Project Report

IO-Link Sensor Monitoring Using Docker, Node-RED, Elasticsearch & Kibana

1. Introduction

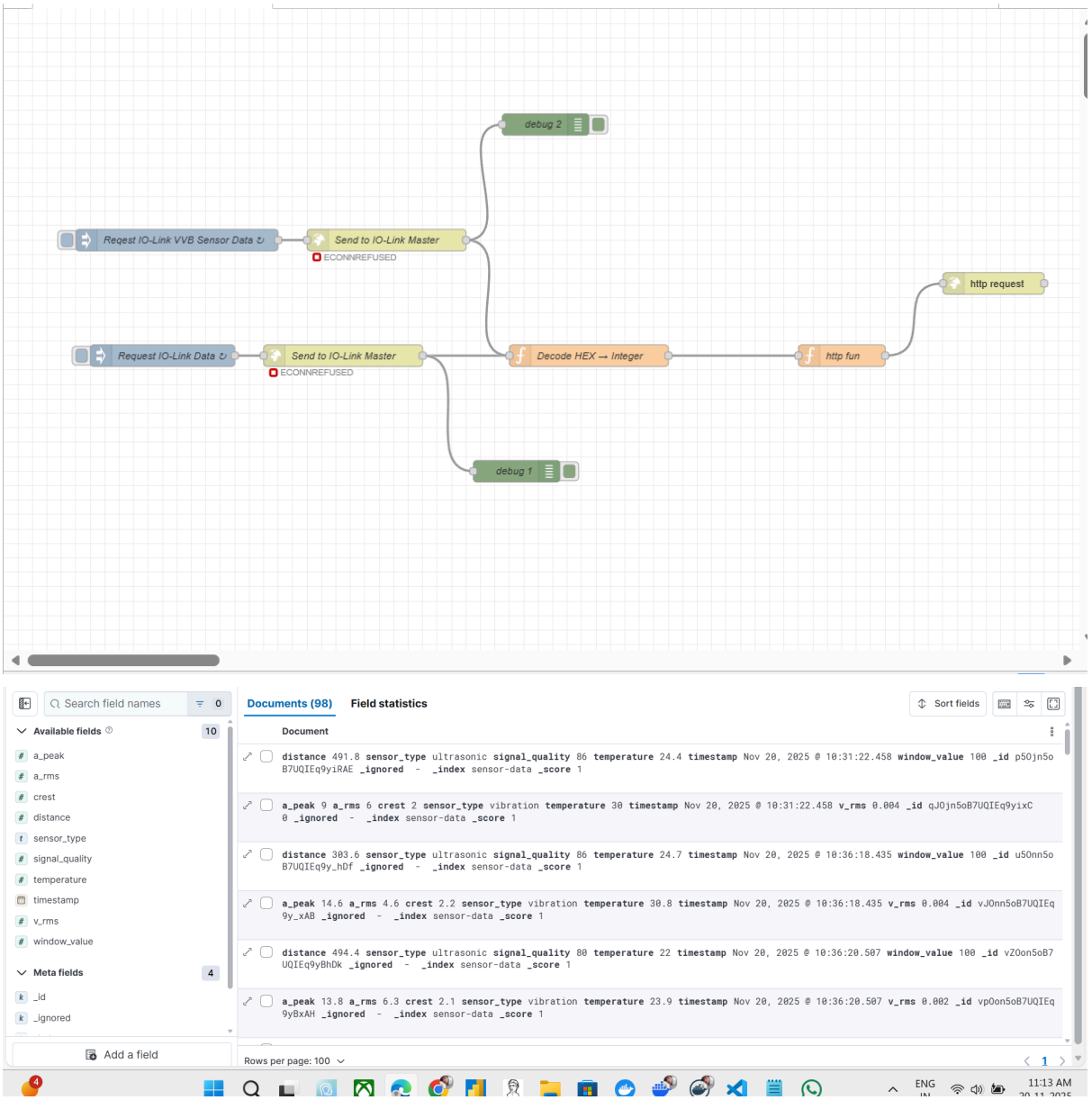
Industrial automation environments generate large amounts of real-time sensor data. Traditional systems lack flexibility for data storage, analytics, and visualization.
This project demonstrates a containerized solution to continuously collect IO-Link sensor data, store it in Elasticsearch, and visualize it in Kibana.

2. Objectives

- Collect IO-Link sensor readings using Node-RED
- Store sensor data in a scalable database (Elasticsearch)
- Visualize real-time metrics using Kibana dashboards
- Package entire solution in Docker for easy deployment

3. System Architecture

IO-Link Sensor → Node-RED → Elasticsearch → Kibana Dashboard



Components:

Component	Role
Node-RED	Collects sensor data & sends to database
Elasticsearch	Stores indexed time-series sensor data
Kibana	Provides UI dashboards and analysis
Docker & Compose	Deploys all components in isolated containers

4. Software Used

- **Node-RED 3.x**
- **Elasticsearch 8.15**
- **Kibana 8.15**
- **Docker & Docker Compose**

5. Implementation

5.1 Docker Deployment

All services are containerized using the docker-compose file:

- Elasticsearch exposed on port **9200**
- Kibana exposed on **5601**
- Node-RED exposed on **1880**
- Services share a common Docker network

5.2 Node-RED Flow

Node-RED periodically:

- Reads or simulates IO-Link sensor values

- Formats data as JSON
- Sends HTTP POST requests to Elasticsearch

Example payload:

```
{
  "sensor": "temperature",
  "value": 45.8,
  "unit": "°C"
}
```

5.3 Elasticsearch Data Index

Sensor documents are stored in index:

sensor_data

5.4 Kibana Visualization

Users can:

- Create Data Views
- Build dashboards
- Monitor real-time sensor performance

6. Testing

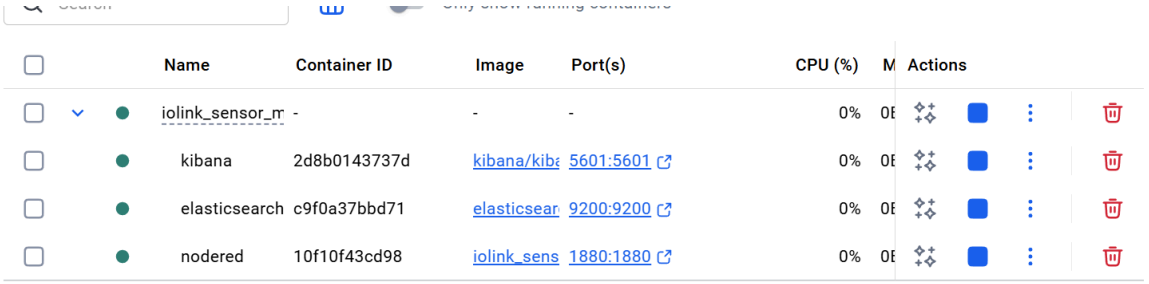
6.1 Insert Test Data (Windows PowerShell)

```
curl -Method Post "http://localhost:9200/sensor_data/_doc" `
-Headers @{ "Content-Type" = "application/json" } `
-Body '{ "sensor": "test", "value": 999 }'
```

6.2 Verify Index

```
curl http://localhost:9200/_cat/indices?v
```

7. Docker Hub Deployment



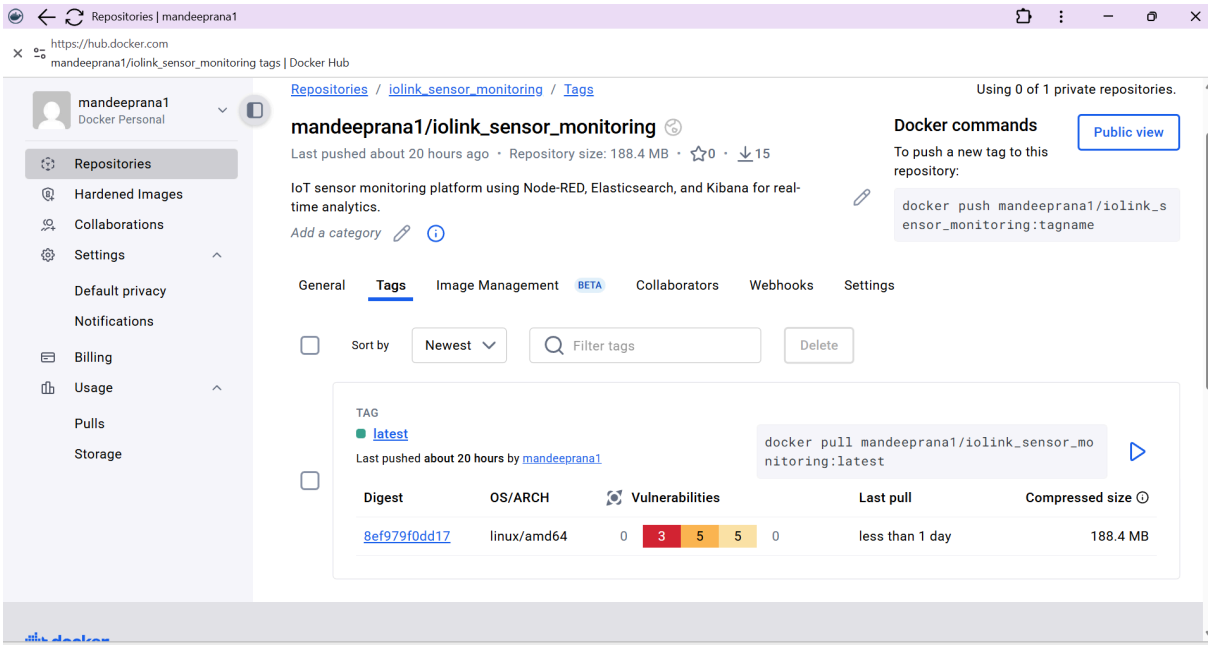


Image is tagged and pushed via:

docker tag iolink_sensor_monitoring-nodered mandeeprana1/iolink_sensor_monitoring:latest
docker push mandeeprana1/iolink_sensor_monitoring:latest

8. Results

- Data flows successfully from Node-RED → Elasticsearch
- Kibana displays real-time charts
- Full system deployable anywhere via Docker

9. Conclusion

This solution demonstrates a scalable, containerized industrial IoT monitoring system suitable for:

- Factory digitalization
- Predictive maintenance
- Real-time monitoring

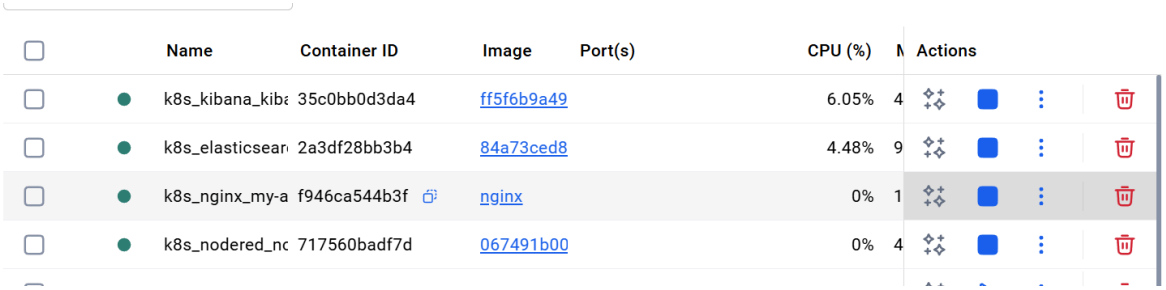
The architecture is modular and can be extended for machine learning, alerting, or edge deployment.

10. Future Enhancements

- Add Grafana as an alternate visualization tool
- Add MQTT broker like Mosquitto
- Implement anomaly detection via ML models

11. Kubernetes Deployment Status

The IO-Link Sensor Monitoring System has been successfully deployed on Kubernetes. Current status is as follows:



Name	Container ID	Image	Port(s)	CPU (%)	M	Actions
k8s_kibana_kib	35c0bb0d3da4	ff5f6b9a49		6.05%	4	
k8s_elasticsear	2a3df28bb3b4	84a73ced8		4.48%	9	
k8s_nginx_my-a	f946ca544b3f	nginx		0%	1	
k8s_nodered_nc	717560badf7d	067491b00		0%	4	

Deployments

Name	Status	Pods
elasticsearch	Available	1/1
kibana	Available	1/1
my-app	Available	1/1
nodered	Available	1/1

Pods

Name	Status
elasticsearch-ffcbc946b-lnbct	Running
kibana-8497f84f94-hj49r	Running
my-app-f85d86645-96bnq	Running
nodered-ddfc8f759-92wtr	Running

Nodes

Name	Status
docker-desktop	Ready

Services

Name	Cluster IP	Ports
elasticsearch-service	10.111.144.162	9200/TCP
kibana-service	10.108.115.135	5601/TCP
kubernetes	10.96.0.1	443/TCP
nodered-service	10.98.158.70	1880/TCP

```
NAME                                 READY   STATUS    RESTARTS   AGE
pod/elasticsearch-ffcbc946b-lnbct    1/1     Running   2          5d10h
pod/kibana-8497f84f94-hj49r          1/1     Running   2          5d10h
pod/my-app-f85d86645-96bnq           1/1     Running   2          5d18h
pod/nodered-ddfc8f759-92wtr          1/1     Running   0          58s

NAME                                 TYPE        CLUSTER-IP       EXTERNAL-IP   PORT(S)          AGE
service/elasticsearch-service        NodePort    10.111.144.162   <none>        9200:30082/TCP   5d20h
service/kibana-service               NodePort    10.108.115.135   <none>        5601:30081/TCP   5d20h
service/kubernetes                   ClusterIP   10.96.0.1        <none>        443/TCP          7d21h
service/nodered-service              NodePort    10.98.158.70     <none>        1880:30080/TCP   5d20h
```

Pods Status

Pod Name	Ready	Status	Restarts	Age
elasticsearch-ffcbc946b-lnbct	1/1	Running	2	5d10h
kibana-8497f84f94-hj49r	1/1	Running	2	5d10h
my-app-f85d86645-96bnq	1/1	Running	2	5d18h
nodered-ddfc8f759-92wtr	1/1	Running	0	58s

Services Status

Service Name	Type	Cluster IP	External IP	Ports	Age
elasticsearch-service	NodePort	10.111.144.162		9200:30082/TCP	5d20h
kibana-service	NodePort	10.108.115.135		5601:30081/TCP	5d20h
kubernetes	ClusterIP	10.96.0.1		443/TCP	7d21h
nodered-service	NodePort	10.98.158.70		1880:30080/TCP	5d20h

Observations

- All pods are running and healthy.
- Node-RED pod is newly started (58 seconds), ready for flow deployment.
- NodePort services allow external access via node IPs:
    - Kibana Dashboard: `http://<node-ip>:30081`
    - Node-RED Editor: `http://<node-ip>:30080`
    - Elasticsearch API: `http://<node-ip>:30082`
- The cluster is operational and ready for further scaling and deployment of additional IoT monitoring features.