ATTENDANCE SYSTEM WITH FACE RECOGNITION

An Internship report submitted in partial fulfillment of the requirement for the Award of the Degree

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE AND ENGINEERING

by

E Kiran Kumar	**(160717733023)**
Aieti Sreekar	**(160717733032)**
C Veeresh	**(160717733011)**

Under the Guidance of

Dr. P Lavanya, Head of Department, Dept. of CSE

Department of Computer Science and Engineering

Methodist College of Engineering and Technology

King Koti, Abids, Hyderabad-500001.

2020-2021

Methodist College of Engineering and Technology,
King Koti, Abids, Hyderabad-500001,

Department of Computer Science and Engineering

DECLARATION BY THE CANDIDATES

I, **E Kiran Kumar (160717733023), Aieti Sreekar (160717733032)** and **C Veeresh (160717733011),** students of Methodist College of Engineering and Technology, pursuing Bachelor's degree in Computer Science and Engineering, hereby declare that this internship report entitled "Attendance System with Face Recognition", carried out under the guidance of **Dr. P Lavanya** submitted in partial fulfillment of the requirements for the degree of Bachelor of Engineering in Computer Science. This is a record work carried out by us and the results embodied in this report have not been reproduced/copied from any source.

E Kiran Kumar	**(160717733023)**
Aieti Sreekar	**(160717733032)**
C Veeresh	**(160717733011)**

Methodist College of Engineering and Technology,
King Koti, Abids, Hyderabad-500001.

Department of Computer Science and Engineering

CERTIFICATE BY THE SUPERVISOR

This is to certify that this Internship report entitled "**ATTENDANCE SYSTEM FACE RECOGNITION**" being submitted by **E Kiran Kumar (160717733023), Aieti Sreekar (160717733032)** and **C Veeresh (160717733011)** submitted in partial fulfillment of the requirements for the degree of Bachelor of Engineering in Computer Science and Engineering, during the academic year 2020-2021, is a bona fide record of work carried out by them.

Dr. P Lavanya,
Professor, HOD
Dept. of CSE

Date:

Methodist College of Engineering and Technology,

King Koti, Abids, Hyderabad-500001.

Department of Computer Science and Engineering

CERTIFICATE BY HEAD OF THE DEPARTMENT

This is to certify that this Internship report **“ATTENDANCE SYSTEM FACE RECOGNITION”** by **E Kiran Kumar (160717733023), Aieti Sreekar (160717733032)** and **C Veeresh (160717733011)** submitted in partial fulfillment of the requirements for the degree of Bachelor of Engineering in Computer Science and Engineering of the Osmania University, Hyderabad, during the academic year 2020-2021, is a bona fide record of work carried out by them.

Dr. P Lavanya,
Professor, HOD
Dept. of CSE

Date:

ACKNOWLEDGEMNTS

We would like to express our sincere gratitude to my Internship guide **Dr. P Lavanya, Professor,** Head of Department**,** Dept. of CSE**,** for giving us the opportunity to work on this topic. It would never be possible for us to take this project to this level without his innovative ideas and his relentless support and encouragement.

We would like to thank our Internship coordinator **Dr. M. Sharada, Professor,** who helped us by being an example of high vision and pushing towards greater limits of achievement.

We would like to express a deep sense of gratitude towards the **Dr. Prabhu G. Benakop, Principal, Methodist College of Engineering and Technology,** for always being an inspiration and for always encouraging us in every possible way.

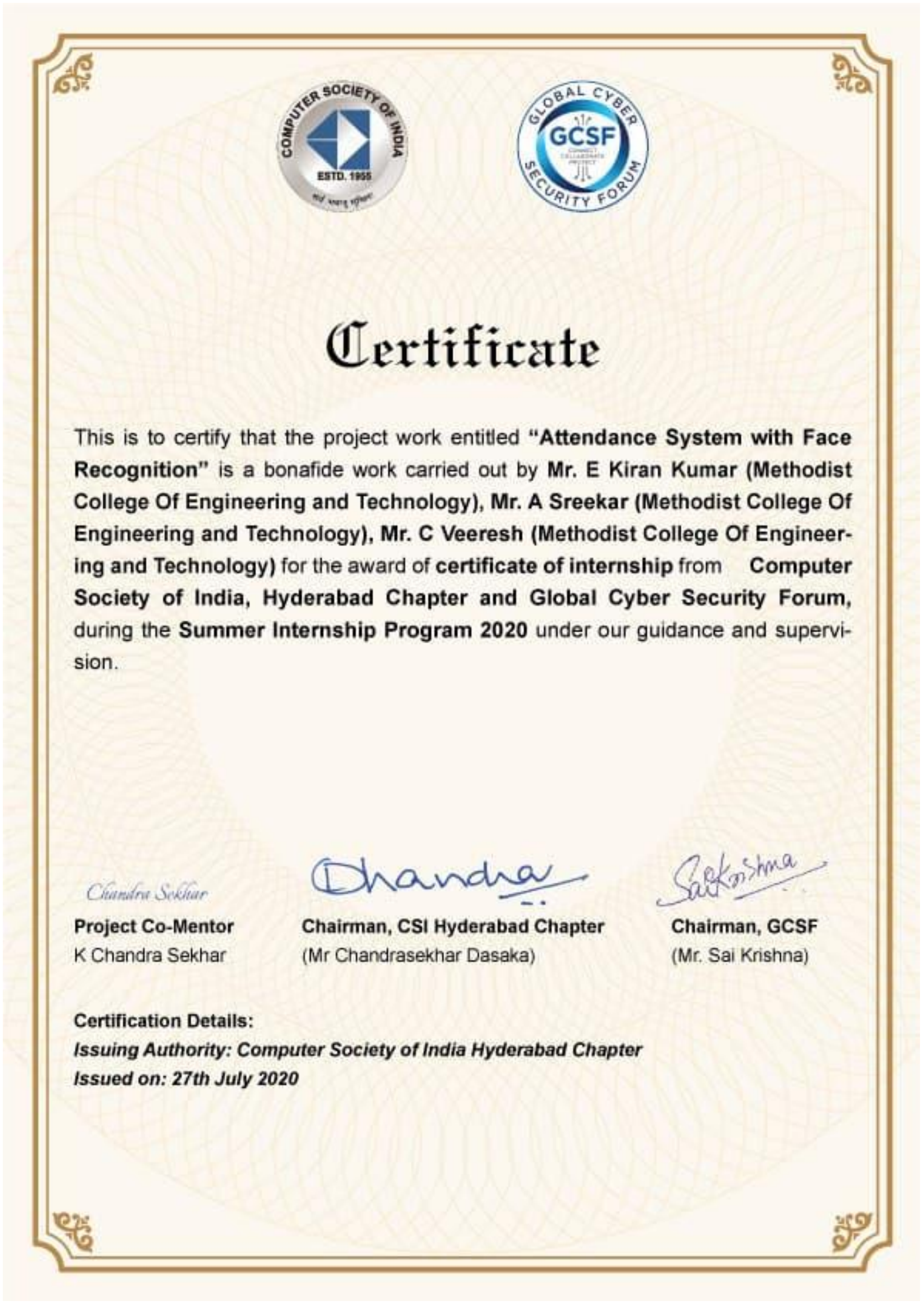
We would like to express a deep sense of gratitude towards the **Dr. Lakshmipathi Rao, Director, Methodist College of Engineering and Technology,** for always being an inspiration and for always encouraging us in every possible way.

We are indebted to the Department of Computer Science & Engineering and Methodist College of Engineering and Technology for providing us with all the required facility to carry our work in a congenial environment. We extend our gratitude to the CSE Department staff for providing us to the needful time to time whenever requested.

We would like to thank our parents for allowing us to realize our potential, all the support they have provided us over the years was the greatest gift anyone has ever given us and also for teaching us the value of hard work and education. Our parents has offered us with tremendous support and encouragement, thanks to our parents for all the moral support and the amazing opportunities they have given us over the years.

INDEX

List of Figures



Certificate

This is to certify that the project work entitled **"Attendance System with Face Recognition"** is a bonafide work carried out by **Mr. E Kiran Kumar (Methodist College Of Engineering and Technology), Mr. A Sreekar (Methodist College Of Engineering and Technology), Mr. C Veeresh (Methodist College Of Engineering and Technology)** for the award of **certificate of internship** from **Computer Society of India, Hyderabad Chapter and Global Cyber Security Forum,** during the **Summer Internship Program 2020** under our guidance and supervision.

Chandra Sekhar	[illegible]	[illegible]
Project Co-Mentor	**Chairman, CSI Hyderabad Chapter**	**Chairman, GCSF**
K Chandra Sekhar	(Mr Chandrasekhar Dasaka)	(Mr. Sai Krishna)

Certification Details:
Issuing Authority: Computer Society of India Hyderabad Chapter
Issued on: 27th July 2020

ABSTRACT

Earlier attendance was taken manually, which is a time consuming process and it’s a huge effort for the teachers every day. We have developed an application “ATTENDANCE SYSTEM WITH FACE RECOGNITION” which recognize faces of the students and generate the attendance automatically. In this project, we have used OpenCV library which is an Image Processing and Machine Learning based approach enables the system to detect and recognize the faces. This process contains three steps where first is to detect the face then extract the features and finally recognize. This project aims to reduce the wastage of time and efforts in taking the attendance

1.INTRODUCTION

Human face recognition is an important part of biometric verification. It is widely used for verification purposes, especially if students attend to lectures. There is a lot of time lost in classical attendance confirmations. In order to solve this time loss, an Attendance System with Face Recognition has been developed which automatically tracks the attendance status of the students. The Attendance System with Face Recognition performs daily activities of the attendance analysis which is an important aspect of face recognition task. By doing this in an automated manner, it saves time and effort in classrooms and meetings. The paired face is then used to mark course attendance. By using the Attendance System with Facial Recognition, the efficiency of lecture times' utilization will be improved. Additionally, it will be possible to eliminate mistakes on attendance sheets.

1.1 Existing System:

The existing system was took attendance with manually by lectures or fingerprint using ID. This took time to done the attendance separately

Disadvantages

- Time consuming process
- In the manual process of attendance, by chances to mismatch attendance
- Manual process has additional work like writing information about students in the record
- Chances are less to check the attendance

1.2 Proposed System:

The Proposed System is taking Attendance with Face Detection by using web cameras. This is easy process to take attendance and will done automatically.

Advantages

- Time consuming less and easy work
- No chance to mismatch the attendance
- There is no additional work, attendance will add automatically by detecting faces in the class

- Anywhere anytime student can check their attendance by login into the attendance application

2.LITERATURE SURVEY

2.1 Project Literature

Literature survey is the most important step in software development process. Before developing the tool, it is necessary to determine the time factor, economy and company strength. Once these things are satisfied, then next step is to determine which operating system and language can be used for developing the tool. Once the programmers start building the tool the programmers need lot of external support. This support can be obtained from senior programmers, from book or from websites. Before building the system, the above consideration are taken into account for developing the proposed system.

2.2 Introduction to Python

Python is a general purpose, dynamic, high level and interpreted programming language. It supports Object Oriented Programming approach to develop applications. It is simple and easy to learn and provides lots of high-level data structures. Python is easy to learn yet powerful and versatile scripting language which makes it attractive for Application Development. Python's system and dynamic typing with its interpreted nature, makes it an ideal language for scripting and rapid application development.

Python supports multiple programming pattern, including object oriented, imperative and functional or procedural programming styles. Python is not intended to work on special area such as web programming. That is why it is known as multipurpose because it can be used with web, enterprise, 3D CAD etc. We don't need to use data types to declare variable because it is dynamically typed so we can write a=10 to assign an integer value in an integer variable. Python makes the development and debugging fast because there is no compilation step included in python development and edit-test-debug cycle is very fast

Python Features

1) Easy to Learn and Use Python is easy to learn and use. It is developer-friendly and high-level programming language.

2) Expressive Language Python language is more expressive means that it is more understandable and readable.

3) Interpreted Language Python is an interpreted language i.e. interpreter executes the code line by line at a time. This makes debugging easy and thus suitable for beginners.

4) Cross-platform Language Python can run equally on different platforms such as Windows, Linux, Unix and Macintosh etc. So, we can say that Python is a portable language.

5) Free and Open Source Python language is freely available at official web address. The source-code is also available. Therefore, it is open source.

6) Object-Oriented Language

Python supports object-oriented language and concepts of classes and objects come into existence.

7)Extensible

It implies that other languages such as C/C++ can be used to compile the code and thus it can be used further in our python code.

8) Large Standard Library

Python has a large and broad library and provides rich set of module and functions for rapid application development.

9) GUI Programming Support

Graphical user interfaces can be developed using Python.

10) Integrated

It can be easily integrated with languages like C, C++, JAVA etc.

Python Application Area

1) Web Applications

2) Desktop GUI Applications

3) Software Development

4) Scientific and Numeric

5) Business Applications

6) Console Based Application

7) Audio or Video based Applications

8) 3D CAD Applications

9) Enterprise Applications

10) Applications for Images

Simple Demo Program

```
print ("hello world by python!")
#Execute this example by using following command.
Python hello.py •
                    OR
>>> a="Welcome To Python"
>>> print a
     Welcome To Python
```

2.2.1 Python Technology

Python's design philosophy emphasizes code readability with its notable use of significant whitespace. Its language constructs and object-oriented approach aim to help programmers write clear, logical code for small and large-scale projects.

Python is dynamically typed and garbage-collected. It supports multiple programming paradigms, including structured (particularly, procedural), object-oriented, and functional programming. Python is often described as a "batteries included" language due to its comprehensive

Python interpreters are available for many operating systems. A global community of programmers develops and maintains CPython, an open source[34] reference implementation. A non-profit organization, the Python Software Foundation, manages and directs resources for Python and CPython development.

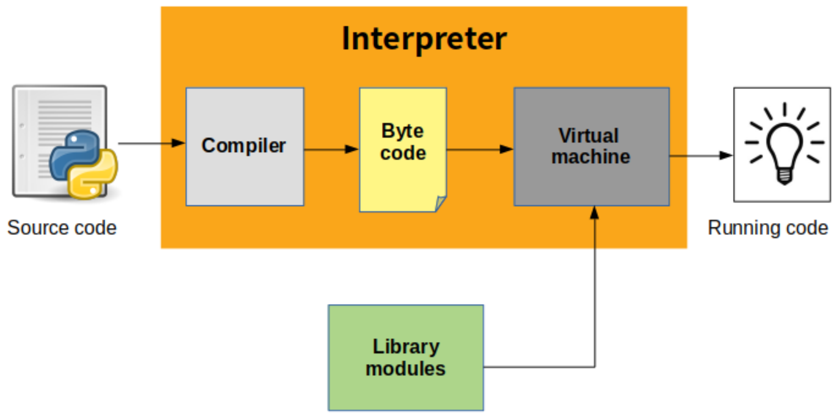


Fig.2.1 Compiling and Interpreting stages

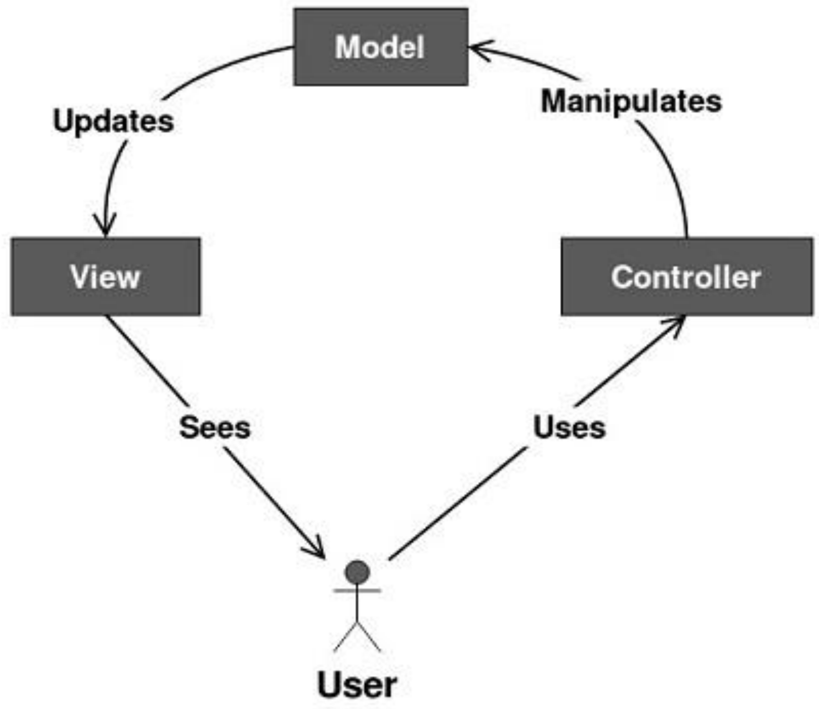
python language is interpreted. But that is half correct the python program is first compiled and then interpreted. The compilation part is hidden from the programmer thus, many programmers believe that it is an interpreted language. The compilation part is done first when we execute our code and this will generate byte code and internally this byte code gets converted by the python virtual machine

(p. v. m)

2.2.2 MVC Architecture

The three components of the MVC pattern are **decoupled** and they are responsible for different things. Model View Controller is the most commonly used design pattern. Developers find it easy to implement this design pattern.

Following is a basic architecture of the Model View Controller



Model

It manages the data and defines rules and behaviour. It represents the business logic of the application. The data can be stored in the Model itself or in a database (only the Model has access to the database).

View

It presents the data to the user. A View can be any kind of output representation: a HTML page, a chart, a table, or even a simple text output. A View should never call its own methods; only a Controller should do it.

Controller

It accepts user's inputs and delegates data representation to a View and data handling to a Model. Since Model, View and Controller are **decoupled**, each one of the three can be extended, modified and replaced without having to rewrite the other two.

Advantages

- Multiple developers can work simultaneously on the model, controller and views.
- MVC enables logical grouping of related actions on a controller together. The views for a specific model are also grouped together.
- Models can have multiple views.

Disadvantages

- The framework navigation can be complex because it introduces new layers of abstraction and requires users to adapt to the decomposition criteria of MVC.
- Knowledge on multiple technologies becomes the norm. Developers using MVC need to be skilled in multiple technologies.

2.2.3Tkinter

The tkinter package ("Tk interface") is the standard Python interface to the Tk GUI toolkit. Both Tk and **tkinter** are available on most Unix platforms, as well as on Windows systems. (Tk itself is not part of Python; it is maintained at Active State.)

Running `python -m tkinter` from the command line should open a window demonstrating a simple Tk interface, letting you know that **tkinter** is properly installed on your system, and also showing what version of Tcl/Tk is installed, so you can read the Tcl/Tk documentation specific to that version.

2.2.3.1Tkinter Modules

Most of the time, tkinter is all you really need, but a number of additional modules are available as well. The Tk interface is located in a binary module named _tkinter. This module contains the low-level interface to Tk, and should never be used directly by application programmers. It is usually a shared library (or DLL), but might in some cases be statically linked with the Python interpreter.

In addition to the Tk interface module, tkinter includes a number of Python modules, tkinter.constants being one of the most important. Importing tkinter will automatically import tkinter.constants, so, usually, to use Tkinter all you need is a simple import statement:

```
import tkinter
```

Or, more often:

```
from tkinter import *
```

class tkinter.**Tk**(*screenName=None*, *baseName=None*, *className='Tk'*, *useTk=0*)

The Tk class is instantiated without arguments. This creates a toplevel widget of Tk which usually is the main window of an application. Each instance has its own associated Tcl interpreter.

tkinter.**Tcl**(*screenName=None*, *baseName=None*, *className='Tk'*, *useTk=0*)

The Tcl() function is a factory function which creates an object much like that created by the Tk class, except that it does not initialize the Tk subsystem. This is most often useful when driving the Tcl interpreter in an environment where one doesn't want to create extraneous toplevel windows, or where one cannot (such as Unix/Linux systems without an X server). An object created by the Tcl() object can have a Toplevel window created (and the Tk subsystem initialized) by calling its loadtk() method.

2.2.4. Libraries Specific to Project

The Python Language Reference describes the exact syntax and semantics of the Python language, this library reference manual describes the standard library that is distributed with Python. It also describes some of the optional components that are commonly included in Python distributions.

Python's standard library is very extensive, offering a wide range of facilities as indicated by the long table of contents listed below. The library contains built-in modules (written in C) that provide access to system functionality such as file I/O

that would otherwise be inaccessible to Python programmers, as well as modules written in Python that provide standardized solutions for many problems that occur in everyday programming. Some of these modules are explicitly designed to encourage and enhance the portability of Python programs by abstracting away platform-specifics into platform-neutral APIs.

2.2.4.1 OpenCV-Python

OpenCV (Open Source Computer Vision Library) is an open source computer vision and machine learning software library. OpenCV was built to provide a common infrastructure for computer vision applications and to accelerate the use of machine perception in the commercial products. Being a BSD-licensed product, OpenCV makes it easy for businesses to utilize and modify the code.

It has C++, Python, Java and MATLAB interfaces and supports Windows, Linux, Android and Mac OS. OpenCV leans mostly towards real-time vision applications and takes advantage of MMX and SSE instructions when available. A full-featured CUDA and OpenCL interfaces are being actively developed right now. There are over 500 algorithms and about 10 times as many functions that compose or support those algorithms. OpenCV is written natively in C++ and has a templated interface that works seamlessly with STL containers.

Packages for standard desktop environments :

Run **Pip install opencv-python** If you need only main modules

Import the package:

import cv2

2.4.2 OpenCV-Contrib-Python

For CV. Face.LBPH Face Recogniser_Create to work we have to install OpenCV-Contrib-Python

To Install this package recommended command:

“**pip install opencv-contrib-python**”

2.2.4.3 OS

The OS module in python provides functions for interacting with the operating system. OS, comes under Python’s standard utility modules. This module provides a portable way of using operating system dependent functionality. The **os** and **os.path** modules include many functions to interact with the file system.

It is possible to automatically perform many operating system tasks. The OS module in Python provides functions for creating and removing a directory (folder), fetching its contents, changing and identifying the current directory, etc.

This module provides a portable way of using operating system dependent functionality. If you just want to read or write a file see open(), if you want to manipulate paths, see the os .path module, and if you want to read all the lines in all the files on the command line see the fileinput module. For creating temporary files and directories see the tempfile module, and for high-level file and directory handling see the shutil module.

Notes on the availability of these functions:

- The design of all built-in operating system dependent modules of Python is such that as long as the same functionality is available, it uses the same interface; for example, the function **os**.stat(path) returns stat information about *path* in the same format (which happens to have originated with the POSIX interface).
- Extensions peculiar to a particular operating system are also available through the **os** module, but using them is of course a threat to portability.
- All functions accepting path or file names accept both bytes and string objects, and result in an object of the same type, if a path or file name is returned.
- On VxWorks, OS.fork, os.execv and os.spawn p

To import os we use **Pip install os-sys**

2.4.4 PILLOW

Pillow is a **Python** Imaging Library (PIL), which adds support for opening, manipulating, and saving images. The current version identifies and reads a large number of formats. Write support is intentionally restricted to the most commonly **used** interchange and presentation formats.

Library used for pillow:

Pip install pillow

2.4.5 Open pyxl

openpyxl is a Python library to read/write Excel 2010 xlsx/xlsm/xltx/xltm files. It was born from lack of existing library to read/write natively from Python the Office Open XML format All kudos to the PHPExcel team as openpyxl was initially based on PHPExcel.

Installation

Install openpyxl using pip. It is advisable to do this in a Python virtualenv

Without system packages:

“pip install openpyxl”

2.4.6 Passlib

Passlib is a password hashing library for Python 2 & 3, which provides cross-platform implementations of over 30 password hashing algorithms, as well as a framework for managing existing password hashes. It’s designed to be useful for a wide range of tasks, from verifying a hash found in /etc/shadow, to providing full-strength password hashing for multi-user application.

Library used for passlib:

“Pip install passlib”

2.4.7 NumPy

NumPy is a python library used for working with arrays.

It also has functions for working in domain of linear algebra, fourier transform, and matrices.NumPy was created in 2005 by Travis Oliphant. It is an open source project and you can use it freely.

NumPy stands for Numerical Python.

Why Use NumPy?

In Python we have lists that serve the purpose of arrays, but they are slow to process.

NumPy aims to provide an array object that is up to 50x faster that traditional Python lists.

The array object in NumPy is called **ndarray**, it provides a lot of supporting functions that make working with **ndarray** very easy.

Arrays are very frequently used in data science, where speed and resources are very important.

Install it using this command:

“pip install numpy”

3. SYSTEM ANALYSIS AND REQUIREMENTS

3.1 Feasibility Study

The feasibility of the project is analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis, the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

Three key considerations involved in the feasibility analysis are

- ECONOMICAL FEASIBILITY
- TECHNICAL FEASIBILITY
- SOCIAL FEASIBILITY

3.1.1 Economical Feasibility

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

3.1.2 Technical Feasibility

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

3.1.3 Social Feasibility

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system.

3.2 Software And Hardware Requirements

3.2.1 Hardware Requirements

•	System	: Pentium IV 2.4 GHz or Above
•	Hard Disk	: 80 GB.
•	Floppy Drive	: 1.44 Mb.
•	Monitor	: 15 VGA Colour.
•	Mouse	: Logitech.
•	Ram	: 2 GB

3.2.2 Software Requirements

●	OS	: Windows 7 or above/Linux
●	Front End	: python 3 or above
●	Tool	: PyCharm or Python IDLE, any other

3.3 Performance Requirements

Performance is measured in terms of the output provided by the application. Requirement specification plays an important part in the analysis of a system. Only when the requirement specifications are properly given, it is possible to design a system, which will fit into required environment. It rests largely with the users of the existing system to give the requirement specifications

because they are the people who finally use the system. This is because the requirements have to be known during the initial stages so that the system can be designed according to those requirements. It is very difficult to change the system once it has been designed and on the other hand designing a system, which does not cater to the requirements of the user, is of no use.

The requirement specification for any system can be broadly stated as given below:

The system should be able to interface with the existing system

- The system should be accurate
- The system should be better than the existing system
- The existing system is completely dependent on the user to perform all the duties.

4. SOFTWARE DESIGN

4.1 Introduction

Software design is the process or art of defining the architecture, components, modules, interfaces, and data for a system to satisfy specified requirements. One could see it as the application of systems theory to product development. There is some overlap and synergy with the disciplines of systems analysis, systems architecture and systems engineering.

4.2 Uml Diagrams

Unified Modeling Language

A UML system is represented using five different views that describe the system from distinctly different perspective. Each view is defined by a set of diagrams, which is as follows.

- User Model View

i. This view represents the system from the user's perspective.
ii. The analysis representation describes a usage scenario from the end-user's perspective.

- Structural model view

i. In this model, the data and functionality are arrived from inside the system.
ii. This model view models the static structures.

- Behavioral Model View

It represents the dynamic of behavioral as parts of the system, depicting the interactions of collection between various structural elements described in the user model and structural model view.

- Implementation Model View

In this the structural and behavioral as parts of the system are represented as they are to be built.

- Environmental Model View

  In this the structural and behavioral aspect of the environment in which the system is to be implemented are represented.

UML is specifically constructed through two different domains they are:

- UML Analysis modeling, this focuses on the user model and structural model views of the system.
- UML design modeling, which focuses on the behavioral modeling, implementation modeling and environmental model views.

  Use case Diagrams represent the functionality of the system from a user's point of view. Use cases are used during requirements elicitation and analysis to represent the functionality of the system. Use cases focus on the behavior of the system from external point of view.

  Actors are external entities that interact with the system. Examples of actors include users like administrator, bank customer …etc., or another system like central database.

4.2.1 Class Diagram

The class diagram is the main building block of object oriented modeling. It is used both for general conceptual modeling of the systematic of the application, and for detailed modeling translating the models into programming code. Class diagrams can also be used for data modeling. The classes in a class diagram represent both the main objects, interactions in the application and the classes to be programmed.

A class exists with three sections. In the diagram, classes are represented with boxes which contain three parts:

- The upper part holds the name of the class
- The middle part contains the attributes of the class
- The bottom part gives the methods or operations the class can take or undertake

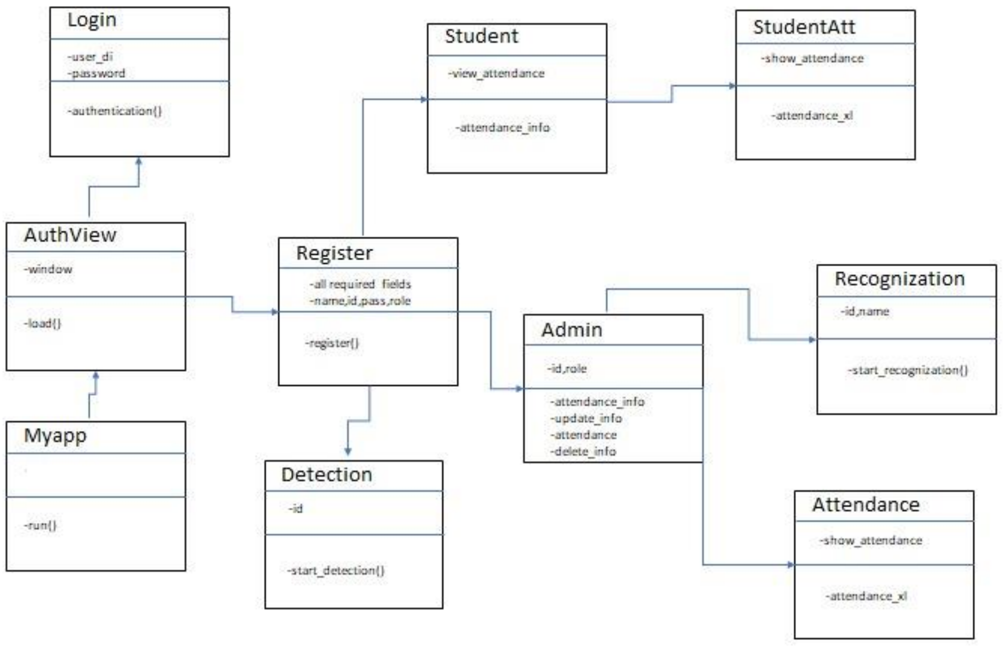
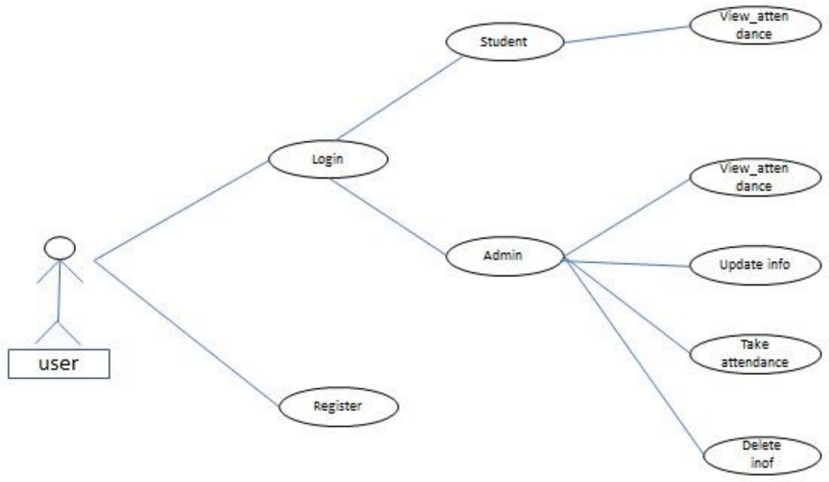


Fig 4.1 Class Diagram

4.2.2 Use Case Diagram

A use case diagram at its simplest is a representation of a user's interaction with the system and depicting the specifications of a use case. A use case diagram can portray the different types of users of a system and the various ways that they interact with the system. This type of diagram is typically used in conjunction with the textual use case and will often be accompanied by other types of diagrams as well.



4.2 Use Case Diagram

4.2.2 Sequence Diagram

A sequence diagram is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. A sequence diagram shows object interactions arranged in time sequence. It depicts the objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario. Sequence diagrams are typically associated with use case realizations in the Logical

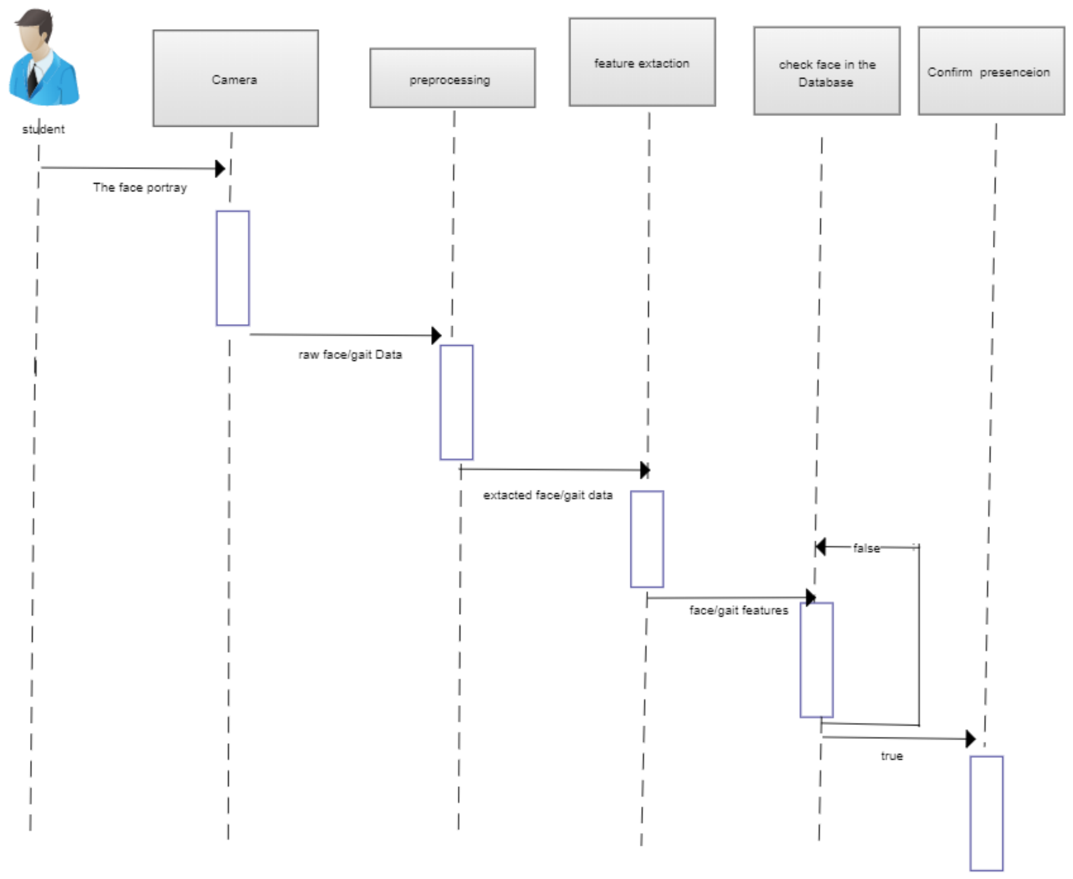


Fig:4.3.

Face Recognition sequence Diagram

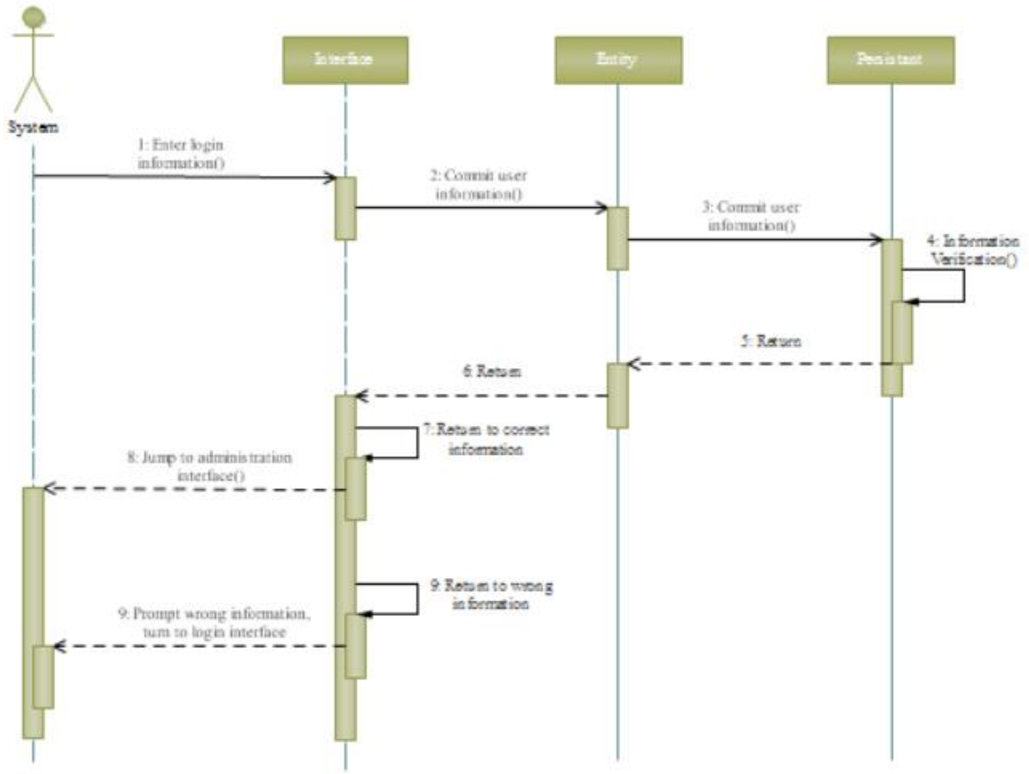

System
Interface
Entity
Persistent
1: Enter login information()
2: Commit user information()
3: Commit user information()
4: Information Verification()
5: Return
6: Return
7: Return to correct information
8: Jump to administration interface()
9: Return to wrong information
9: Prompt wrong information, turn to login interface

Fig:4.4. Login sequence Diagram

4.2.2 Activity Diagram

Activity diagram is another important diagram in UML to describe dynamic aspects of the system. It is basically a flow chart to represent the flow form one activity to another activity. The activity can be described as an operation of the system.

So the control flow is drawn from one operation to another. This flow can be sequential, branched or concurrent.

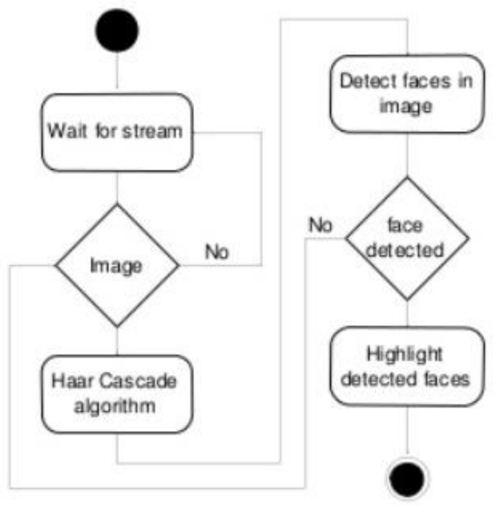
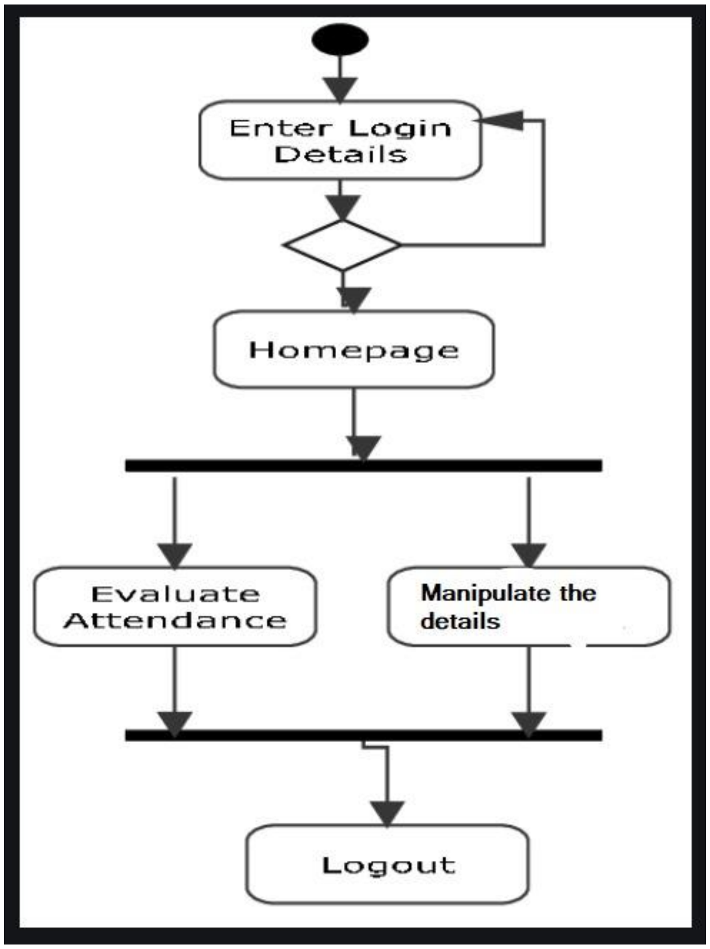


Fig:4.5 Activity Diagram



4.6 Activity Diagram

4.3 MODULE DESCRIPTION

MODULES

4.3.1 Login module

The Login Module is designed in such a way that it takes the user id and password from user and then it will check in the database module.

If the database is present then we can access the application. If the database is not present then it will popup a message box that the user name is invalid.

4.3.2 Register module

The Register module is designed in such a way that it takes the Name, Username, Password, Email, and Phone number of the user and then the data is saved in the database table and a popup message will be shown that the user is successfully registered.

4.3.3 Face Training Controller

In this module a trained data set will be generated, we using the existing data set

4.3.4 Detection module

When user registers some facial data will be collected by Detecting his/her face and the data will be further used for training

4.3.5 Recognition View

The faces are compared predicted and fines the correct user for marking the attendance

4.3.6 Auth Controller

Some precheck s and validating done on the user inputs

4.3.7 Auth Model

Where we will deal with the database (manipulation)

5. CODE

5.1 Auth Controller

```
from root_models.AuthModel import AuthModel
from tkinter import messagebox
from passlib.context import CryptContext

pwd_context = CryptContext(
        schemes=["pbkdf2_sha256"],
        default="pbkdf2_sha256",
        pbkdf2_sha256__default_rounds=30000
)


class AuthController:

    def login(self,id,password):

        if len(id) == 0 or len(password) == 0:
            messagebox.showinfo("alert", "fields cannot be empty")
            message = False
            return message

        am = AuthModel()
        result = am.getUser(id,password)

        if result:
            result1 = pwd_context.verify(password, result[4])
            if result1:
                message = result
```

```
else:
            messagebox.showinfo("Alert",'Wrong password')
            message=False

      else:
         messagebox.showinfo("alert", 'User not found')
         message = False

      return message

   def register(self,id,name,email,phone,password,conform_password,role):
      if password==conform_password:
         password=pwd_context.encrypt(password)
         am = AuthModel()
         result = am.createUser(id,name,email,phone,password,role)
         if result==True:
            return True
         else:
            return result
      else:
         messagebox.showinfo('password conformation','password conformation failed')
def check_role(self,id):
      am=AuthModel()
```

5.2 Auth View

```
from tkinter import *
import tkinter as tk
from tkinter import ttk
from tkinter import messagebox
from root_controllers.AuthController import AuthController
from root_controllers.face_training_controller_OPENCV import FTC



from root_views.detectionView import DetectionView
from root_views.admin_view import Admin
from root_views.student_view import Student
from PIL import ImageTk,Image
import os
class AuthView():

    def load(self):
        self.master=None
        self.master = Tk()
        self.master.title("ATTANDANCE WITH FACIAL RECOGNITION")
        self.master.geometry("1199x600+100+50")
        self.master.resizable(False, False)

        # ==bg image ====
        self.bg = ImageTk.PhotoImage(Image.open("back_ground.jpeg"))
        self.bg_image = Label(self.master, image=self.bg).place(x=0, y=0, relwidth=1, relheight=1)



        # frame directs to login or register page
        self.Frame_Login = Frame(self.master, bg="white")
        self.Frame_Login.place(x=150, y=150, height=340, width=600)
        title = Label(self.Frame_Login, text="ATTANDACE WITH FACE DETECTION",
font=("Impact", 25, "bold"), fg='#d77337',bg='white').place(x=90, y=30)

        #login btn to login page
        login_btn = Button(self.Frame_Login, command=lambda :self.open_login(self),
text="LOGIN", bg="#d77337", fg="white",font=("times new roman", 15)).place(x=150, y=200,
width=150)

        #register btn to register page
        register_btn = Button(self.Frame_Login, command=lambda :self.open_register(self),


text="REGESTER", bg="#d77337", fg="white",font=("times new roman", 15)).place(x=350, y=200,
width=150)

        self.popup_login = None
        self.popup_register = None
        self.master.mainloop()
def open_login(self,av):
        if self.popup_login is None or not self.popup_login.root_login.winfo_exists():
            l=Login_page()
            self.popup_login = l
            l.load(av)
        else:
            self.popup_login.root_login.lift(self.master)
    def open_register(self,av):
        if self.popup_register is None or not self.popup_register.root_register.winfo_exists():
            rp=Register_page()
            self.popup_register = rp
            rp.load(av)
        else:
            self.popup_register.root_register.lift(self.master)
```

5.3 Admin View

```
from tkinter import *
import tkinter as tk
import os
import shutil
from tkinter import messagebox
from PIL import ImageTk,Image
from root_views.recongnitionView_OPENCV import Recognition
from root_controllers.face_training_controller_OPENCV import FTC
from root_controllers.AuthController import AuthController
from root_controllers.xlsx_controller import Create_attendance_sheet
class Admin():

    def load(self,av,id,name,role):
        av.master.destroy()
        self.id=str(id).upper()
        self.name=str(name).upper()
        self.role=str(role).upper()
        self.master = None              #main page code
        self.master = Tk()
        self.master.title("ADMIN")
        self.master.geometry("1200x600+175+50")
        self.master.resizable(False, False)       #fix the window size

        # ==bg image ====
        self.bg = ImageTk.PhotoImage(Image.open("back_ground.jpeg"))
        self.bg_image = Label(self.master, image=self.bg).place(x=0, y=0, relwidth=1, relheight=1)



        # frame for login
        self.Frame_Login = Frame(self.master, bg="white")
        self.Frame_Login.place(x=150, y=100, height=350, width=650)

        self.bg_f = ImageTk.PhotoImage(Image.open('profile_pic.jpeg'))
        self.bg_f_image = Label(self.Frame_Login, image=self.bg_f).place(x=3, y=3)

        self.l1_id = Label(self.Frame_Login, text="ID :", font=('Impact', 12),
fg='#d77337',bg='white').place(x=170, y=15)
        self.l2_name = Label(self.Frame_Login, text="NAME :", font=('Impact', 12),
fg='#d77337',bg='white').place(x=170, y=55)
        self.l3_rle = Label(self.Frame_Login, text="ROLE :", font=('Impact', 12),
fg='#d77337',bg='white').place(x=170, y=95)
        self.txt_id = Label(self.Frame_Login, text=f"{self.id}", font=('Impact',
12),fg='#d77337',bg="white").place(x=225, y=15)
        self.txt_name = Label(self.Frame_Login, text=f"{self.name}", font=('Impact',
12),fg='#d77337',bg="white").place(x=225, y=55)
        self.txt_role = Label(self.Frame_Login, text=f"{self.role}",font=('Impact',
12),fg='#d77337',bg="white").place(x=225, y=95)
        # student_info button
        Student_info = Button(self.Frame_Login, command=lambda :self.open_info(self.id,self.role),
text="View Attendance", bg="#d77337", fg="white",font=("times new roman", 13)).place(x=50,
y=200, width=150)
        #update button
        update = Button(self.Frame_Login, command=self.open_update, text="Update student",
bg="#d77337", fg="white",font=("times new roman", 13)).place(x=225, y=200, width=150)

        #delete button
        delete = Button(self.Frame_Login, command=self.open_delete, text="Delete student",
bg="#d77337", fg="white",font=("times new roman", 13)).place(x=400, y=200, width=150)

        #attandance
        attandance = Button(self.Frame_Login, command=self.open_attendance, text="Attendance",
bg="#d77337", fg="white",font=("times new roman", 13)).place(x=150, y=270, width=150)
        goback_btn=Button(self.Frame_Login,command=lambda :self.log_out(av),text="Log
out",bg="#d77337",fg="white",font=("times new roman",10)).place(x=570,y=20,width=50)
        add_admin=Button(self.Frame_Login,command=self.add_members,text='Add
Members',bg="#d77337", fg="white",font=("times new roman",
13)).place(x=325,y=270,width=150)
```

5.4 Auth Model

```
import sqlite3 as s
from root_models.dbfuns import *
from tkinter import messagebox

class AuthModel:

    def __init__(self):
        self.conn = s.connect('users_db')



self.conn.execute("PRAGMA foreign_keys = 1")
        self.cursor = self.conn.cursor()
        self.create_table(self.cursor)
        self.createAuser()

    def create_table(self,c):
        query1='''CREATE TABLE IF NOT EXISTS "authentic_user" (
                "id"     INTEGER ,
                "role"   TEXT NOT NULL,
                PRIMARY KEY("id")
            )'''
        query2='''CREATE TABLE IF NOT EXISTS "users" (
            "id"        INTEGER,
            "name"      TEXT,
            "email"     TEXT,
            "phone"     INTEGER ,
            "password"          TEXT,
            "role"      TEXT,
            PRIMARY KEY (id),
            FOREIGN KEY (id) REFERENCES authentic_user
            )'''
        query3 = '''create table  if not exists attendance(
            attd_id integer primary key AUTOINCREMENT,
            user_id integer,
            date default current_date ,
            time default current_time ,
            status text,
            foreign key(user_id) references users on delete cascade
            )'''
        try:
            c.execute(query1)
            c.execute(query2)
            c.execute(query3)
            self.conn.commit()
        except:
            print('some db error in create_table')

    def createAuser(self):
        query1 = '''SELECT * FROM authentic_user WHERE id=1'''
        query = '''insert into authentic_user values(1,'admin')'''
        try:
            status = fetchone_fun(self.conn, query1)
            if status:
                pass
            else:
                result = execute_fun(self.conn, query)
                self.conn.commit()
        except:
            print('error')
```

5.5 App

```
from root_views.authView import AuthView, Register_page
class MyApp:
    def run(self):
        av = AuthView()
        av.load()

app = MyApp()
app.run()
```

6. RESULT AND VALIDATION

6.1 Initial Page

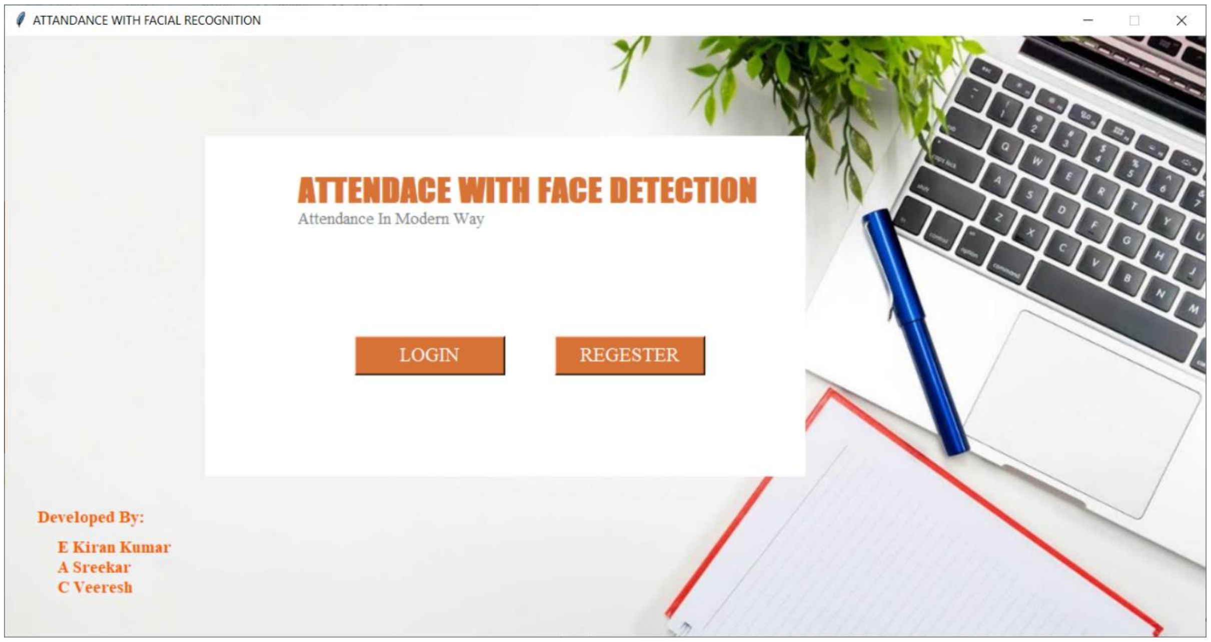
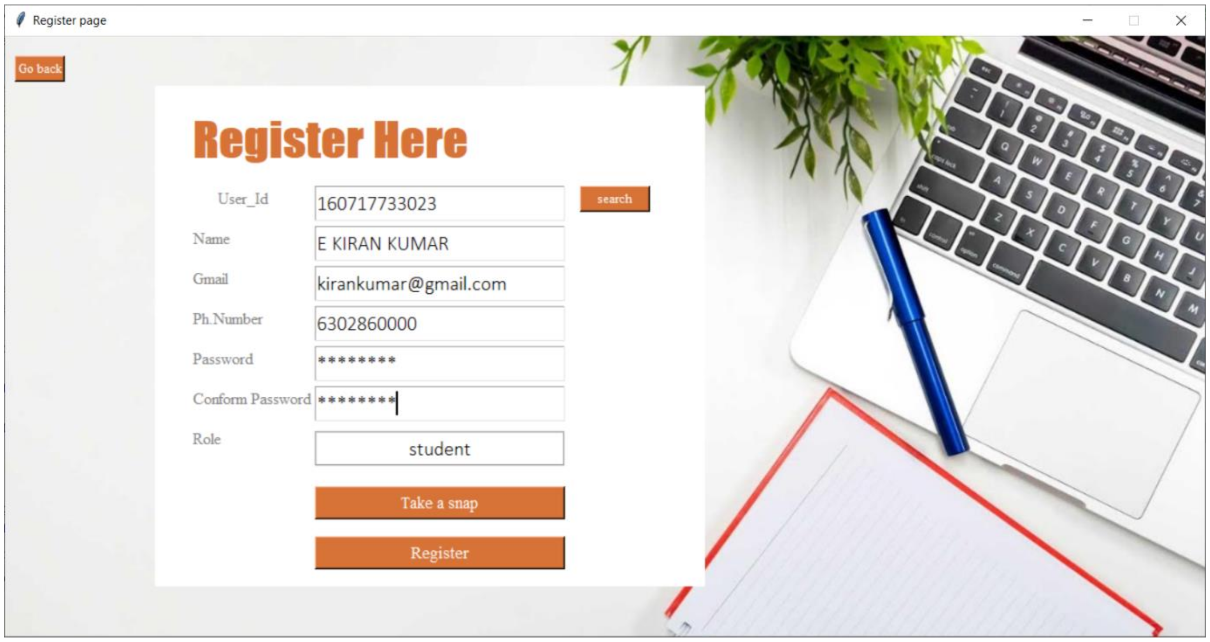


Fig 6.1 Initial Page

6.2 Registration Page



User_Id: 160717733023
Name: E KIRAN KUMAR
Gmail: kirankumar@gmail.com
Ph.Number: 6302860000
Password: ********
Conform Password: ********
Role: student

Fig 6.2 Registration Panel

6.3 Login Page

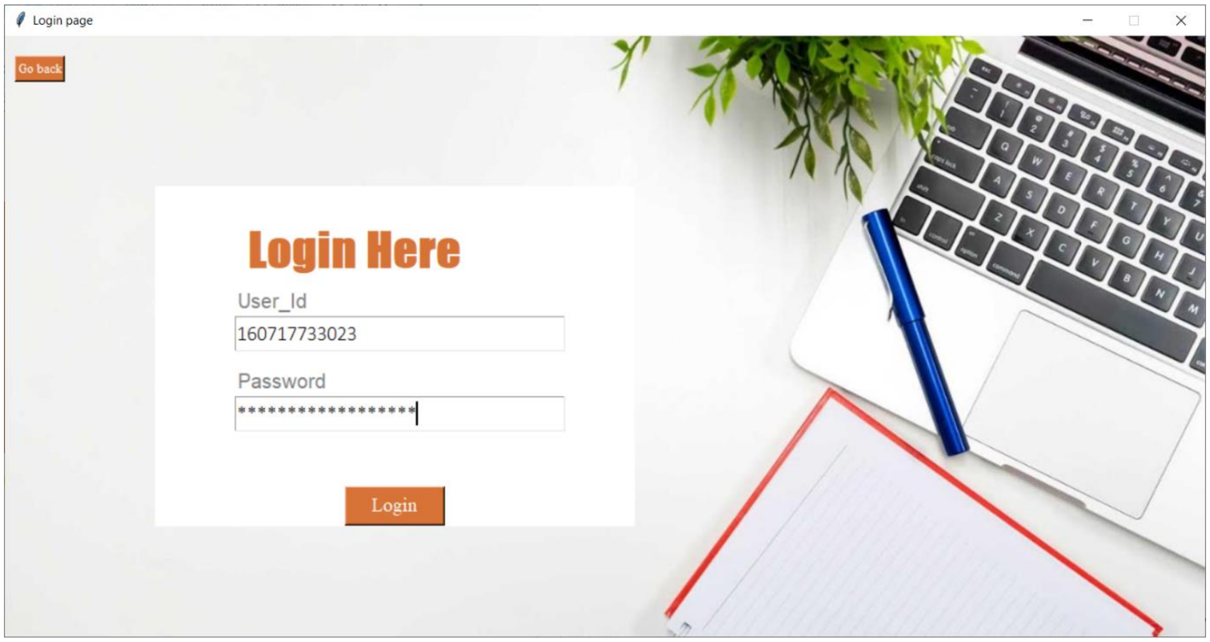


Fig 6.3 Login Page

6.4 Admin Page

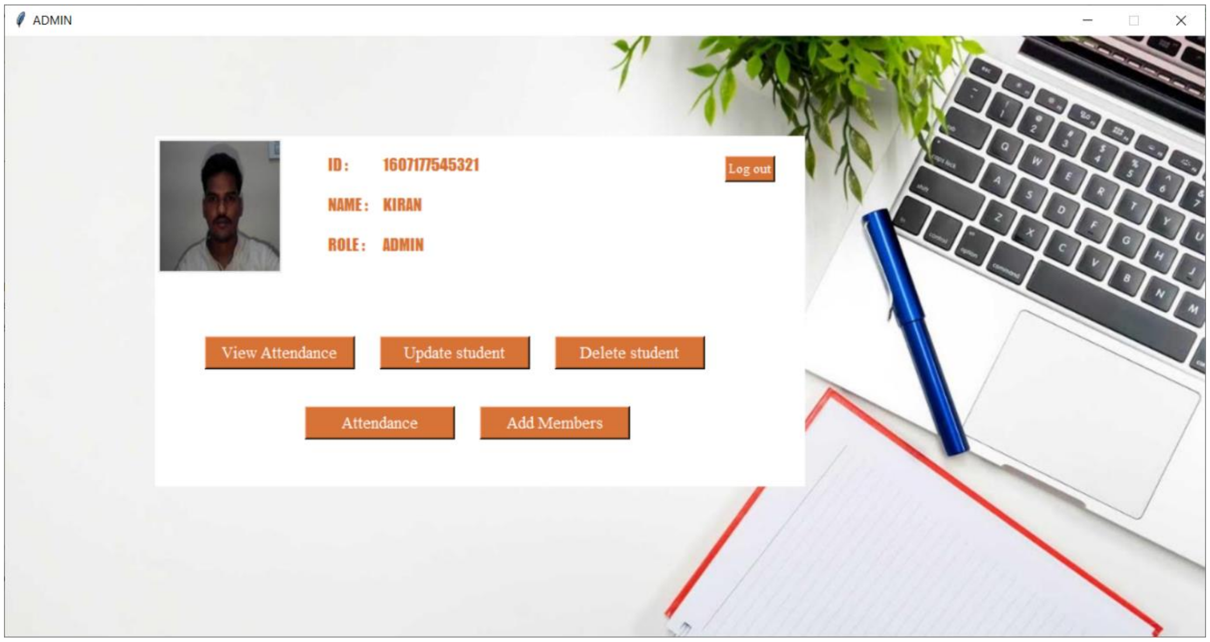


Fig 6.4 Admin Page

6.5 Attendance Information Page

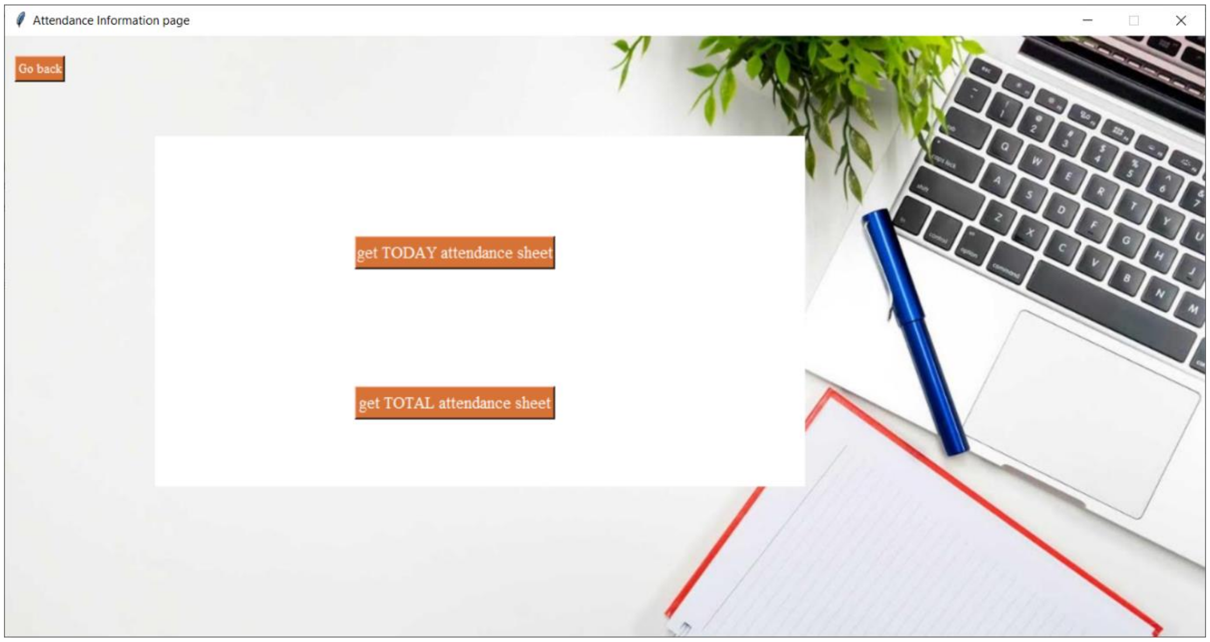
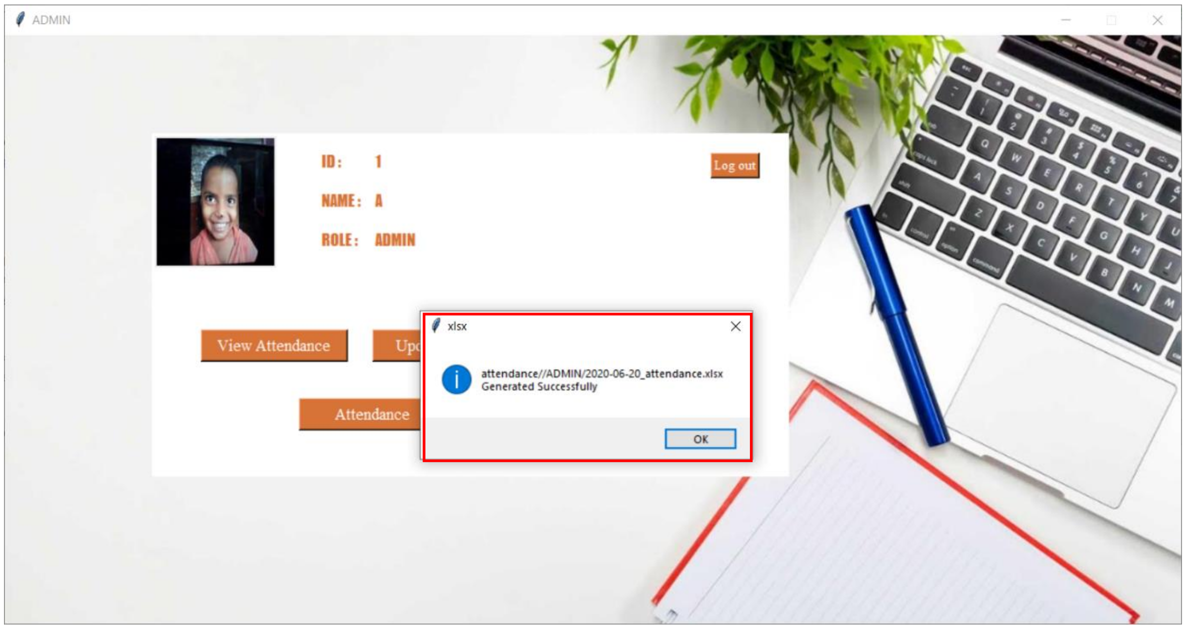

Attendance Information page
Go back
get TODAY attendance sheet
get TOTAL attendance sheet

Fig 6.4 Attendance Information Page

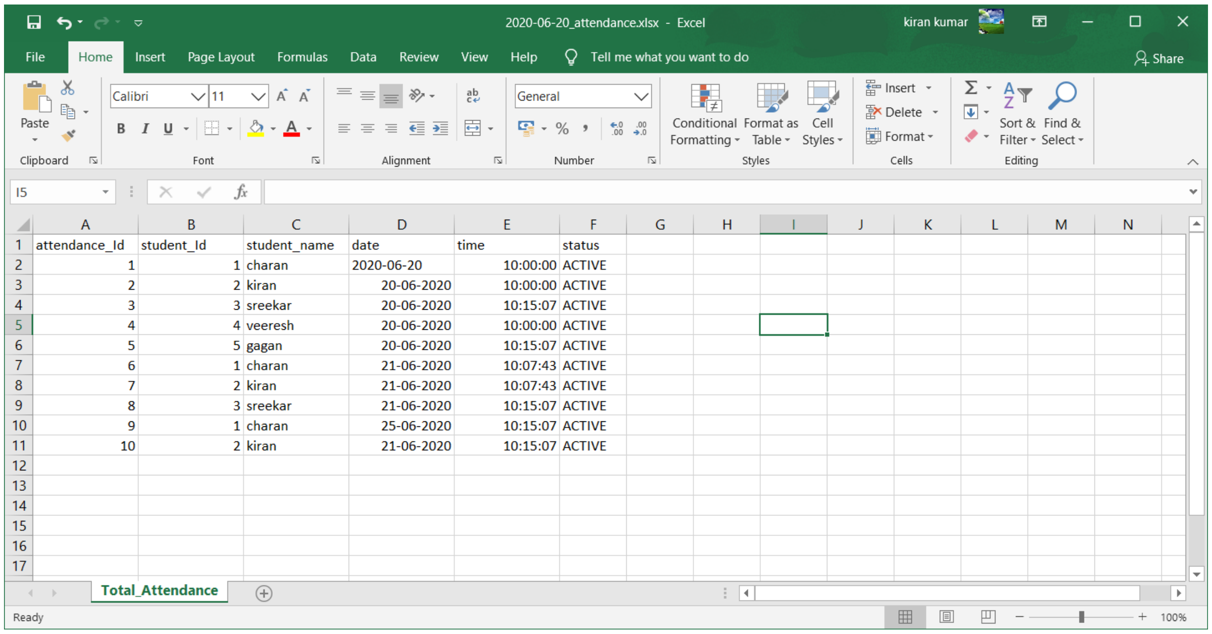
6.6 Attendance Xlsx Sheet Generation



attendance//ADMIN/2020-06-20_attendance.xlsx
Generated Successfully

Fig 6.5 Attendance Xlsx Sheet Generation

6.7 Xlsx Sheet

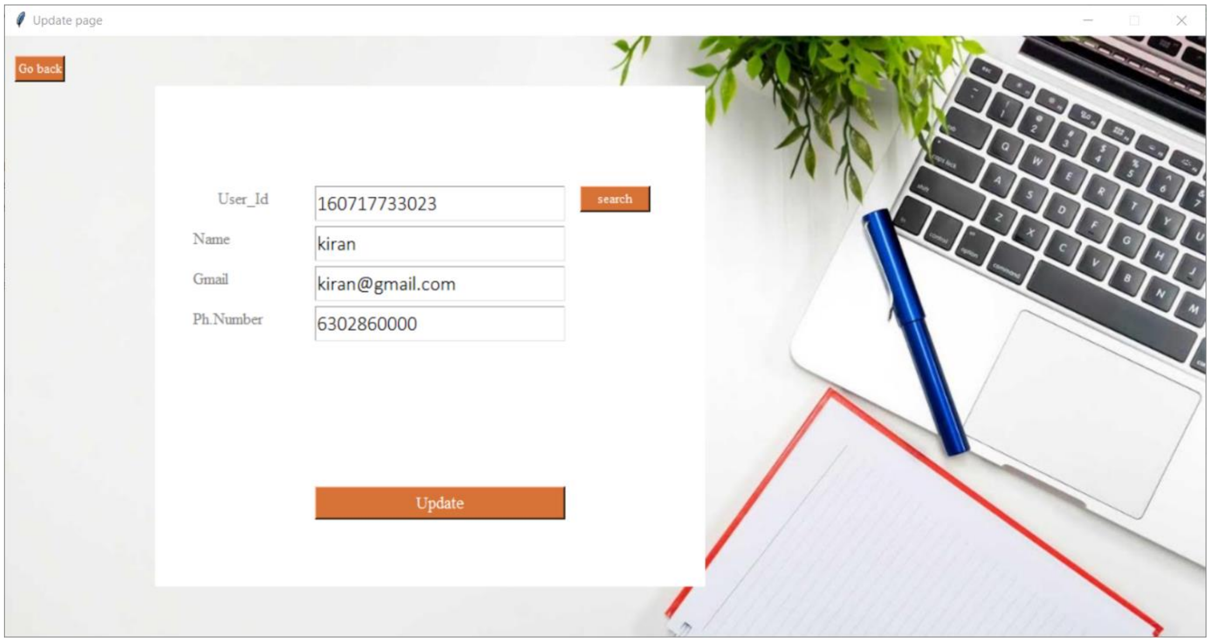


I5

	A	B	C	D	E	F
1	attendance_Id	student_Id	student_name	date	time	status
2	1	1	charan	2020-06-20	10:00:00	ACTIVE
3	2	2	kiran	20-06-2020	10:00:00	ACTIVE
4	3	3	sreekar	20-06-2020	10:15:07	ACTIVE
5	4	4	veeresh	20-06-2020	10:00:00	ACTIVE
6	5	5	gagan	20-06-2020	10:15:07	ACTIVE
7	6	1	charan	21-06-2020	10:07:43	ACTIVE
8	7	2	kiran	21-06-2020	10:07:43	ACTIVE
9	8	3	sreekar	21-06-2020	10:15:07	ACTIVE
10	9	1	charan	25-06-2020	10:15:07	ACTIVE
11	10	2	kiran	21-06-2020	10:15:07	ACTIVE

Fig 6.6 Xlsx Sheet

6.8 Update Page



User_Id 160717733023

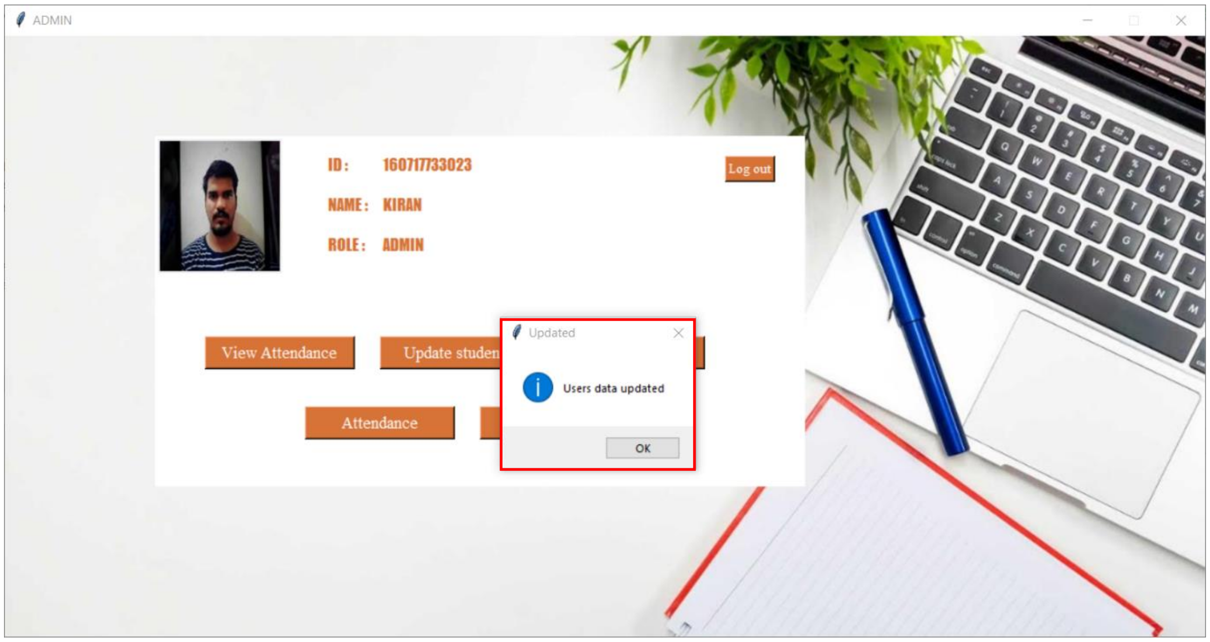
Name kiran

Gmail kiran@gmail.com

Ph.Number 6302860000

Fig 6.7 Update Page

6.9 User Data Updated



ID : 160717733023

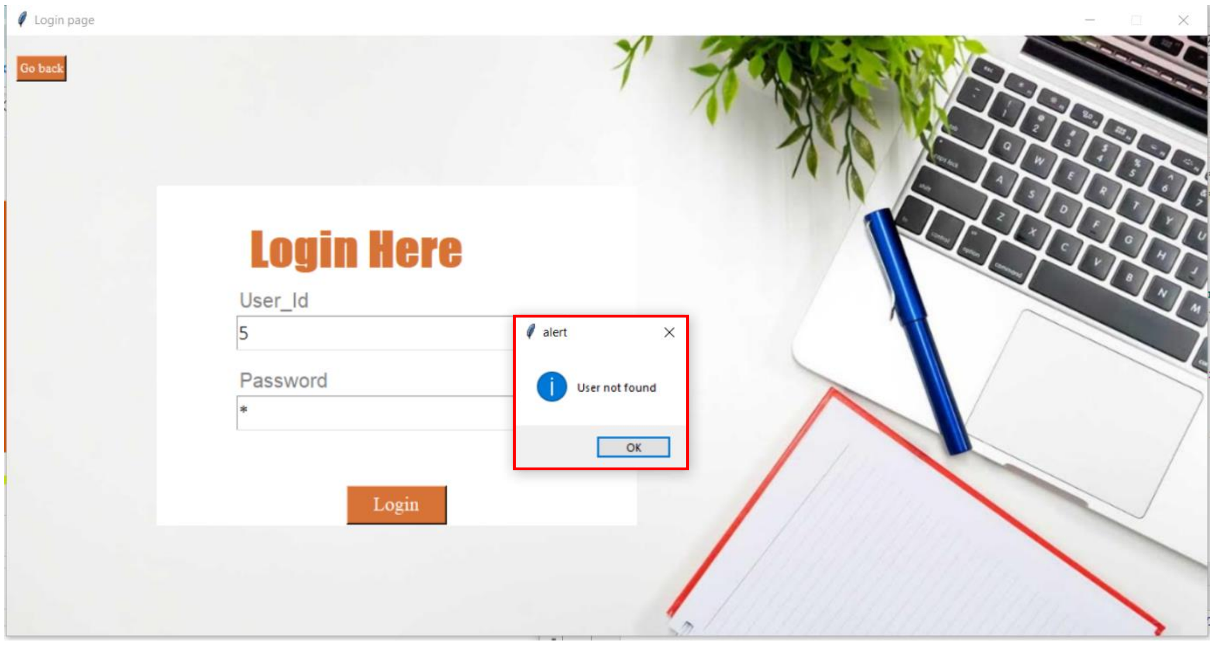
NAME : KIRAN

ROLE : ADMIN

Users data updated

Fig 6.8 User Data Updated

6.10 User Not Found Page



User_Id: 5

Password: *

User not found

Fig 6.9 User Not Found Page

6.11 Delete Page

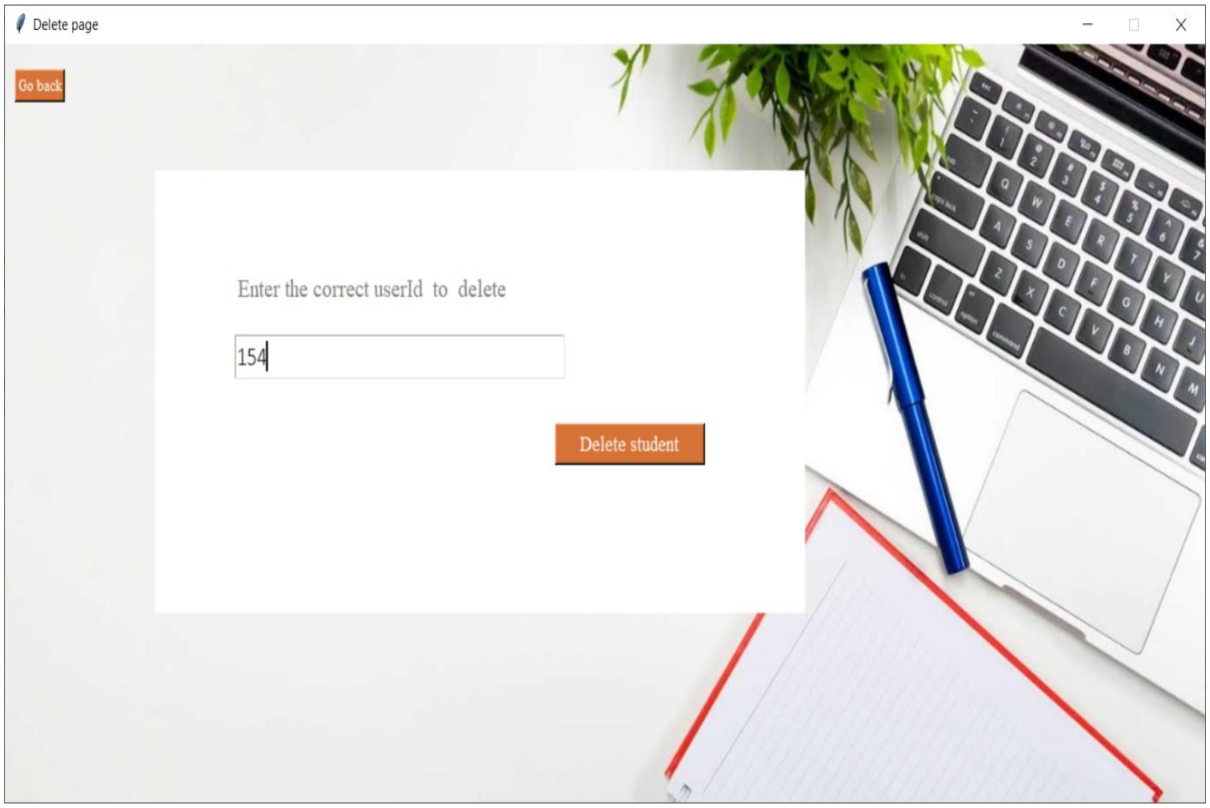


Fig 6.11 Delete Page

6.12 Attendance with Face Recognition

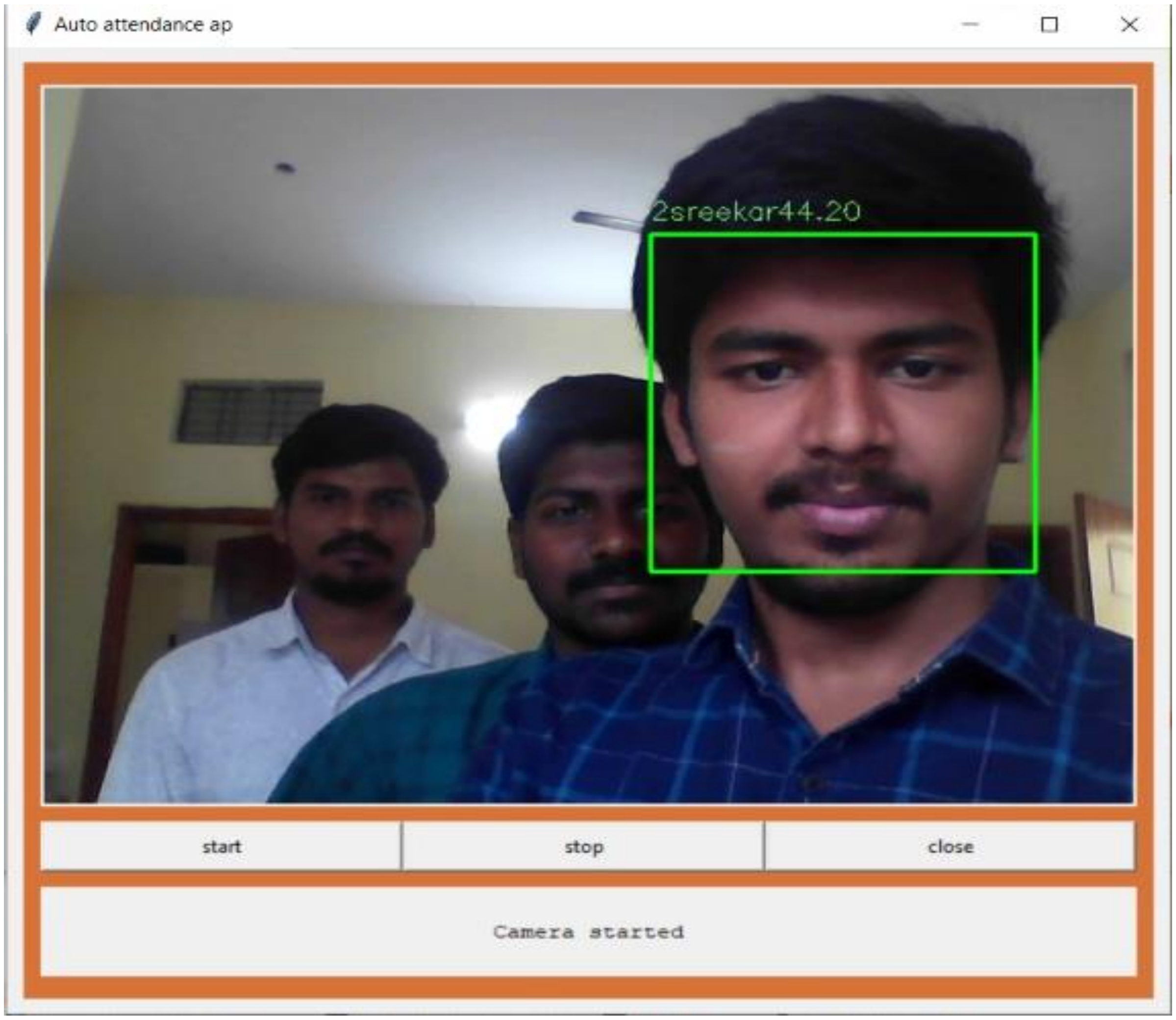
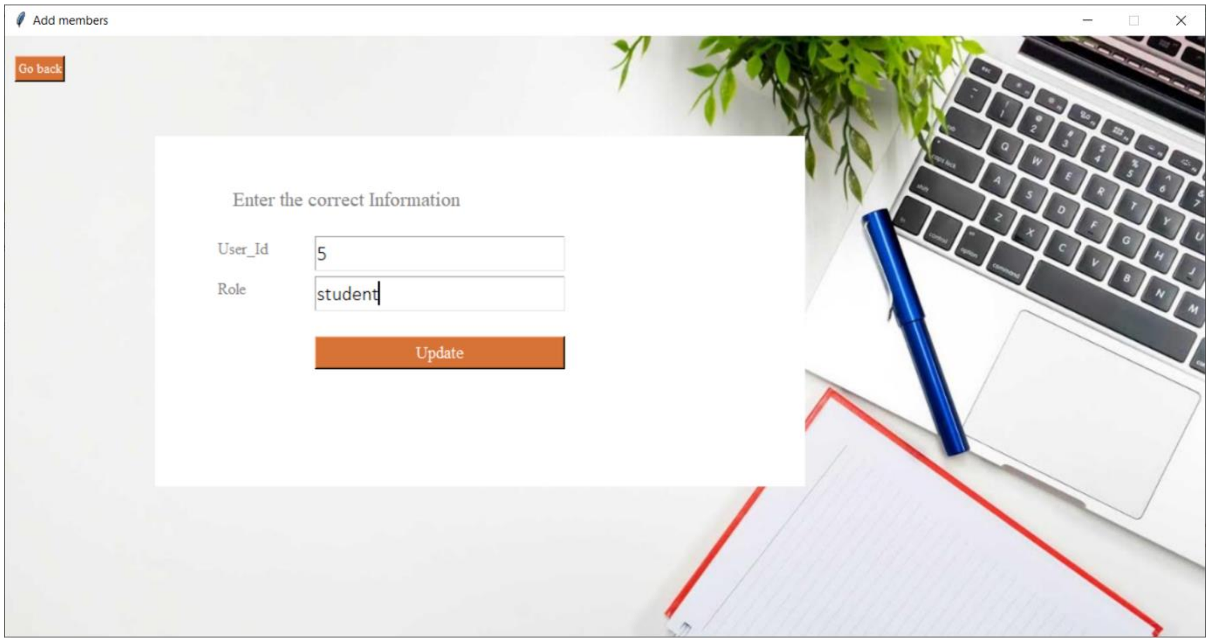


Fig 6.12 Attendance with Face Recognition

6.13 Add Members Page



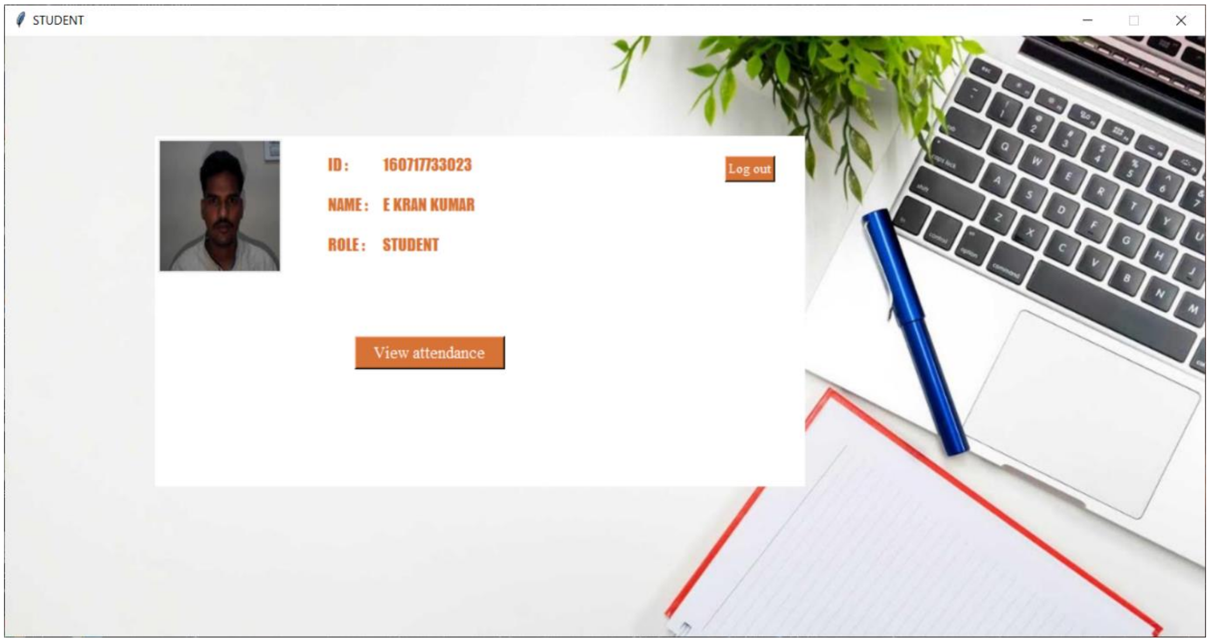
Enter the correct Information

User_Id 5

Role student

Fig 6.13 Add Members Page

6.14 Student Page



ID : 160717733023

NAME : E KRAN KUMAR

ROLE : STUDENT

Fig 6.14 Student Page

6.15 Attendance Table

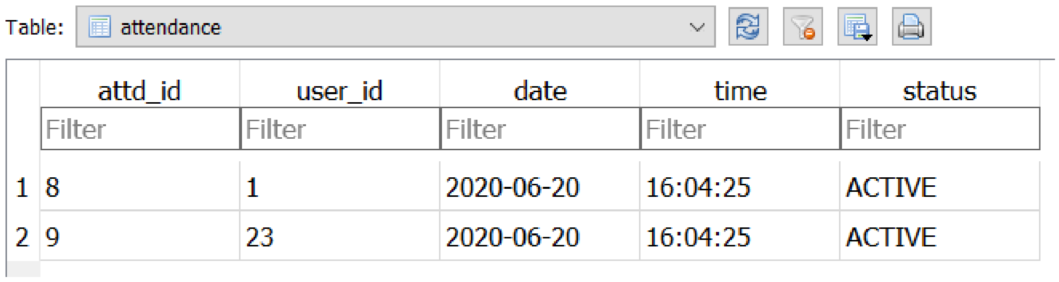


Table: attendance

	attd_id	user_id	date	time	status
	Filter	Filter	Filter	Filter	Filter
1	8	1	2020-06-20	16:04:25	ACTIVE
2	9	23	2020-06-20	16:04:25	ACTIVE

Fig 6.15 Attendance Table

6.16 Authentic Users Table

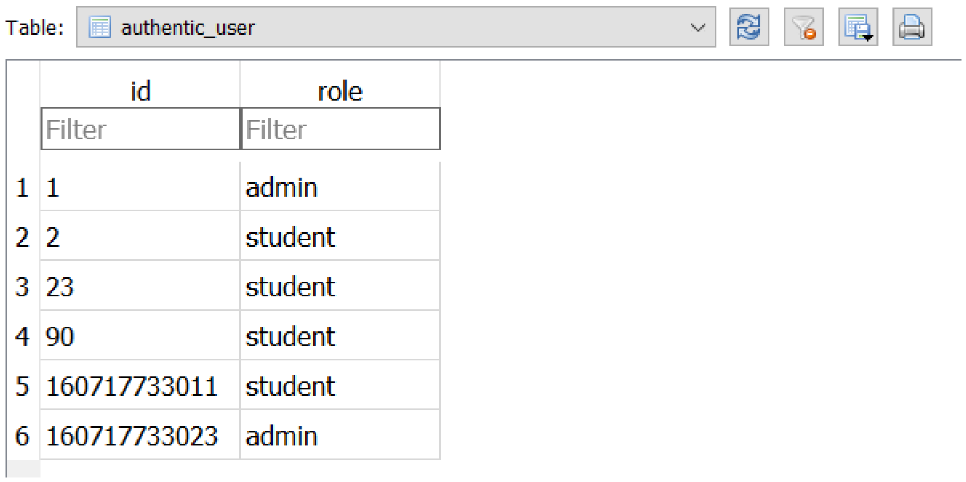


Table: authentic_user

	id	role
	Filter	Filter
1	1	admin
2	2	student
3	23	student
4	90	student
5	160717733011	student
6	160717733023	admin

Fig 6.16 Authentic Users

6.17 Users Table

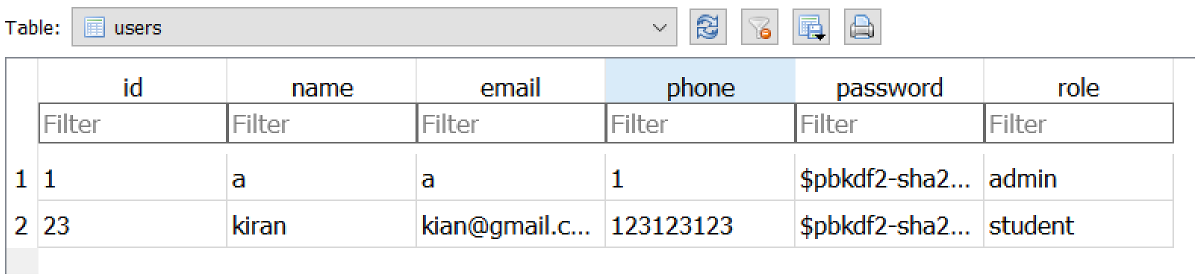


Table: users

	id	name	email	phone	password	role
	Filter	Filter	Filter	Filter	Filter	Filter
1	1	a	a	1	$pbkdf2-sha2...	admin
2	23	kiran	kian@gmail.c...	123123123	$pbkdf2-sha2...	student

Fig 6.17 Users

7. SYSTEM TESTING

7.1 Introduction

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub-assemblies, assemblies and/or a finished product It is the process of exercising software with the intent of ensuring that the

Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

7.2 Types Of Tests

7.2.1 Unit Testing

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

7.2.2 Integration Testing

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

7.2.3 Functional Test

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

Valid Input	: identified classes of valid input must be accepted.
Invalid Input	: identified classes of invalid input must be rejected.
Functions	: identified functions must be exercised.
Output	: identified classes of application outputs must be exercised.
Systems/Procedures	: interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

7.2.4 System Test

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration-oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

7.2.5 White Box Testing

White Box Testing is a testing in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is purpose. It is used to test areas that cannot be reached from a black box level.

7.2.6 Block Box Testing

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box. you cannot “see” into it. The test provides inputs and responds to outputs without considering how the software works.

7.2.7 Acceptance Testing

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

7.3 Test Approach

Testing can be done in two ways

1. Bottom up approach
2. Top down approach

1. Bottom up Approach

Testing can be performed starting from smallest and lowest level modules and proceeding one at a time. For each module in bottom up testing a short program executes the module and provides the needed data so that the module is asked to perform the way it will when embedded within the larger system. When bottom level modules are tested attention turns to those on the next level that use the lower level ones they are tested individually and then linked with the previously examined lower level modules.

2.Top Down Approach

This type of testing starts from upper level modules. Since the detailed activities usually performed in the lower level routines are not provided stubs are written. A stub is a module shell called by upper level module and that when reached properly will return a message to the calling module

indicating that proper interaction occurred. No attempt is made to verify the correctness of the lower level module.

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub-assemblies, assemblies and/or a finished product It is the process of exercising software with the intent of ensuring that the

Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

TEST RESULTS: All the test cases mentioned above passed successfully. No defects encountered.

8. CONCLUSION

In conclusion, more benefits compare with the existing systems like maintaining records or by scanning fingerprints. Attendance with Face Detection application will take attendance automatically and store the data securely. It reduces the human efforts and saves time. There will be no attendance mismatches. Easy to maintain the attendance records.

Further we can implement this application in CCTV footages, so that it can maintain the attendance records.

9. BIBIOLOGRAPHY

- International Journal of Computer and Communication Engineering, Vol. 1.No.2, July 2012-Study of Implementing Automated Attendance System Using Face Recognition Technique by Nirmala Kar, Mrinal Kanti Debbarma, Ashim Saha, and Dwijen Rudra Pal.

- Real time face recognition system using PCA and various distance classifiers by Deepesh Raj – IIT Kanpur.

- Adrian Rhesa Septian Siswanto, Anto Satriyo Nugroho, Maulahikmah Galinium, “Implementation of face recognition algorithm for biometrics based time attendance system”, IEEE, ICT For smart Scociety(ICISS), International Conference, January 2015.

10. REFERENCES

- OpenCV Documentation- http://docs.opencv.org/2.4/modules/contrib/doc/facerec_tutorial.html
- Dlib Documentation- http://dlib.net/imaging.html
- Software Engineering – A Practitioners Approach, 6th Edition, Tata McGraw Hill Tutorial from docs.python.org
- Python Crash Course, Eric Matthes 2016 Edition.
- Pyimage search https://www.pyimagesearch.com/2018/09/24/opencv-face-recognition