DOCUMENTAÇÃO OFICIAL PARA DESENVOLVEDORES

API Gestão Cultural Pro

Módulo Financeiro · Projetos · Espaços de Trabalho · Webhooks

`https://gestaoculturalpro.com.br/api/v1`

Versão 1.0.0 · API 2026-11-01 · Atualizada em 06/11/2026
Desenvolvimento: Bruno Albertini | Gestão Cultural Pro

Sumário

1. Visão geral
2. Início rápido
3. Tipos de integração
4. Autenticação e chaves
5. Permissões (escopos)
6. Convenções
7. Erros
8. Módulo Financeiro
    1. Contas bancárias
    2. Lançamentos
    3. Transferências
    4. Contatos (clientes e fornecedores)
    5. Plano de contas e listas
    6. Planejamento financeiro
    7. Relatório consolidado
9. Projetos
10. Espaços de Trabalho
    1. Membros
    2. Quadros e listas
    3. Cartões
    4. Tarefas de produção
    5. Canais da Equipe e documentos
11. Webhooks e eventos
12. Receitas de integração
13. Equivalência com o mercado
14. Segurança
15. Referência rápida
16. Versões e suporte

1 Visão geral

A API dá a sistemas parceiros acesso seguro, granular e auditável aos dados da sua Instituição/Empresa no GCP. Ela foi desenhada a partir do dia a dia de produtoras, institutos e coletivos: cada lançamento pode ser rateado por projeto e rubrica, cada cartão pode pertencer a um projeto, e tudo o que entra pela API segue as mesmas regras do aplicativo.

R$

Financeiro completo

Contas e saldos, contas a pagar e a receber, baixas, transferências, contatos, plano de contas, planejamento e relatórios.

▦

Espaços de Trabalho

Quadros, listas, cartões com checklists e campos personalizados, tarefas, canais e documentos.

⚡

Tempo real

Webhooks com assinatura HMAC-SHA256 e feed de eventos para sincronização incremental confiável.

🔒

Segurança

Chaves com permissões por módulo, validade, renovação sem interrupção, limite de requisições e auditoria.

↺

Sem duplicidade

Idempotency-Key em criações e controle de versão (If-Match) em alterações concorrentes.

🇧🇷

Feita para o Brasil

Valores em BRL (reais e centavos), CPF/CNPJ validados e datas no calendário de Brasília.

2 Início rápido

1. **Gere a chave.** Um Administrador da Instituição/Empresa acessa *Gestão Global → Instituição/Empresa → API Gestão Cultural Pro → Gerar chave de API*, escolhe os módulos e o tipo de integração.
2. **Guarde em segredo.** A chave aparece uma única vez. Armazene em um cofre de segredos ou em variável de ambiente (ex.: `GCP_API_KEY`) do seu servidor.
3. **Faça a primeira chamada.** `GET /v1/ping` confirma a instituição, as permissões e a validade da chave.

Primeira chamada

CURL

```
curl https://gestaoculturalpro.com.br/api/v1/ping \
  -H "Authorization: Bearer $GCP_API_KEY"
```

NODE.JS

```
// Node.js 18+ (fetch nativo)
const res = await fetch("https://gestaoculturalpro.com.br/api/v1/ping", {
  headers: { Authorization: `Bearer ${process.env.GCP_API_KEY}` },
});
console.log(res.status, await res.json());
```

PYTHON

```
import os, requests

r = requests.get(
    "https://gestaoculturalpro.com.br/api/v1/ping",
    headers={"Authorization": f"Bearer {os.environ['GCP_API_KEY']}"},
    timeout=30,
)
print(r.status_code, r.json())
```

PHP

```
<?php
$ch = curl_init("https://gestaoculturalpro.com.br/api/v1/ping");
curl_setopt_array($ch, [
  CURLOPT_HTTPHEADER ⇒ ["Authorization: Bearer " . getenv("GCP_API_KEY")],
  CURLOPT_RETURNTRANSFER ⇒ true,
]);
$body = json_decode(curl_exec($ch), true);
echo curl_getinfo($ch, CURLINFO_HTTP_CODE);
```

C#

```
using var http = new HttpClient();
http.DefaultRequestHeaders.Authorization =
    new("Bearer", Environment.GetEnvironmentVariable("GCP_API_KEY"));
var res = await http.GetAsync("https://gestaoculturalpro.com.br/api/v1/ping");
Console.WriteLine(await res.Content.ReadAsStringAsync());
```

```
{
  "object": "ping",
  "ok": true,
  "api_version": "2026-11-01",
  "server_time": "2026-11-06T15:04:05.000Z",
  "timezone": "America/Sao_Paulo",
  "institution": {
    "id": "inst_8Hk",
    "name": "Instituto Exemplo"
  },
  "key": {
    "id": "3fa2c1d09b7e4a55",
    "name": "ERP Parceiro",
    "scopes": [
      "financial:read",
      "financial:write",
      "webhooks:manage"
    ],
    "expires_at": "2027-02-04T15:00:00.000Z"
  },
  "rate_limit_per_minute": 120,
  "request_id": "req_Q2m8XvLrT0aZk4Yp1nBc"
}
```

3 Escolha o tipo de integração

O administrador escolhe, em linguagem simples, como os sistemas vão conversar. Cada opção corresponde a um conjunto de permissões, que também pode ser personalizado item por item.

Somente consulta

GCP → parceiro (o parceiro busca)

Para quem

BI, painéis gerenciais, auditoria, conferência contábil

Como funciona

O parceiro lê os dados quando precisar, com filtros por período, conta, projeto e status. Para manter uma cópia atualizada sem reler tudo, use o feed de eventos (GET /v1/events).

Permissões

`financial:read` `workspace:read`

Guia

Ver receita passo a passo →

Só um lado: enviar ao GCP

Parceiro → GCP

Para quem

ERPs e sistemas de compras que geram contas a pagar e a receber

Como funciona

O parceiro cria contatos, lançamentos e transferências, registra baixas e anexa documentos. Use Idempotency-Key em toda criação para nunca duplicar registros.

Permissões

`financial:read` `financial:write`

Guia

Ver receita passo a passo →

Só um lado: o GCP avisa

GCP → parceiro (o GCP envia)

Para quem

ERPs que precisam saber o que foi pago ou alterado no GCP

Como funciona

O parceiro cadastra um webhook e recebe cada alteração em segundos, assinada com HMAC-SHA256. Em caso de indisponibilidade, as entregas são repetidas automaticamente.

Permissões

`financial:read` `webhooks:manage`

Guia

Ver receita passo a passo →

Os dois lados

Parceiro ⇄ GCP

Para quem

Integração financeira completa com ERP; gestão de tarefas sincronizada

Como funciona

Combina envio e avisos. Cada evento informa a origem (app ou api, com a chave), o que evita laços: ignore os eventos gerados pela sua própria chave.

Permissões

`financial:read` `financial:write`
`workspace:read` `workspace:write`
`webhooks:manage`

Guia

Ver receita passo a passo →

Acesso completo

Parceiro ⇄ GCP, com remoção

Para quem

Parceiros de total confiança, migrações e saneamento de dados

Como funciona

Inclui remoções, sempre com salvaguardas: lançamentos pagos ou conciliados, contatos com lançamentos e contas com movimentação não são apagados.

Permissões

`financial:read` `financial:write` `financial:delete` `workspace:read` `workspace:write` `workspace:delete` `webhooks:manage`

Guia

Ver receita passo a passo →

4 Autenticação e chaves

Envie a chave em todas as requisições, no cabeçalho `Authorization`. O cabeçalho `X-API-Key` também é aceito.

```
Authorization: Bearer gcp_live_3fa2c1d09b7e4a55_Xk7…(48 caracteres)
```

CARACTERÍSTICA	COMO FUNCIONA
Formato	`gcp_live_<id de 16 caracteres>_<segredo de 48 caracteres>`. O id identifica a chave nos registros de auditoria; o segredo nunca é armazenado (guardamos apenas o SHA-256).
Quem gera	Somente Administradores da Instituição/Empresa. Outros perfis não acessam a área de chaves.
Escopo de dados	Cada chave acessa exclusivamente a instituição em que foi gerada.
Validade	30, 90 (recomendado), 180 ou 365 dias, ou sem validade.
Renovação	Gera um novo segredo. O anterior pode continuar válido por 1 hora, 24 horas ou 7 dias, para a troca sem interrupção.
Desvinculação	Revoga a chave na hora e pausa os webhooks criados por ela.
Autoria das ações	Alterações são registradas em nome do administrador que gerou a chave, com a indicação “via API (nome do aplicativo)”. Se esse usuário for desativado, a chave para de funcionar (403 owner_inactive).

Uso exclusivo servidor a servidor. Nunca coloque a chave em aplicativos móveis, páginas web ou repositórios de código. A API não envia cabeçalhos CORS, o que impede o uso direto por navegadores.

5 Permissões (escopos)

ESCOPO	NOME NO PAINEL	PERMITE
`financial:read`	Financeiro · Consultar	Contas, saldos, extratos, lançamentos, contatos, planejamento, categorias, projetos, relatórios e eventos financeiros.
`financial:write`	Financeiro · Criar e editar	Lançamentos, baixas e reaberturas, transferências, contas, contatos e planejamento.
`financial:delete`	Financeiro · Remover	Lançamentos em aberto, contatos sem lançamentos, planejamento em rascunho e contas sem movimentação.
`workspace:read`	Espaços · Consultar	Quadros, listas, cartões, comentários, tarefas, canais, documentos, membros, projetos e eventos de Espaços.
`workspace:write`	Espaços · Criar e editar	Quadros, listas, cartões, comentários, tarefas e mensagens nos Canais da Equipe.
`workspace:delete`	Espaços · Remover	Arquivar e remover cartões e listas; remover tarefas.
`webhooks:manage`	Tempo real	Cadastrar, alterar, testar e remover webhooks; receber eventos dos módulos permitidos.

Criar, editar ou remover em um módulo inclui automaticamente a consulta desse módulo. Rotas sem a permissão necessária respondem `403 insufficient_scope`, com a lista em `error.required_scopes`.

6 Convenções

JSON e nomes

Requisições e respostas em JSON UTF-8, com campos em `snake_case`. Envie `Content-Type: application/json`. Parâmetros desconhecidos são recusados (400), para que erros de digitação não passem despercebidos.

Datas e fuso

Datas no formato `AAAA-MM-DD`, sempre no calendário de Brasília (America/Sao_Paulo). Data e hora em ISO 8601 (UTC). Em cartões, uma data sem hora é gravada às 12:00 de Brasília.

Valores

Moeda BRL. Envie `amount` (reais, até 2 casas) **ou** `amount_cents` (inteiro). As respostas trazem os dois. Prefira centavos para evitar arredondamentos.

Enumerações

Valores normalizados em inglês (`income`, `paid`...), compatíveis com qualquer ERP. Os campos `raw_*` trazem o termo original em português (Receita, Pago...).

Paginação por cursor

Listas retornam `{ object: "list", data, has_more, next_cursor }`. Repita a chamada com `cursor=next_cursor` até `has_more = false`. `limit`: 1 a 200 (padrão 50). Em buscas com filtros calculados, uma página pode vir menor e ainda assim ter `has_more = true`.

Concorrência (If-Match)

Leituras individuais trazem `etag`. Envie `If-Match: <etag>` no PATCH/DELETE: se a equipe alterou o registro no aplicativo depois da sua leitura, a API responde 412 em vez de sobrescrever.

Idempotência

Envie `Idempotency-Key` (8 a 255 caracteres, ex.: UUID ou id do seu pedido) em todo POST. Repetições em até 24 h devolvem a mesma resposta, com `Idempotent-Replayed: true`, sem criar duplicatas.

Limites

120 requisições por minuto por chave. Cabeçalhos `RateLimit-Limit`, `RateLimit-Remaining` e `RateLimit-Reset`; ao exceder, 429 com `Retry-After`. Corpo até 1 MB.

Percorrendo todas as páginas

NODE.JS

```
async function* todosOsLancamentos(filtros) {
  let cursor;
  do {
    const qs = new URLSearchParams({ ...filtros, limit: "200", ...(cursor && { cursor }) });
    const res = await fetch(`https://gestaoculturalpro.com.br/api/v1/financial/transactions?${qs}`, {
headers });
    const page = await res.json();
    yield* page.data;
    cursor = page.has_more ? page.next_cursor : null;
  } while (cursor);
}
```

```
for await (const t of todosOsLancamentos({ date_from: "2026-11-01", date_to: "2026-11-30" })) {
  console.log(t.date, t.description, t.amount);
}
```

PYTHON

```
def todos_os_lancamentos(**filtros):
    cursor = None
    while True:
        params = {**filtros, "limit": 200, **({"cursor": cursor} if cursor else {})}
        page = requests.get(f"https://gestaoculturalpro.com.br/api/v1/financial/transactions",
                            headers=HEADERS, params=params, timeout=30).json()
        yield from page["data"]
        if not page["has_more"]:
            break
        cursor = page["next_cursor"]
```

Cabeçalhos de toda resposta: `X-Request-Id` (informe ao suporte), `GCP-Api-Version` (2026-11-01), `Cache-Control: no-store`.

Versionamento: o prefixo `/v1` é estável. Novos campos e novas rotas podem ser adicionados sem aviso; seu código deve ignorar campos que não conhece. Mudanças incompatíveis só ocorrem em uma nova versão (`/v2`), anunciada com antecedência.

7 Erros

Erros seguem sempre o mesmo formato, com mensagem em português pronta para exibir ao usuário final.

```
{
  "error": {
    "type": "conflict",
    "code": "contact_exists",
    "message": "Já existe um contato com este CPF/CNPJ nesta instituição.",
    "existing_id": "ct_17",
    "request_id": "req_Hn3…",
    "doc_url": "https://gestaoculturalpro.com.br/developers/#erros"
  }
}
```

HTTP	TYPE	QUANDO OCORRE
400	`invalid_request`	Parâmetro ausente, inválido ou desconhecido (veja error.param)
401	`authentication`	Chave ausente, inválida, revogada ou expirada
403	`permission`	Escopo insuficiente (error.required_scopes) ou usuário que gerou a chave desativado
404	`not_found`	Registro inexistente ou de outra instituição
405	`invalid_request`	Método não permitido (cabeçalho Allow)
409	`conflict`	Regra de negócio: registro conciliado, duplicado (existing_id), com vínculos…
412	`precondition`	If-Match não confere: o registro mudou depois da sua leitura
413 · 415	`invalid_request`	Corpo acima de 1 MB · Content-Type diferente de JSON
422	`invalid_request` / `idempotency`	Vínculo inexistente (conta, contato, projeto, membro, etiqueta) ou Idempotency-Key reutilizada
429	`rate_limit`	Limite por minuto atingido (aguarde Retry-After segundos)
500	`server`	Falha inesperada — repita com backoff e informe o request_id

CÓDIGOS (ERROR.CODE)

`parameter_missing` `unknown_parameter` `invalid_type` `invalid_enum` `invalid_date` `invalid_number` `invalid_id` `invalid_cursor` `invalid_document` `allocations_mismatch` `related_not_found` `insufficient_scope` `invalid_api_key` `revoked_api_key` `expired_api_key` `owner_inactive` `transaction_reconciled` `transaction_settled` `transaction_linked` `transfer_immutable` `already_settled` `future_settlement` `contact_exists` `contact_in_use` `account_in_use` `planning_processed`

`card_not_archived` `list_not_empty` `unknown_label` `unknown_member` `unknown_custom_field` `version_conflict` `idempotency_key_reused` `idempotency_in_progress` `rate_limit_exceeded` `webhook_limit` `private_url` `insecure_url`

Quando repetir? Repita apenas 429 (após `Retry-After`) e 5xx, com espera exponencial (1 s, 2 s, 4 s...). Erros 4xx indicam algo a corrigir na requisição. Com `Idempotency-Key`, repetir um POST é sempre seguro.

8 Módulo Financeiro

Tudo o que a equipe financeira faz no GCP: contas bancárias, contas a pagar e a receber, transferências, baixas, contatos, plano de contas, planejamento e relatórios. Os registros são gravados exatamente como o aplicativo os grava, preservando saldos, conciliação, rateios por projeto e o histórico de alterações.

Criando uma despesa (com Idempotency-Key)

CURL

```
curl -X POST https://gestaoculturalpro.com.br/api/v1/financial/transactions \
  -H "Authorization: Bearer $GCP_API_KEY" \
  -H "Content-Type: application/json" \
  -H "Idempotency-Key: 6f1c2a9e-erp-pedido-48213" \
  -d '{
    "type": "expense",
    "amount": 3500.00,
    "date": "2026-11-20",
    "account_id": "acc_9f2",
    "contact_id": "ct_17",
    "description": "Sonorização — Festival de Música",
    "allocations": [{ "project_id": "prj_fest", "rubric_id": "rub_som", "amount": 3500.00 }]
  }'
```

NODE.JS

```
const res = await fetch("https://gestaoculturalpro.com.br/api/v1/financial/transactions", {
  method: "POST",
  headers: {
    Authorization: `Bearer ${process.env.GCP_API_KEY}`,
    "Content-Type": "application/json",
    "Idempotency-Key": `erp-pedido-${pedido.id}`, // mesma chave = sem duplicidade
  },
  body: JSON.stringify({
    type: "expense",
    amount_cents: 350000,
    date: "2026-11-20",
    account_id: "acc_9f2",
    description: "Sonorização — Festival de Música",
  }),
});
if (!res.ok) throw new Error((await res.json()).error.message);
```

PYTHON

```
r = requests.post(
    "https://gestaoculturalpro.com.br/api/v1/financial/transactions",
    headers={
        "Authorization": f"Bearer {API_KEY}",
        "Idempotency-Key": f"erp-pedido-{pedido_id}",
    },
    json={
        "type": "expense",
        "amount": 3500.00,
        "date": "2026-11-20",
```

```
        "account_id": "acc_9f2",
        "description": "Sonorização — Festival de Música",
    },
    timeout=30,
)
r.raise_for_status()
transacao = r.json()
```

```php
PHP
$payload = json_encode([
  "type" => "expense",
  "amount" => 3500.00,
  "date" => "2026-11-20",
  "account_id" => "acc_9f2",
  "description" => "Sonorização — Festival de Música",
]);
$ch = curl_init("https://gestaoculturalpro.com.br/api/v1/financial/transactions");
curl_setopt_array($ch, [
  CURLOPT_POST => true,
  CURLOPT_POSTFIELDS => $payload,
  CURLOPT_RETURNTRANSFER => true,
  CURLOPT_HTTPHEADER => [
    "Authorization: Bearer " . getenv("GCP_API_KEY"),
    "Content-Type: application/json",
    "Idempotency-Key: erp-pedido-" . $pedidoId,
  ],
]);
$transacao = json_decode(curl_exec($ch), true);
```

Contas bancárias `account`

Contas correntes, poupanças, investimentos e caixas. O saldo é calculado em tempo real a partir dos lançamentos, com as mesmas regras do aplicativo.

Campos principais do objeto

`id`	`string`	Identificador da conta
`name`	`string`	Nome da conta
`bank_name / bank_code`	`string`	Banco e código (ex.: 001)
`agency / account_number`	`string`	Agência e número
`type`	`enum`	checking · savings · investment · cash
`initial_balance`	`number`	Saldo inicial em reais
`active`	`boolean`	Conta ativa
`project_account / project_id`	`boolean / string`	Conta exclusiva de projeto

`balance.reconciled`	`number`	Saldo conciliado com o extrato bancário (até hoje)
`balance.settled`	`number`	Saldo de tudo que foi pago ou recebido até hoje
`balance.projected`	`number`	Saldo considerando também os lançamentos em aberto
`balance.pending_income` / `pending_expense`	`number`	Valores a receber e a pagar

GET `/v1/financial/accounts` `financial:read`

Listar contas

PARÂMETROS DE CONSULTA

PARÂMETRO	TIPO	DESCRIÇÃO
`active`	boolean	true ou false
`include`	string	balance: inclui os saldos de cada conta
`limit` / `cursor`		Paginação

GET `/v1/financial/accounts/{id}` `financial:read`

Consultar conta com saldos

POST `/v1/financial/accounts` `financial:write`

Criar conta

CORPO (JSON)

PARÂMETRO	TIPO	DESCRIÇÃO
`name` OBRIGATÓRIO	string	Até 120 caracteres
`type`	enum	checking (padrão) · savings · investment · cash
`bank_name, bank_code, agency, account_number, pix_key`	string	Dados bancários
`initial_balance`	number	Pode ser negativo
`project_id`	string	Vincula a conta a um projeto
`color`	string	#RRGGBB
`active`	boolean	Padrão: true

PATCH `/v1/financial/accounts/{id}` `financial:write`

Alterar conta

> Mesmos campos da criação, exceto type e project_id.

DELETE `/v1/financial/accounts/{id}` `financial:delete`

Remover conta sem movimentação

> Contas com lançamentos retornam 409 account_in_use; desative com {"active": false}.

GET `/v1/financial/accounts/{id}/statement` `financial:read`

Extrato com saldo acumulado

PARÂMETROS DE CONSULTA

PARÂMETRO	TIPO	DESCRIÇÃO
`date_from / date_to`	AAAA-MM-DD	Padrão: mês atual até hoje
`status`	enum	settled (padrão) · reconciled · all

EXEMPLO · EXTRATO DO MÊS

```
GET /v1/financial/accounts/acc_9f2/statement?date_from=2026-11-01&date_to=2026-11-30
```


```
{
  "object": "account_statement",
  "account_id": "acc_9f2",
  "date_from": "2026-11-01",
  "date_to": "2026-11-30",
  "status": "settled",
  "opening_balance": 12450,
  "closing_balance": 15980.5,
  "currency": "BRL",
  "entries": [
    {
      "transaction_id": "tx_81a",
      "date": "2026-11-04",
      "description": "Patrocínio — cota ouro",
      "type": "income",
      "amount": 5000,
      "amount_cents": 500000,
      "balance_after": 17450,
      "status": "received",
      "reconciled": true,
      "contact_id": "ct_17",
      "document_number": null,
      "bank_transaction_id": "20261104001"
    }
  ]
}
```


Lançamentos `transaction`

Receitas e despesas (contas a receber e a pagar) com rateio por projeto e rubrica orçamentária, documentos fiscais, anexos, baixa e conciliação. Transferências entre contas são criadas em /transfers.

Campos principais do objeto

`type`	`enum`	income · expense · transfer
`status`	`enum`	pending · scheduled · overdue · paid · received · reconciled
`settled`	`boolean`	true quando pago, recebido ou conciliado
`amount / amount_cents`	`number / integer`	Valor em reais e em centavos (BRL)
`date`	`AAAA-MM-DD`	Data do pagamento/recebimento (previsto ou efetivo)
`due_date / competence_date / issue_date`	`AAAA-MM-DD`	Vencimento, competência e emissão do documento

`account_id / contact_id`	`string`	Conta bancária e cliente/fornecedor
`allocations[]`	`array`	Rateio: project_id, rubric_id, cost_center, amount, cost_type, tags
`document_type / document_number`	`string`	Ex.: Nota Fiscal · NF 1234
`bank_transaction_id`	`string`	Identificador do banco (FITID do OFX) para conciliação
`original_amount / settlements[]`	`number / array`	Valor original e histórico de baixas com diferença
`raw_type / raw_status`	`string`	Valores originais em português (Receita, Pago...)
`etag`	`string`	Versão do registro para If-Match

GET `/v1/financial/transactions` `financial:read`

Listar lançamentos

PARÂMETROS DE CONSULTA

PARÂMETRO	TIPO	DESCRIÇÃO
`type`	enum	income · expense · transfer
`status`	enum	pending · scheduled · overdue · paid · received · reconciled
`settled`	boolean	Somente baixados (true) ou em aberto (false)
`account_id · contact_id · project_id`	string	Filtros por vínculo
`date_from / date_to`	AAAA-MM-DD	Período (ordena por data e inclui total_count)
`date_field`	enum	date (padrão) · due_date · competence_date
`limit / cursor`		Paginação

GET `/v1/financial/transactions/{id}` `financial:read`

Consultar lançamento (inclui etag)

POST `/v1/financial/transactions` `financial:write`

Criar receita ou despesa

CORPO (JSON)

PARÂMETRO	TIPO	DESCRIÇÃO
`type` OBRIGATÓRIO	enum	income · expense
`amount ou amount_cents` OBRIGATÓRIO	number	Valor maior que zero, até 2 casas
`date` OBRIGATÓRIO	AAAA-MM-DD	Data prevista ou efetiva
`account_id` OBRIGATÓRIO	string	Conta bancária
`description` OBRIGATÓRIO	string	Até 500 caracteres
`status`	enum	Padrão: pending. paid/received com data futura vira scheduled
`contact_id`	string	Cliente ou fornecedor
`due_date · competence_date · issue_date`	AAAA-MM-DD	Padrão de vencimento e competência: date
`category · cost_center · payment_method`	string	Classificação
`document_type · document_number · bank_document_number · bank_transaction_id`	string	Documento e banco
`allocations`	array	Soma igual ao valor (até 50 itens)
`attachments`	array	Até 20 { name, url (https), content_type, document_type }
`tags · notes`	array · string	Etiquetas e observações

PATCH `/v1/financial/transactions/{id}` `financial:write`

Alterar lançamento

> Lançamentos conciliados aceitam apenas campos descritivos. Envie If-Match para evitar sobrescrever alterações feitas no aplicativo.

POST /v1/financial/transactions/{id}/settle

financial:write

Registrar baixa (pagamento ou recebimento)

CORPO (JSON)

PARÂMETRO	TIPO	DESCRIÇÃO
`date`	AAAA-MM-DD	Padrão: hoje (não pode ser futura)
`amount`	number	Valor efetivo; a diferença fica registrada em settlements
`reconciled`	boolean	true marca como conciliado com o extrato
`bank_transaction_id`	string	Identificador do banco

POST /v1/financial/transactions/{id}/reopen

financial:write

Desfazer baixa

> Lançamentos conciliados devem ter a conciliação desfeita no aplicativo.

DELETE /v1/financial/transactions/{id}

financial:delete

Excluir lançamento em aberto

> Baixados e vinculados a Vendas, Planejamento ou Logística retornam 409. Em transferências, as duas pernas são excluídas.

EXEMPLO · DESPESA COM RATEIO POR PROJETO E RUBRICA

```
{
  "type": "expense",
  "amount": 3500,
  "date": "2026-11-20",
  "due_date": "2026-11-20",
  "account_id": "acc_9f2",
  "contact_id": "ct_17",
  "description": "Sonorização — Festival de Música",
  "document_type": "Nota Fiscal",
  "document_number": "NF 4821",
  "allocations": [
    {
      "project_id": "prj_fest",
      "rubric_id": "rub_som",
      "amount": 3500
    }
  ],
  "tags": [
    "festival"
  ]
}
```

```
{
  "id": "tx_7Qm2",
  "object": "transaction",
  "type": "expense",
  "raw_type": "Despesa",
  "status": "pending",
  "raw_status": "Pendente",
  "settled": false,
  "reconciled": false,
  "description": "Sonorização — Festival de Música",
  "amount": 3500,
  "amount_cents": 350000,
  "currency": "BRL",
  "date": "2026-11-20",
  "due_date": "2026-11-20",
  "competence_date": "2026-11-20",
  "account_id": "acc_9f2",
  "contact_id": "ct_17",
  "project_ids": [
    "prj_fest"
  ],
  "allocations": [
    {
      "project_id": "prj_fest",
      "rubric_id": "rub_som",
      "amount": 3500,
      "amount_cents": 350000
    }
  ],
  "etag": "1793548800123.0"
}
```


Transferências `transfer`

Movimentação entre duas contas da instituição. Gera dois lançamentos pareados (saída e entrada), exatamente como o aplicativo, e os saldos das duas contas ficam coerentes.

POST `/v1/financial/transfers` `financial:write`

Criar transferência

CORPO (JSON)

PARÂMETRO	TIPO	DESCRIÇÃO
`from_account_id` OBRIGATÓRIO	string	Conta de origem
`to_account_id` OBRIGATÓRIO	string	Conta de destino (diferente da origem)
`amount` ou `amount_cents` OBRIGATÓRIO	number	Valor
`date` OBRIGATÓRIO	AAAA-MM-DD	Data
`status`	enum	pending (padrão) · scheduled · paid · reconciled
`description` · `notes` · `document_number` · `tags`		Opcionais

Contatos (clientes e fornecedores) `contact`

Pessoas físicas e jurídicas usadas nos lançamentos. CPF e CNPJ são validados pelos dígitos verificadores e não se repetem na instituição.

Campos principais do objeto

`kind`	`enum`	individual (PF) · company (PJ)
`document`	`string`	CPF/CNPJ somente dígitos (document_formatted traz a máscara)
`role`	`enum`	customer · supplier · both
`bank`	`object`	bank_name, agency, account, account_type, pix_key, pix_key_type

GET `/v1/financial/contacts` `financial:read`

Listar ou buscar

PARÂMETROS DE CONSULTA

PARÂMETRO	TIPO	DESCRIÇÃO
`document`	string	Busca exata por CPF/CNPJ (com ou sem máscara)
`search`	string	Busca por nome, sem diferenciar acentos
`role`	enum	customer · supplier · both

GET `/v1/financial/contacts/{id}` `financial:read`

Consultar contato

POST `/v1/financial/contacts` `financial:write`

Criar contato

CORPO (JSON)

PARÂMETRO	TIPO	DESCRIÇÃO
`name` OBRIGATÓRIO	string	Nome ou razão social
`document`	string	CPF/CNPJ válido; duplicado retorna 409 com existing_id
`kind · role`	enum	Deduzidos do documento quando omitidos
`email · phone · address · tags · bank`		Opcionais

PATCH `/v1/financial/contacts/{id}` `financial:write`

Alterar contato

DELETE `/v1/financial/contacts/{id}` `financial:delete`

Remover contato sem lançamentos

Plano de contas e listas `category`

Categorias de receita e despesa (plano de contas da instituição e de cada conta bancária), além dos centros de custo, formas de pagamento e tipos de documento cadastrados.

GET `/v1/financial/categories` — `financial:read`

Listar categorias, centros de custo e listas

Planejamento financeiro `planning_item`

Pré-planejamento de pagamentos e recebimentos por mês. Itens em rascunho podem ser alterados; ao serem enviados para Contas a Pagar/Receber no aplicativo, passam a processed e geram um lançamento.

GET `/v1/financial/planning` — `financial:read`

Listar itens

PARÂMETROS DE CONSULTA

PARÂMETRO	TIPO	DESCRIÇÃO
`month`	AAAA-MM	Mês
`status`	enum	draft · processed

GET `/v1/financial/planning/{id}` — `financial:read`

Consultar item

POST `/v1/financial/planning` — `financial:write`

Criar item

CORPO (JSON)

PARÂMETRO	TIPO	DESCRIÇÃO
`beneficiary_name` OBRIGATÓRIO	string	Favorecido
`description` OBRIGATÓRIO	string	Descritivo
`amount` OBRIGATÓRIO	number	Valor
`planned_date` OBRIGATÓRIO	AAAA-MM-DD	Data prevista (define o mês)
`type`	enum	expense (padrão) · income
`beneficiary_email` · `due_date` · `project_id` · `account_id` · `rubric_id` · `payment_method`		Opcionais

PATCH `/v1/financial/planning/{id}` `financial:write`

Alterar item em rascunho

> Aceita também on_hold (true/false) e hold_comment.

DELETE `/v1/financial/planning/{id}` `financial:delete`

Remover item em rascunho

Relatório consolidado `financial_summary`

Receitas, despesas e resultado do período, agrupados como você precisar. Aplicações e resgates de investimento não entram como resultado (somente rendimentos), como no aplicativo.

GET `/v1/financial/summary` `financial:read`

Resumo por período

PARÂMETROS DE CONSULTA

PARÂMETRO	TIPO	DESCRIÇÃO
`date_from / date_to` OBRIGATÓRIO	AAAA-MM-DD	Período
`group_by`	enum	month (padrão) · category · project · account · cost_center
`basis`	enum	cash (data do pagamento, padrão) · competence
`status`	enum	settled (padrão) · all
`account_id · project_id`	string	Filtros

9 Projetos

Projetos culturais são o eixo do GCP: lançamentos são rateados por projeto e rubrica, e quadros e tarefas podem pertencer a um projeto. Disponível com financial:read ou workspace:read (somente leitura).

Projetos e execução orçamentária `project`

Dados do projeto, orçamento aprovado por rubrica e execução financeira (realizado, pendente e disponível por rubrica), pronto para prestação de contas.

GET `/v1/projects` — `financial:read ou workspace:read`

Listar projetos

PARÂMETROS DE CONSULTA

PARÂMETRO	TIPO	DESCRIÇÃO
`status`	enum	planned · in_progress · finished

GET `/v1/projects/{id}` — `financial:read ou workspace:read`

Consultar projeto com orçamento (budget[])

GET `/v1/projects/{id}/financial-summary` — `financial:read`

Execução orçamentária por rubrica

EXEMPLO · EXECUÇÃO POR RUBRICA

```
GET /v1/projects/prj_fest/financial-summary
```

```
{
  "object": "project_financial_summary",
  "project_id": "prj_fest",
  "currency": "BRL",
  "budgeted": 13000,
  "expenses": {
    "realized": 8500,
    "pending": 1500
  },
  "income": {
    "received": 80000,
    "pending": 0
  },
  "execution_percentage": 65.38,
  "rubrics": [
    {
      "rubric_id": "rub_som",
      "code": "1.2",
      "name": "Sonorização",
      "budgeted": 3000,
      "realized": 3000,
      "pending": 0,
      "available": 0
    }
  ]
}
```

10 Espaços de Trabalho

Quadros de Trabalho no estilo kanban, tarefas de produção, Canais da Equipe e Documentos Colaborativos. Alterações feitas pela API aparecem para a equipe em tempo real e ficam no histórico de atividades do cartão com o nome do aplicativo. Quadros vinculados ao Asana continuam sincronizando normalmente.

Membros `member`

Pessoas ativas da instituição (inclusive as que acessam várias instituições). Use os IDs em members, responsible_ids e assignee_id.

GET `/v1/workspace/members` — `workspace:read`

Listar membros ativos

Quadros e listas `board / list`

Um quadro reúne listas (fases) e cartões. Ao criar um quadro, as listas A Fazer, Em Andamento e Concluído são criadas automaticamente (default_lists: false desativa).

GET `/v1/workspace/boards` — `workspace:read`

Listar quadros

PARÂMETROS DE CONSULTA

PARÂMETRO	TIPO	DESCRIÇÃO
`project_id`	string	Quadros de um projeto

GET `/v1/workspace/boards/{id}` — `workspace:read`

Consultar quadro com listas, etiquetas e campos personalizados

PARÂMETROS DE CONSULTA

PARÂMETRO	TIPO	DESCRIÇÃO
`include_archived`	boolean	Inclui listas arquivadas

POST `/v1/workspace/boards` `workspace:write`

Criar quadro

CORPO (JSON)

PARÂMETRO	TIPO	DESCRIÇÃO
`title` OBRIGATÓRIO	string	Título
`description · project_id`	string	Opcionais
`visibility`	enum	workspace (padrão) · private · organization
`members`	array	IDs de membros
`labels`	array	{ name, color }
`default_lists`	boolean	Padrão: true

PATCH `/v1/workspace/boards/{id}` `workspace:write`

Alterar título, descrição ou etiquetas

GET `/v1/workspace/boards/{id}/lists` `workspace:read`

Listar listas (fases)

POST `/v1/workspace/boards/{id}/lists` `workspace:write`

Criar lista

CORPO (JSON)

PARÂMETRO	TIPO	DESCRIÇÃO
`title` OBRIGATÓRIO	string	Título
`color · position`		Opcionais (padrão: ao final)

PATCH `/v1/workspace/lists/{id}` `workspace:write`

Renomear, reordenar ou arquivar lista

DELETE `/v1/workspace/lists/{id}` `workspace:delete`

Arquivar lista (?permanent=true remove lista vazia)

Cartões `card`

Tarefas do quadro, com responsáveis, etiquetas, datas, checklists, campos personalizados, dependências (blocked_by) e comentários. Datas sem hora são gravadas às 12:00 de Brasília.

Campos principais do objeto

`list_id`	`string`	Lista (fase) atual
`description / description_html`	`string`	Texto simples e HTML (o HTML recebido é higienizado)
`completed / archived`	`boolean`	Concluído e arquivado
`start_date / due_date`	`ISO 8601`	Datas (AAAA-MM-DD também é aceito na entrada)
`members / labels`	`array`	IDs de membros e etiquetas do quadro (pelo nome ou id)
`checklists[]`	`array`	{ title, items: [{ text, completed, assignee_id, due_date }] }
`custom_fields`	`object`	{ id_do_campo: valor } conforme o tipo definido no quadro
`blocked_by`	`array`	Cartões que bloqueiam este
`url`	`string`	Link direto para o cartão no GCP

GET `/v1/workspace/cards` `workspace:read`

Listar cartões de um quadro

PARÂMETROS DE CONSULTA

PARÂMETRO	TIPO	DESCRIÇÃO
`board_id` OBRIGATÓRIO	string	Quadro
`list_id`	string	Lista
`completed · archived`	boolean	Padrão: não arquivados
`member_id`	string	Responsável
`due_before / due_after`	AAAA-MM-DD	Vencimento

GET `/v1/workspace/cards/{id}` `workspace:read`

Consultar cartão (inclui etag)

POST `/v1/workspace/cards` `workspace:write`

Criar cartão

CORPO (JSON)

PARÂMETRO	TIPO	DESCRIÇÃO
`board_id` OBRIGATÓRIO	string	Quadro
`title` OBRIGATÓRIO	string	Título
`list_id`	string	Padrão: primeira lista ativa
`description ou description_html`	string	Texto ou HTML
`start_date · due_date`	data	AAAA-MM-DD ou ISO 8601
`members · labels · checklists · custom_fields · blocked_by`		Ver campos acima
`position · completed · estimated_minutes · location`		Opcionais

PATCH `/v1/workspace/cards/{id}` `workspace:write`

Alterar, mover (list_id), concluir ou arquivar

> Envie null em start_date/due_date para limpar a data. custom_fields é mesclado; null remove um campo.

DELETE `/v1/workspace/cards/{id}` `workspace:delete`

Arquivar cartão (?permanent=true remove definitivamente um cartão já arquivado)

GET `/v1/workspace/cards/{id}/comments` `workspace:read`

Listar comentários

POST `/v1/workspace/cards/{id}/comments` `workspace:write`

Comentar

CORPO (JSON)

PARÂMETRO	TIPO	DESCRIÇÃO
`text` OBRIGATÓRIO	string	Até 5.000 caracteres

EXEMPLO · CRIAR CARTÃO COM CHECKLIST E CAMPO PERSONALIZADO

```
{
  "board_id": "brd_31",
  "list_id": "lst_todo",
  "title": "Contratar sonorização",
  "description": "Cotar com três fornecedores.",
  "due_date": "2026-11-25",
  "members": [
    "usr_ana"
  ],
  "labels": [
    "Urgente"
  ],
  "checklists": [
    {
      "title": "Etapas",
      "items": [
        {
          "text": "Cotar"
        },
        {
          "text": "Fechar contrato"
        }
      ]
    }
  ],
  "custom_fields": {
    "cf_valor": 3500
  }
}
```

```
{
  "id": "crd_9Ka",
  "object": "card",
  "board_id": "brd_31",
  "list_id": "lst_todo",
  "title": "Contratar sonorização",
  "completed": false,
  "archived": false,
  "due_date": "2026-11-25T15:00:00.000Z",
  "members": [
    "usr_ana"
  ],
  "labels": [
    {
      "id": "lbl_1",
      "name": "Urgente",
      "color": "#ef4444"
    }
  ],
  "url": "https://gestaoculturalpro.com.br/app/quadro/brd_31?card=crd_9Ka"
}
```

Tarefas de produção `task`

Eventos, etapas de projeto e ações de comunicação da Produção, com responsáveis, prioridade, local e checklist.

Campos principais do objeto

`kind`	`enum`	event · project · communication
`priority`	`enum`	low · medium · high
`status`	`enum`	suggested · in_progress · completed · cancelled
`start_date / end_date`	`string`	AAAA-MM-DD ou AAAA-MM-DDTHH:mm (horário de Brasília)

GET `/v1/workspace/tasks` — `workspace:read`

Listar tarefas

PARÂMETROS DE CONSULTA

PARÂMETRO	TIPO	DESCRIÇÃO
`status · kind · project_id`		Filtros
`date_from / date_to`	AAAA-MM-DD	Tarefas que ocorrem no período

GET `/v1/workspace/tasks/{id}` — `workspace:read`

Consultar tarefa

POST `/v1/workspace/tasks` — `workspace:write`

Criar tarefa

CORPO (JSON)

PARÂMETRO	TIPO	DESCRIÇÃO
`name` OBRIGATÓRIO	string	Nome
`start_date` OBRIGATÓRIO	string	Início
`end_date`	string	Padrão: início
`kind · priority · status · description · location · address · responsible_ids · project_id · color · checklist`		Opcionais

PATCH `/v1/workspace/tasks/{id}` `workspace:write`

Alterar tarefa

DELETE `/v1/workspace/tasks/{id}` `workspace:delete`

Remover tarefa (tarefas com Logística: use status cancelled)

Canais da Equipe e documentos `channel / message / doc`

Publique avisos automáticos nos Canais da Equipe (ex.: "pagamento confirmado no ERP"). Canais privados só ficam visíveis se o administrador que gerou a chave for membro. Documentos colaborativos: lista com metadados e link.

GET `/v1/workspace/channels` `workspace:read`

Listar canais

GET `/v1/workspace/channels/{id}/messages` `workspace:read`

Mensagens recentes (mais novas primeiro)

POST `/v1/workspace/channels/{id}/messages` `workspace:write`

Publicar mensagem

CORPO (JSON)

PARÂMETRO	TIPO	DESCRIÇÃO
`text` OBRIGATÓRIO	string	Até 4.000 caracteres

GET `/v1/workspace/docs` `workspace:read`

Listar documentos colaborativos

11 Webhooks e eventos

Toda alteração nos módulos integrados (feita no aplicativo ou pela API) vira um evento. Você pode recebê-los por webhook, em tempo real, ou buscá-los pelo feed `GET /v1/events`. Os dois caminhos trazem o mesmo conteúdo.

Cadastrando um webhook

```
{
  "url": "https://erp.parceiro.com.br/webhooks/gcp",
  "events": [
    "transaction.created",
    "transaction.settled",
    "transaction.updated"
  ],
  "description": "Sincronização financeira"
}
```

A resposta traz o `secret` (`whsec_…`), exibido somente nesta vez. Use `["*"]` para receber todos os eventos permitidos pelas permissões da chave. Requisitos: HTTPS, até 10 webhooks por instituição, sem endereços internos ou privados e sem redirecionamentos.

Catálogo de eventos

EVENTO	QUANDO
`transaction.created`	Lançamento criado
`transaction.updated`	Lançamento alterado
`transaction.settled`	Pago, recebido ou conciliado
`transaction.reopened`	Baixa desfeita
`transaction.deleted`	Lançamento excluído
`account.created · account.updated · account.deleted`	Contas bancárias
`contact.created · contact.updated · contact.deleted`	Clientes e fornecedores
`planning.created · planning.updated · planning.deleted`	Planejamento
`board.created · board.updated · board.deleted`	Quadros
`list.created · list.updated · list.deleted`	Listas
`card.created · card.updated · card.deleted`	Cartões

EVENTO	QUANDO
`card.moved`	Cartão mudou de lista
`card.completed`	Cartão concluído
`card.archived`	Cartão arquivado
`task.created · task.updated · task.deleted`	Tarefas

Uma baixa gera dois eventos: `transaction.settled` e `transaction.updated`. Eventos financeiros exigem `financial:read`; de Espaços, `workspace:read`.

Formato do evento

```
{
  "id": "evt_17935488001230_9c1e4b7a2d3f8e61",
  "object": "event",
  "type": "transaction.settled",
  "api_version": "2026-11-01",
  "created_at": "2026-11-20T17:32:08.123Z",
  "origin": {
    "type": "app"
  },
  "data": {
    "object": {
      "id": "tx_7Qm2",
      "object": "transaction",
      "type": "expense",
      "status": "paid",
      "amount": 3500,
      "date": "2026-11-20",
      "account_id": "acc_9f2"
    },
    "changed_fields": [
      "status",
      "date"
    ]
  }
}
```

CABEÇALHO	CONTEÚDO
`GCP-Signature`	`t=<timestamp>,v1=<HMAC-SHA256 hex de "t.corpo" com o secret>`
`GCP-Event-Id` · `GCP-Event-Type`	Identificador e tipo do evento (use o id para descartar repetições)
`GCP-Delivery-Id`	Identificador da tentativa de entrega

Origem e prevenção de laços

`origin.type` é `app` quando a alteração veio da equipe no GCP e `api` quando veio de uma integração, com `key_id` e `request_id`. Em integrações bidirecionais, **ignore os eventos cuja `origin.key_id` seja a sua própria chave**: assim, o que você enviou não volta como alteração.

Verificando a assinatura

Validação obrigatória no seu endpoint

NODE.JS

```
import crypto from "node:crypto";
import express from "express";

const app = express();
// Use o corpo BRUTO: a assinatura é calculada sobre os bytes recebidos
app.post("/webhooks/gcp", express.raw({ type: "application/json" }), (req, res) => {
  const header = req.get("GCP-Signature") || "";          // t=1793548800,v1=ab12…
  const { t, v1 } = Object.fromEntries(header.split(",").map(p => p.split("=")));
  const esperado = crypto.createHmac("sha256", process.env.GCP_WEBHOOK_SECRET)
    .update(`${t}.${req.body}`).digest("hex");
  const valido = v1 && crypto.timingSafeEqual(Buffer.from(v1), Buffer.from(esperado));
  const recente = Math.abs(Date.now() / 1000 - Number(t)) < 300; // 5 minutos
  if (!valido || !recente) return res.sendStatus(400);

  const evento = JSON.parse(req.body);
  if (evento.origin.type === "api" && evento.origin.key_id === MINHA_CHAVE_ID) return
res.sendStatus(200); // evita laço
  fila.publicar(evento);     // processe de forma assíncrona
  res.sendStatus(200);      // responda em até 10 segundos
});
```

PYTHON

```
import hmac, hashlib, time, json
from flask import Flask, request, abort

app = Flask(__name__)

@app.post("/webhooks/gcp")
def gcp_webhook():
    partes = dict(p.split("=", 1) for p in request.headers.get("GCP-Signature", "").split(","))
    corpo = request.get_data()  # bytes brutos
    esperado = hmac.new(SECRET.encode(), f"{partes['t']}.".encode() + corpo,
hashlib.sha256).hexdigest()
    if not hmac.compare_digest(esperado, partes.get("v1", "")) or abs(time.time() - int(partes["t"]))
> 300:
        abort(400)
    evento = json.loads(corpo)
    processar_em_segundo_plano(evento)
    return "", 200
```

PHP

```
<?php
$corpo = file_get_contents("php://input");
```

```
parse_str(str_replace(",", "&", $_SERVER["HTTP_GCP_SIGNATURE"] ?? ""), $sig);
$esperado = hash_hmac("sha256", $sig["t"] . "." . $corpo, getenv("GCP_WEBHOOK_SECRET"));
if (!hash_equals($esperado, $sig["v1"] ?? "") || abs(time() - (int)$sig["t"]) > 300) {
  http_response_code(400); exit;
}
$evento = json_decode($corpo, true);
// enfileire e responda rápido
http_response_code(200);
```

Entregas e novas tentativas

TENTATIVA	ESPERA
1ª	Imediata
2ª a 8ª	1 min · 5 min · 30 min · 2 h · 6 h · 12 h · 24 h

É considerada entregue a resposta HTTP 2xx em até 10 segundos. Depois de 30 falhas seguidas, o webhook é pausado automaticamente (reative com `PATCH {"status": "active"}` ou no Painel). O histórico fica em `GET /v1/webhooks/{id}/deliveries`, e `POST /v1/webhooks/{id}/test` envia um evento de teste. A ordem de entrega não é garantida: use `created_at` e `data.object.updated_at`.

Feed de eventos (sincronização incremental)

`GET /v1/events` devolve os eventos em ordem cronológica, com filtro opcional `types` (separados por vírgula). Guarde o cursor e retome de onde parou: é a forma mais robusta de manter uma cópia fiel, e também uma rede de segurança para webhooks perdidos. Os eventos ficam disponíveis por 30 dias.

Consumindo o feed

NODE.JS

```
// Sincronização incremental: guarde o cursor e retome de onde parou
let cursor = await banco.ler("gcp_cursor");
while (true) {
  const qs = new URLSearchParams({ limit: "200", ...(cursor && { cursor }) });
  const page = await (await fetch(`https://gestaoculturalpro.com.br/api/v1/events?${qs}`, { headers
})).json();
  for (const evento of page.data) await aplicar(evento); // idempotente por evento.id
  cursor = page.next_cursor || page.cursor_for_resume || cursor;
  await banco.gravar("gcp_cursor", cursor);
  if (!page.has_more) break;
}
```

12 Receitas de integração

ERP financeiro bidirecional

1. **Mapeie os cadastros.** Leia `/financial/accounts`, `/financial/categories` e `/projects` e associe às contas, ao plano de contas e aos centros de custo do ERP. Guarde os ids do GCP no ERP.
2. **Sincronize contatos por CPF/CNPJ.** Busque com `GET /financial/contacts?document=…`; se não existir, crie. Em 409 `contact_exists`, use o `existing_id`.
3. **Envie contas a pagar e a receber.** `POST /financial/transactions` com `Idempotency-Key` igual ao id do título no ERP e, quando houver, o rateio em `allocations` (projeto e rubrica).
4. **Registre baixas.** Pagamento efetuado no ERP: `POST /financial/transactions/{id}/settle` com data, valor efetivo e `bank_transaction_id`.
5. **Receba o que muda no GCP.** Webhook com `transaction.*`, ignorando a origem da sua própria chave. Use o feed de eventos para reconciliar diariamente.
6. **Confira saldos.** `GET /financial/accounts/{id}/statement` para bater o extrato e `/financial/summary` para o fechamento do mês.

Gestão de tarefas

1. **Estruture.** Para cada projeto/board da ferramenta, crie ou localize um quadro (`/workspace/boards`) e suas listas.
2. **Associe pessoas.** Leia `/workspace/members` e associe pelo e-mail.
3. **Sincronize cartões.** Crie e atualize com `/workspace/cards`; mova entre fases alterando `list_id`; conclua com `completed`.
4. **Escute mudanças.** Webhook com `card.*` e `list.*`; `card.moved` e `card.completed` indicam a transição.

BI e relatórios (somente consulta)

1. **Carga inicial.** Percorra `/financial/transactions` por período (`date_from` / `date_to`), com paginação.
2. **Atualização.** Consuma `/events` a cada poucos minutos e aplique as mudanças pelo `data.object.id`.
3. **Indicadores prontos.** `/financial/summary` (por mês, categoria, projeto, conta ou centro de custo) e `/projects/{id}/financial-summary` (execução por rubrica).

Avisos automáticos para a equipe

Com `workspace:write`, publique em um canal: `POST /workspace/channels/{id}/messages` com `{"text": "Pagamento da NF 4821 confirmado no ERP."}`.

13 Equivalência com softwares de mercado

Para acelerar o mapeamento, a tabela relaciona os conceitos do GCP aos termos usados por ERPs, sistemas financeiros e ferramentas de gestão de tarefas amplamente usados no Brasil (como TOTVS, SAP Business One, Omie, Conta Azul, Bling, Asana, Trello, Jira, monday.com e ClickUp). Os nomes são citados apenas como referência de mercado; as marcas pertencem aos respectivos titulares.

Financeiro

GCP	TERMOS EQUIVALENTES EM ERPS	OBSERVAÇÃO
Lançamento `expense`	Conta a pagar · título a pagar · despesa	`date` = pagamento; `due_date` = vencimento
Lançamento `income`	Conta a receber · título a receber · receita	Patrocínios, editais, bilheteria, vendas
`settle` / `reopen`	Baixa / estorno da baixa · liquidação	Diferenças ficam em `settlements`
`reconciled`	Conciliação bancária	`bank_transaction_id` = FITID do OFX
Transferência	Transferência entre contas · movimentação interna	Dois lançamentos pareados
Contato	Cliente · fornecedor · favorecido · parceiro de negócio	Chave natural: CPF/CNPJ
Categoria	Plano de contas · natureza financeira · categoria	Código hierárquico em `code`
Centro de custo	Centro de custo · departamento	Lista em `/financial/categories`
Projeto + rubrica (`allocations`)	Projeto · obra · centro de resultado + item orçamentário	Exclusivo da gestão cultural: execução por rubrica para prestação de contas
Planejamento	Previsão · orçamento de caixa · provisão	Vira lançamento ao ser processado no GCP

Gestão de tarefas

GCP	TERMOS EQUIVALENTES
Quadro	Projeto · board · espaço · quadro
Lista (fase)	Seção · coluna · status · grupo

GCP	TERMOS EQUIVALENTES
Cartão	Tarefa · card · issue · item
Checklist	Subtarefas · checklist · itens de ação
Etiqueta	Tag · label · rótulo
`members` / `assignee_id`	Responsáveis · assignee
`custom_fields`	Campos personalizados · custom fields · colunas
`blocked_by`	Dependências · bloqueado por
Comentário	Comentário · atualização · activity

14 Segurança

Na plataforma

HTTPS obrigatório com HSTS · segredo armazenado apenas como SHA-256, com comparação em tempo constante · isolamento por instituição em todas as consultas · permissões por módulo e ação · validade e renovação com transição · limite de requisições · auditoria de cada chamada (rota, resultado, tempo, IP) no Painel · HTML recebido é higienizado · remoções com salvaguardas.

Nos webhooks

Assinatura HMAC-SHA256 com timestamp · somente HTTPS · bloqueio de endereços internos, privados e de metadados, verificado no momento da conexão (protege contra DNS rebinding) · sem seguir redirecionamentos · tempo limite de 10 s · pausa automática após falhas consecutivas.

Checklist do parceiro: chave em cofre de segredos · uma chave por sistema · menor permissão necessária · renovação periódica (90 dias) · validação da assinatura e do timestamp dos webhooks · `Idempotency-Key` em todo POST · `If-Match` em alterações · espera exponencial em 429/5xx · registro do `X-Request-Id` nos seus logs.

15 Referência rápida

MÉTODO	ROTA	PERMISSÃO	DESCRIÇÃO
GET	`/v1/ping`	`qualquer`	Diagnóstico da chave
GET	`/v1/financial/accounts`	`financial:read`	Listar contas
GET	`/v1/financial/accounts/{id}`	`financial:read`	Consultar conta com saldos
POST	`/v1/financial/accounts`	`financial:write`	Criar conta
PATCH	`/v1/financial/accounts/{id}`	`financial:write`	Alterar conta
DELETE	`/v1/financial/accounts/{id}`	`financial:delete`	Remover conta sem movimentação
GET	`/v1/financial/accounts/{id}/statement`	`financial:read`	Extrato com saldo acumulado
GET	`/v1/financial/transactions`	`financial:read`	Listar lançamentos
GET	`/v1/financial/transactions/{id}`	`financial:read`	Consultar lançamento (inclui etag)
POST	`/v1/financial/transactions`	`financial:write`	Criar receita ou despesa
PATCH	`/v1/financial/transactions/{id}`	`financial:write`	Alterar lançamento
POST	`/v1/financial/transactions/{id}/settle`	`financial:write`	Registrar baixa (pagamento ou recebimento)
POST	`/v1/financial/transactions/{id}/reopen`	`financial:write`	Desfazer baixa
DELETE	`/v1/financial/transactions/{id}`	`financial:delete`	Excluir lançamento em aberto
POST	`/v1/financial/transfers`	`financial:write`	Criar transferência
GET	`/v1/financial/contacts`	`financial:read`	Listar ou buscar
GET	`/v1/financial/contacts/{id}`	`financial:read`	Consultar contato

MÉTODO	ROTA	PERMISSÃO	DESCRIÇÃO
POST	`/v1/financial/contacts`	`financial:write`	Criar contato
PATCH	`/v1/financial/contacts/{id}`	`financial:write`	Alterar contato
DELETE	`/v1/financial/contacts/{id}`	`financial:delete`	Remover contato sem lançamentos
GET	`/v1/financial/categories`	`financial:read`	Listar categorias, centros de custo e listas
GET	`/v1/financial/planning`	`financial:read`	Listar itens
GET	`/v1/financial/planning/{id}`	`financial:read`	Consultar item
POST	`/v1/financial/planning`	`financial:write`	Criar item
PATCH	`/v1/financial/planning/{id}`	`financial:write`	Alterar item em rascunho
DELETE	`/v1/financial/planning/{id}`	`financial:delete`	Remover item em rascunho
GET	`/v1/financial/summary`	`financial:read`	Resumo por período
GET	`/v1/projects`	`financial:read ou workspace:read`	Listar projetos
GET	`/v1/projects/{id}`	`financial:read ou workspace:read`	Consultar projeto com orçamento (budget[])
GET	`/v1/projects/{id}/financial-summary`	`financial:read`	Execução orçamentária por rubrica
GET	`/v1/workspace/members`	`workspace:read`	Listar membros ativos
GET	`/v1/workspace/boards`	`workspace:read`	Listar quadros
GET	`/v1/workspace/boards/{id}`	`workspace:read`	Consultar quadro com listas, etiquetas e campos personalizados
POST	`/v1/workspace/boards`	`workspace:write`	Criar quadro
PATCH	`/v1/workspace/boards/{id}`	`workspace:write`	Alterar título, descrição ou etiquetas
GET	`/v1/workspace/boards/{id}/lists`	`workspace:read`	Listar listas (fases)

MÉTODO	ROTA	PERMISSÃO	DESCRIÇÃO
POST	`/v1/workspace/boards/{id}/lists`	`workspace:write`	Criar lista
PATCH	`/v1/workspace/lists/{id}`	`workspace:write`	Renomear, reordenar ou arquivar lista
DELETE	`/v1/workspace/lists/{id}`	`workspace:delete`	Arquivar lista (?permanent=true remove lista vazia)
GET	`/v1/workspace/cards`	`workspace:read`	Listar cartões de um quadro
GET	`/v1/workspace/cards/{id}`	`workspace:read`	Consultar cartão (inclui etag)
POST	`/v1/workspace/cards`	`workspace:write`	Criar cartão
PATCH	`/v1/workspace/cards/{id}`	`workspace:write`	Alterar, mover (list_id), concluir ou arquivar
DELETE	`/v1/workspace/cards/{id}`	`workspace:delete`	Arquivar cartão (?permanent=true remove definitivamente um cartão já arquivado)
GET	`/v1/workspace/cards/{id}/comments`	`workspace:read`	Listar comentários
POST	`/v1/workspace/cards/{id}/comments`	`workspace:write`	Comentar
GET	`/v1/workspace/tasks`	`workspace:read`	Listar tarefas
GET	`/v1/workspace/tasks/{id}`	`workspace:read`	Consultar tarefa
POST	`/v1/workspace/tasks`	`workspace:write`	Criar tarefa
PATCH	`/v1/workspace/tasks/{id}`	`workspace:write`	Alterar tarefa
DELETE	`/v1/workspace/tasks/{id}`	`workspace:delete`	Remover tarefa (tarefas com Logística: use status cancelled)
GET	`/v1/workspace/channels`	`workspace:read`	Listar canais
GET	`/v1/workspace/channels/{id}/messages`	`workspace:read`	Mensagens recentes (mais novas primeiro)
POST	`/v1/workspace/channels/{id}/messages`	`workspace:write`	Publicar mensagem

MÉTODO	ROTA	PERMISSÃO	DESCRIÇÃO
GET	`/v1/workspace/docs`	`workspace:read`	Listar documentos colaborativos
GET	`/v1/events`	`leitura do módulo`	Feed de eventos
GET	`/v1/events/{id}`	`leitura do módulo`	Consultar evento
GET	`/v1/webhooks/event-types`	`qualquer`	Catálogo de eventos
GET	`/v1/webhooks`	`webhooks:manage`	Listar webhooks
POST	`/v1/webhooks`	`webhooks:manage`	Cadastrar webhook (retorna o secret uma vez)
GET	`/v1/webhooks/{id}`	`webhooks:manage`	Consultar webhook
PATCH	`/v1/webhooks/{id}`	`webhooks:manage`	Alterar URL, eventos ou status
DELETE	`/v1/webhooks/{id}`	`webhooks:manage`	Remover webhook
POST	`/v1/webhooks/{id}/rotate-secret`	`webhooks:manage`	Gerar novo secret
POST	`/v1/webhooks/{id}/test`	`webhooks:manage`	Enviar evento de teste
GET	`/v1/webhooks/{id}/deliveries`	`webhooks:manage`	Histórico de entregas
POST	`/v1/webhooks/{id}/deliveries/{deliveryId}/retry`	`webhooks:manage`	Reenviar entrega

16 Versões e suporte

VERSÃO	DATA	NOVIDADES
1.0.0	06/11/2026	Lançamento: Módulo Financeiro, Projetos, Espaços de Trabalho, webhooks assinados, feed de eventos, idempotência e controle de versão.

Suporte técnico: abra um chamado em *Suporte e Ajuda* no GCP (ou peça ao administrador da instituição), informando o `request_id`, a rota, o horário (Brasília) e a resposta recebida. Nunca envie a chave de API em chamados ou e-mails.

© 2026 Gestão Cultural Pro · API v1.0.0

Desenvolvimento: Bruno Albertini | Gestão Cultural Pro