Automated 40 Keyword Rank Tracking: User Guide

Version 1.0

Welcome to your new Automated Keyword Rank Tracking tool! This guide will walk you through everything you need to know, from the initial technical setup to strategic daily use, all in simple, easy-to-understand language.

Note: This automation is built using Python and requires a one-time setup on your computer. Once configured, it will run automatically to save you hours of manual work.

Sheet Link: 40 Keywork Rank Tracking

Table of Contents

- Understanding the Log File
- Refreshing Your Chrome Profile

Chapter 4: Strategies & Customization

- How to Use the Data Strategically
- How to Customize the Automation
- Using Different Scraping Modes (Incognito & Mobile)

Chapter 5: How the Magic Happens (A Simple Look at the Script)

- What are Python and Selenium?
- The Journey of a Keyword: The Script's Logic

Appendix: Troubleshooting & Frequently Asked Questions (FAQ)

- Common Problems and Solutions

Chapter 1: Introduction to the Tool

What is this Automation Tool?

Imagine you have a list of 40 keywords and four competitors. To check your rankings, you would have to manually search each keyword on Google, scroll through pages of results, find each competitor's website, and note down their position. This is incredibly time-consuming and prone to errors.

This tool solves that problem.

It is a smart Python script that acts like a digital assistant. It automatically opens a web browser, searches Google for each of your keywords, finds your competitors' URLs in the search results, and records their exact ranking (e.g., 1, 5, 25, or "Not Found") directly into your Google Sheet.

It's designed to be robust and human-like, with built-in alerts for security checks (CAPTCHAs) and script errors, ensuring you are always in control.

Who is this Tool For?

This tool is designed for anyone who needs to track their website's search engine performance against competitors without spending hours on manual checks. You do not need to be a professional coder to set it up and use it.

It's perfect for:

- **SEO Specialists:** To monitor keyword positions and campaign effectiveness daily.
- **Digital Marketers:** To gather competitive intelligence and identify opportunities.
- **Marketing Managers:** To get a clear overview of the company's organic search visibility.
- **Small Business Owners:** To track their local SEO efforts and stay ahead of the competition.

Key Features at a Glance

- **Fully Automated Ranking Checks:** Just add your keywords and URLs to the Google Sheet, and the tool does the rest.
- **Direct Google Sheet Integration:** All results are automatically populated back into your worksheet, creating a centralized dashboard.

- **Human-like Scraping:** Uses random delays and other techniques to mimic human behavior, reducing the chance of being blocked.
- **Intelligent CAPTCHA Handling:** If Google presents a security check, the script pauses and sends you an email alert, allowing you to solve it manually and resume the process.
- **Email Crash Alerts:** If the script encounters a critical error, it immediately notifies you via email with technical details.
- **Persistent Login Session:** Uses a dedicated Chrome profile to maintain your Google login, further preventing security checks.
- **Configurable & Scalable:** Easily change settings like the number of keywords to check per run or the email recipients for alerts.

How It Works: A Simple Overview

The process is straightforward. Think of it as giving a set of instructions to a very efficient assistant.

1. **You Provide the Information:** You add your keywords and competitor URLs to the 'Ranking' sheet in your Google Sheet.
2. **You Configure the Settings:** In a simple configuration file, you tell the script where to find your files, your Google Sheet name, and who to email for alerts.
3. **The Automation Kicks In:** When you run the script, it "wakes up" and starts its work.
4. **It Reads the List:** The script connects to your Google Sheet and reads the list of keywords and URLs you want to track.
5. **It Searches Google:** For each keyword, it opens a real Chrome browser, navigates to Google, and performs the search, just like you would.
6. **It Scans the Results:** It carefully scans up to 5 pages of search results, looking for the competitor URLs you provided.
7. **It Records the Rank:** When it finds a URL, it notes its position and updates the corresponding cell in your Google Sheet. If it can't find a URL after 5 pages, it marks it as "Not Found".
8. **It Repeats:** The script repeats this cycle for every keyword in your batch, taking short breaks in between to appear natural.
9. **It Stays Alert:** If it runs into a CAPTCHA or an error, it immediately follows the rules you set in the configuration file (pause and email you).

This entire process happens automatically, giving you back valuable time to focus on strategy.

Chapter 2: Getting Started: The One-Time Setup Process

This chapter will guide you through the one-time setup of your automation tool. Follow these steps carefully to ensure everything works perfectly.

Prerequisites: What You Need Before You Start

1. **Python:** Ensure you have Python installed on your computer (Version 3.8 or newer is recommended). You can download it from python.org. During installation, make sure to check the box that says **"Add Python to PATH"**.
2. **Code Editor:** A code editor like Visual Studio Code is recommended for easily editing the configuration files.
3. **Project Files:** You should have a folder containing all the project files (`main.py`, `config.py`, `requirements.txt`, etc.).

Step 1: Setting Up Your Python Environment

To avoid conflicts with other projects, we will create a dedicated virtual environment.

1. Open your computer's command prompt (CMD) or terminal.
2. Navigate to your project folder using the `cd` command. For example:
   `cd C:\Users\YourName\Documents\40_Keywork_Rank_Tracking`
3. Create a virtual environment by running this command:
   `python -m venv venv`
4. Activate the environment.
   - **On Windows:** `venv\Scripts\activate`
   - **On Mac/Linux:** `source venv/bin/activate`
5. You will know it's active because you'll see `(venv)` at the beginning of your command prompt line. **You must activate this environment every time you want to run the script.**

Step 2: Installing the Necessary Libraries

With your virtual environment active, run the following command to install all the required Python packages automatically:

```
pip install -r requirements.txt
```

Step 3: Configuring the Google Cloud Platform (GCP) API

This step allows our script to securely access and update your Google Sheet.

1. **Create a GCP Project:**
    - Go to the Google Cloud Console.
    - Click the project dropdown at the top and click **"New Project"**.
    - Give it a name (e.g., "Keyword Ranking Automation") and click **"Create"**.
2. **Enable APIs:**
    - Make sure your new project is selected.
    - In the search bar, search for and enable the **"Google Drive API"**.
    - Search again and enable the **"Google Sheets API"**.
3. **Create a Service Account:**
    - In the search bar, go to **"Service Accounts"**.
    - Click **"+ Create Service Account"**.
    - Give it a name (e.g., "sheets-updater") and a description. Click **"Create and Continue"**.
    - For the role, select **"Project" > "Editor"**. Click **"Continue"**, then **"Done"**.
4. **Generate JSON Key:**
    - Find the service account you just created in the list. Click the three dots under "Actions" and select **"Manage keys"**.
    - Click **"Add Key" > "Create new key"**.
    - Choose **JSON** as the key type and click **"Create"**.
    - A `.json` file will be downloaded. **This file is your password—keep it safe!**
5. **Final Step:** Rename the downloaded file to `gcp_credentials.json` and move it into your main project folder.

Step 4: Setting Up Your Google Sheet

This is where you will manage your keywords and see the results.

1. **Make a Copy of the Sheet:**
    - Open the master Google Sheet template here: 40 Keywork Rank Tracking Sheet
    - In the menu, click **File > Make a copy**.
    - Give your copy a new name and save it to your own Google Drive.
2. **Share the Sheet with the Service Account:**
    - This is the most important step! Open your `gcp_credentials.json` file and find the `client_email` address (it looks like `...gserviceaccount.com`). Copy it.

    - In your new Google Sheet, click the **"Share"** button.
    - Paste the client email into the sharing box, give it **Editor** permissions, and click **"Send"**. If you skip this, the script will fail.
3. **Prepare Your Data:**
    - The sheet is already set up for you. You can use the provided `40 Keywork Rank Tracking - Ranking.csv` file as a sample to understand the structure.
    - **Keyword:** The search term you want to track.
    - **[Competitor] URL:** The exact URL you want the script to find for that competitor.
    - **[Competitor] Ranking:** Leave these columns blank. The script will fill them in.

Step 5: Configuring the Automation Script

Open the `config.py` file in your code editor and update the following settings.

- `PROJECT_ROOT`: The full path to your project folder. **Important:** Use forward slashes `/` or double backslashes `\\` in the path.
- `CHROME_PROFILE_PATH`: The script will create a dedicated Chrome profile here. The default path is usually fine.
- `SHEET_NAME`: The **exact name** of your copy of the Google Sheet.
- `WORKSHEET_NAME`: The name of the tab within your sheet (default is "Ranking").
- **Email Notification Settings:**
    - `ENABLE_EMAIL_NOTIFICATIONS`: Set to `True` to receive alerts.
    - `SENDER_EMAIL`: Your Gmail address that will send the alerts.
    - `SENDER_PASSWORD`: **Do not use your regular password.** You need to generate a Google "App Password".
        1. Go to your Google Account settings.
        2. Go to "Security" and enable 2-Step Verification.
        3. In the Security section, find and click on "App passwords".
        4. Select "Mail" for the app and "Windows Computer" for the device, then click "Generate".
        5. Copy the 16-character password and paste it here.
    -
    - `RECIPIENT_EMAIL`: A list of email addresses that should receive the alerts.

Step 6: Creating Your Master Chrome Profile

This one-time script creates a browser profile with your Google account logged in, which helps avoid CAPTCHAs.

1. Make sure your virtual environment is active.
2. Run the following command in your terminal:
   `python create_master_profile.py`
3. A new Chrome window will open. You have 90 seconds to:
   - Go to `google.com`.
   - Sign in to the Google account you use for tracking.
   - Accept any cookie pop-ups.
   - When you are done, **manually close the browser window**.

Step 7: Running the Automation for the First Time

Congratulations! You are now ready to run the automation.

1. Make sure your virtual environment is active.
2. Run the main script from your terminal:
   `python main.py`
3. A Chrome window will open and the script will begin processing your keywords. You can watch its progress in the terminal and see your Google Sheet update in real-time.

Chapter 3: Using the Automation Day-to-Day

Now that the setup is complete, using the tool is simple.

Your Daily & Weekly Workflow

1. **Update Your Keywords:** Open your Google Sheet and add, remove, or modify keywords and competitor URLs in the 'Ranking' sheet as needed.
2. **Run the Script:**
    - Open your terminal and navigate to the project folder.
    - Activate the virtual environment: `venv\Scripts\activate` (Windows) or `source venv/bin/activate` (Mac/Linux).
    - Execute the script: `python main.py`
3. **Check the Results:** Once the script finishes, open your Google Sheet to see the updated rankings.
4. **Review the Log:** For a detailed report of the run, open the `ranking_automation.log` file in the project folder.

How to Handle a CAPTCHA Alert

Occasionally, Google may present a CAPTCHA ("I'm not a robot" test). The script is designed to handle this gracefully.

1. **You'll Get an Email:** You will receive an email with the subject "ACTION REQUIRED: Ranking Scraper Paused by CAPTCHA".
2. **Find the Browser Window:** The script will be paused, but the Chrome window it opened will still be visible on your screen.
3. **Solve the Puzzle:** Simply solve the CAPTCHA in that browser window.
4. **The Script Resumes:** Once solved, the script will automatically detect it and continue its work. If you don't solve it within 15 minutes, the script will time out, skip that keyword, and move to the next one.

Understanding the Log File

The `ranking_automation.log` file is your eyes and ears. It records every major action the script takes, including:

- Which keyword is being processed.
- Which page of Google it is scraping.
- When it successfully finds a competitor's rank.
- Any warnings or errors it encounters.

This is the first place to look if something doesn't seem right.

Refreshing Your Chrome Profile

If you find you are getting logged out of Google or seeing more CAPTCHAs than usual, your Chrome profile session might be stale. You can easily refresh it.

1. Activate your virtual environment.
2. Run the refresh script: `python refresh_profile.py`
3. Follow the on-screen instructions to log in to Google again, just like you did in the initial setup.

Chapter 4: Strategies & Customization

This tool is powerful, but using it strategically will give you the best results.

How to Use the Data Strategically

- **Track Campaign Impact:** Run the script before and after launching a new SEO campaign or making website changes to measure the direct impact on your rankings.
- **Identify Competitor Movements:** Monitor competitor rankings to see when they launch new content or run campaigns. A sudden jump in their rank for a keyword is a signal to investigate.
- **Find Content Gaps:** If a competitor is consistently ranking for a keyword where you are not, it may indicate a content gap on your site that you need to address.
- **Prioritize Your Efforts:** Focus your SEO efforts on keywords where you are on page 2 (ranks 11-20), as these often require the least effort to push onto the first page.

How to Customize the Automation

You can easily tweak the script's behavior by editing the `config.py` file.

- **Change Batch Size:** The `KEYWORDS_PER_BATCH` setting controls how many keywords the script processes in a single run. Lower this number if you are running it very frequently, or increase it for a more comprehensive weekly check.
- **Adjust CAPTCHA Wait Time:** The `CAPTCHA_WAIT_TIMEOUT` (in seconds) determines how long the script waits for you to solve a CAPTCHA. Increase it if you need more time.
- **Add More Competitors:**
    1. Add new columns to your Google Sheet (e.g., "CompetitorX URL" and "CompetitorX Ranking").
    2. This is an advanced change that requires modifying the `main.py` script to include the new competitor data.

Using Different Scraping Modes (Incognito & Mobile)

The project includes two additional scripts for specialized use cases:

- **:** Runs the scraper in Chrome's Incognito mode. This provides a "clean slate" for each run, ignoring any previous cookies or search history, which can give a more neutral ranking result.
- **:** Runs the scraper while emulating a mobile phone (Google Pixel 5). This is crucial for checking your mobile rankings, which can be very different from desktop rankings.

To use them, simply run `python incognito_main.py` or `python mobile_main.py` instead of the main script.

Chapter 5: How the Magic Happens (A Simple Look at the Script)

You don't need to understand code to use this tool, but it can be helpful to know what's happening behind the scenes.

What are Python and Selenium?

- **Python:** A popular programming language known for its readability and power. It's the language the entire script is written in.
- **Selenium:** A tool that allows Python to control a web browser. It's what lets the script open Chrome, type in the search box, click buttons, and read the content of the page.

The Journey of a Keyword: The Script's Logic

When the script runs, it follows a very logical, step-by-step process for each keyword:

1. **Wake Up & Connect:** The script starts and uses your `gcp_credentials.json` file to securely connect to your Google Sheet.
2. **Read the To-Do List:** It reads all the data from the 'Ranking' tab and picks a random batch of keywords to process.
3. **Launch the Browser:** It opens a new Chrome window using the dedicated profile where you are already logged in.
4. **Perform the Search:** It navigates to `google.com`, types the keyword into the search bar, and presses Enter.
5. **Scan Page 1:** It carefully reads the HTML code of the search results page, looking for any of the competitor URLs you listed.
6. **Ask a Question: Did I find everyone?**
    - If yes, it records all the ranks, updates the Google Sheet, and moves to the next keyword.
    - If not, it proceeds to the next step.
7. **Go to the Next Page:** It finds the "Next" button at the bottom of the page and clicks it, just like a human would.
8. **Repeat the Scan:** It repeats the scanning process for pages 2, 3, 4, and 5, updating ranks as it finds them.

9. **Final Report:** After checking up to 5 pages, it updates the Google Sheet with all the ranks it found. Any URL not found is marked "Not Found".
10. **Take a Break:** It waits for a few random seconds before starting the entire process over with the next keyword. This cycle continues until the batch is complete.

Appendix: Troubleshooting & Frequently Asked Questions (FAQ)

Q: I ran the script, but my Google Sheet isn't updating. What's wrong?
A: This is almost always a sharing permission issue.

1. The most common reason is that you **forgot to share your Google Sheet with the** from the `gcp_credentials.json` file. Go back to Step 4 and make sure the service account has **Editor** access.
2. Check that the `SHEET_NAME` and `WORKSHEET_NAME` in your `config.py` file exactly match the names in your Google Sheets.

Q: The script crashes immediately with an error like
A: This confirms the issue above. The script cannot find the sheet because it either doesn't have permission or the name is wrong. Double-check the sharing settings and the names in `config.py`.

Q: I'm getting an error about email login (
A: This means your email credentials in `config.py` are incorrect.

1. Make sure you are using a Google **App Password** for `SENDER_PASSWORD`, not your regular account password.
2. Verify that the `SENDER_EMAIL` is correct.

Q: I'm seeing a lot of CAPTCHAs.
A: This can happen if you run the script too frequently from the same IP address.

- Try running the script less often (e.g., once a day).
- Ensure you have created the master Chrome profile correctly by running `create_master_profile.py` and logging in. If needed, run `refresh_profile.py`.

Q: How do I completely reset the automation?
A:

1. Delete the `Chrome-Master-Profile` folder from your project directory.
2. Run `python create_master_profile.py` again and log in to create a fresh profile.
3. Clear the ranking columns in your Google Sheet for a clean run.