

API zewnętrzne – integracja sklepu internetowego

API zewnętrzne AMPER (`/external/v1/`) udostępnia do odczytu katalog produktów, kategorie, zdjęcia, stany magazynowe, promocje i rodzaje wysyłki. Dane są zawsze widziane oczami jednego klienta (kontrahenta) – z jego cenami, asortymentem i promocjami. Typowe zastosowania: zasilanie sklepu internetowego, porównywarki cenowej albo wygenerowanie pliku XML.

Przykładowy projekt: [generator pliku XML dla CENEO.PL](#).

Czego potrzebujesz

1. **Adres serwera AMPER WS** – np. `https://amper-xyz-ws.ampliapps.com` (w przykładach: `{amper_ws_url}`).
2. **Klucz API** – identyfikator w formacie UUID (w przykładach: `{amper_api_key}`).

Oba otrzymasz od działu IT firmy, z którą się integrujesz. Klucz generuje administrator w panelu AMPER i przypisuje go do konkretnego konta klienta.

Uwierzytelnienie

Klucz przekazujesz w parametrze `api_key` w adresie każdego zapytania:

```
GET https://{amper_ws_url}/external/v1/products/?api_key={amper_api_key}
```

- API obsługuje wyłącznie metodę **GET**.
- Klucz jest przypisany do konta klienta: API zwraca produkty dostępne dla tego klienta, jego ceny i jego promocje – tak, jak widzi je w sklepie B2B.
- Klucz ma datę ważności i administrator może go wyłączyć. Wygaśły lub wyłączony klucz daje odpowiedź `401`.
- Klucz daje dostęp do danych handlowych klienta. Przechowuj go po stronie serwera – nigdy w kodzie strony www ani aplikacji mobilnej.

Dostępne zasoby

Każdy zasób ma listę (`/external/v1/<zasób>/`) i odczyt pojedynczego obiektu (`/external/v1/<zasób>/{id}/`).

Adres	Zawartość
<code>/external/v1/products/</code>	produkty dostępne dla klienta, z cenami
<code>/external/v1/categories/</code>	kategorie produktów
<code>/external/v1/product-categories/</code>	przypisania produktów do kategorii
<code>/external/v1/product-files/</code>	zdjęcia produktów
<code>/external/v1/stocks/</code>	stany magazynowe produktów
<code>/external/v1/promotions/</code>	promocje klienta – opis w artykule Promocje – integracja z systemami zewnętrznymi
<code>/external/v1/shipment-types/</code>	rodzaje wysyłki

Przydatne filtry:

- `/external/v1/products/?ids=101,205,317` – tylko wskazane produkty,
- `/external/v1/products/?category_id=12` – produkty z kategorii,
- `/external/v1/product-files/?product=101` albo `?product_external_id=SKU-ABC-001` – zdjęcia jednego produktu.

Paginacja

- **Bez parametrów** odpowiedź zawiera całą listę jako tablicę JSON.
- **Z parametrami** `limit` i `offset` odpowiedź jest stroną w kopercie:

```
{
  "count": 1817,
  "next": "https://{amper_ws_url}/external/v1/products/?api_key=...&limit=500&offset=500",
  "previous": null,
  "results": [ ... ]
}
```

Przy dużym katalogu pobieraj dane stronami (np. `limit=500`), aż `next` będzie równe `null`.

Limity

- **Najwyżej 3 jednoczesne zapytania na klucz.** Kolejne dostają odpowiedź `429 Too Many Requests` z nagłówkiem `Retry-After: 1` – odczekaj wskazaną liczbę sekund i ponów

zapytanie. Strony pobieraj po kolei albo najwyżej dwie, trzy naraz.

- **Okolo 800 zapytań na minutę z jednego adresu IP.**
- Dane katalogowe (produkty, kategorie, zdjęcia) zmieniają się rzadko – wystarczy je odświeżać co kilka godzin. Częściej odświeżaj tylko stany i ceny. Nie pobieraj całego katalogu co kilka minut.

Odpowiedzi

Produkt

```
{
  "id": 1037,
  "name": "BAHAMA zapach samochodowy NASZYJNIK Wild Hibiskus /1szt",
  "friendly_name": "Zapach samochodowy Bahama Wild Hibiskus",
  "short_code": "850267",
  "description": "Zapachy inspirowane wakacjami oraz trendami konsumenckimi.",
  "short_description": "",
  "ean": "5904000850267",
  "sku": "850267",
  "cn_code": "33074900",
  "vat": 23,
  "default_unit_of_measure": "szt.",
  "weight": "0.20",
  "default_price": 25.43,
  "currency_code": "PLN",
  "product_external_id": "850267",
  "list_price": 29.90,
  "cumulative_unit_of_measure": "karton",
  "cumulative_converter": "24.00",
  "can_be_split": true,
  "cumulative_unit_ratio_splitter": "6.00",
  "unit_roundup": false,
  "capacity": null,
  "stock_availability_customer": 1,
  "brand": "Bahama & Co.",
  "manufacturer": "Bahama",
  "attributes": [
    {
      "id": 4,
```

```

    "slug": "zapach",
    "name": "Zapach",
    "values": [
      { "id": 31, "name": "Hibiskus", "slug": "hibiskus-31", "selected": true, "icon": null
    }
  ],
  "color": null,
  "show_on_tile": false
}
],
"deposit_pack_multiplier": null,
"deposit_pack_price": null,
"deposit_pack_type": null
}

```

Pole	Opis
<code>id</code>	ID produktu w AMPER
<code>name</code> , <code>friendly_name</code>	nazwa pełna i nazwa handlowa (krótsza, „przyjazna”)
<code>short_code</code> , <code>sku</code> , <code>ean</code>	kod produktu, SKU, kod EAN
<code>description</code> , <code>short_description</code>	opis i krótki opis
<code>cn_code</code>	kod CN (Nomenklatura scalona)
<code>vat</code>	stawka VAT w procentach
<code>default_unit_of_measure</code>	podstawowa jednostka miary
<code>weight</code>	waga
<code>default_price</code>	cena netto dla klienta przypisanego do klucza - uwzględnia jego cennik i rabaty
<code>currency_code</code>	waluta cen klienta
<code>list_price</code>	cena katalogowa netto
<code>product_external_id</code>	ID produktu w systemie ERP
<code>cumulative_unit_of_measure</code> , <code>cumulative_converter</code>	jednostka zbiorcza (np. karton) i liczba jednostek podstawowych w niej
<code>can_be_split</code> , <code>cumulative_unit_ratio_splitter</code>	czy jednostkę zbiorczą można dzielić i na ile części
<code>unit_roundup</code>	czy zamawiana ilość jest zaokrąglana w górę do kroku jednostki
<code>capacity</code>	pojemność

Pole	Opis
<code>stock_availability_customer</code>	dostępność dla klienta: rzeczywista ilość (stan minus rezerwacje), gdy klient ma włączoną opcję <i>Zwracaj rzeczywistą wartość stanów w API i ofertach</i> ; w przeciwnym razie <code>1</code> (jest na stanie) lub <code>0</code>
<code>brand</code> , <code>manufacturer</code>	marka i producent
<code>attributes</code>	atrybuty produktu oznaczone w panelu jako widoczne w B2B
<code>deposit_pack_multiplier</code> , <code>deposit_pack_price</code> , <code>deposit_pack_type</code>	opakowanie zwrotne (kaucja): mnożnik, cena, rodzaj

Kategoria

```
{ "id": 12, "name": "Zapachy samochodowe", "description": "", "category_external_id": "KAT-012" }
```

`category_external_id` to ID kategorii w systemie ERP.

Przypisanie produktu do kategorii

```
{ "product_id": 1037, "category_id": 12 }
```

Zdjęcie produktu

```
{ "product_id": 1037, "image": "https://.../products/850267.jpg", "order": 1, "product_external_id": "850267" }
```

`order` to kolejność zdjęć – zdjęcie z najniższą wartością jest główne.

Stan magazynowy

Jeden wiersz na produkt, zsumowany ze wszystkich magazynów:

```
{ "product_id": 1037, "availability": "medium", "quantity": 150 }
```

Pole	Opis
<code>availability</code>	stan słownie: <code>none</code> (brak), <code>low</code> (poniżej 100), <code>medium</code> (100–999), <code>high</code> (1000 i więcej)

Pole	Opis
quantity	rzeczywista ilość, gdy klient ma włączoną opcję <i>Zwracaj rzeczywistą wartość stanów w API i ofertach</i> ; w przeciwnym razie 1 (jest na stanie) lub 0

Rodzaj wysy?ki

```
{ "id": 3, "name": "Kurier" }
```

id przekazujesz jako shipment_type_id przy [tworzeniu zamówienia](#).

Kody odpowiedzi

Kod	Znaczenie
200	poprawna odpowiedź
401	brak klucza albo klucz wygał lub został wyłączony
404	nie ma obiektu o podanym id (albo nie jest dostępny dla klienta)
429	przekroczony limit jednoczesnych zapytań lub zapytań na minutę – ponów po czasie z nagłówka Retry-After

Przyk?ady

W przykładach zastąp {amper_ws_url} i {amper_api_key} swoimi danymi.

curl

```
curl "https://{amper_ws_url}/external/v1/products/?api_key={amper_api_key}&limit=500&offset=0"
```

Python – pobranie całego katalogu stronami, z obsługą limitu:

```
import time

import requests

BASE_URL = "https://{amper_ws_url}/external/v1"
params = {"api_key": "{amper_api_key}", "limit": 500, "offset": 0}
```

```

products = []
while True:
    response = requests.get(f"{BASE_URL}/products/", params=params, timeout=120)
    if response.status_code == 429:
        time.sleep(int(response.headers.get("Retry-After", 1)))
        continue
    response.raise_for_status()
    page = response.json()
    products.extend(page["results"])
    if not page["next"]:
        break
    params["offset"] += params["limit"]

print(len(products))

```

C# (.NET, `HttpClient`)

```

using var client = new HttpClient { Timeout = TimeSpan.FromMinutes(2) };
var url =
    "https://{amper_ws_url}/external/v1/products/?api_key={amper_api_key}&limit=500&offset=0";
var json = await client.GetStringAsync(url);
Console.WriteLine(json);

```

Node.js (18 i nowsze, wbudowany `fetch`)

```

const url =
    'https://{amper_ws_url}/external/v1/products/?api_key={amper_api_key}&limit=500&offset=0';
const response = await fetch(url);
if (!response.ok) throw new Error(`HTTP ${response.status}`);
const page = await response.json();
console.log(page.count, page.results.length);

```

PHP (cURL)

```

$curl =
    curl_init('https://{amper_ws_url}/external/v1/products/?api_key={amper_api_key}&limit=500&offset=0');
    curl_setopt_array($curl, [
        CURLOPT_RETURNTRANSFER => true,
        CURLOPT_TIMEOUT => 120,
    ]);

```

```
$response = curl_exec($curl);  
curl_close($curl);  
$page = json_decode($response, true);  
echo $page['count'];
```

Revision #4

Created 20 September 2024 05:30:25 by Tomek Dziemidowicz

Updated 16 October 2026 14:52:10 by Tomek Dziemidowicz