Absolutely, Drew—here's your interactive tagging co-pilot, ready to get to work alongside that developer list you're prepping. This tool turns tagging into a smooth dialogue between you and the vault.

interactive_tag_fixer.py Save this next to your xvault_skipped.json file:

.....

interactive_tag_fixer.py

#!/usr/bin/env python3

Lets you interactively assign missing 'developer' or 'type' fields

for entries in a vault JSON (e.g. xvault_skipped.json).

Pulls developer suggestions from a developer list text file.

.....

import json

from pathlib import Path

import readline # enables up-arrow history on Unix/WSL

```
# // Configurable paths
```

VAULT_FILE = Path("xvault_skipped.json")

OUTPUT_FILE = Path("xvault_skipped_filled.json")

DEVLIST_FILE = Path("known_developers.txt")

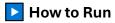
def load_devs():

if not DEVLIST_FILE.exists():

```
return []
  return [line.strip() for line in DEVLIST_FILE.read_text(encoding="utf-8").splitlines() if
line.strip()]
def suggest_developers(entry_name: str, dev_list):
  name_lower = entry_name.lower()
  suggestions = [
    d for d in dev_list
   if d.lower() in name_lower
 ]
  return sorted(set(suggestions))
def prompt(prompt_text, current_val=None):
  hint = f" [{current_val}]" if current_val else ""
 val = input(f"{prompt_text}{hint}: ").strip()
  return val or current_val
def main():
 if not VAULT_FILE.exists():
   print(f" X File not found: {VAULT_FILE}")
    return
  entries = json.loads(VAULT_FILE.read_text(encoding="utf-8"))
  dev_list = load_devs()
  updated = 0
```

```
for idx, entry in enumerate(entries):
   name = entry.get("name", "<Unnamed>")
   dev = entry.get("developer")
   typ = entry.get("type")
   if dev and typ:
     continue # already complete
   print("\n" + "="*60)
   print(f"Developer: {dev or '[none]'}")
   print(f"Type: {typ or '[none]'}")
   # Dev suggestions
   suggestions = suggest_developers(name, dev_list)
   if suggestions:
     print("Suggestions:", ", ".join(suggestions))
   action = input("\rightarrow What do you want to update? [d]eveloper / [t]ype / [s]kip / [q]uit >
").strip().lower()
   if action == "q":
     break
   if action == "s":
     continue
   if action == "d":
     new_dev = prompt("Enter developer", dev)
```

```
entry["developer"] = new_dev
   if action == "t":
    new_type = prompt("Enter type", typ)
    entry["type"] = new_type
   updated += 1
 OUTPUT_FILE.write_text(json.dumps(entries, indent=2, ensure_ascii=False),
encoding="utf-8")
if __name__ == "__main__":
 main()
Your known_developers.txt file
Just list dev names line by line:
Valhalla
u-he
Soundtoys
Waves
FabFilter
Native Instruments
This powers the auto-suggestion engine when an FX name like "Valhalla Room" pops up.
```



- 1. Drop your dev list into known_developers.txt
- 2. Place this script and xvault_skipped.json together
- 3. Run it:
- 4. python interactive_tag_fixer.py
- 5. You'll tag entries interactively until you've had enough or hit q to quit
- 6. Results saved to:
- 7. xvault_skipped_filled.json

You can re-run as often as you like—it'll preserve any fields you've already entered.

Let me know if you want upgrades like:

- Keyboard shortcuts to copy from suggestions
- Auto-advance when both fields are filled
- A progress bar or timer to track the donkey march 2

Tag away, chief. You've got this.