

Project Development Phase

Sprint 1

Team ID	PNT2022TMID30928
Project Title	lot Based Smart Crop Protection System for Agriculture
Date	21 October 2022

In sprint 1, we designed the python code that links with the sensors to the device and processes the required output. The python code and their output after execution are given below.

Python 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  
ea1de2bb015449679e56e8528e52f3b6'),)
```

```
COS_ENDPOINT =
```

```
"https://control.cloud-object-storage.cloud.ibm.com/v2/endpoints"
```

```
COS_API_KEY_ID =
```

```
"nKCctsU9BUZTYGdSBVnVOTKC7tylC_mzkzOY5JE4EC41"
```

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

```
COS_RESOURCE_CRN =
```

```
"crn:v1:bluemix:public:cloud-object-storage:global:a/421c5b1ee84141  
4ba6ff99bb717f2d98:f19bd8fa-ca1d-45c5-9fc2-a27b4696f047::"
```

```
clientdb =  
Cloudant("apikey-v2-22wp0p0ouimzz2ouyc1r7yvf4gmuete8q4e6h59q  
6fib", "e11dd0b3649fff1eb854677a03d2a42e",  
url="https://apikey-v2-22wp0p0ouimzz2ouyc1r7yvf4gmuete8q4e6h59q  
6fib:e11dd0b3649fff1eb854677a03d2a42e@3a7e7b05-3c45-4d96-a7  
5e-b456e05f3eb6-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(e))

def myCommandCallback(cmd):
    print("Command received: %s" % cmd.data)
    command=cmd.data['command']
    print(command)

    if(commamnd=="lighton"):
        print('lighton')
    elif(command=="lightoff"):
        print('lightoff')
    elif(command=="motoron"):
        print('motoron')
    elif(command=="motoroff"):
        print('motoroff')

myConfig = {
    "identity": {

```

```
    "orgId": "fzb72x",
    "typeId": "ESP-",
    "deviceId": "ESP-"
},
"auth": {
    "token": "9944893843"
}
}
```

```
client = wiotp.sdk.device.DeviceClient(config=myConfig,
logHandlers=None)
client.connect()
```

```
database_name = "sample1"
my_database = clientdb.create_database(database_name)
if my_database.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 = 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='a6100c6f4fb74e79ad8b57b1db2f0235',
```

```
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.imwrite(picname+'.jpg',frame)
multi_part_upload('Jade', picname+'.jpg', picname+'.jpg')

json_document={"link":COS_ENDPOINT+'/'+Jade+'/'+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, onPublish=None)  
    print("Publish Ok..")
```

```
client.commandCallback = myCommandCallback
```

```
cv2.imshow('frame',imS)
```

```
if cv2.waitKey(1) & 0xFF == ord('q'):
```

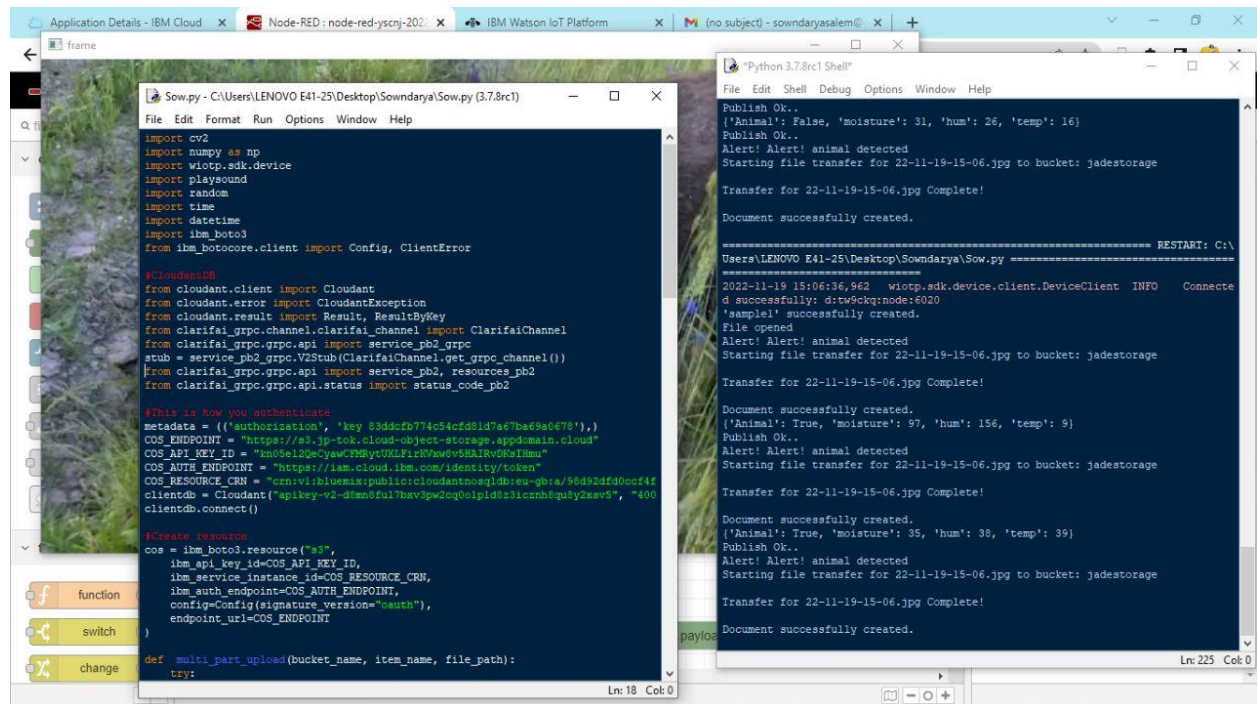
```
    break
```

```
client.disconnect()
```

```
cap.release()
```

```
cv2.destroyAllWindows()
```

OUTPUT:



The screenshot shows a Node-RED workflow on the left and its corresponding Python 3.7.8rc1 shell output on the right. The workflow includes a 'frame' input, a 'function' node, and a 'switch' node. The Python shell output displays the execution of the code, including imports, Cloudant client initialization, and a loop that processes image data and triggers file uploads to a bucket named 'jadestorage'.

```
File Edit Format Run Options Window Help
Sow.py - C:\Users\LENOVO E41-25\Desktop\Sowndarya\Sow.py (3.7.8rc1)
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
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 83d9cfb774c54cf8d81d7a67bae9a0678'),)
COS_ENDPOINT = "https://s3.jp-2024.cloud-object-storage.sfo02.domain.cloud"
COS_API_KEY_ID = "mb0e12e0c9e7c2b5cUkLkRiRvaw6v5BAIRv0d8fma"
COS_AUTH_ENDPOINT = "https://iam.cloud.ibm.com/identity/token"
COS_RESOURCE_CRN = "crn:v1:bluemix:public:cloudantnoql:db:eu-gb:a/98d923df00cf4f"
clientdb = Cloudant("apikey-v2-d8m8ful7bav3pw2cq00lpid8z3icznhs8qu8y2xav5", "400", 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="osauth"),
    endpoint_url=COS_ENDPOINT)

def multi_part_upload(bucket_name, item_name, file_path):
    try:
```

```
Python 3.7.8rc1 Shell
File Edit Shell Debug Options Window Help
Publish Ok..
('Animal': False, 'moisture': 31, 'hum': 26, 'temp': 16)
Publish Ok..
Alert! Alert! animal detected
Starting file transfer for 22-11-19-15-06.jpg to bucket: jadestorage
Transfer for 22-11-19-15-06.jpg Complete!
Document successfully created.
===== RESTART: C:\
Users\LENOVO E41-25\Desktop\Sowndarya\Sow.py =====
2022-11-19 15:06:36,962 wiotp.sdk.device.client.DeviceClient INFO Connecte
d successfully: dtw9ckq;node:6020
'sample1' successfully created.
File opened
Alert! Alert! animal detected
Starting file transfer for 22-11-19-15-06.jpg to bucket: jadestorage
Transfer for 22-11-19-15-06.jpg Complete!
Document successfully created.
('Animal': True, 'moisture': 97, 'hum': 156, 'temp': 9)
Publish Ok..
Alert! Alert! animal detected
Starting file transfer for 22-11-19-15-06.jpg to bucket: jadestorage
Transfer for 22-11-19-15-06.jpg Complete!
Document successfully created.
('Animal': True, 'moisture': 35, 'hum': 38, 'temp': 39)
Publish Ok..
Alert! Alert! animal detected
Starting file transfer for 22-11-19-15-06.jpg to bucket: jadestorage
Transfer for 22-11-19-15-06.jpg Complete!
Document successfully created.
payload
Ln: 225 Col: 0
```