PROMO – Manual de Integração 7.0 – Serviços

Draft

Manual de Integração – Serviços
PROMO V7.0

Índice

- Sobre o manual
  - Objetivo e escopo
  - Documentos do PROMO
- Ferramentas utilizadas
- Segurança: Autenticação de sistemas externos para serviços
- Console: Serviços REST de Consulta

    - Consulta de dados de clientes
    - Serviço de consulta de cupons
    - Serviço de consulta de cartões
    - Serviço de consulta de mapas
    - Serviço de consulta de promoções distribuídas
    - Serviço de consulta de promoções
    - Serviço Próximo Número de cartão
- Console: Serviços REST de carregamento de dados
    - Preços: Serviço de carregamento de Lista Zero
    - Serviço de importação de catálogos
        - Considerações
    - Serviço de TROCA DE PONTOS POR CATÁLOGO
    - Serviço de atribuição de Cartão de Fidelidade

Sobre o manual

Objetivo e escopo

O objetivo deste manual é capacitar os usuários que desejam integrar seu aplicativo de vendas aos serviços que o Promo disponibiliza.
É fornecida uma descrição detalhada das mensagens que devem ser enviadas e como interpretar as mensagens de resposta a solicitações.

Documentos do PROMO

O PROMO fornece os seguintes documentos:

- **Manual do usuário:**

Este documento tem como objetivo capacitar o usuário que deseja usar o console de Administração de Promoções PROMO.

- **Manual de instalação:**

Este documento descreve os procedimentos para instalação dos componentes Motor e Console, a criação e a inicialização do Banco de Dados e os requisitos necessários para o correto funcionamento dos referidos componentes.

- **Manual de Integração – Motor:**

Este documento descreve em detalhes as mensagens que devem ser enviadas ao Motor de Promoções e como interpretar as mensagens de resposta a solicitações.

- **Manual de integração – Serviços:**

Este documento descreve em detalhes os serviços que o console do Promo oferece e como enviar e receber informações a partir do console.

Observação: Antes de prosseguir com a leitura deste manual, é recomendável consultar os capítulos 2 e 3 do manual do usuário.

Ferramentas utilizadas

No desenvolvimento deste manual, basicamente três ferramentas de domínio público serão usadas para exemplos:

1. CURL (https://curl.haxx.se/) **"command line tool and library** for transferring data with URLs": É uma ferramenta de linha de comando para enviar/receber mensagens http.
2. POSTMAN (https://www.postman.com) "The Collaboration Platform for API Development": Ferramenta gráfica para criar, enviar e receber mensagens.
3. SOAPUI (https://www.soapui.org/) "Accelerating API Quality Through Testing": Outra ferramenta gráfica para definir, enviar e receber mensagens.

Segurança: Autenticação de sistemas externos para serviços

Antes de consumir um serviço fornecido pelo Promo, um processo de autenticação OAuth2 deve ser executado, consistindo na obtenção de um TOKEN que deve ser enviado nas mensagens subsequentes.

Assim, como primeiro passo para iniciar a comunicação com os serviços Promo, o sistema externo deve realizar o processo aqui descrito.

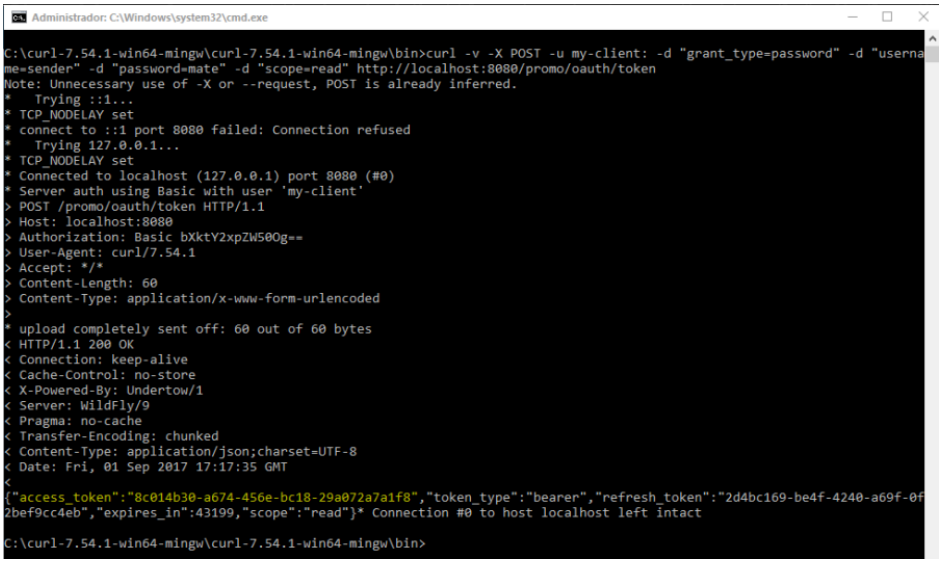
Inicialmente, deve-se solicitar o token de acesso, utilizando CURL, o comando seria:

Exemplo

```
curl –v –X POST –u my-client: -d "grant_type=password" –d "username=sender" –
d "password=mate" –d "scope=read" http://localhost:8080/promo/oauth/token
```

Onde "http://localhost:8080/promo/oauth/token" deve ser alterado para o URL onde se encontra o console do Promo, da mesma forma que o nome de usuário e a senha.

A janela abaixo exibe o exemplo de resposta:

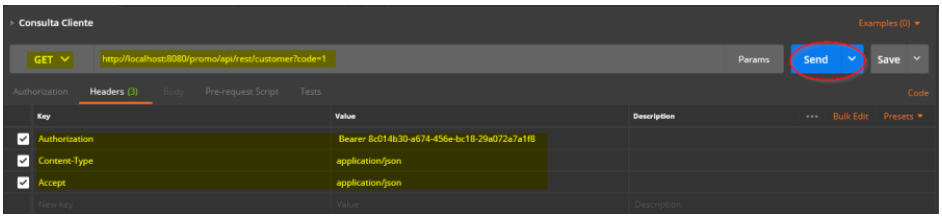


Uma vez obtido o token, este deverá ser enviado em solicitações subsequentes. Por exemplo, realizamos um GET com a ferramenta CURL:

Exemplo

```
curl -X GET -H "Authorization: Bearer 8c014b30-a674-456e-bc18-29a072a7a1f8" -
H "Content-type: application/json" -H "Accept: application/json"
"http://localhost:8080/promo/api/rest/customer?code=1"
```

Igualmente, o mesmo request, mas realizado no POSTMAN, seria:



Console: Serviços REST de Consulta

Seguem os serviços prestados para consulta das informações contidas no PROMO.

Antes de fazer uso do serviço, deve-se obter um TOKEN de identificação, conforme explicado em "Segurança: Autenticação de Sistemas Externos para Serviços"

Consulta de dados de clientes

O PROMO oferece um serviço que permite consultar cartões na BD utilizando a ID do cliente. Juntamente com os campos do cliente, uma lista de Cartões e Cupons, elementos de Fidelidade, também é fornecida.
Se não houver elementos de Fidelidade, Cartões ou Cupons vazios serão fornecidos.
O código da consulta deve ser exato e apenas um resultado será fornecido, se o cliente existir.
A resposta virá no formato JSON.

A solicitação usando CURL será:

Exemplo

```
curl -X GET -H "Authorization: Bearer
 8c014b30-a674-456e-bc18-29a072a7a1f8" -H "Content-type:
application/json" -H "Accept: application/json"
"http://localhost:8080/promo/api/rest/customer?code=1"
```

Obtendo-se a seguinte resposta:

Exemplo


```
{
"_id":
"59a992cacaec3625e8447663",
"code": "1",
"creationDate": "2017-09-
01T17:03:06Z",
"email": "gala @sts.com",
"identificationType": "RG",
"identifier": "54333222",
"isActive": true,
"lastName": "Higgins",
"name": "GALA",
"version": 0,

"cards": [
{
"_id":
"59a99439caec3625e84476f8",
"amount": 254,
"code": "2220000000001",
"created": "2017-09-
01T17:09:13Z",
"isConsumed": false,
"status": "ENABLED",
"storeCode": "BLC",
"terminalCode": "2",
"transactionId":
"BLC_201709011409677",
"type": "2",
"validFrom": "2017-09-
01T03:00:00Z",
"validTo": "2018-09-
01T03:00:00Z",
"version": 2,
"customerId": "1"
}
],

"coupons": []


}
```

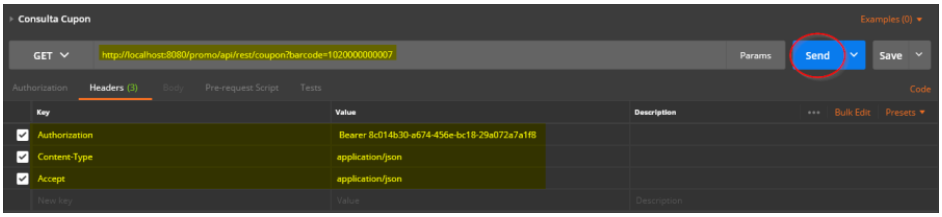

Caso o cliente não exista na base, a resposta será fornecida vazia.

Serviço de consulta de cupons

O PROMO oferece um serviço que permite consultar cupões na base de dados utilizando apenas o número do cupom e, como resposta, são obtidas as transações associadas ao referido cupom.

Os dados devolvidos são: tipo de operação, quantidade, loja.
O código da consulta deve ser exato e apenas um resultado será fornecido, se o cupom existir.

A resposta virá no formato JSON.
Para os exemplos, as consultas de cupons foram realizadas usando a ferramenta "PostMan" do Chrome.



O mesmo, mas usando o CURL:

Exemplo

```
curl
 -X GET -H "Authorization: Bearer 8c014b30-a674-456e-bc18-29a072a7a1f8"
-H "Content-type: application/json" -H "Accept: application/json"
http://localhost:8080/promo/api/rest/coupon?barcode=1020000000007
```

Obtendo-se a seguinte resposta:

Exemplo


```
{
"_id":
"59a993d6caec3625e84476e1",
"barcode": "1020000000007",
"consumed": false,
"couponFormat":
"PRE_PRINTED",
"couponStatus": "ACTIVE",
"couponType": "2",
"issuedDate": "2017-09-
01T17:07:34Z",
"maxUsageTimes": 1,
"storeCode": "BC",
"terminalCode": "2",
"transactionId":
"BC_201709011407171",
"usedTimes": 0,
"validFrom": "2017-09-
01T17:07:34Z",
"validTo": "2117-08-
08T17:07:34Z",
"version": 0,

*"couponHistory": \[*


{
"couponAction": "CREATE",
"date": "2017-09-
01T17:07:34Z",
"storeCode": "BC"
}
]
}
```


Caso o cliente não exista na base, a resposta será fornecida vazia.

Serviço de consulta de cartões

(Atualizado a partir da versão 6.5.2)

O PROMO disponibiliza um serviço que permite consultar cartões na BD utilizando apenas o número do cartão e, como resposta, são obtidos os dados do cartão em questão, bem como

transações associada. Os dados devolvidos são: data, ID da transação, tipo de operação, quantidade, loja.

Para fazer uso deste serviço, deve ser realizada uma solicitação do tipo GET com os parâmetros correspondentes ao seguinte URL: http://servidor:puerto/promo/api/rest/card

Os parâmetros de entrada para consulta são:

Propriedade	**Tipo de dado**	**Descrição**
companyId	Alfanumérico	Código da empresa
code	Alfanumérico	Código do cartão para consulta
limit	Numérico	Quantidade de registros de transações a ser fornecida (padrão: 5) com, no máximo, 100.000 por consulta.
dateFrom	Numérico	Data de início no formato AAAAMMDD
dateTo	Numérico	Data de encerramento no formato AAAAMMDD

Se for especificado um cartão, serão devolvidos os dados e transações associados, podendo-se, opcionalmente, especificar um intervalo de datas e um limite para a quantidade de transações.

Se nenhum cartão for especificado, um intervalo de datas deve ser definido. Todos as transações de cartões nesse intervalo serão fornecidas, incluindo, nesse caso, o número dos cartões em questão.

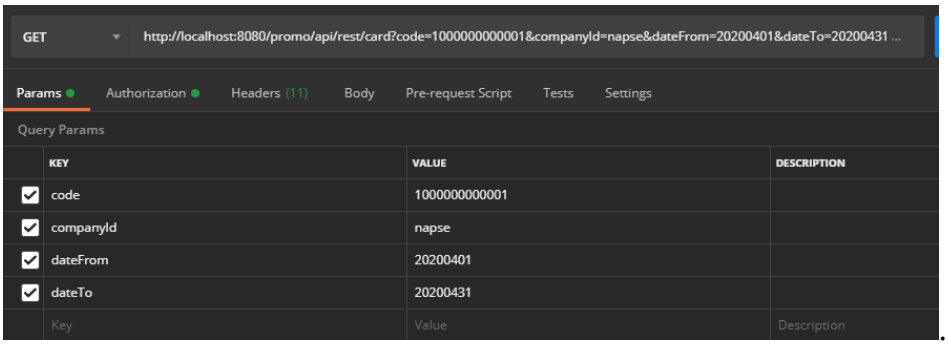
O resposta virá no formato (JSON):

Propriedade	**Tipo de dado**	**Descrição**
_id*	Alfanumérico	Uso interno
amount*	Numérico	Saldo atual do cartão.
code*	Alfanumérico	Número do cartão
Created*	Alfanumérico	Data de criação no formato ISODATE
isConsumed*	Booleano	Se foi ou não consumido
status*	Alfanumérico	Status atual do cartão

storeCode*	Alfanumérico	Código de loja de geração (se aplicável)
terminalCode*	Alfanumérico	Código do terminal de geração (se aplicável)
transactionId*	Alfanumérico	Código da transação que o gerou (se aplicável)
type*	Alfanumérico	Tipo de cartão
version*	Numérico	Uso interno
cardHistory	Coleção	Conjunto de registros que correspondem a cada transação do cartão, com: • amount: quantidade de transações • card (somente se não for consultado um determinado): número de cartão • cardAction: tipo de transação • date: data da transação (DATA ISO) • storeCode: código da loja • transactionId: código da transação.

- Esses dados são fornecidos apenas se forem consultados por um código de cartão específico.

EXEMPLO 1: solicitam-se os dados do cartão 1000000000001 para a empresa Napse, movimentação de 01/04/2020 a 31/04/2020, sem limites especificados, gerando-se, portanto, as últimas 5 transações.



Via CURL

Exemplo

```
curl
 -X GET -H "Authorization: Bearer 8c014b30-a674-456e-bc18-29a072a7a1f8
```

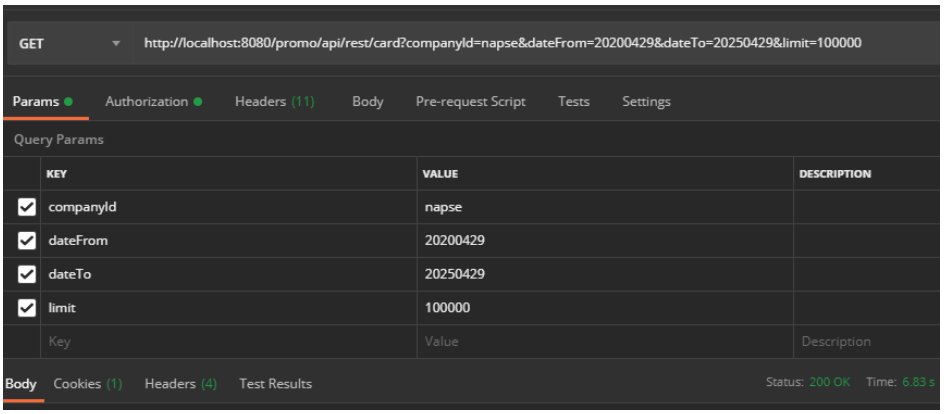
```
-H "Content-type: application/json" -H "Accept: application/json"
"http://localhost:8080/promo/api/rest/card?code=1000000000001&companyId=napse
&dateFrom=20200401&dateTo=20200431"
```

Que receberá uma resposta do tipo:

Exemplo


```
{
    "_id": "5ea975e31c8fcf4aec17953e",
    "amount": 320.0,
    "code": "1000000000001",
    "companyId": "napse",
    "created": "2020-04-29T12:41:07Z",
    "isConsumed": false,
    "status": "ENABLED",
    "storeCode": "BLC",
    "terminalCode": "001",
    "transactionId": "BLC_202004290941889",
    "type": "001",
    "version": 2,
    "cardHistory": [
        {
"amount": "20.0",
"cardAction": "AMOUNT_UPDATE",
"date": "2020-04-29T12:42:26Z",
"storeCode": null,
"transactionId": null
        },
        {
"amount": "300.0",
"cardAction": "CHARGE",
"date": "2020-04-29T12:41:07Z",
"storeCode": null,
"transactionId": null
        },
        {
"amount": "0",
"cardAction": "ACTIVATION",
"date": "2020-04-29T12:41:07Z",
"storeCode": null,
"transactionId": null
        }
    ]
}
```


EXEMPLO 2: solicitam-se todas as movimentações de 29/04/2020 a 29/04/2020 com, no máximo, 100.000 transações.



Via CURL:

Exemplo

```
curl
 -X GET -H "Authorization: Bearer 8c014b30-a674-456e-bc18-29a072a7a1f8
-H "Content-type: application/json" -H "Accept: application/json"
"http://localhost:8080/promo/api/rest/card?companyId=napse&dateFrom=20200429&
dateTo=20200429&limit=100000"
```

Que receberá uma resposta do tipo:

Exemplo


```
   {
    "cardHistory": [
        {
"amount": "10.0",
"card": "1000000000000",
"cardAction": "AMOUNT_UPDATE",
"date": "2020-04-29T12:42:18Z",
"storeCode": null,
"transactionId": null
        },
        {
"amount": "300.0",
"card": "1000000000000",
"cardAction": "CHARGE",
"date": "2020-04-29T12:02:06Z",
"storeCode": null,
"transactionId": null
        },
        {
"amount": "0",
"card": "1000000000000",
"cardAction": "ACTIVATION",
"date": "2020-04-29T12:02:06Z",
"storeCode": null,
```

```
"transactionId": null
        },
        {
"amount": "20.0",
"card": "1000000000001",
"cardAction": "AMOUNT_UPDATE",
"date": "2020-04-29T12:42:26Z",
"storeCode": null,
"transactionId": null
        },
        {
"amount": "300.0",
"card": "1000000000001",
"cardAction": "CHARGE",
"date": "2020-04-29T12:41:07Z",
"storeCode": null,
"transactionId": null
        },
        {
"amount": "0",
"card": "1000000000001",
"cardAction": "ACTIVATION",
"date": "2020-04-29T12:41:07Z",
"storeCode": null,
"transactionId": null
        },
        {
"amount": "300.0",
"card": "1000000000002",
"cardAction": "CHARGE",
"date": "2020-04-29T12:41:07Z",
"storeCode": null,
"transactionId": null
        },
        {
"amount": "0",
"card": "1000000000002",
"cardAction": "ACTIVATION",
"date": "2020-04-29T12:41:07Z",
"storeCode": null,
"transactionId": null
        },
        {
"amount": "300.0",
"card": "1000000000003",
"cardAction": "CHARGE",
"date": "2020-04-29T12:41:07Z",
"storeCode": null,
"transactionId": null
        },
        {
"amount": "0",
"card": "1000000000003",
"cardAction": "ACTIVATION",
"date": "2020-04-29T12:41:07Z",
"storeCode": null,
"transactionId": null
```

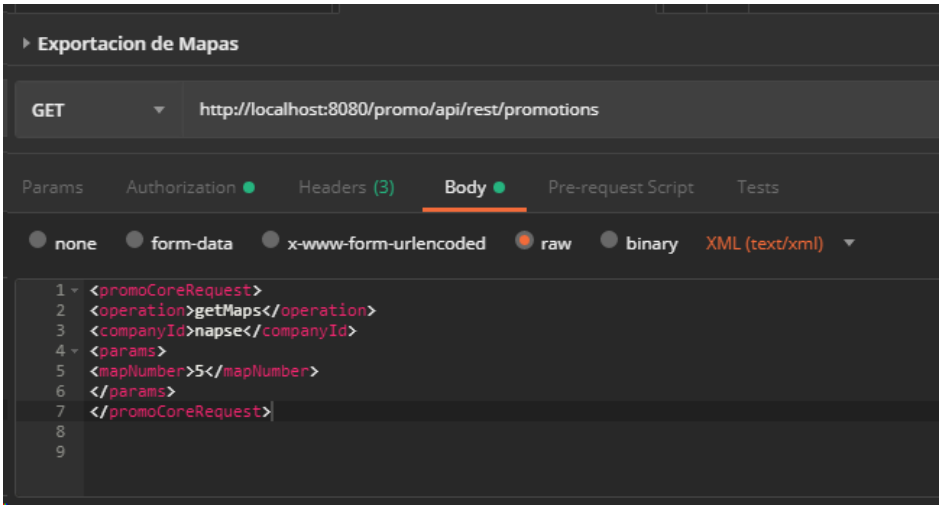
```
        },
        {
"amount": "300.0",
"card": "1000000000004",
"cardAction": "CHARGE",
"date": "2020-04-29T12:41:07Z",
"storeCode": null,
"transactionId": null
        },
        {
"amount": "0",
"card": "1000000000004",
"cardAction": "ACTIVATION",
"date": "2020-04-29T12:41:07Z",
"storeCode": null,
"transactionId": null
        },
        {
"amount": "300.0",
"card": "1000000000005",
"cardAction": "CHARGE",
"date": "2020-04-29T12:41:07Z",
"storeCode": null,
"transactionId": null
        },
        {
"amount": "0",
"card": "1000000000005",
"cardAction": "ACTIVATION",
"date": "2020-04-29T12:41:07Z",
"storeCode": null,
"transactionId": null
        }
    ]
}
```

Serviço de consulta de mapas

O PROMO disponibiliza um serviço que permite consultar mapas utilizando apenas o número do mapa e, como resposta, obtém-se o xml do mapa da mesma forma que pela opção "download".

Como no caso dos serviços REST descritos anteriormente, é necessária uma autorização via OAuth2.

A figura a seguir mostra o conteúdo do request de exemplo:



Em que:

Operação: valor getMaps para obter o mapa solicitado.

companyId: identificação ou código da empresa que realizou o request.

mapNumber: número do mapa solicitado.

A resposta é do tipo:

Response

```
<?xml version="1.0" encoding="iso-8859-1"?>
<promoCoreResponse>
    <ack>0</ack>
    <message>OK</message>
    <maps>
        <map version:="5">
<?xml version="1.0" encoding="iso-8859-1"?>
<promo-engine start-date="15/01/2019 00:00" end-date="22/01/2084 23:59"
validity-after-void="365" map-version="5" suggest="conditional">
<parameter key="Logging" value="true" />
<parameter key="LogConfigurationFile" value="logrsca.xml" />
<parameter key="LogConfigurationCategory" value="com.synthesis.promo" />
<parameter key="Descriptors" value="descriptor.properties" />
<parameter key="DAOConfigurationFeatures"
value="com.synthesis.promo.engine.dao.ConfigurationFeatures" />
<parameter key="EvaluationAlgorithmClass"
value="com.synthesis.promo.engine.evaluation.InStepsAlgorithm" />
<promotions>
<promotion nro="5c3df6e0f7a021227ca8f7d4" name="Promoção 10% de Desconto SKU
1" code="Promoção 10% de Desconto SKU 1"
className="com.synthesis.promo.engine.promotion.ModularPromotion"
```

```
reportParticipants="true" suggest="not" descriptor="Promoção 10% de Desconto
SKU 1">
<sets>
<set name="5c3df6f6f7a021227ca8f7d8" type="item" attribute="code"
comparator="Into" value="1" />
</sets>
<condition type="basic" name="Exists">
<parameter key="use-set" value="5c3df6f6f7a021227ca8f7d8" />
</condition>
<benefits>
<benefit instance="PercentageDiscount" nro="5c3df748f7a021227ca8f7dc">
<parameter key="displayMessage" value="Promoção 10% de Desconto SKU 1" />
<parameter key="printerMessage" value="Promoção 10% de Desconto SKU 1" />
<parameter key="TLOGMessage" value="Promoção 10% de Desconto SKU 1" />
<parameter key="applicationMethod" value="lineByLine" />
<parameter key="prorationMethod" value="proportional" />
<parameter key="applicationPriceType" value="benefited-price" />
<parameter key="name" value="5c3df6e0f7a021227ca8f7d4" />
<parameter key="percent" value="10" />
<parameter key="unit" value="qty" />
<applied-elements>
<use-set name="5c3df6f6f7a021227ca8f7d8" />
</applied-elements>
</benefit>
</benefits>
</promotion>
<promotion nro="5c3dfcd5f7a021227ca8f807" name="Promocao 3x2 em SKU 2"
code="Promocao 3x2 em SKU 2"
className="com.synthesis.promo.engine.promotion.ModularPromotion"
reportParticipants="true" suggest="always" descriptor="Com 3 itens de SKU 2
obtera Promocao 3x2 em SKU 2">
<sets>
<set name="5c3dfd00f7a021227ca8f80c" type="item" attribute="code"
comparator="Into" value="2" />
</sets>
<combo>
<combo-component min="3" max="3" attribute="qty" use-
set="5c3dfd00f7a021227ca8f80c" />
</combo>
<benefits>
<benefit instance="PercentageDiscount" nro="5c3dfd52f7a021227ca8f80e">
<parameter key="displayMessage" value="Promocao 3x1 em SKU 2" />
<parameter key="printerMessage" value="Promocao 3x1 em SKU 2" />
<parameter key="TLOGMessage" value="Promocao 3x1 em SKU 2" />
<parameter key="applicationMethod" value="resume" />
<parameter key="prorationMethod" value="proportional" />
<parameter key="applicationPriceType" value="benefited-price" />
<parameter key="name" value="5c3dfcd5f7a021227ca8f807" />
<parameter key="percent" value="100" />
<parameter key="unit" value="qty" />
<applied-elements>
<use-set name="5c3dfd00f7a021227ca8f80c" max="1.0" attribute="qty" />
</applied-elements>
</benefit>
</benefits>
</promotion>
```

```
<promotion nro="5c3dffd1f7a021227ca8f82e" name="Promocao Cupom de Presente em
SKU 3" code="Promocao Cupom de Presente em SKU 3"
className="com.synthesis.promo.engine.promotion.ModularPromotion"
reportParticipants="true" suggest="not">
<sets>
<set name="5c3e0041f7a021227ca8f835" type="item" attribute="code"
comparator="Into" value="3" />
</sets>
<condition type="basic" name="Exists">
<parameter key="use-set" value="5c3e0041f7a021227ca8f835" />
</condition>
<benefits>
<benefit instance="CouponBenefit" nro="5c3e0001f7a021227ca8f832">
<parameter key="displayMessage" value="Promocao Cupom de Presente em SKU 3"
/>
<parameter key="printerMessage" value="Promocao Cupom de Presente em SKU 3"
/>
<parameter key="TLOGMessage" value="Promocao Cupom de Presente em SKU 3" />
<parameter key="applicationMethod" value="resume" />
<parameter key="prorationMethod" value="proportional" />
<parameter key="applicationPriceType" value="benefited-price" />
<parameter key="name" value="5c3dffd1f7a021227ca8f82e" />
<parameter key="qty" value="1" />
<parameter key="couponid" value="tc01" />
<applied-elements>
<use-set name="5c3e0041f7a021227ca8f835" max="1.0" attribute="qty" />
</applied-elements>
</benefit>
</benefits>
</promotion>
</promotions>
<evaluation-algorithm>
<step>
<function name="ALL">
<function name="SIMPLE">
<promotion-usage name="Promoção 10% de Desconto SKU 1" />
</function>
<function name="SIMPLE">
<promotion-usage name="Promocao 3x2 em SKU 2" />
</function>
<function name="SIMPLE">
<promotion-usage name="Promocao Cupom de Presente em SKU 3" />
</function>
</function>
</step>
</evaluation-algorithm>
</promo-engine>
        </map>
    </maps>
</promoCoreResponse>
```

Em que se observa:

ack: resultado do request, em que 0 é o processamento correto. (Ver códigos de resposta ack anteriormente neste documento).

message: é a mensagem de erro resultante.

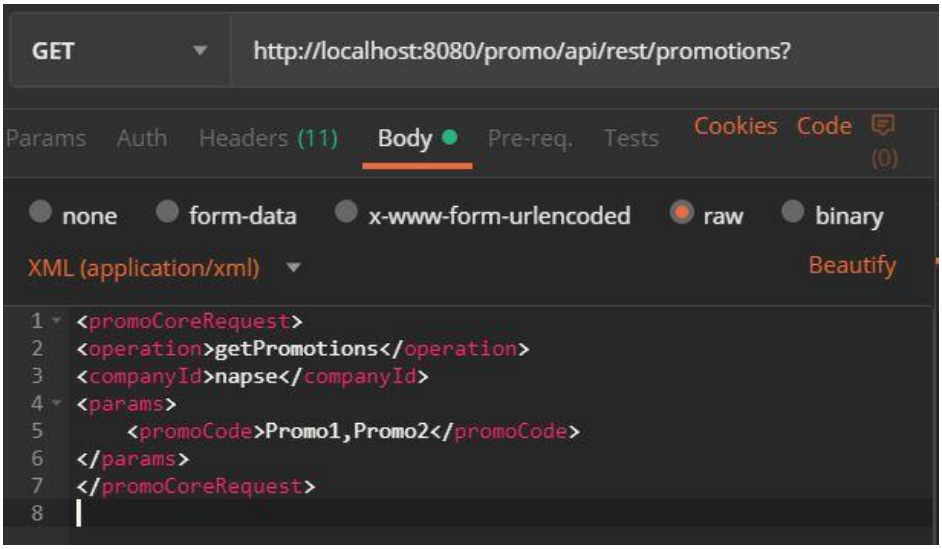
maps: tag geral que contém os elementos do mapa fornecidos na resposta. Cada elemento do mapa contém todos os elementos e tags que um mapa contém, como quando é baixado como arquivo.

Serviço de consulta de promoções distribuídas

O PROMO oferece um serviço que permite **consultar as promoções distribuídas** por meio de um filtro por empresa (obrigatório) e parâmetros de filtro opcionais (loja, região, código) e, como resposta, obtém-se um xml com as promoções.

Como no caso dos serviços REST descritos anteriormente, é necessária uma autorização via OAuth2.

A figura a seguir mostra o conteúdo do request de exemplo:



Em que:

Operation (obrigatório): valor getPromotions para obter promoções.

companyId (obrigatório): identificação ou código da empresa que realiza o request.

stores (opcional): lojas às quais as promoções pertencem.

regions (opcional): zonas às quais as promoções pertencem.

promoCode (opcional): Códigos de promoções solicitadas

A resposta é do tipo:

getPromotions

```
<?xml version="1.0" encoding="iso-8859-1"?>
<promoCoreResponse>
    <ack>0</ack>
    <message value="" />
    <promotions>
        <promotion nro="5c6c61f053c6832cc00eec0f" name="2 plays 1 xbox
gratis" code="Promo1"
className="com.synthesis.promo.engine.promotion.ConditionXComboPromotion"
reportParticipants="false" suggest="not">
<sets>
<set name="5c6c689153c6832cc00eec24" type="item" attribute="code"
comparator="Into" value="111" />
<set name="5c6c625853c6832cc00eec13" type="item" attribute="code"
comparator="Into" value="111" />
<set name="5c6c628d53c6832cc00eec15" type="item" attribute="code"
comparator="Into" value="112" />
<set name="5c6c6aae53c6832cc00eec38" type="item" attribute="code"
comparator="Into" value="112" />
</sets>
<condition type="basic" name="Exists">
<parameter key="use-set" value="5c6c689153c6832cc00eec24" />
</condition>
<combo>
<combo-component min="2" max="2" attribute="qty" use-
set="5c6c625853c6832cc00eec13" />
<combo-component min="1" max="1" attribute="qty" use-
set="5c6c628d53c6832cc00eec15" />
</combo>
<benefits>
<benefit instance="PercentageDiscount" nro="5c6c64fd53c6832cc00eec19">
<parameter key="displayMessage" value="2 plays 1 xbox gratis" />
<parameter key="printerMessage" value="2 plays 1 xbox gratis" />
<parameter key="TLOGMessage" value="2 plays 1 xbox gratis" />
<parameter key="applicationMethod" value="resume" />
<parameter key="prorationMethod" value="proportional" />
<parameter key="applicationPriceType" value="benefited-price" />
<parameter key="name" value="5c6c61f053c6832cc00eec0f" />
<parameter key="percent" value="100" />
<parameter key="unit" value="qty" />
<applied-elements>
<use-set name="5c6c6aae53c6832cc00eec38" />
```

```
</applied-elements>
</benefit>
</benefits>
        </promotion>
        <promotion nro="5c6c61f053c6832cc00eec0frest" name="2 plays 1 xbox
gratisrest" code="Promo2"
className="com.synthesis.promo.engine.promotion.ConditionXComboPromotion"
reportParticipants="false" suggest="not">
<sets>
<set name="5c6c689153c6832cc00eec24" type="item" attribute="code"
comparator="Into" value="111" />
<set name="5c6c625853c6832cc00eec13" type="item" attribute="code"
comparator="Into" value="111" />
<set name="5c6c628d53c6832cc00eec15" type="item" attribute="code"
comparator="Into" value="112" />
<set name="5c6c6aae53c6832cc00eec38" type="item" attribute="code"
comparator="Into" value="112" />
</sets>
<condition type="basic" name="Exists">
<parameter key="use-set" value="5c6c689153c6832cc00eec24" />
</condition>
<combo>
<combo-component min="2" max="2" attribute="qty" use-
set="5c6c625853c6832cc00eec13" />
<combo-component min="1" max="1" attribute="qty" use-
set="5c6c628d53c6832cc00eec15" />
</combo>
<benefits>
<benefit instance="PercentageDiscount" nro="5c6c64fd53c6832cc00eec19">
<parameter key="displayMessage" value="2 plays 1 xbox gratis" />
<parameter key="printerMessage" value="2 plays 1 xbox gratis" />
<parameter key="TLOGMessage" value="2 plays 1 xbox gratis" />
<parameter key="applicationMethod" value="resume" />
<parameter key="prorationMethod" value="proportional" />
<parameter key="applicationPriceType" value="benefited-price" />
<parameter key="name" value="5c6c61f053c6832cc00eec0frest" />
<parameter key="percent" value="100" />
<parameter key="unit" value="qty" />
<applied-elements>
<use-set name="5c6c6aae53c6832cc00eec38" />
</applied-elements>
</benefit>
</benefits>
        </promotion>
    </promotions>
</promoCoreResponse>
```

Em que se observa:

ack: resultado do request, em que 0 é o processamento correto. (Ver códigos de resposta ack anteriormente neste documento).

message: é a mensagem de erro resultante.

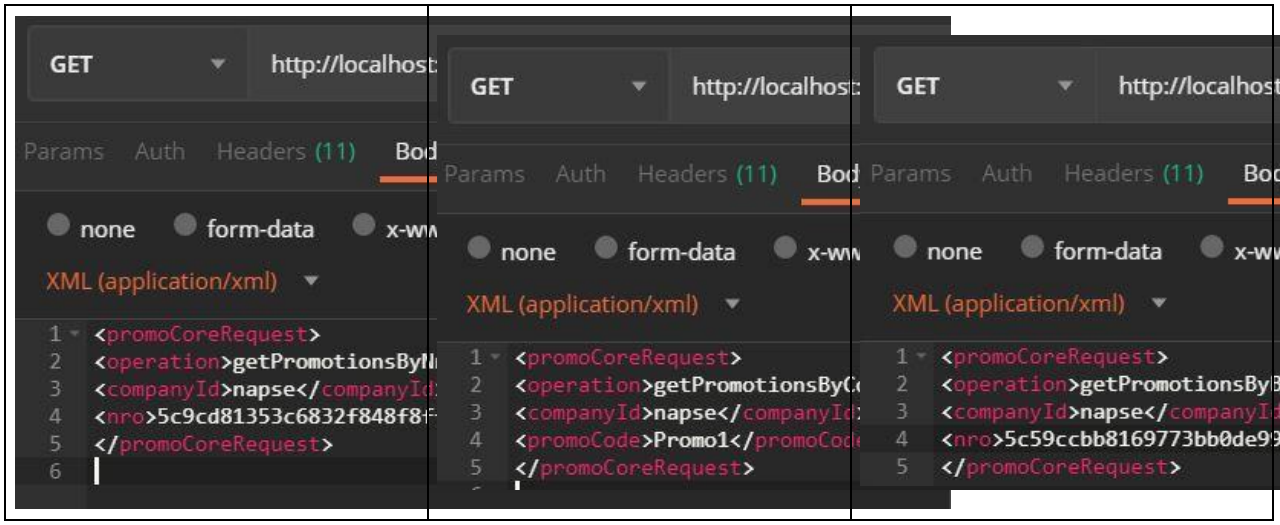
promotions: maps: tag geral que contém os elementos promotion fornecidos na resposta. Cada elemento promotion contém todos os elementos e tags de uma promoção.

Serviço de consulta de promoções

O PROMO disponibiliza três serviços que permitem consultar uma promoção por meio de um filtro por empresa (obrigatório) e por meio do número da promoção, código da promoção ou número do benefício da promoção e, como resposta, obtém-se um xml com a promoção em questão.

Como no caso dos serviços REST descritos anteriormente, é necessária uma autorização via OAuth2.

As figuras a seguir mostram o conteúdo dos requests mencionados



Em que:

Operation (obrigatório): valor getPromotionsByNro, getPromotionsByCode ou getPromotionsByBenefitNro.

companyId (obrigatório): identificação ou código da empresa que realiza o request.

nro (obrigatório): para uma pesquisa usando o número da promoção ou o número do benefício da promoção.

promoCode (obrigatório): para uma pesquisa usando o código da promoção

A resposta é do tipo:

```
<?xml version="1.0" encoding="iso-8859-1"?>
<promoCoreResponse>
    <ack>0</ack>
    <message value="" />
    <promotions>
        <promotion nro="5c9cd81353c6832f848f8ffb1" name="promoParaCancelar1"
code="Promo1"
className="com.synthesis.promo.engine.promotion.ModularPromotion"
reportParticipants="false" suggest="not">
<sets>
<set name="5c9cd83053c6832f848f8ffe" type="item" attribute="code"
comparator="Into" value="111" />
</sets>
<condition type="basic" name="Exists">
<parameter key="use-set" value="5c9cd83053c6832f848f8ffe" />
</condition>
<benefits>
<benefit instance="PercentageDiscount" nro="5c9cd86f53c6832f848f9001">
<parameter key="displayMessage" value="promoParaCancelar" />
<parameter key="printerMessage" value="promoParaCancelar" />
<parameter key="TLOGMessage" value="promoParaCancelar" />
<parameter key="applicationMethod" value="resume" />
<parameter key="prorationMethod" value="proportional" />
<parameter key="applicationPriceType" value="benefited-price" />
<parameter key="name" value="5c9cd81353c6832f848f8ffb1" />
<parameter key="percent" value="10" />
<parameter key="unit" value="qty" />
<applied-elements>
<use-set name="5c9cd83053c6832f848f8ffe" />
</applied-elements>
</benefit>
</benefits>
        </promotion>
    </promotions>
</promoCoreResponse>
```

Em que se observa:

ack: resultado do request, em que 0 é o processamento correto. (Ver códigos de resposta ack anteriormente neste documento).

message: é a mensagem de erro resultante.

promotions: maps: tag geral que contém os elementos promotion fornecidos na resposta. Cada elemento promotion contém todos os elementos e tags de uma promoção.

Serviço Próximo Número de cartão

O PROMO disponibiliza um serviço que permite consultar, para um determinado tipo de cartão, qual é o próximo número de cartão inativo e sem cliente associado.

Para os exemplos, as consultas de cartões foram realizadas usando a ferramenta "PostMan" do Chrome. Um novo Token (cUrl) deverá ser obtido e, então, poderá ser realizada a consulta do próximo número de cartões (PostMan).
O único método aceito de realizar esta solicitação é o POST.

Como todos os serviços rest do console, para realizar qualquer operação será necessário usar um token de acesso, o qual deve ser incluído no cabeçalho http Authorization, por ex.

Authorization
Bearer fb489bee-ff9f-4a4b-905c-28c6969ef180 **Não esquecer o Bearer(Espaço)**

```
curl -v -X POST -u my-client: -d "grant_type=password" -d "username=sender" -
d "password=mate"  -d "scope=read"http://localhost:8080/promo/oauth/token
```

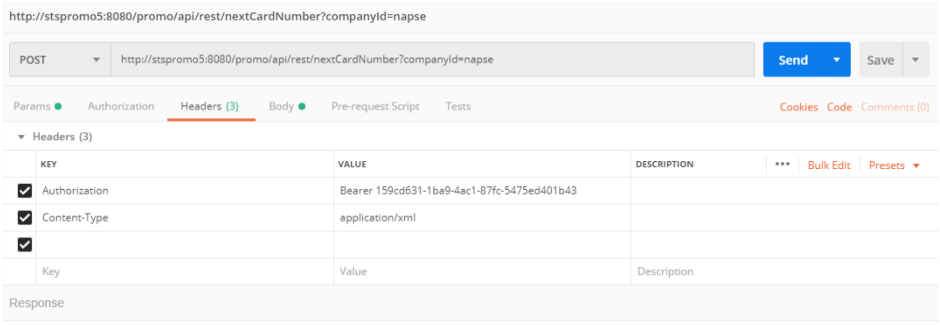
O corpo da mensagem deverá ser um xml. Possíveis solicitações e respostas envolvidas serão detalhadas a seguir.

Exemplo de consulta do próximo número de cartão

```
<promoCoreRequest>
 <operation>NextCardNumber</operation>
     <params>
        <type>nominada</type>
     </params>
</promoCoreRequest>
```

Resposta – O próximo número de cartão é informado

```
<promoCoreResponse>
  <ack>0</ack>
  <message>
    <id>2345000000002</id>
  </message>
</promoCoreResponse>
```



Console: serviços REST de carregamento de dados

Aqui estão os serviços prestados para enviar ou fazer upload de informações no PROMO.

Antes de fazer uso do serviço, deve-se obter um TOKEN de identificação, conforme explicado em "Segurança: Autenticação de Sistemas Externos para Serviços"

Preços: serviço de carregamento de Lista Zero

Foi criado um Serviço Rest que permite o carregamento de produtos de uma lista zero. A informação recebida será processada de forma assíncrona, ou seja, recebida e, depois, será processada.

A lista zero é a lista base onde se encontram todos os preços dos produtos, tem a prioridade mais baixa das listas (o motor, ao consultar os preços, avalia as listas de prioridade existentes da mais alta para a mais baixa).

A lista zero será enviada de um sistema externo para o Promo, o processo cria e atualiza os preços enviados.

No console, foi adicionada a tela de Monitoramento de Importações. Na seção de suporte, a partir desta tela, será possível visualizar o resultado da importação, as informações enviadas na importação (menu contextual Conteúdo), o detalhe da importação mostrará o total dos registros,

aqueles processados corretamente e aqueles que apresentaram erros (estes últimos, se existirem, serão exibidos na tela paginados ou podem ser baixados em arquivo de texto). O monitor também permite o reprocessamento de importações.

O métodp é um POST para o seguinte endereço: **<promo>/api/rest/priceList** com o seguinte body:

```
{
        "companyId": "napse",
        "store": "napse",
        "items": [
                 {
                "sku": "0527-1537",
                "salePrice": 69406.77,
                "costPrice": 50942.13,
                "saleCreditPrice": 119716.68,
                "magnitudePrice": 829,
                "measuredUnit": "cm3",
                "supplierItem": "PR3",
                "supplierFinancial": "PR2",
                "supplierItemAmount": 52471,
                "supplierFinancialAmount": 53604,
                "saleWholesalePrice": 31858,
                "saleWholesaleLimit": 6081,
                "limit": 794,
                "enabledDate": "2018-12-25T10:42:25Z"
          "discountable":true,
          "manualDiscount":false
        },
        {
                "sku": "66897-001",
                "salePrice": 39123.0,
                "costPrice": 10452.59,
                "saleCreditPrice": 42433.6,
                "magnitudePrice": 371,
                "measuredUnit": "kg",
                "supplierItem": "PR2",
                "supplierFinancial": "PR2",
                "supplierItemAmount": 52551,
                "supplierFinancialAmount": 21423,
                "saleWholesalePrice": 78603,
                "saleWholesaleLimit": 9691,
                "limit": 2982,
                "enabledDate": "2019-01-20T10:41:55Z"
        }
        ]
}
```

A resposta do mesmo será, em caso de estar OK (Http.status = 200)

```
{
    "status": "Informacao Recebida OK",
    "description": "A lista de preços será processada"
}
```

Em caso de erro: (Http.status=400)

```
{
    "status": "companyId is Required",
    "description": "Deve enviar o campo companyId"
}
{
    "status": "companyId is Invalid",
    "description": "Deve enviar o campo companyId válido"
}
```

Observação: A empresa enviada no rest deve ter o módulo de precificação habilitado para que o serviço seja habilitado

Propriedade	**Tipo de dado**	**Descrição**
companyId	Alfanumérico	Código da empresa
store	Alfanumérico	Código da Loja do Promo
sku	Alfanumérico	Código do produto
salePrice	Numérico	Preço de venda
costPrice	Numérico	Opcional. Preço de custo
saleCreditPrice	Numérico	Preço de crédito
magnitudePrice	Numérico	Opcional. Preço por magnitude (quilo, litro, pacote etc.)
measuredUnit	Alfanumérico	Opcional. Unidade de medida
supplierItem	Alfanumérico	Opcional. É o código do fornecedor do item
supplierItemAmount	Numérico	Opcional. Esse é o valor que o fornecedor reconhece (valor de recuperação)
supplierFinancial	Alfanumérico	Opcional. É o código do provedor financeiro do item
supplierFinancialAmount	Numérico	Opcional. Esse é o valor que o provedor financeiro reconhece (valor de recuperação)

saleWholesalePrice	Numérico	Preço opcional. Preço de atacado
saleWholesaleLimit	Numérico	Opcional. Quantidade a partir da qual o preço de atacado se aplica
limit	Numérico	Opcional (limite de unidades que poderão ser vendidas por esse preço)
enabledDate	Data	Data que indica a partir de quando o preço é válido (FORMATO ISO 8601 (UTC))
`discountable`	Booleano	Opcional. Determina se pode ser aplicado desconto ao item (se este campo for omitido, será considerado true por padrão)
`manualDiscount`	Booleano	Opcional. Determina se pode ser aplicado desconto manual ao item (se este campo for omitido, será considerado true por padrão)

O processo de importação da lista zero, inserirá/atualizará o produto na lista zero da loja. Caso o campo **enable_date** tenha uma data futura, esse preço será habilitado no momento indicado no enable_date.

O processo de atualização de preços para uma lista zero é bloqueado, enquanto a atualização está sendo processada. Em caso de erro inesperado não contemplado que o mantenha bloqueado, há um configuration data que determina o tempo em minutos de espera de um bloqueio de atualização, que é o seguinte (após esse tempo, o processo pode ser executado novamente para essa lista zero).

```
ConfigurationData.readAsInteger(companyId, "prices",
"priceList.lockForUpate",15)
```

Serviço de importação de catálogos

O PROMO disponibiliza este novo serviço como uma alternativa no carregamento/recebimento de catálogos, o que, até a presente versão, só podia ser realizado por meio de arquivos. Para entender esta seção, é recomendável consultar o gerenciamento de catálogos no manual do usuário.

O esquema de autenticação é o mesmo dos demais serviços REST apresentados aqui (ver acima).

Uma vez obtido o token de acesso, o serviço pode ser executado usando o método POST, a partir do URL: **<promo>/api/rest/catalogs**

O formato geral do request é:

```
{
 "companyId": "myCompanyId",
"catalog":"CatalogId",
"params":[
"param1":"param1value",
"paramN":"paramNvalue"
],
"items": [
{.... primeiro item .......},
{.... outro item .......},
{.... outro item .......},
{.... outro item .......},
{.... outro item .......},
]
}
```

Em que

Campo	Descrição	Tipo de dado
companyId	• Código da empresa	Alfanumérico
catalog	Identificador do catálogo	Alfanumérico

params	Parâmetros que dependem do catálogo	Lista de objetos de tipo de chave, valor. Cada catálogo pode ter seus parâmetros específicos, mas como um valor genérico temos: • **"deleteAllCollectionFirst" : "true"** Indica que antes de importar este conjunto de registros, todos os registros existentes serão deletados. OBSERVAÇÃO: (a partir da 6.5.5 e 7.TRK) Exemplo: [...] "params": [{ "deleteAllCollectionFirst": "true" }], [...] Importante Não se aplica a cartões fidelidade e atribuição de contratos.
items	Registros a serem importados	Conjunto de objetos dependentes de cada catálogo

Cada um dos itens dependerá do catálogo em seu formato, mas existe um campo comum a todos:

Campo	Descrição	Tipo de dado

operation	A operação a ser realizada no elemento. Os valores possíveis são: • I: Inserir • U: Atualizar • R: Eliminar • IU: (a partir da 6.5.5 e 7.TRK) Inserir/Atualizar: Se o registro não existir, será inserido (I); se existir, será atualizado (U). Importante Não se aplica a cartões fidelidade e atribuição de contratos.	Alfanumérico

A resposta será do tipo:

```
{
    "status": "200",
    "description": "CatalogCountry",
    "detail": {
        "result": "ok",
        "detail": "Info adicional"
    }
}
```

Em que:

Campo	Descrição	Tipo de dado	Possíveis valores
status	Código de resposta	Alfanumérico	200 (válido), 400 (erros de validação ou de sintaxe json), 500 (erro genérico)

Descrição	Identificador do catálogo	Alfanumérico	Cada um dos nomes dos catálogos importáveis por mensagem rest, ou uma string vazia se não for possível obtê-los (por exemplo, devido a um erro de sintaxe json).
detail	Informações adicionais	Objeto	Result chave, com os resultados possíveis, ou seja, OK ou erro. Detail chave, com um valor correspondente à mensagem informativa de validação, erro ou processamento válido.

Considerações

1.
    1. O número de registros por pacote (request) será limitado a 1.000 itens para não afetar o desempenho e para que as respostas possam ser processadas.
    2. Em caso de erro, os registros errados serão informados, mas os que estiverem corretos serão incorporados. Os resultados poderão ser visualizados no console, na tela do Monitor de Importação.
    3. Em todos os casos, o número total de linhas com erro e as corretas serão informadas. Os resultados poderão ser visualizados no console, na tela do Monitor de Importação.
    4. O processo utiliza semáforos para evitar a concorrência de clientes que pretendem processar em paralelo, pois implica a possibilidade de erros nas validações.

Serviço de TROCA DE PONTOS POR CATÁLOGO

(A partir da 6.5.2)

Como suporte ao novo benefício de Troca de Pontos por Catálogo (ver acima), a tabela de produtos que possuem pontos equivalentes deve ser especificada, por meio de um catálogo definido para tal uso.

Seguindo o formato especificado acima, teremos:

Campo	Valor
companyId	O código da empresa correspondente.
catalog	"CatalogRedeemBenefit". Não deve ser modificado.
`params`	Uma lista vazia. Não deve ser modificado.
items	Uma lista com cada item do catálogo de Troca de Pontos.

Dentro dos itens, o formato é:

Campo	Valor
`store`	Código da loja. Obrigatório. Alfanumérico. Se for aplicável a todas as lojas, o código da loja deve ser "-".
`code`	Código do produto. Obrigatório. Alfanumérico.
`points`	Pontos obrigatórios. Numérico com decimais.
`discountValue`	Valor do desconto. Numérico com decimais.
`discountType`	Tipo do desconto. Aceita apenas os valores "perc" (porcentagem), "fix" (desconto fixo) ou "nprice" (novo preço)
`operation`	A operação a ser realizada no elemento. Ver tabela com valores possíveis.

Portanto, teremos um exemplo de request de inserção de registros neste catálogo:

```
{
    "companyId": "napse",
    "catalog": "CatalogRedeemBenefit",
    "params": [],
    "items": [
        {
"store": "1",
"code": "10",
"points": "10.5",
"discountValue":"15" ,
"discountType": "perc",
"operation": "I"
        },
        {
"store": "1",
"code": "20",
"points": "20.5",
"discountValue":"20" ,
"discountType": "fix",
"operation": "I"
```

```
        },
                {
"store": "-",
"code": "30",
"points": "50",
"discountValue":"25" ,
"discountType": "nprice",
"operation": "I"
        }
       ]
}
```

Serviço de atribuição de Cartão de Fidelidade

Esta operação equivale à função de atribuição/atualização de cartões, mais a importação de novos clientes (ver processo atual de atribuição de contratos). Neste caso, os campos serão:

```
{
    "companyId": "napse",
    "catalog": "CatalogCardAssign",
    "params": [
        {
"cardType": "tipo1"
        },
        {
"contract": "convenio1"
        }
    ],
    "items": [
        {
"operation": "I",
"id": "1111000000030",
"customer": "codCliente1",
"amount": "90.37",
"name": "Juan",
"surname": "Perez",
"gender": "hom",
"birthDate": "19900506",
"idType": "dni",
"identifier": "33334444",
"idDate": "21001005",
"nacionality": "Argentina",
"email": "jp@gmail.com",
"customerType": "gold",
"address": "Urquiza 20",
"country": "ar",
"state": "bsas",
"city": "tig",
"postalCode": "7669",
"phone": "2222-4444"
        }
        ,{outro item.......}
        ,{outro item.......}
        ,{outro item.......}
```

```
        ,{outro item.......}
        ,{outro item.......}
        ,{outro item.......}
    ]
}
```

Campos em negrito são obrigatórios.

Campo	Descrição
params	cardType: Tipo de cartão a ser importado e validado. contrato: código do contrato ao qual esses cartões/clientes pertencem.
operation	I: Inserção / U: Atualização / R: Eliminação
id	• o O cartão deve existir. o Obrigatório
customer	• o Se o cliente estiver listado e, na BD, o cartão não estiver associado a um cliente, o sistema verificará se o cliente informado está incluído na BD e, se esse for o caso, o campo será atualizado com o valor informado no arquivo. Caso o cliente listado não exista na BD, ***será incluído com os dados correspondentes ao cliente (ver name inclusive a seguir).*** o Se o cliente estiver listado e o cartão tiver o mesmo cliente associado na BD, o campo fica como está. o Se o cliente estiver listado e, na BD, o cartão tiver outro cliente associado, ***será informado no detalhe de erros que o cartão tem outro cliente associado na BD, não sendo permitido o processamento daquele registro***. o Caso o cliente não seja listado, **haverá uma mensagem indicando que o campo cliente é obrigatório, não sendo permitido o processamento daquele registro**.

amount	• ○ Não pode conter um valor negativo ○ Obrigatório ○ Numérico com decimais. **A PARTIR DA 6:5:** • ○ Um operador é agregado à esquerda do valor para que seja adicionado ou subtraído do valor atual do cartão. Para substituir o valor, **nenhum** operador deve ser especificado. ○ Exemplo, com cartão com 100 de valor: ▪ +100 resulta em um total de 200. ▪ -60 resulta em um total de 40. ▪ 70 resulta em um total de 70. Enviar o valor como string.
name	Nome do cliente. String livre.
surname	Sobrenome. String livre.
gender	Gênero (ver valores por catálogo). Depende dos valores carregados no catálogo para este fim.
birthDate	Data de nascimento. String no formato "aaaaMMdd". Mais Informação do formato
idType	Tipo de identificação (ver valores por catálogo). Depende dos valores carregados no catálogo para este fim.
identifier	Número de identificação do cliente
idDate	Data de validade da identificação. String no formato "aaaaMMdd". Mais Informação do formato
nacionality	Nacionalidade. String livre.
email	Endereço de e-mail. String livre.
customerType	Tipo do cliente (ver valores por catálogo). Depende dos valores carregados no catálogo para este fim.
address	Endereço. String livre
country	País (ver valores por catálogo). Depende dos valores carregados no catálogo para este fim.
state	Estado/Província (ver valores por catálogo). Depende dos valores carregados no catálogo para este fim.

city	Cidade (ver valores por catálogo). Depende dos valores carregados no catálogo para este fim.
postalCode	Código postal/CEP
phone	Telefone