Toro Scrape

Version 0.3.1

`toro_scrape.py` automates logging into Toro Identity/Shop, extracts product details and pricing via authenticated API calls, and saves results to CSV. It supports robust retries, automatic re-authentication, configurable logging, graceful interruption with partial saves/resume, and concurrent scraping for speed. An optional FTP upload can publish the final CSV.

What's new in 0.3.1

- Early skip of already-processed product numbers to prevent re-scraping and duplicate requests.
- Automatic re-authentication on 401/403 with a single-flight lock to avoid token races.
- Hardened HTTP backoff: honors `Retry-After` for 429, retries 5xx, and improves error logging.
- Small randomized delays between requests to reduce rate-limit triggers.
- Stable deduplication before both partial and final saves.
- Safer partial resume flow: merges `.partial` and existing output to avoid rework.

Features

- Automates login with Playwright and extracts a Bearer token for API calls.
- Scrapes detailed product/pricing/inventory fields and metadata.
- Robust HTTP with retry/backoff for transient errors.
- Concurrent scraping with a bounded thread pool (configurable).
- Periodic partial saves with automatic resume.
- Final output deduplicated and optionally uploaded to FTP.
- Headless mode supported.

Requirements

Minimum System Requirements

- **Operating System**: Windows 10/11 or modern Linux distributions (Ubuntu 20.04+, Fedora 32+, CentOS 8+, etc)
- **Python Version**: v3.10 or 3.11 recommended (3.8+ minimum)

Dependencies

Install dependencies:

```
pip install -r requirements.txt
python -m playwright install chromium
# Linux only:
# python -m playwright install-deps
```

Included libraries:

- `pandas`: CSV creation and manipulation
- `playwright`: Browser automation
- `requests`: HTTP requests

Running the packaged EXE (no Python required)

Package typically includes:

- ToroScraper.exe
- browses/(Playwright browsers; e.g., chromium_headless_shell-XXXX/...)
- config.txt

Configuration (`config.txt`)

Example config (JSON):

```
{
    "login_url": "https://identity.toro.com/as/authorization.oauth2?
response_type=code&client_id=InsiteCommerceClient&redirect_uri=https%3A%2F%2Fshop.
thetorocompany.com%2Fidentity%2Fexternalcallbackextension&scope=openid%20profile%2
0email%20address",
    "username": "your_username",
    "password": "your_password",
    "headless_mode": false,
    "max_rows": "all",
    "input_file": "input.csv",
    "output_file": "output.csv",
    "overwrite_existing": true,
    "rsv_qty": 1,
    "ftp_host": "",
    "ftp_port": 21,
    "ftp_username": "",
    "ftp_password": "",
    "ftp_directory": "/path/to/directory",
    "log_level": "INFO",
    "log_file": "logs/toro_scrape.log",
    "save_interval": 0,
    "concurrency": 6
}
```

Parameters:

- Authentication
  - login_url: The login URL for Toro Identity.
  - username, password: Credentials for login.
  - headless_mode: true for headless browser, false to show browser window.
- Workload
  - max_rows: "all" for all rows or an integer to limit the number processed.
  - input_file: CSV containing products to scrape.

    - rsv_qty: Quantity used when requesting pricing.
- Output
- output_file: Final CSV output filename.
- overwrite_existing: If false and output_file exists, a timestamp is appended on save.
- save_interval: 0 disables periodic saves; N > 0 saves every N processed products to output_file.partial (used for resuming).
- Concurrency
    - concurrency: Number of worker threads for parallel product processing.
- Logging:
    - log_level: DEBUG, INFO, WARNING, ERROR, CRITICAL (default INFO).
    - log_file: Optional path to write logs to a file in addition to console.
- FTP (optional):
    - ftp_host, ftp_port, ftp_username, ftp_password, ftp_directory
    - Leave ftp_host empty to skip FTP upload.

Running the Script

From source

- To run the script, execute the following command in the terminal:

```
python -m venv .venv
# Windows
.venv\Scripts\activate
# Linux/macOS
#source ./venv/bin/activate

pip install -r requirements.txt
python -m playwright install chromium
# Linux only:
# python -m playwright install-deps

python toro_scrape.py --config config.txt
# optional override concurrency at runtime:
# python toro_scrape.py --config.txt --concurrency 4
```

From EXE

- Double-click ToroScraper.exe
- Or from command line:
    - ToroScraper.exe
    - ToroScraper.exe --config "C:\path\config.txt"

Ensure:

- config.txt is correct and accessible (same folder as EXE unless an absolute path is provided).
- browsers/ folder (Playwright) exists and is not renamed.
- Input CSV exists.

Input CSV

By default, the project uses SolidCommerceProducts.csv format with fields like: LDSKU,Product Name,Date,Weight,UPC,Manufacturer,SKU,Model Number,ReleaseDate,CreateDate,MSRP,Qty Alternate Images,Product Image,KitInfo,HSCode,ScheduleBCode,HSDescription,CaptureSerialNumber,PackageInsuranceRequired,SignatureRequired,PackagingPreferences,SpecialProduct

- Requires a column named "SKU".
- Extracts product numbers from SKUs that start with "TOR~":
  - Takes text after "TOR~" up to next "~" or end of string.
  - Example: SKU: TOR41-6820SOMETHING → Product Number: 41-6820
- Non-Toro SKUs are ignored; empty/invalid SKUs are filtered out.

Output Fields

CSV columns include:
`product_number,product_id,material_id,item_status,unit_list_price,unit_regular_price,unit_net_price,actual_price,is_on_sale,unit_of_measure,distribution_centre,division,category_group,order_group,qty_on_hand,availability_message,available_date,short_description,erp_number,erp_description,large_image_url,shipping_length,shipping_width,shipping_height,shipping_weight,unit_of_measure_description,is_active,is_discontinued,can_back_order,track_inventory,minimum_order_qty,multiple_sale_qty,sku,upc_code,model_number,brand,product_line,tax_code1,tax_code2,tax_category,product_detail_url,is_special_order,is_gift_card,is_subscription,can_add_to_cart,can_add_to_wishlist,can_show_price,can_show_unit_of_measure,can_enter_quantity,requires_real_time_inventory,availability_message_type,meta_description,meta_keywords,page_title`

Notes: - Some fields may be empty depending on API responses. - Results are deduplicated by product_number before both partial and final saves.

Partial Save and Resume

- If save_interval > 0, the script writes a partial CSV at output_file.partial every N records and logs progress.
  - Loads it to continue where it left off.
  - Skips product_numbers already present in the partial file.
- On next run (with the same config), the script:
  - Loads .partial to continue where it left off.
  - Skips product_numbers found in .partial and previously completed output file (if present).
- On normal completion, final CSV is written and .partial is removed.

Concurrency

- Uses a ThreadPoolExecutor to process multiple product numbers concurrently.
- Configure via config.concurrency, or override with --concurrency.
- Thread-safe accumulation and early skip of already-scraped product_numbers prevent duplicates.

Graceful Interruption

- Press Ctrl+C to stop gracefully.
- The script will:
  - Stops submitting new tasks.
  - Saves partial results if save_interval > 0.
  - Exits with an interruption message.

Logging

- Configurable via log_level and log_file.
- Console and optional file logging (directory auto-created).
- Clear messages for retries, rate limits, and re-authentication events.

FTP Upload

- If ftp_host, ftp_username, and ftp_password are provided, uploads the final CSV to the configured directory.
- If overwrite_existing is false and output_file exists, a timestamped filename is used before upload.
- Missing directories on the server are created when possible.

Troubleshooting

- Login/Playwright:
  - Ensure browsers/ exists and matches Playwright's expected structure.
  - Set headless_mode: false to observe the login flow.
- Rate limits/Server errors:
  - Script automatically retries 429 (honors Retry-After) and 5xx with exponential backoff.
  - Consider lowering concurrency and keeping it steady for long runs.
- Re-authentication:
  - 401/403 responses trigger a single automatic re-auth attempt.
  - If you routinely see 403 but with "Customer Product Restriction," those SKUs are intentionally inaccessible.
- Partial resume:
  - Delete output_file.partial if you want a clean restart.
  - Ensure save_interval is a positive integer to enable partial saves.
- Duplicates:
  - The script deduplicates by product_number prior to saving partial/final CSVs.

License

This script is provided as-is. Use at your own risk.

Contact Information

- **Author**: Jim Tyranski
- **Email**: jim@tyranski.com Please include config details and error messages for faster support.

Please ensure to provide detailed information about the issue you're experiencing, including any relevant error messages and the configuration details used when running the script.

Changelog

- 0.1.0 - Initial Release
- 0.2.0 - Added FTP function; Playwright browser location notes
- 0.3.0 - Graceful Ctrl+C, config logging, centralized backoff, partial saves, concurrency, dedupe on save.
- 0.3.1 — Early duplicate skip, automatic re-auth on 401/403, improved backoff with 429 handling, jittered delays, safer resume and dedupe.