**Attendance System using Face Recognition**

An efficient module that comprises face recognition using OpenCV and Dlib library to manage the attendance records of students using :

1. **OpenCV** -

* To detect faces using haar-cascades files.
* To store the face images for training.
* To train images against names using LBPH algorithm.

1. **Dlib / face\_recognition**

* To detect faces using frontal\_face detector.
* To get encodings of trained images using face\_recognition.
* To match the real time face image with the encodings.

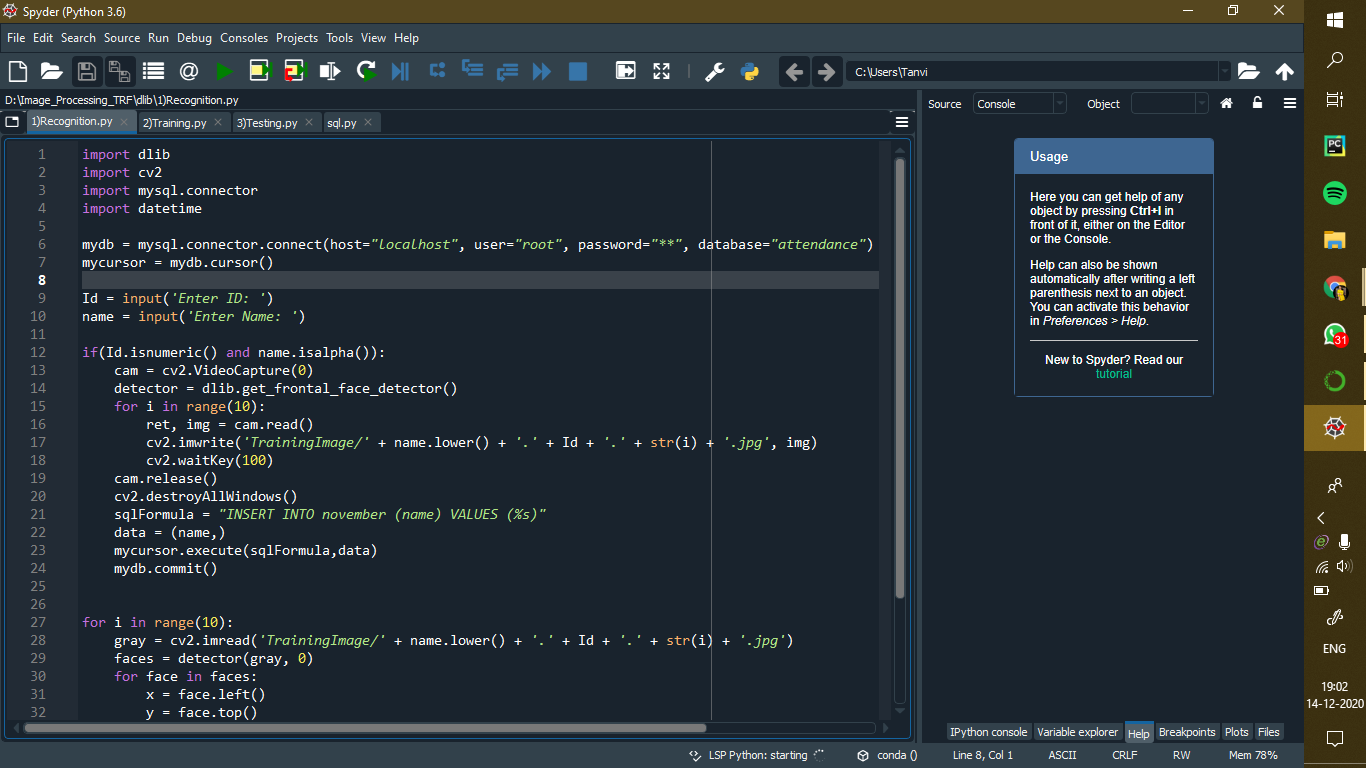
1. **MySQL**

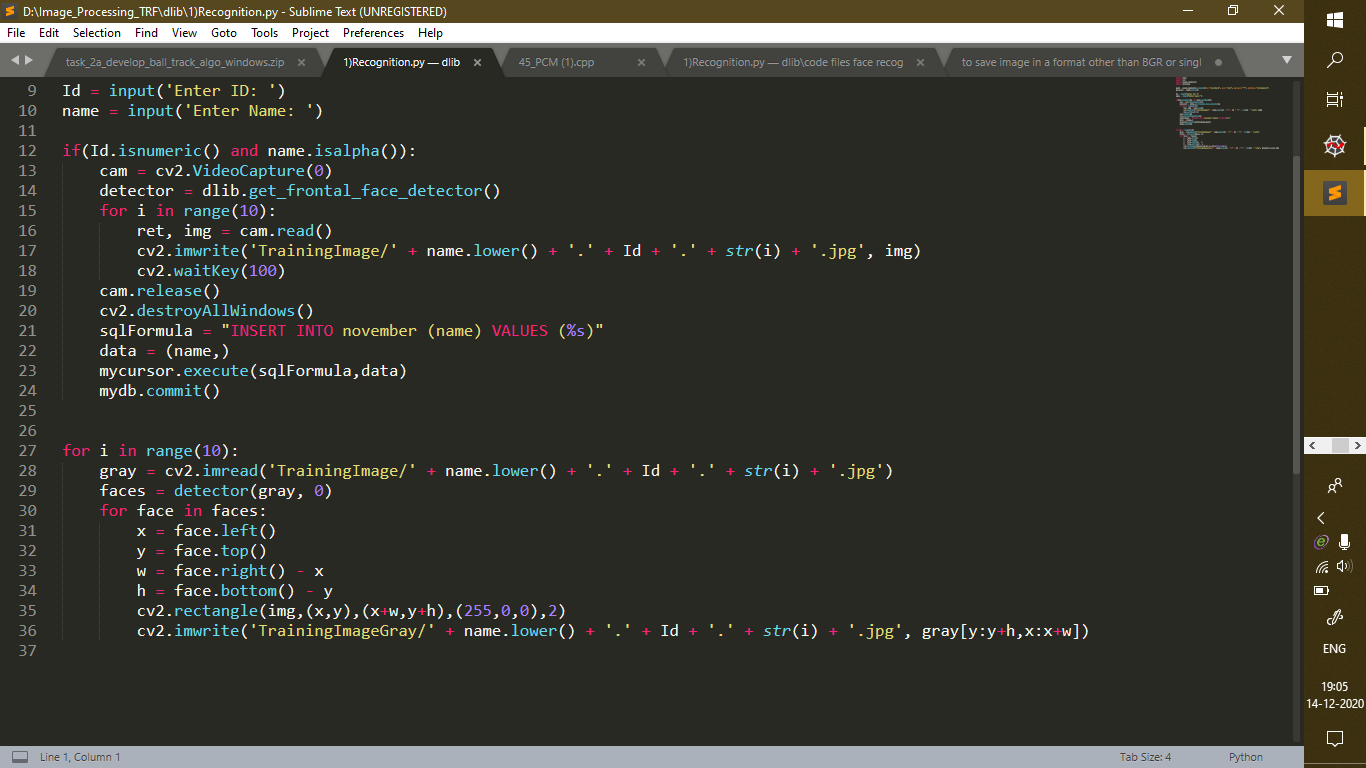
* To mark the attendance time stamp in the database

Steps Involved -

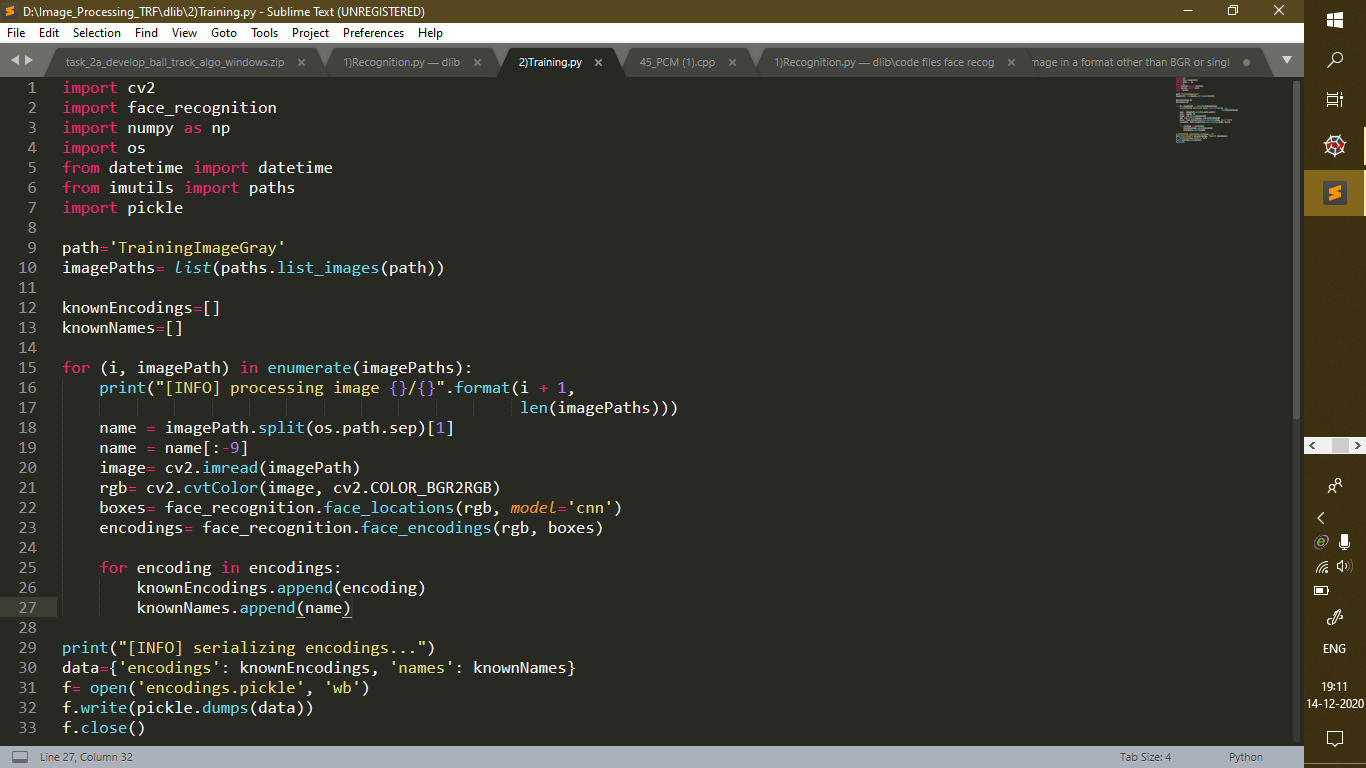
1. Detecting the face using haar cascades (OpenCV) and storing them to form the training dataset.
2. Taking input name and ID from the user and storing it in a CSV file.
3. Converting the training dataset into arrays and training them using LBPH algorithm.
4. Train our model on these values, and save the trained information as **“Trainner.yml”**
5. Loading the ‘haar cascade’ file to identify faces, and the recognizer algorithm to identify the users.

As this method was incapable of detecting faces at all angles accurately so we used Dlib and face\_recognition libraries.Also we replaced the CSV file with SQL database and integrated it with code using MySQL.

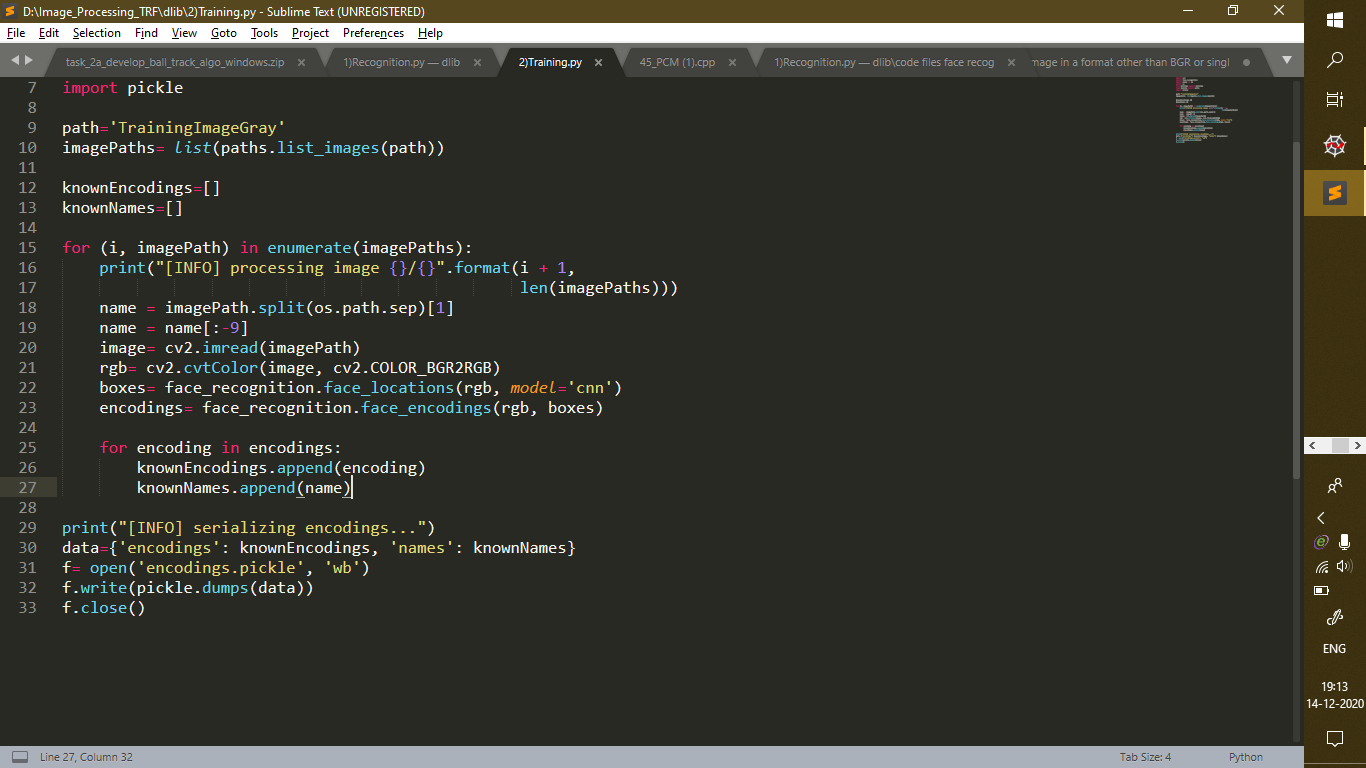


Importing the necessary libraries and establishing connection with the database using mysql.connector. After this we take the name and ID as user input.

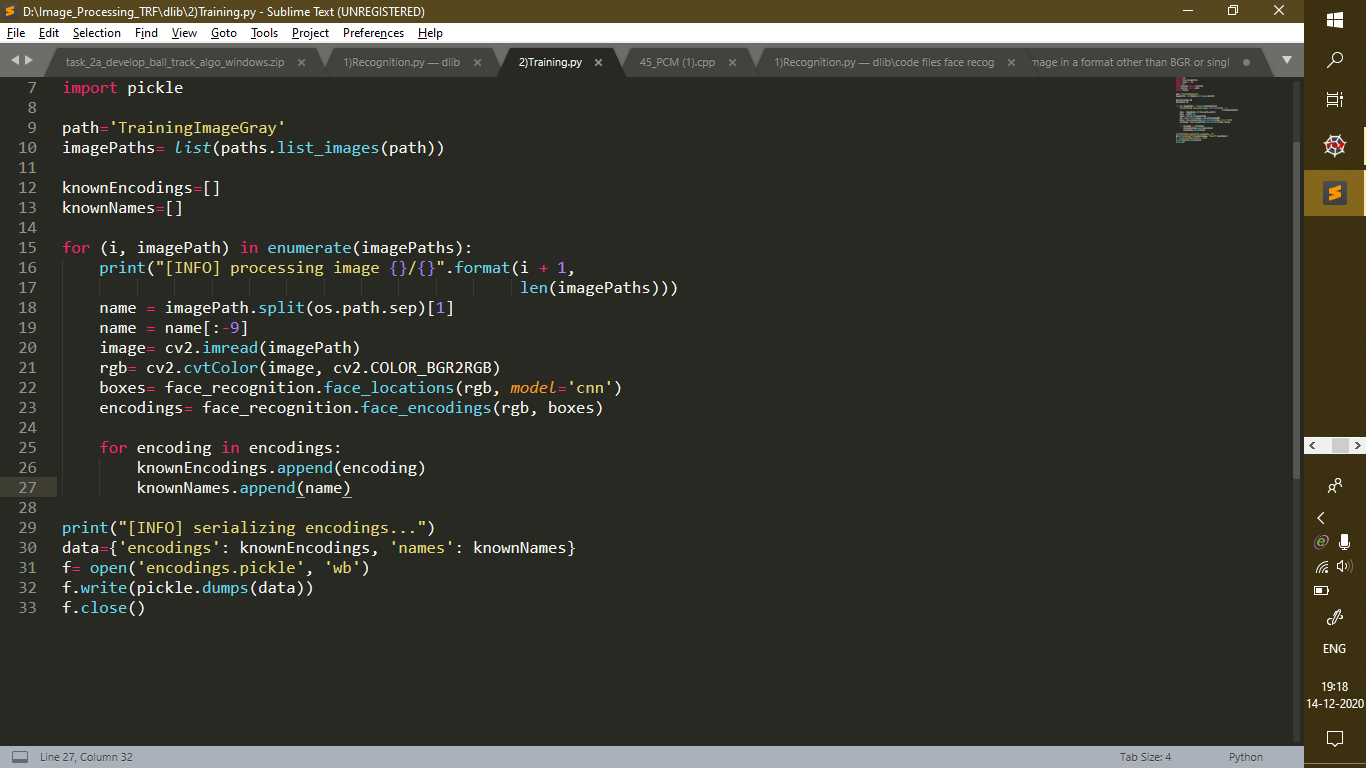
Using for loop capturing images using OpenCV and storing them in the ‘trainingImages’ folder.Inserting the name data into the SQL database.Detecting the face images using dlib’s get\_frontal\_face\_detector().Using this detector we get coordinates for the rectangle around the face that is stored as the training image.



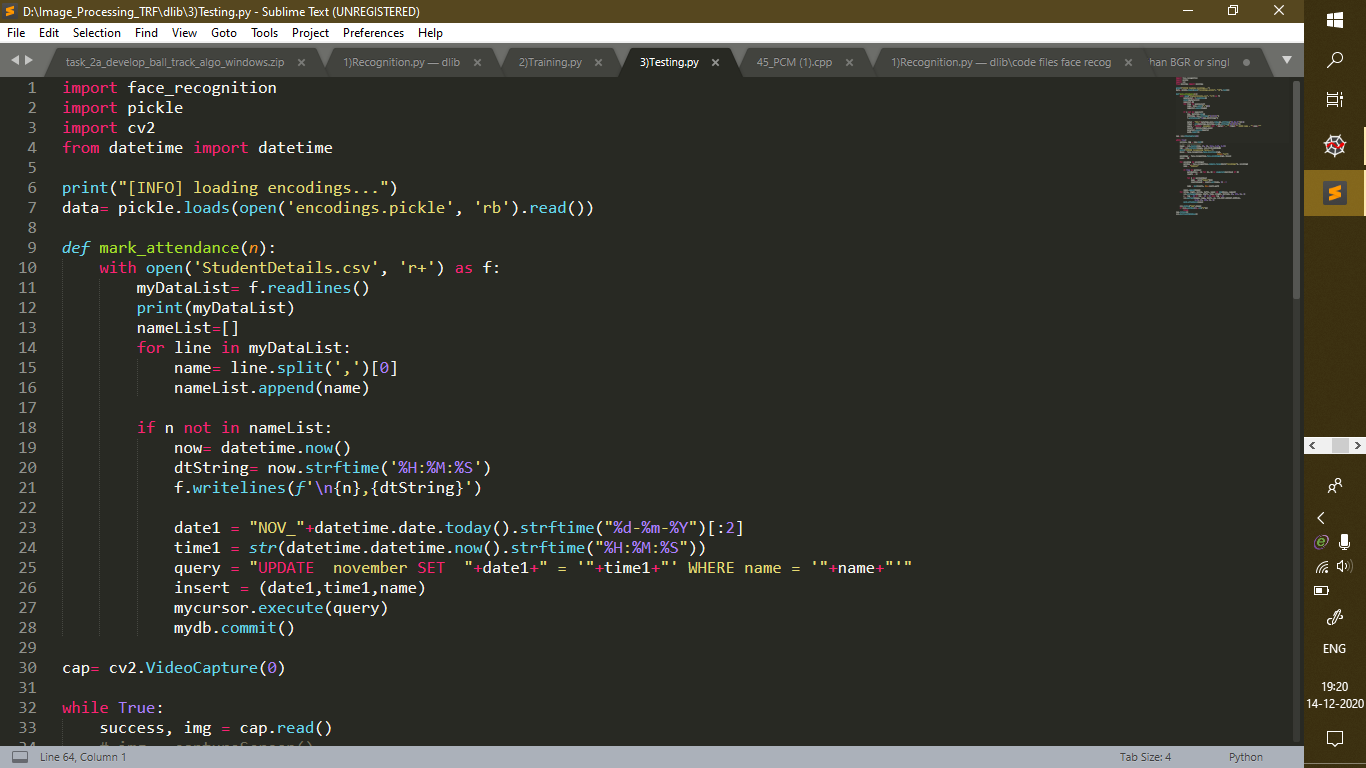
Defining the path of our training dataset and loading images.we created empty lists named knownEncodings and knownNames.



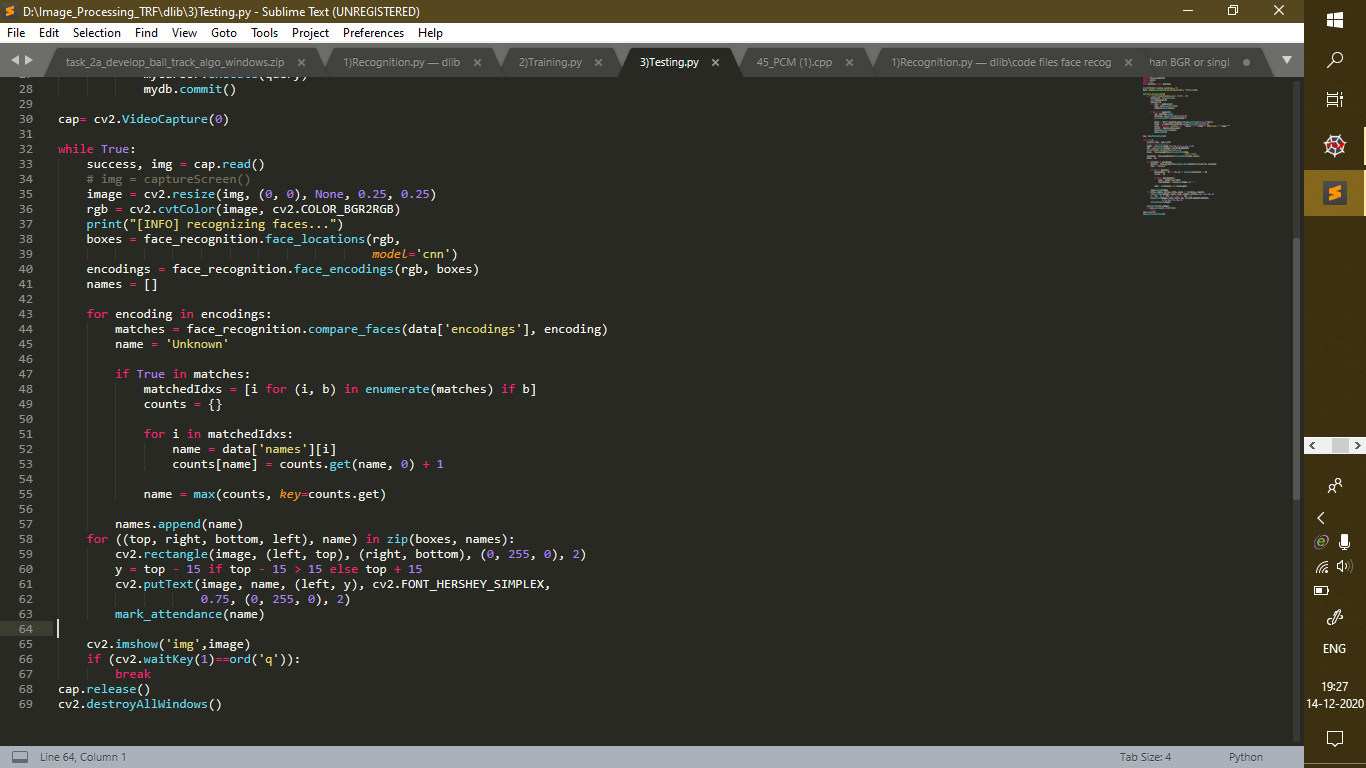
Loading the images and separating names using slicing.Then we converted images from BGR to RGB.Then using face\_locations that works on CNN algorithm we retrieved features. Using face\_encodings we collected encoding for the training images.Then we appended collected names and encodings to the list.



We collected the data from the encoding and name list and converted it into a pickle file for further processing.



Load the pickle file. We got the current date and time using the datetime module. The mark\_attendace function takes the name as input parameter and updates the time stamp in the database.



It accesses the webcam and starts the video stream capturing the face and then converts the BGR to RGB image. Then using face\_locations() and face\_encodings() it extracts the real time face encodings. Then it compares these encodings to the one present in the pickle file and enumerates the count of the times of encodings matched using compare\_faces(). Using these matched encoding it appends the respective name and passes the same to mark\_attendance() function.