**CODE:**

import requests

import pandas as pd

import re

import time

import logging

from typing import Optional

import ipywidgets as widgets

from IPython.display import display, clear\_output

# -------------------------------------------------

# Logging setup

# -------------------------------------------------

logging.basicConfig(

level=logging.INFO,

format="%(asctime)s - %(levelname)s - %(message)s",

handlers=[

logging.FileHandler("serpapi\_scraper.log"),

logging.StreamHandler()

]

)

def extract\_email\_from\_website(url: str, retries: int = 2) -> Optional[str]:

"""Fetch the first email found on the given website."""

if not url:

return None

if not url.startswith(("http://", "https://")):

url = "https://" + url

for attempt in range(retries):

try:

resp = requests.get(

url,

timeout=8,

headers={"User-Agent":

"Mozilla/5.0 (Windows NT 10.0; Win64; x64) "

"AppleWebKit/537.36 (KHTML, like Gecko) "

"Chrome/140.0.0.0 Safari/537.36"}

)

emails = re.findall(r"[A-Za-z0-9.\_%+-]+@[A-Za-z0-9.-]+\.[A-Za-z]{2,}", resp.text)

if emails:

return emails[0]

except Exception as e:

logging.warning(f"Attempt {attempt+1} failed for {url}: {e}")

time.sleep(2)

return None

# -----------------------------

# Fixed SerpApi key in code

# -----------------------------

SERPAPI\_KEY = "7f41b21e0746f2ca2a9065add6d74c71a89bee1dfe8720992ecc9acbb2353737"

# -----------------------------

# GUI widgets (all blank for user)

# -----------------------------

country\_widget = widgets.Text(

value='',

description='Country:',

layout=widgets.Layout(width='500px')

)

city\_widget = widgets.Text(

value='',

description='City/State:',

layout=widgets.Layout(width='500px')

)

categories\_widget = widgets.Text(

value='',

description='Categories:',

layout=widgets.Layout(width='500px')

)

output\_widget = widgets.Text(

value='',

description='Output File:',

layout=widgets.Layout(width='500px')

)

run\_button = widgets.Button(

description="Run Scraper",

button\_style='success'

)

output\_area = widgets.Output()

# -----------------------------

# Scraper function (updated)

# -----------------------------

def run\_scraper(b):

with output\_area:

clear\_output()

print("Starting scraper...\n")

country = country\_widget.value.strip()

city = city\_widget.value.strip()

categories = [c.strip() for c in categories\_widget.value.split(",")]

output\_file = output\_widget.value.strip()

if not output\_file.endswith(".xlsx"):

output\_file += ".xlsx"

data = []

for category in categories:

query = f"{category} in {city}, {country}"

print(f"Searching Google Maps for '{query}'...")

params = {

"engine": "google\_maps",

"q": query,

"hl": "en",

"type": "search",

"api\_key": SERPAPI\_KEY

}

try:

res = requests.get("https://serpapi.com/search.json", params=params)

res.raise\_for\_status()

results = res.json().get("local\_results", [])

except Exception as e:

print(f"Error for {query}: {e}")

continue

print(f"Found {len(results)} businesses for '{category}'.")

category\_count = 0 # Counter for this category

for r in results:

name = r.get("title")

website = r.get("website")

phone = r.get("phone")

address = r.get("address")

email = extract\_email\_from\_website(website)

if name and email:

category\_count += 1

data.append({

"Business Name": name,

"Email": email,

"Phone": phone or "N/A",

"Website": website or "N/A",

"Address": address or "N/A",

"Category": category,

"Category Count": category\_count # Add count column

})

print(f"Added: {name} | {email}")

else:

print(f"Skipped: {name} (no email found)")

if data:

pd.DataFrame(data).to\_excel(output\_file, index=False)

print(f"\nData saved to {output\_file}, total records: {len(data)}")

else:

print("\nNo data scraped. Check logs for details.")

# -----------------------------

# Connect button

# -----------------------------

run\_button.on\_click(run\_scraper)

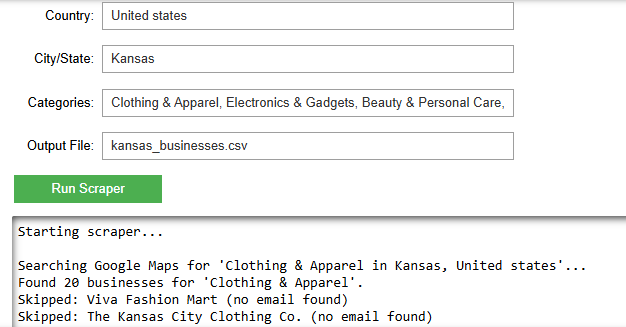
# -----------------------------

# Display GUI

# -----------------------------

display(country\_widget, city\_widget, categories\_widget, output\_widget, run\_button, output\_area)

**OUTPUT:**

****