

NANDHA ENGINEERING COLLEGE

ERODE–638052 (Autonomous)

(Affiliated to Anna University, Chennai)



DEPARTMENT OF ARTIFICIAL INTELLIGENCE AND DATA SCIENCE

22AIC14 – INTERNET OF THINGS AND ITS APPLICATION

MINI PROJECT REPORT ON

TOPIC – WATER PIPELINE BLOCKAGE DETECTION SYSTEM

Submitted by

REGISTER NUMBER	NAME
22AI015	GOBIKA. I
22AI019	KANISHGAA. A
22AI029	NARMATHA. M

NANDHA ENGINEERING COLLEGE

(An Autonomous Institution, Affiliated to Anna University, Chennai)

BONAFIDE CERTIFICATE

This is to certify that the project work entitled “WATER PIPELINE BLOCKAGE DETECTION SYSTEM” is the Bonafide work of GOBIKA I (22AI015), KANISHGAA A (22AI019), NARMATHA M (22AI029) who carried out the work under my supervision.

Signature of the Supervisor

**Dr. K. Lalitha,
Professor,
Department of AI & DS,
Nandha Engineering College,
Erode – 638052.**

Signature of the HOD

**Dr. P. Karunakaran,
Head of the Department,
Department of AI & DS,
Nandha Engineering College,
Erode – 638052.**

Submitted for End semester PBL review held on _____

WATER PIPELINE BLOCKAGE DETECTION SYSTEM

AIM:

The aim of this project is to design and implement an IoT-based system capable of detecting the exact location of a blockage in a water pipeline. The system ensures timely identification and alerts, enabling efficient maintenance and reducing water wastage.

SCOPE:

This project provides a solution to a persistent challenge in water distribution systems—detecting and pinpointing blockages in pipelines. The system is designed to be highly effective in various applications, such as urban water supply networks, industrial pipelines, and agricultural irrigation systems. By leveraging IoT technology, the solution offers real-time data monitoring and enhances operational efficiency. It also significantly reduces the time and effort required for manual inspection, making it a practical choice for scalable water management systems.

BRIEF HISTORY:

Pipeline monitoring has traditionally relied on manual inspections, which are labour-intensive, time-consuming, and often unreliable. Basic techniques like measuring flow rates at entry and exit points lacked precision in detecting blockages. In recent years, advancements in IoT and sensor technologies have revolutionized the way pipelines are monitored. Automated systems now allow for real-time data collection and processing, providing accurate detection of blockages. These systems not only improve efficiency but also reduce costs, paving the way for smarter water management solutions.

PROPOSED METHODOLOGY:

The methodology for the pipeline blockage detection system is as follows:

1. The system is set up by installing a flow rate sensor at strategic points in the pipeline to measure the flow of water. The data collected by the sensor is processed using an Arduino Nano.
2. The flow rate data is continuously compared to a predefined threshold to identify irregularities. Deviations from normal flow patterns indicate the presence of a blockage.
3. When a blockage is detected, an LED is activated to alert users nearby. If integrated with a web interface, the data can also be transmitted for remote monitoring.

4. The exact location of the blockage is determined based on the position of the affected section in the pipeline.
- 5.

COMPONENTS REQUIRED:

S.NO	HARDWARE	QUANTITY
1	Flow rate sensor	1
2	Arduino Nano	1
3	LED	1
4	Resistor (220 ohm)	1
5	Jumper Wires	As required
6	USB Cable	1

DESCRIPTION:

The pipeline blockage detection system is an IoT-based solution that utilizes flow rate sensors and an Arduino Nano microcontroller to monitor and analyse the flow of water through a pipeline. The system works by placing flow rate sensors at multiple locations along the pipeline to measure the water flow at each segment. These sensors generate data in real time, which is processed by the Arduino Nano.

Any deviation in flow rate beyond a predefined threshold is considered abnormal and indicates the presence of a blockage. The Arduino Nano then determines which segment of the pipeline is affected. This process allows the system to pinpoint the exact location of the blockage with high accuracy, minimizing the need for extensive manual investigation.

As a local alert mechanism, an LED is activated whenever a blockage is detected. This visual alert is simple yet effective for on-site monitoring. For advanced applications, the system can be integrated with an IoT platform or a web-based interface to enable remote monitoring. This feature allows users to view real-time data and receive notifications about pipeline conditions even when they are not physically present.

The design of this system emphasizes simplicity and efficiency. It is cost effective and scalable, making it suitable for a wide range of applications. By integrating IoT technology, the system offers automated and precise blockage detection, enhancing the reliability of pipeline maintenance and operation.

Overall, the project not only addresses the issue of blockage detection but also contributes to more sustainable water resource management. Its ability to alert users promptly

and provide detailed information about the pipeline's status ensures quick resolution of issues, reducing downtime and resource wastage.

CODING:

```
// Pin Definitions
```

```
#define FLOW_SENSOR_PIN 2
```

```
#define LED_PIN 13
```

```
// Flow Rate Thresholds
```

```
#define BLOCKED_FLOW_RATE 3.0
```

```
#define SLOW_FLOW_RATE 5.0
```

```
// Pipeline Length (in meters)
```

```
#define PIPELINE_LENGTH 0.28
```

```
// Variables
```

```
volatile int pulseCount = 0;
```

```
float flowRate = 0;
```

```
unsigned long oldTime = 0;
```

```
float blockageLocation = 0;
```

```
void setup() {
```

```
    // Initialize Serial Monitor
```

```
    Serial.begin(9600);
```

```
    // Set up the flow sensor and LED
```

```
    pinMode(FLOW_SENSOR_PIN, INPUT);
```

```
    pinMode(LED_PIN, OUTPUT);
```

```
// Attach interrupt for the flow sensor
attachInterrupt(digitalPinToInterrupt(FLOW_SENSOR_PIN), pulseCounter, RISING);
}

void loop() {
    // Measure flow rate every second
    if (millis() - oldTime > 1000) {
        detachInterrupt(digitalPinToInterrupt(FLOW_SENSOR_PIN));

        // Calculate flow rate in L/min
        flowRate = (pulseCount / 7.5); // Formula: pulses per second / 7.5 = L/min
        pulseCount = 0;
        oldTime = millis();

        // Print flow rate to the Serial Monitor
        Serial.print("Flow Rate: ");
        Serial.print(flowRate);
        Serial.println(" L/min");

        // Check if the flow rate indicates a blockage or slow flow
        if (flowRate <= BLOCKED_FLOW_RATE) {
            Serial.println("Status: Blocked");
            digitalWrite(LED_PIN, HIGH);

            // Calculate and print blockage location
            blockageLocation = estimateBlockageLocation(flowRate);
            Serial.print("Estimated Blockage Location: ");
            Serial.print(blockageLocation);
            Serial.println(" meters");
        }
    }
}
```

```

    } else if (flowRate > BLOCKED_FLOW_RATE && flowRate <= SLOW_FLOW_RATE)
    {
        Serial.println("Status: Flow is Slow");
        blinkLED(LED_PIN);
    } else {
        Serial.println("Status: Normal Flow");
        digitalWrite(LED_PIN, LOW);
    }

    attachInterrupt(digitalPinToInterrupt(FLOW_SENSOR_PIN), pulseCounter, RISING); //
    Re-enable interrupt
}

}

void pulseCounter() {
    pulseCount++;
}

// Function to estimate blockage location (only when flow is blocked)
float estimateBlockageLocation(float currentFlowRate) {
    // If the flow rate is greater than the threshold, assume no blockage
    if (currentFlowRate > BLOCKED_FLOW_RATE) return 0.0;

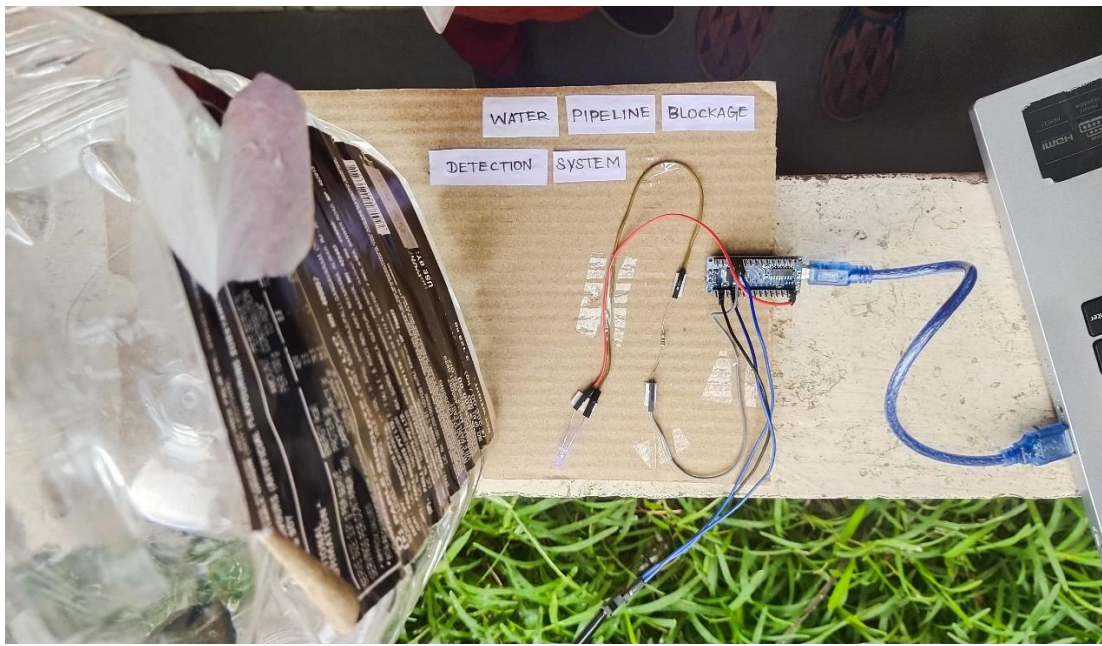
    // Calculate the blockage location as a proportion of the pipeline length
    return PIPELINE_LENGTH * (currentFlowRate / BLOCKED_FLOW_RATE);
}

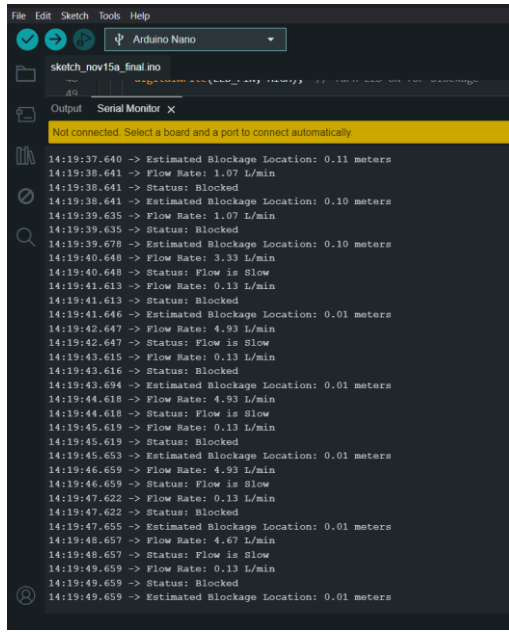
// Function to blink LED
void blinkLED(int pin) {
    digitalWrite(pin, HIGH);

```

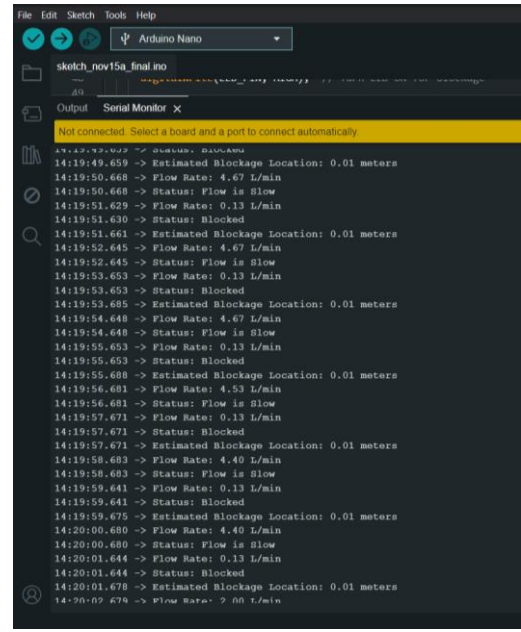
```
delay(500); // LED ON for 500ms  
digitalWrite(pin, LOW);  
delay(500); // LED OFF for 500ms  
}
```

OUTPUT SCREENSHOT:





```
File Edit Sketch Tools Help
sketch_nov15a_final.ino
Output Serial Monitor x
Not connected. Select a board and a port to connect automatically.
14:19:37.640 -> Estimated Blockage Location: 0.11 meters
14:19:38.641 -> Flow Rate: 1.07 L/min
14:19:38.641 -> Status: Blocked
14:19:38.641 -> Estimated Blockage Location: 0.10 meters
14:19:39.635 -> Flow Rate: 1.07 L/min
14:19:39.635 -> Status: Blocked
14:19:39.678 -> Estimated Blockage Location: 0.10 meters
14:19:40.648 -> Flow Rate: 3.33 L/min
14:19:40.648 -> Status: Flow is Slow
14:19:41.613 -> Flow Rate: 0.13 L/min
14:19:41.613 -> Status: Blocked
14:19:41.646 -> Estimated Blockage Location: 0.01 meters
14:19:42.647 -> Flow Rate: 4.93 L/min
14:19:42.647 -> Status: Flow is Slow
14:19:43.615 -> Flow Rate: 0.13 L/min
14:19:43.616 -> Status: Blocked
14:19:43.694 -> Estimated Blockage Location: 0.01 meters
14:19:44.618 -> Flow Rate: 4.93 L/min
14:19:44.618 -> Status: Flow is Slow
14:19:45.619 -> Flow Rate: 0.13 L/min
14:19:45.619 -> Status: Blocked
14:19:45.659 -> Estimated Blockage Location: 0.01 meters
14:19:46.659 -> Flow Rate: 4.93 L/min
14:19:46.659 -> Status: Flow is Slow
14:19:47.622 -> Flow Rate: 0.13 L/min
14:19:47.622 -> Status: Blocked
14:19:47.655 -> Estimated Blockage Location: 0.01 meters
14:19:48.657 -> Flow Rate: 4.67 L/min
14:19:48.657 -> Status: Flow is Slow
14:19:49.659 -> Flow Rate: 0.13 L/min
14:19:49.659 -> Status: Blocked
14:19:49.659 -> Estimated Blockage Location: 0.01 meters
```



```
File Edit Sketch Tools Help
sketch_nov15a_final.ino
Output Serial Monitor x
Not connected. Select a board and a port to connect automatically.
14:19:49.659 -> Estimated Blockage Location: 0.01 meters
14:19:50.668 -> Flow Rate: 4.67 L/min
14:19:50.668 -> Status: Flow is Slow
14:19:51.629 -> Flow Rate: 0.13 L/min
14:19:51.630 -> Status: Blocked
14:19:51.661 -> Estimated Blockage Location: 0.01 meters
14:19:52.645 -> Flow Rate: 4.67 L/min
14:19:52.645 -> Status: Flow is Slow
14:19:53.653 -> Flow Rate: 0.13 L/min
14:19:53.653 -> Status: Blocked
14:19:53.695 -> Estimated Blockage Location: 0.01 meters
14:19:54.648 -> Flow Rate: 4.67 L/min
14:19:54.648 -> Status: Flow is Slow
14:19:55.653 -> Flow Rate: 0.13 L/min
14:19:55.653 -> Status: Blocked
14:19:55.698 -> Estimated Blockage Location: 0.01 meters
14:19:56.681 -> Flow Rate: 4.53 L/min
14:19:56.681 -> Status: Flow is Slow
14:19:57.671 -> Flow Rate: 0.13 L/min
14:19:57.671 -> Status: Blocked
14:19:57.671 -> Estimated Blockage Location: 0.01 meters
14:19:58.683 -> Flow Rate: 4.40 L/min
14:19:58.683 -> Status: Flow is Slow
14:19:59.641 -> Flow Rate: 0.13 L/min
14:19:59.641 -> Status: Blocked
14:19:59.675 -> Estimated Blockage Location: 0.01 meters
14:20:00.680 -> Flow Rate: 4.40 L/min
14:20:00.680 -> Status: Flow is Slow
14:20:01.644 -> Flow Rate: 0.13 L/min
14:20:01.644 -> Status: Blocked
14:20:01.678 -> Estimated Blockage Location: 0.01 meters
14:20:02.678 -> Flow Rate: 2.60 L/min
```

PROTOCOLS USED:

In this project, the following protocols and techniques are used to ensure seamless operation:

1. Data Collection Protocol:

The flow rate sensor collects real-time data based on water movement within the pipeline. This data is transmitted to the Arduino Nano for processing.

2. Serial Communication Protocol:

The Arduino Nano communicates with the computer or other devices through USB using a serial protocol, enabling data visualization and debugging.

3. Threshold-Based Analysis:

A predefined threshold protocol is implemented to detect deviations in flow rates, which helps in identifying potential blockages.

LIMITATIONS:

1. Effective blockage detection requires precise sensor placement, which can be challenging for long or complex pipeline networks.
2. The system relies on continuous power, making it less practical in remote or off-grid locations.
3. The current design lacks wireless communication capabilities for remote monitoring.
4. Sensor readings may be affected by debris or water quality, potentially reducing accuracy.

5. Expanding the system to cover extensive pipeline networks increases cost and complexity.

FUTURE WORK:

1. Incorporating Wi-Fi, GSM, or LoRa modules for remote monitoring and notifications.
2. Integrating machine learning models to predict potential blockages based on historical flow data.
3. Developing self-powered sensors using energy-harvesting techniques like micro-turbines.
4. Adding sensors for water quality metrics, such as pH and turbidity, for a more comprehensive monitoring solution.
5. Designing mobile apps or web dashboards for better user interaction and monitoring.
6. Optimizing the system design to efficiently manage larger pipelines while reducing installation costs.

CONCLUSION:

The pipeline blockage detection system demonstrates an effective use of IoT technology for identifying and locating blockages in pipelines. By using flow rate sensors and an Arduino Nano, the system ensures accurate and timely alerts, reducing manual intervention and maintenance time. Its cost-effective design makes it suitable for diverse applications. Future enhancements could include predictive maintenance features and improved remote monitoring capabilities, making the system even more efficient and scalable.