



Intelligent Web App for Flash Flood Prediction in Nigeria's Coastal Regions

Ijegwa David ACHEME¹, Mohammed ABDULJALAL²

¹Department of Computer Science, College of Science and Computing Wigwe University, Isiokpo
david.acheme@wigweuniversity.edu.ng

²Department of Computer Science, Edo State University, Uzairue
abduljalal20.mohammed@edouniversity.edu.ng

Corresponding Author: david.acheme@wigweuniversity.edu.ng, +234-8062889197

Received: 08/07/2025

Revised: 25/08/2025

Accepted: 22/09/2025

Available online: 19/10/2025

Abstract: Many coastal cities in Nigeria and around the world are faced with the menace of flash floods and many times, it temporarily disrupts the socio-economic activities of residents. This project aims to address this challenge by developing a smart web application using machine learning to intelligently predict flash flood occurrence and offer recommendations. In order to achieve this, the random forest machine learning algorithm is utilized to analyze environmental data such as rainfall, river levels, and soil moisture necessary for the prediction of a flood which are captured in real-time using the OpenMeteo API. The machine learning model is then trained using these environmental variables and integrated into a web application for easy user interaction. The frontend of the web application is built with TypeScript, React.js, and Tree.js, providing an interactive and user-friendly interface for visualizing flood predictions, while the backend is built using MongoDB and python (FLASK framework). The goal is to offer accurate, real-time flood forecasts to help individuals prepare and respond effectively. This project demonstrates the integration of data science and web development to create a practical tool for disaster risk management. The random forest model was evaluated using the standard metrics for evaluating machine learning models and showed the following results; Accuracy of 96%, precision of 75% and recall of 91%. In addition, the model, showed a Real-time latency of less than one second, which is indicative of a fast response to changing environmental data input. Since flood conditions can change rapidly, this low real-time latency shows that the web is able to respond quickly to new sensor or satellite data input.

Keywords: Disaster Management, Real-time Forecasting, Flood Prediction, Machine Learning, Nigeria Coastal regions, Intelligent Web App

1. INTRODUCTION

Floods rank among the most devastating natural phenomena, doing significant harm to lives, property, and the ecosystem. The increasing frequency and intensity of flood events especially in Nigeria's coastal cities like Lagos, parts of Bayelsa, Rivers, Cross River and Delta States in Nigeria are partly attributed to climate change, requiring an urgent need for effective flood prediction systems so as to reduce the negative effects on the social and economic wellbeing of the people [1]. In areas where abrupt, violent floods are common, it is critical to develop a reliable flash flood prediction system to protect people, property, and the environment. Because of their sudden start and catastrophic effects, flash floods remain a serious hazard to communities all over the world.

Researchers and meteorologists have indicated that technical instruments, like remote sensing, radar systems, and weather satellites, provide essential data for enhancing the accuracy of flood predictions [2]. Hydrological models, like the Hydrologic Engineering Center's Hydrologic Modeling System (HEC-HMS) and the Soil and Water Assessment Tool (SWAT), simulate the movement of water through a watershed [3]. These models assess runoff and possible flooding by taking into account rainfall data, land cover, and soil characteristics. Moreover, rainfall-runoff models that quantify the link between precipitation and runoff include the SCS Curve Number technique and the Rational technique [4].

Artificial intelligence (AI) and machine learning approaches have also become more popular in recent years for flash flood prediction. Using historical data, these techniques, such as decision trees and neural networks identify trends and forecast the probability of a flood [5]. These Machine learning and AI systems have continued to evolve from empirical approaches to sophisticated, data-driven systems that are able to integrate meteorological, hydrological, and geographical data, thereby enhancing the accuracy and reliability of the prediction models, which ultimately aids in mitigating the impact of flash floods [6]. While much advancement has been reported, a major problem still remains the inaccessibility of these prediction models to ordinary citizens for usage. Also, the existing flash flood prediction models often struggle to provide high-resolution forecasts both spatially and temporally. This limitation stems from the inherent complexity of meteorological and hydrological processes, making it challenging to capture rapid changes in weather patterns and their localized impacts. The integration of diverse data sources, including meteorological, hydrological, and geographical

information, remains a critical challenge. Achieving seamless interoperability among these datasets is crucial for developing holistic and accurate prediction models. The lack of standardized protocols for data sharing and collaboration across agencies further exacerbates this issue. Real-time assimilation of data poses a significant hurdle in flash flood prediction systems. The ability to incorporate up-to-the-minute information from various sources, such as remote sensing and weather satellites, is crucial for enhancing the timeliness and reliability of predictions. The computational demands and potential delays in data acquisition contribute to this challenge. Although AI and machine learning approaches have the potential to increase flood prediction accuracy, their general integration into flash flood prediction systems is still not very popular. To fully utilize AI in improving prediction capabilities, obstacles pertaining to model interpretability, processing demands and the requirement for substantial dataset training must be removed.

The research work addresses these challenges by deploying REST APIs to capture real-time weather variables which are fed into the machine learning model for a more reliable prediction, while at the same time presenting the results to users with an easy to use web interface for easier accessibility.

The contribution of this research work can be viewed in three areas. First, it shows the integration of real-time environmental variables obtained from the OpenMeteo API with machine learning algorithms, this ensures that the prediction system continually and dynamically adapts to changing weather and environmental conditions. Second, it presents a comparative analysis of different machine learning algorithms; Random Forest, Decision Tree, Logistic Regression, and Support Vector Classifier, to justify the choice of Random Forest for this application. Third, the study also demonstrates the deployment of the trained model into a practical web application with an intuitive and easy to use frontend with a scalable backend, thereby bridging the gap between advanced flood prediction models and real-world accessibility for at-risk communities.

2. RELATED WORKS

Previous research works have reported different techniques for flood disaster management utilizing different technologies. In the research work by Johnson & Gobo [7], in Port Harcourt Metropolis, Nigeria, the researchers conducted an evaluation and modeling of the risk of flash floods due to excessive rainfall and soil permeability. The work utilized rainfall data from the University of California Centre for Hydrometeorology and Remote Sensing and the Nigerian Meteorological Agency, which covered the period from January 1981 to December 2016. The permeability, moisture content, and particle size distribution of soil samples collected from 48 sites in Port Harcourt were examined. The study used a GIS framework that included a Digital Elevation Model produced by the Shuttle Radar Topography Mission, with ArcGIS serving as the main analysis tool.

A review of flood risk analysis in Nigeria was presented by Olajuyigbe *et al.* [8]. The study focused on assessing flood hazards in Mile 12, Lagos, Nigeria, with emphasis on identifying the key factors contributing to recurrent flooding in the study area. The research utilized methods such as household questionnaires, key informant interviews (especially with Lagos State Physical Development Authority officials), participant observation, and data from secondary sources. The findings revealed that persistent flooding in Mile 12 is attributed to consistent high rainfall, water releases from the Oyan dam in Ogun state, blockage of drainage channels by refuse, narrow river channels, and construction along floodplains. Recommendations included creating sufficient setbacks for streams, constructing roads with robust drainage systems, channelization, and building more dams to alleviate the flooding issue.

A machine learning-based flood prediction model was introduced in Lawal *et al.* [9]. The study concentrated on flood prediction in Nigeria's Kebbi state utilizing machine learning algorithms. The researchers evaluated the effectiveness of three machine learning algorithms—Decision Tree, Logistic Regression, and Support Vector Classification (SVR)—using a historical rainfall dataset that covered 33 years. The goal was to develop a model that could forecast floods in Kebbi State and might be used in other high-risk flood states in Nigeria. According to their findings, Logistic Regression was the most accurate method, outperforming the other two in terms of accuracy and recall.

The Decision Tree also performed well, surpassing the Support Vector Classifier, with reasonable accuracy and below-average recall. Notably, the Support Vector Classification exhibited poor performance with a small dataset, yielding below-average accuracy, a recall score of 0, and an average ROC score. The findings highlight the potential of machine learning for flood prediction, with Logistic Regression standing out as the most effective model in this specific case study. However, Their work relied on only the historical datasets for both training and testing on the machine learning models which is prone to Their work relied on only the historical datasets for both training and testing on the machine learning models, which is prone to overfitting and may not generalize well to unseen or future data, a limitation which is solved in this work.

Thiemig *et al* [10] introduced the African Flood Forecasting System (AFFS), a medium-range ensemble system designed for probabilistic flood forecasting in African river basins with lead times of up to 15 days. The system's performance was evaluated in hindcast mode for the flood-prone year 2003, using ground observations from 36 sub-catchments and flood archive reports. AFFS successfully detected around 70% of reported flood events, particularly excelling in long-duration, large-scale riverine floods. However, its performance was limited for short-lived, small-scale events. A case study on the "Save flooding" event highlighted AFFS's accuracy in forecasting flood timing and severity, clear output products, and its ability to issue warnings in ungauged basins.

Table I: Summary of related works and gap analysis

Author	Title	Methodology Used	Limitations
Oloruntoba, Taiwo & Agbogun (2023)	Flood Prediction in Nigeria Using Ensemble Machine Learning	Bagged CART, Random Forest, Gradient Boosting, XGBoost, C5.0	Complex models; limited interpretability
Johnson & Gobo (2021)	A comprehensive assessment and modeling of flash flood risk in Port Harcourt Metropolis, Nigeria.	GIS-based approach to assess flash flood risk in Port Harcourt Metropolis, Nigeria.	The model's effectiveness relies on certain assumptions, and if these assumptions do not accurately represent the real-world conditions, it might affect the validity of the flash flood risk assessment.
Lawal, Yassin & Zakari (2021)	Flood prediction using machine learning models: a case study of Kebbi state Nigeria	Decision Tree, Logistic Regression, and Support Vector Classifier	Relied solely on historical datasets; limited generalization
Thiemig, Bisselink, Pappenberger & Thielen (2015)	A pan-African medium-range ensemble flood forecast system	Integrated hydrological model (LISFLOOD) with ensemble meteorological predictions	Limited performance for small-scale and short-duration flood events
Olajuyigbe, Rotowa & Durojaye (2012)	A review of flood risk analysis in Mile 12, Lagos, Nigeria	Household questionnaires, interviews, and secondary data	Results may be biased due to reliance on self-reported data

In Table 1, the authors' names, the focus of their work, the methodologies used, and the limitations of each study are presented. This format allows for a concise comparison of the different authors' contributions, their approaches, and the potential limitations of their work. The limitations of the reviewed existing works could be summarized as doubtful quality of datasets collected through questionnaires and failure to deploy these models into easy to formats for end users. This research work overcomes this by deploying REST APIs to capture real-time weather variables, these are then passed through the necessary steps of feature selection, cleaning, and training for a more reliable prediction, while at the same time presenting the results to users with an easy to use web interface for easier accessibility. The application of machine learning to solving problems across other domains has also been reported for example; for intrusion detection in computer systems ([12, 13]). In drilling optimization by Acheme *et al.* [14] etc.

3. MATERIALS AND METHODS

The flash flood prediction system is designed by integrating machine learning with modern web technologies to create an effective and user-friendly tool. The backend uses Python, taking advantage of its strong libraries like Scikit-learn for implementing machine learning. Random Forest algorithms are particularly used to predict flash floods based on various data inputs such as rainfall, river levels, and soil moisture. Random Forest is chosen for its accuracy and ability to handle complex data relationships. For the frontend, TypeScript and React.js are used to develop a scalable and easy-to-maintain user interface. Tree.js is included to provide 3D data visualizations, making it easier for users to interact with the prediction models and data. This combination ensures the system is both powerful and easy to use, offering real-time updates and dynamic visualizations to help in early warning and flood risk management. The architecture balances heavy backend processing with a responsive frontend, creating an effective tool for predicting and managing flash floods.

3.1 System Architectural Design

The design of the flash flood prediction system carefully integrates a strong backend with an intuitive frontend interface. Python is used at the core of the backend, utilizing powerful libraries like Scikit-learn for machine learning tasks, specifically employing Random Forest algorithms to predict flash floods based on various data inputs such as rainfall, river levels, and soil moisture. This method ensures high accuracy and reliability in the predictions. On the frontend, TypeScript and React.js are combined to develop a scalable and maintainable user interface. Additionally, Tree.js is integrated to facilitate sophisticated 3D data visualizations, enhancing user interaction and data comprehension. The system is designed

to offer real-time updates and dynamic visualizations, aiding in early warning systems and effective flood risk management. This architecture (Figure 1) strikes a balance between the demanding computational requirements of the backend and the responsive, user-friendly frontend, resulting in a powerful tool for predicting and managing flash floods.

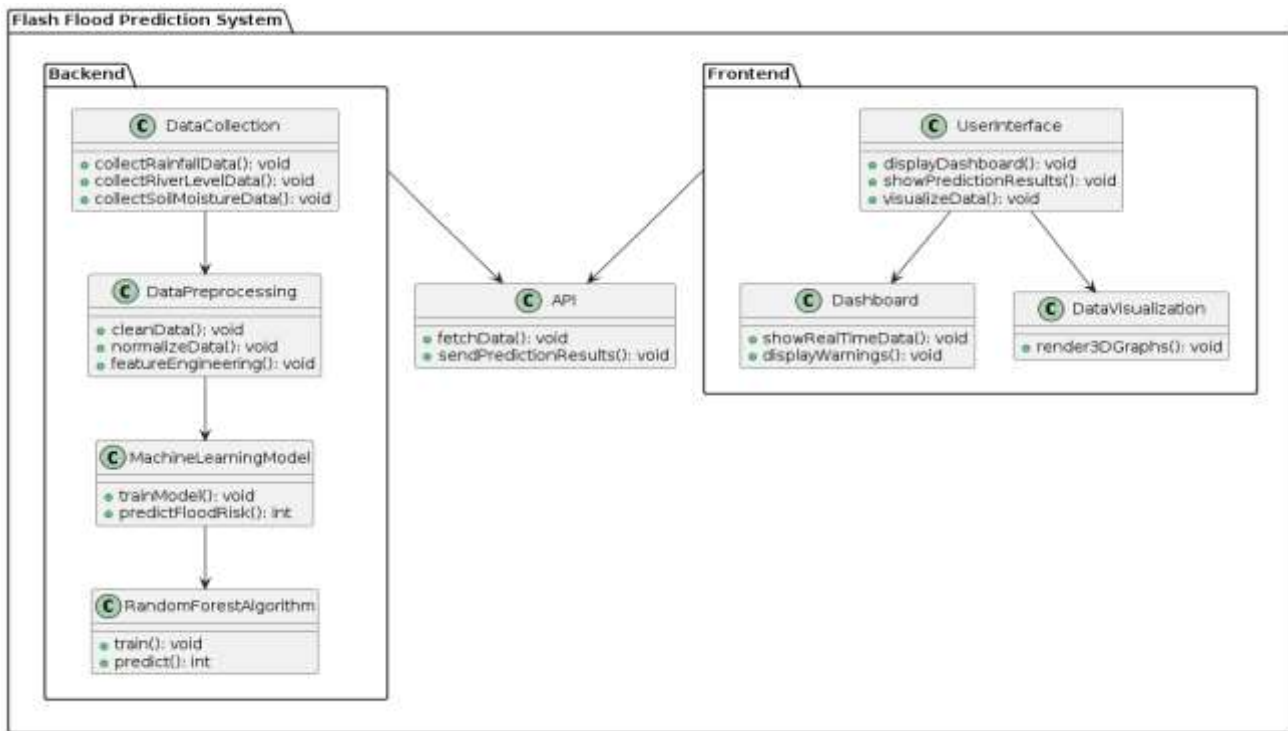


Figure 1: System architectural design

3.2 Data Collection

Historical weather data was retrieved from the Open Meteorology Weather API, a free and open-source weather data platform. The API provides access to a vast repository of historical weather data, spanning multiple decades. Using the API's query parameters, I specified the desired location, date range, and weather parameters (e.g., temperature, precipitation, wind speed) to extract the relevant data. The API returned the data in a JSON format, which was then processed and stored in a mongo database. The Open Meteorology Weather API offers a robust and reliable source of historical weather data, enabling me to train the prediction model.

In total, 15,320 records were collected from the OpenMeteo API, this covered a period of five years (2018–2023). Each of the records consisted of multiple features, including rainfall intensity, temperature, humidity, river discharge, soil moisture, wind speed, and visibility. These features were selected based on their established relevance to hydrological processes influencing flash flood events. The cleaned dataset was divided into training (70%), validation (15%), and testing (15%) subsets.

3.3 Data Flow Diagram

Figure 2 is the data flow diagram which shows the workings of the system. Initially, the user provides their location data to the UI Management, which then passes it to the Weather Data Retrieval component. This component requests current weather information from the Weather API and relays the obtained data back to the UI Management. Subsequently, the UI Management combines the user's location and weather data and sends them to the Prediction Request route. This component interacts with the Prediction Engine, which utilizes a Random Forest model to generate flood predictions. These predictions are then sent back to the Prediction Request route, which forwards the results to the UI Management for the user to see. Within the Prediction Engine, there are steps for data pre-processing and result formatting to ensure the predictions are precise and user-friendly.

The use case model for the flood prediction system details the main interactions between the user and the application, emphasizing essential features (Figure 3). Users can permit the system to access their location, request up-to-date weather information, and start a flood prediction. In response, the system retrieves weather data from an external Weather API, processes it alongside the user's location using the Prediction Engine, and creates flood predictions using a Random Forest algorithm. These predictions are then formatted and shown to the user. This model demonstrates the system's ability to deliver precise and timely flood predictions based on the user's location and real-time weather conditions.

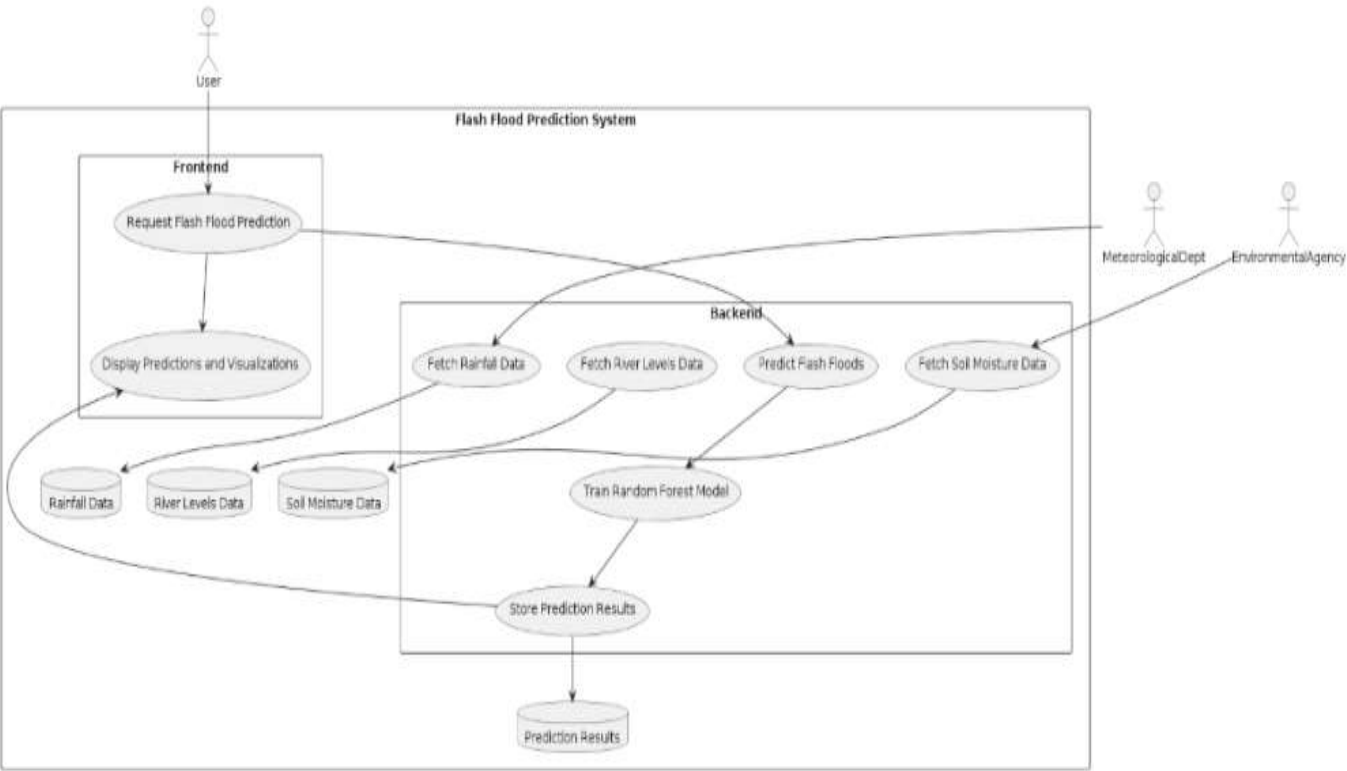


Figure 2: Data flow diagram

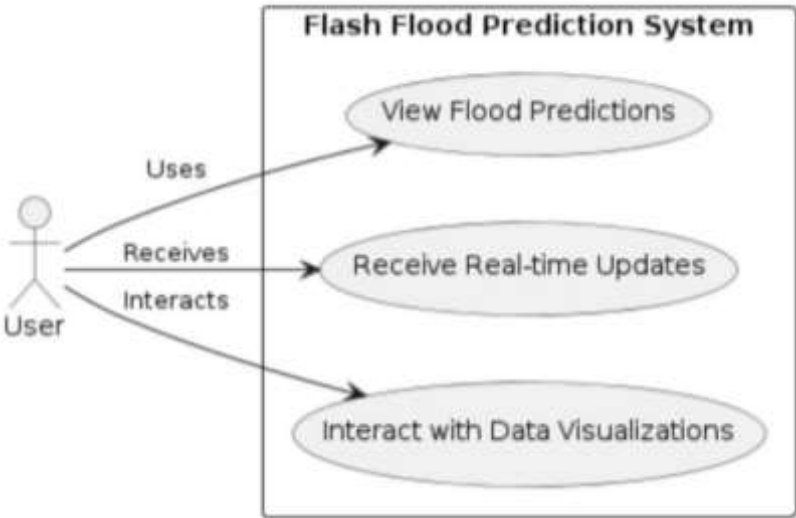


Figure 3: Use case model

3.4 Database Design

The database model for the flood prediction system is based on MongoDB and centers around one main collection that contains historical flood data. Each entry provides a full picture of previous flood events, which helps the Prediction Engine identify patterns and enhance prediction accuracy. By using MongoDB's flexible schema, the system can effectively store and access this historical data, ensuring that it can make strong and dependable flood predictions using both past events and current weather information.

4. RESULTS AND DISCUSSION

4.1 Data Collection

To collect data from the OpenMeteo API for the flash flood prediction system, we began by signing up on the OpenMeteo website to obtain the api url, which granted access to their weather data services. With this url, a PostMan is used to send the HTTP GET requests to the OpenMeteo API endpoints. The request is configured to query for specific data

such as historical rainfall intensity, temperature, humidity, and other relevant meteorological information for the targeted geographic areas prone to flash floods. The API requests were set up to run at regular intervals to ensure real-time data collection. The responses, returned in JSON format, were parsed to extract the necessary data fields, which were then cleaned and validated to remove any anomalies or missing values. The validated data was subsequently stored in a MongoDB database, structured to support efficient querying and integration with the machine learning models used for flood prediction. This automated data collection process ensured a continuous and accurate flow of weather data, which is important for timely and precise flash flood forecasting. Figure 4 shows the fields/column headers of the dataset retrieved.

To better understand the dataset, visualizations were generated. Figure 5a shows a histogram of rainfall intensity distribution, while Figure 5b shows the correlations among the main features of the dataset which are; rainfall, temperature, humidity, and river discharge. From Figure 5b, Strong positive correlations can be clearly observed between rainfall and river discharge, while negative correlations were noted between temperature and soil moisture. These insights confirm the relevance of selected features to flood prediction.

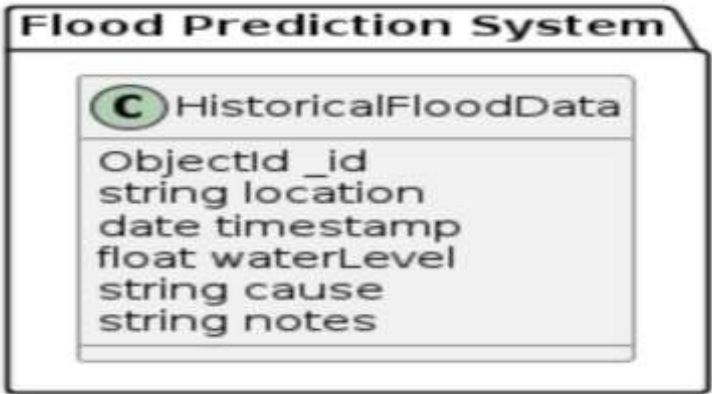


Figure 4: Database design

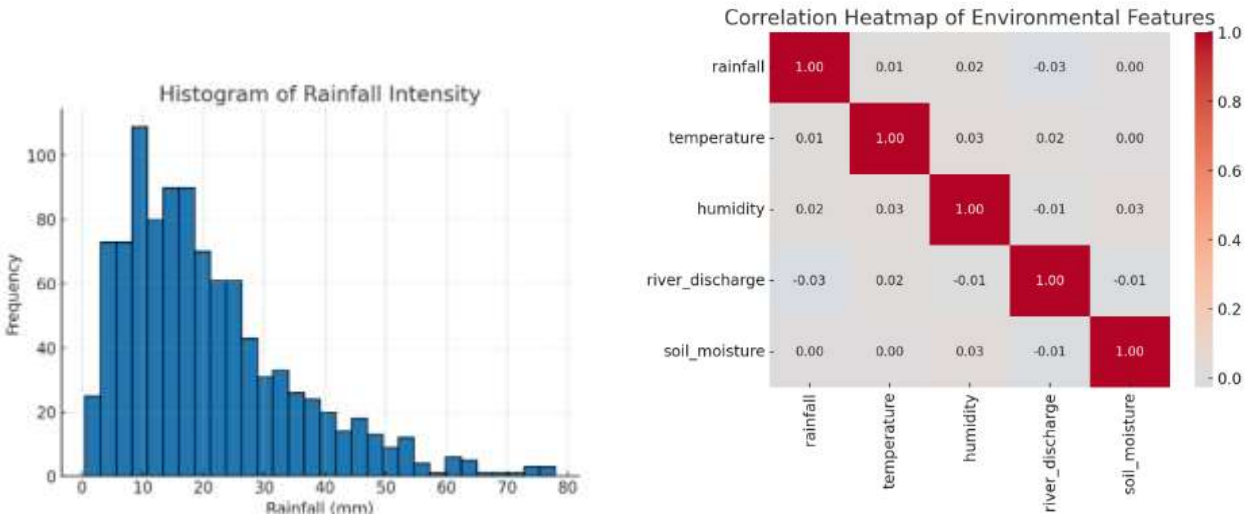


Figure 5a: Rainfall intensity

Figure 5b: Correlation heatmap of chosen environmental variables

4.2 Model Training and Implementation

The Scikit-learn module in Python was used to train a random forest model. The obtained data was first cleaned, normalized, and divided into training and testing sets. To enhance performance, the hyperparameters were adjusted using cross-validation on the training data. In addition to the the Random Forest algorithm which was selected due of its robustness against overfitting and its capacity to manage intricate, non-linear interactions, three other machine learning algorithms were also modelled for comparison and evaluation purposes.

After training the model, the model was evaluated for its effectiveness with the testing data, focusing on metrics like precision, recall, and F1 score. Once satisfactory results were achieved, the model was deployed into production. This involved integrating it with the backend of the prediction system and setting up an API endpoint to receive input data and return predictions, ensuring the system was scalable and responsive in real-time.

To maintain prediction accuracy over time, continuous monitoring and updating of the model to adapt to changing environmental conditions was implemented. Additionally, we exported the trained model using the Python joblib package for easier access. This comprehensive approach helped develop a reliable and effective flash flood prediction system, providing valuable insights for disaster preparedness and risk management.

To justify the selection of Random Forest, we compared its performance against three others widely used machine learning algorithms: Decision Tree, Logistic Regression, and Support Vector Classifier (SVC). The comparison used the same dataset split and evaluation metrics. Table 2 presents the results.

Table 2 – Performance comparison of machine learning models

Model	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
Decision Tree	91.2	71.0	83.5	76.7
Logistic Regression	89.8	68.4	80.1	73.8
Support Vector Classifier	87.6	65.0	78.2	70.9
Random Forest	96.0	75.0	91.0	82.2

The Random Forest algorithm achieved the highest accuracy, recall, and F1-score, demonstrating superior performance in detecting positive flood cases. This robustness and generalization ability make it more suitable for real-time deployment compared to the other models.

4.3 Application Design

The application design stage included the development of the various components of the system. The Model-View-Controller (MVC) architecture was utilized in this study. It partitions an application program into three layers, each with different functions.

- Model Layer: This is not the same as the database design. It is the creation of the layer of code which provides access to the database.
- Controller Layer: This is likewise the business rationale layer. In a web based application, for example, that actualized in this study, the controller layer handles the progression of the site pages as a client executes various tasks.
- View Layer: This layer incorporates the different windows, interfaces, structures and so on that the client cooperates with. This layer houses the code that speaks with the controller object.

In this study we implemented this via Representational State Transfer (REST) API architecture where the front-end view is displayed with React JS depending on data received from the controller. The implementation phase of the systems development life cycle (SDLC) is usually the most time- consuming period of the entire life cycle. In fact, implementation is the accomplishment of a technical specification or algorithm and software module, through computer programming and deployment. At this stage, the physical design specifications were transformed into a working computer code. After which the code passed through the testing stage until the vast majority of the errors and bugs were distinguished and rectified, the system was then installed.

The prediction results are incorporated into the Home Page (Figure 6) through a modular component that interacts with the backend via REST API calls. Once a user requests a prediction, the system retrieves real-time weather variables, processes them through the trained model, and displays the prediction outcome in an interactive modal. To ensure a structured implementation, the Software Development Life Cycle (SDLC) methodology was adopted. Specifically, the iterative prototyping approach was used, allowing continuous refinement of the user interface, data integration, and backend logic. This systematic adoption of SDLC not only facilitated the emergence of a new integrated architecture but also ensured that the final system met usability, reliability, and scalability requirements.

Figure 7 are pop ups where a button for prediction resides, if clicked a request is sent to the backend which calls the model and returns various results and outputs as shown in Figures 8 – 12.

5. CONCLUSION

This research work has reported a flash flood prediction system which is designed to provide accurate and real-time flood forecasts to help communities prepare for and respond to flash floods. In designing the system, python programming language was used for backend processing, which was utilizing selected machine learning algorithms in order to analyze the environmental data such as rainfall intensity, river water levels, and soil moisture content. We developed the frontend of the system with TypeScript, React.js, and Tree.js, this provided an intuitive and interactive interface for users to visualize flood predictions and receive timely alerts. By integrating these machine learning models with modern web interfaces, our system aims to enhance disaster risk management and improve community preparation against flash floods.

In conclusion, the flash flood prediction system which we presented in this work, combined selected machine learning algorithms with modern web development tools to provide an effective tool for accurate and timely flood forecasts. By utilizing Python for data analysis and processing, along with selected machine learning algorithms, the system ensures high accuracy in predicting flood events. The user-friendly user interface which was built with TypeScript,

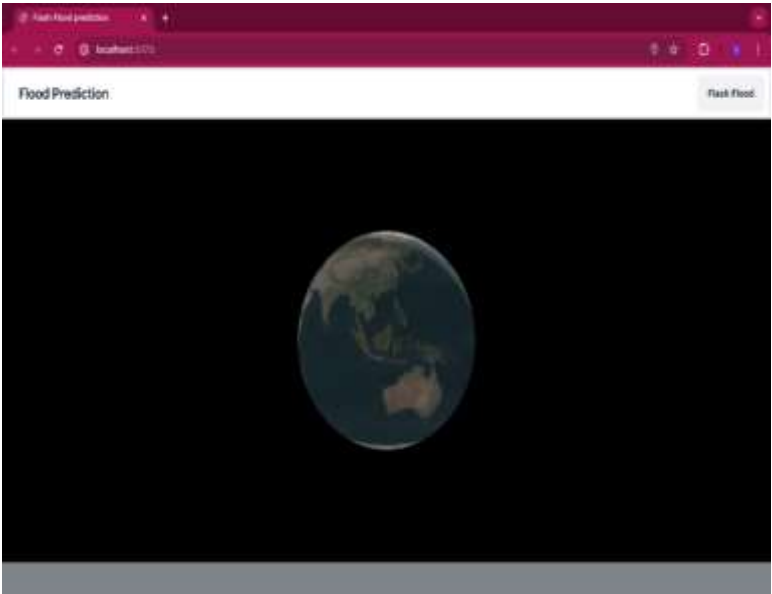


Figure 6: Home page with an interactive 3d model of the earth globe.

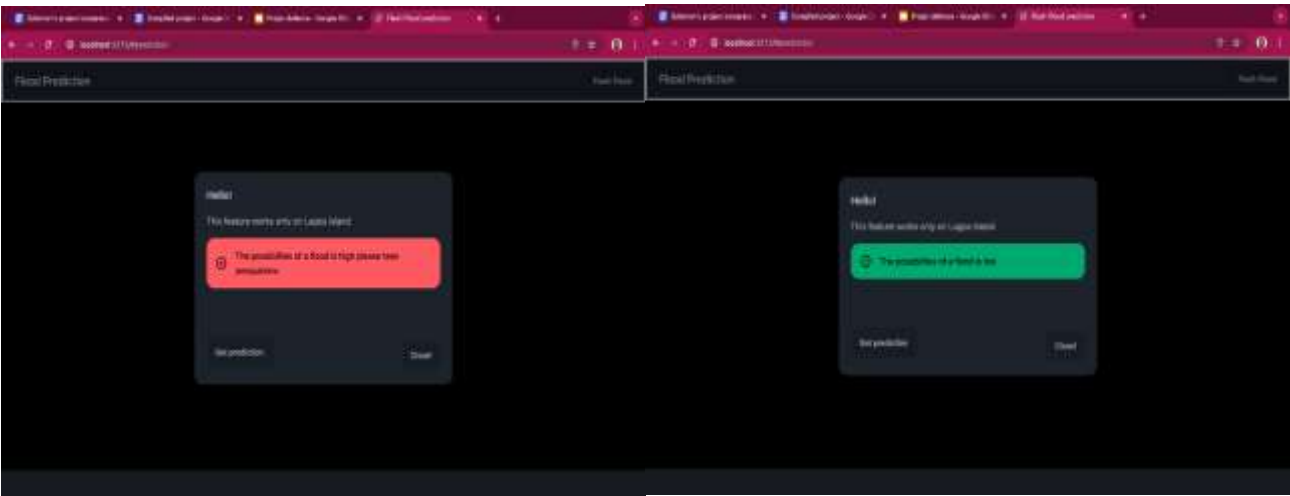
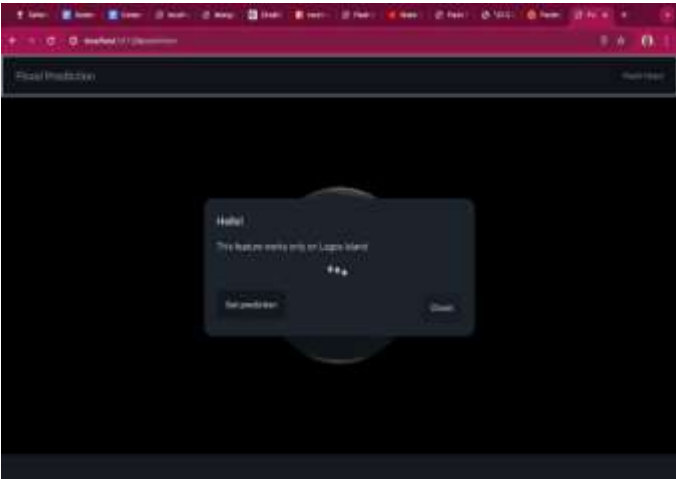


Figure 7: Web interfaces showing operation of the app

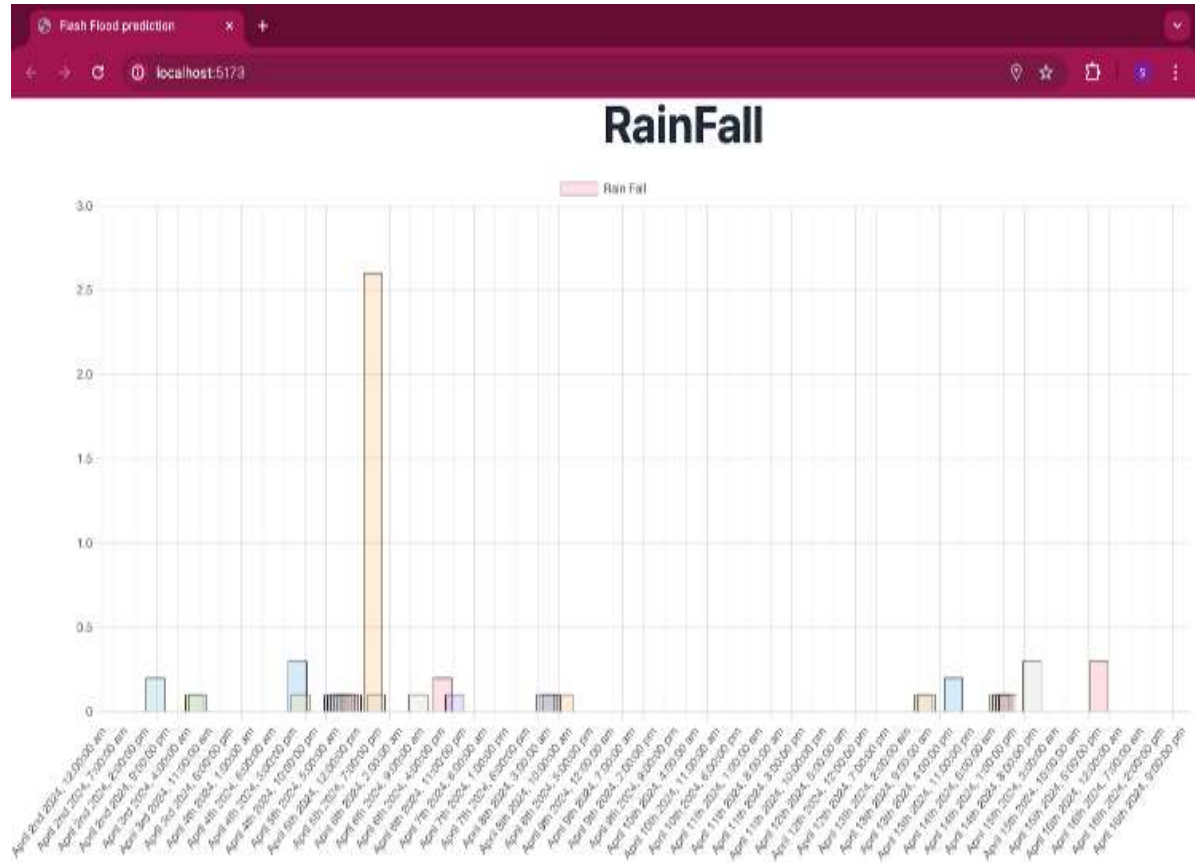


Figure 8: A graph displaying real time rainfall data based on the current location of the user

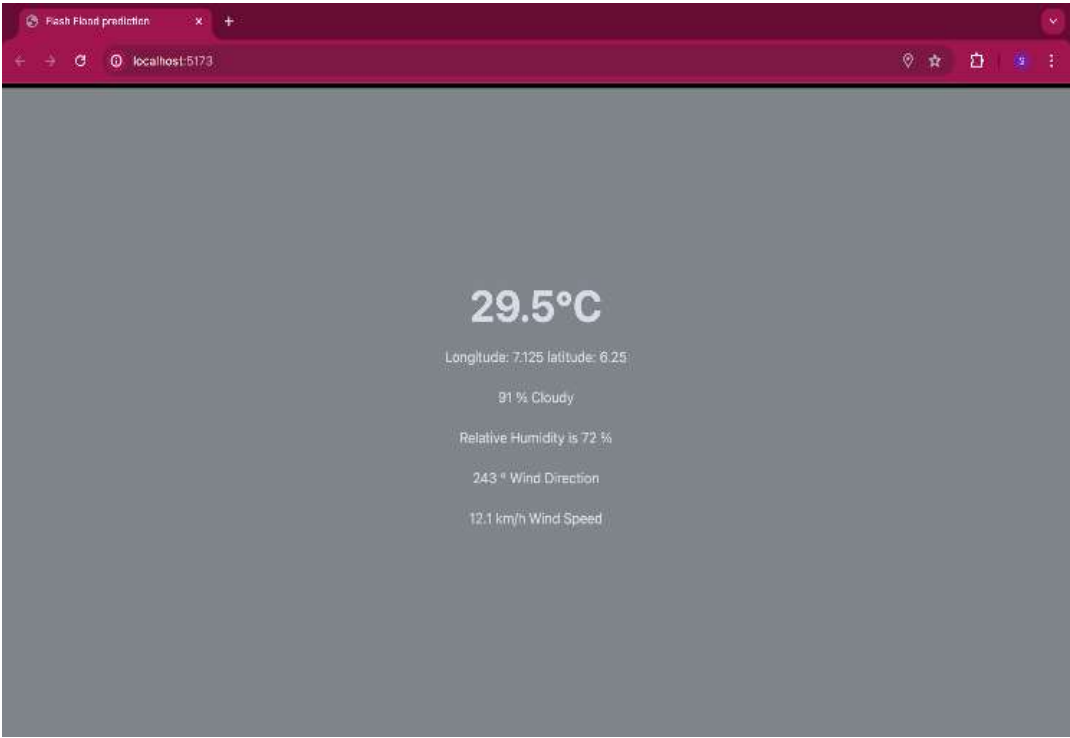


Figure 9: A component installed on the homepage that presents real-time weather data according to the user's current location.



Figure 10: A component mounted on the home pages that requests and displays a real time visibility graph based on the user's current location.

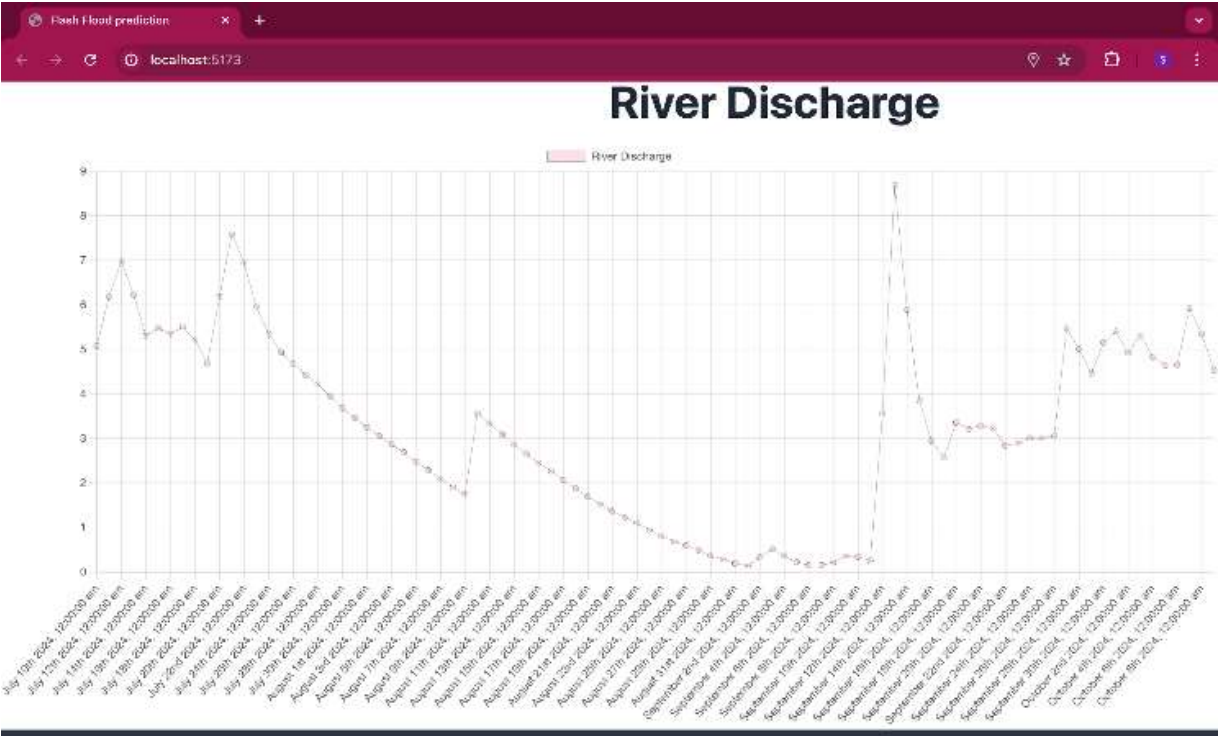


Figure 11: This component requests for real time river discharge data using the user's current location and displays the data in a graph component mounted on the home screen.



Figure 12: The nav bar section, this section has a button that loads up the prediction model.

React.js, and Tree.js, also provides an easy to use interactive platform for visualizing data and receiving real-time alerts. This integration is helpful as it improves the efficiency of disaster response and also enhances community preparedness against flash floods. The research work demonstrates the potential of technology in mitigating the impacts of natural disasters and safeguarding lives and property. The comparative analysis of the different machine learning algorithms further validates the choice of Random Forest as the optimal algorithm for this task, outperforming Decision Tree, Logistic Regression, and Support Vector Classifier across all the selected performance metrics.

For future improvements and wider adoption, it is recommended that the flash flood prediction system incorporate additional data sources, such as satellite imagery and social media feeds, to enhance prediction accuracy. Implementing machine learning models that continuously learn from new data could further refine the system's predictions over time. Collaboration with local governments and emergency response teams can facilitate better data sharing and integration, improving overall system efficacy. Additionally, developing a mobile application version would increase accessibility, allowing users to receive real-time alerts and updates on the go. Regular updates and user feedback should be integrated to ensure the system remains relevant and effective in diverse and changing environments.

REFERENCES

- [1] Agbonaye, A. I., & Izinyon, O. C. (2024). Nigerian Coastal Region's Vulnerability to Climate Change. *Journal of Energy Technology and Environment*, 6(1), 32-48.
- [2] Munawar, H. S., Hammad, A. W., & Waller, S. T. (2022). Remote sensing methods for flood prediction: A review. *Sensors*, 22(3), 960.
- [3] Fanta, S. S., & Sime, C. H. (2022). Performance assessment of SWAT and HEC-HMS model for runoff simulation of Toba watershed, Ethiopia. *Sustainable Water Resources Management*, 8, 1-16.
- [4] Verma, R. K., Verma, S., Mishra, S. K., & Pandey, A. (2021). SCS-CN-based improved models for direct surface runoff estimation from large rainfall events. *Water Resources Management*, 35(7), 2149-2175.
- [5] Bui, D. T., Hoang, N. D., Martínez-Álvarez, F., Ngo, P. T. T., Hoa, P. V., Pham, T. D., ... & Costache, R. (2020). A novel deep learning neural network approach for predicting flash flood susceptibility: A case study at a high frequency tropical storm area. *Science of The Total Environment*, 701, 134413.
- [6] Kumar, V., Sharma, K. V., Caloiero, T., Mehta, D. J., & Singh, K. (2023). Comprehensive overview of flood modeling approaches: A review of recent advances. *Hydrology*, 10(7), 141.
- [7] Johnson, D. V., & Gobo, A. E. (2021). Assessment and modeling of flash flood risk from extreme rainfall and soil permeability in Port Harcourt Metropolis, Nigeria. *Global Journal of Human-Social Science Geography, Geo-Sciences, Environmental Science and Disaster Management*, 21(5), 1-15. <https://doi.org/10.1177/2345678912345678>
- [8] Olajuyigbe, A. E., Rotowa, O. O., & Durojaye, E. (2012). An assessment of flood hazard in Nigeria: The case of mile 12, Lagos. *Mediterranean Journal of Social Sciences*, 3(2), 367-375.
- [9] Lawal, Z. K., Yassin, H., & Zakari, R. Y. (2021, December). Flood prediction using machine learning models: a case study of Kebbi state Nigeria. In *2021 IEEE Asia-Pacific Conference on Computer Science and Data Engineering (CSDE)* (1-6). IEEE.
- [10] Thiemig, V., Bisselink, B., Pappenberger, F., & Thielen, J. (2015). A pan-African medium-range ensemble flood forecast system. *Hydrology and Earth System Sciences*, 19(8), 3365-3385.
- [11] Oloruntoba, K. R., Taiwo, K., & Agbogun, J. B. (2023). Flood Prediction in Nigeria Using Ensemble Machine Learning Techniques. *Ilorin Journal of Science*, 10(1), 44-61.
- [12] Acheme, I. D., Nwankwo, W., Olayinka, A. S., Makinde, A. S., & Nwankwo, C. P. (2023, March). Petroleum Drilling Monitoring and Optimization: Ranking the Rate of Penetration Using Machine Learning Algorithms. In *The International Conference on Artificial Intelligence and Logistics Engineering* (152-164). Cham: Springer Nature Switzerland.
- [13] Acheme, I. D., Wasiu, A. A., & Edegbe, G. N. (2024). A network intrusion prediction model using Bayesian network. *World Journal of Advanced Research and Reviews*, 23(1), 2813-2821.
- [14] Acheme I. D., Makinde A. S. & Uddin O. O. (2023) Investigating and Ranking the Rate of Penetration (ROP) Features for Petroleum Drilling Monitoring and Optimization *International Journal of Innovative Science and Research Technology* 8(10).