Introduction to IOT
Final Project Report

Member: 0716103 潘建琿 & 0716336 郭志龍

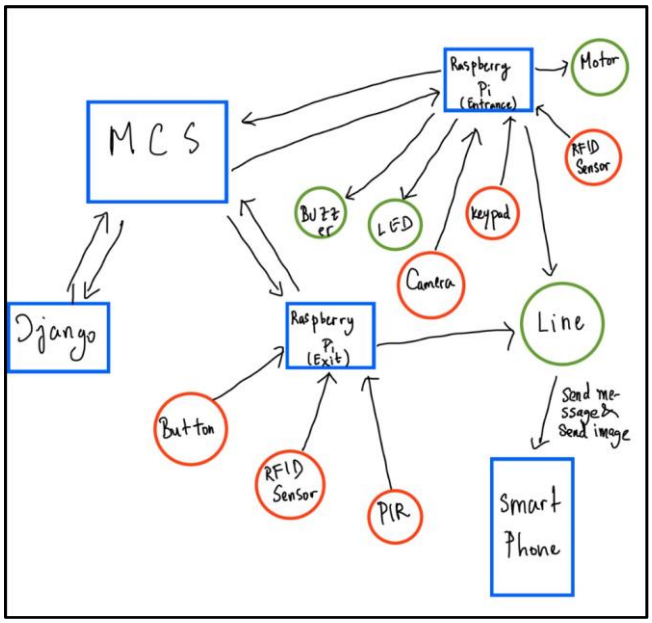
Objective:

1. Improve our IoT infrastructure.
2. Adding more function to our IoT infrastructure.
3. To let the user feel more secure and comfortable when using our system

Specification that we used:

1. 2 Raspberry Pi
2. 2 RFID sensors
3. 3 RFID cards
4. 1 Keypad
5. 1 LED
6. 1 Buzzer
7. 1 Servo motor
8. 1 Button
9. Pi Camera
10. PIR Sensor
11. A lot of Jumper wire
12. 2 Breadboards
13. Machine to edit card information through web service (by Django)
14. Smartphone connect to LINE
15. MCS

System design:



System design overview:

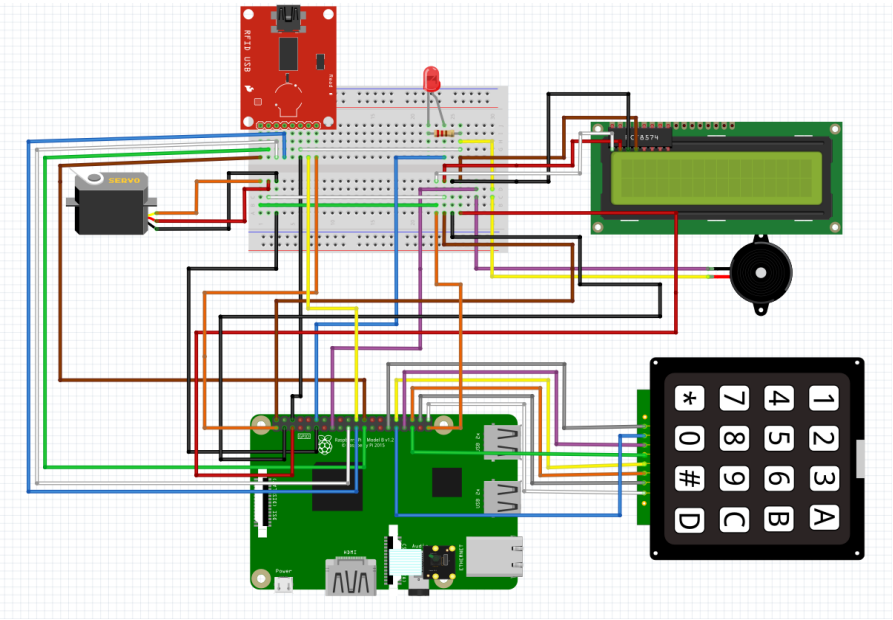
We do several modifications to the previous project (project 2) such as adding several sensors and actuators, and also add some important features in the UI (Django) to make the user more comfortable and feel more secure when using it.

What we modify:

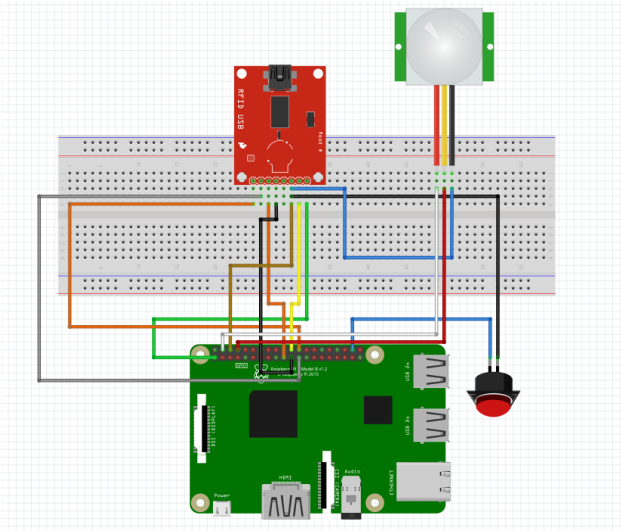
1. Adding face recognition as another authentication
2. Capture photo intruder when “unregistered” card detected
3. Adding PIR Sensor to second Raspberry Pi to detect intruder
4. Send Intruder Photo to Line

Raspberry Pi Structure:

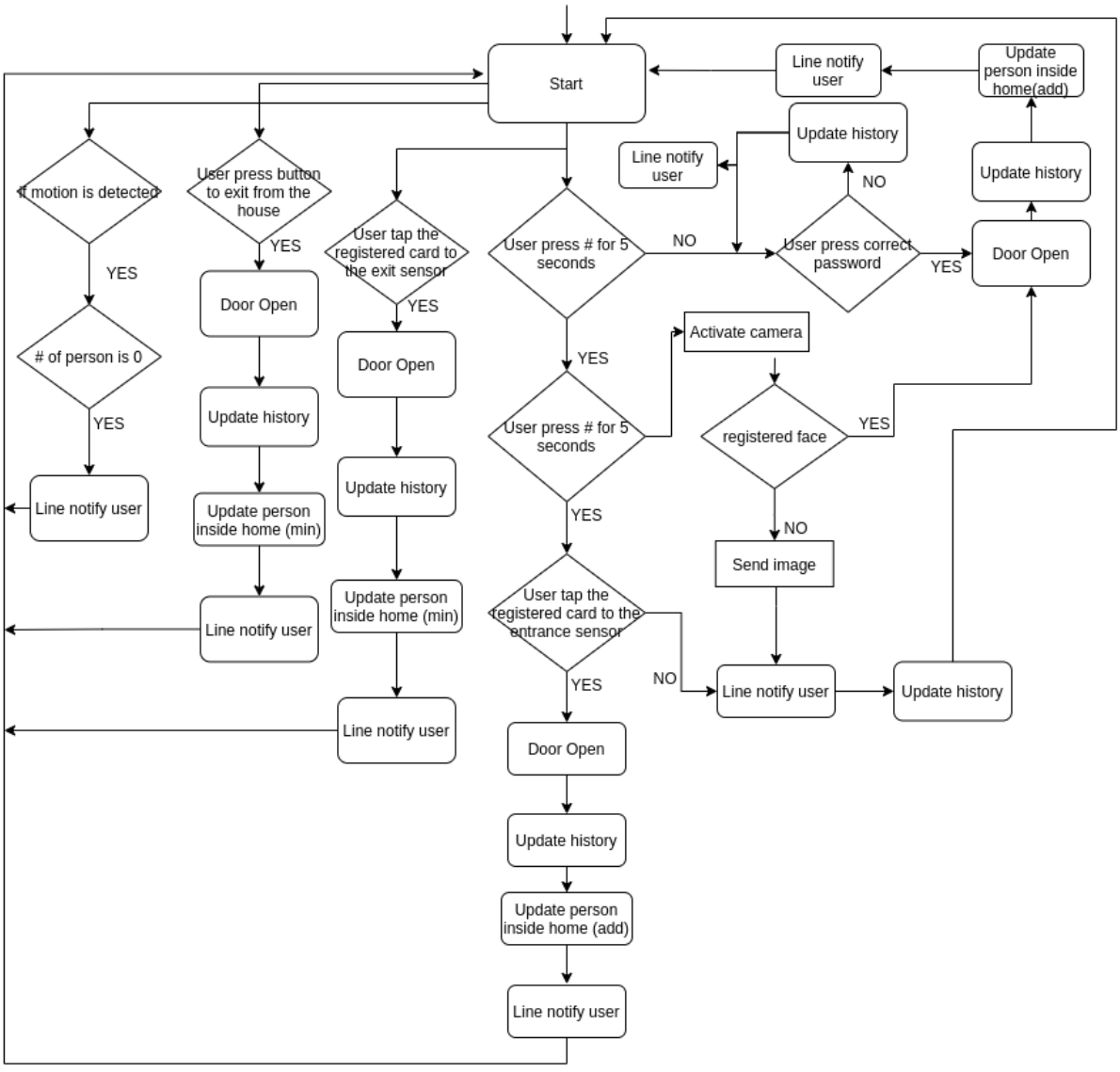
Main Pi:



Second Pi:



Flow chart:


Start
Line notify user
Update person inside home(add)
Update history
Line notify user
Update history
NO
If motion is detected
User press button to exit from the house
User tap the registered card to the exit sensor
User press # for 5 seconds
NO
User press correct password
YES
Door Open
YES
YES
YES
Door Open
Activate camera
# of person is 0
Door Open
Update history
User press # for 5 seconds
registered face
YES
YES
YES
Line notify user
Update person inside home (min)
Update history
NO
Send image
Line notify user
Update person inside home (min)
User tap the registered card to the entrance sensor
Line notify user
NO
Line notify user
Update history
YES
Line notify user
Door Open
Update history
Update person inside home (add)
Line notify user

Code:

Face.py: this code is used to get a face detection model that will be used in the face recognition from our available dataset.
Running this code will give an output "pickle" file that will be used for the face recognition.

```python
from imutils import paths
import face_recognition
import argparse
import pickle
import cv2
import os
# construct the argument parser and parse the arguments
ap = argparse.ArgumentParser()
ap.add_argument("-i", "--dataset", required=True,
    help="path to input directory of faces + images")
ap.add_argument("-e", "--encodings", required=True,
    help="path to serialized db of facial encodings")
ap.add_argument("-d", "--detection-method", type=str, default="cnn",
    help="face detection model to use: either `hog` or `cnn`")
args = vars(ap.parse_args())


print("[INFO] quantifying faces...")
imagePaths = list(paths.list_images(args["dataset"]))
knownEncodings = []
knownNames = []

for (i, imagePath) in enumerate(imagePaths):
    # extract the person name from the image path
    print("[INFO] processing image {}/{}".format(i + 1,
        len(imagePaths)))
    name = imagePath.split(os.path.sep)[-2]
    image = cv2.imread(imagePath)
    rgb = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)
    boxes = face_recognition.face_locations(rgb,
        model=args["detection_method"])

    encodings = face_recognition.face_encodings(rgb, boxes)

    for encoding in encodings:
        knownEncodings.append(encoding)
        knownNames.append(name)

print("[INFO] serializing encodings...")
data = {"encodings": knownEncodings, "names": knownNames}
f = open(args["encodings"], "wb")
f.write(pickle.dumps(data))
f.close()
```

Line 1~6 for importing the used library. Line 8~15 is for argument parsing that is needed by the program. In this program we will analyze the dataset that will be used to do face recognition. The first argument in line 9 is for the directory that has a dataset. The second argument is the result of the encoding. The third argument is the model that we will use in face detection. Line 23~37 is the program that will encode each file in the dataset. Line 39~43 outputs the encoding result that will be used later to do face recognition.

Main Code:

Below is the new added code, this part of the code is 2 functions that will be used when we will photo the intruder that used an “unregistered” RFID Card. The function capture_photo() is the function to take a photo from our camera and write a file. After that we will encode the file and post it to mcs. The function line_and_photo are the function that used to send the photo to line.

```
78  def line_and_photo():
79      line_url = 'https://notify-api.line.me/api/notify'
80      headers = {'Authorization': 'Bearer ' + token}
81      payload = {'message': 'intruder'}
82      files = {'imageFile': open('wrong_card.jpg', 'rb')}
83      r = requests.post(line_url, headers=headers, params=payload, files=files)
84      if files:
85          files['imageFile'].close()
86      return r.status_code
87
88  def capture_photo():
89      frame = vs.read()
90      cv2.imwrite("wrong_card.jpg",frame)
91      with open("wrong_card.jpg","rb") as img_file:
92          EncodeBytes = base64.b64encode(img_file.read())
93
94      EncodeStr=str(EncodeBytes,"utf-8")
95      post_to_mcs(8,EncodeStr)
96      #print("ENCODE:" ,EncodeStr)
97      time.sleep(10)
98      line_and_photo()
99
```

At line 79~81 is the line_url,headers, and payload that will be sent to LINE. Line 82 is adding the photo that we take using the function capture_photo(). Then using the request post function we send the payload into the LINE.

LCD Display Code new Function:

```
217         def welcome_face(name):
218             lcd_init()
219             time.sleep(0.5)
220             # Send some test
221             lcd_string("  Face Recognized ",LCD_LINE_2)
222             lcd_string("  "+name,LCD_LINE_3)
223         def face_recog_mode():
224             lcd_init()
225             time.sleep(0.5)
226             lcd_string("    Welcome...     ",LCD_LINE_1)
227             lcd_string(" Please Face Camera",LCD_LINE_2)
228             lcd_string("  Hold # for RFID ",LCD_LINE_3)
229
```

Because we add one more new “mode” which is Face Recognition Mode. in previous project we have 2 mode which are the keypad mode and the RFID Mode, and now we add more which is the face recognition mode. Therefore we add another function to display the “Face Recognized Mode” in the LCD Display for the user to know. The first function welcome_face is the function that displays that the face is recognized and login allowed. The face_recog_mode() function is the function that will display to the user that the system is at “Face Recognition Mode”.

Pass_key function modification:

```
318         if(liness[0:5]=="####"):
319             print("switch to face recognition mode")
320             lcd_display("face mode","")
321             print("[INFO] loading encodings + face detector...")
322             data = pickle.loads(open("res.pickle", "rb").read())
323             detector = cv2.CascadeClassifier("haarcascade_frontalface_default.xml")
324             #print("[INFO] starting video stream...")
325             #vs = VideoStream(0).start()
326             print("AFTER VS")
327             time.sleep(2.0)
328             liness=""
329             user_name=""
330             while True:
331                 readLine(L1, ["1","2","3","A"])
332                 readLine(L2, ["4","5","6","B"])
333                 readLine(L3, ["7","8","9","C"])
334                 readLine(L4, ["*","0","#","D"])
335                 time.sleep(0.175)
336                 print("HERE:" ,liness)
337 
338                 frame = vs.read()
339                 frame = imutils.resize(frame, width=500)
340 
341                 gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
342                 rgb = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
343                 rects = detector.detectMultiScale(gray, scaleFactor=1.1,
344                     minNeighbors=5, minSize=(30, 30))
345                 boxes = [(y, x + w, y + h, x) for (x, y, w, h) in rects]
346 
347                 encodings = face_recognition.face_encodings(rgb, boxes)
348                 names = []
349                 det=0
350 
351                 for encoding in encodings:
352                     matches = face_recognition.compare_faces(data["encodings"],
353                         encoding)
354                     name = "Unknown"
355 
356                     if True in matches:
357                         matchedIdxs = [i for (i, b) in enumerate(matches) if b]
358                         counts = {}
359                         for i in matchedIdxs:
360                             name = data["names"][i]
361                             counts[name] = counts.get(name, 0) + 1
362                         name = max(counts, key=counts.get)
363                         user_name=name
364                         det=1
365 
366                     names.append(name)
367 
```

Because we add one more “mode” which is face recognition mode we have to adjust the system. Therefore when the system enters the pass_key function which represents the keypad mode. To transition from keypad mode to the face recognition mode we ask the user to enter the “#” key 4 times and the system will change into the face recognition mode.

Face Recognition Mode Code :

At line 320 we will use function LCD Display to display to the user that we are currently in the “Face Recognition Mode”. The line 322 and 323 is we get the data that will be used in the program that are required in order to implement the “Face Recognition Mode” the pickle file that we used are the result from running the “face.py” program using the dataset of face we want to recognised.. At line 331 to 335 is a function to read from the keypad. The keypad input will be used to change into RFID Authentication mode. At lien 338 we read the frame from the camera, and line 339 is resizing the frame in order to display it at the screen. Line 341~345 is the preparation for displaying the video stream windows. At line 351 to 366 is a for loop to keep matching the video stream with the face that we have in our encodings from our “pickle” encoding. At line 356 to 364 is to match the video stream with our data, once it matches we will attach a name to the frame. The “res.pickle” we used are the result after we run the “face.py” code which give us an output of encodings from our dataset.

```
368                 for ((top, right, bottom, left), name) in zip(boxes, names):
369                     # draw the predicted face name on the image
370                     cv2.rectangle(frame, (left, top), (right, bottom),
371                         (0, 255, 0), 2)
372                     y = top - 15 if top - 15 > 15 else top + 15
373                     cv2.putText(frame, name, (left, y), cv2.FONT_HERSHEY_SIMPLEX,
374                         0.75, (0, 255, 0), 2)
375                 cv2.imshow("Frame", frame)
376                 key = cv2.waitKey(1) & 0xFF
377 
378                 if(det or liness[0:5]=="####"):
379                     cv2.destroyAllWindows()
380                     #vs.stop()
381                     if(liness[0:5]=="####"):
382                         print("switch to rfid mode")
383                     elif(det):
384                         s=""
385                         s=get_to_mcs(6)
386                         now=datetime.now()
387                         date_time= now.strftime("%m/%d/%Y, %H:%M:%S")
388                         s=s+user_name+",Face Recognition,"+date_time+","
389                         #time.sleep(0.1)
390                         post_to_mcs(6,s)
391                         post_to_mcs(7,0)
392                         lcd_display("face",user_name)
393                         motor_on()
394                         print("Face Recognized")
395                         msg=name+" Enter Using Face Recognition"
396                         linenotify(token,msg)
397                         people_in = int(get_to_mcs(4))
398                         people_in+=1
399                         post_to_mcs(4,people_in)
400                         #time.sleep(2)
401                         lcd_display("welcome","")
402                     break
403 
404             #print("switch to RFID")
405             if(liness[0:5]=="####"):
406                 break
407 
```

Line 368~375 is adjusting the frame we get in order to put it into the window so we can see the frame we get from the camera. At lines 378 we have an if condition that will check whether we need to change mode or we found a match in the frame. At line 383~401 is when we found a match face from the camera we will add the number of people in the house and send it back to mcs and also add the history of login in the django and sending line notification to line to give information about the login.

RFID Mode modification at the while:

```
504         else:
505             #wrong card
506             s=""
507             s=get_to_mcs(6)
508             now=datetime.now()
509             date_time= now.strftime("%m/%d/%Y, %H:%M:%S")
510             s=s+"Wrong card,"+"anon"+","+date_time+","
511             time.sleep(0.1)
512             post_to_mcs(6,s)
513             msg="Unknown card detected, someone try to enter your home"
514             linenotify(token,msg)
515             #capture_photo here
516             capture_photo()
517             for i in range(2):
518                 GPIO.output(ledPin,GPIO.HIGH)
519                 time.sleep(0.2)
520                 GPIO.output(buzzer,GPIO.HIGH)
521                 print ("Beep")
522                 time.sleep(0.2) # Delay in seconds
523                 GPIO.output(buzzer,GPIO.LOW)
524                 GPIO.output(ledPin,GPIO.LOW)
525 
526             lcd_display("wrong","")
527             print("access denied")
528             time.sleep(3)
529             post_to_mcs(3, str(id))
530             lcd_display("clear","")
531 
532 
533     #pass_key()
534     time.sleep(0.5)
535 
536 
```

The program will enter this part of the code when an unregistered RFID Card is detected. The part we modify here is that we use the function “capture_photo()” that explained above. That

will be used to take the photo of possible intruders that try to enter the house using an “unregistered” RFID Card.

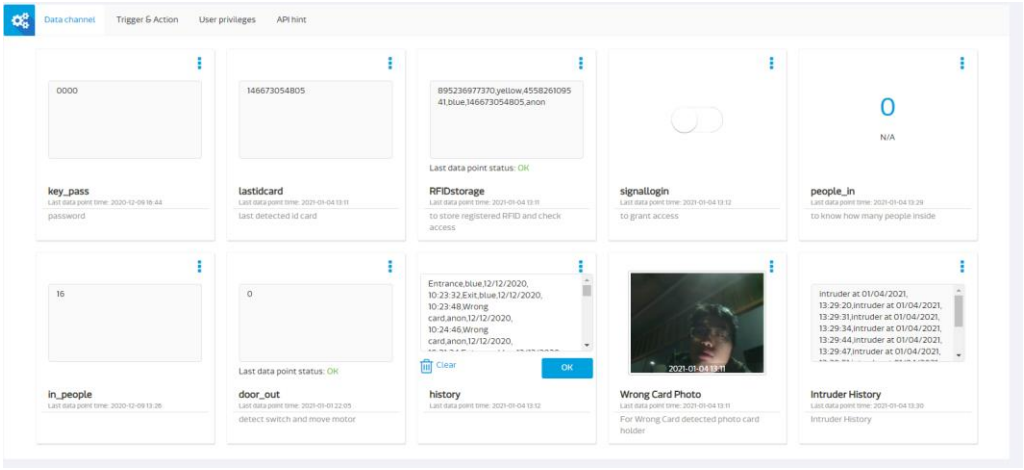
Second PI code:

In the second Pi we added a PIR sensor. The PIR sensor is used to detect if there is a person in a room that is supposed to be empty.

```
130 def join_and_exit():
131     print('Join Process and exit system')
132     p1.join()
133     p2.join()
134     exit()
135 post_to_mcs(9,"")
136 while True:
137     try:
138         i=GPIO.input(PIR)
139         people_in = int(get_to_mcs(4))
140         print("people inside now ",people_in)
141         if(i==0):
142             print('safe')
143             time.sleep(2)
144         elif(i):
145             people_in = int(get_to_mcs(4))
146             if(people_in==0):
147                 print('intruder enter')
148                 msg='intruder enter an empty house'
149                 linenotify(token,msg)
150                 intruder_history=""
151                 intruder_history=get_to_mcs(9)
152                 now=datetime.now()
153                 date_time= now.strftime("%m/%d/%Y, %H:%M:%S")
154                 intruder_history=intruder_history+"intruder at "+date_time+','
155                 print(intruder_history)
156                 post_to_mcs(9,intruder_history)
157                 time.sleep(2)
158     except KeyboardInterrupt:
159         print("End Program")
160         join_and_exit()
161
162
```

At line 138 we use variable i to get input from the PIR sensor. And if the value of PIR Sensor is 0 that means there are no people in the room. At line 144 we check if the PIR Sensor has a value more than 0, then we check at line 146 if there is a registered login then it will be fine. But if the people_in value is 0 but the PIR Sensor value is 1 then we will use a linenotify function to notify the user that there is an intruder that enters the room. At line 150~157 we get the data from the MCS which contains the history of the detected intruder in the room and update the data then post it into the MCS. Line 158~160 is to detect keyboard interrupt and end the program and join the process(For RFID Sensor and Button Sensor) that we used in our program.

MCS Test devices interface



Django Interfaces:

Welcome to Smart Security System

By: 潘建琿 Felix & 郭志龍 Eric

Change Keypad Password

Edit Card Account

Entrance Exit History

Private Area History

Current number of person inside house: 0

Edit Card Account

Current ID Card

Card code	Holder name	Remove
895236977370	yellow	remove
455826109541	blue	remove

Tap your card to the sensor and fill your name

Card holder name:

Add new card

Back

History of home entrance exit

Status	Id	Date	Time
Entrance	blue	12/12/2020	10:23:32
Exit	blue	12/12/2020	10:23:48
Wrong card	anon	12/12/2020	10:24:46
Wrong card	anon	12/12/2020	10:31:34
Entrance	blue	12/12/2020	10:32:40
Entrance	yellow	12/12/2020	10:33:01
Exit	blue	12/12/2020	10:32:53
Entrance	blue	01/01/2021	19:50:00
Entrance	yellow	01/01/2021	19:50:27
Wrong card	anon	01/01/2021	19:50:53
Wrong card	anon	01/01/2021	19:51:43
Wrong card	anon	01/01/2021	19:53:54
Entrance	yellow	01/01/2021	19:55:11
Wrong card	anon	01/01/2021	20:12:09
Wrong card	anon	01/01/2021	20:13:31

Secure area History

Date	Time
01/04/2021	13:29:20
01/04/2021	13:29:31
01/04/2021	13:29:34
01/04/2021	13:29:44
01/04/2021	13:29:47
01/04/2021	13:29:51
01/04/2021	13:29:54
01/04/2021	13:29:57
01/04/2021	13:30:00
01/04/2021	13:30:03

Back

Demo Video Link : https://www.youtube.com/watch?v=NRKN4Bc25BI