```
Import cv2

Import numpy as np

Import wiot.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, resource_pb2

From clarifai_grpc.grpc.api.status import status_code_pb2


#This is how you authenticate

Metadata = ((‘authorization’, ‘key 5797d941-433e-436a-a480-680d9080a990’),)

COS_ENDPOINT = https://s3.tok.ap.cloud-object-storage.appdomain.cloud
```

COS_API_KEY_ID = “v9n8Zn4r5VpcMVz_HyRY0DrS13jSzph2IEFioVj4-vmT”

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

COS_RESOURCE_CRN = “crn:v1:bluemix:public:cloud-object-storage:global:a/3f060ee770d94e20a88f49f3da641d6d:f301cab2-2e94-48a1-a8a0-5b4968527c54::”

Clientdb = cloudant(“apikey-_pIeLXPoaPpnOZ7SMoVKd6tZdsjf54X9LwkFEWB1a0T6”, “0165dca6-1176-4aa5-b0fe-81473e50e35d”, url=https://47643860-3553-4211-ba2a-d8e26dd17c08-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 size
        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€)


Def myCommandCallback(cmd):
    Print(“Command received: %s” % cmd.data)
    Command=cmd.data[‘command’]
```

```
    Print(command)
    If(commamd==”lighton”):
        Print(‘lighton’)
    Elif(command==”lightoff”):
        Print(‘lightoff’)
    Elif(command==”motoron”):
        Print(‘motoron’)
    Elif(command==”motoroff”):
        Print(‘motoroff’)
myConfig = {
    “identity”: {
        “orgId”: “chytun”,
        “typeId”: “NodeMCU”,
        “deviceId”: “12345”
        },
    “auth”: {
        “token”: “12345678”
        }
    }
Client = wiot.sdk.device.DeviceClient(config=myConfig, logHandlers=None)
Client.connect()

Database_name = “sample”
```

```
My_database = clientdb.create_database(database_name)
If my_dtabase.exists():
    Print(f"'(database_name)' successfully created.")
Cap=cv2.VideoCapture("garden.mp4")
If(cap.isOpened()==True):
    Print('File opened')
Else:
    Print('File not found')

While(cap.isOpened()):
    Ret, frame = cap.read()
    Gray = cv3.cvtColor(frame, cv2.COLOR_BGR@GRAY)
    imS= cv2.resize(frame, (960,540))
    cv2.inwrite('ex.jpg',imS)
    with open("ex.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='82eaf1c767a74869964531e4d9de5237',

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: %.f' % (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-%H-%M")
                Cv2.inwrite(picname+'.jpg',frame)
                Multi_part_upload('Umamaheswari', picname+'.jpg', picname+'.jpg')

Json_document={"link":COS_ENDPOINT+'/'+'Umamaheswari'+'/'+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”, daya=myData, qos=0,
onPublish=None)
        print(“Publish Ok..”)
    client.commandCallback = myCommandCallback
    cv2.imshow(‘frame’,imS)
    if cv2.waitKey(1) & 0xFF == ord(‘q’):
        break
client.disconnect()
cap.release()
cv2.destroyAllWindows()
```