Week 1 Submission – Market & Industry Research (Bullet Format)

Industries Using Fluid Pipelines

- Oil & gas transmission pipelines: more than 3 million km worldwide
- Chemical and petrochemical plants operate pipelines continuously (24/7)
- Water distribution systems experience 30–40% water loss due to leaks
- Power plants use cooling water pipelines with flow rates up to 10,000 m^3/hr
- Food and beverage industries rely on hygienic liquid transport pipelines

Common Pipeline Faults – Industrial Data

- Leakage contributes to 60–70% of pipeline failures
- Blockage causes 20–30% reduction in flow efficiency
- Pump-related faults cause 15–20% of unplanned downtime
- Pipeline maintenance costs account for up to 25% of operational cost

Impact of Pipeline Failures

- Annual product loss of 5–10% due to pipeline faults
- High safety risk from chemical leaks
- Industrial downtime cost ranges from $10,000 to $50,000 per hour
- Environmental damage including soil and groundwater contamination

Existing Monitoring Methods and Limitations

- Manual inspection and routine maintenance
- Pressure and flow gauges with fixed alarm thresholds
- Fault detection occurs only after failure
- No predictive or early warning capability
- High dependence on human intervention

Industry Trend and Project Relevance

- Predictive maintenance reduces downtime by 30–40%
- Sensor-based monitoring improves fault detection accuracy above 85%
- Machine learning enables early-stage anomaly detection
- Industry 4.0 focuses on smart sensors and real-time analytics

Problem Statement – Summary

- Pipeline systems are critical in chemical and process industries
- Common faults include leakage, blockage, and pump failure
- Conventional monitoring systems are reactive

- Early fault detection is required to improve safety
- Proposed system uses sensors, microcontroller, and machine learning
- Objective is early warning and improved operational reliability

Pipeline Operations

1. Introduction

Pipeline operations refer to all technical and managerial activities required to transport fluids such as crude oil, refined petroleum products, natural gas, water, and chemicals safely and efficiently from one location to another. Once a pipeline is constructed and commissioned, operations continue throughout its entire service life, focusing on safety, reliability, efficiency, and environmental protection.

Pipelines are widely used because they provide continuous transport, low operating cost per unit volume, and reduced risk compared to road or rail transport when properly managed.[1]

2. Objectives of Pipeline Operations

The main objectives of pipeline operations are:

- To maintain uninterrupted and controlled flow of fluid
- To operate within safe pressure and temperature limits
- To detect leaks, faults, or abnormal conditions at an early stage
- To minimize energy consumption in pumping or compression
- To comply with safety, environmental, and regulatory requirements
- To support inspection, maintenance, and long-term integrity of the pipeline

Achieving these objectives requires continuous monitoring and training operational staff.

3. Key Components in Pipeline Operations

3.1 Field Instrumentation

Field instrumentation includes sensors and measuring devices installed along the pipeline and at stations. These instruments continuously measure:

- Pressure
- Flow rate
- Temperature

The data collected reflects the real-time condition of the pipeline and is essential for safe operation.[2]

3.2 Remote Terminal Units (RTUs)

Remote Terminal Units act as intermediaries between field instruments and the central control room. RTUs:

- Collect data from sensors
- Transmit data using communication systems (fiber optic, radio, satellite, cellular)
- Execute remote commands such as opening or closing valves

RTUs are especially important for long-distance pipelines that pass through remote areas.[3]

3.3 SCADA System (Supervisory Control and Data Acquisition)

SCADA is the core of pipeline operations. It allows operators to:

- Monitor real-time data from the entire pipeline
- Visualize pipeline status through graphical interfaces
- Control pumps, compressors, and valves remotely
- Receive alarms during abnormal conditions

Modern SCADA systems also support leak detection, pig tracking, batch tracking, and predictive analytics.[4]

3.4 Pump and Compressor Stations

- **Pump stations** are used in liquid pipelines to increase pressure and maintain flow.

- **Compressor stations** are used in gas pipelines to compensate for pressure losses due to friction and elevation changes.

These stations are strategically located along the pipeline route.

3.5 Valves and Isolation Systems

Valves control and regulate flow and are critical for safety. Common types include:

- Gate valves
- Ball valves
- Check valves
- Emergency Shut-Down (ESD) valves

They allow isolation of pipeline sections during maintenance or emergencies.[5]

4. Routine Pipeline Operational Activities

4.1 Flow and Pressure Management

Operators continuously adjust pumps, compressors, and valves to:

- Maintain target flow rates
- Prevent overpressure or vacuum conditions
- Match supply with demand

4.2 Monitoring and Control

Using SCADA dashboards, operators can:

- Identify abnormal trends
- Respond to alarms
- Automate routine control actions

4.3 Pigging Operations

Pigging involves sending devices called pigs through the pipeline to:

- Clean internal deposits
- Remove wax or debris
- Inspect internal condition using smart pigs[6]

4.4 Corrosion Control and Cathodic Protection

Cathodic protection systems prevent external corrosion by supplying a controlled electrical current to the pipeline.[7]

4.5 Maintenance and Inspection

Maintenance includes:

- Routine equipment servicing
- Calibration of sensors
- Repair or replacement of damaged components

Inspection data helps plan maintenance before failures occur.

5. Safety and Emergency Management

Pipeline operators implement:

- Emergency response plans
- Automatic shutdown systems
- Operator training programs

These measures reduce risks to people, property, and the environment.

References

[1] “Pipeline - Wikipedia.” Accessed: Jan. 18, 2026. [Online]. Available: https://en.wikipedia.org/wiki/Pipeline

[2] “Industrial process control - Wikipedia.” Accessed: Jan. 18, 2026. [Online]. Available: https://en.wikipedia.org/wiki/Industrial_process_control

[3] “Remote terminal unit - Wikipedia.” Accessed: Jan. 18, 2026. [Online]. Available: https://en.wikipedia.org/wiki/Remote_terminal_unit

[4] “SCADA - Wikipedia.” Accessed: Jan. 18, 2026. [Online]. Available: https://en.wikipedia.org/wiki/SCADA

[5] “Process Systems - Manufacture Valves & Flow Control Equipment.” Accessed: Jan. 18, 2026. [Online]. Available: https://www.valvesonline.com.au/

[6] “Pigging - Wikipedia.” Accessed: Jan. 18, 2026. [Online]. Available: https://en.wikipedia.org/wiki/Pigging

[7] “Cathodic protection - Wikipedia.” Accessed: Jan. 18, 2026. [Online]. Available: https://en.wikipedia.org/wiki/Cathodic_protection

Fault Identification in Pipeline Systems

1. Introduction

Fault identification in pipelines is essential to prevent accidents, environmental damage, and economic losses. Pipeline faults can develop slowly over time or occur suddenly due to operational or external factors. Early detection allows timely corrective action.[1]

2. Common Types of Pipeline Faults

2.1 Corrosion

Corrosion is the gradual loss of metal due to chemical or electrochemical reactions.

- Internal corrosion: caused by transport fluid
- External corrosion: caused by soil and moisture[2]

2.2 Leakages

Leaks may result from corrosion, cracks, weld failures, or mechanical damage. Even small leaks can have serious environmental and safety impacts.[3]

2.3 Cracks and Weld Defects

Cracks and poor welds may develop due to fatigue, stress, or improper construction and can grow over time.[4]

2.4 Blockages

Blockages occur due to sediment, wax buildup, or foreign materials, leading to pressure increase and flow reduction.

3. Fault Detection and Identification Methods

3.1 In-Line Inspection (Smart Pigging)

Smart pigs detect:

- Corrosion
- Cracks
- Deformation

Technologies include Magnetic Flux Leakage (MFL) and Ultrasonic Testing (UT).[5]

3.2 Non-Destructive Testing (NDT)

Common NDT techniques:

- Ultrasonic Testing (UT)
- Magnetic Flux Leakage (MFL)
- Radiographic Testing (RT)
- Eddy Current Testing (ECT)[6]

3.3 Leak Detection Techniques

- Hydrostatic testing
- Acoustic and vibration monitoring
- Fiber optic sensing
- Infrared thermography
- Flow and pressure analysis using SCADA data

[7], [8]

4. Advanced and Modern Approaches

Model-based and AI techniques analyze real-time data to detect abnormal patterns and classify faults more accurately.[9]

5. Best Practices

- Use multiple detection methods together
- Perform regular inspections

- Prioritize high-risk pipeline sections
- Maintain accurate operational data

References

[1] "What Is Pipeline Integrity? Methods, Standards | NWEgroup." Accessed: Jan. 18, 2026. [Online]. Available: https://nwegroup.no/what-is-pipeline-integrity/

[2] "Pipeline Corrosion Monitoring: Techniques and Systems | Polyguard." Accessed: Jan. 18, 2026. [Online]. Available: https://polyguard.com/blog/pipeline-corrosion-monitoring

[3] "Pipeline Leak Detection & Testing: Methods & Best Practices - NiGen." Accessed: Jan. 18, 2026. [Online]. Available: https://nigen.com/pipeline-leak-detection-testing/

[4] Y. Liang, B. Yu, M. Ding, W. Hu, Y. Jin, and Y. Yuan, "WELD-DETR: A Real-Time Welding Defect Detection Framework with Multi-Scale Feature Fusion and Multi-Kernel Perception Optimization," *Sensors 2025, Vol. 25, Page 7024*, vol. 25, no. 22, p. 7024, Nov. 2025, doi: 10.3390/S25227024.

[5] "Overview of Pipeline Inspection: Procedure, Types & Benefits." Accessed: Jan. 18, 2026. [Online]. Available: https://www.petrosync.com/blog/pipeline-inspection/

[6] "Pipeline Corrosion Monitoring: Techniques and Systems | Polyguard." Accessed: Jan. 18, 2026. [Online]. Available: https://polyguard.com/blog/pipeline-corrosion-monitoring

[7] "Hydrostatic test - Wikipedia." Accessed: Jan. 18, 2026. [Online]. Available: https://en.wikipedia.org/wiki/Hydrostatic_test

[8] "Pipeline Leak Detection & Testing: Methods & Best Practices - NiGen." Accessed: Jan. 18, 2026. [Online]. Available: https://nigen.com/pipeline-leak-detection-testing/

[9] S. Ferreno-Gonzalez, V. Diaz-Casas, M. Miguez-Gonzalez, and C. G. San-Gabino, "Detection of Pipe Ruptures in Shipboard Firefighting Systems Using Machine Learning and Deep Learning Techniques," *Applied Sciences 2025, Vol. 15, Page 1181*, vol. 15, no. 3, p. 1181, Jan. 2025, doi: 10.3390/APP15031181.

System Requirements and Methodology Report

1. Introduction

Defining system requirements is a critical step in the design and development of an engineering system. System requirements describe **what the system must do**, **how well it must perform**, and **under what conditions it must operate**. In the proposed project, clear system requirements ensure that the pipeline monitoring system can reliably detect faults such as leaks, blockages, and pump failures using sensor data and machine learning techniques.

2. Functional Requirements

The functional requirements specify the **core functions** that the system must perform.

2.1 Fault Detection Capability

The system shall be capable of detecting the following pipeline faults:

- Fluid leakage
- Pipeline blockage
- Pump malfunction or abnormal operation

Each fault must be identified based on deviations in pressure, flow rate, and temperature patterns.

2.2 Real-Time Monitoring

The system shall continuously monitor pipeline parameters in real time and update sensor readings at regular intervals to ensure early fault detection.

2.3 Data Acquisition

The system shall acquire data from:

- Pressure sensors
- Flow sensors
- Temperature sensors

All data must be converted into digital form and transmitted to the processing unit without loss or distortion.

2.4 Intelligent Decision Making

The system shall use a trained machine learning model to:

- Classify pipeline conditions as normal or faulty
- Distinguish between different types of faults

2.5 Alert Generation

When an abnormal condition is detected, the system shall:

- Trigger a visual alert (LED)
- Trigger an audible alert (buzzer)
- Display a warning message indicating the detected fault

3. Performance Requirements

Performance requirements define **how well** the system must operate.

3.1 Detection Speed

The system shall detect abnormal conditions within a short response time after fault occurrence to prevent further damage or loss.

3.2 Accuracy

The machine learning model shall achieve high classification accuracy in distinguishing normal and faulty conditions to minimize false alarms and missed detections.

3.3 Reliability

The system shall operate continuously for extended periods without system failure or data loss.

4. Hardware Requirements

The hardware components must satisfy the following requirements:

4.1 Sensors

- Sensors must operate within appropriate pressure, flow, and temperature ranges.
- Sensors must provide stable and repeatable readings.

4.2 Microcontroller

- The microcontroller (ESP32) shall support multiple sensor inputs.
- It shall have sufficient processing speed and memory.
- Wireless communication capability is required for data transmission.

5. Software Requirements

5.1 Data Processing

The system software shall:

- Filter noise from sensor readings
- Format data for machine learning analysis

5.2 Machine Learning Model

The machine learning algorithm shall:

- Be trained using labeled sensor data
- Be capable of real-time prediction

5.3 User Interface

The system shall provide a simple and clear interface to display:

- Live sensor values
- Fault detection results

6. Safety and Operational Requirements

- The system shall operate safely under normal laboratory conditions.
- Electrical components shall be properly insulated.
- The system shall not interfere with normal pipeline operation during testing.

7. Scalability and Future Expansion

The system shall be designed such that:

- Additional sensors can be integrated in the future
- The system can be adapted for larger industrial pipelines
- Cloud-based monitoring can be implemented if required

8. Importance of System Requirements

Clearly defining system requirements ensures:

- Proper hardware selection
- Accurate data collection
- Reliable machine learning performance
- Successful fault detection and alerting

Failure to define these requirements accurately could lead to incorrect fault detection, unreliable system performance, and project failure.

Sensors and Controllers Used in Pipeline Monitoring

Sensors are used to continuously monitor physical parameters of the pipeline and detect abnormal conditions.

1.Sensors used in pipeline monitoring

1.1 Pressure Sensor

Purpose:

Measures the internal pressure of fluid inside the pipeline.

Why it is important:

- Sudden pressure drop → **Leak**
- Sudden pressure increase → **Blockage or valve malfunction**

Common Pressure Sensors:

- MPX5010 / MPX5700
- BMP180 / BMP280
- Industrial pressure transducers

Advantages:

- High accuracy
- Fast response
- Essential for leak detection

Limitations:

- Sensitive to vibration
- Requires calibration

1.2 Flow Sensor

Purpose:

Measures the flow rate of fluid through the pipeline.

Why it is important:

- Reduced flow → **Clogging or blockage**
- Irregular flow → **Pump or valve fault**

Common Flow Sensors:

- YF-S201 (water flow)
- Turbine flow sensor
- Ultrasonic flow sensor (industrial)

Advantages:

- Real-time monitoring
- Easy to interface with microcontrollers

Limitations:

- Mechanical wear (turbine type)
- Accuracy affected by fluid impurities

1.3 Temperature Sensor

Purpose:

Measures fluid temperature inside the pipeline.

Why it is important:

- Abnormal temperature → **Chemical reaction issue**
- Overheating → **Friction or equipment failure**

Common Temperature Sensors:

- LM35
- DS18B20
- PT100 / RTD (industrial)

Advantages:

- Simple and reliable
- Low cost

Limitations:

- Slower response in some sensors
- Requires proper insulation

2.Controllers Used in the Project

Controllers collect data from sensors, process it, and send it to the AI system.

2.1 Arduino (Arduino Uno / Mega)

Role in the project:

- Reads sensor data
- Performs basic preprocessing
- Sends data to computer or cloud

Why Arduino is used:

- Easy programming
- Large community support
- Ideal for prototyping

Advantages:

- Low cost
- Simple interfacing
- Good for beginners

Limitations:

- Limited processing power
- No built-in Wi-Fi (Uno)

2.2 ESP32 (Recommended Controller)

Role in the project:

- Reads sensor data
- Sends real-time data via Wi-Fi / Bluetooth
- Connects directly to cloud or server for AI analysis

Why ESP32 is better:

- Built-in Wi-Fi & Bluetooth
- Faster processing
- Suitable for IoT applications

Advantages:

- Low power consumption
- High speed
- Supports real-time monitoring

Limitations:

- Slightly complex programming compared to Arduino

Sensor placement and select sensors&ESp32 for this project

1. Sensor Placement

1.1 Pressure Sensors (most critical)

P1 – Injection Station (Pump Discharge Pressure)

Placement:

- Immediately after the centrifugal pump and before the check valve

P2 – After Intermediate Pump Station

Placement:

- After the second centrifugal pump (pump station discharge)

P3 – Delivery Station (Before Control Valve)

Placement:

- Before control valve, upstream of delivery tank

1.2 Flow Sensors

F1 – Injection Station Flow Meter

Placement:

- After discharge isolation valve

F2 – Delivery Station Flow Meter

Placement:

- At delivery station

1.3 Temperature Sensors

T1 – Main Transmission Pipeline

Placement:

- On pipeline surface (midway between stations)

2.Sensor Selection

Pressure Sensor-MPX series or MPRLS (I2C)

Flow Sensors-YF-S401

Temperature Sensor-DS18B20(waterproof)

3.ESP32 Selection

Selected Controller- ESP32 DevKit V1

Data Flow

Sensors → ESP32 → Feature Extraction → AI Model → Early Warning Alert

AI-Based Pipeline Error Detection Data Recording, Cleaning & Labeling, and ML Training Plan

1. Purpose and Scope

This document describes the planned workflow to (1) record sensor data using ESP32, (2) clean and label the collected dataset, and (3) train a machine learning (ML) classification model to detect pipeline operating conditions such as Normal, Leak, and Blockage. At the current stage, the workflow is defined theoretically. Once sensors and data are available, the same steps will be executed and validated with real measurements.

2. System Overview

Controller: ESP32 Devkit V1 (data acquisition, pre-processing, and communication).
Sensors (selected for prototype):
• Pressure: MPX series (analog) OR MPRLS (I2C) (final selection will depend on availability/budget)
• Flow: Hall-effect flow meter(s) (F1 at Injection, F2 at Delivery)
• Temperature: DS18B20 (T1 on pipeline surface)
Optional: Vibration (MPU6050) for pump health.

3. Data Collection Plan

3.1 Signals to Record

The following raw sensor signals will be recorded at each sample:

Signal	Symbol	Unit	Notes / Purpose
Pressure (Injection pump discharge)	P1	kPa or bar	Detect pump performance; upstream reference
Pressure (after pump station / midline)	P2	kPa or bar	Detect pressure loss between stations
Pressure (delivery side before control valve)	P3	kPa or bar	Detect downstream pressure changes
Flow (Injection	F1	L/min	Inlet flow reference

station)

Flow (Delivery station)	F2	L/min	Outlet flow; compare with F1 for leak indication
Temperature (pipeline surface)	T1	°C	Compensation for pressure/flow variation

3.2 Engineered Features (Computed Columns)

In addition to raw signals, the following derived features will be computed because they improve fault separability for ML:

Feature	Formula	Why it matters
Pressure drop (segment 1)	dP12 = P1 - P2	Helps detect leak/blockage between Injection and Pump Station
Pressure drop (segment 2)	dP23 = P2 - P3	Helps detect downstream restriction or leak
Flow imbalance	dF = F1 - F2	If sustained and above tolerance, indicates possible leak

3.3 Sampling Rate and Logging Duration

Recommended sampling rate for prototype: 5 Hz (one sample every 200 ms) for pressure/flow and 1 Hz for temperature. For simplicity, the ESP32 can log all signals at 5 Hz.

Minimum recommended durations per condition:
• Normal: 20–30 minutes (collect more variety: different valve openings and flow rates)
• Leak: 15–20 minutes (small controlled leak)
• Blockage: 15–20 minutes (partial valve closure)
Optional: Pump fault: 10–15 minutes (reduced voltage or induced disturbance)

3.4 File Naming and Run Management

Each recording session must be stored as a separate CSV file to avoid mixing conditions.
Example filenames:
• run_normal_01.csv
• run_leak_01.csv
• run_blockage_01.csv

Keep sensor placement, sampling rate, and pipe configuration unchanged across all runs.

4. Dataset Format (CSV Schema)

The ESP32 will transmit data in CSV format (Serial USB or Wi-Fi). The schema is:

timestamp_ms,P1,P2,P3,F1,F2,T1

After feature computation and labeling, the final dataset will include additional columns:

timestamp_ms,P1,P2,P3,F1,F2,T1,dP12,dP23,dF,label,source_file

5. Data Cleaning Procedure

5.1 Typical Issues

Common data problems in sensor systems:
- Missing samples (communication drop)
- Spikes/outliers (electrical noise)
- Sensor drift (slow offset change)
- Unit mismatch or scaling errors (calibration not applied)

5.2 Cleaning Steps (Planned)

- Convert all columns to numeric; remove non-numeric rows.
- Drop rows with missing critical values (P1–P3, F1–F2, T1).
- Apply basic plausibility filters (e.g., flows must be >= 0; temperature within a realistic range).
- Apply light smoothing (e.g., moving average over 3 samples) to reduce noise without hiding faults.
- Optional: percentile-based outlier removal (e.g., clip outside 0.1%–99.9%) if spikes remain.
- Standardize units (kPa/bar, L/min) and apply calibration constants (especially for analog MPX).

6. Data Labeling Strategy

Classification requires that each sample is assigned a class label. Labels are defined as:

Condition	Label	How it is created in the test rig
Normal	0	All valves in normal position; no leak; steady pump operation
Leak	1	Open leak valve slightly to

		create controlled fluid loss
Blockage	2	Partially close a valve to restrict flow and increase upstream pressure
Pump fault (optional)	3	Reduce pump voltage or introduce disturbance; monitor vibration/current

Labeling method: since each run is recorded under a single known condition, the label will be assigned automatically based on the filename (e.g., files containing 'leak' are assigned label 1).

7. ML Model Training Plan (Classification)

7.1 Baseline Model Selection

Recommended first model: Random Forest Classifier.
Reasons: strong performance on tabular sensor features, robust to noise, easy to explain in viva, and works well with limited dataset size.

7.2 Training Workflow

1. Load all labeled CSV runs and merge into a single dataset.
2. Compute engineered features (dP12, dP23, dF).
3. Split dataset into Train/Test sets (80/20) with stratification (keep class balance).
4. Train the classifier on the training set.
5. Evaluate using accuracy, precision, recall, F1-score, and confusion matrix.
6. Save the trained model to a file for real-time inference (e.g., .joblib/.pkl).

7.3 Evaluation Outputs to Include in Report

The report should include:
• Confusion matrix figure
• Classification report (precision/recall/F1 per class)
• Short discussion of common misclassifications (e.g., leak vs normal during transients)
• Feature importance (optional) to show which signals most influence decisions

8. Real-Time Prediction Concept

During live operation, ESP32 streams sensor readings. A PC (or edge device) loads the saved model and predicts the current condition in real time. To reduce false alarms, the system should trigger an alert only if the predicted fault persists for multiple consecutive windows (e.g., 5–10 samples).

9. Acceptance Criteria (What “Success” Looks Like)

Minimum acceptance targets for the prototype:

- Stable logging at 5 Hz without frequent missing rows.
- At least 3 runs per class (normal/leak/blockage) collected on different days or settings.
- Model achieves ≥ 85% overall accuracy on the held-out test set (initial target).
- False alarm rate acceptable during steady normal operation (e.g., < 1 false alarm per 10 minutes).

10. Notes for Hardware Implementation Later

If MPX analog pressure sensors are used, include ADC filtering and calibration. If MPRLS (I2C) sensors are used, data will be cleaner and calibration is easier. Regardless of sensor type, keep wiring short, use stable 3.3 V supply, and ensure good grounding.