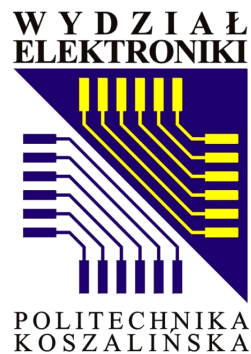


Zastosowanie Programowania Obiektowego
Informatyka III Semestr

Piłka nożna - Katalog (Aplikacja PHP)

Jakub Achtelik, Oliwier Budnik



Politechnika Koszalińska
Koszalin 2023

Spis Treści

1	Koncepcja i funkcjonalność aplikacji	2
1.1	Założenia przyjęte dla tworzenia aplikacji:	2
1.1.1	Środowisko uruchomieniowe:	2
1.2	Założenia przyjęte dla tworzenia aplikacji	3
1.2.1	Cechy	3
1.2.2	Ograniczenia:	3
1.3	Narzędzia programistyczne:	4
1.4	Wykaz funkcjonalności aplikacji	5
1.5	Prototyp	6
2	Diagram przypadków użycia	8
2.1	Rysunek (diagram)	8
2.2	Opis	9
3	Diagram klas	10
3.1	Rysunek (diagram)	10
3.2	Opis przeznaczenia klas	10
3.3	Models	10
3.3.1	Aplikacja	10
3.3.2	BazaDanych	11
3.3.3	ZapytaniaSql	11
3.3.4	FiltrowanieSql	11
3.3.5	OperacjePilkarzy	11
3.3.6	WyswietlaniePilkarzy	11
3.3.7	PobieraczObrazowWikipedia	11
3.4	Controllers	11
3.4.1	Autoryzacja	11
3.4.2	FiltrowanieKontroler	11
3.4.3	KontrolerDanych	12
3.4.4	PilkarzPost	12
3.4.5	ZarzadzaniePilkarzami	12
3.5	Views	12
3.5.1	StronaHtml	12
3.5.2	SzablonHtml	12
3.6	Helpers	12
3.6.1	BazaDanychHelper	12
3.6.2	FormularzHelper	12
3.7	Projekt	13
3.8	FileLoader	13
3.9	plik - index.php	14
4	Projekt Bazy Danych	15
4.1	Model bazy danych - relacyjna SQL	15
4.2	Opis	15
4.2.1	Ustawienie uprawnień dla użytkownika	15
4.3	Rysunek	16

5	Opis działania i obsługi aplikacji	17
5.1	Przewodnik	17
5.1.1	Strona internetowa	17
5.2	Wyświetlanie oraz filtrowanie piłkarzy	17
5.3	Panel Logowania	17
5.4	Panel Administracyjny	18
5.5	Modyfikowanie danych na temat piłkarzy	19
5.5.1	Dodawanie piłkarza	19
5.5.2	Usuwanie piłkarza	20
5.5.3	Edycja piłkarza	21
6	Wnioski	22
6.1	Podsumowanie	22
6.2	Podział pracy	22
6.2.1	Zadania wykonane przez U-20019	22
6.2.2	Zadania wykonane przez U-20041	22
7	Bibliografia - źródła	23

1 Koncepcja i funkcjonalność aplikacji

1.1 Założenia przyjęte dla tworzenia aplikacji:

Rodzaj aplikacji: Aplikacja internetowa (architektura klient-serwer)
Aplikacja uruchamia lokalnie na komputerach oraz zdalnie na serwerze VPS.
Aplikacja jest dostosowana również do urządzeń mobilnych.

Środowisko lokalne: Windows 10, Ubuntu 22.04 LTS

Środowisko produkcyjne (serwer): VPS Linux (Ubuntu 20.04 LTS):

1.1.1 Środowisko uruchomieniowe:

Dla Windows:

- XAMPP 8.2.4 (serwer HTTP Apache, Serwer bazy danych MariaDB, interpreter PHP)

Dla Ubuntu:

- PHP 8.1.2-1ubuntu2.14 Development Server (serwer HTTP + interpreter PHP)
- mysql-server ver 8.0.34-0ubuntu0.22.04.1 (serwer bazy danych MySQL)

Dla VPS:

- Serwer HTTP Apache2
- Serwer MySQL – MariaDB
- Interpreter PHP 8+

1.2 Założenia przyjęte dla tworzenia aplikacji

1.2.1 Cechy

1. Zastosowanie statycznego typowania (zmiennych, funkcji, metod, pól klasy), podobnie jak w językach C/C++, Java, C#. Jest to bardziej przewidywalne i pozwala narzucić określony typ np. zwracanej zmiennej, aby uniknąć wielu błędów. Domyślnie PHP nie wymaga statycznego typowania.
2. Podział projektu na wiele plików według struktury MVC Model-View-Controller (pol. Model-Widok-Kontroler):
 - Model jest pewną reprezentacją problemu bądź logiki aplikacji.
 - Widok opisuje, jak wyświetlić pewną część modelu w ramach interfejsu użytkownika.
 - Kontroler przyjmuje dane wejściowe od użytkownika i reaguje na jego poczynania
3. Logika aplikacji będzie zawarta w sposób obiektowy w klasach, każda klasa to osobny plik.

1.2.2 Ograniczenia:

- PHP jest podatny na pewne rodzaje ataków, takich jak na przykład wstrzykiwanie SQL, dlatego bezpieczeństwo aplikacji nie jest na najwyższym możliwym poziomie i szczegółowa konfiguracja zabezpieczeń nie jest łatwa do wdrożenia w krótkim czasie
- PHP jest językiem interpretowanym dlatego wydajność w stosunku do języków kompilowanych jest niższa
- PHP nie posiada wszystkich elementów obiektowych znanych z innych języków
- Ograniczony czas, przez co nie można zawrzeć wszystkich celów w wzorcowy sposób zgodny w 100% z dokumentacją
- Ograniczenie aktualnej wiedzy, przez co niektóre elementy projektu mogą stanowić wyzwanie
- W naszej aplikacji proces logowania ogranicza się do jednego super użytkownika, którego poświadczenia są statycznie zapisane w pliku **KonfiguracjaApp.php** jako tzw. **plain text** (niezahaszowane hasła przechowywane jako zwykły, łatwy do odczytania i przechwycenia tekst), jest to zła praktyka jednak chcieliśmy uprościć tą część aplikacji ponieważ oddala się ona dość znacząco od pierwotnego tematu, a wymaga dość dużo nadprogramowej pracy.

1.3 Narzędzia programistyczne:

Język: PHP 8+ OOP

Dodatkowe biblioteki: mysqli (łączenie się z bazą danych)

Dodatkowe technologie: HTML, CSS, JavaScript, MySQL, FontAwesone(ikonki)

Visual Studio Code + PHP Code Extensions: IDE (Zintegrowane środowisko programistyczne)

XAMPP: środowisko uruchomieniowe dla Windows

GIT – System Kontroli Wersji

phpMyAdmin - graficzna nakładka na serwer MySQL, ułatwiająca operacje na bazie danych

Brave, Google Chrome – przeglądarka internetowa posiadająca narzędzia Chrome DevTools

Pakiet **make** – automatyzacja poleceń w terminalu

FileZilla – klient FTP

LaTeX - skład tekstu do sprawozdania

PlantUML - narzędzie do tworzenia rysunków i schematów, z poziomu pików tekstowych

Trello – zarządzanie zadaniami w zespole

Figma – Prototypowanie wyglądu aplikacji

GIMP 2.10.34 – Prosta edycja oraz tworzenie grafiki rastrowej

1.4 Wykaz funkcjonalności aplikacji

Interfejs webowy, zarządzanie bazą danych z poziomu przeglądarki internetowej:

- edycje, usuwanie, dodawanie nowego piłkarza,
- sortowanie oraz wyświetlanie zdjęć,
- wyszukiwanie po nazwisku, imieniu itp.
- filtrowanie szczegółowe po np. kraju, pozycji itp.
- logowanie oraz autoryzacja użytkownika przeglądającego aplikację
- dodawanie/edycje, zdjęcia piłkarza

Użytkownik może za pomocą przeglądarki internetowej:

- połączyć się z serwerem na którym hostowana jest aplikacja
- zalogować się do panelu poprzez formularz logowania, uzyskać autoryzację
- Panel umożliwia przeglądanie katalogu piłkarzy w przystępnej formie oraz inne operacje (edycja, usuwaniem, filtrowanie itp.).
- Użytkownik końcowy (klient) nie musi posiadać znajomości obsługi relacyjnej bazy danych aby w intuicyjny sposób zarządzać aplikacją.

1.5 Prototyp

Prototyp graficzny interfejsu użytkownika. Wykonane w programie Figma.



Rysunek 1: Widok strony głównej



Rysunek 2: Widok ekranu logowania

The screenshot shows the 'Dodawanie' (Adding) form. At the top is a header bar with the title 'Piłka Nożna Katalog'. Below it is a navigation bar with four buttons: 'Strona Główna', 'Dodaj', 'Szukaj', and 'Wyloguj'. The main content area is titled 'Dodawanie' and contains six input fields with labels: 'imię', 'nazwisko', 'kraj', 'pozycja', 'wzrost', and a 'dodaj' button at the bottom.

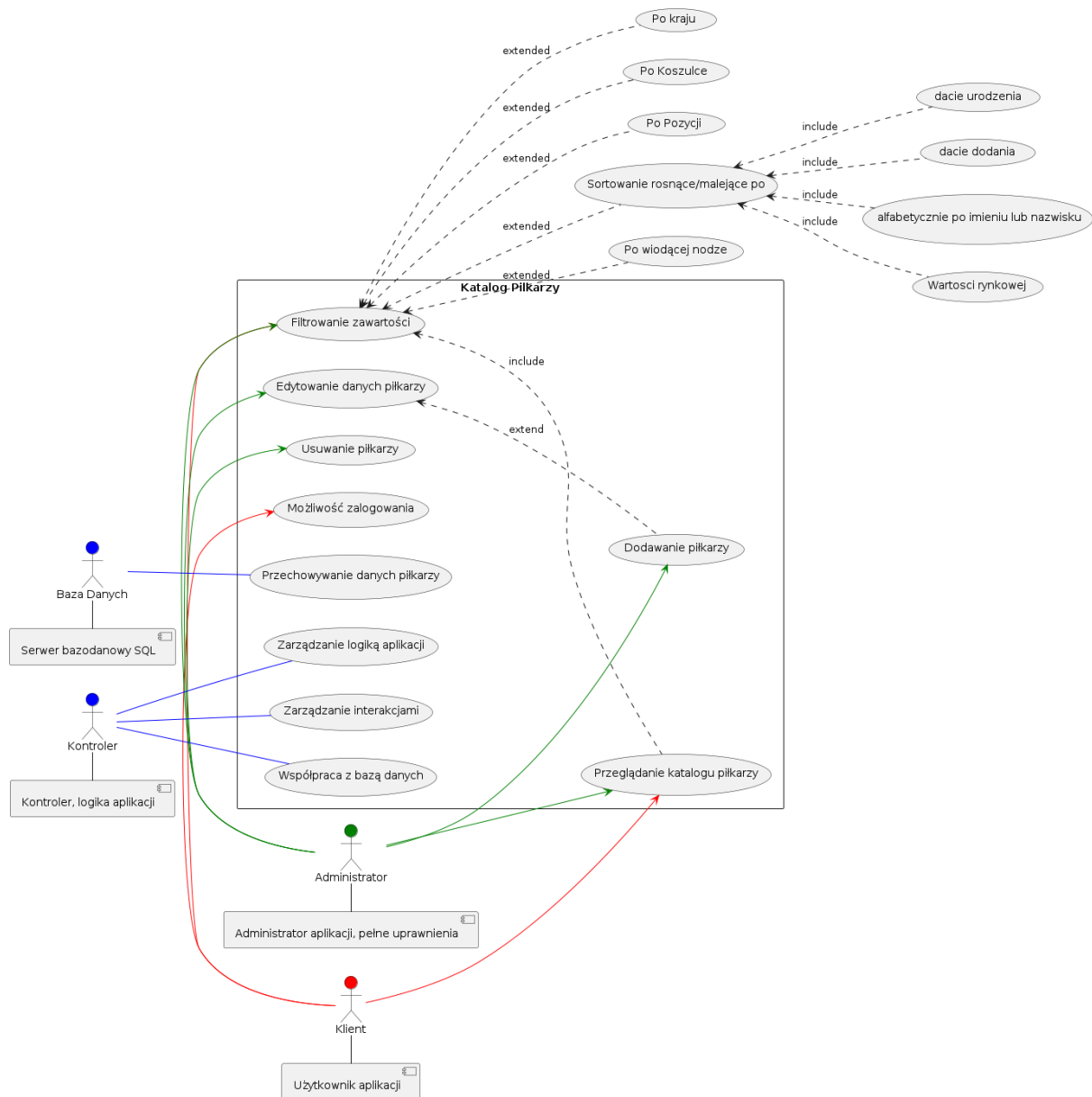
Rysunek 3: Widok strony głównej

The screenshot shows the 'Edycja' (Editing) form. It has the same header and navigation bar as the previous form. The main content area is titled 'Edycja' and contains five input fields with labels: 'Lionel', 'Messi', 'Argentyna', 'napastnik', and '1.62m', followed by an 'edycja' button at the bottom.

Rysunek 4: Widok strony głównej

2 Diagram przypadków użycia

2.1 Rysunek (diagram)



Rysunek 5: Diagram przypadków użycia

2.2 Opis

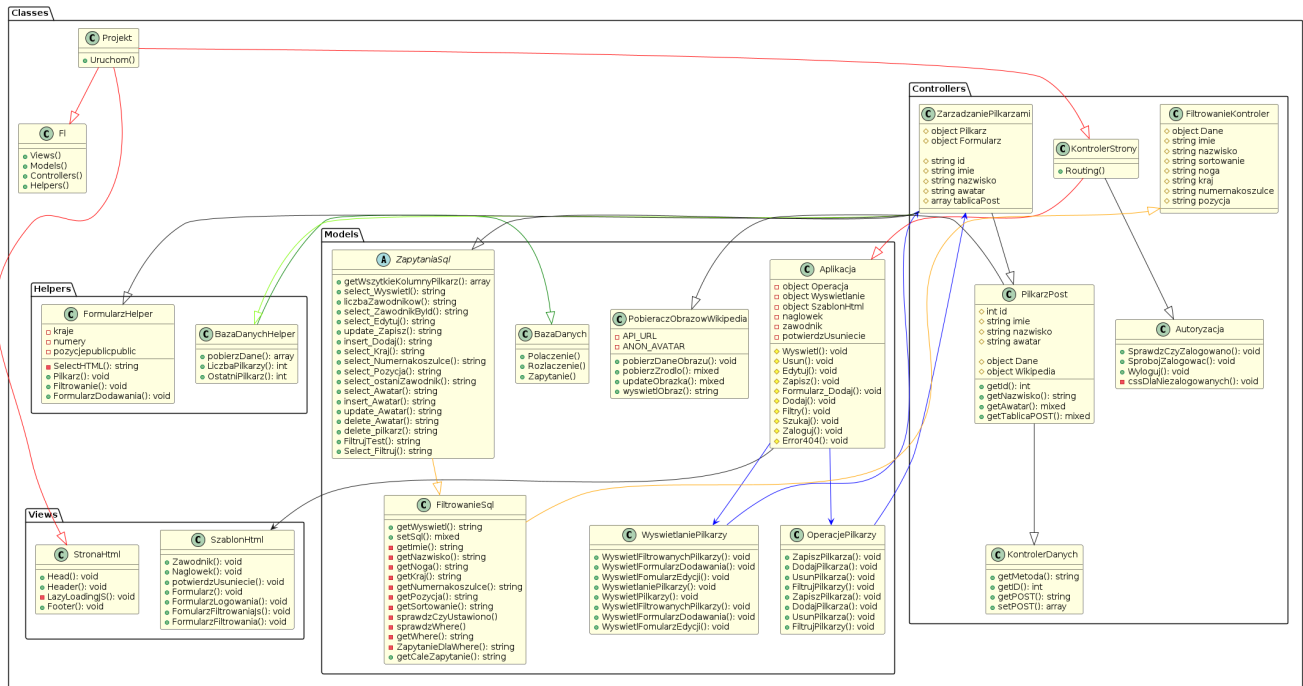
Aktorami osobowymi są: Administrator oraz Klient, są to odbiorcy aplikacji

Aktorami bezosobowymi, abstrakcyjnymi są:

- Kontroler – reprezentuje logikę aplikacji
- BazaDanych – reprezentuje miejsce, gdzie informacje są przechowywane

Podstawowe operacje aktorów:

- Klient -> (Przeglądanie katalogu piłkarzy)
- Klient -> (Filtrowanie zawartości)
- Klient -> (Możliwość zalogowania)
- Administrator -> (Przeglądanie katalogu piłkarzy)
- Administrator -> (Filtrowanie zawartości)
- Administrator -> (Dodawanie piłkarzy)
- Administrator -> (Edytowanie danych piłkarzy)
- Administrator -> (Usuwanie piłkarzy)
- BazaDanych – (Przechowywanie danych piłkarzy)
- Kontroler – (Zarządzanie interakcjami)
- Kontroler – (Współpraca z bazą danych)
- Kontroler – (Zarządzanie logiką aplikacji)



Klasy są segregowane według modelu **MVC** i pomocniczych klas opisanych jako **Helpers**

Stanowi rdzeń całej struktury programu, zarządzając jego ogólnym działaniem, inicjalizacją, oraz kontrolą przepływu danych i interakcji między poszczególnymi komponentami. Jest to kluczowy element, który koordynuje i integruje funkcjonalność innych klas, tworząc spójną aplikację. Skupia się na zarządzaniu operacjami związanymi z piłkarzami w kontekście aplikacji. Posiada różne chronione metody, które współpracują z innymi klasami, takimi jak `OperacjePiłkarzy`, `WyswietlaniePiłkarzy` oraz `SzablonHtml`. Te metody wykonują różne zadania, takie jak wyświetlanie, usuwanie, edytowanie, dodawanie, filtrowanie oraz obsługę błędów w aplikacji związanych z piłkarzami. Klasa ta inicjuje obiekty innych klas w swoim konstruktorze i wykorzystuje je do odpowiedniego przetwarzania i wyświetlania danych w interfejsie użytkownika.

3.3.2 BazaDanych

Odpowiedzialna jest za obsługę połączenia z bazą danych, zarządzanie połączeniem oraz zapewnienie ogólnej komunikacji z nią. Jest kluczowym połączeniem między aplikacją a danymi przechowywanymi w systemie.

3.3.3 ZapytaniaSql

Definiuje i obsługuje generowanie zapytań SQL do bazy danych. Odpowiada za tworzenie struktur zapytań, umożliwiając aplikacji komunikację z bazą w celu pobierania, aktualizacji, usuwania i wstawiania danych.

3.3.4 FiltrowanieSql

Jest odpowiedzialna za filtrowanie danych otrzymanych z bazy danych. Zapewnia mechanizmy filtrowania danych, aby zwracać jedynie pożądane informacje w spójny sposób. Wynikiem końcowym metody *getCaleZapytanie()* jest jedno poprawne zapytanie SQL przekazane do wykonania do bazy danych.

3.3.5 OperacjePilkarzy

Dostarcza zestaw operacji związanych z zarządzaniem danymi dotyczącymi piłkarzy w aplikacji. Obejmuje dodawanie, usuwanie, edycję danych piłkarzy oraz inne operacje z nimi związane.

3.3.6 WyświetlaniePilkarzy

Klasa koncentruje się na prezentacji danych dotyczących piłkarzy. Zapewnia funkcje prezentacji, sortowania, wyświetlania informacji o piłkarzach, dostosowanej do interfejsu użytkownika.

3.3.7 PobieraczObrazowWikipedia

Klasa za pomocą otwartego API Wikipedia, pobiera adres URL do głównego zdjęcia piłkarza z artykułu na Wikipedii. Zdjęcie jest pobierane podczas dodawania nowego piłkarza bądź jego edycji i zapisywana w bazie danych w tabeli **awatar** w postaci linku.

3.4 Controllers

3.4.1 Autoryzacja

Obsługuje mechanizm uwierzytelniania użytkowników. Wykorzystuje wbudowany mechanizm sesji w PHP do uwierzytelniania, zapewniając kontrolę dostępu i identyfikację użytkowników w aplikacji.

3.4.2 FiltrowanieKontroler

Odpowiada za filtrowanie danych otrzymywanych od użytkownika. Jest odpowiedzialna za proces sprawdzania, walidacji i filtrowania danych wejściowych (takich jak parametry GET z adresu URL oraz dane POST z formularzy), aby zapewnić bezpieczeństwo i poprawność przetwarzanych informacji.

3.4.3 KontrolerDanych

Ta klasa ma na celu odbieranie danych od użytkownika z poziomu aplikacji, korzystając z metod GET (parametry z linku) i POST (dane przesyłane z formularzy). Zarządza odbiorem, przetwarzaniem i kontrolą danych, zapewniając ich poprawność i integrację z aplikacją.

3.4.4 PiłkarzPost

Zajmuje się operacjami na danych związanych z piłkarzami, odbierając i przetwarzając informacje przesłane od użytkownika z formularzy POST. Realizuje operacje zapisu danych związanego z piłkarzami do systemu.

3.4.5 ZarządzaniePiłkarzami

Jest klasą, która wykorzystuje klasę pomocniczą BazaDanychHelper do operacji na bazie danych w kontekście informacji o piłkarzach. Ta klasa definiuje podstawowe funkcje i metody obsługi informacji o piłkarzach, wykorzystując metody HTTP - GET i POST do pobierania oraz przesyłania danych.

3.5 Views

Views (pol. Widoki), ta sekcja zawiera metody, które reprezentują wartość graficzną aplikacji w HTML, CSS i JavaScript. Odpowiada za poprawne wyświetlanie informacji, które zostały uzyskane z bazy danych.

3.5.1 StronaHtml

Zawiera szkielet strony HTML, taki jak sekcja <head> <body> <footer> czy <header>. Dzięki temu rozwiązaniu można ustawić Podstawowe dane takie jak tytuł strony, autorów, w kilku miejscach na stronie, wczytywana z pliku konfiguracyjnego **KonfiguracjaApp.php**

3.5.2 SzablonHtml

Klasa zawiera komponenty HTML, które służą do wyświetlania elementów, takich jak card (karta z informacją o piłkarzu), formularze, panel logowania.

3.6 Helpers

3.6.1 BazaDanychHelper

Służy do udostępniania metod wspomagających operacje bazodanowe, takie jak łączenie się z bazą danych, wykonywanie zapytań czy inne operacje związane z bazą danych.

3.6.2 FormularzHelper

Zawiera metody pomocnicze związane z tworzeniem, walidacją czy obsługą formularzy w aplikacji.

3.7 Projekt

Klasa ta jest ostatecznym elementem programu, posiadającym metodę *Uruchom()*, która integruje Routing, StronęHtml oraz klasę Aplikacja w celu właściwego uruchomienia programu z zachowaniem odpowiedniej kolejności działań. Metoda *Uruchom()* łączy te elementy w logiczną sekwencję, umożliwiając kompleksowe działanie aplikacji poprzez współpracę wymienionych składników w spójny sposób.

```

1  <?php
2  // klasa do uruchamiania projektu
3  namespace Pilkanozna;
4
5  use Pilkanozna\Controllers\KontrolerStrony;
6  use Pilkanozna\Views\StronaHtml;
7
8  final class Projekt
9  {
10     public static function Uruchom(): void
11     {
12         $Aplikacja = new KontrolerStrony;
13         $Strona = new StronaHtml;
14
15
16         $Strona->Head();
17         $Strona->Header();
18         $Aplikacja->Routing();
19         $Strona->Footer();
20     }
21 }
```

3.8 FileLoader

Klasa ma jedynie jedno przeznaczenie: załączenie plików z klasami do projektu. Jej rola polega na importowaniu (łączeniu) różnych plików zawierających klasy, co umożliwia poprawne funkcjonowanie projektu poprzez dostęp do niezbędnych klas i ich metod.

```

1  <?php
2
3  class Fl
4  {
5     public static function Views($name) {
6         require_once "../classes/Views/$name.php";
7     }
8
9
10    public static function Models($name) {
11        require_once "../classes/Models/$name.php";
12    }
13
14
15    public static function Controllers($name) {
16        require_once "../classes/Controllers/$name.php";
17    }
18 }
```

```

17     }
18
19     public static function Helpers($name) {
20         require_once "../classes/Helpers/$name.php";
21     }
22 }

```

3.9 plik - index.php

Ten plik inicjuje metodę *Uruchom()* oraz wykorzystuje klasę *FileLoader* (skrót *Fl*) do załączenia plików. Jego głównym zadaniem jest uruchomienie metody *Uruchom()* oraz korzystanie z klasy *FileLoader* w celu załadowania plików do projektu.

```

1  <?php
2  use Pilkanozna\Projekt;
3
4  include_once '../classes/FileLoader.php';
5  include_once '../classes/Projekt.php';
6
7  // kolejnosc ladowania klas - nie zmieniac
8
9  Fl::Controllers("PilkarzPost");
10 Fl::Models("BazaDanych");
11 Fl::Helpers("BazaDanychHelper");
12 Fl::Controllers("FiltrowanieKontroler");
13 Fl::Models("FiltrowanieSql");
14 Fl::Models("ZapytaniaSql");
15 Fl::Helpers("FormularzHelper");
16
17 Fl::Controllers("ZarządzaniePilkarzami");
18 Fl::Models("WyświetlaniePilkarzy");
19 Fl::Models("OperacjePilkarzy");
20
21 Fl::Models("Aplikacja");
22
23 Fl::Controllers("KontrolerDanych");
24 Fl::Controllers("Autoryzacja");
25 Fl::Controllers("KontrolerStrony");
26
27 Fl::Models("PobieraczObrazowWikipedia");
28 Fl::Views("SzablonHtml");
29 Fl::Views("StronaHtml");
30
31
32 Projekt::Uruchom();

```

4 Projekt Bazy Danych

4.1 Model bazy danych - relacyjna SQL

Baza danych jest relacyjna oraz obsługiwana jest na serwerze SQL.

4.2 Opis

Kod SQL definiuje strukturę bazy danych dla systemu zarządzania informacjami o piłkarzach.

1. Tworzenie bazy danych:

Kod rozpoczyna od próby usunięcia bazy danych "pilkanozna"(jeśli istnieje) i następnie tworzy nową bazę o tej samej nazwie. Następnie wybiera tę nowo utworzoną bazę jako aktywną dla dalszych operacji. Definicje tabel:

2. Określa strukturę różnych tabel:

- Tabela "piłkarz"przechowuje informacje o piłkarzach, takie jak imię, nazwisko, dane osobowe, umiejętności piłkarskie i pochodzenie.
- Tabela "awatar"prawdopodobnie zawiera linki do obrazków reprezentujących piłkarzy.
- Tabela "krajpiłkarza"przechowuje informacje o krajach, do których należą piłkarze.
- Tabela "numernakoszulce"odnosi się do numerów na koszulkach piłkarzy.
- Tabela "pozycja"zawiera różne pozycje, na których piłkarze mogą grać.

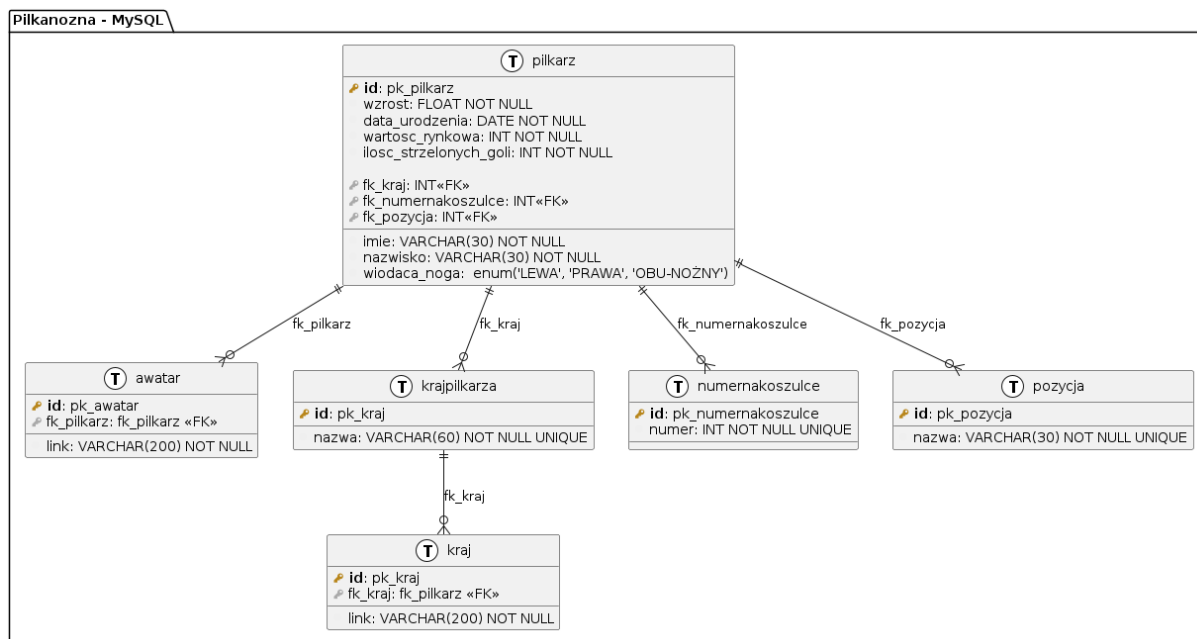
3. Wstawianie danych:

Dodaje przykładowe dane do tabeli "pozycja", "numernakoszulce"i "krajpiłkarza".

4.2.1 Ustawienie uprawnień dla użytkownika

```
1 CREATE USER 'projekt'@'localhost' IDENTIFIED BY 'Pracownia107!';
2 GRANT ALL PRIVILEGES ON pilkanozna.* TO 'projekt'@'localhost';
```

4.3 Rysunek



Rysunek 7: Diagram tabel bazy danych pilkanozna

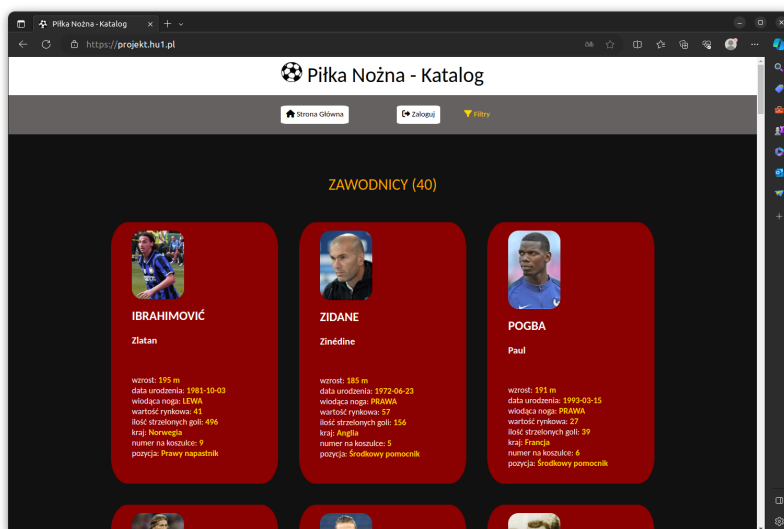
5 Opis działania i obsługi aplikacji

5.1 Przewodnik

5.1.1 Strona internetowa

Aplikacja znajduje się na zewnętrznym serwerze VPS i można ją przetestować bez uruchamiania jej na swoim komputerze, dostępna jest pod adresem:

- <https://projekt.hu1.pl/>



Rysunek 8: Strona Główna

5.2 Wyświetlanie oraz filtrowanie piłkarzy

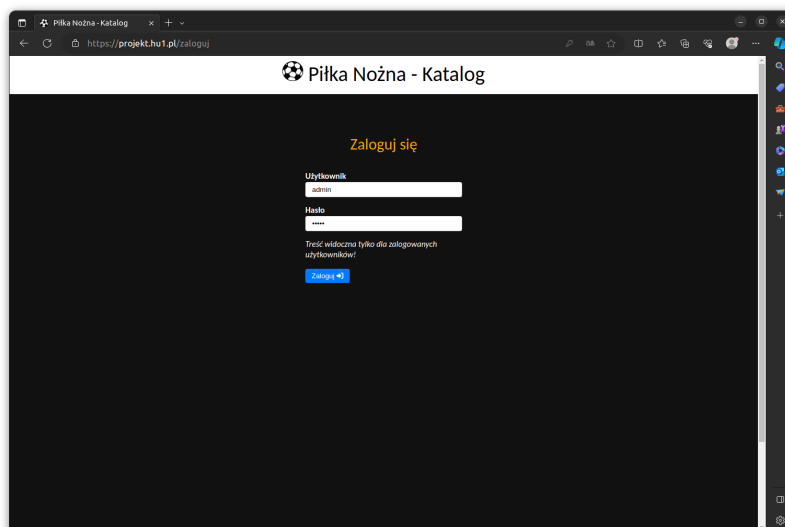
5.3 Panel Logowania

Panel logowania dostępny jest pod adresem:

- <https://projekt.hu1.pl/zaloguj>

Dane do logowania:

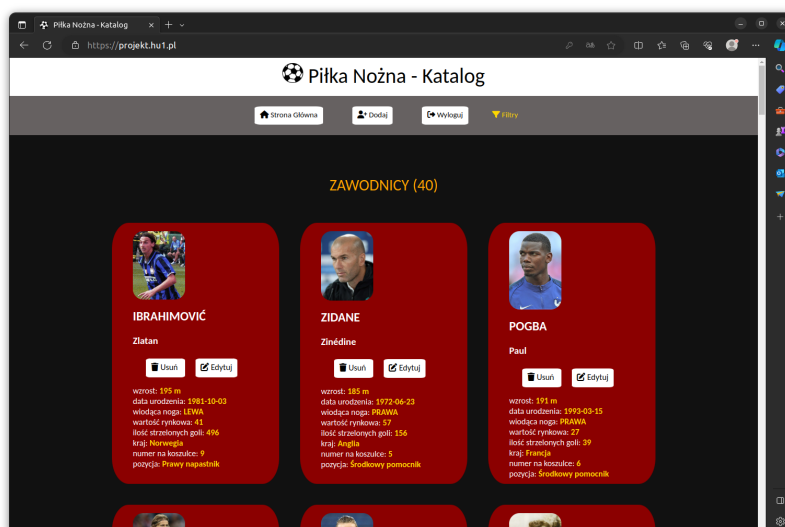
- Użytkownik: **admin**
- Hasło: **admin**



Rysunek 9: Panel logowania

5.4 Panel Administracyjny

Jako administrator użytkownik ma podniesione uprawnienia i dodatkową zakładkę *Dodaj* oraz przycisk *Edytuj* lub *Usuń* nad zdjęciem piłkarza.

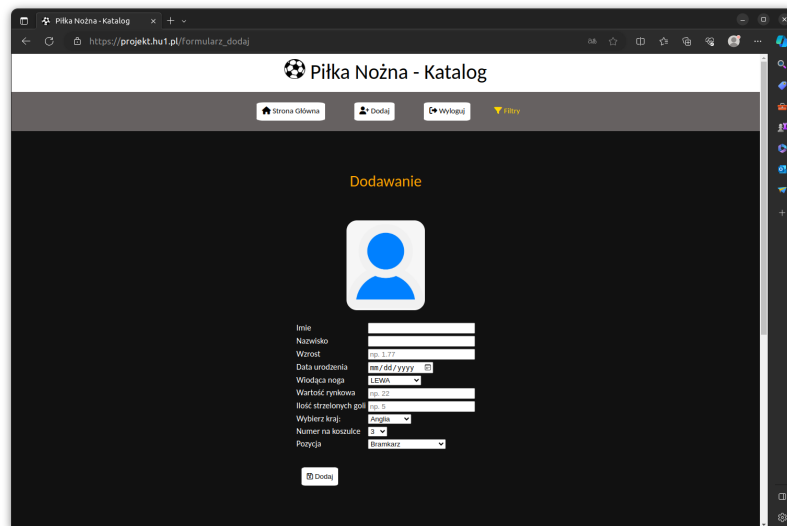


Rysunek 10: Panel Administracyjny

5.5 Modyfikowanie danych na temat piłkarzy

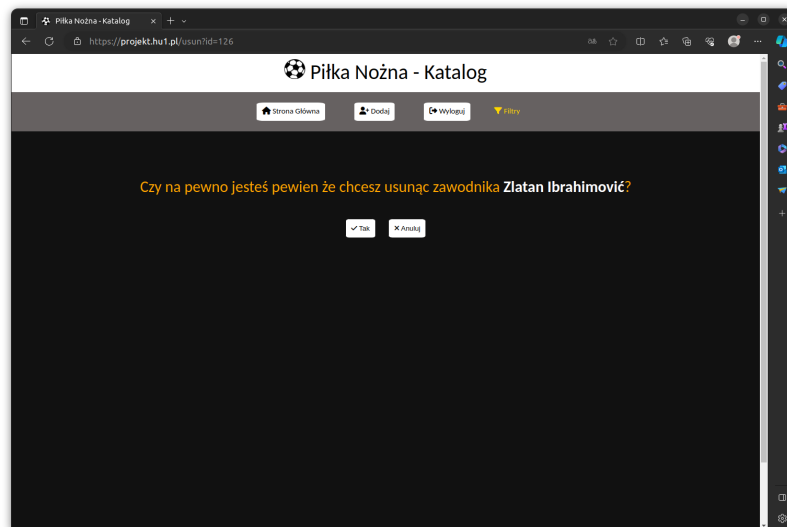
5.5.1 Dodawanie piłkarza

Na podstawie **Imienia** oraz **Nazwiska** zdjęcie piłkarza jest pobierane z serwisu Wikipedia, jeżeli piłkarz ma tam swój artykuł wraz ze zdjęciem (w 90 % przypadkach posiada), w innym wypadku ustawia domyślne zdjęcie anonimowego użytkownika.



Rysunek 11: Formularz, który umożliwia dodanie nowego Piłkarza

5.5.2 Usuwanie piłkarza



Rysunek 12: Potwierdzenie usunięcia Piłkarza

5.5.3 Edycja piłkarza

The screenshot shows a web browser window with the address `https://projekt.hu1.pl/edytuj?id=64`. The page title is "Piłka Nożna - Katalog". The navigation bar includes links for "Strona Główna", "Dodaj", "Wyloguj", and "Filmy". The main content area is titled "EDYCJA" and features a profile picture of a player. Below the picture is a form with the following fields:

Imię	<input type="text" value="Robert"/>
Nazwisko	<input type="text" value="Lewandowski"/>
Wzrost	<input type="text" value="1.88"/>
Data urodzenia	<input type="text" value="08/21/1988"/>
Władająca noga	<input type="text" value="PRAWA"/>
Wartość rynkowa	<input type="text" value="50"/>
Ilość strzałów z gry	<input type="text" value="104"/>
Wybierz kraj	<input type="text" value="Polska"/>
Numer na koszulce	<input type="text" value="9"/>
Pozycja	<input type="text" value="Strzałowy napastnik"/>

At the bottom of the form is a button labeled "Zapisz".

Rysunek 13: Panel służący do edycji informacji na temat piłkarza

6 Wnioski

6.1 Podsumowanie

6.2 Podział pracy

U-20019 - Jakub Achtelik

U-20041 - Oliwier Budnik

6.2.1 Zadania wykonane przez U-20019

- Zaprojektowanie oraz wykonanie bazy danych
- Konstrukcja zapytań oraz klas, które służą do łączenia z bazą danych
- Konfiguracja usług na serwerze VPS
- Skład tekstu sprawozdania w LaTeX
- Implementacja połączenia API Wikipedia z naszą aplikacją
- Proste grafiki w programie GIMP
- Pobieranie danych od użytkownika za pomocą POST oraz GET

6.2.2 Zadania wykonane przez U-20041

- Prototyp wyglądu aplikacji
- Implementacja autoryzacji oraz formularz logowania
- Formularz edycji, dodawania i przekazanie tego do jednej tablicy
- Wyświetlanie zwróconych wyników z bazy danych w formie Szablonów HTML
- Szablon strony HTML oraz dostosowanie wyglądu w CSS
- Routing strony, czyli ustawienie klasy, która na podstawie /linku wykonuje określoną operację
- Implementacja logiki klasy Aplikacja

7 Bibliografia - źródła

Literatura

- [1] php.net - Oficjalna Dokumentacja PHP
<https://www.php.net/docs.php>
- [2] Medium - Informacje na temat MVC w PHP
<https://medium.com/@iamjoestack/how-to-build-a-custom-php-mvc-framework-e5a23da8f73d>
- [3] Konfiguracja Serwera SQL - Ubuntu VPS
<https://ubuntu.com/server/docs/databases-mysql>
- [4] Konfiguracja Serwera HTTP - Apache 2
<https://httpd.apache.org/docs/2.4/>
- [5] Tworzenie diagramów i schematów - PlantUML
<https://plantuml.com/>
- [6] Pobieranie obrazów z API Wikipedia
<https://stackoverflow.com/questions/8363531/accessing-main-picture-of-wikipedia-page-by-api>
- [7] Obsługa programu GIMP
<https://www.gimp.org/docs/>