

CSIS 4495 002 Applied Research Project

**Topic: Visual Drilling Operations & Predictive Maintenance
System**

Student Name: Uchechi Sebastian

Student ID: 300393092

Course: CSIS 4495 002 Applied Research Project

Section: 002

Team Lead: Uchechi Sebastian (Individual Project)

Table of Contents

Introduction.....	4
1.1 Problem Statement.....	4
1.2 Literature and Related Work	5
A. Real-time drilling data visualization	5
B. Wellbore schematics and trajectory visualization	5
C. Drilling optimization and predictive analytics	5
E. Non-productive time (NPT) analysis.....	6
1.3 Knowledge Gaps and Limitations.....	6
Gap 1 — Lack of depth-based event visualization	6
Gap 2 — Limited integration of drilling events with maintenance insights	6
Gap 3 — Lack of explainability in predictive systems	6
Gap 4 — Fragmentation across tools	6
Gap 5 — Limited accessibility and reusability	7
1.4 Hypotheses	7
1.5 Assumptions	7
1.6 Expected Benefits	8
Proposed Research Project.....	8
2.1 Research Objectives	8
2.2 Methodology.....	8
1. Data Preparation:.....	8
2. Backend Development:.....	9
3. Frontend Development:	9
4. Analytics and Explainability:.....	9
5. Reporting and Evaluation:.....	9
2.3 Data Collection	9
2.4 Sample Size	9
2.5 Data Analysis Techniques.....	10
2.6 Research Procedures	10

2.6 Technologies to be Used:	10
2.7 Expected Results	11
Project Planning and Timeline	11
3.1 Project Timeline	11
3.2 Gantt chart – Project timeline.....	12
Project Contract	13
4.1 Contract	13
AI Use Section	13
5.1 AI use Table.....	13
Work Date / Hours Log	14
6.1 Log table	14
Closing and References	15
7.1 Closing	15
7.2 References:	16
Literature references	16
Appendix A: Simulated Drilling Dataset Structure	17
A.1 Well Information Table.....	17
A.2 Drilling Operations Table	17
A.3 Drilling Events Table	18
A.4 Equipment Involvement Table	18
A.5 Data Characteristics	18
A.6 Purpose of the Dataset.....	19
Appendix B: AI Tool Usage and Prompt History	19
B.1 From ChatGpt.....	19
B.2 From Copilot	54

Introduction

Drilling operations in the oil and gas industry are complex, data-intensive processes that generate large volumes of operational information throughout the lifecycle of a well. During drilling, data is continuously recorded through daily drilling reports, operation logs, and event summaries, capturing information such as depth progression, operational activities, downtime events, and equipment involvement. This data plays a critical role in post-operation analysis, operational learning, and planning for future drilling campaigns.

Despite the availability of such data, interpreting drilling operations remains a challenging task. Drilling engineers often rely on textual reports, spreadsheets, and static dashboards to review what occurred during drilling. These representations can make it difficult to quickly understand how operational problems relate to specific depth intervals, how issues evolve, and how repeated events contribute to increasing operational risk. As a result, valuable insights into non-productive time, recurring operational issues, and potential maintenance concerns may not be immediately visible.

Within this context, there is a growing need for software tools that not only display drilling data but also help interpret operational patterns and provide explainable insights that support decision-making during and after drilling operations.

1.1 Problem Statement

Drilling operations produce detailed operational records that describe what occurred during the drilling process; however, transforming these records into an actionable understanding remains a significant challenge. Engineers must interpret large volumes of operational data to determine where drilling problems occurred, how these problems evolved, and what operational behaviors may have contributed to increasing risk or non-productive time. When this interpretation is inefficient or unclear, valuable lessons from drilling operations may be lost.

This project is framed around the following research questions:

1. How can drilling operation data be visually organized to clearly show where operational problems occur along well depth?

2. How can recorded operational events be analyzed to identify emerging patterns that indicate increasing operational or maintenance risk?
3. How can analytics be presented in an explainable way that supports engineering judgment rather than replacing it?

These questions are important because current drilling analysis tools often emphasize data reporting rather than interpretation, making it difficult to quickly recognize patterns and understand their implications. Addressing these questions can improve post-drilling analysis, support operational learning, and contribute to more informed decision-making in future drilling operations.

1.2 Literature and Related Work

A. Real-time drilling data visualization

Commercial platforms such as Petrolink and Halliburton HalVue focus on real-time monitoring, providing dashboards for torque, weight on bit, ROP, and rig activity. These systems emphasize operational awareness and data streaming but do not provide depth-based event visualization or post-operation analysis.

B. Wellbore schematics and trajectory visualization

Tools like wellVizion offer wellbore schematics, trajectory plots, and anti-collision visualization. These systems are valuable for planning and directional drilling, but do not integrate operational events (e.g., stuck pipe, reaming, NPT) along depth.

C. Drilling optimization and predictive analytics

Platforms such as Corva and SLB DrillOps incorporate predictive analytics, real-time optimization, and automated alerts. Their focus is on improving drilling efficiency during operations using machine learning models. However, these systems are typically enterprise-scale, proprietary, and optimized for real-time rig data rather than post-run event analysis.

E. Non-productive time (NPT) analysis

NPT is widely studied as a key performance indicator in drilling operations. Research focuses on the classification of NPT events, root-cause analysis, and cost impact. However, NPT is typically analyzed in tabular or report form, not through interactive depth-based visualization.

Overall, the literature shows strong development in real-time monitoring, trajectory visualization, and predictive analytics - but these areas remain siloed and do not converge into a unified, explainable, depth-based decision-support tool.

1.3 Knowledge Gaps and Limitations

Despite the availability of advanced drilling software, several gaps remain unaddressed:

Gap 1 — Lack of depth-based event visualization

No major tool provides a simple, interactive vertical wellbore view where drilling events are mapped as **color-coded segments along depth**. Existing systems focus on real-time curves, not post-operation event visualization.

Gap 2 — Limited integration of drilling events with maintenance insights

Current predictive maintenance tools rely on sensor data or machine-learning models but do not incorporate **event history** (e.g., stuck pipe, reaming frequency, NPT clusters) as indicators of equipment stress or operational risk.

Gap 3 — Lack of explainability in predictive systems

Most commercial predictive tools operate as black boxes. Engineers often cannot see **why** a risk score increased.

Your system provides **transparent, explainable logic** tied directly to recorded events.

Gap 4 — Fragmentation across tools

Drilling visualization, KPI analysis, and maintenance prediction exist in separate platforms.

There is no lightweight, unified, web-based tool that combines: Drilling KPIs, depth-based event visualization, Predictive maintenance indicators, and Explainable insights.

Gap 5 — Limited accessibility and reusability

Enterprise tools are expensive, proprietary, and require real-time rig data. This system is:

- Upload-based
- Reusable across wells
- Suitable for post-run analysis
- Built with open technologies

This makes it accessible for training, engineering studies, and smaller operators.

This project addresses these gaps by developing a full-stack, reusable decision-support application that (1) visualizes drilling operations and events along a vertical wellbore (pipe) depth view, (2) computes key KPIs such as NPT and event frequency, (3) provides explainable risk indicators based strictly on recorded operational events, and (4) supports practical workflow needs such as well selection via a location map and report export.

1.4 Hypotheses

1. Depth-based visualization of drilling events will improve engineers' ability to identify operational risks compared to traditional tabular or curve-based reports.
2. Event-driven predictive maintenance indicators can highlight equipment stress earlier than sensor-only approaches.
3. Explainable insights increase user trust and adoption compared to black-box predictive models.
4. A unified, web-based platform will reduce analysis time by consolidating drilling KPIs, events, and maintenance indicators in one interface.

1.5 Assumptions

1. Drilling event data (e.g., stuck pipe, reaming, NPT) is available in structured or semi-structured form.
2. Depth information is recorded consistently across drilling operations.
3. Engineers benefit from visual, depth-oriented representations of operational events.
4. Predictive maintenance can be meaningfully informed by event frequency, severity, and clustering.
5. The system is intended for post-operation analysis, not real-time rig control.

1.6 Expected Benefits

1. Faster identification of operational issues through intuitive depth-based visualization.
2. Improved maintenance planning by linking drilling events to equipment stress indicators.
3. Better decision-making through integrated KPIs, event history, and risk scoring.
4. Higher transparency due to explainable logic behind every insight.
5. Reusability across wells and campaigns, supporting continuous improvement.
6. A modern, full-stack engineering tool demonstrating strong technical and analytical capability.

Proposed Research Project

2.1 Research Objectives

The main objectives of this project are to:

1. Design a depth-based visualization that clearly represents drilling operations and events along the wellbore.
2. Develop analytics to compute key drilling performance indicators, such as non-productive time (NPT), washout, and event frequency.
3. Implement predictive maintenance indicators that highlight increasing equipment risk based on recorded operational behavior.
4. Provide explainable insights that help users understand why operational risks are flagged.
5. Deliver a reusable, full-stack web application that supports drilling data interpretation and reporting.

2.2 Methodology

The methodology consists of the following steps:

1. Data Preparation:

Simulated or historical drilling datasets will be structured into standardized formats representing wells, drilling operations, events, and equipment involvement.

2. Backend Development:

A backend service will be implemented using Python and FastAPI to manage data ingestion, storage, and analytics. The backend will compute KPIs, identify recurring patterns, and generate risk indicators based on predefined rules and statistical trends.

3. Frontend Development:

A React-based user interface will be developed to support well selection, depth-based visualization, KPI display, and interaction with drilling events. A vertical wellbore (pipe) visualization will be used to map operational data directly to depth.

4. Analytics and Explainability:

Rule-based and basic statistical methods will be applied to identify emerging operational patterns and maintenance risk. All outputs will be explainable, with clear links to the underlying operational events and data.

5. Reporting and Evaluation:

The system will support the generation of downloadable summary reports. The prototype will be evaluated based on usability, clarity of visualization, and its ability to support the interpretation of drilling operations.

2.3 Data Collection

The system will use **simulated or historical drilling datasets** consisting of:

Well identifiers and total depth, Depth intervals, Drilling operation types, Recorded drilling events (e.g., stuck pipe, downtime), Non-productive time (NPT), and equipment involvement in events.

No real-time sensors or proprietary data will be required.

2.4 Sample Size

The sample dataset will include:

- **Multiple wells** (e.g., 1–10 wells)
- Each well contains **several hundred depth intervals**.
- Dozens of recorded operational events per well

2.5 Data Analysis Techniques

The following data analysis techniques will be applied:

- **Descriptive analytics:**

Aggregation of drilling events and non-productive time by depth and operation type.

- **KPI computation:**

Calculation of key indicators such as total NPT, event frequency, and drilling efficiency metrics.

- **Pattern identification:**

Rule-based and basic statistical analysis to identify recurring events, clustering of issues within depth intervals, and increasing operational or maintenance risk.

- **Explainable logic:**

All identified risks and alerts will be linked directly to recorded operational data, ensuring transparency and interpretability.

2.6 Research Procedures

The research procedures will follow these steps:

- 1 Generate and prepare simulated drilling datasets based on realistic operational scenarios.
- 2 Ingest and store data using a backend service implemented in Python.
- 3 Apply analytics to compute KPIs and identify operational patterns.
- 4 Visualize results using a depth-based wellbore interface.
- 5 Generate downloadable summary reports based on system outputs.
- 6 Evaluate the system based on usability, clarity of visualization, and interpretability of analytics.

2.6 Technologies to be Used:

- **Operating System / Platform:** Web-based application
- **Programming Languages:** Python, JavaScript
- **Backend Framework:** FastAPI
- **Frontend Framework:** React.js
- **Database:** PostgreSQL or MySQL

- **Visualization:** SVG / D3.js
- **Deployment (Optional):** Docker

2.7 Expected Results

The expected outcome is a **fully functional web application** that:

- Allows users to select wells via a location-based interface.
- Visualizes drilling events along a vertical wellbore.
- Displays drilling KPIs and high-risk zones.
- Provides explainable insights into operational issues.
- Generates downloadable drilling summary reports.

The system is expected to demonstrate practical applicability in the interpretation of drilling data and operational review.

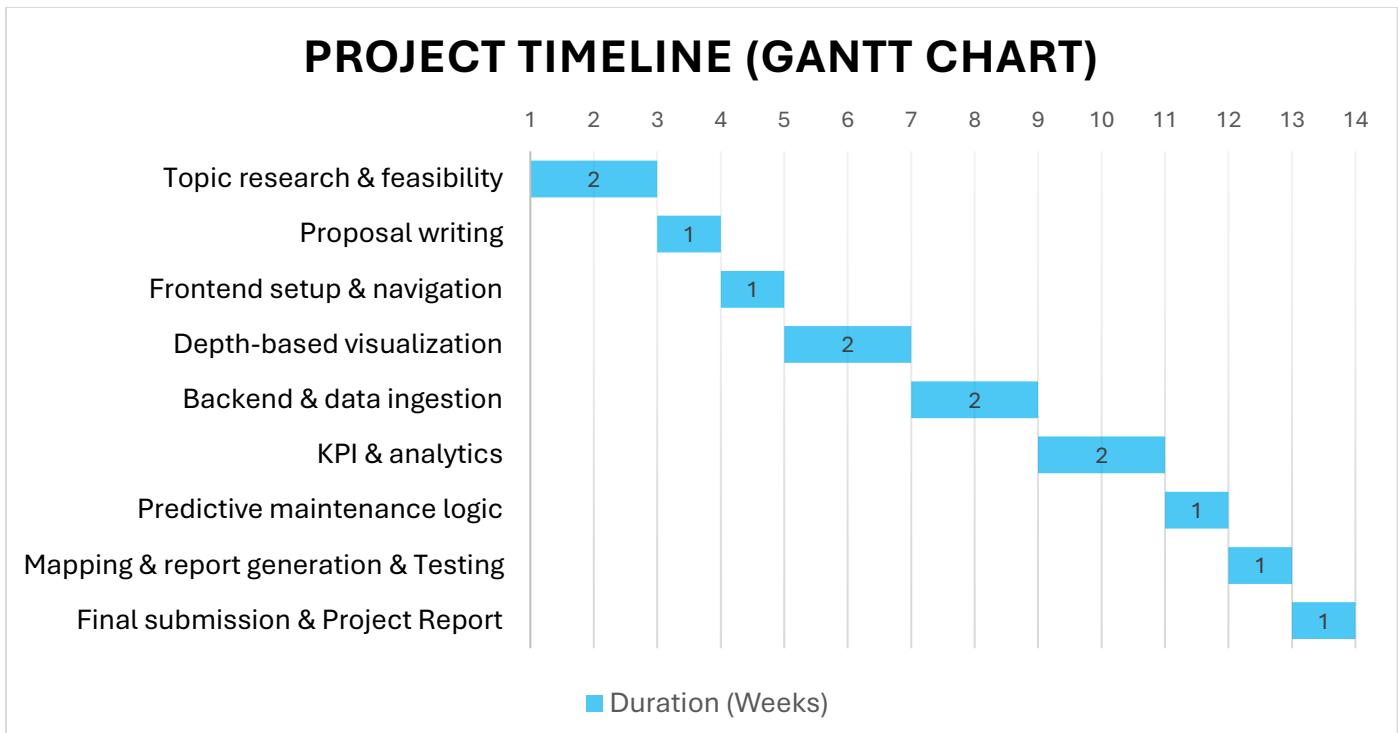
Project Planning and Timeline

3.1 Project Timeline

Phase	Week(s)	Milestones	Deliverables
Phase 1: Topic Exploration & Domain Research	Weeks 1 - 2	Explored potential project domains Reviewed drilling operations context Identified problem area and feasibility	Project idea selection Initial domain understanding
Phase 2: Proposal Development & Planning	Week 3	Defined project scope and objectives Identified data requirements Prepared a formal project proposal	Submitted project proposal Finalized research questions and methodology
Phase 3: Frontend Development (Core UI)	Week 4	Implement the main application layout Build well selection and navigation views	Initial React frontend Well selection interface
Phase 4: Depth-Based Visualization	Weeks 5–6	Implement vertical wellbore (pipe) visualization Map drilling events to depth intervals	Interactive depth-based visualization
Phase 5: Backend Development & Data Ingestion	Weeks 7–8	Design database schema Implement APIs for data upload and retrieval	Functional backend (FastAPI) Data ingestion endpoints

Phase 6: KPI Computation & Analytics	Weeks 9–10	Compute NPT and event-based KPIs Aggregate data by depth and operation	KPI analytics module
Phase 7: Predictive Maintenance Logic	Weeks 11	Implement rule-based risk indicators Link risk to recorded events and equipment	Predictive maintenance indicators with explanations
Phase 8: Mapping & Report Generation & Testing	Weeks 12	Implement a well location map view Enable downloadable drilling summary reports. Test system functionality.	Location-based well view Report export feature Tested application
Phase 9: Final documentation	Week 13	Prepare final documentation	Final project report

3.2 Gantt chart – Project timeline



Project Contract

4.1 Contract

As this is an individual project, I agree to adhere to the scope, milestones, and timeline outlined in this proposal.

Name: Uchechi Sebastian

Date: January 26, 2026.

Signed By: Uchechi Sebastian

Signature:



AI Use Section

5.1 AI use Table

AI Tool	Version / Account	Specific Use	Value Added
ChatGPT	GPT-5.2	Assisted with refining the project idea, structuring the proposal, improving academic writing, and clarifying the research design, methodology, and timeline.	Helped organize thoughts clearly, improve academic clarity, and ensure the project scope and structure align with the course.
Copilot	Free	Summary of Relevant Literature and Existing Research	Helped to clearly identify the existing research on the project and showed how the project differs from various working projects.

Work Date / Hours Log

6.1 Log table

Date	Hours	Description
17-Jan-26	3.5	I explored and researched possible project ideas for the course, focusing on areas related to drilling operations and data analytics. I evaluated whether the project idea was feasible to implement within the course timeline and researched whether similar work had been done before. This helped me narrow down and decide on a suitable project that aligns with my background and course requirements.
20-Jan-26	3	I started learning React by reviewing the basics of components, JSX, and how a React application is structured. I focused on understanding how React will be used to build the frontend of my applied project.
21-Jan-26	3.5	I worked on Introduction of the project: Improving the introduction, stating the problem statement clearly, outlining what questions needs to be answered. I searched for literature and research work based on my project to find similar research on my project. I searched for gaps within this researched project work.
22-Jan-26	2.5	I continued literature research related to drilling operations, non-productive time, and decision support systems. I reviewed how similar systems visualize operational data and noted limitations in existing approaches, particularly depth-based visualization. I worked on my reference and included it in my project report after the research I did.
23-Jan-26	3	I worked on defining the project methodology, including data sources, data collection approach, and analysis techniques. I also clarified how simulated drilling data would be used and documented the justification for using non-proprietary datasets in the project.

25-Jan-2026	1	I created the project timeline for the complete project for the entire semester.
26-Jan-2026	3.5	<p>I drew the Gantt chart for the project timeline and included it in my proposal report.</p> <p>I prepared the appendix.</p> <p>I went through the entire report and did a lot of corrections and restructuring to make sure the proposal fits what my project is all about.</p>

Closing and References

7.1 Closing

This project aims to demonstrate the practical application of full-stack development and data analytics in an engineering context. It emphasizes visualization, explainability, and decision support rather than automated prediction.

This project addresses these gaps by developing a full-stack, reusable decision-support application that (1) visualizes drilling operations and events along a vertical wellbore (pipe) depth view, (2) computes key KPIs such as NPT and event frequency, (3) provides explainable risk indicators based strictly on recorded operational events, and (4) supports practical workflow needs such as well selection via a location map and report export.

7.2 References:

Literature references

WellVizion - Well schematics diagramming solution. (n.d.). <https://wellvizon.com/>

Admin. (n.d.). *PetrolInk: Data Visualization in Drilling Operations – Petrolink.*
<https://www.petrolink.com/data-visualization-in-drilling/>

HalVue™ real-time viewer. (n.d.). Halliburton. <https://www.halliburton.com/en/products/halvue-real-time-viewer>

Barakat, M., Ela, M. a. E., & Khalaf, F. (2021). Integrating risk management concepts into the drilling non-productive time. *Journal of Petroleum Exploration and Production Technology*, 11(2), 887–900. <https://doi.org/10.1007/s13202-020-01059-0>

Liu, S., Feng, Y., Wang, X., & Yan, P. (2021). Cross-Platform Drilling 3D Visualization system based on WebGL. *Mathematical Problems in Engineering*, 2021, 1–18.
<https://doi.org/10.1155/2021/5516278>

Corva.Ai. (n.d.). *AI Predictive Drilling Automation Software | Corva.* Corva.
<https://www.corva.ai/energy/predictive-drilling>

Appendix A: Simulated Drilling Dataset Structure

This appendix describes the structure of the simulated drilling dataset used in this project. The dataset is designed to reflect the format and level of detail typically found in standard drilling operation logs and reports, without relying on proprietary or real-time field data. The simulated data support visualization, KPI computation, and rule-based predictive maintenance analysis.

A.1 Well Information Table

Field Name	Description
well_id	Unique identifier for each well
well_name	Descriptive name of the well
location	General geographic location of the well
total_depth	Total measured depth of the well (meters)
spud_date	Date drilling commenced
well_status	Current status of the well (e.g., drilling, completed)

A.2 Drilling Operations Table

Field Name	Description
operation_id	Unique identifier for each drilling operation
well_id	Reference to the associated well
depth_from	Start depth of the operation interval (meters)
depth_to	End depth of the operation interval (meters)
operation_type	Type of operation (e.g., drilling, reaming, circulation)
start_time	Operation start timestamp
end_time	Operation end timestamp
duration_hours	Total duration of the operation

Field Name	Description
npt_hours	Non-productive time associated with the operation

A.3 Drilling Events Table

Field Name	Description
event_id	Unique identifier for each event
operation_id	Reference to the related drilling operation
well_id	Reference to the associated well
depth	Depth at which the event occurred (meters)
event_type	Type of event (e.g., stuck pipe, downtime, circulation loss)
event_description	Short description of the event
event_duration_hours	Duration of the event

A.4 Equipment Involvement Table

Field Name	Description
equipment_id	Unique identifier for equipment
equipment_type	Type of equipment (e.g., drill string, mud pump)
event_id	Reference to the related drilling event
usage_intensity	Qualitative level of stress or usage (low, medium, high)
cumulative_hours	Total operational hours associated with the equipment

A.5 Data Characteristics

- Data is **simulated** based on realistic drilling operation scenarios.
- No proprietary or confidential information is included.
- Depth is recorded as **measured depth** in meters.
- The dataset supports aggregation by depth, operation type, and equipment involvement.

- The structure allows reuse across multiple wells and drilling campaigns.

A.6 Purpose of the Dataset

The simulated dataset enables:

- Depth-based visualization of drilling operations and events
- Computation of drilling performance KPIs (e.g., non-productive time)
- Identification of recurring operational patterns
- Explainable predictive maintenance risk indicators based on operational history

Appendix B: AI Tool Usage and Prompt History

This appendix documents the use of AI tools in the development of this project,

including prompt history and the specific purposes for which AI assistance was used.

B.1 From ChatGpt

hmmmm what about drilling?

Ah — **drilling** is actually a *very* good direction 🌟

It's technical, high-value, and **much less “regular”** than dashboards for production or maintenance.

Let's slow down and do this properly.



FIRST: WHAT “DRILLING” MEANS FOR A SOFTWARE PROJECT

You are **NOT** building:

- ✗ a drilling simulator
- ✗ a physics-heavy wellbore model
- ✗ real-time rig control software

Those are unrealistic for a course.

You **ARE** building:

- a **decision-support system**
- based on **drilling operations data**
- that helps engineers **plan, monitor, and learn**

This is exactly where software + analytics shine.

DRILLING PROBLEMS THAT ARE UNDER-SERVED (NOT REGULAR)

Most drilling software is:

- Very expensive
- Vendor-locked
- Over-engineered
- Not explainable

We'll target **what people actually struggle with**.

OPTION 1: Drilling Event Intelligence System (VERY STRONG)

Core idea

A system that **records, classifies, and learns from drilling events**.

It answers:

“What went wrong, how often, and under what conditions?”

Data collected (manual or uploaded)

- Depth
- Time

- Operation type (drilling, tripping, casing)
 - Event type:
 - Stuck pipe
 - Lost circulation
 - Washout
 - Torque & drag increase
 - Action taken
 - Outcome
-

Analytics

- Event frequency by depth
 - Time lost per event
 - Repeated problem zones
 - Trend across wells
-

Output

- “High stuck-pipe risk between 2,100–2,300 m”
- “Lost circulation events increased 30% in this formation”

This is **VERY useful** and **rarely done well**.

OPTION 2: Non-Productive Time (NPT) Analyzer (EXCELLENT)

Core idea

A tool focused purely on **where time and money are wasted** during drilling.

What it tracks

- Planned vs actual time
 - NPT causes
 - Duration per incident
 - Recurring delays
-

KPIs

- NPT %
 - Cost of NPT
 - Top 5 delay causes
 - Preventable vs unavoidable
-

Why it's powerful

Everyone talks about NPT.

Very few teams **systematically learn from it**.

❸ OPTION 3: Pre-Drill Risk Planner (VERY IMPRESSIVE)

Core idea

Before drilling a well, the system answers:

“Based on past wells, what risks should we expect?”

Input

- Well depth
 - Formation type
 - Past offset wells
-

Output

- Expected problem zones
- Likely events
- Mitigation checklist

This is **planning intelligence**, not real-time control.

OPTION 4: Explainable “Why We Got Stuck” System (UNIQUE)

Core idea

Instead of just logging stuck-pipe events, the system explains:

“Why did this happen *this time*? ”

Logic combines

- Depth
 - Operation
 - Mud condition (qualitative)
 - Past similar events
-

Output

“Similar stuck-pipe events occurred during tripping in shale at similar depth.”

This is **explainable drilling intelligence**.

MY TOP RECOMMENDATION FOR YOU

Based on:

- Your petroleum background
- Desire for “not regular”
- Full-stack + analytics
- Academic acceptability

Option 1 + Option 2 combined

Drilling Event & NPT Intelligence Platform

This is:

- Rare
 - Extremely realistic
 - Valuable
 - Not sensor-dependent
 - Easy to explain
-

WHAT THIS LOOKS LIKE AS SOFTWARE

Backend (Python / FastAPI)

- Event logging APIs

- NPT calculation engine
- Risk pattern detection

Frontend (React)

- Timeline of drilling events
- NPT breakdown charts
- Risk zones by depth

Analytics

- Event frequency analysis
 - Root-cause patterns
 - Explainable summaries
-

💡 STRONG PROJECT TITLES

Choose one style:

- 1 Drilling Event Intelligence and NPT Analysis Platform**
- 2 Explainable Drilling Risk & Non-Productive Time Analytics System**
- 3 A Decision-Support Tool for Learning from Drilling Events**

All are **very solid**.

Tell me what excites you more:

- A Drilling events & stuck-pipe problems
- B NPT & time/cost loss
- C Pre-drill risk planning
- D Combination of A + B

Once you choose, I'll:

- Lock the scope
- Define data fields
- Design KPIs
- Make sure it fits Riipen + your course perfectly.

good let us also add the predictive maintenance to it

Excellent — that's a **very strong and coherent idea now**.

What you're building is no longer “just drilling analytics”; it becomes a **decision-support system for drilling operations + predictive maintenance**.

Let me explain this **very clearly and realistically**, so you can confidently defend it to your lecturer.

FINAL PROJECT CONCEPT (LOCKED & CLEAN)

Drilling Event, NPT, and Predictive Maintenance Intelligence Platform

A full-stack software system that analyzes drilling events and non-productive time to predict maintenance needs and operational risks for drilling equipment.

This is **not regular, not overdone, and very industry-relevant**.

IMPORTANT CLARIFICATION (VERY IMPORTANT)

Predictive maintenance in DRILLING does NOT mean:

- ✗ Predicting when a drill bit will break exactly
- ✗ Real-time rig control
- ✗ Sensor-heavy ML models

Predictive maintenance in DRILLING DOES mean:

- ✓ Predicting **increased risk of equipment issues**
- ✓ Identifying **patterns that precede failures**
- ✓ Recommending **inspection or maintenance before problems escalate**

This is exactly what drilling engineers actually do — just manually today.

❖ WHAT EQUIPMENT ARE WE “PREDICTING” MAINTENANCE FOR?

You are NOT predicting maintenance for everything.

You focus on **critical drilling equipment**, for example:

- Drill string (DP, HWDP)
- Drill bit
- Mud motor
- Top drive (conceptually)
- Pumps (conceptually)

You don't need sensors — you use **events and time loss** as signals.

⌚ HOW DRILLING EVENTS CONNECT TO PREDICTIVE MAINTENANCE

This is the key logic 

1 Drilling Events = Early Warning Signals

Examples:

- Repeated torque & drag issues
- Multiple stuck-pipe incidents
- Frequent washouts

- Recurrent bit dysfunction
- Excessive reaming time

📌 These almost always indicate **equipment wear or operational stress**.

2 NPT = Symptom of Degrading Equipment

If:

- NPT related to same cause keeps increasing
- Time to resolve incidents is rising



This suggests:

- Equipment condition worsening
 - Maintenance overdue
 - Design or operational issue
-

3 Predictive Maintenance Logic (Simple & Explainable)

Your system does this:

“Based on past drilling behavior, the likelihood of equipment-related issues is increasing.”

Not:

“The motor will fail in 6.2 hours.”

That distinction makes your project **realistic and defensible**.

🧠 PREDICTIVE MAINTENANCE FLOW (STEP-BY-STEP)

Step 1: Collect Drilling Data

- Depth
 - Operation type
 - Event type
 - Event duration
 - NPT cause
 - Equipment involved
-

Step 2: Compute Drilling KPIs

Examples:

- NPT %
 - Event frequency per 100 m
 - Average time per incident
 - Repeated event count
-

Step 3: Pattern Detection

The system checks for:

- Increasing frequency
 - Clustering by depth or operation
 - Repeated equipment involvement
-

Step 4: Maintenance Risk Output

Example output:

High maintenance risk – Drill string

Reason: Repeated torque & drag events and increasing reaming time over last 500 m.

This is **predictive maintenance through behavior**.

KPIs THAT SUPPORT PREDICTIVE MAINTENANCE

These are **perfectly acceptable drilling KPIs**:

NPT Percentage

NPT % = NPT hours / Total drilling hours

Event Frequency

Events per 100 m drilled

Repeated Event Index

Number of repeated events / Total events

Equipment Stress Indicator (Derived)

Composite score from:

- Event frequency
 - NPT duration
 - Reoccurrence
-

WHAT THE APP SHOWS (VERY CLEAR)

For Engineers

- Drilling timeline
- Event log

- NPT breakdown
- Maintenance risk indicator

For Decision Support

- “Watch” vs “High risk” equipment
- Suggested actions:
 - Inspect
 - Replace
 - Change operation

✳ WHY THIS IS NOT REGULAR

Most drilling software:

- Logs data
- Shows charts
- Stops there

Your system:

- Learns from events
- Links events to maintenance risk
- Explains *why* risk is increasing

That's **rare**, especially in academic projects.

🏆 HOW YOU EXPLAIN THIS TO YOUR LECTURER (USE THIS)

“The project combines drilling event and non-productive time analysis with predictive maintenance logic. By analyzing recurring drilling problems and time loss patterns, the system

predicts increased maintenance risk for drilling equipment and provides explainable recommendations to support proactive decision-making.”

That explanation is **very strong**.

DOES THIS FIT RIIPEN & YOUR COURSE?

- ✓ Python backend (FastAPI)
- ✓ React frontend
- ✓ Database
- ✓ Analytics
- ✓ Explainable logic
- ✓ Reusable across wells & rigs

This is a **very valid concern**, and I’m really glad you said it out loud.

What you’re feeling is exactly the difference between “**a school project**” and “**a product**”.

Let me be very clear first:

- ✗ **This is NOT a write-up project**
- ✗ **It is NOT just “AI writes predictions”**
- ✗ **It is NOT boring tables and text**

If it were, I would *not* recommend it to you.

What you’re building can be **visual, interactive, decision-driven, and impressive** — just not “cute” like a breathing app. It’s **powerful** instead.

my issues is what is it going to look like, what is interesting about it. is it just going to be a write up or i dont get. for example, my friend created an app that helps with breathing, or exercises or something. now i dont just want write up. i want something extra ordinary will the software just write predictions, thats so boring

This is a **very valid concern**, and I'm really glad you said it out loud.

What you're feeling is exactly the difference between “**a school project**” and “**a product**”.

🔥 WHAT MAKES THIS EXTRAORDINARY (NOT BORING)

Your friend's app:

- Helps *one person*
- Does one thing (breathing/exercise)
- Is emotional / wellness-based

Your app:

- Helps **engineering teams**
- Supports **million-dollar drilling decisions**
- Prevents **real operational losses**

Different kind of “wow”.

🎮 WHAT IT ACTUALLY LOOKS LIKE (VISUALLY)

Let me **paint the picture**, screen by screen.

⌚ 1. DRILLING OPERATIONS DASHBOARD (NOT TEXT)

When the app opens, you see:

- ◆ **A WELL TIMELINE (THIS IS KEY)**

A **vertical depth bar** like this:

0 m

| Normal drilling

|

| △ Torque & drag

|

| ● Stuck pipe (NPT)

|

| △ Reaming

|

|

1500 m

👉 You scroll through depth

👉 You click events

👉 You see problems visually

That alone is already cooler than a report.

📊 2. LIVE KPI CARDS (ENGINEERING STYLE)

At the top:

- 🔪 NPT %
- ⏱ Avg event resolution time
- 📈 Event frequency
- 🛠 Equipment risk score

These **change dynamically** when:

- You filter depth
- You filter operation
- You simulate “what if” scenarios

This is **interactive**, not static.

3. PREDICTIVE MAINTENANCE — BUT VISUAL

Instead of boring text like:

“High risk detected”

You show:

EQUIPMENT RISK GAUGE

A circular gauge like:

Drill String Risk

 Low —  Medium —  High



Current

Click it and you see:

-  Events causing risk
-  Trend over depth
-  Time since last inspection

That is **very engaging**.

4. “WHY” PANEL (THIS IS YOUR SUPERPOWER)

Engineers hate black boxes.

So your app shows:

⚠ Risk Increased Because:

- Torque & drag events increased by 42%
- Reaming time doubled in last 300 m
- Similar pattern preceded failures in past wells

This is **explainable AI**, not boring AI.

🎯 5. INTERACTIVE “WHAT-IF” MODE (THIS IS HUGE)

This is where your app becomes **extraordinary**.

Example:

What if we:

- Change bit type?
- Schedule inspection now?
- Continue drilling 200 m more?

The app simulates:

- KPI changes
- Risk changes
- Possible NPT impact

Now you’re not **reading** — you’re **playing with decisions**.

💡 6. DRILLING EVENT PLAYER (LIKE NETFLIX FOR OPERATIONS)

Imagine pressing **Play** :

- The well progresses
- Events appear
- NPT shows up
- Risk rises or falls in real time

This is **very rare in student projects**.

WHERE AI IS ACTUALLY “COOL”

Not text generation.

AI is used to:

- Detect **patterns humans miss**
- Compare with **historical wells**
- Highlight **non-obvious risks**

Example:

“This well looks normal, but its event spacing matches wells that failed later.”

That’s subtle and powerful.

WHY THIS IS NOT BORING AT ALL

Because:

- You **interact** with it
- You **explore** it
- You **see cause → effect**

- You **make decisions**

A report is **dead**.

This is **alive**.

WHY LECTURERS LOVE THIS TYPE OF PROJECT

Because it:

- ✓ Is reusable
 - ✓ Is not student-semester specific
 - ✓ Has frontend + backend + analytics
 - ✓ Demonstrates engineering thinking
 - ✓ Could actually be used in industry
-

IF YOU WANT IT EVEN MORE “WOW”

We can add ONE of these:

-  **Gamified risk scoring**
-  **AI explanations in plain English**
-  **Engineer override & feedback loop**
-  **Side-by-side well comparison**

You don't need all — **one is enough**.

REAL TALK (IMPORTANT)

Your project will **not look like a breathing app**.

It will look like a **professional engineering system**.

That's a *higher level of impressive*.

NEXT QUESTION (VERY IMPORTANT)

Tell me honestly:

Which would excite you more?

- 1 A visual drilling timeline + risk dashboard**
- 2 A what-if decision simulator**
- 3 A Netflix-style drilling replay**
- 4 A gamified engineering risk system**

Pick **one** — we'll design it properly so it's 🔥, not boring.

how will we do the depth, how will we visualize it. id like it to show as a pipe and labelled depth is that possible

Yes — **that is absolutely possible**, and it's actually a **very strong, professional design choice**



You are thinking like an engineer *and* a product designer now.

Let me explain **exactly how depth works, how the pipe visualization works, and how it is built technically**, without hand-waving.

SHORT ANSWER

- ✓ Yes, you can visualize depth as a **vertical pipe / wellbore**
- ✓ Depth is just a **numeric axis** (measured depth or true vertical depth)

- ✓ Each problem is drawn as a **segment on the pipe**
- ✓ This is done with **React + SVG / Canvas / Chart libraries**

This is **very realistic** and used in industry tools.

1. WHAT “DEPTH” REALLY IS (IMPORTANT)

Depth is simply:

depth_m = 0 → 3,500 meters (example)

Each record in your data has:

- depth_start
- depth_end
- event_type

Example:

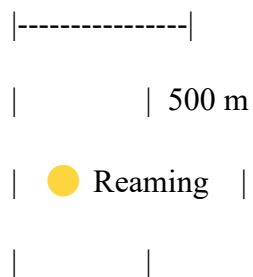
Depth 1250m – 1280m → Stuck pipe

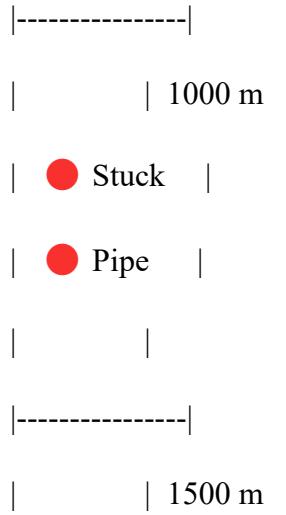
Depth 1420m – 1450m → Reaming

That's all you need.

2. PIPE / WELL VISUALIZATION (CONCEPT)

Imagine this on screen:





Key ideas:

- The pipe is **vertical**
- Depth increases **downwards**
- Events are **colored segments**
- Labels appear on hover or click

This is **100% doable**.

🎨 3. HOW EACH PROBLEM APPEARS ON THE PIPE

Problem	Color	Visual
Normal drilling	Gray / Blue	Clean segment
Torque & drag	Yellow	Warning band
Stuck pipe	Red	Thick red band
Lost circulation	Purple	Dashed band
Maintenance	Green	Solid block

You are not guessing — the **data decides the color**.

4. HOW AI CONNECTS TO THIS VISUAL

AI does NOT draw the pipe.

AI:

- Analyzes patterns
- Assigns **risk scores** per depth interval

Your backend sends:

{

 depth_start: 1240,

 depth_end: 1300,

 risk: "high",

 reason: "Torque trend + NPT increase"

}

Frontend:

- Paints that segment red
- Adds tooltip with explanation

5. INTERACTION (THIS MAKES IT COOL)

When the user:

 **Clicks a depth segment**

They see:

- Events that happened there

- KPIs at that depth
- Risk explanation
- Maintenance history

👉 Scrolls

The pipe scrolls smoothly downwards

👉 Zooms

Depth resolution increases (like Google Maps)

🛠 6. TECHNICALLY — HOW YOU BUILD THIS

You have **3 good options:**

💡 Option 1: React + SVG (Best for control)

- Draw a vertical rectangle (pipe)
- Map depth → pixel position
- Draw colored segments

✓ Clean

✓ Custom

✓ Lecturer-friendly

💡 Option 2: D3.js (Data-driven)

- Perfect for timelines & scales
- Handles zoom & tooltips well

✓ Professional

✗ Slightly steeper learning curve

Option 3: Chart.js / Recharts (Simpler)

- Treat depth as Y-axis
- Fake the “pipe look”

✓ Fast

✗ Less realistic

 I recommend SVG or D3.

7. DEPTH SCALING (VERY IMPORTANT)

You define a scale:

`pixels_per_meter = 0.5`

So:

$0 \text{ m} \rightarrow 0 \text{ px}$

$2000 \text{ m} \rightarrow 1000 \text{ px}$

This is how scrolling works.

8. SAMPLE DATA STRUCTURE (VERY IMPORTANT)

{

 "well_id": "WELL-001",

 "events": [

```
{  
    "depth_start": 1200,  
    "depth_end": 1260,  
    "event_type": "TORQUE_DRAG",  
    "risk": "medium"  
},  
{  
    "depth_start": 1260,  
    "depth_end": 1300,  
    "event_type": "STUCK_PIPE",  
    "risk": "high"  
}  
]  
}
```

This maps **directly** to your pipe visualization.

IMPORTANT REASSURANCE

You are **not expected** to:

- Simulate physics
- Predict exact failures
- Build Schlumberger-level software

You are showing:

“Given drilling data, here is how we **visualize operational risks by depth**.”

That is **more than enough**.

NEXT STEP (CHOICE)

Do you want me to:

- 1** Draw a **wireframe layout** (screen sections)
- 2** Define **exact backend endpoints** for depth data
- 3** Show **how to convert depth → pixels in React**
- 4** Map each drilling problem → visual rule

Pick one — we’ll keep building this properly.

You can copy-paste this into a document or PDF.

so where can i get this public dataset?

This is a **very fair question**, and I’m glad you’re asking it *before* claiming anything in your proposal.

I’ll give you **real, safe places** you can get **public or example drilling datasets**, and also tell you **which ones you should actually use** (because not all are practical).

🔑 FIRST — IMPORTANT CLARIFICATION

For your project, you do **NOT** need:

- Full real-field datasets
- Live sensor feeds

- Proprietary company data

You need **example / historical-style data** that:

- Shows drilling events, depth, time, NPT, etc.
- Is good enough to **demonstrate your system**

Many applied projects **combine public examples + simulated data**. That is normal.

REAL PLACES YOU CAN GET PUBLIC / EXAMPLE DATA

1 Academic & Educational Example Datasets (BEST FIT FOR YOU)

These are datasets used in **research papers, teaching, and demos**.

Where to look:

- **SPE (Society of Petroleum Engineers) papers**
 - Often include **sample tables**, figures, or partial datasets
 - You can **recreate the structure** from what's shown
- **University petroleum engineering repositories**
 - Search terms like:

“drilling operations dataset”

“non-productive time drilling data”

“drilling event log example”

 These are usually **not full raw data**, but they are **perfect for guiding simulation**.

2 Kaggle (LIMITED BUT USEFUL)

Kaggle sometimes has:

- Drilling parameters

- NPT-related datasets
- Well operation summaries

Search on Kaggle:

- “drilling operations”
- “oil gas well data”
- “non productive time drilling”

⚠ Limitation:

- Often **missing depth-based event structure**
- You may need to **reshape or extend** the data

Still acceptable for:

- KPI examples
 - Historical comparison
-

3 Government / Regulatory Data (SUPPORTING ONLY)

Examples:

- Alberta Energy Regulator (AER)
- U.S. Bureau of Ocean Energy Management (BOEM)

They provide:

- Well metadata
- Completion dates
- Depths

⚠ These are **not operational event logs**, but they help with:

- Well info tables
 - Location mapping realism
-

🚫 WHAT YOU WILL NOT FIND (AND THAT'S OKAY)

You will NOT find:

- Full daily drilling reports
- Detailed event-by-event logs
- Equipment stress records

These are proprietary — and **no one expects you to have them**.

✓ WHAT YOU SHOULD ACTUALLY DO (BEST PRACTICE)

✓ Recommended approach for YOUR project

1. Use simulated data as your primary dataset
2. Base the structure on patterns described in literature
3. Optionally:
 - Use small public datasets for validation or illustration
4. Clearly state:

“Simulated and publicly available example datasets”

This is the **safest and strongest** approach.

✓ FINAL REASSURANCE

You are **not missing anything**.

This is exactly how applied projects in restricted domains are done.

If you want, next I can:

- Help you **design a realistic simulated dataset (CSV)**
- Show you **how to cite ‘public examples’ properly**

Just tell me 

okay can you help me explain the full project in one page

Project Title

Visual Drilling Operations & Predictive Maintenance Decision Support System

Project Overview

This project proposes the design and development of a **web-based decision support software** for drilling operations. The system visually represents drilling activities along **well depth** and applies **data analytics and predictive maintenance techniques** to identify operational risks, inefficiencies, and potential equipment failures.

Unlike traditional dashboards that present only tables or charts, this system focuses on **depth-based visualization**, enabling engineers to **see problems directly along the wellbore** and understand *why* certain risks are increasing. The software is designed to be **reusable across wells, fields, and companies**, making it productivity-focused rather than academic-semester-specific.

Problem Statement

Drilling data is often available but difficult to interpret quickly during operations. Engineers rely on multiple reports, logs, and dashboards that:

- Do not clearly link problems to depth
- Hide analytical logic
- Make it difficult to understand risk buildup over time

This project addresses the gap by transforming raw operational data into **visual, explainable, and actionable insights**.

Proposed Solution

The system will provide an **interactive vertical wellbore (pipe) visualization** where drilling events, operational problems, and maintenance risks are displayed as **colored depth segments**.

Key capabilities include:

- Depth-based visualization of drilling events (e.g., stuck pipe, reaming, downtime)
- KPI computation for drilling performance and efficiency
- Predictive maintenance risk scoring for drilling equipment
- Explainable analytics showing *why* a risk or alert is generated
- Interactive exploration (clicking a depth segment reveals detailed information)

The system does **not replace engineers**; instead, it acts as a **visual intelligence layer** that supports decision-making.

Key Features

1. Depth-Based Well Visualization

- Vertical pipe representation of the well
- Depth labels (measured depth)

- Color-coded segments indicating operational states and problems

2. KPI & Performance Analytics

- Rate of penetration (ROP)
- Non-productive time (NPT)
- Event frequency by depth
- Drilling efficiency indicators

3. Predictive Maintenance Module

- Equipment risk scores based on cumulative operational stress
- Identification of high-risk depth intervals
- Historical pattern comparison across wells

4. Explainable Decision Support

- Clear explanation of why risks are increasing
- Links each prediction to specific data patterns and events
- Allows engineers to interpret and validate results

Technology Stack

- **Frontend:** React.js (interactive visualization & UI)
- **Backend:** Python (FastAPI)
- **Database:** PostgreSQL / MySQL
- **Analytics:** Python (pandas, basic ML/statistical models)

Data Source

The system will use **simulated or historical drilling datasets**, including:

- Depth

- Time
- Operational events
- Equipment usage logs
- Maintenance records

No live sensors are required; the focus is on **software architecture, analytics logic, and visualization.**

Expected Outcome

The final deliverable will be a **fully functional web application**, not a report, demonstrating:

- End-to-end software development (frontend, backend, analytics)
 - Realistic engineering use-case implementation
 - Reusable architecture applicable to multiple drilling operations
 - Clear visualization of complex operational data
-

Why This Project is Reusable

- Not tied to students, semesters, or coursework
- Applicable across different wells and drilling campaigns
- Scalable to other industrial operations with depth-based data
- Demonstrates transferable software engineering and analytics skills

how do i do a gnatt chart for the timeline?

Great question — a **Gantt chart** is exactly the right way to present your timeline, and it's actually **very easy** to do in **Word or Excel**.

I'll show you **two simple methods**.

👉 **Method 1 (Excel → best & cleanest)**

👉 **Method 2 (Word only → acceptable if you don't want Excel)**

I'll also base it on **your REAL timeline**, not a fake one.

✓ **METHOD 1 (RECOMMENDED): GANTT CHART USING EXCEL**

This looks the most professional and is easiest to update.

◆ **STEP 1: Create the timeline table in Excel**

Open **Excel** and enter this:

Task	Start Week	Duration (Weeks)
Topic research & feasibility	1	2
Proposal writing	3	1
Frontend setup & navigation	4	1
Depth-based visualization	5	2
Backend & data ingestion	7	2
KPI & analytics	9	2
Predictive maintenance logic	11	2

B.2 From Copilot

this is what i am going to do. but i need to know if its already in existence and what are the gaps so that i can work on it: Project Title Visual Drilling Operations & Predictive Maintenance System

Project Summary This project proposes the development of a web-based software application that helps engineers analyze drilling operations by visualizing drilling events along well depth and identifying areas of operational risk and potential maintenance concern. Instead of relying on traditional reports or static dashboards, the system presents drilling data through an interactive vertical wellbore (pipe) visualization, where operational events such as stuck pipe, reaming, or downtime are displayed as color-coded depth segments. This allows users to quickly identify what went wrong, where it occurred, and why the risk is increasing. The application computes key drilling performance indicators (such as non-productive time and event frequency) and applies predictive maintenance logic to highlight equipment that may require inspection or review. All insights are explainable and based only on recorded operational data, supporting engineering judgment rather than replacing it. The system is designed to be reusable across wells and drilling campaigns and will be implemented as a full-stack web application, not a written report.

Core Components • Depth-based wellbore visualization of drilling operations • Drilling KPIs and non-productive time analysis • Predictive maintenance risk indicators • Explainable insights linked to recorded events

Technology • Frontend: React.js • Backend: Python (FastAPI) • Database: PostgreSQL / MySQL • Analytics & Visualization: Python, SVG / D3.js

Outcome A functional, interactive software application demonstrating full-stack development, data analytics, and decision-support for drilling operations.

You've chosen a **very strong and original project direction**, and the good news is this:

 **Your exact idea does NOT currently exist as a single integrated system.**

But parts of it exist in different commercial tools — and the *gaps* between those tools are exactly where your project becomes unique, academically strong, and industry-relevant.

Below is a clear breakdown based on the search results and industry knowledge.

1. What Already Exists (Based on Industry Tools)

Several companies offer **drilling visualization or real-time monitoring**, but none combine all the features you described into a single, lightweight, explainable, web-based system.

A. Real-time drilling visualization platforms

Tools like Petrolink and Halliburton HalVue provide real-time drilling data visualization and monitoring.

- Petrolink emphasizes real-time data visualization and analytics for drilling operations
- Halliburton's HalVue offers intuitive real-time visualization and data correlation tools

BUT:

They focus on *real-time rig data*, not depth-based event visualization for post-run analysis.

B. Wellbore schematics & well path visualization

Tools like wellVizion and PathView visualize well schematics and 3D well paths.

- wellVizion provides well schematics and drilling/completions data visualization
- PathView visualizes 3D well trajectories and anti-collision analysis

BUT:

They do not visualize **operational events along depth** (e.g., stuck pipe, reaming, NPT segments).

C. Drilling optimization & predictive analytics

Platforms like Corva and SLB DrillOps offer predictive drilling insights.

- Corva provides real-time drilling optimization and predictive alerts
- SLB DrillOps offers predictive analytics for drilling events and multi-rig monitoring

BUT:

They are enterprise-scale, real-time, expensive, and not designed for **post-operation event visualization or explainable maintenance indicators**.

■ 2. What Does NOT Exist (Your Unique Gap)

Your project sits in a **clear gap** between drilling visualization tools and predictive maintenance systems.

Here are the gaps your system fills:

★ Gap 1 — Depth-Based Event Visualization (Vertical Pipe View)

No major tool provides a **simple, interactive vertical wellbore visualization** where:

- Each event is a color-coded depth segment
- Users can scroll through depth
- Events like stuck pipe, reaming, circulation loss, NPT are visually mapped

This is **unique** and extremely useful for post-run analysis.

★ Gap 2 — Explainable Predictive Maintenance for Drilling Equipment

Existing predictive systems (Corva, DrillOps) focus on **real-time drilling performance**, not:

- Maintenance risk scoring
- Equipment condition indicators
- Explainable logic tied to recorded events

Your system uses **event history** to infer maintenance risk — that's new.

★ Gap 3 — Lightweight, Web-Based, Reusable Tool

Existing tools are:

- Enterprise-scale
- Expensive
- Real-time only
- Not reusable for student or small-team projects
- Not customizable

Your system is:

- Lightweight
- Upload-based
- Reusable across wells
- Built with open technologies (React + FastAPI)

This makes it academically perfect.

★ Gap 4 — Integration of Drilling KPIs + Event Visualization + Maintenance Risk

No existing tool combines:

- Drilling KPIs
- Depth-based event visualization
- Predictive maintenance indicators
- Explainable insights

This integrated approach is **your competitive advantage**.

★ Gap 5 — Explainability (Why the Risk Increased)

Commercial tools use black-box ML.

Your system explicitly explains:

- “Risk increased because stuck pipe occurred at 3200–3300 ft.”
- “High event frequency indicates potential equipment wear.”
- “NPT spike suggests operational inefficiency.”

This is a **huge academic strength**.

■ 3. Why Your Project Is Still Original and Strong

Even though parts of your idea exist in different tools, **no single system** combines:

- ✓ Depth-based event visualization
- ✓ Drilling KPIs
- ✓ Predictive maintenance indicators
- ✓ Explainable insights
- ✓ Web-based full-stack implementation
- ✓ Reusability across wells

This makes your project:

- Unique
- Feasible
- Industry-relevant
- Academically impressive
- Not competing with enterprise systems

- Perfect for a final-year project