

API Reference

API Reference

Project: {{PROJECT_NAME}} Version: {{VERSION}} Date: {{DATE}}
Author: {{AUTHOR}} Status: Draft | In Review | Approved Reviewers:
{{REVIEWERS}}

Document History

Version	Date	Author	Changes
0.1	{{DATE}}	{{AUTHOR}}	Initial draft

1. API Overview & Conventions

API style: {{RESTful HTTP/JSON | GraphQL | gRPC}} **API version strategy:** {{URL versioning: /v1/ | Header versioning}} **OpenAPI spec:** {{https://api.domain.com/docs/openapi.json}} **Interactive docs:** {{https://api.domain.com/docs}}

Design conventions:

- Resources named as plural nouns: `/users`, `/orders`, `/products`
- HTTP methods map to CRUD: GET (read), POST (create), PUT (replace), PATCH (update), DELETE (remove)
- Response format: always JSON
- Timestamps: ISO 8601 UTC (`2024-01-15T10:30:00Z`)
- IDs: UUID v4 strings
- Booleans: `true` / `false` (never `1` / `0`)
- Empty collections: `[]` (never `null`)
- Missing optional fields: omitted (never `null` unless semantically null)

2. Base URLs

Environment	Base URL
Development	<code>http://localhost:4000/v1</code>
Staging	<code>https://api-staging.{{domain.com}}/v1</code>
Production	<code>https://api.{{domain.com}}/v1</code>

3. Authentication

Method: Bearer Token (JWT)

Obtain tokens: `POST /auth/login` (see Auth section below)

Include in requests:

```
Authorization: Bearer <access_token>
```

Token lifetimes:

- Access token: 15 minutes
- Refresh token: 30 days (rotate on use)

Refresh tokens: `POST /auth/refresh` with `{ "refreshToken": "..."}` in body.

API Key authentication (machine-to-machine):

```
X-API-Key: <api_key>
```

API keys are scoped and managed at `https://dashboard.domain.com/api-keys`.

4. Common Request/Response Headers

4.1 Request Headers

Header	Required	Description
Authorization	Yes (auth routes)	Bearer <token>
Content-Type	Yes (POST/PUT/PATCH)	application/json
Accept	No	application/json (default)
X-Request-ID	No	Client-provided idempotency ID
Accept-Language	No	en, nb, etc. — affects response locale

4.2 Response Headers

Header	Description
Content-Type	application/json; charset=utf-8
X-Request-ID	Echo of client request ID (or server-generated)
X-RateLimit-Limit	Total requests allowed in window
X-RateLimit-Remaining	Remaining requests in current window
X-RateLimit-Reset	Unix timestamp when window resets
Retry-After	Seconds to wait (set when 429 returned)

5. Error Response Format

```
{
  "error": {
    "code": "VALIDATION_ERROR",
    "message": "Request validation failed",
    "details": [
      {
        "field": "email",
        "message": "Invalid email format",
        "code": "INVALID_FORMAT"
      }
    ],
    "requestId": "req_7f3a2b1c",
    "timestamp": "2024-01-15T10:30:00Z"
  }
}
```

Standard error codes:

HTTP Status	Error Code	Meaning
400	VALIDATION_ERROR	Request body / params failed validation
400	BAD_REQUEST	Malformed request
401	UNAUTHORIZED	Missing or invalid authentication
401	TOKEN_EXPIRED	JWT has expired — refresh required
403	FORBIDDEN	Authenticated but lacks permission
404	NOT_FOUND	Resource does not exist
409	CONFLICT	Resource already exists / version conflict
422	UNPROCESSABLE	Valid format but business rule violation
429	RATE_LIMITED	Too many requests
500	INTERNAL_ERROR	Unexpected server error
503	SERVICE_UNAVAILABLE	Temporary downtime

6. Resources

6.1 Authentication

POST /auth/login

Authenticate user and receive token pair.

Auth required: No

Request body:

```
{
  "email": "user@example.com",
  "password": "{{password}}"
}
```

Response `200 OK`:

```
{
  "accessToken": "eyJhbGciOiJIUzI1NiJ9...",
  "refreshToken": "dGhpcyBpcyBhIHJlZnJlc2g...",
  "expiresIn": 900,
  "user": {
    "id": "usr_01HX7...",
    "email": "user@example.com",
    "name": "John Doe",
    "role": "user"
  }
}
```

Error responses:

Status	Code	Condition
401	INVALID_CREDENTIALS	Wrong email or password
429	RATE_LIMITED	> 5 failed attempts in 15 min

POST /auth/refresh

Rotate access and refresh tokens.

Auth required: No

Request body:

```
{ "refreshToken": "dGhpcyBpcyBhIHJlZnJlc2g..." }
```

Response `200 OK`: Same as login response.

POST /auth/logout

Revoke refresh token.

Auth required: Yes (Bearer)

Request body:

```
{ "refreshToken": "dGhpcyBpcyBhIHJlZnJlc2g..." }
```

Response `204 No Content`

6.2 Users

Endpoints:

Method	Path	Description	Auth
GET	/users	List users (paginated)	Admin
GET	/users/:id	Get user by ID	Self or Admin
POST	/users	Create user	Admin
PATCH	/users/:id	Update user fields	Self or Admin
DELETE	/users/:id	Delete user (soft)	Admin
GET	/users/me	Get current user	Authenticated
PATCH	/users/me	Update current user	Authenticated

GET /users

List users with pagination and filtering.

Auth required: Admin

Query parameters:

Parameter	Type	Default	Description
page	integer	1	Page number
pageSize	integer	25	Items per page (max: 100)
search	string	—	Search name or email (min 2 chars)
role	string	—	Filter by role: admin, user, viewer
status	string	active	Filter by status: active, inactive, all
sort	string	createdAt	Sort field
dir	string	desc	Sort direction: asc, desc

Response 200 OK:

```
{
  "data": [
    {
```

```
    "id": "usr_01HX7...",
    "email": "user@example.com",
    "name": "Jane Doe",
    "role": "user",
    "status": "active",
    "createdAt": "2024-01-15T10:30:00Z",
    "updatedAt": "2024-01-15T10:30:00Z"
  }
],
"pagination": {
  "page": 1,
  "pageSize": 25,
  "total": 142,
  "totalPages": 6
}
}
```

GET /users/:id

Auth required: Self or Admin

Path parameters:

Parameter	Type	Description
id	UUID	User ID

Response **200 OK**:

```
{
  "id": "usr_01HX7...",
  "email": "user@example.com",
  "name": "Jane Doe",
  "role": "user",
  "status": "active",
  "profile": {
    "avatarUrl": "https://cdn.domain.com/avatars/...",
    "bio": "Software developer"
  },
  "createdAt": "2024-01-15T10:30:00Z",
  "updatedAt": "2024-01-15T10:30:00Z"
}
```

```
}
```

Error responses:

Status	Code	Condition
404	NOT_FOUND	User does not exist
403	FORBIDDEN	Non-admin accessing another user

POST /users

Auth required: Admin

Request body:

```
{
  "email": "newuser@example.com",
  "name": "New User",
  "role": "user",
  "password": "{{password}}"
}
```

Response `201 Created`: Full user object (same as GET /users/:id)

PATCH /users/:id

Auth required: Self or Admin

Request body (all fields optional):

```
{
  "name": "Updated Name",
  "profile": {
    "bio": "Updated bio"
  }
}
```

Response `200 OK`: Updated user object.

DELETE /users/:id

Soft-deletes user (sets `deletedAt`, anonymizes PII).

Auth required: Admin

Response 204 No Content

6.3 {{RESOURCE_NAME}}

Endpoints:

Method	Path	Description	Auth
GET	/{{resource}}	List {{resource}}	{{Auth level}}
GET	/{{resource}}/:id	Get by ID	{{Auth level}}
POST	/{{resource}}	Create	{{Auth level}}
PATCH	/{{resource}}/:id	Update	{{Auth level}}
DELETE	/{{resource}}/:id	Delete	{{Auth level}}

TODO: Document all endpoints for this resource following the pattern above.

7. Pagination Format

All list endpoints return the same pagination envelope:

```
{
  "data": [...],
  "pagination": {
    "page": 1,
    "pageSize": 25,
    "total": 142,
    "totalPages": 6,
    "hasNextPage": true,
    "hasPreviousPage": false
  }
}
```

Cursor pagination (high-performance, for infinite scroll):

```
GET /feed?cursor=eyJpZCI6MTIzfQ&pageSize=20
```

Response includes `nextCursor` — pass as `cursor` in next request.

8. Filtering & Sorting Conventions

Filter parameters:

```
GET /orders?status=pending&createdAt[gte]=2024-01-01&total[lte]=1000
```

Operator	Suffix	Example
Equals	(none)	?status=active
Greater than	[gt]	?price[gt]=100
Greater than or equal	[gte]	?price[gte]=100
Less than	[lt]	?price[lt]=500
Less than or equal	[lte]	?price[lte]=500
In list	[in]	?status[in]=active,pending
Not in list	[nin]	?status[nin]=deleted

Sort: ?sort=createdAt&dir=desc (default: createdAt desc)

9. Webhooks Documentation

Webhook endpoint: Configured per-account at `{{https://dashboard.domain.com/webhooks}}`

Delivery: HTTP POST with JSON body, signed with HMAC-SHA256.

Signature verification:

```
const signature = req.headers['x-webhook-signature'];
const computed = crypto
  .createHmac('sha256', webhookSecret)
  .update(rawBody)
  .digest('hex');
const valid = crypto.timingSafeEqual(
  Buffer.from(signature), Buffer.from(computed)
);
```

Event envelope:

```
{
  "id": "evt_01HX7...",
  "type": "user.created",
  "data": { /* resource object */ },
  "timestamp": "2024-01-15T10:30:00Z",
  "version": "1"
}
```

Available events:

Event	Trigger
user.created	New user registered
user.updated	User profile changed
user.deleted	User account deleted
{{resource}}.{{action}}	{{Description}}

Retry policy: 5 retries with exponential backoff. Undeliverable after 24 hours → marked as failed.

10. Rate Limiting

Endpoint Group	Limit	Window
Public endpoints	100 req	1 minute
Authenticated endpoints	1000 req	1 minute
Admin endpoints	5000 req	1 minute
Auth endpoints (login)	5 req	15 minutes
Webhook delivery	N/A	—

Response when rate limited (429 Too Many Requests):

```
{
  "error": {
    "code": "RATE_LIMITED",
    "message": "Too many requests. Please retry after 60 seconds.",
    "retryAfter": 60
  }
}
```

11. Code Examples

cURL

```
# Login
curl -X POST https://api.domain.com/v1/auth/login \
  -H "Content-Type: application/json" \
  -d '{"email": "user@example.com", "password": "secret"}'

# Get users (with token)
curl -X GET "https://api.domain.com/v1/users?page=1&pageSize=10" \
  -H "Authorization: Bearer <access_token>"
```

JavaScript (fetch)

```
const response = await fetch('https://api.domain.com/v1/users', {
  headers: {
    'Authorization': `Bearer ${accessToken}`,
    'Content-Type': 'application/json',
  },
});

if (!response.ok) {
  const error = await response.json();
  throw new Error(error.error.message);
}

const { data, pagination } = await response.json();
```

Python

```
import httpx

client = httpx.Client(
    base_url="https://api.domain.com/v1",
    headers={"Authorization": f"Bearer {access_token}"}
```

)

```
response = client.get("/users", params={"page": 1, "pageSize": 10})  
response.raise_for_status()  
result = response.json()
```

12. SDK Availability

Language	Package	Repository	Status
TypeScript / JavaScript	@{{company}}/api-client	{{URL}}	{{Available/Planned}}
Python	{{company}}-python	{{URL}}	{{Available/Planned}}
Go	{{company}}-go	{{URL}}	{{Planned}}

Approval

Role	Name	Date	Signature
Author			
Backend Lead			
Tech Lead			
Product Owner			

Revision #10

Created 2026-02-18 08:44:20 UTC by John

Updated 2026-05-25 07:33:26 UTC by John