Beds24 + ChatGPT

A self-serve setup guide for connecting Beds24 booking data to ChatGPT through a private, read-only Cloudflare Worker MCP server.

60-90 min	Read-only	Web setup
First-time setup	No booking writes	ChatGPT browser

What this PDF is for

This is the follow-up guide you can send to a nontechnical Beds24 operator after they read the web guide at axelrod.live/chatgpt-guides/beds24.

1. **Create a narrow Beds24 token**
   Use read scopes only. Keep guest personal data optional.
2. **Deploy the Worker starter**
   Cloudflare stores the Beds24 credential as a secret.
3. **Add the Worker to ChatGPT**
   Paste the Worker `/mcp` URL and the connector bearer token.
4. **Run the first access check**
   Ask: `Run the Beds24 access check.`

What you are building

You are not giving ChatGPT your Beds24 password, and you are not installing code inside Beds24. You are deploying a small Cloudflare Worker that speaks to Beds24 API V2 and exposes a few read-only tools to ChatGPT.

Read-only tools

No create, edit, cancel, or message booking tools are included.

Private secrets

Beds24 credentials stay in Cloudflare Worker secrets, not ChatGPT prompts.

Operator friendly

Most steps are copy, paste, login, scan tools, and test.

Plain-English data flow

When you ask ChatGPT about arrivals, ChatGPT calls your Worker. The Worker verifies ChatGPT with `CONNECTOR_BEARER_TOKEN`, calls Beds24 with `BEDS24_TOKEN` or `BEDS24_REFRESH_TOKEN`, then returns the result to ChatGPT.

What you need

Accounts and access

- ChatGPT web with custom app or developer mode access.
- A Cloudflare account that can deploy Workers.
- Beds24 access to create API V2 tokens under `Settings > Marketplace > API`.
- Permission to decide whether guest personal fields should be visible in ChatGPT.

Computer setup

- Node.js 22 or later for the Worker starter.
- A terminal app: macOS Terminal, Windows Terminal, iTerm, or similar.
- A password manager or secure note for temporary token handling.
- The Worker starter folder from the Axelrod guide.

Time estimate

Plan on about 60-90 minutes the first time. The setup is self-serve, but it is not a "no-code" or "30-minute" task because you must create a scoped Beds24 token, deploy a Worker, set Cloudflare secrets, and scan tools in ChatGPT.

Do this from a browser

Open the web versions of Beds24, Cloudflare, and ChatGPT. ChatGPT MCP apps are configured in ChatGPT web, not mobile.

Keep tokens out of chat

Do not paste the Beds24 token into ChatGPT. Paste it only into the Cloudflare Wrangler secret prompt.

ChatGPT setup and admin notes

The exact menu labels can shift as OpenAI updates custom apps, but the permission model matters: someone with the right workspace role may need to enable developer mode or create the custom app.

Business workspaces

- An admin or owner may be required to enable developer mode and create or publish the app.
- Each admin may need to enable developer mode for their own account.
- At launch, published Business apps may need to be recreated if the tool definitions change.

Enterprise and Edu

- Admins can grant developer mode access through workspace controls and RBAC.
- Admins can control which users can access a published custom app.
- New or changed tools may require admin refresh before users can call them.

What to ask your admin for

1. Permission to create a custom MCP app in ChatGPT.
2. Permission to use bearer/API key authentication for the app.
3. Permission to test the app privately before sharing it with the workspace.
4. Confirmation that only approved operators should access Beds24 booking data in ChatGPT.

Remote server requirement

ChatGPT connects to a remote MCP server URL. That is why this guide deploys a Cloudflare Worker and uses the Worker endpoint ending in `/mcp`.

Create the Beds24 API token

Create the Beds24 credential under `Settings > Marketplace > API`. If Beds24 API access is disabled, first allow API access under `Settings > Account > Account Access`.

☑	`read:properties` for property and room names	Required
☑	`read:inventory` for availability and room-calendar questions	Required
☑	`read:bookings` for booking dates, status, channel, and IDs	Required
☐	`read:bookings-personal` for guest names, emails, phones, and personal booking data	Optional
☐	`Allow linked properties` only if ChatGPT must read linked property accounts	Optional

Use

- A long-life token with read-only access, or
- A refresh-token setup if you intentionally want the Worker to refresh access tokens.

Avoid

- `write:*`, `delete:*`, and `all:*`.
- Especially avoid `all:bookings` for this starter.
- Do not add financial or payment scopes unless you have a clear policy reason.

Important limitation

Beds24 scopes are selected when the token or invite code is created. If a scope is missing later, create a new token with the correct read scopes rather than trying to edit the old one.

Understand the Worker starter

The starter is a small Node/Cloudflare project. You do not need to rewrite it for the standard read-only setup. You install it, set secrets, run the dry run, deploy, and copy the health page's `/mcp` URL.

File	What it does	Operator action
`README.md`	Contains the deploy commands, Beds24 scope reminder, ChatGPT setup steps, and troubleshooting notes.	Read before running commands.
`package.json`	Defines dependencies and scripts: `check` runs a Wrangler dry run; `deploy` deploys the Worker.	Do not edit for normal setup.
`wrangler.jsonc`	Names the Worker, points to `src/index.js`, sets compatibility date, and stores the Beds24 API base as a non-secret variable.	Optionally rename the Worker.
`src/index.js`	Implements the MCP server, auth check, health page, and read-only Beds24 tools.	Leave unchanged unless a developer is extending the app.

Tools exposed to ChatGPT

- `beds24_check_access`
- `beds24_get_properties`
- `beds24_get_bookings`
- `beds24_get_calendar`

Tools not included

- No create-booking tool.
- No edit/cancel booking tool.
- No guest-message sending tool.
- No payment or card-data tool.

Install the starter and log in

Download or copy the `worker-starter` folder from the Axelrod guide. Open a terminal in that folder, then run the commands below.

beds24-chatgpt-mcp-worker

```
# 1. Go into the Worker starter folder
$ cd worker-starter

# 2. Install dependencies
$ npm install

# 3. Log in to Cloudflare
$ npx wrangler login

# A browser opens. Finish the Cloudflare login there.
```

If the browser does not open

Wrangler usually prints a login URL in the terminal. Copy that URL into your browser, finish the Cloudflare login, then return to the terminal.

If `npm install` fails

Check Node.js first. The starter expects Node.js 22 or later. Install or switch Node versions, then run `npm install` again.

Do not paste secrets into commands yet

The next step uses `wrangler secret put`, which prompts you securely. Paste the token only when Wrangler asks for the secret value.

Set Cloudflare Worker secrets

Secrets are encrypted values attached to the Worker. ChatGPT never needs the Beds24 token. It only needs the separate connector bearer token.

set required secrets

```
# Use exactly one Beds24 credential type:
$ npx wrangler secret put BEDS24_TOKEN
# or:
$ npx wrangler secret put BEDS24_REFRESH_TOKEN

# Then set the private token ChatGPT will use:
$ npx wrangler secret put CONNECTOR_BEARER_TOKEN
```

BEDS24_TOKEN

Use this for a Beds24 long-life token. This is the simplest path for the read-only starter.

BEDS24_REFRESH_TOKEN

Use this only if you exchanged an invite code and want the Worker to refresh access tokens.

CONNECTOR_BEARER_TOKEN

Create a long random value. Save it in your password manager because you will paste the same value into ChatGPT when configuring bearer/API key authentication.

Normal Cloudflare behavior

Cloudflare Wrangler may create a new Worker version and show deploy-like output when you run `wrangler secret put`. That is expected.

Check, deploy, and copy the Worker URL

Run the dry run before deploying. It compiles the Worker and catches syntax or bundling problems without publishing a new live version.

dry run and deploy

```
# Compile and run Cloudflare deploy checks without live deploy
$ npm run check
... Total Upload: ... KiB / gzip: ... KiB

# Deploy the Worker to Cloudflare
$ npm run deploy
Deployed beds24-chatgpt-mcp
https://beds24-chatgpt-mcp.your-subdomain.workers.dev
```

What success looks like

- The dry run finishes without a red error stack.
- The deploy command prints a `workers.dev` URL or your custom Worker route.
- Opening the URL shows the health page.

If deploy fails

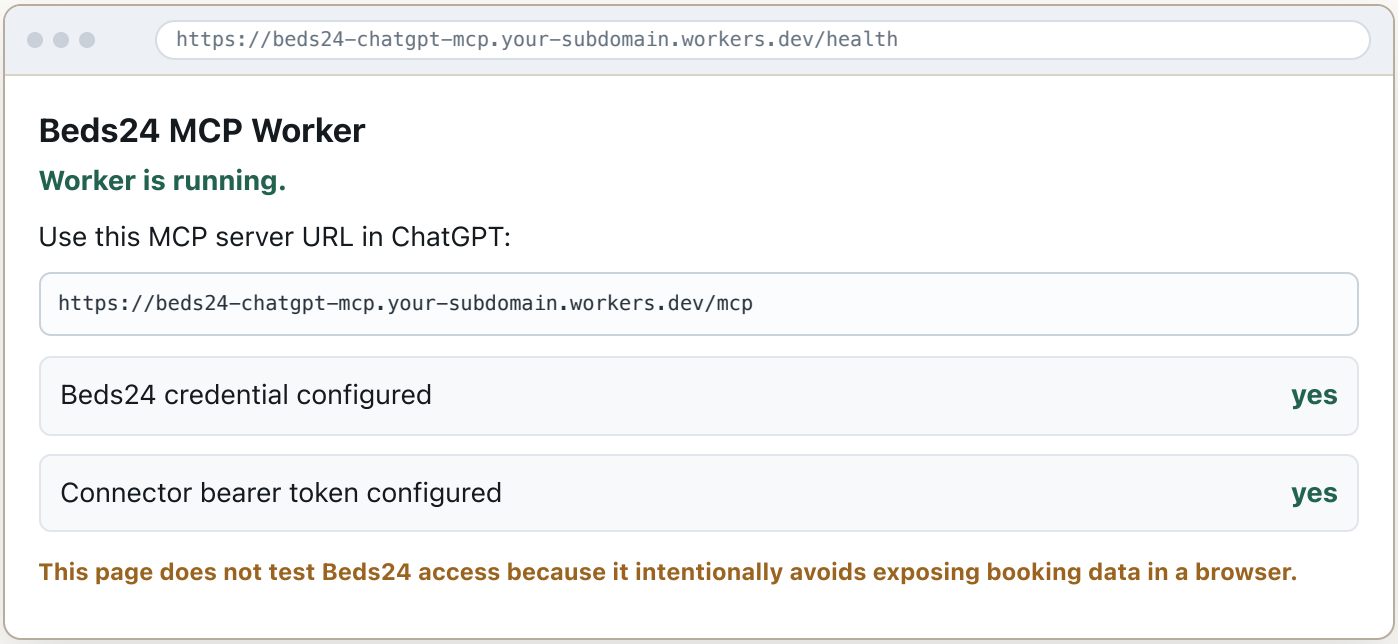
- Confirm `npx wrangler login` completed.
- Confirm the account can create Workers.
- Run `npm install` again if dependencies are missing.

Do not paste the plain Worker URL into ChatGPT

Open the Worker health page first and copy the MCP server URL shown there. It should end in `/mcp`.

Check the Worker health page

After deploy, open the Worker URL in a browser. The health page proves the Worker is running and shows the exact MCP endpoint to paste into ChatGPT.



Copy this

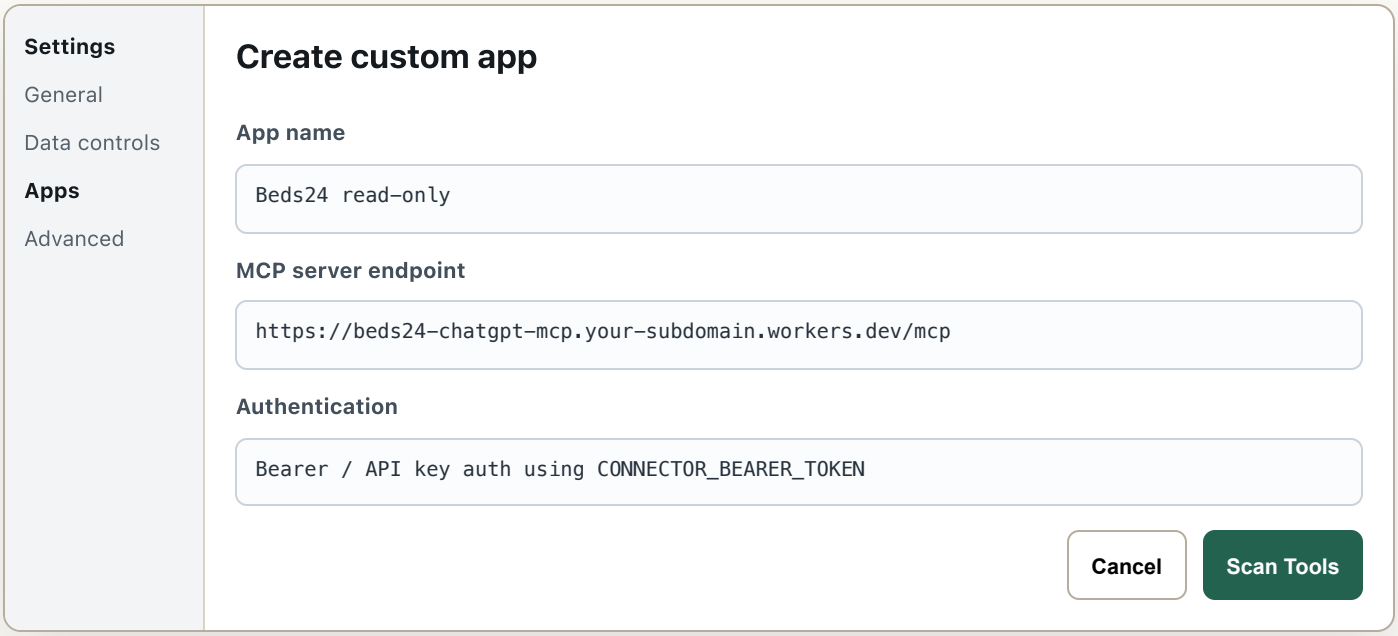
`https://...workers.dev/mcp`

Do not copy this

The root URL without `/mcp`. ChatGPT needs the MCP endpoint, not just the health page.

Add the Worker in ChatGPT

Open ChatGPT in a browser. Go to Settings, then Apps, and create a custom app. If the create option is hidden, ask your workspace admin to enable developer mode or create the app.



1. **Paste the `/mcp` URL**
   Use the URL from the Worker health page.

2. **Choose bearer/API key auth**
   Paste the value saved as `CONNECTOR_BEARER_TOKEN`.

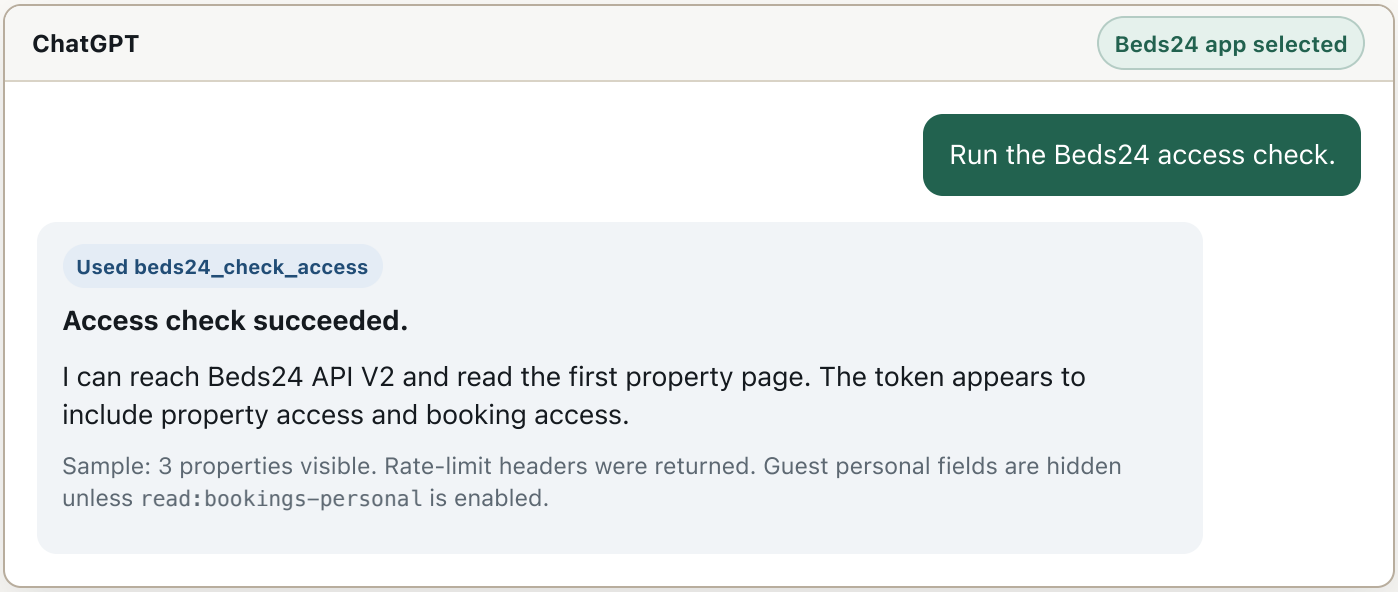
3. **Scan tools and create**
   Confirm ChatGPT sees the four Beds24 read-only tools, then create the app.

Run the first verification prompt

Start a new ChatGPT conversation, select the draft or custom Beds24 app, and send exactly this prompt first:

First prompt

```
Run the Beds24 access check.
```



If it succeeds

Move on to useful prompts. Start with a narrow date range such as today, this weekend, or the next 10 days.

If it fails

Use the troubleshooting table before widening token scopes. Most failures are wrong `/mcp` URL, mismatched bearer token, or missing Beds24 scopes.

Useful first prompts

These prompts keep date ranges small, make privacy expectations explicit, and help operators get practical answers without burning API credits on broad searches.

Daily operations

- "Show arrivals today by property. Include guest names only if the token allows it."
- "Which bookings check out tomorrow, grouped by property?"
- "Find same-day turnovers this weekend."
- "Summarize occupancy for the next 10 days by property."

Data cleanup

- "Find reservations with incomplete guest details for the next 14 days."
- "Which upcoming bookings are missing phone numbers?"
- "Show bookings created in the last 24 hours."
- "Which bookings changed since yesterday?"

Availability checks

- "Show room-calendar availability for the next 10 days."
- "Which properties have open nights this weekend?"
- "Find gaps shorter than three nights next week."
- "Give me a table of available rooms by date."

Access and safety

- "Run the Beds24 access check and tell me which scopes appear missing."
- "Tell me whether guest personal fields are visible."
- "List the Beds24 tools you can use and what each one reads."
- "Use only read-only Beds24 tools for this answer."

Prompting habit

Prefer specific windows such as "today", "this weekend", "next 10 days", or "created in the last 24 hours". Broad prompts across every property and every booking can hit Beds24 API credit limits.

Troubleshooting table

Symptom	Likely cause	Fix
ChatGPT cannot scan tools.	The endpoint is wrong, the Worker is not deployed, or custom app access is not enabled.	Confirm the URL ends in `/mcp`, open the health page, and ask a ChatGPT admin to enable developer mode if the app creation UI is missing.
Worker says `Unauthorized`.	ChatGPT did not send the right bearer token.	Paste the exact `CONNECTOR_BEARER_TOKEN` value into ChatGPT's bearer/API key auth field.
Health page says Beds24 credential is not configured.	The Beds24 secret was not added to Cloudflare.	Run `npx wrangler secret put BEDS24_TOKEN` or `BEDS24_REFRESH_TOKEN`, not both.
Access check gets a Beds24 scope or permission error.	The token was created without a needed read scope.	Create a new token or invite code with `read:properties`, `read:inventory`, and `read:bookings`.
Guest names, emails, or phone numbers are missing.	The token does not include personal booking data scope.	Create a new token with `read:bookings-personal` only if guest personal fields should be visible in ChatGPT.

First rule

Do not add broad write or all scopes to "fix" read errors. Match the missing read scope to the read-only use case.

More troubleshooting

Symptom	Likely cause	Fix
Linked-property bookings are missing.	The Beds24 token does not include linked properties.	Create a new token and enable "Allow linked properties" if the operator expects linked account data.
Arrivals look one day off.	Beds24 booking filters use the property's time zone.	Check property time-zone settings and ask ChatGPT to show dates with property names and local context.
Token expired or stopped working.	Invite code, generated token, refresh token, or long-life token age/usage rules may apply.	Create a new token if needed. Store the credential in Cloudflare only, then rerun access check.
Rate-limit or credit-limit errors.	The prompt requested too much data or repeated broad date ranges.	Narrow the date range, ask one operational question at a time, and wait for the five-minute window to reset.
`npm install` or dry run fails.	Old Node version, partial install, or dependency/network problem.	Use Node.js 22+, delete partial install only if needed, rerun `npm install`, then `npm run check`.

What the health page does test

It confirms the Worker runs and whether the required Cloudflare secrets exist.

What the health page does not test

It does not call Beds24 or expose booking data in a browser. Use ChatGPT's access-check prompt for that.

Security and revocation

Keep

- Beds24 credential in Cloudflare Worker secrets.
- Connector bearer token in ChatGPT app auth and a password manager.
- Read-only tool definitions in the starter.
- Access limited to operators who need booking data.

Avoid

- Pasting Beds24 credentials into ChatGPT messages.
- Adding write, delete, financial, or all scopes to the read-only starter.
- Sharing the Worker URL and bearer token in email threads or public docs.
- Letting the app remain connected after the operator no longer needs it.

How to revoke access

1. Disconnect or disable the custom app in ChatGPT.
2. Delete or rotate `CONNECTOR_BEARER_TOKEN` in Cloudflare.
3. Delete or rotate the Beds24 API token in Beds24.
4. If necessary, delete the Cloudflare Worker deployment.

Privacy decision

If operators do not need guest names, emails, or phone numbers in ChatGPT, leave `read:bookings-personal` off. ChatGPT can still answer many occupancy, arrival, departure, and availability questions from non-personal booking fields.

Final checklist

☐	Beds24 API access is allowed under account access settings.	Beds24
☐	Token was created under `Settings > Marketplace > API` with read scopes only.	Beds24
☐	Linked properties enabled only if needed.	Beds24
☐	`npm install` completed in the Worker starter folder.	Local
☐	`BEDS24_TOKEN` or `BEDS24_REFRESH_TOKEN` is set as a Cloudflare secret.	Cloudflare
☐	`CONNECTOR_BEARER_TOKEN` is set as a Cloudflare secret and saved securely.	Cloudflare
☐	`npm run check` and `npm run deploy` completed.	Cloudflare
☐	Health page shows the `/mcp` URL and both secrets configured.	Worker
☐	ChatGPT custom app scanned the Worker tools successfully.	ChatGPT
☐	First prompt succeeded: `Run the Beds24 access check.`	ChatGPT

Handoff note

Record the Worker URL, who owns the Cloudflare account, who can manage the ChatGPT app, and where the operator can rotate the Beds24 token. Do not record raw tokens in the handoff note.

Official source links

These sources were re-checked for the June 5, 2026 version of this PDF. Use the live documents if product screens or permissions change.

OPENAI	Developer mode and MCP apps in ChatGPT
OPENAI	Building MCP servers for ChatGPT Apps and API integrations
BEDS24	Beds24 API V2 wiki
BEDS24	Beds24 API V2 Swagger / OpenAPI UI
BEDS24	Beds24 API access and usage notes
CLOUDFLARE	Wrangler general commands and login
CLOUDFLARE	Wrangler Worker deploy and secret commands
CLOUDFLARE	Worker secrets

Axelrod guide and starter

GUIDE	Live Beds24 guide
STARTER	Worker starter README
STARTER	Worker package.json
STARTER	Worker wrangler.jsonc
STARTER	Worker source code