London Metropolitan University

CS7080 Cloud Computing and Internet Of Things

Coursework – II

Automated Covid -19 Gate Pass using Temperature Monitor

Group members:

NAME | 20xxxxxx

NAME | 20xxxxxx

Submitted To:

Dr Shahnoor Shanta

Submission Date: 10th June 2021

Table of Contents

1. Introduction

In recent years, the increasing use of technology and smart devices in the field of health has led to dramatic advances in health care practices around the world. In addition, the facilities that IoT and cloud computing provided to remotely control people's health such as controlling blood pressure, body temperature, heart rate, and real-time monitoring on the status of individuals have developed unprecedented methods of healthcare through the integration of physical devices and the Internet (Babu, et al., 2016). On the other hand, the rapid spread of the coronavirus, scientifically known as Covid 19, declared by the World Health Organization in 2019, has affected various aspects of human life from major businesses to daily chores. The Covid-19, which is known as a part of the Mers-COV family of viruses and was first transmitted to humans from bats is associated with mild to intense symptoms of fever, cough, and respiratory problems (Kumar, et al., 2020). The role of IoT technologies in contributing to tracking and preventing the spread of the virus with higher accuracy, efficient monitoring, and real-time diagnosis has been considerable (Ndiaye, et al., 2020). Further, the extreme use of control systems manually has been known as a source of the virus itself which is why this research aims to design and develop an affordable, real-time, scalable, efficient, and remote body temperature control and monitoring system to decrease the probability and risk of the COVID-19 spread. Application of embedded systems and sensors to detect vital signs during surgery in people with Covid-19 disease who had a previous history of asthma and diabetes (Javaid & HaleemKhan, 2021), body temperature measurement, and Face recognition of affected individuals (Kumar, et al., 2020), identification of the location of the infected person by monitoring gadgets and AI-based forecasting system for detection of violations of social distancing (KamalAbdulah , et al., 2020), new remote virus diagnostic systems based on a thermal image without human contacts (Abdulrazaq, et al., 2020) , face mask detection system and Mask type detection (Hussain , et al., 2021) have been the prominent epitomes of application of IoT and Cloud Computing in preventing and controlling the spread of the virus. An Automated Covid -19 Gate Pass system, which is supposed in this paper, provides a smart platform to regulate, control, and alert for access to operation and identify any signs of virus infection in individuals. Similarly, the body temperature measurement is considered as the primary factor to meet the condition of passing the gates using IoT technologies such as MLX90614 non-contact temperature sensor module, Pi-camera, buzzer, and pocket-sized

raspberry pi provide real-time status of parameters in the cloud hence controlling the gate remotely by permitting individuals to pass the gate. The proposed model, which is illustrated in this paper is practically deployed in the Microsoft Azure cloud platform to send, receive and process the data provided by a raspberry pi. In the following, the steps of implementing the proposed model, advantages, and challenges are discussed in detail.

2. Literature review

The IoT applications, which use a network of interconnected devices and sensors to receive, send, and process data, have made tremendous progress in the new generation of health networks. Providing cloud-based health services is one of the most effective IoT applications in reducing the time and cost of accessing these services. In this application, patients are connected to the Cloud application manager by their mobile phones and their geographical location is updated in the cloud environment at the moment which is identified via mobile GPS. Upon demand, health care providers will fulfill the required health services based on patients' status priorities and location (Ghanavati, et al., 2017). During the Coronavirus pandemic, which requires more remote health care, many solutions have been proposed to monitor the status of patients remotely using the IoT. One of these practical applications in the development of portable smart devices such as smartwatches for patients which can receive the real-time status information of patients such as heart rate, body temperature, and room temperature through embedded sensors and store to the relevant servers using the Wi-Fi module (Valsalan, et al., 2020). The results of similar researches show the accuracy and success of these systems up to 97% in open spaces up to 67 meters' distance and indoors up to 16 meters with obstacles (Sollu, et al., 2018). Similarly, in another study proposed a self-monitoring healthcare system utilizing raspberry pi, the heart rate, and body temperature parameters were stored in the Bluemix cloud through the Bluemixuses MQTT (Message Queuing Telemetry Transport) protocol, and the current status of the received parameters could be illustrated at IBM The Watson IoT platform to both the patient and the physician concerned (Kaur & Jasuja, 2017). Furthermore, detection of the face mask use, which became compulsory in all public places at the time of the widespread Covid-19 prevalence to prevent the spread of the virus, is another common use of IoT, cloud computing, and machine learning algorithms to save time and more accurate and intelligent monitoring individuals passing the gates. For instance, the application of fog nodes located in different controller gates to send a streaming video to deploy Mobile

Net face mask detection models. This model also applies "Haar-cascade-classifiers" to identify the face details in the received video. In this control system, only people are allowed to pass the gate if they have used the mask correctly defined in classifications (Rudraraju, et al., 2020). Edge computing is another typical application of face mask detection which consists of 4 overall phases including storing data sets, Data training, edge computing for face mask detection and finally sending the results to the server. As well, in the mentioned application, edge computing can be used to restore the low-quality streaming video to meet the conditions for face mask detection via deep learning methods (Kong, et al., 2021). Another model proposed to control and detect individuals who violate the commute restrictions during the lockdown, is a Three-Layered Decentralized IoT Biometric architecture. This architecture consists of three main layers including first, physical layer, which refers to continuously take photos by distributed devices ported to raspberry pi, send to the cloud server and update the data set, second the edge layer to apply decision-making algorithm for and interpretation and face detection using "CNN Architecture", and finally the Application servers which are used for the local authorities' surveillance on the residents (Kolhar, et al., 2020). More completely, a three-tier system proposed for Coronavirus infectious detection includes the detection of evaluated parameters in the patient-specific layer using sensors in the environment and mobile application to track the patient, the Cloud Computing layer for data processing using the fog grid, and the hospital layer applying "Convolutional Neural Network(CNN)" architecture using deep learning in which cloud can detect the covid-19 infections based on scanned chest X-ray images (El-Rashidy, et al., 2020). In our proposed model, the Microsoft Azure cloud is used to give real-time monitoring of individuals' status as an analytical diagram by time.

3. Research Method

In this research, an Automated Covid -19 Gate Pass proposed which regulates and controls access of operation and identifies any signs of virus infection in individuals. Similarly, the body temperature measurement is the key factor to meet the condition of passing the internal and external gates. The proposed model aims to monitor the body temperature using MLX90614 non-contact temperature sensor module, pi-camera, and pocket-sized raspberry pi provide real-time status of Body temperature in the Microsoft Azure Cloud. Then, evaluates the minimum condition of passing the gate (less than 37.5 C), if it doesn't meet the condition, the speaker, as a buzzer, will be activated and announce a warning sentence. Moreover, the pi cam starts

capturing an image from the target and finally, send the image to system administration via Email using SMTP protocol. Further, the real-time data is represented in the Azure cloud shell. The overall operation is represented in Figure 1.

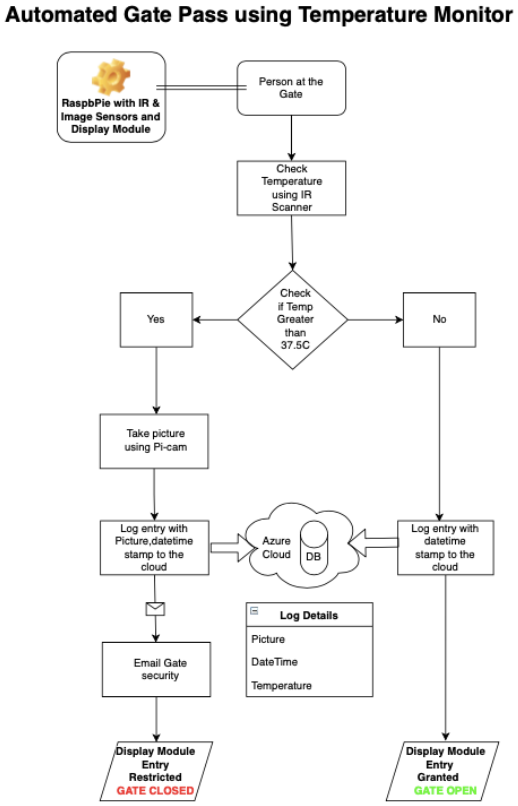


Figure1: Automated Gate Pass System

3.1 Hardware Components

3.1.1. Raspberry Pi board

Raspberry Pi is a package-size computer introduced first time in the UK for educational purposes. Raspberry pi acts as an interface to connect hardware devices, using software and the internet to make things smart known as the Internet of Things. Raspberry pi 4 is used for the implementation of this project which is remotely connected to a laptop.

3.1.2. Contactless IR Body Temperature Sensor (MLX90614)

The MLX90614 sensor is an infrared-based temperature measurement sensor that uses two units for a temperature output. the first one acts as a detector to sense the temperature. then, the second unit acts as a Digital Signal Processor to compute Data sensed by the first unit. It also follows Stefan-Boltzmann law refers to receive IR energy and intensity of things which represents the corresponding temperature of them. Then,

the MLX90614 sensor converts the value into 17-bit ADC through the application of the I2C communication protocol. Additionally, This Sensor operates in 3.6V to 5V. as well measures temperature ranging from -40°C to 125°C for Ambient and -70°C to 382.2°C for Objects with the resolution of Resolution 0.02°C**.** To use this sensor in raspberry pi, there is a need for enabling the I2C interface in the raspberry pi configuration.

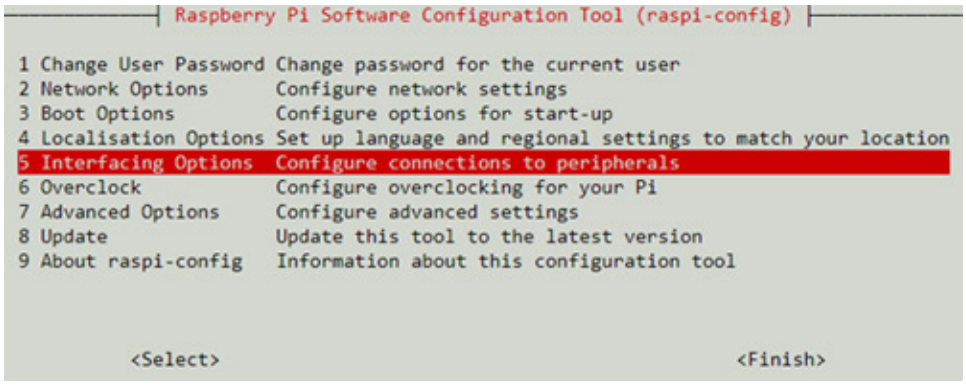
$ sudo apt-get install python-setuptools

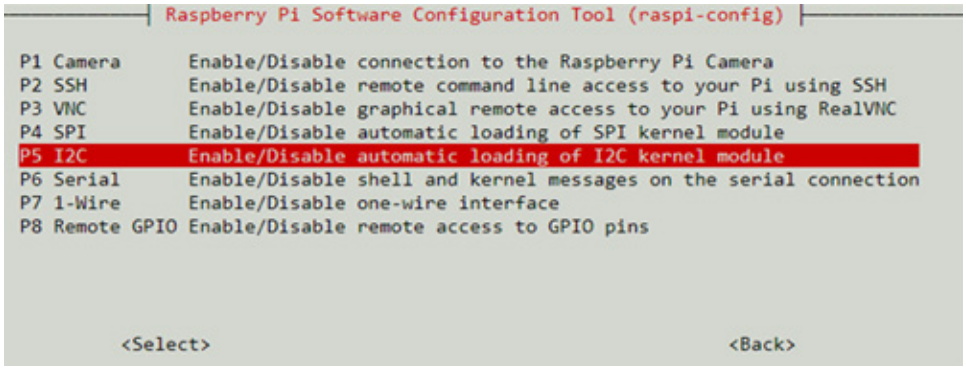
$ sudo apt-get install -y i2c-tools

3.1.3. Pi Camera

To take a picture of a person whose body temperature is detected above the minimum determined, we use the pi camera module, which is activated in the Raspberry Pi settings, and the related libraries are installed. The external pi camera module is connected to raspberry pi board through ribbon cable.

Additionally, we need generic tools like speaker with 3.5mm jack, breadboard, connecting wires and power supply. Before connecting all the hardware we need to do enable I2C using configuration tool(*$ sudo raspi-config*)





3.2. SMTP

SMTP stands for Simple Message Transfer Protocol, is a default protocol for transmitting and receiving email. This protocol, which was known as "Request for Comments (RFC)" 780 by Sluizer and Postel in 1980 (Shitole & Y.Divekar , 2019), using TCP/IP, practically works on ports 25, 2525, or 587 (Bhardwaj & Goundar, 2017), and the RFC 5321 is applied as its latest version. It is also administrated by the Internet Engineering Task Force(IETF). To use this protocol in python code to send an alert Email containing the individual image captured by the pi camera needs to decrease the security of Gmail accounts which is not adaptive with Python. The essential packages of SMTP downloaded as follows:

$ sudo apt-get install ssmtp

$ sudo apt-get install mailutils

3.3. Microsoft Azure Cloud Services

Microsoft Azure, introduced by Microsoft, with support for most devices and programming languages, provides a convenient platform for IoT-based software services including two-way connection to devices and sensors through a gateway, called Azure IoT Hub, for real-time data entry, storage and processing on the cloud environment through "Azure Machine Learning" and "Azure Stream Analytics" (Keswan , et al., 2019). Figure 2 shows the overall Microsoft Azure architecture.

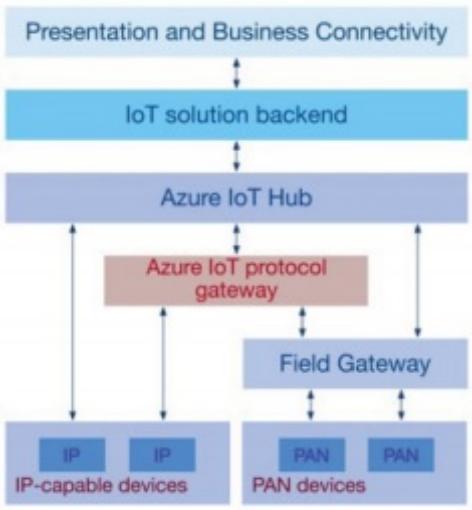


Figure 2: Microsoft Azure Architecture (Keswan , et al., 2019)

To insert data on the Microsoft Azure cloud environment, after creating a new IoT hub, need to register the device that wants to connect to the cloud to send data. To do this, click on IOT

DEVICE and enter the device information including Device ID and Authentication type, provides Data authentication by the symmetric encryption algorithm, and enable the device connection to IoT hub. The device is registered and its information including the Primary key and Connection string-primary key, which is primarily required to connect to the Microsoft Azure, is displayed.

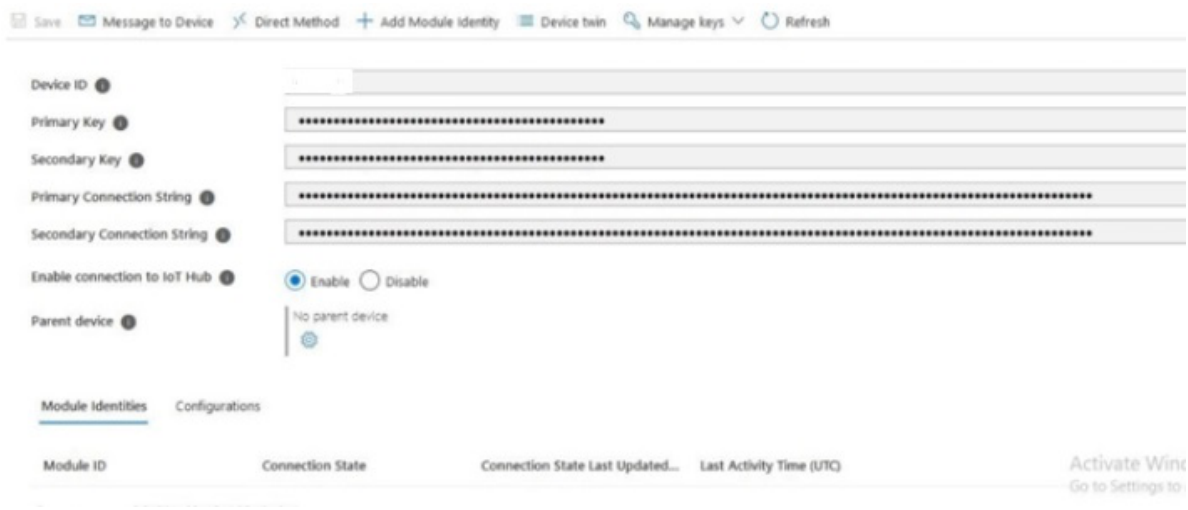


Figure3: Device details in IoT hub

In the next step, to deploy the data in the cloud environment, it is necessary to create an Azure cloud shell(CLI) in the cloud portal. then, the required packages and libraries are installed and imported on Raspberry pi terminal and Python Idle respectively.

 # Packages

$ sudo pip3 install azure-iot-device

$ sudo pip3 install azure-iot-hub

$ sudo pip3 install azure-iothub-service-client

$ sudo pip3 install azure-iot hub-device-client

 # Python Code for importing Libraries

```
from azure.iot.device import IoTHubDeviceClient, Message
import RPi.GPIO as gpio
import picamera
import time
import json
```

```
import smtplib
from email.mime.multipart import MIMEMultipart
from email.mime.text import MIMEText
from email.mime.base import MIMEBase
from email import encoders
from email.mime.image import MIMEImage

from smbus2 import SMBus
from mlx90614 import MLX90614
from pygame import mixer

from azure.iot.device import IoTHubDeviceClient, Message
from azure.storage.blob import BlobClient

#Python Code to define connection string to Azure
conn_str = "HostName=HubIOT047.azure-devices.net;DeviceId=RaspberryPi047;SharedAccessKey=SDbLO2+eIPc8sZE0D1rhyaKLFdeAtNSHlPFnslY0sbA="

my_conn_str = "DefaultEndpointsProtocol=https;AccountName=saraspbpi047cfe6da2ac73a;AccountKey=p07nOhV27E9zwW6PXuzzQxpxKAKe8BJC40ZeaeVLSsW2z+yJx4rPVD77Ma/mF8htFVuaGN4/LDcgNCML6W9KbA==;EndpointSuffix=core.windows.net"
```

3.4. System Modeling

3.4.1. Circuit Configuration:

After installing the related packages mentioned in previous steps, according to Table 1, the MLX 90614 component, consists of 4 external pins, the first pin (VCC) is connected to PIN 2 of raspberry pi (5V power). The second one(GND) is connected to PIN6 (Ground). Then, the third pin(SDA) is connected to PIN 3 which is related to I2C connection protocol and the last one(SCL) is connected to PIN5. Also, the speaker is connected to raspberry pi audio port. The result of the final hardware configuration is also represented in Figure 4.

MLX PIN	Raspberry pi #PIN
VCC	2
GND	6
SDA	3
SCL	5

Table1: Pin Connections of MLX90614

Figure4: Hardware Configuration

3.4.2. Python Programming

Python programming is carried out using the raspberry pi's default python IDE, Thonny

a. Main Method:

The main method consists of infinite loop where ambient and object temperatures are continuously monitored using contactless thermal sensor. When sensor detects object temperature more than 34C multiple functions (nested to keep code simple) are called to capture image, voice alert, email notification and finally cloud upload of the log data.

```
#change the email address accordingly
fromaddr = "abc@gmail.com"
toaddr = "xyz@gmail.com"
While 1:
    bus = SMBus(1)
    sensor = MLX90614(bus, address=0x5A)
```

```
print ("Ambient Temperature :", sensor.get_ambient())
print ("Object Temperature :", sensor.get_object_1())
temp = round(sensor.get_object_1(),2)
bus.close()
if temp > 34:
    print ("High temperature has been detected !!!")
    buzzer()
    print("Temperature in centigrade is",temp)
    capture_image(temp)
    print("Log data and image are successfully uploaded !")
    time.sleep(10)
else:
    time.sleep(0.5)
```

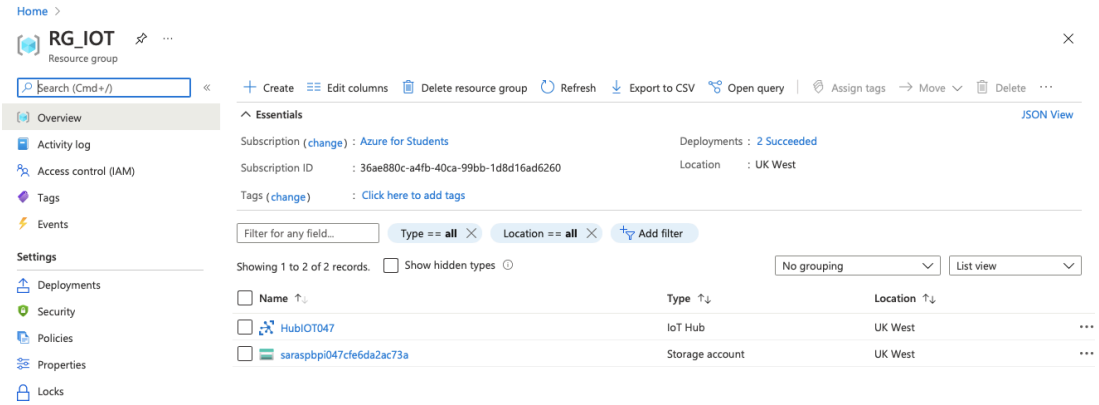
b. Functions:

```
#Function to send email from Pi directly
def sendMail(data):
#Function to upload image to Azure storage blob
def upload_image(file_name):
#Function to upload temperature log to cloud
def upload_cloud(data,temp):
#Function to alert via audio instantly
def buzzer():
#Function to capture image using Picamera
def capture_image(temp):
```

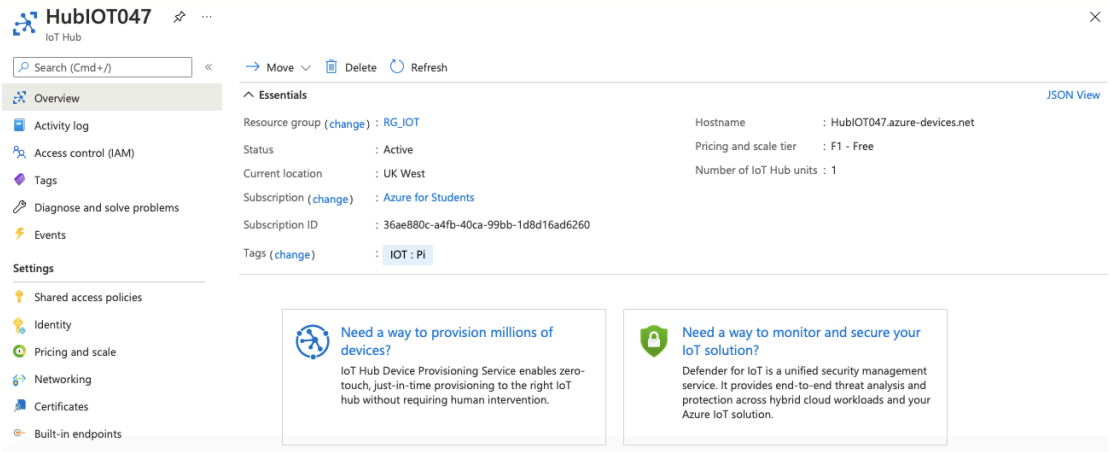
3.4.3 Cloud Deployment

Consists of different resources creation in following order

Resource group creation
IoT Hub creation
-> Adding IoT devices
-> File upload -> Adding Azure storage account -> Select Storage Container
-> Turn on Receive notification



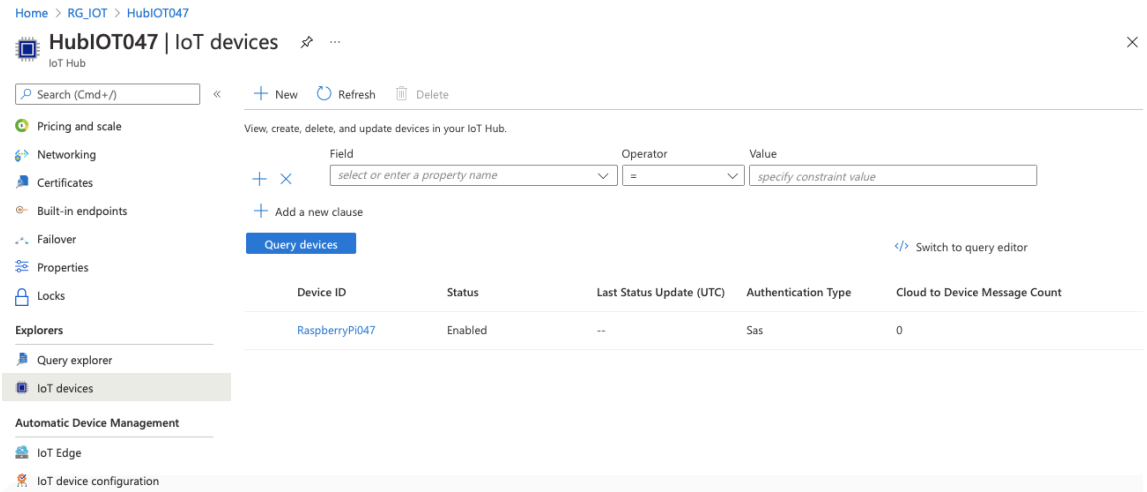

```
{
  "id": "/subscriptions/36ae880c-a4fb-40ca-99bb-1d8d16ad6260/resourceGroups/RG_IOT",
  "name": "RG_IOT",
  "location": "ukwest",
  "properties": {
    "provisioningState": "Succeeded"
  }
}
```

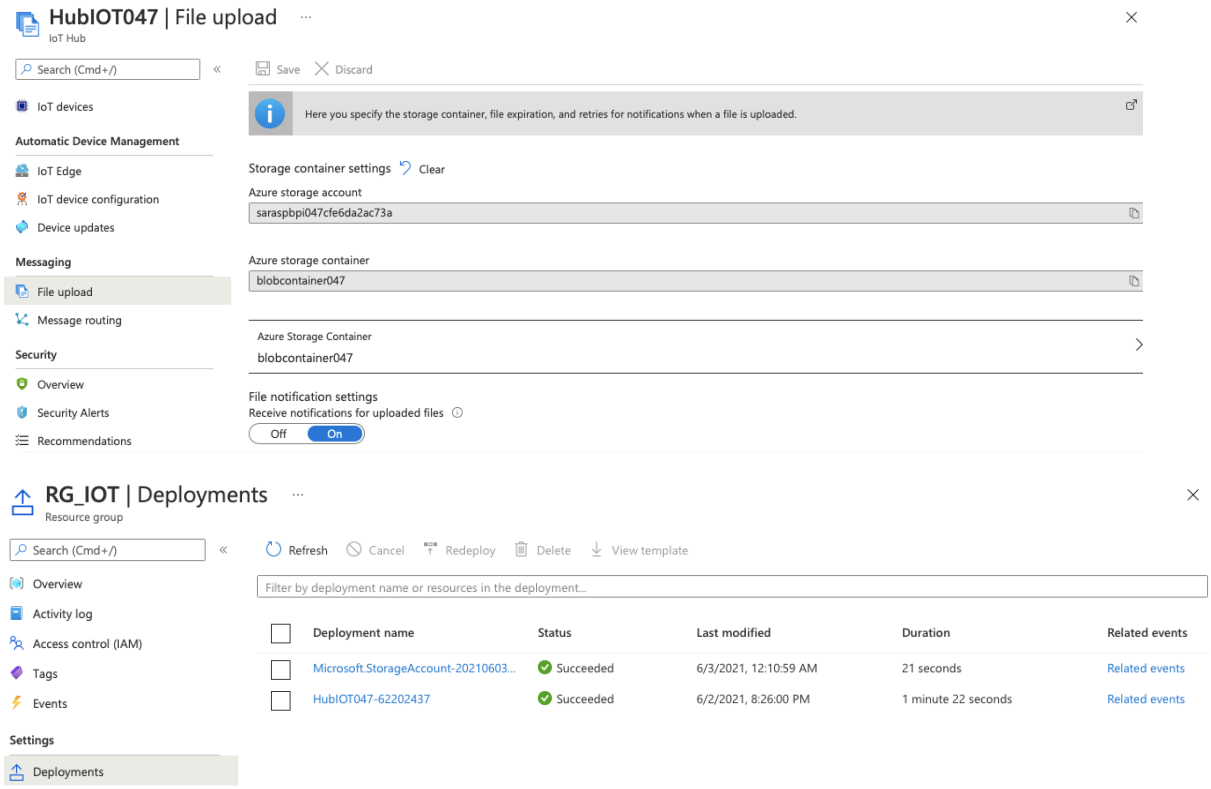




```
{
  "id": "/subscriptions/36ae880c-a4fb-40ca-99bb-
1d8d16ad6260/resourceGroups/RG_IOT/providers/Microsoft.Devices/IotHubs/HubIOT047",
  "name": "HubIOT047",
  "type": "Microsoft.Devices/IotHubs",
  "location": "ukwest",
  "tags": {
    "IOT": "Pi"
  },
  "subscriptionid": "36ae880c-a4fb-40ca-99bb-1d8d16ad6260",
  "resourcegroup": "RG_IOT",
  "etag": "AAAADE1xgDw=",
  "properties": {
    "locations": [
      {
        "location": "UK West",
```

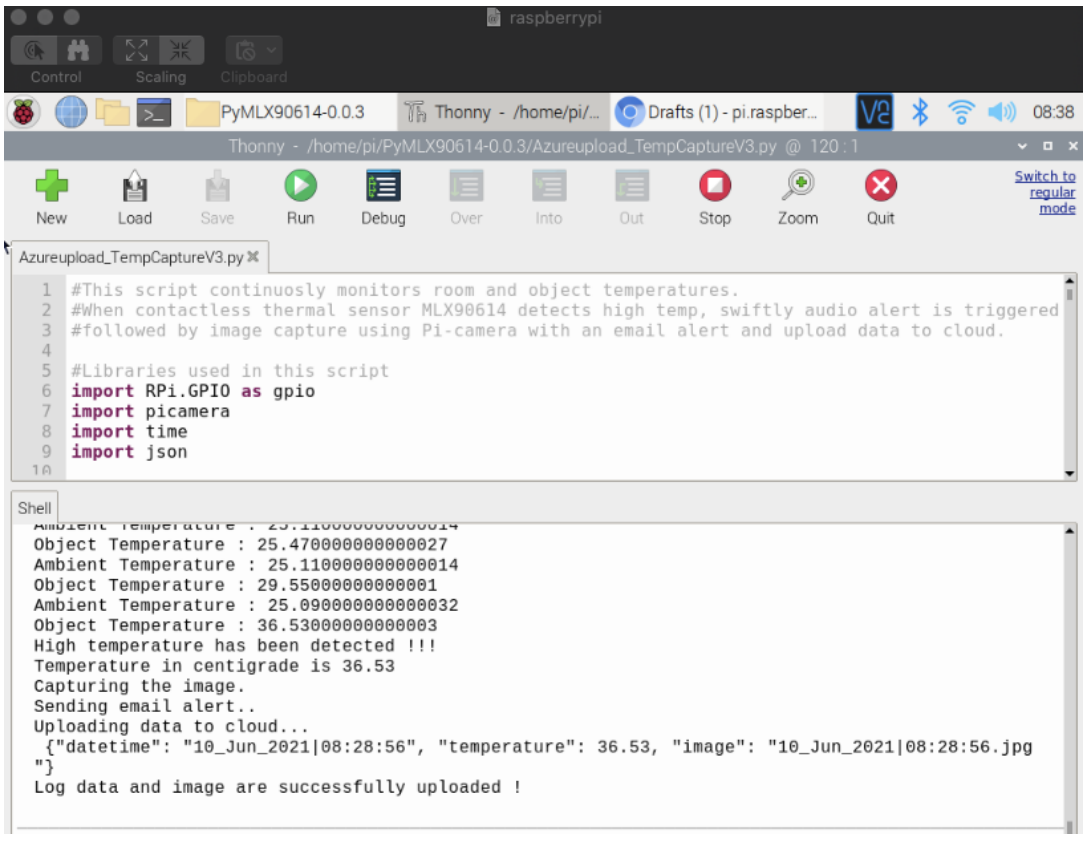
```
            "role": "primary"
          },
          {
            "location": "UK South",
            "role": "secondary"
          }
        ],
        "state": "Active",
        "provisioningState": "Succeeded",
        "ipFilterRules": [],
        "hostName": "HubIOT047.azure-devices.net",
        "eventHubEndpoints": {
          "events": {
            "retentionTimeInDays": 1,
            "partitionCount": 2,
            "partitionIds": [
              "0",
              "1"
            ],
            "path": "iothub-ehub-hubiot047-11637931-79649ea804",
            "endpoint": "sb://ihsuprodcwres008dednamespace.servicebus.windows.net/"
          }
        },
        "routing": {
          "endpoints": {
            "serviceBusQueues": [],
            "serviceBusTopics": [],
            "eventHubs": [],
            "storageContainers": []
          },
          "routes": [],
          "fallbackRoute": {
            "name": "$fallback",
            "source": "DeviceMessages",
            "condition": "true",
            "endpointNames": [
              "events"
            ],
            "isEnabled": true
          }
        },
        "storageEndpoints": {
          "$default": {
            "sasTtlAsIso8601": "PT1H",
            "connectionString": "DefaultEndpointsProtocol=https;EndpointSuffix=core.windows.net;A
ccountName=saraspbpi047cfe6da2ac73a;AccountKey=****",
            "containerName": "blobcontainer047",
            "authenticationType": "keyBased"
          }
        },
        "messagingEndpoints": {
```

```
        "fileNotifications": {
          "lockDurationAsIso8601": "PT1M",
          "ttlAsIso8601": "PT1H",
          "maxDeliveryCount": 10
        }
      },
      "enableFileUploadNotifications": true,
      "cloudToDevice": {
        "maxDeliveryCount": 10,
        "defaultTtlAsIso8601": "PT1H",
        "feedback": {
          "lockDurationAsIso8601": "PT1M",
          "ttlAsIso8601": "PT1H",
          "maxDeliveryCount": 10
        }
      },
      "features": "None"
    },
    "sku": {
      "name": "F1",
      "tier": "Free",
      "capacity": 1
    },
    "identity": {
      "type": "None"
    }
}
```



3.5. Results Analysis



Python 3.7.3 (/usr/bin/python3)
>>> %Run Azureupload_TempCaptureV3.py

#Temperature is monitored continuously

Ambient Temperature : 25.130000000000052

Object Temperature : 24.510000000000048

Ambient Temperature : 25.07000000000005

Object Temperature : 24.53000000000003

Ambient Temperature : 25.110000000000014

Object Temperature : 24.470000000000027

Ambient Temperature : 25.07000000000005

Object Temperature : 24.450000000000045

Ambient Temperature : 25.110000000000014

Object Temperature : 25.470000000000027

Ambient Temperature : 25.110000000000014

Object Temperature : 29.55000000000001

Ambient Temperature : 25.090000000000032

Object Temperature : 36.53000000000003

#When high Temperature is detected

High temperature has been detected !!!

Temperature in centigrade is 36.53

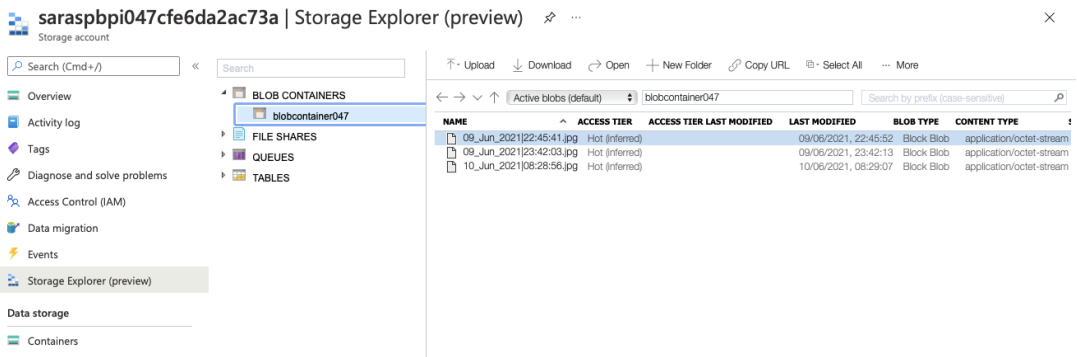
Capturing the image.

Sending email alert..

Uploading data to cloud...

{"datetime": "10_Jun_2021|08:28:56", "temperature": 36.53, "image":
"10_Jun_2021|08:28:56.jpg"}

Log data and image are successfully uploaded !



4. Potential Challenges

4.1 Scalability

Implementing IoT models is always faced with different challenges. With the increasing number of digital devices, interconnected sensors and nodes and the large amounts of data associated with them, scalability is one of the challenges we may meet which potentially, would become more apparent in Covid-19 related projects that are substantially ubiquitous in the world and require wider coverage and more energy to run precisely (KamalAbdulah , et al., 2020).

4.2 Legal and Social Challenges

The issue of large-scale data collection and storage of individual images can point to one of the legal challenges associated with it. In IoT systems, contrary to the Internet, there is no direct connection between the user and the sensors, so obtaining permission and legal authority from individuals to choose the access level to their data for monitoring and analysis is mainly impossible (Keswan , et al., 2019). The matter of violating the independence of the individual in deciding on monitoring personal information (Keswan , et al., 2019), although to maintain public health and prevent the spread of the virus and infect other people in the community is mandatory, and forcing people to be controlled and monitored to get permission to pass the gate can be highlighted as one of the serious social challenges.

4.3 Bandwidth capacity limitation

The problem of bandwidth is another controversial challenge in implementing IoT-based systems, particularly sending data to the cloud environment. As systems get larger and the number of connected devices increases, the need for bandwidth will increase appreciably. in addition, as the most connected devices typically use LTE / 4G networks to send the collected data, the full capacity of bandwidth may lead to experience significant delays in sending data to the cloud center (KamalAbdulah , et al., 2020). In the Covid-19 monitoring systems, where real-time and accurate submission of data is crucially important for monitoring and instant decision, this can be a serious challenge in the large-scale projects.

4.4 Security

Ensuring security is another important issue that is found in abundance in various IoT-based systems which means that data have not been altered, have not been hacked or attacked in the

sending path, and are not accessible to everyone when are sending to Cloud (KamalAbdulah , et al., 2020). Data leakage from cloud-based environments is another potential challenge that can cause serious security issues in large-scale cloud-based platforms when the system administrator has to be contingent on the cloud service providers in the cloud computing refers to Cloud Service Provider (CPS) (Mishra, et al., 2020). Furthermore, "Denial-of-Service (DoS)" and "Distributed Denial-of-Service(DDoS)" attacks, as well as trust and confidentiality in CPSs, are potential issues on cloud-based systems (Rath , et al., 2019). Additionally, Other possible security issues associated with these systems include the high rate of cloud server failures, unauthorized access to unencrypted information, session hijacking, and spoofing attacks (Pramanik, et al., 2019). Data leak and Data loss come from the violation of confidentiality when data is stored in the remote serve, Data integrity refers to the modification of data by an unauthorized person, and Availability could be outlined as the main challenges in providing Data security on the cloud platform. Also, other potential challenges associated with cloud computing include a high rate of Data distribution, points out locality definition of Data in large scale implementation, Data breach which can be caused by different problems including the accidental transmission or high malicious attacks to the cloud server and Data segregation implies to "multi-tenancy" feature of cloud gives permission to multiple users to store data leading to data intrusion risk possibility (Raoa & K. Selvamanib, 2015). Sending emails via the SMTP protocol is another challenge in discussing security and privacy. SMTP does not support the ability of encryption, thus can increase the probability of eavesdropping considerably. SMTP also has a text-based log-in system that sends senders' computer information along with the contents of the e-mail, leading to an increment of phishing attacks. In addition, SMTP does not support the repudiation mechanism, causing the email contents to remain on the server storage for a remarkably long time, which increases the likelihood of intentional misuse of information (Bhardwaj & Goundar, 2017).

5. Enhancements

- In the current prototype IoT Hub is used. However, for more advanced feature and automated monitoring of the cloud deployments and implementation IoT edge can be used. Apparently dockers containers are used and related components are installed in Raspberry Pi.
- Free Google smtp service is used but in real time we need to build separate virtual machine and configure smtp locally to handle the message queue service.

- Currently data is fed to cloud storage and being viewed using storage explorer however, we can also query, analyze and send event-based actions to other IoT devices. We can also implement machine learning algorithms to enhance further capabilities.

6. Conclusion

This prototype helps effective tracking and prevents the spread of virus with higher accuracy, efficient monitoring, and real-time diagnosis. Further, this a completely automated from end to end i.e., detection of high temperature to uploading data to cloud storage without any human interaction. This remote body temperature detection and monitoring system is an affordable and scalable solution developed to decrease the probability and risk of the virus spread.

7. Appendix

```
#This script continuosly monitors room and object temperatures.
#When contactless thermal sensor MLX90614 detects high temp, swiftly
audio alert is triggered
#followed by image capture using Pi-camera with an email alert and
upload data to cloud.

#Libraries used in this script
import RPi.GPIO as gpio
import picamera
import time
import json

import smtplib
from email.mime.multipart import MIMEMultipart
from email.mime.text import MIMEText
from email.mime.base import MIMEBase
from email import encoders
from email.mime.image import MIMEImage

from smbus2 import SMBus
from mlx90614 import MLX90614
from pygame import mixer

from azure.iot.device import IoTHubDeviceClient, Message
from azure.storage.blob import BlobClient

fromaddr = "pi.raspberry047@gmail.com"  #change the email address
accordingly
toaddr = "deepak1p2s@gmail.com"

mail = MIMEMultipart()
```

```
mail['From'] = fromaddr
mail['To'] = toaddr
mail['Subject'] = "Alert: Temperature value exceed threshold 34 C"
body = "Please find the attached image of the person"

data = ""

#Function to send email from Pi directly
def sendMail(data):
    mail.attach(MIMEText(body, 'plain'))
    print ("Sending email alert..")
    dat = '%s.jpg' % data
    #print (dat)
    attachment = open(dat, 'rb')
    image = MIMEImage(attachment.read())
    attachment.close()
    mail.attach(image)
    server = smtplib.SMTP('smtp.gmail.com', 587)
    server.starttls()
    server.login(fromaddr, "Gatepass@047")
    text = mail.as_string()
    server.sendmail(fromaddr, toaddr, text)
    server.quit()

#Function to upload image to Azure storage blob
def upload_image(file_name):
    #print("Uploading image now...")
    my_conn_str =
"DefaultEndpointsProtocol=https;AccountName=saraspbpi047cfe6da2ac73a;Ac
countKey=p07nOhV27E9zwW6PXuzzQxpxKAKe8BJC40ZeaeVLSsW2z+yJx4rPVC77Ma/mF8
htFVuaGN4/LDcgNCML6W9KbA==;EndpointSuffix=core.windows.net"
    blob = BlobClient.from_connection_string(my_conn_str,
container_name = "blobcontainer047", blob_name = file_name)
    with open(file_name, "rb") as data:
        blob.upload_blob(data)


#Function to upload temperature log to cloud
def upload_cloud(data,temp):
    conn_str = "HostName=HubIOT047.azure-
devices.net;DeviceId=RaspberryPi047;SharedAccessKey=SDbLO2+eIPc8sZE0C1r
hyaKLFdeAtNSHlPFnslY0sbA="
    device_client =
IoTHubDeviceClient.create_from_connection_string(conn_str)
    device_client.connect()
    dat = '%s.jpg' % data
    logdata = {}
    logdata['datetime'] = data
    logdata['temperature'] = temp
    logdata['image'] = dat
    json_body = json.dumps(logdata)
    print('Uploading data to cloud...\n', json_body)
    device_client.send_message(json_body)
    time.sleep(1)
    device_client.disconnect()
    upload_image(dat)

#Function to alert via audio instantly
```

```
def buzzer():
    mixer.init()
    sound = mixer.Sound('Door_entry_notification-Loge_the_60th-
95203129.wav')
    sound.play()

#Function to capture image using Picamera
def capture_image(temp):
    data = time.strftime("%d_%b_%Y|%H:%M:%S")
    camera.start_preview()
    time.sleep(5)
    print ("Capturing the image.")
    camera.capture('%s.jpg' % data)
    camera.stop_preview()
    time.sleep(1)
    sendMail(data)
    upload_cloud(data,temp)

#Camere settings
camera = picamera.PiCamera()
camera.rotation = 180
camera.awb_mode = 'auto'
camera.brightness = 55

while 1:
    bus = SMBus(1)
    sensor = MLX90614(bus, address=0x5A)
    print ("Ambient Temperature :", sensor.get_ambient())
    print ("Object Temperature :", sensor.get_object_1())
    temp = round(sensor.get_object_1(),2)
    bus.close()
    if temp > 34:
        print ("High temperature has been detected !!!")
        buzzer()
        print("Temperature in centigrade is",temp)
        capture_image(temp)
        print("Log data and image are successfully uploaded !")
        time.sleep(10)
    else:
        time.sleep(0.5)
```

References

Raoa, R. V. & K. Selvamanib, 2015. Data Security Challenges and Its Solutions in Cloud Computing. *Procedia Computer Science,* pp. 204-209.

Sollu, T. S., Alamsyah, Bachtiar, M. & Bontong, B., 2018. *Monitoring System Heartbeat and Body Temperature.* s.l., EDP Sciences.

Valsalan, P., Baomar, T. A. B. & Baabood, A. H. O., 2020. IOT BASED HEALTH MONITORING SYSTEM. *Journal of Critical Reviews,* 7(4), pp. 739-743.

Abdulrazaq, et al., 2020. NOVEL COVID-19 DETECTION AND DIAGNOSIS SYSTEM USING IOT BASED SMART HELMET. *International Journal of Psychosocial Rehabilitation 24. 2296-2303.*.

Babu, B., K. Srikanth, T. Ramanjaneyulu & Narayana, I. L., 2016. IoT for Healthcare. *International Journal of Science and Research (IJSR).*

Bhardwaj, A. & Goundar, S., 2017. Security challenges for cloud-based email infrastructure. *Network Security.*

El-Rashidy, N. et al., 2020. End-To-End Deep Learning Framework for Coronavirus (COVID-19) Detection and Monitoring. *Electronics,* 9(9).

Ghanavati, S., Abawajy, J. H., Izadi , D. & Alelaiwi , A., 2017. Cloud-assisted IoT-based health status monitoring framework. *Cluster Computer,* p. 1843–1853.

Hussain , S. et al., 2021. IoT and Deep Learning Based Approach for Rapid Screening and Face Mask Detection for Infection Spread Control of COVID-19. *Applied Science.*

Javaid, M. & HaleemKhan, I., 2021. Internet of Things (IoT) enabled healthcare helps to take the challenges of COVID-19 Pandemic. *Journal of Oral Biology and Craniofacial Research,* 11(2), pp. 209-214.

KamalAbdulah , M., Abdulah, A. & Alanazi, E., 2020. *IoT meets COVID-19: Status, Challenges, and Opportunities,* s.l.: eprint arXiv:2007.12268.

Kaur, A. & Jasuja, A., 2017. *Health monitoring based on IoT using Raspberry PI.* Greater Noida, India, IEEE.

Keswan , B., Ch. Mishra , T., Mo, A. G. & Keswani, P., 2019. IOT SECURITY AND PRIVACY PRESERVATION. In: D. L. e. al, ed. *Security Designs for the Cloud, IoT, and Social Networking.* s.l.:Scrivener Publishing, pp. 98-111.

Kolhar, M., Al-Turjman, F., Alameen, A. & Abualhaj, M., 2020. A Three Layered Decentralized IoT Biometric Architecture for City Lockdown During COVID-19 Outbreak. *IEEE Access,* Volume 8.

Kong, X. et al., 2021. Real-time Mask Identification for COVID-19: An Edge Computing-based Deep Learning Framework. *IEEE INTERNET OF THINGS JOURNAL.*

Kumar, K., Kumar, N. & Shah, R., 2020. Role of IoT to avoid spreading of COVID-19. *International Journal of Intelligent Networks,* Volume 1, pp. 32-35.

Mishra, S., Tripathy, N., Kishore Mishra, B. & Mahanty, C., 2020. Analysis of Security Issues in Cloud Environment. In: Dac-Nhuong Le, C. Bhatt & M. Madhukar, eds. s.l.:Scrivener Publishing LLC.

Ndiaye, M. et al., 2020. IoT in the Wake of COVID-19: A Survey on Contributions, Challenges and Evolution. Volume 8.

Pramanik, P. . K. D., Pareek, G. & Nayyar, A., 2019. Security and Privacy in Remote Healthcare: Issues, Solutions and Standards. In: *elemedicine Technologies: Big data, Deep Learning, Robotics, Mobile and Remote Applications for Global Healthcare.* s.l.:Elsevier.

Rath , A., Spasic, B., Boucart, N. & Thiran, P., 2019. Security Pattern for Cloud SaaS: From System and Data Security to Privacy Case Study in AWS and Azure. *Computers,* 8(2).

Rudraraju, S. R., Suryadevara, N. K. & Negi, A., 2020. *Face Mask Detection at the Fog Computing Gateway.* Sofia,Bulgaria, IEEE.

Shitole, H. & Y.Divekar , 2019. Secure Email Software using e-SMTP. *International Research Journal of Engineering and Technology (IRJET),* 6(3), pp. 3967-3971.