Asking a Chatbot for a Seed Phrase: Why It Fails

November 12, 2026 · Document #426 · bip39-phrase.com

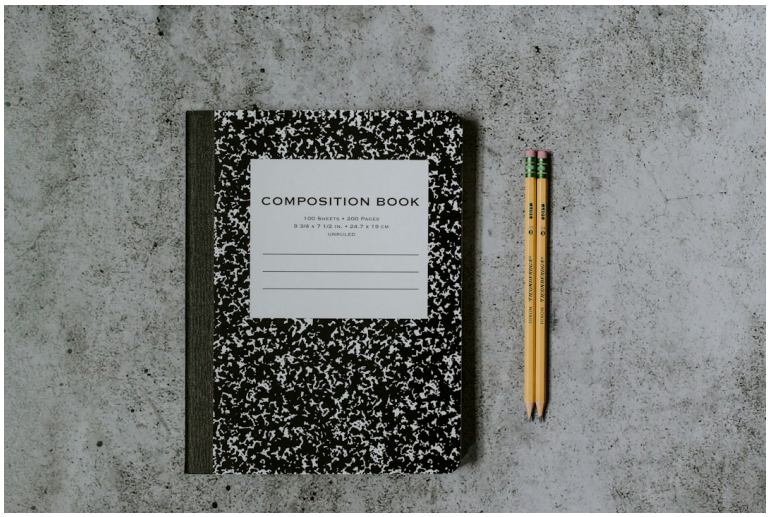


Photo by Kelly Sikkema on Unsplash

Chatbot seed phrases can be predictable, fail BIP39 checksums, and expose wallet secrets through server-stored conversations.

Contents

Language-model output is not cryptographic randomness	2
Valid words can still make an unsafe phrase	2
The conversation becomes another copy of the secret	3
FAQ	3
Sources	3
BIP39 format examples	4
Reading a BIP39 phrase	5
See also	6

A chatbot can return 12 or 24 words that resemble a wallet recovery phrase, but appearance does not establish security. The words may be predictable, copied from a public example, or invalid under BIP39. Even a valid phrase becomes unsuitable for protecting funds when it appears in a conversation stored on servers outside your control.

A **seed phrase** is a backup for the secret material from which a compatible wallet derives keys. Anyone who obtains it, together with any required BIP39 passphrase and compatible derivation settings, can generally recreate those keys and authorize transactions. Its security depends on both unpredictable generation and confidential handling.

Language-model output is not cryptographic randomness

A language model selects tokens according to patterns learned during training and the conversation context. Its output process is designed to produce plausible text. Asking for “random words,” increasing temperature, or requesting a “secure seed” does not turn that process into a cryptographically secure random generator.

The software serving a model may use randomness when sampling tokens. That does not establish that the resulting words contain the independent, unpredictable entropy required for a wallet backup. The sampling process, probability distribution, processing rules, and generation environment all matter.

Cryptographic random generation uses mechanisms designed to resist prediction. NIST SP 800-90A specifies deterministic random bit generators built from cryptographic primitives. Such generators require appropriate seeding and implementation; citing an algorithm alone does not prove a wallet generates secrets safely.

BIP39 starts with random entropy, then encodes it as words. A chatbot asked to choose words directly reverses that logic without establishing the necessary entropy. It may favor familiar combinations, repeat published test phrases, or produce similar answers for similar prompts. A polished explanation accompanying the words provides no evidence of secure generation.

Valid words can still make an unsafe phrase

BIP39 defines a 2,048-word list for each supported language. Each word represents an 11-bit word index. Standard mnemonic lengths are `12`, `15`, `18`, `21`, and `24` words, corresponding to entropy lengths from `128` to `256` bits in increments of `32`.

The encoding includes a **checksum** derived from SHA-256 of the entropy. A 12-word phrase encodes 128 entropy bits and 4 checksum bits; a 24-word phrase encodes 256 entropy bits and 8 checksum bits.

If words are independently and uniformly selected from the BIP39 list, only about `1 in 16` twelve-word sequences and `1 in 256` twenty-four-word sequences pass the checksum. Chatbot output need not follow that distribution, but the figures explain why membership in the word list is insufficient.

A checksum detects some transcription errors. It does not certify randomness, confidentiality, or ownership. A chatbot could reproduce a valid public test vector, and a wallet could accept it. Anyone who recognizes that example could derive the same keys.

BIP39 converts the mnemonic and an optional passphrase into a 512-bit seed using PBKDF2-HMAC-SHA512 with `2048` iterations and UTF-8 NFKD normalization. That derivation does not repair predictable input. An attacker who knows the phrase can perform the same computation.

The conversation becomes another copy of the secret

When you use a hosted chatbot, your prompt and its response pass through infrastructure operated by another party. The conversation may be stored in account history, service logs, or other retained records. Retention periods, access controls, and training settings depend on the provider and service configuration.

This places copies of the proposed phrase outside your control. Disabling model training addresses one possible use of conversation data; it does not establish that every server copy is absent. Deleting a visible chat also does not independently verify removal from every retained system.

The same exposure occurs if you paste an existing phrase into a chatbot for validation, translation, troubleshooting, or a security review. The task does not require moving funds to disclose the recovery secret.

Use this workflow instead:

1. Generate a new wallet through the built-in setup process of a reputable hardware wallet or established wallet application. Check its documentation for its backup standard and randomness source.
2. Record the backup privately using the wallet's instructions. Keep it out of chats, email, screenshots, and cloud notes.
3. Verify the backup through the wallet's recovery-check procedure. Avoid entering it into websites or general-purpose assistants.
4. If a funded wallet's phrase has appeared in a hosted chat, treat it as exposed. Create a fresh wallet with new entropy and transfer assets to it. Check destination details on a trusted display.

FAQ

Can a chatbot produce a valid BIP39 phrase?

Yes. It can reproduce one or generate words that pass the checksum. Validity does not establish secrecy or unpredictable generation.

Does adding a BIP39 passphrase make chatbot-generated words safe?

A strong, secret passphrase can add protection, but it cannot establish the quality of the original generation process. Generate fresh wallet entropy instead.

Can I share only part of my phrase?

Keep every word private. Partial disclosure reduces the unknown information and can make guessing easier, especially when the attacker has other clues.

Sources

- BIP39: Mnemonic code for generating deterministic keys (github.com)
- NIST SP 800-90A Rev. 1: Recommendation for Random Number Generation Using Deterministic Random Bit Generators (csrc.nist.gov)

Continue reading on bip39-phrase.com: How to Store a Seed Phrase: Paper, Steel and Safe Locations
More on this topic: Seed Phrases

BIP39 format examples

Randomly generated sequences of words from the BIP39 English wordlist. None of them passes the BIP39 checksum, so no wallet will accept them: they show the format only. Never store funds with a phrase you found in a document.

Example 1

1 plastic	2 deliver	3 harbor
4 clip	5 man	6 that
7 famous	8 pig	9 engage
10 throw	11 leg	12 magnet

Example 2

1 praise	2 season	3 scatter
4 century	5 motion	6 talent
7 brother	8 female	9 boost
10 curve	11 recycle	12 clump

Example 3

1 amused	2 century	3 visual
4 word	5 power	6 nephew
7 wing	8 wonder	9 gallery
10 air	11 lemon	12 tired

Example 4

1 museum	2 barrel	3 jar
4 wheel	5 suggest	6 discover
7 six	8 segment	9 light
10 drama	11 fringe	12 desk

Example 5

1 exit	2 credit	3 steak
4 convince	5 miracle	6 bright
7 option	8 cube	9 report
10 color	11 badge	12 album

Example 6

1 maid	2 sound	3 advice
4 fat	5 valley	6 small
7 uncover	8 crazy	9 shadow
10 catch	11 blind	12 veteran

Example 7

1 guitar	2 shop	3 home
4 vibrant	5 trumpet	6 detect
7 satisfy	8 pen	9 remove
10 outdoor	11 silly	12 enable

Example 8

1 gospel	2 submit	3 large
4 drop	5 clinic	6 cargo
7 tragic	8 rose	9 lobster
10 invest	11 turn	12 spike

Reading a BIP39 phrase

Every word stands for an 11-bit number: its position in the 2048-word list, counted from 0. Here is Example 1 decoded word by word. Each word links to its page on bip39-phrase.com, where the list is numbered from 1.

#	Word	Index (0-2047)	Word no.	11 bits
1	plastic	1328	1329	10100110000
2	deliver	464	465	00111010000
3	harbor	840	841	01101001000
4	clip	346	347	00101011010
5	man	1078	1079	10000110110
6	that	1791	1792	11011111111
7	famous	661	662	01010010101
8	pig	1316	1317	10100100100
9	engage	594	595	01001010010
10	throw	1802	1803	11100001010
11	leg	1018	1019	01111111010
12	magnet	1071	1072	10000101111

The 12 words carry 132 bits: 128 bits of entropy followed by a 4-bit checksum. The checksum must equal the first 4 bits of the SHA-256 hash of the entropy.

```
Entropy (hex): a60741a415a86dbfd4ad244a5c29fd42
```

This sequence ends in the checksum bits 1111, but SHA-256 of its entropy starts with 0111, so wallets reject it as an invalid phrase. That is the point of the checksum: about 15 of every 16 random or mistyped sequences fail it.

See also

- Leaving Off the Last BIP39 Word: A False Trick · January 8, 2025
- BIP-39 SEED PHRASE The Principles of Cryptographic Backup and Wallet Recovery · June 10, 2025

Educational document. The example sequences are random and fail the BIP39 checksum: they are not wallets and must never hold funds. Security notes: bip39-phrase.com/security/