SPRINT-3

DEVELOP A PYTHON SCRIPT

TEAM ID	PNT2022TMID44357
PROJECT DOMAIN	INTERNET OF THINGS
PROJECT TITLE	IoT BASED SMART CROP PROTECTION SYSTEM FOR AGRICULTURE
DATE	12 NOVEMBER 2022

CODE:

```
import cv2
import numpy as np
import wiotp.sdk.device
import playsound
import random
import time
import datetime
import ibm_boto3
from ibm_botocore.client import Config, ClientError
#CloudantDB
from cloudant.client import Cloudant
from cloudant.error import CloudantException
from cloudant.result import Result, ResultByKey
from clarifai_grpc.channel .clarifai_channel import ClarifaiChannel
from clarifai_grpc.grpc.api import service_pb2_grpc
stub = service_pb2_grpc.V2Stub (ClarifaiChannel.get_grpc_channel())
from clarifai_grpc.grpc.api import service_pb2, resources_pb2
from clarifai_grpc.grpc.api.status import status_code_pb2
# This is how you authenticate.
```

```
metadata = (('authorization', 'Key 511c50ba8cf14c4c9b9110058e11beaa'))

COS_ENDPOINT = "https://s3.jp-tok.cloud-object-storage.appdomain.cloud" # Current list avaiable at https://control.cloud-object-storage.cloud.ibm.com/v2/endpoints

COS_API_KEY_ID = "3OtZ1HxvkGiBAmzrYVMHxNdotCvuVnXyARhHCe94PYoq" # eg "WO0YiRnLW4a3fTjMB-odB-2ySfTrFBIQQWanc--P3byk"

COS_AUTH_ENDPOINT = "https://iam.cloud.ibm.com/identity/token"

COS_RESOURCE_CRN = "crn:v1:bluemix:public:cloud-object-storage:global:a/855b724ffede4b938c015d8333f51eb4:15dfbf35-e8ab-4a60-a963-8f051b6c03ac::"# eg "crn:v1:bluemix:public:cloud-object-stc

clientdb = Cloudant("apikey-v2-2sv177ca28l0r9s7uuj51oemulzrrfyhpu8agz4wvy4c", "0bb872b6a4bef6634cf1d90a82a7fcbd", url="https://apikey-v2-2sv177ca28l0r9s7uuj51oemulzrrfyhpu8agz4wvy4c:0bb872b6a4bef6634cf1d90a82a7fcbd@2a61fe59-9627-4496-9696-ead1be2b50bd-bluemix.cloudantnosqldb.appdomain.cloud")

clientdb.connect()

# Create resource

cos = ibm_boto3. resource ("s3",

  ibm_api_key_id=COS_API_KEY_ID,

  ibm_service_instance_id=COS_RESOURCE_CRN,

  ibm_auth_endpoint=COS_AUTH_ENDPOINT,

  config=Config(signature_version="oauth"),

  endpoint_url=COS_ENDPOINT

)

def multi_part_upload(bucket_name, item_name, file_path):

 try:

   print("Starting file transfer for {0} to bucket: {1}\n".format(item_name, bucket_name))

   # set 5 MB chunks
```

```
  part_size = 1024* 1024 * 5
  # set threadhold to 15 MB
  file_threshold=1024*1024 * 15
  # set the transfer threshold and chunk size
  transfer_config = ibm_boto3.s3.transfer. TransferConfig(
    multipart_threshold=file_threshold,
    multipart_chunksize=part_size
  )
  # the upload_fileobj method will automatically execute a multi-part upload
  # in 5 MB chunks for all files over 15 MB
  with open(file_path, "rb") as file_data:
    cos.Object(bucket_name, item_name).upload_fileobj(
     Fileobj=file_data,
     Config=transfer_config
    )
    print("Transfer for {0} Complete!\n".format(item_name))
 except ClientError as be:
       print("CLIENT ERROR: {0}\n". format(be))
 except Exception as e:
       print("Unable to complete multi-part upload: {0}".format(e))
def myCommandCallback(cmd):
  print("Command received: %s" % cmd.data)
  command=cmd.data['command']
  print(command)
if (command=='lighton'):
   print('lighton')
elif (command=='lightoff'):
```

```
    print('lightoff')
elif (command=='motoron'):
    print('motoron')
elif (command=='motoroff'):
    print('motoroff')
myConfig = {
  "identity": {
    "orgId": "f41515",
    "typeId": "SAIS",
    "deviceId": "6880"
  },
  "auth": {"token":"6385657616"}
}
client = wiotp.sdk.device.DeviceClient(config=myConfig, logHandlers=None)
client.connect()

database_name = "sample"
my_database = clientdb.create_database(database_name)
if my_database.exists():
   print (f"'{database_name}' successfully created.")
cap=cv2.VideoCapture('FIELD.mp4')
if(cap.isOpened() ==True):
   print('File opened')
else:
   print('File not found')
while(cap.isopened()):
   ret, frame = cap.read()
```

```
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
    ims = cv2.resize(frame, (960, 540))
    cv2.imwrite('IMAGE1.jpg',ims)
    with open("IMAGE1.jpg", "rb") as f:
      file_bytes = f.read()
    # This is the model ID of a publicly available General model. You may use
any other public or custom model ID.
    request = service_pb2.PostModelOutputsRequest(
      model_id='aaa03c23b3724a16a56b629203edc62c',

inputs=[resources_pb2.Input(data=resources_pb2.Data(image=resources_pb2.
Image(base64=file_bytes))
      )])
    response = stub.PostModelOutputs (request, metadata=metadata)
    if response.status.code != status_code_pb2.SUCCESS:
      raise Exception ("Request failed, status code: " + str(response.status.code))
    detect=False
    for concept in response.outputs [0].data.concepts:
      #print('%12s: %.2f' % (concept.name, concept.value))
      if (concept.value>0.98):
        #print (concept.name)
        if (concept.name=="animal"):
          print ("Alert! Alert! animal detected")
          playsound.playsound ('alert.mp3')
          picname=datetime.datetime.now().strftime ("%y-%m-%d-%-%M")
          cv2.imwrite (picname+'.jpg', frame)
          multi_part_upload('swasthi', picname+'.jpg', picname+'.jpg')

json_document={"link":COS_ENDPOINT+'/'+'swasthi'+'/'+picname+'.jpg'}
```

```
        new_document = my_database.create_document (json_document)
        if new_document.exists():
          print(f"Document successfully created.")
        time.sleep(5)
        detect=True
   moist-random.randint(0,100)
   humidity=random.randint(0,100)
   myData={'Animal': detect, 'moisture' :moist, 'humidity': humidity}
   print (myData)
if(humidity!=None):
     client.publishEvent (eventId="status",
msgFormat="json",data=myData,qos=0,unPublish=None)
     print("Publish Ok..")
client.commandCallback = myCommandCallback
cv2.imshow('frame', ims)
if cv2.waitKey(1) & 0xFF== ord('q'):
  client.disconnect()
 cap.release()
 cv2.destroyAllWindows()
```