



Uniwersalny PUSHER ORLEN Paczka

1 SPIS TREŚCI

1	Spis treści.....	1
2	Historia zmian dokumentu	2
3	Wprowadzenie	3
4	Zakres dokumentu.....	3
5	Architektura i sposób działania	4
5.1	Proces wdrożenia	4
5.2	Wywołania do API ORLEN Paczka.....	4
5.3	Zachowanie przy niedostępności systemu klienta	4
5.4	Adresy IP serwerów wysyłających powiadomienia (Push).....	5
6	Autoryzacja i wymagania po stronie klienta	5
6.1	Dane dostępowe do API ORLEN Paczka (SOAP)	5
6.2	Typy autoryzacji endpointu klienta	5
6.2.1	Basic.....	6
6.2.2	OAuth2	6
6.2.3	OAuth2StaticToken.....	6
6.3	Wymagania dla endpointu klienta	6
7	Implementacja.....	7
7.1	WebhookRegister	7
7.1.1	Wywołanie.....	7
7.1.2	Odpowiedź.....	8
7.1.3	Przykładowe błędy rejestracji.....	9
7.2	WebhookStatus	9
7.2.1	Wywołanie.....	9
7.2.2	Odpowiedź.....	10
7.3	WebhookUnregister	11
7.3.1	Wywołanie.....	11

7.3.2	Odpowiedź.....	11
8	Format zwracanych danych (payload powiadomienia).....	12
9	Scenariusz wdrożenia i testów	13
9.1	Przygotowanie.....	13
9.2	Test rejestracji – Basic	13
9.3	Test rejestracji – OAuth2	13
9.4	Test rejestracji – OAuth2StaticToken	14
9.5	Test statusu i wyrejestrowania.....	14
10	Zalecenia i dobre praktyki po stronie klienta	14
11	Wsparcie.....	15
12	Przykład endpointu w języku C#.....	15

2 HISTORIA ZMIAN DOKUMENTU

Data	Wersja	Zmiany
04.2026	1.1	Dodanie rozdziału - 5.4 Adresy IP serwerów wysyłających powiadomienia (Push)
11.2025	1.0	Pierwsze wydanie

3 WPROWADZENIE

Uniwersalny Pusher, wewnątrz nazywany **Pack Event Notifier**, jest mechanizmem typu *webhook*, służącym do przekazywania klientowi statusów przesyłek w czasie rzeczywistym. Zamiast cyklicznego odpytywania **API ORLEN Paczka (SOAP)** o statusy, system ORLEN Paczka sam wysyła powiadomienie za każdym razem, gdy status paczki ulegnie zmianie.

Korzyści biznesowe:

- natychmiastowe otrzymywanie informacji o zmianie statusu (*push* wykonywany jest w momencie pojawienia się nowego statusu),
- redukcja ruchu sieciowego po stronie API,
- uproszczenie integracji po stronie klienta (brak potrzeby budowania harmonogramów odpytywania dla metod śledzących statusy).

Aby skorzystać z **Uniwersalnego Pushera**, należy:

1. Udostępnić własny **REST-owy endpoint** (adres URL), który będzie odbierał powiadomienia,
2. Zarejestrować ten **endpoint** w systemie ORLEN Paczka za pomocą metody *WebhookRegister*.

Obsługiwane typy autoryzacji po stronie endpointu klienta:

- **Basic** – login i hasło,
- **OAuth2** – pozyskanie tokena dostępowego z dedykowanego endpointu klienta,
- **OAuth2StaticToken** – statyczny token typu *Bearer* przekazany przez klienta.

4 ZAKRES DOKUMENTU

Niniejszy dokument jest przeznaczony dla zespołów developerskich klientów **ORLEN Paczka**, którzy będą wdrażać integrację **Uniwersalnego Pushera** w swoich systemach. Jest on uzupełnieniem dla dokumentacji **API ORLEN Paczka (SOAP)** (Dostępna na stronie <https://www.orldenpaczka.pl/integracja-z-orlden-paczka/> w sekcji **Dokumentacja API**).

Opisuje on:

- kontekst biznesowo-techniczny działania **Uniwersalnego Pushera**,
- sposoby autoryzacji po stronie klienta,
- szczegółowe użycie metod:
 - *WebhookRegister*,
 - *WebhookStatus*,
 - *WebhookUnregister*,
- zalecany scenariusz wdrożenia i testów,

- format danych przesyłanych w powiadomieniach.

5 ARCHITEKTURA I SPOSÓB DZIAŁANIA

5.1 PROCES WDROŻENIA

1. Klient posiada aktywną umowę z **ORLEN Paczka** i korzysta z **API ORLEN Paczka (SOAP)**.
2. Klient tworzy w swoim systemie **REST-owy endpoint HTTP(S)**, który akceptuje żądania **POST** z treścią w formacie **JSON**.
3. Klient rejestruje ten endpoint wywołując metodę *WebhookRegister* w **API ORLEN Paczka (SOAP)**.
4. Po poprawnej rejestracji system **ORLEN Paczka**:
 - wysyła **testowe powiadomienie** dla paczki o numerze np. **99999999999999**, aby potwierdzić poprawność konfiguracji,
 - następnie, dla każdej paczki obsługiwanej w ramach danego **PartnerID**, wysyła powiadomienie przy każdej zmianie statusu.
5. Klient może w dowolnym momencie:
 - sprawdzić aktualną konfigurację pushera metodą *WebhookStatus*,
 - wyrejestrować endpoint metodą *WebhookUnregister* (Metody opisane w kolejnych rozdziałach).

5.2 WYWOŁANIA DO API ORLEN PACZKA

Metody Pushera są dostępne w publicznym **API ORLEN Paczka (SOAP)**.

- **Środowisko produkcyjne**
<https://api.ornlenpaczka.pl/WebServicePwRProd/WebServicePwR.asmx?wsdl>
- **Środowisko testowe:**
<https://api-test.ornlenpaczka.pl/WebServicePwR/WebServicePwR.asmx?WSDL>

Autoryzacja do **API ORLEN Paczka (SOAP)** odbywa się – tak jak w innych metodach **API ORLEN Paczka (SOAP)** – poprzez przekazanie w żądaniu pól **<PartnerID>** oraz **<PartnerKey>**.

5.3 ZACHOWANIE PRZY NIEDOSTĘPNOŚCI SYSTEMU KLIENTA

Jeżeli system **ORLEN Paczka** nie otrzyma od **endpointu** klienta odpowiedzi **HTTP 200**:

- powiadomienie jest uznawane za **nieprzyjęte**,
- system ponawia wysyłkę **co 5 minut**, do skutku (aż **endpoint** zwróci status **HTTP 200**) lub do chwili wyrejestrowania **pushera** przez klienta,
- w przypadku wyrejestrowania **pushera** za pomocą *WebhookUnregister*, **kolejka oczekujących powiadomień jest czyszczona**.

5.4 ADRESY IP SERWERÓW WYSYŁAJĄCYCH POWIADOMIENIA (PUSH)

Poniżej znajduje się pula adresów IP (w formacie CIDR), z których system ORLEN Paczka wysyła powiadomienia o statusach przesyłek do endpointu klienta. Prosimy o dodanie poniższego zakresu do białej listy (allowlist) na Państwa firewallu, aby umożliwić prawidłowe odbieranie komunikatów:

91.242.220.0/23

6 AUTORYZACJA I WYMAGANIA PO STRONIE KLIENTA

6.1 DANE DOSTĘPOWE DO API ORLEN PACZKA (SOAP)

Dane logowania do **API ORLEN Paczka (SOAP)**:

- *<PartnerID>* – login klienta nadany przez **ORLEN Paczka**,
- *<PartnerKey>* – hasło klienta nadane przez **ORLEN Paczka**.

Dane do środowiska produkcyjnego są przekazywane przez opiekuna handlowego.

6.2 TYPY AUTORYZACJI ENDPOINTU KLIENTA

W parametrach metody *WebhookRegister* klient wskazuje sposób autoryzacji swojego endpointu za pomocą pola *<AuthorizationType>*:

- *Basic*
- *OAuth2*
- *OAuth2StaticToken*

Dla każdego typu wymagany jest inny zestaw pól:

Pole	Basic (AuthorizationType=Basic)	OAuth2 (AuthorizationType=OAuth2)	OAuth2StaticToken (AuthorizationType=OAuth2StaticToken)
<i>UserName</i>	login użytkownika (wymagane)	ClientId (wymagane)	null (nie używane)
<i>Password</i>	hasło (wymagane)	hasło/ClientSecret (wymagane)	token (wymagane)

LoginUrl	null (nie używane)	adres URL do pobrania tokena (wymagane)	null (nie używane)
NotificationUrl	adres webhooka (wymagane)	adres webhooka (wymagane)	adres webhooka (wymagane)

6.2.1 Basic

- **ORLEN Paczka** wysyła powiadomienie *POST* na *NotificationUrl* z nagłówkiem:
 - *Authorization: Basic <base64(Username:Password)>*.
- Po stronie klienta **endpoint** powinien obsłużyć standardową autoryzację **Basic**.

6.2.2 OAuth2

- **ORLEN Paczka** przed wysłaniem powiadomienia:
 1. wywołuje *LoginUrl* klienta, przekazując dane *Username (ClientId)* i *Password (ClientSecret)* wysłane w zapytaniu metody *WebhookRegister*
 2. otrzymuje token dostępu (**access token**),
 3. używa go w nagłówku:
 - *Authorization: Bearer <access_token>*.

6.2.3 OAuth2StaticToken

- Klient przekazuje w polu *Password* statyczny token (np. *eyJhbGciOi...*).
- System **ORLEN Paczka** nie pobiera tokenu dynamicznie – używa zawsze wartości przekazanej w rejestracji:
 - *Authorization: Bearer <Password>*.

6.3 WYMAGANIA DLA ENDPOINTU KLIENTA

Endpoint klienta (**NotificationUrl**) powinien:

- przyjmować żądania **HTTP POST** z zawartością w formacie **JSON** (UTF-8),
- w przypadku poprawnego przetworzenia powiadomienia zwracać **HTTP 200** (bez względu na logikę wewnętrzną – np. zapis do bazy/logów),
- w razie błędu zwrócić kod inny niż **HTTP 200** – system **ORLEN Paczka** potraktuje to jako nieprzyjęcie i będzie ponawiał wysyłkę,
- być dostępny pod protokołem **HTTPS** (zalecane),
- posiadać limit czasu odpowiedzi (**timeout**) umożliwiający zwrócenie komunikatu w optymalnym czasie (np. do **10 sekund**).

7 IMPLEMENTACJA

Metody *WebhookRegister*, *WebhookStatus*, *WebhookUnregister* dostępne są w **API ORLEN Paczka (SOAP)**.

Środowisko produkcyjne:

<https://api.orientpaczka.pl/WebServicePwRProd/WebServicePwR.asmx?wsdl>

Środowisko testowe:

<https://api-test.orientpaczka.pl/WebServicePwR/WebServicePwR.asmx?WSDL>

7.1 WEBHOOKREGISTER

Metoda *WebhookRegister* służy do zarejestrowania nowego endpointu klienta w systemie **ORLEN Paczka**. Jest to pierwszy krok konieczny do rozpoczęcia otrzymywania powiadomień o statusach paczek. Po prawidłowo przeprowadzonej rejestracji wykonywany jest **push** dla przesyłki o numerze **999999999999**.

Przykład:

```
"PackCode":999999999999,"PackCodePrev":null,"Status":"200","Updated":"2000-01-01T00:00:00","Operator":"ORLEN-PACZKA"
```

7.1.1 Wywołanie

Parametry wywołania:

<PartnerID>	Identyfikator Klienta nadany przez ORLEN Paczka	(REQUIRED) 10 char
<PartnerKey>	Hasło Klienta nadane przez ORLEN Paczka	(REQUIRED) 10 char
<AuthorizationType>	Typy autoryzacji - Dopuszczalne wartości: <ul style="list-style-type: none">• Basic• OAuth2• OAuth2StaticToken	(REQUIRED) 30 char
<UserName>	<ul style="list-style-type: none">• dla Basic – login użytkownika,• dla OAuth2 – ClientId,• dla OAuth2StaticToken - nieużywane (null / pusty).	30 char
<Password>	<ul style="list-style-type: none">• dla Basic – hasło użytkownika,• dla OAuth2 – hasło / ClientSecret,• dla OAuth2StaticToken – token (np. Bearer) używany bezpośrednio w nagłówku Authorization.	70 char
<LoginUrl>	<ul style="list-style-type: none">• dla OAuth2 – adres URL do pozyskania tokena dostępowego,	100 char

	<ul style="list-style-type: none"> dla <i>Basic</i> i <i>OAuth2StaticToken</i> – nieużywane (null / puste). 	
<NotificationUrl>	Adres URL webhooka klienta, na który wysyłane będą powiadomienia o zmianie statusu paczek.	(REQUIRED) 100 char

Wymagane pola przy poszczególnych typach autoryzacji <AuthorizationType>:

AuthorizationType	Basic	OAuth2	OAuth2StaticToken
UserName	UserName (required)	ClientId (required)	null
Password	Password (required)	Password (required)	Token (required)
LoginUrl	null	URL logowania (required)	null
NotificationUrl	URL do zdarzeń (required)	URL do zdarzeń (required)	URL do zdarzeń (required)

Przykład z minimalną wymaganą listą pól – autoryzacja Basic:

```

<PartnerID>1234567890</PartnerID>
<PartnerKey>abcdefghijk</PartnerKey>
<AuthorizationType>Basic</AuthorizationType>
<UserName>login</UserName>
<Password>password</Password>
<LoginUrl></LoginUrl>
<NotificationUrl>https://twoj-system.example.com/pack-events/notification</NotificationUrl>

```

7.1.2 Odpowiedź

Zwracane wartości:

<Err>	Numer błędu ('0' oznacza brak błędu)
<ErrDes>	Opis błędu (pusty, jeśli brak błędu)
<Data>	Sekcja z informacjami o zarejestrowanym webhooku

<Data> struktura	
<PartnerID>	Identyfikator Klienta
<Status>	Status rejestracji ('Active', 'Unregistered', itp.)
<Annotation>	Opis rejestracji (np. 'Webhook registered and activated')
<AuthorizationType>	Typ autoryzacji
<UserName>	Login do endpointu klienta / ClientId
<Password>	Hasło lub token do endpointu klienta
<LoginUrl>	adres URL do logowania (dla OAuth2)
<NotificationUrl>	adres webhooka klienta

Przykład:

```

<Err>0</Err>
<ErrDes />
<Data>
<PartnerID>1234567890</PartnerID>
<Status>Active</Status>
<Annotation>Webhook registered and activated</Annotation>

```



```

<AuthorizationType>Basic</AuthorizationType>
<UserName>login</UserName>
<Password>password</Password>
<NotificationUrl>http://domain.com/PackEventNotifierBasicExample/notification</NotificationUrl>
</Data>

```

7.1.3 Przykładowe błędy rejestracji

Typowe problemy wykrywane podczas testów:

1. **Nieprawidłowe dane logowania do endpointu klienta**
 - Np. błędny *UserName* / *Password* (Basic) lub błędny token (*OAuth2StaticToken*).
 - **Skutek:** system klienta odrzuca żądania np. **HTTP 401 Unauthorized**.
 - W logach rejestracji pushera pojawia się wpis, że rejestracja zakończyła się statusem *incorrect* oraz informacja typu:
request failed, received HTTP code Unauthorized (Unauthorized).
2. **Prawidłowe dane autoryzacyjne, ale błędny adres URL**
 - *NotificationUrl* wskazuje nieistniejący **endpoint**.
 - **Skutek:** system klienta zwraca np. **HTTP 404 Not Found**.
 - Przykładowy komunikat:
request failed, received HTTP code NotFound (Not Found).
3. **Brak wymaganych pól w zależności od AuthorizationType**
 - Brak *UserName* lub *Password* przy *Basic*,
 - Brak *LoginUrl* przy *OAuth2*,
 - Brak *Password* (tokenu) przy *OAuth2StaticToken*,
 - Błędna wartość *AuthorizationType* (inna niż *Basic*, *OAuth2*, *OAuth2StaticToken*).

W każdym z powyższych przypadków **Err** będzie różny od **0**, a szczegóły błędu pojawią się w **ErrDes**.

7.2 WEBHOOKSTATUS

Metoda *WebhookStatus* służy do sprawdzenia aktualnej konfiguracji i statusu zarejestrowanego endpointu. Może być wykorzystywana zarówno:

- podczas wdrożenia (weryfikacja czy rejestracja przebiegła poprawnie),
- w trakcie utrzymania (monitoring – np. w cyklicznych checkach integracyjnych).

7.2.1 Wywołanie

Parametry wywołania:

<PartnerID>	Identyfikator Klienta nadany przez ORLEN Paczka	(REQUIRED) 10 char
-------------	-------------------------------------------------	--------------------

<PartnerKey>	Hasło Klienta nadane przez ORLEN Paczka	(REQUIRED) 10 char
--------------	--------------------------------------------	--------------------

Przykład z minimalną wymaganą listą pól:

```
<PartnerID>1234567890</PartnerID>
<PartnerKey>abcdefghijk</PartnerKey>
```

7.2.2 Odpowiedź

Zwracane wartości:

<Err>	Numer błędu
<ErrDes>	Opis błędu
<Data>	Struktura <Data> zwracająca informacje

<Data> struktura	
<PartnerID>	Identyfikator Klienta nadany przez ORLEN Paczka
<Status>	Status pushera (<i>Active</i> lub <i>Unregistered</i>)
<Annotation>	Opis aktualnego statusu (np. <i>Webhook registered and activated</i>)
<AuthorizationType>	Typ autoryzacji
<UserName>	Zgodne z aktualną konfiguracją
<Password>	Zgodne z aktualną konfiguracją
<LoginUrl>	Zgodne z aktualną konfiguracją
<NotificationUrl>	Zgodne z aktualną konfiguracją

Przykład (pusher aktywny):

```
<Err>0</Err>
<ErrDes />
<Data>
<PartnerID>1234567890</PartnerID>
<Status>Active</Status>
<Annotation>Webhook registered and activated</Annotation>
<AuthorizationType>Basic</AuthorizationType>
<UserName>login</UserName>
<Password>password</Password>
<NotificationUrl>http://domain.com/PackEventNotifierBasicExample/notification</NotificationUrl>
</Data>
```

Przykład (pusher wyrejestrowany):

```
<Err>0</Err>
<ErrDes />
<Data>
<PartnerID>1234567890</PartnerID>
<Status>Unregistered</Status>
<Annotation>Webhook unregistered</Annotation>
<AuthorizationType>Basic</AuthorizationType>
<UserName>login</UserName>
<Password>password</Password>
```

```
<NotificationUrl>http://domain.com/PackEventNotifierBasicExample/notification</NotificationUrl>
</Data>
```

7.3 WEBHOOKUNREGISTER

Metoda *WebhookUnregister* służy do **wyłączenia** powiadomień Uniwersalnego Pushera dla danego **PartnerID**. Po wyrejestrowaniu:

- system **ORLEN Paczka** przestaje wysyłać nowe powiadomienia,
- kolejka oczekujących powiadomień (które nie zostały jeszcze skutecznie doręczone) zostaje wyczyszczona.

7.3.1 Wywołanie

Parametry wywołania:

<PartnerID>	Identyfikator Klienta nadany przez ORLEN Paczka	(REQUIRED) 10 char
<PartnerKey>	Hasło Klienta nadane przez ORLEN Paczka	(REQUIRED) 10 char

Przykład z minimalną wymaganą listą pól:

```
<PartnerID>1234567890</PartnerID>
<PartnerKey>abcdefghijk</PartnerKey>
```

7.3.2 Odpowiedź

Zwracane wartości:

<Err>	Numer błędu
<ErrDes>	Opis błędu
<Data>	Struktura <Data> zwracająca informacje

<Data> struktura	
<PartnerID>	Identyfikator Klienta nadany przez ORLEN Paczka
<Status>	Status pushera
<Annotation>	Opis aktualnego statusu
<AuthorizationType>	Typ autoryzacji
<UserName>	konfiguracja, która została wyłączona
<Password>	konfiguracja, która została wyłączona
<LoginUrl>	konfiguracja, która została wyłączona
<NotificationUrl>	konfiguracja, która została wyłączona

Przykład:

```
<Err>0</Err>
<ErrDes />
<Data>
```

```

<PartnerID>1234567890</PartnerID>
<Status>Unregistered</Status>
<Annotation>Webhook unregistered</Annotation>
<AuthorizationType>Basic</AuthorizationType>
<UserName>login</UserName>
<Password>password</Password>
<NotificationUrl>http://domain.com/PackEventNotifierBasicExample/notification</NotificationUrl>
</Data>

```

8 FORMAT ZWRACANYCH DANYCH (PAYLOAD POWIADOMIENIA)

Po każdej zmianie statusu paczki system **ORLEN Paczka** wysyła na *NotificationUrl* żądanie **POST** z ciałem w formacie **JSON**

```

{
  "PackCode": NUMER_PACZKI,
  "PackCodePrev": NUMER_PACZKI_PIERWOTNEJ,
  "Status": STATUS+ATRYBUT,
  "Updated": DATA_CZAS,
  "Operator": OPERATOR
}

```

Opis pól:

- **PackCode:**
Numer paczki (EAN-13), 13-cyfrowa liczba.
- **PackCodePrev:**
Numer paczki pierwotnej (EAN-13), 13-cyfrowa liczba lub null, jeśli brak powiązania.
- **Status:**
Numer statusu przesyłki (np. 200, 210 itp.) – zgodny z listą statusów **ORLEN Paczka**.
ATRYBUT:
 - brak atrybutu – Dla przesyłek standardowych.
Np. „Status”: **200**
 - **_POWROT** lub **_2_POWROT** - Atrybuty zwrotu operacyjnego.
Np. „Status”: **100_POWROT** lub **100_2_POWROT**
 - **_ZWROT** - Atrybut zwrotu konsumenckiego
Np. „Status”: **100_ZWROT**
- **Updated:**
Data i czas aktualizacji statusu w formacie **ISO 8601**, np.
2025-06-26T01:02:03.225694.
Czas zwracany jest w strefie **UTC+1 (lokalny czas polski)**.
- **Operator**
Nazwa operatora: "ORLEN-PACZKA".

Przykład pełnego zdarzenia:

```

{
  "PackCode": 1234567890123,
  "PackCodePrev": null,

```

```
"Status": "200",  
"Updated": "2025-06-26T01:02:03.225694",  
"Operator": "ORLEN-PACZKA"  
}
```

Pełna lista statusów wraz z opisami znajduje się w dokumentacji **API ORLEN Paczka (SOAP)** w sekcji **7. STATUSY PACZEK**.

Dokumentacja dostępna na stronie <https://www.orklenpaczka.pl/integracja-z-orklen-paczka/> w sekcji **Dokumentacja API**.

9 SCENARIUSZ WDROŻENIA I TESTÓW

Poniższy scenariusz jest streszczeniem rzeczywistego wdrożenia i może być zastosowany do dowolnej integracji.

9.1 PRZYGOTOWANIE

1. Odbierz od opiekuna handlowego:
 - o PartnerID i PartnerKey dla środowiska testowego,
2. Po swojej stronie przygotuj:
 - o endpoint NotificationUrl (*REST, HTTPS, POST JSON*),
 - o obsługę wybranej autoryzacji (*Basic / OAuth2 / OAuth2StaticToken*),
 - o logowanie wszystkich przychodzących powiadomień (payload, nagłówki, kody odpowiedzi).

9.2 TEST REJESTRACJI – BASIC

1. Wywołaj w środowisku testowym metodę *WebhookRegister* z:
 - o *AuthorizationType = Basic*,
 - o wypełnionymi polami *UserName, Password, NotificationUrl*.
2. Sprawdź odpowiedź:
 - o *Err = 0*,
 - o *Status = Active*,
 - o poprawny *NotificationUrl*.
3. Zweryfikuj logi swojego systemu:
 - o powinno pojawić się powiadomienie testowe dla paczki **99999999999999**.
4. Zaawizuj testową paczkę na swoim *PartnerID* (np. z wykorzystaniem metod nadawczych API) i sprawdź, czy otrzymałeś dla swojej przesyłki status **200**.

9.3 TEST REJESTRACJI – OAUTH2

1. Skonfiguruj po swojej stronie endpoint *LoginUrl*, który generuje token dostępu.

2. Wywołaj *WebhookRegister* z:
 - *AuthorizationType* = *OAuth2*,
 - *UserName* = *ClientId*,
 - *Password* = *ClientSecret*,
 - *LoginUrl* = *adres endpointu do pobrania tokena*,
 - *NotificationUrl* = *adres webhooka*.
3. Zweryfikuj odpowiedź oraz powiadomienie testowe.
4. Przetestuj poprawne oraz niepoprawne dane (zły *LoginUrl*, zły *NotificationUrl*, błędny *UserName*), aby zaobserwować odpowiednie błędy.

9.4 TEST REJESTRACJI – OAUTH2STATICTOKEN

1. Przygotuj statyczny token, który będzie używany w nagłówku *Authorization: Bearer*
2. Wywołaj *WebhookRegister* z:
 - *AuthorizationType* = *OAuth2StaticToken*,
 - *Password* = *token*,
 - *NotificationUrl* = *adres webhooka*,
 - *UserName* i *LoginUrl* – puste (*null*).
3. Zweryfikuj odbiór powiadomień testowych i statusowych.

9.5 TEST STATUSU I WYREJESTROWANIA

1. Po udanej rejestracji wywołaj *WebhookStatus* i upewnij się, że:
 - *Status* = *Active*,
 - *AuthorizationType*, *NotificationUrl* i pozostałe pola są zgodne z konfiguracją.
2. W kolejnym kroku wywołaj *WebhookUnregister*.
3. Ponownie wywołaj *WebhookStatus*:
 - Status powinien przyjąć wartość *Unregistered*.
4. Nadaj testową paczkę i upewnij się, że **nie otrzymujesz już powiadomień** – oznacza to poprawne wyłączenie pushera.

10 ZALECENIA I DOBRE PRAKTYKI PO STRONIE KLIENTA

Jednokrotność

Implementacja endpointu powinna być **jednokrotna** - jeśli to samo zdarzenie zostanie wysłane więcej niż raz (np. w wyniku ponownej próby), system klienta nie powinien tworzyć duplikatów ani wielokrotnie wykonywać tej samej akcji biznesowej.

Logowanie i monitoring

Zalecane jest logowanie:

- daty/czasu odebrania powiadomienia,

- pełnego payloadu **JSON**,
- nagłówek (min. *Authorization*, *CorrelationId* jeśli używany),
- kodu odpowiedzi systemu klienta.

Bezpieczeństwo

- Endpoint powinien działać w **HTTPS**,
- dane logowania/tokeny powinny być przechowywane w bezpieczny sposób (np. w managerze sekretów),

Obsługa błędów

W przypadku wewnętrznych błędów (np. niedostępność bazy) **endpoint** powinien zwrócić kod inny niż **HTTP 200**, aby pozwolić systemowi **ORLEN Paczka** na ponowną próbę dostarczenia zdarzenia. Sugerujemy wykorzystanie kodów z rodziny **HTTP 400**.

11 WSPARCIE

W przypadku pytań dotyczących:

- konfiguracji **Uniwersalnego Pushera**,
- doboru typu autoryzacji,
- szczegółów dotyczących statusów przesyłek,

prosimy o kontakt z Opiekunem Handlowym **ORLEN Paczka** lub działem wsparcia technicznego (integracje@orlenpaczka.pl), z podaniem:

- numeru klienta (**PartnerID**),
- środowiska (test / produkcja),
- krótkiego opisu problemu,
- przykładowej paczki (**PackCode**) i przedziału czasowego wystąpienia problemu.

12 PRZYKŁAD ENDPOINTU W JĘZYKU C#

Dostępny jest w pliku „PublicUniversalPusherExample.zip”.



ORLEN Paczka Pack Event Notifier Basic Authorization

Projekt `PackEventNotifierBasicExample` to przykładowa implementacja endpointu na który ORLEN Paczka może przysyłać dane autoryzując się określoną nazwą użytkownika i hasłem.

Test lokalny

1. Otwórz w Postmanie [OrlenPaczkaPackEventNotifierExample.postman_collection.json](#)
2. Przejdź do `OrlenPaczkaPackEventNotifierExample > Basic Example`
3. Jeżeli trzeba podmień adres w requście na `http://domena.com/PackEventNotifierBasicExample/notification`
4. Zdarzenie zostanie wyświetlone na panelu poniżej
5. W celu symulacji stanu serwera użyj przełącznika 'Akceptuj notyfikacje'

☒ Akceptuj notyfikacje

Ostatnie odświeżenie:
28.11.2025, 13:38:48

```
2025-11-28 13:37:59Z OK {"PackCode":99999999999999,"PackCodePrev":null,"Status":"200","Updated":"2000-01-01T00:00:00","Operator":"ORLEN-PACZKA"}
```

Checkbox „Akceptuj notyfikacje” pozwala przetestować blokadę przyjmowania komunikatów **push** dla wystawionego **endpointu**. Po odznaczeniu wysłane zapytania nie będą odbierane i na liście pojawią się komunikaty z opisem **REJECT** (zastąpią komunikat **OK** widoczny na zrzucie ekranu).