الأكاديمية العربية للعلوم والتكنولوجيا والنقل البحري

Arab Academy for Science, Technology & Maritime Transport

Computer organization

3-step authentication security system

Report to: Dr. Amr El Saadany & Eng. Hossam Elsayed

Report by: Belal Mohamed Sameh

Email: belalsameh188@gmail.com

Table of Contents

1 Introduction

1.1 Statement of problem

The project implement in this report is a 3-step authentication security system that can be widely used in companies, schools, universities, etc.… The three steps are fingerprint recognition, face mask detection and temperature scanning.

1.2 Report layout

Firstly, section 2 will explain the theoretical work done in the project which are the modelling and implementation of the security system. Secondly, the hardware implementation of the project. Then the software implementation of the project. Finally, the conclusion of the project and the future work to be done.

2 Theoretical works

2.1 Modelling

The model used to build the project was a simple model that uses staff that are available at any home. The model contains two main boxes. One box contains the Raspberry pi 4 processor, camera, temperature sensor and LCD interface. The other box contains 12V power supply, fingerprint sensor, servo motor and the breadboard that connects both boxes together. The boxes are connected using a stainless-steel holder that carries the wires through it.

2.2 Implementation

To implement this model A raspberry pi 4 model B processor has been used which is an ARM based processor (ARM Cortex A-72). The processor was used as it has a high-performance rate and low power consumption. The high-level language used to implement this project is python.

3 Hardware implementation

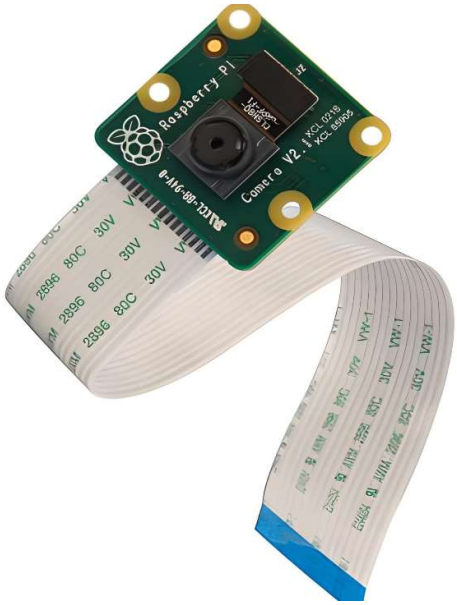
3.1 Fingerprint recognition

The fingerprint authentication step was implemented using a GROW K202 board and R503 capacitive fingerprint sensor. The R502 sensor detects the fingerprint and sends its readings to the K202 board the processes the fingerprint automatically. The board has a capacity of 2000 fingerprint that can be set using the set button on the board. The K202 board outputs whether the fingerprint detected is on its dataset or not using a relay. The relay output pin is connected to one of the raspberry pi GPIO pins so that we can process this input and give an output depending on it which will be explained in the software section. The

sensor requires a 12V power supply so a 12V external power supply is used to power the board and the sensor.

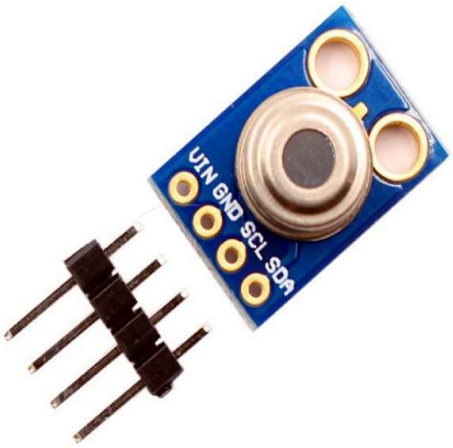
3.2 Face mask detection

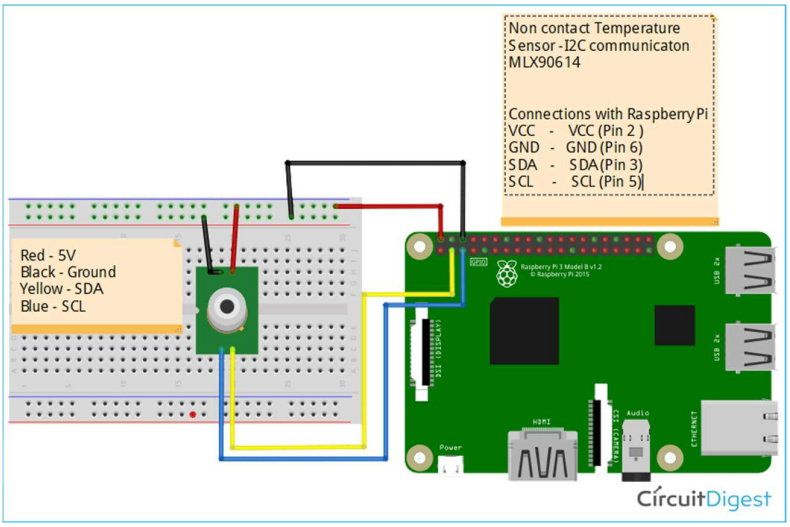
To detect whether the person entering is wearing a face mask or not a camera is used. The camera used in this project is the PI camera V2.1. The camera is connected to the raspberry pi through the camera slot in the Pi and the rest of the implementation in the section is explained in the software section. This camera was used as it has high quality and has 8MP quality and can video stream a video with 1080p.



3.3 Temperature scanning

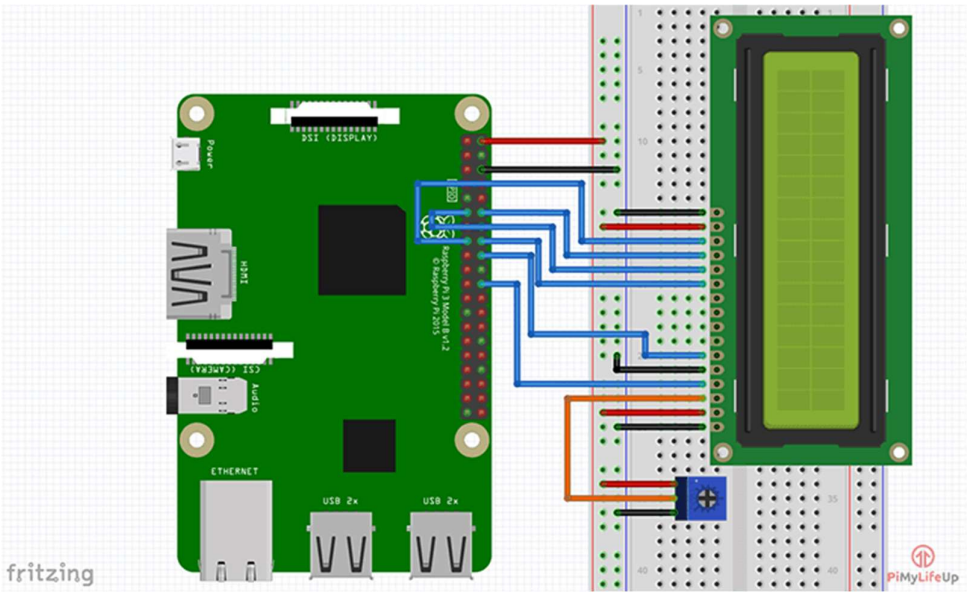
To measure the user entering temperature A MLX90614 sensor is used to measure the person entering temperature. The MLX90614 sensor has 4 pins VIN, GND, SDA, SCL which are connected directly to the Pi. The sensor uses i2c to communicate with the raspberry pi. The sensor works by sending a ray that hits an object and measure the temperature of the object that the ray hits. The sensor has a high measuring accuracy with just +/- 0.2 error.





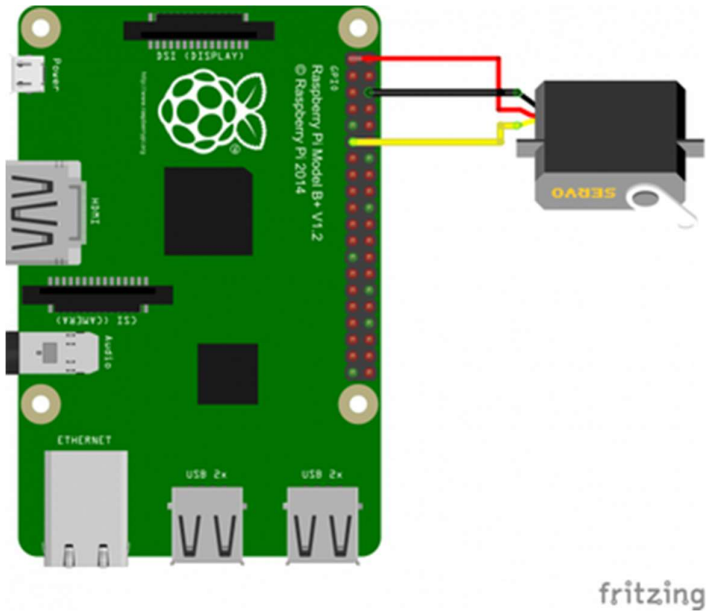
3.4 LCD interface

A 16x2 LCD is used to display messages and outputs to the user depending on the current situation. The LCD is connected to the raspberry pi GPIO, 5V and GND pins as well as to a potentiometer that controls the contrast on the LCD and a 330-ohm resistor.



3.5 Servo motor

The servo motor is used to control the door that is going to be opened when the user passes all the authentication steps. The servo motor 3 pins are connected to the raspberry pi through the 5V, GND and a GPIO pin. The servo rotates 90 degrees to open the door and then returns to its original position as locked.



4 Software implementation

4.1 Startup

Libraries used throughout the whole code and lcd pins initialization.

```
#Voice output

import pyttsx3 as py

################################


#Servo setup

import RPi.GPIO as GPIO
from gpiozero import AngularServo
GPIO.setmode(GPIO.BCM)
################################

#LCD setup

from RPLCD import CharLCD# Importing Adafruit library for LCD
from time import sleep # Importing sleep from time library to add delay in program
# initiate lcd and specify pins
lcd = CharLCD(cols=16, rows=2, pin_rs=26, pin_e=19, pins_data=[13, 6, 5, 21]
              , numbering_mode=GPIO.BCM)
lcd.clear()
```

```
#Facemask setup

from tensorflow.keras.applications.mobilenet_v2 import preprocess_input
from tensorflow.keras.preprocessing.image import img_to_array
from tensorflow.keras.models import load_model
from imutils.video import VideoStream
import numpy as np
import argparse
import imutils
import time
import cv2
import os
```

```
#Temperature setup

from smbus2 import SMBus
from mlx90614 import MLX90614
```

GPIO pins initialization

```
GPIO.setup(18, GPIO.OUT)
servo=AngularServo(18,min_pulse_width=0.0006,max_pulse_width=0.0023)
GPIO.setu
        lcd.clear()
        #bus = SMBus(1)
        #sensor = MLX90614(bus, address=0x5A)
        engine = py.init()
        engine.say("Welcome to A A S T. C A I")
        lcd.write_string(u"Welcome to AAST CAI")
        engine.runAndWait()
        sleep(5)
        lcd.clear()
        fingerflag=0
        maskflag=0
        tempflag=0
```

Initializing flags that refer to the state of each authentication level and giving the welcome message on the LCD and using the audio output connected through the AV jack.

4.2 Fingerprint recognition

The fingerprint code section is simple as it just asks the user to place their finger on the sensor and then the K202 board sends a digital 1 for a correct fingerprint and a 0 for a wrong fingerprint through the GPIO pin. If the pin gets a 1 the user is passed to the next authentication step else the user is not allowed to enter and the next user is then asked to scan their finger.

```
engine.say("Scan your finger")
engine.runAndWait()
lcd.write_string(u"Scan your finger")
sleep(7)
fingerflag=GPIO.input(23)
sleep(3)
if(fingerflag==1):
    lcd.clear()
    engine.say("Finger print detected")
    engine.runAndWait()
    lcd.write_string(u"Finger print detected")
    sleep(5)
elif(fingerflag==0):
    lcd.clear()
    engine.say("Finger print not detected")
    engine.runAndWait()
    lcd.write_string(u"Finger print not detected")
    sleep(5)
    lcd.clear()
    engine.say("Not allowed to enter")
    engine.runAndWait()
    lcd.clear()
    lcd.write_string(u"Not allowed to enter")
    sleep(5)
    lcd.clear()
    continue
```

4.3 Face mask detection

The face mask code section is the most complicated as it uses a previously trained model. The code logic is that the camera starts a video stream, and the user is asked to face the camera and then the camera captures the current frame and processes it using the pre trained model and its dataset. As the image is processed, first of all the face is detected and given certain value which is the confidence level of the current face whether the person wears a mask or not. The function that detects the face and mask confidence returns a confidence value for wearing a mask and not wearing a mask. Both values are compared together and if the confidence level of the mask is greater than without mask the user is passed to the next authentication step rather than that the user is not allowed to enter and is the steps are repeated from the beginning with the next user.

```
engine.say("Please wait")
engine.runAndWait()
lcd.clear()
lcd.write_string(u"Loading ...")
# construct the argument parser and parse the arguments
ap = argparse.ArgumentParser()
ap.add_argument("-f", "--face", type=str,
default="face_detector",
help="path to face detector model directory")
ap.add_argument("-m", "--model", type=str,
default="mask_detector.model",
help="path to trained face mask detector model")
ap.add_argument("-c", "--confidence", type=float, default=0.5,
help="minimum probability to filter weak detections")
args = vars(ap.parse_args())

# load our serialized face detector model from disk
print("[INFO] loading face detector model...")
prototxtPath = os.path.sep.join([args["face"], "deploy.prototxt"])
weightsPath = os.path.sep.join([args["face"],
"res10_300x300_ssd_iter_140000.caffemodel"])
faceNet = cv2.dnn.readNet(prototxtPath, weightsPath)

# load the face mask detector model from disk
print("[INFO] loading face mask detector model...")
maskNet = load_model(args["model"])

# initialize the video stream and allow the camera sensor to warm up
print("[INFO] starting video stream...")
#vs = VideoStream(src=0).start()
vs = VideoStream(usePiCamera=True).start()
time.sleep(2.0)
lcd.clear()
engine.say("Face the camera")
engine.runAndWait()
lcd.write_string(u"Face the camera")
sleep(5)

# grab the frame from the threaded video stream and resize it
# to have a maximum width of 400 pixels
frame = vs.read()
frame = imutils.resize(frame, width=500)

# detect faces in the frame and determine if they are wearing a
# face mask or not
(locs, preds) = detect_and_predict_mask(frame, faceNet, maskNet)
```

```
# loop over the detected face locations and their corresponding
# locations
for (box, pred) in zip(locs, preds):
    # unpack the bounding box and predictions
    (startX, startY, endX, endY) = box
    (mask, withoutMask) = pred

    # determine the class label and color we'll use to draw
    # the bounding box and text
    if mask > withoutMask:
      lcd.clear()
      maskflag=1
      engine.say("Thanks for wearing mask")
      engine.runAndWait()
      label=("Thanks for wearing mask")
      color = (0, 255, 0)
      lcd.write_string(u"Mask detected")
      sleep(5)
    else:
        lcd.clear()
        maskflag=0
        engine.say("Not mask detected")
        engine.runAndWait()
        lcd.write_string(u"No mask detected Try again")
        sleep(5)
        lcd.clear()
        engine.say("Not allowed to enter")
        engine.runAndWait()
        label=("No mask detected")
        color = (0, 0, 255)
        lcd.write_string(u"Not allowed to enter")
        sleep(5)

cv2.destroyAllWindows()
vs.stop()

if(maskflag==0):
    continue
```

4.4 Temperature scanning

The temperature scanning code section is simple as the user is asked to face the sensor and then the sensor is asked to send the ray and get the temperature of the object in front of it. The input temperature is compared with the value 37.3 degree if the user temperature is less than this value the user is allowed to enter rather than that the user is not allowed to enter and is returned to first step of authentication for the next user.

```
y = sensor.get_object_1()
if(y<37.3):
    lcd.clear()
    engine.say("Temperature is normal")
    engine.runAndWait()
    lcd.write_string(u"Temperature is normal")
    sleep(5)
else:
    lcd.clear()
    engine.say("Temperature is high")
    engine.runAndWait()
    lcd.write_string(u"Temperature is high")
    sleep(5)
    lcd.clear()
    engine.say("Not allowed to enter")
    engine.runAndWait()
    lcd.write_string(u"Not allowed to enter")
    sleep(5)
    continue
bus.close()
```

4.5 Servo motor

The last step if the user has passed all the authentication steps is to open the door for the user which is done using a servo motor. If the code reached that part this means that it has not been broken by any authentication step, so the servo automatically rotates by 90 degrees for 5 seconds and then turns back to close the door and the previous operations are repeated with the next user.

```
    engine.say("Allowed to enter. Welcome to College of artificial intelligence")
    engine.runAndWait()
    lcd.write_string(u"Welcome to C A I")
    servo.angle=90
    sleep(5)
    servo.angle=0
    sleep(5)

pwm.stop()
GPIO.cleanup()
```

5 Conclusion

To sum up, the project built has a high accuracy and can be used in many places for many applications rather in companies, schools, or universities. Its nowadays more reliable to use such a system rather than having a human guard standing there for 24 hours.

6 Future work

A lot of future work can be done in this project to improve its performance. Firstly, we could add a function that if any user failed in one of the authentication steps the camera captures an image of this user and saves it in a file. These files could be divided into different files to categories the reasons of why the user was not allowed to get in, such as if the user failed in the fingerprint stage the controller of the system will know by that way who tries to get in and is not allowed to do so. Moreover, if the user is not allowed to get in because of the temperature then the controller of the system can know that the person has attended but was not well in order to get in. Secondly, the face mask detection model could be trained more with a larger dataset that will increase the model accuracy.

7 List of components

	Component (Quantity)	Price
First	Raspberry pi 4 (1)	4200 EGP
Second	Raspberry pi heat sink (1)	30 EGP
Third	Raspberry pi camera (1)	800 EGP
Fourth	Fingerprint scanner (1)	2000 EGP
Fifth	Temperature sensor (1)	560 EGP
Sixth	Servo motor (1)	75 EGP
Seventh	16x2 LCD	120 EGP
Eighth	Resistors (2)	20 EGP
Ninth	12v power supply (1)	130 EGP
Tenth	Jumpers kit (2)	100 EGP
Eleventh	Body	150 EGP
	Total	**EGP 8185**

8 Datasheets

Raspberry pi 4 model b:
https://datasheets.raspberrypi.com/rpi4/raspberry-pi-4-datasheet.pdf

Raspberry pi camera V2.1:
https://www.raspberrypi.com/documentation/accessories/camera.html

MLX90614:
https://www.sparkfun.com/datasheets/Sensors/Temperature/MLX90614_rev001.pdf

GROW K202:
http://myosuploads3.banggood.com/products/20200612/20200612052830Usermanualforcontrolboard K202.pdf

R503:
https://robu.in/wp-content/uploads/2020/10/Grow-K202-R503-DC12V-Low-Power-Consumption-Ring-Fingerprint-Access-Control-Board-1.pdf

16x2 LCD:
https://www.vishay.com/docs/37299/37299.pdf

Servo motor:
https://robojax.com/learn/arduino/robojax-servo-sg90_datasheet.pdf

9 Appendix

Starting idea:
https://nevonprojects.com/iot-temperature-mask-scan-entry-system-for-covid-prevention/

Trained model source:
https://www.tomshardware.com/how-to/raspberry-pi-face-mask-detector