iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 14 **Issue:** V **Month of publication:** May 2026

DOI: https://doi.org/10.22214/ijraset.2026.81840

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 14 Issue V May 2026- Available at www.ijraset.com

A Smart Flood Detection and Warning Prototype Using IoT Sensors and ML Model

Darshan Kawale[1], Vishveshvar Bidve[2], Indrayani Raut[3], Prof. Shilpa Kale[4]
Department of Electronics & Telecommunication Engineering, K J College of Engineering & Management Research, Pune

Abstract: ***Floods are natural disasters that cause significant damage to infrastructure, agriculture, and human life. They can cause a lot of problems and people can lose their lives and homes. This is very sad. The systems we have now to monitor floods are very expensive. They are also made for places. Most of these systems use rules that do not work well and are hard to make bigger. This paper is talking about a system that uses the Internet of Things or IoT for short. This system uses devices called sensors to get information in real time. These sensors measure things like how high the water's how much rain is falling, how wet the soil is and the temperature and humidity. The ESP8266 microcontroller is what collects all this information. Then this information is sent to a server using something called HTTP.A special computer program called a machine learning model is trained on information about floods. This program can tell if a flood is likely to happen. The system can send warnings on time. It uses a webpage called a dashboard and also sends messages to peoples phones and has local signs, like a loud noise and a flashing light. By using IoT and machine learning this system is more accurate. It is also cheaper. Can be used in many places to monitor floods.***

Keywords: IoT, Flood Prediction, ESP8266, Machine Learning, DHT22, Ultrasonic Sensor, GSM, SIM800L, Real-Time Monitoring, Random Forest.

I. INTRODUCTION

Floods are severe natural disasters causing significant damage to infrastructure, agriculture, and human life They result in substantial socio-economic losses and human casualties. Because of climate change and more people moving to cities Floods happen often. Are worse. So it is very important to know when a Flood is coming and warn people on time.The old way of watching for Floods uses machines or people looking or simple methods. These are expensive and not very accurate. They do not work well for towns. New technology like the Internet of Things and Machine Learning can help make Flood detection systems that're cheap, smart and work in real time. The Internet of Things lets us watch the environment all the time with sensors. These sensors measure water level, rain, soil moisture, temperature and humidity.Machine Learning models look at the data from these sensors. They tell us if a Flood is coming accurately than simple systems.This project tries to make a device that uses sensors and a Machine Learning model. It watches Flood conditions in time. The device uses a computer called ESP8266. It gets data from the sensors. Sends it to a server. The Machine Learning model looks at the data.

If a Flood is predicted the system warns people with a noise, lights and a screen. It also sends text messages. We tested this device inside with Flood conditions. It is good for school research. Can be used in the real world.The project uses Floods and Machine Learning to make a system that helps people. Floods are a problem. Machine Learning can help us solve this problem. We want to make a system that can predict Floods and warn people on time. The device is small and cheap. It can be used in places. We hope that this project will help people and make a difference Floods will always be a problem, With this system we can be more prepared, for Floods.

II. LITERATURE SURVEY

Several researchers have explored the application of IoT and machine learning techniques for flood monitoring and early warning systems. Ahmed and Thomas [1] introduced an IoT-enabled environmental monitoring framework primarily focused on agricultural applications, demonstrating the effectiveness of sensor-based automation in real-time data acquisition. Extending this concept, Kumar et al. [2] developed a flood prediction model integrating IoT sensing with machine learning, achieving improved prediction accuracy using historical rainfall and water level data.

Patel et al. [3] proposed a multi-sensor IoT-based flood monitoring system combined with machine learning algorithms to generate early warnings, particularly in urban environments. Similarly, Verma et al. [4] emphasized the use of cloud computing for processing real-time sensor data, enabling scalable and remote flood monitoring solutions. Singh et al. [5] introduced a hybrid IoT-ML framework that demonstrated better performance compared to traditional threshold-based systems by reducing false alarms and improving response time.

Further studies by Sharma et al. [6] highlighted the importance of multi-parameter sensing using pressure and moisture sensors for accurate environmental monitoring. Reddy et al. [7] explored advanced machine learning models, including decision trees and LSTM networks, to enhance prediction accuracy using time-series sensor data. Patel et al. [8] focused on decentralized IoT systems using wireless communication for cost-effective flood detection in community-level deployments. Additionally, Verma et al. [9] incorporated GSM-based alert mechanisms for real-time user notification, improving system responsiveness during emergency situations. Kumar et al. [10] demonstrated that fusion-based machine learning models significantly improve the robustness and reliability of flood prediction systems.

Despite these advancements, most existing systems face limitations such as high implementation cost, limited real-time scalability, dependence on single-parameter sensing, or lack of integrated alert mechanisms. Furthermore, many models rely heavily on historical datasets and lack adaptability to dynamic real-time environmental conditions.

Therefore, there is a need for a low-cost, scalable, and real-time flood monitoring system that integrates multi-sensor IoT data with machine learning techniques and provides reliable early warning alerts. The proposed system aims to address these limitations by combining real-time data acquisition, multi-parameter analysis, and efficient communication mechanisms.

III. HARDWARE ARCHITECTURE

A. *System Overview*

The system I want to talk about uses the ESP8266 NodeMCU microcontroller. The ESP8266 microcontroller acquires sensor data and transmits it to the server for processing. The ESP8266 NodeMCU works with five sensors that help gather information, about what's happening around us and what time it is.The ESP8266 NodeMCU microcontroller is really important here because it gets data from these five sensors.It then sends this data to places so the ESP8266 NodeMCU microcontroller data can be used again.

These sensors include:

- An HC-SR04 sensor that measures how much water is in a place.
- A YL-83 rain sensor that detects when it is raining.
- A DHT22 sensor that measures the temperature and humidity.
- A soil moisture sensor that checks how much water is in the ground.
- A SIM800L GSM module that sends messages to peoples phones when there is a problem.

The system works in a way: the sensors send information to the ESP8266 NodeMCU microcontroller, which then sends this information to a server using the internet. The server uses this information to decide if it needs to send an alert to someone. The ESP8266 NodeMCU microcontroller is a part of this system. It is the thing that makes the system work.

B. *Component Descriptions*

- The ESP8266 NodeMCU: is a computer that can connect to the internet. It runs on an amount of power and can send information to a server. The ESP8266 NodeMCU reads the information from the sensors. Sends it to the server. This is what the ESP8266 NodeMCU does.
- The HC-SR04 Ultrasonic Sensor: measures how much water is in a place by sending out waves and measuring how long it takes for them to come back. This sensor can measure distances from a centimeters to several meters very accurately. It is very good at measuring water levels.
- The DHT22 Sensor: measures the temperature and humidity in the air. It can measure very hot temperatures, as well as very dry and very humid air. This sensor needs a wire to work properly. The DHT22 Sensor is important for knowing the temperature and humidity.
- The YL-83 Rain Sensor: detects when it is raining by checking if the rain is touching its pad. This sensor can tell us how hard it is raining. If it is raining all. The YL-83 Rain Sensor is useful for knowing when it is raining.
- The Soil Moisture Sensor: measures how much water is in the ground. It sends this information to the ESP8266 NodeMCU microcontroller, which turns it into a percentage. This helps us know if the ground is dry or wet. The Soil Moisture Sensor is important for knowing the condition of the ground.
- The SIM800L GSM Module: sends messages to peoples phones when there is a problem, like a flood. This module needs an adapter to work properly because it uses a lot of power. The SIM800L GSM Module is useful for sending alerts.
- The Buzzer and LED: Indicators are, like a warning system. The green light means everything is okay. The yellow light means we need to be careful. The red light and buzzer mean there is a problem. The ESP8266 NodeMCU microcontroller and the sensors work together to turn on these warnings. The Buzzer and LED Indicators help us know what is happening

ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 14 Issue V May 2026- Available at www.ijraset.com

IV. SYSTEM DESIGN & METHODOLOGY

A. Block Diagram

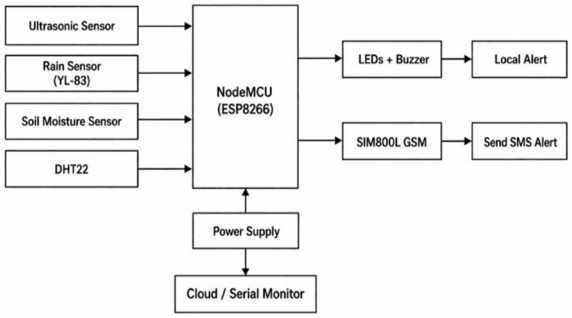


Fig. 1. Block Diagram of Smart Flood Detection and Early Warning System

The block diagram shown in Fig. 1 represents the overall architecture of the proposed smart flood detection and early warning system.

The system is centered around the NodeMCU (ESP8266) microcontroller, which acts as the primary data acquisition and control unit. Various environmental sensors, including an ultrasonic sensor for water level measurement, a rain sensor (YL-83) for rainfall detection, a soil moisture sensor for ground saturation monitoring, and a DHT22 sensor for temperature and humidity measurement, are interfaced with the NodeMCU.

The acquired sensor data is processed by the NodeMCU and utilized for monitoring and decision-making. A power supply unit provides regulated power to all components in the system. Based on the processed data, the system generates local alerts through LEDs and a buzzer. Additionally, a SIM800L GSM module is connected to enable SMS alert notifications.

The system also supports monitoring through a cloud or serial monitor interface, allowing real-time observation of environmental conditions. This architecture ensures effective data acquisition, alert generation, and system monitoring for flood detection applications.

B. Sensor Data Acquisition

The ESP8266 gets sensor data every thirty seconds. It works out the water level by using the time it takes for something to travel to the water and back. Then it turns this into centimeters. The sensor also checks how wet the soil is. It checks how hard it is raining. It turns these into a scale from zero to one hundred percent. The temperature and humidity are read from the DHT22 sensor. All of the sensor data is stored in a format called JSON. This is sent to the Flask backend server using a kind of request called an HTTP POST request.

C. Machine Learning Model

We use something called a Random Forest Classifier that we trained using flood data from Kaggle and our own sensor data from the ESP8266. We look at things like the water level from the sensor data soil moisture from the sensor data, humidity from the sensor data, temperature from the sensor data and rainfall intensity from the sensor data. The model then tells us if it is Safe or if we should be Warned or if it is a Danger. Before we train the model we do some things to the sensor data to make it better. We fill in missing values make sure all the values are on the scale and add data to the classes that do not have enough sensor data from the ESP8266. We save the model using a Python thing called joblib and put it on the Flask server.

D. Alert & Communication System

When the machine learning model says it is a Danger the server sends a message back to the ESP8266. This makes the red light turn on. The buzzer goes off. At the time the SIM800L module sends a text message that says "FLOOD WARNING: Water Level critical. Take action." We also have a dashboard that shows the sensor data from the ESP8266 over time. It has graphs and colors to show if there is a flood risk using the sensor data from the ESP8266. The dashboard is made with a thing called Chart.js. It works with the Flask server and HTML. The sensor data from the ESP8266 is used to make the dashboard and the sensor data, from the ESP8266 is used to make the dashboard show the flood risk.

E. Algorithm

1. I switch on the system. Get everything prepared. This means I ensure all sensors and communication parts like Wi-Fi and GSM are functioning properly.
2. Next I check if Wi-Fi is on and working If it is I. Upload the data., If not I use GSM mode.
3. I examine the sensor readings, which tell me:
   The current water level, Soil moisture, Humidity, Temperature, Rainfall intensity
4. I convert sensor readings into units.
5. I send sensor data to a server using HTTP POST in a format the server can understand.
6. The server uses a computer program, Random Forest to analyze the data.
   It predicts what will happen next which can be Safe, Warning or Danger.
7. I make a decision based on the prediction.
   If it says Safe I turn on the Green LED,
   If it says Warning I turn on the Yellow LED.
   If it says Danger I turn on the Red LED and Buzzer. Send a message.
8. I store the data in a database, such as SQLite or MySQL to improve the computer program.
9. I wait for a while restart, from step 3.
10. The process stops.

V. RESULTS & DISCUSSION

A. Experimental Setup

The prototype was put together. Tested inside a building where we could control things. We wanted to see how it would work in flood situations. We used a bucket to make the water level go up a spray bottle to make it rain and a tray of soil to make the ground all soggy. The prototype was turned on all the time for two hours. We wrote down what the sensors said every 30 seconds. The prototype and the sensors were a part of the test. We used the prototype and the sensors to see how well the system would work in a flood.

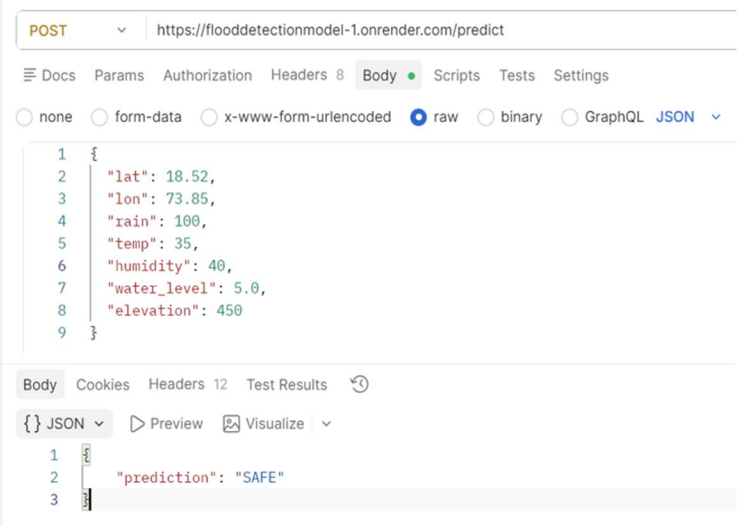


Fig. 2. API Request for Flood Prediction Model (SAFE Condition)

Fig. 2 illustrates the API request and response for the flood prediction system using a REST interface. The sensor data, including latitude, longitude, rainfall, temperature, humidity, water level, and elevation, is sent to the prediction endpoint in JSON format. The backend processes this data using the trained machine learning model and returns the prediction result as “SAFE”, indicating normal environmental conditions with no immediate flood risk.

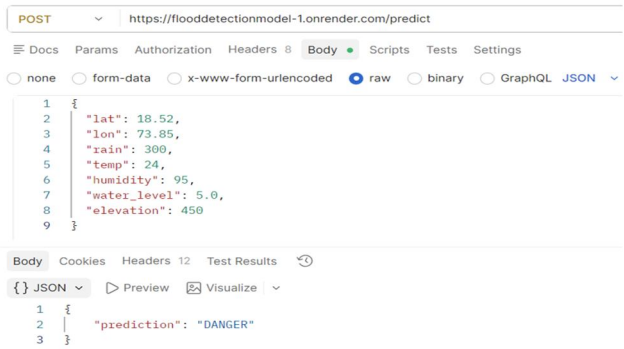


Fig. 3. API Request for Flood Prediction Model (DANGER Condition)

Fig. 3 shows the API request and response under different environmental conditions. The input sensor data is transmitted to the same prediction endpoint using HTTP POST method. Based on the provided values, the backend machine learning model classifies the situation as "DANGER", indicating a high risk of flooding. This demonstrates the system's ability to dynamically analyze real-time data and generate appropriate flood risk predictions.

B. Observed Sensor Readings

Condition	Water Level (cm)	Soil Moisture (%)	Humidity (%)	Rainfall (%)	ML Prediction	Response
Normal	5 - 15	20–40	40–60	0–20	SAFE	Green LED
Light Rain	15–30	40–60	60–75	20–50	WARNING	Yellow LED
Heavy Rain	30–50	60–80	75–90	50–80	DANGER	Red LED + SMS
Overflow	>50	>80	>90	>80	DANGER (Severe)	Alarm + SMS

Table I: Sensor Readings and System Response under Different Conditions.

C. ML Model Performance

- Training Accuracy: 96.4%
- Testing Accuracy: 92.8%
- Precision (Danger class): 0.91
- Recall (Danger class): 0.94
- F1-Score (Danger class): 0.925

The Random Forest model effectively distinguished between Safe, Warning, and Danger classes by analyzing the combined multi-sensor input. The integration of multiple parameters significantly improved prediction reliability over single-sensor threshold-based approaches.

D. System Performance Summary

- Response time from sensor trigger to alert: 3–6 seconds
- GSM SMS delivery delay: 4–6 seconds
- IoT dashboard refresh rate: ~2 seconds
- Ultrasonic sensor accuracy: ±1 cm error
- Power consumption: ~3W (energy efficient)
- System uptime during testing: 100% (2-hour continuous run)

VI. CONCLUSION

This paper presented a Smart Flood Detection and Early Warning Prototype. It effectively combines Internet of Things sensing with machine learning for time environmental monitoring and flood prediction. The system keeps collecting data from five sensors. These include water level, rainfall, soil moisture, temperature and humidity sensors. The data is sent to a server. There a Random Forest machine learning model classifies the flood risk level. When the system detects danger it sets off alarms. It also sends SMS alerts through GSM within seconds. The prototype works well.

It achieved a testing accuracy of ninety-two point eight percent. The response time is between three and six seconds. It showed performance in indoor simulated flood conditions. The hardware cost is low one thousand nine hundred and fifty rupees. The modular design makes it a practical and scalable solution. It can be used for community-level deployment, in flood- regions. Future work will explore some features. These include LoRa-based long-range communication and solar-powered nodes. The team will also look into edge AI inference. They plan to integrate the system with disaster management platforms.

REFERENCES

[1] K. Vinothini and S. Jayanthy, "IoT Based Flood Detection and Notification System using Decision Tree Algorithm," 2019 Int. Conf. on Intelligent Computing and Control Systems (ICCS), Madurai, India, pp. 1481–1486, May.2019.[Online].Available: https://ieeexplore.ieee.org/document/9065799/

[2] A. A. Rashid, M. A. M. Ariffin, and Z. Kasiran, "IoT-Based Flash Flood Detection and Alert Using TensorFlow," 2021 11th IEEE Int. Conf. on Control System, Computing and Engineering (ICCSCE), Penang, Malaysia, pp. 80–85, Nov. 2021. [Online]. Available: https://ieeexplore.ieee.org/document/9530926/

[3] A. Pravin, T. P. Jacob, and R. Rajakumar, "Enhanced Flood Detection System using IoT," 2021 6th Int. Conf. on Communication and Electronics Systems (ICCES), Coimbatore, India, pp. 507–510, Jul. 2021. [Online]. Available: https://ieeexplore.ieee.org/document/9489059/

[4] F. S. Mousavi, S. Yousefi, H. Abghari, and A. Ghasemzadeh, "Design of an IoT-Based Flood Early Detection System Using Machine Learning," 2021 26th Int. Computer Conf., Computer Society of Iran (CSICC), Tehran, Iran, pp. 1–7, Mar. 2021. [Online]. Available: https://ieeexplore.ieee.org/document/9420594/

[5] T. Sharma, A. Pal, A. Kaushik, A. Yadav, and A. Chitragupta, "A Survey on Flood Prediction Analysis Based on ML Algorithm Using Data Science Methodology," 2022 IEEE Delhi Section Conf. (DELCON), New Delhi, India, pp. 1–7, Feb. 2022. [Online].Available: https://ieeexplore.ieee.org/document/9753396/

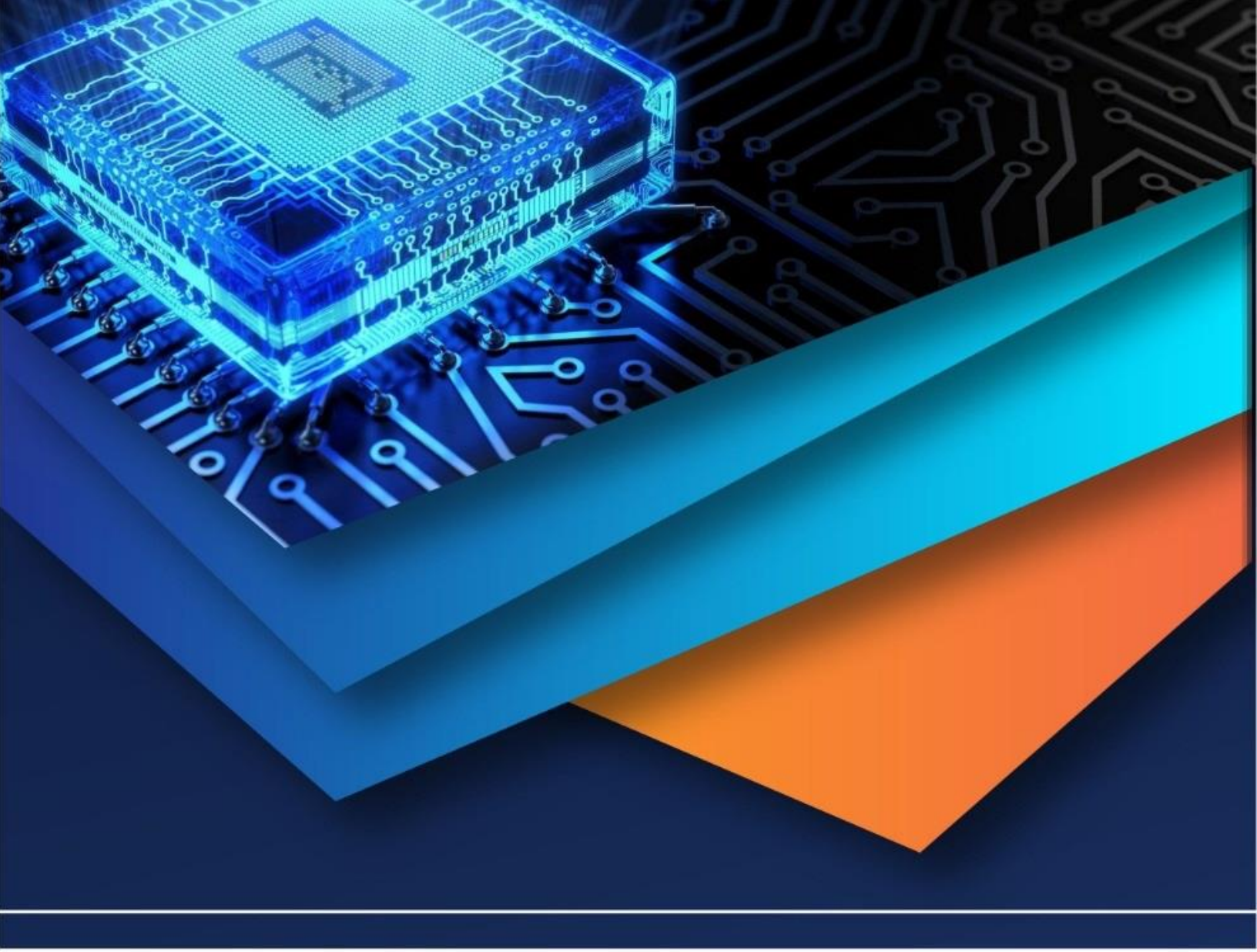
[6] L. Saravanan, D. Vijayanandh, J. R. Arunkumar, K. P. Chandran, and R. T. Prabhu, "A Novel Approach for a Smart Early Flood Detection and Awareness System using IoT," 2022 8th Int. Conf. on Smart Structures and Systems (ICSSS), Chennai, India, pp. 1–6, Apr. 2022. [Online].Available: https://ieeexplore.ieee.org/document/9782286/

[7] M. Khalaf, H. Alaskar, A. J. Hussain, T. Baker, Z. Maamar, R. Buyya, P. Liatsis, W. Khan, H. Tawfik, and D. Al-Jumeily, "IoT-Enabled Flood Severity Prediction via Ensemble Machine Learning Models," IEEE Access, vol. 8, pp. 70375–70386, Apr. 2020. [Online]. Available: https://ieeexplore.ieee.org/document/9057676/

[8] E. Deowan, S. Haque, J. Islam, Md. Hanjalayeamin, Md. T. Islam, and R. T. Meghla, "Smart Early Flood Monitoring System Using IoT," 2022 14th Seminar on Power Electronics and Control (SEPOC), pp. 1–6, Nov. 2022.[Online].Available: https://ieeexplore.ieee.org/document/9976434/

[9] G. Furquim, G. P. R. Filho, R. Jalali, G. Pessin, R. W. Pazzi, and J. Ueyama, "How to Improve Fault Tolerance in Disaster Predictions: A Case Study about Flash Floods Using IoT, ML and Real Data," Sensors (MDPI), vol. 18, no. 3, p. 907, Mar. 2018. [Online]. Available: https://www.mdpi.com/1424-8220/18/3/907

[10] K. Mosavi, P. Ozturk, and K. Chau, "Flood Prediction Using Machine Learning Models: Literature Review," Water (MDPI), vol. 10, no. 11, pp. 1536, 2018. [Online]. Available: https://www.mdpi.com/2073-4441/10/11/1536



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY