PROJECT DEVELOPMENT PHASE
SPRINT 1

Date	**18-11-2022**
Team ID	**PNT2022TMID11779**
Project Name	**IoT Based Smart Crop Protection System for Agriculture**

```
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())
Date 16 November 2022
Team ID PNT2022TMID33625
Project Name IOT BASED CROP PROTECTION SYSTEM
FOR AGRICULTURE
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 0620e202302b4508b90eab7efe7475e4'),)
COS_ENDPOINT = "https://s3.jp-tok.cloud-object-storage.appdomain.cloud"
COS_API_KEY_ID = "g5d4qO8EIgv4TWUCJj4hfEzgalqEjrDbE82AJDWlAOHo"
COS_AUTH_ENDPOINT = "https://iam.cloud.ibm.com/identity/token"
COS_RESOURCE_CRN =
"crn:v1:bluemix:public:cloud-objectstorage:global:a/c2fa2836eaf3434bbc8b5b58fefff3f0:62e450fd-4c82-
4153-ba41-ccb53adb8111::"
clientdb = cloudant("apikey-W2njldnwtjO16V53LAVUCqPwc2aHTLmlj1xXvtdGKJBn",
"88cc5f47c1a28afbfb8ad16161583f5a",
url="https://d6c89f97-cf91-48b7-b14b-c99b2fe27c2fbluemix.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(e))
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='e9359dbe6ee44dbc8842ebe97247b201',

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('Dhakshesh', picname+'.jpg', picname+'.jpg')
 json_document={"link":COS_ENDPOINT+'/'+'Dhakshesh'+'/'+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()
```