~\Downloads\florence final.py

```python
# Imports and global declarations
import re
import zipfile
import time
import datetime
from zipfile import ZipFile as zipfile
from selenium import webdriver
from selenium.webdriver.common.keys import Keys
from selenium.webdriver.common.by import By
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.support import expected_conditions as EC
from selenium.common.exceptions import StaleElementReferenceException
import time
import csv
import os
import glob
import pandas as pd
from datetime import date
from selenium.webdriver.common.action_chains import ActionChains
import getpass
#need to change login stuff
import warnings

driver = webdriver.Chrome()
driver.maximize_window()
# driver.get("https://login.uatv2.researchbinders.com/u/login/identifier?state=hKFo2SBtZFFQa1FXRVVXNm93U1dFNWxyYTFpX0tGTGxUSkc4c6Fur3VuaXZlcnNhbC1sb2dpbqN0aWTZIFlod2RDbV9rV19ja194VHVNQ1JTSUJZWUFjTTBnejc0o2NpZNkgR0N3Z3ZHamF1YlNrMkRybWJycFpuS3VRVnNkUXhpYVA")
driver.get("https://login.v2.researchbinders.com/u/login/identifier?state=hKFo2SA0Z0ZiNlhfaWxpamwtM0F5TG5nMS1XbGF5S1kySmI2NKFur3VuaXZlcnNhbC1sb2dpbqN0aWTZIGVpNlFhcTN1YWxhb2h1cVZ1OFdNN2VOWlJtNzdvT2pCo2NpZNkgQkJkbHpjR1oxWGZDNmxQeXdVSTFDUUJpWnpFbmJYRk8")
def login(username, password):
    # 1: florence sign in page and process of entering username/password and goes though microsoft login to vumc
    try:
        username_box = WebDriverWait(driver, 1).until(EC.element_to_be_clickable((By.XPATH, '/html/body/div/div/div[1]/main/section/div/div/div/div[1]/div/form/div[1]/div/div/div/input')))
        username_box.send_keys(username)

        next_box = WebDriverWait(driver, 1).until(EC.element_to_be_clickable((By.XPATH, '/html/body/div/div/div[1]/main/section/div/div/div/div[1]/div/form/div[2]/button')))
        next_box.click()
        try:
            time.sleep(1)
            email_box = WebDriverWait(driver, 1).until(EC.element_to_be_clickable((By.XPATH, '/html/body/div/form[1]/div/div/div[2]/div[1]/div/div/div/div/div[1]/div[3]/div/div/div/div[2]/div[2]/div/input[1]')))
            email_box.send_keys(username)

            next_box = WebDriverWait(driver, 1).until(EC.element_to_be_clickable((By.XPATH, '/html/body/div/form[1]/div/div/div[2]/div[1]/div/div/div/div/div[1]/div[3]/div/div/div/div[4]/div/div/div/div/input')))
            next_box.click()
            try:
                time.sleep(1)
                password_box = WebDriverWait(driver, 1).until(EC.element_to_be_clickable((By.XPATH, '/html/body/div/form[1]/div/div/div[2]/div[1]/div/div/div/div/div/div[3]/div/div[2]/div/div[3]/div/div[2]/input')))
                password_box.send_keys(password)

                next_box = WebDriverWait(driver, 1).until(EC.element_to_be_clickable((By.XPATH, '/html/body/div/form[1]/div/div/div[2]/div[1]/div/div/div/div/div/div[3]/div/div[2]/div/div[4]/div[2]/div/div/div/div/input')))
                next_box.click()

                next_box = WebDriverWait(driver, 1).until(EC.element_to_be_clickable((By.XPATH, '/html/body/div/form/div/div/div[2]/div[1]/div/div/div/div/div/div[3]/div/div[2]/div/div[3]/div[2]/div/div/div[2]/input')))
                next_box.click()
                # reportclick = WebDriverWait(driver, 1).until(EC.element_to_be_clickable((By.XPATH,'/html/body/div[1]/app-layout/div[2]/div/binder-index/nav-menu/nav/ul/li[4]/a')))
            except:
                return False
        except:
            return False
    except:
        return False
    return True

# Main program loop
c = False
while not c:
    username = input("Username: ")
    password = getpass.getpass("Password: ")
    #password = input("Password: ")
    c = login(username, password)
    if not c:
        # driver.get("https://login.uatv2.researchbinders.com/u/login/identifier?state=hKFo2SBtZFFQa1FXRVVXNm93U1dFNWxyYTFpX0tGTGxUSkc4c6Fur3VuaXZlcnNhbC1sb2dpbqN0aWTZIFlod2RDbV9rV19ja194VHVNQ1JTSUJZWUFjTTBnejc0o2NpZNkgR0N3Z3ZHamF1YlNrMkRybWJycFpuS3VRVnNkUXhpYVA")
        driver.get("https://login.v2.researchbinders.com/u/login/identifier?state=hKFo2SA0Z0ZiNlhfaWxpamwtM0F5TG5nMS1XbGF5S1kySmI2NKFur3VuaXZlcnNhbC1sb2dpbqN0aWTZIGVpNlFhcTN1YWxhb2h1cVZ1OFdNN2VOWlJtNzdvT2pCo2NpZNkgQkJkbHpjR1oxWGZDNmxQeXdVSTFDUUJpWnpFbmJYRk8")
        print('--Incorrect Username/Password--')
    else:
        break
    print('-------------------------------')

# download report

#template for explicit wait clicks
# reportclick = WebDriverWait(driver, 20).until(EC.element_to_be_clickable((By.XPATH,)))
# reportclick.click()
reportclick = WebDriverWait(driver, 20).until(EC.element_to_be_clickable((By.XPATH,'/html/body/div[1]/app-layout/div[2]/div/binder-index/nav-menu/nav/ul/li[4]/a')))
reportclick.click()

reportclick = WebDriverWait(driver, 20).until(EC.element_to_be_clickable((By.XPATH,'/html/body/div[1]/app-layout/div[2]/div/reports-index/section/div/div/button')))
reportclick.click()
```

```python

reportclick = WebDriverWait(driver, 20).until(EC.element_to_be_clickable((By.XPATH,'/html/body/div[1]/app-layout/div[2]/div/reports-index/section/div/div/ul/li[7]/a')))
reportclick.click()

reportclick = WebDriverWait(driver, 20).until(EC.element_to_be_clickable((By.XPATH,'/html/body/div[1]/app-layout/div[2]/div/reports-index/section/div/div[2]/button/span')))
reportclick.click()

reportclick = WebDriverWait(driver, 20).until(EC.element_to_be_clickable((By.XPATH,'/html/body/div[1]/app-layout/div[2]/div/reports-index/section/div/div[2]/ul/li[1]/span/a')))
reportclick.click()

reportclick = WebDriverWait(driver, 20).until(EC.element_to_be_clickable((By.XPATH,'/html/body/div[1]/app-layout/div[2]/div/reports-index/section/div/div[2]/button')))
reportclick.click()

reportclick = WebDriverWait(driver, 20).until(EC.element_to_be_clickable((By.XPATH,'/html/body/div[1]/app-layout/div[2]/div/reports-index/section/reports-table/section/div/div/div/div[1]/a')))
reportclick.click()

time.sleep(1)

reportclick = WebDriverWait(driver, 20).until(EC.element_to_be_clickable((By.XPATH,'/html/body/div[1]/app-layout/div[2]/div/reports-index/nav-menu/nav/ul/li[6]/a')))
reportclick.click()

wait = WebDriverWait(driver, 360)
wait.until(EC.text_to_be_present_in_element((By.XPATH, '/html/body/div[1]/app-layout/div[2]/div/downloads-index/section/table/tbody/tr[1]/td[4]'), 'Downloaded'))

# box for automation of duplicating into osr holding binder

time.sleep(5)
# Set the path where the CSV files are located
path = "C:\\Users\\cruzrl\\Downloads"  # Subject to change when user is confirmed

# Find the most recently uploaded CSV file in the specified path
file = max(glob.iglob(path + "\\*.csv"), key=os.path.getmtime)

# Print the path of the most recently uploaded CSV file
print(file)

# Cleaning of untouched downloads from Florence

# Read the CSV file into a DataFrame
df2 = pd.read_csv(file)

# Filter the DataFrame to include only rows where the 'Location' column contains 'OSR'
df2 = df2[df2['Location'].str.contains('OSR')]

# Create an empty DataFrame with columns 'Row Labels' and 'Count of Location'
clean = pd.DataFrame(columns=['Row Labels', 'Count of Location'])

# Iterate over the 'Location' column of df2
for i in df2['Location']:
    if i.split('/')[0] != 'OSR Holding':  # Check if the first part of the location is not 'OSR Holding'
        if i.split('/')[0] not in clean['Row Labels'].values:  # Check if the location is not already in 'clean'
            # Append a new row to 'clean' with the location and count as 1
            clean = clean.append({'Row Labels': i.split('/')[0], 'Count of Location': 1}, ignore_index=True)
        else:
            # Increment the count for the existing location in 'clean'
            clean.loc[clean['Row Labels'] == i.split('/')[0], 'Count of Location'] += 1

# Remove rows with 'Row Labels' equal to '1.0 Central Study Documents' from 'clean'
clean = clean[clean['Row Labels'] != '1.0 Central Study Documents']

# Reset the index of 'clean' and drop the 'index' column
clean = clean.reset_index()
clean = clean.drop(['index'], axis=1)

# Separate entries in 'clean' into two DataFrames based on the count value

# Create df0 DataFrame with entries having count <= 100
df0 = clean[clean['Count of Location'] <= 100]

# Create df1 DataFrame with entries having count > 100
df1 = clean[clean['Count of Location'] > 100]

# Reset the index of 'df0' and 'df1' and drop the 'index' column
df0 = df0.reset_index()
df1 = df1.reset_index()
df0 = df0.drop(['index'], axis=1)
df1 = df1.drop(['index'], axis=1)

# Print the 'Row Labels' from 'clean'
for i in clean['Row Labels']:
    print(i)

# Print the length of 'clean' DataFrame
print(len(clean))
warnings.simplefilter(action='ignore', category=FutureWarning)
# go to binders page
homec = WebDriverWait(driver, 20).until(EC.element_to_be_clickable((By.XPATH,'/html/body/div[1]/app-layout/navbar/section/nav/a[1]/img')))
homec.click()

# loops through study names
for i in range(0, len(clean.index)):
    studyName = clean['Row Labels'][i]

    # search study
    search_box = WebDriverWait(driver, 20).until(EC.element_to_be_clickable((By.XPATH,'/html/body/div[1]/app-layout/div[2]/div/binder-index/section/div[2]/table-filter/div/input')))
    # search_box= driver.find_element(By.XPATH,'/html/body/div[1]/app-layout/div[2]/div/binder-index/section/div[2]/table-filter/div/input')
    search_box.send_keys(studyName)

    # click study
```

```python
    study = WebDriverWait(driver, 20).until(EC.element_to_be_clickable((By.XPATH,'/html/body/div[1]/app-layout/div[2]/div/binder-index/section/div[3]/div[2]/md-virtual-repeat-container/div/div[2]/div/div[2]/a')))
    # study=driver.find_element(By.XPATH,'/html/body/div[1]/app-layout/div[2]/div/binder-index/section/div[3]/div[2]/md-virtual-repeat-container/div/div[2]/div/div[2]/a')
    study.click()

    # variables to determine if osr is found and if it can be completed
    skip = False
    checkbox_element = None
    element_found = False
    completed_check = False

    # finds osr folder in study
    while not element_found:
        # Wait for elements matching the CSS selector to become clickable
        elements = WebDriverWait(driver, 20).until(EC.presence_of_all_elements_located((By.CSS_SELECTOR, '.c-infinite-table__row.u-user-select-none.ng-scope.folder-show__item--loaded')))

        # Loop through the elements
        for i in elements:
            matching_elements = i.find_elements(By.CSS_SELECTOR, '.c-infinite-table__cell .u-d-inline-block.u-word-break-break-all[class*="test-itemNameLink-"]')
            for element in matching_elements:
                if element.text == 'OSR' or element.text == 'OSRs':
                    class_name = element.get_attribute('class')
                    alphanumeric_code = re.search(r'test-itemNameLink-(\w+)', class_name).group(1)
                    checkbox = i.find_element(By.CSS_SELECTOR, f'.c-infinite-table__cell .checkbox-icon.far.fa-fw.fa-square.test-itemCheckbox-{alphanumeric_code}')
                    checkbox_element = checkbox
                    # Scroll to the element
                    driver.execute_script("arguments[0].scrollIntoView();", checkbox)

                    # Click the checkbox using ActionChains
                    actions = ActionChains(driver)
                    actions.move_to_element(checkbox).click().perform()

                    # Retrieve the document count
                    document_count_element = i.find_element(By.CSS_SELECTOR, f'.test-counterLabel-{alphanumeric_code}')
                    document_count = document_count_element.text

                    if document_count.isdigit():
                        document_count = int(document_count)
                        print('Checkbox found and clicked for element with text OSR or OSRs and alphanumeric code:', alphanumeric_code)
                        print('Document count:', document_count)
                    else:
                        print('Invalid document count:', document_count)
                    if document_count <= 100:
                        completed_check = True
                    element_found = True
                    break  # Exit the inner loop after finding and clicking the checkbox
            if element_found:
                break  # Exit the outer loop if the desired element is found

        if not element_found:
            # Scroll down using the scroller XPath
            scroll_element = driver.find_element(By.XPATH, '/html/body/div[1]/app-layout/div[2]/div/folder-show/section/div[4]/div[2]/md-virtual-repeat-container/div')
            driver.execute_script("arguments[0].scrollBy(0, 100);", scroll_element)
            print('Element with text OSR or OSRs not found. Scrolling down...')

    action_box = WebDriverWait(driver, 20).until(EC.element_to_be_clickable((By.XPATH, '/html/body/div[1]/app-layout/div[2]/div/folder-show/section/div[2]/div[2]/button')))
    # action_box=driver.find_element(By.XPATH, '/html/body/div[1]/app-layout/div[2]/div/folder-show/section/div[2]/div[2]/button')
    action_box.click()

    duplicate = WebDriverWait(driver, 20).until(EC.element_to_be_clickable((By.XPATH, '/html/body/div[1]/app-layout/div[2]/div/folder-show/section/div[2]/div[2]/ul/li[3]/a')))
    # duplicate=driver.find_element(By.XPATH, '/html/body/div[1]/app-layout/div[2]/div/folder-show/section/div[2]/div[2]/ul/li[3]/a')
    duplicate.click()

    find_study = WebDriverWait(driver, 20).until(EC.element_to_be_clickable((By.XPATH, '/html/body/div[1]/div/div/duplicate-modal-wrapper/duplicate/section/div[1]/virtual-tree/div/div[2]/table-filter/div/input')))
    # find_study = driver.find_element(By.XPATH, '/html/body/div[1]/div/div/duplicate-modal-wrapper/duplicate/section/div[1]/virtual-tree/div/div[2]/table-filter/div/input')
    find_study.send_keys('OSR Holding')

    select_for_copy = WebDriverWait(driver, 20).until(EC.element_to_be_clickable((By.XPATH, '/html/body/div[1]/div/div/duplicate-modal-wrapper/duplicate/section/div[1]/virtual-tree/cdk-virtual-scroll-viewport/div[1]/div[2]/div/div/div')))
    # select_for_copy = driver.find_element(By.XPATH, '/html/body/div[1]/div/div/duplicate-modal-wrapper/duplicate/section/div[1]/virtual-tree/cdk-virtual-scroll-viewport/div[1]/div[2]/div/div/div')
    select_for_copy.click()

    name_copy = WebDriverWait(driver, 20).until(EC.element_to_be_clickable((By.XPATH, '/html/body/div[1]/div/div/duplicate-modal-wrapper/duplicate/section/div[1]/form/div[3]/div[1]/input')))
    # name_copy = driver.find_element(By.XPATH, '/html/body/div[1]/div/div/duplicate-modal-wrapper/duplicate/section/div[1]/form/div[3]/div[1]/input')
    actionChains = ActionChains(driver)
    actionChains.double_click(name_copy).perform()
    name_copy.send_keys(studyName)

    # hit duplicate
    duplicate = WebDriverWait(driver, 20).until(EC.element_to_be_clickable((By.XPATH, '/html/body/div[1]/div/div/duplicate-modal-wrapper/duplicate/section/div[2]/div/button')))
    # duplicate = WebDriverWait(driver, 20).until(EC.element_to_be_clickable((By.XPATH, '/html/body/div[1]/div/div/duplicate-modal-wrapper/duplicate/section/div[2]/button')))
    # duplicate = driver.find_element(By.XPATH, '/html/body/div[1]/div/div/duplicate-modal-wrapper/duplicate/section/div[2]/div/button')
    duplicate.click()

    # process to rename osr to osr + date and create new osr folder if successfully duplicated
    if completed_check:
        time.sleep(20)
        already_complete = False
        element_found = False
        date = date.today()
        while not element_found:
```

```python
            elements = WebDriverWait(driver, 20).until(EC.presence_of_all_elements_located((By.CSS_SELECTOR, '.c-infinite-table__row.u-user-select-none.ng-scope.folder-show__item--loaded')))
            for i in elements:
                matching_elements = i.find_elements(By.CSS_SELECTOR, '.c-infinite-table__cell.u-d-inline-block.u-word-break-break-all[class*="test-itemNameLink-"]')
                for element in matching_elements:
                    if element.text == ("Completed OSR " + date.strftime("%m-%d-%y")):
                        print("OSRs already moved to holding")
                        already_complete = True
                        element_found = True
                        break
                    if element.text == 'OSR' or element.text == 'OSRs':
                        class_name = element.get_attribute('class')
                        alphanumeric_code = re.search(r'test-itemNameLink-(\w+)', class_name).group(1)
                        time.sleep(1)
                        checkbox = i.find_element(By.CSS_SELECTOR, f'.c-infinite-table__cell.checkbox-icon.far.fa-fw.fa-square.test-itemCheckbox-{alphanumeric_code}')
                        checkbox_element = checkbox
                        driver.execute_script("arguments[0].scrollIntoView();", checkbox)
                        actions = ActionChains(driver)
                        actions.move_to_element(checkbox).click().perform()
                        if not already_complete:
                            dropdown = i.find_element(By.CSS_SELECTOR, "div.c-infinite-table__cell.mr-3.dropdown-menu-right-nested.dropdown a.test-actionsLink")
                            driver.execute_script("arguments[0].scrollIntoView();", dropdown)
                            actions = ActionChains(driver)
                            actions.move_to_element(dropdown).click().perform()
                            wait = WebDriverWait(driver, 10)
                            time.sleep(2)

                            ul_element = i.find_element(By.CSS_SELECTOR, 'div.c-infinite-table__cell.mr-3.dropdown-menu-right-nested.dropdown.open ul.dropdown-menu.ng-scope')
                            li_elements = WebDriverWait(ul_element, 10).until(EC.presence_of_all_elements_located((By.CSS_SELECTOR, 'ul.dropdown-menu.ng-scope a.test-editLink.ng-scope')))
                            for li_element in li_elements:
                                try:
                                    print(li_element.text)
                                    if "Rename/Update" in li_element.text:
                                        li_element.click()
                                        break
                                except NoSuchElementException:
                                    pass
                            time.sleep(1)
                            name_copy = WebDriverWait(driver, 20).until(EC.element_to_be_clickable((By.XPATH, '/html/body/modal-container/div/div/folder-form/form/div[1]/div/input')))
                            actionChains = ActionChains(driver)
                            actionChains.double_click(name_copy).perform()
                            name_copy.send_keys("Completed OSR " + date.strftime("%m-%d-%y"))
                            time.sleep(1)
                            save = WebDriverWait(driver, 20).until(EC.element_to_be_clickable((By.XPATH, '/html/body/modal-container/div/div/folder-form/form/div[2]/button[2]')))
                            save.click()
                            time.sleep(3)
                            save = WebDriverWait(driver, 20).until(EC.element_to_be_clickable((By.XPATH, '/html/body/div[1]/app-layout/div[2]/div/folder-
show/section/div[2]/div[1]/button')))
                            save.click()
                            save = WebDriverWait(driver, 20).until(EC.element_to_be_clickable((By.XPATH, '/html/body/div[1]/app-layout/div[2]/div/folder-show/section/div[2]/div[1]/ul/li[1]/a')))
                            save.click()

                            save = WebDriverWait(driver, 20).until(EC.element_to_be_clickable((By.XPATH, '/html/body/modal-container/div/div/folder-form/form/div[1]/div/input')))
                            save.send_keys("OSR")
                            save = WebDriverWait(driver, 20).until(EC.element_to_be_clickable((By.XPATH, '/html/body/modal-container/div/div/folder-form/form/div[2]/button[2]')))
                            save.click()
                            # print("check")
                        element_found = True
                        break
                if element_found:
                    break
            if not element_found:
                scroll_element = driver.find_element(By.XPATH, '/html/body/div[1]/app-layout/div[2]/div/folder-show/section/div[4]/div[2]/md-virtual-repeat-container/div')
                driver.execute_script("arguments[0].scrollBy(0, 100);", scroll_element)
                print('Working...')


        completed_check = False
    time.sleep(1)
    max_retries = 3
    retry_count = 0

    # occasionally the binders tab is clickable but not loaded in DOM so this gives the code 3 chances to click the tab
    while retry_count < max_retries:
        try:
            homec = WebDriverWait(driver, 20).until(EC.element_to_be_clickable((By.XPATH,'/html/body/div[1]/app-layout/div[2]/div/folder-show/nav-menu/nav/ul/li[3]/a')))
            homec.click()
            break  # If click is successful, exit the loop
        except StaleElementReferenceException:
            # Increment retry count
            retry_count += 1
            print(f"StaleElementReferenceException occurred. Retrying ({retry_count}/{max_retries})...")
            continue
        except:
            # Increment retry count
            retry_count += 1
            print(f"Error occurred. Retrying ({retry_count}/{max_retries})...")
            continue
search = WebDriverWait(driver, 20).until(EC.element_to_be_clickable((By.XPATH, '/html/body/div[1]/app-layout/div[2]/div/binder-index/section/div[2]/table-filter/div/input')))
search.send_keys('OSR Holding')

search = WebDriverWait(driver, 20).until(EC.element_to_be_clickable((By.XPATH, '/html/body/div[1]/app-layout/div[2]/div/binder-index/section/div[3]/div[2]/md-virtual-repeat-container/div/div[2]/div/div[1]/i')))
```

```python
search.click()

search = WebDriverWait(driver, 20).until(EC.element_to_be_clickable((By.XPATH, '/html/body/div[1]/app-layout/div[2]/div/binder-index/section/div[2]/div[2]/button')))
search.click()

search = WebDriverWait(driver, 20).until(EC.element_to_be_clickable((By.XPATH, '/html/body/div[1]/app-layout/div[2]/div/binder-index/section/div[2]/div[2]/ul/li[4]/a')))
search.click()

search = WebDriverWait(driver, 20).until(EC.element_to_be_clickable((By.XPATH, '/html/body/div[1]/div/div/binder-download-wrapper/binder-download/section/div[2]/button[2]')))
search.click()

max_retries = 3
retry_count = 0

while retry_count < max_retries:
    try:
        homec = WebDriverWait(driver, 20).until(EC.element_to_be_clickable((By.XPATH,'/html/body/div[1]/app-layout/div[2]/div/binder-index/nav-menu/nav/ul/li[6]/a')))
        homec.click()
        break  # If click is successful, exit the loop
    except StaleElementReferenceException:
        # Increment retry count
        retry_count += 1
        print(f"StaleElementReferenceException occurred. Retrying ({retry_count}/{max_retries})...")
        continue

# code waits up to 900 seconds for OSR Holding to download and will time out if longer (can be changed)
wait = WebDriverWait(driver, 900)
wait.until(EC.text_to_be_present_in_element((By.XPATH, '/html/body/div[1]/app-layout/div[2]/div/downloads-index/section/table/tbody/tr[1]/td[4]'), 'Downloaded'))
time.sleep(1)

# delete from osr holding
# deletes top row until nothing is left

delete = WebDriverWait(driver, 20).until(EC.element_to_be_clickable((By.XPATH, '/html/body/div[1]/app-layout/div[2]/div/downloads-index/nav-menu/nav/ul/li[3]/a')))
delete.click()

delete = WebDriverWait(driver, 20).until(EC.element_to_be_clickable((By.XPATH, '/html/body/div[1]/app-layout/div[2]/div/binder-index/section/div[2]/table-filter/div/input')))
delete.send_keys('OSR Holding')

delete = WebDriverWait(driver, 20).until(EC.element_to_be_clickable((By.XPATH, '/html/body/div[1]/app-layout/div[2]/div/binder-index/section/div[3]/div[2]/md-virtual-repeat-container/div/div[2]/div/div[2]/a')))
delete.click()

time.sleep(1)
c = True
while c:
    try:
        time.sleep(1)
        name = WebDriverWait(driver, 5).until(EC.element_to_be_clickable((By.XPATH, '/html/body/div[1]/app-layout/div[2]/div/folder-show/section/div[4]/div[2]/md-virtual-repeat-container/div/div[2]/div[1]/div[2]/a'))).text
        delete = WebDriverWait(driver, 20).until(EC.element_to_be_clickable((By.XPATH, '/html/body/div[1]/app-layout/div[2]/div/folder-show/section/div[4]/div[2]/md-virtual-repeat-container/div/div[2]/div[1]/div[9]/a')))
        delete.click()

        time.sleep(1)

        delete = WebDriverWait(driver, 20).until(EC.element_to_be_clickable((By.XPATH, '/html/body/div[1]/app-layout/div[2]/div/folder-show/section/div[4]/div[2]/md-virtual-repeat-container/div/div[2]/div[1]/div[9]/ul/li[2]/a')))
        delete.click()

        delete = WebDriverWait(driver, 20).until(EC.element_to_be_clickable((By.XPATH, '/html/body/div[1]/div/div/destroy-folder/section/div[1]/div/div/form/div/input')))
        delete.send_keys(name)

        delete = WebDriverWait(driver, 20).until(EC.element_to_be_clickable((By.XPATH, '/html/body/div[1]/div/div/destroy-folder/section/div[1]/div/div/form/div/textarea')))
        delete.send_keys("post automation deletion")

        delete = WebDriverWait(driver, 20).until(EC.element_to_be_clickable((By.XPATH, '/html/body/div[1]/div/div/destroy-folder/section/div[2]/button[2]/span/span')))
        delete.click()
        time.sleep(1)
    except:
        c = False
# Close the browser
driver.quit()
# comparison from florence and osr binder

with zipfile('C:\\Users\\cruzrl\\Downloads\\OSR Holding.zip', 'r') as zip:
    origFileNames = zip.namelist()
fileNames = []
for string in origFileNames:
    print(string)
    string = string[12:]
    string = string.split('/')[0]
    if string not in fileNames:
        fileNames.append(string)
for i in fileNames:
    print(i)
    if i in df0['Row Labels'].to_numpy():
        if i not in fileNames:
            df1 = df1.append({'Row Labels': i, 'Count of Location': 1}, ignore_index = True)
# for i in df1['Row Labels']:
#     print(i)
with zipfile('C:\\Users\\cruzrl\\Downloads\\OSR Holding.zip', 'r') as zip:
    origFileNames = zip.namelist()
fileNames = []
for string in origFileNames:
    string = string[12:]
    string = string.split('/')[0]
    if string not in fileNames:
        fileNames.append(string)
        # print(string)
over = []  # Create a copy of the 'clean' list



for i in clean['Row Labels']:
    if i not in fileNames:
        # print(i)
        over.append(i)
print(len(over))
print(len(clean))
print(len(fileNames))
# for i in df1['Row Labels']:
#     print(i)
```