

한국교통  
대학교

COMPILERS  
& TOOLS

# Raspberry pi & Smart Camera

What is this?

1723405 전수빈



## <실제로 찍은사진을 설명해주는 카메라>

라즈베리 파이에 운영되는 리눅스  
운영체제를 바탕으로

파이썬 프로그램을 구현하여

Cognitive Services를 통해 즉각적  
으로 사진을 분석한다 .

이 결과를 led디스플레이에 렌더링  
한다.

# production environment

## Raspberry pi & Smart Camera

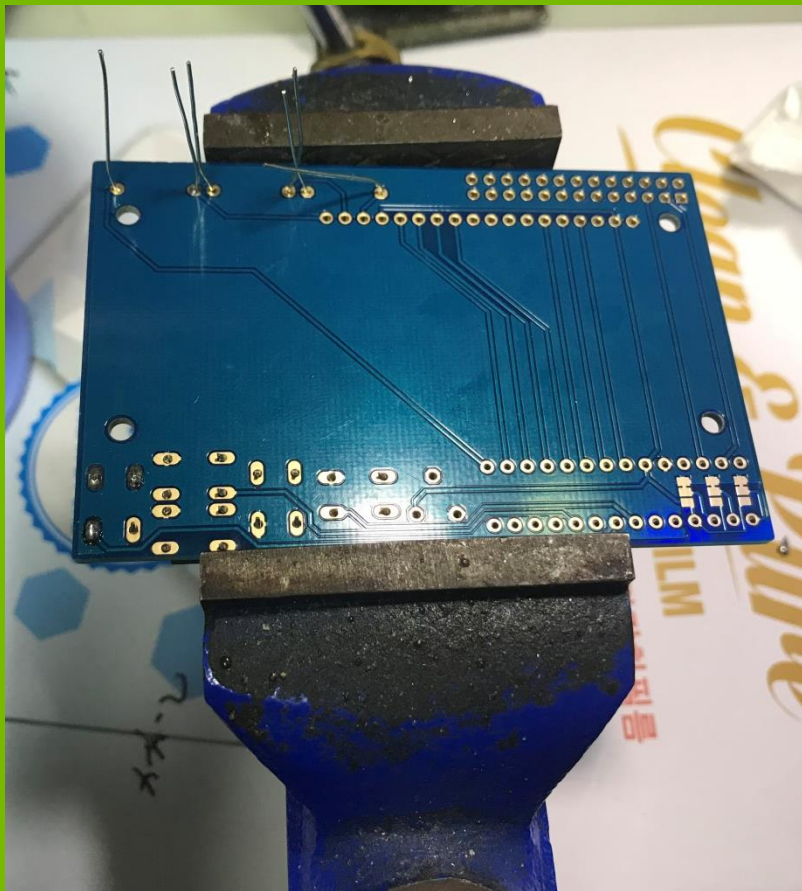
System	Raspberry Pi 3 Model B		
O . S	Raspbian Linux		
사용언어	Python		
Tool	NOOBS	Python IDE	Microsoft Azure Cognitive Services
기 타	인두	카메라모듈	RGB Negative16x2LCD

**Hardware**



- 라즈베리 파이3 모델B
  - 전원어댑터
  - SD카드
  - 카드리더기
  - HDMI케이블
  - LAN케이블
  - 카메라모듈
- 
- 회로기판
  - 저항
  - 버튼
  - 헤더
  - 전위차계
  - I2C 포트 확장기 칩
  - RGB 디스플레이LCD

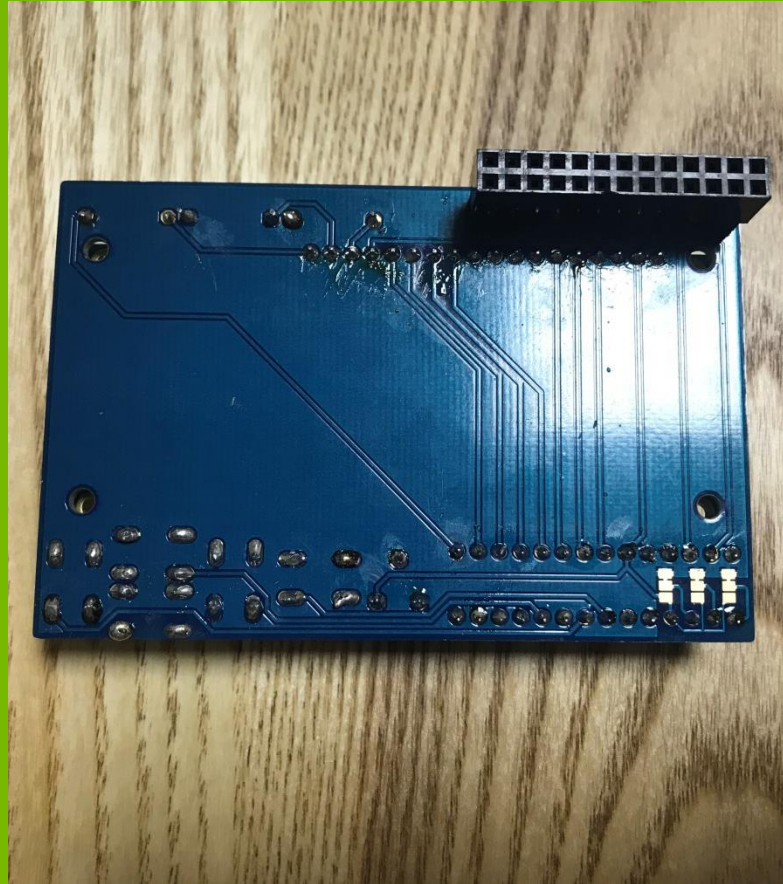
# RGB Negative16x2LCD



- GREEN저항 배치 (녹색 백라이트 핀의 백라이트 제어 저항 역할)
- RED ,BLUE 저항 배치 (LCD의 RGB 백라이트를 위한 직렬 저항)
- 버튼을 배치 (라즈베리 파이에 신호를 보냄)



# RGB Negative 16x2LCD



- 전위차계 배치 (화면의 밝기를 조정)
- I2C 포트 확장기 칩 배치 (명령을 보내고 16 개의 디지털 핀을 제어)
- 헤더 부착
- 스틱, LCD를 배치

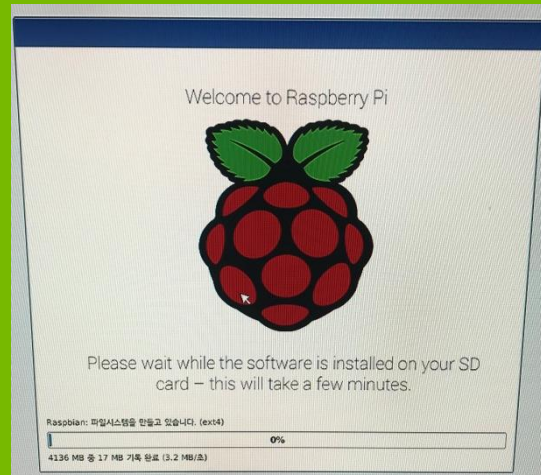
# Raspbian Linux install



“

- sd 카드에 NOOBS를 다운
- 전원케이블, LAN케이블, HDMI케이블 연결
- 라즈베리 파이에 sd카드 삽입하여 Raspbian Linux 설치

”





**python dependencies install**

# python dependencies install

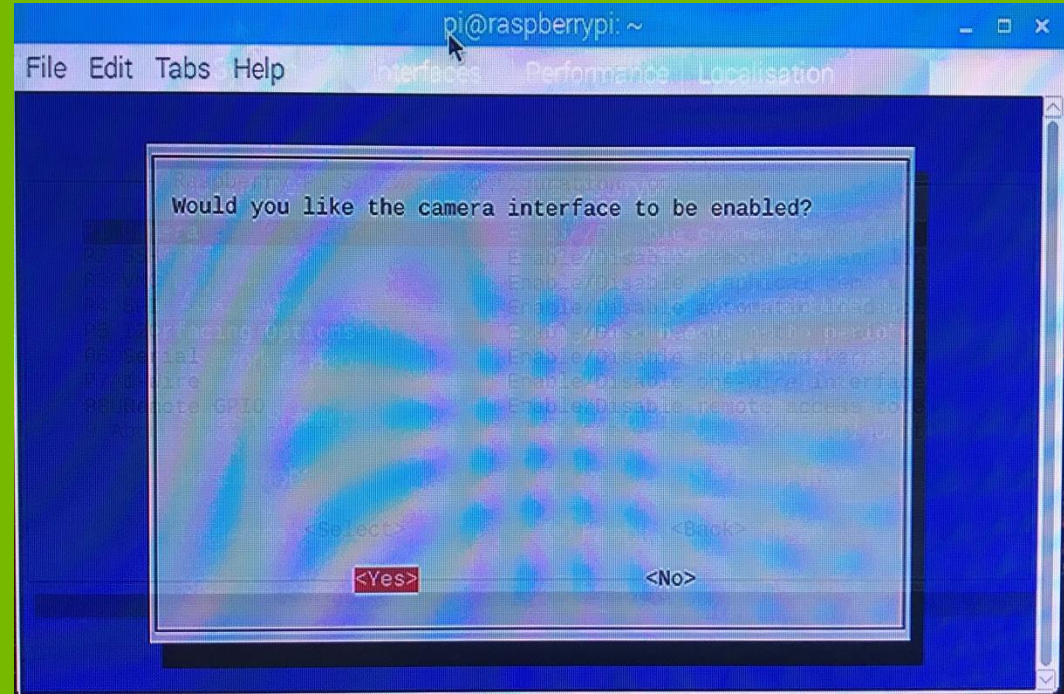
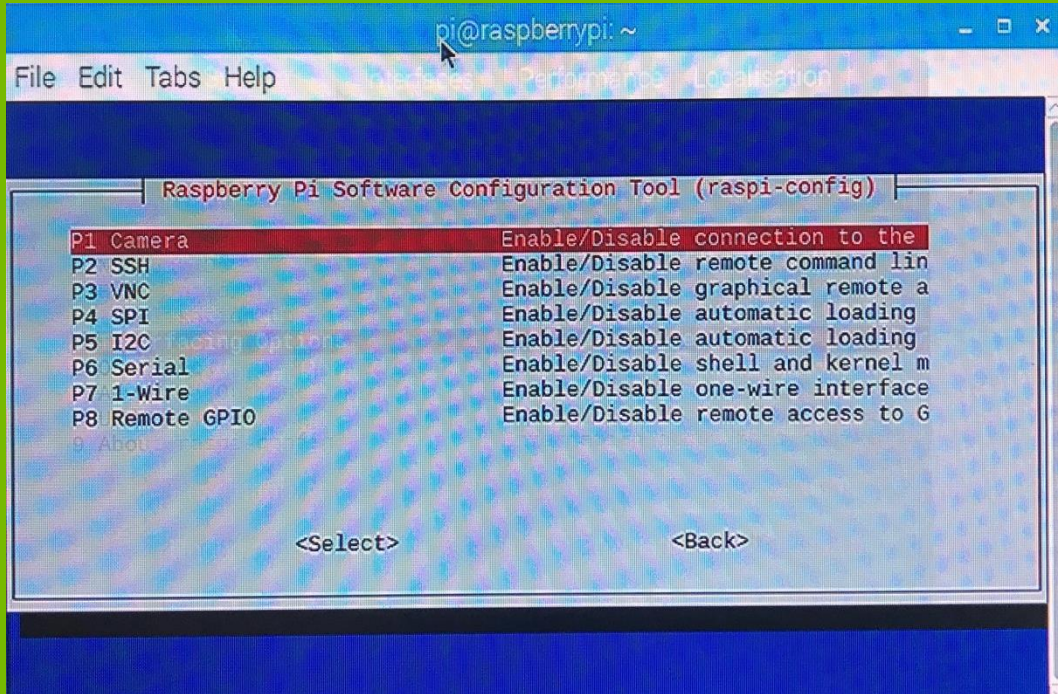
## Configuring I2C

```
pi@raspberrypi: ~  
File Edit Tabs Help  
pi@raspberrypi:~ $  
pi@raspberrypi:~ $ sudo apt-get install -y python-smbus  
Reading package lists... Done  
Building dependency tree  
Reading state information... Done  
python-smbus is already the newest version (3.1.2-3).  
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.  
pi@raspberrypi:~ $ sudo apt-get install -y i2c-tools  
Reading package lists... Done  
Building dependency tree  
Reading state information... Done  
i2c-tools is already the newest version (3.1.2-3).  
i2c-tools set to manually installed.  
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.  
pi@raspberrypi:~ $
```

- 여러 장치가 모듈의 점퍼 설정을 변경하여 설정할 수 있는 고유 주소가 있는 Raspberry Pi에 연결할 수 있도록 함.

# python dependencies install

## Installing Kernel Support (with Raspi-Config)





# python dependencies install

```
pi@raspberrypi: ~  
File Edit Tabs Help  
pi@raspberrypi:~$ sudo i2cdetect -y 1  
    0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f  
00:  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --  
10:  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --  
20: 20  --  --  --  --  --  --  --  --  --  --  --  --  --  --  
30:  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --  
40:  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --  
50:  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --  
60:  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --  
70:  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --  
pi@raspberrypi:~$
```

“

- 모든 주소에 대해 / dev / i2c-0 또는 / dev / i2c-1이 검색됨
- LCD Plate가 연결된 경우 0x20에 표시됩니다.

”

**Python\_CharLCD.py**

LCD\_CLEARDISPLAY = 0x01  
LCD\_RETURNHOME = 0x02  
LCD\_ENTRYMODESET = 0x04  
LCD\_DISPLAYCONTROL = 0x08  
LCD\_CURSORSHIFT = 0x10  
LCD\_FUNCTIONSET = 0x20  
LCD\_SETCGRAMADDR = 0x40  
LCD\_SETDDRAMADDR = 0x80

LCD\_ENTRYRIGHT = 0x00  
LCD\_ENTRYLEFT = 0x02  
LCD\_ENTRYSHIFTINCREMENT = 0x01  
LCD\_ENTRYSHIFTDECREMENT = 0x00

LCD\_DISPLAYON = 0x04  
LCD\_DISPLAYOFF = 0x00  
LCD\_CURSORON = 0x02  
LCD\_CURSOROFF = 0x00  
LCD\_BLINKON = 0x01  
LCD\_BLINKOFF = 0x00

LCD\_DISPLAYMOVE = 0x08  
LCD\_CURSORMOVE = 0x00  
LCD\_MOVERIGHT = 0x04  
LCD\_MOVELEFT = 0x00

LCD\_8BITMODE = 0x10  
LCD\_4BITMODE = 0x00  
LCD\_2LINE = 0x08  
LCD\_1LINE = 0x00  
LCD\_5x10DOTS = 0x04  
LCD\_5x8DOTS = 0x00

명령, 엔트리, 제어, 이동,  
함수

백라이트 저장, 핀 출력설  
성, 디스플레이, 컨트롤, 함  
수 초기화

```
for pin in (rs, en, d4, d5, d6, d7):  
    gpio.setup(pin, GPIO.OUT)  
if backlight is not None:  
    if enable_pwm:  
        pwm.start(backlight, self._pwm_duty_cycle(initial_backlight))  
    else:  
        gpio.setup(backlight, GPIO.OUT)  
        gpio.output(backlight, self._blpol if initial_backlight else not self._blpol)  
  
self.write8(0x33)  
self.write8(0x32)  
self.displaycontrol = LCD_DISPLAYON | LCD_CURSOROFF | LCD_BLINKOFF  
self.displayfunction = LCD_4BITMODE | LCD_1LINE | LCD_2LINE | LCD_5x8DOTS  
self.displaymode = LCD_ENTRYLEFT | LCD_ENTRYSHIFTDECREMENT  
self.write8(LCD_DISPLAYCONTROL | self.displaycontrol)  
self.write8(LCD_FUNCTIONSET | self.displayfunction)  
self.write8(LCD_ENTRYMODESET | self.displaymode)  
self.clear()
```



```

def set_backlight(self, backlight):
    self.set_color(backlight, backlight, backlight)

class Adafruit_CharLCDPlate(Adafruit_RGBCharLCD):

    def __init__(self, address=0x20, busnum=I2C.get_default_bus(), cols=16, lines=2):
        self._mcp = MCP.MCP23017(address=address, busnum=busnum)
        self._mcp.setup(LCD_PLATE_RW, GPIO.OUT)
        self._mcp.output(LCD_PLATE_RW, GPIO.LOW)
        for button in (SELECT, RIGHT, DOWN, UP, LEFT):
            self._mcp.setup(button, GPIO.IN)
            self._mcp.pullup(button, True)

        super(Adafruit_CharLCDPlate, self).__init__(LCD_PLATE_RS, LCD_PLATE_EN,
            LCD_PLATE_D4, LCD_PLATE_D5, LCD_PLATE_D6, LCD_PLATE_D7, cols, lines,
            LCD_PLATE_RED, LCD_PLATE_GREEN, LCD_PLATE_BLUE, enable_pwm=False,
            gpio=self._mcp)

    def is_pressed(self, button):

        if button not in set((SELECT, RIGHT, DOWN, UP, LEFT)):
            raise ValueError('Unknown button, must be SELECT, RIGHT, DOWN, UP, or LEFT.')
        return self._mcp.input(button) == GPIO.LOW

```

```

class Adafruit_CharLCDBackpack(Adafruit_CharLCD):

    def __init__(self, address=0x20, busnum=I2C.get_default_bus(), cols=16, lines=2):

        self._mcp = MCP.MCP23008(address=address, busnum=busnum)
        super(Adafruit_CharLCDBackpack, self).__init__(LCD_BACKPACK_RS, LCD_BACKPACK_EN,
            LCD_BACKPACK_D4, LCD_BACKPACK_D5, LCD_BACKPACK_D6, LCD_BACKPACK_D7,
            cols, lines, LCD_BACKPACK_LITE, enable_pwm=False, gpio=self._mcp)

```

백라이트 활성화,

Led 초기화

버튼을 누르면 true, 그렇지 않으면 false 반환

I2C 및 SPI를 표현, 상호 작용

**ComputerVision.py**

```

from __future__ import print_function

import time
import picamera
from datetime import datetime
import requests
import operator
import numpy as np
import json
import urllib2

def processRequest( json, data, headers, params, lcd ):
    lcd.set_color(1.0, 1.0, 0.0)
    lcd.clear()
    lcd.message('Uploading...^_^')
    retries = 0
    result = None

    while True:
        response = requests.request( 'post', _url, json=_jsonObj, data=_data, headers=_headers, params=_params )
        if response.status_code == 429:
            lcd.message( "Message: %s" % ( _response.json()[ 'message' ] ) )
            if retries <= _maxNumRetries:
                time.sleep(1)
                retries += 1
                continue
            else:
                lcd.message( 'Error: failed after retrying!' )
                break
        elif response.status_code == 200 or response.status_code == 201:
            if 'content-length' in response.headers and int(response.headers[ 'content-length' ]) == 0:
                result = None
            elif 'content-type' in response.headers and isinstance(response.headers[ 'content-type' ], str):
                if 'application/json' in response.headers[ 'content-type' ].lower():
                    result = response.json() if response.content else None
                elif 'image' in response.headers[ 'content-type' ].lower():

```



```
        result = response.content
    else:
        lcd.message("Error code: %d" % (response.status_code))
        lcd.message("Message: %s" % (response.json()['message']))
    break
lcd.set_color(0.0, 1.0, 0.0)
lcd.clear()
lcd.message('Complete!')
time.sleep(1.0)
lcd.clear()
lcd.set_color(1.0, 1.0, 1.0)
return result
```

```
def renderResult (result, lcd) :
    descriptionText = result['description']['captions'][0]['text']
    if len(descriptionText) <= 16:
        lcd.message(descriptionText)
    i = 15
    while i <= len(descriptionText):
        lcd.clear()
        lcd.message(descriptionText[i-15:i])
        time.sleep(0.3)
        if lcd.is_pressed(LCD.SELECT):
            return
        if lcd.is_pressed(LCD.LEFT):
            i = 15
            continue
        if i == len(descriptionText):
            while True:
                if lcd.is_pressed(LCD.SELECT):
                    break
                if lcd.is_pressed(LCD.LEFT):
                    i = 14
                    break
```

```

        break
    i += 1

LCD = None
import Adafruit_CharLCD as LCD

_url = 'https://westcentralus.api.cognitive.microsoft.com/vision/v1.0/analyze'
_key = '40b577665f714eeebc394c4237f8fc76'
params = {'visualFeatures': 'Color, Categories, Description'}
headers = dict()
headers['Ocp-Apim-Subscription-Key'] = _key
headers['Content-Type'] = 'application/octet-stream'
jsonObj = None

lcd = LCD.Adafruit_CharLCDPlate()
lcd.set_color(1.0, 1.0, 1.0)
lcd.clear()

for i in range(1,3):
    for j in range(1,4):
        lcd.clear()
        displayMessage = 'Bootup\Win progress.'
        if j == 2:
            displayMessage += '...'
        if j == 3:
            displayMessage += '...'
        lcd.message(displayMessage)
        time.sleep(1.0)

while True:
    try:
        urllib2.urlopen("http://www.bing.com").close()
    except urllib2.URLError:
        lcd.clear()

```

## 나의 URL 과 API 입력



### Computer Vision API

이 API 키는 현재 활성 상태입니다.

7일 남음

이미지에서 실용적인 정보를 추출

5,000개의 트랜잭션, 분당 20개.

끝점

<https://westcentralus.api.cognitive.microsoft.com/vision/v1.0>

<https://westcentralus.api.cognitive.microsoft.com/vision/v2.0>

키 1: 848ed2d1f9644d4d85c0d89ff74ca6e1

키 2: 40b577665f714eeebc394c4237f8fc76

[빠른 시작 가이드 >](#)

## LED를 초기화 해줌

## 인터넷 연결 확인

```

        lcd.set_color(1.0, 1.0, 0.0)
        lcd.message("Please wait\nfor internet")
        time.sleep(1)
    else:
        lcd.clear()
        lcd.message("Connected to\nthe internet!")
        time.sleep(2)
        break

camera = picamera.PiCamera()
camera.resolution = (1920, 1080)
camera.rotation = 90
lcd.message('Take a picture!')

while True:
    if lcd.is_pressed(LCD.SELECT):
        lcd.clear()
        lcd.message('Capturing...')
        imageName = r'/home/pi/CV/image' + str(datetime.now()) + '.jpg'
        camera.capture(imageName)
        time.sleep(2.0)
        with open(imageName, 'rb') as f:
            data = f.read()
        result = processRequest(json, data, headers, params, lcd)
        if result is not None:
            renderResult(result, lcd)

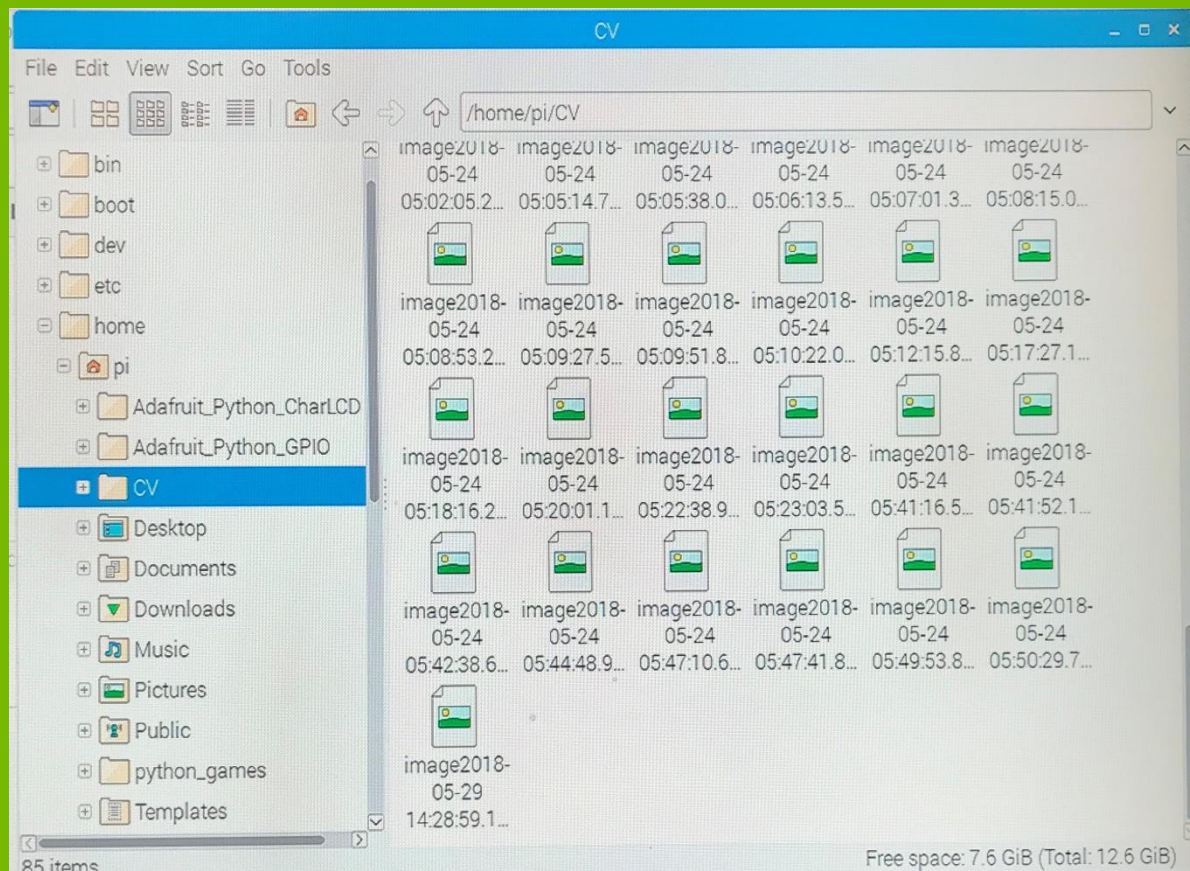
```

카메라 초기화 및 설정

사진찍는 버튼(select), 다른 버튼들의  
입력을 받음



# 모든 사진은 특정 폴더에 저장

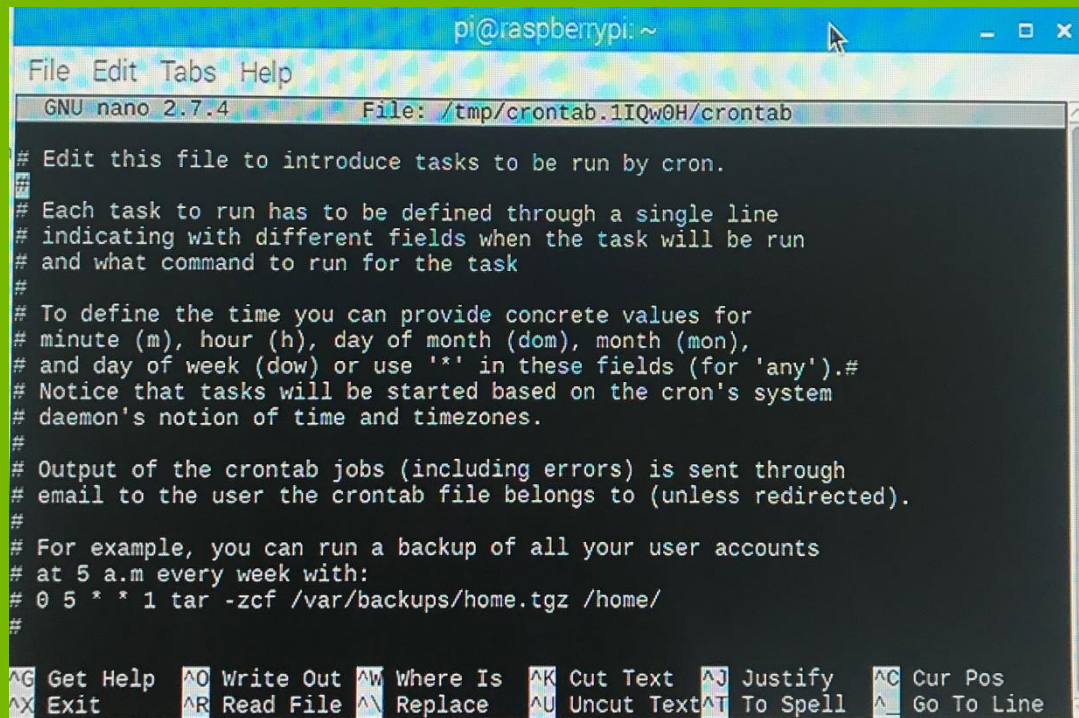


“

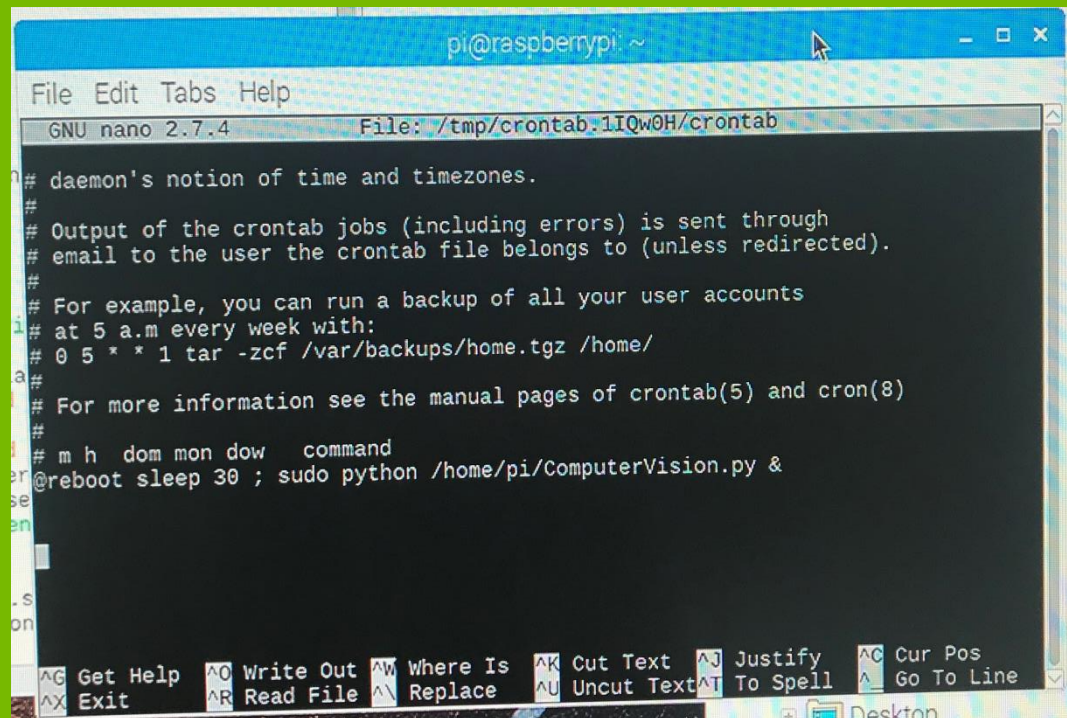
- 사용자가 지정한 폴더에 저장
- 언제든지 변경가능

”

# 바로 실행되도록 설정



```
pi@raspberrypi: ~
File Edit Tabs Help
GNU nano 2.7.4 File: /tmp/crontab.1IQw0H/crontab
# Edit this file to introduce tasks to be run by cron.
#
# Each task to run has to be defined through a single line
# indicating with different fields when the task will be run
# and what command to run for the task
#
# To define the time you can provide concrete values for
# minute (m), hour (h), day of month (dom), month (mon),
# and day of week (dow) or use '*' in these fields (for 'any').#
# Notice that tasks will be started based on the cron's system
# daemon's notion of time and timezones.
#
# Output of the crontab jobs (including errors) is sent through
# email to the user the crontab file belongs to (unless redirected).
#
# For example, you can run a backup of all your user accounts
# at 5 a.m every week with:
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
#
```

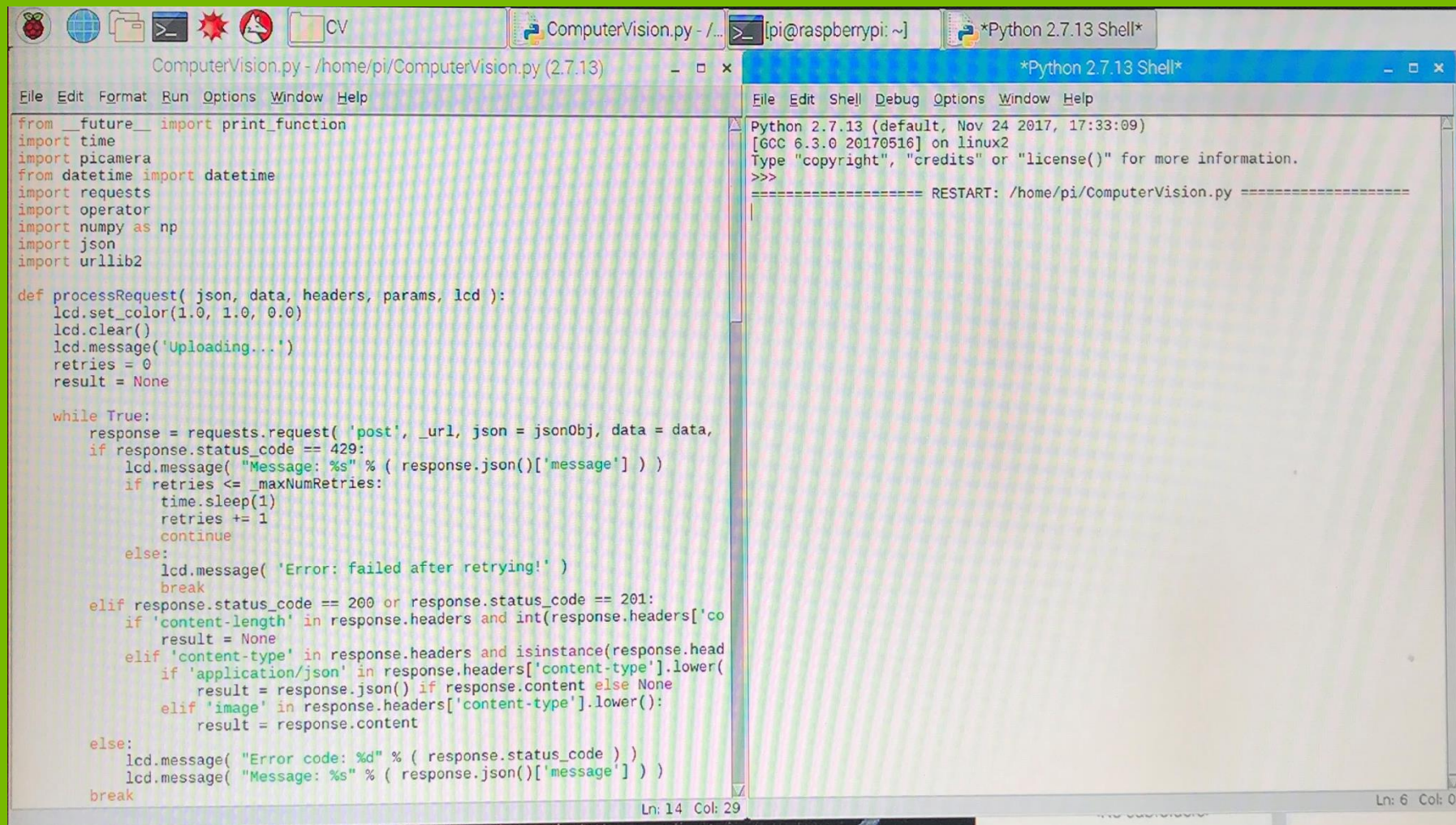


```
pi@raspberrypi: ~
File Edit Tabs Help
GNU nano 2.7.4 File: /tmp/crontab.1IQw0H/crontab
# daemon's notion of time and timezones.
#
# Output of the crontab jobs (including errors) is sent through
# email to the user the crontab file belongs to (unless redirected).
#
# For example, you can run a backup of all your user accounts
# at 5 a.m every week with:
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
#
# For more information see the manual pages of crontab(5) and cron(8)
#
# m h dom mon dow command
@reboot sleep 30 ; sudo python /home/pi/ComputerVision.py &

```



# 실행 화면



The screenshot shows a Raspberry Pi desktop environment. The top taskbar contains icons for a Raspberry Pi, a globe, a folder, a terminal, a red star, a red circle with a white 'X', and a folder named 'CV'. The main window is a code editor titled 'ComputerVision.py - /home/pi/ComputerVision.py (2.7.13)'. It contains Python code for a web server that handles image uploads. The code includes imports for time, picamera, datetime, requests, operator, numpy, json, and urllib2. The 'processRequest' function sets the LCD display color to red, clears the screen, and shows 'Uploading...'. It then makes a POST request to a server. If the status code is 429, it shows an error message and retries. If the status code is 200 or 201, it checks the content type and length, and if valid, it displays the image on the LCD. The terminal window on the right is titled '\*Python 2.7.13 Shell\*' and shows the Python version, GCC version, and the command to restart the script.

```
from __future__ import print_function
import time
import picamera
from datetime import datetime
import requests
import operator
import numpy as np
import json
import urllib2

def processRequest( json, data, headers, params, lcd ):
    lcd.set_color(1.0, 1.0, 0.0)
    lcd.clear()
    lcd.message('Uploading...')
    retries = 0
    result = None

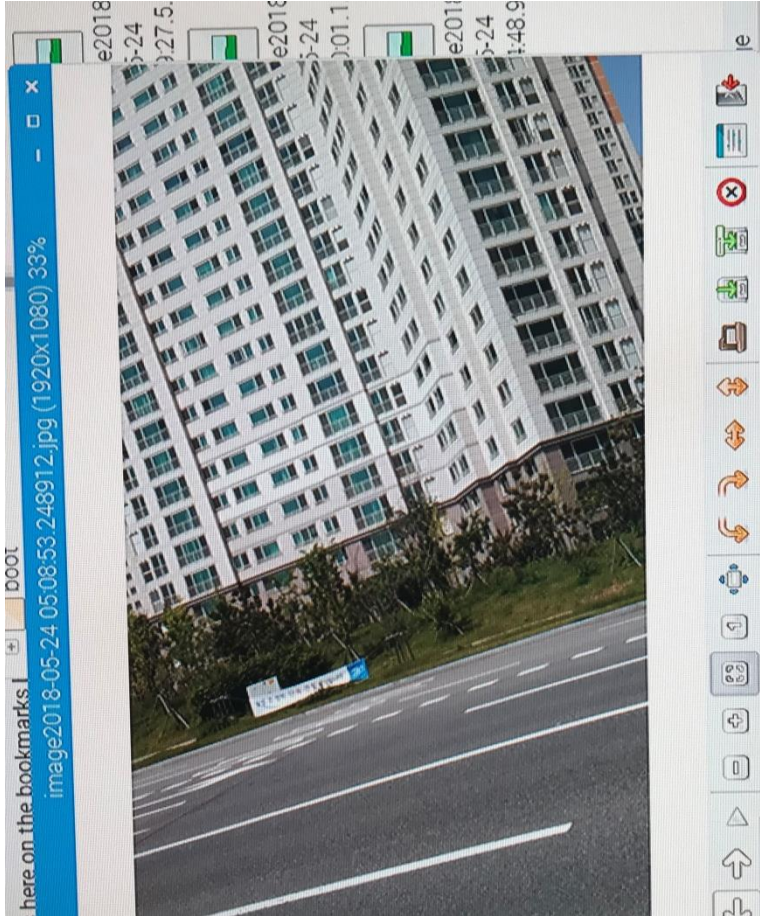
    while True:
        response = requests.request( 'post', _url, json = jsonObj, data = data,
        if response.status_code == 429:
            lcd.message( "Message: %s" % ( response.json()['message'] ) )
            if retries <= _maxNumRetries:
                time.sleep(1)
                retries += 1
                continue
            else:
                lcd.message( 'Error: failed after retrying!' )
                break
        elif response.status_code == 200 or response.status_code == 201:
            if 'content-length' in response.headers and int(response.headers['co
            result = None
            elif 'content-type' in response.headers and isinstance(response.head
            if 'application/json' in response.headers['content-type'].lower(
            result = response.json() if response.content else None
            elif 'image' in response.headers['content-type'].lower():
            result = response.content
        else:
            lcd.message( "Error code: %d" % ( response.status_code ) )
            lcd.message( "Message: %s" % ( response.json()['message'] ) )
            break
```

Ln: 14 Col: 29

```
Python 2.7.13 (default, Nov 24 2017, 17:33:09)
[GCC 6.3.0 20170516] on linux2
Type "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: /home/pi/ComputerVision.py =====
|
```

Ln: 6 Col: 0

# Result Video

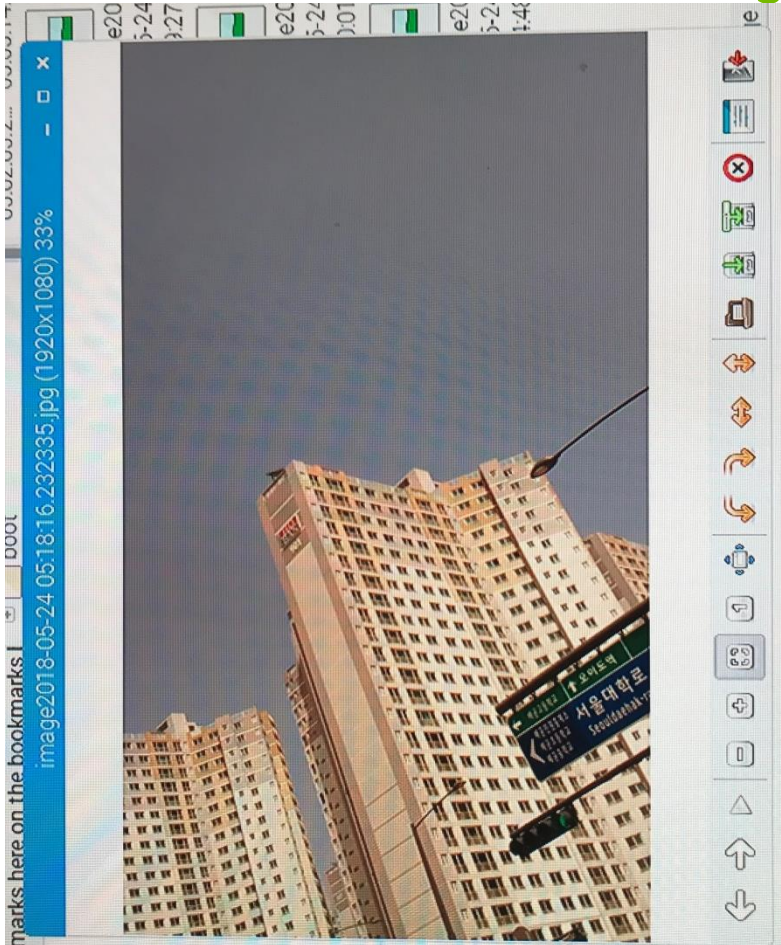


클로우즈업된?



# Result Video

A sign on the side of a building



건물사이간

# Result Video

A laptop computer



컴퓨터.

# field of usage

- 어린이들의 인지능력 향상
- 학생,성인들의 영어공부 도구
- 일반카메라
- 가장멋진! 장난감

# Reinforcement Point

- 케이스 장착
- 카메라 모듈 고정
- USB 무선 네트워크 어댑터 장착  
(간편히 휴대)