

Autorzy:

Gabriela Ławrynowicz

Michał Lipkowski

SplashTrainer - Generator planów treningowych dla pływaków

SplashTrainer to nowoczesna aplikacja internetowa zaprojektowana z myślą o entuzjastach pływania, którzy pragną monitorować swoje postępy, planować treningi oraz osiągać cele sportowe w zorganizowany sposób. Aplikacja oferuje zestaw funkcjonalności, które umożliwiają użytkownikom personalizację planów treningowych, tworzenie własnych ćwiczeń oraz analizowanie swoich wyników.

Celem aplikacji jest wspieranie użytkowników w rozwijaniu umiejętności pływackich oraz budowanie społeczności osób aktywnie zaangażowanych w ten sport. **SplashTrainer** zapewnia intuicyjny interfejs użytkownika, bogaty katalog ćwiczeń oraz narzędzia analityczne, które pomagają w realizacji indywidualnych celów.

Dzięki zaawansowanemu systemowi zarządzania użytkownikami, aplikacja oferuje personalizację treści – każdy użytkownik ma dostęp do swoich prywatnych planów treningowych oraz może dodawać własne ćwiczenia. Jednocześnie globalny katalog ćwiczeń jest dostępny dla wszystkich użytkowników, co pozwala na łatwe inspirowanie się istniejącymi zasobami. Dodatkowe funkcje obejmują możliwość eksportu planu do pliku PDF i wizualizację postępu w szczegółach każdego treningu.

1. Specyfikacja wykorzystanych technologii

Technologie backendowe:

- **Framework:** .NET 8.0 (ASP.NET Core)
- **Język programowania:** C#
- **ORM:** Entity Framework Core (z obsługą SQLite i SQL Server)

Baza danych:

- **SQLite:** Używana jako baza danych w projekcie.

Biblioteki i pakiety NuGet:

1. **Microsoft.AspNetCore.Diagnostics.EntityFrameworkCore:** Narzędzia diagnostyczne dla Entity Framework Core.
2. **Microsoft.AspNetCore.Identity.EntityFrameworkCore:** Obsługa systemu tożsamości w aplikacjach ASP.NET Core.
3. **Microsoft.AspNetCore.Identity.UI:** Interfejs użytkownika dla funkcjonalności tożsamości.
4. **Microsoft.EntityFrameworkCore:** Główna biblioteka ORM.
5. **Microsoft.EntityFrameworkCore.Design:** Narzędzia projektowe dla Entity Framework.
6. **Microsoft.EntityFrameworkCore.Sqlite:** Provider SQLite dla Entity Framework.
7. **Microsoft.EntityFrameworkCore.Tools:** Narzędzia CLI do migracji i pracy z bazą danych.
8. **Microsoft.Extensions.Logging.Console:** Logowanie do konsoli w aplikacjach ASP.NET Core.
9. **jQuery.Validation:** Walidacja formularzy po stronie klienta za pomocą jQuery.
10. **jQuery.Validation.Unobtrusive:** Dyskretna walidacja formularzy w ASP.NET.
11. **PdfSharpCore:** Core'owa wersja PdfSharp obsługująca platformy .NET Core i .NET 5+.

2. Uruchomienie projektu :

1. Uzyskanie dostępu do kodu projektu

Otwórz link do publicznego repozytorium na GitHub.

Na stronie repozytorium znajdź zielony przycisk "Code".

Skopiuj link HTTPS lub SSH do repozytorium (zalecane HTTPS, jeśli nie korzystasz z konfiguracji SSH).

2. Sklonowanie repozytorium na komputer

Otwórz terminal lub narzędzie Git Bash.

Przejdź do folderu, w którym chcesz umieścić projekt.

Wykonaj polecenie: `git clone <link_do_repozytorium>`

Przykład : `git clone https://github.com/nazwa_uzytkownika/nazwa_repozytorium.git`

3. Otworzenie projektu w IDE

Otwórz Visual Studio 2022 lub inną kompatybilną wersję IDE.

Wybierz opcję "Open a project or solution" i wskaż plik .csproj lub .sln znajdujący się w głównym katalogu projektu.

4. Przygotowanie bazy danych

Projekt domyślnie korzysta z SQLite.

Upewnij się, że masz zainstalowane narzędzia Entity Framework Core w Visual Studio:

Wejdź w Tools > NuGet Package Manager > Manage NuGet Packages for Solution i sprawdź obecność następujących pakietów:

Microsoft.EntityFrameworkCore

Microsoft.EntityFrameworkCore.Sqlite

Microsoft.EntityFrameworkCore.Tools.

W Menedżerze pakietów NuGet w Visual Studio wykonaj migrację bazy danych:

Polecenie: Update-Database

5. Uruchomienie aplikacji

W Visual Studio wybierz plik startowy (z rozszerzeniem .sln)

Uruchom aplikację, klikając zielony przycisk Run lub używając skrótu klawiszowego F5.

Uwagi:

Jeśli wystąpią problemy z migracją bazy danych:

Zainstaluj narzędzia EF CLI: `dotnet tool install --global dotnet-ef` (w terminalu)

Następnie powtórz proces migracji.

3. Struktura projektu

Projekt **SplashTrainer** został stworzony w technologii ASP.NET Core 8.0 z wykorzystaniem Entity Framework Core i SQLite jako bazy danych. Struktura projektu została podzielona zgodnie z zasadami MVC (Model-View-Controller), aby zachować czytelność i łatwość w zarządzaniu kodem. Poniżej znajdują się główne foldery i ich przeznaczenie:

1. Controlery

- Zawiera wszystkie kontrolery, które obsługują logikę aplikacji. Każdy kontroler jest odpowiedzialny za przetwarzanie żądań użytkownika, zarządzanie danymi oraz przekazywanie ich do widoków.
- Kontrolery: `AccountController`, `HomeController`, `SwimmingPlanController`, `SwimmingExerciseController`, `ActivityLogController`

2. Modele

- Zawiera modele danych, które definiują strukturę danych przechowywanych w bazie. Modele są używane w warstwie logiki aplikacji i do komunikacji z Entity Framework Core.
- Modele: `SwimmingPlan`, `SwimmingExercise`, `PlanExercise`, `ActivityLog`, `ErrorViewModel`, `GeneratePlanViewModel`.

3. Widoki

- Zawiera wszystkie widoki aplikacji. Widoki to szablony HTML renderowane na podstawie danych przekazanych przez kontrolery. Są one podzielone na foldery odpowiadające nazwom kontrolerów (np. `Home`, `SwimmingPlan`).
- Wszystkie widoki są wyszczególnione niżej.

4. Data

- Ten folder zawiera pliki związane z konfiguracją bazy danych oraz migracjami.
- Kluczowy plik: `ApplicationDbContext`, który odpowiada za połączenie z bazą danych oraz definiowanie `DbSet`-ów dla modeli i danych do bazy.

5. wwwroot

- Folder publiczny, w którym przechowywane są pliki statyczne, takie jak CSS, JavaScript i obrazy. Te pliki są bezpośrednio dostępne z poziomu przeglądarki.
- Pliki: `styles.css`, `site.js`, `img`.

6. Migrations

- Zawiera pliki migracji wygenerowane przez Entity Framework Core. Migracje są używane do tworzenia i aktualizacji struktury bazy danych.

7. Properties

- Zawiera plik konfiguracyjny projektu `launchSettings.json`, który określa sposób uruchamiania aplikacji (np. adres URL lokalnego serwera).

8. Program.cs

- Główny plik startowy aplikacji. Odpowiada za konfigurację całego projektu, w tym ustawienie routingu, usług Dependency Injection i bazy danych.

9. appsettings.json

- Plik konfiguracyjny aplikacji. Zawiera ustawienia, takie jak połączenie z bazą danych, opcje logowania czy konfiguracja zewnętrznych usług.

Jak to działa w praktyce:

- Użytkownik przesyła żądanie (np. poprzez kliknięcie przycisku w widoku).
- Żądanie trafia do odpowiedniego kontrolera.
- Kontroler przetwarza logikę i pobiera dane z bazy (poprzez modele).
- Dane są przekazywane do widoku, który generuje odpowiedź w postaci HTML.

4. Modele:

ActivityLog

Model `ActivityLog` służy do śledzenia i rejestrowania aktywności użytkownika związanych z planami treningowymi. Jego celem jest zapewnienie historii zdarzeń, co pozwala użytkownikowi i administratorowi monitorować zmiany, działania i postępy.

Pola:

- **Id** (*int*): Klucz główny w tabeli.
- **SwimmingPlanId** (*int*): Klucz obcy, wskazujący na powiązany plan treningowy.
- **SwimmingPlan** (*SwimmingPlan*): Obiekt relacyjny, pozwalający na dostęp do pełnych danych planu.
- **CreatedDate** (*DateTime*): Data i czas zarejestrowania aktywności.

Relacje:

- Powiązany z modelem `SwimmingPlan` przez klucz obcy `SwimmingPlanId`.

ErrorViewModel

Model ErrorViewModel jest używany głównie do obsługi błędów w aplikacji. Został zaprojektowany w celu wyświetlania szczegółowych informacji w przypadku problemów technicznych lub błędów użytkownika.

Pola:

- **RequestId** (*string?*): Identyfikator żądania HTTP, pomocny przy śledzeniu błędów.
 - **ShowRequestId** (*bool*): Wartość logiczna wskazująca, czy należy wyświetlić identyfikator żądania.
-

GeneratePlanViewModel

Model GeneratePlanViewModel to kluczowy element odpowiedzialny za przesyłanie danych wejściowych od użytkownika podczas tworzenia nowego planu treningowego. Jego zadaniem jest zapewnienie walidacji i struktury danych.

Pola:

- **Name** (*string*): Nazwa planu treningowego. *(Wymagane, maksymalnie 100 znaków)*.
- **Goal** (*string*): Cel treningu (np. "Koordynacja", "Wytrzymałość"). *(Wymagane, wyrażenie regularne definiujące dozwolone wartości)*.
- **Level** (*string*): Poziom trudności (np. "Początkujący", "Zaawansowany"). *(Wymagane)*.
- **Style** (*string*): Styl pływacki (np. "Grzbiet", "Kraul"). *(Wymagane)*.
- **MinDistancePerTraining** (*int*): Minimalny dystans przypisany do jednego treningu. *(Wymagane, walidowane wyrażeniem regularnym)*.
- **WeeklyTrainingDays** (*int*): Liczba dni treningowych w tygodniu. *(Wymagane, zakres 1–7)*.
- **PlanDurationWeeks** (*int*): Długość planu w tygodniach. *(Wymagane, zakres 1–12)*.
- **TrainingDays** (*List<DayOfWeek>*): Lista wybranych dni tygodnia.

Walidacje:

- Wyrażenia regularne i ograniczenia zakresów zapewniają poprawność danych wprowadzonych przez użytkownika.
-

PlanExercise

Model PlanExercise stanowi połączenie między planami treningowymi a ćwiczeniami. Każdy obiekt reprezentuje jedno ćwiczenie przypisane do konkretnego planu.

Pola:

- **Id** (*int*): Klucz główny.
- **SwimmingPlanId** (*int*): Klucz obcy do modelu SwimmingPlan.
- **SwimmingPlan** (*SwimmingPlan*): Obiekt relacyjny umożliwiający dostęp do danych planu.
- **SwimmingExerciseId** (*int*): Klucz obcy do modelu SwimmingExercise.
- **SwimmingExercise** (*SwimmingExercise*): Obiekt relacyjny umożliwiający dostęp do danych ćwiczenia.
- **ExerciseDate** (*string*): Data przypisana do konkretnego treningu (np. "Trening 1").
- **ExerciseTime** (*TimeSpan?*): Czas wykonania danego ćwiczenia.
- **Distance** (*int*): Dystans przypisany do danego ćwiczenia.
- **IsCompleted** (*bool*): Wskazuje, czy ćwiczenie zostało ukończone (domyślnie false).

Relacje:

- Łączy modele SwimmingPlan i SwimmingExercise.
-

SwimmingExercise

Model SwimmingExercise reprezentuje wszystkie dostępne ćwiczenia, które mogą być przypisane do planów treningowych.

Pola:

- **Id** (*int*): Klucz główny.
 - **IsDefault** (*bool*): Flaga wskazująca, czy ćwiczenie jest domyślne.
 - **Name** (*string*): Nazwa ćwiczenia. (*Wymagane*).
 - **Distance** (*int*): Dystans przypisany do ćwiczenia. (*Wymagane, wielokrotność 25*).
 - **Category** (*string*): Kategoria ćwiczenia (np. "Koordynacja", "Wytrzymałość"). (*Wymagane*).
 - **Description** (*string*): Opis ćwiczenia. (*Wymagane*).
 - **Level** (*string*): Poziom trudności (np. "Początkujący"). (*Wymagane*).
 - **Style** (*string*): Styl pływacki (np. "Grzbiet"). (*Wymagane*).
-

SwimmingPlan

Model SwimmingPlan reprezentuje cały plan treningowy użytkownika.

Pola:

- **Id** (*int*): Klucz główny.
- **Name** (*string*): Nazwa planu. (*Wymagane*).
- **UserId** (*string*): Identyfikator użytkownika, który utworzył plan.
- **DateCreated** (*DateTime*): Data utworzenia planu.
- **TotalDistance** (*int*): Całkowity dystans w planie.
- **Exercises** (*ICollection<PlanExercise>*): Lista ćwiczeń wchodzących w skład planu.
- **ActivityLogs** (*ICollection<ActivityLog>*): Lista aktywności związanych z planem.
- **Goal** (*string*): Cel treningu (np. "Koordynacja"). (*Wymagane*).
- **Level** (*string*): Poziom trudności (np. "Początkujący"). (*Wymagane*).
- **Style** (*string*): Styl pływacki (np. "Grzbiet"). (*Wymagane*).
- **TrainingDays** (*ICollection<DayOfWeek>*): Lista dni treningowych w ramach planu.

Relacje:

- Powiązany z PlanExercise i ActivityLog.

5. Kontrolery

AccountController

AccountController jest odpowiedzialny za zarządzanie procesami rejestracji, logowania oraz wylogowania użytkowników w aplikacji. Wykorzystuje on mechanizmy dostarczane przez ASP.NET Identity do obsługi użytkowników oraz ich autoryzacji.

Konstruktor

- **Parametry:**
 - UserManager<IdentityUser> - zarządza operacjami na użytkownikach, takimi jak tworzenie użytkownika, resetowanie hasła, itp.
 - SignInManager<IdentityUser> - odpowiada za logowanie i wylogowywanie użytkowników.
- Konstruktor inicjalizuje te komponenty w celu ich wykorzystania w metodach kontrolera.

Metody

1. **Register()** - GET

- **Opis:** Wyświetla widok rejestracji.
 - **Metoda HTTP:** GET
 - **Zwracany widok:** Register
-

2. **Register(string email, string password)** - POST

- **Opis:** Rejestruje nowego użytkownika w systemie na podstawie podanego adresu e-mail i hasła.
 - **Metoda HTTP:** POST
 - **Parametry:**
 - email - adres e-mail użytkownika, który jednocześnie służy jako nazwa użytkownika.
 - password - hasło użytkownika.
 - **Działanie:**
 1. Tworzy nowego użytkownika za pomocą UserManager.
 2. Jeśli proces rejestracji się powiedzie:
 - Loguje nowego użytkownika za pomocą SignInManager.
 - Przekierowuje do strony głównej.
 3. Jeśli rejestracja się nie powiedzie:
 - Dodaje błędy walidacji do ModelState.
 - Wyświetla widok rejestracji.
 - **Zwracany widok:**
 - W przypadku sukcesu: Przekierowanie do Index w kontrolerze Home.
 - W przypadku błędów: Register.
-

3. **Login()** - GET

- **Opis:** Wyświetla widok logowania.
 - **Metoda HTTP:** GET
 - **Zwracany widok:** Login
-

4. **Login(string email, string password)** - POST

- **Opis:** Loguje użytkownika na podstawie podanego adresu e-mail i hasła.

- **Metoda HTTP:** POST
 - **Parametry:**
 - email - adres e-mail użytkownika (nazwa użytkownika).
 - password - hasło użytkownika.
 - **Działanie:**
 1. Próbuje zalogować użytkownika za pomocą SignInManager.
 2. Jeśli logowanie się powiedzie:
 - Przekierowuje użytkownika do strony głównej.
 3. Jeśli logowanie się nie powiedzie:
 - Dodaje komunikat o błędzie do ModelState.
 - Wyświetla widok logowania.
 - **Zwracany widok:**
 - W przypadku sukcesu: Przekierowanie do Index w kontrolerze Home.
 - W przypadku błędów: Login.
-

5. Logout() - POST

- **Opis:** Wylogowuje obecnie zalogowanego użytkownika.
 - **Metoda HTTP:** POST
 - **Działanie:**
 1. Wylogowuje użytkownika za pomocą SignInManager.
 2. Przekierowuje do strony głównej.
 - **Zwracany widok:** Przekierowanie do Index w kontrolerze Home.
-

Podsumowanie

Kontroler korzysta z ASP.NET Identity, aby zapewnić funkcjonalność rejestracji, logowania i wylogowania. Jego działanie opiera się na dwóch głównych komponentach:

- UserManager<IdentityUser>: Zarządza operacjami na użytkownikach.
- SignInManager<IdentityUser>: Obsługuje logowanie i wylogowywanie.

Każda metoda jest odpowiedzialna za jeden aspekt autoryzacji i korzysta z dedykowanych widoków (Register, Login). Dzięki temu AccountController jest dobrze oddzielony od innych funkcji aplikacji i pełni wyłącznie rolę zarządzania tożsamością użytkownika.

ActivityLogController

ActivityLogController jest odpowiedzialny za zarządzanie dziennikiem aktywności użytkownika, szczegóły planów treningowych oraz generowanie raportów w formacie PDF. Kontroler pozwala użytkownikowi przeglądać swoje plany, edytować ich nazwy, usuwać je oraz śledzić postępy w ramach ukończonych treningów.

Konstruktor

- **Parametry:**
 - ApplicationDbContext context: Dostarcza kontekst bazy danych do pracy z encjami aplikacji.
 - Konstruktor inicjalizuje kontekst bazy danych.
-

Metody

1. Index() - GET

- **Opis:** Wyświetla listę planów treningowych zalogowanego użytkownika.
 - **Działanie:**
 1. Pobiera UserId zalogowanego użytkownika.
 2. Filtruje plany treningowe przypisane do użytkownika.
 3. Dołącza powiązane ćwiczenia.
 - **Zwracany widok:** Lista planów treningowych użytkownika.
-

2. EditPlanName(int id, string planName) - POST

- **Opis:** Pozwala edytować nazwę wybranego planu treningowego.
- **Parametry:**
 - id: Identyfikator planu.
 - planName: Nowa nazwa planu.
- **Działanie:**
 1. Wyszukuje plan o podanym id.
 2. Jeśli plan istnieje, aktualizuje jego nazwę.
 3. Zapisuje zmiany w bazie danych.
- **Zwracana wartość:** JSON z informacją o powodzeniu operacji.

3. Delete(int id) - DELETE

- **Opis:** Usuwa plan treningowy.
 - **Parametry:**
 - id: Identyfikator planu.
 - **Działanie:**
 1. Wyszukuje plan o podanym id.
 2. Jeśli plan istnieje, usuwa go z bazy danych.
 - **Zwracana wartość:** JSON z informacją o powodzeniu operacji.
-

4. Details(int id) - GET

- **Opis:** Wyświetla szczegóły wybranego planu treningowego.
 - **Parametry:**
 - id: Identyfikator planu.
 - **Działanie:**
 1. Pobiera plan treningowy z bazy danych wraz z powiązanymi ćwiczeniami.
 2. Oblicza liczbę ukończonych i wszystkich treningów.
 3. Oblicza procent ukończenia planu.
 4. Grupuje ćwiczenia według sesji treningowych.
 - **Zwracany widok:** Szczegóły planu treningowego.
-

5. MarkTrainingAsCompleted(int trainingId) - POST

- **Opis:** Oznacza wybrany trening jako ukończony.
 - **Parametry:**
 - trainingId: Identyfikator treningu.
 - **Działanie:**
 1. Wyszukuje trening o podanym trainingId.
 2. Oznacza trening jako ukończony (IsCompleted = true).
 3. Zapisuje zmiany w bazie danych.
 - **Zwracana wartość:** Przekierowanie do szczegółów planu.
-

6. **GeneratePdf(int planId)** - GET

- **Opis:** Generuje raport w formacie PDF dla wybranego planu treningowego.
 - **Parametry:**
 - planId: Identyfikator planu treningowego.
 - **Działanie:**
 1. Pobiera plan treningowy i powiązane ćwiczenia z bazy danych.
 2. Grupuje ćwiczenia według sesji treningowych.
 3. Tworzy dokument PDF, zawierający szczegóły planu oraz ćwiczenia.
 4. Zwraca plik PDF do pobrania.
 - **Zwracana wartość:** Plik PDF z raportem planu treningowego.
-

Podsumowanie

ActivityLogController jest kluczowym elementem aplikacji, zapewniającym funkcjonalność:

- Przeglądania planów treningowych.
- Śledzenia postępów w realizacji planu.
- Edytowania i usuwania planów.
- Eksportowania planów treningowych w formacie PDF.

Funkcjonalności te umożliwiają użytkownikom efektywne zarządzanie swoimi treningami oraz śledzenie postępów w ramach wyznaczonych celów.

HomeController

HomeController pełni rolę kontrolera głównej strony aplikacji, dostarczając funkcjonalności wyświetlania strony głównej, strony prywatności oraz obsługi błędów. Kontroler jest dostępny dla wszystkich użytkowników, niezależnie od tego, czy są zalogowani, czy nie, dzięki atrybutowi [AllowAnonymous].

Konstruktor

- **Parametry:**
 - ILogger<HomeController> logger: Dostarcza mechanizm logowania błędów i informacji w aplikacji.
 - Konstruktor inicjalizuje obiekt loggera, który może być używany do śledzenia aktywności i debugowania aplikacji.
-

Metody

1. **Index()** - GET

- **Opis:** Wyświetla stronę główną aplikacji.
 - **Działanie:**
 - Zwraca widok Index, który prezentuje główną stronę aplikacji.
 - **Zwracany widok:** Index.
-

2. **Privacy()** - GET

- **Opis:** Wyświetla stronę z polityką prywatności aplikacji.
 - **Działanie:**
 - Zwraca widok Privacy, który zawiera informacje dotyczące ochrony danych użytkowników.
 - **Zwracany widok:** Privacy.
-

3. **Error()** - GET

- **Opis:** Obsługuje błędy aplikacji i wyświetla szczegóły błędu.
 - **Działanie:**
 1. Tworzy obiekt modelu ErrorViewModel, który zawiera identyfikator błędu (RequestId).
 2. RequestId jest uzyskiwany z bieżącej aktywności lub z obiektu HttpContext.
 3. Zwraca widok Error, który prezentuje szczegóły błędu użytkownikowi.
 - **Zwracany widok:** Error.
-

Podsumowanie

HomeController pełni rolę centralnego punktu wejścia dla użytkowników aplikacji. Oferuje:

- Dostęp do strony głównej.
- Informacje o polityce prywatności.
- Mechanizm obsługi błędów.

Chociaż HomeController ma prostą funkcjonalność, odgrywa kluczową rolę w tworzeniu intuicyjnego interfejsu i zapewnieniu przyjaznego doświadczenia użytkownika.

SwimmingExerciseController

SwimmingExerciseController obsługuje logikę biznesową dotyczącą zarządzania ćwiczeniami pływackimi w systemie. Kontroler pozwala na wyświetlanie, filtrowanie, dodawanie, przeglądanie szczegółów oraz usuwanie ćwiczeń. Wykorzystuje bazę danych poprzez ApplicationDbContext do przechowywania i manipulacji danymi.

Konstruktor

- **Parametry:**
 - ApplicationDbContext context: Instancja kontekstu bazy danych aplikacji.
 - Konstruktor inicjalizuje _context, umożliwiając dostęp do bazy danych i operacje CRUD na ćwiczeniach.
-

Metody

1. Index() - GET

- **Opis:** Wyświetla wszystkie dostępne ćwiczenia.
 - **Działanie:**
 1. Pobiera listę ćwiczeń z bazy danych (SwimmingExercises).
 2. Zwraca widok Index, który prezentuje listę ćwiczeń.
 - **Zwracany widok:** Index.
-

2. Filter(string style, string category) - GET

- **Opis:** Filtruje ćwiczenia na podstawie stylu pływania (style) i kategorii (category).
 - **Parametry:**
 - style: Styl pływania (np. "Grzbiet", "Kraul").
 - category: Kategoria ćwiczenia (np. "Wytrzymałość", "Koordynacja").
 - **Działanie:**
 1. Pobiera z bazy ćwiczenia, które spełniają kryteria filtrowania.
 2. Zwraca częściowy widok _ExerciseListPartial z przefiltrowaną listą.
 - **Zwracany widok:** _ExerciseListPartial.
-

3. Add() - GET

- **Opis:** Wyświetla formularz umożliwiający dodanie nowego ćwiczenia.
 - **Działanie:**
 - Zwraca widok Add, który zawiera formularz dodawania ćwiczenia.
 - **Zwracany widok:** Add.
-

4. Add(SwimmingExercise exercise) - POST

- **Opis:** Obsługuje dodawanie nowego ćwiczenia do bazy danych.
 - **Parametry:**
 - exercise: Obiekt zawierający dane nowego ćwiczenia.
 - **Działanie:**
 1. Waliduje dane wejściowe.
 2. Dodaje ćwiczenie do bazy danych.
 3. Zwraca odpowiedź JSON z informacją o sukcesie lub błędach.
 - **Zwracana odpowiedź:** JSON.
-

5. Details(int id) - GET

- **Opis:** Wyświetla szczegóły konkretnego ćwiczenia.
 - **Parametry:**
 - id: Identyfikator ćwiczenia.
 - **Działanie:**
 1. Pobiera ćwiczenie z bazy danych na podstawie identyfikatora.
 2. Zwraca częściowy widok Details z informacjami o ćwiczeniu.
 - **Zwracany widok:** Details.
-

6. Delete(int id) - DELETE

- **Opis:** Usuwa ćwiczenie z bazy danych.
- **Parametry:**
 - id: Identyfikator ćwiczenia.
- **Działanie:**
 1. Sprawdza, czy ćwiczenie istnieje.
 2. Jeśli ćwiczenie jest domyślne (IsDefault), nie pozwala na jego usunięcie.

3. Usuwa ćwiczenie z bazy danych.
 4. Zwraca odpowiedź JSON z informacją o sukcesie lub błędach.
 - **Zwracana odpowiedź:** JSON.
-

Podsumowanie

SwimmingExerciseController jest kluczowym komponentem w systemie, który umożliwia użytkownikom zarządzanie ćwiczeniami pływackimi:

- Wyświetla pełną listę ćwiczeń.
- Pozwala filtrować ćwiczenia według określonych kryteriów.
- Obsługuje dodawanie nowych ćwiczeń z walidacją danych.
- Umożliwia przeglądanie szczegółów i usuwanie ćwiczeń.

Rozwiązanie to wspiera zarówno użytkowników końcowych, jak i administratorów w skutecznym zarządzaniu katalogiem ćwiczeń.

SwimmingPlanController

SwimmingPlanController obsługuje logikę związaną z planami treningowymi. Umożliwia użytkownikom tworzenie, przeglądanie i zarządzanie swoimi planami pływackimi. Kontroler integruje się z bazą danych za pomocą ApplicationDbContext, co pozwala na zarządzanie planami i ich szczegółami.

Konstruktor

- **Parametry:**
 - ApplicationDbContext context: Instancja kontekstu bazy danych aplikacji.
 - **Opis:** Konstruktor inicjalizuje _context, umożliwiając wykonywanie operacji CRUD na planach treningowych.
-

Metody

1. Index() - GET

- **Opis:** Wyświetla stronę główną modułu planów treningowych z opcjami dla użytkownika (np. generowanie planu, ręczne tworzenie planu).
- **Działanie:**
 - Renderuje widok Index.
- **Zwracany widok:** Index.

2. GeneratePlan() - GET

- **Opis:** Wyświetla formularz umożliwiający użytkownikowi generowanie nowego planu treningowego na podstawie preferencji.
- **Działanie:**
 - Renderuje widok GeneratePlan.
- **Zwracany widok:** GeneratePlan.

3. GeneratePlan(GeneratePlanViewModel model) - POST

- **Opis:** Generuje nowy plan treningowy na podstawie preferencji użytkownika z formularza.
- **Parametry:**
 - GeneratePlanViewModel model: Model danych zawierający odpowiedzi użytkownika z formularza (np. poziom trudności, styl pływania, cel treningu).
- **Działanie:**
 1. Waliduje dane wejściowe (ModelState.IsValid).
 2. Tworzy nowy obiekt planu treningowego (SwimmingPlan) i oblicza całkowity dystans (TotalDistance).
 3. Pobiera ćwiczenia z bazy danych, które pasują do preferencji użytkownika (kategoria, poziom trudności, styl).
 4. Generuje listę ćwiczeń (PlanExercise) dla każdego dnia treningowego:
 - Dodaje ćwiczenia, aż osiągnie dzienny dystans lub całkowity dystans planu.
 - Jeśli nie ma pasujących ćwiczeń, dodaje najmniejsze dostępne ćwiczenie.
 5. Zapisuje plan i ćwiczenia w bazie danych.
- **Zwracane widoki:**
 - W przypadku sukcesu: przekierowanie do szczegółów planu (Details).
 - W przypadku błędu: renderuje widok GeneratePlan z komunikatem błędu.

4. Details(int id) - GET

- **Opis:** Wyświetla szczegóły wygenerowanego planu treningowego.
- **Parametry:**
 - id: Identyfikator planu treningowego.

- **Działanie:**

1. Pobiera plan z bazy danych na podstawie identyfikatora (id).
2. Jeśli plan istnieje:
 - Pobiera jego nazwę i datę utworzenia.
 - Przekazuje dane do widoku za pomocą ViewData.
3. Jeśli plan nie istnieje, zwraca status 404 (NotFound).

- **Zwracany widok:** Details.

Podsumowanie

SwimmingPlanController to centralny element logiki biznesowej obsługujący zarządzanie planami treningowymi. Umożliwia użytkownikom:

1. **Tworzenie planów:**

- Na podstawie preferencji użytkownika.
- Automatyczne generowanie zestawów ćwiczeń z bazy danych.

2. **Przeglądanie szczegółów planu:**

- Wyświetlanie podstawowych informacji (nazwa, data utworzenia).
- Możliwość dalszego zarządzania w innych modułach (np. dziennik aktywności).

Funkcjonalność ta została zaprojektowana w sposób pozwalający na przyszłą rozbudowę, np. dodanie ręcznego edytowania planów czy eksportu do plików PDF.

6. Opis systemu użytkowników i zarządzania dostępem

Aplikacja **SplashTrainer** została zaprojektowana z myślą o zapewnieniu indywidualnych możliwości dla użytkowników zalogowanych oraz ograniczonym dostępie dla gości. Dzięki takiemu podejściu system łączy elementy współdzielonych zasobów (globalnych) z prywatnymi danymi użytkownika, co pozwala na personalizację bez utraty ogólnej funkcjonalności.

Role użytkowników

W aplikacji występuje jedna kluczowa rola – **zalogowany użytkownik**:

- **Gość (niezalogowany użytkownik):**

- Gość ma dostęp jedynie do strony głównej, która wprowadza w tematykę aplikacji i przedstawia jej możliwości. Nie ma dostępu do innych funkcjonalności aplikacji.

- Gość nie widzi szczegółów katalogu ćwiczeń ani nie ma możliwości interakcji z planami treningowymi.
 - **Zalogowany użytkownik:**
 - Po zalogowaniu użytkownik uzyskuje dostęp do pełnej funkcjonalności aplikacji, w tym:
 - Generowania planów treningowych.
 - Przeglądania i filtrowania katalogu ćwiczeń.
 - Dodawania własnych ćwiczeń do katalogu.
 - Zarządzania swoimi planami treningowymi (przegląd, edycja, usuwanie, oznaczanie postępów).
 - Generowania raportów PDF z postępami swoich planów.
 - Wszystkie operacje na danych są przypisane do zalogowanego użytkownika i dostępne tylko dla niego.
-

Dostęp do informacji

System dostępu do danych jest podzielony na informacje **globalne** i **prywatne**:

- **Globalne dane:**
 - **Ćwiczenia w katalogu:** Katalog ćwiczeń jest współdzielony między wszystkimi użytkownikami. Ćwiczenia w bazie danych, które zostały zdefiniowane jako domyślne (globalne), są widoczne i dostępne dla każdego użytkownika.
 - **Styl i kategoria ćwiczeń:** Wszyscy użytkownicy, niezależnie od zalogowania, mogą przeglądać style i kategorie ćwiczeń.
- **Prywatne dane:**
 - **Ćwiczenia dodane przez użytkownika:**
 - Ćwiczenia dodane przez użytkownika są widoczne tylko dla niego.
 - Użytkownik może zarządzać swoimi ćwiczeniami, np. usuwać lub edytować je, ale inne osoby ich nie widzą.
 - **Plany treningowe:**
 - Plany treningowe są powiązane z konkretnym użytkownikiem za pomocą jego identyfikatora (UserId) i są widoczne wyłącznie dla niego.
 - Użytkownik może generować, przeglądać szczegóły, edytować nazwę, usuwać lub śledzić postępy w ramach swoich planów.
 - **Postępy użytkownika:**
 - Informacje o ukończonych treningach i procentowy postęp w realizacji planu są przechowywane jako prywatne dane użytkownika.

- Dane te są dostępne tylko dla zalogowanego użytkownika.

Techniczne aspekty przypisywania ról i danych

- **System logowania i rejestracji:**
 - Wykorzystano mechanizmy Identity Framework do zarządzania użytkownikami. Użytkownicy są identyfikowani za pomocą unikalnego identyfikatora (**UserId**), który jest przechowywany w bazie danych.
 - Każdy nowo zarejestrowany użytkownik automatycznie otrzymuje uprawnienia zalogowanego użytkownika.
- **Powiązanie danych z użytkownikami:**
 - **Ćwiczenia użytkownika:** W modelu danych ćwiczenia zawierają pole identyfikujące ich właściciela, co pozwala na ścisłe powiązanie danych z kontem użytkownika.
 - **Plany treningowe:** Podobnie jak ćwiczenia, plany treningowe są powiązane z użytkownikami poprzez **UserId**. Dzięki temu każdy użytkownik widzi tylko swoje dane.
- **Kontrola dostępu:**
 - Wszystkie kontrolery, oprócz tych związanych z rejestracją i logowaniem, wymagają autoryzacji użytkownika. Mechanizm ten został zrealizowany za pomocą atrybutu [Authorize] w kontrolerach, co zapobiega nieautoryzowanemu dostępowi do danych.

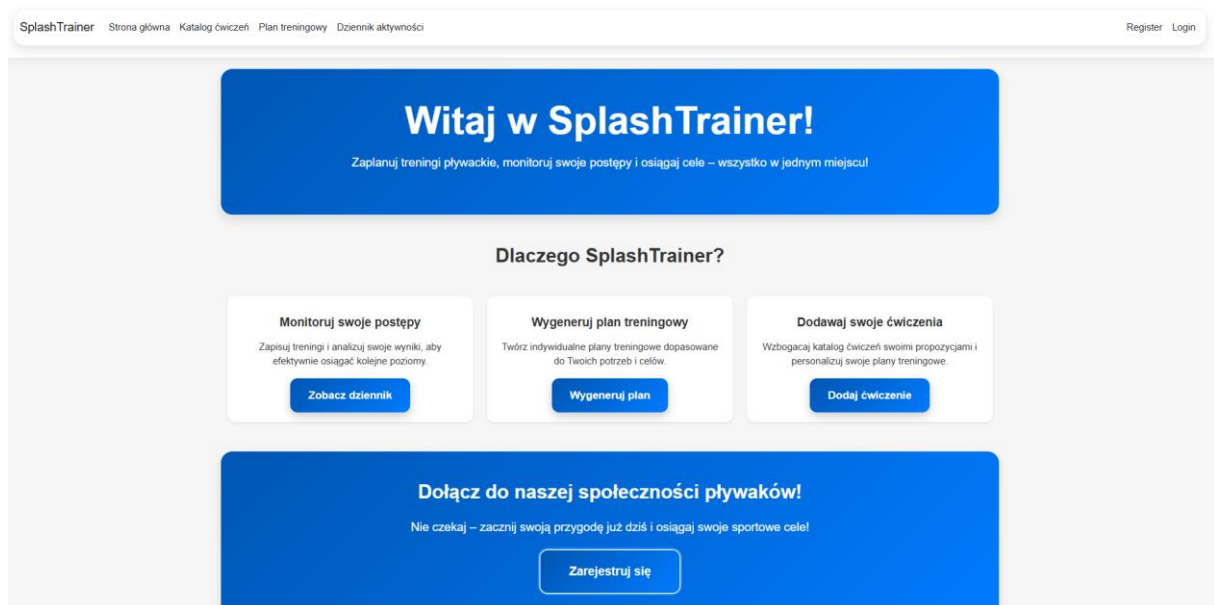
Widoki i zobrazowane funkcje kontrolerów :

Widok - Główny

Strona główna aplikacji **SplashTrainer** to centralny punkt startowy, który w przejrzysty sposób prezentuje wszystkie kluczowe funkcjonalności platformy. Nagłówek "Witaj w SplashTrainer!" zachęca do rozpoczęcia przygody z planowaniem treningów pływackich, monitorowaniem postępów i osiąganiem sportowych celów.

Sekcja "Dlaczego SplashTrainer?" przedstawia trzy główne funkcje: monitorowanie postępów, generowanie planów treningowych oraz możliwość dodawania własnych ćwiczeń do katalogu. Każda funkcja jest wizualnie podkreślona w formie kafelków z opisami i przyciskami akcji, które prowadzą do odpowiednich sekcji aplikacji.

Na dole strony znajduje się wezwanie do rejestracji – "Dołącz do naszej społeczności pływaków!" – z wyraźnym przyciskiem "Zarejestruj się", co zachęca nowych użytkowników do założenia konta. Dzięki przemyślanemu układowi i intuicyjnej nawigacji, strona główna jest przyjazna dla użytkownika i skutecznie wprowadza w możliwości aplikacji.



Widok – Katalog ćwiczeń

Widok katalogu ćwiczeń w aplikacji **SplashTrainer** umożliwia użytkownikowi przeglądanie, filtrowanie oraz zarządzanie ćwiczeniami pływackimi, które są dostępne w systemie. Górna część widoku zawiera przyciski umożliwiające wybór stylu pływackiego (Grzbiet, Kraul, Żaba, Delfin) oraz kategorii ćwiczeń (Nogi, Ręce, Koordynacja, Wytrzymałość, Szybkość). Dzięki temu użytkownik może łatwo zawęzić listę ćwiczeń do tych, które najlepiej odpowiadają jego potrzebom i preferencjom.

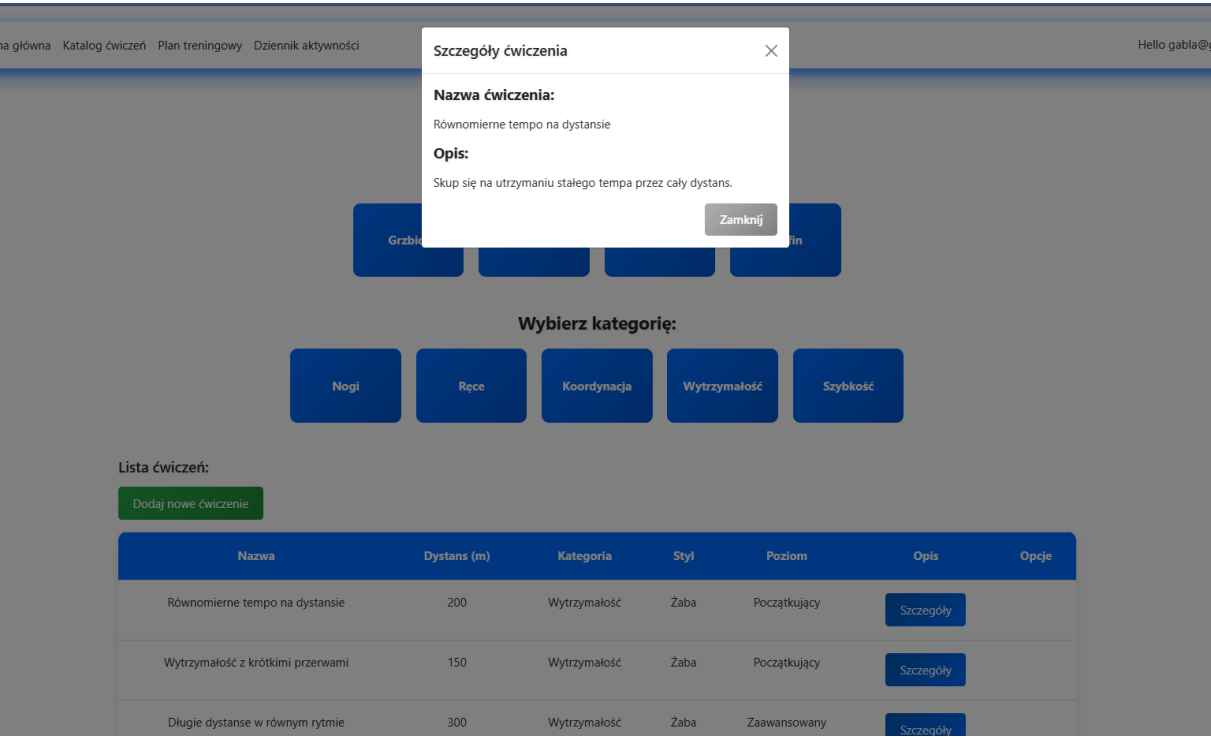
Pod sekcją filtrów znajduje się lista ćwiczeń w formie tabeli. Każdy wiersz tabeli zawiera szczegóły dotyczące ćwiczenia, takie jak nazwa, dystans, kategoria, styl, poziom trudności oraz opcję przejścia do szczegółowych informacji o danym ćwiczeniu. Przycisk "Szczegóły" pozwala użytkownikowi uzyskać dodatkowe informacje o ćwiczeniu, takie jak dokładny opis i przeznaczenie.

Dostępny jest również przycisk "Dodaj nowe ćwiczenie", który umożliwia dodanie własnego ćwiczenia do katalogu. Ta funkcjonalność pozwala na wzbogacenie bazy danych ćwiczeń o nowe propozycje i personalizację katalogu.

Cały widok katalogu jest przejrzysty, a intuicyjny układ i responsywne przyciski zapewniają pozytywne doświadczenie użytkownika. Dzięki możliwości filtrowania i edytowania ćwiczeń użytkownik może łatwo dostosować katalog do swoich potrzeb.



Katalog prezentujący ćwiczenia



Widok szczegółów zawiera cenne informacje dot. techniki

Dodaj nowe ćwiczenie

Nazwa

Dystans (m)

Kategoria

Wybierz kategorię

Styl

Grzbiet

Poziom

Wybierz poziom

Opis

Dodaj

Anuluj

Widok – Plan treningowy

Strona zachęcająca do wygenerowania planu treningowego została zaprojektowana w taki sposób, aby inspirować użytkownika i nakłonić go do skorzystania z funkcjonalności aplikacji. Główny nagłówek „Zacznij swój trening pływacki!” jasno wskazuje cel strony, a pod nim znajduje się zachęcający podtytuł, który informuje użytkownika o możliwości stworzenia planu treningowego dostosowanego do jego indywidualnych potrzeb. Centralnym elementem wizualnym jest obraz przedstawiający pływaka w trakcie treningu, który pełni rolę inspiracyjną i pomaga użytkownikowi utożsamić się z ideą systematycznego treningu.

Po prawej stronie obrazu znajduje się sekcja tekstowa podkreślająca profesjonalizm oferowanych usług. Treść „Twój plan oparty na wiedzy ekspertów!” informuje użytkownika, że plany treningowe korzystają z bogatej bazy ćwiczeń, opracowanej przez doświadczonych trenerów pływania. Dodatkowo zaznaczono, że oferowane plany są skuteczne i dostosowane zarówno do indywidualnych celów, jak i poziomu zaawansowania użytkownika. Taki przekaz buduje zaufanie do aplikacji oraz jej funkcjonalności.


Na stronie umieszczono również wyraźny i atrakcyjny wizualnie przycisk akcji z napisem „Wygeneruj plan”. Przycisk został zaprojektowany w kontrastującym niebieskim kolorze, co sprawia, że łatwo przyciąga wzrok użytkownika. Kliknięcie przycisku przenosi użytkownika bezpośrednio do formularza generowania planu treningowego, ułatwiając mu rozpoczęcie korzystania z tej funkcji.

Dzięki przejrzystej strukturze, inspirującej grafice oraz klarownym komunikatom, strona ta skutecznie zachęca użytkowników do działania i korzystania z kluczowej funkcji aplikacji, jaką jest generowanie planu treningowego. Jest to doskonałe połączenie motywacji, prostoty i funkcjonalności, które sprawia, że użytkownik czuje się pewnie, podejmując pierwszy krok w kierunku realizacji swoich sportowych celów.

[SplashTrainer](#) [Strona główna](#) [Katalog ćwiczeń](#) [Plan treningowy](#) [Dziennik aktywności](#) [Hello gabla@gmail.com!](#) [Logout](#)

Zacznij swój trening pływacki!

Skorzystaj z naszego generatora, aby stworzyć plan treningowy, który odpowiada Twoim potrzebom.



Twój plan oparty na wiedzy ekspertów!

Każdy plan treningowy korzysta z naszej rozbudowanej bazy ćwiczeń, które zostały stworzone przez doświadczonych trenerów pływania. Dzięki temu masz pewność, że Twoje treningi są skuteczne i dostosowane do Twoich celów oraz poziomu zaawansowania.

[Wygeneruj plan](#)

© 2024 - SplashTrainer - Wszelkie prawa zastrzeżone - [Strona główna](#)

Formularz generowania planu treningowego w SplashTrainer pozwala użytkownikowi dostosować plan do swoich indywidualnych potrzeb. Składa się z prostych pól, takich jak: **nazwa planu**, **cel treningu** (np. wytrzymałość, szybkość), **poziom trudności** (początkujący lub zaawansowany) oraz **styl pływania** (grzbiet, kraul, żaba, delfin). Dodatkowo użytkownik określa **dystans na treningu**, **częstotliwość tygodniową** i **czas trwania planu w tygodniach**.

Po uzupełnieniu formularza i kliknięciu przycisku "Generuj plan", aplikacja tworzy spersonalizowany program oparty na podanych preferencjach. Intuicyjny interfejs i wbudowana walidacja sprawiają, że korzystanie z formularza jest proste, szybkie i skuteczne.

Wygeneruj swój plan treningowy

Nazwa planu:

Trening Wytrzymałość - Kraul 2km

Cel treningu:

Wytrzymałość

Poziom trudności:

Zaawansowany

Styl pływania:

Kraul

Dystans na treningu (metry) :

2000

Ilość treningów w tygodniu:

3 dni w tygodniu

Długość planu (w tygodniach) :

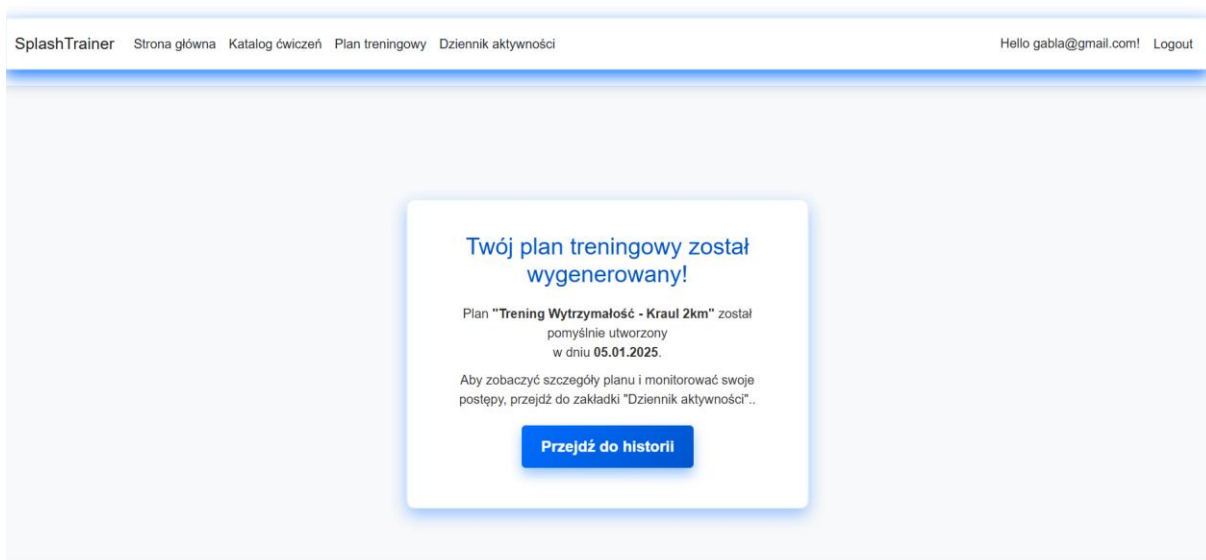
4

Generuj plan

© 2024 - SplashTrainer - Wszelkie prawa zastrzeżone - [Strona główna](#)

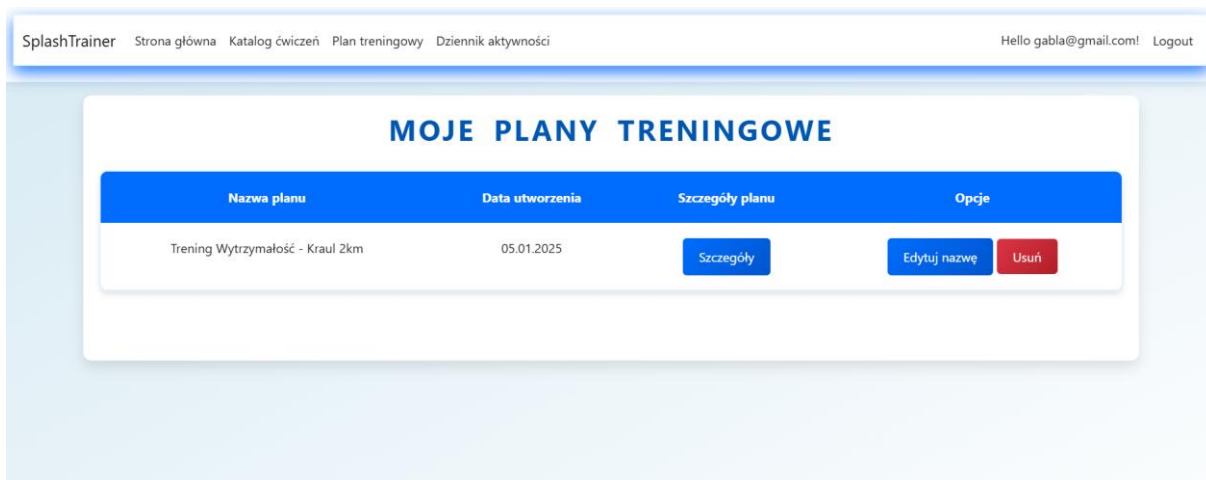
Widok po wygenerowaniu planu treningowego w SplashTrainer informuje użytkownika o pomyślnym utworzeniu spersonalizowanego planu. Znajduje się tutaj szczegółowy komunikat z nazwą planu, datą jego utworzenia oraz prostą instrukcją, jak przejść do zakładki "Dziennik aktywności", aby zobaczyć szczegóły i śledzić swoje postępy. W centralnym punkcie widoku znajduje się wyróżniony przycisk "Przejdź do historii", który pozwala szybko przejść do odpowiedniego miejsca w aplikacji.

Minimalistyczny design i czytelny komunikat zapewniają, że użytkownik dokładnie wie, co zrobić, aby kontynuować korzystanie z aplikacji w płynny sposób.



Widok – Dziennik aktywności

Widok „Moje Plany Treningowe” w aplikacji SplashTrainer umożliwia użytkownikowi łatwe zarządzanie swoimi planami treningowymi w przejrzysty i intuicyjny sposób. Każdy plan wyświetlany jest w formie tabeli, gdzie użytkownik może zobaczyć nazwę planu, datę jego utworzenia oraz skorzystać z dostępnych opcji. Przyciski akcji umożliwiają wykonywanie kluczowych operacji, takich jak przeglądanie szczegółów planu, edycja jego nazwy lub usunięcie go z listy. Funkcja „Szczegóły” pozwala użytkownikowi zapoznać się z pełnym harmonogramem treningów, w tym z informacjami o ćwiczeniach, dystansach oraz stylach pływania. Opcja edycji nazwy daje możliwość personalizacji, co ułatwia organizację planów, natomiast możliwość usunięcia niepotrzebnych planów pozwala na zachowanie porządku w zakładce. Całość została zaprojektowana w sposób prosty i przejrzysty, z wyróżnionymi kolorystycznie przyciskami, które zapewniają wygodne i szybkie korzystanie z funkcjonalności. Dzięki temu widok ten stanowi centralne miejsce do zarządzania planami treningowymi i dostosowywania ich do indywidualnych potrzeb użytkownika.



Widok – Szczegóły planu treningowego

Widok szczegółów planu treningowego prezentuje pełny harmonogram treningu w uporządkowany i przejrzysty sposób. Użytkownik może zobaczyć nazwę planu, całkowity dystans do pokonania, szczegóły dotyczące każdego treningu, w tym listę ćwiczeń, ich dystanse, poziom zaawansowania i styl pływania. W prawym górnym rogu znajduje się dynamiczny wskaźnik postępu planu, który na bieżąco informuje użytkownika o ukończonych treningach. Dodatkowo dostępne są przyciski akcji, takie jak „Pobierz PDF”, umożliwiający wygenerowanie szczegółowego raportu w formacie PDF, oraz „Zaznacz jako zrobione”, który pozwala oznaczyć trening jako ukończony, co automatycznie aktualizuje postęp w planie. Widok jest uzupełniony o przycisk „Powrót do listy”, umożliwiający szybki powrót do widoku listy planów użytkownika.

7. Najciekawsze funkcje w aplikacji SplashTraining

1. Generator planu treningowego

Generator planu treningowego to zaawansowane narzędzie, które na podstawie odpowiedzi użytkownika (takich jak cel treningu, poziom zaawansowania, preferowany styl pływania, dystans na trening i częstotliwość) automatycznie tworzy spersonalizowany harmonogram treningów. Dzięki wbudowanym algorytmom generator dobiera odpowiednie ćwiczenia z bazy danych, uwzględniając dystans, styl oraz poziom zaawansowania, a także równomiernie rozkłada obciążenie na tygodnie i dni treningowe.

2. Dynamiczny wskaźnik postępu planu

Wskaźnik postępu w widoku szczegółów planu umożliwia użytkownikowi śledzenie swoich osiągnięć w czasie rzeczywistym. Po oznaczeniu treningu jako ukończonego, wskaźnik automatycznie aktualizuje się, pokazując procentowy postęp oraz liczbę ukończonych treningów w stosunku do całości.

3. Generowanie raportu PDF

Funkcjonalność generowania raportu PDF pozwala użytkownikowi na wygenerowanie szczegółowego harmonogramu planu treningowego w formacie PDF. Dokument zawiera dane o nazwie planu, dacie jego utworzenia, ćwiczeniach, dystansach, stylach oraz poziomach. To przydatne narzędzie umożliwia łatwe drukowanie lub udostępnianie planu w formacie offline.

4. Filtracja ćwiczeń w katalogu

W katalogu ćwiczeń użytkownik może skorzystać z intuicyjnego systemu filtrowania, który umożliwia wybranie ćwiczeń na podstawie stylu pływania i kategorii. Dzięki temu szybko znajduje odpowiednie ćwiczenia, które spełniają jego potrzeby, np. poprawę koordynacji, siły czy wytrzymałości.

5. Personalizacja planów treningowych

Aplikacja umożliwia użytkownikowi edytowanie nazw swoich planów treningowych, co pozwala na lepszą organizację oraz dopasowanie do jego indywidualnych potrzeb. Możliwość zmiany nazw sprawia, że użytkownik może łatwo zarządzać wieloma planami jednocześnie.

6. Dziennik aktywności

Dziennik aktywności rejestruje wszystkie kluczowe działania użytkownika, takie jak tworzenie, modyfikowanie i usuwanie planów treningowych. Dzięki temu użytkownik ma pełną historię swoich działań, co ułatwia śledzenie postępów i zarządzanie aktywnością.

7. Responsywny design i intuicyjny interfejs

Cała aplikacja została zaprojektowana z myślą o łatwości użytkowania. Interfejs jest prosty, responsywny i intuicyjny, co zapewnia użytkownikom komfortowe korzystanie zarówno na komputerze, jak i na urządzeniach mobilnych.