VISVESVARAYA TECHNOLOGICAL UNIVERSITY, BELGAUM 590014

Internet of Things Project Report on

"Security Surveillance System"

By

ASHISH CHANDER (1BM17CS148)
SHREYANK JAIN(1BM17CS097)
RAVI PRAKASH(1BM17CS074)

Under the Guidance of

Dr.K.Panimozhi
Assistant Professor, Department of CSE
BMS College of Engineering

IoT Application Development carried out at

DepartmentofComputer Scienceand Engineering
BMS Collegeof Engineering
(Autonomous college under VTU)
P.O. Box No.: 1908, Bull Temple Road, Bangalore-560 019

2019-2020

BMS COLLEGE OF ENGINEERING

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that the Internet of Things project titled "**Security Surveillance System**" has been carried out by **ASHISH CHANDER(1BM17CS148), SHREYANK JAIN(1BM17CS097) and RAVI PRAKASH N(1BM17CS074)** during the academic year 2019-2020.

Signature of the guide
Dr.K.Panimozhi
Assistant Professor
Department of Computer Science and Engineering
BMS College of Engineering, Bangalore

Examiners

Name	**Signature**
1.	
2.	

BMS COLLEGE OF ENGINEERING

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

DECALARATION

We, ASHISH CHANDER(1BM17CS148), SHREYANK JAIN(1BM17CS097) and RAVI PRAKASH N(1BM17CS074) students of 5th Semester, B.E, Department of Computer Science and Engineering, BMS College of Engineering, Bangalore, hereby declare that, this IoT Application development work entitled "Security Surveillance System" has been carried out by us under the guidance of **Dr.K.Panimozhi**, Assistant Professor, Department of CSE, BMS College of Engineering, Bangalore during the academic semester Aug-Dec 2019

We also declare that to the best of our knowledge and belief, the development reported here is not from part of any other report by any other students.

Signature

ASHISH CHANDER (1BM17CS148)

SHREYANK JAIN(1BM17CS097)

RAVI PRAKASH N (1BM17CS074)

Objective of the project

Surveillance plays a major role in the field of security. CCTV cameras serves this purpose by providing live footage. The footages are stored and can be accessed as per requirement.

However, storage required is very high and searching for activities in a particular location at a specific time is very time consuming. One has to go through hours of CCTV footage to find what they are looking for.

Our Security surveillance system resolves these problems by optimizing memory usage and detecting criminals in real time and also alerts the security personnel if required. It also recognizes the criminals and stores the details for future reference.

Literature Survey

Sl.No	Name of the Project or Product (Existing)	Commercial or Non-Commercial	Features
1.	OpenCv FaceRecognition	Non-Commercial	Face recognition

Opencv(open source computer vision) is a library of programming functions mainly aimed at real-time computer vision, developed by Intel. The library is a cross-platform and free to use under open source license.

Drawbacks in older systems:

Each of the CCTV cameras of utmost 2mp cameras. Each of the cameras have a wired connection with a central systems where the footages are stored .The footages can be accessed by the computer/display screen to which the system is attached.

The main disadvantage of this system is insufficient storage space and very poor efficiency in searching the footages. Upon any criminal activities or to identify a criminal, the police have to look at all the footages of that day till they find it. This takes a lot of time and effort.

Solutions to the prevailing problem:

Security Surveillance System captures high quality photos every 5 seconds and wirelessly sends it to the controller node. The controller runs a facial recognition algorithm based on Opencv module to recognize if any criminals are present in the photo sent. The controller has a database where all the criminal details are stored. On recognizing any criminals ,the controller stores the date, time and location where the criminal was located.

The system makes it easier while searching the records to get the details of a particular criminal. The systems allows the user to search by date and location which provides a list of criminals and their primary details ,who were spotted in that location on that specific date. Hence it saves the time of going through hours of video footages.

The system displays the raw image captured and the process image. The quick search section allows the user to get the primary details of the criminal. Each time a criminal is spotted, the last known location, time and date of the sighting is updated .

Advantages:

1. **The Picamera captures high resolution images than most of the other CCTV cameras out there in the market.**
2. **The controller not only stores the images , but also recognizes if any criminals are there in the image.**
3. **Instead of storing a 24hr long videos ,we store pictures that are taken every 5 seconds ,and also delete duplicate images to save storage space.**
4. **Upon detecting a high level threat/spotted a wanted criminal ,it quickly alerts the security personnel at the desk and also alerts the police by means of text message followed by a voice call.**
5. **The user friendly GUI allows easy searching of criminal details and also allows date and location specific searching of criminals. It also stores the timestamp at which the criminal was spotted in the location.**

Hardware and Software Requirements

Hardware Requirements:

1. **Raspberry pi model 2B**
2. **PiCamera**
3. **Arduino Uno board**
4. **GSM Module**
5. **Wifi Reciever Module**

Cost Estimation:

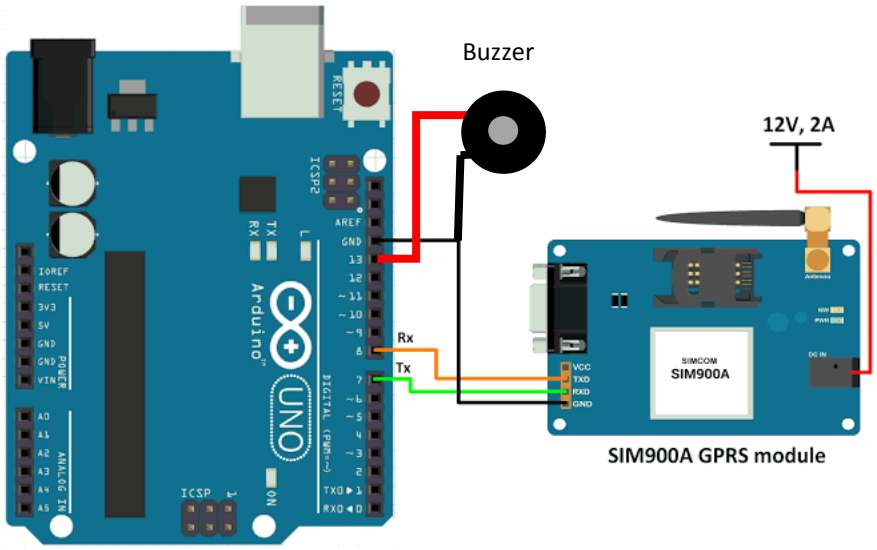
1.	**Raspberry Pi 2B**	**: Rs.3000/-**
2.	**PiCamera**	**: Rs.400/-**
3.	**Arduino Uno**	**: Rs.400/-**
4.	**GSM Module**	**: Rs.1000/-**
5.	**Wifi Reciever Module**	**: Rs.200/-**
	Total Cost of the Project	**:Rs.5000/-**

Software Requirements:

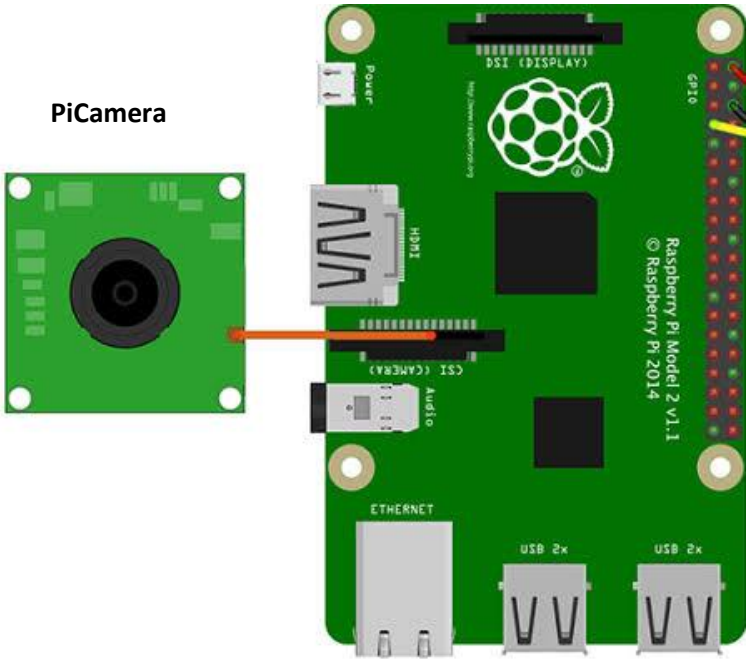
1. **Rasberian OS**
2. **Arduino IDE**
3. **Python IDLE**
4. **OpenCv module**
5. **Serial module**

Design and Circuit Diagram:

Arduino Uno and GSM Module:



Raspberry Pi and PiCamera



Implementation

1. **Make the required connections as per the above diagrams**
2. **Boot the Raspberry Pi**
3. **Run the Server program on the controller node**
4. **Run the Client program on Raspberry Pi**
5. **Upload and run the Alert code on Arduino Uno**
6. **Run the Facial Recognition program on the controller node**
7. **Run the GUI/user interaction program on the controller node**

Source codes

1.GSM Arduino code

```
#include<SoftwareSerial.h>

SoftwareSerial cell(4,5);

String message;

void setup() {

  cell.begin(9600);

  delay(500);

  Serial.begin(9600);

}

void message1(String mess)

{

   cell.println("messaging");

  cell.println("AT+CMGF=1");

  delay(1000);

  cell.println("AT+CMGS=\"+916360658384\"\r");

  delay(1000);

  cell.println(mess);

  delay(100);

  cell.println((char)26);

}

void call()

{

  cell.println("ATD+916360658384;");

  delay(10000);

  cell.println("ATH");
```

```
}
void loop() {
if(Serial.available()){
  String message=Serial.readString();
  call();
  delay(2000);
  message1(message);
  delay(2000);
}
}
```

2.Code for client and server sockets:

Client socket:

```
import cv2
import io
import socket
import struct
import time
import pickle
import zlib
import time

client_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
client_socket.connect(('192.168.0.103', 8000))
connection = client_socket.makefile('wb')

cam = cv2.VideoCapture(0)

cam.set(3,900);
```

```
cam.set(4,720);


img_counter = 0


encode_param = [int(cv2.IMWRITE_JPEG_QUALITY), 90]


while True:
    ret, frame = cam.read()
    result, frame = cv2.imencode('.jpg', frame, encode_param)
    data = pickle.dumps(frame, 0)
    size = len(data)



    print("{}: {}".format(img_counter, size))
    client_socket.sendall(struct.pack(">L", size) + data)
    img_counter += 1
    time.sleep(10)
cam.release()
```

Server socket:

```
import socket
import sys
import cv2
import pickle
import numpy as np
import struct
import zlib


HOST='192.168.43.90'
```

```
PORT=8000

s=socket.socket(socket.AF_INET,socket.SOCK_STREAM)
print('Socket created')

s.bind((HOST,PORT))
print('Socket bind complete')
s.listen(10)
print('Socket now listening')
a=0
conn,addr=s.accept()

data = b""
payload_size = struct.calcsize(">L")
print("payload_size: {}".format(payload_size))
while True:
    a=a+1
    while len(data) < payload_size:
        print("Recv: {}".format(len(data)))
        data += conn.recv(4096)

    print("Done Recv: {}".format(len(data)))
    packed_msg_size = data[:payload_size]
    data = data[payload_size:]
    msg_size = struct.unpack(">L", packed_msg_size)[0]
    print("msg_size: {}".format(msg_size))
    while len(data) < msg_size:
        data += conn.recv(4096)
```

```
    frame_data = data[:msg_size]
    data = data[msg_size:]

    frame=pickle.loads(frame_data, fix_imports=True, encoding="bytes")
    frame = cv2.imdecode(frame, cv2.IMREAD_COLOR)
    cv2.imwrite("D:\\ASHSIH\\PROJECTS\\version2\\Ry\\ "+"r"+str(a)+".jpg",frame)
    cv2.waitKey(0)
```

Code for GUI:

```
import tkinter
import sqlite3
from tkinter import *
from PIL import ImageTk, Image
import os
import time
from tkinter import messagebox
import datetime
from datetime import datetime
import sys


top=None
tbox=None
filename=''
conn=sqlite3.connect('criminalDb.db')
c=conn.cursor()
```

```
def displaySpec(name1):
    sql='select * from criminal where name="'+str(name1)+'"'
    c.execute(sql )
    global record
    record=c.fetchone()
    return record


def clear_history():
    pass
def search_history(ent1,ent2):
    global top,tbox
    search_date=ent1
    search_loc=ent2
    conn=sqlite3.connect('criminalDb.db')
    c=conn.cursor()
    tup=(search_date,search_loc)
    sql='''select name,TLevel,LastKnownTime,offence from criminal WHERE LastknownDate=? AND
LastKnownLoc=?'''
    c.execute(sql,tup)
    rows=c.fetchall()
    col=len(rows[0])
    row=len(rows)
    #print(rows[0][0])
    #a=0;b=20;c=23;d=33
    xp = 60
    yp = 190
    l = Label(top,text="NAME").place(x=60,y=160, width=120, height=25)
    l = Label(top,text="THREAT LEVEL").place(x=185,y=160, width=120, height=25)
```

```
    l = Label(top,text="LAST SEEN").place(x=310,y=160, width=120, height=25)
    l = Label(top,text="COMITTED CRIMES").place(x=435,y=160, width=200, height=25)


    for i in range(row):
        yp+=30
        xp = 60
        for j in range(col):
            if j == 3:
                string=str(rows[i][j])
                l = Label(top,text=string)
                #l.configure(width=400, height=25)
                l.place(x=xp,y=yp, width=200, height=25)
            else:
                string=str(rows[i][j])
                l = Label(top,text=string)
                l.place(x=xp,y=yp,width=120, height=25)
                xp+=125
class display:
    def clear(self): #clear all field in section C
        self.text9.delete(0,END)
        self.text10.configure(state='normal')
        self.text10.delete(0.0,END)
        self.text10.configure(state='disabled')
        self.text6.configure(state='normal')
        self.text6.delete(0.0,END)
        self.text6.configure(state='disabled')
        self.text5.configure(state='normal')
        self.text5.delete(0.0,END)
```

```
        self.text5.configure(state='disabled')
        self.text2.configure(state='normal')
        self.text2.delete(0.0,END)
        self.text2.configure(state='disabled')
        self.text4.configure(state='normal')
        self.text4.delete(0.0,END)
        self.text4.configure(state='disabled')
        self.text3.configure(state='normal')
        self.text3.delete(0.0,END)
        self.text3.configure(state='disabled')


    def setcolor(self,lvl):
        if lvl==1:
            self.frameimage.configure(background='yellow')
        elif lvl== 2:
            self.frameimage.configure(background='yellow2')
        elif lvl== 3:
            self.frameimage.configure(background='tomato')
        elif lvl== 4:
            self.frameimage.configure(background='orange')
        elif lvl== 5:
            self.frameimage.configure(background='red')
        else:
            self.frameimage.configure(background='black')

    def clearimg(self):
        try:
```

```
            self.panel3.destroy()
        except:
            pass
        self.frameimage.configure(background='black')
        self.clear()

    def search(self):
        global filename
        name=str(self.text9.get())
        records=displaySpec(name)
        if records!=None:
            lvl=records[2]
            self.clear()
            filename=str(records[6])
            self.img3=Image.open(filename)
            self.img3=self.img3.resize((180,200),Image.ANTIALIAS)
            self.img3=ImageTk.PhotoImage(self.img3)
            self.panel3=Label(window,image=self.img3,width=180,height=200)
            self.panel3.place(x=905,y=165)
            self.setcolor(lvl)
            self.text10.configure(state='normal')
            self.text10.insert(0.0,"   "+records[0])
            self.text10.configure(state='disabled')

            self.text6.configure(state='normal')
            self.text6.insert(0.0,"   "+records[5])
            self.text6.configure(state='disabled')
            self.text5.configure(state='normal')
```

```
            self.text5.insert(0.0,"   "+records[4]+"  "+records[7])
            self.text5.configure(state='disabled')
            self.text2.configure(state='normal')
            self.text2.insert(0.0,"   "+records[1])
            self.text2.configure(state='disabled')
            self.text4.configure(state='normal')
            self.text4.insert(0.0,"   "+records[3])
            self.text4.configure(state='disabled')
            self.text3.configure(state='normal')
            self.text3.insert(0.0,"   "+str(records[2]))
            self.text3.configure(state='disabled')
        else:
            self.clearimg()
            messagebox.showinfo("invalid","Record Doesnt Exist!")
    def Next(self):
        # next image
        #if self.count==8:
            #sys.exit()
        if self.count<2:
            nextpic=self.count
            nextpic+=3
            img=Image.open("D:\\ASHSIH\\PROJECTS\\version2\\Ryy\\"+"r"+str(nextpic)+".jpg")
            img=img.resize((540,350),Image.ANTIALIAS)
            self.my_images1.append(ImageTk.PhotoImage(img))

        # change image
        self.canvas1.itemconfig(self.image_on_canvas1, image = self.my_images1[self.count])
        window.after(6000,self.Next2)
```

```
    def Next2(self):
        # next image
        #if self.count==8:
            #sys.exit()
        if self.count<2:
            nextpic=self.count
            nextpic+=3



            img=Image.open("Processed/"+"p"+str(nextpic)+".jpg")
            img=img.resize((540,350),Image.ANTIALIAS)
            self.my_images2.append(ImageTk.PhotoImage(img))
        # change image
        self.canvas2.itemconfig(self.image_on_canvas2, image = self.my_images2[self.count])
        self.count+=1
        window.after(5000,self.Next)
#-----------------------------------------------------------------
    def __init__(self,window):
         self.canvas1=Canvas(window,width=640,height=360)
         self.canvas1.configure(background='black')
         self.canvas1.place(x=70,y=10)
#-------------------------------------------------
         self.canvas2=Canvas(window,width=640,height=360)
         self.canvas2.configure(background='black')
         self.canvas2.place(x=70,y=385)
#-----------------------------------------------------------------
        # images
```

```
        self.my_images1 = []
        img=Image.open("D:\\ASHSIH\\PROJECTS\\version2\\Ryy\\r1.jpg")
        img=img.resize((540,350),Image.ANTIALIAS)
        self.my_images1.append(ImageTk.PhotoImage(img))
        img=Image.open("D:\\ASHSIH\\PROJECTS\\version2\\Ryy\\r2.jpg")
        img=img.resize((540,350),Image.ANTIALIAS)
        self.my_images1.append(ImageTk.PhotoImage(img))
        self.count =0
#-----------------------------------------------------
        self.my_images2 = []
        img=Image.open("Processed/p1.jpg")
        img=img.resize((540,350),Image.ANTIALIAS)
        self.my_images2.append(ImageTk.PhotoImage(img))
        img=Image.open("Processed/p2.jpg")
        img=img.resize((540,350),Image.ANTIALIAS)
        self.my_images2.append(ImageTk.PhotoImage(img))
       # set first image on canvas
        self.image_on_canvas1 = self.canvas1.create_image(50,7,  anchor = NW, image =
self.my_images1[self.count])
        self.image_on_canvas2 = self.canvas2.create_image(50,7, anchor = NW, image =
self.my_images2[self.count])
#-----------------------------------------------------
#Frame for the Side Section C
        self.label1=Label(window, text="Search Criminal Records",bg='black',fg='white',font='Times 20 bold ')
        self.label1.place(x=840,y=8)
#Side Section C contents
        self.text9=Entry(window,width='30',bg='black',fg='white',font='none 12 bold')
        self.text9.pack(ipady=10)
        self.text9.place(x=855,y=60)
```

```
        self.search=Button(window,text='Search',width=10,height=1,bg='red',fg='white',font='none 14
bold',command=self.search)

        self.search.place(x=855,y=100)

        self.reset=Button(window,text='Clear',width=10,height=1,bg='red',fg='white',font='none 14
bold',command=self.clearimg)

        self.reset.place(x=1000,y=100)

#Suspect Image Frame

        self.frameimage=Frame(window,width=195,height=215)

        self.frameimage.configure(background='black')

        self.frameimage.place(x=900,y=160)

#Suspect PrisonerID Label

        self.label10=Label(window,text='ID :',bg='black',fg='white',font='none 12 bold')

        self.label10.place(x=820,y=400)

        self.text10=Text(window,width=30,height=2,bg='black',fg='white',font='none 12
normal',state='disabled')

        self.text10.place(x=900,y=400)

#Suspect Name Label

        self.label2=Label(window,text='Name :',bg='black',fg='white',font='none 12 bold')

        self.label2.place(x=820,y=450)

        self.text2=Text(window,width=30,height=2,bg='black',fg='white',font='none 12
normal',state='disabled')

        self.text2.place(x=900,y=450)

#Suspect Wanted Level Label

        self.label3=Label(window,text='Level :',bg='black',fg='white',font='none 12 bold')

        self.label3.place(x=820,y=500)

        self.text3=Text(window,width=30,height=2,bg='black',fg='white',font='none 12
normal',state='disabled')

        self.text3.place(x=900,y=500)

#Suspect Crime/offence Type

        self.label4=Label(window,text='Offence :',bg='black',fg='white',font='none 12 bold')
```

```
        self.label4.place(x=820,y=550)

        self.text4=Text(window,width=30,height=2,bg='black',fg='white',font='none 12
normal',state='disabled')

        self.text4.place(x=900,y=550)


#Last Found Date

        self.label5=Label(window,text='Date :',bg='black',fg='white',font='none 12 bold')

        self.label5.place(x=820,y=600)

        self.text5=Text(window,width=30,height=2,bg='black',fg='white',font='none 12
normal',state='disabled')

        self.text5.place(x=900,y=600)

#Last Known Location

        self.label6=Label(window,text='Last Known Location :',bg='black',fg='white',font='none 12 bold')

        self.label6.place(x=820,y=650)

        self.text6=Text(window,width=30,height=2,bg='black',fg='white',font='none 12 normal')

        self.text6.place(x=900,y=680)

        self.Next()

#------------------------------------------------------------

#Main Window

window=Tk()

window.title("Security Surveillance System - S3")

window.geometry('1250x770')

window.resizable(0,0)

window.configure(background='black')

#-----------------------------------------------------

#Main menu bar

menubar=Menu(window)

live=Menu(menubar,tearoff=0)

menubar.add_cascade(label='Live',menu=live)
```

```
history=Menu(menubar,tearoff=0)
def popup_history():
    global top,tbox
    top=Toplevel(window)
    top.title("Date wise Records")
    top.geometry('700x600')
    top.resizable(0,0)
    top.configure(background='black')
    #tbox=Text(top,width=300,height=4,bg='blue',fg='white',text='',font='none 10 bold').place(x=50,y=120)
    mystring =StringVar(top)
    mystring1 =StringVar(top)
    lbl1=Label(top,bg='black',fg='white',text="Enter Date : ",font='none 10 bold').place(x=20,y=20)
    ent1=Entry(top,width=20,textvariable = mystring).place(x=160,y=20)
    lbl2=Label(top,bg='black',fg='white',text="Enter Location : ",font='none 10 bold').place(x=20,y=50)
    ent2=Entry(top,width=20,textvariable = mystring1).place(x=160,y=50)
    submit = Button(top,width=10,height=1,text="Search", fg="white", bg="Red",font='none 10
bold',command=lambda:search_history(mystring.get(),mystring1.get()))
    submit.place(x=60,y=80)
    clear = Button(top,width=10,height=1,text="Clear", fg="white", bg="Red",font='none 10
bold',command=clear_history)
    clear.place(x=160,y=80)
    #tbox=Text(top,width=100,height=4,bg='blue',fg='white',font='none 10 bold')
    #tbox.place(x=50,y=120)
    top.mainloop()
   menubar.add_cascade(label='History',menu=history)
history.add_command(label="HISTORY",command=popup_history)
window.config(menu=menubar)
mb=display(window)window.mainloop()
```

Conclusion

The Security Surveillance System hence provides a user friendly access and at the same time provides efficient memory utilization and accessing methods .The system also supports wireless multiple camera setup too. Hence it can form a large network of cameras connected to one or more controller nodes that co-ordinates the working of the system. This tremendously reduces the time taken to locate a wanted criminal and also helps in taking them down in real time.