SRPCE

NAME:- S.Arun

ROLL.NO:- 3

REG.NO:- 422021104003

SEMESTER:- 5

DEPARTMENT:-CSE

SUBJECT:- cloud Application development

SESSION:- 2021-2025

Innovation is the process of creating new or improved products, services, or processes. It can also involve new ways of thinking and doing things. Innovation is important because it can lead to economic growth, improved quality of life, and new solutions to challenges.

There are many different types of innovation. Some examples include:

- Product innovation: This involves developing new products or improving existing products. For example, the invention of the smartphone was a major product innovation.
- Service innovation: This involves developing new services or improving existing services. For example, the development of online banking was a major service innovation.
- Process innovation: This involves developing new or improved ways of producing or delivering products or services. For example, the development of the assembly line was a major process innovation.
- Business model innovation: This involves developing new ways of doing business. For example, the development of the freemium business model was a major business model innovation.

Innovation can be driven by many different factors, including:

- Customer needs: Businesses are often motivated to innovate in order to meet the changing needs of their customers. For example, the development of the electric car was driven by the need for more sustainable transportation options.
- Technological advances: New technologies can create new opportunities for innovation. For example, the development of artificial intelligence has led to new innovations in many different industries.
- Government policies: Governments can support innovation through policies such as tax breaks and research funding. For example, the US government has invested heavily in research and development for many decades, which has led to many major innovations.

Innovation is an essential part of economic growth and social progress. It helps businesses to compete and grow, and it can lead to new solutions to challenges such as climate change and poverty.

Here are some examples of innovation in the real world:

- The development of the mRNA vaccine was a major innovation in the fight

You took the trouble of installing a webcam or a surveillance camera at your home, or office or any place that you own. Obviously, you would want to be able to watch the live stream of the video anyplace and anytime you like.

Most of the people use IP cameras (Internet Protocol cameras) instead of CCTV (Closed-Circuit Television) for surveillance purposes as they have a much higher resolution and reduced cost for cabling. You can find the detailed differences between both these systems here. In this article we will be focusing on IP cameras.

IP camera, is a type of digital video camera that receives control data and sends image data via an IP network and require no local recording device. Most IP cameras are RTSP (Real Time Streaming Protocol) based and is therefore "not supported" natively in internet browsers.

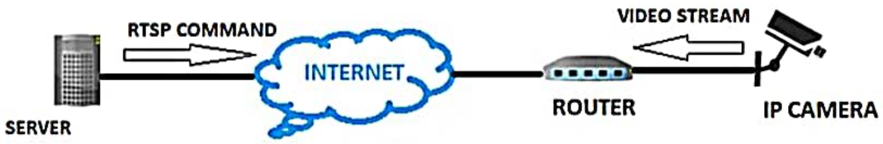


Image by Author-Working of RTSP Protocol

So how do you use your web browser to view the live streaming ?

In this article we will learn how to do that using Computer Vision.

> *Computer Vision is an interdisciplinary field that deals with how computers can be made to gain a high-level understanding from digital images or videos.*

For implementing the computer vision part we will use the **OpenCV** module in Python and to display the live stream in the web browser we will use the **Flask** web framework. Before diving into the coding part let us first know about these modules briefly. If you are already familiar with these modules, you can directly jump to the next section.

*According to the Wikipedia, **Flask** is a micro web framework written in Python. It is classified as a microframework because it does not require particular tools or libraries. It has no database abstraction layer, form validation, or any other components where pre-existing third-party libraries provide common functions.*

*According to GeeksForGeeks, **OpenCV** is the huge open-source library for the computer vision, machine learning, and image processing and now it plays a major role in real-time operation which is very important in today's systems.*

. . .

Step1- Install Flask & OpenCV :

You can use the '*pip install* flask' and '*pip install opencv-python*' command. I use the PyCharm IDE to develop flask applications. ***To easily install libraries in PyCharm*** follow these steps.

Step2- Import necessary libraries, initialize the flask app :

Step2- Import necessary libraries, initialize the flask app :

We will now import the necessary libraries and initialize our flask app.

```
#Import necessary libraries
from flask import Flask, 
render_template, Response
import cv2
#Initialize the Flask app
app = Flask(__name__)
```

Step3- Capture Video using OpenCV :

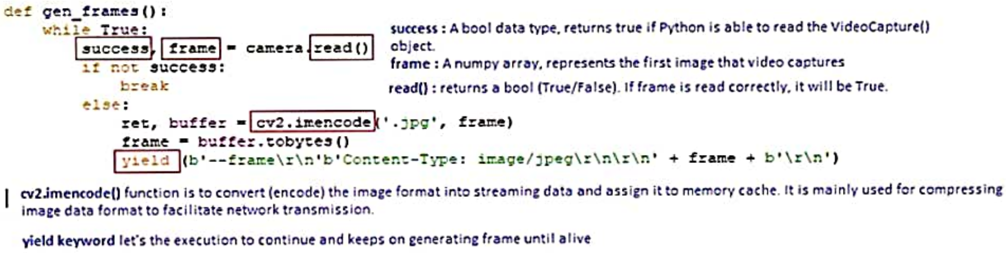
Create a VideoCapture() object to trigger the camera and read the first image/frame of the video. We can either provide the path of the video file or use numbers to specify the use of local webcam. To trigger the webcam we pass '0' as the argument. To capture the live feed from an IP Camera we provide the **RTSP link** as the argument. To know the RTSP address for your IP Camera go through this — **Finding RTSP addresses**.

```
camera = cv2.VideoCapture(0)
```

```
'''
for ip camera use -
rtsp://username:password@ip_a
ddress:554/user=username_pass
word='password'_channel=chann
el_number_stream=0.sdp'

for local webcam use
cv2.VideoCapture(0)
'''
```

Step4- Adding window and generating frames from the camera:



```
def gen_frames():
    while True:
        success, frame = camera.read()
        if not success:
            break
        else:
            ret, buffer = cv2.imencode('.jpg', frame)
            frame = buffer.tobytes()
            yield (b'--frame\r\n'b'Content-Type: image/jpeg\r\n\r\n' + frame + b'\r\n')
```

success : A bool data type, returns true if Python is able to read the VideoCapture() object.

frame : A numpy array, represents the first image that video captures

read() : returns a bool (True/False). If frame is read correctly, it will be True.

cv2.imencode() function is to convert (encode) the image format into streaming data and assign it to memory cache. It is mainly used for compressing image data format to facilitate network transmission.

yield keyword let's the execution to continue and keeps on generating frame until alive

Image by Author-Frame generation function

The gen_frames() function enters a loop where it continuously returns frames from the camera as response chunks. The function asks the camera to provide a frame then it yields with this frame formatted as a response chunk with a content type of `image/jpeg`, as shown above. The code is shown below :

```
def gen_frames():
    while True:
        success, frame = camera.read(
        if not success:
            break
        else:
            ret, buffer = cv2.imencod
            frame = buffer.tobytes()
            yield (b'--frame\r\n'
                   b'Content-Type: im
```

frames.py hosted with ❤ by **GitHub** **view raw**

Frame generating function

Step5- Define app route for default page of the web-app :

Routes refer to URL patterns of an app (such as myapp.com/home or myapp.com/about). `@app.route("/")` is a Python decorator that Flask provides to assign URLs in our app to functions easily.

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

The decorator is telling our `@app` that whenever a user visits our app domain

The decorator is telling our `@app` that whenever a user visits our app domain (*localhost:5000 for local servers*) at the given `.route()`, execute the `index()` function. Flask uses the Jinja template library to render templates. In our application, we will use templates to render HTML which will display in the browser.

Step6- Define app route for the Video feed:

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

The '/video_feed' route returns the streaming response. Because this stream returns the images that are to be displayed in the web page, the URL to this route is in the “src” attribute of the image tag (see ‘index.html’ below). The browser will automatically keep the image element updated by displaying the stream of JPEG images in it, since multipart responses are supported in most/all browsers

Let’s have a look at our **index.html** file

:

```
<body>
<div class="container">
    <div class="row">
        <div class="col-lg-8
offset-lg-2">
            <h3 class="mt-
5">Live Streaming</h3>
            <img src="{{
url_for('video_feed') }}"
width="100%">
        </div>
    </div>
</div>
</body>
```

Step7- Starting the Flask Server :

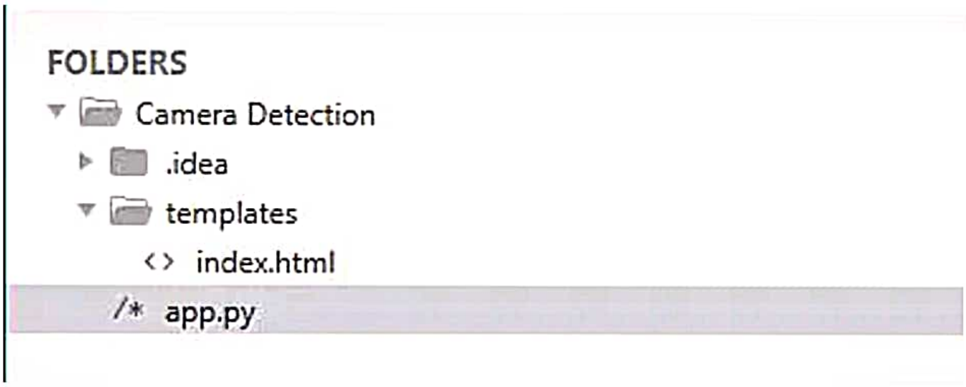
```
if __name__ == "__main__":
    app.run(debug=True)
```

app.run() is called and the web-application is hosted locally on *[localhost:5000].*

"debug=True" makes sure that we don't require to run our app every time we makes changes, we can simply refresh our web page to see the changes while the server is still running.

. . .

Project Structure :



```
FOLDERS
▼ Camera Detection
  ▶ .idea
  ▼ templates
      index.html
    app.py
```

Image by Author-Project Structure

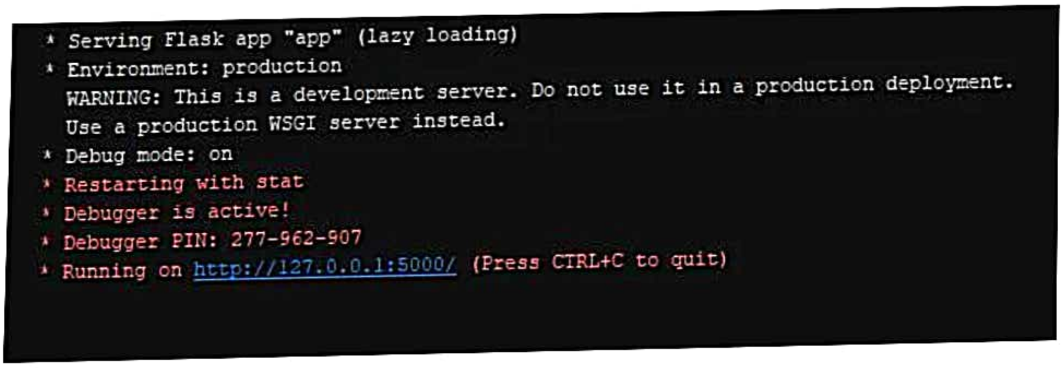
The project is saved in a folder called "Camera Detection". We run the

> *You can simply type "localhost:5000" on your web browser to open your web-application after running 'app.py'*

- app.py — This is the Flask application we created above

- templates — This folder contains our 'index.html' file. This is mandatory in Flask while rendering templates. All HTML files are placed under this folder.

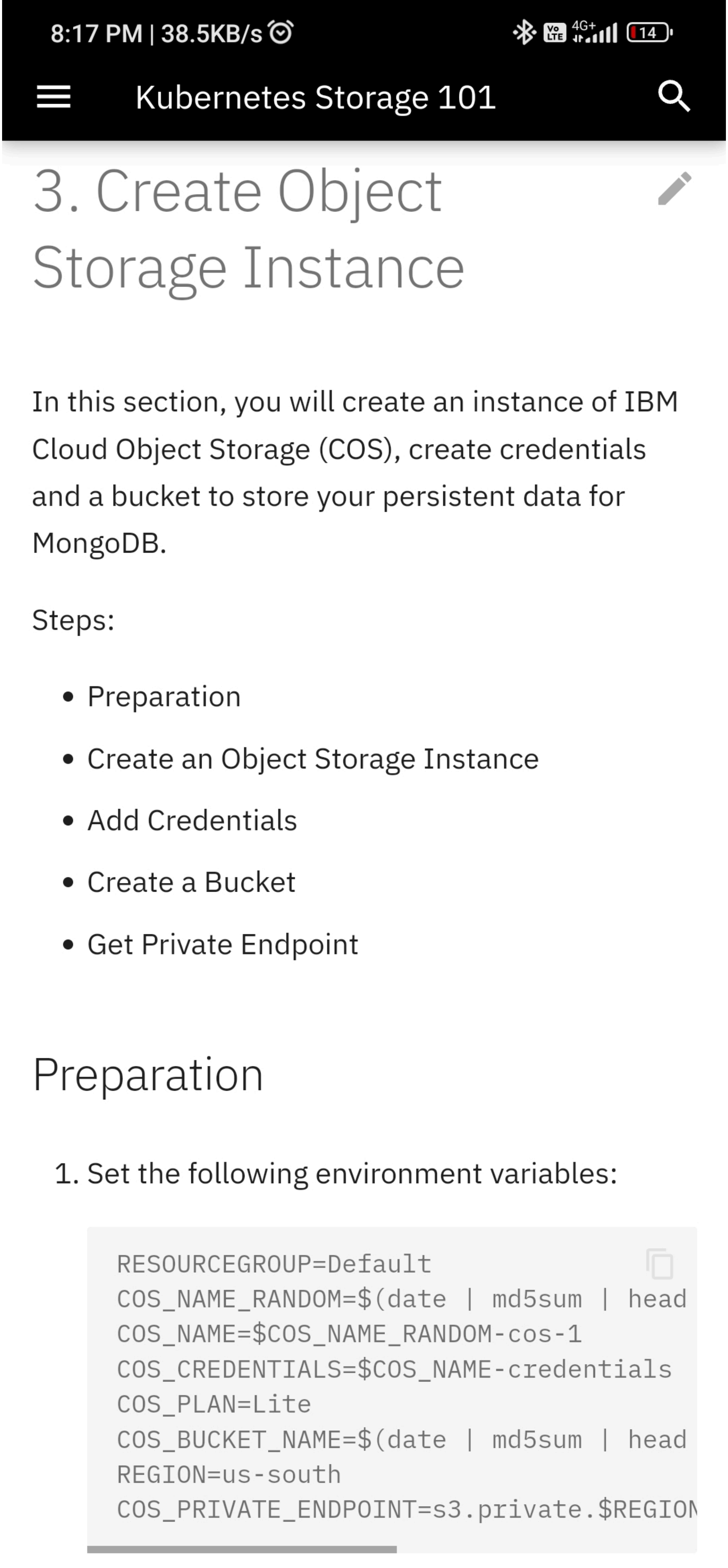
Let's see what happens when we run 'app.py' :



```
 * Serving Flask app "app" (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
 * Debugger is active!
 * Debugger PIN: 277-962-907
 * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
```

Image by Author-Local Server for Flask Application

On clicking on the provided URL, our web browser opens up with the live feed. Since I used **VideoCapture(0)**



3. Create Object Storage Instance

In this section, you will create an instance of IBM Cloud Object Storage (COS), create credentials and a bucket to store your persistent data for MongoDB.

Steps:

- Preparation
- Create an Object Storage Instance
- Add Credentials
- Create a Bucket
- Get Private Endpoint

Preparation

1. Set the following environment variables:

```
RESOURCEGROUP=Default
COS_NAME_RANDOM=$(date | md5sum | head
COS_NAME=$COS_NAME_RANDOM-cos-1
COS_CREDENTIALS=$COS_NAME-credentials
COS_PLAN=Lite
COS_BUCKET_NAME=$(date | md5sum | head
REGION=us-south
COS_PRIVATE_ENDPOINT=s3.private.$REGION
```