Marketplace

Author: Josh S. Sakweli, Backend Lead Team
Last Updated: 2026-06-04
Version: v1.0

Base URL: `api/v1/e-commerce/marketplace`

Short Description: The Marketplace API is the primary product discovery layer of the Nexgate e-commerce platform. It exposes personalised product feeds, trending rankings, hot deals, live group purchases, and a powerful advanced filter with keyword search — all driven by a scoring formula built on real purchase, view, and cart signals.

Hints:

- All endpoints work for **anonymous users** (no token required). Authenticated users automatically receive personalised results where applicable — no extra parameter needed.
- `TRENDING` and `FOR_YOU` feeds re-rank within each page using the full scoring formula. Other sort modes (`NEWEST`, `PRICE_ASC`, etc.) are sorted at the database level and return exact ordering.
- `hasMultipleColors`, `maxGroupSeatsLeft`, and `minGroupDiscountPercent` filters are applied **after** the database query, so the `totalElements` count in the response reflects the pre-filtered DB count — not the post-filtered count.
- `viewCount` increments each time `GET /shops/{shopId}/products/{productId}` is called. `cartAddCount` increments when a product is added to a cart for the first time (not on quantity updates).
- Pages are **1-based** (`page=1` returns the first page).

Standard Response Format

All API responses follow a consistent structure using the Globe Response Builder pattern:

Success Response Structure

```
{
  "success": true,
  "httpStatus": "OK",
  "message": "Operation completed successfully",
```

```
  "action_time": "2026-06-04T10:30:45",
  "data": {
    "content": [...],
    "currentPage": 1,
    "pageSize": 20,
    "totalElements": 158,
    "totalPages": 8,
    "hasNext": true,
    "hasPrevious": false
  }
}
```

Error Response Structure

```
{
  "success": false,
  "httpStatus": "BAD_REQUEST",
  "message": "Error description",
  "action_time": "2026-06-04T10:30:45",
  "data": "Error description"
}
```

Standard Response Fields

Field	Type	Description
`success`	boolean	`true` for success, `false` for errors
`httpStatus`	string	HTTP status name (OK, BAD_REQUEST, NOT_FOUND, etc.)
`message`	string	Human-readable operation result
`action_time`	string	ISO 8601 timestamp of response generation
`data`	object	Paginated payload for success, error detail for failures

---

Scoring Formulas

These formulas are the engine behind `TRENDING`, `FOR_YOU`, and `BEST_DEAL` sort modes. Understanding them helps predict how products are ranked.

Log Normalization

All count-based signals (soldQuantity, viewCount, cartAddCount) are log-normalized before applying weights. This prevents one product with 100,000 sales from dominating everything else.

```
normalize(value) = min(1.0,  log(1 + value) / log(1 + 10,000))


Examples:
  0 sales     →  0.000
  100 sales   →  0.501
  1,000 sales →  0.750
  10,000 sales → 1.000   ← reference ceiling
  50,000 sales → 1.000   ← capped at 1.0
```

Formula 1 — Trending Score (global, same for everyone)

```
┌──────────────────────────────────────────────────────────────────────┐
│                      TRENDING SCORE FORMULA                          │
├──────────────────────────────────────────────────────────────────────┤
│                                                                      │
│  trendingScore =                                                     │
│      normalize(soldQuantity)   × 0.30   ← real money paid           │
│    + normalize(viewCount)      × 0.25   ← real browsing intent      │
│    + groupHeat                 × 0.20   ← social urgency            │
│    + normalize(cartAddCount)   × 0.15   ← purchase intent           │
│    + discountStrength          × 0.07   ← deal attractiveness       │
│    + recencyBonus              × 0.03   ← freshness tiebreaker      │
│                                                                      │
│  groupHeat       = seatsOccupied / totalSeats  (hottest live group)  │
│  discountStrength= (comparePrice - price) / comparePrice            │
│  recencyBonus    = 1.0 if ≤ 7 days old                              │
│                    0.5 if ≤ 30 days old                              │
│                    0.0 otherwise                                     │
└──────────────────────────────────────────────────────────────────────┘
```

Formula 2 — Personalized Trending Score (authenticated users)

```
┌──────────────────────────────────────────────────────────────────┐
│                 PERSONALIZED TRENDING SCORE FORMULA              │
├──────────────────────────────────────────────────────────────────┤
│                                                                  │
│  personalizedScore =                                             │
│      trendingScore                                               │
│    + 0.25   (if product's shop is in user's subscriptions)       │
│    + 0.00   (otherwise)                                          │
│                                                                  │
│  This is Option A — soft priority. Subscribed-shop products float│
│  near the top but a genuinely viral product still beats them.    │
└──────────────────────────────────────────────────────────────────┘
```

Formula 3 — Relevance Score (For You feed)

```
┌──────────────────────────────────────────────────────────────────┐
│                     RELEVANCE SCORE FORMULA                      │
├──────────────────────────────────────────────────────────────────┤
│                                                                  │
│  relevanceScore =                                                │
│      categoryMatch  × 0.40   ← product's category is in cart     │
│    + favShopBoost   × 0.40   ← shop is in user's subscriptions   │
│    + trendingScore  × 0.20   ← global popularity tiebreaker      │
│                                                                  │
│  categoryMatch = 1.0 if product's category matches any category  │
│                  currently in the user's cart, else 0.0          │
│  favShopBoost  = 1.0 if shop is subscribed, else 0.0             │
│                                                                  │
│  Fallback: if user has no cart items and no subscriptions,       │
│  the endpoint falls back to the Personalized Trending feed.      │
└──────────────────────────────────────────────────────────────────┘
```

Advanced Filter — UI Reference

The advanced filter groups its parameters into logical sections. Below is a reference layout showing how a frontend filter panel would be structured:

```
┌──────────────────────────────────────────────────────────┐
│                   MARKETPLACE FILTERS                    │
├──────────────────────────────────────────────────────────┤
│  □□ SEARCH                                               │
│  ┌──────────────────────────────────────────────────┐    │
│  │  ?q=  keyword search across name & description...  │  │
│  └──────────────────────────────────────────────────┘    │
├──────────────────────────────────────────────────────────┤
│  □□ PRICE RANGE                                          │
│      Min [ minPrice _________ ]                          │
│      Max [ maxPrice _________ ]                          │
├──────────────────────────────────────────────────────────┤
│  □□□ PRODUCT                                             │
│      Category    [ categoryId ▼ ]                        │
│      Condition   ○ NEW   ○ USED   ○ REFURBISHED          │
│      Type        ○ Physical   ○ Digital                  │
│      Urgency     ○ NONE  ○ LIMITED_TIME  ○ LOW_STOCK     │
│                  ○ FLASH_SALE                            │
│      [ ] hasMultipleColors  — show colour variants only  │
├──────────────────────────────────────────────────────────┤
│  □□ AVAILABILITY                                         │
│      [ ] inStock           — in-stock only               │
│      Min stock [ minStockQuantity ]  — bulk buyers       │
├──────────────────────────────────────────────────────────┤
│  □□ GROUP DEALS                                          │
│      [ ] hasGroupBuying    — supports group buying       │
│      [ ] hasActiveGroup    — live OPEN group right now   │
│      Max seats left  [ maxGroupSeatsLeft ]  ← urgency    │
│      Min group disc. [ minGroupDiscountPercent ]%        │
├──────────────────────────────────────────────────────────┤
│  □□ PAYMENT                                              │
│      [ ] onSale            — currently discounted        │
│      [ ] hasInstallments   — instalment plans available  │
├──────────────────────────────────────────────────────────┤
│  □□ SHOP TRUST                                           │
```

```
│      [ ] shopVerified      — verified shops only          │
│      Min trust score  [ minTrustScore ] / 5.00            │
├──────────────────────────────────────────────────────────┤
│  □□ POPULARITY                                           │
│      Min sold count  [ minSoldCount ]                     │
├──────────────────────────────────────────────────────────┤
│  ↕□  SORT BY                                             │
│      ○ TRENDING      ← formula-ranked                    │
│      ○ FOR_YOU       ← personalised relevance            │
│      ○ NEWEST        ← createdAt DESC                    │
│      ○ PRICE_ASC     ← cheapest first                    │
│      ○ PRICE_DESC    ← most expensive first              │
│      ○ MOST_SOLD     ← soldQuantity DESC                 │
│      ○ BEST_DEAL     ← highest discount % first          │
│      ○ MOST_VIEWED   ← viewCount DESC                    │
│      ○ MOST_CARTED   ← cartAddCount DESC                 │
└──────────────────────────────────────────────────────────┘
```

Endpoints

---

1. Main Discovery Feed

Purpose: Returns the primary marketplace product feed with optional filters and all sort strategies. Works for both anonymous and authenticated users — authenticated users receive personalised scoring automatically.

Endpoint: `GET` `api/v1/e-commerce/marketplace/feed`

Access Level: □□ Public (personalised when authenticated)

Authentication: Bearer Token (optional — improves results when provided)

Query Parameters:

Parameter	Type	Required	Description	Default

`sortBy`	enum	No	Sort strategy. Values: `TRENDING`, `FOR_YOU`, `NEWEST`, `PRICE_ASC`, `PRICE_DESC`, `MOST_SOLD`, `BEST_DEAL`, `MOST_VIEWED`, `MOST_CARTED`	`TRENDING`
`page`	integer	No	Page number (1-based)	`1`
`size`	integer	No	Items per page	`20`
`minPrice`	decimal	No	Minimum product price	—
`maxPrice`	decimal	No	Maximum product price	—
`categoryId`	UUID	No	Filter by product category	—
`condition`	enum	No	`NEW`, `USED`, `REFURBISHED`	—
`productType`	enum	No	`PHYSICAL`, `DIGITAL`	—
`inStock`	boolean	No	`true` = in-stock products only	—
`onSale`	boolean	No	`true` = discounted products only	—
`hasActiveGroup`	boolean	No	`true` = products with a live OPEN group right now	—
`shopVerified`	boolean	No	`true` = verified shops only	—

Success Response JSON Sample:


```
{
  "success": true,
  "httpStatus": "OK",
  "message": "Feed retrieved successfully",
  "action_time": "2026-06-04T10:30:45",
  "data": {
    "content": [
      {
        "productId": "3fa85f64-5717-4562-b3fc-2c963f66afa6",
        "productName": "Samsung Galaxy S24",
        "productSlug": "samsung-galaxy-s24-techstore",
```

```
"productPrimaryMedia": {
  "fileId": "9c1a2b3d-4e5f-6789-abcd-ef0123456789",
  "mediaType": "IMAGE",
  "status": "READY",
  "thumbUrl": "https://cdn.nexgate.com/products/galaxy-s24/thumb.webp",
  "mimeType": "image/webp"
},
"productType": "PHYSICAL",
"price": 850000.00,
"comparePrice": 1050000.00,
"discountPercentage": 19.05,
"effectiveDiscountPercentage": 31.43,
"stockQuantity": 42,
"soldQuantity": 318,
"viewCount": 2741,
"cartAddCount": 195,
"urgencyTag": "LOW_STOCK",
"condition": "NEW",
"inStock": true,
"onSale": true,
"hasInstallments": true,
"shopId": "7cb3a812-1234-4abc-b3fc-9d84f55bce12",
"shopName": "TechStore Tanzania",
"shopSlug": "techstore-tanzania",
"shopLogoMedia": {
  "fileId": "1d2e3f4a-5b6c-7890-defa-bc1234567890",
  "status": "READY",
  "thumbUrl": "https://cdn.nexgate.com/shops/techstore/thumb.webp",
  "mimeType": "image/jpeg"
},
"shopVerified": true,
"shopTrustScore": 4.80,
"categoryId": "a1b2c3d4-1234-5678-abcd-ef0123456789",
"categoryName": "Smartphones",
"hasActiveGroup": true,
"activeGroupHeat": 0.78,
"activeGroupPrice": 720000.00,
"activeGroupSeatsLeft": 4,
"activeGroupExpiresAt": "2026-06-04T18:00:00",
"createdAt": "2026-05-28T09:15:00"
```

```
      }
    ],
    "currentPage": 1,
    "pageSize": 20,
    "totalElements": 1284,
    "totalPages": 65,
    "hasNext": true,
    "hasPrevious": false
  }
}
```

Success Response Fields:

Field	Description
`content`	Array of product cards for this page
`content[].productId`	Unique product identifier
`content[].productPrimaryMedia`	Resolved primary media of the product — `{fileId, mediaType, status, thumbUrl, mimeType}`. `null` if the product has no media yet
`content[].price`	Current selling price
`content[].comparePrice`	Original price (present only if product is on sale)
`content[].discountPercentage`	Regular sale discount % — `null` if no `comparePrice`
`content[].effectiveDiscountPercentage`	Best available deal across ALL discount types — `max(salePct, activeGroupPct) × 100`. `null` if product has no discount of any kind. This is what the Hot Deals feed sorts by
`content[].soldQuantity`	Total units sold — visible only if shop has enabled this
`content[].viewCount`	Cumulative public views since tracking began
`content[].cartAddCount`	Cumulative times added to any cart
`content[].urgencyTag`	`NONE`, `LIMITED_TIME`, `LOW_STOCK`, `FLASH_SALE`
`content[].hasActiveGroup`	`true` if there is a live OPEN group purchase right now
`content[].activeGroupHeat`	Group fill ratio — `0.0` (empty) to `1.0` (full). `null` if no active group
`content[].activeGroupPrice`	Discounted price available inside the active group
`content[].activeGroupSeatsLeft`	Remaining seats in the active group
`content[].activeGroupExpiresAt`	When the active group purchase expires
`content[].shopTrustScore`	Shop trust rating from `0.00` to `5.00`
`currentPage`	Current page number (1-based)

Field	Description
`totalElements`	Total matching products across all pages
`hasNext` / `hasPrevious`	Pagination navigation flags

Standard Error Types:

- `500 INTERNAL_SERVER_ERROR`: Unexpected server failure

---

2. Trending Products

Purpose: Returns products ranked by the trending score formula. Anonymous users receive the global trending score. Authenticated users receive a personalised ranking — products from subscribed shops receive a +0.25 score boost so they float near the top.

Endpoint: `GET` `api/v1/e-commerce/marketplace/trending`

Access Level: □□ Public (personalised when authenticated)

Authentication: Bearer Token (optional)

Scoring applied:

```
Anonymous  →  trendingScore
Authenticated →  trendingScore + 0.25 (if product's shop is subscribed)
```

> **Note**: The database pre-sorts by `soldQuantity DESC` as an approximation, then the full formula re-ranks within each page. This means the absolute order across pages may differ slightly from a pure formula sort — which is intentional (pagination stability).

Query Parameters:

Parameter	Type	Required	Description	Default
`page`	integer	No	Page number (1-based)	`1`
`size`	integer	No	Items per page	`20`
`categoryId`	UUID	No	Filter by category	—
`minPrice`	decimal	No	Minimum price	—

Parameter	Type	Required	Description	Default
`maxPrice`	decimal	No	Maximum price	—
`inStock`	boolean	No	In-stock only	—
`onSale`	boolean	No	On sale only	—
`shopVerified`	boolean	No	Verified shops only	—

Success Response: Same structure as the Main Feed endpoint.

Standard Error Types:

- `500 INTERNAL_SERVER_ERROR`: Unexpected server failure

---

3. For You — Personalised Recommendations

Purpose: Returns products ranked by relevance to the authenticated user based on their cart categories and shop subscriptions. Falls back to the trending feed for unauthenticated users or users with no cart items and no subscriptions.

Endpoint: `GET` `api/v1/e-commerce/marketplace/for-you`

Access Level: □□ Public (best results when authenticated)

Authentication: Bearer Token (optional — falls back to trending if absent)

Scoring applied:

```
relevanceScore =
    categoryMatch  × 0.40   (product's category is in user's cart)
  + favShopBoost   × 0.40   (shop is in user's subscriptions)
  + trendingScore  × 0.20   (global popularity tiebreaker)

Pool strategy: fetches 3× the requested page size (max 150), scores in Java,
then slices the requested page from the ranked result.
```

Query Parameters:

Parameter	Type	Required	Description	Default

`page`	integer	No	Page number (1-based)	`1`
`size`	integer	No	Items per page	`20`
`categoryId`	UUID	No	Narrow recommendations to a specific category	—
`inStock`	boolean	No	In-stock products only	—
`shopVerified`	boolean	No	Verified shops only	—

Success Response: Same structure as the Main Feed endpoint.

Standard Error Types:

- `500 INTERNAL_SERVER_ERROR`: Unexpected server failure

---

4. Hot Deals

Purpose: Returns products with any form of discount — regular sale price **or** an active group purchase discount — ranked by the best available saving. A product with a 35% group discount ranks higher than one with a 15% regular sale, even if it has no `comparePrice` set.

Endpoint: `GET` `api/v1/e-commerce/marketplace/hot-deals`

Access Level: Public

Authentication: None required

Scoring applied:

```
┌──────────────────────────────────────────────────────────────────┐
│                  EFFECTIVE DISCOUNT FORMULA                      │
├──────────────────────────────────────────────────────────────────┤
│                                                                  │
│  salePct      = (comparePrice - price) / comparePrice            │
│                 0.0 if no comparePrice or comparePrice ≤ price   │
│                                                                  │
│  activeGroupPct = (regularPrice - groupPrice) / regularPrice     │
│                   0.0 if no live OPEN group exists for the product │
│                                                                  │
│  effectiveDiscountPct = max(salePct, activeGroupPct)             │
```

```
│                                                                  │
│  Products included: onSale = true  OR  hasActiveGroup = true     │
│  Sorted by: effectiveDiscountPct DESC (within each page)         │
└──────────────────────────────────────────────────────────────────┘


Example:
  Product A — regular sale 15% off, no group     → effectivePct = 15%
  Product B — no sale, group discount 35% off    → effectivePct = 35%  ← ranks first
  Product C — sale 20% off + group 40% off       → effectivePct = 40%  ← ranks first
```

Query Parameters:

Parameter	Type	Required	Description	Default
`page`	integer	No	Page number (1-based)	`1`
`size`	integer	No	Items per page	`20`
`categoryId`	UUID	No	Filter by category	—
`minPrice`	decimal	No	Minimum price	—
`maxPrice`	decimal	No	Maximum price	—
`shopVerified`	boolean	No	Verified shops only	—
`inStock`	boolean	No	In-stock only	—

Success Response: Same structure as the Main Feed endpoint. `effectiveDiscountPercentage` is always present and non-null in this feed.

Standard Error Types:

- `500 INTERNAL_SERVER_ERROR`: Unexpected server failure

---

5. New Arrivals

Purpose: Returns recently published products, newest first. Ideal for users who want to discover what just dropped.

Endpoint: GET `api/v1/e-commerce/marketplace/new-arrivals`

Access Level: □□ Public

Authentication: None required

Sorting: `createdAt DESC` — exact database-level sort, no formula applied.

Query Parameters:

Parameter	Type	Required	Description	Default
`page`	integer	No	Page number (1-based)	`1`
`size`	integer	No	Items per page	`20`
`categoryId`	UUID	No	Filter by category	—
`productType`	enum	No	`PHYSICAL` or `DIGITAL`	—
`shopVerified`	boolean	No	Verified shops only	—

Success Response: Same structure as the Main Feed endpoint.

Standard Error Types:

- `500 INTERNAL_SERVER_ERROR`: Unexpected server failure

---

6. Live Group Purchases

Purpose: Returns products that currently have an active OPEN group purchase, sorted by group heat (most seats filled = most urgent = shown first). Useful for showing users time-sensitive social buying opportunities.

Endpoint: `GET` `api/v1/e-commerce/marketplace/live-groups`

Access Level: Public

Authentication: None required

Sorting applied:

```
groupHeat = seatsOccupied / totalSeats

Products sorted by groupHeat DESC — a group at 90% capacity appears
before one at 30%, creating urgency awareness for the user.
```

Query Parameters:

Parameter	Type	Required	Description	Default

`page`	integer	No	Page number (1-based)	`1`
`size`	integer	No	Items per page	`20`

Success Response JSON Sample:


```
{
  "success": true,
  "httpStatus": "OK",
  "message": "Live group purchases retrieved successfully",
  "action_time": "2026-06-04T10:30:45",
  "data": {
    "content": [
      {
        "productId": "3fa85f64-5717-4562-b3fc-2c963f66afa6",
        "productName": "Samsung Galaxy S24",
        "price": 850000.00,
        "hasActiveGroup": true,
        "activeGroupHeat": 0.92,
        "activeGroupPrice": 720000.00,
        "activeGroupSeatsLeft": 2,
        "activeGroupExpiresAt": "2026-06-04T18:00:00"
      }
    ],
    "currentPage": 1,
    "pageSize": 20,
    "totalElements": 37,
    "totalPages": 2,
    "hasNext": true,
    "hasPrevious": false
  }
}
```


Standard Error Types:

- `500 INTERNAL_SERVER_ERROR`: Unexpected server failure

---

7. Advanced Filter

Purpose: The most powerful endpoint in the marketplace. Combines every available filter with an optional keyword search (`?q=`) so users can express highly specific queries like: "*show me NEW Samsung smartphones under 1M from verified shops with an active group deal saving at least 20%, sorted by trending.*"

Endpoint: `GET` `api/v1/e-commerce/marketplace/advanced-filter`

Access Level: □□ Public (personalised scoring when authenticated)

Authentication: Bearer Token (optional — enables `FOR_YOU` and personalised `TRENDING`)

Scoring applied per `sortBy`:

```
┌──────────────┬──────────────────────────────────────────────────┐
│ sortBy       │ How it works                                     │
├──────────────┼──────────────────────────────────────────────────┤
│ TRENDING     │ Full formula re-rank in Java after DB fetch      │
│ FOR_YOU      │ Relevance formula re-rank in Java after DB fetch │
│ NEWEST       │ createdAt DESC — exact DB sort                   │
│ PRICE_ASC    │ price ASC — exact DB sort                        │
│ PRICE_DESC   │ price DESC — exact DB sort                       │
│ MOST_SOLD    │ soldQuantity DESC — exact DB sort                │
│ BEST_DEAL    │ discount % DESC — re-ranked in Java after DB fetch │
│ MOST_VIEWED  │ viewCount DESC — exact DB sort                   │
│ MOST_CARTED  │ cartAddCount DESC — exact DB sort                │
└──────────────┴──────────────────────────────────────────────────┘
```

Post-fetch Java filters (applied after DB query — `totalElements` reflects pre-filter count):

- `hasMultipleColors` — checks `colors.size() > 1` in memory
- `maxGroupSeatsLeft` — checks active group seats remaining in memory
- `minGroupDiscountPercent` — calculates group discount % in memory

Query Parameters:

Group	Parameter	Type	Required	Description	Default
Sort & Page	`sortBy`	enum	No	Sort strategy (see table above)	`TRENDING`
	`page`	integer	No	Page number (1-based)	`1`
	`size`	integer	No	Items per page	`20`

Group	Parameter	Type	Required	Description	Default
Search	`q`	string	No	Keyword — searches product name and description (case-insensitive LIKE)	—
Price	`minPrice`	decimal	No	Minimum price	—
	`maxPrice`	decimal	No	Maximum price	—
Product	`categoryId`	UUID	No	Filter by product category	—
	`condition`	enum	No	`NEW`, `USED`, `REFURBISHED`	—
	`productType`	enum	No	`PHYSICAL`, `DIGITAL`	—
	`urgencyTag`	enum	No	`NONE`, `LIMITED_TIME`, `LOW_STOCK`, `FLASH_SALE`	—
	`hasMultipleColors`	boolean	No	`true` = colour variant products only *(post-fetch)*	—
Availability	`inStock`	boolean	No	`true` = in-stock only	—
	`minStockQuantity`	integer	No	Minimum stock units (bulk buyers)	—
Deals	`onSale`	boolean	No	`true` = discounted only	—
	`hasGroupBuying`	boolean	No	`true` = group buying enabled on product	—
	`hasActiveGroup`	boolean	No	`true` = live OPEN group right now	—
	`maxGroupSeatsLeft`	integer	No	Max seats remaining in active group *(post-fetch)*	—
	`minGroupDiscountPercent`	integer	No	Min group discount % e.g. `20` *(post-fetch)*	—
	`hasInstallments`	boolean	No	`true` = instalment plans available	—

Group	Parameter	Type	Required	Description	Default
Shop Trust	`shopVerified`	boolean	No	`true` = verified shops only	—
	`minTrustScore`	decimal	No	Min shop trust score e.g. `4.00` (0.00–5.00)	—
Popularity	`minSoldCount`	integer	No	Min total units sold	—

Example Requests:

Find trending Samsung phones on sale from verified shops:

```
GET /marketplace/advanced-filter
  ?q=samsung
  &categoryId=a1b2c3d4-...
  &onSale=true
  &shopVerified=true
  &sortBy=TRENDING
```

Find live group deals saving at least 25% with fewer than 5 seats left:

```
GET /marketplace/advanced-filter
  ?hasActiveGroup=true
  &minGroupDiscountPercent=25
  &maxGroupSeatsLeft=5
  &sortBy=BEST_DEAL
```

Find digital products under 50,000 newly published, for anonymous browsing:

```
GET /marketplace/advanced-filter
  ?productType=DIGITAL
  &maxPrice=50000
  &sortBy=NEWEST
```

Success Response: Same structure as the Main Feed endpoint.

Standard Error Types:

- `500 INTERNAL_SERVER_ERROR`: Unexpected server failure

MarketplaceProductResponse — Full Field Reference

Field	Type	Nullable	Description
`productId`	UUID	No	Unique product identifier
`productName`	string	No	Product display name
`productSlug`	string	No	URL-safe unique slug
`productPrimaryMedia`	object	Yes	Resolved primary media — `{fileId, mediaType, status, thumbUrl, mimeType}`. Same lean shape used in cart/wishlist/checkout. `null` if no media
`productType`	enum	No	`PHYSICAL` or `DIGITAL`
`price`	decimal	No	Current selling price
`comparePrice`	decimal	Yes	Original price — present only when product is on sale
`discountPercentage`	decimal	Yes	Regular sale discount % — `(comparePrice - price) / comparePrice × 100`. `null` if not on sale
`effectiveDiscountPercentage`	decimal	Yes	Best available deal: `max(salePct, activeGroupPct) × 100`. Covers both regular sale AND live group discounts. This is the field the Hot Deals feed sorts by. `null` if no discount of any kind
`stockQuantity`	integer	No	Available stock units
`soldQuantity`	integer	No	Total units sold
`viewCount`	long	No	Cumulative public views
`cartAddCount`	long	No	Cumulative first-time cart adds
`urgencyTag`	enum	No	`NONE`, `LIMITED_TIME`, `LOW_STOCK`, `FLASH_SALE`
`condition`	enum	No	`NEW`, `USED`, `REFURBISHED`
`inStock`	boolean	No	`true` if `stockQuantity > 0`
`onSale`	boolean	No	`true` if `comparePrice > price`

Field	Type	Nullable	Description
`hasInstallments`	boolean	No	`true` if instalment plans exist
`shopId`	UUID	No	Owning shop identifier
`shopName`	string	No	Shop display name
`shopSlug`	string	No	Shop URL slug
`shopLogoMedia`	object	Yes	Resolved shop logo — `{fileId, status, thumbUrl, mimeType}`. `null` if shop has no logo
`shopVerified`	boolean	No	Whether shop has passed verification
`shopTrustScore`	decimal	No	Shop trust rating `0.00–5.00`
`categoryId`	UUID	Yes	Product category identifier
`categoryName`	string	Yes	Product category display name
`hasActiveGroup`	boolean	No	`true` if a live OPEN group exists
`activeGroupHeat`	decimal	Yes	Fill ratio `0.0–1.0` of hottest live group
`activeGroupPrice`	decimal	Yes	Discounted group price
`activeGroupSeatsLeft`	integer	Yes	Remaining seats in active group
`activeGroupExpiresAt`	datetime	Yes	Expiry of active group purchase
`createdAt`	datetime	No	When the product was published

Quick Reference — All Marketplace Endpoints

#	Endpoint	Auth	Sort Mode
1	`GET /marketplace/feed`	Optional	All `MarketplaceSortBy` values
2	`GET /marketplace/trending`	Optional	Formula-ranked
3	`GET /marketplace/for-you`	Optional	Relevance-ranked

#	Endpoint	Auth	Sort Mode
4	`GET /marketplace/hot-deals`	None	`effectiveDiscountPercentage` DESC (sale + group)
5	`GET /marketplace/new-arrivals`	None	`createdAt` DESC
6	`GET /marketplace/live-groups`	None	Group heat DESC
7	`GET /marketplace/advanced-filter`	Optional	All `MarketplaceSortBy` values

Revision #4
Created 4 June 2026 09:34:59 by Admin Qbit
Updated 9 July 2026 09:30:40 by Admin Qbit