SPRINT -4

Team -ID	PNT20222TMID37200
Project Name	Project - IOT Based Smart Crop Protection System for Agriculture

In this sprint, after detection of animals using clarifai code, the alert sound is played to prevent animals from destroying the crops.

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_ph2, resources_pb2
from clarifai_grpc.grpc.api.status import status_code_pb2


#This is how you authenticate.


metadata=(('authorization', 'Key b7cc869daea7115a61b178f999304658'),)
COS_ENDPOINT = "https://control.cloud-object-storage.cloud.ibm.com/v2/endpoints"
COS_API_KEY_ID = "dF_ajTKrMs93h4pTSBf-1E4FIQSuIj0z5mgsJWyX3wgc"
COS_AUTH_ENDPOINT = "https://iam.cloud.ibm.com/identity/token"
COS_RESOURCE_CRN= "crn:v1:bluemix:public:cloud-object-storage:global:a/ca16b24add6f4ffe84e8ff57a0ec74d0:0eab0b40-b7a1-4c06-a75d-19bfaf767617::"
clientdb=Cloudant("apikey-v2-nxBFlhXL-q6vLV9J-vD93mfcEDEYXw1v5ij45LB6n-o5 2457a0ac-7d9a-4aad-9f7a-40035440ad9e-0", "e2099327-ee98-489f-a023-24caa0f33ed3", url="https://apikey-v2-nxBFlhXL-q6vLV9J-vD93mfcEDEYXw1v5ij45LB6n-o5 2457a0ac-7d9a-4aad-9f7a-40035440ad9e-0")
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="auth"),
    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
with open(file_path,"rb") as file_data:
   cos.Object(bucket_name, item_name).upload_fileobj(
      Fileobj=file_data,
      Conifg=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":"v4i36y",
      "typeId":"iot",
      "deviceId":"123"
   },
   "auth":{
      "token": "1234567890"
   }
   }
}
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.")
capecv2.VideoCapture=("video1.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('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='aaa03c23b3724a16a56b629203e3c62c',
    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('12: .2 (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.imwrite(picname+'.jpg',frame)
            multi_part_upload('imagenew', picname+'.jpg', picname+'.jpg')
            json_document={"link":COS_ENDPOINT+'/'+'imagenew'+'/'+picname+'.jpg'}
            new_document=my_database.create_document(json_document)
            if new_document.exists():
                print("Document successfully created.")
            time.sleep(5)
            detect=True
moist=random.randint(0,100)
humidity=random.randint(0, 100)
myData={'Animal':detect,'moisture': moistURE, 'humidity':humidity}
print(myData)
if (humidity!=one):
    client.publishEvent(eventId="status", msgFormat="json", data=myBata, qos=0, onPublish=None)
        client.publishEvent(eventId="status", msgFormat="json", data=myBata, 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()
```

Outputs:

File opened

Alert! Alert! animal detected

Document successfully created.

animal: detected

moisture: 26 humidity: 59

Publish ok..