Todo List API — Especificação (v1)

Uma API HTTP/JSON para gerenciar listas e tarefas com suporte a filtros, ordenação, paginação, webhooks e operações em lote.

Visão geral

- **Base URL (prod):** `https://api.exemplo.com/v1`
- **Formato:** `application/json; charset=utf-8`
- **Autenticação:** Bearer Token (OAuth2 ou API Key)
- **Versionamento:** via caminho (`/v1`) e cabeçalho opcional `X-API-Version: 1`
- **Fuso horário:** todos os `*At` em **UTC** (ISO-8601, ex: `2025-09-04T13:00:00Z`)
- **Idempotência:** `Idempotency-Key` (UUID v4 recomendado) em requisições **POST/PATCH/DELETE** não-seguras
- **Paginação:** baseada em cursor (`page[limit]`, `page[after]`, `page[before]`)
- **Ordenação:** `sort=campo` ou `sort=-campo` (descendente)
- **Cache/Concorrência:** `ETag` + `If-None-Match` (GET) e `If-Match` (PATCH/DELETE)

Autenticação & Escopos

- **API Key:** `Authorization: Bearer <token>`
- **OAuth2 (Client Credentials / Authorization Code):**
    - **Scopes:**

- `tasks:read`, `tasks:write`
- `lists:read`, `lists:write`
- `comments:read`, `comments:write`
- `webhooks:manage`

Erros de auth: `401 Unauthorized` (token inválido/expirado), `403 Forbidden` (escopo insuficiente).

Modelos

Task (Tarefa)

```
{
  "id": "tsk_01HV7E2D1KXW3M2C3B4A5",
  "listId": "lst_01HV7E1Z9PABCDE",
  "title": "Revisar contrato",
  "description": "Checar cláusulas 3 e 5",
  "status": "open",              // open | in_progress | completed | archived
  "priority": "medium",          // low | medium | high | urgent
  "dueAt": "2025-09-10T23:59:59Z",
  "completedAt": null,
  "assignees": ["usr_01AB..."],
  "tags": ["jurídico", "Q3"],
  "checklist": [
    {"id":"chk_01","title":"Cláusula 3","checked":true},
    {"id":"chk_02","title":"Cláusula 5","checked":false}
  ],
  "createdAt": "2025-09-04T13:00:00Z",
  "updatedAt": "2025-09-04T13:00:00Z",
  "version": 3                    // para controle otimista
}
```

List (Lista)

```
{
  "id": "lst_01HV7E1Z9PABCDE",
  "name": "Backoffice",
  "description": "Tarefas do time financeiro",
  "createdAt": "2025-08-01T10:00:00Z",
  "updatedAt": "2025-09-01T12:30:00Z"
}
```

Comment (Comentário)

```
{
  "id": "cmt_01H...",
  "taskId": "tsk_01H...",
  "authorId": "usr_01H...",
  "body": "Faltou anexar o PDF.",
  "createdAt": "2025-09-04T13:05:00Z"
}
```

Error

```
{
  "error": {
    "code": "validation_error", // auth_error | not_found | rate_limit | conflict | ...
    "message": "Campo 'title' é obrigatório",
    "details": [{"field":"title","rule":"required"}],
    "requestId": "req_01XYZ..."
  }
}
```

Convenções de consulta

- **Paginação:**
  `GET ...?page[limit]=50&page[after]=<cursor>`
  Resposta inclui `next`/`prev` em `links` e `X-Next-Cursor`/`X-Prev-Cursor` nos headers.

- **Ordenação:**
  `sort=dueAt,-createdAt`

- **Filtros comuns (tasks):**
  - `filter[status]=open,in_progress`
  - `filter[listId]=lst_...`
  - `filter[assigneeId]=usr_...`
  - `filter[tag]=Q3`
  - `filter[dueAt][lte]=2025-09-30T23:59:59Z`
  - `filter[q]=contrato` (busca textual)

Endpoints

Health

- `GET /v1/health` → `200 OK` com `{ "status": "ok" }`

Lists

- `GET /v1/lists`
- `POST /v1/lists`
- `GET /v1/lists/{listId}`

- `PATCH /v1/lists/{listId}` *(requer `If-Match: <ETag>` ou `If-Match: "v<version>"`)*
- `DELETE /v1/lists/{listId}`

Exemplo – criar lista

- `POST /v1/lists`
- `Authorization: Bearer <token>`
- `Content-Type: application/json`
- `Idempotency-Key: 7b2a2dbe-9a1d-4b2a-9d3a-2b4f0a6e9f10`
-
- `{"name":"Backoffice","description":"Tarefas do time financeiro"}`

Tasks

- `GET /v1/tasks` (paginado, filtros, ordenação)
- `POST /v1/tasks`
- `GET /v1/tasks/{taskId}`
- `PATCH /v1/tasks/{taskId}` *(requer `If-Match`)*
- `DELETE /v1/tasks/{taskId}`
- **Ações:**
  - `POST /v1/tasks/{taskId}:complete`
  - `POST /v1/tasks/{taskId}:reopen`
  - `POST /v1/tasks/{taskId}:archive`
  - `POST /v1/tasks/{taskId}:restore`

Exemplo – listar tarefas abertas da lista, ordenadas por prazo

- GET /v1/tasks?filter[listId]=lst_01HV7E1Z9PABCDE&filter[status]=open&sort=dueAt&page[limit]=20
- Authorization: Bearer <token>

Exemplo – criar tarefa

- POST /v1/tasks
- Authorization: Bearer <token>
- Content-Type: application/json
- Idempotency-Key: 2f3a8b3b-6a4f-4b67-9ee1-6d1d8b9c1a2f
-
- {
- "listId": "lst_01HV7E1Z9PABCDE",
- "title": "Revisar contrato",
- "description": "Checar cláusulas 3 e 5",
- "priority": "medium",
- "dueAt": "2025-09-10T23:59:59Z",
- "assignees": ["usr_01AB..."],
- "tags": ["jurídico","Q3"],
- "checklist": [{"title":"Cláusula 3"},{"title":"Cláusula 5"}]
- }

Exemplo – atualizar tarefa com controle otimista

- PATCH /v1/tasks/tsk_01HV7E2D1KXW3M2C3B4A5
- Authorization: Bearer <token>
- Content-Type: application/json
- If-Match: "v3"
-
- {"status":"in_progress","priority":"high"}

Exemplo – completar

- POST /v1/tasks/tsk_01HV7E2D1KXW3M2C3B4A5:complete
- Authorization: Bearer <token>

- `Idempotency-Key: 9a7b1c5d-...`

Comments

- `GET /v1/tasks/{taskId}/comments`
- `POST /v1/tasks/{taskId}/comments`
- `DELETE /v1/tasks/{taskId}/comments/{commentId}`

Operações em lote

- `POST /v1/tasks:batch`

```
{
  "operations": [
    {"op":"create","data":{"listId":"lst_...","title":"A"}},
    {"op":"update","id":"tsk_...","data":{"priority":"urgent"}},
    {"op":"delete","id":"tsk_..."}
  ],
  "continueOnError": false
}
```

Resposta: retorna array posicional com `status`, `id`/`error` por operação.

Webhooks

- **Eventos:** `task.created`, `task.updated`, `task.completed`, `task.deleted`, `comment.created`, `comment.deleted`
- **Gerenciar:** `POST /v1/webhooks`, `GET /v1/webhooks`, `DELETE /v1/webhooks/{id}`

- **Segurança:** assinatura HMAC-SHA256 em header `X-Signature: t=<timestamp>, s=<hexdigest>` usando segredo do webhook
- **Retentativas:** exponencial backoff até 24h; considere `2xx` sucesso

Payload exemplo (`task.updated`):

```
{
  "id": "evt_01J...",
  "type": "task.updated",
  "createdAt": "2025-09-04T13:01:00Z",
  "data": { "task": { "id": "tsk_01...", "status": "completed", "version": 4 } }
}
```

Rate limiting

- **Padrão:** 120 requisições/minuto por token
- **Headers:**
  - `X-RateLimit-Limit: 120`
  - `X-RateLimit-Remaining: 87`
  - `X-RateLimit-Reset: 1693832400` *(epoch seconds)*
- **Excedeu:** `429 Too Many Requests` com `Retry-After`

Códigos de status (resumo)

- `200 OK`, `201 Created`, `202 Accepted` (webhook teste/lote), `204 No Content`

- `400 Bad Request` (validação)
- `401 Unauthorized`, `403 Forbidden`
- `404 Not Found`
- `409 Conflict` (falha de `If-Match`/concorrência)
- `412 Precondition Failed` (pré-condições ausentes)
- `415 Unsupported Media Type`
- `422 Unprocessable Entity`
- `429 Too Many Requests`
- `5xx` Erros do servidor

Cabeçalhos importantes

- **Idempotência:** `Idempotency-Key`
- **Versão:** `X-API-Version`
- **Cache/Concorrência:** `ETag`, `If-None-Match`, `If-Match`
- **Correlação:** `X-Request-Id` (retornado em erros e sucesso)

Regras de validação (exemplos)

- `title`: obrigatório, 1–240 chars
- `priority`: enum `low|medium|high|urgent`

- `status`: enum `open|in_progress|completed|archived`
- `dueAt`: ISO-8601 futuro ou `null`
- `checklist.title`: obrigatório se presente; `checked` default `false`
- `assignees[]`: IDs válidos de usuários existentes
-