



**Chandigarh Engineering College Jhanjeri  
Mohali - 140307**  
**Department of Computer Science & Engineering**

**Project Report  
On  
Live Logging Facial Recognition Attendance System**

Project – II



**Department of Computer Science & Engineering  
CHANDIGARH ENGINEERING COLLEGE JHANJERI, MOHALI**

**In partial fulfilment of the requirements for the award of the Degree of  
Bachelor of Technology in Computer Science & Engineering**

**SUBMITTED BY:**

Prince Kumar	1816590
Abhinandan Kumar	1816823
Prerna Kumar	1816589

**Under the Guidance of**

Mrs. Sakshi Sharma  
Asst. Professor

DEC 2021



**Affiliated to I. K. Gujral Punjab Technical University, Jalandhar  
(Batch: 2018 – 2022)**



**Chandigarh Engineering College Jhanjeri  
Mohali - 140307**  
**Department of Computer Science & Engineering**

**DECLARATION**

I, Prince Kumar and my team hereby declare that the report of the project entitled “Live Logging Facial Recognition Attendance System” has not presented as a part of any other academic work to get my degree or certificate except Chandigarh Engineering College Jhanjeri, Mohali, affiliated to I.K. Gujral Punjab Technical University, Jalandhar, for the fulfilment of the requirements for the degree of B Tech in Computer Science & Engineering.

**Prince Kumar**

1816590

Semester: 7<sup>th</sup>

**Mrs. Sakshi Sharma**

Asst Professor, CSE

**Dr. Anil Kumar Lamba**

**HOD CSE**



# Chandigarh Engineering College Jhanjeri

## Mohali - 140307

### Department of Computer Science & Engineering

#### ACKNOWLEDGEMENT

It is great pleasure to present this report on the project title "**Live Logging Facial Recognition Attendance System**" undertaken by me as part of my B. Tech degree in Computer Science & Engineering final year. I am thankful to my university for offering me such a wonderful challenging opportunity and I express my deepest thanks to all coordinators, for providing all the possible help and assistance and their constant encouragements.

I am extremely thankful to **Dr. Anil Kumar Lamba**, HoD, Computer Science & Engineering, Chandigarh Engineering College Jhanjeri, Mohali (Punjab) for valuable suggestions and motivation.

I am also thankful to all my teachers who have taught me till date in my degree, management of institute and **Dr. Rajneesh Talwar**, Director Engineering for providing me the opportunity to get the knowledge.

**Prince Kumar**



**Chandigarh Engineering College Jhanjeri**  
**Mohali - 140307**  
**Department of Computer Science & Engineering**

**TABLE OF CONTENTS**

<b>PARTICULARS</b>	<b>PAGE NO</b>
Declaration by the Candidate	I
Acknowledgement	II
Table of Contents	III– IV
<b>CHAPTER 1: INTRODUCTION</b>	<b>1-3</b>
1.1 Introduction	1
1.2 What is a Computer Vision?	1
1.3 Objectives of Face Recognition Attendance System	2 – 3
<b>CHAPTER 2: REVIEW OF LITERATURE</b>	<b>4 – 7</b>
2.1 Anaconda 3	4
2.2 Spyder	4 – 7
2.2.1 Advantages of Spyder	4
2.2.2 Steps to installing libraries	5 – 7
<b>CHAPTER 3: PROBLEM DEFINATION AND OBJECTIVES</b>	<b>8 – 12</b>
3.1 Facial Landmarks	8 – 9
3.2 Camera Analysis	9 – 10
3.2.1 Video Stream	9
3.2.2 Illumination and image quality	9
3.2.3 Scene and camera position	10
3.3 Skills	10
3.4 Libraries	10 – 12



**Chandigarh Engineering College Jhanjeri  
Mohali - 140307**  
**Department of Computer Science & Engineering**

<b>CHAPTER 4: DESIGN AND IMPLEMENTATION</b>	<b>13 - 22</b>
4.1 Database	13 – 14
4.1.1 Database Categories	13
4.1.2 Data Model	13 – 14
4.2 Flow Chart	14
4.3 ER Diagram	15
4.4 Methods to Build Live Logging Facial Recognition Attendance Systems	15 – 16
4.4.1 Loading image data	15
4.4.2 Encoding of load image	16
4.4.3 Features Detection	16
4.4.4 Video Streaming & Real Times	16
4.4.5 Create and Update CSV sheet	16
4.5 Coding of Live Logging Facial Recognition Attendance Systems	16 – 22
<b>CHAPTER 5: RESULTS AND DISCUSSION</b>	<b>23 – 38</b>
<b>CHAPTER 6: CONCLUSION AND FUTURE SCOPE</b>	<b>39 – 40</b>
6.1 Conclusion	39
6.2 Future Scope	40
<b>REFERENCES</b>	<b>41 – 42</b>



# Chandigarh Engineering College Jhanjeri

## Mohali - 140307

### Department of Computer Science & Engineering

## Chapter- 1

### Introduction

#### 1.1 Introduction

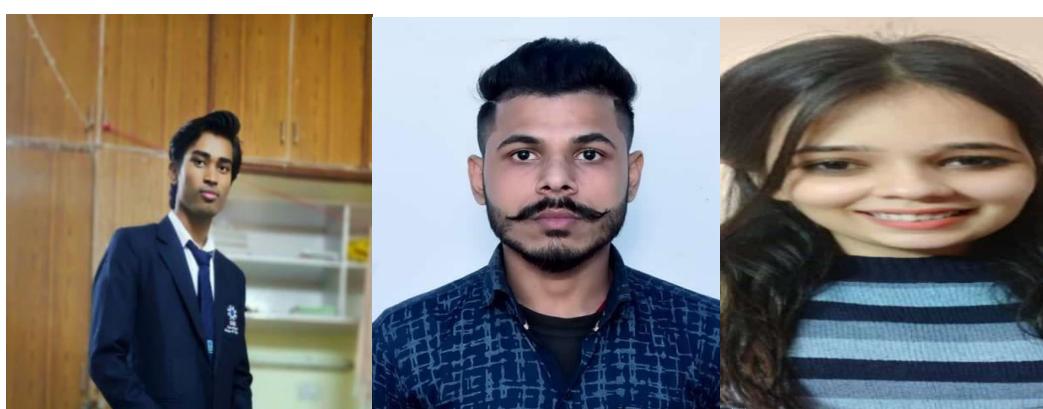
This project is based on the technology of Computer Vision in AI, it marking attendance from detecting faces of person. It detects the face of person in real time using webcam. The Computer Vision techniques we use in this project is image analysis, face detection, face-recognition.

This Model is checks that captured image attendance is marked or not and take the decision based on the data which it will be get, processed it and give the output. If the attendance of that capture image is not marked then model mark it. If the attendance of that captured image is already marked then the model shows attendance already marked of that captured person image. So, we can proceed it to the next one and same process is repeated for the every person.

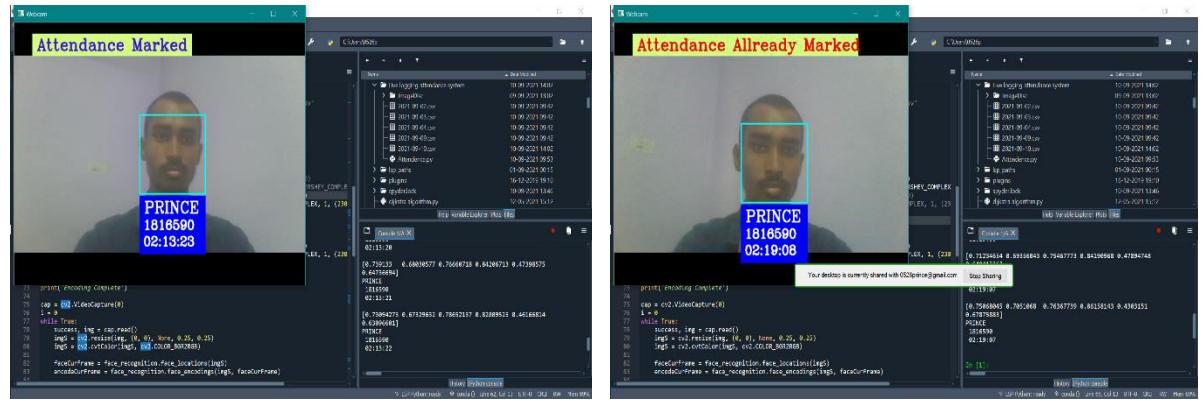
#### 1.2 What is a Computer Vision?

Computer Vision is a field of Artificial Intelligence that enables computers and systems to grab information from digital images, videos and in real time videos and takes actions or make decision and recommendations based on that information. Computer Vision is simply means we make a model which enables to see, observe, understand and make decisions.

Computer vision works as like human visions. Computer visions trains machines to perform these functions but it has to do in very less times with cameras, data and algorithms. Because a model trained to observe products or process a minute, detect defects or issues, it can quickly surpass human capabilities.



Person, they want to mark their attendance



### Marking Attendance

#### 1.3 Objectives of Face Recognition Attendance System

From a long term we use the old method for taking attendance using paper or pen. One by one we call the roll no or name of student and then we mark it present or absent. We also uses many papers or pages for store the data of attendance. And also teachers are facing many problems while taking attendance. Sometimes due to miscommunication teacher mark the wrong attendance.

For store the data of attendance we use lot of papers. So, we have to cut a large amount of tree in whole world for accruing the demand of papers. According to the survey a single tree produces 10,000 to 20,000 papers.

Papermaking has an impact on environment because it destroys trees continuously in every single day. According to the data from the Global Forest Resource Assessment roughly 80,000 to 160,000 trees are cut down each day around the world for papermaking by the paper industry. So we are made a digital method to mark attendance of peoples without using pen and paper. Therefore, we are work productively and comfortably.



The goal of this project is to detect and locate human faces in webcam and encode it compare the images which are present in the student or specific directory. We need to make a directory of images where all the images of those student or persons are present which one I want to mark attendance. A set of images were provided for this purpose. The objective was to design and implement a face detector that will detect human faces in webcam similar to the images present in the directory. Where the model compares both images and mark attendance for more accurate input images. If the attendance is not marked, then attendance of particular person is marked. On the other side if the attendance is already marked then it do not overwrite it. All the data of attendance system is mark automatically and maintain the CSV sheet automatically by the model. We do not need to care about that what will be going on the backend. Person only work to show his/her faces in the webcam.



# Chandigarh Engineering College Jhanjeri

## Mohali - 140307

# Department of Computer Science & Engineering

## Review of Literature

**2.1 Anaconda 3:** You must have anaconda 3 in your system. You can install anaconda 3 using any browser. After successful installation you have a lot of choice for your programming interface such as Jupyter, Pycharm, Visual Studio, Spyder etc. For professional programming and for convenient Spyder is better than anyone and also for huge programming Spyder is more comfortable for any of those. It has more helpful tools which helps you while coding.

**2.2 Spyder:** As we know there are many open-source software are available for making a perfect model some of the open-source software is Pycharm, Atom, Spyder, Sublime Text, Jupyter. In this project we were using Spyder from Anaconda for building this model. You can also use Jupyter or any other software as according to your comfort. But we personally recommended uses Spyder for writing code and implementation. There have lots of advantages why you choose this software. Few of them are given below.

### 2.2.1 Advantages of Spyder:

- It is a cross – platforms and open – source IDE for python. It is available for Windows, Linux, and MAC OS
- It analyse code to provide automatic code completion, horizontal/vertical splitting, and a go-to-definition.
- It is specifically designed for data scientists; hence, it integrates well with data science libraries like NumPy, Pandas, Tensorflow, OpenCV, Scipy, and so on.
- It allows you to run and create the multiple IPython console you have no restriction to work on the particular or only on the current console. You are free to work and run code or check debugging in any console that's the more interesting features why we choose Spyder.
- It includes a powerful integrated debugger which is very beneficial for analysing code in real time what is the error in particular section of program.
- It contains an integrated documentation browser which helps you study or read any time anywhere like online help, are great for beginners.

### 2.2.2 Steps for installing libraries:



# Chandigarh Engineering College Jhanjeri

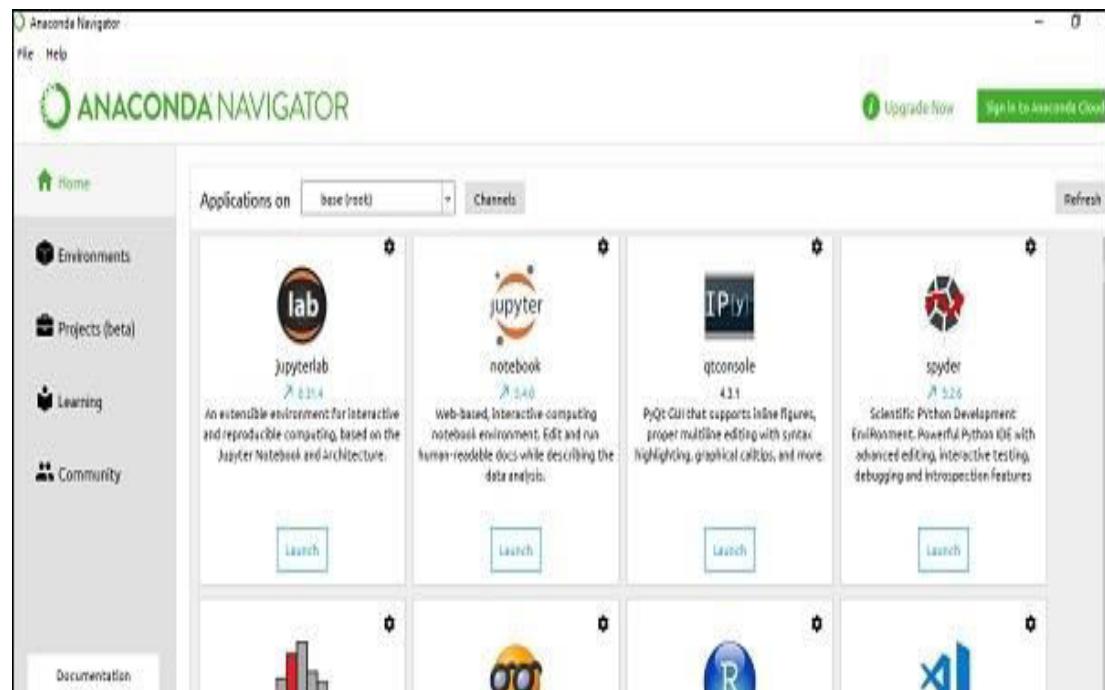
## Mohali - 140307

### Department of Computer Science & Engineering

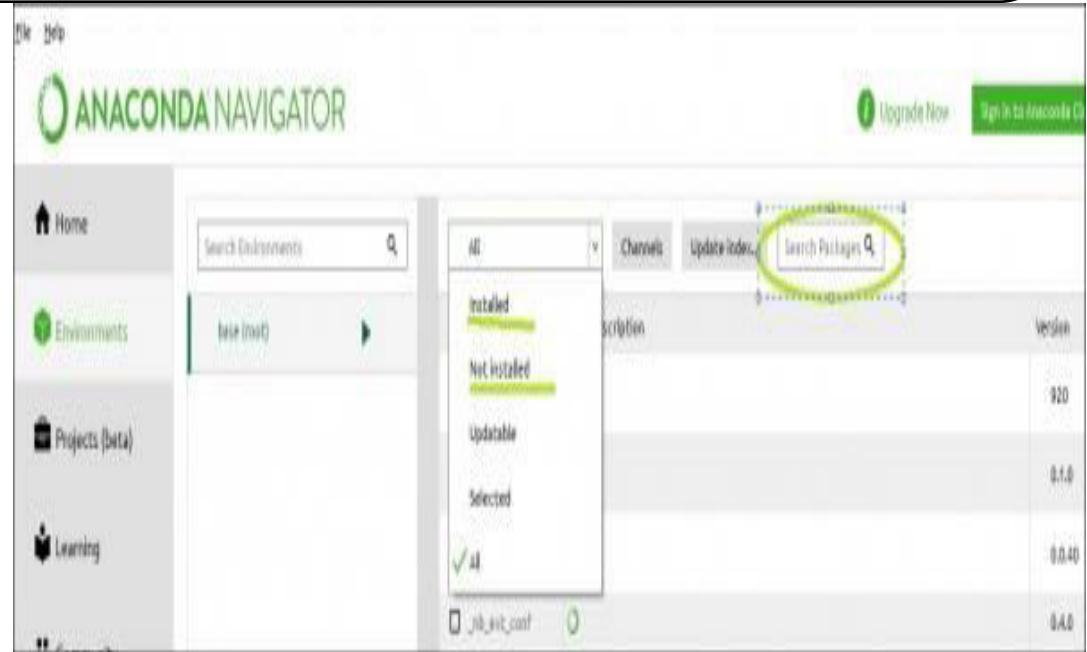
In Spyder most of the libraries are preinstalled but some of the libraries according to our requirement need to be installed. We can install all those libraries in two ways either Anaconda Prompt/terminal or directly from Anaconda environment. We mention both of the methods you can go through any of them:

#### Method 1:

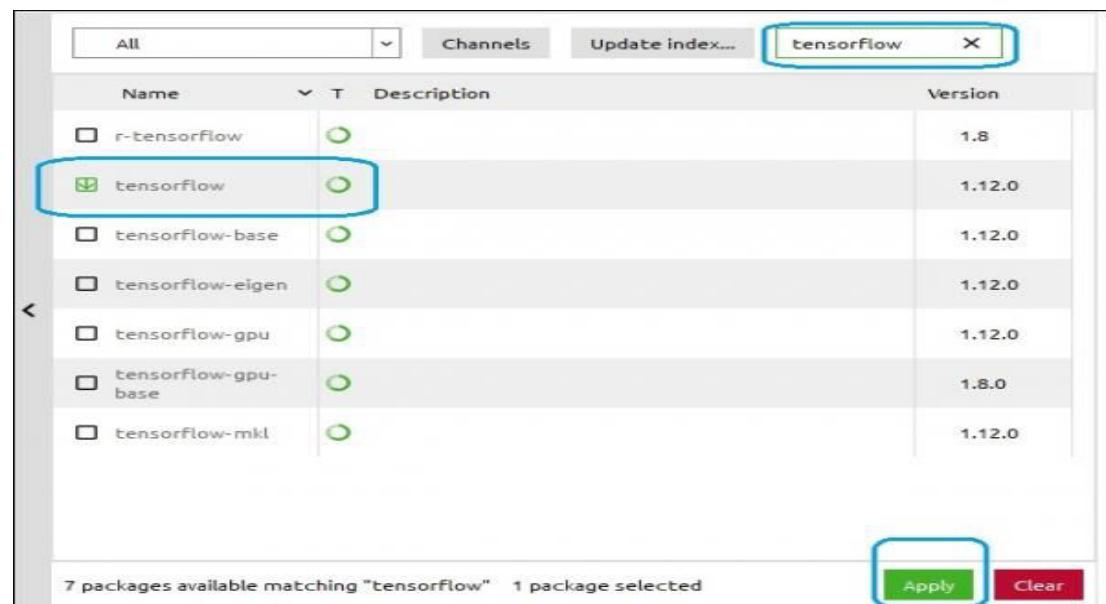
- Go to the anaconda navigator



- Click on the “Environment” tab and from there we can check which libraries are installed or not. After that click on the search bar then type libraries name which one you want to install.



- It shows if the library is installed or not. If not then select the library option you want to install and click the apply button your library is installed in your Anaconda environment.



#### Method 2:

- Open Anaconda prompt or terminal (it also supports pip command)



# Chandigarh Engineering College Jhanjeri

## Mohali - 140307

### Department of Computer Science & Engineering

conda install numpy

```
(base) C:\Users\0526p> conda install numpy
Collecting package metadata (current_repodata.json): done
Solving environment: done

## Package Plan ##

environment location: C:\Users\0526p\Anaconda3

added / updated specs:
- numpy

The following packages will be downloaded:

  package          |      build
ca-certificates-2021.4.13 | ha95532_1    115 KB
Total:                115 KB

The following packages will be UPDATED:

  ca-certificates          2021.1.19-haa95532_1 --> 2021.4.13-haa95532_1

Proceed ([y]/n)? y

Downloading and Extracting Packages
ca-certificates-2021 | 115 KB  | #####| 100%
Preparing transaction: done
Verifying transaction: done
Executing transaction: done

(base) C:\Users\0526p>
```

pip install tensorflow (it also works)

```
(base) C:\Users\0526p> pip tensorflow
ERROR: unknown command "tensorflow"
(base) C:\Users\0526p> pip install tensorflow
Requirement already satisfied: tensorflow in c:\users\0526p\anaconda3\lib\site-packages (2.4.1)
Requirement already satisfied: tensorboard<=2.4 in c:\users\0526p\anaconda3\lib\site-packages (from tensorflow) (2.4.1)
Requirement already satisfied: tensorflow==1.1.0 in c:\users\0526p\anaconda3\lib\site-packages (from tensorflow) (1.1.0)
Requirement already satisfied: opt-einsum<=3.3.0 in c:\users\0526p\anaconda3\lib\site-packages (from tensorflow) (3.3.0)
Requirement already satisfied: flatbuffers<=1.12.0 in c:\users\0526p\anaconda3\lib\site-packages (from tensorflow) (1.12)
Requirement already satisfied: wrapt<=1.12.1 in c:\users\0526p\anaconda3\lib\site-packages (from tensorflow) (1.12.1)
Requirement already satisfied: typing-extensions<=3.7.4 in c:\users\0526p\anaconda3\lib\site-packages (from tensorflow) (3.7.4.3)
Requirement already satisfied: gast==0.3.3 in c:\users\0526p\anaconda3\lib\site-packages (from tensorflow) (0.3.3)
Requirement already satisfied: wheel<0.35 in c:\users\0526p\anaconda3\lib\site-packages (from tensorflow) (0.36.2)
Collecting six<=1.15.0
  Using cached numpy-1.19.5-cp37-cp37m-win_amd64.whl (13.2 MB)
Requirement already satisfied: protobuf<=3.9.2 in c:\users\0526p\anaconda3\lib\site-packages (from tensorflow) (3.14.0)
Requirement already satisfied: h5py<=2.10.0 in c:\users\0526p\anaconda3\lib\site-packages (from tensorflow) (2.18.0)
Requirement already satisfied: keras-preprocessing<1.1.2 in c:\users\0526p\anaconda3\lib\site-packages (from tensorflow) (1.1.2)
Requirement already satisfied: google-pasta<=0.2 in c:\users\0526p\anaconda3\lib\site-packages (from tensorflow) (0.2.0)
Requirement already satisfied: absl-py<=0.18 in c:\users\0526p\anaconda3\lib\site-packages (from tensorflow) (0.11.0)
Requirement already satisfied: grpcio<=1.32.0 in c:\users\0526p\anaconda3\lib\site-packages (from tensorflow) (1.32.0)
Requirement already satisfied: markdown<=2.6.8 in c:\users\0526p\anaconda3\lib\site-packages (from tensorflow) (3.3.3)
Requirement already satisfied: requests<3,>2.21.0 in c:\users\0526p\anaconda3\lib\site-packages (from tensorflow) (2.22.0)
Requirement already satisfied: werkzeug<0.11.15 in c:\users\0526p\anaconda3\lib\site-packages (from tensorflow) (0.16.0)
Requirement already satisfied: tensorflow-plugin-wpt<1.6.0 in c:\users\0526p\anaconda3\lib\site-packages (from tensorflow) (1.6.0)
Requirement already satisfied: google-auth-oauthlib<0.5,>=0.4.1 in c:\users\0526p\anaconda3\lib\site-packages (from tensorflow) (0.4.2)
Requirement already satisfied: setuptools<41.0.0 in c:\users\0526p\anaconda3\lib\site-packages (from tensorflow) (41.4.0)
Requirement already satisfied: google-auth<2,>1.6.3 in c:\users\0526p\anaconda3\lib\site-packages (from tensorflow) (1.25.8)
Requirement already satisfied: cachetools<5.0,>2.0.0 in c:\users\0526p\anaconda3\lib\site-packages (from google-auth<2,>1.6.3->tensorflow) (4.2.1)
Requirement already satisfied: pyasn1-modules<0.2.1 in c:\users\0526p\anaconda3\lib\site-packages (from google-auth<2,>1.6.3->tensorflow) (0.2.8)
Requirement already satisfied: rsa<3.1.4 in c:\users\0526p\anaconda3\lib\site-packages (from google-auth<2,>1.6.3->tensorflow) (4.7)
Requirement already satisfied: requests-oauthlib<0.7.0 in c:\users\0526p\anaconda3\lib\site-packages (from google-auth-oauthlib<0.5,>0.4.1->tensorflow) (1.3.0)
Requirement already satisfied: importlib-metadata in c:\users\0526p\anaconda3\lib\site-packages (from markdown<2.6.8->tensorflow) (0.23)
Requirement already satisfied: pyasn1<0.5.0,>=0.4.6 in c:\users\0526p\anaconda3\lib\site-packages (from pyasn1-modules<0.2.1->google-auth<2,>1.6.3->tensorflow) (0.4.8)
```

## Chapter- 3

### Problem Definition and Objectives

#### 3.1 Facial Landmarks

Facial landmarks allow us to automatically infer the location of facial structures, including:  
 Eyes, Eyebrows, Nose, Mouth, Jawline

- To use facial landmarks to build a dataset of faces wearing face masks, we need to first start with an image of a person *not* wearing a face mask:



- From there, we apply face detection to compute the bounding box location of the face in the image. The next step is to apply face detection. Here we've used a deep learning method to perform face detection with OpenCV. Once we know *where* in the image the face is, we can extract the face Region of Interest (RoI).



- The next step is to extract the face ROI with OpenCV and NumPy slicing. And from there, we apply facial landmarks, allowing us to localize the eyes, nose, mouth, etc. Then, we detect facial landmarks using dlib so that we know where to place a mask on the face. Calculate the encoding of image and face distance of every image. Apply the same process on the real time image in the video frame. And compare the encoding of both the image (image in database and capture image from real time video frame). If both are matched then mark attendance else not continue to capture the face in the real time video frame.



### **3.2 Camera Analysis**

#### **3.2.1 Video Stream**

1. Optimum resolution for the module's operation: HD or Full HD.
2. Frame rate: 10 fps or more.
3. No mirrored (horizontally flipped) stream.

#### **3.2.2 Illumination and image quality**

1. Illumination of faces in the frame must be uniform.
2. If the camera is installed opposite a bright source of light (sun behind the entrance door, etc.), it is required to adjust the exposure or brightness in such a way that the face in the frame is light.
3. The image quality must be medium or better. Significant compression artefacts are not acceptable.
4. No blurring of moving people's faces is allowed.

#### **3.2.3 Scene and camera position**

1. The faces must be fully seen in the frame.
2. There must be no mirror surfaces giving reflections in the frame (glass, mirrors, etc.).
3. Strong lateral illumination (e.g., sunlight from the window) resulting in the overexposure of one part of the face is not acceptable.



# Chandigarh Engineering College Jhanjeri

## Mohali - 140307

### Department of Computer Science & Engineering

4. The camera may be placed above the face level, directly facing the people to be recognized. In such a case, the camera elevation angle must not exceed 35°. The camera must directly face the people to be recognized.

### 3.3 Skills

Knowledge of python is first and most compulsory key for making any Machine Learning and AI projects. Must know how and where to uses loop, conditional statements. You have to be good command in functions and OOPs. You can learn python on my github account, link is given in chapter 6 (reference).

### 3.4 Libraries

For building this model we uses some of the useful libraries. We need to install libraries using the above method we discuss above in chapter 2 (“steps to install libraries” topic). Libraries we need to build this system are as given below:

1. **tkinter:** Tkinter is python’s default GUI library. It is based on the Tk toolkit, originally designed for the Tool Command Language (TCL). It has varies according to the other scripting languages, including Perl (Perl/ Tk), Ruby (Ruby/ Tk), and Python (Tkinter). Tk have the most robust set of GUI building blocks, but it is fairly simple to use, and with it, you can build GUIs that run on most platforms. Python, along with Tkinter, provides a fast and exciting way to build useful applications that would have taken much longer if you have to program directly in C/ C++ with the native windowing system’s libraries.  
You will use basic building blocks known as widgets to piece together the desired. Once you get Tkinter up on your system, it will take less than 15 minutes to get your first GUI application running.
2. **cv2:** For image and video processing, we use the cv2 library for read the file directory. OpenCV (Open Source Computer Vision Library)-Python is a library of python binding designed to solve computer vision problems.  
cv2.imread() method loads an image from the specified file. If the image cannot be read (because of missing file, improper permissions, unsupported or invalid format then this method returns an empty matrix.

**Note:** The image should be in the working directory or a full path of image should be given.



3. **numpy:** NumPy is the fundamental package for scientific computing in Python. It is a Python library that provides a multidimensional array object, derived objects such as masked arrays and matrices and performs fast operations on array like mathematical, logical, shape, manipulation, sorting, selecting, I/O, basic linear algebra, basic statistical operations and many more.  
.
4. **face\_recognition:** Face recognition is a method of identifying or verifying the identity of persons using their face. Face recognition is used for detecting faces in the images or in video frames. It contains lots of function such as “encoding” we use it for encode faces from images or video frames. “face distance” calculate face distance to check the best match “Compare” compare two or more faces using the face encoding if both face encode is match then its face distance is less and vice versa.  
Facial recognition uses computer-generated filters to transform face images into numerical expressions that can be compared to determine their similarity. These filters are usually generated by using deep learning, which uses artificial neural networks to process data
5. **os:** The OS library in python provides functions for interacting with the operating systems. It is very efficient for handling file directories or handling file directories in GUI. This library provides a portable way of using operating system-dependent functionality. The \*os\* and \*os.path\* modules include many functions in this library used for interact with the file system.
6. **datetime:** It is used to deal with the date or time. 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. We are using the library for that purpose for is datetime.
7. **pytz:** It is used for grab the Indian standard time Asia/ Kolkata. Without depends on CPU or any other devices time source. It grabs automatically when it will be execute. It supports almost all time zones.  
**Syntax:** astimezone(t)  
**Parameter:** t is time to be converted  
**Return:** converted timezone



**Chandigarh Engineering College Jhanjeri**

**Mohali - 140307**

**Department of Computer Science & Engineering**

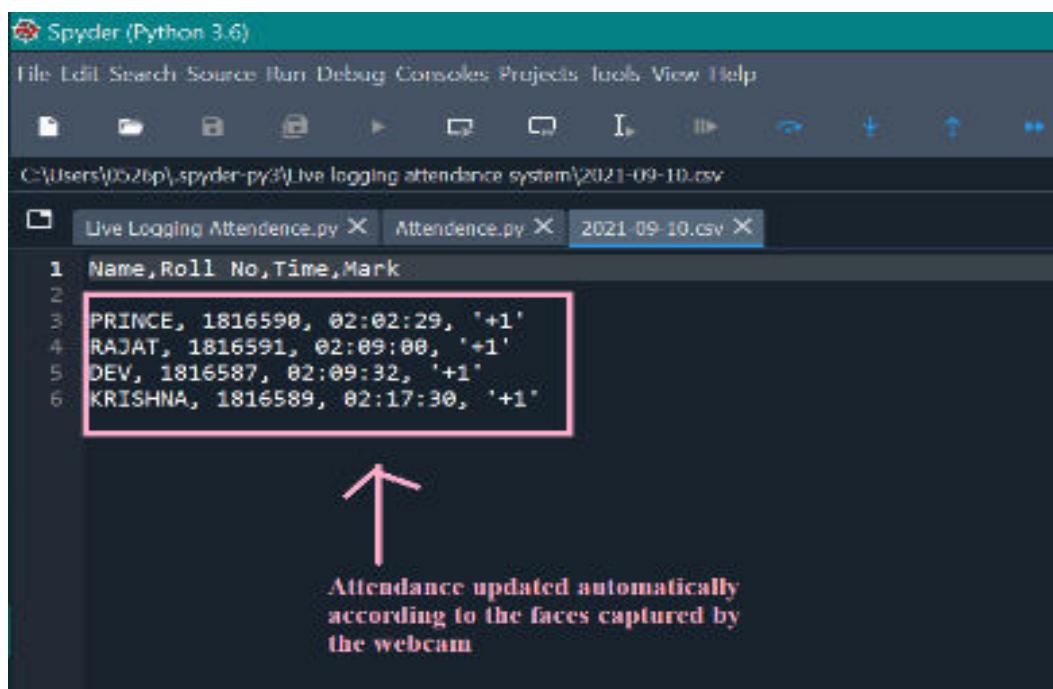
8. **csv:** A Comma Separated Value (CSV) file is a text file that has a specific format which allows data to be saved in a table structured format. It uses to create CSV data sheet for mark attendance or update the attendance. We can check it on externally. Or we can also convert into excel files according to their needs and desire. It is commonly used file extension when it comes to spreadsheets.
9. **PIL:** Python Imaging Library is a free and open-source library. It is used for opening, manipulating, and saving many different image file formats. Pillow supports a large number of image file formats including BMP, PNG, JPEG, and TIFF.
10. **pathlib:** A pathlib library in python provides various classes representing file system paths with semantics appropriate for different operating systems. File paths can be represented by proper path objects instead of plain strings as before. These objects make code dealing with file paths. Easier to read, especially because / is used to join paths together.

## Chapter-4

### Design and Implementation

#### 4.1 Database

**4.1.1 Database Categories:** We create Comma Separated Values (CSV) and store the data whatever output given by model. Data categories are tables of data, which are organized by rows and columns. Columns are also known as data fields. A row of data has entries for one or more columns in the category. When you add a data field onto a report you are seeing the information in one column of data for every row in the category.



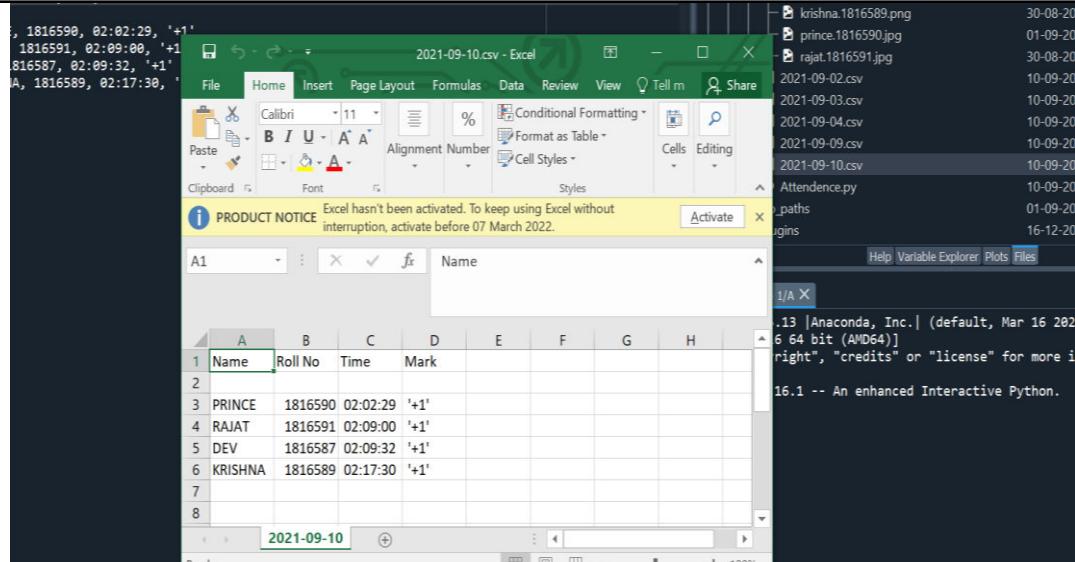
The screenshot shows the Spyder Python 3.6 IDE interface. The title bar says "Spyder (Python 3.6)". The menu bar includes File, Edit, Search, Source, Run, Debug, Consoles, Projects, Tools, View, Help. Below the menu is a toolbar with various icons. The main area shows a file path: C:\Users\0526p\spyder-py3\live logging attendance system\2021-09-10.csv. There are three tabs open: "Live Logging Attendance.py", "Attendance.py", and "2021-09-10.csv". The "2021-09-10.csv" tab is active and displays the following CSV data:

	Name	Roll No.	Time	Mark
1	PRINCE	1816598	02:02:29	+'1'
2	RAJAT	1816591	02:09:00	+'1'
3	DEV	1816587	02:09:32	+'1'
4	KRISHNA	1816589	02:17:30	+'1'

A pink arrow points from the text "Attendance updated automatically according to the faces captured by the webcam" to the data in the CSV file.

Figure 4.1.1: Database Categories

**4.1.2 Data Model:** Then we convert the CSV file to the Excel file. A database model is a type of data model that determines the logical structure of database and fundamentally determines in which manner data can be stored, organized and manipulated.



The screenshot shows an Excel spreadsheet titled '2021-09-10.csv'. The columns are labeled 'Name', 'Roll No', 'Time', and 'Mark'. The data consists of the following rows:

	Name	Roll No	Time	Mark
1	PRINCE	1816590	02:02:29	'+1'
2	RAJAT	1816591	02:09:00	'+1'
3	DEV	1816587	02:09:32	'+1'
4	KRISHNA	1816589	02:17:30	'+1'

Figure 4.1.2: Data Model

## 4.2 Flow Chart

A flowchart is simply a graphical representation of steps. It shows steps in sequential order and is widely used in presenting the flow of algorithms, workflow or processes. Typically, a flowchart shows the steps as boxes of various kinds, and their order by connecting them with arrows.

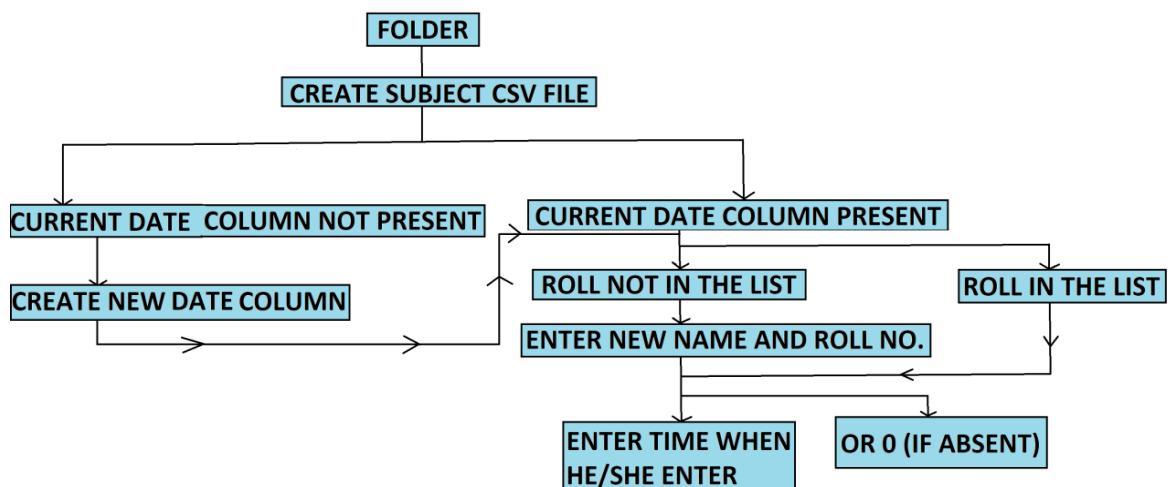


Figure 4.2: Flowchart of LLFRAS

## 4.3 ER Diagram

ER Diagram stands for Entity Relationship Diagram, also known as ERD is a diagram that displays the relationship of entity sets stored in a database. In other words, ER diagrams help to explain the logical structure of databases. At first look, an ER diagram looks very similar to the flowchart. However, ER Diagram includes many specialized symbols, and its meanings make this model unique. The purpose of ER Diagram is to represent the entity framework infrastructure.

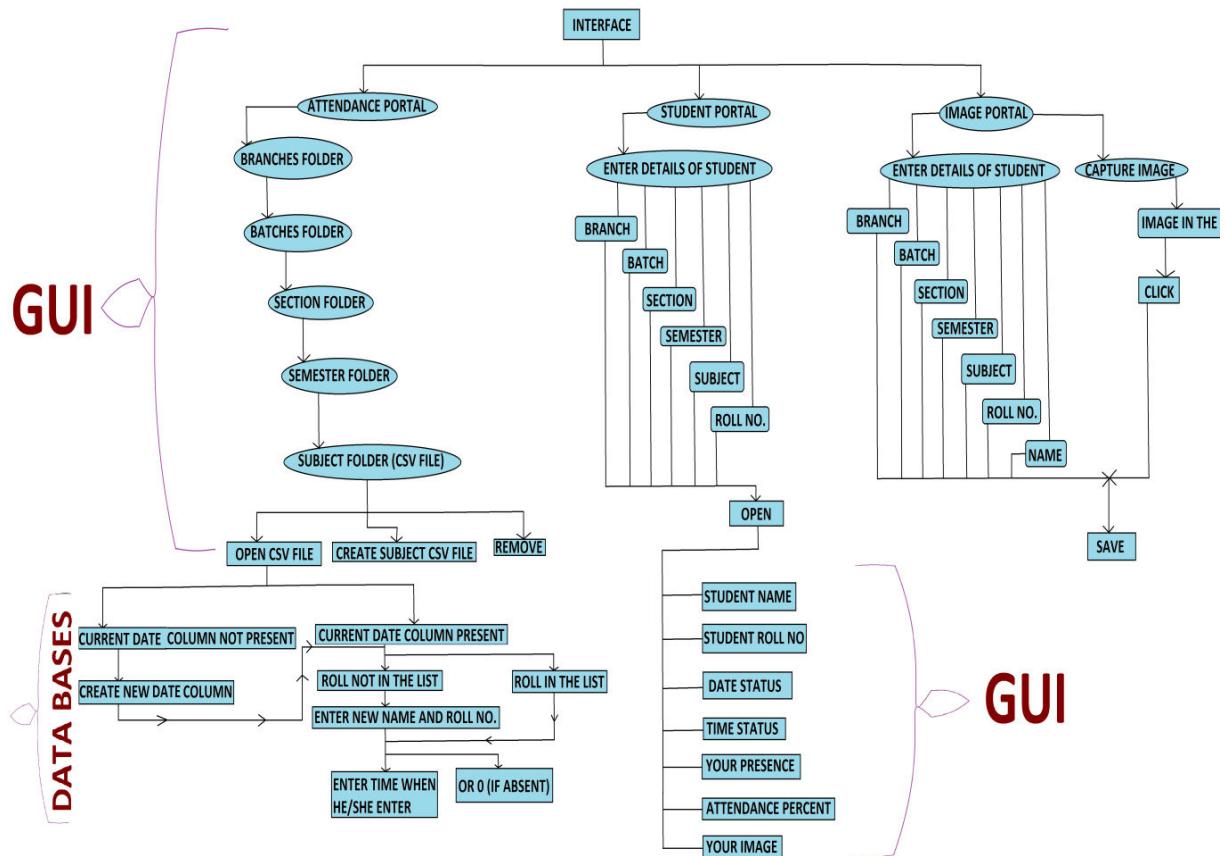


Figure 4.3: ER Diagram of LLFRAS

## 4.4 Methods to Build Live Logging Facial Recognition Attendance Systems

### 4.4.1 Loading Image Data

1. Method to read image
2. Method to display image
3. Resizing image
4. Video frames

### 4.4.2 Encoding of Load Image

1. Encode all the images present in the directory
2. Store all the encode list inside the list



**Chandigarh Engineering College Jhanjeri**

**Mohali - 140307**

## **Department of Computer Science & Engineering**

3. While use these encode at the time of compare faces
4. Calculate the face distance of all images present in the directory
5. Lower face distance is the best match and height face distance is the worst match

### **4.4.3 Features Detection**

1. Capture web cam images
2. Using face recognition to extract the ROI
3. ROI(Region Of Interest)

### **4.4.4 Video Streaming & Real Times**

1. Method to capture video
1. Looping through the Frames
2. Video Streaming
3. Encode the faces in the current frame
4. Compare the captured face in real time

### **4.4.5 Create and Update CSV sheet**

1. Create csv sheet of particular date
2. Update name, roll, time of matched persons
3. Check data sheet if name is already present then it is not overwrite or update again

## **4.5 Coding of Live Logging Facial Recognition Attendance System**

From line 7 to 16 import the libraries which is used in this project. In **line 7** import all the functions which is perform by the **tkinter** library which is used for GUI making interface. It provides the interfaces for user to access whatever information and task he want to perform it.

In **line 8** import cv2 library for image and video processing. **Line 9** is **numpy** library which is used for store the pixels array of the images captured in video frame. **Line 10** is **face-recognition** library which is used for comparing database images with captured frame images. **Line 11** is **os** library which is used for file handling. **Line 12** is **datetime** library is used for get the actual time and date.

**Line 13** is **pytz** library which is used for getting time of a particular timezone. **Line 14** is **csv** libraries which is used for store values seprated by commas. **Line 15** is **PIL (pillow)** library for handling images in Tkinter. **Line 16** is **pathlib** library which is used for getting the path of your os.



# Chandigarh Engineering College Jhanjeri

## Mohali - 140307

### Department of Computer Science & Engineering

The screenshot shows the Spyder Python IDE interface. The code editor displays a Python script named 'Merge Capture with Get Attendance.py'. The script includes imports for Tkinter, cv2, numpy, face\_recognition, os, datetime, pytz, and csv, along with PIL for ImageTk and Path from pathlib. It defines functions for creating a subject CSV file, finding encodings of images, and encoding sections of the database. The file browser on the right shows two directories: 'Attendance Directory' and 'Image Directory', both modified on 01-10-2021 at 18:31 and 18:45 respectively. The Python console at the bottom shows the command 'In [2]: runcell(0, 'C:/Users/Lenovo/.spyder-py3/LLFRAS/Merge Capture with Get Attendance.py')' and 'In [3]:'. The taskbar at the bottom of the screen shows various application icons.

```
temp.py | Live Logging Attendance 3.py | Merge Capture with Get Attendance.py

1 # -*- coding: utf-8 -*-
2 """
3 Created on Fri Sep 24 02:51:49 2021
4
5 @author: B526p
6 """
7 from tkinter import *
8 import cv2
9 import numpy as np
10 import face_recognition
11 import os
12 import datetime
13 import pytz
14 import csv
15 from PIL import ImageTk, Image
16 from pathlib import Path
17
18 root = Tk()
19
20 def createSubjectCsvFile(getInput, path_6):
21     path = path_6 + '/' + getInput + '.csv'
22     with open(path, 'w', newline='') as f:
23         theWriter = csv.writer(f)
24         theWriter.writerow(['Name', 'Roll No'])
25
26 def findEncodings(images):
27     encodeList = []
28     for img in images:
29         img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
30         encode = face_recognition.face_encodings(img)[0]
31         encodeList.append(encode)
32     return encodeList
33
34 def encoding_section():
35     global images, classNames, RollNo
36     path_13 = new_path + '/Image Directory/branch'
37     path_14 = path_13 + '/' + image_directory_section
38
39     images = []
40     classNames = []
```

In line 18 we make a root window where we put widgets and icons. In Line 20 to 24 create a function `createSubjectCsvFile` which makes subjects csv file sheets. In line 26 to 32 we encode the database images using `face_encoding` and return encodes list of all images.

In line 34 make a function where we make a list of images which we get from assigned directory. All image name are preceded with person name and roll no or person id. So, we split it the also we make the list of person name and roll no/person id.

On every particular date a new column is made on that particular date. Before we made new date column of that particular date we need to check first, is there any column of that date is present or not. In line 51, function returns the last date column name which is made recently.



**Chandigarh Engineering College Jhanjeri  
Mohali - 140307**

**Department of Computer Science & Engineering**

The screenshot shows the Spyder Python IDE interface. The top menu bar includes File, Edit, Search, Source, Run, Debug, Consoles, Projects, Tools, View, Help. The left sidebar has tabs for temp.py, Live Logging, Attendance 3.py, and Merge Capture with Get Attendance.py. The main code editor window displays Python code for merging attendance data. The right sidebar shows a file browser with two entries: 'Attendance Directory' (modified 01-10-2021 18:31) and 'Image Directory' (modified 01-10-2021 18:45). Below the file browser is a 'Console I/A' tab with two code cells. Cell [2] contains the command `In [2]: runcell(0, 'C:/Users/Lenovo/.spyder-py3/LLFRAS/Merge Capture with Get Attendance.py')`. Cell [3] is currently active with the prompt `In [3]:`. The bottom status bar shows LSP Python: ready, conda env: lib (Python 3.8.11), Line 1, Col 1, 07:24, 18°C Sunny, ENG, and Mem 60%.

```
temp.py X Live Logging Attendance 3.py X Merge Capture with Get Attendance.py X

34 def encoding_section():
35     global images, classNames, RollNo
36     path_13 = new_path + '/Image Directory/branch'
37     path_14 = path_13 + '/' + image_directory_section
38
39     images = []
40     classNames = []
41     RollNo = []
42
43     myList_1 = os.listdir(path_14)
44     for cl in myList_1:
45         curImg = cv2.imread(f'{path_14}/{cl}')
46         images.append(curImg)
47         text = os.path.splitext(cl)[0]
48         classNames.append(text.split('.')[0])
49         RollNo.append(text.split('.')[1])
50
51 def columnCheck(getInput, path_6):
52     path = path_6 + '/' + getInput+'.csv'
53     with open(path, 'r') as read_obj:
54         theReader = csv.reader(read_obj, delimiter = ',')
55         csvList = []
56         for l in theReader:
57             csvList.append(l)
58     return csvList[0][-1]
59
60 def fillStudentNameAndRoll(name, roll, getInput, path_6):
61     path = path_6 + '/' + getInput+'.csv'
62     with open(path, 'r+') as f:
63         myDataList = csv.reader(f)
64         rollList = []
65         for line in myDataList:
66             if line != []:
67                 rollList.append(line[1])
68
69         if roll not in rollList:
70             f.writelines(f'\n{name},{roll}')
71             New_roll = roll
72             return New_roll
73
```

In line 60, `fillStudentNameAndRoll` function it open the csv sheet in read and write mode using ‘`r+`’ parameters. We make empty `rollList` list which stores the roll no of all person name contain in the csv sheet. Then it create a new row of that person name and roll no/person id if it is not present in the csv sheet. If it is already present then pass the function.

The screenshot shows the Spyder Python IDE interface. The top menu bar includes File, Edit, Search, Source, Run, Debug, Consoles, Projects, Tools, View, Help. The title bar shows 'C:/Users/Lenovo/.spyder-py3/LLFRAS/Merge Capture with Get Attendance.py'. The left pane contains three tabs: 'temp.py' (empty), 'Live Logging Attendance.py' (empty), and 'Merge Capture with Get Attendance.py'. The code editor displays Python code for creating directory structures and handling file paths. The right pane shows a file browser with two entries: 'Attendance Directory' (modified 01-10-2021 18:31) and 'Image Directory' (modified 01-10-2021 18:45). Below the file browser is a 'Console 1/A X' tab, which has run cell 2 with the command 'runcell(0, 'C:/Users/Lenovo/.spyder-py3/LLFRAS/Merge Capture with Get Attendance.py')'. The bottom status bar shows 'LSP Python: ready', 'conda: env\_dib (Python 3.8.11)', line numbers 19, 19, Col 1, UTF-8, CR/LF, RW, Mem 61%, and system information like CPU, RAM, and battery level.

```
1436 def image_path():
1437     global path_1, path_8
1438     path_1 = None
1439     path_8 = new_path + '/Image Directory'
1440     try:
1441         os.mkdir(path_8)
1442     except FileNotFoundError:
1443         pass
1444
1445 def attendance_path():
1446     global path_1, path_8
1447     path_8 = None
1448     path_1 = new_path + '/Attendance Directory'
1449     try:
1450         os.mkdir(path_1)
1451     except FileNotFoundError:
1452         pass
1453
1454 def get_attendance_path():
1455     global path_15
1456     path_15 = new_path + '/Attendance Directory/branch'
1457
1458 global new_path
1459 get_path = str(Path.home())
1460 spli = get_path.split('\\')
1461 new_path = '/'.join(spli)+'/Desktop/Attendance'
1462 try:
1463     os.mkdir(new_path)
1464 except FileNotFoundError:
1465     pass
1466 os.chdir(new_path)
1467
1468 proceed_button = Button(root, text = 'Attendance Portal', command = lambda: [root.withdraw(), attendance])
1469 student_button = Button(root, text = "Student's Portal", command = lambda: [root.withdraw(), get_attend])
1470 image_button = Button(root, text = 'Image Portal', command = lambda: [root.withdraw(), image_path()])
1471 label(root, text = '').grid(row = row_value + 1, column = 5)
1472 button_exit = Button(root, text = 'Exit', command = root.destroy, padx = 10).grid(row = row_value+2, col
1473 1474
root.mainloop()
```



# Chandigarh Engineering College Jhanjeri

## Mohali - 140307

### Department of Computer Science & Engineering

The screenshot shows the Spyder Python IDE interface. The code editor displays a script named 'temp.py' which merges attendance data from multiple branches. The code uses os.listdir and os.makedirs to handle directory structures, and a lambda function to create buttons for each branch. The file path is C:/Users/Lenovo/.spyder-py3/LLFRAS/Merge Capture with Get Attendance.py.

```
1240 def branch_directories():
1241     global branch_window
1242     branch_window = Toplevel()
1243
1244     branch = ['Computer science', 'Electrical', 'Mechanical', 'Civil', 'Agriculture']
1245     if path_8 == None:
1246         global branch_list, path_9
1247         path_9 = None
1248         os.chdir(path_1)
1249         if 'branch' not in os.listdir(path_1):
1250             global path_2
1251             path_2 = path_1 + '/' + 'branch'
1252             os.mkdir(path_2)
1253             os.chdir(path_2)
1254             for b in branch:
1255                 os.mkdir(path_2 + '/' + b)
1256             elif 'branch' in os.listdir(path_1):
1257                 path_2 = path_1 + '/' + 'branch'
1258                 os.chdir(path_2)
1259                 branch_window.title(path_2)
1260                 branch_list = os.listdir(path_9)
1261
1262     branch_list = os.listdir(path_9)
1263     branch_window.iconbitmap('C:/Users/Lenovo/Downloads/images.ico')
1264
1265     def show_forward():
1266         Button(root, text='>', command= lambda: [branch_window.deiconify(), root.withdraw()], padx = 10)
1267
1268     Button(branch_window, text='<', command= lambda: [branch_window.withdraw(), root.deiconify(), show_forward()])
1269     Button(branch_window, text='Start', command= lambda:[branch_window.withdraw(), root.deiconify()])
1270     Button(branch_window, text='Home', state = DISABLED, padx = 20).grid(row=0, column=2)
1271     Button(branch_window, text='>>', state = DISABLED, padx=10).grid(row=0, column=3)
1272
1273     new = Button(branch_window, text='New', command=new_branch).grid(row=0, column=5)
1274
1275
1276
1277
1278
1279
1280
1281
1282
1283
1284
1285
1286
1287
1288
1289
1290
1291
1292
1293
1294
1295
1296
1297
1298
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1310
1311
1312
1313
1314
1315
1316
1317
1318
1319
1320
1321
1322
1323
1324
1325
1326
1327
1328
1329
1330
1331
1332
1333
1334
1335
1336
1337
1338
1339
1340
1341
1342
1343
1344
1345
1346
1347
1348
1349
1350
1351
1352
1353
1354
1355
1356
1357
1358
1359
1360
1361
1362
1363
1364
1365
1366
1367
1368
1369
1370
1371
1372
1373
1374
1375
1376
1377
1378
1379
1380
1381
1382
1383
1384
1385
1386
1387
1388
1389
1390
1391
1392
1393
1394
1395
1396
1397
1398
1399
1400
1401
1402
1403
1404
1405
1406
1407
1408
1409
1410
1411
1412
1413
1414
1415
1416
1417
1418
1419
1420
1421
1422
1423
1424
1425
1426
1427
1428
1429
1430
1431
1432
1433
1434
1435
1436
1437
1438
1439
1440
1441
1442
1443
1444
1445
1446
1447
1448
1449
1450
1451
1452
1453
1454
1455
1456
1457
1458
1459
1460
1461
1462
1463
1464
1465
1466
1467
1468
1469
1470
1471
1472
1473
1474
1475
1476
1477
1478
1479
1480
1481
1482
1483
1484
1485
1486
1487
1488
1489
1490
1491
1492
1493
1494
1495
1496
1497
1498
1499
1500
1501
1502
1503
1504
1505
1506
1507
1508
1509
1510
1511
1512
1513
1514
1515
1516
1517
1518
1519
1520
1521
1522
1523
1524
1525
1526
1527
1528
1529
1530
1531
1532
1533
1534
1535
1536
1537
1538
1539
1540
1541
1542
1543
1544
1545
1546
1547
1548
1549
1550
1551
1552
1553
1554
1555
1556
1557
1558
1559
1560
1561
1562
1563
1564
1565
1566
1567
1568
1569
1570
1571
1572
1573
1574
1575
1576
1577
1578
1579
1580
1581
1582
1583
1584
1585
1586
1587
1588
1589
1590
1591
1592
1593
1594
1595
1596
1597
1598
1599
1600
1601
1602
1603
1604
1605
1606
1607
1608
1609
1610
1611
1612
1613
1614
1615
1616
1617
1618
1619
1620
1621
1622
1623
1624
1625
1626
1627
1628
1629
1630
1631
1632
1633
1634
1635
1636
1637
1638
1639
1640
1641
1642
1643
1644
1645
1646
1647
1648
1649
1650
1651
1652
1653
1654
1655
1656
1657
1658
1659
1660
1661
1662
1663
1664
1665
1666
1667
1668
1669
1670
1671
1672
1673
1674
1675
1676
1677
1678
1679
1680
1681
1682
1683
1684
1685
1686
1687
1688
1689
1690
1691
1692
1693
1694
1695
1696
1697
1698
1699
1700
1701
1702
1703
1704
1705
1706
1707
1708
1709
1710
1711
1712
1713
1714
1715
1716
1717
1718
1719
1720
1721
1722
1723
1724
1725
1726
1727
1728
1729
1730
1731
1732
1733
1734
1735
1736
1737
1738
1739
1740
1741
1742
1743
1744
1745
1746
1747
1748
1749
1750
1751
1752
1753
1754
1755
1756
1757
1758
1759
1760
1761
1762
1763
1764
1765
1766
1767
1768
1769
1770
1771
1772
1773
1774
1775
1776
1777
1778
1779
1780
1781
1782
1783
1784
1785
1786
1787
1788
1789
1790
1791
1792
1793
1794
1795
1796
1797
1798
1799
1800
1801
1802
1803
1804
1805
1806
1807
1808
1809
1810
1811
1812
1813
1814
1815
1816
1817
1818
1819
1820
1821
1822
1823
1824
1825
1826
1827
1828
1829
1830
1831
1832
1833
1834
1835
1836
1837
1838
1839
1840
1841
1842
1843
1844
1845
1846
1847
1848
1849
1850
1851
1852
1853
1854
1855
1856
1857
1858
1859
1860
1861
1862
1863
1864
1865
1866
1867
1868
1869
1870
1871
1872
1873
1874
1875
1876
1877
1878
1879
1880
1881
1882
1883
1884
1885
1886
1887
1888
1889
1890
1891
1892
1893
1894
1895
1896
1897
1898
1899
1900
1901
1902
1903
1904
1905
1906
1907
1908
1909
1910
1911
1912
1913
1914
1915
1916
1917
1918
1919
1920
1921
1922
1923
1924
1925
1926
1927
1928
1929
1930
1931
1932
1933
1934
1935
1936
1937
1938
1939
1940
1941
1942
1943
1944
1945
1946
1947
1948
1949
1950
1951
1952
1953
1954
1955
1956
1957
1958
1959
1960
1961
1962
1963
1964
1965
1966
1967
1968
1969
1970
1971
1972
1973
1974
1975
1976
1977
1978
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1989
1990
1991
1992
1993
1994
1995
1996
1997
1998
1999
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
2020
2021
2022
2023
2024
2025
2026
2027
2028
2029
2030
2031
2032
2033
2034
2035
2036
2037
2038
2039
2040
2041
2042
2043
2044
2045
2046
2047
2048
2049
2050
2051
2052
2053
2054
2055
2056
2057
2058
2059
2060
2061
2062
2063
2064
2065
2066
2067
2068
2069
2070
2071
2072
2073
2074
2075
2076
2077
2078
2079
2080
2081
2082
2083
2084
2085
2086
2087
2088
2089
2090
2091
2092
2093
2094
2095
2096
2097
2098
2099
2099
2100
2101
2102
2103
2104
2105
2106
2107
2108
2109
2109
2110
2111
2112
2113
2114
2115
2116
2117
2118
2119
2119
2120
2121
2122
2123
2124
2125
2126
2127
2128
2129
2129
2130
2131
2132
2133
2134
2135
2136
2137
2138
2139
2139
2140
2141
2142
2143
2144
2145
2146
2147
2148
2149
2149
2150
2151
2152
2153
2154
2155
2156
2157
2158
2159
2159
2160
2161
2162
2163
2164
2165
2166
2167
2168
2169
2169
2170
2171
2172
2173
2174
2175
2176
2177
2178
2179
2179
2180
2181
2182
2183
2184
2185
2186
2187
2188
2189
2189
2190
2191
2192
2193
2194
2195
2196
2197
2198
2199
2199
2200
2201
2202
2203
2204
2205
2206
2207
2208
2209
2209
2210
2211
2212
2213
2214
2215
2216
2217
2218
2219
2219
2220
2221
2222
2223
2224
2225
2226
2227
2228
2229
2229
2230
2231
2232
2233
2234
2235
2236
2237
2238
2239
2239
2240
2241
2242
2243
2244
2245
2246
2247
2248
2249
2249
2250
2251
2252
2253
2254
2255
2256
2257
2258
2259
2259
2260
2261
2262
2263
2264
2265
2266
2267
2268
2269
2269
2270
2271
2272
2273
2274
2275
2276
2277
2278
2279
2279
2280
2281
2282
2283
2284
2285
2286
2287
2288
2289
2289
2290
2291
2292
2293
2294
2295
2296
2297
2298
2298
2299
2299
2300
2301
2302
2303
2304
2305
2306
2307
2308
2309
2309
2310
2311
2312
2313
2314
2315
2316
2317
2318
2319
2319
2320
2321
2322
2323
2324
2325
2326
2327
2328
2329
2329
2330
2331
2332
2333
2334
2335
2336
2337
2338
2339
2339
2340
2341
2342
2343
2344
2345
2346
2347
2348
2349
2349
2350
2351
2352
2353
2354
2355
2356
2357
2358
2359
2359
2360
2361
2362
2363
2364
2365
2366
2367
2368
2369
2369
2370
2371
2372
2373
2374
2375
2376
2377
2378
2379
2379
2380
2381
2382
2383
2384
2385
2386
2387
2388
2389
2389
2390
2391
2392
2393
2394
2395
2396
2397
2398
2398
2399
2399
2400
2401
2402
2403
2404
2405
2406
2407
2408
2409
2409
2410
2411
2412
2413
2414
2415
2416
2417
2418
2419
2419
2420
2421
2422
2423
2424
2425
2426
2427
2428
2429
2429
2430
2431
2432
2433
2434
2435
2436
2437
2438
2439
2439
2440
2441
2442
2443
2444
2445
2446
2447
2448
2449
2449
2450
2451
2452
2453
2454
2455
2456
2457
2458
2459
2459
2460
2461
2462
2463
2464
2465
2466
2467
2468
2469
2469
2470
2471
2472
2473
2474
2475
2476
2477
2478
2479
2479
2480
2481
2482
2483
2484
2485
2486
2487
2488
2489
2489
2490
2491
2492
2493
2494
2495
2496
2497
2498
2498
2499
2499
2500
2501
2502
2503
2504
2505
2506
2507
2508
2509
2509
2510
2511
2512
2513
2514
2515
2516
2517
2518
2519
2519
2520
2521
2522
2523
2524
2525
2526
2527
2528
2529
2529
2530
2531
2532
2533
2534
2535
2536
2537
2538
2539
2539
2540
2541
2542
2543
2544
2545
2546
2547
2548
2549
2549
2550
2551
2552
2553
2554
2555
2556
2557
2558
2559
2559
2560
2561
2562
2563
2564
2565
2566
2567
2568
2569
2569
2570
2571
2572
2573
2574
2575
2576
2577
2578
2579
2579
2580
2581
2582
2583
2584
2585
2586
2587
2588
2589
2589
2590
2591
2592
2593
2594
2595
2596
2597
2598
2598
2599
2599
2600
2601
2602
2603
2604
2605
2606
2607
2608
2609
2609
2610
2611
2612
2613
2614
2615
2616
2617
2618
2619
2619
2620
2621
2622
2623
2624
2625
2626
2627
2628
2629
2629
2630
2631
2632
2633
2634
2635
2636
2637
2638
2639
2639
2640
2641
2642
2643
2644
2645
2646
2647
2648
2649
2649
2650
2651
2652
2653
2654
2655
2656
2657
2658
2659
2659
2660
2661
2662
2663
2664
2665
2666
2667
2668
2669
2669
2670
2671
2672
2673
2674
2675
2676
2677
2678
2679
2679
2680
2681
2682
2683
2684
2685
2686
2687
2688
2689
2689
2690
2691
2692
2693
2694
2695
2696
2697
2698
2698
2699
2699
2700
2701
2702
2703
2704
2705
2706
2707
2708
2709
2709
2710
2711
2712
2713
2714
2715
2716
2717
2718
2719
2719
2720
2721
2722
2723
2724
2725
2726
2727
2728
2729
2729
2730
2731
2732
2733
2734
2735
2736
2737
2738
2739
2739
2740
2741
2742
2743
2744
2745
2746
2747
2748
2749
2749
2750
2751
2752
2753
2754
2755
2756
2757
2758
2759
2759
2760
2761
2762
2763
2764
2765
2766
2767
2768
2769
2769
2770
2771
2772
2773
2774
2775
2776
2777
2778
2779
2779
2780
2781
2782
2783
2784
2785
2786
2787
2788
2789
2789
2790
2791
2792
2793
2794
2795
2796
2797
2798
2798
2799
2799
2800
2801
2802
2803
2804
2805
2806
2807
2808
2809
2809
2810
2811
2812
2813
2814
2815
2816
2817
2818
2819
2819
2820
2821
2822
2823
2824
2825
2826
2827
2828
2829
2829
2830
2831
2832
2833
2834
2835
2836
2837
2838
2839
2839
2840
2841
2842
2843
2844
2845
2846
2847
2848
2849
2849
2850
2851
2852
2853
2854
2855
2856
2857
2858
2859
2859
2860
2861
2862
2863
2864
2865
2866
2867
2868
2869
2869
2870
2871
2872
2873
2874
2875
2876
2877
2878
2879
2879
2880
2881
2882
2883
2884
2885
2886
2887
2888
2889
2889
2890
2891
2892
2893
2894
2895
2896
2897
2898
2898
2899
2899
2900
2901
2902
2903
2904
2905
2906
2907
2908
2909
2909
2910
2911
2912
2913
2914
2915
2916
2917
2918
2919
2919
2920
2921
2922
2923
2924
2925
2926
2927
2928
2929
2929
2930
2931
2932
2933
2934
2935
2936
2937
2938
2939
2939
2940
2941
2942
2943
2944
2945
2946
2947
2948
2949
2949
2950
2951
2952
2953
2954
2955
2956
2957
2958
2959
2959
2960
2961
2962
2963
2964
2965
2966
2967
2968
2969
2969
2970
2971
2972
2973
2974
2975
2976
2977
2978
2979
2979
2980
2981
2982
2983
2984
2985
2986
2987
2988
2989
2989
2990
2991
2992
2993
2994
2995
2996
2997
2998
2998
2999
2999
3000
3001
3002
3003
3004
3005
3006
3007
3008
3009
3009
3010
3011
3012
3013
3014
3015
3016
3017
3018
3019
3019
3020
3021
3022
3023
3024
3025
3026
3027
3028
3029
3029
3030
3031
3032
3033
3034
3035
3036
3037
3038
3038
3039
3039
3040
3041
3042
3043
3044
3045
3046
3047
3048
3048
3049
3049
3050
3051
3052
3053
3054
3055
3056
3057
3058
3059
3059
3060
3061
3062
3063
3064
3065
3066
3067
3068
3069
3069
3070
3071
3072
3073
3074
3075
3076
3077
3078
3079
3079
3080
3081
3082
3083
3084
3085
3086
3087
3088
3089
3089
3090
3091
3092
3093
3094
3095
3096
3097
3097
3098
3099
3099
3100
3101
3102
3103
3104
3105
3106
3107
3108
3109
3109
3110
3111
3112
3113
3114
3115
3116
3117
3118
3119
3119
3120
3121
3122
3123
3124
3125
3126
3127
3128
3129
3129
3130
3131
3132
3133
3134
3135
3136
3137
3138
3139
3139
3140
3141
3142
3143
3144
3145
3146
3147
3148
3148
3149
3149
3150
3151
3152
3153
3154
3155
3156
3157
3158
3159
3159
3160
3161
3162
3163
3164
3165
3166
3167
3168
3169
3169
3170
3171
3172
3173
3174
3175
3176
3177
3178
3179
3179
3180
3181
3182
3183
3184
3185
3186
3187
3188
3189
3189
3190
3191
3192
3193
3194
3195
3196
3196
3197
3198
3199
3199
3200
3201
3202
3203
3204
3205
3206
3207
3208
3209
3209
3210
3211
3212
3213
3214
3215
3216
3217
3218
3219
3219
3220
3221
3222
3223
3224
3225
3226
3227
3
```



# Chandigarh Engineering College Jhanjeri

## Mohali - 140307

### Department of Computer Science & Engineering

The screenshot shows the Spyder Python IDE interface. The left pane displays the Python script `Merge Capture with Get Attendance.py`. The right pane shows the file explorer with two directories: `Attendance Directory` and `Image Directory`, both modified on 01-10-2021 at 18:31. Below the file explorer is the IPython console, which contains the command `In [2]: runcell(0, 'C:/Users/Lenovo/.spyder-py3/LLFRAS/Merge Capture with Get Attendance.py')`. The status bar at the bottom indicates the system is at 18°C, sunny, and the date is 22-10-2021.

```
def makeNewHeader(getInput, path_6):
    path = path_6 + '/' + getInput+'.csv'
    rowList = []
    with open(path, 'r') as f:
        theReader = csv.reader(f)
        for row in theReader:
            if row != []:
                rowList.append(row)

    with open(path, 'w') as f:
        writer = csv.writer(f)
        for i in rowList:
            if rowList.index(i) == 0:
                i.append(now2)
            writer.writerow(i)

def markAttendance(getInput, roll, path_6):
    path = path_6 + '/' + getInput+'.csv'
    with open(path, 'r+') as f:
        myDataList = csv.reader(f)
        rowList = []
        rollList = []
        dateList = []
        for line in myDataList:
            if line != []:
                rowList.append(line[1])
                dateList.append(line[-1])
                rowList.append(line)
        indexPosition = rollList.index(roll)
        value = dateList[indexPosition]

    with open(path, 'w') as f:
        writer = csv.writer(f)
        if value == '0':
            for i in rowList:
                if rowList.index(i) == indexPosition:
                    i.remove('0')
                    i.append(now1)
```

The screenshot shows the Spyder Python IDE interface. The left pane displays the Python script `Merge Capture with Get Attendance.py`. The right pane shows the file explorer with two directories: `Attendance Directory` and `Image Directory`, both modified on 01-10-2021 at 18:31. Below the file explorer is the IPython console, which contains the command `In [2]: runcell(0, 'C:/Users/Lenovo/.spyder-py3/LLFRAS/Merge Capture with Get Attendance.py')`. The status bar at the bottom indicates the system is at 18°C, sunny, and the date is 22-10-2021.

```
def subject_directories():
    global subject_window
    subject_window = Toplevel()
    subject_window.title(path_6)
    subject_window.iconbitmap('C:/Users/Lenovo/Downloads/images.ico')

    now = datetime.datetime.now(pytz.timezone('Asia/Kolkata'))
    global now1, now2
    now1 = now.strftime('%I:%M')
    now2 = now.strftime('%Y-%m-%d')

    os.chdir(path_6)
    def show_forward():
        Button(subject_window, text='>>', command= lambda: [subject_window.deiconify(), semester_window.deiconify()])
        Button(section_window, text='>>', command= lambda: [semester_window.deiconify(), section_window.deiconify()])
        Button(batch_window, text='>>', command= lambda: [section_window.deiconify(), batch_window.withdraw()])
        Button(branch_window, text='>>', command= lambda: [batch_window.deiconify(), branch_window.withdraw()])
        Button(root, text='>>', command= lambda: [branch_window.deiconify(), root.withdraw()], padx = 10)
```



# Chandigarh Engineering College Jhanjeri

## Mohali - 140307

### Department of Computer Science & Engineering

Spyder (Python 3.8)

File Edit Search Source Run Debug Consoles Projects Tools View Help

C:\Users\Lenovo\spyder-py3\LLFRAS\Merge Capture with Get Attendance.py

```
temp.py Live Logging Attendance 3.py Merge Capture with Get Attendance.py
```

```
1240 def branch_directories():
1241     global branch_window
1242     branch_window = Toplevel()
1243
1244     branch = ['Computer science', 'Electrical', 'Mechanical', 'Civil', 'Agriculture']
1245     if path_8 == None:
1246         global branch_list, path_9
1247         path_9 = None
1248         os.chdir(path_1)
1249         if 'branch' not in os.listdir(path_1):
1250             global path_2
1251             path_2 = path_1 + '/' + 'branch'
1252             os.mkdir(path_2)
1253             os.chdir(path_2)
1254             for b in branch:
1255                 os.mkdir(path_2 + '/' + b)
1256             elif 'branch' in os.listdir(path_1):
1257                 path_2 = path_1 + '/' + 'branch'
1258                 os.chdir(path_2)
1259                 branch_window.title(path_2)
1260                 branch_list = os.listdir(path_2)
1261
1262     elif path_1 == None:
1263         path_2 = None
1264         os.chdir(path_8)
1265         path_9 = path_8 + '/' + 'branch'
1266         os.chdir(path_9)
1267         branch_window.title(path_9)
1268         branch_list = os.listdir(path_9)
1269         branch_window.iconbitmap('C:/Users/Lenovo/Downloads/images.ico')
1270
1271     def show_forward():
1272         Button(root, text='>>', command= lambda: [branch_window.deiconify(), root.withdraw()], padx = 10)
1273
1274     Button(branch_window, text='<<', command= lambda: [branch_window.withdraw(), root.deiconify(), show_forward()])
1275     Button(branch_window, text = 'Start', command = lambda:[branch_window.withdraw(), root.deiconify()])
1276     Button(branch_window, text='Home', state = DISABLED, padx = 20).grid(row=0, column=2)
1277     Button(branch_window, text='>>', state = DISABLED, padx=10).grid(row=0, column=3)
1278
1279     new = Button(branch_window, text='New', command=new_branch).grid(row=0, column=5)
```

Name Date Modified

> Attendance Directory 01-10-2021 18:31

> Image Directory 01-10-2021 18:45

Console I/A X

In [2]: runcell(0, 'C:/Users/Lenovo/.spyder-py3/LLFRAS/Merge Capture with Get Attendance.py')

In [3]:

LSP Python: ready conda: env\_dlib (Python 3.8.11) Line 19, Col 1 UTF-8 CRLF RW Mem 60%

18°C Sunny 07:25 22-10-2021

Spyder (Python 3.8)

File Edit Search Source Run Debug Consoles Projects Tools View Help

C:\Users\Lenovo\spyder-py3\LLFRAS\Merge Capture with Get Attendance.py

```
temp.py Live Logging Attendance 3.py Merge Capture with Get Attendance.py
```

```
1021 def batch_directories():
1022     global batch_window
1023     batch_window = Toplevel()
1024
1025     if path_10 == None:
1026         global batch_list
1027         batch_window.title(path_3)
1028         os.chdir(path_3)
1029         batch_list = os.listdir(path_3)
1030     elif path_3 == None:
1031         batch_window.title(path_10)
1032         os.chdir(path_10)
1033         batch_list = os.listdir(path_10)
1034
1035     batch_window.iconbitmap('C:/Users/Lenovo/Downloads/images.ico')
1036
1037     def show_forward():
1038         Button(batch_window, text='>>', command= lambda: [batch_window.deiconify(), branch_window.withdraw()])
1039         Button(root, text='<<', command= lambda: [batch_window.withdraw(), branch_window.deiconify(), root.withdraw()], padx = 10)
1040
1041     Button(batch_window, text='<<', command= lambda: [batch_window.withdraw(), branch_window.deiconify()])
1042     Button(batch_window, text = 'Start', command = lambda:[batch_window.withdraw(), root.deiconify()])
1043     Button(batch_window, text='Home', command = lambda: [batch_window.withdraw(), branch_window.deiconify()])
1044     Button(batch_window, text='>>', state = DISABLED, padx=10).grid(row=0, column=3)
1045
1046     Button(batch_window, text='New', command=new_batch).grid(row=0, column=5)
1047     Button(batch_window, text='Remove', command=remove_batch).grid(row=0, column=6)
1048     Label(batch_window, text='').grid(row=1, column=4)
1049
1050     Label(batch_window, text='Sl No').grid(row=2, column=0)
1051     Label(batch_window, text='Batch Name').grid(row=2, column=1)
1052     Label(batch_window, text='').grid(row=3, column=0)
1053
1054     i = 3
1055     j = 1
1056     for batch in batch_list:
1057         Label(batch_window, text=j).grid(row=i+1, column=0)
1058         Label(batch_window, text=batch).grid(row=i+1, column=1)
1059         i = i+1
1060         j = j+1
```

Name Date Modified

> Attendance Directory 01-10-2021 18:31

> Image Directory 01-10-2021 18:45

Console I/A X

In [2]: runcell(0, 'C:/Users/Lenovo/.spyder-py3/LLFRAS/Merge Capture with Get Attendance.py')

In [3]:

LSP Python: ready conda: env\_dlib (Python 3.8.11) Line 19, Col 1 UTF-8 CRLF RW Mem 60%

18°C Sunny 07:25 22-10-2021



# Chandigarh Engineering College Jhanjeri

## Mohali - 140307

### Department of Computer Science & Engineering

This screenshot shows the Spyder Python IDE interface. The main window displays a Python script named 'Merge Capture with Get Attendance.py'. The code implements a graphical user interface (GUI) using Tkinter for managing windows, buttons, and labels. It also includes logic for capturing video frames using OpenCV's VideoCapture module and processing them with the face\_recognition library to detect faces and compare them against a known list of students.

```
530
531 def semester_directories():
532     global semester_window
533     semester_window = Toplevel()
534     semester_window.title('path_5')
535     semester_window.iconbitmap('C:/Users/Lenovo/Downloads/images.ico')
536
537     os.chdir(path_5)
538     def show_forward():
539         button(section_window, text='>', command= lambda: [semester_window.deiconify(), section_window.withdraw()])
540         button(batch_window, text='>', command= lambda: [section_window.deiconify(), batch_window.withdraw()])
541         button(branch_window, text='>', command= lambda: [batch_window.deiconify(), branch_window.withdraw()])
542         button(root, text='>', command= lambda: [branch_window.deiconify(), root.withdraw()], padx= 18)
543
544     Button(semester_window, text='<', command= lambda: [semester_window.withdraw(), section_window.deiconify()])
545     Button(semester_window, text='Start', command= lambda: [semester_window.withdraw(), section_window.deiconify()])
546     Button(semester_window, text='Home', command= lambda: [semester_window.withdraw(), section_window.deiconify()])
547     Button(semester_window, text='>', state = DISABLED, padx=10).grid(row=0, column=3)
548
549     Button(semester_window, text='New', command=new_semester).grid(row=0, column=5)
550     Button(semester_window, text='Remove', command=remove_semester).grid(row=0, column=6)
551     Label(semester_window, text='').grid(row=1, column= 4)
552
553     encoding_section()
554     Label(semester_window, text='STUDENTS').grid(row=2, column=1)
555     Label(semester_window, text=' IN ').grid(row=2, column=2)
556     Label(semester_window, text='SECTION (input_3)').grid(row=2, column=3)
557     Label(semester_window, text='').grid(row=3)
558
559     Label(semester_window, text='SL No').grid(row=4, column=0)
560     Label(semester_window, text='First Name').grid(row=4, column=1)
561     Label(semester_window, text='Last Name').grid(row=4, column=2)
562     Label(semester_window, text='Roll No').grid(row=4, column=3)
563
564     k = 5
565     l = 1
566     for name, roll in zip(classNames, RollNo):
567         spli = name.split(' ')
568         first_name = spli[0].capitalize()
569         last_name = spli[1].capitalize()
570
571         k = k + 1
572         l = l + 1
573
574         if k == 5:
575             l = 1
576
577         if l == 5:
578             k = 5
579             l = 1
580
581         Label(semester_window, text=name).grid(row=k, column=l)
582         Entry(semester_window, textvariable=roll).grid(row=k, column=l+1)
583
584         l = l + 2
585
586     k = 5
587     l = 1
588
589     for name, roll in zip(classNames, RollNo):
590         spli = name.split(' ')
591         first_name = spli[0].capitalize()
592         last_name = spli[1].capitalize()
593
594         k = k + 1
595         l = l + 1
596
597         if k == 5:
598             l = 1
599
600         if l == 5:
601             k = 5
602             l = 1
603
604         Label(semester_window, text=name).grid(row=k, column=l)
605         Entry(semester_window, textvariable=roll).grid(row=k, column=l+1)
606
607         l = l + 2
608
609     k = 5
610     l = 1
611
612     for name, roll in zip(classNames, RollNo):
613         spli = name.split(' ')
614         first_name = spli[0].capitalize()
615         last_name = spli[1].capitalize()
616
617         k = k + 1
618         l = l + 1
619
620         if k == 5:
621             l = 1
622
623         if l == 5:
624             k = 5
625             l = 1
626
627         Label(semester_window, text=name).grid(row=k, column=l)
628         Entry(semester_window, textvariable=roll).grid(row=k, column=l+1)
629
630         l = l + 2
631
632     k = 5
633     l = 1
634
635     for name, roll in zip(classNames, RollNo):
636         spli = name.split(' ')
637         first_name = spli[0].capitalize()
638         last_name = spli[1].capitalize()
639
640         k = k + 1
641         l = l + 1
642
643         if k == 5:
644             l = 1
645
646         if l == 5:
647             k = 5
648             l = 1
649
650         Label(semester_window, text=name).grid(row=k, column=l)
651         Entry(semester_window, textvariable=roll).grid(row=k, column=l+1)
652
653         l = l + 2
654
655     k = 5
656     l = 1
657
658     for name, roll in zip(classNames, RollNo):
659         spli = name.split(' ')
660         first_name = spli[0].capitalize()
661         last_name = spli[1].capitalize()
662
663         k = k + 1
664         l = l + 1
665
666         if k == 5:
667             l = 1
668
669         if l == 5:
670             k = 5
671             l = 1
672
673         Label(semester_window, text=name).grid(row=k, column=l)
674         Entry(semester_window, textvariable=roll).grid(row=k, column=l+1)
675
676         l = l + 2
677
678     k = 5
679     l = 1
680
681     for name, roll in zip(classNames, RollNo):
682         spli = name.split(' ')
683         first_name = spli[0].capitalize()
684         last_name = spli[1].capitalize()
685
686         k = k + 1
687         l = l + 1
688
689         if k == 5:
690             l = 1
691
692         if l == 5:
693             k = 5
694             l = 1
695
696         Label(semester_window, text=name).grid(row=k, column=l)
697         Entry(semester_window, textvariable=roll).grid(row=k, column=l+1)
698
699         l = l + 2
700
701     k = 5
702     l = 1
703
704     for name, roll in zip(classNames, RollNo):
705         spli = name.split(' ')
706         first_name = spli[0].capitalize()
707         last_name = spli[1].capitalize()
708
709         k = k + 1
710         l = l + 1
711
712         if k == 5:
713             l = 1
714
715         if l == 5:
716             k = 5
717             l = 1
718
719         Label(semester_window, text=name).grid(row=k, column=l)
720         Entry(semester_window, textvariable=roll).grid(row=k, column=l+1)
721
722         l = l + 2
723
724     k = 5
725     l = 1
726
727     for name, roll in zip(classNames, RollNo):
728         spli = name.split(' ')
729         first_name = spli[0].capitalize()
730         last_name = spli[1].capitalize()
731
732         k = k + 1
733         l = l + 1
734
735         if k == 5:
736             l = 1
737
738         if l == 5:
739             k = 5
740             l = 1
741
742         Label(semester_window, text=name).grid(row=k, column=l)
743         Entry(semester_window, textvariable=roll).grid(row=k, column=l+1)
744
745         l = l + 2
746
747     k = 5
748     l = 1
749
750     for name, roll in zip(classNames, RollNo):
751         spli = name.split(' ')
752         first_name = spli[0].capitalize()
753         last_name = spli[1].capitalize()
754
755         k = k + 1
756         l = l + 1
757
758         if k == 5:
759             l = 1
760
761         if l == 5:
762             k = 5
763             l = 1
764
765         Label(semester_window, text=name).grid(row=k, column=l)
766         Entry(semester_window, textvariable=roll).grid(row=k, column=l+1)
767
768         l = l + 2
769
770     k = 5
771     l = 1
772
773     for name, roll in zip(classNames, RollNo):
774         spli = name.split(' ')
775         first_name = spli[0].capitalize()
776         last_name = spli[1].capitalize()
777
778         k = k + 1
779         l = l + 1
780
781         if k == 5:
782             l = 1
783
784         if l == 5:
785             k = 5
786             l = 1
787
788         Label(semester_window, text=name).grid(row=k, column=l)
789         Entry(semester_window, textvariable=roll).grid(row=k, column=l+1)
790
791         l = l + 2
792
793     k = 5
794     l = 1
795
796     for name, roll in zip(classNames, RollNo):
797         spli = name.split(' ')
798         first_name = spli[0].capitalize()
799         last_name = spli[1].capitalize()
800
801         k = k + 1
802         l = l + 1
803
804         if k == 5:
805             l = 1
806
807         if l == 5:
808             k = 5
809             l = 1
810
811         Label(semester_window, text=name).grid(row=k, column=l)
812         Entry(semester_window, textvariable=roll).grid(row=k, column=l+1)
813
814         l = l + 2
815
816     k = 5
817     l = 1
818
819     for name, roll in zip(classNames, RollNo):
820         spli = name.split(' ')
821         first_name = spli[0].capitalize()
822         last_name = spli[1].capitalize()
823
824         k = k + 1
825         l = l + 1
826
827         if k == 5:
828             l = 1
829
830         if l == 5:
831             k = 5
832             l = 1
833
834         Label(semester_window, text=name).grid(row=k, column=l)
835         Entry(semester_window, textvariable=roll).grid(row=k, column=l+1)
836
837         l = l + 2
838
839     k = 5
840     l = 1
841
842     for name, roll in zip(classNames, RollNo):
843         spli = name.split(' ')
844         first_name = spli[0].capitalize()
845         last_name = spli[1].capitalize()
846
847         k = k + 1
848         l = l + 1
849
850         if k == 5:
851             l = 1
852
853         if l == 5:
854             k = 5
855             l = 1
856
857         Label(semester_window, text=name).grid(row=k, column=l)
858         Entry(semester_window, textvariable=roll).grid(row=k, column=l+1)
859
860         l = l + 2
861
862     k = 5
863     l = 1
864
865     for name, roll in zip(classNames, RollNo):
866         spli = name.split(' ')
867         first_name = spli[0].capitalize()
868         last_name = spli[1].capitalize()
869
870         k = k + 1
871         l = l + 1
872
873         if k == 5:
874             l = 1
875
876         if l == 5:
877             k = 5
878             l = 1
879
880         Label(semester_window, text=name).grid(row=k, column=l)
881         Entry(semester_window, textvariable=roll).grid(row=k, column=l+1)
882
883         l = l + 2
884
885     k = 5
886     l = 1
887
888     for name, roll in zip(classNames, RollNo):
889         spli = name.split(' ')
890         first_name = spli[0].capitalize()
891         last_name = spli[1].capitalize()
892
893         k = k + 1
894         l = l + 1
895
896         if k == 5:
897             l = 1
898
899         if l == 5:
900             k = 5
901             l = 1
902
903         Label(semester_window, text=name).grid(row=k, column=l)
904         Entry(semester_window, textvariable=roll).grid(row=k, column=l+1)
905
906         l = l + 2
907
908     k = 5
909     l = 1
910
911     for name, roll in zip(classNames, RollNo):
912         spli = name.split(' ')
913         first_name = spli[0].capitalize()
914         last_name = spli[1].capitalize()
915
916         k = k + 1
917         l = l + 1
918
919         if k == 5:
920             l = 1
921
922         if l == 5:
923             k = 5
924             l = 1
925
926         Label(semester_window, text=name).grid(row=k, column=l)
927         Entry(semester_window, textvariable=roll).grid(row=k, column=l+1)
928
929         l = l + 2
930
931     k = 5
932     l = 1
933
934     for name, roll in zip(classNames, RollNo):
935         spli = name.split(' ')
936         first_name = spli[0].capitalize()
937         last_name = spli[1].capitalize()
938
939         k = k + 1
940         l = l + 1
941
942         if k == 5:
943             l = 1
944
945         if l == 5:
946             k = 5
947             l = 1
948
949         Label(semester_window, text=name).grid(row=k, column=l)
950         Entry(semester_window, textvariable=roll).grid(row=k, column=l+1)
951
952         l = l + 2
953
954     k = 5
955     l = 1
956
957     for name, roll in zip(classNames, RollNo):
958         spli = name.split(' ')
959         first_name = spli[0].capitalize()
960         last_name = spli[1].capitalize()
961
962         k = k + 1
963         l = l + 1
964
965         if k == 5:
966             l = 1
967
968         if l == 5:
969             k = 5
970             l = 1
971
972         Label(semester_window, text=name).grid(row=k, column=l)
973         Entry(semester_window, textvariable=roll).grid(row=k, column=l+1)
974
975         l = l + 2
976
977     k = 5
978     l = 1
979
980     for name, roll in zip(classNames, RollNo):
981         spli = name.split(' ')
982         first_name = spli[0].capitalize()
983         last_name = spli[1].capitalize()
984
985         k = k + 1
986         l = l + 1
987
988         if k == 5:
989             l = 1
990
991         if l == 5:
992             k = 5
993             l = 1
994
995         Label(semester_window, text=name).grid(row=k, column=l)
996         Entry(semester_window, textvariable=roll).grid(row=k, column=l+1)
997
998         l = l + 2
999
1000     k = 5
1001     l = 1
1002
1003     for name, roll in zip(classNames, RollNo):
1004         spli = name.split(' ')
1005         first_name = spli[0].capitalize()
1006         last_name = spli[1].capitalize()
1007
1008         k = k + 1
1009         l = l + 1
1010
1011         if k == 5:
1012             l = 1
1013
1014         if l == 5:
1015             k = 5
1016             l = 1
1017
1018         Label(semester_window, text=name).grid(row=k, column=l)
1019         Entry(semester_window, textvariable=roll).grid(row=k, column=l+1)
1020
1021         l = l + 2
1022
1023     k = 5
1024     l = 1
1025
1026     for name, roll in zip(classNames, RollNo):
1027         spli = name.split(' ')
1028         first_name = spli[0].capitalize()
1029         last_name = spli[1].capitalize()
1030
1031         k = k + 1
1032         l = l + 1
1033
1034         if k == 5:
1035             l = 1
1036
1037         if l == 5:
1038             k = 5
1039             l = 1
1040
1041         Label(semester_window, text=name).grid(row=k, column=l)
1042         Entry(semester_window, textvariable=roll).grid(row=k, column=l+1)
1043
1044         l = l + 2
1045
1046     k = 5
1047     l = 1
1048
1049     for name, roll in zip(classNames, RollNo):
1050         spli = name.split(' ')
1051         first_name = spli[0].capitalize()
1052         last_name = spli[1].capitalize()
1053
1054         k = k + 1
1055         l = l + 1
1056
1057         if k == 5:
1058             l = 1
1059
1060         if l == 5:
1061             k = 5
1062             l = 1
1063
1064         Label(semester_window, text=name).grid(row=k, column=l)
1065         Entry(semester_window, textvariable=roll).grid(row=k, column=l+1)
1066
1067         l = l + 2
1068
1069     k = 5
1070     l = 1
1071
1072     for name, roll in zip(classNames, RollNo):
1073         spli = name.split(' ')
1074         first_name = spli[0].capitalize()
1075         last_name = spli[1].capitalize()
1076
1077         k = k + 1
1078         l = l + 1
1079
1080         if k == 5:
1081             l = 1
1082
1083         if l == 5:
1084             k = 5
1085             l = 1
1086
1087         Label(semester_window, text=name).grid(row=k, column=l)
1088         Entry(semester_window, textvariable=roll).grid(row=k, column=l+1)
1089
1090         l = l + 2
1091
1092     k = 5
1093     l = 1
1094
1095     for name, roll in zip(classNames, RollNo):
1096         spli = name.split(' ')
1097         first_name = spli[0].capitalize()
1098         last_name = spli[1].capitalize()
1099
1100         k = k + 1
1101         l = l + 1
1102
1103         if k == 5:
1104             l = 1
1105
1106         if l == 5:
1107             k = 5
1108             l = 1
1109
1110         Label(semester_window, text=name).grid(row=k, column=l)
1111         Entry(semester_window, textvariable=roll).grid(row=k, column=l+1)
1112
1113         l = l + 2
1114
1115     k = 5
1116     l = 1
1117
1118     for name, roll in zip(classNames, RollNo):
1119         spli = name.split(' ')
1120         first_name = spli[0].capitalize()
1121         last_name = spli[1].capitalize()
1122
1123         k = k + 1
1124         l = l + 1
1125
1126         if k == 5:
1127             l = 1
1128
1129         if l == 5:
1130             k = 5
1131             l = 1
1132
1133         Label(semester_window, text=name).grid(row=k, column=l)
1134         Entry(semester_window, textvariable=roll).grid(row=k, column=l+1)
1135
1136         l = l + 2
1137
1138     k = 5
1139     l = 1
1140
1141     for name, roll in zip(classNames, RollNo):
1142         spli = name.split(' ')
1143         first_name = spli[0].capitalize()
1144         last_name = spli[1].capitalize()
1145
1146         k = k + 1
1147         l = l + 1
1148
1149         if k == 5:
1150             l = 1
1151
1152         if l == 5:
1153             k = 5
1154             l = 1
1155
1156         Label(semester_window, text=name).grid(row=k, column=l)
1157         Entry(semester_window, textvariable=roll).grid(row=k, column=l+1)
1158
1159         l = l + 2
1160
1161     k = 5
1162     l = 1
1163
1164     for name, roll in zip(classNames, RollNo):
1165         spli = name.split(' ')
1166         first_name = spli[0].capitalize()
1167         last_name = spli[1].capitalize()
1168
1169         k = k + 1
1170         l = l + 1
1171
1172         if k == 5:
1173             l = 1
1174
1175         if l == 5:
1176             k = 5
1177             l = 1
1178
1179         Label(semester_window, text=name).grid(row=k, column=l)
1180         Entry(semester_window, textvariable=roll).grid(row=k, column=l+1)
1181
1182         l = l + 2
1183
1184     k = 5
1185     l = 1
1186
1187     for name, roll in zip(classNames, RollNo):
1188         spli = name.split(' ')
1189         first_name = spli[0].capitalize()
1190         last_name = spli[1].capitalize()
1191
1192         k = k + 1
1193         l = l + 1
1194
1195         if k == 5:
1196             l = 1
1197
1198         if l == 5:
1199             k = 5
1200             l = 1
1201
1202         Label(semester_window, text=name).grid(row=k, column=l)
1203         Entry(semester_window, textvariable=roll).grid(row=k, column=l+1)
1204
1205         l = l + 2
1206
1207     k = 5
1208     l = 1
1209
1210     for name, roll in zip(classNames, RollNo):
1211         spli = name.split(' ')
1212         first_name = spli[0].capitalize()
1213         last_name = spli[1].capitalize()
1214
1215         k = k + 1
1216         l = l + 1
1217
1218         if k == 5:
1219             l = 1
1220
1221         if l == 5:
1222             k = 5
1223             l = 1
1224
1225         Label(semester_window, text=name).grid(row=k, column=l)
1226         Entry(semester_window, textvariable=roll).grid(row=k, column=l+1)
1227
1228         l = l + 2
1229
1230     k = 5
1231     l = 1
1232
1233     for name, roll in zip(classNames, RollNo):
1234         spli = name.split(' ')
1235         first_name = spli[0].capitalize()
1236         last_name = spli[1].capitalize()
1237
1238         k = k + 1
1239         l = l + 1
1240
1241         if k == 5:
1242             l = 1
1243
1244         if l == 5:
1245             k = 5
1246             l = 1
1247
1248         Label(semester_window, text=name).grid(row=k, column=l)
1249         Entry(semester_window, textvariable=roll).grid(row=k, column=l+1)
1250
1251         l = l + 2
1252
1253     k = 5
1254     l = 1
1255
1256     for name, roll in zip(classNames, RollNo):
1257         spli = name.split(' ')
1258         first_name = spli[0].capitalize()
1259         last_name = spli[1].capitalize()
1260
1261         k = k + 1
1262         l = l + 1
1263
1264         if k == 5:
1265             l = 1
1266
1267         if l == 5:
1268             k = 5
1269             l = 1
1270
1271         Label(semester_window, text=name).grid(row=k, column=l)
1272         Entry(semester_window, textvariable=roll).grid(row=k, column=l+1)
1273
1274         l = l + 2
1275
1276     k = 5
1277     l = 1
1278
1279     for name, roll in zip(classNames, RollNo):
1280         spli = name.split(' ')
1281         first_name = spli[0].capitalize()
1282         last_name = spli[1].capitalize()
1283
1284         k = k + 1
1285         l = l + 1
1286
1287         if k == 5:
1288             l = 1
1289
1290         if l == 5:
1291             k = 5
1292             l = 1
1293
1294         Label(semester_window, text=name).grid(row=k, column=l)
1295         Entry(semester_window, textvariable=roll).grid(row=k, column=l+1)
1296
1297         l = l + 2
1298
1299     k = 5
1300     l = 1
1301
1302     for name, roll in zip(classNames, RollNo):
1303         spli = name.split(' ')
1304         first_name = spli[0].capitalize()
1305         last_name = spli[1].capitalize()
1306
1307         k = k + 1
1308         l = l + 1
1309
1310         if k == 5:
1311             l = 1
1312
1313         if l == 5:
1314             k = 5
1315             l = 1
1316
1317         Label(semester_window, text=name).grid(row=k, column=l)
1318         Entry(semester_window, textvariable=roll).grid(row=k, column=l+1)
1319
1320         l = l + 2
1321
1322     k = 5
1323     l = 1
1324
1325     for name, roll in zip(classNames, RollNo):
1326         spli = name.split(' ')
1327         first_name = spli[0].capitalize()
1328         last_name = spli[1].capitalize()
1329
1330         k = k + 1
1331         l = l + 1
1332
1333         if k == 5:
1334             l = 1
1335
1336         if l == 5:
1337             k = 5
1338             l = 1
1339
1340         Label(semester_window, text=name).grid(row=k, column=l)
1341         Entry(semester_window, textvariable=roll).grid(row=k, column=l+1)
1342
1343         l = l + 2
1344
1345     k = 5
1346     l = 1
1347
1348     for name, roll in zip(classNames, RollNo):
1349         spli = name.split(' ')
1350         first_name = spli[0].capitalize()
1351         last_name = spli[1].capitalize()
1352
1353         k = k + 1
1354         l = l + 1
1355
1356         if k == 5:
1357             l = 1
1358
1359         if l == 5:
1360             k = 5
1361             l = 1
1362
1363         Label(semester_window, text=name).grid(row=k, column=l)
1364         Entry(semester_window, textvariable=roll).grid(row=k, column=l+1)
1365
1366         l = l + 2
1367
1368     k = 5
1369     l = 1
1370
1371     for name, roll in zip(classNames, RollNo):
1372         spli = name.split(' ')
1373         first_name = spli[0].capitalize()
1374         last_name = spli[1].capitalize()
1375
1376         k = k + 1
1377         l = l + 1
1378
1379         if k == 5:
1380             l = 1
1381
1382         if l == 5:
1383             k = 5
1384             l = 1
1385
1386         Label(semester_window, text=name).grid(row=k, column=l)
1387         Entry(semester_window, textvariable=roll).grid(row=k, column=l+1)
1388
1389         l = l + 2
1390
1391     k = 5
1392     l = 1
1393
1394     for name, roll in zip(classNames, RollNo):
1395         spli = name.split(' ')
1396         first_name = spli[0].capitalize()
1397         last_name = spli[1].capitalize()
1398
1399         k = k + 1
1400         l = l + 1
1401
1402         if k == 5:
1403             l = 1
1404
1405         if l == 5:
1406             k = 5
1407             l = 1
1408
1409         Label(semester_window, text=name).grid(row=k, column=l)
1410         Entry(semester_window, textvariable=roll).grid(row=k, column=l+1)
1411
1412         l = l + 2
1413
1414     k = 5
1415     l = 1
1416
1417     for name, roll in zip(classNames, RollNo):
1418         spli = name.split(' ')
1419         first_name = spli[0].capitalize()
1420         last_name = spli[1].capitalize()
1421
1422         k = k + 1
1423         l = l + 1
1424
1425         if k == 5:
1426             l = 1
1427
1428         if l == 5:
1429             k = 5
1430             l = 1
1431
1432         Label(semester_window, text=name).grid(row=k, column=l)
1433         Entry(semester_window, textvariable=roll).grid(row=k, column=l+1)
1434
1435         l = l + 2
1436
1437     k = 5
1438     l = 1
1439
1440     for name, roll in zip(classNames, RollNo):
1441         spli = name.split(' ')
1442         first_name = spli[0].capitalize()
1443         last_name = spli[1].capitalize()
1444
1445         k = k + 1
1446         l = l + 1
1447
1448         if k == 5:
1449             l = 1
1450
1451         if l == 5:
1452             k = 5
1453             l = 1
1454
1455         Label(semester_window, text=name).grid(row=k, column=l)
1456         Entry(semester_window, textvariable=roll).grid(row=k, column=l+1)
1457
1458         l = l + 2
1459
1460     k = 5
1461     l = 1
1462
1463     for name, roll in zip(classNames, RollNo):
1464         spli = name.split(' ')
1465         first_name = spli[0].capitalize()
1466         last_name = spli[1].capitalize()
1467
1468         k = k + 1
1469         l = l + 1
1470
1471         if k == 5:
1472             l = 1
1473
1474         if l == 5:
1475             k = 5
1476             l = 1
1477
1478         Label(semester_window, text=name).grid(row=k, column=l)
1479         Entry(semester_window, textvariable=roll).grid(row=k, column=l+1)
1480
1481         l = l + 2
1482
1483     k = 5
1484     l = 1
1485
1486     for name, roll in zip(classNames, RollNo):
1487         spli = name.split(' ')
1488         first_name = spli[0].capitalize()
1489         last_name = spli[1].capitalize()
1490
1491         k = k + 1
1492         l = l + 1
1493
1494         if k == 5:
1495             l = 1
1496
1497         if l == 5:
1498             k = 5
1499             l = 1
1500
1501         Label(semester_window, text=name).grid(row=k, column=l)
1502         Entry(semester_window, textvariable=roll).grid(row=k, column=l+1)
1503
1504         l = l + 2
1505
1506     k = 5
1507     l = 1
1508
1509     for name, roll in zip(classNames, RollNo):
1510         spli = name.split(' ')
1511         first_name = spli[0].capitalize()
1512         last_name = spli[1].capitalize()
1513
1514         k = k + 1
1515         l = l + 1
1516
1517         if k == 5:
1518             l = 1
1519
1520         if l == 5:
1521             k = 5
1522             l = 1
1523
1524         Label(semester_window, text=name).grid(row=k, column=l)
1525         Entry(semester_window, textvariable=roll).grid(row=k, column=l+1)
1526
1527         l = l + 2
1528
1529     k = 5
1530     l = 1
1531
1532     for name, roll in zip(classNames, RollNo):
1533         spli = name.split(' ')
1534         first_name = spli[0].capitalize()
1535         last_name = spli[1].capitalize()
1536
1537         k = k + 1
1538         l = l + 1
1539
1540         if k == 5:
1541             l = 1
1542
1543         if l == 5:
1544             k = 5
1545             l = 1
1546
1547         Label(semester_window, text=name).grid(row=k, column=l)
1548         Entry(semester_window, textvariable=roll).grid(row=k, column=l+1)
1549
1550         l = l + 2
1551
1552     k = 5
1553     l = 1
1554
1555     for name, roll in zip(classNames, RollNo):
1556         spli = name.split(' ')
1557         first_name = spli[0].capitalize()
1558         last_name = spli[1].capitalize()
1559
1560         k = k + 1
1561         l = l + 1
1562
1563         if k == 5:
1564             l = 1
1565
1566         if l == 5:
1567             k = 5
1568             l = 1
1569
1
```



# Chandigarh Engineering College Jhanjeri

## Mohali - 140307

### Department of Computer Science & Engineering

Spyder (Python 3.8)

C:\Users\Lenovo.spyder-py3\LLFRAS\Merge Capture with Get Attendance.py

```
temp.py Live Logging Attendance 3.py Merge Capture with Get Attendance.py
```

```
293
294
295 def new_subject():
296     global new_subject_window
297     new_subject_window = Toplevel()
298     new_subject_window.geometry('370x175')
299     new_subject_window.title('Live Logging Face Recognition Attendance System')
300     new_subject_window.iconbitmap('C:/Users/Lenovo/Downloads/images.ico')
301
302     Label(new_subject_window, text='Enter New Subject Name: ').grid(row=0, column=0)
303     e = Entry(new_subject_window, width=20,
304               fg='blue', bg='white', borderwidth=10)
305     e.grid(row=0, column=1)
306
307     def save():
308         if e.get() == '':
309             #new_subject_window.destroy()
310             new_subject()
311             Label(new_subject_window, text='Enter New Subject Name: ').grid(row=1, column=0)
312             e1 = Entry(new_subject_window, width=20,
313                         fg=e.get().capitalize(), bg='white', borderwidth=10)
314             e1.grid(row=1, column=1)
315             os.chdir(path_6 + '/' + e.get())
316             if e.get().capitalize()+'csv' not in os.listdir(path_6):
317                 getinput = e.get().capitalize()
318                 createSubjectCSVFile(getinput, path_6)
319                 makeNewHeader(getinput, path_6)
320                 subject_window.destroy()
321                 subject_directories()
322
323     i = 2
324     for i in range(6):
325         Label(new_subject_window, text='').grid(row=i)
326         i = i+1
327
328     Button(new_subject_window, text='Save', command=lambda: [save(), new_subject_window.destroy()]).grid(
329     row=6, column=0)
330     Button(new_subject_window, text='Exit', command=new_subject_window.destroy, padx=10).grid(row=6, column=1)
331
332 def remove_subject():
333     global remove_subject_window
```

File Edit Search Source Run Debug Consoles Projects Tools View Help

C:\Users\Lenovo\Desktop\Attendance

Name Date Modified

> Attendance Directory 01-10-2021 18:31

> Image Directory 01-10-2021 18:45

Help Variable Explorer Plots Files

Console I/A X

In [2]: runcell(0, 'C:/Users/Lenovo/.spyder-py3/LLFRAS/Merge Capture with Get Attendance.py')

In [3]:

LSP Python: ready conda: env\_dlib (Python 3.8.11) Line 19, Col 1 UTF-8 CRLF RW Mem 61%

18°C Sunny 07:24 22-10-2021

Spyder (Python 3.8)

C:\Users\Lenovo.spyder-py3\LLFRAS\Merge Capture with Get Attendance.py

```
temp.py Live Logging Attendance 3.py Merge Capture with Get Attendance.py
```

```
369
370 def subject_directories():
371     global subject_window
372     subject_window = Toplevel()
373     subject_window.title(path_6)
374     subject_window.iconbitmap('C:/Users/Lenovo/Downloads/images.ico')
375
376     now = datetime.datetime.now(pytz.timezone('Asia/Kolkata'))
377     global now1, now2
378     now1 = now.strftime('%I:%M')
379     now2 = now.strftime('%Y-%m-%d')
380
381     os.chdir(path_6)
382     def show_forward():
383         Button(subject_window, text='>>', command=lambda: [subject_window.deiconify(), semester_window.withdraw()])
384         Button(subject_window, text='>', command=lambda: [semester_window.deiconify(), section_window.withdraw()])
385         Button(batch_window, text='>>', command=lambda: [section.window.deiconify(), batch_window.withdraw()])
386         Button(branch_window, text='>>', command=lambda: [batch.window.deiconify(), branch_window.withdraw()])
387         Button(root, text='>>', command=lambda: [branch.window.deiconify(), root.withdraw()], padx = 10)
388
389     Button(subject_window, text='>>', command=lambda: [subject_window.withdraw(), semester_window.deiconify()])
390     Button(subject_window, text='Start', command=lambda: [subject_window.withdraw(), semester.window])
391     Button(subject_window, text='Home', command=lambda: [subject_window.withdraw(), semester_window.withdraw()])
392     Button(subject_window, text='>>>', state = DISABLED, padx=10).grid(row=0, column=3)
393
394     Button(subject_window, text='New', command=new_subject).grid(row=0, column=5)
395     Button(subject_window, text='Remove', command=remove_subject).grid(row=0, column=6)
396     Label(subject_window, text='').grid(row=1, column = 4)
397
398     Label(subject_window, text='Sl No').grid(row=2, column=0)
399     Label(subject_window, text='Subject Name').grid(row=2, column=1)
400     Label(subject_window, text='').grid(row=3, column = 0)
401
402     subject_list = os.listdir(path_6)
403     i = 3
404     j = 1
405     for subject in subject_list:
406         Label(subject_window, text=j).grid(row=i+1, column=0)
407         Label(subject_window, text=subject).grid(row=i+1, column=1)
408         i = i+1
```

File Edit Search Source Run Debug Consoles Projects Tools View Help

C:\Users\Lenovo\Desktop\Attendance

Name Date Modified

> Attendance Directory 01-10-2021 18:31

> Image Directory 01-10-2021 18:45

Help Variable Explorer Plots Files

Console I/A X

In [2]: runcell(0, 'C:/Users/Lenovo/.spyder-py3/LLFRAS/Merge Capture with Get Attendance.py')

In [3]:

LSP Python: ready conda: env\_dlib (Python 3.8.11) Line 19, Col 1 UTF-8 CRLF RW Mem 61%

18°C Sunny 07:24 22-10-2021



# Chandigarh Engineering College Jhanjeri

## Mohali - 140307

### Department of Computer Science & Engineering

Spyder (Python 3.8)

C:\Users\Lenovo.spyder-py3\LLFRAS\Merge Capture with Get Attendance.py

temp.py Live Logging Attendance 3.py Merge Capture with Get Attendance.py

```
129
1298 def get_my_attendance():
1299     global get_my_attendance_window
1300     get_my_attendance_window = Toplevel()
1301     get_my_attendance_window.title(path_17)
1302     get_my_attendance_window.iconbitmap('C:/Users/Lenovo/Downloads/images.ico')
1303
1304     def show_forward():
1305         Button(fill_student_data_window, text='>', command=lambda: [get_my_attendance_window.deiconify(), fill_student_data_window.withdraw()], fg='red').grid(row=0, column=1)
1306         Button(root, text='>', command=lambda: [fill_student_data_window.deiconify(), root.withdraw()]).grid(row=0, column=0)
1307
1308     Button(get_my_attendance_window, text='<', command=lambda: [get_my_attendance_window.withdraw(), fill_student_data_window.deiconify()], fg='red').grid(row=0, column=3)
1309     Button(get_my_attendance_window, text = 'Start', command = lambda:[get_my_attendance_window.withdraw(), fill_student_data_window.deiconify()])
1310     Button(get_my_attendance_window, text='Home', command = lambda:[get_my_attendance_window.withdraw()])
1311     Button(get_my_attendance_window, text='>', state = DISABLED, padx=10).grid(row=0, column=3)
1312     Label(get_my_attendance_window, text = '').grid(row = 1)
1313
1314     Label(get_my_attendance_window, text = 'Name').grid(row = 2, column = 1)
1315     Label(get_my_attendance_window, text = 'Roll No').grid(row = 2, column = 2)
1316
1317     with open(path_17 , 'r') as f:
1318         datalist = csv.reader(f)
1319         x = 0
1320         for data in datalist:
1321             if x == 0:
1322                 global list_of_date
1323                 list_of_date = data[2:]
1324
1325             if stu_roll.get() in data:
1326                 global student_list
1327                 student_list = data
1328             Label(get_my_attendance_window, text = data[0].capitalize()).grid(row = 3, column = 1)
1329             Label(get_my_attendance_window, text = data[1]).grid(row = 3, column = 2)
1330             x = x+1
1331
1332     my_time_list = student_list[2:]
1333     total_days = len(my_time_list)
1334     Label(get_my_attendance_window, text = '').grid(row = 4)
1335     Label(get_my_attendance_window, text = 'Date Status :').grid(row = 5, column = 1)
1336     Label(get_my_attendance_window, text = 'Time Status :').grid(row = 5, column = 2)
```

Name Date Modified

> Attendance Directory 01-10-2021 18:31

> Image Directory 01-10-2021 18:31

Help Variable Explorer Plots Files

Console I/A X

In [2]: runcell(0, 'C:/Users/Lenovo/.spyder-py3/LLFRAS/Merge Capture with Get Attendance.py')

In [3]:

LSP Python: ready conda: env\_dlib (Python 3.8.11) Line 19, Col 1 UTF-8 CRLF RW Mem 60%

18°C Sunny 07:25 22-10-2021

Spyder (Python 3.8)

C:\Users\Lenovo.spyder-py3\LLFRAS\Merge Capture with Get Attendance.py

temp.py Live Logging Attendance 3.py Merge Capture with Get Attendance.py

```
1356 def access_my_attendance():
1357     global path_17
1358     path_16 = f'/{stu_branch.get().capitalize()}/{stu_batch.get().capitalize()}/{stu_section.get().capitalize()}' + path_15 + path_16
1359
1360     try:
1361         get_my_attendance()
1362     except FileNotFoundError:
1363         get_my_attendance_window.destroy()
1364         fill_student_data()
1365         Label(fill_student_data_window, text = 'Entre Valid Data').grid(row = 5, column = 5)
1366
1367     def fill_student_data():
1368         global fill_student_data_window
1369         fill_student_data_window = Toplevel()
1370         fill_student_data_window.title('Live Logging Face Recognition Attendance System')
1371         fill_student_data_window.iconbitmap('C:/Users/Lenovo/Downloads/images.ico')
1372
1373     def show_forward():
1374         Button(root, text='>', command=lambda: [fill_student_data_window.deiconify(), root.withdraw()])
1375
1376         Button(fill_student_data_window, text='<', command=lambda: [fill_student_data_window.withdraw(), fill_student_data_window.deiconify()])
1377         Button(fill_student_data_window, text = 'Start', command = lambda:[fill_student_data_window.withdraw()])
1378         Button(fill_student_data_window, text='Home', state = DISABLED, padx = 20).grid(row=0, column=2)
1379         Button(fill_student_data_window, text='>', state = DISABLED, padx=10).grid(row=0, column=3)
1380
1381         Label(fill_student_data_window, text = '').grid(row = 1)
1382
1383         global stu_branch, stu_batch, stu_section, stu_semester, stu_subject, stu_roll
1384         Label(fill_student_data_window, text = 'Branch').grid(row = 2, column = 1)
1385         stu_branch = Entry(fill_student_data_window, width = 20)
1386         stu_branch.grid(row = 2, column = 2)
1387         stu_branch.insert(0, 'Short form not allow')
1388         Label(fill_student_data_window, text = 'Batch').grid(row = 2, column = 4)
1389         stu_batch = Entry(fill_student_data_window, width = 20)
1390         stu_batch.grid(row = 2, column = 5)
1391         stu_batch.insert(0, 'Fill passout year')
```

Name Date Modified

> Attendance Directory 01-10-2021 18:31

> Image Directory 01-10-2021 18:45

Help Variable Explorer Plots Files

Console I/A X

In [2]: runcell(0, 'C:/Users/Lenovo/.spyder-py3/LLFRAS/Merge Capture with Get Attendance.py')

In [3]:

LSP Python: ready conda: env\_dlib (Python 3.8.11) Line 19, Col 1 UTF-8 CRLF RW Mem 61%

18°C Sunny 07:25 22-10-2021



# Chandigarh Engineering College Jhanjeri

## Mohali - 140307

### Department of Computer Science & Engineering

Spyder (Python 3.8)

```
C:\Users\Lenovo.spyder-py3\LLFRAS\Merge Capture with Get Attendance.py
```

temp.py | Live Logging Attendance 3.py | Merge Capture with Get Attendance.py

```
1419 root.title('Live Logging Face Recognition Attendance System')
1420 root.iconbitmap('C:/Users/Lenovo/Downloads/images.ico')
1421
1422 Button(root, text='<<', state = DISABLED, padx = 10).grid(row=0, column=0)
1423 Button(root, text='>>', state = DISABLED, padx=10).grid(row=0, column=1)
1424
1425 Label(root, text = ' ').grid(row = 1)
1426 Label(root, text = 'Welcome To').grid(row = 2, column = 3)
1427 Label(root, text = 'The Live Logging Facial Recognition').grid(row = 3, column = 3)
1428 Label(root, text = 'Attendance System').grid(row=4, column = 3)
1429
1430
1431 row_value = 3
1432 for i in range(5):
1433     welcome_3 = Label(root, text = '          ').grid(row=row_value)
1434     row_value = row_value +1
1435
1436 def image_path():
1437     global path_1, path_8
1438     path_1 = None
1439     path_8 = new_path + '/Image Directory'
1440     try:
1441         os.mkdir(path_8)
1442     except FileNotFoundError:
1443         pass
1444
1445 def attendance_path():
1446     global path_1, path_8
1447     path_8 = None
1448     path_1 = new_path + '/Attendance Directory'
1449     try:
1450         os.mkdir(path_1)
1451     except FileNotFoundError:
1452         pass
1453
1454 def get_attendance_path():
1455     global path_15
1456     path_15 = new_path + '/Attendance Directory/branch'
1457
1458 global new_path
```

Type here to search

File Edit Search Source Run Debug Consoles Projects Tools View Help

C:\Users\Lenovo\Desktop\Attendance

Name Date Modified

> Attendance Directory 01-10-2021 18:31

> Image Directory 01-10-2021 18:45

Console I/A X

In [2]: runcell(0, 'C:/Users/Lenovo/.spyder-py3/LLFRAS/Merge Capture with Get Attendance.py')

In [3]:

LSP Python: ready conda: env\_dlib (Python 3.8.11) Line 19, Col 1 UTF-8 CRLF RW Mem 60%

18°C Sunny 07:25 22-10-2021

Spyder (Python 3.8)

```
C:\Users\Lenovo.spyder-py3\LLFRAS\Merge Capture with Get Attendance.py
```

temp.py | Live Logging Attendance 3.py | Merge Capture with Get Attendance.py

```
1436 def image_path():
1437     global path_1, path_8
1438     path_1 = None
1439     path_8 = new_path + '/Image Directory'
1440     try:
1441         os.mkdir(path_8)
1442     except FileNotFoundError:
1443         pass
1444
1445 def attendance_path():
1446     global path_1, path_8
1447     path_8 = None
1448     path_1 = new_path + '/Attendance Directory'
1449     try:
1450         os.mkdir(path_1)
1451     except FileNotFoundError:
1452         pass
1453
1454 def get_attendance_path():
1455     global path_15
1456     path_15 = new_path + '/Attendance Directory/branch'
1457
1458 global new_path
1459 get_path = str(Path.home())
1460 split = get_path.split('\\')
1461 new_path = '/'.join(split)+'/Desktop/Attendance'
1462 try:
1463     os.mkdir(new_path)
1464 except FileNotFoundError:
1465     pass
1466 os.chdir(new_path)
1467
1468 proceed_button = Button(root, text = 'Attendance Portal', command = lambda: [root.withdraw(), attendance])
1469 student_button = Button(root, text = "Student's Portal", command = lambda: [root.withdraw(), get_attend])
1470 image_button = Button(root, text = 'Image Portal', command = lambda: [root.withdraw(), image_path()])
1471 button_exit = Button(root, text = 'Exit', command = root.destroy, padx = 10).grid(row = row_value+2, column = 3)
1472
1473 root.mainloop()
```

Type here to search

File Edit Search Source Run Debug Consoles Projects Tools View Help

C:\Users\Lenovo\Desktop\Attendance

Name Date Modified

> Attendance Directory 01-10-2021 18:31

> Image Directory 01-10-2021 18:45

Console I/A X

In [2]: runcell(0, 'C:/Users/Lenovo/.spyder-py3/LLFRAS/Merge Capture with Get Attendance.py')

In [3]:

LSP Python: ready conda: env\_dlib (Python 3.8.11) Line 19, Col 1 UTF-8 CRLF RW Mem 61%

18°C Sunny 07:25 22-10-2021



**Chandigarh Engineering College Jhanjeri  
Mohali - 140307**

**Department of Computer Science & Engineering**

## Results and Discussion

The screenshot shows the Spyder Python IDE interface. On the left, the code editor displays a script named `#5 message box #1 Root Window FARS V39.py`. The script uses the Tkinter library to create a window with a listbox containing various branch names: Aero, Agriculture, Ai, Civil, Computer science, Electrical, and Mechanical. An 'Exit' button is also present. On the right, the IPython console window is open, showing the Python environment details and the command `In [1]: runcell(0, 'C:/Users/0526p/.spyder-py3/GUI in LLFRAS/#5 message box #1 Root Window FARS V39.py')`.

```
C:/Users/0526p/.spyder-py3/GUI in LLFRAS/#5 message box #1 Root Window FARS V39.py

20
21 root = Tk()
22
23
24 def createSubjectCsvFile(getInput, path_6):
25     path = path_6 + '/' + getInput+'.csv'
26     with open(path, 'w', newline='') as f:
27         theWriter = csv.writer(f)
28         theWriter.writerow(['Name', 'Roll No'])

29
30
31 def findEncodings(images):
32     encodeList = []
33     for img in images:
34         img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
35         encode = face_recognition.face_encodings(img)[0]
36         encodeList.append(encode)

37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
59
```

The screenshot shows the Spyder Python IDE interface. The top bar displays the path 'C:/Users/0526p/.spyder-py3/GUI in LLFRAS - Spyder (Python 3.6)'. The main window has a title 'Live Logging Face Recognition Attendance System' and contains a purple header bar with the text 'Welcome To The Live Logging Facial Recognition Attendance System'. Below this are three buttons: 'Attendance Portal', 'Student's Portal', and 'Image Portal', followed by an 'Exit' button. The central area is a light yellow placeholder. On the left, there's a file tree with a single item 'GU'. The bottom-left pane shows the Python script code:

```
13 import numpy as np
14 import face_recognition
15 import os
16 import datetime
17 import pytz
18 import csv
19
20
21 root = Tk()
22
23
24 def createSubjectCsvFile(getInput, path_6):
25     path = path_6 + '/' + getInput+'.csv'
26     with open(path, 'w', newline='') as f:
27         theWriter = csv.writer(f)
28         theWriter.writerow(['Name', 'Roll No'])
29
30
31 def findEncodings(images):
32     encodeList = []
33     for img in images:
34         img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
35         encode = face_recognition.face_encodings(img)[0]
36         encodeList.append(encode)
```

The right side of the interface includes a file browser showing '#1 Root Window FARS V39.py 01-12-2021 13:39', a 'Console' tab with Python and IPython history, and a status bar at the bottom.



# Chandigarh Engineering College Jhanjeri

## Mohali - 140307

### Department of Computer Science & Engineering

The screenshot shows the Spyder IDE interface with a Python script and its execution results.

**Python Script (C:/Users/0526p/spyder-py3/GUI in LLFRAS/#5 message box/#1 Root Window FARS V39.py):**

```
23
24     def createSubjectCsvFile(getInput, path_6):
25         path = path_6 + '/' + getInput+'.csv'
26         with open(path, 'w', newline='') as f:
27             thewriter = csv.writer(f)
28             thewriter.writerow(['Name', 'Roll No'])
29
30
31     def findEncodings(images):
32         encodeList = []
33         for img in images:
34             img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
35             encode = face_recognition.face_encodings(img)[0]
```

**Execution Results:**

```
In [1]: runcell(0, 'C:/Users/0526p/spyder-py3/GUI in LLFRAS/#5 message box/#1 Root Window FARS V39.py')
```

The screenshot also shows a Windows file explorer window showing a file named "#1 Root Window FARS V39.py" from 01-12-2021 at 13:39.

The screenshot shows the Spyder IDE interface with a Python script and its execution results.

**Python Script (C:/Users/0526p/spyder-py3/GUI in LLFRAS/#5 message box/#1 Root Window FARS V39.py):**

```
19
20
21     root = Tk()
22
23
24     def createSubjectCsvFile(getInput, path_6):
25         path = path_6 + '/' + getInput+'.csv'
26         with open(path, 'w', newline='') as f:
27             thewriter = csv.writer(f)
28             thewriter.writerow(['Name', 'Roll No'])
29
30
31     def findEncodings(images):
32         encodeList = []
33         for img in images:
34             img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
35             encode = face_recognition.face_encodings(img)[0]
```

**Execution Results:**

```
In [1]: runcell(0, 'C:/Users/0526p/spyder-py3/GUI in LLFRAS/#5 message box/#1 Root Window FARS V39.py')
```

The screenshot also shows a Windows file explorer window showing a file named "#1 Root Window FARS V39.py" from 01-12-2021 at 13:39.



# Chandigarh Engineering College Jhanjeri

## Mohali - 140307

### Department of Computer Science & Engineering

The screenshot shows the Spyder IDE interface. On the left, a tree view displays a 'GUI' folder containing several sub-folders like 'Attendance' and 'Attendance Directory/branch'. A list box titled 'Branch Name' shows items such as Aero, Agriculture, Ai, Chemical engineering, Civil, Computer science, Electrical, and Mechanical. The main code editor window contains the following Python code:

```
18 import csv
19
20
21 root = Tk()
22
23
24 def createSubjectCsvFile(getInput, path_6):
25     path = path_6 + '/' + getInput+'.csv'
26     with open(path, 'w', newline='') as f:
27         thewriter = csv.writer(f)
28         thewriter.writerow(['Name', 'Roll No'])
29
30
31 def findEncodings(images):
32     encodeList = []
33     for img in images:
34         img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
35         encode = face_recognition.face_encodings(img)[0]
```

The right side of the interface includes a file browser, a variable explorer, plots, and an IPython console. The IPython console shows the command `In [1]: runcell(0, 'C:/Users/0526p/spyder-py3/GUI in LLFRAS/#5 message box/#1 Root Window FARS V39.py')`.

This screenshot is similar to the previous one but shows checkboxes next to the branch names in the list box. The code in the editor remains the same as the first screenshot.



# Chandigarh Engineering College Jhanjeri

## Mohali - 140307

### Department of Computer Science & Engineering

C:/Users/0526p/spyder-py3/GUI in LLFRAS - Spyder (Python 3.6)

File Edit Search Source Run Debug Consoles Projects Tools View Help

24. matplotlib charts.py #1 Root Window FARS V39.py

```
1 # -*- coding: utf-8 -*-
2
3
4
5
6
7 fr = SI No Branch Name
8 fr 1 Agriculture
9 fr 2 Ai
10 fr 3 Civil
11 im 4 Computer science
12 im 5 Electrical
13 im 6 Mechanical
14
15
16
17
18
19
20
21
22
23
24 def createSubjectCsvFile(getInput, path_6):
25     path = path_6 + '/' + getInput+'.csv'
26     with open(path, 'w', newline='') as f:
27         thewriter = csv.writer(f)
28         thewriter.writerow(['Name', 'Roll No'])
29
30
31 def findEncodings(images):
32     encodeList = []
33     for img in images:
34         img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
35         encode = face_recognition.face_encodings(img)[0]
```

Console 1/A X

```
Python 3.6.13 |Anaconda, Inc.| (default, Mar 16 2021, 11:37:27) [MSC v.1916 64 bit (AMD64)]
Type "copyright", "credits" or "license" for more information.

IPython 7.16.1 -- An enhanced Interactive Python.

In [1]: runcell(0, 'C:/Users/0526p/spyder-py3/GUI in LLFRAS/#5 message box/#1 Root Window FARS V39.py')
```

LSP Python: ready conda() Line 14, Col 24 UTF-8 CRLF RW Mem 90%

C:/Users/0526p/spyder-py3/GUI in LLFRAS - Spyder (Python 3.6)

File Edit Search Source Run Debug Consoles Projects Tools View Help

24. matplotlib charts.py #1 Root Window FARS V39.py

```
1 # -*- coding: utf-8 -*-
2
3
4
5
6
7 fr = SI No Branch Name
8 fr 1 Agriculture
9 fr 2 Ai
10 fr 3 Civil
11 im 4 Computer science
12 im 5 Electrical
13 im 6 Mechanical
14
15
16
17
18
19
20
21
22
23
24 def createSubjectCsvFile(getInput, path_6):
25     path = path_6 + '/' + getInput+'.csv'
26     with open(path, 'w', newline='') as f:
27         thewriter = csv.writer(f)
28         thewriter.writerow(['Name', 'Roll No'])
29
30
31 def findEncodings(images):
32     encodeList = []
33     for img in images:
34         img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
35         encode = face_recognition.face_encodings(img)[0]
```

Console 1/A X

```
Python 3.6.13 |Anaconda, Inc.| (default, Mar 16 2021, 11:37:27) [MSC v.1916 64 bit (AMD64)]
Type "copyright", "credits" or "license" for more information.

IPython 7.16.1 -- An enhanced Interactive Python.

In [1]: runcell(0, 'C:/Users/0526p/spyder-py3/GUI in LLFRAS/#5 message box/#1 Root Window FARS V39.py')
```

LSP Python: ready conda() Line 14, Col 24 UTF-8 CRLF RW Mem 90%



# Chandigarh Engineering College Jhanjeri

## Mohali - 140307

### Department of Computer Science & Engineering

The screenshot shows the Spyder Python IDE interface. On the left, the code editor displays a Python script named `LLFRAS/#5 message box/#1 Root Window FARS V39.py`. The script uses the `tkinter` module to create a window with a listbox containing batch names (2022, 2023, 2024) and a button labeled "Exit". On the right, the IPython console shows the command `In [1]: runcell(0, 'C:/Users/0526p/spyder-py3/GUI in LLFRAS/#5 message box/#1 Root Window FARS V39.py')` and its output, which is the same window displayed in the code editor.

```
File Edit Search Source Run Debug Consoles Projects Tools View Help
File Edit Search Source Run Debug Consoles Projects Tools View Help
C:\Users\0526p\spyder-py3\GUI in LLFRAS\#5 message box\#1 Root Window FARS V39.py
C:\Users\0526p\Desktop\Attendance\Attendance Dire...
SI No Batch Name
1 2022
2 2023
3 2024
Exit
import numpy as np
import face_recognition
import os
import datetime
import pytz
import csv
root = Tk()
def createSubjectCsvFile(getInput, path_6):
    path = path_6 + '/' + getInput+'.csv'
    with open(path, 'w', newline='') as f:
        thewriter = csv.writer(f)
        thewriter.writerow(['Name', 'Roll No'])
def findEncodings(images):
    encodeList = []
    for img in images:
        img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
        encode = face_recognition.face_encodings(img)[0]
        encodeList.append(encode)
SI No Section Name
1 A
import numpy as np
import face_recognition
import os
import datetime
import pytz
import csv
root = Tk()
def createSubjectCsvFile(getInput, path_6):
    path = path_6 + '/' + getInput+'.csv'
    with open(path, 'w', newline='') as f:
        thewriter = csv.writer(f)
        thewriter.writerow(['Name', 'Roll No'])
def findEncodings(images):
    encodeList = []
    for img in images:
        img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
        encode = face_recognition.face_encodings(img)[0]
        encodeList.append(encode)
In [1]: runcell(0, 'C:/Users/0526p/spyder-py3/GUI in LLFRAS/#5 message box/#1 Root Window FARS V39.py')
Out[1]: C:\Users\0526p\spyder-py3\GUI in LLFRAS\#5 message box\#1 Root Window FARS V39.py
Python 3.6.13 |Anaconda, Inc.| (default, Mar 16 2021, 11:37:27) [MSC v.1916 64 bit (AMD64)]
Type "copyright", "credits" or "license" for more information.
IPython 7.16.1 -- An enhanced Interactive Python.
In [1]: runcell(0, 'C:/Users/0526p/spyder-py3/GUI in LLFRAS/#5 message box/#1 Root Window FARS V39.py')
History IPython console
LSP Python: ready conda() Line 14, Col 24 UTF-8 CRLF RW Mem 91%
```

The screenshot shows the Spyder Python IDE interface. On the left, the code editor displays a Python script named `LLFRAS/#5 message box/#1 Root Window FARS V39.py`. The script uses the `tkinter` module to create a window with a listbox containing section names (A) and a button labeled "Exit". On the right, the IPython console shows the command `In [1]: runcell(0, 'C:/Users/0526p/spyder-py3/GUI in LLFRAS/#5 message box/#1 Root Window FARS V39.py')` and its output, which is the same window displayed in the code editor.

```
File Edit Search Source Run Debug Consoles Projects Tools View Help
File Edit Search Source Run Debug Consoles Projects Tools View Help
C:\Users\0526p\spyder-py3\GUI in LLFRAS\#5 message box\#1 Root Window FARS V39.py
C:\Users\0526p\Desktop\Attendance\Attendance Dire...
SI No Section Name
1 A
Exit
import numpy as np
import face_recognition
import os
import datetime
import pytz
import csv
root = Tk()
def createSubjectCsvFile(getInput, path_6):
    path = path_6 + '/' + getInput+'.csv'
    with open(path, 'w', newline='') as f:
        thewriter = csv.writer(f)
        thewriter.writerow(['Name', 'Roll No'])
def findEncodings(images):
    encodeList = []
    for img in images:
        img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
        encode = face_recognition.face_encodings(img)[0]
        encodeList.append(encode)
SI No Section Name
1 A
import numpy as np
import face_recognition
import os
import datetime
import pytz
import csv
root = Tk()
def createSubjectCsvFile(getInput, path_6):
    path = path_6 + '/' + getInput+'.csv'
    with open(path, 'w', newline='') as f:
        thewriter = csv.writer(f)
        thewriter.writerow(['Name', 'Roll No'])
def findEncodings(images):
    encodeList = []
    for img in images:
        img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
        encode = face_recognition.face_encodings(img)[0]
        encodeList.append(encode)
In [1]: runcell(0, 'C:/Users/0526p/spyder-py3/GUI in LLFRAS/#5 message box/#1 Root Window FARS V39.py')
History IPython console
LSP Python: ready conda() Line 14, Col 24 UTF-8 CRLF RW Mem 90%
```



# Chandigarh Engineering College Jhanjeri

## Mohali - 140307

### Department of Computer Science & Engineering

The screenshot shows a Spyder IDE interface with a Python script named 'V39.py' open. The script contains code for creating a Tkinter application titled 'STUDENTS IN SECTION A'. The application has two main components: a listbox displaying student names and a dropdown menu for selecting a semester. The code uses the 'face\_recognition' library to handle face encodings.

```
20
21 root = Tk()
22
23
24 def createSubjectCsvFile(getInput, path_6):
25     path = path_6 + '/' + getInput+'.csv'
26     with open(path, 'w', newline='') as f:
27         thewriter = csv.writer(f)
28         thewriter.writerow(['Name', 'Roll No'])
29
30
31 def findEncodings(images):
32     encodeList = []
33     for img in images:
34         img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
35         encode = face_recognition.face_encodings(img)[0]
```

The screenshot shows a Spyder IDE interface with a Python script named 'V39.py' open. The script contains code for creating a Tkinter application titled 'STUDENTS IN SECTION A'. The application has two main components: a listbox displaying student names and a dropdown menu for selecting a semester. The code uses the 'face\_recognition' library to handle face encodings.

```
20
21 root = Tk()
22
23
24 def createSubjectCsvFile(getInput, path_6):
25     path = path_6 + '/' + getInput+'.csv'
26     with open(path, 'w', newline='') as f:
27         thewriter = csv.writer(f)
28         thewriter.writerow(['Name', 'Roll No'])
29
30
31 def findEncodings(images):
32     encodeList = []
33     for img in images:
34         img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
35         encode = face_recognition.face_encodings(img)[0]
```



# Chandigarh Engineering College Jhanjeri

## Mohali - 140307

### Department of Computer Science & Engineering

The screenshot shows the Spyder Python IDE interface. The code editor displays a script named `Attendance.py` which creates a file explorer application using Tkinter. The application has a main window with tabs for 'Start', 'Home', and 'New/Remove'. A sidebar lists files in a directory tree. The Python console shows the command `In [1]: runcell(0, 'C:/Users/0526p/spyder-py3/GUI in LLFRAS/#1 Root Window FARS V39.py')` and the response "IPython 7.16.1 -- An enhanced Interactive Python". The status bar at the bottom indicates "LSP Python: ready" and memory usage of 90%.

```
7 from tkinter import *
8 from tkinter import ttk
9 from tkinter import messagebox
10 from PIL import ImageTk, Image
11 from pathlib import Path
12 import cv2
13 import numpy as np
14 import face_recognition
15 import os
16 import datetime
17 import pytz
18 import csv
19
20
21 root = Tk()
22
23
24 def createSubjectCsvFile(getInput, path_6):
25     path = path_6 + '/' + getInput+'.csv'
26     with open(path, 'w', newline='') as f:
27         thewriter = csv.writer(f)
28         thewriter.writerow(['Name', 'Roll No'])
29
30
31 def findEncodings(images):
32     encodeList = []
33     for img in images:
34         img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
35         encode = face_recognition.face_encodings(img)[0]
```

The screenshot shows the Spyder Python IDE interface. The code editor displays the same `Attendance.py` script, but the application window now shows a dialog box titled "Enter Subject Name" with an input field containing "Enter Subject Name". The Python console shows the command `In [1]: runcell(0, 'C:/Users/0526p/spyder-py3/GUI in LLFRAS/#1 Root Window FARS V39.py')` and the response "IPython 7.16.1 -- An enhanced Interactive Python". The status bar at the bottom indicates "LSP Python: ready" and memory usage of 89%.

```
7 from tkinter import *
8 from tkinter import ttk
9 from tkinter import messagebox
10 from PIL import ImageTk, Image
11 from pathlib import Path
12 import cv2
13 import numpy as np
14 import face_recognition
15 import os
16 import datetime
17 import pytz
18 import csv
19
20
21 root = Tk()
22
23
24 def createSubjectCsvFile(getInput, path_6):
25     path = path_6 + '/' + getInput+'.csv'
26     with open(path, 'w', newline='') as f:
27         thewriter = csv.writer(f)
28         thewriter.writerow(['Name', 'Roll No'])
29
30
31 def findEncodings(images):
32     encodeList = []
33     for img in images:
34         img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
35         encode = face_recognition.face_encodings(img)[0]
```



# Chandigarh Engineering College Jhanjeri

## Mohali - 140307

### Department of Computer Science & Engineering

The screenshot shows the Spyder Python IDE interface. On the left, there's a code editor with Python script code. In the center, a Tkinter-based application window displays a list of subjects with 'Farmacy' selected. On the right, an IPython console window shows the command 'In [1]: runcell(0, 'C:/Users/0526p/spyder-py3/GUI in LLFRAS/#5 message box/#1 Root Window FARS V39.py')' and its output.

```
from tkinter import *
from tkinter import ttk
from tkinter import messagebox
from PIL import ImageTk, Image
from pathlib import Path
import cv2
import numpy as np
import face_recognition
import os
import datetime
import pytz
import csv

root = Tk()

def createSubjectCsvFile(getInput, path_6):
    path = path_6 + '/' + getInput+'.csv'
    with open(path, 'w', newline='') as f:
        thewriter = csv.writer(f)
        thewriter.writerow(['Name', 'Roll No'])

def findEncodings(images):
    encodeList = []
    for img in images:
        img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
        encode = face_recognition.face_encodings(img)[0]
        encodeList.append(encode)
```

This screenshot is nearly identical to the one above, showing the same Spyder interface with the Tkinter application displaying 'Farmacy' and the IPython console running the same command and output.

```
from tkinter import *
from tkinter import ttk
from tkinter import messagebox
from PIL import ImageTk, Image
from pathlib import Path
import cv2
import numpy as np
import face_recognition
import os
import datetime
import pytz
import csv

root = Tk()

def createSubjectCsvFile(getInput, path_6):
    path = path_6 + '/' + getInput+'.csv'
    with open(path, 'w', newline='') as f:
        thewriter = csv.writer(f)
        thewriter.writerow(['Name', 'Roll No'])

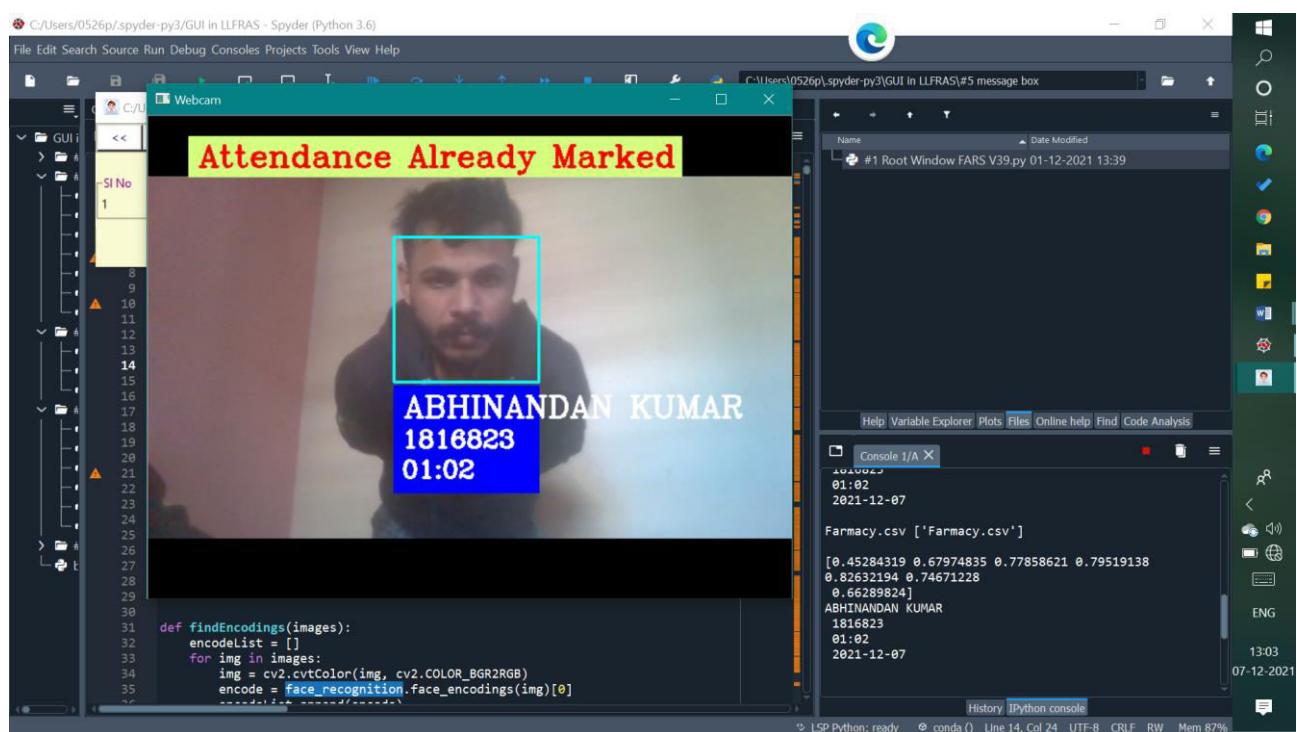
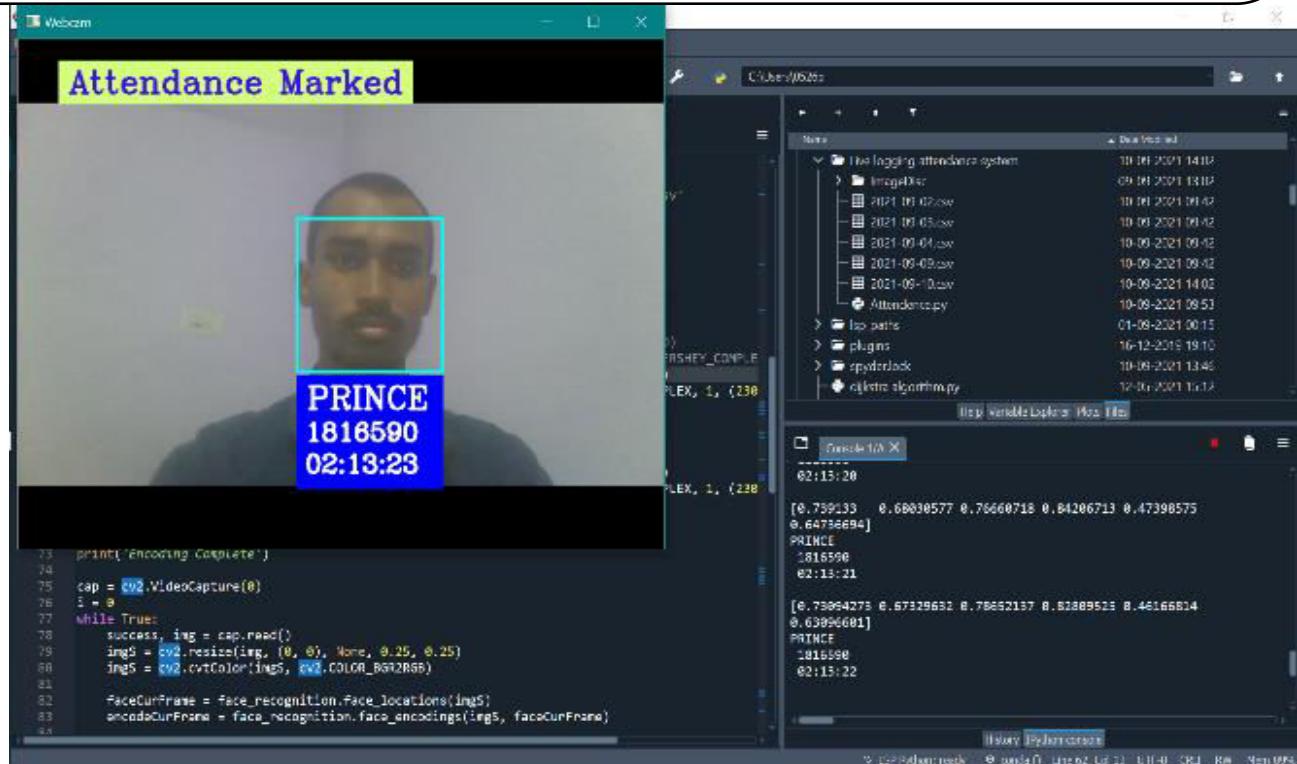
def findEncodings(images):
    encodeList = []
    for img in images:
        img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
        encode = face_recognition.face_encodings(img)[0]
```



# Chandigarh Engineering College Jhanjeri

## Mohali - 140307

### Department of Computer Science & Engineering





**Chandigarh Engineering College Jhanjeri  
Mohali - 140307**

**Department of Computer Science & Engineering**

A screenshot of the Spyder Python IDE interface. On the left is a code editor with a script named 'FARS V39.py' containing Python code for a face recognition application. A message box titled 'Quit Window!' is displayed in the center, asking 'Do you want to quit FARS?' with 'Yes' and 'No' buttons. To the right of the code editor is a file explorer window showing a directory tree with various projects like 'Agriculture', 'AI', 'Civil', etc. At the bottom right is a terminal window showing command-line history and error messages related to image processing.

The screenshot shows the Spyder Python IDE interface. The top menu bar includes File, Edit, Search, Source, Run, Debug, Consoles, Projects, Tools, View, Help. The left sidebar shows a tree view of files and folders, with 'GUI i' selected. The main code editor window displays a Python script named 'V39.py'. The script contains functions for creating a CSV file and finding encodings of images. A message box titled 'Root Window FARS V39' is displayed in the center, showing 'Subject Name: Farmacy'. To the right, a file explorer window shows a list of directories in 'Attendance Directory\branch'. The bottom right corner shows the Jupyter Notebook interface with three cells run, and the status bar at the bottom indicates 'LSP Python: ready'.

```
C:\Users\0526p\spyder-py3\GUI in LLFRAS - Spyder (Python 3.6)
File Edit Search Source Run Debug Consoles Projects Tools View Help

C:\Users\0526p\Desktop\Attendance\Attendance Dire...
C:\Users\0526p\Desktop\Attendance\Attendance Directory\branch

24. matplotlib charts nv x #1 Root Window FARS V39 nv x
C:\Users\0526p\Desktop\Attendance\Attendance Dire...
1
2
3
4
5
6 Subject Name:
7 1 Farmacy
8
9
10
11 from pathlib import Path
12 import cv2
13 import numpy as np
14 import face_recognition
15 import os
16 import datetime
17 import pytz
18 import csv
19
20
21 root = Tk()
22
23
24 def createSubjectCsvFile(getInput, path_6):
25     path = path_6 + '/' + getInput + '.csv'
26     with open(path, 'w', newline='') as f:
27         theWriter = csv.writer(f)
28         theWriter.writerow(['Name', 'Roll No'])
29
30
31 def findEncodings(images):
32     encodeList = []
33     for img in images:
34         img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
35         encode = face_recognition.face_encodings(img)[0]
36         encodeList.append(encode)

Name Date Modified
> Agriculture 29-11-2021 11:59
> Ai 30-11-2021 20:10
> Civil 08-10-2021 01:51
> Computer science 26-09-2021 04:09
> Electrical 25-09-2021 23:47
> Mechanical 25-09-2021 04:45

Help Variable Explorer Plots Files Online help Find Code Analysis
Console I/O X
message box(#1 Root Window FARS V39.py , line 159, in capture
capture
    imgS = cv2.resize(img, (0, 0), None, 0.25, 0.25)
cv2.error: OpenCV(4.2.0) C:\Miniconda3\conda-bld\opencv-suite-1534379934306\work\modules\imgproc\src\resize.cpp:4944: error: (-215:Assertion failed) !ssize.empty() in function 'cv::resize'

In [2]: runcell(0, 'C:/Users/0526p/.spyder-py3/GUI in LLFRAS/#5 message box/#1 Root Window FARS V39.py')
In [3]: runcell(0, 'C:/Users/0526p/.spyder-py3/GUI in LLFRAS/#5 message box/#1 Root Window FARS V39.py')

History IPython console
13:06 07-12-2021
```



# Chandigarh Engineering College Jhanjeri

## Mohali - 140307

### Department of Computer Science & Engineering

The screenshot shows the Spyder IDE interface. On the left, the code editor displays a Python script named '24\_matplotlib\_charts.py'. The script contains code for creating a Tkinter application with a listbox and a message box. The message box asks 'Do you want to quit FARS?'. On the right, the IPython console shows the execution of the script, including the creation of a 'branch' directory and its contents (Agriculture, Ai, Civil, Computer science, Electrical, Mechanical). The console also shows an error related to cv2.resize.

```
C:\Users\0526p\spyder-py3\GUI in LLFRAS #5 message box #1 Root Window FARS V39.py
File Edit Search Source Run Debug Consoles Projects Tools View Help
C:\Users\0526p\Desktop\Attendance\Attendance Directory\branch
Name Date Modified
> Agriculture 29-11-2021 11:59
> Ai 30-11-2021 20:10
> Civil 08-10-2021 01:51
> Computer science 26-09-2021 04:09
> Electrical 25-09-2021 23:47
> Mechanical 25-09-2021 04:45

Help Variable Explorer Plots Files Online help Find Code Analysis
Console 1/A X
capture(subjectName, myList_2, path_6, i, j)
  File "C:/Users/0526p/.spyder-py3/GUI in LLFRAS/#5 message box #1 Root Window FARS V39.py", line 190, in capture
    imgS = cv2.resize(img, (0, 0), None, 0.25, 0.25)
cv2.error: OpenCV(3.4.2) C:/Miniconda3/conda-bld/opencv-suite_1534379934306/work/modules/imgproc/src/resize.cpp:4044: error: (-215:Assertion failed) !ssize.empty() in function 'cv::resize'

In [2]: runcell(0, 'C:/Users/0526p/.spyder-py3/GUI in LLFRAS/#5 message box #1 Root Window FARS V39.py')
13:03 07-12-2021

History IPython console
LSP Python: ready conda () Line 16, Col 16 UTF-8 CRLF RW Mem 88%
```

The screenshot shows the Spyder IDE interface. On the left, the code editor displays the same Python script '24\_matplotlib\_charts.py'. The script has been modified to include a timestamp and author information at the top. On the right, the IPython console shows the execution of the script, including the creation of a 'branch' directory and its contents (Agriculture, Ai, Civil, Computer science, Electrical, Mechanical). The console also shows an error related to cv2.resize.

```
C:\Users\0526p\spyder-py3\GUI in LLFRAS #5 message box #1 Root Window FARS V39.py
File Edit Search Source Run Debug Consoles Projects Tools View Help
C:\Users\0526p\Desktop\Attendance\Attendance Directory\branch
Name Date Modified
> Agriculture 29-11-2021 11:59
> Ai 30-11-2021 20:10
> Civil 08-10-2021 01:51
> Computer science 26-09-2021 04:09
> Electrical 25-09-2021 23:47
> Mechanical 25-09-2021 04:45

Help Variable Explorer Plots Files Online help Find Code Analysis
Console 1/A X
capture(subjectName, myList_2, path_6, i, j)
  File "C:/Users/0526p/.spyder-py3/GUI in LLFRAS/#5 message box #1 Root Window FARS V39.py", line 190, in capture
    imgS = cv2.resize(img, (0, 0), None, 0.25, 0.25)
cv2.error: OpenCV(3.4.2) C:/Miniconda3/conda-bld/opencv-suite_1534379934306/work/modules/imgproc/src/resize.cpp:4044: error: (-215:Assertion failed) !ssize.empty() in function 'cv::resize'

In [2]: runcell(0, 'C:/Users/0526p/.spyder-py3/GUI in LLFRAS/#5 message box #1 Root Window FARS V39.py')
13:06 07-12-2021

In [3]: runcell(0, 'C:/Users/0526p/.spyder-py3/GUI in LLFRAS/#5 message box #1 Root Window FARS V39.py')
In [4]:
History IPython console
LSP Python: ready conda () Line 16, Col 16 UTF-8 CRLF RW Mem 84%
```



# Chandigarh Engineering College Jhanjeri

## Mohali - 140307

### Department of Computer Science & Engineering

The screenshot shows the Spyder Python IDE interface. On the left, the code editor displays the `FARS V39.py` script. The script imports various Python libraries and defines functions for creating subject CSV files and finding encodings of images. On the right, there is a file browser window titled "Attendance Directory\branch" showing a list of sub-directories: Agriculture, Ai, Civil, Computer science, Electrical, and Mechanical. Below the file browser is a terminal window with several command-line entries related to running the script.

```
from tkinter import Tk
import cv2
import numpy as np
import face_recognition
import os
import datetime
import pytz
import csv

root = Tk()

def createSubjectCsvFile(getInput, path_6):
    path = path_6 + '/' + getInput+'.csv'
    with open(path, 'w', newline='') as f:
        thewriter = csv.writer(f)
        thewriter.writerow(['Name', 'Roll No'])

def findEncodings(images):
    encodeList = []
    for img in images:
        img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
        encode = face_recognition.face_encodings(img)[0]
        encodeList.append(encode)
```

This screenshot is similar to the one above, but the GUI window now displays filled student details: Branch (Batch 2021), Section, Semester (1), Subject, and Roll No. The rest of the interface, including the code editor and file browser, remains the same.



# Chandigarh Engineering College Jhanjeri

## Mohali - 140307

### Department of Computer Science & Engineering

```
C:/Users/0526p/spyder-py3/GUI in LLFRAS - Spyder (Python 3.6)
File Edit Search Source Run Debug Consoles Projects Tools View Help
Live Logging Face Recognition Attendance...
Start Home >>
C:/Users/0526p/Desktop/Attendance/Attendance Directory/branch
Name Date Modified
Agriculture 29-11-2021 11:59
Ai 30-11-2021 20:10
Civil 08-10-2021 01:51
Computer science 26-09-2021 04:09
Electrical 25-09-2021 23:47
Mechanical 25-09-2021 04:45
Help Variable Explorer Plots Files Online help Find Code Analysis
Console 1/A X
cv2.error: OpenCV(3.4.2) C:\MINICONDA\conda\3\lib\site-packages\opencv-suite_1534379934306\work\modules\imgproc\src\resize.cpp:4044: error: (-215:Assertion failed) !ssize.empty() in function 'cv::resize'
In [2]: runcell(0, 'C:/Users/0526p/spyder-py3/GUI in LLFRAS/#5 message box/#1 Root Window FARS V39.py')
In [3]: runcell(0, 'C:/Users/0526p/spyder-py3/GUI in LLFRAS/#5 message box/#1 Root Window FARS V39.py')
In [4]: runcell(0, 'C:/Users/0526p/spyder-py3/GUI in LLFRAS/#5 message box/#1 Root Window FARS V39.py')
History IPython console
LSP Python: ready conda () Line 17, Col 12 UTF-8 CRLF RW Mem 84%
```

File Edit Search Source Run Debug Consoles Projects Tools View Help

Live Logging Face Recognition Attendance...

Start Home >>

C:/Users/0526p/Desktop/Attendance/Attendance Directory/branch

Name Date Modified

Agriculture 29-11-2021 11:59

Ai 30-11-2021 20:10

Civil 08-10-2021 01:51

Computer science 26-09-2021 04:09

Electrical 25-09-2021 23:47

Mechanical 25-09-2021 04:45

Help Variable Explorer Plots Files Online help Find Code Analysis

Console 1/A X

cv2.error: OpenCV(3.4.2) C:\MINICONDA\conda\3\lib\site-packages\opencv-suite\_1534379934306\work\modules\imgproc\src\resize.cpp:4044: error: (-215:Assertion failed) !ssize.empty() in function 'cv::resize'

In [2]: runcell(0, 'C:/Users/0526p/spyder-py3/GUI in LLFRAS/#5 message box/#1 Root Window FARS V39.py')

In [3]: runcell(0, 'C:/Users/0526p/spyder-py3/GUI in LLFRAS/#5 message box/#1 Root Window FARS V39.py')

In [4]: runcell(0, 'C:/Users/0526p/spyder-py3/GUI in LLFRAS/#5 message box/#1 Root Window FARS V39.py')

History IPython console

LSP Python: ready conda () Line 17, Col 12 UTF-8 CRLF RW Mem 84%

```
C:/Users/0526p/spyder-py3/GUI in LLFRAS - Spyder (Python 3.6)
File Edit Search Source Run Debug Consoles Projects Tools View Help
Live Logging Face Recognition Attendance...
Start Home >>
C:/Users/0526p/Desktop/Attendance/Attendance Directory/branch
Name Date Modified
Agriculture 29-11-2021 11:59
Ai 30-11-2021 20:10
Civil 08-10-2021 01:51
Computer science 26-09-2021 04:09
Electrical 25-09-2021 23:47
Mechanical 25-09-2021 04:45
Help Variable Explorer Plots Files Online help Find Code Analysis
Console 1/A X
cv2.error: OpenCV(3.4.2) C:\MINICONDA\conda\3\lib\site-packages\opencv-suite_1534379934306\work\modules\imgproc\src\resize.cpp:4044: error: (-215:Assertion failed) !ssize.empty() in function 'cv::resize'
In [2]: runcell(0, 'C:/Users/0526p/spyder-py3/GUI in LLFRAS/#5 message box/#1 Root Window FARS V39.py')
In [3]: runcell(0, 'C:/Users/0526p/spyder-py3/GUI in LLFRAS/#5 message box/#1 Root Window FARS V39.py')
In [4]: runcell(0, 'C:/Users/0526p/spyder-py3/GUI in LLFRAS/#5 message box/#1 Root Window FARS V39.py')
History IPython console
LSP Python: ready conda () Line 17, Col 12 UTF-8 CRLF RW Mem 85%
```

File Edit Search Source Run Debug Consoles Projects Tools View Help

Live Logging Face Recognition Attendance...

Start Home >>

C:/Users/0526p/Desktop/Attendance/Attendance Directory/branch

Name Date Modified

Agriculture 29-11-2021 11:59

Ai 30-11-2021 20:10

Civil 08-10-2021 01:51

Computer science 26-09-2021 04:09

Electrical 25-09-2021 23:47

Mechanical 25-09-2021 04:45

Help Variable Explorer Plots Files Online help Find Code Analysis

Console 1/A X

cv2.error: OpenCV(3.4.2) C:\MINICONDA\conda\3\lib\site-packages\opencv-suite\_1534379934306\work\modules\imgproc\src\resize.cpp:4044: error: (-215:Assertion failed) !ssize.empty() in function 'cv::resize'

In [2]: runcell(0, 'C:/Users/0526p/spyder-py3/GUI in LLFRAS/#5 message box/#1 Root Window FARS V39.py')

In [3]: runcell(0, 'C:/Users/0526p/spyder-py3/GUI in LLFRAS/#5 message box/#1 Root Window FARS V39.py')

In [4]: runcell(0, 'C:/Users/0526p/spyder-py3/GUI in LLFRAS/#5 message box/#1 Root Window FARS V39.py')

History IPython console

LSP Python: ready conda () Line 17, Col 12 UTF-8 CRLF RW Mem 85%



**Chandigarh Engineering College Jhanjeri  
Mohali - 140307**

**Department of Computer Science & Engineering**

The screenshot shows the Spyder Python IDE interface. On the left, there's a 'Live Logging Face Recognition Attendance' window containing a Tkinter-based user interface for selecting student details like Branch (Civil), Batch (2022), Section, Semester (1), Subject, and Roll No. It includes 'Get' and 'Exit' buttons. Below this is a code editor with Python script FARS V39.py. The script imports numpy, face\_recognition, os, datetime, pytz, and csv, then defines functions to create subject CSV files and find encodings of images using OpenCV and face\_recognition libraries. On the right, there's a file explorer showing a directory tree with various branches like Agriculture, Ai, Civil, Computer science, Electrical, and Mechanical, each with a timestamp. At the bottom, the IPython console displays command history and output related to running the script.

```
13 import numpy as np
14 import face_recognition
15 import os
16 import datetime
17 import pytz
18 import csv
19
20
21 root = Tk()
22
23
24 def createSubjectCsvFile(getInput, path_6):
25     path = path_6 + '/' + getInput+'.csv'
26     with open(path, 'w', newline='') as f:
27         theWriter = csv.writer(f)
28         theWriter.writerow(['Name', 'Roll No'])
29
30
31 def findEncodings(images):
32     encodeList = []
33     for img in images:
34         img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
35         encode = face_recognition.face_encodings(img)[0]
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
```

The screenshot shows the Spyder Python IDE interface. On the left, there's a file browser window titled 'Live Logging Face Recognition Attendance' with tabs for 'Start' and 'Home'. The main workspace displays a Python script named 'Attendance FARS V39.py'. The code implements a GUI for entering student details and saving them as CSV files. The right side of the screen shows a file explorer listing various branches like Agriculture, Ai, Civil, etc., along with their dates modified. Below the code editor is a 'Console' tab showing command-line output related to image processing errors. At the bottom, there are tabs for 'History' and 'IPython console'.

```
C:/Users/0526p/spyder-py3/GUI - Spyder (Python 3.6)
File Edit Search Source Run Debug Consoles Projects Tools View Help
Live Logging Face Recognition Attendan... Start Home >> Attendance FARS V39.py
C:\Users\0526p\Desktop\Attendance\Attendance Directory\branch
Name Date Modified
> Agriculture 29-11-2021 11:59
> Ai 30-11-2021 20:10
> Civil 08-10-2021 01:51
> Computer science 26-09-2021 04:09
> Electrical 25-09-2021 23:47
> Mechanical 25-09-2021 04:45

Help Variable Explorer Plots Files Online help Find Code Analysis
Console / | X
cv2.error: OpenCV(4.4.0) C:\OpenCV\build\opencv\include\opencv\imgproc\src\resize.cpp:4044: error: (-215:Assertion failed) ! ssize.empty() in function 'cv::resize'

In [2]: runcell(0, 'C:/Users/0526p/spyder-py3/GUI in LLFRAS/#5 message box/#1 Root Window FARS V39.py')
In [3]: runcell(0, 'C:/Users/0526p/spyder-py3/GUI in LLFRAS/#5 message box/#1 Root Window FARS V39.py')
In [4]: runcell(0, 'C:/Users/0526p/spyder-py3/GUI in LLFRAS/#5 message box/#1 Root Window FARS V39.py')

History IPython console
LSP Python: ready conda () Line 12, Col 12 UTE-B CRLF RW Mem 84%
07-12-2021
```



**Chandigarh Engineering College Jhanjeri  
Mohali - 140307**

**Department of Computer Science & Engineering**

The screenshot shows the Spyder Python IDE interface. On the left, there's a 'Live Logging Face Recognition Attendance' window containing a form with fields for Branch (Civil), Batch (2022), Section (a), Semester (1), and Subject (empty). Below the form are 'Get' and 'Exit' buttons. The main code editor area displays Python code for face recognition, including imports for numpy, face\_recognition, os, datetime, pytz, and csv, followed by functions to create subject CSV files and find encodings. On the right, a file explorer shows a directory tree with various projects like Agriculture, Ai, Civil, Computer science, Electrical, and Mechanical, each with their last modified dates. At the bottom, an IPython console window shows command history and output related to running the script.

```
13 import numpy as np
14 import face_recognition
15 import os
16 import datetime
17 import pytz
18 import csv
19
20
21 root = Tk()
22
23
24 def createSubjectCsvFile(getInput, path_6):
25     path = path_6 + '/' + getInput+'.csv'
26     with open(path, 'w', newline='') as f:
27         theWriter = csv.writer(f)
28         theWriter.writerow(['Name', 'Roll No'])
29
30
31 def findEncodings(images):
32     encodeList = []
33     for img in images:
34         img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
35         encode = face_recognition.face_encodings(img)[0]
36         encodeList.append(encode)
```

The screenshot shows the Spyder Python IDE interface. On the left, a 'Live Logging Face Recognition Attendance' window displays a form for entering student details: Branch (Civil), Batch (2022), Section (a), Semester (1), Subject (maths), and Roll No (1816590). Below the form are two buttons: 'Get' and 'Exit'. The main workspace contains the following Python code:

```
import numpy as np
import face_recognition
import os
import datetime
import pytz
import csv

root = Tk()

def createSubjectCsvFile(getInput, path_6):
    path = path_6 + '/' + getInput + '.csv'
    with open(path, 'w', newline='') as f:
        theWriter = csv.writer(f)
        theWriter.writerow(['Name', 'Roll No'])

def findEncodings(images):
    encodeList = []
    for img in images:
        img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
        encode = face_recognition.face_encodings(img)[0]
        encodeList.append(encode)

#-----
```

To the right of the code editor is a file browser showing a directory structure under 'Attendance Directory\branch':

Name	Date Modified
Agriculture	29-11-2021 11:59
Ai	30-11-2021 20:10
Civil	08-10-2021 01:51
Computer science	26-09-2021 04:09
Electrical	25-09-2021 23:47
Mechanical	25-09-2021 04:45

The bottom right corner shows the Spyder console output:

```
In [2]: runcell(0, 'C:/Users/0526p/spyder-py3/GUI in LLFRAS/#5 message box/#1 Root Window FARS V39.py')

In [3]: runcell(0, 'C:/Users/0526p/spyder-py3/GUI in LLFRAS/#5 message box/#1 Root Window FARS V39.py')

In [4]: runcell(0, 'C:/Users/0526p/spyder-py3/GUI in LLFRAS/#5 message box/#1 Root Window FARS V39.py')
```



# Chandigarh Engineering College Jhanjeri

## Mohali - 140307

### Department of Computer Science & Engineering

The screenshot shows the Spyder Python IDE interface. On the left, there's a file browser with a tree view of files and a code editor window titled 'Root Window FARS V39.py'. The code editor contains Python code for a GUI application. In the center, a message box displays student information and attendance data. On the right, there's a file explorer showing a folder named 'branch' containing various subfolders like 'Agriculture', 'Ai', 'Civil', etc. Below the file explorer is an IPython console window with several command-line entries. The status bar at the bottom indicates 'LSP Python: ready'.

```
C:/Users/0526p/spyder-py3/GUI in LLFRAS - Spyder (Python 3.6)
File Edit Search Source Run Debug Consoles Projects Tools View Help
C:/Users/0526p/Desktop/...
Name Roll No
Prince 1816590
Sl Date Status Time Status
1 2021-09-16 01:46
2 2021-09-17 01:45
3 2021-09-18 10:00
4 2021-09-24 05:48
5 2021-09-26 07:46
6 2021-10-08 12:34
Your Presence : 6 day out of 9 days
Attendance Percent: 66.67%
path_6):
Exit '+'.csv'
with open(path, "w", newline='') as f:
    thewriter = csv.writer(f)
    thewriter.writerow(['Name', 'Roll No'])

def findEncodings(images):
    encodeList = []
    for img in images:
        img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
        encode = face_recognition.face_encodings(img)[0]
        encodeList.append(encode)

2021-09-16 01:46
2021-09-17 01:45
2021-09-18 10:00
2021-09-24 05:48
2021-09-26 07:46
2021-10-08 12:34
Agriculture 29-11-2021 11:59
Ai 30-11-2021 20:10
Civil 08-10-2021 01:51
Computer science 26-09-2021 04:09
Electrical 25-09-2021 23:47
Mechanical 25-09-2021 04:45
Help Variable Explorer Plots Files Online help Find Code Analysis
Console 1/A X
cv2.error: OpenCV(3.4.2) C:\MINICONDA\conda-3\lib\openCV-suite_1534379934306\work\modules\imgproc\src\resize.cpp:4044: error: (-215:Assertion failed) !szie.empty() in function 'cv::resize'
In [2]: runcell(0, 'C:/Users/0526p/spyder-py3/GUI in LLFRAS/#5 message box/#1 Root Window FARS V39.py')
In [3]: runcell(0, 'C:/Users/0526p/spyder-py3/GUI in LLFRAS/#5 message box/#1 Root Window FARS V39.py')
In [4]: runcell(0, 'C:/Users/0526p/spyder-py3/GUI in LLFRAS/#5 message box/#1 Root Window FARS V39.py')
History IPython console
LSP Python: ready conda() Line 17, Col 12 UTF-8 CRLF RW Mem 86%
```

This screenshot shows the same Spyder IDE setup as the previous one, but the central message box has been replaced by a new application window titled 'Live Logging Face Recognition Attendance System'. This window features a purple header with the title and three blue buttons: 'Attendance Portal', 'Student's Portal', and 'Image Portal'. Below the header is a small text area and an 'Exit' button. The code editor on the left remains the same, displaying the 'Root Window FARS V39.py' script. The right side of the screen includes the file explorer, IPython console, and status bar.

```
C:/Users/0526p/spyder-py3/GUI in LLFRAS - Spyder (Python 3.6)
File E Live Logging Face Recognition Attendance System
Name Roll No
Prince 1816590
Sl Date Status Time Status
1 2021-09-16 01:46
2 2021-09-17 01:45
3 2021-09-18 10:00
4 2021-09-24 05:48
5 2021-09-26 07:46
6 2021-10-08 12:34
Your Presence : 6 day out of 9 days
Attendance Percent: 66.67%
path_6):
Exit '+'.csv'
with open(path, "w", newline='') as f:
    thewriter = csv.writer(f)
    thewriter.writerow(['Name', 'Roll No'])

def findEncodings(images):
    encodeList = []
    for img in images:
        img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
        encode = face_recognition.face_encodings(img)[0]
        encodeList.append(encode)

2021-09-16 01:46
2021-09-17 01:45
2021-09-18 10:00
2021-09-24 05:48
2021-09-26 07:46
2021-10-08 12:34
Agriculture 29-11-2021 11:59
Ai 30-11-2021 20:10
Civil 08-10-2021 01:51
Computer science 26-09-2021 04:09
Electrical 25-09-2021 23:47
Mechanical 25-09-2021 04:45
Help Variable Explorer Plots Files Online help Find Code Analysis
Console 1/A X
cv2.error: OpenCV(3.4.2) C:\MINICONDA\conda-3\lib\openCV-suite_1534379934306\work\modules\imgproc\src\resize.cpp:4044: error: (-215:Assertion failed) !szie.empty() in function 'cv::resize'
In [2]: runcell(0, 'C:/Users/0526p/spyder-py3/GUI in LLFRAS/#5 message box/#1 Root Window FARS V39.py')
In [3]: runcell(0, 'C:/Users/0526p/spyder-py3/GUI in LLFRAS/#5 message box/#1 Root Window FARS V39.py')
In [4]: runcell(0, 'C:/Users/0526p/spyder-py3/GUI in LLFRAS/#5 message box/#1 Root Window FARS V39.py')
History IPython console
LSP Python: ready conda() Line 17, Col 12 UTF-8 CRLF RW Mem 84%
```



## Chapter 6

# Conclusion and Future Scope

### 6.1 Conclusion

We made an AI that handles frontend and backend databases, data access controls, file handling, exception handling, Computer Vision and GUI. System controls the interface how to represent data in interfaces, thus user interact with models without any trouble and comfortably. We use Tkinter library for build interface and explain all libraries in chapter 2 which we used for building interface. Using SQL build databases of students. Computer Vision capture the video frames and compare the images form databases. If that person attendance is doesn't marks then it marks the attendance else it marks to 0 for that person. For all particular date it make a new date column where we marks attendance of that particular person. If any person is new in the class and other then it check first that person data is already present in the sheet or not. If that person data is not present in the list then it make a new row for that person and marks the attendance from that date when he presence in the class and rest of the previous date column is marked zero.

There is a three portals Attendance Portal, Student Portal, and Image Portal. In Attendance Portal we generally use these portal for marking attendance and all the process we explain in first paragraph. In Student portal we represent the data of that particular person. So, for accessing the data of particular person first we make an interface where people fill his/her basic information then after he/she will be able to get the data. The data will be like your name, roll no, date-status (all date when you will be present in the class or appear in the class), and times status (the time when you are come on that particular date). After that table it also shows your total no of date (where you are appear for the class) from the total number of date (where class will be done). In the very next row your total percentage (where you present in the class) from the total no days (where class will be happen)

In the Image Portal, we create data of new student or person so, they mark their attendance further. There is provide an interface where we enter the basic data of student or person such as for student branch, batch, section, roll no, first name, middle name, last name, phone no, email id. After that in the next row we click the button and capture the screenshot of person or students.



## **6.2 Future Scope**

As we know few years ago biometric attendance system is launched which is very efficient, very fast, and secure method. We use this method for marking attendance using fingerprint of every single person by person. It reduces the physical effort of file handling and wasting time. It has one disadvantages which we easily mark proxy attendance of any person. System doesn't differentiate that is that person same or different. So, I build the AI which mark attendance without any proxy, and secure method by using face.

Our AI, mark attendance of person using face of particular person. For reducing the proxy attendance marking system, we use a method to track eye blinking and eye motion tracking system which differentiate that is that any image or video file, if it is then it should not mark attendance. If it is not which means person present in real time and the image capture is in real time frame. It using the face patterns and LBPH algorithm for making the face patterns and using face recognition we calculate the face distance of persons/students face. Thus we differentiate the faces between the person/students.

We use this model to everywhere in the world where we need to mark attendance of person. In/out times when he/she will enter in and when he/she will be out such as Security system, Bank security, business purposes, IT organization, School, Institute, College, Universities, and so on...

We also use this technology in online meeting system, online classes, conference, seminar, webinar, and so on.

We also uses the features of entry and exit control in this system. Time limit for marking attendance. If you are not present within this time limit then your attendance will be mark zero. If you are present within this time limit then your attendance will be mark present. It also has a features of entry control what is the time limit you are able to mark attendance. If you are enter the class or something else within this time limit then you are able to mark your attendance.



# Chandigarh Engineering College Jhanjeri

## Mohali - 140307

# Department of Computer Science & Engineering

### 6.1 Download Anaconda –

<https://www.anaconda.com/products/individual - Downloads>

### 6.2 Learn Python –

<https://github.com/PrinceKumarKeshri/Python-Basic>

### 6.3 Tkinter documentation –

<https://docs.python.org/3/library/tkinter.html>

### 6.4 OpenCV documentation –

<https://opencv.org/releases/>

### 6.5 NumPy installation –

<https://numpy.org/install/>

### 6.6 NumPy documentation –

<https://numpy.org/doc/stable/>

### 6.7 Face Recognition –

[https://github.com/ageitgey/face\\_recognition](https://github.com/ageitgey/face_recognition)

### 6.8 OS documentation –

<https://docs.python.org/3/library/os.html>

### 6.9 DateTime documentation –



**Chandigarh Engineering College Jhanjeri**

**Mohali - 140307**

**Department of Computer Science & Engineering**

<https://docs.python.org/3/library/datetime.html>

**6.10 Pytz installation –**

<https://pypi.org/project/pytz/>

**6.11 Pytz documentation –**

<https://www.askpython.com/python-modules/python-pytz-module>

**6.12 CSV documentation –**

<https://docs.python.org/3/library/csv.html>

**6.13 PIL documentation –**

<https://pillow.readthedocs.io/en/stable/>

**6.14 Pathlib documentation –**

<https://docs.python.org/3/library/pathlib.html>