Mind Map Coaching

▪ Educate ▪ Inform ▪ Inspire ▪

"EURO 24 Web Scraping"

A tutorial of scraping EURO 24 Stats from FBRef by a complete beginner

Lorcán Mason

Table of contents

1 Introduction to Web Scraping and Writing Data to CSV and Excel Files

Web scraping is a powerful technique used to extract data from websites, allowing you to gather information that might not be readily available in structured formats like databases or APIs. This data can then be cleaned and processed for analysis or storage in various formats, such as CSV (Comma-Separated Values) or Excel files. Using `R` and its powerful packages such as `rvest`, `tidyverse`, `stringr`, `readr`, and `openxlsx`, you can efficiently scrape, clean, and store data in formats suitable for analysis.

I've no doubt that this can be done more efficiently than I have done it but this is what I have learned so far. I hope this helps in some way and if you are a more experienced R user, I would love to hear your feedback on how I can improve this process - feel free to reach out to me on Twitter or LinkedIn. Below is a step-by-step guide on how to scrape all stats pages from EURO 24 via the FBRef website and write it to an Excel file and a CSV file. To learn from the experts, please check out TidyX by Patrick Ward and Ellis Hughes on Youtube. Mladen Jovanović also has fantastic courses on his website Complementary Training.

1.1 Load libraries for Web Scraping

```r
library(rvest)
library(tidyverse)
```

```
-- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
v dplyr     1.1.4     v readr     2.1.5
v forcats   1.0.0     v stringr   1.5.1
v ggplot2   3.5.1     v tibble    3.2.1
v lubridate 1.9.3     v tidyr     1.3.1
v purrr     1.0.2
-- Conflicts ------------------------------------------ tidyverse_conflicts() --
x dplyr::filter()         masks stats::filter()
x readr::guess_encoding() masks rvest::guess_encoding()
x dplyr::lag()            masks stats::lag()
i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```r
library(stringr)
library(readr)
library(openxlsx)
```

1.2 Define the Live HTML URLs for each Stats Web Page from FBRef

```r
general_stats_url <- "https://fbref.com/en/comps/676/stats/UEFA-Euro-Stats"
gk_url <- "https://fbref.com/en/comps/676/keepers/UEFA-Euro-Stats"
adv_gk_url <- "https://fbref.com/en/comps/676/keepersadv/UEFA-Euro-Stats"
shooting_url <- "https://fbref.com/en/comps/676/shooting/UEFA-Euro-Stats"
passing_stats <- "https://fbref.com/en/comps/676/passing/UEFA-Euro-Stats"
passing_types_stats <- "https://fbref.com/en/comps/676/passing_types/UEFA-Euro-Stats"
creation_stats <- "https://fbref.com/en/comps/676/gca/UEFA-Euro-Stats"
defensive_stats <- "https://fbref.com/en/comps/676/defense/UEFA-Euro-Stats"
possession_stats <- "https://fbref.com/en/comps/676/possession/UEFA-Euro-Stats"
playing_time_stats <- "https://fbref.com/en/comps/676/playingtime/UEFA-Euro-Stats"
misc_stats <- "https://fbref.com/en/comps/676/misc/UEFA-Euro-Stats"
```

2 Scrape General Stats Page

2.1 Read the live html tables from FBRef

```
standard_euro_tables <- read_html_live(general_stats_url) %>%
  html_table(fill = T)
```

2.2 Scrape the table of interest (3rd in this case) and convert it to a data frame - this has the individual player data

```
standard_players <- standard_euro_tables %>%
  .[[3]] %>%
  as.data.frame()
```

2.3 Convert the 1st row to column names and remove it from the data

```
colnames(standard_players) <- standard_players[1, ]
standard_players <- standard_players[-1, ]
```

2.4 Clean the column names and remove the rows with recurring column names

```
standard_players <- standard_players %>%
  janitor::clean_names() %>%
  filter(player != "Player")
```

2.5 Remove the "matches" column as it is not needed

```
standard_players <- standard_players %>%
  select(-matches)
```

2.6 Convert the numeric columns to numeric

```
standard_players <- standard_players %>%
  mutate(across(c(1, 5:ncol(.)), as.numeric))
```

2.6.1 Scrape the player ids from the player links in the live html table

```
player_id <- read_html_live(general_stats_url) %>%
  html_nodes("table") %>%
  html_nodes("tbody") %>%
  html_elements("a") %>%
  html_attr("href") %>%
  as.data.frame() %>%
  setNames("url_info") %>%
  # to this point the url also contains the player matchlogs so we need to filter out those
  mutate(get_players = ifelse(grepl(pattern = '/players/', url_info), 1, 0)) %>%
  mutate(get_matchlogs = ifelse(grepl(pattern = '/matchlogs/', url_info), 1, 0)) %>%
  filter(get_players == 1 & get_matchlogs == 0) %>%
  select(-get_players) %>%
  select(-get_matchlogs) %>%
  mutate(player_id = gsub("\\..*","", url_info),
         player_id = gsub(".*/[a-z]/","", player_id))
```

2.7 Join the datasets together

```
standard_players <- standard_players %>%
  bind_cols(player_id) %>%
  mutate(url_info = paste0("https://fbref.com/", url_info)) %>%
  rename(link_to_player_page = url_info)
```

2.8 As we don't need the 'rk' and 'age' columns, we can remove it to clean up the data into its final format

```
standard_players <- standard_players %>%
  select(-rk, -age)
```

2.9 Calculate an new 'age' column from the 'born' column to 2024

```
standard_players <- standard_players %>%
  mutate(age = 2024 - as.numeric(str_sub(born, start = -4)))
```

2.10 Remove the ‘born’ column as it is no longer needed and position the ‘age’ column next to the ‘player’ column

```
standard_players <- standard_players %>%
  select(-born) %>%
  select(player, age, everything())
```

2.11 Rename every column to make them more descriptive

```
standard_players <- standard_players %>%
  rename(
    position = pos,
    country = squad,
    matches_played = mp,
    minutes_played = min,
    played_90 = x90s,
    goals = gls,
    assists = ast,
    combined_goals_and_assists = g_a,
    non_penalty_goals = g_pk,
    penalties_scored = pk,
    penalties_taken = p_katt,
    yellow_cards = crd_y,
    red_cards = crd_r,
    expected_goals = x_g,
    non_penalty_expected_goals = npx_g,
    expected_assisted_goals = x_ag,
    non_penalty_expected_goals_and_assisted_goals = npx_g_x_ag,
    progressive_carries = prg_c,
    progressive_passes = prg_p,
    progressive_passes_received = prg_r,
    per90_goals = gls_2,
    per90_assists = ast_2,
    per90_goals_and_assists = g_a_2,
    per90_non_penalty_goals = g_pk_2,
    per90_non_penalty_goals_and_assists = g_a_pk,
    per90_expected_goals = x_g_2,
    per90_expected_assisted_goals = x_ag_2,
    per90_expected_goals_and_assisted_goals = x_g_x_ag,
    per90_non_penalty_expected_goals = npx_g_2,
    per90_non_penalty_expected_goals_and_assisted_goals = npx_g_x_ag_2,
    player_link = link_to_player_page) %>% # remove 'player_id' as it is no longer needed
  select(-player_id)
```

2.12 View the final data

```
head(standard_players)
```

```
            player age position        country matches_played starts
1        Che Adams  28       FW   sct Scotland              3      3
2 Michel Aebischer  27       DF ch Switzerland              5      5
3     Arlind Ajeti  31       DF      al Albania              3      3
4    Manuel Akanji  29       DF ch Switzerland              5      5
5    Samet Akaydın  30       DF      tr Türkiye              4      4
6       Nathan Aké  29       DF nl Netherlands              6      6
  minutes_played played_90 goals assists combined_goals_and_assists
1            209       2.3     0       0                          0
2            476       5.3     1       2                          3
3            270       3.0     0       0                          0
4            480       5.3     0       0                          0
5            335       3.7     1       0                          1
6            470       5.2     0       1                          1
  non_penalty_goals penalties_scored penalties_taken yellow_cards red_cards
1                 0                0               0            0         0
2                 1                0               0            0         0
3                 0                0               0            0         0
4                 0                0               0            0         0
5                 1                0               0            2         0
6                 0                0               0            1         0
  expected_goals non_penalty_expected_goals expected_assisted_goals
1            0.1                        0.1                     0.1
2            0.1                        0.1                     1.2
3            0.0                        0.0                     0.0
4            0.0                        0.0                     0.0
5            0.4                        0.4                     0.0
6            0.1                        0.1                     0.4
  non_penalty_expected_goals_and_assisted_goals progressive_carries
1                                           0.2                   3
2                                           1.3                   4
3                                           0.0                   1
```

```
4                                         0.0                   2
5                                         0.4                   1
6                                         0.6                   7
  progressive_passes progressive_passes_received per90_goals per90_assists
1                  2                          10        0.00          0.00
2                 28                          18        0.19          0.38
3                  5                           0        0.00          0.00
4                 25                           1        0.00          0.00
5                 15                           0        0.27          0.00
6                 21                          11        0.00          0.19
  per90_goals_and_assists per90_non_penalty_goals
1                    0.00                    0.00
2                    0.57                    0.19
3                    0.00                    0.00
4                    0.00                    0.00
5                    0.27                    0.27
6                    0.19                    0.00
  per90_non_penalty_goals_and_assists per90_expected_goals
1                                0.00                 0.05
2                                0.57                 0.02
3                                0.00                 0.00
4                                0.00                 0.00
5                                0.27                 0.09
6                                0.19                 0.02
  per90_expected_assisted_goals per90_expected_goals_and_assisted_goals
1                          0.03                                    0.09
2                          0.22                                    0.24
3                          0.00                                    0.00
4                          0.00                                    0.00
5                          0.00                                    0.09
6                          0.08                                    0.11
  per90_non_penalty_expected_goals
1                             0.05
2                             0.02
3                             0.00
4                             0.00
5                             0.09
6                             0.02
  per90_non_penalty_expected_goals_and_assisted_goals
1                                                0.09
2                                                0.24
3                                                0.00
```

```
4                                                0.00
5                                                0.09
6                                                0.11
                                              player_link
1        https://fbref.com//en/players/f2bf1b0f/Che-Adams
2 https://fbref.com//en/players/f9c927de/Michel-Aebischer
3     https://fbref.com//en/players/f9714604/Arlind-Ajeti
4    https://fbref.com//en/players/89ac64a6/Manuel-Akanji
5    https://fbref.com//en/players/7328bee5/Samet-Akaydin
6       https://fbref.com//en/players/eaeca114/Nathan-Ake
```

2.13 Write the data to a csv file

```
write_csv(standard_players, "standard_players.csv")
```

2.14 Write the data to an Excel file

```
write.xlsx(standard_players, "standard_players.xlsx")
```

3 Scrape General GK Stats Page

3.1 Read the live html tables from FBRef

```
gk_euro_tables <- read_html_live(gk_url) %>%
  html_table(fill = T)
```

3.2 Scrape the table of interest (3rd in this case) and convert it to a data frame - this has the individual player data

```
gk_players <- gk_euro_tables %>%
  .[[3]] %>%
  as.data.frame()
```

3.3 Convert the 1st row to column names and remove it from the data

```r
colnames(gk_players) <- gk_players[1, ]
gk_players <- gk_players[-1, ]
```

3.4 Clean the column names and remove the rows with recurring column names

```r
gk_players <- gk_players %>%
  janitor::clean_names() %>%
  filter(player != "Player")
```

3.5 Remove the "matches" column as it is not needed

```r
gk_players <- gk_players %>%
  select(-matches)
```

3.6 Convert the numeric columns to numeric

```r
gk_players <- gk_players %>%
  mutate(across(c(1, 5:ncol(.)), as.numeric))
```

3.6.1 Scrape the player ids from the player links in the live html table

```r
player_id <- read_html_live(gk_url) %>%
  html_nodes("table") %>%
  html_nodes("tbody") %>%
  html_elements("a") %>%
  html_attr("href") %>%
  as.data.frame() %>%
  setNames("url_info") %>%
  # to this point the url also contains the player matchlogs so we need to filter out those
  mutate(get_players = ifelse(grepl(pattern = '/players/', url_info), 1, 0)) %>%
  mutate(get_matchlogs = ifelse(grepl(pattern = '/matchlogs/', url_info), 1, 0)) %>%
  filter(get_players == 1 & get_matchlogs == 0) %>%
  select(-get_players) %>%
  select(-get_matchlogs) %>%
  mutate(player_id = gsub("\\..*","", url_info),
         player_id = gsub(".*/[a-z]/","", player_id))
```

3.7 Join the datasets together

```r
gk_players <- gk_players %>%
  bind_cols(player_id) %>%
  mutate(url_info = paste0("https://fbref.com/", url_info)) %>%
  rename(link_to_player_page = url_info)
```

3.8 As we don’t need the ‘rk’ and ‘age’ columns, we can remove it to clean up the data into its final format

```r
gk_players <- gk_players %>%
  select(-rk, -age)
```

3.9 Calculate an new ‘age’ column from the ‘born’ column to 2024

```r
gk_players <- gk_players %>%
  mutate(age = 2024 - as.numeric(str_sub(born, start = -4)))
```

3.10 Remove the ‘born’ column as it is no longer needed and position the ‘age’ column next to the ‘player’ column

```r
gk_players <- gk_players %>%
  select(-born) %>%
  select(player, age, everything())
```

3.11 Rename every column to make them more descriptive

```r
gk_players <- gk_players %>%
  rename(
    position = pos,
    country = squad,
    matches_played = mp,
    minutes_played = min,
    played_90 = x90s,
    goals_against = ga,
    goals_against_per_90 = ga90,
    shots_on_target_against = so_ta,
    wins = w,
    draw = d,
    losses = l,
    clean_sheets = cs,
    clean_sheets_percentage = cs_percent,
    penalty_kicks_attempted = p_katt,
    penalty_kicks_allowed = pka,
    penalty_kicks_saved = p_ksv,
    penalty_kicks_missed = p_km,
    penalty_kicks_saved_percentage = save_percent_2,
    player_link = link_to_player_page) %>% # remove 'player_id' as it is no longer needed
  select(-player_id)
```

3.12 View the data

```
glimpse(gk_players)
```

```
Rows: 29
Columns: 24
$ player                         <chr> "Altay Bayındır", "Koen Casteels", "Dio~
$ age                            <dbl> 26, 32, 25, 25, 35, 34, 28, 35, 24, 29,~
$ position                       <chr> "GK", "GK", "GK", "GK", "GK", "GK", "GK~
$ country                        <chr> "tr Türkiye", "be Belgium", "pt Portuga~
$ matches_played                 <dbl> 1, 4, 5, 4, 4, 3, 3, 4, 1, 3, 1, 6, 4, ~
$ starts                         <dbl> 1, 4, 5, 4, 4, 3, 3, 4, 0, 3, 1, 6, 4, ~
$ minutes_played                 <dbl> 90, 360, 510, 360, 390, 270, 270, 360, ~
$ played_90                      <dbl> 1.0, 4.0, 5.7, 4.0, 4.3, 3.0, 3.0, 4.0,~
$ goals_against                  <dbl> 3, 2, 3, 5, 5, 5, 7, 5, 1, 6, 3, 3, 8, ~
$ goals_against_per_90           <dbl> 3.00, 0.50, 0.53, 1.25, 1.15, 1.67, 2.3~
$ shots_on_target_against        <dbl> 3, 14, 14, 17, 16, 14, 19, 19, 2, 15, 5~
$ saves                          <dbl> 1, 13, 11, 12, 11, 9, 12, 15, 1, 9, 2, ~
$ save_percent                   <dbl> 0.0, 85.7, 85.7, 70.6, 75.0, 64.3, 68.4~
$ wins                           <dbl> 0, 1, 2, 1, 1, 1, 0, 3, 0, 0, 0, 2, 1, ~
$ draw                           <dbl> 0, 1, 2, 1, 1, 0, 1, 0, 0, 2, 0, 3, 1, ~
$ losses                         <dbl> 1, 2, 1, 2, 2, 2, 2, 1, 0, 1, 1, 1, 2, ~
$ clean_sheets                   <dbl> 0, 2, 3, 0, 1, 1, 0, 0, 0, 0, 0, 4, 1, ~
$ clean_sheets_percentage        <dbl> 0.0, 50.0, 60.0, 0.0, 25.0, 33.3, 0.0, ~
$ penalty_kicks_attempted        <dbl> 0, 0, 1, 1, 1, 0, 1, 0, 0, 0, 0, 1, 0, ~
$ penalty_kicks_allowed          <dbl> 0, 0, 1, 0, 1, 0, 1, 0, 0, 0, 0, 1, 0, ~
$ penalty_kicks_saved            <dbl> 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, ~
$ penalty_kicks_missed           <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ~
$ penalty_kicks_saved_percentage <dbl> NA, NA, 0, 100, 0, NA, 0, NA, NA, NA, N~
$ player_link                    <chr> "https://fbref.com//en/players/072e68ed~
```

3.13 Write the data to a csv file

```r
write_csv(gk_players, "gk_players.csv")
```

3.14 Write the data to an Excel file

```r
write.xlsx(gk_players, "gk_players.xlsx")
```

4 Scrape Advanced GK Stats Page

4.1 Read the live html tables from FBRef

```r
adv_gk_euro_tables <- read_html_live(adv_gk_url) %>%
  html_table(fill = T)
```

4.2 Scrape the table of interest (3rd in this case) and convert it to a data frame - this has the individual player data

```r
adv_gk_players <- adv_gk_euro_tables %>%
  .[[3]] %>%
  as.data.frame()
```

4.3 Convert the 1st row to column names and remove it from the data

```r
colnames(adv_gk_players) <- adv_gk_players[1, ]
adv_gk_players <- adv_gk_players[-1, ]
```

4.4 Clean the column names and remove the rows with recurring column names

```r
adv_gk_players <- adv_gk_players %>%
  janitor::clean_names() %>%
  filter(player != "Player")
```

4.5 Remove the "matches" column as it is not needed

```r
adv_gk_players <- adv_gk_players %>%
  select(-matches)
```

4.6 Convert the numeric columns to numeric

```r
adv_gk_players <- adv_gk_players %>%
  mutate(across(c(1, 5:ncol(.)), as.numeric))
```

4.6.1 Scrape the player ids from the player links in the live html table

```r
player_id <- read_html_live(adv_gk_url) %>%
  html_nodes("table") %>%
  html_nodes("tbody") %>%
  html_elements("a") %>%
  html_attr("href") %>%
  as.data.frame() %>%
  setNames("url_info") %>%
  # to this point the url also contains the player matchlogs so we need to filter out those
  mutate(get_players = ifelse(grepl(pattern = '/players/', url_info), 1, 0)) %>%
  mutate(get_matchlogs = ifelse(grepl(pattern = '/matchlogs/', url_info), 1, 0)) %>%
  filter(get_players == 1 & get_matchlogs == 0) %>%
  select(-get_players) %>%
  select(-get_matchlogs) %>%
  mutate(player_id = gsub("\\..*","", url_info),
         player_id = gsub(".*/[a-z]/","", player_id))
```

4.7 Join the datasets together

```r
adv_gk_players <- adv_gk_players %>%
  bind_cols(player_id) %>%
  mutate(url_info = paste0("https://fbref.com/", url_info)) %>%
  rename(link_to_player_page = url_info)
```

4.8 As we don’t need the ‘rk’ and ‘age’ columns, we can remove it to clean up the data into its final format

```r
adv_gk_players <- adv_gk_players %>%
  select(-rk, -age)
```

4.9 Calculate an new ‘age’ column from the ‘born’ column to 2024

```r
adv_gk_players <- adv_gk_players %>%
  mutate(age = 2024 - as.numeric(str_sub(born, start = -4)))
```

4.10 Remove the ‘born’ column as it is no longer needed and position the ‘age’ column next to the ‘player’ column

```r
adv_gk_players <- adv_gk_players %>%
  select(-born) %>%
  select(player, age, everything())
```

4.11 Rename every column to make them more descriptive

```r
adv_gk_players <- adv_gk_players %>%
  rename(
    position = pos,
    country = squad,
    played_90 = x90s,
    goals_against = ga,
    penalty_kicks_allowed = pka,
    free_kick_goals_against = fk,
    corner_kick_goals_against = ck,
    own_goals_against = og,
    post_shot_xg = p_sx_g,
    post_shot_xg_per_shot_on_target = p_sx_g_so_t,
    post_shot_xg_excluding_conceded = p_sx_g_2,
    post_shot_xg_excluding_conceded_per90 = x90,
    passes_completed_over_40yrds = cmp,
    passes_attempted_over_40yrds = att,
    passes_completed_over_40yrds_percent = cmp_percent,
    passes_attempted = att_gk,
    throws_attempted = thr,
    passes_attempted_over_40yrds_percent = launch_percent,
    average_pass_length = avg_len,
    goal_kicks_attempted = att_2,
    goal_kicks_over_40yrds_percent= launch_percent_2,
    average_goal_kick_length = avg_len_2,
    crosses_faced = opp,
    crosses_stopped = stp,
    crosses_stopped_percent = stp_percent,
    number_of_defensive_actions_outside_penalty_area = number_opa,
    number_of_defensive_actions_outside_penalty_area_per90 = number_opa_90,
    distance_from_goal_of_all_defensive_actions_yrds = avg_dist,

    player_link = link_to_player_page) %>% # remove 'player_id' as it is no longer needed
  select(-player_id)
```

4.12 View the data

```
head(adv_gk_players)
```

```
                player age position     country played_90 goals_against
1        Altay Bayındır  26       GK  tr Türkiye       1.0             3
2         Koen Casteels  32       GK  be Belgium       4.0             2
3           Diogo Costa  25       GK pt Portugal       5.7             3
4 Gianluigi Donnarumma  25       GK    it Italy       4.0             5
5      Martin Dúbravka  35       GK sk Slovakia       4.3             5
6         Péter Gulácsi  34       GK  hu Hungary       3.0             5
  penalty_kicks_allowed free_kick_goals_against corner_kick_goals_against
1                     0                       0                         0
2                     0                       0                         0
3                     1                       0                         0
4                     0                       0                         0
5                     1                       0                         0
```

```
6                       0                        0                          0
  own_goals_against post_shot_xg post_shot_xg_per_shot_on_target
1                 1          1.6                            0.54
2                 1          2.7                            0.20
3                 0          3.5                            0.18
4                 1          6.0                            0.30
5                 0          5.0                            0.25
6                 0          5.4                            0.39
  post_shot_xg_excluding_conceded post_shot_xg_excluding_conceded_per90
1                            -0.4                                 -0.37
2                             1.7                                  0.43
3                             0.5                                  0.11
4                             2.0                                  0.49
5                             0.0                                  0.01
6                             0.4                                  0.14
  passes_completed_over_40yrds passes_attempted_over_40yrds
1                            3                            7
2                            8                           31
3                            6                           18
4                           10                           23
5                           18                           61
6                           10                           30
  passes_completed_over_40yrds_percent passes_attempted throws_attempted
1                                 42.9               33                3
2                                 25.8              107               19
3                                 33.3              129               29
4                                 43.5               86               16
5                                 29.5              138               20
6                                 33.3               83               17
  passes_attempted_over_40yrds_percent average_pass_length goal_kicks_attempted
1                                 21.2                31.2                    2
2                                 17.8                27.1                   35
3                                 13.2                27.2                   26
4                                 22.1                26.3                   16
5                                 31.9                30.1                   23
6                                 28.9                31.8                   14
  goal_kicks_over_40yrds_percent average_goal_kick_length crosses_faced
1                            0.0                      7.5             8
2                           34.3                     34.7            48
3                            3.8                     25.6            41
4                           25.0                     27.1            41
5                           73.9                     54.7            57
```

```
6                           42.9                       39.0            43
  crosses_stopped crosses_stopped_percent
1               2                    25.0
2               4                     8.3
3               2                     4.9
4               3                     7.3
5               2                     3.5
6               1                     2.3
  number_of_defensive_actions_outside_penalty_area
1                                                0
2                                                4
3                                                6
4                                                2
5                                                4
6                                                1
  number_of_defensive_actions_outside_penalty_area_per90
1                                                   0.00
2                                                   1.00
3                                                   1.20
4                                                   0.50
5                                                   1.00
6                                                   0.33
  distance_from_goal_of_all_defensive_actions_yrds
1                                             12.2
2                                             13.8
3                                             21.0
4                                             10.1
5                                             11.8
6                                             10.1
                                                 player_link
1      https://fbref.com//en/players/072e68ed/Altay-Bayindir
2       https://fbref.com//en/players/db401046/Koen-Casteels
3         https://fbref.com//en/players/93fffbcf/Diogo-Costa
4 https://fbref.com//en/players/08f5afaa/Gianluigi-Donnarumma
5     https://fbref.com//en/players/3a949a25/Martin-Dubravka
6       https://fbref.com//en/players/bcb2ccd8/Peter-Gulacsi
```

4.13 Write the data to a csv file

```
write_csv(adv_gk_players, "adv_gk_players.csv")
```

4.14 Write the data to an Excel file

```r
write.xlsx(adv_gk_players, "adv_gk_players.xlsx")
```

5 Scrape Shooting Stats Page

5.1 Read the live html tables from FBRef

```r
shooting_euro_tables <- read_html_live(shooting_url) %>%
  html_table(fill = T)
```

5.2 Scrape the table of interest (3rd in this case) and convert it to a data frame - this has the individual player data

```r
shooting_players <- shooting_euro_tables %>%
  .[[3]] %>%
  as.data.frame()
```

5.3 Convert the 1st row to column names and remove it from the data

```r
colnames(shooting_players) <- shooting_players[1, ]
shooting_players <- shooting_players[-1, ]
```

5.4 Clean the column names and remove the rows with recurring column names

```r
shooting_players <- shooting_players %>%
  janitor::clean_names() %>%
  filter(player != "Player")
```

5.5 Remove the "matches" column as it is not needed

```r
shooting_players <- shooting_players %>%
  select(-matches)
```

5.6 Convert the numeric columns to numeric

```r
shooting_players <- shooting_players %>%
  mutate(across(c(1, 5:ncol(.)), as.numeric))
```

5.6.1 Scrape the player ids from the player links in the live html table

```r
player_id <- read_html_live(shooting_url) %>%
  html_nodes("table") %>%
  html_nodes("tbody") %>%
  html_elements("a") %>%
  html_attr("href") %>%
  as.data.frame() %>%
  setNames("url_info") %>%
  # to this point the url also contains the player matchlogs so we need to filter out those
  mutate(get_players = ifelse(grepl(pattern = '/players/', url_info), 1, 0)) %>%
  mutate(get_matchlogs = ifelse(grepl(pattern = '/matchlogs/', url_info), 1, 0)) %>%
  filter(get_players == 1 & get_matchlogs == 0) %>%
  select(-get_players) %>%
  select(-get_matchlogs) %>%
  mutate(player_id = gsub("\\..*","", url_info),
         player_id = gsub(".*/[a-z]/","", player_id))
```

5.7 Join the datasets together

```r
shooting_players <- shooting_players %>%
  bind_cols(player_id) %>%
  mutate(url_info = paste0("https://fbref.com/", url_info)) %>%
  rename(link_to_player_page = url_info)
```

5.8 As we don’t need the ‘rk’ and ‘age’ columns, we can remove it to clean up the data into its final format

```r
shooting_players <- shooting_players %>%
  select(-rk, -age)
```

5.9 Calculate an new ‘age’ column from the ‘born’ column to 2024

```r
shooting_players <- shooting_players %>%
  mutate(age = 2024 - as.numeric(str_sub(born, start = -4)))
```

5.10 Remove the ‘born’ column as it is no longer needed and position the ‘age’ column next to the ‘player’ column

```r
shooting_players <- shooting_players %>%
  select(-born) %>%
  select(player, age, everything())
```

5.11 Rename every column to make them more descriptive

```r
shooting_players <- shooting_players %>%
  rename(
    position = pos,
    country = squad,
    played_90 = x90s,
    goals = gls,
    shots = sh,
    shots_on_target = so_t,
    shots_on_target_percent = so_t_percent,
    shots_per90 = sh_90,
    shots_on_target_per90 = so_t_90,
    goals_per_shot = g_sh,
    goals_per_shot_on_target = g_so_t,
    average_shot_distance = dist,
    free_kick_shots = fk,
    penalty_scored = pk,
    penalty_kicks_attempted = p_katt,
    expected_goals = x_g,
    non_penalty_expected_goals = npx_g,
    non_penalty_expected_goals_shots = npx_g_sh,
    goals_minus_expected_goals = g_x_g,
    non_penalty_expected_goals_minus_expected_goals = np_g_x_g,
    player_link = link_to_player_page) %>% # remove 'player_id' as it is no longer needed
  select(-player_id)
```

5.12 Write the data to a csv file

```
write_csv(shooting_players, "shooting_players.csv")
```

5.13 Write the data to an Excel file

```
write.xlsx(shooting_players, "shooting_players.xlsx")
```

6 Scrape Passing Stats Page

6.1 Read the live html tables from FBRef

```
passing_euro_tables <- read_html_live(passing_stats) %>%
  html_table(fill = T)
```

6.2 Scrape the table of interest (3rd in this case) and convert it to a data frame - this has the individual player data

```
passing_players <- passing_euro_tables %>%
  .[[3]] %>%
  as.data.frame()
```

6.3 Convert the 1st row to column names and remove it from the data

```
colnames(passing_players) <- passing_players[1, ]
passing_players <- passing_players[-1, ]
```

6.4 Clean the column names and remove the rows with recurring column names

```
passing_players <- passing_players %>%
  janitor::clean_names() %>%
  filter(player != "Player")
```

6.5 Remove the “matches” column as it is not needed

```r
passing_players <- passing_players %>%
  select(-matches)
```

6.6 Convert the numeric columns to numeric

```r
passing_players <- passing_players %>%
  mutate(across(c(1, 5:ncol(.)), as.numeric))
```

6.6.1 Scrape the player ids from the player links in the live html table

```r
player_id <- read_html_live(passing_stats) %>%
  html_nodes("table") %>%
  html_nodes("tbody") %>%
  html_elements("a") %>%
  html_attr("href") %>%
  as.data.frame() %>%
  setNames("url_info") %>%
  # to this point the url also contains the player matchlogs so we need to filter out those
  mutate(get_players = ifelse(grepl(pattern = '/players/', url_info), 1, 0)) %>%
  mutate(get_matchlogs = ifelse(grepl(pattern = '/matchlogs/', url_info), 1, 0)) %>%
  filter(get_players == 1 & get_matchlogs == 0) %>%
  select(-get_players) %>%
  select(-get_matchlogs) %>%
  mutate(player_id = gsub("\\..*","", url_info),
         player_id = gsub(".*/[a-z]/","", player_id))
```

6.7 Join the datasets together

```r
passing_players <- passing_players %>%
  bind_cols(player_id) %>%
  mutate(url_info = paste0("https://fbref.com/", url_info)) %>%
  rename(link_to_player_page = url_info)
```

6.8 As we don't need the 'rk' and 'age' columns, we can remove it to clean up the data into its final format

```r
passing_players <- passing_players %>%
  select(-rk, -age)
```

6.9 Calculate an new 'age' column from the 'born' column to 2024

```r
passing_players <- passing_players %>%
  mutate(age = 2024 - as.numeric(str_sub(born, start = -4)))
```

6.10 Remove the 'born' column as it is no longer needed and position the 'age' column next to the 'player' column

```r
passing_players <- passing_players %>%
  select(-born) %>%
  select(player, age, everything())
```

6.11 Rename every column to make them more descriptive

```r
passing_players <- passing_players %>%
  rename(
    position = pos,
    country = squad,
    played_90 = x90s,
    passes_completed = cmp,
    passes_attempted = att,
    pass_completion_percent = cmp_percent,
    passing_distance_total = tot_dist,
    passing_distance_progressive = prg_dist,
    short_passes_completed = cmp_2,
    short_passes_attempted = att_2,
    short_passes_completion_percent = cmp_percent_2,
    medium_passes_completed = cmp_3,
    medium_passes_attempted = att_3,
    medium_passes_completion_percent = cmp_percent_3,
    long_passes_completed = cmp_4,
    long_passes_attempted = att_4,
    long_passes_completion_percent = cmp_percent_4,
    assists = ast,
    expected_assisted_goals = x_ag,
    expected_assists = x_a,
    assists_minus_expected_assisted_goals = a_x_ag,
    key_passes = kp,
    passes_into_final_third = x1_3,
    passes_into_penalty_area = ppa,
    crosses_into_penalty_area = crs_pa,
    progressive_passes = prg_p,

    player_link = link_to_player_page) %>% # remove 'player_id' as it is no longer needed
  select(-player_id)
```

6.12 Write the data to a csv file

```
write_csv(passing_players, "passing_players.csv")
```

6.13 Write the data to an Excel file

```
write.xlsx(passing_players, "passing_players.xlsx")
```

7 Scrape Passing Types Stats Page

7.1 Read the live html tables from FBRef

```
types_passing_euro_tables <- read_html_live(passing_types_stats) %>%
  html_table(fill = T)
```

7.2 Scrape the table of interest (3rd in this case) and convert it to a data frame - this has the individual player data

```
types_passing_players <- types_passing_euro_tables %>%
  .[[3]] %>%
  as.data.frame()
```

7.3 Convert the 1st row to column names and remove it from the data

```
colnames(types_passing_players) <- types_passing_players[1, ]
types_passing_players <- types_passing_players[-1, ]
```

7.4 Clean the column names and remove the rows with recurring column names

```
types_passing_players <- types_passing_players %>%
  janitor::clean_names() %>%
  filter(player != "Player")
```

7.5 Remove the "matches" column as it is not needed

```
types_passing_players <- types_passing_players %>%
  select(-matches)
```

7.6 Convert the numeric columns to numeric

```
types_passing_players <- types_passing_players %>%
  mutate(across(c(1, 5:ncol(.)), as.numeric))
```

7.6.1 Scrape the player ids from the player links in the live html table

```
player_id <- read_html_live(passing_types_stats) %>%
  html_nodes("table") %>%
  html_nodes("tbody") %>%
  html_elements("a") %>%
  html_attr("href") %>%
  as.data.frame() %>%
  setNames("url_info") %>%
  # to this point the url also contains the player matchlogs so we need to filter out those
  mutate(get_players = ifelse(grepl(pattern = '/players/', url_info), 1, 0)) %>%
  mutate(get_matchlogs = ifelse(grepl(pattern = '/matchlogs/', url_info), 1, 0)) %>%
  filter(get_players == 1 & get_matchlogs == 0) %>%
  select(-get_players) %>%
  select(-get_matchlogs) %>%
```

```r
  mutate(player_id = gsub("\\..*","", url_info),
         player_id = gsub(".*/[a-z]/","", player_id))
```

7.7 Join the datasets together

```r
types_passing_players <- types_passing_players %>%
  bind_cols(player_id) %>%
  mutate(url_info = paste0("https://fbref.com/", url_info)) %>%
  rename(link_to_player_page = url_info)
```

7.8 As we don’t need the ‘rk’ and ‘age’ columns, we can remove it to clean up the data into its final format

```r
types_passing_players <- types_passing_players %>%
  select(-rk, -age)
```

7.9 Calculate an new ‘age’ column from the ‘born’ column to 2024

```r
types_passing_players <- types_passing_players %>%
  mutate(age = 2024 - as.numeric(str_sub(born, start = -4)))
```

7.10 Remove the ‘born’ column as it is no longer needed and position the ‘age’ column next to the ‘player’ column

```r
types_passing_players <- types_passing_players %>%
  select(-born) %>%
  select(player, age, everything())
```

7.11 As column 15 contains the inswinging corner kicks, we can rename it to make it more descriptive

```r
colnames(types_passing_players)[15] <- "inswinging_corner_kicks"
```

7.12 Rename every other column to make them more descriptive

```r
types_passing_players <- types_passing_players %>%
  rename(
    position = pos,
    country = squad,
    played_90 = x90s,
    passes_attempted = att,
    live_ball_passes = live,
    dead_ball_passes = dead,
    free_kick_passes = fk,
    through_balls_completed = tb,
    switches_over_40yrds = sw,
    crosses = crs,
    throws_ins_taken = ti,
    corner_kicks = ck,
    outswinging_corner_kicks = out,
    straight_corner_kicks = str,
    completed_passes = cmp,
    passes_offside = off,
    passes_blocked = blocks,

    player_link = link_to_player_page) %>% # remove 'player_id' as it is no longer needed
  select(-player_id)
```

7.13 Write the data to a csv file

```r
write_csv(types_passing_players, "types_passing_players.csv")
```

7.14 Write the data to an Excel file

```r
write.xlsx(types_passing_players, "types_passing_players.xlsx")
```

8 Scrape Creation Stats Page

8.1 Read the live html tables from FBRef

```r
creation_euro_tables <- read_html_live(creation_stats) %>%
  html_table(fill = T)
```

8.2 Scrape the table of interest (3rd in this case) and convert it to a data frame - this has the individual player data

```r
creation_players <- creation_euro_tables %>%
  .[[3]] %>%
  as.data.frame()
```

8.3 Convert the 1st row to column names and remove it from the data

```r
colnames(creation_players) <- creation_players[1, ]
creation_players <- creation_players[-1, ]
```

8.4 Clean the column names and remove the rows with recurring column names

```r
creation_players <- creation_players %>%
  janitor::clean_names() %>%
  filter(player != "Player")
```

8.5 Remove the "matches" column as it is not needed

```r
creation_players <- creation_players %>%
  select(-matches)
```

8.6 Convert the numeric columns to numeric

```r
creation_players <- creation_players %>%
  mutate(across(c(1, 5:ncol(.)), as.numeric))
```

8.6.1 Scrape the player ids from the player links in the live html table

```r
player_id <- read_html_live(creation_stats) %>%
  html_nodes("table") %>%
  html_nodes("tbody") %>%
  html_elements("a") %>%
  html_attr("href") %>%
  as.data.frame() %>%
  setNames("url_info") %>%
  # to this point the url also contains the player matchlogs so we need to filter out those
  mutate(get_players = ifelse(grepl(pattern = '/players/', url_info), 1, 0)) %>%
  mutate(get_matchlogs = ifelse(grepl(pattern = '/matchlogs/', url_info), 1, 0)) %>%
  filter(get_players == 1 & get_matchlogs == 0) %>%
  select(-get_players) %>%
  select(-get_matchlogs) %>%
  mutate(player_id = gsub("\\..*","", url_info),
         player_id = gsub(".*/[a-z]/","", player_id))
```

8.7 Join the datasets together

```r
creation_players <- creation_players %>%
  bind_cols(player_id) %>%
  mutate(url_info = paste0("https://fbref.com/", url_info)) %>%
  rename(link_to_player_page = url_info)
```

8.8 As we don't need the 'rk' and 'age' columns, we can remove it to clean up the data into its final format

```r
creation_players <- creation_players %>%
  select(-rk, -age)
```

8.9 Calculate an new 'age' column from the 'born' column to 2024

```r
creation_players <- creation_players %>%
  mutate(age = 2024 - as.numeric(str_sub(born, start = -4)))
```

8.10 Remove the ‘born’ column as it is no longer needed and position the ‘age’ column next to the ‘player’ column

```r
creation_players <- creation_players %>%
  select(-born) %>%
  select(player, age, everything())
```

8.11 Rename every column to make them more descriptive

```r
creation_players <- creation_players %>%
  rename(
    position = pos,
    country = squad,
    played_90 = x90s,
    shot_creating_actions = sca,
    shot_creating_actions_per_90 = sca90,
    live_pass_shot_creating_actions = pass_live,
    dead_ball_shot_creating_actions = pass_dead,
    take_on_to_shot = to,
    shot_to_shot_creating_actions = sh,
    fouled_to_shot_creating_actions = fld,
    defensive_action_to_shot_creating_actions = def,
    goal_creating_actions = gca,
    goal_creating_actions_per_90 = gca90,
    live_pass_goal_creating_actions = pass_live_2,
    dead_ball_goal_creating_actions = pass_dead_2,
    take_on_to_goal_creating_actions = to_2,
    shot_to_goal_creating_actions = sh_2,
    fouled_to_goal_creating_actions = fld_2,
    defensive_action_to_goal_creating_actions = def_2,

    player_link = link_to_player_page) %>% # remove 'player_id' as it is no longer needed
  select(-player_id)
```

8.12 Write the data to a csv file

```r
write_csv(creation_players, "creation_players.csv")
```

8.13 Write the data to an Excel file

```r
write.xlsx(creation_players, "creation_players.xlsx")
```

9 Scrape Defensive Stats Page

9.1 Read the live html tables from FBRef

```r
defence_euro_tables <- read_html_live(defensive_stats) %>%
  html_table(fill = T)
```

9.2 Scrape the table of interest (3rd in this case) and convert it to a data frame - this has the individual player data

```r
defence_players <- defence_euro_tables %>%
  .[[3]] %>%
  as.data.frame()
```

9.3 Convert the 1st row to column names and remove it from the data

```r
colnames(defence_players) <- defence_players[1, ]
defence_players <- defence_players[-1, ]
```

9.4 Clean the column names and remove the rows with recurring column names

```r
defence_players <- defence_players %>%
  janitor::clean_names() %>%
  filter(player != "Player")
```

9.5 Remove the “matches” column as it is not needed

```r
defence_players <- defence_players %>%
  select(-matches)
```

9.6 Convert the numeric columns to numeric

```r
defence_players <- defence_players %>%
  mutate(across(c(1, 5:ncol(.)), as.numeric))
```

9.6.1 Scrape the player ids from the player links in the live html table

```r
player_id <- read_html_live(defensive_stats) %>%
  html_nodes("table") %>%
  html_nodes("tbody") %>%
  html_elements("a") %>%
  html_attr("href") %>%
  as.data.frame() %>%
  setNames("url_info") %>%
  # to this point the url also contains the player matchlogs so we need to filter out those
  mutate(get_players = ifelse(grepl(pattern = '/players/', url_info), 1, 0)) %>%
  mutate(get_matchlogs = ifelse(grepl(pattern = '/matchlogs/', url_info), 1, 0)) %>%
  filter(get_players == 1 & get_matchlogs == 0) %>%
  select(-get_players) %>%
  select(-get_matchlogs) %>%
  mutate(player_id = gsub("\\..*","", url_info),
         player_id = gsub(".*/[a-z]/","", player_id))
```

9.7 Join the datasets together

```r
defence_players <- defence_players %>%
  bind_cols(player_id) %>%
  mutate(url_info = paste0("https://fbref.com/", url_info)) %>%
  rename(link_to_player_page = url_info)
```

9.8 As we don’t need the ‘rk’ and ‘age’ columns, we can remove it to clean up the data into its final format

```r
defence_players <- defence_players %>%
  select(-rk, -age)
```

9.9 Calculate an new 'age' column from the 'born' column to 2024

```
defence_players <- defence_players %>%
  mutate(age = 2024 - as.numeric(str_sub(born, start = -4)))
```

9.10 Remove the 'born' column as it is no longer needed and position the 'age' column next to the 'player' column

```
defence_players <- defence_players %>%
  select(-born) %>%
  select(player, age, everything())
```

9.11 Rename every column to make them more descriptive

```
defence_players <- defence_players %>%
  rename(
    position = pos,
    country = squad,
    played_90 = x90s,
    players_tackled = tkl,
    tackles_won = tkl_w,
    tackles_def_3rd = def_3rd,
    tackles_mid_3rd = mid_3rd,
    tackles_att_3rd = att_3rd,
    dribblers_tackled = tkl_2,
    dribbles_challenged = att,
    dribblers_tackled_success_percent = tkl_percent,
    dribblers_tackled_unsuccessful = lost,
    blocks = blocks,
    shots_blocked = sh,
    passes_blocked = pass,
    interceptions = int,
    tackles_plus_interceptions = tkl_int,
    clearances = clr,
    errors = err,

    player_link = link_to_player_page) %>% # remove 'player_id' as it is no longer needed
  select(-player_id)
```

9.12 Write the data to a csv file

```
write_csv(defence_players, "defence_players.csv")
```

9.13 Write the data to an Excel file

```
write.xlsx(defence_players, "defence_players.xlsx")
```

10 Scrape Possession Stats Page

10.1 Read the live html tables from FBRef

```
possession_euro_tables <- read_html_live(possession_stats) %>%
  html_table(fill = T)
```

10.2 Scrape the table of interest (3rd in this case) and convert it to a data frame - this has the individual player data

```
possession_players <- possession_euro_tables %>%
  .[[3]] %>%
  as.data.frame()
```

10.3 Convert the 1st row to column names and remove it from the data

```
colnames(possession_players) <- possession_players[1, ]
possession_players <- possession_players[-1, ]
```

10.4 Clean the column names and remove the rows with recurring column names

```
possession_players <- possession_players %>%
  janitor::clean_names() %>%
  filter(player != "Player")
```

10.5 Remove the “matches” column as it is not needed

```r
possession_players <- possession_players %>%
  select(-matches)
```

10.6 Convert the numeric columns to numeric

```r
possession_players <- possession_players %>%
  mutate(across(c(1, 5:ncol(.)), as.numeric))
```

10.6.1 Scrape the player ids from the player links in the live html table

```r
player_id <- read_html_live(possession_stats) %>%
  html_nodes("table") %>%
  html_nodes("tbody") %>%
  html_elements("a") %>%
  html_attr("href") %>%
  as.data.frame() %>%
  setNames("url_info") %>%
  # to this point the url also contains the player matchlogs so we need to filter out those
  mutate(get_players = ifelse(grepl(pattern = '/players/', url_info), 1, 0)) %>%
  mutate(get_matchlogs = ifelse(grepl(pattern = '/matchlogs/', url_info), 1, 0)) %>%
  filter(get_players == 1 & get_matchlogs == 0) %>%
  select(-get_players) %>%
  select(-get_matchlogs) %>%
  mutate(player_id = gsub("\\..*","", url_info),
         player_id = gsub(".*/[a-z]/","", player_id))
```

10.7 Join the datasets together

```r
possession_players <- possession_players %>%
  bind_cols(player_id) %>%
  mutate(url_info = paste0("https://fbref.com/", url_info)) %>%
  rename(link_to_player_page = url_info)
```

10.8 As we don’t need the ‘rk’ and ‘age’ columns, we can remove it to clean up the data into its final format

```r
possession_players <- possession_players %>%
  select(-rk, -age)
```

10.9 Calculate an new ‘age’ column from the ‘born’ column to 2024

```r
possession_players <- possession_players %>%
  mutate(age = 2024 - as.numeric(str_sub(born, start = -4)))
```

10.10 Remove the ‘born’ column as it is no longer needed and position the ‘age’ column next to the ‘player’ column

```r
possession_players <- possession_players %>%
  select(-born) %>%
  select(player, age, everything())
```

10.11 Rename every column to make them more descriptive

```r
possession_players <- possession_players %>%
  rename(
    position = pos,
    country = squad,
    played_90 = x90s,
    defensive_penalty_area_touches = def_pen,
    defensive_third_touches = def_3rd,
    middle_third_touches = mid_3rd,
    attacking_third_touches = att_3rd,
    attacking_penalty_area_touches = att_pen,
    live_ball_touches = live,
    take_ons_attempted = att,
    take_ons_successful = succ,
    take_ons_success_percent = succ_percent,
    tackled_during_take_on = tkld,
    tackled_during_take_on_percent = tkld_percent,
    total_dribbling_distance_yrds = tot_dist,
    progressive_dribbling_distance_yrds = prg_dist,
    progressive_carries = prg_c,
    carries_into_final_third = x1_3,
    carries_into_penalty_area = cpa,
    miscontrols = mis,
    dispossessed = dis,
    passes_recieved = rec,
    progressive_passes_recieved = prg_r,

    player_link = link_to_player_page) %>% # remove 'player_id' as it is no longer needed
  select(-player_id)
```

10.12 Write the data to a csv file

```
write_csv(possession_players, "possession_players.csv")
```

10.13 Write the data to an Excel file

```
write.xlsx(possession_players, "possession_players.xlsx")
```

11 Scrape Playing Time Stats Page

11.1 Read the live html tables from FBRef

```
playing_time_euro_tables <- read_html_live(playing_time_stats) %>%
  html_table(fill = T)
```

11.2 Scrape the table of interest (3rd in this case) and convert it to a data frame - this has the individual player data

```
playing_time_players <- playing_time_euro_tables %>%
  .[[3]] %>%
  as.data.frame()
```

11.3 Convert the 1st row to column names and remove it from the data

```r
colnames(playing_time_players) <- playing_time_players[1, ]
playing_time_players <- playing_time_players[-1, ]
```

11.4 Clean the column names and remove the rows with recurring column names

```r
playing_time_players <- playing_time_players %>%
  janitor::clean_names() %>%
  filter(player != "Player")
```

11.5 Remove the “matches” column as it is not needed

```r
playing_time_players <- playing_time_players %>%
  select(-matches)
```

11.6 Convert the numeric columns to numeric

```r
playing_time_players <- playing_time_players %>%
  mutate(across(c(1, 5:ncol(.)), as.numeric))
```

11.6.1 Scrape the player ids from the player links in the live html table

```r
player_id <- read_html_live(playing_time_stats) %>%
  html_nodes("table") %>%
  html_nodes("tbody") %>%
  html_elements("a") %>%
  html_attr("href") %>%
  as.data.frame() %>%
  setNames("url_info") %>%
  # to this point the url also contains the player matchlogs so we need to filter out those
  mutate(get_players = ifelse(grepl(pattern = '/players/', url_info), 1, 0)) %>%
  mutate(get_matchlogs = ifelse(grepl(pattern = '/matchlogs/', url_info), 1, 0)) %>%
  filter(get_players == 1 & get_matchlogs == 0) %>%
  select(-get_players) %>%
  select(-get_matchlogs) %>%
  mutate(player_id = gsub("\\..*","", url_info),
         player_id = gsub(".*/[a-z]/","", player_id))
```

11.7 Join the datasets together

```
playing_time_players <- playing_time_players %>%
  bind_cols(player_id) %>%
  mutate(url_info = paste0("https://fbref.com/", url_info)) %>%
  rename(link_to_player_page = url_info)
```

11.8 As we don’t need the ‘rk’ and ‘age’ columns, we can remove it to clean up the data into its final format

```
playing_time_players <- playing_time_players %>%
  select(-rk, -age)
```

11.9 Calculate an new ‘age’ column from the ‘born’ column to 2024

```
playing_time_players <- playing_time_players %>%
  mutate(age = 2024 - as.numeric(str_sub(born, start = -4)))
```

11.10 Remove the ‘born’ column as it is no longer needed and position the ‘age’ column next to the ‘player’ column

```
playing_time_players <- playing_time_players %>%
  select(-born) %>%
  select(player, age, everything())
```

11.11 Rename every column to make them more descriptive

```
playing_time_players <- playing_time_players %>%
  rename(
    position = pos,
    country = squad,
    matches_played = mp,
    minutes_played = min,
    minutes_per_match_played = mn_mp,
    percent_of_minutes_played = min_percent,
    played_90 = x90s,
    minutes_per_match_started = mn_start,
    completed_matches = compl,
    substitute_appearances = subs,
    minutes_per_substitute_appearance = mn_sub,
    unused_substitute = un_sub,
    points_per_match = ppm,
    goals_scored = on_g,
    goals_conceded = on_ga,
    goal_difference = x,
    total_goal_difference_per_90 = x90,
    goal_difference_per_appearance = on_off,
    xg_while_on_the_pitch = onx_g,
    xg_against_while_on_the_pitch = onx_ga,
    xg_difference = x_g,
    xg_difference_per_90 = x_g_90,
    xg_difference_per_appearance = on_off_2,

    player_link = link_to_player_page) %>% # remove 'player_id' as it is no longer needed
  select(-player_id)
```

11.12 Write the data to a csv file

```
write_csv(playing_time_players, "playing_time_players.csv")
```

11.13 Write the data to an Excel file

```
write.xlsx(playing_time_players, "playing_time_players.xlsx")
```

12 Scrape Misc Stats Page

12.1 Read the live html tables from FBRef

```
misc_euro_tables <- read_html_live(misc_stats) %>%
  html_table(fill = T)
```

12.2 Scrape the table of interest (3rd in this case) and convert it to a data frame - this has the individual player data

```
misc_players <- misc_euro_tables %>%
  .[[3]] %>%
  as.data.frame()
```

12.3 Convert the 1st row to column names and remove it from the data

```
colnames(misc_players) <- misc_players[1, ]
misc_players <- misc_players[-1, ]
```

12.4 Clean the column names and remove the rows with recurring column names

```
misc_players <- misc_players %>%
  janitor::clean_names() %>%
  filter(player != "Player")
```

12.5 Remove the "matches" column as it is not needed

```
misc_players <- misc_players %>%
  select(-matches)
```

12.6 Convert the numeric columns to numeric

```
misc_players <- misc_players %>%
  mutate(across(c(1, 5:ncol(.)), as.numeric))
```

12.6.1 Scrape the player ids from the player links in the live html table

```r
player_id <- read_html_live(misc_stats) %>%
  html_nodes("table") %>%
  html_nodes("tbody") %>%
  html_elements("a") %>%
  html_attr("href") %>%
  as.data.frame() %>%
  setNames("url_info") %>%
  # to this point the url also contains the player matchlogs so we need to filter out those
  mutate(get_players = ifelse(grepl(pattern = '/players/', url_info), 1, 0)) %>%
  mutate(get_matchlogs = ifelse(grepl(pattern = '/matchlogs/', url_info), 1, 0)) %>%
  filter(get_players == 1 & get_matchlogs == 0) %>%
  select(-get_players) %>%
  select(-get_matchlogs) %>%
  mutate(player_id = gsub("\\..*","", url_info),
         player_id = gsub(".*/[a-z]/","", player_id))
```

12.7 Join the datasets together

```r
misc_players <- misc_players %>%
  bind_cols(player_id) %>%
  mutate(url_info = paste0("https://fbref.com/", url_info)) %>%
  rename(link_to_player_page = url_info)
```

12.8 As we don't need the 'rk' and 'age' columns, we can remove it to clean up the data into its final format

```r
misc_players <- misc_players %>%
  select(-rk, -age)
```

12.9 Calculate an new 'age' column from the 'born' column to 2024

```r
misc_players <- misc_players %>%
  mutate(age = 2024 - as.numeric(str_sub(born, start = -4)))
```

12.10 Remove the ‘born’ column as it is no longer needed and position the ‘age’ column next to the ‘player’ column

```r
misc_players <- misc_players %>%
  select(-born) %>%
  select(player, age, everything())
```

12.11 Rename every column to make them more descriptive

```r
misc_players <- misc_players %>%
  rename(
    position = pos,
    country = squad,
    played_90 = x90s,
    yellow_cards = crd_y,
    red_cards = crd_r,
    second_yellow_cards = x2crd_y,
    fouls_committed = fld,
    fouls_drawn = fls,
    offsides = off,
    crosses = crs,
    interceptions = int,
    tackles_won = tkl_w,
    penalty_kicks_won = p_kwon,
    penalty_kicks_conceded = p_kcon,
    own_goals = og,
    loose_ball_recoveries = recov,
    aerial_duels_won = won,
    aerial_duels_lost = lost,
    aerial_duels_won_percent = won_percent,

    player_link = link_to_player_page) %>% # remove 'player_id' as it is no longer needed
  select(-player_id)
```

12.12 Write the data to a csv file

```r
write_csv(misc_players, "misc_players.csv")
```

12.13 Write the data to an Excel file

```r
write.xlsx(misc_players, "misc_players.xlsx")
```

13 Combine all csv files into one

```r
files <- list.files(pattern = "*.csv")
all_players <- map_df(files, read_csv)
```

```
Rows: 29 Columns: 31
-- Column specification --------------------------------------------------------
Delimiter: ","
chr  (4): player, position, country, player_link
dbl (27): age, played_90, goals_against, penalty_kicks_allowed, free_kick_go...

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
Rows: 621 Columns: 199
-- Column specification --------------------------------------------------------
Delimiter: ","
chr   (4): player, player_link, position, country
dbl (195): age, played_90, goals_against, penalty_kicks_allowed, free_kick_g...

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
Rows: 493 Columns: 22
-- Column specification --------------------------------------------------------
Delimiter: ","
chr  (4): player, position, country, player_link
dbl (18): age, played_90, shot_creating_actions, shot_creating_actions_per_9...

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
Rows: 493 Columns: 22
-- Column specification --------------------------------------------------------
Delimiter: ","
chr  (4): player, position, country, player_link
dbl (18): age, played_90, players_tackled, tackles_won, tackles_def_3rd, tac...
```

```
i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
Rows: 29 Columns: 24
-- Column specification --------------------------------------------------------
Delimiter: ","
chr  (4): player, position, country, player_link
dbl (20): age, matches_played, starts, minutes_played, played_90, goals_agai...

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
Rows: 493 Columns: 22
-- Column specification --------------------------------------------------------
Delimiter: ","
chr  (4): player, position, country, player_link
dbl (18): age, played_90, yellow_cards, red_cards, second_yellow_cards, foul...

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
Rows: 493 Columns: 29
-- Column specification --------------------------------------------------------
Delimiter: ","
chr  (4): player, position, country, player_link
dbl (25): age, played_90, passes_completed, passes_attempted, pass_completio...

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
Rows: 621 Columns: 27
-- Column specification --------------------------------------------------------
Delimiter: ","
chr  (4): player, position, country, player_link
dbl (23): age, matches_played, minutes_played, minutes_per_match_played, per...

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
Rows: 493 Columns: 28
-- Column specification --------------------------------------------------------
Delimiter: ","
chr  (4): player, position, country, player_link
dbl (24): age, played_90, touches, defensive_penalty_area_touches, defensive...

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
Rows: 493 Columns: 23
-- Column specification --------------------------------------------------------
Delimiter: ","
chr  (4): player, position, country, player_link
dbl (19): age, played_90, goals, shots, shots_on_target, shots_on_target_per...

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
Rows: 493 Columns: 34
-- Column specification --------------------------------------------------------
Delimiter: ","
chr  (4): player, position, country, player_link
dbl (30): age, matches_played, starts, minutes_played, played_90, goals, ass...

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
Rows: 493 Columns: 21
-- Column specification --------------------------------------------------------
Delimiter: ","
chr  (4): player, position, country, player_link
dbl (17): age, played_90, passes_attempted, live_ball_passes, dead_ball_pass...

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

13.1 Move 'player_link' column to after 'player' column and Move 'age' column to after 'country' column

```r
all_players <- all_players %>% select(player, player_link, position, country,
                                      age, everything())
```

13.2 Replace all numeric NA values with 0

```r
all_players <- all_players %>% mutate_if(is.numeric, ~replace(., is.na(.), 0))
```

13.3 Remove all rows with dublicated player_link

```
all_players <- all_players %>% distinct(player_link, .keep_all = TRUE)
```

13.4 Write the combined data to a csv file

```
write_csv(all_players, "all_players.csv")
```

13.5 Write the combined data to an Excel file

```
write.xlsx(all_players, "all_players.xlsx")
```

Now you have a nice shiny new dataset with all the player data from the Euro 2024 tournament in both csv and Excel formats. Now to explore - Enjoy!