TEAMS

Dokumentacja projektowa PZSP2

WERSJA 1

06.11.2025R.

Semestr 25z

Zespół nr 1 w składzie:

Damian D’Souza Kamil Marszałek Michał Suski Michał Szwejk

Mentor zespołu: dr Marcin Szlenk
Właściciel tematu: prof. dr hab. Robert Nowak

Spis treści

1 Wprowadzenie

1.1 Cel projektu

Celem projektu jest wykorzystanie API aplikacji MS Teams, aby umożliwić tworzenie własnych rozwiązań.

1.2 Wstępna wizja projektu

Najważniejszą częścią projektu jest biblioteka umożliwiająca komunikowanie się z API. Biblioteka będzie umożliwiała wykonanie konkretnych requestów zapewniając przy tym odpowiedni poziom abstrakcji. Użytkownik będzie mógł z niej skorzystać w celu automatyzacji procesów. W celu demonstracji możliwości biblioteki powstanie aplikacja (command-line interface lub terminal-user interface). Projekt zostanie zaimplementowany w języku GO.

2 Metodologia wytwarzania

W pierwszej kolejności zaimplementowana zostanie biblioteka wraz z testami jednostkowymi i odpowiednią dokumentacją. Kiedy pokrycie testami będzie wystarczające zostanie rozpoczęty proces wytwarzania drugiego modułu systemu – aplikacji terminalowej demonstrującej jej działanie.

W procesie wytwarzania oprogramowania zostaną wykorzystane mechanizmy serwisu GitHub: GitHub Actions (pipeline), Issues.

3 Analiza wymagań

3.1 Wymagania użytkownika i biznesowe

Użytkownik wykorzystując bibliotekę lub aplikację terminalową będzie mógł:

- Automatycznie utworzyć kanały dla zespołów projektowych definiując listę z podziałem na grupy.

- Ustawić automatyczne wysłanie wiadomości na kanale o konkretnej porze (np. w celu przypomnienia o spotkaniu).

- Automatycznie wysłać parametryzowane wiadomości prywatne do wielu wskazanych użytkowników.

- Automatycznie wysłać parametryzowane wiadomości na wiele wskazanych kanałów.

- Przejrzeć wszystkie wiadomości otrzymane na kanałach zespołów we wskazanym oknie czasowym.

(Lista funkcjonalności z dużym prawdopodobieństwem rozrośnie się w czasie implementacji projektu.)

3.2 Wymagania funkcjonalne i niefunkcjonalne

W celu obsługi wymagań użytkownika biblioteka zapewni dostęp do następujących funkcji:

1. Zarządzanie zespołami teams.
2. Zarządzanie kanałami zespołu.
3. Zarządzanie członkami zespołu.
4. Pobieranie i wysyłanie wiadomości na kanale.
5. Zarządzanie chatami.
6. Pobieranie wiadomości nieodczytanych.

(Lista obsłużonych funkcjonalności może się rozrosnąć)

3.3 Przypadki użycia

Jedynym aktorem jest Użytkownik.

Przypadki biznesowe

PB1. Wysłanie wiadomości parametryzowanych.

Scenariusz główny:

1. Użytkownik definiuje szablon wiadomości i parametry.
2. Użytkownik adresuje wiadomość.
3. Użytkownik zleca wysłanie wiadomości.
4. System wysyła wiadomości do podanych adresatów.

Scenariusz alternatywny – Adresat nie istnieje

1-3. Jak w scenariuszu głównym.

4. System powiadamia o nieistniejącym adresacie.

PB2. Utworzenie kanałów dla zespołów projektowych.

Scenariusz główny:

1. Użytkownik wskazuje zespół, gdzie utworzyć kanały.
2. Użytkownik podaje parametryzowaną nazwę kanałów i listę członków.
3. Użytkownik zleca utworzenie kanałów.
4. System tworzy kanały dla zdefiniowanych zespołów.
5. System dodaje do utworzonych kanałów członków według listy.

PB3. Odczytanie nowych wiadomości.

Scenariusz główny:

1. Użytkownik wskazuje okno czasowe.
2. System wyświetla listę nieodczytanych wiadomości.

Scenariusz alternatywny – brak nieodczytanych wiadomości w wskazanym oknie czasowym:

1. Jak w scenariuszu głównym.
2. System powiadamia o braku nieodczytanych wiadomości.

PB4. Wysłanie wiadomości według ustalonego harmonogramu.

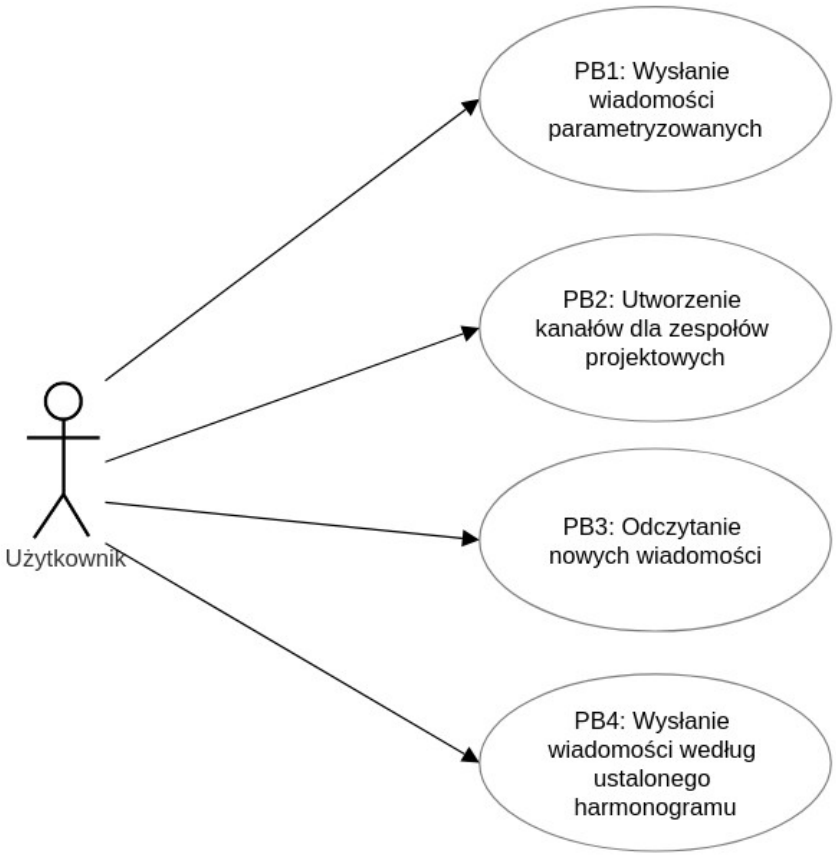
Scenariusz główny:

1. Użytkownik definiuje wiadomości i ich harmonogram.
2. Użytkownik adresuje wiadomości.
3. Użytkownik zleca wysłanie wiadomości.
4. System wysyła wiadomości zgodnie ze wskazanym harmonogramem.

Scenariusz alternatywny – Adresat nie istnieje

1-3. Jak w scenariuszu głównym.

4. System powiadamia o nieistniejącym adresacie.



Przypadki systemowe

FU1. Logowanie użytkownika.

Funkcjonalność potrzebna do wszystkich PB.

Scenariusz główny:

1. System za pośrednictwem przeglądarki wyświetla okno logowania.

2. Użytkownik loguje się na platformę Teams.
3. System sprawdza, czy operacja logowania zakończyła się sukcesem.
4. System zamyka okno logowania i zapamiętuje token użytkownika.

Scenariusz alternatywny 1 – nieudane logowanie:

1-3. Jak w scenariuszu głównym.

4. Powrót do kroku 1 scenariusza głównego.

Scenariusz alternatywny 2 – użytkownik był już zalogowany:

1. System wczytuje zapamiętany token użytkownika.

FU2. Wysyłanie wiadomości.

Wspiera PB1 i PB4. Korzysta z FU1 i FU3.

Scenariusz główny:

1. Użytkownik zleca wysłanie wiadomości.
2. System ustala treść wiadomości, adresata, porę wysłania wiadomości odczytując je z szablonu.
3. System wysyła request POST do MS Graph API.
4. System potwierdza wysłanie wiadomości.

Scenariusz alternatywny – nieudana próba wysłania wiadomości:

1-3. Jak w scenariuszu głównym.

4. System powiadamia o nieudanej próbie wysłania wiadomości.

FU3. Korzystanie z szablonów.

Wspiera PB1, PB2 i PB4.

Scenariusz główny:

1. System określa, w jaki sposób zdefiniować dane do funkcjonalności.
2. Użytkownik wypełnia dane.
3. System odczytuje wypełniony szablon.
4. System powiadamia o poprawnym odczytaniu szablonu.

Scenariusz alternatywny – szablon niepoprawnie wypełniony:

1-3. Jak w scenariuszu głównym.

4. System powiadamia o nieudanym odczycie szablonu.

FU4. Pobieranie wiadomości przychodzących.

Wspiera PB3. Korzysta z FU1.

Scenariusz główny:

1. Użytkownik zleca pobranie wiadomości przychodzących.

2. System odczytuje specyfikację listy wiadomości, którą ma pobrać.
3. System pobiera wiadomości przychodzące na Teams odpytując MS Graph API przez request GET.
4. System potwierdza poprawne pobranie listy wiadomości.

FU5. Tworzenie i zarządzanie kanałami zespołów.

Wspiera PB2. Korzysta z FU1 i FU3.

Scenariusz główny:

1. Użytkownik wypełnia szablon korzystając z FU3.
2. System ustala, jakie kanały trzeba utworzyć.
3. System wysyła request POST do MS Graph API w celu utworzenia kanału.
4. System potwierdza utworzenie kanału.

Scenariusz alternatywny – kanał o tej nazwie już istnieje:

1-4. Jak w scenariuszu głównym.
5. System powiadamia o duplikacie nazwy kanału.

FU6. Zarządzanie członkami kanałów.

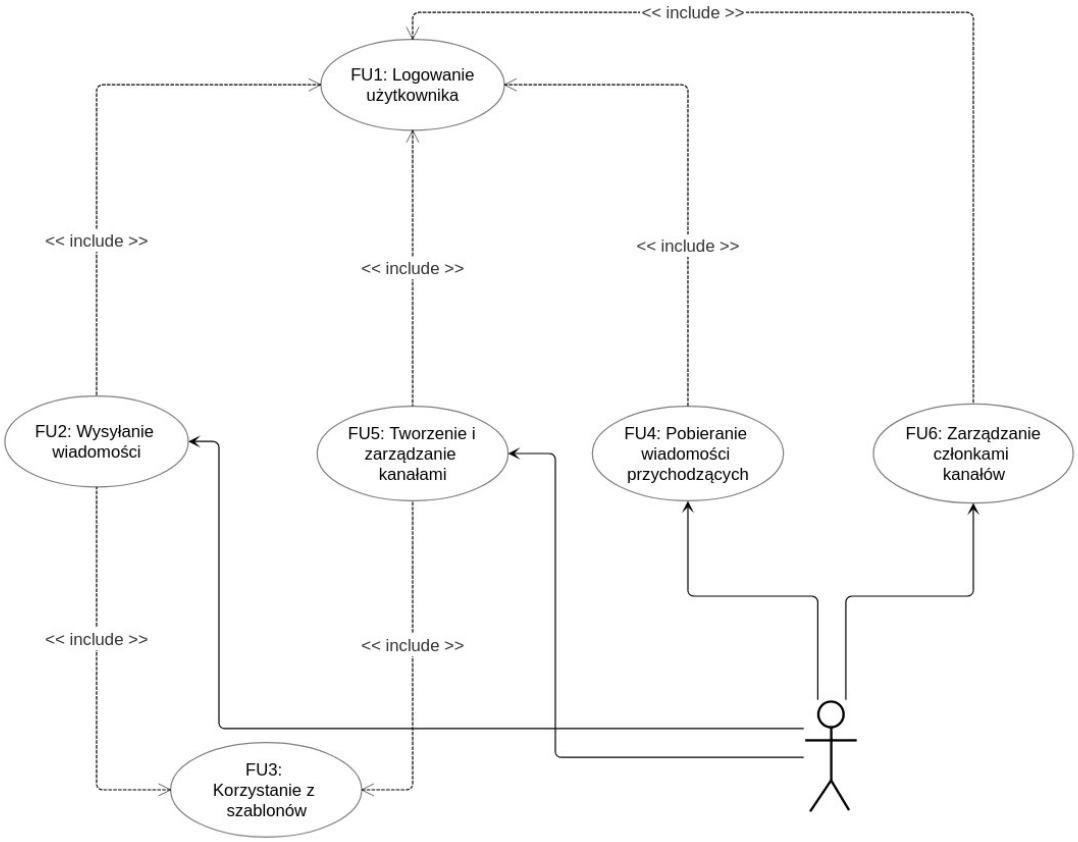
Wspiera PB2. Korzysta z FU1.

Scenariusz główny:

1. Użytkownik wskazuje kanał.
2. Użytkownik wskazuje listę członków do dodania.
3. System wysyła odpowiednie zapytania do MS Graph API.
4. System potwierdza poprawną aktualizację członków kanału.

Scenariusz alternatywny – użytkownik nie istnieje:

1-3. Jak w scenariuszu głównym.
4. System powiadamia o błędnych kontach.



3.4 Potwierdzenie zgodności wymagań

Należy wkleić wycinek ekranu pokazujący korespondencję z Mentorem i Właścicielem tematu.

Nowak Robert czwartek 08:55
Dzień dobry, ja już wcześniej zaakceptowałem wymagania i podtrzymuję pozytywną ocenę tego punktu.

Szlenk Marcin piątek 12:51
Zespół 1 [MS] - teams Ja również potwierdzam akceptację wymagań.

4 Definicja architektury

[plan struktury systemu – model komponentów, modułów, serwisów

zastosowane szablony architektoniczne (np. MVC, mikroserwisy, SOA, szyna usług)

interfejsy kluczowych elementów struktury oraz wyjaśnienie połączeń i interakcji pomiędzy nimi

perspektywa logiczna/funkcjonalna oraz perspektywa sprzętowa (tzw. deployment): systemy operacyjne, serwer aplikacyjny, system bazy danych i inne mechanizmy utrwalania danych, system raportowania , system analityczny/BI, mechanizmy zarządzania, mechanizmy bezpieczeństwa

diagramy]

5 Dane trwałe

5.1 Model logiczny danych

[diagramy]

5.2 Przetwarzanie i przechowywanie danych

[opis planowanego sposobu implementacji: jaka baza danych lub inne oprogramowanie, jak będzie realizowany dostęp do danych (użyte API itp.), czy planowane jest programowanie w b.d. (wyzwalacze, mechanizmy optymalizacji zapytań, itp.)]

6 Specyfikacja analityczna i projektowa

Obowiązkowo odnośnik do repozytorium kodu (jedno repozytorium na projekt, jeżeli więcej proszę uzasadnić)

Obowiązkowo określenie metod realizacji: języki programowania, frameworki, środowisko programowania/ uruchamiania/ wdrażania, środowisko ciągłej integracji]

Obowiązkowo Diagram klas lub model pojęciowy struktury informacyjnej: E-R l

Opcjonalnie model struktury systemu (diagram wdrożenia)

Opcjonalnie specyfikacja realizacji przypadków użycia: diagramy sekwencji lub współpracy

Obowiązkowo statystyki: licaba plików, linie kodu, liczba testów jednostkowych

7 Projekt standardu interfejsu użytkownika

[wykorzystanie narzędzi do modelowania oraz tworzenia makiet warstwy prezentacyjnej (np. storyboards, wireframes, wireflows, mockups, prototypes etc)]

8 Specyfikacja testów

[standardy obsługi błędów i sytuacji wyjątkowych

rodzaje testów, specyfikacja i opis sposobu realizacji poszczególnych rodzajów testów, scenariusze testowe

miary jakości testów]

9 Wirtualizacja/konteneryzacja

10 Bezpieczeństwo

11 Podręcznik użytkownika

[instrukcja użycia funkcjonalności systemu]

12 Podręcznik administratora

[- instrukcja budowy systemu z kodu źródłowego

- instrukcja instalacji i konfiguracji systemu

- instrukcja aktualizacji oprogramowania

- instrukcja zarządzania użytkownikami i uprawnieniami

- instrukcja tworzenia kopii zapasowych i odtwarzania systemu

- instrukcja zarządzania zasobami systemu]

13 Podsumowanie

[Krytyczna analiza osiągniętych wyników, mocne i słabe strony

Możliwe kierunki rozwoju]

14 Bibliografia

[Wykaz materiałów źródłowych, opis zgodny ze standardem sporządzania opisów bibliograficznych - https://bg.pw.edu.pl/index.php/przypisy-i-bibliografia]

Zatwierdzam dokumentację.	…… Data i podpis Mentora