A

Project Report

on

SMART VOTING SYSTEM

Submitted for partial fulfillment of the requirements for the award of the degree of

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE AND ENGINEERING

By

Kolkundha Preethi (2451-17-733-012)

Theddu Rithik (2451-17-733-029)

Linga Akhil (2451-17-733-302)

Under the guidance of

Vikram Narayandas

Assistant Professor

Department of CSE

Maturi Venkata Subba Rao (MVSR) ENGINEERING COLLEGE

Department of Computer Science and Engineering (Affiliated to Osmania University & Recognized by AICTE) Nadergul, Saroor Nagar Mandal, Hyderabad – 501 510 Academic Year: 2020-21

CERTIFICATE

*This is to certify that the project work entitled "**Smart Voting System**" is a bonafide work carried out by **Ms. Kolkundha Preethi (2451-15-733-012), Mr.Theddu Rithik (2451-15-733-029), Mr.Linga Akhil (2451-15-733-302)** in partial fulfilment of the requirements for the award of degree of **Bachelor of Engineering** in **Computer Science and Engineering** from **Maturi Venkata Subba Rao (MVSR) Engineering College,** affiliated to OSMANIA UNIVERSITY, Hyderabad, during the Academic Year 2020-21 under our guidance and supervision.*

The results embodied in this report have not been submitted to any other university or institute for the award of any degree or diploma to the best of our knowledge and belief..

Internal Guide
Mr.Vikram Naryandas
Assistant Professor
Department of CSE
MVSREC.

Head of the Department
J.Prasanna Kumar
Professor & Head
Department of CSE
MVSREC.

DECLARATION

This is to certify that the work reported in the present project entitled “Smart Voting System” is a record of bonafide work done by us in the Department of Computer Science and Engineering, Maturi Venkata Subba Rao (MVSR) Engineering College, Osmania University during the Academic Year 2020-21. The reports are based on the project work done entirely by us and not copied from any other source. The results embodied in this project report have not been submitted to any other University or Institute for the award of any degree or diploma.

Kolkundha Preethi	**Theddu Rithik**	**Linga Akhil**
(2451-17-733-012)	(2451-17-733-029)	(2451-17-733-302)

ACKNOWLEDGEMENT

I take this opportunity to express my profound and sincere gratitude to all those who helped us in carrying out this project successfully.

At the very outset, I am thankful to our principal **Dr. G. Kanaka Durga** and **Prof J. Prasanna Kumar,** Professor and Head, Department of Computer Science and Engineering, MVSR Engineering College, Hyderabad for their consent to do the project work as a part of our B.E Degree (CSE). We thank them for their valuable suggestions and advice throughout our stay at the college, during our project work.

I would like to thank our Internal Project Guide **Mr.Vikram Narayandas,** Assistant Professor, Department of Computer Science and Engineering, MVSR Engineering College for his/her useful suggestions, guidance and encouragement.

We thank the teaching and non-teaching staff of CSE for extending their support.

Finally, we are thankful to our parents for their cooperation and support throughout all endeavors in our life.

Kolkundha Preethi (2451-17-733-012)
Theddu Rithik (2451-17-733-029)
Linga Akhil (2451-17-733-302)

VISION

- To impart technical education of the highest standards, producing competent and confident engineers with an ability to use computer science knowledge to solve societal problems.

MISSION

- To make learning process exciting, stimulating and interesting.
- To impart adequate fundamental knowledge and soft skills to students.
- To expose students to advanced computer technologies in order to excel in engineering practices by bringing out the creativity in students.
- To develop economically feasible and socially acceptable software.

PEOs:

PEO-1: Demonstrate technical competence to successfully execute industry related software projects as a team member, leader or entrepreneur to meet customer business objectives.

PEO-2: Engage in life-long learning process by pursuing professional certifications, higher education or research in the emerging areas of information processing and intelligent systems at a global level. **PEO-3:** Advance in their professional careers by understanding the impact of computing on society or environment to make technical contributions using a multidisciplinary and ethical approach.

PROGRAM OUTCOMES(POs)

At the end of the program the students (Engineering Graduates) will be able to:

1. **Engineering knowledge:** Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
2. **Problem analysis:** Identify, formulate, review research literature, and analyse complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.
3. **Design/development of solutions:** Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.
4. **Conduct investigations of complex problems:** Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.

5. **Modern tool usage:** Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modelling to complex engineering activities with an understanding of the limitations.
6. **The engineer and society:** Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.
7. **Environment and sustainability:** Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.
8. **Ethics:** Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.
9. **Individual and team work:** Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.
10. **Communication:** Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.
11. **Project management and finance:** Demonstrate knowledge and understanding of the engineering and management principle and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.
12. **Lifelong learning:** Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

PROGRAM SPECIFIC OUTCOMES (PSOs)

13. (PSO-1) Demonstrate competence to build effective solutions for computational real-world problems using software and hardware across multi-disciplinary domains.
14. (PSO-2) Adapt to current computing trends for meeting the industrial and societal needs through a holistic professional development leading to pioneering careers or entrepreneurship

COURSE OBJECTIVES AND OUTCOMES

Course Objectives:

- Understand the significance of survey in necessary domains for problem identification.
- Understand to map requirements into software specification.
- Learn to apply concepts of software engineering for design of the identified real world problem
- Improve the coding capabilities by implementing the various modules of project
- Comprehend the suitable documentation procedure for a technical project.
- Exhibit effective communication and presentation skills.
- Learn the process of planning the complete lifecycle of a project

Course Outcomes:

CO1: Review recent advancements to formulate a precise problem statement with applications towards society.

CO2: Identify system requirements, explore design alternatives to design software based solution within the scope of project.

CO3: Implement, test and build the solution using contemporary technologies and tools.

CO4: Exhibit effective communication to present ideas clearly and produce well-structured technical report.

CO5: Demonstrate qualities necessary to work in a team and execute project as per plan.

ABSTRACT

In this project a new authentication technique in voting system using facial recognition of the voter is used. In India, currently there are two types of voting system in practice. They are secret Ballet paper and Electronic Voting Machines (EVM), but both of the processes have some limitations or demerits. The current voting system is not safe and secure too. So, parties in power try to make use of this situation and poll votes in favor to them which may leads to many problems. That's why in this project we have proposed a system or way which is very effective or useful in voting. In our approach we have three level security in voting process.

- First level is the verification of election id number (EID).
- Second level is OTP verification
- Third level is face recognition or face matching.

The security level of our system is greatly improved by the new application method for each voter. The user authentication process of the system is improved by adding face recognition in application which will identify whether the particular user is authenticated or not. To implement this idea, we will build a python app which has its backend up and running with Voters pictures, it takes Voter ID as input and matches with the live picture using Local Binary Patterns(LBP) Machine Learning Algorithm, if it matches then the voter can poll his vote.

K.Preethi (2451-17-733-012)
T.Rithik (2451-17-733-029)
Linga Akhil (2451-17-733-302)

TABLE OF CONTENTS

LIST OF FIGURES

1. INTRODUCTION

In this project a new authentication technique in voting system using facial recognition of the voter is used. In India, currently there are two types of voting system in practice. They are secret Ballet paper and Electronic Voting Machines (EVM), but both of the process have some limitation or demerits. The current voting system is not safe and secure too.

The voter who is not eligible can also cast his/her vote by fake means which may leads to many problems. That's why in this project we have to propose a system or way for voting which is very effective or useful in voting.

- First level is the verification of election id number (EID).
- Second level is OTP verification
- Third level is face recognition or face matching,

third level of security which is the main security level where the system recognizes the face of the real voter from the current database of face images given by the election commission. If the captured image is matched with the respective image of the voter in the backend, then a voter can cast their vote in the election.as you have to know that in existing system is not much more secure because in existing system security level is only voter card so any one can give other person vote with voter card but here, we proposed a way for voting which is more secure than existing system.

1.1 PROBLEM STATEMENT

- The vote is precious. It is the most powerful non-violent tool we have in a democratic society, and we must use it.
- Booth capturing, malfunctioning, Vote manipulation, tampering are major drawbacks of traditional voting systems.
- These wanted or unwanted errors could change the result of voting and there by change the fate of the country

1.2 OBJECTIVES

- Election department is getting updated every time but, malpractices are still taking place. So, using face recognition would be more permissible.
- Using face recognition system to get results that are precise and fast
- Low chances of malfunctioning, tampering and no chance of manipulation

1.3 MOTIVATION

- This Project involves application of all the mathematics combined with programming abilities that we have learnt at the University.
- This Project is an application of voting and has a real value.
- Working on data set to find the best architecture to identify people using their faces.
- Extracting Useful features from data which vary from person to person.

1.4 SCOPE

- Highly secured because in this project we have to use face recognition and face comparison. So, false user can't give votes.
- We can access result (counting) faster than existing system. Because ballet system takes much more time for counting process.
- Online voting system increase voting percentage in India. Because lots of people don't give vote. they think that the voting process is too lengthy but, in our approach, anyone can give vote easily.

SOFTWARE AND HARDWARE REQUIREMENTS

1.5.1. Software Requirements

- Visual code studio
- Python
- GIT
- Libraries (tkinter, random, re, numpy, datetime, csv, cv2.)

1.5.2.Hardware Requirements

- A custom PC with 4GB RAM and web cam

2. LITERATURE SURVEY

2.1 Smart Voting System (Existing Work)

2.1.1 Location-free Voting System with IOT:

This concept uses fingerprint sensor of the voter's phone to cast his vote. Mobile application to be installed in voter's phone which identifies the fingerprint. Smart phone's fingerprint sensor is linked with the provided application to identify the uniqueness of the original voter. Security provided is medium. This proposed system allows the voter to cast their votes only on the day of Election. The voters are not be able to cast the vote other than the day of Election. On the day of voting, voter scans their fingers along the fingerprint sensor of the smartphone. Accuracy depends on fingerprint sensor of the mobile phone. Detection of fraud is one of the core components of this system. It avoids casting of multiple votes by any voter.

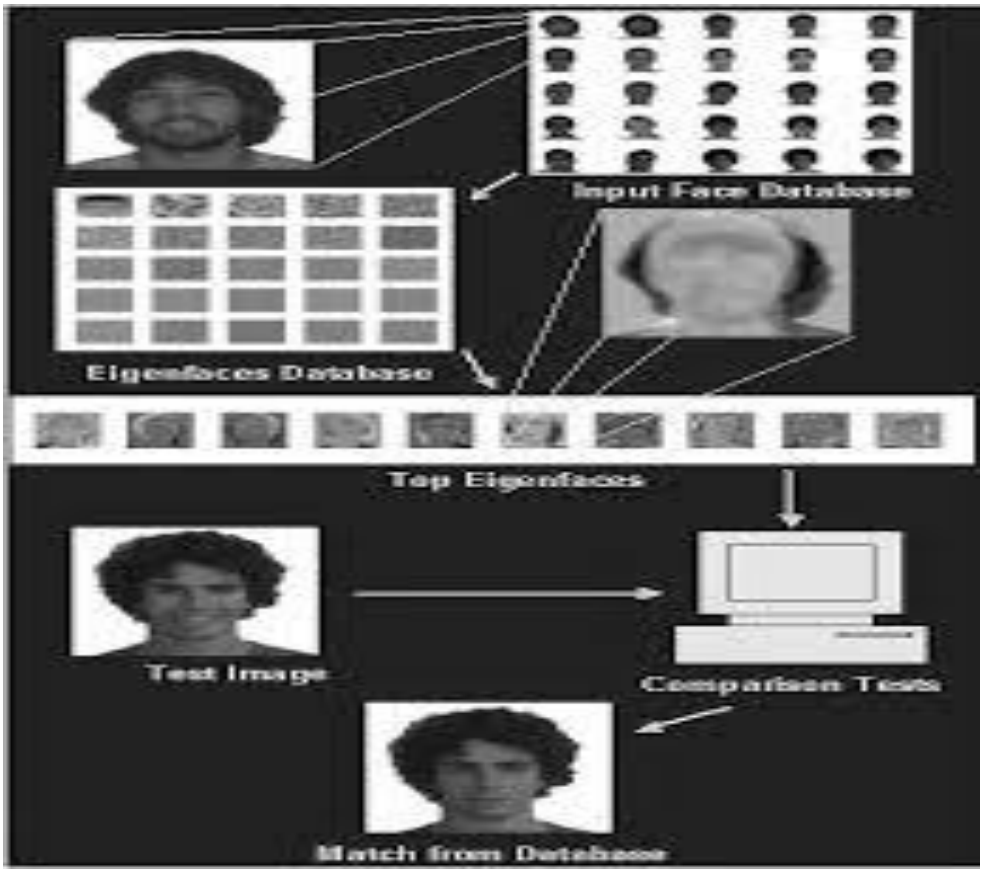
2.1.2 Secure Reliable Multimodal Biometric Fingerprint and Face Recognition:

This concept focuses on Image scanned for faces at different resolutions in a hierarchical face detector. Hierarchical system is a component-based face detector which precisely localizes the faces and extracts the components. The extracted components are merged into a single feature vector and then fed to the face recognizer. This proposed approach has been implemented in MATLAB. Implementation of this procedure results in the accurate recognition of face, more over the time complexity is very less and the total cost required for this implementation is very less,This can be adopted in commercial or law enforcement departments in future. Biometric image recognition is the process of studying the closest match region in between the examining images. The study of the recognition is done about the spatial pixels (picture element) among the image. Recognition of two different biometric features, finger print and face images are attempted. Finger print images are analyzed through every available pixel. And face images are analyzed on the basis of distance between various face mark identities and their relational distance difference. In this paper we have used principal component analysis which completely out forms standards technique and curve let transform. In the multi modal biometric technique, it is able to achieve good results with all security system for finger print and face recognition

2.1.3 E-Smart Voting System with Secure Data Identification using Cryptography:

This model focuses on the strategy used and functions of E smart voting system (ESVS) which is very much secured, biometric authentication system which is provided with verification system based on OTP that also improves the voting process in the time of election. The vote casted by the user is first encrypted and then stored in the database. In this concept Aadhar number of users is used for verification and identification of the voter. With smart voting system, voter is able to cast their vote with their mobile phone and avoid all kind of queues at polling booth. At first, Aadhar number is provided by the voter in the ESVS. The ESVS then uses the Aadhar number to authenticate the user through OTP which will be received on their registered Aadhar linked mobile number.

People without Smart phones are also able to cast their vote through ESVS with an additional step of authentication through highly secured Aadhar based biometric authentication. Smart Voting System successfully allows people to vote through their smart phones which helps to reduce the queues at the polling booth. Also, highly reliable biometric authentication mechanism is provided for people who do not want vote using smart phones thus prevent electoral fraud.



2.1.1.1 Face detection and comparison

2.2 Smart Voting System (Proposed Work)

- A python application is made which has user registration and user login
- Administrator has functionalities like accepting registrations and checking results using admin mail id. It helps to get results very fast.
- Voter login has functions, self-registration and voting.
- As the whole system is made of python tkinter, it makes the system very efficient and with very less bugs and light weight.
- First a face recognition system is built which makes use of LBP technique. It detects faces and identifies common patterns and gives the result how much percentage is matched.
- If it reaches the minimum value, it accepts the voter and permits voting.
- Then an OTP is sent to the registered email for verification.
- This is done using SMPLIB library.
- When the OTP sent to email is entered in the text box provided then it validates the OTP and permits for voting.
- There will be candidates and their parties in the application provided with an option to vote on their side.
- When clicked on it and pressed q button it stores all the values (name, id, email id, time of vote, date of vote, and leader voted for) and pops a message by thanking the voter for using the smart voting system.
- In this way the whole voting system is designed to minimize frauds like rigging and vote manipulation.

3. SYSTEM DESIGN

3.1 System Architecture (Flow Chart)

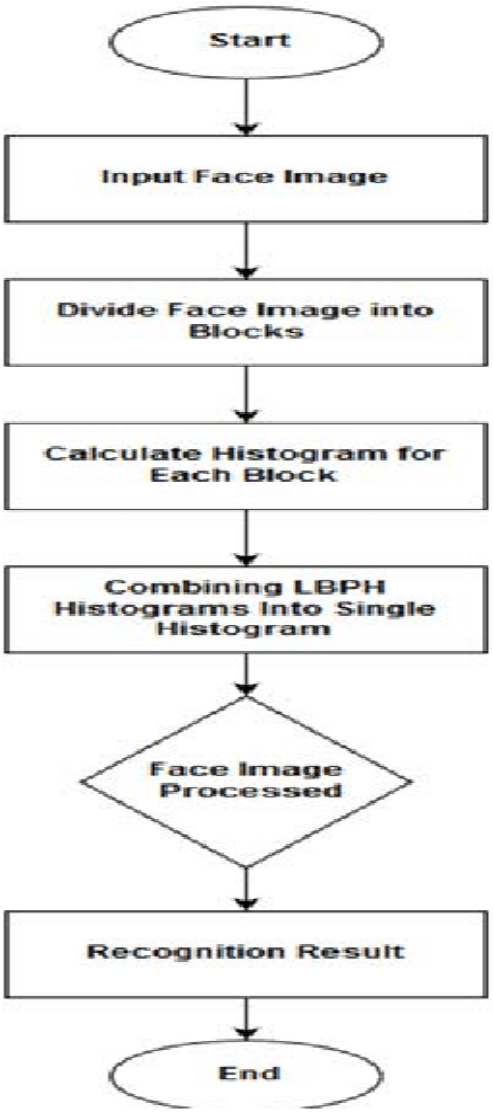


Fig 3.1.1 Algorithm Flow chart

3.2 Module-1 Description:

Registration: Taking voter Id, Name, Email ID and 60 photos of the voter
Appending the details to excel sheet them in Excel sheets
Storing images in a folder
Inputs: Voter Id, Name, Email ID, 60 photos
Outputs: Excel sheet which stores voter's details

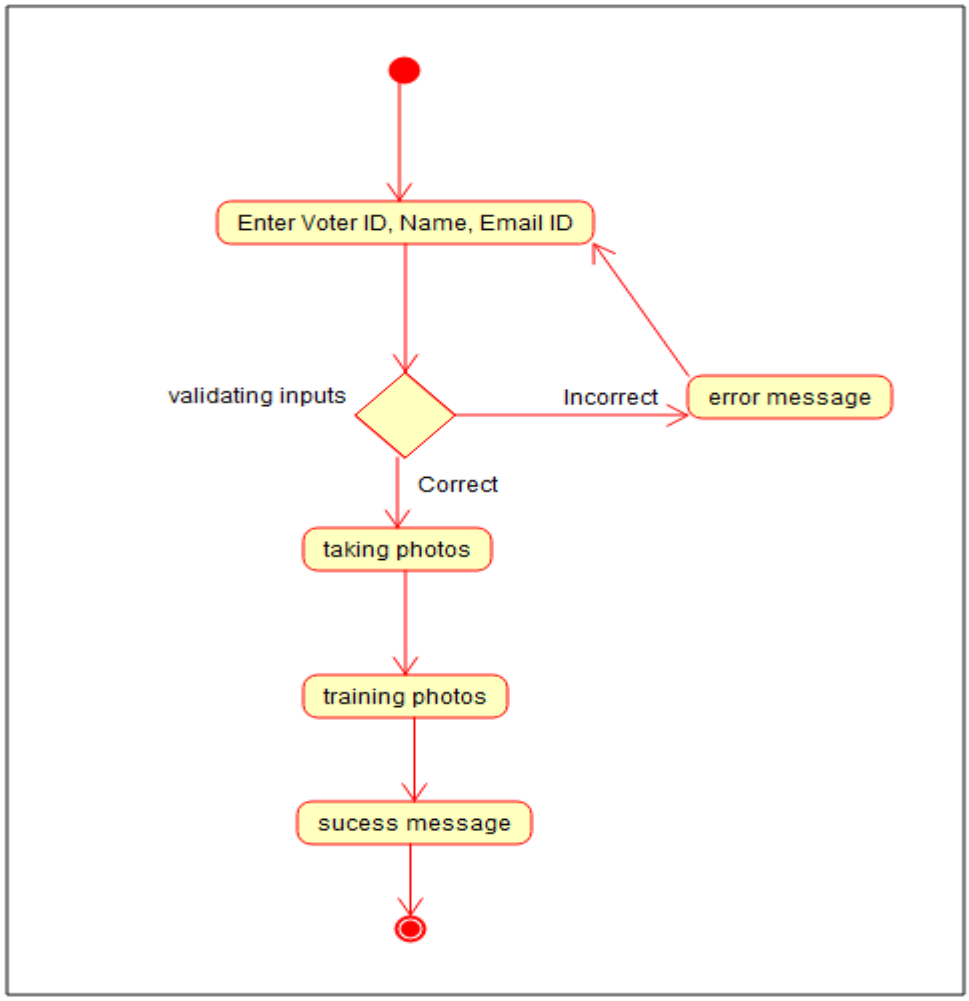


Fig 3.2.1 Activity Diagram for Module-1(Registration Description)

3.3 Module-2 Description:

Poll Vote: Enter email Id and request for OTP. Enter 4-digit OTP
Enter voter Id and Name and validate all input details
Track images and validate face
Poll Vote and quit. The vote along with details will be submitted if that's for the first time else it is shows error message
Inputs: Email Id, Voter ID, Name, photo, Vote
Outputs: Excel sheet of all votes

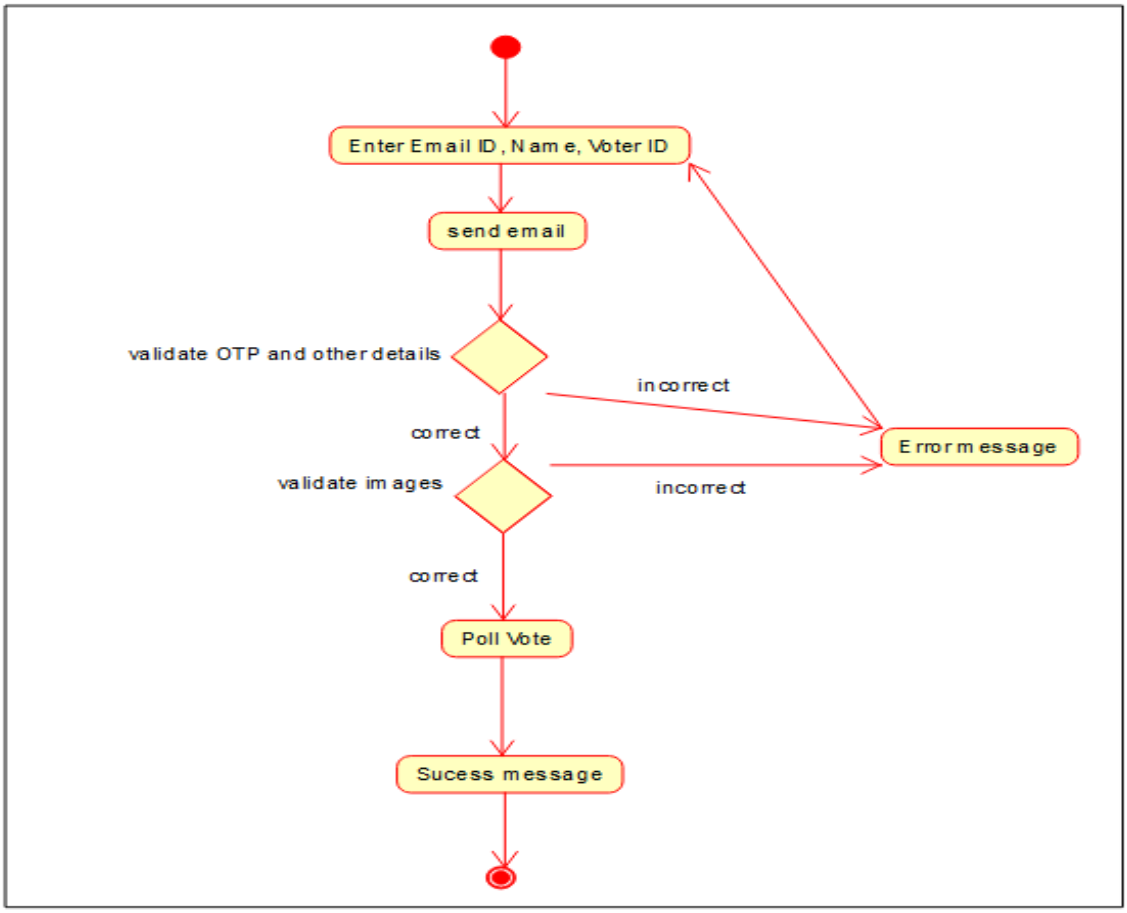


Fig 3.3.1 Activity Diagram for Module-2(Polling Vote Description)

4. IMPLEMENTATION

4.1 Environment Setup

We used tkinter for our working environment. The tkinter package ("Tk interface") is the standard Python interface to the Tk GUI toolkit. Both Tk and tkinter are available on most Unix platforms, as well as on Windows systems.

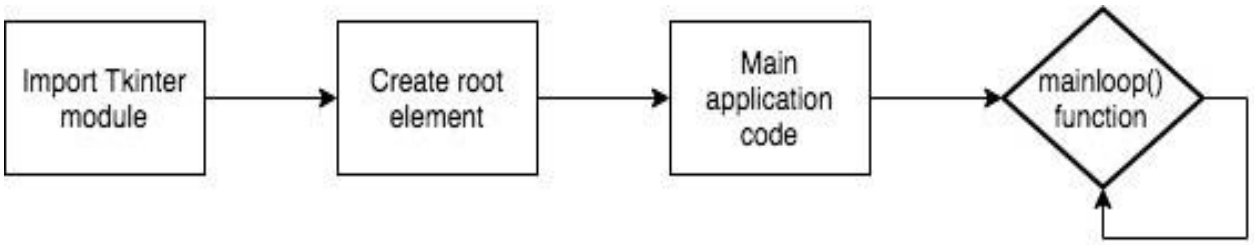


Fig 4.1.1 Environment Setup

Tkinter **is** the standard GUI library for Python. Python when combined with Tkinter provides a fast and easy way to create GUI applications. Tkinter provides a powerful object-oriented interface to the Tk GUI toolkit.

- Import the Tkinter module. Create the GUI application main window. Add one or more of the above-mentioned widgets to the GUI application. Enter the main event loop to take action against each event triggered by the user.

Example:

```
#!/usr/bin/python

import Tkinter
top = Tkinter.Tk()
# Code to add widgets will go here...
top.mainloop()
```

This would create a following window:

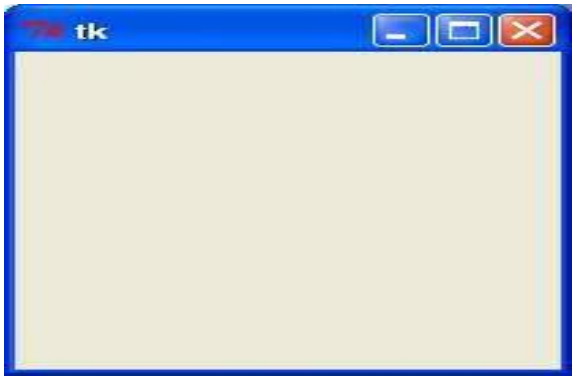


Fig 4.1.2 GUI Application Window

- root **is** the root window into which all other widgets go. It **is** an instance of the class Tk, and every tkinter application must have exactly one instance of this class. App **is** an instance of the class App, which **is** a subclass of Frame.

  To initialize tkinter, we have to create a Tk root widget, which is a window with a title bar and other decoration provided by the window manager. The root widget has to be created before any other widgets and there can only be one root widget.

```
root = tk.Tk()
```

- Write the main application code.
- mainloop() tells Python to run the Tkinter event loop. This method listens for events, such as button clicks or keypresses, and blocks any code that comes after it from running until the window it's called on is closed i.e. mainloop() is simply a method in the main window that executes what we wish to execute in an application (lets Tkinter to start running the application). As the name implies it will loop forever until the user exits the window or waits for any events from the user.

4.2 Implementation of each Module

4.2.1 Implementation of Module-1 - Registration

- Software Environment Used: Python tkinter, random, re, numpy, datetime, csv, cv2.
- Major Functions used: Registration: Taking voter Id, Name, Email ID and 60 photos of the voter
- Appending the details to excel sheet them in Excel sheets
- Storing images in a folder

Test cases for Module 1:
(Mention at least 2 test cases for each scenario)
Scenario 1: Voter Registration
Test case 1: Venkat
Input: Venkat, VoterID-ecw1112221, venkat***@gmail.com, OTP (varies every time), 60 pictures
Output: voter details, training algorithm.

Scenario 1: Voter Registration
Test case 2: Rithik
Input: Rithik, voterID-cse1234567, rithik***@gmail.com, OTP (varies every time), 60 pictures
Output: voter details, training algorithm.
……

4.2.2 Implementation of Module-2 - Poll Vote

- Software Environment Used: Python tkinter, random, re, numpy, datetime, csv, cv2, PIL, smtlib.
- Major Functions used Enter email Id and request for OTP. Enter 4-digit OTP
- Enter voter Id and Name and validate all input details
- Track images and validate face
- Poll Vote and quit. The vote along with details will be submitted if that's for the first time else it is shows error message

Test cases for Module 2:
(Mention at least 2 test cases for each scenario)
Scenario 1: Polling Vote
Test case 1: Venkat
Input: Venkat, VoterID-ecw1112221, venkat***@gmail.com, OTP (varies every time), vote
Output: Votes sheet

Scenario 1: Polling Vote
Test case 2: Rithik
Input: Rithik, voterID-cse1234567, rithik***@gmail.com, OTP (varies every time), vote
Output: Votes Sheet

4.3 Integration and Deployment

- The project is having the inputs taken from the user interface which is integrated to send the inputs both text and images to the backend and process the input using the algorithm and verify the image of the input with the previous data to match the face which helps to authenticate the voter.
- The project is deployed on custom PC with a Webcam which takes the inputs and compares with the existing data and authenticate the user

Fig 4.3.1 Pictures of voter taken as input

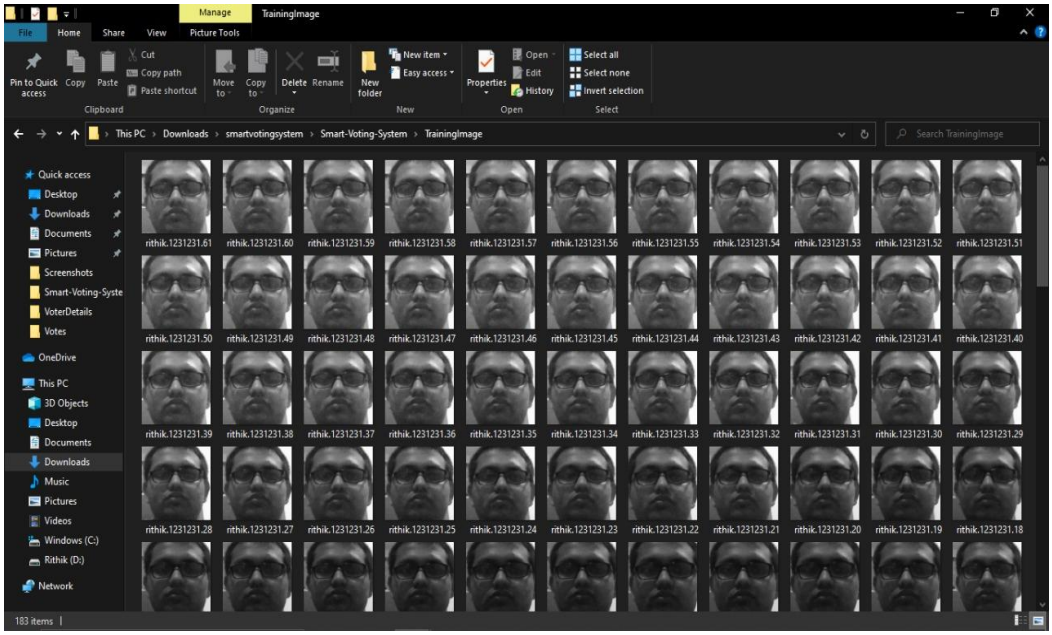
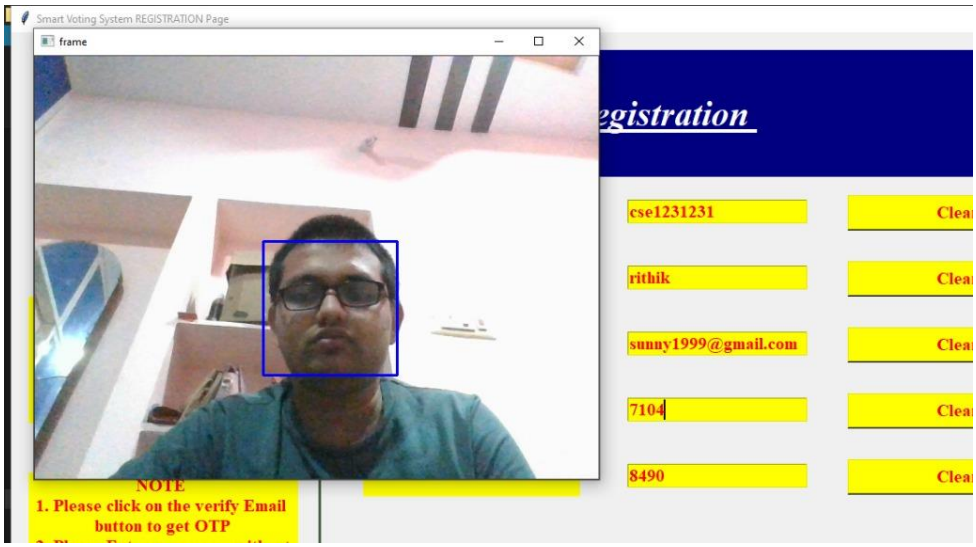


Fig 4.3.2 Taking voters face pictures for future reference



5. EVALUATION

5.1 Evaluation Procedure

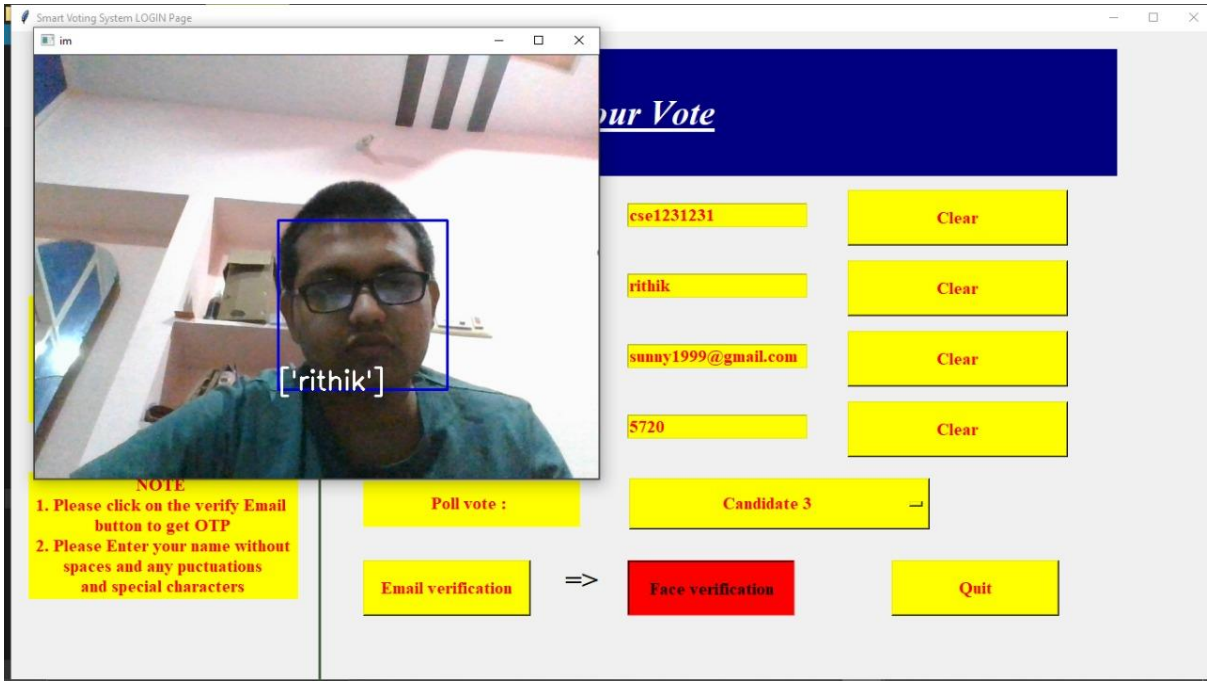
Objective Evaluation

(Using Evaluation Metrics)

Data set:

- 60 Images of every voter are taken from webcam
- Each image is named by id, name and its unique number from 1 to 60
- Sample image name is 12312. Rithik.52 and stored in a folder
- They are trained with Local Binary Pattern Histogram algorithm in open CV
- Unknown images which are found while validating face at the voting time are saved into a file
- Local Binary Pattern Histogram algorithm gives highest accuracy when good amount of light is present
- So, it identifies the person with a very high accuracy
- It identifies the faces even it is changed due to aging

Fig 5.1.1 Face verification of the voter



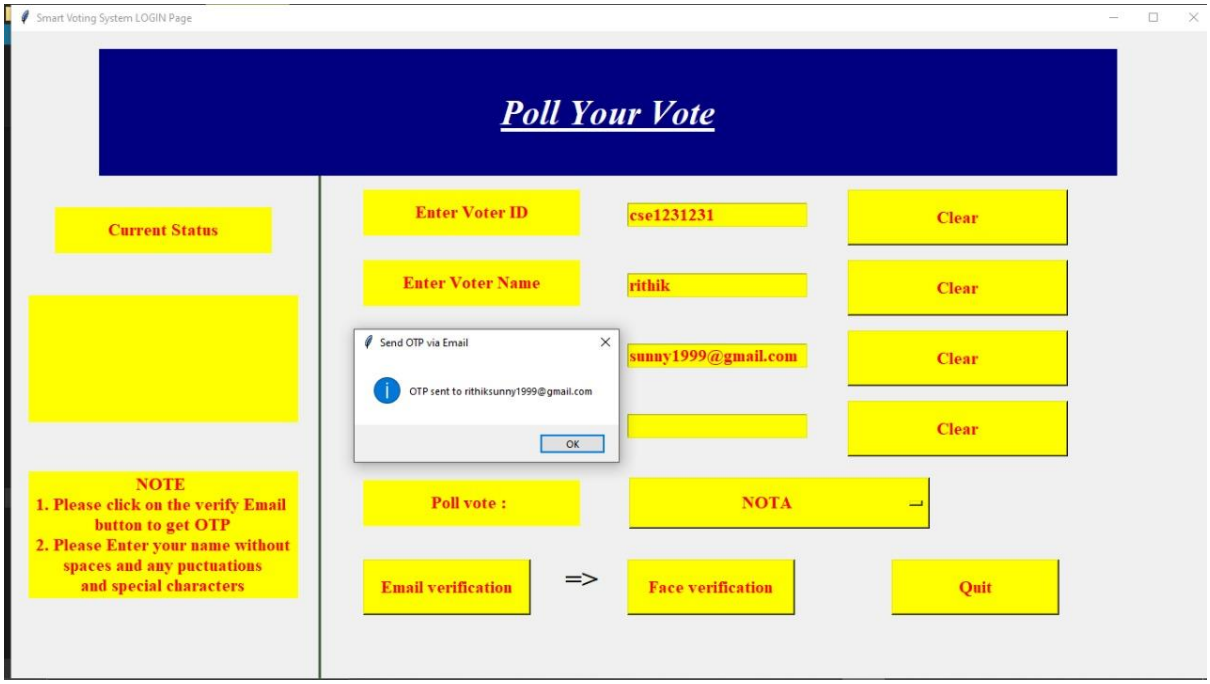
- If the voters face is registered and authenticated, then name of the voter will be displayed while verifying the face during polling of vote.

5.2 Test Cases

Fig 5.2.1 Verifying the Email of the voter

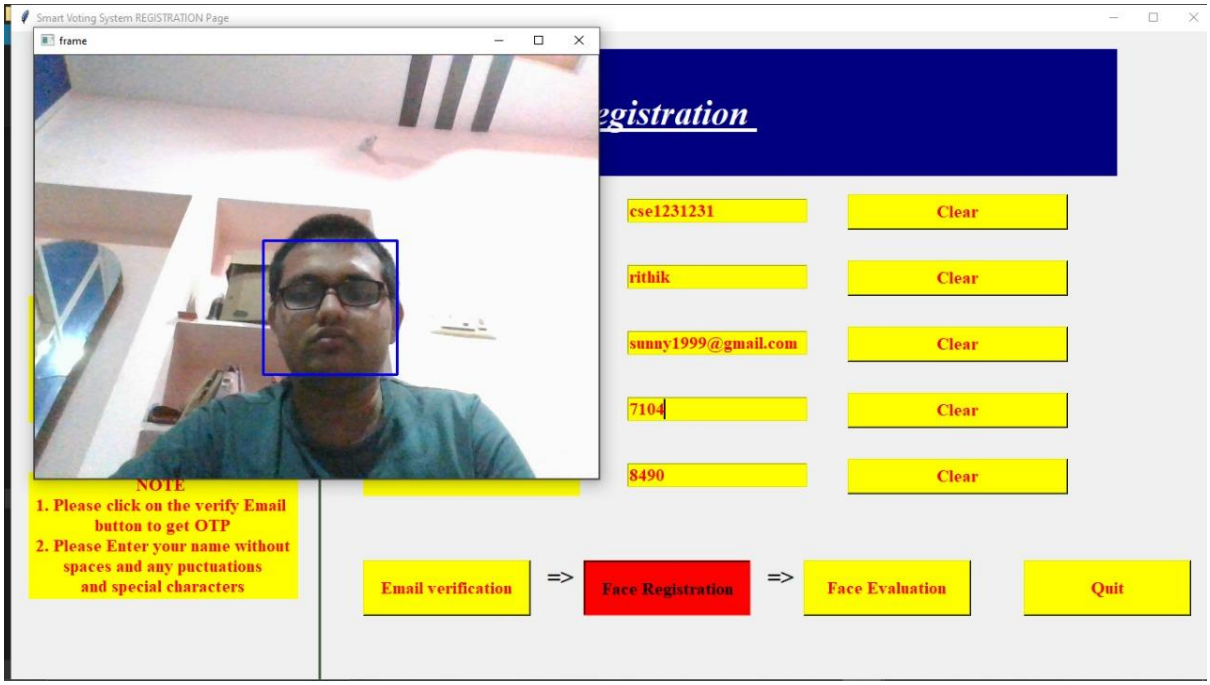
- Here the voters voter ID ,name and registered mail id should be given so that we can verify whether the Voter is registered or not which is first level of security.

Fig 5.2.2 Sending OTP to the verified email of the voter



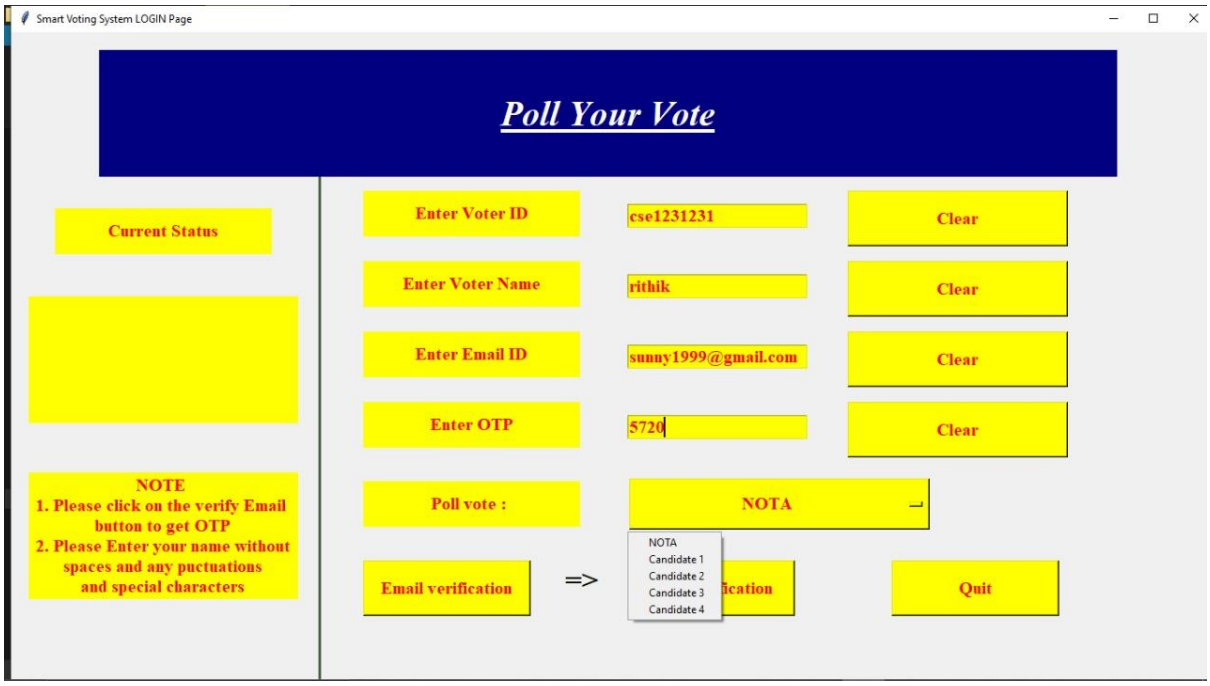
- An OTP will be sent to the mail id of the voter, after the mail is registered and verification is done.

Fig 5.2.3 Face Registration of the voter



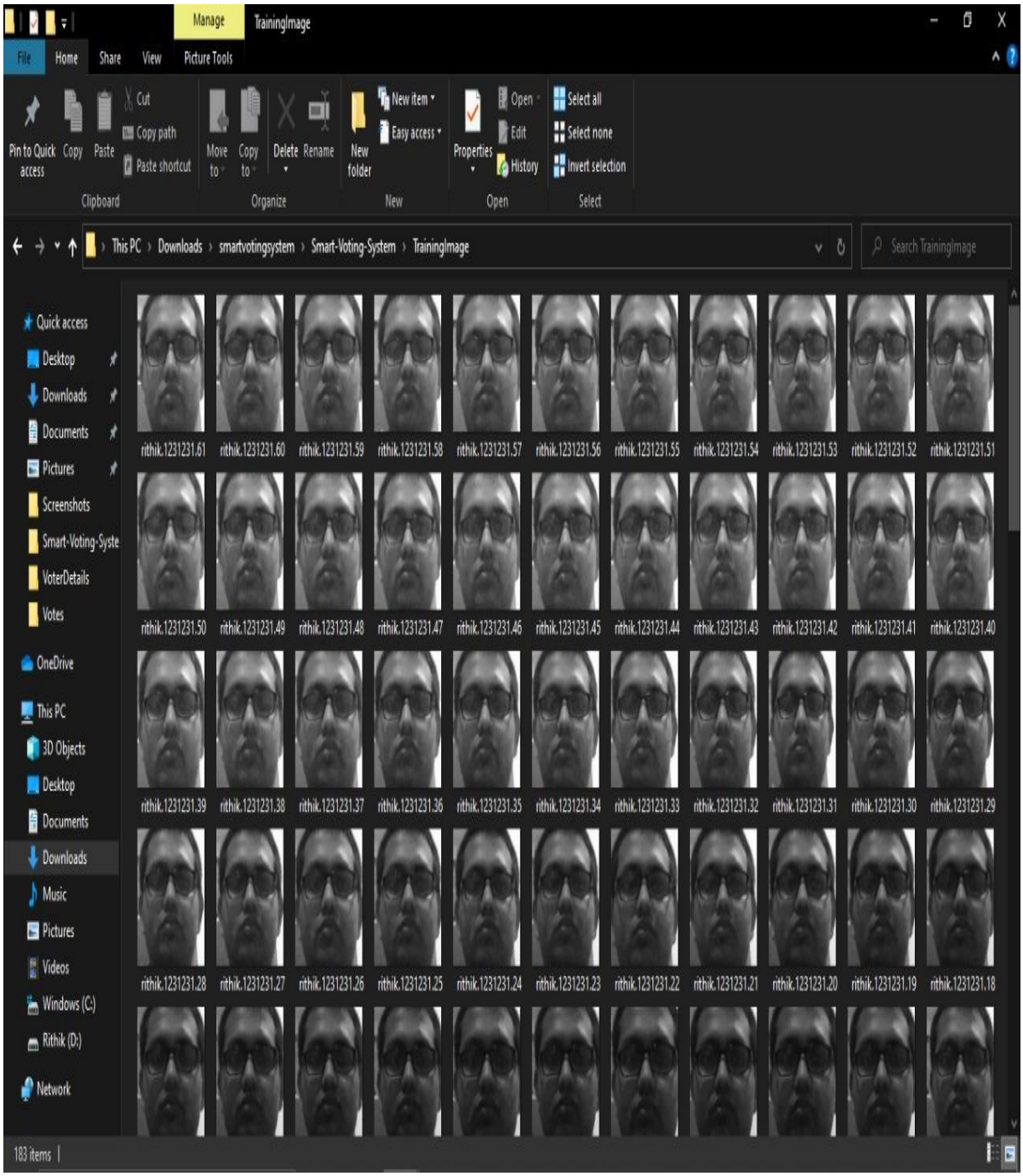
- Once the Email verification is done we will proceed to face registration of the voter,where 60 images of the voter will be clicked for future reference.

Fig 5.2.4 Selection of vote options(NOTA, candidate 1, candidate 2 …)



- Here the voter has to select the candidate to whom he wants to give his vote i.e., candidate 1/candidate 2/NOTA etc.

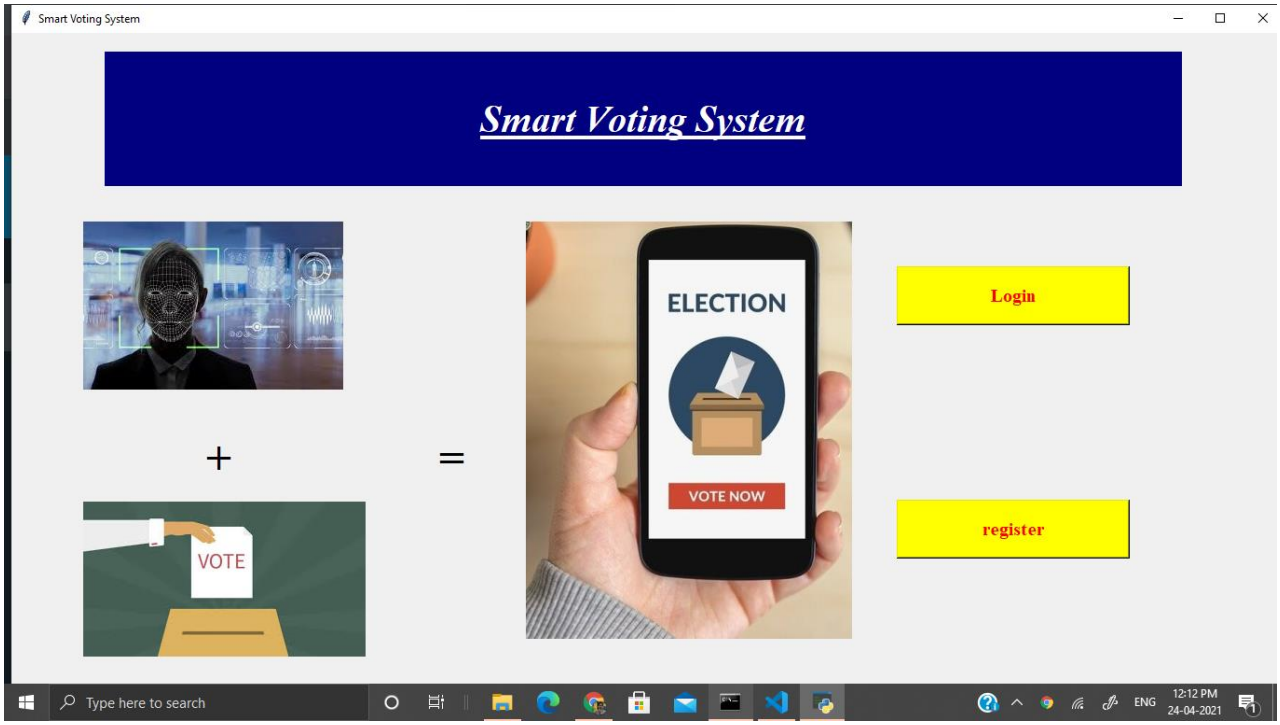
Fig 5.2.5 Storing the images of voter in database



- These are the images of the voter taken at the time of registration so that, we can verify the face of the voter by comparing the images of the voter existing in database with the image of the voter at the time of voting.

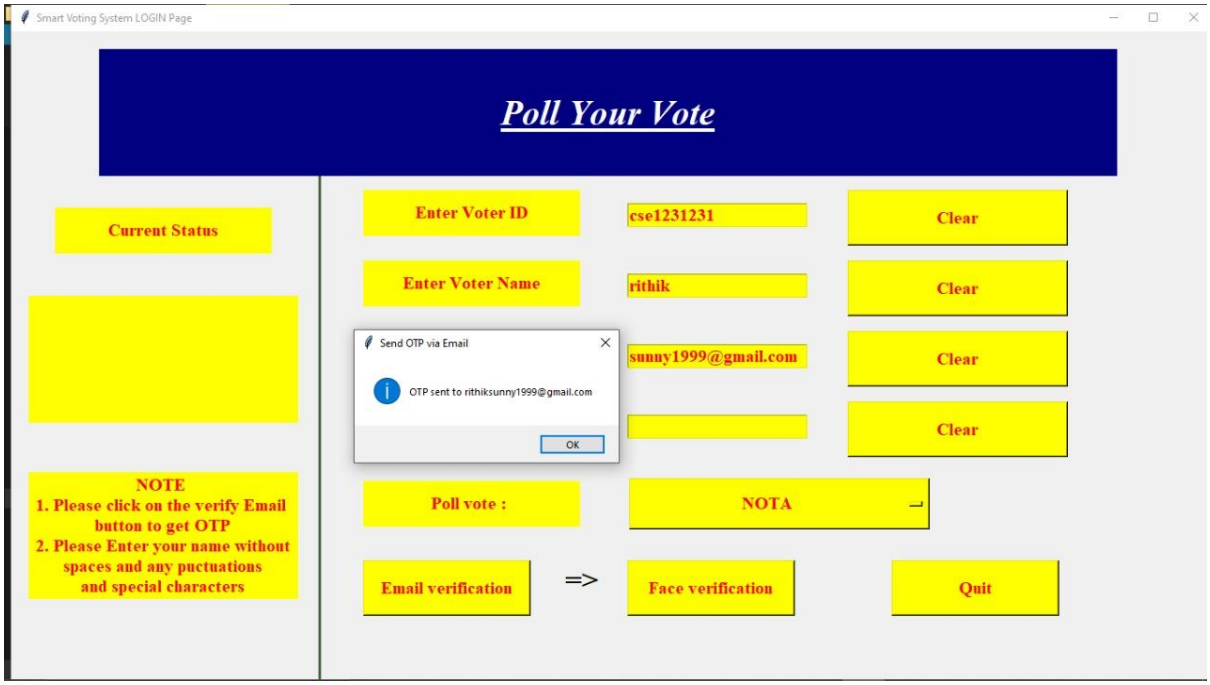
5.3 Results

Fig 5.3.1 Home Page to Login/Register



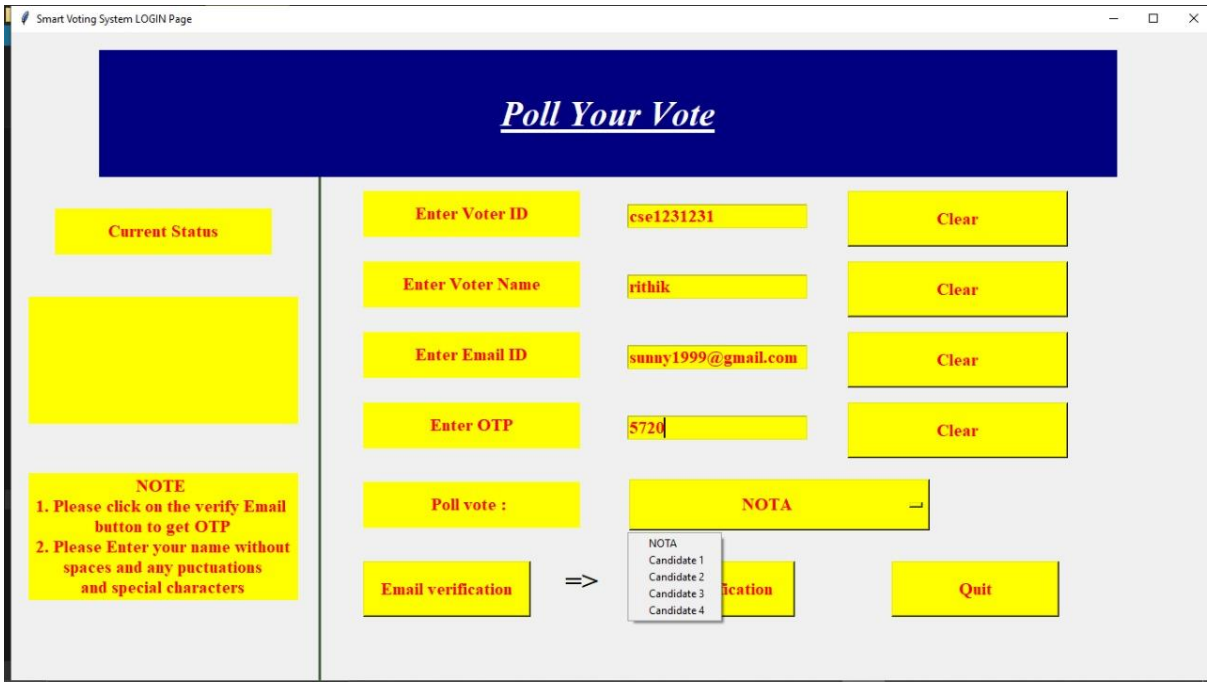
➢ This is the home page where the registered users can login with their details given at the time of registration and the new users can get registered.

Fig 5.3.2 Entering the voter details and OTP sent to registered Email



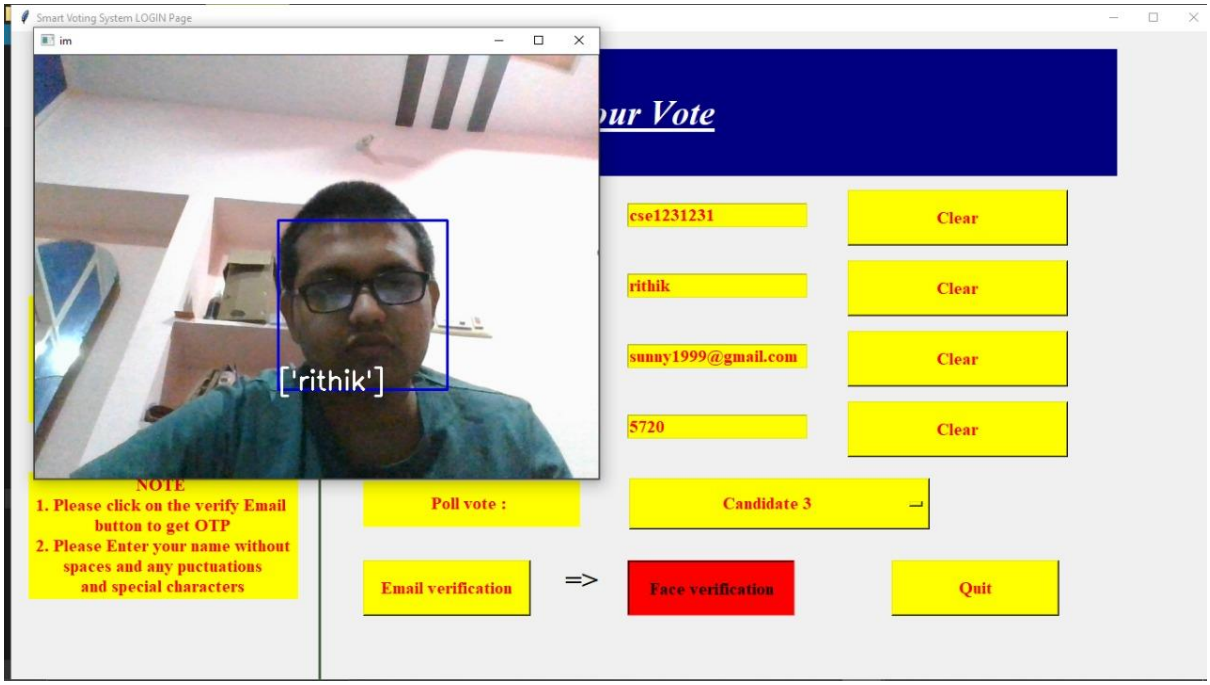
➢ Once the Voter enters his Voter ID, Name and mail id an OTP will be sent to mail id which was given at the time of registration .

Fig 5.3.3 Entering the OTP received at registered Email (Email Verification) and poling vote



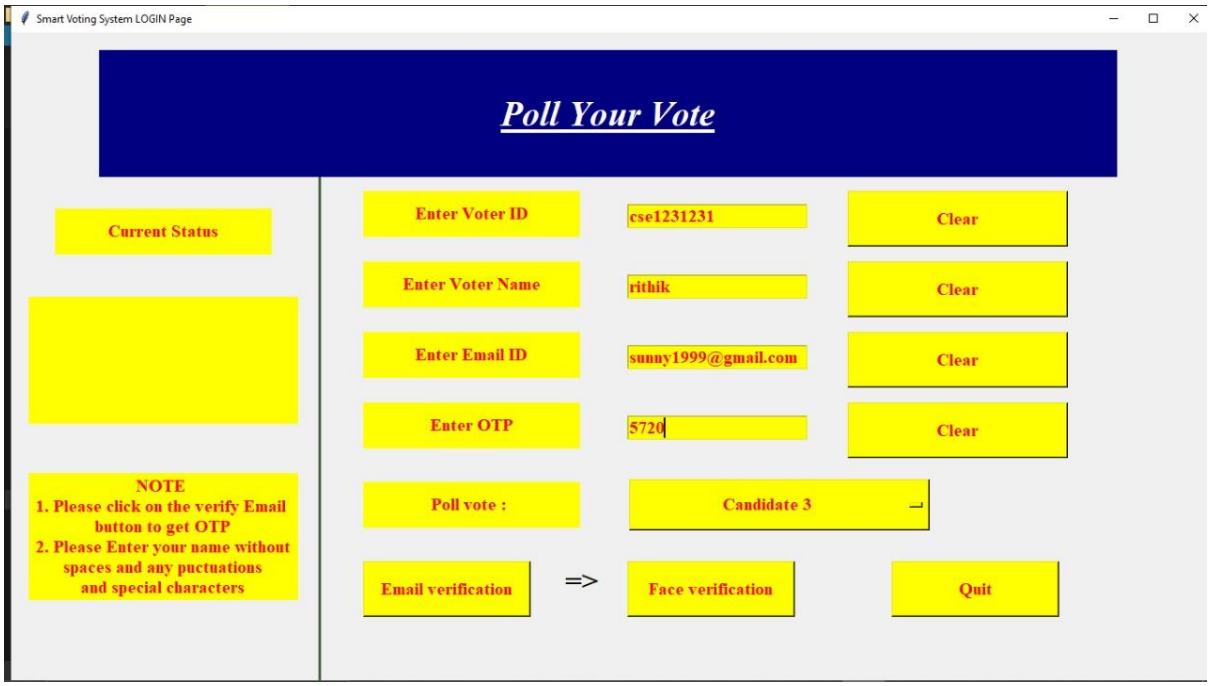
- The OTP which was sent to voter's mail id should be entered if it is valid only then the voter can move to next step of voting process, i.e.,it is the second level of security.

Fig 5.3.4 Face Verification by comparing with existing database



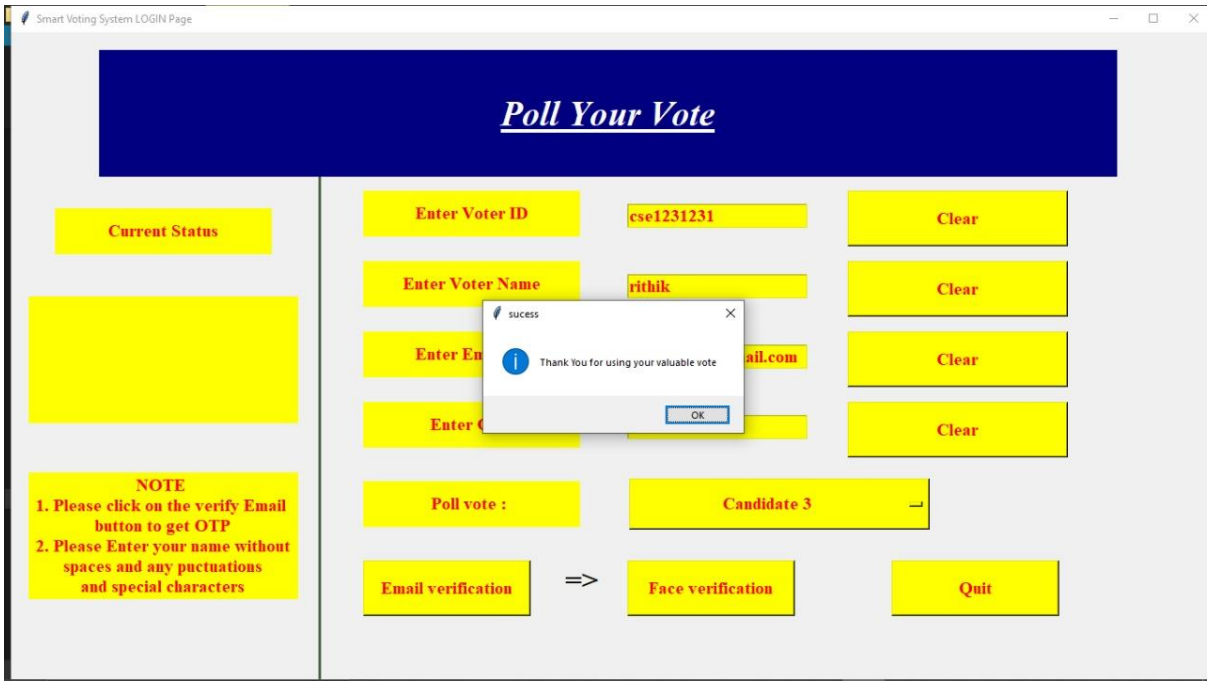
- After Email verification, Face verification is done which is third level of security ,where if the voter is registered then details of the voter will be displayed by comparing the image of the voter at the time voting with image of the voter with existing data.

Fig 5.3.5 Selecting the Candidate to whom voter wants to vote



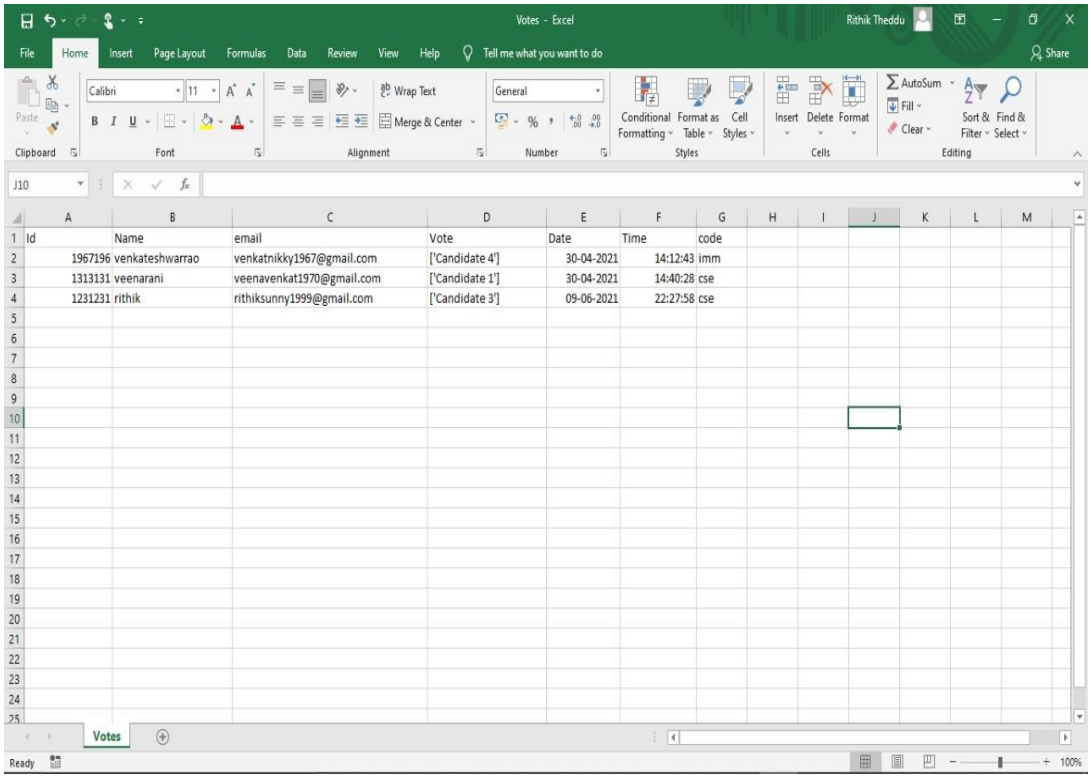
- Once the three steps verification is done,the voter has to select the candidate to whom he wants to give his valuable vote.

Fig 5.3.6 Polling the vote



- Once polling of the vote is successfully done, a message will be displayed saying “Thank You for using your valuable vote”.

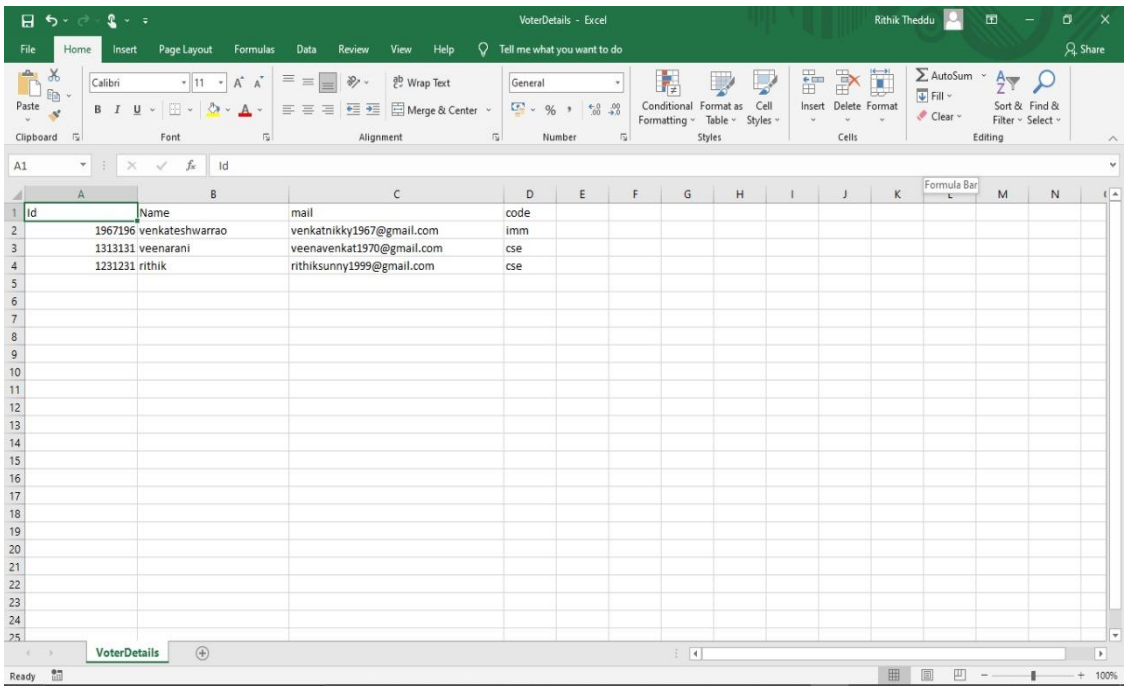
Fig 5.3.7 Details and votes of the registered and verified voters(1)



Id	Name	email	Vote	Date	Time	code
1967196	venkateshwarrao	venkatnikky1967@gmail.com	['Candidate 4']	30-04-2021	14:12:43	imm
1313131	veenarani	veenavenkat1970@gmail.com	['Candidate 1']	30-04-2021	14:40:28	cse
1231231	rithik	rithiksunny1999@gmail.com	['Candidate 3']	09-06-2021	22:27:58	cse

- The details of the voters and their votes are automatically saved in an excel sheet where we can see the total number of votes given to an particular candidate.

Fig 5.3.8 Details and votes of the registered and verified voters(2)



Id	Name	mail	code
1967196	venkateshwarrao	venkatnikky1967@gmail.com	imm
1313131	veenarani	veenavenkat1970@gmail.com	cse
1231231	rithik	rithiksunny1999@gmail.com	cse

- These are details of the registered voters stored in an excel sheet so that we can have a count of total number of registered voters.

6. CONCLUSION AND FUTURE ENHANCEMENTS

As we see that existing voting system has many defects such as lengthy process, time taking, not secure, bogus voting, no security level but now we can say that our approach is more useful and secure from the existing system. Since, we are using three level of security in this proposed system the false voters can be easily identified.

The facial authentication technique is very much useful in identifying the fraud voters. As data is stored in decentralized repository so, data is accessible at any time as well as backup of the data is possible. It requires less man power and resources.

The database needs to be updated every year or before election so that new eligible citizens may be enrolled and those who are dead are removed from the voter list

Future scope:

1. A secured database can be used to save details and votes
2. An API can be used to save data base in a cloud server with secure communication.

REFERENCES

[1] https://towardsdatascience.com/face-recognition-how-lbph-works90ec258c3d6b

[2] http://www.ijirst.org/articles/IJIRSTV5I11016.pdf

[3] https://www.ijitee.org/wpcontent/uploads/papers/v9i6/F4806049620.pdf

[4] https://www.codershubb.com/send-otp-via-email-using-python

[5] https://www.tutorialspoint.com/python/python_gui_programming.html

[6] https://datatofish.com/convert-text-file-to-csv-using-python-tool-included

[7] https://www.c-sharpcorner.com/blogs/basics-for-displaying-image-in-tkinter-python

[8] https://www.google.co.in/amp/s/www.geeksforgeeks.org/create-a-guito-convert-csv-file-into-excel-file-using-python/amp

[9] https://www.javatpoint.com/python-read-excel-file

[10] https://www.google.co.in/amp/s/www.geeksforgeeks.org/tkinter-application-to-switch-between-different-page-frames/amp

APPENDIX

Voter Registration

```
import tkinter as tk
from tkinter import Message ,Text, Canvas
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
from tkinter import StringVar
from tkinter import OptionMenu
from tkinter import Label
from tkinter import messagebox
import smtplib
import random
import re
window = tk.Tk()
#helv36 = tk.Font(family='Helvetica', size=36, weight='bold')
window.title("Smart Voting System REGISTRATION Page")
width= window.winfo_screenwidth()
height= window.winfo_screenheight()
window.geometry("%dx%d" % (width, height))

w = Canvas(window, width=width, height=height)
w.pack()
w.create_line(350, 150, 350, 800, fill="#476042", width=3)
```

```
dialog_title = 'QUIT'
dialog_text = 'Are you sure?'

#answer = messagebox.askquestion(dialog_title, dialog_text)

#window.geometry('1280x720')
window.configure(background='blue')

#window.attributes('-fullscreen', True)

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

#path = "profile.jpg"

#Creates a Tkinter-compatible photo image, which can be used everywhere Tkinter expects
an image object.
#img = ImageTk.PhotoImage(Image.open(path))

#The Label widget is a standard Tkinter widget used to display a text or image on the screen.
#panel = tk.Label(window, image = img)



#panel.pack(side = "left", fill = "y", expand = "no")

#cv_img = cv2.imread("img541.jpg")
#x, y, no_channels = cv_img.shape
#canvas = tk.Canvas(window, width = x, height =y)
#canvas.pack(side="left")
#photo = PIL.ImageTk.PhotoImage(image = PIL.Image.fromarray(cv_img))
# Add a PhotoImage to the Canvas
#canvas.create_image(0, 0, image=photo, anchor=tk.NW)

#msg = Message(window, text='Hello, world!')
```

```
# Font is a tuple of (font_family, size_in_points, style_modifier_string)

message = tk.Label(window, text=" Voters Registration
" ,bg="#000080"  ,fg="white"  ,width=50  ,height=3,font=('times', 30, 'italic bold underline'))

message.place(x=100, y=20)


lbl = tk.Label(window, text="Enter Voter
ID",width=20  ,height=2  ,fg="red"  ,bg="yellow" ,font=('times', 15, ' bold ') )
lbl.place(x=400, y=175)

txt = tk.Entry(window,width=20  ,bg="yellow" ,fg="red",font=('times', 15, ' bold '))
txt.place(x=700, y=190)

lbe = tk.Label(window, text="Enter Email
ID",width=20  ,height=2  ,fg="red"  ,bg="yellow" ,font=('times', 15, ' bold ') )
lbe.place(x=400, y=325)

txte = tk.Entry(window,width=20  ,bg="yellow" ,fg="red",font=('times', 15, ' bold '))
txte.place(x=700, y=340)


lbev = tk.Label(window, text="Enter
OTP",width=20  ,height=2  ,fg="red"  ,bg="yellow" ,font=('times', 15, ' bold ') )
lbev.place(x=400, y=400)


txtev = tk.Entry(window,width=20  ,bg="yellow" ,fg="red",font=('times', 15, ' bold '))
txtev.place(x=700, y=415)

lbl2 = tk.Label(window, text="Enter Voter
Name",width=20  ,fg="red"  ,bg="yellow"    ,height=2 ,font=('times', 15, ' bold '))
lbl2.place(x=400, y=250)
```

```
txt2 = tk.Entry(window,width=20  ,bg="yellow"  ,fg="red",font=('times', 15, ' bold ')  )
txt2.place(x=700, y=265)


lbev2 = tk.Label(window, text="Enter admin
OTP",width=20  ,height=2  ,fg="red"  ,bg="yellow" ,font=('times', 15, ' bold ') )
lbev2.place(x=400, y=475)


txtev2 = tk.Entry(window,width=20  ,bg="yellow" ,fg="red",font=('times', 15, ' bold '))
txtev2.place(x=700, y=490)

lbl3 = tk.Label(window, text=" Current Status
",width=20  ,fg="red"  ,bg="yellow"  ,height=2 ,font=('times', 15, ' bold'))
lbl3.place(x=50, y=200)

message = tk.Label(window, text="" ,bg="yellow"  ,fg="red"  ,width=25  ,height=6,
activebackground = "yellow" ,font=('times', 15, ' bold '))
message.place(x=20, y=300)

note = tk.Label(window, text="NOTE \n1. Please click on the verify Email \nbutton to get
OTP\n2. Please Enter your name without\nspaces and any puctuations\nand special
characters" ,bg="yellow"  ,fg="red"  ,width=25  ,height=6, activebackground =
"yellow" ,font=('times', 15, ' bold '))
note.place(x=20, y=500)

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

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

```
def clear3():
    txte.delete(0, 'end')
    res = ""
    message.configure(text= res)

def clear4():
    txtev.delete(0, 'end')
    res = ""
    message.configure(text= res)
def clear5():
    txtev2.delete(0, 'end')
    res = ""
    message.configure(text= res)

def is_number(s):
    try:
        float(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())
```

```
 mailId=(txte.get())

  x = re.findall("[a-z]{3}[0-9]{7}", Id)
   if(len(x)==1 and name.isalpha() and otp==txtev.get() and otp1==txtev2.get()):
      cam = cv2.VideoCapture(0)
      global Idalpha
      Idalpha=Id[:3]
      Id=Id[3:]

    harcascadePath = "haarcascade_frontalface_default.xml"
      detector=cv2.CascadeClassifier(harcascadePath)
      sampleNum=0
      while(True):
         ret, img = cam.read()
         gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
         faces = detector.detectMultiScale(gray, 1.2, 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 miliseconds
         if cv2.waitKey(100) & 0xFF == ord('q'):
            break
         # break if the sample number is morethan 100
         elif sampleNum>60:
            break
      cam.release()
      cv2.destroyAllWindows()
      res = "Registered Face for ID : " + Id +"\n Name : "+ name
```

```
 row = [Id , name, mailId, Idalpha]
        with open('VoterDetails\VoterDetails.csv','a+') as csvFile:
            writer = csv.writer(csvFile)
            writer.writerow(row)
        csvFile.close()
        message.configure(text= res)

else:
if(len(x)!=1):

 res = "Enter correct Id with 3 alpahabets\n and 7 numbers"
            message.configure(text= res)
        if(name.isalpha() != True):
            res = "Enter Alphabetical Name"
            message.configure(text= res)
        if(len(txte.get())==0):
            res = "Enter Email ID"
            message.configure(text= res)
        if(len(txtev.get())==0):
            res = "Enter OTP sent to your Email ID"
            message.configure(text= res)
        if(txtev.get()==otp):
            res = "Enter correct OTP sent to \nyour Email ID"
            message.configure(text= res)
        if(len(txtev2.get())==otp1):
            res = "Enter correct OTP sent to \nadmin's Email ID"
            message.configure(text= res)

def TrainImages():
    recognizer = cv2.face_LBPHFaceRecognizer.create()#recognizer =
cv2.face.LBPHFaceRecognizer_create()#$cv2.createLBPHFaceRecognizer()
    harcascadePath = "haarcascade_frontalface_default.xml"
    detector =cv2.CascadeClassifier(harcascadePath)
    faces,Id = getImagesAndLabels("TrainingImage")
```

```
 recognizer.train(faces, np.array(Id))
    recognizer.save("TrainingImageLabel\Trainner.yml")
    res = "Face Evaluated"#+",".join(str(f) for f in Id)
    message.configure(text= res)
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)]
    #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 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.append(imageNp)
        Ids.append(Id)
    return faces,Ids
def send():
    try:
        s = smtplib.SMTP("smtp.gmail.com" , 587)  # 587 is a port number
        s.starttls()
        s.login("********************" , "************")
        global otp,otp1
        otp=random.randint(1000, 9999)
        otp1=random.randint(1000, 9999)
```

```
  otp = str(otp)
        otp1 = str(otp1)
        s.sendmail("****************" , txte.get() , otp)
        s.sendmail("**************" , "********************" , otp1)
        messagebox.showinfo("Send OTP via Email", f"OTP sent to {txte.get()} and admin")
        s.quit()

    except:
        messagebox.showinfo("Send OTP via Email", "Please enter the valid email address OR
check an internet connection")

sendMail = tk.Button(window, text="Email verification",

command=send  ,fg="red"  ,bg="yellow"  ,width=15  ,height=2, activebackground =
"Red" ,font=('times', 15, ' bold '))

sendMail .place(x=400, y=600)
lbx = tk.Label(window, text="=>",width=2  ,height=1  ,fg="black"   ,font=('times', 20, ' bold
') )
lbx.place(x=605, y=600)

clearButton = tk.Button(window, text="Clear",
command=clear  ,fg="red"  ,bg="yellow"  ,width=20  ,height=1 ,activebackground =
"Red" ,font=('times', 15, ' bold '))
clearButton.place(x=950, y=185)
clearButton2 = tk.Button(window, text="Clear",

command=clear2  ,fg="red"  ,bg="yellow"  ,width=20  ,height=1, activebackground =
"Red" ,font=('times', 15, ' bold '))
clearButton2.place(x=950, y=260)
clearButton3 = tk.Button(window, text="Clear",
command=clear3  ,fg="red"  ,bg="yellow"  ,width=20  ,height=1 ,activebackground =
"Red" ,font=('times', 15, ' bold '))
```

```
clearButton3.place(x=950, y=335)
clearButton4 = tk.Button(window, text="Clear",
command=clear4  ,fg="red"  ,bg="yellow"  ,width=20  ,height=1, activebackground =
"Red" ,font=('times', 15, ' bold '))
clearButton4.place(x=950, y=410)
clearButton5 = tk.Button(window, text="Clear",
command=clear5  ,fg="red"  ,bg="yellow"  ,width=20  ,height=1, activebackground =
"Red" ,font=('times', 15, ' bold '))
clearButton5.place(x=950, y=485)
takeImg = tk.Button(window, text="Face Registration",
command=TakeImages  ,fg="red"  ,bg="yellow"  ,width=15  ,height=2, activebackground =
"Red" ,font=('times', 15, ' bold '))
takeImg.place(x=650, y=600)
lby = tk.Label(window, text="=>",width=2  ,height=1  ,fg="black"   ,font=('times', 20, ' bold
') )
lby.place(x=855, y=600)
trainImg = tk.Button(window, text="Face Evaluation",
command=TrainImages  ,fg="red"  ,bg="yellow"  ,width=15  ,height=2, activebackground =
"Red" ,font=('times', 15, ' bold '))
trainImg.place(x=900, y=600)

quitWindow = tk.Button(window, text="Quit",
command=window.destroy  ,fg="red"  ,bg="yellow"  ,width=15  ,height=2, activebackground
= "Red" ,font=('times', 15, ' bold '))
quitWindow.place(x=1150, y=600)

window.mainloop()
```

Voter Login

```
import tkinter as tk
from tkinter import Message ,Text, Canvas
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
from tkinter import StringVar
from tkinter import OptionMenu
from tkinter import Label

from tkinter import messagebox
import smtplib
import random
import re

window = tk.Tk()
#helv36 = tk.Font(family='Helvetica', size=36, weight='bold')
window.title("Smart Voting System LOGIN Page")
width= window.winfo_screenwidth()
height= window.winfo_screenheight()
window.geometry("%dx%d" % (width, height))
w = Canvas(window, width=width, height=height)
w.pack()
w.create_line(350, 150, 350, 800, fill="#476042", width=3)

dialog_title = 'QUIT'
dialog_text = 'Are you sure?'
```

```
#answer = messagebox.askquestion(dialog_title, dialog_text)

#window.geometry('1280x720')
window.configure(background='blue')

#window.attributes('-fullscreen', True)

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

#path = "profile.jpg"

#Creates a Tkinter-compatible photo image, which can be used everywhere Tkinter expects
an image object.
#img = ImageTk.PhotoImage(Image.open(path))

#The Label widget is a standard Tkinter widget used to display a text or image on the screen.
#panel = tk.Label(window, image = img)

#panel.pack(side = "left", fill = "y", expand = "no")

#cv_img = cv2.imread("img541.jpg")
#x, y, no_channels = cv_img.shape
#canvas = tk.Canvas(window, width = x, height =y)
#canvas.pack(side="left")
#photo = PIL.ImageTk.PhotoImage(image = PIL.Image.fromarray(cv_img))
# Add a PhotoImage to the Canvas
#canvas.create_image(0, 0, image=photo, anchor=tk.NW)

#msg = Message(window, text='Hello, world!')
# Font is a tuple of (font_family, size_in_points, style_modifier_string)

message = tk.Label(window, text="Poll Your
Vote" ,bg="#000080"  ,fg="white"  ,width=50  ,height=3,font=('times', 30, 'italic bold
underline'))
```

```
message.place(x=100, y=20)

lbl = tk.Label(window, text="Enter Voter
ID",width=20  ,height=2  ,fg="red"  ,bg="yellow" ,font=('times', 15, ' bold ') )
lbl.place(x=400, y=180)

txt = tk.Entry(window,width=20  ,bg="yellow" ,fg="red",font=('times', 15, ' bold '))
txt.place(x=700, y=195)

lbe = tk.Label(window, text="Enter Email
ID",width=20  ,height=2  ,fg="red"  ,bg="yellow" ,font=('times', 15, ' bold ') )
lbe.place(x=400, y=340)

txte = tk.Entry(window,width=20  ,bg="yellow" ,fg="red",font=('times', 15, ' bold '))
txte.place(x=700, y=355)



lbev = tk.Label(window, text="Enter
OTP",width=20  ,height=2  ,fg="red"  ,bg="yellow" ,font=('times', 15, ' bold ') )
lbev.place(x=400, y=420)

txtev = tk.Entry(window,width=20  ,bg="yellow" ,fg="red",font=('times', 15, ' bold '))
txtev.place(x=700, y=435)

lbl2 = tk.Label(window, text="Enter Voter
Name",width=20  ,fg="red"  ,bg="yellow"    ,height=2 ,font=('times', 15, ' bold '))
lbl2.place(x=400, y=260)

# Dropdown menu options
options = [
    "NOTA",
    "Candidate 1",
    "Candidate 2",
    "Candidate 3",
    "Candidate 4"
]
```

```
# datatype of menu text
clicked = StringVar()

# initial menu text
clicked.set( "NOTA" )

# Create Dropdown menu
drop = OptionMenu( window , clicked , *options )
drop.config( bg="yellow"  ,fg="red"  ,width=30  ,height=2, activebackground = "yellow",
font=('times', 15, 'bold'))
drop.pack()
drop.place(x=700, y=505)


txt2 = tk.Entry(window,width=20  ,bg="yellow"  ,fg="red",font=('times', 15, ' bold ')  )
txt2.place(x=700, y=275)

lbl3 = tk.Label(window, text=" Current Status
",width=20  ,fg="red"  ,bg="yellow"  ,height=2 ,font=('times', 15, ' bold'))
lbl3.place(x=50, y=200)

message = tk.Label(window, text="" ,bg="yellow"  ,fg="red"  ,width=25  ,height=6,
activebackground = "yellow" ,font=('times', 15, ' bold '))
message.place(x=20, y=300)

note = tk.Label(window, text="NOTE \n1. Please click on the verify Email \nbutton to get
OTP\n2. Please Enter your name without\nspaces and any puctuations\nand special
characters" ,bg="yellow"  ,fg="red"  ,width=25  ,height=6, activebackground =
"yellow" ,font=('times', 15, ' bold '))
note.place(x=20, y=500)

lbl3 = tk.Label(window, text="Poll vote :
",width=20  ,fg="red"  ,bg="yellow"  ,height=2 ,font=('times', 15, ' bold'))
lbl3.place(x=400, y=510)
```

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

def clear2():
    txt2.delete(0, 'end')
    res = ""
    message.configure(text= res)
def clear3():
    txte.delete(0, 'end')
    res = ""
    message.configure(text= res)



def clear4():
    txtev.delete(0, 'end')
    res = ""
    message.configure(text= res)

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

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

```
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)]
    #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 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.append(imageNp)
        Ids.append(Id)
    return faces,Ids
def send():
    try:

        s = smtplib.SMTP("smtp.gmail.com" , 587)  # 587 is a port number
        s.starttls()
        s.login("**************" , "**********")
        global otp
        otp=random.randint(1000, 9999)
        otp = str(otp)
        s.sendmail("**************" , txte.get() , otp)
        messagebox.showinfo("Send OTP via Email", f"OTP sent to {txte.get()}")
        s.quit()
```

```
 except:
        messagebox.showinfo("Send OTP via Email", "Please enter the valid email address OR
check an internet connection")

def TrackImages():
    Vote=(clicked.get())
    Id=(txt.get())
    name=(txt2.get())

   x = re.findall("[a-z]{3}[0-9]{7}", Id)
    global Idalpha,IdwithoutAlpha
    Idalpha=Id[:3]
    Id=Id[3:]
    IdwithoutAlpha=Id
    if(len(x)==1 and name.isalpha() and len(txte.get())>0 and len(txtev.get())>0):

        recognizer = cv2.face.LBPHFaceRecognizer_create()#cv2.createLBPHFaceRecognizer()
        recognizer.read("trainingImageLabel/Trainner.yml")
        harcascadePath = "haarcascade_frontalface_default.xml"
        faceCascade = cv2.CascadeClassifier(harcascadePath)
        df=pd.read_csv("VoterDetails\\VoterDetails.csv")
        global MailFromDetails, IdalphaFromDetails
        MailFromDetails=df.loc[df['Id'] == int(IdwithoutAlpha)]['mail'].values
        IdalphaFromDetails=df.loc[df['Id'] == int(IdwithoutAlpha)]['code'].values

   cam = cv2.VideoCapture(0)
        df['Vote']=Vote
        font = cv2.FONT_HERSHEY_SIMPLEX
        col_names =  ['Id','Name','email','Vote','Date','Time','code']
        votings = pd.DataFrame(columns = col_names)
        aa=''
        Id=0
        #messagebox.showinfo("error",f"Invalid credentials {txte.get()} {MailFromDetails[0]}
{type(txte.get())} {type(MailFromDetails)}")
```

```
while True:
        _ret, im =cam.read()
        gray=cv2.cvtColor(im,cv2.COLOR_BGR2GRAY)
        faces=faceCascade.detectMultiScale(gray, 1.2,5)
        for(x,y,w,h) in faces:
           cv2.rectangle(im,(x,y),(x+w,y+h),(225,0,0),2)
           Id, conf = recognizer.predict(gray[y:y+h,x:x+w])



if(conf < 50):
                ts = time.time()
                global bb,date,timeStamp
                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
                bb=df.loc[df['Id'] == Id]['Vote'].values
                tt=aa
            else:
                Id='Unknown'
                tt=str(Id)
            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, 1,(255,255,255),2)


        cv2.imshow('im',im)
        if (cv2.waitKey(1)==ord('q') ):
           break
      if aa[0]==txt2.get() and Id==int(IdwithoutAlpha) and otp==txtev.get() and
MailFromDetails[0]==txte.get() and IdalphaFromDetails[0]==Idalpha:
        votings.loc[len(votings)] =
[IdwithoutAlpha,txt2.get(),txte.get(),bb,date,timeStamp,Idalpha]
```

```
votings= votings.drop_duplicates(subset=['Id'],keep='first')
        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="Votes\\Votes.csv"
        df=pd.read_csv(fileName)
        flag=1
        CheckVote=df['Id'].values.tolist()


for i in CheckVote:
            if i==int(IdwithoutAlpha):
                flag=0
        if flag==1:
            vv=[IdwithoutAlpha,txt2.get(),txte.get(),bb,date,timeStamp,Idalpha]
            with open(fileName,'a+') as csvFile:
                writer = csv.writer(csvFile)
                writer.writerow(vv)
            csvFile.close()
            #votings.to_csv(fileName,index=False)
            cam.release()
            cv2.destroyAllWindows()
            res=votings
            messagebox.showinfo("sucess",f" Thank You for using your valuable vote")
        else:
            messagebox.showinfo("failure",f" You have already voted")
      else:

        messagebox.showinfo("error",f"Invalid credentials {aa[0]} {txt2.get()} {Id}
{int(IdwithoutAlpha)} {otp} {txtev.get()} {MailFromDetails[0]} {txte.get()} {Idalpha}
{IdalphaFromDetails[0]}")
        #{aa[0]} {txt2.get()} {Id} {int(IdwithoutAlpha)} {otp} {txtev.get()}
{MailFromDetails[0]} {txte.get()} {Idalpha} {IdalphaFromDetails[0]}
```

```
else:
        if(len(x)!=1):
            res = "Enter correct Id with 3 alpahabets\n and 7 numbers"
            message.configure(text= res)
        if(name.isalpha() != True):
            res = "Enter Alphabetical Name"
            message.configure(text= res)
        if(len(txte.get())==0):
            res = "Enter Email ID"
            message.configure(text= res)
        if(len(txtev.get())==0):


    res = "Enter OTP sent to your Email ID"
            message.configure(text= res)

sendMail = tk.Button(window, text="Email verification",
command=send  ,fg="red"  ,bg="yellow"  ,width=15  ,height=2, activebackground =
"Red" ,font=('times', 15, ' bold '))

sendMail .place(x=400, y=600)

  clearButton = tk.Button(window, text="Clear",
command=clear  ,fg="red"  ,bg="yellow"  ,width=20  ,height=2 ,activebackground =
"Red" ,font=('times', 15, ' bold '))
clearButton.place(x=950, y=180)
clearButton2 = tk.Button(window, text="Clear",
command=clear2  ,fg="red"  ,bg="yellow"  ,width=20  ,height=2, activebackground =
"Red" ,font=('times', 15, ' bold '))

clearButton2.place(x=950, y=260)
clearButton3 = tk.Button(window, text="Clear",
command=clear3  ,fg="red"  ,bg="yellow"  ,width=20  ,height=2 ,activebackground =
"Red" ,font=('times', 15, ' bold '))
```

```
clearButton3.place(x=950, y=340)
clearButton4 = tk.Button(window, text="Clear",
command=clear4  ,fg="red"  ,bg="yellow"  ,width=20  ,height=2, activebackground =
"Red" ,font=('times', 15, ' bold '))
clearButton4.place(x=950, y=420)

lby = tk.Label(window, text="=>",width=2  ,height=1  ,fg="black"   ,font=('times', 25, ' bold
') )
lby.place(x=625, y=600)

trackImg = tk.Button(window, text="Face verification",
command=TrackImages  ,fg="red"  ,bg="yellow"  ,width=15  ,height=2, activebackground =
"Red" ,font=('times', 15, ' bold '))


trackImg.place(x=700, y=600)
quitWindow = tk.Button(window, text="Quit",
command=window.destroy  ,fg="red"  ,bg="yellow"  ,width=15  ,height=2, activebackground
= "Red" ,font=('times', 15, ' bold '))
quitWindow.place(x=1000, y=600)

window.mainloop()
```

Execution

```
from tkinter import *
import tkinter as tk
from PIL import ImageTk,Image

ws = Tk()
width= ws.winfo_screenwidth()
height= ws.winfo_screenheight()
ws.geometry("%dx%d" % (width, height))
ws.title('Smart Voting System')



f = ("Times bold", 14)
message = tk.Label(ws, text="Smart Voting
System" ,bg="#000080"  ,fg="white"  ,width=50  ,height=3,font=('times', 30, 'italic bold
underline'))
message.place(x=100, y=20)
lbe = tk.Label(ws, text="+",width=5  ,height=1  ,fg="black"   ,font=('times', 35, ' bold ') )
lbe.place(x=150, y=425)
lb2 = tk.Label(ws, text="=",width=5  ,height=1  ,fg="black"   ,font=('times', 35, ' bold ') )
lb2.place(x=400, y=425)

image1 = Image.open("faceRecog.jpg")
face = ImageTk.PhotoImage(image1)
label1 = tk.Label(image=face)
label1.image = face
label1.place(x=75, y=200)
image2 = Image.open("vote.jpg")
vote = ImageTk.PhotoImage(image2)
label2 = tk.Label(image=vote)
label2.image = vote
label2.place(x=75, y=500)
image3 = Image.open("election.jpg")
```

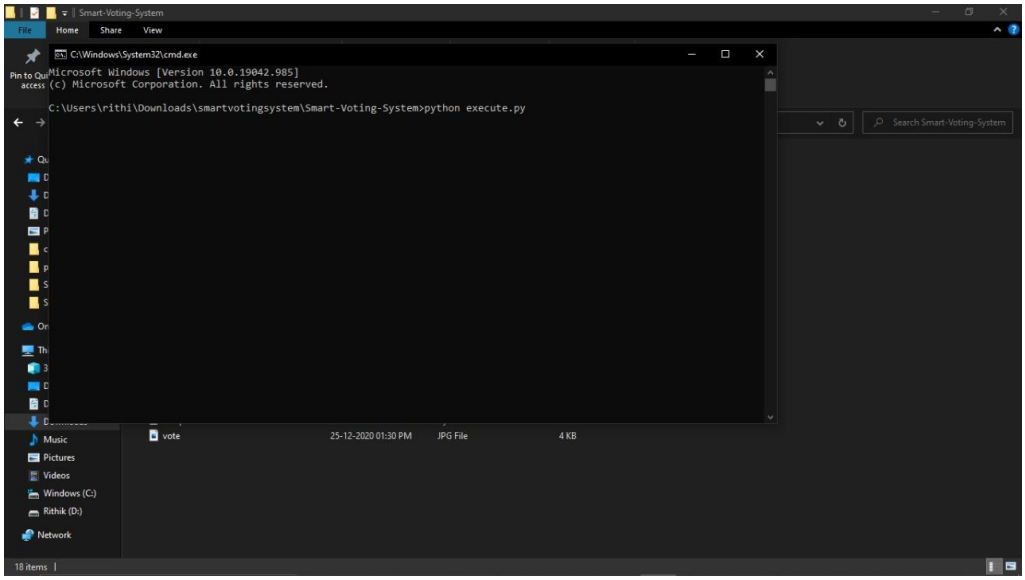
```
election = ImageTk.PhotoImage(image3)
label3 = tk.Label(image=election)
label3.image = election
label3.place(x=550, y=200)

 def loginPage():
    import login

def registerPage():
    import registerc

loginButton = tk.Button(ws, text="Poll Vote",
command=loginPage  ,fg="red"  ,bg="yellow"  ,width=20  ,height=2 ,activebackground =
"Red" ,font=('times', 15, ' bold '))
loginButton.place(x=950, y=250)
registerButton = tk.Button(ws, text="Register Vote",
command=registerPage  ,fg="red"  ,bg="yellow"  ,width=20  ,height=2 ,activebackground =
"Red" ,font=('times', 15, ' bold '))
registerButton.place(x=950, y=500)

ws.mainloop()
```



Execution of the code