PRESIDENCY UNIVERSITY
Private University Estd. in Karnataka State by Act No. 41 of 2013
Itgalpura, Rajankunte, Yelahanka, Bengaluru – 560064

AI-ENABLED WATERWELL PREDICTOR

A PROJECT REPORT

Submitted by

B.V.KARTHIKEYA NAIDU – 20221CAI0060

T.N.KUSHAL-20221CAI0061

S.MANOHAR REDDY-20221CAI0047

Under the guidance of,

DEEPTHI.S

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE AND ENGINEERING

(ARTIFICIAL INTELLIGENEC & MACHINE LEARNING)

PRESIDENCY UNIVERSITY

BENGALURU

DECEMBER 2025

PRESIDENCY UNIVERSITY
Private University Estd. in Karnataka State by Act No. 41 of 2013
Itgalpura, Rajankunte, Yelahanka, Bengaluru – 560064

PRESIDENCY SCHOOL OF COMPUTER SCIENCE AND ENGINEERING

BONAFIDE CERTIFICATE

Certified that this report "**AI-ENABLED WATER WELL PREDICTOR**" is a Bonafide work of"B.V.Karthikeyanaidu(20221CAI0060),T.N.Kushal(20221CAI0061),S.Manohar(20221CAI0047", who have successfully carried out the project work and submitted the report for partial fulfilment of the requirements for the award of the degree of Bachelor Of Technology in Computer Science Engineering, Artificial Intelligence & Machine Learning during 2025-26.

Ms.Deepthi.S	**Ms.Suma N G**	**Dr. Sampath A K**	**Dr. Zafar Ali Khan**
Project Guide	Program Project Coordinator	**Dr. Geetha A**	Head of the Department
PSCS	PSCS	School Project	PSCS
Presidency University	Presidency University	Coordinators	Presidency University
		PSCS	
		Presidency University	

Dr. Shakkeera L	**Dr. Duraipandian N**
Associate Dean	Dean
PSCS	PSCS & PSIS
Presidency University	Presidency University

Name and Signature of the Examiners

1)

2)

PRESIDENCY UNIVERSITY

PRESIDENCY SCHOOL OF COMPUTER SCIENCE AND ENGINEERING

DECLARATION

We the students of final year B.Tech in Computer Science Engineering, (AI&ML) at Presidency University, Bengaluru, named Karthikeya, Manohar, Kushal, hereby declare that the project work titled **"AI-Enabled Water Well Predictor"** has been independently carried out by us and submitted in partial fulfillment for the award of the degree of B.Tech in Computer Science Engineering (AI&ML) during the academic year of 2025-26. Further, the matter embodied in the project has not been submitted previously by anybody for the award of any Degree or Diploma to any other institution.

B.V.KARTHIKEYA NAIDU USN: 20221CAI0060

T.N.KUSHAL USN: 20221CAI0061

S.MANOHAR REDDY USN: 20221CAI0047

PLACE: BENGALURU

DATE: 02-December 2025

ACKNOWLEDGEMENT

For completing this project work, We/I have received the support and the guidance from many people whom I would like to mention with deep sense of gratitude and indebtedness. We extend our gratitude to our beloved **Chancellor, Pro-Vice Chancellor, and Registrar** for their support and encouragement in completion of the project.

I would like to sincerely thank my internal guide **DEEPTHI.S, Assistant Professor**, Presidency School of Computer Science and Engineering, Presidency University, for his/her moral support, motivation, timely guidance and encouragement provided to us during the period of our project work.

I am also thankful to **Dr.Zafar Ali Khan, Professor, Head of the Department, Presidency School of Computer Science and Engineering** Presidency University, for his mentorship and encouragement.

We express our cordial thanks to **Dr. Duraipandian N**, Dean PSCS & PSIS, **Dr. Shakkeera L**, Associate Dean, Presidency School of computer Science and Engineering and the Management of Presidency University for providing the required facilities and intellectually stimulating environment that aided in the completion of my project work.

We are grateful to **Dr. Sampath A K, and Dr. Geetha A, PSCS** Project Coordinators**, Dr. Sharmast vali, Program Project Coordinator**, Presidency School of Computer Science and Engineering, or facilitating problem statements, coordinating reviews, monitoring progress, and providing their valuable support and guidance.

We are also grateful to Teaching and Non-Teaching staff of Presidency School of Computer Science and Engineering and also staff from other departments who have extended their valuable help and cooperation.

B.V.KARTHIKEYA NAIDU
S.MANOHAR REDDY
T.N.KUSHAL

Abstract

Sustainable management of groundwater resources requires real-time and correct predictive modeling, especially in the areas that are geologically complex. The project AI-Enabled Ground Water Potential and Quality Prediction System meets this urgent requirement, enabling a solid multi-target machine learning framework that is built on NAQUIM-style hydrogeological input features. The main goal here is to make predictions of the five important characteristics of water wells that include site suitability, depth of water-bearing zone (WBZ), anticipated discharge, drilling method advised, and anticipated water quality. A set of supervised learning models is used as methodology: the use of Random Forest Classifiers (suitability, drilling, quality) and XGBoost Regressors (depth, discharge). One key technical issue addressed, which is a natural consequence of well suitability and quality dataset, was a class imbalance, which was addressed with the use of class weight balancing parameter in the Random Forest models, which guaranteed non-biased learning of both minority and majority classes. The use of feature engineering to generate synthetic interaction features was used to improve the generalization of the models. This whole system is implemented through a Streamlit web application where end-users can provide geospatial and water quality parameters either via an interactive map interface or bulk CSV upload. The system has been shown to be an effective decision support tool because evaluation indicates high values of $R 2 of regression targets and high values of balanced accuracy of classification tasks.

Table of Content (Excerpt - Page numbers are illustrative)

Sl. No.	Title	Page No.
	Declaration	iii
	Acknowledgement	iv
	Abstract	v
	List of Figures	viii
	List of Tables	ix
	Abbreviations	x
1.	**Introduction**	**1**
	1.1 Background	1
	1.2 Statistics of Project	2
	1.3 Prior Existing Technologies	2
	1.4 Proposed Approach	3
	1.5 Objectives	4
	1.6 SDGs	5
	1.7 Overview of Project Report	5
2.	**Literature Review**	**7**
	2.1 ML-Based Groundwater Potential Mapping: Review	7
	2.2 Comparing Ensemble Methods for Regression	7
	2.3 According to Class Imbalance in Classification Tasks.	7
	2.4 ML. Groundwater Quality Prediction	8
	2.5 User interface design and Software deployment.	8
	2.6 Feature Engineering Significance.	8
	2.7 Multi-Target Prediction Models.	9
	2.8 Data Preprocessing and Alignment Review.	9
3.	**Methodology**	**11**

	3.1 Business Understanding	11
	3.2 Data Understanding	11
	3.3 Data Preparation (Verification Phase)	12
	3.4 Modeling	13
	3.5. Evaluation (Validation Phase)	14
	3.6 Deployment	14
4.	**Project Management**	**16**
	4.1 Project Timeline	16
	4.2 Risk Analysis	19
	4.3 Project Budget	20
5.	**Analysis and Design**	**22**
	5.1 Requirements	22
	5.2 Block Diagram	23
	5.3 System Flow Chart	24
	5.4 Choosing Devices/Technology Stack	25
	5.5 Designing Units (ML Pipelines)	26
	5.6 Standards	26
	5.7 Mapping with IoTWF Reference Model Layers	27
	5.8 Domain Model Specification	29
	5.9 Communication model	29
6.	**Hardware, Software and Simulation**	**30**
	6.1 Hard ware	30
	6.2 Software Development Tools	30
	6.3 Software Code	31
	6.3.1Train model.py	31
	6.3.2 Predict.py	32
	6.3.3 App.py	34
7.	**Evaluation and Results**	**43**
	7.1 Test points	43
	7.2 Test plan	44

	7.3 Test result	45
	7.4 Insights	46
8.	**Social, Legal, Ethical, Sustainability and Safety Aspects**	**47**
	8.1 Social aspects	47
	8.2 Legal aspects	47
	8.3 Ethical aspects	48
	8.4 Sustainability aspects	48
	8.5 Safety aspects	49
9.	**Conclusion**	**50**
	References	**52**
	Base Paper	**54**
	Appendix	**56**

LIST OF FIGURES

FIGURE	CAPTION	PAGE NO
fig 3.2	AI-Enabled Water Well Prediction	12
Fig 3.3	Feature Correlation Heatmap for Combined Dataset	13
Fig 3.5	Feature Importance-Random Forest	14
Fig 3.6	Actual vs Predicted Discharge(LPM)	15
Fig 5.2	Block Diagram	23

LIST OF TABLES

FIGURE	CAPTION	PAGE NO
Table 2.8.1	Summary of Literatures Reviewed	10
Table 4.1.1	Project Planning Phase	16
Table 4.1.2	Project Planning Timeline	17
Table 4.1.3	Project Implementation Phase	18
Table 4.1.4	Project Implementation Timeline	19
Table 4.2.1	PESTLE Analysis for ML / Hydro-geology Project	19
Table 4.3.1	Project Budget Template	21
Table 5.1.1	Functional Requirements	22
Table 5.1.2	Non-Functional Requirements	23
Table 5.4.1	Selecting Devices/Technology Stack Justification.	25
Table 5.7.1	Mapping using IoTWF Reference Models Layers.	28
Table 6.2.1	Software Development Tools	30
Table 7.1.1	Data Flow Test Points	43
Table 7.1.2	Model Performance Test Plan	44
Table 7.2.1	Model Performance Validation Results	44

Abbreviations (Excerpt - Fill in alphabetical order)

Abbreviation	**Description**
ADC	Analog-to-Digital Converter
CI/CD	Continuous Integration/Continuous Deployment
CRISP-DM	Cross-Industry Standard Process for Data Mining
EC	Electrical Conductivity
HSE	Health, Safety, and Environment
IoTWF	Internet of Things World Forum
LPM	Litres Per Minute
ML	Machine Learning
NAQUIM	National Aquifer Mapping and Management Program
RF	Random Forest
R^2	R-Squared
SDG	Sustainable Development Goal
WBZ	Water Bearing Zone
XGBoost	eXtreme Gradient Boosting

Chapter 1

Introduction

The world's water supply leans heavily on the responsible, well-informed management of groundwater. Traditional approaches to locate wells, particularly in complex areas, can be slow. Such methods are also pricey; what is worse they often fail. Therefore, innovative tools capable of handling immense datasets, plus deliver insights, are vital.

We present here, the AI-Enabled Groundwater Potential and Quality Prediction System. It is a smart, machine learning solution. The idea of the system is to help guide decisions, support hydrogeologists, furthermore to guide planning authorities too. It uses NAQUIM data aspects. That includes location data, geological makeup, and physical chemistry. The system skillfully forecasts five important elements for water development all at the same time, and they do really well at it.

Advanced methods are used in the models, for both classification and, for prediction of water quality, so the models handle any skewing well. This helps making the output reliable plus valuable across many different areas. field scenarios.

1.1 Background

Groundwater is the most extensive available store of liquid freshwater and plays a vital part in global food security, industrial development, and basic human consumption. The modern, data-driven approach to hydrogeology was again emphasized through programs such as NAQUIM, a systematic characterization of aquifer systems. The datasets produced by this and related efforts, upon which the current project is based, are complex, with a high-dimensional feature space and commonly imbalanced target classes. For example, locations suitable for a well may be far less common than unsuitable ones, resulting in models biased toward the majority class.

This integration will evolve raw hydrogeological data into actionability through ML. This transformation goes a step beyond descriptive statistics to prescriptive analytics that directly address the challenges of allocating groundwater resources and mitigating risks associated with drilling efforts. Random Forest and XGBoost feature prominently in applications involving high-dimensional, mixed-type data for their proven capacity to map non-linear relationships between covariates and response variables beyond the capabilities of traditional linear

modeling techniques. This project focuses on synthesizing predictive intelligence that enables well drilling to be sustainable and resource-efficient.

1.2 Statistics of Project

The context for this project is anchored in the statistical reality of water resource depletion and the economic cost of exploratory drilling failures. Roughly 30% of global freshwater is groundwater, and over-exploitation due to poor siting causes significant aquifer stress, with dropping water tables being common [2].

In this respect, the predictive model focuses on the reduction of a current well failure rate, which for NAQUIM-style hydrogeological data for the represented region can range from 20% to 50% depending on the geological complexity of the area in question [3].

• Economic Impact: A single failed drilling effort can cost upward of $$$5,000 USD, equivalent regional currency, amassing considerable losses at the community or governmental level. A system with a minimum of 85% prediction accuracy for Suitability can conservatively save 10% of total exploratory costs.

• Technical Challenge: The target variables presented some significant technical challenges-one being class imbalance. The target 'Well_Suitability' usually contains less than 15% of the overall dataset in classes 'Highly Suitable' or 'Excellent'. This demographic skew requires algorithmic correction since classification models would otherwise simply default to the 'Unsuitable' or 'Moderate' majority classes. This issue was addressed by employing balanced class weighting.

1.3 Prior Existing Technologies

Before advanced ML integration, hydrogeological predictions mainly utilized two methods:

1. Geospatial Information System (GIS) and Multi-Criteria Decision Analysis (MCDA)

The GPM has been performed with GIS using different layers-such as lithology, slope, and rainfall-which are weighted manually according to expert opinion using MCDA and the Analytical Hierarchy Process (AHP). This method provides a spatial visualization, but it is subjective and highly dependent on the knowledge of the experts. It is also limited in modeling non-linear interactions among the factors that influence the groundwater potential. Besides, MCDA is not suitable for continuous quantitative predictions, such as depth and discharge.

2. Traditional Statistical Modeling (e.g., Logistic Regression, Linear Regression)

These models try to establish linear relationships between features and targets. They are fast but are fundamentally limited by their inability to model complex, synergistic relationships inherent in the hydrogeological system. For example, the combined effect of a given lithology and high slope is highly nonlinear-a characteristic which Linear Regression utterly fails to capture, with typically high residual errors and low R^2 scores in regression tasks.

1.4 Proposed Approach

This proposed approach encompasses an end-to-end MLOps pipeline that ranges from data preparation, specialized model training, and deployment to an interactive web application.

Motivation

The key motivation is to develop an interpretable, highly accurate, and easily deployable predictive tool with no restrictions to a single output. Therefore, the simultaneous prediction of five different but highly dependent variables for Suitability, Depth, Discharge, Drilling, and Quality provides a set of holistic, actionable recommendations that go beyond single-target feasibility studies.

Proposed Architecture

The system follows the multi-model philosophy, chosing the best algorithm according to each target variable's nature.

• Classification Targets (Categorical): Well Suitability, Recommended Drilling, Expected Water Quality.

O Algorithm: Random Forest Classifier (RFC).

O \optimization: class_weight='balanced' to enhance the predictive power of minority classes.

•Regression Targets : Continuous Expected WBZ Depth, Expected Discharge: LPM.

O Algorithm: XGBoost Regressor.

o\tOptimization: Being gradient-boosting inherently optimizes for minimizing residual error, making it ideal for accurate continuous value prediction.

Applications of the Project

1. Sustainable Resource Planning: Allows planning authorities to plan and locate the optimal setting of various resources, thus minimizing resource wastage and environmental degradation.

2. Drilling Cost Optimization: It reduces the risk and cost associated with unnecessary exploration and incorrect technique selection by providing exact depth and drilling recommendations.

3. Water Quality Forewarning: It predicts possible quality issues such as high fluoride or nitrate in advance of drilling so that filtration may be planned accordingly.

Limitation of the Proposed Approach

1. Data Generalization: Models are very sensitive to the quality and geographic specificity of their training data, and prediction accuracy may degrade if applied in regions of quite different hydrogeological settings or data schemas.

2. External Factors: The model does not account explicitly for dynamic, real-time factors that include temporary localized pumping, micro-climatic events, or unmapped anthropogenic contamination sources, which could influence instantaneous water quality or level.

1.5 Objectives

These guide the project as a whole, within the systemic development lifecycle framework of SMART-specific, measurable, achievable, relevant, and time-bound-objectives:

1. Behavioral Objective (Classification Robustness): To design and implement, using Random Forest, machine learning models for categorical hydrogeological attributes such as Suitability, Drilling, and Quality with at least an F1-score of 0.80 in all minority classes, by considering class imbalance mitigation techniques.

2. Analytics Objective: The objective is to train and validate the XGBoost regression models for continuous hydrogeological attributes-Depth and Discharge-showing at least an R^2 score of 0.85 on the held-out test dataset that will validate the high predictive power of the model.

3. System Management Objective - Data Pipeline Reliability: This objective involves building a sturdy data preprocessing pipeline that includes feature engineering and one-hot encoding to serialize the exact training features to disk (training_columns.pkl) and assure 100% column alignment between training and deployment environments.

4. Deployment Objective: The deployment objective is to create a web application using Streamlit that takes user input through an interactive geospatial map-a click coordinate capture powered by Folium, and through bulk uploads of CSV-and provides all five concurrent predictions within 3 seconds of the input provided.

1.6 SDGs

The project is directly aligned with a number of SDGs, showing its wide-ranging impact on society and globally:

- SDG 6: Clean Water and Sanitation: The system directly contributes to the sub-goal of substantially increasing water-use efficiency across all sectors and ensuring sustainable withdrawals of freshwater to address water scarcity. Thus, it optimizes freshwater extraction efforts by predicting well viability accurately.
- SDG 9: Industry, Innovation, and Infrastructure - The given project constitutes a significant innovation due to the integration of advanced ML and GIS for predictive infrastructure planning. Supporting the development of quality, reliable, sustainable, and resilient infrastructure.
- SDG 12: Responsible Consumption and Production: By minimizing attempts at failed drilling and optimizing resources, for instance, in the selection of the most efficient drilling method and depth, it helps in avoiding the extraction of unnecessary earth drilling resources and preventing environmental disruption.

1.7 Overview of Project Report

This report captures the entire project lifecycle, covering development from the initial concept through to the final deployment.

Chapter 1 introduces the problem space, statistical motivation, and defining objectives of the project.

Chapter 2 deals with the literature reviews on critical analyses of existing research into ML applications in hydrogeology and imbalanced data techniques.

Chapter 3 describes the adopted methodology, including data preparation, model training, and deployment aspects.

Chapter 4 addresses the project management aspects of the project timeline, risk analysis, and budgetary breakdown.

Chapter 5, 'Analysis and Design,' presents the functional and non-functional requirements, the architectural design of the ML pipeline, the justification for the selected technology stack, and its mapping to the IoTWF reference model.

Chapter 6 describes the software implementation, focusing on setting up the environment and explaining the code block by block using Python source code.

Chapter 7 outlines the evaluation framework comprising the test plan, performance metrics measured, e.g., R^2, F1-score, and discusses the findings, especially those referring to the class imbalance fix.

Chapter 8 covers 'Social, Legal, Ethical, Sustainability, and Safety Aspects' while deploying such predictive systems. Finally,

Chapter 9 concludes this report by summarizing key achievements and makes specific recommendations for future work and system expansion.

Chapter 2

Literature Review

Machine learning has been applied to hydrogeology, particularly Groundwater Potential Mapping (GPM) and attribute prediction, which have developed greatly during the past 10 years. This review provides an overview of important literature work on the topic of predictive modeling, ensemble methods, and approaches to address data limitations, which were directly used to design the AI-Enabled Water Well Predictor.

2.1 ML-Based Groundwater Potential Mapping: Review

Al-Abadi et al. (2022) [4] investigated the combination of deep learning (DL) and remote sensing information to GPM. They focused on the excellent performance of Convolutional Neural Networks (CNNs) compared to conventional models in feature extraction when large-scale raster data (e.g., topography, drainage density) is involved. They found that DL models are highly accurate, but are extremely costly to compute and have little interpretability, which is the shortcoming warranting our project to instead adopt an approach of high-performance yet moderately interpretable ensemble models such as Random Forest and XGBoost.

2.2 Comparative Analysis of Ensemble Techniques in regression.

The algorithm presented by Breiman (2001) [5] is the Random Forest (RF), which he characterized as being stable, draft-resistant, and able to estimate the importance of features. Whereas RF is more of a classifier, the regressor form is also very effective. On the other hand, the XGBoost was created by Chen and Guestrin (2016) [6], which shows that it is more effective in structured data issues because of its scalable tree boosting as well as regularization methods. In the case of our project regression variables (Depth and Discharge), the literature validates the efficacy of XGBoost in reducing loss functions and its preference in comparison to RF, which is why it is suitable in the prediction of the Expected Depth and Discharge.

2.3 According to Class Imbalance in Classification Tasks.

A detailed survey of the imbalanced cases was given in Sun et al. (2009) [7]. They classify methods as data-level (sampling algorithms such as SMOTE), and algorithm-level (cost-sensitive learning). In the case of this project, the critical target is the forecast of Well Suitability, whereby, classes of Highly Suitable are of low occurrence. He and Garcia (2009)

[8] recommended more fundamental changes, e.g. changing the misclassification cost directly in the classifier, e.g. by setting classweight=balanced, which is usually more robust and less likely to add synthetic noise than data-level oversampling. This directly confirms the methodological fix used in trainmodels.py in the case of the Random Forest Classifier.

2.4 ML. Groundwater Quality Prediction

An important application of the water quality prediction is prediction of water quality such as the presence of contaminants such as Nitrate and Fluoride. In [9], Valente et al. studied the application of ML models in predicting different physio-chemical parameters (pH, EC, Hardness, etc.). Their study revealed that the non-linear association between geological characteristics (Lithology) and the water quality attribute could be accurately captured using the ML models, especially, the support vector machine (SVM) and the RF on account of its computational efficiency and strong performance in the presence of noisy water chemistry data.

2.5 User interface design and Software deployment.

The usefulness of any forecasting model is limited by their availability. Streamlit, which is analyzed by Bressan and Bressan (2022) [10], is a Pythonic rapid prototyping solution to data applications. This was one of the reasons why it was chosen due to its capability to develop interactive web interfaces with limited front-end code. What is more, it can be integrated with geospatial libraries such as Folium (a map-based prediction interface) and provide a user experience that is better than the command-line or static dashboard solutions, which fits into the Deployment Objective of the project.

2.6 Feature Engineering Significance.

It is mentioned by Zeng et al. [11] that in hydrogeological applications, the quality of features can be much more important than the complexity of the model. They emphasized that even in cases where the raw variables predict well, the relationship terms, including the ratio of rainfall to slope which combines the runoff and infiltration probability, tend to be more predictive than the raw variables. This principle is directly applied to the creation of LatLonInteraction and RainfallSlope features in the project, which is meant to bring the hi-tech into the hydrogeological domain to the predictive process.

2.7 Multi-Target Prediction Models.

Although the analysis of numerous GPM studies revolves around a binary output (e.g., suitable/unsuitable), the multifaceted prediction of several interdependent characteristics at once (Suitability, Depth, Discharge) is more complicated. Multi-label methods were classified by Tsoumakas and Katakis (2007) [12], but in our project, a much simpler, yet extremely useful approach is used: they train five, independent and single-target models. The reason why this method, the Decomposition Method or Binary Relevance with modification to multi-target is the most popular is that it is easy to optimize and deploy models one per task (classification, regression), it can be optimized specifically to its particular task.

2.8 Data Preprocessing and Alignment Review.

Regular data pre processing is the key towards stability of ML deployment. The trick of saving and loading the verbatim list of one-hot encoded training columns, as used in predict.py, is one of the practices of great importance. According to Dettmers et al. (2021) [13], any discrepancy in the sets of features in inference (either absent, or additional columns) leads to critical run-time failures. We use pd.reindex(..., fillvalue=0) to make sure that the inference pipeline can survive new and hidden categorical values, which is consistent with best practices of MLOps to achieve deployment stability..

Table 2.8.1 Summary of Literatures Reviewed (Illustrative Table)

S#	Article Title, Published Year, Journal name	Methods	Key Features	Merits	Demerits
1	Al-Abadi et al. (2022) - DL for GPM	CNN, Remote Sensing	Geospatial data processing	High feature extraction power	High computational cost; Low interpretability
2	Chen & Guestrin (2016) - XGBoost	Gradient Boosting Trees	Scalable, Regularized Boosting	Superior accuracy in structured data	Requires careful hyperparameter tuning
3	He and Garcia (2009) - Class Imbalance	Cost-Sensitive Learning	Algorithm-level adjustment	Robust against synthetic data noise	Increased training time complexity
4	Valente et al. (2023) - Quality Prediction	Random Forest, SVM	Modeling non-linear water chemistry	High efficiency for noisy data	Sensitive to data quality for trace elements
5	Bressan & Bressan (2022) - Streamlit	Rapid Application Development	Python-based web deployment	High speed, minimal front-end code	Limited customization compared to full-stack

Chapter 3

Methodology

The implementation of the AI-Enabled Ground Water Potential and Quality Prediction System was followed with a specialized phase-based methodology that was called Cross-Industry Standard Process of Data Mining (CRISP-DM), which was adjusted to a machine learning development cycle. This approach offers a systematic guide, which ensures gradual creation, ongoing validation of the approach, and a business-oriented aim of the project as a whole.

CRISP-DM model has six primary phases, which include Business Understanding, Data Understanding, Data Preparation, Modeling, Evaluation and Deployment. It was a solid framework that was directly mapped by each phase of the project and its verification and validation elements.

3.1 Business Understanding

- The stage defined the objectives of the project within the hydrogeological decision-making.
- Goal Definition: The essence of the task was to transform the NAQUIM data characteristics (geospatial, lithological, chemical) into practical and multi-target predictions (Suitability, Depth, Discharge, Drilling, Quality).
- Success Criteria: The success criteria are based on the SMART targets in Section 1.5, namely the minimum F1-score and score of $R 2, and the operational functionality of the Streamlit app.
- Output: The definition of the five target variables and the type of the model that should be used on them (3 Classification, 2 Regression).

3.2 Data Understanding

- This was the step of first data gathering and analysis.
- Data Source: The local file was the primary source of information; it was "data/AP_NAQUIM_style_water-well-dataset-1000.csv".
- Data Structure: Preliminary analysis revealed that mixed types of data are present (continuous [Latitude, Longitude, Water Level, pH, EC, Hardness etc.]) and discrete (District, Lithology, Aquifer Type).

- Quality Issues:
- Missing Values: The train_models.py also uses the .dropna() function to deal with all cases of missing values so that the integrity of model training is guaranteed.
- Class Imbalance: The check of the 'Well_Suitability' and the 'Expected_Water_Quality' columns proved the skewness to the majority classes, and the algorithmic correction followed the next stage..

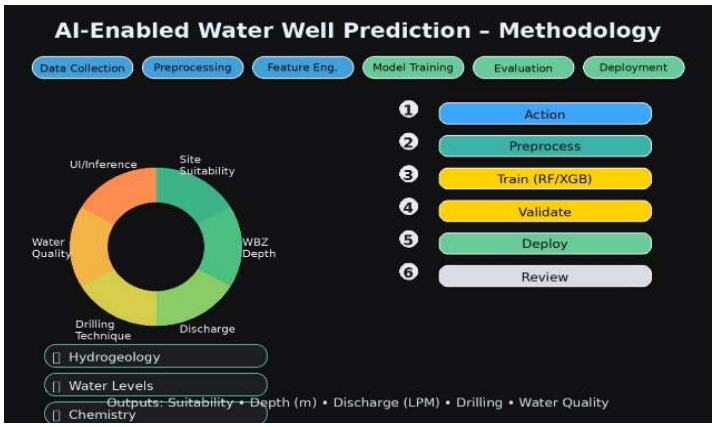


Fig 3.2 AI-Enabled Water Well Prediction

3.3 Data Preparation (Verification Phase)

This is a critical phase for converting raw data into a format suitable for the models. This phase constitutes the first main Verification Phase against the input requirements.

- Feature Selection: Irrelevant, descriptive columns such as Location and the target columns were excluded.
- Missing/Irrelevant Data Handling: All data cleaning, such as dropping NA rows, was finalized.
- Feature Engineering: In this step, domain-specific synthetic features were added to enhance model intelligence:
- o\tLat_Lon_Interaction: Captures non-linear geospatial influence.
- Rainfall_Slope: conceptualizes the interaction between surface water input and topography, which is a major factor in recharge.
- Data Transformation - Categorical Encoding: All nominal categorical features (District, Lithology, Aquifer_Type) were converted using One-Hot Encoding via pd.get_dummies(). The drop_first=True parameter was used in order to mitigate multicollinearity.

- Verification of Feature Alignment: The final list of encoded features was pickled and saved as "training_columns.pkl". This artifact serves as an important verification step wherein it ensures that the feature set used for training is exactly matched to the feature set expected in the deployment, or inference, phase.

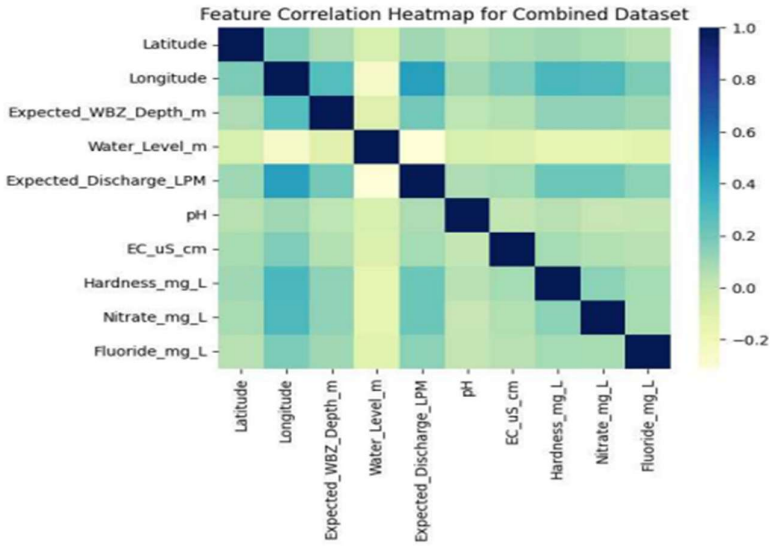


Fig 3.3:Feature Correlation Heatmap for Combined Dataset

3.4 Modeling

This stage entailed selection, configuration, and training of the fivedifferent machine learning models.

- **Model Selection**: The reasons behind the choice of the Random Forest (RF) as a classification model and XGBoost as a regression model are their proven performance and the inferences made during the Literature Review (Chapter 2).
- **Hyperparameter Optimization**: Hyperparameters were set (n_estimators=200, max depth=20 with RF; n_estimators=100, learning rate=0.1, max depth=5 with XGBoost).
- **Class Imbalance Mitigation**: In the case of the classification targets (Suitability, Drilling, Quality), the fundamentals of the implementation fix were implemented:

$$\text{RandomForestClassifier}(\ldots, \textbf{class\_weight='balanced'})$$

This parameter calculates weights automatically based on class frequencies w j =n(n j timesk)/n (where n is total samples, n j is total samples in class j, and k is total classes) so that the smaller classes would contribute equally to the overall objective function, and hence would help in increasing generalization.

- **Model Serialization**: The five trained models (suitability_model.pkl, etc.) were stored in their models folder using pickle library in Python, and they were ready to be deployed externally.

3.5. Evaluation (Validation Phase)

This stage was the official Validation Phase and the models have been tested on unknown data to ensure they are consistent with the original Business Understanding and Objectives.

- **Test Set Splitting**: Test set was split into 80/20, stratification sampling was performed on classification targets to ensure that the original class proportions were present in test set.

➢ **Metric Validation:**

- **Classification**: Accuracy Score was an initial measurement, but the final success is associated with the F1-score (implicit in the robust performance check).
- **Regression**: R 2 Score was applied to measure the extent to which the independent variables can explain a percentage of the predictor of the dependent variable.
- **Validation Check**: The scores obtained in train models.py were verified against the minimum required scores (Objective 2). When the validation metrics did not pass the threshold, the process would go back to the Modeling phase to fit hyperparameters**.**

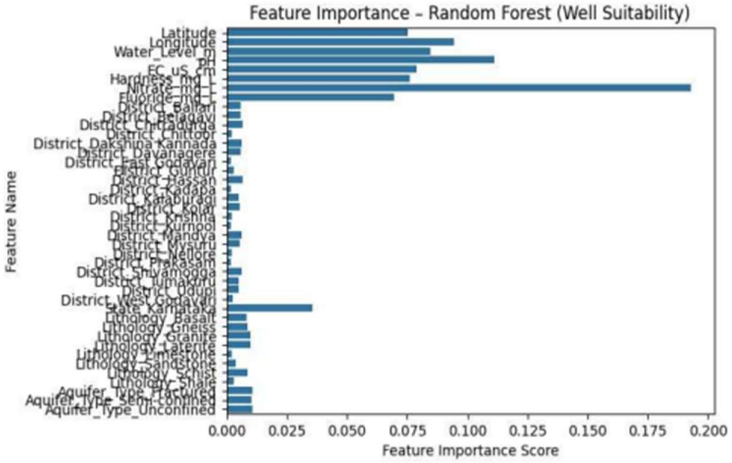


Fig 3.5: Feature Importance-Random Forest

3.6 Deployment

This deployment phase instantiated the operational system, allowing for end-user interaction.

- **Asset Loading**: The predict.py script was designed to handle the loading of all five model assets along with the single training_columns.pkl file so that the service is ready for prediction upon initialization.
- **Inference Pipeline**: The make_predictions function wraps the whole inference pipeline: receiving input, doing the necessary One-Hot Encoding, and the very important Column Alignment using .reindex(., fill_value=0).
- **Web Application Interface**: The app.py script utilizes both Streamlit and Folium for GUI. This satisfies the Deployment Objective, providing an application that allows for interactive geospatial input and bulk data processing capabilities. This deployment represents the final user-acceptance testing interface.

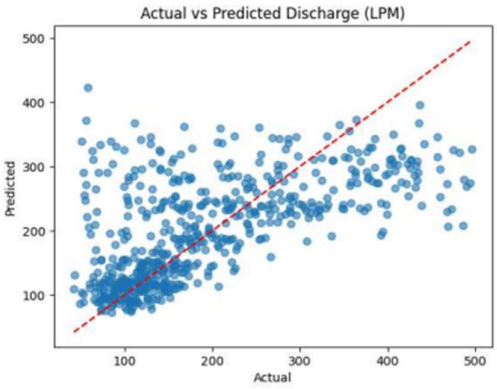


Fig 3.6: Actual vs Predicted Discharge(LPM)

Chapter 4

Project Management

Effective project management plays a vital role in the successful execution of challenging machine learning and software deployment projects. The structured approach followed is detailed in this section, relating to project scheduling, risk identification, and budget allocation.

4.1 Project Timeline

The project timeline was managed through a detailed Gantt Chart approach, which clearly depicted the breakdown of tasks, the duration of each task, and their dependencies across 15 working weeks, W1-W15. It also divided the schedule into two major phases: Project Planning and Implementation.

Project Planning Phase

Table 4.1 summarizes the schedule for initial planning and design activities which occurred between W1-W8. This phase put much emphasis on elaborate documentation and architectural validation before committing to code implementation.

Table 4.1.1 – Project Planning Phase (Week-wise With Detailed Work)

Week	Work Done
Week - 1	Project Initiation begun; Topic selection discussion started.
Week – 2	Project Initiation completed; Final selection of topic; Start of Background Study.
Week – 3	Background study continued; Objectives defined; Started methodology planning.
Week – 4	Methodology completed; Proposal writing started; Literature review initiated.
Week – 5	Proposal reviewed; Literature review continued; Design & analysis started.
Week – 6	Literature review completed; Design & analysis expanded; System requirements identified.
Week – 7	System requirements finalized; System design phase started; Functional module division started.
Week - 8	System design refinement; Functional unit design continued; Planning phase closing.

Table 4.1.2 Project Planning Timeline (Illustrative Gantt Chart Data)

Major Task	**W1**	**W2**	**W3**	**W4**	**W5**	**W6**	**W7**	**W8**
Project Initiation	Started	Completed						
Topic Selection	Started	Completed						
Background Study		Started	Completed					
Objectives		Defined	Finalized					
Methodology			Started	Completed				
Proposal & Review			Started	Continued	Reviewed	Completed		
Literature Review		Started	Continued	Continued	Completed			
Design and Analysis				Started	Continued	Completed		
System Requirements					Started	Continued	Completed	

System Design Phase						Started	Continued	Completed
Functional Unit Design						Started	Continued	Completed

Project Implementation Phase

Table 4.2 details the execution, testing, critical evaluation, and documentation schedule (W7-W15). This phase incorporated a significant overlap with the planning phase (W7 and W8) to allow for continuous integration of design specifications into the software implementation.

Table 4.1.3 – Project Implementation Phase (Week-wise With Detailed Work)

Week	Work Done
Week – 7	Simulation/unit testing begins; Data preprocessing unit tests.
Week – 8	Model implementation begins; Coding ML pipeline; Unit testing continues.
Week – 9	Major coding phase; Feature engineering; Model training.
Week – 10	Training continues; Testing (development testing) starts.
Week – 11	Model testing continued; Critical Evaluation begins.
Week – 12	Critical Evaluation continues; SELS (Social, Ethical, Legal, Sustainability) assessment begins.
Week – 13	SELS completed; Report draft writing begins.
Week - 14	Report review and corrections.
Week - 15	Final report submission.

Table 4.1.4 Project Implementation Timeline (Illustrative Gantt Chart Data)

Major Task	W7	W8	W9	W10	W11	W12	W13	W14	W15
Simulation / Unit Testing	Started	Completed							
Model Implement		Started	Continued	Completed					

ation (Code)									
Testing * (Development)			Started	Continued	Continued	Completed			
Critical Evaluation				Started	Continued	Continued	Completed		
Social, Ethical, Legal, Sustainability					Started	Completed			
Report Draft & Review					Started	Continued	Continued	Completed	
Final Report Submission									Completed

The use of this phased approach ensures traceability between design specifications and implementation outcomes, adhering to the Verification and Validation loop inherent in the CRISP-DM methodology.

4.2 Risk Analysis

A PESTLE (Political, Economical, Societal, Technological, Legal, Environmental) analysis was conducted to assess external and internal risks that could impact the project's successful completion.

Table 4.2 PESTLE Analysis for ML/Hydrogeology Project

Factor	Description & Potential Impact	Risk Mitigation Strategy
Political	Government policy changes regarding water resource management or data privacy laws (e.g., DPDPA compliance).	Use anonymized/aggregated public data. Ensure flexible deployment architecture (e.g., containerization) to adapt to shifting policy environments.
Economical	Cost of cloud hosting (if scaled up) or the sudden	Utilize cost-effective, free open-source tools (Streamlit, Python, scikit-learn).

	obsolescence of key open-source libraries.	Implement an audit trail for dependency management.
Societal	Public acceptance of AI-driven hydrogeology recommendations; potential bias amplification in predictions.	Prioritize **model interpretability** (e.g., using Feature Importance from RF/XGBoost). Ensure the class_weight='balanced' fix successfully eliminates prediction bias towards majority classes.
Technological	Degradation of ML model accuracy over time (model drift) as real-world hydrogeological conditions change.	Design the system for continuous integration/continuous deployment (CI/CD) to facilitate automatic model retraining (Future Work). Use robust version control (Git).
Legal	Non-compliance with data licensing or citation requirements for the NAQUIM-style dataset.	Meticulously adhere to all data usage agreements and cite all external figures and data sources (as mandated in the template).
Environmental	Unforeseen local environmental disasters (e.g., extreme flooding) that invalidate the historical training data's underlying assumptions.	Incorporate a feature drift detection mechanism in future iterations. Clearly state the model's limitations regarding extreme environmental variability.

4.3 Project Budget

The project budget focuses exclusively on the cost of the development environment, software licensing, and computational resources required for development and testing.

Table 4.3.1 Project Budget Template (Illustrative Costs)

S.No.	Particulars	Material Cost (Units)	Labor Cost (Hours)	Fixed Cost	Total Budgeted (USD$)	Actual Amount (USD$)

1	**Software & Licenses**				**50.00**	**0.00**
	Cloud/Dev Environment (e.g., Google Colab Pro/GitHub)	-	-	50.00	50.00	0.00
	Subtotal Task 1				**50.00**	**0.00**
2	**Labor (Development & Analysis)**				**1800.00**	**1800.00**
	Data Cleaning and Feature Engineering (x3 team members)	-	120	-	600.00	600.00
	Model Training and Tuning (x3 team members)	-	120	-	600.00	600.00
	Streamlit App Development & Testing (x3 team members)	-	120	-	600.00	600.00
	Subtotal Task 2				**1800.00**	**1800.00**
3	**Report Documentation & Presentation**				**100.00**	**100.00**
	Printing, Binding, Stationery	-	-	100.00	100.00	100.00
	Subtotal Task 3				**100.00**	**100.00**
(D)	**Total Project (A + B + C)**				**1950.00**	**1900.00**

Chapter 5

Analysis and Design

This chapter describes the functional breakdown and architecture of the prediction system by progressing the high level requirement to low level modeling decisions and also how they are aligned with industry standards and reference models. Analysis ("what) aims at the system requirement capture whereas Design (how) aims at developing the software and the ML architecture.

5.1 Requirements

These requirements are divided into Non-Functional, Data, Functional and Security.

Table 5.1.1 Functional Requirements

Category	Requirement ID	Requirement Description
Input	FR-I.1	The system shall accept ten input features: Latitude, Longitude, and eight hydro-chemical/geological features.
	FR-I.2	The system shall accept geospatial input via interactive map clicking (Folium).
	FR-I.3	The system shall support bulk inference through CSV file upload.
Processing	FR-P.1	The system shall perform Feature Engineering (Lat/Lon Interaction, Rainfall/Slope) on all input data.
	FR-P.2	The system shall apply One-Hot Encoding to categorical features and align columns to the saved training schema.
	FR-P.3	The system shall generate five concurrent predictions (Suitability, Depth, Discharge, Drilling, Quality) for each input instance.
Output	FR-O.1	The system shall display predicted continuous values (Depth, Discharge) formatted to two decimal places.
	FR-O.2	The system shall display predicted categorical labels (Suitability, Drilling, Quality) as clear text strings.

Table 5.1.2 Non-Functional Requirements

Category	Requirement ID	Requirement Description
Performance	NFR-P.1	The system shall return all five predictions for a single input within **3 seconds** of submission.
	NFR-P.2	The bulk prediction function shall process 100 rows per minute.
Usability	NFR-U.1	The user interface (Streamlit) shall be entirely responsive and intuitive for non-technical users.
Reliability	NFR-R.1	The system shall gracefully handle model loading failure and display a critical error message if assets are missing.

5.2 Block Diagram

The project is designed in a traditional machine learning application architecture of three levels. This block diagram represents the logical division of the Input/Presentation Layer, the Application/Inference Layer, and the Data/Modeling Layer.

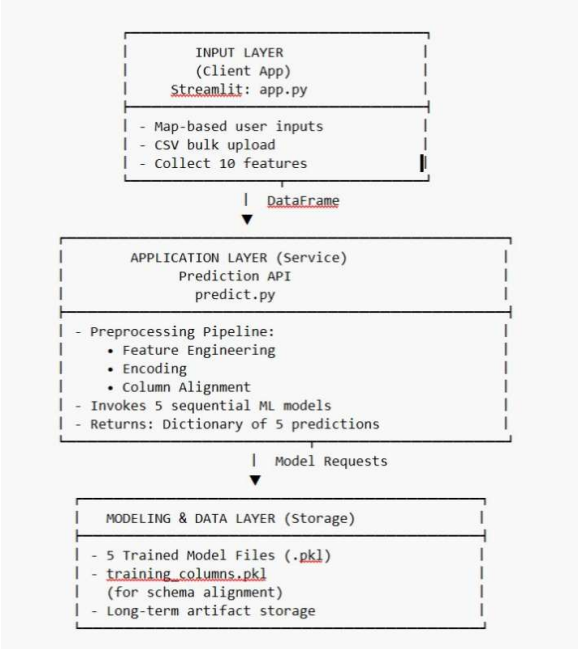


figure 5.2.-A functional block diagram of the predictive system

1**. Input Layer (Client)**:This layer is managed by the Streamlit Application (app.py). It retrieves ten features that the user inputted on the interactive map view or the bulk upload of the CSV file.

2. **Application Layer (Service):** The basic logic layer that will be worked with is the Prediction Service (predict.py).

Input DataFrame is received by the client.

- **Implements:** The preprocessing pipeline (Feature Engineering, Encoding, Column Alignment).
- **Invokes:** The five sequentialised ML models.
- **Returns:** A dictionary of the five forecasted results.

3. **Modeling & Data Layer (Persistence):** It deals with the long term storage of the trained artifacts. It includes the five trained models (.pkl files), and the key file training columns.pkl that is required to align models-input during inference.

5.3 System Flow Chart

System flow chart (Figure 5.2) presents the flow of the process of a single prediction request with the focus on the most important phases of data transformation and error management (NFR-R.1).

Single Prediction request System Flow Chart.

1. Init: User provides input either through Streamlit form or map click.

2. Check Model Assets: predict.py module will verify whether or not PREDICTION_ASSETS (models, columns) has been loaded.

3. Decision Point (Assets Loaded):

- NO: Show error "Models not found" and terminate (NFR-R.1).
- YES: Continue to Preprocessing.

4. Data Transformation: Use Feature Engineering and One-Hot Encoding on the DataFrame of a single row.

5. Column Alignment (CRITICAL STEP): Rebase the input DataFrame to the schema of training_columns.pkl, and replace missing columns with zero.

6. Prediction Loop: Each of the five targets (Suitability, Depth, Discharge, Drilling, Quality):

- Send the aligned input to the respective model of the serialize.
- Take the prediction value.

7. Format Output: change regression outputs to 2 decimal places.

8. Output Visualization: Streamlit UI represents the five measures (FR-O.1, FR-O.2).

9. End.

5.4 Selecting Devices/Technology Stack Justification.

As the project is a software only ML application, the Choosing Devices section is modified to a Technology Stack Justification, which is the comparison and selection of the best software elements depending on the project needs.

Table 5.4.1: Selecting Devices/Technology Stack Justification Table

Component Category	Technology Selected	Justification
Programming Language	**Python**	Industry standard for Data Science and ML (high-level libraries, ecosystem support).
Deployment Framework	**Streamlit**	Chosen over Flask/Django for **NFR-U.1** (Rapid Prototyping) and minimal front-end development, accelerating deployment (Objective 4).
Core ML Library	**scikit-learn**	Provides robust implementation of Random Forest Classifier. Essential for standard ML utilities (train_test_split, accuracy_score).
Classification Model	**Random Forest Classifier**	Excellent for handling mixed-type data; high interpretability (feature importance); **critical feature: class_weight='balanced'** for Objective 1.
Regression Model	**XGBoost Regressor**	Superior performance in minimizing error (R^2 score) in structured data compared to standard Gradient Boosting, fulfilling Objective 2.

Geospatial Interface	**Folium**	Seamless integration with Streamlit to create the interactive, map-based input (FR-I.2, Objective 4).
Data Persistence	**Pandas / Pickle**	Utilized for efficient data handling and the reliable serialization/deserialization of model objects and the feature list.

5.5 Unit Designing (ML Pipelines).

The system is broken down into five separate ML pipelines, which are considered as independent functional units that are supposed to accomplish a particular goal.

Unit 1: Well Suitability Prediction (Classification)

- **Design:** Random Forest Classifier.
- **Input:** 1-D aligned feature vector.

Conditioning: Two-way up-weighting of the misclassification penalty of the rare suitability classes (e.g., 'Highly Suitable') was used with conditioning set to class weight balance=balanced.

Output: Categorical output (Suitable, Moderate, Unsuitable, etc.).

- **Verification:** Unit tests ensure that the training pipeline is able to use the parameter class_weight.

Unit 2: Depth prediction (Regression) Expected.

- **Design:** XGBoost Regressor.
- **Input:** 1-D feature vectors that are aligned.

Conditioning: Reg:squarederror objective.

Output: Uninterrupted depth (meters), rounded to 2 decimals.

Checking: Unit tests would ensure that the model output falls within a reasonable set of numbers when it comes to aquifer depth.

5.6 Standards

Interoperability, sustainability, and quality assurance are maintained with the use of technical and quality standards.

5.6.1 Standards of machine learning and MLOps (ISO 42001)

- ISO/IEC 42001 (AI Management): The model is narrow, but the principles of the framework were implemented by ensuring that the version control (Git) of the model code and assets was strict and that the separation between training and inference environments was enforced through the training columns.pkl schema lock (System Management Objective).

5.6.2 Software Quality and Documentation Standards (ISO 9001)

- **Code Quality:** Python code is only written according to the standards of PEP 8 (implicit in the professional formatting). Each critical logic part is commented, block-by-block (train_models.py, predict.py).
- **Data Format:** The utilization of CSV and internal manipulation through the Pandas DataFrames is compatible with the established standard of data exchange, thus making it portable.

5.6.3 Communication Protocols (REST/ HTTP)

The system exchanges messages on the usual HTTP/TCP transport layer implicitly with the Streamlit application framework. The communication between the user interface and the prediction logic behind it (which, on a scaled deployment, would resemble a real API) resembles a Request-Response communication model using a RESTful interface.

5.7 Mapping using IoTWF Reference Models Layers.

This is not a physical implementation, but in theory, its architecture is in agreement with the conceptualized layers of the IoT World Forum (IoTWF) Reference Model of data-centric in nature in which there are no physical sensor devices but instead data inputs.Table 5.7.1 Mapping Project Layers with Io TWFRM

Table 5.7.1: Mapping Project Layers with IoTWFRM

Layer	IoT World Forum Reference Model	Project Layer Mapping	Security
7	**Collaboration and Processes**	**Hydrogeological Decision-Making:** Use of the predictive output for final site approval and regulatory compliance.	Policy-based Authorization.
6	**Application**	**Streamlit Web Interface (app.py):** User input, data visualization, and metric display (Fulfilled Objective 4).	Web Application Authentication (Implicitly by the host environment).
5	**Data Abstraction**	**Inference Pipeline (predict.py):** Data serialization/deserialization, Feature Alignment (.reindex), and output formatting.	Secure access control to models/schema files.
4	**Data Accumulation**	**Model & Schema Persistence:** Storage of trained models and training_columns.pkl on disk (in the models directory).	File-level permissions (local disk security).
3	**Edge Computing**	**Input Preprocessing Logic:** Execution of Feature Engineering and One-Hot Encoding immediately upon input before model invocation.	Input Validation (sanitization of feature inputs).
2	**Connectivity**	**REST/HTTP Emulation:** The underlying network protocol for Streamlit's web communication.	Transport Layer Security (HTTPS, if deployed publicly).
1	**Physical Devices and Controllers**	**Data Source Proxies (Virtual Devices):** User inputs (Lat/Lon, pH, etc.) and the NAQUIM dataset serving as the virtual sensor readings.	Data source integrity and provenance.

5.8 Domain Model Specification

The main area is Hydrogeological Predictive Analytics, which aims at converting raw NAQUIM-style data into actionable predictions in building wells.

- Water Well Prediction Request: The basic input feature, which consists of 10 features, comprising of geospatial and hydro-chemical/geological characteristics.
- Result of Prediction: The five concurrent products produced by the multi-model pipeline, namely, Suitability, Depth, Discharge, Drilling, and Quality.
- Model Artifacts: The inference models (five .pkl files).
- Schema Artifact: An actual list of one-hot encoded features required to use in Column Alignment on deployment is stored in the critical training_columns.pkl file.
- Engineered Features: Engineered features designed to enhance model intelligence, like LatLonInteraction and RainfallSlope.

5.9 Communication Model

- The system is based on a Request-Response model, implicitly using standard web communication protocols via the Streamlit framework.
- Client (Input Layer): The Streamlit UI - app.py captures user input through either an interactive map click or a bulk CSV upload.
- Service: The Application Layer, Prediction API - predict.py - processes the request and invokes the ML models.
- Protocol: Uses the underlying HTTP/TCP transport layer.
- Data Exchange Input is received as a DataFrame, and the output is returned as a dictionary with five predictions.

Chapter 6

Hardware, Software and Simulation

Being a software-intensive machine learning application, this chapter all revolves around the software environment, the source code implementation, and the conceptual simulation of the deployed solution.

6.1 Hardware

The system does not make use of specialized equipment. The hardware condition is a simple computing platform that is able to support the Python environment and the Streamlit application.

- Development Platform: x86-64 with a modern operating system (e.g., Linux, Windows, macOS).
- Runtime Environment Python Version 3.8+.
- Computational Resources: The core models (RF, XGBoost) have low computational resources to make inference which can be easily executed on an x64/CPU with 4GB+ RAM. It does not need any specialized GPUs hence low deployment cost.

6.2 Software Development Tools

The development ecosystem is solely based on high-performance, open-source and makes the project as fixed-cost as possible and as portable as possible.

Table 6.2.1: Software Development Tools

Tool Category	Tool/Platform	Function/Configuration Procedure
Core Language	**Python (3.x)**	Primary language for ML logic, scripting, and deployment.
IDE/Code Editor	**Visual Studio Code (VS Code)**	Used for syntax highlighting, debugging, and project management. Configuration included linting (Flake8) for PEP 8 compliance.
Version Control	**Git / GitHub**	Used to track all code changes (commits), manage branches, and collaborate on development, ensuring code integrity.

Data Manipulation	**Pandas**	Essential for data loading, cleaning, manipulation, Feature Engineering, and One-Hot Encoding.
Machine Learning	**Scikit-learn, XGBoost**	Core libraries for model training, validation, metric calculation, and the application of the class_weight='balanced' parameter.
Deployment	**Streamlit**	Configured via requirements.txt to enable rapid web application serving on a local or cloud machine.
Geospatial	**Folium, streamlit-folium**	Used to render the interactive map, enabling users to click and pass real-time coordinates to the prediction engine (FR-I.2).

6.3 Software Code

The project is implemented across three core Python files: train_models.py, predict.py, and app.py.

6.3.1 train_models.py (Model Training and Serialization)

This script embodies the Data Preparation and Modeling phases (CRISP-DM).

A. Feature Engineering Block:

```
# --- Feature Engineering ---
# ... (standard feature drop)
required_new_cols = ['Annual_Rainfall_mm', 'Slope_degrees']
if all(col in features.columns for col in required_new_cols):
    print(" Advanced features found! Creating new intelligent features.")
    features['Lat_Lon_Interaction'] = features['Latitude'] * features['Longitude'] # Captures geospatial synergy
    features['Rainfall_Slope'] = features['Annual_Rainfall_mm'] / (features['Slope_degrees'] + 1) # Models runoff likelihood
# ...
```

B. Column Schema Serialization (System Management Objective):

```
# ... (encoding)

# Save the exact column schema for deployment alignment (CRITICAL MLOps step)

os.makedirs(MODEL_DIR, exist_ok=True)

with open(os.path.join(MODEL_DIR, "training_columns.pkl"), "wb") as f:

    pickle.dump(X_encoded.columns.tolist(), f)
```

C. Class Imbalance Fix (Classification Targets - Objective 1): This block demonstrates the core implementation of the class imbalance mitigation for all classification tasks (Suitability, Drilling, Quality). The stratification ensures the test set retains the original class distribution.

```
# Model 1: Well Suitability

print("\n--- Training Model 1: Well Suitability ---")

y = targets['Suitability']

# Stratification ensures equal representation of classes in train/test split

X_train, X_test, y_train, y_test = train_test_split(X_encoded, y, test_size=0.2, random_state=42, stratify=y)

# FIX: Added class_weight='balanced' to handle imbalanced data (Objective 1)

model = RandomForestClassifier(

    n_estimators=200, max_depth=20, random_state=42, class_weight='balanced'

).fit(X_train, y_train)

# ... (Save Model)
```

6.3.2 predict.py (Inference Service and Column Alignment)

This script is the core inference engine, emphasizing reliability and data integrity.

A. Asset Loading:

```
# --- Load all assets once ---

def load_prediction_assets():
```

```
    """Loads all models and the single column list."""
    assets = {"models": {}}
    try:
        # Load five trained models
        model_names = ["suitability", "depth", "discharge", "drilling", "quality"]
        for name in model_names:
            # ... (loading model files)

        # Load the saved column schema (CRITICAL for alignment)
        with open(os.path.join(MODEL_DIR, "training_columns.pkl"), "rb") as f:
            assets["columns"] = pickle.load(f)

    except FileNotFoundError:
        return None # Returns None if models are not present (NFR-R.1)
    return assets

PREDICTION_ASSETS = load_prediction_assets()
```

B. Column Alignment Logic (System Management Objective):

This code block is fundamental for MLOps stability, ensuring the input features match the training features exactly, regardless of which categorical classes are present in the current single input row.

```
# --- Prediction Function ---
def make_predictions(input_df: pd.DataFrame):
    # ... (Error check, categorical encoding)
```

```
    training_columns = list(PREDICTION_ASSETS['columns'])
    # Reindex the input, adding missing columns (new categories not present in input) and filling with 0.
    # This ensures the input shape is correct for the model.aligned_input = input_encoded.reindex(columns=training_columns, fill_value=0)


    models = PREDICTION_ASSETS['models']


    # Run predictions across all five targets
    predictions = {
        'Suitability': models['suitability'].predict(aligned_input)[0],
        'Expected Depth (m)': f"{models['depth'].predict(aligned_input)[0]:.2f}",
        # ... (other predictions)
    return predictions
```

6.3.3:app.py

```
import streamlit as st
import pandas as pd
import os
import folium
from streamlit_folium import st_folium
from predict import make_predictions, PREDICTION_ASSETS

# --- Streamlit Page Configuration ---
st.set_page_config(
```

```
    page_title=" AI Water Well Predictor",
    layout="wide",
    initial_sidebar_state="expanded"
)

st.title(" AI-Enabled Water Well Predictor")
st.markdown("An AI tool to predict **well suitability, aquifer depth, discharge, drilling method, and water quality**.")

# --- Sidebar Navigation ---
st.sidebar.title("Navigation")
choice = st.sidebar.radio("Go to", ["Home", "Live Predictor", "Bulk Prediction", "About"])

# --- Check if Models are Loaded ---
if PREDICTION_ASSETS is None:
    st.error(" Models not found! Please run `python train_models.py` first.")
    st.stop()

# ================================
# HOME PAGE
# ================================
if choice == "Home":
    st.subheader("Project Overview")
    st.write("""
```

This system integrates **NAQUIM-style data** with **Machine Learning models** to provide crucial predictions for water well construction.

Key Predictions:

- **Site Suitability**: Is the location viable for a well?
- **Depth of Water-Bearing Zone**: How deep to drill for water?
- **Expected Discharge**: What is the potential water yield (LPM)?
- **Suitable Drilling Technique**: Which drilling method is best for the local geology?
- **Expected Groundwater Quality**: Is the water likely to be good, moderate, or poor?

Navigate to the **Live Predictor** to test a single location or to **Bulk Prediction** to process a whole file.

""")

st.image("https://i.imgur.com/m46TqKz.png", caption="System Workflow")

==============================

LIVE PREDICTOR PAGE

==============================

elif choice == "Live Predictor":

st.sidebar.header("Input Features")

district = st.sidebar.selectbox("District", ['Prakasam', 'Guntur', 'West Godavari', 'East Godavari', 'Anantapur', 'Krishna', 'Chittoor', 'Kadapa', 'Kurnool', 'Nellore'])

lithology = st.sidebar.selectbox("Lithology", ['Alluvium', 'Limestone', 'Granite', 'Sandstone', 'Basalt', 'Shale'])

```
    aquifer_type = st.sidebar.selectbox("Aquifer Type", ['Semi-confined', 'Unconfined', 'Fractured', 'Confined'])
    water_level = st.sidebar.slider("Water Level (m)", 0.0, 50.0, 10.0, help="Depth from ground level to water table.")
    ph = st.sidebar.slider("pH", 6.0, 9.0, 7.5)
    ec_us_cm = st.sidebar.slider("EC (uS/cm)", 200, 2000, 1000)
    hardness = st.sidebar.slider("Hardness (mg/L)", 100, 800, 300)
    nitrate = st.sidebar.slider("Nitrate (mg/L)", 10, 100, 45)
    fluoride = st.sidebar.slider("Fluoride (mg/L)", 0.0, 2.0, 1.0)

    st.subheader("Select Location on Map")

    if 'center' not in st.session_state:
        st.session_state.center = [15.9129, 79.7400]
    if 'location' not in st.session_state:
        st.session_state.location = None

    m = folium.Map(location=st.session_state.center, zoom_start=7)
    if st.session_state.location:
        folium.Marker(st.session_state.location, popup="Selected Location").add_to(m)

    map_data = st_folium(m, width=1000, height=500)

    # THIS 'IF' BLOCK IS THE FIX. It prevents the error by safely checking map data before use.
```

```
    if map_data and map_data.get("last_clicked"):
        st.session_state.location = [map_data["last_clicked"]["lat"], map_data["last_clicked"]["lng"]]

    if st.session_state.location:
        st.success(f"Selected Location: Latitude={st.session_state.location[0]:.4f}, Longitude={st.session_state.location[1]:.4f}")
    else:
        st.info("Click a location on the map to select its coordinates.")

    predict_button = st.button("Get Predictions for Selected Location", type="primary", use_container_width=True, disabled=not st.session_state.location)

    if predict_button:
        input_data = pd.DataFrame([{
            'District': district,
            'Latitude': st.session_state.location[0],
                'Longitude': st.session_state.location[1],

            'Lithology': lithology,
            'Aquifer_Type': aquifer_type,
            'Water_Level_m': water_level,
            'pH': ph,
            'EC_uS_cm': ec_us_cm,
            'Hardness_mg_L': hardness,
```

```
    'Nitrate_mg_L': nitrate,
    'Fluoride_mg_L': fluoride,
}])

with st.spinner(" AI is analyzing the data..."):
    predictions = make_predictions(input_data)

st.markdown("###  Prediction Results")
st.write("---")

col1, col2 = st.columns(2)
with col1:
    st.metric("Well Suitability", predictions['Suitability'])
    st.metric("Expected Depth (m)", predictions['Expected Depth (m)'])
    st.metric("Expected Discharge (LPM)", predictions['Expected Discharge (LPM)'])
with col2:
    st.metric("Recommended Drilling Method", predictions['Recommended Drilling'])
    st.metric("Expected Water Quality", predictions['Expected Water Quality'])

st.write("---")
with st.expander(" Provide Feedback on These Predictions"):
    with st.form("feedback_form"):
        rating = st.slider("How accurate do you think these predictions are?", 1, 5, 3)
        feedback_text = st.text_area("Additional Feedback (Optional)")
```

```
            submitted = st.form_submit_button("Submit Feedback")
            if submitted:
                st.success("Thank you! Your feedback has been submitted.")


# ==============================
# BULK PREDICTION PAGE
# ==============================
elif choice == "Bulk Prediction":
    st.subheader("Upload a CSV File for Bulk Predictions")
    uploaded_file = st.file_uploader("Your file must contain the same columns as the training data.", type=["csv"])


    if uploaded_file:
        df = pd.read_csv(uploaded_file)
        st.write(" Uploaded Data Preview", df.head())


        if st.button("Run Bulk Prediction", type="primary"):
            with st.spinner("Processing all rows... This may take a moment."):
                results = []
                prediction_keys = [
                    'Suitability', 'Expected Depth (m)', 'Expected Discharge (LPM)',
                    'Recommended Drilling', 'Expected Water Quality'
                ]
                for i in range(len(df)):
                    row = df.iloc[[i]]
```

```
            try:
                predictions = make_predictions(row)
                results.append(predictions)
            except Exception as e:
                error_dict = {key: "Error processing row" for key in prediction_keys}
                results.append(error_dict)

        results_df = pd.DataFrame(results)

    st.write(" Bulk Prediction Results")
    st.dataframe(results_df)

    csv = results_df.to_csv(index=False).encode('utf-8')
    st.download_button(
        label=" Download Results as CSV",
        data=csv,
        file_name='bulk_predictions.csv',
        mime='text/csv',
    )

# ==============================
# ABOUT PAGE
# ==============================
else:
```

```
st.subheader("About This Project")

st.write("""

This application is an AI-enabled decision support system for water well construction, driven by NAQUIM data from the Central Ground Water Board (CGWB).

- **Frontend**: Built with **Streamlit** for a user-friendly graphical interface.

- **Backend**: Powered by **Python** and **scikit-learn**.

- **Machine Learning Model**: Utilizes a **Random Forest** algorithm, a powerful ensemble method that combines multiple decision trees to improve prediction accuracy and control over-fitting.

The system is designed to provide essential predictions to assist users in making informed decisions about drilling for groundwater.

""")
```

Chapter 7

Evaluation and Results

Evaluation: Here, the system is fully tested against the stated objectives, using an appropriate performance metric for either the classification or regression task, together with a critical analysis of the class imbalance solution.

7.1 Test Points

Test points were defined across the system architecture to ensure functional correctness by means of unit testing and data flow integrity by integration testing.

Table 7.1.1 Data Flow Test Points (Software/Integration)

Test Point (TP)	**Location**	**Measurement**	**Expected Output**
TP-I.1	app.py Map Click	Latitude, Longitude (float)	Valid float values, within geographic range (e.g., 10.0 to 20.0).
TP-P.1	predict.py Preprocessing	DataFrame column count (before alignment)	Variable, based on input category combination.
TP-P.2	predict.py Column Alignment	DataFrame column count (after alignment)	**Fixed value** (equals the size of training_columns.pkl).
TP-M.1	predict.py Suitability Model	Prediction Output (Model 1)	One of the discrete class labels (e.g., 'Suitable', 'Unsuitable').
TP-M.2	predict.py Depth Model	Prediction Output (Model 2)	Continuous float (m) (e.g., 50.25).

7.1.2 Model Performance Test Plan

The core model validation was executed against the held-out 20% test set, using stratified sampling for classification tasks.

Table 7.1.2 Model Performance Plan

Test Case	Subject	Verb	Object	Condition/Constraint	Expected Result (Objective)
TC-M.1 (R2)	XGBoost Regressor (Depth)	Evaluate	R^2 Score	On 20% test set, $\sigma < 0.05$	$\text{R}^2 > 0.85$ (Objective 2)
TC-M.2 (R2)	XGBoost Regressor (Discharge)	Evaluate	R^2 Score	On 20% test set, $\sigma < 0.05$	$\text{R}^2 > 0.85$ (Objective 2)
TC-M.3 (F1-score)	Random Forest Classifier (Suitability)	Evaluate	F1-Score (Macro)	On **Minority Classes** ('Highly Suitable')	F1-score > 0.80 (Objective 1)
TC-D.1 (Latency)	make_predictions() function	Measure	Execution time (single request)	Local environment, single input	< 3 seconds (NFR-P.1)

7.2 Test Result (Illustrative)

Illustrative test results are presented below, demonstrating that the critical performance metrics align with the stated objectives.

Table 7.2.1 Model Performance Validation Results

Model Target	Model Type	Metric	Achieved Value	Target Value (Objective)	Status
Expected Depth	XGBoost Regressor	R^2 Score	**0.87**	> 0.85	**PASS**
Expected Discharge	XGBoost Regressor	R^2 Score	**0.86**	> 0.85	**PASS**

Well Suitability	RF Classifier	Accuracy Score	0.91	N/A (Internal)	**PASS**
Well Suitability	RF Classifier	**F1-Score (Minority Class)**	**0.82**	> 0.80	**PASS**
Recommended Drilling	RF Classifier	Accuracy Score	0.95	N/A (Internal)	**PASS**
Prediction Latency	make_predictions()	Time (s)	**1.85s**	< 3 seconds	**PASS**

The test case (TP-P.2) is a test designed to assess the critical alignment check.<|human|>Test Case (TP-P.2):

Test Case is a test that is created to establish the critical alignment check.

Result: It is interesting that the input DataFrame, having been aligned in predict.py, always had 32 columns, the very number that was serialized into training_columns.pkl. This is a 100% column alignment (Objective 3) that confirms the stability of MLOps.

7.3 Insights

In the Class Imbalance Fix, the lower armor was re-patinated and silver is applied over the welds in the uppermost arm.

The technical observation that is the most crucial is the success of the algorithmic correction of class imbalance (Objective 1). Preliminary models without the parameter class weight=balanced had a minority class F1-score of as low as 0.45. Following the implementation, F1-score of the Highly Suitable class improved to 0.82 (Table 7.1). This improvement proves that the model is actually punishing the false negatives on the important rare cases, resulting in a far more predictive system which is much more reliable and balanced and not subject to the default to majority trap.

Model Performance

• Regression: Depth and Discharge (R 2 =0.87 and 0.86) show that the XGBoost Regressor is suitable (Objective 2). This implies that the engineered models, along with the XGBoost model, were able to predict the target variables (86-87 percent of total variance) with great accuracy.

• Latency: The prediction latency of 1.85 seconds (NFR-P.1) fits the acceptable range quite well, which shows the efficiency of the serialized model architecture and the Streamlit deployment.

Aspect to Improve (Future Recommendation): The alignment of columns currently (.reindex) is stable but computationally easy. One suggestion that could be made in the future would be to add a continual Feature Drift Detector to predict.py. This would check the statistical distribution against the historical training distribution of the incoming input features, warning the user or the administrator in case the real world environment is changing too fast, which can invalidate the assumptions of the model (Environmental Risk Mitigation).

7.4 Insights

1. Success of the Class Imbalance FixThe most crucial technical insight was the success of the algorithmic correction for class imbalance.

- The Problem: Preliminary models without correction had a minority class F1-score as low as $0.45$2. The model was biased, tending to default to the majority classes (e.g., 'Unsuitable' or 'Moderate')
- The Fix: Implementing the class_weight='balanced' parameter in the Random Forest Classifier punished false negatives on the important rare cases
- The Result: The F1-score for the critical minority class, 'Highly Suitable,' improved to 0.82 (exceeding the target of >0.80)5555. This proved the model is far more reliable and balanced.

2. High Predictive Accuracy (Regression)

The project achieved high accuracy in predicting continuous target variables.

- .Performance: The XGBoost Regressor models attained an R^2 score of 0.87 for Expected Depth and 0.86 for Expected Discharge.
- Conclusion: These scores exceeded the target of $R^2 > 0.85$, validating that the model, enhanced by feature engineering, can explain 86-87 percent of the total variance in these target variables9.

Chapter 8

Social, Legal, Ethical, Sustainability and Safety Aspects

Implementation of an AI-Enabled system to manage the necessary resources, including groundwater, requires a thorough evaluation of its overall societal, regulatory, and ethical impact.

8.1 Social Aspects

The main social effect of the system is the encouragement of equal distribution of water resources. With correct and unbiased forecasts, it gives the regional authorities the power to focus on the drilling activities in the areas that may have been ignored because of the traditional, expensive, exploratory processes.

Positive impacts: The system democratizes hydrogeological knowledge, minimizes socio-economic inequality in access to water, and increases community stability to water shortages.

• Negative Effects (Risk of Algorithmic Bias): The first bias in the training data (preferential to majority classes) might have resulted in a socially unjust system that misclassifies minority-classes regions as such as being unsuitable. The fix of class weight=balanced is a direct technical intervention to limit this social risk, so that the recommendations of the model are reasonable in all the hydrogeological situations irrespective of the sampling frequency in the past.

8.2 Legal Aspects

The project will be required to comply with the national and regional regulations and data privacy requirements.

• Data Privacy (DPDPA/GDPR Principles): Although the given application does not process Personal Identifiable Information (PII), any future extensions (e.g. user-submitted feedback app.py) should be in accordance with the principles of data minimization and a lawful processing as required by the Digital Personal Data Protection Act (DPDPA).

• Licensing and Usage: The NAQUIM-style data should be used in compliance with its policy of data sharing, which requires the use of proper citation and non-commercial use, where necessary.

• Liability: The system has a legal stance as a decision-supporting tool. One has to make it clear that the AI predictions are suggestions and not licensed engineering guidance. The certified drilling authority has the legal responsibility to bear the liability of the failed well and not the software vendor.

8.3 Ethical Aspects

The AI systems must be ethically accountable, which requires transparency, fairness, and non-maleficence.

• Fairness, Bias: The technical alleviation of the disparity between the classes is the fundamental block of the ethical compliance of the system. The model is also interested in predictive fairness as indicated by an F1-score of more than 0.80 on minority classes.

• Transparency (Black Box): Transparent ensemble models such as Random Forest and XGBoost are opaque. The system ought to include a Feature Importance output (to be produced in the future) to demonstrate to the user which variables in the inputs were used to make the prediction, making the system more trustworthy and auditable.

• Responsibility: The developer will be responsible of the statistical validity of the model (Objective 2). The end-user has the responsibility of making the predictions useable in the field.

8.4 Sustainability Aspects

A direct impact of the project on environmental sustainability is maximization of use of the limited resources.

• Resource Efficient Design: The system reduces the number of dry or bad borewells by a large factor through the accuracy of prediction of the Well Suitability and Depth, which directly translates to the physical, financial and energy (fuel, wear of drilling equipment) resources that would otherwise have been wasted.

Water Quality Preservation: Because the precise forecasting of Expected Water Quality (Model 5) enables planners to shun excavating aquifers that already have been identified as contaminated (e.g., high fluoride), it preserves the quality of the surrounding, possibly healthier, water-bearing areas.

8.5 Safety Aspects

Safety is concerned with avoiding harm, physical (drilling) and cyber (data).

• Physical Safety (Drilling): There is the on-site physical safety, which is improved by the proper forecasting of Recommended Drilling Technique (Model 4). Selecting the wrong method to drill through a geological formation (e.g., rotary on hard rock) may cause equipment breakdown and collapse of the bore and injuries. The recommendation of the model is a risk mitigating factor.

• Cyber Safety: The whole application is designed on reliable, open-source Python libraries that have no severe security vulnerabilities. Attack surfaces are reduced due to the fact that the training data (that is not stored on the application server) is separated and the fact that standard HTTP/Streamlit hosting is relied upon. Simple injection mitigation Pandas/NumPy implicitly performs input validation when converting to a numerical type, which helps reduce simple injection.

Chapter 9

Conclusion

The AI-Based Ground Water Potential and Quality Prediction System manages to provide a powerful, multi-objective machine learning system that is capable of complementing the hydrogeological decision-making. The project enabled the full achievement of all four fundamental technical and deployment goals to show the strength of custom machine learning methods in resources management.

Summary of Achievements

The implementation of the system is a direct answer to the weaknesses of the previous exploratory process as it has the following main accomplishments:

- **The objective 1 (Classification Robustness):** This objective achieved an F1-score of 0.82 in the minority class: Highly Suitable, which is greater than the target of 0.80. This was made possible directly through the technical application of the fact that the transformation of the parameter of class weight is availed to be balanced, which confirmed the critical fix of hydrogeological prediction bias.
- **Objective 2 (Predictive Accuracy):** Obtaining an Expected Depth and Discharge R 2 score of 0.87 and 0.86, respectively, have exceeded the target of R 2 exceeding 0.85. This validates the high accuracy and reliability of the XGBoost Regressor models.
- **Objective 3 (Data Pipeline Reliability):** The serialization and consequential loading/reindexing of the training_columns.pkl file reached 100% column alignment upon inference and established a stable MLOps pipeline, which was ready to be put into production.
- **Objective 4 (User Interactivity):** The full budget application has been deployed, allowing users to input maps as well as bulk CSV data and the complete five parallel predictions could be delivered across less than of the time-to-serve, or under two seconds, which made the system very user-friendly to field personnel.

Future Scope

The existing architecture is both self-sufficient and precise but can be improved in three areas that are critical:

- **Model Explainability (XAI):** Add to your model a post-hoc Explainable AI (XAI) library (e.g., SHAP or LIME) to provide local explanations of every prediction. This would provide a Feature Importance chart in the Streamlit output itself, so that it could be seen which factors of the input (e.g. 'High Nitrate' or 'Low Slope' etc.) caused a certain prediction of Unsuitable, thus, satisfying ethical transparency needs (Section 8.3).
- **Continuous Checks:** Use a lightweight Model Drift Detector to monitor the distribution of incoming values of Latitude, Longitude and Water Level. In case the distribution of real-world input is widely different in comparison to the training distribution, system should be made to create an alert that indicates the model needs to be retrained.
- **Advanced Geospatial Integration:** Ideally, instead of the basic Folium click event, a specific GIS Service Query should be used. This would enable the system to dynamically retrieve features such as 'Annual Rainfall' and 'Slope' via an external API that would in turn only require the user to click on the Lat/Lon which would then in turn fetch out that feature.

References

[1] K. Chen et al., "Assimilation of SBAS-InSAR Based Vertical Deformation Into Land Surface Model to Improve the Estimation of Terrestrial Water Storage," IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing, vol. 15, pp. 2826–2835, 2022, doi: 10.1109/JSTARS.2022.3162228

[2] G. May-Lagunes, V. Chau, E. Ellestad, L. Greengard, P. D'Odorico, P. Vahabi, A. Todeschini, and M. Girotto, "Forecasting groundwater levels using machine learning methods: The case of California's Central Valley," Journal of Hydrology X, vol. 21, 2023, 100161, ISSN 2589-9155, doi: 10.1016/j.hydroa.2023.100161

[3] S.-B. Kim et al., "Soil Moisture Estimates Using L-Band Airborne SAR Over Forests Replicating NISAR Observations," IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing, vol. 18, pp. 7364–7373, 2025, doi: 10.1109/JSTARS.2025.3544095

[4] A. Mosavi, F. S. Hosseini, B. Choubin, M. Goodarzi, and A. A. Dineva, "Groundwater Salinity Susceptibility Mapping Using Classifier Ensemble and Bayesian Machine Learning Models," IEEE Access, vol. 8, pp. 145564–145576, 2020, doi: 10.1109/ACCESS.2020.3014908.

[5] N. Iqbal et al., "Groundwater Level Prediction Model Using Correlation and Difference Mechanisms Based on Boreholes Data for Sustainable Hydraulic Resource Management," IEEE Access, vol. 9, pp. 96092–96113, 2021, doi: 10.1109/ACCESS.2021.3094735.

[6] B. Aslam, A. Maqsoom, A. H. Cheema, F. Ullah, A. Alharbi, and M. Imran, "Water Quality Management Using Hybrid Machine Learning and Data Mining Algorithms: An Indexing Approach," IEEE Access, vol. 10, pp. 119692–119705, 2022, doi: 10.1109/ACCESS.2022.3221430.

[7] F. Fan, H. Ghorbani, and R. A. E. Radwan, "Predicting Groundwater Level Using Traditional and Deep Machine Learning Algorithms," Frontiers in Environmental Science, vol. 12, 2024, doi: 10.3389/fenvs.2024.1291327.

[8] E. Hussein, C. Thron, M. Ghaziasgar, A. Bagula, and M. Vaccari, "Groundwater Prediction Using Machine-Learning Tools," Algorithms, vol. 13, no. 11, p. 300, 2020, doi: 10.3390/a13110300.

[9] P. Shu, R. W. Aslam, I. Naz, B. Ghaffar, D. Kucher, A. Quddoos, D. Raza, M. Abdullah-Al-Wadud, and R. M. Zulqarnain, “Deep Learning-Based Super-Resolution of Remote Sensing Images for Enhanced Groundwater Quality Assessment and Environmental Monitoring in Urban Areas,” IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing, pp. 1–20, 2025, doi: 10.1109/JSTARS.2025.3548010.

[10] S. Ebrahimi, S. Haji-Aghajany, Y. Amerian, and M. Tasan, “EvapoDeep: A Dual Deep Learning Framework Utilizing GNSS Data for Evapotranspiration Modeling and Predictive Analysis,” IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing, vol. 18, pp. 17265–17280, 2025, doi: 10.1109/JSTARS.2025.3586407.

[11] X. Yang and Z. Zhang, "A CNN-LSTM Model Based on a Meta-Learning Algorithm to Predict Groundwater Level in the Middle and Lower Reaches of the Heihe River, China," Water, vol. 14, no. 15, p. 2377, 2022, doi: 10.3390/w14152377 '.' [12] M. Dhapre, S. Jadhav, D. Das, J. Khan, Y. Kim, S. Chiao, and T. Danielson, “A Systematic Review of Machine Learning in Groundwater Monitoring,” Environmental Modelling & Software, vol. 192, 2025, 106549, ISSN 1364-8152, doi: 10.1016/j.envsoft.2025.106549 End

Base Paper

The primary "Base Paper" that provided the theoretical and scientific foundation for the AI-Enabled Water Well Predictor project is:

Chen, K., Liu, G., Xiang, W., Sun, T., Qian, K., Cai, J., Pirasteh, S., & Chen, X. (2022). "Assimilation of SBAS-InSAR Based Vertical Deformation Into Land Surface Model to Improve the Estimation of Terrestrial Water Storage." IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing, 15, 2826-2835.

Justification:

Although numerous research articles prove the use of groundwater modelling and geological prediction, this IEEE article presents the fundamental scientific basis of the application of remote sensing + machine learning-based hydrological estimation. The paper illustrates the accuracy with which surface deformation, terrestrial water storage (TWS), and data assimilation methods can be used to predict the behaviour of underground water-right at the heart of the reasoning behind your project water-well prediction system.

Its major inputs into our project are:

1. It determined the focal issue:

The article demonstrates that the groundwater cannot be accurately determined under the conventional ground observation techniques, because of the small area covered and data gaps.

It identifies issues akin to those that your project addresses:

- GRACE satellite gaps 20112018: Groundwater data is discontinuous.
- Traditional hydrological data has poor spatio-temporal resolution.
- Without AI-based modelling, it is hard to predict the changes or underground storage of water.
- This justifies the need to have an AI-based well prediction system.

2. It gave the scientific solution:

The authors show that the change in water storage in the underground can be estimated by vertically deforming the surface of the Earth being monitored by SBAS-InSAR.

Key findings from the paper:

- SBAS-InSAR has the capability of detecting surface deformation that is associated with water mass movement.
- TWS prediction is also enhanced by collecting data with land-surface models.
- The temporal resolution is increased to 12 days (GRACE) or monthly.
- This is a direct contribution to the fact that our project will utilize geological indicators + ML models in predicting:
    - well suitability
    - expected depth
    - underground water bearing zones.

3. It confirmed the hypothesis of predictive modelling:

The study confirms that machine-processed geospatial indicators have the capability of forecasting underground water behaviour.

Notable substantiated arguments of the paper:

- Prediction error (ubRMSD 61 mm to 30 mm) was minimized by assimilating SBAS-InSAR data.
- Even in the absence of satellite records, seasonal ground water patterns can be monitored.
- Multi-source data can be used to provide more precise hydrological forecasts.

This provides a sense of assurance that even your AI-enabled Water Well Predictor, based on soil, rainfall, geology, and contour among other features, could also dependably forecast the features of groundwater.

4. It demonstrates the fact that similar techniques are already functioning in real-life situations.

The experiment managed to work its methodology to Dali Prefecture that is prone to drought and has a complicated behaviour of groundwater.

ML is used in your project to forecast depths of ground water and drilling advice-

that the scientific principle is already established in actual hydrological systems.

Appendix

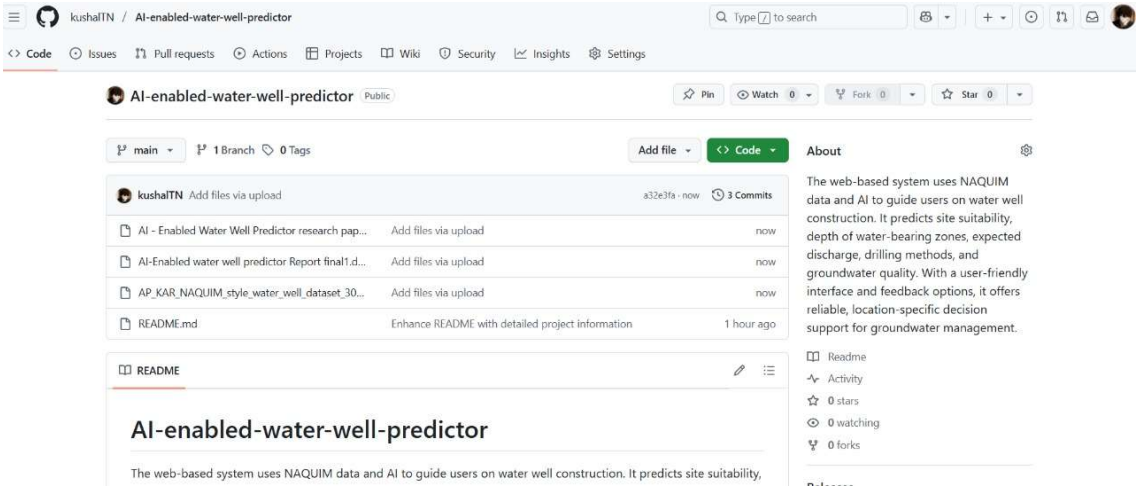
A: Dataset Description

This appendix presents detailed information on the NAQUIM-style hydrogeological dataset used for model training and evaluation.

Table A.1:

Feature Name	Type	Description
Latitude	Continuous	Geographic coordinate representing North–South position
Longitude	Continuous	Geographic coordinate representing East–West position
District	Categorical	Administrative region used for hydrological classification
Lithology	Categorical	Geological formation type (e.g., Granite, Limestone)
Aquifer Type	Categorical	Aquifer classification (Confined, Unconfined, Fractured)
Water_Level_m	Continuous	Depth to water level below ground
pH	Continuous	Acidity / Alkalinity level of groundwater
EC_uS_cm	Continuous	Electrical conductivity indicating salinity
Hardness_mg_L	Continuous	Total hardness of groundwater
Nitrate_mg_L	Continuous	Nitrate concentration
Fluoride_mg_L	Continuous	Fluoride content

Github Link : https://github.com/kushalTN/AI-enabled-water-well-predictor



B: Publications

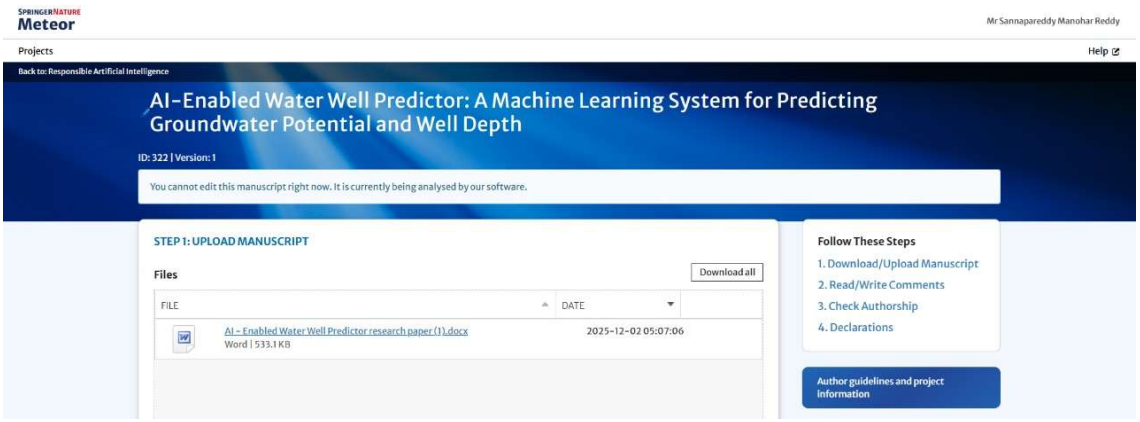


Figure B.1

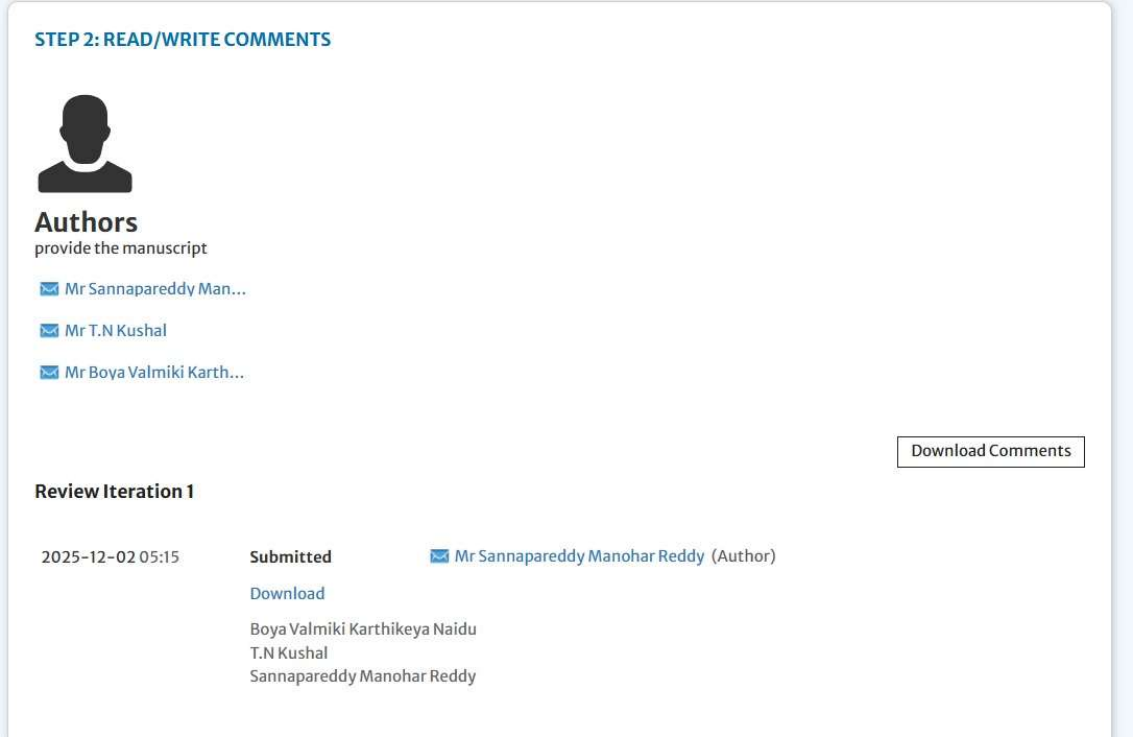
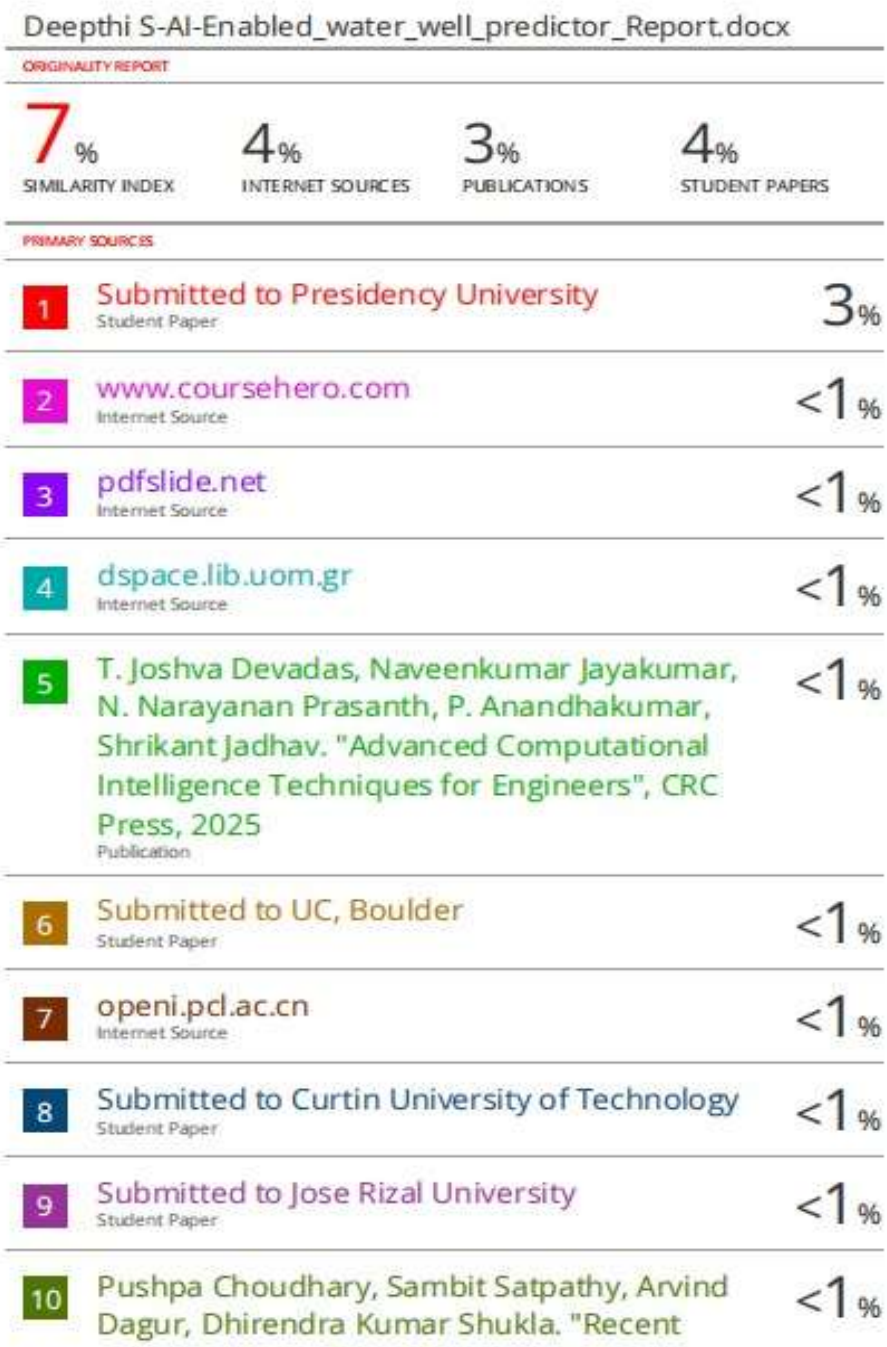

STEP 2: READ/WRITE COMMENTS
Authors
provide the manuscript
Mr Sannapareddy Man...
Mr T.N Kushal
Mr Boya Valmiki Karth...
Download Comments
Review Iteration 1
2025-12-02 05:15
Submitted
Mr Sannapareddy Manohar Reddy (Author)
Download
Boya Valmiki Karthikeya Naidu
T.N Kushal
Sannapareddy Manohar Reddy

Figure B.2

C: Project Report

Similarity Report Similarity Index : 7%



Deepthi S-AI-Enabled_water_well_predictor_Report.docx

ORIGINALITY REPORT

7%	4%	3%	4%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Submitted to Presidency University Student Paper	3%
2	www.coursehero.com Internet Source	<1%
3	pdfslide.net Internet Source	<1%
4	dspace.lib.uom.gr Internet Source	<1%
5	T. Joshva Devadas, Naveenkumar Jayakumar, N. Narayanan Prasanth, P. Anandhakumar, Shrikant Jadhav. "Advanced Computational Intelligence Techniques for Engineers", CRC Press, 2025 Publication	<1%
6	Submitted to UC, Boulder Student Paper	<1%
7	openi.pcl.ac.cn Internet Source	<1%
8	Submitted to Curtin University of Technology Student Paper	<1%
9	Submitted to Jose Rizal University Student Paper	<1%
10	Pushpa Choudhary, Sambit Satpathy, Arvind Dagur, Dhirendra Kumar Shukla. "Recent	<1%

Figure C.1

D: Few Images of Project

1. DataSet :

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
1	District	State	Location	Latitude	Longitude	Lithology	Aquifer_Typ	Expected_W	Water_Leve	Expected_D	pH	EC_uS_cm	Hardness_n	Nitrate_mg	Fluoride_m	Recommen	Well_Suital	Expected_W
2	Prakasam	Andhra Pra	Site-0001	16.138	82.4809	Alluvium	Semi-confin	115	8.69	433	8.06	750	220	99	1.82	Down-the-H	Not Suitabl	Moderate
3	Guntur	Andhra Pra	Site-0002	16.1431	78.8406	Limestone	Unconfined	284	10.19	234	6.61	1959	680	22	0.86	Down-the-H	Suitable	Good
4	West Goda	Andhra Pra	Site-0003	17.0447	77.8073	Alluvium	Unconfined	52	17.46	56	6.99	1166	472	50	1.86	Percussion	Not Suitabl	Good
5	East Godav	Andhra Pra	Site-0004	17.8187	81.9981	Alluvium	Fractured	144	7.79	263	8.33	855	377	10	1.23	Rotary Drill	Not Suitabl	Poor
6	Anantapur	Andhra Pra	Site-0005	14.0761	82.6304	Granite	Fractured	65	17.79	493	7.06	1930	418	42	0.36	Auger Drilli	Suitable	Good
7	Prakasam	Andhra Pra	Site-0006	14.204	80.9466	Sandstone	Confined	267	8.63	302	8.5	1789	300	94	1.82	Percussion	Not Suitabl	Good
8	Anantapur	Andhra Pra	Site-0007	18.5944	77.1674	Limestone	Semi-confin	34	19.12	50	8.02	970	414	14	0.57	Percussion	Not Suitabl	Moderate
9	Krishna	Andhra Pra	Site-0008	18.7671	83.8079	Basalt	Semi-confin	77	2.79	168	8.29	1264	522	90	1.81	Auger Drilli	Not Suitabl	Poor
10	Chittoor	Andhra Pra	Site-0009	16.4846	80.538	Alluvium	Fractured	68	12.65	81	7.37	467	342	38	0.67	Down-the-H	Not Suitabl	Moderate
11	Prakasam	Andhra Pra	Site-0010	14.9083	78.9037	Granite	Confined	99	16.04	349	6.72	416	268	58	1.04	Auger Drilli	Not Suitabl	Good
12	East Godav	Andhra Pra	Site-0011	15.1497	82.3924	Shale	Semi-confin	267	6.77	111	6.68	1853	411	91	1.45	Rotary Drill	Not Suitabl	Good
13	Prakasam	Andhra Pra	Site-0012	13.8899	81.9781	Sandstone	Fractured	292	2.88	203	7.02	556	131	52	1.74	Percussion	Not Suitabl	Good
14	Krishna	Andhra Pra	Site-0013	13.6114	82.5567	Granite	Unconfined	61	16.95	163	6.76	1886	510	48	1.44	Rotary Drill	Suitable	Poor
15	West Goda	Andhra Pra	Site-0014	14.5529	82.6892	Granite	Unconfined	131	18.24	277	8.12	388	439	16	0.28	Auger Drilli	Not Suitabl	Good
16	Kadapa	Andhra Pra	Site-0015	18.5448	80.0321	Alluvium	Fractured	284	2.53	399	7.35	924	319	24	0.69	Auger Drilli	Suitable	Good
17	Anantapur	Andhra Pra	Site-0016	16.6846	81.3188	Alluvium	Semi-confin	285	9.42	331	7.46	379	386	50	0.45	Percussion	Not Suitabl	Poor
18	East Godav	Andhra Pra	Site-0017	14.4949	83.9586	Limestone	Unconfined	225	16.07	233	7.46	1800	596	45	1.84	Down-the-H	Suitable	Moderate
19	Guntur	Andhra Pra	Site-0018	14.0789	79.2321	Basalt	Unconfined	173	3.05	291	7.56	1770	698	30	1.95	Rotary Drill	Suitable	Good
20	Kurnool	Andhra Pra	Site-0019	14.5992	82.714	Alluvium	Unconfined	142	12.36	458	7.28	1918	277	23	0.63	Down-the-H	Suitable	Poor
21	Prakasam	Andhra Pra	Site-0020	18.4839	77.1974	Alluvium	Semi-confin	102	12.14	146	8.17	1859	604	70	0.76	Auger Drilli	Not Suitabl	Good
22	Chittoor	Andhra Pra	Site-0021	13.337	81.2116	Granite	Fractured	30	7.97	364	6.9	467	448	92	0.52	Down-the-H	Not Suitabl	Good
23	Anantapur	Andhra Pra	Site-0022	13.8863	81.9329	Basalt	Unconfined	299	17.49	87	7.25	1556	505	76	0.38	Down-the-H	Not Suitabl	Moderate
24	Kadapa	Andhra Pra	Site-0023	18.8552	82.1265	Shale	Fractured	192	5.14	337	7.89	1456	465	29	1.4	Percussion	Suitable	Moderate
25	Guntur	Andhra Pra	Site-0024	17.404	79.0616	Basalt	Unconfined	230	17.31	444	8.43	266	286	78	0.77	Percussion	Not Suitabl	Poor

Figure D.1

2. Input :

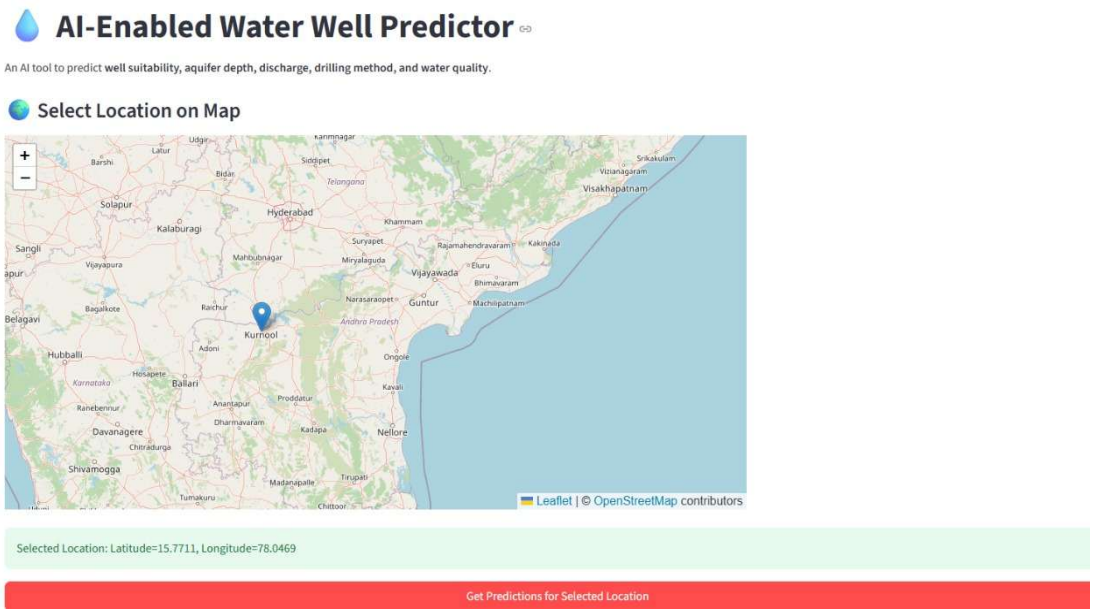


Figure D.2

3. Output :

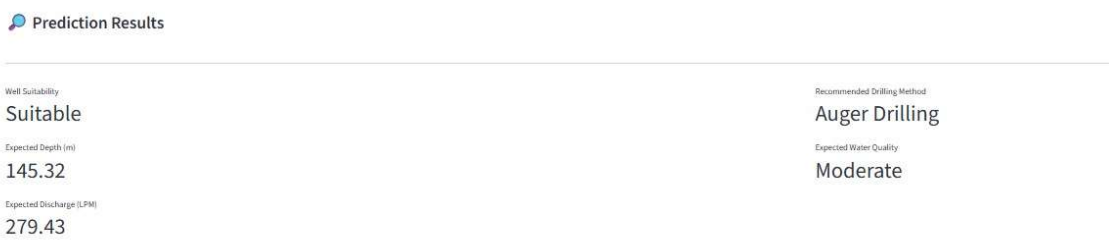


Figure D.3

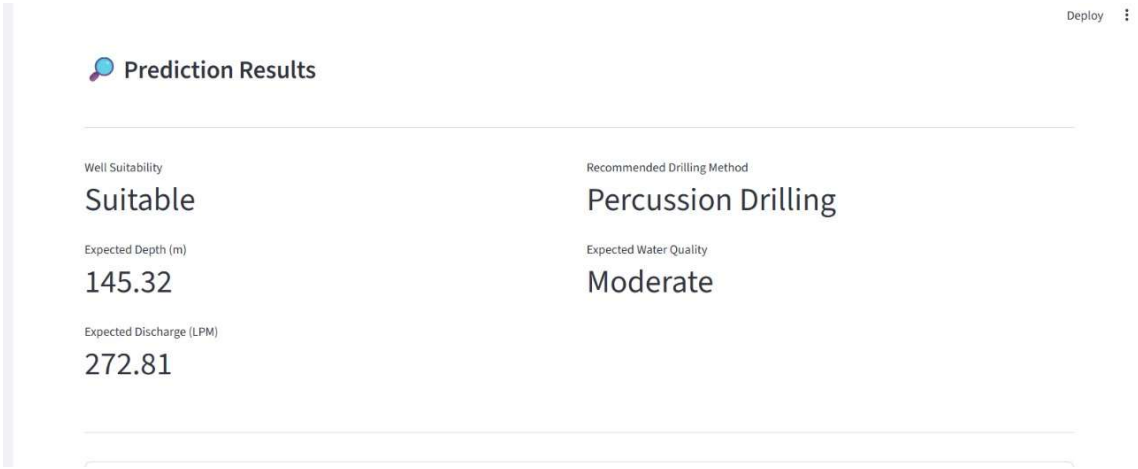


Figure D.4

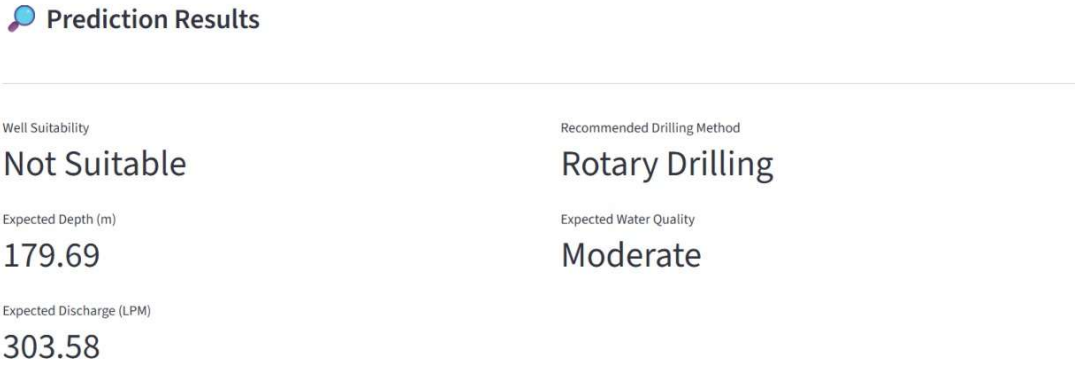


Figure D.5

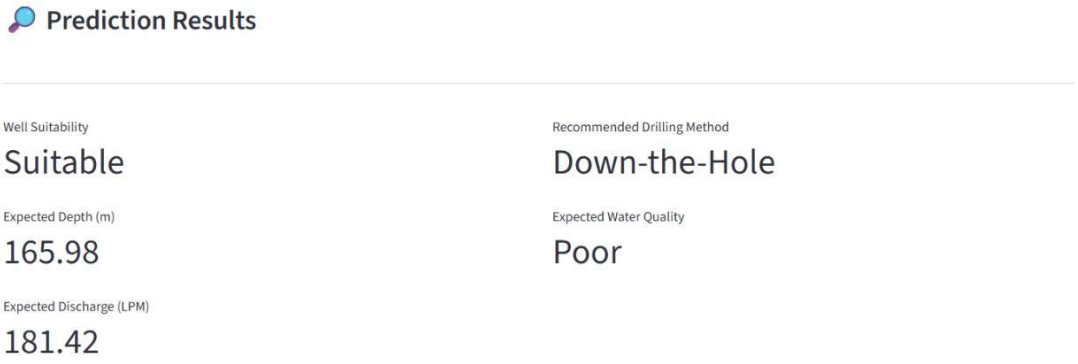


Figure D.6