**To Mark Attendance Using Face Recognition**

Submitted in partial fulfillment of the requirements for the award of degree of

**BACHELOR OF ENGINEERING**

**IN**

**COMPUTER SCIENCE & ENGINEERING**



**Submitted to:**

**Dr. Rajendra Kumar**

**Submitted By:**

Md. Tauhid Alam, Shreyansh Raut, Subham Sah, Bibek Rawat

18BCS6588, 18BCS1703, 18BCS6537, 18BCS6728

**DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING**

**Chandigarh University, Gharuan**

**Table of Contents**

1. Introduction.
2. Objectives.
3. Feasibility Study.
4. Modules.
5. Innovation in Project.
6. Team Work.
7. Implementation.
8. Output and Validation.
9. References.

**Introduction:**

Attendance in a large class room is hard to be handled by traditional marking system, as it is a Time consuming and prone to error issue. In this project we use Face Recognition, an Image Processing application as a solution to solve this issue. Maintenance and monitoring of attendance records plays a vital role in the analysis of performance of any organization. By developing attendance management system by Face recognition will computerize the traditional way of taking attendance. The system aims to overcome the pitfalls of the existing systems and provides features such as detection of faces, extraction of faces, Detection of extracted features and analysis of student’s attendance. The system uses a large number of features for an increased accuracy. Better accuracy is attained in results as the system takes into account the changes that occur in the face over the period of time and deploys suitable learning algorithms. The modern computation power and huge amounts of data that is been produced by us all through the day are the fuelling ingredients which booms the Artificial Intelligence. We use some Machine Learning Algorithms in this project to train and test our model which makes up a part of Artificial Intelligence.

I have trained the model on data which was previously collected from the student during registration and upon a successful recognition of the candidate, the model will mark him/ her present accordingly. Student’s attendance will be marked on to an excel sheet along with time of detection for a better understanding of student.

**Field:**

**Machine Learning:** It is a specific field in Computer Science which is at its infancy and in which still the development is in progress. The use cases of this domain stretches from predicting whether it would rain today or not to creating a strong Neural Network to make it play and win against any strong human competition and even sometimes win upon itself. The project mainly encompasses the ML domain and makes use of a sophisticated Face Recognition Method for detecting the faces and make use of those facial features to for recognition and Evaluation.

**Objectives:**

Processing which occurs before detection involving face detection and alignment and later recognition is done using the feature extraction and matching steps.

1. **Face Detection:**

The primary function of this step is to conclude whether the human faces emerge in a given image, and what is the location of these faces. The expected outputs of this step are patches which contain each face in the input image. In order to get a more robust and easily designable face recognition system. Face alignment is performed to rationalize the scales and orientation of these patches.

1. **Feature Extraction:**

Following the face detection step the extraction of human face patches from images is done. After this step, the conversion of face patch is done into vector with fixed coordinates or a set of landmark points.

1. **Face Recognition:**

The last step after the representation of faces is to identify them. For automatic recognition we need to build a face database. Various images are taken for each person and their features are extracted and stored in the database. Then when an input image is fed the face detection and feature extraction is performed and its feature to each class is compare and stored in the database.

**Feasibility Study:**

Face detection in humans is a Natural Instinct where we can easily identify a person at any cost (any disturbances, irregularities). But, when it comes to computers it is computationally intensive work. Face detection being a biometric technique implies determination if the image of the face of any particular person matches any of the face images that are stored in a database. This difficulty is tough to resolve automatically because of the changes that several factors, like facial expression, aging and even lighting can affect the image. Facial recognition among the various biometric techniques may not the image. Facial recognition among the various biometric techniques may not be the most authentic but it has various advantages over the others. Here in our case a web cam will be used for capturing the images of students. The captured images are detected and compared with the images in database and the attendance is marked.

As most of the uninteresting and boring topics should be automated, Attendance marking is one of such a topic which consumes a lot of time and we know how the proxies could happen as we see them during every attendance session.

**Modules:**

1. **Setting up Environment.**

The core modules (face\_recognition, dlib) that make up the model were specially made to work only in the Linux Environment. So, In order to install those modules it took a slice of researching the old commuted stackoverflow and github builds and took quite amount of time to come up with a solution.

Like the main idea of installing the C/C++ modules in the Windows OS Environment by using the Visual Studio.

The whole code was done by using Pycharm in Anaconda Environment.

1. **Creating the Core Model.**

The Core model was mainly divided into 3 parts.

1. Detecting the Faces on camera.
2. Recognizing the faces captured and evalutaing with student data.
3. Recording the Attendance information into the database.
4. **Wrapping a Flask application around the Model.**

Building a basic flask web Application to make use of the core model features and to output the results.

**Innovations in Project:**

As we know, Facial recognition is not a new topic. It has been a part of our natural instinct for any living being from the very past in order to classify between the friends and foe. By automating the whole system, fixing the misconceptions and by introducing some security measures we will end up saving a lot of time which is very precious. As of my knowledge we use the HOG (Histogram of Oriented Gradients) as a feature extractor as it uses the gradients which is new boom in this facial recognition task and to classify the face after the feature extraction we may use the SVM (Support Vector Machines) as a classifier for our classification purpose. By introducing the fraud and mischief detection with means of these classifier we can easily get to know about the candidate, who is responsible.

**Team Member wise distribution of work:**

Md. Tauhid Alam (18BCS6588), Shreyansh Raut(18BCS1703), Subham Sah(18BCS6537), Bibek Rawat(18BCS6728), the leader and the group member in the group to do the project. We divided the whole project schema into two parts:

1. Learning Phase.
2. Implementation Phase.

We are currently in the Learning phase as we introducing ourself to the core concepts which make up the project. Before the next submission, I am sure that I would have been started the Implementation part and will show some progress towards the goal. The work distribution of all the team members are given below:

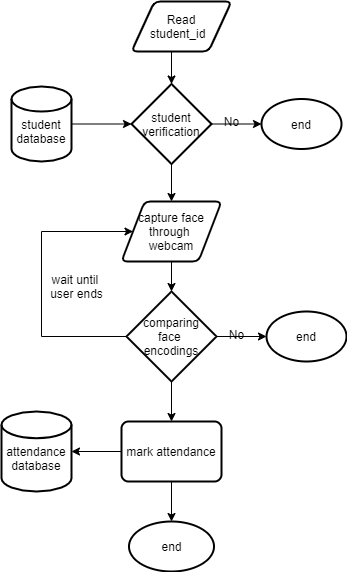
Code: Bibek Rawat, Subham Sah

Designing: Md. Tauhid Alam

Database: Shreyansh Raut

**Implementation:**

The project has completed successfully following all the constraints of project design. I have also built a Flask Application around the system, to make use of it as Web Application.



The core system works as shown in the above flow chart, during the Implementation ‘Python’ was the base programming language and ‘Pycharm’ was used as the coding platform with ‘Anaconda’ as the base Interpreter.

**CODE:**

**For camera:**

import cv2

import face\_recognition

class VideoCamera(object):

    def \_\_init\_\_(self):

        self.video = cv2.VideoCapture(0)

    def \_\_del\_\_(self):

        self.video.release()

    def get\_frame(self):

        success, image = self.video.read()

        # We are using Motion JPEG, but OpenCV defaults to capture raw images,

        # so we must encode it into JPEG in order to correctly display the video stream.

        ret, jpeg = cv2.imencode('.jpg', image)

        return jpeg.tobytes()

    def compare\_faces(self, uid):

        imgA = cv2.cvtColor(cv2.imread('data/imgs/{}.jpg'.format(uid)), cv2.COLOR\_BGR2RGB)

        faceAEncode = face\_recognition.face\_encodings(imgA)[0]

        i = 0

        while True:

            try:

                \_, imgB = self.video.read()

                imgB = cv2.resize(imgB, (0, 0), None, 0.25, 0.25)

                imgB = cv2.cvtColor(imgB, cv2.COLOR\_BGR2RGB)

                facesB = face\_recognition.face\_locations(imgB)

                facesBEncode = face\_recognition.face\_encodings(imgB, facesB)

                match = face\_recognition.compare\_faces(faceAEncode, facesBEncode)

                if match[0]:

                    return 1

                if i > 5:

                    return 2

                i = i + 1

            except:

                return 0

**Main code:**

from flask import Flask, render\_template, Response, request

from camera import VideoCamera

from readWrite import id\_check, mark\_attendence

app = Flask(\_\_name\_\_)

@app.route('/', methods=['GET', 'POST'])

def index():

    message = ''

    if request.method == 'POST':

        uid = request.form.get('uid')

        if id\_check(uid):

            message = '{} working'.format(uid)

            if VideoCamera().compare\_faces(uid) == 1:

                if mark\_attendence(uid):

                    message = '{} marked present.'.format(uid)

                else:

                    message = '{}, Already marked present.'.format(uid)

            elif VideoCamera().compare\_faces(uid) == 2:

                message = 'face Unmatched, Align Correctly!!'

            else:

                message = 'Internal Server Error!!'

        else:

            message = 'Candidate, Not yet Registered'

    return render\_template('index.html', message=message)

def gen(camera):

    while True:

        frame = camera.get\_frame()

        yield (b'--frame\r\n'

               b'Content-Type: image/jpeg\r\n\r\n' + frame + b'\r\n\r\n')

@app.route('/video\_feed')

def video\_feed():

    return Response(gen(VideoCamera()),

                    mimetype='multipart/x-mixed-replace; boundary=frame')

if \_\_name\_\_ == '\_\_main\_\_':

    app.run(host='localhost', debug=True)

**ReadWrite code:**

import os

import csv

from datetime import datetime, date

def id\_check(sid):

    with open('data/student\_data.csv') as file:

        data = {}

        rawData = csv.DictReader(file)

        for row in rawData:

            if row['uid'] == sid:

                data['uid'] = row['uid']

                data['name'] = row['name']

                return True

        else:

            return False

def mark\_attendence(sid):

    file\_name = 'data/Attendence/'+date.today().strftime("%d-%m-%Y")+'.csv'

    if not os.path.isfile(file\_name):

        with open(file\_name, 'w') as file:

            file.write('uid,time')

    with open(file\_name, 'r+') as file:

        rawData = csv.DictReader(file)

        idList = []

        for row in rawData:

            idList.append(row['uid'])

        if sid not in idList:

            now = datetime.now()

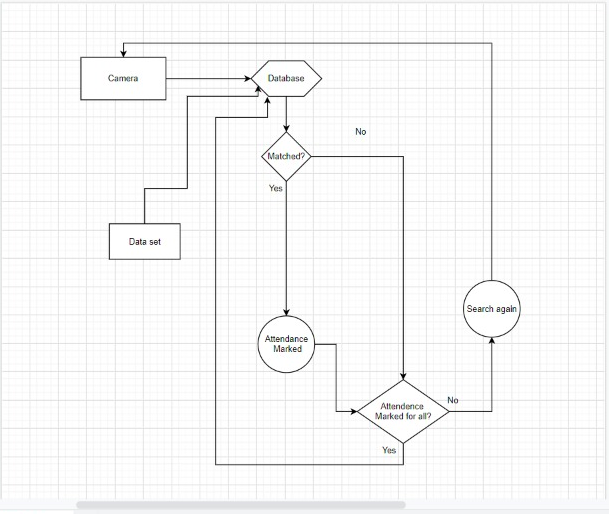
            time = now.strftime('%H:%M:%S')

            file.writelines('\n{},{}'.format(sid, time))

            return True

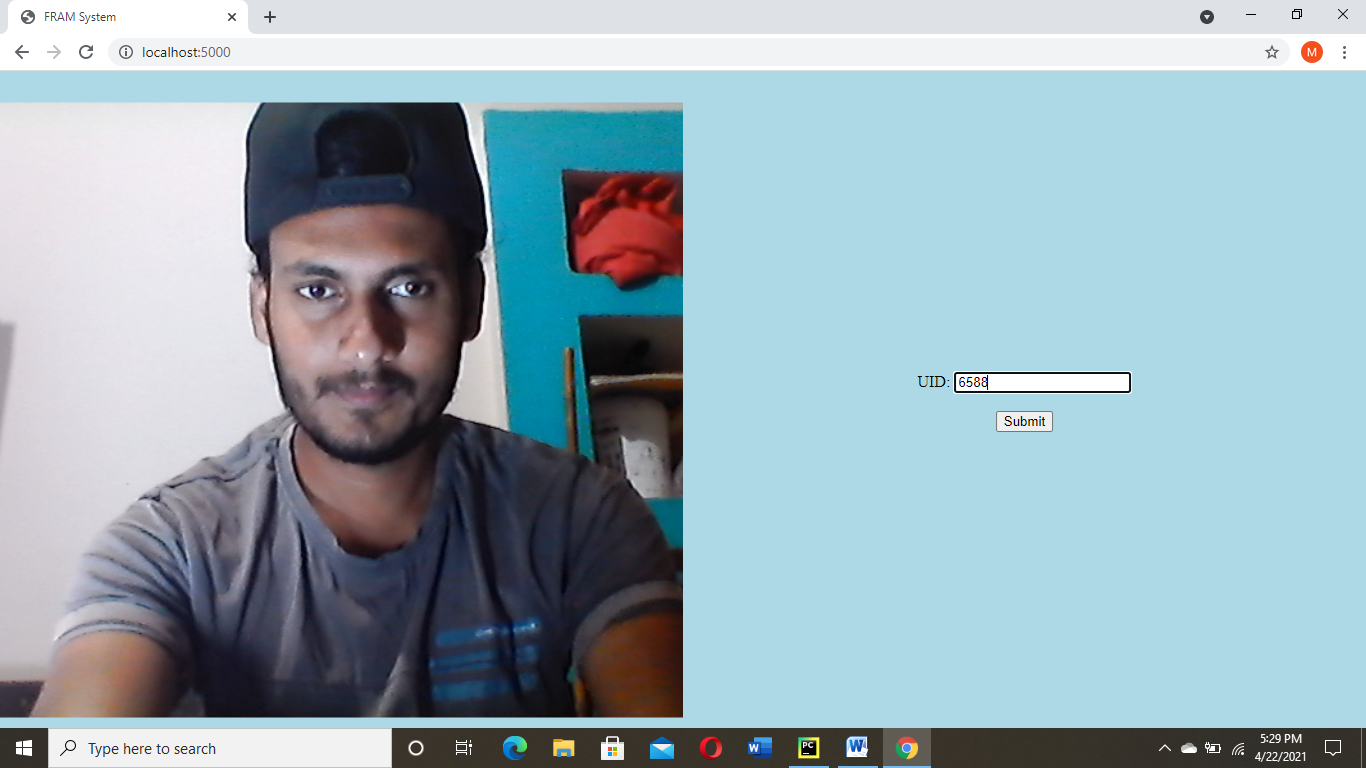
    return False

**DFD (Data Flow Diagram):**

****

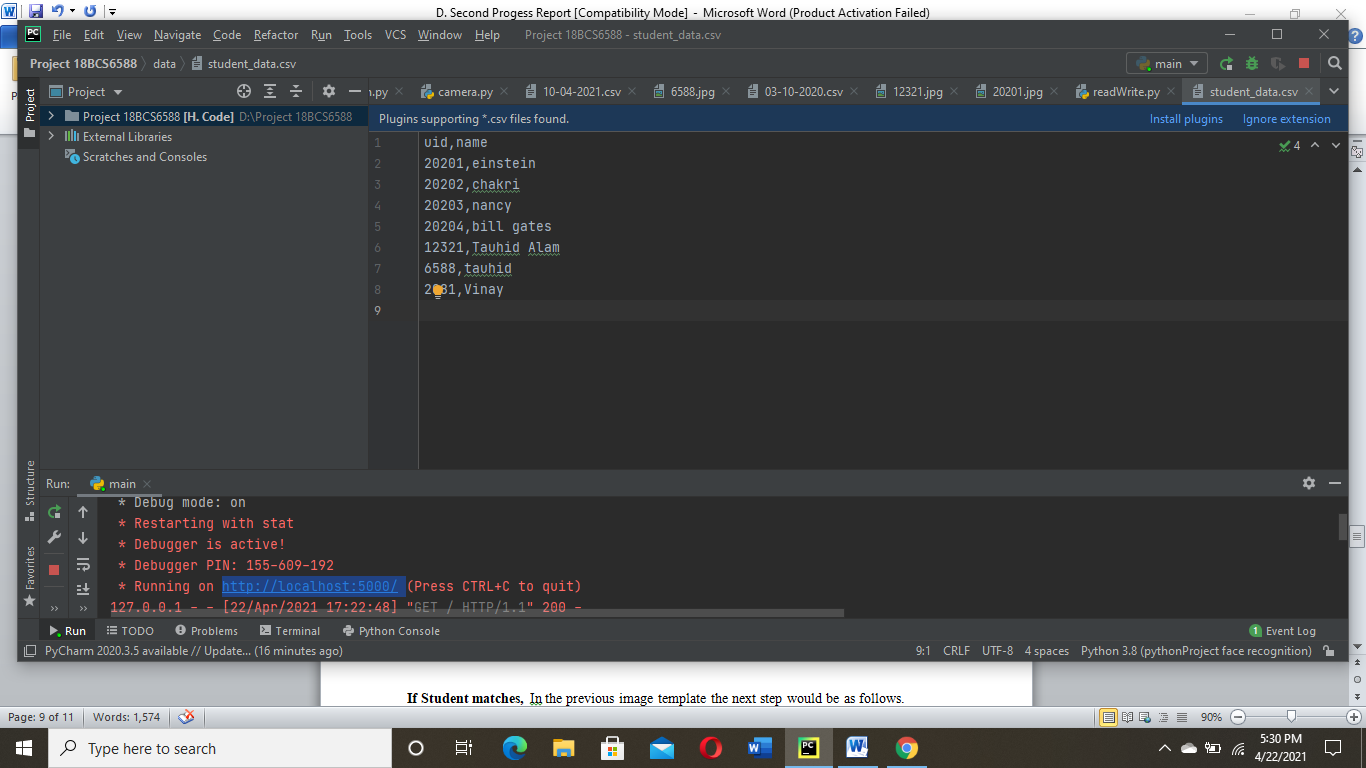
**Typical working:**

**Website Template**



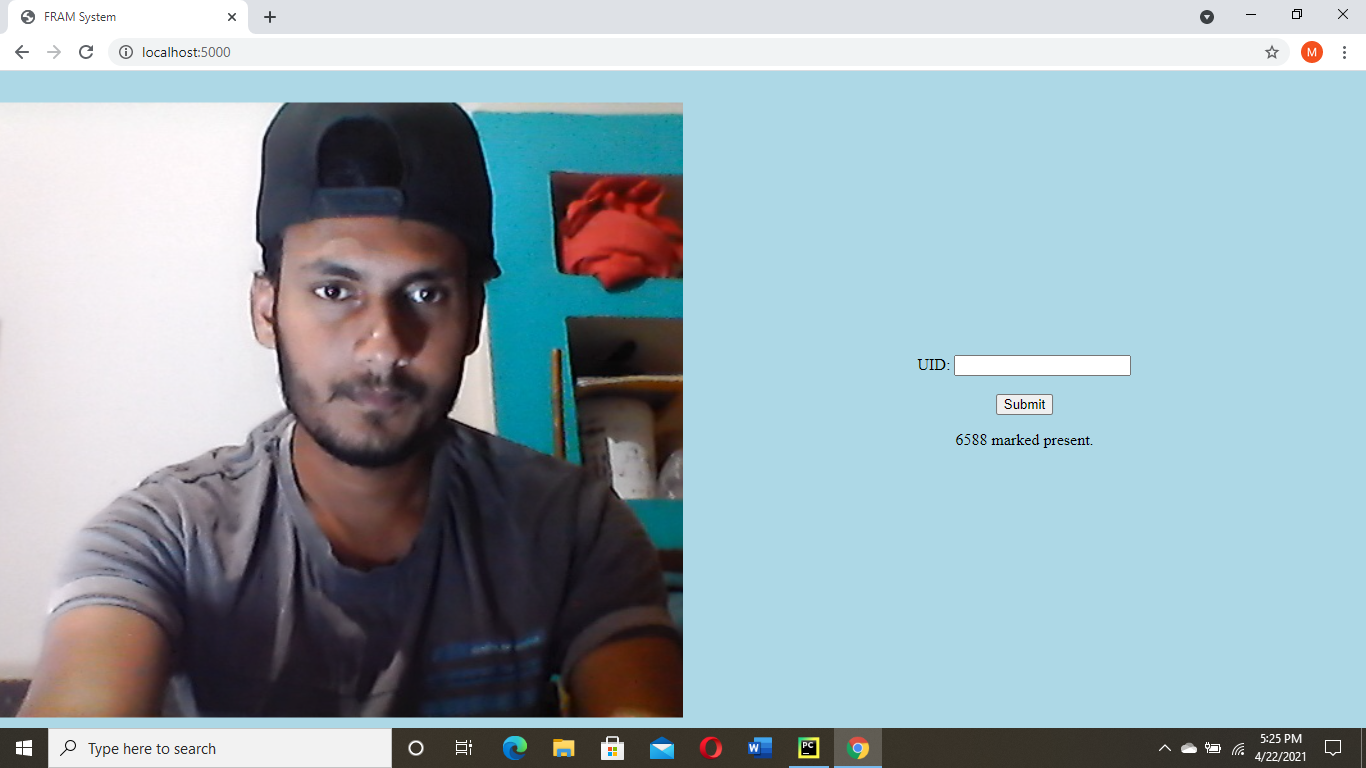
The whole page is divided into two parts, the left half shows the real time face feed directly from the front camera and the other half takes the input from the student (read student\_id) as a verification. On submission the model evaluates the person on the camera and returns the value like shown below and marks the attendance into a csv file.

**Student data and Train image,** The following image is stored on the uid ‘6588’ and the remaining data is shown as follow.

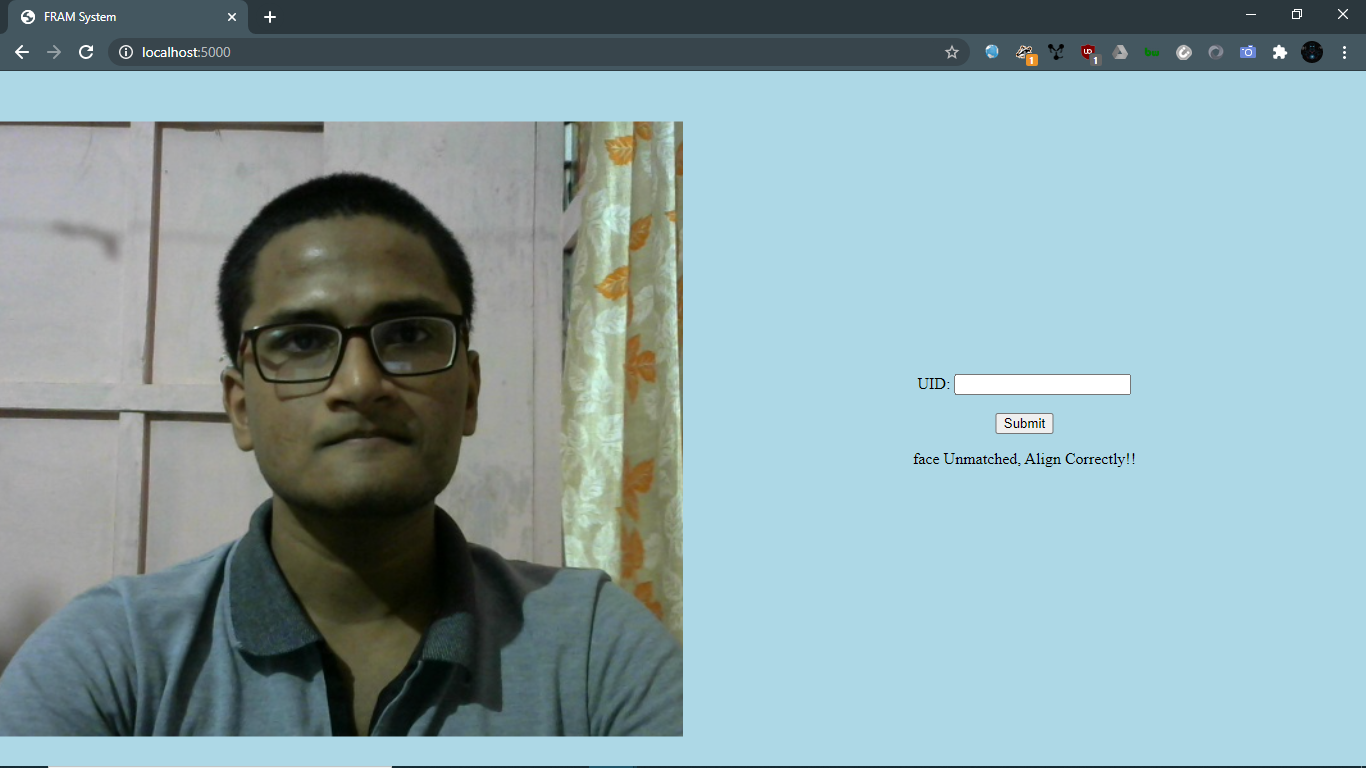




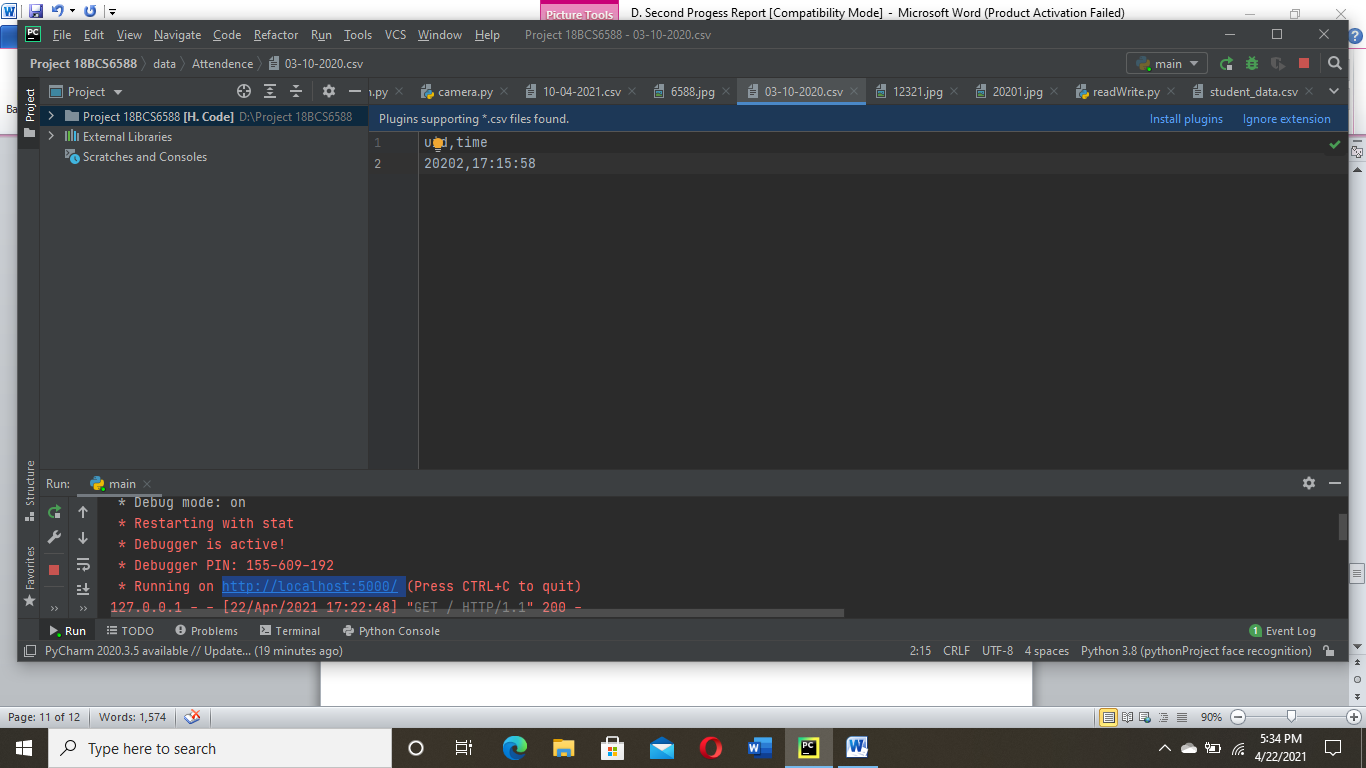
**If Student matches,** In the previous image template the next step would be as follows.



**If Student doesn’t match**



**Csv file after marking**



**Output Comparison & Validation:**

In the table below we will compare three most commonly used face recognition methods,

|  |  |  |  |
| --- | --- | --- | --- |
|  | **LBP** | **HAAR** | **HOG** |
| **TPR%** | 60.37 | 78.23 | 92.68 |
| **FNR%** | 39.65 | 21.76 | 7.31 |

The above table is based on a research paper with DOI: [10.1109/SSD.2019.8893214](https://doi.org/10.1109/SSD.2019.8893214) here we compare LBP (Local Binary Pattern), HAAR (Haar-like cascade), HOG (Histogram of Oriented Gradients) base on two factors TPR (True Positive Rate) and FNR (False Negative Rate) and from the table we can easily say that HOG method way more accurate than others.

The model is working with high Accuracy during optimal and medium light conditions and in different angles. While, during minimal light conditions the model is a bit struggling.

**References:**

* <https://medium.com/analytics-vidhya/a-take-on-h-o-g-feature-descriptor-e839ebba1e52>
* <https://medium.com/@ageitgey/machine-learning-is-fun-part-4-modern-face-recognition-with-deep-learning-c3cffc121d78>
* <https://www.researchgate.net/publication/337590875_Face_Recognition_based_smart_attendance_system_using_IOT>