MeDriouX - Schnittstelle (API) v1.0

Stand: 06.10.2026 - die jeweils aktuelle Fassung liegt unter /hilfe/api

Mit der Schnittstelle (API) kannst du deinen Shop automatisch verwalten: Artikel anlegen und pflegen, Bestellungen abrufen und bearbeiten, Rechnungen erstellen - zum Beispiel aus deiner Warenwirtschaft, einem eigenen Programm oder einem Automatisierungs-Tool wie Zapier oder Make.

Alle Anfragen und Antworten nutzen JSON. Die Feldnamen sind deutsch und entsprechen genau dem, was du in der Plattform siehst.

Anmeldung & Schlüssel

Jede Anfrage braucht deinen persönlichen API-Schlüssel im Kopf (Header) der Anfrage:

```
X-API-Key: dein-schluessel
```

Deinen Schlüssel erzeugst und verwaltest du in der Plattform unter Menü -> Konto -> Meine Daten -> API-Zugang. Behandle ihn wie ein Passwort: Wer den Schlüssel hat, kann in deinem Namen arbeiten. Bei Verdacht auf Missbrauch einfach einen neuen erzeugen - der alte wird damit sofort ungültig.

Du siehst und bearbeitest über die API ausschließlich deine eigenen Websites und Daten. Erlaubt sind bis zu 1000 Anfragen pro Stunde.

Fehler & Status-Codes

Fehler kommen immer als JSON mit einer verständlichen Meldung:

```
{"ok": false, "fehler": "Artikel 123 nicht gefunden"}
```

Die wichtigsten Status-Codes: 200 (alles gut), 201 (angelegt), 400 (Eingabe fehlerhaft), 401 (Schlüssel fehlt oder ist falsch), 403 (gehört dir nicht), 404 (gibt es nicht), 429 (zu viele Anfragen - kurz warten).

Die Objekte und ihre Felder

Artikel

`id (Zahl)`
Eindeutige Nummer (vergibt das System)

`website_id (Zahl)`
Zu welcher Website der Artikel gehört

`name (Text)`
Artikelname (Pflicht beim Anlegen)

`beschreibung (Text)`
Kurzer Text unter dem Namen

`preis (Zahl)`
Preis in Euro, z. B. 19.9 - 0 bedeutet "auf Anfrage"

`einheit (Text)`
Zusatz vor dem Preis, z. B. "ab" oder "pro Std."

`kategorie (Text)`
Freie Kategorie (alt) - gruppiert Artikel ohne Katalog

`katalog (Text)`
Name des zugeordneten Katalogs (Warengruppe; nur lesbar - Zuordnung über die Oberfläche)

`streichpreis (Zahl)`

"Vorher"-Preis (UVP) - ist er höher als der Preis, zeigt der Shop einen Sale an

sku (Text)
Artikelnummer

marke (Text)
Marke / Hersteller

tags (Text)
Schlagworte, durch Komma getrennt - werden von der Shop-Suche gefunden

lagerbestand (Zahl)
Stück auf Lager; leer/null = Bestand wird nicht verfolgt. Bei 0 zeigt der Shop "ausverkauft", Bestellungen buchen automatisch ab

bild_url (Text)
Bild-Adresse des Hauptbilds (https://...)

bilder (Liste)
Weitere Bild-Adressen für die Galerie auf der Artikel-Seite

ust_satz (Zahl)
Umsatzsteuersatz in Prozent (19, 7 oder 0) - steuert den USt-Ausweis auf Rechnungen

aktiv (Ja/Nein)
false = im Shop unsichtbar

sortierung (Zahl)
Kleinere Zahlen stehen weiter oben

Bestellung

id (Zahl)
Bestellnummer

website_id (Zahl)
Website, auf der bestellt wurde

name (Text)
Name des Käufers

firma (Text)
Firma des Käufers (optional)

kaeufer_id (Zahl)
Kundenkonto des Käufers auf der Website (leer bei Gast-Bestellung)

email (Text)
E-Mail des Käufers

telefon (Text)
Telefon des Käufers

lieferart (Text)
abholung oder lieferung

adresse (Text)
Lieferadresse (bei Lieferung)

wunschtermin (Text)
Wunschtermin des Käufers

nachricht (Text)
Nachricht des Käufers

positionen (Liste)
Bestellte Artikel: id, name, preis, menge, ust (Steuersatz in %)

summe (Zahl)
Gesamtsumme in Euro (inklusive Versand, abzüglich Gutschein)

versand (Zahl)
Berechnete Versandkosten in Euro

rabatt (Zahl)
Abgezogener Gutschein-Betrag in Euro

gutschein_code (Text)
Eingelöster Gutschein-Code (leer = keiner)

sendungsnummer (Text)
Sendungsnummer - beim Setzen in der Verwaltung geht automatisch die Versand-E-Mail raus

zahlungsart (Text)
abholung, ueberweisung, paypal oder absprache

status (Text)

neu, in_arbeit, erledigt oder storniert

bezahlt (Ja/Nein)
Zahlungseingang markiert?

interne_notiz (Text)
Eure interne Notiz - sieht der Käufer nie

erstellt_am (Text)
Zeitpunkt der Bestellung (ISO-Format)

Rechnung

id (Zahl)
Interne Nummer

nummer (Text)
Rechnungsnummer, z. B. S9-0001

website_id (Zahl)
Zugehörige Website

bestellung_id (Zahl)
Zugehörige Bestellung (falls vorhanden)

kunde_name (Text)
Rechnungsempfänger

summe (Zahl)
Rechnungsbetrag in Euro

status (Text)
offen, bezahlt oder storniert

sevdesk_id (Text)
ID des sevDesk-Entwurfs (falls übergeben)

erstellt_am (Text)
Rechnungsdatum (ISO-Format)

Alle Endpunkte

Clearing Center

POST /api/v1/clearing/nachrichten

EDI-Datei einliefern

Liefert eine EDI-Datei ins Clearing Center ein (Kanal "api"). Die Nachrichtenart wird automatisch erkannt - bei EDIFACT verbindlich aus dem UNH-Segment. Nur für Konten, die als Clearing-Teilnehmer freigeschaltet sind.

? datei (Pflicht): Die Datei als multipart-Feld (XML, CSV, JSON, TXT oder EDIFACT)
? typ (optional): Nachrichtenart vorgeben (PRICAT, DESADV, INVOIC, SLSRPT, INVRPT ...)
? empfaenger_gln (optional): GLN des Empfängers; leer = global (bei PRICAT)

```
curl -X POST https://DEINE-PLATTFORM/api/v1/clearing/nachrichten \
  -H "X-API-Key: dein-schluessel" -F "datei=@katalog.edi"
```

```
Antwort: {"ok": true, "daten": {"id": 12, "typ": "PRICAT", "format": "edifact", "status": "verarbeitet"}}
```

Websites

GET /api/v1/websites

Eigene Websites auflisten

Liefert alle deine Websites mit Shop-Status. Die id brauchst du für alle weiteren Aufrufe.

```
Antwort: {"ok": true, "daten": [{"id": 9, "titel": "B&W Haarliebhaber", "slug": "b-w-haarliebhaber-f57612", "domain": "", "shop_aktiv": true, "shop_modus": "shop"}]}
```

Artikel

GET /api/v1/websites/{website_id}/artikel

Artikel einer Website auflisten

Alle Artikel, auch unsichtbare.

```
Antwort: {"ok": true, "daten": [{"id": 12, "name": "Herrenhaarschnitt", "preis": 24.9, "kategorie": "Herren", "aktiv": true, ...}]}
```

`POST /api/v1/websites/{website_id}/artikel`

Artikel anlegen

Legt einen neuen Artikel an. Nur name ist Pflicht.

? name (Pflicht): Artikelname
? preis (optional): Zahl in Euro, z. B. 19.9
? beschreibung / einheit / kategorie / bild_url (optional): wie im Objekt Artikel
? streichpreis / sku / marke / tags / lagerbestand (optional): wie im Objekt Artikel
? ust_satz / bilder / aktiv / sortierung (optional): wie im Objekt Artikel; bilder als Liste von Adressen
? aktiv (optional): true/false, Standard true
? sortierung (optional): Zahl, Standard 0

```
curl -X POST https://DEINE-PLATTFORM/api/v1/websites/9/artikel \
  -H "X-API-Key: dein-schluessel" -H "Content-Type: application/json" \
  -d '{"name": "Haarwachs", "preis": 12.9, "kategorie": "Produkte"}'
```

```
Antwort: {"ok": true, "daten": {"id": 31, "name": "Haarwachs", ...}}
```

`PATCH /api/v1/artikel/{artikel_id}`

Artikel ändern

Ändert nur die Felder, die du mitschickst - alles andere bleibt.

? beliebige Artikel-Felder (optional): z. B. nur {"preis": 14.5}

```
Antwort: {"ok": true, "daten": {"id": 31, "preis": 14.5, ...}}
```

`DELETE /api/v1/artikel/{artikel_id}`

Artikel löschen

Endgültig - zum bloßen Verstecken lieber PATCH mit {"aktiv": false}.

```
Antwort: {"ok": true}
```

Bestellungen

`GET /api/v1/websites/{website_id}/bestellungen`

Bestellungen auflisten

Neueste zuerst. Mit ?status=neu (oder in_arbeit, erledigt, storniert) filterst du.

? status (optional): Filter auf einen Status

```
curl https://DEINE-PLATTFORM/api/v1/websites/9/bestellungen?status=neu \
  -H "X-API-Key: dein-schluessel"
```

```
Antwort: {"ok": true, "daten": [{"id": 4, "name": "Max Käufer", "summe": 49.8, "status": "neu", "bezahlt": false, ...}]}
```

`GET /api/v1/bestellungen/{bestellung_id}`

Eine Bestellung abrufen

Alle Details inklusive Positionen.

```
Antwort: {"ok": true, "daten": {"id": 4, "positionen": [{"name": "Herrenhaarschnitt", "preis": 24.9, "menge": 2}], ...}}
```

`PATCH /api/v1/bestellungen/{bestellung_id}`

Bestellung bearbeiten

Status setzen, Zahlungseingang markieren oder die interne Notiz schreiben.

? status (optional): neu, in_arbeit, erledigt oder storniert

? bezahlt (optional): true/false
? interne_notiz (optional): Text, sieht nur ihr

```
curl -X PATCH https://DEINE-PLATTFORM/api/v1/bestellungen/4 \
  -H "X-API-Key: dein-schluessel" -H "Content-Type: application/json" \
  -d '{"status": "erledigt", "bezahlt": true}'
```

```
Antwort: {"ok": true, "daten": {"id": 4, "status": "erledigt", "bezahlt": true, ...}}
```

Rechnungen

`POST /api/v1/bestellungen/{bestellung_id}/rechnung`

Rechnung aus Bestellung erstellen

Erstellt die Rechnung an deinen Käufer. Gibt es schon eine, bekommst du die vorhandene zurück (es entsteht nie eine doppelte). Mit hinterlegtem sevDesk-Token wird sie automatisch als Entwurf an dein sevDesk übergeben.

```
Antwort: {"ok": true, "daten": {"id": 2, "nummer": "S9-0002", "summe": 49.8, "status": "offen", ...}}
```

`GET /api/v1/websites/{website_id}/rechnungen`

Rechnungen auflisten

Alle Rechnungen dieser Website, neueste zuerst.

```
Antwort: {"ok": true, "daten": [{"id": 2, "nummer": "S9-0002", "status": "offen", ...}]}
```