

Data: 20.05.2018

NTime

Aplikacja do przeliczania wyników Web API

Dokumentacja Projektu

Autorzy:

Grzegorz Krzysiak
Tomasz Chudzik

Spis treści

Specyfikacja.....	5
Opis biznesowy	5
Wymagania funkcjonalne	5
Wymagania нефункционалне	6
Harmonogram projektu	6
Architektura aplikacji	6
Dokumentacja końcowa (powykonawcza)	7
Wymagania systemowe:	8
Biblioteki wraz z określeniem licencji	8
Wdrożenie projektu	9
Uruchomienie projektu w Visual Studio	9
Instrukcja użycia.....	10
Instrukcja utrzymania	14
Raport odstępstw od specyfikacji wymagań.....	14
Dokumentacja utworzonego Web API.....	14
Założenia:	15
Logowanie:.....	15
POST Token	15
Kontroler Account:.....	16
POST api/Account/Logout.....	16
POST api/Account/ChangePassword	16
POST api/Account/Register.....	16
GET api/Account/Role.....	16
Kontroler Competition	17
Model danych	17
GET /api/Competition?ItemsOnPage=10&PageNumber=0.....	17
GET /api/Competition/FromPlayerAccount/1?ItemsOnPage=10&PageNumber=0.....	17
GET /api/Competition/1	17
PUT /api/Competition/1	17
POST /api/Competition	17
POST /api/Competition/1/OrganizerLock/true	17
POST /api/Competition/1/OrganizerLock/false.....	17
Kontrolery AgeCategory, Distance, ExtraPlayerInfo	17
Model danych AgeCategory.....	17
Model danych Distance.....	18

Model danych ExtraPlayerInfo.....	18
GET api/<attr>/FromCompetition/1	18
GET api/<attr>/1	18
PUT api/<attr>/1	18
POST api/<attr>/IntoCompetition/1	18
DELETE api/<attr>/1.....	18
Kontroler Player	18
Model danych filtru.....	18
Model danych do publicznej listy (tylko do wglądu).....	19
Model danych do zapisów	19
Model danych pełny.....	19
POST api/Player/TakeSimpleList/FromCompetition/1?ItemsOnPage=10&PageNumber=0	20
POST api/Player/TakeFullList/FromCompetition/1?ItemsOnPage=10&PageNumber=0	20
GET api/Player/TakeFullList/FromPlayerAccount/1?ItemsOnPage=10&PageNumber=0	20
GET api/Player/FromPlayerAccount/1/FromCompetition/1	20
GET api/Player/1	20
PUT api/Player/1	20
DELETE api/Player/1.....	20
GET api/Player/Register/1	21
PUT api/Player/Register/1	21
POST api/Player/Register/IntoCompetition/1	21
Kontroler PlayerAccount	21
Model Danych	21
GET api/PlayerAccount/Search/Filtr?ItemsOnPage=10&PageNumber=0.....	21
GET api/PlayerAccount/Search?ItemsOnPage=10&PageNumber=0.....	21
GET api/PlayerAccount	21
GET api/PlayerAccount/1.....	21
GET api/PlayerAccount/FromCompetition/1?ItemsOnPage=10&PageNumber=0	21
PUT api/PlayerAccount/1.....	21
Kontroler OrganizerAccount	22
Model Danych	22
GET api/OrganizerAccount/Search/Filtr?ItemsOnPage=10&PageNumber=0	22
GET api/OrganizerAccount/Search?ItemsOnPage=10&PageNumber=0	22
GET api/OrganizerAccount/ByCompetition/1	22
GET api/OrganizerAccount/1	22
PUT api/OrganizerAccount/1.....	22

POST api/OrganizerAccount.....	22
DELETE api/OrganizerAccount/1.....	22
POST api/OrganizerAccount/1/PasswordReset	22
POST api/OrganizerAccount/1/SetCompetition/1.....	22
POST api/OrganizerAccount/1/UnsetCompetition/1	22
Kontroler ModeratorAccount	22
GET api/ModeratorAccount.....	22
POST api/ModeratorAccount.....	22
DELETE api/ModeratorAccount/aaa.....	23
POST api/ModeratorAccount/aaa/PasswordReset	23
POST api/ModeratorAccount/aaa/Bust.....	23
POST api/ModeratorAccount/aaa/Unbust	23
Kontroler AdministratorAccount	23
GET api/AdministratorAccount	23
POST api/AdministratorAccount.....	23
DELETE api/AdministratorAccount/aaa	23
POST api/AdministratorAccount/aaa/PasswordReset	23
Dokumentacja końcowa (powykonawcza) - punkty wymagane przez prowadzącego zajęcia.....	23
Model danych	24

Specyfikacja

Opis biznesowy

Aplikacja służy firmie przeprowadzającej pomiary czasu na zawodach sportowych. Dzięki tej aplikacji użytkownicy za pomocą przeglądarki mogą rejestrować się na zawody oraz przeglądać dane odnośnie wszystkich zawodów zorganizowanych przez firmę pomiarową. Ponadto organizator każdych zawodów może założyć sobie specjalne konto, dzięki któremu będzie w stanie edytować dane dla swoich zawodów i księgować wpłaty zawodników.

Wymagania funkcjonalne

1. Zawodnicy mają udostępniony interfejs webowy do przeglądania wszystkich zawodów organizowanych przez firmę pomiarową.
2. Dla każdych zawodów widoczna jest lista zawodników już zapisanych na dane zawody
3. Każdemu użytkownikowi poza podstawowymi danymi osobowymi wyświetla się informacja czy dany zawodnik został już opłacony
4. Jeżeli czas zapisu na zawody jeszcze nie upłynął każdy użytkownik strony może się zapisać na zawody poprzez formularz
5. Formularz zabezpieczony jest walidacją zarówno po stronie przeglądarki jak i po stronie serwera
6. Serwer pozwala na migracje między nowym i starym modelem danych
7. Formularz zabezpieczony jest również przeciwko robotom poprzez Google ReCAPTCHA.
8. Organizatorzy Mają udostępnione specjalne konta, które są przypisane do zawodów, które organizują
9. Podczas zakładania konta przy pomocy maila – adres email jest walidowany czy rzeczywiście istnieje, a konto potwierdzane poprzez link wysłany na adres email posiadacza konta
10. Zalogowanym organizatorom, dla ich zawodów, wyświetla się nowy widok zawodników, w którym mogą księgować wpłaty zawodników
11. API, z którego korzystają organizatorzy musi być zabezpieczone przeciwko nieautoryzowanym użytkownikom
12. Aplikacja przekierowywana jest na konkretną domenę zamawiającego
13. Dane szyfrowane są przy pomocy certyfikatu SSL
14. Strona jest responsywna i dostosowana do różnych rozmiarów ekranu
15. Kontrolki zastosowane w aplikacji mają spójny, ładny design - w tym wypadku oparty o Angular Material.
16. Codziennie tworzone są backupy bazy danych
17. Aplikacja została wdrożona praktycznego użytku i zostać udostępniona użytkownikom
18. Strona musi być w stanie obsłużyć minimum 200 zapisów w ciągu dnia.
19. Aplikacja została przetestowana przy pomocy ponad 80 testów jednostkowych

Wymagania niefunkcjonalne

Baza danych: Microsoft SQL Server Express

ORM: Entity Framework

Testy jednostkowe: NUnit, Jasmine, Karma, Angular CLI

Interfejs graficzny: Angular, Angular Material

Technologia serwera REST API: .NET Web API

Technologia klienta REST API: Angular HttpClient

Zabezpieczenia autoryzacji i bezpieczeństwa: OAuth2, ASP.NET, Google ReCAPTCHA

Harmonogram projektu

20.12.2017 – Ustalenie wymagań aplikacji z zamawiającym

28.12.2017 – Zaplanowanie interfejsu i funkcjonalności aplikacji

6.12.2017 – Prezentacja pierwszego etapu

15.01.2018 – Wstępny interfejs graficzny aplikacji

20.01.2018 – Obsługa logowania przez użytkowników o różnym poziomie uprawnień

05.01.2018 – Wstępna budowa architektury w Angularze

10.01.2018 – Udostępnienie kontrolerów ASP.NET wraz z zabezpieczeniami autoryzacyjnymi

20.01.2018 – Stworzenie testowej wersji aplikacji

01.02.2018 – Stworzenie podstawowej bazy testów jednostkowych

01.03.2018 – Wdrożenie ReCAPTCHA

01.04.2018 – Wdrożenie podstawowej wersji na serwer testowej

10.04.2018 – Przekierowanie domeny na serwer i wprowadzenie certyfikatu SSL

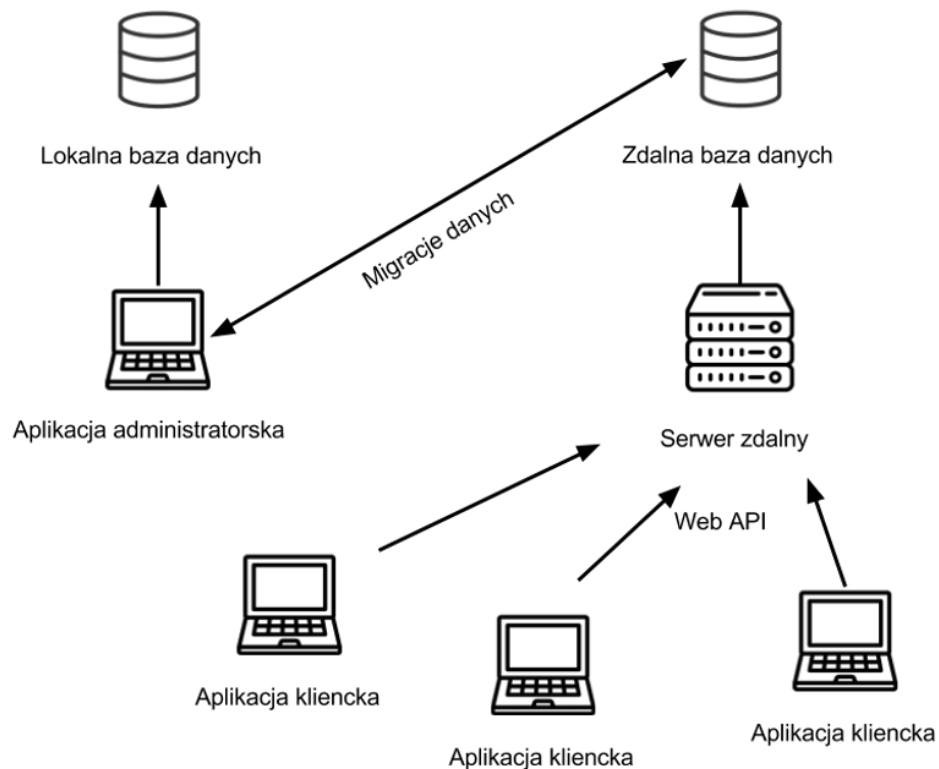
15.04.2018 – Wdrożenie aplikacji na serwer i udostępnienie jej użytkownikom

02.05.2018 – Dodanie edycji wielu użytkowników na raz

20.05.2018 – Oddanie całości aplikacji

Architektura aplikacji

Na poniższym rysunku przedstawiono architekturę platformy NTime z rozbiciem na poszczególne komponenty



Rysunek 1 Architektura NTime

Na przedstawionym rysunku wyróżniamy następujące elementy:

- **Lokalna baza danych** - Local DB, będąca w pogotowiu, by uruchomić jedną z lokalnych, binarnych kopii bazy danych na wypadek utraty połączenia z siecią
- **Zdalna baza danych** – Główna baza danych znajdująca się na serwerze, w której znajdują się wszystkie dane odnośnie przeprowadzonych zawodów oraz kont użytkowników
- **Aplikacja administratorska** – Aplikacja stworzona w pierwszym etapie projektu pozwalająca administratorowi na edycję wszystkich danych dotyczących zawodów oraz na konfigurację i obliczenie wyników w dniu imprezy
- **Serwer zdalny** – Serwer, na którym znajduje się REST API, z którego korzystają wszystkie aplikacje klienckie i który łączy się ze zdalną bazą danych
- **Aplikacja kliencka** – Aplikacja typu SPA wykonana w Angularze 5 z pomocą biblioteki do elementów wizualnych - Angular Material.

Dokumentacja końcowa (powykonawcza)

Wymagania systemowe:

Aplikacja użytkownika:

System Operacyjny Windows 7 lub wyższej,

Platforma .net Framework 4.7,

Dostęp do internetu

Server:

System Operacyjny Windows Server 2012,

Baza danych MS SQL 2012,

IIS 8.0

Biblioteki wraz z określeniem licencji

nr	Komponent i wersja	Opis	Licencja
1	.Net Framework 4.7	Platforma programistyczna opracowana przez Microsoft, obejmująca środowisko uruchomieniowe (Common Language Runtime – CLR) oraz biblioteki klas dostarczające standardowej funkcjonalności dla aplikacji.	Freeware
2	Entity Framework 7	Narzędzie typu ORM (Object Relational Mapping), pozwalającym odwzorować relacyjną bazę danych za pomocą architektury obiektowej.	Apache License
3	NUnit 3	Platforma do testów jednostkowych	MIT License for 3.0, BSD-style (modified zlib license) for 2.x
4	Angular 5	Otwarty framework i platforma do tworzenia aplikacji SPA, napisany w języku TypeScript i wspierany oraz rozwijany przez Google	MIT
5	Jasmine	Otwarty framework służący do pisania testów jednostkowych dla aplikacji korzystających z JavaScript	MIT
6	Karma	Framework służący do uruchamiania w przeglądarce napisanych testów. Jest zintegrowany z Angular CLI i z korzystaniem z Karmy jest niewidoczne dla programisty	MIT
7	ASP.NET	Zbiór technologii opartych na frameworku zaprojektowanym przez firmę Microsoft. Przeznaczony jest do budowy różnorodnych aplikacji internetowych, a także aplikacji typu XML Web Services.	Apache License

Wdrożenie projektu

Aplikacja użytkownika:

Produkt nie wymaga instalacji. Potrzebne jest jedynie spełnienie wymagań opisanych w sekcji Wymagania systemowe

Server:

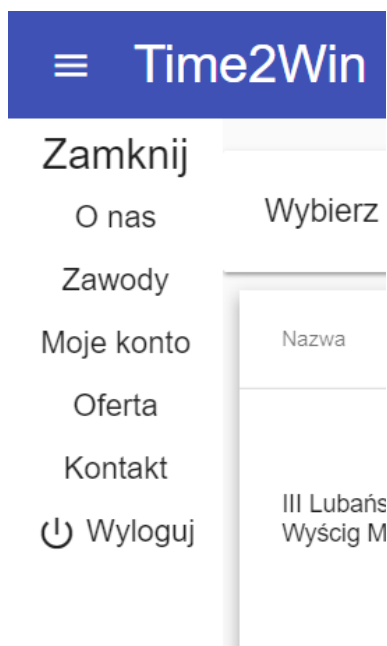
Wymagane jest wdrożenie aplikacji na serwer IIS. Następnie należy skonfigurować connection stringi o nazwach DefaultConnection i NTime. W folderze witryny należy umieścić wygenerowane pliki wdrożenia i folder bundles, który znajduje się w folderze Server projektu i jest generowany przy uruchomienie opcji *Publish*.

Uruchomienie projektu w Visual Studio

1. Najprościej proces instalacji przeprowadzić na komputerze z systemem Windows 10
2. Pozostałe wymagania: .Net framework 4.7, Visual Studio 2017.
3. Należy rozpakować plik ntime.zip do folderu *ntime*
4. Należy wejść do tego folderu i następnie wejść w następującą ścieżkę *chudytom-ntime-bef80e073688\NTime\Server\ntimeAngular*
5. W tym kroku konieczne będzie zainstalowanie wszystkich pakietów npm. W tym celu zakładamy, że npm został zainstalowany na komputerze. Można to łatwo sprawdzić włączając wiersz poleceń, np. Powershell i wpisać komendę *npm --version* Jeśli wyświetli się wersja npm, to znaczy, że jest on poprawnie zainstalowany. Jeśli nie, to należy zainstalować pełnego node.js ze strony pod linkiem: <https://nodejs.org/en/download/> dla swojej platformy
6. Po upewnieniu się, że npm jest zainstalowany na naszym komputerze, w folderze do którego weszliśmy w punkcie 4, wpisujemy w wierszu poleceń komendę *npm install*. Pobranych zostanie około 300 MB pakietów potrzebnych do uruchomienia aplikacji
7. W tym momencie można uruchomić aplikację korzystając z przekierowania na serwer testowy wpisując w wierszu poleceń komendę *npm start*
8. Jeśli chcemy przejść do debugowania w Visual Studio wykonujemy kolejne kroki
9. Wpisujemy komendę *npm run-script build* Pozwoli to w odpowiedni sposób zbudować front aplikacji stworzony w Angularze
10. Następnie cofamy się do ścieżki *ntime\chudytom-ntime-bef80e073688\NTime*
11. Włączamy plik solucji *NTime.sln* przy pomocy Visual Studio 2017.
12. W zakładce *Solution Explorer* klikamy PPM na projekt *Server* i wybieramy opcję *Set as StartUp Project*
13. Przełączamy budowanie projektu w tryb *Release*
14. Należy upewnić się, że w Visualu jak wejdziemy w widok *SQL Server Object Explorer* w ścieżce *SQL Server/(localdb)/MSSQLLocalDB(...)/Databases* nie istnieje baza danych *aspnet-Server-20171216125843*
Jeśli taka baza istnieje należy ją usunąć przed uruchomieniem programu. Ważne, żeby zaznaczyć obydwa checkboxy: *Delete backup (...)* i *Close existing connections*.
15. Przy pomocy klawiszy *ctrl + F5* uruchamiamy projekt

Instrukcja użycia

Strona składa się z 5 zakładek:



Po wyborze zakładki "Zawody", można obejrzeć listę dostępnych zawodów, sprawdzić listę zawodników i zarejestrować się na któreś.

Time2Win						
O nas Zawody Moje konto Oferta Kontakt Wyloguj						
Wybierz interesujące Cię zawody						
Nazwa	Miasto	Data zawodów	Data końca zapisów	Link do strony		
Maraton "MTB Forteca Lwówka Śląskiego"	Lwówek Śląski	22.09.2018	20.09.2018	mtb-gp.pl	Lista zawodników	Zarejestruj się
Otwarte Mistrzostwa Dolnego Śląska w Maratonie MTB im. Artura Filipiaka	Zieleniec k. Dusznik Zdrój	29.07.2018	28.07.2018	www.xcm-filipiak.pl	Lista zawodników	Zarejestruj się
VII Świdwiński Szosowy Maraton Rowerowy	Świdwin	14.07.2018	13.07.2018	www.supermaratony.org	Lista zawodników	Zarejestruj się
Kryterium Uliczne Powiatu Bolesławieckiego	Bolesławiec	3.06.2018	2.06.2018		Lista zawodników	Zarejestruj się
Mistrzostwa Obornik w Kolarstwie Szosowym - Jazda indywidualna na czas (Wpisz LICENCJĘ obok klubu)	Oborniki	2.06.2018	2.06.2018	www.kolarstwo.oborniki.pl	Lista zawodników	Zarejestruj się
III Lubański Wyścig MTB	Lubań	27.05.2018	25.05.2018	www.kwisaluban.pl	Lista zawodników	Zarejestruj się
Gatta Prestige Race Klasyki Łódzkie - 2 etap	Zduńska Wola	19.05.2018	19.05.2018	www.gattabiketeam.com/gatta-prestige-race-2018	Lista zawodników	
Aleksandrów Łódzki Klasyki Łódzkie - 1 etap	Aleksandrów Łódzki	6.05.2018	5.05.2018	www.komisjamasters.pl/wyścigi-punktowane-2018	Lista zawodników	
Liczba elementów na stronę 20 1 - 8 / 8 < > >>						

Żeby zarejestrować się na zawody należy uzupełnić następujące pola oznaczone gwiazdkami

Zapisy III Lubański Wyścig MTB

Wszystkie zawody

Lista zawodników

Imię *

Nazwisko *

Płeć *

Data urodzenia *

Miejscowość zamieszkania *

Dystans *

Klub

Adres e-mail

Numer telefonu

I zaznaczyć checkboxy. Na końcu należy kliknąć na przycisk zarejestruj. Może być wymagane wykonanie testu potwierdzającego to, że jest się człowiekiem.



Oświadczam, iż upoważniam Organizatora oraz firmę Time2Win do przetwarzania moich danych osobowych przekazanych przeze mnie przy okazji zgłoszenia udziału w niniejszym wyścigu, w celu jego przeprowadzenia, w celu przekazywania komunikatów informacyjnych związanych z organizacją niniejszego wyścigu, archiwizacji oraz prowadzenia klasyfikacji zawodów odbywających się w ramach niniejszego wyścigu i jeśli dotyczy również w ramach całego cyklu.

Oświadczam także, iż zostakam (-em) pouczon(-a) o prawie wglądu do własnych danych osobowych, prawie żądania ich poprawienia, a także prawie żądania zaprzestania przetwarzania danych osobowych oraz możliwości przekazania moich danych osobowych uprawnionym organom.



Oświadczam, iż zapozna(-em) się z Regulaminem zawodów: *III Lubański Wyścig MTB* dostępnym na stronie internetowej www.kwisaluban.pl, akceptuję jego postanowienia i zobowiązuję się do ich przestrzegania.



Wyrażam zgodę na wykorzystanie przez Organizatora mojego wizerunku, w tym na obrót egzemplarzami, na których utrwalał ten wizerunek, oraz na zwielokrotnienie wizerunku wszelkimi dostępnymi aktualnie technikami i metodami, rozpowszechnianie oraz publikowanie, także wraz z wizerunkami innych osób utwalonymi w ramach wyścigu, materiałach służących popularyzacji działań w zakresie imprez sportowych organizowanych przez Organizatora poprzez rozpowszechnianie wizerunku w:

- mediach elektronicznych, w szczególności na stronach internetowych,
- prasie,
- broszurach, ulotkach, gazetkach itp.

Oświadczam, że wykorzystanie wizerunku zgodnie z niniejszą zgodą nie narusza żadnych dóbr osobistych ani innych praw.

Organizator: Miejsko - Gminny Ludowy Klub Sportowy "Kwisa" Lubań

Zarejestruj się



Można się zarejestrować jako organizator zawodów, w tym celu należy skorzystać z formularza znajdującego się na stronie <https://www.t2w.pl/konto?role=organizer>

[Przejdź do zawodów](#)

Zaloguj się na swoje konto Time2Win.

Adres e-mail *

Hasło *

Zaloguj się

Nie masz konta? Szybko i wygodnie załóż nowe.

Adres e-mail *

Hasło *

Powtórz hasło *

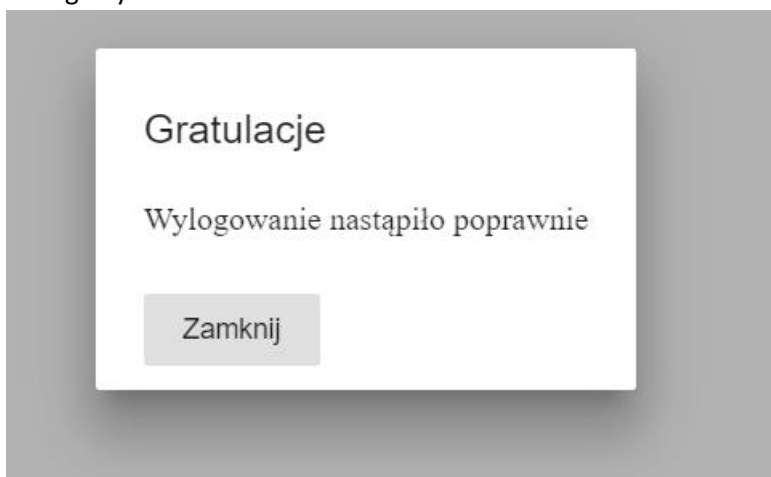
Załącz konto

Po pomyślnym zarejestrowaniu można przypisać organizatora do konkretnych zawodów przy pomocy kontrolera POST `api/OrganizerAccount/1/SetCompetition/1`. Taka operacja może zostać wykonana jedynie przez administratora

Po wybraniu przypisanych zawodów, organizator może zaznaczyć grupowo zawodników których chce mieć opłaconych. Wyświetli mu się wtedy taki widok:

Oznacz wybrane jako opłacone		Oznacz wybrane jako nieopłacone				
<input type="checkbox"/>	Imię	Nazwisko	Płeć	Klub	Kategoria	Oplacony
<input checked="" type="checkbox"/>	ANNA	DOMORADZKA	Kobieta	CROSS TEAM ZGORZELEC	OPEN K3	Tak
<input type="checkbox"/>	Borys	Szachowicz	Kobieta	Kwisa Lubań	OPEN K4	Nie
<input type="checkbox"/>	Borys	Szachowicz	Mężczyzna	Kwisa Lubań	OPEN M4	Nie
<input type="checkbox"/>	Damian	Bahaj	Mężczyzna	Romet Factory Team	OPEN M3	Nie
<input type="checkbox"/>	Dariusz	Paluch	Mężczyzna		OPEN M4	Nie
<input checked="" type="checkbox"/>	Dmytro	Lenets	Mężczyzna	-	OPEN M3	Tak
<input checked="" type="checkbox"/>	Grzegorz	Bujak	Mężczyzna	Jelenia góra	OPEN M4	Tak
<input type="checkbox"/>	Grzegorz	Jakubowski	Mężczyzna		OPEN M3	Nie
<input checked="" type="checkbox"/>	Jan	Czajka	Mężczyzna	Kwisa Lubań	OPEN M3	Tak
<input type="checkbox"/>	Jarosław	Malung	Mężczyzna	Nadleśnictwo Świeradów	OPEN M4	Nie

Po zakończeniu pracy należy wybrać opcję wyloguj. Potwierdzenie pojawi się w oknie dialogowym



Jeśli użytkownik nie ma uprawnień administratora wyświetli mu się inny widok

Lista zawodników
Zawody Integracyjne 20.03.2018

Wszystkie zawody

Zarejestruj się

Imię	Nazwisko	Płeć	Klub	Kategoria	Oplacony
aa	qq	Kobieta			Tak
Adam	aaa	Mężczyzna			Tak
Adam	Adamski	Mężczyzna			Nie
Adam	Borowski	Mężczyzna	ACTIVEXTREME	MEGA Minnys	Tak
Adam	Czekaj	Mężczyzna	OMEGA MIECHÓW	MINI Minnys	Tak
Adam	Dywan	Mężczyzna		MINI Minnys	Tak
Adam	Graczyk	Mężczyzna	PAFANA	MINI Minnys	Tak
Adam	Krawczuk	Mężczyzna	KS Uznam Świnoujście	MINI Minnys	Tak

Żeby móc samemu przetestować działanie strony należy można wejść na stronę www.t2w.pl Jest to jednak wersja produkcyjna i prosilibyśmy się na niej nie rejestrować. Ale listy zawodników są publiczne i jak najbardziej można je obejrzeć. Tak samo jak przetestować pozostałe funkcjonalności aplikacji.

Żeby móc przetestować pełną funkcjonalność aplikacji, możemy wejść na stronę wersji testowej pod adresem: www.testing.time2win.aspnet.pl (Inną możliwością jest wykonanie poleceń 1-7 z działu: *Uruchamianie projektu w Visual Studio*, kończących się *npm start*). Tutaj znajdują się testowe zawody, dla których jak najbardziej można się zalogować. W dodatku, żeby przetestować możliwość operacji możliwych dla konta organizatora, należy zarejestrować się pod linkiem <https://www.testing.time2win.aspnet.pl/konto?role=organizer> Następnie przestać adres e-mail, pod którym zostało założone konto do administratorów, po jeden z poniższych maili tomasz.chudzik.pl@gmail.com krzysiak.grzegorz@gmail.com

Podepniemy wtedy konto organizatora do zawodów testowych i po zalogowaniu będzie można przetestować dodatkowe funkcjonalności.

Instrukcja utrzymania

Dane zawodów są przechowywane w bazie danych MS SQL. Do archiwizacji danych można wykorzystać wbudowane mechanizmy tego systemu.

Raport odstępstw od specyfikacji wymagań

W trakcie realizacji projektu wprowadzonych zostało kilka zmian, które powodują, że część funkcjonalności odbiega od tych zawartych w specyfikacji wstępnej. Zmiany te powstawały podczas bieżących konsultacji z klientem. Często były to sugestie od zamawiającego, który zmieniał swoje wymagania. Niemniej jednak nie były to znaczące modyfikacje. Zdarzało się również, że na wniosek autorów projektu, niektóre zmiany funkcjonalności były zgłaszane klientowi i po jego akceptacji uwzględniane podczas wykonywania projektu. Odstępstwa od specyfikacji wymagań zostało zaprezentowane poniżej:

- Poza samymi controllerami do obsługi kont użytkowników oraz do zapisów powstały również controllery obsługujące konta administratora, organizatora oraz moderatora. Dzięki nim istnieje możliwość stworzenia aplikacji webowej, która pozwoli wykonać część takich samych czynności jak w aplikacji desktopowej stworzonej w pierwszym etapie. Funkcjonalności te nie będą raczej realizowane nawet w trakcie trzeciego etapu, niemniej jednak zostaną one wykorzystane podczas tworzenia ostatecznego produktu dla klienta
- W porównaniu ze specyfikacją wstępną zdecydowano, że podczas zapisów to zawodnik sam decyduje na jakim dystansie będzie odbywał zawody i która z dodatkowych informacji będzie dla niego odpowiednia. Możliwość dokonania tego wyboru wprowadzono podczas zapisów i podczas edycji danych dla zawodów, na które zawodnik już się zapisał
-

Dokumentacja utworzonego Web API

Strona: <http://projektnet.mini.pw.edu.pl/NTime>

Administrator:

Login/Email: admin@ntime.pl

Hasło: admin1

Założenia:

- Są 4 role:
 - Player - P
 - Organizer - O
 - Moderator - M
 - BustModerator – Moderator z odebranymi uprawnieniami
 - Administrator - A
- Organizator ma dostęp do jakiegoś zasobu związanego z zawodami tylko wtedy, gdy jest powiązany z danymi zawodami i może edytować je jak flaga w zawodach "ModeratorEditLock" jest na false
- Player może zapisywać się na zawody i modyfikować owy wpis o ile operacje te nie są wykonywane po "SignUpEndDate" w obiekcie zawodów
- Player może mieć pełny wgląd tylko na własne wyniki
- Jeśli w ścieżce jest napisane "1" to oznacza, że trzeba tam podać id do danego zasobu (liczba dodatnia, całkowita), chyba że jest poprzedzony przez "/FromCompetition/", "/IntoCompetition/", "/ByCompetition/", "/SetCompetition/" lub "/UnsetCompetition/" wtedy należy podać id do zawodów, a jeśli "/FromPlayerAccount/" to podajemy id do konta zawodnika.
- W przypadku kontrolerów "AdministratorAccount", "ModeratorAccount" id to ciąg znaków
- Api obsługuje Json o ile Content-Type jest nie wskazuje inaczej
- Jeśli adres kończy się na "?ItemsOnPage=10&PageNumber=0" to oznacza, że wynik jest stronicowany, w odpowiedzi zostanie się całkowitą liczbę obiektów i tablicę przeznaczoną do wyświetlenia na wskazanej stronie (Uwaga: strony liczymy od zera). Zwracana jest struktura z polem TotalCount – int i Items – tablica.
- Hasła powinny zawierać co najmniej 8 znaków w tym co najmniej jedną cyfrę i mały znak
- Można ignorować pole id przy tworzeniu nowych zasobów, ale nie można go pominąć, może być dowolne a najlepiej –1
- Jeśli przy polu w wypisanej strukturze danych jest "*" to pole nie może przyjąć null-a, więc nie może być pominięte
- 404 jeśli id-ki się nie zgadzają
- Czas liczymy w sekundach jako zmienna typu decimal (oznaczone przy niektórych zmiennych)

Logowanie:

POST Token

Mogą: wszyscy

Content-Type: application/x-www-form-urlencoded

Body: grant_type=password&username=admin&password=admin1

Przykładowa odpowiedź:

{

```
"access_token": "BMxniHiUjhexqWm1RwexXkAK9Pq70n4-
QZNdjRgle1gL15a1mrGLM3IHhjQ0DOCaWsGh0T2Nr9KWqStfQuPxD8iYDDYDKzRbU2Rv_sXPeEbBDPp
vgAVsOnL9aiWZsG2ve811uGBiH0l0Nfko50uQDQcMAuHAbBQjXuwlChO-up5DTLgrHozllc8han4U-
t1nDWeiVd1vPqPcaJi6TvFm6BfNGjZhrm8G2xAoWF2C60894Qjs-
VqWJIER9eSBLk6B6TfPFHrEpwHixf5oIC2DQjpv1Jd-3PFNu2RQTnlbqMCI5pjJXasYGuzPGWW5n8J58D-
tC7lxOYXqIKZTHfGKzAE2oYsN8kMxmC0QCc0tqgnBJvbGiJ-w0t8JktmicFKpJR0zUIC3YLQryPx-
HvcvLnaiwXpenVATYosSDoCmvfT6kpy4Sg34qQYYw7dKf8RvJraxbnlqYmJqrZ6L9uD9oQ6CHHAj7Khsfa
h6KXSSexWZvL4S3vE20-TEIfMiC09UaSzANhDaYXIAMB7_gJo0g",

"token_type": "bearer",

"expires_in": 1209599,

"userName": "admin",

".issued": "Thu, 28 Dec 2017 20:42:00 GMT",

".expires": "Thu, 11 Jan 2018 20:42:00 GMT"

}
```

W każdym następnym żądaniu należy umieścić w nagłówku:

Authorization: Bearer BMxniHiUjhexqWm1Rwe....

Kontroler Account:

POST api/Account/Logout

Mogą: wszyscy zalogowani

POST api/Account/ChangePassword

Mogą: wszyscy zalogowani

Body:

- *OldPassword
- *NewPassword
- *ConfirmPassword

POST api/Account/Register

Mogą: wszyscy

Body:

- *Email
- *Password
- *ConfirmPassword

GET api/Account/Role

Mogą: wszyscy zalogowani

Zwraca id konta (string), email i przypisaną rolę

Kontroler Competition

Model danych

- *Id
- *Name
- *EventDate
- SignUpEndDate - jeśli null to nie ma deadline-u na zapisy
- Description
- Link
- Organizer - string
- City
- *OrganizerEditLock – nie da się zmodyfikować to poprzez PUT i POST (przy tworzeniu), po stworzeniu nowych zawodów ustawiony jest na false

GET /api/Competition?ItemsOnPage=10&PageNumber=0

Mogą: wszyscy

Pobiera stronicowaną listę zawodów sortowaną po czasie rozpoczęcia

GET /api/Competition/FromPlayerAccount/1?ItemsOnPage=10&PageNumber=0

Mogą: A, M, P

Pobiera stronicowaną listę zawodów, na które dany użytkownik jest zapisany.

GET /api/Competition/1

Mogą: wszyscy

Pobiera dane jednych zawodów

PUT /api/Competition/1

Mogą: A, O

Modyfikuje zawody

POST /api/Competition

Mogą: A

Tworzy nowe zawody

POST /api/Competition/1/OrganizerLock/true

POST /api/Competition/1/OrganizerLock/false

Mogą: A

Nic nie trzeba podawać nic w body

Ustawia flagę OrganizerEditLock

Kontrolery AgeCategory, Distance, ExtraPlayerInfo

W miejsce "<attr>" wstaw AgeCategory, Distance lub ExtraPlayerInfo

Model danych AgeCategory

- *Id
- *Name
- *YearFrom

- *YearTo

Model danych Distance

- *Id
- *Name
- *Length
- *DistanceTypeId – int: może przyjąć
 - 0 – na dystans
 - 1 – na okrążenia - wtedy LapsCount jest istotne
 - 2 – na czas w pętli - wtedy TimeLimit jest istotne
- *LapsCount
- *TimeLimit - decimal

Model danych ExtraPlayerInfo

- *Id
- *Name
- *ShortName

GET api/<attr>/FromCompetition/1

Mogą: wszyscy

Wylistuje dane z danych zawodów

GET api/<attr>/1

Mogą: wszyscy

Pobiera obiekt o podanym id

PUT api/<attr>/1

Mogą: A, O

Modyfikuje obiekt o podanym id

POST api/<attr>/IntoCompetition/1

Mogą: A, O

Tworzy nowy obiekt w danych zawodach

DELETE api/<attr>/1

Usuwa obiekt o podanym id

Mogą: A

Kontroler Player

Dotyczy "wpisu" na konkretne zawody i ich ewentualne wyniki, a nie konta zawodników

Model danych filtru

Jeśli jest nullem to filtr nie jest brany pod uwagę

- *PlayerSort - int
 - 0 - imię
 - 1 – nazwisko
 - 2 – klub

- 3 – numer startowy
 - 4 – czas startu
 - 5 – kategoria
 - 6 – data urodzenia
- *DescendingSort - bool
- Query – string, zaawansowane wyszukiwanie po napisie
 - "30 - 50" wybierze numery startowe z przedziału
 - "30, 40, 45, 50" wybierze podane numery startowe
 - "jan kow MINI" wybierze wpisy, w których zawierają się podane po spacji selektory
np. Jan Kowalski z wystansu MINI i Jan Kowalik z wystansu MINI+
- Men – bool
- WithoutStartTime - bool
- Invalid - bool
- CompletedCompetition - bool
- HasVoid - bool
- Distance – id
- AgeCategory – id
- ExtraPlayerInfo – id

Model danych do publicznej listy (tylko do wglądu)

- *Id
- *FirstName
- *LastName
- *IsMale - bool
- Team - string
- *StartNumber - int
- StartTime – decimal (liczone od godziny 00:00)
- FullCategory - string

Model danych do zapisów

- *Id
- *FirstName
- *LastName
- *BirthDate
- *IsMale
- Team
- *ExtraPlayerInfoId - id
- *DistanceId - id
- *CompetitionId -id

Model danych pełny

- *Id
- *FirstName
- *LastName
- *BirthDate
- *IsMale
- Team

- *StartNumber
- StartTime – decimal (liczone od godziny 00:00)
- *IsStartTimeFromReader - być może zostanie usunięte
- FullCategory – aktualizuje się automatycznie
- *LapsCount
- *Time - decimal
- *DistancePlaceNumber
- *CategoryPlaceNumber
- *CompetitionCompleted - bool
- ExtraPlayerInfold - id
- Distanceld - id
- CompetitionId - id
- PlayerAccountId - id

POST [api/Player/TakeSimpleList/FromCompetition/1?ItemsOnPage=10&PageNumber=0](#)

Mogą: wszyscy

W body podać obiekt filtru

Zwraca dane publicznej listy

POST [api/Player/TakeFullList/FromCompetition/1?ItemsOnPage=10&PageNumber=0](#)

Mogą: A, O, M

W body podać obiekt filtru

Zwraca pełne dane

GET [api/Player/TakeFullList/FromPlayerAccount/1?ItemsOnPage=10&PageNumber=0](#)

Mogą: A, M, P

Zwraca zapisy (pełne dane) z odbytych zawodów i zawodów na które się zapisało podanego zawodnika

GET [api/Player/FromPlayerAccount/1/FromCompetition/1](#)

Mogą: A, M, P

Zwraca pełne dane zarejestrowane użytkownika dla wybranych zawodów

GET [api/Player/1](#)

Mogą: A, O, M, P

Zwraca pełne dane o podanym id

PUT [api/Player/1](#)

Mogą: A, O, M

Modyfikuje pełne dane

DELETE [api/Player/1](#)

Mogą: A, O, M, P

Kasuje zapis

GET [api/Player/Register/1](#)

Mogą: A, O, M, P

Pobiera zapis w o podanym id

PUT [api/Player/Register/1](#)

Mogą: P

Modyfikuje zapis

POST [api/Player/Register/IntoCompetition/1](#)

Mogą: wszyscy

Zapisuje się na dane zawody

[Kontroler PlayerAccount](#)

Model Danych

- *Id
- *FirstName
- *LastName
- *BirthDate
- *IsMale
- Team
- PhoneNumber
- *EMail

GET [api/PlayerAccount/Search/Filtr?ItemsOnPage=10&PageNumber=0](#)

GET [api/PlayerAccount/Search?ItemsOnPage=10&PageNumber=0](#)

Może: A

Zwraca stronicowaną listę kont zawodników z lub bez filtru ("Filtr") który jest stringiem

GET [api/PlayerAccount](#)

Może: P

Zwraca dane aktualnie zalogowanego zawodnika

GET [api/PlayerAccount/1](#)

Może: A, P

Zwraca dane zawodnika o podanym id

GET [api/PlayerAccount/FromCompetition/1?ItemsOnPage=10&PageNumber=0](#)

Może: A

Zwraca listę kont zawodników z których zapisano się na konkretne zawody

PUT [api/PlayerAccount/1](#)

Może: A, P

Edycja

Kontroler OrganizerAccount

Wszystkie operacje może tylko administrator

Model Danych

- *Id
- *FirstName
- *LastName
- *PhoneNumber - string
- *EMail
- CompetitionDtos – tablica zawodów do których jest przypisanych (model danych zawodów), przy wylistowaniu null, pole pomijane przy dodawaniu i modyfikacjach

GET api/OrganizerAccount/Search/Filtr?ItemsOnPage=10&PageNumber=0

GET api/OrganizerAccount/Search?ItemsOnPage=10&PageNumber=0

Zwraca stronicowaną listę organizatorów z lub bez filtru ("Filtr") który jest stringiem

GET api/OrganizerAccount/ByCompetition/1

Zwraca organizatorów przypisanych do danych zawodów

GET api/OrganizerAccount/1

Pojedynczy organizator o podanym id, z polem CompetitionDtos

PUT api/OrganizerAccount/1

Edycja

POST api/OrganizerAccount

Stworzenie nowego organizatora z losowym hasłem. Hasło nie jest zwracane.

DELETE api/OrganizerAccount/1

Usunięcie organizatora

POST api/OrganizerAccount/1/PasswordReset

Reset hasła. Nowe, losowe hasło jest zwracane.

POST api/OrganizerAccount/1/SetCompetition/1

Ustawia organizatorowi o podanym id dostęp do zawodów o podanym id

POST api/OrganizerAccount/1/UnsetCompetition/1

Analogicznie

Kontroler ModeratorAccount

Wszystkie operacje może tylko administrator

Model danych taki jak w akcji "Role"

Id to string, (w prototypach oznaczony jako "aaa")

GET api/ModeratorAccount

Pobiera listę

POST api/ModeratorAccount

Tworzy nowe kont,

W body to samo jak przy rejestracji

DELETE api/ModeratorAccount/aaa

Usuwa

POST api/ModeratorAccount/aaa/PasswordReset

Zmienia hasło na losowe i je zwraca

POST api/ModeratorAccount/aaa/Bust

Zmienia rolę na "BustModerator"

POST api/ModeratorAccount/aaa/Unbust

Analogicznie

Kontroler AdministratorAccount

Wszystkie operacje może tylko administrator

Analogicznie jak w Moderatorze

GET api/AdministratorAccount

POST api/AdministratorAccount

DELETE api/AdministratorAccount/aaa

POST api/AdministratorAccount/aaa/PasswordReset

Dokumentacja końcowa (powykonawcza) - punkty wymagane przez prowadzącego zajęcia

