Linux-Windows API

Based on project and what we want (run smooth on **Windows dev** and **Linux server**), here are the **likely issues that could break cross-platform deployment**:

**Places Where Problems Will Happen**

**1. Chrome + Chrome driver Setup**

* On Windows you probably just installed Chrome and it works.
* On Linux servers:
  + Chrome may not exist (or version mismatch).
  + SessionNotCreatedException: user data directory is already in use → happens if multiple scrapers reuse same Chrome profile.
  + Missing flags like --no-sandbox or --disable-dev-shm-usage cause instant crash.  
    👉 This is **the #1 reason scrapers work on Windows but fail on Linux**.

**2. Hardcoded File Paths**

* Your cleaners & scrapers save CSVs into subfolders like:
* data/boat/
* data/boult/
* data/noise/
* If you wrote paths like "data\\boat\\products.csv" → it will **break on Linux**.
* Even relative paths (../) sometimes misbehave depending on where the Flask API is run.

👉 You need pathlib or os.path.join() everywhere.

**3. CSV Handling**

* Linux is case-sensitive → if one script writes Products.csv and another looks for products.csv, it fails.  
  👉 I noticed your cleaner scripts expect exact file names; mismatch will break on Linux.

**4. Parallel Scraping / Multiple Sessions**

* If your API triggers scrapers in parallel (/scrape while /scrape-next runs), Selenium may conflict:
  + One Chrome instance locks profile.
  + Causes "user data directory is already in use" error.

👉 Fix: use **temporary profiles** per run or kill stale Chrome processes before launch.

**5. Logging**

* You log to app.log.
* On Linux, if API runs as a different user, it may not have permission to write to that file.  
  👉 Make sure logs/ folder is writable.

**6. Process Management**

* On Windows you just python app.py.
* On Linux server, Flask won’t survive a restart/crash unless you wrap with:
  + gunicorn (production)
  + systemd or supervisord (auto restart)

👉 Without this, your API will run for a day and then mysteriously stop.

**app.py (Flask API)**

* ⚠ If you run python app.py on server, Flask will die when SSH closes.
* Must deploy with:

gunicorn -w 4 -b 0.0.0.0:5000 app:app

**7. Dependencies**

* Your requirements.txt may install fine on Windows but fail on Linux (e.g., lxml or selenium require extra system libs).  
  👉 Needs testing in Docker or server.

**8. NLTK Data (nltk\_data/)**

* You already included it inside preprocessor/. But:
  + If you use nltk.download("punkt"), it will try downloading into user’s home by default.
  + To fix, set NLTK\_DATA env variable in your app:

import nltk

import os

nltk.data.path.append(os.path.join(os.path.dirname(\_\_file\_\_), "nltk\_data"))

* This ensures your server (Linux or Windows) won’t fail due to missing punkt.

A screenshot of a computer

AI-generated content may be incorrect.

**Put on run for boat category\_2**

**All file came cleaned   
but reviews file didn’t came clean though scraped**

**Then we check log files to see the issue using**

**tail -n 50 app.log**

**and we find the issue**

C(env) root@srv922949:~/new\_dataextraction# tail -n 50 app.log

^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^

File "/root/new\_dataextraction/env/lib/python3.12/site-packages/pandas/core/algorithms.py", line 1743, in map\_array

return lib.map\_infer(values, mapper, convert=convert)

^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^

File "lib.pyx", line 2972, in pandas.\_libs.lib.map\_infer

File "/root/new\_dataextraction/preprocessor/cleaner\_reviews.py", line 57, in clean\_text

words = word\_tokenize(text)

^^^^^^^^^^^^^^^^^^^

File "/root/new\_dataextraction/env/lib/python3.12/site-packages/nltk/tokenize/\_\_init\_\_.py", line 142, in word\_tokenize

sentences = [text] if preserve\_line else sent\_tokenize(text, language)

^^^^^^^^^^^^^^^^^^^^^^^^^^^^^

File "/root/new\_dataextraction/env/lib/python3.12/site-packages/nltk/tokenize/\_\_init\_\_.py", line 119, in sent\_tokenize

tokenizer = \_get\_punkt\_tokenizer(language)

^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^

File "/root/new\_dataextraction/env/lib/python3.12/site-packages/nltk/tokenize/\_\_init\_\_.py", line 105, in \_get\_punkt\_tokenizer

return PunktTokenizer(language)

^^^^^^^^^^^^^^^^^^^^^^^^

File "/root/new\_dataextraction/env/lib/python3.12/site-packages/nltk/tokenize/punkt.py", line 1744, in \_\_init\_\_

self.load\_lang(lang)

File "/root/new\_dataextraction/env/lib/python3.12/site-packages/nltk/tokenize/punkt.py", line 1749, in load\_lang

lang\_dir = find(f"tokenizers/punkt\_tab/{lang}/")

^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^

File "/root/new\_dataextraction/env/lib/python3.12/site-packages/nltk/data.py", line 579, in find

raise LookupError(resource\_not\_found)

LookupError:

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Resource punkt\_tab not found.

Please use the NLTK Downloader to obtain the resource:

>>> import nltk

>>> nltk.download('punkt\_tab')

For more information see: https://www.nltk.org/data.html

Attempted to load tokenizers/punkt\_tab/english/

Searched in:

- '/root/nltk\_data'

- '/root/new\_dataextraction/env/nltk\_data'

- '/root/new\_dataextraction/env/share/nltk\_data'

- '/root/new\_dataextraction/env/lib/nltk\_data'

- '/usr/share/nltk\_data'

- '/usr/local/share/nltk\_data'

- '/usr/lib/nltk\_data'

- '/usr/local/lib/nltk\_data'

- '/root/new\_dataextraction/nltk\_data'

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

[2025-08-22 13:35:29,281] [INFO] [app.py:173] Updated last\_scraped for https://www.boat-lifestyle.com/collections/product-personalization

[2025-08-22 13:35:29,281] [INFO] [\_internal.py:97] 49.36.91.100 - - [22/Aug/2025 13:35:29] "POST /scrape-next HTTP/1.1" 200 -

(env) root@srv922949:~/new\_dataextraction#

**A screenshot of a computer

AI-generated content may be incorrect.**

**[2025-09-03 10:26:12,227] [INFO] [\_internal.py:97] WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.**

**\* Running on all addresses (0.0.0.0)**

**\* Running on http://127.0.0.1:5050**

**\* Running on http://69.62.78.167:5050**

**[2025-09-03 10:26:12,228] [INFO] [\_internal.py:97] Press CTRL+C to quit**

**[2025-09-03 10:26:12,229] [INFO] [\_internal.py:97] \* Restarting with stat**

**[2025-09-03 10:26:16,941] [WARNING] [\_internal.py:97] \* Debugger is active!**

**[2025-09-03 10:26:17,036] [INFO] [\_internal.py:97] \* Debugger PIN: 975-712-078**

**[2025-09-03 10:26:17,431] [INFO] [app.py:159] Dispatching next scrape: {'brand': 'noise', 'url': 'https://www.gonoise.com/collections/wireless-earbuds', 'category\_id': 'category\_1'}**

**[2025-09-03 10:26:17,436] [INFO] [app.py:104] Scrape request: brand=noise, category\_id=category\_1, url=https://www.gonoise.com/collections/wireless-earbuds**

**[2025-09-03 10:35:28,133] [INFO] [app.py:124] Scraping completed for noise**

**[2025-09-03 10:35:28,526] [INFO] [app.py:173] Updated last\_scraped for https://www.gonoise.com/collections/wireless-earbuds**

**[2025-09-03 10:35:28,527] [INFO] [\_internal.py:97] 49.36.88.17 - - [03/Sep/2025 10:35:28] "POST /scrape-next HTTP/1.1" 200 -**

**(env) root@srv922949:~/new\_dataextraction#**