YOUR OWN DM AUTOMATION

BUILD YOUR OWN “MANYCHAT”

A step-by-step guide to building your own branded Instagram keyword reply tool with Codex and Cloudflare.

01 COMMENT	Someone asks for your guide.
02 MATCH	Your saved keyword triggers.
03 DELIVER	Your link arrives in a DM.

Includes an installable Codex skill, starter source code and a practical launch checklist. Independent guide; not affiliated with ManyChat or Meta.

01 What you are building

A small tool with a clear job

The outcome

A person comments a keyword on a selected Reel. Your server receives the event, matches a saved rule and sends one private reply containing your link. An incoming-DM rule can respond to a keyword separately.

The pieces

Instagram delivers events. A Cloudflare Worker verifies and processes them. D1 stores rules and delivery records. A private dashboard lets you edit replies and pause sending. Codex helps build and maintain the code.

What is included

The ZIP contains this guide, a Codex skill and a sanitized starter with dashboard, webhook handler, database migrations and tests. It contains no AO account credentials. This is an independent ManyChat-style tool, not ManyChat software or an affiliation.

Cost reality

Preset replies require no AI API calls. Workers Free currently allows 100,000 requests per day; D1 has separate usage limits. Workers Paid starts at $5 per month plus applicable usage. A workers.dev address avoids buying a domain. Codex access and your maintenance time are separate. Check sources 3 and 4 before launch.

Scope

This version serves one connected Instagram account. It has no visual funnel builder, full inbox, email capture, billing, automatic token refresh or multi-client onboarding. Expect Meta account setup and possibly review; approval time is not guaranteed.

02 Install the Codex skill

Give Codex the repeatable build process

Step 1 Extract the download

Unzip the package. Keep the build-instagram-dm-automation folder intact; its top level must contain SKILL.md, references and assets. Do not install the outer ZIP folder as the skill.

Step 2 Install for your project

Ask Codex to install that folder into the current project skill directory. Current documentation uses .agents/skills/ at the project root. Some existing desktop setups also expose ~/.codex/skills; have Codex verify the installed version before choosing a personal location. Source 5.

Step 3 Start a fresh task

Reopen the project or start a new task if the skill is not discovered. Invoke $build-instagram-dm-automation explicitly. Installation supplies instructions and code, not a connected Instagram account.

Copy this request

Use $build-instagram-dm-automation to build my branded Instagram reply tool. My professional account is @YOUR_HANDLE. When someone comments PROMPT on my selected Reel, send my free-download link. Use Cloudflare and the included starter. Keep sending paused until we test with my second account. Help me complete Meta setup.

Before you begin

Have a Creator or Business Instagram account, access to Meta for Developers, a Cloudflare account, your download URL and a second account for testing. Use your own accounts and project values throughout.

03 Prepare and host the app

Create the dashboard and database first

Step 4 Copy the starter

Ask Codex to copy assets/starter into a new project. Install its dependencies with Node and npm install, or use the included pnpm lockfile with pnpm. Run npm test before changing account settings.

Step 5 Create D1

Run npx wrangler login, then npx wrangler d1 create your-dm-tool. Put the returned database_id into wrangler.jsonc. Confirm the Worker name and database_name belong to your new project.

Step 6 Apply migrations

Run npx wrangler d1 migrations apply your-dm-tool --remote. Both rule rows and the global switch begin disabled. Replace example.com links, @your_handle, page branding and reply copy in the starter.

Step 7 Add private dashboard access

Generate a strong random access key locally and store it with npx wrangler secret put ADMIN_PASSWORD. Save the key in a password manager. The frontend must never contain it. Keep SEND_ENABLED set to false while configuring the connection.

Step 8 Publish the shell

Run npx wrangler deploy. Record the returned Worker URL. Open /direct and sign in; /health should report paused. Publish accurate /privacy, /terms and /data-deletion pages. The starter returns setup-required responses until these are replaced.

DEPLOY CHECK

npm test > npx wrangler deploy > Open /direct

04 Connect Instagram through Meta

Use the Instagram Login route consistently

Step 9 Create your app

Open Meta for Developers and create or configure an app with the Instagram API use case and Instagram Login. Choose the business portfolio if required. The account must be professional. This route does not require a linked Facebook Page; Facebook Login is a different route. Sources 1 and 2.

Step 10 Check access and roles

For a role-based test, add the appropriate app/Instagram tester and accept its invitation while signed into the intended account. Confirm the account handle at every authorization screen. If the popup fails in an embedded browser, use a supported regular browser and allow that site’s authorization popup.

Step 11 Request the right scopes

Use instagram_business_basic, instagram_business_manage_messages and instagram_business_manage_comments for this workflow. Confirm which permissions and access level Meta requires for your use. Publishing the app is separate from permission approval; other account owners can require additional review.

Step 12 Save the connection

Complete Instagram authorization and obtain the professional account ID and access token through the official setup flow. Put the account ID into IG_ACCOUNT_ID. Store the token with npx wrangler secret put IG_ACCESS_TOKEN. Verify the API version currently supported by your app before using GRAPH_VERSION.

Checkpoint

The account is authorized with the required capabilities. A token exists only in server secrets. Do not copy an Instagram password into code. Record the token expiration and renewal method; the included starter does not refresh tokens automatically.

05 Wire the webhook

Let Instagram notify your server

Step 13 Set verification secrets

Generate a separate random VERIFY_TOKEN and save it with npx wrangler secret put VERIFY_TOKEN. Obtain the app secret associated with the Instagram webhook setup and save META_APP_SECRET the same way. The verify token is for the initial handshake; the app secret checks event signatures.

Step 14 Register the callback

Deploy the updated configuration. In Meta’s Instagram webhook configuration, enter https://YOUR-WORKER.workers.dev/webhook and the matching verify token. The GET challenge must succeed. Never disable signature verification to make this step pass.

Step 15 Subscribe the account

Subscribe the required messages and comments fields, and enable the connected Instagram account’s webhook subscription. Field subscription and account subscription are distinct. Use the setup shown for your app and confirm the result.

Step 16 Verify real events

Send a fresh DM from the test account while sending is paused. Inspect limited server logs or add sanitized test instrumentation to confirm a signed incoming event for the correct account. Also test a fresh comment. Meta’s sample payload is only a transport check.

How delivery works

Incoming event > verify signature > confirm account and age > match enabled rule > atomically claim event > send once > record status. The starter handles processing in the request; add a durable queue before workloads require longer processing.

06 Build your first rule

One keyword and one clear response

Step 17 Select the trigger

Open /direct and create or edit a comment rule. Enter PROMPT as the keyword and write the exact reply, including an HTTPS download link that works without an admin login. A separate incoming-DM rule is optional.

Step 18 Resolve the post ID

Give Codex the published Reel URL. The authenticated GET /api/media-lookup?shortcode=YOUR_SHORTCODE endpoint searches the connected account’s media and returns its API id and permalink. Paste that returned id into Instagram media ID. It is not the URL shortcode or a page-source number.

Step 19 Restrict and preview

Keep a media ID on the first comment rule so it watches only your selected post. A blank ID applies the comment rule across the account. Preview with “PROMPT please!” and a nonmatching sentence. Preview runs locally and never sends a message.

Example reply

Here is the free guide you requested: https://YOUR-SITE.com/free-kit. Enjoy building your next project! Replace the URL before saving. Clear, useful replies are easier to test than long sales sequences.

Two switches matter

A rule must be enabled and the dashboard’s global automation switch must be on. The deployment variable SEND_ENABLED is a third availability gate: change it to true only for the authorized test or launch, redeploy, then use the dashboard Start control.

07 Test before opening the funnel

Prove receipt on a real account

Step 20 Run a controlled test

Agree on the second account, test post, keyword and exact reply. Enable only the intended rule. Set the runtime gate to true, deploy and press Start. From the second account, add a NEW matching comment to the chosen Reel.

Step 21 Confirm both sides

Check Recent deliveries for sent, then inspect the recipient’s Instagram inbox and Message Requests. Open the delivered download link. A sent status means the API accepted a message with an ID; recipient confirmation is a separate test.

Step 22 Check boundaries

A nonmatching comment should not reply. A comment on another post should not match a post-specific rule. Pause automation and confirm a new matching comment sends nothing. Unit tests should prove replaying the same webhook does not create a second send.

Step 23 Understand timing

This starter accepts DM events up to 24 hours old and comment notifications up to one hour old; those are implementation filters, not a complete statement of Meta policy. It does not backfill old comments. Check current private-reply and messaging-window rules before extending it.

Record the result

Keep a short test record: date, trigger, expected reply, delivery status, recipient confirmation and link result. Real DM success does not prove comment-to-DM success. If the comment test fails, leave that rule paused until its access or delivery issue is resolved.

08 Launch and maintain it

Keep the service understandable

Step 24 Launch the approved rule

After the real test passes, enable the intended post and keyword. Tell viewers what they will receive. Save the exact live rule details and show the operator where to pause automation. Do not enable unrelated rules by default.

Check delivery health

Review failures and unknown outcomes after launch. A failed result contains an API error code; an unknown result may already have reached the recipient. The starter deliberately does not automatically retry unknown sends. Investigate before manually resending.

Maintain your connection

Track token expiry, review account permissions and check Cloudflare usage. Back up rule configuration before major changes. Rotate a compromised dashboard key; existing signed sessions then fail validation. Avoid putting credentials in exported backups.

Handle data deliberately

The starter stores event IDs, rule IDs, timestamps, status and error codes. It processes message text transiently and has no automatic record retention job. Publish truthful contact and deletion instructions, then implement the process you promise.

Expand when needed

Add a durable processing queue, monitored renewal, per-recipient cooldown, post picker and stronger admin identity before expanding usage. Multi-client operation needs separate account credentials, access isolation and Meta onboarding/review. Optional AI replies add model costs and require their own testing.

09 Fix the common blockers

Follow evidence through the system

No authorization window

Use a regular supported browser, verify the intended account is signed in, and check site popup permissions. Accept the tester invitation if the account is in a test flow. A role listing alone does not prove authorization completed.

Webhook verification fails

Check the exact HTTPS /webhook URL and matching VERIFY_TOKEN. For a signed POST returning 403, confirm raw-body verification and the app secret for the configured Instagram webhook. Do not substitute a secret based only on similar app names.

DM works but comments do not

Check comment permission/access, comments field subscription, account subscription and media ID. Use a fresh comment on owned media. Do not infer comment capability from a successful text-message send.

Rule saves but nothing sends

Check runtime SEND_ENABLED, the global Start switch, the rule checkbox, trigger type, keyword and selected post. The first matching rule in ID order wins. The same event will not send twice; use a fresh authorized test event.

Dashboard loops or returns an error

Keep assets.html_handling set to none with this starter’s /direct rewrite. For login failures, check the private key and its 15-minute attempt throttle. Never put a Worker secret into public JavaScript to fix authentication.

Unknown or failed delivery

Inspect the sanitized error code and current Meta documentation. Check token validity and messaging eligibility. Do not continuously resend to “see if it works”; a timeout can happen after Meta accepted the message.

10 Your handoff checklist

A repeatable build you can reuse

Before you call it complete

Confirm the correct professional account, working public policies, required permissions, verified webhook, account subscription, authenticated dashboard, correct media ID, working download URL and passing duplicate-event tests.

Before you call it live

Confirm the test recipient actually received the intended comment reply, the pause switch works, tokens have an owner and renewal plan, and usage/retention expectations are documented. A dashboard preview is not a live test.

What the new owner receives

Provide the deployment URL, project source, configuration instructions, private credential location, test record, maintenance responsibilities and known limitations. Do not package a running operator’s secrets or delivery data.

Sources and further reading

1. Meta Instagram Login setup and permissions.
2. Meta’s official Instagram API collection and messaging examples.
3. Cloudflare Workers pricing and free-tier limits.
4. Cloudflare D1 pricing.
5. OpenAI Codex skill structure and installation.
6. Meta private replies reference. Links below are clickable.

Version note

Prepared September 13, 2026. The starter is based on the AO Direct implementation. Permission screens, API versions and pricing can change. Meta’s developer site limited automated access during preparation; its official API collection was accessible. Recheck current private-reply requirements before launch.

1 Meta Instagram Login

2 Meta API collection

3 Workers pricing

4 D1 pricing

5 Codex skills

6 Private replies