



<https://www.meetup.com/cannabis-data-science/>



Cannlytics (Cannabis Analytics Free Open Source Software)
- web scraping with Selenium and Beautiful Soup
- parsing pdf files with PDFplumber

Candace O`Sullivan-Sutherland 8/27/2022

<https://github.com/candy-o>

[2021-present] Software Engineer/Data Scientist @ cannlytics.com

[2019-present] Software Engineer/Data Scientist @ TSSG

[1998-2018] Senior Software Performance Engineer @ Dell Technologies DellEMC (DG CLARiiON)

[1995-1998] Senior Software Engineer @ Digital Equipment Corporation, StorageWorks Division

<https://www.meetup.com/cannabis-data-science/>

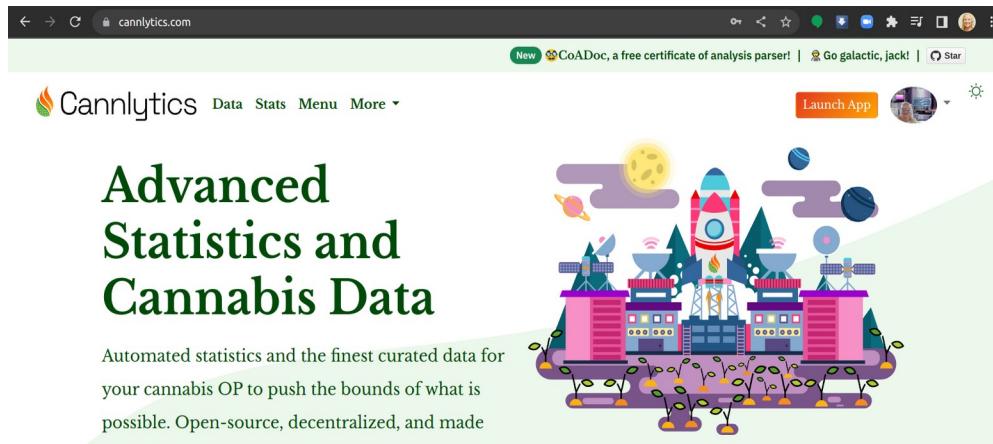
<https://cannlytics.com>

<https://github.com/cannlytics>



*Created using LibreOffice Impress <https://www.libreoffice.org/>
(free software / open source software)*

“Cannlytics” is an Open-source, Decentralized Cannabis Data Science/Analytics organization offering a suite of free software for the Cannabis industry.



Cannlytics Data Stats Menu More ▾

Advanced Statistics and Cannabis Data

Automated statistics and the finest curated data for your cannabis OP to push the bounds of what is possible. Open-source, decentralized, and made with ❤!

Why Cannlytics?



Smart Integrations

We believe that everyone benefits when people are able to study and tinker with their software. With the freedom provided by Cannlytics, users control their software and what it does for them.

[Begin customizing →](#)



Analysis Tailored

Cannlytics provides a user-friendly interface to quickly receive samples, perform analyses, collect and review results, and publish certificates of analysis (CoAs). There are also built in logistics, CRM (client relationship management), inventory management, and invoicing tools.

[View capabilities →](#)

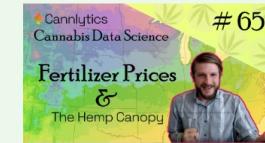


Community Driven

Built by scientist for scientists. Cannlytics empowers you with control over the development process, resources, and decision making authority. We believe that the Cannlytics community is the best judge of how Cannlytics can be improved, so, we have entrusted the Cannlytics source code with you.

[Join today →](#)

Want to explore Cannabis Data Science?



Fertilizer Prices and the US Hemp Canopy | Cannabis Data Science #65

Every piece of the puzzle must be filled in, so we diligently collect all public fertilizer price and hemp yield, harvest, and acreage data under the sun. Please en...

[View more →](#) [Join the Meetup](#) [Get the code & data](#)



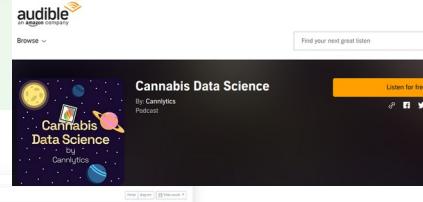
Fertilizers: Costs, Benefits, and Plant Hardiness | Cannabis Data Science #64

What are Nitrogen (N), Phosphorus (P), and Potassium (K) and why should a cannabis cultivator care? This is the exact topic of the day. We follow the money and conn...



Cannabis Consumption: Estimating Consumer Demand | Cannabis Data Science #63

Someone call a plumber! The data dam just burst! The Cannabis Data Science team is all hands on deck serving you up the holy grail of



How can I use Cannlytics?



The Cannlytics Console

If you need a hosted solution, then you can take advantage of the ready to use Cannlytics console for hosting, analysis, and data and file storage. You can manage your lab results and provide your clients with a simple, yet com

[Create an account →](#)



The Cannlytics API

Do you have a technical staff? Then you can utilize the Cannlytics API to programmatically manage your operations and interface with your data. You have all of your data and the power of all laboratory software at your finger tips.

```
{  
    analyses: [...],  
    lab_id: "Cann-d0",  
    sample_name: "Super Easy",  
    results: [...],  
}
```

[Read the Docs →](#)

Cannlytics (creator “Keegan” and Cannlytics Team) built with ❤️

The screenshot shows a web browser displaying the Cannlytics team page at cannlytics.com/team. The page features a header with a red heart icon and the text "Cannlytics is built with the love of...". Below this, there are five team member profiles and two mascot sections.

Keegan Skeate
CEO
Keegan is an economist, software developer, and data scientist who cut his chops as a laboratory analyst. Keegan will work tirelessly to deliver you value for value.

Alice Allafort, Ph.D.
Data Scientist
Alice graduated with a Physics Ph.D. from Stanford University and is always happy to work on interesting data science projects.

Jeffrey Miller
Customer Outreach Developer
Jeff is a customer outreach specialist who revolves around creating value and lives and breathes customer satisfaction.

Raymond Benton
International Communications Specialist / Linguist
Raymond is experienced in agriculture, international trade, and language services. Raymond is also the founder of Advanced Language Liaison.

Michael Pilosov, Ph.D.
Project Manager
Michael is an applied mathematician on a mission to democratize access to quantitative decision-making tools. Michael is here to help you know your data inside and out and interface with your analytics to get the insights that you need.

Candace O’Sullivan-Sutherland
Data Scientist
Candace is a talented data and computer scientist with all the tools in her toolbelt to handle the biggest of big data. Candace brings an extensive suite of hardware capabilities to ensure that your data pipelines are pristine.

CannBot
Chief Scientific Robot
CannBot does the dirty work, working meticulously around the clock to automate the boring stuff and do the heavy lifting.

Cat the Caterpillar
Chief Organic Organism
Cat, the data-pipeline caterpillar, ensures that your data is clean, organized, and gets to where you need it.



Interested in joining?

Do you have the chops and / or passion to help deliver industry-leading analytics, software, and support to everyone in the cannabis industry? Then please email the team at dev@cannlytics.com and explain who, what, where, why, when, and how you can contribute to Cannlytics.

[Join today →](#)

`cannlytics 0.0.11` pipy.org released Aug 20, 2022

The screenshot shows the PyPI project page for 'cannlytics 0.0.11'. At the top, there's a yellow banner with a warning about a phishing attack against PyPI users and a 'Read Details' button. Below the banner, the header includes a search bar, navigation links for Help, Sponsors, Log in, and Register, and a dropdown menu showing 'Latest version'. The main title 'cannlytics 0.0.11' is displayed with a subtitle 'pip install cannlytics'. A note below the title says 'Released: Aug 20, 2022'. A small note at the bottom of the page reads: 'Cannlytics = cannabis + analytics. Data pipelines, user interfaces, and the best statistics in the game. Made with ❤️!'

Navigation

- Project description (selected)
- Release history
- Download files

Project links

- Homepage

Statistics

GitHub statistics:

- Stars: 11
- Forks: 6
- Open issues/PRs: 9

View statistics for this project via [Libraries.io](#), or by using [our public dataset on Google BigQuery](#).

Project description

Cannlytics Engine

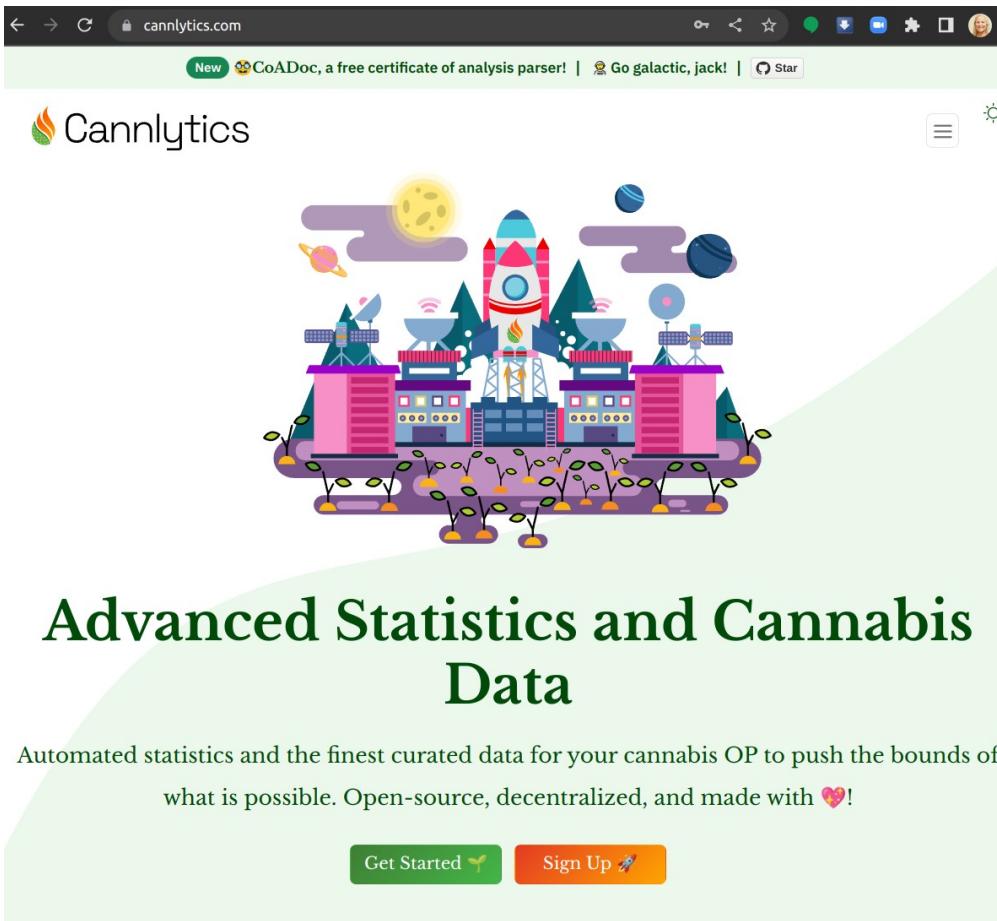
Simple, easy, cannabis analytics.

<https://cannlytics.com>

License: MIT | pypi v0.0.11 | downloads 128/month

Cannlytics is a set of useful tools to get (wrangle), curate, augment, analyze, archive, and market cannabis data.

NEW Cannlytics engine CoADoc reads and parses COA Cannabis test lab “Certificates of Analysis” pdf file(s) and saves to .xlsx format.



New 🚀CoADoc, a free certificate of analysis parser! | 🚀 Go galactic, jack! | ⚡ Star

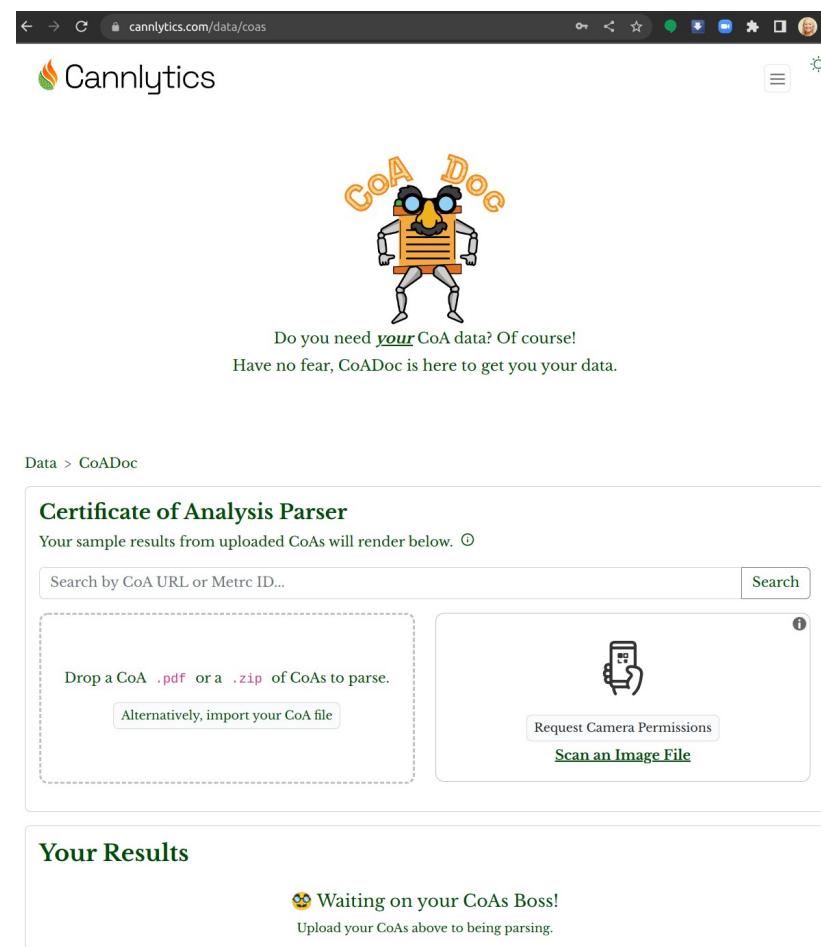
Cannlytics

Advanced Statistics and Cannabis Data

Automated statistics and the finest curated data for your cannabis OP to push the bounds of what is possible. Open-source, decentralized, and made with ❤️!

Get Started 🌱 Sign Up 🚀

Copyright (c) 2020-2022 Cannlytics and the Cannabis Data Science Team



Cannlytics

Do you need *your* CoA data? Of course!
Have no fear, CoADoc is here to get you your data.

Data > CoADoc

Certificate of Analysis Parser

Your sample results from uploaded CoAs will render below. ⓘ

Search by CoA URL or Metrc ID... Search

Drop a CoA .pdf or a .zip of CoAs to parse.
Alternatively, import your CoA file

Request Camera Permissions Scan an Image File

Your Results

⌚ Waiting on your CoAs Boss!
Upload your CoAs above to begin parsing.

CoADoc read, parsed and placed our data into a table

Your Results

Download Results Clear

Woodnote Farms:
Pistachio
Flower, Inhalable
HUMBOLDT
SUNGROWERS GUILD,
LLC
02/19/2022

Sample Results | Woodnote Farms: Pistachio

Woodnote Farms: Pistachio
Flower, Inhalable
HUMBOLDT SUNGROWERS GUILD, LLC
02/19/2022

Analysis	Compound	Result	Units	Status
terpenes	Nerolidol	0.1103	percent	
terpenes	Linalool	0.066	percent	
terpenes	Terpineol	0.0339	percent	
terpenes	Fenchol	0.0322	percent	
terpenes	trans-b-Farnesene	0.024	percent	
terpenes	Caryophyllene Oxide	0.0213	percent	
terpenes	b-Ocimene	0.0174	percent	
terpenes	Camphene	0.0166	percent	
terpenes	Borneol	0.0132	percent	
terpenes	Valencene	0.01	percent	
terpenes	Terpinolene	0.0061	percent	
terpenes	Sabinene Hydrate	0.0054	percent	
cannabinoids	THCa	26.875	percent	
cannabinoids	D9-THC	1.15	percent	
cannabinoids	CBGa	0.96	percent	
cannabinoids	CBCa	0.43	percent	
cannabinoids	THCVa	0.13	percent	
cannabinoids	CBDa	0.126	percent	
cannabinoids	CBG	<LOQ	percent	
cannabinoids	CBC	<LOQ	percent	
cannabinoids	D8-THC	ND	percent	
cannabinoids	THCV	ND	percent	
cannabinoids	CBD	ND	percent	
cannabinoids	CBDV	ND	percent	
pesticides	Cyfluthrin	ND	µg/g	PASS
pesticides	Cypermethrin	ND	µg/g	PASS
pesticides	Diazinon	ND	µg/g	PASS
pesticides	Dimethomorph	ND	µg/g	PASS
pesticides	Etoxazole	ND	µg/g	PASS
pesticides	Fenhexamid	ND	µg/g	PASS
pesticides	Fenpyroximate	ND	µg/g	PASS
pesticides	Flonicamid	ND	µg/g	PASS
pesticides	Fludioxonil	ND	µg/g	PASS
pesticides	Hexythiazox	ND	µg/g	PASS
pesticides	Imidacloprid	ND	µg/g	PASS
pesticides	Kresoxim-methyl	ND	µg/g	PASS

Custom CoA Parsing

At this time, CoADoc can only parse certificates of analysis (COAs) from labs and LIMS with validated parsing algorithms. We've validated:

Labs	LIMS
Anresco Laboratories	Confident Cannabis
Cannalysis	TagLeaf LIMS
Green Leaf Lab	
MCR Labs	
SC Labs	
Sonoma Lab Works	
Veda Scientific	

If you want your favorite lab or LIMS added, then please email dev@cannlytics.com and chances are that they can be included. Alternatively, because

Download - saved our CoA data to an .xlsx worksheet

coa-data-2022-08-26-17-50-48.xlsx

coa-data-2022-08-26-17-50-48.xlsx - LibreOffice Calc																
EN1	EB	EC	ED	EE	EF	EG	EH	EI	EJ	EK	EL	EM	EN	EO		
1	acequinocid	acetamiprid	azoxystrobin	bifenazate	bifenthrin	boscalid	captan	carbaryl	chlorantraniliprole	clofentezine	water_actives	alpha_hex	piperonyl	dichlorvos		
2	1E-09	1E-09	1E-09	1E-09	1E-09	1E-09	1E-09	1E-09	1E-09	1E-09	1E-09	0.175	1E-09	1E-09		
3																
4																
5																
6																
7																
8																
9																
10																
11																
12																
13																
14																
15																
16																
17																
18																
19																
20																
21																
22																
23																
24																
25																
26																
27																
28																
29																
30																
31																
32																
33																
34																
35																
36																
37																
38																
39																
40																
41																
42																
43																
44																
45																
46																
47																

Let's Start the Debugger and step thru some code...

<https://github.com/cannlytics/>

- web_scraping_script.py (Selenium demo)
- cs_labs.py (Beautiful Soup demo)
- coa.py (pdf_plumber demo)

Code is available on <https://github.com/cannlytics/>

The screenshot shows the GitHub repository page for 'cannlytics'. The repository name is 'Cannlytics' and it is described as 'Simple, easy, end-to-end cannabis analytics. Open source. Made by scientists for scientists.' It has 20 repositories, 3 projects, and 2 packages. The README file is visible, and the repository is set to 'Public'. It also lists 'People' and 'Top languages' (Python, JavaScript, Dart, CSS, HTML) and 'Most used topics' (cannabis, cannabis-app, cannabis-data, cannabisapp, python).

Cannlytics main repository

Copyright (c) 2020-2022 Cannlytics and the Cannabis Data Science Team

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Please cite the following if you use the code examples in your research:

```
@misc{cannlytics2022,
  title={Cannabis Data Science},
  author={Skeate, Keegan and Rice, Charles and O'Sullivan-Sutherland, Candace},
  journal={https://github.com/cannlytics/cannabis-data-science},
  year={2022}
}
```

Machine Learning algorithms require lots of Data for training accuracy

```
File Edit Selection View Go Run Terminal Help

EXPLORER
CANNLYTICS
> .firebase
> .vscode
> ai
> api
> build
> cannlytics
> auth
> data
> ccrs
> coas
> __init__.py
> anresco.py
> cannalysis.py
> coas.py
> confidentcannabis.py
> greenleaflab.py
> mcrlabs.py
① readme.md
> sclabs.py
> sonoma.py
> tagleaf.py
> veda.py
> web_scraping_script.py
> market
> __init__.py
> data.py
> figures.py
> flower_art.py
> gis.py
> opendata.py
> patents.py
① readme.md
> web.py
> firebase
> lims
> metrc
> models
> paypal
> quickbooks
> stats
> utils
> __init__.py
> cannlytics.py
① readme.md

sclabs.py  web_scraping_script.py x

cannlytics > data > coas > web_scraping_script.py > ...
1 """
2 Web Scraping of PSI Labs Test Results
3 Copyright (c) 2022 Cannlytics
4
5 Authors: Keegan Skeate <https://github.com/keeganskeate>
6 Created: July 4th, 2022
7 Updated: 7/5/2022
8 License: MIT License <https://opensource.org/licenses/MIT>
9
10 Description:
11     Archive all of the PSI Labs test results.
12
13 Data Sources:
14
15     - PSI Labs Test Results
16     URL: <https://results.psilabs.org/test-results/>
17
18 Resources:
19
20     - ChromeDriver
21     URL: <https://chromedriver.chromium.org/home>
22
23 Setup:
24
25     1. Create a folder `../../../../datasets/michigan` to store your data.
26
27     2. Download ChromeDriver and put it in your `C:\Python39\Scripts` folder
28     or pass the `executable_path` to the `Service`.
29
30     3. Pick the `PAGES` that you want to collect.
31 """
32 # Standard imports.
33 from datetime import datetime
34 from hashlib import sha256
35 import hmac
36 import os
37 import os

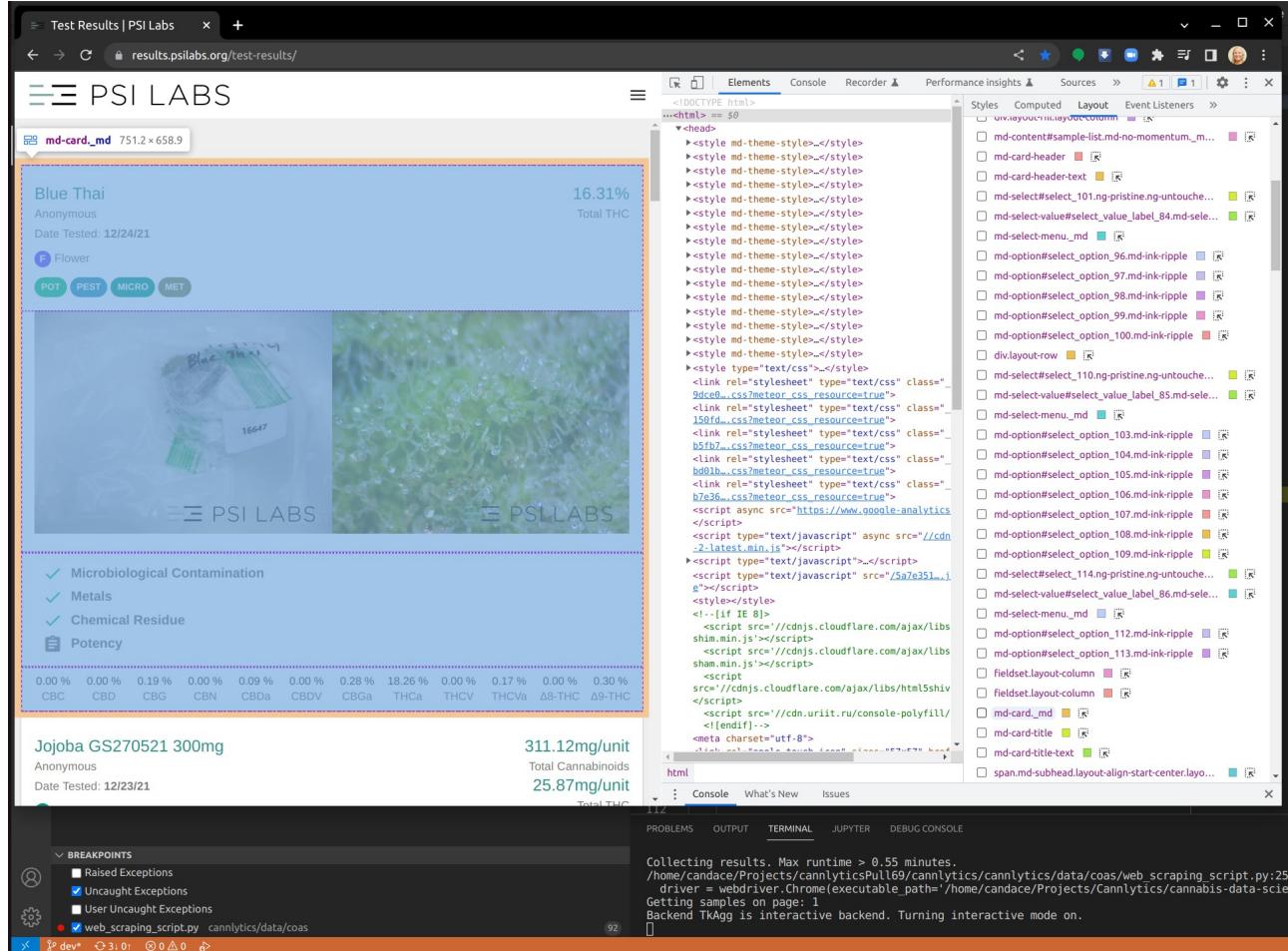
PROBLEMS OUTPUT TERMINAL JUPYTER DEBUG CONSOLE

(cannlyticsPull69) candace@candace-OMEN-by-HP-Laptop-17-cb1xxx:~/Pr
(cannlyticsPull69) candace@candace-OMEN-by-HP-Laptop-17-cb1xxx:~/Projects/cannlyticsPull69/cannlytics$
```

- Cannlytics collects and cleans available Cannabis USA data for robust datasets.
 - There is much data to collect (Universities, Freedom of Information Act, Cannabis Lab Compliance Test facilities, NIH.gov... with 38 states legalizing medical Cannabis to-date...)
 - I will demo `web_scraping_script.py` to webscrape PSI Labs test data results using Selenium API with chrome driver for browser automation, working with core Javascript concepts(DOM) and handling AJAX/PJAX requests.
 - PSI Labs operates in Cannabis legal USA states MI and CA.

<https://www.psilabs.org/>

Use Browser Inspector Tool to view website Elements



- Click on the three vertical dots on chrome's top menu bar, choose More tools, then select Developer Tools
- Right-click on the web page and choose Inspect to access Developer Tools
- Use keyboard shortcuts Cntl-Shift+I or F12 for Windows or Linux and cmd+option+I for macOS users

OR

- Right-click on the web page and choose Inspect to access Developer Tools

OR

- Use keyboard shortcuts Cntl-Shift+I or F12 for Windows or Linux and cmd+option+I for macOS users

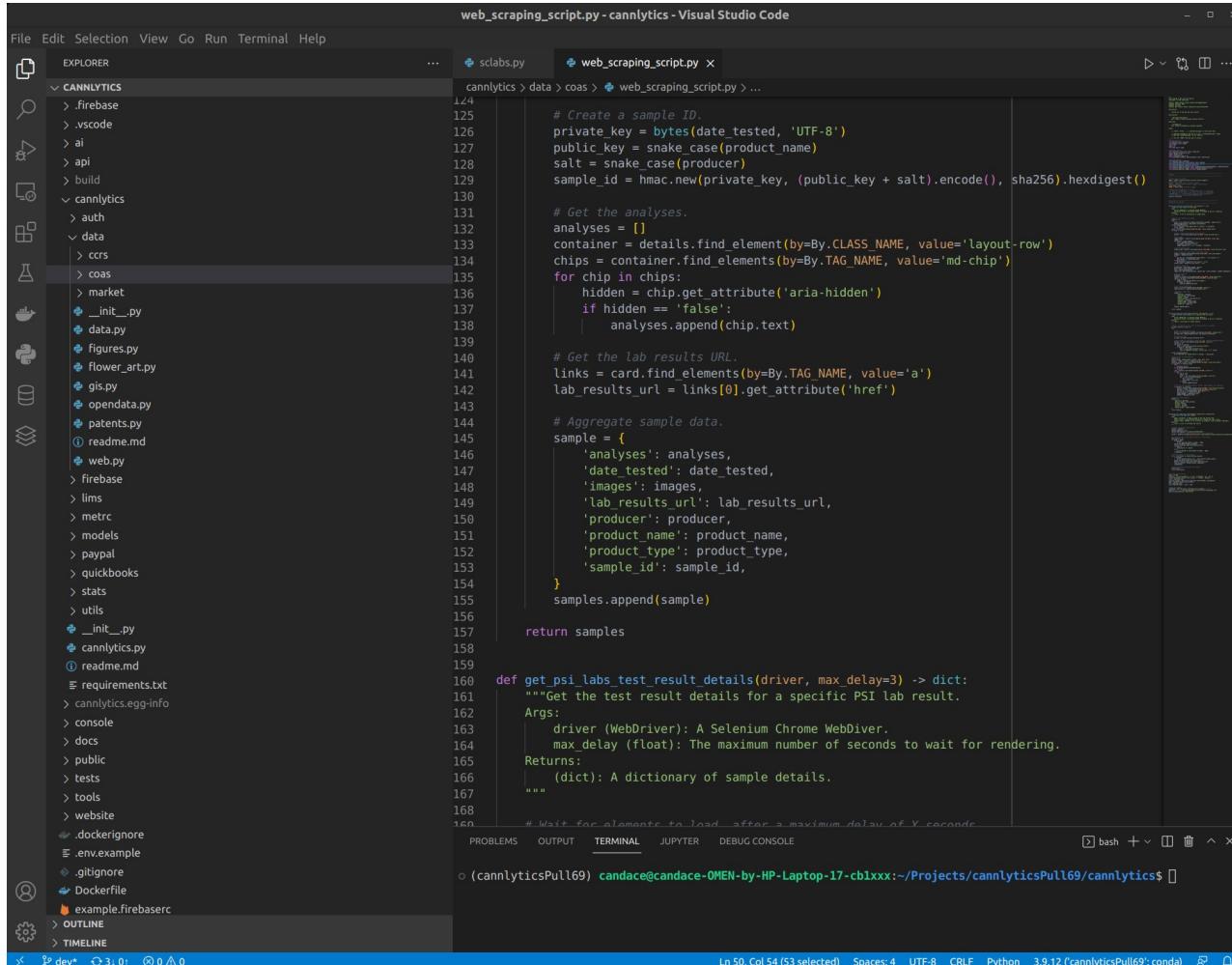
```
`web_scraping.py` Selenium Demo: specify chromedriver, declare variables, function  
`get_psi_labs_test_result` return List "samples"
```

The screenshot shows a Visual Studio Code interface with the following details:

- File Explorer:** On the left, it shows a tree view of the project structure under the "CANNLYTICS" folder. The "COAS" folder contains several Python files: __init__.py, anresco.py, cannalysis.py, coas.py, confidentcannabis.py, greenleaflab.py, mcrlabs.py, README.md, sclabs.py, sonoma.py, tagleaf.py, veda.py, and web_scraping_script.py.
- Code Editor:** The main editor area displays the content of the "web_scraping_script.py" file. The code uses Selenium to interact with a web browser. It defines a function "get_psilabs_test_results" that takes a WebDriver and a maximum delay as arguments, returning a list of dictionaries representing sample data from PSI labs.
- Terminal:** At the bottom, the terminal window shows the command "candace@candace-0MEN-by-HP-Laptop-17-cb1xxx:~/Projects/cannlyticsPull69/cannlytics\$".
- Bottom Bar:** The bottom bar includes tabs for PROBLEMS, OUTPUT, TERMINAL, JUPYTER, and DEBUG CONSOLE, along with status icons for file changes and battery level.

- Specify chromedriver in c:\Python\Scripts or equivalent
OR
Specify the full path to the driver
`#DRIVER_PATH='anaconda3/envs/web_scraping/bin/chromedriver' #full_driver_path=os.path.abspath(DRIVER_PATH)`
 - Setup variables are declared
Base URL, range of pages, total pages to collect
 - Service = Service() is where we pass the Selenium Service object imported previously to use an instance of the Service() class from selenium.webdriver.chrome.service
 - Define function `get_psi_labs_test_results` which collects test results for PSI labs and returns List "samples"
 - Get all the samples on the page
 - Get sample details from card
 - Images, product_name, producer, date tested, product_type
 - Create sample ID
 - Get the analyses
 - Get the Lab results URL
 - Aggregate "sample" data then append to/return List "samples" data

Declare function `get_psi_labs_test_details` return Dictionary “details”



The screenshot shows a Visual Studio Code interface with two tabs open: 'sclabs.py' and 'web_scraping_script.py'. The 'web_scraping_script.py' tab contains the following Python code:

```
# Create a sample ID.
private_key = bytes(date_tested, 'UTF-8')
public_key = snake_case(product_name)
salt = snake_case(producer)
sample_id = hmac.new(private_key, (public_key + salt).encode(), sha256).hexdigest()

# Get the analyses.
analyses = []
container = details.find_element(by=By.CLASS_NAME, value='layout-row')
chips = container.find_elements(by=By.TAG_NAME, value='md-chip')
for chip in chips:
    hidden = chip.get_attribute('aria-hidden')
    if hidden == 'false':
        analyses.append(chip.text)

# Get the lab results URL.
links = card.find_elements(by=By.TAG_NAME, value='a')
lab_results_url = links[0].get_attribute('href')

# Aggregate sample data.
sample = {
    'analyses': analyses,
    'date_tested': date_tested,
    'images': images,
    'lab_results_url': lab_results_url,
    'producer': producer,
    'product_name': product_name,
    'product_type': product_type,
    'sample_id': sample_id,
}
samples.append(sample)

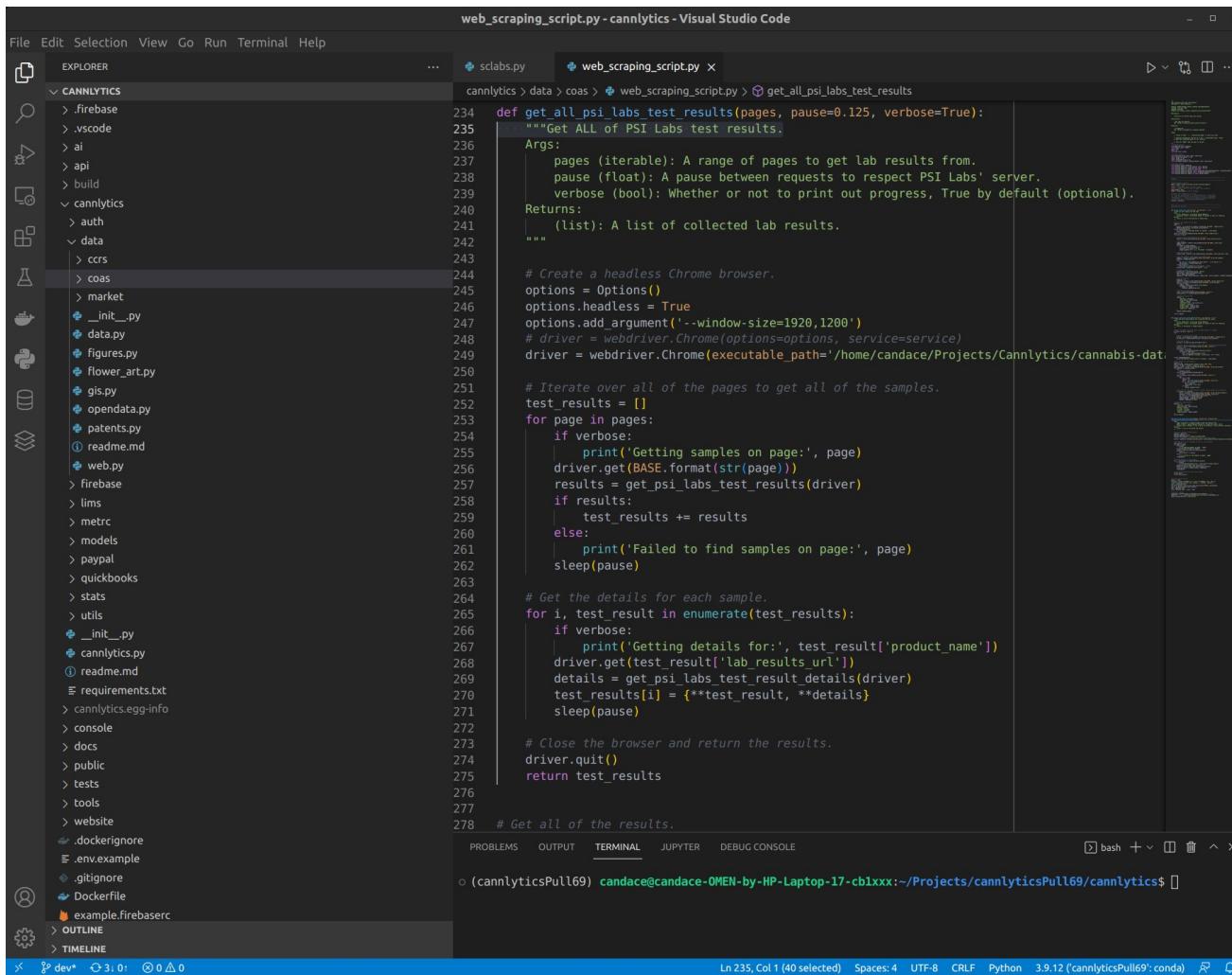
return samples

def get_psi_labs_test_result_details(driver, max_delay=3) -> dict:
    """Get the test result details for a specific PSI lab result.
    Args:
        driver (WebDriver): A Selenium Chrome WebDiver.
        max_delay (float): The maximum number of seconds to wait for rendering.
    Returns:
        (dict): A dictionary of sample details.
    """
    # Wait for elements to load after a maximum delay of X seconds.
```

The code is used for web scraping, specifically for extracting analysis details and lab results from a page. It uses Selenium WebDriver to interact with the browser and BeautifulSoup to parse the HTML. The 'samples' variable is a list of dictionaries, where each dictionary represents a sample with its analysis details, date tested, images, lab results URL, producer, product name, product type, and sample ID.

- Define function
`get_psi_labs_test_results_details` collects the test results details for a specific PSI labs test result and returns Dictionary “details”
- Wait for elements to load
- Wait for QR code to load then Get
- Get CoA Certificate of Analysis URLs
- Get Results for each analysis
- Aggregate sample details
- Return Dictionary “Details”

Declare function `get_all_psi_labs_test_result` return List “all_test_results”



```
File Edit Selection View Go Run Terminal Help
... sclabs.py web_scraping_script.py ...
cannlytics > data > coas > web_scraping_script.py > get_all_psi_labs_test_results
234 def get_all_psi_labs_test_results(pages, pause=0.125, verbose=True):
235     """Get ALL of PSI Labs test results.
236     Args:
237         pages (iterable): A range of pages to get lab results from.
238         pause (float): A pause between requests to respect PSI Labs' server.
239         verbose (bool): Whether or not to print out progress, True by default (optional).
240     Returns:
241         (list): A list of collected lab results.
242     """
243
244     # Create a headless Chrome browser.
245     options = Options()
246     options.headless = True
247     options.add_argument('--window-size=1920,1200')
248     # driver = webdriver.Chrome(options=options, service=service)
249     driver = webdriver.Chrome(executable_path='/home/candace/Projects/Cannlytics/cannabis-data/.chromedriver')
250
251     # Iterate over all of the pages to get all of the samples.
252     test_results = []
253     for page in pages:
254         if verbose:
255             print('Getting samples on page:', page)
256         driver.get(BASE.format(str(page)))
257         results = get_psi_labs_test_results(driver)
258         if results:
259             test_results += results
260         else:
261             print('Failed to find samples on page:', page)
262         sleep(pause)
263
264     # Get the details for each sample.
265     for i, test_result in enumerate(test_results):
266         if verbose:
267             print('Getting details for:', test_result['product_name'])
268         driver.get(test_result['lab_results_url'])
269         details = get_psi_labs_test_result_details(driver)
270         test_results[i] = {**test_result, **details}
271         sleep(pause)
272
273     # Close the browser and return the results.
274     driver.quit()
275     return test_results
276
277 # Get all of the results.
278
PROBLEMS OUTPUT TERMINAL JUPYTER DEBUG CONSOLE
(cannlyticsPull69) candace@candace-OMEN-by-HP-Laptop-17-cb1xxx:~/Projects/cannlyticsPull69/cannlytics$ 
```

- Define function `get_all_psi_labs_test_results` which collects ALL of PSI Labs’ test results and returns List “test_results”
- Create headless chrome browser
- Chromedriver gets the Base URL and Page
- Iterate over all the pages to place all the samples into Dictionary “test_results”
- Call `get_psi_labs_test_results` passing chromedriver to get “results” and add to “test_results”
- Call `get_psi_labs_test_result_details` passing chromedriver to get Dictionary “details” to add to List “test_results”
- “all_test_results” contains “test_results”
- Convert List “all_test_results” to Dataframe then write out to .xlsx file

VSCode Debugger used to view variable values

The screenshot shows the Visual Studio Code interface with the following sections visible:

- File Edit Selection View Go Run Terminal Help**: The top navigation bar.
- RUN AND DEBUG**: A toolbar with icons for run, stop, and step operations.
- VARIABLES**: A sidebar showing the current state of variables. It includes sections for **Locals** (function variables) and **WATCH** (samples).
- scabs.py**: The code editor window containing Python code related to web scraping. A red dot indicates the current line of execution.
- CALL STACK**: Shows the current call stack with frames from `get_ps1_labs_test_results`, `get_all_ps1_labs_test_results`, and `web_scraping_script.py`.
- BREAKPOINTS**: A sidebar showing breakpoints set in the code.
- PROBLEMS**, **OUTPUT**, **TERMINAL**, **JUPYTER**, **DEBUG CONSOLE**: The bottom navigation tabs for the terminal and output panes.
- Bottom Status Bar**: Shows file paths like `dev`, `3:0`, `0:0`, and other status indicators.

The main code editor area displays the following Python code:

```
PAGES = range(1, 4921) # Collect N pages.
PAGES = range(1, 2) # Collect N pages.

# Specify your chromedriver. You can:
# 1. Put your chromedriver in 'C:\Python39\Scripts' or equivalent.
# 2. Specify the full path to the driver, but this is troublesome.
# DRIVER_PATH = '../assets/tools/chromedriver_win32/chromedriver'
# full_driver_path = os.path.abspath(DRIVER_PATH)

service = Service()

# Getting ALL the data.

def get_ps1_labs_test():
    """Get all test samples"""
    Args:
        driver (WebD
    max_delay (f
    Returns:
        (list): A li
    """
    # Get all the sa
    samples = []
    try:
        detect = EC.
    except TimeoutError:
        print('Failed')
        return sample
    cards = driver.f
    for card in card
        # Begin gett
        details = card.find_element(by=By.TAG_NAME, value='md-card-title')

    # Get images.
    image_elements = details.find_elements(by=By.TAG_NAME, value='img')
    images = []
    for image in image_elements:
        src = image.get_attribute('src')
        filename = src.split('/')[-1]
        images.append({'url': src, 'filename': filename})

    Collecting results. Max runtime > 0.55 minutes.
    /home/candace/Projects/cannlyticsPull69/cannlytics/cannlytics/data/coas/web_scraping_script.py:250
    driver = webdriver.Chrome(executable_path='/home/candace/Projects/Cannlytics/cannabis-data-scienc
    Getting samples on page: 1
    Backend TkAgg is interactive backend. Turning interactive mode on.
```

- Hover over a variable to see its value
- OR
- Right-click on the web page and choose Inspect to access Developer Tools
- OR
- Use keyboard shortcuts Cntl-Shift+I or F12 for Windows or Linux and cmd+option+I for macOS users

VSCode Breakpoint(s) used to pause debugger execution

The screenshot shows the Visual Studio Code interface with the following details:

- File Explorer:** Shows files like `scabls.py` and `web_scraping_script.py`.
- Run and Debug View:** Shows variables, locals, and special variables. A breakpoint is set at line 92 of `web_scraping_script.py`.
- Code Editor:** The code for `web_scraping_script.py` is displayed. Line 92 is highlighted with a yellow background and contains the breakpoint instruction: `samples = []`.
- Call Stack:** Shows the call stack with frames: `get_psilabs_test_results` at line 92, `get_all_psilabs_test_results` at line 258, and `` at line 286.
- Breakpoints View:** Shows a list of breakpoints, including one for `web_scraping_script.py` at line 92.
- Terminal:** Shows output from the terminal related to collecting results and backend configuration.

- Breakpoint set at the beginning of `get_psilabs_test_results` line 92
`samples = []` called from `get_all_psilabs_test_results` called from <module>
- view “State of Code”
 - Variables
 - Locals
 - driver:
 - command_executor:
 - current_url:
‘https://results.psilabs.org/test-results/?page=1’
- view “Call Stack”
 - get_psilabs_test_results
 - get_all_psilabs_test_results
 - <module>
- Step into the code

Let's step thru the Selenium web scraping code...

... stay tuned for Beautiful Soup and pdfplumber code snippets and code step thru