

AI Face Recognition Attendance System (Website)

A Project Report

Submitted in partial fulfillment of the
Requirements for the award of the Degree of
BACHELOR OF SCIENCE (INFORMATION TECHNOLOGY)

By

RAMESH CHAUHAN

UID: 19BIT007

**Under the esteemed guidance of
Mr. Wilson Rao and Ms. Bertilla Fernandes**



DEPARTMENT OF INFORMATION TECHNOLOGY

JAI HIND COLLEGE

(Autonomous)

MUMBAI, 400020

MAHARASHTRA

2022-23

JAI HIND COLLEGE
(Autonomous)
MUMBAI, 400020
MAHARASHTRA

DEPARTMENT OF INFORMATION TECHNOLOGY



CERTIFICATE

This is to certify that the project entitled, “AI Face Recognition Attendance System”, is bonafied work of Ramesh Chauhan bearing UID / Roll No.: (19BIT007) submitted in partial fulfillment of the requirements for the award of degree of BACHELOR OF SCIENCE in INFORMATION TECHNOLOGY from Jai Hind College Autonomous (University of Mumbai).

Internal Guide

Coordinator

External Examiner

Date:

College Seal

DECLARATION

I hereby declare that the project entitled, “AI Face Recognition Attendance System” done at Jai Hind College, has not been in any case duplicated to submit to any other university for the award of any degree. To the best of my knowledge other than me, no one has submitted to any other university.

The project is done in partial fulfilment of the requirements for the award of degree of BACHELOR OF SCIENCE (INFORMATION TECHNOLOGY) to be submitted as final semester project as part of our curriculum.

Name and Signature of the Student

ACKNOWLEDGEMENT

I am extremely grateful for the guidance of our Head of Department (Information Technology & Software Development) Mr. Wilson Rao. Sir had great involvement in making sure my project is a well-rounded and a flawless system by constantly guiding us till the completion of our project work by providing all the necessary information for developing a good system.

I would like to express immense gratitude to the people who have helped me throughout the course of my project. I am grateful to Prof. Ms. Bertilla Fernandes for her constant encouragement and support.

I would also like to thank all of my friends and my seniors who supported and helped me in completing the project, where they all had their own interesting takes on the technology stack, and the own interesting ideas on how to finesse the system even further. I would also like to thank my family for their constant support and encouragement.

Abstract

Face recognition is among the most productive image processing applications and has a pivotal role in the technical field. Recognition of the human face is an active issue for authentication purposes specifically in the context of attendance of students. Attendance system using face recognition is a procedure of recognizing students by using face biostatistics based on the high definition monitoring and other computer technologies. The development of this system is aimed to accomplish digitization of the traditional system of taking attendance by calling names and maintaining pen-paper records. Present strategies for taking attendance are tedious and time-consuming. Attendance records can be easily manipulated by manual recording. After face recognition attendance reports will be generated and stored in excel format. The system is tested under various conditions like illumination, head movements, the variation of distance between the student and cameras. After vigorous testing overall complexity and accuracy are calculated. The Proposed system proved to be an efficient and robust device for taking attendance in a classroom without any time consumption and manual work. The system developed is cost-efficient and need less installation.

TABLE OF CONTENTS

1. INTRODUCTION

1.1 Background.....	8
1.2 Objectives.....	9
1.3 Purpose, Scope, and Applicability.....	10
1.3.1 Purpose.....	10
1.3.2 Scope.....	10
1.3.3 Applicability.....	10
1.4 Achievements.....	11
1.5 Organization of Report.....	12

2. SURVEY OF TECHNOLOGIES 13

3. REQUIREMENT AND ANALYSIS 15

3.1 Problem Definition.....	15
3.2 Requirement Specification.....	15
3.3 Planning and Scheduling.....	16
3.4 Software and Hardware Requirements.....	17
3.5 Preliminary Product Description.....	18
3.6 Conceptual Models.....	19

4. SYSTEM DESIGN 27

4.1 Basic Modules.....	27
4.2 User Interface Design.....	29
4.3 Security Issues.....	36
4.4 Test Cases Design.....	37

5. IMPLEMENTATION AND TESTING 40

5.1 Implementation Approach.....	40
5.2 Coding Details and Code Efficiency.....	43
5.3 Testing Approach.....	59
5.3.1 Unit Testing.....	59
5.3.2 Integration Testing.....	59
5.3.3 System Testing.....	59
5.3.4 Validating Testing.....	59
5.3.5 Output Testing.....	59
5.3.6 User Acceptance Testing.....	59

5.4 Modifications and Improvements.....	60
---	----

6. RESULTS AND DISCUSSION 61

6.1 User Documentation.....	61
-----------------------------	----

7. CONCLUSION 72

7.1 Conclusion.....	72
7.2 Limitations of the System.....	72
7.3 Future Scope of the Project.....	72

List of Figures

1. Gantt Chart.....	16
2. PERT Chart.....	16
3. Event Table.....	19
4. ER Diagram.....	20
5. Class Diagram.....	21
6. Activity Diagram.....	22
7. Sequence Diagram.....	23
8. Use Case Diagram.....	23
9. State Diagram.....	24
10. Package Diagram.....	24
11. Deployment Diagram.....	25
12. Component Diagram.....	25
13. Data Flow Level 0 Diagram.....	26
14. Home page.....	62
15. Teacher Registration Page.....	62
16. Admin Login Page.....	63
17. Student Registration page.....	64
18. Teacher Login Page.....	66
19Attendance panel Page.....	67
20. Report Page.....	68

Chapter-1:- INTRODUCTION

1.1] Background:-

Every organization requires a robust and stable system to record the attendance of their students. and every organization have their own method to do so, some are taking attendance manually with a sheet of paper by calling their names during lecture hours and some have adopted biometrics system such as fingerprint, RFID card reader, Iris system to mark the attendance. The conventional method of calling the names of students manually is time consuming event. The RFID card system, each student assigns a card with their corresponding identity but there is chance of card loss or unauthorized person may misuse the card for fake attendance. While in other biometrics such as finger print, iris or voice recognition, they all have their own flaws and also they are not 100% accurate.

Use of face recognition for the purpose of attendance marking is the smart way of attendance management system. Face recognition is more accurate and faster technique among other techniques and reduces chance of proxy attendance. Face recognition provide passive identification that is a person which is to be identified does not to need to take any action for its identity. Face recognition involves two steps, first step involves the detection of faces and second step consist of identification of those detected face images with the existing database. There are number of face detection and recognition methods introduced. Face recognition works either in form of appearance based which covers the features of whole face or feature based which covers the geometric feature like eyes, nose, eye brows, and cheeks to recognize the face.

Our system uses face recognition approach to reduce the flaws of existing system with the help of machine learning, it requires a good quality camera to capture the images of students, the detection process is done by histogram of oriented gradient. And recognizing performs through deep learning.

1.2] Objectives:

- To detect the face segment from the video frame.
- To extract the useful features from the face detected.
- To classify the features in order to recognize the face detected.
- To record the attendance of the identified student.

1.3] Purpose, Scope, and Applicability

1.3.1] Purpose:-

The motivation behind this proposed work comes from the advancement of technologies like image processing, AI and machine learning and Face unlock feature given by smart-phone manufacturers. The classroom often consists of huge number of students which usually takes a lot of time thus creating a system which will automatically detect the present students and then marking the students accordingly will be very helpful, Just like staff Id scanning but way too easier than that. This system will also reduce manipulation of attendance records by the students.

1.3.2] Scope:-

- Automating the whole process so that we have digital environment.
- Use the Live face Recognition to recognize each individual and mark their attendance automatically.
- Utilizes video and image processing to provide input to the system
- Automate update in the attendance sheet without human intervention.
- To keep the student update with their attendance ratio.

1.3.3] Applicability:-

1. To verify identities in Government organizations.
2. Enterprises.
3. Attendance in Schools and colleges.
4. To detect fake entries at international borders.
5. Industries.

1.4] Achievements:-

There is also problem that student can mark their friends attendance by showing their photo from mobile i.e. spoof attack.

For spoof detection I am using tensor flow inception model by retraining it's last layer so that it can detect mobile phones in an image.

1.5] Organization Report

- **Requirement Specification**
- **Functional Requirements**

- The system has different functionalities for an admin and teachers. Admin has higher privileges than teachers. Taking and tracking student attendance by facial recognition in specific time.
- Sending the names of the absent student directly to the respective parents via email.

Admin Module

Admin has the highest privileges among all as admin is responsible to design the system. Admin register teacher and provide unique id to the teacher. They are responsible to take images of the students and add them to the database. Admin can view and update the details of both students and teachers. They can also view the attendance report

Teacher Module

Teachers can log in to the system. They can open the application and the images of the students for attendance. They can also view the attendance report.

2] Survey of Technology:

Project contains two websites developed using flask and python3.

Flask is a web application framework written in Python. Flask is based on Werkzeug WSGI toolkit and Jinja2 template engine. Both are Pocco projects. Python 2.6 or higher is usually required for installation of Flask. Although Flask and its dependencies work well with Python 3 (Python 3.3 onwards), many Flask extensions do not support it properly. Hence, it is recommended that Flask should be installed on Python 2.7.

Install virtualenv for development environment

Virtualenv is a virtual Python environment builder. It helps a user to create multiple Python environments side-by-side. Thereby, it can avoid compatibility issues between the different versions of the libraries.

The following command installs virtualenv

```
pip install virtualenv
```

```
virtualenv venv
```

To activate corresponding environment:-

```
venv\scripts\activate
```

We are now ready to install Flask in this environment.

```
pip install Flask
```

In order to test Flask installation, type the following code in the editor as Hello.py

```
from flask import Flask
app = Flask(__name__)

@app.route('/')
def hello_world():
    return 'Hello World'

if __name__ == '__main__':
    app.run()
```

Database used : MySQL community edition.

- 1) For face reocognition I used python3 "face_recognition" by ageitgey. Built using dlib's state-of-the-art face recognition built with deep learning. The model has an accuracy of 99.38% on the Labeled Faces in the Wild benchmark.
- 2) For spoof detection I used tensorflow inception model by retraining it's last layer so that it can detect mobile phones in an image.
- 3) To generate and manage excel I used xlsx and xlrd and pandas.
- 4) For sending email's I used flask-mail.
- 5) admin site dependency : flask, mysqlclient, sklearn, numpy, scipy, pillow, dlib, face_recognition
- 6) teachers site dependency : flask_bootstrap, pytz, xlswriter, pandas, flask_mail, tensorflow,xlrd

Image Acquisition: The system consists of a camera on a mobile/laptop/Computer that captures the images of the student and sends it to the image pre-processing. Then that image is sends for face detection.

B. Face Detection: This process separates the facial area from the rest of the background image. The faces which are stored in the database.

C. Feature Extraction: Feature extraction is done for distinguishing faces of different student. In this system, eyes, nose and mouth are extracted. Feature extraction is helpful in face detection and recognition.

D.Face Recognition: The face image is then compared with the stored image. If the face image is matched with the stored image then the face is recognized. Then for that particular student the attendance is recorded.

3] Requirement analysis:

3.1] Problem Definition:

Taking and tracking students' attendance manually, losing attendance sheets, dishonesty, wasted time and high error scales are problems facing the lecturers use the existing attendance system. It is a hard process, take time and cause a lot of paper-based work. As a result, in order to solve these problems and avoid errors we suggest computerizing this process by providing a system that record and manage students' attendance automatically without needing to lecturers' interference.

3.2] Requirement Specification:

3.2.1] Functional Requirements:-

After getting valuable information we reached to the following important conclusions: -

1. It should meet the functional requirements as mentioned in Objectives.
2. It should be able to handle 'png,jpg' and 'jpeg' images.

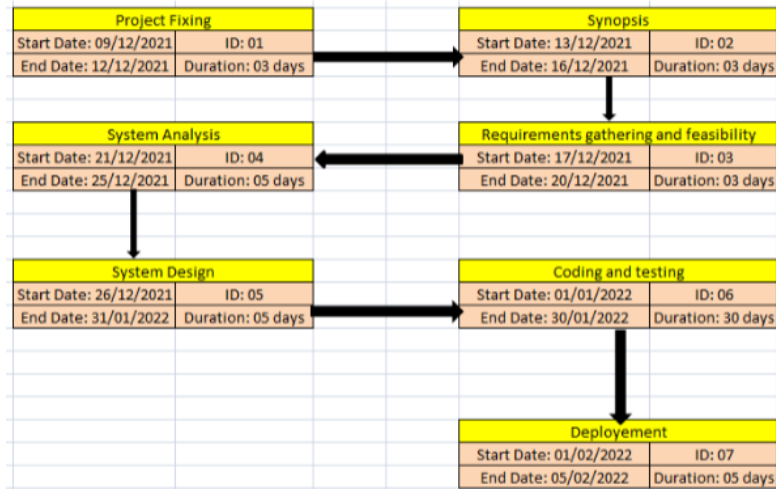
3.2.2] Non-functional Requirements:-

This project is intended to meet the following non functional requirements: -

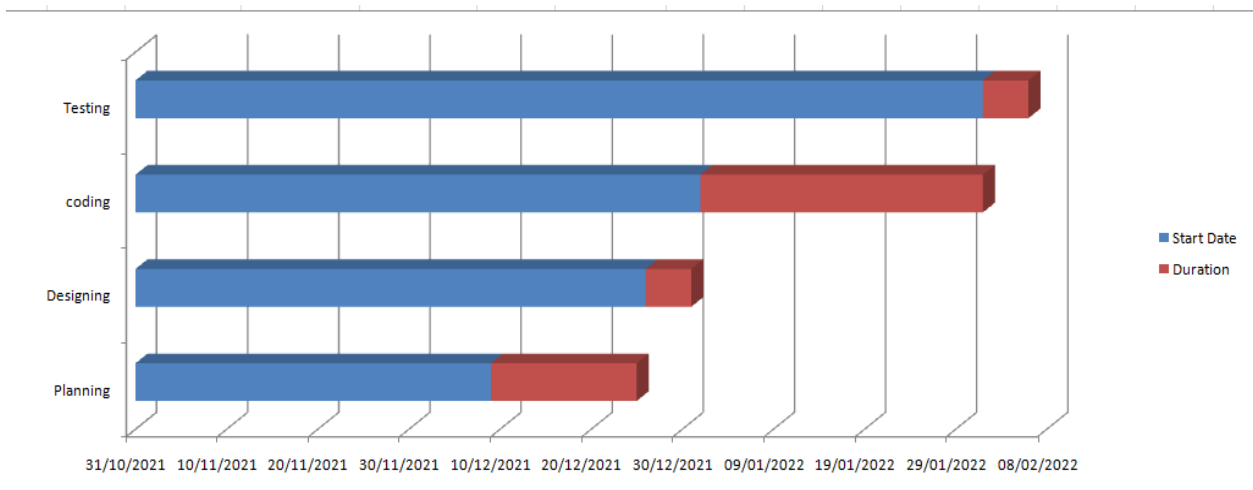
- 1) This face recognition software should be available on the Internet, to enable the users to use, download it any time.
- 2) The program should be platform independent.

3.3] Planning and Scheduling

Pert Chart



Gantt Chart:-



3.4] Software and Hardware Requirement

Hardware

Processor	Intel@ Pentium quad core
Memory	4GB
ROM	1TB

Software:-

IDE	VsCode
Language	Html+Css+JavaScript+Python+Flask
Database	Mysql
Operating system	Windows 10

3.5] Preliminary Product Description:-

Modules:-1) Admin Home page:-System initiate with admin home screen. Here user will get 2 Navigation tab as admin login and Teacher Register. Admin can login with register credential. Admin first need to register the teacher.

Modules:- 2) Teacher Registration: Admin first need to Register the teacher then he/she will get access of Admin-site. Here admin needs to put name, email, and password to register the account.

Modules:- 3) Admin Login: Admin need to first login to account to get access of Student registration form. Here admin will need to put username and password to login. While username and password must match to registration username and password. Otherwise admin won't able to login into system

Module:-4) Student Registration: After admin login successfully he/she will get access of student registration page. Admin need to first register the student details and also upload the picture of student while registration to create the model of that particular student.

Teacher-site:-

Module-5) Teacher Login: Now using teacher's site (It will be used when teacher will actually enter the class), teacher has to login first.

Module:-6) Attendance panel:- After teacher successfully login teacher will get two options attendance panel and report panel. After clicking on attendance tab there will be no back button as teacher will pass on the phone to student. Student's will then have to just click a snap enter class and roll id and press enter to mark their attendance.

After that there is also a problem of spoof attack in face recognition i.e. someone will show someone's image of face through their mobile phone and trick our webapp and they will mark the attendance of their friends who were not present.

But I solved this issue using tensorflow, by training inception to detect mobile phones in an image then I used that model in the webapp as soon as student's click a snap it will first check if the face is spoof or original. To retrain inception's last layer I used 200 images of mobile phones and I feed them to tensorflow to retrain the last layer of inception.

Module:-8) View Report: Now after student's part is done. Teacher can then login again and then go to report's tab to see attendance. Here there are several options I have given. If teacher want to see today's attendance, just select date and time to see the attendance. And there is also an option to download the attendance sheet in excel form and then again reupload it after making any changes if sometime required by the teacher. And the teacher can also see total attendance for his or her lecture. so that they can analyze how many lectures each student from particular class had attended so far.

One additional feature is that teacher can send email for the attendance marked to all the parents as well as students by selecting class and clicking on send mail button.

3.6] Conceptual Models

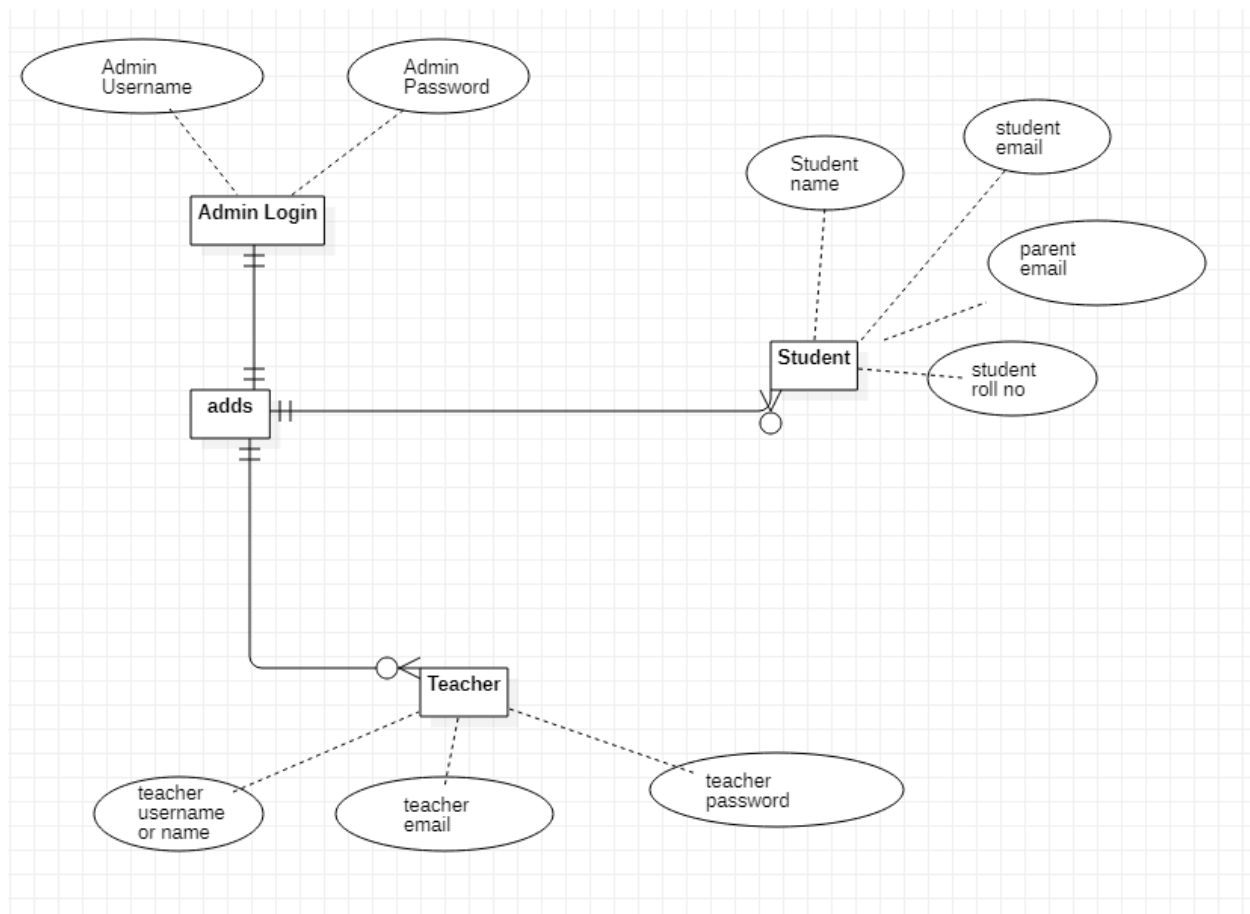
Event Table:

Sr No.	Event	Trigger	Source	Activity	Response	Destination
Admin-site						
1	Teacher-Registration	Registration request	website	Email and password authentication	Account created successful or not	Mysql
2	Student-Registration	Registration Request	website	Name and student email-id and parent-email id get register and roll no	Register successfully or not	Mysql
3	Student model-image Training	Image upload	website	Creating training data	Model get created	Mysql

Sr No.	Event	Trigger	Source	Activity	Response	Destination
Teacher-site						
1	Teacher-Login	Login request	website	username and password credential check	Login with register credential or not	Mysql
2	Attendance panel	Face Recognition	website	Proper face recognition	Face match or not with the database image	Mysql
3 View-Report panel		excel sheet generate	website	Excel sheet download	download	excel
		Send mail to parent	website	Sending mail	Received or not	Gmail.com

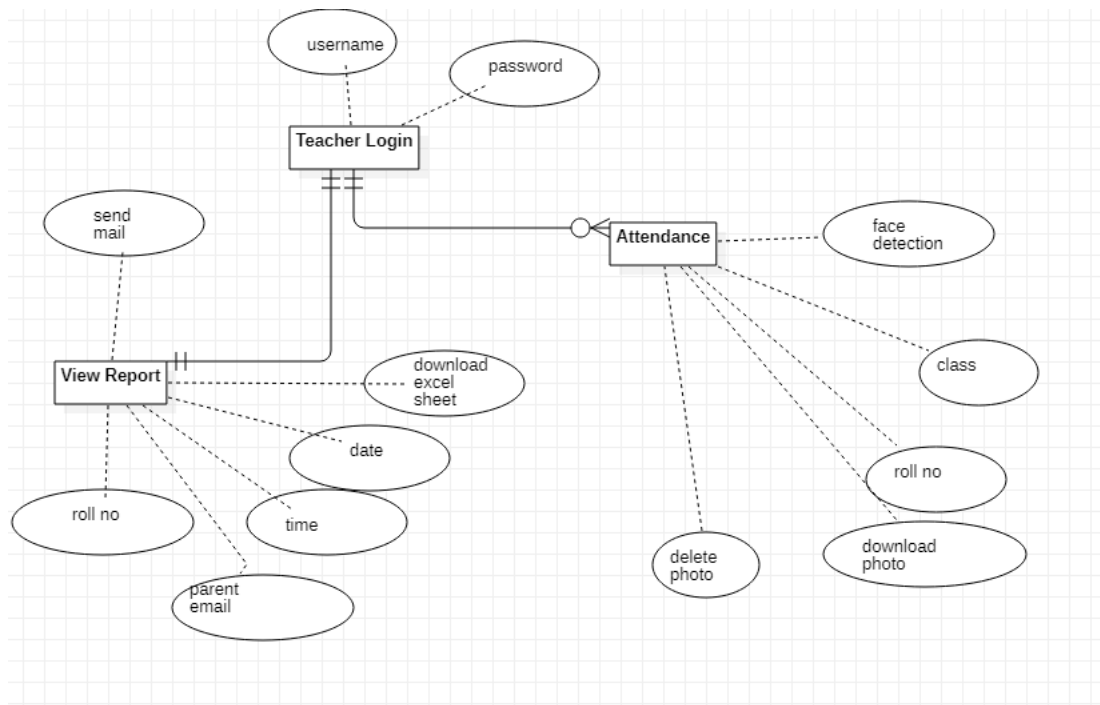
Admin-site

Admin ER Diagram:-

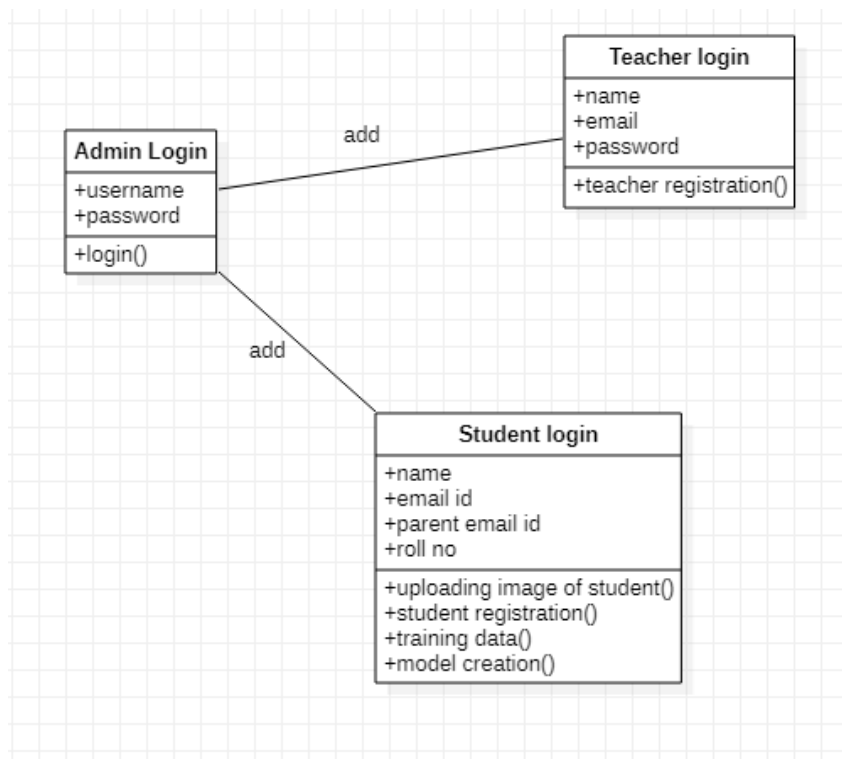


Teacher-site

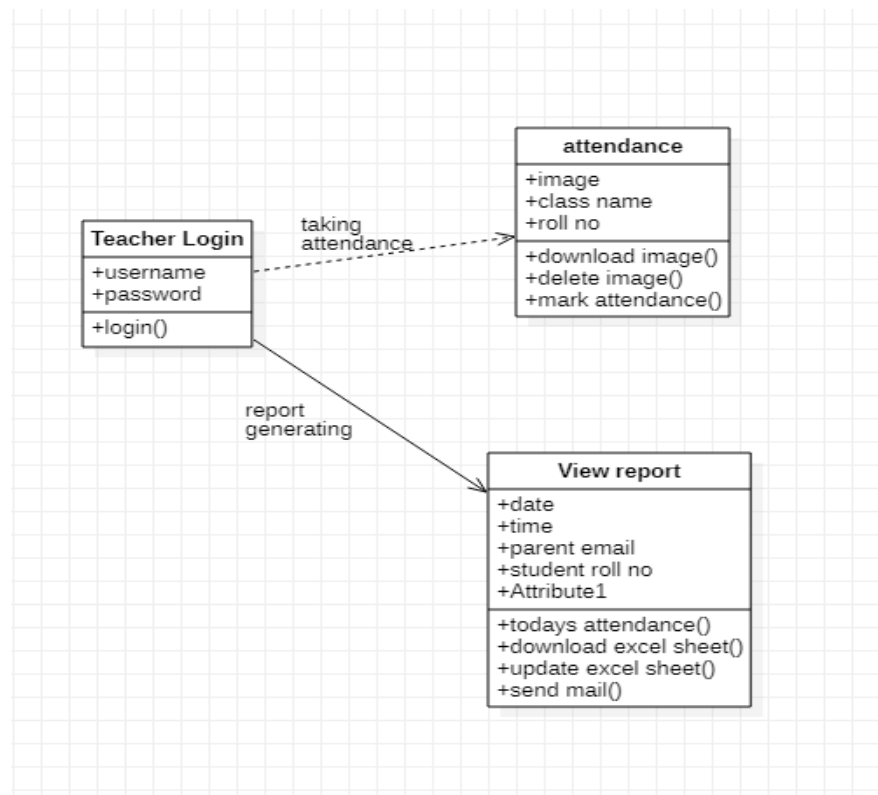
Teacher-ER diagram:-



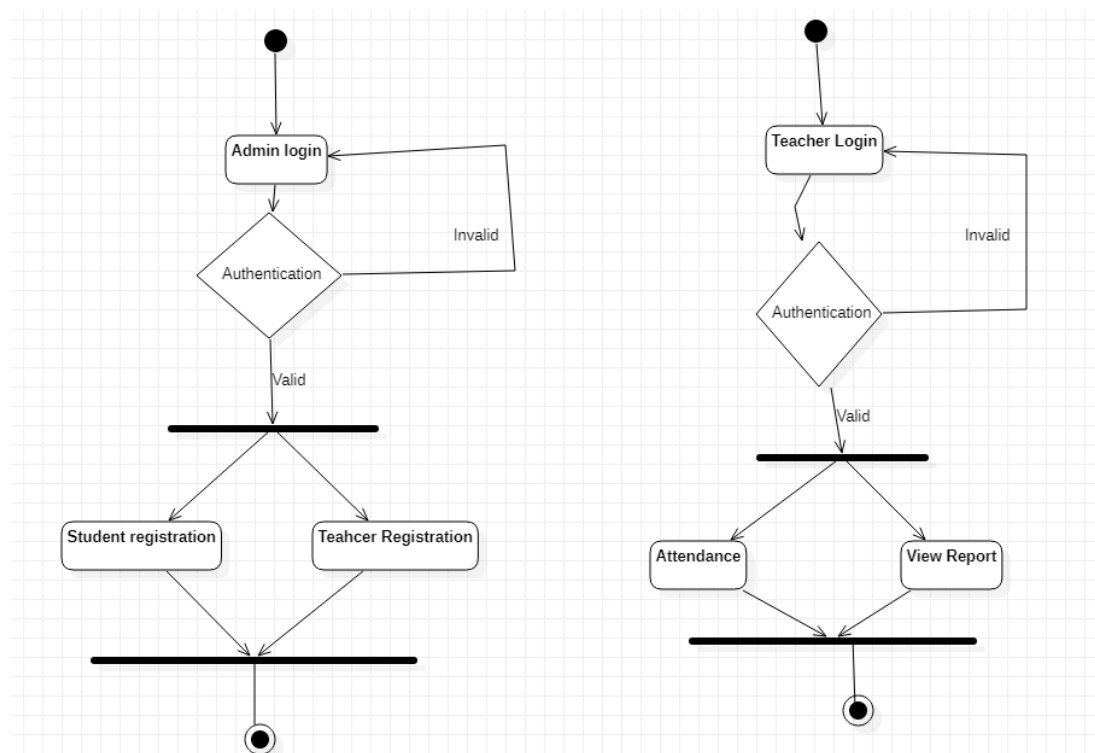
Admin Class diagram



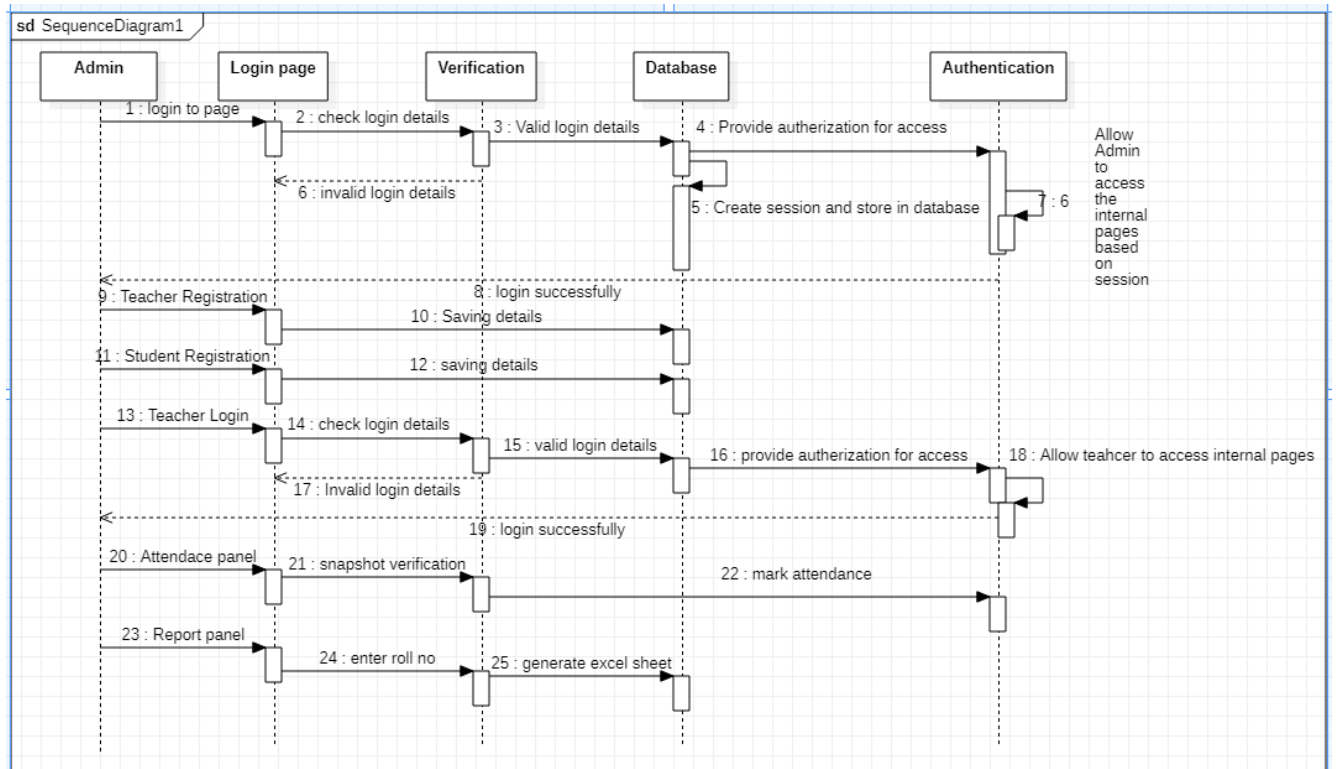
Teacher-site Class diagram:



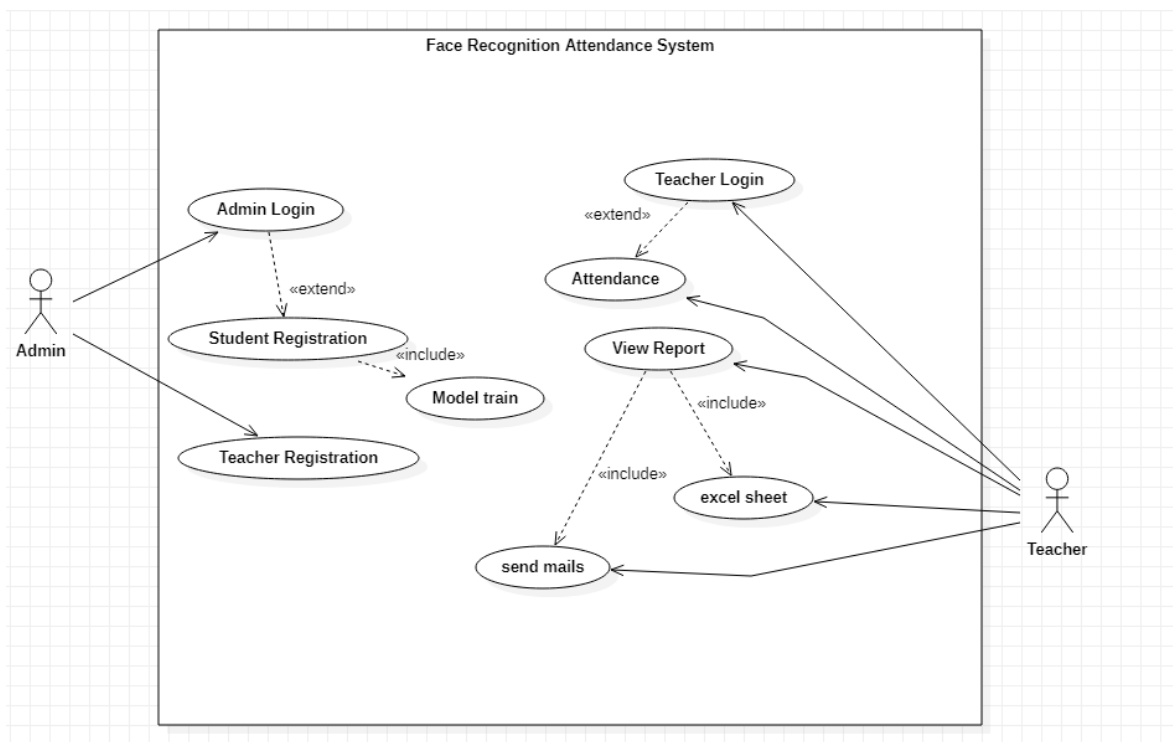
Activity diagram:-



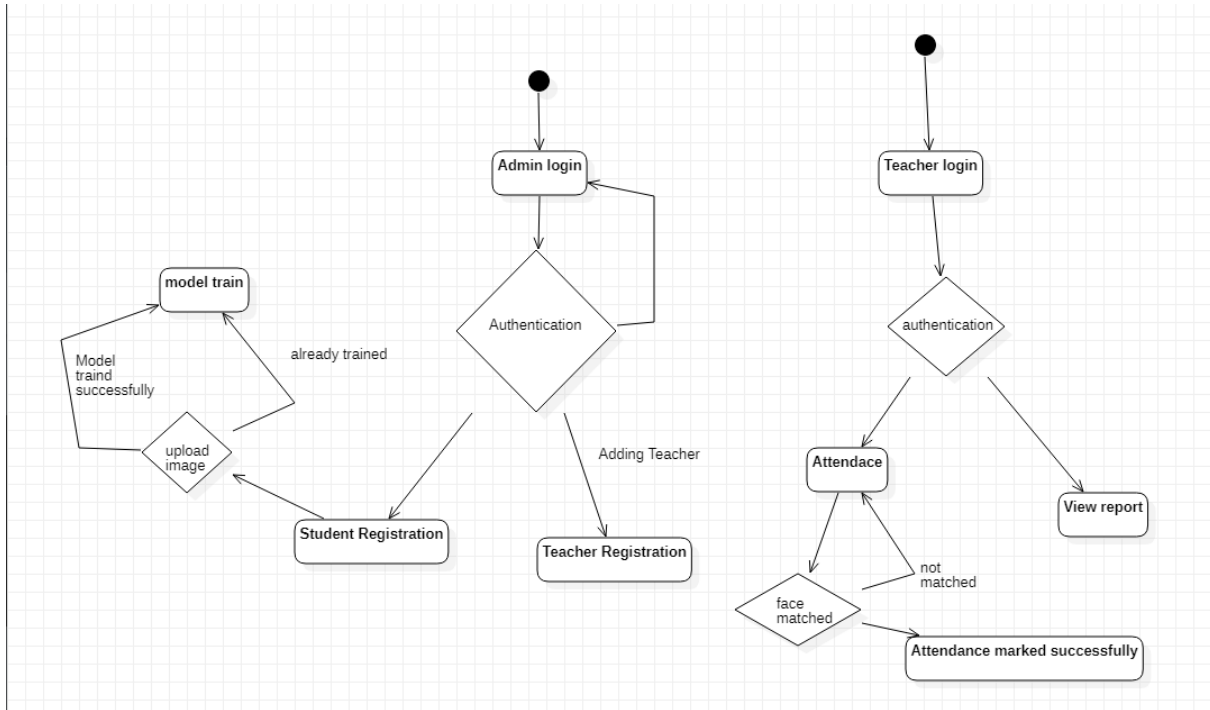
Sequence Diagram:



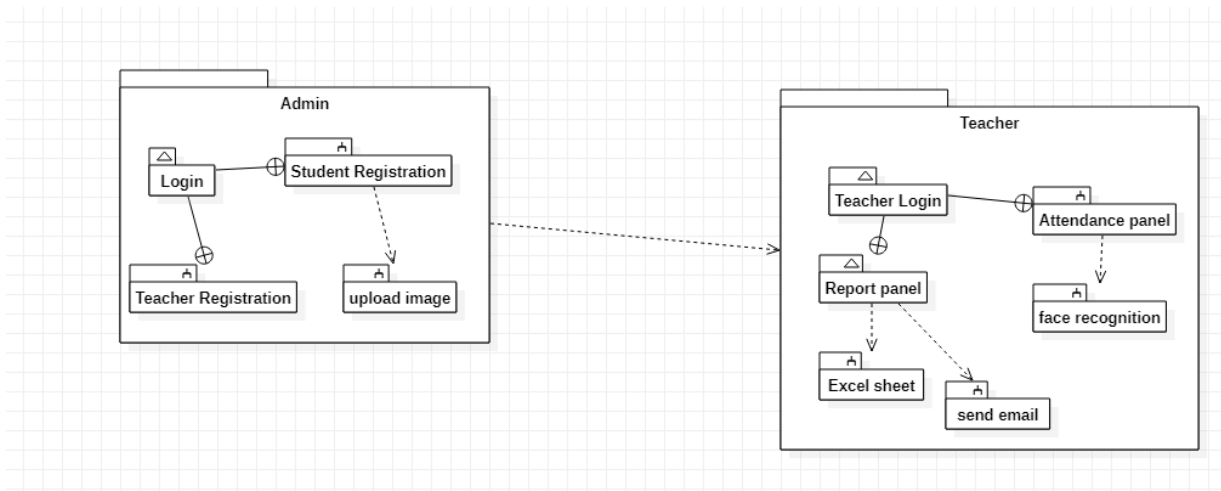
USE CASE:-



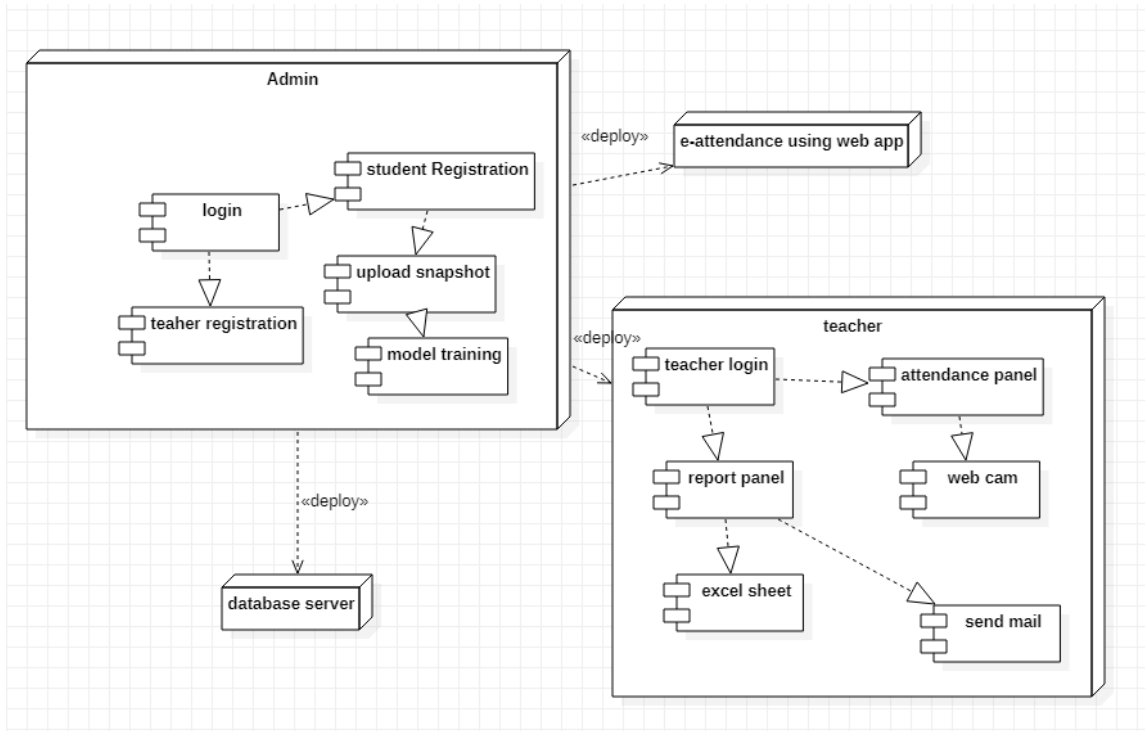
Statechart



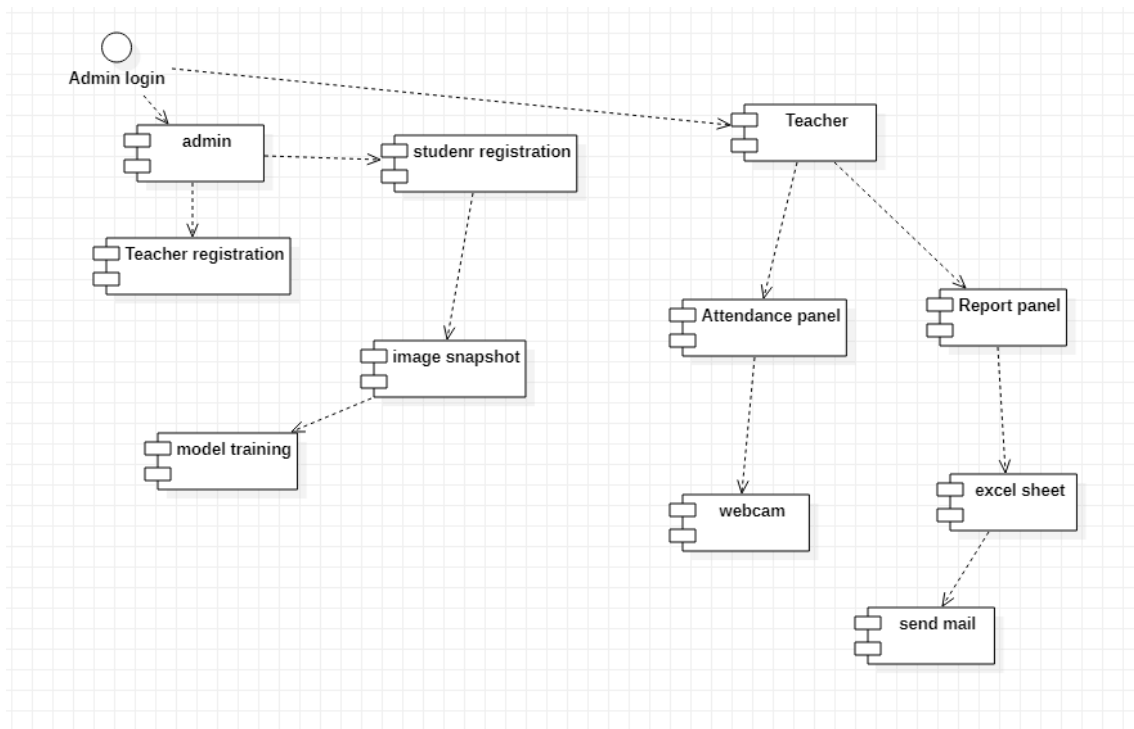
Package Diagram:-



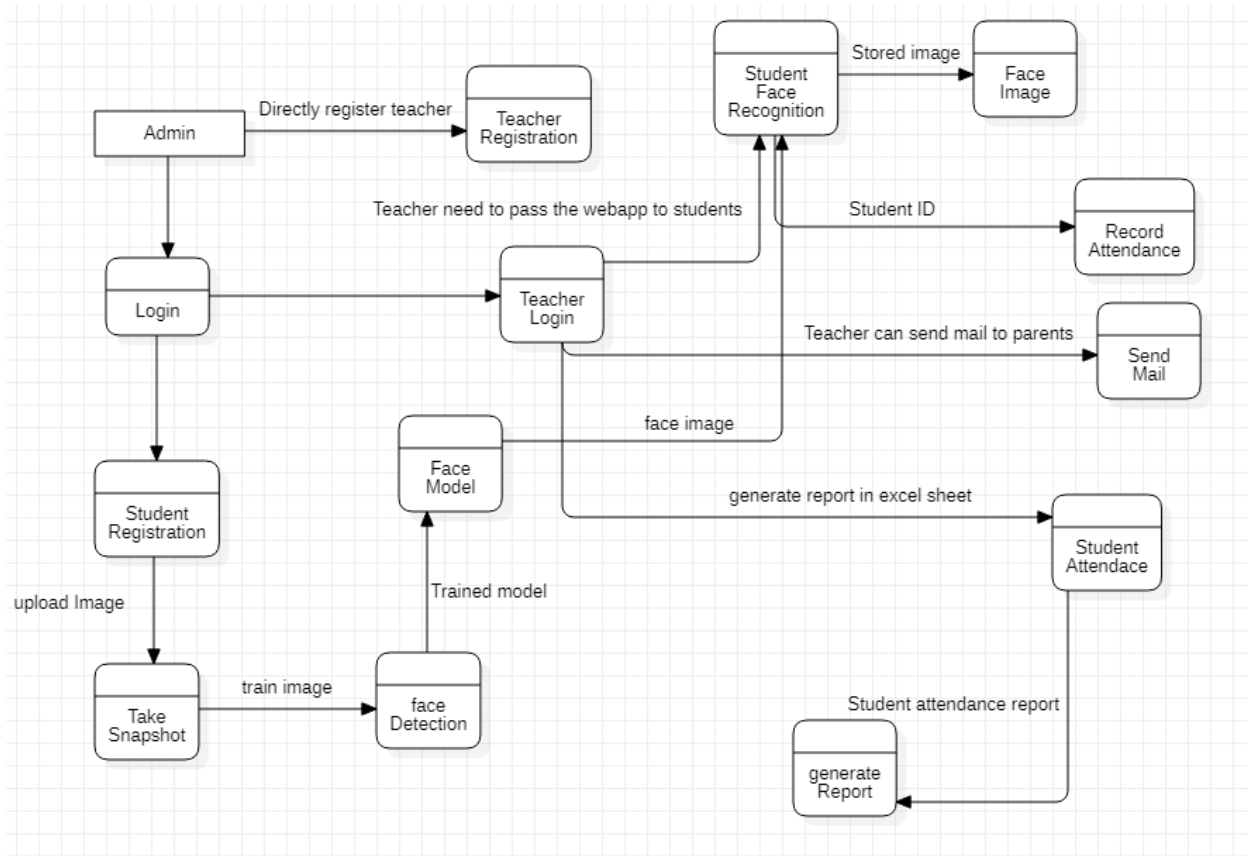
Deployment diagram:



Component diagram



DFD:



4] System Design

4.1] Basic Module

Project contains two webapp's developed using flask and python3. Database used : MySQL community edition. For face reecognition I used python3 "face_recognition" by ageitgey. (https://github.com/ageitgey/face_recognition), Built using dlib's state-of-the-art face recognition built with deep learning. The model has an accuracy of 99.38% on the Labeled Faces in the Wild benchmark.

For spoof detection I used tensorflow inception model by retraining it's last layer so that it can detect mobile phones in an image.

To generate and manage excel I used xlsx and xlrd and pandas.

For sending email's I used flask-mail.

admin site dependency : flask, mysqlclient, sklearn, numpy, scipy, pillow, dlib, face_recognition

teachers site dependency : flask_bootstrap, pytz, xlsxwriter, pandas, flask_mail, tensorflow,xlrd

Admin Site:-

Home Page Module:- Web application starts with home page module. Navigation bar contain three things Admin login , Teacher registration and contact page. It is static page.

Admin Login Module:- Admin is the super user. To access the Student Registration page Admin need to login first with Register Credentials.

Teacher Registatrion module:- Admin can also register teachers using this site.

Student Registration and Model training Module:- The whole concept is at the time of admission to college or school admin should register the students details such as his name email address and also create training data of each student by entering his roll id and taking snaps of his or her frontal face and then webapp will automatically create model for that particular roll id and save it on server, The model which is created for each student is about 8kb in size.

Teachers Site:-

Teacher Login Page Module:- Now using teacher's site (It will be used when teacher will actually enter the class), teacher has to login first to access the Attendace module.

Attendance Module:-Then after clicking on attendance tab there will be no back button as teacher will pass on the phone to student. Student's will then have to just click a snap enter class and roll id and press enter to mark their attendance.

After that there is also a problem of spoof attack in face recognition i.e. someone will show someone's image of face through their mobile phone and trick our webapp and they will mark the attendace of their friends who were not present.

But I solved this issue using tensorflow, by training inception to detect mobile phones in an image then I used that model in the webapp as soon as student's click a snap it will first check if the face is spoof or original. To retrain inception's last layer I used 200 images of mobile phones and I feed them to tensorflow to retrain the last layer of inception.

View Report module:- Now after student's part is done. Teacher can then login again and then go to report's tab to see attendance. Here there are several option's I have given.

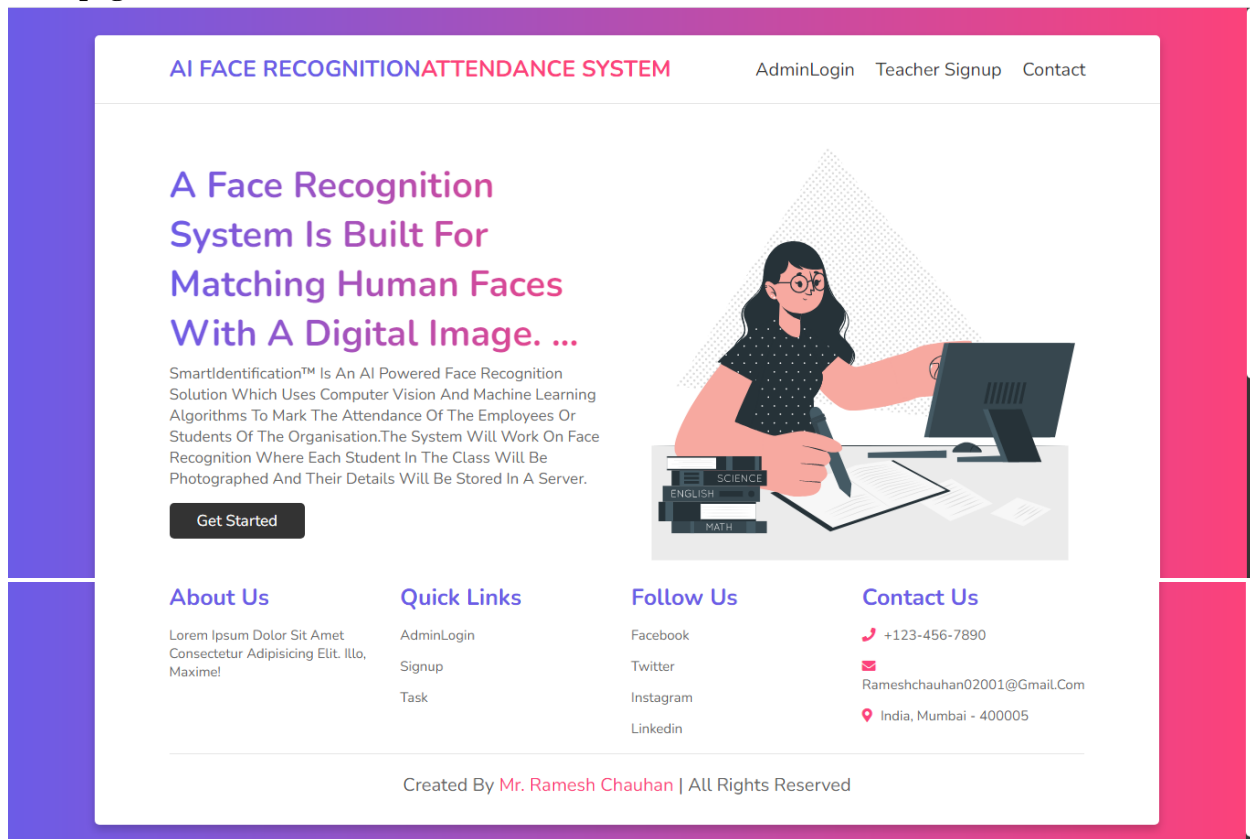
If teacher want to see today's attendace, just select date and time to see the attendance. And there is also an option to download the attendace sheet in excel form and then again reupload it after making any changes if sometime required by the teacher. And the teacher can also see total attendance for his or her lecture. so that they can analyze how many lectures each student from particular class had attended so far.

Send Mail module:- One additional feature is that teacher can send email for the attendance marked to all the parents as well as students by selecting class and clicking on send mail button.

4.2] User Interface

Admin Site

Homepage




Teacher Registration page:-

The screenshot shows the 'Teacher Registration page' for the 'AI Face Recognition Attendance System'. The page has a blue gradient background. A central white box contains a purple header with the text 'REGISTRATION FORM' and 'Enter Your Personal Data'. Below this, there are three input fields labeled 'USERNAME:', 'EMAIL:', and 'PASSWORD:', each with a placeholder text ('Name', 'Email', and 'Password' respectively). A purple 'Submit' button is positioned below the password field. At the bottom of the white box is a blue link labeled 'BackToHome'.

Contact Page:-

AI FACE RECOGNITION ATTENDANCE SYSTEM

CONTACT US



About Us

Lorem Ipsum Dolor Sit Amet Consectetur
Adipiscing Elit. Illo, Maxime!

Follow Us

- Facebook
- Twitter
- Instagram
- Linkedin


Contact Us

- +123-456-7890
- Rameshchauhan02001@Gmail.Com
- India, Mumbai - 400005

Created By **Mr. Ramesh Chauhan** | All Rights Reserved


Admin login Page

AI FACE RECOGNITION ATTENDANCE SYSTEM



Student Registration and Model Training Page

AI FACE RECOGNITION ATTENDANCE SYSTEM

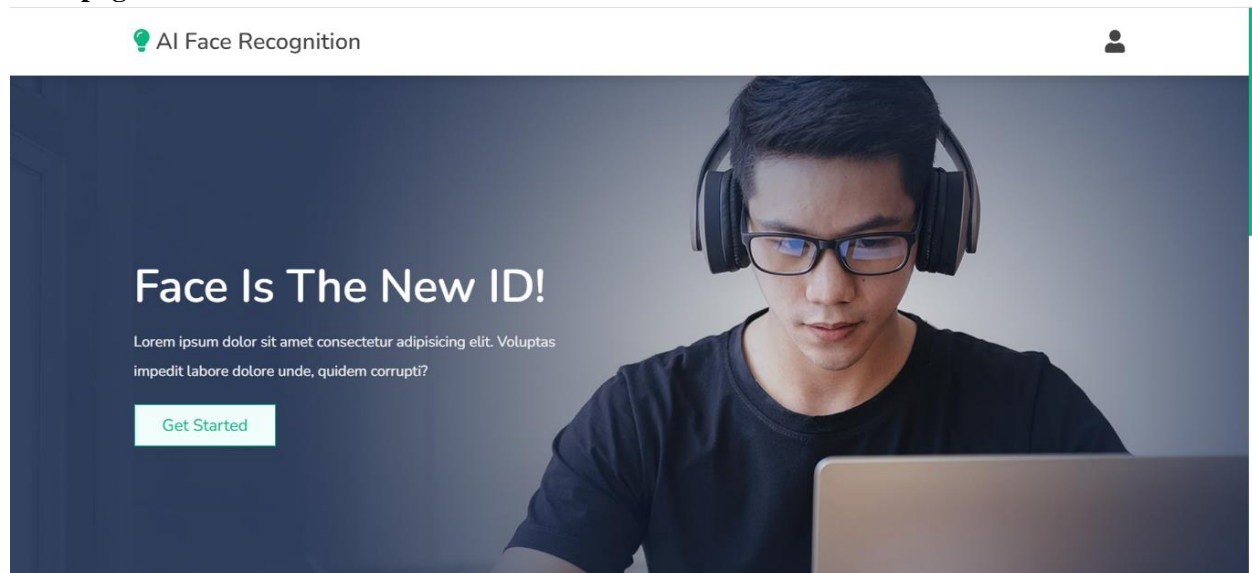


FILE UPLOAD

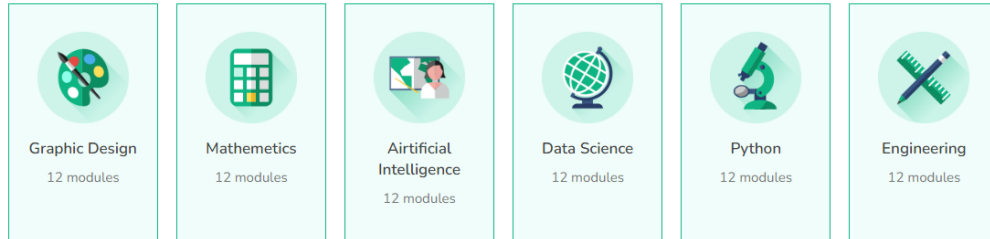
No file chosen

Teachers Site:

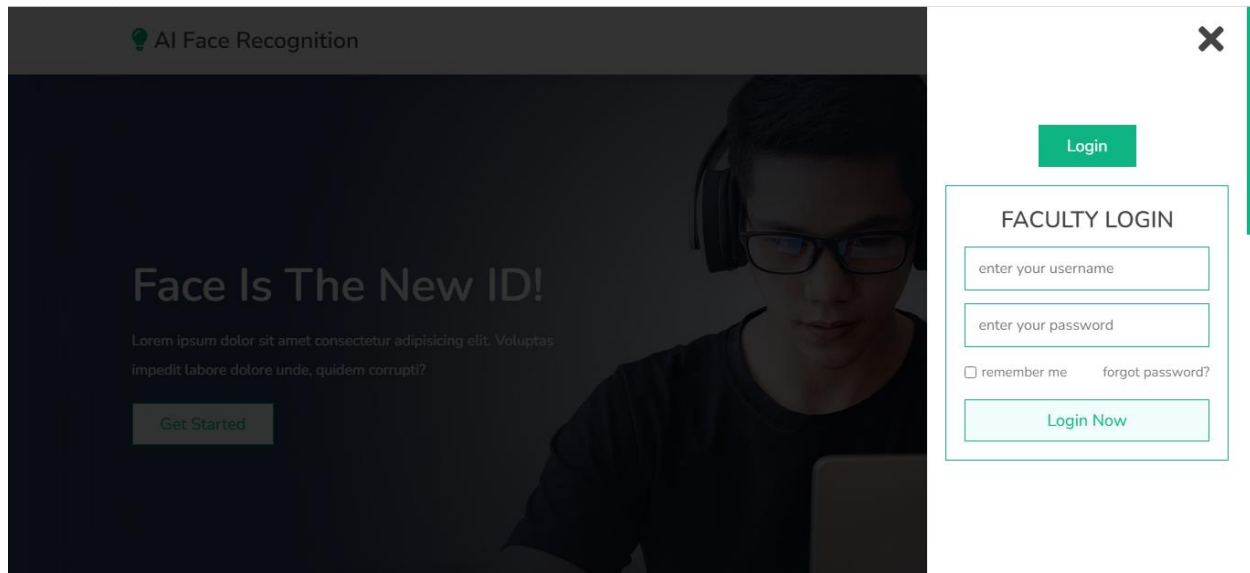
Homepage:



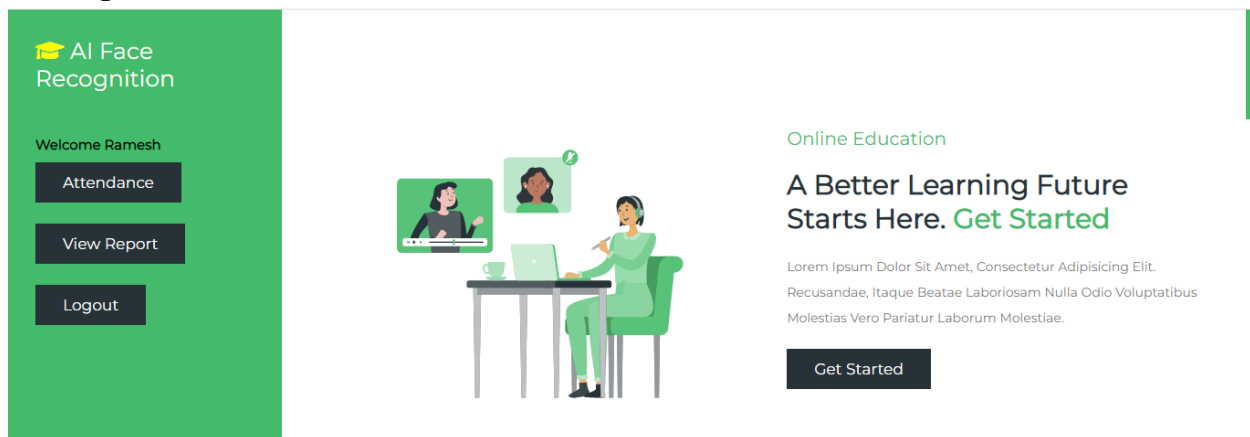
Our Popular Subjects



LoginPage:



TaskPage:



AI Face Recognition

Welcome Ramesh

Attendance

View Report

Logout



+450
Students



+70
Courses



+199
Teachers



+520
Subjects

AI Face Recognition

Welcome Ramesh

Attendance

View Report

Logout

OUR CATEGORY

OUR TOP CATEGORY



Programing

Lorem Ipsum Dolor Sit Amet,
Consectetur Adipiscing Elit. Omnis,
Reiciendis.



Graphic Design

Lorem Ipsum Dolor Sit Amet,
Consectetur Adipiscing Elit. Omnis,
Reiciendis.



Marketing

Lorem Ipsum Dolor Sit Amet,
Consectetur Adipiscing Elit. Omnis,
Reiciendis.



Business Management

Lorem Ipsum Dolor Sit Amet,
Consectetur Adipiscing Elit. Omnis,
Reiciendis.



Music & Dance

Lorem Ipsum Dolor Sit Amet,
Consectetur Adipiscing Elit. Omnis,
Reiciendis.



Science & Biology

Lorem Ipsum Dolor Sit Amet,
Consectetur Adipiscing Elit. Omnis,
Reiciendis.

AI Face Recognition

Welcome Ramesh

Attendance

View Report

Logout

CLIENT REVIEWS

WHAT THEY SAY?



John Deo

Lorem Ipsum, Dolor Sit Amet
Consectetur Adipiscing Elit.
Dignissimos Repellat Sunt A
Temporibus Suscipit Distinctio
Maiores Iste Tenetur Voluptate
Nostrum.



John Deo

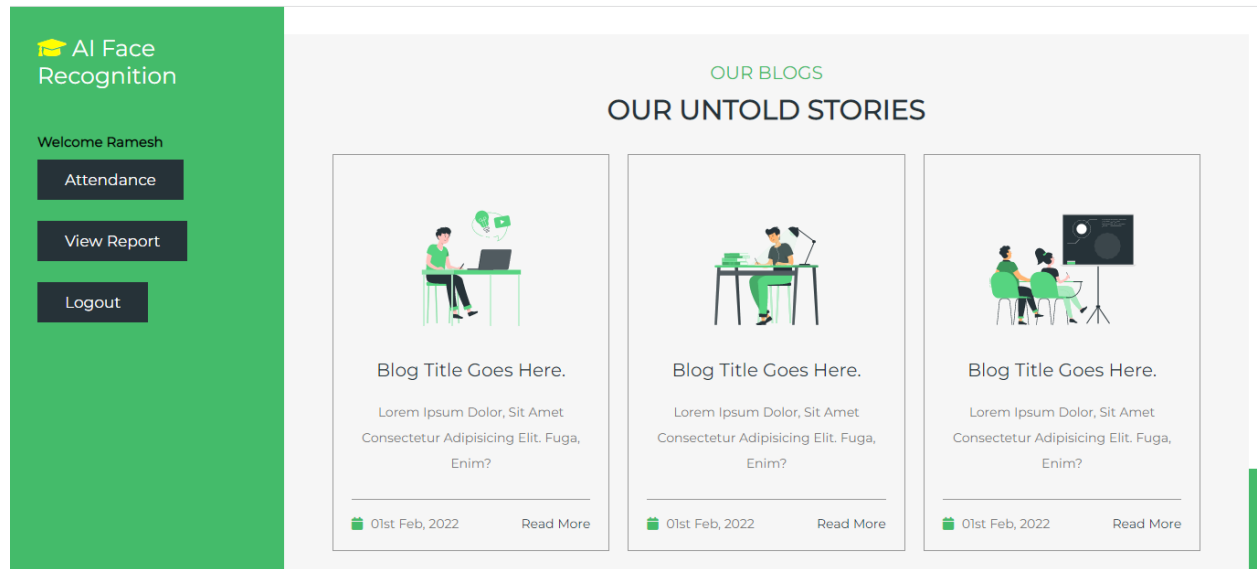
Lorem Ipsum, Dolor Sit Amet
Consectetur Adipiscing Elit.
Dignissimos Repellat Sunt A
Temporibus Suscipit Distinctio
Maiores Iste Tenetur Voluptate
Nostrum.



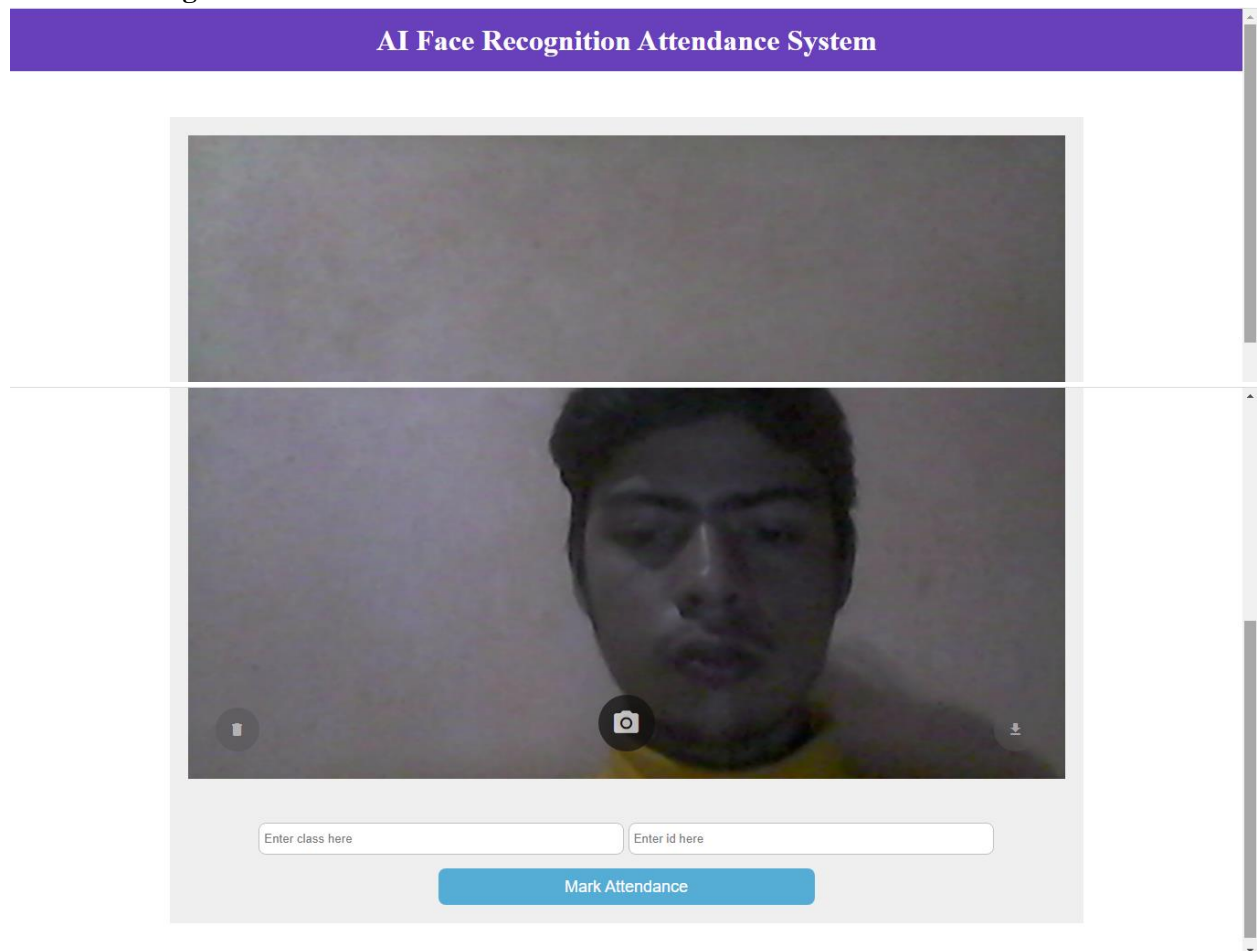
John Deo

Lorem Ipsum, Dolor Sit Amet
Consectetur Adipiscing Elit.
Dignissimos Repellat Sunt A
Temporibus Suscipit Distinctio
Maiores Iste Tenetur Voluptate
Nostrum.





AttendancePage:



View Report Page:

AI FACE RECOGNITION ATTENDANCE SYSTEM



Today's Result

dd/mm/yyyy



09:00



Enter class here

VIEW EXCEL

SEND MAIL

Update Attendance

Choose File No file chosen

Enter class here

UPDATE EXCEL

Check overall result

Enter class here

CALCULATE

BACK

LOGOUT

4.3] Security Issues:-

There is also a problem of spoof attack in face recognition i.e. someone will show someone's image of face through their mobile phone and trick our webapp and they will mark the attendance of their friends who were not present.

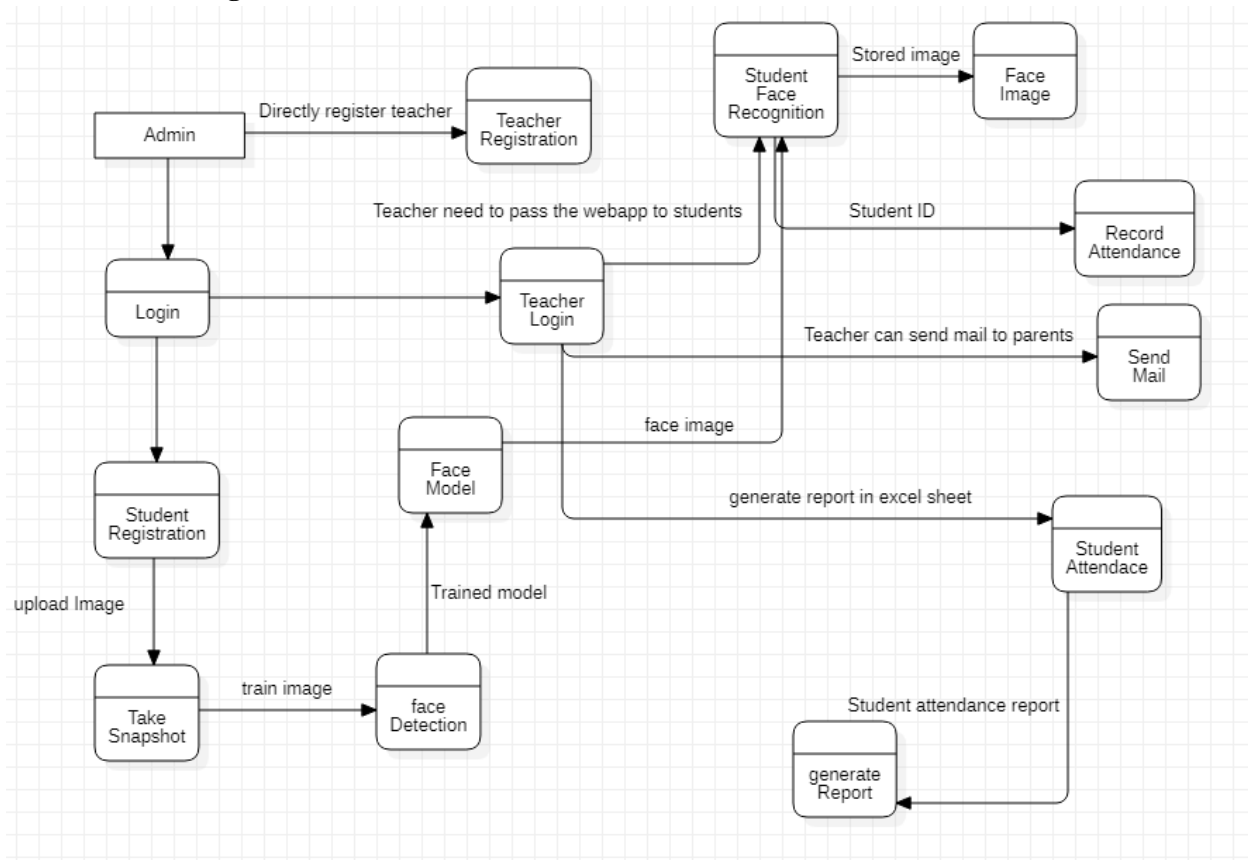
But I solved this issue using tensorflow, by training inception to detect mobile phones in an image then I used that model in the webapp as soon as student's click a snap it will first check if the face is spoof or original. To retrain inception's last layer I used 200 images of mobile phones and I feed them to tensorflow to retrain the last layer of inception.

4.4] Test case Design:-

Test Condition	Input Selection	Expected	Actual
Admin Login	username="" Password=""	username and password fields are should not be empty	Login with super user Credentials
	username="admin" Password="abc123"	Verify and redirect	
Teacher Login	username="" Password=""	username and password fields are should not be empty	. Login with Register Credentials
	username="teacher" Password="abc123"	Verify and redirect	Verify and redirect

4.5] Data Design:-

4.5.1 Schema Design:-



4.5.2 data Integrity and constraints.

Admin

Fields	Datatypes	Null	Value
username	Varchar	No	30
Password	varchar	No	20

Teacher Registration:-

Fields	Datatypes	Null	Value
Username/name	Varchar	No	50
Password	varchar	No	50
email	varchar	No	50

Student registration

Fields	Datatypes	Null	Value
Username/name	Varchar	No	30

Password	varchar	No	50
Student email	varchar	No	50
Parent email	varchar	No	50
Roll id	varchar	No	10

5] Implementation and Testing

5.1] Implementation Approach.

List of Modules:-

Admin module:-

- 1) Login module
- 2) Signup panel
- 3) Student panel:-
 - Upload file
- 4) logout

Teacher module:-

- 1) teacher login
- 2) attendance panel
- 3) view report
 - 1) view excel
 - 2) send mail
- 4) logout

Admin Module:- login module, Signup panel, Student panel, logout

admin should register the students details such as his name email address and also create training data of each student by entering his roll id and taking snaps of his or her frontal face and then webapp will automatically create model for that particular roll id and save it on server, The model which is created for each student is about 8kb in size. Admin can also register teachers using this site.

Teacher Module:- teacher login , Attendance panel, view report, logout

After teacher login you can see there are three tabs one for attendance panel ,other is view report panel . In the view report module I am having 2 sub modules. One is view excel sheet and send mail notifications module. And last is logout module.

Attendance panel:- After going to attendance tab teacher will pass phone to student and they have to show their face to mark their attendance one by one. If face is matched with the model saved for the particular roll id attendance is marked.

Note:- There is also problem that student can mark their friends attendance by showing their photo from mobile i.e. spoof attack.

For spoof detection I am using tensorflow inception model by retraining it's last layer so that it can detect mobile phones in an image.

View Report:- after going to View report tab teacher need to enter date,time and class then teacher can see the attendance and can also download the excel. Teacher can also see that how many lecture each student is present from that particular class.

Note:- To generate and manage excel I am using xlsx and xlrd and pandas.

Send mail:- teacher can also send mail after attendance for that particular day is over for particular class.

Note:- For sending email's I am using flask-mail.

Technologies:-

Hardware

Processor	Intel@ Pentium quad core
Memory	4GB
ROM	1TB

Software:-

IDE	VsCode
Language	Html+Css+JavaScript+Python+Flask
Database	Mysql
Operating system	Windows 10

Timelines:-

Start date	Duration	Module name
16 /12/2021	4 days	Admin login page
21/12/2021	5 days	Admin Signup page
27/01/22	10 days	Student panel and upload file
08/01/22	2 days	Teacher login
11/01/22	10 days	Attendance panel(face detection)
22/01/22	10 days	View report page(view excel)

03/01/22	2 days	Send mail
----------	--------	-----------

5.2] Code details and code efficiency

Admin-site

App.py

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

#import for face recongition
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

app = Flask(__name__)

# getting the location of root directory of the webapp
APP_ROOT = os.path.dirname(os.path.abspath(__file__))
print(APP_ROOT)

#connecting Mysql

conn = MySQLdb.connect(host="localhost",user="root",password="",db="login_info")

#-----linking navbars-----#
@app.route("/")
def index():
    return render_template("home.html")

@app.route("/index")
def AdminLoginPage():
    return render_template("index.html")

@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("/signup")
def AdminSignupPage():
    return render_template("Registration.html")

@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("/contact")
def AdminTaskPage():
    return render_template("contact.html")

@app.route("/Registration")
def BackToHome():
    return render_template("home.html")

@app.route("/upload")
def Logout():
    return render_template("home.html")

#-----#-----#

#database
@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 is 1):
        return render_template("upload.html")
    else:
        return render_template("index.html",msg="The username or password is incorrect")

```

```

@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("Registration.html",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 username=%s",[user])
    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('/changetask',methods=['POST'])
def changetask():
    return render_template("home.html")

@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()

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)

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("Chose 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)

if(__name__ == '__main__'):
    app.secret_key = 'secretkey'
    app.run(host="0.0.0.0",port=4555,debug=True)

```

Teachers-site

App.py

```
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 xlswriter
import pandas as pd
from glob import glob
from flask_mail import Mail, Message
from io import BytesIO
import base64
import lable_image
from flask import send_from_directory

app = Flask(__name__)

# mail settings

app.config.update(
    DEBUG = True,
    #Email settings
    MAIL_SERVER = 'smtp.gmail.com',
    MAIL_PORT = 465,
    MAIL_USE_SSL = True,
    MAIL_USERNAME = 'rameshchauhan00000007@gmail.com',
    MAIL_PASSWORD = '7715865438',
    MAIL_DEFAULT_SENDER = 'rameshchauhan00000007@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"
```



```

# getting the location of root directory of the webapp
APP_ROOT = os.path.dirname(os.path.abspath(__file__))
#APP_ROOT = os.path.abspath(os.curdir)
print('Path =',APP_ROOT)

APP_ROOT1 = APP_ROOT.split('teachers_site')
print('Path=',APP_ROOT1)

# connection with mysql database using python package MySQLdb

conn = MySQLdb.connect(host="localhost",user="root",password="",db="login_info")

@app.route("/")
def home():
    return render_template("homepage.html")

@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():
    print(APP_ROOT)
    print(APP_ROOT1[0])
    user = str(request.form["user"])
    session['user'] = user
    paswd = str(request.form["password"])
    username = user.split(".",1)[0]
    username = str(username)
    print(username)
    print(type(username))
    cursor = conn.cursor()
    result = cursor.execute("SELECT * from teacher_login where binary username=%s and binary
password=%s",[user,paswd])
    if(result is 1):
        return render_template("task.html",uname=username)
    else:
        return render_template("homepage.html",title="Faculty Login",msg="The username or password")

```

```

@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('/upload_redirect', methods=['POST'])
def upload_redirect():
    if os.path.isfile(APP_ROOT+"\\image.jpeg"):
        os.remove(APP_ROOT + "\\image.jpeg")
    return render_template("upload.html")

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

@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

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 through 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 = xlswriter.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
    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')
    result.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")

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

# view route to download excel files

@app.route('/view',methods=['POST'])
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")
@app.route('/excel/<path:filename>', methods=['POST'])
def download(filename):
    return send_from_directory(directory='excel', filename=filename)

# route to send emails to parents and students

@app.route('/send_mail',methods=['POST'])
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>"

#.....update attendance .....#
@app.route('/update',methods=['POST'])
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")

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

```

```

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 repetitive 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)

def viewfinal(final):
    test_append = str(session.get('test_append'))
    final_path = str(session.get('final'))
    df = pd.read_excel(final_path)
    df.index += 1
    return render_template("files.html",msg=final,course=test_append,df=df)

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

@app.route('/logout',methods=['POST'])
def logout():
    return render_template("index.html",title="Faculty Login",msg1="Logged out please login again")

@app.route('/hello',methods=['POST'])
def hello():
    data_url = request.values['imageBase64']
    data_url= data_url[22:]
    im = Image.open(BytesIO(base64.b64decode(data_url)))
    print(type(im))
    im.save('image.jpeg')
    filepath = APP_ROOT + "\\\" + "image.jpeg"
    var = lable_image.function(filepath)
    print(var)
    for i in range(len(var)):
        if(var[i] > 0.8):
            os.remove(filepath)
    return "

```



```
if(__name__ == '__main__'):
    app.secret_key = 'super secret key'
    app.run(host="0.0.0.0",port=4555,debug=True)
```

Label-image.py

```
# This requires retrained_labels.txt and retrained_graph.pb files which are available at this link.
# https://drive.google.com/open?id=1TMCTLEqekAh1zMwdSfWfXMKNKYyfgIYB
# https://drive.google.com/open?id=1d93B2ehbxrpbAhm7OqtvvgDT5Mszk4_jc

# Change both file path accordingly.

import tensorflow as tf,sys
import os

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

print('path=',APP_ROOT)

def function(xyz):
    output = []

    image_path = xyz

    image_data = tf.io.gfile.GFile(image_path,'rb').read()

    label_lines = [line.rstrip() for line
                    in tf.io.gfile.GFile(APP_ROOT + "\\tensor\\tf_files\\retrained_labels.txt")]

    with tf.io.gfile.GFile(APP_ROOT + "\\tensor\\tf_files\\retrained_graph.pb", 'rb') as f:
        graph_def = tf.compat.v1.GraphDef()
        graph_def.ParseFromString(f.read())
        _ = tf.import_graph_def(graph_def,name=")

    with tf.compat.v1.Session() as sess:
        softmax_tensor = sess.graph.get_tensor_by_name('final_result:0')
        predictions = sess.run(softmax_tensor, \
                                {'DecodeJpeg/contents:0': image_data})

        top_k = predictions[0].argsort()[-len(predictions[0]):][::-1]
        i = 0

        for node_id in top_k:
            human_string = label_lines[node_id]
```

```
score = predictions[0][node_id]
print('%s (score = %.5f)' % (human_string,score))
output.append(score)
return output
```

5.3] Testing approach:-

A test approach is the test strategy implementation of a project, defines how testing would be carried out.

5.3.1 Unit Testing:-

All the modules like registration, login, image snapshot, face detection, face cam recognition and attendance record page will be test individual.

5.3.2 Integration Testing:-

In integration testing we are checking after successful registration teacher also able login into the webapp. Here we are checking once admin register teacher and student details will get store into Mysql database. Basically here we are checking all the modules are connected or not with each other.

5.3.3 System Testing:-

Here we are checking whatever input we are giving to the system and we are getting according output or not. Also The whole system working properly or not.

5.3.4 Validating Testing:-

Validation testing can be defined in many ways, but a simple definition is that validation succeeds when the software function in a manner that can be reasonably expected by the Users.

5.3.5 Output Testing:-

After performing the validation testing the next test is output testing of the proposed system since no system could be useful if it does not produce the required output in the specified format. Asking the user about the format required by them tests the outputs generated or displayed by the system under consideration. The output format on the screen is found to be correct as the format was designed in the system phase according to the user's needs. Hence, output testing does not result in any correction in the system.

5.3.6 User Acceptance Testing:-

User acceptance of a system is the key factory for the success of any system. The system under consideration is tested for user acceptance by constantly keeping in touch with the perspective system.

5.4] Modifications and Improvement

After testing and finding all errors, exceptions and bugs, errors were solved by troubleshooting, exceptions were handled by try catch block at the correct places avoiding redundancy of code from model class. Bugs can never be removed completely but most of them are solved and remaining are under development.

6] RESULTS AND DISCUSSION

6.1] User Documentation:-

There are mainly two webapps for this project one is say admin site and other one is teacher's site.

The whole concept is at the time of admission to college or school admin should register the students details such as his name email address and also create training data of each student by entering his roll id and taking snaps of his or her frontal face and then webapp will automatically create model for that particular roll id and save it on server, The model which is created for each student is about 8kb in size. Admin can also register teachers using this site.

Now using teacher's site (It will be used when teacher will actually enter the class), teacher has to login first and then after clicking on attendance tab there will be no back button as teacher will pass on the phone to student.

Student's will then have to just click a snap enter class and roll id and press enter to mark their attendance.

After that there is also a problem of spoof attack in face recognition i.e. someone will show someone's image of face through their mobile phone and trick our webapp and they will mark the attendance of their friends who were not present.

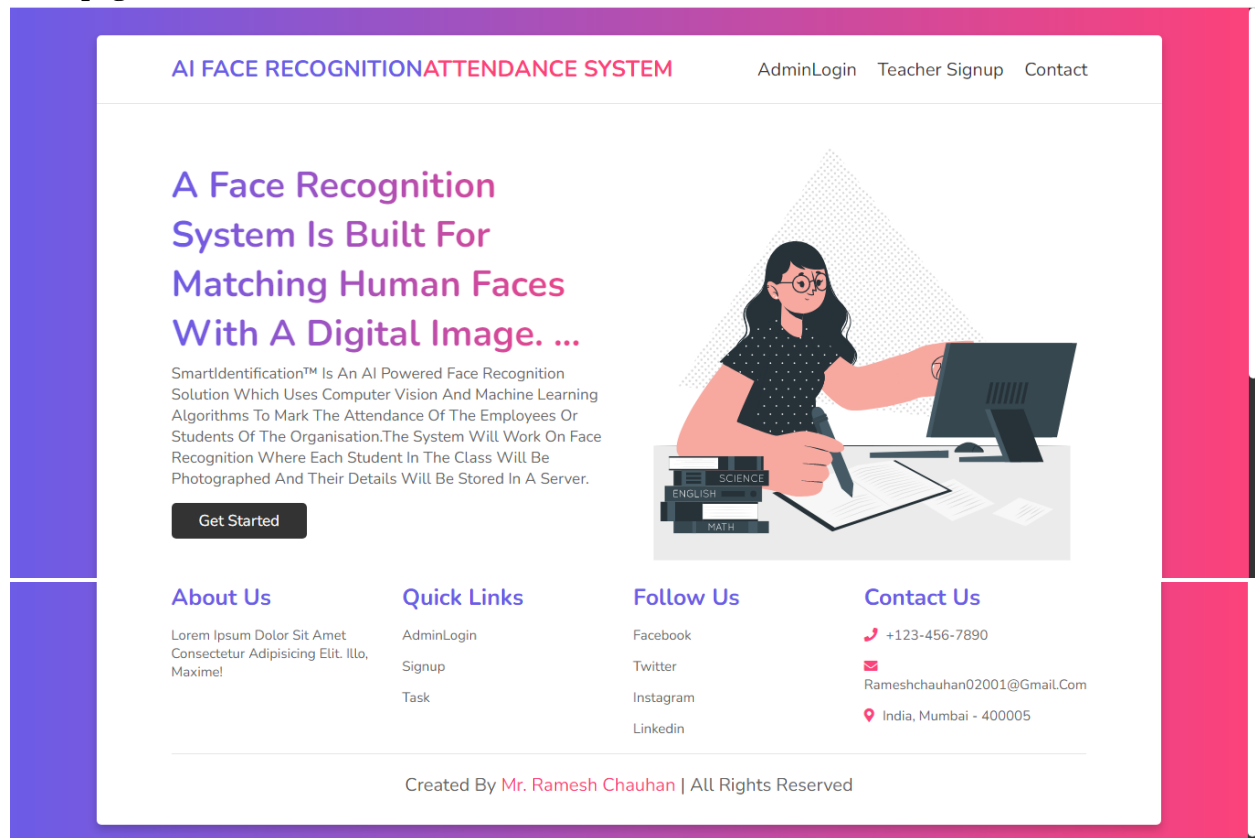
But I solved this issue using tensorflow, by training inception to detect mobile phones in an image then I used that model in the webapp as soon as student's click a snap it will first check if the face is spoof or original. To retrain inception's last layer I used 200 images of mobile phones and I feed them to tensorflow to retrain the last layer of inception. To do this follow <https://codelabs.developers.google.com/codelabs/tensorflow-for-poets/#0> To download 200 images at once use fatkun-batch-download chrome extension.

Now after student's part is done. Teacher can then login again and then go to report's tab to see attendance. Here there are several options I have given.

If teacher want to see today's attendance, just select date and time to see the attendance. And there is also an option to download the attendance sheet in excel form and then again reupload it after making any changes if sometime required by the teacher. And the teacher can also see total attendance for his or her lecture. so that they can analyze how many lectures each student from particular class had attended so far.

One additional feature is that teacher can send email for the attendance marked to all the parents as well as students by selecting class and clicking on send mail button.

Admin Site Homepage



Teacher Registration page:-

AI Face Recognition Attendance System

REGISTRATION FORM
Enter Your Personal Data

USERNAME:

Name

EMAIL:

Email

PASSWORD:


Password

[BackToHome](#)

Contact Page:-

AI FACE RECOGNITION ATTENDANCE SYSTEM

CONTACT US



About Us

Lorem Ipsum Dolor Sit Amet Consectetur
Adipiscing Elit. Illo, Maxime!

Follow Us

Facebook
Twitter
Instagram
Linkedin


Contact Us

+123-456-7890
Rameshchauhan02001@Gmail.Com
India, Mumbai - 400005

Created By **Mr. Ramesh Chauhan** | All Rights Reserved


Admin login Page

AI FACE RECOGNITION ATTENDANCE SYSTEM



Student Registration and Model Training Page

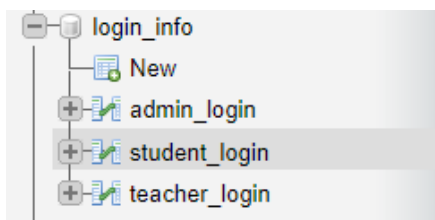
AI FACE RECOGNITION ATTENDANCE SYSTEM



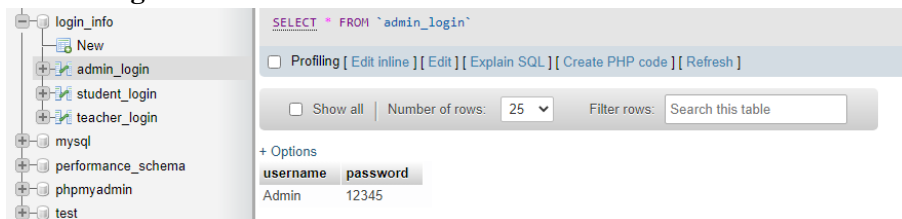
FILE UPLOAD

No file chosen

Database:-



Admin Login



Teacher Registration:-

login_info	SELECT * FROM `teacher_login`
New	
admin_login	
student_login	
teacher_login	
mysql	
performance_schema	
phpmyadmin	
test	

+ Options		
username	password	email
Ramesh	12345	realcricketchamp@gma
Jay	jay1234	jay@gmail.com
Sunil	sunil123	sunil@gmail.com
Uzair	uzair1234	Uzair@gmail.com
Ashok	ashok12345	ashok25@gmail.com
vishal	123456789	vishal@gmail.com
sahil	sahil12345	sahil@gmail.com
bertilla	12345	bertilla@gmail.com

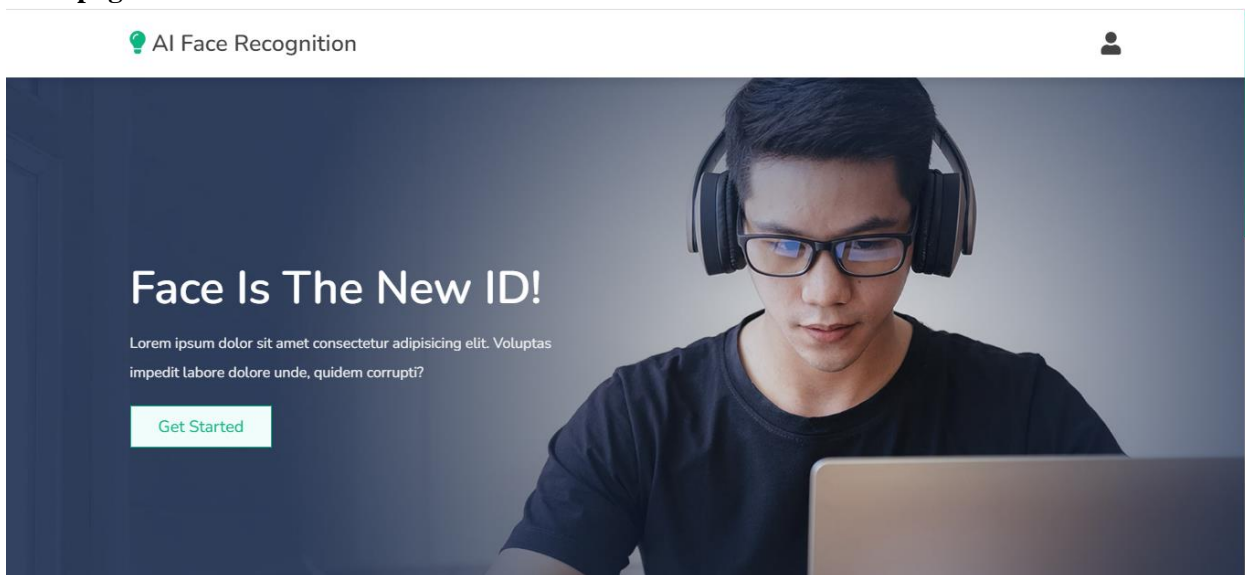
Student registration:-

Recent	Favorites
New	
information_schema	
login_info	
New	
admin_login	
student_login	
teacher_login	
mysql	
performance_schema	
phpmyadmin	
test	

+ Options			
username	student_email	parent_email	roll_id
daksh Mandal	dakshmandal@gmail.co	dakshmandal@gmail.co	19bit029
Nidhi	nidhi@gmail.com	raju@gmail.com	19bit010
Sunil Rathod	sunil123@gmail.com	sunil456@gmail.com	19bit0039
Ramesh	rameshchauhan02001@g	rameshchauhan007@gma	19bit007
vikas	vikas@gmail.com	vikas12@gmail.com	19bit007
jhon	jhon@gmail.com	jhon123@gmail	19bit067
Vinc	vinc@gmail.com	vinc12@gmail.com	19bit010
vikesh	vikesh@gmail.com	vikesh12@gmail.com	05
Ramesh chauhan	ramesh45@gmail.com	ramesh78@gmail.com	65
siraj	siraj@gmail.com	siraj789@gmail.com	90
kishan	kishan@gmail.com	kishan12@gmail.com	80
Ram	ram@gmail.com	ram90@gmail.com	90
krish	krish@gmail.com	krish90@gmail.com	40
vikas sharma	vk@gmail.com	vk80@gmail.com	09
harsh	harsh@gmail.com	harsh1234@gmail.com	150
Akash	akash56@gmail.com	akash89@gmail.com	63

Teachers Site:

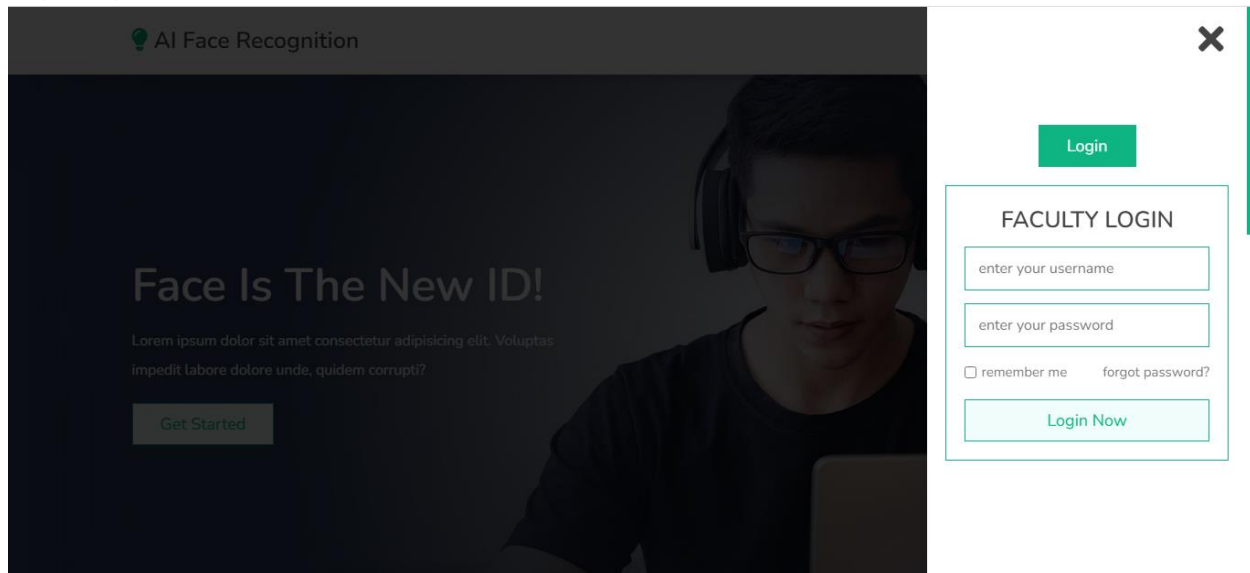
Homepage:



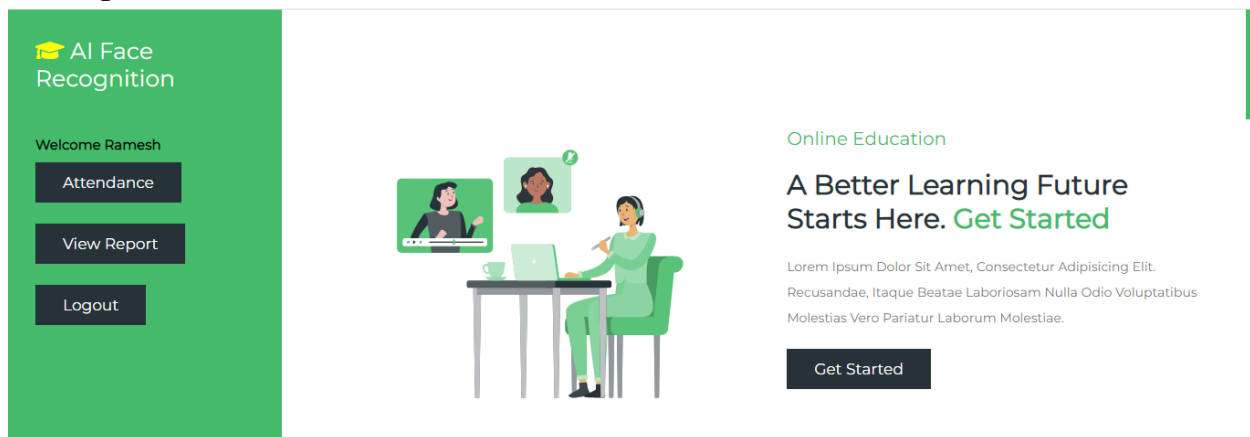
Our Popular Subjects



LoginPage:



TaskPage:



AI Face Recognition

Welcome Ramesh

Attendance

View Report

Logout

+450
Students

+70
Courses

+199
Teachers

+520
Subjects

AI Face Recognition

Welcome Ramesh

Attendance

View Report

Logout

OUR CATEGORY

OUR TOP CATEGORY

Programing

Lorem Ipsum Dolor Sit Amet,
Consectetur Adipisicing Elit. Omnis,
Reiciendis.

Graphic Design

Lorem Ipsum Dolor Sit Amet,
Consectetur Adipisicing Elit. Omnis,
Reiciendis.

Marketing

Lorem Ipsum Dolor Sit Amet,
Consectetur Adipisicing Elit. Omnis,
Reiciendis.

Business Management

Lorem Ipsum Dolor Sit Amet,
Consectetur Adipisicing Elit. Omnis,
Reiciendis.

Music & Dance

Lorem Ipsum Dolor Sit Amet,
Consectetur Adipisicing Elit. Omnis,
Reiciendis.

Science & Biology

Lorem Ipsum Dolor Sit Amet,
Consectetur Adipisicing Elit. Omnis,
Reiciendis.

AI Face Recognition

Welcome Ramesh

Attendance

View Report

Logout

CLIENT REVIEWS

WHAT THEY SAY?

John Deo

Lorem Ipsum, Dolor Sit Amet
Consectetur Adipisicing Elit.
Dignissimos Repellat Sunt A
Temporibus Suscipit Distinctio
Maiores Iste Tenetur Voluptate
Nostrum.

★★★★☆

John Deo

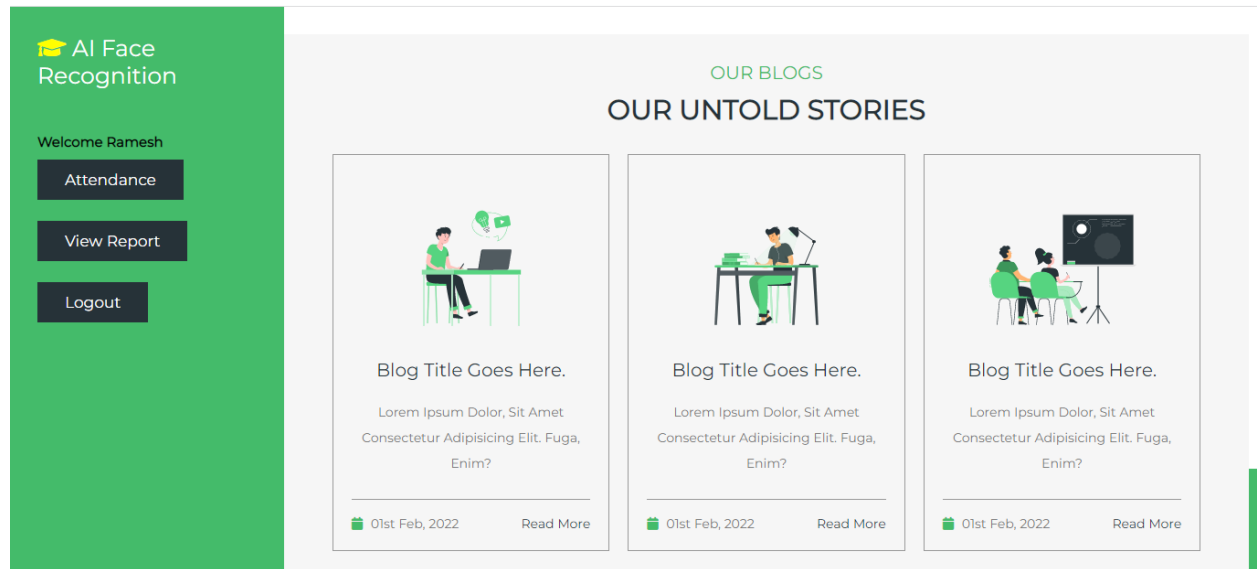
Lorem Ipsum, Dolor Sit Amet
Consectetur Adipisicing Elit.
Dignissimos Repellat Sunt A
Temporibus Suscipit Distinctio
Maiores Iste Tenetur Voluptate
Nostrum.

★★★★☆

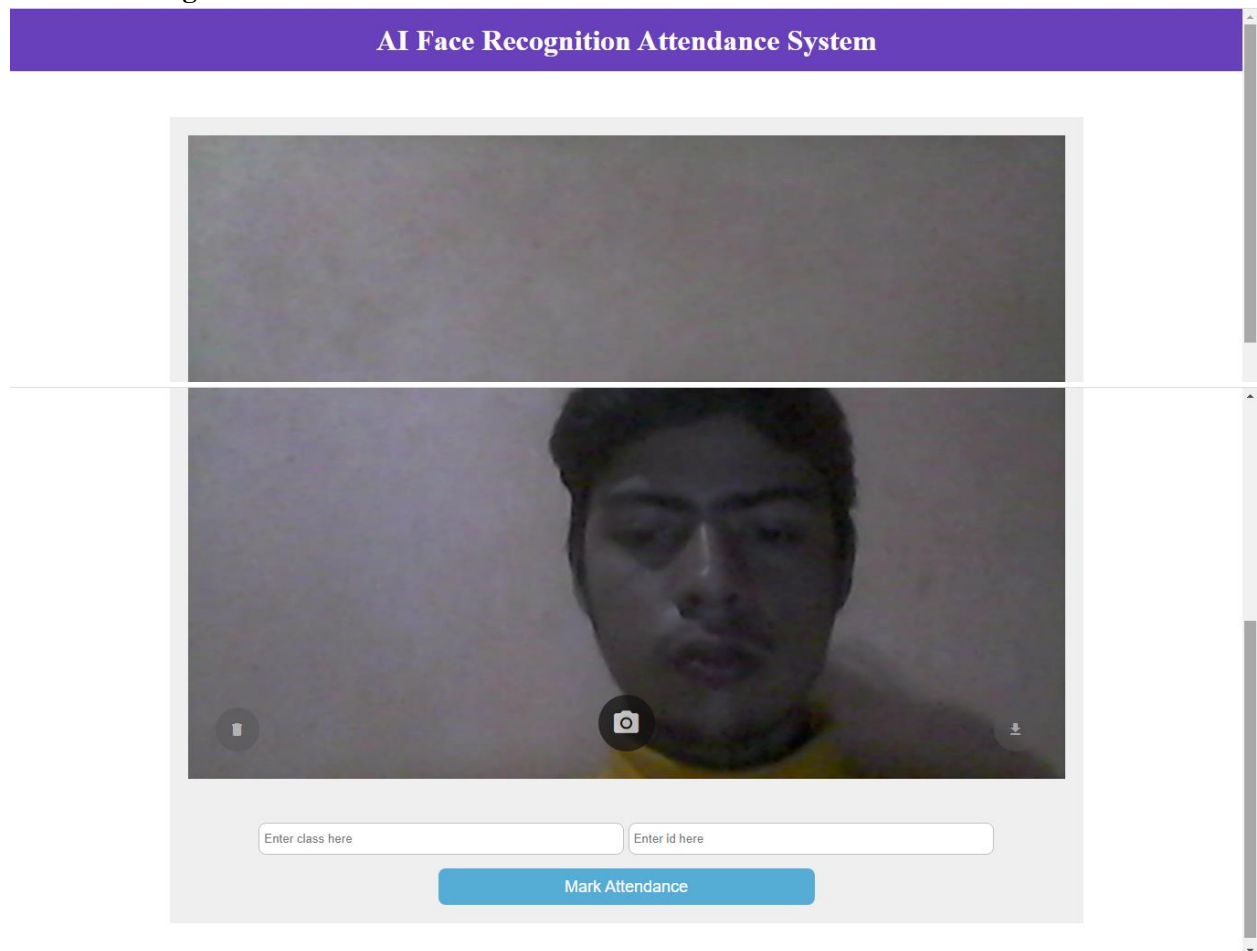
John Deo

Lorem Ipsum, Dolor Sit Amet
Consectetur Adipisicing Elit.
Dignissimos Repellat Sunt A
Temporibus Suscipit Distinctio
Maiores Iste Tenetur Voluptate
Nostrum.

★★★★☆




AttendancePage:





View Report Page:

AI FACE RECOGNITION ATTENDANCE SYSTEM



Today's Result

dd/mm/yyyy 

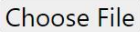
09:00 

Enter class here

VIEW EXCEL

SEND MAIL

Update Attendance

Choose File  No file chosen

Enter class here

UPDATE EXCEL

Check overall result

Enter class here


CALCULATE

BACK


LOGOUT


Excel sheet:-

AI FACE RECOGNITION ATTENDANCE SYSTEM



Today's Result

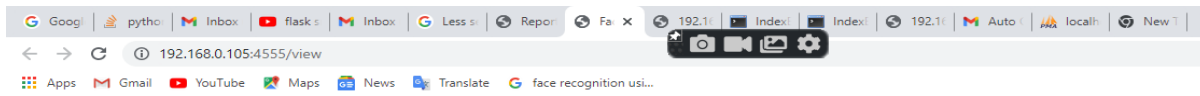
29/01/2022 

16:22 

AI

VIEW EXCEL

SEND MAIL



Attendance for 2022-01-29@16hrs

Roll Id
1 150

[Download C:\Users\Ramesh\Desktop\TYIT-PROJECT\teachers_site\excel\AI\Ramesh\2022-01-29@16hrs.xlsx](#)

> This PC > Desktop > TYIT-PROJECT > teachers_site > excel > AI > Ramesh

Name	Date modified	Type	Size
final	01/02/2022 17:41	File folder	
2022-01-29@16hrs.xlsx	29/01/2022 16:22	Microsoft Office E...	6 KB
2022-02-01@16hrs.xlsx	01/02/2022 17:36	Microsoft Office E...	6 KB

	A	B
1	Roll Id	
2	150	
3		

Final xlsx:-


> This PC > Desktop > TYIT-PROJECT > teachers_site > excel > AI > Ramesh > final

Name	Date modified	Type	Size
final.xlsx	01/02/2022 17:40	Microsoft Office E...	6 KB

	A	B	C
1	Roll Id	Total presenty	
2	150	2	
3			

Send mail:-

AI FACE RECOGNITION ATTENDANCE SYSTEM



Today's Result

29/01/2022

16:22

AI

VIEW EXCEL

SEND MAIL

mail sent

Compose

Inbox 3

Starred

Snoozed

Sent

Drafts

More

Meet

Auto Generated

rameshchauhan00000007@gmail.com

to harsh, harsh1234

Hi. 150 is present for the lecture of Prof. Ramesh, which is held on 2022-01-29@16hrs

Reply

Reply to all

Forward

12:06 (14 minutes ago)

1 of 3

Reading list

Compose

Inbox 3

Starred

Snoozed

Sent

Drafts

More

Meet

Auto Generated

rameshchauhan00000007@gmail.com

to harsh, harsh1234

Hi. 150 is present for the lecture of Prof. Ramesh, which is held on 2022-01-29@16hrs

Reply

12:06 (15 minutes ago)

1 of 3

Reading list

from: rameshchauhan00000007@gmail.com

to: harsh@gmail.com, harsh1234@gmail.com

date: 7 Feb 2022, 12:06

subject: Auto Generated

mailed-by: gmail.com

7] Conclusion:-

7.1] Conclusion:-

The goal of the project was to build a facial recognition system for student's attendance. The result of the project was a successful prototype of a facial recognition system where the admin can create a teacher account and add students and their information to the database. Teachers then can log in to the system and take attendance of the student. The student's face is detected by a camera and attendance is recorded in the database. Teachers and admin could see the attendance report of the students. From experiment, I noticed the face recognition was sensitive to face background, light, and head orientations. This technique described the accurate and efficient method of automatic attendance in the classroom which could replace the traditional method. An automatic attendance has many advantages, most of the existing systems are time consuming and require semi manual interference from lecturers, and our system seeks to solve these issues by using face recognition in the process to save the time and labor. And No need for installing complex hardware for taking the attendance in classroom, all we need is a camera and laptop. We used algorithms that can detect and recognize faces in the image.

7.2. Limitation:-

- The identical twin issue. I am having a challenge of getting different recognitions for twins.

7.3. Future Scope of the Project:-

- Automatic attendance system can be improved by increasing the number of features which can be extracted to increase accuracy of face recognition.
- Automating the whole process so that we have digital environment.
- Use the Live face Recognition to recognize each individual and mark their attendance automatically.
- Utilizes video and image processing to provide input to the system
- Automate update in the attendance sheet without human intervention.
- To keep the student update with their attendance ratio.