API документація v1.0

Базовий хост: `https://[ВАШ САБДОМЕН]-api.marchroute.com` · Storefront API: `/api/v1` · Export API: `/api/v1/export` · Актуально на 16.09.2026

[ВАШ САБДОМЕН] — субдомен вашої компанії в Marchroute. Точну адресу з уже підставленим субдоменом видно у вашій CRM: Налаштування → Модулі → API → Документація.

Публічний API Marchroute (CRM) складається з двох груп маршрутів:

- **Storefront API** — `https://[ВАШ САБДОМЕН]-api.marchroute.com/api/v1`: створення замовлень, оплати, синхронізація товарів, кабінет клієнта.
- **Export API** — `https://[ВАШ САБДОМЕН]-api.marchroute.com/api/v1/export`: доступ тільки на читання для вивантаження даних.

Ваш базовий хост: `https://[ВАШ САБДОМЕН]-api.marchroute.com`

Авторизація

Запити авторизуються **API-токеном** у заголовку. Токен створюється у вкладці «API ключі» і показується лише один раз.

Заголовок

```
Authorization: Bearer YOUR_API_TOKEN
```

Кожен токен має набір **scopes** (доступів). Ендпоінт перевіряє, що токен містить потрібний scope — інакше повертається `403 Insufficient scope`.

Scope	Опис
`products:sync`	Синхронізація товарів і категорій із зовнішньої системи
`orders:create`	Створення замовлень і операції з клієнтом
`payments:manage`	Обробка оплат із магазину
`person:auth`	Авторизація клієнтів магазину та видача їм токенів авторизації
`data:read`	Доступ тільки на читання (Export API)

Маршрути кабінету контрагента (`/person*`) використовують токен контрагента, а не API-токен інтеграції.

Термін дії та помилки авторизації

Ключ може бути безстроковим або мати дату завершення (за замовчуванням — 90 днів). Після цієї дати всі запити з ним отримують `401`. Причину відмови завжди видно з поля `error` — його і слід перевіряти в інтеграції, а не текст `message`. Той самий код дублюється в заголовку `WWW-Authenticate`.

HTTP	error	Коли виникає
401	`missing_token`	Заголовок Authorization: Bearer відсутній або порожній
401	`invalid_token`	Токен не знайдено: неправильний або відкликаний (видалений) ключ
401	`token_expired`	Термін дії ключа сплив. У відповіді додається expired_at — точний час завершення
403	`insufficient_scope`	Ключ дійсний, але не має потрібного доступу. У відповіді: required_scopes, token_scopes, missing_scopes

401 — токен прострочений

```
{
    "message": "Token has expired.",
    "error": "token_expired",
    "error_message": "Термін дії токена сплив, створіть новий ключ у налаштуваннях API.",
    "expired_at": "2026-07-01T10:15:00+03:00"
}
```

403 — бракує доступу

```
{
    "message": "Insufficient scope.",
    "error": "insufficient_scope",
```

```
    "error_message": "Токену бракує доступів для цього маршруту.",
    "required_scopes": [
        "orders:create"
    ],
    "token_scopes": [
        "products:sync"
    ],
    "missing_scopes": [
        "orders:create"
    ]
}
```


Прострочений ключ не можна продовжити — створіть новий у вкладці «API ключі» та замініть його в інтеграції

Товари та категорії

Спершу — хто є хто. У CRM три сутності навколо товару, читайте зверху вниз: • Виробник — «чий товар». Кожен товар має vendor_id. • Постачальник — «хто відвантажує замовлення». Пов’язаний з виробниками (багато-до-багатьох, у зв’язки є position — пріоритет). Саме по постачальниках CRM розподіляє замовлення. • Склад — фізичний склад CRM. Склад має власного службового постачальника: якщо замовлення розподілилось на нього — відвантаження йде зі складу. Такого постачальника видно за полем warehouse_id.

Розподіл замовлення (спрощено): товари кошика → їх виробники → постачальник, пов’язаний з цими виробниками. Повна механіка з прикладами — у POST /order, блок «Розподіл по постачальниках».

Модель даних

```
Товар (vendor_id)
  └ Виробник — чий товар ..................... GET /vendors
       └ Постачальник — хто відвантажує ...... GET /suppliers
            ├ без складу = віртуальний склад (відправляє контрагент)
            └ складський (має warehouse_id)
                 └ Склад CRM ................. GET /warehouses
```

GET `https://[ВАШ САБДОМЕН]-api.marchroute.com/api/v1/warehouses`

Довідник складів scope: products:sync

Список активних складів CRM. Для кожного складу: його службовий постачальник (supplier_id), виробники цього постачальника (vendors) і уточнюючі параметри (clarifications). Навіщо clarifications: коли виробник пов’язаний з КІЛЬКОМА складськими постачальниками, серед них треба обрати один — перемагає склад, чиї clarifications збігаються із замовленням (його організацією, товаром або категорією товару). Немає збігу — перший складський: фізичний склад завжди пріоритетніший за віртуальний, віртуальний обирається лише коли складських постачальників у виробника немає.

Поле	Тип	Обов.	Де	Опис
`id`	integer	—	query	Один конкретний склад за ID

Приклад відповіді


```
[
    {
        "id": 3,
        "name": "Основний склад",
        "supplier_id": 8,
        "vendors": [
            {
                "id": 2,
                "name": "ТОВ «Мебельний цех»"
            }
        ],
        "clarifications": [
            {
                "type": "organization",
                "value": 1
            },
            {
```

```
                "type": "category",
                "value": 12
            }
        ]
    }
]
```

- supplier_id — службовий постачальник складу (null, якщо складу ще не призначали виробників/відправку). Це той самий id, що й у GET /vendors → suppliers[].id.
- vendors[] — виробники, чиї замовлення можуть відвантажуватись з цього складу.
- clarifications[] — уточнюючі параметри: type = organization | product | category, value = id відповідної сутності. Порожній масив — склад без уточнень (обирається за position).

GET `https://[ВАШ САБДОМЕН]-api.marchroute.com/api/v1/vendors`

Довідник виробників scope: products:sync

Список виробників (id + назва) із масивом постачальників, з якими пов’язаний виробник. id виробника передається у товар як vendor_id під час синхронізації (POST /products). Постачальник із warehouse_id — «складський»: замовлення, розподілене на нього, відвантажується зі складу.

Поле	Тип	Обов.	Де	Опис
`name`	string	—	query	Частковий збіг за назвою
`id`	integer	—	query	Один конкретний виробник за ID

Приклад відповіді

```
[
    {
        "id": 2,
        "name": "ТОВ «Мебельний цех»",
        "suppliers": [
            {
                "id": 5,
                "name": "ТОВ «Постачальник»",
                "warehouse_id": null,
                "is_clarification": false
            },
            {
                "id": 8,
                "name": "Склад: Основний",
                "warehouse_id": 3,
                "is_clarification": true
            }
        ]
    }
]
```

- suppliers[] — постачальники, пов’язані з цим виробником (порожній масив, якщо зв’язків немає).
- warehouse_id — id складу, якщо постачальник складський. null — постачальник без складу, тобто ВІРТУАЛЬНИЙ СКЛАД: замовлення відвантажує сам постачальник зі свого боку, фізичний склад CRM у русі товару не бере участі.
- is_clarification — склад постачальника має уточнюючі параметри розподілу (див. GET /warehouses): при кількох рівнозначних постачальниках виробника замовлення піде на склад, що збігся за параметрами.

GET `https://[ВАШ САБДОМЕН]-api.marchroute.com/api/v1/categories`

Довідник категорій scope: products:sync

Плоский список категорій (id, назва, inner_id) із зазначенням батьківської категорії, з пагінацією. Без вкладеності — ієрархія відновлюється за parent_id / parent.inner_id на боці клієнта.

Поле	Тип	Обов.	Де	Опис
`name`	string	—	query	Частковий збіг за назвою
`inner_id`	string	—	query	Частковий збіг за inner_id

`page`	integer	—	query	Номер сторінки (за замовч. 1)

Приклад відповіді


```
{
    "current_page": 1,
    "per_page": 30,
    "total": 2,
    "last_page": 1,
    "from": 1,
    "to": 2,
    "data": [
        {
            "id": 10,
            "name": "Меблі",
            "inner_id": "1001",
            "parent_id": null,
            "parent": null
        },
        {
            "id": 12,
            "name": "Дивани",
            "inner_id": "1002",
            "parent_id": 10,
            "parent": {
                "id": 10,
                "name": "Меблі",
                "inner_id": "1001"
            }
        }
    ]
}
```


- Пагінація: до 30 категорій на сторінку, параметр page.
- parent_id — id батьківської категорії у CRM (null для кореневих).
- parent — коротка інформація про батька ({ id, name, inner_id }) або null.

GET `https://[ВАШ САБДОМЕН]-api.marchroute.com/api/v1/product`

Отримати інформацію про товар scope: products:sync

Повертає товар з усіма зв’язками: категорії, виробник, додатки, характеристики, варіації та набір знижок. Потрібно передати або id (внутрішній id CRM), або inner_id (зовнішній). id має пріоритет, якщо передані обидва. Опис полів товару — див. нижче в POST /products.

Поле	Тип	Обов.	Де	Опис
`id`	integer	—	query	Внутрішній id товару в CRM
`inner_id`	string	—	query	Зовнішній inner_id товару

Приклад відповіді


```
{
    "id": 1,
    "name": "Стіл",
    "name_print": "Червоний",
    "sku": "ST-1",
    "price": 1500,
    "custom_fields": {
        "vashe_kustomne_pole": "https://..."
    },
    "categories": [],
    "vendor": [],
    "additionals": [],
    "features": [],
    "variations": [],
    "variations_parameters": [
        {
```

```
            "original_sku": "ST-1RED",
            "sku": "ST-1-RED",
            "price": 1550,
            "custom_fields": {
                "url_print": "https://..."
            }
        }
    ],
    "collection": {
        "id": 77,
        "name": "Диван Люкс",
        "parent_id": 12,
        "colors": [
            {
                "id": 1,
                "name": "Диван Люкс Червоний",
                "name_print": "Червоний",
                "sku": "ST-1"
            }
        ]
    },
    "discountSet": []
}
```


- Потрібен щонайменше один із параметрів (id або inner_id), інакше — 422.
- 404, якщо товар не знайдено.
- custom_fields — об’єкт кастомних полів {code: value}; довідник полів ведеться у картці товару CRM (Розширені налаштування). Варіанти (variations_parameters[]) мають власні custom_fields з тим самим довідником.
- collection — присутнє лише якщо товар входить у колекцію кольорів (категорія type="collection"): id/назва колекції, parent_id (категорія-батько) і colors[] — усі товари-кольори колекції.

POST `https://[ВАШ САБДОМЕН]-api.marchroute.com/api/v1/products`

Синхронізувати товари (upsert за inner_id) scope: products:sync

Приймає один товар або масив товарів. Ключ upsert — inner_id (string; число приймається і тихо приводиться до рядка). Назва декодується з HTML-entity. vendor_id прив’язує товар до виробника напряму (id з GET /vendors). category_id прив’язує до категорії за внутрішнім id CRM; category_inner_id — за зовнішнім inner_id категорії (пріоритет у category_id, якщо передані обидва).

Поле	Тип	Обов.	Де	Опис
`inner_id`	string	так	body	Зовнішній унікальний ID, ключ upsert (число приймається і приводиться до рядка)
`name`	string	так	body	Назва товару
`price`	numeric	так	body	Ціна
`cost`	numeric	—	body	Собівартість
`description`	string	—	body	Опис
`sku`	string	—	body	Артикул
`vendor_sku`	string	—	body	Артикул постачальника
`uktzed`	string	—	body	Код УКТЗЕД
`vendor_id`	integer	—	body	ID виробника (з GET /vendors). Прив’язує товар до виробника напряму
`category_id`	integer	—	body	Внутрішній id категорії в CRM (з GET /categories)
`category_inner_id`	string	—	body	Зовнішній inner_id категорії (як раніше працював category_id)
`availability_days`	integer	—	body	Середня кількість днів до появи товару в доступі
`max_installment_months`	integer	—	body	Максимум місяців розстрочки для товару; у замовленні береться мінімум серед товарів
`url`	string	—	body	URL товару
`url_image`	string	—	body	URL зображення для синхронізації

`document_name`	string	—	body	Назва для документів
`custom_fields`	object	—	body	Кастомні поля товару {code: value}. Точковий merge: передані ключі перекривають, решта зберігається
`name_print`	string	—	body	Назва кольору/принту — реальне поле товару, підпис кольору в колекції
`prepayment`	numeric	—	body	Передоплата: сума або відсоток (див. prepayment_is_percent)
`prepayment_is_percent`	boolean	—	body	Передоплата у відсотках, а не сумою
`is_paid_delivery`	boolean	—	body	Чи доставка товару платна
`dimensions`	array	—	body	Масив габаритів: weight,length,width,height (>0), hand(bool)
`variations`	array	—	body	Варіації (sku/name/options[])
`variations_params`	array	—	body	Параметри SKU-комбінацій (див. details)
`display_logic`	array	—	body	Логіка зв’язків між SKU: зміна ціни / показ-приховування (див. details)
`additionals`	array	—	body	Додаткові товари
`discount_set`	array	—	body	Опис набору/знижки

Тіло запиту

```
{
    "inner_id": "1001",
    "name": "Стіл",
    "price": 1500,
    "cost": 900,
    "sku": "ST-1",
    "category_id": 5,
    "dimensions": [
        {
            "weight": 12,
            "length": 100,
            "width": 60,
            "height": 75,
            "hand": false
        }
    ]
}
```

Приклад відповіді

```
{
    "processed_count": 2,
    "processed": [
        1001,
        1002
    ],
    "is_failed": true,
    "failed": [
        {
            "data": 1003,
            "error": {
                "price": [
                    "The price field is required."
                ]
            }
        }
    ]
}
```

dimensions — габарити (валідуються)

Масив габаритів. Це єдиний масив місць: кожен елемент означає конкретне місце weight (кг), length (см), width (см), height (см) (усі numeric > 0) та hand (boolean. Показує, чи товар потребує ручної обробки на пошті). Зазвичай передають одне місце.

```
[
    {
        "weight": 12.5,
        "length": 100,
        "width": 60,
        "height": 75,
        "hand": false
    }
]
```

variations — варіації та опції

Масив варіацій. Елемент без sku або name пропускається. options[] — значення варіації; кожне без sku/name теж пропускається. Примітка: опція з name "Немає" вважається порожнім значенням — у назві товару в документах вона не виводиться. value_type — як трактувати price/cost опції: "default" (за замовч.) — надбавка до базової, "procent" — відсоток від базової, "absolute" — фіксоване значення замість базової. sku_mode — як артикул опції формує артикул варіанта: "append" (за замовч.) — приклеюється до артикула варіації (ST-1 + RED), "replace" — артикул варіанта буде саме таким, як задано в опції (без базового).

```
[
    {
        "sku": "COLOR",
        "name": "Колір",
        "options": [
            {
                "sku": "RED",
                "name": "Червоний",
                "price": 0,
                "cost": 0,
                "value_type": "default",
                "sku_mode": "append",
                "display_type": "",
                "url_image": "https://..."
            }
        ]
    }
]
```

variations_params — параметри SKU-комбінацій

Уточнення для конкретної комбінації варіацій. Ідентифікується через original_sku АБО variation_by_text (інакше елемент пропускається). Поля price/cost/active застосовуються лише якщо передані; dimensions нормалізуються. original_sku — згенерований артикул комбінації (базовий артикул товару + артикули опцій), він же ключ зіставлення. sku — переозначення артикула варіанта: підсумковий артикул саме такий, як задано (повна заміна original_sku). Порожній sku = переозначення немає, діє original_sku. Правило «додавати до базового чи задавати повністю» налаштовується на рівні опції варіації (sku_mode в options[]). Увага: при зміні базового артикула товару комбінації перегенеруються з новими original_sku, тож переозначення, надіслані зі старим original_sku, не зіставляться і зникнуть — надсилайте variations_params уже з новими original_sku. url_image — картинка конкретного варіанта (як в опцій): порожнє значення знімає її. Пріоритет показу: власна картинка варіанта → картинка опції комбінації → картинка товару. custom_fields — кастомні змінні варіанта {code: value}, довідник полів спільний з товаром. Якщо поля з таким code ще немає в довіднику — значення однаково збережеться, але колонкою в редакторі воно з’явиться лише після створення поля (розділ «Кастомні поля» товару).

```
[
    {
        "original_sku": "ST-1RED",
        "sku": "ST-1-RED",
        "price": 1550,
        "cost": 920,
        "active": true,
        "document_name": "Стіл (червоний)",
        "vendor_sku": "V-RED",
        "uktzed": "9403",
        "url_image": "https://...",
        "custom_fields": {
            "url_print": "https://..."
        },
        "dimensions": [
```

```
            {
                "weight": 12,
                "length": 100,
                "width": 60,
                "height": 75,
                "hand": false
            }
        ]
    }
]
```

display_logic — логіка зв’язків між SKU (ціна / показ)

Масив правил зв’язку двох SKU (опцій або додаткових товарів товару) — sku_a (тригер) і sku_b (ціль). Кожне правило має тип type: • type="price" — якщо клієнт обрав sku_a, ціна цілі sku_b перераховується за полем price (форматом «формула ціни»). Не змінює базову ціну sku_b, поки тригер не активний. • type="show" — якщо обрано sku_a, ціль sku_b показується; якщо тригер не обрано — sku_b приховується (керує видимістю опції в калькуляторі товару). Правила застосовуються по черзі. price-формула: «200» → +200 до ціни sku_b; «-200» → -200; «20%» → +20% до ціни sku_b (мінус-відсоток теж працює через знак у числі перед %). Тобто це дельта до поточної ціни цілі, а не нова ціна. sku_a/sku_b — це sku конкретних опцій/додаткових товарів цього ж товару.

```
[
    {
        "sku_a": "RED",
        "sku_b": "WARR-1",
        "type": "price",
        "price": "20%"
    },
    {
        "sku_a": "COLOR-RED",
        "sku_b": "COLOR-GOLD",
        "type": "show",
        "price": "0"
    }
]
```

additionals — додаткові товари

Масив супутніх (додаткових) товарів. Ключ зіставлення — inner_id або sku. name обов’язкове; решта опційна. display_type за замовч. "checkbox", default_text — "Не обрано". Попередні зв’язки позначаються застарілими перед синхронізацією.

```
[
    {
        "inner_id": "2001",
        "name": "Гарантія",
        "sku": "WARR-1",
        "price": 199,
        "cost": 0,
        "display_type": "checkbox",
        "default_text": "Не обрано",
        "url_image": "https://..."
    }
]
```

discount_set — набір/бандл зі знижкою

Один об’єкт (не масив). Створює бандл: товар отримує type="bundle". products[] — масив inner_id товарів набору (резолвляться у локальні id, відсутні ігноруються). value/is_percent/name визначають знижку (name за замовч. = назва товару).

```
{
    "value": 10,
    "is_percent": true,
    "name": "Комплект «Стіл+стільці»",
    "products": [
        "1002",
        "1003",
        "1004"
    ]
}
```

- Тіло може бути як одним об’єктом товару, так і масивом об’єктів.
- inner_id — рядок; число приймається і тихо приводиться до string.
- custom_fields — об’єкт {code: value}; значення читаються назад у GET /product як custom_fields. Merge точковий: передані ключі перекривають, непередані значення зберігаються. Плоскі url_print/code_print застарілі й мапляться в custom_fields автоматично; name_print — знову реальне поле товару (іде напряму).
- Категорія: category_id — за внутрішнім id CRM (з GET /categories); category_inner_id — за зовнішнім inner_id. Якщо передані обидва — використовується category_id. Невідома категорія → товар потрапляє у failed.
- У failed[].error для помилок валідації — об’єкт {поле: [помилки]}, а для винятку — рядок повідомлення; failed[].data містить inner_id, name або "N/A".

POST `https://[ВАШ САБДОМЕН]-api.marchroute.com/api/v1/products/refresh`

Позначити застарілі товари scope: products:sync

Позначає товари з джерела API, які не оновлювалися понад тиждень, як застарілі та відв’язує їхні категорії. Тіло не читається.

Приклад відповіді

```
{
    "message": "Products refreshed successfully!",
    "count": 42
}
```

POST `https://[ВАШ САБДОМЕН]-api.marchroute.com/api/v1/categories`

Синхронізувати категорії (upsert за inner_id) scope: products:sync

Масив категорій. Батьки прив’язуються через inner_parent_id (0 = корінь). Обов’язкові лише inner_id і name — решта полів мають значення за замовчуванням.

Поле	Тип	Обов.	Де	Опис
`inner_id`	string	так	body	Зовнішній унікальний ID, ключ upsert (число приймається і приводиться до рядка)
`name`	string	так	body	Назва
`type`	string	—	body	Тип (category/collection). Не передали — category (в існуючої категорії тип не змінюється)
`inner_parent_id`	string	—	body	Зовнішній ID батька (0 або відсутній = корінь; число приймається і приводиться до рядка)
`img`	string	—	body	URL зображення
`icon`	string	—	body	Код іконки з довідника (кнопка «Іконки категорій» нижче). CRM показує іконку лише в категорій верхнього рівня
`order_index`	string	—	body	Порядок сортування

Тіло запиту

```
[
    {
        "inner_id": "cat-5",
        "name": "Взуття"
    },
    {
        "inner_id": "cat-51",
        "name": "Кросівки",
        "inner_parent_id": "cat-5"
    }
]
```

Приклад відповіді

```
[
    {
        "message": "Category updated successfully!",
        "category": {
            "id": 5,
            "inner_id": "cat-5",
            "name": "Взуття"
        }
    }
]
```

type — колекції кольорів (type="collection")

Колекція — особлива категорія, що об’єднує пов’язані товари (кольори/принти одного виробу) зі спільними характеристиками. • Кожен колір — окремий повноцінний товар зі своїми name, sku, inner_id, ціною та фото; у колекцію він потрапляє звичайною прив’язкою до цієї категорії (category_id / category_inner_id при синку товару). • name_print товару — підпис кольору (показується на картці кольору в CRM). • У списку товарів CRM колекція показується одним рядком (товар-представник із бейджем «Колекція»); пошук знаходить будь-який колір, але відкриває колекцію цілком. У редакторі всі кольори редагуються разом: спільні поля — фолбек, перевизначення — на конкретному кольорі. • Ціна в списку показується лише коли вона однакова у всіх кольорів. • Колекції не з’являються в дереві каталогу як категорії — це технічне групування. • Файловий імпорт/експорт: колонка collection_key — рядки з однаковим значенням збираються в колекцію (значення стає її назвою); товар живе лише в одній колекції — інша прив’язка знімається.

```
{
    "inner_id": "col-divan-lux",
    "name": "Диван Люкс",
    "type": "collection",
    "inner_parent_id": "cat-5"
}
```

- Частковий upsert: поля, не передані в елементі, в існуючої категорії не змінюються (зокрема type і батько).
- Порожнє тіло → 400 { "error": "No categories provided" }.
- Обробка поелементна, відповідь завжди 200 (крім порожнього тіла). category у відповіді — повний об’єкт категорії.
- Невалідний елемент: { message: "Validation failed", category: <inner_id>, error: {…} }. Якщо inner_parent_id≠0, а батька не знайдено: { message: "Parent category not found", category: {…} } (елемент пропускається).

Замовлення

Порядок роботи: спочатку довідники (способи доставки, події, знижки) — їх id передаються при створенні замовлення. Головний ендпоінт розділу — POST /order: у відповідь приходить створене замовлення (201) або, у двокроковому сценарії підтвердження, 202 з order_hash — тоді оформлення завершується через POST /order/confirm.

Після створення замовлення живе у воронці CRM: PUT /order/{id} оновлює поля, PUT /order/{id}/events/{code} рухає його подіями, GET /order/{id} і /history читають стан.

Життєвий цикл замовлення

```
Довідники: /events · /delivery/types · /pickup/points · /discount/types
     │
POST /order ──────────────────────────► 201 створено (+ auth-токен клієнта)
     │ is_need_confirm=true, клієнт існує, запит без auth
     ▼
    202 order_hash ──► POST /order/confirm ──► 201 створено

Далі:  GET /order/{id} · GET /order/{id}/history     — читання
       PUT /order/{id}                               — оновлення (організація, знижка, склад/постачальник)
       PUT /order/{id}/events/{code}                 — рух по воронці
       DELETE /person/{person_id}/order/{id}         — скасування клієнтом
```

GET `https://[ВАШ САБДОМЕН]-api.marchroute.com/api/v1/organizations`

Довідник організацій scope: payments:manage

Усі організації з рахунками та способами оплати. Відповідь — масив організацій. Опціональний пошук за назвою через query-параметр name.

Поле	Тип	Обов.	Де	Опис
`name`	string	—	query	Пошук за назвою організації (часткове співпадіння)

Приклад відповіді

```
[
    {
        "id": 1,
        "name": "ТОВ ...",
        "accounts": [
            {
                "id": 7,
                "iban": "UA...",
                "name": "mono",
                "bank": [],
                "paymentMethods": []
            }
        ]
    }
]
```

GET `https://[ВАШ САБДОМЕН]-api.marchroute.com/api/v1/events`

Довідник подій замовлення scope: orders:create

Список подій обробки замовлення (code для PUT /order/{id}/events/{code}).

Приклад відповіді

```
[
    {
        "id": 1,
        "name": "Підтверджено клієнтом",
        "code": "confirmByClient",
        "description": null
    }
]
```

GET `https://[ВАШ САБДОМЕН]-api.marchroute.com/api/v1/delivery/types`

Довідник способів доставки scope: orders:create

Активні способи доставки.

Приклад відповіді

```
[
    {
        "id": 1,
        "name": "Нова Пошта",
        "code": "nova",
        "is_default": true,
        "is_pickup": false
    }
]
```

GET `https://[ВАШ САБДОМЕН]-api.marchroute.com/api/v1/pickup/points`

Довідник точок самовивозу scope: orders:create

Усі точки самовивозу з геолокацією. ID точки передається у delivery.pickup_point_id при створенні замовлення зі способом самовивозу (is_pickup=true).

Приклад відповіді

```
[
    {
```

```
        "id": 1,
        "name": "Магазин на Хрещатику",
        "is_permanent": true,
        "location": {
            "id": 4,
            "country": "Україна",
            "region": "Київська",
            "city": "Київ",
            "street": "Хрещатик",
            "build": "1"
        }
    }
]
```


GET `https://[ВАШ САБДОМЕН]-api.marchroute.com/api/v1/discount/types`

Довідник типів знижок scope: orders:create

Типи знижок, відсортовані за назвою. За замовчуванням бандли виключені; передайте is_bundle, щоб змінити вибірку.

Поле	Тип	Обов.	Де	Опис
`is_bundle`	boolean	—	query	Фільтр за ознакою бандла. Без параметра — лише не-бандли (is_bundle=false)
`name`	string	—	query	Частковий збіг за назвою
`value`	numeric	—	query	Точний збіг за значенням знижки
`is_percent`	boolean	—	query	Фільтр за ознакою відсоткової знижки
`inner_id`	string	—	query	Зовнішній inner_id товару бандла (повертає бандли цього товару)

Приклад відповіді


```
[
    {
        "id": 3,
        "name": "Кімната",
        "value": "10.00",
        "is_percent": true
    }
]
```


POST `https://[ВАШ САБДОМЕН]-api.marchroute.com/api/v1/order`

Створити замовлення scope: orders:create

Створює замовлення з кошика магазину

Поле	Тип	Обов.	Де	Опис
`person`	object	—	body	Блок клієнта. Можна не передавати — замовлення створиться без клієнта (person_id: null) з warning.
`person.id`	integer	—	body	ID наявного клієнта CRM. Перекриває пошук за phone/email — замовлення прив’язується саме до цього клієнта. Немає такого id → ідентифікатор ігнорується, warning. Не збігається з клієнтом auth-токена → auth ігнорується, warning.
`person.phone`	string	—	body	Телефон клієнта. Невалідний формат не валить запит: телефон відкидається з warning.
`person.email`	string	—	body	Email клієнта. Невалідний формат не валить запит: email відкидається з warning.
`person.f_name`	string	—	body	Ім’я
`person.l_name`	string	—	body	Прізвище
`person.m_name`	string	—	body	По батькові
`person.password`	string(4-128)	—	body	Пароль акаунта (опц.)

`address`	object	—	body	Блок адреси доставки. Можна не передавати — тоді береться спосіб доставки за замовчуванням без адреси
`address.id`	integer	—	body	ID наявної адреси CRM. Перекриває побудову адреси з решти полів блоку (delivery_id/location/params ігноруються). Немає такого id → 422.
`address.delivery_id`	integer	—	body	ID способу доставки (з GET /delivery/types)
`address.pickup_point_id`	integer	—	body	ID точки самовивозу (з GET /pickup/points), якщо спосіб самовивозу
`address.params`	object	—	body	Ідентифікатори для поштових інтеграцій (REF-и довідників, індекс тощо). Набір полів залежить від інтегратора — див. блок «Поля доставки за інтегратором» нижче.
`address.location`	object	—	body	Гео-адреса. Якщо передано — type і city обов’язкові
`address.location.type`	string	—	body	Warehouse | Doors (обов’язкове, якщо передано address.location)
`address.location.city`	string	—	body	Місто
`address.location.ware`	string	—	body	Обов’язкове, якщо address.location.type=Warehouse
`address.location.street`	string	—	body	Обов’язкове, якщо address.location.type=Doors
`address.location.country`	string	—	body	Країна
`address.location.region`	string	—	body	Область
`address.location.district`	string	—	body	Район
`address.location.build`	string	—	body	Будинок
`address.location.flat`	string	—	body	Квартира
`address.location.place`	string	—	body	Місце
`products`	array	так	body	Позиції кошика
`products.*.inner_id`	string	—	body	Зовнішній inner_id товару (обов’язковий, якщо не передано products.*.id)
`products.*.id`	integer	—	body	Внутрішній ID товару в CRM. Перекриває пошук за inner_id. Немає такого id → 422.
`products.*.quantity`	integer	так	body	Кількість
`products.*.options`	array	—	body	Опції (кожна з sku)
`products.*.additionals`	array	—	body	Додаткові товари (кожен з sku)
`products.*.discount`	numeric	—	body	Пряма знижка на позицію (сума або відсоток) ЗА ОДНУ ОДИНИЦЮ товару — CRM сама множить на quantity. Пріоритетніша за discount_id; якщо не задано — підтягується знижка товару.
`products.*.discount_is_percent`	boolean	—	body	Чи знижка products.*.discount у відсотках. По замовч. false (сума). Якщо discount не задано — ігнорується.
`products.*.discount_id`	integer	—	body	ID наявного типу знижки
`is_need_confirm`	boolean	—	body	За замовч. false. true → не створювати без підтвердження (див. нижче)
`is_have_orders`	boolean	—	body	За замовч. true. Чи враховувати наявність попередніх замовлень клієнта при кроці підтвердження. false → не вимагати підтвердження навіть для наявного клієнта
`discount_id`	integer	—	body	ID знижки рівня замовлення (не бандл). Автоматично накидається на всі товари.
`organization_id`	integer	—	body	Задати організацію замовлення вручну (ID наявної організації). Якщо передано — автовизначення організації працювати для замовлення НЕ буде. Відповідальність за коректність організації (наявність необхідного рахунку тощо) на стороні клієнта.
`auth`	string	—	body	Токен клієнта (ідентифікує запит, пропускає підтвердження)

`is_split_by_supplier`	boolean	—	body	За замовч. true. Чи розбивати кошик на окремі замовлення по постачальниках (див. блок «Розподіл по постачальниках»). false → завжди одне замовлення; постачальник призначається, лише якщо один спільний покриває всіх виробників кошика.
`analytics`	object	—	body	UTM/аналітика (структура не валідується)
`comment`	string	—	body	Коментар до замовлення
`custom_fields`	object	—	body	Кастомні поля замовлення {code: value}. Довідник полів ведеться у картці замовлення CRM (блок «Кастомні поля»); значення читаються назад у GET /order/{id} та приходять у вебхуку order. При розбитті кошика по постачальниках — однакові для всіх створених замовлень

Тіло запиту

```
{
    "person": {
        "phone": "+380501234567",
        "f_name": "Іван",
        "l_name": "Петренко",
        "email": "ivan@example.com"
    },
    "comment": "Зателефонувати перед доставкою",
    "custom_fields": {
        "promo_code": "SPRING10"
    },
    "address": {
        "delivery_id": 1,
        "location": {
            "type": "Warehouse",
            "city": "Київ",
            "ware": "12"
        }
    },
    "products": [
        {
            "inner_id": "1001",
            "quantity": 2,
            "options": [
                {
                    "sku": "ST-1-RED"
                }
            ]
        }
    ]
}
```

Приклад відповіді

```
{
    "message": "Order created successfully",
    "order_id": 12345,
    "order": {
        "base_number": "250001",
        "total_amount": 1499,
        "address": "...",
        "person": [],
        "analytic": [],
        "products": []
    },
    "person_id": 12,
    "auth": "12|plainTextToken..."
}
```

discount — матриця знижок — пріоритети застосування

На позицію замовлення зберігаються ДВА поля знижки: пряме число (discount) і тип-знижка (discount_id). При розрахунку ціни спрацьовує лише ОДНА за таким пріоритетом: ① ПРЯМА знижка позиції → products[i].discount (з products[i].discount_is_percent) Якщо задано і > 0 — застосовується ЗАВЖДИ, перебиває будь-який тип-знижку (і позиції, і

замовлення). Якщо в запиті не задано — підставляється власна знижка товару (product.discount). ② ТИП-знижка ПОЗИЦІЇ → products[i].discount_id Перебиває тип-знижку рівня замовлення на цій позиції. ③ ТИП-знижка рівня ЗАМОВЛЕННЯ → discount_id (корінь тіла) Fallback для позицій, де тип-знижку позиції не задано. Бандли (is_bundle=true) ігноруються. ④ Немає знижки — ціна без знижки. ПРІОРІТЕТ: products[i].discount ⟶ перебиває ⟶ products[i].discount_id ⟶ перебиває ⟶ discount_id замовлення ⟶ fallback ⟶ product.discount (БД) Тобто: пряме число позиції > тип-знижка позиції > тип-знижка замовлення. Пряме число товару з БД працює лише коли в запиті знижку позиції не передали. ЗНИЖКА × QUANTITY: знижка застосовується до ціни ОДНІЄЇ одиниці, а вже уцінена ціна множиться на quantity. Тобто передавайте знижку за одну штуку — сумарну за всю кількість передавати НЕ потрібно (discount=100 при quantity=2 → −200 з підсумку).

Назва	Опис
`products[i].discount`	Пряме число (сума або %). НАЙВИЩИЙ пріоритет у розрахунку ціни. З discount_is_percent.
`products[i].discount_is_percent`	true → discount у відсотках; false/нема → абсолютна сума.
`products[i].discount_id`	Тип-знижка позиції. Перебиває discount_id замовлення на цій позиції. Діє, якщо немає прямого discount.
`discount_id (корінь)`	Тип-знижка рівня замовлення. Fallback для позицій без власної тип-знижки. Бандли не приймаються.
`product.discount (БД)`	Власна знижка товару. Підставляється, лише якщо в запиті не передано products[i].discount.

```
[
    {
        "_приклад": "Пряме число перебиває будь-який тип",
        "вхід": {
            "products[0].discount": 100,
            "products[0].discount_is_percent": false,
            "discount_id": 5,
            "price": 500
        },
        "застосовано": "−100 грн (пряме число позиції), discount_id=5 проігноровано у розрахунку",
        "ціна": "500 − 100 = 400 грн"
    },
    {
        "_приклад": "Тип позиції перебиває тип замовлення",
        "вхід": {
            "products[0].discount_id": 10,
            "discount_id": 5,
            "price": 1000,
            "DiscountType(10)": "30%"
        },
        "застосовано": "discount_id=10 позиції (30%), тип замовлення 5 проігноровано на цій позиції",
        "ціна": "1000 − 30% = 700 грн"
    },
    {
        "_приклад": "Відсоткова знижка позиції",
        "вхід": {
            "products[0].discount": 50,
            "products[0].discount_is_percent": true,
            "price": 1000
        },
        "застосовано": "−50% (пряме число позиції)",
        "ціна": "1000 − 500 = 500 грн"
    },
    {
        "_приклад": "Знижка за одиницю × quantity",
        "вхід": {
            "products[0].discount": 100,
            "products[0].quantity": 2,
            "price": 500
        },
        "застосовано": "−100 грн з ціни КОЖНОЇ одиниці, потім × quantity",
        "ціна": "(500 − 100) × 2 = 800 грн"
    }
]
```

is_need_confirm — двокроковий сценарій підтвердження

За замовчуванням false. Замовлення НЕ створюється одразу і повертається 202 з order_hash ЛИШЕ якщо одночасно: потрібне підтвердження (is_need_confirm=true) І клієнт уже існує (за phone/email) І запит не ідентифікований (немає auth-

токена). CRM зберігає дані у кеш на 24 год. Магазин показує клієнту підтвердження (напр. SMS-код), а потім завершує оформлення викликом POST /order/confirm з цим хешем. За замовчуванням замовлення створюється одразу (201).

```
{
    "message": "Order data saved.",
    "order_hash": "a1b2c3...",
    "person_id": 12
}
```

auth — токен клієнта у відповіді

У відповіді 201 поле auth містить auth-токен клієнта: акаунт створюється/забезпечується автоматично. Якщо в person передано password і клієнт новий — встановлюється саме цей пароль. Токен можна одразу використати для запитів кабінету клієнта (/person*). Виняток: замовлення без клієнта (person_id: null) — auth буде null.

warnings — м’які помилки клієнта (warnings)

Проблеми з ідентифікацією клієнта НЕ валять створення замовлення. Замовлення створюється (201), а деталі повертаються масивом warnings у відповіді (об’єкти {warning: код, message: текст}) та дописуються в коментар замовлення (існуючий коментар не затирається). 422 за клієнтом не буває взагалі — будь-яка проблема лише деградує до warning.

Назва	Опис
`phone_not_valid`	Невалідний телефон — відкидається.
`email_not_valid`	Невалідний email — відкидається.
`person_id_not_found`	person.id не знайдено — ідентифікатор ігнорується.
`auth_mismatch`	person.id не збігається з клієнтом auth-токена — auth ігнорується, замовлення йде на person.id.
`contacts_conflict`	Телефон і email належать різним клієнтам — пріоритет за телефоном, email лишається його власнику.
`phone_taken`	Телефон зайнятий іншим клієнтом (при person.id/auth) — контакт не додається.
`email_taken`	Email зайнятий іншим клієнтом (при person.id/auth) — контакт не додається.
`person_missing`	Жодного валідного ідентифікатора (id/телефон/email) — замовлення створюється БЕЗ клієнта: person_id null, auth null.

```
{
    "message": "Order created successfully",
    "order_id": 12345,
    "person_id": null,
    "warnings": [
        {
            "warning": "email_not_valid",
            "message": "Попередження: імейл невалідний - g.cc@l"
        },
        {
            "warning": "person_missing",
            "message": "Попередження: замовлення створено без клієнта — не передано жодного валідного ідентифікатора (id,
телефон або email)"
        }
    ]
}
```

is_split_by_supplier — розподіл по постачальниках

Кожен товар належить виробнику, а виробник пов’язаний з одним чи кількома постачальниками. При створенні замовлення кошик ділиться так: ① Збираються виробники всіх товарів кошика. ② Обирається постачальник, що покриває НАЙБІЛЬШЕ виробників кошика — його товари стають одним замовленням. Крок повторюється для решти, тож кошик розбивається на мінімальну кількість замовлень (номери base_number-1, base_number-2...). ③ Якщо кілька постачальників покривають однаково — ФІЗИЧНИЙ СКЛАД ПРІОРИТЕТНІШИЙ ЗА ВІРТУАЛЬНИЙ: спершу складський, чий склад збігається з замовленням за уточнюючими параметрами (organization/product/category — див. GET /warehouses, clarifications), далі просто перший складський з активним складом, і лише якщо складських немає — перший без складу за position зв’язки vendor-supplier. ④ Товари, виробники яких не мають постачальника, потрапляють в замовлення без постачальника. Постачальник із warehouse_id — «складський» (GET /vendors → suppliers[].warehouse_id): замовлення, розподілене на нього, автоматично відвантажується з його складу. is_split_by_supplier=false вимикає РОЗБИТТЯ: завжди створюється одне замовлення (так працюють інтеграції eCommerce — зовнішнє замовлення не можна дробити). Постачальник при цьому все одно резолвиться: якщо один спільний покриває ВСІХ виробників кошика — призначається він (при кількох —

те саме уточнення складами, крок ③); спільного немає — замовлення лишається без постачальника.

analytics — аналітика замовлення (UTM, gclid, джерело)

Об’єкт зберігається в аналітиці замовлення «як є»: кожен ненульовий ключ записується, тож зберігається будь-який переданий ключ. Якщо gclid не передано, але клієнт уже мав замовлення за останню добу — gclid (і timestamp/ga_timestamp) підтягуються з попереднього замовлення. Поля comment, e_commerce_id, external_order_id, source_id, передані на верхньому рівні тіла, також потрапляють у аналітику. Розпізнавані ключі:

Назва	Опис
`utm_source`	Джерело трафіку (google, facebook, direct…)
`utm_medium`	Канал (cpc, organic, email, referral…)
`utm_campaign`	Назва кампанії
`utm_term`	Ключове слово (для пошукових кампаній)
`utm_content`	Варіант оголошення / контент
`gclid`	Google Click ID — для офлайн-конверсій Google Ads
`fbclid`	Facebook Click ID
`referrer`	URL сторінки-джерела переходу
`source_id`	ID джерела зі списку sources (канал)
`external_order_id`	Номер замовлення у зовнішній системі
`timestamp`	Unix-час кліку/візиту (для атрибуції Google Ads)
`ga_timestamp`	Unix-час події для Google Analytics

```
{
    "utm_source": "google",
    "utm_medium": "cpc",
    "utm_campaign": "spring",
    "utm_term": "диван",
    "gclid": "Cj0KCQ...",
    "source_id": 3
}
```

analytics.touchpoints — точки контакту (історія візитів)

Необов’язковий масив touchpoints[] усередині analytics — хронологія візитів клієнта до оформлення. Кожна точка зберігається окремим записом. UTM з останнього непорожнього touchpoint агрегуються в analytics (явно передані поля analytics перекривають їх). Поля точки: ts (Unix-час візиту), utm_source, utm_medium, utm_campaign, utm_term, utm_content, pages (масив переглянутих сторінок), engaged_seconds (секунди залученості).

```
{
    "touchpoints": [
        {
            "ts": 1716200000,
            "utm_source": "google",
            "utm_medium": "cpc",
            "utm_campaign": "spring",
            "pages": [
                "/",
                "/catalog",
                "/product/1001"
            ],
            "engaged_seconds": 142
        },
        {
            "ts": 1716203600,
            "utm_source": "direct",
            "utm_medium": "none",
            "pages": [
                "/cart",
                "/checkout"
            ],
            "engaged_seconds": 95
```

```
        }
    ]
}
```

organization_id — ручний вибір організації та автовизначення

За замовчуванням організація замовлення визначається автоматично за правилами розподілу (постачальник/категорії товарів, оплати тощо). Щоб зафіксувати організацію вручну, передайте organization_id — тоді автовизначення для цього замовлення взагалі не запускається, і замовлення прикріплюється саме до переданої організації. Окремий прапорець для режиму при створенні не передається: режим визначається лише наявністю organization_id (є → ручний/фіксований, немає → автовизначення). Якщо замовлення розбивається на кілька (різні постачальники), задана організація застосовується до всіх частин. CRM не перевіряє коректність переданої організації (наявність активного рахунку тощо) — відповідальність повністю на стороні клієнта. Змінити організацію вже створеного замовлення (зокрема повернути автовизначення) можна через PUT /order/{id}.

```
{
    "organization_id": 1
}
```

address — доставка — загальна логіка

Перевізник визначається способом доставки address.delivery_id (GET /delivery/types): за ним CRM знаходить інтегратора (Нова Пошта / Укрпошта / Meest / самовивіз). Тип доставки задається в address.location.type: • Warehouse — на відділення; • Doors — адресна (кур’єром до дверей). Ідентифікатори з довідників перевізника передаються в address.params (city / ware / street — це REF-и, а не назви). Текстова адреса (місто, вулиця, будинок) — в address.location. Назву населеного пункту в address.location.city БАЖАНО передавати ЗАВЖДИ — незалежно від режиму та типу доставки. ПІДЙОМ НА ПОВЕРХ: поверх передається в address.params.floor (ціле число 1–57), для Укрпошти додатково address.params.lift (true/false — чи є ліфт). Значення підтягуються у форму ТТН і їдуть у накладну автоматично. Працює лише для адресної доставки (location.type = "Doors") і лише з указаною квартирою (location.flat) — без квартири підйом недоступний (помилка). Потрібен увімкнений перемикач «Підтримувати підйом на поверх» у налаштуваннях модуля доставки. Нижче — точний набір полів для кожного інтегратора (тисніть кнопку).

- 422: { "error": "Validation failed", "details": { ... } }.
- Доставка: перевізник — за address.delivery_id; тип (Warehouse/Doors) — за address.location.type; REF-и довідників — в address.params. Деталі за інтегратором — у кнопках вище.

POST `https://[ВАШ САБДОМЕН]-api.marchroute.com/api/v1/order/confirm`

Підтвердити відкладене замовлення scope: orders:create

Створює раніше відкладене замовлення за його order_hash (отриманим у відповіді 202 від POST /order). Відповідь ідентична POST /order (201): з ключами order та auth.

Поле	Тип	Обов.	Де	Опис
`order_hash`	string	так	body	Хеш відкладеного замовлення (з відповіді 202)

Приклад відповіді

```
{
    "message": "Order created successfully",
    "order_id": 12345,
    "order": [],
    "person_id": 12,
    "auth": "12|plainTextToken..."
}
```

- 404: { "error": "Order data not found" } — якщо кеш порожній або прострочений (24 год).

GET `https://[ВАШ САБДОМЕН]-api.marchroute.com/api/v1/order/{id}`

Отримати замовлення scope: orders:create

Одне замовлення зі зв’язками: клієнт (person), товари (products), доставка (delivery), накладна, статус, аналітика, організація (лише id/назва) та очікувані оплати.

Поле	Тип	Обов.	Де	Опис

`id`	integer	так	path	ID замовлення

Приклад відповіді


```
{
    "status": "success",
    "data": {
        "id": 12345,
        "number": "250001",
        "total_amount": 1499,
        "total_payed": 0,
        "total_remaining": 1499,
        "state": "green",
        "current_state": {
            "checkpoint_name": "Очікує створення",
            "description": null,
            "status": {
                "code": "start",
                "name": "Узгодження"
            },
            "state": "green"
        },
        "person": [],
        "products": [],
        "delivery": [],
        "status": [],
        "organization": [],
        "custom_fields": {
            "promo_code": "SPRING10"
        }
    }
}
```


- 404: { "status": "error", "message": "Order not found" }.
- custom_fields — об’єкт кастомних полів замовлення {code: value}; якщо жодне не заповнене — {}. Довідник ведеться у картці замовлення CRM (блок «Кастомні поля»), значення задаються там же або через POST /order і PUT /order/{id}.

GET `https://[ВАШ САБДОМЕН]-api.marchroute.com/api/v1/order/{id}/history`

Історія станів замовлення scope: orders:create

Масив записів історії (status, checkpoint), по одному на кожен унікальний статус, найновіші зверху.

Поле	Тип	Обов.	Де	Опис
`id`	integer	так	path	ID замовлення

Приклад відповіді


```
[
    {
        "status": {
            "name": "Узгодження"
        },
        "checkpoint": [],
        "created_at": "2026-05-21T10:00:00Z"
    }
]
```


PUT `https://[ВАШ САБДОМЕН]-api.marchroute.com/api/v1/order/{id}`

Оновити замовлення scope: orders:create

Оновлює організацію/прапорець ручної організації/знижку/аналітику, а також склад або постачальника замовлення. Поля organization_id та is_manual_organization записуються незалежно одне від одного.

Поле	Тип	Обов.	Де	Опис
`id`	integer	так	path	ID замовлення

`organization_id`	integer	—	body	Змінює організацію замовлення (ID наявної організації або null). При передачі ненульового значення, is_manual_organization автоматично стає true (автовизначення вимикається).
`is_manual_organization`	boolean	—	body	Чи застосовувати правила автовизначення організації. true → автовизначення вимкнено (фіксована організація); false → правила застосовуються (повертає автовизначення). УВАГА: якщо передати false, організація може бути ПЕРЕВИЗНАЧЕНА правилами розподілу — навіть та, що ви передали в цьому ж запиті. Записується незалежно від organization_id.
`discount_id`	integer	—	body	ID знижки рівня замовлення (не бандл; або null). Автоматично накидається на всі товари замовлення з перерахунком сум.
`warehouse_id`	integer	—	body	Перенести замовлення на склад (ID з GET /warehouses). Постачальник складу прив’язується АВТОМАТИЧНО. Не можна передавати разом із supplier_id → 422.
`supplier_id`	integer	—	body	Змінити постачальника замовлення (ID з GET /suppliers). Постачальник має бути пов’язаний з виробниками ВСІХ товарів замовлення, інакше 422 «непідходяще значення». Якщо постачальник складський (має warehouse_id) — замовлення автоматично переноситься на його склад.
`analytics`	object	—	body	Повна логіка як при створенні: touchpoints[], агрегація UTM, fallback gclid; оновлює наявну аналітику замовлення.
`custom_fields`	object	—	body	Кастомні поля замовлення {code: value}. Точковий merge: передані ключі перекривають, порожнє значення (null / "") видаляє ключ, непередані значення зберігаються

Тіло запиту

```
{
    "organization_id": 1,
    "custom_fields": {
        "promo_code": "SPRING10"
    },
    "analytics": {
        "utm_source": "google",
        "touchpoints": [
            {
                "ts": 1716200000,
                "utm_source": "google",
                "pages": [
                    "/cart"
                ]
            }
        ]
    }
}
```

Приклад відповіді

```
{
    "message": "Order updated successfully"
}
```

- Блок analytics обробляється тією ж логікою, що й при POST /order (туди ж дивіться структуру touchpoints).
- is_manual_organization:false вмикає автовизначення — задана організація (навіть передана поруч) може бути замінена правилами розподілу. Щоб зафіксувати організацію, передавайте лише organization_id (is_manual_organization виставиться у true автоматично).
- warehouse_id і supplier_id — взаємовиключні: передавайте ЛИШЕ ОДНЕ з них. Склад завжди тягне за собою свого постачальника; складський постачальник тягне за собою свій склад.
- При зміні складу резерви замовлення знімаються зі старого складу і переоформлюються на новому автоматично.
- 422 { "details": { "supplier_id": ["Непідходяще значення: постачальник не пов’язаний з виробниками товарів замовлення"] } } — постачальник не покриває виробників товарів.

PUT `https://[ВАШ САБДОМЕН]-api.marchroute.com/api/v1/order/{id}/events/{code}`

Ініціювати подію замовлення scope: orders:create

Запускає подію обробки замовлення {code} (коди — з GET /events).

Поле	Тип	Обов.	Де	Опис
`id`	integer	так	path	ID замовлення
`code`	string	так	path	Код події
`custom`	array	—	body	Дані події (за замовч. [])

Приклад відповіді

```
{
    "error": "Event initialized successfully."
}
```

- Успіх повертається під ключем "error" (історична назва поля).
- 429: { "error": "Зачекайте і повторіть спробу" } — якщо заблоковано.

DELETE `https://[ВАШ САБДОМЕН]-api.marchroute.com/api/v1/person/{person_id}/order/{id}`

Скасувати замовлення клієнтом scope: orders:create

Скасовує замовлення клієнта. Замовлення має належати вказаному клієнту — інакше 404. Без reason_id — причина «Скасовано клієнтом» (статус «Скасовано»). З reason_id назва та поведінка беруться зі довідника: якщо причина is_total — замовлення видаляється повністю, інакше переводиться у статус «Скасовано».

Поле	Тип	Обов.	Де	Опис
`person_id`	integer	так	path	ID клієнта — власника замовлення
`id`	integer	так	path	ID замовлення
`reason_id`	integer	—	body	ID причини скасування зі справочника причин

Приклад відповіді

```
{
    "message": "Order deleted successfully"
}
```

- 404: { "error": "Order not found" } — замовлення не існує або не належить клієнту.

GET `https://[ВАШ САБДОМЕН]-api.marchroute.com/api/v1/checks/redirect/{from}`

Чеки за коротким посиланням scope: orders:create

За коротким посиланням на чек знаходить замовлення і повертає всі URL фіскальних чеків. З параметром strong=true повертається лише той чек, на який вказує саме це посилання.

Поле	Тип	Обов.	Де	Опис
`from`	string	так	path	Токен короткого посилання на чек (redirect_code)
`strong`	boolean	—	query	true — лише чек цього redirect_code, без решти чеків замовлення

Приклад відповіді

```
{
    "order_id": 12345,
    "order_number": "250001",
    "checks": [
        {
            "id": 1,
            "ref": "abc123",
```

```
            "amount": 1499,
            "created_at": "2026-05-21 10:00:00",
            "url_to": "https://.../abc123/html"
        }
    ]
}
```

Авторизація клієнта

В API два різні токени — не плутайте. API-токен ІНТЕГРАЦІЇ створюється в CRM і авторизує запити вашого магазину (саме він у заголовку всіх ендпоінтів цієї документації). Auth-токен КЛІЄНТА — особистий токен покупця для його кабінета (розділ «Кабінет клієнта»).

Цей розділ — про отримання auth-токена клієнта. Два шляхи: безпарольний (GET /auth/person/{id}/link → код → POST /auth/login/code → токен) і класичний логін/пароль (POST /auth/login; реєстрація — POST /auth/register). Токен клієнта також повертається одразу у відповіді POST /order (поле auth).

POST `https://[ВАШ САБДОМЕН]-api.marchroute.com/api/v1/auth/login/code`

Обміняти код на токен (ліміт 60/хв) scope: person:auth

Обмінює тимчасовий код авторизації (дійсний 2 тижні) на токен клієнта. Код приходить із персонального посилання авторизації — його можна отримати за ID клієнта через GET /auth/person/{id}/link або з розсилок (напр. поле url у подіях eSputnik).

Поле	Тип	Обов.	Де	Опис
`code`	string	так	body	Тимчасовий код авторизації

Приклад відповіді

```
{
    "status": "success",
    "token": "1|plainTextToken...",
    "person_id": 123
}
```

- 404: { "status": "error", "message": "Code not found" }.

GET `https://[ВАШ САБДОМЕН]-api.marchroute.com/api/v1/auth/person/{id}/link`

Код авторизації клієнта за ID (ліміт 60/хв) scope: person:auth

Обмінює ID клієнта на його персональний код авторизації. Повертається код (7 символів), який клієнт обмінює на токен через POST /auth/login/code — наприклад, для безпарольного входу за посиланням. Код дійсний 2 тижні; протухлий перевипускається автоматично при зверненні.

Поле	Тип	Обов.	Де	Опис
`id`	integer	так	path	ID клієнта (person_id)

Приклад відповіді

```
{
    "person_id": 12,
    "auth_link": "a1b2c3d"
}
```

- 404, якщо клієнта або його користувача не знайдено.

POST `https://[ВАШ САБДОМЕН]-api.marchroute.com/api/v1/auth/login`

Вхід (логін+пароль) → auth-токен scope: person:auth

Логін за email або телефоном + пароль.

Поле	Тип	Обов.	Де	Опис

`login`	string	так	body	Email або телефон
`password`	string	так	body	Пароль

Приклад відповіді

```
{
    "status": "success",
    "token": "1|plainTextToken...",
    "person": {
        "id": 12,
        "f_name": "Іван",
        "l_name": "Петренко",
        "phone": "+380...",
        "email": "user@example.com",
        "full_name": "Іван Петренко"
    }
}
```

- 401 — невірні дані або телефонний логін заборонено; 403 — tenant denied; 503 — сервіс недоступний.

POST `https://[ВАШ САБДОМЕН]-api.marchroute.com/api/v1/auth/register`

Реєстрація клієнта → auth-токен scope: person:auth

Поле	Тип	Обов.	Де	Опис
`phone`	string	—	body	Потрібен phone або email (хоча б один)
`email`	string	—	body	Потрібен phone або email; перевіряється формат
`f_name`	string	так	body	Ім’я (макс. 255)
`l_name`	string	так	body	Прізвище (макс. 255)
`m_name`	string	—	body	По батькові
`password`	string(8-128)	—	body	Пароль (випадковий, якщо не задано)

Приклад відповіді

```
{
    "status": "success",
    "token": "1|plainTextToken...",
    "person": {
        "id": 12,
        "full_name": "Іван Петренко"
    }
}
```

- Якщо phone або email уже зареєстровано — 409: { "status": "error", "message": "Phone is already registered." } (або Email).

POST `https://[ВАШ САБДОМЕН]-api.marchroute.com/api/v1/auth/person`

Перевірити існування клієнта scope: person:auth

Поле	Тип	Обов.	Де	Опис
`phone`	string	—	body	Потрібен phone або email
`email`	string	—	body	Потрібен phone або email

Приклад відповіді

```
{
    "person_id": 12,
    "phone": "+380...",
    "email": "user@example.com",
```

```
    "participant": "false"
}
```

- participant — рядок "true"/"false". 404, якщо не знайдено.

POST `https://[ВАШ САБДОМЕН]-api.marchroute.com/api/v1/auth/person/generate`

Згенерувати пароль клієнту scope: person:auth

Генерує новий пароль для контрагента, надсилає SMS (пріоритетно) або email, інвалідує токени.

Поле	Тип	Обов.	Де	Опис
`phone`	string	—	body	Потрібен phone або email
`email`	string	—	body	Потрібен phone або email
`length`	integer(4-64)	—	body	Довжина пароля (за замовч. 4)
`regex`	string	—	body	numeric (за замовч.) | string

Приклад відповіді

```
{
    "message": "New password generated",
    "person_id": 12,
    "phone": "+380...",
    "participant": "false"
}
```

Кабінет клієнта (auth)

Кабінет покупця. Ендпоінти БЕЗ {id} (/person, /person/orders, /person/password) працюють з auth-токеном КЛІЄНТА у заголовку Authorization — його дає розділ «Авторизація клієнта» або поле auth у відповіді POST /order. Ендпоінти З {id} (/person/{id}, /person/{id}/orders) — серверні: викликаються з API-токеном інтеграції (scope orders:create), коли токена клієнта на руках немає.

GET `https://[ВАШ САБДОМЕН]-api.marchroute.com/api/v1/person`

Профіль поточного клієнта scope: auth:customer

Потрібен токен клієнта (Authorization: Bearer <token> з /auth/login), а не API-токен інтеграції.

Приклад відповіді

```
{
    "status": "success",
    "data": {
        "id": 12,
        "email": "user@example.com",
        "phone": "+380...",
        "f_name": "Іван",
        "l_name": "Петренко",
        "m_name": "",
        "avatar": null
    }
}
```

GET `https://[ВАШ САБДОМЕН]-api.marchroute.com/api/v1/person/orders`

Замовлення поточного клієнта scope: auth:customer

Замовлення авторизованого клієнта з усіма зв’язками, платежами, чеками та підсумками (включно з архівними). Пагінація: 20 замовлень на сторінку.

Поле	Тип	Обов.	Де	Опис
`page`	integer	—	query	Номер сторінки (за замовч. 1, по 20 замовлень)

Приклад відповіді

```
{
    "status": "success",
    "data": [
        {
            "id": 12345,
            "number": "250001",
            "state": "green"
        }
    ],
    "meta": {
        "current_page": 1,
        "last_page": 5,
        "per_page": 20,
        "total": 92
    }
}
```

GET `https://[ВАШ САБДОМЕН]-api.marchroute.com/api/v1/person/order/{id}`

Замовлення поточного клієнта за ID scope: auth:customer

Повертає замовлення, лише якщо воно належить авторизованому клієнту (за токеном клієнта). Якщо замовлення не існує або належить іншому клієнту — 404. Таким запитом можна переконатися, що замовлення належить користувачу.

Поле	Тип	Обов.	Де	Опис
`id`	integer	так	path	ID замовлення

Приклад відповіді

```
{
    "status": "success",
    "data": {
        "id": 12345,
        "number": "250001",
        "state": "green"
    }
}
```

- 404: { "status": "error", "message": "Order not found" } — не існує або чуже замовлення.

POST `https://[ВАШ САБДОМЕН]-api.marchroute.com/api/v1/person/password`

Змінити пароль scope: auth:customer

Поле	Тип	Обов.	Де	Опис
`current_password`	string	так	body	Має збігатися з поточним
`password`	string(4-128)	так	body	Новий пароль

Приклад відповіді

```
{
    "status": "success",
    "message": "Password updated"
}
```

- 401, якщо поточний пароль Неправильний.

GET `https://[ВАШ САБДОМЕН]-api.marchroute.com/api/v1/person/{id}/orders`

Замовлення клієнта за ID scope: orders:create

Пагінація: 20 замовлень на сторінку.

Поле	Тип	Обов.	Де	Опис
`id`	integer	так	path	ID клієнта
`page`	integer	—	query	Номер сторінки (за замовч. 1, по 20 замовлень)

Приклад відповіді

```
{
    "status": "success",
    "data": [],
    "meta": {
        "current_page": 1,
        "last_page": 5,
        "per_page": 20,
        "total": 92
    }
}
```

GET `https://[ВАШ САБДОМЕН]-api.marchroute.com/api/v1/person/{id}`

Профіль клієнта за ID scope: orders:create

Поле	Тип	Обов.	Де	Опис
`id`	integer	так	path	ID клієнта

Приклад відповіді

```
{
    "status": "success",
    "data": {
        "id": 12,
        "email": "user@example.com",
        "phone": "+380...",
        "f_name": "Іван",
        "l_name": "Петренко"
    }
}
```

Оплати

Порядок роботи: довідники (методи, банки, типи оплат) → створення оплати → за потреби фіскалізація чеків. Способів створити оплату кілька, кожен під свій сценарій — щоб не гадати, одразу відкрийте гід «Який метод оплати обрати?» нижче. Оплата завжди прив’язується до замовлень масивом orders[] = [{ order_id, amount? }].

GET `https://[ВАШ САБДОМЕН]-api.marchroute.com/api/v1/payment/methods`

Доступні методи оплати scope: payments:manage

Методи, що видимі та придатні для використання. Можна звузити вибірку за банком та/або типом оплати. Поле payment_mode визначає, як саме створюється та повертається платіж: «module» — через платіжний модуль (шлюз), який налаштовано в CRM; «callback» — CRM надсилає запит у ваш магазин (вебхук create_payment), і ви обробляєте платіж самі; «instant» — платіж створюється одразу, без шлюзу й вебхуку (готівка, термінал тощо); «manual» — ручне додавання IBAN-платежу: у extra передаються account_id, payer та purpose, платіж створюється одразу. Поле is_auto = true означає, що метод видимий і автоматично повертається (режим module з активним модулем або callback).

Поле	Тип	Обов.	Де	Опис
`bank_id`	numeric	—	query	Фільтр за ID банку (точний збіг)
`payment_type_id`	numeric	—	query	Фільтр за ID типу оплати (точний збіг)

Приклад відповіді

```
[
    {
```

```
        "id": 5,
        "is_view": true,
        "payment_mode": "module",
        "is_auto": true,
        "name": "Оплата карткою",
        "bank": {
            "id": 2,
            "name": "mono"
        },
        "payment_type": {
            "id": 1,
            "code": "card",
            "name": "Картка"
        },
        "module": {
            "code": "monopay",
            "active": true
        }
    }
]
```


- payment_mode: module | callback | instant | manual.

GET `https://[ВАШ САБДОМЕН]-api.marchroute.com/api/v1/payment/banks`

Довідник банків scope: payments:manage

Список усіх банків (для побудови способів оплати/фільтрів), відсортований за назвою.

Поле	Тип	Обов.	Де	Опис
`name`	string	—	query	Пошук за назвою (частковий збіг)

Приклад відповіді


```
[
    {
        "id": 2,
        "name": "mono",
        "icon": "mono.svg",
        "full_name": "Monobank"
    }
]
```


GET `https://[ВАШ САБДОМЕН]-api.marchroute.com/api/v1/payment/types`

Довідник типів оплат scope: payments:manage

Список усіх типів оплат (code/name), відсортований за назвою.

Поле	Тип	Обов.	Де	Опис
`code`	string	—	query	Фільтр за кодом типу (точний збіг)

Приклад відповіді


```
[
    {
        "id": 1,
        "name": "Картка",
        "code": "card"
    }
]
```


Який метод оплати обрати?

У CRM є кілька ендпоінтів створення оплати — кожен під свій сценарій. Коротко: якщо платіж має провести МОДУЛЬ, налаштований у CRM, — беріть POST /payment (отримаєте посилання). Якщо гроші вже отримані і треба просто

ЗАФІКСУВАТИ факт оплати в CRM — POST /payment/manual. Якщо це оплата за реквізитами на конкретний рахунок — POST /payment/iban. Якщо оплату створюєте ВИ на своєму боці (напр. інтеграція, якої немає в CRM) і хочете спершу завести її як очікувану, а підтвердити після фактичної оплати клієнтом — POST /payment/pending, а потім POST /payment/pending/confirm. Оберіть сценарій нижче.

— Схема вибору

Платіж проводить шлюз, налаштований у CRM (Plata by Mono, monopay тощо)? → ТАК → POST /payment → отримуєте посилання (або текст-інструкцію) → клієнт платить. Готово. Гроші вже отримані, треба лише внести факт у CRM і прив’язати до замовлень? → POST /payment/manual Оплата за реквізитами на конкретний рахунок (IBAN)? → POST /payment/iban Оплату створюєте ВИ у себе (стороння інтеграція, якої немає в CRM)? → POST /payment/pending (заводимо очікувану) → коли клієнт оплатив → POST /payment/pending/confirm (підтверджуємо).

POST `https://[ВАШ САБДОМЕН]-api.marchroute.com/api/v1/payment`

Створити оплату scope: payments:manage

Створює оплату обраним способом (payment_method_id) для одного або кількох замовлень. Замовлення завжди передаються масивом orders[] = [{ order_id, amount? }] — від одного до будь-якої кількості. Поле amount у кожному рядку необов’язкове: якщо його не передано (або значення ≤ 0), сума береться як поточний залишок замовлення (total_remaining). Так само можна передати amount лише для частини замовлень — решта порахується за залишком. Підсумкова сума платежу = сума всіх amount (явних і порахованих за залишком). Якщо всі замовлення вже сплачені (загальна сума 0) — повертається помилка. Подальша поведінка залежить від режиму способу оплати (payment_mode): module — створюється сесія шлюзу, яка повертає АБО посилання на оплату (status=link), АБО лише текст-інструкцію без посилання (status=awaiting, напр. «клієнт повинен підтвердити в застосунку») — залежно від провайдера; callback — запит у ваш магазин (вебхук create_payment), магазин сам обирає режим відповіді (посилання або текст); instant — платіж створюється одразу (status=paid); manual — ручний IBAN-платіж: створюється одразу (status=paid), обов’язкові extra.account_id та extra.payer.

Поле	Тип	Обов.	Де	Опис
`orders`	array	так	body	Масив замовлень для оплати (мін. 1)
`orders.*.order_id`	integer	так	body	ID замовлення
`orders.*.amount`	numeric	—	body	Сума на це замовлення (>0). Якщо не передано — береться залишок замовлення
`payment_method_id`	integer	так	body	ID наявного способу оплати
`extra`	array	—	body	Додаткова інформація, специфічна для модуля — набір полів залежить від провайдера
`extra.phone`	string	—	body	Телефон клієнта для цього платежу, якщо відрізняється від указаного в замовленні
`extra.card`	string	—	body	Номер/останні цифри картки — для модулів, що цього потребують (напр. УкрСиб)
`extra.months`	integer	—	body	Кількість місяців розстрочки для installments-модулів (за замовч. 3)
`extra.account_id`	integer	—	body	Для manual: ID IBAN-рахунку організації, на який зараховано платіж
`extra.payer`	string	—	body	Для manual: платник (обов’язково)
`extra.purpose`	string	—	body	Для manual: призначення платежу

Тіло запиту

```
{
    "payment_method_id": 5,
    "orders": [
        {
            "order_id": 12345
        },
        {
            "order_id": 12346,
            "amount": 500
        }
    ],
    "extra": {
        "phone": "380501112233",
        "months": 6
    }
```

```
}
```

Приклад відповіді

```
{
    "status": "link",
    "link": "https://pay...",
    "pending_id": 45,
    "text": "..."
}
```

- status: paid | link | awaiting | failed.
- paid → { payment_id }; link → { link, pending_id, text? }; awaiting → { pending_id, text }.
- failed → HTTP 400 з полем { error }.

POST `https://[ВАШ САБДОМЕН]-api.marchroute.com/api/v1/payment/manual`

Додати оплату і прив’язати до замовлень scope: payments:manage

Замовлення передаються масивом orders[] = [{ order_id, amount? }] — будь-яка кількість. amount у рядку необов’язковий: якщо не передано (або ≤ 0) — береться залишок замовлення. Загальна сума платежу рахується автоматично як сума всіх amount.

Поле	Тип	Обов.	Де	Опис
`payment_method_id`	integer	так	body	ID способу оплати (GET /payment/methods) — тип платежу визначається ним
`transaction_number`	string	так	body	Унікальний ID транзакції (ключ блокування)
`organization_id`	numeric	так	body	ID організації
`orders`	array	так	body	Масив замовлень (мін. 1)
`orders.*.order_id`	integer	так	body	ID замовлення
`orders.*.amount`	numeric	—	body	Сума на це замовлення (>0). Якщо не передано — береться залишок
`account_id`	numeric	так	body	ID рахунку
`commission`	numeric	—	body	Комісія
`info`	string	—	body	Примітка

Тіло запиту

```
{
    "payment_method_id": 5,
    "transaction_number": "card_1716",
    "organization_id": 1,
    "orders": [
        {
            "order_id": 12345,
            "amount": 1499
        },
        {
            "order_id": 12346
        }
    ]
}
```

Приклад відповіді

```
{
    "message": "Payment created successfully",
    "payment_id": 123
}
```

POST `https://[ВАШ САБДОМЕН]-api.marchroute.com/api/v1/payment/iban`

Створити IBAN-оплату scope: payments:manage

Поле	Тип	Обов.	Де	Опис
`amount`	numeric	так	body	Сума
`account_id`	numeric	так	body	ID IBAN-рахунку
`payer`	string	так	body	Платник
`transaction_number`	string	—	body	ID транзакції (за замовч. "-")
`purpose`	string	—	body	Призначення платежу (за замовч. "-")
`date`	date	—	body	Дата (за замовч. now)

Тіло запиту

```
{
    "amount": 1499,
    "account_id": 7,
    "payer": "Іван Петренко",
    "purpose": "Оплата за замовлення 250001"
}
```

Приклад відповіді

```
{
    "message": "Payment created successfully",
    "payment_id": 123
}
```

POST `https://[ВАШ САБДОМЕН]-api.marchroute.com/api/v1/payment/pending`

Створити очікувану оплату scope: payments:manage

Замовлення передаються масивом orders[] = [{ order_id, amount? }] — будь-яка кількість. amount у рядку необов’язковий: якщо не передано (або ≤ 0) — береться залишок замовлення. Загальна сума очікуваної оплати рахується автоматично як сума всіх amount.

Поле	Тип	Обов.	Де	Опис
`payment_method_id`	integer	так	body	ID способу оплати (GET /payment/methods) — тип платежу визначається ним
`transaction_number`	string	так	body	ID транзакції
`organization_id`	numeric	так	body	ID організації
`orders`	array	так	body	Масив замовлень (мін. 1)
`orders.*.order_id`	integer	так	body	ID замовлення
`orders.*.amount`	numeric	—	body	Сума на це замовлення (>0). Якщо не передано — береться залишок
`account_id`	integer	так	body	ID рахунку
`commission`	numeric	—	body	Комісія (за замовч. 0)
`info`	string	—	body	Примітка

Тіло запиту

```
{
    "payment_method_id": 5,
    "transaction_number": "tx_pending_1",
    "organization_id": 1,
    "account_id": 7,
    "orders": [
        {
            "order_id": 12345,
```

```
            "amount": 999
        },
        {
            "order_id": 12346
        }
    ]
}
```

Приклад відповіді

```
{
    "message": "Payment created successfully",
    "payment_id": 123
}
```

GET `https://[ВАШ САБДОМЕН]-api.marchroute.com/api/v1/payment/pending/{transaction_number}/orders`

Замовлення за очікуваною оплатою scope: payments:manage

Поле	Тип	Обов.	Де	Опис
`transaction_number`	string	так	path	transaction_number pending

Приклад відповіді

```
{
    "orders": [
        101,
        102,
        103
    ]
}
```

- 404: { "error": "Pending payment not found" }.

GET `https://[ВАШ САБДОМЕН]-api.marchroute.com/api/v1/payment/pending/organization`

Організація за очікуваною оплатою scope: payments:manage

Поле	Тип	Обов.	Де	Опис
`transaction_number`	string	так	query	transaction_number pending

Приклад відповіді

```
{
    "organization": {
        "id": 1,
        "name": "ТОВ ..."
    },
    "account": {
        "id": 7,
        "iban": "UA..."
    },
    "credentials": {
        "token": "****"
    }
}
```

POST `https://[ВАШ САБДОМЕН]-api.marchroute.com/api/v1/payment/pending/confirm`

Підтвердити очікувану оплату scope: payments:manage

Підтверджує pending за transaction_number, перетворюючи на реальну оплату.

Поле	Тип	Обов.	Де	Опис

`transaction_number`	string	так	body	ID транзакції pending
`commission`	numeric	—	body	Перекриває комісію pending

Приклад відповіді

```
{
    "message": "Payment confirmed successfully"
}
```

POST `https://[ВАШ САБДОМЕН]-api.marchroute.com/api/v1/order/{id}/checks/fiskalize`

Фіскалізувати замовлення scope: payments:manage

Пробиває фіскальний чек по замовленню за поточними політиками модуля фіскалізації (передоплата, післяплата, залишок нефіскалізованої суми). Дія аналогічна кнопці «Фіскалізувати» в CRM: чек створюється одразу, відкладений перевипуск (якщо був) скасовується.

Поле	Тип	Обов.	Де	Опис
`id`	integer	так	path	ID замовлення

Приклад відповіді

```
{
    "success": "checkCreated",
    "checks": [
        {
            "id": 1,
            "ref": "abc-123",
            "amount": 1500,
            "view_url": "https://check.checkbox.ua/...",
            "return": 0,
            "return_check": 0
        }
    ]
}
```

- checks — актуальні (несторновані) чеки замовлення після фіскалізації.
- 200 { "success": "skipped" } — фіскалізація пропущена політикою (напр., часткова оплата при вимкненому чеку передоплати).
- 200 { "error": "Order is fiskalized" } — замовлення вже повністю фіскалізоване.
- 422 — модуль фіскалізації не активний; 503 — касова зміна не відкрита; 429 — повторіть пізніше.

POST `https://[ВАШ САБДОМЕН]-api.marchroute.com/api/v1/order/{id}/checks/return`

Повернути (сторнувати) чеки замовлення scope: payments:manage

Сторнує всі актуальні чеки замовлення (чек повернення в ПРРО). Чеки, додані вручну або без модуля фіскалізації, пропускаються і повертаються у полі skipped.

Поле	Тип	Обов.	Де	Опис
`id`	integer	так	path	ID замовлення

Приклад відповіді

```
{
    "success": "checksReturned",
    "skipped": [
        {
            "check_id": 5,
            "ref": "abc-123",
            "reason": "вручну додані чеки не підлягають автоматичному поверненню"
        }
    ]
}
```

- skipped присутній лише якщо частину чеків не вдалося сторнувати автоматично.
- 503 — касова зміна не відкрита (повторіть після відкриття зміни).

DELETE `https://[ВАШ САБДОМЕН]-api.marchroute.com/api/v1/payment/pending/{transaction_number}`

Відв’язати замовлення від очікуваної оплати scope: payments:manage

Відв’язує всі замовлення від очікуваної оплати (сам запис оплати не видаляється).

Поле	Тип	Обов.	Де	Опис
`transaction_number`	string	так	path	ID транзакції очікуваної оплати

Приклад відповіді

```
{
    "message": "Pending payment removed successfully"
}
```

GET `https://[ВАШ САБДОМЕН]-api.marchroute.com/api/v1/orders/organization`

Організація/рахунок за замовленнями scope: payments:manage

Визначає організацію/рахунок/реквізити для набору замовлень за обраним способом оплати. Параметри — у query.

Поле	Тип	Обов.	Де	Опис
`order_ids`	array	так	query	ID замовлень
`payment_method_id`	integer	так	query	ID способу оплати (визначає організацію та рахунок)

Приклад відповіді

```
{
    "organization": {
        "id": 1,
        "name": "ТОВ ..."
    },
    "account": {
        "id": 7,
        "iban": "UA...",
        "name": "mono"
    },
    "credentials": {
        "token": "****"
    }
}
```

- 404: { "error": "No order IDs provided" } або { "error": "No organization found for the provided order IDs" }.

GET `https://[ВАШ САБДОМЕН]-api.marchroute.com/api/v1/organizations/{id}`

Організація за ID scope: payments:manage

Поле	Тип	Обов.	Де	Опис
`id`	integer	так	path	ID організації

Приклад відповіді

```
{
    "organization": {
        "id": 1,
        "name": "ТОВ ...",
        "accounts": []
    }
}
```

- 404: { "error": "Organization not found" }.

Товари

Правила спільні для BCIX розділів Export API: доступ лише на читання (scope data:read); списки посторінкові (page, limit); майже до кожного списку є парний ендпоінт «.../filter» — він повертає доступні випадні фільтри та варіанти сортування, а обрані значення передаються у списковий ендпоінт query-параметром filter (масив {value, data:[{value}]}).

GET `https://[ВАШ САБДОМЕН]-api.marchroute.com/api/v1/export/products`

Список товарів (пагінація) scope: data:read

Виключає застарілі товари.

Поле	Тип	Обов.	Де	Опис
`search`	string	—	query	Частковий збіг за назвою або артикулом
`sort`	string	—	query	created_at|name|price|cost|sku _asc/_desc (за замовч. created_at_desc)
`page`	integer	—	query	Номер сторінки (за замовч. 1)
`limit`	integer	—	query	Записів на сторінку (за замовч. 15)
`dateRange`	array[2]	—	query	[from, to] — фільтр за датою
`filter`	array	—	query	Масив фільтрів {value, data:[{value}]}

Приклад відповіді


```
{
    "current_page": 1,
    "data": [
        {
            "id": 1,
            "inner_id": "1001",
            "name": "Стіл",
            "sku": "ST-1",
            "price": 1500,
            "cost": 900,
            "is_custom": false,
            "url_image": "https://...",
            "vendor": {
                "id": 3,
                "name": "..."
            },
            "categories": []
        }
    ],
    "per_page": 15,
    "total": 1,
    "last_page": 1,
    "from": 1,
    "to": 1,
    "oldestFilterDate": "2024-01-01T00:00:00Z"
}
```


GET `https://[ВАШ САБДОМЕН]-api.marchroute.com/api/v1/export/products/search/{value}`

Пошук товарів (до 100) scope: data:read

Активні товари (без застарілих і чернеток): частковий збіг за назвою або точний за артикулом. Повертає простий масив.

Поле	Тип	Обов.	Де	Опис
`value`	string	так	path	Пошуковий запит

Приклад відповіді

```
[
    {
        "id": 1,
        "name": "Стіл",
        "sku": "ST-1",
        "price": 1500,
        "url_image": "https://..."
    }
]
```


GET `https://[ВАШ САБДОМЕН]-api.marchroute.com/api/v1/export/products/filter`

Метадані фільтрів товарів scope: data:read

Опис випадних фільтрів для списку товарів і доступні варіанти сортування.

Приклад відповіді


```
{
    "vendors": {
        "value": "vendor",
        "label": "Постачальник",
        "data": []
    },
    "categories": {
        "value": "product_category",
        "label": "Категорія",
        "data": []
    },
    "productTypes": {
        "value": "type",
        "label": "Тип",
        "data": []
    },
    "sortOptions": []
}
```


Замовлення та чеки

GET `https://[ВАШ САБДОМЕН]-api.marchroute.com/api/v1/export/orders`

Усі замовлення (пагінація, пошук) scope: data:read

Включає архівні. search (за наявності) перекриває filter. Дефолтний limit — 50; oldestFilterDate у відповіді відсутній.

Поле	Тип	Обов.	Де	Опис
`sort`	string	—	query	date_asc(за замовч.)|date_desc|amount_asc|amount_desc
`search`	string	—	query	Текстовий пошук (номер/телефон/ПІБ/ТТН)
`person_id`	integer	—	query	Фільтр за клієнтом
`page`	integer	—	query	Номер сторінки (за замовч. 1)
`limit`	integer	—	query	Записів на сторінку (за замовч. 50)
`dateRange`	array[2]	—	query	[from, to] — фільтр за датою
`filter`	array	—	query	Масив фільтрів {value, data:[{value}]}

Приклад відповіді


```
{
    "data": [
        {
            "id": 12345,
            "number": "250001",
            "total_amount": 1499,
```

```
            "total_remaining": 1499,
            "current_state": {
                "state": "green"
            },
            "person": [],
            "products": []
        }
    ],
    "current_page": 1,
    "per_page": 50,
    "total": 1,
    "last_page": 1,
    "from": 1,
    "to": 1,
    "first_page_url": "...",
    "next_page_url": null,
    "prev_page_url": null,
    "path": "...",
    "all_ids": []
}
```


- all_ids заповнюється лише за наявності права на аналітику, інакше []. oldestFilterDate у цій відповіді відсутній (на відміну від /checks, /payments, /realizations).

GET `https://[ВАШ САБДОМЕН]-api.marchroute.com/api/v1/export/orders/search`

Швидкий пошук (до 10) scope: data:read

Поле	Тип	Обов.	Де	Опис
`request`	string	—	query	Запит (потрібно ≥3 символів); порожній → []

Приклад відповіді


```
[
    {
        "id": 12345,
        "number": "250001",
        "current_state": {
            "state": "green"
        }
    }
]
```


GET `https://[ВАШ САБДОМЕН]-api.marchroute.com/api/v1/export/orders/{number}`

Замовлення за номером (повне) scope: data:read

Замовлення з усіма зв’язками, обчисленим статусом і підсумками клієнта.

Поле	Тип	Обов.	Де	Опис
`number`	string	так	path	Номер замовлення

Приклад відповіді


```
{
    "id": 12345,
    "number": "250001",
    "status": "green",
    "custom_fields": {
        "promo_code": "SPRING10"
    },
    "recipient": [],
    "total_orders_person": 7,
    "total_orders_person_amount": 10493
}
```


- 404: { "error": "Order not found" }.

- custom_fields — кастомні поля замовлення {code: value} (довідник — блок «Кастомні поля» в картці замовлення); порожньо — {}.

GET `https://[ВАШ САБДОМЕН]-api.marchroute.com/api/v1/export/orders/filter`

Метадані фільтрів замовлень scope: data:read

Опис випадних фільтрів для списку замовлень. UTM/джерело — лише за наявності права на аналітику.

Приклад відповіді

```
{
    "oldestFilterDate": "2024-01-01T00:00:00Z",
    "statuses": {
        "value": "status",
        "label": "Статус",
        "data": []
    },
    "organizations": {
        "value": "organization",
        "label": "Організація",
        "data": []
    }
}
```

GET `https://[ВАШ САБДОМЕН]-api.marchroute.com/api/v1/export/checks`

Чеки (пагінація) scope: data:read

Поле	Тип	Обов.	Де	Опис
`page`	integer	—	query	Номер сторінки (за замовч. 1)
`limit`	integer	—	query	Записів на сторінку (за замовч. 15)
`dateRange`	array[2]	—	query	[from, to] — фільтр за датою
`filter`	array	—	query	Масив фільтрів {value, data:[{value}]}

Приклад відповіді

```
{
    "current_page": 1,
    "data": [
        {
            "id": 1,
            "ref": "abc",
            "amount": 1499,
            "order": [],
            "organization": {
                "id": 1,
                "name": "..."
            },
            "created_at": "..."
        }
    ],
    "per_page": 15,
    "total": 1,
    "last_page": 1,
    "from": 1,
    "to": 1,
    "oldestFilterDate": "2024-01-01T00:00:00Z"
}
```

GET `https://[ВАШ САБДОМЕН]-api.marchroute.com/api/v1/export/checks/filter`

Метадані фільтрів чеків scope: data:read

Приклад відповіді

```
{
    "oldestFilterDate": "2024-01-01T00:00:00Z",
    "orders": {
        "value": "order",
        "label": "Замовлення",
        "custom": true,
        "data": []
    },
    "organizations": {
        "value": "organization",
        "label": "Організація",
        "data": []
    }
}
```


Оплати

GET `https://[ВАШ САБДОМЕН]-api.marchroute.com/api/v1/export/payments`

Список оплат (пагінація) scope: data:read

Виключає типи return і post. Додаткові поля у відповіді: remaining_amount, type_name, is_auto, method_type.

Поле	Тип	Обов.	Де	Опис
`page`	integer	—	query	Номер сторінки (за замовч. 1)
`limit`	integer	—	query	Записів на сторінку (за замовч. 15)
`dateRange`	array[2]	—	query	[from, to] — фільтр за датою
`filter`	array	—	query	Масив фільтрів {value, data:[{value}]}

Приклад відповіді


```
{
    "current_page": 1,
    "data": [
        {
            "id": 123,
            "amount": "1000.00",
            "type": "card",
            "transaction_number": "card_1716",
            "remaining_amount": 1000,
            "type_name": "Картка",
            "organization": {
                "id": 1,
                "name": "..."
            },
            "orders": [],
            "account": []
        }
    ],
    "per_page": 15,
    "total": 1,
    "last_page": 1,
    "from": 1,
    "to": 1,
    "oldestFilterDate": "2024-01-01T00:00:00Z"
}
```


GET `https://[ВАШ САБДОМЕН]-api.marchroute.com/api/v1/export/payments/filter`

Метадані фільтрів оплат scope: data:read

Приклад відповіді

```
{
    "oldestFilterDate": "2024-01-01T00:00:00Z",
    "organizations": {
        "value": "organization",
        "label": "Організація",
        "data": []
    },
    "types": {
        "value": "type",
        "label": "Рахунок",
        "data": []
    },
    "purposes": {
        "value": "purpose",
        "label": "Призначення",
        "custom": true,
        "data": []
    },
    "payers": {
        "value": "payer",
        "label": "Платник",
        "custom": true,
        "data": []
    }
}
```


GET `https://[ВАШ САБДОМЕН]-api.marchroute.com/api/v1/export/payments/total`

Сума оплат scope: data:read

SUM(amount), виключаючи return/post, з урахуванням фільтрів.

Поле	Тип	Обов.	Де	Опис
`dateRange`	array[2]	—	query	[from, to]
`filter`	array	—	query	Фільтри

Приклад відповіді


```
{
    "total_amount": 123456.77999999999883584678173065185546875
}
```


Реалізації, накладні, чати

GET `https://[ВАШ САБДОМЕН]-api.marchroute.com/api/v1/export/realizations`

Реалізації (пагінація) scope: data:read

Записи реалізацій із supplier, warehouse і глибоко вкладеним order (checks, products, images).

Поле	Тип	Обов.	Де	Опис
`page`	integer	—	query	Номер сторінки (за замовч. 1)
`limit`	integer	—	query	Записів на сторінку (за замовч. 15)
`dateRange`	array[2]	—	query	[from, to] — фільтр за датою
`filter`	array	—	query	Масив фільтрів {value, data:[{value}]}

Приклад відповіді


```
{
    "current_page": 1,
    "data": [
        {
            "id": 88,
```

```
            "supplier_id": 5,
            "warehouse_id": 1,
            "supplier": {
                "id": 5,
                "name": "..."
            },
            "warehouse": {
                "id": 1,
                "name": "..."
            },
            "order": {
                "id": 99021,
                "number": "250012",
                "total_amount": 4500
            }
        }
    ],
    "per_page": 15,
    "total": 1,
    "last_page": 1,
    "from": 1,
    "to": 1,
    "oldestFilterDate": "2024-01-01T00:00:00Z"
}
```


GET `https://[ВАШ САБДОМЕН]-api.marchroute.com/api/v1/export/realizations/total`

Підсумок реалізацій scope: data:read

Поле	Тип	Обов.	Де	Опис
`dateRange`	array[2]	—	query	[from, to]
`filter`	array	—	query	Фільтри

Приклад відповіді


```
{
    "total_amount": 1250000,
    "count": 342
}
```


GET `https://[ВАШ САБДОМЕН]-api.marchroute.com/api/v1/export/realizations/filter`

Метадані фільтрів реалізацій scope: data:read

Приклад відповіді


```
{
    "suppliers": {
        "value": "supplier",
        "label": "Постачальник",
        "data": []
    },
    "warehouses": {
        "value": "warehouse",
        "label": "Склад",
        "data": []
    },
    "oldestFilterDate": "2024-01-01T00:00:00Z"
}
```


GET `https://[ВАШ САБДОМЕН]-api.marchroute.com/api/v1/export/supplier-invoices`

Накладні постачальників (пагінація) scope: data:read

Сортування: спершу термінові (is_asap) та невиконані (is_done=0), далі найновіші. Зі зв’язками: організація, замовлення, постачальник, файли, рахунок.

Поле	Тип	Обов.	Де	Опис
`page`	integer	—	query	Номер сторінки (за замовч. 1)
`limit`	integer	—	query	Записів на сторінку (за замовч. 15)
`dateRange`	array[2]	—	query	[from, to] — фільтр за датою
`filter`	array	—	query	Масив фільтрів {value, data:[{value}]}

Приклад відповіді


```
{
    "current_page": 1,
    "data": [
        {
            "id": 720,
            "organization_id": 3,
            "supplier_id": 5,
            "amount": 4500,
            "number": "INV-1023",
            "type": "accounts",
            "is_asap": 1,
            "is_done": 0,
            "organization": {
                "id": 3,
                "name": "..."
            },
            "orders": [],
            "supplier": {
                "id": 5,
                "name": "..."
            }
        }
    ],
    "per_page": 15,
    "total": 1,
    "last_page": 1,
    "from": 1,
    "to": 1,
    "oldestFilterDate": "2024-01-01T00:00:00Z"
}
```


GET `https://[ВАШ САБДОМЕН]-api.marchroute.com/api/v1/export/supplier-invoices/filter`

Метадані фільтрів накладних scope: data:read

Приклад відповіді


```
{
    "oldestFilterDate": "2024-01-01T00:00:00Z",
    "organizations": {
        "value": "organization",
        "label": "Організація",
        "data": []
    },
    "suppliers": {
        "value": "supplier",
        "label": "Постачальник",
        "data": []
    },
    "orders": {
        "value": "order",
        "label": "Замовлення",
        "custom": true,
        "data": []
    }
}
```

GET `https://[ВАШ САБДОМЕН]-api.marchroute.com/api/v1/export/chats`

Чати (пагінація, модуль channels) scope: data:read

Список чатів із зовнішніми ідентифікаторами, закріпленими повідомленнями, контактом і відповідальним. Без oldestFilterDate.

Поле	Тип	Обов.	Де	Опис
`page`	integer	—	query	Сторінка (за замовч. 1)
`limit`	integer	—	query	На сторінку (за замовч. 15)

Приклад відповіді


```
{
    "current_page": 1,
    "data": [
        {
            "id": 5012,
            "channel_id": 4,
            "unread": 2,
            "external_id": "380501112233",
            "type": "telegram",
            "driver": "telegram",
            "messages_count": 47,
            "contact": {
                "id": 880,
                "name": "Іван"
            }
        }
    ],
    "per_page": 15,
    "total": 1
}
```


GET `https://[ВАШ САБДОМЕН]-api.marchroute.com/api/v1/export/chats/{id}/messages`

Повідомлення чату (read-only) scope: data:read

Повідомлення без позначення чату прочитаним. message_id центрує вікно (~25 до + ціль + 25 після).

Поле	Тип	Обов.	Де	Опис
`id`	integer	так	path	ID чату (404, якщо не знайдено)
`thread_id`	integer	—	query	Фільтр за гілкою
`search`	string	—	query	Частковий збіг за текстом повідомлення
`notes_only`	boolean	—	query	Лише нотатки (is_note=1)
`date`	string	—	query	Фільтр/якір за датою
`page`	integer	—	query	Сторінка (за замовч. 1)
`limit`	integer	—	query	На сторінку (за замовч. 50)
`message_id`	integer	—	query	Центрувати вікно навколо повідомлення

Приклад відповіді


```
{
    "messages": [
        {
            "id": 91002,
            "chat_id": 5012,
            "content": "Доброго дня!",
            "direction": "in",
            "is_note": false,
            "model": "message",
            "files": []
        }
```

```
    ],
    "has_more": false,
    "chat": {
        "id": 5012,
        "external_id": "380501112233",
        "type": "telegram"
    }
}
```

GET `https://[ВАШ САБДОМЕН]-api.marchroute.com/api/v1/export/chats/{id}/info`

Інформація про чат (модуль channels) scope: data:read

Повний об’єкт чату з усіма зв’язками (контакт, канал, остання активність). 404, якщо не знайдено.

Поле	Тип	Обов.	Де	Опис
`id`	integer	так	path	ID чату

Приклад відповіді

```
{
    "id": 5012,
    "channel_id": 4,
    "unread": 2,
    "external_id": "380501112233",
    "type": "telegram",
    "driver": "telegram",
    "messages_count": 47,
    "contact": {
        "id": 880,
        "name": "Іван"
    },
    "channel": [],
    "lastActivity": "2026-05-21T10:00:00Z"
}
```

Аналітика

GET `https://[ВАШ САБДОМЕН]-api.marchroute.com/api/v1/export/analytics/{tab}/{type}`

Аналітичний звіт scope: data:read

Звіт за комбінацією tab/type. Доступні: general/(list|territory), products/(list|categories), orders/(statuses|checkpoints|realizations|cancelled), utms/general, managers/(activity|efficiency|origin). Невідома комбінація → 400.

Поле	Тип	Обов.	Де	Опис
`tab`	string	так	path	general | products | orders | utms | managers
`type`	string	так	path	list/territory/categories/statuses/... (залежно від tab)
`dateRange`	array[2]	—	query	[from, to]
`filter`	array	—	query	Масив фільтрів (Interval, reports, utm_type, attribution тощо)
`groupBy`	string	—	query	Групування (за замовч. region) — для general/territory
`sort`	string	—	query	Сортування
`utm`	string	—	query	Поле UTM (за замовч. utm_campaign) — для utms

Приклад відповіді

```
{
    "data": []
}
```

- 400: { "error": "Invalid analytics type" } — для невідомої комбінації tab/type.

GET `https://[ВАШ САБДОМЕН]-api.marchroute.com/api/v1/export/analytics/{tab}/{type}/filter`

Метадані фільтрів аналітики scope: data:read

Поле	Тип	Обов.	Де	Опис
`tab`	string	так	path	general | products | orders | utms | managers
`type`	string	так	path	Підтип звіту

Webhook та підпис HMAC

CRM надсилає **вихідні** запити (`POST`) на **URL магазину** (вкладка «Webhook»). Тип події передається у заголовку `MR-Endpoint` , а тіло — JSON. Кожен запит підписується секретом **HMAC-SHA256**, щоб ваш сервер міг переконатися, що запит надійшов саме від CRM і не був змінений.

Секрет генерується кнопкою у вкладці «Webhook» і показується лише один раз — зберігайте його як пароль.

Заголовки запиту

Заголовок	Опис
`MR-Endpoint`	Тип події (напр. payment_confirmed). За ним маршрутизуйте обробку
`Signature`	HMAC-SHA256 від канонічного payload (hex)
`Timestamp`	Unix-час (с) формування запиту. Вікно валідності — 300 с
`Client-Id`	Ідентифікатор клієнта CRM
`Content-Hash`	SHA-256 від тіла запиту (hex)

Як формується підпис

Підписується не саме тіло, а **канонічний payload** — рядок із 5 частин, розділених `\n` :

payload

```
METHOD            ← напр. POST (у верхньому регістрі)
RESOURCE_PATH     ← шлях URL магазину без хосту
CLIENT_ID         ← ваш Client-Id
TIMESTAMP         ← те саме значення, що в заголовку Timestamp
BODY_HASH         ← sha256(тіло запиту), hex
```

Далі `signature = hash_hmac('sha256', payload, secret)` .

Важливо про тіло. Тіло серіалізується канонічно: асоціативні ключі рекурсивно сортуються (`ksort`), порядок елементів списків зберігається, JSON кодується з прапорами `JSON_UNESCAPED_UNICODE` , `JSON_UNESCAPED_SLASHES` , `JSON_PRESERVE_ZERO_FRACTION` . На прийомі підписуйте **сирий байтовий рядок** тіла — не перепарсюйте JSON, інакше підпис «попливе».

Події (MR-Endpoint)

Значення заголовка `MR-Endpoint` та приклади тіл, які CRM надсилає магазину:

POST `MR-Endpoint: order`

Замовлення (основні поля)

Коли надсилається: Коли замовлення переходить на чекпоінт, в якому вказана відповідна функція при вході. Надсилає актуальний стан замовлення.

custom_fields — кастомні поля замовлення {code: value} з довідника CRM (блок «Кастомні поля» в картці замовлення; задаються там або через POST /order / PUT /order/{id}); якщо жодне не заповнене — {}.

Тіло запиту

```
{
    "order_id": 10234,
    "status_id": 3,
    "checkpoint_id": 7,
    "number": "250001",
    "base_number": "250001",
    "state": "green",
    "total_amount": 1499,
    "total_cost": 980,
    "total_commission": 0,
    "person_id": 12,
    "organization_id": 1,
    "supplier_id": 5,
    "delivery_id": 2,
    "warehouse_id": 1,
    "done_at": null,
    "created_at": "2026-05-21T10:00:00+00:00",
    "updated_at": "2026-05-21T10:00:00+00:00",
    "custom_fields": {
        "promo_code": "SPRING10"
    }
}
```


POST `MR-Endpoint: payment_confirmed`

Оплату підтверджено

Коли надсилається: Коли в замовленні з’являється підтверджений платіж (клієнт оплатив, або менеджер підтвердив оплату вручну). Надсилається разом зі списком замовлень, яких стосується платіж.

Тіло запиту


```
{
    "payment_id": 777,
    "transaction_number": "card_1716200000",
    "amount": 1499,
    "commission": 0,
    "organization_id": 1,
    "payment_method_id": 5,
    "orders": [
        {
            "id": 10234,
            "number": "ORD-10234",
            "amount": 1499
        }
    ],
    "created_at": "2026-05-21T10:00:00+00:00"
}
```


POST `MR-Endpoint: check`

Фіскальний чек створено

Коли надсилається: Після кожного створеного фіскального чека по замовленню (фіскалізація або ручний чек). Надсилає реф та урл чека.

У кожного чека є короткий redirect_code — використовуйте його як скорочувач посилань: зробіть на своєму сайті сторінку, відправляйте клієнта на неї з цим кодом і обмінюйте його на реальні посилання запитом GET /checks/redirect/{redirect_code} — він поверне всі пов’язані чеки замовлення, а з параметром strong=true — лише цей конкретний чек.

Тіло запиту


```
{
    "id": 5012,
    "ref": "a1b2c3d4-...",
    "redirect_code": "c3d4",
    "amount": 1499,
    "order_id": 10234,
```

```
    "order_number": "250001",
    "organization_id": 1,
    "fiskalization": "checkbox",
    "is_return": false,
    "created_at": "2026-05-21T10:00:00+00:00"
}
```

POST `MR-Endpoint: confirm_payment`

Підтвердження очікуваної оплати

Коли надсилається: Коли підтверджується очікувана оплата за способом, який обробляє ваш магазин (магазин сам проводить транзакцію — callback-режим). CRM просить магазин завершити підтвердження на своєму боці.

Тіло запиту

```
{
    "transaction_number": "card_1716200000",
    "type": "card",
    "organization_id": 1,
    "account": "mono",
    "order": [
        101,
        102
    ]
}
```

POST `MR-Endpoint: return_payment`

Повернення коштів

Коли надсилається: Коли по замовленню оформлюється повернення коштів за способом оплати, який обробляє ваш магазин (callback-режим). Магазин має провести повернення на своєму боці.

Тіло запиту

```
{
    "transaction_number": "card_1716200000",
    "amount": 500,
    "organization_id": 1,
    "payment_method_id": 5
}
```

POST `MR-Endpoint: create_payment`

Створити платіж (callback-режим)

Коли надсилається: Коли спосіб оплати працює в режимі «Колбек»: CRM надсилає запит у ваш магазин, а магазин обробляє платіж і повертає режим відповіді. Режим «link» — повертається посилання на оплату; режим «default» — повертається лише текст, який показується оператору (напр. «Клієнт повинен підтвердити в застосунку»).

Тіло запиту

```
{
    "amount": 1499,
    "payment_method_id": 5,
    "orders": [
        {
            "order_id": 10234,
            "amount": 999
        },
        {
            "order_id": 10235,
            "amount": 500
        }
    ],
    "max_month": 5,
    "additional_info": "Останні цифри картки"
```

```
}
```

Очікувана відповідь магазину

```
{
    "data": {
        "mode": "link",
        "link": "https://pay...",
        "text": "Клієнт повинен підтвердити в застосунку",
        "transaction_number": "tx_abc123"
    }
}
```

POST `MR-Endpoint: warehouse`

Залишки складу

Коли надсилається: Кожну годину CRM розраховує доступні залишки й передає їх, згруповані за виробником.

Тіло запиту

```
{
    "grouped": {
        "Виробник": [
            {
                "inner_id": "1001",
                "sku": "ST-1",
                "quantity": 12
            },
            {
                "inner_id": "1002",
                "sku": "ST-2",
                "quantity": 0
            }
        ]
    }
}
```

POST `MR-Endpoint: call`

Дані дзвінка (CID/IP) для атрибуції

Коли надсилається: Коли по замовленню застосовується колтрекінг: CRM запитує у магазину дані про джерело/UTM за ідентифікатором сесії (cid) та IP.

Тіло запиту

```
{
    "cid": "GA1.2.1234567890.1716200000",
    "ip": "203.0.113.10"
}
```

Очікувана відповідь магазину

```
{
    "data": {
        "data": {
            "id": 1,
            "cid": "GA1.2...",
            "utm_source": "google",
            "utm_medium": "cpc",
            "utm_campaign": "spring",
            "gclid": "Cj0KCQ..."
        }
    }
}
```

Перевірка підпису — PHP

```php
<?php

function verifyWebhook(string $secret, string $clientId): bool
{
    $signature   = $_SERVER['HTTP_SIGNATURE']    ?? '';
    $timestamp   = $_SERVER['HTTP_TIMESTAMP']     ?? '';
    $bodyHashHdr = $_SERVER['HTTP_CONTENT_HASH']  ?? '';
    $receivedId  = $_SERVER['HTTP_CLIENT_ID']     ?? '';
    $endpoint    = $_SERVER['HTTP_MR_ENDPOINT']?? ''; // яка це подія

    if (!$signature || !$timestamp) return false;

    // 1. Захист від replay: вікно 300 секунд
    if (abs(time() - (int) $timestamp) > 300) return false;

    // 2. Перевірка client id
    if (!hash_equals($clientId, $receivedId)) return false;

    // 3. Хеш тіла — підписується рівно той байтовий рядок, що прийшов
    $body = file_get_contents('php://input');
    $bodyHash = hash('sha256', $body);
    if ($bodyHashHdr && !hash_equals($bodyHash, $bodyHashHdr)) return false;

    // 4. Збираємо payload і звіряємо підпис
    $path = parse_url($_SERVER['REQUEST_URI'], PHP_URL_PATH);
    $payload = strtoupper($_SERVER['REQUEST_METHOD']) . "\n"
        . $path . "\n"
        . $clientId . "\n"
        . $timestamp . "\n"
        . $bodyHash;

    $expected = hash_hmac('sha256', $payload, $secret);

    return hash_equals($expected, $signature);
    // далі: switch ($endpoint) { case 'payment_confirmed': ... }
}
```

Перевірка підпису — Node.js

```node
const crypto = require('crypto')

// Express. Тіло має зчитуватися як RAW (express.raw / verify-hook),
// бо підписується саме байтовий рядок.
function verifyWebhook(secret, clientId) {
  return (req, res, next) => {
    const signature  = req.get('Signature')
    const timestamp  = req.get('Timestamp')
    const bodyHashHdr = req.get('Content-Hash')
    const receivedId = req.get('Client-Id')
    const endpoint   = req.get('MR-Endpoint')  // подія

    if (!signature || !timestamp) return res.sendStatus(401)
    if (Math.abs(Date.now() / 1000 - Number(timestamp)) > 300) return res.sendStatus(401)
    if (receivedId !== clientId) return res.sendStatus(401)

    const body = req.rawBody.toString('utf8')
    const bodyHash = crypto.createHash('sha256').update(body).digest('hex')
    if (bodyHashHdr && bodyHash !== bodyHashHdr) return res.sendStatus(401)

    const payload = [req.method.toUpperCase(), req.path, clientId, timestamp, bodyHash].join('\n')
    const expected = crypto.createHmac('sha256', secret).update(payload).digest('hex')

    const ok = crypto.timingSafeEqual(Buffer.from(expected), Buffer.from(signature))
    if (!ok) return res.sendStatus(401)

    req.pulseEvent = endpoint
```

```
    return next()
  }
}
```

Перевірка підпису — Python

```
import hmac, hashlib, time
from urllib.parse import urlparse

def verify_webhook(secret: str, client_id: str, request) -> bool:
    signature   = request.headers.get('Signature', '')
    timestamp   = request.headers.get('Timestamp', '')
    body_hash_h = request.headers.get('Content-Hash', '')
    received_id = request.headers.get('Client-Id', '')
    endpoint    = request.headers.get('MR-Endpoint', '')  # подія

    if not signature or not timestamp:
        return False
    if abs(time.time() - int(timestamp)) > 300:
        return False
    if not hmac.compare_digest(client_id, received_id):
        return False

    body = request.get_data()  # bytes — НЕ перепарсюйте JSON
    body_hash = hashlib.sha256(body).hexdigest()
    if body_hash_h and not hmac.compare_digest(body_hash, body_hash_h):
        return False

    path = urlparse(request.url).path
    payload = "\n".join([request.method.upper(), path, client_id, timestamp, body_hash])
    expected = hmac.new(secret.encode(), payload.encode(), hashlib.sha256).hexdigest()
    return hmac.compare_digest(expected, signature)
```

Поширені помилки

- Підписали перепарсений JSON замість сирого тіла — хеш не збігається.
- Невідповідність часу сервера > 300 с — запит відхиляється як replay.
- Порівняння підпису через `==` замість constant-time (`hash_equals` / `timingSafeEqual` / `compare_digest`).
- У payload потрапив повний URL із хостом замість шляху.