

# DDL Automation User Guide

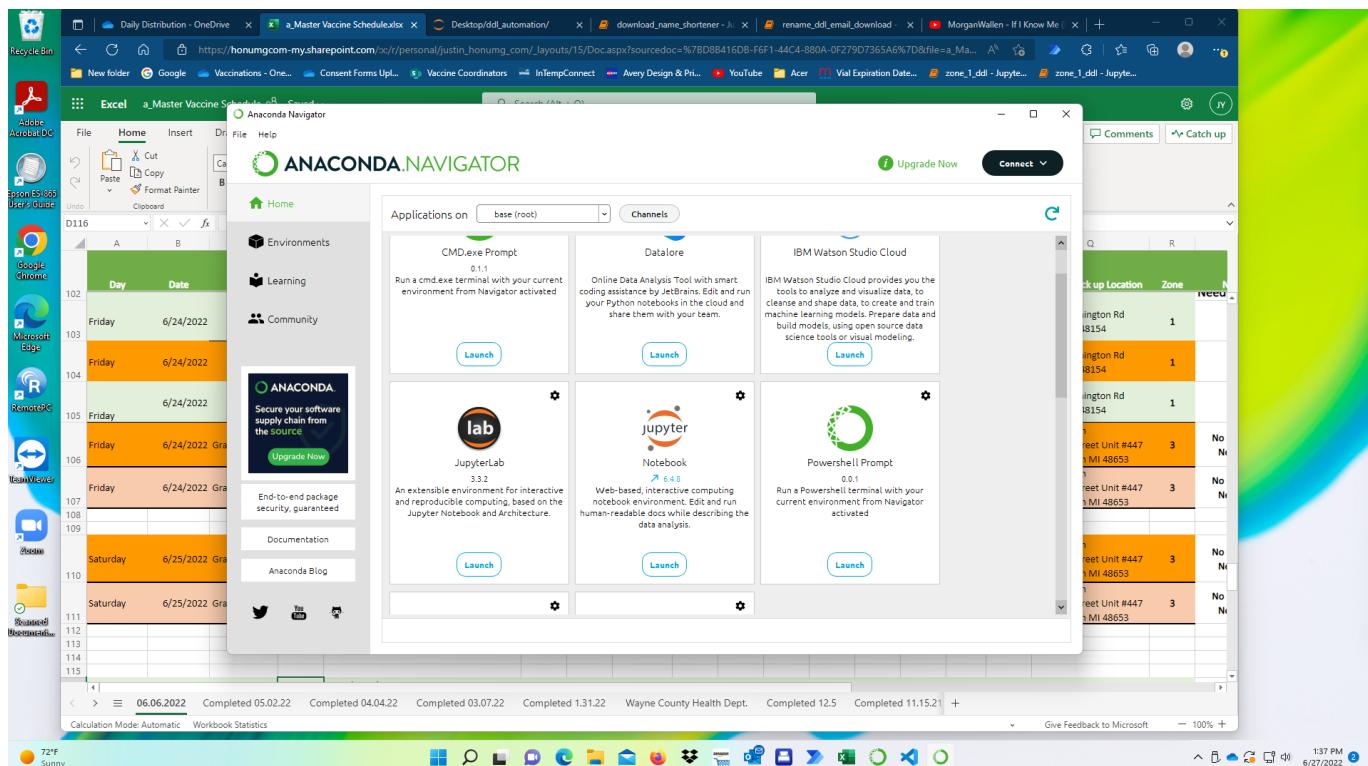


This document is to be used as a user guide for the DDL automation process. This guide will cover the essential tasks of how to Initialize the python environment through anaconda. How to open and run operational files to assist in tasks.

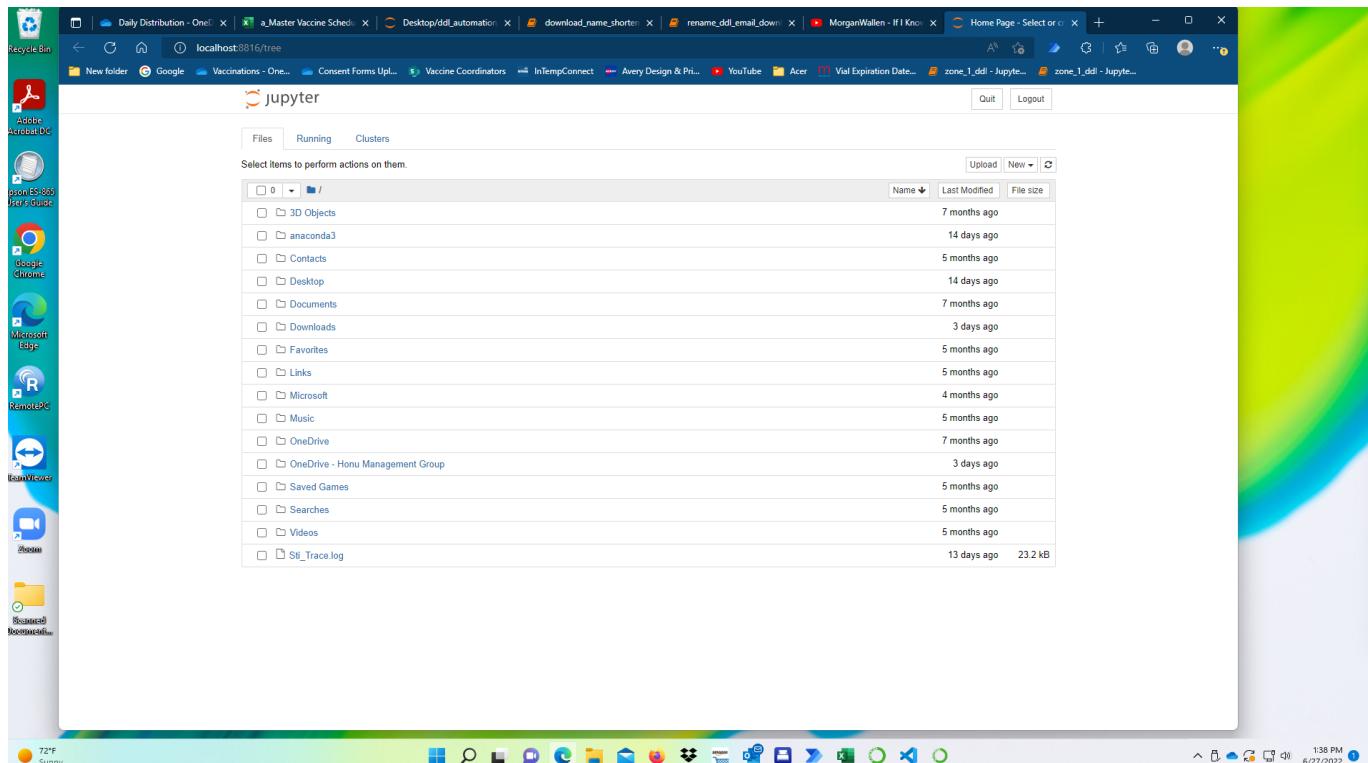
The necessary files have been downloaded and are able to be run on the Jon User account. After logging into the specific user account follow the steps below to access the automation files.

The screenshot shows a Microsoft Excel window titled "a\_Master Vaccine Schedule.xlsx" in the foreground. The spreadsheet contains data for vaccination events across different locations and dates. The columns include Day, Date, County, Event Code, Location, and POC. Several rows are highlighted in orange, indicating specific entries like "TC Latino Grocery" at various locations. The background features a Windows taskbar with icons for various applications like OneDrive, Google, and YouTube. A Microsoft Edge browser window is open, displaying a search bar with the placeholder "Type here to search" and a list of top apps including Microsoft Edge, File Explorer, Word, Excel, and Outlook. The overall interface is a standard Windows desktop environment.

## Open Anaconda

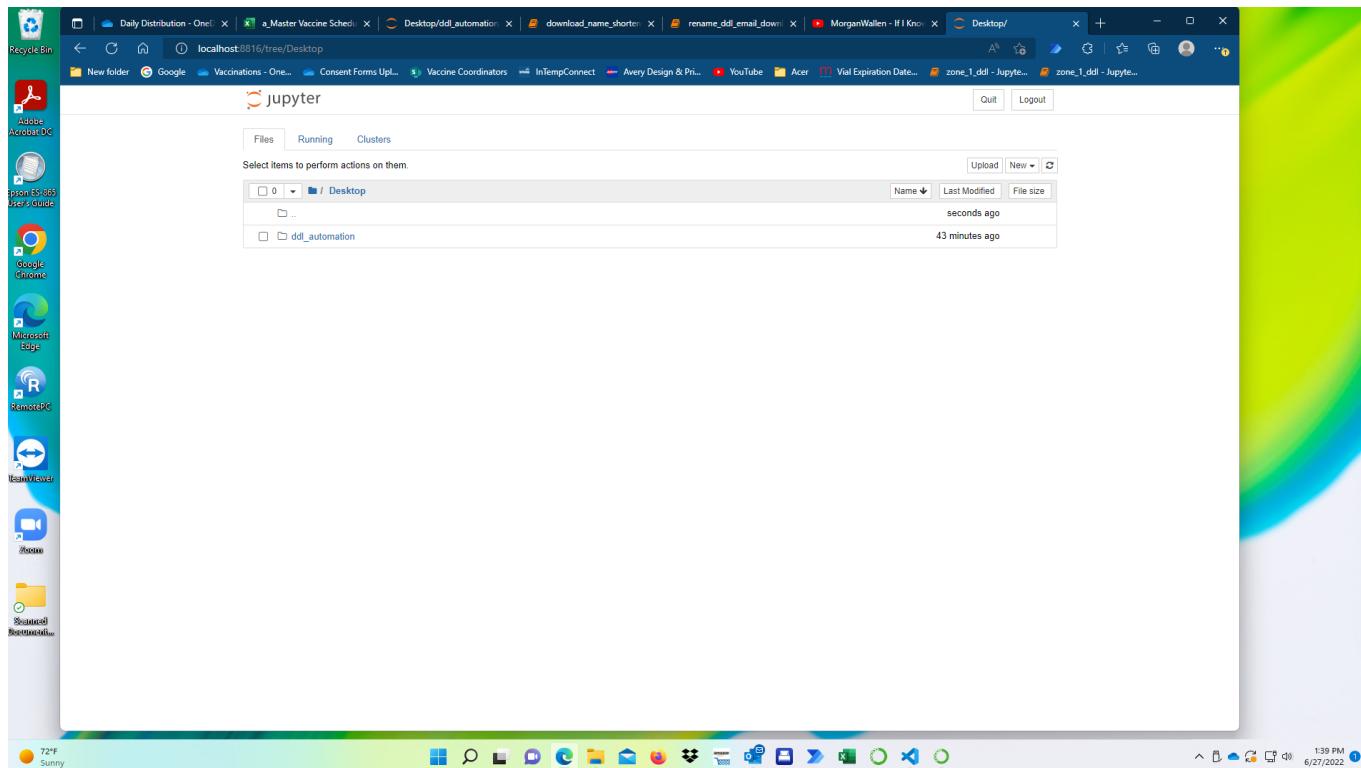


## Launch Jupiter Notebook

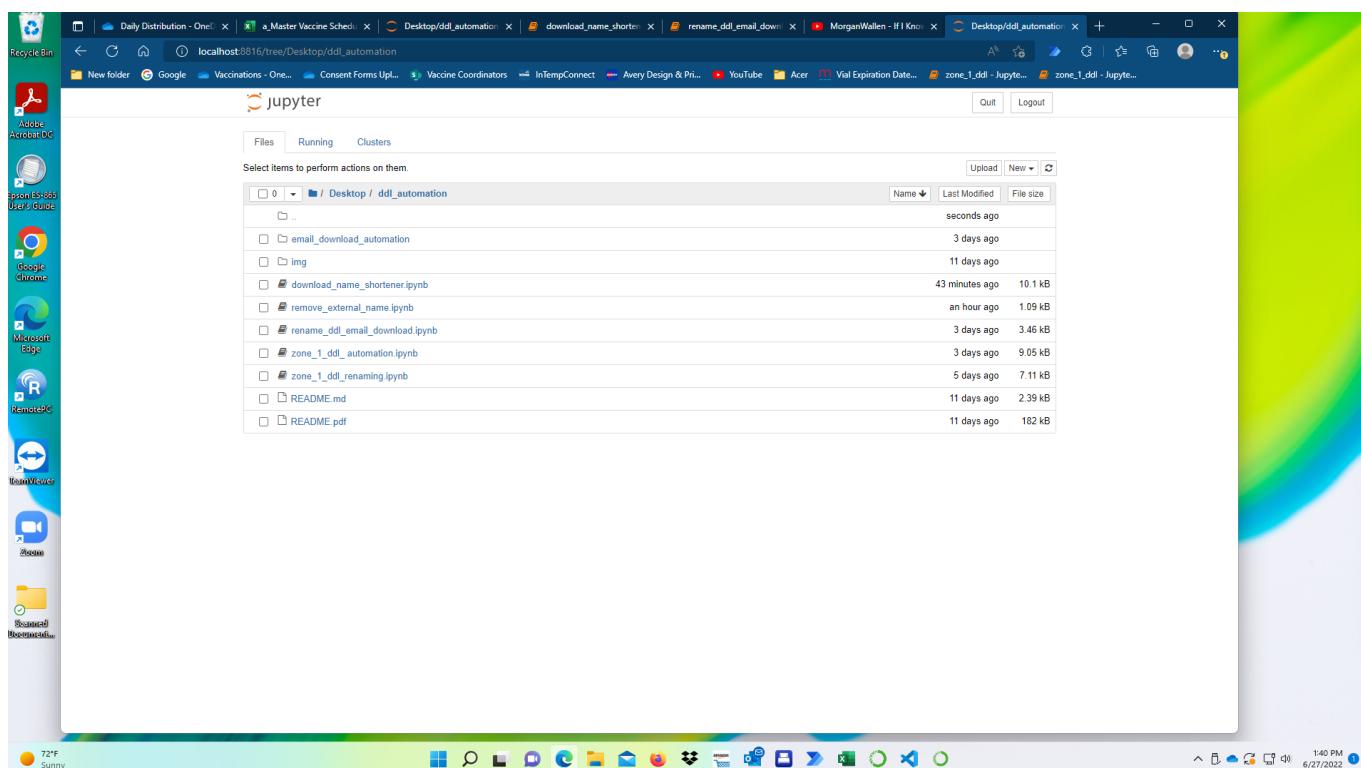


Click through the file path

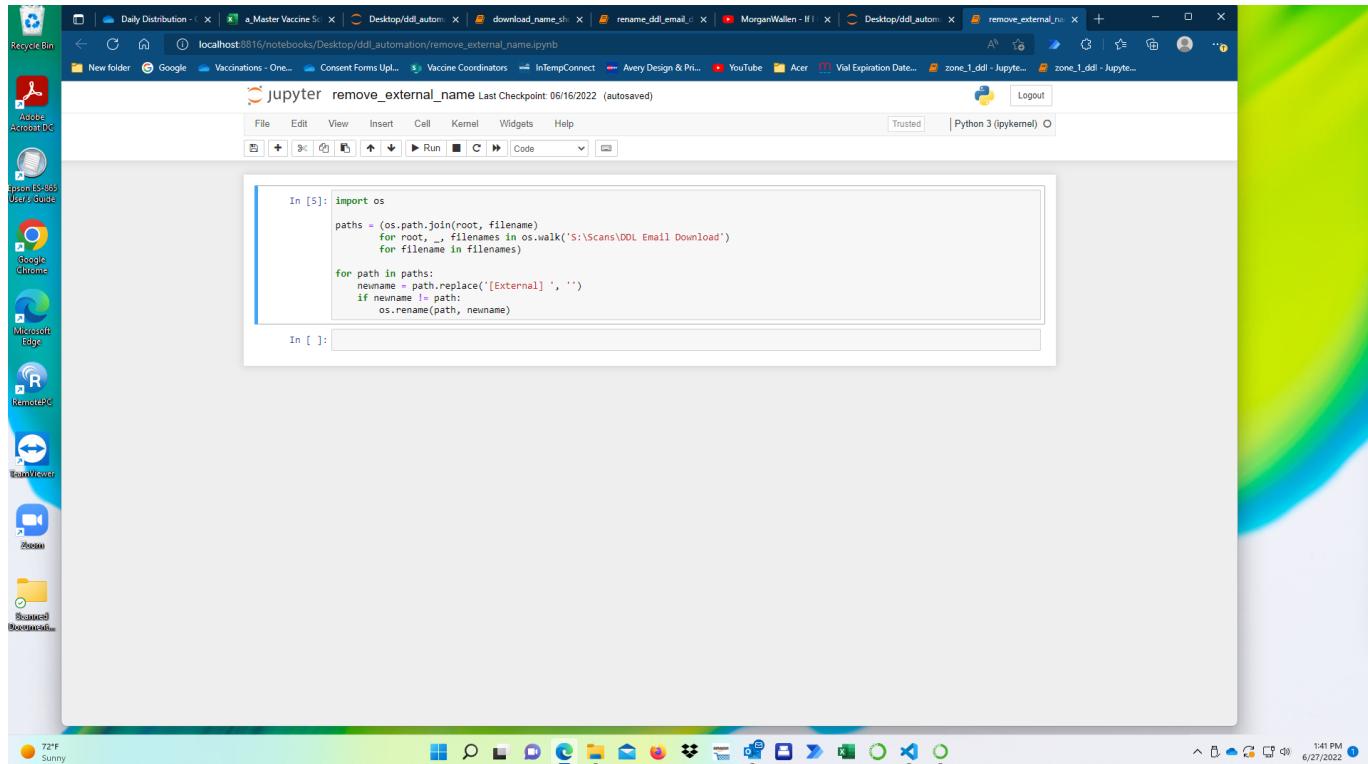
"C:\Users\Jon\Desktop\ddl\_automation"



Open the DDL automation directory

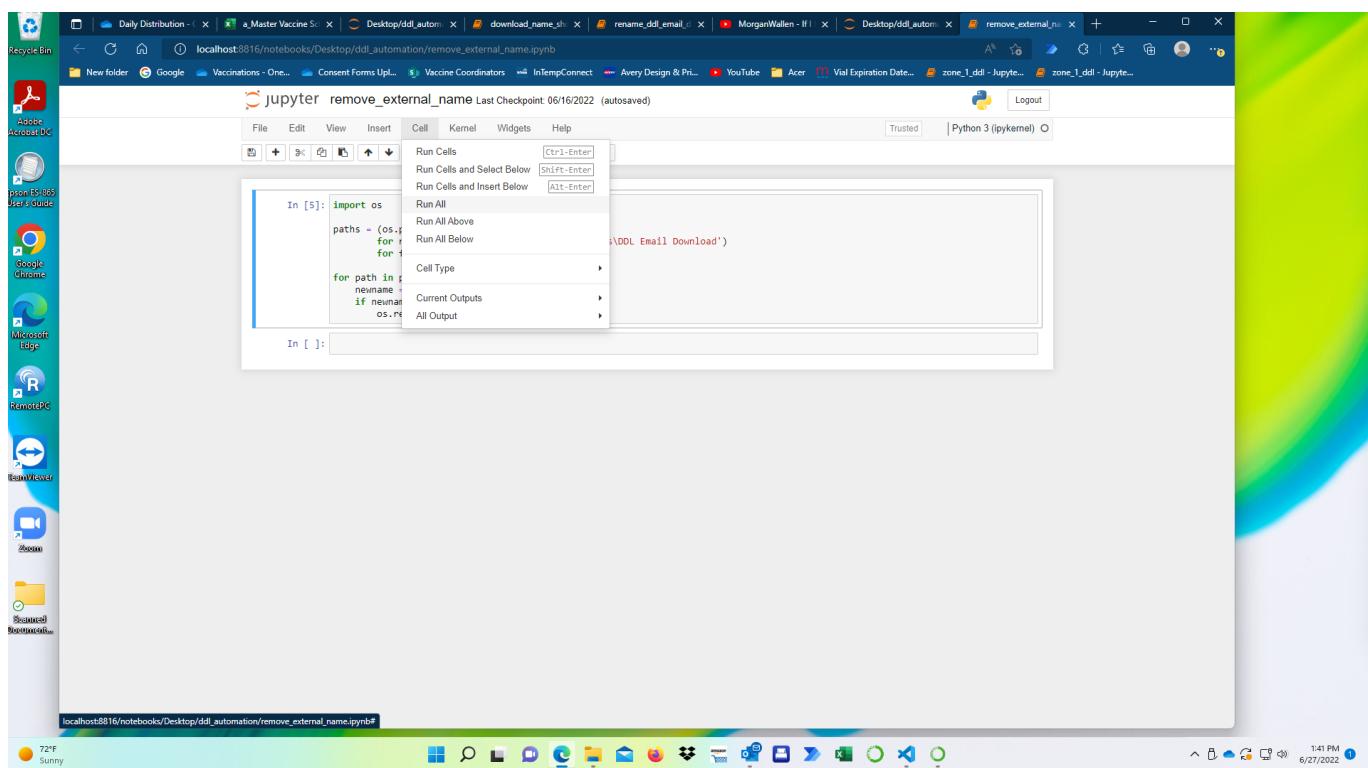


Open the file that you need to run



Click the specific cell you wish to execute and then click the run or play button

---



Some files have multiple cells and you can run all at once if necessary

zone\_1\_dll\_automation file will require you to run all cells at once

---

```

In [21]: import time, os
from selenium import webdriver
from selenium.webdriver.common.keys import Keys
from selenium.webdriver.common.by import By
from selenium.webdriver.chrome.service import Service
from selenium.webdriver.chrome.options import Options
from dotenv import load_dotenv
load_dotenv()

Out[21]: True

In [22]: # .env
chrome_path = r"C:\Users\Jon\Downloads\chromedriver_win32\chromedriver.exe"
email = os.environ.get("EMAIL")
password = os.environ.get("PASSWORD")
url = os.environ.get("URL")

dd11 = "Fridge 10"
dd12 = "Fridge 11"
dd13 = "Fridge 12"
dd14 = "Freezer 18"

date = 'June 19, 2022'

In [23]: print("process Started -----")
process Started -----

In [24]: op = Options()
op.add_argument("--start-maximized") #open Browser in maximized mode
op.add_argument("--no-sandbox") #bypass OS security model
op.add_argument("--disable-dev-shm-usage") #overcome limited resource problems
# op.add_experimental_option('excludeSwitches', ['enable-automation'])
# op.add_experimental_option('useAutomationExtension', False)
op.headless = False # change to true if we want to hide the browser
s = Service(chrome_path)

with webdriver.Chrome(service=s, options=op) as d:
    d.get(url)
    d.find_element(by=By.NAME, value='login-form:j_idt120').send_keys(email)
    d.find_element(by=By.NAME, value='login-form:j_idt125').send_keys(password)
    d.find_element(by=By.ID, value='login-form:j_idt127').click()

```

Before running zone\_1\_ddl\_automation file be sure to edit the date variable to the date that you wish to receive data from

```

In [21]: import time, os
from selenium import webdriver
from selenium.webdriver.common.keys import Keys
from selenium.webdriver.common.by import By
from selenium.webdriver.chrome.service import Service
from selenium.webdriver.chrome.options import Options
from dotenv import load_dotenv
load_dotenv()

Out[21]: True

In [22]: # .env
chrome_path = r"C:\Users\Jon\Downloads\chromedriver_win32\chromedriver.exe"
email = os.environ.get("EMAIL")
password = os.environ.get("PASSWORD")
url = os.environ.get("URL")

dd11 = "Fridge 10"
dd12 = "Fridge 11"
dd13 = "Fridge 12"
dd14 = "Freezer 18"

date = 'June 19, 2022'

In [23]: print("process Started -----")
process Started -----

In [24]: op = Options()
op.add_argument("--start-maximized") #open Browser in maximized mode
op.add_argument("--no-sandbox") #bypass OS security model
op.add_argument("--disable-dev-shm-usage") #overcome limited resource problems
# op.add_experimental_option('excludeSwitches', ['enable-automation'])
# op.add_experimental_option('useAutomationExtension', False)
op.headless = False # change to true if we want to hide the browser
s = Service(chrome_path)

with webdriver.Chrome(service=s, options=op) as d:
    d.get(url)
    d.find_element(by=By.NAME, value='login-form:j_idt120').send_keys(email)
    d.find_element(by=By.NAME, value='login-form:j_idt125').send_keys(password)
    d.find_element(by=By.ID, value='login-form:j_idt127').click()

```

---

For more information please review the README file