LOVELY
PROFESSIONAL
UNIVERSITY

MYCLASS PLATFORM AUTOMATION

As a project for course

PYTHON PROGRAMMING(INT213)

Name: Yaswanth Bolisetty

Registration Number: 12009486

Name: Kumar Ashish

Registration Number:12010912

Program: B. Tech Computer Science Engineering (CSE)

Semester: Third

Name of the university: Lovely Professional University

Date of Submission:20th November 2021

MYCLASS PLATFORM AUTOMATION

ABSTRACT:

Students like us need to attend and listen to classes regularly because this allows us to learn new skills, improve them every day and build up our foundation for learning new ones. Even though missing classes is not good, there are times when we must take time away from classes due to a busy schedule or an emergency.

In that case, this myclass platform automation project automatically attends your class for you and ensures that you do not lose your attendance even in emergencies.

ACKNOWLEDGEMNT:

It has been a pleasure working on this project under the guidance of Prof. Sagar Pande.

I must also thank my parents and friends for their immense support and help during this project. Without their help, completing this project would have been very difficult.

Contents of Report

Introduction

Context:

Under the supervision of Sagar Pande, I have done this project as part of my Computer Science Engineering (CSE) course at Lovely Professional University. I have two months to complete the requirements to pass the module.

Motivations:

Because we were interested in automation, we chose a project that included automation. Then we began looking for project ideas involving automation, but we couldn't decide. Then one day, I received a note from one of my juniors, who was concerned about his low attendance. He was unable to attend his classes because he was relocating to a different state. This gave me an idea: if I automate these online classes, they will be useful in some time-sensitive scenarios.

Idea:

We wanted to create a unique and exciting project that included Python automation.

The idea was simple: once the code is executed, the myclass platform will automatically attend the classes. The code is provided with timetable. It checks the system time, joins the session according to the timetable provided, and leaves when the teacher ends the session.

Team Members:

Team Leader:

Yaswanth Bolisetty

Contributions:

1.Selenium

2.WebDriver related classes

3.Login function

4.Discordwebhook

5.Report

Team Member:

Kumar Ashish:

Contributions:

1.Selenium

2.Date and Time related classes

3. Apscheduler

4.Join Function

5.Report

Libraries Used and why they are used:

Selenium:

Selenium is an open-source web-based automation tool. Python language is used with Selenium for testing. It has far less verbose and easy to use than any other programming language. ... Selenium can send the standard Python commands to different browsers, despite variation in their browser's design.

Time:

The Python time module provides many ways of representing time in code, such as objects, numbers, and strings. It also provides functionality other than representing time, like waiting during code execution and measuring the efficiency of your code.

Calendar:

Python defines an inbuilt module calendar that handles operations related to the calendar.
The calendar module allows output calendars like the program and provides additional useful functions related to the calendar. Functions and classes defined in the Calendar module use an idealized calendar, the current Gregorian calendar extended in in both directions. By default, these calendars have Monday as the first day of the week, and Sunday as the last (the European convention).

Datetime:

In Python, date and time are not a data type of their own, but a module named datetime can be imported to work with the date as well as time. Python Datetime module comes built into Python, so there is no need to install it externally.
Python Datetime module supplies classes to work with date and time. These classes provide a number of functions to deal with dates, times, and time intervals. Date and datetime are an object in Python, so when you manipulate them, you are manipulating objects and not string or timestamps.

APScheduler:

Advanced Python Scheduler (APScheduler) is a Python library that lets you schedule your Python code to be executed later, either just once or periodically. ... That said, APScheduler does provide some building blocks for you to build a scheduler service or to run a dedicated scheduler process.

Discord:

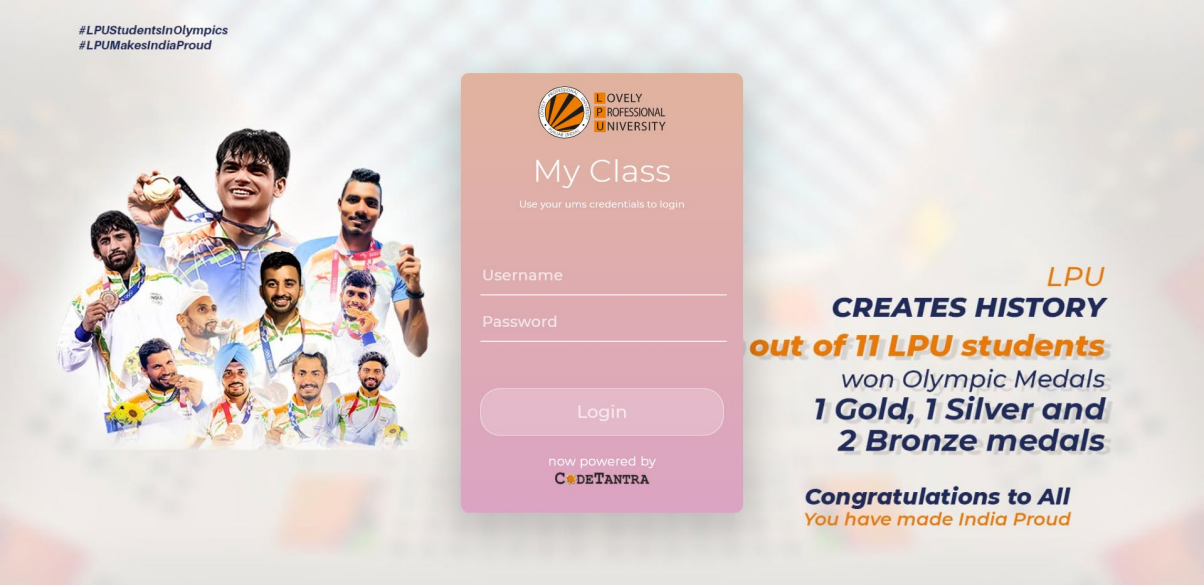
Discord module is made to make bots very simple. You can send messages, create channels and everything else you need. If you need something we doesn't have, create an issue, and let us know.

Screenshots:

Interface of the working:

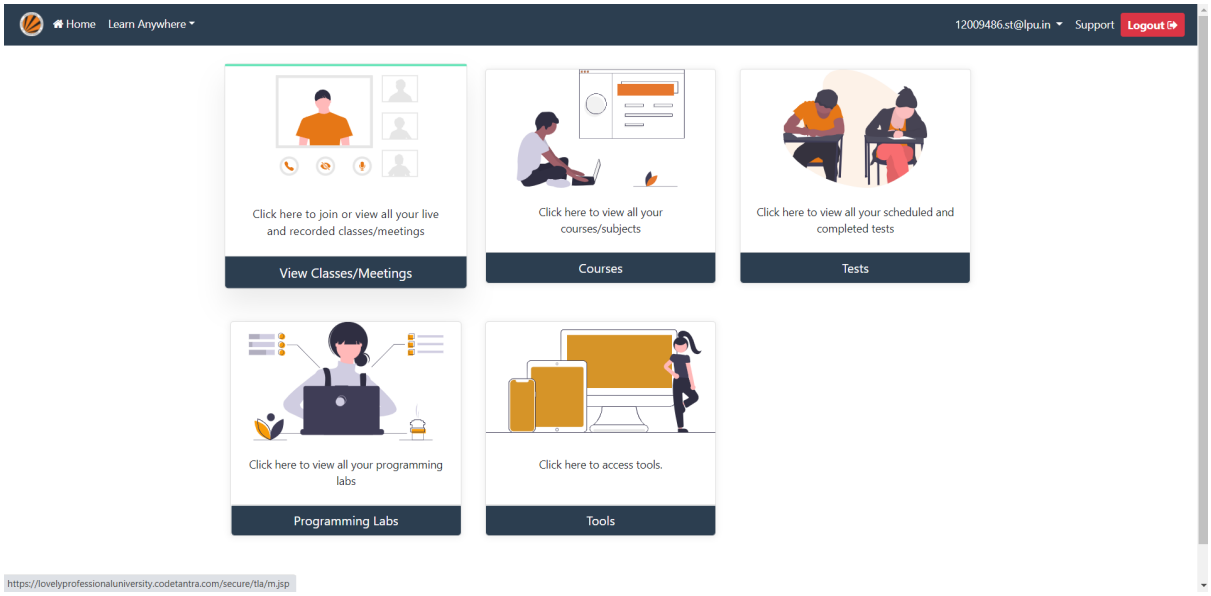
Once its time for the first class to start this process starts executing.

1.Opens the My Class website:

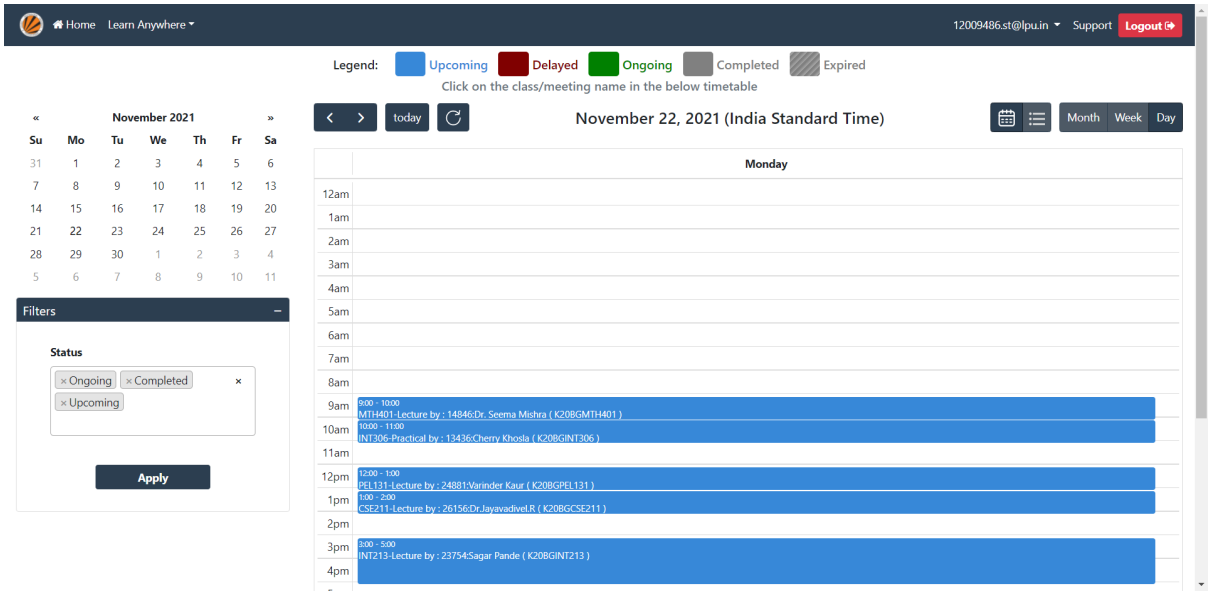


2.Enters the login credentials and clicks login:

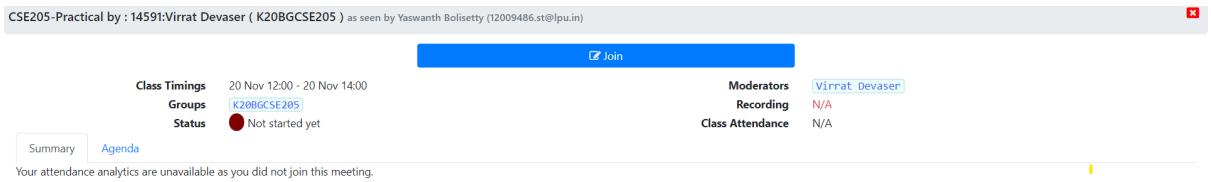
3. Clicks View Classes Card-Container:



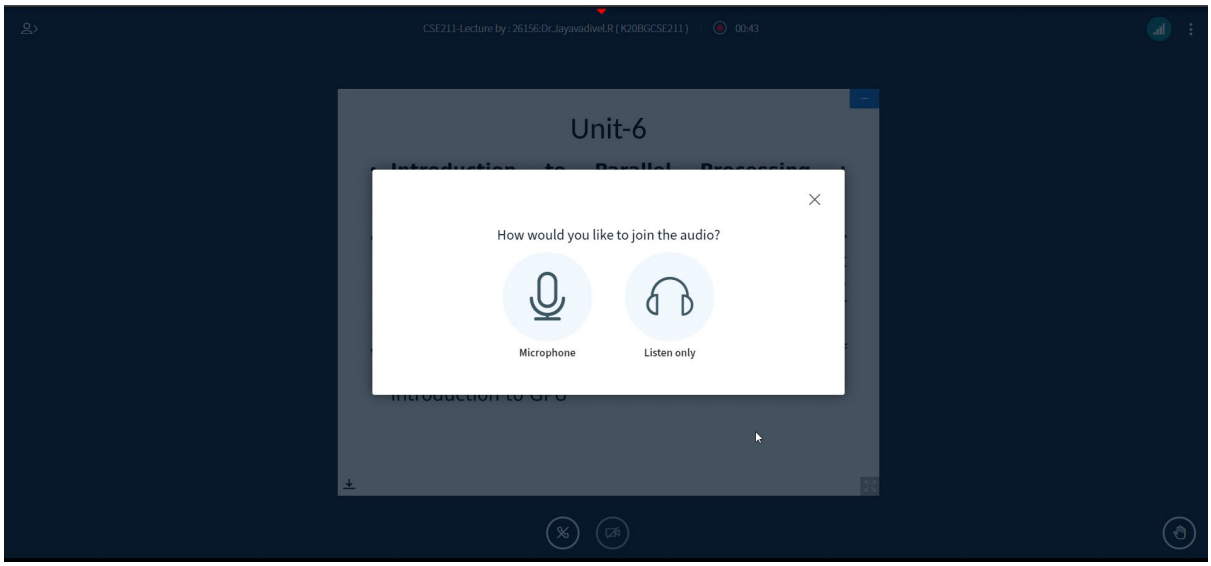
4.Selects the First class to attend:



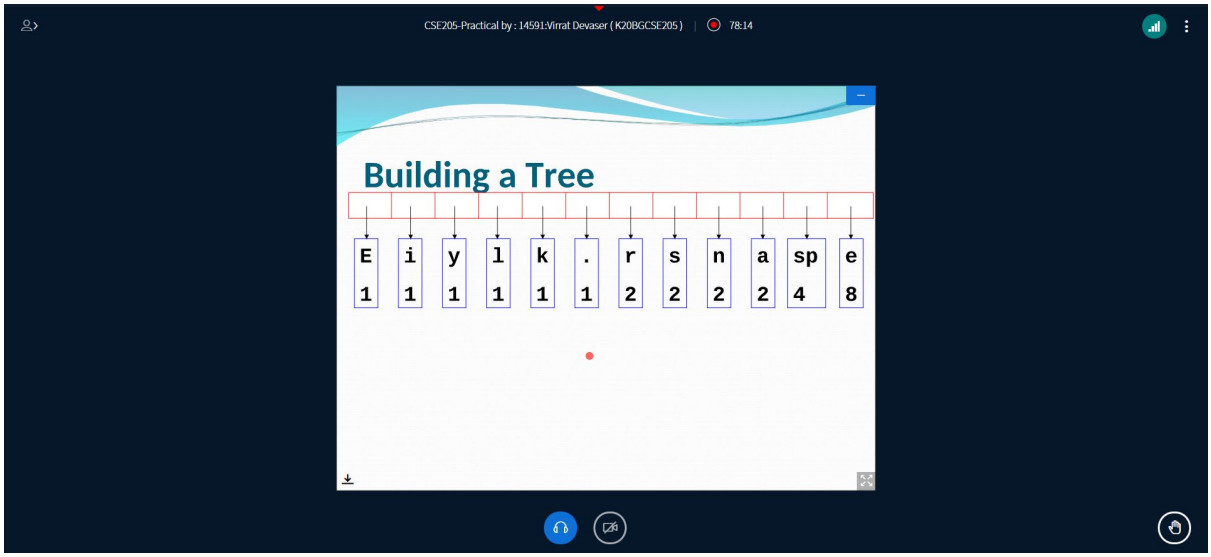
5.Clicks the Join Button:



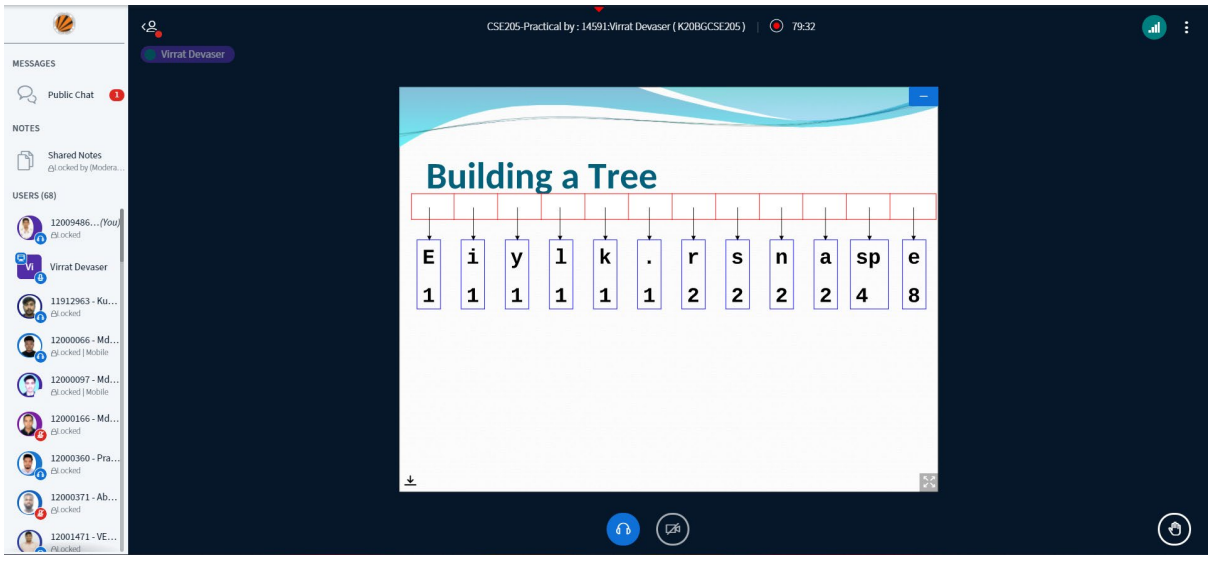
6. Clicks the Listen only Button:



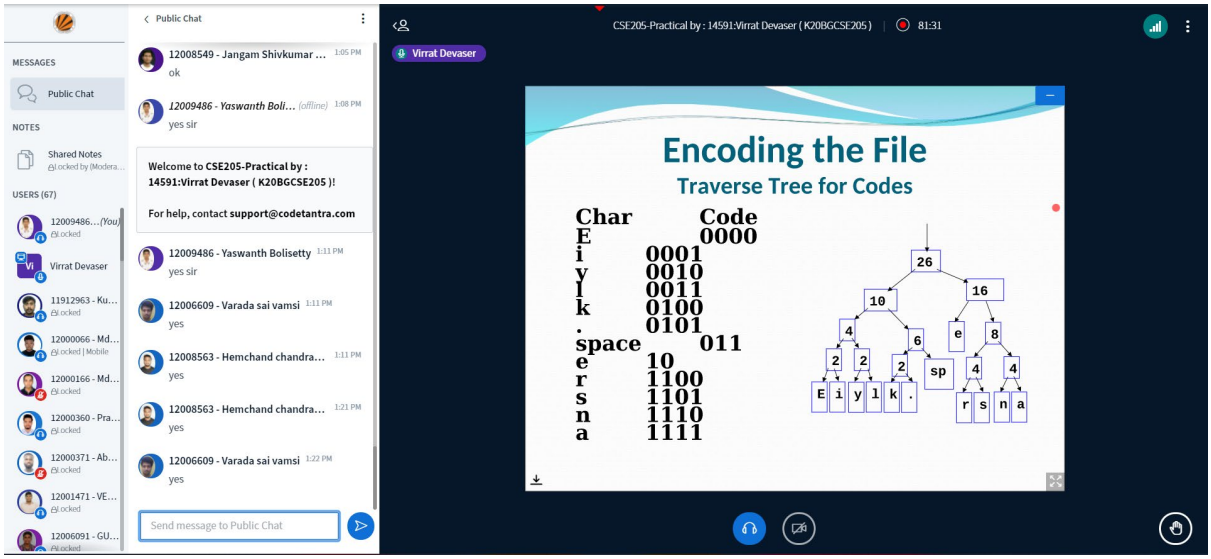
7.Once u click this public chat icon which is on the top left corner:



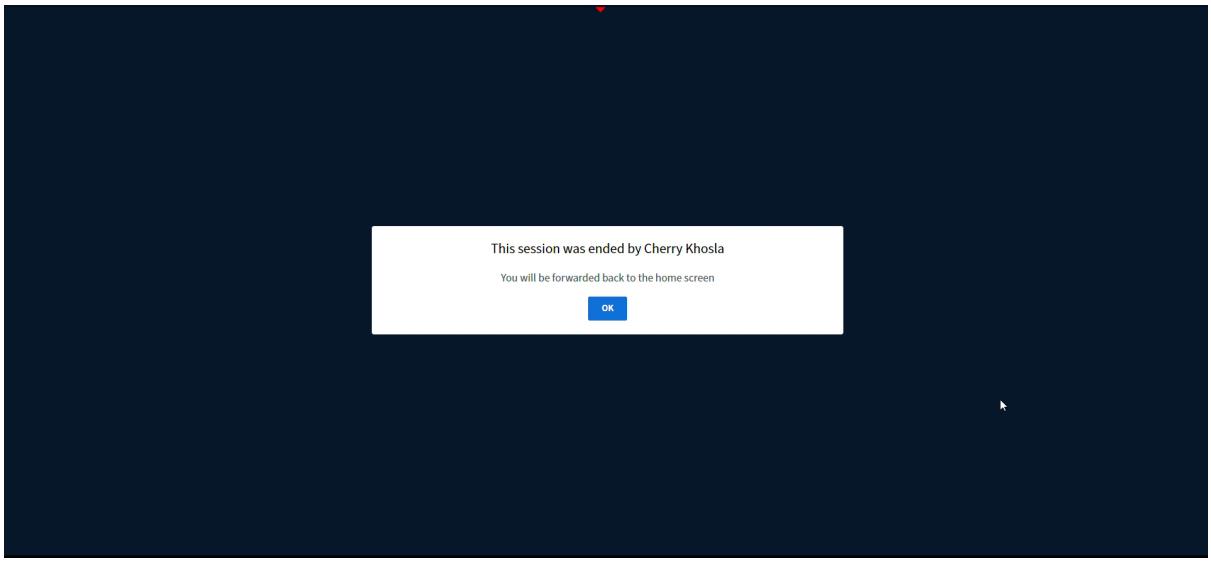
8. This opens page opens, then it automatically clicks public chat container :



9. Then this page appears, it will send “Good morning” or “Good afternoon” message according to the time:



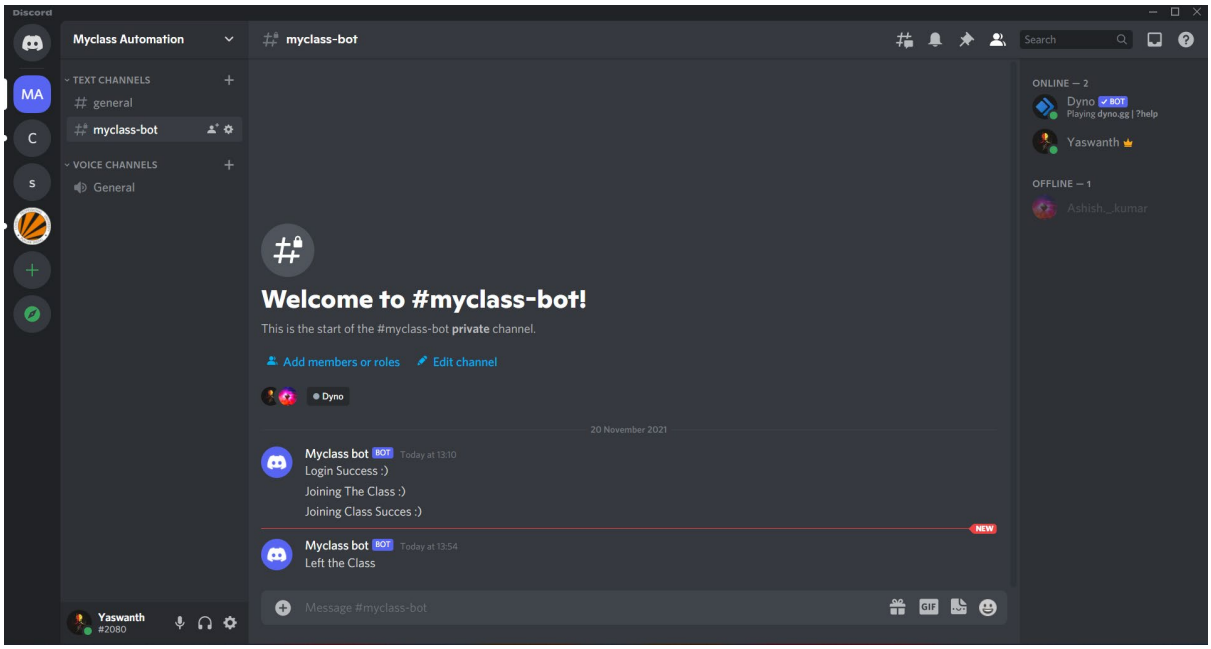
10. Once the teacher ends the meeting, this page appears then it closes the chrome window:



After this process it will wait for next class to start once it starts the same process is repeated but this time it selects the second class as its time for second class and next time third class and this process continues up to the last class present in the timetable.

Discord Notifications:

It notifies the actions done by it:



Explanation of Code:

1. Importing of the modules and libraries used:

```
import time
import calendar
from datetime import datetime,date
from selenium import webdriver
from selenium.webdriver.common.by import By
from selenium.webdriver.common.keys import Keys
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.support import expected_conditions as EC
from apscheduler.schedulers.blocking import BlockingScheduler
from discord import Webhook,RequestsWebhookAdapter
```

2. Declared these as Global Variables so that all the functions can access:

```
webhook =
Webhook.from_url("https://discord.com/api/webhooks/910720477427798026/3KAIFgmj
pKP5tWp9QZyOSNF2J5OQQu0-JpIIcbvatiRevLqXQQ5wu4-y3si6l4Z_I71u",
adapter=RequestsWebhookAdapter())
#webhook link of a discord channel
datetime_now=datetime.now()
#gets the date and time of current day
```

3.Login Function:

```
def login(Id):
    web=webdriver.Chrome()#chrome webdriver
    web.get("https://myclass.lpu.in")#opens myclass website
    time.sleep(1)#waits 1 sec
    web.maximize_window()#maximizes the chrome window
    username=web.find_element_by_xpath('/html/body/div[2]/div/form/div[6]/inpu
t[1]')#locates username field
    username.send_keys("Your registration number here")#enters registration
number  in username field
    time.sleep(1)#waits 1 sec
    password=web.find_element_by_xpath('/html/body/div[2]/div/form/div[6]/inpu
t[2]')#locates password field
    password.send_keys("Your password here")#enters password in password field
    time.sleep(1)#waits 1 sec
    login=web.find_element_by_xpath("/html/body/div[2]/div/form/div[7]/button"
)#locates login button
```

```
    login.click()#clicks login button
    webhook.send("Login Success :)")#sends "Login Success :)" notification to
discord channel
    time.sleep(1)#waits 1 sec
    View_classes=web.find_element_by_xpath("/html/body/div[9]/div/div[1]/div/d
iv/div[1]/div/div[2]/a")
    #locates view_classes container
    View_classes.click()#Clicks View Classees Button
    time.sleep(1)#waits 1 sec
    class_elements=web.find_element_by_xpath("/html/body/div[1]/div[6]/div[3]/
div[2]/div/div[2]/div/table/tbody/tr/td/div/div/div[3]/table/tbody/tr/td[2]/di
v/div[2]/a[{}]".format(Id))
    #Locates the class container respectively according to time
    class_elements.click()#clicks that class container
    time.sleep(1)#waits 1 sec
    try:
        join_button=WebDriverWait(web,150).until(EC.presence_of_element_locate
d((By.XPATH,'//*[@id="meetingSummary"]/div/div/a/i')))
        #waits for 150 sec until the join button appears
        join_button.click()#clicks the join button once it appears
        webhook.send("Joining The Class :)")#sends "Joining The Class :)"
notification to discord channel
    except:
        webhook.send("Joining class failed :(")#if the join button does'nt
appear in 150sec then
        #sends "Joining class failed :(" notification to discord channel
    try:
        time.sleep(5)#waits 5 sec
        web.switch_to.frame(web.find_element_by_id("frame"))#finds the frame
by id and switch to it
        listen_only_button=WebDriverWait(web,150).until(EC.presence_of_element
_located((By.XPATH,'/html/body/div[2]/div/div/div[1]/div/div/span/button[2]/sp
an[1]/i')))
        #waits for 150 sec until the listen_only button appears
        listen_only_button.click()#clicks listen_only button once it appears
        webhook.send("Joining Class Succes :)")#sends "Joining Class Succes
:)" notification to discord channel
    except:
        webhook.send("Not able to Join class :(")#if the listen_only button
does'nt appear in 150sec then
        #sends "Not able to Join class :(" notification to discord channel
    #chatbox-----------------------------------------------------------------
-----------------------------------
    time_now=datetime_now.strftime("%H:%M:%S")#gets current time in
hours,minutes,seconds format
    if time_now>="12:00:00":#checks if the current time is greater or equal to
12:00:00
        chat=("Good Afternoon")#if true then "Good Afternoon" value is given
to chat variable
```

```
    else:
        chat=("Good Morning")#if false then "Good Morning" value is given to
chat variable
    try:
        public_chat=WebDriverWait(web,60).until(EC.presence_of_element_located
((By.ID,'chat-toggle-button')))
        #waits for 30 sec until the public-chat container appears
        public_chat.click()#clicks public-chat container once it appears
    except :#if the public-chat container does'nt appear in 60sec then
            pass#nothing happends
    try:
        greeting=WebDriverWait(web,60).until(EC.presence_of_element_located((B
y.ID,'message-input')))
        #waits for 30 sec until the chat-box appears
        greeting.send_keys(chat)#enters value present in chat variable in
chat-box
        greeting.send_keys(Keys.RETURN)#presses enter key
    except :#if the chat-box does'nt appear in 60sec then
            pass#nothing happends
    #chatbox-----------------------------------------------------------------
-----------------------------------
    try:
        class_ended_button=WebDriverWait(web,7200).until(EC.presence_of_elemen
t_located((By.XPATH,'//*[@id="app"]/div/div/div/div/button/span')))
        #waits for 7200 sec until the class-ended by teacher page appears
        class_ended_button.click()#clicks ok
        web.close()#closes the chrome window
        webhook.send("Left the Class")#sends "Left the Class" notification to
discord channel
    except:#if the class-ended by teacher page does'nt appear in 7200sec then
        pass#nothing happends
    time.sleep(5)#waits 5 sec
```

4. Exits the program if time is greater at 5:00pm:

```
total_time_now=datetime_now.strftime("%H")#gets the hour value from current
time
if total_time_now>"17":#checks if the current time hour value is greater 17
    quit()#if it is true then exits the program
```

5. Join Function:

```
def join(WEEK):
    sched=BlockingScheduler()#BlockingScheduler
    Year,Month,Day=datetime_now.year,datetime_now.month,datetime_now.day
    #Year variable is assigned with current year,
    #Month variable is assigned with current Month,
    #Day variable is assigned with current Day.
    for i,Id in zip(WEEK.values(),WEEK.keys()):
        #i variable is assigned with values of dictionary which matches with
current weekday
        #Id variable is assigned with keys of dictionary which matches with
current weekday
        print(i,Id)#prints i and Id
        x=i.split(":")#splits dictionary values
        sched.add_job(login,run_date=datetime(Year, Month, Day, int(x[0]),
int(x[1]), int(x[2])),args=Id)
        #does this job at specific time mentioned in the dictionary
    sched.start()#starts the BlockingScheduler
```

6. Timetable of the whole week is Inputted as dictionary:

```
#Time Table entered as dictionary-----------------------------------------------
----
MON  =
{"1":"9:00:00","2":"10:00:00","3":"12:00:00","4":"13:00:00","5":"15:00:00"}
TUE  =
{"1":"9:00:00","2":"12:00:00","3":"13:00:00","4":"15:00:00","5":"16:00:00"}
WED  = {"1":"3:00:00"}
THUR =
{"1":"9:15:00","2":"12:00:00","3":"13:00:00","4":"15:00:00","5":"16:00:00"}
FRI  = {"1":"9:00:00","2":"12:00:00","3":"13:00:00","4":"15:00:00"}
SAT  = {"1":"12:00:00","2":"15:00:00"}
#Time Table entered as dictionary-----------------------------------------------
----
```

7. Checks the current weekday and assigns the corresponding dictionary to the function join:

```
week_day=calendar.day_name[datetime_now.weekday()]#gets the weekday name
#Assigns the dictionary according to current weekday
if week_day=="Monday":
    join(MON)
elif week_day=="Tuesday":
    join(TUE)
elif week_day=="Wednesday":
    join(WED)
elif week_day=="Thursday":
    join(THUR)
elif week_day=="Friday":
    join(FRI)
elif week_day=="Saturday":
    join(SAT)
else:
    print("Enjoy Your Sunday")
```

References:

https://selenium-python.readthedocs.io/

https://www.youtube.com/

https://pypi.org/

https://www.geeksforgeeks.org/

https://docs.python.org/3/library/

https://stackoverflow.com/