EsnafPazar

Satıcı REST API v1

Entegratör Geliştirici Dokümanı

Temel adres	https://www.esnafpazar.com/api/v1/
Biçim	JSON · UTF-8 · Bearer token
Sürüm	24.11.2026

Bu doküman EsnafPazar ile entegrasyon yapacak yazılım firmaları içindir.

İçindekiler

1. Giriş
2. Hızlı başlangıç
3. Kimlik doğrulama
4. Ortak kurallar
5. Uçlar
6. Sipariş ve iade durum kodları
7. Önerilen entegrasyon akışı
8. Sınırlamalar ve bilinen durumlar
9. Sürüm notları

> Sürüm: 24.11.2026 · Hazırlayan: Kerberos Bilgi Teknolojileri Bu doküman API'nin **gerçek kod davranışına** göre yazılmıştır.

1. Giriş

EsnafPazar Satıcı API'si, EsnafPazar pazaryerinde mağazası olan bir satıcının ürün, stok, fiyat, sipariş, iade ve kargo işlemlerini yazılımla yönetmesini sağlar. Entegratör firmalar (çoklu pazaryeri yazılımları) her satıcı için o satıcının kendi API anahtarıyla bağlanır.

Konu	Değer
Temel URL	`https://www.esnafpazar.com/api/v1/`
Biçim	İstek ve cevap gövdeleri JSON (`Content-Type: application/json`). Tek istisna: görsel dosyası yükleme (`multipart/form-data`) ve etiket PDF'i (`application/pdf`).
Karakter seti	UTF-8. Türkçe karakterler kaçışsız döner (`"Gömlek"`).
Para birimi	TRY (sabit). Tutarlar ondalık sayıdır; **ürün detayında** tutarlar metin olarak döner (`"249.90"`).
Kimlik	Satıcıya özel API anahtarı + gizli anahtar → 1 saatlik erişim belirteci (JWT).
Test ortamı	**Yok.** Ayrıntı: §8.

Tarih ve saat biçimleri (alan alan)

API'de tek bir tarih biçimi yoktur. Aşağıdaki tablo kodun gerçekte döndürdüğü biçimdir. "Unix (metin)" = saniye cinsinden unix zaman damgası, JSON'da metin olarak (`"1795516800"`).

Nerede	Alan	Biçim
Her cevabın zarfı	`timestamp`	`"2026-11-24 14:05:09"` (`Y-m-d H:i:s` , Türkiye saati)
Ürün **listesi** (`GET` `/products`)	`created_at` , `updated_at`	`"2026-11-24 14:05:09"` (`Y-m-d H:i:s` , Türkiye saati)
Ürün **detayı** (`GET` `/products/{id}`)	`created_at` , `updated_at` , `stock[].updated_at`	Unix (metin)
Ürün görselleri	`created_at`	Unix (metin)
`GET` `/products/{id}/stock`	`variants[].created_at` , `updated_at`	Unix (metin)
`POST/PATCH /products/{id}/stock` cevabı	`updated_at`	`Y-m-d H:i:s` (Türkiye saati)
`GET` `/products/stock-alerts`	`last_updated`	Unix (metin)

Nerede	Alan	Biçim
Stok uçları (`/stock`)	`updated_at` , `created_at`	Unix (metin)
Siparişler (liste ve detay)	`created_at` , `updated_at`	Unix (metin)
Sipariş faturası	`invoice.date`	ISO 8601 (`"2026-11-24T14:05:09+03:00"`)
Geliver kargo hareketleri	`geliver.tracking.events[].at` , `last_event_at`	ISO 8601
İadeler	`return_date` , `updated_at`	Unix (metin)
Markalar, kategoriler, webhook listesi	`created_at` , `updated_at` , `last_triggered_at`	Unix (metin)
Webhook gövdesi	`timestamp`	Unix (**sayı**)
`order.created` webhook verisi	`data.created_at`	ISO 8601

Saat dilimi: API'nin tamamı **Europe/Istanbul (UTC+3)** saatiyle çalışır. `Y-m-d H:i:s` biçimindeki tüm alanlar (zarftaki `timestamp` dahil) Türkiye saatidir ve saat dilimi eki taşımaz. Unix alanlar saat diliminden bağımsızdır. ISO 8601 alanlar eklerini (`+03:00`) kendileri taşır. Örnek: ürün listesindeki `"created_at": "2026-11-24 14:03:33"` ile aynı ürünün detayındaki `"created_at": "1795518213"` aynı anı gösterir.

Filtre parametrelerinde kabul edilen biçimler:

Parametre	Kabul edilen
GET `/products?updated_after=`	Unix saniye **veya** ISO 8601 / `strtotime` ile okunabilen tarih. Geçersizse 400.
GET `/orders?created_after=` , `created_before=` , `updated_after=`	`strtotime` ile okunabilen her biçim (ör. `2026-11-24` , `2026-11-24T10:00:00+03:00`). Unix saniye **göndermeyin** (bu uçta metin tarih olarak yorumlanır).
`invoice_date`	Unix saniye veya ISO 8601.

> İpucu: Unix zaman alanları JSON'da **metin** olarak gelir (`"1795521909"`); sayıya kendiniz çevirin. Sayfalama `meta` alanları sayıdır (§4.2). Veritabanından okunan bazı kimlik alanları (ör. iade `product_id`) metin gelebilir; kimlikleri karşılaştırmadan önce tipini eşitleyin.

2. Hızlı başlangıç

Adım 1 — API anahtarını alın

EsnafPazar API'sinde kimlik bilgisi **entegratöre değil, her satıcıya** verilir. Entegrasyon yazılımınızda "EsnafPazar mağazanı bağla" ekranı, satıcıdan aşağıdaki iki bilgiyi istemelidir:

Satıcıdan istenecek	Açıklama
API Key	64 karakter, satıcıya özel
API Secret	64 karakter, gizli tutulmalı (şifre gibi saklayın, loglara yazmayın)

Satıcı bu bilgileri **Satıcı Paneli › API Yönetimi › API Ayarları** (`https://www.esnafpazar.com/bayi/api-ayarlari`) sayfasından üretir.

> **Test hesabı:** Ayrı bir sandbox ortamı yoktur. Geliştirme ve test için EsnafPazar'dan size özel bir **test satıcı hesabı** isteyin; anahtarı o hesabın panelinden kendiniz üretebilirsiniz. Gerçek mağazalarda sipariş/iade uçlarını denerken müşteriye bildirim gideceğini unutmayın.

Satıcı, **Satıcı Paneli › API Yönetimi › API Ayarları** sayfasından "API anahtarı oluştur" ile bir `api_key` ve `api_secret` üretir (her biri 64 karakter). Satıcı bu ikisini entegratöre iletir.

- Anahtar satıcıya özeldir; entegratör için ayrı bir "uygulama/partner" anahtarı yoktur. Her satıcı için ayrı anahtar çifti saklanır.
- Satıcı anahtarı panelden **sıfırlarsa** eski anahtar ve tüm refresh token'lar geçersiz olur; zaten verilmiş access token'lar süresi (en çok 1 saat) dolana kadar çalışır.

Adım 2 — Token alın

```
curl -X POST https://www.esnafpazar.com/api/v1/auth/token \
  -H "Content-Type: application/json" \
  -d '{"api_key":"<API_KEY>","api_secret":"<API_SECRET>"}'
```

```
{
  "success": true,
  "timestamp": "2026-11-24 14:05:09",
  "request_id": "req_6743a1b2c3d4e5.12345678",
  "data": {
    "access_token": "<ACCESS_TOKEN>",
    "token_type": "Bearer",
    "expires_in": 3600,
    "refresh_token": "<REFRESH_TOKEN>",
    "seller": { "id": 128, "name": "Örnek Mağaza", "email": "magaza@ornek.com" }
  },
  "message": "Token generated successfully"
}
```

Adım 3 — İlk istek

```
curl https://www.esnafpazar.com/api/v1/products?per_page= 10   \
  -H "Authorization: Bearer <ACCESS_TOKEN>"
```

3. Kimlik doğrulama

Token uçları (`/auth/token` , `/auth/token/refresh`) dışındaki **tüm** uçlar şu başlığı ister:

```
Authorization: Bearer <ACCESS_TOKEN>
```

`X-Seller-ID` gibi ek başlık **gerekmez**; satıcı kimliği token'dan okunur.

Belirteç	Ömür	Not
Access token	3600 sn (1 saat)	HS256 imzalı JWT. İçinde `seller_id` , `iat` , `exp` .
Refresh token	30 gün	64 karakter rastgele değer. Her yenilemede **yenisi verilir, eskisi geçersiz olur** (rotasyon).

> **Önemli:** Bir satıcı için sunucuda **tek** refresh token tutulur. Aynı satıcıya iki farklı sistem (ör. entegratör + satıcının başka bir aracı) aynı anahtarla bağlanırsa, biri token aldığında diğerinin refresh token'ı geçersiz olur. Böyle bir durumda 401 alan taraf POST `/auth/token` ile anahtar/gizli anahtardan yeniden token almalıdır.

3.1 Token al — POST `/auth/token`

Gövde alanı	Tip	Zorunlu	Açıklama
`api_key`	string	evet	Panelden üretilen anahtar
`api_secret`	string	evet	Panelden üretilen gizli anahtar

Hatalar: `400 API key and secret are required` · `401 Invalid API credentials` (anahtar yanlış ya da satıcı pasif).

3.2 Token yenile — POST `/auth/token/refresh`

```
curl -X POST https://www.esnafpazar.com/api/v1/auth/token/refresh \
  -H "Content-Type: application/json" \
  -d '{"refresh_token":"<REFRESH_TOKEN>"}'
```

Cevap `data` : `access_token` , `token_type` , `expires_in` , `refresh_token` (yeni). Hatalar: `400 Refresh token is required` · `401 Invalid refresh token` (süresi dolmuş, başka istemci yenilemiş veya anahtar sıfırlanmış).

3.3 Token doğrula — GET `/auth/token/verify`

Bearer ile çağrılır. Cevap: `{"seller_id": 128, "seller_name": "Örnek Mağaza", "valid": true}` .

3.4 Oturumu kapat — POST `/auth/token/revoke`

Bearer ile çağrılır; satıcının refresh token'ını siler. HTTP **204** döner (gövde beklemeyin). Access token süresi dolana kadar çalışmaya devam eder.

3.5 Kimlik hataları (tüm uçlar)

HTTP	`error.message`	Anlamı
401	`Authentication required`	`Authorization` başlığı yok ya da Bearer biçimi yanlış
401	`Invalid token`	İmza geçersiz veya süresi dolmuş → yenileyin
401	`Seller not found`	Satıcı hesabı pasif/kapalı

Önerilen istemci davranışı: Access token'ı süresinden ~5 dakika önce yenileyin; 401 `Invalid token` gelirse bir kez refresh deneyin, o da 401 ise POST `/auth/token` ile baştan alın.

---

4. Ortak kurallar

4.1 Cevap zarfı

Başarılı cevap:

```
{
  "success": true,
  "timestamp": "2026-11-24 14:05:09",
  "request_id": "req_6743a1b2c3d4e5.12345678",
  "data": { },
  "message": "..."
}
```

Hatalı cevap:

```
{
  "success": false,
  "timestamp": "2026-11-24 14:05:09",
  "request_id": "req_6743a1b2c3d4e5.12345678",
  "error": {
    "message": "Status transition not allowed: 10 -> 5 (no further status change is allowed for this order)",
    "code": "Conflict",
    "details": { "current_status_code": 10, "requested_status_code": 5, "allowed_status_codes": [] }
  }
}
```

- `data` ve `message` alanları boş olduğunda hiç gelmeyebilir.
- `error.code` makine kodu değil, HTTP durum metnidir (`Bad Request` , `Unauthorized` , `Not Found` , `Method Not Allowed` , `Conflict` , `Unprocessable Entity` , `Too Many Requests` , `Internal Server Error` , `Bad Gateway` , `Service Unavailable`). Karar verirken **HTTP durum kodunu** kullanın.
- `error.message` İngilizce serbest metindir; değişebilir, metne göre dallanmayın.
- `request_id` : İsteğe `X-Request-ID` başlığı (harf, rakam, `.` , `_` , `-` ; en çok 64 karakter) koyarsanız aynısı döner; koymazsanız sunucu üretir. Destek talebinde bu değeri iletin.
- **Toplu uçlarda kısmi hata:** Bazı uçlar (POST `/products/bulk` , `/products/bulk-stock-update` , PATCH `/stock/bulk`) hata olduğunda `success:false` döner ve sonuç gövdesi `data` yerine **`error.details`** içinde gelir. Ayrıntı her uçta belirtilmiştir.
- 204 cevaplarında gövde beklemeyin.

Örnek hata zarfları:

```
{ "success": false, "timestamp": "2026-11-24 14:05:09", "request_id": "req_6743a1b2c3d4e5.12345678",
  "error": { "message": "Product not found", "code": "Not Found", "details": null } }
```

```
{ "success": false, "timestamp": "2026-11-24 14:05:09", "request_id": "req_6743a1b2c3d4e5.12345678",
  "error": { "message": "Creating orders via the seller API is not supported. Orders are created only by customer checkout on EsnafPazar.",
             "code": "Method Not Allowed", "details": null } }
```

```
{ "success": false, "timestamp": "2026-11-24 14:05:09", "request_id": "req_6743a1b2c3d4e5.12345678",
  "error": { "message": "stock_updates array is required", "code": "Bad Request", "details": null } }
```


`details` hata ayrıntısı yoksa `null` gelir.

4.2 Sayfalama

Sayfalı uçlar: `GET` `/products`, `GET` `/orders`, `GET` `/stock`, `GET` `/brands`, `GET` `/categories`.

Parametre	Varsayılan	Sınır
`page`	1	≥ 1
`per_page`	50	1–100 (100'den büyük değer 100'e çekilir)

Cevaptaki `meta`:


```
"meta": {
  "current_page": 1,
  "total_pages": 3,
  "per_page": 50,
  "total_items": 127,
  "has_next": true,
  "has_prev": false
}
```


`meta` alanlarının hepsi sayı (`current_page`, `total_pages`, `per_page`, `total_items`) veya boolean (`has_next`, `has_prev`) olarak gelir. Sonuç yoksa `total_pages: 0`, `total_items: 0`, `has_next: false` döner. Döngüyü `has_next` ile kurun.

4.3 Hız sınırı (rate limit)

- Satıcı başına **saatte 1000 istek** (sabit saatlik pencere; sayaç her saat başında sıfırlanır, kayan pencere değildir).
- `/auth/*` uçları (token al, yenile, doğrula, kapat) sınıra dahil değildir ve rate limit başlığı **döndürmez**. `GET` `/auth/token/verify` Bearer istese de bu kurala girer.
- Diğer tüm uçların cevabında şu başlıklar gelir:

Başlık	Anlamı
`X-RateLimit-Limit`	Pencere başına izin (1000)
`X-RateLimit-Remaining`	Kalan istek

Başlık	Anlamı
`X-RateLimit-Reset`	Pencerenin sıfırlanacağı unix saniye (bir sonraki saat başı)

Başlık adları cevapta küçük harfle gelebilir (HTTP/2'de böyledir). HTTP başlık adları büyük/küçük harf duyarsızdır; başlığı okurken adı harf duyarsız karşılaştırın. Gerçek cevaptan örnek:

```
x-ratelimit-limit: 1000
x-ratelimit-remaining: 983
x-ratelimit-reset: 1795528800
```

- Aşımda `429 Rate limit exceeded`, `Retry-After` başlığı (saniye) ve `error.details: {"limit":1000,"reset":<unix>}`.
- Toplu uçlar (bulk) tek istek sayılır; stok/fiyat senkronunu toplu uçlarla yapın.

4.4 HTTP durum kodları

Kod	Ne zaman
200	Başarılı
201	Oluşturuldu (ürün, marka, webhook, kısmi gönderim)
204	Silindi / iptal edildi (gövde yok)
400	Eksik alan, geçersiz JSON, geçersiz durum adı, geçersiz kargo grubu, eksik id, geçersiz sorgu parametresi
401	Kimlik hatası (§3.5)
403	Kod tarafından üretilmez; sunucu/güvenlik katmanı (ör. CDN güvenlik duvarı) engellerse görülebilir. Tekrarlıyorsa destekle iletişime geçin.
404	Kayıt yok **veya** kayıt bu satıcıya ait değil. Mesaj uca göre değişir: okuma uçlarında `Product not found`, `Order not found` gibi; yazma uçlarında `... not found or access denied`. Mesaja değil HTTP koduna göre karar verin.
405	Desteklenmeyen metot (ör. kategoriye POST, `POST` `/orders`)
409	Çakışma: kural dışı sipariş/iade durum geçişi, iptal edilemeyecek sipariş, aynı isimde marka, Geliver kuralı
422	**Doğrulama hatası** (istek biçimi doğru ama içerik kabul edilmedi): marka bulunamadı/onaysız, geçersiz filtre çifti, kategori yok, görsel indirilemedi/desteklenmeyen tür/5 MB üstü, SKU 255 karakteri aşıyor, kargo kuralı ve bilinmeyen `shipping_company_id`, bilinmeyen stok kodu, iade onayında eksik `return_code`. Toplu işlemde satırların tamamı (stok uçları) veya en az biri (`POST` `/products/bulk`) başarısız. Geliver adres/alıcı hatası.
429	Hız sınırı aşıldı
500	Yalnız sunucu hatası. Mesaj genel olur: `... (internal error, request_id: req_...)`
502 / 503	Geliver kargo servisi hatası (`Bad Gateway`) / servis kullanılamıyor (`Service Unavailable`)

> **Not (doğrulama hataları 422):** Ürün ve stok uçlarındaki iş kuralı hataları HTTP **422** ile açıklayıcı mesaj ve `error.details` içinde hatalı alanla döner, ör.: `json { "success": false, "error": { "message": "Failed to create product: Brand not found", "code": "Unprocessable Entity", "details": { "field": "brand_id", "value": 512 } } }` `details.field` değerleri: `brand_id`, `filters`, `category_id`, `images`, `sku`, `stock_code`, `main_stock`, `stock_variants`, `shipping_company_id`, `shipping_cost`. Stok kodu hatalarında ayrıca `details.error_code: "stock_code_not_found"`. 422'yi isteği düzeltip yeniden göndererek çözün; 500 sunucu tarafı hatadır, `request_id` ile destekle iletişime geçin. (Bu güncellemeden önce bu hatalar 500 dönüyordu.)

4.5 URL kullanımı

- Yol sonunda `/` kullanmayın.
- Kategoriler uçlarında (`/categories/{id}`, `/categories/tree`, `/categories/{id}/attributes`) yola **sorgu dizesi eklemeyin**; ekleyince istek kategori listesine düşer. Liste ucunda (`/categories?parent_id=...`) sorgu dizesi normaldir.

---

5. Uçlar

Örneklerde `BASE=https://www.esnafpazar.com/api/v1` ve `TOKEN=<ACCESS_TOKEN>` kabul edilmiştir.

5.1 Kategoriler ve özellikler

Salt okunurdur (POST/PUT/DELETE → 405). Tüm satıcılar aynı kategori ağacını görür.

Kategori listesi — GET `/categories`

Sorgu	Tip	Zorunlu	Açıklama
`page`, `per_page`	int	hayır	Sayfalama
`parent_id`	int	hayır	Bu kategorinin alt kategorileri (`0` = ana kategoriler)
`level`	int	hayır	`1`, `2` veya `3` (ağaç derinliği)
`search`	string	hayır	Kategori adında arama

Cevap `data.categories[]`: `id`, `name`, `parent_id`, `parent_name`, `order`, `commission_rate`, `seo{title,keyword,description}`, `images{icon,image}`, `features{show_on_homepage,has_legal_warning,installment_available}`, `stats{child_count,product_count}`, `created_at`, `updated_at`; ayrıca `data.meta`.

Kategori ağacı — GET `/categories/tree`

Sayfasız, tüm ağaç. `data.category_tree[]`: `id`, `name`, `order`, `commission_rate`, `show_on_homepage`, `images`, `children[]` (aynı yapı). `data.total_categories`.

Ürün yalnız **yaprak** kategoriye (çocuğu olmayan, `children: []`) eklenmelidir. API bunu zorlamaz, ama yaprak olmayan kategorideki ürün vitrinde doğru listelenmeyebilir.

Kategori detayı — GET `/categories/{id}`

Listedeki alanlara ek: `path[]{id,name}` (kökten bu kategoriye), `seo.tags`, `configuration{variants[],filters[]}` (bu kategoriye bağlı varyant ve filtre kimlikleri, metin dizisi), `subcategories[]`. Kategori yoksa 404.

Kategori özellikleri — GET `/categories/{id}/attributes`

Ürün eklerken kullanacağınız **varyant** (beden/renk gibi, stok tutan seçenekler) ve **filtre** (marka, materyal gibi özellikler) sözlüğü.

```
curl $BASE/categories/899/attributes -H "Authorization: Bearer $TOKEN"
```

```
{
  "success": true,
  "data": {
    "category_id": 899,
    "variants": [
      {
        "id": 3, "name": "Beden", "system_name": "beden", "sort_order": 1,
        "is_color": false, "is_active": true,
        "options": [ { "id": 41, "name": "S", "sort_order": 1 }, { "id": 42, "name": "M", "sort
_order": 2 } ]
      }
    ],
    "filters": [
      {
        "id": 7, "name": "Materyal", "system_name": "materyal", "sort_order": 2,
        "is_brand_filter": false, "is_active": true,
        "options": [ { "id": 310, "name": "Deri", "sort_order": 1 } ]
      }
    ]
  }
}
```


- Ürün oluştururken varyant için `variants[].attributes: {"<variants.id>": <options.id>}`, filtre için `filters[]: {"filter_id": <filters.id>, "option_id": <options.id>}` gönderilir.
- Hangi özelliğin zorunlu olduğu, serbest metin olup olmadığı bilgisi API'de **yoktur**; tüm özellikler seçim listesidir.

5.2 Varyant sözlüğü

Tüm varyant tanımları — GET `/variants`

Kategoriden bağımsız, aktif tüm varyantlar ve seçenekleri. `data.variants[]` : `id` , `name` , `system_name` , `sort_order` , `is_color` , `is_active` , `options[]{id,name,sort_order}` . Sayfasız. Kimlik eşlemesini önbelleğe alıp günde bir yenilemeniz yeterlidir.

5.3 Markalar

Markalar satıcıya özeldir: bu uçlar **yalnız satıcının kendi oluşturduğu** markaları gösterir; platformun genel marka listesi API'de yoktur.

Marka listesi — GET `/brands`

Sorgu	Tip	Açıklama
`page` , `per_page`	int	Sayfalama
`status`	string	`pending` , `active` , `rejected`

`data.brands[]` : `id` , `brand_name` , `logo` (URL veya null), `categories[]` (kategori **adları**), `status` , `rejection_reason` , `created_at` , `updated_at` ; `data.meta` .

Marka detayı — GET `/brands/{id}`

Listedeki alanlar + `last_admin_action`. Başka satıcının markası → 404.

Marka oluştur — POST `/brands`

Gövde	Tip	Zorunlu	Açıklama
`brand_name`	string	evet	Marka adı (satıcı içinde benzersiz)
`category_ids`	int[] veya int	evet	Markanın geçerli olduğu kategori kimlikleri (en az biri var olmalı)
`logo`	string	hayır	Logo **dosya adı** (bkz. not)

```
curl -X POST $BASE/brands -H "Authorization: Bearer $TOKEN" -H "Content-Type: application/json"
\
  -d '{"brand_name":"Atölye Deri","category_ids":[899,900]}'
```


```
{ "success": true, "data": { "brand_id": 512, "message": "Brand created and activated successfu
lly.", "status": "active" }, "message": "Brand created and activated successfully" }
```


- HTTP 201. API ile oluşturulan marka **doğrudan aktif** olur ve hemen ürünlerde kullanılabilir.
- Hatalar: `400 Missing required fields` · `400 Invalid category IDs` · `409 Brand name already exists for this seller`.
- `logo` alanı URL kabul etmez; değer `https://www.esnafpazar.com/dosya/marka/` önüne eklenerek gösterilir. Logo yükleme ucu yoktur; logoyu boş bırakmanızı öneririz.

Marka güncelle / sil — PUT `/brands/{id}`, DELETE `/brands/{id}`

Yalnız **onay bekleyen** (`pending`) markalarda çalışır. API ile oluşturulan markalar aktif olduğundan bu uçlar pratikte `400 Only pending brands can be updated/deleted` döner. Marka değişikliği için EsnafPazar destek ekibine başvurun.

5.4 Ürünler

Ürün durumları ve onay süreci (önce okuyun)

API `status`	Anlamı
`passive`	Pasif. **Yeni eklenen her ürün** bu durumda başlar.
`active`	Satıcı aktif etmiş. Vitrinde görünmesi için ayrıca EsnafPazar onayı gerekir.
`deleted`	EsnafPazar yönetimi ürünü **reddetmiş** (yayından kaldırmış). API'den durumu değiştirilemez. `review_status` bu üründe `rejected`'tır.

Her ürün cevabında (§5.4.1, §5.4.2) inceleme alanları da döner:

Alan	Tip	Anlamı
`review_status`	string	`pending` (EsnafPazar incelemesi bekleniyor) · `approved` (incelendi, onaylı) · `rejected` (reddedildi / yayından kaldırıldı). Satıcı panelindeki ürün durum etiketiyle aynı kural. Sitede ürün onayı kapalıysa reddedilmemiş her ürün `approved` görünür.
`is_listed`	boolean	Ürün şu an vitrinde (müşteriye) görünüyor mu: `status` `active` **ve** (ürün onayı açıksa) `review_status` `approved` . Stok 0 olması bu alanı değiştirmez.
`review_note`	string | null	Yönetimin ret gerekçesi (panelde "Açıklama" olarak görünen metin). Yalnız `rejected` üründe dolu; diğerlerinde `null` .

- API ile eklenen ürün `status: "passive"` , `review_status: "pending"` olarak oluşur ve EsnafPazar yönetim incelemesine düşer.
- Onay geldikten sonra ürün **otomatik aktif olmaz**; satıcının (entegratörün) POST `/products/bulk-update` ile `status: 1` göndermesi gerekir (§5.4.8).
- Onay henüz gelmemişse aktif etme isteği satır hatası döner: `Product is awaiting marketplace review; it can be activated after approval` .
- Onay durumunu `review_status` alanından izleyin; onay bekleyenleri GET `/products?review_status=pending` , reddedilenleri gerekçesiyle GET `/products?review_status=rejected` ile listeleyin (bkz. §7.2).
- Ürünün **adı, açıklaması, kategorisi, barkodu veya SKU'su** gerçekten değişirse ürün yeniden incelemeye düşer ve onay gelene kadar vitrinden kalkar. Bu sırada API'de `status` `active` görünmeye devam eder; `review_status` `pending` , `is_listed` `false` olur. Fiyat, stok, görsel ve kargo değişikliği yeniden inceleme gerektirmez. Aynı değeri tekrar göndermek incelemeye düşürmez.

5.4.1 Ürün listesi — GET `/products`

Sorgu	Tip	Açıklama
`page` , `per_page`	int	Sayfalama
`status`	string	`active` , `passive` , `deleted` . Verilmezse reddedilenler (`deleted`) hariç hepsi.
`review_status`	string	`pending` , `approved` , `rejected` . `rejected` verildiğinde (`status` verilmese de) reddedilen ürünler listelenir. Geçersiz değer → `400` . Ürün onayı kapalı sitede `pending` boş liste döner.
`category_id`	int	Kategori
`updated_after`	unix / ISO	Bu andan sonra değişen ürünler (stok/fiyat değişimi dahil)
`sku`	string	Ürün SKU'su (tam eşleşme)
`barcode`	string	Barkod (tam eşleşme)
`include`	string	Virgülle: `images` , `variants` , `features` , `stock`

Sıralama: son değişen önce.

```
{
  "success": true,
  "data": {
    "products": [
      {
        "id": 10452, "sku": "ATL-CNT-001", "title": "Deri Omuz Çantası", "description": "<p>...
</p>",
        "barcode": "8690000000017", "price": 749.9, "discount_price": 749.9, "market_price": 0,
        "category_id": 899, "category_name": "Omuz Çantası", "brand_name": "Atölye Deri", "bran
d_id": 2231,
        "status": "passive", "review_status": "approved", "is_listed": false, "review_note": nu
ll,
        "image": "https://www.esnafpazar.com/dosya/urun/10452-6743a1b2c3d4e.webp",
        "stock_quantity": 25, "kargo_grubu": null,
        "updated_at": "2026-11-24 14:05:09", "created_at": "2026-11-24 14:03:33"
      }
    ],
    "meta": { "current_page": 1, "total_pages": 1, "per_page": 50, "total_items": 1, "has_nex
t": false, "has_prev": false }
  }
}
```


- `discount_price` her zaman `price` ile aynıdır (indirimli fiyat ayrı tutulmaz; satış fiyatı `price`'tır).
- `brand_id` burada markanın **filtre seçeneği kimliğidir**, POST `/brands` cevabındaki `brand_id` ile aynı değildir. Marka eşlemesini `brand_name` ile yapın.
- `kargo_grubu` listede her zaman `null` döner; değeri görmek için kullanmayın.
- `stock_quantity`: tüm stok satırlarının toplamı.
- `market_price` ürün kartındaki piyasa fiyatıdır; kartta girilmemişse listede `0` döner (yukarıdaki örnek). Ürün detayında (GET `/products/{id}`) bu durumda stok satırlarının en yüksek `market_price` değeri, o da yoksa `price` gösterilir. Varyant bazlı piyasa fiyatı için `stock[]` / `variants[]` alanlarını kullanın.
- `created_at`, `updated_at` Türkiye saatidir (§1).
- `review_status`, `is_listed`, `review_note`: §5.4 başındaki tablo.

5.4.2 Ürün detayı — GET `/products/{id}`


```
{
  "success": true,
  "data": {
    "id": 10452, "sku": "ATL-CNT-001", "title": "Deri Omuz Çantası", "description": "<p>...</p
>",
    "barcode": "8690000000017", "price": "749.90", "discount_price": "749.90", "market_price":
"899.90",
    "category_id": 899, "category_name": "Omuz Çantası", "brand_name": "Atölye Deri", "brand_i
d": 2231,
    "status": "passive", "review_status": "pending", "is_listed": false, "review_note": null,
    "shipping": {
      "requires_payment": false, "cost": 0, "estimated_delivery_days": 0,
      "company": { "id": null, "name": null }
    },
    "images": [ { "url": "https://www.esnafpazar.com/dosya/urun/10452-6743a1b2c3d4e.webp", "is_
main": true } ],
    "variants": [
      { "sku": "ATL-CNT-001-S", "variant_id": 88120, "variant_option_id": 41,
        "attributes": { "Beden": "S" }, "price": 749.9, "market_price": 899.9, "stock_quantit
y": 10 }
    ],
    "features": { "Materyal": "Deri" },
    "stock": [ { "sku": "ATL-CNT-001-S", "price": 749.9, "market_price": 899.9, "quantity": 10,
"updated_at": "1795521909" } ],
    "updated_at": "1795521909",
    "created_at": "1795521909"
  }
}
```


- Tutarlar burada **metin**dir.
- `market_price` kartta girilmemişse stok satırlarının en yüksek piyasa fiyatına, o da yoksa `price`'a düşer. Bu yüzden aynı ürün listede `market_price: 0`, detayda `"749.90"` görünebilir.
- Varyantsız üründe (tek stok satırı) `variants` boş dizidir (`[]`); stok kodu yalnız `stock[].sku` alanındadır. Örnek:


```
{
  "id": 10452, "sku": "ATL-CNT-001", "price": "749.90", "discount_price": "749.90", "market_pri
ce": "749.90",
  "status": "passive", "review_status": "approved", "is_listed": false, "review_note": null,
  "shipping": { "requires_payment": false, "cost": 0, "estimated_delivery_days": 1, "company":
{ "id": null, "name": null } },
  "images": [ { "url": "https://www.esnafpazar.com/dosya/urun/10452-6743a1b2c3d4e.webp", "is_ma
in": true } ],
  "variants": [],
  "features": { "Materyal": "Deri", "Renk": "Siyah" },
  "stock": [ { "sku": "1045207567019", "price": 749.9, "market_price": 749.9, "quantity": 4, "u
pdated_at": "1795521909" } ],
  "updated_at": "1795522502", "created_at": "1795521909"
}
```


- `shipping.requires_payment` : `true` yalnız alıcı ödemeli kargoda.
- `stock[].updated_at` stok satırının **oluşturulma** zamanıdır; değişiklik takibi için GET `/products?updated_after=` kullanın.
- Başka satıcının ürünü → 404.

- Reddedilen üründe örnek: `"status": "deleted", "review_status": "rejected", "is_listed": false, "review_note": "Ürün görselleri kurallara uygun değil."`

5.4.3 Ürün oluştur — POST `/products`

Gövde	Tip	Zorunlu	Açıklama
`title`	string	evet	Ürün adı
`category_id`	int	evet	Yaprak kategori. Sistemde olmayan kategori → `422` (`details.field: "category_id"`).
`price`	number	evet	Satış fiyatı (KDV dahil)
`market_price`	number	hayır	Piyasa (üstü çizili) fiyatı
`description`	string	hayır	HTML açıklama
`sku`	string	hayır	Ürün (model) kodu, ≤ 255 karakter. **Kendi SKU'nuzu mutlaka gönderin**: toplu upsert ve `?sku=` araması bununla çalışır.
`barcode`	string	hayır	Barkod (metin gönderin; baştaki sıfırlar korunur)
`brand_id`	int	hayır	POST `/brands` / GET `/brands` cevabındaki marka `id` 'si. Markanın **bu satıcıya ait ve aktif** olması gerekir.
`filters`	object[]	hayır	`[{"filter_id": 7, "option_id": 310}]` — kategori özellikleri
`images`	array	hayır	Görsel URL'leri: `["https://..."]` veya `[{"url":"https://...","is_main":true}]` . İlk görsel (veya `is_main:true`) kapak olur.
`variants`	object[]	hayır	Bkz. aşağıdaki tablo
`stock_quantity`	int	hayır	Varyantsız ürünün stoğu
`stock_code`	string	hayır	Varyantsız ürünün stok kodu (verilmezse `sku` , o da yoksa `DEFAULT`). `stock_quantity` ve `variants` verilmezse de ürün bu kodla **stok 0** olan bir stok satırıyla oluşur (satıcı paneli gibi); stoğu sonra §5.7 uçlarıyla güncellersiniz.
`kargo_grubu`	int	hayır	Kargo grubu (§5.11). Verilmezse kategorinin grubu atanır.
`shipping_type`	int	koşullu	Yalnız **kendi kargo anlaşmasıyla** gönderen satıcıda: `1` alıcı öder, `2` satıcı öder (varsayılan 2)
`shipping_cost`	number	koşullu	`shipping_type=1` ise > 0 zorunlu
`shipping_company_id`	int	koşullu	Kendi kargolu satıcıda **zorunlu**. Kimlikleri GET `/shipping/companies` verir (§5.11); listede olmayan kimlik → `422` .

`variants[]` öğesi:

Alan	Tip	Zorunlu	Açıklama
`sku`	string	evet	Varyantın stok kodu (≤ 255). Boşsa varyant atlanır.

Alan	Tip	Zorunlu	Açıklama
`attributes`	object	evet*	`{"<varyant_id>": <seçenek_id>}`, ör. `{"3": 41}`. Kimlikler `/categories/{id}/attributes` veya `/variants`'tan. İsim (`{"Beden":"S"}`) **kabul edilmez**, sessizce atlanır. *Gönderilmezse varyant tanımsız bir kayıtla oluşur; her zaman gönderin.
`price`	number	hayır	Varyant fiyatı (yoksa ürün fiyatı)
`market_price`	number	hayır	Yoksa `price`
`stock_quantity`	int	hayır	Varsayılan 0

Örnek:

```
curl -X POST $BASE/products -H "Authorization: Bearer $TOKEN" -H "Content-Type: application/jso
n" -d '{
  "title": "Deri Omuz Çantası",
  "category_id": 899,
  "price": 749.90,
  "market_price": 899.90,
  "sku": "ATL-CNT-001",
  "barcode": "8690000000017",
  "description": "<p>El yapımı hakiki deri.</p>",
  "brand_id": 512,
  "filters": [ { "filter_id": 7, "option_id": 310 } ],
  "images": [ { "url": "https://cdn.ornek.com/urun/atl-cnt-001-1.jpg", "is_main": true }, "http
s://cdn.ornek.com/urun/atl-cnt-001-2.jpg" ],
  "variants": [
    { "sku": "ATL-CNT-001-S", "attributes": { "3": 41 }, "price": 749.90, "stock_quantity": 10
},
    { "sku": "ATL-CNT-001-M", "attributes": { "3": 42 }, "price": 749.90, "stock_quantity": 15
}
  ],
  "kargo_grubu": 3
}'
```

Başarılı cevap (HTTP 201):

```
{
  "success": true,
  "data": { "product_id": 10452, "status": "passive", "review_status": "pending", "message": "P
roduct created successfully" },
  "message": "Product created successfully"
}
```

Hatalar:

HTTP	Mesaj (özet)	Sebep
400	`Invalid JSON data`	Gövde JSON değil
400	`Missing required fields: ...`	`title`, `category_id`, `price` eksik veya boş/0

HTTP	Mesaj (özet)	Sebep
400	`Invalid kargo_grubu ...`	Pasif/olmayan grup
422	`shipping_company_id is required ...` / `shipping_company_id not found ...` / `shipping_cost must be greater than 0 ...`	Kendi kargolu satıcıda kargo kuralı (`details.field`)
422	`Failed to create product: category_id not found ...`	Kategori sistemde yok
422	`Failed to create product: Brand not found` / `Brand not found or access denied` / `Brand is waiting for admin approval`	`brand_id` bu satıcının aktif markası değil
422	`Failed to create product: Invalid filter/option pair ...` / `Invalid filter format ...` / `filters must be an array`	`filters` hatalı
422	`Failed to create product: Remote image could not be downloaded ...` / `Unsupported remote image type ...` / `Remote image too large (max 5MB)`	Görsel indirilemedi (tek bir hatalı görsel tüm isteği geri alır)
422	`Failed to create product: sku maximum length is 255 characters`	SKU / stok kodu çok uzun
500	`Failed to create product (internal error, request_id: ...)`	Sunucu hatası

İşlem tek parça (transaction): herhangi bir adım hata verirse ürün hiç oluşmaz.

Görsel URL kuralları (oluşturma, güncelleme ve toplu uçlarda aynı):

- Yalnız `http://` veya `https://` ; port 80, 443, 8080 veya 8443.
- Adres herkese açık bir sunucuya çözülmelidir (yerel/özel ağ adresleri reddedilir).
- Yönlendirme en çok 3 adım izlenir ve her adım aynı kurallarla doğrulanır.
- Dosya en çok 5 MB; JPEG, PNG, GIF veya WebP (içerikten tespit edilir). Bağlantı 5 sn, toplam indirme 15 sn içinde bitmelidir.
- Görsel sunucunuza kopyalanır; cevaplarda EsnafPazar adresiyle (`https://www.esnafpazar.com/dosya/urun/...`) döner.

5.4.4 Ürün güncelle — PUT `/products/{id}` veya PATCH `/products/{id}`

İki metot **aynı** davranır: yalnız gönderdiğiniz alanlar değişir.

Gövde	Tip	Açıklama
`title` , `description` , `barcode` , `sku` , `category_id`		Değişirse ürün yeniden incelemeye düşer
`price`	number	Satış fiyatı; ürünün **tüm stok satırlarına (varyantlara)** aynı fiyat yazılır
`discount_price`	number	Gönderilirse satış fiyatı olarak `price` 'ın yerine geçer
`market_price`	number	Tüm stok satırlarına yazılır
`images`	array	**Tüm görselleri değiştirir.** Mevcut görseli korumak için `GET` cevabındaki URL'yi aynen listeye koyun (yeniden indirilmez).

Gövde	Tip	Açıklama
`variants`	object[]	**Tüm varyant tanımlarını silip yeniden yazar** (stok satırlarını silmez). Yalnız fiyat/stok için §5.7 uçlarını kullanın.
`filters`	object[]	**Tüm filtreleri (marka dahil) değiştirir**
`stock_quantity`, `stock_code`		Var olan bir stok satırını günceller; **yeni satır açmaz**. `stock_code` verilmezse ürünün SKU satırı, o yoksa ürünün tek stok satırı kullanılır. Bilinmeyen `stock_code` (veya çok satırlı üründe eşleşmeyen SKU) → `422` (`details.error_code: "stock_code_not_found"`).
`kargo_grubu`	int	Kargo grubu

- Bu uçta `status` (aktif/pasif), `brand_id` ve kargo alanları (`shipping_*`) **işlenmez**. Durum için `bulk-update` , marka için `filters` (marka filtresinin `filter_id` + seçenek) veya POST `/products/bulk` kullanın.
- Cevap: `{"product_id": 10452, "message": "Product updated successfully"}` .
- Hatalar: `404 Product not found or access denied` · `400 Invalid JSON data` · `400 Invalid kargo_grubu` · `422 Failed to update product: ...` (kategori/görsel/filtre/stok kodu hataları, §4.4 notu) · `500` yalnız sunucu hatası.

```
curl -X PATCH $BASE/products/10452 -H "Authorization: Bearer $TOKEN" -H "Content-Type: applicat
ion/json" \
  -d '{"price": 699.90, "market_price": 899.90}'
```

5.4.5 Ürün sil — DELETE `/products/{id}`

Kalıcı silme: ürün, görselleri, stok satırları, varyant ve filtre kayıtları silinir; geri alınamaz. Cevap: `{"product_id": 10452}` , mesaj `Product permanently deleted successfully` .

> Siparişi olan ürünü silmeyin; sipariş detayında o kalem görünmez hâle gelir. Satıştan kaldırmak için `bulk-update` ile `status: 0` (pasif) yapın.

5.4.6 Toplu oluştur/güncelle (SKU ile upsert) — POST `/products/bulk`

En çok **100** ürün. Her satır `sku` ile aranır: satıcıda bu SKU varsa günceller, yoksa oluşturur. Her satır kendi transaction'ındadır.

Satır alanı	Zorunlu	Not
`sku`	evet	Eşleştirme anahtarı
`title`	evet	
`price`	evet	
`category_id`	yeni üründe evet	

Satır alanı	Zorunlu	Not
`description`, `barcode`, `market_price`, `images`, `variants`, `filters`, `brand_id`, `stock_quantity`, `kargo_grubu`	hayır	POST `/products` ile aynı anlam
`estimated_shipping_days`	hayır	Tahmini kargoya teslim günü (varsayılan 1)
`shipping_type`, `shipping_cost`, `shipping_company_id`	koşullu	Kendi kargolu satıcıda yeni üründe zorunlu kural

Güncelleme dalında dikkat:

- `title`, `price`, `market_price` her zaman yazılır (gönderdiğiniz değer). `market_price` göndermezseniz **0 yazılır**.
- `description`, `barcode`, `category_id` yalnız dolu gönderilirse değişir.
- `images` gönderilirse tüm görseller değişir; `filters` / `brand_id` gönderilirse tüm filtreler değişir.
- `variants` güncellemede mevcut varyant tanımlarını **silmez, ekler**. Var olan ürünün varyantlarını bu uçla göndermeyin; stok/fiyatı §5.7 ile yönetin.
- `stock_quantity` yalnız 0'dan büyükse işlenir ve ürün SKU'sunun stok satırına yazılır; SKU satırı yoksa ürünün tek stok satırına yazılır. Var olan üründe yeni stok satırı **açılmaz** (çok satırlı üründe eşleşme yoksa satır hatası `Stock code not found for this product ...`). Yeni üründe stok satırı her zaman açılır.
- Yeni ürünler `passive` + inceleme bekler olarak oluşur.
- Satır hata mesajları §5.4.3 hata tablosundakilerle aynıdır (marka, filtre, kategori, görsel, kargo); satır hatası başka satırları etkilemez.

Örnek:

```
{
  "products": [
    { "sku": "ATL-CNT-001", "title": "Deri Omuz Çantası", "price": 749.90, "market_price": 899.
90, "category_id": 899, "stock_quantity": 25 },
    { "sku": "ATL-CZD-002", "title": "Deri Cüzdan", "price": 349.90, "market_price": 399.90, "c
ategory_id": 905 }
  ]
}
```

Hepsi başarılıysa HTTP 200:

```
{
  "success": true,
  "data": {
    "message": "Bulk create completed",
    "results": {
      "created": 1, "updated": 1, "errors": 0, "total": 2,
      "details": [
        { "index": 1, "product_id": 10452, "sku": "ATL-CNT-001", "status": "success", "action":
"updated", "message": "Updated successfully" },
        { "index": 2, "product_id": 10460, "sku": "ATL-CZD-002", "status": "success", "action":
"created", "message": "Created successfully" }
      ]
    }
  }
}
```

En az bir satır hatalıysa HTTP **422**, `success:false` ve aynı sonuç `error.details` içinde gelir (başarılı satırlar yine işlenmiştir):


```
{
  "success": false,
  "error": {
    "message": "One or more products could not be processed",
    "code": "Unprocessable Entity",
    "details": {
      "message": "Bulk create completed",
      "results": { "created": 1, "updated": 0, "errors": 1, "total": 2,
        "details": [
          { "index": 1, "product_id": 10460, "sku": "ATL-CZD-002", "status": "success", "actio
n": "created", "message": "Created successfully" },
          { "index": 2, "sku": "ATL-XXX-003", "status": "error", "message": "category_id is req
uired for new products" }
        ] }
    }
  }
}
```


`index` 1'den başlar. İstek düzeyi hatalar: `400 Products array is required` · `400 Maximum 100 products allowed per request`.

5.4.7 Stok uyarıları — GET `/products/stock-alerts`

Sorgu	Varsayılan	Açıklama
`threshold`	10	Düşük stok eşiği
`type`	`all`	`all` (stok ≤ eşik), `low` (0 < stok ≤ eşik), `out` (stok ≤ 0)

`data.alerts[]`: `product_id`, `product_name`, `sku`, `stock_code`, `variant_name`, `price`, `stock_quantity`, `alert_type` (`out_of_stock` / `low_stock`), `last_updated`; `data.summary{total_alerts,out_of_stock,low_stock,low_stock_threshold}`; `data.filters`. Sayfasız.

5.4.8 Toplu alan güncelleme — POST `/products/bulk-update`

Ürün kimliğiyle toplu güncelleme. **Ürünü aktif/pasif yapmanın tek API yolu budur.**

Gövde	Tip	Zorunlu	Açıklama
`update_fields`	string[]	evet	Hangi alanlar işlensin: `product_name`, `price`, `market_price`, `category_id`, `description`, `status`, `images`
`products`	object[]	evet	Her öğe `product_id` + `update_fields`'taki alanlar

- `status`: `1` / `"active"` veya `0` / `"passive"`. Reddedilmiş ürünün durumu değiştirilemez; onaylanmamış ürün aktif yapılamaz.
- `price` / `market_price`: ürünün **tüm stok satırlarına** aynı değer yazılır.
- `images`: tüm görselleri değiştirir.
- `product_name`, `description`, `category_id` değişirse yeniden inceleme.
- Satır sınırı yoktur; yine de istek başına **en çok 100 ürün** göndermenizi öneririz.

- HTTP her zaman **200** döner; satır sonuçlarını `results.details[]` içinden okuyun (`index` 0'dan başlar).

```
{
  "update_fields": ["status"],
  "products": [ { "product_id": 10452, "status": 1 }, { "product_id": 10460, "status": 1 } ]
}
```

```
{
  "success": true,
  "data": {
    "message": "Bulk update completed",
    "results": {
      "success": 1, "errors": 1, "total": 2,
      "details": [
        { "index": 0, "product_id": 10452, "status": "success", "message": "Updated successfull
y" },
        { "index": 1, "product_id": 10460, "status": "error", "message": "Product is awaiting m
arketplace review; it can be activated after approval" }
      ]
    }
  }
}
```

Satır hata mesajları: `Product ID is required` · `Product not found or access denied` · `category_id not found: use a category id from GET /categories` · `status must be 1/"active" or 0/"passive"` · `Product was rejected by marketplace review; its status cannot be changed via API` · `Product is awaiting marketplace review; it can be activated after approval` · `No valid fields to update` . İstek düzeyi: `400 Invalid input. Expected products array` · `400 No fields to update specified` .

5.4.9 Toplu silme — POST `/products/bulk-delete`

```
{ "product_ids": [10452, 10460], "confirm": true }
```

Kalıcı silme (§5.4.5 uyarısı geçerli). `confirm: true` yoksa 400. HTTP 200; `results{success,errors,total,details[]{index,product_id,product_name,sku,status,message}}` .

5.5 Ürün varyantları

Varyant listesi — GET `/products/{id}/variants`

`data.variants[]` : `variant_id` , `stock_code` , `variant_name` , `price` , `market_price` , `stock_quantity` , `attributes{"Beden":"S"}` .

Varyantsız üründe (tek stok satırı) `variants` boş dizidir (`{"variants": []}`), ürünün stok satırı olsa bile. Stok kodunu GET `/products/{id}` → `stock[].sku` ya da GET `/products/{id}/stock` → `variants[].stock_code` alanından alın.

Varyant ekle — POST `/products/{id}/variants`

Gövde	Tip	Zorunlu	Açıklama
`attributes`	object	evet	`{"<varyant_id>": <seçenek_id>}`
`price`	number	evet	
`stock_quantity`	int	evet	
`market_price`	number	hayır	Yoksa `price`

Stok kodunu **sunucu üretir** (`SKU-<ürün_id>-xxxxxxxx`); kendi SKU'nuzu veremezsiniz. Kendi SKU'nuzla varyant için ürünü oluştururken `variants[]` kullanın. Cevap HTTP 200: `{"variant_id": 88131, "stock_code": "SKU-10452-5f1c2a9b", "message": "..."}` .

Varyant güncelle — PUT `/products/{id}/variants/{variant_id veya stok_kodu}`

Gövde: `price` , `market_price` , `stock_quantity` (gönderilenler değişir). Yolda sayısal değer varyant kimliği, değilse stok kodu kabul edilir. Cevap: `{"variant_id", "stock_code", "message"}` . Bulunamazsa 404.

```
curl -X PUT $BASE/products/10452/variants/ATL-CNT-001-S -H "Authorization: Bearer $TOKEN" -H "C
ontent-Type: application/json" \
  -d '{"price": 729.90, "stock_quantity": 8}'
```

> Stok kodunuz yalnız rakamdan oluşuyorsa yolda varyant kimliği sanılır; bu durumda PATCH `/stock/{product_id}` kullanın. Sistemin varyantsız ürünlere ürettiği 13 haneli sayısal stok kodları (ör. `1045207567019`) bu durumdadır.

Varyant sil — DELETE `/products/{id}/variants/{variant_id}`

Varyant tanımı ve stok satırı silinir. Yalnız sayısal `variant_id` ile.

5.6 Ürün görselleri

Metot ve yol	Amaç
GET `/products/{id}/images`	Görseller: `images[]` `{image_id,image_url,is_main,sort_order,created_at}` , `total_count` , `main_image`
POST `/products/{id}/images`	Dosya yükle (multipart)
PUT `/products/{id}/images/{image_id}`	`{"is_main": true}` ile kapak yap
DELETE `/products/{id}/images/{image_id}`	Görseli sil

Dosya yükleme:

```
curl -X POST $BASE/products/10452/images -H "Authorization: Bearer $TOKEN" \
  -F "image=@/yol/urun-1.jpg" -F "is_main=1"
```

- Alan adı `image`; JPEG/PNG/GIF/WebP (içerikten tespit edilir), en çok 5 MB.
- `is_main=1` kapak yapar. `sort_order` gönderilebilir ama **kaydedilmez**; sıra: kapak önce, sonra yüklenme sırası.
- Cevap: `{"image_id", "image_url", "is_main", "sort_order", "message"}`.
- Hatalar: `400 No image file uploaded or upload error` · `400 Invalid file type...` · `400 File size too large...`.

URL ile görsel için ürün oluşturma/güncellemede `images` alanını kullanın (§5.4.3).

5.7 Stok

Stok, ürünün **stok satırlarında** tutulur. Varyantsız ürünün tek satırı vardır (stok kodu ürün SKU'su, `DEFAULT` veya sistemin ürettiği sayısal kod, ör. `1045207567019`); varyantlı üründe her varyantın kendi stok kodu (SKU) vardır.

> **Stok uçları yeni stok satırı açmaz.** Stok satırı yalnız ürün oluşturulurken (POST `/products`, POST `/products/bulk` yeni ürün) ve varyant eklenirken (`variants[]`, POST `/products/{id}/variants`) açılır — satıcı paneliyle aynı. PATCH `/stock/{product_id}`, PATCH `/stock/bulk`, POST `/products/bulk-stock-update` ve `POST|PATCH /products/{id}/stock` uçlarında ürüne ait olmayan bir stok kodu (yazım hatası dahil) gönderirseniz satır açılmaz, hata döner (`stock_code_not_found`). Eşleştirme kuralı:
>
> 1. Ürünün bu stok koduyla satırı varsa o satır güncellenir.
> 2. Yoksa ve ürünün **tek** stok satırı varsa: stok kodu boş/`DEFAULT` gönderildiyse o satır güncellenir; satırın kodu `DEFAULT` ise satır gönderdiğiniz koda taşınır (eski davranış, SKU geçişi için).
> 3. Diğer her durumda (çok satırlı ürün, satırsız ürün, kodu farklı tek satır) → hata.
>
> Stok kodlarını daima GET `/products/{id}` veya GET `/stock/{id}` cevabından alın. Stok kodlarında `& < > " '` karakterlerini kullanmayın (bazı uçlarda HTML kaçışına uğrar ve eşleşmez).

Stok değeri 0'ın altına inmez.

Stok listesi — GET `/stock`

Sorgu	Açıklama
`page`, `per_page`	Sayfalama
`level`	`low` (eşik altı), `out` (0), `normal` (eşik ve üstü); eşik satıcının paneldeki stok uyarı tabanı
`search`	Ürün adında arama

`data.stock_items[]`: `product_id`, `product_name`, `product_sku`, `image` (kapak görselinin tam URL'si, `https://www.esnafpazar.com/dosya/urun/<dosya>`; görsel yoksa `null`), `status` (`normal` / `low_stock` / `out_of_stock`), `stock_variants[]{sku,price,quantity,updated_at}`, `total_quantity`, `average_price`, `min_quantity`, `threshold`, `product_status` (`active` / `inactive`); `data.summary`, `data.threshold`, `data.meta`.

```
{
  "product_id": 10452, "product_name": "Deri Omuz Çantası", "product_sku": "ATL-CNT-001",
  "image": "https://www.esnafpazar.com/dosya/urun/10452-6743a1b2c3d4e.webp",
  "status": "normal",
  "stock_variants": [ { "sku": "1045207567019", "price": 749.9, "quantity": 4, "updated_at": "1
795522476" } ],
  "total_quantity": 4, "average_price": 749.9, "min_quantity": 4, "threshold": 0, "product_stat
us": "inactive"
}
```


`threshold` satıcının paneldeki stok uyarı tabanıdır (ayarlanmamışsa `0`).

Ürün stoğu — GET `/stock/{product_id}`

`data`: `product_id`, `stock_variants[]{sku,price,quantity,created_at,updated_at}`, `total_quantity`. (`product_name`, `product_sku`, `image` bu uçta boş döner; ürün bilgisi için GET `/products/{id}`.)

Daha ayrıntılı alternatif: GET `/products/{id}/stock` → `product{product_id,product_name,sku,barcode,price,market_price,total_stock,variant_count}`, `variants[]{stock_code,variant_name,price,market_price,stock_quantity,attributes,created_at,updated_at}`, `alerts`, `summary`.


```
{
  "product": { "product_id": 10452, "product_name": "Deri Omuz Çantası", "sku": "ATL-CNT-001",
"barcode": "8690000000017",
              "price": 749.9, "market_price": 0, "total_stock": 4, "variant_count": 1 },
  "variants": [ { "stock_code": "1045207567019", "variant_name": "Deri Omuz Çantası", "price":
749.9, "market_price": 749.9,
                  "stock_quantity": 4, "attributes": [], "created_at": "1795521909", "updated_a
t": "1795522476" } ],
  "alerts": { "out_of_stock": 0, "low_stock": 1, "low_stock_threshold": 10 },
  "summary": { "total_variants": 1, "in_stock_variants": 1, "out_of_stock_variants": 0, "low_st
ock_variants": 1 }
}
```


- `product.market_price` ürün kartındaki değerdir (girilmemişse `0`); `variants[].market_price` stok satırının değeridir.
- `alerts` ve `summary` bu uçta **sabit eşik 10** ile hesaplanır. Paneldeki eşiğinize göre durum için GET `/stock` → `threshold` / `status` kullanın. Bu yüzden aynı ürün burada `low_stock: 1`, `/stock`'ta `status: "normal"` görünebilir.

Tekil stok (+fiyat) güncelle — PATCH `/stock/{product_id}`

Gövde	Tip	Açıklama
`stock_variants`	object[]	`[{"sku": "ATL-CNT-001-S", "quantity": 8, "price": 729.90}]` — `price` isteğe bağlı
`quantity` + `stock_code`	int + string	Tek satır (stok kodu verilmezse `DEFAULT` = ürünün tek satırı)

```
curl -X PATCH $BASE/stock/10452 -H "Authorization: Bearer $TOKEN" -H "Content-Type: applicatio
n/json" \
  -d '{"stock_variants":[{"sku":"ATL-CNT-001-S","quantity":8},{"sku":"ATL-CNT-001-M","quantit
y":0}]}'
```

Cevap: `{"product_id": 10452, "message": "Stock updated successfully"}`. Tek transaction: bir satır hata verirse hiçbiri yazılmaz.

Hatalar: `404 Product not found or access denied` · `400 Invalid JSON data` · `422` bilinmeyen stok kodu (hiçbir satır yazılmaz):

```
{ "success": false, "error": { "message": "Failed to update stock: Stock code not found for thi
s product: ATL-CNT-001-XL (stock rows are created only by creating the product or a variant)",
  "code": "Unprocessable Entity", "details": { "field": "stock_variants", "stock_code": "ATL-CN
T-001-XL", "error_code": "stock_code_not_found" } } }
```

Çok satırlı üründe `stock_code` vermeden `quantity` gönderirseniz: `422 ... stock_code is required: this product has no single default stock row ...`.

Alternatif: `POST` veya **PATCH** `/products/{id}/stock` — `main_stock` (yalnız tek satırlı üründe; çok satırlı üründe `422`, `details.field: "main_stock"`), `variants[]{stock_code,stock_quantity,price?,market_price?}`, `bulk_update[]{stock_code,stock_quantity,operation:set|add|subtract}`. Bilinmeyen stok kodu → `422` (`details.error_code: "stock_code_not_found"`), tek transaction.

Toplu stok — PATCH `/stock/bulk`

```
{ "stock_updates": [ { "product_id": 10452, "stock_code": "ATL-CNT-001-S", "quantity": 8 }, {
"product_id": 10460, "quantity": 40 } ] }
```

- `stock_code` yoksa `DEFAULT`. Satır sınırı yok; istek başına **en çok 500 satır** öneririz.
- Cevap alanları: `total_items`, `successful_updates`, `failed_updates`, `errors[]` (metin: `"Item 1: Product not found or access denied"`), `items[]` (satır bazlı sonuç), `message`.
- `items[]` öğesi: `index` (0'dan), `product_id`, `stock_code`, `status` (`success` / `error`); başarılıda `new_quantity`, hatalıda `error_code` (`stock_code_not_found`, `product_not_found`, `validation_error`, `internal_error`) ve `message`.
- Bilinmeyen stok kodlu satır yazılmaz, `stock_code_not_found` ile hatalı sayılır; diğer satırlar işlenir.
- Hepsi başarılı → 200 `success:true` (`data` içinde). Bazıları hatalı → 200 `success:false` (sonuç `error.details` içinde). Hepsi hatalı → 422.
- `400 stock_updates array is required`: gövdede `stock_updates` dizisi yok ya da boş (ör. alan adı `items` gönderilmiş). Hiçbir satır işlenmez.

Toplu stok (işlem tipli) — POST `/products/bulk-stock-update`

```
{ "operation": "set", "stock_updates": [ { "product_id": 10452, "stock_code": "ATL-CNT-001-S",
"quantity": 8 } ] }
```

- `operation`: `set` (varsayılan), `add`, `subtract`; başka değer `set` gibi davranır. `stock_code` zorunlu.

- Cevap: `results{success,errors,total,details[]{index,product_id,stock_code,old_quantity,new_quantity,operation,status,message}}`, `operation`. Hatalı satırda ayrıca `error_code`: `stock_code_not_found` (stok kodu bu üründe yok, satır açılmadı), `product_not_found`, `validation_error`, `internal_error`. Başarı/hata zarfı PATCH `/stock/bulk` ile aynı mantıkta (kısmi hata → 200 `success:false`, tümü hata → 422).

5.8 Fiyat

Varyant (SKU) bazında **toplu fiyat** ucu yoktur. Seçenekler:

İhtiyaç	Uç
Birçok ürünün fiyatı (ürünün tüm varyantlarına aynı fiyat)	POST `/products/bulk-update` + `update_fields:["price","market_price"]`
Tek ürünün fiyatı (tüm varyantlara)	PATCH `/products/{id}` `{"price":..., "market_price":...}`
Bir ürünün varyant fiyatları (tek istekte birden çok SKU)	PATCH `/stock/{product_id}` `stock_variants[].price` veya POST `/products/{id}/stock` `variants[].price/market_price`
Tek varyant	PUT `/products/{id}/variants/{sku}`

```
{
  "update_fields": ["price", "market_price"],
  "products": [ { "product_id": 10452, "price": 699.90, "market_price": 899.90 } ]
}
```

Fiyatlar KDV dahil satış fiyatıdır. `market_price` satış fiyatından küçük olmamalıdır (API zorlamaz).

5.9 Siparişler

Siparişler yalnız müşterinin EsnafPazar'da ödeme yapmasıyla oluşur. **API ile sipariş oluşturulamaz** (POST `/orders` → 405). Bir siparişte birden çok satıcının ürünü olabilir; API size yalnız **kendi kalemlerinizi** ve kendi satıcı durumunuzu gösterir. `order_id` metindir ve tire içerir (ör. `"812-4471093"`); sayıya çevirmeyin, URL yolunda olduğu gibi kullanın (`/orders/812-4471093`). `order_number` aynı değerdir.

5.9.1 Sipariş listesi — GET `/orders`

Sorgu	Tip	Açıklama
`page`, `per_page`	int	Sayfalama
`status`	string	Durum adı (§6), ör. `pending`, `approved`, `shipped`, `cancelled`. Sayısal kod kabul edilmez.
`created_after`, `created_before`	tarih	Sipariş oluşturma zamanı
`updated_after`	tarih	Satıcı tarafında son değişiklik zamanı (yeni + güncellenen siparişleri yakalamak için kullanın)

Sıralama: en yeni sipariş önce. Listede **kalemler yoktur**; kalemler için detay çağrılır.


```
{
  "success": true,
  "data": {
    "orders": [
      {
        "order_id": "812-4471093", "order_number": "812-4471093", "status": "pending",
        "customer": { "name": "Ayşe Yılmaz", "email": "ayse@ornek.com", "phone": "5551112233",
                      "tc_no": null, "tax_office": null, "tax_number": null, "company_name": nu
ll },
        "shipping_address": { "address": "Atatürk Cad. No:1 D:2", "district": "Kadıköy", "cit
y": "İstanbul", "country": "Türkiye" },
        "totals": { "subtotal": 1499.8, "shipping": 0, "discount": 0, "grand_total": 1499.8, "c
urrency": "TRY" },
        "shipping_detail": {
          "kargo_secim": 2,
          "seller_contract": { "carrier_shipping_total": 0, "order_aggregate_in_siparis": 0 },
          "platform": { "system_shipping_fee_in_siparis": 0 },
          "support_carrier_deduction": 0,
          "geliver": null
        },
        "item_count": 2,
        "created_at": "1795521909",
        "updated_at": "1795523349"
      }
    ],
    "meta": { "current_page": 1, "total_pages": 1, "per_page": 50, "total_items": 1, "has_nex
t": false, "has_prev": false }
  }
}
```


- `totals.subtotal` sizin kalemlerinizin toplamıdır; `discount` ve `grand_total` siparişin **tamamına** aittir (çok satıcılı siparişte diğer satıcıların tutarını da içerir).
- `shipping_detail.kargo_secim` : `1` kendi kargo anlaşmanız, `2` platform (EsnafPazar) kargosu.
- `totals.shipping` satıcı kargo anlaşması / sistem kargo ücretidir. **Geliver ile fiyatlanan siparişte** müşterinin ödediği kargo burada görünmez; GET `/orders/{id}` → `shipping.geliver.customer_paid_shipping` alanını okuyun. Bu yüzden `grand_total` , `subtotal + shipping − discount` ile tutmayabilir (ör. `subtotal: 10` , `shipping: 0` , `grand_total: 125.85`).
- `updated_at` : satıcı siparişinin son işlem zamanı (unix, metin). `updated_after` filtresi de aynı alana bakar; yoklama döngüsünü bu değerle kurabilirsiniz.
- `shipping_detail.geliver` listede her zaman `null` ; Geliver bilgisini detayda okuyun.
- Kurumsal faturada `tax_office` , `tax_number` , `company_name` ; bireyselde `tc_no` dolu olabilir.

5.9.2 Sipariş detayı — GET `/orders/{order_id}`


```
{
  "success": true,
  "data": {
    "order_id": "812-4471093", "order_number": "812-4471093", "status": "approved",
    "created_at": "1795521909", "updated_at": "1795525509",
    "customer": { "name": "Ayşe Yılmaz", "email": "ayse@ornek.com", "phone": "5551112233", "tc_
no": null, "tax_office": null, "tax_number": null, "company_name": null },
    "shipping_address": { "name": "Ayşe Yılmaz", "address": "Atatürk Cad. No:1 D:2", "distric
t": "Kadıköy", "city": "İstanbul", "country": "Türkiye", "phone": "5551112233" },
    "billing_address": { "name": "Ayşe Yılmaz", "address": "Atatürk Cad. No:1 D:2", "district":
"Kadıköy", "city": "İstanbul", "country": "Türkiye", "phone": "5551112233" },
    "items": [
      {
        "product_id": 10452, "sku": "ATL-CNT-001-S", "title": "Deri Omuz Çantası", "barcode":
"8690000000017",
        "quantity": 2, "price": 899.9, "discount_price": 749.9, "vat_rate": 20, "total": 1499.
8,
        "image": "https://www.esnafpazar.com/dosya/urun/10452-6743a1b2c3d4e.webp",
        "shipping": { "kargo_secim": 2, "cost": 0, "requires_payment": false, "fee_refunded": f
alse, "company": { "id": 3, "name": "Yurtiçi Kargo" } }
      }
    ],
    "shipping": {
      "requires_payment": false, "cost": 0, "kargo_secim": 2, "mode": "platform", "estimated_de
livery_days": 1,
      "company": { "id": 3, "name": "Yurtiçi Kargo" },
      "order_level": { "seller_contract_aggregate": 0, "platform_fee_in_siparis": 0 },
      "support_carrier_deduction": 0,
      "method": "Standard Shipping",
      "tracking_number": "",
      "carrier_note": "",
      "geliver": null
    },
    "invoice": { "url": null, "number": null, "date": null, "has_uploaded_file": false },
    "payment": { "status": "paid", "method": "-", "total": 1499.8, "currency": "TRY" }
  }
}
```


- Kalemde `discount_price` müşterinin ödediği birim fiyat, `total = discount_price × quantity`. Kalemde indirim yoksa `discount_price` = `price` ve `total` = `price × quantity` olur (0 dönmez).
- Kalem `image`: ürün görselinin tam URL'si (`https://www.esnafpazar.com/dosya/urun/<dosya>`), yoksa `null`.
- `payment.method` serbest metindir; değer gelmezse `"-"` döner. Ödeme yöntemine göre dallanmayın.
- `shipping.company.id` Geliver siparişinde `null` olabilir (`{"id": null, "name": "Geliver"}`); kalem düzeyindeki `shipping.company` de `{"id": null, "name": null}` gelebilir.
- `vat_rate` sabit 20 döner; gerçek KDV oranını kendi ürün kartınızdan alın.
- `billing_address` teslimat adresiyle aynıdır; fatura kimliği `customer` altındadır.
- `payment.status` her zaman `paid` (siparişler ödeme sonrası oluşur).
- `shipping.mode`: `platform` veya `seller_contract`. Geliver ile gönderilen siparişte `shipping.geliver` doludur (§5.11).
- Başka satıcının siparişi → 404 `Order not found`.

5.9.3 Durum güncelle — PATCH `/orders/{order_id}/status`

Gövde	Tip	Zorunlu	Açıklama
`status`	string veya int	evet	`approved` (1), `supplying` (2), `preparing` (3), `packaging` (4), `shipped` (5), `delivered` (6). Sayısal kod da olur.
`note`	string	hayır	Kayıt notu

```
curl -X PATCH $BASE/orders/812-4471093/status -H "Authorization: Bearer $TOKEN" -H "Content-Type: application/json" \
  -d '{"status":"preparing"}'
```

Cevap: `{"order_id": "812-4471093", "status": "preparing", "message": "Order status updated successfully"}`.

- Geçiş kuralı §6'daki tabloya bağlıdır. Onay bekleyen (0) siparişe doğrudan `preparing`, `shipped` gibi ileri adım gönderebilirsiniz (onay örtük kabul edilir).
- Takip numaralı kargo bildirimi için `shipped` yerine POST `/orders/{id}/ship` kullanın.
- Hatalar: `400 Status is required` · `400 Invalid status` (iade/kısmi kodları ve `pending` bu uçta kabul edilmez) · `404` · `409 Use POST /orders/{id}/cancel to cancel an order ...` (`cancelled` gönderildiğinde) · `409 Status transition not allowed: X -> Y`.
- Başarılı değişiklik `order.updated` webhook'unu tetikler.

5.9.4 Kargoya ver (takip numarası) — POST `/orders/{order_id}/ship`

Gövde	Tip	Zorunlu	Açıklama
`tracking_number`	string	evet	Kargo takip numarası
`carrier`	string	evet	Kargo firması adı (ör. `"Yurtiçi Kargo"`, boşluk/harf farkı önemsiz) veya kargo firması kimliği

```
curl -X POST $BASE/orders/812-4471093/ship -H "Authorization: Bearer $TOKEN" -H "Content-Type: application/json" \
  -d '{"tracking_number":"1234567890","carrier":"Yurtiçi Kargo"}'
```

```
{ "success": true, "data": { "order_id": "812-4471093", "tracking_number": "1234567890", "carrier": "Yurtiçi Kargo", "carrier_id": 3, "status": "shipped", "message": "Order marked as shipped" } }
```

- Sipariş durumu 5 (Kargoda) olur; müşteriye bildirim gider; `order.updated` webhook'u tetiklenir.
- Yalnız 0–4 durumundaki (ve aynı durumdaki 5) siparişte çalışır; iptal/teslim edilmiş siparişte `409`.
- Firma adı tanınmazsa gönderdiğiniz metin aynen kaydedilir ve `carrier_id: null` döner.
- **Geliver siparişinde** bu uç otomatik olarak "kendi anlaşmamla gönderdim" (`ship-own`) akışına yönlenir; etiket zaten alınmışsa `409`. Bkz. §5.11.

5.9.5 Takip numarası — `GET` / `PATCH` `/orders/{order_id}/tracking`

- `GET` : `tracking_number` , `status` (durum adı), `delivered_at` (panelin yazdığı teslim zamanı); Geliver siparişinde ek olarak `carrier` , `tracking_url` , `tracking_status` , `events[]` .
- `PATCH` `{"tracking_number": "..."}` : yalnız takip numarasını düzeltir; durum değişmez, webhook tetiklenmez.

5.9.6 Kısmi gönderim — `POST` `/orders/{order_id}/partial-ship`

Bu uç kalem bazında gerçek bir kısmi gönderim **oluşturmaz**. Siparişin satıcı durumunu panelin "Kısmi Gönderim (iadeli)" (11) koduna çeker ve takip numarasını yazar. Panel kuralına göre yalnız **teslim edilmiş (6), iade bekleniyor (7) veya iade geldi (8)** durumundaki siparişte kullanılabilir.

Gövde	Tip	Zorunlu	Açıklama
`items`	array	evet	Kayıt amaçlı (loga yazılır, stok/kalem değiştirmez)
`tracking_number`	string	hayır	
`carrier`	string	hayır	Yalnız loga yazılır
`note`	string	hayır	Durum açıklamasına eklenir

HTTP 201: `{"order_id","tracking_number","carrier","status":"partial_shipment","message"}` . Kural dışı durumda 409.

- `GET` `/orders/{id}/partial-shipments` : Sipariş 11 veya 12 durumundaysa tek kayıt döner (`tracking_number` , `status` , `note` , `created_at` , `updated_at`).
- `PATCH` `/orders/{id}/partial-shipments/{n}` : `tracking_number` ve/veya `note` günceller (`n` sayısal olmalı, ör. `1` ; değeri yok sayılır).

Kalem bazında eksik gönderim yapacaksanız gönderemeyeceğiniz ürünler için siparişi kargoya vermeden önce EsnafPazar satıcı panelinden kısmi iptal yapın (API'de kalem bazlı iptal yoktur).

5.9.7 İptal — `POST` `/orders/{order_id}/cancel`

Satıcının kendi kalemlerinin **tamamını** iptal eder (panelle aynı akış): durum 10 olur, stok iade edilir, müşteriye ve yönetime bildirim gider. Para iadesini EsnafPazar yapar.

Gövde	Tip	Zorunlu	Açıklama
`reason`	string	hayır	Şu listeden biri: `Ürün stokta yok` , `Ürün hasarlı / kusurlu` , `Fiyat ya da ürün bilgisi hatalı` , `Müşteri iptal istedi` , `Teslimat adresine gönderim yapılamıyor` , `Diğer` . Listede yoksa `Diğer` kaydedilir ve gönderdiğiniz metin açıklamaya eklenir.
`note`	string	hayır	Açıklama (reason ile birlikte en çok 255 karakter)

```
curl -X POST $BASE/orders/812-4471093/cancel -H "Authorization: Bearer $TOKEN" -H "Content-Type: application/json" \
  -d '{"reason":"Ürün stokta yok","note":"Tedarikçi teslim edemedi"}'
```

```
{ "success": true, "data": { "order_id": "812-4471093", "status": "cancelled", "refund_amount": 1499.8, "message": "Order cancelled successfully. Stock restored; the refund is completed by the marketplace." } }
```


- Yalnız kargoya verilmeden önce (0–4). Geliver etiketi alınmış ama paket henüz kargo şubesine verilmemişse iptal yine mümkündür.
- Hatalar: `404` · `409 Order can only be cancelled before it is shipped` · `409 Order is already cancelled` · `500 Failed to cancel order`.
- `order.cancelled` webhook'u tetiklenir (`source: "api"`).

5.9.8 Fatura — `GET` / `POST` / `PATCH` `/orders/{order_id}/invoice`

E-faturayı kendi sisteminizde kesip linkini ve numarasını siparişe yazarsınız (dosya yükleme yoktur). `POST` ve `PATCH` aynıdır.

Gövde	Tip	Zorunlu	Açıklama
`invoice_url`	string	biri zorunlu	`https://` ile başlayan geçerli URL, ≤ 500 karakter
`invoice_number`	string	biri zorunlu	≤ 64 karakter
`invoice_date`	string/int	hayır	ISO 8601 veya unix; yoksa şimdiki zaman

```
curl -X POST $BASE/orders/812-4471093/invoice -H "Authorization: Bearer $TOKEN" -H "Content-Type: application/json" \
  -d '{"invoice_url":"https://efatura.ornek.com/f/ABC2026000123.pdf","invoice_number":"ABC2026000123","invoice_date":"2026-11-24T15:00:00+03:00"}'
```

Cevap: `{"order_id", "invoice": {"url", "number", "date"}}`, mesaj `Invoice saved`. `order.updated` webhook'u tetiklenir. `GET` aynı `invoice` nesnesini ve `has_uploaded_file` (satıcının panelden dosya yükleyip yüklemediği) döner.

5.9.9 Sipariş notu — `POST` `/orders/{order_id}/notes`

`{"note": "Hediye paketi istendi"}` → satıcının sipariş notuna tarihli satır olarak eklenir. Cevap: `{"message": "Order note added successfully"}`.

5.10 İadeler

Müşteri iadesi EsnafPazar üzerinde başlatılır (müşteri iade talebi oluşturur, sipariş "iade bekleniyor" durumuna geçer). Entegratör iadeleri sipariş bazında izler ve sonraki adımları bildirir. Satıcının tüm iadelerini tek seferde listeleyen uç yoktur; `GET` `/orders?status=return_pending` ile iade bekleyen siparişleri bulun.

İade kalemleri — `GET` `/orders/{order_id}/returns`

`data.returns[]` : `item_id`, `product_id`, `product_name`, `quantity` (iade adedi), `return_date`, `status` (`return_pending`, `return_received`, `return_completed`; diğer durumlarda `normal` / `unknown`), `reason`, `updated_at`. Yalnız iade adedi > 0 olan kalemler döner.

İade durumunu ilerlet — PATCH `/orders/{order_id}/returns/{return_id}`

`return_id` şu an yok sayılır (iade sipariş düzeyinde izlenir); herhangi bir değer, ör. kalemin `item_id`'si gönderilebilir.

Gövde `status`	Sipariş kodu	Ne zaman
`pending`	7 İade bekleniyor	Teslim edilmiş (6) siparişte iade beklemeye al
`approved`	— / 8	Siparişte müşteri iade talebi varsa **iade onayı** (aşağıdaki bölüm, `return_code` aynı kuralla). İade talebi kaydı yoksa (eski iade) eski davranış: 8 İade geldi (7'den).
`completed`	9 Tamamlandı (iadeli)	İade süreci kapandı (6, 7, 8 veya 11'den)

`note` (isteğe bağlı) açıklamaya eklenir. Cevap: `{"order_id", "status", "message"}`. Kural dışı geçişte 409 (§6). İade onayı ve reddi için aşağıdaki bölümlere bakın; bu iki durumda `return_id` kullanılır.

İadeyi onayla — `POST` (veya `PUT`) `/orders/{order_id}/returns/{return_id}/approve`

(aynı davranış: PATCH `/orders/{order_id}/returns/{return_id}` gövdesinde `"status": "approved"`)

Müşterinin iade talebini satıcı panelindeki **"İadeyi kabul et"** ile birebir aynı şekilde kabul eder. Yalnız talep sizin cevabınızı beklerken (`pending_seller_response`) kullanılabilir. Müşteriye bildirim (uygulama + e-posta) gider.

- **Sipariş EsnafPazar (Geliver) etiketiyle gönderildiyse:** gövde gerekmez. Müşteri iade kargo etiketini sipariş sayfasından kendisi alır; etiket ücreti kargo hesabınızdan düşülür. Sipariş durumu (`status_code`) değişmez.
- **Diğer siparişlerde:** `return_code` **zorunlu** — kargo firmanızdan aldığınız iade kodu; müşteri ürünü bu kodla kargoya verir. Sipariş 6 → 7 (İade bekleniyor) olur ve kod siparişe yazılır; satıcı ödemeniz iade sonuçlanana kadar bekletilir (panelle aynı).

`return_id`: GET `/orders/{order_id}/returns` cevabındaki `data.return_request.id` (kalemin `item_id`'si veya `current` de kabul edilir).

Alan	Tip	Zorunlu	Açıklama
`return_code`	string	Geliver etiketsiz siparişte evet	İade kargo kodu, en fazla 100 karakter. (`iade_kodu` eş anlamlı kabul edilir.) Geliver etiketli siparişte gönderilse de yok sayılır.

```
{ "return_code": "YK-123456789" }
```

200 OK:

```
{
  "success": true,
  "data": {
    "order_id": "812-4471093",
    "return_id": 42,
    "status": "approved",
    "return_request_status": "approved",
    "return_shipping": "return_code",
    "return_code": "YK-123456789",
    "carrier": null,
    "responded_at": 1795537200,
    "message": "Return request approved. The customer has been notified with the return code."
  }
}
```


Geliver etiketli siparişte `return_shipping: "geliver_label"`, `return_code: null`, `carrier`: siparişin kargo firması adı.

Hatalar:

HTTP	Ne zaman	details
401	Token yok ya da geçersiz	—
404	Sipariş size ait değil / yok ya da `return_id` bu siparişin iade talebiyle eşleşmiyor	`return_request_id`
409	Talep artık cevap beklemiyor (onaylanmış, reddedilmiş, itirazda, otomatik onaylanmış…) veya arada durumu değişti	`return_request_id`, `return_request_status`, `allowed_from: ["pending_seller_response"]`
422	Geliver etiketsiz siparişte `return_code` yok ya da 100 karakterden uzun	`return_request_id`, `return_code_required`, `geliver_label`
429	İstek sınırı aşıldı	—
500	Sunucu hatası (ham hata metni yok, yalnız `request_id`)	—

> **Eski iadeler:** Siparişte müşteri iade talebi kaydı yoksa (ör. iade **POST** `/orders/{order_id}/return` ile API'den açılmışsa) bu uç eski davranışı sürdürür: gövdesiz, siparişi 8 (İade geldi) yapar, yalnız 6 veya 7 durumundan; kural dışı geçişte 409. Cevap: `{"order_id", "status": "approved", "message"}`. Bu durumda müşteriye bildirim gitmez.
> İadenin para ve stok tarafı EsnafPazar tarafından yürütülür.

Satıcı adına iade kaydı — **POST** `/orders/{order_id}/return`

Müşteri iadesini panel dışında (ör. elden) aldıysanız kalemlere iade adedi yazar ve siparişi 7'ye çeker. Yalnız **teslim edilmiş (6)** siparişte.


```
{ "items": [ { "product_id": 10452, "quantity": 1, "reason": "Beden uymadı" } ], "reason": "Müşteri iadesi", "description": "" }
```

Cevap: `{"order_id", "status": "return_pending", "returned_items": [...]}`. Hatalar: `400 Return items are required` · `400 ... Product 10452 not found in order` · `400 ... Return quantity exceeds order quantity ...` · `409` (durum uygun değil).

İade reddi

POST `/api/v1/orders/{order_id}/returns/{return_id}/reject` (aynı davranış: **PUT** `…/reject` ya da **PATCH** `/api/v1/orders/{order_id}/returns/{return_id}` gövdesinde `"status": "rejected"`)

Müşterinin iade talebini satıcı paneliyle **birebir aynı** şekilde reddeder. Yalnız talep sizin cevabınızı beklerken (`pending_seller_response`) kullanılabilir; talep açıldıktan sonra 3 gün içinde cevap verilmezse talep otomatik onaylanır. Siparişin durumu (`status_code`) değişmez. Müşteriye bildirim (uygulama + e-posta) gider; müşteri neden ve açıklamayı görür, EsnafPazar'a itiraz edebilir, son kararı EsnafPazar verir.

`return_id`: **GET** `/orders/{order_id}/returns` cevabındaki `data.return_request.id`. İade edilen kalemin `item_id` değeri ya da `current` kelimesi de kabul edilir.

Başlıklar: `Authorization: Bearer <token>`, `Content-Type: application/json`

Alan	Tip	Zorunlu	Açıklama
`reason`	string	evet	`used` (ürün kullanılmış) · `not_our_product` (gönderdiğimiz ürün değil) · `missing_or_damaged` (eksik/hasarlı) · `other` (diğer). Paneldeki Türkçe metnin birebiri de kabul edilir.
`description`	string	evet	En fazla 400 karakter. Müşteriye gösterilir. (`note` eş anlamlı kabul edilir.)

```
{ "reason": "used", "description": "Ürün kullanılmış, etiketi sökülmüş olarak gönderilmiş." }
```

200 OK:

```
{
  "success": true,
  "timestamp": "2026-11-24 18:40:00",
  "request_id": "…",
  "data": {
    "order_id": "812-4471093",
    "return_id": 42,
    "status": "rejected",
    "return_request_status": "rejected",
    "reason_code": "used",
    "reason": "Ürün kullanılmış",
    "description": "Ürün kullanılmış, etiketi sökülmüş olarak gönderilmiş.",
    "rejection_text": "Ürün kullanılmış: Ürün kullanılmış, etiketi sökülmüş olarak gönderilmi
ş.",
    "responded_at": 1795537200,
    "customer_can_object": true,
    "message": "Return request rejected. The customer has been notified, sees the reason and ca
n object to EsnafPazar."
  }
}
```

Hatalar (`error.message`, `error.code`, `error.details`):

HTTP	Ne zaman	details
401	Token yok ya da geçersiz	—
404	Sipariş size ait değil / yok ya da `return_id` bu siparişin iade talebiyle eşleşmiyor	`return_request_id`
409	Siparişte iade talebi yok	`return_request_status: null`, `allowed_from`
409	Talep artık cevap beklemiyor (onaylanmış, reddedilmiş, itirazda, otomatik onaylanmış…)	`return_request_id`, `return_request_status`, `allowed_from: ["pending_seller_response"]`
422	`reason` yok ya da listede değil	`allowed_reasons`
422	`description` boş ya da 400 karakterden uzun	—
429	İstek sınırı aşıldı	—
500	Sunucu hatası (ham hata metni yok, yalnız `request_id`)	—

`return_request_status` değerleri: `pending_seller_response` (satıcı cevabı bekleniyor), `approved`, `rejected`, `under_objection_review` (itiraz inceleniyor), `in_return_shipping` (iade kargoda), `received_by_seller` (satıcıya ulaştı), `seller_reported_issue` (satıcı sorun bildirdi), `refund_in_progress` (para iadesi yapılıyor), `completed`, `cancelled`.

GET `/api/v1/orders/{order_id}/returns` cevabı artık şu alanı da içerir:


```
"return_request": { "id": 42, "status": "pending_seller_response", "answer_deadline": 179579640
0, "rejection_text": "" }
```


Siparişte müşteri iade talebi yoksa `return_request` `null` döner; iade kalemi yoksa `returns` boş dizidir:


```
{ "success": true, "timestamp": "2026-11-24 14:05:09", "request_id": "req_6743a1b2c3d4e5.123456
78",
  "data": { "returns": [], "return_request": null } }
```


> Not: POST `/orders/{order_id}/return` ile API'den açılan iadeler iade talebi kaydı oluşturmaz; bu tür iadeler için ret uygulanamaz (409), onay ise eski davranışla çalışır (yukarıdaki "Eski iadeler" notu).

5.11 Kargo

Kargo modelleri

kargo_secim	Model	Entegratör ne yapar
1	Satıcının kendi kargo anlaşması	Ürünlerde `shipping_company_id` zorunlu; siparişi kendi kargonuzla gönderip POST `/orders/{id}/ship` ile takip no bildirirsiniz.

kargo_secim	Model	Entegratör ne yapar
2	Platform kargosu	Takip no bildirimi aynı. Sipariş **Geliver** ile fiyatlandıysa etiketi API'den alabilirsiniz (aşağıda).

Kargo firmaları — GET `/shipping/companies`

Kendi kargo anlaşmasıyla gönderen (`kargo_secim=1`) satıcının ürünlerinde zorunlu olan `shipping_company_id` değerlerinin listesi. Satıcı panelindeki ürün formunun "Kargo Firması" listesiyle aynıdır (firmaları EsnafPazar yönetimi ekler). POST `/orders/{id}/ship` gövdesindeki `carrier` alanına da bu kimlik ya da ad verilebilir. Parametre almaz; sayfasızdır.

```
curl $BASE/shipping/companies -H "Authorization: Bearer $TOKEN"
```


```
{
  "success": true,
  "data": {
    "own_shipping": true,
    "shipping_company_id_required": true,
    "companies": [
      { "id": 3, "name": "Yurtiçi Kargo", "logo_url": "https://www.esnafpazar.com/dosya/kargofir/1234-yurtici.webp", "sort_order": 1 },
      { "id": 5, "name": "Aras Kargo", "logo_url": "https://www.esnafpazar.com/dosya/kargofir/5678-aras.webp", "sort_order": 2 }
    ]
  }
}
```


- `own_shipping` / `shipping_company_id_required` : satıcı kendi kargolu ise `true` (platform kargolu satıcıda `shipping_company_id` gönderilmez, kargo alanları yok sayılır).
- `companies[]` : `id` (`shipping_company_id` olarak gönderin), `name` , `logo_url` (yoksa `null`), `sort_order` (paneldeki sıra). Liste bu sıraya göre gelir.
- Firmalar için aktif/pasif ayrımı ve takip URL şablonu tutulmaz; listedeki tüm firmalar seçilebilir.
- Hatalar: `401` · `405` (GET dışı metot) · `500` sunucu hatası.

Kargo grupları (`kargo_grubu`)

Ürünün paket büyüklüğü sınıfı (platform kargo ücretlendirmesinde kullanılır). Verilmezse kategorinin grubu atanır.

Değer	Grup	Desi
1	Zarf	1
2	Küçük	2
3	Orta	5
4	Büyük	10
5	Çok Büyük	20
6	Ağır / Hacimli	30

Açıkça gönderilen grup "satıcı seçti" olarak işaretlenir; kategorinin grubu sonradan değişse de bu ürün etkilenmez. Pasif/olmayan grup → `400 Invalid kargo_grubu` (toplu uçta satır hatası).

Platform kargo firma tercihi — `GET` / `PUT` `/shipping/carriers`

Platform kargolu (2) satıcının Geliver'de çalışmak istediği firmalar.

- `GET` → `{"platform_shipping": true, "mode": "all"|"selected", "selected": ["YURTICI"], "carriers": [{"code": "YURTICI", "name": "...", "selected": true}, ...]}`
- `PUT` (veya `PATCH`) `{"carriers": ["YURTICI", "ARAS"]}`; boş dizi = tüm firmalar. Geçerli kodlar: `YURTICI`, `SURAT`, `ARAS`, `MNG`, `PTT`, `HEPSIJET`, `KOLAYGELSIN`, `SENDEO`.
- Hatalar: `400 Invalid carrier code` (`details.invalid`, `details.allowed`) · `409` (satıcı kendi kargolu) · `503 Service unavailable`.

Geliver etiketi (yalnız Geliver ile fiyatlanan siparişler)

Sipariş detayında `shipping.geliver` doluysa sipariş Geliver siparişidir. Geliver olmayan siparişte bu uçlar `409 This order is not shipped with Geliver ...` döner.

Metot ve yol	Amaç
`GET` `/orders/{id}/label`	Etiket durumu ve Geliver özeti
`POST` `/orders/{id}/label`	Etiket al → sipariş Kargoda (5) olur
`GET` `/orders/{id}/label.pdf`	Etiket PDF'i (`application/pdf` veya 302 yönlendirme)
`POST` `/orders/{id}/ship-own`	Geliver siparişini kendi anlaşmanızla gönderin: `{"carrier": "...", "tracking_number": "..."}`

`POST` `/orders/{id}/label` gövdesi (hepsi isteğe bağlı): `length_desi` (≤ 500), `weight_kg` (≤ 500), `carrier_code` (ör. `"YURTICI"`). Hatalar: `400 Package size out of range` / `Invalid carrier_code` · `409` (`details.code`: `uygun_degil`, `islemde`, `firma_yok`; firma uygun değilse `details.available_carriers[] {carrier_code,carrier,price}`) · `422` (`adres`, `alici`) · `502` (Geliver hatası).

`geliver` nesnesi: `carrier_code`, `carrier`, `customer_paid_shipping`, `seller_deduction`, `package{desi,weight_kg}`, `shipping_type` (`geliver_label` / `own_contract` /null), `label{status,test,error,pdf_url}` (`status`: `not_created`, `processing`, `created`, `cancelled`, `error`), `actual_shipping_cost`, `seller_size_difference`, `seller_receivable`, `tracking{status,step,delivered,problem,stale,tracking_number,tracking_url,last_event_at,events[] {status,code,sub_code,location,at}}`.

- `label.pdf_url` site köküne göre **göreli yoldur** (`"/api/v1/orders/812-4471093/label.pdf"`). Başına `https://www.esnafpazar.com` ekleyip **`Authorization: Bearer` başlığıyla** çağırın (herkese açık link değildir). Etiket oluşturulmadıysa `null`.
- `label.test: true` olan etiketlerde `tracking.tracking_url` gerçek bir takip adresi değildir; müşteriye göstermeyin.

Geliver siparişinin `shipping` bölümünden temsilî örnek (değerler uydurma):

```
"shipping": {
  "requires_payment": false, "cost": 0, "kargo_secim": 2, "mode": "platform", "estimated_delive
ry_days": 1,
  "company": { "id": null, "name": "Geliver" },
  "order_level": { "seller_contract_aggregate": 0, "platform_fee_in_siparis": 0 },
  "support_carrier_deduction": 0, "method": "Standard Shipping", "tracking_number": "1234567890
1", "carrier_note": "",
  "geliver": {
    "carrier_code": "GELIVER_STANDART", "carrier": "Geliver", "customer_paid_shipping": 115.85,
"seller_deduction": 0,
    "package": { "desi": 1, "weight_kg": 0.3 }, "shipping_type": "geliver_label",
    "label": { "status": "created", "test": false, "error": null, "pdf_url": "/api/v1/orders/81
2-4471093/label.pdf" },
    "actual_shipping_cost": 0, "seller_size_difference": null, "seller_receivable": null,
    "tracking": {
      "status": "Kargoya verildi", "step": 0, "delivered": false, "problem": false, "stale": fa
lse,
      "tracking_number": "12345678901", "tracking_url": "https://kargo.ornek.com/takip/12345678
901",
      "last_event_at": "2026-11-24T14:32:21+03:00",
      "events": [
        { "status": "Kargoya verildi", "code": "TRANSIT", "sub_code": "package_accepted", "loca
tion": "", "at": "2026-11-24T14:32:21+03:00" },
        { "status": "Etiket hazırlandı", "code": "PRE_TRANSIT", "sub_code": "information_receiv
ed", "location": "Kadıköy", "at": "2026-11-24T14:31:58+03:00" }
      ]
    }
  }
}
```


Geliver siparişinde teslim bilgisi kargo hareketlerinden otomatik işlenir (sipariş 6 olur ve `order.updated` gelir).

5.12 Webhooklar

Webhook, EsnafPazar'da bir olay olduğunda sizin sunucunuza `POST` isteği gönderir. Sipariş çekme sıklığını azaltır; ancak **yeniden deneme yoktur**, bu yüzden periyodik çekmeyle birlikte kullanın (§7).

Abonelik oluştur — `POST` `/webhooks` (eş: `POST` `/webhooks/subscribe`)

Gövde	Tip	Zorunlu	Açıklama
`url`	string	evet	Herkese açık **https** adresi (port 443 veya 8443), yönlendirmesiz
`events`	string[]	evet	Aşağıdaki olaylar. **Boş dizi `[]` = tüm olaylar.**
`secret`	string	hayır	16–128 karakter; harf, rakam ve `. _ ~ + / = -`. Verilmezse sunucu 64 karakter üretir.
`active`	bool	hayır	Varsayılan `true`

```
curl -X POST $BASE/webhooks -H "Authorization: Bearer $TOKEN" -H "Content-Type: application/jso
n" \
  -d '{"url":"https://entegrator.ornek.com/hooks/esnafpazar/128","events":["order.created","ord
er.updated","order.cancelled"]}'
```

HTTP 201:

```
{ "success": true, "data": { "webhook_id": 41, "url": "https://entegrator.ornek.com/hooks/esnaf
pazar/128", "events": ["order.created","order.updated","order.cancelled"], "secret": "<WEBHOOK_
SECRET>", "is_active": true, "message": "Webhook subscription created successfully" } }
```


`secret` **yalnız bu cevapta** döner; saklayın. Hatalar: `400 URL and events are required` · `400 Invalid URL format` · `400 Webhook URL must be a public https URL` · `400 Invalid events: ...` · `400 secret must be 16–128 characters ...`.

Diğer abonelik uçları

Metot ve yol	Açıklama
GET `/webhooks`	`webhooks[]` `{id,url,events,is_active,created_at,updated_at,last_triggered_at,failure_count}` (secret dönmez)
PUT `/webhooks/{id}`	`url`, `events`, `is_active` (dikkat: burada `active` değil `is_active`) değiştirilebilir. Secret değiştirilemez; yenilemek için silip yeniden oluşturun.
DELETE `/webhooks/{id}`	Siler, HTTP 204

Olaylar (gerçekte gönderilenler)

Abonelikte kabul edilen olay adları: `order.created`, `order.updated`, `order.cancelled`, `order.paid`, `product.updated`, `stock.low`. Bunlardan **yalnız ilk üçü gönderilir**; `order.paid`, `product.updated`, `stock.low` şu an hiç tetiklenmez (seçmeniz zararsızdır).

Olay	Ne zaman	`data` alanları
`order.created`	Müşteri ödemeyi tamamladığında (siparişte ürünü olan her satıcıya ayrı)	`order_id`, `seller_total` (satıcının ürün toplamı), `created_at` (ISO 8601)
`order.updated`	Durum değişikliği: API (`status`, `ship`), satıcı paneli, Geliver etiketi/teslim, kısmi iptal; fatura kaydı	Kaynağa göre: `order_id`, `status_code`, `status` (API), `tracking_number`, `carrier` (API ship), `tracking_url`, `source`, `invoice_url`, `invoice_number`, `cancel_reason`, `partial`
`order.cancelled`	Tam iptal (satıcı paneli, API `cancel`, müşteri iptali)	`order_id`, `status_code` (10), `cancel_reason`, `partial` (false), `source` (`api`, `seller_panel`, `customer`)
`order.return_requested`	Müşteri iade talebi oluşturduğunda	`order_id`, `reason`, `items[]{sku,quantity}`

> `order.return_requested` olay listesinde seçilemez; bu olayı almak için aboneliği `events: []` (tüm olaylar) ile oluşturun.

`data` alanları kaynağa göre değişir; zorunlu sayabileceğiniz tek alan `order_id`'dir. Olayı aldıktan sonra güncel hâli `GET` `/orders/{order_id}` ile okuyun. Kendi API çağrılarınızın sonucu da size webhook olarak geri gelir (ör. `ship` → `order.updated`); bunu döngüye sokmayın.

Teslimat biçimi

```
POST /hooks/esnafpazar/128 HTTP/ 1.1
Content-Type: application/json
User-Agent: EsnafPazar-Webhook/1.0
X-EsnafPazar-Event: order.updated
X-EsnafPazar-Signature: sha256=5d41402abc4b2a76b9719d911017c592...

{"event":"order.updated","timestamp":1795525509,"data":{"order_id":"812-4471093","status":"ship
ped","status_code":5,"tracking_number":"1234567890","carrier":"Yurtiçi Kargo"}}
```

- Zaman aşımı: bağlantı 2 sn, toplam 4 sn. 2xx dışı her cevap (ve zaman aşımı) hata sayılır.
- **Yeniden deneme yoktur.** Üst üste 5 hatada abonelik pasif olur (`is_active: false`). Sunucunuzu düzelttikten sonra `PUT` `/webhooks/{id}` `{"is_active": 1}` ile açın; hata sayacı ilk başarılı teslimatta sıfırlanır.
- Yönlendirme izlenmez; yalnız https.
- Hızlı cevap verin (200), işi kuyruğa alın.

İmza doğrulama

İmza, **ham istek gövdesinin** abonelik secret'ı ile HMAC-SHA256'sıdır (onaltılık, küçük harf), başında `sha256=` vardır. Gövdeyi JSON'a çevirip yeniden serileştirmeden, geldiği gibi kullanın.

PHP:

```php
<?php
$secret    = getenv('ESNAFPAZAR_WEBHOOK_SECRET'); // abonelikte dönen secret
$govde     = file_get_contents('php://input');      // ham gövde
$gelen     = $_SERVER['HTTP_X_ESNAFPAZAR_SIGNATURE'] ?? '';
$beklenen  = 'sha256=' . hash_hmac('sha256', $govde, $secret);

if (!hash_equals($beklenen, $gelen)) {
    http_response_code(401);
    exit('invalid signature');
}

$olay = json_decode($govde, true);
// $olay['event'], $olay['timestamp'], $olay['data']['order_id'] → kuyruğa al
http_response_code(200);
echo 'ok';
```

Node.js (Express):

```
const crypto = require('crypto');
const express = require('express');
const app = express();

app.post('/hooks/esnafpazar/:sellerId', express.raw({ type: 'application/json' }), (req, res) =
> {
  const secret = process.env.ESNAFPAZAR_WEBHOOK_SECRET;
  const gelen = req.get('X-EsnafPazar-Signature') || '';
  const beklenen = 'sha256=' + crypto.createHmac('sha256', secret).update(req.body).digest('he
x');

  const a = Buffer.from(gelen);
  const b = Buffer.from(beklenen);
  if (a.length !== b.length || !crypto.timingSafeEqual(a, b)) {
    return res.status(401).send('invalid signature');
  }

  const olay = JSON.parse(req.body.toString('utf8'));
  // olay.event, olay.timestamp, olay.data.order_id → kuyruğa al
  res.status(200).send('ok');
});
```

Tekrar saldırısına karşı `timestamp` (unix saniye) çok eskiyse (ör. 10 dakikadan eski) olayı reddedebilirsiniz.

> 24.11.2026 öncesinde oluşturulmuş aboneliklerin secret'ı hatalı kaydedilmiş olabilir; bu aboneliklerde imza doğrulanmaz. Bu tarihten önce abonelik oluşturduysanız aboneliği silip yeniden oluşturun.

---

6. Sipariş ve iade durum kodları

Sipariş durumu **satıcı bazındadır** (aynı siparişteki her satıcının kendi durumu vardır).

Kod	API adı	Panel karşılığı
0	`pending`	Onay bekliyor
1	`approved`	Onaylandı
2	`supplying`	Tedarik sürecinde
3	`preparing`	Hazırlanıyor
4	`packaging`	Paketleniyor
5	`shipped`	Kargoda
6	`delivered` (filtrede `completed` da olur)	Teslim edildi
7	`return_pending`	İade bekleniyor
8	`return_received`	İade geldi (inceleniyor)
9	`return_completed`	İşlem tamamlandı (iadeli)
10	`cancelled`	İşlem tamamlandı (iptalli)
11	`partial_shipment_return`	Kısmi gönderim (iadeli)
12	`partial_shipment_cancel`	Kısmi gönderim (iptalli)

İzinli geçişler (satıcı paneliyle aynı kural)

Aynı duruma "geçiş" her zaman izinlidir (değişiklik olmaz). Bunun dışında yalnız şu geçişler yapılabilir:

Mevcut	Gidilebilecek durumlar
0	1
1	2, 3, 4, 5, 12
2	3, 4, 5, 12
3	4, 5, 12
4	5, 12
5	6
12	6

Mevcut	Gidilebilecek durumlar
6	7, 8, 9, 11
7	8, 9, 11
8	9, 11
11	9
9	— (son durum)
10	— (son durum)

Ek kurallar:

- **Örtük onay:** PATCH `/status` ve POST `/ship` uçlarında, 0 durumundaki siparişe 1'den gidilebilen durumlar (2, 3, 4, 5, 12) doğrudan gönderilebilir.
- **10 (iptal)** yalnız POST `/orders/{id}/cancel` ile verilir; yalnız 0–4 durumunda (ve paketi şubeye verilmemiş Geliver etiketli siparişte).
- PATCH `/status` yalnız 1–6 hedeflerini kabul eder; 7, 8, 9, 11 hedefleri iade/kısmi gönderim uçlarıyla verilir. 12 (kısmi gönderim-iptalli) için API'de uç yoktur; satıcı paneliyle verilir.
- Kural dışı istek **409** döner:

```
{
  "success": false,
  "error": {
    "message": "Status transition not allowed: 5 -> 7 (allowed from current status: 6)",
    "code": "Conflict",
    "details": { "current_status_code": 5, "requested_status_code": 7, "allowed_status_codes":
[6] }
  }
}
```

(PATCH `/status` ucunun 409 mesajında `details` yer almaz; yalnız `Status transition not allowed: X -> Y`.)

Uç → hedef kod eşlemesi

Uç	Hedef kod
PATCH `/orders/{id}/status`	Gönderilen (1–6)
POST `/orders/{id}/ship`, POST `/orders/{id}/label`, POST `/orders/{id}/ship-own`	5
POST `/orders/{id}/cancel`	10
POST `/orders/{id}/return`	7
PATCH `/orders/{id}/returns/{rid}`	`pending` 7, `approved` iade talebi varsa panel kabulü (Geliver'siz siparişte 6→7, Geliver'lide değişmez), yoksa 8; `completed` 9

Uç	Hedef kod
POST `/orders/{id}/returns/{rid}/approve`	İade talebi varsa panel kabulü (Geliver'siz siparişte 6→7, Geliver'lide değişmez), yoksa 8
POST `/orders/{id}/partial-ship`	11

7. Önerilen entegrasyon akışı

7.1 Kurulum (satıcı başına bir kez)

1. Satıcıdan `api_key` / `api_secret` alın, şifreli saklayın. POST `/auth/token` ile doğrulayın.
2. GET `/categories/tree` ve GET `/variants` ile sözlükleri indirin (günde bir yenileyin). Ürün eşleştireceğiniz kategoriler için GET `/categories/{id}/attributes`.
3. Marka gerekiyorsa GET `/brands` ile satıcının markalarını alın, yoksa POST `/brands` ile oluşturun; dönen `brand_id`'yi saklayın.
4. POST `/webhooks` ile sipariş olaylarına abone olun (`events: []` önerilir; iade talebini de alırsınız). `secret`'ı saklayın.

7.2 Ürün yükleme ve onay

1. Ürünleri POST `/products/bulk` ile 100'lük paketler hâlinde gönderin (kendi SKU'nuzla). Yeni ürünler `passive` + inceleme bekler.
2. Satır hatalarını (`422` → `error.details.results.details[]`) düzeltip yeniden gönderin; aynı SKU güncelleme olarak işlenir.
3. **Onay takibi:** Periyodik olarak (ör. 1–4 saatte bir) GET `/products?review_status=approved&status=passive` ile onaylanmış ama pasif ürünleri bulun ve POST `/products/bulk-update` `{"update_fields":["status"], ...status:1}` ile aktif edin. `review_status=pending` henüz incelemede olanları, `review_status=rejected` reddedilenleri `review_note` (ret gerekçesi) ile verir; gerekçeye göre ürünü düzeltip güncelleyin.
4. Vitrinde görünürlüğü `is_listed` alanıyla doğrulayın.
5. Satıcı ürünü pasif tutmak istiyorsa aktif etme adımını atlayın.

7.3 Stok ve fiyat senkronu

- Değişiklik oldukça (veya 5–15 dakikalık periyotlarla) yalnız **değişen** satırları gönderin:
- Stok: PATCH `/stock/bulk` (ürün kimliği + stok kodu + adet).
- Fiyat: POST `/products/bulk-update` (ürün düzeyi) veya PATCH `/stock/{product_id}` (varyant düzeyi).
- EsnafPazar tarafındaki değişiklikleri (sipariş sonrası stok düşüşü, panelden düzenleme) yakalamak için GET `/products?updated_after=<son_senkron_unix>&include=stock` kullanın.
- Ürün adı/açıklamasını her senkronda göndermeyin; değişmediyse sorun olmaz, ama yanlışlıkla değişirse ürün incelemeye düşer.
- Saatlik 1000 istek sınırını hesaba katın: 1000 ürünü tek tek güncellemek limiti bitirir; toplu uçları kullanın.

7.4 Sipariş çekme

- **Webhook + yoklama birlikte:** Webhook geldiğinde GET `/orders/{order_id}` ile detayı çekin. Webhook'ta yeniden deneme olmadığından ayrıca **5–10 dakikada bir** GET `/orders?updated_after=<son_çekme_zamanı − 5 dk>` ile yoklayın (5 dakikalık örtüşme saat farklarına karşı). Tarihi `2026-11-24T14:00:00+03:00` biçiminde gönderin.

- Listede kalem olmadığından yeni/değişen her sipariş için detay çağrısı gerekir; istek bütçesini buna göre planlayın.
- Siparişi `order_id` ile tekilleştirin (aynı sipariş hem webhook hem yoklamayla gelebilir).
- Onay: Siparişi işleme aldığınızda PATCH `/status` `approved` (veya doğrudan `preparing`).

7.5 Kargo ve fatura bildirimi

1. Paket çıktığında POST `/orders/{id}/ship` (`tracking_number`, `carrier`). Geliver siparişinde etiketi POST `/orders/{id}/label` ile alabilirsiniz.
2. E-fatura kesildiğinde POST `/orders/{id}/invoice`.
3. Teslim: Geliver siparişlerinde otomatik. Diğerlerinde teslim bilgisini aldığınızda PATCH `/status` `delivered`.
4. İptal gerekiyorsa kargoya vermeden önce POST `/orders/{id}/cancel`.
5. İade: `order.return_requested` webhook'u → GET `/orders/{id}/returns` (`return_request.id`, `answer_deadline`) → 3 gün içinde `approve` (Geliver etiketsiz siparişte `return_code` ile) veya `reject`. Cevap verilmezse talep otomatik onaylanır. Sonraki adımlar (ürün geldi, para iadesi) satıcı panelinden ve EsnafPazar tarafından yürütülür.

---

8. Sınırlamalar ve bilinen durumlar

Konu	Durum
Test ortamı (sandbox)	**Yoktur.** Tüm istekler canlı pazaryerine gider. Geliştirme için EsnafPazar'dan ayrı bir **test satıcı hesabı** isteyin, ürünleri pasif tutun ve gerçek müşteri siparişine dokunmayın.
Sipariş oluşturma	Yoktur (`POST` `/orders` → 405). Siparişler yalnız müşteri ödemesiyle oluşur.
Toplu işlem limitleri	`POST` `/products/bulk` : en çok 100 satır (sunucu zorlar). `bulk-update` , `bulk-delete` , `bulk-stock-update` , `PATCH` `/stock/bulk` : sunucu sınırı yok; zaman aşımına düşmemek için en çok 100 ürün / 500 stok satırı önerilir.
Hız sınırı	Satıcı başına saatte 1000 istek. `/auth/*` uçları sayılmaz ve rate limit başlığı döndürmez.
Platform marka listesi	Yok; yalnız satıcının kendi markaları.
Kategori özellik kuralları	Zorunlu/serbest metin bilgisi yok.
Varyant kendi SKU'su	`POST` `/products/{id}/variants` stok kodunu sunucu üretir; kendi SKU'nuz için ürün oluştururken `variants[]` kullanın.
Görsel sırası	Saklanmaz; kapak önce, sonra eklenme sırası.
İade süreci sonraki adımları	API'de iade talebi yalnız onaylanır/reddedilir. "Ürün geldi", "Onayla (para iadesi)", "Sorun bildir" ve sonradan iade kodu girme satıcı panelindedir. Müşteri iade talebi kaydı olmayan eski iadelerde (`POST` `/orders/{id}/return`) onay yalnız sipariş durumunu 8 yapar (§5.10).
Kalem bazlı iptal / kısmi gönderim	API'de yok; satıcı paneli kullanılır. `partial-ship` yalnız durum kodu yazar (§5.9.6).
Refresh token	Satıcı başına tek; birden çok istemci birbirini düşürür (§3).
Webhook	Yeniden deneme yok; `order.paid` , `product.updated` , `stock.low` gönderilmiyor; `order.return_requested` yalnız `events: []` ile.
Raporlar (`/reports/*`)	Toplamlar henüz doğru hesaplanmadığı için bu dokümanda yer almıyor; **yakında**. Kullanmayın.
Excel/CSV içe aktarma (`/products/bulk-import`)	Çalışmıyor; kullanmayın. `POST` `/products/bulk` kullanın.
Ödeme iadesi (`/orders/{id}/refund`)	Çalışmıyor; kullanmayın. Para iadesini EsnafPazar yapar.
Sipariş geçmişi (`/orders/{id}/history`)	Güvenilir değil (başka kayıtlarla karışabilir); kullanmayın.
Ürün listesindeki `brand_id` , `kargo_grubu`	Sırasıyla filtre seçeneği kimliği ve her zaman `null` (§5.4.1).
Sipariş `vat_rate`	Sabit 20; gerçek oranı kendi verinizden alın.

Konu	Durum
`market_price`	Listede ve `/products/{id}/stock` → `product` içinde kart değeri (girilmemişse `0`); detayda stok satırına / `price`'a düşer (§5.4.1).
Düşük stok eşiği	GET `/products/{id}/stock` sabit 10, GET `/stock` panel eşiği (§5.7).
Para birimi	Yalnız TRY.
Hata kodları	Genel makine-okur hata kodu yok; HTTP durum koduna göre karar verin. Doğrulama hatalarında (422) `error.details.field`, stok uçlarında `error_code` döner. Mesajlar İngilizce.

Destek talebi açarken `request_id`, isteğin yolu ve zamanı iletin; API anahtarı veya token paylaşmayın.

9. Sürüm notları

24.11.2026 — canlı test düzeltmeleri

- Kategori uçlarında `icon` ve `image` adresleri artık `https://www.esnafpazar.com/dosya/kategori/<dosya>` (eski `/dosya/<dosya>` adresi 404 dönüyordu).
- PATCH `/products/{id}/stock` gövdesinde `main_stock`, `variants[]` veya `bulk_update[]` alanlarından hiçbiri yoksa artık `422` döner (önceden hiçbir şey yazmadan `200` dönüyordu).

Canlı ortamda uçtan uca test sonrası yapılan kod ve doküman düzeltmeleri.

Davranışı değişenler (entegrasyonunuzu kontrol edin):

- **Saat dilimi tek.** API'nin tamamı artık Europe/Istanbul (UTC+3) ile çalışıyor. Önce zarftaki `timestamp`, ürün listesi `created_at` / `updated_at` ve `POST/PATCH /products/{id}/stock` `updated_at` UTC dönüyordu, yalnız sipariş detayı TR saatiydi. Unix alanlar değişmedi (§1).
- **Görsel URL'leri düzeldi.** GET `/stock`, GET `/stock/{id}` ve sipariş kalemindeki `image` artık `https://www.esnafpazar.com/dosya/urun/<dosya>` (önce `/dosya/<dosya>` dönüyordu ve açılmıyordu).
- **Sipariş listesinde `updated_at` dolu.** Satıcı siparişinin son işlem zamanını (unix) veriyor; `updated_after` filtresiyle aynı alan. Önce listede boş geliyordu.
- **Sipariş kalemi tutarı.** İndirim yoksa `discount_price` = `price` ve `total` = `price × quantity` (önce ikisi de `0` dönüyordu).
- **Hata zarfı `error.code`.** 405 için `Method Not Allowed`, 502 için `Bad Gateway`, 503 için `Service Unavailable` (önce `Unknown`).

Doküman düzeltmeleri:

- `order_id` gerçek biçimi (`"812-4471093"` gibi tireli metin) tüm örneklere işlendi (§5.9).
- `market_price`'ın liste/detay/stok uçlarındaki farkı, varyantsız üründe `variants: []`, düşük stok eşiği farkı (sabit 10 / panel eşiği) yazıldı (§5.4, §5.5, §5.7).
- Geliver siparişinde müşterinin ödediği kargonun `totals.shipping`'de görünmediği, `label.pdf_url`'nin göreli yol olup Bearer ile çağrıldığı yazıldı (§5.9.1, §5.11).
- `return_request: null` durumu, 404 mesajlarının uca göre değiştiği, PATCH `/stock/bulk` `400 stock_updates array is required` satırı eklendi.
- Rate limit başlıklarının `/auth/*` uçlarında (`verify` dahil) gelmediği ve adlarının küçük harfle gelebileceği notu eklendi (§4.3).
- `meta` alanlarının tipleri (sayı/boolean) ve temsilî örnekler gerçek cevap yapısına göre güncellendi.
- Örneklerdeki unix zaman damgaları 2026 tarihlerine hizalandı (önce 2025'e düşüyordu); saat dilimi notu eklendi.

24.11.2026 — satıcı API eksikleri (ek güncelleme)

Yeni:

- **Ürün inceleme durumu.** GET `/products` ve GET `/products/{id}` cevaplarına `review_status` (`pending` / `approved` / `rejected`), `is_listed` (vitrinde görünür mü) ve `review_note` (ret gerekçesi, yalnız reddedilende) eklendi. GET `/products?review_status=` filtresi eklendi (§5.4).
- **Kargo firmaları ucu.** GET `/shipping/companies` : kendi kargolu satıcının `shipping_company_id` kimlikleri (§5.11).
- **İade onayı panelle aynı.** `returns/{id}/approve` (ve PATCH `returns/{id}` `status: approved`) artık satıcı panelindeki "İadeyi kabul et" ile birebir çalışır: yalnız `pending_seller_response` , Geliver etiketsiz siparişte `return_code` zorunlu, müşteri bilgilendirilir (§5.10). İade talebi kaydı olmayan eski iadelerde eski davranış sürer.

Davranışı değişenler (entegrasyonunuzu kontrol edin):

- **Doğrulama hataları 422.** Marka/filtre/kategori/görsel/SKU uzunluğu/kargo kuralı hataları 500 yerine **422** + `error.details.field` döner (§4.4). POST `/products` 'taki kargo kuralı hataları 400 yerine 422. 500 artık yalnız sunucu hatasıdır.
- **Kategori ve kargo firması doğrulanıyor.** Sistemde olmayan `category_id` ve GET `/shipping/companies` listesinde olmayan `shipping_company_id` 422 döner (önce kabul ediliyordu).
- **Stok uçları yeni satır açmıyor.** PATCH `/stock/{product_id}` , PATCH `/stock/bulk` , POST `/products/bulk-stock-update` , `POST|PATCH /products/{id}/stock` ile ürün güncellemedeki `stock_quantity` bilinmeyen stok kodunda satır açmaz; hata döner (`stock_code_not_found` ; tekil uçta 422, toplu uçlarda satır hatası). Toplu stokta `items[]` / `error_code` eklendi (§5.7).
- **Yeni ürün her zaman bir stok satırıyla oluşur** (stok 0; `stock_quantity` / `variants` verilmese de), satıcı paneli gibi.

24.11.2026 — güvenlik ve tutarlılık güncellemesi

Davranışı değişenler (entegrasyonunuzu kontrol edin):

- **Yeni ürünler pasif ve incelemede oluşur.** POST `/products` ve POST `/products/bulk` ile eklenen ürünler artık `passive` + yönetim incelemesinde başlar (POST `/products` cevabında `status: "passive"` , `review_status: "pending"`). Onay sonrası `bulk-update` ile `status: 1` gönderilmelidir.
- **İçerik değişince yeniden inceleme.** Ad, açıklama, kategori, barkod veya SKU gerçekten değişen ürün (satıcı paneliyle aynı kuralla) yeniden incelemeye düşer. Fiyat, stok, görsel, kargo değişikliği etkilemez.
- **Durum değiştirme kısıtı.** `bulk-update` `status` yalnız `1/active` veya `0/passive` kabul eder; reddedilmiş ürünün durumu değiştirilemez, onaylanmamış ürün aktif yapılamaz.
- **İade reddi panelle aynı.** `returns/{id}/reject` (ve PATCH `returns/{id}` `status: rejected`) artık satıcı panelindeki retle birebir çalışır: `reason` + `description` zorunlu, yalnız `pending_seller_response` durumunda, müşteri bilgilendirilir ve itiraz edebilir; sipariş durumu değişmez. GET `/orders/{id}/returns` cevabına `return_request` alanı eklendi.
- POST `/orders` **kapatıldı (405).** Satıcının API ile sipariş oluşturması desteklenmiyor.
- **Durum geçiş kuralı tüm uçlarda.** `ship` , `return` , `returns/{id}` , `returns/{id}/approve` , `partial-ship` uçları da satıcı panelinin geçiş kuralına bağlandı; kural dışı istek 409 + `details{current_status_code, requested_status_code, allowed_status_codes}` . Örnek: iptal edilmiş sipariş kargoya verilemez, iade yalnız teslimden sonra açılır.
- **Webhook secret.** Secret artık değiştirilmeden saklanıyor; 16–128 karakter, `[A-Za-z0-9._~+/=-]` . Daha önce oluşturulan aboneliklerin secret'ı hatalı kaydedilmiş olabilir: **silip yeniden oluşturun.**
- **Görsel URL güvenliği.** Yalnız http/https, herkese açık adres, port 80/443/8080/8443; yönlendirme en çok 3 adım ve her adım doğrulanır. Özel ağ veya standart dışı port adresleri reddedilir.

- **API anahtarı sıfırlama.** Panelden anahtar sıfırlanınca veya yeniden üretilince tüm refresh token'lar iptal olur. Refresh token artık 64 karakter kriptografik rastgele değer.
- **Hata mesajları.** Sunucu iç hatalarında ham veritabanı/PHP mesajı yerine genel mesaj ve `request_id` döner.
- **Sorgu dizesi.** `products`, `stock`, `webhooks` uçlarında yola eklenen sorgu dizesi artık yönlendirmeyi bozmuyor (ör. POST `/products/bulk-update?x=1` doğru uca gider).

Düzeltmeler:

- GET `/products/{id}/stock` artık çalışıyor (önce 500 dönüyordu); `attributes` dolu geliyor.
- GET `/products/{id}/variants` ve `/stock` yalnız o ürünün satırlarını döndürüyor.
- POST `/products/{id}/images` dosya yükleme düzeltildi; dosya uzantısı içerikten belirleniyor.
- Ürün görselleri değiştirilirken/silinirken başka ürünlerin kullandığı dosyalar artık silinmiyor; mevcut kendi görsel URL'nizi `images` listesinde geri göndermek görseli korur.
- `brand_id` alanında eski (filtre seçeneği) kimlikler yalnız satıcının kendi aktif markasına karşılık geliyorsa kabul ediliyor.