

**A PROJECT REPORT
ON
“Face Recognize Based Attendance System”**

Submitted by
Rajkumar Premchand Gupta

In partial fulfilment for the award of the degree
Of
BACHELOR OF SCIENCE

In
University of Mumbai

Under the Guidance of
Prof. Mrs. Anita Gaikwad

Final Year Computer Science



**DEPARTMENT OF COMPUTER SCIENCE
Ramniranjan Jhunjhunwala College of Arts, Science &
Commerce**

Affiliated to University of Mumbai.

Mumbai(2019-20)

CERTIFICATE

This is to certify that **Rajkumar Gupta** has successfully completed the project titled “ **Face Recognize Attendance System** ” under our guidance towards the partial fulfilment of degree of **Bachelors of Science (Computer Science – SEM V)** submitted to **Ramniranjan Jhunjhunwala College, Ghatkopar (W)** during the academic year **2019 – 2020** as specified by **UNIVERSITY OF MUMBAI**.

Prof. (Mrs.) Anita Gaikwad
Project Guide
In-Charge
Dept. Of ComputerScience

Acknowledgement

I have great pleasure for representing this project report entitled “**Face Recognize Attendance System**” and I grab this opportunity to convey my immense regards towards all the distinguished people who have their valuable contribution in the hour of need.

I take this opportunity to thank “**Dr. Mrs. Usha Mukundan**”, our **Principal of Ramniranjan Jhunjhunwala College, Ghatkopar (W)** for giving me an opportunity to study in the institute and the most needed guidance throughout the duration of the course.

I also like to extend my gratitude to “**Prof.Mrs.Anita Gaikwad**”, **Head of the Department** for their timely and prestigious guidance.

I also owe to my fellow friends who have been constant source of help to solve the problems and also helped me during the project development phase.

Thanking You,
Rajkumar Gupta

Abstract

Automatic face recognition (AFR) technologies have seen dramatic improvements in performance over the past years, and such systems are now widely used for security and commercial applications.

An automated system for human face recognition in a real time background for a college to mark the attendance of their employees. So Smart Attendance using Real Time Face Recognition is a real world solution which comes with day to day activities of handling employees. The task is very difficult as the real time background subtraction in an image is still a challenge. To detect real time human face are used and a simple fast Principal as a component Analysis has used to recognize the faces detected with a high accuracy rate. The matched face is used to mark attendance of the employee.

Our system maintains the attendance records of employees automatically. Manual entering of attendance in logbooks becomes a difficult task and it also wastes the time. So we designed an efficient module that comprises of face recognition to manage the attendance records of employees. Our module enrolls the student's face . This enrolling is a onetime process and their face will be stored in the database. During enrolling of face we require a system since it is a onetime process. You can have your own roll number as your id which will be unique for each student. The presence of each student will be updated in a database. The results showed improved performance over manual attendance management system. Attendance is marked after student identification.

This product gives much more solutions with accurate results in user interactive manner rather than existing attendance manage systems.

Index

Acknowledgement.....	3
Abstract.....	4
Chapter1: Introduction.....	7-8
1.1 Introduction to Face RecognizeAttendanceSystem.....	8
Chapter 2:Preliminary Investigation.....	9-12
2.1 Introduction.....	10
2.2 Scope of the System.....	10
2.3 Existing System and its Disadvantages.....	10
2.4 Proposed System and its Advantages.....	11
2.5 Feasibility Study.....	12
Chapter 3: Requirement.....	13-14
3.1 Introduction.....	14
3.2 Technical Requirement.....	14
3.2.1 Programming Language.....	14
3.2.2 Software.....	14
3.2.3 Hardware.....	14
Chapter 4: Gantt Chart.....	15-16
4.1 Introduction.....	16
4.2 Task Performed.....	16
Chapter 5: Software Development Model.....	17-20
5.1 Introduction.....	18
5.2 V-Model.....	18
Chapter 6: System Analysis.....	21-40
6.1 Introduction.....	22
6.2 E-R diagram.....	22
6.3 Class diagram.....	24
6.4 Object diagram.....	26
6.5 Use Case diagram.....	28
6.6 Sequence diagram.....	30
6.7 Activity diagram.....	34
6.8 Component diagram.....	36
6.9 Deployment diagram.....	38
6.10 Table design.....	39
Chapter 7: System Coding And Output.....	41-53
7.1 Program Code.....	41
7.2 Output.....	49
Chapter 8: System Testing.....	54-55
8.1 Validation.....	55
Chapter 9: Conclusion.....	56-57
9.1 Conclusion.....	56
9.2 FutureScope.....	57
References.....	58-59

Declaration

I declare that this written submission represents our own ideas in our own words and where other's ideas or words have been included, I have adequately cited and referenced the original sources. I also declare that I have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/fact/data/source in our submission. I understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

Rajkumar Gupta

Introduction

Chapter 1

Introduction

1.1 Introduction To Face Recognize Attendance System

Maintaining the attendance is very important in all the institutes for checking the performance of students. Every institute has its own method in this regard. Some are taking attendance manually using the old paper or file based approach and some have adopted methods of automatic attendance using some biometric techniques. But in these methods student have to wait for long time in making a queue at time they enter the office. Many biometric systems are available but the key authentications are same is all the techniques. Every biometric system consists of enrolment process in which unique features of a person is stored in the database and then there are processes of identification and verification. These two processes compare the biometric feature of a person with previously stored template captured at the time of enrollment. Biometric templates can be of many types like Fingerprints, Eye Iris, Face, Hand Geometry, Signature, Gait and voice. Our system uses the face recognition approach for the automatic attendance of students in the class room room environment without student intervention. Face recognition consists s of two steps, in first step faces are detected in the image and then these detected faces are compared with the database for verification.

A number of methods have been proposed for face detection i.e. Ada Boost algorithm, the Float Boost algorithm, the S-Ada Boost algorithm Support Vector Machines (SVM), and the Bayes classifier. The efficiency of face recognition algorithm can be increased with the fast face detection algorithm. In all the above methods SURF is most efficient.Face recognition techniques can be Divided into two types Appearance based which use texture features that is applied to whole face or some specific Regions, other is Feature based which uses geometric features like mouth, nose, eyes, eye brows, cheeks and Relation between them. Statistical tools such as Linear Discriminant Analysis (LDA), Principal Component Analysis (PCA), Kernel Methods, and Neural Networks, Eigen-faces have been used for construction of face templates. Illumination invariant algorithm is utilized for removing the lighting effect inside the office room

Preliminary Investigation

Chapter 2

Preliminary Investigation

2.1 Introduction

A preliminary investigation refers to limited scope inquiry undertaken to verify wheather an allegation merits a full investigation

2.2 Scope of the System

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 to computerize this process by providing a system that record and manage students' attendance automatically by capturing face while entering in the class without needing to lecturers' interference.

Project Objective

- a) Reduce manual process by providing automated and reliable attendance system.
- b) Increase privacy and security which student cannot present himself or his friend when they are not.
- c)Reduce time loss as time is a valuable resource

2.3 Existing System and it's Disadvantages

The current method that institute uses faculty passes an attendance sheet, make roll calls or scan the id by scanner mark the attendance of the students.which sometimes disturb the discipline.Not only discipline but also there are following Problems They are:

1. There is always a chance of forgery(one person signing the presence of the other one)
Since these are manually so there is great risk of error.
2. In case of id scanner, student get distract from the professor's lecture.
3. Difficult to maintain a database or register in manual system
4. More manpower is required.

2.4 Proposed System and It's Advantages

To overcome a problem in existing system, I shall develop a face based attendance system over a simple attendance system.

ADVANTAGES

1. It will mark the attendance of student via face id
2. It will detect the face by simple camera which is connected to system application.
3. It is more suitable application in institute.
4. Low maintenance levels are required by the system.
5. The admin will able to print record of attendance afterward.
6. Low cost system, providing maximum automation.
7. Very Easy to update the students details .
8. Environment friendly(save paper).

2.5 Feasibility Study

2.5.1 Introduction

Feasibility study is to check the viability of the project under consideration. Theoretically various types of feasibilities are conducted, but I have conducted three types of feasibilities explained as under.

2.5.2 Economical feasibility

Equipment required for developing the software are easily available. Saving of paper work and manpower reduced. Once the required hardware and software requirements get fulfilled there is no need for the users of our system to spend for any additional overhead.

2.5.3 Technical feasibility

At first it is necessary to check that the proposed system is technically feasible or not and to determine the technology and skill necessary to carry out the project. If they are not available then find out the solution to obtain them.

2.5.4 Operational feasibility

Proposed system is beneficial only if it can be turned into system that will meet the need of the client's operating requirements. The proposed system is operationally feasible due to the following reasons:

- a) It is easy to use and is very simple.
- b) The application will avoid confusion and resistance by catching the user's attention, as it is presentable.

Requirement

Chapter 3

Requirements

3.1 Introduction

To develop any project there is a requirement of software, hardware and programming language. In our project camera, opencv module play a vital role.

3.2 Technical Requirement

3.2.1 Programming Language

1. Python
2. Mysql

3.2.2 Software Requirements

1. Windows 10 OS.
2. Python 3.6
3. Mysql 8.0

3.2.3 Hardware Requirements

1. Monitor
2. Processor: Intel core i3-6006U or Latest Version
3. Processor Speed: 2.0GHZ or More
4. Ram: 4GB or More
5. HardDisk: 1TB or More
6. Keyboard: Any keyboard having proper function
7. Mouse: Any Mouse having proper function
8. Camera

Chapter 4

Gantt chart

4.1 Introduction

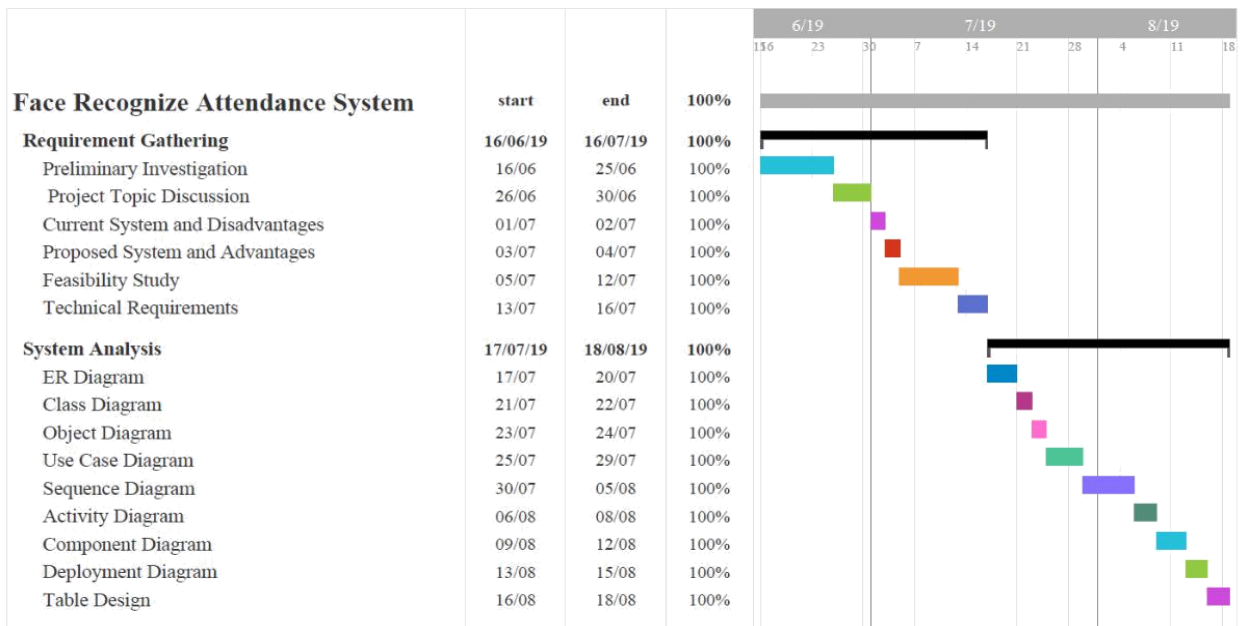
A Gantt chart is a type of bar chart that illustrates a project schedule. Gantt chart illustrates the start and finish dates of the terminal elements and summary elements of a project. Terminal elements and summary elements comprise the work breakdown structure of the project. Gantt Charts are useful tools for analyzing and planning more complex projects.

They:

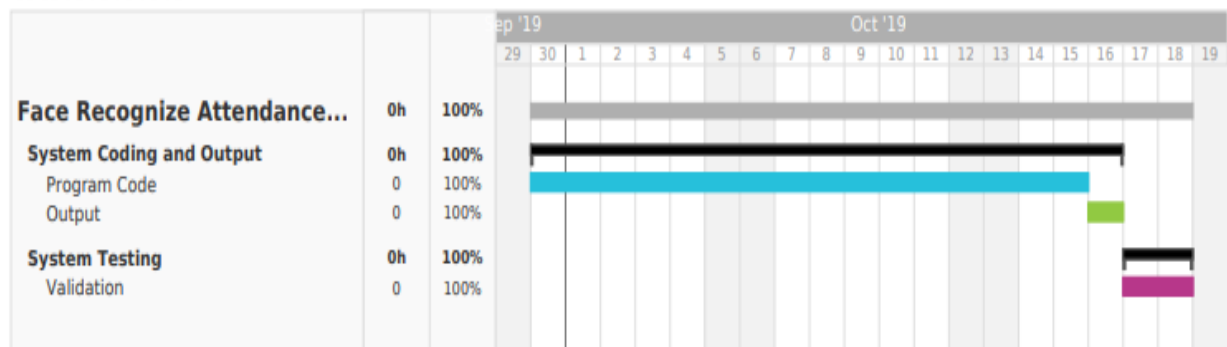
1. Help you to plan out the tasks that need to be completed.
2. Give you a basis for scheduling when these tasks will be carried out.
3. Help you to work out critical path for a project where you must be complete it by a particular date.

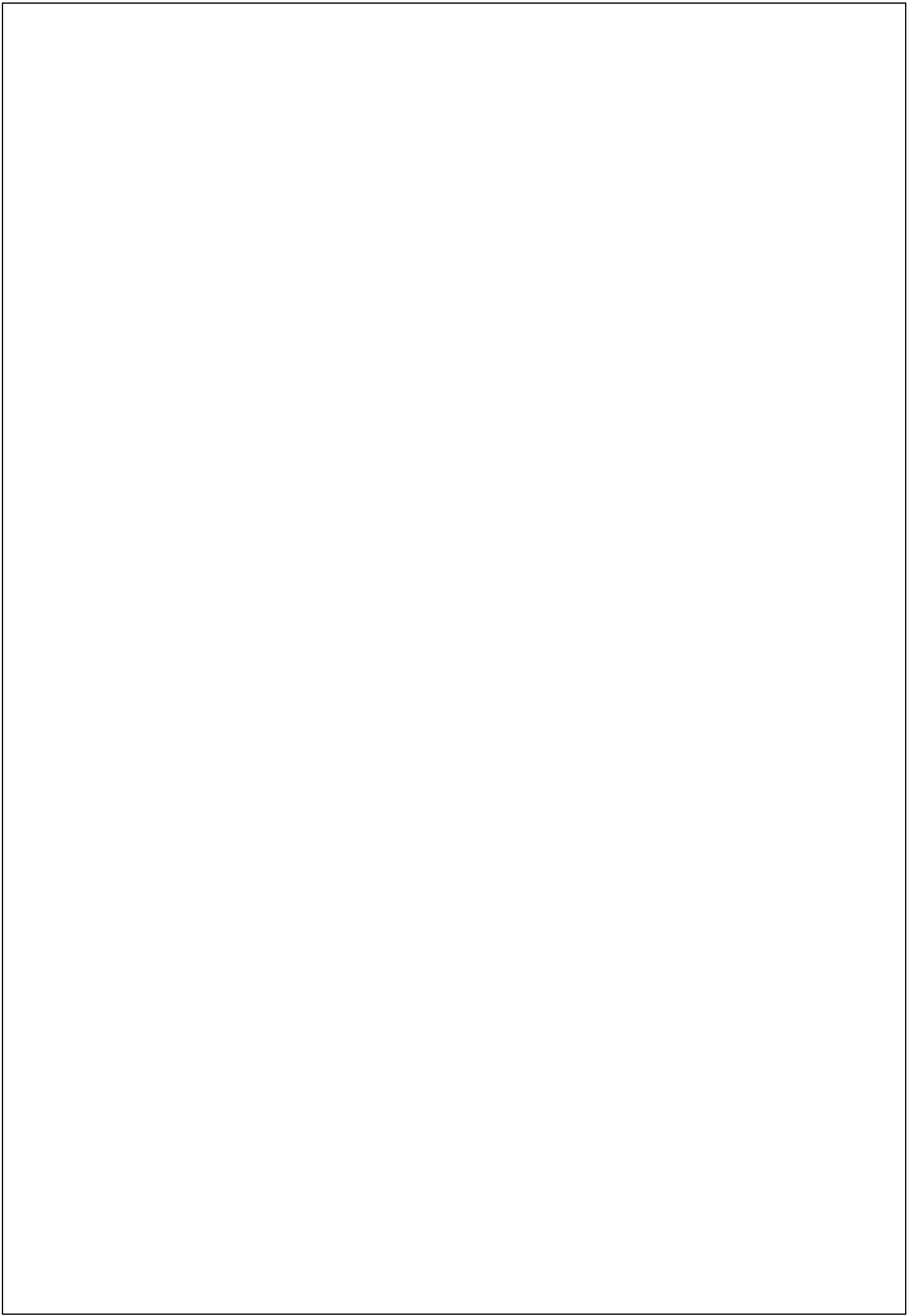
When a project is under way, Gantt chart helps you to monitor whether the project is on schedule. If it is not, it allows you to pinpoint the remedial action necessary to put it back on schedule. This chart is also used in Information Technology to represent data that has been collected.

4.2 Task Performed



teamgantt
Created with Free Edition





Software Development Model

Chapter 5

Software Development Model

5.1 Introduction

The software development models are the various processes or methodologies that are being selected for the development of the project depending on the project's aims and goals. There are many development life cycle models that have been developed in order to achieve different required objectives. The models specify the various stages of the process and the order in which they are carried out. The selection of model has very high impact on the testing that is carried out. It will define the what, where and when of our planned testing, influence regression testing and largely determines which test techniques to use.

There are various Software development models or methodologies; I have used one of it i.e **V-model**

5.2 V-Model

The V-model is an SDLC model where execution of processes happens in a sequential manner in a V-shape. It is also known as Verification and Validation model.

The V-Model is an extension of the waterfall model and is based on the association of a testing phase for each corresponding development stage. This means that for every single phase in the development cycle, there is a directly associated testing phase. This is a highly-disciplined model and the next phase starts only after completion of the previous phase.

Requirement Analysis

This is the first phase in the development cycle where the product requirements are understood from the customer's perspective. This phase involves detailed communication with the customer to understand his expectations and exact requirement. This is a very important activity and needs to be managed well, as most of the customers are not sure about what exactly they need. The acceptance test design planning is done at this stage as business requirements can be used as an input for acceptance testing.

System Design

Once you have the clear and detailed product requirements, it is time to design the complete system. The system design will have the understanding and detailing the complete hardware and communication setup for the product under development. The system test plan is developed based on the system design. Doing this at an earlier stage leaves more time for the actual test execution later.

Architectural Design

Architectural specifications are understood and designed in this phase. Usually more than one technical approach is proposed and based on the technical and financial feasibility the final decision is taken. The system design is broken down further into modules taking up different functionality. This is also referred to as High Level Design (HLD).

The data transfer and communication between the internal modules and with the outside world (other systems) is clearly understood and defined in this stage. With this information, integration tests can be designed and documented during this stage.

Module Design

In this phase, the detailed internal design for all the system modules is specified, referred to as Low Level Design (LLD). It is important that the design is compatible with the other modules in the system architecture and the other external systems. The unit tests are an essential part of any development process and helps eliminate the maximum faults and errors at a very early stage. These unit tests can be designed at this stage based on the internal module designs.

Coding Phase

The actual coding of the system modules designed in the design phase is taken up in the Coding phase. The best suitable programming language is decided based on the system and architectural requirements.

The coding is performed based on the coding guidelines and standards. The code goes through numerous code reviews and is optimized for best performance before the final build is checked into the repository.

Validation Phases

The different Validation Phases in a V-Model are explained in detail below.

Unit Testing

Unit tests designed in the module design phase are executed on the code during this validation phase. Unit testing is the testing at code level and helps eliminate bugs at an early stage, though all defects cannot be uncovered by unit testing.

Integration Testing

Integration testing is associated with the architectural design phase. Integration tests are performed to test the coexistence and communication of the internal modules within the system.

System Testing

System testing is directly associated with the system design phase. System tests check the entire system functionality and the communication of the system under development with external systems. Most of the software and hardware compatibility issues can be uncovered during this system test execution.

Acceptance Testing

Acceptance testing is associated with the business requirement analysis phase and involves testing the product in user environment. Acceptance tests uncover the compatibility issues with the other systems available in the user environment. It also discovers the non-functional issues such as load and performance defects in the actual user environment.

System Analysis

Chapter 6

System Analysis

6.1 Introduction

System analysis in software engineering is, therefore, the activities that comprise software engineering as a process in the production of software. It is a software process.


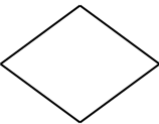


6.2 E-R Diagram

6.2.1 Introduction

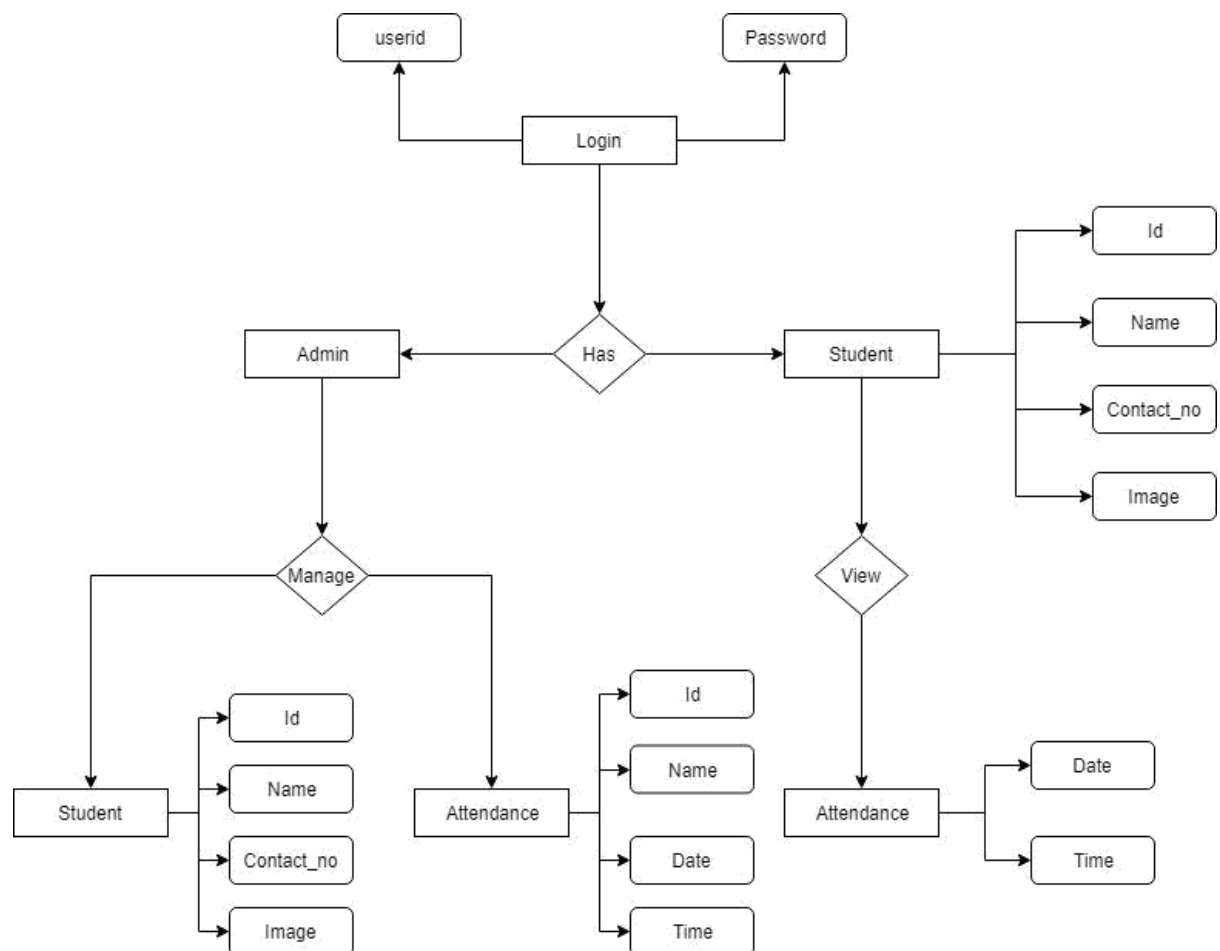
ER diagrams are a visual tool which is helpful to represent the ER model. It was proposed by Peter Chen in 1971. He aimed to use an ER model as a conceptual modelling approach.

The ER or (Entity Relational Model) is a high-level conceptual data model diagram. Entity-Relation model is based on the notion of real-world entities and the relationship between them.

6.2.2 Diagram Notations

Name	Symbol	Description
Rectangles		This symbol represents entity types.
Diamonds		This symbol represents relationship types.
Lines		It links attributes to entity types and entity types with other relationship types.
Oval		This symbol represents an attribute of the entity.

6.2.3 Diagram:



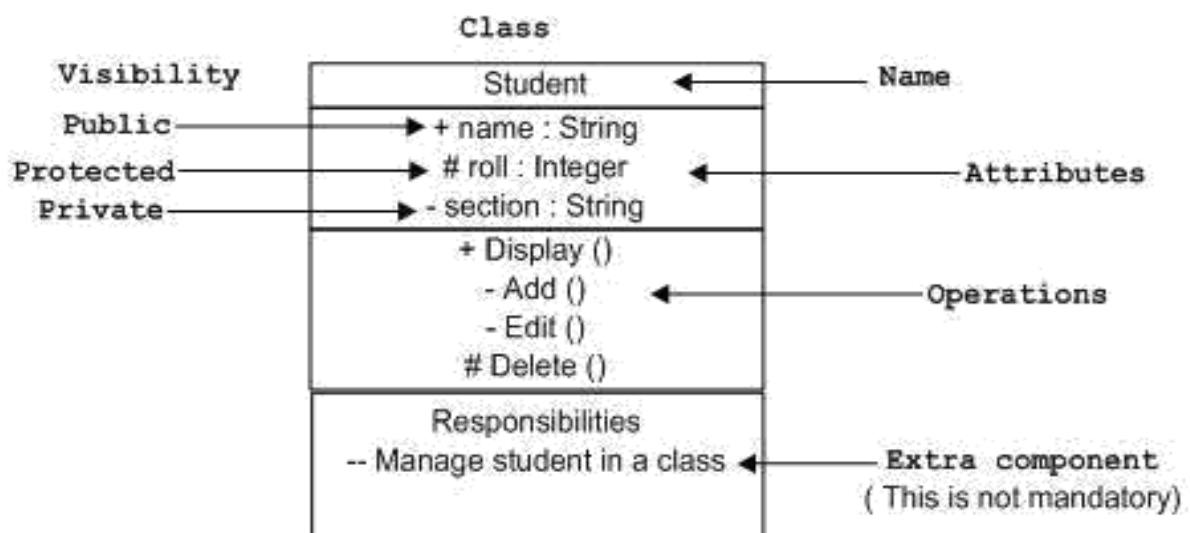
6.3 Class Diagram

6.3.1 Introduction

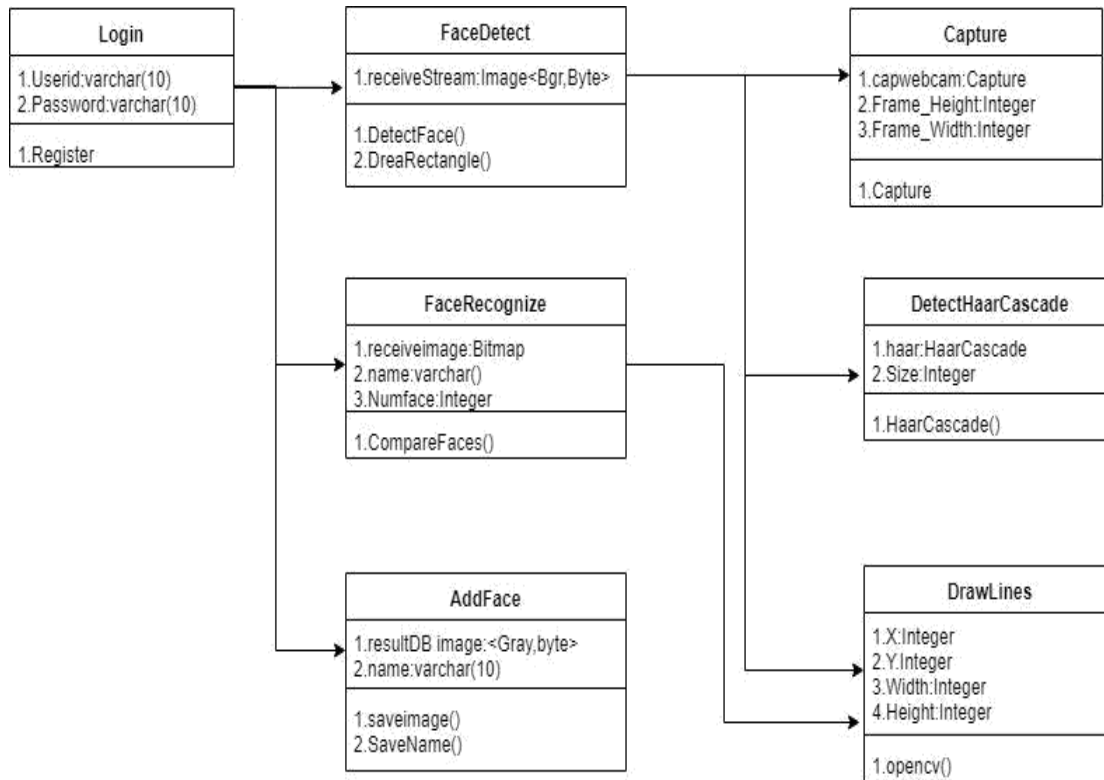
It is a model which is used to show the classes constituting a system and their interrelationship. It is based on UML. Only the important attributes and methods are shown in Class diagrams. In the initial period of analysis, the important attributes of the classes, which must be captured and the functionalities provided by the class may not be very clear. As the analysis progresses, the attributes and methods may be added. If more focus is on interrelationships of classes, then the attributes and methods may not be shown in the class diagram.

The class diagram is used to identify and classify the objects which constitute a system. It also includes the important attributes of the objects which must be captured.

6.3.2Diagram Notation



6.3.3 Diagram:



6.4 Object Diagram

6.4.1 Introduction

Object diagrams are derived from class diagrams so object diagrams are dependent upon class diagrams. Object diagrams represent an instance of a class diagram. The basic concepts are similar for class diagrams and object diagrams. Object diagrams also represent the static view of a system but this static view is a snapshot of the system at a particular moment.

Object diagrams are used to render a set of objects and their relationships as an instance.

The purpose of a diagram should be understood clearly to implement it practically. The purposes of object diagrams are similar to class diagrams.

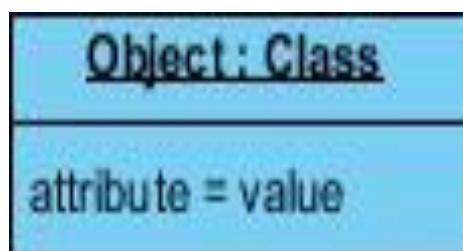
The difference is that a class diagram represents an abstract model consisting of classes and their relationships. However, an object diagram represents an instance at a particular moment, which is concrete in nature.

It means the object diagram is closer to the actual system behavior. The purpose is to capture the static view of a system at a particular moment.

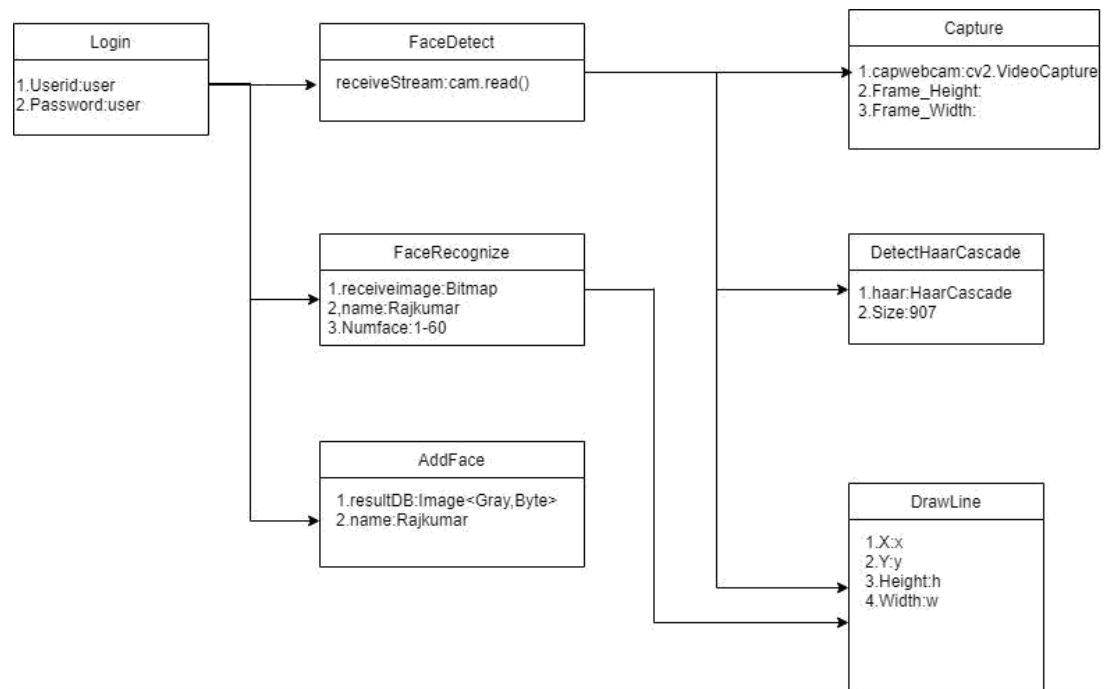
The purpose of the object diagram can be summarized as –

- Forward and reverse engineering.
- Object relationships of a system
- Static view of an interaction.
- Understand object behaviour and their relationship from practical perspective

6.4.2 Diagram Notation



6.4.3 Diagram:



6.5 Use-Case Diagram

6.5.1 Introduction

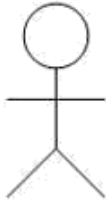
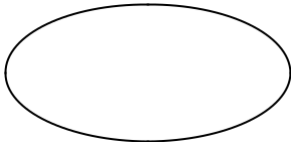

A use case diagram is a graphic depiction of the interactions among the elements of a system. A use case diagram is dynamic in nature, there should be some internal or external factors for making the interaction.

These internal and external agents are known as actors. Use case diagrams consist of actors, use cases and their relationships.

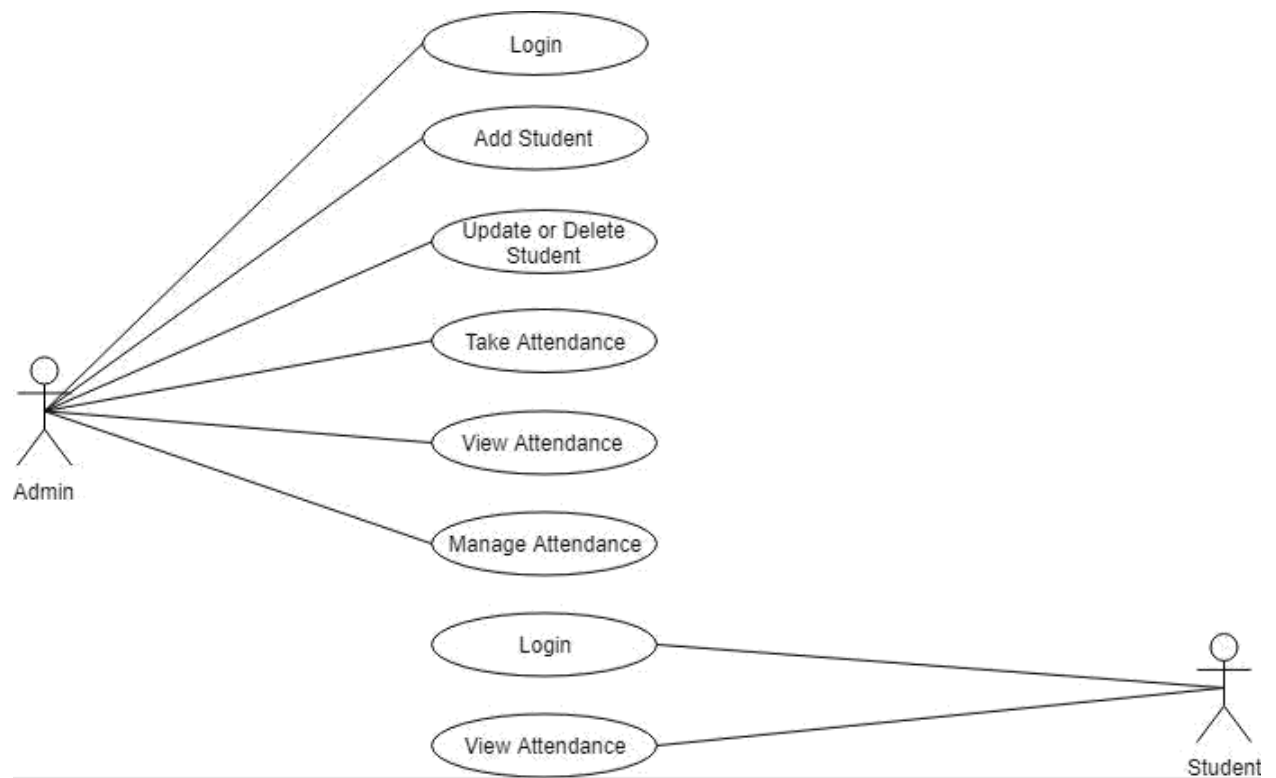
To draw a use case diagram, we should have the following items identified.

1. Functionalities to be represented as use case
2. Actors
3. Relationships among the use cases and actors.

6.5.2 Diagram Notations

Name	Symbols	Description
Actor		Actor specifies a role played by a user or any other system that interacts with the subject.
Use Case		Use case is a list of steps, typically defining interactions between an actor and a system, to achieve a goal.
Association		A line between actors and use cases. In complex diagrams, it is important to know which actors are associated with which use cases.

6.5.3 Diagram:



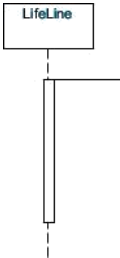


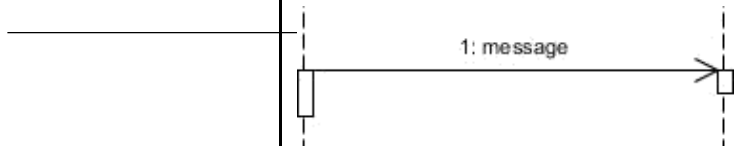
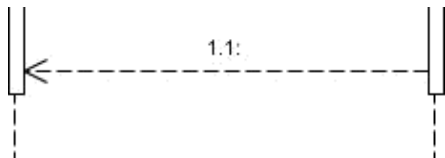
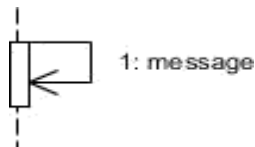
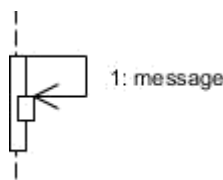
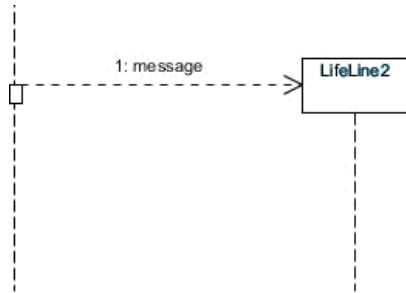
6.6 Sequence Diagram

6.6.1 Introduction

Sequence Diagrams are interaction diagrams that detail how operations are carried out. They capture the interaction between objects in the context of collaboration. Sequence Diagrams are time focus and they show the order of the interaction visually by using the vertical axis of the diagram to represent time what messages are sent and when.

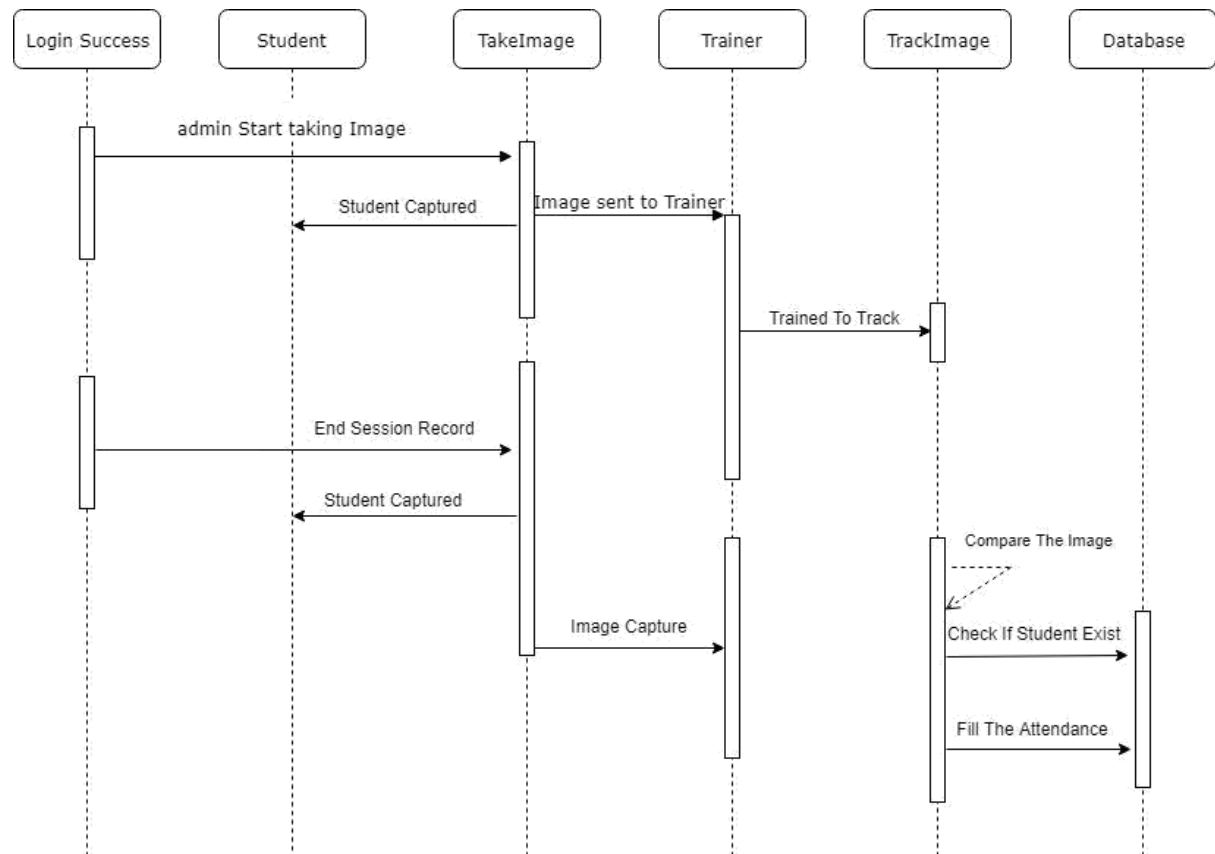
6.6.2 Diagram Notation

Name	Symbol	Description
Actor		<ol style="list-style-type: none">1. A type of role played by an entity that interacts with the subject (e.g., by exchanging signals and data)2. Represent roles played by human users, external hardware, or other subjects.
Lifeline		<ol style="list-style-type: none">1. A lifeline represents an individual participant in the Interaction
Activations		<ol style="list-style-type: none">1. A thin rectangle on a lifeline) represents the period during which an element is performing an operation.2. The top and the bottom of the of the rectangle are aligned with the initiation and the completion time respectively
Call Message		<ol style="list-style-type: none">1. A message defines a particular communication

		<p>between Lifelines of an Interaction.</p> <p>2. Call message is a kind of message that represents an invocation of operation of target lifeline.</p>
Return Message		<p>1. A message defines a particular communication between Lifelines of an Interaction.</p> <p>2. Return message is a kind of message that represents the pass of information back to the caller of a corresponded former message</p>
Self Message		<p>1. A message defines a particular communication between Lifelines of an Interaction.</p> <p>2. Self message is a kind of message that represents the invocation of message of the same lifeline.</p>
Recursive Message		<p>1. A message defines a particular communication between Lifelines of an Interaction.</p> <p>2. Recursive message is a kind of message that represents the invocation of message of the same lifeline.</p> <p>3. Its target points to an activation on top of the activation where the message was invoked from.</p>
Create Message		<p>1. A message defines a particular communication between Lifelines of an Interaction.</p> <p>2. Create message is a kind of message that represents the instantiation of (target) lifeline.</p>
Destroy Message		<p>1. A message defines a particular communication</p>

		<p>between Lifelines of an Interaction.</p> <p>2. Destroy message is a kind of message that represents the request of destroying the lifecycle of target.</p>
Duration Message		<p>1. A message defines a particular communication between Lifelines of an Interaction.</p> <p>2. Duration message shows the distance between two time instants for a message invocation</p>
Note		<p>1. A note (comment) gives the ability to attach various remarks to elements.</p> <p>2. A comment carries no semantic force, but may contain information that is useful to a modeller.</p>

6.6.3 Diagram:



6.7 Activity Diagrams

6.7.1 Introduction

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modelling Language, activity diagrams can be used to describe the business and operational step-by-step workflows of components in system. An activity diagram shows the overall flow of control. An activity diagram shows the overall flow of control. Activity diagrams are constructed from a limited repertoire of shapes, connected with arrows

The most important shape types:

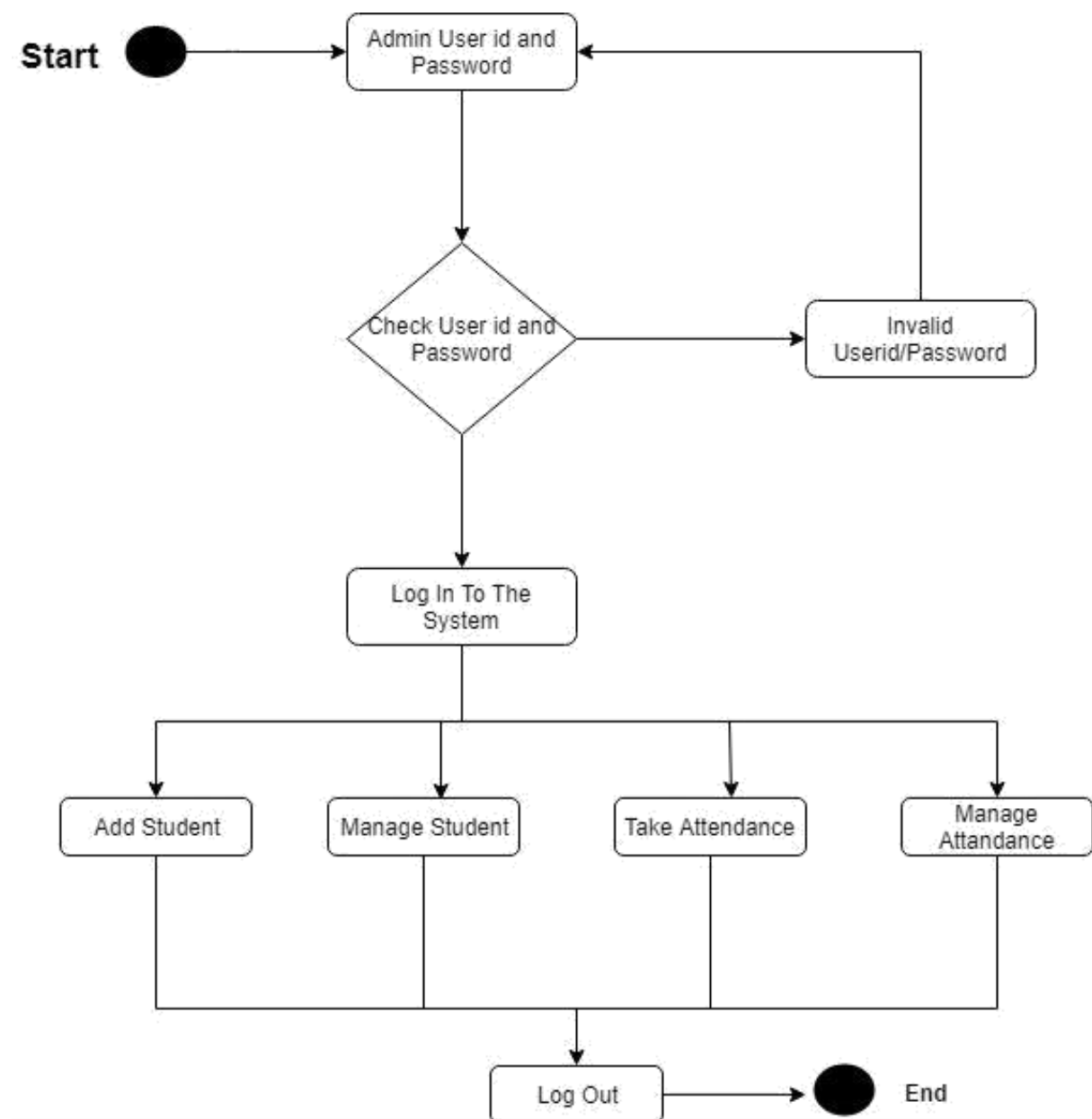
1. Rounded rectangle represents activities.
2. Diamonds represent decisions.
3. Bars represent the start (split) or end (join) of concurrent activities.
4. A black circle represents the start (initial state) of the workflow.
5. An encircled black circle represents the end (final state).
6. Arrows run from the start towards the end and represent the order in which activities happen.

6.7.2 Diagram Notation

The most important shape types:

1. Rounded rectangle represents activities.
2. Diamonds represent decisions.
3. Bars represent the start (split) or end (join) of concurrent activities.
4. A black circle represents the start (initial state) of the workflow.
5. An encircled black circle represents the end (final state).
6. Arrows run from the start towards the end and represent the order in which activities happen.

6.7.3 Diagram:



.8 Component Diagram

6.8.1 Introduction

A component diagram, also known as a UML component diagram, describes the organization and wiring of the physical components in a system. Component diagrams are often drawn to help model implementation details and double-check that every aspect of the system required functions is covered by planned development.

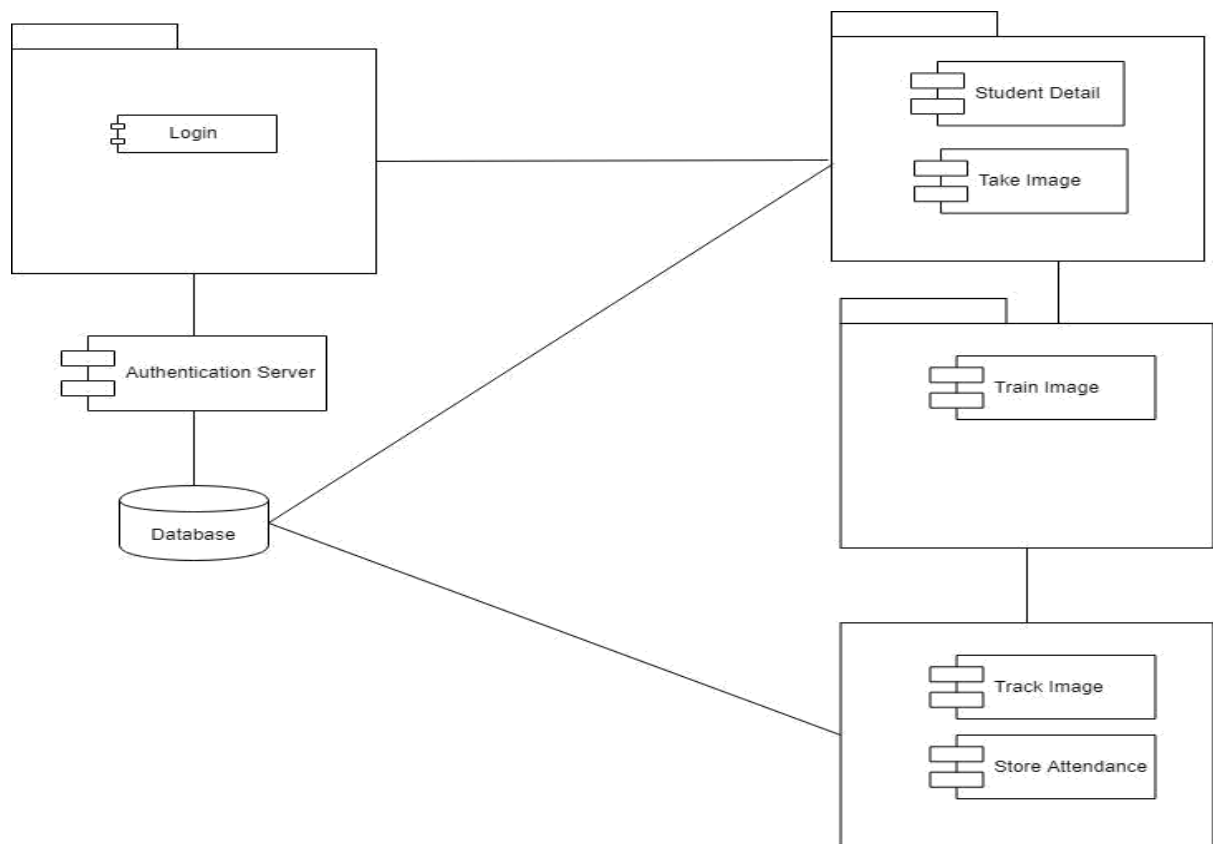
Component Diagrams:

- Give the physical view of the system in terms of implementation aspect. This is important for reusability and performance purpose.
- Constitute the Components, their interfaces and realizations, and dependencies between components.

Component Diagrams are used:

- To depict organizations and dependencies among Component type.
- To show allocation of “Classes” and “objects” to components in the physical design of the system.
- To indicate the “physical layering” and “partitioning” of the system Architecture. A component typically encompasses:
- Structure and behavior of a “Collaboration of classes” from the system design.
- Interfaces that describe a group of operations implemented by components.

6.8.2 Diagram:

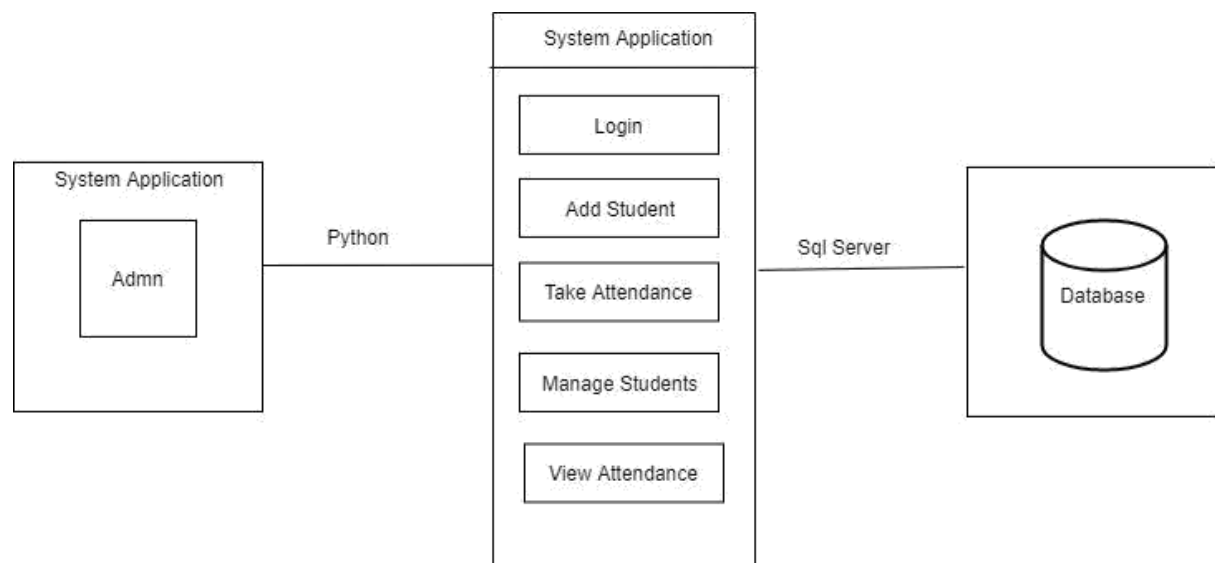


6.9 Deployment Diagram

6.9.1 Introduction

A deployment diagram is a UML diagram type that shows the execution architecture of a system, including nodes such as hardware or software execution environments, and the middleware connecting them. Deployment diagrams are typically used to visualize the physical hardware and software of a system.

6.9.2 Diagram:



6.10 Table Design

6.10.1 Introduction

A table is a collection of related data held in a table format within a database. It consists of columns and rows.

In relational database and flat file databases a *table* is a set of data elements (values) using a model of vertical columns (identifiable by name) and horizontal rows, the cell being the unit where a row and column intersect. A table has a specified number of columns, but can have any number of rows. Each row is identified by one or more values appearing in a particular column subset. A specific choice of columns which uniquely identify rows is called the primary key.

6.10.2 Table for admin Login

Sr.No	Column	Datatype	Description
1	username	Varchar(10)	It store the username of admin.
2	Password	Number	It store the password.

6.10.3 Table for Students

Sr.No	Column	Datatype	Description
1	Id	Int(10)	It store the Unique id of students
2	Name	Varchar(25)	It store the password.
3	Contact_no	Number	It store the contact number
4	Image	Image	It store the image of student

6.10.4 Table for Attendance

Sr.No	Column	Datatype	Description
1	Id	Int(10)	It store the Unique id of students
2	Name	Varchar(25)	It store the password.
3	Date	Date	It store the current date of attendance
4	Time	Time	It store the current time of attendance

Chapter 7

7.1 Program Source Code

```
from tkinter import *
import tkinter as tk
from tkinter import messagebox, Text
import cv2, os
import shutil
import csv
import numpy as np
from PIL import Image, ImageTk
import pandas as pd
import datetime
import time
import tkinter.ttk as ttk
import tkinter.font as font
import mysql.connector
import winsound
from tkinter import filedialog

window = tk.Tk()
window.title("Face_Recogniser")

#window.geometry("1280*720")
window.configure(background='lightblue')

window.attributes('-fullscreen', True)

window.grid_rowconfigure(0, weight=1)
window.grid_columnconfigure(0, weight=1)

lbl0 = tk.Label(window, text="Face-Recognition-Based-Attendance-System", bg='lightblue', fg="white", width=45, height=2, font=('times', 20, 'italic bold underline'))
lbl0.place(x=1, y=1)

lbl1 = tk.Label(window, text="Enter ID", width=20, height=2, fg="black", bg="lightblue", font=('times', 15, 'bold'))
lbl1.place(x=50, y=110)

txt = tk.Entry(window, width=20, bg="aliceblue", fg="black", font=('times', 15, 'bold'))
txt.place(x=250, y=125)

lbl2 = tk.Label(window, text="Enter Name", width=20, fg="black", bg="lightblue", height=2, font=('times', 15, 'bold'))
lbl2.place(x=50, y=180)

txt2 = tk.Entry(window, width=20, bg="aliceblue", fg="black", font=('times', 15, 'bold'))
txt2.place(x=250, y=200)

lbl3 = tk.Label(window, text="Contact No", width=20, height=2, fg="black", bg="lightblue", font=('times', 15, 'bold'))
lbl3.place(x=50, y=250)

txt3 = tk.Entry(window, width=20, bg="aliceblue", fg="black", font=('times', 15, 'bold'))
txt3.place(x=250, y=260)

lbl4 = tk.Label(window, text="Class", width=20, fg="black", bg="lightblue", height=2, font=('times', 15, 'bold'))
lbl4.place(x=50, y=320)

txt4 = tk.Entry(window, width=20, bg="aliceblue", fg="black", font=('times', 15, 'bold'))
txt4.place(x=250, y=330)
```

```

lbl5 = tk.Label(window, text="Message:
",width=20 ,fg="black" ,bg="lightblue" ,height=2 ,font=('times', 15, ' bold '))
lbl5.place(x=50, y=450)

message = tk.Label(window, text="" ,bg="aliceblue" ,fg="black" ,width=40 ,height=2,
activebackground = "yellow" ,font=('times', 15, ' bold '))
message.place(x=280, y=450)

def terminate():
    dialog_title = 'QUIT'
    dialog_text = 'Are you sure?'
    if tk.messagebox.askyesno(dialog_title,dialog_text,icon='question'):
        window.withdraw()

def attendance():
    at=tk.Tk()
    at.title("Attendance Viewer")
    #at.attributes('-fullscreen', True)
    at.configure(background='lightblue')
    at.grid_rowconfigure(0, weight=1)
    at.grid_columnconfigure(0, weight=1)
    lbx=Listbox(at,width=90,height=30,bg="aliceblue" ,font=('times','15'))
    lbx.place(x=1,y=40)
    def attendenced():
        filename=filedialog.askopenfilename(initialdir="D:\ProjectWork\working\Attendance",title=
        "select a file",filetype=(("CSV File","*.csv"),("All files","*")))
        lbl=Label(at,text=filename)
        lbl.place(x=400,y=20)
        #row=['Id','Name','Date','Time']
        with open(filename) as csvfile:
            reader=csv.DictReader(csvfile,delimiter=" ")
            for row in reader:
                lbx.insert(200,row)
            btn1=Button(at,text="view attend",command=attendenced,bg="skyblue")
            btn1.place(x=250,y=15)
            quitWindow = tk.Button(at, text="Quit",
            command=at.destroy,fg="black" ,bg="skyblue" ,width=10 ,height=2, activebackground =
            "red" ,font=('times', 15, ' bold '))
            quitWindow.place(x=1000, y=540)
        at.mainloop()

def create_studentw():
    student=tk.Tk()
    student.attributes('-fullscreen', True)
    student.title("Student Details")
    student.configure(background="lightblue")

    lbx=Listbox(student,width=30,height=18, font=('times','15'))
    lbx.place(x=2,y=240)
    def fetch():
        mydb=mysql.connector.connect(host="localhost",user="root",password="2901",db="facerec
        ognize")
        mycursor=mydb.cursor()
        mycursor.execute("select id,name,contact,class from tyscs order by id desc")
        ss=mycursor.fetchall()
        for s in ss:
            lbx.insert(0,s)
        btn4 = tk.Button(student, text="fetch",
        command=fetch ,fg="black" ,bg="skyblue" ,width=10 ,height=1 ,activebackground =
        "red" ,font=('times', 15, ' bold '))
        btn4.place(x=2, y=200)
        def clr():
            lbx.delete('0','end')

```

```

btn5 = tk.Button(student, text="clear",
command=clr ,fg="black" ,bg="skyblue" ,width=10 ,height=1 ,activebackground =
"red" ,font=('times', 15, ' bold '))
btn5.place(x=2, y=660)

```

def delete():

```

mydb=mysql.connector.connect(host="localhost",user="root",password="2901",db="tmp")
mycursor=mydb.cursor()
sql_stmt="delete from tycs where id=%s"
data=txt1.get()
mycursor.execute(sql_stmt,(data,))
mydb.commit()
lbl = tk.Label(student, text="Enter
ID",width=15 ,height=2 ,fg="black" ,bg="lightblue" ,font=('times', 15, ' bold '))
lbl.place(x=10, y=10)
txt1 = tk.Entry(student,width=20 ,bg="aliceblue" ,fg="black",font=('times', 15, ' bold '))
txt1.place(x=225, y=15)
btn1 = tk.Button(student, text="Delete",
command=delete ,fg="black" ,bg="skyblue" ,width=10 ,height=1 ,activebackground =
"red" ,font=('times', 15, ' bold '))
btn1.place(x=450, y=18)

```

def update():

```

mydb=mysql.connector.connect(host="localhost",user="root",password="2901",db="facerec
ognize")
mycursor=mydb.cursor()
sqldelete="update tycs set contact=%s where id=%s"
data=txt3.get()
data2=txt2.get()
mycursor.execute(sqldelete,(data,data2))
mydb.commit()

lbl2 = tk.Label(student, text="Enter
ID",width=15 ,fg="black" ,bg="lightblue" ,height=2 ,font=('times', 15, ' bold '))
lbl2.place(x=10, y=80)
txt2 = tk.Entry(student,width=20 ,bg="aliceblue" ,fg="black",font=('times', 15, ' bold '))
txt2.place(x=225, y=80)
lbl2 = tk.Label(student, text="Enter New
Contact",width=15 ,fg="black" ,bg="lightblue" ,height=2 ,font=('times', 15, ' bold '))
lbl2.place(x=10, y=120)
txt3 = tk.Entry(student,width=20 ,bg="aliceblue" ,fg="black",font=('times', 15, ' bold '))
txt3.place(x=225, y=120)
btn2 = tk.Button(student, text="Update contact",
command=update ,fg="black" ,bg="skyblue" ,width=10 ,height=2 ,activebackground =
"red" ,font=('times', 15, ' bold '))
btn2.place(x=450, y=80)

```

```

quitWindow = tk.Button(student, text="Quit",
command=student.destroy,fg="black" ,bg="skyblue" ,width=10 ,height=2,
activebackground = "red" ,font=('times', 15, ' bold '))
quitWindow.place(x=1000, y=540)
student.mainloop()

```

def opn3():

```

label=Label(window,text='Image removed',width=45,bg="lightblue")
label.place(x=650,y=10)
imabelbl=Label(width=40,height=15,bg="powderblue")
imabelbl.place(x=650,y=35)

```

```

def opn():
    global myimg

    filename=filedialog.askopenfilename(initialdir="D:\\ProjectWork\\working\\TrainingImage",title="Select A File",filetypes=(("Jpg files","*.jpg"),("all files","*..*")))
    label=Label(window,text=filename)
    label.place(x=650,y=10)
    myimg=ImageTk.PhotoImage(Image.open(filename))
    imagelbl=Label(image=myimg,bg="lightblue")
    imagelbl.place(x=730,y=40)
    lbli1=Label(window,text="",width=40,height=15,bg="powderblue")
    lbli1.place(x=650,y=35)
    btni1=Button(window,text="View Images",command=opn,fg="black",bg="skyblue",width=20,height=1, activebackground = "red",font=('times', 15, ' bold '))
    btni1.place(x=680,y=270)
    btni3=Button(window,text="clear",command=opn3,fg="black",bg="skyblue",width=10,height=1, activebackground = "red",font=('times', 15, ' bold '))
    btni3.place(x=740,y=310)

def opn4():
    label=Label(window,text="Image removed",width=45,bg="lightblue")
    label.place(x=1050,y=10)
    imagelbl=Label(width=40,height=15,bg="powderblue")
    imagelbl.place(x=1050,y=35)

def opn2():
    global unimg

    filename=filedialog.askopenfilename(initialdir="D:\\ProjectWork\\working\\ImagesUnknown",title="Select A File",filetypes=(("Jpg files","*.jpg"),("all files","*..*")))
    label=Label(window,text=filename)
    label.place(x=1050,y=10)
    unimg=ImageTk.PhotoImage(Image.open(filename))
    imagelbl=Label(image=unimg,bg="lightblue")
    imagelbl.place(x=1060,y=35)
    lbli2=Label(window,text="",width=40,height=15,bg="powderblue",)
    lbli2.place(x=1050,y=35)
    btni2=Button(window,text="View Unknown Images",command=opn2,fg="black",bg="skyblue",width=20,height=1, activebackground = "red",font=('times', 15, ' bold '))
    btni2.place(x=1070,y=270)
    btni4=Button(window,text="clear",command=opn4,fg="black",bg="skyblue",width=10,height=1, activebackground = "red",font=('times', 15, ' bold '))
    btni4.place(x=1140,y=310)

def clear():
    txt.delete(0, 'end')
    res = "Enter New Id"
    message.configure(text= res)

def clear2():
    txt2.delete(0, 'end')
    res = "Enter New Name"
    message.configure(text= res)

def clear3():
    txt3.delete(0, 'end')
    res="Enter New Contact"
    message.configure(text=res)

def clear4():
    txt3.delete(0, 'end')
    res="Enter Proper Class"
    message.configure(text=res)

```

```

def is_number(s):
    try:
        int(s)
        return True
    except ValueError:
        pass

    try:
        import unicodedata
        unicodedata.numeric(s)
        return True
    except (TypeError, ValueError):
        pass

    return False

def TakeImages():
    Id=(txt.get())
    Name=(txt2.get())
    Contact=(txt3.get())
    Class=(txt4.get())
    row=[Id,Name,Contact,Class]
    if(is_number(Id) and Name.isalpha()):
        try:
            mydb=mysql.connector.connect(host="localhost",user="root",password="2901",db="facerec
ognize")
            mycursor=mydb.cursor()
            mycursor.execute("create table if not exists `tycs` (id int,name varchar(25),contact
varchar(10), class varchar(10),primary key(id))")
            sql_stmt="insert into tycs(Id,Name,Contact,Class) values(%s,%s,%s,%s)"
            data=txt.get()
            data2=txt2.get()
            data3=txt3.get()
            data4=txt4.get()
            mycursor.execute(sql_stmt,(data,data2,data3,data4))
            mydb.commit()
        except mysql.connector.errors.IntegrityError:
            res="Id already Exist"
            message.configure(text=res)
        except mysql.connector.errors.DataError:
            res="please enter 10 digit contact number"
            message.configure(text=res)
        else:
            with open("StudentDetails\\StudentDetails.csv","a",newline="") as csvFile:
                writer = csv.writer(csvFile)
                writer.writerow(row)
                csvFile.close()
            cam = cv2.VideoCapture(0)
            harcascadePath = "haarcascade_frontalface_default.xml"
            detector=cv2.CascadeClassifier(harcascadePath)
            sampleNum=0
            row=[Id,Name,Contact,Class]
            while(True):
                ret, img = cam.read()
                gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
                faces = detector.detectMultiScale(gray, 1.5, 5)
                for (x,y,w,h) in faces:
                    cv2.rectangle(img,(x,y),(x+w,y+h),(255,0,0),2)
                    #incrementing sample number
                    sampleNum=sampleNum+1
                    #saving the captured face in the dataset folder TrainingImage
                    cv2.imwrite("TrainingImage\\"+Name+"."+Id+"."+str(sampleNum)+ ".jpg",
gray[y:y+h,x:x+w])
                    #display the frame
                    cv2.imshow('frame',img)

```

```

#wait for 100 milliseconds
    if cv2.waitKey(1) & 0xFF == ord('q'):
        break
    # break if the sample number is morethan 100
    elif sampleNum>99:
        break
    cam.release()
    cv2.destroyAllWindows()
    res = "Images Saved for ID:" + Id + " Name: " + Name
    message.configure(text=res)
    row = [Id,Name,Contact,Class]

else:
    if(is_number(Id)):
        res="Enter Alphabetical Name"
        message.configure(text=res)
    if(Name.isalpha()):
        res="Enter Numeric Id"
        message.configure(text=res)

def TrainImages():
    Name=(txt2.get())
    recognizer = cv2.face_LBPHFaceRecognizer.create()#recognizer =
cv2.face.LBPHFaceRecognizer_create()#cv2.createLBPHFaceRecognizer()
    harcascadePath = "haarcascade_frontalface_default.xml"
    detector =cv2.CascadeClassifier(harcascadePath)
    faces,Id= getImagesAndLabels(path)
    recognizer.train(faces, np.array(Id))
    recognizer.save("Trained\Trainer.yml")
    res = "Image Trained For "+Name#.join(str(f) for f in Id)
    message.configure(text= res)
path='TrainingImage'
def getImagesAndLabels(path):
    #get the path of all the files in the folder
    imagePath=[os.path.join(path,f) for f in os.listdir(path)]
    #print(imagePaths)

    #create empth face list
    faces=[]
    #create empty ID list
    Ids=[]
    #now looping through all the image paths and loading the Ids and the images
    for imagePath in imagePath:
        #loading the image and converting it to gray scale
        pilImage=Image.open(imagePath).convert('L')
        #Now we are converting the PIL image into numpy array
        imageNp=np.array(pilImage,'uint8')
        #getting the Id from the image
        Id=int(os.path.split(imagePath)[-1].split(".")[1])
        # extract the face from the training image sample
        faces.append(imageNp)
        Ids.append(Id)
    return faces,Ids

def TrackImages():
    recognizer = cv2.face.LBPHFaceRecognizer_create()#cv2.createLBPHFaceRecognizer()
    recognizer.read("Trained\Trainer.yml")
    harcascadePath = "haarcascade_frontalface_default.xml"
    faceCascade = cv2.CascadeClassifier(harcascadePath);
    df=pd.read_csv("StudentDetails\StudentDetails.csv")
    cam = cv2.VideoCapture(0)
    font = cv2.FONT_HERSHEY_SIMPLEX
    col_names = ['Id','Name','Date','Time','Subject']
    attendance = pd.DataFrame(columns = col_names)
    subject=""

```

```

while True:
    ret, im =cam.read()
    gray=cv2.cvtColor(im,cv2.COLOR_BGR2GRAY)
    faces=faceCascade.detectMultiScale(gray, 1.5,5)

for(x,y,w,h) in faces:
    cv2.rectangle(im,(x,y),(x+w,y+h),(0,255,0),2)
    Id, conf = recognizer.predict(gray[y:y+h,x:x+w])
    if(conf < 50):
        ts = time.time()
        date = datetime.datetime.fromtimestamp(ts).strftime('%Y-%m-%d')
        timeStamp = datetime.datetime.fromtimestamp(ts).strftime('%H:%M:%S')
        aa=df.loc[df['Id'] == Id]['Name'].values
        tt=str(Id)+"-"+aa
        tm=datetime.datetime.now()
        if tm.hour==7 and tm.minute>=00:
            subject='Python'
        elif tm.hour==8 and tm.minute>=00:
            subject='Android'
        elif tm.hour==9 and tm.minute>=00:
            subject='Ethical hacking'
        elif tm.hour==10 and tm.minute>=00:
            subject='Data Science'
        elif tm.hour==11 and tm.minute>=00:
            subject='c programming'
        elif tm.hour==12 and tm.minute>=00:
            subject='Advance DBMS'
        elif tm.hour==13 and tm.minute>=00:
            subject='Cloud Computing'
        elif tm.hour==14 and tm.minute>=00:
            subject='Ethical hacking'
        elif tm.hour==15 and tm.minute>=00:
            subject='Data Structure'
        else:
            subject='Unknown'
        attendance.loc[len(attendance)] = [Id,aa,date,timeStamp,subject]

    else:
        Id='Unknown'
        tt=str(Id)
        fr=2500
        dur=750
        winsound.Beep(fr,dur)
        if(conf > 75):
            noOfFile=len(os.listdir("ImagesUnknown"))+1
            cv2.imwrite("ImagesUnknown\Image"+str(noOfFile) + ".jpg", im[y:y+h,x:x+w])
            cv2.putText(im,str(tt),(x,y+h), font, 0.8,(255,255,255),2)
            attendance=attendance.drop_duplicates(subset=['Id'],keep='first')
            cv2.imshow('im',im)
            if (cv2.waitKey(1)==ord('q')):
                break
        ts = time.time()
        date = datetime.datetime.fromtimestamp(ts).strftime('%Y-%m-%d')
        timeStamp = datetime.datetime.fromtimestamp(ts).strftime('%H:%M:%S')
        Hour,Minute,Second=timeStamp.split(":")
        #fileName="Attendance\Attendance.csv"
        fileName="Attendance\Attendance_"+subject+".csv"
        attendance.to_csv(fileName,index=False)
        cam.release()
        cv2.destroyAllWindows()
        #print(attendance)
        res="Successfully Attendance Taken "
        message.config(text= res)

```



```

clearButton = tk.Button(window, text="Clear",
command=clear ,fg="black" ,bg="skyblue" ,width=10 ,height=1 ,activebackground =
"cyan" ,font=('times', 15, ' bold '))
clearButton.place(x=500, y=120)
clearButton2 = tk.Button(window, text="Clear",
command=clear2 ,fg="black" ,bg="skyblue" ,width=10 ,height=1 ,activebackground =
"red" ,font=('times', 15, ' bold '))
clearButton2.place(x=500, y=190)
clearButton3 = tk.Button(window, text="Clear",
command=clear3 ,fg="black" ,bg="skyblue" ,width=10 ,height=1 ,activebackground =
"red" ,font=('times', 15, ' bold '))
clearButton3.place(x=500, y=260)

```

47

```

clearButton4 = tk.Button(window, text="Clear",
command=clear4 ,fg="black" ,bg="skyblue" ,width=10 ,height=1 ,activebackground =
"red" ,font=('times', 15, ' bold '))
clearButton4.place(x=500, y=330)

```

```

takeImg = tk.Button(window, text="Register",
command=TakeImages ,fg="black" ,bg="skyblue" ,width=10 ,height=2, activebackground
= "red" ,font=('times', 15, ' bold '))
takeImg.place(x=200, y=540)
trainImg = tk.Button(window, text="Train Images",
command=TrainImages ,fg="black" ,bg="skyblue" ,width=10 ,height=2, activebackground
= "red" ,font=('times', 15, ' bold '))
trainImg.place(x=500, y=540)
trackImg = tk.Button(window, text="Take Attendance",
command=TrackImages ,fg="black" ,bg="skyblue" ,width=15 ,height=2,
activebackground = "red" ,font=('times', 15, ' bold '))
trackImg.place(x=800, y=540)
quitWindow = tk.Button(window, text="Quit",
command=terminate ,fg="black" ,bg="skyblue" ,width=10 ,height=2, activebackground =
"lightskyblue" ,font=('times', 15, ' bold '))
quitWindow.place(x=1100, y=540)

```

```

StudentsDetails= tk.Button(window, text="Students",
command=create_studentw ,fg="black" ,bg="skyblue" ,width=10 ,height=2,
activebackground = "red" ,font=('times', 15, ' bold '))
StudentsDetails.place(x=600, y=640)

```

```

AttendanceDetails= tk.Button(window, text="Attendance",
command=attendance ,fg="black" ,bg="skyblue" ,width=10 ,height=2, activebackground =
"red" ,font=('times', 15, ' bold '))
AttendanceDetails.place(x=800, y=640)

```

```



window.mainloop()

```

Chapter 8.2 Output

1.Main Window

Face-Recognition-Based-Attendance-System

Enter ID	<input type="text"/>	<input type="button" value="Clear"/>		
Enter Name	<input type="text"/>	<input type="button" value="Clear"/>		
Contact No	<input type="text"/>	<input type="button" value="Clear"/>		
Class	<input type="text"/>	<input type="button" value="Clear"/>		

Message:

Activate Windows
Go to Settings to activate Windows.

2.RegisterStudent

Face-Recognition-Based-Attendance-System

Enter ID

Enter Name

Contact No

Class

Message:

Activate Windows
Go to Settings to activate Windows.

3.Take Photo



Face-Recognition-Based-Attendance-System

frame

Activate Windows
Go to Settings to activate Windows.

4.Saving Details And Photo

Face-Recognition-Based-Attendance-System



Enter ID	<input type="text" value="1"/>	<input type="button" value="Clear"/>		
Enter Name	<input type="text" value="Rajkumar"/>	<input type="button" value="Clear"/>		
Contact No	<input type="text" value="8149803489"/>	<input type="button" value="Clear"/>		
Class	<input type="text" value="Tycs"/>	<input type="button" value="Clear"/>		

Message: Images Saved for ID:1 Name: Rajkumar

Activate Windows
Go to Settings to activate Windows.

5.Train The Machine

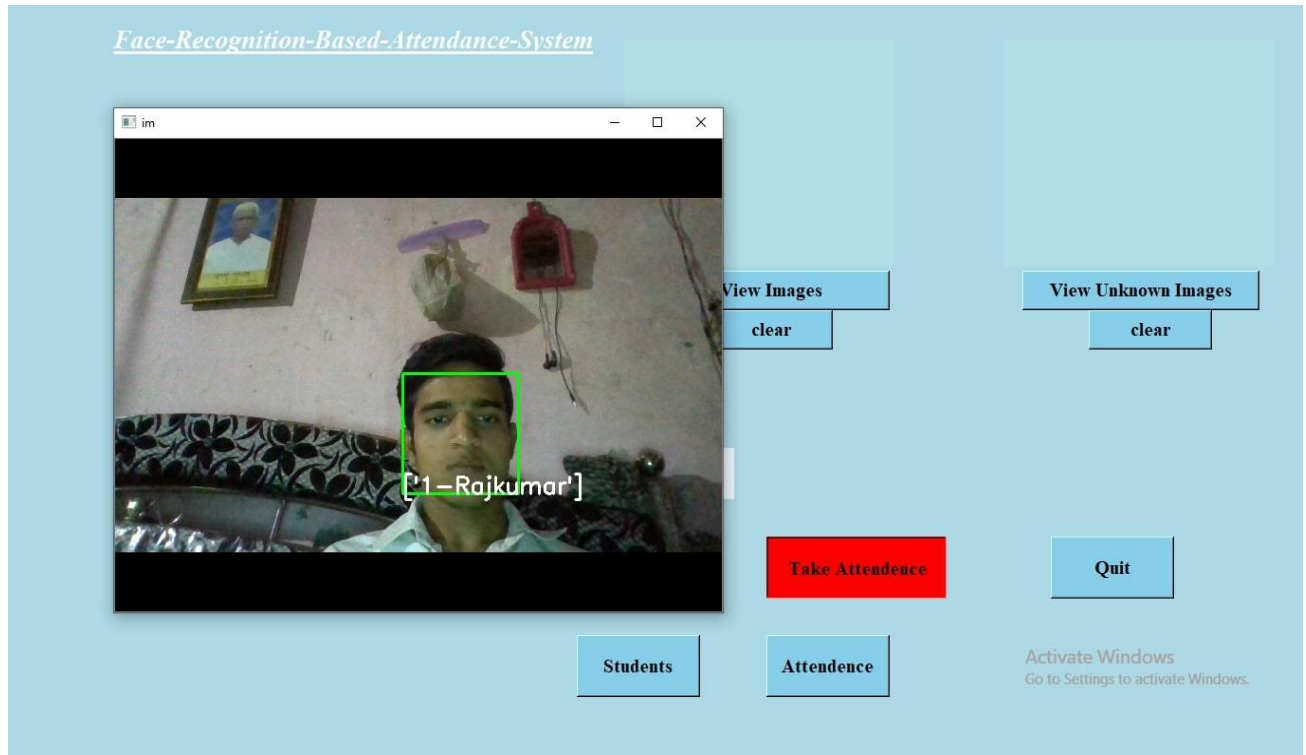
Face-Recognition-Based-Attendance-System

Enter ID	<input type="text" value="1"/>	<input type="button" value="Clear"/>		
Enter Name	<input type="text" value="Rajkumar"/>	<input type="button" value="Clear"/>		
Contact No	<input type="text" value="8149803489"/>	<input type="button" value="Clear"/>		
Class	<input type="text" value="Tycs"/>	<input type="button" value="Clear"/>		

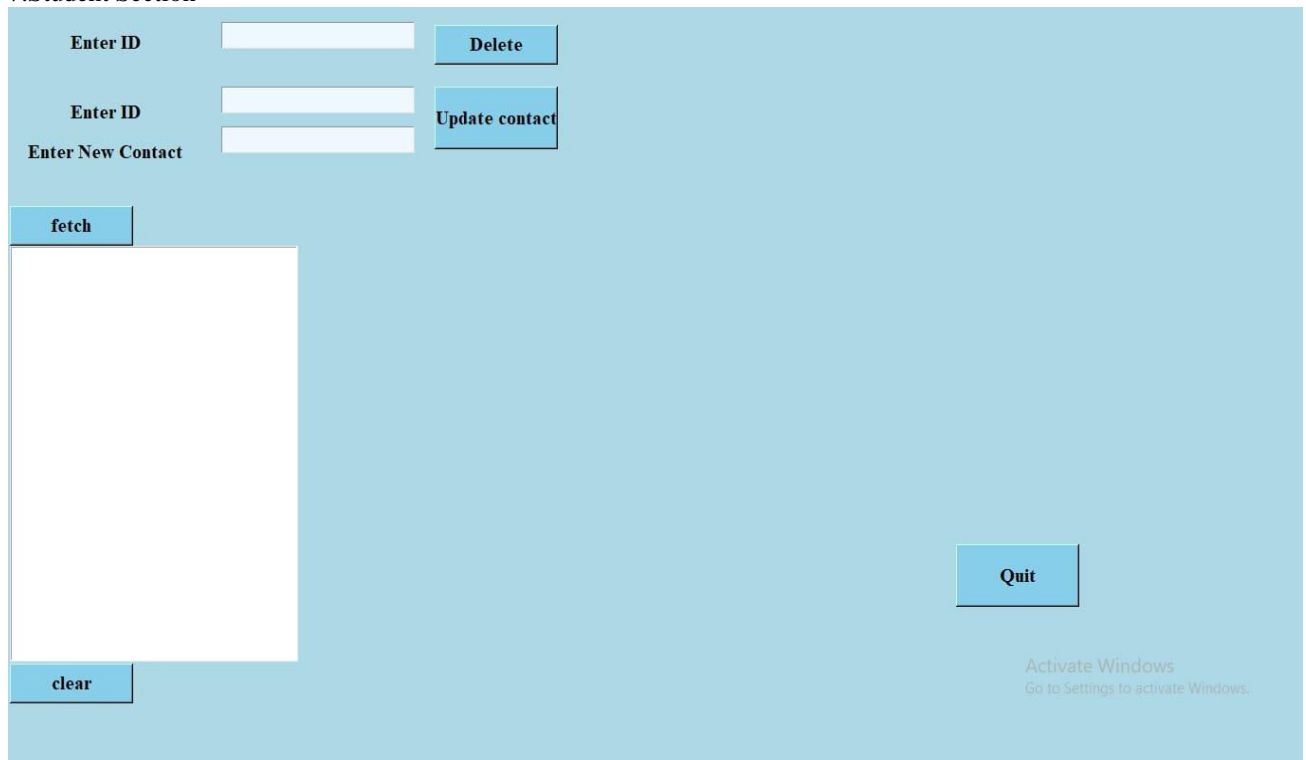
Message: Image Trained For Rajkumar

Activate Windows
Go to Settings to activate Windows.

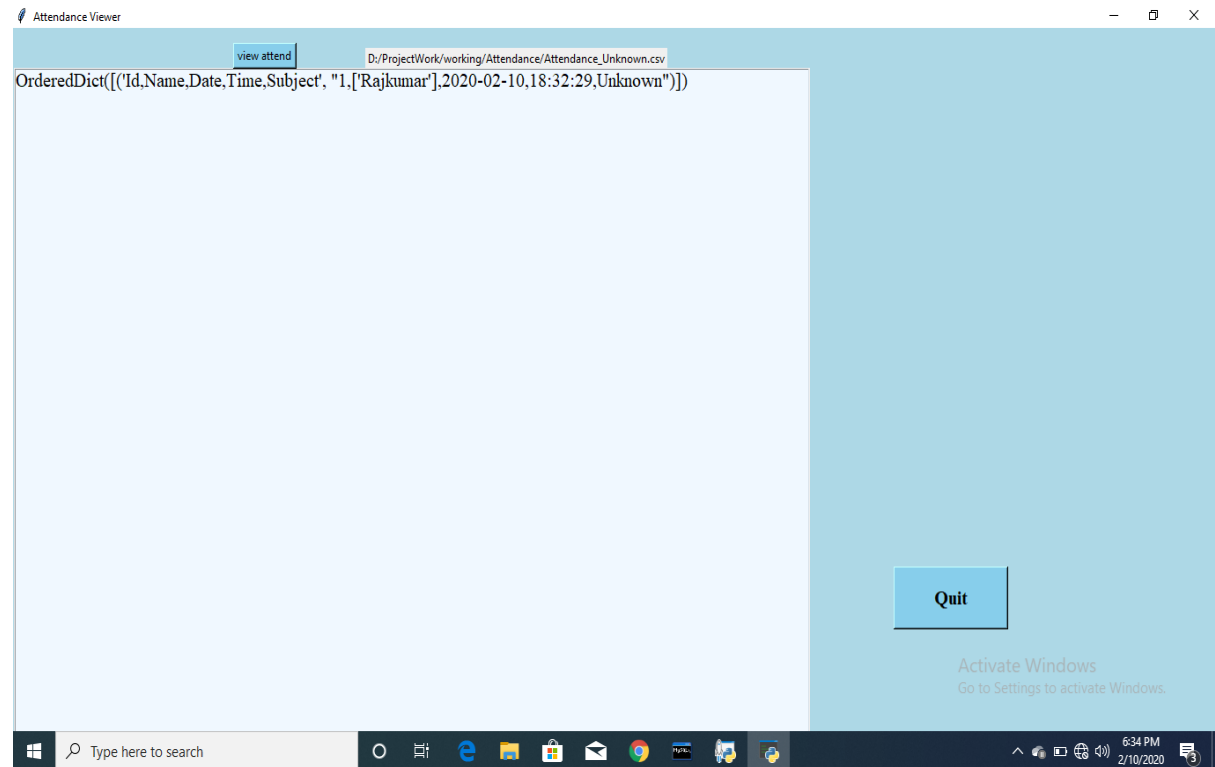
6. Take attendance



7. Student Section



8..Attendance Section



Chapter 8

System Testing

System testing of software or hardware is testing conducted on a complete, integrated system to evaluate the system's compliance with its specified requirements. System testing falls within the scope of black-box testing, and as such, should require no knowledge of the inner design of the code or logic. As a rule, system testing takes, as its input, all of the "integrated" software components that have passed integration testing and also the software system itself integrated with any applicable hardware system(s). The purpose of integration testing is to detect any inconsistencies between the software units that are integrated together (called assemblages) or between any of the assemblages and the hardware. System testing is a more limited type of testing; it seeks to detect defects both within the "inter-assemblages" and also within the system as a whole.

Testing the whole system

System testing is performed on the entire system in the context of a Functional Requirement Specification(s) (FRS) and/or a System Requirement Specification (SRS). System testing tests not only the design, but also the behavior and even the believed expectations of the customer. It is also intended to test up to and beyond the bounds defined in the software/hardware requirements specification(s).

Testing Used

1. At the beginning when Graphical User Interface (GUI) and functioning of application were incorporated together face processing wasn't as smooth as expected for that various set of functions has to be created to implement the whole program.
2. Testing of every function with real human face was done at the beginning it took time for processing but as machine got use to words and pic-based inputs, processing became smoother as compared to early stage.
3. Application was tested by human face with different person's face to test the image processing.

8.1 Validation

Test Cases, Test Results and Test Data

What is Test Case?

“A Test Case has a component that describe an input, action or event expected response, to determine if a feature of an application is working correctly.” Software testing can be stated as the process of validating and verifying that a computer program/application/product:

- 1.Meets the requirements that guided its design and development.
- 2.Works as expected
- 3.Can be implemented with the same characters.
- 4.And satisfies the needs of Stakeholders.

Why we Write Test Case?

A Test Case in Software Engineering is a set of conditions or variables under which a tester will determine whether an application, software system or one of its features is working as it was originally established for it to do. Test Cases bring some sort of standardization and minimize the ad-hoc approach in testing.

Test Case

Case 1

Purpose	Check for face base input and outut
Assumption	Application has been launched
Steps	<ul style="list-style-type: none">● click on register button and take the student face● wait for the output showing as image saved For student name and Id number
Expected Result	Processing Should be less and result showing fastly
Actual Result	Processing time is moderate

Conclusion

Chapter 9

Conclusion

9.1 Conclusion

Automated Attendance System has been envisioned for the purpose of reducing the errors that occur in the traditional (manual) attendance taking system. The aim is to automate and make a system that is useful to the organization such as an institute. The efficient and accurate method of attendance in the office environment that can replace the old manual methods. This method is secure enough, reliable and available for use. No need for specialized hardware for installing the system in the office. It can be constructed using a camera and computer

9.2 Future Scope

Today, one of the fields that uses facial recognition the most is security. Facial recognition is a very effective tool that can help law enforcers recognize criminals and software companies are leveraging the technology to help users access their technology. This technology can be further developed to be used in other avenues such as ATMs, accessing confidential files, or other sensitive materials. This can make other security measures such as passwords and keys obsolete.

Another way that innovators are looking to implement facial recognition is within subways and other transportation outlets. They are looking to leverage this technology to use faces as credit cards to pay for your transportation fee. Instead of having to go to a booth to buy a ticket for a fare, the face recognition would take your face, run it through a system, and charge the account that you've previously created. This could potentially streamline the process and optimize the flow of traffic drastically. The future is here.

References

References

OpenCV

https://docs.opencv.org/master/d9/df8/tutorial_root.html

https://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_objdetect/py_face_detection/py_face_detection.html

Websites

https://www.tutorialspoint.com/uml/uml_class_diagram.html

<https://stackoverflow.com/questions/10154633/load-csv-data-into-mysql-in-python>

YouTube Videos

<https://www.youtube.com/watch?v=oqMTdjcrAGk>