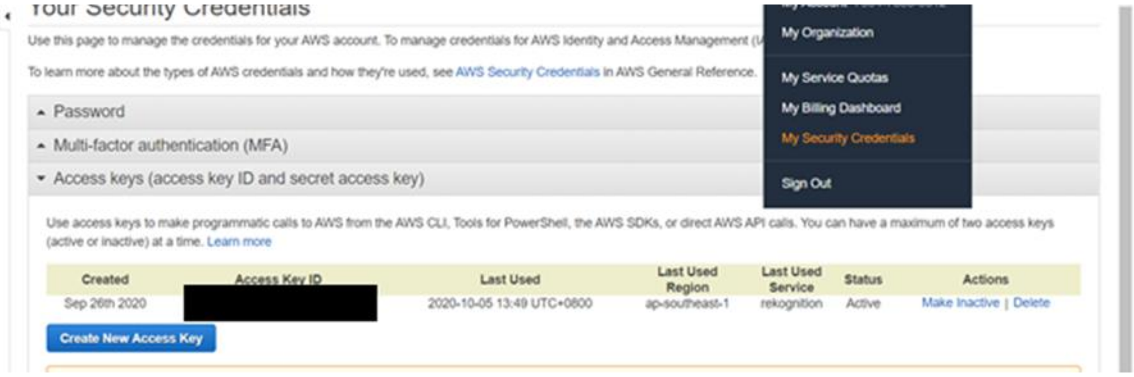
Before running the project

1. Save the zip file from our Google Drive and extract to a location of your preference. Alternatively, you can move the SystemCode file from our submitted zip file/Google Drive link (SimpleFoodDiary > SystemCode to a location of your preference).

2. Get an AWS secret and access key. First, sign up for an AWS account. Then select My Security Credentials after clicking your account name from the top navigation bar. Then click the blue Create New Access Key.



Your Security Credentials

Use this page to manage the credentials for your AWS account. To manage credentials for AWS Identity and Access Management (I

To learn more about the types of AWS credentials and how they're used, see AWS Security Credentials in AWS General Reference.

▲ Password

▲ Multi-factor authentication (MFA)

▼ Access keys (access key ID and secret access key)

Use access keys to make programmatic calls to AWS from the AWS CLI, Tools for PowerShell, the AWS SDKs, or direct AWS API calls. You can have a maximum of two access keys (active or inactive) at a time. Learn more

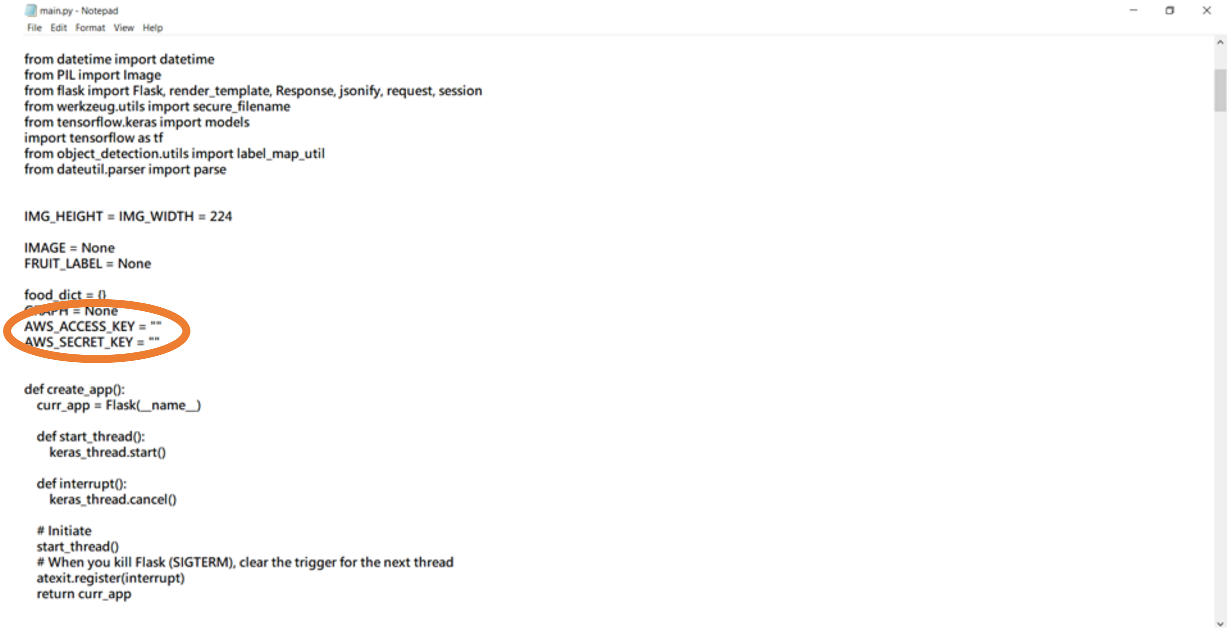
Created	Access Key ID	Last Used	Last Used Region	Last Used Service	Status	Actions
Sep 26th 2020	[redacted]	2020-10-05 13:49 UTC+0800	ap-southeast-1	rekognition	Active	Make Inactive | Delete

Setting up the project (through command line)

Here we are assuming the use of Windows to set up the project.

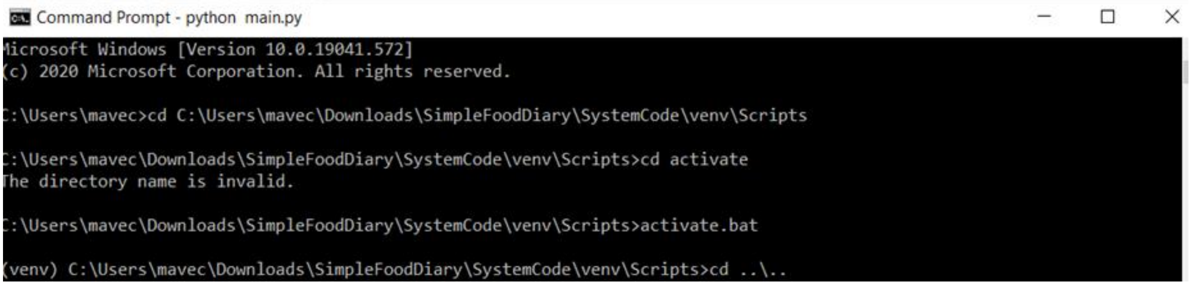
This requires a bit of time loading all the packages into the virtual environment, so we recommend using the IDE.

1. Open the main.py file from SimpleFoodDiary > SystemCode > main.py and add in your AWS secret key and access key into AWS_SECRET_KEY and AWS_ACCESS_KEY



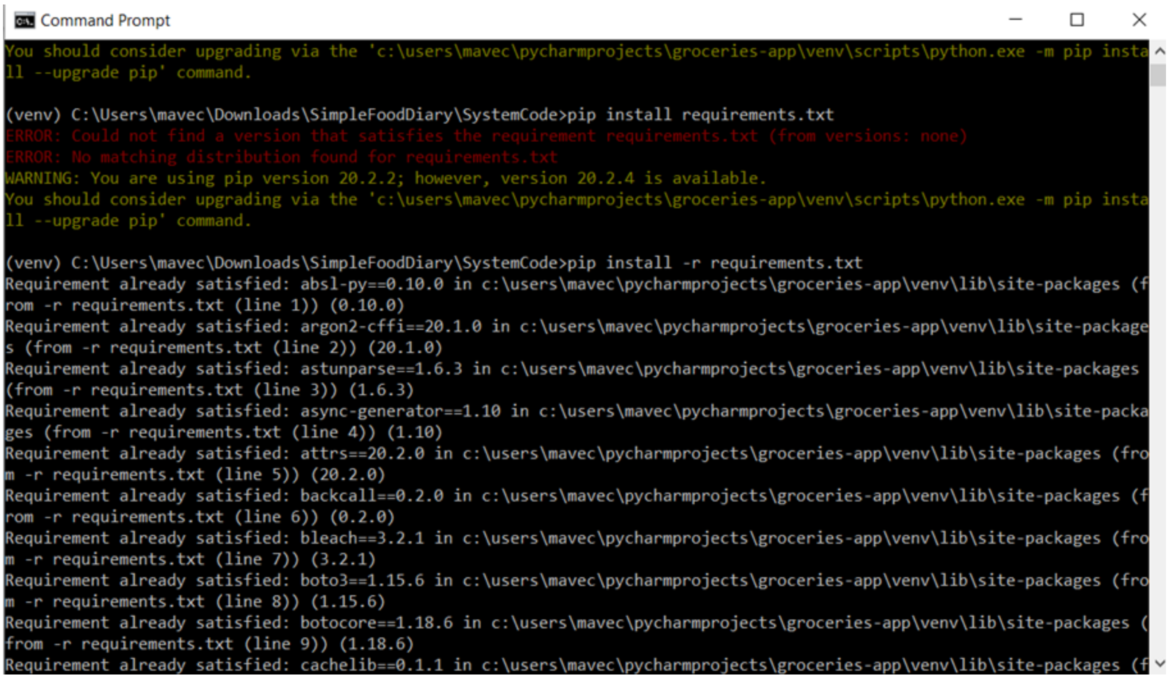
```
from datetime import datetime
from PIL import Image
from flask import Flask, render_template, Response, jsonify, request, session
from werkzeug.utils import secure_filename
from tensorflow.keras import models
import tensorflow as tf
from object_detection.utils import label_map_util
from dateutil.parser import parse


IMG_HEIGHT = IMG_WIDTH = 224

IMAGE = None
FRUIT_LABEL = None

food_dict = {}
GRAPH = None
AWS_ACCESS_KEY = ""
AWS_SECRET_KEY = ""


def create_app():
  curr_app = Flask(__name__)

  def start_thread():
    keras_thread.start()

  def interrupt():
    keras_thread.cancel()

  # Initiate
  start_thread()
  # When you kill Flask (SIGTERM), clear the trigger for the next thread
  atexit.register(interrupt)
  return curr_app
```

2. Open Command Terminal and change directory to the groceries-app (or SystemCode)\venv\Scripts. For the image below, the SimpleFoodApp folder is stored in my Downloads folder
3. Next type "activate.bat". This should activate the virtualenv



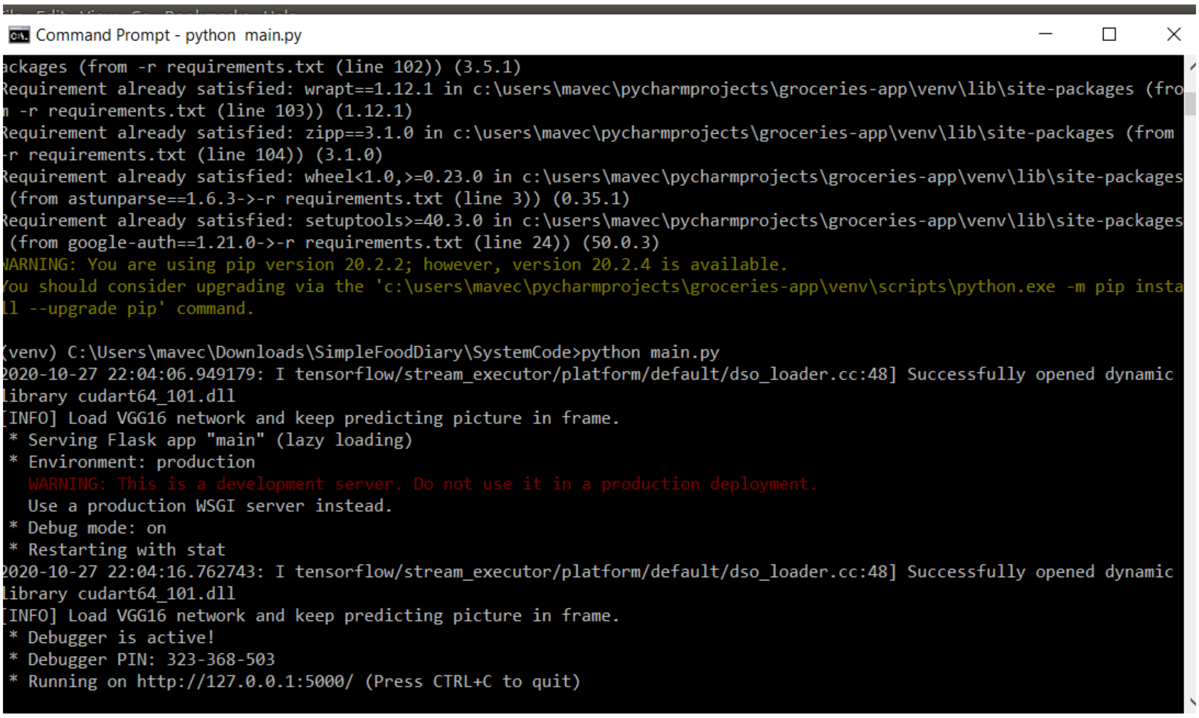
```
licrosoft Windows [Version 10.0.19041.572]
(c) 2020 Microsoft Corporation. All rights reserved.

:\Users\mavec>cd C:\Users\mavec\Downloads\SimpleFoodDiary\SystemCode\venv\Scripts

:\Users\mavec\Downloads\SimpleFoodDiary\SystemCode\venv\Scripts>cd activate
The directory name is invalid.

:\Users\mavec\Downloads\SimpleFoodDiary\SystemCode\venv\Scripts>activate.bat

(venv) C:\Users\mavec\Downloads\SimpleFoodDiary\SystemCode\venv\Scripts>cd ..\..
```

4. We now go back to the SystemCode or groceries-app folder by "cd ..\..". Here, we enter "pip install -r requirements.txt". You should see a whole chain of installations, this may take some time.



```
You should consider upgrading via the 'c:\users\mavec\pycharmprojects\groceries-app\venv\scripts\python.exe -m pip insta
ll --upgrade pip' command.

(venv) C:\Users\mavec\Downloads\SimpleFoodDiary\SystemCode>pip install requirements.txt
ERROR: Could not find a version that satisfies the requirement requirements.txt (from versions: none)
ERROR: No matching distribution found for requirements.txt
WARNING: You are using pip version 20.2.2; however, version 20.2.4 is available.
You should consider upgrading via the 'c:\users\mavec\pycharmprojects\groceries-app\venv\scripts\python.exe -m pip insta
ll --upgrade pip' command.

(venv) C:\Users\mavec\Downloads\SimpleFoodDiary\SystemCode>pip install -r requirements.txt
Requirement already satisfied: absl-py==0.10.0 in c:\users\mavec\pycharmprojects\groceries-app\venv\lib\site-packages (f
rom -r requirements.txt (line 1)) (0.10.0)
Requirement already satisfied: argon2-cffi==20.1.0 in c:\users\mavec\pycharmprojects\groceries-app\venv\lib\site-package
s (from -r requirements.txt (line 2)) (20.1.0)
Requirement already satisfied: astunparse==1.6.3 in c:\users\mavec\pycharmprojects\groceries-app\venv\lib\site-packages
(from -r requirements.txt (line 3)) (1.6.3)
Requirement already satisfied: async-generator==1.10 in c:\users\mavec\pycharmprojects\groceries-app\venv\lib\site-packa
ges (from -r requirements.txt (line 4)) (1.10)
Requirement already satisfied: attrs==20.2.0 in c:\users\mavec\pycharmprojects\groceries-app\venv\lib\site-packages (fro
m -r requirements.txt (line 5)) (20.2.0)
Requirement already satisfied: backcall==0.2.0 in c:\users\mavec\pycharmprojects\groceries-app\venv\lib\site-packages (f
rom -r requirements.txt (line 6)) (0.2.0)
Requirement already satisfied: bleach==3.2.1 in c:\users\mavec\pycharmprojects\groceries-app\venv\lib\site-packages (fro
m -r requirements.txt (line 7)) (3.2.1)
Requirement already satisfied: boto3==1.15.6 in c:\users\mavec\pycharmprojects\groceries-app\venv\lib\site-packages (fro
m -r requirements.txt (line 8)) (1.15.6)
Requirement already satisfied: botocore==1.18.6 in c:\users\mavec\pycharmprojects\groceries-app\venv\lib\site-packages (
from -r requirements.txt (line 9)) (1.18.6)
Requirement already satisfied: cachelib==0.1.1 in c:\users\mavec\pycharmprojects\groceries-app\venv\lib\site-packages (f
```

5. Lastly, we run the "python main.py". Only open http://127.0.0.1:5000/ when you see the link available.

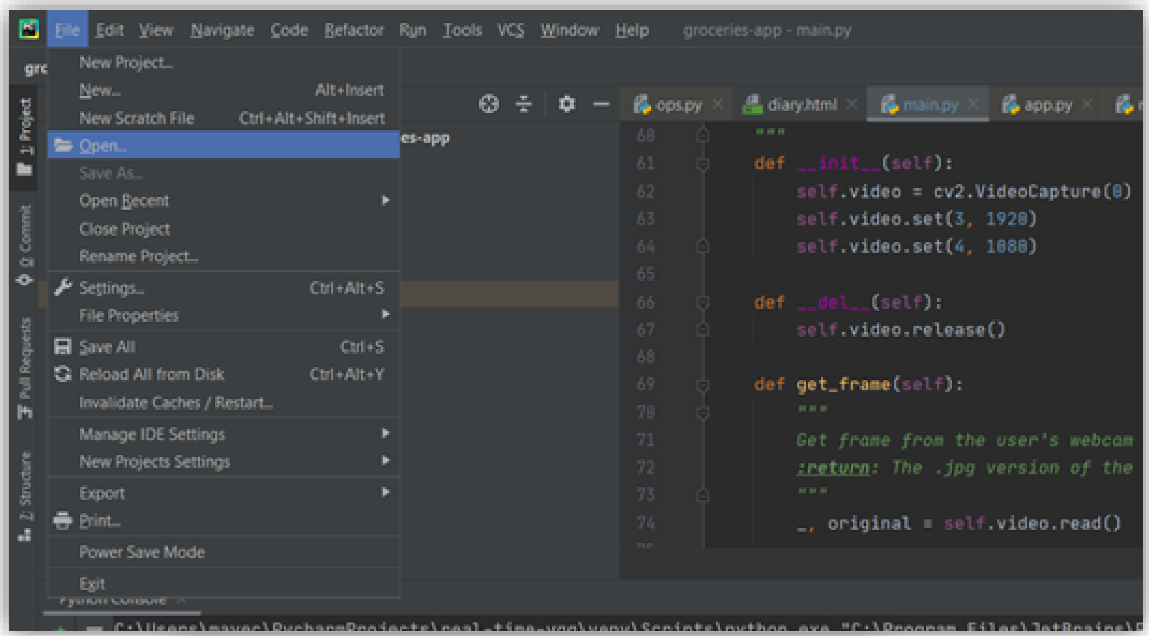


```
ackages (from -r requirements.txt (line 102)) (3.5.1)
Requirement already satisfied: wrapt==1.12.1 in c:\users\mavec\pycharmprojects\groceries-app\venv\lib\site-packages (fro
n -r requirements.txt (line 103)) (1.12.1)
Requirement already satisfied: zipp==3.1.0 in c:\users\mavec\pycharmprojects\groceries-app\venv\lib\site-packages (from
-r requirements.txt (line 104)) (3.1.0)
Requirement already satisfied: wheel<1.0,>=0.23.0 in c:\users\mavec\pycharmprojects\groceries-app\venv\lib\site-packages
 (from astunparse==1.6.3->-r requirements.txt (line 3)) (0.35.1)
Requirement already satisfied: setuptools>=40.3.0 in c:\users\mavec\pycharmprojects\groceries-app\venv\lib\site-packages
 (from google-auth==1.21.0->-r requirements.txt (line 24)) (50.0.3)
WARNING: You are using pip version 20.2.2; however, version 20.2.4 is available.
You should consider upgrading via the 'c:\users\mavec\pycharmprojects\groceries-app\venv\scripts\python.exe -m pip insta
ll --upgrade pip' command.

(venv) C:\Users\mavec\Downloads\SimpleFoodDiary\SystemCode>python main.py
2020-10-27 22:04:06.949179: I tensorflow/stream_executor/platform/default/dso_loader.cc:48] Successfully opened dynamic
library cudart64_101.dll
[INFO] Load VGG16 network and keep predicting picture in frame.
 * Serving Flask app "main" (lazy loading)
 * Environment: production
   WARNING: This is a development server. Do not use it in a production deployment.
   Use a production WSGI server instead.
 * Debug mode: on
 * Restarting with stat
2020-10-27 22:04:16.762743: I tensorflow/stream_executor/platform/default/dso_loader.cc:48] Successfully opened dynamic
library cudart64_101.dll
[INFO] Load VGG16 network and keep predicting picture in frame.
 * Debugger is active!
 * Debugger PIN: 323-368-503
 * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
```

Setting up the project (through IDE)

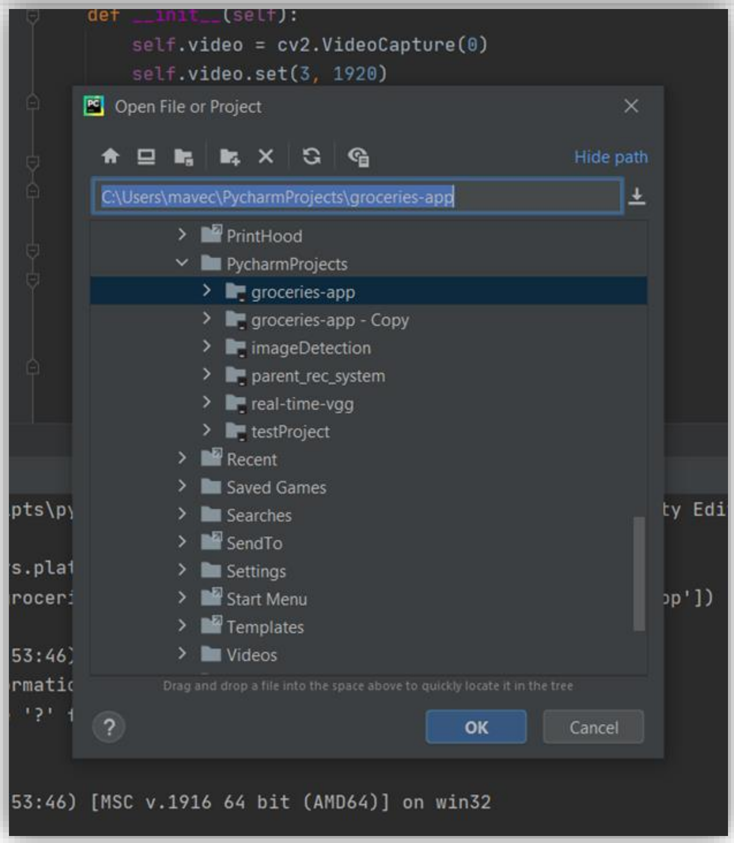
Here we are assuming the use of Windows to set up the project.

6. For the guide below, we will be using PyCharm, our preferred IDE. You may wish to use other IDEs as well, but we recommend using PyCharm to ensure no issues using our system. The link for downloading PyCharm is here.
7. Open the groceries-app (or SystemCode) by clicking File at the toolbar above > Then Open > Then navigate to where you stored groceries-app (or SystemCode) > Click OK button.



```
        """
    def __init__(self):
        self.video = cv2.VideoCapture(0)
        self.video.set(3, 1920)
        self.video.set(4, 1080)

    def __del__(self):
        self.video.release()

    def get_frame(self):
        """
        Get frame from the user's webcam
        :return: The .jpg version of the
        """
        _, original = self.video.read()
```

```
C:\Users\mavec\PycharmProjects\real-time-vgg\venv\Scripts\python.exe "C:\Program Files\JetBrains\
```

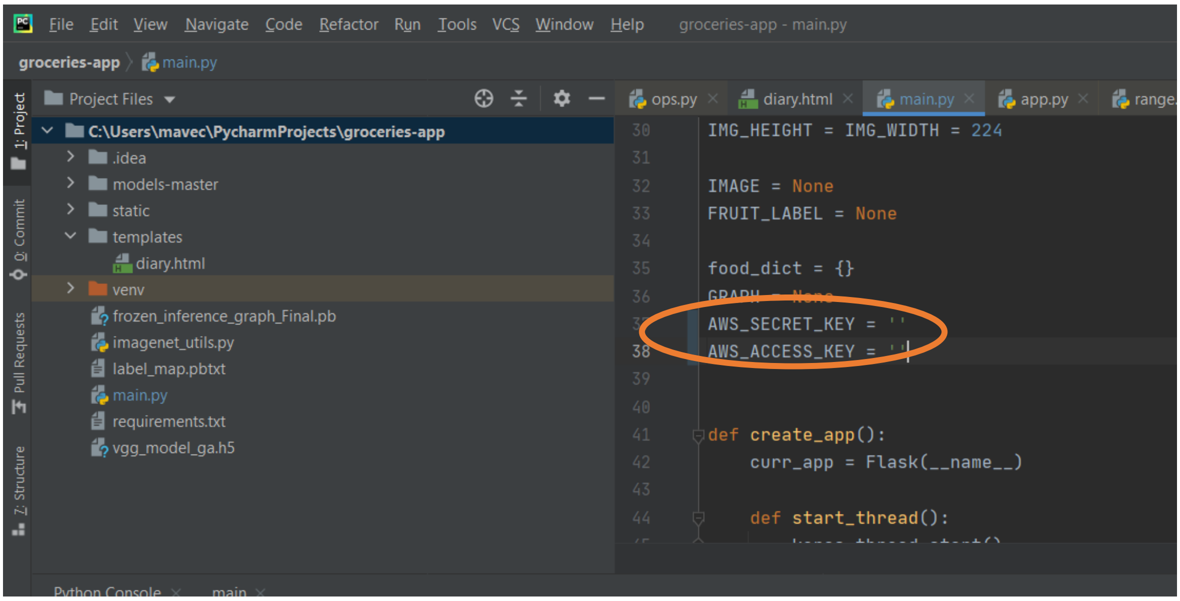


```
    def __init__(self):
        self.video = cv2.VideoCapture(0)
        self.video.set(3, 1920)
```

C:\Users\mavec\PycharmProjects\groceries-app

```
pts\py                                                  ty Edi
s.plat
roceri                                                  pp'])
53:46)
rmatic
 '?' i
53:46) [MSC v.1916 64 bit (AMD64)] on win32
```

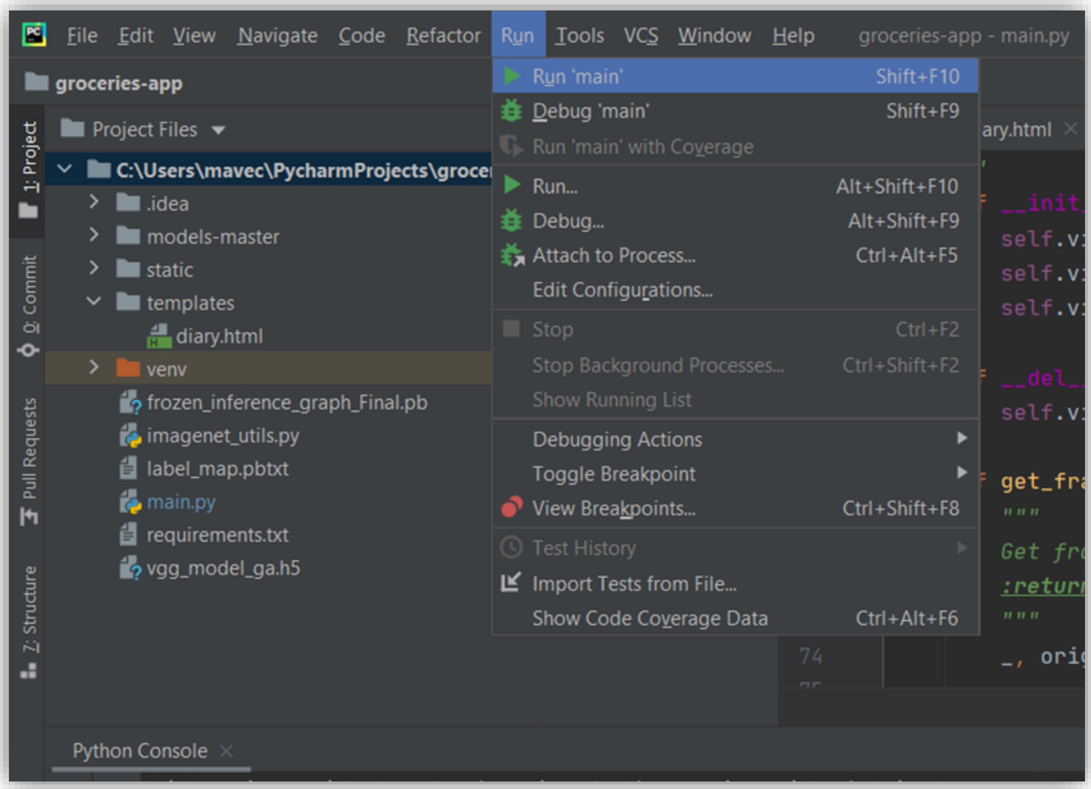
8. Add your AWS secret key and access key in main.py under AWS_SECRET_KEY and AWS_ACCESS_KEY



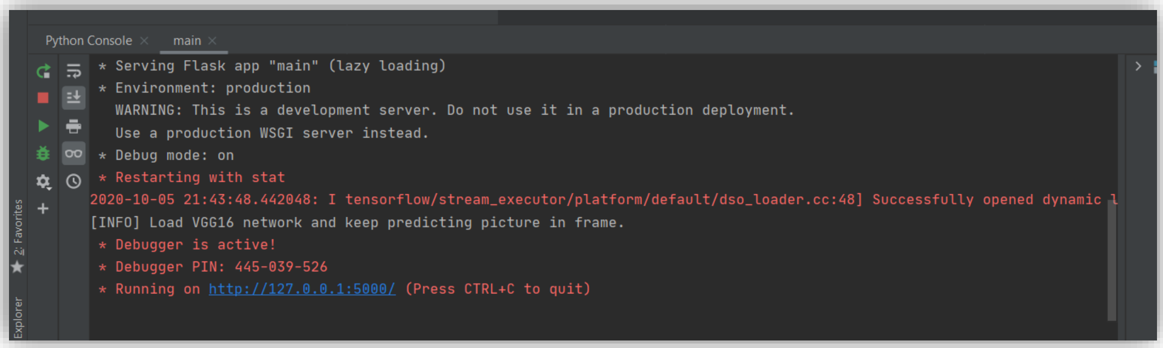
```
IMG_HEIGHT = IMG_WIDTH = 224

IMAGE = None
FRUIT_LABEL = None

food_dict = {}
GRAPH = None
AWS_SECRET_KEY = ''
AWS_ACCESS_KEY = ''


def create_app():
    curr_app = Flask(__name__)

    def start_thread():
```

9. If requirements not fulfilled, it will prompt you to install requirements. You can ignore any errors on ipython and traitlets.

10. Go to Run at the top tool bar > Click Run ‘main’.

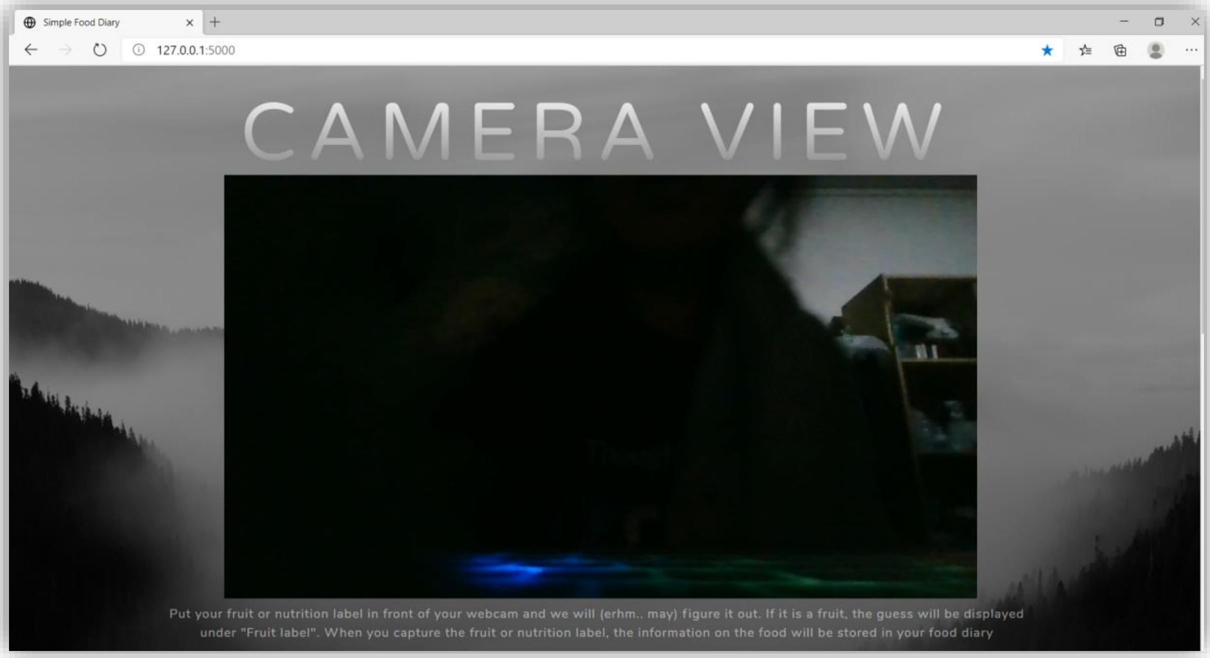


11. Wait for this to appear, then click on the blue link with the URL – http://127.0.0.1:5000/.
    This app should be able to run in Edge or Chrome browser.



```
 * Serving Flask app "main" (lazy loading)
 * Environment: production
   WARNING: This is a development server. Do not use it in a production deployment.
   Use a production WSGI server instead.
 * Debug mode: on
 * Restarting with stat
2020-10-05 21:43:48.442048: I tensorflow/stream_executor/platform/default/dso_loader.cc:48] Successfully opened dynamic l
[INFO] Load VGG16 network and keep predicting picture in frame.
 * Debugger is active!
 * Debugger PIN: 445-039-526
 * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
```

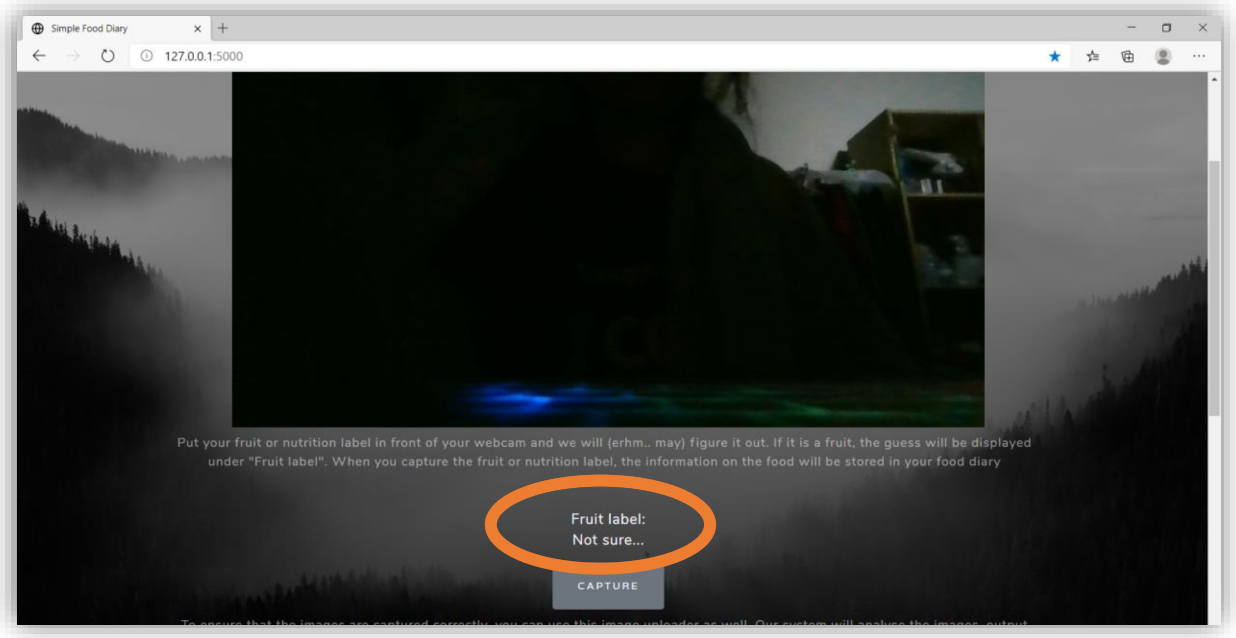
12. If you have a webcam on, you should be able to see this:



Put your fruit or nutrition label in front of your webcam and we will (erhm.. may) figure it out. If it is a fruit, the guess will be displayed under "Fruit label". When you capture the fruit or nutrition label, the information on the food will be stored in your food diary

Using the app

1. You can show these five types of fruits ('Apple', 'Longbean', 'Onion', 'Pineapple', 'Potato') to the camera to see if it is able to predict the fruit label correctly. In the event that the model is unsure, it will indicate “Not sure…” accordingly. Once you are happy with the label and the picture, just click on the Capture button.

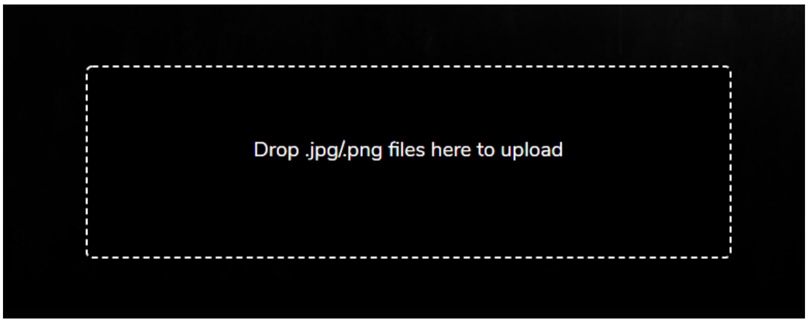


Put your fruit or nutrition label in front of your webcam and we will (erhm.. may) figure it out. If it is a fruit, the guess will be displayed under "Fruit label". When you capture the fruit or nutrition label, the information on the food will be stored in your food diary

Fruit label:
Not sure...

To ensure that the images are captured correctly, you can use this image uploader as well. Our system will analyse the images, output

2. Alternatively, you can show any nutrition label to the camera and click capture. It should be able to crop the nutrition label correctly and extract whatever text it can find.
3. If you are unhappy with the pictures in the camera, you can choose to upload images as well, by either dragging and dropping into the dotted box or click on the dotted box and select the image.



Drop .jpg/.png files here to upload

4. The Food Diary will then show you the label with/without extracted text as well as the date the image was captured or uploaded.