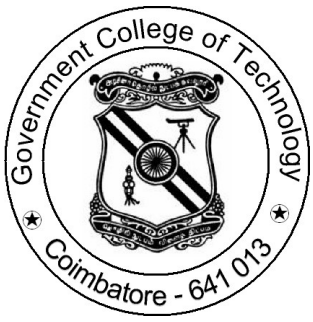


ATTENDANCE MANAGEMENT SYSTEM USING FACE RECOGNITION

PROJECT WORK

PROJECT SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE AWARD OF THE DEGREE OF **BACHELOR OF TECHNOLOGY** IN **INFORMATION TECHNOLOGY** OF THE ANNA UNIVERSITY

2021

Submitted by

ARUN I

(1718105)

RATHESH P

(1718131)

SANJAI M

(1718134)

Under the Guidance of

Prof. T. Suguna M.Tech

DEPARTMENT OF INFORMATION TECHNOLOGY

GOVERNMENT COLLEGE OF TECHNOLOGY

(An Autonomous Institution affiliated to Anna University)

COIMBATORE - 641 013

DEPARTMENT OF INFORMATION TECHNOLGOY

GOVERNMENT COLLEGE OF TECHNOLOGY

(An Autonomous Institution affiliated to Anna University)

COIMBATORE - 641 013

PROJECT WORK

AUGUST 2021

This is to certify that this project work entitled

ATTENDANCE MANAGEMENT SYSTEM USING FACE RECOGNITION

is the bonafide record of project work done by

ARUN I

(1718105)

RATHESH P

(1718131)

SANJAI M

(1718134)

of B.Tech. (Information Technology) during the year 2020 - 2021

Project Guide	Head of the Department
Prof. T. Suguna M.Tech.,	**Dr. S.Rathi M.E., Ph.D**

Submitted for the Project Viva-Voce examination held on _____

Internal Examiner	External Examiner

ACKNOWLEDGEMENT

Great achievements are not possible without standing on the shoulders of giants. Without the active involvement of the following experts this project would not have been a reality.

We express our sincere gratitude to **Dr.P.Thamarai M.E, Ph.D.,** Principal, Government College of Technology, Coimbatore for providing us all facilities that we needed for the completion of this project.

We whole-heartedly express our thankfulness and gratitude to **Dr.S.Rathi M.E., Ph.D.,** Associate Professor and Head of the Department of Information Technology, Government College of Technology, for helping us to successfully carry out this project.

Our thankfulness and gratitude to our respectable project guide **Mrs.T.Suguna M.Tech.,** Assistant Professor, who has been of immense help through the various phases of the project. With her potent ideas and excellent guidance, we were able to comprehend the essential aspects involved.

We would like to thank our faculty advisor **Mrs.R.Malavika M.Tech.,** Assistant Professor for her continuous support and encouragement throughout this project.

We extend our sincere thanks to the staff members of Information Technology department, **Mrs.R.Devi M.Tech.,** Assistant Professor, **Mrs.C.Aswini M.Tech.,** Assistant Professor **Thiru.S.Gladson Oliver M.Tech.,** Assistant Professor, **Ms.M.Jeyanthi M.Tech.,** Assistant Professor, **Thiru.M.Gowri Shankar M.E.,** Assistant Professor, **Mrs.R.Malavika M.Tech.,** Assistant Professor, **Dr.M.Blessy Queen Mary M.E.,Ph.D.,** Assistant Professor, for rendering their help for the completion of this project. We also thank all our friends for their cooperation and suggestions towards the successful completion of this project.

SYNOPSIS

Face Recognition is a technology capable of matching a human face from a digital image or a video frame against a database of a face works by pinpointing and measuring facial features from a given image. Face Recognition techniques authenticate users through ID verification services. These techniques are used for identifying people in photos, video, or in real-time etc. System generates the attendance of the student on the basis of presence in class. In the existing attendance management system we have to mark attendance manually and it takes a lot of time. The model detects the person by analyzing the image and provides the exact result after the extraction of the facial features. This technique ensures feasibility and provides a better solution for real time problems. The proposed method is based on capturing and detecting images using face recognition methodology.

CONTENTS

CHAPTER NO	TITLE	PAGE NO
	BONAFIDE CERTIFICATE	i
	ACKNOWLEDGEMENT	iii
	SYNOPSIS	iv
	TABLE OF CONTENTS	v
	LIST OF FIGURES	ix
	LIST OF ABBREVATIONS	x
1	**INTRODUCTION**	**1-3**
	1.1 DESCRIPTION	1
	1.2 EXISTING SYSTEM	1
	1.3 PROBLEM DEFINITION	2
	1.4 PROPOSED SYSTEM	2
	1.5 ORGANIZATION OF PROJECT	3
2	**LITERATURE REVIEW**	**4-8**
	2.1 AUTOMATIC ATTENDANCE MANAGEMENT SYSTEM USING FACE DETECTION	4
	2.1.1 DESCRIPTION	
	2.1.2 MERIT	
	2.1.3 DEMERIT	
	2.2 CLASS ATTENDANCE MANAGEMENT SYSTEM USING FACE RECOGNITION	4

2.2.1 DESCRIPTION

2.2.2 MERIT

2.2.3 DEMERIT

2.3 AUTOMATED ATTENDANCE SYSTEM USING IMAGE PROCESSING 5

2.3.1 DESCRIPTION

2.3.2 MERIT

2.3.3 DEMERIT

2.4 REAL TIME ATTENDANCE USING FACE RECOGNITION TECHNIQUE 6

2.4.1 DESCRIPTION

2.4.2 MERIT

2.4.3 DEMERIT

2.5 REAL TIME SMART ATTENDANCE SYSTEM USING FACE RECOGNITION TECHNIQUES 6

2.5.1 DESCRIPTION

2.5.2 MERIT

2.5.3 DEMERIT

2.6 ATTENDANCE MANAGEMENT SYSTEM USING FACE RECOGNITION 7

2.6.1 DESCRIPTION

2.6.2 MERIT

2.6.3 DEMERIT

2.7 STUDENT ATTENDANCE SYSTEM USING FACE RECOGNITION 8

2.7.1 DESCRIPTION

2.7.2 MERIT

2.7.3 DEMERIT

3 SYSTEM SPECIFICATION 9-10

3.1 SYSTEM REQUIREMENTS

3.1.1 HARDWARE REQUIREMENT 9

3.1.2 SOFTWARE REQUIREMENT 9

3.2 SOFTWARE DESCRIPTION 9

3.2.1 ABOUT PYTHON 9

3.2.2 CHARACTERISTICS OF PYTHON 10

3.2.3 ABOUT FLASK 10

3.2.4 CHARACTERISTICS OF FLASK 10

4 PROJECT DESIGN 11-12

4.1 MODULE DESCRIPTION OF ADMIN PHASE 11

4.2 MODULE DESCRIPTION OF TEACHERS PHASE 12

5 IMPLEMENTATION AND RESULT 13-33

5.1 IMPLEMENTATION

5.1.1 ADMIN – APP.PY 13

5.1.1.1 PACKAGES 13

5.1.1.2 AUTHENTICATION SYSTEM 14

5.1.1.3 TRAINING PHASE 17

5.1.1.4 MODEL GENERATION 18

5.1.2 TEACHER – APP.PY 18

5.1.2.1 PACKAGES 18

5.1.2.2 IMAGE CAPTURING 19

5.1.2.3 IMAGE DETECTION 20

5.1.2.4 ATTENDANCE VIEW 23

5.1.2.5 MAIL CONFIGURATION 23

5.1.2.6 MAIL ALERT 24

5.1.2.7 ATTENDANCE UPDATION 24

5.1.2.8 CONSOLIDATED REPORT GENERATION 25

5.2 SAMPLE OUTPUT

5.2.1 TEST IMAGES 27

5.1.3 ADMIN PANEL 28

5.1.4 ADMIN DASHBOARD 28

5.1.5 STUDENT ADDITION 29

5.1.6 FACULTY LOGIN 29

5.1.7 FACULTY DASHBOARD 30

5.1.8 TAKE ATTENDANCE 31

5.1.9 REPORT PANEL 32

5.1.10 ATTENDANCE VIEW 32

5.1.11 REPORT VIEW 33

6 CONCLUSION 34

6.1 CONCLUSION 34

7 REFERENCES 36-37

LIST OF FIGURES

FIGURE NO	TITLE	PAGE NO
4.1	ADMIN PHASE	11
4.2	TEACHER'S PHASE	12
5.2.1	TEST IMAGES	27
5.2.2	ADMIN PANEL	28
5.2.3	ADMIN DASHBOARD	28
5.2.4	STUDENT ADDITION	29
5.2.5	FACULTY LOGIN	29
5.2.6	FACULTY DASHBOARD	30
5.2.7	TAKE ATTENDANCE	31
5.2.8	REPORT PANEL	32
5.2.9	ATTENDANCE VIEW	32
5.2.10	REPORT VIEW	33

LIST OF ABBREVIATIONS

LBP	LOCAL BINARY PATTERN
LBPH	LOCAL BINARY PATTERN HISTOGRAM
PCA	PRINCIPAL COMPONENT ANALYSIS
CNN	CONVOLUTIONAL NEURAL NETWORK
KNN	K-NEAREST NEIGHBOUR
SVM	SUPPORT VECTOR MACHINE

CHAPTER 1
INTRODUCTION

DESCRIPTION

Attendance Management System is an application developed for daily evaluation of student's attendance. It is facilitated to access the information of attendance of a particular Student in a particular subject of study. The information is sorted by the faculties and admin, as provided by the system for a particular student throughout a complete semester. This system will also enable the evaluation of student regular presence in various subjects.

Attendance being a very necessary side of administration may normally become an arduous, redundant activity, pushing itself to inaccuracies. The traditional approach of making roll calls proves itself to be a statute of limitations as it is very difficult to call names and maintain its record especially when the ratio of students is high. Some organizations use document-oriented approach and others have implemented these digital methods such as biometric fingerprinting techniques and card swapping techniques. However, these methods prove to be a statute of limitations as it subjects students to wait in a time-consuming queue. Post completion of face recognition, the system generates the name and identification number of the students who are present and identified in the image. Then attendance is marked in front of the student names in the excel format with respective date and subject of the lecture.

EXISTING SYSTEM

In the existing methods, some of the disadvantages are listed.

- Staff has to take attendance manually.
- If the student fails to bring his id card then he will not be able get attendance.
- It is a tedious job to maintain records.
- It is difficult to generate consolidated reports.
- Sheets may get lost.

PROBLEM DEFINITION

- This project attempts to make use of dlib library to perform the face recognition for the purpose of marking attendance.
- Though many traditional methods are existed, those are vulnerable but our system will minimize the proxies.
- Face recognition proved to be a productive method for taking attendance in an institution with high precision and less computational complexity.

PROPOSED SYSTEM

- To overcome the drawbacks of the existing system proposed has been evolved.
- It aims to reduce the paperwork and save time for generating the accurate results from the attendance system.
- The system provides the best user interface and efficient reports are generated.

ORGANIZATION OF THE PROJECT

- Literature reviews of already existing proposals are discussed in chapter 2.
- Chapter 3 has system specification which tells about the software and hardware requirements.
- Chapter 4 discusses the overall project and design which tells the brief description of each of the modulus in this project.
- Chapter 5 has the implementation and experimental result of the project.
- Chapter 6 deals with the conclusion and future work.
- Finally chapter 7 deals with the references.

CHAPTER 2

LITERATURE REVIEW

2.1 AUTOMATIC ATTENDANCE MANAGEMENT SYSTEM USING FACE DETECTION [2016]

2.1.1 DESCRIPTION

This paper is about biometric attendance management. In this method camera is fixed in the classroom. After fixing the camera the image will be captured. During the training phase all students' images are trained. In the captured image the faces are recognized and faces get cropped from the captured image. During the attendance marking phase the recognized faces are checked against the database image. And if the image is matched with the database image the attendance gets marked. If the image is not found it will ask the admin to add the student image in the database. If the attendance is marked as absent it will send an alert to the concerned student parent. For the face recognition it uses a method called Eigen face. Eigen face is one of the fastest approaches. The Eigen face method decomposes images into small features. Discrete images are mapped using linear values of Eigen face. Finally all the faces are recognized using the best Eigen face value.

2.1.2 MERIT

- Hardware installation is easy.

2.1.3 DEMERIT

- Accuracy is low and face recognition is difficult.

2.2 CLASS ATTENDANCE MANAGEMENT SYSTEM USING FACE RECOGNITION [2018]

2.2.1 DESCRIPTION

This paper proposes a comprehensive embedded class attendance system with controlling the door access. The proposed scheme is based on face recognition. By using Local Binary Pattern algorithm (LBP) the face gets recognized. After capturing image it will be passed to the raspberry pi which handles the implementation of face recognition algorithm. If the student image was matched the door will be opened with the help of servo motor. In addition, the attendance result will be stored in the MySQL database. The experimental results show that higher accuracy can be achieved with our proposed one. The numerical results are provided to show the performances of the proposed scheme in different cases of face recognition.

2.2.2 MERIT

- Easily accessible.

2.2.3 DEMERIT

- Less sensitivity and not effective in darkness.

2.3 AUTOMATED ATTENDANCE SYSTEM USING IMAGE PROCESSING [2018]

2.3.1 DESCRIPTION

This paper proposes an idea of using image detection and recognition which can automatically handle the attendance system. In this paper, Viola Jones algorithm used for face detection and Fisher Face Algorithm are used for face recognition. At first video segments are captured from the classroom. Then, the face was detected by preprocessing the video input .Face cropping has been done At last; the cropped image would be compared with the database. If the cropped face matches with the database it marks the attendance and if the image is not found it will ask the user to register.

2.3.2 MERIT

- Maintains attendance in high efficient manner

2.3.3 DEMERIT

- Accuracy rate is less than 50%

2.4 REAL TIME SMART ATTENDANCE SYSTEM USING FACE RECOGNITION TECHNIQUES [2019]

2.4.1 DESCRIPTION

Generally attendance system is executed with the help of biometrics. Face recognition is one of the best methods to improve this system. This paper proposes a method to automate the attendance management system by making use of recognition techniques like Eigen face values, Principal Component Analysis(PCA),Convolution Neural Network(CNN).Firstly student has to enroll and details will be stored in the database and the camera will be fixed outside the classroom. If the face is not present in the database it prompts the student to register. In face recognition module the images are recovered from the camera inside the classroom. It will mark present for the students who are present in the class. Marked attendance is stored in the database for the future purpose.

2.4.2 MERIT

- Proxy less with high accuracy

2.4.3 DEMERIT

- Less efficient

2.5 REAL TIME ATTENDANCE USING FACE RECOGINITION TECHNIQUE [2020]

2.5.1 DESCRIPTION

This paper is about integrating the face recognition technique with Open Source Computer Vision (OpenCV) for attendance management system. It will

facilitate the attendance automation process and enable staff to enquire student details based on the check in and checkout time. Firstly the camera has been installed in the classrooms. After that image will be captured and the captured image will be passed to the face recognition module. The recognized images are checked with images of the students which is stored in a database with individual id .if the image is not found in the database by performing haar cascade method student can add their image into the database. Student information and attendance are stored in Excel Sheet.

2.5.2 MERIT

- Accuracy high in smaller area

2.5.3 DEMERIT

- Detection rate is low in larger area

2.6 ATTENDANCE MANAGEMENT SYSTEM USING FACE RECOGNITION [2020]

2.6.1 DESCRIPTION

This paper describes the face recognition technique used for the attendance management system. It uses an algorithm called cascade classifier and Local Binary Pattern Histogram (LBPH) for face recognition. In this application students have to register by entering the ID and Name registered with Face recognition. This data will be stored in database. Local Binary Pattern is a basic algorithm used to detect face from front side. Haar cascade is based on Haar wavelet technique to analyze the pixels from the captured image. The recognized images are checked with images of the students which is stored in a database with individual id .if the image is not found in the database by performing haar cascade method student can add their image into the database. Student information and attendance are stored and it will be available for the respective staffs.

2.6.2 MERIT

- Robust Against monotonic grey image

2.6.3 DEMERIT

- Take more than 5 secs to recognize.

2.7 STUDENT ATTENDANCE SYSTEM USING FACE RECOGNITION [2020]

2.7.1 DESCRIPTION

This paper describes the face recognition technique used for the attendance management system. It uses an algorithm called haar cascade classifier, K-Nearest Neighbor (KNN), Convolution Neural Network (CNN), Support Vector Machine (SVM) and Local Binary Pattern Histogram (LBPH) for face recognition. In this application attendance report will generated and stored in excel format .This system is tested against various conditions. The system developed need less installation .The recognized images are checked with images of the students which is stored in a database .if the image is not found in the database student can add their image into the database. Student information and attendance are stored and it will be available in data base.

2.7.2 MERIT

- Cost efficient

2.7.3 DEMERIT

- Less computational Complexity

CHAPTER 3

SYSTEM SPECIFICATION

3.1 SYSTEM REQUIREMENTS

3.1.1 HARDWARE REQUIREMENTS

- System : Pentium IV 2.4 GHz.
- Hard Disk : 1 TB
- RAM : 4 GB

3.1.2 SOFTWARE REQUIREMENTS

- Operating system : Windows 10
- Coding Language : PYTHON
- Tool : PYTHON 3.9, FLASK

3.2 SOFTWARE DESCRIPTION

3.2.1 ABOUT PYTHON

PYTHON is an interpreted, object-oriented, high level programming language with dynamic semantics. It supports modules and package which encourages code reuse.Typical uses include:

- User-friendly Data Structures
- Software development
- Writing system scripts
- Data analysis, exploration, and visualization
- Scientific and engineering graphics

PYTHON is often used as a scripting language for Web applications. It means it can automate specific series of tasks in an efficient way. It works on different platforms like Windows, Mac, Linux, etc.

3.2.2 CHARACTERISTICS OF PYTHON

- Python is a very easy to use developer-friendly language.
- Open Source Programming language.
- Support a wide area of GUI.
- Easily Portable language.
- Supports other Programming Languages like C,C++,Java,etc.
- Tools for building applications with custom user interface.

3.2.3 ABOUT FLASK

FLASK is a micro web framework written in python. It does not require specific tools or libraries. It allows us to build web applications. It will not force any dependency. It has features like url routing, template engine.

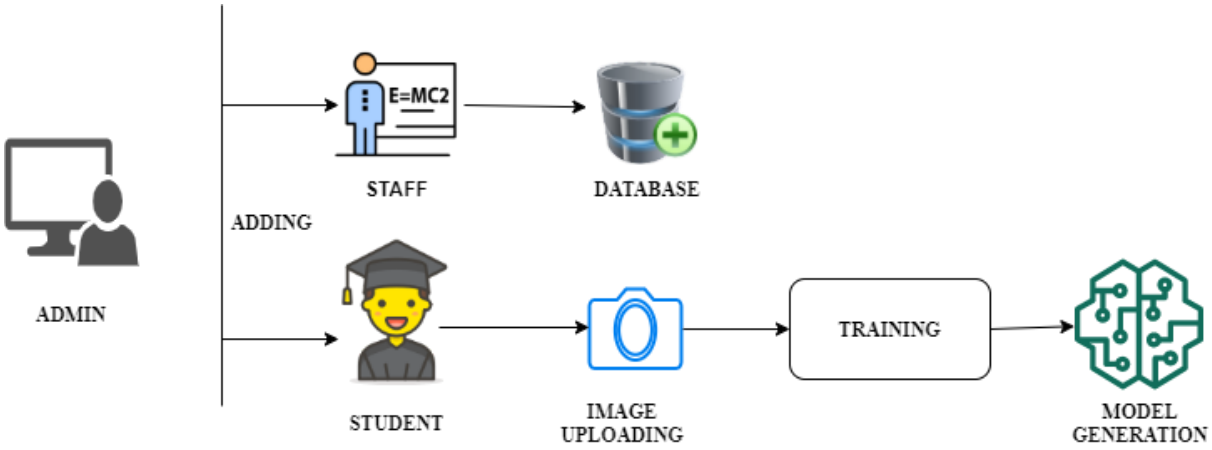
3.2.4 CHARACTERISTICS OF FLASK

- Development server and debugger.
- Integrated support for unit testing.
- RESTful request dispatching.
- Uses Jinja templating.
- Support for secure cookies (client side sessions).

CHAPTER 4

PROJECT DESIGN

4.1 MODULE DESCRIPTION OF ADMIN PHASE



4.1 FIGURE : ADMIN PHASE

Input : Student image

Output : Model of the student

Step 1 : Addition of staff and students.

Step 2 : Storing the details of student and staff into the database.

Step 3 : Allowing students to capture the image.

Step 4 : Uploading the captured image of the student.

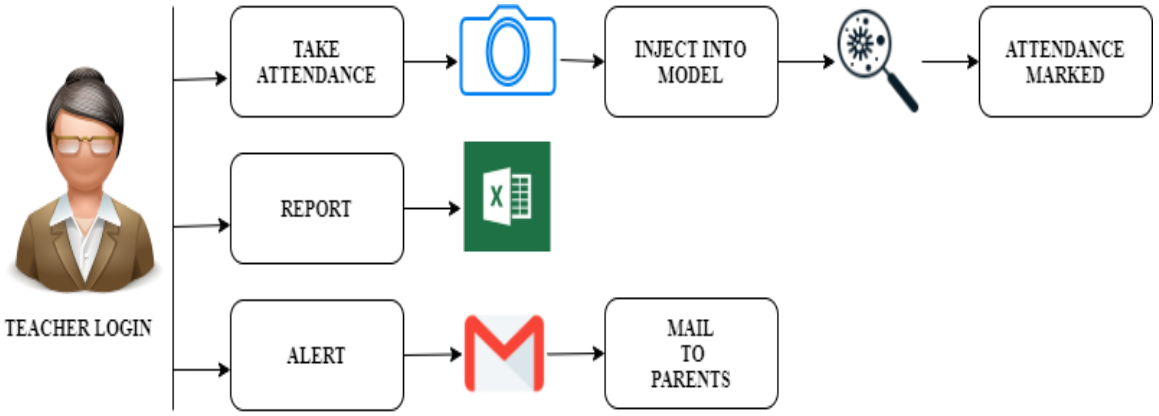
Step 5 : Using ML algorithm the training has been done for the uploaded image of a particular student.

Step 6 : Model is generated for each student.

Step 7 : Generated models are stored in separate folders.

Step 8 : Save the folder name with the ID number of the student.

4.2 MODULE DESCRIPTION OF TEACHER PHASE



4.2 FIGURE : TEACHER PHASE

Input : Student image

Output : Attendance report.

Step 1 : Teacher allows students to capture images of the student.

Step 2 : Captured image is injected into the model.

Step 3 : The image is compared with the generated model.

Step 4 : By entering the corresponding roll number and subject code the attendance will be marked.

Step 5 : If the model is not present it will prompt the staff to add the student.

Step 6 : Hence the attendance is stored in the Excel sheet.

Step 7 : In the report generation tab the staff can generate the attendance based on hours and particular week and consolidated report.

Step 8 : Staff can send mail to the student parents about the daily presence of student in the classroom.

CHAPTER 5

IMPLEMENTATION AND RESULTS

5.1 IMPLEMENTATION

Implementation is the most crucial stage in achieving a successful system and giving the users confidence that the new system is effective and workable. Implementation of this project refers to the installation of the packages in its real environment to the full satisfaction of the users and operations of the system. Testing is done individually at the time of development using the data and verification is done the way specified in the program specification. In short, implementation constitutes all activities that are required to put an already tested and completed package into operation. The success of any information system lies in its successful implementation. System implementation is the stage in the project where the theoretical design is turned into a working system. The most critical stage is achieving a successful system and in giving confidence on the new system for the user that it will work efficiently and effectively. The existing system was a long time process.

5.1.1 ADMIN - app.py

5.1.1.1 PACKAGES

```
       from flask import
Flask,render_template,request,redirect,url_for,session
       import MySQLdb
       import os

       #import for face recognition
       from math import sqrt
       from sklearn import neighbors
       from os import listdir
       from os.path import isdir, join, isfile, splitext
       import pickle
       from PIL import Image, ImageFont, ImageDraw, ImageEnhance
       import face_recognition
       from face_recognition import face_locations
       from face_recognition.face_recognition_cli import
image_files_in_folder
```

5.1.1.2 AUTHENTICATION SYSTEM

```
app = Flask(__name__)

APP_ROOT = os.path.dirname(os.path.abspath(__file__))
print(APP_ROOT)

conn                                                                    =
MySQLdb.connect(host="localhost",user="root",password="Arasii@5670",db="lo
gin_info")

@app.route('/')
def index():
    return render_template("index.html",title="Admin Login")
@app.after_request
def set_response_headers(response):
    response.headers['Cache-Control']   =   'no-cache,   no-store,   must-
revalidate'
    response.headers['Pragma'] = 'no-cache'
    response.headers['Expires'] = '0'
    return response

@app.route('/login',methods=['POST'])
def login():
    user = str(request.form["user"])
    paswd = str(request.form["password"])
    cursor = conn.cursor()
    result  =  cursor.execute("SELECT  *  from  admin_login  where  binary
username=%s and binary password=%s",[user,paswd])
    if(result==1):
        return render_template("task.html")
    else:
        return  render_template("index.html",title="Admin  Login",msg="The
username or password is incorrect")


@app.route('/register_teacher',methods=['POST'])
def register_teacher():
    return render_template("signup.html",title="SignUp")
@app.after_request
def set_response_headers(response):
    response.headers['Cache-Control']   =   'no-cache,   no-store,   must-
revalidate'
```

```
    response.headers['Pragma'] = 'no-cache'
    response.headers['Expires'] = '0'
    return response


@app.route('/student',methods=['POST'])
def file_upload():
    return render_template("upload.html")


@app.route('/signup_teacher',methods=['POST'])
def signup():
    user = str(request.form["user"])
    paswd = str(request.form["password"])
    email = str(request.form["email"])
    cursor = conn.cursor()
    result = cursor.execute("SELECT * from teacher_login where binary
username=%s",[user])
    print (result)
    if(result == 1):
        return
render_template("signup.html",title="SignUp",uname=user,msg="already
present")
    cursor.execute("INSERT INTO teacher_login (username,password,email)
VALUES(%s, %s, %s)",(user,paswd,email))
    conn.commit()
    return render_template("signup.html",title="SignUp",msg="successfully
signup",uname=user)


@app.route('/signup_student',methods=['POST'])
def signup_student():
    user = str(request.form["student_name"])
    email = str(request.form["student_email"])
    roll_id = str(request.form["roll_id"])
    email1 = str(request.form["parent_email"])
    cursor = conn.cursor()
    result = cursor.execute("SELECT * from student_login where binary
roll_id=%s",[roll_id])
    print (result)
    if(result == 1):
        return render_template("upload.html",uname=user,msg=" already
present")
    cursor.execute("INSERT INTO student_login
```

```
(username,student_email,parent_email,roll_id)    VALUES(%s,    %s,    %s,
%s)",(user,email,email1,roll_id))
    conn.commit()
    return   render_template("upload.html",uname=user,msg="   successfully
signup")


@app.route("/upload", methods=['POST'])
def upload():
    target = os.path.join(APP_ROOT,"train/")
    if not os.path.isdir(target):
        os.mkdir(target)
    classfolder = str(request.form['class_folder'])
    session['classfolder'] = classfolder
    target1 = os.path.join(target,str(request.form["class_folder"])+"/")
    session['target1']=target1
    print(target1)
    model = os.path.join(APP_ROOT,"model/")
    if not os.path.isdir(model):
        os.mkdir(model)
    classname = str(request.form['class_folder']+"/")
    model = os.path.join(model,classname)
    if not os.path.isdir(model):
        os.mkdir(model)
    session['model']=model
    session['classname'] = classname
    if not os.path.isdir(target1):
        os.mkdir(target1)
    id_folder = str(request.form["id_folder"])
    session['id_folder']= id_folder
    target2 = os.path.join(target1,id_folder+"/")
    if not os.path.isdir(target2):
        os.mkdir(target2)
    target3 = os.path.join(target2,id_folder+"/")
    if not os.path.isdir(target3):
        os.mkdir(target3)
    for file in request.files.getlist("file"):
        print(file)
        filename = file.filename
        destination = "/".join([target3,filename])
        print(destination)
        file.save(destination)
    return call_train()
```

5.1.1.3 TRAINING PHASE

This function is responsible for training student data through face recognition and storing it in a database. The whole training process involves preprocessing the image and the model generation.

```
def train(train_dir, model_save_path = "", n_neighbors = None,
knn_algo = 'ball_tree', verbose=True):
    id_folder=str(session.get('id_folder'))
    X = []
    y = []
    z = 0
    for class_dir in listdir(train_dir):
        if not isdir(join(train_dir, class_dir)):
            continue
        for img_path in image_files_in_folder(join(train_dir,
class_dir)):
            image = face_recognition.load_image_file(img_path)
            faces_bboxes = face_locations(image)
            if len(faces_bboxes) != 1:
                if verbose:
                    print("image {} not fit for training:
{}".format(img_path, "didn't find a face" if len(faces_bboxes) < 1
else "found more than one face"))
                    os.remove(img_path)
                    z = z + 1
                continue
            X.append(face_recognition.face_encodings(image,
known_face_locations=faces_bboxes)[0])
            y.append(class_dir)
    print(listdir(train_dir+"/"+id_folder))
    train_dir_f = listdir(train_dir+"/"+id_folder)
    for i in range(len(train_dir_f)):
        if(train_dir_f[i].startswith('.')):
            os.remove(train_dir+"/"+id_folder+"/"+train_dir_f[i])

    print(listdir(train_dir+"/"+id_folder))

    if(listdir(train_dir+"/"+id_folder)==[]):
        return render_template("upload.html",msg1="training data
empty, upload again")
    elif(z >= 1):
        return render_template("upload.html",msg1="Data trained for
"+id_folder+", But one of the image not fit for training")
```

```
    if n_neighbors is None:
        n_neighbors = int(round(sqrt(len(X))))
        if verbose:
            print("Choose n_neighbors automatically as:", n_neighbors)

    knn_clf = neighbors.KNeighborsClassifier(n_neighbors=n_neighbors,
algorithm=knn_algo, weights='distance')
    knn_clf.fit(X, y)

    if model_save_path != "":
        with open(model_save_path, 'wb') as f:
            pickle.dump(knn_clf, f)

    return render_template("upload.html",msg1="Data trained for "+
id_folder)
```

5.1.1.4 MODEL GENERATION

```
def call_train():
    id_folder = str(session.get('id_folder'))
    model=str(session.get('model'))
    if not os.path.isdir(model + id_folder):
        os.mkdir(model + id_folder)
    model = model + id_folder + "/"
    model = model + "model"
    target1=str(session.get('target1'))
    print(id_folder)
    print (target1)
    target1 = target1 +id_folder
    print(target1)
    print(model)
    return train(train_dir=target1,model_save_path=model)
```

5.1.2 TEACHERS: APP.PY

5.1.2.1 PACKAGES

```
from flask import
Flask,render_template,request,redirect,url_for,session
from flask_bootstrap import Bootstrap
import MySQLdb
```

```
import os
from math import sqrt
from sklearn import neighbors
from os import listdir
from os.path import isdir, join, isfile, splitext
import shutil
import pickle
from PIL import Image, ImageFont, ImageDraw, ImageEnhance
import face_recognition
from face_recognition import face_locations
from face_recognition.face_recognition_cli import
image_files_in_folder
from datetime import datetime,timedelta
from pytz import timezone
import xlsxwriter
import pandas as pd
from glob import glob
from flask_mail import Mail, Message
from io import BytesIO
import base64
import lable_image
```

5.1.2.2 IMAGE CAPTURING

```
def upload():
    if not os.path.isfile(APP_ROOT+"/image.jpeg"):
        return render_template("upload.html",msg="spoof detected")
    id_folder = str(request.form['id_folder'])
    session['id_folder']= id_folder
    target = os.path.join(APP_ROOT,"test/")
    if not os.path.isdir(target):
        os.mkdir(target)
    target1 =
os.path.join(target,str(request.form["folder_name"])+"/")
    test_append = str(request.form["folder_name"])
    session['test_append']= test_append
    print(target1)
    if not os.path.isdir(target1):
        os.mkdir(target1)
    shutil.copyfile(APP_ROOT+"/"+"image.jpeg",target1+"image.jpeg")
    destination = APP_ROOT + "/" + "test/" + test_append + "/" +
"image.jpeg"
```

```
    session['destination'] = destination
    teacher_name = str(session.get('user'))
    session['teacher_name'] = teacher_name
    #return render_template("upload.html",msg="uploaded successfully")
    return match()
```

5.1.2.3 IMAGE DETECTION

```
def match():
    destination = str(session.get('destination'))
    print(destination)
    if os.path.isfile(destination):
        test_append = str(session.get('test_append'))
        session['test_append'] = test_append
        id_folder = str(session.get('id_folder'))

        train_dir = APP_ROOT1[0]+"admin_site/train/"+ test_append
        try:
            model = APP_ROOT1[0]+"admin_site/model/"+test_append+"/" +
id_folder + "/" +"model"
            print(model)
            return predict1(model)
        except FileNotFoundError:
            os.remove(APP_ROOT1[0]+"teachers_site/image.jpeg")
            return render_template("upload.html",msg="trained model
not present for " + test_append + ": "+id_folder)


def predict(X_img_path, knn_clf = None, model_save_path ="",
DIST_THRESH = .45):
    if knn_clf is None and model_save_path == "":
        raise Exception("must supply knn classifier either thourgh
knn_clf or model_save_path")

    if knn_clf is None:
        with open(model_save_path, 'rb') as f:
            knn_clf = pickle.load(f)

    X_img = face_recognition.load_image_file(X_img_path)
    X_faces_loc = face_locations(X_img)
    if len(X_faces_loc) == 0:
        return []
```

```
    faces_encodings = face_recognition.face_encodings(X_img,
known_face_locations=X_faces_loc)


    closest_distances = knn_clf.kneighbors(faces_encodings,
n_neighbors=1)

    is_recognized = [closest_distances[0][i][0] <= DIST_THRESH for i
in range(len(X_faces_loc))]


    return [(pred) if rec else ("unknown") for pred, rec in
zip(knn_clf.predict(faces_encodings), is_recognized)]

def predict1(model):
    test_append = str(session.get('test_append'))
    test_dir = APP_ROOT1[0]+"teachers_site/test/" + test_append
    f_preds = []
    for img_path in listdir(test_dir):
        preds = predict(join(test_dir, img_path)
,model_save_path=model)
        f_preds.append(preds)
        print(f_preds)
    print(len(preds))
    print(len(f_preds))
    for i in range(len(f_preds)):
        if(f_preds[i]==[]):
            os.remove(APP_ROOT1[0]+"teachers_site/image.jpeg")
            return render_template("upload.html",msg="upload again,
face not found")
        else:
            os.remove(APP_ROOT1[0]+"teachers_site/image.jpeg")
    excel = os.path.join(APP_ROOT,"excel/")
    if not os.path.isdir(excel):
        os.mkdir(excel)
    excel1 = os.path.join(excel,test_append)
    if not os.path.isdir(excel1):
        os.mkdir(excel1)
    teacher_name = str(session.get('teacher_name'))
    excel2 = os.path.join(excel1,teacher_name)
    if not os.path.isdir(excel2):
        os.mkdir(excel2)
    session['excel2'] = excel2
```

```
    excel3 = excel2+"/"+date+'.xlsx'
    if not os.path.isfile(excel3):
        workbook = xlsxwriter.Workbook(excel2+"/"+date+'.xlsx')
        worksheet = workbook.add_worksheet()
        worksheet.set_column(0,0,20)
        worksheet.write('A1','Roll Id')
        f_preds.sort()
        row = 1
        col = 0
        if f_preds[0][0] == 'unknown':
            return render_template("upload.html",msg= "Student Not
Matched")
        for i in range(len(f_preds)):
            for j in range(len(f_preds[i])):
                worksheet.write_string(row,col,f_preds[i][j])
                row += 1
        workbook.close()
        return render_template("upload.html",msg= f_preds[0][0] + "
present")
    else:
        df = pd.read_excel(excel2+"/"+date+'.xlsx')
        writer = pd.ExcelWriter(excel2 + "/" + date+'.xlsx')
        df.to_excel(writer,sheet_name="Sheet1",index=False)
        workbook  = writer.book
        worksheet = writer.sheets['Sheet1']
        rows=df.shape[0]
        worksheet.write_string(rows+1,0,f_preds[0][0])
        writer.save()
        df = pd.read_excel(excel2+"/"+date+'.xlsx')
        df.drop_duplicates(['Roll Id'],keep='first',inplace=True)
        # result = df.sort_values("Roll Id")
        writer = pd.ExcelWriter(excel2 + "/" + date+'.xlsx')
        df.to_excel(writer,'Sheet1',index=False)
        workbook = writer.book
        worksheet = writer.sheets['Sheet1']
        worksheet.set_column(0,0,20)
        writer.save()
        return render_template("upload.html",msg= f_preds[0][0] + "
present")
```

5.1.2.4 ATTENDANCE VIEW

```
def view():
    test_append = str(request.form['folder_name'])
    session['test_append']=test_append
    teacher_name = str(session.get('user'))
    excel_dir = APP_ROOT+"/excel/"+test_append+"/"+teacher_name+"/"
    excel_date = request.form['fname']
    time = request.form['ftime']
    time = time[:2]
    print(time)
    final_excel=glob(excel_dir + "/" + excel_date+ "@" + time
+"*.xlsx")[0]
    print(final_excel)

    df = pd.read_excel(final_excel)
    df.index += 1
    return
render_template("files.html",msg=final_excel,df=df,date=excel_date+"@"
+time+"hrs")
```

5.1.2.5 MAIL CONFIGURATION

```
app.config.update(
    DEBUG = True,
    #Email settings
    MAIL_SERVER = 'smtp.gmail.com',
    MAIL_PORT = 465,
    MAIL_USE_SSL = True,
    MAIL_USERNAME = 'college1118@gmail.com',
    MAIL_PASSWORD = 'yog12345',
    MAIL_DEFAULT_SENDER = 'college1118@gmail.com'
    )
mail = Mail(app)

# declaring timezone then creating custom date format

india = timezone('Asia/Kolkata')
date = str(datetime.now(india))[:10] + "@" +
str(datetime.now())[11:13] + "hrs"
```

5.1.2.6 MAIL ALERT

```
def send_mail():
    test_append = str(request.form['folder_name'])
    teacher_name = str(session.get('user'))
    excel_dir = APP_ROOT+"/excel/"+test_append+"/"+teacher_name+"/"
    excel_date = request.form['fname']
    time = request.form['ftime']
    time = time[:2]
    final_send = glob(excel_dir + "/" + excel_date+ "@" + time
+"*.xlsx")[0]
    print(final_send)
    df = pd.read_excel(final_send)
    roll_id = list(df['Roll Id'])
    print(type(roll_id))
    print(roll_id)
    cursor = conn.cursor()
    for i in range(len(roll_id)):
        cursor.execute("SELECT student_email,parent_email from
student_login where binary roll_id=%s",[roll_id[i]])
        email = list(cursor.fetchone())
        print(type(email[1]))
        print(email[0])
        print(email[1])
        msg = Message('Auto Generated',recipients=
[email[0],email[1]])
        msg.body = "Hi.. " + roll_id[i] + " is present for the lecture
of " + "Prof. " +str(teacher_name.split('.',1)[0]) + ", which is held
on " + excel_date + "@" + time + "hrs"
        msg.html = "Hi.. " + roll_id[i] + " is present for the lecture
of " + "Prof. " +str(teacher_name .split('.',1)[0])+ ", which is held
on " + excel_date + "@" + time + "hrs"
        mail.send(msg)
    return "<h1>mail sent<h1>"
```

5.1.2.7 ATTENDANCE UPDATION

```
def update():
    test_append = str(request.form['excel_folder'])
    print(test_append)
    teacher_name = str(session.get('user'))
    print(teacher_name)
```

```
    excel_dir = APP_ROOT + "/excel/" + test_append + "/" +
teacher_name + "/"
    print(excel_dir)
    for file in request.files.getlist("updated_excel"):
        print(file)
        filename = file.filename
        print(filename)
        destination = "/".join([excel_dir,filename])
        print(destination)
        file.save(destination)
    return render_template("excel.html",msg="updated successfully")
```

5.1.2.8 CONSOLIDATED REPORT GENERATION

```
def calculate():
    test_append = str(request.form['final_class'])
    print(test_append)
    teacher_name = str(session.get('user'))
    print(teacher_name)
    excel_root = APP_ROOT + "/excel/" + test_append + "/" +
teacher_name + "/"
    print(excel_root)
    excel_names = os.listdir(excel_root)
    print(excel_names)
    for i in range(len(excel_names)):
        if excel_names[i].startswith("."):
            os.remove(excel_root+excel_names[i])
        else:
            if os.path.isdir(excel_root+excel_names[i]):
                shutil.rmtree(excel_root+excel_names[i],
ignore_errors=False, onerror=None)
    excel_names = os.listdir(excel_root)

    if(excel_names==[]):
        return render_template("excel.html",msg1="No excel files
found")

    for i in range(len(excel_names)):
        excel_names[i] = excel_root + excel_names[i]
    print(type(excel_names))
    # read them in
    excels = [pd.ExcelFile(name) for name in excel_names]
```

```
    # turn them into dataframes
    frames = [x.parse(x.sheet_names[0], header=None,index_col=None)
for x in excels]
    # delete the first row for all frames except the first
    # i.e. remove the header row -- assumes it's the first
    frames[1:] = [df[1:] for df in frames[1:]]
    # concatenate them..
    combined = pd.concat(frames)
    if not os.path.isdir(excel_root+"final/"):
        os.mkdir(excel_root + "final/")
    final = excel_root + "final/"
    print(final)
    # write it out
    combined.to_excel(final+"final.xlsx", header=False, index=False)

    # below code is to find actual repetative blocks

    workbook = pd.ExcelFile(final+"final.xlsx")
    df = workbook.parse('Sheet1')
    sample_data = df['Roll Id'].tolist()
    print (sample_data)
    #a dict that will store the poll results
    results = {}
    for response in sample_data:
        results[response] = results.setdefault(response, 0) + 1
    finaldf = (pd.DataFrame(list(results.items()), columns=['Roll Id',
'Total presenty']))
    finaldf = finaldf.sort_values("Roll Id")
    print (finaldf)
    writer = pd.ExcelWriter(final+"final.xlsx")
    finaldf.to_excel(writer,'Sheet1',index=False)
    workbook  = writer.book
    worksheet = writer.sheets['Sheet1']
    worksheet.set_column(0,1,20)
    writer.save()
    final = final + "final.xlsx"
    session['final']=final
    final = final[91:]
    return viewfinal(final)
```

5.2 SAMPLE OUTPUT

Fig 5.2.1 TEST IMAGES

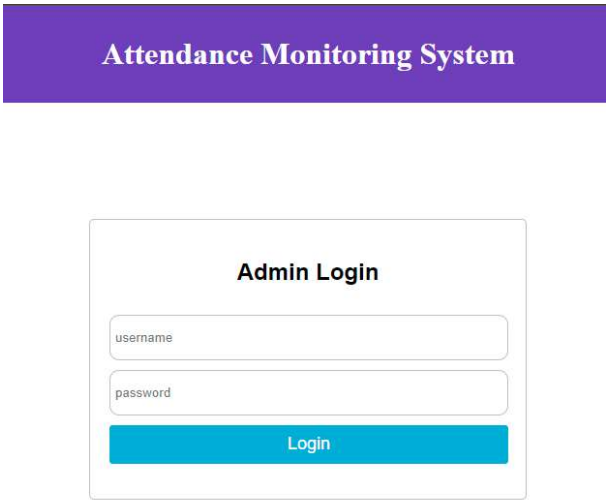


Fig 5.2.2 ADMIN PANEL

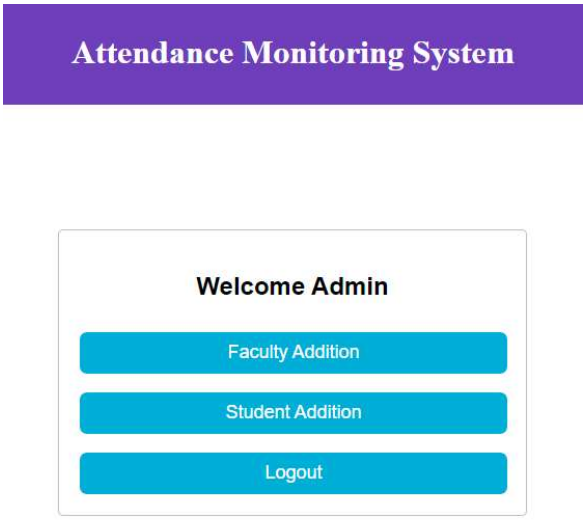


Fig 5.2.3 ADMIN DASHBOARD

Fig 5.2.4 STUDENT ADDITION

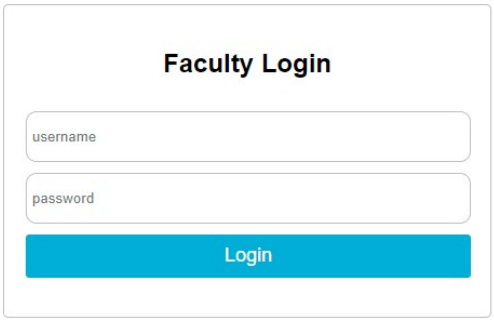


Fig 5.2.5 FACULTY LOGIN

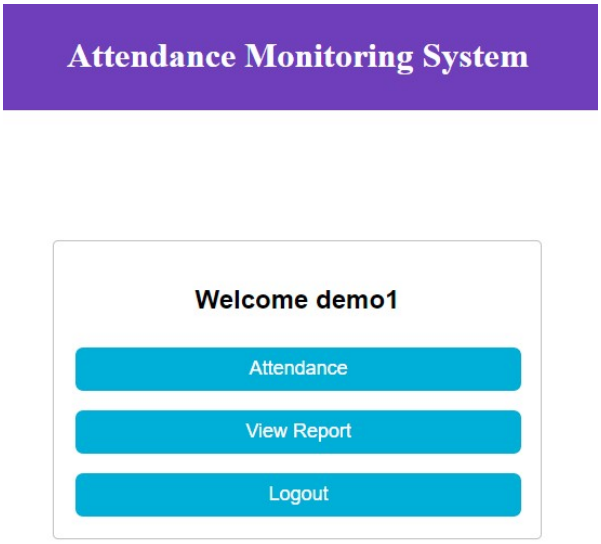
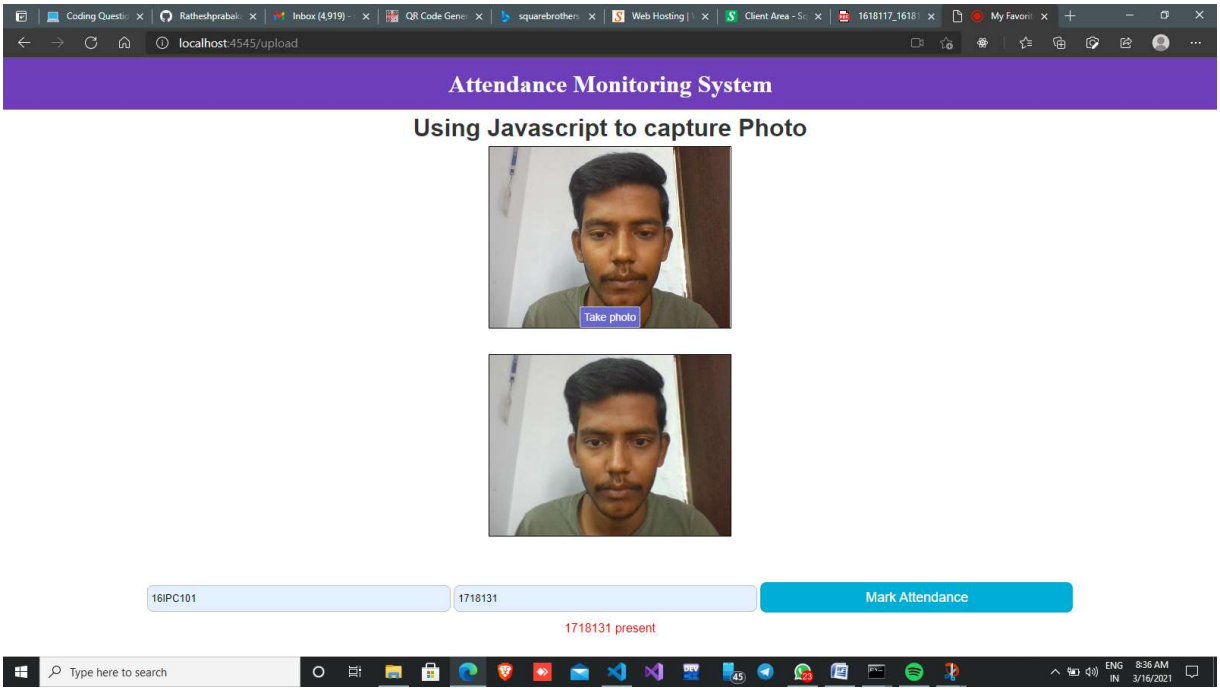


Fig 5.2.6 FACULTY DASHBOARD



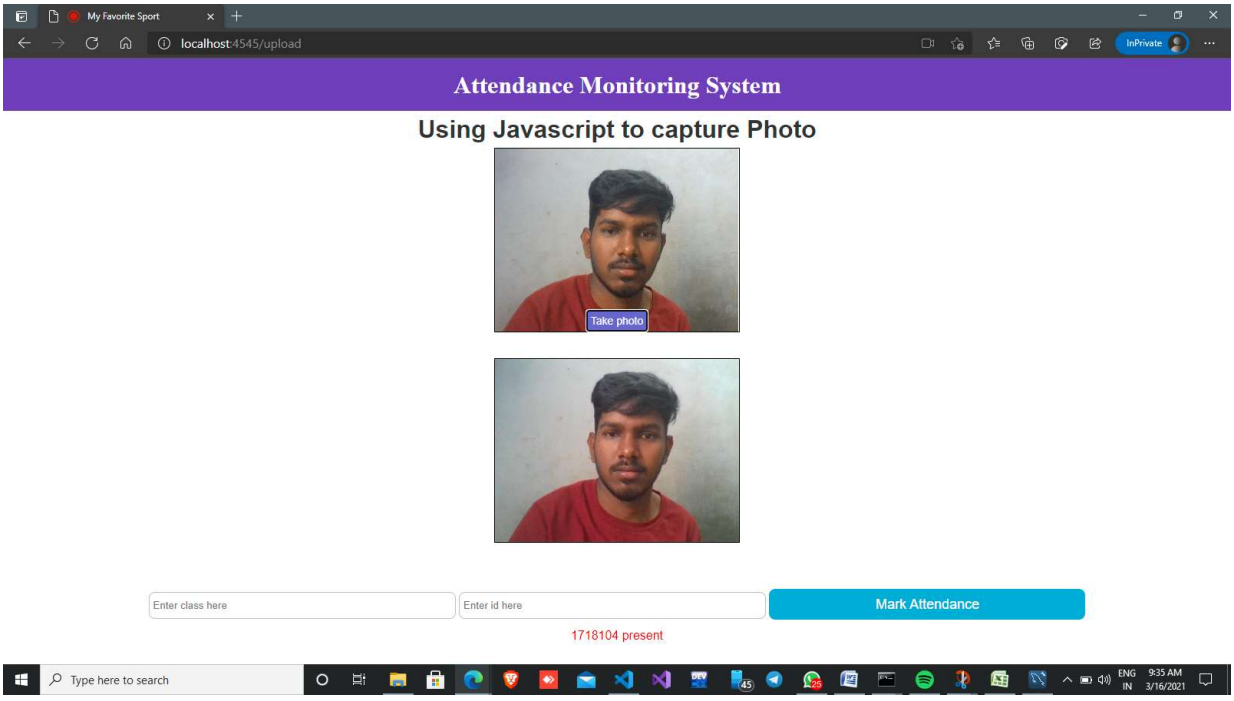


Fig 5.2.7 TAKE ATTENDANCE

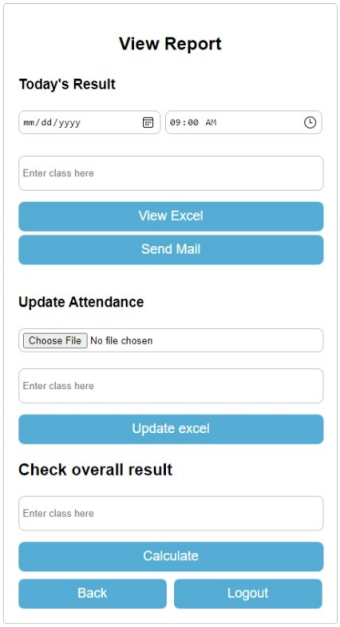


Fig 5.2.8 REPORT PANEL

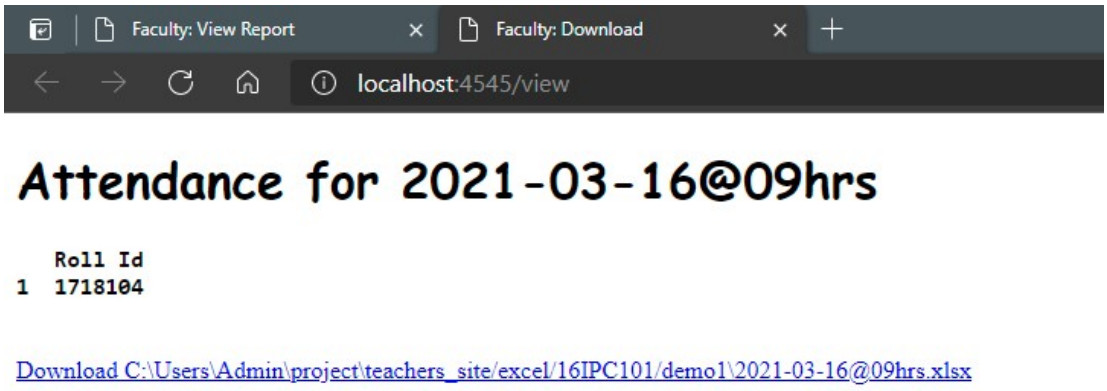


Fig 5.2.9 ATTENDANCE VIEW

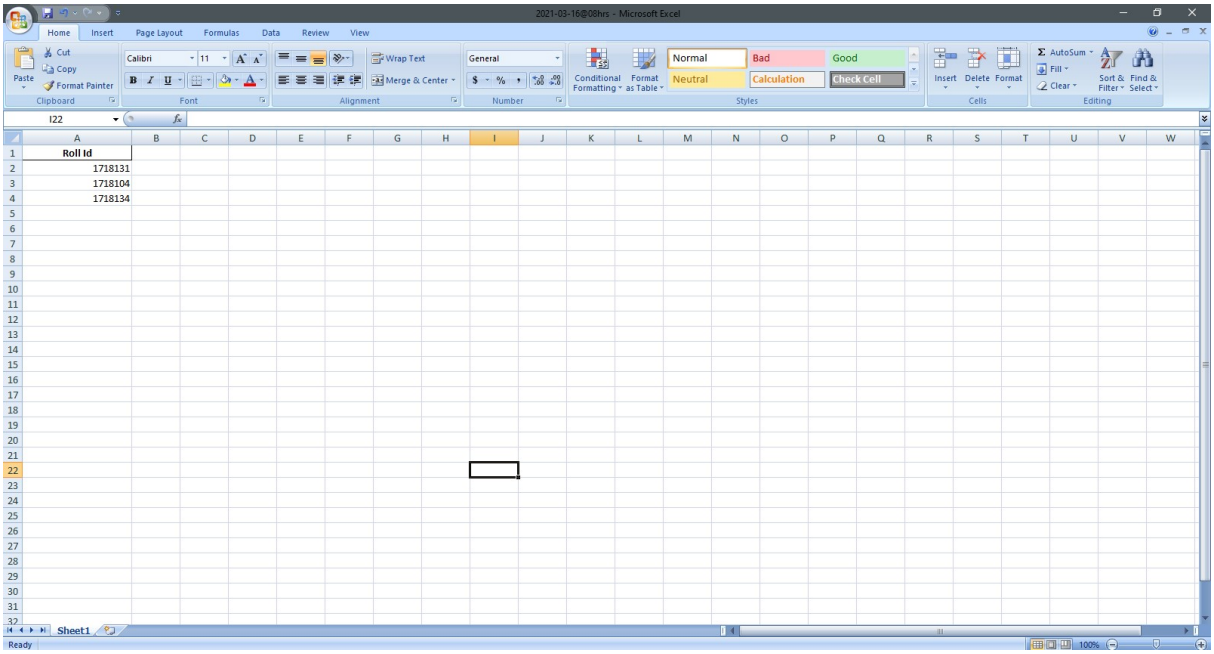


Fig 5.2.10 REPORT VIEW

CHAPTER 6

CONCLUSION

In this project, we have developed a smart attendance system with face recognition technology. A model has been developed by training the students using face_recognition through dlib library while the models that have been generated can be used to mark the attendance. Thus the comparable detection can be obtained with satisfactory prediction accuracy. Different from the traditional methods, the process of marking and maintaining attendance can be performed in a face detection manner with the proposed model. The experimental results have shown that the attendance of the student can be marked and store it in the database.

CHAPTER 7

REFERENCES

- **[1]** Yohei Kawaguchi, Tetsuo Shoji, **Face Recognition-Based Lecture Attendance System**, 3rd AERU, 2005.

- **[2]** B. Kavinmathi, S.Hemalatha, **Attendance System for Face Recognition using GSM module**, 4th International Conference on Signal Processing and Integrated Networks, 2018.

- **[3]** Ketan N. Mahajan, Nagaraj V. Dharwadkar, **Classroom Attendance System Using Surveillance Camera**, International Conference on Computer Systems, Electronics and Control ,2017.

- **[4]** Shubhobrata Bhattacharya, Gowtham Sandeep Nainala, Prosenjit Das, Aurobinda Routray ,**Smart Attendance Monitoring System (SAMS): A Face Recognition based Attendance System for Classroom Environment**, IEEE 18th International Conference on Advanced Learning Technologies, 2018.

- **[5]** E.Varadharajan, R.Dharani, S.Jeevitha, **Automatic Attendance Management System using Face Detection,** 2017.

- **[6]** Guo, Jing-Ming, **Complexity Reduced Face Detection using Probability- based Face Mask Prefiltering and pixel-based Hierarchical-feature Adaboosting, Signal Processing Letters**, IEEE 2011.

- **[7]** K.Senthamil Selvi1, P.Chitrakala, A.Antony, Jenitha S, **Face Recognition based Attendance Marking System**, International Journal of Computer Science and Mobile Computing, 2014.

- **[8]** Chen, Joy Iong Zong. **Smart Security System for Suspicious Activity Detection in Volatile Areas**. Journal of Information Technology 2, 2020.

- **[9]** Jacob, I. Jeena. **Capsule Network based Biometric Recognition System.** Journal of Artificial Intelligence 1,2019.

- **[10]** Kirtiraj Kadam, Manasi Jadhav, Shivam Mulay, Tushar Indalkar, **Attendance Monitoring System Using Image Processing and Machine Learning**, International Journal of Advance Engineering and Research Development, 2017.

- **[11]** Rajat Kumar Chauhan, Vivekanand Pandey, Lokanath M, **Smart Attendance System Using CNN**, International Journal of Pure and Applied Mathematics,2018.

- **[12]** Mayank Yadav, Anmol Aggarwal, **Motion based Attendance System in Real Time Environment for Multimedia Application**, 2018.

- **[13]** Wei Wu, Chuanchang Liu, Zhiyuan Su, **Novel Real-time Face Recognition from Video Streams**, International Conference on Computer Systems, Electronics and Control, 2017.

- **[14]** Changxing Ding, Dacheng Tao, **Trunk-Branch Ensemble Convolutional Neural Networks for Video-Based Face Recognition**, IEEE transactions on pattern analysis and machine intelligence, 2018.

- **[15]** Aziza Ahmedi, Dr Suvarna Nandyal, **An Automatic Attendance System Using Image processing**, The International Journal of Engineering and Science, 2015.