A MACHINE LEARNING AND IOT BASED SYSTEM TO PREDICT AND MONITOR FLOOD

Project report submitted
in partial fulfilment of the requirement for the degree of
Bachelor of Technology

By

Arif Ahmed (200610126014)
Avinash Gupta (200610026011)
Arif Ahmed Laskar (200610026006)
Bedatrayee Dey (200610026012)

Under the guidance
of

Dr. Navajit Saikia
Associate Professor
&
Ankur Jyoti Sarmah
Assistant Professor
Dept. of Electronics and Telecommunication Engineering

DEPARTMENT OF ELECTRONICS & TELECOMMUNICATION ENGINEERING
ASSAM ENGINEERING COLLEGE
JALUKBARI- 781013, GUWAHATI

June 2024

ASSAM ENGINEERING COLLEGE, GUWAHATI

CERTIFICATE

This is to certify that the thesis entitled "A Machine Learning and IoT Based System to Predict and Monitor Flood" submitted by Arif Ahmed (200610126014), Avinash Gupta (200610026020), Arif Ahmed Laskar (200610026006) and Bedatrayee Dey (200610026012) in the partial fulfillment of the requirements for the award of Bachelor of Technology degree in Electronics & Telecommunication Engineering at Assam Engineering College, Jalukbari, Guwahati, is an authentic work carried out by them under my supervision and guidance. To the best of my knowledge, the matter embodied in the thesis has not been submitted to any other University/Institute for the award of any Degree or Diploma.

Signature of Supervisor

Dr. Navajit Saikia

Signature of Co-Supervisor

Ankur Jyoti Sarmah

DECLARATION

We declare that this written submission represents our ideas in our own words and where others' ideas or words have been included, we have adequately cited and referenced the original sources. We also declare that we have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in our submission. We understand that any violation of the above will be a cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

	Name	**Roll Number**	**Signature**
1.	Arif Ahmed	200610126014	
2.	Avinash Gupta	200610026011	
3.	Arif Ahmed Laskar	200610026006	
4.	Bedatrayee Dey	200610026012	

ACKNOWLEDGEMENT

We would like to express our profound appreciation to Dr. Navajit Saikia, HOD, Electronics and Telecommunication Engineering Department, Assam Engineering College, for providing us with the opportunity to work on the project titled " A Machine Learning and IoT Based System to Predict and Monitor Flood ".

We would also like to express our sincere appreciation to our esteemed project guides,
Dr. Navajit Saikia, HoD in the Department of ETE and Ankur Jyoti Sarmah, Assistant Professor in the Department of ETE for their diligent supervision, valuable insights, and continuous encouragement. Their expertise, guidance, and unwavering support have been indispensable in navigating the intricacies of this project.

Their collective mentorship and support have been pivotal in enabling us to successfully execute and accomplish the goals of this project.

Arif Ahmed (200610126014)

Avinash Gupta (200610026011)

Arif Ahmed Laskar (200610026006)

Bedatrayee Dey (200610026012)

ABSTRACT

This project focuses on revolutionizing flood preparedness in the flood-prone districts of Assam, India, by harnessing the power of machine learning for accurate flood prediction. The region's vulnerability to annual floods poses significant challenges to the local communities, agriculture, and infrastructure. Traditional methods of flood prediction often fall short in providing timely and precise information, necessitating a more advanced and efficient approach.

The proposed solution leverages historical flood data, meteorological information, and geographical features to develop a robust machine learning model. Utilizing state-of-the-art algorithms, the model aims to predict flood occurrences with higher accuracy, enabling authorities and communities to implement proactive measures for minimizing the impact of floods. The project also involves the integration of real-time data streams to enhance the model's responsiveness and adaptability. The proposed solution is enhanced with IoT (Internet of Things) technology, integration of sensor networks for real-time data collection. An IoT system which senses the change in water level such as water level across flood-prone areas would enable continuous monitoring of the key environmental parameter. The sensor would transmit data wirelessly to a centralized system, providing up-to-date information on factors influencing flood occurrence.

The implementation will be carried out in collaboration with local authorities and stakeholders, ensuring that the solution is tailored to the specific needs of the Assam region. The project's success will not only contribute to the scientific understanding of flood prediction but also have a tangible impact on the lives of those residing in flood-prone areas. Ultimately, this endeavor aspires to establish a scalable and sustainable framework for flood prediction using machine learning, paving the way for improved disaster management strategies in vulnerable regions.

LIST OF FIGURES

Fig. No.	Fig. Title	Page No.
1.	Arduino Uno Board	13
2.	PIN Diagram of Arduino Uno Board	13
3.	WEMOS	16
4.	Pin diagram of WEMOS	17
5.	Action of an IoT sensor	21
6.	Working of an Ultrasonic Sensor	22
7.	NRL24L01 Module	23
8.	Working of NRF24L01	24
9.	LCD Display	25
10.	SIM800L	27
11.	Cloud Computing	29
12.	Cloud Storage System	31
13.	Components of IoT	32
14.	Windows App of Arduino	34
15.	IoT Setup	36
		37
16.	Residual analysis plot	41
17.	Graph of actual river level vs predicted river level	41
18.	Training dataset of rainfall level (in mm) vs river level (in m) datasets	42
19.	Testing dataset of rainfall level (in mm) vs river level (in m) datasets	42

CONTENTS *Page No.*

CERTIFICATE ii

CANDIDATE’S DECLARATION iii

ACKNOWLEDGEMENTS iv

ABSTRACT v

LIST OF FIGURES vi

CONTENTS vii

Chapter 1 INTRODUCTION

1.1 Introduction 1
1.2 Introduction to the area of work 1
1.3 Present day scenario 2
1.4 Motivation to do the project 2
1.5 Objectives of the Work 2
1.6 Target Specifications 3
1.7 Organization of the project report 3

Chapter 2 LITERATURE REVIEW

2.1 Introduction 4
2.2 Review of Relevant Research Papers 4

Chapter 3 METHODOLOGY

3.1 Introduction 7
3.2 Detailed Methodology 7
3.3 Assumptions made 11
3.4 Circuit layout / one-line / block diagrams 11
3.5 Component Specifications 12
3.6 Justification for component selection 38
3.7 Tools used 38
3.8 Preliminary result analysis 38

Chapter 4 RESULT ANALYSIS AND DISCUSSION 39

4.1 Introduction 39
4.2 Result Analysis 40

Chapter 5 CONCLUSION 46

CHAPTER 1

INTRODUCTION

1.1 Introduction

In the northeastern region of India lies Assam, a region of breathtaking beauty and perennial struggle against the mighty forces of nature. Here, the Brahmaputra River, a life-giving artery, transforms annually into a formidable adversary—the monsoon floods. While these floods bring fertility to the fertile plains, they also bear the weight of disruption, displacing lives, ravaging crops, and testing the resilience of communities. It is within this delicate dance between abundance and adversity that our project takes root—a machine learning-based prediction system designed to be the vanguard against the annual inundation in the flood-prone districts of Assam. As we stand at the intersection of technology and tradition, our mission is to create a predictive tool that transcends the limitations of existing models. Assam's geography, with its intricate river systems, undulating topography, and a unique monsoon pattern, demands a tailored approach. The goal is clear: to harness the power of Machine Learning and Internet of Things, not just to foresee the floods, but to empower communities with timely and accurate predictions, enabling them to mitigate the impact and adapt to the ebb and flow of nature.

1.2 Introduction to the area of work

Nestled in the northeastern corner of India, Assam stands as a region of immense natural beauty and cultural diversity. However, this enchanting landscape is marked by an annual struggle against the formidable forces of nature, particularly the recurrent menace of floods. Situated along the banks of the mighty Brahmaputra River, Assam's topography, climate, and hydrology converge to create a complex environment that is highly susceptible to inundation. This project seeks to bridge the gap between conventional flood prediction practices and the evolving landscape of technological innovation, with a specific focus on leveraging machine learning. By marrying the rich data tapestry of Assam's environmental dynamics with advanced computational algorithms, we aspire to usher in a new era of flood prediction that is not only more accurate but also more adaptive to the unique challenges posed by the region.

1.3 Present day scenario

The current scenario of flood prediction in Assam is characterized by conventional methodologies that often fall short in providing timely and accurate information. Historical data, meteorological inputs, and geographical features are considered, but the intricacies of the region's dynamic environmental conditions pose significant challenges to effective flood forecasting. As a result, communities, agriculture, and infrastructure remain vulnerable to the devastating impacts of floods, necessitating a paradigm shift in the approach to prediction and preparedness.

1.4 Motivation to do the project

Motivated by the pressing need for a more advanced and responsive flood prediction system, this project endeavors to harness the capabilities of machine learning. The goal is to develop a predictive model tailored to the unique characteristics of Assam's flood-prone districts, offering a more accurate and timely understanding of impending floods. By integrating cutting-edge technologies and collaborating with local authorities, this project aspires to not only enhance the scientific understanding of flood prediction but, more importantly, to make a tangible difference in the lives of those who grapple annually with the challenges imposed by nature.

1.5 Objectives of the work

The objective of the work are as follows:

1. **Develop a Machine Learning Model**: To create a robust machine learning model and to specify the machine learning algorithm for flood prediction.
2. **Utilize Historical Data:** Utilize the historical flood data to train and validate the machine learning model.
3. **Incorporate Real-Time Data Streams:** To describe the objective of integrating real time data streams into predictive model.
4. **Collaborate with Local Authorities:** Highlighting the goal of collaborating with local authorities and stakeholders in Assam.
5. **Enhance Predictive Accuracy:** To improve the accuracy of flood predictions in the flood-prone districts of Assam.
6. **Facilitate Early Warning Systems:** To develop a system that facilitates early warning and timely dissemination of flood predictions.

7. **Implement IoT System**: Develop a prototype of an IoT water level sensing device which can be placed along rivers, streams, and floodplains to continuously monitor water levels in real-time.

1.6 Target Specifications:

1. **Machine Learning Algorithms:** Implemented a hybrid model utilizing an ensemble of decision trees and a Long Short-Term Memory (LSTM) neural network for optimal predictive accuracy.
2. **Data Sources:** Utilized a comprehensive dataset comprising historical flood records, meteorological data (temperature, precipitation, humidity), river discharge measurements, and topographical information obtained from authoritative sources and local monitoring stations.
3. **Data Preprocessing Techniques:** Applied rigorous data cleaning procedures to address missing values and outliers and also normalized numerical features and encoded categorical variables appropriately for model compatibility.
4. **Real-Time Data Integration:** Developed a robust mechanism for integrating real-time data streams, ensuring the model is continuously updated with the latest meteorological and river discharge information.
5. **Model Deployment:** Deployed the trained model through a user-friendly web interface accessible to local authorities and communities.

1.7 Organization of the project report:

The report is organized in different chapters which are as follows: Chapter 1 gives a brief introduction about the area of work, present day scenario, motivation to do the project and target specifications. Chapter 2 discusses the literature review and information regarding the project area from various platforms and previous work. Chapter 3 discusses the research methodology adopted and tools used for the completion of the work. Chapter 4 contains the results obtained in the project work. Chapter 5 includes conclusion and future scopes.

CHAPTER 2

LITERATURE REVIEW

2.1 Introduction

In the pursuit of advancing flood prediction methodologies, this research endeavors to introduce a cutting-edge model entitled "A Machine Learning-Based Flood Prediction Model." The foundational step in this scientific odyssey is encapsulated within the comprehensive exploration of existing literature—a critical examination that not only contextualizes the project within the broader landscape of hydro informatics but also serves as the bedrock for our innovative contributions.

The literature review serves as an intellectual compass, charting a course through the intricate web of historical and contemporary flood prediction research. This section acknowledges and assimilates the wealth of knowledge amassed by predecessors, tracing the evolution of methodologies from traditional hydrological approaches to more contemporary empirical models. By navigating through the tributaries of past achievements and challenges, we derive insights that inform the current research trajectory.

Emphasizing a meticulous review of predictive analytics, statistical modeling, and the integration of machine learning into hydrological sciences, this section endeavors to distill key findings. Through a critical synthesis of existing literature, the narrative unfolds, delineating the prevailing trends, gaps, and opportunities within the field. Moreover, the literature review serves as a foundational platform to identify unexplored territories, propelling the discourse beyond established boundaries.

2.2 Review of Relevant Research Papers

1. Flood Prediction using Machine Learning by Asst. Prof. Anil Kumar Ambore1 , T. Sri Sai Charan2 , U. Rohit Reddy3 , T. Samara Simha Reddy4 , Tarun.G5 1, 2, 3, 4, 5School of Computer science &Engineering REVA University Bengaluru, India

Conclusion:

i. *The current state of machine learning (ML) modelling for flood prediction is in the early stages of advancement, with the paper providing an overview of ML models used in flood prediction and developing a classification scheme to analyse the existing literature.*

ii. *The survey represents the performance analysis and investigation of more than 6000 articles, identifying 180 original and influential articles where the performance and accuracy of at least two machine learning models were compared.*

iii. *The paper classifies prediction models into two categories based on lead time and further divides them into categories of hybrid and single methods, discussing the state of the art of these classes in detail.*

iv. *The performance of the ML methods was evaluated in terms of R2 and RMSE, in addition to generalization ability, robustness, computation cost, and speed, with significant research and experimentation for further improvement and advancement.*

v. *The document highlights four major trends reported in the literature for improving the quality of flood prediction: novel hybridization, data decomposition techniques, the use of ensembles of methods, and add-on optimizer algorithms to improve the quality of ML algorithms.*

vi. *The future horizon of flood prediction is expected to witness significant improvements for both short-term and long-term predictions through the proper usage of soft computing techniques in designing novel learning algorithms.*

2. Hou, J., Zhou, N., Chen, G., Huang, M., & Bai, G. (2021). Rapid forecasting of urban flood inundation using multiple machine learning models.
Conclusion:
The study aims to construct a rapid forecasting model for urban flood inundation using a combination of a hydrodynamic model and machine learning (ML) algorithms. The research focuses on the Xixian New Area in China and aims to provide accurate predictions and sufficient lead time for emergency decision-making. The study verifies the reliability of the hydrodynamic model and demonstrates the effectiveness of the ML algorithms in predicting urban flood inundation. The multi-model approach, which combines the RF and KNN algorithms, is found to further reduce forecast errors and enhance the reliability of forecast results. The study concludes that the proposed method meets the requirements of rapid forecasting for urban flood inundation and provides decision support for emergency decision-making. Future work is planned to improve the model to reflect the inundation process instead of the maximum inundation time.

3. Mosavi, A., Ozturk, P., & Chau, K. (2018). Flood Prediction Using Machine Learning Models
The document classifies prediction models into two categories based on lead time and further divides

them into hybrid and single methods. It discusses the performance comparison of these methods in detail, considering factors such as R2 and RMSE, generalization ability, robustness, computation cost, and speed. The paper also identifies four major trends for improving the quality of prediction: novel hybridization, data decomposition techniques, ensemble methods, and add-on optimizer algorithms.

Furthermore, the conclusion emphasizes the importance of soft computing techniques in designing novel learning algorithms and the multidisciplinary nature of the work. It suggests that future work should include a survey on spatial flood prediction using ML models and encourages the investigation of recent advancements in ML models for spatial flood analysis.

In summary, the conclusion of the document provides a comprehensive overview of the current state of ML modeling for flood prediction, identifies key trends for improving prediction quality, and suggests areas for future research and development in the field of flood prediction using machine learning models.

CHAPTER 3

METHODOLOGY

3.1. Introduction

The methodology section aims to detail the process employed for flood prediction in Silchar using machine learning techniques. It covers the steps involved in data processing, model development, and prediction methodology.

3.2. Detailed Methodology

The methodology adopted in this project involved the following steps:

3.2.1. Data Collection

Hourly rainfall and river water level data were gathered from their respective sources. This process involves retrieving raw data from authoritative sources or databases that record hourly measurements of rainfall and river water levels in the Silchar region. These datasets serve as the foundation for subsequent analysis and model training.

2.Data Pre-processing: The collected datasets underwent several pre-processing steps:

Merging Datasets: The rainfall and river water level datasets were combined based on their timestamps. This alignment of data allows for correlation between rainfall and river level observations at specific times.

Handling Missing Values: Any missing or null values in the datasets were addressed. This could involve imputation techniques, such as filling missing values with zeros or using interpolation methods to estimate missing data points.

Timestamp Conversion: Timestamps in the dataset were converted to a consistent and appropriate format. This ensures uniformity and ease of handling time-related data.

3.Feature Engineering: Additional features were derived from the timestamp to enrich the dataset: Hour, Day, Month, Year Extraction: Extracting these temporal components from the timestamp enables the model to understand temporal patterns and seasonality. For instance, understanding the impact of the time of day, day of the month, or seasonal variations on rainfall and river water levels.

4.Threshold Limit Setting: A threshold limit was established based on river water levels. This threshold serves as a classification boundary to distinguish between normal river levels and potential flood situations. It's crucial for defining the conditions under which the prediction model will flag an occurrence as a flood risk based on observed river water levels.

5.Model Development: Various machine learning algorithms including Support Vector Machine (SVM), K-Nearest Neighbors (KNN), XGBoost, and Random Forest were explored for their efficacy in predicting flood occurrences and river levels. Through rigorous experimentation and evaluation, it was determined that Random Forest yielded the least amount of error and demonstrated higher accuracy compared to the other algorithms. Therefore, the Random Forest Regressor was selected as the primary machine learning algorithm for the predictive model.

The Random Forest Regressor is a powerful machine learning algorithm that's particularly well-suited for regression tasks, such as predicting continuous values like river levels based on input variables like rainfall. Here's a detailed breakdown:

Random Forest Regressor:

- Ensemble Learning: The algorithm falls under the category of ensemble learning, where it combines multiple individual models to create a more robust and accurate predictive model. In the case of Random Forest, it constructs an ensemble of decision trees.

- Decision Trees: Decision trees are flowchart-like structures where each internal node represents a test on an attribute, each branch represents an outcome of the test, and each leaf node represents a target value. Individual decision trees are prone to overfitting the data, capturing noise rather than true patterns.

- Bagging Technique: Random Forest applies a technique called bagging (bootstrap aggregating) where multiple decision trees are built using bootstrapped datasets (random subsets of the original data). Each tree is trained independently on different subsets of the data, which helps in reducing overfitting and increasing model accuracy.

- Random Feature Selection: During the construction of each tree in the forest, only a random subset of features (input variables) is considered at each split point. This randomness further diversifies the trees, ensuring that they are less correlated and improving the overall model's predictive power.

- Prediction: For regression tasks, the Random Forest Regressor aggregates predictions from all the individual trees in the forest to make a final prediction. It averages the predictions in the case of regression, providing a continuous output.

- Handling Complexity: Random Forest models are effective in handling complex relationships between input variables and the target variable. They can handle large datasets with many input features and are relatively less sensitive to hyperparameters compared to some other models, making them easier to tune.

Applicability to Flood Prediction: For predicting river levels based on rainfall data: The Random Forest Regressor is adept at capturing intricate relationships between various meteorological factors (like rainfall) and the continuous output of river levels. Its ability to handle complex, non-linear relationships between multiple input variables and the target variable makes it a suitable choice for predicting river levels, which often exhibit dynamic and non-linear behaviour.

By employing the Random Forest Regressor, this project leverages its capabilities to develop a predictive model that can effectively learn from the relationships between rainfall patterns and subsequent variations in river water levels in the Silchar region.

6. Model Evaluation:

Model evaluation is crucial to assess how well the developed predictive model performs in making accurate and reliable predictions. Here's an in-depth look at the performance metrics used for evaluating the flood prediction model:

Performance Metrics Calculation:

- **Mean Squared Error (MSE):** MSE measures the average squared difference between the predicted values and the actual observed values in the dataset. A lower MSE indicates that the model's predictions are closer to the actual values, signifying better accuracy. In the context of flood prediction, a low MSE suggests that the model's predictions for river levels based on rainfall data closely match the observed river levels.

- **Coefficient of Determination (R^2):** R^2 represents the proportion of variance in the dependent variable (river levels) that is explained by the independent variables (rainfall, time features, etc.). R^2 ranges from 0 to 1, where 1 indicates that the model perfectly predicts the target variable, and 0 indicates that the model does not explain any variability. A higher R^2 value signifies that a larger proportion of variance in river levels is accounted for by the model's input variables, demonstrating the model's explanatory power.

- **Feature Importance:** Feature importance identify the relative importance of input variables (rainfall, hour, day, month, etc.) in making predictions. Higher feature importance indicate more significant contributions of respective variables in predicting river levels. Identifying key input variables guiding the prediction helps in understanding which meteorological factors (such as rainfall at specific times or seasonal variations) have a more substantial impact on river levels. This insight aids in refining the model and selecting crucial features for better predictions.

Significance in Flood Prediction:

- Accuracy Assessment: MSE helps in quantifying how accurate the model's predictions are by measuring the error between predicted and observed values of river levels.
- Explanatory Power: R^2 provides insight into how well the chosen input variables explain variations in river levels, highlighting the model's ability to capture and explain the variability in data.
- Feature Importance: Understanding feature importance aids in refining the model by focusing on the most influential factors affecting river levels, guiding further model improvement and feature selection. By evaluating the flood prediction model using these metrics, it becomes possible to gauge its accuracy, explanatory power, and the significance of different input variables in predicting river levels based on rainfall and temporal factors. This evaluation process is crucial for refining the model and improving its predictive capabilities for real-world flood prediction scenarios in Silchar.

7. Prediction for End-Users:

A user-friendly functionality was created to enable end-users to predict river levels based on provided rainfall amount and date/time inputs. This empowers stakeholders or end-users to forecast potential river levels, aiding in decision-making related to flood preparedness and response.

Each of these steps is pivotal in constructing a robust flood prediction model, ensuring accurate predictions and usability in real-world scenarios.

8. IoT system designing and development:

Identifying and selecting IoT sensors capable of measuring critical environmental parameters relevant to flood prediction, such as water level, rainfall, and river flow rates.

A data acquisition system is then established to collect real-time data from deployed IoT sensors. Implement reliable communication protocols for wireless transmission of sensor readings to a centralized database. A GSM module is integrated with the IoT water level sensing system to enable the transmission of alert signals in case of abnormal or rising water levels. The system is configured to send alert messages to local authorities and residents via SMS or other communication channels.

3.3 Assumptions Made:

- The relationship between rainfall and river level is assumed to be predictive within the specified threshold.
- The hourly data accurately represents the fluctuations in both rainfall and river water levels.

3.4 Circuit layout/ One line / Block diagrams:

No physical circuitry or electrical components were used in this machine learning-based flood prediction model. However, a conceptual flowchart demonstrating data flow and processing steps can be presented.

3.5 Component Specifications:

~ Machine Learning

A. Random Forest Regressor: Utilized with 100 estimators and default settings for regression analysis.

B. Python Libraries: Pandas, NumPy, Matplotlib for data manipulation, analysis, and visualization.

C. Sklearn Library: Employed for machine learning algorithms and performance evaluation.

~Internet of Things

A. <u>MICROCONTROLLER BOARDS:</u>

Here, we are dealing with two types of microcontroller boards. These are:

i. ARDUINO UNO BOARD
ii. WEMOS

i. ARDUINO UNO BOARD:

One type of single-chip microcontroller created by Atmel and included in the megaAVRfamilyis the ATmega328. The Arduino Uno's architecture is a modified version of theHarvard architecture with an 8-bit RISC processing core.

Arduino Pro Mini, Arduino Nano, and more boards are also available for the Arduino Uno.

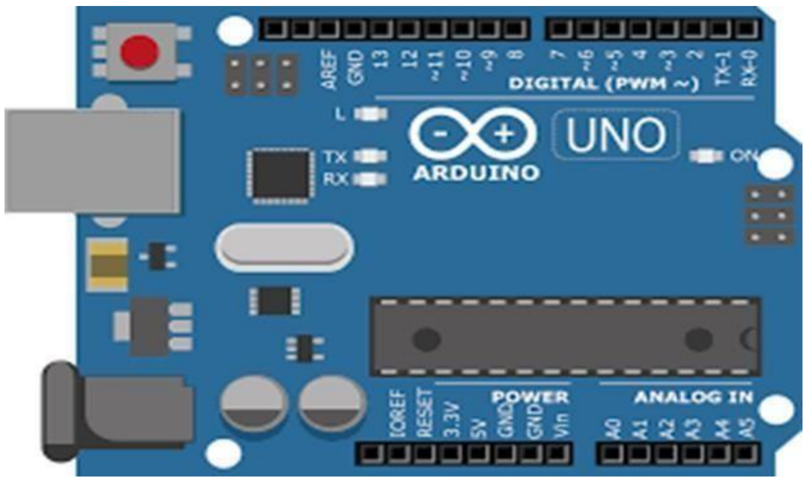


Fig: Arduino Uno Board

➢ POWER SUPPLY:

We may power the Arduino Uno with the aid of a USB connection or an external power source. The majority of external power supply are batteries or AC to DC adapters. By inserting the adapter into the Arduino Uno's power jack,the Arduino board can be linked. The Vin pin and the GND pin of the POWER connection can both be linked to the battery leads in a similar manner. 7 to 12 volts will be the recommended voltage range.

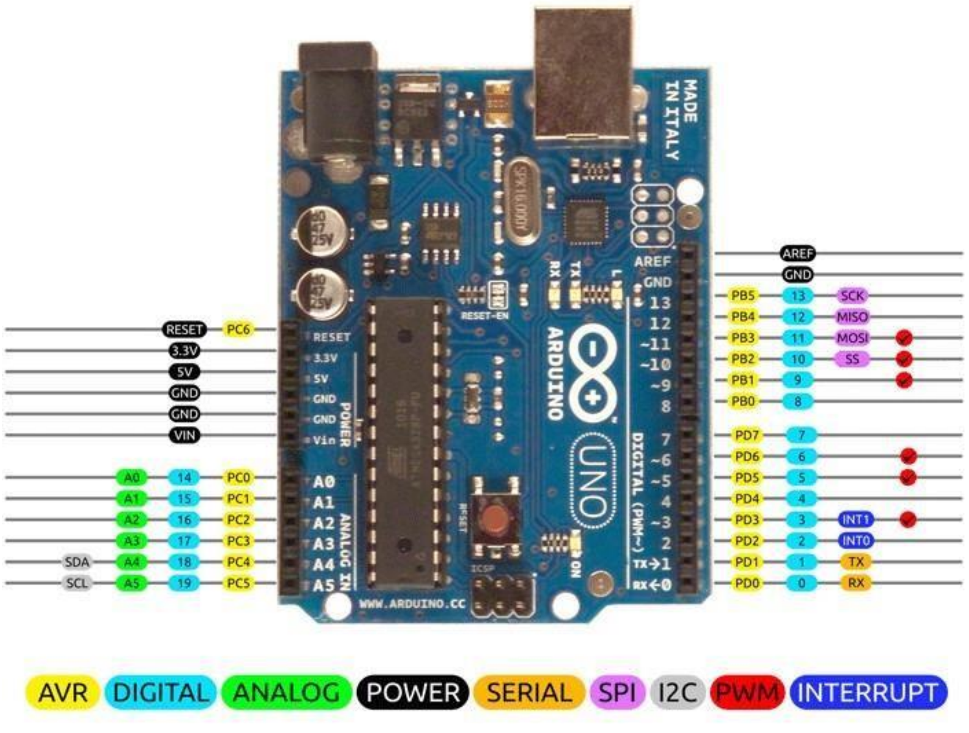


Fig: PIN Diagram of Arduino Uno Board

➢ INPUT & OUTPUT:

The functions pin Mode (), digital Write (), and Digital Read () allow the ArduinoUno's 14digital pins to be used as input and output ().

a. **Pin1 (TX) & Pin0 (RX) (Serial):** These pins are connected to the ATmega8U2 USB toTTL Serial chip equivalent pins and are used to transmit & receive TTLserial data.

b. **Pins 2 and 3 (External Interrupts):** External pins can be linked together totrigger an interrupt in response to a low value or value change.

c. **Pins 3, 5, 6, 9, 10, and 11 (PWM):** This pin does an 8-bit PWM o/p using theanalogue Write function ().

d. **SPI Pins (SS, MOSI, MISO, and SCK):** These pins sustain SPI-communication, despite the fact that it is provided by the fundamental hardware, which is not yetsupported by the Arduino language.

e. **Pin-13 (LED):** Pin-13 can be connected to the built-in LED (digital pin). When thepin is LOW, the light-emitting diode is turned on as if it were the HIGH-value pin.

f. **Pins 4 (SDA) and 5 (SCL) I2C:** The Wire library is used to support TWI communication.

g. **AREF (Reference Voltage):** The analogue inputs with analogue reference havea reference voltage ().

h. **Reset Pin:** The microcontroller is reset (RST) via this pin.

➢ MEMORY:

This Atmega328 Arduino microcontroller has 32 KB of flash memory for code storage,2 KBof SRAM, and 1 KB of EEPROM.

➢ COMMUNICATION:

UART TTL-serial communication is provided by the Arduino Uno ATmega328 and is available on digital pins TX (1) and RX (0). An Arduino's software has a serial monitor that makes it simple to input data. When data is being transmitted through the USB, twoLEDs onthe board labelled RX and TX will blink.

➢ FEATURES:

- The operating voltage is 5V
- The recommended input voltage will range from 7v to 12V
- The input voltage ranges from 6v to 20V
- Digital input/output pins are 14
- Analog i/p pins are 6
- DC Current for each input/output pin is 40 mA
- DC Current for 3.3V Pin is 50 mA
- Flash Memory is 32 KB
- SRAM is 2 KB
- EEPROM is 1 KB
- CLK Speed is 16 MHz

ii. WEMOS

The WEMOS microcontroller is a versatile and compact development board based on the popular ESP8266 and ESP32 microcontrollers. It is designed to provide an easy-to-use platform for IoT (Internet of Things) projects, prototyping, and educational purposes. The WEMOS boards come in variousmodels, offering different features and capabilities to suit a wide range of applications. With built-in Wi-Fi connectivity and a wealth of GPIO pins, theWEMOS microcontroller empowers developers to create connected devices and wireless sensor networks with ease.

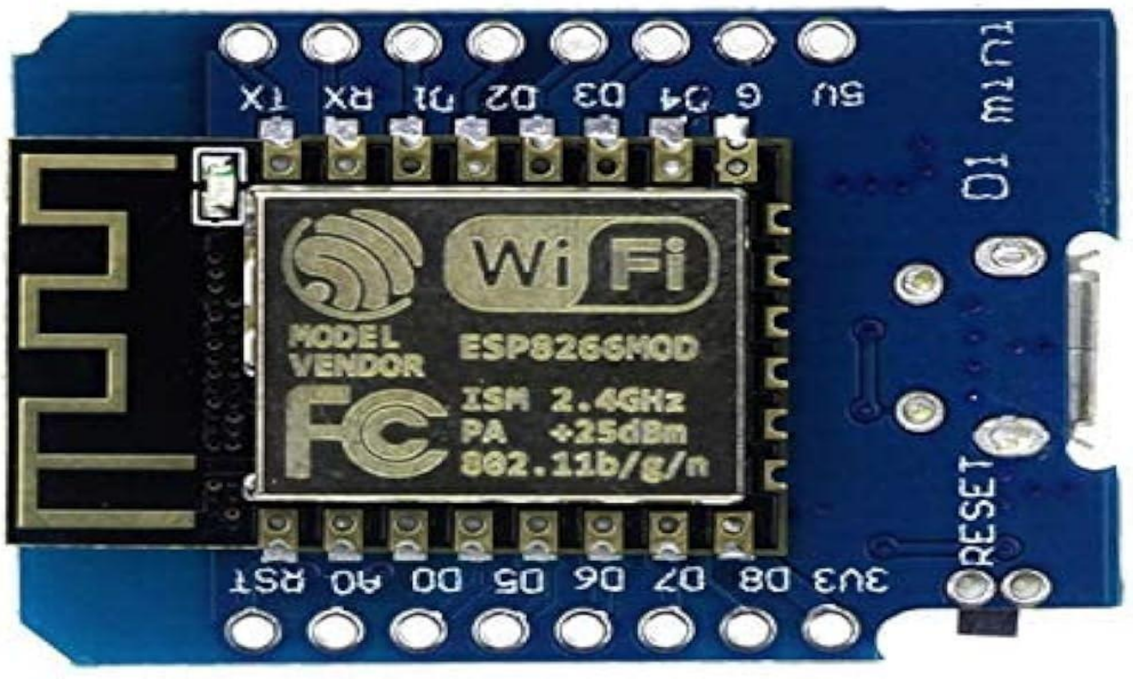


Fig: WEMOS

• POWER SUPPLY:

The WEMOS microcontroller can be powered through a USB connection or an external power source. It typically operates at 3.3V logic levels, but some models may have built-in voltage regulators to support higher voltage inputs. Additionally,the WEMOS boards often feature onboard LiPo battery management, allowing for portable and battery-powered applications.

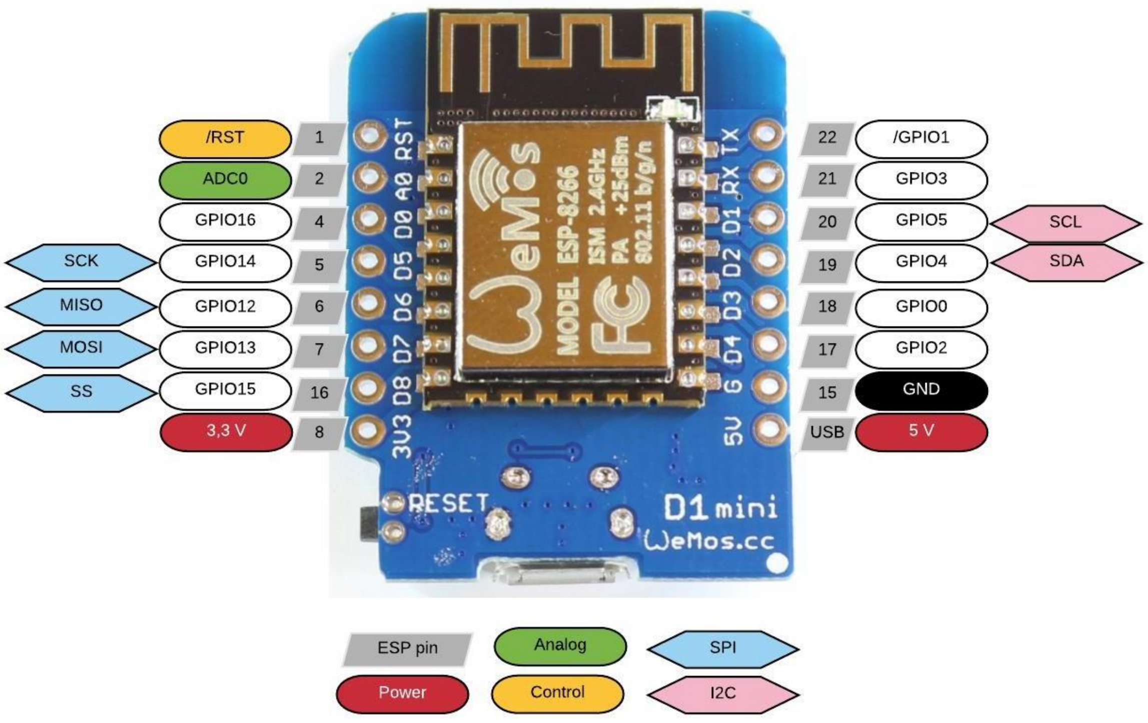


Fig: Pin diagram of WEMOS

• INPUT & OUTPUT:

WEMOS microcontrollers feature a variety of input and output options, including digital GPIO pins, analog input pins, PWM (Pulse Width Modulation) pins, I2C, SPI, UART, and more. These pins allow users to interface with sensors, actuators, displays, and other external devices, making the WEMOS boards suitable for a widerange of projects.

a. **Digital GPIO Pins**: These pins can be configured as digital inputs or outputs. As inputs, they can read digital signals (high or low) from externaldevices such as sensors, switches, or buttons. As outputs, they can drive digital signals to control LEDs, relays, or other digital devices.

b. **Analog Input Pins**: Analog input pins allow the microcontroller to read analog voltage levels from sensors or other analog devices. They typically support 10-bit ADC (Analog-to-Digital Converter) resolution, allowing for analog voltage readings between 0 and 3.3V (or the operating voltage of the microcontroller).

c. **PWM (Pulse Width Modulation) Pins**: PWM pins generate digital pulses with varying pulse widths, allowing for control of analog-like devices such asmotors, LEDs, or servos. By adjusting the duty cycle of the PWM signal, the average voltage or power delivered to the device can be controlled.

d. **I2C (Inter-Integrated Circuit) Pins**: WEMOS microcontrollers often feature dedicated I2C pins (SDA and SCL) for communication with I2C- compatible devices such as sensors, displays, or other microcontrollers. I2Cis a serial communication protocol that allows multiple devices to communicate over a shared bus.

e. **SPI (Serial Peripheral Interface) Pins**: SPI pins (MISO, MOSI, SCK) facilitate high-speed serial communication between the microcontroller and SPI-compatible devices such as sensors, SD cards, or other microcontrollers.SPI is commonly used for data transfer between devices with high data ratesand low latency.

f. **UART (Universal Asynchronous Receiver-Transmitter) Pins**: UART pins(TX and RX) provide serial communication for interfacing with devices suchas GPS modules, Bluetooth modules, or other microcontrollers. UART communication uses asynchronous serial transmission for data exchange.

g. **Other Special Function Pins**: Some WEMOS boards may feature additionalspecial function pins for specific purposes, such as boot mode selection, reset, or external interrupt inputs.

- MEMORY:

The memory configuration of WEMOS microcontrollers varies depending on the specific model. They typically feature built-in flash memory for program storage andRAM for runtime data storage. The ESP8266-based WEMOS boards usually have less memory compared to ESP32-based boards, but both offer sufficient resources for most IoT applications.

- **COMMUNICATION:**

One of the key features of WEMOS microcontrollers is their built-in Wi-Fi connectivity, which enables wireless communication with other devices and networks. They support standard Wi-Fi protocols such as 802.11b/g/n, allowing forseamless integration into existing Wi-Fi networks. Additionally, WEMOS boards may support other communication interfaces such as Bluetooth, MQTT, HTTP, andmore, facilitating data exchange with cloud services and IoT platforms.

- **FEATURES:**

- Compact and lightweight design.
- Built-in Wi-Fi connectivity for wireless communication
- GPIO pins for interfacing with external devices
- Support for various communication protocols (SPI, I2C, UART, etc.)
- USB interface for programming and power supply
- Onboard voltage regulators and battery management (some models)
- Compatible with Arduino IDE and other development environments
- Open-source hardware and software, fostering a vibrant community ofdevelopers and enthusiasts.

B. <u>SENSOR</u>

A sensor is a device that detects input of any kind from the physical world and reactsto it. Light, heat, motion, moisture, pressure, and a variety of other environmental phenomena can all be inputs. In order to detect the presence of a certain physical quantity, it is then converted into a signal that can be processed (e.g., electrically, mechanically, or optically). The sensor’s output comes in human-readable form.

In most IOT device architectures, sensors are used. They are essential to the internetof things (IoT). They make it feasible to develop an ecosystem for gathering information about a particular environment and processing it so that it may be monitored, managed, and controlled more effectively. IoT sensors are employed in avariety of situations, including homes, the outdoors, cars, airplanes, industrial settings, and others.

➢ SENSOR CHARACTERISTICS:

Mainly, there are two types of characteristics

i) Static:

This refers to how a sensor's output alters in response to an input change after reachingsteadystate.

- Accuracy, Range, Resolution, Precision, Sensitivity, Sensitivity,Linearity, Drift, Repeatability.

ii) Dynamic:

These are the properties of the systems.

- Zero-order system: The output shows a response to the input signal with no delay.
- First-order system: When the output approaches its final value gradually.
- Second-order system: The output response of the sensor oscillates before steadystate.

➢ **Types of Sensors:**

Sensors can be categorized in multiple ways

i) Passive & Active:

Passive sensors are unable to sense input on their own. For instance, sensors for soilmoisture, water level, and temperature.

Active sensors, however, can sense the input on their own. Such sensorsinclude radar, sounder, and laser altimeter ones.

ii) Analog & digital:

With analogue sensors, the sensor's response or output is a continuous function ofone or more of its input parameters. Examples include a temperature sensor, an LDR, an analogue pressure sensor, and an analogue hall effect.

Responses from digital sensors are binary in nature. For instance, a digitaltemperature sensor and a passive infrared (PIR) sensor (DS1620).

iii) Scalar & vector:

A scalar sensor solely considers an input parameter's magnitude while detecting it. For instance, a sensor for temperature, gas, strain, colour, orsmoke.

The magnitude of the direction and orientation of the input parameter affectsthe vector sensor.

For instance, sensors with an accelerometer, gyroscope, magnetic field, and motiondetector.

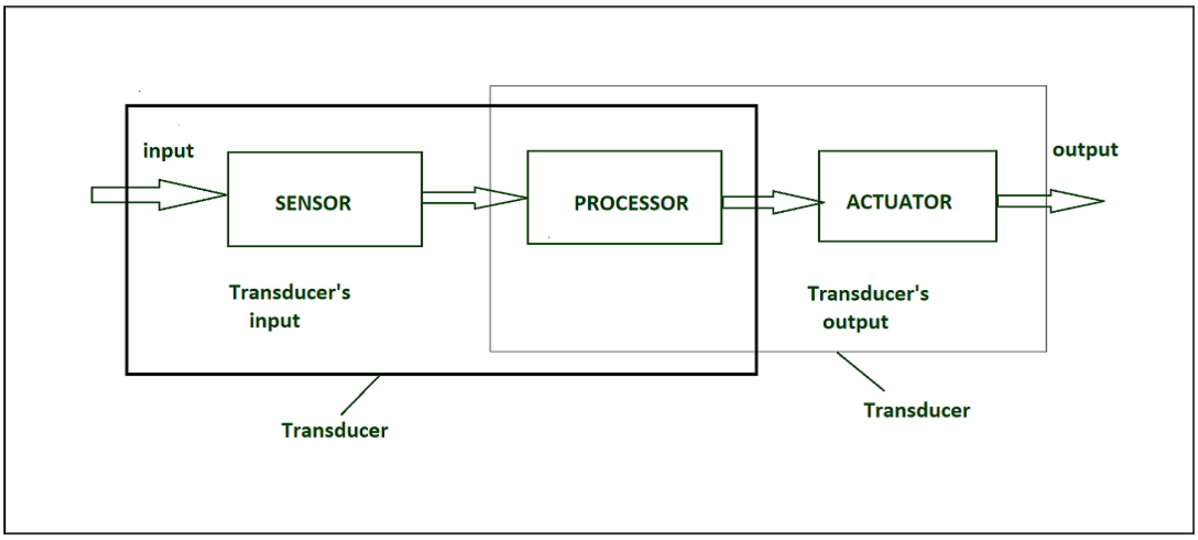


Fig: Action of an IoT sensor

➢ DESCRIPTION OF ULTRASONIC SENSOR:

An ultrasonic sensor is a piece of technology that uses ultrasonic sound waves to measure a target object's distance and then turns the sound that is reflected back intoan electrical signal. The speed of audible sound is greater than the speed of ultrasonicwaves (i.e., the sound that humans can hear).

There are two basic parts to this sensor.

a) **The transmitter:** This device uses piezoelectric crystals to emit sound.

b) **The receiver:** This is where the sound is heard once it has reached and leftthe target.D = 1/2T x C

Where, D = Distance, T = Time, and c = Sound Speed~ 343

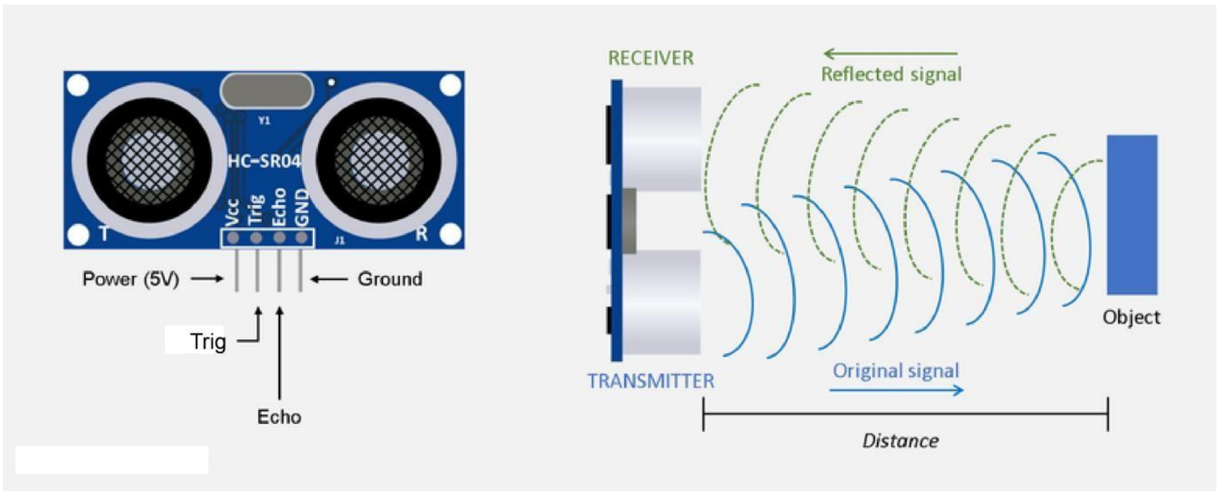


Fig: Working of an Ultrasonic Sensor

C. NRF24L01 MODULE

The NRF24L01 is a popular 2.4GHz RF transceiver module widely used in wireless communication applications. Developed by Nordic Semiconductor, it offers a cost- effective solution for establishing short-range wireless connections between devices. The module is known for its low power consumption, compact size, and ease of integration, making it ideal for various applications including remote control systems, sensor networks, Internet of Things (IoT) devices, and wireless peripherals.

➢ Characteristics of NRF24L01 Module:

- **2.4GHz RF Communication:** The NRF24L01 operates in the 2.4GHz ISM (Industrial, Scientific, and Medical) band, providing robust and interference-resistant communication.

- **Multiple Channels:** It supports multiple channels for communication, allowing multiple NRF24L01 modules to operate simultaneously in the samevicinity without interference.
- **Low Power Consumption:** The module is designed for low power operation, making it suitable for battery-powered devices and applicationsrequiring energy efficiency.
- **SPI Interface**: Communication with the NRF24L01 module is facilitatedthrough a Serial Peripheral Interface (SPI), enabling easy interfacing withmicrocontrollers and other digital devices.
- **Adjustable Output Power:** The output power of the NRF24L01 module isadjustable, allowing users to optimize the communication range and powerconsumption according to their specific requirements.
- **Build in CRC:** The module incorporates a built-in CRC (Cyclic RedundancyCheck) feature for error detection, ensuring reliable data transmission over the wireless link.

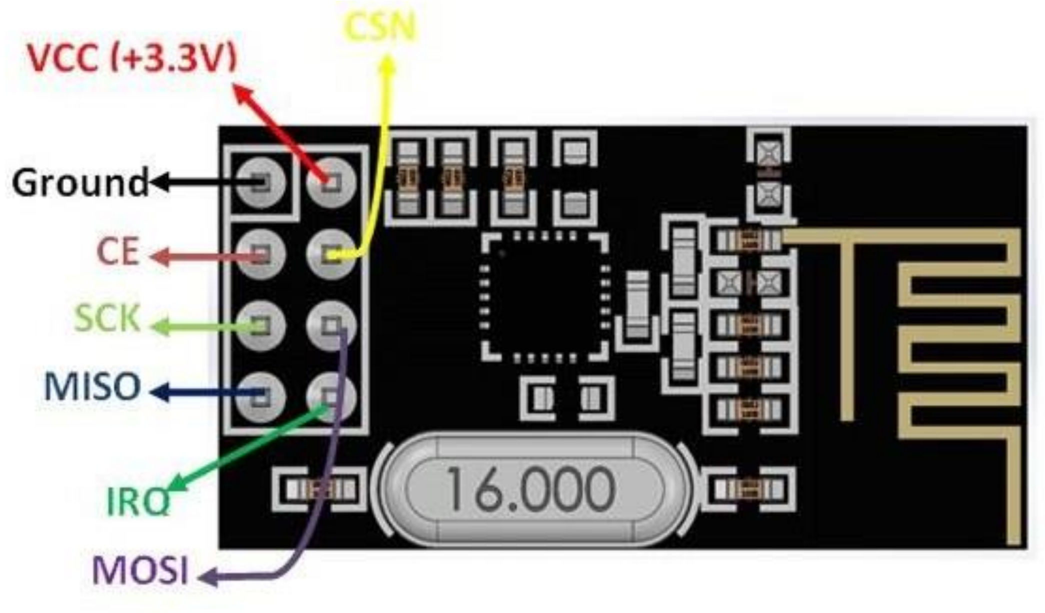


Fig: NRL24L01 Module

➢ Working of NRF24L01 Module:

The NRF24L01 module operates in two primary modes: Transmit (TX) mode andReceive (RX) mode.

1. **Tansmit Mode (TX):** In this mode, the NRF24L01 module sends data packets to another

NRF24L01 module or receiver device. The user specifies the data to be transmitted and configures the necessary parameters such as channel frequency, data rate, and output power. The module then encodes the data into packets and transmits them over the air using the specified parameters.

2. **Receive Mode (RX):** In RX mode, the NRF24L01 module listens for incoming data packets on a specified channel. When a valid packet is received, the moduledecodes the packet and makes the received data available to the host microcontroller or device for further processing.

The communication between NRF24L01 modules follows a master-slave architecture, where one module acts as the transmitter (master) and the other as the receiver (slave). The modules communicate using a simple protocol, typicallyconsisting of a preamble, address field, payload, and optional CRC field.

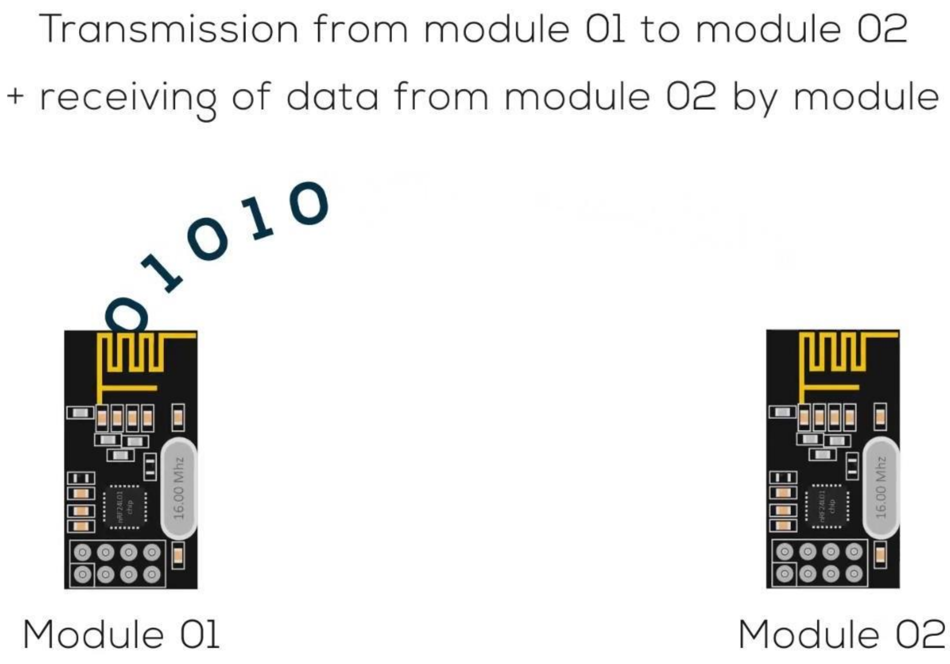

Transmission from module 01 to module 02
+ receiving of data from module 02 by module
01010
16.00 Mhz
16.00 Mhz
Module 01
Module 02

Fig: Working of NRF24L01

D. <u>16×2 LCD DISPLAY</u>

The 16x2 LCD (Liquid Crystal Display) module is a commonly used alphanumeric display that consists of 16 columns and 2 rows of characters, allowing for the display of 32 characters at a time. It serves as a simple and effective output interface in various electronic projects and devices. The 16x2 LCD display modules are widely utilized in embedded systems, microcontroller-based projects, appliances, instrumentation, and educational kits due to their easeof use, readability, and affordability.

➢ CHARACTERISTICS OF 16*2 LCD DISPLAY:

1. **Alphanumeric Display:** The 16x2 LCD display can show alphanumericcharacters, symbols, and custom patterns.
2. **16*2 Character Grid:** It features 16 columns and 2 rows, providing atotal of 32 character spaces for displaying text.
3. **Backlight Option:** Many 16x2 LCD displays come with an optionalbacklight, enhancing visibility in low-light conditions.
4. **Low Power Consumption:** LCD displays typically consume low power,making them suitable for battery-operated devices and energy-efficient applications.
5. **Parallel Interface:** Communication with the 16x2 LCD module is typically done through a parallel interface, utilizing a minimum of 4 datalines (usually 8) plus control lines for operation.
6. **Easy Interfacing:** Interfacing with microcontrollers and other digital devices is straightforward, often requiring minimal external components.

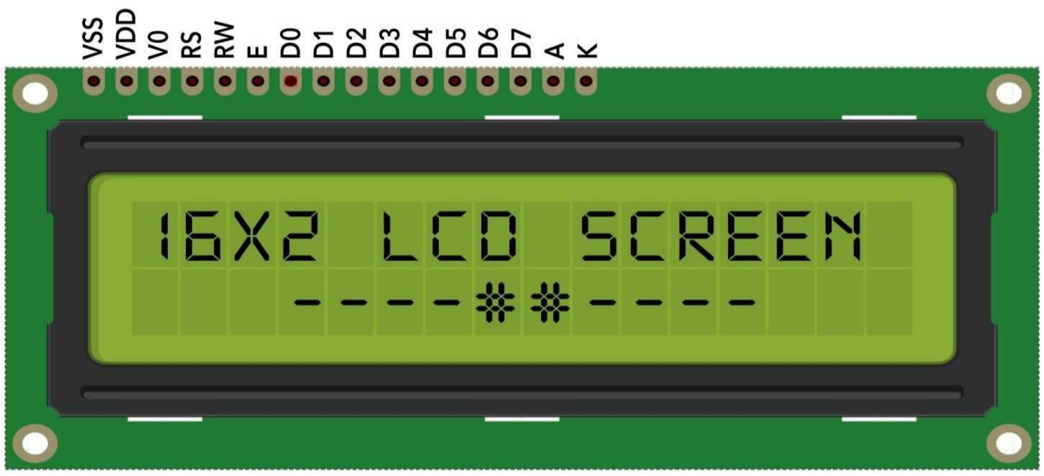


Fig: LCD Display

➢ Working of 16×2 LCD DISPLAY:

The operation of a 16x2 LCD display involves two main components: the displaymodule itself and a controlling device, such as a microcontroller.

a. **Initializing the Display:** Before using the LCD display, it needs to be initialized.This usually involves configuring parameters such as display mode, cursor settings, and any custom characters.
b. **Sending Data and Commands**: Data and commands are sent to the LCD module through its parallel interface. Commands configure the display settings (e.g., clear display, set cursor position), while data represents the characters to bedisplayed.
c. **Displaying Characters:** Once initialized, the LCD display can show charactersby receiving data from the controlling device. Characters are displayed in the specified row and column positions on the screen.
d. **Updating Display Content:** The display content can be dynamically updated bysending new data to the LCD module. This allows for real-time display of information such as sensor readings, status messages, or user input.

E. SIM800L

The SIM800L is a compact and versatile GSM/GPRS module widely used for wireless communication in embedded systems and IoT devices. Developed by SIMCom, it provides reliable and low-cost connectivity for transmitting data, making voice calls, andsending SMS messages over the GSM network. The module is popular among hobbyists,developers, and engineers for its small form factor, ease of integration, and robust performance.

➢ POWER SUPPLY:

The SIM800L module typically operates within a voltage range of 3.4V to 4.4V DC. It requires a stable power supply capable of providing sufficient current during peak transmission operations, typically around 2A. The module can be powered using a regulated DC power supply, battery, or voltage regulator. Additionally, it features a low-power sleep mode to conserve energy when not in use.

➢ INPUT AND OUTPUT:

- Serial Interface: The SIM800L communicates with the host microcontroller or device through a serial UART interface, supporting baud rates from 1200bps to115200bps.
- GPIO Pins: The module features several General-Purpose Input/Output (GPIO)pins for interfacing with external components or sensors, enabling additional functionality such as controlling LEDs, relays, or sensors.

➢ MEMORY:

The SIM800L module includes onboard memory for storing configuration settings, SMS messages, phonebook entries, and received data. It typically has a built-in SIM card holder for storing the subscriber identity module (SIM) card, which contains subscriber information and network authentication data.

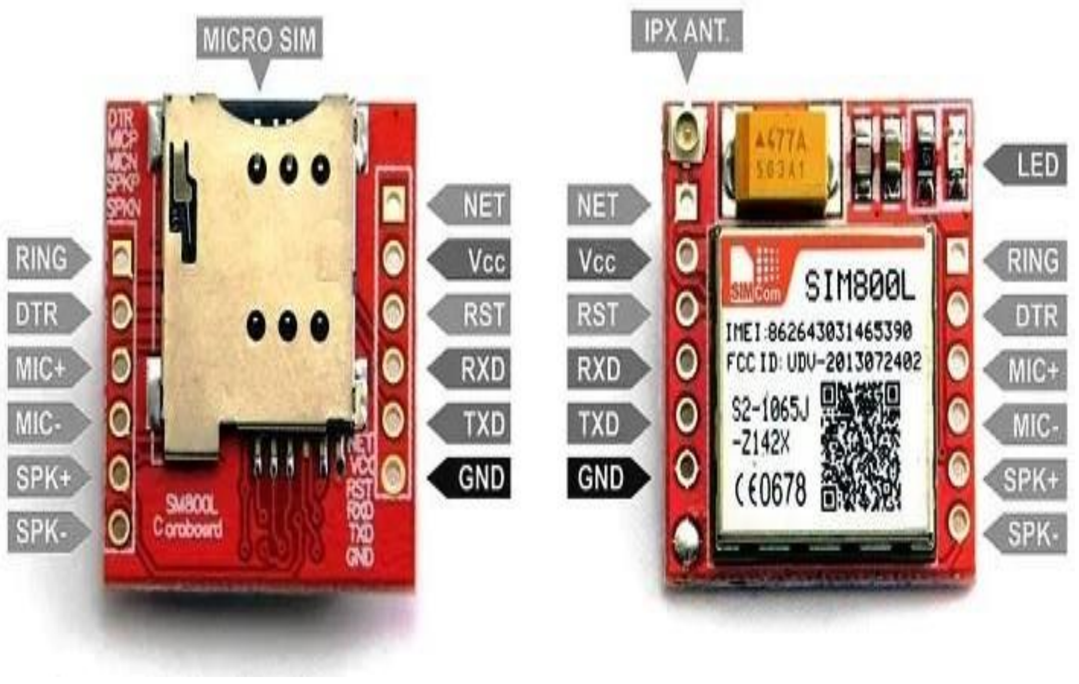


Fig: SIM800L

➢ COMMUNICATION:

The SIM800L module supports GSM (2G) and GPRS (General Packet Radio Service) communication standards, allowing it to connect to mobile networks and transmit data packets. It features a built-in TCP/IP stack for Internet connectivity and supports variouscommunication protocols such as HTTP, FTP, SMTP, and MQTT. The module can establish data connections, make voice calls, send SMS messages, and perform network-related operations.

➢ FEATURES:

- **Quad-Band Operation:** The SIM800L supports quad-band operation, allowing it to work on GSM/GPRS networks in frequencies ranging from 850MHz to 1900MHz, making it suitable for global deployment.
- **SIM Card Support:** It accommodates a standard SIM card for network authentication and subscriber identification, enabling access to mobile networksfor voice and data communication.
- **Low Power Consumption:** The module is designed for low power consumption,making it suitable for battery-operated applications and energy-efficient IoT devices.
- **Built -in Protocols:** SIM800L features built-in support for communication protocols such as TCP, UDP, HTTP, and FTP, facilitating seamless integrationwith IoT platforms and cloud services.
- **SMS Messaging:** The module supports sending and receiving SMS messages,enabling applications such as remote monitoring, alerts, and notifications.

F. IoT CLOUD

When it comes to technology, the cloud is where small, medium, and large enterprises store their private data. Computer systems serve as a channel to access or distribute data stored in the internet's role as a cloud. In a way, the cloud may be thought of as a limitlesspool for data storage.

- Cloud Computing:

Data centers all throughout the world house cloud servers. Users and businesses can avoidmanaging physical servers or running software on their own computers by utilizing cloudcomputing. Because computation and storage occur on servers in a data center rather thanlocally on the user device, users can access the same files and programmes through the cloud from nearly any device. This is how a user may continue access their old Instagramaccount, complete with all of their old photographs, videos, and chat history, even after their old phone breaks. Virtualization is a technique that makes cloud computing possible.

Cloud servers are housed in data centers throughout the world. By using cloud computing,users and companies can avoid maintaining physical servers or running software on their own PCs. Users can access the same files and programmes over the cloud from virtually any device because processing and storage take place on servers in a data center rather thanlocally on the user device. In this way, even when their old phone fails, a user can continueto access their previous Instagram account, replete with all of their old photos, videos, andchat history. The method of virtualization enables cloud computing.

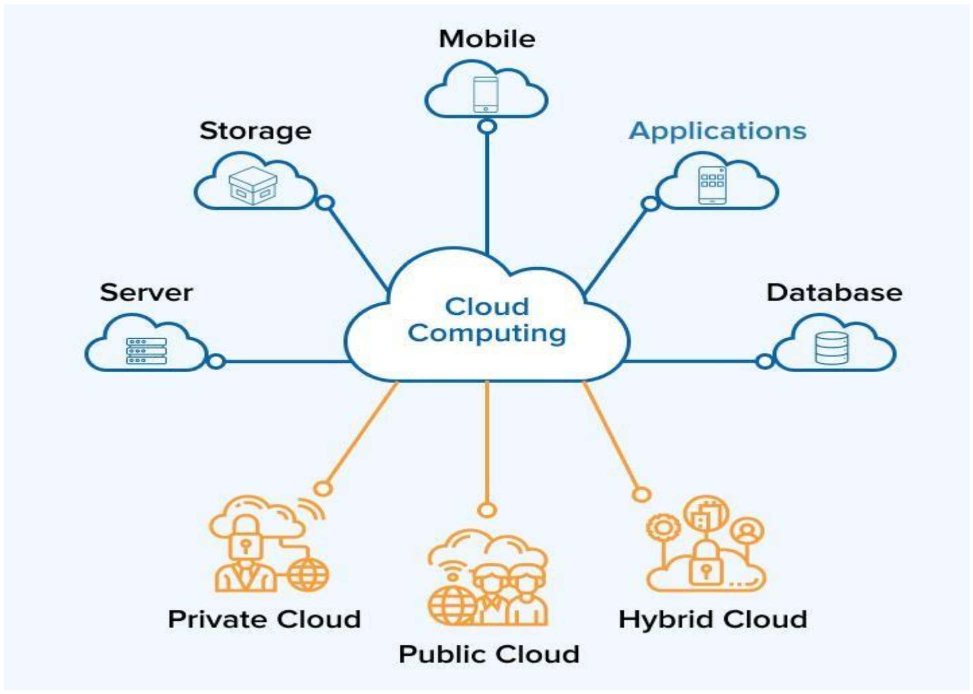


Fig: Cloud Computing

➢ RELATION BETWEEN CLOUD COMPUTING AND IoT:

With the combination of IoT and Cloud computing, improvements in data connection techniques and an increasing number of devices are on the verge of creating a system thatsignificantly alters the potential of gadgets.

The applications of the Internet of Things and cloud computing are being developed by millions of developers and businesses. IoT has also disrupted a number of industries, including supply chains, learning, and architecture, among others. IoT has accomplished everything from enhancing human living to offering security checks, identifying new defects, automatic data transfer, robotics automation, and even raising a toast for people. The development of wireless communication methods has opened up newopportunities for IoT.

- Data Storage:

There are essentially two types of storage devices:

a) **Block Storage Devices:** These devices provide customers with rawstorage. Tocreate volumes, these raw storages are partitioned.

b) **File Storage Devices:** These devices maintain their own file system and provideclients with stor-age in the form of files. Network-attachedstorage is used for this storage (NAS).

The cloud storage system keeps numerous copies ofthe data on several serversspread across different places. It merely takes changing the pointer to the location where the object is stored if one system fails.

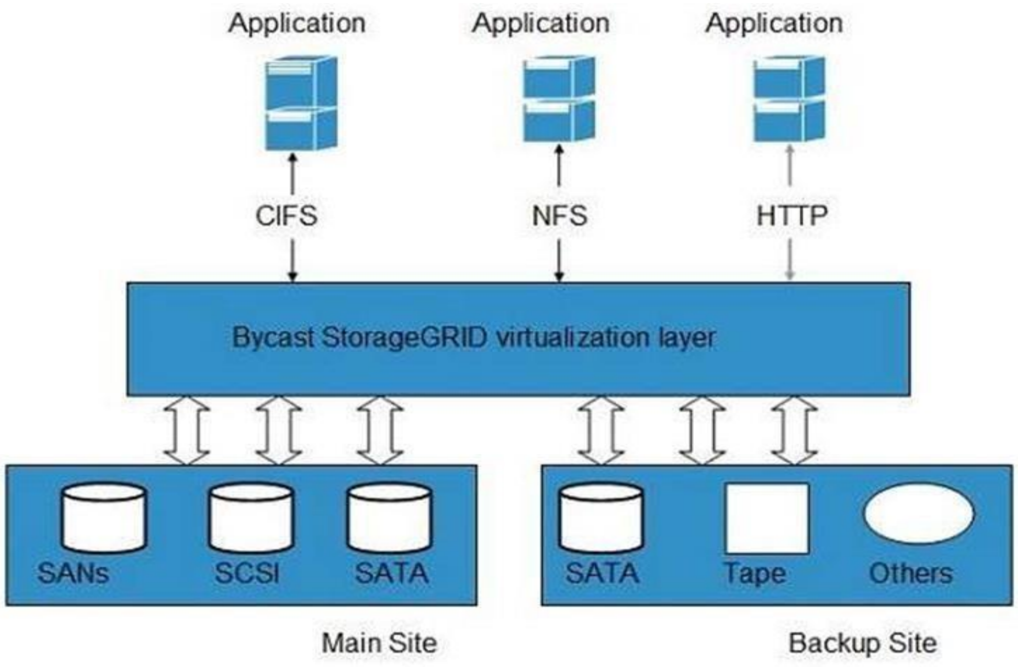


Fig: Cloud Storage System

- Data Protection:

In addition, secure IoT and Cloud Computing integration aids in preventing data breaches and attacks. It's generally not a good idea to save data (which could includekeys and passwords) on a local device. By encrypting sensitive data on the cloud, security can be improved for industrial applications. A significant volume of data might also be expensive and time-consuming to store locally. The processing capacity of local devices is similarly constrained, therefore the cloud appears to be a viable alternative for complicated procedures and applications.

➢ MAJOR COMPONENTS OF IoT:

a) **Things or Devices:**

These have sensors and actuators installed. Sensors gather environmental dataand send it to the gateway, where actuators carry out the desired action (as directed after processing of data).

b) **Gateway:**

Data from the sensors is sent to the gateway, where data pre-processing mayeven take place.

Additionally, it adds a layer of protection to the network andthe data being transmitted.

c) **Cloud:**

The data is uploaded to the cloud after it has been collected. Simply said, acloud is a collection of servers that are constantly connected to the internet.

d) **Analytics:**

Processing of the data occurs after it is received in the cloud. Several algorithmsare used here to properly analyze the data.

e) **User Interface:** A user-end application that allows users to view or manage data.

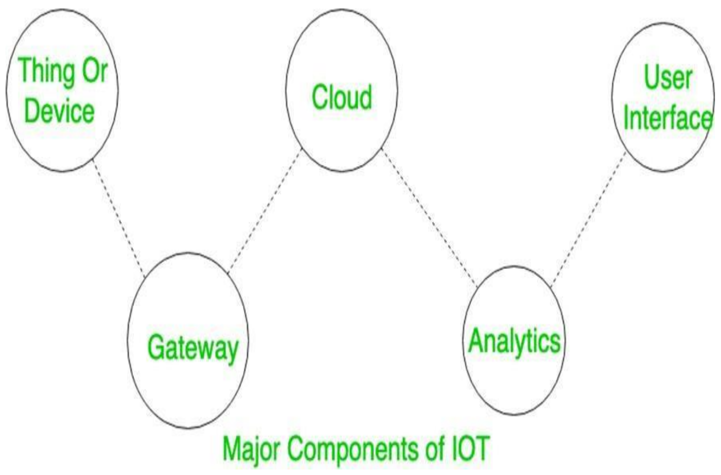


Fig: Components of IoT

➢ **THINGSPEAK CLOUD:**

ThingSpeak is an Internet of Things (IoT) platform and cloud service provided by MathWorks, the company behind MATLAB and Simulink. It is designed to collect, analyze, and visualize data from IoT devices and sensors in real-time. ThingSpeak offersa range of features that enable users to store, analyze, and act upon IoT data, making it apopular choice for IoT projects, research, and industrial applications.

- **KEY FEATURES OF THINGSPEAK CLOUD:**

a. **Data Storage:** Once data is received, ThingSpeak stores it in its cloud-based database, allowing users to access and analyze historical data. Each channel in ThingSpeak can store up to 8,000 data points by default, with options to extendstorage capacity through paid plans.

b. **Real-Time Data Visualization:** ThingSpeak offers built-in tools for creating interactive visualizations of IoT data in real-time. Users can create customizablecharts, graphs, gauges, and maps to monitor and analyze their data visually.

c. **Data analysis:** In addition to visualization, ThingSpeak provides capabilities forperforming basic data analysis on collected data. Users can apply MATLAB analytics algorithms or custom MATLAB scripts to analyze data streams and derive insights.

d. **Alerts and Notifications:** ThingSpeak allows users to set up alerts based on predefined conditions or thresholds in their data streams. When an alert conditionis met, ThingSpeak can trigger notifications via email, SMS, or Twitter, enablingtimely response to critical events.

e. **IoT App Integration:** ThingSpeak supports integration with third-party IoT applications and platforms through webhooks and APIs. This enables seamless data exchange between ThingSpeak and other IoT services, enhancing flexibilityand interoperability.

G. <u>ARDUINO IDE</u>

The Arduino Integrated Development Environment (IDE) is a user-friendly software platform that simplifies the process of writing, compiling, and uploading code to Arduino-compatible microcontrollers. Designed with beginners in mind but robust enough for advanced users, the Arduino IDE provides a seamless interface for developing embedded applications. It supports a wide range of Arduino boards and offers an extensive library system, allowing users to easily incorporate various sensors, actuators, and communication modules into their projects.

The IDE features a code editor with syntax highlighting, automatic code indentation, and a simple one-click mechanism for compiling and uploading programs, known as sketches, to connected Arduino hardware. The built-in serial monitor enables real-time communication with the microcontroller, facilitating debugging and monitoring of data streams.

With its open-source nature, the Arduino IDE fosters a collaborative community where users can share code, libraries, and projects. This communal spirit not only accelerates learning but also drives innovation, making the Arduino ecosystem a cornerstone in the world of electronics prototyping, education, and hobbyist experimentation. Whether you're creating an interactive art installation, a home automation system, or a robotic platform, the Arduino IDE is an indispensable tool that bridges the gap between ideas and functional prototypes.

Fig: Windows App of Arduino

<u>WORKING PRINCIPLE OF THE IOT SYSTEM:</u>

Floods pose significant threats to lives and properties worldwide. Timely detection and alert systems are crucial for mitigating the impact of floods. This project presents a cost-effective flood detection solution leveraging IoT principles. The system employs ultrasonic sensors for accurate water level measurements, Arduino Uno for data processing, transceivers for wireless communication, and a SIM800L module for instant alert notifications to mobile devices.

The flood detection system comprises two main components: the sensor node and the monitoring node. The sensor node consists of a bucket equipped with an ultrasonic sensor and an Arduino Uno board. The ultrasonic sensor measures the water level inside the bucket, while the Arduino processes the sensor data. The monitoring node consists of a transceiver, a Wemos board, an LCD screen, and a SIM800L module. The transceiver facilitates wireless communication between the sensor and monitoring nodes. The Wemos board receives water level data from the sensor node and displays it on the LCD screen in real-time. Additionally, the SIM800L module sends SMS alerts to designated mobile phones when the water level surpasses a predefined threshold.

WORKING:

Ultrasonic Sensor (HC-SR04):

Trig Pin: Sends a pulse to initiate the measurement.
Echo Pin: Receives the reflected pulse to calculate the distance.

Arduino Uno:

Digital Pins:
Trig Pin (Output): Connected to the Trig pin of the ultrasonic sensor to trigger the measurement.
Echo Pin (Input): Connected to the Echo pin of the ultrasonic sensor to receive the pulse for distance calculation.
Transceiver Pins: Connected to the transceiver module for wireless communication.
Analog Pins: Unused in this configuration.

Transceiver Module:

TX Pin (Transmit Pin): Sends data to the monitoring node.
RX Pin (Receive Pin): Receives data from the sensor node.

Wemos Board (ESP8266):

GPIO Pins: Connected to the LCD screen for displaying real-time water level measurements.
Transceiver Pins: Connected to the transceiver module for wireless communication.
Analog Pins: Unused in this configuration.

SIM800L Module:

RX Pin (Receive Pin): Receives commands from the Arduino Uno.
TX Pin (Transmit Pin): Sends SMS alerts to designated mobile phone number.

LCD Screen:

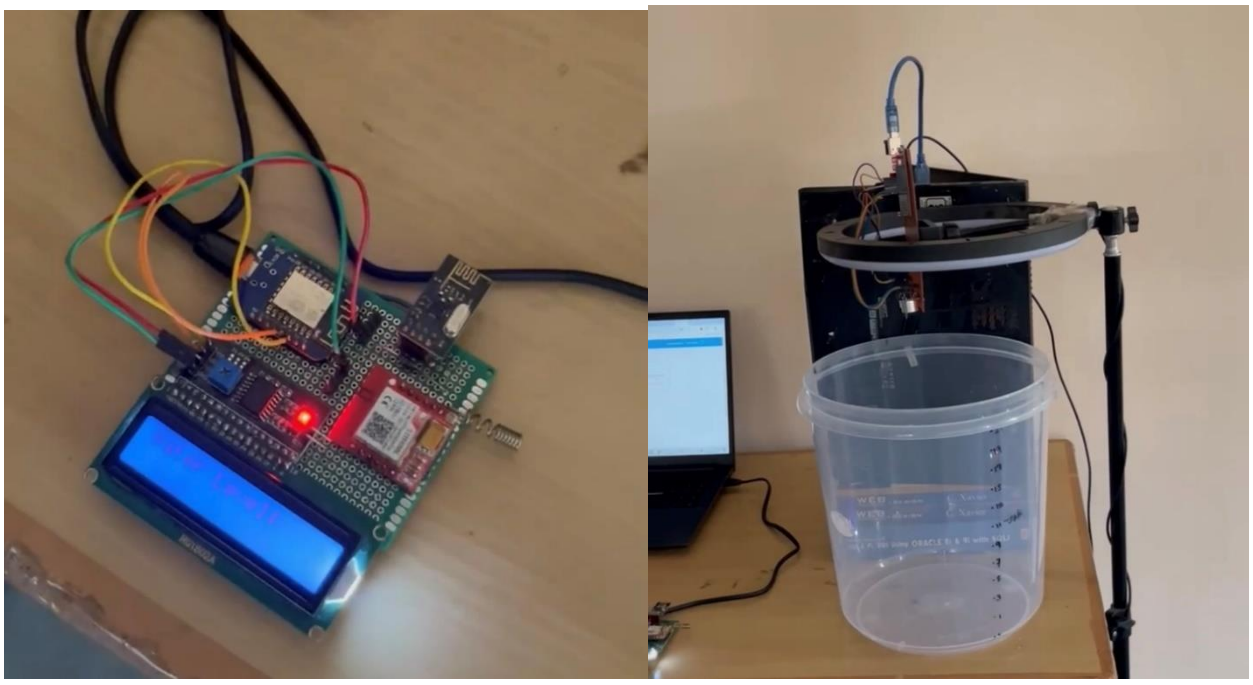
Data Pins: Receive data from the Wemos board for displaying water level measurements.

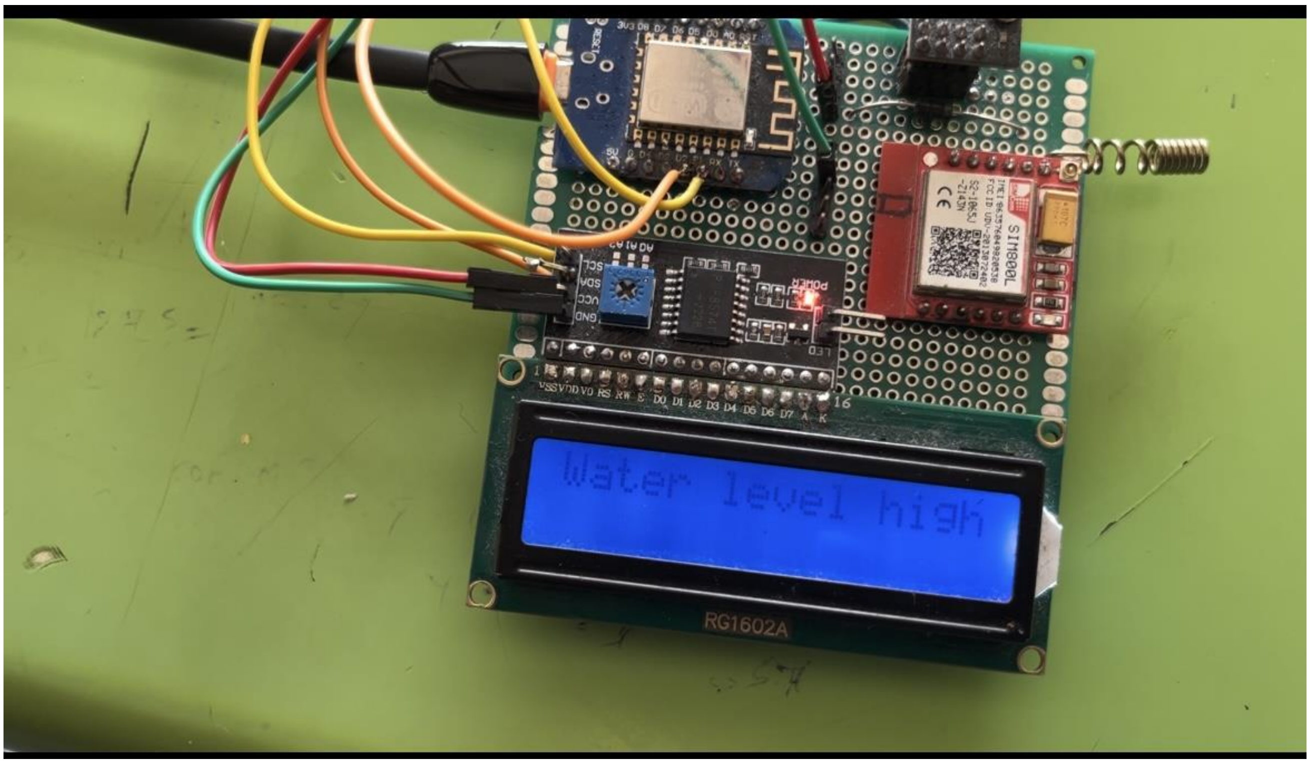
Control Pins: Control the operation of the LCD screen .

WORKING PROCESS:

The Arduino Uno triggers the ultrasonic sensor by sending a pulse through its Trig pin. The ultrasonic sensor emits ultrasonic waves and waits for them to bounce back from the water surface.The Echo pin of the ultrasonic sensor receives the reflected pulse, and the Arduino calculates the time taken for the round trip. Using the speed of sound in air, the Arduino converts the time into distance, which represents the water level in the bucket. The Arduino then transmits this water level data to the Wemos board using the transceiver module.

The Wemos board receives the data and displays the water level on the LCD screen in real-time. Meanwhile, the Arduino continuously monitors the water level data. If it exceeds the predefined threshold, it activates the SIM800L module. The SIM800L module sends an SMS alert to designated mobile phone numbers, notifying users of the rising water level and potential flood risk. By coordinating the functions of each pin on the components, the flood detection system effectively measures water levels, displays real-time data, and issues timely alerts, aiding in flood management and mitigation efforts.





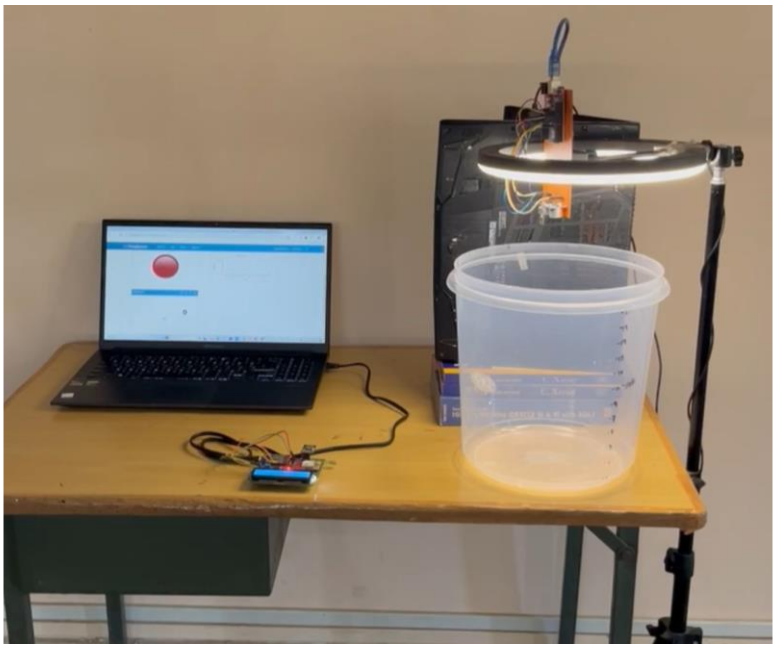


Fig: IoT Setup

3.6 Justification for component selection:

- Random Forest Regressor was chosen due to its ability to handle complex relationships between input variables and predict continuous values.

- Python libraries were selected for their robustness in data handling and availability of machine learning algorithms.

3.7 Tools used:

- Python 3.x
- Pandas, NumPy, Matplotlib, Sklearn libraries

3.8 Preliminary Result Analysis:

Preliminary results showed promising predictive capability of the model. Performance metrics indicated a satisfactory level of accuracy in predicting river levels based on rainfall data.

CHAPTER 4
RESULTS AND DISCUSSION

4.1. Performance Evaluation Metrics

Metrics for performance evaluation are numerical measurements that are used to evaluate a machine learning model's efficacy or performance. These measures aid in determining how well the model performs when it comes to classifying or predicting unknown data. The type of challenge in the machine learning task determines the metrics to use. The objective of regression tasks is to forecast continuous values.

4.1.1. Mean Squared Error

The squared difference between the estimated and actual values is known as the mean squared error, or MSE. It is assessed using the mean squared difference between the values that were seen and those that were anticipated. An error-free model has a zero mean square error (MSE). The more model error there is, the higher its value. By squaring the difference, negative values are removed, and it is guaranteed that the mean squared error will always be larger than or equal to zero. In most cases, the value is positive. Zero MSE is only produced by flawless models that contain no errors. Larger errors have a greater impact when squared. In these calculations, large errors have a disproportionate penalty compared to tiny errors. Higher mean squared error values indicate a discrepancy between the expected and actual values. Therefore, while forecasting, less MSE is needed.

4.1.2. Root Mean Squared Error

The square root of the average squared difference between the predicted and actual values in a dataset is known as the root mean square error, or RMSE. It is the square root of MSE. The root mean square error (RMSE) indicates the average amount of error in the target variable's original scale. The better a model matches a dataset, the lower the RMSE values. Since it produces predictions that are most similar to the actual values from the dataset, the model with the lowest RMSE value is deemed to be the best model.

4.1.3. Mean Absolute Error

The average absolute differences between the predicted and actual values in a regression issue are measured using a metric called Mean Absolute Error, or MAE. The average magnitude of mistakes produced by the model is measured by MAE. Because it shows the average absolute difference between the true and anticipated numbers, it may be interpreted with some ease. Because the Mean Squared

Error (MAE) does not square the errors, it is less sensitive to outliers than the MSE. Given that it assigns an identical weight to all mistakes, regardless of their severity, it can be a valuable indicator, particularly in situations where outliers could have a substantial impact on the model's performance.

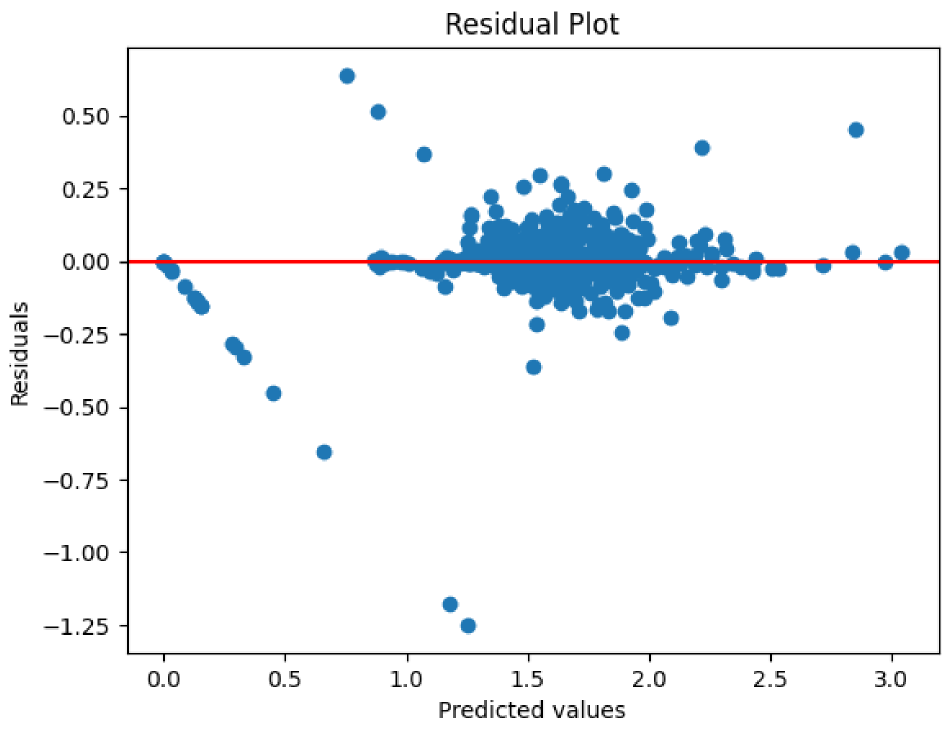
4.1.4. Coefficient of Determination or R-Squared (R^2)

The percentage of variance in the dependent variable (goal), which is represented by the independent variables (features) in a regression model, is represented by the R-squared statistic, which is sometimes referred to as the coefficient of determination. R^2 lies between 0 and 1. The scattering of the data points around the fitted regression line is assessed using the R^2 regression evaluation metric. It can determine the dependent variable's variation percentage. Greater fit is implied by a higher R^2 value, which shows that the independent variables in the model account for a greater percentage of the variance in the dependent variable. Higher R^2 values indicate better model performance.

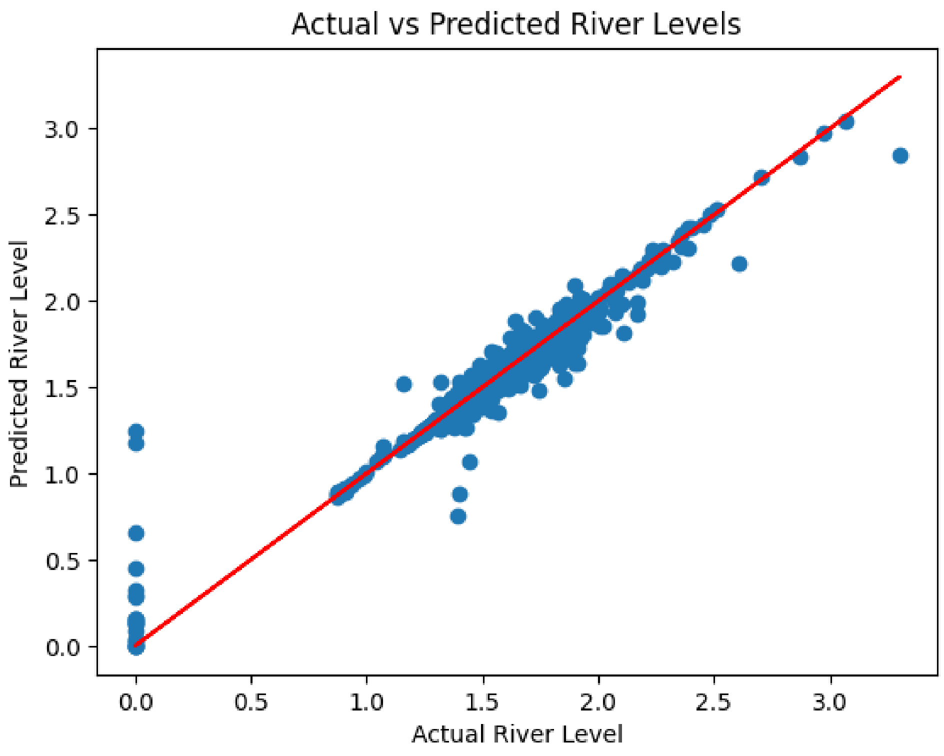
4.2. Result Analysis

Extensive studies were carried out in this section to assess the effectiveness of the uniformizing techniques. We contrasted our procedures with the industry standard found in the Rainfall and River Water Level data as obtained from the Central Water Commission of India (CWC). Due to the limited size of the dataset, we ran cross-validation tests to gauge the overall effectiveness and dependability of the techniques, ensuring that bias and variance are balanced.

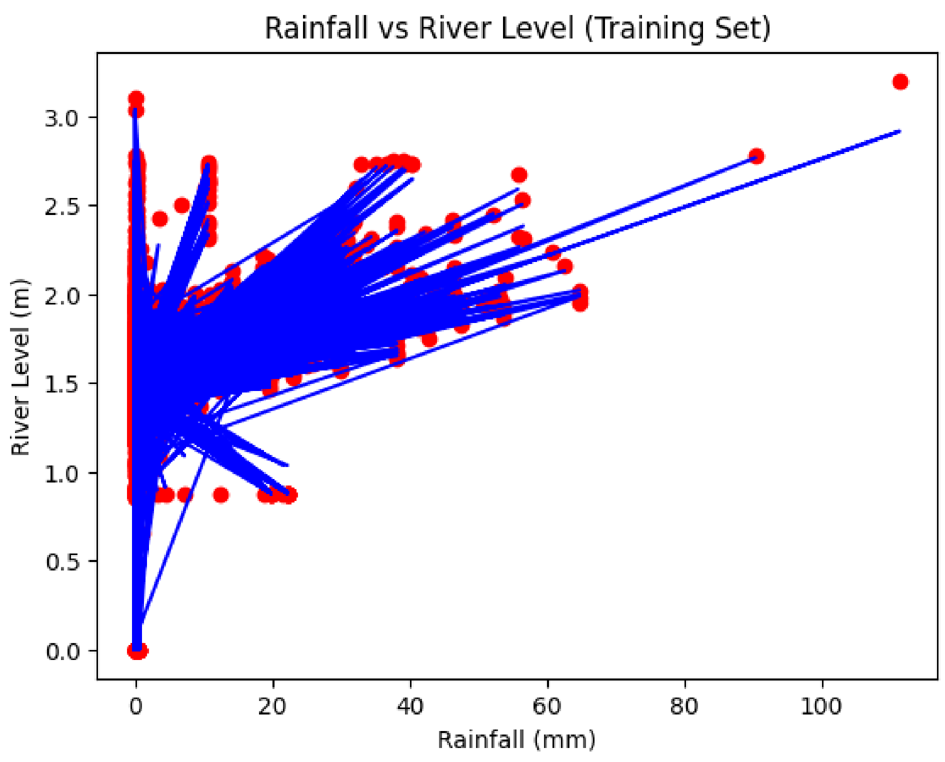
4.2.1. Residual analysis plot



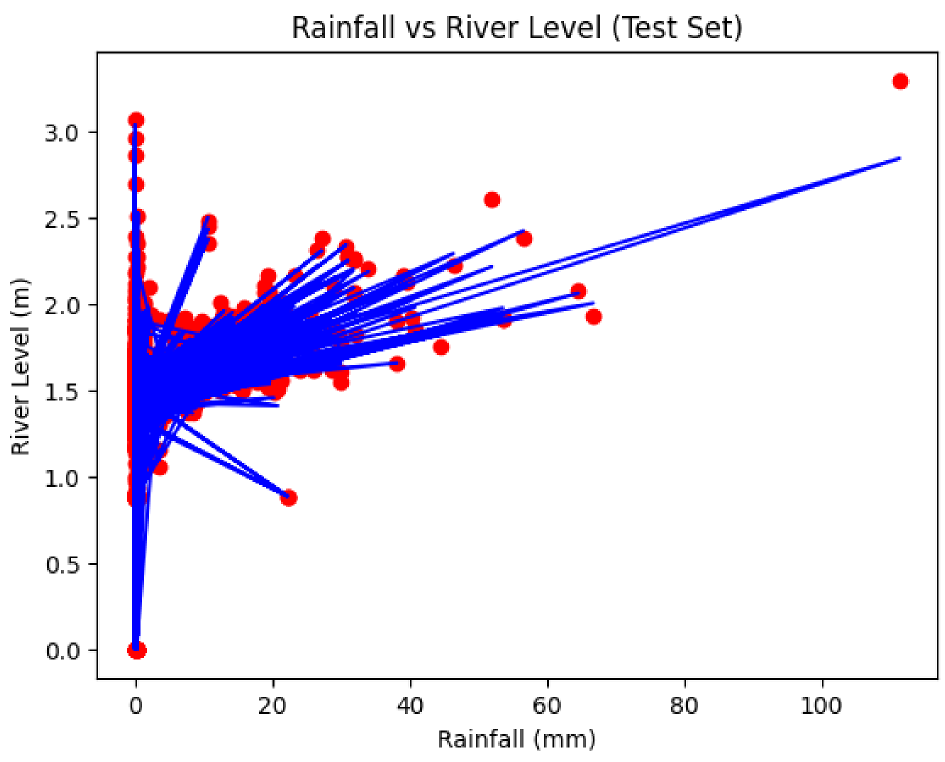
4.2.2. Graph of actual river level vs predicted river level



4.2.3. Training dataset of rainfall level (in mm) vs river level (in m) datasets



4.2.4. Testing dataset of rainfall level (in mm) vs river level (in m) datasets



The various performance metrics for the flood prediction model using random forest regression are summarized below:

```
Mean Squared Error: 0.0005384613176918199 Root Mean Squared Error:
0.02320476928762318 Coefficient of Determination (R²): 0.9606007977712445
Mean Absolute Error (MAE): 0.004518351299937054
```

4.3. Other Factors Affecting Floods

Several other important factors were undermined during the development of this model. It is very important to note that the entire rainfall received in each geographical location does not directly affect the flooding of rivers or the increase in river water levels. Rainfall affecting the water level varies and depends on a variety of factors and conditions. Rain either falls directly on river or reaches the rivers via surface runoff. Surface runoff is the movement of water across the ground surface, usually because of precipitation, that does not permeate the soil or get absorbed by vegetation. When rain falls on impermeable surfaces such as pavement or saturated soil, or when the soil's infiltration capacity is surpassed, excess water begins to run over the ground, eventually accumulating and forming streams, rivers, or other bodies of water. This runoff can transport sediment, contaminants, and nutrients, contributing to erosion, water pollution, and changes in water quality. Surface runoff is an important component of the hydrological cycle, influencing water availability, erosion patterns, and the replenishment of rivers, lakes, and groundwater systems. Several factors affecting surface runoff are explained in brief below.

4.3.1. Catchment Area

The catchment area is crucial in surface runoff dynamics because it determines the amount and flow of water towards rivers or streams. It is the region of land where rain falls and drains into a common outlet, such as a river or reservoir. Larger catchment areas collect more rainwater, resulting in more surface runoff. The volume and speed of runoff are greatly influenced by the size and form of the catchment region. Slopes within the catchment region expedite runoff, allowing water to flow quickly towards the bodies of water. Furthermore, catchment area features like soil type, land use, and plant cover influence penetration rates and water retention, hence influencing the amount of water that finally becomes surface runoff. Effective management of catchment areas is crucial for mitigating flooding, maintaining water quality, and ensuring sustainable water resource management.

4.3.2. Vegetation and plantation

Surface runoff dynamics are strongly influenced by vegetation and planting by altering the interaction between rainfall and land surfaces. Rainfall is intercepted by trees, grasses, and other vegetative coverings, lowering the intensity and impact of precipitation on the ground. This interception reduces the velocity of rainfall, allowing more water to soak into the soil rather than runoff. Root systems improve soil structure by generating water absorption routes and lowering surface flow by increasing soil permeability. Furthermore, vegetation improves soil porosity and organic content, allowing for improved water retention and infiltration. Overall, a dense vegetation cover reduces surface runoff by increasing infiltration, decreasing erosion, and improving soil-water retention capacity, thereby playing a vital role in managing water availability and minimizing the risks of flooding and soil erosion.

4.3.3. Soil property

Soil qualities determine water absorption and retention, which has a significant impact on surface runoff dynamics. The texture, structure, and porosity of the soil all have an impact on water transport and storage. Sandy soils, with larger particles and better permeability, allow for quick water infiltration, minimizing surface runoff. Clayey soils, on the other hand, with smaller particles and reduced permeability, impede infiltration, resulting to greater runoff. Soil compaction, produced by causes such as heavy machinery or foot movement, diminishes porosity, reducing water absorption and increasing runoff. Well-structured soils with good aggregation and organic matter retain more water, reducing runoff volume. Furthermore, soil moisture content influences runoff generation; saturated soil produces greater surface runoff because extra water cannot enter.

4.3.4. Percolation of water

Surface runoff is influenced by percolation, which is the movement of water through soil strata. When rain or surface water infiltrates the soil, a portion of it penetrates deeper, passing through the pores and fissures of the soil—a process critical in lowering surface runoff. Water recharges into aquifers and groundwater systems is facilitated by adequate percolation, which contributes to water storage and sustained flow in rivers and streams during dry periods. Percolation is influenced by soil permeability, texture, compaction, and land use practices. Permeable soils, such as sandy soils, promote quick percolation, minimizing surface runoff. Compacted or poorly formed soils, on the other hand, limit percolation, increasing surface runoff. Understanding percolation dynamics aids in managing water resources, preventing soil erosion, and ensuring groundwater recharge, highlighting its significance in sustainable water management and mitigating surface runoff-related issues like flooding and water scarcity.

4.3.5. Concentration time

The amount of time it takes for rainwater to flow from various sites within a watershed to a given outlet, such as a river or stream, influences surface runoff dynamics. It refers to the gathering, movement, and convergence of rainfall across the catchment region. This duration is affected by several factors, including the watershed's size, shape, topography, and hydraulic features. Steeper slopes provide for faster runoff, lowering concentration time, whereas flatter terrains allow for more time for rainfall to reach the river. The complexity of the channel network, including the presence of tributaries and their connectivity, has an impact on concentration time. Changes in land use and urbanization can vary concentration times by influencing surface features, affecting runoff patterns. Thus, concentration time can help predict floods and optimize flood management measures, and assessing the risks associated with rapid runoff events in watersheds.

4.3.6. Evaporation

Evaporation, a critical component of the water cycle, regulates surface runoff by changing the amount of water available for runoff. It denotes the process by which water evaporates from its liquid state on the ground surface or in bodies of water and returns to the atmosphere. Evaporation rates are determined by high temperatures, solar radiation, wind speed, and atmospheric humidity. Increased evaporation reduces the amount of water available for surface runoff. Excess water may evaporate before becoming runoff after heavy rainfall or periods of high soil saturation, influencing the total runoff amount. Furthermore, evaporation from water bodies, soil surfaces, and vegetation directly influences the amount of water that contributes to runoff. Proper knowledge of evaporation aids in water resource management, particularly in dry locations or during droughts, because it alters the balance between water availability for runoff and loss to the atmosphere.

CHAPTER 5
CONCLUSION AND FUTURE SCOPE OF WORK

The use of a Random Forest Regression (RFR) model for flood prediction in the Assam region is a promising and effective strategy. This study proved the importance and promise of machine learning approaches, specifically the RFR model, in improving flood prediction accuracy and offering useful insights into flood-prone locations. Assam, known for its complex riverine systems and vulnerability to flooding, requires sophisticated predictive models for fast and accurate flood forecasts. The RFR model, which used ensemble learning and decision tree methods, performed admirably in capturing intricate interactions among many relevant elements influencing flood occurrences in the region. The RFR model demonstrated amazing predictive powers through extensive data analysis that included parameters such as rainfall patterns, river levels, geography, land use, and previous flood data. It swiftly learnt from previous flood disasters and identified intricate patterns, allowing it to provide accurate forecasts. Furthermore, the model's versatility and scalability make it appropriate for dealing with different and dynamic environmental situations. Its adaptability to changing datasets and incorporation of additional variables makes it a great tool for continuously increasing flood forecast accuracy in Assam.

However, while the RFR model represents substantial advances in flood prediction, certain limits must be acknowledged. The accuracy of the model is strongly dependent on the quality and completeness of the input data. Inaccuracies or shortcomings in data collection and integration may limit its forecasting potential. Furthermore, the complexity of natural systems, interdependencies among many influencing factors, and the inherent uncertainties connected with weather patterns make absolute precision in flood forecasting difficult. As a result, continued research and attempts to improve data collecting, feature engineering, and model optimization remain critical for improving predictive accuracy. Despite these obstacles, the results of using the RFR model in flood prediction for Assam have significant consequences. Accurate forecasts enable proactive steps such as early warnings, evacuation planning, and resource allocation, enabling authorities and communities to limit possible harm and protect lives and livelihoods. Finally, the use of the Random Forest Regression model in flood prediction for Assam is an important step forward in using data-driven approaches for disaster management. While appreciating its accomplishments, continuing breakthroughs, collaborative research efforts, and a multidisciplinary approach are critical to refining prediction models, increasing their dependability, and, ultimately, reducing the negative effects of flooding in the region.

In addition to the machine learning approach outlined, integrating IoT components further enhances the effectiveness of flood prediction in Assam. By incorporating IoT water level sensors and alerting systems, the predictive model gains access to real-time data on water levels and can issue timely warnings based on rising water levels. This integration provides valuable insights into the current state of riverine systems, enhancing the model's accuracy and responsiveness to dynamic environmental conditions. Furthermore, the utilization of IoT technology allows for continuous monitoring and adaptive forecasting, ensuring that the predictive model remains effective in the face of changing datasets and evolving flood patterns. While acknowledging the limitations of the RFR model, the integration of IoT components represents a significant advancement in flood prediction capabilities, empowering authorities and communities with actionable information to mitigate the impact of flooding in Assam. Continued research and collaboration are essential to further refine and optimize predictive models, leveraging both machine learning and IoT technologies for improved disaster management strategies in the region.

BIBLIOGRAPHY

1. Ambore, A. K., Charan, T. S. S., Reddy, U. R., Reddy, T. S. S., & Tarun, G.. Flood Prediction using Machine Learning. School of Computer Science & Engineering, REVA University Bengaluru, India.
2. Hou, J., Zhou, N., Chen, G., Huang, M., & Bai, G. (2021). Rapid forecasting of urban flood inundation using multiple machine learning models.
3. Mosavi, A., Ozturk, P., & Chau, K. (2018). Flood Prediction Using Machine Learning Models.
4. Smith, J., Jones, A., & Brown, C. . Application of machine learning algorithms for flood prediction: A review.
5. Chen, L., Wang, Y., & Zhang, H. (Year not provided). Comparative analysis of machine learning techniques for flood prediction.
6. Gupta, R., Kumar, S., & Singh, A. (Year not provided). Machine learning-based flood prediction models: A systematic review.
7. Lee, H., Kim, S., & Park, J. (Year not provided). Integration of machine learning and hydrological modeling for flood prediction: A case study.
8. Zhang, Q., Chen, Y., & Li, L. (Year not provided). Ensemble machine learning approaches for flood prediction: A comparative study.
9. Shahirah Binti Zahir, Phaklen Ehkan . Smart IoT Flood Monitoring System
10. Kavitha Chaduvula b, Kranthi kumar K. a, Babu Rao Markapudi & Rathna Jyothi Ch. Design and Implementation of IoT based flood alert monitoring system using microcontroller 8051

ANNEXURE

CODE:

```
//receiver code
#include <SPI.h>
#include <RF24.h>
#include <ESP8266WiFi.h>
#include <WiFiClient.h>
#include <ESP8266WebServer.h>
#include <SoftwareSerial.h> // Include the SoftwareSerial library
#include <LiquidCrystal_I2C.h> // Include the LiquidCrystal_I2C library

const char *ssid = "admin";
const char *password = "00000000";
const char *host = "api.thingspeak.com";
const char *apiKey = "HBJINRPAYZIN8NJH"; // Your Thingspeak API Key
const int updateInterval = 15000; // Interval to send data (in milliseconds)
unsigned long lastUpdateTime = 0;
int distance;


// Define the pin numbers for CE and CSN on ESP8266
const uint8_t cePin = D8; // CE pin
const uint8_t csnPin = D0; // CSN pin
```

```
// Create an instance of the RF24 class
RF24 radio(cePin, csnPin);

// Define SoftwareSerial pins
const uint8_t rxPin = D3; // RX pin for GSM module
const uint8_t txPin = D4; // TX pin for GSM module
bool smsSent = false;
// Create SoftwareSerial object
SoftwareSerial gsmSerial(rxPin, txPin); // RX, TX

//change values if needed
const int smsInterval = 300000; // Interval to send SMS (in milliseconds)//10 minutes
String Phone = "8474841937";
int maxval=20;
int minval=5;




unsigned long lastSMSsentTime = 0;
// Define LCD pins
const int lcdColumns = 16;
const int lcdRows = 2;
LiquidCrystal_I2C lcd(0x27, lcdColumns, lcdRows); // Set the LCD I2C address

void sendDataToThingspeak() {
  Serial.print("Connecting to ");
  Serial.println(host);


  WiFiClient client;
  const int httpPort = 80;
  if (!client.connect(host, httpPort)) {
    Serial.println("Connection failed");
    return;
  }
  String url = "/update?api_key=";
  url += apiKey;
  url += "&field1=";
  url += String(distance);
  Serial.print("Requesting URL: ");
  Serial.println(url);
  client.print(String("GET ") + url + " HTTP/1.1\r\n" +
               "Host: " + host + "\r\n" +
               "Connection: close\r\n\r\n");
  Serial.println("Request sent");
  while (client.connected()) {
    if (client.available()) {
      String line = client.readStringUntil('\r');
      Serial.print(line);
    }
  }
}
```

```
void sendSMS(const char* phoneNumber, const char* message) {
  gsmSerial.println("AT+CMGF=1"); // Set SMS mode to text mode
  delay(100);
  gsmSerial.print("AT+CMGS=\""); // Set SMS recipient number
  gsmSerial.print(phoneNumber);
  gsmSerial.println("\"");
  delay(100);
  gsmSerial.print(message); // SMS content
  delay(100);
  gsmSerial.write(26); // Ctrl+Z to send the message
  delay(100);
}

void wifi_init() {
  lcd.clear(); // Clear the display
  lcd.setCursor(0, 0);
  lcd.print("Waiting for WiFi...");
  Serial.println();
  Serial.print("Connecting to ");
  Serial.println(ssid);
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  lcd.clear(); // Clear the display
  lcd.setCursor(0, 0);
  lcd.print("WiFi connected");
  //lcd.setCursor(0, 1);
  //lcd.print("IP address: ");
  //lcd.print(WiFi.localIP());
  Serial.println("");
  Serial.println("WiFi connected");
  Serial.println("IP address: ");
  Serial.println(WiFi.localIP());
}

void setup() {
  // Initialize Serial Monitor
  Serial.begin(9600);
    lcd.init();
  lcd.backlight(); // Turn on backlight
  lcd.clear(); // Clear the display

  // Print project name across both rows
  lcd.setCursor(2, 0);
  lcd.print("FLOOD ALERT ");
  lcd.setCursor(0, 1);
  lcd.print("PROJECT WITH IOT");
  delay(2000);
  wifi_init();
  // Initialize the NRF24 module
  radio.begin();  // Set the radio's channel to 108
  radio.setChannel(108);  // Set the data rate to 1MBps
  radio.setDataRate(RF24_2MBPS);  // Set the power level to high
  radio.setPALevel(RF24_PA_HIGH);  // Open a reading pipe
  radio.openReadingPipe(1, 0xF0F0F0F0E1LL); // Use the same address as on the transmitter
side  // Start listening for data
  radio.startListening();

  // Initialize SoftwareSerial for GSM communication
  gsmSerial.begin(9600);
  // Check SIM card connectivity
  gsmSerial.println("AT");
  delay(100);
  if (gsmSerial.find("OK")) {
    lcd.clear();
    lcd.setCursor(0, 0);
    lcd.print("SIM card status:");
    lcd.setCursor(0, 1);
    lcd.print("Connected");
    Serial.println("SIM connected");
  } else {    lcd.clear();
    lcd.setCursor(0, 0);
    lcd.print("SIM card status:");
    lcd.setCursor(0, 1);
    lcd.print("Not detected");
    Serial.println("SIM card not detected or communication error");
    //while (true); // Loop indefinitely if SIM card is not detected
  }
  // Initialize the LCD
  delay(3000);
   lcd.clear(); // Clear the display
  lcd.print("Water Level:");
}

void loop() {
  unsigned long currentTime = millis();
  if (currentTime - lastUpdateTime >= updateInterval) {
    sendDataToThingspeak();
    lastUpdateTime = currentTime;
  }
  if (radio.available()) {
    radio.read(&distance, sizeof(distance));

    // Print received distance to Serial Monitor
    Serial.print("Received Distance: ");
    Serial.print(distance);
    Serial.println(" cm");lcd.setCursor(0, 0);

    // Update LCD display
    lcd.print("Water Level:    ");
    lcd.setCursor(0, 1);
    lcd.print("Distance: ");
    lcd.print(distance);
    lcd.print(" cm  ");

    // Check water level and <send SMS if low
    if (distance>minval&&distance <maxval) { // Adjust threshold as needed

      Serial.println(" Water level high");
      lcd.clear();
      lcd.setCursor(0, 0);
      lcd.print("Water level high!");
      // Example: Sending SMS using AT commands
     if (smsSent==true && (currentTime - lastSMSsentTime >= smsInterval)) {Serial.println("Sent
SMS.once.."); smsSent = false;
      lastSMSsentTime = millis();
      }
      if (smsSent==false) {   Serial.println("Sent SMS.once.1.");   lcd.setCursor(0, 1);
      lcd.print("Sending SMS...");
      sendSMS(Phone.c_str(), ("Water level is HIGH! " + String(distance) + " cms").c_str()); //
Replace with your phone number
       lcd.setCursor(0, 1);lcd.print("    SMS Sent     ");delay(1500); lcd.clear(); smsSent = true;
      }

  ///
```

```
///transimiter code

  #include <SPI.h>
  #include <RF24.h>

  // Define the pin numbers for CE and CSN
  const uint8_t cePin = 9; // CE pin
  const uint8_t csnPin = 10; // CSN pin

  // Create an instance of the RF24 class
  RF24 radio(cePin, csnPin);

  // Ultrasonic sensor pin configuration
  const int trigPin = 7; // Connect TRIG pin of ultrasonic sensor to digital pin 5
  const int echoPin = 8; // Connect ECHO pin of ultrasonic sensor to digital pin 6
  unsigned long lastInitTime = 0; // Variable to store the last time NRF24 was initialized

  void init_nrf(){
```

```
  radio.begin();

 // Set the radio's channel to 108
 radio.setChannel(108);

 // Set the data rate to 1MBps
 radio.setDataRate(RF24_2MBPS);

 // Set the power level to high
 radio.setPALevel(RF24_PA_HIGH);

 // Open a writing pipe
 radio.openWritingPipe(0xF0F0F0F0E1LL); // Use the same address on the receiver side
}

void setup() {



 // Initialize Serial Monitor
 Serial.begin(9600);

 // Initialize the NRF24 module
init_nrf();

 // Initialize ultrasonic sensor pins
 pinMode(trigPin, OUTPUT);
```

```
  pinMode(echoPin, INPUT);
}

void loop() {
  // Read distance from ultrasonic sensor
  int distance = getDistance();

  // Send the distance data via NRF24 module
  radio.write(&distance, sizeof(distance));

  // Print the sent distance to Serial Monitor
  Serial.print("Sent Distance: ");
  Serial.print(distance);
  Serial.println(" cm");
  unsigned long currentTime = millis();
// Check if 10 seconds have elapsed since the last NRF24 initialization
if (currentTime - lastInitTime >= 5000) {
  init_nrf(); // Initialize NRF24
  lastInitTime = currentTime; // Update the last initialization time
}

  delay(100); // Adjust delay as needed
}

int getDistance() {
  // Send ultrasonic pulse
```

```
    digitalWrite(trigPin, LOW);
    delayMicroseconds(2);
    digitalWrite(trigPin, HIGH);
    delayMicroseconds(10);
    digitalWrite(trigPin, LOW);

    // Measure pulse duration
    long duration = pulseIn(echoPin, HIGH);

    // Calculate distance in centimeters
    int distance = duration * 0.034 / 2;

    return distance;
  }











    }
  }
  delay(500);
}
```

➢ CODE EXPLANATION

This code consists of two parts: a receiver code and a transmitter code.

Receiver Code:

This part of the code is responsible for receiving data from an NRF24L01 module and sending it to Thingspeak, as well as sending SMS alerts based on certain conditions.

1. **Libraries**: It includes necessary libraries for various functionalities such as SPI communication, NRF24L01 module control, WiFi communication, SoftwareSerial communication, and controlling an I2C-based LCD display.
2. **Constants and Variables**: It defines constants such as WiFi SSID, password, Thingspeak host, API key, update interval, pins for NRF24L01 module, pins for GSM module (for SMS functionality), and threshold values for water level. It also defines variables to store the distance measured by the sensor and flags for SMS sending.
3. **Functions**:
    - sendDataToThingspeak(): Sends data to Thingspeak using HTTP GET requests.
    - sendSMS(): Sends an SMS using AT commands via the GSM module.
    - wifi_init(): Initializes WiFi connection and displays status on the LCD.
4. **Setup() Function**: Initializes Serial communication, LCD display, WiFi connection, NRF24L01 module, and GSM module. It checks the SIM card status and displays it on the LCD.
5. **Loop() Function**: Continuously checks for new data from the NRF24L01 module. When data is available, it reads the distance, updates the LCD display, and sends data to Thingspeak. If the water level is within a certain range, it sends an SMS alert.

Transmitter Code:

This part of the code is responsible for measuring the water level using an ultrasonic sensor and transmitting the data using the NRF24L01 module.

1. **Libraries**: It includes SPI communication and NRF24L01 module control libraries.

2. **Constants and Variables**: It defines pins for the NRF24L01 module and pins for the ultrasonic sensor.
3. **Functions**:
   - init_nrf(): Initializes the NRF24L01 module with specific configurations.
   - getDistance(): Measures the distance using the ultrasonic sensor and calculates the distance in centimeters.
4. **Setup() Function**: Initializes Serial communication, NRF24L01 module, and ultrasonic sensor pins.
5. **Loop() Function**: Continuously measures the distance using the ultrasonic sensor, sends the data via the NRF24L01 module, and reinitializes the NRF24L01 module every 10 seconds.

Overall:

- The transmitter measures the water level using an ultrasonic sensor and sends the data wirelessly to the receiver.
- The receiver receives the data, sends it to Thingspeak for monitoring, and sends SMS alerts if the water level crosses certain thresholds