EEL 7170 : Introduction to IoT

Lab Report

Name:	**Anushkaa Ambuj**
Roll Number:	**B21ES006**
Program:	**B.Tech in ES**

Lab 6: InfluxDB and Grafana Integration

21 October, 2024

1 Assignment

1.1 Objective

The objective of this lab is to set up data pipelines from sensors to store real-time sensor data in an InfluxDB database and visualize it using Grafana dashboards for effective monitoring and visualization.

1.2 Components Used

- Raspberry Pi
- InfluxDB
- Grafana
- Python (for data insertion)
- LIDAR Sensor (VL53L0X)

1.3 Procedure

Part 1: Installing InfluxDB on Raspberry Pi

- **Step 1: Update and upgrade all packages**

```
sudo apt update
sudo apt upgrade
```

- **Step 2: Add the InfluxDB repository**

```
curl https://repos.influxdata.com/influxdata-archive.key | gpg --dearmor |
sudo tee /usr/share/keyrings/influxdb-archive-keyring.gpg >/dev/null
```

- **Step 3: Register the InfluxDB repository in your system**

```
echo "deb [signed-by=/usr/share/keyrings/influxdb-archive-keyring.gpg]
https://repos.influxdata.com/debian $(lsb_release -cs) stable"
| sudo tee /etc/apt/sources.list.d/influxdb.list
```

- **Step 4: Install InfluxDB**

  ```
  sudo apt install influxdb
  ```

- **Step 5: Start and Enable InfluxDB**

  ```
  sudo systemctl unmask influxdb
  sudo systemctl enable influxdb
  sudo systemctl start influxdb
  ```

- **Step 6: Create a database and insert data**
  - Start the InfluxDB client by typing `influx` in the terminal.
  - Create a new database:

    ```
    CREATE DATABASE <YOUR_DATABASE_NAME>
    USE <YOUR_DATABASE_NAME>
    ```

  - Insert sample data:

    ```
    INSERT temperature,location=living_room value=20
    INSERT temperature,location=living_room value=10
    ```

Part 2: Installing Grafana on Raspberry Pi

- **Step 1: Add the APT key used to authenticate packages**

  ```
  sudo mkdir -p /etc/apt/keyrings/
  ```

- **Step 2: Add the InfluxDB repository**

  ```
  curl https://repos.influxdata.com/influxdata-archive.key | gpg --dearmor |
  sudo tee /usr/share/keyrings/influxdb-archive-keyring.gpg >/dev/null
  ```

- **Step 3: Register the InfluxDB repository in your system**

  ```
  echo "deb [signed-by=/usr/share/keyrings/influxdb-archive-keyring.gpg]
  https://repos.influxdata.com/debian $(lsb_release -cs) stable"
  | sudo tee /etc/apt/sources.list.d/influxdb.list
  ```

- **Step 4: Install Grafana**

  ```
  sudo apt-get update
  sudo apt-get install -y grafana
  ```

- **Step 5: Start and Enable Grafana**

```
sudo /bin/systemctl enable grafana-server
sudo /bin/systemctl start grafana-server
```

- **Step 6: Access Grafana**
  - Open a browser and navigate to `localhost:3000`.
  - Login using `admin` as both username and password.
- **Step 7: Configure InfluxDB as a data source**
  - Go to Data Sources and select InfluxDB.
  - Configure the InfluxDB settings, including the database name and network configurations.

```
Enter URL => http://localhost:8086
```

- **Step 8: Create a Grafana dashboard**
  - Go to Dashboards and create a new one.
  - Add visualization panels for your data.
  - Choose the Table in the input space beside `FROM`.

Part 3: Writing to InfluxDB using Python

- **Step 1: Install the InfluxDB Python library**

```
python3 -m pip install influxdb
```

- **Step 2: Use Python script to insert Lidar data into the database**
  - Use the same virtual environment we have used in Lab3.
  - Modify the script to use your database and table names.
  - Modify the json-body in the influx.py code file provided.

1.4 Code

1.4.1 lidar.py

```python
import time
import board
import busio
import adafruit_vl53l0x

# Initialize I2C bus and sensor.
i2c = busio.I2C(board.SCL, board.SDA)
vl53 = adafruit_vl53l0x.VL53L0X(i2c)

# try:
#     Main loop will read the range and print it every second.
#     while True:
#         print("Range: {0}mm".format(vl53.range))
#         time.sleep(0.2)
# except KeyboardInterrupt:
```

```
#     print("Exit")  # Exit on CTRL+C

def fetch_data():
    try:
        return v153.range  ## Function returns the LIDAR data
    except:
        return -1000       ## Error Handling
```

Listing 1: Function to fetch data from LIDAR

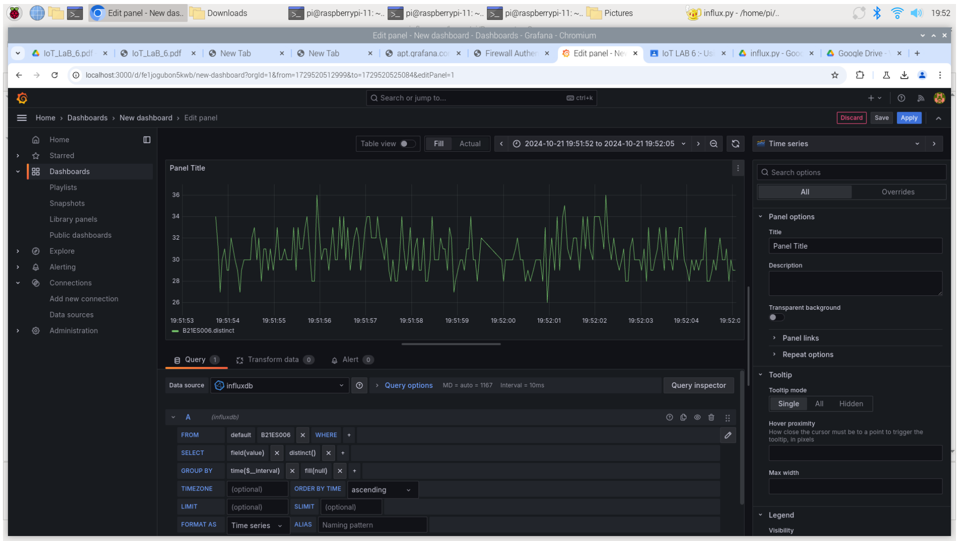
1.4.2 influx.py

```
from influxdb import InfluxDBClient
from lidar import fetch_data

# Connect to InfluxDB
client = InfluxDBClient(host='localhost', port=8086)
# Choosing the DB
client.switch_database('ROOM6')

# Running infinite loop to insert data from LIDAR continuously.
# Stop the INSERT by KeyboardInterrupt(Ctrl+C)
while True:
    try:
        data = fetch_data()
        json_body = [
            {
                "measurement": "B21ES006",
                "fields": {
                    "value"   : data
                }
            }
        ]
        print(client.write_points(json_body))  ## prints TRUE if insert is successful.
    except KeyboardInterrupt:
        print("exit")
```

Listing 2: Function in Listing 1 is used to INSERT the fetched data from LIDAR to InfluxDB (Here, DATABASE_NAME = 'ROOM6' & TABLE_NAME = 'B21ES006')

1.5 Observations & Results

- Successfully installed and configured InfluxDB and Grafana.
- Created a database and inserted sensor data using Python.
- Configured Grafana to visualize real-time data from InfluxDB.
- The Lidar data was visualized successfully on the Grafana dashboard.


Edit panel - New dashboard - Dashboards - Grafana - Chromium
localhost:3000/d/fe1jogubon5kwb/new-dashboard?orgId=1&from=1729520512999&to=1729520525084&editPanel=1
Home › Dashboards › New dashboard › Edit panel
Discard
Save
Apply
Table view
Fill
Actual
2024-10-21 19:51:52 to 2024-10-21 19:52:05
Time series
Panel Title
36
34
32
30
28
26
19:51:53
19:51:54
19:51:55
19:51:56
19:51:57
19:51:58
19:51:59
19:52:00
19:52:01
19:52:02
19:52:03
19:52:04
B21ES006.distinct

Query 1 Transform data 0 Alert 0

Data source: influxdb — Query options MD = auto = 1167 Interval = 10ms

A (influxdb)

FROM default B21ES006 WHERE

SELECT field(value) distinct()

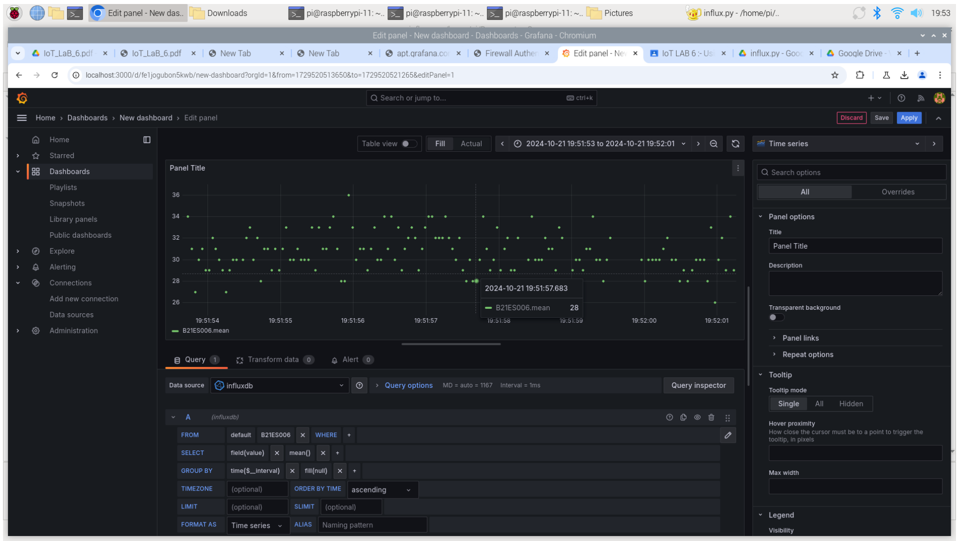
GROUP BY time($__interval) fill(null)

TIMEZONE (optional) ORDER BY TIME ascending

LIMIT (optional) SLIMIT (optional)

FORMAT AS Time series ALIAS Naming pattern

Figure 1: Real-time LIDAR data inserted in InfluxDB


Edit panel - New dashboard - Dashboards - Grafana - Chromium
localhost:3000/d/fe1jogubon5kwb/new-dashboard?orgId=1&from=1729520513650&to=1729520521265&editPanel=1
Home › Dashboards › New dashboard › Edit panel
Table view
Fill
Actual
2024-10-21 19:51:53 to 2024-10-21 19:52:01
Time series
Panel Title
36
34
32
30
28
26
19:51:54
19:51:55
19:51:56
19:51:57
19:51:58
19:51:59
19:52:00
19:52:01
2024-10-21 19:51:57.683
B21ES006.mean 28
B21ES006.mean

Query 1 Transform data 0 Alert 0

Data source: influxdb — Query options MD = auto = 1167 Interval = 1ms

A (influxdb)

FROM default B21ES006 WHERE

SELECT field(value) mean()

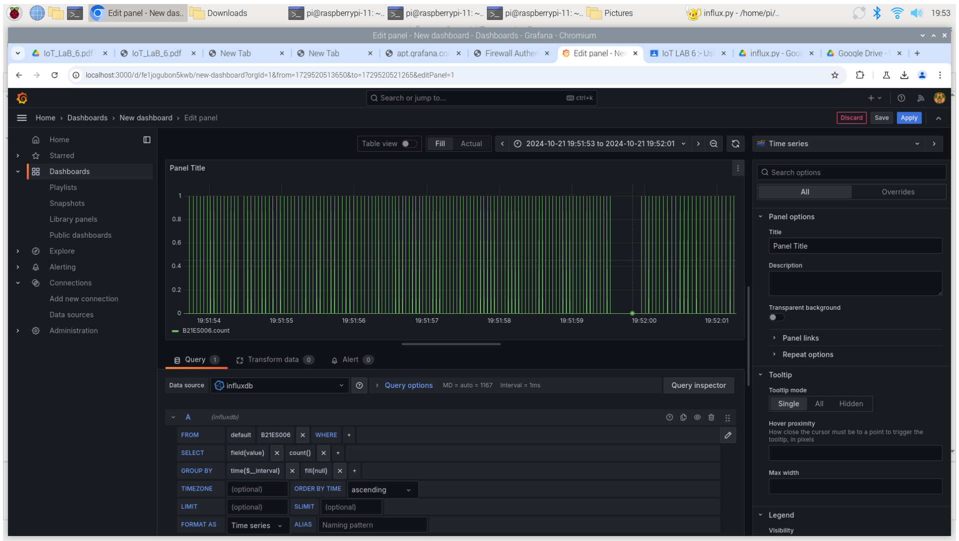
GROUP BY time($__interval) fill(null)

TIMEZONE (optional) ORDER BY TIME ascending

LIMIT (optional) SLIMIT (optional)

FORMAT AS Time series ALIAS Naming pattern

Figure 2: Visualizing the mean values of LIDAR data inserted into InfluxDB with a timestamp.


Edit panel - New dashboard - Dashboards - Grafana - Chromium
localhost:3000/d/fe1jogubon5kwb/new-dashboard?orgId=1&from=1729520513650&to=1729520521265&editPanel=1
Home › Dashboards › New dashboard › Edit panel
Table view
Fill
Actual
2024-10-21 19:51:53 to 2024-10-21 19:52:01
Time series
Panel Title
1
0.8
0.6
0.4
0.2
0
19:51:54
19:51:55
19:51:56
19:51:57
19:51:58
19:51:59
19:52:00
19:52:01
B21ES006.count

Query 1 Transform data 0 Alert 0

Data source: influxdb — Query options MD = auto = 1167 Interval = 1ms

A (influxdb)

FROM default B21ES006 WHERE

SELECT field(value) count()

GROUP BY time($__interval) fill(null)

TIMEZONE (optional) ORDER BY TIME ascending

LIMIT (optional) SLIMIT (optional)

FORMAT AS Time series ALIAS Naming pattern

Figure 3: Visualizing the number of entries inserted in the InfluxDB

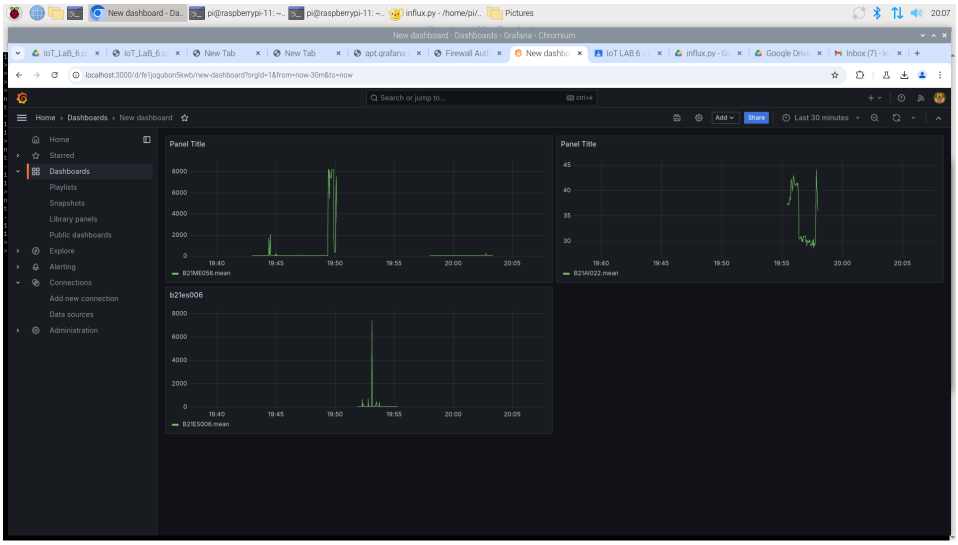


Figure 4: Visualizing the LIDAR data entries from all tables in our InfluxDB