

Stop Feeding Your Secrets to ChatGPT

<https://scottslab.io/posts/scrambler-anonymize-data-llms>



TL;DR

Pasting a log full of SSNs into ChatGPT is a compliance problem, not a shortcut. Scrambler swaps PII for realistic fake data before it leaves your browser, then unmask the AI's reply on the way back: regex matching, no accounts, nothing uploaded. Text mode runs entirely client-side; PDF mode auto-redacts and hands back a clean document.

 Try it now: scramble.scottslab.io. No signup required.

Stop Feeding Your Secrets to ChatGPT

We've all done it.

You're debugging a customer issue at 11pm. You've got a log file full of names, emails, SSNs, and that one guy's phone number who keeps calling about his account. You need AI help. So you...

...paste the whole thing into ChatGPT.

Congratulations. You just sent someone's social security number to OpenAI's training data. HR would like a word.

The Problem

Public LLMs are incredible tools. But they have a dirty secret: **your data might stick around.** Training data. Logging. That intern at OpenAI who's definitely not reading your prompts (they are).

HIPAA doesn't care that you "really needed help with that regex." Neither does your compliance officer. Or whoever's about to lawyer up.

The Solution: Scrambler

I built [Scrambler](#) because I was tired of manually find-replacing "Acme Corp" with "REDACTED" like some kind of caveman.

Scrambler has two modes:

- **Text Mode:** Paste text, mask PII, copy to AI, unmask response (100% browser-side)
 - **PDF Mode:** Upload PDFs, auto-redact PII, download clean document
-



Text Masking (Browser-Side)

Here's the deal:

1. Paste your sensitive text
2. Click "Mask"
3. It auto-detects PII and replaces it with fake data
4. Copy the safe version to any AI
5. Paste the AI's response back
6. Click "Unmask" - originals restored

That's it. No accounts. No logins. No data leaving your browser.

What It Catches

The tool uses regex pattern matching (no AI, ironically) to detect:

- **SSNs** - 123-45-6789 → XXX-XX-1000
- **Emails** - john.smith@acme.com → marlow.quintrell@bexley-harrow.example
- **Phone numbers** - (317) 555-1234 → (555) 100-1000
- **IP addresses** - 192.168.1.50 → 192.0.2.1
- **Dates of birth** - DOB: 03/15/1985 → DOB: XX/XX/1950
- **Medical record numbers** - MRN: 12345678 → MRN-100000
- **Credit card numbers** - 4532-1234-5678-9012 → XXXX-XXXX-XXXX-1000

- **Driver's license** - DL# A1234567 → DL# DL-100000
- **Account numbers** - Account: 9876543 → ACCT-100000

Yeah, the fake values don't move much on their own. It returns the same name and the same account number every time you mask. That's on purpose, and it's not laziness. Read on.

Why the Fake Names Look Like That

The obvious move is the names you'd expect: Contoso, Fabrikam, Alex, Smith, private IPs like 10.0.45.12. Standard placeholder stuff. There's a failure mode hiding in that choice, though, and I didn't love it once I traced it through.

You paste the masked version into an LLM. The LLM writes a response. That response might, on its own, contain the word "Contoso": it's a famous Microsoft sample company, and language models say famous sample company names sometimes, for no reason related to you at all. Now unmask that response. Scrambler finds "Contoso" in the reply, assumes it's the placeholder it planted, and swaps in your real company name. Silently. At the very last step. The tool built to hide your data just leaked it, and there'd be nothing in the output to tell you.

So the replacement values changed. Names now come from pools of invented first names, surnames, and companies that don't exist anywhere (think "Marlow Quintrell" and "Bexley Harrow Ltd") paired with email domains on .example, which is a domain reserved by spec for exactly this and will never resolve to a real site. IP addresses use 192.0.2.x, the documentation range set aside for the same reason. All of it is chosen to read like a normal name or address to a model, while being close to impossible for that model to produce on its own by coincidence.

One more piece worth knowing about: when you paste the AI's response back in and click Unmask, Scrambler tells you exactly how many values it found and restored, and lists any it couldn't locate. If the model reformatted a value, or a collision genuinely happens, you'll see it. The tool won't quietly get it wrong.

Deciding What Gets Masked

Names are the hard case. The tool can't automatically know that "John Smith" is a person and "Main Street" is not (well, it could, but that would require... AI. The irony is not lost on me).

So there's a configuration panel for this:

- **Per-type toggles:** flip SSN, email, phone, IP, DOB, MRN, account, credit card, and driver's license detection on or off individually.
- **Always mask:** a list of your own terms, such as names, company names, codenames, internal hostnames, and project names. Type one in, pick whether it should read back as a Name or a

Company, click Add. Nothing here is pattern-matched: it's an exact list, because no pattern is ever going to catch "Project Nightshade."

- **Never mask:** terms that should stay exactly as-is no matter what. This wins over everything else, including your own always-mask list, in case the two ever conflict.
- **Presets:** *Everything* turns every pattern on. *Technical documentation* drops DOB and IP addresses, because a config file is full of IPs you actually want to keep readable. *Medical records* turns everything on and puts extra emphasis on DOB and MRN.

One deliberate choice: your Always-mask and Never-mask lists are **not** saved to your browser by default. Those lists usually contain the exact names and codenames you're trying to keep quiet, and auto-saving them would just create a second, unprotected copy of the secret sitting in your browser's storage. There's a "Remember my custom terms" checkbox if you want them to persist across visits. It's off unless you turn it on. The toggles and your chosen preset save automatically either way, since there's nothing sensitive about knowing you like the Medical Records preset.

PDF Redaction

Sometimes you have an entire document that needs to be sanitized before sharing. Maybe it's:

- A medical record you need to send to a consultant
- A police report for a case study
- Financial statements for an audit
- Any document with scattered PII

Here's how it works:

1. Click the "**PDF Redact**" tab
2. **Choose your redaction style:**
 - **[REDACTED]** : Clean text labels
 - **[_____]** : A redaction bar, if you want it to look like the real thing
3. **Drag & drop your PDF** (or click to upload)
4. **Review what was found:** See exactly what PII was detected
5. **Download the clean PDF:** All PII replaced, document structure preserved

How It's Different From Text Mode

- **Server-side processing:** PDFs are too complex for browser-only
- **No disk, ever:** the file never touches a hard drive at any point, not even briefly
- **No storage:** no database, no logs, no traces
- **Layout preserved:** headers, paragraphs, structure stays intact

- **Not identical detection:** the two engines are separate code and don't catch exactly the same things (below)

What Gets Redacted

Mostly the same patterns as text mode: SSNs, emails, phone numbers, IP addresses, credit cards, DOB (with context like "Date of Birth:"), medical record numbers, and account/patient IDs. One gap worth knowing: driver's license numbers are only caught in Text Mode. The PDF engine doesn't have that pattern yet. Don't assume the two modes are interchangeable. Audit whichever one you're trusting.

Example

Original PDF text:

PATIENT INTAKE FORM

Patient: John Smith

SSN: 123-45-6789

DOB: 03/15/1985

Phone: 317-555-8421

Email: john.smith@acmecorp.com

After redaction (text style):

PATIENT INTAKE FORM

Patient: John Smith

SSN: [REDACTED]

[DOB REDACTED]

Phone: [REDACTED]

Email: [REDACTED]

Note the date line: the PDF side swallows the "DOB:" label along with the date, where the browser side leaves it standing. Same category, two engines, two behaviours.

Notice "John Smith" is still sitting right there. That's not a bug. It's the same limitation as text mode. A name isn't a pattern the PDF engine can search for on its own. If you're redacting documents that always have the same patient or client name in them, add that name to the always-mask list before you upload, and it'll get caught with everything else.

The Privacy Part (It's Actually Private)

Text Mode

Everything runs in your browser.

- No server processing
- No data transmitted anywhere
- No accounts or cookies
- No logging

Open DevTools. Watch the Network tab. Nothing gets sent. Your HIPAA officer can sleep soundly.

PDF Mode

Ephemeral server processing, and I mean actually ephemeral.

- The file is held in memory only: read into RAM, streamed to the redaction process over its input and output pipes, streamed back. It is never written to a disk anywhere in the pipeline.
- No permanent storage
- No database entries
- Max 5 concurrent sessions (DoS protection)
- 10MB file size limit

The PDF has to hit a server (PDFs are too complex for a browser to redact on its own), but "hit a server" doesn't mean "gets saved." It means the bytes pass through memory and come back out the other side.

Scanned PDFs Are the One Place You Can Get Fooled

Here's the honest gap, and it matters more than any of the others: a scanned document is a picture of text, not text. There's no text layer for the pattern matching to search: it's pixels. So Scrambler finds nothing, because there's nothing there *to* find, and hands the file back looking processed.

It will tell you which pages it couldn't read. But a fully scanned document can come back reporting zero detections and still have every SSN in it sitting untouched in the image data. If a document might be scanned, don't trust a clean report. Check it yourself first.

What This Doesn't Catch

Pattern matching has edges, and I'd rather list them than have you find out the hard way:

- No IPv6 addresses
- No international or non-U.S.-style phone numbers

- No passport numbers or national ID numbers, from any country
- Day-first dates (31/12/1990) and standalone dates without a "DOB" or "born" nearby are missed. The tool only catches a birth date when it's flagged by a keyword
- The driver's license pattern is loose enough that it also grabs invoice numbers, ticket IDs, and part numbers that happen to look similar
- Any 16-digit number in groups of four gets flagged as a credit card, whether or not it actually is one. There's no real validation, just shape-matching

None of this is a reason not to use the tool. It's a reason to look at what it found before you hit send.

Open Source, So You Don't Have to Take My Word for It

The whole thing is on GitHub: [schlangens/scrambler](https://github.com/schlangens/scrambler), MIT licensed. Every claim above about what runs where and what gets stored is something you can go verify yourself instead of trusting a blog post.

There are 34 automated tests, all passing. Two of them do the paranoid thing directly: one snapshots the filesystem before and after a PDF redaction run and asserts nothing new showed up anywhere, and another pulls the text back out of a redacted PDF and confirms the sensitive strings are actually gone: not covered by a black rectangle sitting on top of readable text underneath, which is a mistake other redaction tools have made.

Run It With Zero Network Access

There's also a single-file offline build: [scrambler-offline.html](https://schlangens.github.io/scrambler-offline.html). Save it, disconnect from the internet, open it straight from your own machine. It does text masking only. PDF redaction needs the server, so that part isn't in the offline build.

The build process refuses to produce this file if it contains any network call at all: a fetch, an XMLHttpRequest, a script tag or stylesheet pointing somewhere else, even a stray `http://` in a comment. And it's not just checked once: the automated pipeline regenerates the file on every change and fails the build if the committed copy has drifted from what the source actually produces. I downloaded the live file and checked it myself: zero occurrences of any of those.

Real World Example

You have (with "John Smith" already sitting in your always-mask list, because the patterns will never find him on their own):

Patient John Smith (DOB: 03/15/1985, SSN: 123-45-6789)
called from (317) 555-0199 regarding prescription refill.
Contact: john.smith@acme.com
Account: 98765432

Scrambler gives you:

Patient Marlow Quintrell (DOB: XX/XX/1950, SSN: XXX-XX-1000)
called from (555) 100-1000 regarding prescription refill.
Contact: marlow.quintrell@bexley-harrow.example
ACCT-100000

Note the account line: the pattern eats the "Account:" label along with the number, because the label is what tells it the digits are an account number in the first place.

Paste that into ChatGPT. Ask your question. Get your answer.

Paste the answer back into Scrambler. Click Unmask. "Marlow Quintrell" becomes "John Smith" again, and you get a count of exactly what came back.

When You Need This

- Healthcare workers using AI for documentation help
- Support teams debugging customer issues
- Developers with production logs
- Legal teams sanitizing documents for case studies
- Anyone handling financial data
- Literally anyone who values not getting fired

When You Don't Need This

- Your grocery list (unless you're buying... suspicious groceries?)
- That fanfic you're writing (no judgment)
- Anything already public

Try It

scramble.scottslab.io

No signup. No payment. No tracking. Just paste and go.

Built with JavaScript, paranoia, and an unreasonable number of regex patterns.

Written by Scott S. — Contact: scott@scottschlangen.com — scottslab.io — <https://scottslab.io/posts/scrambler-anonymize-data-llms>