# Humind module for PrestaShop

The official PrestaShop module of Humind, the AI shopping assistant. It synchronizes your catalog with Humind and adds the Humind widgets (chat, product questions, Idea Machine) to your storefront. Install it, connect it to your Humind account in one click, and your products, variants, prices, stock and categories flow to Humind on their own.

Full documentation: PrestaShop catalog sync and Install on PrestaShop.

## Requirements

- PrestaShop 1.7.8, 8.x or 9.x, PHP 7.1 or later with the curl extension.
- A shop served over HTTPS on a public domain (for the one-click connection and the widgets).
- A Humind account. You can create it during the connection.
- One currency: currency conversions are not synchronized. With several shops, select the shop in the selector at the top of the back office: a connection and a key belong to one shop, and each shop sends its own products, categories and CMS pages.

## Installation

1. Download the module: `https://embed.thehumind.com/prestashop/humind-latest.zip`.
2. In the back office, open **Modules > Module Manager**, click **Upload a module** and pick the zip.
3. Click **Configure** on the **Humind** module.

Installing the module changes nothing on your storefront until you connect it.

Coming from a former Humind synchronization module: installing this module switches it off. Click **Connect to Humind**: the key of the former module is not picked up. Then uninstall the former module. The cron and feed URLs change (they carry the module name): update them where they are scheduled.

## Connection

**Connect to Humind** (recommended): click the button, sign in to Humind or create your account, choose the company to link, confirm. You come back to the module (through an address of your shop: the address of your back office never leaves it), which shows "Connected to ". Humind created a live API key labeled "PrestaShop module" for this shop, allowed your shop domain for the widgets, and will call the cron and full sync URLs of the module itself (every minute and every night). The incremental synchronization starts at once.

During the connection Humind reads a proof on your shop to confirm you control it. If a firewall or a CDN blocks that call, the connection still succeeds and the synchronization works, but the module asks you to allow your domain yourself, with a link to the right page.

**A merchant who already sent a catalog through the API**: Humind keeps what this module writes apart from it, so the old products, collections and pages must be cleaned up before the first synchronization. Ask Humind to do it before connecting the module.

**API key entered by hand**: create a key in your Humind dashboard under Settings > Developer > Credentials with the `catalog:read`, `catalog:write`, `imports:write`, `knowledge:write`, `orders:write` and `customers:identity` scopes, paste it in the module and save. Then allow your shop domain on the allowed domains page yourself, and schedule the cron URL yourself.

**Disconnect** revokes the key in Humind (when it was created by the connection) and removes it from PrestaShop. Uninstalling the module does the same. Your products stay in Humind.

## Catalog synchronization

1. The first full synchronization starts by itself after connecting in one click, and a shop already synchronized starts nothing: every product a visitor of the shop can see is exported as NDJSON and imported in Humind asynchronously. Disabled products and products kept for some customer groups are left out, and Humind archives them: their content never leaves the shop. The module page shows its progress, and its failure if there is one. With a key entered by hand, click **Sync everything again** once. This button sends the catalog, then the collections when the import ends, and the CMS pages.
2. **Catalog changes** (advanced options) is on after connecting or saving a key. Product creations, updates, deletions and stock changes enter a queue in PrestaShop. Nothing is sent from the back office itself: the hooks only write to the queue, so saving a product stays fast whatever the state of the network.
3. The **cron URL** empties the queue: at most 100 products per call, sent in one batch request, failed products retried on the next call and logged in Advanced Parameters > Logs. Humind calls it every minute once the shop is connected in one click; otherwise call it from your own cron. The module shows the last run and warns after 5 minutes without one, which usually means a firewall blocks the calls from Humind:

```
* * * * * curl -s "CRON_URL_FROM_THE_MODULE" > /dev/null
```

4. The **full sync URL** re-exports the whole catalog (every night by Humind, or on demand). Both URLs carry a secret token: keep them confidential.

Categories become Humind collections, sent by the cron when a full sync ends and whenever a category or a product changes. A collection only holds the products Humind already has: when some are missing, the module sends it again 15 minutes later.

A **product feed** URL (token-protected CSV, one row per variant) is also available for merchants who prefer Humind to pull the catalog on a schedule (Settings > Catalog > Catalog import > Product feed in the dashboard).

### What is synchronized

| Humind | PrestaShop |
| --- | --- |
| Product external ID | Product ID |
| Title, description (HTML), summary (short description as text) | Product name, description, short description |
| URL slug and product URL | Link rewrite and product page |
| Categories | Category names (root and home skipped) |
| Images (cover first), brand | Product images, manufacturer |
| Translations | Same fields in every other active language |
| Status | `unlisted` when visibility is "nowhere", `active` otherwise. A disabled product, or one kept for some customer groups, is archived in Humind by its ID alone: its content is not sent |
| Metafields `prestashop.<feature>` | Product features, translated; several values become a list |
| Metafields `prestashop.*` | Tags, availability and delivery texts, condition, weight, shipping dimensions, unit price |
| Complementary products | Accessories |
| Variant external ID | `product-<id>` without combinations, `combination-<id>` otherwise |
| SKU, EAN-13, MPN | Combination values, else product values |
| Price, compare-at price, currency | Tax-inclusive shop price with reductions, regular price when reduced, default currency |
| Stock, availability | Available quantity (never below 0), backorder setting, available for order |
| Options and variant title, variant image | Combination attributes, first combination image |
| Add to cart | Native PrestaShop add-to-cart action (`cart_action` of type `prestashop`) |
| Collections `category-<id>` | Categories: title (prefixed by the parent when two share a name), description, URL, the products a visitor can see, translations, archived when disabled or deleted |

Language codes: `gb` and `us` become `en`, `tw` and `cn` become `zh`, `qc` becomes `fr`, `br` becomes `pt`, `mx` becomes `es`: the base code the widgets send. When two languages share a code, the default language wins, then the plain one.

Prices are those of the default country of the shop. With country detection on (`PS_DETECT_COUNTRY`), a visitor whose browser is in another country can see another price on the page than in the chat: for example 19.12 EUR on the page and 22.94 EUR in the chat, the cart charging 22.94 EUR.

## Orders, signed-in customers and CMS pages

- **Orders** (advanced options, on by default): when an order is validated, the module reads the `humind_attribution` cookie the chat widget sets on your domain and queues the order; status changes are queued too. The cron sends them to Humind (`POST /orders`, then `PATCH` with the status, label, tracking number and URL). Humind measures the sales the assistant contributed to, and order lines added to the cart from the chat are flagged. Nothing is sent during checkout itself. Orders stay queued while Humind refuses them (for example a key entered by hand without the `orders:write` scope): the module page then shows the error until the key is fixed.
- **Signed-in customers**: the chat widget calls `/module/humind/identity` on your shop. For a logged-in customer the module obtains a short-lived identity token from Humind server to server and hands it to the widget, so the customer can follow their orders in the chat. Anonymous visitors get an empty answer. The module sends Humind the customer id only, never the email, and the id never goes through the browser.
- **CMS pages** (advanced options, on by default): your CMS pages (delivery, returns, about...) become `webpage` entries of the Humind knowledge base, with their HTML and translations. Sent by the cron whenever a page changes. Disabled or deleted pages are archived.

## Storefront widgets

The page of the module sets where each block appears, among the places your theme really has. The module is also a PrestaShop widget: it can be placed under Design > Positions or with `{widget name="humind" block="questions"}`. A block appears once per page. Codes to paste in a CMS page, a description, a text module or a page builder: `[humind bloc="generateur"]`, `[humind bloc="questions"]` (the current product; outside a product page, nothing is shown).

Three switches on the configuration page, all driven by the catalog identifiers above:

- **Chat assistant**: `<humind-widget>` on every page (not on checkout), with the page type (`template-name`), the product on product pages (`product-id`), the current language and country. The look is configured in your Humind dashboard.
- **Product questions**: `<humind-product-questions>` under the add-to-cart block of product pages (hook `displayReassurance`).
- **Gift finder**: `<humind-gift-finder>` on the home page (hook `displayHome`), off by default.

The loader scripts are registered through the theme asset manager (deferred, bottom of the page). If your theme caches pages, clear the PrestaShop cache after changing a switch. On a `localhost` shop, enter your company ID in the module and allow localhost in the Humind dashboard for the widgets to load.

The Idea Machine and the product questions open the chat: they only appear while the chat assistant is on and attached to the page. Otherwise their buttons would lead nowhere, so the block is not rendered and a pasted code is removed.

Activating a theme can take modules off the hooks it lists, as the classic theme of PrestaShop 8 does on `displayHome` and `displayReassurance`. The module puts its blocks back right after, on that shop. A removal under Design > Positions on a hook the theme leaves alone stays. On a hook

it does clear, PrestaShop has already wiped it, along with its page exceptions, so the place chosen on the module page comes back: untick it there to remove it for good.

## Personal data

The module keeps, for each cart, the attribution seen by the browser (visitor identifier, products suggested), and the orders waiting to be sent. The GDPR module of PrestaShop (`psgdpr`) exports and erases these. Humind receives the orders (id, email, lines) for attribution and order tracking, and the customer id of a signed-in customer so that they can follow their orders in the chat; the conversations with the assistant are kept by Humind. To export or erase what Humind holds, contact Humind support.

## License

Academic Free License 3.0, see LICENSE.