LPR2800-DL and LPR2811-DL for JSON Server example

```
from __future__ import print_function
from argparse import ArgumentParser

from flask import Flask, jsonify
import json

# Python 2.x & 3.x compatible
from distutils.log import warn as printf
#printf('Hello World!')

from PIL import Image
import base64

import cv2
import numpy as np

from OpenSSL import SSL
#context = SSL.Context(SSL.PROTOCOL_TLSv1_2) //ubunutu 16.04(python 2.7.12 NG), need python 2.7.14
#context.use_privatekey_file('key.pem')
#context.use_certificate_file('cert.pem')
context = ('cert.pem', 'key.pem')

def save(encoded_data, filename):
    #nparr = np.fromstring(encoded_data.decode('base64'), np.uint8)
    #img = cv2.imdecode(nparr, cv2.IMREAD_ANYCOLOR)
    imgdata = base64.b64decode(str(encoded_data))
    jpg_recovered = np.fromstring(imgdata, np.uint8)
    #jpg_recovered = base64.b64decode(encoded_data) #NG
    #jpg_recovered = encoded_data.decode('base64')
    #jpg_recovered = np.fromstring( jpg_recovered, np.uint8)
    img = cv2.imdecode(jpg_recovered, cv2.IMREAD_ANYCOLOR)
    return cv2.imwrite(filename, img)

app = Flask(__name__)
app.debug= True #Enable debug mode

#openssl req -x509 -newkey rsa:4096 -nodes -out cert.pem -keyout key.pem -days 365

# http://192.168.1.228:5000/1234
# https://192.168.1.228:5000/1234
@app.route("/")
def hello():
        return "Hello World!"

# Step 1. python flask_server.py
# Step 2. python SendPost2Flask_server.py
from flask import request, jsonify
#@app.route('/api/add_message/<uuid>', methods=['GET', 'POST'])
@app.route('/<uuid>', methods=['GET', 'POST'])
def add_message(uuid):

    contents = request.json

    if( contents['DataNum'] >0 ):
      data_number = contents['DataNum']#list(range(contents['DataNum']))
      printf( "data_number    :%d " , data_number )
      printf( "Device_UTCTime [%s] " ,contents['Device_UTCTime'] )
      printf( "Query_UTCTime  [%s] " ,contents['Query_UTCTime'] )
      for data_idx in range( 0, contents['DataNum']):

          printf( "data_idx ******** [%d] ******** " , data_idx )

          printf( "UTC Time         [%s] " , contents['data'][data_idx]['UTC'] )
          printf( "Local Time       [%s] " , contents['data'][data_idx]['Local'] )
          printf( "PlateID          [%s] " , contents['data'][data_idx]['PlateID'] )
          printf( "RecognizedPlateID [%s] " , contents['data'][data_idx]['RecognizedPlateID'] )
          printf( "CardCode         [%s] " , contents['data'][data_idx]['CardCode'] )
          printf( "CardBits         %d " , contents['data'][data_idx]['CardBits'] )

          #printf( "PhotoName        [%s] " , contents['base64jpeg'][data_idx]['PhotoName'] )
          if contents['base64jpeg']:
              if contents['base64jpeg'][data_idx]:
                    if contents['base64jpeg'][data_idx]['base64']:
                          imgstring = contents['base64jpeg'][data_idx]['base64']
                          imgdata = base64.b64decode(imgstring)
              else:
                    printf( "contents['base64jpeg'][%i] is NULL", data_idx  )
          else:
               printf( "contents['base64jpeg'] is NULL\n, %s \n ", contents  )
          #save( imgstring , "testfile.jpg")


    return jsonify({"uuid":uuid})

if __name__ == "__main__":
      parser = ArgumentParser()
      #parser.add_argument("pos1", help="positional argument 1")
      #parser.add_argument("n", help="repeat time", type=int)
      parser.add_argument('--ssl','-s', action='store_true', default=False, help='Choose ahq')
      #parser.add_argument("-o", "--optional-arg", help="optional argument", dest="opt", default="default")
      args = parser.parse_args()

      if args.ssl:
            #python ./flask_server.py -s
            # app.run(host="0.0.0.0",port=5000,ssl_context=('cert.pem', 'key.pem'))
            app.run(host="0.0.0.0",ssl_context=context)
      else:
            #python ./flask_server.py
            app.run(host="0.0.0.0",port=5000)
```