Face Dictionary

Group Members: Abhinav Parkash Sharma, Divyansh Chowdhary, Shantanu Gupta

Introduction –

Human face detection by computer systems has become a major field of interest. Face detection algorithms are used in a wide range of applications, such as security control, video retrieving, human computer interface, face recognitions and image database management. Face recognition has wide range of possibilities in the near future. Automation and reducing man force is the main goal to achieve using the power of computing.

Technical details –

Python2.7.1 environment along with OpenCV to implement image processing was used in developing the face recognition application. The vast inbuilt and external library support for python helps us in built better and faster algorithms for face recognition.

The application first stores a person's features and facial expressions in a dataset on the local disk. Then he enters his details like Name, unique id (register no) , age and other details which get stored in sqliteStudio database which is supported by python. Then we train our algorithm to recognize the a person when a hardware camera is opened to analyse the faces in real time. This application can recognize multiple faces at a time.

Business Approach –

The product so developed will be pitched to national security agencies for cheap border control as well as to home security industry for improving house security with very minimum cost. Next the code will ported to android, windows phone, IOS using iron python in the .Net framework.

The app will be launched on the all the app hosting websites free of cost for a period of an year or so; So as to implement some deep learning and reinforced learning algorithm. The code will run on a centralized server and the app will just behave as clients to search the person so as to maintain security integrity.

Applications –

Can be used in border security to continuously monitor for intruders across the border which will save time for the Indian Army to locate terrorist attacks.

Can be used to authorize persons solely on the basis of facial features, which will eliminate the need of carrying any kind ID proof. Like for instance in a train the ticket checker would just have to carry a camera to authenticate if the passengers have issued a proper ticket or not.

Can be integrated with cab applications to recognize if the driver is registered one or not. This will help the women of India to feel safe in their own country without being cheated on.

Can be used in places where restricted access is allowed to allow those authorized persons only to enter. Like in a locker room of a bank.

It can also be installed outside buildings to keep a count of the flow residents and visitors.

Social Values –

Privacy of people will not be leaked and the access of the application and database can be given to only authorized person or a group.

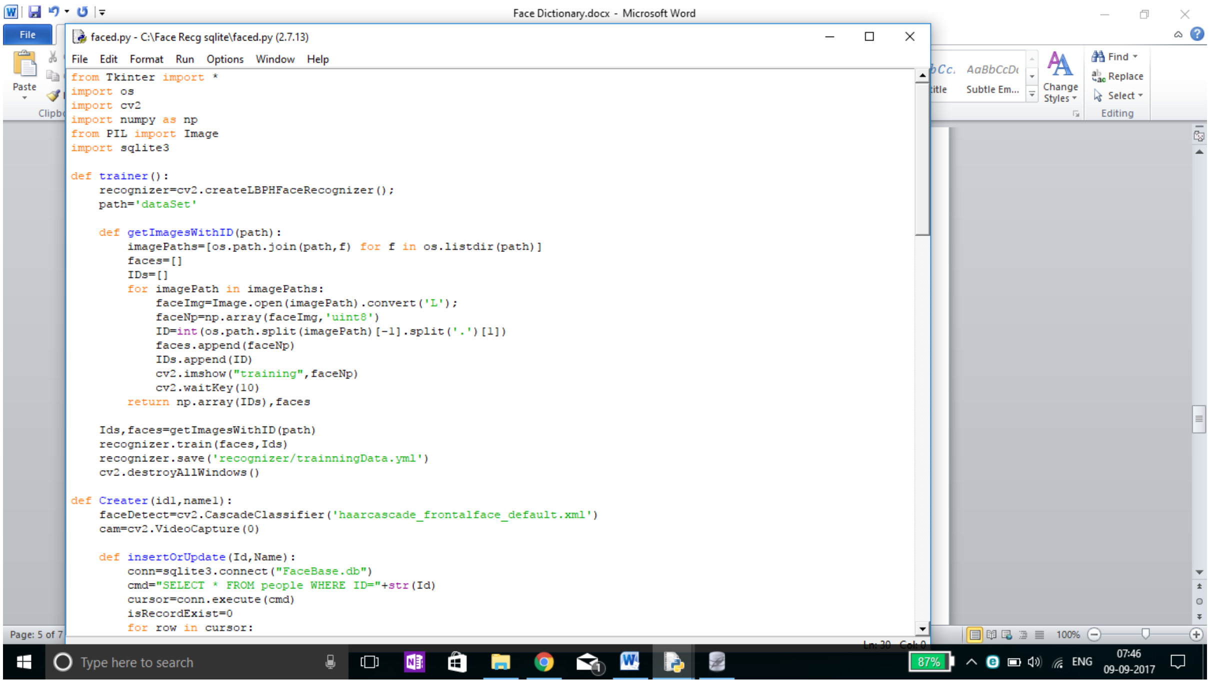
CODE->

```
from Tkinter import *
import os
import cv2
import numpy as np
from PIL import Image
import sqlite3


def trainer():
recognizer=cv2.createLBPHFaceRecognizer();
path='dataSet'


def getImagesWithID(path):
imagePaths=[os.path.join(path,f) for f in os.listdir(path)]
faces=[]
IDs=[]
for imagePath in imagePaths:
faceImg=Image.open(imagePath).convert('L');
faceNp=np.array(faceImg,'uint8')
ID=int(os.path.split(imagePath)[-1].split('.')[1])
faces.append(faceNp)
IDs.append(ID)
cv2.imshow("training",faceNp)
cv2.waitKey(10)
return np.array(IDs),faces


Ids,faces=getImagesWithID(path)
recognizer.train(faces,Ids)
recognizer.save('recognizer/trainningData.yml')
cv2.destroyAllWindows()


def Creater(id1,name1):
```

```
faceDetect=cv2.CascadeClassifier('haarcascade_frontalface_default.xml')
cam=cv2.VideoCapture(0)


def insertOrUpdate(Id,Name):
conn=sqlite3.connect("FaceBase.db")
cmd="SELECT * FROM people WHERE ID="+str(Id)
cursor=conn.execute(cmd)
isRecordExist=0
for row in cursor:
isRecordExist=1
if(isRecordExist==1):
cmd="UPDATE people SET Name='"+str(Name)+"'WHERE ID="+str(Id)
else:
cmd="INSERT INTO people(ID,Name) Values("+str(Id)+",'"+str(Name)+"')"
conn.execute(cmd)
conn.commit()
conn.close()


insertOrUpdate(id1,name1)
sampleNum=0
while(True):
ret,img=cam.read()
gray=cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)
faces=faceDetect.detectMultiScale(gray,1.3,5)
for(x,y,w,h) in faces:
sampleNum=sampleNum+1


cv2.imwrite("dataSet/User."+str(id1)+"."+str(sampleNum)+".jpg",gray[y:y+h,x:x+w])


cv2.rectangle(img,(x,y),(x+w,y+h),(0,0,255),2)
cv2.waitKey(100)
cv2.imshow("Face",img)
cv2.waitKey(1)
if(sampleNum>20):
print("done")
break
cam.release()
cv2.destroyAllWindows()
trainer()


def detect():
faceDetect=cv2.CascadeClassifier('haarcascade_frontalface_default.xml');
```

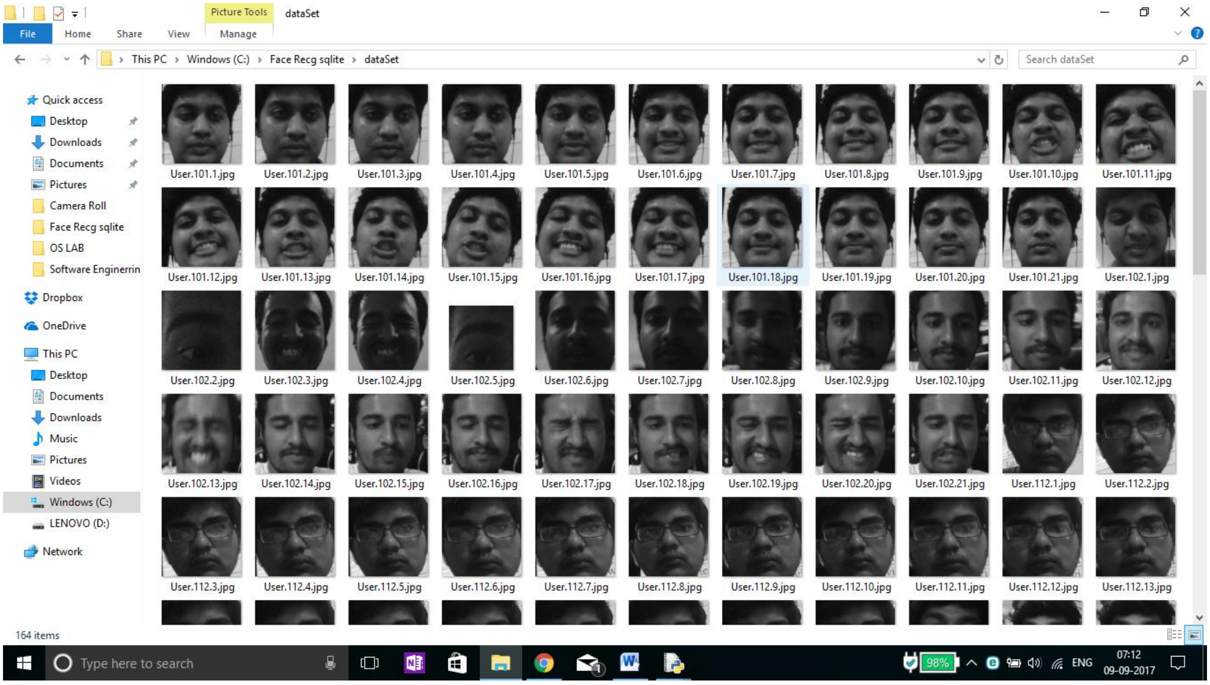
```
cam=cv2.VideoCapture(0);
rec=cv2.createLBPHFaceRecognizer();
rec.load("recognizer\\trainningData.yml")
id=0
path='dataSet'


def getProfile(id):
conn=sqlite3.connect("FaceBase.db")
cmd="SELECT * FROM people WHERE ID="+str(id)
cursor=conn.execute(cmd)
profile=None
for row in cursor:
profile=row
conn.close()
return profile



font=cv2.cv.InitFont(cv2.cv.CV_FONT_HERSHEY_SIMPLEX,1,1,0,1,1)
while(True):
ret,img=cam.read();
gray=cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)
##faces=faceCascade.detectMultiScale(gray,scaleFactor=1.2,minNeighbors=5,minSize=(100,100),flags)
faces=faceDetect.detectMultiScale(gray,1.3,5);
for(x,y,w,h) in faces:
cv2.rectangle(img,(x,y),(x+w,y+h),(255,0,0),2)
id,conf=rec.predict(gray[y:y+h,x:x+w])
##if(id==101):
##  id="Shantanu"
profile=getProfile(id)
if(profile!=None):
    cv2.cv.PutText(cv2.cv.fromarray(img),str(profile[1]),(x,y+h+30),font,255);
    cv2.cv.PutText(cv2.cv.fromarray(img),str(profile[2]),(x,y+h+60),font,255);
    cv2.cv.PutText(cv2.cv.fromarray(img),str(profile[3]),(x,y+h+90),font,255);
    ##cv2.cv.PutText(cv2.cv.fromarray(img),str(profile[4]),(x,y+h+120),font,255);
cv2.imshow("Face",img);
if(cv2.waitKey(1)==ord('q')):
break;
cam.release()
cv2.destroyAllWindows()


top=Tk()
top.title("Face Detector")
L1 = Label(top, text="Name")
L1.pack(side = LEFT)
E1 = Entry(top,bd=5)
```

```
E1.pack()


L2 = Label(top, text="Id")
L2.pack( side = LEFT)
E2 = Entry(top, bd =5)
E2.pack()
def store():
input_name=E1.get()
input_id=E2.get()
Creater(input_id,input_name)


def analyse():


detect();
StoreButton = Button(top,text="Store",command=store)
AnalyseButton = Button(top,text="Analyse",command=analyse)


StoreButton.pack()
AnalyseButton.pack()


top.mainloop()
```

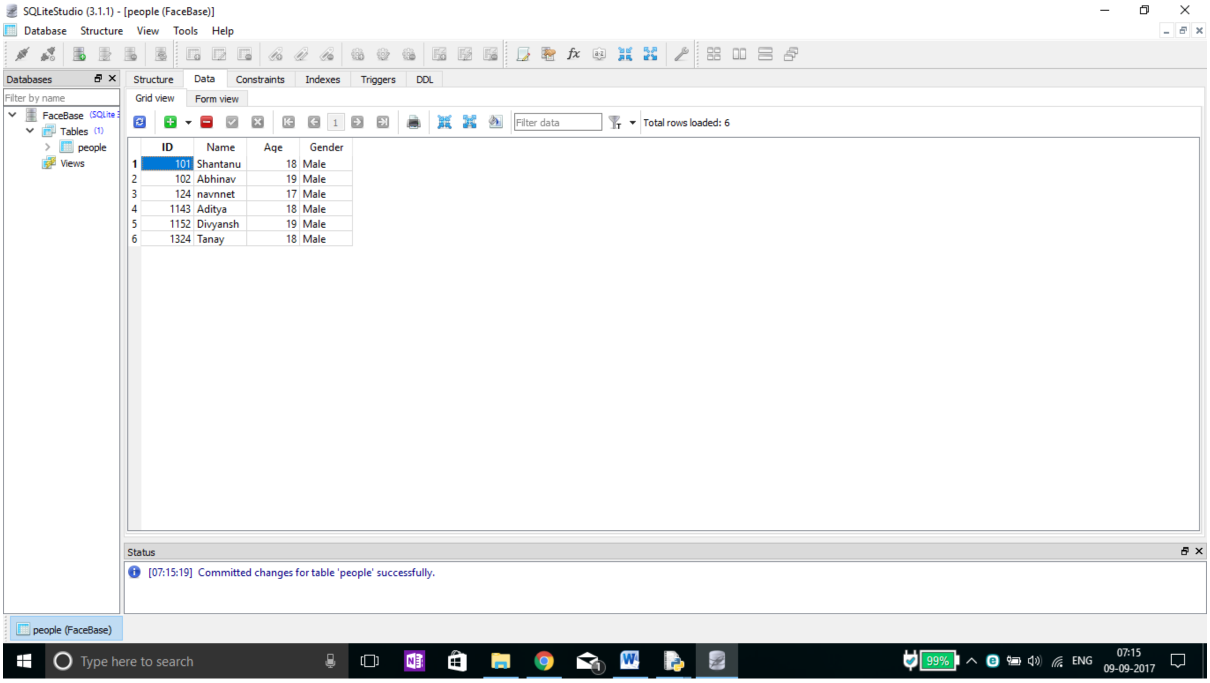


```
from Tkinter import *
import os
import cv2
import numpy as np
from PIL import Image
import sqlite3

def trainer():
    recognizer=cv2.createLBPHFaceRecognizer();
    path='dataSet'

    def getImagesWithID(path):
        imagePaths=[os.path.join(path,f) for f in os.listdir(path)]
        faces=[]
        IDs=[]
        for imagePath in imagePaths:
            faceImg=Image.open(imagePath).convert('L');
            faceNp=np.array(faceImg,'uint8')
            ID=int(os.path.split(imagePath)[-1].split('.')[1])
            faces.append(faceNp)
            IDs.append(ID)
            cv2.imshow("training",faceNp)
            cv2.waitKey(10)
        return np.array(IDs),faces

    Ids,faces=getImagesWithID(path)
    recognizer.train(faces,Ids)
    recognizer.save('recognizer/trainningData.yml')
    cv2.destroyAllWindows()

def Creater(id1,name1):
    faceDetect=cv2.CascadeClassifier('haarcascade_frontalface_default.xml')
    cam=cv2.VideoCapture(0)

    def insertOrUpdate(Id,Name):
        conn=sqlite3.connect("FaceBase.db")
        cmd="SELECT * FROM people WHERE ID="+str(Id)
        cursor=conn.execute(cmd)
        isRecordExist=0
        for row in cursor:
```

Datasets –



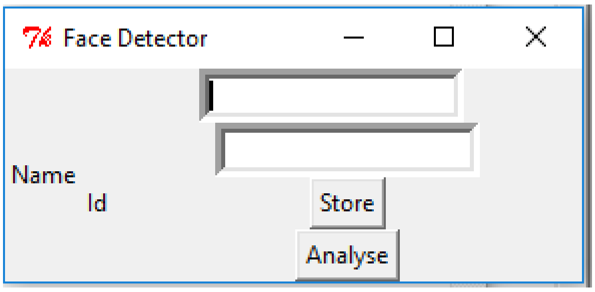
Database –



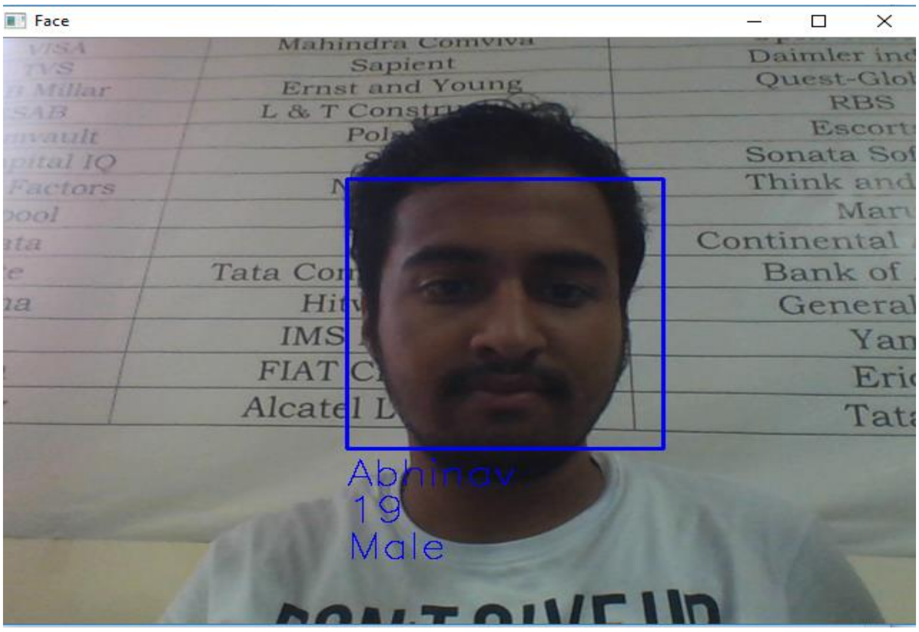
	ID	Name	Age	Gender
1	101	Shantanu	18	Male
2	102	Abhinav	19	Male
3	124	navnnet	17	Male
4	1143	Aditya	18	Male
5	1152	Divyansh	19	Male
6	1324	Tanay	18	Male

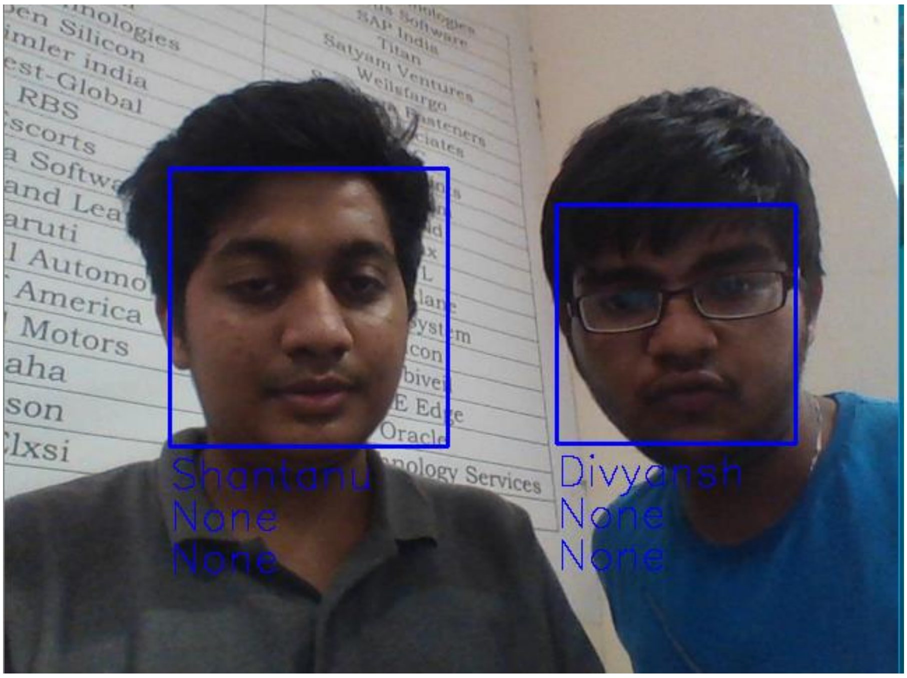
```
[07:15:19]  Committed changes for table 'people' successfully.
```

GUI –


Face Detector
Name
Id
Store
Analyse

Analyser


Face
Abhinav
19
Male


Shantanu
None
None
Divyansh
None
None