Dokumentacja techniczna integracji z systemem transakcyjnym imoje

System transakcyjny imoje

Wersja dokumentu

Wersja	Data	Autor	Kontakt
1.0.0	14.03.2018	ING	kontakt.tech@imoje.pl
1.1.0	11.05.2018	ING	kontakt.tech@imoje.pl
1.2.0	28.06.2018	ING	kontakt.tech@imoje.pl
1.3.0	08.02.2019	ING	kontakt.tech@imoje.pl
1.3.1	12.03.2019	ING	kontakt.tech@imoje.pl
1.3.2	29.05.2019	ING	kontakt.tech@imoje.pl
1.3.3	29.07.2019	ING	kontakt.tech@imoje.pl

Spis treści

1. Wprowadzenie

Dokument zawiera informacje na temat RESTful API bramki płatności imoje.

2. Statusy transakcji

Transakcje mogą przyjmować następujące statusy:

Status	Opis
`new`	Nowa, nieobsłużona transakcja
`authorized`	Autoryzacja transakcji
`pending`	Oczekiwanie na status
`submitted`	Wysłana do realizacji
`rejected`	Transakcja odrzucona
`settled`	Transakcja zrealizowana
`error`	Błąd w transakcji
`canceled`	Transakcja anulowana

3. Metody autoryzacji

Bramka pozwala na autoryzację za pomocą tokenu autoryzacyjnego. Token można znaleźć w panelu administracyjnym imoje, w części `Ustawienia` a następnie w zakładce `Klucze API`.

3.1. Token autoryzacyjny

Aby dokonać autoryzacji zapytania należy w nagłówkach żądania do serwera umieścić dane autoryzacyjne:

```
Accept: application/json
Content-Type: application/json
Authorization: Bearer ad8y3hdoashco8fh49fhiahfb237f8hoihsd0f2hfikljf023h8
```

Kody odpowiedzi HTTP:

Kod HTTP	Znaczenie
`200`	Autoryzacja poprawna
`401`	Nieautoryzowany dostęp, żądanie zasobu, który wymaga uwierzytelnienia
`500`	Błąd serwera

4. Metody RESTful API

Base URL: `https://api.imoje.pl/v1/merchant`

Każdy poprawny adres składa się z trzech części:

- adresu bazowego zapytania (`https://api.imoje.pl/v1`),
- identyfikatora klienta (`/merchant/{identyfikator klienta}`),
- funkcji jednoznacznie określającej zakres danych, których dotyczy zapytanie (np. `/transaction` lub `/services`).

Każde zapytanie do serwera powinno zawierać dane autoryzacyjne w nagłówkach (Token autoryzacyjny).

Endpoint	Zastosowanie	Metody
`/{merchantId}/transaction`	Tworzenie nowej transakcji	`POST`
`/{merchantId}/transaction/{transactionId}`	Pobieranie danych transakcji	`GET`
`/{merchantId}/services`	Zwraca informacje o sklepach akceptanta	`GET`
`/{merchantId}/service/{serviceId}`	Zwraca informacje o sklepie akceptanta o określonym `serviceId` oraz możliwych metodach płatności.	`GET`
`/{merchantId}/settings/ips`	Pobierz zaufane adresy IP	`GET`
`/{merchantId}/settings/ips`	Ustaw zaufane adresy IP	`PUT`
`/{merchantId}/stats/transactions/count`	Zwraca liczbę wszystkich transakcji	`GET`
`/{merchantId}/stats/transactions/count/{serviceId}`	Zwraca liczbę wszystkich transakcji sklepu określonej przez `serviceId`	`GET`
`/{merchantId}/stats/transactions/states`	Zwraca liczbę transakcji grupowane po statusach transakcji	`GET`
`/{merchantId}/stats/transactions/states/{serviceId}`	Zwraca liczbę transakcji transakcji sklepu określonej przez `serviceId`	`GET`
`/{merchantId}/payment`	Tworzenie nowego linku płatności	`POST`
`/{merchantId}/transaction/profile`	Obciążenie istniejącego profilu	`POST`

Endpoint	Zastosowanie	Metody
`/{merchantId}/profile/cid/{cid}`	Pobierz profile o podanym `cid`	`GET`
`/{merchantId}/profile/id/{paymentProfileId}`	Pobierz profil o podanym `id`	`GET`
`/{merchantId}/profile/deactivate`	Dezaktywuj profil o podanym w żądaniu identyfikatorze	`POST`
`/{merchantId}/profile/id/{paymentProfileId}`	Dezaktywuj profil o podanym `paymentProfileId`	`DELETE`

gdzie:

- `merchantId` - identyfikator klienta,
- `accessToken` - token autoryzacyjny,
- `serviceId` - identyfikator sklepu z którym związane są określone metody realizacji transakcji (UUID v4),
- `state` - status transakcji,
- `cid` - identyfikator klienta/płatnika nadany przez akceptanta,
- `transactionId` - unikalny identyfikator transakcji (UUID v4),
- `paymentProfileId` - identyfikator profilu.

5. Tworzenie nowej transakcji przez API

> Zgodnie z wymaganiami PCI DSS (ustawionymi przez organizacje płatnicze) zabronione jest przetwarzanie, przekazywanie czy przechowywanie numerów i innych danych dotyczących kart płatniczych czy kredytowych. Jeśli posiadasz właściwy certyfikat PCI DSS i chcesz udostępnić formatkę płatności kartami na stronie Twojego sklepu - prosimy o kontakt z Twoim opiekunem handlowym.

W celu utworzenia nowego zamówienia należy za pomocą metody `POST` przesłać komunikat na endpoint API `https://api.imoje.pl/v1/merchant/{merchantId}/transaction` zawierający informacje o nowym zamówieniu.

gdzie:

- `merchantId` - identyfikator klienta.

5.1. HTTP Request dla płatności PBL

5.1.1. Przykładowy adres na który należy wysłać żądanie POST

```
https://api.imoje.pl/v1/merchant/6yt3gjtm9p7b8h9xsdqz/transaction
```

5.1.2. Nagłówki zapytania

```
Accept: application/json
Authorization: Bearer ad8y3hdoashco8fh49fhiahfb237f8hoihsd0f2hfikljf023h8
Content-Length: 895
Content-Type: application/json
```

5.1.3. Payload zapytania


```
{
  "type": "sale",
  "serviceId": "62f574ed-d4ad-4a7e-9981-89ed7284aaba",
  "amount": 100,
  "currency": "PLN",
  "title": "",
  "orderId": "123123123",
  "paymentMethod": "ing",
  "paymentMethodCode": "ing",
  "successReturnUrl": "https://domain.com/success",
  "failureReturnUrl": "https://domain.com/failure",
  "customer": {
    "firstName": "Jan",
    "lastName": "Kowalski",
    "cid": "123",
    "company": "",
    "phone": "",
    "email": "jan.kowalski@example.com"
  },
  "billing": {
    "firstName": "Jan",
    "lastName": "Kowalski",
    "company": "Company",
    "street": "Street",
    "city": "City",
    "region": "Region",
    "postalCode": "",
    "countryCodeAlpha2": "PL"
  },
  "shipping": {
    "firstName": "Jan",
    "lastName": "Kowalski",
    "company": "Company",
    "street": "Street",
    "city": "City",
    "region": "Region",
    "postalCode": "",
    "countryCodeAlpha2": "PL"
  }
}
```

5.1.4. Parametry payload

Parametr	Typ	Parametr wymagany	Opis
`type`	string	`WYMAGANE`	Typ transakcji. Dopuszczalne wartości: `sale`, `refund`.
`serviceId`	string(36)	`WYMAGANE`	identyfikator sklepu jako UUID v4.
`amount`	integer	`WYMAGANE`	Kwota transakcji w najmniejszej jednostce waluty np. grosze.
`currency`	string(3)	`WYMAGANE`	Waluta transakcji w standardzie ISO 4217.
`orderID`	string(100)	`WYMAGANE`	Numer zamówienia akceptanta - dopuszczalne znaki: od 0 do 9, od a do z, od A do Z, znak spacji (0x20).
`title`	string(255)	`NIE`	Tytuł zamówienia - dopuszczalne znaki: od 0 do 9, od a do z, od A do Z, znak spacji (0x20).
`paymentMethod`	string	`WYMAGANE`	Metoda realizacji zamówienia. Dopuszczalne są wartości: `pbl`, `card`, `blik`, `ing`.
`paymentMethodCode`	string	`WYMAGANE`	Oznaczenie kanału płatności. Szczegóły opisane są w punkcie 5.2.
`successReturnUrl`	string(300)	`WYMAGANE`	Adres powrotu z zewnętrznej strony obsługującej płatność w przypadku dokonania płatności z powodzeniem.
`failureReturnUrl`	string(300)	`WYMAGANE`	Adres powrotu z zewnętrznej strony obsługującej płatność w przypadku wystąpienia błędu płatności.
`customer`	object	`WYMAGANE`	Dane klienta.
`billing`	object	`NIE`	Dane płatnika.
`shipping`	object	`NIE`	Dane dot. dostawy.
`card`	object	`NIE`	Dane dot. płatności kartą. (Wymagane podczas płatności `card`)

Jeżeli w zapytaniu wystąpi parametr `customer`, `billing`, lub `shipping` konieczne jest dostarczenie parametrów:

5.1.4.1. customer

Parametr	Typ	Parametr wymagany	Opis
`firstName`	string(100)	`WYMAGANE`	Imię klienta - dopuszczalne znaki: od 0 do 9, od a do z, od A do Z, znak spacji (0x20).
`lastName`	string(100)	`WYMAGANE`	Nazwisko klienta - dopuszczalne znaki: od 0 do 9, od a do z, od A do Z, znak spacji (0x20).
`cid`	string(100)	`NIE`	Identyfikator klienta. (Wymagane podczas płatności `oneclick`, `recurring`) - dopuszczalne znaki: od 0 do 9, od a do z, od A do Z, myślnik (0x2D)
`company`	string(200)	`NIE`	Nazwa firmy - dopuszczalne znaki: od 0 do 9, od a do z, od A do Z, znak spacji (0x20).
`phone`	string(20)	`NIE`	Numer telefonu.
`email`	string(200)	`WYMAGANE`	Adres email.

5.1.4.2. billing

Parametr	Typ	Parametr wymagany	Opis
`firstName`	string(100)	`WYMAGANE`	Imię klienta - dopuszczalne znaki: od 0 do 9, od a do z, od A do Z, znak spacji (0x20).
`lastName`	string(100)	`WYMAGANE`	Nazwisko klienta - dopuszczalne znaki: od 0 do 9, od a do z, od A do Z, znak spacji (0x20).
`company`	string(200)	`NIE`	Nazwa firmy - dopuszczalne znaki: od 0 do 9, od a do z, od A do Z, znak spacji (0x20).
`street`	string(200)	`NIE`	Ulica - dopuszczalne znaki: od 0 do 9, od a do z, od A do Z, znak spacji (0x20).
`city`	string(100)	`NIE`	Miasto - dopuszczalne znaki: od 0 do 9, od a do z, od A do Z, znak spacji (0x20).
`region`	string(100)	`NIE`	Dopuszczalne znaki: od 0 do 9, od a do z, od A do Z, znak spacji (0x20)
`postalCode`	string(30)	`NIE`	Kod pocztowy.
`countryCodeAlpha2`	string(20)	`NIE`	Kod kraju Alpha2.

5.1.4.3. shipping

Parametr	Typ	Parametr wymagany	Opis
`firstName`	string(100)	`WYMAGANE`	Imię klienta - dopuszczalne znaki: od 0 do 9, od a do z, od A do Z, znak spacji (0x20).
`lastName`	string(100)	`WYMAGANE`	Nazwisko klienta - dopuszczalne znaki: od 0 do 9, od a do z, od A do Z, znak spacji (0x20).
`company`	string(200)	`NIE`	Nazwa firmy - dopuszczalne znaki: od 0 do 9, od a do z, od A do Z, znak spacji (0x20).
`street`	string(200)	`NIE`	Ulica - dopuszczalne znaki: od 0 do 9, od a do z, od A do Z, znak spacji (0x20).
`city`	string(100)	`NIE`	Miasto - dopuszczalne znaki: od 0 do 9, od a do z, od A do Z, znak spacji (0x20).
`region`	string(100)	`NIE`	Dopuszczalne znaki: od 0 do 9, od a do z, od A do Z, znak spacji (0x20)
`postalCode`	string(30)	`NIE`	Kod pocztowy.
`countryCodeAlpha2`	string(2)	`NIE`	Kod kraju Alpha2.

5.1.4.4. card

Parametr	Typ	Parametr wymagany	Opis
`firstName`	string(100)	`WYMAGANE`	Imię właściciela karty - dopuszczalne znaki: od 0 do 9, od a do z, od A do Z, znak spacji (0x20).
`lastName`	string(100)	`WYMAGANE`	Nazwisko właściciela karty - dopuszczalne znaki: od 0 do 9, od a do z, od A do Z, znak spacji (0x20).
`number`	string(16)	`WYMAGANE`	Numer karty
`month`	string(2)	`WYMAGANE`	Ważność karty - miesiąc
`year`	string(4)	`WYMAGANE`	Ważność karty - rok
`cvv`	string(4)	`WYMAGANE`	Kod cvv karty

5.2 Metody i kanały realizacji transakcji

Transakcje można realizować następującymi metodami:

Nazwa	Wartość parametru `paymentMethod`	Wartość parametru `paymentMethodCode`
Przelewy online.	`pbl`	Tabela poniżej (punkt 5.2.1)
Płatność kart.	`card`	Tabela poniżej (punkt 5.2.2)
Płatność BLIK.	`blik`	`blik`
Płać z ING.	`ing`	`ing`

5.2.1 Przelew online Pay By Link

Przelewy Pay By Link można realizować za pomocą następujących usług banków:

Wartość parametru	Nazwa usługi
`mtransfer`	mTransfer - mBank
`bzwbk`	Przelew24
`pekao24`	Pekao24Przelew - Bank Pekao
`inteligo`	Płacę z Inteligo
`ipko`	Płać z iPKO
`getin`	Płacę z Getin Bank
`noble`	Płacę z Noble Bank
`ideabank`	Płacę z Idea Bank - IdeaBank
`creditagricole`	Credit Agricole e-przelew
`tmobile`	Płacę z T-mobile Usługi Bankowe dostarczane przez Alior Bank
`eurobank`	Płacę z Eurobankiem
`alior`	Płacę z Alior Bankiem
`pbs`	Płacę z PBS
`millennium`	Millennium - płatności internetowe
`raiffeisenpolbank`	R-Przelew
`citi`	Przelew z Citi Handlowego
`bos`	Płać z BOŚ
`bnpparibas`	Płacę z BGŻ BNP Paribas
`orange`	Płacę z Orange

Wartość parametru	Nazwa usługi
`pocztowy`	Pocztowy24
`plusbank`	Płacę z Plus Bank
`bs`	Bank Spółdzielczy
`bspb`	Bank Spółdzielczy w Brodnicy
`nest`	nestPrzelew
`envelo`	Envelo Bank

5.2.2. Płatność kartą

Płatność kartą można realizować za pomocą następujących usług:

Wartość parametru	Nazwa usługi
`ecom3ds`	Płatność kartą 3DS
`oneclick`	Płatność za pomocą usługi `oneclick`
`recurring`	Płatność za pomocą usługi `recurring`

5.3. HTTP Response

5.3.1. Odpowiedź serwera

W przypadku wykonania poprawnego zapytania rejestrującego nowe zamówienie serwer odpowie statusem HTTP `200` oraz informacją o nowo utworzonej transakcji:


```
{
   "transaction": {
      "id": "f115d23d-a943-4585-a3d7-09f6c417200d",
      "status": "new",
      "source": "api",
      "created": 1501188304,
      "modified": 1501188304,
      "notificationUrl": "https://notification_url",
      "serviceId": "63f574ed-d4ad-407e-9981-39ed7584a7b7",
      "amount": 100,
      "currency": "PLN",
      "title": "",
      "orderId": "123123123",
      "paymentMethod": "ing",
      "paymentMethodCode": "ing",
      "successReturnUrl": "https://domain.com/success",
      "failureReturnUrl": "https://domain.com/failure",
      "customer": {
         "firstName": "Jan",
         "lastName": "Kowalski",
         "cid": "123",
         "company": "",
         "phone": "",
         "email": "jan.kowalski@example.com"
      },
      "billing": {
         "firstName": "Jan",
         "lastName": "Kowalski",
         "company": "Company",
         "street": "Street",
         "city": "City",
         "region": "Region",
         "postalCode": "",
         "countryCodeAlpha2": "PL"
      },
      "shipping": {
         "firstName": "Jan",
         "lastName": "Kowalski",
         "company": "Company",
         "street": "Street",
         "city": "City",
         "region": "Regino",
         "postalCode": "",
         "countryCodeAlpha2": "PL"
```

```
        }
    },
    "action": {
        "type": "redirect",
        "url": "https://eblik.pl/blikweb/transaction/transaction_init/submit",
        "method": "POST",
        "contentType": "application/x-www-form-urlencoded",
        "contentBodyRaw":
"Type=Ipay2&MerchantID=7510&Currency=PLN&Amount=100&CustomParam=z9bfzsr7a&Descriptio
n=z9bfzsr7a%7CTest+transaction%7C%7Cg2a.com%2F&ControlData=2224807C0459BBE508011756
21EF717EE725F428575E40508A46006BC24C7F96"
    }
}
```


W odpowiedzi otrzymujemy dwa obiekty: `transaction` oraz `action`.

Obiekt `transaction` jest identyczny z wysłanym w zapytaniu rejestrującym zamówienie i zawiera kilka dodatkowych parametrów:

Parametr	Typ	Opis
`id`	string	Identyfikator transakcji w formacie UUID v4. Unikalny dla każdego zamówienia.
`status`	string	Status zamówienia.
`source`	string	Źródło zamówienia. Może posiadać wartości: `api` lub `web`.
`created`	integer	Data utworzenia zamówienia w formacie UNIX TIMESTAMP czasu UTC.
`modified`	integer	Data ostatniej zmiany statusu transakcji w formacji UNIX TIMESTAMP czasu UTC.

Drugim dodatkowym obiektem jest `action`. Obiekt ten wystąpi tylko w przypadku konieczności przekierowania płatnika na zewnętrzną stronę jak to ma miejsce w przypadku płatności `Pay-By-Link`. Obiekt ten zawiera dodatkowe pola których znaczenie jest opisane poniżej:

Parametr	Typ	Opis
`type`	string	Typ akcji.
`url`	string	W przypadku konieczności wykonania przekierowania płatnika na inną stronę (np. banku) adres URL.
`method`	string	Metoda POST lub GET.
`contentType`	string	Pozycja w nagłówku zapytania do banku określająca typ payloadu.
`contentBodyRaw`	string	Payload zapytania.

5.3.2. Statusy HTTP

Kod HTTP	Znaczenie
200	Zapytanie wykonane poprawnie. Utworzono transakcję
400	Błędne żądanie, niepoprawny payload żądania.
401	Nieautoryzowany dostęp. Żądanie zasobu, który wymaga uwierzytelnienia.
403	Brak uprawnień do wykonania żądania.
404	Nieznany zasób.
422	Payload jest poprawny ale nie zawiera wymaganaych parametrów.
500	Błąd serwera.
503	System niedostępny.

5.3.3. Status 400

Przyczyny wystąpienia kodu odpowiedzi 400 oraz treść odpowiedzi może być następująca:

- payload to niepoprawny JSON i nie mógł być przetworzony przez serwer:

```
{
   "apiErrorResponse": {
      "status": 400,
      "message": "Bad Request"
   }
}
```

5.3.4. Status 422

- payload żądania zawiera poprawny string JSON jednak nie zawiera wszystkich wymaganych parametrów:


```
{
  "apiErrorResponse": {
    "message": "Incorrect Payload",
    "code": "TRX-ERROR-120001",
    "instance": {
      "type": "sale",
      "serviceId": "63f574ed-d4ad-407e-9981-39ed7584a7b7",
      "amount": 0.03,
      "currency": "PLN",
      "title": "",
      "orderId": "123123",
      "paymentMethod": "ing",
      "paymentMethodCode": "ing",
      "customer": {
        "firstName": "",
        "lastName": "",
        "cid": "",
        "company": null,
        "phone": "",
        "email": ""
      }
    },
    "errors": [
      {
        "property": "instance.customer.firstName",
        "message": "does not meet minimum length of 1"
      },
      {
        "property": "instance.customer.lastName",
        "message": "does not meet minimum length of 1"
      },
      {
        "property": "instance.customer.cid",
        "message": "does not match pattern "^[\\w\\s-#.\\\\/]{1,100}$""
      },
      {
        "property": "instance.customer.cid",
        "message": "does not meet minimum length of 1"
      }
    ],
  }
}
```

gdzie:

- `message` - opis błędu,
- `code` - kod błędu walidacji treści żądania,
- `instance` - zawiera treść zapytania wysłana do serwera imoje,
- `errors` - zawiera listę błędów, które wystąpiły podczas walidacji treści zapytania.

6. Wykonanie zwrotu

Poprawne wykonanie zwrotu polega na wysłaniu żądania metodą POST na adres:

```
https://api.imoje.pl/v1/merchant/{merchantId}/transaction/{transactionId}/refund
```

gdzie:

- `merchantId` - identyfikator klienta,
- `transactionId` - unikalny identyfikator dla każdej transakcji której dotyczy zwrot.

W zawartości żądania należy wprowadzić:

```
{
  "type": "refund",
  "serviceId": "12341234-3efa-4ea2-b193-59401da40f18",
  "amount": 100
}
```

gdzie:

- `serviceId` - identyfikator sklepu klienta,
- `amount` - kwota do zwrotu.

W żądaniu powinny być również zawarte nagłówki:

```
Content-Type: application/json
Authorization: {method} {token}
```

Dla różnych metod zawartość nagłówka `Authorization` będzie inna.

Metoda	Zawartość nagłówka
Token autoryzacyjny	Bearer {token}

Parametr `token` - to ciąg znaków dostępny w panelu merchanta dla odpowiedniej metody.

7. Tworzenie nowego linku płatności przez API

W celu utworzenia nowego linku płatności należy za pomocą metody `POST` przesłać komunikat na endpoint API `https://api.imoje.pl/v1/merchant/{merchantId}/payment` zawierający informacje o nowym linku płatności.

gdzie:

- `merchantId` - identyfikator klienta.

7.1. HTTP Request

7.1.1. Przykładowy adres na który należy wysłać żądanie POST

```
https://api.imoje.pl/v1/merchant/6yt3gjtm9p7b8h9xsdqz/payment
```

7.1.2. Nagłówki zapytania

```
Accept: application/json
Authorization: Bearer sMkJhausOksa87Hbagt+8salsnsJjayPmznx
Content-Type: application/json
Cache-Control: no-cache
```

7.1.3. Payload zapytania

```
{
   "serviceId": "62f574ed-d4ad-4a7e-9981-89ed7284aaba",
   "amount": 100,
   "currency": "PLN",
   "title": "",
   "orderId": "123123123",
   "returnUrl": "https://domain.com/return",
   "successReturnUrl": "https://domain.com/success",
   "failureReturnUrl": "https://domain.com/failure",
   "simp": "12345678901234567890123456",
   "customer": {
      "firstName": "Jan",
      "lastName": "Kowalski",
      "email": "jan.kowalski@example.com",
      "phone": "501501501"
   }
}
```

7.1.4. Parametry payload

Parametr	Typ	Parametr wymagany	Opis
`serviceId`	string(36)	`WYMAGANE`	identyfikator sklepu jako UUID v4.
`amount`	integer	`WYMAGANE`	Kwota transakcji w najmniejszej jednostce waluty np. grosze.
`currency`	string(3)	`WYMAGANE`	Waluta transakcji w standardzie ISO 4217.
`orderID`	string(100)	`WYMAGANE`	Numer zamówienia akceptanta - dopuszczalne znaki: A-Za-z0-9_-`#&'",./\ oraz białe znaki i znaki z zakresu UNICODE 00C0 - 02C0 (m.in. polskie znaki diakrytyczne).
`title`	string(255)	`NIE`	Tytuł zamówienia.
`returnUrl`	string(300)	`NIE`	Adres powrotu z zewnętrznej strony obsługującej płatność w przypadku nie rozstrzygnięcia statusu transakcji.
`successReturnUrl`	string(300)	`NIE`	Adres powrotu z zewnętrznej strony obsługującej płatność w przypadku dokonania płatności z powodzeniem.
`failureReturnUrl`	string(300)	`NIE`	Adres powrotu z zewnętrznej strony obsługującej płatność w przypadku wystąpienia błędu płatności.
`simp`	string(26)	`NIE`	Pełny numer rachunku wirtualnego którego dotyczy wpłata. Dotyczy tylko sklepów, które obsługują płatności SIMP.
`customer`	object	`WYMAGANE`	Dane klienta.

Parametry dla `customer`:

Parametr	Typ	Parametr wymagany	Opis
`firstName`	string(100)	`WYMAGANE`	Imię klienta.
`lastName`	string(100)	`WYMAGANE`	Nazwisko klienta.
`email`	string(200)	`WYMAGANE`	Adres email.
`phone`	string(20)	`NIE`	Numer telefonu.

7.2. HTTP Response

7.2.1. Odpowiedź serwera

W przypadku wykonania poprawnego zapytania rejestrującego nowy link płatności serwer odpowie statusem HTTP `200` oraz informacją o nowo utworzonym linku płatności:


```
{
  "payment": {
    "id": "f9b9a613-5133-4d81-9d09-a3c83f505136",
    "url": "https://paywall.imoje.pl/pay/fsb9a613-5133-4d81-9d09-a3c83f505136",
    "serviceId": "63f574ed-d4ad-407e-9981-39ed7584a7b7",
    "orderId": "123123123",
    "title": "",
    "simp": "12345678901234567890123456",
    "amount": 100,
    "currency": "PLN",
    "returnUrl": "https://domain.com/return",
    "failureReturnUrl": "https://domain.com/failue",
    "successReturnUrl": "https://domain.com/success",
    "customer": {
      "firstName": "Jan",
      "lastName": "Kowalski",
      "email": "jan.kowalski@example.com",
      "phone": "501501501"
    },
    "isActive": true,
    "validTo": null,
    "created": 1539679647,
    "modified": 1539679647
  }
}
```


W odpowiedzi otrzymujemy obiekt `payment` z informacjami które zostały przekazane w requeście i zawiera kilka dodatkowych parametrów:

Parametr	Typ	Opis
`id`	string	Identyfikator transakcji w formacie UUID v4. Unikalny dla każdego zamówienia.
`url`	string	Adres URL linku płatności.
`validTo`	integer, null	Data ważności linku płatności w formacie UNIX TIMESTAMP czasu UTC. Jeśli jest null - nigdy nie wygaśnie.
`created`	integer	Data utworzenia linku płatności w formacie UNIX TIMESTAMP czasu UTC.
`modified`	integer	Data ostatniej zmiany statusu linku płatności w formacji UNIX TIMESTAMP czasu UTC.

7.2.2. Statusy HTTP

Kod HTTP	Znaczenie
200	Zapytanie wykonane poprawnie. Utworzono transakcję
400	Błędne żądanie, niepoprawny payload żądania.
401	Nieautoryzowany dostęp. Żądanie zasobu, który wymaga uwierzytelnienia.
403	Brak uprawnień do wykonania żądania.
404	Nieznany zasób.
422	Payload jest poprawny ale nie zawiera wymaganaych parametrów.
500	Błąd serwera.
503	System niedostępny.

7.2.3. Status 400

Przyczyny wystąpienia kodu odpowiedzi 400 oraz treść odpowiedzi może być następująca:

- payload to niepoprawny JSON i nie mógł być przetworzony przez serwer:


```
{
   "apiErrorResponse": {
      "code": "REQ-ERROR-100001",
      "message": "Bad request. Incorrect content-type. Expected application/json.",
      "instance": {},
      "errors": []
   }
}
```

7.2.4. Status 422

- payload żądania zawiera poprawny string JSON jednak nie zawiera wszystkich wymaganych parametrów:


```
{
  "apiErrorResponse": {
    "code": "PMT-ERROR-150001",
    "message": "Unprocessable Entity.",
    "instance": {
      "serviceId": "63f574ed-d4ad-407e-9981-39ed7584a7b7",
      "amount": 100,
      "currency": "PLN",
      "title": "",
      "orderId": "123123123",
      "customer": {
        "firstName": "",
        "lastName": "",
        "email": ""
      }
    },
    "errors": [
      {
        "property": "instance.customer.firstName",
        "message": "does not meet minimum length of 1"
      },
      {
        "property": "instance.customer.lastName",
        "message": "does not meet minimum length of 1"
      },
      {
        "property": "instance.customer.email",
        "message": "does not conform to the \"email\" format"
      }
    ],
  }
}
```


gdzie:

- `code` - kod błędu walidacji treści żądania,
- `message` - opis błędu,
- `instance` - zawiera treść zapytania wysłana do serwera imoje,
- `errors` - zawiera listę błędów, które wystąpiły podczas walidacji treści zapytania.

8. Notyfikacje

Aby poprawnie skonfigurować wysyłkę notyfikacji odnośnie transakcji do sklepu, należy wprowadzić url prowadzacy do weryfikacji w panelu administracyjnym imoje (zakładka Sklepy, później należy wybrać sklep w którym wykonywana jest integracja, kliknąć w `Szczegóły`. W otworzonej stronie przejść do sekcji `Dane do integracji` i zedytować pole `Adres notyfikacji`)

W momencie zmiany statusu transakcji serwery imoje wysyłają powiadomienia na wskazany przez akceptanta adres URL. Wymagane jest by serwer akceptanta (np. sklep) odpowiedział statusem `200 OK`, który będzie oznaczał poprawne odebranie i przetworzenie notyfikacji przez serwer akceptanta. Notyfikacje wysyłane są w następującym cyklu:

- 5 razy co 5 minut, następnie,
- 5 razy co 60 minut, następnie,
- 5 razy co 180 minut, następnie,
- 5 razy co 360 minut.

> Jeśli notyfikacja nie zostanie odebrana serwery imoje zaprzestają powtórzeń wysyłki notyfikacji.

8.1. Zawartość BODY notyfikacji

Notyfikacje wysyłane są jako obiekt `JSON` metodą `POST` i mają następującą postać:


```
{
   "transaction": {
      "id": "f115d23d-a943-4585-a3d7-09f6c417200d",
      "type": "sale",
      "status": "settled",
      "source": "api",
      "created": 1483381278,
      "modified": 1483382521,
      "notificationUrl": "https://notification_url",
      "serviceId": "63f574ed-d4ad-407e-9981-39ed7584a7b7",
      "amount": 100,
      "currency": "PLN",
      "title": "Tytul",
      "orderId": "123123123",
      "paymentMethod": "ing",
      "paymentMethodCode": "ing",
      "customer": {
         "firstName": "Jan",
         "lastName": "Kowalski",
         "cid": "123",
         "company": "",
         "phone": "",
         "email": "jan.kowalski@example.com"
      },
      "billing": {
         "firstName": "Jan",
         "lastName": "Kowalski",
         "company": "Company",
         "street": "Street",
         "city": "City",
         "region": "Region",
         "postalCode": "",
         "countryCodeAlpha2": "PL"
      },
      "shipping": {
         "firstName": "Jan",
         "lastName": "Kowalski",
         "company": "Company",
         "street": "Street",
         "city": "City",
         "region": "Region",
         "postalCode": "",
         "countryCodeAlpha2": "PL"
      }
   }
}
```

8.2. Zawartość nagłówków notyfikacji

Dodatkowo w nagłówkach HTTP umieszczane są następujące parametry:

```
Content-Type: application/json; charset=UTF-8
X-Imoje-Signature: merchantid=6yt3gjtm9p7b8h9xsdqz;serviceid=63f574ed-d4ad-407e-9981-
39ed7584a7b7;signature=5e2ac79f4f02d368cdd6eaee17d2089dec13577c1d6e3364d6fc123c85029
e82;alg=sha256
```

gdzie:

- `merchantid` - identyfikator klienta,
- `serviceid` - identyfikator sklepu,
- `signature` - podpis notyfikacji,
- `alg` - algorytm funkcji skrótu (możliwe wartości: `sha256`).

> Weryfikacja podpisu notyfikacji jest krytycznym elementem uwierzytelnienia informacji przesyłanych w pakiecie notyfikacji.

8.3. Metoda weryfikacji podpisu notyfikacji

Nagłówek zawierający podpis notyfikacji ma postać:

```
X-Imoje-Signature: merchantid=[...];serviceid=[...];signature=[...];alg=[...]
```

Aby uwierzytelnić pochodzenie oraz zweryfikować integralność wiadomości powiadomienia należy wykonać następujące czynności:

1. Z nagłówków pakietu przychodzącego na adres notyfikacji należy pobrać zawartość `X-Imoje-Signature`,
2. Następnie należy pobrać wartość parametru `signature` oraz `alg`,
3. W zależności od algorytmu funkcji skrótu określonego w parametrze `alg` należy obliczyć odpowiednią funkcją skrót:

   ```
   string incoming_signature = x_imoje_signature[signature]
   string body = notification_body
   string own_signature = hash(body + private_key, alg)
   ```

4. Obliczoną wartość `own_signature` należy porównać z wartością `incoming_signature`, która została pobrana z nagłówka,
5. Jeżeli wartości `own_signature` i `incoming_signature` są identyczne oznacza to, że wiadomość notyfikacji jest poprawna i pochodzi z zaufanego źródła.

> Zmiany statusów transakcji w należy dokonywać tylko gdy weryfikacja podpisu przebiegła poprawnie.

8.3.1. Przykład weryfikacji podpisu notyfikacji

1. W nagłówku otrzymujemy sygnaturę imoje:

```
X-Imoje-Signature: merchantid=6yt3gjtm9p7b8h9xsdqz;serviceid=63f574ed-d4ad-407e-9981-39ed7584a7b7;signature=8cddb407c5f1a46604660cf18e5670e68c6e384f9c8c4aa39b3af18c3c4bba1f;alg=sha256
```

2. W treści pakietu notyfikacji otrzymujemy `JSON`:


```
{
   "transaction": {
      "id": "f115d23d-a943-4585-a3d7-09f6c417200d",
      "type": "sale",
      "status": "settled",
      "source": "api",
      "created": 1483381278,
      "modified": 1483382521,
      "notificationUrl": "https://notification_url",
      "serviceId": "63f574ed-d4ad-407e-9981-39ed7584a7b7",
      "amount": 1.03,
      "currency": "PLN",
      "title": "Tytul",
      "orderId": "123123123",
      "paymentMethod": "ing",
      "paymentMethodCode": "ing",
      "customer": {
         "firstName": "Jan",
         "lastName": "Kowalski",
         "cid": "123",
         "company": "",
         "phone": "",
         "email": "jan.kowalski@example.com"
      },
      "billing": {
         "firstName": "Jan",
         "lastName": "Kowalski",
         "company": "Company",
         "street": "Street",
         "city": "City",
         "region": "Region",
         "postalCode": "",
         "countryCodeAlpha2": "PL"
      },
      "shipping": {
         "firstName": "Jan",
         "lastName": "Kowalski",
```

```
            "company": "Company",
            "street": "Street",
            "city": "City",
            "region": "Region",
            "postalCode": "",
            "countryCodeAlpha2": "PL"
        }
    }
}
```


1. Obliczamy sygnaturę:

   `private_key`: 25d19e0b0b4ec0ba989c94ddabc161d468d6ae1f05f0c7d4cf38a915bd68326d

   `own_signature = sha256(body + private_key)`

   `own_signature`: 8cddb407c5f1a46604660cf18e5670e68c6e384f9c8c4aa39b3af18c3c4bba1f

2. Porównujemy sygnaturę obliczoną z otrzymaną w nagłówku notyfikacji:

```
let body = "{...}";
let headerSignature = "8cddb407c5f1a46604660cf18e5670e68c6e384f9c8c4aa39b3af18c3c4bba1f";
let privateKey = "25d19e0b0b4ec0ba989c94ddabc161d468d6ae1f05f0c7d4cf38a915bd68326d";
let mySignature = crypto.createHash("sha256").update(body + privateKey).digest("hex");
if (mySignature === headerSignature) {
    // Notyfikacja zweryfikowana poprawnie. Przetwarzaj dalej.
} else {
    // Notyfikacja zweryfikowana negatywnie. Ignoruj notyfikację.
}
```

9. Pobieranie danych transakcji

Aby pobrać dane zamówienia należy wysłać żądanie `GET` na adres `https://api.imoje.pl/v1/merchant/{merchantId}/transaction/{transactionId}`.

Na przykład:

```
https://api.imoje.pl/v1/merchant/6yt3gjtm9p7b8h9xsdqz/transaction/f115d23d-a943-4585-a3d7-09f6c417200d
```

W odpowiedzi otrzymamy:


```
{
  "transaction": {
    "id": "f115d23d-a943-4585-a3d7-09f6c417200d",
    "type": "sale",
    "status": "pending",
    "source": "api",
    "created": 1483381278,
    "modified": 1483382515,
    "notificationUrl": "https://notification_url",
    "serviceId": "63f574ed-d4ad-407e-9981-39ed7584a7b7",
    "amount": 1.03,
    "currency": "PLN",
    "title": "",
    "orderId": "123123123",
    "paymentMethod": "ing",
    "paymentMethodCode": "ing",
    "customer": {
      "firstName": "Jan",
      "lastName": "Kowalski",
      "cid": "123",
      "company": "",
      "phone": "",
      "email": "jan.kowalski@example.com"
    },
    "billing": {
      "firstName": "Jan",
      "lastName": "Kowalski",
      "company": "Company",
      "street": "Street",
      "city": "City",
      "region": "Region",
      "postalCode": "",
      "countryCodeAlpha2": "PL"
    },
    "shipping": {
      "firstName": "Jan",
```

```
            "lastName": "Kowalski",
            "company": "Company",
            "street": "Street",
            "city": "City",
            "region": "Region",
            "postalCode": "",
            "countryCodeAlpha2": "PL"
        }
    }
}
```

10. Płatność oneclick i rekurencyjna

> Niżej wymienione metody płatności są dostępne tylko i wyłącznie dla akceptantów którzy posiadają certyfikat PCI-DSS. Obciążanie karty może odbywać się jedynie na podstawie wyraźnych zgód posiadacza instrumentu płatniczego. Płatnik musi mieć możliwość rezygnacji z subskrypcji. Więcej szczegółów można znaleźć u opiekuna handlowego.

Płatności `oneclick` polegają na obciążeniu zarejestrowanego już w systemie imoje profilu klienta/płatnika. Pierwsza płatność wymaga weryfikacji i podania pełnych danych instrumentu płatniczego. Każda kolejna płatność odbywa się za pomocą obciążenia na podstawie przydzielonego do instrumentu płatniczego identyfikatora (punkt 10.1.).

Płatności `rekurencyjne` polegają na cyklicznym obciążaniu instrumentu płatniczego klienta bez weryfikacji. Rejestracja instrumentu płatniczego odbywa się podobnie jak w płatności `oneclick`. Zgodnie z regulacjami organizacji płatniczych płatność musi być powtarzalna co do kwoty oraz okresu czasu.

Dokonując płatności należy pamiętać aby każdy instrument płatniczy posiadał swój unikalny identyfikator klienta/płatnika - `cid`.

10.1. Obciążenie istniejącego profilu

Wysyłając żądanie `POST` na adres `https://api.imoje.pl/v1/merchant/{merchantId}/transaction/profile` możemy obciążyć istniejący profil w systemie.

Payload zapytania powinien zawierać JSON:

```
{
   "serviceId": "63f574ed-d4ad-407e-9981-39ed7584a7b7",
   "paymentProfileId": "39ac1087-e632-41ff-acb8-8d661068a9d5",
   "amount": 100,
   "currency": "PLN",
   "orderId": "123123123"
}
```

gdzie:

- `serviceId` - identyfikator sklepu z którym związane są określone metody realizacji transakcji (UUID v4),
- `paymentProfileId` - identyfikator profilu który ma zostać obciążony,
- `amount` - kwota na którą ma zostać obciążony profil podana w groszach,
- `currency` - waluta,
- `orderId` - numer zamówienia akceptanta - dopuszczalne znaki: A-Za-z0-9_-`#&'",./\ oraz białe znaki i znaki z zakresu UNICODE 00C0 - 02C0 (m.in. polskie znaki diakrytyczne).

W odpowiedzi otrzymamy request o strukturze notyfikacji płatności wzbogacony o parametry `statusCode` oraz `paymentProfile`:


```
{
  "transaction": {
    "id": "57a105fe-af75-41eb-9195-6d0bf9183c7e",
    "status": "rejected",
    "source": "api",
    "created": 1549621088,
    "modified": 1549621088,
    "notificationUrl": "https://notificationurl.com/ipn",
    "serviceId": "6879ff96-3efa-4ea2-b193-59401da40f18",
    "amount": 1000,
    "currency": "PLN",
    "orderId": "2171ef23-828e-47e1-a3f6-80fdd4030863",
    "paymentMethod": "card",
    "paymentMethodCode": "oneclick",
    "statusCode": "PAYMENT_ERROR",
    "paymentProfile": {
      "id": "d6d59d6c-8e9c-496e-beb2-f0ca08215645",
      "merchantMid": "6yt3gjtm9p7b8h9xsdqz",
      "merchantCustomerId": "39ac1087-e632-41ff-acb8-8d661068a9d5",
      "firstName": "John",
      "lastName": "Doe",
      "maskedNumber": "****1791",
      "month": "10",
      "year": "2020",
      "organization": "MASTERCARD",
      "isActive": 1,
      "profile": "ONE_CLICK"
    }
  }
}
```


gdzie:

- `statusCode` - kod odpowiedzi od prowidera. Pełna lista odpowiedzi znajduje się w punkcie `10.1.1.`,
- obiekt `paymentProfile`:
  - `id` - identyfikator profilu,
  - `merchantMid` - identyfikator klienta,
  - `merchantCustomerId` - identyfikator płatnika, dopuszczalne znaki: A-Za-z0-9_-
  - `firstName` - imię posiadacza instrumentu płatniczego na który jest zarejestrowany profil,
  - `lastName` - nazwisko posiadacza instrumentu płatniczego na który jest zarejestrowany profil,
  - `maskedNumber` - cztery gwiazdki oraz ostatnie cztery cyfry instrumentu płatniczego,
  - `month` - data ważności karty: miesiąc,
  - `year` - data ważności karty: rok,
  - `organization` - organizacja płatnicza która wydała zarejestrowaną kartę,
  - `isActive` - aktywność profilu: 1 - aktywna, 0 - nieaktywna,
  - `profile` - rodzaj profilu.

10.1.1. Kody odpowiedzi prowidera

Kod odpowiedzi	Opis kodu odpowiedzi
`AUTHORIZED`	Płatność została zaakceptowana
`PAYMENT_ERROR`	Błąd płatności
`CARD_EXPIRED`	Data ważności instrumentu płatniczego wygasła

10.2. Zarejestrowanie nowego profilu

Rejestracja nowego profilu odbywa się za pomocą utworzenia nowej transakcji (punkt 5.) kartowej z parametrem `paymentMethodType` o wartości `oneclick` lub `recurring`. Gdy transakcja zostanie zaakceptowana, standardowa notyfikacja zostanie zwrócona wzbogacona o dodatkowe pole `statusCode` oraz sekcje `profile` o strukturze:


```
{
   "transaction": {
      "id": "57a105fe-af75-41eb-9195-6d0bf9183c7e",
      "status": "rejected",
      "source": "api",
      "created": 1549621088,
      "modified": 1549621088,
      "notificationUrl": "https://notificationurl.com/ipn",
      "serviceId": "6879ff96-3efa-4ea2-b193-59401da40f18",
      "amount": 1000,
      "currency": "PLN",
      "orderId": "2171ef23-828e-47e1-a3f6-80fdd4030863",
      "paymentMethod": "card",
      "paymentMethodCode": "oneclick",
      "statusCode": "PAYMENT_ERROR",
      "paymentProfile": {
         "id": "d6a5bd6c-8e9c-496e-beb2-f0ca08215645",
         "merchantMid": "6yt3gjtm9p7b8h9xsdqz",
         "merchantCustomerId": "39ac1087-e632-41ff-acb8-8d661068a9d5",
         "firstName": "John",
         "lastName": "Doe",
         "maskedNumber": "****1791",
         "month": "10",
         "year": "2020",
         "organization": "MASTERCARD",
         "isActive": 1,
         "profile": "ONE_CLICK"
      }
   }
}
```

gdzie:

- `statusCode` - kod odpowiedzi od prowidera. Pełna lista odpowiedzi znajduje się w punkcie `10.1.1.`,
- obiekt `paymentProfile`:
  - `id` - identyfikator profilu,
  - `merchantMid` - identyfikator klienta,
  - `merchantCustomerId` - identyfikator płatnika, dopuszczalne znaki: A-Za-z0-9_-
  - `firstName` - imię posiadacza instrumentu płatniczego na który jest zarejestrowany profil,
  - `lastName` - imię posiadacza instrumentu płatniczego na który jest zarejestrowany profil,
  - `maskedNumber` - cztery gwiazdki oraz ostatnie cztery cyfry instrumentu płatniczego,
  - `month` - data ważności karty: miesiąc,
  - `year` - data ważności karty: rok,
  - `organization` - organizacja płatnicza która wydała zarejestrowaną kartę,
  - `isActive` - aktywność profilu: 1 - aktywna, 0 - nieaktywna,
  - `profile` - rodzaj profilu.

W przypadku próby obciążenia nieaktywnego profilu odpowiedź będzie wyglądać:


```
{
   "apiErrorResponse": {
      "code": "TRX-ERROR-120301",
      "message": "Payment profile inactive.",
      "instance": {
         "serviceId": "6879ff96-3efa-4ea2-b193-59401da40f18",
         "paymentProfileId": "d6a5bd6c-8e9c-496e-beb2-f0ca08215645",
         "amount": 100,
         "currency": "PLN",
         "orderId": "2171ef23-828e-47e1-a3f6-80fdd4030863"
      },
      "errors": []
   }
}
```

10.3. Pobranie informacji o profilu na podstawie identyfikatora profilu

Wysyłając żądanie `GET` na adres `https://api.imoje.pl/v1/merchant/{merchantId}/profile/id/{paymentProfileId}` możemy pobrać szczegółowe informacje o profilu.

W odpowiedzi otrzymamy `JSON`:

```
{
   "paymentProfile": {
      "id": "d6a5bd6c-8e9c-496e-beb2-f0ca08215645",
      "firstName": "John",
      "lastName": "Doe",
      "maskedNumber": "****1791",
      "month": "10",
      "year": "2020",
      "organization": "MASTERCARD",
      "isActive": 1,
      "profile": "ONE_CLICK"
   }
}
```

gdzie:

- obiekt `paymentProfile`:
  - `id` - identyfikator profilu,
  - `firstName` - imię posiadacza instrumentu płatniczego na który jest zarejestrowany profil,
  - `lastName` - imię posiadacza instrumentu płatniczego na który jest zarejestrowany profil,
  - `maskedNumber` - cztery gwiazdki oraz ostatnie cztery cyfry instrumentu płatniczego,
  - `month` - data ważności karty: miesiąc,
  - `year` - data ważności karty: rok,
  - `organization` - organizacja płatnicza która wydała zarejestrowaną kartę,
  - `isActive` - aktywność profilu: 1 - aktywna, 0 - nieaktywna,
  - `profile` - rodzaj profilu.

10.4. Pobranie informacji o profilach na podstawie customerId

Wysyłając żądanie `GET` na adres `https://api.imoje.pl/v1/merchant/{merchantId}/profile/cid/{customerId}` możemy pobrać szczegółowe informacje o zarejestrowanych profilach dla podanego identyfikatora `customerId`.

W odpowiedzi otrzymamy `JSON`:


```
{
   "paymentProfiles": [
      {
      "id": "3b9a1aa2-a1ac-4b35-9d05-db4d637e5f91",
      "firstName": "John",
      "lastName": "Doe",
      "maskedNumber": "****1791",
      "month": "12",
      "year": "2019",
      "organization": "MASTERCARD",
      "isActive": 1,
      "profile": "ONE_CLICK"
      },
      {
      "id": "d6a59d6c-8e9c-496e-beb2-f0ca08215645",
      "firstName": "John",
      "lastName": "Doe",
      "maskedNumber": "****2741",
      "month": "11",
      "year": "2021",
      "organization": "VISA",
      "isActive": 1,
      "profile": "ONE_CLICK"
      },
      {
      "id": "39ac5087-e622-41ff-acb8-8d621068a9d5",
      "firstName": "John",
      "lastName": "Doe",
      "maskedNumber": "****1461",
      "month": "10",
      "year": "2020",
      "organization": "MASTERCARD",
      "isActive": 1,
      "profile": "ONE_CLICK"
      }
   ]
}
```

gdzie:

- pojedynczy obiekt `paymentProfile`:
  - `id` - identyfikator profilu,
  - `firstName` - imię posiadacza instrumentu płatniczego na który jest zarejestrowany profil,
  - `lastName` - imię posiadacza instrumentu płatniczego na który jest zarejestrowany profil,
  - `maskedNumber` - cztery gwiazdki oraz ostatnie cztery cyfry instrumentu płatniczego,
  - `month` - data ważności karty: miesiąc,
  - `year` - data ważności karty: rok,
  - `organization` - organizacja płatnicza która wydała zarejestrowaną kartę,
  - `isActive` - aktywność profilu: 1 - aktywna, 0 - nieaktywna,
  - `profile` - rodzaj profilu.

10.5. Dezaktywacja profilu na podstawie identyfikatora

Wysyłając żądanie `POST` na adres `https://api.imoje.pl/v1/merchant/{merchantId}/profile/deactivate` możemy dezaktywować profil.

Payload zapytania powinien zawierać JSON:

```
{
   "paymentProfileId": "3b9a1aa2-a1ac-4b35-9d05-db4d637e5f91"
}
```

gdzie:

- `paymentProfileId` - identyfikator profilu który ma zostać dezaktywowany.

W odpowiedzi otrzymamy obiekt `JSON`:

```
{
   "paymentProfile": {
      "id": "3b9a1aa2-a1ac-4b35-9d05-db4d637e5f91",
      "firstName": "John",
      "lastName": "Doe",
      "maskedNumber": "****1791",
      "month": "10",
      "year": "2020",
      "organization": "MASTERCARD",
      "isActive": 0,
      "profile": "ONE_CLICK"
   }
}
```

gdzie:

- obiekt `paymentProfile`:
  - `id` - identyfikator profilu,
  - `merchantMid` - identyfikator klienta,
  - `merchantCustomerId` - identyfikator płatnika, dopuszczalne znaki: A-Za-z0-9_-
  - `firstName` - imię posiadacza instrumentu płatniczego na który jest zarejestrowany profil,
  - `lastName` - imię posiadacza instrumentu płatniczego na który jest zarejestrowany profil,
  - `maskedNumber` - cztery gwiazdki oraz ostatnie cztery cyfry instrumentu płatniczego,
  - `month` - data ważności karty: miesiąc,
  - `year` - data ważności karty: rok,
  - `organization` - organizacja płatnicza która wydała zarejestrowaną kartę,
  - `isActive` - aktywność profilu: 1 - aktywna, 0 - nieaktywna,
  - `profile` - rodzaj profilu.

10.5.1. Alternatywne dezaktywowanie profilu metodą DELETE

Wysyłając żądanie `DELETE` na adres `https://api.imoje.pl/v1/merchant/{merchantId}/profile/id/{paymentProfileId}` możemy dezaktywować profil, gdzie:

- `paymentProfileId` - identyfikator profilu.

Odpowiedź będzie taka sama jak w punkcie `10.5.`

10.6. Mockup płatności

> Token autoryzacyjny dla środowiska testowego:
> `0aqswcjn2xg0gqc1pb2fd3pct191ame4amt5rg1c6bd95lk3iahlxlq3nyhbv8d6`

Aby przeprowadzić test płatności oneclick/recurring należy wysłać żądanie zgodnie z punktem `4.` na bazowy adres `https://sandbox.api.imoje.pl/v1/{endpoint}`. Niezależnie od przesłanych danych środowisko testowe zawsze zwróci statyczne dane, które można wykorzystać aby sprawdzić proces wykonywania płatności.

Przykładowo - w celu utworzenia transakcji należy wysłać JSON metodą POST na endpoint `https://sandbox.api.imoje.pl/v1/merchant/123456/transaction`. Odpowiedź będzie za każdym razem taka sama o strukturze opisanej w punkcie `10.2.`

11. Multiwypłaty

Opcja dostępna w przypadku włączonej funkcji multiwypłaty. Wykonujemy transakcje zgodnie z opisem punktu 5. z jednym dodatkowym parametrem:

- `multipayout` - tablica, której każdy element powinien zawierać wszystkie poniższe pola:
  - `ban` - numer konta bankowego,
  - `amount` - kwota transakcji podana w groszach,
  - `label` - nazwa odbiorcy (max 35 znaków),

Każda wypłata zawarta w obiekcie poniżej powinna zawierać kolejne indexy numerowane od 0.


```
{
  "type":"sale",
  "serviceId":"62f574ed-d4ad-4a7e-9981-89ed7284aaba",
  "amount":300,
  "currency":"PLN",
  "title":"",
  "orderId":"123123123",
  "paymentMethod":"ing",
  "paymentMethodCode":"ing",
  "successReturnUrl":"https://domain.com/success",
  "failureReturnUrl":"https://domain.com/failure",
  "customer":{
    "firstName":"Jan",
    "lastName":"Kowalski",
    "cid":"123",
    "company":"",
    "phone":"",
    "email":"jan.kowalski@example.com"
  },
  "billing":{
    "firstName":"Jan",
    "lastName":"Kowalski",
    "company":"Company",
    "street":"Street",
    "city":"City",
    "region":"Region",
    "postalCode":"",
    "countryCodeAlpha2":"PL"
  },
  "shipping":{
    "firstName":"Jan",
    "lastName":"Kowalski",
    "company":"Company",
    "street":"Street",
    "city":"City",
    "region":"Region",
    "postalCode":"",
    "countryCodeAlpha2":"PL"
```

```
  },
  "multipayout":[
    {
      "ban":"55105000026800208114085773",
      "amount":"100",
      "label":"Nazwa firmy 0",
    },
    {
      "ban":"55105000026800208114085773",
      "amount":"200",
      "label":"Nazwa firmy 1",

    }
  ]
}
```

12. Pozostałe funkcje API

12.1. Pobieranie danych o sklepach akceptanta

Wysyłając żądanie `GET` na adres `https://api.imoje.pl/v1/merchant/{merchantId}/services` otrzymamy dokument `JSON` zawierający informacje o wszystkich sklepach akceptanta.

W odpowiedzi otrzymamy:


```
[
  {
    "service": {
      "id": "63f574ed-d4ad-407e-9981-39ed7584a7b7",
      "created": 1483381278,
      "isActive": true,
      "paymentMethods": [
        {
          "paymentMethod": "ing",
          "paymentMethodCode": "ing",
          "isActive": true,
          "isOnline": true,
          "description": "24h/7",
          "currency": "PLN",
          "transactionLimits": {
            "minTransaction": {
              "type": "number",
              "value": 0
            },
            "maxTransaction": {
              "type": "number",
              "value": 99999999
            }
          }
        },
        {
          "paymentMethod": "pbl",
          "paymentMethodCode": "creditagricole",
          "isActive": true,
          "isOnline": false,
          "description": "03:00-24:00",
          "currency": "PLN",
          "transactionLimits": {
            "minTransaction": {
              "type": "number",
              "value": 0
            },
            "maxTransaction": {
              "type": "number",
              "value": 99999999
            }
```

```
            }
          },
          ...
        ]
      }
    },
    {
      "service": {
        ...
      }
    }
]
```

12.2. Pobieranie danych o sklepie akceptanta

Wysyłając żądanie `GET` na adres `https://api.imoje.pl/v1/merchant/{merchantId}/service/{serviceId}` otrzymamy dokument `JSON` zawierający informacje o określonym przez `serviceId` sklepie akceptanta.

W odpowiedzi otrzymamy:

```
{
  "service": {
    "id": "63f574ed-d4ad-407e-9981-39ed7584a7b7",
    "created": 1483381278,
    "isActive": true,
    "paymentMethods": [
      {
        "paymentMethod": "ing",
        "paymentMethodCode": "ing",
        "isActive": true,
        "isOnline": true,
        "description": "24h/7",
        "currency": "PLN",
        "transactionLimits": {
          "minTransaction": {
            "type": "number",
            "value": 0
          },
          "maxTransaction": {
            "type": "number",
            "value": 99999999
          }
        }
      },
      {
        "paymentMethod": "pbl",
        "paymentMethodCode": "creditagricole",
        "isActive": true,
        "isOnline": false,
        "description": "03:00-24:00",
```

```
            "currency": "PLN",
            "transactionLimits": {
              "minTransaction": {
                "type": "number",
                "value": 0
              },
              "maxTransaction": {
                "type": "number",
                "value": 99999999
              }
            }
          },
          ...
        ]
    }
}
```

12.3. Pobieranie zaufanych adresów IP

System umożliwia konfigurację bezpiecznych adresów IP z których możliwa jest komunikacja z API imoje. Domyślnie lista jest pusta, co oznacza, że obsługiwane będą zapytania z każdego adresu IP.

Listę zaufanych adresów IP można otrzymać wysyłając żądanie `GET` na adres `https://api.imoje.pl/v1/merchant/{merchantId}/settings/ips`.

W odpowiedzi otrzymamy dokument `JSON`:

```
{
  "trustedIps": ["127.0.0.1","10.10.10.10",...]
}
```

lub dla pustej listy:

```
{
  "trustedIps": null
}
```

12.4. Ustawianie zaufanych adresów IP

Wysyłając żądanie `PUT` na adres `https://api.imoje.pl/v1/merchant/{merchantId}/settings/ips` możemy skonfigurować listę zaufanych adresów IP.

Payload zapytania powinien zawierać listę adresów w postaci:

```
{
   "trustedIps": ["127.0.0.1","10.10.10.10"]
}
```

12.5. Pobieranie danych statystycznych

12.5.1. Liczba zarejestrowanych transakcji

Wysyłając żądanie `GET` na adres `https://api.imoje.pl/v1/merchant/{merchantId}/stats/transactions/count` otrzymamy dokument `JSON` zawierający informację o liczbie wszystkich zarejestrowanych transakcji.

W odpowiedzi otrzymamy:

```
{
   "count": 23
}
```

12.5.2. Liczba zarejestrowanych transakcji sklepu

Wysyłając żądanie `GET` na adres `https://api.imoje.pl/v1/merchant/{merchantId}/stats/transactions/count/{serviceId}` otrzymamy dokument `JSON` zawierający informację o liczbie wszystkich zarejestrowanych transakcji o określonym przez `serviceId` sklepie.

W odpowiedzi otrzymamy:

```
{
   "count": 23
}
```

12.5.3. Liczba transakcji - grupowane po statusie

Wysyłając żądanie `GET` na adres `https://api.imoje.pl/v1/merchant/{merchantId}/stats/transactions/states` otrzymamy dokument `JSON` zawierający informację o liczbie transakcji w określonym statusie.

W odpowiedzi otrzymamy:

```
{
   "new": 12,
   "authorized": 1,
   "pending": 0,
   "submitted": 22,
   "rejected": 2312,
   "settled": 21234,
   "refunded": 2
}
```

12.5.4. Liczba transakcji sklepu - grupowane po statusie

Wysyłając żądanie `GET` na adres `https://api.imoje.pl/v1/merchant/{merchantId}/stats/transactions/states/{serviceId}` otrzymamy dokument `JSON` zawierający informację o liczbie transakcji w określonym statusie o określonym przez `serviceId` sklepie.

W odpowiedzi otrzymamy:

```
{
   "new": 1,
   "authorized": 1,
   "pending": 0,
   "submitted": 3,
   "rejected": 123,
   "settled": 762,
   "refunded": 0
}
```

13. Minimalne wartości kwot transakcji, zwrotów

Dla każdej metody płatności obowiązują następujące limity:

Metoda płatności	Minimalna kwota płatności (PLN)
Przelewy online Pay-By-Link	1
Płatność za pomocą BLIK	0.10
Płatność kartą	0.01
Płatność oneclick	0.01
Płatność recurring	0.01

Poniżej tego progu dana metoda płatności nie jest dostępna.

14. Dane kontaktowe, wsparcie techniczne

Adres e-mail: kontakt.tech@imoje.pl
Telefon: +48 32 319 35 70
WWW: https://www.imoje.pl