Local Business Scraper: User Guide

1. Introduction

Welcome to your automated Local Business Scraper!

This guide will walk you through every step of setting up and running a simple but powerful tool. This tool automatically finds businesses on Google Maps (like restaurants, cafes, or stores) and saves their information neatly into a Google Sheet for you.

What does it do?

- **Searches** for a specific type of business in a location you choose (e.g., "bookstores in London").
- **Collects** important details: name, address, rating, website, phone number, etc.
- **Saves** this information to a Google Sheet.
- **Avoids Duplicates**: If you run it again, it will only add new businesses it hasn't found before.

No programming experience is needed! Just follow the steps below carefully.

Table of Contents

2. How It Works (The Big Picture)

Imagine you're hiring a robot assistant to build a list of businesses for you. Here are the three parts of our system:

1. **The Notebook (Google Sheets):** This is a blank online spreadsheet where your robot will write down all the information it finds. It's organized and easy to access.
2. **The Robot's Brain (The Python Script):** This is the set of instructions we give our robot. The instructions are written in a language called Python. This file tells the robot exactly what to search for and where to save the information.
3. **The Helpful Librarian (SerpApi Service):** Instead of sending our robot to wander

through the giant library of Google Maps (which is difficult and against the rules), we send it to a special librarian called an **API**. We ask the librarian for the information, and it gives us a perfectly organized list. This is much faster and more reliable.

3. Part 1: One-Time Setup

This section covers all the tools and accounts you need to set up. You only have to do this once.

Step 1: Install Python on Your Computer

Python is the language our script is written in.

1. Go to the official Python website: python.org
2. Click the big yellow **"Download Python"** button. It will suggest the correct version for your computer (Windows or Mac).
3. Open the file you just downloaded to start the installation.
4. **CRITICAL STEP:** On the first installation screen, you **must** check the box at the bottom that says **"Add Python to PATH"** or **"Add python.exe to PATH"**. This is very important.
5. Click "Install Now" and let the installation finish.

Step 2: Create Your Google Sheet

1. Go to sheets.google.com.
2. Click on the **"Blank"** spreadsheet with the large green plus sign to create a new one.
3. Look at the URL (the web address) in your browser. It will look like this: https://docs.google.com/spreadsheets/d/1aBcDeFgHiJkLmNoPqRsTuVwXyZa_12345AbCdEfG/edit
4. The long string of random characters in the middle is the **Sheet ID**.
5. **Copy this Sheet ID** and paste it into a temporary text file. We will need it later.

Step 3: Get Your "Librarian" API Key (from SerpApi)

1. Go to the SerpApi website: serpapi.com and sign up for a free account.
2. After signing up, go to your dashboard. You will see your **Private API Key**.
3. **Copy this API Key** and paste it into your text file. Keep it safe!

Step 4: Get Your Google "Robot" Credentials

This is the most technical step. Follow it carefully to give our script permission to write to your Google Sheet.

1. **Go to the Google Cloud Console:** console.cloud.google.com.
2. **Create a New Project:** At the top of the page, click the project dropdown menu (it might say "Select a project") and click **"NEW PROJECT"**.
   - Give it a name, like Business Scraper Project, and click **"CREATE"**.
3. **Enable the APIs:** We need to turn on the tools for Google Drive and Google Sheets.
   - Click the navigation menu (☰) in the top left, go to **APIs & Services > Library**.

    - Search for **"Google Drive API"**, click on it, and then click the **"ENABLE"** button.
    - Go back to the Library and do the same for the **"Google Sheets API"**.
4. **Create the Robot User (Service Account):**
    - In the navigation menu (☰), go to **APIs & Services > Credentials**.
    - Click **"+ CREATE CREDENTIALS"** at the top and select **"Service account"**.
    - Give it a name, like sheets-writer-robot.
    - Click **"CREATE AND CONTINUE"**.
    - For the "Role", select **Basic > Editor**.
    - Click **"CONTINUE"**, then click **"DONE"**.
5. **Download the Secret Key:**
    - You will now be back on the Credentials screen. Under "Service Accounts", find the robot user you just created and click on its email address.
    - Click on the **"KEYS"** tab at the top.
    - Click **"ADD KEY"** and then **"Create new key"**.
    - Choose **JSON** as the key type and click **"CREATE"**.
    - A JSON file will automatically download to your computer. **This is your robot's secret key.**
6. **Share the Google Sheet with Your Robot:**
    - Open the downloaded JSON file with a text editor (like Notepad). Find the line that says "client_email" and copy the email address next to it (it will end with @gserviceaccount.com).
    - Go back to your Google Sheet, click the green **"Share"** button in the top right.
    - Paste the robot's email address into the box, make sure it has **"Editor"** permissions, and click **"Send"**.

4. Part 2: Preparing the Script

Now we will set up the folder and the code file.

1. On your computer (e.g., on your Desktop), create a **new folder**. Name it Business-Scraper.
2. Find the **secret key JSON file** you downloaded in the previous step. Move it into the Business-Scraper folder and **rename it to google_creds.json**.
3. Open a plain text editor (like Notepad). Copy all the code from the block below and paste it into the editor.
4. Save the file as main_scraper.py inside your Business-Scraper folder.

```
# main_scraper.py

import os
import requests
import pandas as pd
import gspread
from google.oauth2.service_account import Credentials
```

```
import logging

# --- Configuration Section ---
logging.basicConfig(level=logging.INFO, format='%(asctime)s - %(message)s')

# The script will get these secrets from the commands you run in the terminal.
SERPAPI_KEY = os.getenv('SERPAPI_KEY')
GOOGLE_SHEET_ID = os.getenv('GOOGLE_SHEET_ID')
GOOGLE_CREDENTIALS_FILE = 'google_creds.json'

# --- !!! CUSTOMIZE YOUR SEARCH HERE !!! ---
# Change the text below to search for anything you want.
SEARCH_QUERY = "cafes in New Delhi"
# ----------------------------------------

def get_gspread_client():
    logging.info("Connecting to Google Sheets...")
    scopes = ["https://www.googleapis.com/auth/spreadsheets"]
    creds = Credentials.from_service_account_file(GOOGLE_CREDENTIALS_FILE,
scopes=scopes)
    client = gspread.authorize(creds)
    logging.info("Successfully connected to Google!")
    return client

def get_worksheet(client, sheet_id):
    spreadsheet = client.open_by_key(sheet_id)
    worksheet = spreadsheet.sheet1
    if worksheet.cell(1, 1).value is None:
        headers = ["Name", "Category", "Address", "Rating", "Reviews Count", "Phone",
"Website"]
        worksheet.append_row(headers)
        logging.info("Added headers to the empty sheet.")
    return worksheet

def fetch_google_maps_data(api_key, query):
    logging.info(f"Asking SerpApi for: '{query}'...")
    params = {
        "engine": "google_local",
        "q": query,
        "api_key": api_key,
        "num": 100 # Ask for up to 100 results
    }
    response = requests.get("https://serpapi.com/search", params=params)
    if response.status_code == 200:
        return response.json().get("local_results", [])
    else:
        logging.error(f"API request failed: {response.text}")
        return []

def main():
    logging.info("🤖 Robot is starting its job!")

    if not SERPAPI_KEY or not GOOGLE_SHEET_ID:
        logging.error("ERROR: Secrets not found. Please set SERPAPI_KEY and
GOOGLE_SHEET_ID before running.")
        return

    raw_data = fetch_google_maps_data(SERPAPI_KEY, SEARCH_QUERY)
    if not raw_data:
        logging.warning("The API didn't find any businesses. Stopping.")
        return

    logging.info(f"Found {len(raw_data)} businesses from the API.")

    new_businesses = []
    for item in raw_data:
        new_businesses.append({
            "Name": item.get("title"),
            "Category": item.get("type"),
            "Address": item.get("address"),
            "Rating": item.get("rating"),
            "Reviews Count": item.get("reviews"),
            "Phone": item.get("phone"),
            "Website": item.get("website"),
        })

    new_data_df = pd.DataFrame(new_businesses).dropna(subset=['Name', 'Address'])

    try:
        gspread_client = get_gspread_client()
        worksheet = get_worksheet(gspread_client, GOOGLE_SHEET_ID)
        existing_records = worksheet.get_all_records()
        existing_df = pd.DataFrame(existing_records)
        logging.info(f"Found {len(existing_df)} businesses already in the sheet.")
    except Exception as e:
        logging.error(f"Could not talk to Google Sheets! Error: {e}")
        return

    if not existing_df.empty:
        merged_df = pd.concat([existing_df[['Name', 'Address']], new_data_df[['Name',
'Address']]]).drop_duplicates(keep=False)
        final_new_rows = new_data_df.merge(merged_df, on=['Name', 'Address'], how='inner')
    else:
        final_new_rows = new_data_df

    if not final_new_rows.empty:
        logging.info(f"Adding {len(final_new_rows)} new unique businesses to the sheet...")
        rows_to_append = final_new_rows.fillna('').values.tolist()
        worksheet.append_rows(rows_to_append, value_input_option='USER_ENTERED')
        logging.info("Successfully added the new businesses!")
    else:
        logging.info("No new businesses to add. Everything is up to date!")

    logging.info("✅ Robot has finished the job!")

if __name__ == "__main__":
    main()
```

5. Part 3: Running the Scraper

This is the final step to bring your robot to life.

1. **Open the Command Line:**
   - **On Windows:** Press the Windows Key, type cmd, and press Enter to open "Command Prompt".
   - **On Mac:** Open the "Terminal" app (you can find it in Applications > Utilities).
2. **Navigate to the Folder:** Use the cd (change directory) command to go into your project folder.
   - Example: If your Business-Scraper folder is on your Desktop, you would type:
     cd Desktop/Business-Scraper
3. Install Python Tools: Run this command to install the libraries our script needs.
   pip install requests pandas gspread google-auth-oauthlib
4. **Set Your Secret Keys:** You must provide your secret keys to the script. These commands do it securely, without saving them in the code.
   - On Windows (in Command Prompt):
     set SERPAPI_KEY=PASTE_YOUR_SERPAPI_KEY_HERE
     (Press Enter)
     set GOOGLE_SHEET_ID=PASTE_YOUR_SHEET_ID_HERE

    (Press Enter)
   - On Mac/Linux (in Terminal):
     export SERPAPI_KEY="PASTE_YOUR_SERPAPI_KEY_HERE"
     (Press Enter)
     export GOOGLE_SHEET_ID="PASTE_YOUR_SHEET_ID_HERE"
     (Press Enter)
     Replace the placeholder text with the actual keys you saved in your text file.
5. Run the Robot!
   Execute the final command to start the script:
   python main_scraper.py

You will see status messages appear in the terminal window. If everything is correct, it will end with **"✅ Robot has finished the job!"**. Go check your Google Sheet—it should now be populated with data!

6. Customization & Troubleshooting

- How to Change Your Search?
  Open the main_scraper.py file. Find the line that says SEARCH_QUERY = "cafes in New Delhi" and change the text in the quotes to whatever you want to search for. Save the file and run the script again.
- **Common Errors:**
  - **"File not found: google_creds.json"**: Make sure you have renamed the secret key file correctly and that it is in the *same folder* as your main_scraper.py file.
  - **"API request failed"**: Double-check that you set your SERPAPI_KEY correctly in the terminal. Make sure there are no extra spaces.
  - **"Permission Denied" or errors connecting to Google:** This usually means you forgot to share your Google Sheet with the robot's email address, or you didn't give it "Editor" permissions.