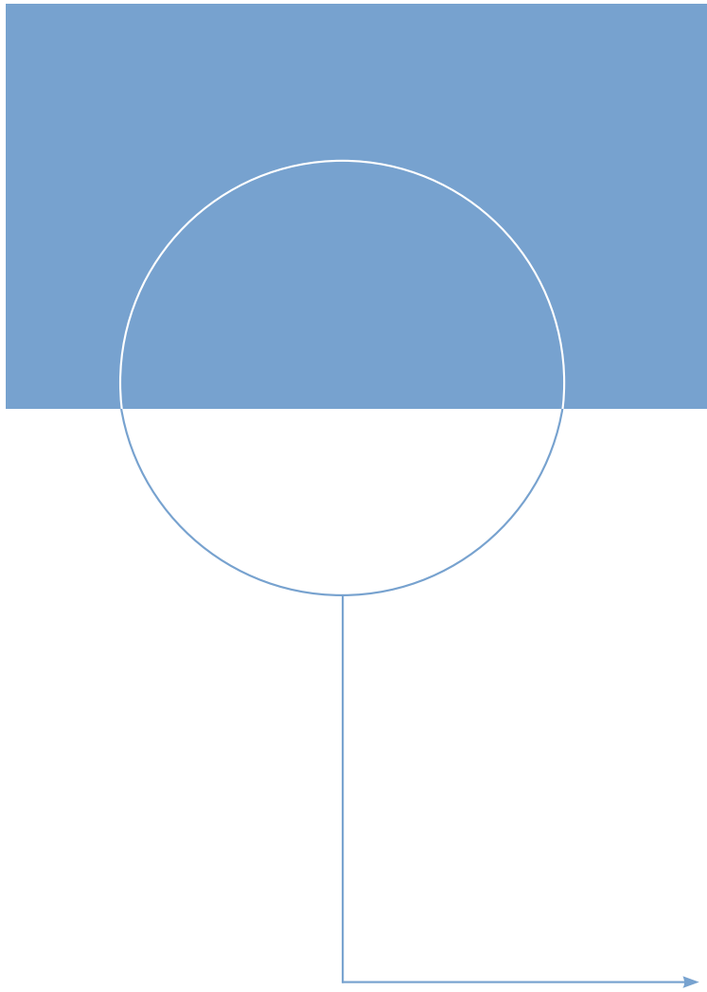


Master's thesis

2024

Amin Amini Azady

NTNU
Norwegian University of
Science and Technology
Faculty of Engineering
Department of Energy and Process Engineering

Amin Amini Azady

PlugDigitize: Developing Digital Solutions for Well Abandonment Operations

Innovative Approaches to Digital Technology for Permanent Plugging and Abandoning Oil Wells

January 2024

NTNU
Norwegian University of
Science and Technology

NTNU
Norwegian University of
Science and Technology

NTNU
Norwegian University of
Science and Technology

PlugDigitize: Developing Digital Solutions for Well Abandonment Operations

Innovative Approaches to Digital Technology for Permanent Plugging and Abandoning Oil Wells

Amin Amini Azady

Petroleum Engineering
Submission date: January 2024
Supervisor: Behzad Elahifar
Co-supervisor: Bjørn Astor Brechan

Norwegian University of Science and Technology
Department of Energy and Process Engineering

Amin Amini Azady

PlugDigitize: Developing Digital Solutions for Well Abandonment Operations

Innovative Approaches to Digital Technology for Permanent Plugging and Abandoning Oil Wells

Master's thesis in Petroleum Engineering
Supervisor: Behzad Elahifar
Co-supervisor: Bjørn Astor Brechan
January 2024

Norwegian University of Science and Technology
Faculty of Engineering
Department of Geoscience and Petroleum Engineering

NTNU

(This page intentionally left blank)

ABSTRACT

This thesis introduces "PlugDigitize," an innovative digital framework designed to optimize well abandonment operations in the oil and gas industry. Centering on the pivotal stage of cement plug placement, the research underscores the integration of digital technologies to augment accuracy, efficiency, and adherence to environmental standards. Developed using Python and PyQt5, the software application features an interactive interface for precise input and calculations in cement plugging scenarios. The study highlights the crucial role of digitalization in enhancing operational procedures, reducing costs, and bolstering safety in well abandonment practices. The thesis demonstrates PlugDigitize's potential to revolutionize traditional practices, contributing to sustainable and responsible well decommissioning. The findings emphasize the urgency for continued innovation and digital integration in the oil and gas sector, with a particular focus on well abandonment processes. Furthermore, this research delves into the regulatory aspects of well abandonment, aligning with procedures outlined in the "Introduction to Permanent Plug and Abandonment of Wells (2020)." This includes a comprehensive understanding of operator responsibilities and the systematic approach to the three key phases of abandonment: reservoir, intermediate, and wellhead and conductor removal. These phases are pertinent regardless of the well's location, type, or status. The thesis not only proposes a technological advancement but also advocates for an industry-wide shift towards more efficient and environmentally conscious practices. The integration of PlugDigitize into standard operating procedures could mark a significant leap in the way the industry approaches the end-of-life phase of wells, setting a new standard for environmental stewardship and operational excellence.

SAMMENDRAG

Denne avhandlingen presenterer "PlugDigitize," et nyskapende digitalt rammeverk designet for å optimalisere operasjoner for nedleggelse av brønner i olje- og gassindustrien. Med fokus på den kritiske fasen av sementpluggplassering, understreker forskningen integreringen av digitale teknologier for å forbedre nøyaktighet, effektivitet og overholdelse av miljøstandarder. Ved bruk av Python og PyQt5 ble en brukervennlig programvareapplikasjon utviklet, med et interaktivt grensesnitt for presis inndata og beregninger i sementpluggscenarioer. Studien fremhever betydningen av digitalisering for å forbedre operasjonelle prosedyrer, redusere kostnader og sikre sikkerhet i brønnforlatelsesprosesser. Avhandlingen demonstrerer PlugDigitize's potensial til å revolusjonere tradisjonelle praksiser, og bidra til bærekraftig og ansvarlig nedleggelse av brønner. Funnene understreker behovet for kontinuerlig innovasjon og digital integrering i olje- og gasssektoren, spesielt i brønnforlatelsesprosesser. Videre går denne forskningen inn på de regulatoriske aspektene ved brønnnedleggelse, i samsvar med prosedyrer beskrevet i "Introduction to Permanent Plug and Abandonment of Wells (2020)." Dette inkluderer en grundig forståelse av operatøransvar og en systematisk tilnærming til de tre nøkkelfasene av nedleggelse: reservoar, mellomliggende, og fjerning av brønnhode og konduktor. Disse fasene er relevante uavhengig av brønnens beliggenhet, type eller status. Avhandlingen foreslår ikke bare en teknologisk fremgang, men taler også for en bransjeomfattende overgang mot mer effektive og miljøbevisste praksiser. Integreringen av PlugDigitize i standard driftsprosedyrer kan markere et betydelig sprang i måten industrien nærmer seg sluttfasen av brønner, og sette en ny standard for miljøforvaltning og operasjonell utmerkelse.

PREFACE

This thesis marks the culmination of my journey as a master's student in petroleum engineering at the Norwegian University of Science and Technology in Trondheim, Norway, and is an integral part of the "TPG4920 Petroleum Engineering, Master's Thesis" course. More than an academic endeavor, it represents a personal challenge to explore and extend my limits.

I am profoundly grateful to Dr. Behzad Elahifar for introducing me to this captivating topic. His optimism and insightful guidance have immensely boosted my confidence and dedication. His calm and positive demeanor provided invaluable support throughout my research journey.

My sincere thanks also go to Dr. Bjørn Astor Brechan for his unwavering support and encouragement. His insightful input and discussions have not only deepened my understanding but also emphasized the significance and potential impact of this research.

A special note of appreciation to my family, whose unwavering support turned this dream into a reality. Their constant encouragement and moral support were pillars of strength, especially during the most challenging moments.

I would like to express special gratitude to my flatmate and best friend, Ali Hashemi. His support, thought-provoking discussions, and feedback have been invaluable. The late-night conversations and the memories we have created together have been truly enriching and have left a lasting impact on me.

This thesis is not only an academic milestone but also a journey of learning, growth, and forming invaluable human connections.

CONTENTS

ABBREVIATIONS

List of all abbreviations in alphabetic order:

- **NTNU** Norwegian University of Science and Technology
- **P&A** Plug and Abandonment
- **WBE** Well Barrier Element
- **WBS** Well Barrier Schematics
- **BOP** Blowout Preventer
- **PSA** Petroleum Safety Authority
- **GUI** Graphical User Interface
- **MD** Measured Depth
- **TVD** True Vertical Depth
- **TOC** Top of Cement

(This page intentionally left blank)

CHAPTER

ONE

INTRODUCTION

1.1 Background

The oil and gas industry, a vital component of the global economy, plays an essential role in our daily lives, fueling transportation, powering industries, and providing energy for homes. Within this sector, the process of well abandonment, the permanent decommissioning of depleted oil and gas wells, stands out as a critical operation. This process is not merely a technical formality; it carries significant environmental, safety, and regulatory implications.

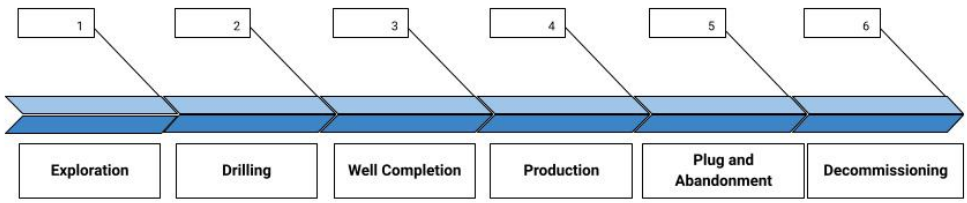


Figure 1.1.1: Lifecycle of an Oil and Gas Well.

Properly abandoned wells are crucial for preventing the leakage of hydrocarbons, which can have severe effects on ecosystems and groundwater. Unsecured wells pose considerable risks, including potential uncontrolled releases of oil or gas. Global regulatory frameworks enforce strict adherence to well abandonment procedures, ensuring operations uphold high standards of environmental and safety compliance.

Despite its critical nature, the methodology for well abandonment has seen little change over the past decades. The oil and gas industry, unlike others that have rapidly advanced through digitalization, has often lagged behind in updating its practices. The COVID-19 pandemic has underscored the urgency for modernization in this field. In response, companies are increasingly recognizing the need for digitalization to enhance efficiency, reduce costs, and save time in well abandonment operations.

Abandoning a well, particularly in the pay sand range where hydrocarbons are extracted, presents intricate challenges. This layer's porosity and permeability, characteristics that once facilitated oil or gas flow, now demand secure sealing. Accurate placement of cement plugs to isolate the pay sand layer is a technical hurdle, requiring robust isolation to withstand long-term environmental pressures and prevent post-abandonment leakage. The complexity is further heightened by the need to navigate existing well infrastructure like casing shoes and production tubing.

The impact of the COVID-19 pandemic has complicated the well abandonment landscape, with plummeting oil and gas demand leading to an increase in idle wells. In the United States alone, over 2.1 million wells are reported as idle, posing environmental and safety hazards if not properly decommissioned. The pandemic has amplified the need for efficient well abandonment processes, propelling the adoption of advanced digitalization techniques. These techniques are essential not only for addressing the immediate need for decommissioning wells but also for contributing to long-term environmental and safety goals.

Ensuring well integrity during abandonment is imperative. Each well must be left in a state where it poses no future environmental or safety risks, which involves a comprehensive understanding of the geological environment, precise execution of plugging operations, and strict adherence to regulatory guidelines. Each abandoned well represents a significant technical undertaking, demanding meticulous planning, execution, and verification.

1.2 Significance of Digitalization

Digitalization has emerged as a transformative force across industries, revolutionizing traditional practices and unlocking new avenues for efficiency and productivity. The oil and gas sector, including well abandonment operations, is no exception to this digital revolution. In recent years, the significance of digitalization in the context of well abandonment has become increasingly evident, offering numerous advantages that extend beyond mere process optimization.

1.2.1 Enhanced Efficiency

Digitalization streamlines and accelerates well abandonment procedures. It enables the automation of various tasks, reducing the reliance on manual labor and minimizing the risk of human error. As a result, the overall efficiency of the well abandonment process is significantly improved. Tasks that once consumed substantial time and resources can now be executed swiftly and accurately.

1.2.2 Cost Reduction

Efficient well abandonment translates to cost savings. Digitalization reduces the need for extensive manpower and resources, resulting in reduced operational expenses ([1]). It optimizes resource allocation, making it possible to achieve more with fewer resources. Cost-effectiveness is particularly critical in an industry characterized by fluctuating oil prices and increasing pressure to maximize returns.

1.2.3 Data-Driven Decision Making

Digitalization empowers decision-makers with access to real-time data and analytics. This data-driven approach enables informed decision-making throughout the well abandonment process. Operators can monitor and analyze crucial parameters, such as cement placement, well integrity, and environmental impact, in real time. These insights facilitate proactive adjustments and ensure that the operation aligns with regulatory standards and best practices.

1.2.4 Environmental Stewardship

In the era of heightened environmental awareness, digitalization plays a pivotal role in enhancing environmental stewardship. By providing accurate monitoring and control mechanisms, it mitigates the risk of environmental harm during well abandonment. Digital tools enable operators to closely track well integrity, preventing hydrocarbon leakage and safeguarding ecosystems and groundwater sources.

1.2.5 Safety Enhancement

Safety is paramount in well abandonment operations. Digitalization introduces safety enhancements by minimizing manual interventions and reducing exposure to hazardous conditions. Automated processes, coupled with remote monitoring capabilities, allow for safer execution of tasks. This ensures the well-being of personnel involved in well abandonment activities.

1.2.6 Regulatory Compliance

Regulatory frameworks governing well abandonment continue to evolve, becoming increasingly stringent. Digitalization facilitates compliance by providing a systematic and auditable approach to the process. Digital records and documentation streamline regulatory reporting, making it easier for operators to demonstrate adherence to standards and regulations.

1.2.7 Future-Proofing

As the oil and gas industry faces ongoing challenges, including fluctuating market conditions and environmental pressures, digitalization offers a pathway to future-proof operations. Adaptable digital solutions can evolve alongside industry dynamics, ensuring that well abandonment processes remain efficient, compliant, and environmentally responsible.

In conclusion, digitalization in well abandonment is not merely a technological trend but a strategic imperative. Its significance lies in its ability to enhance efficiency, reduce costs, improve safety, and ensure environmental compliance ([2]). As the industry grapples with changing landscapes, embracing digitalization becomes a cornerstone for achieving sustainable and responsible well abandonment practices.

1.3 Research Gap and Thesis Objective

The field of well abandonment has seen limited innovation and digitalization over the decades, despite its critical importance in the oil and gas industry. The methods and practices employed in well abandonment have remained largely unchanged for nearly 70 years. This persistence in conventional approaches highlights a significant research gap in the industry. The absence of substantial technological advancements and digital solutions presents a clear opportunity for improvement and optimization.

Well abandonment is a complex operation, particularly in scenarios involving the pay sand range. The intricacies of sealing permeable layers securely while ensuring well integrity demand innovative solutions. Traditional methods, though tried and tested, lack the adaptability and efficiency required to address contemporary challenges effectively. As the world grapples with environmental concerns, safety regulations, and cost pressures, the need for a modernized approach to well abandonment becomes increasingly evident.

This thesis project aims to bridge the existing research gap in the digitalization of well abandonment processes, offering a pioneering step toward modernizing this critical operation. The objectives of this thesis are as follows:

1.3.1 Development of a Digital Framework

The primary objective is to develop a digital framework tailored to the planning and execution of well abandonment operations. This framework will serve as a foundation for incorporating digital technologies into the well abandonment process in the future.

1.3.2 Enhanced Efficiency

The thesis seeks to enhance the efficiency of well abandonment procedures. By introducing digital tools and automation, it aims to streamline various aspects of the process, from cement plug placement to well integrity monitoring, reducing time and resource consumption.

1.3.3 Cost Reduction

Cost-effectiveness is a key focus. The thesis endeavors to reduce operational expenses associated with well abandonment. By optimizing resource allocation and minimizing manual interventions, it aims to achieve cost savings.

1.3.4 Environmental Stewardship

Environmental stewardship is fundamental in well abandonment, underscoring the necessity for innovative digital solutions. This project is committed to developing such technologies, aiming to reduce environmental impacts by averting hydrocarbon leaks and upholding environmental regulations. A significant advancement proposed is the adoption of digital operations. By digitizing operational procedures and integrating them with software that controls automated rig equipment, we foresee a substantial increase in accuracy over traditional manual methods. This enhancement will lead to the production of cement plugs that are not only more consistent but also inherently safer, setting a new standard in well abandonment practices.

1.3.5 Safety Enhancement

Safety is paramount in well abandonment operations. The thesis aims to enhance safety by reducing manual labor and introducing automated processes. It seeks to create a safer working environment for personnel involved in well abandonment activities.

1.3.6 Adherence to Regulatory Standards

Regulatory compliance is a key objective. The thesis aims to develop digital tools that facilitate adherence to evolving regulatory standards governing well abandonment.

1.3.7 Future-Proofing

Recognizing the dynamic nature of the oil and gas industry, the project aims to develop adaptable digital solutions. These solutions will be designed to evolve alongside industry changes, ensuring that well abandonment processes remain efficient and effective.

1.3.8 Preliminary Implementation

While the thesis marks the beginning of the digitalization journey in well abandonment, it aims to provide a tangible prototype and demonstrate its feasibility. The project serves as a foundation upon which further digitalization efforts can build.

In summary, this thesis acknowledges the extensive research gap in the digitalization of well abandonment and sets forth ambitious objectives to address this gap.

It aims to develop a digital framework that enhances efficiency, reduces costs, promotes environmental stewardship, enhances safety, ensures regulatory compliance, and offers a pathway for future-proofing well abandonment operations. While this project represents a crucial initial step, it is envisioned as a catalyst for continued advancements in the digitalization of well abandonment processes.

CHAPTER

TWO

THEORY

2.1 Plug and Well Abandonment Processes

The lifecycle of an oil and gas well culminates in the Plug and Abandonment (P&A) phase, a complex, time-intensive, and costly operation. While successful P&A is contingent upon factors like casing conditions and well barriers, the implementation of permanent P&A often hinges on regulatory and economic drivers. Currently, without a strong regulatory mandate, companies may prioritize profitable operations over non-profitable ones like P&A, especially for 'dead wells' that are monitored and deemed 'safe.' However, should governing authorities revise regulations to mandate more rigorous P&A procedures, the industry may witness a significant surge in these operations, termed the 'Plug Wave.

P&A activities are driven by the overarching principle of well integrity, defined as "the application of technical, operational, and organizational solutions to reduce the risk of uncontrolled release of formation fluids throughout the life cycle of a well" ([3]). Loss of well integrity can have serious implications, including equipment damage, personnel injuries, and environmental harm, leading to additional costs and risks ([4]).

The use of well barriers is pivotal in preventing leakage and ensuring the complete isolation of mobile fluids within the wellbore and from the seabed or surface ([4]). These barriers, which can consist of one or several well barrier elements (WBEs), are crucial throughout the well's lifecycle, including during the P&A phase.

This thesis focuses exclusively on permanent well abandonment, which dictates that the well must be sealed with a perspective of eternity, ensuring that barriers are in place to endure all expected environmental and geological loads it may encounter post-abandonment. The NORSOK D-010 standard is instrumental in defining the functional and performance-oriented requirements for P&A operations, including the design and selection of cement plugs, which are fundamental to establishing permanent barriers.

With the growing recognition of P&A's importance, early planning becomes vital. As Dale Carnegie suggests, effective planning can significantly reduce execution time. The oil and gas industry's evolving landscape, coupled with potential regulatory changes, underscores the need for efficient, cost-effective P&A strategies. Digital technologies in planning and executing P&A operations present a

promising solution to enhance efficiency and reduce costs, ensuring that the final chapter in a well's life is concluded with the utmost diligence and foresight.

In summary, P&A is a critical stage in a well's life cycle, necessitating meticulous planning and execution to ensure environmental safety and regulatory compliance. The development of digital tools and innovative techniques is key to improving the efficiency and reducing the cost of P&A operations, thereby ensuring that this final chapter in a well's life is concluded with diligence and foresight.

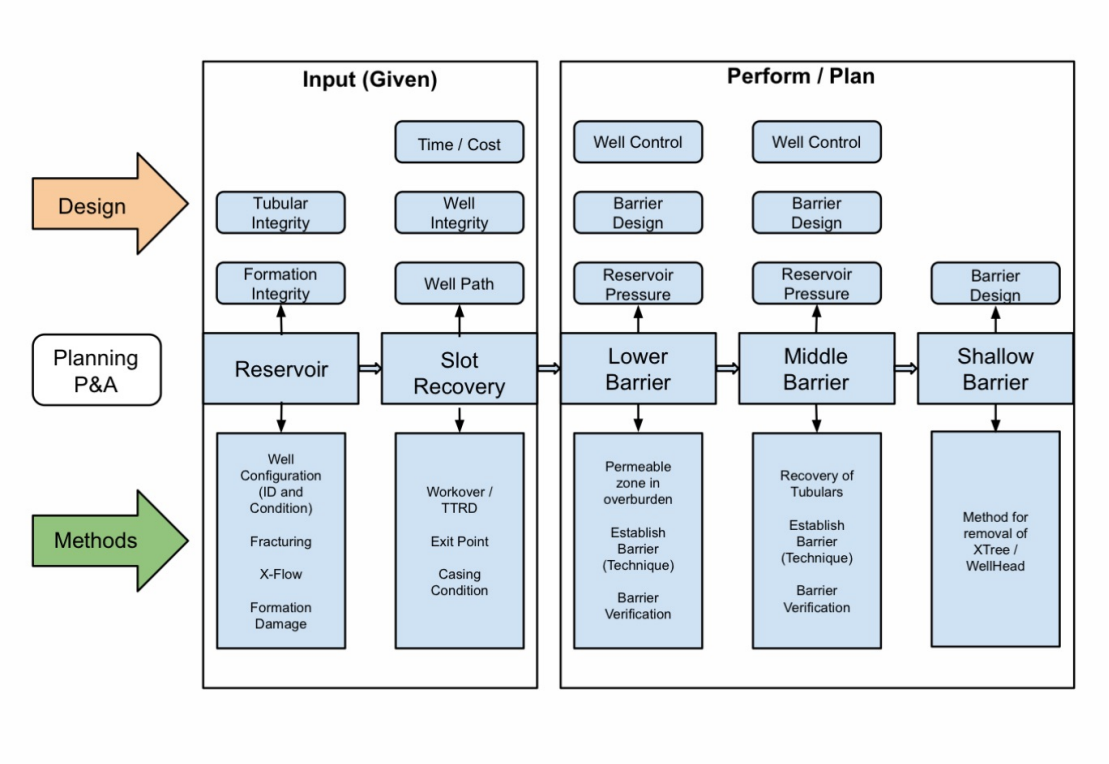


Figure 2.1.1: Strategic Framework for Plug and Abandonment Operations

2.2 Definitions

2.2.1 Well Integrity

Well integrity refers to the condition of a well that enables it to contain any fluids within it throughout its entire lifecycle, preventing any leaks or uncontrolled releases ([4]). It encompasses the technical, operational, and organizational processes to ensure that well barriers are in place, maintained, and effective. This concept is fundamental to environmental protection, operational safety, and efficiency, as it minimizes the risk of accidents that could lead to personnel hazards, environmental incidents, or loss of production

2.2.2 Well Barriers

Well barriers are integral components of maintaining well integrity. They are defined as a series of dependent barrier elements that collectively prevent the unintended flow of formation fluids, gases, or other materials from the wellbore

into surrounding formations or to the surface. These barriers play a pivotal role throughout all stages of a well's lifecycle, including the critical phase of plugging and abandonment, ensuring that mobile fluids are fully isolated within the wellbore and from the external environment.

Well Barrier Elements (WBEs) constitute the individual physical components that, when used in combination, create a well barrier. The design and utilization of WBEs are vital; they must be capable of withstanding the full spectrum of operational loads and environmental conditions anticipated during the well's operational phase and throughout the abandonment period. This robustness is paramount to ensuring the longevity and reliability of the well barriers, particularly when a well is permanently abandoned.

Well Barrier Schematics (WBS) serve as detailed illustrations or diagrams that depict the primary and secondary well barriers, showcasing their independence and the lack of common WBEs between them (Fig. 2.2.1). A WBS includes detailed information such as a list of WBEs, tubular, and cement, providing a visual representation of the well's integrity at various stages of its lifecycle. The WBS is not merely an illustrative tool but a documented assurance of the well's integrity, clearly displaying the independent nature of the barrier systems in place.

Additionally, the Well Barrier Acceptance Criteria are the standards set to gauge the effectiveness and reliability of the well barriers. These criteria include assessments of the well barriers' compressive and tensile strengths, permeability, and ability to maintain these properties over time without showing a deteriorating trend. The criteria also ensure that barriers can withstand mechanical loads and chemical exposure while maintaining a secure bond to steel tubulars and other contact surfaces. This set of criteria is fundamental in verifying that the well barriers will perform their intended functions throughout the expected life of the well, including the abandonment phase.

2.2.3 Suspension

Suspension of a well occurs when operations are temporarily stopped with the intention that the well might be brought back into use in the future. During suspension, the well is secured with necessary barriers, and monitoring measures are implemented to maintain well integrity and environmental safety. The well is essentially put into a safe and stable condition, which can last for an unspecified period, depending on various factors like market conditions, strategic planning, or awaiting new technology for enhanced recovery.

2.2.4 Temporary Abandonment VS. Permanent Abandonment

Temporary abandonment refers to the short-term suspension of a well when there's a possibility that it may be used again. In this scenario, the well is secured with barriers that can be removed if the decision is made to re-enter the well. Permanent abandonment, on the other hand, is the process of sealing a well indefinitely with no intention of re-entry. It involves placing permanent barriers and plugs to ensure the well is safely decommissioned and poses no future risk to the environment or

public safety. The decision between temporary and permanent abandonment is based on the well's potential for future productivity and economic viability.

2.3 Well Abandonment Phases [5]

Upon finalizing the abandonment design, the operator presents the plan to the relevant regulatory authority for review. This authority then either requests modifications or approves the design. Following approval, the operator is authorized to begin the P&A process. It is important to note that the regulatory body's approval does not transfer any operational responsibilities to it; the operator retains all responsibilities during the P&A process and subsequent operations.

As detailed in the book "Introduction to Permanent Plug and Abandonment of Wells ([5])," a P&A operation generally encompasses three phases: reservoir abandonment, intermediate abandonment, and wellhead and conductor removal. These phases apply universally, regardless of well location (offshore or onshore), type (exploratory, producing, injecting, etc.), or status (partially abandoned, shut-in, etc.).

2.3.1 Abandonment of Reservoir

The initial step in reservoir abandonment involves examining the wellhead and setting up a wireline unit. This unit is used to assess the wellbore accessibility and evaluate the condition of the production tubing by conducting a caliper log. This initial stage, termed Phase 0—well intervention, plays a crucial role in reducing the time during Plug and Abandonment (P&A) operations. Additionally, systems for managing liquid and solid waste are put in place ([6]). This phase continues with a test to check the well's integrity. If the well is intact, cement slurry is injected to seal the main reservoir. The cement plug's effectiveness is then verified through pressure testing once it has sufficiently hardened. This portion of the process does not require a rig. However, if the well's integrity is compromised, a rig must be brought in and a Blowout Preventer (BOP) connected. The cement which is using for sealing is classified as primary and secondary barriers following the reservoir abandonment phase. Generally, Phase 1 is considered complete when permanent primary and secondary barriers have been established to secure the main reservoir. The production tubing can either be removed or left in the well as part of the well barrier system.

2.3.2 Intermediate Abandonment

The Intermediate Abandonment involves activities such as milling, removing casing, establishing barriers to segregate any intermediate zones containing hydrocarbons or water, and placing an environmental plug. If not already done in phase 1, part of the production tubing may also be removed during this phase. Phase 2 is considered finished once all identified flow potentials in the overlying layers have been effectively secured.

2.3.3 Removals of Conductor and Wellhead

During this phase, both the conductor and wellhead are severed below the surface or the seabed level and then extracted. This action is primarily taken to prevent any potential conflicts with other marine activities, such as fishing. In the Norwegian sector of the North Sea, this particular phase is typically classified as a marine operation rather than a drilling activity.

2.4 Regulations of Plug and Abandonment Process

In Norway, the oversight of Plugging and Abandonment (P&A) operations falls under the jurisdiction of the Norwegian Petroleum Safety Authority (PSA) and is further guided by the Norwegian Petroleum Act. This act ensures that petroleum activities are managed with a focus on safety, efficiency, and protection of national interests. The NORSOK D-010:2021 standard complements these regulations by setting out the minimum requirements for well integrity, which includes the entire well lifecycle up to and including abandonment (NORSOK D-010:2021, p. 82).

2.4.1 Well Integrity and Barrier Requirements

Well integrity is central to P&A operations and is defined as the application of solutions to mitigate the risk of uncontrolled formation fluid release throughout a well's lifecycle. Integrity is maintained through well barriers, established at every stage from drilling to abandonment, to ensure full isolation of mobile fluids. These well barriers, composed of WBEs, must withstand operational and environmental conditions throughout the abandonment period (NORSOK D-010:2021, p. 83).

2.4.2 Design and Load Considerations

The design basis for P&A must account for all potential inflow sources and the load cases that could affect well integrity. For permanently abandoned wells, designs should account for natural reservoir re-pressurization and documented environmental conditions, considering the long-term geological and chemical processes that could impact well barriers (NORSOK D-010:2021, pp. 83-84).

2.4.3 Temporary and Permanent Abandonment Practices

Temporary abandonment may involve continuous monitoring and routine testing of barriers, with specific provisions for subsea wells regarding visual observation programs. Conversely, wells without monitoring must have WBEs with integrity to cover the planned abandonment duration (NORSOK D-010:2021, p. 86).

For permanent abandonment, the standards demand that barriers are designed with an eternal perspective, ensuring that they extend across the full well cross-section and seal all annuli vertically and horizontally. These barriers must be placed adjacent to formations with adequate integrity to endure the maximum anticipated pressure (NORSOK D-010:2021, p. 97).

2.4.4 Material Suitability and Verification

The chosen plugging materials' suitability must be verified, documented, and considered for long-term casing degradation. This verification is crucial to ensure the integrity of the permanent well barriers, which are essential to prevent fluid migration and maintain isolation (NORSOK D-010:2021, p. 98).

In line with Norway's regulatory framework and NORSOK standards, the process of P&A must prioritize the establishment of robust well barriers to protect the environment and comply with safety regulations. With an increasing number of wells reaching the end of their productive life, Norway's structured approach ensures that P&A operations are conducted with diligence and foresight, safeguarding the environment and adhering to the highest standards of well integrity (NORSOK D-010:2021, p. 82).

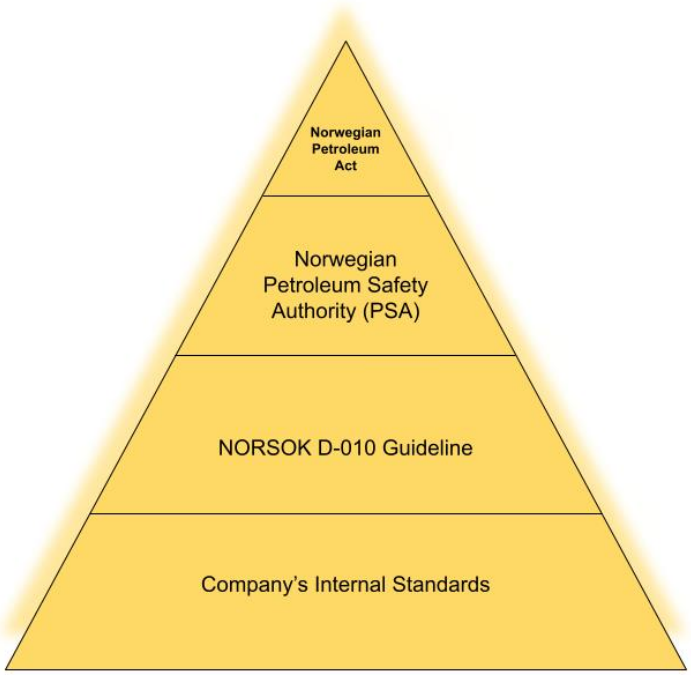


Figure 2.4.1: Regulatory Hierarchy in Norwegian Oil and Gas Operation

2.5 Techniques of Plug Placement [5]

In the processes of well construction and plug and abandonment (P&A), cement placement plays a crucial role in ensuring well integrity and environmental safety. This section offers an in-depth review of the cement placement techniques employed in the oil and gas industry, highlighting their advantages, limitations, and the latest advancements in the field.

The classification of these methods is derived from the book "Introduction to Permanent Plug and Abandonment of Wells" authored by Mahmoud Khalifeh and Arild Saasen.

2.5.1 Balanced Plug Method

This method is the most widely used for installing permanent plugs. To set the base of the plug, a work string is lowered into the well to a specific depth. Since the work string is encased in mud, spacer and chemical wash are injected both before and after the cement mixture to prevent the mud from contaminating it and to ensure proper surface adherence to the casing or formation. The cement slurry is then pumped down through the work string, rising in the annulus between the work string and the casing or formation. The quantities of the spacer used before and after the slurry are carefully calculated to ensure that the spacer reaches the same height both inside and outside the work string, as shown in Fig.2.5.1.

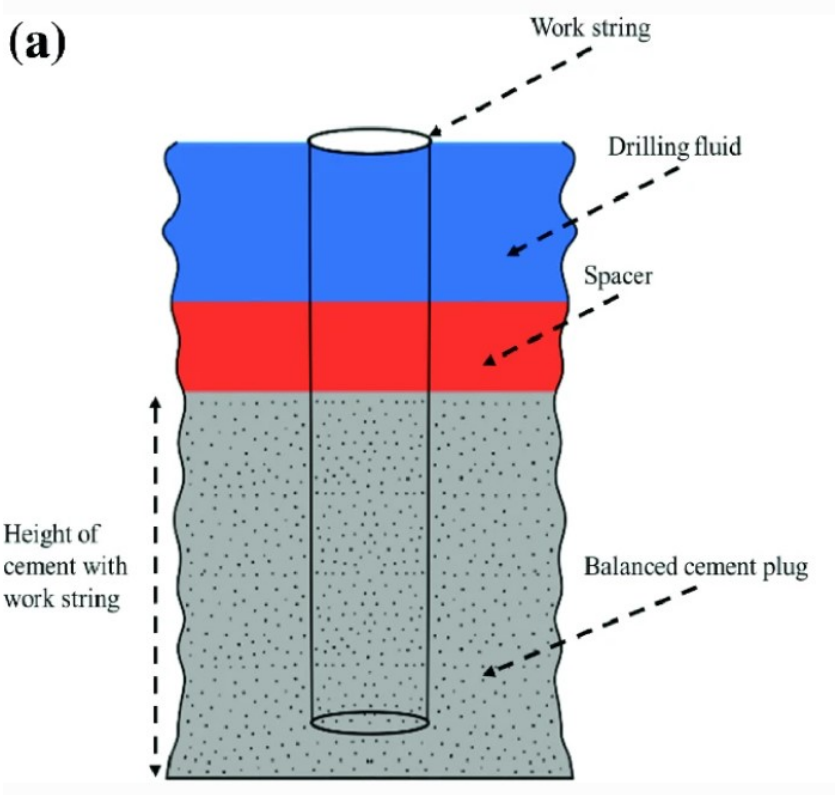


Figure 2.5.1: Balanced-plug placement technique.*Introduction to Permanent Plug and Abandonment of Wells* by Mahmoud Khalifeh and Arild Saasen (2020), Chapter 7, p. 191

2.5.2 Two-Plug Method

To reduce the risk of contaminating the cement plug with the fluids preceding and following it, the two-plug method is employed, as shown in Fig.2.5.2. This approach involves using a wiper dart in front of the cement plug (situated between the lead cement slurry and the spacer) and another wiper dart after the slurry (located between the tail cement slurry and spacer). This arrangement ensures

that the slurry is completely isolated from the spacer from the surface down to a point near the tailpipe or stinger, thereby minimizing contamination risk. Each wiper dart contains a diaphragm designed to withstand pressure up to a certain limit, which then ruptures under higher pressure. Near the stinger or tailpipe, the work string includes a locator sub. When the first wiper dart reaches the locator sub, the pressure builds up until the diaphragm breaks, allowing the cement to flow through the first dart. Subsequently, the second wiper dart lands on the first, leading to another pressure increase. This increased pressure causes the second diaphragm to rupture, allowing the spacer to pass through.

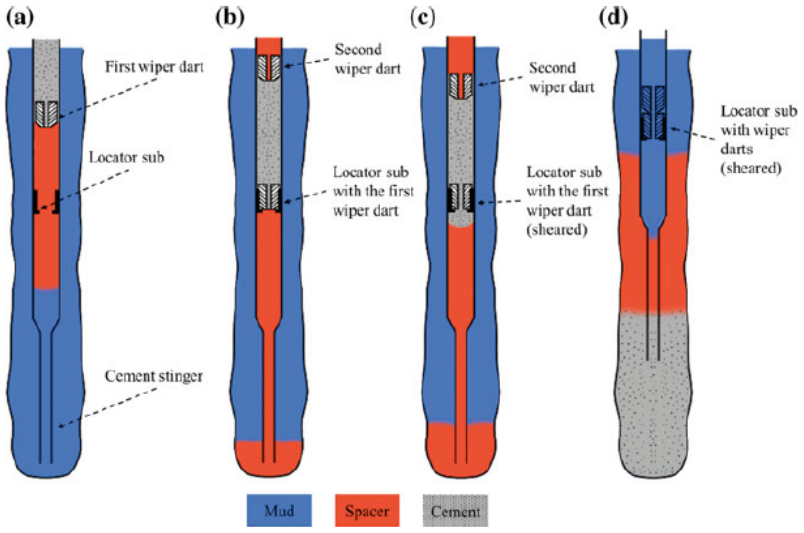


Figure 2.5.2: Two-Plug Method. *Introduction to Permanent Plug and Abandonment of Wells* by Mahmoud Khalifeh and Arild Saasen (2020), Chapter 7, p. 195

2.5.3 Dump Bailer Method

The dump bailer is a wireline tool designed for deploying small amounts of slurries precisely at a targeted depth with minimal contamination, and it is typically used in onshore wells. The tool is loaded with cement and lowered into the wellbore. Upon reaching the desired depth, the bailer cap is opened either electronically through a signal or mechanically by contacting a mechanical foundation. Usually, a mechanical foundation is employed when using a dump bailer, as shown in Fig.2.5.3. The advantages of this method include reduced contamination risk, cost-effectiveness, no need for a drilling rig during operation, precise control over plug depth, and significantly shorter operational time compared to other methods. However, there are limitations, such as the limited capacity of bailers requiring multiple runs, the possibility of cement setting inside the bailer due to static conditions while descending, and uncertainties in removing mud or spacer. It's important to prevent slurry gelation or instability to ensure the slurry can be effectively released from the dump bailer. While this method proves effective in small-diameter pipes, creating a high-quality plug in large-bore pipes presents challenges due to the limited capacity of the bailer and the interface issues between each trip.

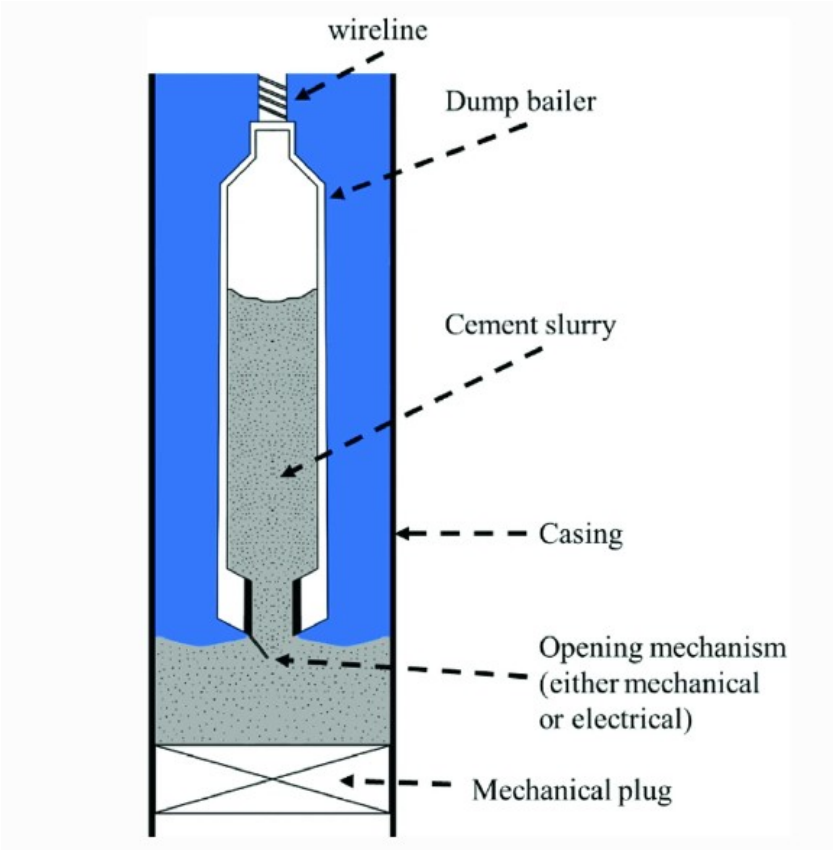


Figure 2.5.3: Dump Bailer technique.*Introduction to Permanent Plug and Abandonment of Wells* by Mahmoud Khalifeh and Arild Saasen (2020), Chapter 7, p. 195

2.5.4 Coiled Tubing Method

Coiled tubing refers to a long, continuous pipe wound around a reel. Before insertion into the wellbore, the pipe is straightened, and after use, it is rewound back onto the transport and storage spool. The length of coiled tubing can vary between 2,000 to 15,000 feet or more, depending on the pipe diameter and spool size. The table in Fig.2.5.4, details typical sizes of coiled tubing.

The use of coiled tubing in remedial cementing started in the early 1980s and has since gained significant attention. This method has been particularly cost-effective for placing small volumes of cement slurries in various applications, such as fixing channeling behind tubulars, sealing off perforations, injecting cement into perforations, addressing lost circulation zones during drilling, and setting cement whipstocks. One of the main advantages of coiled tubing is that it's a single, continuous pipe, which reduces the issues related to making connections and eliminates the need for a conventional rig, thus making it a cost-effective approach.

However, there are some challenges associated with the use of coiled tubing for cement plug placement. These include concerns about pipe fatigue, the efficiency of hole cleaning, the need for specially designed cement slurries, limitations in unit space and capacity, crane capacity, and adherence to local regulations.

- **Fatigue Problems:** Fatigue life of coiled tubing is a significant concern,

especially as the diameter increases for cementing applications. The tubing undergoes stress each time it is wound and unwound from the reel and passes over the coiled tubing unit's gooseneck. Larger diameters exacerbate this issue. Internal pressure within the coiled tubing during bending and straightening also contributes to fatigue. Since there is no practical non-destructive method to measure cumulative damage, predictive models have been developed to estimate the lifespan of coiled tubing.

- **Hole Cleaning:** The smaller size of coiled tubing limits its flow capacity, and the absence of mechanical agitation from pipe rotation diminishes its efficiency in cleaning larger holes.
- **Unit Space and Capacity:** The feasibility of coiled tubing for cementing requires consideration of the available deck area for equipment such as the reel, injector, and various cementing and testing equipment. The structure must support the weight of these equipments without risk of failure. This is true for both onshore (soil and area capacity) and offshore settings (platform, drillship, vessel, or semi-submersible weight capacities).
- **Crane Capacity:** In offshore operations, platform cranes must be capable of lifting equipment from supply boats. Larger pipe diameters necessitate higher crane capacities and introduce additional hazards.
- **Local Regulations:** These regulations are focused on ensuring the safety of coiled tubing operations, encompassing aspects like quality control, well-site safety standards, and safe tool deployment. Factors like well control equipment, pressure ratings, fatigue prediction, unit capacity, and crane capacity are key regulatory concerns, though criteria vary among regulatory authorities.
- **Cement Slurry Design:** Due to the lower flow capacity of coiled tubing compared to drillpipe, the composition of cement slurries used in coiled tubing differs from standard primary cement. The mixing energy imparted by coiled tubing affects the slurry, necessitating adjustments like longer thickening times and lower viscosity and yield stress.

2.6 Digitalization in the Oil and Gas Industry

The advent of digitalization in the oil and gas industry, particularly in the context of P&A operations, has emerged as a transformative force, offering efficiency, precision, and significant time savings. Digital tools, extending beyond simple calculators, facilitate critical decision-making processes and streamline complex operations. For instance, the development of a prototype cement plugging calculator in the P&A sector illustrates how digitalization can simplify the selection of cement placement techniques and automate the generation of outputs such as pumping schedules. These innovations highlight the potential of digital tools to enhance operational efficiency and precision in the P&A process.

The development of a sophisticated software system, grounded in extensive field experience and seamlessly linked to advanced planning tools, is another example of such transformation. This software, harnessing data-driven insights and

integrating with the latest technological advancements in automated rig equipment, elevates the precision and efficiency of operations. Its potential to anticipate operational needs and adapt to varying conditions ensures optimal performance, thereby streamlining workflow and enhancing the quality of tasks, from drilling to well abandonment.

Despite its evident benefits, the integration of digitalization in the oil and gas industry, particularly in the P&A sector, faces several challenges ([5]). One significant hurdle is the reluctance of operators to share data from completed P&A operations, compounded by issues of confidentiality. This lack of data availability hampers the development and effectiveness of digital tools. Moreover, the vast volumes of data generated by digitalization pose challenges in terms of storage, analysis, sharing, and security. Addressing these challenges is crucial for the successful development and implementation of digital solutions in the industry.

The incorporation of digital planning software into the P&A sector presents a promising avenue for industry transformation. Such software, encompassing all disciplines involved in a well's lifecycle, could significantly reduce human errors, a common cause of operational failures. The impending need to plug and abandon a growing number of out-of-service wells further underscores the urgency for integrating digital planning tools in the drilling industry, especially in the P&A sector.

In conclusion, as we venture further into the realm of automation and digitalization, it's crucial to recognize the transformative potential these technologies hold for the P&A sector. The integration of advanced software systems, particularly those that enhance control and precision in automated rig operations, will be pivotal in redefining operational standards. While challenges such as regulatory constraints and data confidentiality remain, the path forward is clear: embracing digital innovation is not just a choice but a necessity for the future sustainability and efficiency of the oil and gas industry, especially in the critical area of well abandonment.

2.7 Python and PyQt5

2.7.1 Why Python?

From the onset of this project, it was evident that the endeavor would require a platform conducive to future improvements and extensive customization. Initially, my inclination was towards MATLAB, owing to my familiarity and previous experience with it. However, after thorough discussions with my professor, we concluded that Python would be the more suitable choice, primarily due to its versatility and flexibility.

Python is renowned for its adaptability across a myriad of applications, ranging from web development to scientific computing, and from data analysis to machine learning and artificial intelligence. This extensive applicability is a testament to Python's ability to handle both straightforward scripting tasks and more intricate, sophisticated applications. The language's adaptability ensures that it can comfortably accommodate the evolving and diverse needs of this project.

Another significant advantage of Python is its expansive standard library, equipped with modules and functions that cater to a wide array of tasks. This

extensive library is a major asset, as it substantially diminishes the time and effort required for code development, enabling more focus on project-specific functionalities rather than foundational elements.

Furthermore, Python's open-source nature makes it freely available for use and distribution, even for commercial purposes. This aspect is crucial for the scalability and accessibility of the project. The language is also supported by an extensive range of third-party libraries, which further broadens its capabilities, offering an array of tools and functionalities that can be seamlessly integrated into our project.

Given these factors, Python emerges as the ideal choice for this project. Its ease of development, coupled with the aforementioned benefits, ensures that we can build a robust, adaptable, and future-proof platform, aligning perfectly with the project's long-term objectives and requirements.

2.7.2 PyQt5 Library

In the development of our project, a critical component was the integration of a user-friendly and efficient graphical user interface (GUI) ([7]). The GUI is pivotal in showcasing the case to users in an accessible and intuitive manner. To achieve this, we evaluated various GUI libraries, seeking one that combined ease of use with powerful features. Our quest led us to PyQt5, which stood out due to its remarkable capabilities in GUI development and presentation.

PyQt5, an extensive set of Python bindings for the Qt application framework, offers a comprehensive solution for creating robust and responsive GUIs ([8]). Its strength lies in its versatility and the breadth of its features. PyQt5 provides a wide range of tools and widgets that can be utilized to build customized, interactive, and highly functional interfaces ([7]). This adaptability makes it an excellent choice for our project, which requires a blend of simplicity and sophistication in its GUI.

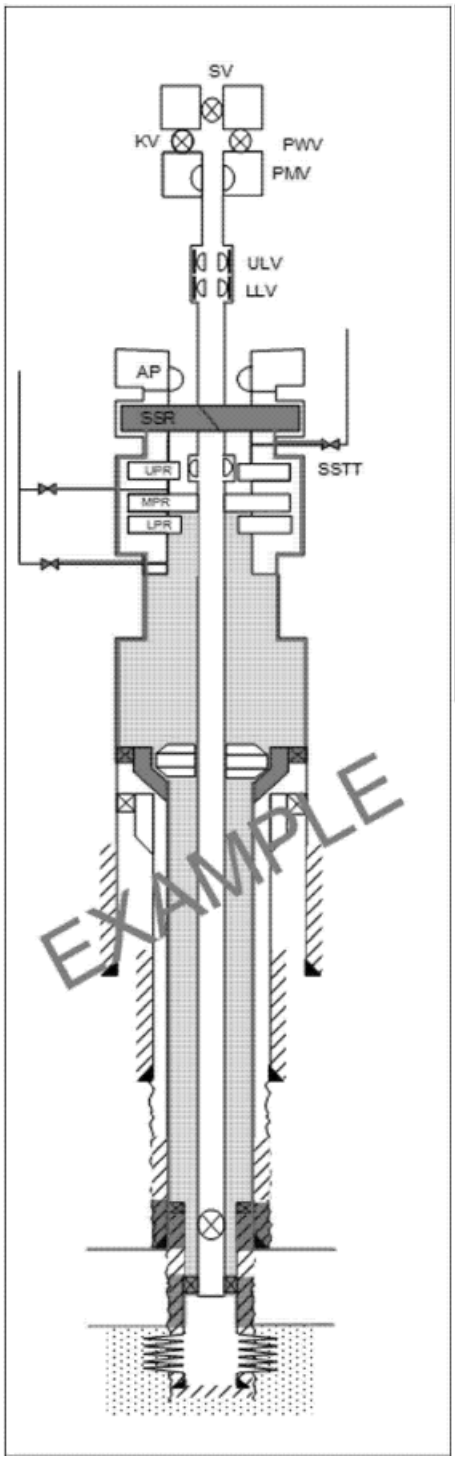
One of the key advantages of PyQt5 is its integration with Qt Designer. Qt Designer is a powerful tool that simplifies the process of designing complex GUIs. It allows developers to visually construct their interfaces, dragging and dropping elements to create layouts. This visual approach to GUI design not only accelerates the development process but also enables immediate visualization of the end result. The ability to instantly see and modify the interface makes the development cycle more efficient and reduces the likelihood of errors.

Additionally, PyQt5's extensive documentation and supportive community play a significant role in easing the development process. The library's documentation provides clear guidelines and examples, which are invaluable for both novice and experienced developers. Furthermore, the active community surrounding PyQt5 offers a wealth of knowledge and support, assisting in troubleshooting and sharing best practices.

Another notable feature of PyQt5 is its cross-platform capability. Applications developed with PyQt5 can run seamlessly on various operating systems, including Windows, macOS, and Linux. This cross-compatibility ensures that our project can reach a wider audience without requiring significant modifications for different platforms.

In conclusion, the selection of PyQt5 as our GUI library is a strategic decision

that aligns with our project's goals. Its combination of powerful features, ease of use with Qt Designer, extensive documentation, and cross-platform capabilities positions it as the ideal tool for developing a user-friendly and effective GUI for our project.



WBE	EAC table	Verification/monitoring
Primary well barrier		
In-situ formation	51	
Annulus (liner) cement	22	
Casing (production liner)	2	
Liner top packer	43	
Well test packer	35	
Fluid column	1	
Downhole tester valve (body only)	46	
Well test string	27	
Subsea test tree	32	
Subsea lubricator valve	45	
Surface test tree	34	
Secondary well barrier		
In-situ formation	51	
Annulus cement	22	
Casing	2	
Wellhead	5	
Drilling BOP	4	

Figure 2.2.1: An example of WBS, NORSOK D-010 2021, Chapter 7, p. 58

Specified dimensions				Axial load capacity		Pressure capacity		Torsional strength		External displacement	Internal capacity per 1000 ft
Outside diameter (in.)	Wall thickness (in.)	Inside diameter d (in.)	Nominal weight (lb/ft)	Yield load (lb) t_{nom}	Tensile load (lb) t_{nom}	Yield pressure (psi)	Hydrotest pressure 90% (psi)	Yield (ft/lb) t_{min}	Ultimate (ft/lb) t_{min}	Barrels	Barrels
1.250	0.190	0.870	2.16	50,620	55,680	23,040	15,000	1,097	1,206	1.52	0.74
1.500	0.087	1.326	1.32	30,900	33,990	8,850	7,970	955	1,051	2.19	1.71
1.750	0.109	1.532	1.91	44,950	49,950	9,510	8,560	1,609	1,770	2.97	2.28
2.000	0.109	1.782	2.21	51,800	56,980	8,320	7,490	2,149	2,364	3.89	3.08
2.375	0.125	2.125	3.01	70,690	77,750	7,950	7,160	3,463	3,809	5.48	4.39
2.625	0.134	2.357	3.57	83,890	92, 280	7, 740	6,970	4,571	5,028	6.69	5.40
2.875	0.156	2.563	4.54	106,600	117,260	8,240	7,420	6,330	6,963	8.03	6.38
3.250	0.156	2.938	5.17	121,310	133,440	7,290	6,560	8,236	9,060	10.26	8.39
3.500	0.175	3.150	6.23	146,240	160,870	7,630	6,870	10,708	11,778	11.90	9.64
4.500	0.224	4.052	10.25	240,730	264,800	7,610	6,850	22,639	24,962	19.67	15.95
5.000	0.276	4.448	13.96	327,690	360,460	8,510	7,660	34,231	37,654	24.29	19.22

Figure 2.5.4: Current sizes and material grades of industry coiled tubing.*Introduction to Permanent Plug and Abandonment of Wells* by Mahmoud Khalifeh and Arild Saasen (2020), Chapter 7, p. 196

(This page intentionally left blank)

CHAPTER

THREE

METHODS

3.1 Scenario Overview for Cement Plugging

This section of our software application focuses on a critical operation in well abandonment – the placement of the initial cement plug within the pay sand range. This layer, often rich in hydrocarbons, is crucial for the integrity of well abandonment processes. Our approach, while deviating slightly from conventional practices, mirrors the complexities frequently encountered in field operations. It adheres to the principles outlined in NORSOK standards, emphasizing the maintenance of well integrity and effective isolation of the reservoir.

This particular scenario was selected due to its technical and environmental significance. Accurate placement of the cement plug in the pay sand range is vital for preventing potential hydrocarbon leakage and aligning with environmental safety standards and regulatory compliance. The scenario's implementation in our software showcases its capability to handle nuanced and industry-relevant challenges, demonstrating both the flexibility and precision of our application in real-world contexts. This scenario not only tests the software's functional boundaries but also aligns with the broader objectives of this thesis, which include developing a tool that is both technically robust and applicable in practical well abandonment scenarios.

3.1.1 Interactive Diagram as a Methodological Tool

Central to this scenario is the use of an interactive diagram integrated within the software. This diagram serves a dual purpose: firstly, as a visual aid for understanding the complex structure of the well, and secondly, as an intuitive interface for data input.

3.1.2 Functionality and User Interaction

- Visualizing the Well Structure: The diagram illustrates key elements such as the pay sand layer and the casing shoe. This visual representation helps users grasp the geological and structural aspects of the well.

- Data Input through Diagram: Users interact with the diagram to input

essential parameters. For instance, they can specify the depth of the pay sand layer and the positioning of the cement plug. This method of input is designed to be user-friendly and reduce the likelihood of data entry errors.

- Scenario Execution: Utilizing the diagram, users apply the specifics of their scenario — like the depth of the pay sand layer — and then engage the software to calculate the optimal placement and volume of the cement plug.

3.1.3 Advantages of the Diagram Approach

The software includes a special diagram that helps users in two important ways. First, it makes the cement plugging process easier to understand by showing a simple picture of the well. This helps users see what needs to be done more clearly. Second, because the diagram is clear and easy to use, it helps users put in the right information. This is really important to make sure the cement plug goes exactly where it should in the well. All these features make the software not only easier to use but also more effective for the job it's designed to do.

3.2 Programming Language and Tools

The initial phase of planning centered on selecting the most suitable programming language that would offer flexibility for future enhancements and the potential for integration with a variety of systems. The selection process was influenced by the need for a robust solution capable of evolving alongside market demands.

3.2.1 MATLAB Consideration and Shift to Python

While MATLAB's specialized capabilities for matrix computations and simulations made it a contender, it was Python's versatility and interoperability that ultimately tipped the scales in its favor. The decision to pivot to Python was reinforced by discussions with academic advisors, highlighting Python's widespread adoption and its expansive array of libraries which enable extensive future development.

3.2.2 Advantages of Python

Python, with its open-source status and vast community support, emerged as the optimal choice. Its ability to integrate with other programming languages and frameworks ensures that the project remains adaptable and extensible, a crucial feature for keeping pace with continuous market evolution.

3.2.3 GUI Development Libraries: Tkinter to PyQt5

Initial considerations for the GUI included various libraries, with Tkinter being a primary candidate due to its simplicity and ease of use. However, it became apparent that Tkinter might not possess the robustness required for more complex or feature-rich interface development. This prompted a pivot towards PyQt5, renowned for its strength and flexibility in creating sophisticated GUIs.

3.2.4 Embracing PyQt5

PyQt5, a set of Python bindings for the Qt application framework, offered a more advanced suite of tools compared to its counterparts. The decision to adopt PyQt5 was motivated by its comprehensive features such as intricate graphics handling, and the ability to customize components extensively, which are indispensable for a professional and modern application.

3.2.5 Cross-platform Compatibility

In addition to these capabilities, the combination of Python and PyQt5 ensures cross-platform operability, allowing the application to function seamlessly across various operating systems—this universality is a significant asset for broad deployment.

3.2.6 Conclusion

The strategic decision to commit to Python, and specifically PyQt5 for GUI development, lays down a solid technological foundation for both current and future project requirements. This foresightful approach not only fulfills the technical prerequisites but also secures a path for continuous improvement and adaptation in response to the dynamic landscape of industry needs.

3.3 Design of the GUI

3.3.1 Interface Planning and User Experience

Creating a user-friendly and visually appealing graphical user interface (GUI) was paramount in our design process. We aimed for a design that would provide an intuitive experience, allowing users to navigate the application's features with ease and efficiency.

3.3.2 Utilizing Qt Designer

To ensure the GUI met these standards, we employed Qt Designer, a powerful tool within the PyQt5 ecosystem. Qt Designer enables the creation of sophisticated user interfaces through a drag-and-drop interface, offering a real-time preview of how the GUI will appear to the end-user.

3.3.3 Wireframing and Prototyping

The design phase was initiated with wireframing, which laid out the structural blueprint for the GUI. Utilizing Qt Designer, we transformed these wireframes into interactive prototypes. This approach allowed us to visualize and refine the positioning of elements such as buttons, input fields, and data display areas before finalizing the layout.

3.3.4 Design Principles

Throughout the design process, we adhered to established design principles, focusing on a clean and organized layout that avoids clutter while maximizing usability. The selection of colors, fonts, and icons was purposefully made to create a harmonious and accessible interface.

3.3.5 From Design to Code

After finalizing the designs in Qt Designer, we converted the visual layouts into Python code by exporting them as .py files. This conversion facilitated the seamless integration of the GUI design with the underlying application logic.

3.3.6 Customization and Functionality

In Qt Designer, we could efficiently customize widgets and define their properties, which later translated into functional elements within the PyQt5 framework. Event handlers and layout managers were configured to ensure the GUI was not only aesthetically pleasing but also responsive to user actions.

3.3.7 User-Centric Design Approach

Our design process was iterative and user-centric. Each prototype underwent rigorous user testing to gather feedback on usability and aesthetics. The insights gained from this feedback were crucial in refining the GUI to ensure it was aligned with user requirements.

3.3.8 Conclusion

The use of Qt Designer was instrumental in developing the GUI for our application. By bridging the gap between visual design and functional code, we were able to create an interface that is both appealing and user-friendly. The GUI we developed stands as a testament to the synergy between well-planned design and technical functionality, ensuring that our application is not only powerful but also a pleasure to use.

3.4 Implementation

Our software implements a multi-tabbed graphical user interface (GUI) using PyQt5, structured to facilitate the plug and abandonment process in oil and gas wells. This section details the implementation of each tab, providing the user with both input and computational functionalities.

3.4.1 Tab 1: Well Information

In the initial tab of the software, users are greeted with fields to input essential information about the well (Fig. 3.4.1). This includes the well's name, which in this case is "6503", its type (vertical), and the reservoir characteristics (sandstone).

Here, users can also define the objective and scope of the operation, which is to place a cement plug in the "6503" well. Additionally, this section offers space to outline the planned events and activities associated with the project, ensuring a clear understanding of the objectives and the sequence of actions.

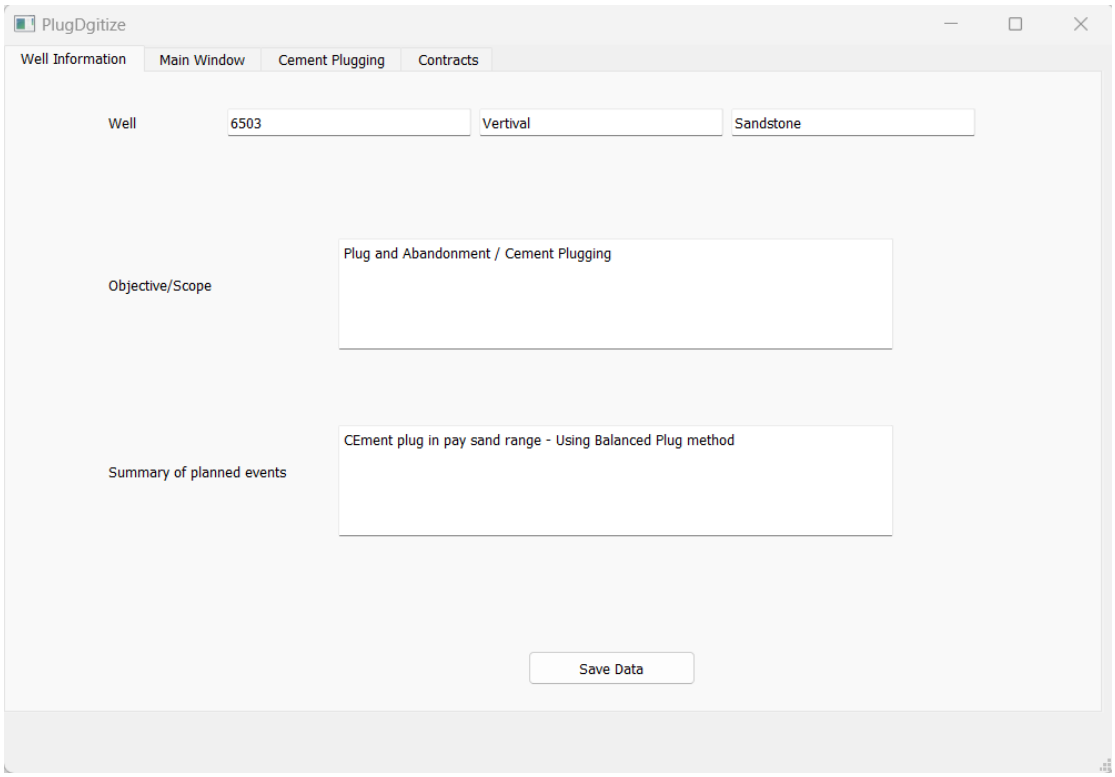


Figure 3.4.1: Tab 1: Well Information

3.4.2 Tab 2: Main Window

Moving to the next tab, users are directed to select the appropriate discipline and techniques for their operation (Fig. 3.4.2). Since the software currently only supports the Plug and Abandonment discipline, this will be pre-selected with a focus on deep plugging. The objectives and events are intricately linked, with the selection of an objective dynamically updating the events combo box to display relevant options. Upon selecting 'Cement Plug' as the objective, the software presents a range of cement plugging techniques to choose from. This tab is dedicated to cement plugging as the sole engineering subject. Further details regarding contracts and associated procedures will be addressed in subsequent sections of the software.

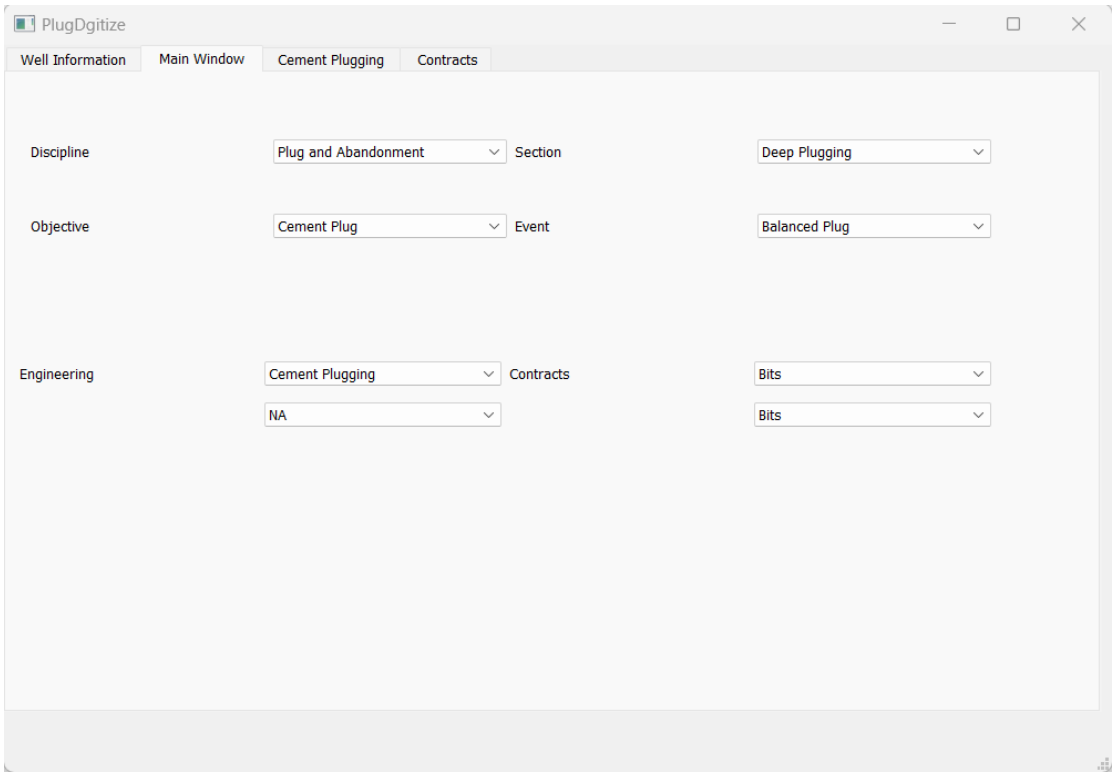


Figure 3.4.2: Tab 2: Main Window

3.4.3 Tab 3: Cement Plugging

The third tab is central to the operation, encompassing the calculation aspects of the project. It opens up to reveal five sub-tabs, each dedicated to a specific aspect of the calculations. The first sub-tab provides an overview of the software, its purpose, and a primer on the calculations and equations that will be used (Fig. 3.4.3).

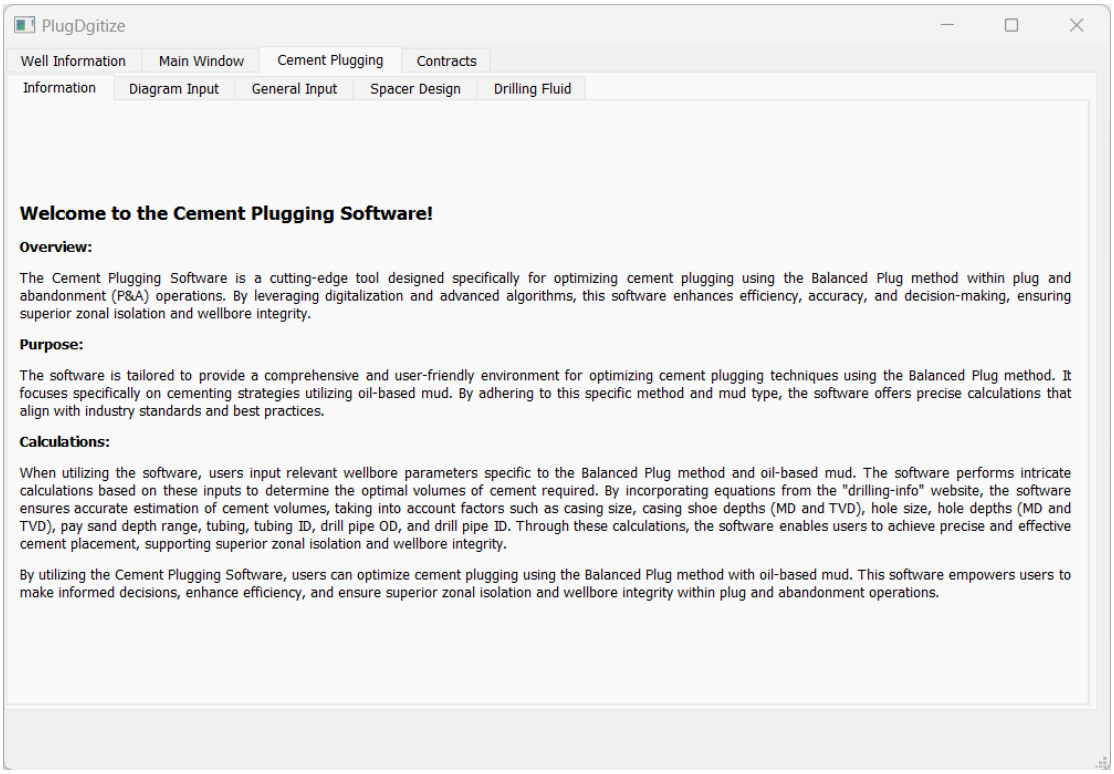


Figure 3.4.3: Tab 3: Cement Plugging Calculator's Information Sub-tab

The second sub-tab is designed to simulate the specific scenario. In our example, it details the 6503 well, with parameters like the pay sand range (9000 ft to 9500 ft), measured depth (MD) of 12000ft, true vertical depth (TVD) of 10000ft, and other critical measurements such as the cement placement depth, top of cement (TOC), and stinger dimensions as shown in Fig. 3.4.4.

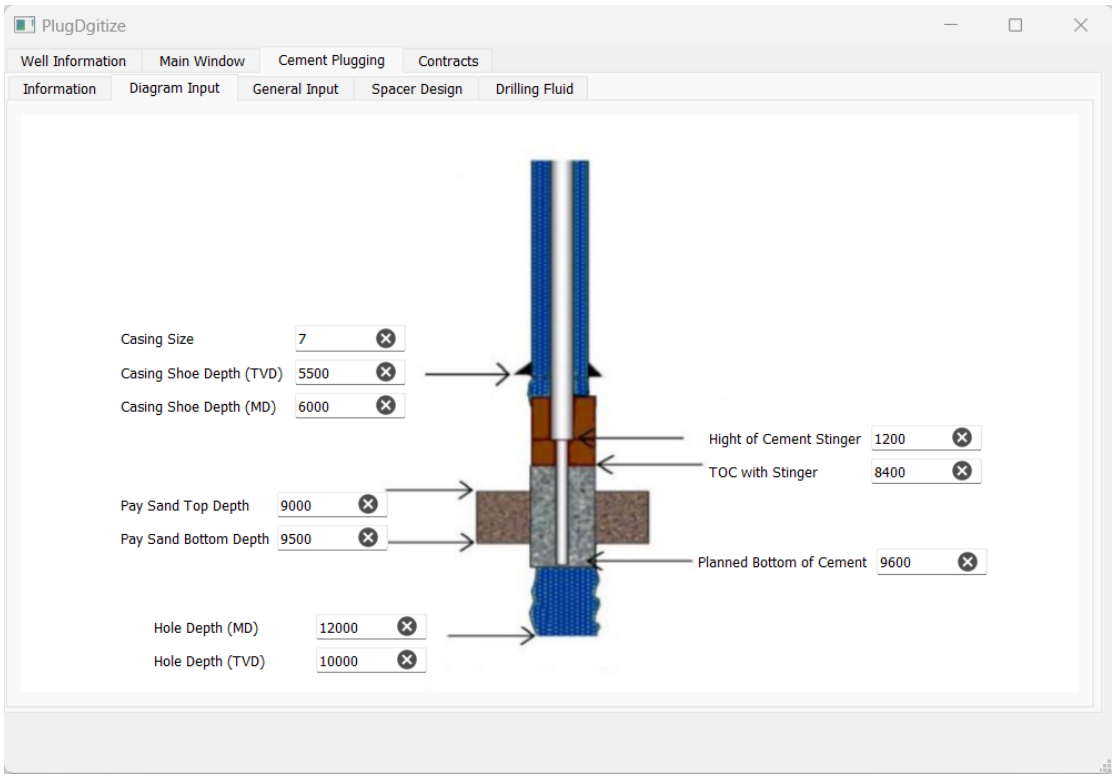


Figure 3.4.4: Diagram with Inputs for Cement Plugging Calculator

The third sub-tab requires additional information for precise calculations (Fig. 3.4.5). It asks for the inner and outer diameters of the tube (stinger) and drill pipe, the bit size (assumed to be the same as the hole size), and the length of the cementing operation. The density of the cement and a safety factor are also included in this sub-tab.

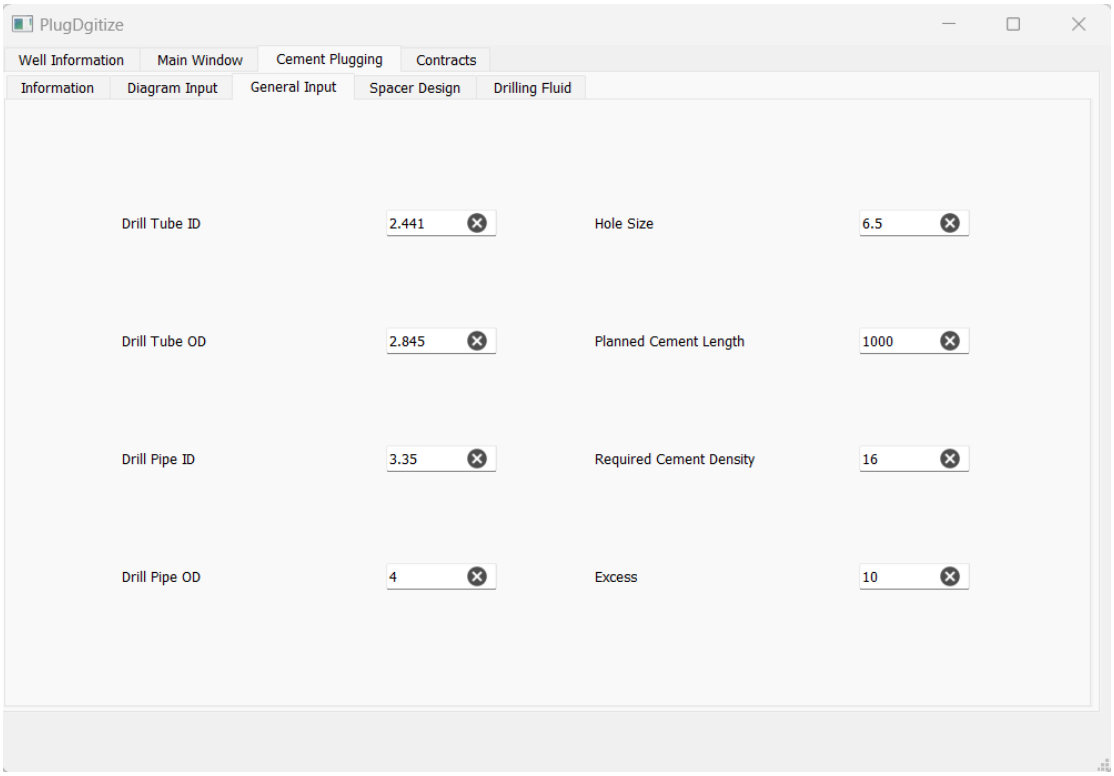


Figure 3.4.5: General Inputs for Cement Plugging Calculator

The following two sub-tabs focus on the specifics of the spacer design and the drilling fluid (mud) used. Users must input details such as the volume and density of the spacer in the fourth sub-tab (Fig. 3.4.6).

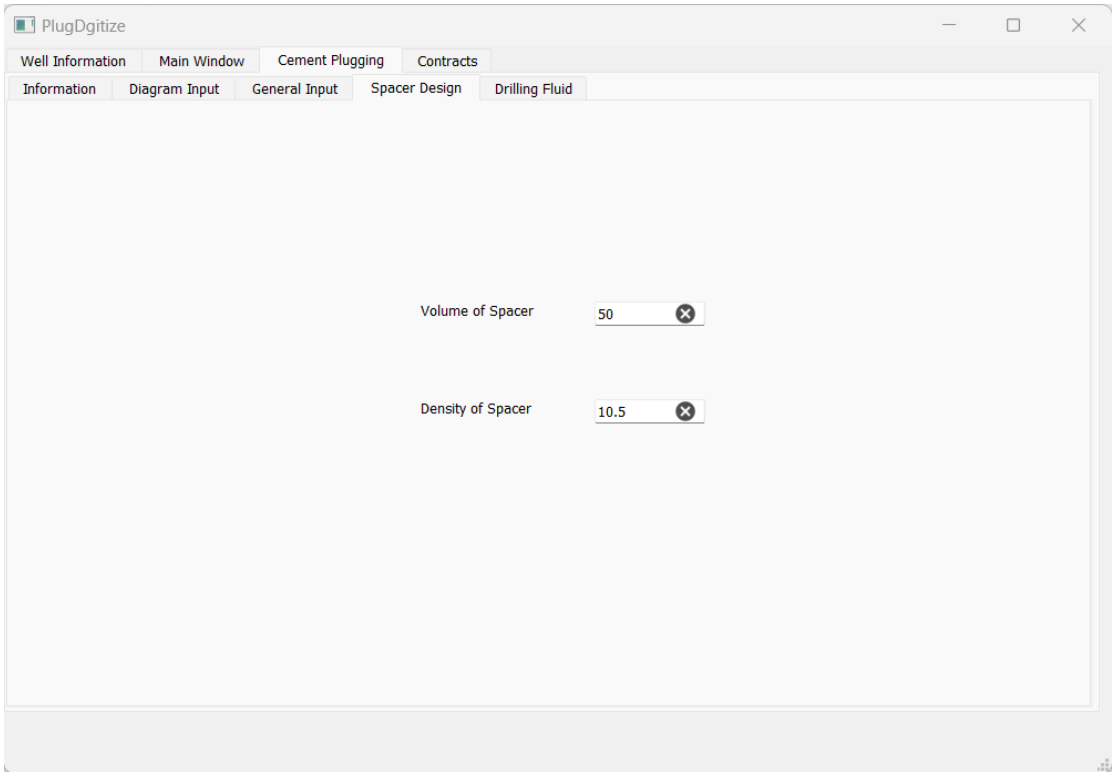


Figure 3.4.6: Inputs Related to Spacer for Cement Plugging Calculator

The final sub-tab allows users to specify drilling fluid properties, including type and density (Fig. 3.4.7). A customizable combo box provides flexibility for users to define and recall specific mud codes.

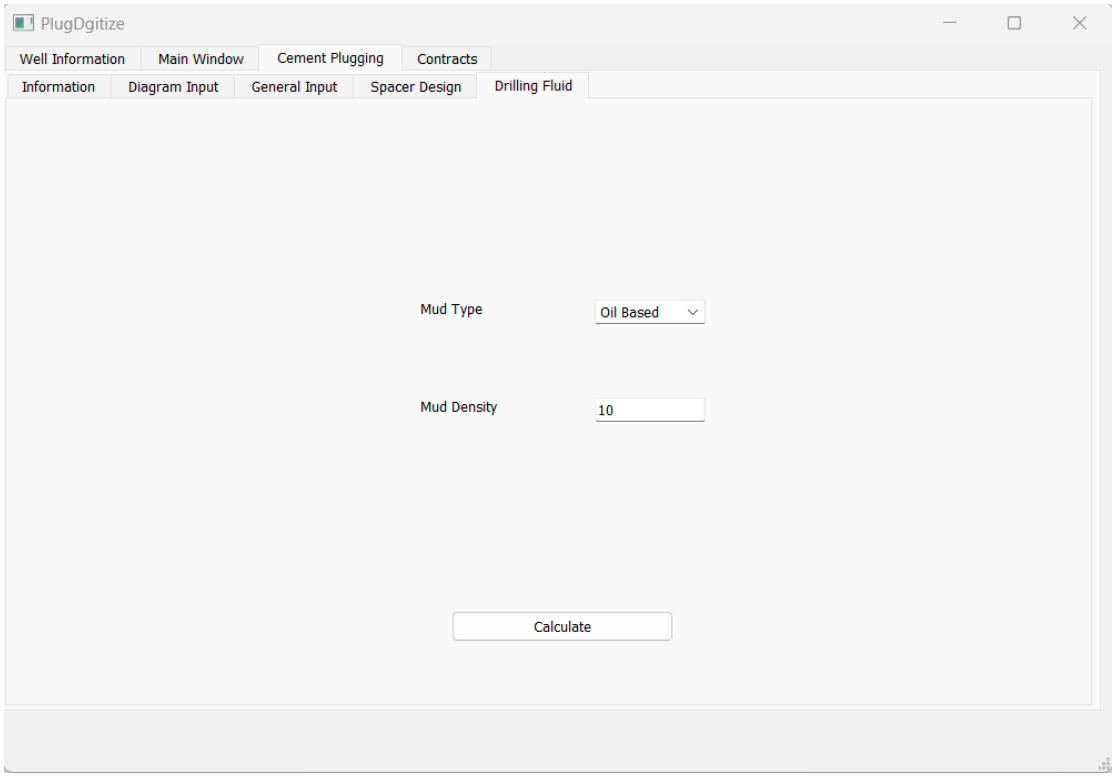


Figure 3.4.7: Inputs Related to Drilling Fluid Specifications for Cement Plugging Calculator

Additionally, in the final sub-tab, there is a 'calculate' button. Once all inputs are entered, pressing the 'Calculate' button brings up a new window displaying the outputs (Fig. 3.4.8). These outputs are categorized into several series: capacities, cement data (including volume and height of cement with the cement string in the well, etc.), spacer-related data, and the volume of mud needed for displacement. The final series of outputs presents a simple schedule for pumping, guiding the engineer through each step of the process with calculated numbers. This includes detailed instructions like the volume of spacer and cement to pump and the sequence of introducing new drilling fluid. The software also advises underdisplacing a small volume of mud to facilitate the cement falling effect in the drill pipe.

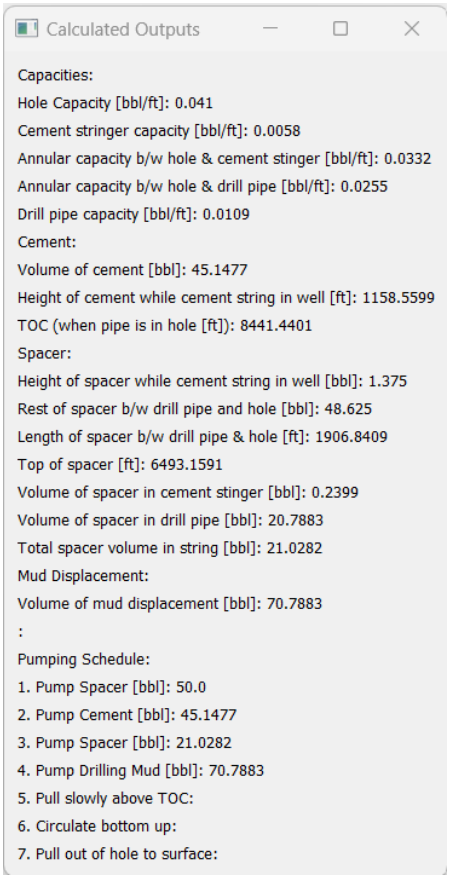


Figure 3.4.8: Outputs and a simple pumping schedule

3.4.4 Tab 4: Contracts

Currently a placeholder for future enhancements, this tab will eventually contain detailed contracts information. The vision is to develop this area into a comprehensive contract management system that could potentially facilitate automated ordering and service company engagement, streamlining the plug and abandonment process.

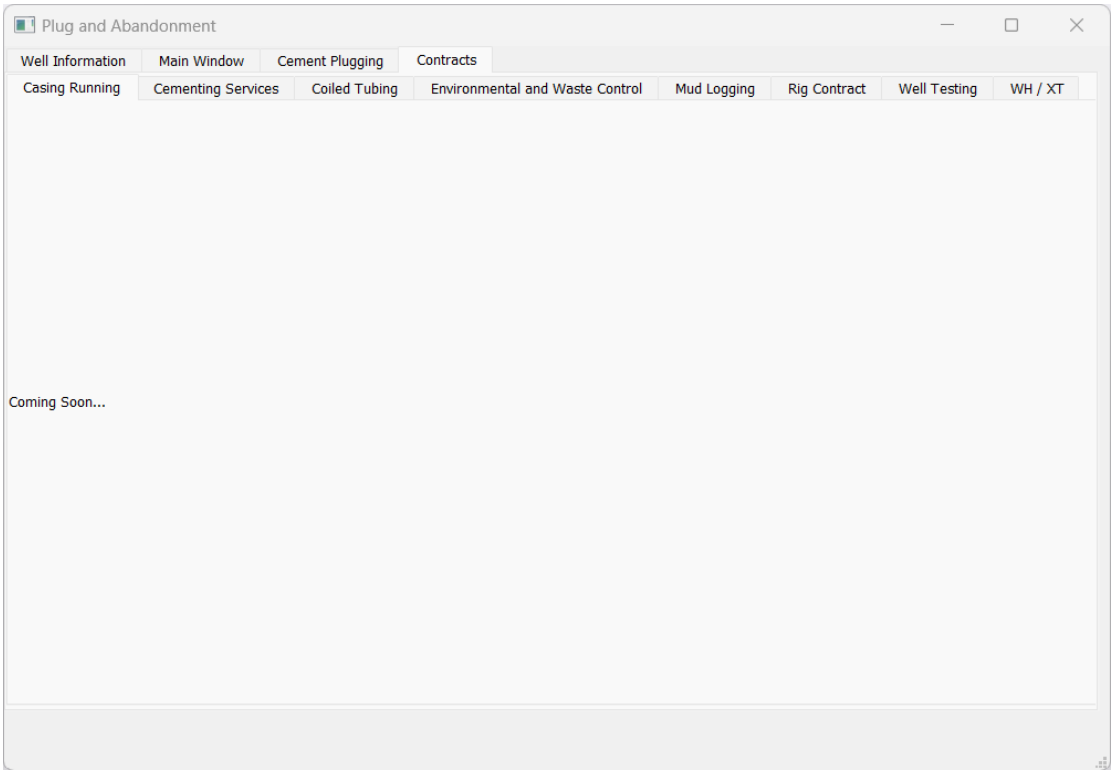


Figure 3.4.9: Tab 4: Contracts

3.4.5 Event Handling and Computation

The GUI is designed to be responsive, with PyQt5 signals and slots managing user interactions. For example, the objectives and events combo boxes are interconnected, with the selection in one affecting the available choices in the other.
The 'Calculate' button within the Cement Plugging tab activates a series of computations based on user inputs, employing equations and algorithms to determine the specifics of the cement plugging operation.

3.4.6 Conclusion

The implementation of this software is characterized by a user-centric approach with an intuitive GUI that organizes complex well abandonment operations into manageable segments. Through thoughtful design and structured programming, the software serves as an effective tool for professionals in the oil and gas industry, with ample scope for future development and integration of advanced features.

3.5 Summary of Methods

In conclusion, the development of our software for the digitalization of plugs and the abandonment of wells has been methodically detailed in this section. Utilizing Python for its flexibility and compatibility, complemented by PyQt5 for its robust GUI capabilities, we have constructed an application tailored to the needs of plug and abandonment operations in the oil and gas industry.

The software's architecture is centered around a user-friendly, tabbed GUI, which facilitates efficient data management and operation:

- Well Information Tab: This tab allows for the comprehensive input of well data, including general information, objectives, and summaries of planned events, ensuring all crucial details are recorded and easily accessible.
- Main Window Tab: It features interactive combo boxes for selecting discipline and section, with a focus on plug and abandonment operations. The application dynamically updates the event options based on the objective selected, showcasing its responsive design.
- Cement Plugging Tab: Dedicated to the engineering calculations for cement plugging, this tab divides the process into various sub-sections for inputting well parameters, spacer fluid properties, cement specifications, and drilling fluid details. The 'Calculate' button here stands out as a central feature, initiating the computation of crucial cementing parameters.
- Contracts Tab: Currently conceptual, this section is envisaged to evolve into a comprehensive module for managing contracts and automating service engagements, highlighting the potential for future development and automation in the software.

Our implementation strategy emphasizes modularity and extendability, ensuring that the software is not only effective in its current state but also poised for future enhancements. The software is designed to be intuitive, with a keen focus on a user-friendly experience, while maintaining the adaptability to meet evolving market demands. This combination of thoughtful design, responsive UI, and comprehensive functionality underscores the software's potential as a valuable tool in the field of oil and gas well abandonment.

CHAPTER

FOUR

DISCUSSION

4.1 Interpretation of Findings

The outcome of the developed software underscores its ability to increase efficiency and significantly reduce both time and costs involved in the plug and abandonment process. The framework established by this research lays a foundational pathway for future enhancements, potentially leading to a fully automated process. These results align perfectly with the initial objective of the project, which focused on enhancing well abandonment operations through digital solutions. The approach, complemented by a user-friendly interface, has demonstrated its potential in reducing human error and streamlining the abandonment process. This alignment not only validates the effectiveness of the software but also highlights the importance of integrating digital tools in traditionally manual operations

4.2 Methodological Considerations

The research methodology for this project was twofold, encompassing both a comprehensive literature review and extensive technical development work. Initially, the scope of work was defined through an in-depth analysis of existing literature, setting a solid foundation for the subsequent phases of the project. This process involved identifying current methodologies, challenges, and gaps in the field of well abandonment, which directly informed the development of PlugDigitize. The technical aspect of the research was then undertaken, utilizing programming skills to transform the conceptual framework into a functional software tool. Despite these strengths, the study faced certain limitations. The most notable constraint was the time factor, which inevitably restricted the depth of development and testing that could be achieved within the project timeline. This limitation is a common challenge in cutting-edge research areas, where the novelty of the subject means that there is a relative scarcity of comprehensive data and extensive prior studies to draw upon. As such, while the study has made significant strides in the field of digital solutions for well abandonment, it represents just the beginning of a broader, more in-depth exploration that is needed. Future research, with the benefit of more time and potentially more resources, could delve deeper into this topic, expanding upon the foundational work laid out by PlugDigitize and

exploring new avenues for innovation and application in the industry

4.3 Innovation and Technological Impact

As the need for digitalization and automation has become increasingly evident after covid situation, finding digital solution for traditional operation is a must. PlugDigitize stands as a testament to the potential of digital innovation in traditional oil and gas operations. Its development not only demonstrates the feasibility of such digital tools but also underscores the broader impact technology can have on enhancing efficiency and safety in the sector. More than just a functional software, PlugDigitize exemplifies the transformative impact that technology can have in enhancing operational efficiency and safety in a sector that is traditionally reliant on manual processes. The current trend in industry research predominantly focuses on the automation of the drilling process. However, there is an evident need for auxiliary software systems that can supplement these automated rigs, especially for other critical stages in an oil well's lifecycle. This is particularly true for the plug and abandonment stage, which is often the final and a financially taxing phase of a well's life. During this stage, costs are incurred at a time when the well ceases to contribute to the company's profitability, thus representing an additional financial burden. PlugDigitize addresses this challenge by offering a digital solution to streamline and reduce the expenses associated with well abandonment. Its role in filling this gap is crucial, as it not only aids in the current focus on drilling automation but also extends its benefits to the end-of-life stages of well operations, thereby enhancing the overall cost-effectiveness and efficiency of oil and gas industry practices.

4.4 Economic Analysis

From an economic perspective, the implementation of PlugDigitize marks a significant advancement over traditional methods used in well abandonment processes. One of the primary benefits of this digital tool is its ability to reduce the time required for planning and executing the calculations necessary for well abandonment. By streamlining these processes, PlugDigitize not only saves valuable hours but also minimizes the resources expended in these operations. This efficiency translates into substantial long-term financial savings for companies, a benefit that becomes increasingly pronounced in large-scale operations. Additionally, by creating an environment that is intuitive and easy to navigate, the PlugDigitize significantly reduces the learning curve and operational time for users. This aspect is particularly beneficial in reducing the time taken to understand complex situations and determine the next steps in the abandonment process. Consequently, the reduced time in decision-making and execution directly impacts the overall operational costs, making PlugDigitize an economically advantageous solution in the oil and gas industry. This dual advantage of cost and time efficiency positions PlugDigitize not only as a technological innovation but also as a tool with significant economic implications for the future of well abandonment.

CHAPTER

FIVE

CONCLUSIONS

This thesis has successfully developed a comprehensive digital framework, which called "PlugDigitize", aimed at revolutionizing well abandonment operations in the oil and gas industry. The primary objective was to integrate digital solutions to enhance efficiency, reduce costs, and ensure environmental compliance and safety in well abandonment processes. Through meticulous research and practical application, the study has demonstrated the potential of digitalization to streamline well abandonment operations, highlighting significant advancements over traditional methods.

Key achievements of this research include the development of a user-friendly software interface that simplifies complex operational processes. This interface not only enhances operational efficiency but also minimizes the risk of human error, a critical factor in ensuring safety and compliance with industry standards. The use of Python and PyQt5 has been instrumental in achieving this, providing a robust and adaptable platform that meets the industry's evolving needs.

The environmental implications of this research are significant. By facilitating more accurate and efficient well abandonment operations, the project contributes to the prevention of hydrocarbon leakage, thereby protecting ecosystems and supporting sustainable industry practices.

Moreover, the economic benefits of this research are evident. The digital solutions developed offer a cost-effective alternative to traditional well abandonment processes, crucial in an industry characterized by fluctuating market dynamics.

The study opens avenues for further research and development. Future work could focus on integrating advanced data analytics and machine learning algorithms to predict and optimize well abandonment strategies. Additionally, exploring the scalability of the framework to accommodate varying well conditions and regulatory environments globally would enhance its applicability and impact.

In conclusion, this thesis represents a significant stride towards the digital transformation of well abandonment processes in the oil and gas industry. It not only addresses the current challenges faced by the industry but also lays a foundation for future innovations, aligning with the global shift towards digitalization and sustainability.

(This page intentionally left blank)

CHAPTER

SIX

FUTURE WORK

- **Contract Management:** This subsection will focus on developing features within "PlugDigitize" for managing rig contracts more efficiently. The software will be designed to analyze different rig contracts and suggest appropriate personnel for each duty. It will also have the capability to automatically generate contracts and initiate contact with operators relevant to specific contracts. Additionally, the software will provide a procedural outline for each event related to a contract, enhancing operational efficiency and accuracy. This feature aims to streamline the contractual and operational aspects of well abandonment, ensuring a more integrated and automated approach.

- **Various Scenarios:** Future development of "PlugDigitize" will also focus on incorporating diverse scenarios based on NORSOK regulations. The software, exemplified through a specific scenario in this thesis, will be enhanced to include various other scenarios, adapting to different operational contexts. Additionally, it will be designed to handle different section plugs, such as shallow or middle plugs, ensuring compliance with varied regulatory requirements. The software's adaptability will also extend to real-time data monitoring, enabling it to detect any issues like leakage in wells and alert users promptly. This feature is aimed at ensuring that operations not only comply with regulations but are also proactive in identifying and addressing potential problems.

- **Economic Analysis and Cost Prediction:** An important future enhancement for "PlugDigitize" is the inclusion of an economic analysis feature. This function will enable the software to not only plan the entire well abandonment process but also to analyze and predict associated costs. By assessing various operational parameters and market conditions, the software will provide companies with detailed cost projections, helping them to make informed financial decisions. This capability will be crucial for optimizing budget allocation and enhancing the economic feasibility of well abandonment projects.

- **Integration of Machine Learning:** A significant future development for "PlugDigitize" involves leveraging machine learning algorithms to predict the sequence of operations and foresee potential challenges. This predictive

capability will not only streamline the procedural workflow but also anticipate future scenarios, enhancing strategic planning. Moreover, machine learning can play a pivotal role in the software's economic analysis component, offering more accurate cost predictions and financial insights based on complex data patterns and historical trends. This integration will significantly elevate the software's effectiveness in both operational and financial decision-making processes.

REFERENCES

[1] Bjorn Brechan et al. "Interactive Experience and Learning Model can Reduce Non-Productive Time NPT". In: *Offshore Technology Conference Asia*. OTC. 2018, D041S039R001.

[2] P. Vaghela. "How Digital Transformation in the Oil and Gas Industry Streamlines Operation". In: *Bigscal - Software Development Company* (Sept. 2023). URL: https://www.bigscal.com/blogs/oil-and-gas-industry/digital-transformation-in-oil-and-gas-industry/.

[3] Standard Norge. *NORSOK Standard D-010: Well Integrity in drilling and well operations*. Retrieved from: https://www.npd.no/globalassets/1-npd/regelverk/skjema/bronnregistreing/eng/norsok-d-010-2013-well-integrity-andwell-operations-rev-4.pdf. Jan. 2021.

[4] Lewaa Hmadeh. "The Beginning of the End: A Digital Planning of P&A Operations". Master's thesis. Petroleum Engineering, Norwegian University of Science and Technology, June 2021.

[5] Mahmoud Khalifeh and Arild Saasen. "Fundamentals of Plug Placement". In: *Introduction to Permanent Plug and Abandonment of Wells*. Cham: Springer International Publishing, 2020, pp. 185–212. ISBN: 978-3-030-39970-2. DOI: 10.1007/978-3-030-39970-2_7. URL: https://doi.org/10.1007/978-3-030-39970-2_7.

[6] G. Pierce. "Subsea Plugging & Abandonment: Light Well Intervention Vessel Overview". In: *Drillers* (Mar. 2021). URL: https://drillers.com/subsea-plugging-abandonment-light-well-intervention-vessel-overview/.

[7] Tech A. Updates. "Develop Custom Python pyqt5". In: *Medium* (Nov. 2023). URL: https://medium.com/@aiupdate/develop-custom-python-pyqt5-e1ec06b82cbd.

[8] I. I. Coach. "Top 25 PyQt5 Interview Questions and Answers - InterviewPrep". In: *InterviewPrep* (Aug. 2023). URL: https://interviewprep.org/pyqt5-interview-questions/.

[9] *An Electrifying Integrated Solution Towards a Safe and Environmentally Sound Well Abandonment in Urban Setting*. Vol. Day 3 Thu, March 24, 2022. Offshore Technology Conference Asia. D031S019R006. Mar. 2022. DOI: 10.4043/31394-MS. eprint: https://onepetro.org/OTCASIA/proceedings-pdf/22OTCA/3-22OTCA/D031S019R006/2662280/otc-31394-ms.pdf. URL: https://doi.org/10.4043/31394-MS.

[10] S. Minev. "Exploring the Power of PyQt5: Building Interactive GUI Applications". In: *Medium* (Aug. 2023). URL: https://medium.com/@stefanminev_54009/exploring-the-power-of-pyqt5-building-interactive-gui-applications-4718529a902e.

[11] PetroWiki. *Well Integrity*. https://petrowiki.spe.org/Well_integrity. Nov. 2022.

[12] Marcelo Aviles. "Technology Innovation and Adoption in Oil & Gas Industry – Why Did It Slow?" In: *Forbes* (July 2015). URL: https://www.forbes.com/sites/drillinginfo/2015/07/14/technology-innovation-andadoption-in-oil-gas-industry-why-did-it-slow/?sh=ab60d353d1c9.

[13] Silvio Marcacci. "Plugging Abandoned Oil Wells Is One 'Green New Deal' Aspect Loved by Both Republicans and Democrats". In: *Forbes* (Sept. 2020). URL: https://www.forbes.com/sites/energyinnovation/2020/09/21/plugging-abandoned-wellsthe-green-new-deal-jobs-plan-republicans-and-democrats-love/?sh=5ec60df2e100.

[14] Bjørn Brechan. "Framework for automated well planning and Digital Well Management". Thesis for the Degree of Philosophiae Doctor. Nov. 2020. URL: https://ntnuopen.ntnu.no/ntnu-xmlui/handle/11250/2686548.

[15] Bjørn Brechan et al. "Work Process and Systematization of a New Digital Life Cycle Well Integrity Model". In: *SPE Norway One Day Seminar*. Bergen, Norway, Apr. 2018. DOI: 10.2118/191299-MS.

[16] C.R. Clark and G.L. Carter. "Mud displacement with cement slurries". In: *Journal of Petroleum Technology* 25.07 (1973), pp. 775–783. DOI: 10.2118/4090-PA. URL: https://doi.org/10.2118/4090-PA.

[17] A. Shadravan et al. "Engineering the mud-spacer-cement rheological hierarchy improves wellbore integrity". In: *SPE E&P Health, Safety, Security and Environmental Conference-Americas*. Society of Petroleum Engineers. Denver, Colorado, USA, 2015. DOI: 10.2118/173534-MS. URL: https://doi.org/10.2118/173534-MS.

[18] D. Denney. "Rheological targets for mud removal and cement-slurry design". In: *Journal of Petroleum Technology* 53.08 (2001), pp. 65–66. DOI: 10.2118/0801-0065-JPT.

[19] J. Jakobsen et al. "Displacements in eccentric annuli during primary cementing in deviated wells". In: *SPE Production Operations Symposium*. Society of Petroleum Engineers. Oklahoma City, Oklahoma, USA, 1991. DOI: 10.2118/21686-MS.

[20] C.W. Sauer. "Mud displacement during cementing state of the art". In: *Journal of Petroleum Technology* 39.09 (1987), pp. 1091–1101. DOI: 10.2118/14197-PA.

[21] T.R. Smith. "Cementing displacement practices: application in the field". In: *SPE/IADC Drilling Conference*. Society of Petroleum Engineers. New Orleans, Louisiana, 1989. DOI: 10.2118/18617-MS.

[22] S. Regan, J. Vahman, and R. Ricky. "Challenging the limits: Setting long cement plugs". In: *SPE Latin American and Caribbean Petroleum Engineering Conference*. Society of Petroleum Engineers. Port-of-Spain, Trinidad and Tobago, 2003. DOI: 10.2118/81182-MS.

[23] R.M. Beirute. "All-purpose cement-mud spacer". In: *SPE Symposium on Formation Damage Control*. Society of Petroleum Engineers. Houston, Texas, 1976. DOI: 10.2118/5691-MS.

[24] Renato Rios and Francois Ars. "Plug and Abandonment Materials-Technology Landscape". In: *Offshore Technology Conference*. OTC. 2021, D041S051R007.

[25] Paul Carragher and Jeff Fulks. "Well abandonment solutions utilizing bismuth and thermite". In: *Abu Dhabi International Petroleum Exhibition and Conference*. SPE. 2018, D021S031R004.

[26] B Brechan, SI Dale, and S Sangesland. "Digital Well Planning-New Cost Saving Well Construction and Life Cycle Well Integrity Model". In: *Offshore Technology Conference*. OTC. 2018, D031S031R005.

(This page intentionally left blank)

APPENDICES

.1 Python Codes

```
from PyQt5 import QtCore, QtWidgets, QtGui
from PyQt5.QtWidgets import  QWidget, QLabel, QVBoxLayout,
    QMainWindow, QMessageBox
from collections import OrderedDict
import sqlite3
import sys

conn = sqlite3.connect('mylist.db')

c = conn.cursor()

c.execute("""CREATE TABLE if not exists well_list(
    list_item text)
    """)

conn.commit()

conn.close()

class OutputWindow(QMainWindow):
    def __init__(self, output_values):
        super().__init__()
        self.setWindowTitle('Calculated Outputs')
        self.setGeometry(200, 200, 500, 500)

        # Create a central widget and layout
        self.central_widget = QWidget()
        self.central_layout = QVBoxLayout(self.central_widget)

        # Create labels to display the output values
        for label, value in output_values.items():
            output_label = QLabel("{}: {}".format(label, value))
            self.central_layout.addWidget(output_label)

        self.setCentralWidget(self.central_widget)
        self.adjustSize()



class Ui_Software(object):
    def setupUi(self, Software):
```

```
        Software.setObjectName("Software")
        Software.resize(1000, 655)
        Software.setMinimumSize(QtCore.QSize(1000, 500))
        self.centralwidget = QtWidgets.QWidget(Software)
        self.centralwidget.setObjectName("centralwidget")
        self.tabWidget = QtWidgets.QTabWidget(self.centralwidget)
        self.tabWidget.setGeometry(QtCore.QRect(0, 0, 991, 601))
        self.tabWidget.setTabPosition(QtWidgets.QTabWidget.North)
        self.tabWidget.setMovable(True)
        self.tabWidget.setObjectName("tabWidget")
        self.tab_3 = QtWidgets.QWidget()
        self.tab_3.setObjectName("tab_3")
        self.SaveButton = QtWidgets.QPushButton(self.tab_3)
        self.SaveButton.setGeometry(QtCore.QRect(470, 520, 151,
    31))
        self.SaveButton.setObjectName("SaveButton")
        self.layoutWidget = QtWidgets.QWidget(self.tab_3)
        self.layoutWidget.setGeometry(QtCore.QRect(300, 60, 2, 2))
        self.layoutWidget.setObjectName("layoutWidget")
        self.horizontalLayout = QtWidgets.QHBoxLayout(self.
    layoutWidget)
        self.horizontalLayout.setContentsMargins(0, 0, 0, 0)
        self.horizontalLayout.setObjectName("horizontalLayout")
        self.layoutWidget1 = QtWidgets.QWidget(self.tab_3)
        self.layoutWidget1.setGeometry(QtCore.QRect(80, 20, 851,
    470))
        self.layoutWidget1.setObjectName("layoutWidget1")
        self.verticalLayout = QtWidgets.QVBoxLayout(self.
    layoutWidget1)
        self.verticalLayout.setContentsMargins(0, 0, 0, 0)
        self.verticalLayout.setObjectName("verticalLayout")
        self.frame_3 = QtWidgets.QFrame(self.layoutWidget1)
        self.frame_3.setFrameShape(QtWidgets.QFrame.StyledPanel)
        self.frame_3.setFrameShadow(QtWidgets.QFrame.Raised)
        self.frame_3.setObjectName("frame_3")
        self.horizontalLayout_2 = QtWidgets.QHBoxLayout(self.
    frame_3)
        self.horizontalLayout_2.setObjectName("horizontalLayout_2"
    )
        self.label = QtWidgets.QLabel(self.frame_3)
        self.label.setMinimumSize(QtCore.QSize(100, 0))
        self.label.setMaximumSize(QtCore.QSize(100, 16777215))
        self.label.setObjectName("label")
        self.horizontalLayout_2.addWidget(self.label)
        self.Nameline = QtWidgets.QLineEdit(self.frame_3)
        self.Nameline.setMinimumSize(QtCore.QSize(220, 25))
        self.Nameline.setMaximumSize(QtCore.QSize(220, 16777215))
        self.Nameline.setObjectName("Nameline")
        self.horizontalLayout_2.addWidget(self.Nameline)
        self.Typeline = QtWidgets.QLineEdit(self.frame_3)
        self.Typeline.setMinimumSize(QtCore.QSize(220, 25))
        self.Typeline.setMaximumSize(QtCore.QSize(220, 16777215))
        self.Typeline.setObjectName("Typeline")
        self.horizontalLayout_2.addWidget(self.Typeline)
        self.Resline = QtWidgets.QLineEdit(self.frame_3)
        self.Resline.setMinimumSize(QtCore.QSize(220, 25))
        self.Resline.setMaximumSize(QtCore.QSize(220, 16777215))
        self.Resline.setObjectName("Resline")
```

```
93          self.horizontalLayout_2.addWidget(self.Resline)
94          spacerItem = QtWidgets.QSpacerItem(40, 20, QtWidgets.
    QSizePolicy.Expanding, QtWidgets.QSizePolicy.Minimum)
95          self.horizontalLayout_2.addItem(spacerItem)
96          self.verticalLayout.addWidget(self.frame_3)
97          self.frame_4 = QtWidgets.QFrame(self.layoutWidget1)
98          self.frame_4.setFrameShape(QtWidgets.QFrame.StyledPanel)
99          self.frame_4.setFrameShadow(QtWidgets.QFrame.Raised)
100         self.frame_4.setObjectName("frame_4")
101         self.gridLayout_4 = QtWidgets.QGridLayout(self.frame_4)
102         self.gridLayout_4.setObjectName("gridLayout_4")
103         self.label_2 = QtWidgets.QLabel(self.frame_4)
104         self.label_2.setMinimumSize(QtCore.QSize(200, 0))
105         self.label_2.setMaximumSize(QtCore.QSize(200, 16777215))
106         self.label_2.setObjectName("label_2")
107         self.gridLayout_4.addWidget(self.label_2, 0, 0, 1, 1)
108         self.Sumtext = QtWidgets.QTextEdit(self.frame_4)
109         self.Sumtext.setMinimumSize(QtCore.QSize(500, 100))
110         self.Sumtext.setMaximumSize(QtCore.QSize(500, 100))
111         self.Sumtext.setObjectName("Sumtext")
112         self.gridLayout_4.addWidget(self.Sumtext, 1, 1, 1, 1)
113         self.label_3 = QtWidgets.QLabel(self.frame_4)
114         self.label_3.setMaximumSize(QtCore.QSize(200, 16777215))
115         self.label_3.setObjectName("label_3")
116         self.gridLayout_4.addWidget(self.label_3, 1, 0, 1, 1)
117         self.Objtext = QtWidgets.QTextEdit(self.frame_4)
118         self.Objtext.setMinimumSize(QtCore.QSize(500, 100))
119         self.Objtext.setMaximumSize(QtCore.QSize(500, 100))
120         self.Objtext.setObjectName("Objtext")
121         self.gridLayout_4.addWidget(self.Objtext, 0, 1, 1, 1)
122         spacerItem1 = QtWidgets.QSpacerItem(40, 20, QtWidgets.
    QSizePolicy.Expanding, QtWidgets.QSizePolicy.Minimum)
123         self.gridLayout_4.addItem(spacerItem1, 0, 2, 1, 1)
124         spacerItem2 = QtWidgets.QSpacerItem(40, 20, QtWidgets.
    QSizePolicy.Expanding, QtWidgets.QSizePolicy.Minimum)
125         self.gridLayout_4.addItem(spacerItem2, 1, 2, 1, 1)
126         self.verticalLayout.addWidget(self.frame_4)
127         self.tabWidget.addTab(self.tab_3, "")
128         self.tab = QtWidgets.QWidget()
129         self.tab.setObjectName("tab")
130         self.frame = QtWidgets.QFrame(self.tab)
131         self.frame.setGeometry(QtCore.QRect(10, 10, 891, 191))
132         self.frame.setFrameShape(QtWidgets.QFrame.StyledPanel)
133         self.frame.setFrameShadow(QtWidgets.QFrame.Raised)
134         self.frame.setObjectName("frame")
135         self.gridLayout = QtWidgets.QGridLayout(self.frame)
136         self.gridLayout.setObjectName("gridLayout")
137         self.DisciplineLabel = QtWidgets.QLabel(self.frame)
138         self.DisciplineLabel.setObjectName("DisciplineLabel")
139         self.gridLayout.addWidget(self.DisciplineLabel, 0, 0, 1,
    1)
140         self.DisciplineBox = QtWidgets.QComboBox(self.frame)
141         self.DisciplineBox.setObjectName("DisciplineBox")
142         self.DisciplineBox.addItem("Plug and Abandonment")
143         self.DisciplineBox.setItemText(1, "")
144         self.gridLayout.addWidget(self.DisciplineBox, 0, 1, 1, 1)
145         self.EventLabel = QtWidgets.QLabel(self.frame)
146         self.EventLabel.setObjectName("EventLabel")
```

```
147         self.gridLayout.addWidget(self.EventLabel, 1, 2, 1, 1)
148         self.ObjectiveLabel = QtWidgets.QLabel(self.frame)
149         self.ObjectiveLabel.setObjectName("ObjectiveLabel")
150         self.gridLayout.addWidget(self.ObjectiveLabel, 1, 0, 1, 1)
151         self.ObjectiveBox = QtWidgets.QComboBox(self.frame)
152         self.ObjectiveBox.setObjectName("ObjectiveBox")
153         self.ObjectiveBox.addItem("Choose an option...")
154         self.ObjectiveBox.addItem("Cementing",["Cement Plug","
    Cementing","Circ/Cond","Loss","MU,LD","Other","Prepare","RU,RD"
    ,"Test","Trip"])
155         self.ObjectiveBox.addItem("Cementing_int",["BOP", "Guide
    Posts","Bullhead","Cement","Clean hub", "DHSV - Brush Valve","
    DHSV - Insert Valve","Drift Well", "Manipulate Comp. eq.","
    Flowback","Injection Test","Logging CH", "LPR", "Lubticator
    Section","LS/SSTT","Mechanical Plug","Mob/Demob Vessel","Mob/
    Demobe/Spot","Move","Other","Scraper Run","Rig Over","Rig Up/
    Down","Rig/general","Survey","Tbg Hng/XT Plug","Template
    hatches","Tree Cap","Tree Debris Cap/THISL","Well Control","X-
    mas Tree"])
156         self.ObjectiveBox.addItem("Cement Plug",["Balanced Plug","
    Pump and Pull","Two-Plug Method","Dump Bailer"])
157         self.ObjectiveBox.addItem("Pumping",["Inject_Bullhead","
    Circulate","Displace","Flow Rate","Other","TBT","Safety Meeting
    ","General Investigation","Permit To Work","No Activity","
    Waiting, other","Safety Related","ROV Operations","Site survey"
    ,"Manipulate Compl. eq.","Repair or Replace","Tbg Hng/XT Plug",
    "X-mas Tree"])
158         self.ObjectiveBox.addItem("Logging_DP",["BOP","MU BHA","BO
     BHA", "RIH","POOH","Manipulate Compl. eq.","Flowback","
    Injection test","Logging CH","LRP","Lubricator section","LS/
    SSTT","Mechanical Plug","Mob/Demob Vessel","Mob/Demob/Spot","
    Other"])
159         self.ObjectiveBox.addItem("Section Milling",["Swarf
    Handling","Well control","Well Design","Casing Configuration","
    Milling fluid selection","Cut-out / Window in casing","Milling
    BHA configuration","Set Cement plug","Clean runs/ BOP cleaning"
    ,"Set Mechanical plug","Section mill 165 ft (50m) of 9 5/8","
    Underream Open Hole"])
160         self.ObjectiveBox.addItem("Plug_int",["Rigging_plug","
    Mechanical_Plug","Junk_basket","Well_Control_int","
    Scraper_cleanup_run","Rig_over","Mob_demob","Other_plug_int"])
161         self.ObjectiveBox.addItem("Logging",["BOP","Guide Posts","
    Bullhead","Clean hub","DHSV - Brush valve","DHSV - Insert valve
    ","Drift well","Manipulate Compl. eq.","Flowback","Injection
    test","Logging CH","LRP","Lubricator section","LS/SSTT","
    Mechanical Plug","Mob/Demob Vessel","Mob/Demob/Spot","Move","
    Other","Scraper Run","Rig Over","Rig Up/Down","Rig/general","
    Survey","Tbg Hng/XT Plug","Template hatches","Tree Cap","Tree
    Debris Cap/THISL","Well Control","X-mas Tree"])
162         self.ObjectiveBox.addItem("Casing",["MU_LD_Hanger","
    MU_Shoetrack","Run_casing_Liner_joints","
    Run_casing_Liner_stands","Run_casing_Liner_Flowcheck","
    Run_casing_Liner_circ_condition", "Run_casing_Liner_Handling
    equipment","Run_casing_Liner_Wash_work","Run_casing_Liner_Other
    "])
163         self.ObjectiveBox.addItem("Liner",["RU liner equipment", "
    MU Shoe", "Run liner", "Run liner, Circ", "Run liner, wash/work
    ", "Change handling equipment", "Change handling equipment/MU
```

```
Hanger", "Run liner on landing string", "Run liner on landing
string, wash/work", "Circulation/prep. time setting hanger/
packer", "Gravel pack liner/liner", "PU RT", "LD RT"])
164         self.ObjectiveBox.addItem("Drift Well",["Drift_run", "
Mob_demob", "Rig_over", "Well_Control_int", "Pumping", "
Rigging_Drift_run_int"])
165         self.ObjectiveBox.addItem("Fishing DP",["R/U, R/D", "M/U,
L/D", "RIH/POOH", "Trip", "Fishing", "Stuck/Lost Circ", "Other"
, "Well control", "Cut Casing/liner/tieback", "Mill Casing/
liner/tieback", "Release fish incl. Casing/liner/tieback", "
Other"])
166         self.ObjectiveBox.addItem("Fishing_int",["BOP", "Guide
Posts", "Bullhead", "Clean hub", "Cutting", "DHSV - Brush valve
", "DHSV - Insert valve", "Drift well", "Manipulate Compl. eq."
, "Fishing", "Flowback", "Junk basket", "LRP", "Lubricator
section", "LS/SSTT", "Mechanical Plug", "Milling", "Mob/Demob
Vessel", "Mob/Demob/Spot", "Move", "Other", "Scraper Run", "Rig
 Over", "Rig Up/Down", "Rig/general", "Survey", "Tbg Hng/XT
Plug", "Template hatches", "Tree Cap", "Tree Debris Cap/THISL",
 "Well Control", "X-mas Tree"])
167         self.ObjectiveBox.addItem("Suspension",["BOP's etc -
Install/Remove", "BPV - TH Plug", "Braided Line Operations", "
Bridge Plug - Install via DP/Tubing", "Cement Squeeze", "Cement
 Stinger RIH/POOH", "Cut & Retrieve Casing / Install Suspension
 Equipment", "Drill / Mill Cement", "Drill String - RIH / POOH"
, "Electric Line Operations", "General Investigation", "
Hydraulic Workover Operations", "Inflow Test", "Maintain/Change
 Out Drillstring", "Maintenance - Rig", "Manipulate Downhole
Equipment", "Mill / Cut / Push", "Moving / Skidding Rig / Work
unit", "No Activity", "Permit To Work", "Preparation / Clearing
 Site", "Pressure Test", "Production Riser - Install/Remove", "
Pump / Circulate / Displace", "Repair or Replace ROV", "
Retrievable Packer - Install and Remove via DP/Tubing", "Rig Up
/Down", "ROV Operations", "Safety Related", "Scraper Run", "
Seabed Survey SCP", "Set Cement Plug", "Slick Line Operations",
 "String - Works / Jar / Free Stuck Equipment", "Suspension
Flange - Install/Remove", "Tie-Back String -Install/Retrieve",
"Waiting on Cement", "Well Control (Losses)", "Well Control (
Pressure)", "Wellhead maintenance"])
168         self.ObjectiveBox.addItem("Suspension_intv",["BOP", "BPV -
 TH Plug", "Lubricator section", "LS/SSTT", "Mechanical Plug",
"Mob/Demob Vessel", "Mob/Demob/Spot", "Move", "Other", "Inflow
Test", "Scraper Run", "Rig Over", "Rig Up/Down", "Rig/general",
 "Survey", "Tbg Hng/XT Plug", "Template hatches", "Tree Cap", "
Tree Debris Cap/THISL", "Well Control", "X-mas Tree", "General
Investigation", "Maintenance - Rig", "Install Suspension
Equipment", "Manipulate Downhole Equipment", "No Activity", "
Permit To Work", "Preparation / Clearing Site", "Pressure Test"
, "Production Riser - Install/Remove", "Repair or Replace ROV",
 "ROV Operations", "Safety Related", "Seabed Survey SCP", "
Suspension Flange - Install/Remove", "Well Control (Losses)", "
Well Control (Pressure)", "Wellhead maintenance", "Guide Posts"
, "Bullhead", "Clean hub", "DHSV - Brush valve", "DHSV - Insert
 valve", "Drift well", "Manipulate Compl. eq.", "Flowback", "
Junk basket", "LRP", "Transit", "Transponder", "Tubeclean", "
Umbilical", "Waste Injection"])
169         self.ObjectiveBox.addItem("Well Control",["Observation", "
Circulate", "Flowcheck", "Kill well", "Cement plug", "Stuck/
```

```
    Lost Circ"])

        #Click The Objective

        self.ObjectiveBox.activated.connect(self.clicker)
        self.SaveButton.pressed.connect(self.save_it)

        self.gridLayout.addWidget(self.ObjectiveBox, 1, 1, 1, 1)
        self.SectionBox = QtWidgets.QComboBox(self.frame)
        self.SectionBox.setObjectName("SectionBox")
        self.SectionBox.addItem("Deep Plugging")
        self.SectionBox.addItem("Middle/Secondary Plugging")
        self.SectionBox.addItem("Shalow Plugging")
        self.gridLayout.addWidget(self.SectionBox, 0, 3, 1, 1)
        self.EventBox = QtWidgets.QComboBox(self.frame)
        self.EventBox.setObjectName("EventBox")
        self.gridLayout.addWidget(self.EventBox, 1, 3, 1, 1)
        self.SectionLabel = QtWidgets.QLabel(self.frame)
        self.SectionLabel.setObjectName("SectionLabel")
        self.gridLayout.addWidget(self.SectionLabel, 0, 2, 1, 1)
        self.frame_2 = QtWidgets.QFrame(self.tab)
        self.frame_2.setGeometry(QtCore.QRect(0, 240, 901, 101))
        self.frame_2.setFocusPolicy(QtCore.Qt.NoFocus)
        self.frame_2.setFrameShape(QtWidgets.QFrame.StyledPanel)
        self.frame_2.setFrameShadow(QtWidgets.QFrame.Raised)
        self.frame_2.setObjectName("frame_2")
        self.gridLayout_2 = QtWidgets.QGridLayout(self.frame_2)
        self.gridLayout_2.setObjectName("gridLayout_2")
        self.EngBox1 = QtWidgets.QComboBox(self.frame_2)
        self.EngBox1.setObjectName("EngBox1")
        self.EngBox1.addItem("NA")
        self.EngBox1.addItem("T&D")
        self.EngBox1.addItem("Hydeaulic")
        self.EngBox1.addItem("Bit")
        self.EngBox1.addItem("ECD")
        self.EngBox1.addItem("Pump Design")
        self.EngBox1.addItem("Kill Well")
        self.EngBox1.addItem("Kick Tolerance")
        self.EngBox1.addItem("Casing Design")
        self.EngBox1.addItem("Tubing Design")
        self.EngBox1.addItem("Casing Wear")
        self.EngBox1.addItem("Milling")
        self.EngBox1.addItem("Cement Plugging")
        self.gridLayout_2.addWidget(self.EngBox1, 0, 1, 1, 1)
        self.label_9 = QtWidgets.QLabel(self.frame_2)
        self.label_9.setObjectName("label_9")
        self.gridLayout_2.addWidget(self.label_9, 0, 2, 1, 1)
        self.label_7 = QtWidgets.QLabel(self.frame_2)
        self.label_7.setObjectName("label_7")
        self.gridLayout_2.addWidget(self.label_7, 0, 0, 1, 1)
        self.ContBox1 = QtWidgets.QComboBox(self.frame_2)
        self.ContBox1.setObjectName("ContBox1")
        self.ContBox1.addItem("Bits")
        self.ContBox1.addItem("Casing Accessories")
        self.ContBox1.addItem("Casing running")
        self.ContBox1.addItem("Cementing Services")
        self.ContBox1.addItem("Completion tubing")
        self.ContBox1.addItem("Coring")
```

```
227         self.ContBox1.addItem("Coiled Tubing (CT)")
228         self.ContBox1.addItem("D&I services and tools")
229         self.ContBox1.addItem("Fishing & milling")
230         self.ContBox1.addItem("Environmental & Waste control")
231         self.ContBox1.addItem("Gyro Surveying")
232         self.ContBox1.addItem("Inspection Services")
233         self.ContBox1.addItem("Mud chemicals")
234         self.ContBox1.addItem("Mud logging")
235         self.ContBox1.addItem("Rig contract")
236         self.ContBox1.addItem("Well testing services")
237         self.ContBox1.addItem("WH /XT")
238         self.gridLayout_2.addWidget(self.ContBox1, 0, 3, 1, 1)
239         self.ContBox2 = QtWidgets.QComboBox(self.frame_2)
240         self.ContBox2.addItem("Bits")
241         self.ContBox2.addItem("Casing Accessories")
242         self.ContBox2.addItem("Casing running")
243         self.ContBox2.addItem("Cementing Services")
244         self.ContBox2.addItem("Completion tubing")
245         self.ContBox2.addItem("Coring")
246         self.ContBox2.addItem("Coiled Tubing (CT)")
247         self.ContBox2.addItem("D&I services and tools")
248         self.ContBox2.addItem("Fishing & milling")
249         self.ContBox2.addItem("Environmental & Waste control")
250         self.ContBox2.addItem("Gyro Surveying")
251         self.ContBox2.addItem("Inspection Services")
252         self.ContBox2.addItem("Mud chemicals")
253         self.ContBox2.addItem("Mud logging")
254         self.ContBox2.addItem("Rig contract")
255         self.ContBox2.addItem("Well testing services")
256         self.ContBox2.addItem("WH /XT")
257         self.ContBox2.setObjectName("ContBox2")
258         self.gridLayout_2.addWidget(self.ContBox2, 1, 3, 1, 1)
259         self.EngBox2 = QtWidgets.QComboBox(self.frame_2)
260         self.EngBox2.setObjectName("EngBox2")
261         self.EngBox2.addItem("NA")
262         self.EngBox2.addItem("T&D")
263         self.EngBox2.addItem("Hydeaulic")
264         self.EngBox2.addItem("Bit")
265         self.EngBox2.addItem("ECD")
266         self.EngBox2.addItem("Pump Design")
267         self.EngBox2.addItem("Kill Well")
268         self.EngBox2.addItem("Kick Tolerance")
269         self.EngBox2.addItem("Casing Design")
270         self.EngBox2.addItem("Tubing Design")
271         self.EngBox2.addItem("Casing Wear")
272         self.EngBox2.addItem("Milling")
273         self.EngBox2.addItem("Cement Plugging")
274         self.gridLayout_2.addWidget(self.EngBox2, 1, 1, 1, 1)
275         self.tabWidget.addTab(self.tab, "")
276         self.tab_2 = QtWidgets.QWidget()
277         self.tab_2.setObjectName("tab_2")
278         self.CPTab = QtWidgets.QTabWidget(self.tab_2)
279         self.CPTab.setGeometry(QtCore.QRect(0, 0, 981, 571))
280         self.CPTab.setLayoutDirection(QtCore.Qt.LeftToRight)
281         self.CPTab.setTabPosition(QtWidgets.QTabWidget.North)
282         self.CPTab.setObjectName("CPTab")
283         self.CP1Tab = QtWidgets.QWidget()
284         self.CP1Tab.setObjectName("CP1Tab")
```

```
285         self.Info_Label = QtWidgets.QLabel(self.CP1Tab)
286         self.Info_Label.setGeometry(QtCore.QRect(10, 0, 951, 541))
287         self.Info_Label.setScaledContents(True)
288         self.Info_Label.setWordWrap(True)
289         self.Info_Label.setObjectName("Info_Label")
290         self.CPTab.addTab(self.CP1Tab, "")
291         self.CP2Tab = QtWidgets.QWidget()
292         self.CP2Tab.setObjectName("CP2Tab")
293         self.gridLayout_8 = QtWidgets.QGridLayout(self.CP2Tab)
294         self.gridLayout_8.setObjectName("gridLayout_8")
295         self.InputFrame = QtWidgets.QFrame(self.CP2Tab)
296         self.InputFrame.setAccessibleName("")
297         self.InputFrame.setStyleSheet("")
298         self.InputFrame.setFrameShape(QtWidgets.QFrame.NoFrame)
299         self.InputFrame.setFrameShadow(QtWidgets.QFrame.Raised)
300         self.InputFrame.setObjectName("InputFrame")
301         self.label_4 = QtWidgets.QLabel(self.InputFrame)
302         self.label_4.setGeometry(QtCore.QRect(-10, -10, 981, 561))
303         self.label_4.setText("")
304         self.label_4.setPixmap(QtGui.QPixmap("IMG.jpg"))
305         self.label_4.setScaledContents(True)
306         self.label_4.setObjectName("label_4")
307         self.layoutWidget_2 = QtWidgets.QWidget(self.InputFrame)
308         self.layoutWidget_2.setGeometry(QtCore.QRect(90, 190, 257,
     84))
309         self.layoutWidget_2.setObjectName("layoutWidget_2")
310         self.gridLayout_9 = QtWidgets.QGridLayout(self.
    layoutWidget_2)
311         self.gridLayout_9.setContentsMargins(0, 0, 0, 0)
312         self.gridLayout_9.setObjectName("gridLayout_9")
313         self.CasingSizeLabel = QtWidgets.QLabel(self.
    layoutWidget_2)
314         self.CasingSizeLabel.setMinimumSize(QtCore.QSize(100, 0))
315         self.CasingSizeLabel.setMaximumSize(QtCore.QSize(150,
    16777215))
316         self.CasingSizeLabel.setLayoutDirection(QtCore.Qt.
    LeftToRight)
317         self.CasingSizeLabel.setAlignment(QtCore.Qt.AlignLeading|
    QtCore.Qt.AlignLeft|QtCore.Qt.AlignVCenter)
318         self.CasingSizeLabel.setObjectName("CasingSizeLabel")
319         self.gridLayout_9.addWidget(self.CasingSizeLabel, 0, 0, 1,
     1)
320         self.CSLineEdit = QtWidgets.QLineEdit(self.layoutWidget_2)
321         self.CSLineEdit.setMinimumSize(QtCore.QSize(100, 0))
322         self.CSLineEdit.setMaximumSize(QtCore.QSize(100, 16777215)
    )
323         self.CSLineEdit.setLayoutDirection(QtCore.Qt.LeftToRight)
324         self.CSLineEdit.setAutoFillBackground(False)
325         self.CSLineEdit.setText("")
326         self.CSLineEdit.setClearButtonEnabled(True)
327         self.CSLineEdit.setObjectName("CSLineEdit")
328         self.gridLayout_9.addWidget(self.CSLineEdit, 0, 1, 1, 1)
329         self.gridLayout_3 = QtWidgets.QGridLayout()
330         self.gridLayout_3.setObjectName("gridLayout_3")
331         self.CasingShoeTVDLabel = QtWidgets.QLabel(self.
    layoutWidget_2)
332         self.CasingShoeTVDLabel.setMinimumSize(QtCore.QSize(100,
    0))
```

```
333         self.CasingShoeTVDLabel.setMaximumSize(QtCore.QSize(150,
    16777215))
334         self.CasingShoeTVDLabel.setAlignment(QtCore.Qt.
    AlignLeading|QtCore.Qt.AlignLeft|QtCore.Qt.AlignVCenter)
335         self.CasingShoeTVDLabel.setObjectName("CasingShoeTVDLabel"
    )
336         self.gridLayout_3.addWidget(self.CasingShoeTVDLabel, 0, 0,
     1, 1)
337         self.CasingShoeTVDEditLine = QtWidgets.QLineEdit(self.
    layoutWidget_2)
338         self.CasingShoeTVDEditLine.setMinimumSize(QtCore.QSize
    (100, 0))
339         self.CasingShoeTVDEditLine.setMaximumSize(QtCore.QSize
    (100, 100))
340         self.CasingShoeTVDEditLine.setLayoutDirection(QtCore.Qt.
    LeftToRight)
341         self.CasingShoeTVDEditLine.setAutoFillBackground(False)
342         self.CasingShoeTVDEditLine.setText("")
343         self.CasingShoeTVDEditLine.setClearButtonEnabled(True)
344         self.CasingShoeTVDEditLine.setObjectName("
    CasingShoeTVDEditLine")
345         self.gridLayout_3.addWidget(self.CasingShoeTVDEditLine, 0,
     1, 1, 1)
346         self.CasingShoeMDLabel = QtWidgets.QLabel(self.
    layoutWidget_2)
347         self.CasingShoeMDLabel.setMinimumSize(QtCore.QSize(100, 0)
    )
348         self.CasingShoeMDLabel.setMaximumSize(QtCore.QSize(150,
    16777215))
349         self.CasingShoeMDLabel.setAlignment(QtCore.Qt.AlignLeading
    |QtCore.Qt.AlignLeft|QtCore.Qt.AlignVCenter)
350         self.CasingShoeMDLabel.setObjectName("CasingShoeMDLabel")
351         self.gridLayout_3.addWidget(self.CasingShoeMDLabel, 1, 0,
    1, 1)
352         self.CasingShoeMDLineEdit = QtWidgets.QLineEdit(self.
    layoutWidget_2)
353         self.CasingShoeMDLineEdit.setMinimumSize(QtCore.QSize(100,
     0))
354         self.CasingShoeMDLineEdit.setMaximumSize(QtCore.QSize(100,
     100))
355         self.CasingShoeMDLineEdit.setLayoutDirection(QtCore.Qt.
    LeftToRight)
356         self.CasingShoeMDLineEdit.setAutoFillBackground(False)
357         self.CasingShoeMDLineEdit.setText("")
358         self.CasingShoeMDLineEdit.setClearButtonEnabled(True)
359         self.CasingShoeMDLineEdit.setObjectName("
    CasingShoeMDLineEdit")
360         self.gridLayout_3.addWidget(self.CasingShoeMDLineEdit, 1,
    1, 1, 1)
361         self.gridLayout_9.addLayout(self.gridLayout_3, 1, 0, 1, 2)
362         self.layoutWidget_3 = QtWidgets.QWidget(self.InputFrame)
363         self.layoutWidget_3.setGeometry(QtCore.QRect(120, 450,
    246, 53))
364         self.layoutWidget_3.setObjectName("layoutWidget_3")
365         self.gridLayout_10 = QtWidgets.QGridLayout(self.
    layoutWidget_3)
366         self.gridLayout_10.setContentsMargins(0, 0, 0, 0)
367         self.gridLayout_10.setObjectName("gridLayout_10")
```

```
368         self.HoleDepthMDLabel = QtWidgets.QLabel(self.
    layoutWidget_3)
369         self.HoleDepthMDLabel.setMinimumSize(QtCore.QSize(100, 0))
370         self.HoleDepthMDLabel.setMaximumSize(QtCore.QSize(150,
    16777215))
371         self.HoleDepthMDLabel.setAlignment(QtCore.Qt.AlignLeading|
    QtCore.Qt.AlignLeft|QtCore.Qt.AlignVCenter)
372         self.HoleDepthMDLabel.setObjectName("HoleDepthMDLabel")
373         self.gridLayout_10.addWidget(self.HoleDepthMDLabel, 0, 0,
    1, 1)
374         self.HDMDLineEdit = QtWidgets.QLineEdit(self.
    layoutWidget_3)
375         self.HDMDLineEdit.setMinimumSize(QtCore.QSize(100, 0))
376         self.HDMDLineEdit.setMaximumSize(QtCore.QSize(100,
    16777215))
377         self.HDMDLineEdit.setLayoutDirection(QtCore.Qt.LeftToRight
    )
378         self.HDMDLineEdit.setAutoFillBackground(False)
379         self.HDMDLineEdit.setText("")
380         self.HDMDLineEdit.setClearButtonEnabled(True)
381         self.HDMDLineEdit.setObjectName("HDMDLineEdit")
382         self.gridLayout_10.addWidget(self.HDMDLineEdit, 0, 1, 1,
    1)
383         self.HoleDepthTVDLabel = QtWidgets.QLabel(self.
    layoutWidget_3)
384         self.HoleDepthTVDLabel.setMinimumSize(QtCore.QSize(100, 0)
    )
385         self.HoleDepthTVDLabel.setMaximumSize(QtCore.QSize(150,
    16777215))
386         self.HoleDepthTVDLabel.setAlignment(QtCore.Qt.AlignLeading
    |QtCore.Qt.AlignLeft|QtCore.Qt.AlignVCenter)
387         self.HoleDepthTVDLabel.setObjectName("HoleDepthTVDLabel")
388         self.gridLayout_10.addWidget(self.HoleDepthTVDLabel, 1, 0,
     1, 1)
389         self.HDTVDLineEdit = QtWidgets.QLineEdit(self.
    layoutWidget_3)
390         self.HDTVDLineEdit.setMinimumSize(QtCore.QSize(100, 0))
391         self.HDTVDLineEdit.setMaximumSize(QtCore.QSize(100,
    16777215))
392         self.HDTVDLineEdit.setLayoutDirection(QtCore.Qt.
    LeftToRight)
393         self.HDTVDLineEdit.setAutoFillBackground(False)
394         self.HDTVDLineEdit.setText("")
395         self.HDTVDLineEdit.setClearButtonEnabled(True)
396         self.HDTVDLineEdit.setObjectName("HDTVDLineEdit")
397         self.gridLayout_10.addWidget(self.HDTVDLineEdit, 1, 1, 1,
    1)
398         self.layoutWidget_5 = QtWidgets.QWidget(self.InputFrame)
399         self.layoutWidget_5.setGeometry(QtCore.QRect(610, 380,
    261, 47))
400         self.layoutWidget_5.setObjectName("layoutWidget_5")
401         self.gridLayout_13 = QtWidgets.QGridLayout(self.
    layoutWidget_5)
402         self.gridLayout_13.setContentsMargins(0, 0, 0, 0)
403         self.gridLayout_13.setObjectName("gridLayout_13")
404         self.BottomLineEdit = QtWidgets.QLineEdit(self.
    layoutWidget_5)
405         self.BottomLineEdit.setMinimumSize(QtCore.QSize(100, 0))
```

```
406         self.BottomLineEdit.setMaximumSize(QtCore.QSize(100,
    16777215))
407         self.BottomLineEdit.setLayoutDirection(QtCore.Qt.
    LeftToRight)
408         self.BottomLineEdit.setAutoFillBackground(False)
409         self.BottomLineEdit.setText("")
410         self.BottomLineEdit.setClearButtonEnabled(True)
411         self.BottomLineEdit.setObjectName("BottomLineEdit")
412         self.gridLayout_13.addWidget(self.BottomLineEdit, 0, 1, 1,
     1)
413         self.BottomLabel = QtWidgets.QLabel(self.layoutWidget_5)
414         self.BottomLabel.setMinimumSize(QtCore.QSize(100, 0))
415         self.BottomLabel.setMaximumSize(QtCore.QSize(155,
    16777215))
416         self.BottomLabel.setLayoutDirection(QtCore.Qt.LeftToRight)
417         self.BottomLabel.setAlignment(QtCore.Qt.AlignLeading|
    QtCore.Qt.AlignLeft|QtCore.Qt.AlignVCenter)
418         self.BottomLabel.setObjectName("BottomLabel")
419         self.gridLayout_13.addWidget(self.BottomLabel, 0, 0, 1, 1)
420         self.layoutWidget_6 = QtWidgets.QWidget(self.InputFrame)
421         self.layoutWidget_6.setGeometry(QtCore.QRect(90, 340, 241,
     53))
422         self.layoutWidget_6.setObjectName("layoutWidget_6")
423         self.gridLayout_11 = QtWidgets.QGridLayout(self.
    layoutWidget_6)
424         self.gridLayout_11.setContentsMargins(0, 0, 0, 0)
425         self.gridLayout_11.setObjectName("gridLayout_11")
426         self.PSDRLabel = QtWidgets.QLabel(self.layoutWidget_6)
427         self.PSDRLabel.setMinimumSize(QtCore.QSize(100, 0))
428         self.PSDRLabel.setMaximumSize(QtCore.QSize(150, 16777215))
429         self.PSDRLabel.setAlignment(QtCore.Qt.AlignLeading|QtCore.
    Qt.AlignLeft|QtCore.Qt.AlignVCenter)
430         self.PSDRLabel.setObjectName("PSDRLabel")
431         self.gridLayout_11.addWidget(self.PSDRLabel, 0, 0, 1, 1)
432         self.PSDRLineEdit = QtWidgets.QLineEdit(self.
    layoutWidget_6)
433         self.PSDRLineEdit.setMinimumSize(QtCore.QSize(100, 0))
434         self.PSDRLineEdit.setMaximumSize(QtCore.QSize(100,
    16777215))
435         self.PSDRLineEdit.setLayoutDirection(QtCore.Qt.LeftToRight
    )
436         self.PSDRLineEdit.setAutoFillBackground(False)
437         self.PSDRLineEdit.setText("")
438         self.PSDRLineEdit.setMaxLength(32763)
439         self.PSDRLineEdit.setFrame(True)
440         self.PSDRLineEdit.setEchoMode(QtWidgets.QLineEdit.Normal)
441         self.PSDRLineEdit.setDragEnabled(False)
442         self.PSDRLineEdit.setClearButtonEnabled(True)
443         self.PSDRLineEdit.setObjectName("PSDRLineEdit")
444         self.gridLayout_11.addWidget(self.PSDRLineEdit, 0, 1, 1,
    1)
445         self.PSDRLabel2 = QtWidgets.QLabel(self.layoutWidget_6)
446         self.PSDRLabel2.setMinimumSize(QtCore.QSize(100, 0))
447         self.PSDRLabel2.setMaximumSize(QtCore.QSize(150, 16777215)
    )
448         self.PSDRLabel2.setAlignment(QtCore.Qt.AlignLeading|QtCore
    .Qt.AlignLeft|QtCore.Qt.AlignVCenter)
449         self.PSDRLabel2.setObjectName("PSDRLabel2")
```

```
450         self.gridLayout_11.addWidget(self.PSDRLabel2, 1, 0, 1, 1)
451         self.PSDRLineEdit2 = QtWidgets.QLineEdit(self.
    layoutWidget_6)
452         self.PSDRLineEdit2.setMinimumSize(QtCore.QSize(100, 0))
453         self.PSDRLineEdit2.setMaximumSize(QtCore.QSize(100,
    16777215))
454         self.PSDRLineEdit2.setLayoutDirection(QtCore.Qt.
    LeftToRight)
455         self.PSDRLineEdit2.setAutoFillBackground(False)
456         self.PSDRLineEdit2.setText("")
457         self.PSDRLineEdit2.setMaxLength(32763)
458         self.PSDRLineEdit2.setFrame(True)
459         self.PSDRLineEdit2.setEchoMode(QtWidgets.QLineEdit.Normal)
460         self.PSDRLineEdit2.setDragEnabled(False)
461         self.PSDRLineEdit2.setClearButtonEnabled(True)
462         self.PSDRLineEdit2.setObjectName("PSDRLineEdit2")
463         self.gridLayout_11.addWidget(self.PSDRLineEdit2, 1, 1, 1,
    1)
464         self.widget = QtWidgets.QWidget(self.InputFrame)
465         self.widget.setGeometry(QtCore.QRect(620, 280, 246, 53))
466         self.widget.setObjectName("widget")
467         self.gridLayout_7 = QtWidgets.QGridLayout(self.widget)
468         self.gridLayout_7.setContentsMargins(0, 0, 0, 0)
469         self.gridLayout_7.setObjectName("gridLayout_7")
470         self.HightCementLabel = QtWidgets.QLabel(self.widget)
471         self.HightCementLabel.setMinimumSize(QtCore.QSize(100, 0))
472         self.HightCementLabel.setMaximumSize(QtCore.QSize(150,
    16777215))
473         self.HightCementLabel.setLayoutDirection(QtCore.Qt.
    LeftToRight)
474         self.HightCementLabel.setAlignment(QtCore.Qt.AlignLeading|
    QtCore.Qt.AlignLeft|QtCore.Qt.AlignVCenter)
475         self.HightCementLabel.setObjectName("HightCementLabel")
476         self.gridLayout_7.addWidget(self.HightCementLabel, 0, 0,
    1, 1)
477         self.HightCementLineEdit = QtWidgets.QLineEdit(self.widget
    )
478         self.HightCementLineEdit.setMinimumSize(QtCore.QSize(100,
    0))
479         self.HightCementLineEdit.setMaximumSize(QtCore.QSize(100,
    16777215))
480         self.HightCementLineEdit.setLayoutDirection(QtCore.Qt.
    LeftToRight)
481         self.HightCementLineEdit.setAutoFillBackground(False)
482         self.HightCementLineEdit.setText("")
483         self.HightCementLineEdit.setClearButtonEnabled(True)
484         self.HightCementLineEdit.setObjectName("
    HightCementLineEdit")
485         self.gridLayout_7.addWidget(self.HightCementLineEdit, 0,
    1, 1, 1)
486         self.TOCLabel = QtWidgets.QLabel(self.widget)
487         self.TOCLabel.setMinimumSize(QtCore.QSize(100, 0))
488         self.TOCLabel.setMaximumSize(QtCore.QSize(150, 16777215))
489         self.TOCLabel.setLayoutDirection(QtCore.Qt.LeftToRight)
490         self.TOCLabel.setAlignment(QtCore.Qt.AlignLeading|QtCore.
    Qt.AlignLeft|QtCore.Qt.AlignVCenter)
491         self.TOCLabel.setObjectName("TOCLabel")
492         self.gridLayout_7.addWidget(self.TOCLabel, 1, 0, 1, 1)
```

```
493         self.TOCLineEdit = QtWidgets.QLineEdit(self.widget)
494         self.TOCLineEdit.setMinimumSize(QtCore.QSize(100, 0))
495         self.TOCLineEdit.setMaximumSize(QtCore.QSize(100,
    16777215))
496         self.TOCLineEdit.setLayoutDirection(QtCore.Qt.LeftToRight)
497         self.TOCLineEdit.setAutoFillBackground(False)
498         self.TOCLineEdit.setText("")
499         self.TOCLineEdit.setClearButtonEnabled(True)
500         self.TOCLineEdit.setObjectName("TOCLineEdit")
501         self.gridLayout_7.addWidget(self.TOCLineEdit, 1, 1, 1, 1)
502         self.gridLayout_8.addWidget(self.InputFrame, 0, 0, 1, 1)
503         self.CPTab.addTab(self.CP2Tab, "")
504         self.CP4Tab = QtWidgets.QWidget()
505         self.CP4Tab.setObjectName("CP4Tab")
506         self.gridLayout_5 = QtWidgets.QGridLayout(self.CP4Tab)
507         self.gridLayout_5.setObjectName("gridLayout_5")
508         self.GeneralFrame = QtWidgets.QFrame(self.CP4Tab)
509         self.GeneralFrame.setMaximumSize(QtCore.QSize(16777215,
    16777215))
510         self.GeneralFrame.setFrameShape(QtWidgets.QFrame.
    StyledPanel)
511         self.GeneralFrame.setFrameShadow(QtWidgets.QFrame.Raised)
512         self.GeneralFrame.setObjectName("GeneralFrame")
513         self.gridLayout_6 = QtWidgets.QGridLayout(self.
    GeneralFrame)
514         self.gridLayout_6.setObjectName("gridLayout_6")
515         self.TubeIDLineEdit = QtWidgets.QLineEdit(self.
    GeneralFrame)
516         self.TubeIDLineEdit.setMaximumSize(QtCore.QSize(100,
    16777215))
517         self.TubeIDLineEdit.setText("")
518         self.TubeIDLineEdit.setClearButtonEnabled(True)
519         self.TubeIDLineEdit.setObjectName("TubeIDLineEdit")
520         self.gridLayout_6.addWidget(self.TubeIDLineEdit, 0, 1, 1,
    1)
521         self.HoleSizeLineEdit = QtWidgets.QLineEdit(self.
    GeneralFrame)
522         self.HoleSizeLineEdit.setMaximumSize(QtCore.QSize(100,
    16777215))
523         self.HoleSizeLineEdit.setText("")
524         self.HoleSizeLineEdit.setClearButtonEnabled(True)
525         self.HoleSizeLineEdit.setObjectName("HoleSizeLineEdit")
526         self.gridLayout_6.addWidget(self.HoleSizeLineEdit, 0, 3,
    1, 1)
527         self.TubeODLabel = QtWidgets.QLabel(self.GeneralFrame)
528         self.TubeODLabel.setMinimumSize(QtCore.QSize(100, 0))
529         self.TubeODLabel.setMaximumSize(QtCore.QSize(150,
    16777215))
530         self.TubeODLabel.setAlignment(QtCore.Qt.AlignLeading|
    QtCore.Qt.AlignLeft|QtCore.Qt.AlignVCenter)
531         self.TubeODLabel.setObjectName("TubeODLabel")
532         self.gridLayout_6.addWidget(self.TubeODLabel, 3, 0, 1, 1)
533         self.TubeODLineEdit = QtWidgets.QLineEdit(self.
    GeneralFrame)
534         self.TubeODLineEdit.setMaximumSize(QtCore.QSize(100,
    16777215))
535         self.TubeODLineEdit.setText("")
536         self.TubeODLineEdit.setClearButtonEnabled(True)
```

```
537         self.TubeODLineEdit.setObjectName("TubeODLineEdit")
538         self.gridLayout_6.addWidget(self.TubeODLineEdit, 3, 1, 1,
    1)
539         self.ExcessLabel = QtWidgets.QLabel(self.GeneralFrame)
540         self.ExcessLabel.setMinimumSize(QtCore.QSize(100, 0))
541         self.ExcessLabel.setMaximumSize(QtCore.QSize(150,
    16777215))
542         self.ExcessLabel.setLayoutDirection(QtCore.Qt.LeftToRight)
543         self.ExcessLabel.setAlignment(QtCore.Qt.AlignLeading|
    QtCore.Qt.AlignLeft|QtCore.Qt.AlignVCenter)
544         self.ExcessLabel.setObjectName("ExcessLabel")
545         self.gridLayout_6.addWidget(self.ExcessLabel, 5, 2, 1, 1)
546         self.TubeIDLabel = QtWidgets.QLabel(self.GeneralFrame)
547         self.TubeIDLabel.setMinimumSize(QtCore.QSize(100, 0))
548         self.TubeIDLabel.setMaximumSize(QtCore.QSize(150,
    16777215))
549         self.TubeIDLabel.setAlignment(QtCore.Qt.AlignLeading|
    QtCore.Qt.AlignLeft|QtCore.Qt.AlignVCenter)
550         self.TubeIDLabel.setObjectName("TubeIDLabel")
551         self.gridLayout_6.addWidget(self.TubeIDLabel, 0, 0, 1, 1)
552         self.HoleSizeLabel = QtWidgets.QLabel(self.GeneralFrame)
553         self.HoleSizeLabel.setMinimumSize(QtCore.QSize(100, 0))
554         self.HoleSizeLabel.setMaximumSize(QtCore.QSize(150,
    16777215))
555         self.HoleSizeLabel.setAlignment(QtCore.Qt.AlignLeading|
    QtCore.Qt.AlignLeft|QtCore.Qt.AlignVCenter)
556         self.HoleSizeLabel.setObjectName("HoleSizeLabel")
557         self.gridLayout_6.addWidget(self.HoleSizeLabel, 0, 2, 1,
    1)
558         self.PipeIDLabel = QtWidgets.QLabel(self.GeneralFrame)
559         self.PipeIDLabel.setMinimumSize(QtCore.QSize(100, 0))
560         self.PipeIDLabel.setMaximumSize(QtCore.QSize(150,
    16777215))
561         self.PipeIDLabel.setAlignment(QtCore.Qt.AlignLeading|
    QtCore.Qt.AlignLeft|QtCore.Qt.AlignVCenter)
562         self.PipeIDLabel.setObjectName("PipeIDLabel")
563         self.gridLayout_6.addWidget(self.PipeIDLabel, 4, 0, 1, 1)
564         self.PipeIDLineEdit = QtWidgets.QLineEdit(self.
    GeneralFrame)
565         self.PipeIDLineEdit.setMaximumSize(QtCore.QSize(100,
    16777215))
566         self.PipeIDLineEdit.setText("")
567         self.PipeIDLineEdit.setClearButtonEnabled(True)
568         self.PipeIDLineEdit.setObjectName("PipeIDLineEdit")
569         self.gridLayout_6.addWidget(self.PipeIDLineEdit, 4, 1, 1,
    1)
570         self.CementDensLineEdit = QtWidgets.QLineEdit(self.
    GeneralFrame)
571         self.CementDensLineEdit.setMaximumSize(QtCore.QSize(100,
    16777215))
572         self.CementDensLineEdit.setText("")
573         self.CementDensLineEdit.setClearButtonEnabled(True)
574         self.CementDensLineEdit.setObjectName("CementDensLineEdit"
    )
575         self.gridLayout_6.addWidget(self.CementDensLineEdit, 4, 3,
     1, 1)
576         self.PipeODLineEdit = QtWidgets.QLineEdit(self.
    GeneralFrame)
```

```
577         self.PipeODLineEdit.setMaximumSize(QtCore.QSize(100,
    16777215))
578         self.PipeODLineEdit.setText("")
579         self.PipeODLineEdit.setClearButtonEnabled(True)
580         self.PipeODLineEdit.setObjectName("PipeODLineEdit")
581         self.gridLayout_6.addWidget(self.PipeODLineEdit, 5, 1, 1,
    1)
582         self.PipeODLabel = QtWidgets.QLabel(self.GeneralFrame)
583         self.PipeODLabel.setMinimumSize(QtCore.QSize(100, 0))
584         self.PipeODLabel.setMaximumSize(QtCore.QSize(150,
    16777215))
585         self.PipeODLabel.setAlignment(QtCore.Qt.AlignLeading|
    QtCore.Qt.AlignLeft|QtCore.Qt.AlignVCenter)
586         self.PipeODLabel.setObjectName("PipeODLabel")
587         self.gridLayout_6.addWidget(self.PipeODLabel, 5, 0, 1, 1)
588         self.CementDensLabel = QtWidgets.QLabel(self.GeneralFrame)
589         self.CementDensLabel.setMinimumSize(QtCore.QSize(100, 0))
590         self.CementDensLabel.setMaximumSize(QtCore.QSize(150,
    16777215))
591         self.CementDensLabel.setLayoutDirection(QtCore.Qt.
    LeftToRight)
592         self.CementDensLabel.setAlignment(QtCore.Qt.AlignLeading|
    QtCore.Qt.AlignLeft|QtCore.Qt.AlignVCenter)
593         self.CementDensLabel.setObjectName("CementDensLabel")
594         self.gridLayout_6.addWidget(self.CementDensLabel, 4, 2, 1,
     1)
595         self.ExcessLineEdit = QtWidgets.QLineEdit(self.
    GeneralFrame)
596         self.ExcessLineEdit.setMaximumSize(QtCore.QSize(100,
    16777215))
597         self.ExcessLineEdit.setText("")
598         self.ExcessLineEdit.setClearButtonEnabled(True)
599         self.ExcessLineEdit.setObjectName("ExcessLineEdit")
600         self.gridLayout_6.addWidget(self.ExcessLineEdit, 5, 3, 1,
    1)
601         self.CementLengthLabel = QtWidgets.QLabel(self.
    GeneralFrame)
602         self.CementLengthLabel.setMinimumSize(QtCore.QSize(100, 0)
    )
603         self.CementLengthLabel.setMaximumSize(QtCore.QSize(150,
    16777215))
604         self.CementLengthLabel.setAlignment(QtCore.Qt.AlignLeading
    |QtCore.Qt.AlignLeft|QtCore.Qt.AlignVCenter)
605         self.CementLengthLabel.setObjectName("CementLengthLabel")
606         self.gridLayout_6.addWidget(self.CementLengthLabel, 3, 2,
    1, 1)
607         self.CementLengthLineEdit = QtWidgets.QLineEdit(self.
    GeneralFrame)
608         self.CementLengthLineEdit.setMaximumSize(QtCore.QSize(100,
     16777215))
609         self.CementLengthLineEdit.setText("")
610         self.CementLengthLineEdit.setClearButtonEnabled(True)
611         self.CementLengthLineEdit.setObjectName("
    CementLengthLineEdit")
612         self.gridLayout_6.addWidget(self.CementLengthLineEdit, 3,
    3, 1, 1)
613         self.gridLayout_5.addWidget(self.GeneralFrame, 0, 0, 1, 1)
614         self.CPTab.addTab(self.CP4Tab, "")
```

```
615         self.CP3Tab = QtWidgets.QWidget()
616         self.CP3Tab.setObjectName("CP3Tab")
617         self.SDFrame = QtWidgets.QFrame(self.CP3Tab)
618         self.SDFrame.setGeometry(QtCore.QRect(10, 0, 981, 541))
619         self.SDFrame.setFrameShape(QtWidgets.QFrame.StyledPanel)
620         self.SDFrame.setFrameShadow(QtWidgets.QFrame.Raised)
621         self.SDFrame.setObjectName("SDFrame")
622         self.SpacerDensLabel = QtWidgets.QLabel(self.SDFrame)
623         self.SpacerDensLabel.setGeometry(QtCore.QRect(362, 268,
    150, 16))
624         self.SpacerDensLabel.setMinimumSize(QtCore.QSize(150, 0))
625         self.SpacerDensLabel.setMaximumSize(QtCore.QSize(150,
    16777215))
626         self.SpacerDensLabel.setLayoutDirection(QtCore.Qt.
    LeftToRight)
627         self.SpacerDensLabel.setAlignment(QtCore.Qt.AlignLeading|
    QtCore.Qt.AlignLeft|QtCore.Qt.AlignVCenter)
628         self.SpacerDensLabel.setObjectName("SpacerDensLabel")
629         self.SpacerDensLineEdit = QtWidgets.QLineEdit(self.SDFrame
    )
630         self.SpacerDensLineEdit.setGeometry(QtCore.QRect(519, 268,
     100, 22))
631         self.SpacerDensLineEdit.setMaximumSize(QtCore.QSize(100,
    16777215))
632         self.SpacerDensLineEdit.setText("")
633         self.SpacerDensLineEdit.setClearButtonEnabled(True)
634         self.SpacerDensLineEdit.setObjectName("SpacerDensLineEdit"
    )
635         self.SpacerVLineEdit = QtWidgets.QLineEdit(self.SDFrame)
636         self.SpacerVLineEdit.setGeometry(QtCore.QRect(519, 180,
    100, 22))
637         self.SpacerVLineEdit.setMaximumSize(QtCore.QSize(100,
    16777215))
638         self.SpacerVLineEdit.setText("")
639         self.SpacerVLineEdit.setClearButtonEnabled(True)
640         self.SpacerVLineEdit.setObjectName("SpacerVLineEdit")
641         self.SpacerVLabel = QtWidgets.QLabel(self.SDFrame)
642         self.SpacerVLabel.setGeometry(QtCore.QRect(362, 180, 150,
    16))
643         self.SpacerVLabel.setMinimumSize(QtCore.QSize(150, 0))
644         self.SpacerVLabel.setMaximumSize(QtCore.QSize(150,
    16777215))
645         self.SpacerVLabel.setLayoutDirection(QtCore.Qt.LeftToRight
    )
646         self.SpacerVLabel.setAlignment(QtCore.Qt.AlignLeading|
    QtCore.Qt.AlignLeft|QtCore.Qt.AlignVCenter)
647         self.SpacerVLabel.setObjectName("SpacerVLabel")
648         self.CPTab.addTab(self.CP3Tab, "")
649         self.CP5Tab = QtWidgets.QWidget()
650         self.CP5Tab.setObjectName("CP5Tab")
651         self.DFFrame = QtWidgets.QFrame(self.CP5Tab)
652         self.DFFrame.setGeometry(QtCore.QRect(10, 0, 971, 501))
653         self.DFFrame.setFrameShape(QtWidgets.QFrame.StyledPanel)
654         self.DFFrame.setFrameShadow(QtWidgets.QFrame.Raised)
655         self.DFFrame.setObjectName("DFFrame")
656         self.MudTypeLabel = QtWidgets.QLabel(self.DFFrame)
657         self.MudTypeLabel.setGeometry(QtCore.QRect(362, 180, 150,
    16))
```

```
658         self.MudTypeLabel.setMinimumSize(QtCore.QSize(150, 0))
659         self.MudTypeLabel.setMaximumSize(QtCore.QSize(150,
    16777215))
660         self.MudTypeLabel.setObjectName("MudTypeLabel")
661         self.MudDensLabel = QtWidgets.QLabel(self.DFFrame)
662         self.MudDensLabel.setGeometry(QtCore.QRect(362, 268, 150,
    16))
663         self.MudDensLabel.setMinimumSize(QtCore.QSize(150, 0))
664         self.MudDensLabel.setMaximumSize(QtCore.QSize(150,
    16777215))
665         self.MudDensLabel.setObjectName("MudDensLabel")
666         self.MudTypeComboBox = QtWidgets.QComboBox(self.DFFrame)
667         self.MudTypeComboBox.setGeometry(QtCore.QRect(519, 180,
    100, 22))
668         self.MudTypeComboBox.setMaximumSize(QtCore.QSize(100,
    16777215))
669         self.MudTypeComboBox.setEditable(True)
670         self.MudTypeComboBox.setObjectName("MudTypeComboBox")
671         self.MudTypeComboBox.addItem("")
672         self.MudTypeComboBox.addItem("")
673         self.MudTypeComboBox.addItem("")
674         self.MudDensLineEdit = QtWidgets.QLineEdit(self.DFFrame)
675         self.MudDensLineEdit.setGeometry(QtCore.QRect(519, 268,
    100, 22))
676         self.MudDensLineEdit.setMaximumSize(QtCore.QSize(100,
    16777215))
677         self.MudDensLineEdit.setObjectName("MudDensLineEdit")
678         #self.gridLayout_7.addWidget(self.MudTypeComboBox, 1, 1,
    1, 1)
679         #spacerItem8 = QtWidgets.QSpacerItem(40, 20, QtWidgets.
    QSizePolicy.Expanding, QtWidgets.QSizePolicy.Minimum)
680         #self.gridLayout_7.addItem(spacerItem8, 1, 3, 1, 1)
681         self.CalcButton = QtWidgets.QPushButton(self.CP5Tab)
682         self.CalcButton.setGeometry(QtCore.QRect(400, 460, 200,
    28))
683         self.CalcButton.setMinimumSize(QtCore.QSize(150, 0))
684         self.CalcButton.setMaximumSize(QtCore.QSize(250, 16777215)
    )
685         self.CalcButton.setCheckable(False)
686         self.CalcButton.setObjectName("CalcButton")
687         self.CPTab.addTab(self.CP5Tab, "")
688         self.tabWidget.addTab(self.tab_2, "")
689         self.tab_4 = QtWidgets.QWidget()
690         self.tab_4.setObjectName("tab_4")
691         self.ContractsTab = QtWidgets.QTabWidget(self.tab_4)
692         self.ContractsTab.setGeometry(QtCore.QRect(0, 0, 991, 571)
    )
693         self.ContractsTab.setTabPosition(QtWidgets.QTabWidget.
    North)
694         self.ContractsTab.setObjectName("ContractsTab")
695         self.tab_5 = QtWidgets.QWidget()
696         self.tab_5.setObjectName("tab_5")
697         self.CasingRunningLabel = QtWidgets.QLabel(self.tab_5)
698         self.CasingRunningLabel.setGeometry(QtCore.QRect(0, 0,
    981, 541))
699         self.CasingRunningLabel.setScaledContents(True)
700         self.CasingRunningLabel.setWordWrap(True)
701         self.CasingRunningLabel.setObjectName("CasingRunningLabel"
```

```
    )
        self.ContractsTab.addTab(self.tab_5, "")
        self.tab_6 = QtWidgets.QWidget()
        self.tab_6.setObjectName("tab_6")
        self.CemServLabel = QtWidgets.QLabel(self.tab_6)
        self.CemServLabel.setGeometry(QtCore.QRect(0, 0, 971, 531)
    )
        self.CemServLabel.setScaledContents(True)
        self.CemServLabel.setWordWrap(True)
        self.CemServLabel.setObjectName("CemServLabel")
        self.ContractsTab.addTab(self.tab_6, "")
        self.tab_7 = QtWidgets.QWidget()
        self.tab_7.setObjectName("tab_7")
        self.CTLabel = QtWidgets.QLabel(self.tab_7)
        self.CTLabel.setGeometry(QtCore.QRect(0, 0, 971, 531))
        self.CTLabel.setScaledContents(True)
        self.CTLabel.setWordWrap(True)
        self.CTLabel.setOpenExternalLinks(False)
        self.CTLabel.setObjectName("CTLabel")
        self.ContractsTab.addTab(self.tab_7, "")
        self.tab_8 = QtWidgets.QWidget()
        self.tab_8.setObjectName("tab_8")
        self.EnvLabel = QtWidgets.QLabel(self.tab_8)
        self.EnvLabel.setGeometry(QtCore.QRect(0, 0, 971, 531))
        self.EnvLabel.setScaledContents(True)
        self.EnvLabel.setWordWrap(True)
        self.EnvLabel.setObjectName("EnvLabel")
        self.ContractsTab.addTab(self.tab_8, "")
        self.tab_9 = QtWidgets.QWidget()
        self.tab_9.setObjectName("tab_9")
        self.MLLabel = QtWidgets.QLabel(self.tab_9)
        self.MLLabel.setGeometry(QtCore.QRect(0, 0, 971, 531))
        self.MLLabel.setScaledContents(True)
        self.MLLabel.setWordWrap(True)
        self.MLLabel.setObjectName("MLLabel")
        self.ContractsTab.addTab(self.tab_9, "")
        self.tab_10 = QtWidgets.QWidget()
        self.tab_10.setObjectName("tab_10")
        self.RigLabel = QtWidgets.QLabel(self.tab_10)
        self.RigLabel.setGeometry(QtCore.QRect(0, 0, 971, 531))
        self.RigLabel.setScaledContents(True)
        self.RigLabel.setWordWrap(True)
        self.RigLabel.setObjectName("RigLabel")
        self.ContractsTab.addTab(self.tab_10, "")
        self.tab_11 = QtWidgets.QWidget()
        self.tab_11.setObjectName("tab_11")
        self.WellTLabel = QtWidgets.QLabel(self.tab_11)
        self.WellTLabel.setGeometry(QtCore.QRect(0, 0, 971, 531))
        self.WellTLabel.setScaledContents(True)
        self.WellTLabel.setWordWrap(True)
        self.WellTLabel.setObjectName("WellTLabel")
        self.ContractsTab.addTab(self.tab_11, "")
        self.tab_12 = QtWidgets.QWidget()
        self.tab_12.setObjectName("tab_12")
        self.WHLabel = QtWidgets.QLabel(self.tab_12)
        self.WHLabel.setGeometry(QtCore.QRect(0, 0, 971, 531))
        self.WHLabel.setScaledContents(True)
        self.WHLabel.setWordWrap(True)
```

```
        self.WHLabel.setObjectName("WHLabel")
        self.ContractsTab.addTab(self.tab_12, "")
        self.tabWidget.addTab(self.tab_4, "")
        Software.setCentralWidget(self.centralwidget)
        self.menubar = QtWidgets.QMenuBar(Software)
        self.menubar.setGeometry(QtCore.QRect(0, 0, 1000, 26))
        self.menubar.setObjectName("menubar")
        Software.setMenuBar(self.menubar)
        self.statusbar = QtWidgets.QStatusBar(Software)
        self.statusbar.setObjectName("statusbar")
        Software.setStatusBar(self.statusbar)
        self.actionSave = QtWidgets.QAction(Software)
        self.actionSave.setObjectName("actionSave")
        self.actionOpen = QtWidgets.QAction(Software)
        self.actionOpen.setObjectName("actionOpen")
        self.actionHome = QtWidgets.QAction(Software)
        icon = QtGui.QIcon()
        icon.addPixmap(QtGui.QPixmap("C:/Users/AMIN/OneDrive/
    Pictures/884555.jpeg"), QtGui.QIcon.Normal, QtGui.QIcon.Off)
        self.actionHome.setIcon(icon)
        self.actionHome.setObjectName("actionHome")

        self.CalcButton.clicked.connect(self.calculate)

        self.retranslateUi(Software)
        self.tabWidget.setCurrentIndex(0)
        QtCore.QMetaObject.connectSlotsByName(Software)

        self.grab_all()

    def grab_all(self):

        conn = sqlite3.connect('mylist.db')

        c = conn.cursor()

        c.execute("Select * FROM well_list")
        records = c.fetchall()




        for record in records:
            self.mylist_listWidget.addItem(str(record[0]))

    def clicker(self,index):

        #Clear the second box
        self.EventBox.clear()
        #Dependent
        self.EventBox.addItems(self.ObjectiveBox.itemData(index))


    def save_it(self):

            conn = sqlite3.connect('mylist.db')

            c = conn.cursor()
```

```
815
816             c.execute('DELETE FROM well_list;',)
817
818             items = [self.Nameline,self.Typeline,self.Resline,self
    .Objtext,self.Sumtext]
819
820             for item in items:
821                 c.execute("INSERT INTO well_list VALUES (:item)",
822                 {
823                     'item': item.Info(),
824                 })
825
826             conn.commit()
827
828             conn.close()
829
830             msg = QMessageBox()
831             msg.setWindowTitle("Well Information saved!")
832             msg.setText("The infoprmation has been saved in
    database.")
833             msg.setIcon(QMessageBox.Information)
834             # x = msg.exec_()
835
836     def calculate(self):
837
838         csg = float(self.CSLineEdit.text())
839         csgs_MD = float(self.CasingShoeMDLineEdit.text())
840         csgs_TVD = float(self.CasingShoeTVDEditLine.text())
841         Hole_Size = float(self.HoleSizeLineEdit.text())
842         HD_MD = float(self.HDMDLineEdit.text())
843         HD_TVD = float(self.HDTVDLineEdit.text())
844         PSDR = float(self.PSDRLineEdit.text())
845         Tubing = float(self.TubeODLineEdit.text())
846         Tubing_ID = float(self.TubeIDLineEdit.text())
847         DP_OD = float(self.PipeODLineEdit.text())
848         DP_ID = float(self.PipeIDLineEdit.text())
849         V_s = float(self.SpacerVLineEdit.text())
850         Spacer_Den = float(self.SpacerDensLineEdit.text())
851         PLC = float(self.CementLengthLineEdit.text())
852         PBCP = float(self.BottomLineEdit.text())
853         TOC = float(self.TOCLineEdit.text())
854         HCS = float(self.HightCementLineEdit.text())
855         Cement_DEN = float(self.CementDensLineEdit.text())
856         EX = float(self.ExcessLineEdit.text())
857         Mud_dens = float(self.MudDensLineEdit.text())
858
859         #Equations
860         Hole_Cap = Hole_Size**2 / 1029.4
861         CSC = Tubing_ID**2 / 1029.4
862         ACCS = (Hole_Size**2 - Tubing**2) / 1029.4
863         ACDP = (Hole_Size**2 - DP_OD**2 )/ 1029.4
864         DP_CAP = DP_ID**2 / 1029.4
865
866         #Cement:
867         VC = PLC * Hole_Cap * (1+0.01*EX)
868         HC_s = VC / (CSC + ACCS)
869         TOC_p = PBCP - HC_s
870
```

```
        #Spacer:
        HS_s = ACCS * (TOC_p - TOC)
        RS_dp_h = V_s - HS_s
        LC_dp_h = RS_dp_h / ACDP
        TOS = TOC - LC_dp_h
        VS_s = CSC * (TOC_p - TOC)
        VS_dp = DP_CAP * (TOC - TOS)
        TSV_s = VS_dp + VS_s

        #Mud Displacement:
        V_md = TOS * DP_CAP

        output_values = OrderedDict({
              "Capacities": "",
              "Hole Capacity [bbl/ft]":  round(Hole_Cap,4),
              "Cement stringer capacity [bbl/ft]": round(CSC,4),
              "Annular capacity b/w hole & cement stinger [bbl/ft]"
: round(ACCS,4),
              "Annular capacity b/w hole & drill pipe [bbl/ft]":
round(ACDP,4),
              "Drill pipe capacity [bbl/ft]": round(DP_CAP,4),
              "Cement": "",
              "Volume of cement [bbl]": round(VC,4),
              "Height of cement while cement string in well [ft]":
round(HC_s,4),
              "TOC (when pipe is in hole [ft])": round(TOC_p,4),
              "Spacer": "",
              "Height of spacer while cement string in well [bbl]":
 round(HS_s,4),
              "Rest of spacer b/w drill pipe and hole [bbl]": round
(RS_dp_h,4),
              "Length of spacer b/w drill pipe & hole [ft]": round(
LC_dp_h,4),
              "Top of spacer [ft]": round(TOS,4),
              "Volume of spacer in cement stinger [bbl]": round(
VS_s,4),
              "Volume of spacer in drill pipe [bbl]": round(VS_dp
,4),
              "Total spacer volume in string [bbl]": round(TSV_s,4)
,
              "Mud Displacement": "",
              "Volume of mud displacement [bbl]": round(V_md,4),
              "":"",
              "":"",
              "":"",
              "Pumping Schedule": "",
              "1. Pump Spacer [bbl]": round(V_s,4),
              "2. Pump Cement [bbl]": round(VC,4),
              "3. Pump Spacer [bbl]": round(TSV_s,4),
              "4. Pump Drilling Mud [bbl]": round(V_md,4),
              "5. Pull slowly above TOC": "",
              "6. Circulate bottom up": "",
              "7. Pull out of hole to surface": ""
         })

        for key, value in output_values.items():
            if value == "":
                print(key)
```

```
            else:
                print(key, value)


        self.output_window = OutputWindow(output_values)
        self.output_window.show()

    def retranslateUi(self, Software):
        _translate = QtCore.QCoreApplication.translate
        Software.setWindowTitle(_translate("Software", "PlugDgitize"))
        self.SaveButton.setText(_translate("Software", "Save Data"))
        self.label.setText(_translate("Software", "Well"))
        self.Nameline.setPlaceholderText(_translate("Software", "Name"))
        self.Typeline.setPlaceholderText(_translate("Software", "Type"))
        self.Resline.setPlaceholderText(_translate("Software", "Reservoir"))
        self.label_2.setText(_translate("Software", "Objective/Scope\n"""))
        self.Sumtext.setPlaceholderText(_translate("Software", "Write here..."))
        self.label_3.setText(_translate("Software", "Summary of planned events \n"""))
        self.Objtext.setPlaceholderText(_translate("Software", "Write here..."))
        self.tabWidget.setTabText(self.tabWidget.indexOf(self.tab_3), _translate("Software", "Well Information"))
        self.DisciplineLabel.setText(_translate("Software", "Discipline"))
        self.EventLabel.setText(_translate("Software", "Event"))
        self.ObjectiveLabel.setText(_translate("Software", "Objective"))
        self.SectionLabel.setText(_translate("Software", "Section"))
        self.label_9.setText(_translate("Software", "Contracts"))
        self.label_7.setText(_translate("Software", "Engineering"))
        self.tabWidget.setTabText(self.tabWidget.indexOf(self.tab), _translate("Software", "Main Window"))
        self.Info_Label.setText(_translate("Software", "<html><head/><body><p align=\"justify\"><span style=\" font-size:10pt; font-weight:600;\">Welcome to the Cement Plugging Software!</span></p><p align=\"justify\"><span style=\" font-weight:600;\">Overview:</span></p><p align=\"justify\">The Cement Plugging Software is a cutting-edge tool designed specifically for optimizing cement plugging using the Balanced Plug method within plug and abandonment (P&amp;amp;A) operations. By leveraging digitalization and advanced algorithms, this software enhances efficiency, accuracy, and decision-making, ensuring superior zonal isolation and wellbore integrity.</p><p align=\"justify\"><span style=\" font-weight:600;\">Purpose:</span></p><p align=\"justify\">The software is tailored to provide a comprehensive and user-friendly environment for optimizing cement plugging techniques using the Balanced Plug method. It focuses specifically on cementing strategies utilizing oil-
```

```
based mud. By adhering to this specific method and mud type,
the software offers precise calculations that align with
industry standards and best practices.</p><p align=\"justify
\"><span style=\" font-weight:600;\">Calculations:</span></p><p
 align=\"justify\">When utilizing the software, users input
relevant wellbore parameters specific to the Balanced Plug
method and oil-based mud. The software performs intricate
calculations based on these inputs to determine the optimal
volumes of cement required. By incorporating equations from the
 &amp;quot;drilling-info&amp;quot; website, the software ensures
accurate estimation of cement volumes, taking into account
factors such as casing size, casing shoe depths (MD and TVD),
hole size, hole depths (MD and TVD), pay sand depth range,
tubing, tubing ID, drill pipe OD, and drill pipe ID. Through
these calculations, the software enables users to achieve
precise and effective cement placement, supporting superior
zonal isolation and wellbore integrity.</p><p align=\"justify
\">By utilizing the Cement Plugging Software, users can
optimize cement plugging using the Balanced Plug method with
oil-based mud. This software empowers users to make informed
decisions, enhance efficiency, and ensure superior zonal
isolation and wellbore integrity within plug and abandonment
operations.</p></body></html>"))
948         self.CPTab.setTabText(self.CPTab.indexOf(self.CP1Tab),
    _translate("Software", "Information"))
949         self.HDMDLineEdit.setPlaceholderText(_translate("Software"
    , "[feet]"))
950         self.PSDRLineEdit.setPlaceholderText(_translate("Software"
    , "[feet]"))
951         self.PSDRLabel2.setText(_translate("Software", "Pay Sand
    Bottom Depth"))
952         self.PSDRLabel2.setAccessibleName(_translate("Software", "
    [feet]"))
953         self.PSDRLineEdit2.setPlaceholderText(_translate("Software
    ", "[feet]"))
954         self.CasingShoeMDLabel.setText(_translate("Software", "
    Casing Shoe Depth (MD)"))
955         self.TubeODLineEdit.setPlaceholderText(_translate("
    Software", "[inches]"))
956         self.TubeIDLabel.setText(_translate("Software", "Drill
    Tube ID"))
957         self.TubeIDLineEdit.setPlaceholderText(_translate("
    Software", "[inches]"))
958         self.CasingShoeMDLineEdit.setPlaceholderText(_translate("
    Software", "[feet]"))
959         self.TubeODLabel.setText(_translate("Software", "Drill
    Tube OD"))
960         self.PipeIDLabel.setText(_translate("Software", "Drill
    Pipe ID"))
961         self.CasingSizeLabel.setText(_translate("Software", "
    Casing Size"))
962         self.PipeIDLineEdit.setPlaceholderText(_translate("
    Software", "[inches]"))
963         self.PipeODLabel.setText(_translate("Software", "Drill
    Pipe OD"))
964         self.PipeODLineEdit.setPlaceholderText(_translate("
    Software", "[inches]"))
965         self.HoleDepthMDLabel.setText(_translate("Software", "Hole
```

```
 Depth (MD)"))
966         self.HDTVDLineEdit.setPlaceholderText(_translate("Software
    ", "[feet]"))
967         self.CSLineEdit.setPlaceholderText(_translate("Software",
    "[inches]"))
968         self.CasingShoeTVDLabel.setText(_translate("Software", "
    Casing Shoe Depth (TVD)"))
969         self.HoleSizeLabel.setText(_translate("Software", "Hole
    Size"))
970         self.CasingShoeTVDEditLine.setPlaceholderText(_translate("
    Software", "[feet]"))
971         self.HoleSizeLineEdit.setPlaceholderText(_translate("
    Software", "[inches]"))
972         self.HoleDepthTVDLabel.setText(_translate("Software", "
    Hole Depth (TVD)"))
973         self.PSDRLabel.setText(_translate("Software", "Pay Sand
    Top Depth"))
974         self.CPTab.setTabText(self.CPTab.indexOf(self.CP2Tab),
    _translate("Software", "Diagram Input"))
975         self.SpacerDensLabel.setText(_translate("Software", "
    Density of Spacer"))
976         self.SpacerDensLineEdit.setPlaceholderText(_translate("
    Software", "[ppg]"))
977         self.SpacerVLineEdit.setPlaceholderText(_translate("
    Software", "[bbl]"))
978         self.SpacerVLabel.setText(_translate("Software", "Volume
    of Spacer"))
979         self.CPTab.setTabText(self.CPTab.indexOf(self.CP3Tab),
    _translate("Software", "Spacer Design"))
980         self.CementDensLineEdit.setPlaceholderText(_translate("
    Software", "[ppg]"))
981         self.CementDensLabel.setText(_translate("Software", "
    Required Cement Density"))
982         self.ExcessLineEdit.setPlaceholderText(_translate("
    Software", "[%]"))
983         self.HightCementLabel.setText(_translate("Software", "
    Hight of Cement Stinger"))
984         self.CementLengthLineEdit.setPlaceholderText(_translate("
    Software", "[feet]"))
985         self.TOCLabel.setText(_translate("Software", "TOC with
    Stinger"))
986         self.CementLengthLabel.setText(_translate("Software", "
    Planned Cement Length"))
987         self.TOCLineEdit.setPlaceholderText(_translate("Software",
     "[feet]"))
988         self.ExcessLabel.setText(_translate("Software", "Excess"))
989         self.BottomLineEdit.setPlaceholderText(_translate("
    Software", "[feet]"))
990         self.HightCementLineEdit.setPlaceholderText(_translate("
    Software", "[feet]"))
991         self.BottomLabel.setText(_translate("Software", "Planned
    Bottom of Cement Plug"))
992         self.CPTab.setTabText(self.CPTab.indexOf(self.CP4Tab),
    _translate("Software", "General Input"))
993         self.MudTypeLabel.setText(_translate("Software", "Mud Type
    "))
994         self.MudDensLineEdit.setPlaceholderText(_translate("
    Software", "[ppg]"))
```

```
995         self.MudDensLabel.setText(_translate("Software", "Mud
    Density"))
996         self.MudTypeComboBox.setItemText(0, _translate("Software",
     "Oil Based"))
997         self.MudTypeComboBox.setItemText(1, _translate("Software",
     "Water Based"))
998         self.MudTypeComboBox.setItemText(2, _translate("Software",
     "Custom"))
999         self.CalcButton.setText(_translate("Software", "Calculate"
    ))
1000        self.CPTab.setTabText(self.CPTab.indexOf(self.CP5Tab),
    _translate("Software", "Drilling Fluid"))
1001        self.tabWidget.setTabText(self.tabWidget.indexOf(self.
    tab_2), _translate("Software", "Cement Plugging"))
1002        self.CasingRunningLabel.setText(_translate("Software", "
    Coming Soon..."))
1003        self.ContractsTab.setTabText(self.ContractsTab.indexOf(
    self.tab_5), _translate("Software", "Casing Running"))
1004        self.CemServLabel.setText(_translate("Software", "Coming
    Soon..."))
1005        self.ContractsTab.setTabText(self.ContractsTab.indexOf(
    self.tab_6), _translate("Software", "Cementing Services"))
1006        self.CTLabel.setText(_translate("Software", "Coming Soon
    ..."))
1007        self.ContractsTab.setTabText(self.ContractsTab.indexOf(
    self.tab_7), _translate("Software", "Coiled Tubing"))
1008        self.EnvLabel.setText(_translate("Software", "Coming Soon
    ..."))
1009        self.ContractsTab.setTabText(self.ContractsTab.indexOf(
    self.tab_8), _translate("Software", "Environmental and Waste
    Control"))
1010        self.MLLabel.setText(_translate("Software", "Coming Soon
    ..."))
1011        self.ContractsTab.setTabText(self.ContractsTab.indexOf(
    self.tab_9), _translate("Software", "Mud Logging"))
1012        self.RigLabel.setText(_translate("Software", "Coming Soon
    ..."))
1013        self.ContractsTab.setTabText(self.ContractsTab.indexOf(
    self.tab_10), _translate("Software", "Rig Contract"))
1014        self.WellTLabel.setText(_translate("Software", "Coming
    Soon..."))
1015        self.ContractsTab.setTabText(self.ContractsTab.indexOf(
    self.tab_11), _translate("Software", "Well Testing"))
1016        self.WHLabel.setText(_translate("Software", "Coming Soon
    ..."))
1017        self.ContractsTab.setTabText(self.ContractsTab.indexOf(
    self.tab_12), _translate("Software", "WH / XT"))
1018        self.tabWidget.setTabText(self.tabWidget.indexOf(self.
    tab_4), _translate("Software", "Contracts"))
1019        self.actionSave.setText(_translate("Software", "Save"))
1020        self.actionOpen.setText(_translate("Software", "Open"))
1021        self.actionHome.setText(_translate("Software", "Home"))
1022
1023
1024 if __name__ == "__main__":
1025     app = QtWidgets.QApplication(sys.argv)
1026     Software = QtWidgets.QMainWindow()
1027     ui = Ui_Software()
```

```
1028     ui.setupUi(Software)
1029     Software.show()
1030     sys.exit(app.exec_())
```