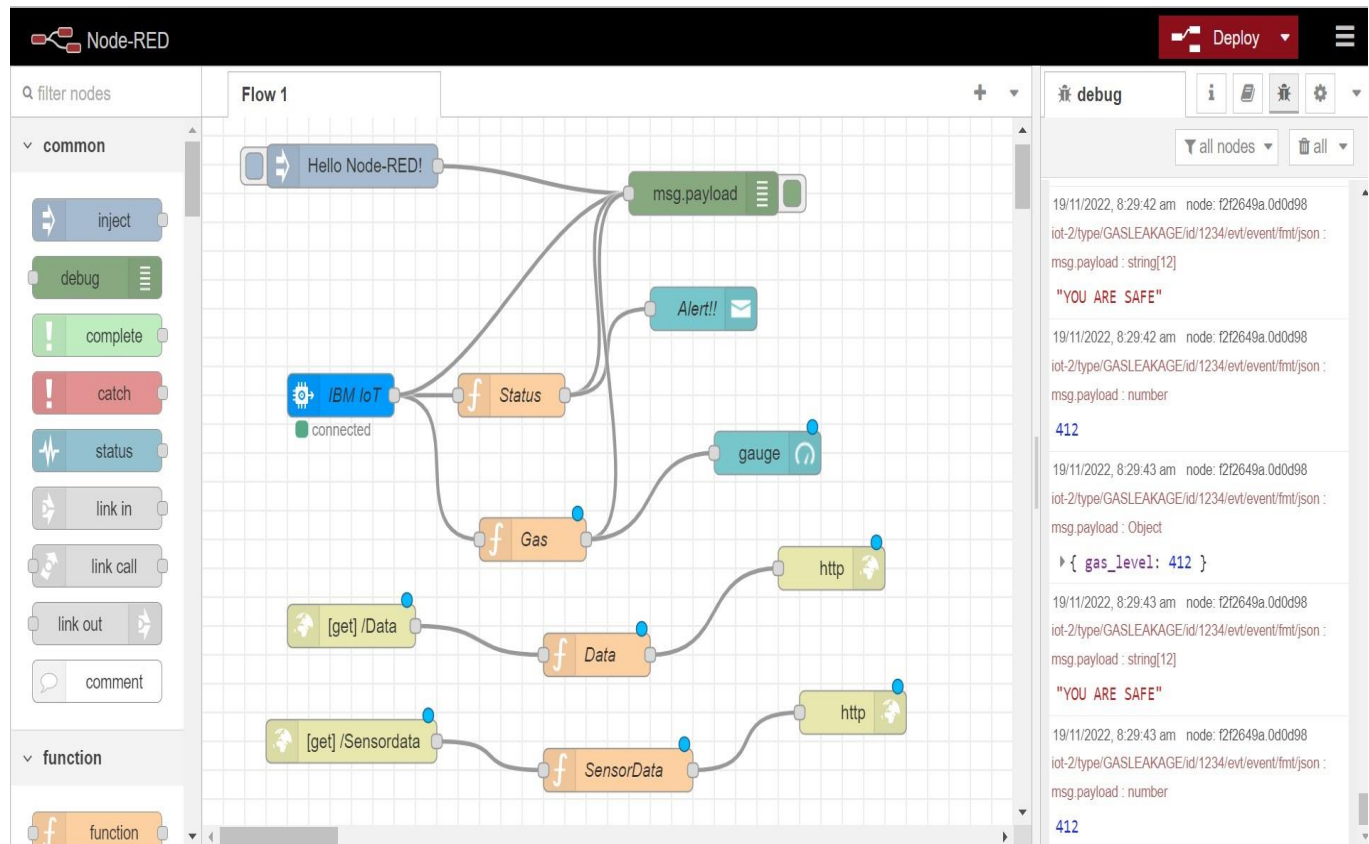


SPRINT-IV WEB UI

Team ID : PNT2022TMID00378
Project Name : Gas Leakage Monitoring and Alerting System

Creating a Web UI:

- Creating a web UI to make user to interact with the software.



Source code :

```
# Importing Required modules  
import time  
import sys  
import wiotp.sdk.device # IBM IoT Watson Platform Module  
import ibmiotf.device  
import tkinter as tk # Python GUI Package  
from tkinter import ttk # Python GUI
```

```

import time
from threading import Thread

organization = "vens1r" # Organization ID
deviceType = "GASLEAKAGE" # Device type
deviceId = "1234" # Device ID
authMethod = "token" # Authentication Method
authToken = "12345678" #Replace the authtoken

# Tkinter root window
root = tk.Tk()
root.geometry('350x300') # Set size of root window
root.resizable(False, False) # root window non-resizable
root.title('Gas Leakage Monitoring And Alerting System for
Industries (PNT2022TMID18536)')

# Layout Configurations
root.columnconfigure(0, weight=1)
root.columnconfigure(1, weight=3)

current_gas = tk.DoubleVar()

def get_current_gas(): # function returns current gas level value
    return '{:
    .2f}'.format(current_gas.get())

def slider_changed(event): # Event Handler for changes in sliders
    print('
    ----- ')
    print('Gas Level: {: .2f}'.format(current_gas.get()))
    print('
    ----- ')
    gas_label.configure(text=str(get_current_gas()) + " ppm") #Displays
    current gas level as label content

```

```
# Tkinter Labels
```

```
# label for the gas level slider
```

```
slider_gas_label = ttk.Label(root,text='Set Gas Level:')
```

```
slider_gas_label.grid(column=0,row=0,sticky='w')
```

```
# Gas Level slider
```

```
slider_gas = ttk.Scale(root,from_=0,to=3000,orient='horizontal',  
command=slider_changed,variable=current_gas)
```

```
slider_gas.grid(column=1,row=0,sticky='we')
```

```
# current gas level label
```

```
current_gas_label = ttk.Label(root,text='Current Gas Level:')
```

```
current_gas_label.grid(row=1,columnspan=2,sticky='n',ipadx=10,ipady=10)
```

```
# Gas level label (value gets displayed here)
```

```
gas_label = ttk.Label(root,text=str(get_current_gas()) + " ppm")
```

```
gas_label.grid(row=2,columnspan=2,sticky='n')
```

```
def publisher_thread():
```

```
    thread = Thread(target=publish_data)
```

```
    thread.start()
```

```
def publish_data():
```

```
    # Exception Handlingtry:
```

```
        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()

deviceCli.connect() # Connect to IBM Watson IoT Platformwhile

True:
    gas_level = int(current_gas.get())

    data = {'gas_level' : gas_level}def

    myOnPublishCallback():
        print("Published Gas Level = %s ppm" % gas_level, "to IBM
Watson")

        success = deviceCli.publishEvent("event", "json", data, qos=0,
on_publish=myOnPublishCallback)
        if not success:
            print("Not connected to IoT")
            time.sleep(1)

publisher_thread()

root.mainloop() # startup Tkinter GUI

# Disconnect the device and application from the clouddeviceCli.disconnect()

```

CODE:

```
Final.py - C:/Python/Python310/Final.py (3.10.7)
File Edit Format Run Options Window Help

# current gas level label
current_gas_label = ttk.Label(root, text='Current Gas Level:')
current_gas_label.grid(row=1, columnspan=2, sticky='n', ipadx=10, ipady=10)

# Gas level label (value gets displayed here)
gas_label = ttk.Label(root, text=str(get_current_gas()) + " ppm")
gas_label.grid(row=2, columnspan=2, sticky='n')

def publisher_thread():
    thread = Thread(target=publish_data)
    thread.start()

def publish_data():
    # Exception Handling 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()
    deviceCli.connect() # Connect to IBM Watson IoT Platform while True:
    gas_level = int(current_gas.get())
    data = {'gas_level': gas_level}
    def myOnPublishCallback():
        print("Published Gas Level = %s ppm" % gas_level, "to IBM Watson")

    success = deviceCli.publishEvent("event", "json", data, qos=0, on_publish=myOnPublishCallback)
    if not success:
        print("Not connected to IoT")
        time.sleep(1)

    publisher_thread()

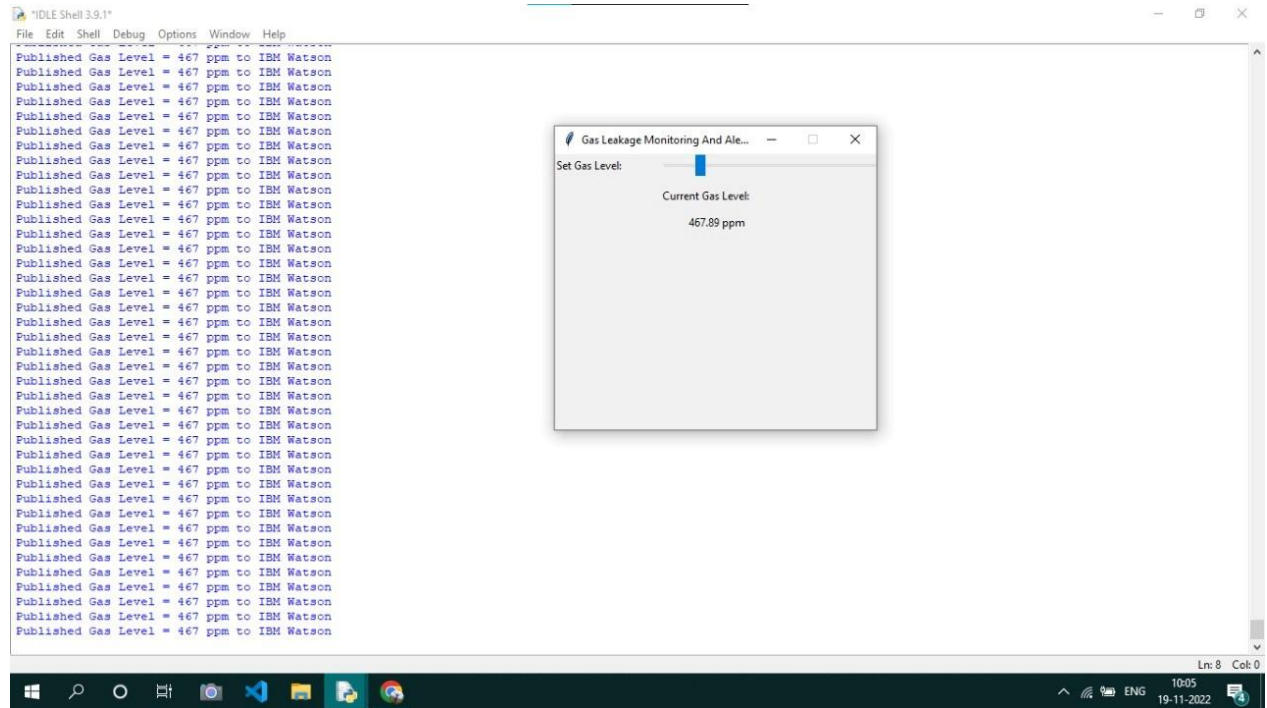
root.mainloop() # startup Tkinter GUI

# Disconnect the device and application from the cloud deviceCli.disconnect()

|
```

Activate Windows
Go to Settings to activate Windows.

OUTPUT:



The screenshot shows a software development environment with an IDE Shell window and a modal dialog box.

The IDE Shell window, titled "IDE Shell 3.9.1", contains a menu bar (File, Edit, Shell, Debug, Options, Window, Help) and a text area displaying a repeating message: "Published Gas Level = 467 ppm to IBM Watson". The text is wrapped across multiple lines.

Overlaid on the IDE Shell is a dialog box titled "Gas Leakage Monitoring And Alert...". The dialog box has a "Set Gas Level:" label next to a blue progress bar. Below the progress bar, it displays "Current Gas Level: 467.89 ppm".

The Windows taskbar at the bottom shows the system clock as 10:05 on 19-11-2022, along with network, volume, and language (ENG) icons.

Testing Web UI:

IBM Watson IoT Platform

vickybmsv@gmail.com
ID: vens1r

Browse

Action

Device Types

Interfaces

Add Device +

The recent events listed show the live stream of data that is coming and going from this device.

Event	Value	Format	Last Received
event	{"gas_level":467}	json	a few seconds ago
event	{"gas_level":467}	json	a few seconds ago
event	{"gas_level":467}	json	a few seconds ago
event	{"gas_level":467}	json	a few seconds ago
event	{"gas_level":467}	json	a few seconds ago

Items per page 50 | 1-1 of 1 item

1 Simulation running

Gas

MESSAGE
YOU ARE SAFE

Sensordata

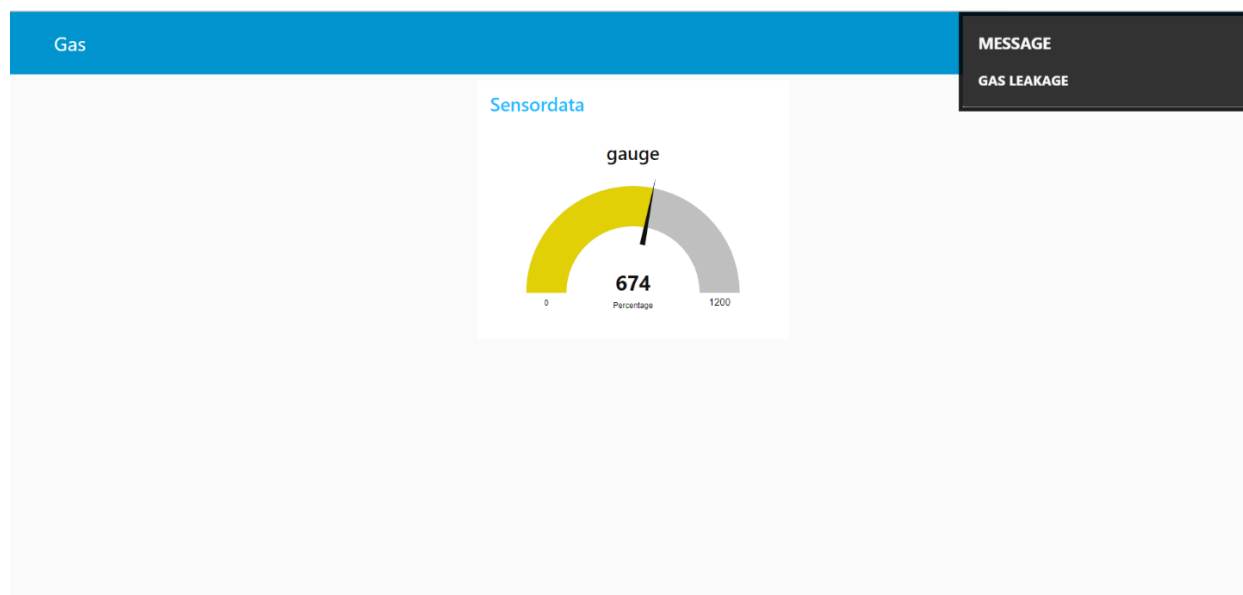
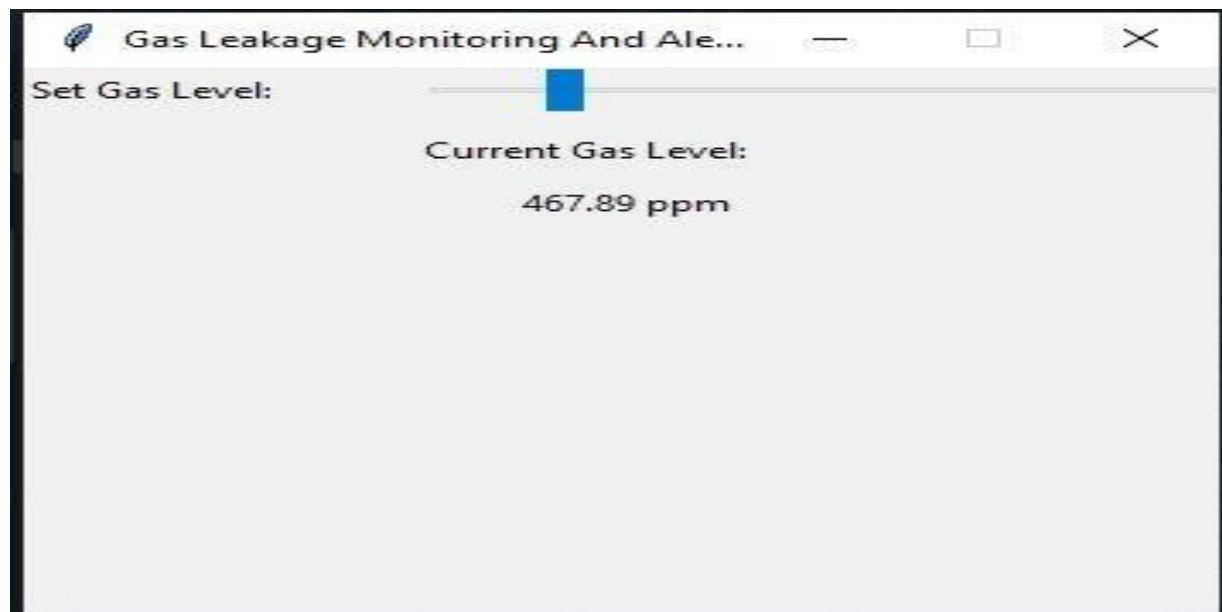
gauge

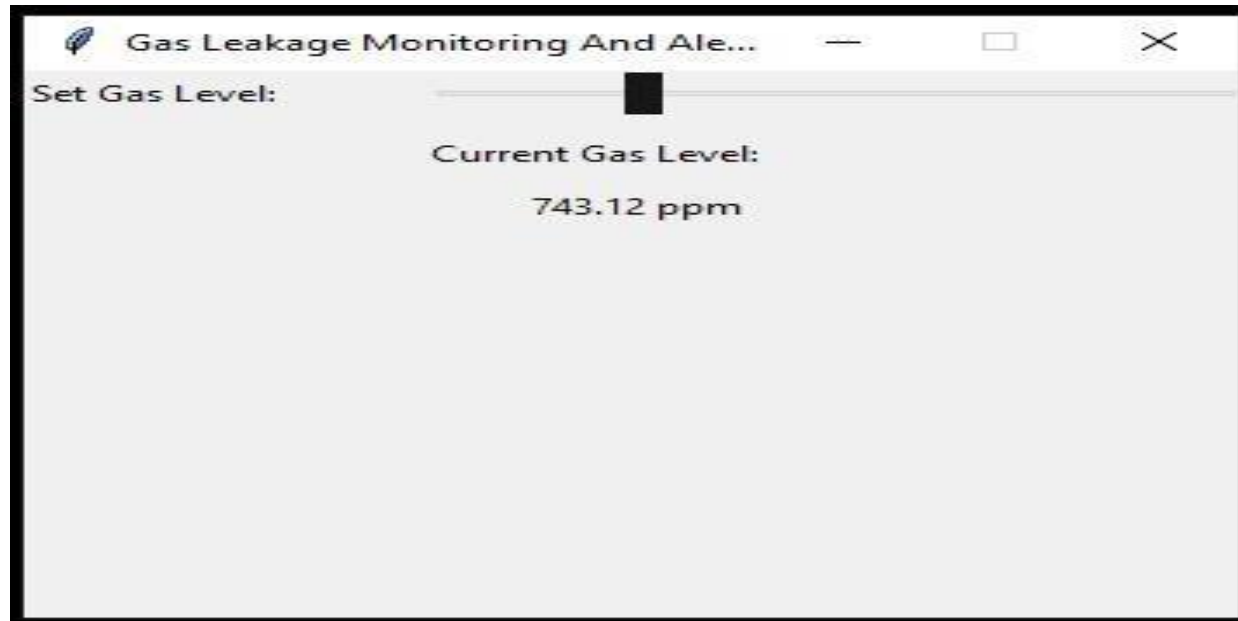
0

412

1200

Percentage





RESULT:

The Web UI is created successfully to monitor the GAS LEAKAGE.