



STOPER PROGRAMISTY

SPIS TREŚCI

1.SKŁAD GRUPY.....	3
2.DOKUMENTACJA DLA KLIENTA	4-16
3.TECHNOLOGIE UŻYTE W PROJEKCIE.....	17-20
4.PRZYPADKI UŻYCIA.....	21-30
5.SCENARIUSZE TESTOWE.....	31-43
6.TESTY FUNKCJONALNE I NIEFUNKCJONALNE.....	44-56
7.ARCHITEKTURA SYSTEMU.....	57-64
8.API.....	65-69
9.INFRASTRUKTURA.....	70-73
10.DOKUMENTACJA KODU.....	74-78

SKŁAD GRUPY

Kierownik Projektu: Maksym Kokociński

Inżynier Procesu: Piotr Wrzodak

Analityk Systemowy: Krzysztof Małagowski,
Dominik Zawadzki

Architekt Systemu: Paweł Lewicki

Integrator: Mateusz Szczeciński

Administrator Systemu: Tomasz Dzierzbicki

Menadżer Testów: Łukasz Mioduszeński

Projektant GUI: Sebastian Jarowicki

Przegląd Projektu: Paweł Grzonka

Programista: Jędrzej Kupcewicz, Przemek Kluska

DOKUMENTACJA DLA KLIENTA

Dokument opisuje ogólne założenia i cele projektu, a także użytkowników i ich role. Na końcu znajduje się rysunek pokazujący rozmieszczenie interfejsu.

SPIS TREŚCI

Założenie i cel projektu (str. 5)

Użytkownicy (str. 6)

Instrukcja Obsługi(str. 7-16)

ZAŁOŻENIE I CEL PROJEKTU

Stoper Programisty jest aplikacją webową.

Jego celem jest rejestrowanie czasów pracy odpowiednich użytkowników.

Dostępna jest pula zadań z różną ilością czasów (w sekundach) ich wykonania, zalogowany użytkownik zaczyna jedno zadanie przez wciśnięcie przycisku "Start", a program zaczyna mierzyć czas. Dana praca znika wtedy z ogólnodostępnej puli i jest przypisana do określonego loginu. Zadanie można w dowolnym momencie zatrzymać i wznowić po jakimś czasie, program wtedy nie bierze pod uwagę tej przerwy. Można mieć kilka zaczętych zadań, ale w danym momencie może być wykonywane tylko jedno zadanie. W wypadku wylogowania czas zadania dalej jest mierzony!

Dana praca definitywnie kończy się, kiedy użytkownik naciśnie odpowiedni przycisk. Przechodzi ona wtedy do puli zakończonych zadań, kolorowana na zielono jeśli zostało wykonana w czasie lub szybciej, na czerwono jeśli przekroczono dozwolony czas.

Zadaniem Stopera jest obliczenie różnicy czasów ustalonych a faktycznych i dodanie go do odpowiedniego parametru.

UŻYTKOWNICY

W projekcie mamy dwa typy użytkowników:

1. Administrator - zarządza zadaniami, może dodawać nowe, usuwać już istniejące lub modyfikować czasy
2. Zwykły użytkownik - wykonuje zadania z puli, posiada parametr określający sumę różnic między czasami faktycznymi a rzeczywistymi

INSTRUKCJA OBSŁUGI

Dokumentacja składa się z części merytorycznej oraz nawigacyjnej. Dzięki tej formie użytkownik jest w stanie sprawnie obsługiwać się programem oraz zna działanie wszystkich funkcjonalności programu.

• Każda z części posiada własną instrukcję obsługi przedstawioną za pomocą poniższych podpunktów:

1. Widok wstępny – opis zawierający wygląd interfejsu widzianego przez klienta oraz opis poszczególnych przycisków
2. Założenia – informacje dotyczące celu oraz przebiegu danego zadania
3. Lista kroków – kolejne etapy, które wykonać musi użytkownik w celu zrealizowania zadania oraz uzyskania pożądanego efektu końcowego
4. Rezultat – przedstawienie widoku interfejsu po zakończeniu wykonywanego zadania oraz informacje na temat wyniku końcowego

SPIS TREŚCI

Dodanie nowego
zadania (str. 5)

Edycja istniejącego
zadania (str. 6)

Usuwanie istniejącego
zadania (str. 7)

Wylogowanie się z
aplikacji (str. 8)

Zalogowanie do
aplikacji (str. 9)

Rozpoczęcie zadania (str.
10)

Przerwanie zadania (str.11)

Wznowienie zadania (str.
12)

Zakończenie zadania (str.
13)

1. DODANIE NOWEGO ZADANIA

Widok wstępny:

System wyświetla interfejs poziomu admina zawierający pola

- Zadanie - określenie nazwy zadania
- Czas - określenie czasu na wykonanie zadania
- Dodaj – zatwierdza parametry zadania

Założenia:

Administrator dodaje zadanie do wykonania przez użytkowników o wyznaczonym czasie.

Lista Kroków:

1. Administrator wypełnia pola i zatwierdza przyciskiem

dodaj. Scenariusz alternatywny:

- Administrator wpisuje w pole 'Czas' nieprawidłowe znaki (różne niż liczby bądź 0)
- System zwraca informację o błędnej wartości w polu.

Rezultat:

Zadanie zostaje dodane.

2. EDYCJA ISTNIEJĄCEGO ZADANIA

Widok wstępny:

System wyświetla interfejs poziomu admina.

Założenia:

Administrator edytuje zadanie do wykonania przez użytkowników.

Lista Kroków:

1. Administrator inicjuje zmianę klikając przycisk edytuj.
2. Zmiana nazwy zadania bądź jego czasu.

Scenariusz alternatywny:

- Administrator wpisuje w pole 'Czas' nieprawidłowe znaki (różne niż liczby bądź 0)
- System zwraca informację o błędnej wartości w polu.

Rezultat:

- Zadanie zostaje edytowane.

3. USUWANIE ISTNIEJĄCEGO ZADANIA

Widok wstępny:

System wyświetla interfejs poziomu admina.

Założenia:

Administrator usuwa zadanie do wykonania przez użytkowników.

Lista Kroków:

1. Administrator inicjuje proces klikając przycisk usuń.
2. Administrator potwierdza usunięcie zadania.

Rezultat:

- Zadanie zostaje usunięte

4. WYLOGOWANIE SIĘ Z APLIKACJI

Widok wstępny:

System wyświetla interfejs poziomu admina bądź użytkownika.

Założenia:

Użytkownik wyloguje się z aplikacji

Lista Kroków:

1. Użytkownik inicjuje proces klikając przycisk wyloguj.
2. Użytkownik potwierdza wylogowanie z systemu.

Rezultat:

- Użytkownik zostaje wylogowany z aplikacji.

5. ZALOGOWANIE DO APLIKACJI

Widok wstępny:

System wyświetla stronę logowania do aplikacji

Założenia:

Użytkownik loguje się do systemu

Lista Kroków:

1. Użytkownik wpisuje login oraz hasło
2. Użytkownik klika przycisk zaloguj.

Scenariusz alternatywny:

- Użytkownik błędnie wpisał login lub hasło
- System zwraca informację o błędnym loginie lub hasle.

Rezultat:

- Użytkownik zostaje zalogowany do systemu.

6. ROZPOCZĘCIE ZADANIA

Widok wstępny:

System wyświetla interfejs poziomu użytkownika.

Założenia:

Użytkownik rozpoczyna zadanie.

Lista Kroków:

1. Użytkownik klika przycisk start aby rozpocząć dane zadanie.

Rezultat:

- Użytkownik rozpoczyna dane zadanie
- Zadanie trafia do puli wykonywanych.

7. PRZERWANIE ZADANIA

Widok wstępny:

System wyświetla interfejs poziomu użytkownika.

Założenia:

Użytkownik przerywa zadanie.

Lista Kroków:

1. Użytkownik klika przycisk przerwij aby przerwać dane zadanie. Rezultat:

- Użytkownik przerywa dane zadanie.
- Zadanie trafia do puli wykonywanych i zmienia kolor na czerwony

8. WZNOWIENIE ZADANIA

Widok wstępny:

System wyświetla interfejs poziomu użytkownika.

Założenia:

Użytkownik wznowia przerwane zadanie.

Lista Kroków:

1. Użytkownik klika przycisk wznów aby wznowić przerwane zadanie.

- Użytkownik wznowia dane zadanie.
- Zadanie trafia do puli wykonywanych.

9. ZAKOŃCZENIE ZADANIA

Widok wstępny:

System wyświetla interfejs poziomu użytkownika.

Założenia:

Użytkownik zakańcza wykonywane zadanie.

Lista Kroków:

1. Użytkownik klika przycisk zakończ aby zakończyć wykonywane

Rezultat:

- Użytkownik zakańcza wykonywanie zadania.
- Zadanie trafia do puli wykonywanych z kolorem:
 - czerwonym – jeśli wykonanie zadania zajęło więcej czasu niż wyznaczył admin
 - zielone – jeśli wykonanie zadania zmieściło się w ramach czasowych.

TECHNOLOGIE UŻYTE W PROJEKCIE

Dokument zawiera informacje na temat technologii oraz programów pomocniczych użytych w projekcie oraz uzasadnienie ich wyboru.

SPIS TREŚCI

Język(str. 18)

Technologia(str. 19)

Baza Danych(str. 20)

JĘZYK

- Językiem programowania, którym będziemy posługiwać się wykonując projekt jest **c#** od firmy **Microsoft**.
- Uzasadnienie wyboru:
 1. Jest to język hierarchiczny wykorzystujący obiektowość, który jest powszechnie używany przy projektach webowych
 2. Zawiera bogatą bibliotekę klas oraz jest przejrzysty co poprawia komfort pracy
 3. Część naszego zespołu odpowiedzialna za kwestie programistyczne na codzień ma styczność właśnie z tym językiem co przyspieszy wykonywanie projektu

TECHNOLOGIA

- Technologia, za pomocą której będziemy tworzyć projekt:

ASP .NET CORE

- Uzasadnienie wyboru:
 1. Wieloplatformowość – ten framework pozwala na korzystanie z aplikacji na różnych platformach – nie tylko Windows ale także Linux czy też macOS
 2. Wysoka wydajność (możliwość tworzenia setek mikrousług)
 3. .NET Core pozwala na obsługę z poziomu linii poleceń co preferuje spora część programistów
- Pracować będziemy na wzorcu **MVC** (Model-View-Controller) czyli na podziale aplikacji na trzy części:
 1. (Model) - **Model danych**- opis struktur danych i powiązań między nimi
 2. (View) - **Interfejs**, czyli to co widzi użytkownik
 3. (Controller) - **Logika działania** – powiązania między zdarzeniami zachodzącymi w systemie

BAZA DANYCH

Baza danych jest podzielona na dwie tabele, z których pierwsza zawiera podstawowe informacje o zadaniu, a druga służy do obsługi czasu zadania.

W tabeli tasks dane są pola:

- id będące kluczem głównym,
- name (string) oznaczające nazwę zadania,
- time_admin (int) oznaczające czas na wykonanie zadania zadeklarowany przez administratora przy jego tworzeniu
- time_user (int) będące sumą interwałów od rozpoczęcia do zatrzymania lub zakończenia zadania (wartość pojedynczego interwału jest wyznaczana w tabeli holdTime)
- difference (int) oznaczające różnicę pomiędzy wartościami time_admin i time_user i wyznaczające terminowość wykonania zadania
- stan (bool) wskazujący na status wykonania zadania

Natomiast w tabeli holdTime dane są pola:

- id będące kluczem głównym tabeli,
- time_start (int) oznaczające początek interwału (rozpoczęcia lub wznowienia zadania)
- time_end (int) oznaczające zakończenie interwału (zakończenie lub zatrzymanie zadania)
- time_difference (int) oznaczające różnicę pomiędzy time_end a time_start i będące podstawą do przepisania do tasks.time_user

PRZYPADKI UŻYCIA

Dokumentacja składa się z części merytorycznej oraz nawigacyjnej. Dzięki tej formie użytkownik jest w stanie sprawnie obsługiwać się programem oraz zna działanie poszczególnych przypadków użycia.

• Każdy z przypadków posiada własną instrukcję obsługi przedstawioną za pomocą poniższych podpunktów:

1. Widok wstępny – opis zawierający wygląd interfejsu widzianego przez klienta oraz opis poszczególnych przycisków
2. Założenia – informacje dotyczące celu oraz przebiegu danego zadania
3. Lista kroków – kolejne etapy, które wykonać musi użytkownik w celu zrealizowania zadania oraz uzyskania pożądanego efektu końcowego
4. Rezultat – przedstawienie widoku interfejsu po zakończeniu wykonywanego zadania oraz informacje na temat wyniku końcowego

SPIS TREŚCI

Przypadek 1 (str. 22)

Przypadek 2 (str. 23)

Przypadek 3 (str. 24)

Przypadek 4 (str. 25)

Przypadek 5 (str. 26)

Przypadek 6 (str. 27)

Przypadek 7 (str. 28)

Przypadek 8 (str. 29)

Przypadek 9 (str. 30)

PRZYPADEK 1 (DODANIE NOWEGO ZADANIA)

Widok wstępny:

System wyświetla interfejs poziomu admina zawierający pola

- Zadanie - określenie nazwy zadania
- Czas - określenie czasu na wykonanie zadania
- Dodaj – zatwierdza parametry zadania

Założenia:

Administrator dodaje zadanie do wykonania przez użytkowników o wyznaczonym czasie.

Lista Kroków:

1. Administrator wypełnia pola i zatwierdza przyciskiem

dodaj. Scenariusz alternatywny:

- Administrator wpisuje w pole 'Czas' nieprawidłowe znaki (różne niż liczby bądź 0)
- System zwraca informację o błędnej wartości w polu.

Rezultat:

Zadanie zostaje dodane.

PRZYPADEK 2 (EDYCJA ISTNIEJĄCEGO ZADANIA)

Widok wstępny:

System wyświetla interfejs poziomu admina.

Założenia:

Administrator edytuje zadanie do wykonania przez użytkowników.

Lista Kroków:

1. Administrator inicjuje zmianę klikając przycisk edytuj.
2. Zmiana nazwy zadania bądź jego czasu.

Scenariusz alternatywny:

- Administrator wpisuje w pole 'Czas' nieprawidłowe znaki (różne niż liczby bądź 0)
- System zwraca informację o błędnej wartości w polu.

Rezultat:

- Zadanie zostaje edytowane.

PRZYPADEK 3 (USUWANIE INSTIEJĄCEGO ZADANIA)

Widok wstępny:

System wyświetla interfejs poziomu admina.

Założenia:

Administrator usuwa zadanie do wykonania przez użytkowników.

Lista Kroków:

1. Administrator inicjuje proces klikając przycisk usuń.
2. Administrator potwierdza usunięcie zadania.

Rezultat:

- Zadanie zostaje usunięte

PRZYPADEK 4 (WYLOGOWANIE SIĘ Z APLIKACJI)

Widok wstępny:

System wyświetla interfejs poziomu admina bądź użytkownika.

Założenia:

Użytkownik wyloguje się z aplikacji

Lista Kroków:

1. Użytkownik inicjuje proces klikając przycisk wyloguj.
2. Użytkownik potwierdza wylogowanie z systemu.

Rezultat:

- Użytkownik zostaje wylogowany z aplikacji.

PRZYPADEK 5 (ZALOGOWANIE DO APLIKACJI)

Widok wstępny:

System wyświetla stronę logowania do aplikacji

Założenia:

Użytkownik loguje się do systemu

Lista Kroków:

1. Użytkownik wpisuje login oraz hasło
2. Użytkownik klika przycisk zaloguj.

Scenariusz alternatywny:

- Użytkownik błędnie wpisał login lub hasło
- System zwraca informację o błędnym loginie lub hasle.

Rezultat:

- Użytkownik zostaje zalogowany do systemu.

PRZYPADEK 6 (ROZPOCZĘCIE ZADANIA)

Widok wstępny:

System wyświetla interfejs poziomu użytkownika.

Założenia:

Użytkownik rozpoczyna zadanie.

Lista Kroków:

1. Użytkownik klika przycisk start aby rozpocząć dane zadanie.

Rezultat:

- Użytkownik rozpoczyna dane zadanie
- Zadanie trafia do puli wykonywanych.

PRZYPADEK 7 (PRZERWANIE ZADANIA)

Widok wstępny:

System wyświetla interfejs poziomu użytkownika.

Założenia:

Użytkownik przerywa zadanie.

Lista Kroków:

1. Użytkownik klika przycisk przerwij aby przerwać dane zadanie. Rezultat:

- Użytkownik przerywa dane zadanie.
- Zadanie trafia do puli wykonywanych i zmienia kolor na czerwony

PRZYPADEK 8 (WZNOWIENIE ZADANIA)

Widok wstępny:

System wyświetla interfejs poziomu użytkownika.

Założenia:

Użytkownik wznowia przerwane zadanie.

Lista Kroków:

1. Użytkownik klika przycisk wznów aby wznowić przerwane zadanie.

- Użytkownik wznowia dane zadanie.
- Zadanie trafia do puli wykonywanych.

PRZYPADEK 9 (ZAKOŃCZENIE ZADANIA)

Widok wstępny:

System wyświetla interfejs poziomu użytkownika.

Założenia:

Użytkownik zakańcza wykonywane zadanie.

Lista Kroków:

1. Użytkownik klika przycisk zakończ aby zakończyć wykonywane

Rezultat:

- Użytkownik zakańcza wykonywanie zadania.
- Zadanie trafia do puli wykonywanych z kolorem:
 - czerwonym – jeśli wykonanie zadania zajęło więcej czasu niż wyznaczył admin
 - zielone – jeśli wykonanie zadania zmieściło się w ramach czasowych.

SCENARIUSZE

TESTOWE

Dokument opisuje przewidywane scenariusze działania programu, przypadki użycia oraz to, czego można oczekiwać podczas jego użytkowania.

SPIS TREŚCI

Od strony
Administratora (str. 32-
37)

Od strony Użytkownika
(str. 38-41)

Przykładowe zestawy
danych (str. 42-43)

OD STRONY ADMINISTRATORA

Scenariusz testowy nr 1.

Dodanie nowego zadania bez wypełnienia pól „Zadanie” i/lub „Czas”.

Kroki:

1. Zalogować się (login: admin, hasło: admin123).
2. Kliknąć przycisk „Dodaj”.

Oczekiwany wynik:

System wyświetla okienko, w którym informuje Administratora o niewypełnieniu pola „Zadanie” i/lub „Czas”.

Scenariusz testowy nr 2.

Dodanie nowego zadania z nazwą, która już jest używana.

Kroki:

1. Zalogować się (login: admin, hasło: admin123).
2. Wypełnić pole „Zadanie” z istniejącą już w systemie nazwą.
3. Wypełnić pole „Czas”.
4. Kliknąć przycisk „Dodaj”.

Oczekiwany wynik:

System wyświetla okienko, w którym informuje Administratora o konieczności zmiany nazwy dodawanego zadania.

Scenariusz testowy nr 3.

Dodanie nowego zadania z nieprawidłowym wypełnieniem pola „Czas”.

Kroki:

1. Zalogować się (login: admin, hasło: admin123).
2. Wypełnić pole „Zadanie”.
3. Wypełnić pole „Czas” np. literami.
4. Kliknąć przycisk „Dodaj”.

Oczekiwany wynik:

System wyświetla okienko, w którym informuje Administratora o użyciu niedozwolonych znaków w polu „Czas”.

Scenariusz testowy nr 4.

Dodanie nowego zadania z prawidłowym wypełnieniem pól „Zadanie” oraz „Czas”.

Kroki:

1. Zalogować się (login: admin, hasło: admin123).
2. Wypełnić pole „Zadanie”.
3. Wypełnić pole „Czas”
4. Kliknąć przycisk „Dodaj”.

Oczekiwany wynik:

Zadanie pojawia się w tabeli „Zadania”.

Scenariusz testowy nr 5.

Edycja istniejącego zadania bez wprowadzenia żadnych zmian.

Kroki:

1. Zalogować się (login: admin, hasło: admin123).
2. Kliknąć przycisk „Edytuj”.
3. Kliknąć przycisk „Akceptuj”.

Oczekiwany wynik:

1. Pojawia się okienko, w którym można edytować pole „Zadanie” oraz „Czas”.
2. Po kliknięciu „Akceptuj” system wyświetla okno, w którym wyświetlany jest komunikat o braku dokonanych zmian oraz zapytanie, czy chcemy kontynuować.

Scenariusz testowy nr 6.

Edycja istniejącego zadania z nieprawidłowym wypełnieniem pola „Zadanie” (wpisanie używanej już nazwy).

Kroki:

1. Zalogować się (login: admin, hasło: admin123).
2. Kliknąć przycisk „Edytuj”.
3. Wypełnić pole „Zadanie” istniejącą już w systemie nazwą.
4. Kliknąć przycisk „Akceptuj”.

Oczekiwany wynik:

System wyświetla okienko, w którym informuje Administratora o braku możliwości użycia danej nazwy.

Scenariusz testowy nr 7.

Edycja istniejącego zadania z nieprawidłowym wypełnieniem pola „Czas”.

Kroki:

1. Zalogować się (login: admin, hasło: admin123).
2. Kliknąć przycisk „Edytuj”.
3. Wypełnić pole „Czas” np. literami.
4. Kliknąć przycisk „Akceptuj”.

Oczekiwany wynik:

System wyświetla okienko, w którym informuje Administratora o użyciu niedozwolonych znaków w polu „Czas”.

Scenariusz testowy nr 8.

Edycja istniejącego zadania z prawidłowym wypełnieniem pola „Zadanie” i/lub „Czas”.

Kroki:

1. Zalogować się (login: admin, hasło: admin123).
2. Kliknąć przycisk „Edytuj”.
3. Wypełnić pole „Zadanie”.
4. Wypełnić pole „Czas”.
5. Kliknąć przycisk „Akceptuj”.

Oczekiwany wynik:

Zadanie zostaje zaktualizowane.

Scenariusz testowy nr 9.

Usuwanie danego zadania (z potwierdzeniem).

Kroki:

1. Zalogować się (login: admin, hasło: admin123).
2. Kliknąć przycisk „Usuń”.

Oczekiwany wynik:

System wyświetla okienko potwierdzenia, w którym mamy możliwość kliknięcia „Tak” lub „Nie”.

Zadanie zostaje usunięte z tabeli „Zadania”.

Scenariusz testowy nr 10.

Usuwanie danego zadania (bez potwierdzenia).

Kroki:

1. Zalogować się (login: admin, hasło: admin123).
2. Kliknąć przycisk „Usuń”.

Oczekiwany wynik:

System wyświetla okienko potwierdzenia, w którym mamy możliwość kliknięcia „Tak” lub „Nie”.

Zadanie wciąż jest w tabeli „Zadania”.

Scenariusz testowy nr 11.

Wylogowywanie się (z potwierdzeniem).

Kroki:

1. Zalogować się (login: admin, hasło: admin123).
2. Kliknąć przycisk „Wyloguj”.

Oczekiwany wynik:

System wyświetla okienko potwierdzenia, w którym mamy możliwość kliknięcia „Tak” lub „Nie”.

Administrator zostaje wylogowany.

Scenariusz testowy nr 12.

Wylogowywanie się (bez potwierdzenia).

Kroki:

1. Zalogować się (login: admin, hasło: admin123).
2. Kliknąć przycisk „Wyloguj”.

Oczekiwany wynik:

System wyświetla okienko potwierdzenia, w którym mamy możliwość kliknięcia „Tak” lub „Nie”.

Administrator nie zostaje wylogowany.

OD STRONY UŻYTKOWNIKA

Scenariusz testowy nr 1.

Rozpoczęcie nowego zadania.

Kroki:

1. Zalogować się (login: user, hasło: user123).
2. Kliknąć przycisk „Start”.

Oczekiwany wynik:

Zadanie zostaje przeniesione do tabeli „Przypisane”.

Scenariusz testowy nr 2.

Zatrzymanie wykonywania zadania.

Kroki:

1. Zalogować się (login: user, hasło: user123).
2. Kliknąć przycisk „Start”.
3. Kliknąć przycisk „Zatrzymaj”.

Oczekiwany wynik:

Zadanie zostaje oznaczone kolorem szarym, czas zatrzymuje się.

Scenariusz testowy nr 3.

Wznowienie zatrzymanego zadania.

Kroki:

1. Zalogować się (login: user, hasło: user123).
2. Kliknąć przycisk „Start”.
3. Kliknąć przycisk „Zatrzymaj”.
4. Kliknąć przycisk „Wznów”.

Oczekiwany wynik:

Zadanie zostaje oznaczone ponownie kolorem białym, czas wznowia naliczanie.

Scenariusz testowy nr 4.

Zakończenie zadania (z potwierdzeniem).

Kroki:

1. Zalogować się (login: user, hasło: user123).
2. Kliknąć przycisk „Start”.
3. Kliknąć przycisk „Zakończ”.

Oczekiwany wynik:

System wyświetla okienko potwierdzenia, w którym mamy możliwość kliknięcia „Tak” lub „Nie”.

Zadanie zostaje przeniesione do tabeli „Zakończone”.

Scenariusz testowy nr 5.

Zakończenie zadania bez wcześniejszego zatrzymania (bez potwierdzenia).

Kroki:

1. Zalogować się (login: user, hasło: user123).
2. Kliknąć przycisk „Start”.
3. Kliknąć przycisk „Zakończ”.

Oczekiwany wynik:

System wyświetla okienko potwierdzenia, w którym mamy możliwość kliknięcia „Tak” lub „Nie”.

Zadanie pozostaje w tabeli „Przypisane”, czas jest naliczany.

Scenariusz testowy nr 6.

Zakończenie zadania z wcześniejszym zatrzymaniem (bez potwierdzenia).

Kroki:

1. Zalogować się (login: user, hasło: user123).
2. Kliknąć przycisk „Start”.
3. Kliknąć przycisk „Zatrzymaj”.
4. Kliknąć przycisk „Zakończ”.

Oczekiwany wynik:

System wyświetla okienko potwierdzenia, w którym mamy możliwość kliknięcia „Tak” lub „Nie”.

Zadanie pozostaje w tabeli „Przypisane”, czas nie wznawia się automatycznie.

Scenariusz testowy nr 7.

Wylogowywanie się (z potwierdzeniem).

Kroki:

1. Zalogować się (login: user, hasło: user123).
2. Kliknąć przycisk „Wyloguj”.

Oczekiwany wynik:

System wyświetla okienko potwierdzenia, w którym mamy możliwość kliknięcia „Tak” lub „Nie”.

Użytkownik zostaje wylogowany.

Scenariusz testowy nr 8.

Wylogowywanie się (bez potwierdzenia).

Kroki:

1. Zalogować się (login: user, hasło: user123).
2. Kliknąć przycisk „Wyloguj”..

Oczekiwany wynik:

System wyświetla okienko potwierdzenia, w którym mamy możliwość kliknięcia „Tak” lub „Nie”.

Użytkownik nie zostaje wylogowany.

ZESTAWY

Zestaw I.

Administrator:

1. **Zaloguj**
2. **Zadanie:** Example1
2. **Czas/czas ustalony:** 5.00
3. **Dodaj:** Example1
4. **Edytuj:** Example1, 5.00 - > Example2, 4.50
5. **Akceptuj:** Example2, 4.50

Użytkownik:

1. **Zaloguj**
2. **Start:** Example2, 4.50
3. **Zatrzymaj:** Example2
4. **Wznów:** Example2
5. **Zakończ:** Example2
6. **Wyloguj**

Zestaw II.

Administrator

1. **Zaloguj**
2. **Zadanie** (kolejno): Example2, Example3, Example4
3. **Czas/czas ustalony** (kolejno): 11.00, 3.00, 5.00
4. **Dodaj** (kolejno): Example2, Example3, Example4
5. **Edytuj**: Example3, 3.00 -> Example, 3.50
6. **Akceptuj**: Example, 3.50
7. **Usuń**: Example2, 11.00
8. **Wyloguj**

Użytkownik:

1. **Start** (kolejno): Example, Example4
2. **Zatrzymaj**: Example4
3. **Zakończ**: Example
4. **Wyloguj**

Zestaw III.

Podczas odbioru klient doda swój trzeci zestaw danych.

TESTY FUNKCJONALNE I NIEFUNKCJONALNE

Dokument opisuje wyniki testów funkcjonalnych i
niefunkcjonalnych oprogramowania.

SPIS TREŚCI

Testy funkcjonalne (str.
45-50)

Testy niefunkcjonalne
(str. 51-56)

TESTY FUNKCJONALNE

PRZYPADEK 1 (DODANIE NOWEGO ZADANIA)

Aktor	Administrator
Warunki początkowe	Administrator jest zalogowany na konto z uprawnieniami do modyfikowania zadań
Zdarzenie inicjujące	Kliknięcie przez administratora przycisku „Stwórz zadanie”

Przebieg w krokach:

1. System wyświetla interface dodawania nowego zadania zawierający pola

- Nazwa zadania - określenie nazwy zadania (limit 50 znaków)

- Czas ustalony - określenie czasu na wykonanie zadania (w sekundach)

2. Administrator wypełnia pole „Nazwa zadania” nie przekraczając limitu 50 znaków oraz pole „Czas ustalony” w sekundach.

3. Administrator zatwierdza zadanie przyciskiem „Create”.

4. Zadanie pojawia się w tabeli „Zadania”.

PRZYPADEK 2 (EDYCJA ISTNIEJĄCEGO ZADANIA)

Aktor	Administrator
Warunki początkowe	Administrator jest zalogowany na konto z uprawnieniami do modyfikowania zadań
Zdarzenie inicjujące	Kliknięcie przez administratora przycisku „Edit”

Przebieg w krokach:

1. System wyświetla interface edycji zadania zawierający pola
 - Nazwa zadania - określenie nazwy zadania (limit 50 znaków)
 - Czas ustalony - określenie czasu na wykonanie zadania (w sekundach)
2. Administrator wypełnia pole „Nazwa zadania” nie przekraczając limitu 50 znaków oraz pole „Czas ustalony” w sekundach.
3. Administrator zatwierdza zadanie przyciskiem „Save”.
4. Zaktualizowane zadanie pojawia się w tabeli „Zadania”.

PRZYPADEK 3 (USUWANIE ISTNIEJĄCEGO ZADANIA)

Aktor	Administrator
Warunki początkowe	Administrator jest zalogowany na konto z uprawnieniami do modyfikowania zadań
Zdarzenie inicjujące	Kliknięcie przez administratora przycisku "Delete"

Przebieg w krokach:

1. System wyświetla interface z możliwością usunięcia zadania lub wycofania się.
2. Administrator zatwierdza usuwanie przyciskiem „Delete”
3. Zadanie zostaje usunięte z tabeli „Zadania”.

PRZYPADEK 4 (ZATRZYMANIE ZADANIA)

Aktor	Użytkownik
Warunki początkowe	Istnieje co najmniej jedno przypisane zadanie do danego użytkownika
Zdarzenie inicjujące	Kliknięcie przycisku „Zatrzymaj” przy wybranym zadaniu

Przebieg w krokach:

1. Czas przestaje być naliczany, zaktualizowane zostają kolumny „Czas realny” oraz „Różnica”, wiersz z zadaniem podświetla się na szaro.

PRZYPADEK 5 (WZNOWIENIE ZADANIA)

Aktor	Użytkownik
Warunki początkowe	Istnieje co najmniej jedno przerwane zadanie przypisane do danego użytkownika
Zdarzenie inicjujące	Kliknięcie przycisku „Wznów” przy wybranym zadaniu

Przebieg w krokach:

1. Czas wznowia naliczanie, wiersz z zadaniem jest biały.

PRZYPADEK 6 (ZAKOŃCZENIE ZADANIA Z NIEUJEMNĄ RÓŻNICĄ)

Aktor	Użytkownik
Warunki początkowe	Istnieje co najmniej jedno zadanie w toku przypisane do danego użytkownika
Zdarzenie inicjujące	Kliknięcie przycisku „Zakończ” przy wybranym zadaniu

Przebieg w krokach:

1. Dane zadanie zostaje przeniesione do tabeli „Zakończone” w kolorze zielonym.
2. Zaktualizowane zostaje pole „Suma różnic”.

PRZYPADEK 7 (ZAKOŃCZENIE ZADANIA Z UJEMNĄ RÓŻNICĄ)

Aktor	Użytkownik
Warunki początkowe	Istnieje co najmniej jedno zadanie w toku przypisane do danego użytkownika
Zdarzenie inicjujące	Kliknięcie przycisku „Zakończ” przy wybranym zadaniu

Przebieg w krokach:

1. Dane zadanie zostaje przeniesione do tabeli „Zakończone” w kolorze czerwonym.
2. Zaktualizowane zostaje pole „Suma różnic”.

TESTY NIEFUNKCJONALNE

PRZYPADEK 1 (DODANIE NOWEGO ZADANIA V1)

Aktor	Administrator
Warunki początkowe	Administrator jest zalogowany na konto z uprawnieniami do modyfikowania zadań
Zdarzenie inicjujące	Kliknięcie przez administratora przycisku „Stwórz zadanie”

Przebieg w krokach:

1. System wyświetla interface dodawania nowego zadania zawierający pola

- Nazwa zadania - określenie nazwy zadania (limit 50 znaków)

- Czas ustalony - określenie czasu na wykonanie zadania (w sekundach)

2. Administrator wypełnia pole „Nazwa zadania” przekraczając limit 50 znaków oraz pole „Czas ustalony” w sekundach.

3. System informuje o nieprawidłowym wypełnieniu pola „Nazwa zadania”.

PRZYPADEK 2 (DODANIE NOWEGO ZADANIA V2)

Aktor	Administrator
Warunki początkowe	Administrator jest zalogowany na konto z uprawnieniami do modyfikowania zadań
Zdarzenie inicjujące	Kliknięcie przez administratora przycisku „Stwórz zadanie”

Przebieg w krokach:

1. System wyświetla interface dodawania nowego zadania zawierający pola

- Nazwa zadania - określenie nazwy zadania (limit 50 znaków)
- Czas ustalony - określenie czasu na wykonanie zadania (w sekundach)

2. Administrator wypełnia pole „Nazwa zadania” nie przekraczając limitu 50 znaków oraz pole „Czas ustalony” jako łańcuch znaków.

3. System informuje o nieprawidłowym wypełnieniu pola „Czas ustalony”.

PRZYPADEK 3 (DODANIE NOWEGO ZADANIA V3)

Aktor	Administrator
Warunki początkowe	Administrator jest zalogowany na konto z uprawnieniami do modyfikowania zadań
Zdarzenie inicjujące	Kliknięcie przez administratora przycisku „Stwórz zadanie”

Przebieg w krokach:

1. System wyświetla interface dodawania nowego zadania zawierający pola

- Nazwa zadania - określenie nazwy zadania (limit 50 znaków)
- Czas ustalony - określenie czasu na wykonanie zadania (w sekundach)

2. Administrator wypełnia pole „Nazwa zadania” nie przekraczając limitu 50 znaków oraz pole „Czas ustalony” liczbą spoza przedziału $\langle 0, 100\ 000 \rangle$.

3. System informuje o nieprawidłowym wypełnieniu pola „Czas ustalony”.

PRZYPADEK 4 (EDYCJA ISTNIEJĄCEGO ZADANIA V1)

Aktor	Administrator
Warunki początkowe	Administrator jest zalogowany na konto z uprawnieniami do modyfikowania zadań
Zdarzenie inicjujące	Kliknięcie przez administratora przycisku „Edit”

Przebieg w krokach:

1. System wyświetla interface edycji zadania zawierający pola
 - Nazwa zadania - określenie nazwy zadania (limit 50 znaków)
 - Czas ustalony - określenie czasu na wykonanie zadania (w sekundach)
2. Administrator wypełnia pole „Nazwa zadania” przekraczając limit 50 znaków oraz pole „Czas ustalony” w sekundach.
3. System informuje o nieprawidłowym wypełnieniu pola „Nazwa zadania”.

PRZYPADEK 5 (EDYCJA ISTNIEJĄCEGO ZADANIA V2)

Aktor	Administrator
Warunki początkowe	Administrator jest zalogowany na konto z uprawnieniami do modyfikowania zadań
Zdarzenie inicjujące	Kliknięcie przez administratora przycisku „Edit”

Przebieg w krokach:

1. System wyświetla interface edycji zadania zawierający pola
 - Nazwa zadania - określenie nazwy zadania (limit 50 znaków)
 - Czas ustalony - określenie czasu na wykonanie zadania (w sekundach)
2. Administrator wypełnia pole „Nazwa zadania” nie przekraczając limitu 50 znaków oraz pole „Czas ustalony” jako łańcuch znaków.
3. System informuje o nieprawidłowym wypełnieniu pola „Czas ustalony”.

PRZYPADEK 6 (EDYCJA ISTNIEJĄCEGO ZADANIA V3)

Aktor	Administrator
Warunki początkowe	Administrator jest zalogowany na konto z uprawnieniami do modyfikowania zadań
Zdarzenie inicjujące	Kliknięcie przez administratora przycisku „Edit”

Przebieg w krokach:

1. System wyświetla interface edycji zadania zawierający pola
 - Nazwa zadania - określenie nazwy zadania (limit 50 znaków)
 - Czas ustalony - określenie czasu na wykonanie zadania (w sekundach)
2. Administrator wypełnia pole „Nazwa zadania” nie przekraczając limitu 50 znaków oraz pole „Czas ustalony” liczbą spoza przedziału $\langle 0, 100\ 000 \rangle$.
3. System informuje o nieprawidłowym wypełnieniu pola „Czas ustalony”.

ARCHITEKTURA SYSTEMU

Dokumentacja zawiera diagramy architektury systemu, schemat bazy danych, oraz diagramu UML będące reprezentacją przypadków użycia.

SPIS TREŚCI

Przypadek 1 (str. 58)

Przypadek 2 (str. 58)

Przypadek 3 (str. 59)

Przypadek 4 (str. 59)

Przypadek 5 (str. 60)

Przypadek 6 (str. 60)

Przypadek 7 (str. 61)

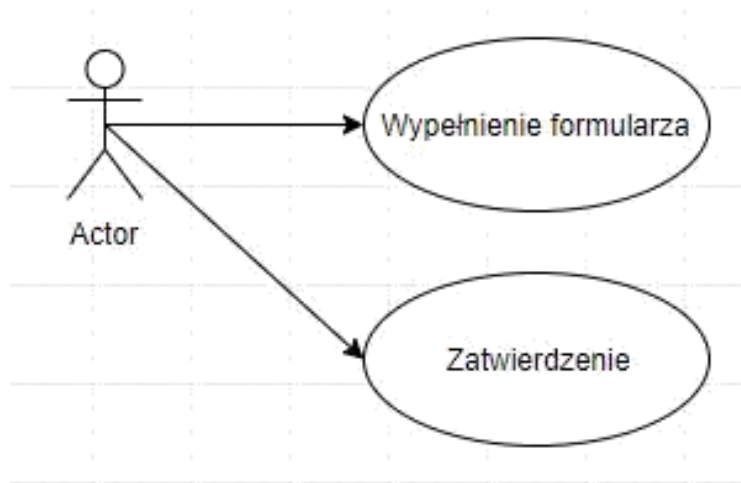
Przypadek 8 (str. 61)

Przypadek 9 (str. 62)

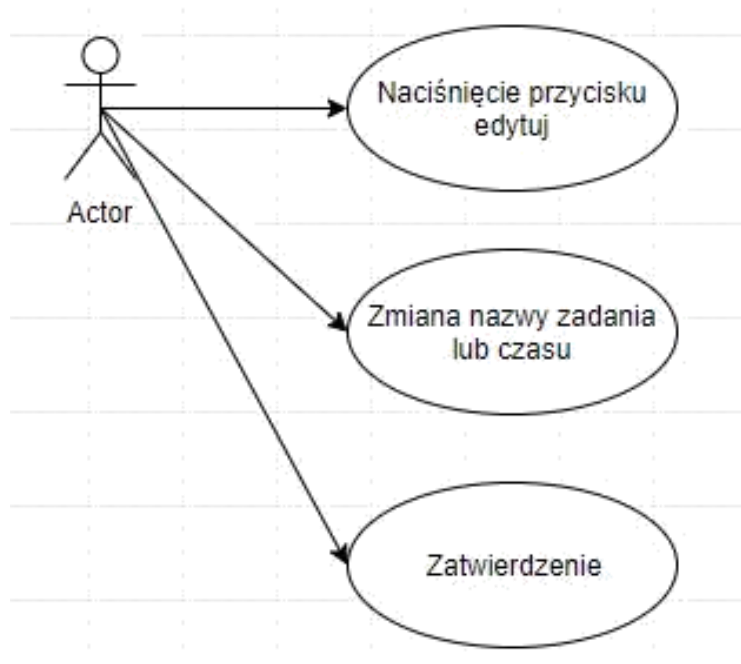
Architektura systemu (str.
63)

Schemat bazy danych (str.
64)

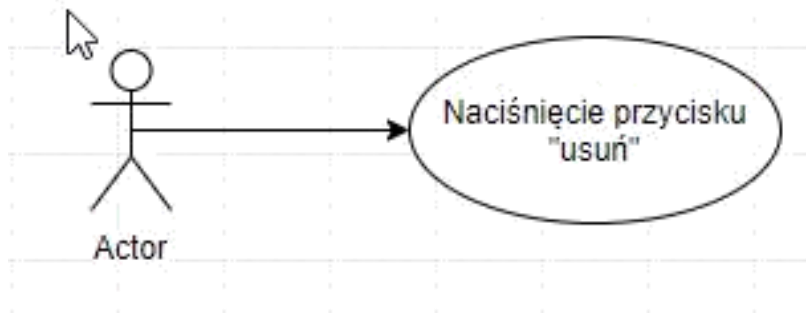
PRZYPADEK 1 (DODANIE NOWEGO ZADANIA)



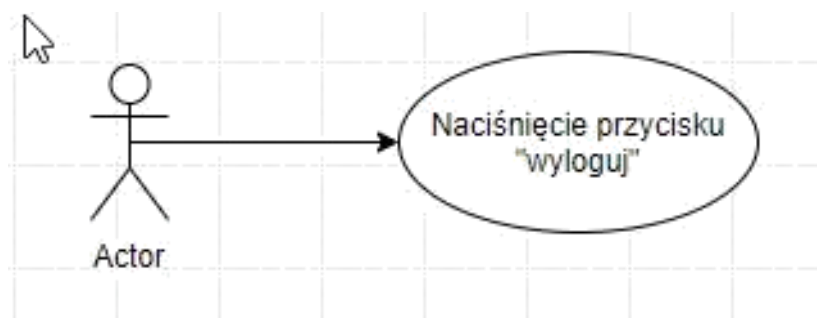
PRZYPADEK 2 (EDYCJA ISTNIEJĄCEGO ZADANIA)



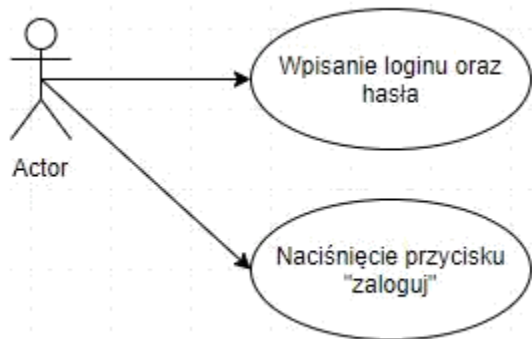
PRZYPADEK 3 (USUWANIE INSTIEJĄCEGO ZADANIA)



PRZYPADEK 4 (WYLOGOWANIE SIĘ Z APLIKACJI)



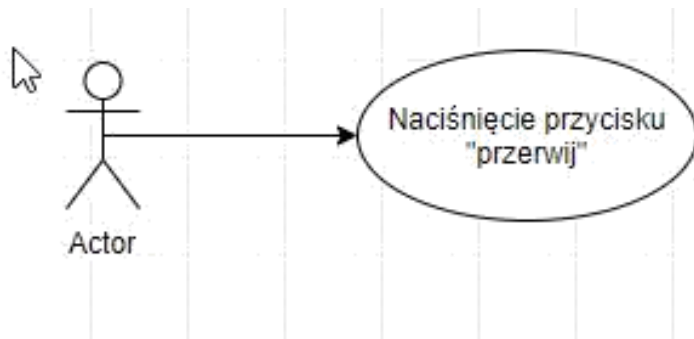
PRZYPADEK 5 (ZALOGOWANIE DO APLIKACJI)



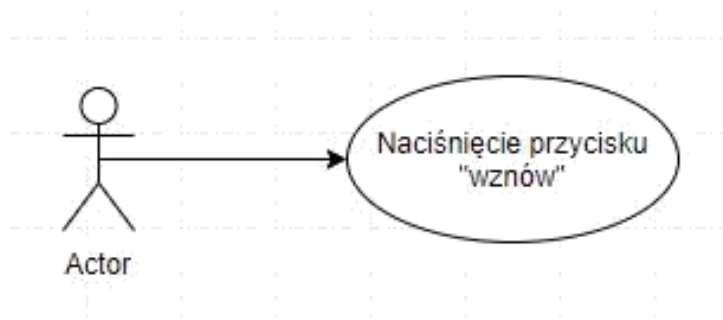
PRZYPADEK 6 (ROZPOCZĘCIE ZADANIA)



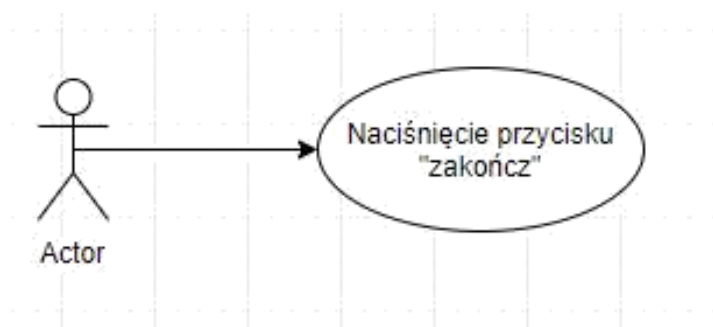
PRZYPADEK 7 (PRZERWANIE ZADANIA)



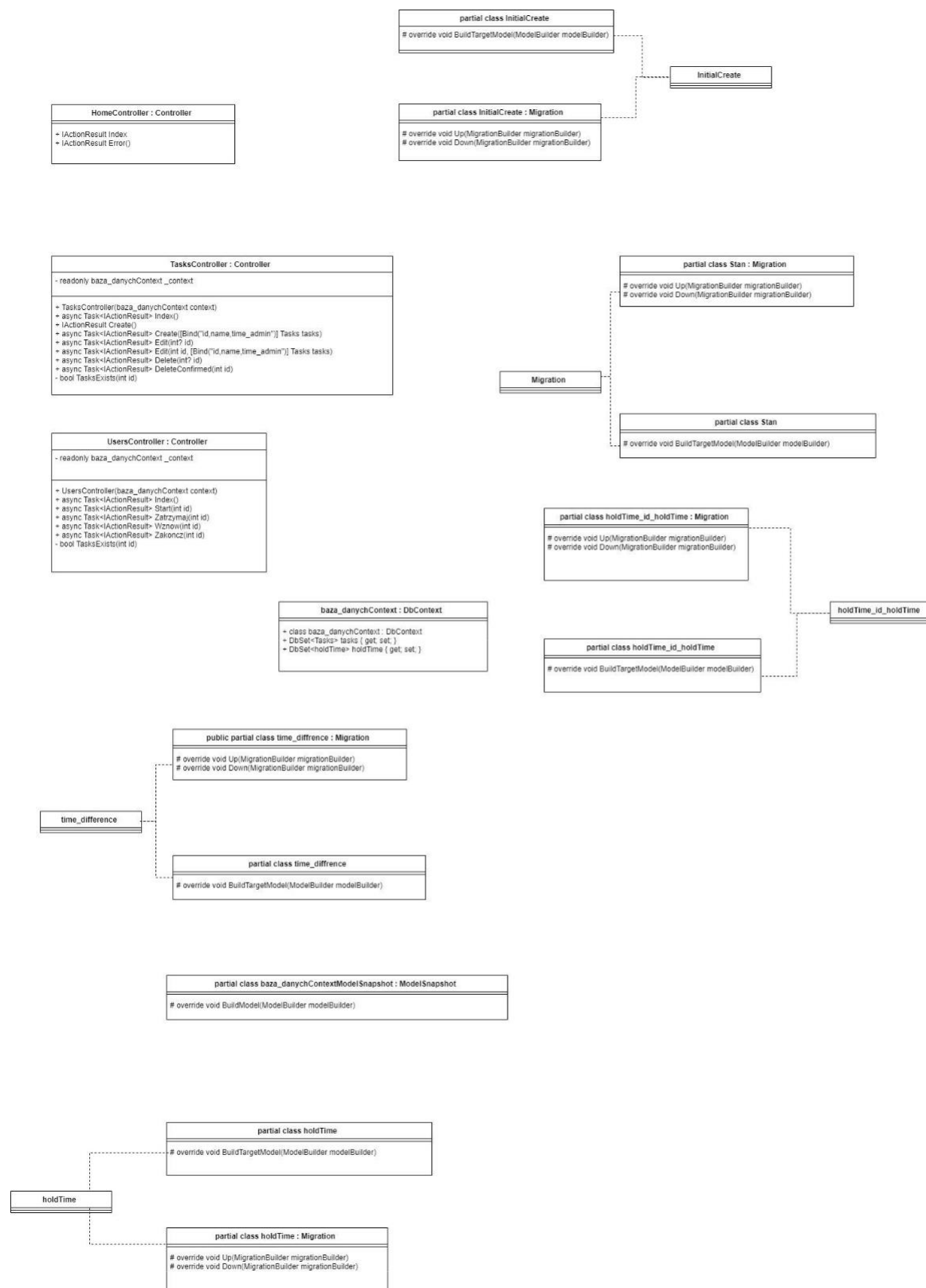
PRZYPADEK 8 (WZNOWIENIE ZADANIA)



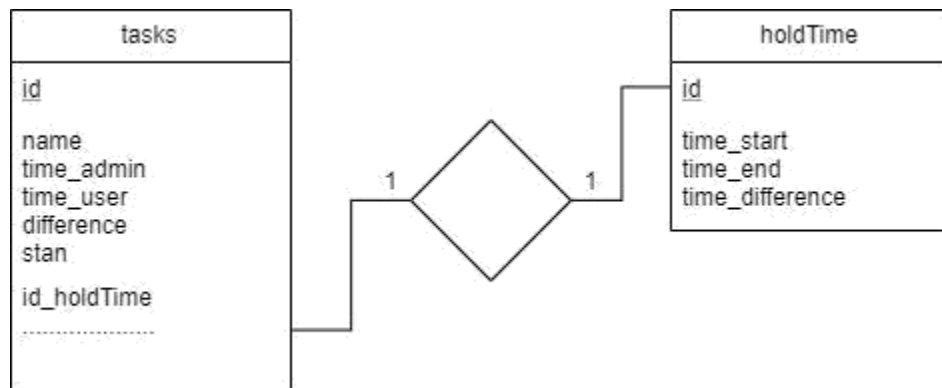
PRZYPADEK 9 (ZAKOŃCZENIE ZADANIA)



Architektura systemu



Architektura systemu



DOKUMENTACJA API

Wprowadzenie

API umożliwia wymianę informacji pomiędzy zewnętrznym systemem a Stoperem programisty.

Dostęp do projektu:

<https://projektrup.projektstudencki.pl/>

Dozwolone zapytania HTTPs

POST : Edycja lub usunięcie danych

GET : Tworzenie lub wyświetlanie danych

Przewidywane odpowiedzi serwera

- 200 - OK - Zapytanie wykonane pomyślnie
- 201 - Created - Zapytanie wykonane pomyślnie i element został stworzony
- 204 - No Content - Zapytanie wykonane pomyślnie ale nie ma co zwrócić
- 400 - Bad Request - Zapytanie jest błędne lub brak potrzebnych parametrów
- 404 - Not Found - Nie znaleziono danego zapytania
- 405 - Method Not Allowed - Dana metoda nie jest wspomagana

SPIS TREŚCI

Metody

Administradora(str.66-67)

Metody

Użytkownika(str.68-69)

Metody

Administratora: /Tasks/

- GET: Tasks - Pobranie informacji o całym panelu administratora
- GET: Tasks/Create - Przejście do widoku Create
- POST: Tasks/Create - Służy do pobierania danych nowo utworzonego zadania z widoku(Tasks/Create) oraz zapisaniu ich do bazy

Parametry Wejściowe:

name	string	Nazwa nowego zadania
time_admin	int	Maksymalny czas na wykonanie zadania

Dane Wyjściowe:

id	int	Ustalone id zadania
name	string	Ustalona nazwa nowego zadania
time_admin	int	Ustalony maksymalny czas na wykonanie zadania
time_user	int	Aktualny czas użytkownika
diffrence	int	Różnica czasu
stan	bool	stan wykonania zadania, domyślnie false
id_holdTime	int	Przechowuje id do czasu zadania

- GET: Tasks/Edit - Służy do pokazania danych zadania o id oraz przedstawienia ich w widoku(Tasks/Edit)

Parametry Wejściowe:

id	int	Ustalone id zadania
----	-----	---------------------

Dane wyjściowe:

name	string	Nazwa zadania
time_admin	int	Maksymalny czas na wykonanie zadania

- POST: Tasks/Edit - Edycja zadania o danym id

Parametry Wejściowe:

id	int	Ustalone id zadania
name	string	Nazwa nowego zadania
time_admin	int	Maksymalny czas na wykonanie zadania

Dane Wyjściowe:

id	int	Ustalone id zadania
name	string	Ustalona nazwa nowego zadania
time_admin	int	Ustalony maksymalny czas na wykonanie zadania
time_user	int	Aktualny czas użytkownika
diffrence	int	Różnica czasu
stan	bool	stan wykonania zadania, domyślnie false
id_holdTime	int	Przechowuje id do czasu zadania

- GET: Tasks/Delete- służy do sprawdzenia czy istnieje zadanie o danym id oraz przekazuje rekord tego zadania do widoku(Tasks/Delete)

Parametry Wejściowe:

id	int	Ustalone id zadania
----	-----	---------------------

Dane Wyjściowe:

name	string	Nazwa zadania
time_admin	int	Maksymalny czas na wykonanie zadania

- POST: Tasks/Delete - Jeśli zatwierdzone zostało usunięcie zadania id to zajmuje się usunięciem tego z bazy

Dane Wyjściowe:

id	int	Ustalone id zadania
----	-----	---------------------

Użytkownika: /Users/

- GET: Users - Pobranie informacji o całym panelu użytkownika
- GET: Start - Służy do stworzenia zegara(w bazie danych) dla zadania o danym id

Parametry Wejściowe:

id	int	Ustalone id zadania
----	-----	---------------------

Dane Wyjściowe:

id	int	Ustalone id zadania
name	string	Ustalona nazwa nowego zadania
time_admin	int	Ustalony maksymalny czas na wykonanie zadania
time_user	int	Aktualny czas użytkownika
diffrence	int	Różnica czasu
stan	bool	stan wykonania zadania, domyślnie false
id_holdTime	int	Przechowuje id do czasu zadania

- GET: Zatrzymaj - Służy do zaktualizowania aktualnego czasu jaki użytkownik przeznaczył na zadanie oraz zatrzymania zegara

Parametry Wejściowe:

id	int	Ustalone id zadania
----	-----	---------------------

Dane Wyjściowe:

time_user	int	Zaktualizowany czas użytkownika
time_diffrence	int	Różnica pomiędzy wznowieniem a zakończeniem
time_end	int	Nadpisuje czas końcowy
diffrence	int	Zaktualizowany różnica czasu

- GET: Wznów - Służy do przestawienia i uruchomienia zegara na aktualny czas i liczeniu nowej różnicy

Parametry Wejściowe:

id	int	Ustalone id zadania
----	-----	---------------------

Dane Wyjściowe:

id	int	Ustalone id zadania
name	string	Ustalona nazwa nowego zadania
time_admin	int	Ustalony maksymalny czas na wykonanie zadania
time_user	int	Aktualny czas użytkownika
diffrence	int	Różnica czasu
stan	bool	stan wykonania zadania, domyślnie false
id_holdTime	int	Przechowuje id do czasu zadania

- GET: Zakoncz - Służy do zaktualizowania czasu użytkownika oraz zatwierdzenia zakończenia zadania

Parametry Wejściowe:

id	int	Ustalone id zadania
----	-----	---------------------

Dane Wyjściowe:

time_user	int	Zaktualizowany czas użytkownika
time_diffrence	int	Różnica pomiędzy wznowieniem a zakończeniem
time_end	int	Nadpisuje czas końcowy
diffrence	int	Zaktualizowany różnica czasu
stan	bool	Stan wykonania zadania

OPIS WYKORZYSTANEJ INFRASTRUKTURY

Część dokumentacji przeznaczona na zapoznanie się z infrastrukturą projektu czyli o tym gdzie jest zainstalowany projekt, jak go uruchomić, przenieść oraz w jaki sposób go zatrzymać oraz uruchomić ponownie

- Dokumentacja składa się z części opisowej danej sekcji oraz z ilustracji pomagających w zrozumieniu działania.
- Każdy z opisów umieszczony jest pod ilustracją.

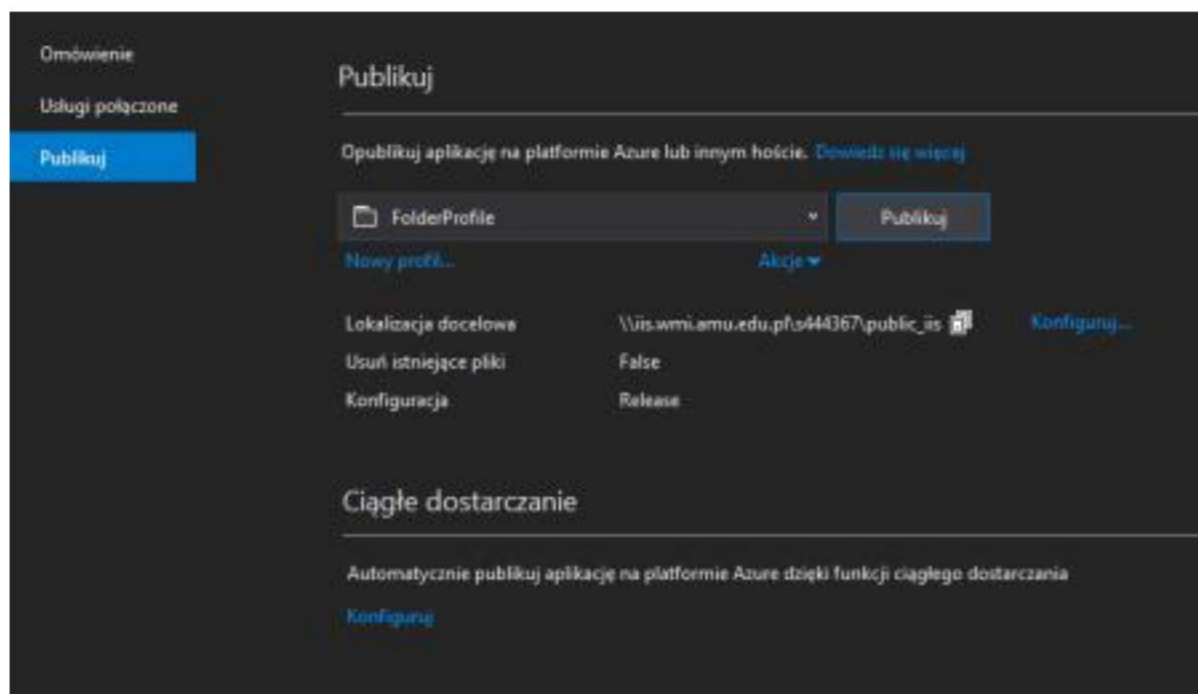
SPIS TREŚCI

1. Instalacja (str. 71)
2. Poł. z siecią (str. 71-72)
3. Wprowadzanie zmian (str. 72)
4. Przenoszenie / ponowna instalacja (str. 73)

1 – INSTALACJA

Instalacja następuje po publikacji projektu na serwerze zawierającym usługę iis.

Przy publikowaniu projektu ustawiamy folder serwera który ma być naszym hostem.































2 – POŁĄCZENIE Z SIECIĄ



Znajdujemy się w folderze, w którym ustawiliśmy lokalizację docelową publikacji naszego projektu. [\\iis.wmi.amu.edu.pl\s444367\public_iis]

Strona, pod która podpięta jest usługa: projektrup.projektstudencki.pl

 aspnet_client	2019-12-12 19:37	Folder plików	
 wwwroot	2019-12-15 21:25	Folder plików	
 app_offline	2019-12-16 16:46	Plik HTM	1 KB
 appsettings.Development	2019-12-11 09:37	JSON File	1 KB
 appsettings	2019-12-15 20:48	JSON File	1 KB
 dotnet-aspnet-codegenerator-design.dll	2019-03-02 19:51	Rozszerzenie aplik...	52 KB
 Microsoft.CodeAnalysis.CSharp.Workspa...	2019-11-02 17:51	Rozszerzenie aplik...	674 KB
 Microsoft.CodeAnalysis.Workspaces.dll	2019-11-02 17:52	Rozszerzenie aplik...	2 564 KB
 Microsoft.VisualStudio.Web.CodeGenera...	2019-03-02 19:51	Rozszerzenie aplik...	23 KB
 Microsoft.VisualStudio.Web.CodeGenera...	2019-03-02 19:51	Rozszerzenie aplik...	72 KB
 Microsoft.VisualStudio.Web.CodeGenera...	2019-03-02 19:51	Rozszerzenie aplik...	35 KB
 Microsoft.VisualStudio.Web.CodeGenera...	2019-03-02 19:51	Rozszerzenie aplik...	68 KB
 Microsoft.VisualStudio.Web.CodeGenera...	2019-03-02 19:51	Rozszerzenie aplik...	28 KB
 Microsoft.VisualStudio.Web.CodeGenera...	2019-03-02 19:51	Rozszerzenie aplik...	34 KB
 Microsoft.VisualStudio.Web.CodeGenera...	2019-03-02 19:51	Rozszerzenie aplik...	154 KB
 NuGet.Frameworks.dll	2019-11-02 17:50	Rozszerzenie aplik...	107 KB
 Projekt.deps	2019-12-15 22:02	JSON File	242 KB
 Projekt.dll	2019-12-15 22:02	Rozszerzenie aplik...	37 KB
 Projekt.pdb	2019-12-15 22:02	Program Debug D...	8 KB
 Projekt.runtimeconfig	2019-12-15 22:02	JSON File	1 KB
 Projekt.Views.dll	2019-12-15 22:02	Rozszerzenie aplik...	120 KB
 Projekt.Views.pdb	2019-12-15 22:02	Program Debug D...	11 KB
 System.Composition.AttributedModel.dll	2019-11-02 17:50	Rozszerzenie aplik...	25 KB
 System.Composition.Convention.dll	2019-11-02 17:50	Rozszerzenie aplik...	58 KB
 System.Composition.Hosting.dll	2019-11-02 17:50	Rozszerzenie aplik...	61 KB
 System.Composition.Runtime.dll	2019-11-02 17:50	Rozszerzenie aplik...	30 KB
 System.Composition.TypedParts.dll	2019-11-02 17:50	Rozszerzenie aplik...	64 KB
 web.config	2019-12-15 22:02	XML Configuratio...	1 KB

3 – WPROWADZANIE ZMIAN

Aby zachować płynność aplikacji potrzebne jest wstrzymanie usługi na czas wprowadzania zmian. Aby zatrzymać usługę należy stworzyć plik “app.offline.htm” dzięki niemu usługa zostanie wstrzymana, a po jego usunięciu zostanie wznowiona. (To co pojawi się w tym pliku nie ma znaczenia).

Zmian dokonujemy lokalnie w środowisku np. “Visual Studio”, po publikacji pliki przesyłane są na folder serwerowy przez co są aktualizowane. Należy mieć pewność, że zmian dokonujemy gdy usługa jest wyłączona, w przeciwnym razie może to prowadzić do błędów aplikacji.

4 – PRZENOSZENIE / PONOWNA INSTALACJA

Jeżeli chcemy przenieść nasz projekt wystarczy, że zmienimy lokalizację docelową podczas publikacji (instalacji) projektu. Sam proces publikacji projektu na serwerze jest jego instalacją. Uruchomiony pozostaje on do momentu wyłączenia usługi (za pomocą pliku “app.offline.htm” w folderze na serwerze).

PREZENTACJA SPOSOBU DOKUMENTACJI KODU

- Dokumentacja składa się z części ilustrującej jak wygląda dana część kodu oraz części odpowiedzialnej za opis.
- Dokument został podzielony na dwie główne części: Opis użytej technologii (wzorzec projektowy) oraz na część odpowiedzialną za opis struktury kodu oraz jego dokumentowania w projekcie

SPIS TREŚCI

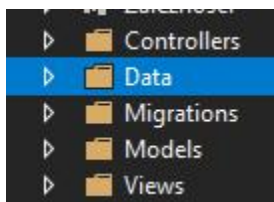
Wzorzec projektowy (str. 75)

Struktura kodu (str. 75-78)

1 – WZORZEC PROJEKTOWY

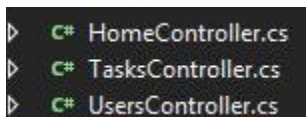
Wzorcem projektu jest architektura **MVC** (Model-View-Controller). Koncepcją tego wzorca jest podział aplikacji na trzy warstwy reprezentujące kolejno:

1. **Model** – model danych czyli opis struktur danych i powiązań między nimi
2. **View** – interfejs czyli to co widzi użytkownik
3. **Controller** – logika działania czyli powiązania między zdarzeniami zachodzącymi w systemie



2 – STRUKTURA KODU

W przypadku tego projektu występują trzy **kontrolery**:



1. **Home** – strona główna
2. **Tasks** – strona dla administratora
3. **Users** – strona dla użytkownika

Home zawiera dwie podstrony:

1. **Index** – obsługującą główną funkcję strony Home

```
public IActionResult Index()
```

2. **Error** – odpowiedzialną za wyłapywanie errorów

```
public IActionResult Error()
```

Tasks zawiera podstrony z POST/GET:

1. **Index** – strona główna (łapana za pomocą get)

```
// GET: Tasks
Odwołania: 3 | 0 żądań | 0 wyjątki
public async Task<IActionResult> Index()
```

2. **Create (GET)** – służy do przejścia do widoku (View) Tasks/Create

```
// GET: Tasks/Create
Odwołania: 0 | 0 żądań | 0 wyjątki
public IActionResult Create()...
```

3. **Create (POST)** – służy do pobierania danych nowo utworzonego zadania z widoku (Tasks/Create) oraz zapisaniu go do bazy

```
// POST: Tasks/Create
// To protect from overposting attacks, please enable the specific properties you want to bind to, for
// more details see http://go.microsoft.com/fwlink/?LinkId=317598.
[HttpPost]
[ValidateAntiForgeryToken]
Odwołania: 0 | 0 żądań | 0 wyjątki
public async Task<IActionResult> Create([Bind("id,name,time_admin")] Tasks tasks)...
```

4. **Edit (GET)** – służy do pokazania danych zadania o id równym pobranemu z GET'a oraz przedstawienia ich w widoku (Tasks/Edit)

```
// GET: Tasks/Edit/5
Odwołania: 0 | 0 żądań | 0 wyjątki
public async Task<IActionResult> Edit(int? id)...
```

5. **Delete (GET)** – służy do sprawdzenia czy istnieje zadanie o id równym temu z GET'a oraz przekazuje rekord tego zadania do widoku (Tasks/Delete)

```
// GET: Tasks/Delete/5
Odwołania: 0 | 0 żądań | 0 wyjątki
public async Task<IActionResult> Delete(int? id)...
```

6. **Delete (POST)** – jeśli zatwierdzone zostało usunięcie zadania o danym id to zajmuje się usunięciem go z bazy

```
// POST: Tasks/Delete/5
[HttpPost, ActionName("Delete")]
[ValidateAntiForgeryToken]
Odwołania: 0 | 0 żądań | 0 wyjątki
public async Task<IActionResult> DeleteConfirmed(int id)
{
    var tasks = await _context.tasks.FindAsync(id);
    _context.tasks.Remove(tasks);
    await _context.SaveChangesAsync();
    return RedirectToAction(nameof(Index));
}
```

Users – zawiera podstrony:

1. **Index** – służy do przekazywania 2 tabel do widoku (Index)

```
// GET: Users
Odwołania: 4 | 0 żądań | 0 wyjątki
public async Task<IActionResult> Index()
```

2. **Start** – służy do stworzenia zegara (w bazie danych) dla zadania o danym id, uruchomieniu zegara i zatrzymania dotychczas uruchomionego

```
// GET: Start
Odwołania: 0 | 0 żądań | 0 wyjątki
public async Task<IActionResult> Start(int id)
```

3. **Zatrzymaj** – służy do zaktualizowania czasu jaki użytkownik przeznaczył na zadanie oraz do zatrzymania zegara (zmieniając wartości w bazie danych) dla zadania o danym id

```
// GET: Zatrzymaj
Odwołania: 0 | 0 żądań | 0 wyjątki
public async Task<IActionResult> Zatrzymaj(int id)
```

4. **Wznów** – służy do przestawienia zegara na aktualny czas oraz uruchomienia go, zatrzymania dotychczas aktywnego zegara i

liczenia nowej różnicy

```
// GET: Wznow  
Odwołania: 0 | 0 żądań | 0 wyjątki  
public async Task<IActionResult> Wznow(int id)
```

5. **Zakończ** – służy do zaktualizowania czasu użytkownika oraz zatwierdzenia zakończenia zadania

```
// GET: Zakoncz  
Odwołania: 0 | 0 żądań | 0 wyjątki  
public async Task<IActionResult> Zakoncz(int id)
```