MEGA PROJECT REPORT ...

SMART SAFE

Smart Safe Security System

DEPARTMENT OF ELECTRONICS

BATCH 2016-19

- ✓ **CREW NAME :** JEIS ROBOTICS
- ✓ **CREW MEMBERS :**
  - ***Arun jyothish k***
  - ***Arun k***
  - ***Ashwin Krishna C***
  - ***Vyshak MS***
  - ***Jishal***
  - ***Shanid***
  - ***Shibil***
  - ***Rishad Babu***
  - ***Sadhique Ali***
  - ***Jijesh kp***

GUIDED BY: .

PRAKASH P

GOVERNMENT POLYTECHNIC COLLEGE PERINTHALMANNA

DEPARTMENT OF ELECTRONICS ENGINEERING

CERTIFICATE

*This is to certify that the project report entitled "***Smart Safe Security System***"submitted as the record of the project presented by them, is accepted as the mega project report submission in fulfilment of the requirements for the award of diploma in electronics engineering under the Technical Education Department during the year 2018-19*

Guided By **HOD**

Internal Examiner **External Examiner**

ACKNOWLEDGEMENTS

We feel profound pleasure in bringing out this projects report for which we have to go from pillar to post to make it a reality. This project work reflects contributions of many people with whom we had long discussions and without which it would not have been possible .We must first express our heartiest gratitude to respected teacher PRAKASH P (Lecturer in Electronics Department) for providing us all guidance to complete the project.

It would be unfair if we do not mention the invaluable contribution and timely co-operation extended to us by staff members of our department. And especially we can never forget the most worthy advices given by RAJESH P (H.O.D, Electronics Department), that would help us the entire lifetime

Last but not the least we express our sincere thanks to the institute Govt. Polytechnic college Perinthalmanna for providing such a platform for implementing the idea in our mind

INDRODUCTION

Now a day's security is very important, security is a vast term, it is a reliable keeping of datas, things...etc. from peoples we don't want to share and easy access to those whom we need to share

Here we presenting a secure safe (locker), which is also smart.

We are presenting this for home application, as well industrial application

This type security system is applicable in many fields here we are demonstrating this with a "Locker" hence the name of project "Smart Safe"

CONTENTS

WORKING EXPLANATION

"Secure Safe" provide fingerprint security and code lock, the fingerprint have enrolled using program. When a finger is placed in the fingerprint, it detects automatically the presence of finger and search the match of preset fingerprints. If it matches, it allows the next step authentication i.e. pin lock if it is correct the lock opens,

For an exception if the fingerprint authentication

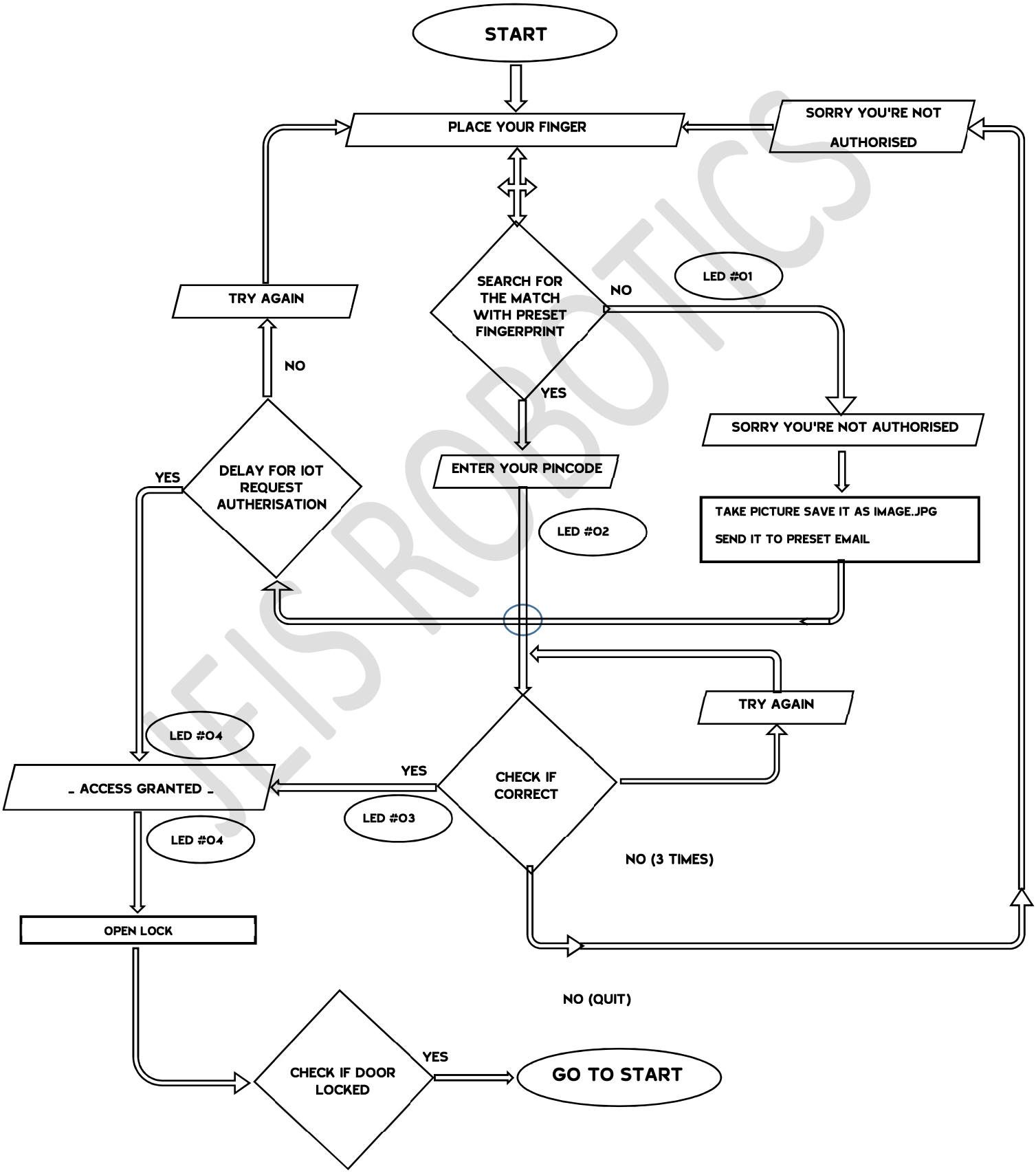
Failed The camera on the "Smart Safe"
Capture a picture, send it to the users (verified users) email, and alert the attempt. This abroad user can grant the attempt if it is a safe person and can alarm if the attempt is unsafe

The system have an oled display for further notification

The "Smart Safe" have built in audio synthesis program that allows blind persons to handle.

Verified user can also send a message to the person who attempt to open lock with this voice synthesis software via email message

WORKING BLOCK DAIGRAM



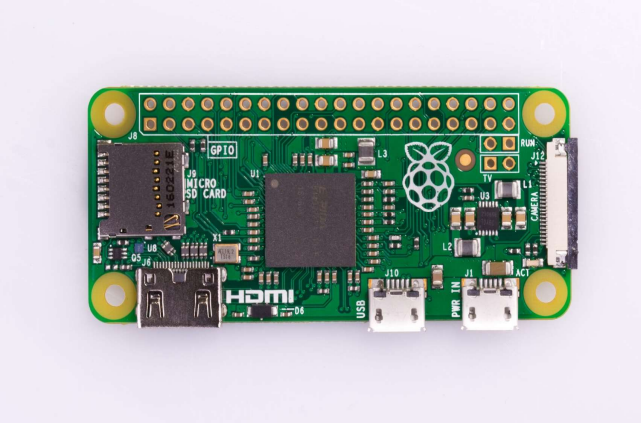
COMPONENTS USED

- **RASPBERRY PI ZERO**
- **USB TO TTL CONVERTER**
- **RASPBERRY CAMERA V1.3 5MP**
- **L293D H BRIDGE IC**
- **L78L33 3.3 V REGULATOR**
- **RESISTOR 220R*6,10K*14**
- **CAPACITOR**
- **PAM8403 AMPLIFIER MODULE**
- **SPEAKER 3WATT**
- **ZENER DIODE 3.3V**
- **LM7805 5V REGULATOR**
- **JM101 FINGERPRINT SENSOR ADAFRUIT**
- **OLED DISPLAY I2C SSD1306**
- **Li-ion BATTERY 3.7V * 2**
- **Li-ion CHARGING MODULE**

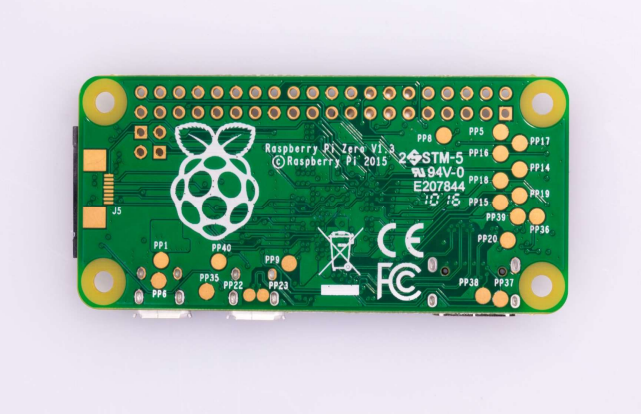
RASPBERRY PI

The Raspberry Pi is a low cost, credit-card sized computer that plugs into a computer monitor or TV, and uses a standard keyboard and mouse. It is a capable little device that enables people of all ages to explore computing, and to learn how to program in languages like Scratch and Python. It's capable of doing everything you'd expect a desktop computer to do, from browsing the internet and playing high-definition video, to making spreadsheets, word-processing, and playing games.

What's more, the Raspberry Pi has the ability to interact with the outside world, and has been used in a wide array of digital maker projects, from music machines and parent detectors to weather stations and tweeting birdhouses with infra-red cameras. We want to see the Raspberry Pi being used by kids all over the world to learn to program and understand how computers work.



- Raspberry pi zero w (upside)



- Raspberry pi zero w (bottomside)

specifications:

1GHZ BCM2835 SINGLE-CORE PROCESSOR		512MB RAM	SDCARD UPTO 64GB
INBUILT HDMI PORT	INBUILT CSI PORT	INBUILT DSI PORT	INBUILT WIFI+BLUETOOTH

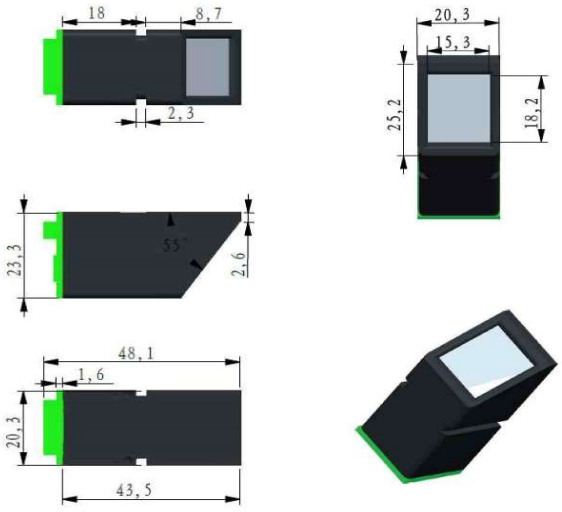
FINGERPRINT SENSOR

JM-101 fingerprint module is a fingerprint-processing module for integrating the light path and fingerprint processing part, has small volume, low power consumption, simple interface, high reliability, fast recognition and good adaptability for dry or wet finger, quick search speed of fingerprint

PIN No	Name	type	Function description
1	+3.3V	In	Module power positive input
2	TX	Out	Serial data output. TTL logic level.
3	RX	In	Serial data input. TTL logic level.
4	GND	—	Signal ground. Connected with power ground internally.
5	Touch	Out	sense signal output, high level effectivelyas default
6	TouchVin	In	Touch-sensitive power input, 3.3 v power supply
7	D+	—	USB D+
8	D-	—	USB D-

Working Principle:

Through the optical imaging principle, the grain caused by uneven medial skin of fingers, can form a variety of fingerprint image. The skin texture on the pattern, breakpoints, and intersection are different, called "feature points" in information processing, the characteristics of each finger is different, that is to say, is the only, rely on this kind of uniqueness, we can match a person with his fingerprint , through his fingerprints and preserved one in advance, comparing the fingerprints for verifying his real identity.



Adafruit fingerprint sensor jm101

OLED DISPLAY

OLED means Organic Light Emiting Diode SSD1306 is a low cost display unit it is very small but have 128*64 Pixels which is sufficient for many application it is 0.96 inch size,

Also consist of I2C interface which uses 2 wires so it is simple to handle

It operates with low voltage also the power consumption is less than that of an led

Also since it is an oled no backlight required. Adafruit is manufacturing this oled displays

Adafruit SSD1306 oled display

Front facing

Adafruit SSD1306 oled display

Back side view

Working voltage 3.3v to 5v

Logic level **0 >> 2.8-3.3**

2* I2C address **0x3C(default)** **0x3D**

Inbuilt i2c pull up resistors so can connect directly to MCU or MPU

CAMERA

The underlying technology of the digital camera is a light sensor and a program. The light sensor is most often a Charge Coupled Device (CCD) and the program is firmware that is embedded right into the circuit board of the camera.

I'll focus on the CCD first. Yes, there is another kind of light sensor that can be used and that's the Complimentary Metal Oxide Semiconductor (CMOS) type. The mechanics of how they do what they do differ, but the principles are the same.

Think of the CCD as being a grid of millions of little squares

Each of those little squares on the CCD takes light energy and converts it to electrical energy. Each condition of the light – like brightness and intensity – generates a very specific electrical charge. Those charges for each little square are then transported through an array of electronics to where it can be interpreted by the firmware. The firmware knows what each specific charge means and translates it to information that includes the colour and other qualities of the light that the CCD picked up. This process is done for each of the squares in the grid of the CCD

5MP Omnivision 5647 Camera Module

Still Picture Resolution: 2592 x 1944

Video: Supports 1080p @ 30fps, 720p @ 60fps and 640x480p 60/90 Recording

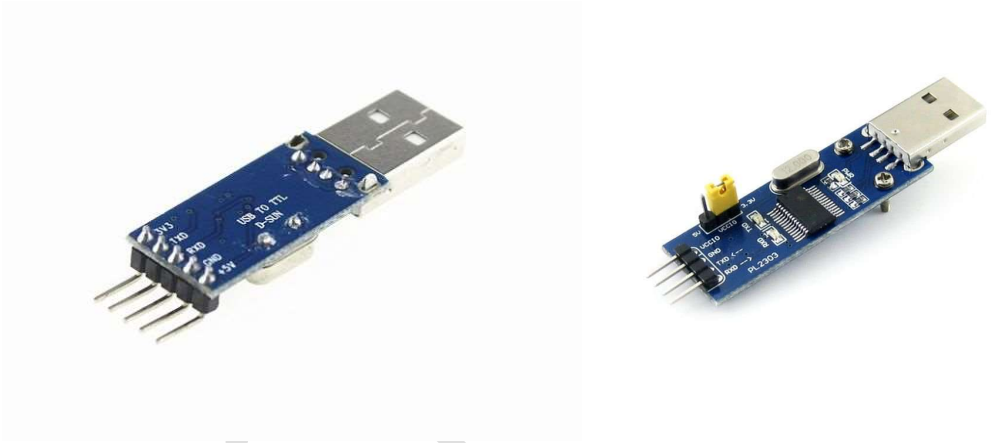
15-pin MIPI Camera Serial Interface - Plugs Directly into the Raspberry Pi Board

Size: 20 x 25 x 9mm

Weight 3g

FTDI ADAPTER

Universal Serial Bus (USB) is now established as the de-facto interface for connecting systems with a reliable, low-cost digital link. USB has expanded beyond PC usage, and can now be found in all market segments, including Industrial, Medical, Consumer, Communications, Networking and many more. Enabling designers to implement USB quickly into a design, FTDI Chip provides total solutions including silicon chips, development tools, application notes, and software support. Expertise in USB bridges provides seamless integration for a variety of interfaces such as UART, FIFO, I^2C, SPI, PWM and GPIO, where the bridge converts the signalling and protocol from the selected interface to USB. USB solutions are delivered in packages as small as 10 pin DFN (3x3mm) or as modules that can be inserted into boards for development and production or in cables that bridge USB to numerous interfaces. Here we are USB to TTL converter FDTI Adapter

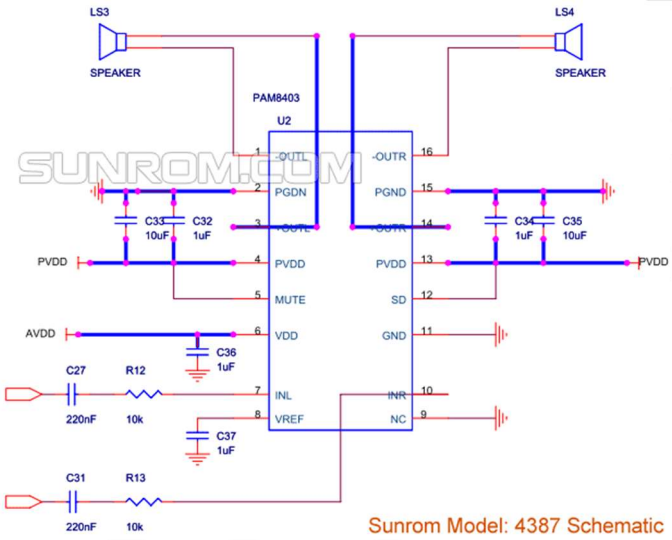


USB to TTL converter FTDI adapter

PAM8403 Audio Amplifier Module

Specifications:

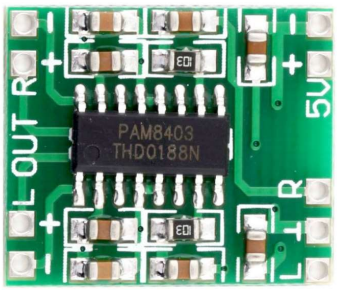
- Dual channel stereo output 3 w + 3 w power Class D
- Works with 2.5V-5v power supply
- High amplification efficiency 85%
- can directly drive 4 Ω/8 Ω small speakers
- Good sound quality & noise suppression
- Unique without LC filter class D digital power board
- Can use computer USB power supply directly
- Small Size, 1.85 x 2.11 cm can easily fit in a variety of products

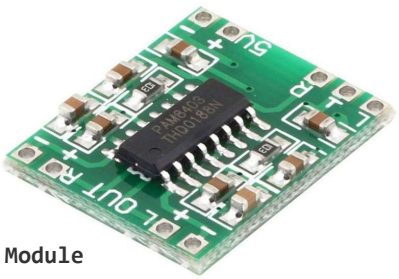


Sunrom Model: 4387 Schematic

If We Are Using 'Festival' Speech Synthesis Program It Have Very Low Sound Hence Amplification Of This Module Preset Isn't Enough So That

We Added Extra 10k Resistors Across R12 And R13 In ordered To Get Double Gain





PAM8403 Audio Amplifier Module

Picture

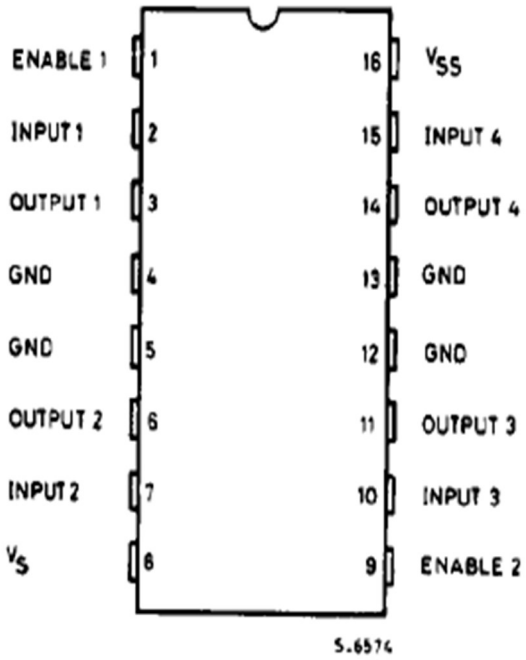
HBRIDGE IC

The L293D is a famous 16-Pin Motor Driver IC. As the name suggests it is mainly used to drive motors. A single L293D IC **is capable of running two DC motors at the same time; also the direction of these two motors can be controlled independently. So if you have motors which has operating voltage less than 36V and operating current less than 600mA, which are to be controlled by digital circuits like Op-Amp, 555 timers, digital gates or even Micron rollers like Arduino, PIC, ARM etc.. this IC will be the right choice for you**

Using this L293D motor driver IC is very simple. The IC works on the principle of Half H-Bridge, let us not go too deep into what H-Bridge means, but for now just know that H bridge is a set up which is used to run motors both in clock wise and anti clockwise direction. As said earlier this IC is capable of running two motors at the any direction at the same time.

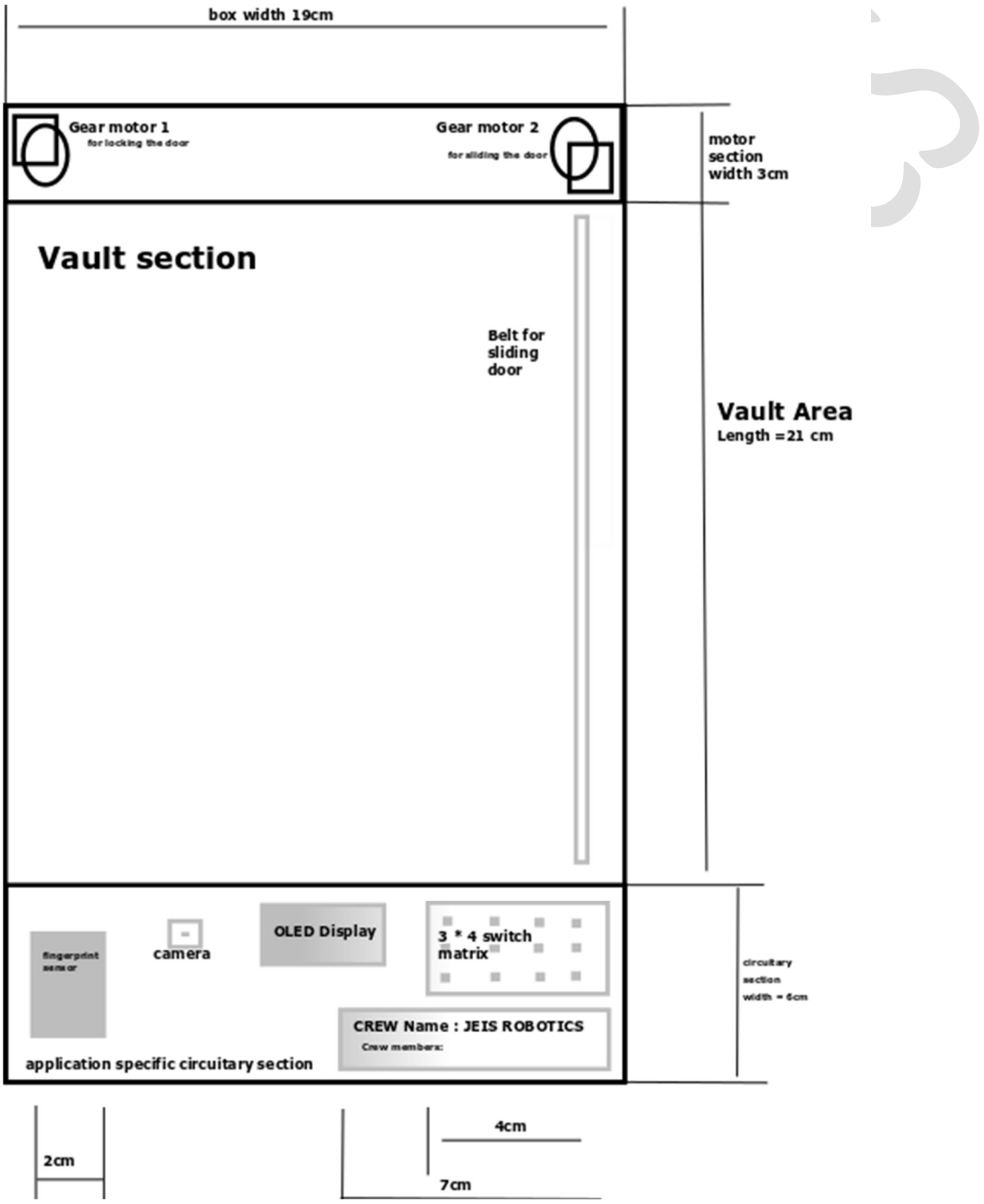
Features

- **Can be used to run Two DC motors with the same IC.**
- **Speed and Direction control is possible**
- **Motor voltage Vcc2 (Vs): 4.5V to 36V**
- **Maximum Peak motor current: 1.2A**
- **Maximum Continuous Motor Current: 600mA**
- **Supply Voltage to Vcc1(vss): 4.5V to 7V**
- **Transition time: 300ns (at 5Vand 24V)**
- **Automatic Thermal shutdown is available**
- **Available in 16-pin DIP, TSSOP, SOIC packages**



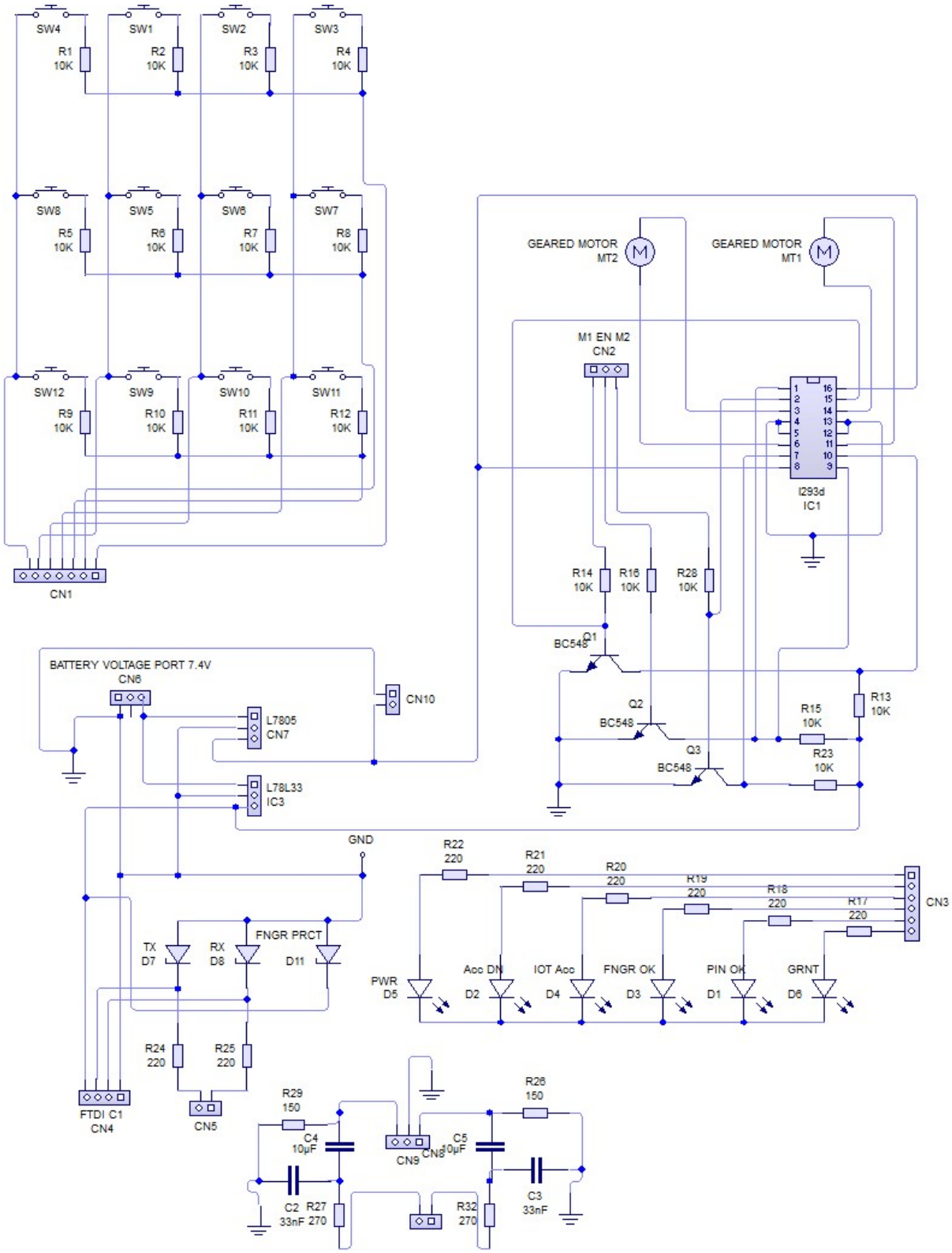
Project Drawings &Dimensions

“Smart Safe” is 30cm (length)*18cm (width)*4cm (Height) dimension box it is divided as 3 section one for fixing motor, second for keeping things ,and third for circuits



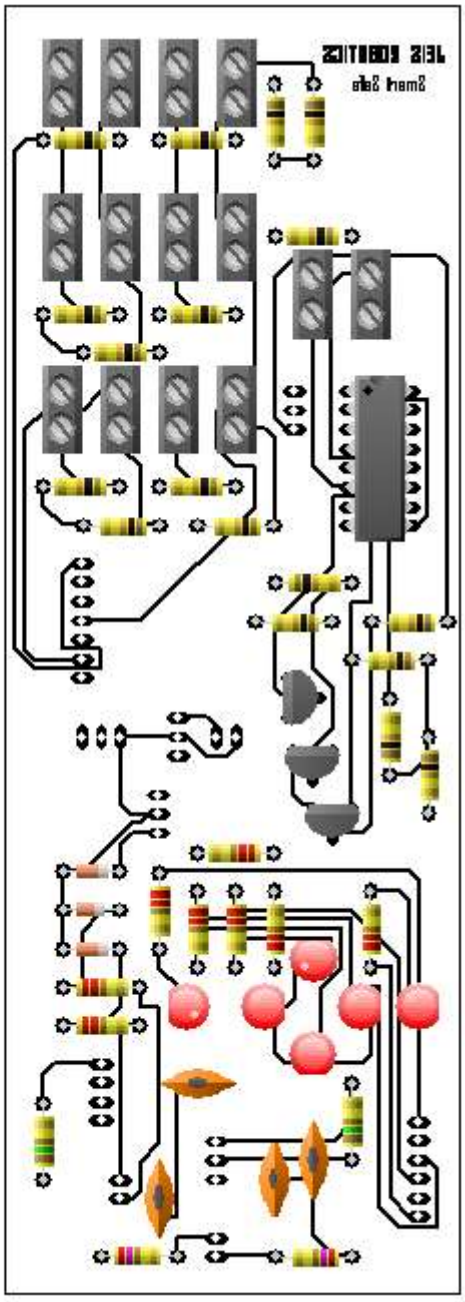
** Smart safe drawing

APPLICATION SPECIFIC CIRCUITRY



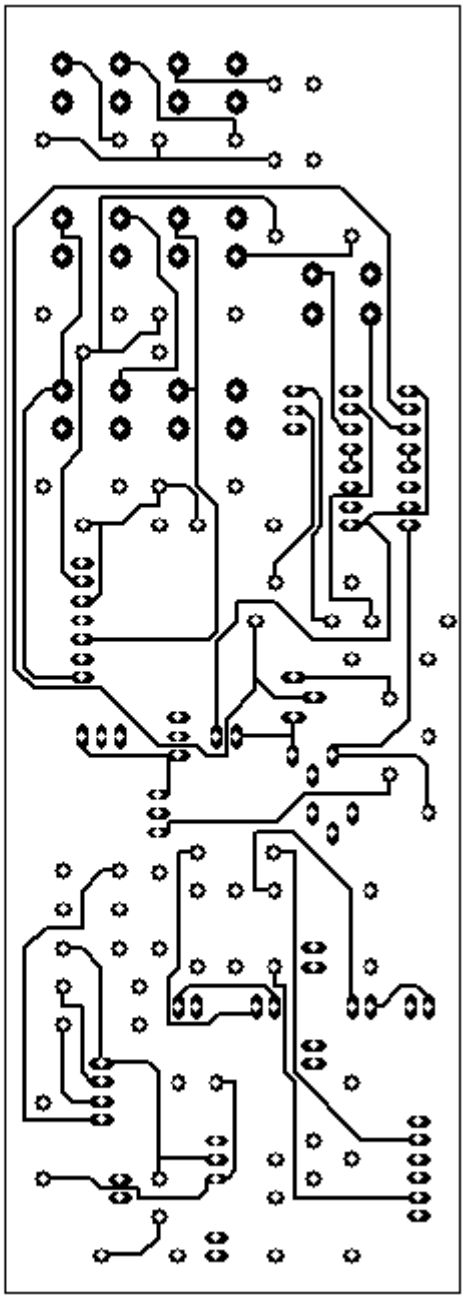
PCB DESIGN

COMPONENTS+ART WORK



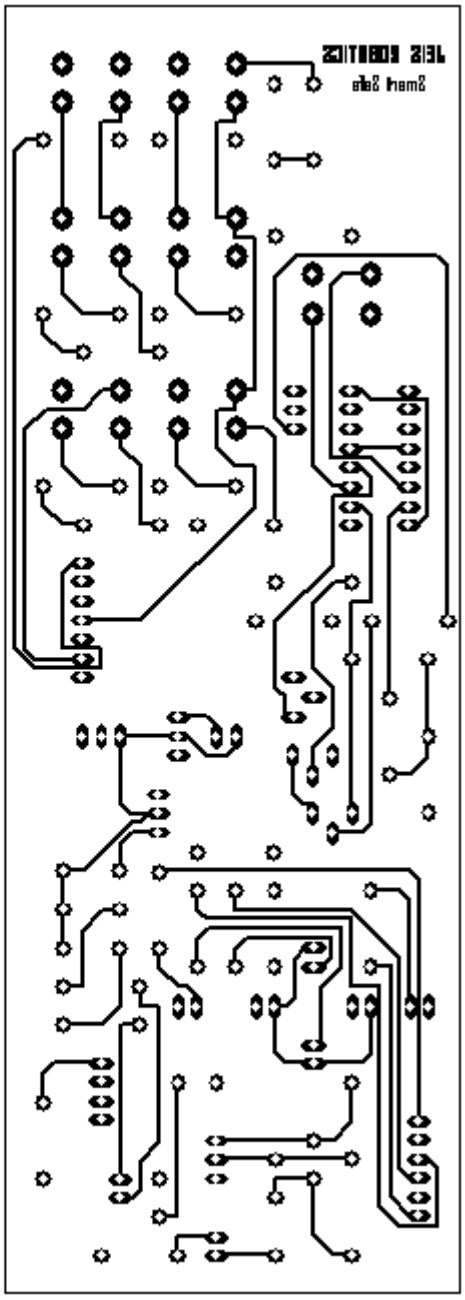
PCB DESIGN

PCB COMPONENT SIDE :



PCB DESIGN

PCB BOTTOM SIDE :

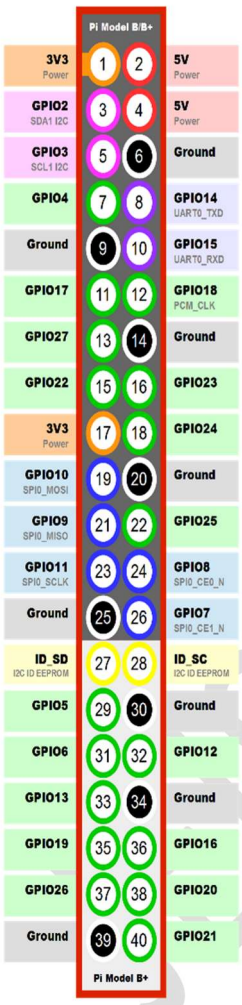


Interfacing

In PCB CN6 connected to 3.7v+3.7v 2 lion battery pack (If it is using) if battery is not using short CN7(LM7805) 1st pin to 3rd pin and connect CN10 with 5v regulated power supply

In PCB CN1 connected to raspberry pi pin no. 1st >> 7th 31,33,35,37,15,13, 11

CN9 to PAM 8403 amplifier module input Left,GND,Right

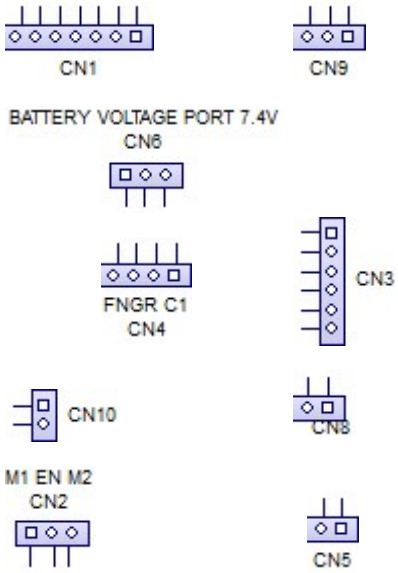


CN8 connected to PWM pins GPIO18(pin12),GPIO19(pin35) so we can get audio from raspberry pi zero

CN4 Connected to fingerprint sensor Gnd,Rx,Tx,Vcc from pcb secure 3.3v logic logic level

CN5 connected to USB to TTL converter Tx,Rx for UART protocol from raspberry zero usb connection

CN3 connected to Raspberry pin 18,22,32,36,38,40 these are for LED connection



CN2 connected to Raspberry pin 11,13,15 this connection to pcb goes to L293d H-bridge IC for driving two gear motors 13th pin enables both motor driver other 2 pins for two gear motors logic level determine direction

The I2C pins of raspberry pi gpio2 & gpio3 are connected to SSD1306 I2C oled display SDA and SCL pins respectively , and a 3.3 v supply are provided to it

Raspberry pi Zero W GPIO header pinout

Connector Header On PCB

**Note the connector pin counts from square indication*

PROGRAMMING

After proper hardware interfacing we need to program the raspberry pi

I. **Installing raspbian stretch-lite OS to raspberry pi**
   - **Download raspbian stetch-lite OS from official website**
   - **Flash it to a memory card using etcher | win32 diskimager**

II. **Install necessary packages**

```
sudo apt-get update
sudo apt-get upgrade
```

Get speech synthesis software:

```
sudo apt-get install festival
```

get email utils:

```
sudo apt-get ssmtp
sudo apt-get mailutils
```

Get source list for python:

```
sudo wget -O - http://apt.pm-codeworks.de/pm-codeworks.de.gpg  apt-key add -

wget http://apt.pm-codeworks.de/pm-codeworks.list -P /etc/apt/sources.list.d/
```

Get python fingerprint package:

```
sudo apt-get update
sudo apt-get install python-fingerprint --yes
```

Add fingerprint templates:

```
cd  /usr/share/doc/python-fingerprint/examples/example_enroll.py
```

Download Adafruit_Python_SSD1306-master

```
wget https://github.com/adafruit/Adafruit_Python_SSD1306/archive/master.zip
```

Setup Adafruit oled SSD1306:

```
pip install master.zip
```

Smart Safe Source Code:

_main.py is the python script which runs at boot up

Further it calls another program script named jeis_ss@jotix.py

During the excecution a sub program runs parrellel named sub.py

The jeis_ss@jotix.py considered as the souce program that runs ,branch the main features

The files that this project consist of :

main.py

jeis_ss@jotix.py

sub.py

mail_alert.py

log.txt

The program for this project that I written is mentioning below

```
"""
 ================================================================================
 Name        :jeis_ss@jotix.py

 Author      : ARUN JYOTHISH  K
 Version     :v1.0              | 12/12/2018 4:50 pm
 Copyright   : "Smart Safe" Open Source
 Description : Smart safe project, secure locker system including iot,finger
                              print,speech synthesis,display,pin code,using raspberry pi
 ================================================================================
 """

import Adafruit_SSD1306
import time
import datetime
from PIL import Image
from PIL import ImageDraw
from PIL import ImageFont
import hashlib
from pyfingerprint.pyfingerprint import PyFingerprint
import RPi.GPIO as GPIO

GPIO.setmode(GPIO.BCM)
GPIO.setwarnings(False)

matr0=[22,27,17]
matrI=[10,9,11,5]
```

```
fngrI=[18]
mot=[23,24,25,8]
led0=[26,19,16,20,6,21]

op=led0+matr0+fngrI+mot
ip=matrI+fngrI
for i in op:
          GPIO.setup(i,GPIO.OUT)
          GPIO.output(i,False)
for i in ip:
          GPIO.setup(i,GPIO.IN)

## Tries to initialize the sensor
try:
          f = PyFingerprint('/dev/ttyS0', 57600, 0xFFFFFFFF, 0x00000000)

          if ( f.verifyPassword() == False ):
                    raise ValueError('The given fingerprint sensor password is wrong!')
except Exception as e:
          print('The fingerprint sensor could not be initialized!')
          log('Exception message: ' + str(e))
          exit(1)

GPIO.output(led0[0],True)

import subprocess

RST = None

# 128x64 display with hardware I2C:
disp = Adafruit_SSD1306.SSD1306_128_64(rst=RST)


# Initialize library.
disp.begin()

# Clear display.
disp.clear()
disp.display()

# Create blank image for drawing.
# Make sure to create image with mode '1' for 1-bit color.
width = disp.width
height = disp.height
image = Image.new('1', (width, height))

# Get drawing object to draw on image.
draw = ImageDraw.Draw(image)

# Draw a black filled box to clear the image.
draw.rectangle((0,0,width,height), outline=0, fill=0)

# Draw some shapes.
# First define some constants to allow easy resizing of shapes.
padding = -2
top = padding+28
bottom = height-padding
# Move left to right keeping track of the current x position for drawing shapes.
x = 0
un = 0
# Load default font.
font = ImageFont.load_default()

#some global variable
prc=[""]
tmp=[""]
pos=-1
positionNumber = -1
us=[-1]
l2=30
l22=15
usrname=['jyothish k','Arun k','Ashwin c','Vyshak ms','jishal p','shanid p','shibil p','Rishad Babu','Sadhique Ali','jijesh
kp","Unknown..!"]
name=['jo','ar','as','vy','ji','sh','sp','ri','sa','jk','un']
mailaddresses=["arunjyothishvikku@gmail.com",
               "arunkvnb738@gmail.com",
               "ashwinkrishna62@gmail.com",
               "jishalarimbra@gmail.com",
               "cheekodeshand@gmail.com",
               "shibi4702@gmail.com",
               "vysakhms2016@gmail.com",
               "rishadbaburkz@gmail.com",
               "sadhiquealichulliyil404@gmail.com",
               "jijeshkpj@gmail.com"
               ]
password=["0010","7623","1122","8976","6452","0912","6104","8959","1324","8546","7483"]

def voice(file):
    log("voice : "+file)
    cmd="omxplayer /home/pi/project/Audio/voice_clip_"
    cmd+=file+".mp3"
    process=subprocess.call(cmd,shell=True)
    time.sleep(.03)
#    print(file+" playing status  : "+process)

def mailaddr(type):
#    print("mailaddr usrname us[0]  : "+usrname[us[0]])
    Time=datetime.datetime.now().replace(microsecond=0).time()
    Date=datetime.datetime.now().date()

    for addr in mailaddresses:
        if type=="unknown":
#            print("mailaddr if excecuted")

            cmd= "echo \" Smart Safe , Detected an Unknown  \'UNLOCK\' Attempt \n \n"
            cmd+="Date : " +str(Date)+"\nTime : "+str(Time)+"\n \n ___ JEIS ROBOTICS\" | mail -s \" SMART SAFE SECURITY ALERT ! \" "
            cmd+=addr
            cmd+=" & "
        elif type=="login":
#            print("mailaddr elif excecuted")
            cmd= "echo \" Smart Safe ,  "
            cmd+=str.upper(usrname[us[0]])
            cmd+="  logged in \n \n"
            cmd+="Date : "+str(Date)+"\nTime : "+str(Time)+" \n \n ___ JEIS ROBOTICS\" | mail -s \" SMART SAFE LOG IN ALERT ! \" "
            cmd+=addr
            cmd+=" & "
```

```
        else:
            pass
#            print("mailaddr else excecuted")
        mail_cmd=subprocess.call(cmd,shell=True)
        log(mail_cmd)


def stats_disp(k):
    if k == False:
        return
    else:
        # Draw a black filled box to clear the image.
        draw.rectangle((0,0,width,height), outline=0, fill=0)
        # Shell scripts for system monitoring from here : https://unix.stackexchange.com/questions/119126/command-to-display-memory-
usage-disk-usage-and-cpu-load
        cmd = "hostname -I | cut -d\' \' -f1"
        IP = subprocess.check_output(cmd, shell = True )
        cmd = "top -bn1 | grep load | awk '{printf \"CPU Load: %.2f\", $(NF-2)}'"
        CPU = subprocess.check_output(cmd, shell = True )
        cmd = "free -m | awk 'NR==2{printf \"Mem: %s/%sMB %.2f%%\", $3,$2,$3*100/$2 }'"
        MemUsage = subprocess.check_output(cmd, shell = True )
        cmd = "df -h | awk '$NF==\"/\"{printf \"Disk: %d/%dGB %s\", $3,$2,$5}'"
        Disk = subprocess.check_output(cmd, shell = True )
        draw.text((25,0),"JEIS ROBOTICS",font=font,fill=255)
        draw.text((0,14),"Smart Safe",font=font,fill=255)
        draw.text((x, top),       "IP: " + str(IP),  font=font, fill=255)
        draw.text((x,top+8),     str(CPU), font=font, fill=255)
        draw.text((x, top+16),    str(MemUsage),  font=font, fill=255)
        draw.text((x, top+25),    str(Disk),  font=font, fill=255)
        disp.image(image)
        disp.display()
        time.sleep(.1)

def fingerfn_disp(k):
    if k == False:
        return
    else:
        voice("scfp")
        # Draw a black filled box to clear the image.
        draw.rectangle((0,0,width,height), outline=0, fill=0)
       # print("value of top",top)
        draw.text((x, 15),       "S C A N N I N G... " ,  font=font, fill=255)
        draw.text((x+15, 12),     "fingerprint..../", font=font, fill=255)
        draw.text((x, 12+16),   "Tmpl/$."+ str(f.getTemplateCount()),  font=font, fill=255)
        disp.image(image)
        disp.display()
        time.sleep(.3)


def grnt_disp(k):
    if k == False:
        return
    else:
        voice("accgrnt")
        # Draw a black filled box to clear the image.
        draw.rectangle((0,0,width,height), outline=0, fill=0)
       # print("value of top",top)
        draw.text((x, 15),       "A C C E S S " ,  font=font, fill=255)
        draw.text((x+50, 12),     "G R A N T E D...", font=font, fill=255)
        draw.text((x+50, top+35),   "***",  font=font, fill=255)
        draw.text((x+45, top+40),   "**",  font=font, fill=255)
        GPIO.output(led0[5],True)
        disp.image(image)
        disp.display()
        time.sleep(.1)

def denied_disp(k):
    if k == False:
        return
    else:
        voice("accdenied")
        # Draw a black filled box to clear the image.
        draw.rectangle((0,0,width,height), outline=0, fill=0)
       # print("value of top",top)
        draw.text((x, 15),       "A C C E S S " ,  font=font, fill=255)
        draw.text((x+50, 12),     "D E N I E D....!", font=font, fill=255)
        draw.text((x+50, top+35),   "***",  font=font, fill=255)
        draw.text((x+45, top+40),   "**",  font=font, fill=255)
        GPIO.output(led0[4],True)
        disp.image(image)
        disp.display()
        time.sleep(.1)

def passed_disp(k):
    if k == False:
        return

    else:
        voice("succ")
        # Draw a black filled box to clear the image.
        draw.rectangle((0,0,width,height), outline=0, fill=0)
        draw.text((x, 15),       "P A S S E D .... " ,  font=font, fill=255)
        draw.text((x+25, 12),     "Routing to usr..../", font=font, fill=255)
        GPIO.output(led0[3],True)
        disp.image(image)
        disp.display()
        time.sleep(.1)


def failed_disp(k):
    if k == False:
        return
    else:
        voice("failed")
        voice("unknown")
        # Draw a black filled box to clear the image.
        draw.rectangle((0,0,width,height), outline=0, fill=0)
        draw.text((x, 15),       "F A I L E D .... " ,  font=font, fill=255)
        draw.text((x+15, 12),     "No usr match found.", font=font, fill=255)
        GPIO.output(led0[2],True)
        disp.image(image)
        disp.display()

        try:
            cmd="python /home/pi/mail_alert.py &"
            mail_alert=subprocess.call(cmd,shell=True)
```

```
#           print(mail_alert)
            time.sleep(.1)
        except Exception as e:
            log("Can not send mall_alert : "+str(e))


def usr_disp(k,pos):
    if k == False:
        return
    else:
        if not pos==-1 :
            mailaddr("login")
            voice("wl")
            voice(name[pos])
        else:
            mailaddr("unknown")

        # Draw a black filled box to clear the image.
        draw.rectangle((0,0,width,height), outline=0, fill=0)
        Time=datetime.datetime.now().replace(microsecond=0).time()
        Date=datetime.datetime.now().date()
       # print("value of top",top)
        draw.text((x, 05),       "LOG .... " ,  font=font, fill=255)
        draw.text((x, 122+5),     "usr: "+str(usrname[pos]), font=font, fill=255)
        draw.text((x, 40),     "Date : "+str(Date), font=font, fill=255)
        draw.text((x, 50),     "Time : "+str(Time), font=font, fill=255)
        disp.image(image)
        disp.display()
        time.sleep(.4)

def iot_disp(k,pos):
    if k == False:
        return
    else:
        voice("iot")
        # Draw a black filled box to clear the image.
        draw.rectangle((0,0,width,height), outline=0, fill=0)
        draw.text((x, 15),       "U N L O C K E D .... " ,  font=font, fill=255)
        draw.text((x+30, 12),     "via IoT key", font=font, fill=255)
        draw.text((x, top+26),   "_ by usr : "+str(usrname[pos]),  font=font, fill=255)
        GPIO.output(led0[1],True)
        disp.image(image)
        disp.display()
        time.sleep(.1)

def pin_disp(k,pos):
    if k == False:
        return
    else:
        voice("plsveri")
        # Draw a black filled box to clear the image.
        draw.rectangle((0,0,width,height), outline=0, fill=0)
        draw.text((x, 0),       "E N T E R __P I N " ,  font=font, fill=255)
        draw.text((x, 122),     str(usrname[pos])+"@ : ", font=font, fill=255)
        disp.image(image)
        disp.display()
        time.sleep(.1)

maskeyCount=[5]
def maskey(k):
    if k:
        maskeyCount[0]+=20
#        print(maskeyCount)
        v=35
        x=maskeyCount[0]
        draw.text((x, v),    " "+prc[0] , font=font, fill=255)
        disp.image(image)
        disp.display()
        time.sleep(.1)
    else:
        pass



def unlocked_disp(k):
    if k == False:
        return
    else:
        voice("unl")
        # Draw a black filled box to clear the image.
        draw.rectangle((0,0,width,height), outline=0, fill=0)
        draw.text((x, 15),       "U N L O C K E D /... " ,  font=font, fill=255)
        disp.image(image)
        disp.display()
        time.sleep(.1)

def locking_disp(k):
    if k == False:
        return
    else:
        voice("loc")
        draw.rectangle((0,0,width,height),outline=0,fill=0)
        draw.text((x,15),"L O C K I N G /...",font=font,fill=255)
        disp.image(image)
        disp.display()
        time.sleep(.1)

def fscanner_disp(k):
    if k == False:
        return -1
    else:
          templates=str(f.getTemplateCount())
          storageof=str(f.getStorageCapacity())
          ## Gets some sensor information
#         print('Currently used templates: ' + templates +'/'+ storageof )
          ## Tries to search the finger and calculate hash
          try:
                    log('Waiting for finger...')

                    ## Wait that finger is read
           while ( f.readImage() == False ):
               print ("in fingerprint scanning waiting")
               pass
                    ## Converts read image to characteristics and stores it in charbuffer 1
                    f.convertImage(0x01)
              ## Searchs template
```

```
                    result = f.searchTemplate()

                    positionNumber = result[0]
                    accuracyScore = result[1]
#                   log("match found")
                    if ( positionNumber == -1 ):
                              log('No match found!')
                              failed_disp(1)
                    else:
                              passed_disp(1)
                              log('Found template at position #' + str(positionNumber))
#                             print('The accuracy score is: ' + str(accuracyScore))

                    ## OPTIONAL stuff
          except Exception as e:
                    log('Operation failed!')
                    log('Exception message: ' + str(e))
          return positionNumber


list=[]
but=["NONE"]
#MATRIX BUTTONS FN
button=[[1,2,3,'$'],[4,5,6,0],[7,8,9,'%']]
#mmatrix switch fn
def push():
    rw=0
    cl=0
    but[0]="NONE"
    for i in matr0:
        GPIO.output(i,False)
        for j in matrI:
            if not GPIO.input(j):
#                list.append(i)
                rw=matr0.index(i)
#                list.append(j)
                cl=matrI.index(j)
                but[0]=button[rw][cl]
            else:
                pass

    #        list.append("__")
        GPIO.output(i,True)
def MOT(k,j):

    if k=="latch":
        if j=="forward":
            GPIO.output(23,True)
            GPIO.output(25,False)
        elif j=="reverse":
            GPIO.output(23,False)
            GPIO.output(25,True)
        else:
            GPIO.output(23,False)
            GPIO.output(25,False)
    else:
        pass

    if k=="slide":
        if j=="forward":
            GPIO.output(24,True)
            GPIO.output(8,False)
        elif j=="reverse":
            GPIO.output(24,False)
            GPIO.output(8,True)
        else:
            GPIO.output(24,False)
            GPIO.output(8,False)
    else:
        pass

def unl(k):
    if k:
        MOT("latch","reverse")
        grnt_disp(True)
        time.sleep(.5)
        MOT("slide","reverse")
        unlocked_disp(True)
        time.sleep(2)
        MOT("latch","")
        MOT("slide","")
    else:
        pass

    while True:
        push()
        if but[0]=="%":
            MOT("slide","forward")
            locking_disp(True)
            time.sleep(.5)
            MOT("latch","forward")
            time.sleep(2)
            MOT("latch","")
            MOT("slide","")
            voice("clsd")
            return main()
        else:
            pass
def log(para):
    if not tmp[0]==para:
        date_time=datetime.datetime.now()
        tmp[0]=para
        print(para)
        log=open("log.txt","a")
        log.write("\n\n"+str(date_time)+" :     "+str(para)+"\n")
        log.close()
    else:
        pass

def main():
    prc[0]=""
#    print(us[0])
    Tries=3
    touch=GPIO.input(18)
    stats_disp( not touch )
```

```
    fingerfn_disp(touch)
    if touch:
        us[0]=fscanner_disp(touch)
    else:
        pass
#    print(but[0])
#    time.sleep(.3)
    usr_disp(touch,us[0])
#    time.sleep(.3)
#    print(us[0])
    line=" User logged in: " +usrname[us[0]]
    log(line)
    if usrname[us[0]]=="Unknown..!":
        return run()
    else:
        while Tries:
            log("Tries loop running: "+str(Tries))
            push()
            pin=4
            if but[0]=="%":
                us[0]=-1
                break
            elif but[0]=="$":
#                print ("ok button")
                Tries-=1
                psk=""
                pin_disp(True,us[0])
                while pin:
#                    print("pin digits: "+str(pin))
                    push()
                    if but[0]=="%":
                        break
                    else:
                        if not but[0]=="NONE":
                            pin-=1
                            prc[0]=str(but[0])
                            psk+=prc[0]
#                            print("Enter code : "+psk)
                            maskey(True)
                        else:
#                            print("else excecuted")
                            pass
                    log("psk ip: "+psk)
                    pswd=password[us[0]]
#                    print("password set: " +pswd)
                if psk==pswd:
                    log("Code matched ..")
                    unl(True)
                    continue
                else:
                    log("Code error !")
                    denied_disp(True)
                    time.sleep(.3)
#                    print("Try..failed")
                    maskeyCount[0]=5
                    pin_disp(True,us[0])
                    pin=4
                    continue
            else:
                pass
        log("Attempt Exceeded !")
        us[0]=-1

def run():
#    print("run function ")
    while 1:

        try:
            cmd="python /home/pi/sub.py"
            mail_recieve=subprocess.call(cmd,shell=True)
        except Exception as e:
#            print("Can not recieve command from mail : "+str(e))
            pass

        main()

run()

"""
Smart Safe security Locker
Source code:
                    programmed by Arun jyothish k

"""
```

mail_alert.py code:

""

```
import picamera
import time

import smtplib
from email.MIMEMultipart import MIMEMultipart
from email.MIMEText import MIMEText
from email.MIMEBase import MIMEBase
from email import encoders
from email.mime.image import MIMEImage

fromaddr = "jeisdevice0@gmail.com"
toaddr = "arunjyothishvikku@gmail.com"

mail = MIMEMultipart()

mail['From'] = fromaddr
mail['To'] = toaddr
mail['Subject'] = "Security Alert"
body = "Unknown Attempt Detected..
data=""

def sendMail(data):

    mail.attach(MIMEText(body, 'plain'))
    print data
    dat='%s.jpg'%data
    print dat
    attachment = open(dat, 'rb')
    image=MIMEImage(attachment.read())
    attachment.close()
    mail.attach(image)

    server = smtplib.SMTP('smtp.gmail.com', 587)
    server.starttls()
    server.login(fromaddr, "jyothujyothu")
    text = mail.as_string()
    server.sendmail(fromaddr, toaddr, text)
    server.quit()

def capture_image():
    camera = picamera.PiCamera()
    camera.rotation=180
    camera.awb_mode= 'auto'
    camera.brightness=55

    data= time.strftime("%d_%b_%Y|%H:%M:%S")
```

```
    camera.start_preview()
    time.sleep(.05)
    print data
    camera.capture('%s.jpg'%data)
    camera.stop_preview()
    time.sleep(.05)
    sendMail(data)



capture_image()



"""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""
```

main.py code:

```
import subprocess
import Adafruit_SSD1306
import time
import datetime
from PIL import Image
from PIL import ImageDraw
from PIL import ImageFont
import hashlib



print(" starting...program..")


cmd = "python /home/pi/jeis_ss@jotix.py"


err=subprocess.call(cmd,shell=True)


print(err)


cmd = " omxplayer /home/pi/project/Audio/voice_clip_errordtct.mp3"
subprocess.call(cmd,shell=True)


RST = None
disp = Adafruit_SSD1306.SSD1306_128_64(rst=RST)
font = ImageFont.load_default()
# Initialize library.
disp.begin()
# Clear display.
disp.clear()
disp.display()
# Create blank image for drawing.
# Make sure to create image with mode '1' for 1-bit color.
width = disp.width
height = disp.height
```

```
image = Image.new('1', (width, height))
# Get drawing object to draw on image.
draw = ImageDraw.Draw(image)
# Draw a black filled box to clear the image.
draw.rectangle((0,0,width,height), outline=0, fill=0)
draw.text((0, 5),  "Error Detected !",  font=font, fill=255)
draw.text((0, 20),  "Rebooting System ",  font=font, fill=255)
draw.text((10, 40),  "in 10 seconds... ",  font=font, fill=255)
disp.image(image)
disp.display()
time.sleep(10)
cmd="sudo reboot now"
#subprocess.call(cmd,shell=True)
```

CONCLUSION

We Designed This Security System For A Locker For Proving Its Reliability ,In Future There Are Many Application For This Security System Such As Room Lock, Manufacturing Centres ,Its Stripped Version Can Be Used For Car Lock,..Etc

But In Real-time The Iot Security We Provided Is Not Enough It Should Be Encrypted

Because The Fraudsters Are Being Developing Their Skills

We Hope You Got A New Knowledge From Our Sincere Effort

REFERENCES

- https://www.raspberrypi.org/documentation/configuration/wireless/wireless-cli.md
- http://21st-century-spring.kr/wordpress/?page_id=794
- https://learn.adafruit.com/adding-basic-audio-ouput-to-raspberry-pi-zero/overview
- https://www.dexterindustries.com/howto/run-a-program-on-your-raspberry-pi-at-startup/
- https://www.electronicwings.com/raspberry-pi/raspberry-pi-uart-communication-using-python-and-c
- https://circuitdigest.com/microcontroller-projects/raspberry-pi-iot-intruder-alert-system
- https://github.com/recalbox/recalbox-os/wiki/Analog-Audio-Pi-Zero-(EN)
- https://learn.adafruit.com/assets/59441