```
{
 "cells": [
  {
   "cell_type": "code",
   "execution_count": 22,
   "id": "f86316b7",
   "metadata": {},
   "outputs": [],
   "source": [
    "from keras.preprocessing.image import ImageDataGenerator"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 23,
   "id": "fe4bb9e0",
   "metadata": {},
   "outputs": [],
   "source": [
    "from keras.preprocessing import image"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 24,
   "id": "160934a5",
   "metadata": {},
   "outputs": [],
   "source": [
    "from flask import Flask,render_template,request"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 25,
   "id": "eab6055e",
   "metadata": {},
   "outputs": [],
   "source": [
    "import os"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 26,
   "id": "3624454f",
   "metadata": {},
   "outputs": [],
   "source": [
    "import numpy as np"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 27,
   "id": "544829f4",
   "metadata": {},
   "outputs": [],
```

```
   "source": [
    "from tensorflow.keras.models import load_model"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 28,
   "id": "df07eee4",
   "metadata": {},
   "outputs": [],
   "source": [
    "from tensorflow.keras.preprocessing import image"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 29,
   "id": "fdf179ff",
   "metadata": {},
   "outputs": [],
   "source": [
    "import requests"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 30,
   "id": "cde4b0cc",
   "metadata": {},
   "outputs": [],
   "source": [
    "app = Flask(__name__,template_folder=\"templates\")"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 31,
   "id": "585313c6",
   "metadata": {},
   "outputs": [
    {
     "name": "stdout",
     "output_type": "stream",
     "text": [
      "WARNING:tensorflow:No training configuration found in the save
file, so the model was *not* compiled. Compile it manually.\n"
     ]
    }
   ],
   "source": [
    "model=load_model('nutrition.h5')"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 32,
   "id": "e3888077",
   "metadata": {},
```

```
   "outputs": [
    {
     "name": "stdout",
     "output_type": "stream",
     "text": [
      "loaded model from disk\n"
     ]
    }
   ],
   "source": [
    "print(\"loaded model from disk\")"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 33,
   "id": "b1da8083",
   "metadata": {},
   "outputs": [],
   "source": [
    "def home():\n",
    "    return render_template('home.html')#rendering the home page\n",
    "\n",
    "\n"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 34,
   "id": "68d7dd55",
   "metadata": {},
   "outputs": [],
   "source": [
    "@app.route('/image',methods=['GET','POST'])# routes to the index
html\n",
    "def image():\n",
    "    return render_template(\"image.html\")"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 35,
   "id": "84735861",
   "metadata": {},
   "outputs": [],
   "source": [
    "\n",
    "@app.route('/predict',methods=['GET', 'POST'])# route to show the
predictions in a web UI\n",
    "def launch():\n",
    "    if request.method=='POST':\n",
    "        f=request.files[''] #requesting the file\n",
    "        basepath=os.path.dirname('__file__')#storing the file
directory\n",
    "
filepath=os.path.join(basepath,\"uploads\",f.filename)#storing the file
in uploads folder\n",
    "        f.save('')#saving the file\n",
```

```
    "        "
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 36,
   "id": "5edd7210",
   "metadata": {},
   "outputs": [],
   "source": [
    "from tensorflow.keras.models import load_model"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 37,
   "id": "a198923d",
   "metadata": {},
   "outputs": [],
   "source": [
    "from keras.preprocessing import image"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 38,
   "id": "1486f60d",
   "metadata": {},
   "outputs": [],
   "source": [
    "from tensorflow.keras.models import load_model"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 40,
   "id": "f7ef03af",
   "metadata": {},
   "outputs": [
    {
     "ename": "AttributeError",
     "evalue": "module 'keras.preprocessing.image' has no attribute
'load_img'",
     "output_type": "error",
     "traceback": [
      "\u001b[1;31m-------------------------------------------------------------
----------------------\u001b[0m",
      "\u001b[1;31mAttributeError\u001b[0m
Traceback (most recent call last)",
      "Input \u001b[1;32mIn [40]\u001b[0m, in \u001b[0;36m<cell line:
1>\u001b[1;34m()\u001b[0m\n\u001b[1;32m----> 1\u001b[0m img
\u001b[38;5;241m=\u001b[39m
\u001b[43mimage\u001b[49m\u001b[38;5;241;43m.\u001b[39;49m\u001b[43mload_
img\u001b[49m(\u001b[38;5;124mr\u001b[39m\u001b[38;5;124m\"\u001b[39m\u00
1b[38;5;124mE:\u001b[39m\u001b[38;5;124m\\\u001b[39m\u001b[38;5;124mFlask
\u001b[39m\u001b[38;5;124m\\\u001b[39m\u001b[38;5;124mSample_Images-
20221104T061454Z-
001\u001b[39m\u001b[38;5;124m\\\u001b[39m\u001b[38;5;124mSample_Images\u0
```

```
01b[39m\u001b[38;5;124m\\\u001b[39m\u001b[38;5;124mTest_Image1.jpg\u001b[
39m\u001b[38;5;124m\"\u001b[39m,target_size\u001b[38;5;241m=\u001b[39m(\u
001b[38;5;241m64\u001b[39m,\u001b[38;5;241m64\u001b[39m))
\u001b[38;5;66;03m#load and reshaping the
image\u001b[39;00m\n\u001b[0;32m      2\u001b[0m x
\u001b[38;5;241m=\u001b[39m
image\u001b[38;5;241m.\u001b[39mimg_to_array(img)\u001b[38;5;66;03m#conve
rting image to an array\u001b[39;00m\n\u001b[0;32m      3\u001b[0m x
\u001b[38;5;241m=\u001b[39m
np\u001b[38;5;241m.\u001b[39mexpand_dims(x,axis
\u001b[38;5;241m=\u001b[39m \u001b[38;5;241m0\u001b[39m)\n",
      "\u001b[1;31mAttributeError\u001b[0m: module
'keras.preprocessing.image' has no attribute 'load_img'"
     ]
    }
   ],
   "source": [
    "img = image.load_img(r\"E:\\Flask\\Sample_Images-20221104T061454Z-
001\\Sample_Images\\Test_Image1.jpg\",target_size=(64,64)) #load and
reshaping the image\n",
    "x = image.img_to_array(img)#converting image to an array\n",
    "x = np.expand_dims(x,axis = 0)\n",
    "pred = np.argmax(model.predict(x)) \n",
    "pred"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 43,
   "id": "15fc85c1",
   "metadata": {},
   "outputs": [
    {
     "ename": "NameError",
     "evalue": "name 'x' is not defined",
     "output_type": "error",
     "traceback": [
      "\u001b[1;31m---------------------------------------------------------
----------------------\u001b[0m",
      "\u001b[1;31mNameError\u001b[0m
Traceback (most recent call last)",
      "Input \u001b[1;32mIn [43]\u001b[0m, in \u001b[0;36m<cell line:
1>\u001b[1;34m()\u001b[0m\n\u001b[1;32m----> 1\u001b[0m
pred\u001b[38;5;241m=\u001b[39mnp\u001b[38;5;241m.\u001b[39margmax(model\
u001b[38;5;241m.\u001b[39mpredict(\u001b[43mx\u001b[49m),
axis\u001b[38;5;241m=\u001b[39m\u001b[38;5;241m1\u001b[39m)\n\u001b[0;32m
2\u001b[0m
\u001b[38;5;28mprint\u001b[39m(\u001b[38;5;124m\"\u001b[39m\u001b[38;5;12
4mprediction\u001b[39m\u001b[38;5;124m\"\u001b[39m,pred)\u001b[38;5;66;03
m#printing the prediction\u001b[39;00m\n\u001b[0;32m      3\u001b[0m
index\u001b[38;5;241m=\u001b[39m[\u001b[38;5;124m'\u001b[39m\u001b[38;5;1
24mAPPLES\u001b[39m\u001b[38;5;124m'\u001b[39m,\u001b[38;5;124m'\u001b[39
m\u001b[38;5;124mBANANA\u001b[39m\u001b[38;5;124m'\u001b[39m,\u001b[38;5;
124m'\u001b[39m\u001b[38;5;124mORANGE\u001b[39m\u001b[38;5;124m'\u001b[39
m,\u001b[38;5;124m'\u001b[39m\u001b[38;5;124mPINEAPPLE\u001b[39m\u001b[38
;5;124m'\u001b[39m,\u001b[38;5;124m'\u001b[39m\u001b[38;5;124mWATERMELON\
u001b[39m\u001b[38;5;124m'\u001b[39m]\n",
      "\u001b[1;31mNameError\u001b[0m: name 'x' is not defined"
```

```
     ]
     }
    ],
    "source": [
     " pred=np.argmax(model.predict(x), axis=1)\n",
     " print(\"prediction\",pred)#printing the prediction\n",
     " index=['APPLES','BANANA','ORANGE','PINEAPPLE','WATERMELON']\n",
     "  \n",
     "    \n",
     " result=str(index[pred[0]])\n",
     "           \n",
     " x=result\n",
     " print(x)\n",
     " result=nutrition(result)\n",
     " print(result)\n",
     " return
render_template(\"0.html\",showcase=(result),showcase1=(x))\n"
    ]
   },
   {
    "cell_type": "code",
    "execution_count": null,
    "id": "02980762",
    "metadata": {},
    "outputs": [
     {
      "name": "stdout",
      "output_type": "stream",
      "text": [
       " * Serving Flask app \"__main__\" (lazy loading)\n",
       " * Environment: production\n",
       "\u001b[31m   WARNING: This is a development server. Do not use it
in a production deployment.\u001b[0m\n",
       "\u001b[2m   Use a production WSGI server instead.\u001b[0m\n",
       " * Debug mode: off\n"
      ]
     },
     {
      "name": "stderr",
      "output_type": "stream",
      "text": [
       " * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)\n"
      ]
     }
    ],
    "source": [
     "def nutrition(index):\n",
     "\n",
     "\n",
     "    url = \"https://calorieninjas.p.rapidapi.com/v1/nutrition\"\n",
     "    \n",
     "    querystring = {\"query\":tomato}\n",
     "    \n",
     "    headers = {\n",
     "        'x-rapidapi-key':
\"5d797ab107mshe668f26bd044e64p1ffd34jsnf47bfa9a8ee4\",\n",
     "        'x-rapidapi-host': \"calorieninjas.p.rapidapi.com\"\n",
     "        }\n",
```

```
    "    \n",
    "    response = requests.request(\"GET\", url, headers=headers,
params=querystring)\n",
    "    \n",
    "    print(response.text)     \n",
    "    return response.json()['items']\n",
    "if __name__ == \"__main__\":\n",
    "   # running the app\n",
    "    app.run(debug=False)\n"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": null,
   "id": "fe5d99b3",
   "metadata": {},
   "outputs": [],
   "source": [
    " \n",
    "\n"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": null,
   "id": "a0635d29",
   "metadata": {},
   "outputs": [],
   "source": []
  },
  {
   "cell_type": "code",
   "execution_count": null,
   "id": "b649242c",
   "metadata": {},
   "outputs": [],
   "source": []
  },
  {
   "cell_type": "code",
   "execution_count": null,
   "id": "b881fa11",
   "metadata": {},
   "outputs": [],
   "source": []
  }
 ],
 "metadata": {
  "kernelspec": {
   "display_name": "Python 3 (ipykernel)",
   "language": "python",
   "name": "python3"
  },
  "language_info": {
   "codemirror_mode": {
    "name": "ipython",
    "version": 3
   },
```

```
   "file_extension": ".py",
   "mimetype": "text/x-python",
   "name": "python",
   "nbconvert_exporter": "python",
   "pygments_lexer": "ipython3",
   "version": "3.9.12"
  }
 },
 "nbformat": 4,
 "nbformat_minor": 5
}
```