Fube

GUIDE

Connect Fube via API or MCP (ChatGPT, Claude, VS Code)



fube.es

Updated · 24 September 2026

> Fube stores your customers, your invoices and your expenses, but your business rarely lives in Fube alone: you have a website that collects contacts, a script that reconciles the figures at the end of the month or an AI assistant you already work with all day long. Connecting those pieces to Fube saves you retyping data by hand and making copy-and-paste mistakes. There are two ways to do it, and this guide explains which one suits you and how to set it up: the **API key**, for your own apps and scripts, and the **MCP server with OAuth**, to link ChatGPT, Claude or VS Code without ever giving them your password.

Before you start: API key or MCP?

They are not the same thing, and choosing correctly will save you trouble:

- **API key.** This is a fixed credential —a long token— that you generate and paste into your own tool: your website, your ERP, a script. It gives **full access to the account**, so only use it with tools that you control. It is the direct route for automating without intermediaries.
- **MCP server with OAuth.** This is the recommended route when what connects is an **AI assistant or a third-party app** (ChatGPT, Claude Desktop, the VS Code extension). Instead of a fixed token, the tool receives a temporary permission that you explicitly authorise and that expires on its own; your password is never sent.

The quick rule: if you write the code yourself, use an API key. If you are connecting an AI or a third-party app, use MCP with OAuth. Both options are in **Settings › Advanced** and **both come with API access, which is part of the Elite plan**: API keys and also the OAuth clients that assistants use to connect. If your plan does not include it, instead of the screen you will see a card prompting you to upgrade your plan. You can find it on the **pricing** page.

Part 1 — Create an API key

Step 1 — Generate the key

Go to **Settings › Advanced › API keys** and click **Create API key**. If your plan is not Elite, this is where you will find the upgrade card instead of the list. With the right plan, **Create new API key** opens with these fields:

- A descriptive **name** (for example, 'Corporate website' or 'Invoicing script'). Good practice is one key per integration: that way, if one is no longer needed, you revoke it without touching the rest.
- An **expiry date**, which is **mandatory**: later than today and, at most, within a year. From that day on, the key no longer authenticates. You cannot create a key that never expires, because a key with no date is a door left open forever: make a note of when it expires so you can create the next one beforehand.
- **Permissions** (optional). If you choose some, the key will only be able to do that, and never more than you. Left empty, it can do everything you can, so it is best to restrict it to what the integration needs.
- **Allowed IPs** (optional). If your integration always runs from the same server, enter its addresses or ranges, one per line, and the key will not work from anywhere else.
- A **verification code**. Click **Send code** and type in the six-digit code sent to your account's email address; if you have **two-step verification** with an app, you can choose **Authenticator app** and use the one from your app. An API key acts on your behalf without a password, so Fube does not let you create one just by having a session open.

Step 2 — Copy it: it is only shown once

When you create it, Fube generates a key in the format `fube_` followed by a long string (64 hexadecimal characters). **That key is shown only once, at that moment.** Fube stores only a hash, so it cannot show it to you again: copy it and keep it in a secrets manager before closing the dialogue. If you lose it, it cannot be recovered; you would have to create a new one and update the integration.

Step 3 — Use it in your requests

Your tool sends the key with every HTTP request using the `X-API-Key` header. An example with `curl` would be:

```
curl https://fube.es/api/invoices \
  -H "X-API-Key: tu_clave_api_aqui"
```

From then on, your app can query, create or sync on your behalf programmatically: it can do the same as you could in the application, within the permissions you have given the key.

Step 4 — Monitor and revoke

The list shows each key with its **name**, how long it is valid for ('Valid until' and the date, or **Expired** in red if it has already passed) and its **last use** ('Never' if it has not been used). Last use is your radar: it reveals dormant keys that nobody has called for months and that are worth cleaning up. In its menu, **View information** adds the creation date and the status. When a key is no longer used or you suspect it has been leaked, open its menu and click **Delete API Key**. Deletion is immediate and irreversible: any integration that used it stops working instantly.

You can find the details of this feature in **API and keys**.

Part 2 — Connecting via MCP with OAuth

Fube's **MCP server** lets an external assistant call specific tools in your account — looking up customers and invoices, creating documents — through an open standard. Authorisation uses **OAuth 2.1 with PKCE**: you approve access from Fube and the tool receives a temporary permission, never your password.

Step 1 — Create an OAuth client

Go to **Settings › Advanced › OAuth clients** and click **Create OAuth client**. The form has these fields:

- **Client name.** A name to recognise it by, such as 'Claude Desktop' or 'VS Code'.
- **Redirect URIs.** The URLs the tool can redirect to after authorising. Each tool tells you its own (for example, ChatGPT uses its own redirect URL, which you copy from its settings). You can add several.

- **Scopes.** The identity permissions the client requests. By default it is set to `openid profile email`, and in practice that is what you need: these are the identity **OIDC scopes**; leave the default value unless the tool asks you for something else.
- **Permissions** (optional). Restrict what the tool can reach: if you choose permissions, it will only be able to do what they allow, and never more than the user who connects it. Left empty, it can do the same as that user.
- **Allowed IPs** (optional). If the tool always connects from the same addresses, restrict it to them. Left empty, from anywhere.

When you save it, Fube generates a **Client ID**: this is what you will paste into the tool, and it is not secret. If you also see a **Client Secret** when creating it, you do not need it: Fube does not use it, because the connection is protected with PKCE: when collecting the permission, the tool proves that it is the same one that requested it.

Step 2 — Copy the MCP server URL

On the same screen, Fube shows you your account's **MCP server URL** and step-by-step instructions for each tool, in three tabs: **ChatGPT**, **Claude Desktop** and **Visual Studio Code**. The URL ends in `/sse` on your Fube domain. That URL, together with the Client ID, is all you need to configure the tool.

Step 3 — Configure your assistant

With the Client ID and the MCP server URL to hand, go to the tool:

- **ChatGPT.** In Settings › Connectors (with developer mode enabled), create an app with the MCP server URL, OAuth authentication and your Client ID; leave the client secret field empty. Remember to have added to the OAuth client the Redirect URI that ChatGPT gives you.
- **Claude Desktop.** Add the Fube MCP server to your `mcpServers` configuration, pointing to the server URL. The Fube screen provides the exact block ready to copy.
- **Visual Studio Code.** Install an extension that supports MCP and configure the server with the Fube URL.

In every case, the specific instructions and the text to copy are on the OAuth clients screen itself, so you do not have to remember them.

Step 4 — Authorise from Fube

The first time the tool connects, it takes you to Fube so that you can **approve access** (if you are not logged in, it asks you to sign in first). You decide: the assistant does not get in until you confirm.

Fube opens the **Connect 'client name' to your account** screen, which tells you three things before you accept: which application is asking to connect, what it will be able to do — the same as you or, if the client has selected permissions, only those — and which address you will return to when you confirm. **If you did not start this connection yourself, do not continue: click Cancel.**

To confirm, the screen asks for a **verification code**: click **Send code** and type in the six-digit code sent to your account's email address. If you have **two-step verification** with an app, you can choose **Authenticator app** and use the one from your app. Click **Connect** and you return to the tool, now linked. It is the same thing you are asked for when creating an API key, and for the same reason: having your session open is not enough to obtain access that works on your behalf.

This is done **once per connection**: from then on, the tool renews its permission by itself, without asking you for the code again, until you disconnect it (step 5). If one day you remove it and connect it again, it will ask you again.

The connection belongs to the company in which the OAuth client was created, so sign in with your access for that company. If you sign in with another company's, the screen says **This application cannot be connected** and explains that it is not registered in the company you have signed in with: sign out, sign in with the correct company's access and connect again from the tool.

And it is worth properly sizing up what you are opening up to it, because this is not a little read-only window. Fube's MCP server exposes **nearly two hundred tools spread across fourteen areas** of the application — invoicing, customers and CRM, expenses, catalogue, company details, analytics and reports, treasury, projects, time tracking, settings, master data, knowledge base, AI agent and support — and they are not read-only: they also **create and modify** documents and records. Your assistant can read and write in practically every corner of your account.

That is precisely what makes it useful, but it is also the reason to connect only tools you trust, and the reason step 5 exists.

You can find the full walkthrough in **MCP and OAuth**.

Step 5 — Cut off access

The OAuth client is **immutable**: once created, it cannot be edited. From its menu you can only **View information** (name, Client ID, status, redirect URIs, scopes and date) or **Delete client**. If you need to change the name or the redirect URLs, the way to do it is to delete the client and create a new one.

Deleting the client cuts off access for anything already connected through it. On its next request, the tool finds that its permissions are no longer valid and cannot renew them: it is disconnected. If you want to use it again, create a new client and connect it again.

There is another way to disconnect, which acts on **your** account rather than on a client: in **My profile › Security**, the **Devices and access** card has the **Sign out of other sessions** button. It signs you out of all your browsers except the one you are using and **also disconnects every application you have connected** with your user (ChatGPT, Claude, VS Code…), from any client: they can no longer renew their permission, and the one they hold at that moment expires within a quarter of an hour at most. You confirm it with the same code from your email or your app, and the applications will have to connect again. Removing a single browser from that list, on the other hand, does not affect the applications.

If you suspect someone has connected a tool they shouldn't have, delete it this way and sign out of other sessions. And if you also think they have your password, **change it**: that closes all sessions and revokes all renewal permissions for that user, including those of any application connected with it. It is a heavy-handed measure — that person will have to sign in again everywhere — which is why it is best to authorise integrations with a user set up for that purpose rather than the one you use every day.

Edge cases and common errors

- **I lost the API key.** It cannot be recovered: Fube only stores a hash and shows it just once. Generate a new key and update the integration. An OAuth client's Client ID cannot be lost: it is not secret and you can always see it in **View information**.
- **I want to ‘edit’ an OAuth client.** You can't: it is immutable. Delete it and create another with the correct details.

- **I'm looking for the 'Revoke' option in OAuth.** There is no separate button because none is needed: **deleting the client is the revocation**, and it cuts off the tools connected through it on their next request. And to disconnect everything you have connected in one go, use **Sign out of other sessions**. Both are covered in step 5.
- **When connecting, Fube says 'This application cannot be connected'.** Read the reason given below it. If it says the application is not registered in your company, you have signed in with another company's access: sign out, sign in with the access for the OAuth client's company and connect again. If it says your plan does not include connecting assistants, that is the API access of the Elite plan. And if it says the request is not valid, go back to the tool and start the connection again.
- **I'm not receiving the code when connecting.** Check your spam folder; after 30 seconds you can click **Resend code**, and the last one you receive is the valid one. With app-based 2FA, you can also use the code from the app.
- **Instead of API keys or OAuth clients I see an upgrade card.** That is normal outside the Elite plan: both come with API access, which belongs to that plan.
- **My integration stopped working overnight.** Check the key in the list: if it says **Expired**, it has reached its expiry date and no longer authenticates. It cannot be extended: create a new one and replace it in the integration. To avoid being caught out, create the next one a few days before the previous one expires.
- **An old key says 'Never expires'.** These are keys created before expiry became mandatory. They still work, but it is best to replace them with one that has a date and delete them.
- **It won't let me set the date I want.** The expiry date must be after today and no more than one year ahead; the calendar does not allow you to choose outside that range.
- **I can't find the screens.** Both are under **Settings › Advanced**: 'API keys' and 'OAuth clients'. If your role does not have admin permissions, you may not see the option to create them.

Frequently asked questions

Does the API key have limited permissions? If you want it to, yes: when you create it you can choose its **permissions**, and then it can only do that (never more than you). If you don't choose any, it can do the same as you.

Can I restrict a key or a client by IP address? Yes. API keys and OAuth clients have an **Allowed IPs** field: if you fill it in, they only work from those addresses or ranges. And to cut off access, in both cases all you need to do is delete it: the key stops authenticating instantly and the OAuth client disconnects the tool on its next request.

Does my password reach the AI assistant? No. That is the point of OAuth: the tool receives a permission that you explicitly authorise, and your password never leaves Fube. That said, the permission being temporary does not mean it switches off by itself over time: it is renewed as long as the integration remains active. To withdraw it you have to do so manually, as explained in 'Cutting off access'.

Do the scopes control which data the AI can access? No: the OAuth client's scopes (`openid profile email`) are for **OIDC identity**. What limits what the assistant can reach is the client's **Permissions** field: with permissions selected, it can only do that; without them, it can reach the same as the user who connected it, with the full catalogue of tools on the MCP server.

Does this affect Verifactu or my tax obligations? Not directly. API and MCP are ways of connecting tools; Verifactu compliance is still guaranteed by Fube when issuing invoices, whoever creates them. An invoice generated by your assistant goes through the same compliant flow — signature, chaining and QR code — as one created by hand. If you like, review **getting started with Verifactu**.

Do I need to know how to code to use MCP? To connect ChatGPT, Claude Desktop or VS Code, no: the OAuth clients screen includes the instructions for each tool and it all comes down to copying and pasting the Client ID and the server URL. The API key, however, is intended for integrations you code yourself.

Connect Fube with your stack

You have two routes and neither asks you to share your password: the **API key** for your own apps — generate it, copy it once and use it with the `X-API-Key` header — and the **MCP server with OAuth** to link your trusted AI assistant without intermediaries. Both live in Settings › Advanced and both come with the Elite plan. And before connecting anything, be clear about the way out: deleting the OAuth client disconnects the tool on its next request. **Create your free account** and, with the Elite plan, connect your first tool in a few minutes. When you want to go deeper, you have the details of **API and keys** and **MCP and OAuth**.

Fube

START TODAY

Put this into practice with Fube

Create your free account and issue your first Verifactu-compliant invoice in minutes. No card, nothing to install.

fube.es

Invoicing and business management for freelancers and SMEs in Spain