

AUTOMATIC GARAGE DOOR OPENER

Final Project

Team –

SANSKAR VIDYARTHI (18BEC0013)

JAVVAJI JASWANTH SAI (18BEC0511)

RISHI RANJAN (19BEC0798)

ROHAN CHANDRASHEKAR (20BCE0765)

1. Introduction

Purpose – To construct an IoT based automatic garage door opening system.

Overview –

- Automatic opening of the garage doors by detecting the authorized vehicles.
- The garage door will be integrated with the camera which will detect the car it will automatically open the garage doors to park the car.
- the lights will be switched on if the garage door is opened and it will be switched off if the door is closed.
- He can also control the garage doors and lights using the mobile app.
- All the captured images of the vehicles will be stored.

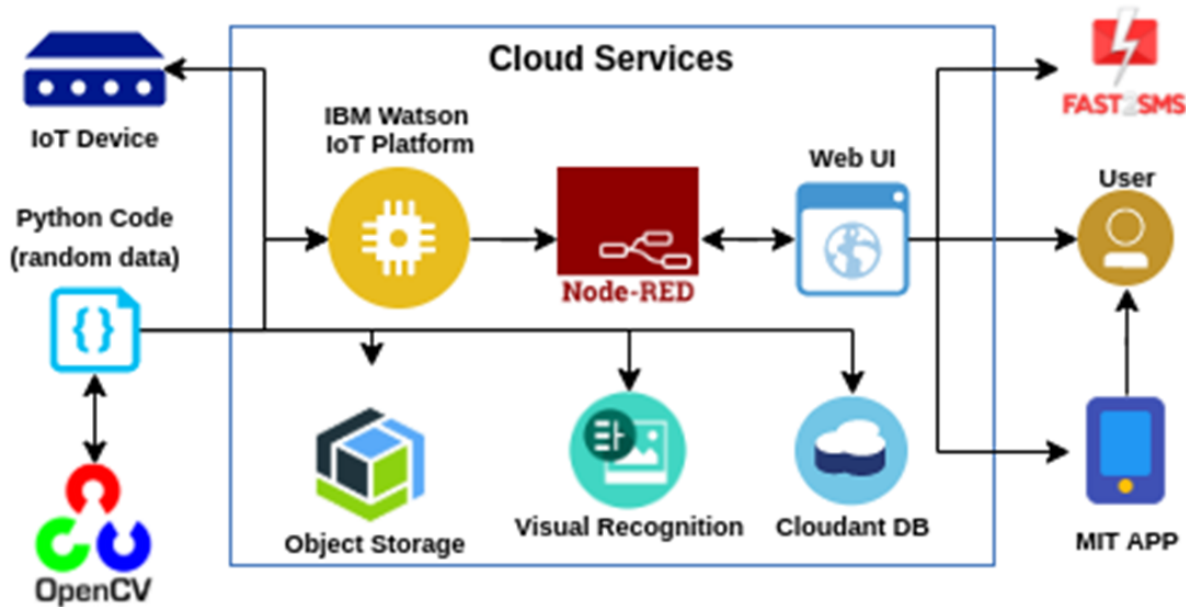
2. Literature survey

Existing problem – Every time there's a vehicle to be parked or taken out of the garage, someone or the driver himself has to open and close the garage door which is very inefficient and could be a time-consuming process especially in public garages, parking lots and basement parking. During night time, lighting could also be an issue.

Proposed solution – The solution is to implement an automatic garage door opener wherever required with both automatic and manual controls of opening and closing of the garage along with light controls. Here, we would use an ultrasonic sensor (python code alternate) to detect the proximity of the vehicle and open the door and Node-Red-Web-UI or mobile application to get the most recent image of the vehicle being parked which is beneficial for security purposes too. The app could also be used to control the lighting. The data of the vehicle parked is also recorded over the cloudant database.

3. Theoretical Analysis

Block Diagram



Software Designing

We have used clarifai to check if the object is a vehicle. Python code is used to create random integers between 5 and 500 which is used as sensor data to measure distance of vehicle from the garage door. The threshold is set as 15 so when the value generated is less than 15, the garage door opens, image is captured and stored into the cloud and the lights switch on. We've used Node Red for the Web UI and IBM cloud to store the data.

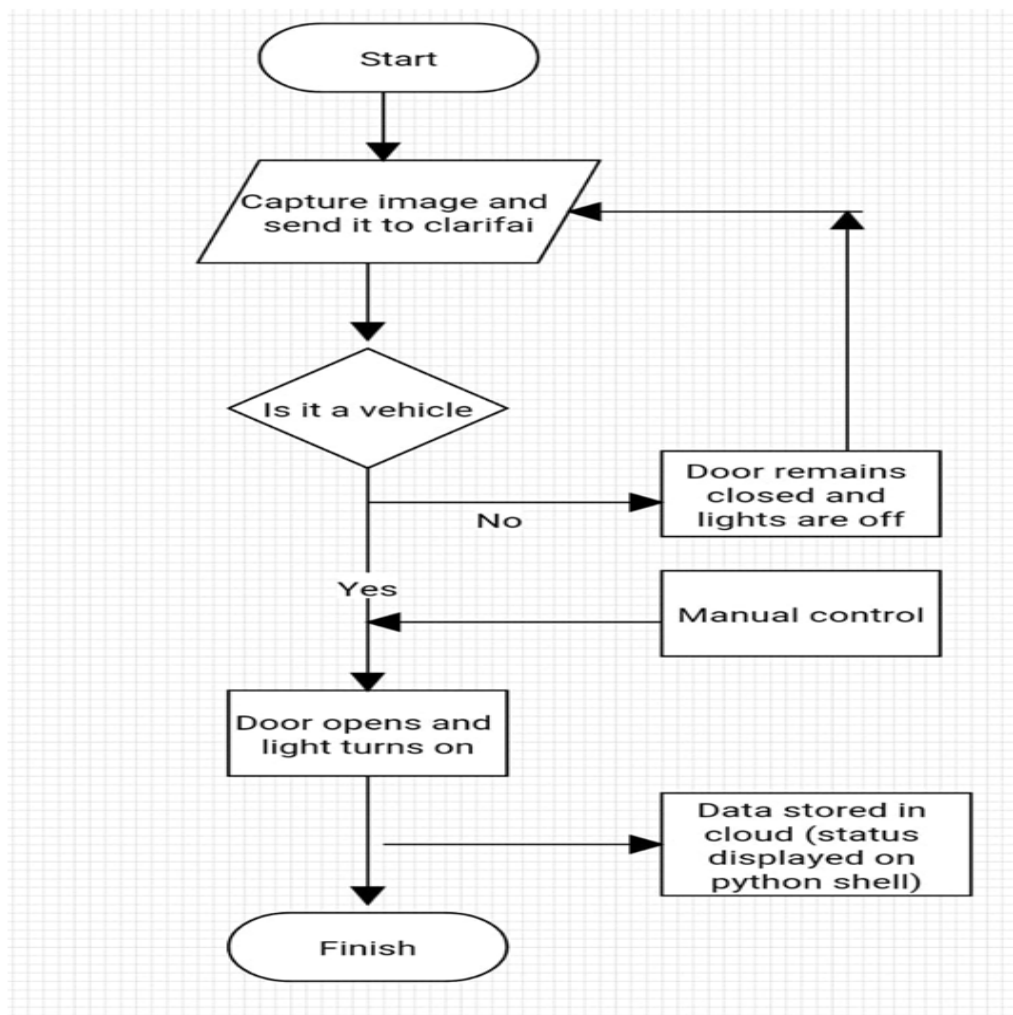
4. Experimental Investigations

In the python program, instead of the sensor, we get random distance values from 5 to 500 meters. If the distance is less than 15 meters then we take a picture and send it to clarifai to check if it is a vehicle. If it is a vehicle then we open the door and switch on the lights and send the image to cloud and image details(including URL) to cloudant database.

From the cloudant, we take the URL of the latest image and display it in the node-red web UI. In the web UI, the user has buttons to control the door and light. Whatever they choose, it is displayed in python shell.

In mobile app made using MIT App Inventor, again we take the URL from the database and display the latest image. There again, there are buttons to control door and light. The user selection is sent to node-red from where it is displayed in python shell.

5. Flowchart



6. Result

We receive the most recent image of the vehicle being parked as we press the get image button. The message of garage door being opened and lights being turned on and off are received on the python shell when their respective event occurs.

7. Advantages and Applications

1. Convenience

One of the main benefits of an automatic garage door opener is convenience. You can stay in your car until you're in (or out of) your garage, so you don't have to get out of your car to open or close your garage.

You won't need to leave your children or dog in the car, or go outside in bad weather just to open or close your garage door.

2. Lighting

Modern automatic garage door openers are available with built in lights to will illuminate your garage or the path to your garage. This type of lighting can be invaluable at night, in the winter or in bad weather.

3. Security

Garage security is of paramount importance, and so you'll want to make sure that your automatic garage door opener helps to keep your family, home, and possessions safe. That's when the image capture feature comes in handy.

4. Safety

You won't need to worry about the garage door crashing down on something or somebody in the way, or fingers getting caught anywhere. For those with a medical condition such as arthritis, or young children, one of these could be essential.

5. Automatic closing

Automatic closing means that you won't need to get out of your car to close the door, or remember to press the button on your remote; your garage door opener can close itself too. This is ideal if you have children, often forget to close your garage door, or get to work and worry that you may not have closed your garage.

6. Low energy use

Although automatic garage door openers can be highly advanced and technical, they do not need a lot of power to work. This makes them highly cost effective to run, and less damaging to the environment.

7. Technology

Modern garage door openers are available with various features and facilities to provide you with more convenience. Some other other benefits of an automatic garage door opener include the ability to be controlled by smart phone apps. These enable you to ensure your door is closed, wherever you are, or to sound an alarm to let you know your door is open.

8. Maintenance

One the benefits of an automatic garage door opener is that it doesn't require much maintenance, but you'll want to keep yours in accordance to the manufacturer's recommendations, so that it always works when you need it to.

9. Cost

Despite their convenience and features, automatic garage doors are relatively inexpensive to buy and fit. When you think about what they offer, and what they mean to a family with a couple of cars, kids who keep their bikes in the garage and who are concerned about security, they are an absolute bargain.

10. Peace of mind

Whether you simply want to know that your garage door is closed, have a medical condition and can't open your door yourself, or simply want to save a few minutes every day, an automatic garage door open makes a lot of sense to many people. What seems like an unnecessary luxury becomes invaluable if you use your garage a lot.

8. Disadvantages

There are a few downsides in the use of garage doors, including:

- The installation process requires the help of a professional garage door technician in order for it to be accurate and efficient.
- Automatic garage doors are more costly than manual garage doors.
- Automatic garage doors require regular maintenance and checkup which costs money. This means that the expenses do not end upon installation.
- They are also more complex to clean since you also need to polish internal

parts like springs and bolts to avoid the accumulation of rust.

- Automatic garage door repairs can be costly especially when some parts need to be replaced.

9. Conclusions

In this project we use an ultrasonic sensor (random values in python for sensor data as physical hardware is not available) to measure the distance of the vehicle from the garage and automatically open the garage when the vehicle crosses a set proximity limit. The garage door can also be manually opened using a mobile application or Node Red web UI. The information and the image of the vehicle is stored over the cloudant database and the image can be seen in the web UI as well.

10. Future Scope

In the future an alarm system could be attached to the project which will ring if an unregistered vehicle or person is detected in the proximity range of the garage door and put the garage on lock down and instantly inform the owner about the breach.

11. Bibliography

1. Smart Internz internship reference materials
2. Softwares – MIT App Inventor, IBM Watson, Clarifai, Python IDLE, Node-Red
3. AA Garage doors, “10 Benefits of an Automatic Garage Door Opener”, 8th June 2016

12. APPENDIX

i) Python Code-

```
#Vehicle Detection
```

```
import cv2
```

```
#IBM IOT connection
```

```
import wiotp.sdk.device
```

```
#IBM COS
```

```
import ibm_boto3
```

```
#IBM Cloudant
```

```
from ibm_botocore.client import Config, ClientError
```

```
from ibmcloudant.cloudant_v1 import CloudantV1
```

```
from ibmcloudant import CouchDbSessionAuthenticator
```

```
from ibm_cloud_sdk_core.authenticators import BasicAuthenticator
```

```
#Clarifai
```

```
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
```

```
#Miscellaneous
```

```
import time
```

```
import random
```

```
import datetime
```

```
#IOT device connection
```

```
myConfig = {
```

```
    "identity": {
```

```
        "orgId": "j8rgpm",
```

```
        "typeId": "First_Device",
```

```
        "deviceId": "123"
```



```

    },
    "auth": {
        "token": "First_Device_123"
    }
}

```

Constants for IBM COS values

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

```
COS_API_KEY_ID = "78NLIHvfEUxheNWr4jehqUGifUPewWVaPrk3N2HXaMKy"
```

```
COS_INSTANCE_CRN =
"crn:v1:bluemix:public:cloud-object-storage:global:a/abaf1723c9c84a148e04f11a73d2442a:880edc7f-ce14-4df9-8545-5c592c737b87::"
```

Create resource

```

cos = ibm_boto3.resource("s3",
    ibm_api_key_id=COS_API_KEY_ID,
    ibm_service_instance_id=COS_INSTANCE_CRN,
    config=Config(signature_version="oauth"),
    endpoint_url=COS_ENDPOINT
)

```

```

authenticator = BasicAuthenticator('apikey-v2-10xypw9nevga82oqb993uwcfdqgl748fcoiznj3jfrzn',
'3fb54d29a2d64e40f7a0657467d33a27')

```

```
service = CloudantV1(authenticator=authenticator)
```

```

service.set_service_url('https://apikey-v2-10xypw9nevga82oqb993uwcfdqgl748fcoiznj3jfrzn:3fb54d29a2d64e40f7a0657467d33a27@c6a728a6-a633-4d59-a8e9-c06a612f8176-bluemix.cloudantnosqldb.appdomain.cloud')

```

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

```
client.connect()
```

```

#Image upload to COS bucket

bucket = "rohanbucket"

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))
        part_size = 1024 * 1024 * 5
        file_threshold = 1024 * 1024 * 15
        transfer_config = ibm_boto3.s3.transfer.TransferConfig(
            multipart_threshold=file_threshold,
            multipart_chunksize=part_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))

```

```

video = cv2.VideoCapture('cars.avi')

```

```

# Clarifai Authentication

```

```

metadata = (('authorization', 'Key f9c413355966419bbd1b0e0125a9a5bc'),)

```

```
#Initial Conditions
```

```
print("Door is Closed")
```

```
print("Light is Off")
```

```
def myCommandCallback(cmd):
```

```
    m=cmd.data['command']
```

```
    print(m,"\n")
```

```
while True:
```

```
    distance=random.randint(5,500) # random distance values from 5 to 500 meters
```

```
    check,frame=video.read()
```

```
    frame = cv2.resize(frame, (600,400))
```

```
    if (distance<=15): #picture is sent to clarifai when distance is less than 15 meters
```

```
        myData={'distance':distance}
```

```
        print(myData)
```

```
        client.publishEvent(eventId="status", msgFormat="json", data=myData, qos=0, onPublish=None)
```

```
        picname=datetime.datetime.now().strftime("%y-%m-%d-%H-%M-%S")
```

```
        path="D:\\rohan\\Personal\\Courses\\IOT\\Final Project\\pics\\" +picname + ".jpg"
```

```
        cv2.imwrite(picname+".jpg",frame) # Images are stored locally in a folder
```

```
        time.sleep(1)
```

```
        with open(path, "rb") as f:
```

```
            file_bytes = f.read()
```

```
request = service_pb2.PostModelOutputsRequest(
```

```
    # This is the model ID of a publicly available General model. You may use any other public or custom model ID.
```

```
    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))

a= []

for concept in response.outputs[0].data.concepts:
    if(concept.value > 0.8):
        a.append(concept.name)

t=1

for i in a:
    if(i == "car" or i == "vehicle" or i=="bike"):
        print("Vehicle is detected\n")
        print("Door is open\nLight is On")
        #if vehicle is detected document is uploaded to cloudant database
        multi_part_upload(bucket, picname+'.jpg', picname+'.jpg')
        json_document={"link":COS_ENDPOINT+'/'+bucket+'/'+picname+'.jpg'}
        response = service.post_document(db='sample', document=json_document).get_result()
        break

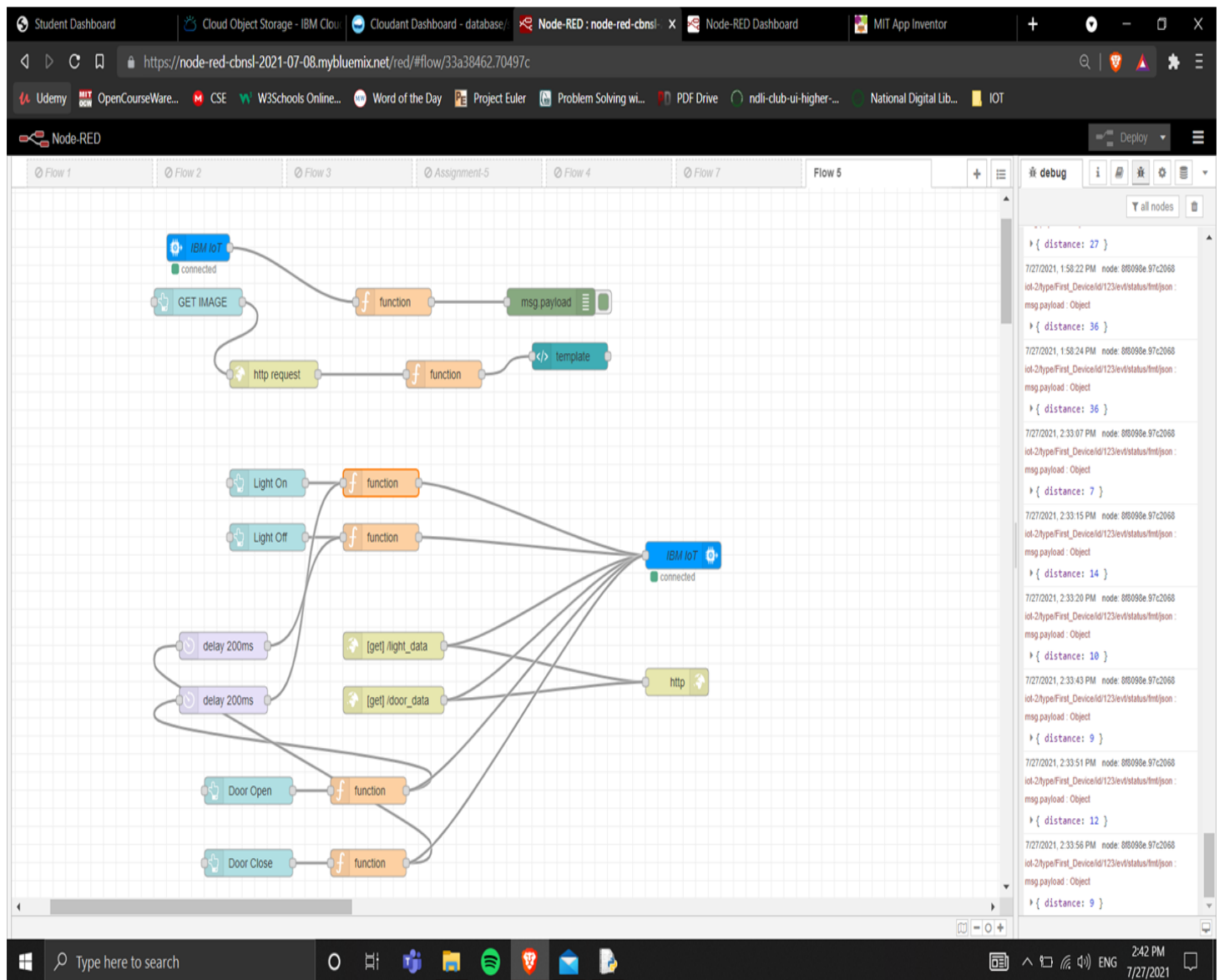
Key=cv2.waitKey(1)

client.commandCallback = myCommandCallback

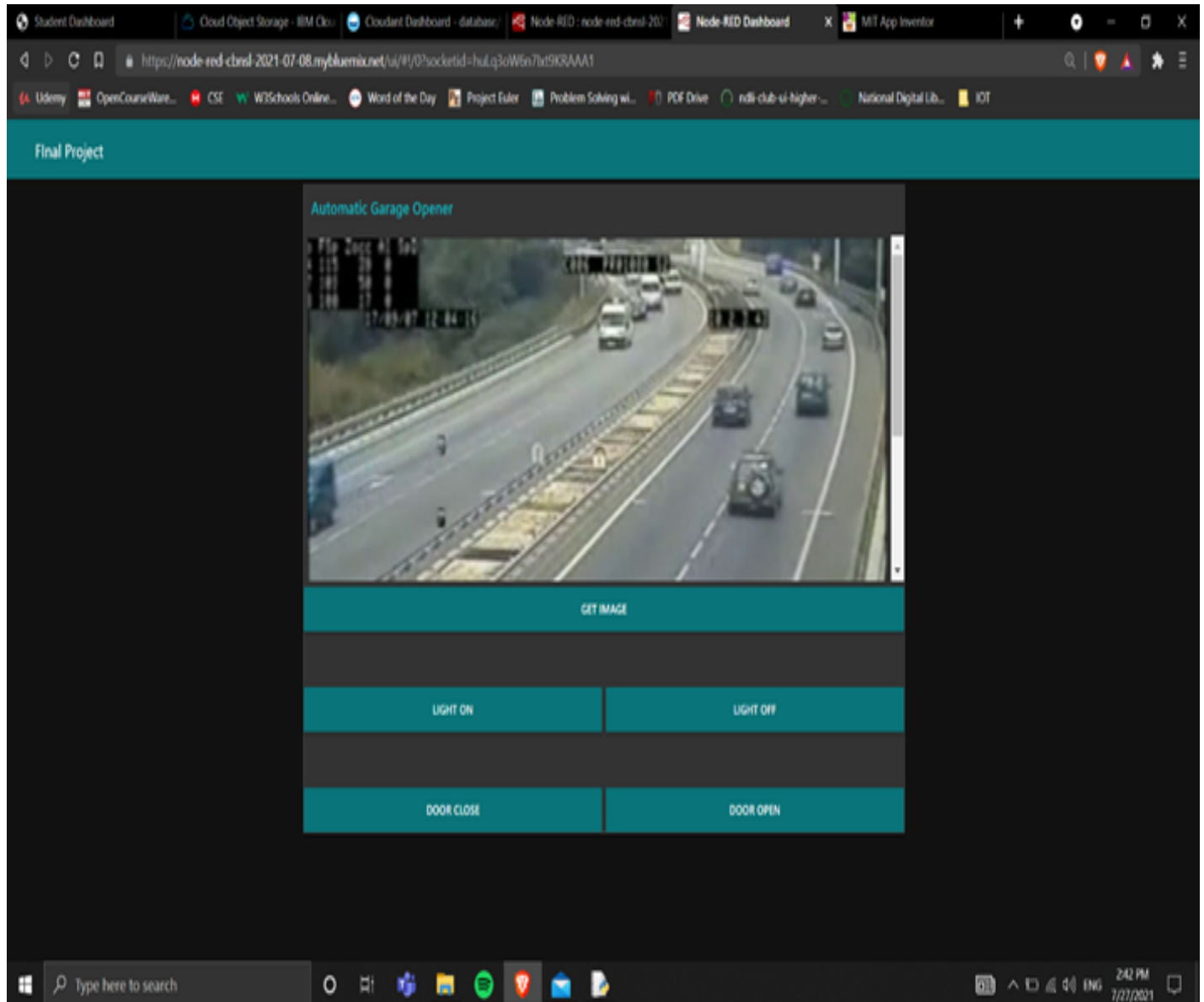
client.disconnect()

```

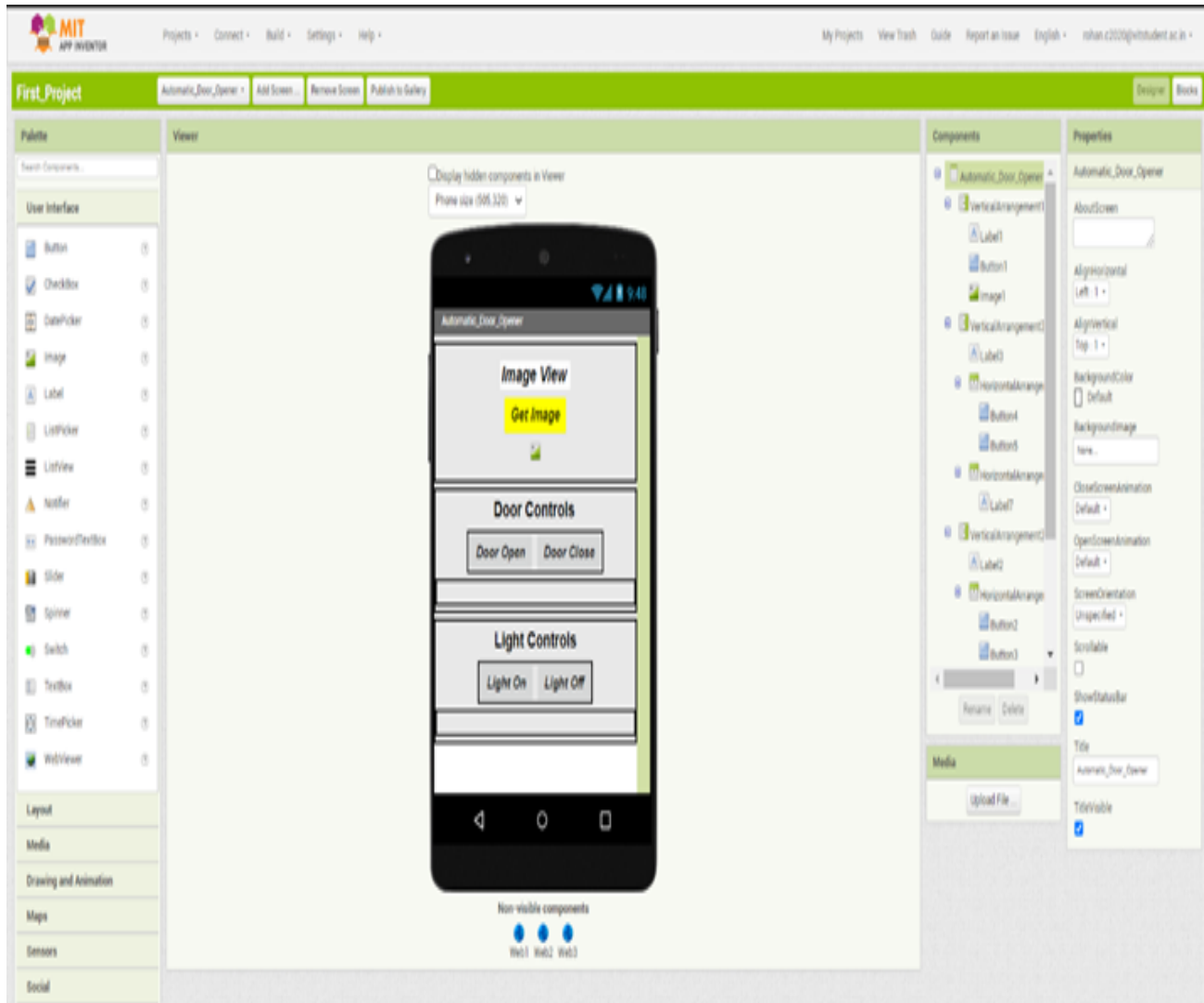
ii) Node Red-



WEB UI-



iii) MOBILE APP-



Backend for the App -

The image shows a Scratch script designed to act as a backend for a smart home application. It features several global variables and three main event-driven blocks.

Global Variables:

- `global 1` to `2`
- `global 3` to `6`
- `global list` to `0`
- `global a2` to `1`
- `global a4` to `0`

When Button1 is Clicked:

- Set `Web1` `Uri` to `https://c6a728a6-a633-4d59-abe9-c05a6128176-blu`
- Call `Web1` `Get`

When Web1 Got Text:

- Set `global list` to `call Web1` `JsonTextDecode` `jsonText` `get responseContent`
- Set `Image1` `Picture` to a loop of `select list item list` followed by `index` and `get global a` (repeated 8 times).

When Button2 is Clicked:

- Do `if` `0` `Label5` `Text` `is` `Light is On` `or` `Label5` `Text` `is` `Light is already On`
- Then `set Label5` `Text` to `Light is already On`
- Set `Web2` `Uri` to `https://node-red-cnsl-2021-07-08.mybluemix.net/...`

The interface includes a 'Show Warnings' button on the left and a vertical toolbar on the right with icons for zooming, undo, redo, and deleting.

First Project

Automatic_Door_Opener Add Screen ... Remove Screen Publish to Gallery

Designer Blocks

Blocks

Built-in

- Control
- Logic
- Math
- Text
- Lists
- Extensions
- Colors
- Variables
- Procedures

Automatic_Door_Opener

VerticalManagement1

Label1

Button1

Image1

VerticalManagement2

Label2

Label3

Label4

Label5

Label6

Label7

Label8

Label9

Label10

Label11

Label12

Label13

Label14

Label15

Label16

Label17

Label18

Label19

Label20

Label21

Label22

Label23

Label24

Label25

Label26

Label27

Label28

Label29

Label30

Label31

Label32

Label33

Label34

Label35

Label36

Label37

Label38

Label39

Label40

Label41

Label42

Label43

Label44

Label45

Label46

Label47

Label48

Label49

Label50

Label51

Label52

Label53

Label54

Label55

Label56

Label57

Label58

Label59

Label60

Label61

Label62

Label63

Label64

Label65

Label66

Label67

Label68

Label69

Label70

Label71

Label72

Label73

Label74

Label75

Label76

Label77

Label78

Label79

Label80

Label81

Label82

Label83

Label84

Label85

Label86

Label87

Label88

Label89

Label90

Label91

Label92

Label93

Label94

Label95

Label96

Label97

Label98

Label99

Label100

Label101

Label102

Label103

Label104

Label105

Label106

Label107

Label108

Label109

Label110

Label111

Label112

Label113

Label114

Label115

Label116

Label117

Label118

Label119

Label120

Label121

Label122

Label123

Label124

Label125

Label126

Label127

Label128

Label129

Label130

Label131

Label132

Label133

Label134

Label135

Label136

Label137

Label138

Label139

Label140

Label141

Label142

Label143

Label144

Label145

Label146

Label147

Label148

Label149

Label150

Label151

Label152

Label153

Label154

Label155

Label156

Label157

Label158

Label159

Label160

Label161

Label162

Label163

Label164

Label165

Label166

Label167

Label168

Label169

Label170

Label171

Label172

Label173

Label174

Label175

Label176

Label177

Label178

Label179

Label180

Label181

Label182

Label183

Label184

Label185

Label186

Label187

Label188

Label189

Label190

Label191

Label192

Label193

Label194

Label195

Label196

Label197

Label198

Label199

Label200

Label201

Label202

Label203

Label204

Label205

Label206

Label207

Label208

Label209

Label210

Label211

Label212

Label213

Label214

Label215

Label216

Label217

Label218

Label219

Label220

Label221

Label222

Label223

Label224

Label225

Label226

Label227

Label228

Label229

Label230

Label231

Label232

Label233

Label234

Label235

Label236

Label237

Label238

Label239

Label240

Label241

Label242

Label243

Label244

Label245

Label246

Label247

Label248

Label249

Label250

Label251

Label252

Label253

Label254

Label255

Label256

Label257

Label258

Label259

Label260

Label261

Label262

Label263

Label264

Label265

Label266

Label267

Label268

Label269

Label270

Label271

Label272

Label273

Label274

Label275

Label276

Label277

Label278

Label279

Label280

Label281

Label282

Label283

Label284

Label285

Label286

Label287

Label288

Label289

Label290

Label291

Label292

Label293

Label294

Label295

Label296

Label297

Label298

Label299

Label300

Label301

Label302

Label303

Label304

Label305

Label306

Label307

Label308

Label309

Label310

Label311

Label312

Label313

Label314

Label315

Label316

Label317

Label318

Label319

Label320

Label321

Label322

Label323

Label324

Label325

Label326

Label327

Label328

Label329

Label330

Label331

Label332

Label333

Label334

Label335

Label336

Label337

Label338

Label339

Label340

Label341

Label342

Label343

Label344

Label345

Label346

Label347

Label348

Label349

Label350

Label351

iv) Cloud Object Storage-

The screenshot displays the IBM Cloud Object Storage console interface. The top navigation bar includes the IBM Cloud logo, a search bar, and links to Catalog, Docs, Support, and Manage. The user's name, Rohan Chandra, is visible in the top right corner. The main content area shows the 'rohanbucket' bucket with a table of objects. The table has columns for Object name, Archived status, Size, and Last modified. The objects listed are all .jpg files with names starting with '21-07-27-14-13-58.jpg' and ending with '21-07-27-14-14-54.jpg'. The sizes range from 64.4 KB to 65.9 KB, and the last modified dates are all 2021-07-27. The console also shows a 'Usage details' section on the left and a 'Transfers' section on the right. The bottom of the console features a 'Drag and drop files (objects) here or click to upload' instruction and a pagination bar showing 'page 1' of 1-10 items.

Object name	Archived	Size	Last modified
21-07-27-13-58.jpg		65.3 KB	2021-07-27 1:58 PM
21-07-27-14-13-08.jpg		65.6 KB	2021-07-27 2:13 PM
21-07-27-14-14-28.jpg		65.0 KB	2021-07-27 2:14 PM
21-07-27-14-14-38.jpg		65.9 KB	2021-07-27 2:14 PM
21-07-27-14-14-42.jpg		65.4 KB	2021-07-27 2:14 PM
21-07-27-14-14-45.jpg		65.0 KB	2021-07-27 2:14 PM
21-07-27-14-14-48.jpg		64.5 KB	2021-07-27 2:14 PM
21-07-27-14-14-50.jpg		64.4 KB	2021-07-27 2:14 PM
21-07-27-14-14-54.jpg		65.4 KB	2021-07-27 2:14 PM

Cloudant Database-

The image displays two screenshots of the Cloudant Dashboard interface, showing a list of documents in a database named 'sample'.

Top Screenshot: The dashboard shows a list of 14 documents. The columns are '_id' and 'link'. The documents are sorted by '_id' in descending order.

_id	link
13fb3920c65d29280392b1749cf3f59	https://s3.jp-tok.cloud-object-storage.appdomain.cloud/rohanbucket/2...
13fb3920c65d29280392b1749cf4c3a	https://s3.jp-tok.cloud-object-storage.appdomain.cloud/rohanbucket/2...
2762b84a82244cf505fbcba25a187f17	https://s3.jp-tok.cloud-object-storage.appdomain.cloud/rohanbucket/2...
279a8ab4d3aa13d1116a719d3528c543	https://s3.jp-tok.cloud-object-storage.appdomain.cloud/rohanbucket/2...
2b0280c2d6431786269eb51c19582673	https://s3.jp-tok.cloud-object-storage.appdomain.cloud/rohanbucket/2...
5f33af93c026582d5e9694f0f66dce	https://s3.jp-tok.cloud-object-storage.appdomain.cloud/rohanbucket/2...
648e93a3f7fe63c2d65b8d1313495976	https://s3.jp-tok.cloud-object-storage.appdomain.cloud/rohanbucket/2...
648e93a3f7fe63c2d65b8d13134977b7	https://s3.jp-tok.cloud-object-storage.appdomain.cloud/rohanbucket/2...
648e93a3f7fe63c2d65b8d1313541315	https://s3.jp-tok.cloud-object-storage.appdomain.cloud/rohanbucket/2...
747faa0824b160396701fb1dae07a56	https://s3.jp-tok.cloud-object-storage.appdomain.cloud/rohanbucket/2...
85d748facace9439753f2cbd8bab0ae	https://s3.jp-tok.cloud-object-storage.appdomain.cloud/rohanbucket/2...
85d748facace9439753f2cbd8bab63b	https://s3.jp-tok.cloud-object-storage.appdomain.cloud/rohanbucket/2...
M38N5rR3ar94M7d899Vaa77x1h3a7f3A	https://s3.jp-tok.cloud-object-storage.appdomain.cloud/rohanbucket/2...

Bottom Screenshot: The dashboard shows a list of 30 documents. The columns are '_id' and 'link'. The documents are sorted by '_id' in descending order.

_id	link
648e93a3f7fe63c2d65b8d1313495976	https://s3.jp-tok.cloud-object-storage.appdomain.cloud/rohanbucket/2...
648e93a3f7fe63c2d65b8d13134977b7	https://s3.jp-tok.cloud-object-storage.appdomain.cloud/rohanbucket/2...
648e93a3f7fe63c2d65b8d1313541315	https://s3.jp-tok.cloud-object-storage.appdomain.cloud/rohanbucket/2...
648e93a3f7fe63c2d65b8d131359e05e	https://s3.jp-tok.cloud-object-storage.appdomain.cloud/rohanbucket/2...
648e93a3f7fe63c2d65b8d13135a67c0	https://s3.jp-tok.cloud-object-storage.appdomain.cloud/rohanbucket/2...
747faa0824b160396701fb1dae07a56	https://s3.jp-tok.cloud-object-storage.appdomain.cloud/rohanbucket/2...
747faa0824b160396701fb1dae69407	https://s3.jp-tok.cloud-object-storage.appdomain.cloud/rohanbucket/2...
85d748facace9439753f2cbd8bab0ae	https://s3.jp-tok.cloud-object-storage.appdomain.cloud/rohanbucket/2...
85d748facace9439753f2cbd8bab63b	https://s3.jp-tok.cloud-object-storage.appdomain.cloud/rohanbucket/2...
85d748facace9439753f2cbd8d660dac	https://s3.jp-tok.cloud-object-storage.appdomain.cloud/rohanbucket/2...
K0c6cf12431f38390c047fa4389e5b	https://s3.jp-tok.cloud-object-storage.appdomain.cloud/rohanbucket/2...
K0c6cf12431f38390c047fa43906d6	https://s3.jp-tok.cloud-object-storage.appdomain.cloud/rohanbucket/2...
M3805c83ac94f2d896ee72d1b0a036	https://s3.jp-tok.cloud-object-storage.appdomain.cloud/rohanbucket/2...
M3805c83ac94f2d896ee72d1c240b4	https://s3.jp-tok.cloud-object-storage.appdomain.cloud/rohanbucket/2...
fb7840d6dd177c46efeb382ea49ad54	https://s3.jp-tok.cloud-object-storage.appdomain.cloud/rohanbucket/2...

Python Shell Output-

```
Python 3.9 Shell
File Edit Shell Debug Options Window Help

Python 3.9.0 (tags/v3.9.0:9cf6752, Oct 5 2020, 15:34:40) [MSC v.1927 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: D:\rohan\Personal\Courses\IoT\Final Project\pics\final.py =====
2021-07-27 14:46:56.909 wiotp.sdk.device.client.DeviceClient INFO Connected successfully: dij8rgpm:First_Device:123
Door is Closed
Light is Off
{'distance': 8}
Vehicle is detected

Door is open
Light is On
Starting file transfer for 21-07-27-14-46-57.jpg to bucket: rohanbucket

Transfer for 21-07-27-14-46-57.jpg Complete!

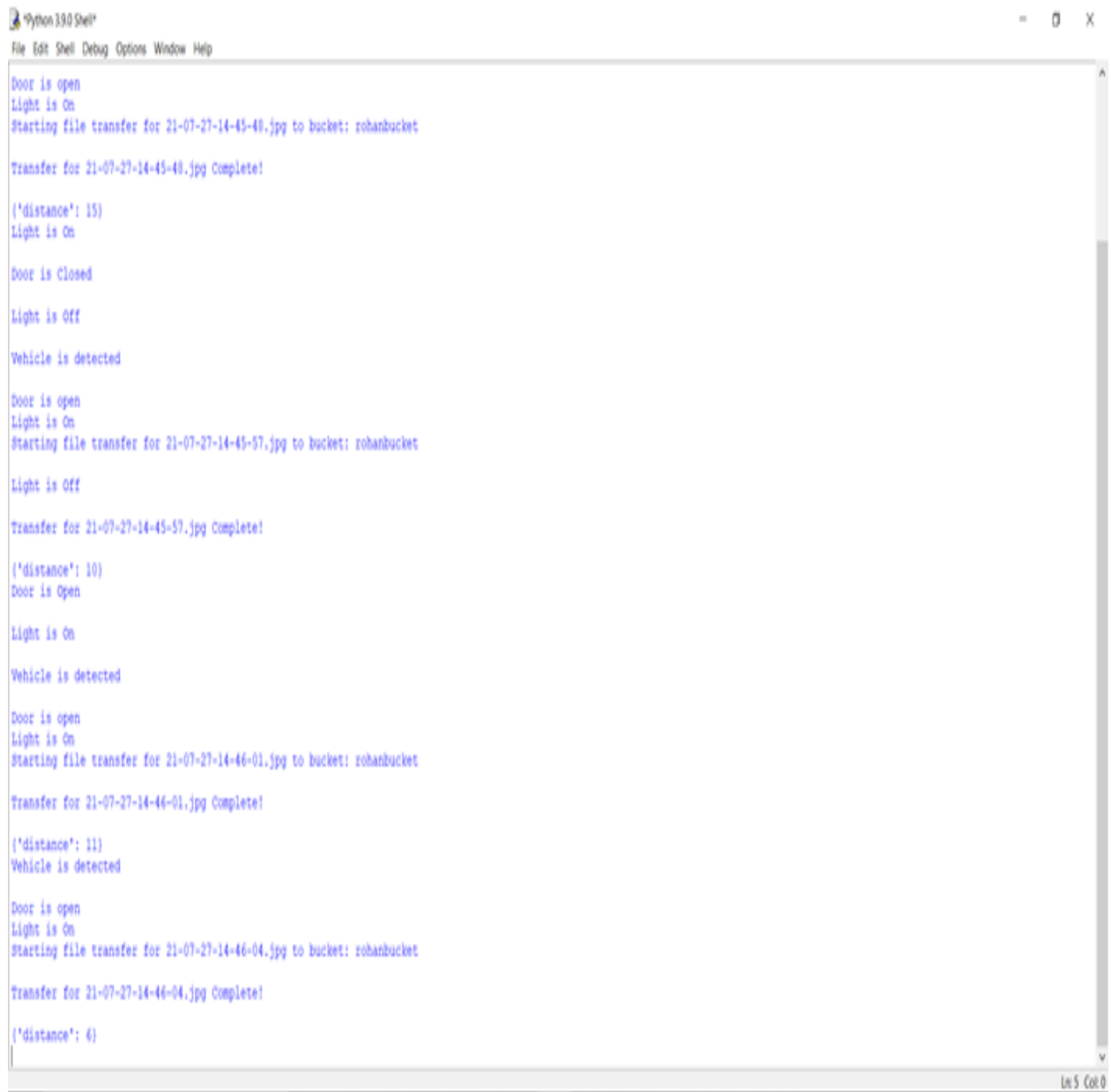
{'distance': 7}
Light is On

Vehicle is detected

Door is open
Light is On
Starting file transfer for 21-07-27-14-47-03.jpg to bucket: rohanbucket

Transfer for 21-07-27-14-47-03.jpg Complete!

{'distance': 14}
```



```
Python 3.9 Shell
File Edit Shell Debug Options Window Help

Door is open
Light is On
Starting file transfer for 21-07-27-14-45-48.jpg to bucket: rohanbucket

Transfer for 21-07-27-14-45-48.jpg Complete!

({'distance': 15})
Light is On

Door is Closed

Light is Off

Vehicle is detected

Door is open
Light is On
Starting file transfer for 21-07-27-14-45-57.jpg to bucket: rohanbucket

Light is Off

Transfer for 21-07-27-14-45-57.jpg Complete!

({'distance': 10})
Door is Open

Light is On

Vehicle is detected

Door is open
Light is On
Starting file transfer for 21-07-27-14-46-01.jpg to bucket: rohanbucket

Transfer for 21-07-27-14-46-01.jpg Complete!

({'distance': 11})
Vehicle is detected

Door is open
Light is On
Starting file transfer for 21-07-27-14-46-04.jpg to bucket: rohanbucket

Transfer for 21-07-27-14-46-04.jpg Complete!

({'distance': 6})

Ln 5 Col 0
```

Link for working video of the project-

https://drive.google.com/file/d/1zNX_MmuEzVsTT6IHcvUY1QoT4O_g_y_2/view?usp=sharing