종합설계작품계획서

작품명 : 물류관리 시스템

4차 발표

학 번	이 름
20191629	정승훈
20191633	이남웅

목차

작품개요

1. 작품 소개

물류관리시스템은 프로세서인 Raspberry PI를 기반으로 IR 센서, 화재 감지 센서, 물 감지 센서 등을 활용하여 효율적으로 창고를 관리하고 현장에서는 LCD와 Buzzer로 창고의 상태를 확인할 수 있다.

2. 제작 배경 및 목적

인력중심의 물류관리 시스템을 벗어나기 위해 4차 산업혁명에 걸맞는 자동화 시스템을 구현하였다. 제품에 붙어있는 바코드를 통해 저장하려는 위치를 읽어내어 해당 제품을 자동화 시스템을 통해 사람 손을 거치지 않고 제품을 효율적으로 이동 및 관리할 수 있다.

이러한 시스템을 통해 물건을 손쉽게 관리 할 수 있고 바코드로 인한 물건의 데이터화로 이는 앞으로 빅데이터가 중요하게 여겨지는 4차 산업혁명 시대의 시스템에 대한 물류 업무표준화를 이루어 낼 수 있습니다.

업무 분담

H/W
정승훈

S/W
이남웅

기구부
공동

업 무	담 당 자
자료수집	정승훈, 이남웅
도면작성	정승훈
자재구입	정승훈, 이남웅
PCB조립	정승훈
프로그램 작성	이남웅
기구부 제작	정승훈, 이남웅
작품연동검사	정승훈, 이남웅

진행일정표

		6월	7월				8월					9월	
		5	1	2	3	4	1	2	3	4	5	1	2
자료수집	계획												
	실적												
하드웨어 구성	계획												
	실적												
프로그래밍	계획												
	실적												
기구물 제작	계획												
	실적												
동작테스트	계획												
	실적												
발표(1차, 2차, 3차, 최종)	계획	1차			2차						3차	4차	최종
	실적												
최종보고서	계획												
	실적												

계획	

실적	

동작설명서

- 물류관리 시스템은 Pi 카메라를 통해 제품에 있는 바코드를 읽어 위치정보를 불러들여 제품이 있는 컨베이어 벨트가 작동하게 되고, 제품이 교차점에 도달하게 되면 교차점에 있는 IR 센서가 제품 위치를 인식하게 되고 제품을 이동시킬지 판별한다. Pi 카메라에서 바코드 인식을 못할 경우 Buzzer가 작동하여 경적음을 울린다.

- 만약 해당 교차점에서 이동하려고 하면 서브모터가 제품을 이동시키게 되고, 컨베이어 벨트에서 지정된 위치로 이동시키게 된다. 제품의 효율적 관리를 위해 Flame 센서와 Gas 센서, Water 센서, 온습도 센서를 내장하게 되고 LCD 와 Buzzer를 통해 물류창고 상태를 확인할 수 있다.

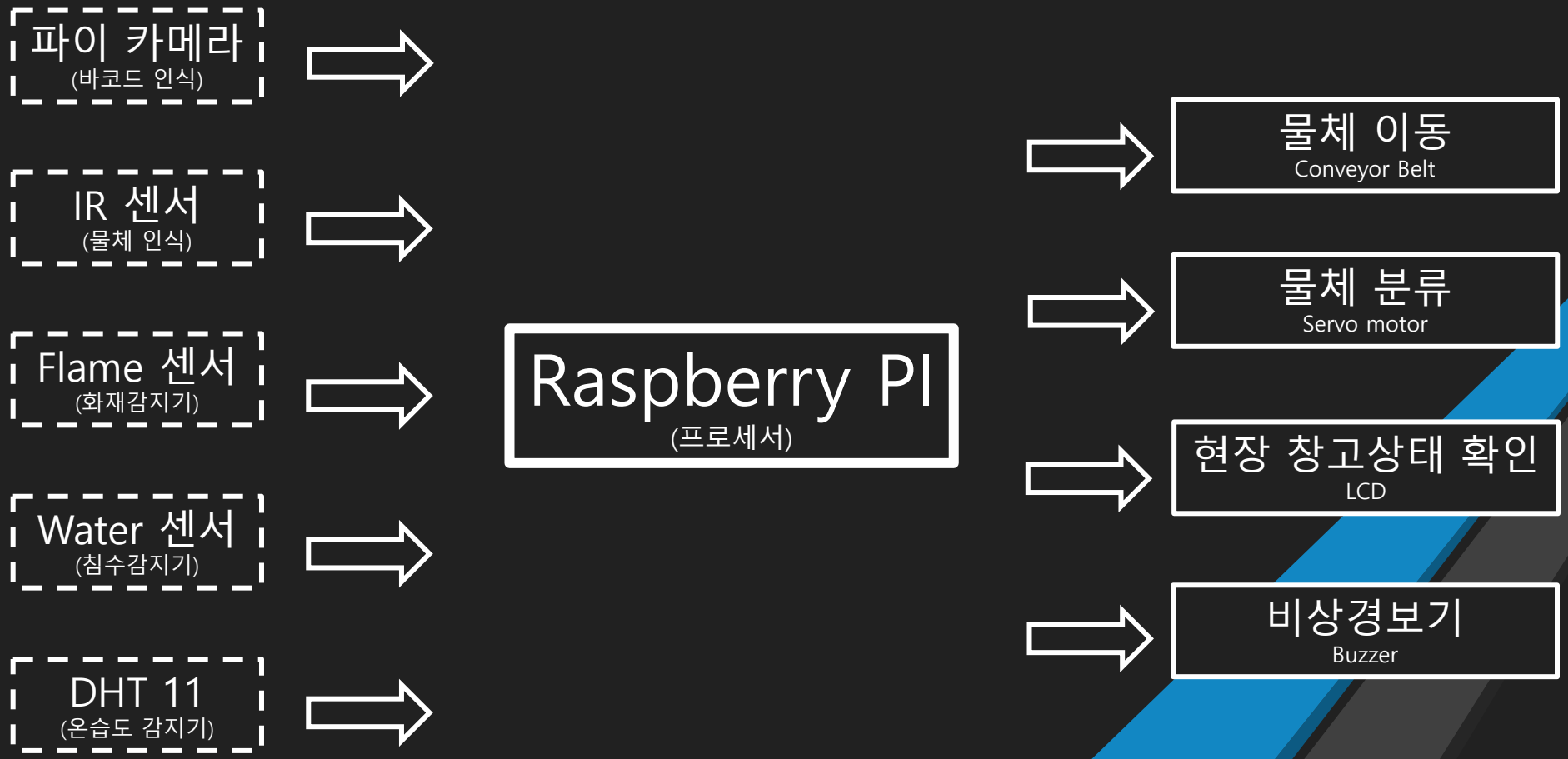
동작설명서

- 순서도

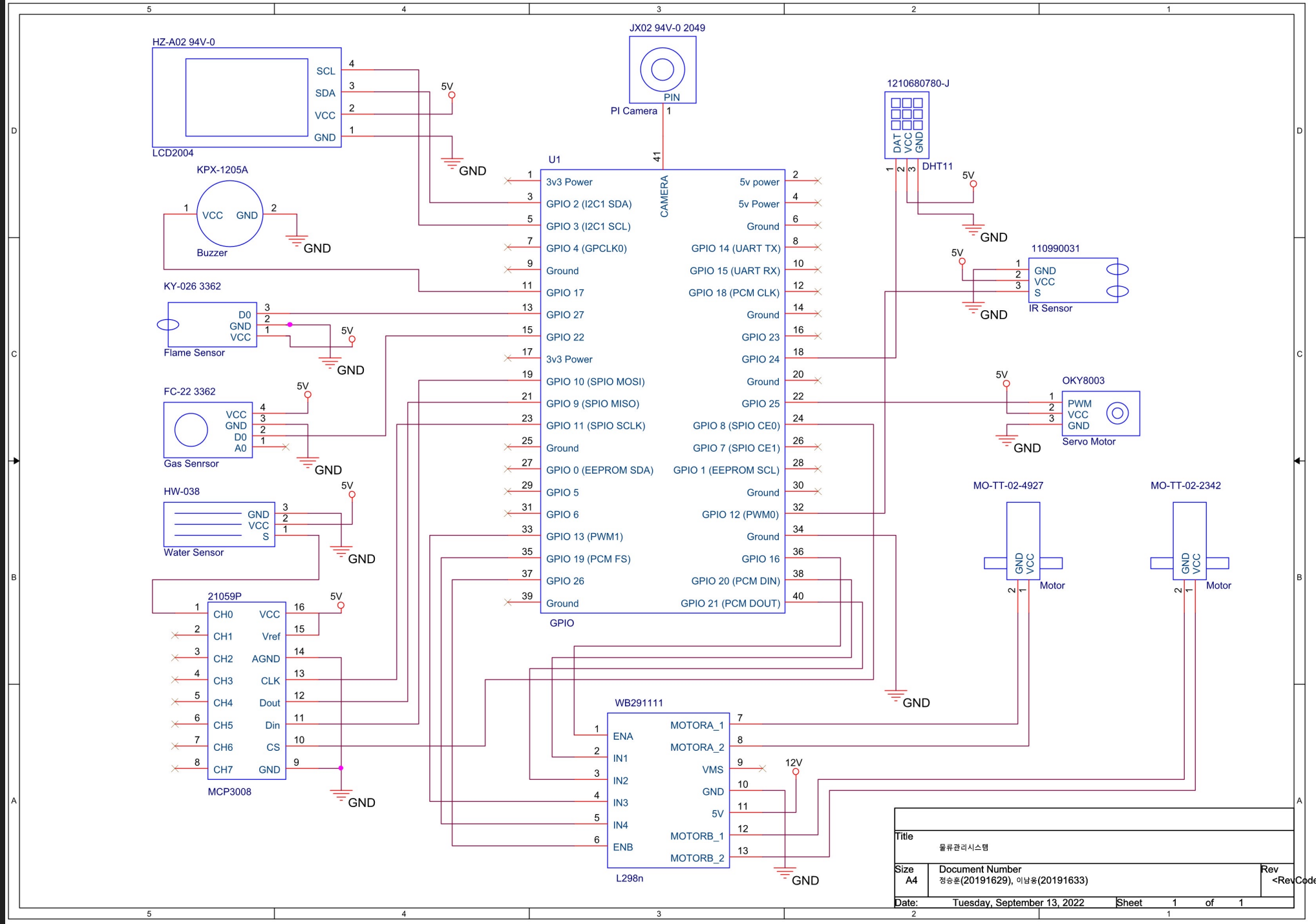

시작
PI카메라 작동
온습도 센서 상태 정보 읽어들이기
LCD에 온습도 센서 상태 정보 출력
물 감지 센서가 물을 인식했는가?
Yes
부저와 LCD, 어플을 통해 홍수 재난 상황 알림
No
불꽃 감지 센서가 불꽃을 인식했는가?
Yes
부저와 LCD, 어플을 통해 화재 재난 상황 알림
No
가스 감지 센서가 가스를 인식했는가?
Yes
부저와 LCD, 어플을 통해 가스 누출 재난 상황 알림
No
PI카메라 QR코드 포착
잘못된 QR코드인가?
Yes
부저와 LCD를 통해 잘못된 QR라고 알린다.
No
DB 상 해당 창고의 재고가 찼는가?
Yes
부저와 LCD를 통해 창고의 재고 초과 라고 알린다
No
이동시킬 사물이 초과하지 않는가?
Yes
부조와 LCD를 통해 이동시킬 사물이 없다는 것을 알린다
No
DB상의 첫번째 창고인가?
Yes
컨베이어 벨트 작동
IR센서가 사물을 인식했는가?
Yes
서브 모터가 첫번째 창고로 분류
No
No
DB상의 두번째 창고인가?
Yes
컨베이어 벨트 작동
IR센서가 사물을 인식했는가?
Yes
서브 모터가 두번째 창고로 분류
No
No
DB상의 세번째 창고인가?
Yes
컨베이어 벨트 작동
IR센서가 사물을 인식했는가?
Yes
서브 모터가 세번째 창고로 분류
No
No

동작설명서

- 블록도

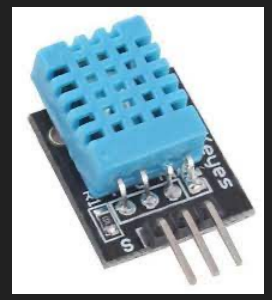

파이 카메라
(바코드 인식)
IR 센서
(물체 인식)
Flame 센서
(화재감지기)
Water 센서
(침수감지기)
DHT 11
(온습도 감지기)
Raspberry PI
(프로세서)
물체 이동
Conveyor Belt
물체 분류
Servo motor
현장 창고상태 확인
LCD
비상경보기
Buzzer

회로도


HZ-A02 94V-0
SCL
SDA
VCC
GND
LCD2004
5V
GND
JX02 94V-0 2049
PIN
PI Camera
1210680780-J
DAT
VCC
GND
DHT11
KPX-1205A
VCC
GND
Buzzer
U1
CAMERA
41
3v3 Power
GPIO 2 (I2C1 SDA)
GPIO 3 (I2C1 SCL)
GPIO 4 (GPCLK0)
Ground
GPIO 17
GPIO 27
GPIO 22
3v3 Power
GPIO 10 (SPIO MOSI)
GPIO 9 (SPIO MISO)
GPIO 11 (SPIO SCLK)
Ground
GPIO 0 (EEPROM SDA)
GPIO 5
GPIO 6
GPIO 13 (PWM1)
GPIO 19 (PCM FS)
GPIO 26
Ground
5v power
5v Power
Ground
GPIO 14 (UART TX)
GPIO 15 (UART RX)
GPIO 18 (PCM CLK)
Ground
GPIO 23
GPIO 24
Ground
GPIO 25
GPIO 8 (SPIO CE0)
GPIO 7 (SPIO CE1)
GPIO 1 (EEPROM SCL)
Ground
GPIO 12 (PWM0)
Ground
GPIO 16
GPIO 20 (PCM DIN)
GPIO 21 (PCM DOUT)
GPIO
110990031
GND
VCC
S
IR Sensor
KY-026 3362
D0
GND
VCC
Flame Sensor
FC-22 3362
VCC
GND
D0
A0
Gas Senrsor
OKY8003
PWM
VCC
GND
Servo Motor
HW-038
GND
VCC
S
Water Sensor
MO-TT-02-4927
MO-TT-02-2342
GND
VCC
Motor
21059P
CH0
CH1
CH2
CH3
CH4
CH5
CH6
CH7
VCC
Vref
AGND
CLK
Dout
Din
CS
GND
MCP3008
WB291111
ENA
IN1
IN2
IN3
IN4
ENB
MOTORA_1
MOTORA_2
VMS
GND
5V
MOTORB_1
MOTORB_2
L298n
12V
Title
물류관리시스템
Size
A4
Document Number
정승훈(20191629), 이남웅(20191633)
Rev
<RevCode>
Date:
Tuesday, September 13, 2022
Sheet
1
of
1

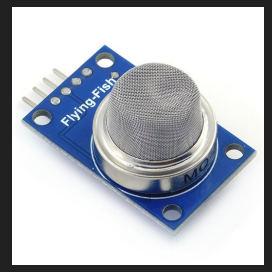
부품 소개

Pi Camera : QR 코드를 인식함

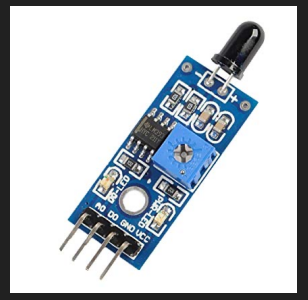


DHT11 Sensor : 스토리지 온습도를 측정

Water Sensor : 물이 닿으면 센서가 작동 (침수 방지)



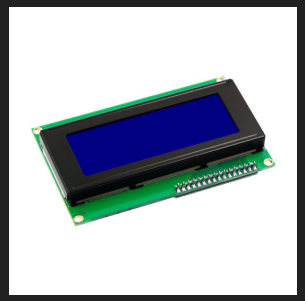
Gas Sensor : 가스가 근처에 발생할 경우 작동



Flame Sensor : 불이 근처에 있을 경우 센서가 작동 (화재 방지)

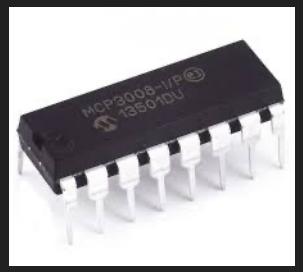
Buzzer : 재난 감지 센서 작동 시 경보를 울림

부품 소개

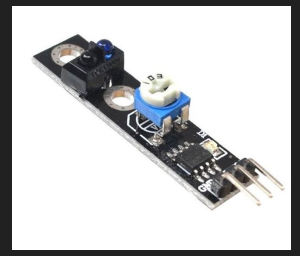


LCD2004 : 평시 스토리지 적재량과 온습도를 알려줌
재난 발생 시 경고 문구 알림

TT Motor : 컨베이어 벨트를 조작하기 위해
양 끝단에 설치



MCP3008 : Water Sensor 와 같은 아날로그 신호를 받는
센서를 디지털 신호로 변환하기 위함



IR Sensor : 컨베이어 벨트 위 물체를 인식

Servo Motor : 물체를 스토리지 별로 분류하기 위함

시작 코드

```
try:                                                                                                    #오류가 있어도 실행
motor_stop()                                                                                            #motor 초기화
main_p=Process(target=main())                                                                           #멀티프로세스 메인 함수 선언
main_p.start()                                                                                          #멀티프로레스 메인 함수 시작
except KeyboardInterrupt:                                                                               #만약 ctrl+c로 종료할 경우
GPIO.cleanup()                                                                                          #GPIO 초기화
cv2.destroyAllWindows()                                                                                 #모든 창 닫기
lcd.clear()                                                                                             #lcd 초기화
spi.close()                                                                                             #spi 초기화
print("------------------------------------------------₩nSystem Shutdown₩n------------------------------------------------")   #종료 상태 출력
sys.exit(0)                                                                                             #시스템 종료
```

메인 코드_(바코드 인식 코드)

```
def main():                                                    #메인 함수 선언
motor_state = 0                                                #motor 초기화
while(cap.isOpened()):                                         #Pi카메라가 작동하는 한 무한반복
ret, img = cap.read()                                          #Pi카메라 값 불러오기
if not ret:                                                    #만약 ret값이 없는 경우
continue                                                       #그래도 계속 진행
gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)                   #회색 상태로 바코드 읽기
decoded = pyzbar.decode(gray)                #회색으로 디코드
flood_state = (flood() > 100)                #water sensor 값이 100인 경우 참 아닌 경우 거짓
flame_state = GPIO.input(flame_pin)          #flame sensor 상태 저장
gas_state = GPIO.input(gas_pin)              #gas sensor 상태 저장
state_check_p=Process(target=state_check(0, 0, flood_state, flame_state, gas_state))
                                             #멀티프로세스 시작할 상태 확인 함수 지정
state_check_p.start()                        #멀티프로세스 상태 확인 함수 시작
motor_action_p=Process(target=motor_action(motor_state)) #멀티프로세스 모터 작동 함수 지정
motor_action_p.start()                       #멀티프로세스 모터 작동 함수 시작
for d in decoded:                            #디코드
x, y, w, h = d.rect                          #x, y, w, h 값 지정
if not((flood_state)) and flame_state and gas_state:           #재난 상태가 아닌 경우
if (d.data.decode('utf-8') == object_list[0]) and (object_num[1] < 3) and (object_num[0] <
5):                                                            #만약 첫번째 창고 QR코드인 경우
object_num[1] += 1                                             #첫번째 창고 재고 추가
iden_obj = int(d.data.decode('utf-8'))                         #임시 사물 상태 저장
print_text = ' First Storage '                                 #출력 메세지 저장
object_num[0] = object_num[1] + object_num[2] + object_num[3]
                                                               #모든 창고 재고 계산
state_check_p=Process(target=state_check(1, print_text, 0, 1, 1))
                                                               #멀티프로세스 상태 확인 함수 선언
state_check_p.start()                                          #멀티프로세스 상태 확인 함수 시작
motor_state = 1                                                #모터 작동
cap.release()                                                  #카메라 초기화
time.sleep(1.5)                                                #1.5초간 딜레이
cap.open(0)                                                    #카메라 시작
elif (d.data.decode('utf-8') == object_list[1]) and (object_num[2] < 3) and (object_num[0] <
5):                                                            #만약 두번째 창고 QR코드인 경우
object_num[2] += 1                                             #두번재 창고 재고 추가
iden_obj = int(d.data.decode('utf-8'))                         #임시 사물 상태 저장
print_text = ' Second Storage '                                #출력 메세지 저장
object_num[0] = object_num[1] + object_num[2] + object_num[3]
                                                               #모든 창고 재고 계산
state_check_p=Process(target=state_check(1, print_text, 0, 1, 1))
                                                               #멀티프로세스 상태 확인 함수 선언
state_check_p.start()                                          #멀티프로세스 상태 확인 함수 시작
motor_state = 1                                                #모터 작동
cap.release()                                                  #카메라 초기화
time.sleep(1.5)                                                #1.5초간 딜레이
cap.open(0)                                                    #카메라 시작
elif (d.data.decode('utf-8') == object_list[2]) and (object_num[3] < 3) and (object_num[0] <
5):                                                            #만약 세번째 창고 QR코드인 경우
object_num[3] += 1                                             #세번째 창고 재고 추가
iden_obj = int(d.data.decode('utf-8'))                         #임시 사물 상태 저장
print_text = ' Third Storage '                                 #출력 메세지 저장
object_num[0] = object_num[1] + object_num[2] + object_num[3]
                                                               #모든 창고 재고 계산
state_check_p=Process(target=state_check(1, print_text, 0, 1, 1))
                                                               #멀티프로세스 상태 확인 함수 선언
state_check_p.start()                                          #멀티프로세스 상태 확인 함수 시작
motor_state = 1                                                #모터 작동
cap.release()                                                  #카메라 초기화
time.sleep(1.5)                                                #1.5초간 딜레이
cap.open(0)                                                    #카메라 시작
elif ((d.data.decode('utf-8') == object_list[0]) or (d.data.decode('utf-8') == object_list[1]) or
(d.data.decode('utf-8') == object_list[2])) and ₩
(object_num[0] == 5):                                          #만약 사물이 초과된 경우
print_text = ' Object Over '                                   #출력 메세지 저장
object_num[0] = object_num[1] + object_num[2] + object_num[3]
                                                               #모든 창고 재고 계산
state_check_p=Process(target=state_check(2, print_text, 0, 1, 1))
                                                               #멀티프로세스 상태 확인 함수 선언
state_check_p.start()                                          #멀티프로세스 상태 확인 함수 시작
cap.release()                                                  #카메라 초기화
time.sleep(1.5)                                                #1.5초간 딜레이
cap.open(0)                                                    #카메라 시작
elif ((d.data.decode('utf-8') == object_list[0]) or (d.data.decode('utf-8') == object_list[1]) or
(d.data.decode('utf-8') == object_list[2])) and ₩
((object_num[1] == 3) or (object_num[2] == 3) or (object_num[3] == 3)):
                                                               #만약 창고에 사물이 가득 찬 경우
print_text = ' Storage is full '                               #출력 메세지 저장
object_num[0] = object_num[1] + object_num[2] + object_num[3]
                                                               #모든 창고 재고 계산
state_check_p=Process(target=state_check(2, print_text, 0, 1, 1))
                                                               #멀티프로세스 상태 확인 함수 선언
state_check_p.start()                                          #멀티프로세스 상태 확인 함수 시작
cap.release()                                                  #카메라 초기화
time.sleep(1.5)                                                #1.5초간 딜레이
cap.open(0)                                                    #카메라 시작
else:                                                          #만약 잘못된 바코드인 경우
print_text = ' Wrong QR code '                                 #출력 메세지 저장
object_num[0] = object_num[1] + object_num[2] + object_num[3]
                                                               #모든 창고 재고 계산
state_check_p=Process(target=state_check(2, print_text, 0, 1, 1))
                                                               #멀티프로세스 상태 확인 함수 선언
state_check_p.start()                                          #멀티프로세스 상태 확인 함수 시작
cap.release()                                                  #카메라 초기화
time.sleep(1.5)                                                #1.5초간 딜레이
cap.open(0)                                                    #카메라 시작
```

상태 체크 코드_(재난 상황, 재고 관리)

```
def state_check(state_mode, print_text, flood_state, flame_state, gas_state):                  #상태 체크 함수 선언
global dht11, dht11_state, motor_state                                                         #전역변수 dht11, dht11 상태, motor 상태 선언
if flood_state and flame_state and gas_state:                                                  #만약 홍수 상태인 경우
crisis_string[1] = '###### FLOOD #######'                                                      #2번째 줄 홍수 상태 저장
crisis_string[2] = '##### OUTBREAK #####'                                                      #3번째 줄 홍수 상태 저장
elif (not(flood_state)) and (not(flame_state)) and gas_state:                                  #만약 불꽃 발생 상태인 경우
crisis_string[1] = '###### FLAME #######'                                                      #2번째 줄 불꽃 발생 저장
crisis_string[2] = '##### OUTBREAK #####'                                                      #3번째 줄 불꽃 발생 저장
elif (not(flood_state)) and flame_state and (not(gas_state)):                                  #만약 가스 누출 발생 상태인 경우
crisis_string[1] = '####### GAS ########'                                                      #2번째 줄 가스 누출 발생 저장
crisis_string[2] = '####### LEAK #######'                                                      #3번째 줄 가스 누출 발생 저장
elif flood_state and (not(flame_state)) and gas_state:                                         #만약 홍수, 불꽃 발생 상태인 경우
crisis_string[1] = '### FLOOD FLAME ####'                                                      #2번째 줄 홍수, 불꽃 발생 저장
crisis_string[2] = '##### OUTBREAK #####'                                                      #3번째 줄 홍수, 불꽃 발생 저장
elif flood_state and flame_state and (not(gas_state)):                                         #만약 홍수, 가스 누출 상태인 경우
crisis_string[1] = '## FLOOD OUTBREAK ##'                                                      #2번째 줄 홍수, 가스 누출 발생 저장
crisis_string[2] = '##### GAS LEAK #####'                                                      #3번째 줄 홍수, 가스 누출 발생 저장
elif (not(flood_state)) and (not(flame_state)) and (not(gas_state)):                           #만약 불꽃 발생, 가스 누출인 경우
crisis_string[1] = '## FLAME OUTBREAK ##'                                                      #2번째 줄 불꽃 발생, 가스 누출 저장
crisis_string[2] = '##### GAS LEAK #####'                                                      #3번째 줄 불꽃 발생, 가스 누출 저장
elif flood_state and (not(flame_state)) and (not(gas_state)):                                  #만약 모든 재난 상태인 경우
crisis_string[1] = 'FLOOD FLAME OUTBREAK'                                                      #2번째 줄 모든 재난 저장
crisis_string[2] = '##### GAS LEAK #####'                                                      #3번째 줄 모든 재난 저장
else:                                                                                          #재난 상태가 아니인 경우
if state_mode == 0 and (dht11.is_valid() and ((dht11_state[0] != dht11.temperature) or (dht11_state[1]
!= dht11.humidity))):                                                                          #만약 온습도가 갱신될 때
crisis_string[0] = ' Camera is Ready '                                                         #1번째 줄 카메라 준비 상태 저장
crisis_string[1] = '1st St: %d 2st St: %d ' %(object_num[1], object_num[2])
                                                                                               #2번째 줄 1, 2번째 재고 상태 저장
crisis_string[2] = '3st St: %d All St: %d ' %(object_num[3], object_num[0])
                                                                                               #2번째 줄 3, 4번째 재고 상태 저장
crisis_string[3] = 'Tem: %-3.1f°C Hum: %d%%' % (dht11.temperature, dht11.humidity)
                                                                                               #4번째 줄 온습도 상태 저장
dht11_state[0] = dht11.temperature                                                             #dht11 온도 저장
dht11_state[1] = dht11.humidity                                                                #dht11 습도 저장
state_print(state_mode, crisis_string)                                                         #상태 출력
elif state_mode == 0 and crisis_string[0] != ' Camera is Ready ':                              #만약 창고 재고가 변경될 때
crisis_string[0] = ' Camera is Ready '                                                         #1번째 줄 카메라 준비 상태 저장
crisis_string[1] = '1st St: %d 2st St: %d ' %(object_num[1], object_num[2])
                                                                                               #2번째 줄 1, 2번째 재고 상태 저장
crisis_string[2] = '3st St: %d All St: %d ' %(object_num[3], object_num[0])
                                                                                               #3번째 줄 3, 4번째 재고 상태 저장
dht11_state[0] = dht11.temperature                                                             #4번째 줄 온습도 상태 저장
dht11_state[1] = dht11.humidity                                                                #dht11 온도 저장
state_print(state_mode, crisis_string)                                                         #dht 습도 저장
elif state_mode > 0:                                                                           #상태 출력
crisis_string[0] = str(print_text)                                                             #만약 QR코드가 인식 되었을 때
crisis_string[1] = '1st St: %d 2st St: %d ' %(object_num[1], object_num[2])                    #인식된 QR코드 상태 저장
                                                                                               #2번째 줄 1, 2번째 재고 상태 저장
crisis_string[2] = '3st St: %d All St: %d ' %(object_num[3], object_num[0])
                                                                                               #3번째 줄 3, 4번째 재고 상태 저장
if dht11.is_valid() and ((dht11_state[0] != dht11.temperature) or (dht11_state[1] != dht11.humidity)):
                                                                                               #만약 온습도가 갱신될 때
crisis_string[3] = 'Tem: %-3.1f°C Hum: %d%%' % (dht11.temperature, dht11.humidity)
                                                                                               #4번째 줄 온습도 상태 저장
dht11_state[0] = dht11.temperature                                                             #dht11 온도 저장
dht11_state[1] = dht11.humidity                                                                #dht11 습도 저장
state_print(state_mode, crisis_string)                                                         #상태 출력
buzzer(state_mode)                                                                             #buzzer 작동
if flood_state or (not(flame_state)) or (not(gas_state)):                                      #만약 재난 상태일 경우
crisis_string[0] = '########## WARNING #'                                                      #1번째 줄 경고 상태 저장
crisis_string[3] = '# WARNING ##########'                                                      #4번째 줄 경고 상태 저장
state_print(5, crisis_string)                                                                  #상태 출력
buzzer(3)                                                                                      #buzzer를 3번 울림
crisis_string[0] = '# WARNING ##########'                                                      #1번째 줄 경고 상태 저장
crisis_string[3] = '########## WARNING #'                                                      #4번째 줄 경고 상태 저장
state_print(5, crisis_string)                                                                  #상태 출력
dht11 = dht11_pin.read()                                                                       #dht11에서 온습도 읽기
time.sleep(0.25)                                                                               #0.25초간 딜레이
```

상태 출력 코드_(파이썬 출력, LCD)

```
def state_print(state_mode, crisis_string):                    #상태 출력 함수 선언
global dht11, dht11_state                                      #전연변수 dht11, dht11 상태 선언
print("------------------------------------------------")      #출력창 구분
print("%s₩n%s₩n%s₩n%s" % (crisis_string[0], crisis_string[1], crisis_string[2], crisis_string[3]))
                                                               #저장된 각 줄의 값 출력
if state_mode < 3:                                             #재난 상태가 아닌 경우
for i in range(0, 2):                                          #lcd 1, 2번째 출력
lcd.setCursor(0,i)                                             #lcd 1, 2번째 줄 커서 설정
lcd.print(crisis_string[i])                                    #lcd 1, 2번째 줄 출력
for i in range(2, 3):                                          #lcd 3, 4번째 출력
lcd.setCursor(4,i)                                             #lcd 3, 4번째 줄 커서 설정
lcd.print(crisis_string[i])                                    #lcd 3, 4번째 줄 출력
lcd.setCursor(4,3)                                             #lcd 4번째 줄 커서 설정
lcd.print("Tem: %-3.1f" % dht11_state[0])                      #lcd상 온도 출력
lcd.createChar(0, lcd_celsius_symbol)                          #lcd상 온도 섭씨 기호 출력
lcd.setCursor(13,3)                                            #lcd 4번째 줄 커서 설정
lcd.write(0)                                                   #lcd 값 출력
lcd.setCursor(14,3)                                            #lcd 4번째 줄 커서 설정
lcd.print("C Hum: %d%%" % dht11_state[1])                      #lcd상 습도 출력
else:                                                          #재난 상태인 경우
for i in range(0, 2):                                          #lcd 1, 2번째 출력
lcd.setCursor(0,i)                                             #lcd 1, 2번째 줄 커서 설정
lcd.print(crisis_string[i])                                    #lcd 1, 2번째 줄 출력
for i in range(2, 4):                                          #lcd 3, 4번째 출력
lcd.setCursor(4,i)                                             #lcd 3, 4번째 줄 커서 설정
lcd.print(crisis_string[i])                                    #lcd 3, 4번째 줄 출력
```

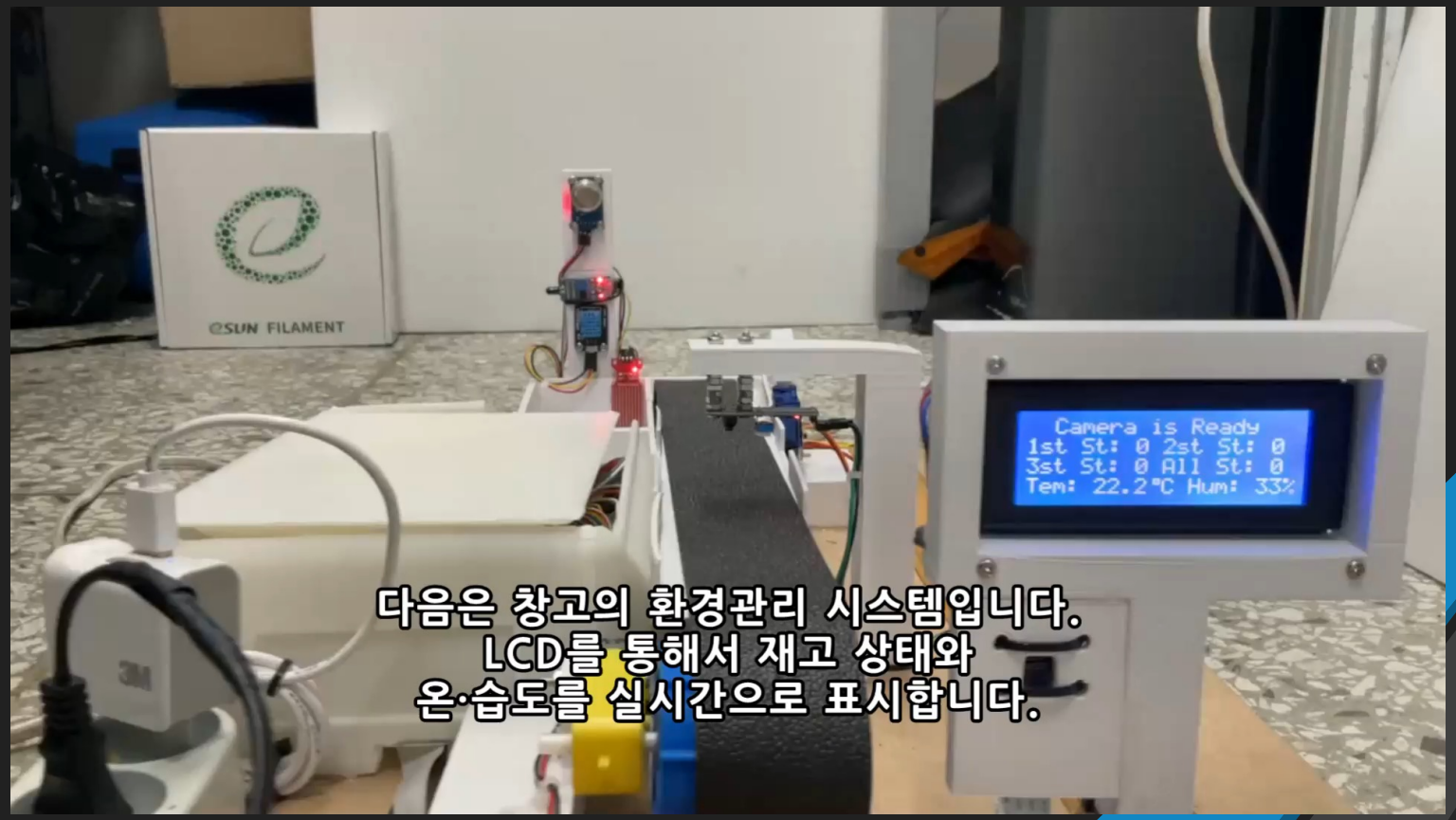
기타 함수 코드

```
def motor_start():                          #motor 시작 함수 선언
pwm1.ChangeDutyCycle(100)                   #motor1 속도 지정
pwm2.ChangeDutyCycle(100)                   #motor2 속도 지정
GPIO.output(motor1_pin[1], True)            #motor1 직진
GPIO.output(motor1_pin[2], False)
GPIO.output(motor2_pin[0], True)            #motor2 직진
GPIO.output(motor2_pin[1], False)
def motor_stop():                           #motor 정지 함수 선언
pwm1.ChangeDutyCycle(0)                     #motor1 속도 지정
pwm2.ChangeDutyCycle(0)                     #motor2 속도 지정
def first_servo():                          #첫번째 창고 이동을 위한 서브모터 동작 함수 선언
servo.ChangeDutyCycle(6.85)                 #서브모터 조정
time.sleep(3)                               #3초간 딜레이
def second_servo():                         #두번째 창고 이동을 위한 서브모터 동작 함수 선언
servo.ChangeDutyCycle(4.9)                  #서브모터 조정
time.sleep(3)                               #3초간 딜레이
def third_servo():                          #세번째 창고 이동을 위한 서브모터 동작 함수 선언
servo.ChangeDutyCycle(3.0)                  #서브모터 조정
time.sleep(3)                               #3초간 딜레이
def buzzer(buzzer_num):                     #buzzer 작동 함수 선언
for i in range(0, buzzer_num):              #buzzer를 몇번 울릴지에 따른 반복문
GPIO.output(buzzer_pin, True)               #buzzer 울림
time.sleep(0.0625)                          #0.0625초간 딜레이
GPIO.output(buzzer_pin, False)              #buzzer 정지
time.sleep(0.0625)                          #0.0625초간 딜레이
def flood():                                #water sensor 작동을 위한 함수 선언
assert 0 <= spi_channel <= 0                #water sensor spi chnnel 설정
if spi_channel:                             #water sensor가 spi channel일 경우
cbyte = 0b11000000                          #0b110000000로 세팅
else:                                       #water sensor가 spi channel이 아닌 경우
cbyte = 0b10000000                          #0b10000000 로 세팅
r = spi.xfer2([1, cbyte, 0])                #spi channel 값 받기
return ((r[1] & 31) << 6) + (r[2] >> 2)     #수위에 따른 water sensor 값 반환
```

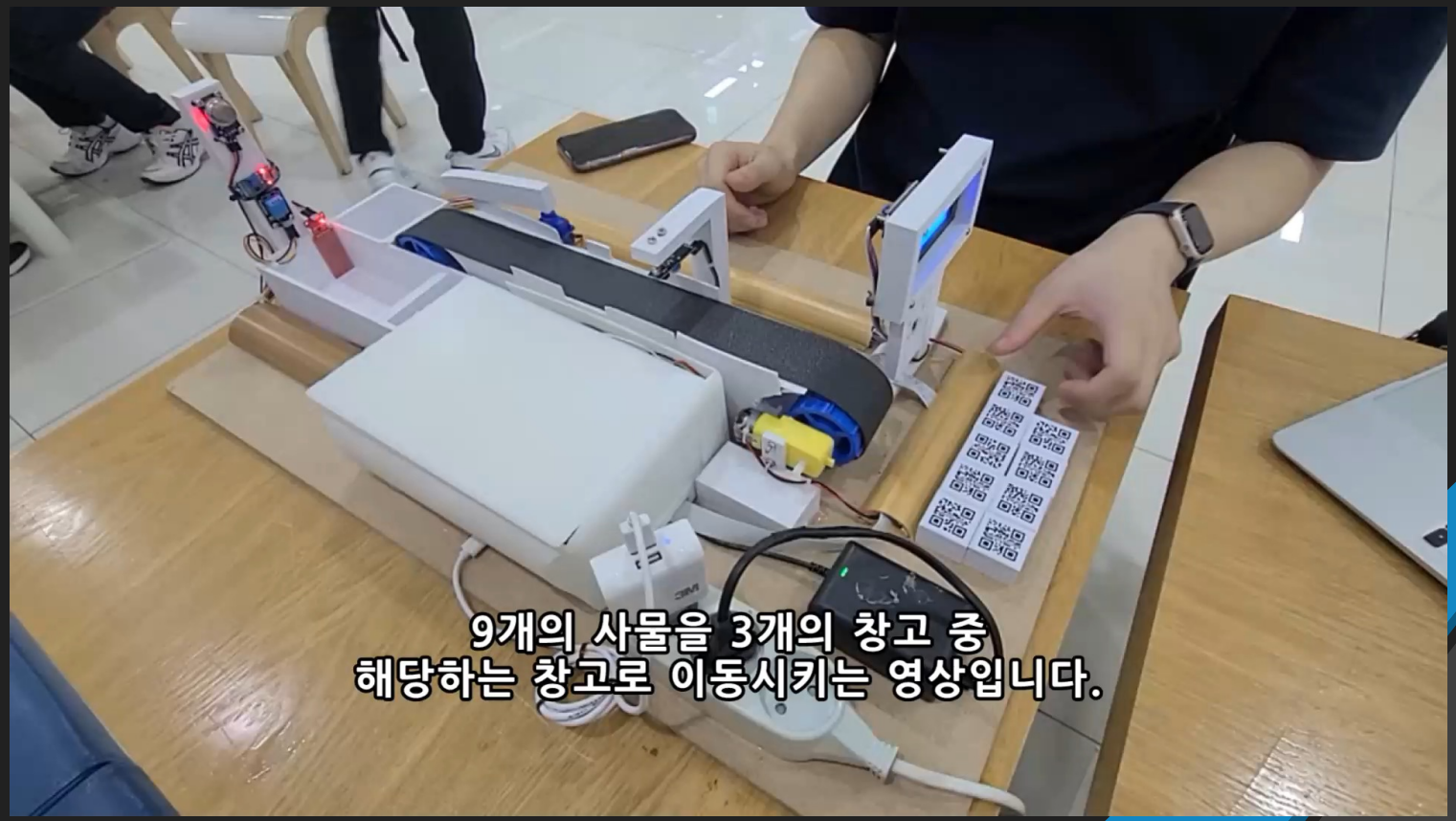
변수 선언 및 세팅값 설정 코드

```
import sys, time, RPi.GPIO as GPIO, cv2, pyzbar.pyzbar as pyzbar, dht11, spidev
from multiprocessing import Process
from RPi_I2C_LCD_driver import RPi_I2C_driver
GPIO.setmode(GPIO.BCM)                  #GPIO BCM 핀 번호 설정
GPIO.setwarnings(False)                 #GPIO 오류 알림 끄기
GPIO.cleanup()                          #GPIO 초기화
cap = cv2.VideoCapture(0)               #Pi카메라 포트 설정
motor1_pin = (16, 20, 21)               #motor1 핀 설정
motor2_pin = (13, 19, 26)               #motor2 핀 설정
GPIO.setup(motor1_pin, GPIO.OUT)        #GPIO motor1 세팅 설정
GPIO.setup(motor2_pin, GPIO.OUT)        #GPIO motor2 세팅 설정
pwm1 = GPIO.PWM(motor1_pin[0], 50)      #GPIO motor1 pwm 세팅
pwm2 = GPIO.PWM(motor2_pin[2], 50)      #GPIO motor2 pwm 세팅
pwm1.start(50)                          #motor1 pwm 시작
pwm2.start(50)                          #motor2 pwm 시작
motor_spe, motor_dur = 0, 0             #motor 스피드, 지속시간 변수 선언
motor_state = 0                         #motor 상태 변수 선언
servo_pin = (25)                        #servo motor 핀 설정
GPIO.setup(servo_pin, GPIO.OUT)         #GPIO servo motor 세팅 설정
servo = GPIO.PWM(servo_pin, 50)         #GPIO servo motor pwm 세팅
servo.start(0)                          #servo motor 시작
ir_pin = (12)                           #ir sensor 핀 설정
GPIO.setup(ir_pin, GPIO.IN)             #GPIO ir sensor 세팅 설정
ir_sta = 0                              #ir sensor 상태 변수 선언
buzzer_pin = (17)                       #buzzer 핀 설정
GPIO.setup(buzzer_pin, GPIO.OUT)        #GPIO buzzer 세팅 설정
GPIO.output(buzzer_pin, False)          #buzzer 초기화
lcd = RPi_I2C_driver.lcd(0x27)          #lcd i2c 주소 설정
lcd.clear()                             #lcd 초기화
lcd_celsius_symbol = (0b00111, 0b00101, 0b00111, 0b00000, 0b00000, 0b00000, 0b00000, 0b00000)
                                        #섭씨 이모티콘 변수 선언
crisis_string = ['', '', '', '']        #lcd 4줄 리스트 선언
state_temp = 0                          #임시 상태 변수 선언
dht11_pin = dht11.DHT11(pin = 18)       #dht11 핀 설정
dht11 = dht11_pin.read()                #dht11 읽어들이는 변수 선언
dht11_state = [0, 0]                    #dht11 온습도 변수 선언
spi = spidev.SpiDev()                   #water sensor spi 객체 선언
spi.open(0, 0)                          #water sensor 신호선 설정
spi.max_speed_hz = 250000               #water sensor 최대 주파수 설정
spi_channel = (0)                       #water sensor spi 채널 설정
flood_state = ''                        #water sensor 상태 변수 선언
flame_pin = (27)                        #flame sensor 핀 설정
GPIO.setup(flame_pin, GPIO.IN)          #GPIO flame sensor 세팅 설정
flame_state = 0                         #flame sensor 상태 변수 선언
gas_pin = (22)                          #gas sensor 핀 설정
GPIO.setup(gas_pin, GPIO.IN)            #GPIO gas sensor 세팅 설정
gas_state = 0                           #gas sensor 상태 변수 선언
object_list = ('1001', '1002', '1003')  #QR코드값 창고 3개 지정
object_num = [0, 0, 0, 0]               #모든 사물값, 첫번째 창고, 두번째 창고, 세번째 창고 개수 변수 선언
iden_obj = 0                            #임시 사물 상태 변수 선언
object_order = []                       #사물 순서 리스트 선언
print_text = ''                         #출력 메세지 변수 선언
```

작품 동작 영상


eSUN FILAMENT
Camera is Ready
1st St: 0 2st St: 0
3st St: 0 All St: 0
Tem: 22.2℃ Hum: 33%
다음은 창고의 환경관리 시스템입니다.
LCD를 통해서 재고 상태와
온·습도를 실시간으로 표시합니다.

작품 동작 영상



활용방안

- 4차 산업혁명에 맞춰서 인력중심의 물류관리 시스템에서 벗어나 적은 인력으로도 효율적인 물류관리 시스템을 구축할 수 있고, LCD나 Buzzer를 통해서 재고관리 뿐만 아니라 재난위기 상황을 확인할 수 있고 온습도에 민감한 제품까지도 효율적으로 관리 할 수 있다.