

MAJOR PROJECT

Auto Attendance using Face Recognition



**GURU GOBIND SINGH INDRAPIRASTHA UNIVERSITY
UNIVERSITY SCHOOL OF INFORMATION
COMMUNICATION AND TECHNOLOGY**

Under the supervision of

Mentor:

Prof. Navin Rajpal

Submitted by:

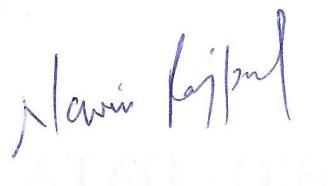
Vikrant Singh Jamwal

B.TECH. – IT 8th Sem

03116401517

CERTIFICATE

This is to certify that the project entitled "**Auto attendance using Face Recognition**" submitted by "**Vikrant Singh Jamwal**" to University School of Information, Communication and Technology, GGSIPU in partial fulfilment of the requirement for the award of the degree of B. Tech in Information Technology. This project work carried out by him under my guidance. The project fulfils the requirement as per the regulation of this institute and in my opinion meets the necessary standards for submission. The contents of this report have not been submitted and will not be submitted either in part or in full, for the award of any other degree or diploma in this institute or any other institute or university to the best of my knowledge and belief.

A handwritten signature in blue ink, appearing to read "Navin Rajpal". Below the signature, there is a faint, illegible printed name, possibly "GGSIPU".

Date: 02/07/2021

Prof. Navin Rajpal
USICT, GGSIPU
New Delhi

DECLARATION

This is to certify that the work being presented in the project entitled "**Auto attendance using Face Recognition**" submitted by undersigned student of Bachelors in **Information Technology** to University School of Information Communication and Technology in the fulfilment for award of the degree of Bachelors in **Information Technology** is a record of my own work carried out by me under guidance and supervision of **Professor Navin Rajpal**. I certify the content of this work has not submitted elsewhere for award of any other degree.

Date: 30/06/2021



Signature of Student

Vikrant Singh Jamwal

B.Tech.-IT,8th semester

(03116401517)

ACKNOWLEDGMENT

I would like to express my sincere gratitude towards **Prof. Pravin Chandra**, Dean, University School of Information & Technology, GGSIPU for giving me this opportunity of working on this project and **Dr. Sanjay Kumar Malik**, coordinator, who provided a wide vision of scope of the project which was very important to me.

I am also thankful to my mentor, **Prof. Navin Rajpal** for his most valuable and significant guidance throughout the course of the project and in elaborating our view of studying the project's details and getting the right vision for its implementation.

At last, I would like to thank the Department of Information Technology - USICT, GGSIPU, that has played an important role in strengthening my career by providing full cooperation and encouragement throughout the tenure of the project.

Contents

S. no.	Topics	Page no.
1	Certificate	2
2	Declaration	3
3	Acknowledgement	4
4	Introduction	6
5	Problem statement	9
6	Idea Behind the Project	10
7	Flow Chart	14
8	Use Case	15
9	Requirements	16
	• Hardware	16
	• Software	16
	• Libraries	16
10	Procedures, Codes and UI	17
	• Login	18
	• Registration	21
	• Detect	24
	• Train	27
	• Update Excel	29
	• View Excel	31
11	References	33

Introduction

Attendance management system is a necessary tool for taking attendance in any environment where attendance plays an important part. However, most of the ways of measuring attendance is time consuming and inefficient. Some institutions use paper based approach and others have adopted automated methods such as fingerprint biometric techniques. This research is aimed at developing a less intrusive, more efficient and cost effective automatic attendance system using face detection and recognition.

As the technology is significantly growing every second, this project aims at updating the old system of recording the attendance to the more efficient way by detection and recognition of the face and creating a complete list of all the people present at the event.

Face recognition technology has also been used for both verifying and identifying the students in a classroom. This research is aimed at developing a less intrusive, cost effective and more efficient automated attendance management system.

Problem Statement

Attendance marking in a classroom during a lecture is not only a difficult job but also time consuming. Due to high number of students present during the lecture there will always be a probability of proxy or missed attendance. Marking the attendance with conventional methods has been a challenge for teachers. The need of saving time and creating an automatic techniques of marking attendance is very important for the future of efficient work.

In recent years, the problem of managing the attendance manually is being tackled by new technology including bio metric checks or fingerprints. However, these techniques lack in reliability. In this project an automatic attendance marking and management system is to be created using face detection and recognition.

Idea behind the project

Main idea behind the project is divided into four parts:

1. Video capture:

The video of the class can be captured from the platform where the meeting is occurring. This video is then moved to the detection folder inside the software.

OpenCV provides the `VideoCapture()` function which is used to work with the Camera or by using a file from storage by specifying the index. (0) is for camera and (1) is for a file path in the storage.

First we have to import `cv2` library to get the window capture function in our program. To get this we have to install OpenCV in our python files.

We can do the following task:

-Read video, display video, and save video.

Read (`ret, frame = vid.read()`) - reads the video present the storage.

Display (`cv2.imshow()`) - displays the video by opening it in a window.

Save (`cap.release()`) - releases the video and saves it.

-Capture from the camera and display it.

```
vid = cv2.VideoCapture(0)
```

This line of code is used to open camera and view the vision in a small window.

2. Face Detection:

First the face is detected using Haar-Cascade and OpenCV by detecting the smile, eyes and nose.

Haar Cascade algorithm for detection is one of the old but efficient face detection algorithms present with us. Haar-Cascade Features were used to detect faces, nose, eyes and also lips.

Basic idea of this algorithm is to get the sum of all the pixels in an image lying on the dark area and the sum of all the pixels of an image lying on the light area of the Haar feature.

Haar Cascade is present in the CV2 folder in the python directory and can be imported just by specifying the absolute path of the file in the cascade function.

```
-cv2.CascadeClassifier('haarcascade_frontalface_default.xml')
```

This line of code classifies the cascade in the program.

The Algorithm looks for specific Haar feature of a face. This detection feature takes the image and converts it into a 24X24 window and puts each Haar feature to that window pixel by pixel. First, the algorithm requires a lot of positive images or we can say image of a face and negative images or image excluding the face to train the classifier. Then, these features are being extracted.

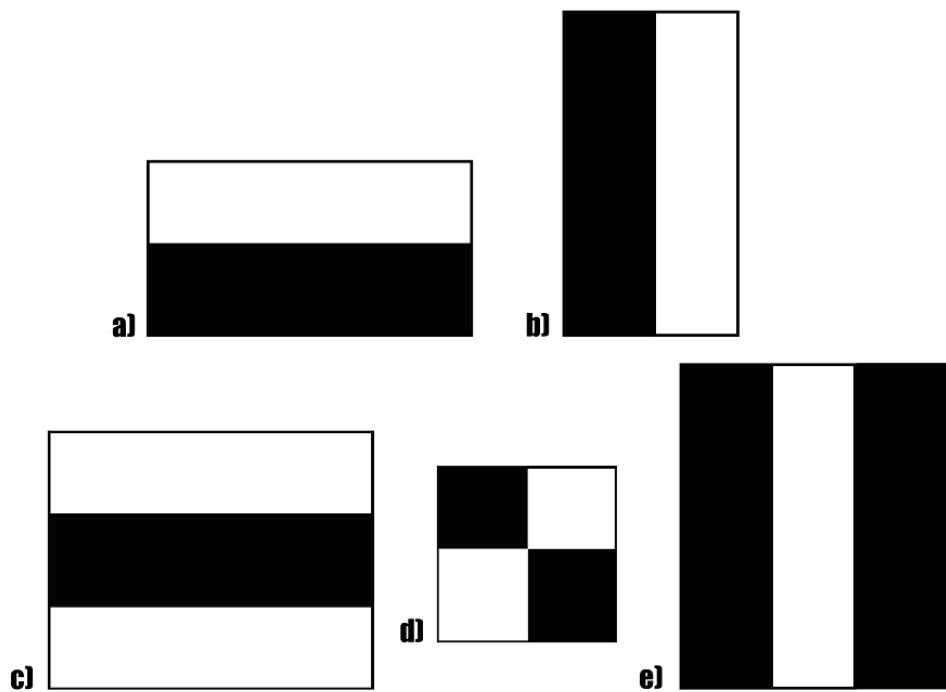


Fig.1 a) and b) are edge features c) and e) are line features and d) is centre surround feature.

Features are numerical values determined from images that are used to differentiate between different images, every feature is single valued by calculating the difference between the sum of white area and sum of black area in the haar cascade triangle.

$$\text{Feature} = \Sigma (\text{pixels in black part}) - \Sigma (\text{pixels in white part}).$$

This formula is used to find out the face features in numbers and these number are to be stored somewhere to recognise the faces later by comparing them to these numbers. Hence these numbers are saved in a YML document with a specific ID for each face.

3. Face recognition:

After detection the face is then compared to the images present in the database. Multiple frames of the captured videos are used to compare with the images of the students in the database.

Local Binary Pattern (LBP) is an effective texture based operator which give the threshold value of the neighbouring pixels and consider the result as a binary number.

It was found to be a powerful feature for texture classification till date. When LBP(Local binary pattern) is combined with Histogram of oriented gradients, it improves the detection considerably on some data.

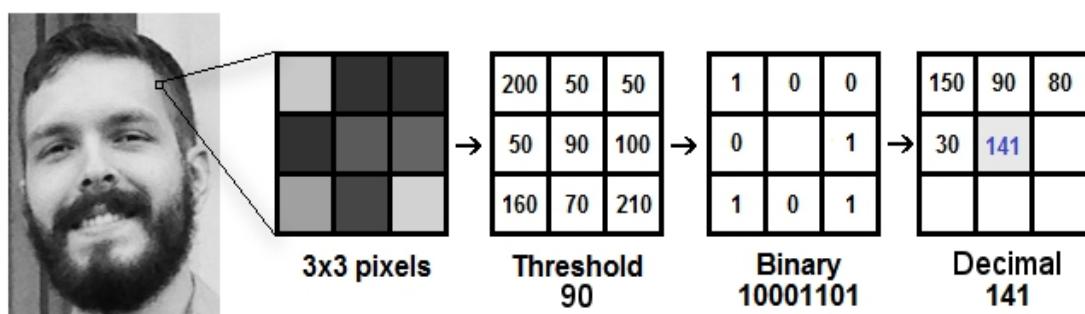


Fig.2 Threshold of neighbouring pixels in LBP

It can easily be available in the OpenCV version 2 and above and can be accessed using a simple line of code.

```
-cv2.face.LBPHFaceRecognizer_create()
```

The LBPH uses 4 parameters:

- 1) Radius: radius is used to build the local binary pattern and represents the radius around the middle pixel. It is mostly set to 1.
- 2) Neighbours: the number of points to build the local binary pattern and is mostly set to 8.
- 3) Grid X: It is the number of cells present in the horizontal direction. The more cells, the better the grid, the higher the dimensionality of the resulting feature. It is mostly set to 8.
- 4) Grid Y: It is the number of cells present in the vertical direction. The more cells, the better the grid, the higher the dimensionality of the resulting feature. It is mostly set to 8.

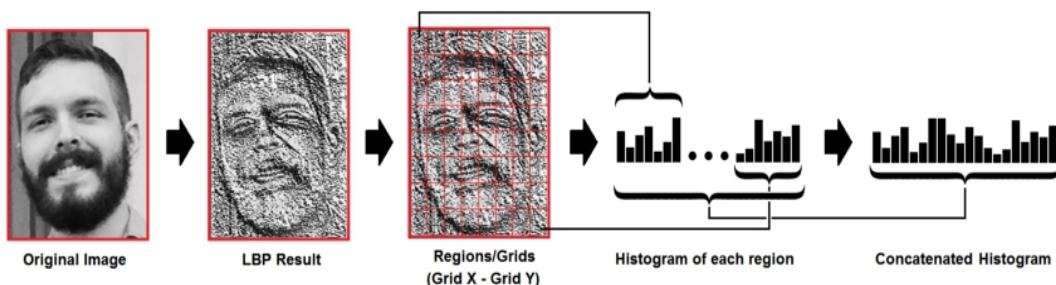


Fig.3 Based on the image above, we can extract the histogram of each region.

As we have an image in grayscale, each histogram (from each grid) will contain only 256 positions (0~255) representing the occurrences of each pixel intensity.

Then, we need to concatenate each histogram to create a new and bigger histogram. So to find the image which matches with the closest image by comparing their histograms. Output will be ID and Confidence.

4. Data updating:

After recognition of the face, the excel file is accessed using libraries like pandas and csv. The face IDs that were recognised are recorded and marked present in the csv file and then converted back to the excel file.

There are multiple ways of updating the excel file, one of them is by using openpyxl library to create a work station for excel updating.

	Term	Explanation
Table 1	Spreadsheet or Workbook	A Spreadsheet is the main file you are forming or working with.
Terms used in openpyxl	Worksheet or Sheet	A Sheet is used to split different kinds of content within the same spreadsheet. A Spreadsheet can have one or more Sheets .
	Column	A Column is a vertical line in the file.
	Row	A Row is a horizontal line in the File.
	Cell	A Cell is a combination of Column and Row together making a particular usnit of the file.

Pandas and can also be used to manipulate the data and save the updates in the file present in the storage. Pandas Data Frame will be formed by opening the data from existing storage can be SQL Database or Excel File.

Flow Chart

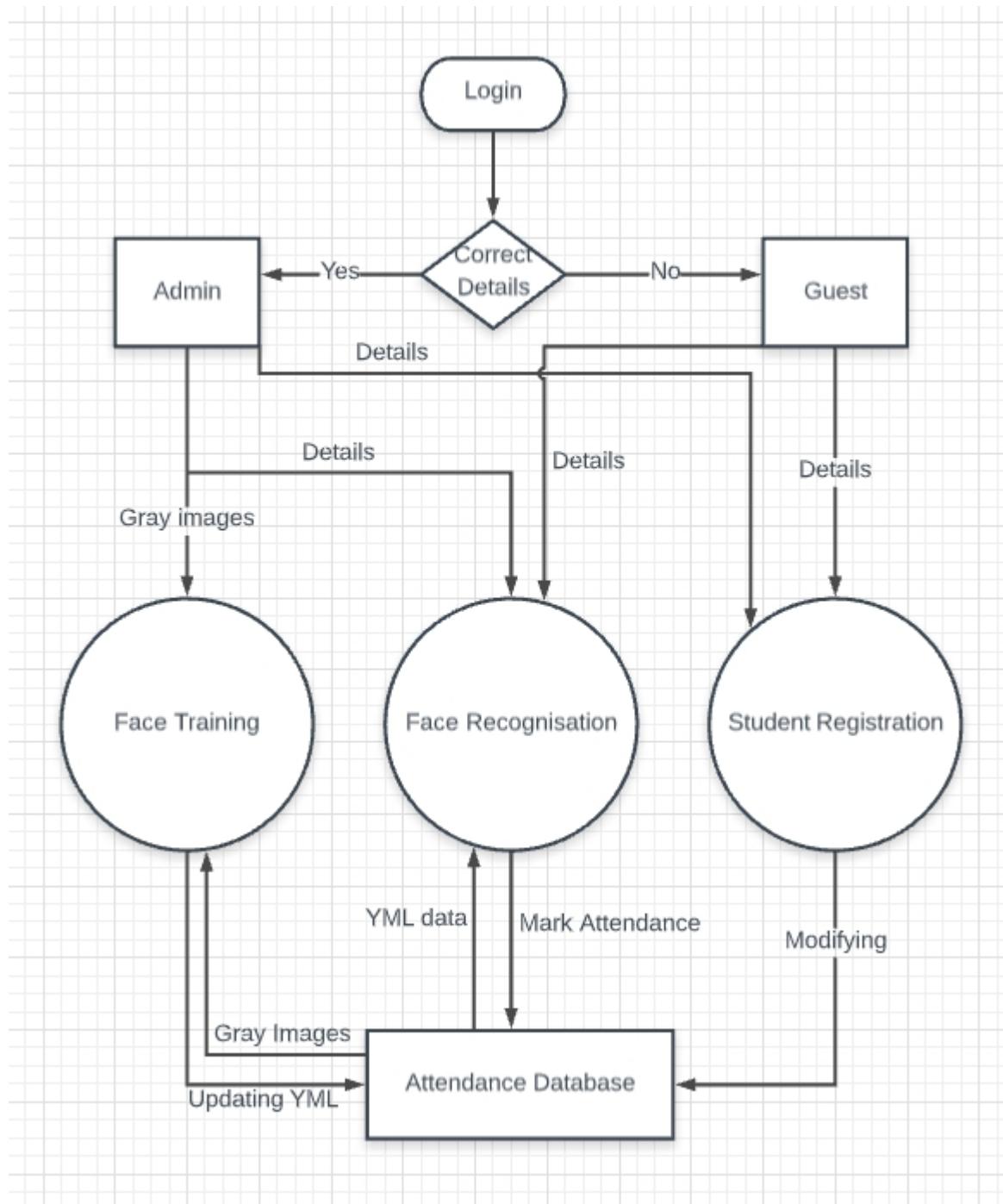


Fig.4 Data Flow Diagram of Auto Attendance System using Face recognition.

Use Case

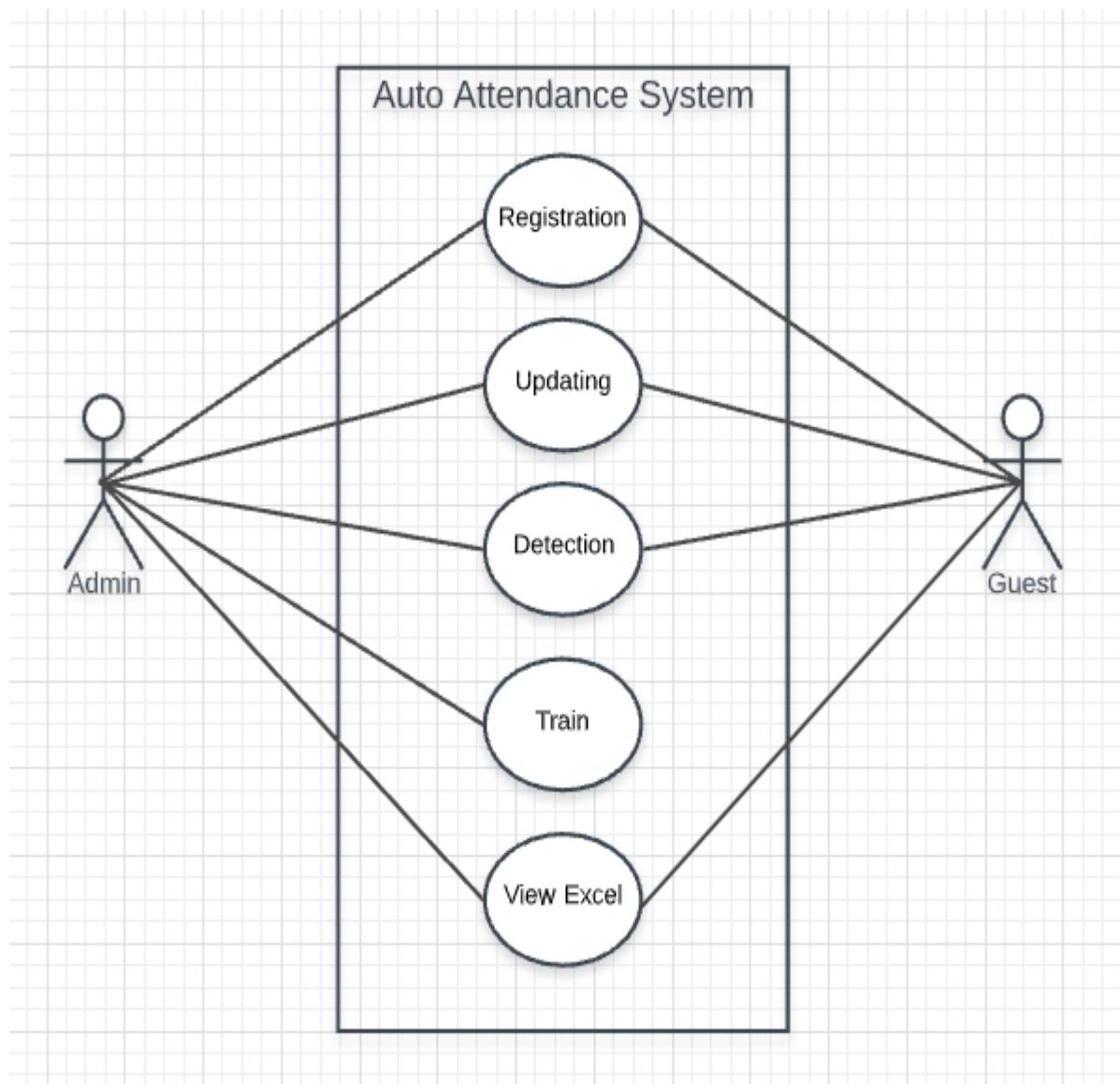


Fig.5 Use case of Auto Attendance System Using Face Recognition

Requirements

Hardware requirement

1. Processor: Intel Atom processor or above
2. Clock speed: 1.2 GHz or faster
3. Storage: 10 GB at least
4. RAM: More than 1GB

Software Requirements

1. Python36
2. Visual Source code
3. Microsoft Excel
4. Windows 7 and above

Libraries Used

1. Tkinter
2. Haar Cascade
3. Pillow
4. Panda
5. NumPy
6. OpenCV

Procedure, Codes and UI

The project is created in Visual Studio Code in Python language. The User Interface of the project is created using Tkinter library of Python and the Excel file is accessed and modified by CSV, Pandas, Numpy and Openpyxl libraries.

Haar-Cascade, OpenCV and LPBH recogniser is used for image and video processing to detect and recognise the faces in the given video sample.

The project is divided into following parts:

- Login screen
 - 1) Admin
 - 2) Guest user
- Registration of a student
 - 1) Register
 - 2) Update
 - 3) Delete
- Training the images
- Recognising the face
- Updating the excel
 - 1) Insert Date
 - 2) Mark Attendance
- Viewing the excel

Login Screen

The login screen accepts two entries, ID and a Password. There are two types of logging in is provided, one with guest user and the other with admin.

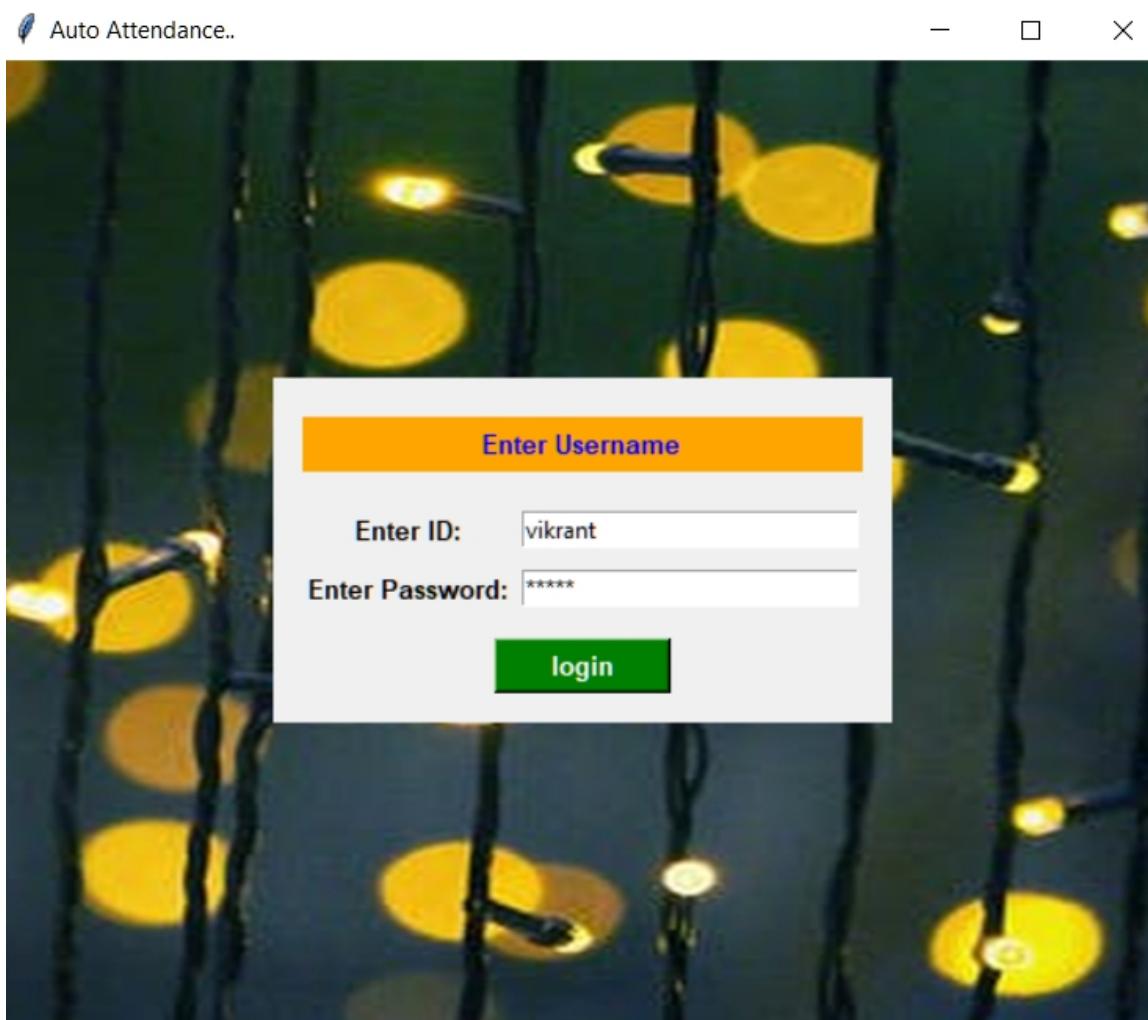
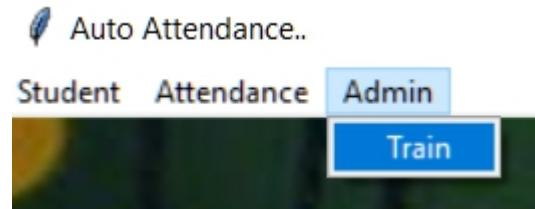


Fig.6 Login Screen

Admin

Admin gets few extra features like Training the images of the students making the program more efficient.



Guest

Guest user doesn't need a user ID or password, they have access to viewing the excel file and detecting the attendance from the video.



```

File Edit Selection View Go Run Terminal Help main_gui.py - Automated-Attendance-System-By-Real-Time-Face-Recognition-master - Visual Studio Code
main_gui.py > Login
184 **** Code for Login window starts *****
185 global login_by
186 Admin_id="vikrant"
187 Admin_pass="12345"
188
189 def login():
190     global login_by
191     lid=id.get()
192     lpass=pass.get()
193     print(lid.get(),lpass.get())
194
195     if(lid==Admin_id and lpass==Admin_pass):
196         login_by="Vikrant"
197         Proceed_menu()
198     else:
199         login_by="Guest"
200         Proceed_menu()
201
202
203 l_login=Label(f_login,image=photo)
204 f_login=Frame(l_login,pady="5",padx="15") #retaining a Frame which can expand according to the size of the window
205
206 lbo =Label(f_login,text="Enter Username",bg="orange",fg="blue",font="lucida 10 bold",width="35",pady="4").grid(columnspan=3,row=0,pady="15")
207 lbi =Label(f_login,text="Enter ID: ",font="lucida 10 bold").grid(column=0,row=2,pady="4")
208
209 lid =StringVar()
210 e1 =Entry(f_login, textvariable=lid,width="28").grid(column=1,row=2)
211 lb2 =Label(f_login, text="Enter Password: ",font="lucida 10 bold").grid(column=0,row=3,pady="4")
212
213 lpass=StringVar()
214 e2=Entry(f_login, show="*", textvariable=lpass,width="28").grid(column=1,row=3)
215 btn=Button(f_login, text="Login",bg="green",fg="white",width="10",font="lucida 10 bold",command=login)
216 btn.grid(columnspan=3, row=5,pady="10")
217
218 f_login.pack(pady="160")
219
220 l_login.pack(ipadx="100",fill=BOTH)

```

Fig.7 Login Screen code

```

File Edit Selection View Go Run Terminal Help
• main_gui.py - Automated-Attendance-System
main_gui.py  Dates.py  detect.py  train.py
main_gui.py > Proceed_menu
285
286     def Proceed_menu():
287         global login_by
288         l_login.pack_forget()
289
290         mainmenu = Menu(root)
291
292         m1 = Menu(mainmenu, tearoff=0)
293         m1.add_command(label="Register", command=lambda: student(1))
294         m1.add_command(label="View", command=lambda: student(2))
295         m1.add_separator()
296         m1.add_command(label="Update", command=lambda: student(3))
297         m1.add_command(label="Delete", command=lambda: student(4))
298         root.config(menu=mainmenu)
299         mainmenu.add_cascade(label="Student", menu=m1)
300
301         m2 = Menu(mainmenu, tearoff=0)
302         m2.add_command(label="Detect", command=lambda:attendance(1))
303         m2.add_separator()
304         m2.add_command(label="View Excel", command=lambda:attendance(2))
305         root.config(menu=mainmenu)
306         mainmenu.add_cascade(label="Attendance", menu=m2)
307
308         if(login_by=="Vikrant"):
309             m4 = Menu(mainmenu, tearoff=0)
310             m4.add_command(label="Train", command=lambda:admin(1))
311             root.config(menu=mainmenu)
312             mainmenu.add_cascade(label="Admin", menu=m4)
313
314

```

Fig. Proceed menu Code

```

def admin(x):
    if(x==1):
        print("i will train")
        tra.getImageswithId("../gray_images")

```

Fig.9 Train function called by admin

Registration

For registration of new students, their information is gathered and updated in the database file. The file's name is created on the basis of the information sent by the student. This is done to create an efficient database with the students of same class together.

Divided into Register Update and Delete.

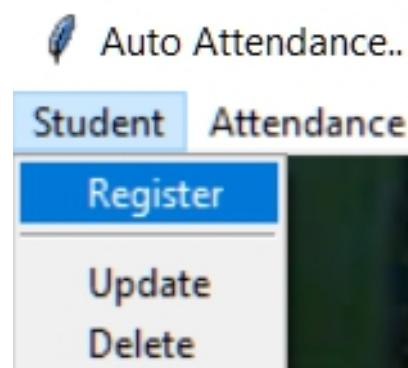


Fig.10 Menu of Register, Update and Delete

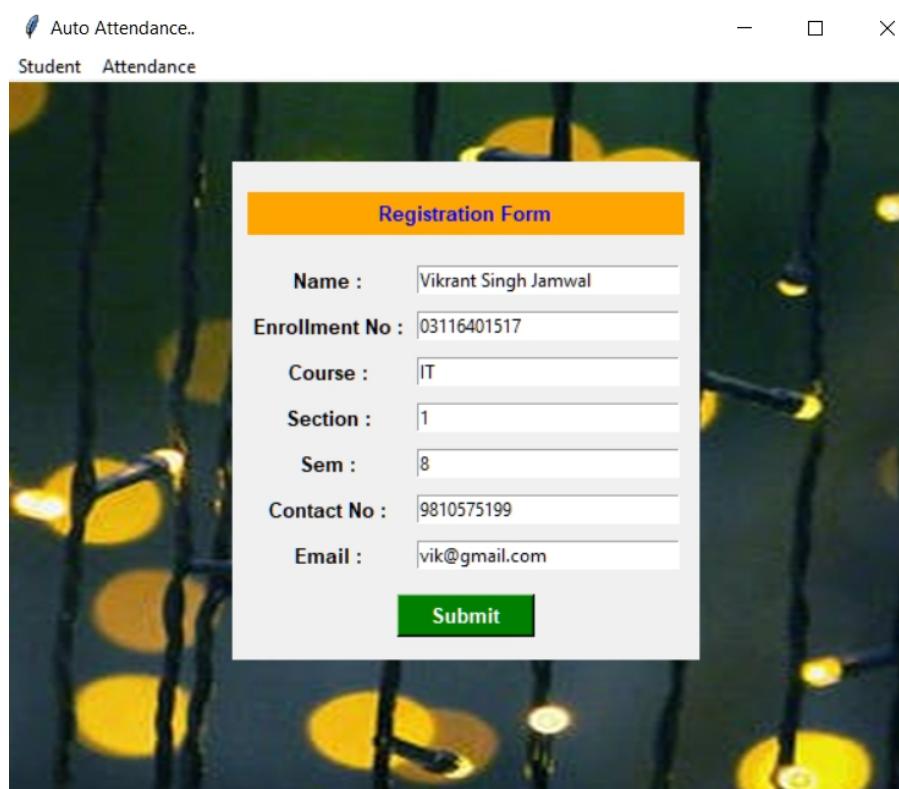


Fig.11 Registration window

```

File Edit Selection View Go Run Terminal Help • main_gui.py - Automated-Attendance-System-By-Real-Time-Face-Recognition-master - Visual Studio Code
main_gui.py • Dates.py detect.py train.py
main_gui.py > Proceed_menu
316     **** ***** Code for Registration starts **** ****
317
318
319     l = Label(image=photo)
320     f1=Frame(f1,pady="5",padx="10")
321     l0=Label(f1,text="Registration Form",bg="orange",fg="blue",font="lucida 10 bold",width="35",pady="4").grid(columnspan=3,row=0,pady="15")
322
323     l1=Label(f1,text="Name : ",font="lucida 10 bold").grid(column=0,row=1,pady="4")
324     name=Entry(f1,width="28")
325     name.grid(column=1,row=1)
326
327     l2=Label(f1,text="Enrollment No : ",font="lucida 10 bold").grid(column=0,row=2,pady="4")
328     enroll=Entry(f1,width="28")
329     enroll.grid(column=1,row=2)
330
331     l3=Label(f1,text="course : ",font="lucida 10 bold").grid(column=0,row=3,pady="4")
332     course=Entry(f1,width="28")
333     course.grid(column=1,row=3)
334
335     l32=Label(f1,text="Section : ",font="lucida 10 bold").grid(column=0,row=4,pady="4")
336     sectionn=Entry(f1,width="28")
337     sectionn.grid(column=1,row=4)
338
339
340     l33=Label(f1,text="Sem : ",font="lucida 10 bold").grid(column=0,row=5,pady="4")
341     semester=Entry(f1,width="28")
342     semester.grid(column=1,row=5)
343
344     l5=Label(f1,text="Contact No : ",font="lucida 10 bold").grid(column=0,row=6,pady="4")
345     contact=Entry(f1,width="28")
346     contact.grid(column=1,row=6)
347
348     l6=Label(f1,text="Email : ",font="lucida 10 bold").grid(column=0,row=7,pady="4")
349     email=Entry(f1,width="28")
350     email.grid(column=1,row=7)
351
352     btn=Button(f1,text="Submit",bg="green",fg="white",width="10",font="lucida 10 bold",command = Register_in_record)

```

Fig.12 Code for creating UI of the registration window

```

File Edit Selection View Go Run Terminal Help • main_gui.py - Automated-Attendance-System-By-Real-Time-Face-Recog
main_gui.py • Dates.py detect.py train.py
main_gui.py > Proceed_menu
70     def Register_in_record():
71
72         if semester.get() == '1' or semester.get() == '2':
73             year = 'first_year'
74         elif semester.get() == '3' or semester.get() == '4':
75             year = 'second_year'
76         elif semester.get() == '5' or semester.get() == '6':
77             year = 'third_year'
78         else:
79             year = 'fourth_year'
80
81         print(sectionn.get())
82
83         wb = load_workbook(f'data/excel/{year}/{year}_{semester.get()}sem_IT{sectionn.get()}.xlsx')
84
85         sheet = wb.active
86         sheet.column_dimensions['A'].width = 20
87         sheet.column_dimensions['B'].width = 30
88         sheet.column_dimensions['C'].width = 20
89         sheet.column_dimensions['D'].width = 20
90
91         sheet.cell(row=1, column=1).value = "Enrollment Number"
92         sheet.cell(row=1, column=2).value = "Name"
93         sheet.cell(row=1, column=3).value = "Email ID"
94         sheet.cell(row=1, column=4).value = "Mobile"
95
96         current_row = sheet.max_row
97         current_column = sheet.max_column
98
99         sheet.cell(row=current_row + 1, column=1).value = enroll.get()
100        sheet.cell(row=current_row + 1, column=2).value = name.get()
101        sheet.cell(row=current_row + 1, column=3).value = email.get()
102        sheet.cell(row=current_row + 1, column=4).value = contact.get()
103
104        wb.save(f'data/excel/{year}/{year}_{semester.get()}sem_IT{sectionn.get()}.xlsx')
105
106        print(name.get()," is Registered successfully")

```

Fig.13 Code for registering in the excel file

Update

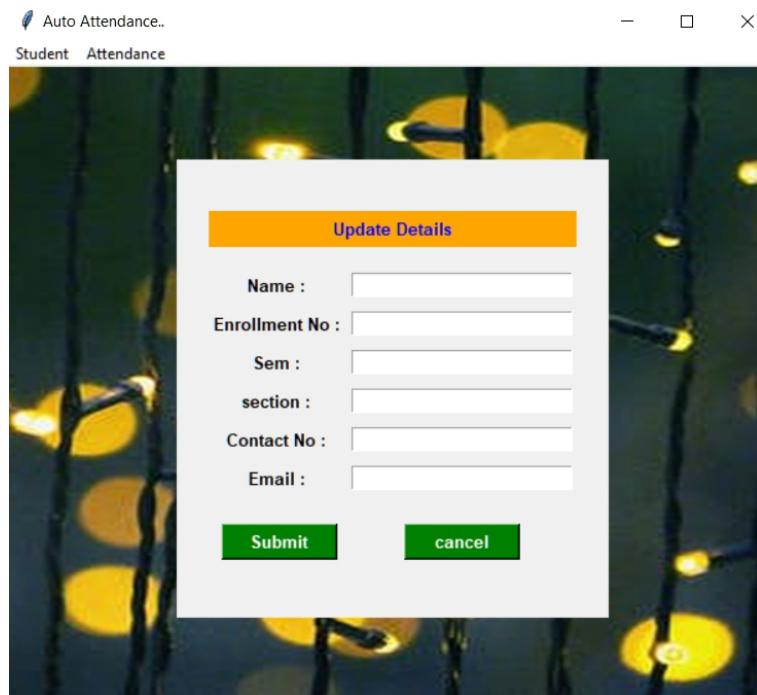


Fig.14 Update window

Delete

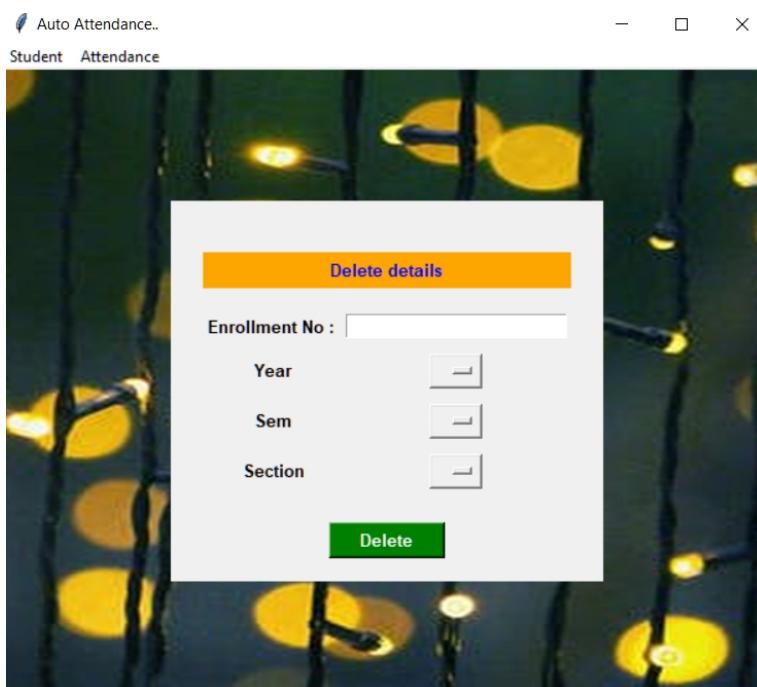


Fig.15 Delete Window

Detect

Detect opens up the given video and starts recognizing the faces seen in the video. As a face is recognised in 20 frames and has a confidence <80 , then it is saved in an array which later is updated as attendance in the database.

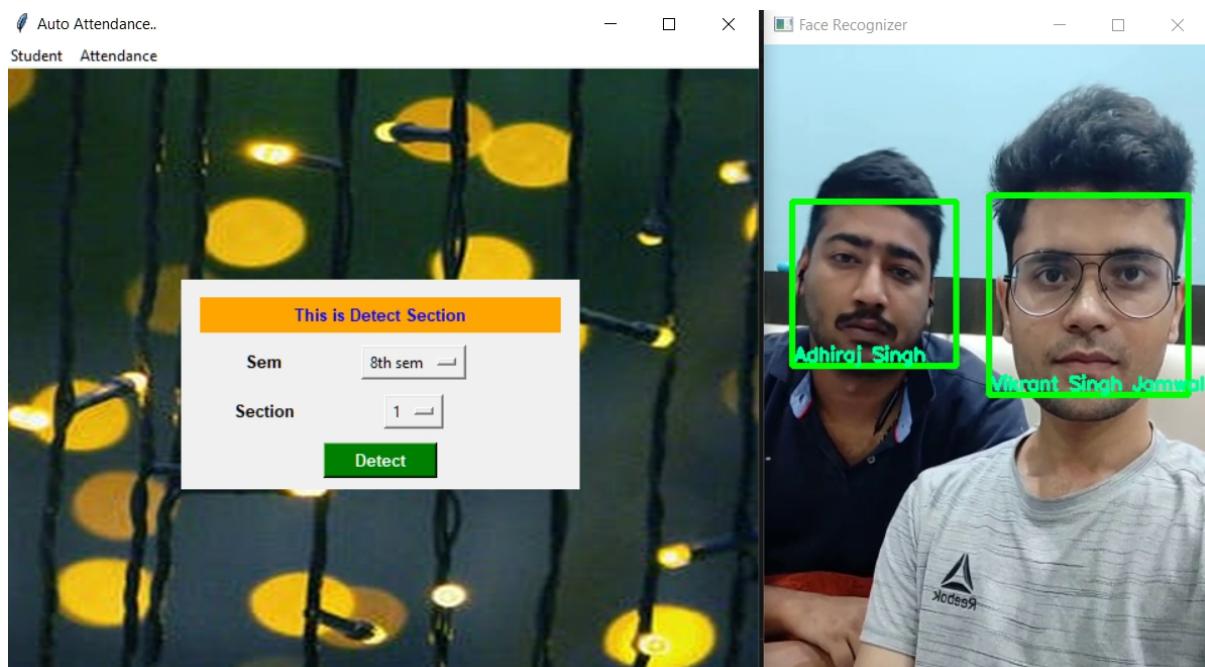
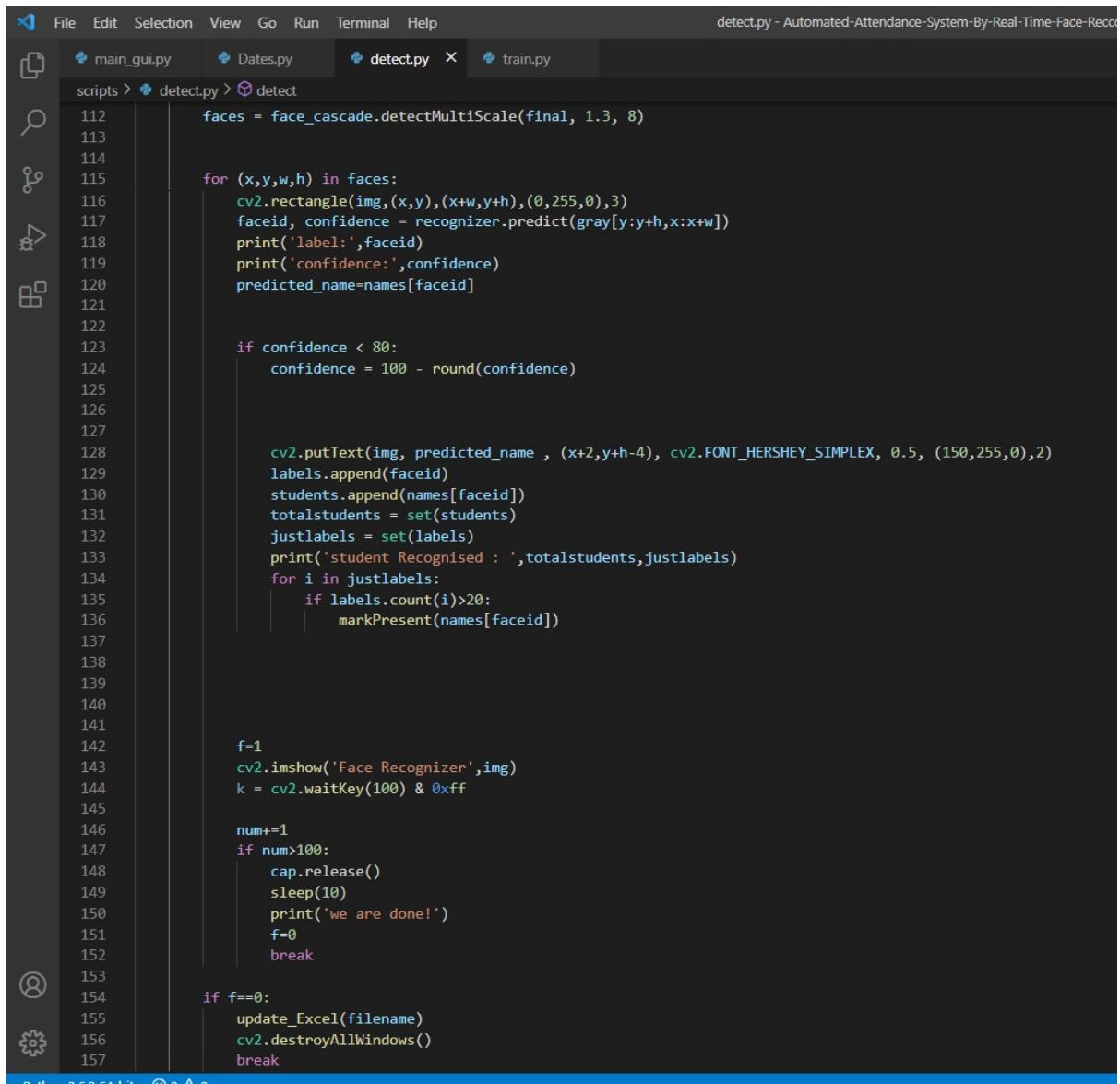


Fig.16 Detect and recogniser window

```
label: 0
confidence: 67.79053179760716
student Recognised : {'Vikrant Singh Jamwal', 'Adhiraj Singh'} {0, 5}
label: 5
confidence: 65.37350455916452
student Recognised : {'Vikrant Singh Jamwal', 'Adhiraj Singh'} {0, 5}
label: 0
confidence: 66.10602002899022
student Recognised : {'Vikrant Singh Jamwal', 'Adhiraj Singh'} {0, 5}
label: 5
confidence: 64.14227643762891
student Recognised : {'Vikrant Singh Jamwal', 'Adhiraj Singh'} {0, 5}
```

Fig.17 Terminal view of detection of faces



The screenshot shows a code editor window with the following details:

- Title Bar:** detect.py - Automated-Attendance-System-By-Real-Time-Face-Reco
- File Tabs:** main_gui.py, Dates.py, detect.py (active), train.py
- Code Area:**

```

112     faces = face_cascade.detectMultiScale(final, 1.3, 8)
113
114
115     for (x,y,w,h) in faces:
116         cv2.rectangle(img,(x,y),(x+w,y+h),(0,255,0),3)
117         faceid, confidence = recognizer.predict(gray[y:y+h,x:x+w])
118         print('label:',faceid)
119         print('confidence:',confidence)
120         predicted_name=names[faceid]
121
122
123         if confidence < 80:
124             confidence = 100 - round(confidence)
125
126
127
128         cv2.putText(img, predicted_name , (x+2,y+h-4), cv2.FONT_HERSHEY_SIMPLEX, 0.5, (150,255,0),2)
129         labels.append(faceid)
130         students.append(names[faceid])
131         totalstudents = set(students)
132         justlabels = set(labels)
133         print('student Recognised : ',totalstudents,justlabels)
134         for i in justlabels:
135             if labels.count(i)>20:
136                 markPresent(names[faceid])
137
138
139
140
141
142         f=1
143         cv2.imshow('Face Recognizer',img)
144         k = cv2.waitKey(100) & 0xff
145
146         num+=1
147         if num>100:
148             cap.release()
149             sleep(10)
150             print('we are done!')
151             f=0
152             break
153
154         if f==0:
155             update_Excel(filename)
156             cv2.destroyAllWindows()
157             break

```
- Status Bar:** Python 3.6.2 64-bit

Fig.18 LBPH code for recognising and marking the attendance

Detect function has 4 more functions present in it.

- `from_excel_to_csv`- Converts excel file to CSV file.
- `getData`- acquiring the data(names) from the file
- `markPresent`- to the last column in the database, P is written in front of the name of the student who is recognised by the LBPH recogniser.
- `update_Excel`- updating excel file from the csv file and saving it.

```

File Edit Selection View Go Run Terminal Help
detect.py - Automated-Attendance-S
scripts > detect.py > detect
53     def markPresent(name):
54         with open('../data.csv','r') as f:
55             data = csv.reader(f)
56             lines = list(data)
57
58             for line in lines:
59
60                 if line[3] == name:
61                     line[6] = 'P'
62
63
64
65                 with open('../data.csv','w') as g:
66                     writer = csv.writer(g,lineterminator='\n')
67                     writer.writerows(lines)
68             break
69
70
71
72     def update_Excel(filename):
73         with open('./data.csv') as f:
74
75             data = csv.reader(f)
76             lines = list(data)
77
78             with open('./data.csv','w') as g:
79                 writer = csv.writer(g,lineterminator='\n')
80
81                 writer.writerow(lines)
82
83             df = pd.read_csv(r'../data.csv')
84
85             filename = f'data/Attendance_xlsx/{year}_{sem}sem_IT{sec}.xlsx'
86             df.to_excel(filename,index = False)
87
88
89             print('Attendance is marked in excel')

```

Fig.19 markPresent and update_Excel

```

def from_excel_to_csv():
    filename = f'data/Attendance_xlsx/{year}_{sem}sem_IT{sec}.xlsx'
    df = pd.read_excel(filename)
    df.to_csv('../data.csv')

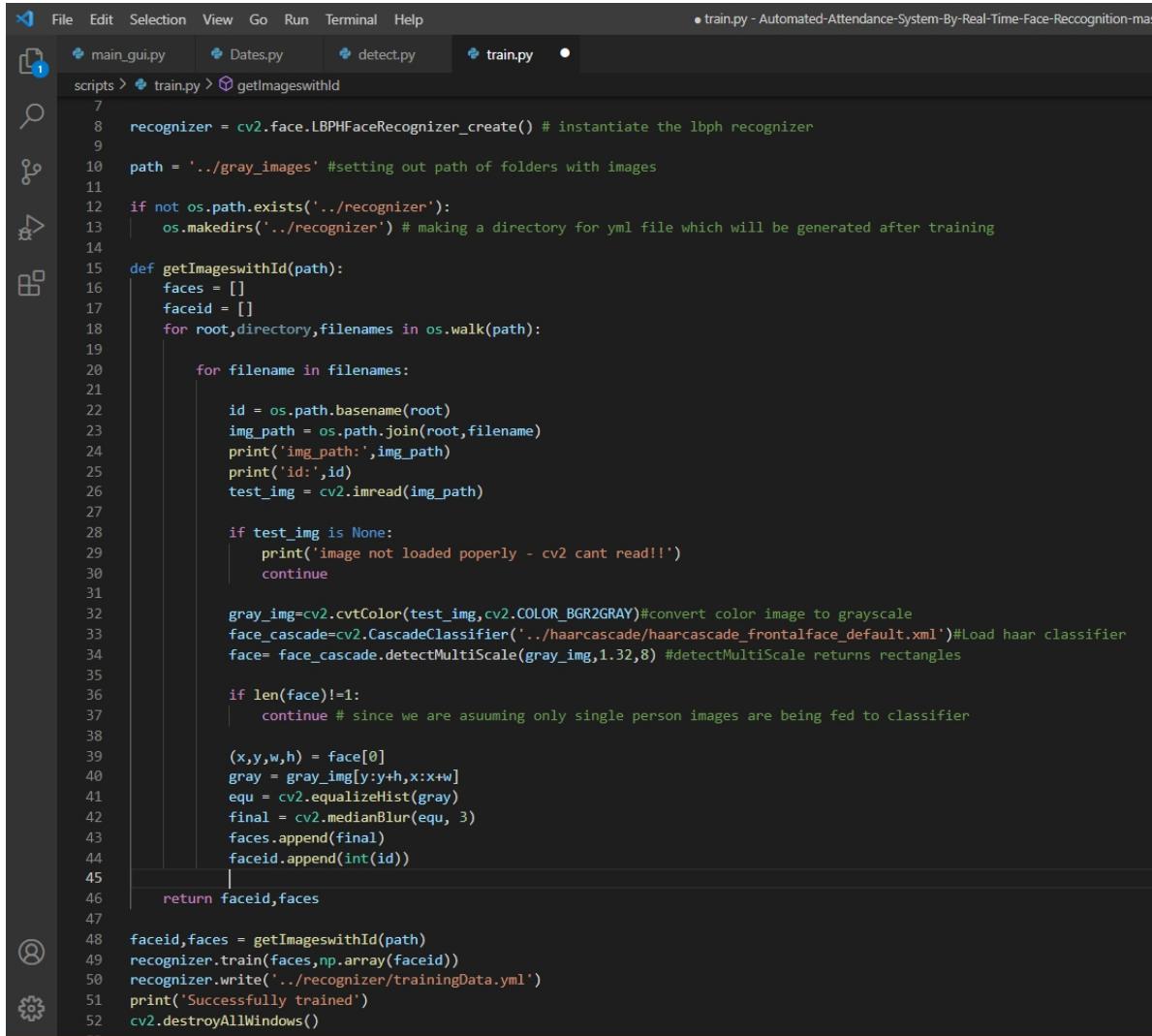

def getdata():
    with open('../data.csv','r') as f:
        data = csv.reader(f)
        next(data)
        lines = list(data)
        for line in lines:
            names[int(line[0])] = line[3]

```

Fig.20 from_excel_to_csv and getData

Train

Train program is used to create a YML document which saves the threshold of the 8 neighbouring pixels in the image. IDs are created and linked to the face detected using haar-cascade and are saved in the YML document.



```

File Edit Selection View Go Run Terminal Help
scripts > train.py > getImageswithId
7
8 recognizer = cv2.face.LBPHFaceRecognizer_create() # instantiate the lbph recognizer
9
10 path = '../gray_images' #setting out path of folders with images
11
12 if not os.path.exists('../recognizer'):
13     os.makedirs('../recognizer') # making a directory for yml file which will be generated after training
14
15 def getImageswithId(path):
16     faces = []
17     faceid = []
18     for root,directory,filenames in os.walk(path):
19
20         for filename in filenames:
21
22             id = os.path.basename(root)
23             img_path = os.path.join(root,filename)
24             print('img_path:',img_path)
25             print('id: ',id)
26             test_img = cv2.imread(img_path)
27
28             if test_img is None:
29                 print('image not loaded poperly - cv2 cant read!!')
30                 continue
31
32             gray_img=cv2.cvtColor(test_img,cv2.COLOR_BGR2GRAY)#convert color image to grayscale
33             face_cascade=cv2.CascadeClassifier('../haarcascade/haarcascade_frontalface_default.xml')#Load haar classifier
34             face= face_cascade.detectMultiScale(gray_img,1.32,8) #detectMultiScale returns rectangles
35
36             if len(face)!=1:
37                 continue # since we are asuuming only single person images are being fed to classifier
38
39             (x,y,w,h) = face[0]
40             gray = gray_img[y:y+h,x:x+w]
41             equ = cv2.equalizeHist(gray)
42             final = cv2.medianBlur(equ, 3)
43             faces.append(final)
44             faceid.append(int(id))
45
46     return faceid,faces
47
48 faceid,faces = getImageswithId(path)
49 recognizer.train(faces,np.array(faceid))
50 recognizer.write('../recognizer/trainingData.yml')
51 print('Successfully trained')
52 cv2.destroyAllWindows()

```

Fig.21 Train program code

```

img_path: ../gray_images\5\WhatsApp Image
id: 5
img_path: ../gray_images\5\WhatsApp Image
id: 5
img_path: ../gray_images\5\WhatsApp Image
id: 5
Successfully trained

```

Fig.22 Terminal view

YML document

Contains ID and linked face threshold pixel values.

Fig.23 YML document

Pictures of the students are kept in a folder, all the pictures are first converted to gray images and then get detected and saved in YML document.



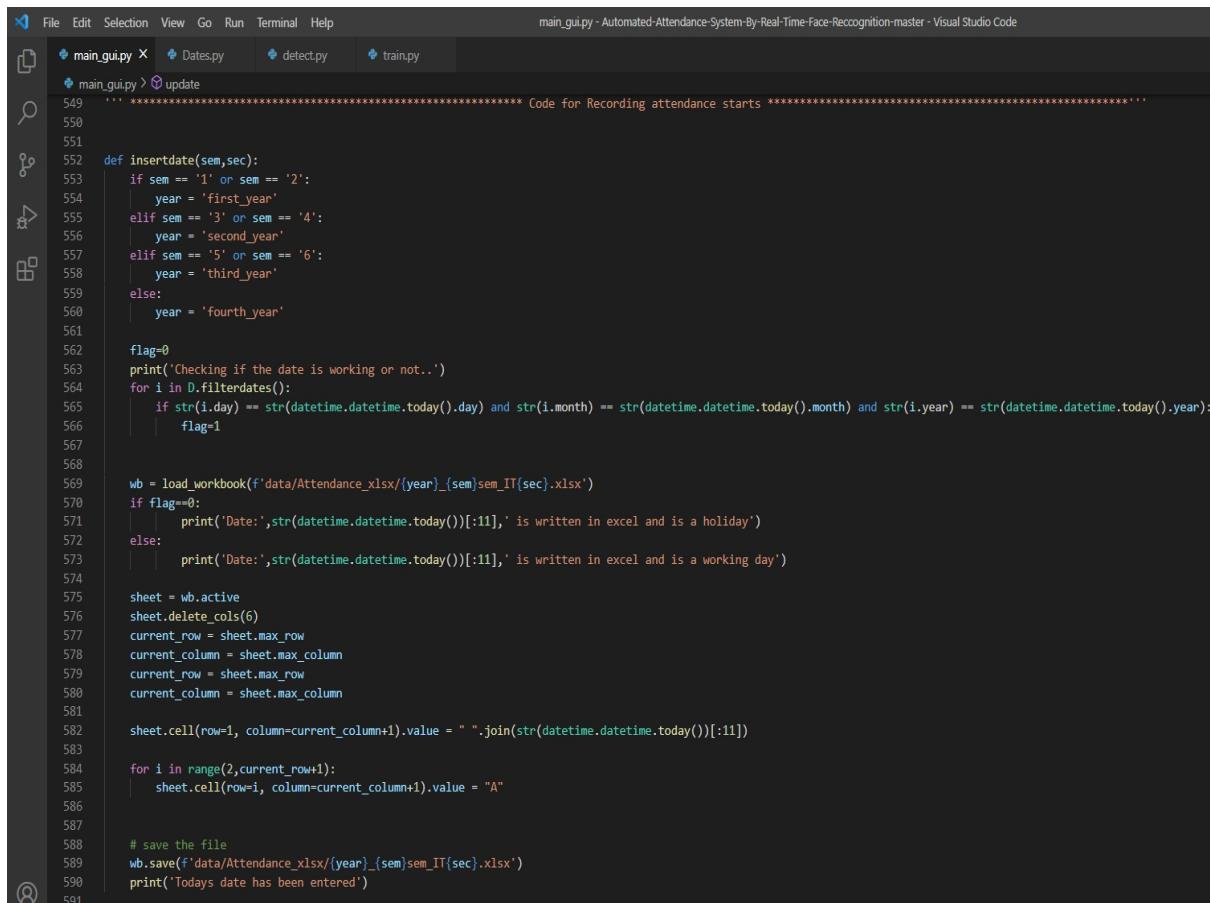
Fig.24 Pictures of the Students

Updating Excel file

After recognising the students, the excel file is updated. The heading attribute is the present Date and the default value under it is A(Absent). As the excel file is updated the P(Present) is written in front of the names of the students recognised in the given video.

Holidays are imported from the python packages to find out that any holiday lies on the present date or not.

Openpyxl is used to create a Workbook to manage and modify the excel file.



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

- File Explorer:** Shows files: main_gui.py (active), Dates.py, detect.py, and train.py.
- Code Editor:** Displays Python code for updating an Excel file. The code includes imports for `datetime` and `openpyxl`, defines a function `insertdate(sem,sec)` to handle semesters and years, and uses `openpyxl` to load and save an Excel workbook named after the current year and semester. It checks if the current date is a holiday and updates the Excel file by setting the first column of rows 2 and above to "A".
- Status Bar:** Shows the file path: main_gui.py - Automated-Attendance-System-By-Real-Time-Face-Recognition-master - Visual Studio Code.

```

File Edit Selection View Go Run Terminal Help
main_gui.py - Automated-Attendance-System-By-Real-Time-Face-Recognition-master - Visual Studio Code
main_gui.py X Dates.py detect.py train.py
main_gui.py > update
549     **** ***** Code for Recording attendance starts **** ****
550
551
552 def insertdate(sem,sec):
553     if sem == '1' or sem == '2':
554         year = 'first_year'
555     elif sem == '3' or sem == '4':
556         year = 'second_year'
557     elif sem == '5' or sem == '6':
558         year = 'third_year'
559     else:
560         year = 'fourth_year'
561
562 flag=0
563 print('Checking if the date is working or not..')
564 for i in D.filterdates():
565     if str(i.day) == str(datetime.datetime.today().day) and str(i.month) == str(datetime.datetime.today().month) and str(i.year) == str(datetime.datetime.today().year):
566         flag=1
567
568
569 wb = load_workbook(f'data/Attendance_xlsx/{year}_{sem}_sem_IT{sec}.xlsx')
570 if flag==0:
571     print('Date:',str(datetime.datetime.today())[:11],' is written in excel and is a holiday')
572 else:
573     print('Date:',str(datetime.datetime.today())[:11],' is written in excel and is a working day')
574
575 sheet = wb.active
576 sheet.delete_cols(6)
577 current_row = sheet.max_row
578 current_column = sheet.max_column
579 current_row = sheet.max_row
580 current_column = sheet.max_column
581
582 sheet.cell(row=1, column=current_column+1).value = " ".join(str(datetime.datetime.today())[:11])
583
584 for i in range(2,current_row+1):
585     sheet.cell(row=i, column=current_column+1).value = "A"
586
587
588 # save the file
589 wb.save(f'data/Attendance_xlsx/{year}_{sem}_sem_IT{sec}.xlsx')
590 print('Todays date has been entered')
591

```

Fig.25 Code for inserting the date in the Excel file.

```

53     def markPresent(name):
54         with open('../data.csv','r') as f:
55             data = csv.reader(f)
56             lines = list(data)
57
58             for line in lines:
59
60                 if line[3] == name:
61                     line[6] = 'P'
62
63
64
65                 with open('../data.csv','w') as g:
66                     writer = csv.writer(g,lineterminator='\n')
67                     writer.writerows(lines)
68             break
69

```

Fig.26 Code for marking present in the CSV file.

```

def update_Excel(filename):
    with open('./data.csv') as f:

        data = csv.reader(f)
        lines = list(data)

        with open('./data.csv','w') as g:
            writer = csv.writer(g,lineterminator='\n')

            writer.writerow(lines)

    df = pd.read_csv(r'../data.csv')

    filename = f'data/Attendance_xlsx/{year}_{sem}sem_IT{sec}.xlsx'
    df.to_excel(filename,index = False)

    print('Attendance is marked in excel')

```

Fig.27 Transferring data from CSV to Excel File.

Viewing The Excel File

View Excel option in the menu bar is used to open the excel file from the software itself.

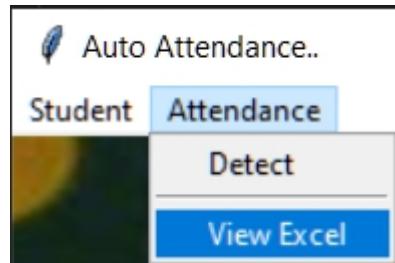


Fig.28 View Excel Option

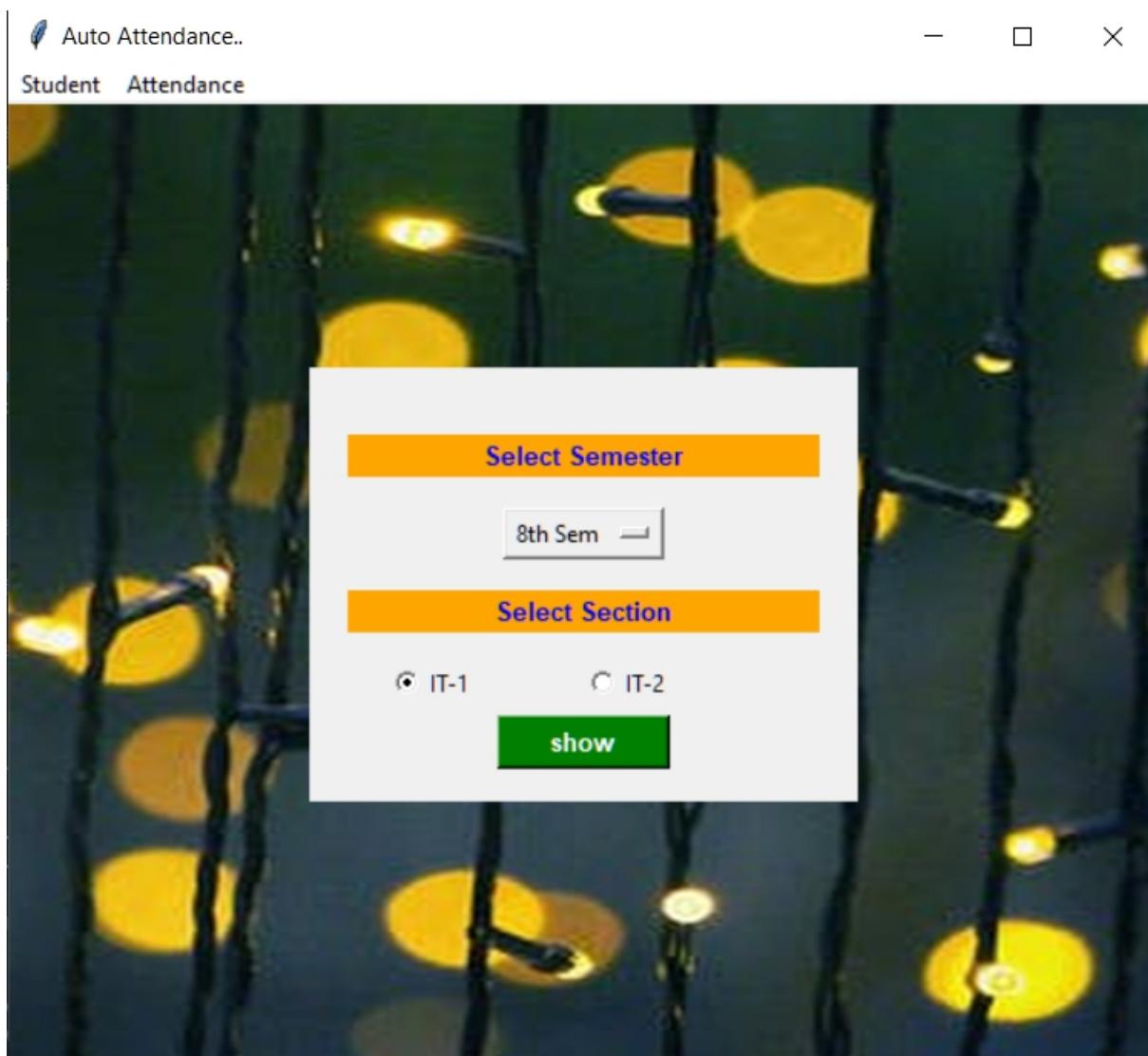


Fig.29 UI for opening the Excel file

	A	B	C	D	E	F
1	ID	Enrollment Number	Name	Email ID	Mobile	2021-07-01
2	1	345615890	Adhiraj Singh	adi@gmail.com	9810675489	P
3	2	42116401517	Devesh Gupta	dev@gmail.com	9810010101	A
4	3	4116401517	Lakshya	lakshu@gmail.com	9810075610	A
5	4	1516401517	Navdha Singh	nav@gmail.com	9810575637	A
6	5	271641517	Tanishq Khanna	tan@gmail.com	9817890647	A
7	6	3116401517	Vikrant Singh Jamwal	vik@gmail.com	8936765898	P
8						
9						

Fig.30 Preview of the Updated Excel file.

```

File Edit Selection View Go Run Terminal Help
main_gui.py - Automated-Attendance-System-By-Real-Time-Face-Recognition-master - Visual Studio Code

654  **** Code for viewing Excel starts *****
655
656 def open_excel():
657     sem = seme.get()[0]
658     sec= sect.get()
659
660     if sem == '1' or sem == '2':
661         year = 'first_year'
662     elif sem == '3' or sem == '4':
663         year = 'second_year'
664     elif sem == '5' or sem == '6':
665         year = 'third_year'
666     else:
667         year = 'fourth_year'
668
669
670     file='./Automated-Attendance-System-By-Real-Time-Face-Recognition-master/data/Attendance_xlsx/{year}_{sem}_IT{sec}.xlsx'
671     os.startfile(os.path.normpath(file))
672
673 fa=Frame(11,pady="25",padx="20",height=300)
674
675 Label(fa, text = "Select Semester",bg="orange",fg="blue",font="lucida 10 bold",width="30",height=1).grid(columnspan=3,row=2,pady="10")
676 seme=StringVar()
677 seme.set("8th Sem") # default value
678 w1 = OptionMenu(fa,seme,"1st Sem","2nd Sem","3rd Sem","4th Sem","5th Sem","6th Sem","7th Sem","8th Sem").grid(columnspan=3,row=3,pady="4")
679
680
681 if sem == '1' or sem == '2':
682     year = 'first_year'
683 elif sem == '3' or sem == '4':
684     year = 'second_year'
685 elif sem == '5' or sem == '6':
686     year = 'third_year'
687 else:
688     year = 'fourth_year'
689
690 sect = StringVar()
691 sect.set("1")
692 Label(fa, text = "Select Section ",bg="orange",fg="blue",font="lucida 10 bold",width="30",height=1).grid(columnspan=3,row=5,pady="10")
693 radio = Radiobutton(fa, text="IT-1",variable=sect, value="1").grid(column=0,row=6,pady="4")
694 radio = Radiobutton(fa, text="IT-2", padx=14, variable=sect, value="2").grid(column=1,row=6,pady="4")
695 btn=Button(fa,text="show",bg="green",fg="white",width="10",font="lucida 10 bold",command=open_excel)
696 btn.grid(columnspan=3,row=7,pady="0")
697 fa.pack(pady="30")
698 ll.pack(ipadx="100",fill=BOTH)

```

Fig.31 Code for viewing Excel file

References

1. <https://docs.python.org/3/library/tkinter.html>
2. www.intel.in
3. <https://code.visualstudio.com>
4. <https://docs.opencv.org/>
5. <https://www.youtube.com/>
6. <https://www.w3schools.com/>
7. <https://stackoverflow.com/>

Books

1. 'Principles of digital image processing' by Burge and Burger
2. 'Head-First Python' by Paul Barry