CHAPTER-1: INTRODUCTION

i)Existing system

Now a days, In most of the colleges are using paper based attendance system. Mass production of paper and ink consumes a lot of energy and utilization of resources. With the world becoming aware about global warming, everyone wants to cut down on waste of such resources. Attendance monitoring and working hour calculation is very essential for almost every institution or organization. Typically there are two types of attendance system available,

a) Manual and
b) Automated.

a)Manual(pen and paper based attendance system)

Manual system involves the use of sheets of paper or books in taking attendance where employees fill out and managers oversee for accuracy. This method could be erroneous because sheets could be lost or damaged. Also the extraction of relevant data and the manual computation of working time is very time consuming. It takes an extra employee to check for the attendance and timing of other employees which includes cost overhead for the organization as well.

b)Automated(machine based attendance system)

On the other hand, automated time and attendance systems implies the use of electronic tags, barcode badges, magnetic stripe cards, biometrics (hand, fingerprint, or facial), and touch screens in place of paper sheets. In these aforementioned techniques, employees touch or swipe in order to provide their identification and also the entering and leaving time to calculate working hours. The provided information are recorded and automatically transferred to a computer for processing. Using an automated system for time and attendance monitoring reduces the errors of manual system and conserve optimal amount of time. But these automated systems require heterogeneous devices need to be located in the organization which is costly

ii)Proposed system

Facial recognition attendance

Identification of individuals in an organization for the purpose of attendance is one such application of face recognition. Maintenance and monitoring of attendance records plays a vital role in the analysis of performance of any organization. The purpose of developing attendance management system is to computerize the traditional way of taking attendance. Automated Attendance Management System performs the daily activities of attendance marking and analysis with reduced human intervention. The prevalent techniques and methodologies for detecting and recognizing face fail to overcome issues such as scaling, pose, illumination, variations, rotation, and occlusions. The proposed system aims to overcome the pitfalls of the existing systems and provides features such as detection of faces, extraction of the features, detection of extracted features, and analysis of students' attendance. The system integrates techniques such as image contrasts, integral images, color features and cascading classifier for feature detection. The system provides an increased accuracy due to use of a large number of features (Shape, Colour, LBP, wavelet, Auto-Correlation) of the face. Faces are recognized using Euclidean distance and k-nearest neighbor algorithms. Better accuracy is attained in results as the system takes into account the changes that occur in the face over the period of time and employs suitable learning algorithms. The system is tested for various use cases. We consider a specific area such as classroom attendance for the purpose of testing the accuracy of the system. The metric considered is the percentage of the recognized faces per total number of tested faces of the same person. The system is tested under varying lighting conditions, various facial expressions, presence of partial faces (in densely populated classrooms) and presence or absence of beard and spectacles. An increased accuracy (nearly 85%) is obtained in most of the cases considered.

Objectives:

The objectives of the project are given below:
1. Detection of unique face image amidst the other natural components such as walls, backgrounds etc.
2. Extraction of unique characteristic features of a face useful for face recognition.
3. Detection of faces amongst other face characters such as beard, spectacles etc.
4. Effective recognition of unique faces in a crowd(individual recognition in crowd). 5. Automated update in the database without human intervention.

Advantages

- The software can be used for security purposes in organizations and in secured zones.
- The software stores the faces that are detected and automatically marks attendance.
- The system is convenient and secure for the users.
- It saves their time and efforts.

Disadvantages

- The system don’t recognize properly in poor light so may give false results.
- It can only detect face from a limited distance.

iii)Software Requirements

1)python 3.6

2)opencv3.4 2

3)tkinter

4)firebase

5)numpy

6)pillow and

7) xlwrite

1.PYTHON

Python is an interpreted high-level programming language for general-purpose programming. Created by Guido van Rossum and first released in 1991, Python has a design philosophy that emphasizes code readability, notably using significant whitespace. It provides constructs that enable clear programming on both small and large scales.

Python features a dynamic type system and automatic memory management. It supports multiple programming paradigms, including object-oriented, imperative, functional and procedural, and has a large and comprehensive standard library.

Python interpreters are available for many operating systems. CPython, the reference implementation of Python, is open source software and has a community-based development model, as do nearly all of its variant implementations. CPython is managed by the non-profit Python Software Foundation.

Feature and Philosophy

Python is a multi-paradigm programming language. Object-oriented programming and structured programming are fully supported, and many of its features support functional programming and aspect- oriented programming (including by metaprogramming and metaobjects (magic methods). Many other paradigms are supported via extensions, including design by contract and logic programming.

Python uses dynamic typing, and a combination of reference counting and a cycle-detecting garbage collector for memory management. It also features dynamic name resolution (late binding), which binds method and variable names during program execution.

Python's design offers some support for functional programming in the Lisp tradition. It has filter (), map (), and reduce () functions; list comprehensions, dictionaries, and sets; and generator expressions. The standard library has two modules (itertools and functools) that implement functional tools borrowed from Haskell and Standard ML.
The language's core philosophy is summarized in the document The Zen of Python (PEP 20), which includes aphorisms such as:

- Beautiful is better than ugly.
- Explicit is better than implicit.
- Simple is better than complex.
- Complex is better than complicated.
- Readability counts.

Rather than having all of its functionality built into its core, Python was designed to be highly extensible. This compact modularity has made it particularly popular as a means of adding programmable interfaces to existing applications. Van Rossum's vision of a small core language with a large standard library and easily extensible interpreter stemmed from his frustrations with ABC, which espoused the opposite approach.

While offering choice in coding methodology, the Python philosophy rejects exuberant syntax (such as that of Perl) in favor of a simpler, less-cluttered grammar. As Alex Martelli put it: "To describe something as 'clever' is not considered a compliment in the Python culture." Python's philosophy rejects the Perl "there is more than one way to do it" approach to language design in favor of "there should be one—and preferably only one—obvious way to do it".Python's developers strive to avoid premature optimization and reject patches to non-critical parts of CPython that would offer marginal increases in speed at the cost of clarity. When speed is important, a Python programmer can move time-critical functions to extension modules written in languages such as C, or use PyPy, a just-in-time compiler. Cython is also available, which translates a Python script into C and makes direct C-level API calls into the Python interpreter.

An important goal of Python's developers is keeping it fun to use. This is reflected in the language's name— a tribute to the British comedy group Monty Python—and in occasionally playful approaches to tutorials and reference materials, such as examples that refer to spam and eggs (from a famous Monty Python sketch) instead of the standard foo and bar.

A common neologism in the Python community is pythonic, which can have a wide range of meanings related to program style. To say that code is pythonic is to say that it uses Python idioms well, that it is natural or shows fluency in the language, that it conforms with Python's minimalist philosophy and emphasis on readability. In contrast, code that is difficult to understand or reads like a rough transcription from another programming language is called unpythonic.

Users and admirers of Python, especially those considered knowledgeable or experienced, are often referred to as Pythonists, Pythonistas, and Pythoneers.

Syntaxes and Semantics:

Python is meant to be an easily readable language. Its formatting is visually uncluttered, and it often uses English keywords where other languages use punctuation. Unlike many other languages, it does not use curly brackets to delimit blocks, and semicolons after statements are optional. It has fewer syntactic exceptions and special cases than C or Pascal.

2. opencv3.4

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.

The library has more than 2500 optimized algorithms, which includes a comprehensive set of both classic and state-of-the-art computer vision and machine learning algorithms. These algorithms can be used to detect and recognize faces, identify objects, classify human actions in videos, track camera movements, track moving objects, extract 3D models of objects, produce 3D point clouds from stereo cameras, stitch images together to produce a high resolution image of an entire scene, find similar images from an image database, remove red eyes from images taken using flash, follow eye movements, recognize scenery and establish markers to overlay it with augmented reality, etc. OpenCV has more than 47 thousand people of user community and estimated number of downloads exceeding 14 million. The library is used extensively in companies, research groups and by governmental bodies.

Along with well-established companies like Google, Yahoo, Microsoft, Intel, IBM, Sony, Honda, Toyota that employ the library, there are many startups such as Applied Minds, VideoSurf, and Zeitera, that make extensive use of OpenCV. OpenCV's deployed uses span the range from stitching streetview images together, detecting intrusions in surveillance video in Israel, monitoring mine equipment in China, helping robots navigate and pick up objects at Willow Garage, detection of swimming pool drowning accidents in Europe, running interactive art in Spain and New York, checking runways for debris in Turkey, inspecting labels on products in factories around the world on to rapid face detection in Japan.

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 CUDAand 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.

•OpenCV's application areas include:

- •2D and 3D feature toolkits
- •Ego motion estimation
- •Facial recognition system
- •Gesture recognition
- •Human–Computer interaction (HCI)
- •Mobile robotics
- •Motion understanding
- •Object identification
- •Segmentation and recognition
- •Stereopsis stereo vision: depth perception from 2 cameras
- •Structure from motion (SFM)
- •Motion tracking
- •Augmented reality

•To support some of the above areas, OpenCV includes a statistical machine learning library that contains:

- •Boosting
- •Decision tree learning
- •Gradient boosting trees
- •Expectation-maximization algorithm
- •k-nearest neighbor algorithm
- •Naive Bayes classifier
- •Artificial neural networks
- •Random forest
- •Support vector machine (SVM)

3. Tkinter

Most of the time, the `tkinter` module 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. The two most important modules are the tkinter module itself, and a module called **Tkconstants**. The former automatically imports the latter, so to use Tkinter, all you need to do is to import one module:

```
import Tkinter
```

Or, more often:

```
from Tkinter import *
```

class Tkinter.**Tk**(screenName=None, baseName=None, className='Tk', useTk=1)

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.

Other modules that provide Tk support include:

ScrolledText

Text widget with a vertical scroll bar built in.

tkColorChooser

Dialog to let the user choose a color.

tkCommonDialog

Base class for the dialogs defined in the other modules listed here.

tkFileDialog

Common dialogs to allow the user to specify a file to open or save.

tkFont

Utilities to help work with fonts.

tkMessageBox

Access to standard Tk dialog boxes.

tkSimpleDialog

Basic dialogs and convenience functions.

Tkdnd

Drag-and-drop support for **Tkinter**. This is experimental and should become deprecated when it is replaced with the Tk DND.

turtle

Turtle graphics in a Tk window.

These have been renamed as well in Python 3; they were all made submodules of the new tkinter package.

This section is not designed to be an exhaustive tutorial on either Tk or Tkinter. Rather, it is intended as a stop gap, providing some introductory orientation on the system.

Credits:

- Tkinter was written by Steen Lumholt and Guido van Rossum.
- Tk was written by John Ousterhout while at Berkeley.
- This Life Preserver was written by Matt Conway at the University of Virginia.
- The html rendering, and some liberal editing, was produced from a FrameMaker version by Ken Manheimer.
- Fredrik Lundh elaborated and revised the class interface descriptions, to get them current with Tk 4.2.
- Mike Clarkson converted the documentation to LaTeX, and compiled the User Interface chapter of the reference manual.

4. Firebase

Firebase evolved from Envolve, a prior startup founded by James Tamplin and Andrew Lee in 2011. Envolve provided developers an API that enables the integration of online chat functionality into their websites. After releasing the chat service, Tamplin and Lee found that it was being used to pass application data that weren't chat messages. Developers were using Envolve to sync application data such as game state in real time across their users. Tamplin and Lee decided to separate the chat system and the real-time architecture that powered it.They founded Firebase as a separate company in September 2011 and it launched to the public in April 2012.

Firebase's first product was the Firebase Realtime Database, an API that synchronizes application data across iOS, Android, and Web devices, and stores it on Firebase's cloud. The product assists software developers in building real-time, collaborative applications.

In May 2012, one month after the beta launch, Firebase raised $1.1M in seed funding from venture capitalists Flybridge Capital Partners, Greylock Partners, Founder Collective, and New Enterprise Associates

In June 2013, the company further raised $5.6M in Series A funding from venture capitalists Union Square Ventures and Flybridge Capital Partners.

In 2014, Firebase launched two products. Firebase Hosting and Firebase Authentication. This positioned the company as a mobile backend as a service.

In October 2014, Firebase was acquired by Google.

In October 2015, Google acquired Divshot to merge it with the Firebase team.

In May 2016, at Google I/O, the company's annual developer conference, Firebase expanded their services to become a unified platform for mobile developers. Firebase now integrates with various other Google services, including Google Cloud Platform, AdMob, and Google Ads to offer broader products and scale for developers.[16] Google Cloud Messaging, the Google service to send push notifications to Android devices, was superseded by a Firebase product, Firebase Cloud Messaging, which added the functionality to deliver push notifications to iOS and Web devices.

In January 2017, Google acquired Fabric and Crashlytics from Twitter to add those services to Firebase.

In October 2017, Firebase launched Cloud Firestore, a realtime document database as the successor product to the original Firebase Realtime Database.

Services

Analytics

Firebase Analytics

Firebase Analytics is a cost-free app measurement solution that provides insight into app usage and user engagement.

Develop

Firebase Cloud Messaging

Formerly known as Google Cloud Messaging (GCM), Firebase Cloud Messaging (FCM) is a cross-platform solution for messages and notifications for Android, iOS, and web applications, which as of 2016 can be used at no cost.

Firebase Auth

Firebase Auth is a service that can authenticate users using only client-side code. It supports social login providers Facebook, GitHub, Twitter and Google (and Google Play Games). Additionally, it includes a user management system whereby developers can enable user authentication with email and password login stored with Firebase.

Realtime database

Firebase provides a realtime database and backend as a service. The service provides application developers an API that allows application data to be synchronized across clients and stored on Firebase's cloud.The company provides client libraries that enable integration with Android, iOS, JavaScript, Java, Objective-C, Swift and Node.jsapplications. The database is also accessible through a REST API and bindings for several JavaScript frameworks such as AngularJS, React, Ember.js and Backbone.js. The REST API uses the Server-Sent Events protocol, which is an API for creating HTTP connections for receiving push notifications from a server. Developers using the realtime database can secure their data by using the company's server-side-enforced security rules.[29] Cloud Firestore which is Firebase's next generation of the Realtime Database was released for beta use.

Firebase Storage

Firebase Storage provides secure file uploads and downloads for Firebase apps, regardless of network quality. The developer can use it to store images, audio, video, or other user-generated content. Firebase Storage is backed by Google Cloud Storage.

Firebase Hosting

Firebase Hosting is a static and dynamic web hosting service that launched on May 13, 2014. It supports hosting static files such as CSS, HTML, JavaScript and other files, as well as support through Cloud Functions.[31] The service delivers files over a content delivery network (CDN) through HTTP Secure (HTTPS) and Secure Sockets Layer encryption (SSL). Firebase partners with Fastly, a CDN, to provide the CDN backing Firebase Hosting. The company states that Firebase Hosting grew out of customer requests; developers were using Firebase for its real-time database but needed a place to host their content.

ML Kit

ML Kit is a mobile machine learning system for developers launched on May 8, 2018 in beta during the Google I/O 2018.[34] ML Kit API's feature a variety of features including text recognition, detecting faces, scanning barcodes, labelling images and recognising landmarks. It is currently available for iOS or Android developers. You may also import your own TensorFlow Lite models, if the given API's aren't enough.[35] The API's can be used on-device or on cloud.

5. Numpy

Numpy is a general-purpose array-processing package. It provides a high-performance multidimensional array object, and tools for working with these arrays. It is the fundamental package for scientific computing with Python. Besides its obvious scientific uses, Numpy can also be used as an efficient multi-dimensional container of generic data.

Arrays in Numpy

Array in Numpy is a table of elements (usually numbers), all of the same type, indexed by a tuple of positive integers. In Numpy, number of dimensions of the array is called rank of the array.A tuple of integers giving the size of the array along each dimension is known as shape of the array. An array class in Numpy is called as ndarray. Elements in Numpy arrays are accessed by using square brackets and can be initialized by using nested Python Lists.

Creating a Numpy Array

Arrays in Numpy can be created by multiple ways, with various number of Ranks, defining the size of the Array. Arrays can also be created with the use of various data types such as lists, tuples, etc. The type of the resultant array is deduced from the type of the elements in the sequences.

Note: Type of array can be explicitly defined while creating the array.

```
# Python program for
# Creation of Arrays
import numpy as np

# Creating a rank 1 Array
arr = np.array([1, 2, 3])
print("Array with Rank 1: \n",arr)

# Creating a rank 2 Array
arr = np.array([[1, 2, 3],
                [4, 5, 6]])
print("Array with Rank 2: \n", arr)

# Creating an array from tuple
arr = np.array((1, 3, 2))
print("\nArray created using "
      "passed tuple:\n", arr)
```

```
Array with Rank 1:
```

```
[1 2 3]

Array with Rank 2:

[[1 2 3]

[4 5 6]]



Array created using passed tuple:

[1 3 2]
```

Accessing the array Index

In a numpy array, indexing or accessing the array index can be done in multiple ways. To print a range of an array, slicing is done. Slicing of an array is defining a range in a new array which is used to print a range of elements from the original array. Since, sliced array holds a range of elements of the original array, modifying content with the help of sliced array modifies the original array content.

```
# Python program to demonstrate
# indexing in numpy array
import numpy as np

# Initial Array
arr = np.array([[-1, 2, 0, 4],
                [4, -0.5, 6, 0],
                [2.6, 0, 7, 8],
                [3, -7, 4, 2.0]])
print("Initial Array: ")
print(arr)

# Printing a range of Array
# with the use of slicing method
sliced_arr = arr[:2, ::2]
print ("Array with first 2 rows and"
    " alternate columns(0 and 2):\n", sliced_arr)

# Printing elements at
# specific Indices
Index_arr = arr[[1, 1, 0, 3],
                [3, 2, 1, 0]]
print ("\nElements at indices (1, 3), "
```

```
"(1, 2), (0, 1), (3, 0):\n", Index_arr)
```

Output:-Initial Array:

```
[[-1.   2.   0.   4. ]
 [ 4.  -0.5  6.   0. ]
 [ 2.6  0.   7.   8. ]
 [ 3.  -7.   4.   2. ]]
Array with first 2 rows and alternate columns(0 and 2):
 [[-1.  0.]
 [ 4.  6.]]


Elements at indices (1, 3), (1, 2), (0, 1), (3, 0):
 [ 0. 54.  2.  3.]
```

6. Pillow

Python Imaging Library (abbreviated as **PIL**) (in newer versions known as Pillow) is a free library for the Python programming language that adds support for opening, manipulating, and saving many different image file formats. It is available for Windows, Mac OS X and Linux. The latest version of PIL is 1.1.7, was released in September 2009 and supports Python 1.5.2–2.7, with Python36 support to be released "later".

Development appears to be discontinued with the last commit to the PIL repository coming in 2011.Consequently, a successor project called **Pillow** has forked the PIL repository and added Python 3.x support. This fork has been adopted as a replacement for the original PIL in Linux distributions including Debain and Ubuntu

Capabilities

Pillow offers several standard procedures for image manipulation. These include:

•per-pixel manipulations,
•masking and transparency handling,
•image filtering, such as blurring, contouring, smoothing, or edge finding,
•image enhancing, such as sharpening, adjusting brightness, contrast or color,
•adding text to images and much more.

7. xl write

The Packages There are python packages available to work with Excel files that will run on any Python platform and that do not require either Windows or Excel to be used. They are fast, reliable and open source:

openpyxl

The recommended package for reading and writing Excel 2010 files (ie: .xlsx)

xlsxwriter

An alternative package for writing data, formatting information and, in particular, charts in the Excel 2010 format (ie: .xlsx)

xlrd

This package is for reading data and formatting information from older Excel files (ie: .xls)

xlwt

This package is for writing data and formatting information to older Excel files (ie: .xls)

xlutils

This package collects utilities that require both xlrd and xlwt, including the ability to copy and modify or filter existing excel files

The Mailing List / Discussion Group
There is a Google group dedicated to working with Excel files in Python, including the libraries listed above along with manipulating the Excel application via COM.

Commercial Development
The following companies can provide commercial software development and consultancy and are specialists in working with Excel files in Pytho

iv)Hardware Requirements

1)A standalone computer needs to be installed in the office room where the system is to be deployed. 🕐

2)Camera must be positioned in the class room to obtain the snapshots.

3)Optimum Resolution: 512 by 512 pixels.

4)Secondary memory to store all the images and database

Chapter – 2: Design of the Project

2.1 UML Diagrams

UML Concepts

The Unified Modelling Language (UML) is a standard language for writing software blue prints. UML stands for Unified Modeling Language. UML is different from the other common programming languages like C++, Java, COBOL etc. UML is a pictorial language used to make software blue prints. So UML can be described as a general purpose visual modeling language to visualize, specify, construct and document software system. Although UML is generally used to model software systems but it is not limited within this boundary. It is also used to model non software systems as well like process flow in a manufacturing unit etc.

UML is not a programming language but tools can be used to generate code in various languages using UML diagrams. UML has a direct relation with object oriented analysis and design. After some standardization UML has become an OMG (Object Management Group) standard. UML diagrams are the ultimate output of the entire discussion. All the elements, relationships are used to make a complete UML diagram and the diagram represents a system.

The UML is a language for

1.Visualizing.

2.Specifying

3.Constructing

4.Documenting

The UML is a language which provides vocabulary and the rules for combining words in that vocabulary for the purpose of communication. A modelling language is a language whose vocabulary and the rules focus on the conceptual and physical representation of a system. Modelling yields an understanding of a system.

Building Blocks of the UML:

The vocabulary of the UML encompasses three kinds of building blocks:

• Things

• Relationships

• Diagrams

Things are the abstractions that are first-class citizens in a model;

relationships tie these things together; diagrams group interesting collections of things.

Types of UML Diagrams

There are several UML Diagrams some of them are as follow

• Activity Diagram

• Class Diagram

• Use case Diagram

• Sequence Diagram

•Collaboration Diagram

•Component Diagram

• Deployment Diagram

2.1.1 Usecase Diagram

Use case diagrams are usually referred to as behavior diagrams used to describe a set of actions (use cases) that some system or systems (subject) should or can perform in collaboration with one or more external users of the system (actors).

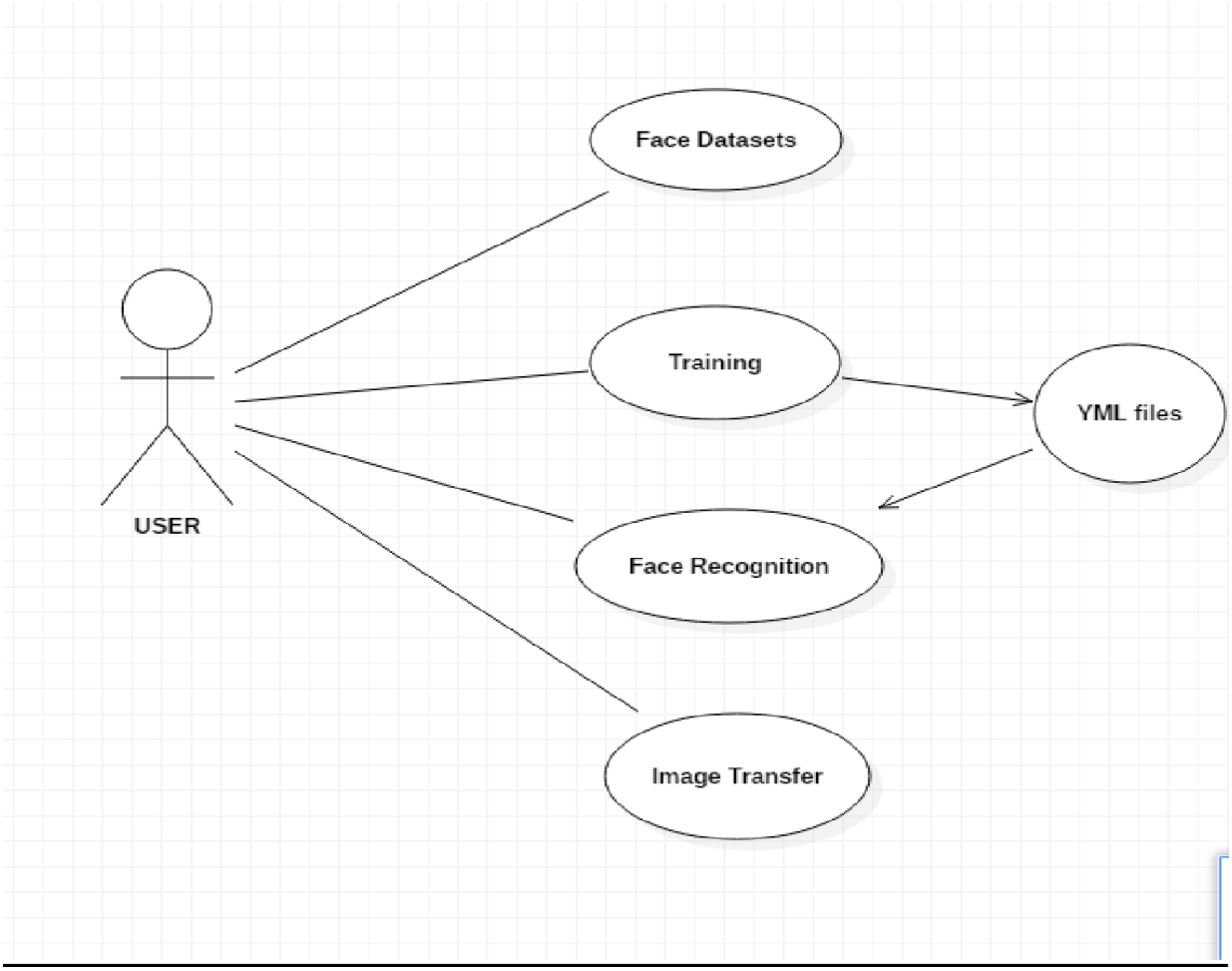


Fig no:1 Usecase diagram

2.1.2 Class Diagram

Class diagram is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among objects.

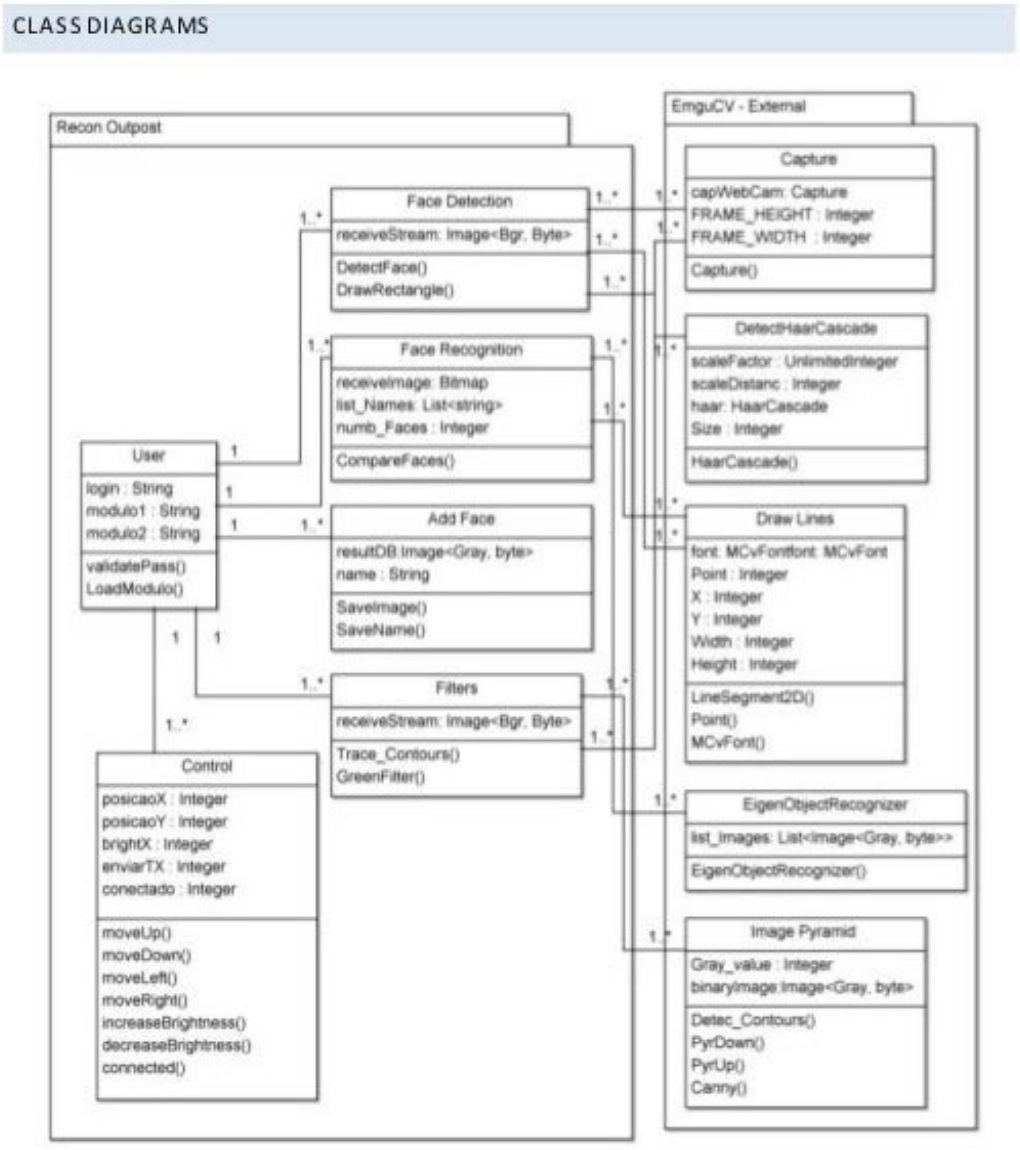


Fig no:2 Class diagram

2.1.3 Sequence Diagram

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.

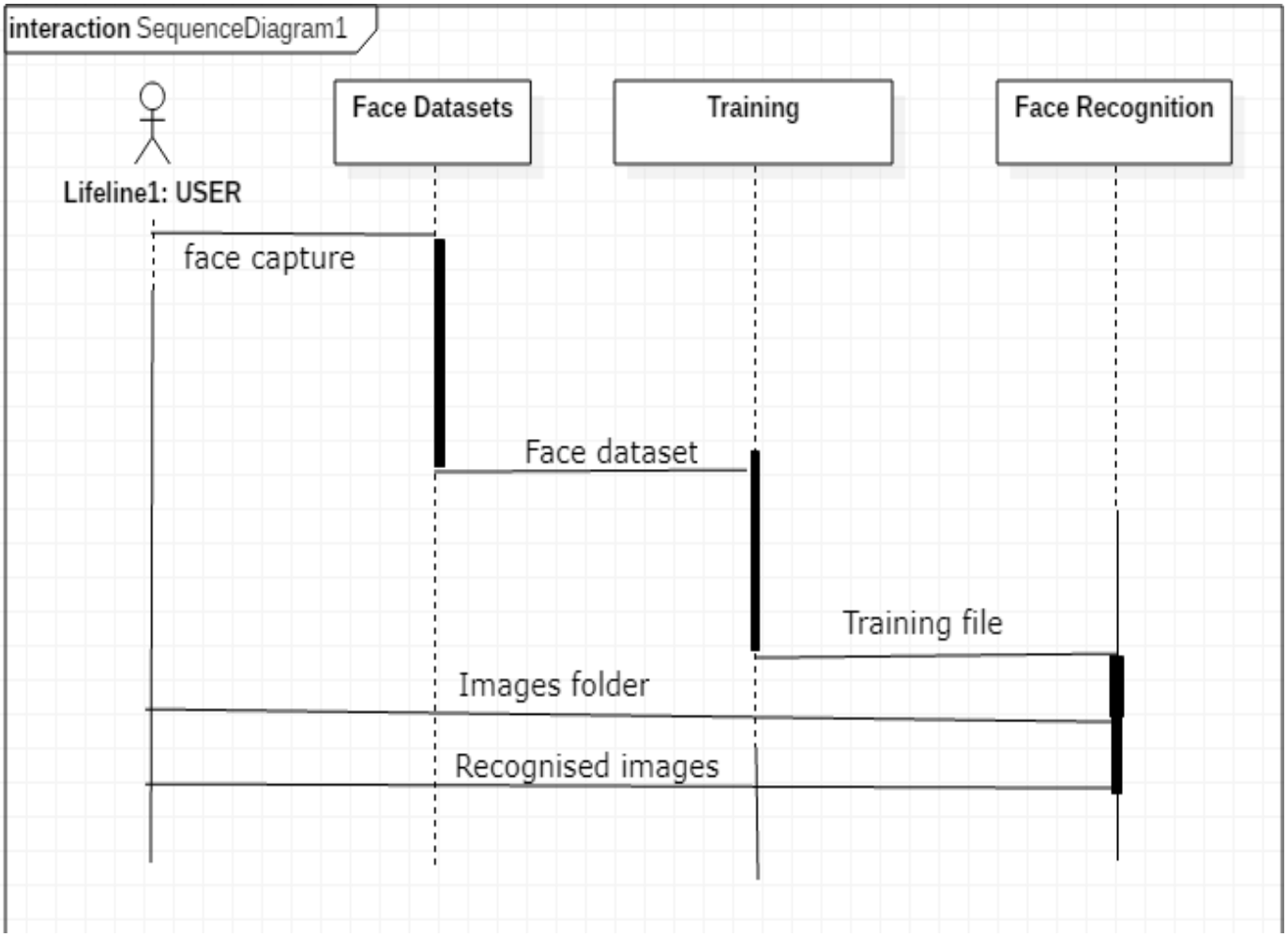


Fig no:3 Sequence diagram

2.1.4 Activity Diagram

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

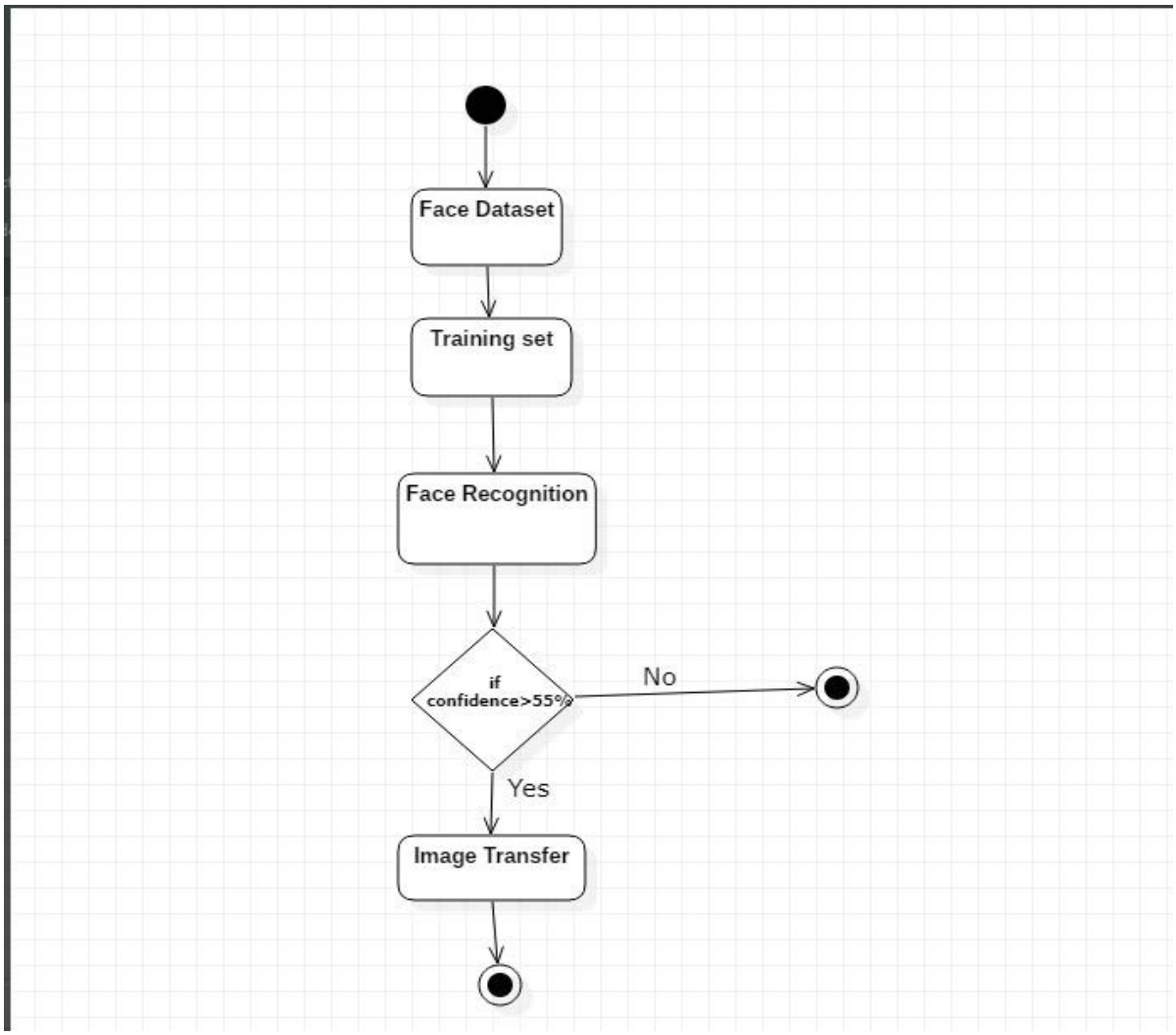


Fig no:4 Activity diagram

2.1.5 Component Diagram

Component diagram is a special kind of diagram in UML. The purpose is also different from all other diagrams discussed so far. It does not describe the functionality of the system but it describes the components used to make those functionalities.

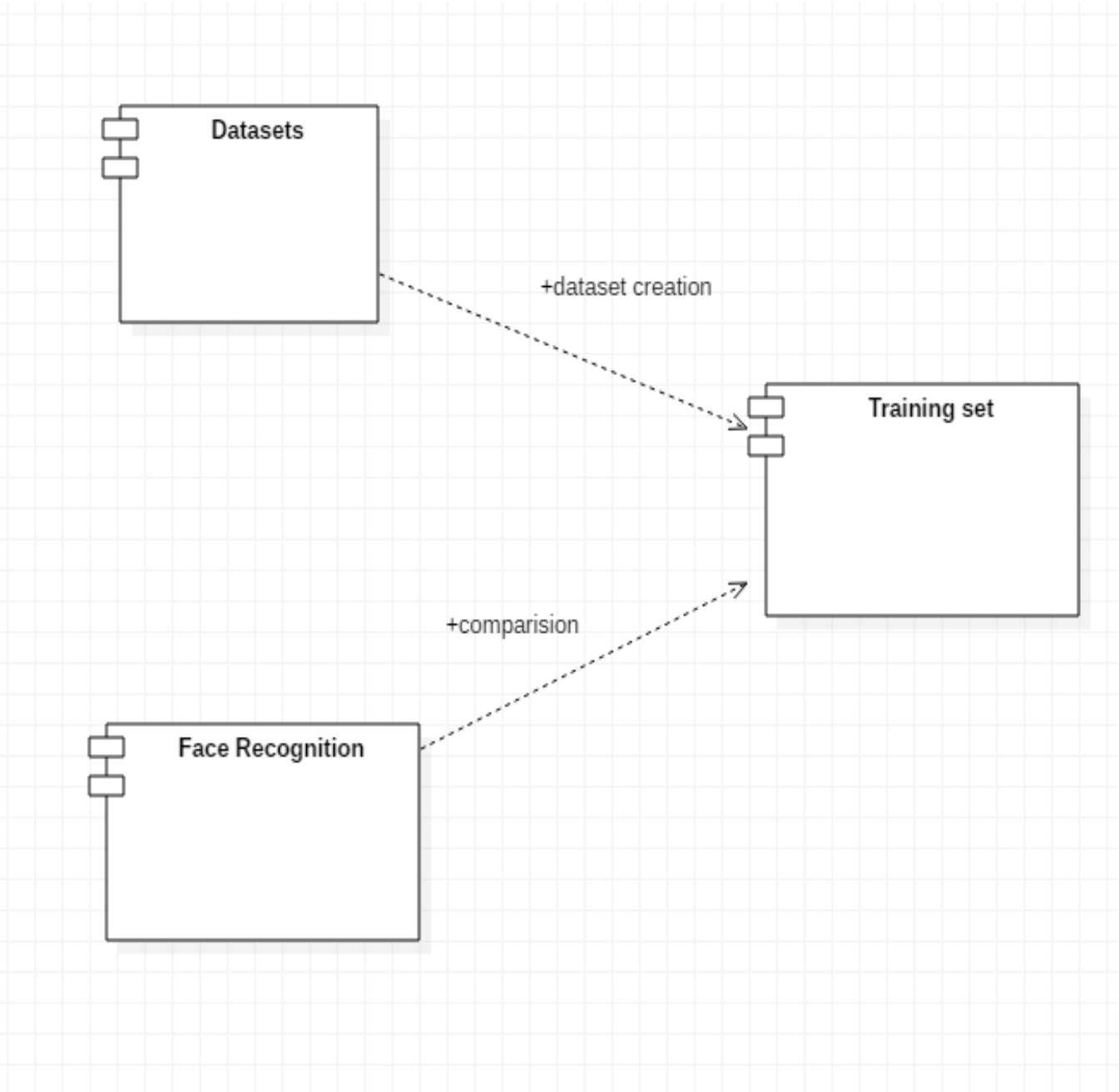


Fig no:5 Component diagram

2.1.6 Deployment Diagram

Deployment diagram is a structure diagram which shows architecture of the system as deployment(distribution) of software artifacts to deployment targets. Artifacts represent concrete elements in the physical world that are the result of a development process.

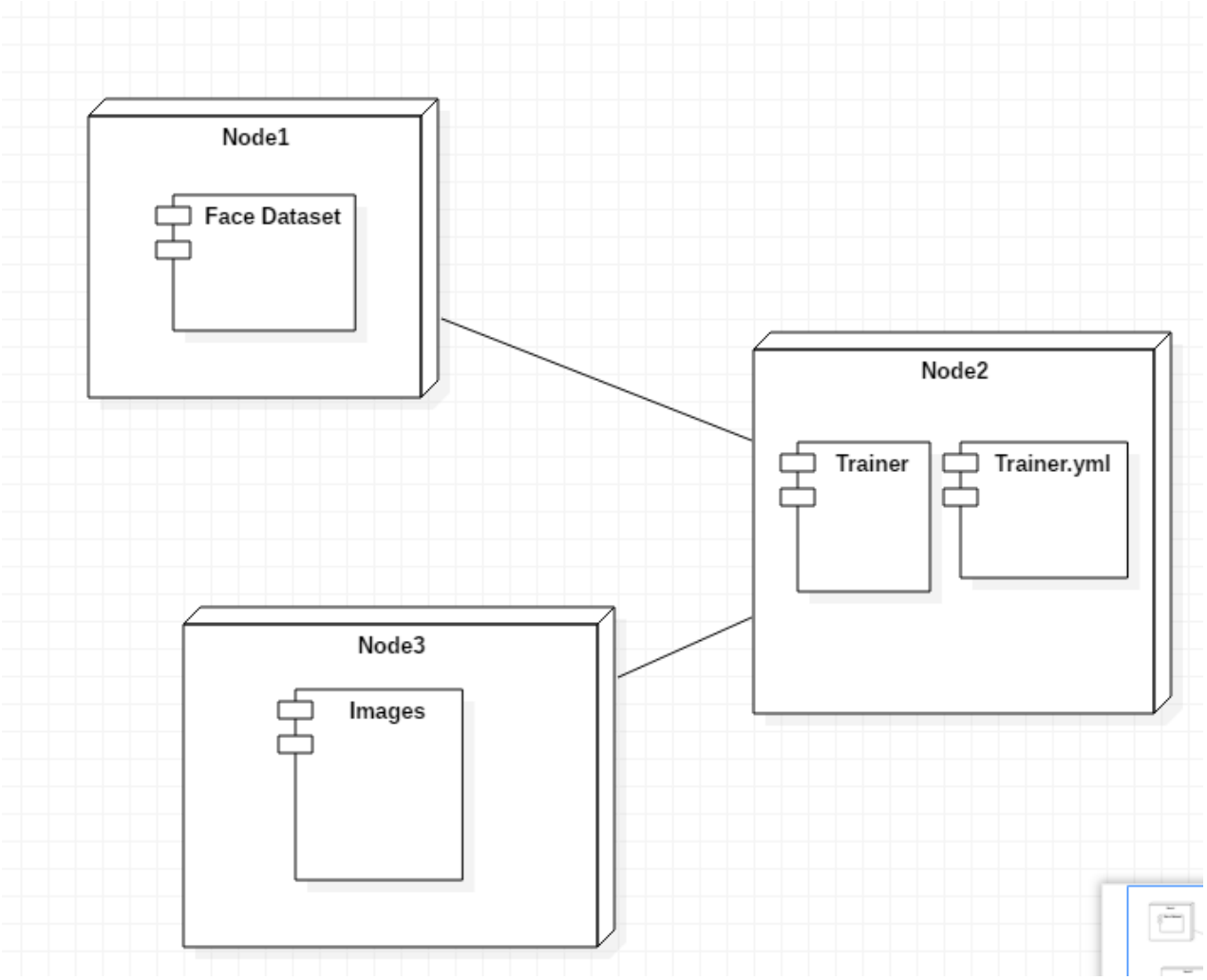


Fig no:6 Deployment diagram

2.1.7 Collaboration Diagram

A collaboration diagram, also called a communication diagram or interaction diagram, is an illustration of the relationships and interactions among software objects in the Unified Modeling Language (UML). The relationships between the objects are shown as lines connecting the rectangles.

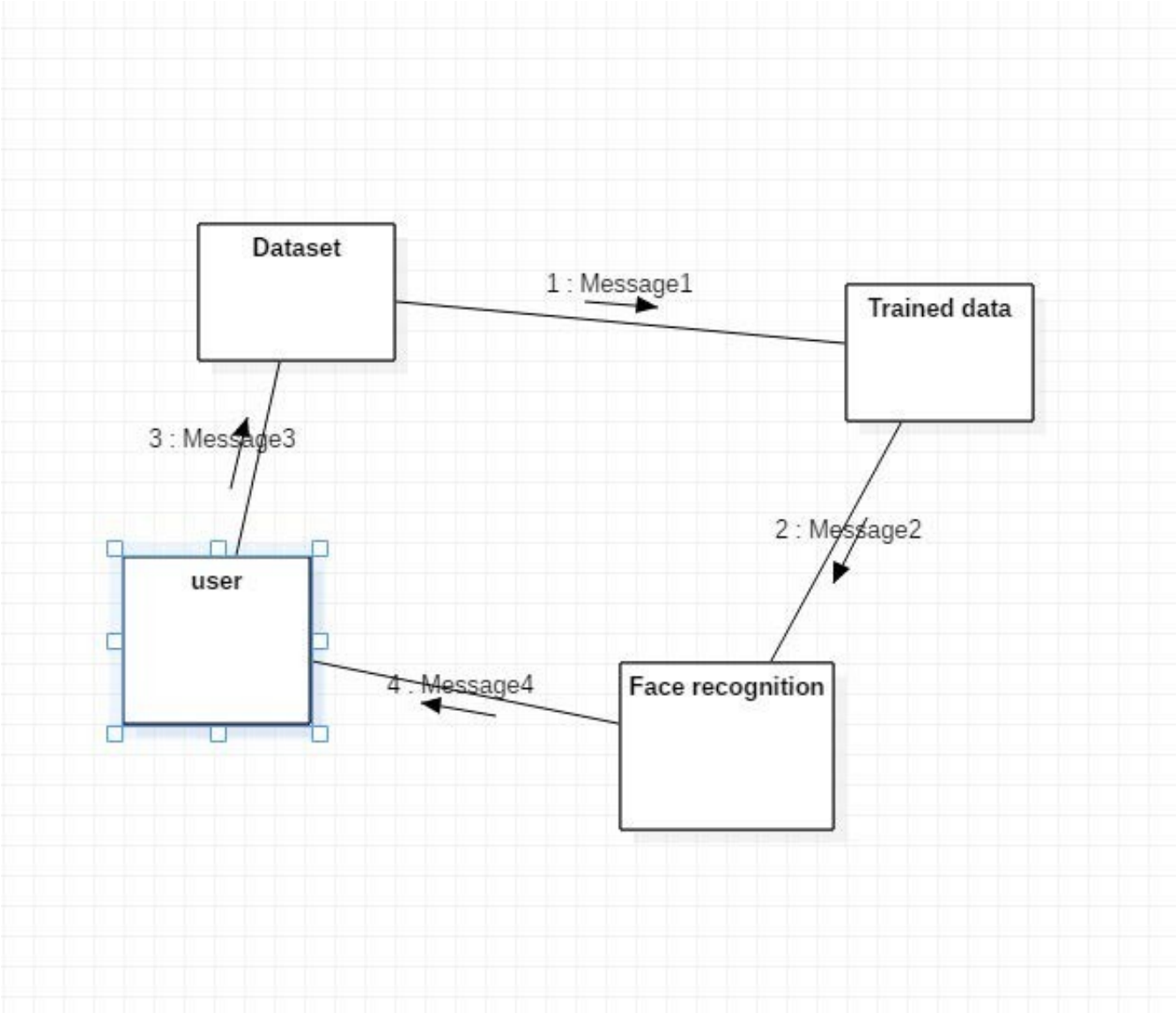


Fig no:7 Collaboration diagram

2.2 Statechart Diagram

Statechart diagram is one of the five UML diagrams used to model the dynamic nature of a system. They define different states of an object during its lifetime and these states are changed by events. Statechart diagrams are useful to model the reactive systems.

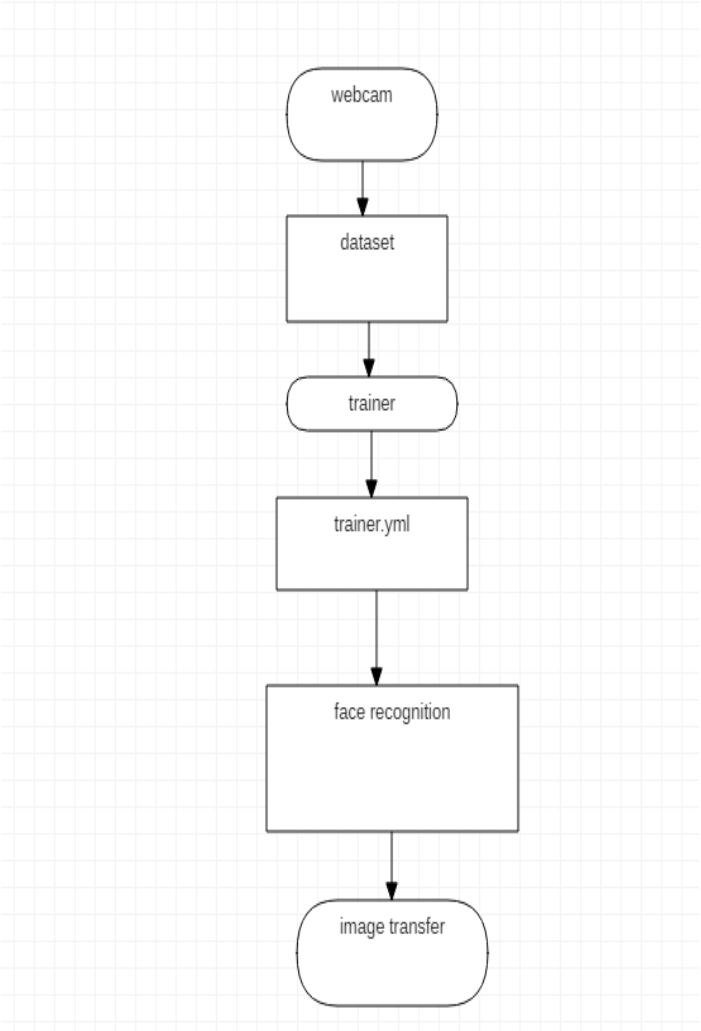


Fig no:8 Statechart diagram

2.3 Flowchart Diagram

A flowchart is a type of diagram that represents an algorithm, workflow or process. The flowchart shows the steps as boxes of various kinds, and their order by connecting the boxes with arrows. This diagrammatic representation illustrates a solution model to a given problem.

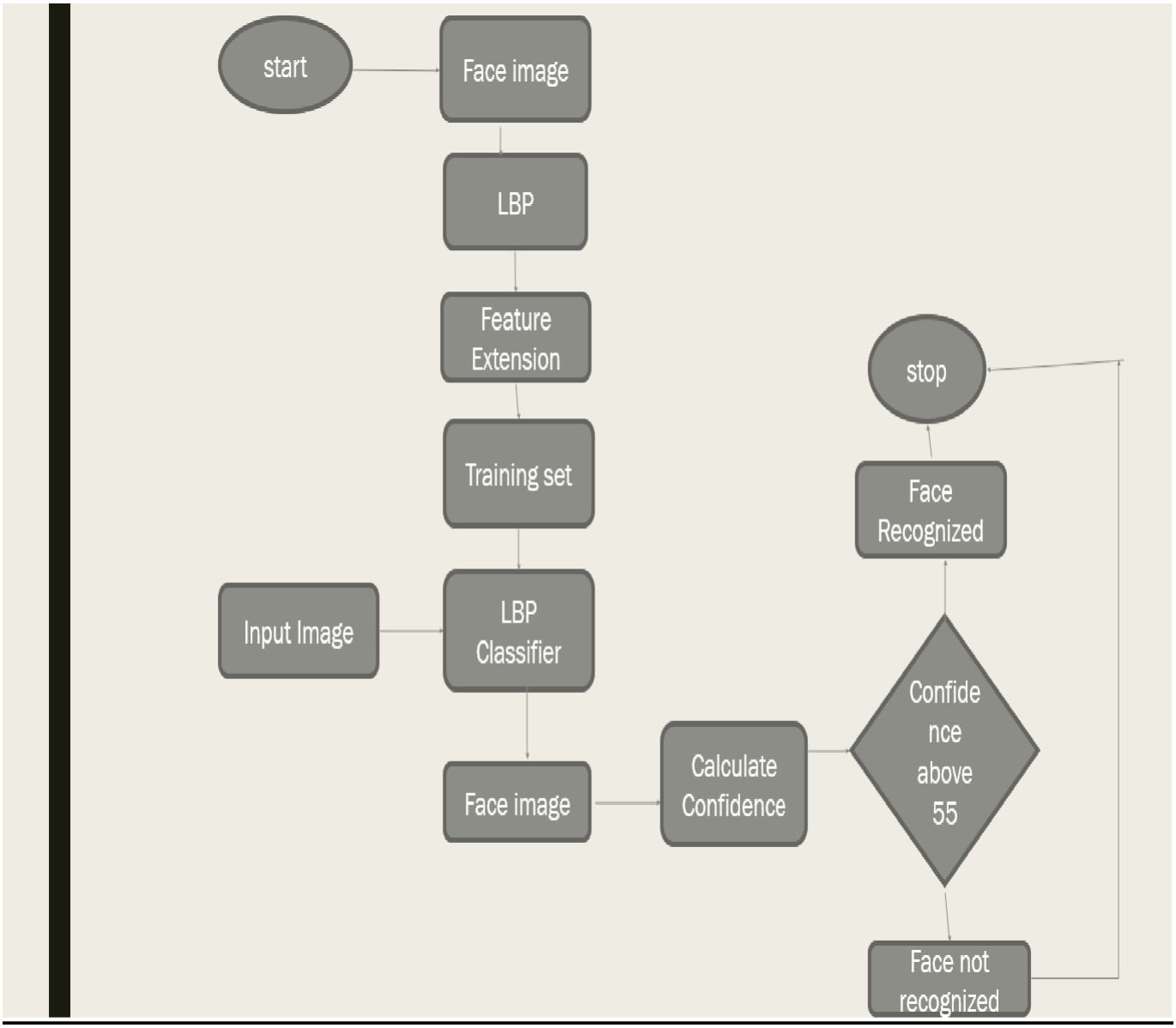


Fig no:9 Flowchart diagram

Chapter – 3 : Implementations

Code 0 : tkinter(UI of the application)

```
from tkinter import *
from playsound import playsound
import os
from datetime import datetime;
#creating instance of TK
root=Tk()

root.configure(background="white")

#root.geometry("300x300")

def function1():

    os.system("py dataset_capture.py")

def function2():

    os.system("py training_dataset.py")

def function3():

    os.system("py recognizer.py")
    playsound('sound.mp3')

def function5():
   os.startfile(os.getcwd()+"/developers/diet1frame1first.html");

def function6():

    root.destroy()

def attend():
    os.startfile(os.getcwd()+"/firebase/attendance_files/attendance"+str(datetime.now().date())
+'.xls')

#stting title for the window
root.title("AUTOMATIC ATTENDANCE MANAGEMENT USING FACE RECOGNITION")

#creating a text label
Label(root, text="FACE RECOGNITION ATTENDANCE SYSTEM",font=("times new
roman",20),fg="white",bg="maroon",height=2).grid(row=0,rowspan=2,columnspan=2,sticky=N
+E+W+S,padx=5,pady=5)

#creating first button
Button(root,text="Create Dataset",font=("times new
roman",20),bg="#0D47A1",fg='white',command=function1).grid(row=3,columnspan=2,sticky=
```

```
W+E+N+S,padx=5,pady=5)

#creating second button
Button(root,text="Train Dataset",font=("times new
roman",20),bg="#0D47A1",fg='white',command=function2).grid(row=4,columnspan=2,sticky=
N+E+W+S,padx=5,pady=5)

#creating third button
Button(root,text="Recognize + Attendance",font=('times new
roman',20),bg="#0D47A1",fg="white",command=function3).grid(row=5,columnspan=2,sticky=
N+E+W+S,padx=5,pady=5)

#creating attendance button
Button(root,text="Attendance Sheet",font=('times new
roman',20),bg="#0D47A1",fg="white",command=attend).grid(row=6,columnspan=2,sticky=N+
E+W+S,padx=5,pady=5)

Button(root,text="Developers",font=('times new
roman',20),bg="#0D47A1",fg="white",command=function5).grid(row=8,columnspan=2,sticky=
N+E+W+S,padx=5,pady=5)

Button(root,text="Exit",font=('times new
roman',20),bg="maroon",fg="white",command=function6).grid(row=9,columnspan=2,sticky=N
+E+W+S,padx=5,pady=5) hapter – 3:          Implementations
```

Code 1 : Dataset_capture(it will capture your face at beginning)

```
# Import OpenCV2 for image processing
import cv2
import os
def assure_path_exists(path):
    dir = os.path.dirname(path)
    if not os.path.exists(dir):
        os.makedirs(dir)
face_id=input('enter your id')
# Start capturing video
vid_cam = cv2.VideoCapture(0)
# Detect object in video stream using Haarcascade Frontal Face
face_detector = cv2.CascadeClassifier('haarcascade_frontalface_default.xml')
# Initialize sample face image
count = 0
assure_path_exists("dataset/")
# Start looping
while(True):
  # Capture video frame
   _, image_frame = vid_cam.read()
  # Convert frame to grayscale
   gray = cv2.cvtColor(image_frame, cv2.COLOR_BGR2GRAY)
  # Detect frames of different sizes, list of faces rectangles
   faces = face_detector.detectMultiScale(gray, 1.3, 5)
  # Loops for each faces
   for (x,y,w,h) in faces:
      # Crop the image frame into rectangle
       cv2.rectangle(image_frame, (x,y), (x+w,y+h), (255,0,0), 2)

       # Increment sample face image
```

```
        count += 1
        # Save the captured image into the datasets folder
        cv2.imwrite("dataset/User." + str(face_id) + '.' + str(count) + ".jpg", gray[y:y+h,x:x+w])
        # Display the video frame, with bounded rectangle on the person's face
        cv2.imshow('frame', image_frame)
    # To stop taking video, press 'q' for at least 100ms
    if cv2.waitKey(100) & 0xFF == ord('q'):
        break
    # If image taken reach 100, stop taking video
    elif count>=30:
        print("Successfully Captured")
        break
# Stop video
vid_cam.release()
# Close all started windows
cv2.destroyAllWindows()
```

Code 2 : Recognizer(it will recognize your face)

```
import cv2, numpy as np;
import xlwrite,firebase.firebase_ini as fire;
import time
import sys
from playsound import playsound
start=time.time()
period=8
face_cas = cv2.CascadeClassifier('haarcascade_profileface.xml')
cap = cv2.VideoCapture(0);
recognizer = cv2.face.LBPHFaceRecognizer_create();
recognizer.read('trainer/trainer.yml');
flag = 0;
id=0;
filename='filename';
dict = {
        'item1': 1
}
#font = cv2.InitFont(cv2.cv.CV_FONT_HERSHEY_SIMPLEX, 5, 1, 0, 1, 1)
font = cv2.FONT_HERSHEY_SIMPLEX
while True:
    ret, img = cap.read();
    gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY);
    faces = face_cas.detectMultiScale(gray, 1.3, 7);
    for (x,y,w,h) in faces:
        roi_gray = gray[y:y + h, x:x + w]
        cv2.rectangle(img, (x,y), (x+w, y+h), (255,0,0),2);
        id,conf=recognizer.predict(roi_gray)
        if(conf < 50):
         if(id==1):
```

```
    id='Shubam Walia'
    if((str(id)) not in dict):
        filename=xlwrite.output('attendance','class1',1,id,'yes');
        dict[str(id)]=str(id);

 elif(id==2):
    id = 'Rohan'
    if ((str(id)) not in dict):
        filename =xlwrite.output('attendance', 'class1', 2, id, 'yes');
        dict[str(id)] = str(id);

 elif(id==3):
    id = 'Raveen'
    if ((str(id)) not in dict):
        filename =xlwrite.output('attendance', 'class1', 3, id, 'yes');
        dict[str(id)] = str(id);

 elif(id==4):
    id = 'Sonu'
    if ((str(id)) not in dict):
        filename =xlwrite.output('attendance', 'class1', 4, id, 'yes');
        dict[str(id)] = str(id);

else:
     id = 'Unknown, can not recognize'
     flag=flag+1
     break

cv2.putText(img,str(id)+" "+str(conf),(x,y-10),font,0.55,(120,255,120),1)
#cv2.cv.PutText(cv2.cv.fromarray(img),str(id),(x,y+h),font,(0,0,255));
```

```
    cv2.imshow('frame',img);
    #cv2.imshow('gray',gray);
    if flag == 10:
        playsound('transactionSound.mp3')
        print("Transaction Blocked")
        break;
    if time.time()>start+period:
        break;
    if cv2.waitKey(100) & 0xFF == ord('q'):
        break;

cap.release();
cv2.destroyAllWindows();
```

Code 3 : Reports

```
#import module from tkinter for UI
from tkinter import *
import os
#creating instance of TK
root=Tk()
root.configure(background="white")
#root.geometry("600x600")
def function1():
        os.system("synopsis.pdf")
       def function2():
      os.system("report1.pdf")
      def function3():
os.system("report2.pdf")
def function4():
   root.destroy()
    os.system("py firstpage.py")
#stting title for the window
root.title("AUTOMATIC ATTENDANCE MANAGEMENT USING FACE RECOGNITION")
#creating a text label
Label(root, text="Project Reports",font=("times new
roman",40),fg="white",bg="red",height=2).grid(row=0,rowspan=2,columnspan=2,sticky=N+E+
W+S,padx=5,pady=5)
#creating a button
Button(root,text="Synopsis",font=("times new
roman",30),bg="#3F51B5",fg='white',command=function1).grid(row=3,columnspan=2,sticky=
W+E+N+S,padx=5,pady=5)

#creating second button
Button(root,text="Report 1",font=("times new
roman",30),bg="#3F51B5",fg='white',command=function2).grid(row=4,columnspan=2,sticky=
```

```
N+E+W+S,padx=5,pady=5)

#creating third button

Button(root,text="Report 2",font=('times new
roman',30),bg="#3F51B5",fg="white",command=function3).grid(row=5,columnspan=2,sticky=
N+E+W+S,padx=5,pady=5)

#creating five button

Button(root,text="Back",font=('times new
roman',40),bg="red",fg="white",command=function4).grid(row=8,columnspan=2,sticky=N+E+
W+S,padx=5,pady=5)

root.mainloop()
```

Code 4 : xlwrite(to store the attendance in xlsheet)

```
import os,cv2;
import numpy as np
from PIL import Image;

recognizer = cv2.face.LBPHFaceRecognizer_create()
detector= cv2.CascadeClassifier("haarcascade_frontalface_default.xml");

def getImagesAndLabels(path):
    #get the path of all the files in the folder
    imagePaths=[os.path.join(path,f) for f in os.listdir(path)]
    #create empth face list
    faceSamples=[]
    #create empty ID list
    Ids=[]
    #now looping through all the image paths and loading the Ids and the images
    for imagePath in imagePaths:
        #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=detector.detectMultiScale(imageNp)
        #If a face is there then append that in the list as well as Id of it
        for (x,y,w,h) in faces:
            faceSamples.append(imageNp[y:y+h,x:x+w])
            Ids.append(Id)
    return faceSamples,Ids
```

```
faces,Ids = getImagesAndLabels('dataSet')
s = recognizer.train(faces, np.array(Ids))
print("Successfully trained")
recognizer.write('trainer/trainer.yml')
```

Code 5: Train the dataset

```
import xlwt;
from datetime import datetime;
from xlrd import open_workbook;
from xlwt import Workbook;
from xlutils.copy import copy
from pathlib import Path

'''style0 = xlwt.easyxf('font: name Times New Roman, color-index red, bold on',
    num_format_str='#,##0.00')
style1 = xlwt.easyxf(num_format_str='D-MMM-YY')

wb = xlwt.Workbook()
ws = wb.add_sheet('A Test Sheet')

ws.write(0, 0, 1234.56, style0)
ws.write(1, 0, datetime.now(), style1)
ws.write(2, 0, 1)
ws.write(2, 1, 1)
ws.write(2, 2, xlwt.Formula("A3+B3"))

wb.save('example.xls')
'''
def output(filename, sheet,num, name, present):
    my_file = Path('firebase/attendance_files/'+filename+str(datetime.now().date())+'.xls');
    if my_file.is_file():
        rb = open_workbook('firebase/attendance_files/'+filename+str(datetime.now().date())
+'.xls');
        book = copy(rb);
        sh = book.get_sheet(0)
```

```
    # file exists
else:
    book = xlwt.Workbook()
    sh = book.add_sheet(sheet)
style0 = xlwt.easyxf('font: name Times New Roman, color-index red, bold on',
                     num_format_str='#,##0.00')
style1 = xlwt.easyxf(num_format_str='D-MMM-YY')

#variables = [x, y, z]
#x_desc = 'Display'
#y_desc = 'Dominance'
#z_desc = 'Test'
#desc = [x_desc, y_desc, z_desc]
sh.write(0,0,datetime.now().date(),style1);



col1_name = 'Name'
col2_name = 'Present'



sh.write(1,0,col1_name,style0);
sh.write(1, 1, col2_name,style0);

sh.write(num+1,0,name);
sh.write(num+1, 1, present);
#You may need to group the variables together
#for n, (v_desc, v) in enumerate(zip(desc, variables)):
fullname=filename+str(datetime.now().date())+'.xls';
book.save('firebase/attendance_files/'+fullname)
return fullname;
```

Chapter – 4: Testing (include Test cases)

SYSTEM TESTING

Software testing is a critical element of software quality assurance and represents the ultimate review of specification, design and code generation.

Testing Objectives

• To ensure that during operation the system will perform as per specification.

• To make sure that system meets the user requirements during operation.

• To make sure that during the operation, incorrect input, processing and output will be detected.

• To see that when correct inputs are fed to the system the outputs are correct.

• To verify that the controls incorporated in the same system as intended.

• Testing is a process of executing a program with the intent of finding an error.

• A good test case is one that has a high probability of finding an as yet undiscovered error.

The software developed has been tested successfully using the following testing strategies and any errors that are encountered are corrected and again the part of the program or the procedure or function is put to testing until all the errors are removed. A successful test is one that uncovers an as yet undiscovered error.

Note that the result of the system testing will prove that the system is working correctly. It will give confidence to system designer, users of the system, prevent frustration during implementation process etc.,

Manual testing:

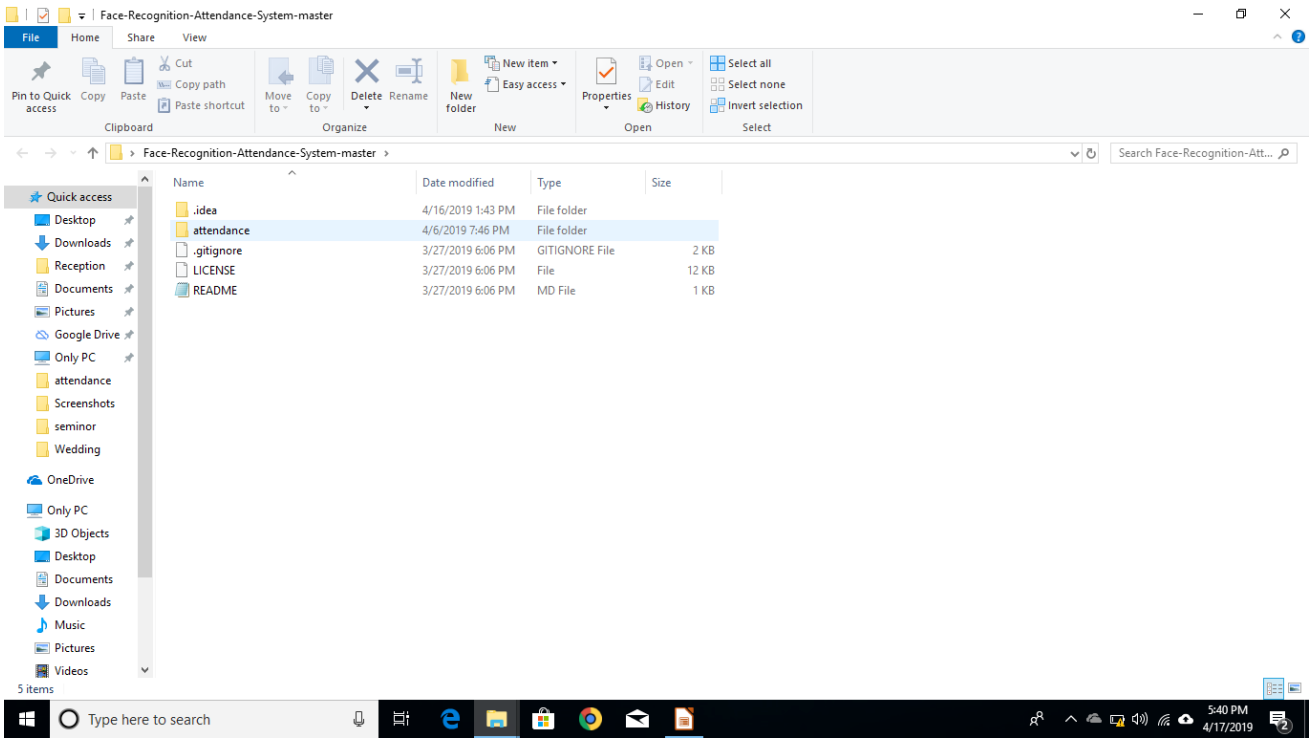
Manual Testing is a type of Software Testing where Testers manually execute test cases without using any automation tools.

Manual Testing is the most primitive of all testing types and helps find bugs in the software system.

Any new application must be manually tested before its testing can be automated. Manual Testing requires more effort, but is necessary to check automation feasibility.

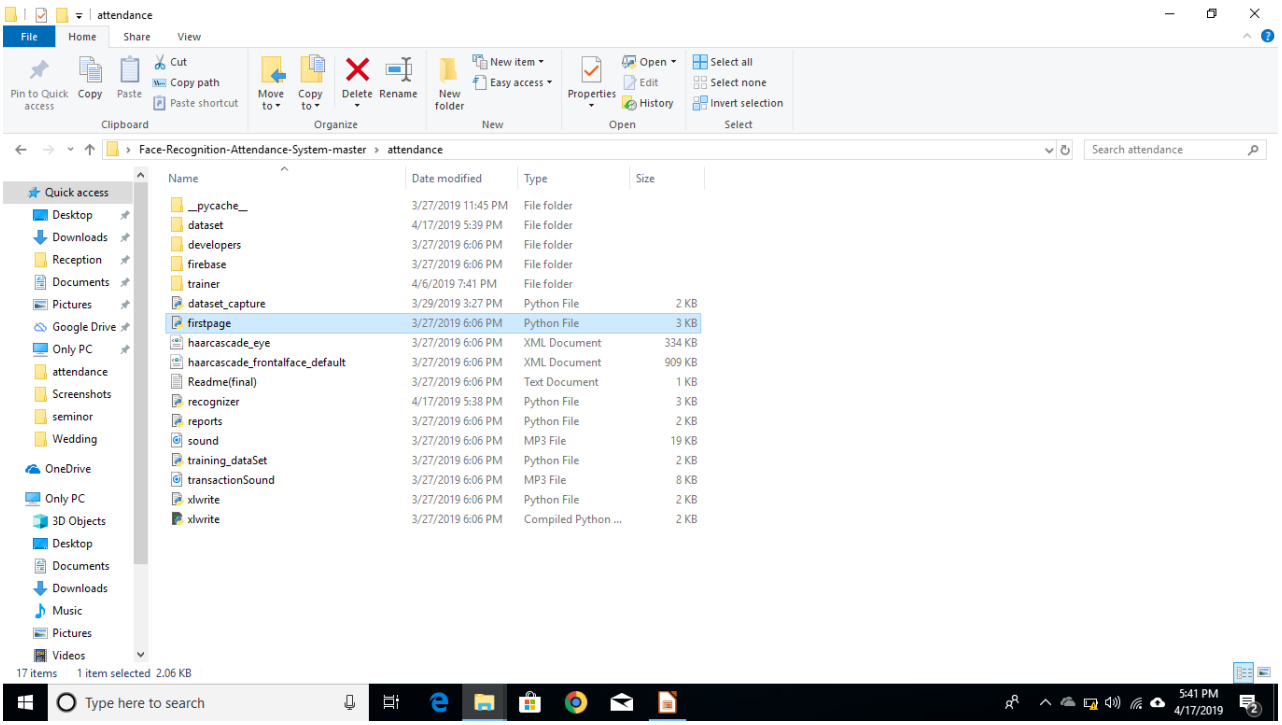
Chapter – 5: Results (include Screen shots)

Screenshots-1 : Attendance folder



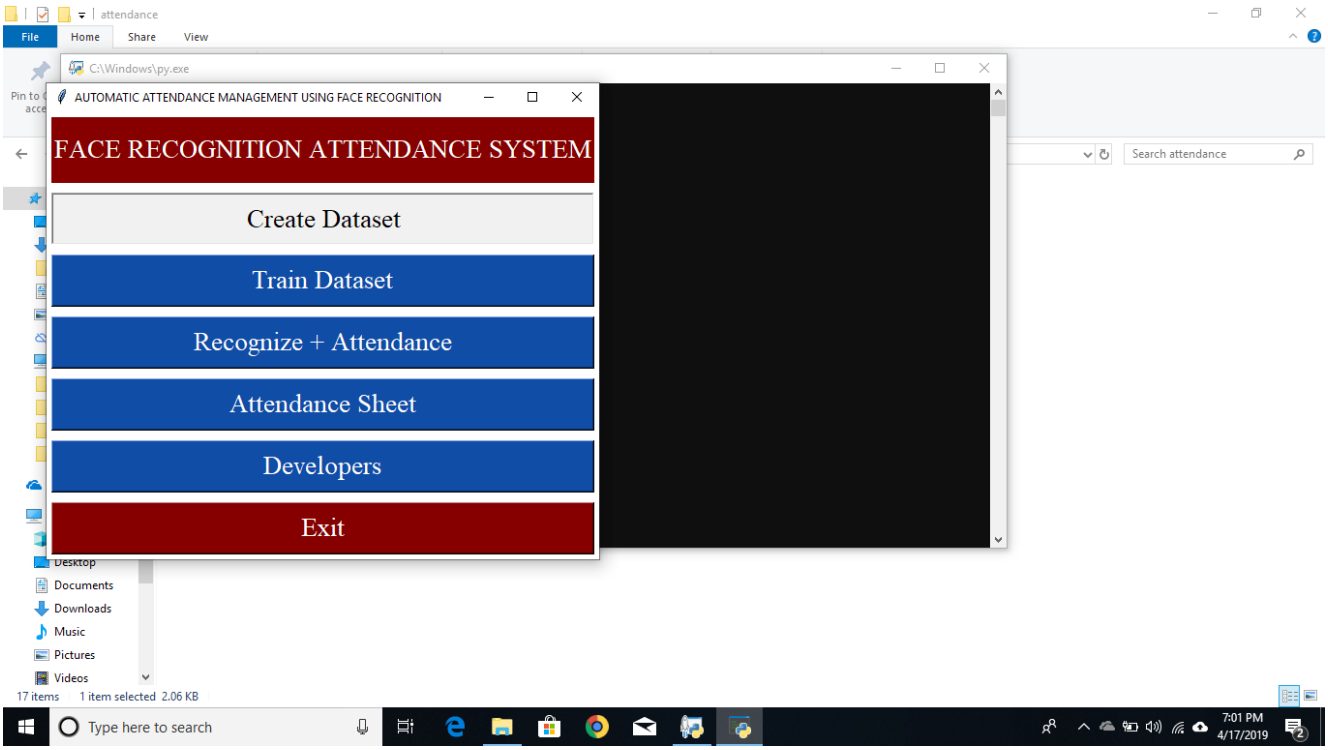
Step-1

Screenshot-2 : first page (It will open the UI of the application)

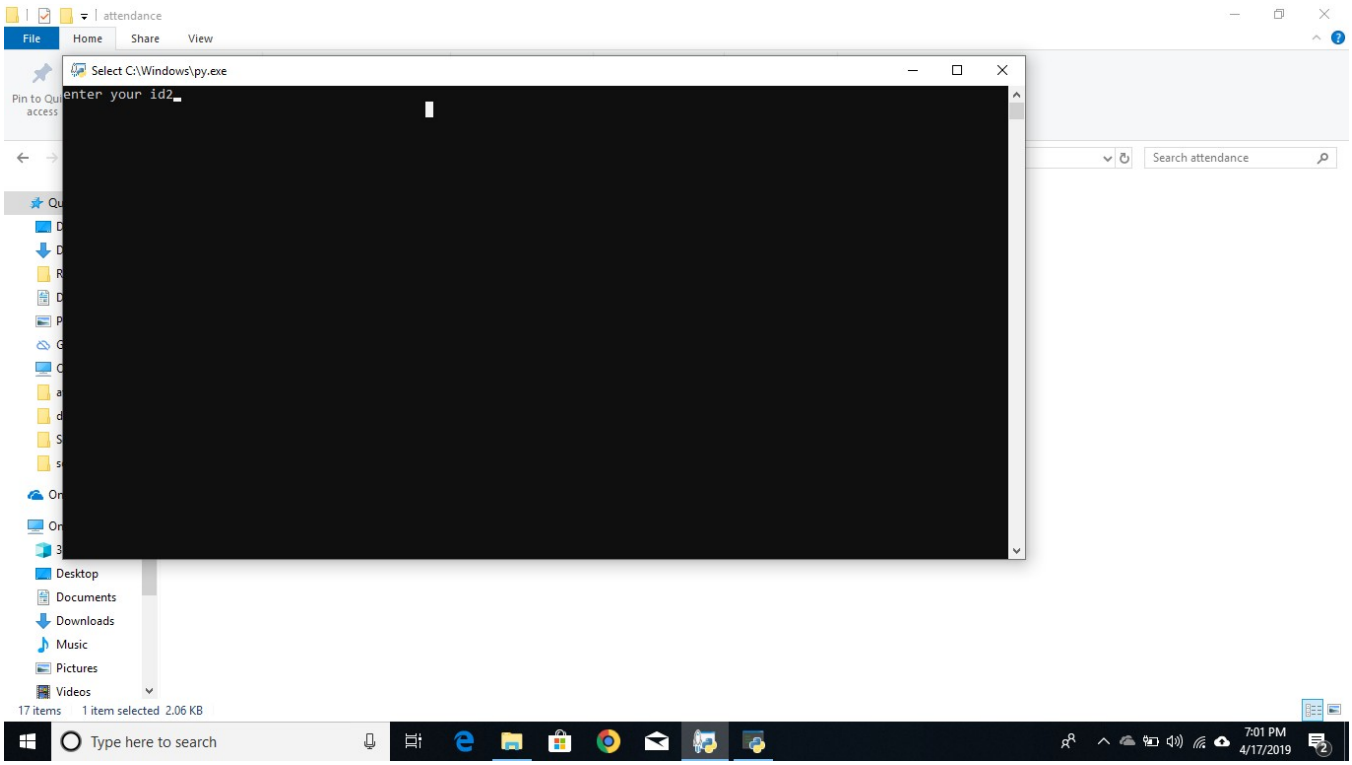


Step-2

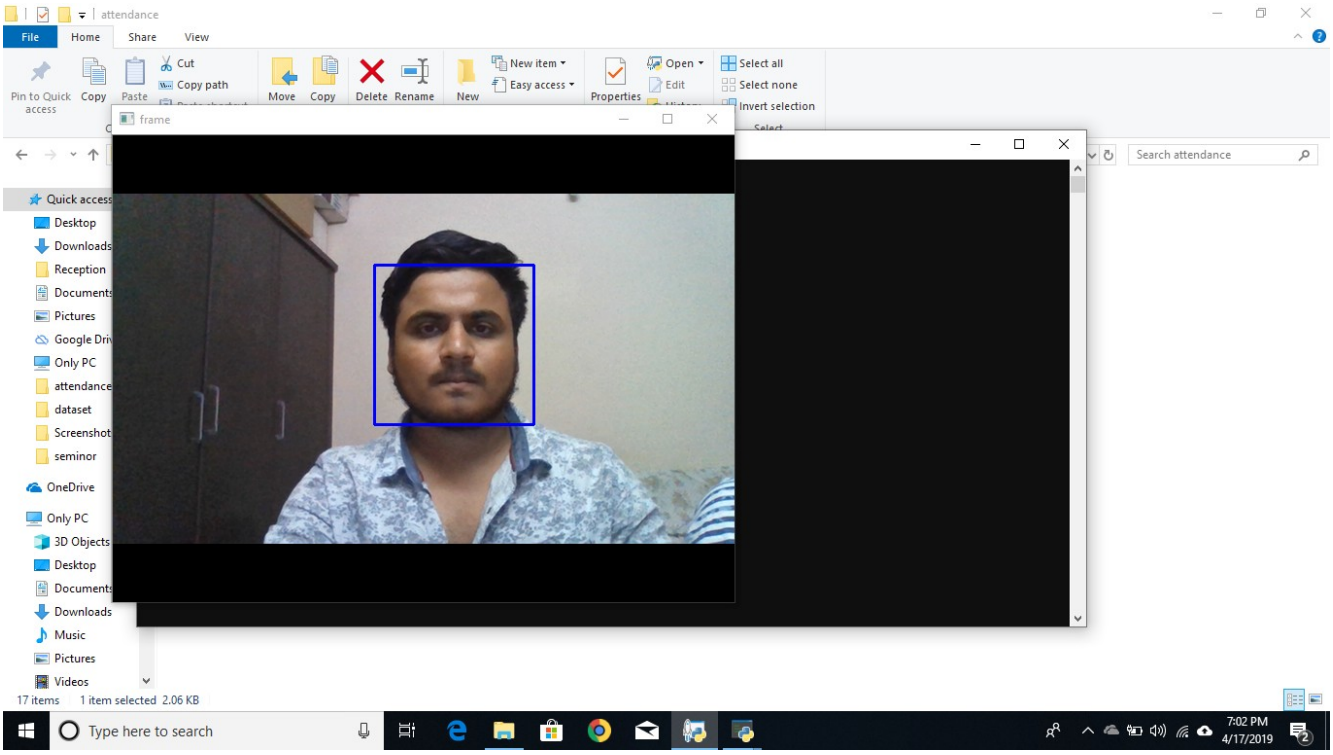
Screenshots-3 : UI(to create dataset)



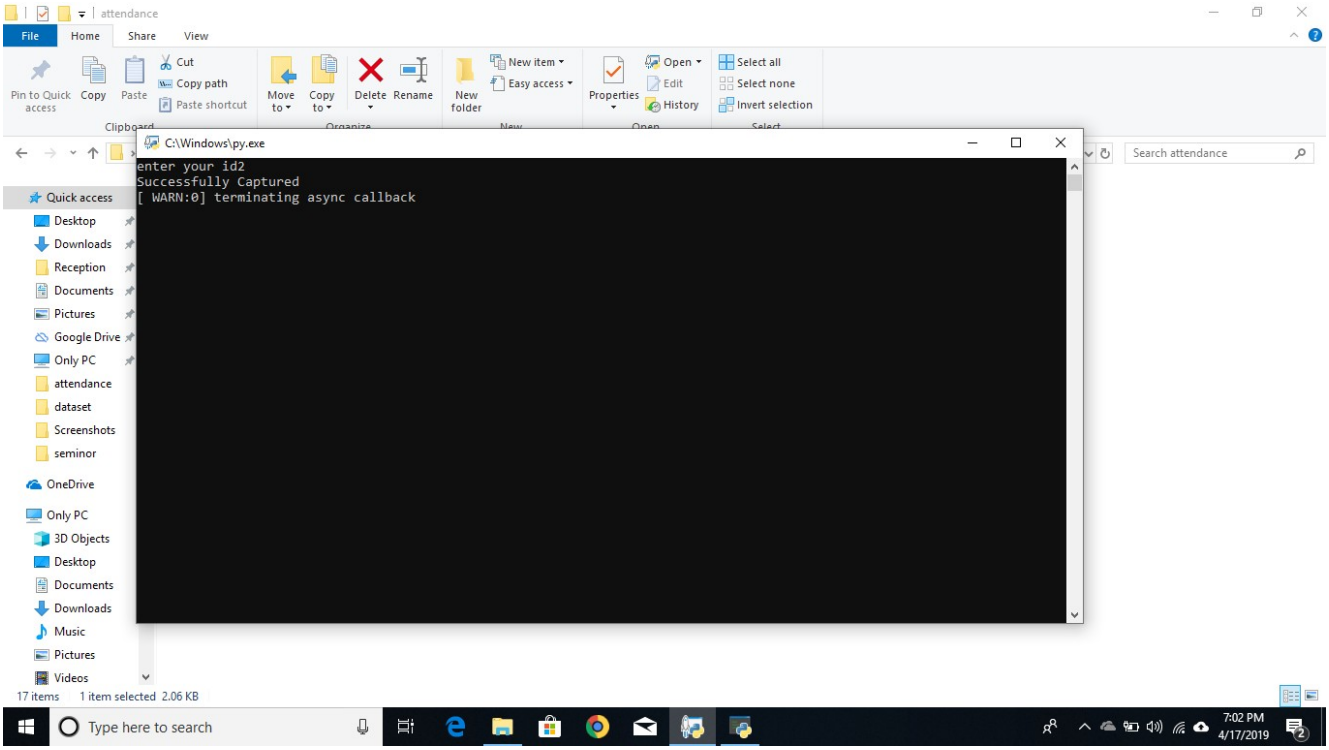
Screenshots -4 : Enter the id



Screenshots -5 : Dataset of face will be stored in database

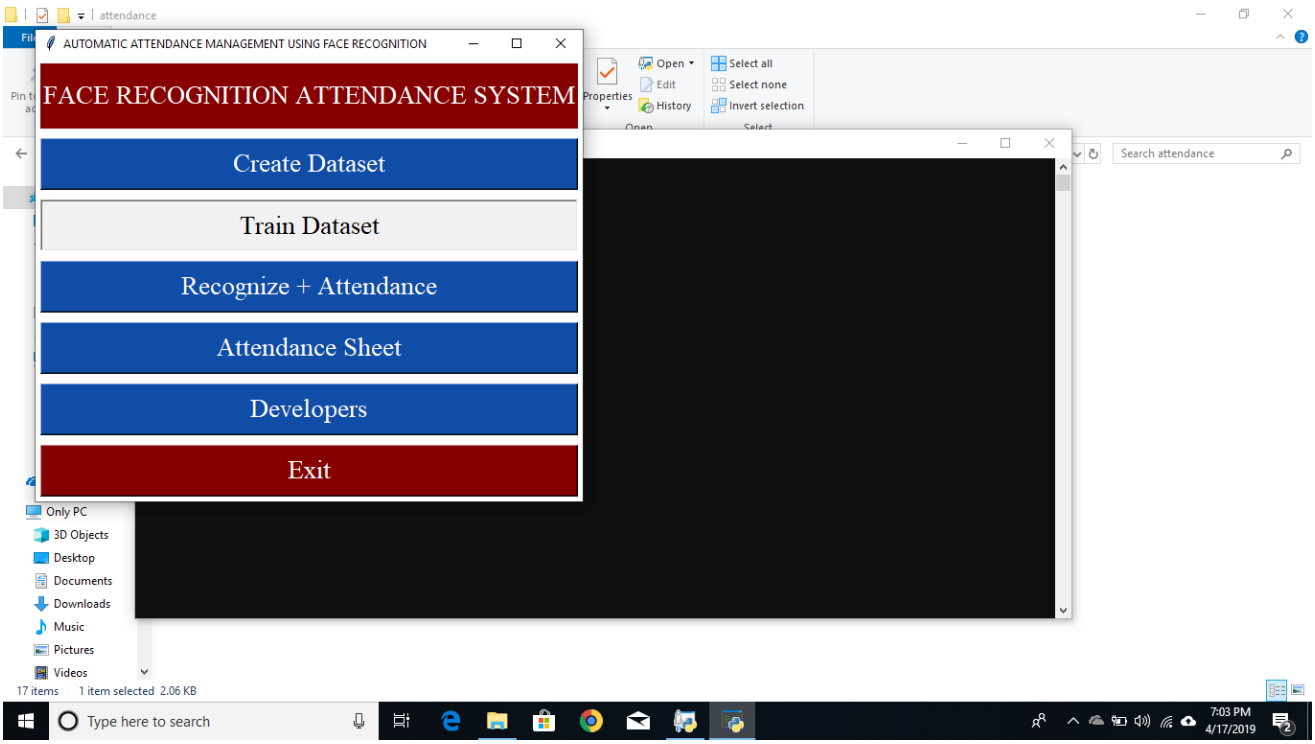


Screenshots -6 : successfully captured the image

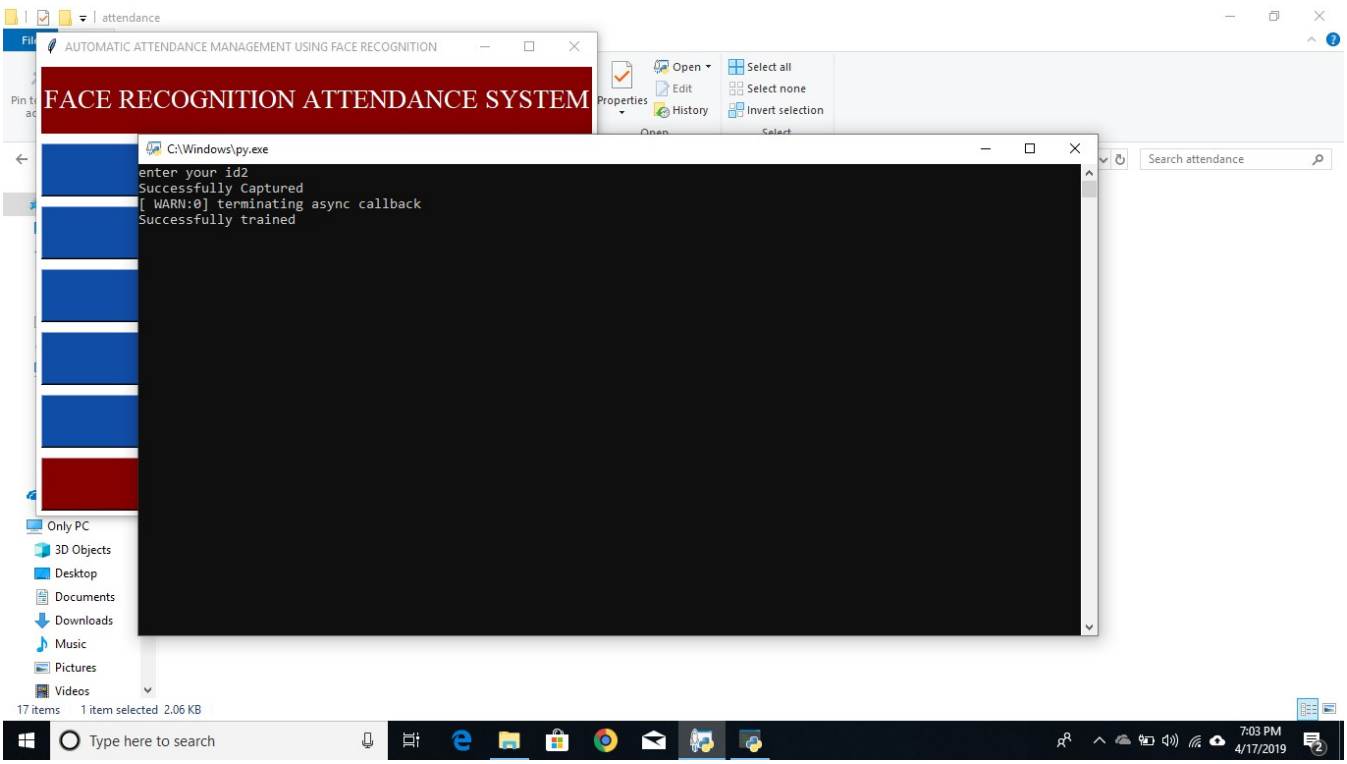


Step-3

Screenshots-7 : Train the dataset

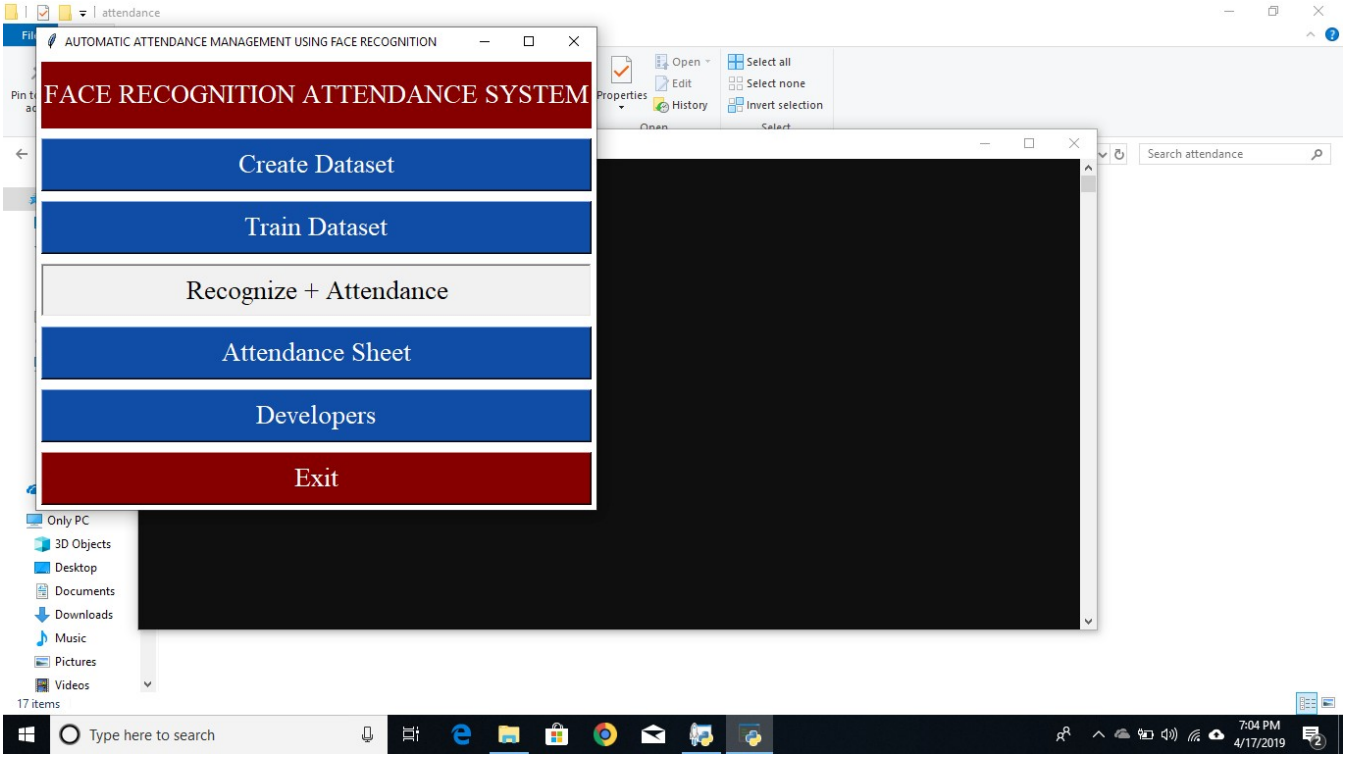


Screenshots -8 : successfully trained the dataset

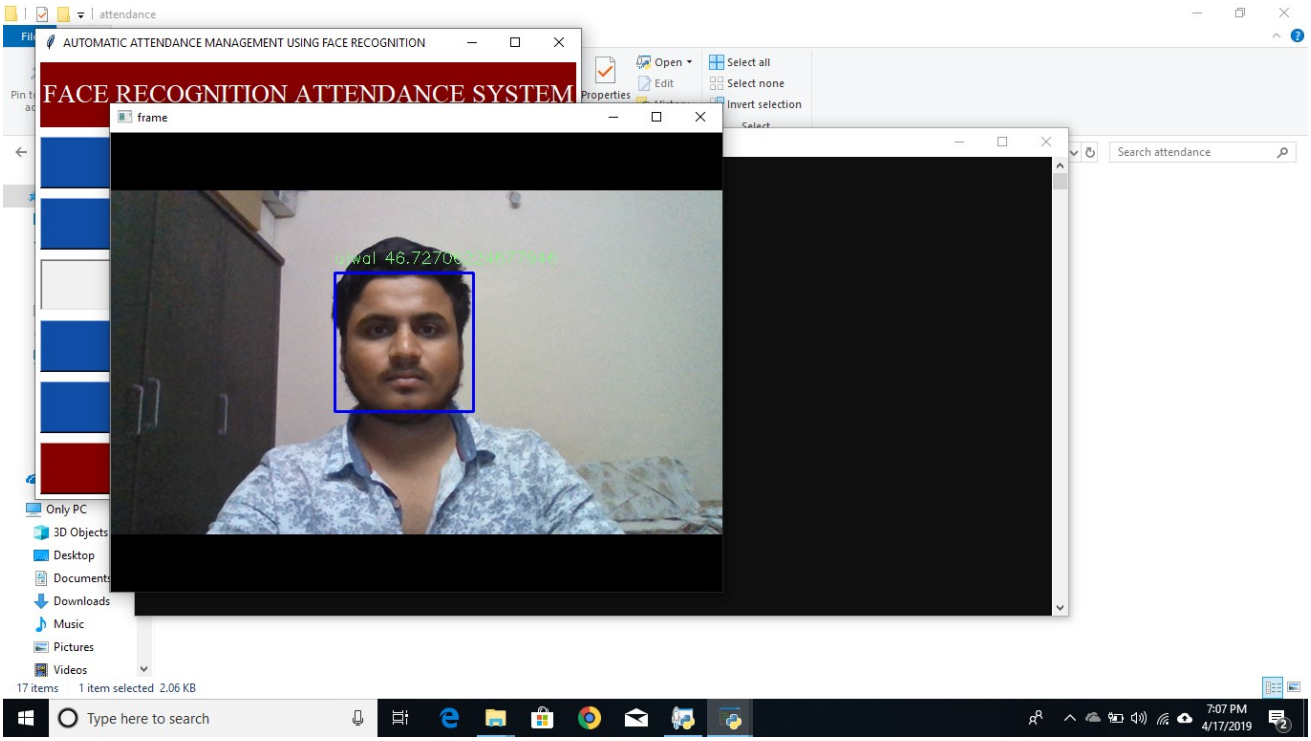


Step -4

Screenshots -9 : recognize and give the Attendance

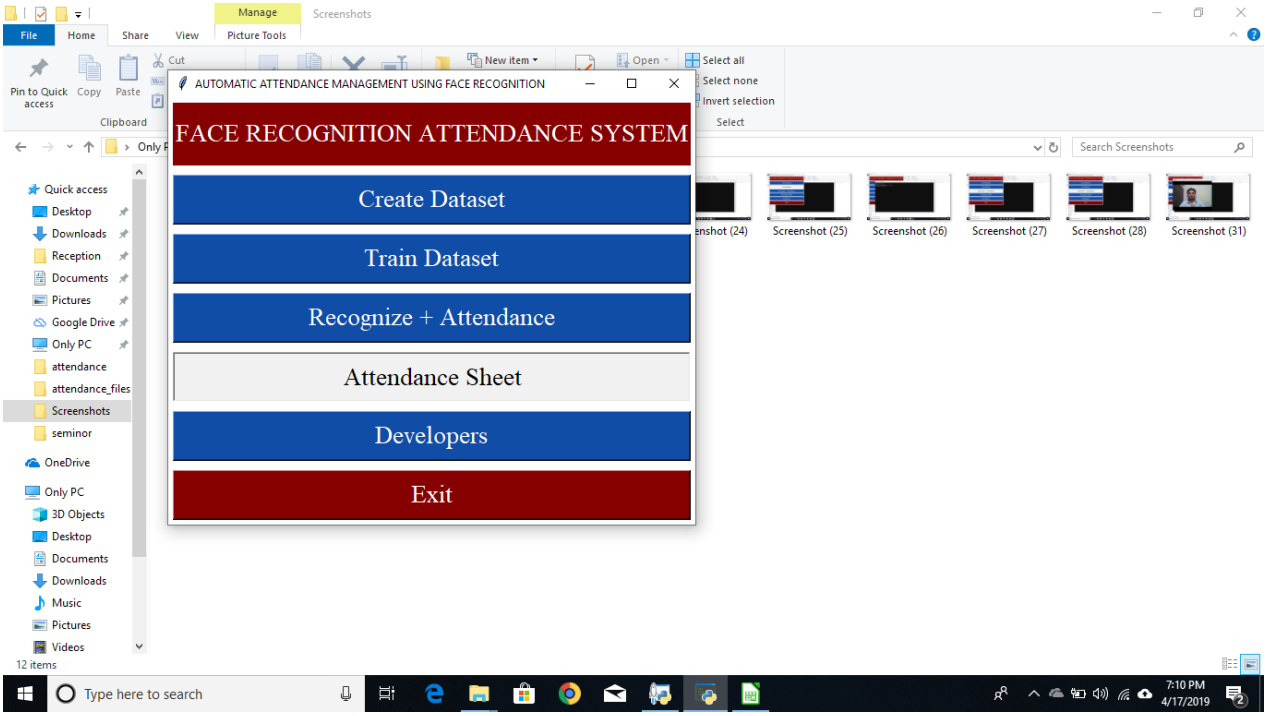


Screenshots -10 : successfully recognising

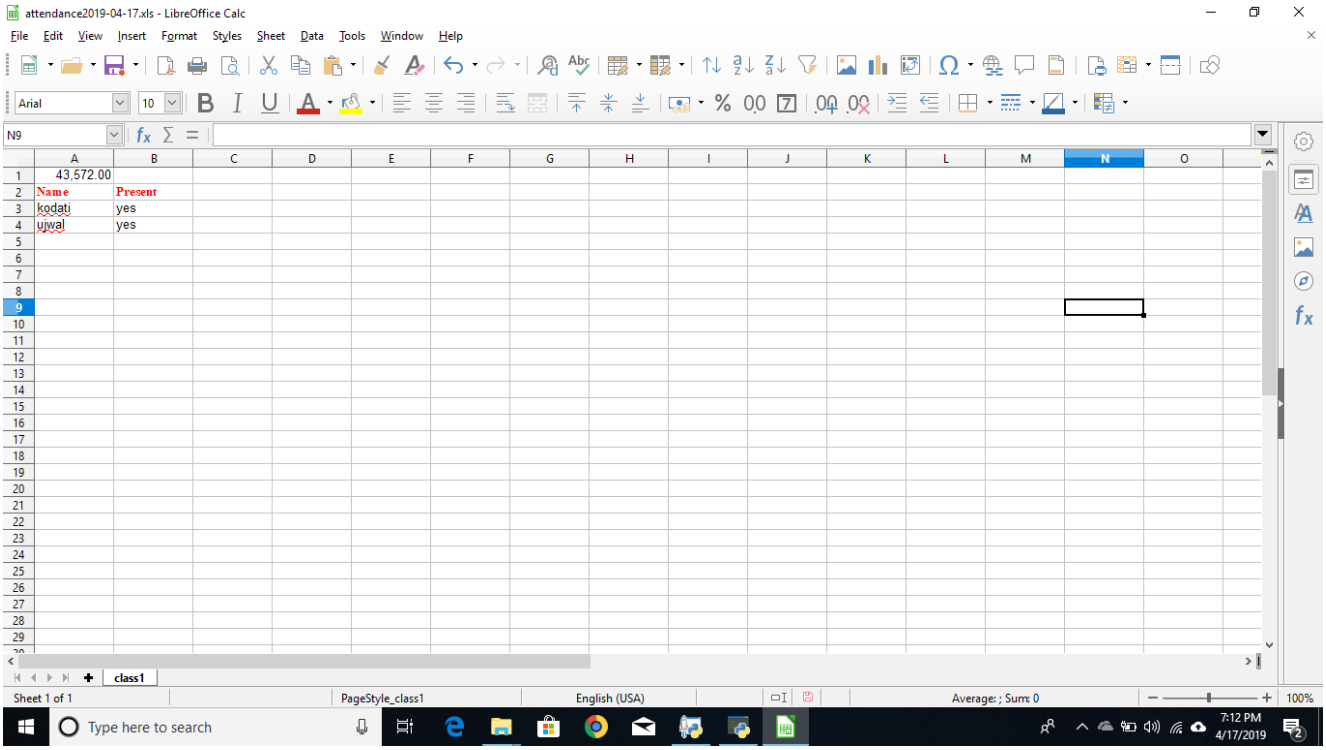


Step-5

Screenshots-11 : Attendance sheet



Screenshots -12 : present members



Chapter –6: Conclusions and Future Scope

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. T he 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**.**

There may be various types of lighting conditions, seating arrangements and environments in various classrooms. Most of these conditions have been tested on the system and system has shown 100% accuracy for most of the cases. There may also exist students portraying various facial expressions, varying hair styles, beard, spectacles etc. All of these cases are considered and tested to obtain a high level of accuracy and efficiency. Thus, it can be concluded from the above discussion that a reliable, secure, fast and an efficient system has been developed replacing a manual and unreliable system. This system can be implemented for better results regarding the management of attendance and leaves. The system will save time, reduce the amount of work the administration has to do and will replace the stationery material with electronic apparatus and reduces the amount of human resource required for the purpose. Hence a system with expected results has been developed but there is still some room for improvement

SCOPE FOR FUTURE WORK:

1. Currently, the system has reached the accuracy level up to 80% for partial and dense images. It can further be improved to obtain higher accuracy levels.
2. Further, 2 or more IP cameras can be employed and each image can be processed separately. The results of these can be merged to obtain better results and accuracy in denser classrooms.

Bibliography

[1]. W. Zhao, R. Chellappa, P. J. Phillips, and A. Rosenfeld,“Face recognition: A literature survey,” ACM Computing Surveys, 2003, vol. 35, no. 4, pp. 399-458.

[2]. Herbert Bay, Andreas Ess, Tinne Tuytelaars, and Luc Van Gool. Surf: Speeded up robust features. Computer Vision and Image Understanding (CVIU), 110(3):346–359.

[3]. H.K.Ekenel and R.Stiefelhagen,Analysis of local appearance based face recognition: Effects of feature selection and feature normalization. In CVPR Biometrics Workshop, New York, USA, 2006

[4]. IJCSI International Journal of Computer Science Issues, Vol. 9, Issue 4, No 1, July 2012 ISSN (Online): 1694-0814

[5]. Javier Ruiz Del Solar, Rodrigo Verschae, and Mauricio Correa. Face recognition in unconstrained environments: A comparative study. In ECCV Workshop on Faces in Real-Life Images: Detection, Alignment, and Recognition, Marseille, France, October 2008.

[6].Kyungnam Kim “Face Recognition using Principle Component Analysis”,

Department of Computer Science, University of Maryland, College Park, MD 20742, USA.

[7]. Osuna, E., Freund, R. and Girosit, F. (1997). "Training support vector machines:

an application to face detection." 130-136