Bartosz Korta, 246709

DOKUMENTACJA PROJEKTU

„Personal NBA Tracker”

Przedmiot:	Języki skryptowe
Imię i nazwisko autora:	Bartosz Korta
Nr indeksu:	246709
Semestr studiów:	5
Data ukończenia pracy:	Styczeń 2021 r.
Prowadzący laboratorium:	mgr inż. Natalia Piórkowska

1. Wymagania projektowe:

Celem projektu było stworzenie aplikacji desktopowej umożliwiającej użytkownikom śledzenie postępów swoich ulubionych drużyn z koszykarskiej ligi NBA.

1.1. Wymaganie funkcjonalne:

- System pozwala przeprowadzić autentykację użytkowników. Tzn.: Logowanie i wylogowywanie na stworzone konto oraz rejestrację nowego konta
- System pozwala zalogowanym użytkownikom wybrać swoje ulubione drużyny NBA, których postępy chce śledzić
- System pozwala zmienić wcześniej wybrane drużyny przez użytkownika, w dowolnej chwili
- System w chwili uruchamiania aplikacji pobierania wymagane dane z internetu, ze stron internetowych dostarczających pożądane statystyki
- System umożliwia przeglądanie wyników meczy, które się odbyły z udziałem wybranych wcześniej przez siebie drużyn oraz meczy, które zostaną rozegrane w przyszłości
- System umożliwia przeglądanie aktualnie tabeli NBA, wraz z różnymi statystykami drużyn
- System umożliwia przeglądanie nagłówków najświeższych artykułów, dotyczących ulubionych drużyn, oraz umożliwia uruchomienie wybranego artykułu w nowym oknie przeglądarki za pomocą jednego przycisku
- System umożliwia przeglądanie danych zawodników wybranych wcześniej drużyn oraz za pomocą kliknięcia na nazwisko zawodnika, uruchamia stronę internową poświęconą dogłębnym statystykom wybranego zawodnika

1.2. Wymagania niefunkcjonalne

- Aby dokonać autentykacji konta oraz uruchomić główne okno aplikacji należy mieć załączone wszystkie pliki w jednym folderze, w tym rownież plik bazy danych

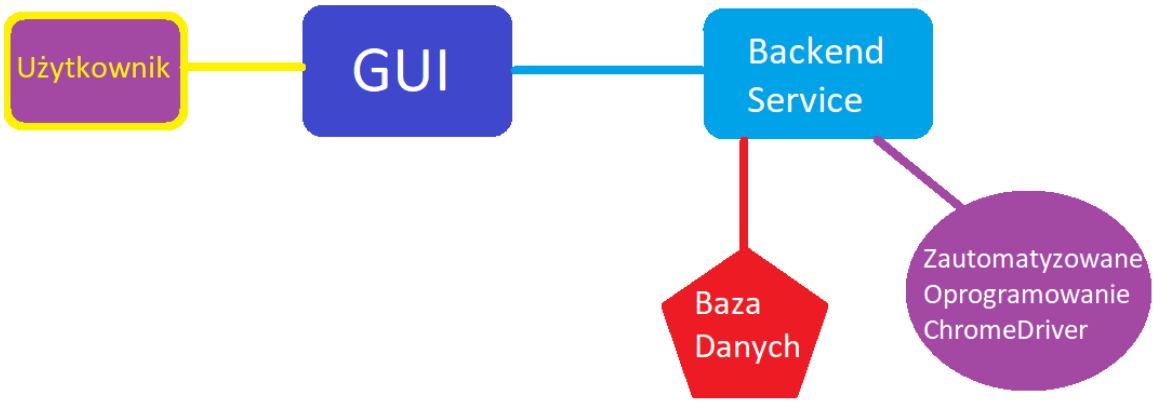
- Aby aplikacja mogła prawidłowo korzystać należy mieć zainstalowane następujące programy: selenium webdriver, Chromedriver oraz poprawnie ustawioną zmienną PATH w pliku main.py. Wszystkie procesy instalacji w PATRZ: Punkt X.X
- Aby aplikacja mogła prawidłowo pobierać dane z przeglądarki oraz uruchamiać w nowym oknie wybrane strony należy mieć nawiązanie połączenie z siecią.
- Aby uruchomić aplikację należy otworzyć plik z rozszerzeniem .exe
- Aby w pełni korzystać z funkcjonalności aplikacji, po zarejestrowaniu oraz zalogowaniu należy wybrać co najmniej jedną drużynę w zakładce: „Fav Teams” (Dopóki nie zostanie ona wybrana pozostałe funkcjonalności są zablokowane)

2. Architektura systemu

2.1. Opis architektury systemu

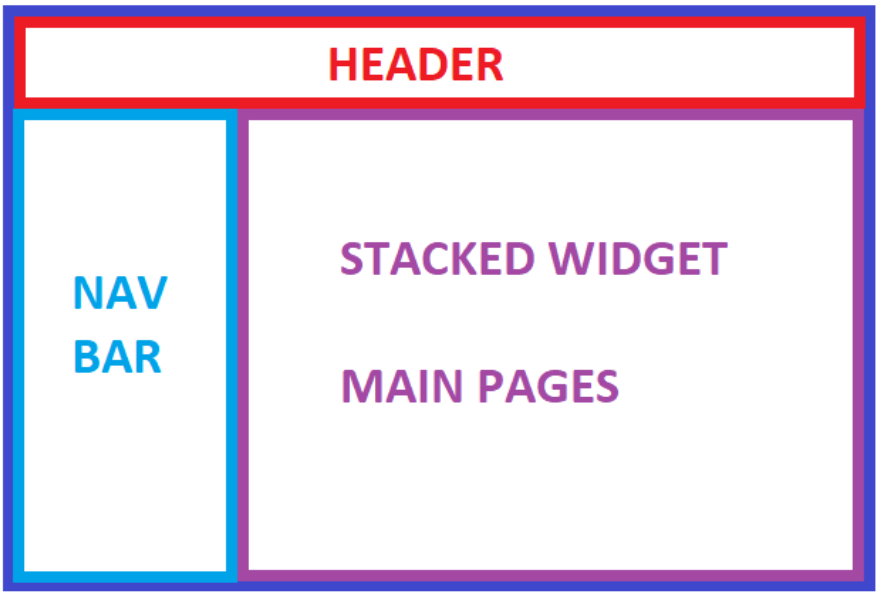
System można podzielić na następujące części: Graficzny interfejs użytkownika, którego obsługą zajmuje się główną częścią logiczny aplikacji. Serwis logiczny komunikuje się bazą danych. Serwis logiczny również nawiązuje połączenie ze stronami internetowymi, z których pobiera dane oraz przekierowuje na wskazane strony. Dokonuje się to przy użyciu zautomatyzowanego oprogramowania, którym część logiczna aplikacji steruje.

2.2. Schemat architektury systemu



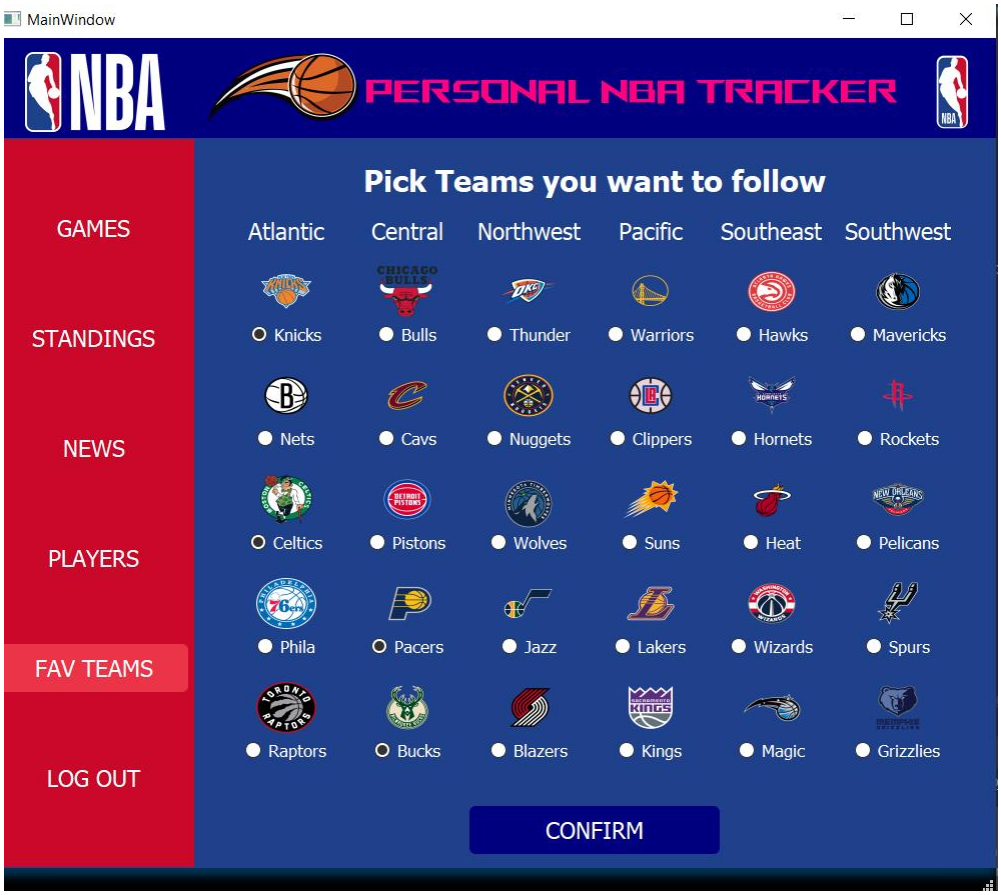
3. Prezentacja interfejsu wraz ze sposobem użycia

Główne okno aplikacji podzielone jest na trzy wyróżniające się cześci. Header, nagłówek jest niezmieniającym się elementem. Navigation Bar jest elementem, gdzie możemy sprawnie przemieszczać się pomiędzy stronami aplikacji. Tam znajdują się przyciski, które umożliwiają zmiane widoku Main Pages. Main Pages jest to element, którego widok zmienia się wraz z wybranym przyciskiem przez użytkownika w Nav Barze. StackedWidget jest rozwiązaniem, które na to pozwala, sprawnie poruszając się pomiędzy kolejnymi stronami. Całość oddaje następujący szkic:



Korzystanie z różnych funkcjonalności umożliwiają przyciski umieszczone w oknie aplikacji. Wyróżnione jest 5 głównych okien, odpowiedzialnych za różne funkcjonalności. „Games” – wyświetla mecze rozegrane oraz terminarz dla wybranych przez użytkownika drużyn. „Standings” – wyświetla aktualną tabelę NBA. „News” – dla każdej wybranej drużyny przez użytkownika, wyświetla TOP 5 najświeższych artykułów dotyczących drużyny i zapewnia mozliwość uruchomienia ich w nowym oknie przeglądarki. „Players” – wyświetla podstawowe dane zawodników, którzy grają w wybranych drużynach wraz z możliwością wyświetlania szczegółowych statystyk w nowym oknie przeglądarki. „Fav Teams” – umożliwia dokonania wyboru ulubionych drużyn. Dodatkowo „Log Out” zapewnia możliwość wylogowania się z obecnego konta i zamknięcia sesji.

Poniżej kilka zdjęć prezentujących główne okno interfejsu:



Okna interfejsu odpowiedzialne za autentykację, przy każdym uruchomieniu aplikacji wyświetlają się jako pierwsze. Komunikację między nimi odwzorowuje zdjęcie zamieszczone poniżej.

Na początku otwierane jest okno Sign In, które umożliwia zalogowanie się użytkownikom na swoje konto. Gdy użytkownik nie ma utworzonego konta w bazie, powinien wybrać przycisk „CLICK HERE”, który następnie otworzy okno Sign Up. W tym oknie użytkownik może stworzyć nowe konto. Po przyciśnieciu „REGISTER”, jeżeli wszystko poszło pomyślnie okno Sign Up jest zamykane i następuje powrót do Sign In, gdzie użytkownik może się już zalogować na nowo utworzone konto.

Po przyciśnieciu Log In następuje uruchomienie nowego okna „Loading Screen”, w trakcie którego w tle część logiczna aplikacji pobiera wymagane dane do funkcjonowania programu.

Następnie po załadowaniu się wszystkich wymaganych danych uruchamia się główne okno aplikacji opisane wcześniej.

4. Schemat bazy danych

Baza danych systemu znajduje się w pliku nba.db, jest ona utworzona, aktualizowana za pomocą biblioteki sqlite3. Przechowywane są w niej informacje, które są statyczne, tzn nie ulegają zmianie z dnia na dzień. Pozostałe dane takie jak wyniki meczy, tabela czy dane nowych artykułów są pobierane na bieżąco w momencie uruchomienia programu.
Baza danych przechowuje następujące tabele:

Teams – Tabela przechowująca stałe informacje o drużynach pola: nazwa, skrót_drużyny, link_do_strony_z_artykułami

Users – Tabela przechowująca dane użytkownika, pola: username i password

Favteams – Tabela przechowująca wybrane ulubione drużyny przez użytkownika pola: username oraz nazwa_drużyny

Players – Tabela przechowująca dane zawodników grających w NBA. Dane te zostały pobrane przy użyciu oprogramowania sterującego przeglądarką i zapakowane do bazy danych. Pola: drużyna, imie_nazwisko, pozycja, wiek, wzrost, waga, college, pensja, link_do_szczegółowych_statystyk

Standings – Tabela przechowująca aktualną punktacje drużyn. Jest to jedyna tabela, które przechowuje dynamiczne dane. Ponieważ mecze rozgrywane są codziennie, rekordy są modyfikowane przy każdym uruchomieniu Aplikacji. Najświeższe wyniki są pobierane przez zautomatyzowane oprogramownie. Pola: pozycja, nazwa_drużyny, wygrane, porażki, ratio, punkty_w_plecy, w_domu, na_wyjeździe, dywizja, konferencja, punkty_na_mecz, pkt_przeciwnika_na_mecz, seria, last10

5. Logika systemu oraz kod źródłowy

System rozbity jest na odpowiednie pliki odpowiedzialne za konkretne funkcjonalności aplikacji. Wyróżniamy następujące pliki:

Pliki odpowiedzialne za układ interfejsu graficznego:

ui.py – Plik okna głównego. Są tu osadzone wszystkie przyciski, labele i inne elementy umożliwiające obsługe tego okna.
signin.py – Plik okna logowania. Osadzone są przyciski i inne elementy odpowiedzialne za wygląd tego okna.
signup.py – Plik okna rejestracji. Odpowiedzialny za wygląd tego okna
loadingscreen.py – Plik okna ładowania. Odpowiedzialny za jego wygląd.

Plik odpowiedzialne za logikę interfejsu logicznego:

logic.py – Jest to plik, w którym elementy wszystkich okien interfejsu są podpięte pod logikę aplikacji, tzn.: obsługę pewnych zdarzeń. Zaimplementowano w nim klasy powyższych interfejsów, wraz z metodą init oraz wszystkim innymi metodami odpowiedzialnymi za prawidłowe funkcjonowanie. Dodatkowo znajduje się tu klasa ActiveUser, która pełni funkcję kontroli, aktywnego użytkownika oraz dostarczyciela danych, potrzebnych do obsługi danego użytkownika

Pliki odpowiedzialne za konfigurację bazy danych oraz danych dynamicznych:

database_config.py – Plik, w którym zostały zaimplementowane metody, tworzące nowe tabele, wyciągające dane z tabel, wkładające nowe dane do bazy oraz pobierające dane z internetu i przygotowujące je do zapisania do bazy (wraz z zapisem). Odpowiedzialny za utrzymanie całej bazy i sterowanie nią

nba_info_config.py – Plik, który pobiera dynamicznie najświeższe informacje (artykuły czy mecze i terminarz), rozpracowuje je oraz zapisuje do danych sesji (tzn. Pól klasy ActiveUser, która przechowuje tą część logiczną tylko dla aktywnego użytkownika)

Inne pliki:

test.py – plik przechowujący obrazki używane w GUI statycznie, przekonwertowane na plik pythona.

folder images – dodatkowy folder gdzie trzymane są loga zespołów, używane przy dynamicznie zmieniających się danych

Prezentacja ważniejszych metod/funkcji:

Klasa AcitveUser, która trzyma dynamiczne dane obecnego użytkownika oraz wskaźniki na wybierane drużyny czy daty za pomocą przycisków

```
class ActiveUser:
    NAME = ''
    teams = []
    currentIt = 0
    newstable = []
    day_diff = 0
    matches = []
    @staticmethod
    def changeUsername(newusername):
        ActiveUser.NAME = newusername

    @staticmethod
    def clear():
        ActiveUser.NAME = ''

    @staticmethod
    def updateteams():
        ActiveUser.teams.clear()
        n=ActiveUser.NAME
        conn = sqlite3.connect('nba.db')
        c = conn.cursor()
        tcurs = c.execute("SELECT team FROM favteams WHERE username=?", (n,))
        teams = tcurs.fetchall()
        for i in teams:
            ActiveUser.teams.append(i[0])
```

Działanie: Dzięki niej bez problemu możemy powiązać wybory danego użytkownika z prawidłowym funkcjonowaniem aplikacji, tzn. Obsługą wybranych przez niego drużyn. Poza tym stanowi część autentykacji zapamiętując dane.

Metoda insert_rows_for_players, klasy MainWindow (Logika głównego okna):

```
def insert_rows_for_players(self):
    self.ui.playersTable.setEditTriggers(QtWidgets.QTreeView.NoEditTriggers)
    self.ui.playersTable.horizontalHeader().setVisible(True)
    conn = sqlite3.connect('nba.db')
    c = conn.cursor()
    tplayers = c.execute("SELECT playername, position, age, ht, wt, college, salary FROM players WHERE team=?",
                         (ActiveUser.teams[ActiveUser.currentIt],))
    players=tplayers.fetchall()
    counter = 0
    self.ui.playersTable.setColumnCount(7)
    self.ui.playersTable.setRowCount(len(players))
    for i in players:
        for j in range(0,len(i)):
                self.ui.playersTable.setItem(counter,j,QtWidgets.QTableWidgetItem(str(i[j])))

        counter+=1
    self.ui.playersTable.resizeColumnsToContents()
    conn.close()
```

Działanie: Wyciąga dane z tabeli players, w zależności od tego, która obecnie drużyna została wybrana przez użytkownika. Następnie dane to wkłada do kolejnych komórek tabeli, która zostanie wyświetlona. Ważna metoda z uwagi na to, że analogicznie do niej w taki sam sposób pracują jeszcze 3 inne metody

Metoda open_link Klasy MainWindow

```
def open_link(self):
    rows=self.ui.playersTable.selectedItems()
    plname=rows[0].text()
    conn = sqlite3.connect('nba.db')
    c = conn.cursor()
    c.execute("SELECT link FROM players WHERE team=? AND playername=?",(ActiveUser.teams[ActiveUser.currentIt],plname))
    currlink = c.fetchone()
    if currlink!=None:
        driver = webdriver.Chrome(PATH)
        driver.get(currlink[0])
```

Działanie: Metoda uruchamiana jest w momencie przyciśniecia nazwiska zawodnika, który znajduje się w tabeli (tej opisanej w metodzie wyżej). Następnie pobiera ona z bazy danych link, pod którym znajduje się adres strony internetowej trzymającej dane zawodnika. Potem uruchamia link używając ChromeDrivera. Ważna metoda w kontekście otwierania nowych kart przeglądarki zautomatyzowanym oprogramowaniem.

Metoda login w Klasie SignIn Window, odpowiada za sprawdzanie poprawności logowania.

```
def login(self):
    self.ui.alert.setText('')
    nick = self.ui.userInp.text()
    password = self.ui.passInp.text()
    conn = sqlite3.connect('nba.db')
    c = conn.cursor()
    name = c.execute("SELECT * FROM users WHERE username=? AND password=?", (nick,password,))
    odp2 = name.fetchone()
    if odp2==None:
        self.ui.alert.setText('Wrong password or username')
        return
    ActiveUser.NAME = nick
    ActiveUser.updateteams()
    get_articles(ActiveUser.teams)
    ActiveUser.setMatches(nba_info_config.get_matches(ActiveUser.teams))
    mainWindow = MainWindow()
    mainWindow.show()
    self.hide()
```

Działanie: Ustawia również tekst komunikatów w razie niepowodzenia logowania i weryfikuje poprawność tego procesu.

Metoda get_matches w plik nba_info_config.py

```
def get_matches(act_teams):
    opt = Options()
    opt.add_argument("--headless")
    driver = webdriver.Chrome(PATH,options=opt)
    driver.get('https://www.flashscore.pl/koszykowka/usa/nba/wyniki/')
    teamshome = driver.find_elements_by_class_name("event__participant.event__participant--home")
    teamsaway = driver.find_elements_by_class_name("event__participant.event__participant--away")
    homescores = driver.find_elements_by_class_name('event__score.event__score--home')
    awayscores = driver.find_elements_by_class_name('event__score.event__score--away')
    matchdate = driver.find_elements_by_class_name('event__time')
    for i in range(0,len(teamshome)):
        time =matchdate[i].text[0:5]
        if act_teams.__contains__(teamshome[i].text) or act_teams.__contains__(teamsaway[i].text):
            Matches.append(Match(teamshome[i].text,teamsaway[i].text,homescores[i].text,awayscores[i].text,time))

    driver.get('https://www.flashscore.pl/koszykowka/usa/nba/spotkania/')
    teamshome2 = driver.find_elements_by_class_name("event__participant.event__participant--home")
    teamsaway2 = driver.find_elements_by_class_name("event__participant.event__participant--away")
    matchdate2 = driver.find_elements_by_class_name('event__time')
    for i in range(0,len(matchdate2)):
        time = matchdate2[i].text[0:5]
        if act_teams.__contains__(teamshome2[i].text) or act_teams.__contains__(teamsaway2[i].text):
            Matches.append(Match(teamshome2[i].text,teamsaway2[i].text,'0','0',time))
    driver.close()
    return Matches
```

Działanie: Używając zautomatyzowanej przeglądarki dostaje się na odpowiednią stronę a następnie wyciąga z niej pożądane wskazane dane. Działa w trybie –headless dzięki czemu nie przeszkadza użytkownikowi i cały ten proces jest rozgrywany w tle. Następnie pobrane dane odpowiednio zwraca, dzięki czemu są przechowywane w

klasie ActiveUser w polu Mecze, gdzie zapewniony jest do nich szybki dostęp.

6. Sposób uruchomienia oraz wymagania.

- Wymagane jest stałe podłączenie do internetu, ponieważ jest on źródłem danych, z których aplikacja korzysta
- Wymagane jest zainstalowanie aplikacji ChromeDriver oraz biblioteki Selenium.
    - Proces instalacji ChromeDriver:

      Należy pobrać program ze strony:

      https://chromedriver.storage.googleapis.com/index.html?path=87.0.4280.88/

      Oraz zainstalować go w folderze „C:\Program Files\

      W przypadku wybrania innego folderu należy zmienić na początku pliku „main.py” zmienną PATH na odpowiedni folder + chromedriver.exe Przykład: „C:\Temp\chromedriver.exe”

      W razie problemu, film prezentujący instalację:

      https://www.youtube.com/watch?v=dz59GsdvUF8&ab_channel=ArturSpirin

    Proces instalacji biblioteki Selenium:

      Należy wpisać w oknie terminala komendę:

          pip install selenium

      (Wymagane posiadanie python oraz pip.

          1.Python install:
          https://www.python.org/downloads/

          Po instalacji Pythona:

          2.Pip install:

          Należy wpisać w wierszu poleceń komendę

`python get-pip.py`)

W razie problemu, film prezentujący instalację biblioteki Selenium:

https://www.youtube.com/watch?v=gVXcVcTRXd0&ab_channel=ArturSpirin

- Wymagane jest przechowywanie wszystkich plików źródłowych w tym samym folderze.
- Proces uruchomienia:
    - Gdy spełnimy wszystkie wymienione warunki wyżej możemy uruchomić aplikację
    - Należy uruchomić plik z rozszerzeniem .exe
    - Następnie kliknąć przycisk „Click Here” odpowiadający na „Don’t have account?”
    - W oknie rejestracji wpisać wymagane dane i wybrać przycisk „Register”
    - Następnie wprowadzić dane z utworzonego konta w oknie „Sign In” i kliknąć przycisk „Login”
    - Następnie wyskoczy ekran ładowania, poczekać aż skończy się ładować.
    - Następnie w głównym oknie, które zostanie wyświetlone zaznaczyć co najmniej jedną drużynę oraz kliknąć przycisk „Confirm”
    - Nastąpi ponowne przeładowanie aplikacji (dzieje się to tylko gdy wprowadzamy zmiany w „ulubionych drużynach”
    - Po przeładowaniu możemy korzystać z wszystkich funkcjonalności zapewnionych przez aplikację.

7. Podsumowanie

Aplikacja, która została przeze mnie utworzona ma w łatwy i przyjemny sposób umożliwić fanom koszykówki poczynania ich ulubionych drużyn. Zamiast przeskakiwać ze strony na stronę żeby wyciągąć interesujące informacje, aplikacja przechowuje najważniejsze dane bez zbędnych

wypełniaczy, które utrudniają sprawne przeglądanie statystyk, artykułów czy ciekawostek. Zbiera ona informacje z różnych stron internetowych, dzięki czemu użytkownik nie musi się martwić „skakaniem” między nimi, (być może w ogóle ich nie zna) aplikacja zrobi to za niego.

Strony internetowe, z których wyciągane są przeze mnie najważniejsze statystyki i informacje:

1. Bleacher Report - https://bleacherreport.com/ <- Artykuły najświeższych newsów ze świata NBA. (Oprócz tego Bleacher Report zbiera w sobie mnóstwo adresów do zewnętrznych stron z artykułami, które również są przeze mnie pobierane)
2. ESPN - https://www.espn.com/nba <- Statystyki poszczególnych zawodników oraz aktualna tabela ligi
3. FlashScore - https://www.flashscore.pl/koszykowka/usa/nba/ <- Wyniki meczy oraz Terminarz drużyn