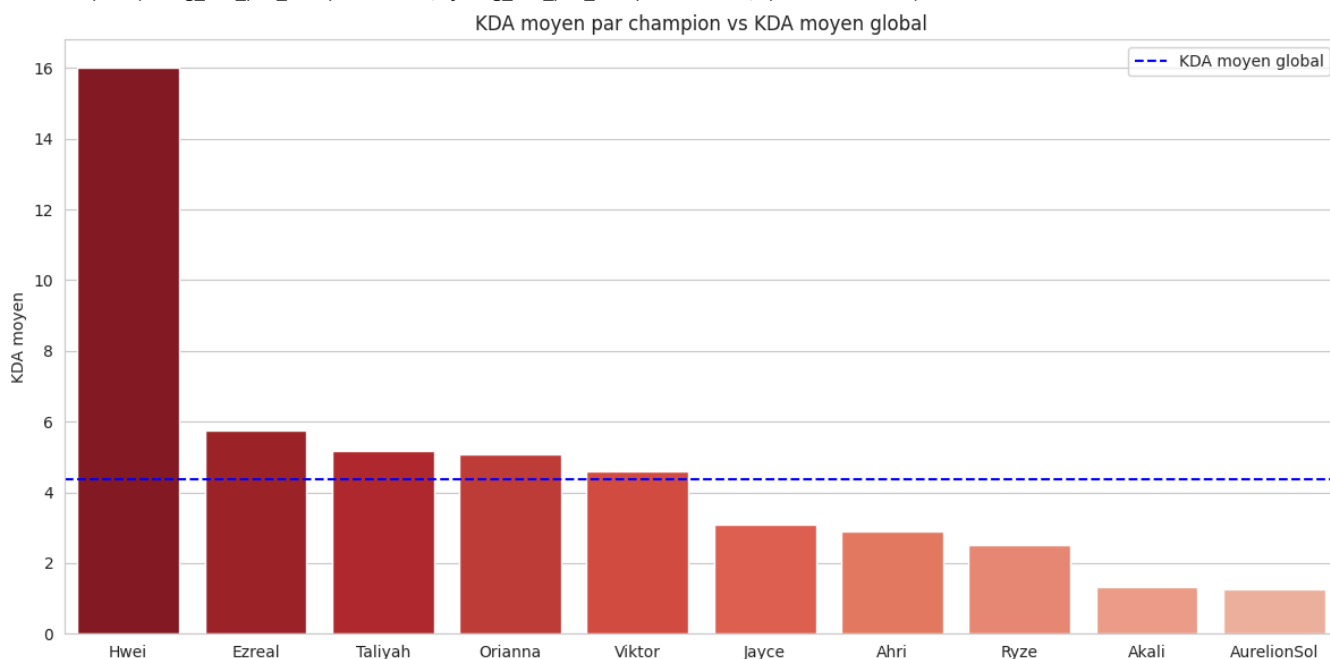
See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
df_minus['kda'] = (df_minus['kills'] + df_minus['assists']) / df_minus['deaths'].replace(0, np.nan)

<ipython-input-13-08025c4b991e>: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
df_minus['kda'] = df_minus['kda'].fillna(df_minus['kills'] + df_minus['assists'])
<ipython-input-13-08025c4b991e>:13: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `legend`

```
sns.barplot(x=avg_kda_per_champion.index, y=avg_kda_per_champion.values, palette=colors)
<ipython-input-13-08025c4b991e>:13: UserWarning: Numpy array is not a supported type for `palette`. Please convert your palette to a list
sns.barplot(x=avg_kda_per_champion.index, y=avg_kda_per_champion.values, palette=colors)
```



```
win_rate_per_champion = df_minus.groupby('championName')['win'].mean() * 100
games_per_champion = df_minus['championName'].value_counts()
```

```
win_rate_df = pd.DataFrame({
    'Win Rate': win_rate_per_champion,
    'Games Played': games_per_champion
})
```

```
win_rate_df = win_rate_df[win_rate_df['Win Rate'] > 0]
```

```
win_rate_df.sort_values(by='Win Rate', ascending=False, inplace=True)
```

```
plt.figure(figsize=(14, 8))
```

```

sns.set_style("whitegrid")

norm = mcolors.TwoSlopeNorm(vmin=win_rate_df['Win Rate'].min(), vcenter=56, vmax=win_rate_df['Win Rate'].max())
sm = plt.cm.ScalarMappable(cmap="RdYlGn", norm=norm)
colors = sm.to_rgba(win_rate_df['Win Rate'])

ax = sns.barplot(
    x=win_rate_df.index,
    y=win_rate_df['Win Rate'],
    palette=colors
)


plt.title('Win Rate par Champion', fontsize=16, weight='bold')
plt.xlabel('', fontsize=12)
plt.ylabel('Win Rate (%)', fontsize=12)
plt.xticks(fontsize=10)
plt.ylim(0, 100)

sm.set_array([])
plt.colorbar(sm, ax=ax, orientation='vertical', label='Win Rate (%)')

ax.axhline(y=50, color='red', linestyle='--', linewidth=2)

plt.tight_layout()
plt.show()

```

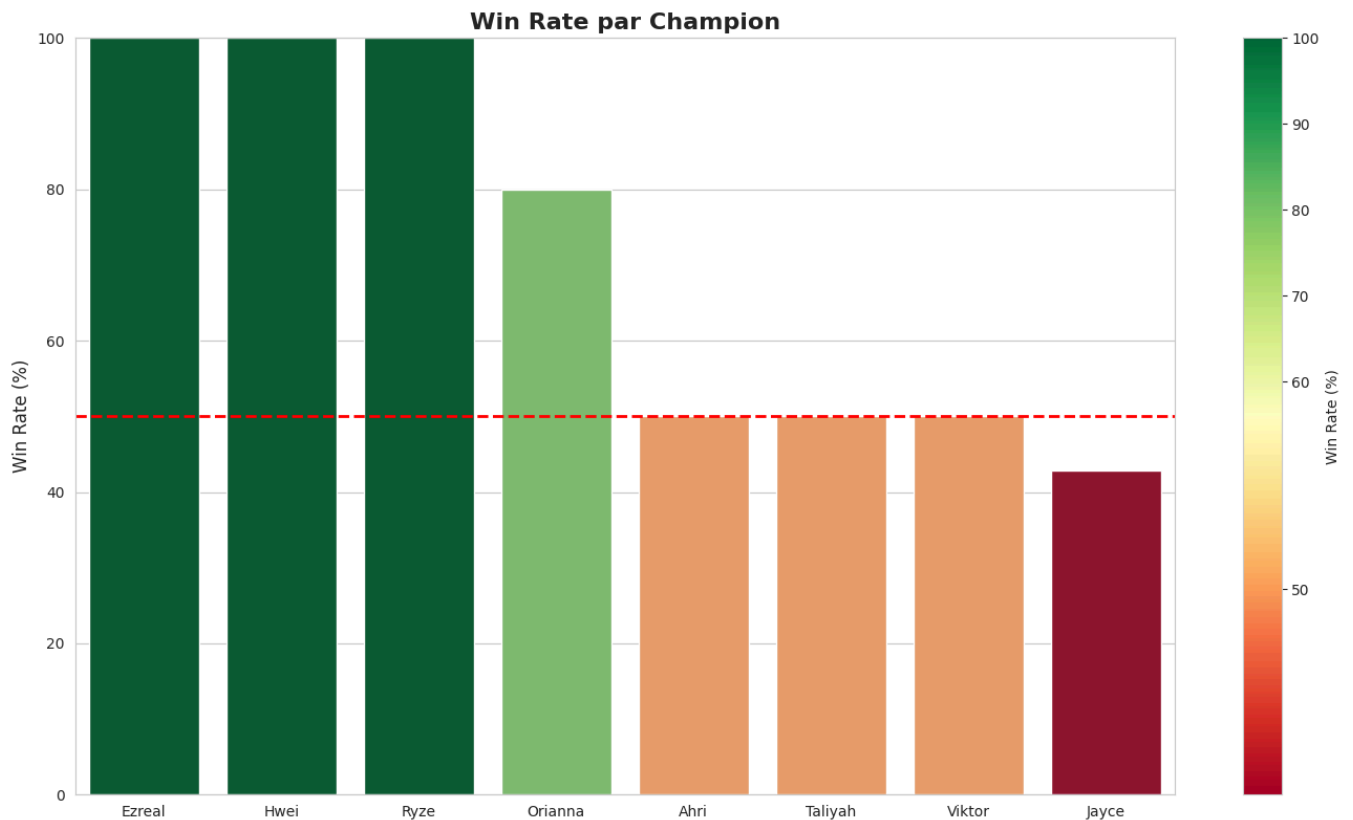
 <ipython-input-14-0712d5f3f8ee>:25: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `le`

```

ax = sns.barplot(
<ipython-input-14-0712d5f3f8ee>:25: UserWarning: Numpy array is not a supported type for `palette`. Please convert your palette to a
ax = sns.barplot(

```



```

fig = plt.figure(figsize=(18, 10))
gs = gridspec.GridSpec(2, 2)

```

```

ax1 = fig.add_subplot(gs[0, 0])
champion_counts = df_minus['championName'].value_counts()
sns.barplot(y=champion_counts.index, x=champion_counts.values, palette=sns.color_palette("RdYlGn", len(champion_counts)), hue=champion_counts.index, ax=ax1)
ax1.set_title('Nombre de parties jouées par champion (50 dernières parties)', fontsize=12, weight='bold')
ax1.set_ylabel('')
ax1.set_xlabel('Nombre de parties', fontsize=10)
for i, value in enumerate(champion_counts.values):
    ax1.text(value + 0.5, i, str(value), va='center', ha='left', fontsize=8)

ax2 = fig.add_subplot(gs[0, 1])
avg_kda_per_champion = df_minus.groupby('championName')['kda'].mean().sort_values(ascending=False)
overall_avg_kda = df_minus['kda'].mean()
num_champions = len(avg_kda_per_champion)
colors = plt.cm.Reds(np.linspace(0.9, 0.3, num_champions))
sns.barplot(x=avg_kda_per_champion.index, y=avg_kda_per_champion.values, palette=colors, ax=ax2)
ax2.axhline(overall_avg_kda, color='blue', linestyle='--', label='KDA moyen global')
ax2.set_title('KDA moyen par champion vs KDA moyen global', fontsize=12, weight='bold')
ax2.set_xlabel('')
ax2.set_ylabel('KDA moyen', fontsize=10)
ax2.tick_params(axis='x', labelrotation=45)
ax2.legend()

ax3 = fig.add_subplot(gs[1, :])
win_rate_per_champion = df_minus.groupby('championName')['win'].mean() * 100
games_per_champion = df_minus['championName'].value_counts()
win_rate_df = pd.DataFrame({'Win Rate': win_rate_per_champion, 'Games Played': games_per_champion})
win_rate_df = win_rate_df[win_rate_df['Win Rate'] > 0]
win_rate_df.sort_values(by='Win Rate', ascending=False, inplace=True)
norm = mcolors.TwoSlopeNorm(vmin=win_rate_df['Win Rate'].min(), vcenter=56, vmax=win_rate_df['Win Rate'].max())
sm = plt.cm.ScalarMappable(cmap="RdYlGn", norm=norm)
colors = sm.to_rgba(win_rate_df['Win Rate'])
sns.barplot(x=win_rate_df.index, y=win_rate_df['Win Rate'], palette=colors, ax=ax3)
ax3.set_title('Win Rate par Champion', fontsize=12, weight='bold')
ax3.set_xlabel('')
ax3.set_ylabel('Win Rate (%)', fontsize=10)
ax3.tick_params(axis='x')
ax3.set_ylim(0, 100)
sm.set_array([])
plt.colorbar(sm, ax=ax3, orientation='vertical', label='Win Rate (%)')
ax3.axhline(y=50, color='red', linestyle='--', linewidth=2)

plt.tight_layout()
plt.subplots_adjust(top=0.9)
fig.suptitle('Analyse des Performances du Joueur par Champion', fontsize=16, weight='bold')
plt.show()

```

 <ipython-input-19-fca9ec2d3051>:19: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `le

```
sns.barplot(x=avg_kda_per_champion.index, y=avg_kda_per_champion.values, palette=colors, ax=ax2)
<ipython-input-19-fca9ec2d3051>:19: UserWarning: Numpy array is not a supported type for `palette`. Please convert your palette to a
sns.barplot(x=avg_kda_per_champion.index, y=avg_kda_per_champion.values, palette=colors, ax=ax2)
<ipython-input-19-fca9ec2d3051>:37: FutureWarning:
```

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `le

```
sns.barplot(x=win_rate_df.index, y=win_rate_df['Win Rate'], palette=colors, ax=ax3)
<ipython-input-19-fca9ec2d3051>:37: UserWarning: Numpy array is not a supported type for `palette`. Please convert your palette to a
sns.barplot(x=win_rate_df.index, y=win_rate_df['Win Rate'], palette=colors, ax=ax3)
```

Analyse des Performances du Joueur par Champion

