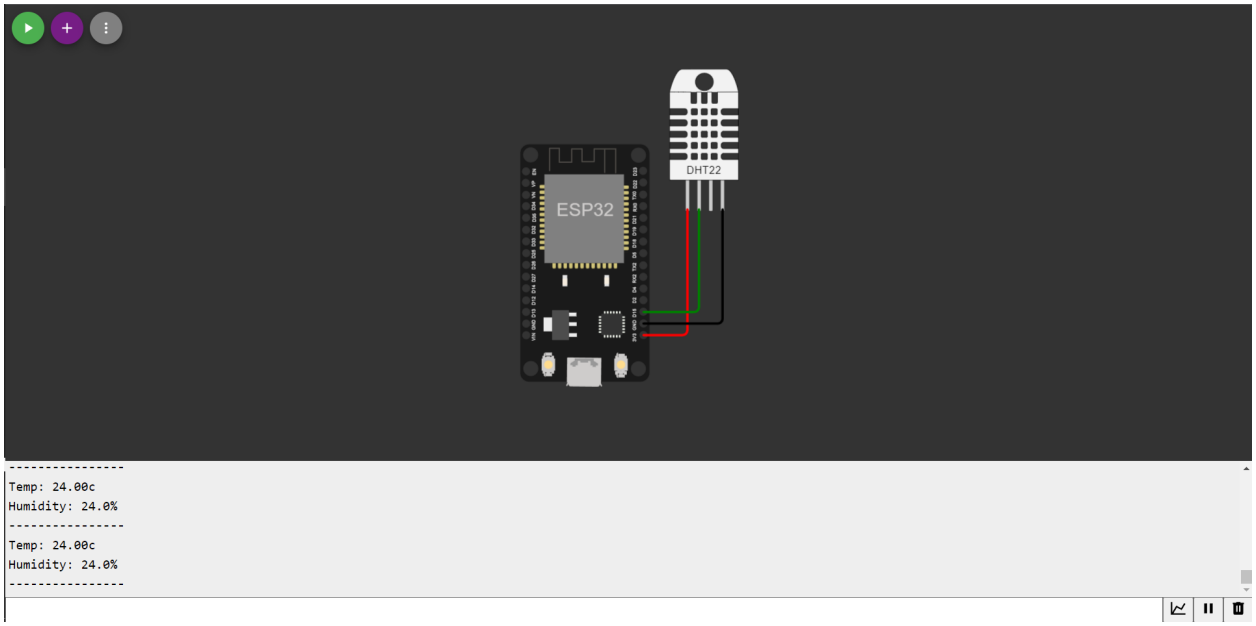
TEAM ID	PNT2022TMID04587
PROJECT NAME	Industry - specific intelligent fire management system

WOWKI SIMULATION

Using the wowki online simulator we have simulated the hardware part of the project. The circuit consists of NodeMCU and sensors to monitor parameters like temperature, humidity, gas and to detect flame and take necessary actions like turning on sprinkler and exhaust fan is done accordingly.

CODE:

```
#include "DHTesp.h"
const int DHT_PIN = 15;
DHTesp dhtSensor;

void setup() {
  // put your setup code here, to run once:
  Serial.begin(115200);
  Serial.println("Hello, ESP32!");
  dhtSensor.setup(DHT_PIN,DHTesp::DHT22);
}

void loop() {
  TempAndHumidity data = dhtSensor.getTempAndHumidity();
  Serial.println("Temp: "+ String(data.temperature,2)+"c");
  Serial.println("Humidity: "+ String(data.temperature,1)+"%");
  Serial.println("----------------");
  delay(100); // this speeds up the simulation
}
```



PYTHON

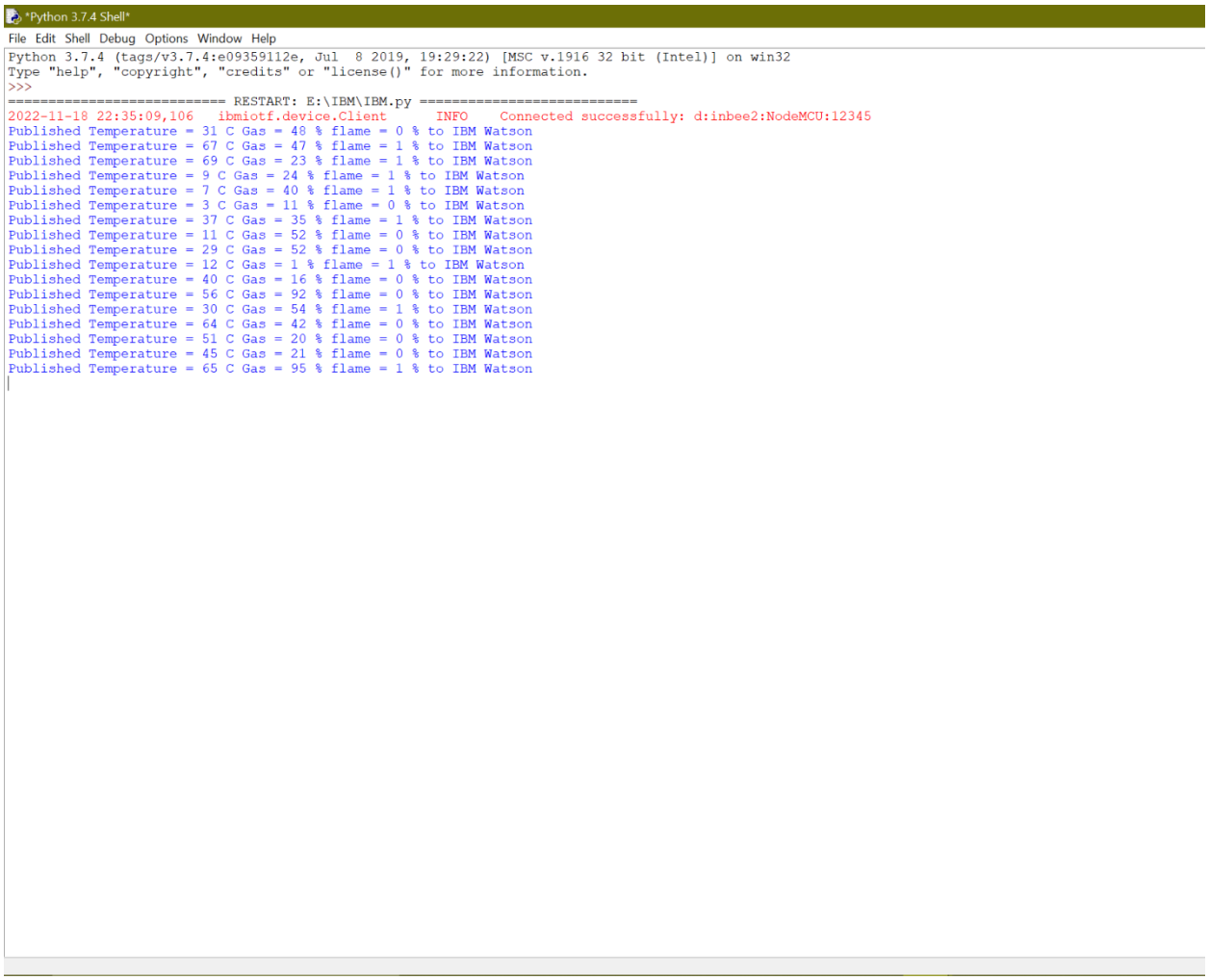
The python code is used to randomly generate temperature, humidity, gas and flame values and it can be pushed to the IBM cloud.

CODE:

```
import time
import sys
import ibmiotf.device
import ibmiotf.application
import random

organization = "fvg3cq"
deviceType = "NodeMCU"
deviceId = "12345"
authMethod = "token"
authToken = "12345678"

def myCommandCallback (cmd):
    print("Command received: %s" % cmd.data['command'])
    status = cmd.data['command']
    if status == "sprinkleron":
        print("sprinkler is on")
```

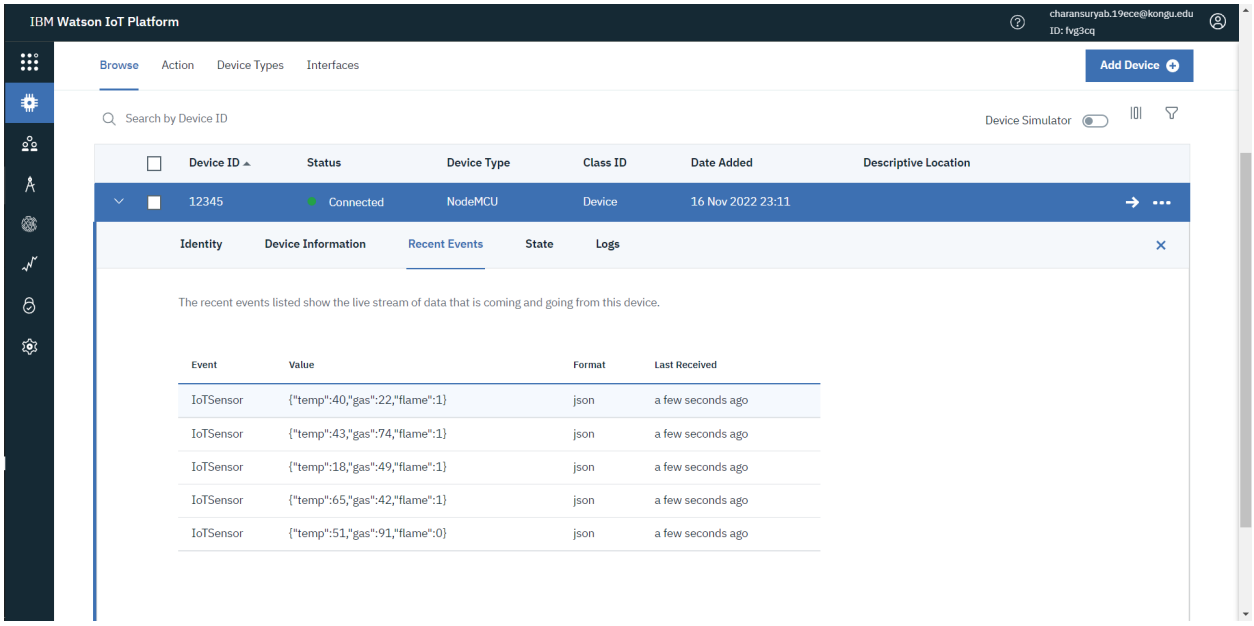
```
    elif status == "sprinkleroff":
        print("sprinkler is off")
    elif status==("fanon"):
        print("fan is on")
    elif status==("fanoff"):
        print("fan is off")
    else:
        print("Please send proper command")

    print(cmd)

'''def myCommandCallback2(cmd):
     print("Command received: %s" % cmd.data['command'])
     status = cmd.data['command']
     if status == "fanon":
         print("fan is on")
     else:
         print("fan is off")
     print(cmd)'''

try:
    deviceOptions = {"org": organization, "type": deviceType, "id":
deviceId, "auth-method": authMethod, "auth-token": authToken}
    deviceCli = ibmiotf.device.Client(deviceOptions)
    #.....................................................

except Exception as e:
    print("Caught exception connecting device: %s" % str(e))
    sys.exit()
# Connect and send a datapoint "hello" with value "world" into the cloud as an event of
type "greeting" 10 times
deviceCli.connect()

while True:
    #Get Sensor Data from DHT11
    temp=random.randint(0,70)
    gas=random.randint(0,100)
    flame=random.randint(0,1)
    data = { 'temp' : temp, 'gas': gas, 'flame': flame }
    #print data
```

```
    def myOnPublishCallback():
        print ("Published Temperature = %s C" % temp, "Gas = %s %%" %
gas, "flame = %s %%" % flame,"to IBM Watson")
    success = deviceCli.publishEvent("IoTSensor", "json", data, qos=0,
on_publish=myOnPublishCallback)
    if not success:
        print("Not connected to IoTF")
    time.sleep(1)
    deviceCli.commandCallback = myCommandCallback
    #deviceCli.commandCallback2 = myCommandCallback2
# Disconnect the device and application from the cloud
deviceCli.disconnect()
```



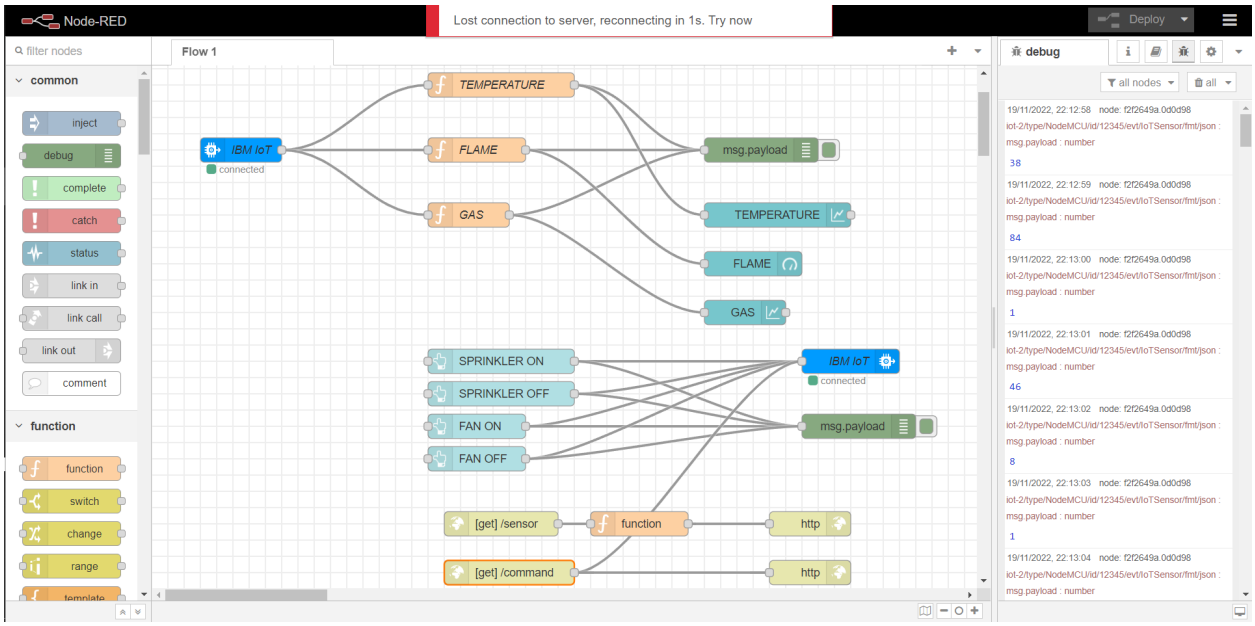
IBM CLOUD

Data from the wowki simulator or python random generator code is updated to the IBM cloud.



NODE-RED SERVICE

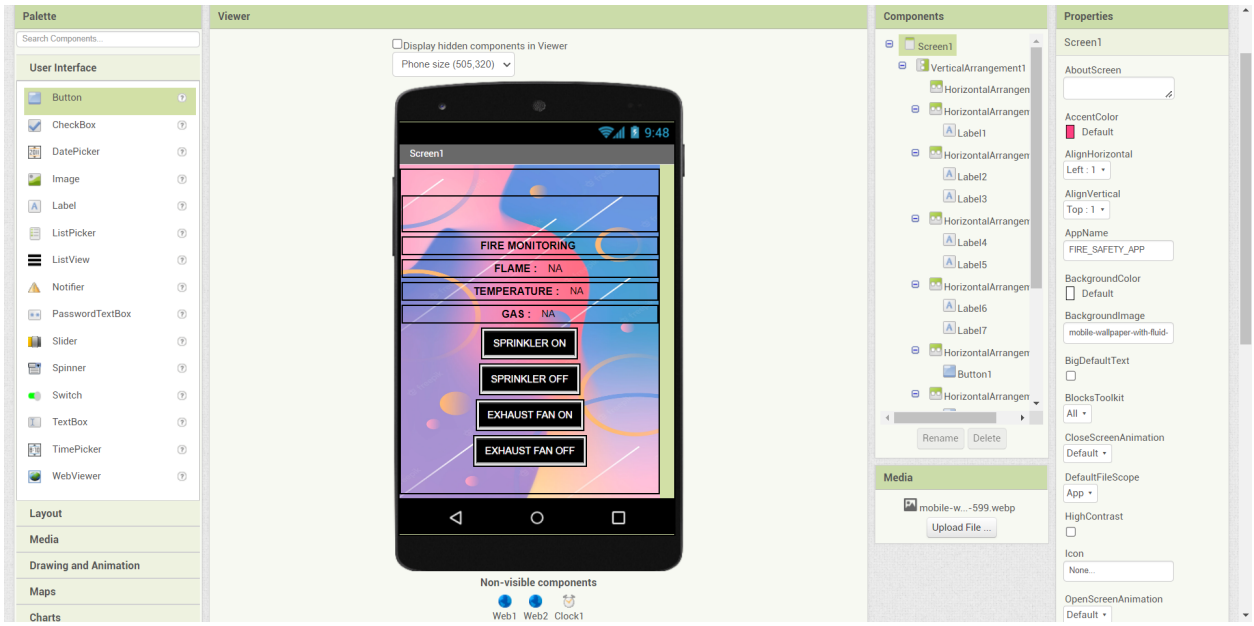
Node-Red service is used to create a dashboard to have a visual image of the various parameters.

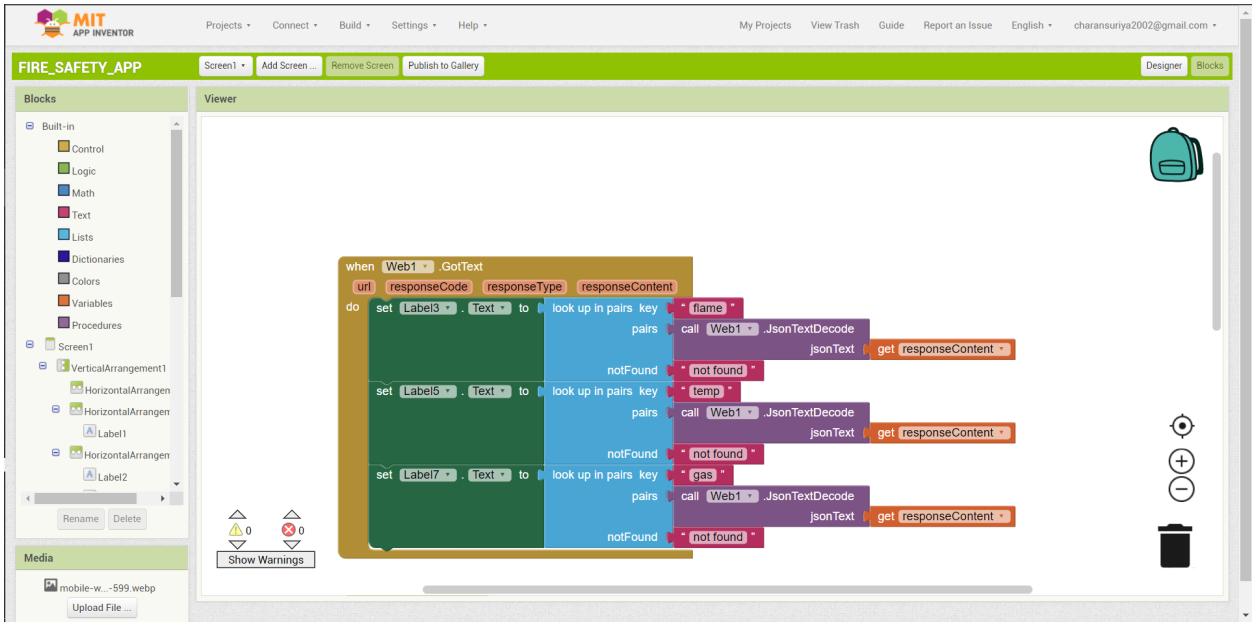


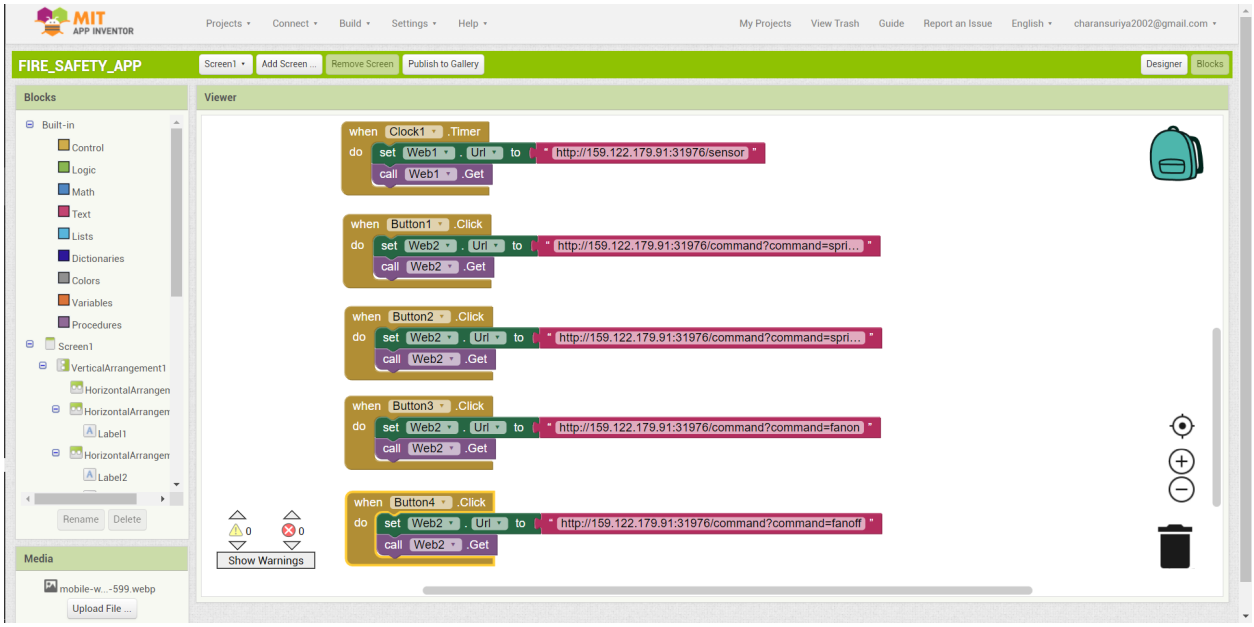
NODE-RED DASHBOARD

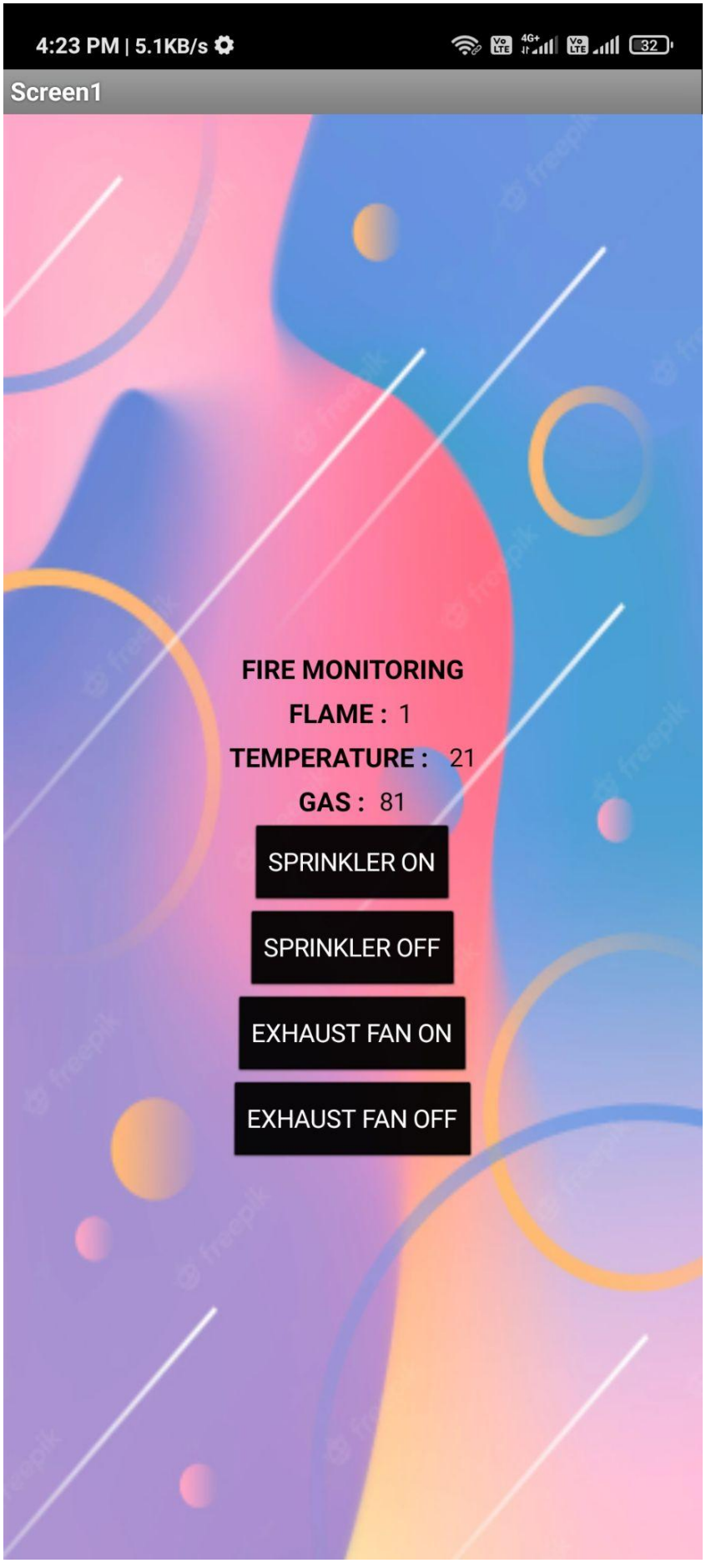
MIT APP INVENTOR

The MIT app inventor is used to create an mobile application with which we can monitor the parameters like temperature,flame,gas and take precautionary action using the controls provided in it.




MIT APP INVENTOR
Projects
Connect
Build
Settings
Help
My Projects
View Trash
Guide
Report an Issue
English
charansuriya2002@gmail.com
FIRE_SAFETY_APP
Screen1
Add Screen ...
Remove Screen
Publish to Gallery
Designer
Blocks
Viewer
Built-in
Control
Logic
Math
Text
Lists
Dictionaries
Colors
Variables
Procedures
Screen1
VerticalArrangement1
HorizontalArrangen
HorizontalArrangen
Label1
HorizontalArrangen
Label2
Rename
Delete
Media
mobile-w...-599.webp
Upload File ...
Show Warnings
when Web1 .GotText
url responseCode responseType responseContent
do set Label3 . Text to look up in pairs key " flame "
pairs call Web1 .JsonTextDecode
jsonText get responseContent
notFound " not found "
set Label5 . Text to look up in pairs key " temp "
pairs call Web1 .JsonTextDecode
jsonText get responseContent
notFound " not found "
set Label7 . Text to look up in pairs key " gas "
pairs call Web1 .JsonTextDecode
jsonText get responseContent
notFound " not found "


MIT APP INVENTOR
Projects
Connect
Build
Settings
Help
My Projects
View Trash
Guide
Report an Issue
English
charansuriya2002@gmail.com
FIRE_SAFETY_APP
Screen1
Add Screen ...
Remove Screen
Publish to Gallery
Designer
Blocks
Viewer
Built-in
Control
Logic
Math
Text
Lists
Dictionaries
Colors
Variables
Procedures
Screen1
VerticalArrangement1
HorizontalArrangen
HorizontalArrangen
Label1
HorizontalArrangen
Label2
Rename
Delete
Media
mobile-w...-599.webp
Upload File ...
Show Warnings
when Clock1 .Timer
do set Web1 . Url to " http://159.122.179.91:31976/sensor "
call Web1 .Get
when Button1 .Click
do set Web2 . Url to " http://159.122.179.91:31976/command?command=spri... "
call Web2 .Get
when Button2 .Click
do set Web2 . Url to " http://159.122.179.91:31976/command?command=spri... "
call Web2 .Get
when Button3 .Click
do set Web2 . Url to " http://159.122.179.91:31976/command?command=fanon "
call Web2 .Get
when Button4 .Click
do set Web2 . Url to " http://159.122.179.91:31976/command?command=fanoff "
call Web2 .Get


4:23 PM | 5.1KB/s
Screen1
FIRE MONITORING
FLAME : 1
TEMPERATURE : 21
GAS : 81
SPRINKLER ON
SPRINKLER OFF
EXHAUST FAN ON
EXHAUST FAN OFF