Cross-Listing Tracker & Auto-Delister Guide

1. Overview

This guide covers how to build a Python-based program to track listing status and automatically delist sold items across Depop, eBay, and Mercari. You will learn what APIs exist, how to authenticate, and how to organize your code structure.

Platform	API Access	Key Endpoints / Method	Notes
eBay	■ Official API	• GET /sell/inventory/v1/inventory_item • GET /sell/fulfillment/v1/order • DELETE /sell/inventory/v1/inventory_item/{sku}	Create a Developer Account, use OAuth
Depop	■■ Unofficial API	• /api/v2/user/{id}/products/ • /api/v1/products/{id}/	Use login token or automate via Seleniu
Mercari	■■ Unofficial API	• GET /v2/entities • DELETE /v2/entities/{id}	Use cookies, device ID, or headless aut

2. Base Python Project Structure

Recommended folder structure:

```
project_root/
■■■ main.py
■■■ ebay_api.py
■■■ depop_scraper.py
■■■ mercari_scraper.py
■■■ database.py
■■■ .env (API keys, tokens)
```

3. Example main.py (Skeleton)

```
from ebay_api import EbayAPI from depop_scraper import DepopScraper from
mercari_scraper import MercariScraper def main(): ebay = EbayAPI() depop =
DepopScraper() mercari = MercariScraper() sold_items = ebay.get_sold() +
depop.get_sold() + mercari.get_sold() for item in sold_items: if item.platform !=
'eBay': ebay.delist(item.sku) if item.platform != 'Depop': depop.delist(item.id) if
item.platform != 'Mercari': mercari.delist(item.id) if __name__ == "__main__": main()
```

4. Tips & Security

• Use environment variables for API keys and tokens.
• Add rate-limiting to avoid bans.
• Use SQLite or JSON for mapping listings.
• Automate with cron jobs or serverless functions.
• Secure your credentials with encryption or dotenv.