SOURCE CODE

fdetect1.py

```
import sys
import os
from face_detect import *
from PyQt5 import QtWidgets, QtGui, QtCore

class MyForm(QtWidgets.QMainWindow):
  def___init_(self,parent=None):
     QtWidgets.QWidget._init_(self,parent)
     self.ui = Ui_MainWindow()
     self.ui.setupUi(self)
     self.ui.pushButton.clicked.connect(self.stdetails)
     self.ui.pushButton_2.clicked.connect(self.analyse)
     self.ui.pushButton_3.clicked.connect(self.snapshot)
     self.ui.pushButton_4.clicked.connect(self.pdtls)

  def stdetails(self):
    os.system("python person1.py")

  def analyse(self):
    os.system("python identify1.py")

  def snapshot(self):
    os.system("python snpsht1.py")

  def pdtls(self):
    os.system("python prisonerdtls1.py")

if___name___== "_main_":
    app = QtWidgets.QApplication(sys.argv)
    myapp = MyForm()
    myapp.show()
    sys.exit(app.exec_())
```

identify1.py

```
import sys
import os
#import xlsxwriter
from identify import *
from reportlab import platypus
from reportlab.lib.styles import ParagraphStyle as PS
from reportlab.platypus import SimpleDocTemplate
import cv2
import numpy as np
import smtplib
from PyQt5 import QtWidgets, QtGui, QtCore

import sqlite3
con = sqlite3.connect('wbcam1')

class MyForm(QtWidgets.QMainWindow):
  def___init_(self,parent=None):
     QtWidgets.QWidget._init__(self,parent)
     self.ui = Ui_MainWindow()
     self.ui.setupUi(self)
     self.ui.pushButton.clicked.connect(self.analyse)
     self.ui.pushButton_2.clicked.connect(self.genreport)

  def analyse(self):

     with con:

        cur = con.cursor()
        image2 = str(self.ui.lineEdit_4.text())
        large_image = cv2.imread(image2)
        cur.execute('SELECT * FROM persons');
        result1 = cur.fetchall()
        for row in result1:
          image1 = row[1]
   # Read the images from the file
          small_image = cv2.imread(image1)
          if not cv2.matchTemplate(small_image, large_image,
cv2.TM_SQDIFF_NORMED):
             print('not identified')
          else:
             result = cv2.matchTemplate(small_image, large_image,
cv2.TM_SQDIFF_NORMED)
```

```
   # We want the minimum squared difference
          mn,_,mnLoc,_ = cv2.minMaxLoc(result)

   # Draw the rectangle:
   # Extract the coordinates of our best match
          MPx,MPy = mnLoc

   # Step 2: Get the size of the template. This is the same size as the match.
          trows,tcols = small_image.shape[:2]

   # Step 3: Draw the rectangle on large_image
          cv2.rectangle(large_image,
(MPx,MPy),(MPx+tcols,MPy+trows),(0,0,255),2)

   # Display the original image with the rectangle around the match.
          cv2.imshow('output',large_image)
          print('{0} Identified'.format(row[0]))
   # The image is only displayed if we call this
          cv2.waitKey(2000)

  def genreport(self):

    with con:

      cur = con.cursor()
      image2 = str(self.ui.lineEdit_4.text())
      large_image = cv2.imread(image2)
      cur.execute('SELECT * FROM persons');
      result1 = cur.fetchall()
      cnt1 = 0
      result2 = str(" People identified in the given image are:")
      for row in result1:
        image1 = row[1]
   # Read the images from the file
        small_image = cv2.imread(image1)

        result = cv2.matchTemplate(small_image, large_image,
cv2.TM_SQDIFF_NORMED)

   # We want the minimum squared difference
        mn,_,mnLoc,_ = cv2.minMaxLoc(result)
   # Draw the rectangle:
   # Extract the coordinates of our best match
        MPx,MPy = mnLoc
   # Step 2: Get the size of the template. This is the same size as the match.
        trows,tcols = small_image.shape[:2]

   # Step 3: Draw the rectangle on large_image
        cv2.rectangle(large_image,
(MPx,MPy),(MPx+tcols,MPy+trows),(0,0,255),2)
        cnt1 = cnt1 +1

   # Display the original image with the rectangle around the match.
        cv2.imshow('output',large_image)
        print('{0} Identified'.format(row[0]))
   # The image is only displayed if we call this
        cv2.waitKey(2000)
        result2 = result2 + ', ' + row[0]
      items = []
      result2 = result2 + "<br/>" + "No.of people identified: " + str(cnt1)
      items.append(platypus.Paragraph(result2,PS('body')))
      doc = SimpleDocTemplate('found1.pdf')
      doc.multiBuild(items)

if__name__== "_main_":
    app = QtWidgets.QApplication(sys.argv)
    myapp = MyForm()
    myapp.show()
    sys.exit(app.exec_())
```

person1.py

```
import sys
import os
from person import *
from PyQt5 import QtWidgets, QtGui, QtCore

import sqlite3
con = sqlite3.connect('wbcam1')

class MyForm(QtWidgets.QMainWindow):
  def__init_(self,parent=None):
     QtWidgets.QWidget. init_(self,parent)
     self.ui = Ui_MainWindow()
     self.ui.setupUi(self)
     self.ui.pushButton.clicked.connect(self.insertvalues)
  def insertvalues(self):

     with con:

        cur = con.cursor()
        s4 = str(self.ui.lineEdit_4.text())
        s5 = str(self.ui.lineEdit_5.text())
        #picdata = open(s5, 'rb').read()
 #s6 = str(self.ui.lineEdit_6.text())
 #s7 = str(self.ui.lineEdit_7.text())
        cur.execute('INSERT INTO persons VALUES(?,?)',(s4,s5))
        con.commit()

#  def imagedetails(self):
#    os.system("python disease1.py")

if__name__== "_main_":
    app = QtWidgets.QApplication(sys.argv)
    myapp = MyForm()
    myapp.show()
    sys.exit(app.exec_())
```

<u>prisonerdtls1.py</u>

```
import          sys
from prisoner import *
from PyQt5 import QtWidgets, QtGui, QtCore

import sqlite3
con = sqlite3.connect('wbcam1')

class MyForm(QtWidgets.QMainWindow):
  def___init_(self,parent=None):
     QtWidgets.QWidget. init (self,parent)
     self.ui = Ui_MainWindow()
     self.ui.setupUi(self)
     self.ui.pushButton.clicked.connect(self.insertvalues)



  def insertvalues(self):

     with con:



        cur = con.cursor()
        aadhar = str(self.ui.lineEdit.text())
        pid = str(self.ui.lineEdit_3.text())
        pname = str(self.ui.lineEdit_4.text())
        addr1 = str(self.ui.lineEdit_5.text())
        addr2 = str(self.ui.lineEdit_6.text())
        mobile = str(self.ui.lineEdit_2.text())
        cur.execute('INSERT INTO prisoner
VALUES(?,?,?,?,?,?)',(pid,pname,addr1,addr2,aadhar,mobile))
        con.commit()

if___name___== "_main_":
    app = QtWidgets.QApplication(sys.argv)
    myapp = MyForm()
    myapp.show()
    sys.exit(app.exec_())
```

snpsht1.py

```
import cv2
cam = cv2.VideoCapture(0)
cv2.namedWindow("test")
img_counter = 0
while True:
    ret, frame = cam.read()
    cv2.imshow("test", frame)
    if not ret:
        break
    k = cv2.waitKey(1)
    if k%256 == 27:
        # ESC pressed
        print("Escape hit, closing...")
        break
    elif k%256 == 32:
        # SPACE pressed
        img_name = "frame{}.png".format(img_counter)
        cv2.imwrite(img_name, frame)
        print("{} written!".format(img_name))
        img_counter += 1
cam.release()
cv2.destroyAllWindows()
```