Booklix Public API

Версія документа: 2026-05-06
API version: v1
Base URL: {BASE_URL}

Цей документ описує публічні API Booklix для партнерських інтеграцій, маркетплейсу, бронювань, ресторанних резервацій, подарункових карт, публічного акаунта клієнта та перевірки платежів.

У прикладах нижче `{businessSlug}` означає slug бізнесу, а `{BASE_URL}` - домен інсталяції Booklix, наприклад `https://app.example.com`.

1. Формат і загальні правила

- Формат даних: JSON, окрім QR endpoint, який повертає PNG.
- Content-Type для POST/PATCH: `application/json`.
- Дати: `YYYY-MM-DD`.
- Час: `HH:mm`, у часовій зоні бізнесу.
- Гроші: усі суми передаються в центах як integer (`priceCents`, `amountCents`, `nominalCents`).
- Валюта: ISO code, наприклад `CHF`.
- Публічні API key endpoints повертають заголовок `X-Booklix-API-Version: v1` і мають no-store cache headers.

Стандартна помилка:

```json
{
  "error": "Bad Request",
  "code": "INVALID_NAME",
  "message": "Category name is required"
}
```

Не всі клієнтські public endpoints мають поле `code`; мінімальна форма помилки:

```json
{
  "error": "Business not found"
}
```

2. Авторизація

Є два типи публічних сценаріїв.

1. Partner/Public API key endpoints
   - Використовують Bearer API key.
   - Ключ прив'язаний до одного бізнесу.
   - Якщо slug у URL не збігається з бізнесом ключа, API повертає `403`.

2. Customer public endpoints
   - Використовуються фронтендом marketplace.
   - Частина endpoint-ів доступна без API key, але створення booking або restaurant reservation вимагає активну customer session cookie.

2.1 Bearer API key

Header:

```http
```

```
Authorization: Bearer sk_bx_live_12345678.full-secret-value
```

Формат ключа:

```text
{prefix}.{secret}
```

Префікси:

- `pk_bx_live_*` / `pk_bx_test_*` - publishable key.
- `sk_bx_live_*` / `sk_bx_test_*` - secret key.
- `rk_bx_live_*` / `rk_bx_test_*` - restricted key.

Підтримані scopes у поточній реалізації:

- `services.read` - читання сервісів, категорій, staff, promotions.
- `services.write` - створення service category через public API.
- `webhooks.read` - читання public webhook endpoints.
- `webhooks.write` - створення, оновлення, вимкнення та ротація public webhook endpoints.
- `availability.read` - scope показується для publishable key, але поточні API key endpoints нижче напряму перевіряють `services.read`.
- `customers.read`, `bookings.write` - scopes показуються для secret key у dashboard, але public API key routes у цьому документі їх поки не вимагають.

Обмеження ключа:

- `allowedOrigins`: якщо список порожній, origin дозволений; інакше request origin має збігатися зі списком або `*`.
- `allowedIps`: якщо список порожній, IP дозволений; інакше IP має збігатися зі списком або `*`.
- Rate limits: `rateLimitPerMinute`, `rateLimitPerHour`.
- Кожен успішний API key запит логиться в `ApiRequestLog`.

Типові auth помилки:

- `401 MISSING_AUTH_HEADER`
- `401 INVALID_AUTH_SCHEME`
- `401 EMPTY_API_KEY`
- `401 INVALID_API_KEY_FORMAT`
- `403 API_KEY_NOT_FOUND`
- `403 MISSING_REQUIRED_SCOPES`
- `403 ORIGIN_NOT_ALLOWED`
- `403 IP_NOT_ALLOWED`
- `429 RATE_LIMITED`

3. Partner endpoints з API key

Усі endpoints у цьому розділі потребують:

```http
Authorization: Bearer {apiKey}
```

3.1 Список сервісів

```http
GET /api/public/v1/businesses/{businessSlug}/services
Scope: services.read
```

Query params:

- `categoryId` optional string.
- `serviceId` optional string.
- `includeVariants` optional boolean, default `true`.
- `includeAddons` optional boolean, default `true`.

Response `200`:

```json
[
  {
    "id": "srv_123",
    "name": "Haircut",
    "publicName": "Classic haircut",
    "displayName": "Classic haircut",
    "description": "Short description",
    "imageUrl": "https://...",
    "durationMin": 60,
    "duration": 60,
    "priceCents": 9000,
    "price": 90,
    "currency": "CHF",
    "bufferBeforeMin": 0,
    "bufferAfterMin": 10,
    "sortOrder": 0,
    "categoryId": "cat_123",
    "category": {
      "id": "cat_123",
      "name": "Hair"
    },
    "standardOption": {
      "id": null,
      "kind": "STANDARD",
      "isStandard": true,
      "name": "Classic haircut",
      "description": "Short description",
      "durationMin": 60,
      "priceCents": 9000,
      "price": 90,
      "sortOrder": 0,
      "isActive": true
    },
    "pricingOptions": [],
    "variants": [],
    "addons": []
  }
]
```

3.2 Service categories: list

```http
GET /api/public/v1/businesses/{businessSlug}/service-categories
Scope: services.read
```

Response `200`:

```json
[
```

```
  {
    "id": "cat_123",
    "name": "Hair",
    "imageUrl": "https://...",
    "sortOrder": 0,
    "isActive": true,
    "slug": "hair"
  }
]
```

3.3 Service categories: create

```http
POST /api/public/v1/businesses/{businessSlug}/service-categories
Scope: services.write
```

Request:

```json
{
  "name": "Hair",
  "imageUrl": "https://example.com/hair.jpg",
  "sortOrder": 0,
  "isActive": true
}
```

Validation:

- `name` is required.
- Duplicate category name for the same business returns `409`.
- If `sortOrder` is omitted, API appends the category after the current last category.

Response `201`:

```json
{
  "id": "cat_123",
  "name": "Hair",
  "imageUrl": "https://example.com/hair.jpg",
  "sortOrder": 0,
  "isActive": true,
  "slug": "hair",
  "createdAt": "2026-05-06T10:00:00.000Z",
  "updatedAt": "2026-05-06T10:00:00.000Z"
}
```

3.4 Staff

```http
GET /api/public/v1/businesses/{businessSlug}/staff
Scope: services.read
```

Query params:

- `onlyAssignedToServices=true` optional.
- `withServices=true` optional alias for `onlyAssignedToServices=true`.

Response `200`:

```json
[
  {
    "id": "staff_123",
    "name": "Anna",
    "title": "Senior stylist",
    "imageUrl": "https://...",
    "sortOrder": 0,
    "serviceIds": ["srv_123"],
    "services": [
      {
        "id": "srv_123",
        "name": "Classic haircut",
        "category": {
          "id": "cat_123",
          "name": "Hair"
        }
      }
    ]
  }
]
```

Staff records with hidden internal emails are filtered out.

3.5 Staff for category

```http
GET /api/public/v1/businesses/{businessSlug}/service-categories/{categoryId}/sta
ff
Scope: services.read
```

Response `200`:

```json
{
  "category": {
    "id": "cat_123",
    "name": "Hair",
    "imageUrl": "https://..."
  },
  "staff": []
}
```

If category is missing or inactive, response is `404`.

3.6 Promotions

```http
GET /api/public/v1/businesses/{businessSlug}/promotions
Scope: services.read
```

Query params:

- `active=false` optional. Default is active promotions only.
- `type` optional promotion type.

- `scope` optional promotion scope.
- `categoryId` optional.
- `serviceId` optional. Filters promotions applicable to a service.

Response `200`:

```json
[
  {
    "id": "promo_123",
    "name": "Spring offer",
    "description": "Short description",
    "imageUrl": "https://...",
    "badgeLabel": "Deal",
    "type": "PERCENT",
    "scope": "SERVICE",
    "serviceIds": ["srv_123"],
    "discountPercent": 20,
    "discountAmountCents": null,
    "promotionalPriceCents": null,
    "bonusTitle": null,
    "startsAt": "2026-05-01T00:00:00.000Z",
    "endsAt": "2026-05-31T23:59:59.000Z",
    "activeDays": [1, 2, 3, 4, 5],
    "dailyStartTime": "09:00",
    "dailyEndTime": "18:00",
    "isActive": true,
    "category": null,
    "service": {
      "id": "srv_123",
      "name": "Classic haircut"
    },
    "services": [
      {
        "id": "srv_123",
        "name": "Classic haircut"
      }
    ]
  }
]
```

3.7 Public webhooks

Public webhooks дозволяють партнерській інтеграції отримувати server-to-server події після успішних public booking flows та після змін публічного catalog business-даних у Booklix.

Підтримані events:

- `booking.created`
- `reservation.created`
- `catalog.changed`
- `service.created`
- `service.updated`
- `service.deleted`
- `service.reordered`
- `service_price.updated`
- `service_variant.changed`
- `service_category.created`
- `service_category.updated`
- `service_category.deleted`

- `service_category.reordered`
- `promotion.created`
- `promotion.updated`
- `promotion.deleted`
- `staff.created`
- `staff.updated`
- `staff.deleted`
- `staff.reordered`
- `addon.created`
- `addon.updated`
- `addon.deleted`

Для website sync можна підписатися лише на `catalog.changed`, щоб отримувати будь-яку зміну catalog, або на granular events, якщо потрібна точніша реакція.

List webhook endpoints

```http
GET /api/public/v1/businesses/{businessSlug}/webhooks
Scope: webhooks.read
```

Response `200`:

```json
[
  {
    "id": "wh_123",
    "businessId": "biz_123",
    "name": "Partner CRM",
    "url": "https://partner.example.com/booklix/webhooks",
    "events": ["booking.created"],
    "status": "ACTIVE",
    "lastDeliveredAt": null,
    "lastFailedAt": null,
    "failureCount": 0,
    "createdAt": "2026-05-10T10:00:00.000Z",
    "updatedAt": "2026-05-10T10:00:00.000Z"
  }
]
```

Create webhook endpoint

```http
POST /api/public/v1/businesses/{businessSlug}/webhooks
Scope: webhooks.write
```

Request:

```json
{
  "name": "Partner CRM",
  "url": "https://partner.example.com/booklix/webhooks",
  "events": ["booking.created", "reservation.created"]
}
```

Response `201` includes `signingSecret`. The secret is only returned on create or explicit rotation.

Update, disable, or rotate

```http
PATCH /api/public/v1/businesses/{businessSlug}/webhooks/{id}
Scope: webhooks.write
```

Request fields are optional:

```json
{
  "name": "Partner CRM production",
  "url": "https://partner.example.com/booklix/webhooks",
  "events": ["booking.created"],
  "status": "ACTIVE",
  "rotateSecret": true
}
```

`DELETE /api/public/v1/businesses/{businessSlug}/webhooks/{id}` disables the endpoint and returns `{ "ok": true }`.

Delivery format

Booklix sends a `POST` with JSON body and these headers:

- `X-Booklix-Event`: event name.
- `X-Booklix-Delivery`: delivery id.
- `X-Booklix-Signature`: `t={unixTimestamp},v1={hmacSha256}`.

Signature verification uses HMAC-SHA256 over:

```text
{unixTimestamp}.{rawRequestBody}
```

Example payload:

```json
{
  "id": "evt_123",
  "type": "booking.created",
  "createdAt": "2026-05-10T10:00:00.000Z",
  "business": {
    "id": "biz_123",
    "slug": "madame-figaro",
    "name": "Madame Figaro",
    "timezone": "Europe/Zurich"
  },
  "data": {
    "bookingIds": ["booking_123"],
    "bookings": [
      {
        "id": "booking_123",
        "status": "CONFIRMED",
        "source": "PUBLIC",
        "startAt": "2026-05-10T08:00:00.000Z",
        "endAt": "2026-05-10T09:00:00.000Z",
        "guest": {
          "name": "Anna Keller",
          "email": "anna@example.com",
          "phone": "+41790000000"
```

```
        },
        "service": {
          "id": "srv_123",
          "name": "Classic haircut",
          "internalName": "Haircut"
        },
        "totals": {
          "amountCents": 9000,
          "currency": "CHF"
        }
      }
    ]
  }
}
```

Catalog webhook payload example:

```json
{
  "id": "evt_456",
  "type": "service.updated",
  "createdAt": "2026-05-10T10:05:00.000Z",
  "business": {
    "id": "biz_123",
    "slug": "onsen-head-spa-beauty",
    "name": "Onsen Head Spa & Beauty",
    "timezone": "Europe/Zurich"
  },
  "data": {
    "resource": {
      "type": "service",
      "id": "srv_123",
      "ids": ["srv_123"]
    },
    "action": "updated",
    "changed": {
      "price": true,
      "variants": true,
      "addons": false,
      "staffAssignments": false,
      "resourceRequirements": false
    },
    "object": {
      "id": "srv_123",
      "name": "Japanese Head Spa",
      "publicName": null,
      "description": "Relaxing scalp treatment with massage and deep cleansing."
,
      "imageUrl": null,
      "durationMin": 60,
      "priceCents": 12000,
      "currency": "CHF",
      "bufferBeforeMin": 0,
      "bufferAfterMin": 0,
      "sortOrder": 0,
      "status": "ACTIVE",
      "categoryId": "cat_123",
      "deletedAt": null,
      "updatedAt": "2026-05-10T10:05:00.000Z",
      "category": {
        "id": "cat_123",
```

```
        "name": "Head Spa"
      },
      "variants": [],
      "addonLinks": []
    }
  }
}
```

For reorder events, `resource.id` is `null`, `resource.ids` contains all changed ids, and `data.items` contains the submitted order payload.

4. Marketplace booking endpoints

Ці endpoints не використовують API key. Вони призначені для публічного booking flow. Створення booking потребує customer session cookie.

4.1 Availability for service booking

```http
GET /api/public/v1/businesses/{businessSlug}/availability
```

Query params:

- `date` required, `YYYY-MM-DD`.
- `serviceIds` required, comma-separated або repeated serialized value from frontend helper.
- `services` optional alias for `serviceIds`.
- `service` optional alias for a single service id.
- `staffId` optional preferred staff id.
- `config` optional JSON string with selected variants/add-ons.

Response `200`:

```json
{
  "date": "2026-05-10",
  "serviceIds": ["srv_123"],
  "slots": [
    {
      "start": "09:00",
      "end": "10:00",
      "serviceIds": ["srv_123"]
    }
  ]
}
```

Validation errors:

- Missing or invalid `date` or `serviceIds`: `400`.
- Business not found: `404`.
- Unknown service: `404`.

4.2 Validate checkout discount

```http
POST /api/public/v1/businesses/{businessSlug}/checkout-discount
```

Request:

```json
{
  "serviceIds": ["srv_123"],
  "config": "[{\"serviceId\":\"srv_123\",\"variantId\":null,\"addons\":[]}]",
  "code": "PROMO20"
}
```

Response `200`:

```json
{
  "code": "PROMO20",
  "kind": "PROMO_CODE",
  "title": "Spring offer",
  "description": "20% discount",
  "amountCents": 1800,
  "originalTotalCents": 9000,
  "adjustedTotalCents": 7200,
  "currency": "CHF",
  "remainingBalanceCents": null
}
```

Special response:

- `409` can include `autoAddServiceId` and `autoAddServiceTitle` when a gift card requires the matching service in the booking.

Rate limit:

- Failed discount attempts are rate-limited and may return `429` with `Retry-After`.

4.3 Create booking

```http
POST /api/public/v1/businesses/{businessSlug}/bookings
Requires: customer session cookie
Optional header: x-booklix-locale
```

Request:

```json
{
  "serviceIds": ["srv_123"],
  "config": "[{\"serviceId\":\"srv_123\",\"variantId\":null,\"addons\":[]}]",
  "staffId": "staff_123",
  "date": "2026-05-10",
  "start": "09:00",
  "guestName": "Anna Customer",
  "guestEmail": "anna@example.com",
  "guestPhone": "+41790000000",
  "notes": "Please be on time",
  "discountCode": "PROMO20",
  "privacyConsentAccepted": true
}
```

Important:

- User must be signed in. If not, response is `401`.
- `privacyConsentAccepted` must be `true`.
- `serviceIds`, `date`, `start`, `guestName`, `guestEmail` are required.
- Start time must be in the future.
- If the business requires confirmation, booking status is `PENDING`;
  otherwise `CONFIRMED`.

Response `201`:

```json
{
  "ids": ["booking_123"],
  "count": 1,
  "status": "CONFIRMED",
  "clientId": "client_123",
  "totalPriceCents": 7200,
  "payment": {
    "id": "pay_123",
    "status": "PENDING",
    "amountCents": 2000,
    "currency": "CHF",
    "lookupToken": "payment-lookup-token"
  },
  "appliedDiscount": {
    "code": "PROMO20",
    "kind": "PROMO_CODE",
    "amountCents": 1800
  }
}
```

Rate limit:

- Max 8 booking attempts per 15 minutes per signed-in user/business.
- Response `429` includes `Retry-After`.

5. Restaurant reservation endpoints

These endpoints are for restaurant-style table reservations.

5.1 Reservation availability

```http
GET /api/public/v1/businesses/{businessSlug}/reservations/availability
```

Query params:

- `date` required, `YYYY-MM-DD`.
- `partySize` required integer.

Response `200`:

```json
[
  {
    "date": "2026-05-10",
    "time": "19:00",
    "available": true,
    "tablePreview": {
      "id": "table_123",
```

```
      "name": "T1",
      "zoneId": "zone_123",
      "zone": "Terrace",
      "zoneImageUrl": "https://...",
      "capacity": 4
    },
    "reason": null
  }
]
```

5.2 Seating options

```http
GET /api/public/v1/businesses/{businessSlug}/reservations/seating
```

Query params:

- `date` required.
- `time` required.
- `partySize` required integer.

Response `200`:

```json
{
  "date": "2026-05-10",
  "time": "19:00",
  "partySize": 2,
  "groups": [
    {
      "id": "zone_123",
      "kind": "ZONE",
      "name": "Terrace",
      "imageUrl": "https://...",
      "tables": [
        {
          "id": "table_123",
          "name": "T1",
          "zoneId": "zone_123",
          "zoneName": "Terrace",
          "zoneImageUrl": "https://...",
          "capacity": 4
        }
      ]
    }
  ]
}
```

5.3 Create reservation

```http
POST /api/public/v1/businesses/{businessSlug}/reservations
Requires: customer session cookie
Optional header: x-booklix-locale
```

Request:

```json
```

```
{
  "date": "2026-05-10",
  "time": "19:00",
  "tableId": "table_123",
  "zoneId": "zone_123",
  "partySize": 2,
  "guestName": "Anna Customer",
  "guestEmail": "anna@example.com",
  "guestPhone": "+41790000000",
  "notes": "Window seat if possible",
  "privacyConsentAccepted": true
}
```

Important:

- User must be signed in with an email.
- `privacyConsentAccepted` must be `true`.
- API checks that requested slot is available and table is online bookable.
- If `tableId` is omitted, the first available table in the requested zone is selected.

Response `200`:

```json
{
  "id": "booking_123",
  "startAt": "2026-05-10T17:00:00.000Z",
  "endAt": "2026-05-10T18:30:00.000Z",
  "status": "CONFIRMED",
  "partySize": 2,
  "guest": {
    "name": "Anna Customer",
    "email": "anna@example.com",
    "phone": "+41790000000"
  },
  "table": {
    "id": "table_123",
    "name": "T1",
    "zoneId": "zone_123",
    "zone": "Terrace",
    "zoneImageUrl": "https://...",
    "capacity": 4
  },
  "notes": "Window seat if possible"
}
```

Rate limit:

- Max 8 reservation attempts per 15 minutes per signed-in user/business.

6. Gift cards

6.1 Create gift card request

```http
POST /api/public/v1/businesses/{businessSlug}/gift-cards
Optional header: Idempotency-Key
```

Idempotency:

- `Idempotency-Key` header or `idempotencyKey` body field.
- Allowed pattern: `A-Z`, `a-z`, `0-9`, `.`, `_`, `:`, `-`.
- Length: 8 to 120 chars.
- Stored response can be reused for 24 hours.

Money gift card request:

```json
{
  "type": "MONEY",
  "nominalCents": 10000,
  "receiverName": "Anna",
  "receiverEmail": "anna@example.com",
  "senderName": "Mark",
  "senderEmail": "mark@example.com",
  "message": "Happy birthday",
  "deliveryDate": "2026-05-10"
}
```

Service gift card request:

```json
{
  "type": "SERVICE",
  "serviceId": "srv_123",
  "serviceSelections": [
    {
      "serviceId": "srv_123",
      "variantId": null,
      "addons": [
        {
          "addonId": "addon_123",
          "quantity": 1
        }
      ]
    }
  ],
  "receiverName": "Anna",
  "receiverEmail": "anna@example.com",
  "senderName": "Mark",
  "senderEmail": "mark@example.com"
}
```

Response `201`:

```json
{
  "id": "gift_123",
  "status": "PENDING_PAYMENT",
  "payment": {
    "id": "pay_123",
    "status": "PENDING",
    "amountCents": 10000,
    "currency": "CHF",
    "lookupToken": "payment-lookup-token"
  }
}
```

Validation:

- `type` must be `MONEY` or `SERVICE`.
- Receiver name, sender name and sender email are required.
- Money gift card max is `200000` cents.
- Service gift card requires valid active service selections.
- Services with zero/negative price cannot be sold as online gift cards.

Rate limit:

- Max 5 gift card create attempts per 15 minutes per business/request fingerprint.

7. Public customer account

These endpoints use the customer session cookie.

7.1 Current customer account

```http
GET /api/public/v1/me
```

If not authenticated:

```json
{
  "user": null,
  "bookings": [],
  "giftCards": []
}
```

Authenticated response:

```json
{
  "user": {
    "id": "user_123",
    "name": "Anna Customer",
    "email": "anna@example.com",
    "phone": "+41790000000",
    "phoneVerified": true,
    "locale": "de"
  },
  "bookings": [
    {
      "id": "booking_123",
      "status": "CONFIRMED",
      "startAt": "2026-05-10T09:00:00.000Z",
      "endAt": "2026-05-10T10:00:00.000Z",
      "business": {
        "name": "Booklix Demo",
        "slug": "booklix-demo"
      },
      "service": {
        "name": "Haircut",
        "publicName": "Classic haircut"
      }
    }
  ],
  "giftCards": []
```

}
```

Notes:

- Returns up to 20 latest bookings by matching guest/client email.
- Gift card code is returned only for statuses `ACTIVE`, `PARTIALLY_USED`, `USED`; otherwise `code` is `null`.

### 7.2 Update customer profile

```http
PATCH /api/public/v1/me
Requires: customer session cookie
```

Request:

```json
{
  "firstName": "Anna",
  "lastName": "Customer",
  "phone": "+41790000000"
}
```

Response `200`:

```json
{
  "user": {
    "id": "user_123",
    "name": "Anna Customer",
    "email": "anna@example.com",
    "phone": "+41790000000",
    "phoneVerified": false
  }
}
```

Validation:

- Invalid phone returns `400`.
- Phone already used by another account returns `409`.
- If phone changes, `phoneVerified` is reset to `false`.

7.3 Export personal data

```http
GET /api/public/v1/me/export
Requires: customer session cookie
```

Response:

- JSON payload with personal data export.
- Header: `Content-Disposition: attachment; filename="booklix-personal-data-{user
Id}.json"`.
- Header: `Cache-Control: no-store`.

7.4 Delete/anonymize personal data
```

```http
DELETE /api/public/v1/me
Requires: customer session cookie
```

Response `200`:

```json
{
  "ok": true,
  "affectedRecords": 12
}
```

8. Booking confirmation and cancellation

8.1 Booking QR code

```http
GET /api/public/v1/bookings/qr?token={confirmationToken}&locale={locale}
```

Response:

- `200 image/png`
- Width: 480 px.
- Invalid token returns `400`.

The QR code points to the public booking confirmation page.

8.2 Cancellation request

```http
POST /api/public/v1/bookings/cancellation-request
```

Request:

```json
{
  "token": "booking-confirmation-token"
}
```

Response `200`:

```json
{
  "ok": true,
  "count": 1
}
```

If cancellation was already requested:

```json
{
  "ok": true,
  "count": 0,
  "alreadyRequested": true
}
```

Validation:

- Missing token: `400`.
- Confirmation not found: `404`.
- Booking already cancelled/completed/no-show: `400`.

Rate limit:

- Max 5 cancellation requests per 15 minutes per business/request fingerprint.

9. Public payment lookup

```http
GET /api/public/v1/payments/{paymentId}?token={lookupToken}
```

Response `200`:

```json
{
  "id": "pay_123",
  "provider": "MANUAL",
  "status": "PENDING",
  "amountCents": 10000,
  "currency": "CHF",
  "bookingId": "booking_123",
  "giftCards": [
    {
      "id": "gift_123",
      "status": "PENDING_PAYMENT"
    }
  ],
  "createdAt": "2026-05-06T10:00:00.000Z",
  "updatedAt": "2026-05-06T10:00:00.000Z"
}
```

Invalid or missing token returns `404 Payment not found`.

10. cURL examples

Read services with API key

```bash
curl -X GET "{BASE_URL}/api/public/v1/businesses/{businessSlug}/services?include
Variants=true&includeAddons=true" \
  -H "Authorization: Bearer sk_bx_live_12345678.full-secret-value"
```

Create service category with API key

```bash
curl -X POST "{BASE_URL}/api/public/v1/businesses/{businessSlug}/service-categor
ies" \
  -H "Authorization: Bearer sk_bx_live_12345678.full-secret-value" \
  -H "Content-Type: application/json" \
  -d '{"name":"Hair","isActive":true}'
```

Check booking availability

```bash
curl -X GET "{BASE_URL}/api/public/v1/businesses/{businessSlug}/availability?dat
e=2026-05-10&serviceIds=srv_123"
```

Validate promo code

```bash
curl -X POST "{BASE_URL}/api/public/v1/businesses/{businessSlug}/checkout-discou
nt" \
  -H "Content-Type: application/json" \
  -d '{"serviceIds":["srv_123"],"config":"[]","code":"PROMO20"}'
```

Lookup payment status

```bash
curl -X GET "{BASE_URL}/api/public/v1/payments/pay_123?token=payment-lookup-toke
n"
```

Create webhook endpoint

```bash
curl -X POST "{BASE_URL}/api/public/v1/businesses/{businessSlug}/webhooks" \
  -H "Authorization: Bearer sk_bx_live_12345678.full-secret-value" \
  -H "Content-Type: application/json" \
  -d '{"name":"Onsen website sync","url":"https://onsen-headspa.ch/booklix/webho
oks","events":["catalog.changed","booking.created"]}'
```

11. Endpoint summary

API key protected:

- `GET /api/public/v1/businesses/{businessSlug}/services`
- `GET /api/public/v1/businesses/{businessSlug}/service-categories`
- `POST /api/public/v1/businesses/{businessSlug}/service-categories`
- `GET /api/public/v1/businesses/{businessSlug}/staff`
- `GET /api/public/v1/businesses/{businessSlug}/service-categories/{categoryId}/
staff`
- `GET /api/public/v1/businesses/{businessSlug}/promotions`
- `GET /api/public/v1/businesses/{businessSlug}/webhooks`
- `POST /api/public/v1/businesses/{businessSlug}/webhooks`
- `PATCH /api/public/v1/businesses/{businessSlug}/webhooks/{id}`
- `DELETE /api/public/v1/businesses/{businessSlug}/webhooks/{id}`

Marketplace/customer:

- `GET /api/public/v1/businesses/{businessSlug}/availability`
- `POST /api/public/v1/businesses/{businessSlug}/checkout-discount`
- `POST /api/public/v1/businesses/{businessSlug}/bookings`
- `GET /api/public/v1/businesses/{businessSlug}/reservations/availability`
- `GET /api/public/v1/businesses/{businessSlug}/reservations/seating`
- `POST /api/public/v1/businesses/{businessSlug}/reservations`
- `POST /api/public/v1/businesses/{businessSlug}/gift-cards`
- `GET /api/public/v1/me`
- `PATCH /api/public/v1/me`
- `DELETE /api/public/v1/me`
- `GET /api/public/v1/me/export`
- `GET /api/public/v1/bookings/qr`
- `POST /api/public/v1/bookings/cancellation-request`

- `GET /api/public/v1/payments/{paymentId}`

12. Security notes for public sharing

- Не публікувати real secret/restricted keys у документації.
- Для партнерів краще видавати restricted key з мінімальними scopes.
- Для браузерних інтеграцій використовувати publishable key лише там, де це явно дозволено продуктом.
- Для server-to-server інтеграцій використовувати secret або restricted key.
- Вмикати `allowedOrigins` для браузерних інтеграцій.
- Вмикати `allowedIps` для server-to-server інтеграцій, якщо IP партнерів стабільні.
- Обробляти `429` і поважати `Retry-After`.
- Усі booking, reservation, gift-card, payment flows повинні відображати суму в центах і валюту з відповіді API.