```
from keras_preprocessing.image import load_img, img_to_array
from keras.models import load_model
import tensorflow as tf
import keras
from flask import Flask, render_template, Response, request
import cv2
import datetime
import time
import os
import sys
import numpy as np
from threading import Thread
# csv code
import pandas as pd
read_file = pd.read_excel("C:\\Users\\admin\\IbmProject\\NutritionAPP\\book.xlsx")
read_file.to_csv("Test.csv",
                 index=None,
                 header=True)
df = pd.DataFrame(pd.read_csv("Test.csv"))
df.to_csv("Test.csv")
df = df.set_index("Food Name")


def Nutrients(Name):
    name = Name
    return(df.loc[(name), :])



global capture, rec_frame, grey, switch, neg, face, rec, out, p, d
capture = 0
grey  0
neg  0
face  0
switch  1
rec  0
# ML
# import PIL.Image
# from tensorflow.keras.utils import to_categorical
# from tensorflow.keras.preprocessing.image import load_img, img_to_array
# from tensorflow.python.keras.preprocessing.image import ImageDataGenerator
# from keras.preprocessing.image import ImageDataGenerator

# import tensorflow.compat.v2 as tf
model = keras.models.load_model('C:\\Users\\admin\\IbmProject\\NutritionAPP\\daiyan.h5')


CATEGORIES = ['Vegetable-Fruit', 'Egg', 'Bread', 'Soup', 'Seafood', 'Meat', 'vada pav',
    'Fried food', 'pizza', 'Dessert', 'Dairy product', 'Rice', 'burger', 'Noodles-Pasta']


def image(path):
    img = cv2.imread(path, cv2.IMREAD_GRAYSCALE)
    new_arr = cv2.resize(img, (60, 60))
    new_arr = np.array(new_arr)
    new_arr = new_arr.reshape(-1, 60, 60, 1)
    return new_arr


# make shots directory to save pics
try:
    os.mkdir('./shots')
except OSError as error:
    pass
```

```
# instatiate flask app
app = Flask(enamel, template_folder='./templates')

camera = cv2.VideoCapture(0)
# def Path(d):
#     a=d
#     return a


def gen_frames():  # generate frame by frame from camera
    global out, capture, rec_frame, d
    while True:
        success, frame = camera.read()
        if success:
            if(capture):
                capture = 0
                now = datetime.datetime.now()
                p = os.path.sep.join(
                    ['shots', “shot_{}.png“.format(str(now).replace(“:", ''))])
                # d=(“C:\\Users\\admin\\IbmProject\\NutritionAPP”+p)
                cv2.imwrite(p, frame)
                d   p

            try:
                ret, buffer = cv2.imencode('.jpg', cv2.flip(frame, 1))
                frame = buffer.tobytes()
                yield (b'--frame\r\n'
                       b'Content-Type: image/jpeg\r\n\r\n' + frame + b'\r\n')
            except Exception as e:
                pass

        else:
            pass


@app.route('/')
def index():
    return render_template('index.html')


@app.route('/uplod')
def uplod():
    return render_template('index.html')


@app.route('/video_feed')
def video_feed():
    return Response(gen_frames(), mimetype='multipart/x-mixed-replace; boundary=frame')


@app.route('/requests', methods=['POST', 'GET'])
def tasks():
    global switch, camera
    if request.method == 'POST':
        if request.form.get(’click')    ’Capture':
            global capture
            capture = 1

        elif request.form.get('detect') == 'Detect':
            # prediction =
model.predict([image(“C:\\Users\\admin\\IbmProject\\NutritionAPP\\download.jfif“)])

            path = os.getcwd()
            print(d)
            p = os.path.join(path, "", d)

            prediction = model.predict([image(p)])

            name = (CATEGORIES[prediction.argmax()])
            Product_name = name
            data = Nutrients(Product_name)
            return render_template('Predect.html', name=name, data=data)

        elif request.form.get('stop')    'Stop/Start' :
            if(switch == 1):
                switch   0
                camera.release()
                cv2.destroyAllWindows()

            else:
                camera = cv2.VideoCapture(0)
                switch   1

    elif request.method == 'GET':
        return render_template('index.html')
    return render_template('index.html')


if enamel == '  mains':
    app.run()

camera.release()
cv2.destroyAllWindows()
```