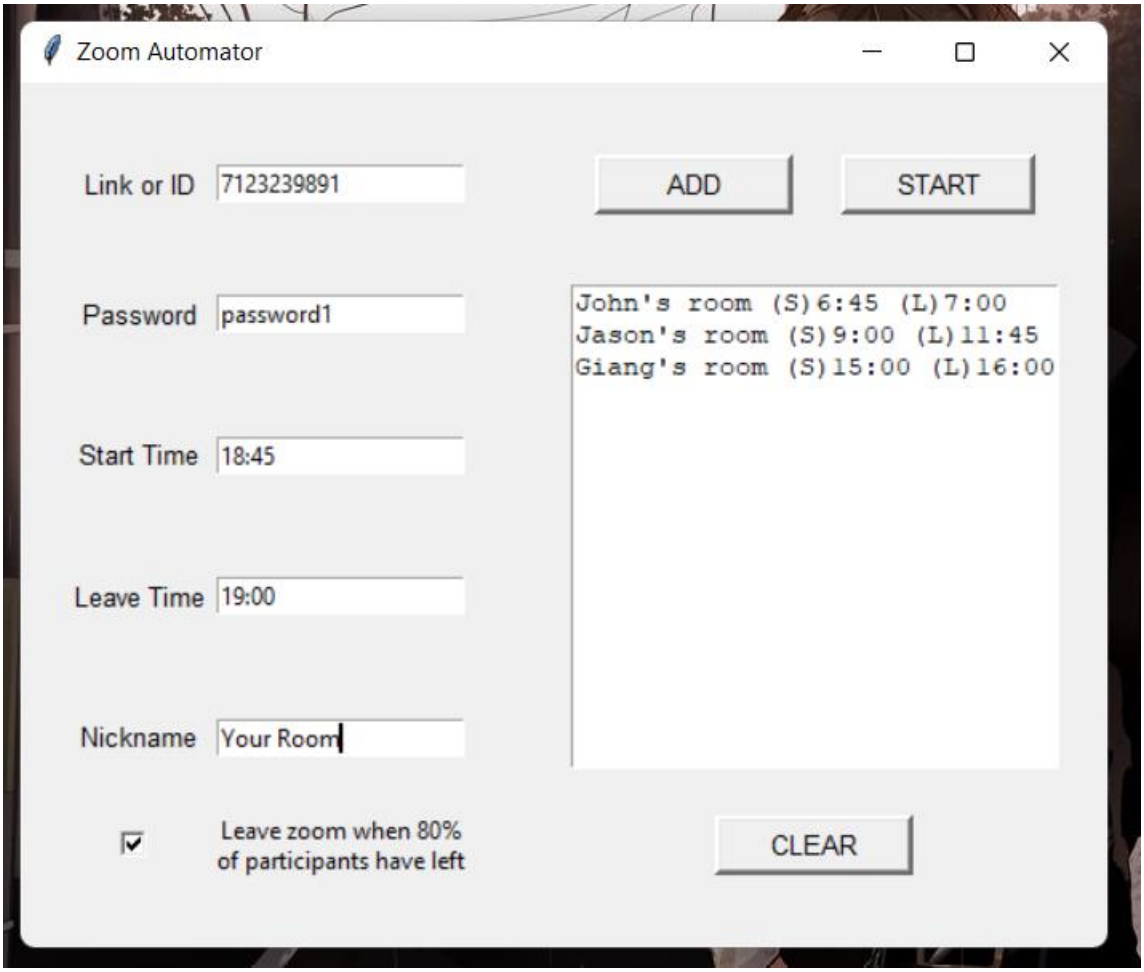
Zoom Automator

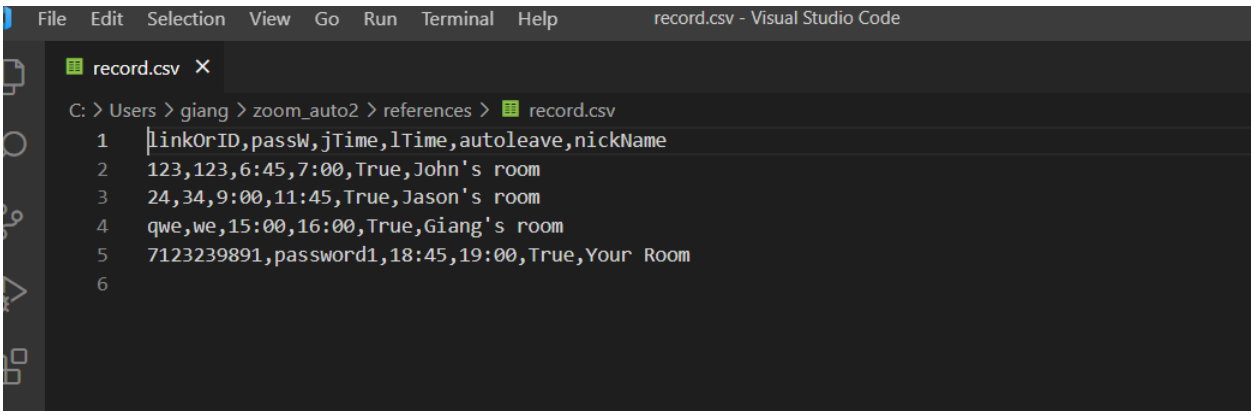
By Giang Pham

The code in ZoomAutoGUI will run first, prompting a simple user interface where users can add their zoom meetings in.



If a meeting is already in place but new meeting is scheduled, the program will automatically stop the current meeting and start the new one.

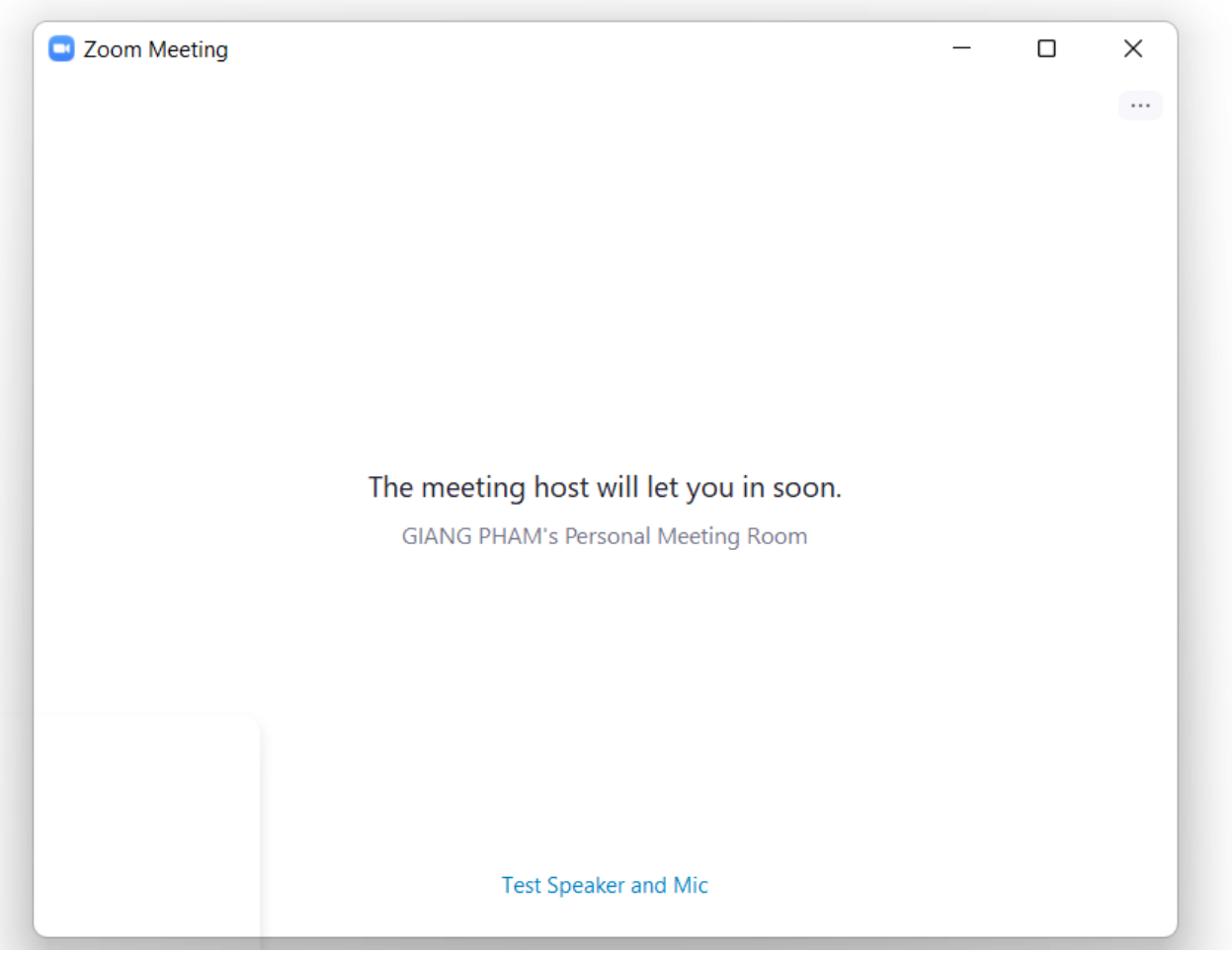
To start the ZoomAuto program, the user simply needs to press “Start”. The code will compile the meeting data into a CSV file. The ZoomAuto.exe will then run.



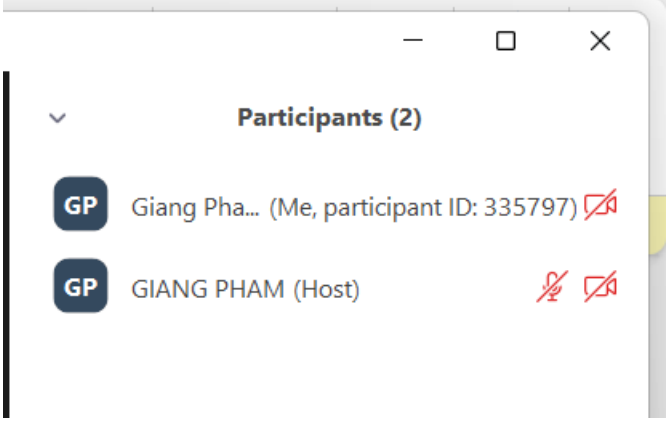
Every 40 seconds, ZoomAuto.exe will check the current time and scan the csv for that time. If there is a match, the code initiate zoom.

```
while True:
    df = pd.read_csv("C:\\Users\\giang\\zoom_auto2\\references\\record.csv")
    now = datetime.datetime.now().strftime("%H:%M")
    if now in str(df['jTime']):
        row = df.loc[df['jTime'] == now]
        linkOrID = str(row.iloc[0,0])
        if str(row.iloc[0,1]) == "nan":
            passw = ""
        else:
            passw = str(row.iloc[0,1])
        jTime = str(row.iloc[0,2])
        lTime = str(row.iloc[0,3])
        if str(row.iloc[0,4]) == "True":
            autoleave = True
        else:
            autoleave = False
        nickName = str(row.iloc[0,5])
        zoomInstance = ZoomObject(linkOrID, passw, jTime, lTime, autoleave, nickName)
        zoomInstance.startAndEnd()
    else:
        t.sleep(40)
```

Using Pyautogui, the code will find the necessary buttons and click them until the user is brought to the zoom waiting room. The program will also automatically disable face cam and computer audio.



Once, the host let the user in, the code will check if the “leave Zoom when 80% of participants have left” is selected. If so, it will open the participant window.



Then, the code will update the maximum number of participants every 40 seconds. If the program detects that x% of max participants have already, it will end zoom.

(Note: on the code, I put threshold as 0.25 instead of 0.2 due to my lack of zoom accounts to test this feature.)

```
            break
    if (self.autoleave):
        threshold = maxNum * .25
        part = pyautogui.locateOnScreen('C:\\Users\\giang\\zoom_auto2\\references\\participant.png', cc
        if (part != None):
            im = pyautoqui.screenshot(reqion=(part[0], part[1], part[2] + 50, part[3]))
            image = numpy.array(im)
            rgb = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)
            options = "outputbase digits"
            pytesseract.pytesseract.tesseract_cmd = r'C:\Program Files\Tesseract-OCR\tesseract.exe'
            text = pytesseract.image_to_string(rgb, config=options)
            num = int(text)
            if (num > maxNum):
                maxNum = num
            elif (num <= threshold):
                break
    t.sleep(40)
os.system("taskkill /im Zoom.exe")
```

This process will continue until ether the user press “clear” on the GUI or the user kills the ZoomAuto.exe. Pressing “clear” will also remove those meetings from the csv file.

Other Notes

- ZoomAutoGUI does not need to be a .exe file, but ZoomAuto does.
- The reason I can’t combine the two is because of threading issues. I tried multiple approaches to try to resolve it, but this is just the cleanest method I could find/do.
- The cv2 is used to add “confidence levels” to image selection. Because otherwise, pyautogui requires the image to match pixel by pixel, which is very inconsistent in practice.
- What I learned from this project: I learned how to use python. Before this, I usually code in C++ or java, but doing this project familiarize me with python as well as concepts on graphical interface and threading.

In [9]:

```
import os
from tkinter import *
import pandas as pd




# UI set up
root = Tk()


root.title("Zoom Automator")
scroll = Scrollbar(root)
canvas = Canvas(root, width = 350, height = 350)
canvas.grid(columnspan = 7, rowspan = 7)
left_margin = Label(root, text = "  ", padx = 7)
left_margin.grid(row = 0, column = 0)
bottom_margin = Label(root, text = "  ", pady = 7)
bottom_margin.grid(row = 7, column = 0)
right_margin = Label(root, text = "  ", padx = 7)
right_margin.grid(row = 0, column = 7)
top_margin = Label(root, text = "  ", padx = 7)
top_margin.grid(row = 0, column = 3)

r = IntVar()
link_id_label = Label(root, text = "Link or ID", font = ("TkDefaultFont", 10))
passw_label = Label(root, text = "Password", font = ("TkDefaultFont", 10))
join_time = Label(root, text = "Start Time", font = ("TkDefaultFont", 10))
leave_time = Label(root, text = "Leave Time", font = ("TkDefaultFont", 10))
nick_name = Label(root, text = "Nickname", font = ("TkDefaultFont", 10))
check_btn = Checkbutton(root, variable=r, onvalue = 1, offvalue = 0)

link_id_label.grid(row = 1, column = 1)
passw_label.grid(row = 2, column = 1)
join_time.grid(row = 3, column = 1)
leave_time.grid(row = 4, column = 1)
nick_name.grid(row = 5, column = 1)
check_btn.grid(row = 6, column = 1)


entry1 = Entry(root, width = 20)
entry1.grid(row = 1, column = 2)
entry2 = Entry(root, width = 20)
entry2.grid(row = 2, column = 2)
entry3 = Entry(root, width = 20)
entry3.grid(row = 3, column = 2)
entry4 = Entry(root, width = 20)
entry4.grid(row = 4, column = 2)
entry5 = Entry(root, width = 20)
entry5.grid(row = 5, column = 2)
check_text = Label(root, text = "Leave zoom when 80%\nof participants have left")
check_text.grid(row = 6, column = 2)

divider = Label(root, text = "  ", padx = 20)
divider.grid(row = 1, column = 3)

text_box = Text(root, height = 15, width = 30, state = DISABLED, pady = 0)
text_box.grid(row = 2, column = 4, rowspan = 4, columnspan = 2, sticky = N+S+E+W, pady = 20)
text_box.configure(yscrollcommand=scroll.set)


# Interaction setup for Add and Clear button
listZoom = []

def clickClear():
    # Clears textbox and exits
    listZoom.clear()
    text_box.config(state = NORMAL)
    text_box.delete("1.0", "end")
    text_box.config(state = DISABLED)

    # Display tutorial
    #... to be implemented

    # Clears CSV
    df = pd.read_csv("C:\\Users\\giang\\zoom_auto2\\references\\record.csv")
    colNames = df.columns
    df_new = pd.DataFrame(data=[], columns=colNames)
    df_new.to_csv("C:\\Users\\giang\\zoom_auto2\\references\\record.csv", index=False)

    # Enabled buttons again
    add_btn.config(state = NORMAL)
    start_btn.config(state = NORMAL)

    #os._exit(1)
    os.system("taskkill /im ZoomAuto.exe")

clear_btn = Button(root, text = "CLEAR", font = ("TkDefaultFont", 10), padx = 22, borderwidth = 3, command = cl
clear_btn.grid(row = 6, column = 4, columnspan = 2, pady = 0)



def clickAdd():
    # Get the entries and create new zoom object
    if (r.get() == 1):
        autoL = "True"
    else:
        autoL = "False"
    print("r = " + str(r))
    newZoom = [entry1.get(), entry2.get(), entry3.get(), entry4.get(), autoL, entry5.get()]

    # Clear textboxes
    entry1.delete(0, END)
    entry2.delete(0, END)
    entry3.delete(0, END)
    entry4.delete(0, END)
    entry5.delete(0, END)


    # Add to listZoom and input data into textbox
    listZoom.append(newZoom)
    text_box.config(state = NORMAL)
    text_box.delete("1.0", "end")
    for i in listZoom:
        text_box.insert(END, i[5] + " (S)" + i[2] + " (L)" + i[3] + "\n")
    text_box.config(state = DISABLED)

add_btn = Button(root, text = "ADD", font = ("TkDefaultFont", 10), padx = 30, borderwidth = 3, command = clickA
add_btn.grid(row = 1, column = 4, pady = 15)


def clickStart():
    # sets add button and start button to disabled
    start_btn.config(state = DISABLED)
    add_btn.config(state = DISABLED)

    # Adds elements of listZoom onto CSV
    for i in listZoom:
        df_old = pd.read_csv("C:\\Users\\giang\\zoom_auto2\\references\\record.csv")
        newData = [[i[0],i[1],i[2],i[3],i[4],i[5]]]
        colNames = df_old.columns
        df_new = pd.DataFrame(data=newData, columns=colNames)
        df_complete = pd.concat([df_old, df_new], axis = 0)
        df_complete.to_csv("C:\\Users\\giang\\zoom_auto2\\references\\record.csv", index=False)
    print("data appended")

    #Start ZoomAuto.exe
    os.startfile("C:\\Users\\giang\\zoom_auto2\\dist\\ZoomAuto\\ZoomAuto.exe")
    print("exe launched")


start_btn = Button(root, text = "START", font = ("TkDefaultFont", 10), padx = 22, borderwidth = 3, command = cl
start_btn.grid(row = 1, column = 5, pady = 0)

root.mainloop()
```

```
r = PY_VAR7
data appended
exe launched
```

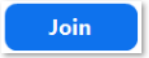
In [ ]:

```
import pyautogui
import datetime
import time as t
import cv2
import os
import webbrowser
import pygetwindow
import numpy
import pytesseract
import pandas as pd

class ZoomObject:

    def __init__(self, linkOrID, passw, jTime, lTime, autoleave, nickName):
        self.linkOrID = linkOrID
        self.passw = passw
        self.jTime = jTime
        self.lTime = lTime
        self.autoleave = autoleave
        self.nickName = nickName
        self.active = True

    def zoomStart(self):
        success = False
        os.system("taskkill /im Zoom.exe")
        clickBtn("C:\\Users\\giang\\zoom_auto2\\references\\leave.png", 3)
        t.sleep(2)
        if ("zoom.us") in self.linkOrID:
            webbrowser.open_new(self.linkOrID)
            if (len(self.passw) > 0):
                found = findImage("C:\\Users\\giang\\zoom_auto2\\references\\join_meeting_greyed.png", 15)
                if (found != None):
                    pyautogui.write(self.passw)
                    pyautogui.press('enter')
            found = findImage('C:\\Users\\giang\\zoom_auto2\\references\\join_no_vid.png', 60)
            if (found != None):
                pyautogui.moveTo(found)
                pyautogui.click()
                success = True
            else:
                quit()
        else:
            with pyautogui.hold('win'):
               pyautogui.press(['m'])
            os.startfile("C:\\Users\\giang\\AppData\\Roaming\\Zoom\\bin\\Zoom.exe")
            findImage('C:\\Users\\giang\\zoom_auto2\\references\\zoom_app.png', 15)
            clickBtn('C:\\Users\\giang\\zoom_auto2\\references\\join_meeting.png', 15)
            found = findImage('C:\\Users\\giang\\zoom_auto2\\references\\meeting_id.png', 15)
            if (found != None):
                pyautogui.moveTo(found)
                pyautogui.click()
                pyautogui.write(self.linkOrID)
            else:
                quit()
            sec = 0
            while True:
                check_btn = pyautogui.locateAllOnScreen('C:\\Users\\giang\\zoom_auto2\\references\\check_btn.pn
                if (check_btn != None):
                    for btn in check_btn:
                        pyautogui.moveTo(btn)
                        pyautogui.click()
                        t.sleep(.005)
                    break
                elif (sec > 15):
                    quit()
                    break
                sec = sec + 1
                time.sleep(1)
            found = findImage('C:\\Users\\giang\\zoom_auto2\\references\\join_btn.png', 15)
            if (found != None):
                pyautogui.moveTo(found)
                pyautogui.click()
                success = True
            else:
                quit()
            found = findImage('C:\\Users\\giang\\zoom_auto2\\references\\password.png', 15)
            if (found != None):
                pyautogui.moveTo(found)
                pyautogui.click()
                pyautogui.write(self.passw)
                pyautogui.press('enter')
        t.sleep(4)
        if (self.autoleave and success):
            sec = 0
            while True:
                waiting_room = pyautogui.locateCenterOnScreen('C:\\Users\\giang\\zoom_auto2\\references\\waitin
                if(waiting_room == None):
                    break
                elif (sec > 200):
                    break
                sec = sec + 1
                t.sleep(1.5)
            t.sleep(3)
            win = pygetwindow.getWindowsWithTitle('Zoom')[0]
            win.activate()
            with pyautogui.hold('alt'):
                pyautogui.press(['u'])
    def zoomEnd(self):
        endTime = datetime.datetime.strptime(self.lTime, '%H:%M')
        if (self.autoleave):
            maxNum = 1
        while True:
            currTime = datetime.datetime.now()
            now = datetime.datetime.now().strftime("%H:%M")
            if (currTime.hour == endTime.hour) and (currTime.minute == endTime.minute):
                break
            setting = True
            df = pd.read_csv("C:\\Users\\giang\\zoom_auto2\\references\\record.csv")
            if (setting and now in str(df['jTime'])):
                row = df.loc[df['jTime'] == now]
                if str(row.iloc[0,2]) != self.jTime:
                    break
            if (self.autoleave):
                threshold = maxNum * .25
                part = pyautogui.locateOnScreen('C:\\Users\\giang\\zoom_auto2\\references\\participant.png', co
                if (part != None):
                    im = pyautogui.screenshot(region=(part[0], part[1], part[2] + 50, part[3]))
                    image = numpy.array(im)
                    rgb = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)
                    options = "outputbase digits"
                    pytesseract.pytesseract.tesseract_cmd = r'C:\Program Files\Tesseract-OCR\tesseract.exe'
                    text = pytesseract.image_to_string(rgb, config=options)
                    num = int(text)
                    if (num > maxNum):
                        maxNum = num
                    elif (num <= threshold):
                        break
            t.sleep(40)
        os.system("taskkill /im Zoom.exe")
        clickBtn('C:\\Users\\giang\\zoom_auto2\\references\\leave.png', 15)
    def startAndEnd(self):
        self.zoomStart()
        self.zoomEnd()
def clickBtn(image, maxSec):
    sec = 0
    while True:
        btn = pyautogui.locateCenterOnScreen(image, confidence = .8)
        if (btn != None):
            pyautogui.moveTo(btn)
            pyautogui.click()
            break
        elif (sec > maxSec):
            break
        sec = sec + 1
        t.sleep(1)
def findImage(image, maxSec):
    sec = 0
    while True:
        imageArea = pyautogui.locateCenterOnScreen(image, confidence = .8)
        if (imageArea != None):
            break
        elif (sec > maxSec):
            break
        sec = sec + 1
        t.sleep(1)
    return imageArea
while True:
    df = pd.read_csv("C:\\Users\\giang\\zoom_auto2\\references\\record.csv")
    now = datetime.datetime.now().strftime("%H:%M")
    if now in str(df['jTime']):
        row = df.loc[df['jTime'] == now]
        linkOrID = str(row.iloc[0,0])
        if str(row.iloc[0,1]) == "nan":
            passw = ""
        else:
            passw = str(row.iloc[0,1])
        jTime = str(row.iloc[0,2])
        lTime = str(row.iloc[0,3])
        if str(row.iloc[0,4]) == "True":
            autoleave = True
        else:
            autoleave = False
        nickName = str(row.iloc[0,5])
        zoomInstance = ZoomObject(linkOrID, passw, jTime, lTime, autoleave, nickName)
        zoomInstance.startAndEnd()
    else:
        t.sleep(40)
```

This PC › OS (C:) › Users › giang › zoom_auto2 › references

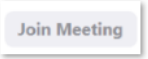
Search references


check_btn

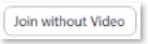


join_btn

join_meeting

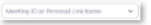


join_meeting_greyed

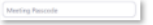


join_no_vid

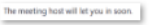
leave


meeting_id

participant


password


record


waiting_room

zoom_app