PLANA Solutions

Payhawk × Odoo

Odoo 19.0

Complete integration guide

Generated from docs.planapulse.com
Built 2026-06-09 · git ee11e9c5

Payhawk × Odoo 19 — Quick Start

This guide takes you from a fresh Odoo 19 instance to a working, two-way Payhawk integration. Follow the steps in order. The whole process takes about **20–30 minutes**.

Which version?

You are reading the **Odoo 19** guide (module `19.0.x`). Switch versions from the **Payhawk version** menu in the top bar. See version differences for what changed.

Download the guide

Prefer to read offline, or working in an environment where `docs.planapulse.com` isn't reachable? Get the entire guide as a single PDF.

Video walkthrough

Four short, topic-scoped demos — watch the general walkthrough first, then dive into the specifics.

1. **General walkthrough** 14 min

   End-to-end overview of the integration.

   Watch online: youtube.com/watch?v=-Vmy-1UZFQU

2. **Settings passed between Payhawk and Odoo** 7 min

   What configuration syncs in each direction and how to set it up.

   Watch online: youtube.com/watch?v=v82Rwu6oo18

3. **Expenses in Payhawk** 5 min

   Capturing and reviewing expenses on the Payhawk side.

   Watch online: youtube.com/watch?v=y_MC9O6jru4

4 How the info is transformed into Odoo

2 min

How reviewed expenses become bills, receipts, and bank statement lines.

Watch online: youtube.com/watch?v=CyIlLbFXz94

Download the module

Get the latest packaged build of the connector for Odoo 19.0. Unzip into your Odoo addons path, then update the apps list and install.

At a glance

Step	What you do	Where
1	Check prerequisites	Odoo apps + server
2	Install the module	Odoo Apps
3	Get an API key	Payhawk portal
4	Configure the connection	Settings → Payhawk
5	Run the first sync	Settings → Payhawk
6	Enable webhooks	Odoo + Payhawk
7	Go live	Switch to Production

How the integration works

```
       Odoo 19                          Payhawk
  ┌──────────────┐                 ┌──────────────┐
  │ Expense accts │ ──push codes──▶ │ Account codes │
  │ Taxes         │ ──push rates──▶ │ Tax rates     │
  │ Products      │ ──categories──▶ │ Categories    │
  │ Analytics     │ ──cust fields─▶ │ Custom fields │
  │ Suppliers     │ ◀──two-way────▶ │ Suppliers     │
  │ Bills/Receipts│ ◀──expenses──── │ Expenses      │
  │ Bank stmts    │ ◀──fund accts── │ Fund accounts │
  └──────────────┘                 └──────────────┘
        ▲                                  │
        └──────── webhooks (real-time) ────┘
            + scheduled sync every 15 min
```

Data is pushed and pulled both by a **scheduled job** (every 15 minutes) and in **real time via webhooks** when an expense is reviewed or a supplier changes.

Start with Prerequisites →

1. Prerequisites

Before installing the Payhawk Connector, make sure the following are in place.

Odoo apps

The module depends on these apps. Install them first (Apps → search → Install):

- **Invoicing / Accounting** (`account`)
- **Analytic Accounting** (`analytic`)
- **Purchase** (`purchase`)
- **Inventory** (`stock`)
- **Employees** (`hr`)

> **WARNING**
>
> If any dependency is missing, the module will refuse to install. Install the apps above **before** the Payhawk Connector.

Python dependency

The module requires the `ratelimit` Python package on the Odoo server:

```bash
pip install ratelimit
```

On a containerized / managed PLANA deployment this is already baked into the image. On a self-managed server, install it into the same environment that runs Odoo, then restart the Odoo service.

Permissions

You need **Accounting** → **Account Manager** rights to see and edit the Payhawk settings. The integration creates two inactive service users automatically — you do not manage them manually:

- `[CRON] Payhawk Data Broker` — runs the scheduled sync
- `[WEBHOOK] Payhawk Data Broker` — handles inbound webhooks

A Payhawk account

You need a Payhawk account with access to the **Developer API** so you can generate an API key. Start with a **Sandbox** key for testing; switch to **Production** once verified. See Get a Payhawk API key.

Recommended order

1. Install the Odoo apps above
2. Confirm `ratelimit` is installed
3. Install the Payhawk Connector module

Next: Install the module →

2. Install the module

The connector module is named `payhawk` ("Payhawk Connector").

On a PLANA-managed instance

The module ships with your PLANA Pulse / Plana Business Cloud deployment. To enable it:

1. Turn on **Developer Mode** (Settings → scroll down → *Activate the developer mode*).
2. Go to **Apps**, click **Update Apps List**.
3. Search for **Payhawk Connector** and click **Install**.

On a self-managed instance

1. Place the `payhawk` module in your Odoo addons path.
2. Restart Odoo.
3. **Apps** → **Update Apps List** → **Install "Payhawk Connector"**.

Verify the install

After installation you should see a new **Payhawk** section under **Settings** (Accounting must be installed). If you do not see it:

- Confirm you have **Account Manager** rights
- Confirm the dependencies are installed
- Re-run **Update Apps List** and retry

The module also registers:

- A scheduled action **"Payhawk Data Sync Runner"** (runs every 15 minutes)
- Two inactive service users (cron + webhook brokers)
- A supplier contact tag **"Payhawk Supplier"**

Next: Get a Payhawk API key →

3. Get a Payhawk API key

The connector authenticates to Payhawk with an **API key** sent in the `X-Payhawk-ApiKey` header. You generate this key in Payhawk.

> **Start in Sandbox**
>
> Always validate the integration with a **Sandbox** key first. Switch to a **Production** key only after a clean test sync. The Odoo side keeps Sandbox and Production keys separately, so you can hold both at once.

Generate the key in Payhawk

1. Sign in to your Payhawk account.
2. Open **Settings** → **Developer API** (you need developer/admin access). See Payhawk's API overview.
3. Create an API key and **copy it** — you will paste it into Odoo.

Group (multi-entity) accounts

If your organization uses a **Payhawk group account** (multiple legal entities under one group), you will also have a **Group ID** and **Group API key**. The connector supports group accounts — you enter these in the configuration step.

> **v19 specifics**
>
> In Odoo 19 the Group ID and Group Key are **company-dependent** — each Odoo company holds its own group settings. (In v17/v18 they were global.) See version differences.

What you do not need to fill

The **Account ID** is discovered automatically on the first sync — leave it blank. Payhawk returns it from the `/accounts` endpoint and the connector stores it.

Next: Configuration →

4. Configuration

Open **Settings** → **Payhawk**. (Requires Account Manager rights. Settings are per-company — switch to the right company first if you run multi-company.)

Core connection

Field	What to enter
API Mode	`Sandbox` for testing, `Production` for live. Sandbox and Production keys are stored separately.
API Key	Paste the key from Payhawk. Bound to the selected API Mode.
Account ID	Leave blank — filled automatically on first sync.
Bills Journal	Purchase journal for Payhawk expenses/bills. Defaults to the company purchase journal if empty.
Receipts Journal	Purchase journal for expenses that have receipts. Defaults to purchases if empty.

Group accounts

If you operate a Payhawk **group** (multi-entity) account, also fill:

Field	What to enter
Group ID	Your Payhawk group identifier.
Group Key	The group-level API key.

v19

In Odoo 19 these group fields are **company-dependent** — set them per company.

Choose what to sync

Open the **Miscellaneous** tab. Each toggle controls one sync direction. Defaults are sensible for most clients:

Toggle	Default	Effect
Sync Tax Rates	on	Push Odoo purchase taxes → Payhawk tax rates
Sync Chart of Accounts	on	Push expense accounts → Payhawk account codes
Sync Departments	on	Sync Odoo departments ↔ Payhawk teams
Sync Suppliers	on	Two-way supplier sync
Sync Analytic Accounts	on	Push analytic plans/accounts → custom fields
Sync Branches	on	Sync branches
Sync Assets	on	Push asset models (OCA or Enterprise)
Import Bank Statements	on	Pull fund-account transactions into Odoo
Sync Categories	off	Push expense products as categories

Other useful options on this tab:

- **Auto Post** — automatically post imported bills/receipts
- **Supplier on Bank Statements** — include supplier on statement lines
- **Clear Teams** — remove Payhawk teams with no matching Odoo department
- **Delete Logs Older Than** — log retention in days (0 = keep forever)
- **Request Timeout** — API timeout in seconds (default 10)
- **Disable Error Reporting** — stop sending error diagnostics to PLANA (leave **on** so we can help proactively)

See the full reference in Sync options.

Select expense products & taxes to push

- **Expense Products** tab — pick the products exported to Payhawk as categories (use the *Export Expense Categories* button).
- **Tax Rates** tab — pick the purchase taxes pushed to Payhawk.

Next: First synchronization →

Configuration check

The connector ships a **self-service configuration check** that validates the setup and reports exactly what is wrong — so most onboarding issues are caught without contacting support.

Settings → Payhawk → Sync Log → "Run Configuration Check". It opens a report with one line per check, colour-coded **OK / Warning / Error**.

What it validates

Check	What it means
API key	A key is set for the active mode (Sandbox/Production).
Payhawk account	The key authenticates and an account is resolved.
Instance binding	The Payhawk group's `odoo_internal_data → instance_id` matches **this** Odoo instance (see below).
Taxes — type	Only **purchase** taxes are selected (Payhawk is spend management).
Taxes — duplicate code	No two selected taxes share the same name → Payhawk rejects duplicate codes.
Taxes — mapping	Each selected tax matches a Payhawk tax-rate code (catches EN/ES translation mismatches).
Currencies	Every Payhawk fund-account currency exists and is active in Odoo.
Supplier external IDs	No Payhawk supplier carries a stale external id from another Odoo instance.
Bills / Fund journals	A purchase journal exists; `PHWK` / `PSND` fund journals are present.
Chart of accounts / Webhooks	Expense accounts available; webhook secret set when enabled.

Required configuration

1. **API key** — per mode. Sandbox and Production keys are stored separately.
2. **Group ID + Group Key** — **required when the Payhawk account belongs to a group** (most partner/sandbox accounts do). Without them, creating the integration's custom field returns `403 Forbidden` and the whole sync aborts. Set both in Settings → Payhawk.
3. **Odoo instance binding** — on the first sync the connector writes this Odoo's instance id into the Payhawk group custom field **`odoo_internal_data → instance_id`** (anti cross-instance corruption guard). On a **shared sandbox group** a previous Odoo may already own it; the check then reports a mismatch. **Fix:** edit `odoo_internal_data → instance_id` in the Payhawk group UI to the instance id shown by the check.
4. **Taxes** — select **only purchase taxes**, and make sure each selected tax has a **unique name** (the name becomes the Payhawk code; duplicates → `400 duplicate_code`).
5. **Departments → Teams** — needs team-create permission in Payhawk. If the account/key can't create teams (`403 /teams`), turn off **Sync Departments**.

Reset supplier external IDs

When suppliers were previously synced from a **different** Odoo instance (e.g. a shared sandbox), their Payhawk external ids point at the wrong partner and won't update. A one-time maintenance action clears them so the next sync re-links by VAT/name.

It lives in **developer mode** (Settings → Payhawk → Sync Log → **Reset Supplier External IDs**) — intentionally hidden so it cannot be triggered by accident. After running it, start a normal sync.

> **WARNING**
>
> This rewrites every Payhawk supplier's external id. Use it only for the stale-id case above.

5. First synchronization

With the API key and toggles set, run the first sync.

1. Go to **Settings** → **Payhawk** → **Sync Log** tab.
2. Click **Synchronize data with Payhawk**.

What happens on the first sync

The connector runs through the enabled steps in order and logs each one:

1. **Account ID discovery** — calls Payhawk `/accounts`, stores the returned ID (the previously blank Account ID field is now filled).
2. **Fund accounts** — for each Payhawk fund account, a bank journal is auto-created in Odoo with a prefixed code:
    - `PHWK…` for **Production**
    - `PSND…` for **Sandbox**
3. **Push** — chart of accounts, tax rates, categories, analytics, departments (per your toggles).
4. **Suppliers** — two-way reconciliation by VAT / external ID / name.
5. **Pull** — reviewed expenses become bills/receipts; fund transactions become bank statement lines.

Check the **Fund Accounts** tab — the auto-created journals are listed there.

Reading the log

Entries are colour-coded:

- **success** (green) — a sync completed cleanly; the timestamp marks the watermark for the next incremental pull
- **info** — normal progress messages
- **warning** (amber) — non-fatal (e.g. a supplier email that failed validation was skipped)
- **error** (red) — something failed; see Troubleshooting

Re-pull from the past

The connector pulls expenses created **after** the last success entry. To force a re-pull, edit the date on the most recent success log entry to an earlier time and run the sync again.

Automatic sync

After the first manual run, the **Payhawk Data Sync Runner** scheduled action keeps things in sync **every 15 minutes** — you do not need to click the button again.

Next: Webhooks →

6. Webhooks

Webhooks give you **real-time** updates: when an expense is reviewed or a supplier changes in Payhawk, Odoo is notified immediately instead of waiting for the 15-minute scheduled sync.

Enable webhooks in Odoo

1. **Settings** → **Payhawk** → **Miscellaneous** tab.
2. Turn on **Enable Webhooks**.
3. A **Webhook Secret** (a UUID) is generated automatically. This secret is part of the webhook URL, so each company has a unique, unguessable endpoint.

Your webhook URL

The endpoint is:

```
https://<your-odoo-domain>/payhawk/webhooks/<webhook-secret>
```

For example:

```
https://erp.example.com/payhawk/webhooks/bd46acfc-f5ca-438f-bd2c-d871013f8b16
```

Register the URL in Payhawk

In Payhawk, add this URL as a webhook endpoint. The connector handles these events:

Event	Effect in Odoo
`expense.reviewed`	Create/refresh the bill or receipt for the expense
`expense.payment-details.updated`	Update payment details on the linked move
`supplier.created`	Import the new supplier
`supplier.updated`	Update the matching supplier

How requests are authenticated

The secret in the URL **is** the authentication. An unknown secret is rejected with `401 Unauthorized`. If webhooks are toggled off in Odoo, inbound calls return a `disabled` status and nothing is processed — so you can pause without removing the endpoint in Payhawk.

> **Keep the secret private**
>
> Anyone with the full webhook URL can post events to your Odoo. Treat it like a password. Toggling **Enable Webhooks** off and on regenerates behavior tied to the stored secret — coordinate with the Payhawk-side endpoint if you rotate it.

Next: Go live →

7. Go live

Once the Sandbox sync is clean, switch to Production.

Switch to Production

1. **Settings** → **Payhawk** → set **API Mode** to `Production`.
2. Paste the **Production** API key (the Sandbox key is kept separately and stays).
3. If you use a group account, confirm the **Group ID / Group Key** for Production.
4. Run **Synchronize data with Payhawk** once manually.
5. Confirm new `PHWK…` fund-account journals appear (Production prefix).
6. Re-register the webhook URL in your Production Payhawk account.

Sandbox vs Production journals

Sandbox creates `PSND…` journals; Production creates `PHWK…` journals. They coexist — expect both sets if you tested in Sandbox first.

Go-live checklist

- [ ] Production API key entered, API Mode = Production
- [ ] First manual Production sync completed without errors in the Sync Log
- [ ] Account ID auto-filled
- [ ] `PHWK…` fund-account journals created
- [ ] Bills/Receipts journals selected (or default purchase journal acceptable)
- [ ] Expense products & tax rates selected and pushed
- [ ] Suppliers reconciled (spot-check a few VAT numbers both sides)
- [ ] Webhooks enabled and URL registered in Production Payhawk
- [ ] Scheduled action "Payhawk Data Sync Runner" is active
- [ ] Error reporting left **enabled** (so PLANA can help proactively)

After go-live

- The scheduled sync runs every **15 minutes**; webhooks handle real-time events.
- Watch the **Sync Log** for the first day; warnings are usually data-quality issues (e.g. an invalid supplier email) and are safe.
- Old log entries auto-prune per the **Delete Logs Older Than** setting.

Need help? See Troubleshooting.

Data flow reference

What moves between Odoo and Payhawk, and in which direction.

Push: Odoo → Payhawk

Odoo object	Becomes in Payhawk	Trigger
Expense accounts (`account.account`, group *expense*)	Account codes	Scheduled / manual sync
Purchase taxes (`account.tax`, selected)	Tax rates	Scheduled / manual sync
Expense products (`product.product`, selected)	Categories	*Export Expense Categories* / sync (if enabled)
Analytic plans (`account.analytic.plan`)	Custom fields	Scheduled / manual sync
Analytic accounts (`account.analytic.account`)	Custom field values	Scheduled / manual sync
Departments (`hr.department`)	Teams	Scheduled / manual sync
Asset models (OCA / Enterprise)	Amortizations	Scheduled / manual sync

Pull: Payhawk → Odoo

Payhawk object	Becomes in Odoo	Trigger
Reviewed expense	Vendor bill or receipt (linked)	Webhook `expense.reviewed` + sync
Expense payment details	Payment info on the move	Webhook `expense.payment-details.updated`
Fund-account transactions	Bank statement lines	Scheduled sync (if *Import Bank Statements*)
Suppliers	`res.partner` (tagged *Payhawk Supplier*)	Sync + webhooks

Two-way: Suppliers

Suppliers reconcile in both directions. Matching priority:

1. **VAT / Tax ID**
2. **External ID** (Odoo partner ID stored on the Payhawk supplier)
3. **Name**

New Odoo suppliers are pushed to Payhawk (batches of 50/run); new Payhawk suppliers without an external ID are imported into Odoo and back-linked. Invalid emails are skipped with a warning rather than failing the run.

Identity & integrity

- The connector stores the **Odoo instance identifier** in a hidden Payhawk custom field (`odoo_internal_data` → `instance_id`) to guard against cross-instance mixups.
- The **Account ID** links the Odoo company to a specific Payhawk account; it is discovered and stored on first sync.

Timing

- **Scheduled:** "Payhawk Data Sync Runner" every **15 minutes** (incremental — pulls what changed since the last success watermark).
- **Real-time:** webhooks for expense review, payment changes, supplier create/update.

Sync options reference

Every option lives on **Settings** → **Payhawk** → **Miscellaneous**. Defaults suit most clients.

Sync directions

Option	Default	Direction	Notes
Sync Tax Rates	on	Odoo → Payhawk	Pushes selected purchase taxes as Payhawk tax rates. Select taxes on the *Tax Rates* tab.
Sync Chart of Accounts	on	Odoo → Payhawk	Pushes accounts with internal group *expense* as Payhawk account codes.
Sync Departments	on	Odoo ↔ Payhawk	Maps departments to Payhawk teams (by external ID / name, preserving hierarchy).
Sync Suppliers	on	Odoo ↔ Payhawk	Two-way; matches by VAT, external ID, then name. Batches of 50 per run.
Sync Analytic Accounts	on	Odoo → Payhawk	Analytic plans become custom fields; accounts become field values.
Sync Branches	on	Odoo ↔ Payhawk	Branch sync.
Sync Assets	on	Odoo → Payhawk	Pushes asset models — supports OCA *account_asset_management* and Odoo Enterprise *account_asset*.
Import Bank Statements	on	Payhawk → Odoo	Pulls fund-account transactions into bank statement lines.
Sync Categories	off	Odoo → Payhawk	Pushes expense products as Payhawk categories.

Behavior options

Option	Default	Effect
Auto Post	on	Posts imported bills/receipts automatically instead of leaving them in draft.
Supplier on Bank Statements	on	Adds the supplier to imported statement lines.
Clear Teams	off	Deletes Payhawk teams that have no matching Odoo department.
Delete Logs Older Than	10	Days of sync-log retention; `0` keeps logs forever.
Request Timeout	10	Per-request API timeout in seconds.
Max Records	0	Cap on records fetched per request; `0` = all (Payhawk still caps at 1000/request).
Disable Error Reporting	off	When off, error diagnostics are sent to PLANA for proactive support. Leave off unless required.

Journals & prefixes

Fund-account journals are auto-created with a mode prefix:

- **Production** → `PHWK…`
- **Sandbox** → `PSND…`

You may set a dedicated **Bills Journal** and **Receipts Journal**; otherwise the company's default purchase journal is used.

Rate limiting

API calls are rate-limited client-side to **15 requests/second** with automatic back-off, so large syncs are throttled gracefully rather than rejected.

Posting bills vs receipts

Corporate-card expenses fall into two natural categories:

Document type	Use for
Bill	A real vendor invoice with a registered supplier and a due date (SaaS subscription, vendor with `BG/INV-12345`, recurring services)
Receipt	A cash-like purchase with no invoice (petrol-pump receipt, restaurant, ad-hoc retail)

By default Payhawk treats every reviewed expense as a bill. Two settings let the reviewer pick **Document type** per expense and post receipts to a separate Odoo journal — keeping the AP ageing report clean and matching the accounting reality.

Step 1 — Enable the Document Type field in Payhawk

In Payhawk:

1. **Settings** → **Expense fields**
2. Find the field **Document type**
3. Toggle it **Enabled**

After this, every expense card in Payhawk has a **Document type** picker during review with values **Bill** and **Receipt**. The reviewer is prompted to choose before approval.

Step 2 — Configure the Receipts journal in Odoo

In Odoo:

1. **Settings** → **Payhawk** (Account Manager rights required)
2. Set **Receipts Journal** to the journal where you want receipts to land

Field	Pick
Bills Journal	The standard purchase journal — formal vendor invoices
Receipts Journal	A separate purchase journal — cash-like receipts

If **Receipts Journal** is empty, every expense posts to the **Bills Journal** regardless of Document type — the connector defaults to bills for backwards compatibility.

Creating the Receipts journal

A common pattern is a dedicated **Purchase**-type journal named `PHWK Receipts` or `Petty Cash — Payhawk`, with its default credit account set to the corporate-card clearing account. Bills and receipts then share the same expense GL accounts but post via different journals, so reports can break them apart cleanly.

How it flows

```
Reviewer in Payhawk → Document type = Receipt
        │
        ▼
Webhook fires → Payhawk × Odoo connector
        │
   ┌────┴──────────────────┐
   │                       │
   ▼                       ▼
Document type = Bill    Document type = Receipt
   │                       │
   ▼                       ▼
Bills Journal          Receipts Journal
(account.move,         (account.move,
 in_invoice)            in_receipt)
```

Bills go through Accounts Payable as usual (3-way match, due date, ageing). Receipts post directly without an AP step — useful when the expense has no formal invoice to chase.

Verify

After enabling both settings, smoke-test once:

1. In Payhawk Sandbox, create an expense and **review → Document type = Receipt → approve**
2. In Odoo, **Accounting → Vendors → Bills** — the expense should **not** appear here
3. **Accounting → Vendors → Receipts** (or filter the journal items by your Receipts journal) — the entry should appear there
4. Repeat with **Document type = Bill** — entry appears under Bills as normal

If a receipt-typed expense still lands in Bills:

- Confirm **Document type** is **Enabled** in Payhawk
- Confirm **Receipts Journal** is **set** in Odoo (not blank)
- Force a sync via **Settings → Payhawk → Sync now** and re-check

Where to read more

- Configuration — the full settings reference
- Sync options — every toggle on the Miscellaneous tab
- Reimbursements — the related employee-payout setup
- PLANA Business Cloud → Vendor bills — the accounting-side workflow

Working with reimbursements

When someone pays for a business expense from their own pocket, Payhawk records it as a **reimbursement** — an expense paid back to the person rather than charged to a corporate card. For that to work end-to-end, Payhawk needs to know who your reimbursable people are and how to pay them. The Payhawk × Odoo connector syncs that directory for you.

Pick: Users or Employees

In Odoo there are two related models that can stand for "a person":

Odoo model	What it is
`res.users`	A user who can log into Odoo
`hr.employee`	An employee on the payroll, with HR fields (manager, bank account, contract...)

They are usually the same people but not always. A contractor might be an employee without a login. An external accountant might be a user without being on payroll.

In **Settings** → **Payhawk**, the **Reimbursement subjects** option picks which model gets synced to Payhawk:

Option	Sync source	When to pick it
Users	Every active `res.users` record	You don't use the HR app; reimbursable people log into Odoo
Employees	Every active `hr.employee` record	You use the HR app (Pro tier and up); payroll bank accounts live on the employee record

Pick **Employees** if HR is installed — the bank-account-number field is on the employee record, and that's where finance teams keep payroll banking details. Pick **Users** only if HR isn't installed.

Where the bank account comes from

The payout uses the bank account number stored on the synced record:

Mapping	Bank account read from
Users	`res.users.partner_id.bank_ids` (the user's contact has a related bank account)
Employees	`hr.employee.bank_account_id` — the **Private Information** tab on the employee form

Bank account must be set before sync

An employee with no bank account number syncs to Payhawk just fine, but the reimbursement payout step in Payhawk will have no destination IBAN. Populate the field on the Odoo side first, then sync.

How it flows

```
Person in your company submits a reimbursable expense in Payhawk
            │
            ▼
Reviewer approves → Payhawk knows who, how much, which IBAN
                    (IBAN already synced from Odoo)
            │
            ▼
Payhawk schedules the payout to the employee's IBAN
            │
            ▼
Expense posts back to Odoo as a vendor bill (or receipt) against
the employee's partner record — not against an external vendor
```

The person shows up in Payhawk's picker on every reimbursement, with their name, role, and IBAN already attached. No retyping per expense.

Initial sync

After changing **Reimbursement subjects**:

1. **Settings** → **Payhawk** → **Sync now** (full sync) — or specifically **Sync Employees** / **Sync Users** depending on which mapping you chose
2. Wait ~30 seconds
3. In Payhawk → **Settings** → **People** — confirm the list matches what you'd expect from your Odoo data

The scheduled sync (every 15 minutes) keeps things current after that. New employees in Odoo appear in Payhawk on the next sync; archived employees are deactivated in Payhawk.

Payhawk login is separate

A person synced to Payhawk is a **record in Payhawk's directory** — it does **not** automatically give them a Payhawk login. Whether they can sign in to Payhawk and submit their own reimbursements is a separate choice in Payhawk → **Settings** → **People** → invite-by-email.

This is intentional: the reimbursement payout still works for any synced person regardless of whether they have Payhawk login access. A submitter who isn't a Payhawk user has their expense entered by a reviewer on their behalf.

Common gotchas

Issue	Fix
Reimbursement appears in Payhawk but **IBAN is blank**	The Odoo record has no `bank_account_id` (or `partner_id.bank_ids` for User mapping). Fill it, then re-sync.
Employees missing from Payhawk after sync	They are archived or not marked Active in Odoo — Payhawk syncs active records only
User mapping picks the company bank account, not the personal one	On the user's partner record, ensure the **personal** bank account is set; remove or unmark the company account on that partner
Names in Payhawk don't match	The sync uses `hr.employee.name` (or `res.users.name`) verbatim. Update in Odoo and force a sync.

Switching mappings after going live

Changing from **Users → Employees** (or back) is supported but disruptive:

1. Records under the old mapping are archived in Payhawk on the next sync
2. Records under the new mapping appear
3. **Open reimbursements** assigned to old records do **not** move automatically — settle them under the old mapping first, or expect a manual cleanup

Best to pick the right mapping at setup and stick with it. If you must switch, schedule a maintenance window with the Payhawk reviewers.

Where to read more

- Configuration — the full settings reference
- Sync options — every toggle on the Miscellaneous tab

- Bills vs receipts — the related Document Type setup
- PLANA Business Cloud → Employees — where bank accounts live in the HR app

Release notes — Payhawk × Odoo 19

Reverse-chronological list of every release of the Payhawk × PLANA connector packaged for **Odoo 19**. The same release usually ships simultaneously for v17 and v18 — see those versions' release notes for the matching entries.

Want the package?

The Payhawk × PLANA connector is proprietary PLANA software. Open the v19 overview and click **Request the download** — we reply with the package within one business day. This page is the changelog of what's in each release.

Older history

This changelog starts on **2026-05-30**. For details of releases before that date, contact PLANA support via your workspace's Matrix room.

19.0.3.1 — 2026-05-30

Current packaged release. Production-ready for Odoo 19 tenants.

📦 Build

- **Package:** `payhawk-19.0.3.1.zip`
- **Compatible with:** Odoo 19 Community
- **License:** proprietary — PLANA Digital Ltd.

✨ Capabilities verified in this build

- **Document type field** — reviewers can mark each Payhawk expense as `Bill` or `Receipt`. See Bills vs receipts.
- **Separate Receipts Journal** — receipts post to a dedicated journal when configured; otherwise they fall back to the Bills Journal.
- **Reimbursement subjects** — pick **Users** or **Employees** as the Payhawk directory source; bank account flows through automatically. See Reimbursements.

- **Reimbursement support on payment registration** — the `Is Reimbursement` option on `account.payment.register` (inherited from v18 and forward).
- **All standard sync directions** — chart of accounts, taxes, suppliers (two-way), departments ↔ teams, analytic accounts as custom fields, branches, asset models, bank statements pull. Full reference: Sync options.
- **15 req/sec rate limiting** with automatic back-off.
- **Per-company Group ID / Group Key** — each Odoo company stores its own group-account settings. In v16–v18 these were a single global value.
- **Odoo 19 list views** — the modernised list-view syntax.

📌 Differences vs v17 / v18

	v17	v18	v19
Group ID / Group Key scope	Global	Global	**Per-company**
View layer	`<tree>`	`<list>`	`<list>`
`Is Reimbursement` on payment register	—	✓	✓
Document type field	✓	✓	✓
Separate Receipts Journal	✓	✓	✓
Reimbursement subjects (Users/Employees)	✓	✓	✓

📌 Configuration recap

Field	Where
API Mode + API Key	Settings → Payhawk → Core
Bills Journal · Receipts Journal	Same panel
Group ID / Group Key (per company)	Same panel — set for each company you operate
Sync toggles	Settings → Payhawk → Miscellaneous
Document Type field	Payhawk → Settings → Expense fields
Reimbursement subjects	Settings → Payhawk → Configuration

For end-to-end setup, follow the seven-step Quick Start.

🔗 Where to read more

- Configuration check
- Sync options
- Bills vs receipts
- Reimbursements
- Troubleshooting

How to update

When a new release ships:

1. Request the new build from the v19 overview → Request the download. We reply within one business day with the package.
2. Stop the Payhawk cron in Odoo (Settings → Technical → Scheduled Actions)
3. Replace the module files on your Odoo addons path
4. **Apps → Update Apps List → Payhawk module → Upgrade**
5. Re-enable the cron

6. Run [Configuration check]

PLANA-hosted tenants: PLANA staff perform steps 2–5 during the agreed maintenance window.

Troubleshooting

Start at **Settings** → **Payhawk** → **Sync Log**. Entries are colour-coded; red = error, amber = warning, green = success.

"Payhawk API key not set!"

The API key for the current **API Mode** is empty. Check you entered the key for the *selected* mode (Sandbox vs Production keys are stored separately).

"Payhawk account not set!" / Account ID stays empty

The first sync could not reach `/accounts`. Usually a wrong API key or a network/firewall issue. Verify the key in Payhawk and that the Odoo server can reach `api.payhawk.com`.

"Could not create custom field odoo_internal_data in Payhawk"

You are on a **Payhawk group account** but the **Group ID** and **Group API key** fields in Odoo are blank. The connector uses the per-account API key to read expenses and the group key to create the `odoo_internal_data` custom field that links each expense back to its Odoo bill — without the group key, Payhawk returns 403 on every group-level write.

Fix:

1. In Payhawk, generate a Group API key — see Get an API key → Group (multi-entity) accounts.
2. In Odoo, **Settings** → **Payhawk** → **Group accounts**, paste the Group ID and Group Key for the active mode (Sandbox / Production are stored separately) — see Configuration → Group accounts.

> **Odoo 19 specific**
>
> Group ID and Group Key are **company-dependent** in v19 — make sure you're filling them for the active company (top-right company switcher). Each company in a multi-company database holds its own group settings.

3. Re-run the sync. The custom field is created on the next attempt.

If you are NOT on a group account, this error shouldn't appear — contact PLANA support with the `event_id` from the log entry.

"403 Client Error: Forbidden for url: https://api.payhawk.com/api/v3/accounts/.../custom-fields"

Same root cause as **"Could not create custom field odoo_internal_data"** above: the per-account API key cannot write at the group level. Follow the steps in that section to add the missing **Group ID** + **Group API key** and re-sync.

Any other `403 Forbidden` from `api.payhawk.com` that is **not** on `/custom-fields` is usually an expired or revoked API key — regenerate it in Payhawk and paste the new value into the active API Mode's slot in Odoo.

A supplier is skipped (warning, not error)

Common causes, all non-fatal:

- **Invalid email** — Payhawk rejects malformed addresses; the supplier is skipped and flagged. Fix the email in Odoo and re-sync.
- **Duplicate name** — a same-named supplier already exists in Payhawk; the connector links by name instead of creating a duplicate.

"Value with the same full label exists"

Two analytic accounts resolve to the same Payhawk label. The connector appends the Odoo record ID to disambiguate (e.g. `BP-General (4)`). No action needed, but you can rename the analytic account in Odoo to a unique label if you prefer clean names.

Currency errors when creating fund journals

A Payhawk fund account uses a currency that is missing or inactive in Odoo. Enable the currency (Accounting → Configuration → Currencies) and re-sync.

Webhooks not arriving

- Confirm **Enable Webhooks** is on (Miscellaneous tab).
- Confirm the URL registered in Payhawk matches `https://<domain>/payhawk/webhooks/<secret>` exactly, with the current secret.
- An unknown secret returns `401`; a disabled toggle returns a `disabled` status.
- Check the Odoo server logs for `Payhawk webhook data:` entries.

Nothing pulls / "No new expenses"

The connector pulls expenses created **after** the last *success* log entry. If you expect older expenses, edit the date on the latest success entry to an earlier time and re-run the sync (see First sync).

Error reporting to PLANA

Unless you tick **Disable Error Reporting**, error-level log entries are sent to PLANA Solutions (company name, user, module version, DB name, base URL, and the log entries) so we can troubleshoot and fix proactively. Disable it only if your policy forbids this.

Still stuck?

Note the **event_id** shown on the log entries (it groups one sync run) and contact PLANA support with it — it lets us find the exact run in our diagnostics.