Published by :
https://www.ijert.org/
An International Peer-Reviewed Journal

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14 Issue 12 , December - 2025

Development of Automated Rain Gauge Monitoring System

Harold Jay S. Variacion[1], Mark Anthony M. Bugahod[2], Kaye Franzine A. Tubio[3*], Nathaniel D. Tillor[4], Jessriel F. Rangas[5], Vilma E. Salapang[6], Kent Russell N. Casiño[7], Neptale S. Roa III[8]

Tagoloan Community College, Philippines

Abstract The Automated Rain Gauge Monitoring System addresses challenges in manual rainfall data collection by automating measurement, improving accuracy, and providing real-time visualization for disaster preparedness. The system integrates hardware components, including the Arduino Uno, a GSM module, and tipping bucket rain gauge. with software developed using Laravel, MySQL, Arduino IDE, and front-end technologies such as HTML, CSS, Tailwind, Alpine.js, and Leaflet.js. Agile methodology guides development, enabling iterative improvements and user-centered design. Designed for staff to monitor rainfall and manage disasters, the system features automated data collection, real-time charts and maps, historical records, alert messages, and an intuitive dashboard. Functional testing and System Usability Scale evaluation yielded a score of 81.5, indicating high reliability, accuracy, and usability. By reducing manual workload and supporting timely, data-driven decisions, the system enhances disaster response. Future enhancements may include dynamic updates and integration with other platforms to improve efficiency and adaptability.

Keywords: *Automated Rain Gauge, Rainfall Monitoring, Real-Time Data, Arduino, Web-Based System, Disaster Preparedness, Data Visualization*.**.**

I. INTRODUCTION

In today's fast-paced world, technology has profoundly reshaped how we monitor and understand our environment, particularly in weather tracking and rainfall measurement. Advanced tools like Internet of Things (IoT) devices like smart rain gauges, wireless sensor networks (WSNs), and real-time monitoring systems have replaced traditional manual methods, improving accuracy and efficiency. The Internet of Things (IoT) transformed everyday life by enabling smart communication between devices and people (Mouha, 2021). These innovations made rainfall tracking more reliable and accessible, particularly for disaster preparedness and environmental management. With climate change intensifying weather unpredictability, the demand for real-time and automated rainfall monitoring solutions become more crucial than ever. Inefficiencies existed in the manual monitoring of rainfall data collection which relied on human intervention. The manual methods often caused delays in data measurements, manual logging and calculation made it difficult to track rainfall patterns over time, reducing the reliability of historical data for decision-making, and affecting the disaster response teams' ability to issue warnings. The rainfall monitoring and measurement were critical for various applications, especially in disaster preparedness. (Kumari et al., 2021).

The development and implementation of an Automated Rain Gauge Monitoring System for the Municipal Disaster Risk Reduction and Management Office (MDRRMO) was critically important in the context of increasing rainfall variability and the need for timely disaster response. At the time, the rainfall monitoring processes relied heavily on manual data collection, which was time-consuming and prone to human errors, leading to delays and inaccuracies in rainfall data. Manual rainfall monitoring methods resulted in unreliable data that hindered disaster preparedness and response efforts (Saniel et al. 2023). The proposed system aimed to improve the accuracy and efficiency of rainfall data collection by automating the monitoring process and enabling real-time tracking of rainfall levels. This enhancement supports MDRRMO's ability to issue timely warnings and make informed decisions during critical rainfall levels. Additionally, the data gathered by the system provided valuable information for researchers studying drainage systems and the effects of rainfall on the community's environment. By automating rainfall monitoring, this project addressed key challenges and contributed to better disaster risk management and climate research (Rohmah et al.,2024). The system promises to optimize rainfall monitoring operations, minimize errors, and provide reliable data for improving environmental management and public safety.

This study was conducted at the Municipal Disaster Risk Reduction and Management Office (MDRRMO) in Tagoloan, Misamis Oriental, where rainfall monitoring was performed manually. The MDRRMO used a Tipping Bucket Rain Gauge (TBRG) to collect rainfall, while a digital display unit showed the number of tips (clicks), with each tip equivalent to 0.2 mm of rainfall. The accumulative rainfall and corresponding intensity level were then manually recorded into Microsoft Excel for documentation and analysis. The proposed system aimed to improve this process by automating data collection, calculation, analysis, and report generation.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14 Issue 12 , December - 2025

This institution was chosen to implement an automated system to address the gaps and limitations of its existing manual process, by providing a more reliable solution for rainfall monitoring, early warning dissemination, and data utilization, thereby enhancing disaster preparedness and risk management in the community. Effective rainfall measurement played a critical role in mitigating the risks associated with extreme weather conditions (Panganiban, 2020).

The current rainfall monitoring process required human intervention, with staff having to go outside every 15 minutes (Time interval) to check the rain gauge display unit for rain tip. Data calculation, logging, and retrieval for report generation and analysis were all done manually, which often led to inconsistencies, delays, and human errors. These inefficiencies slowed down report preparation and data access, making it more difficult for administrators and staff to issue timely warnings and respond effectively to support disaster prevention and preparedness. Manual data collection methods may result in delays and inaccuracies, which hinder the timely dissemination of warnings and effective disaster response World Meteorological Organization, 2021. Automated systems were recommended to ensure real-time data availability and to support proactive disaster risk management.

The proposed system integrates both hardware and software to automate real-time monitoring and data collection, analysis, and report generation for historical records. It utilized a Tipping Bucket Rain Gauge (TBRG) to collect rainfall water. The generated data from the TBRG device was processed by an Arduino microcontroller and a GSM Module, which then transmitted, stored the data in a centralized database, and displayed on a web-based dashboard for real-time monitoring. This automation eliminated manual processes in rainfall monitoring. Modern technologies have been incorporated to improve accuracy, efficiency, and ease of data access (Kamdi, et al., 2024). With this system, staff and admin were able to monitor rainfall more easily and enhance community disaster preparedness and response. Rainfall measurement and flood warning systems played a critical role in mitigating the risks associated with extreme weather conditions (Panganiban, 2020).

The implementation of this project integrated various IT principles and software development methodologies, utilizing a web-based application built in the VS Code environment. MySQL was used for database management, API communication between the hardware and the backend system. JavaScript handled system functionalities, while HTML and CSS and other open-source application managed the front end of the system by dynamically updating the user interface and ensuring that real-time rainfall data was displayed. The system enhances the user experience through interactive charts and data visualizations in the user interface.

The hardware components included a Tipping Bucket Rain Gauge (TBRG), an Arduino microcontroller, and a GSM module to establish connectivity between the hardware and the web-based system. The integration of Arduino microcontrollers and GSM modules has proven effective in automating rainfall monitoring systems. Previous implementations, such as an Arduino-based rainfall and flood monitoring system, successfully provided real-time alerts through SMS notifications (Zamarro, et al., 2025). A web-based dashboard was developed specifically for the administrators and staff of the organization, providing real-time data access and monitoring capabilities, and data utilization.

Purpose and Description

The Automated Rain Gauge Monitoring System aimed to provide an efficient and reliable solution for monitoring rainfall levels. This system was designed to address the limitations of the manual rainfall monitoring process in the MDRRMO by integrating hardware sensors and a web-based platform, ensuring real-time monitoring and data collection, automated calculations, report generation, and data analysis. The collected data was to accurately measure rainfall intensity and to store, analyze, and visualize this information for decision-making, early warning, and disaster preparedness. Rainfall intensity-duration thresholds are commonly used to assess flood potential in both urban and rural environments (Georganta et al., 2022).

Real-time monitoring of rainfall data enabled the MDRRMO to track rainfall conditions continuously and issue timely alerts to barangay officials during heavy rain events or critical situations. These officials utilized the data to inform residents of potential flood risks and implement preventive measures. Additionally, the data was automatically stored in a central database, where it was used to generate reports, identify rainfall patterns, and analyze long-term rainfall trends. These data can serve as a reference for future studies and for planning infrastructure improvements, such as drainage systems, to address urban flooding within the community and enhance local disaster resilience. Rain gauges were the validated technological instruments used to measure rainfall, and monitoring rainfall proved crucial for communities at risk of flooding during high-intensity rainfall events (Ortega-Gonzalez, et al., 2021). The direct beneficiaries of this project are the MDRRMO organization and the local community. MDRRMO staff benefit from an automated system that provides real-time monitoring and data collection. Barangay officials receive timely alerts during heavy rainfall and critical situations, enabling them to proactively disseminate disaster preparedness information. Accurate and timely prediction of heavy rainfall events was crucial for effective flood risk management and disaster preparedness (Salcedo, 2024).

Conceptual Framework

Figure 1 illustrates the conceptual framework of the proposed Automated Rain Gauge Monitoring System. The researchers adopted the Input-Process-Output (IPO) model

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14 Issue 12 , December - 2025

to clearly represent how various system inputs are transformed through a series of processes into valuable outputs.

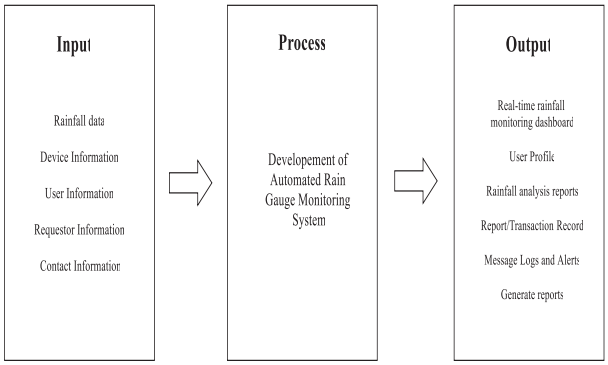


Figure 1. Conceptual Framework`

The input stage of the system consisted of critical data resources necessary for accurate monitoring and efficient disaster response. Rainfall data served as the primary input, collected from the tipping bucket rain gauge, where each tip corresponded to a fixed amount of precipitation. The system recorded the date and time of each tip as timestamps to capture the precise occurrence of rainfall events. Device information included the serial number and device location to identify the sites of installed rain gauges. User information was managed to operate and control the system. Requestor information was recorded to track individuals requesting data, facilitating the generation of reports and maintaining a transaction log of data access requests and contact information of barangay or facility officials was stored to enable timely alerts, disaster preparedness, effective communication, and coordinated responses during critical events.

In the process stage the system receives data from the Tipping Bucket Rain Gauge (TBRG). Each tip corresponds to 0.2 millimeters (mm) of rainfall. For a given time period the system calculates total rainfall by multiplying the number of recorded tips by 0.2 mm.

$$\text{Total Rainfall (mm)} = \text{Number of Tips} \times 0.2.$$

After calculating the total rainfall, the system compares the value against predefined intensity thresholds to classify the rainfall level and corresponding threshold color.

Table 1. Rainfall Thresholds

Rain Type	mm/hr	Threshold color
Light	: 0 – 2.5 mm	
Moderate	2.5 – 7.5 mm	
Heavy	7.5– 15 mm	
Intense	15 – 30 mm	
Torrential	> 35 mm	

Example calculation:

If 25 tips are recorded within 15 minutes:
25 tips × 0.2 mm= 5 mm

Since 5 mm falls within the Moderate range (2.5– 7.5 mm), the rainfall is classified as Moderate.

If 80 tips are recorded within 15 minutes:
80 tips × 0.2 mm = 16 mm

Since 16 mm falls within the heavy range (15–30 mm), the rainfall is classified as heavy.

If 50 tips are recorded within 15 minutes:
50 tips × 0.2 mm = 10 mm

Since 10 mm falls within the intense range (7.5 – 15 mm), the rainfall is classified as Intense.

If 200 tips are recorded within 15 minutes:
200 tips × 0.2 mm = 40 mm

Since 40 mm falls within the Torrential range (>35 mm), the rainfall is classified as Torrential.

Once rainfall intensity is classified, the system generates alerts based on the defined thresholds and notifies staff and administrators. Administrators can then forward alert messages to barangay officials, particularly during torrential levels, to strengthen disaster preparedness.

Additionally, the system securely stores each rainfall tip and its corresponding accumulated rainfall data in a central database. The stored records include details such as the date and time of measurement, the rain gauge location, the total rainfall amount recorded, and the classified intensity level (Light, Moderate, Heavy, Intense, or Torrential).

The system's output phase delivered real-time insights and actionable information to enhance disaster preparedness and response. A real-time rainfall monitoring dashboard displayed live precipitation levels, rainfall intensity classifications, and trend graphs for easy visualization and analysis. The system also maintained user profiles, ensuring secure access and proper role management for administrators and staff. For decision-making and historical reference, the system generated rainfall analysis reports, which included data categorized by date, timestamp, rainfall tips (clicks), accumulated rainfall, intensity level, and device location. To document system usage, it produced reports and transaction records, capturing details of data requests and report generation activities. During critical rainfall conditions, the system issued message logs and alerts, recording the

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14 Issue 12 , December - 2025

notifications sent to barangay officials and their corresponding timestamps. Ensured accountability and traceability in emergency communication. In addition, the system supported automatic report generation, enabling administrators to create structured summaries of rainfall events and system activity for disaster preparedness and future planning.

Objectives of the Study

The objective of this study was to design and develop an Automated Rain Gauge Monitoring System that enabled real-time rainfall data collection, analysis, and automated report generation. This system aimed to enhance rainfall monitoring, disaster preparedness, and response coordination by providing accurate, timely information to the Municipal Disaster Risk Reduction and Management Office (MDRRMO) and emergency response teams.

Specific Objectives

- To design and develop an automated rainfall monitoring system using a Tipping Bucket Rain Gauge (TBRG) device.
- To integrate a web-based system for real-time rainfall data monitoring visualization.
- To automate the collection of rainfall data and store historical data for data analysis and utilization.
- To support decision-making, disaster preparedness, and response efforts through timely and accurate information.

Scope and Limitation

This study focused on the design and development of an Automated Rain Gauge Monitoring System for the Municipal Disaster Risk Reduction and Management Office (MDRRMO) in the municipality of Tagoloan, Misamis Oriental, to enhance rainfall monitoring for disaster preparedness. It utilized a Tipping Bucket Rain Gauge (TBRG) to generate data, which was displayed in a web-based system for visualization and management. The primary users of the system were the admin and staff, with potential benefits extending to the community through improved disaster preparedness.

The study had certain limitations that were beyond the researchers' control. The accuracy of rainfall measurements depended on the proper installation and calibration of the rain gauge sensors, as well as environmental factors such as obstructions, extreme weather conditions, and potential maintenance issues. The system did not provide predictive weather analysis, as it focused solely on monitoring rainfall intensity. Additionally, the system did not automatically send alert messages when rainfall intensity reached heavy or critical levels; instead, it relied on the administrator's instructions or decisions. Despite these limitations, the system aimed to enhance community rainfall monitoring and disaster response capabilities significantly.

Definition of Terms

- Arduino Microcontroller - A programmable electronic board that processed data from sensors and controlled system operations. In this study, it served as the central processing unit that collected rainfall data from the Tipping Bucket Rain Gauge (TBRG) and transmitted it to the monitoring system.
- Internet of Things (IoT) – A network of devices that communicated and shared data over the internet. This study enabled real-time transmission of rainfall data from the monitoring system to the web platform.
- JavaScript - is a high-level, interpreted programming language primarily used to create interactive, dynamic web applications. In this study, JavaScript updates the user interface in real time and displays rainfall data.
- Magnetic sensor - an electronic device that detects and measures changes in magnetic fields. In this study, a magnetic sensor is used to detect movement or changes in the rain gauge mechanism, enabling accurate rainfall measurement.
- GSM Module - It was a communication module that allowed the Arduino microcontroller to send and receive data over mobile networks—in this study, it was used to transmit rainfall data and alerts to authorized personnel via SMS.
- MySQL - It was an open-source relational database management system (RDBMS) that stored and organized data using structured tables. In this study, MySQL was used to store and manage rainfall data collected from the automated rain gauge.
- PHP - It was a server-side scripting language primarily used for web development. In this study, PHP handled backend operations, including processing and storing rainfall data in the database and generating reports and alerts for disaster preparedness.
- Rainfall - The amount of precipitation, in the form of water droplets, fell to the ground over a specific period. In this study, rainfall was measured by the automated rain gauge.
- Rainfall Thresholds – Predefined rainfall intensity categories are used to classify precipitation levels and trigger system alerts. In this study, these include Light (0.1– 2.5 mm/hr), Moderate (2.6–7.5 mm/hr), Heavy (7.6–15 mm/hr), Intense (16–30 mm/hr), and Torrential (>30 mm/hr).
- Tipping Bucket Rain Gauge (TBRG) – A funnel that collected and channeled the precipitation into a small seesaw-like container. In this study, the rain gauge used in the Automated Rain Gauge Monitoring System collected rainfall data and transmitted it for real-time monitoring and analysis.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14 Issue 12 , December - 2025

- Web-based - A system or platform that operates through an internet connection, allowing users to access and manage data remotely. In this study, the term "platform" referred to the platform used to display real-time rainfall data and generate reports.

II. REVIEW OF RELATED LITERATURE

This chapter reviewed systems relevant to the development of the Automated Rain Gauge (ARG) Monitoring System. It examined thirteen relevant studies, emphasizing the methodologies employed, the technologies and tools used, and the results obtained. The reviewed literature was categorized by the type of technology applied, including hardware and software components, methodological approaches, and web-based development technologies. Each study was critically analyzed not only for its contributions but also through comparisons that highlighted limitations and gaps. A synthesis was provided at the end to emphasize the significance and necessity of developing the proposed system.

The Development of the automated rain gauge monitoring system involved both hardware and software components. This section focused on the core technologies that underpinned the proposed system. For hardware, the system used a rain gauge, a GSM communication module, an Arduino microcontroller, and a magnetic sensor to collect and transmit data. On the software side, MySQL handled database management. API for efficient communication and data exchange between the hardware and the backend components. HTML and CSS were used for the front-end to display real-time rainfall data, and JavaScript was used to enhance the system's interactivity. These technologies automated the manual monitoring process. Each component was crucial to the system's functionality, and by evaluating the strengths and weaknesses of these technologies, the most effective approach for system development was determined. Understanding these technologies also helped better integrate them into the monitoring system. The subsequent sections detailed how these components worked together to address the gaps identified in the literature review.

Comparative Review of Related Studies

Megantoro et al. (2021) developed an Internet of Things (IoT) system for monitoring weather, rainfall, and air pollutants, with data displayed via an HTML-based web application. The system transmitted data at set intervals, which limited its suitability for real-time applications that require continuous data streams. Each sensor collected data, converted it to a string, and the web application downloaded and processed it for real-time visualization. The authors stated, "The HTML platform is very suitable to be used for this purpose. The web interface is also made attractive and informative for the users." The system utilized Google Firebase as a real-time database for sensor data, but only stored data temporarily, as each new upload every minute replaced the previous record. Afandi et al. (2024) implemented a rainfall monitoring system using the Aloptama Automatic Rain Gauge (ARG) with contemporary IoT protocols and visualization tools. This configuration enabled real-time monitoring and accurate rainfall data collection, thereby supporting environmental assessment and risk management. The study identified inefficiencies in data synchronization between observation posts and the main office when relying on traditional data management methods. The software stack included XAMPP and MySQL for structured data storage, Node-RED for visual programming, and Grafana for data visualization, all of which demonstrated reliability for rainfall monitoring. Shaheen and Manasra (2022) designed an advanced weather-monitoring system using IoT technology to enable real-time data collection. However, their system was limited to measuring temperature, humidity, and atmospheric pressure, omitting critical variables such as rainfall and wind direction. The system used XAMPP, which included the Apache HTTP Server and MariaDB, simplifying the transition from a local test server to a live environment. Notepad++, a Windows-based code editor that supports multiple files, was also used. A primary limitation was the system's inability to transmit data or notifications, and its web pages required an active internet connection.

These three studies utilized Internet of Things (IoT) technology for real-time weather or rainfall monitoring. However, Afandi et al. (2024), who relied on existing data visualization platforms, faced challenges such as integration issues and limited customization options. Shaheen and Manasra (2022) developed a system that could not transmit data or function without an internet connection. Meanwhile, in the study by Megantoro et al. (2021), data storage was limited to a temporary setup, as each newly uploaded dataset replaced the previous one.

The study by Rahmadani et al. (2023) developed an IoT-based rainfall monitoring system for Chili Farming Land. The study followed a structured development approach starting from the Problem Identification stage, where the challenges faced by chili farmers, such as manual monitoring and the impact of environmental factors, were identified. This was followed by the System Design phase, where a conceptual framework was established, and the Preparation of Materials and Equipment phase in the development stage, where the materials were integrated into a functional system. Testing was then performed to ensure the system operated as intended, providing real-time data and remote monitoring capabilities. Faisal et al. (2023) developed a prototype of a Water Level and Rainfall Detection System as a flood warning solution based on the Blynk IoT application. The system functioned as both a water level and rainfall monitoring tool. It was developed using the tool design method, flowcharts, and system analysis based on the Software Development Life Cycle (SDLC). This approach involved several key steps, including identifying system requirements, designing components, and developing both hardware and software applications. Using these methods, they enabled a clear visual representation of the system's workflow while ensuring optimal functionality and performance in detecting water levels and rainfall for early warning. Fajar et al. (2023) developed an IoT-based rainfall monitoring system using Use Case 2.0. This approach extended the traditional use-case driven development by integrating user stories and agile methodology concepts to improve system maintainability

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14 Issue 12 , December - 2025

and clarity in modeling. The study followed the SDLC process, beginning with system requirements and analysis, and proceeding through design, implementation, and evaluation. The Use Case 2.0 approach served as a guide for each of these stages, particularly in the requirements and analysis phase (modeling). In the evaluation step, the implementation code of selected use cases that had been defined was tested. This approach was successfully applied to model and design an IoT-based rainfall monitoring system. Hu et al. (2023) utilized the Agile methodology in developing a real-time weather data system to support Environment, Social, and Governance (ESG) decision-making. Their project integrated sensors, including Automated Rain Gauge (ARG) stations, and visualized data across multiple platforms. The development process followed six Agile phases: requirements analysis, planning, design, development, testing, and deployment. In the requirements analysis phase, user needs were gathered through surveys and interviews, along with a feasibility study. The planning phase involved defining the project timeline, identifying risks, allocating resources, and estimating the budget. During the design phase, the focus was on creating a user-friendly UI/UX, designing the database schema, and developing UML diagrams. The development phase covered both front-end and back-end implementation, ensuring clean, well-documented code. In the testing phase, functional and non-functional tests were conducted to detect and fix issues. Finally, the deployment phase prepared the system for production, ensuring compatibility with the target platforms.

These four studies employed structured development methodologies to create IoT-based rainfall monitoring systems. The survey of Fajar et al. (2023) focused on applying the Use Case 2.0 approach throughout the Software Development Life Cycle (SDLC) to enhance system maintainability, modularity, and quality. Maddikera et al. (2023) developed a Rainfall and Weather Monitoring System using the Internet of Things (IoT). They used technology such as an Arduino microcontroller for data processing, connected to various sensors, and a Tipping Bucket Rain Gauge (TBRG) for measuring rainfall. The system was energy-efficient, easy to use, and cost-effective. It showed strong potential for real-time applications in monitoring rainfall and weather conditions. Hudiono et al. (2021) developed a telemetering system for rainfall measurement using 433 MHz wireless transmission, employing a rain gauge to monitor rainfall intensity in real time. They utilized a Tipping Bucket Rain Gauge, stating, “A rain gauge is a tool used to measure rainfall. The amount of rain that enters a rain gauge is the rainfall representing the area around the measurement. In their study, a rain gauge with a bucket capable of holding a specific volume of rainfall (in mm) was used. The collected data were displayed through a web-based application and automatically saved in downloadable text tables. The system demonstrated high accuracy with 2% precision and produced outputs containing the date, time, and recorded volume. Mali et al. (2022) developed an IoT-based Tipping Bucket Rain Gauge Data Processing System to measure rainfall intensity. The system used a seesaw-like container that tipped when a preset volume of rainwater was collected. Each tip activated a magnetic sensor (reed switch), sending an electrical signal for data recording. This mechanism enabled classification of rainfall into light, moderate, and heavy categories and supported disaster monitoring via wireless networks. The study also noted that tipping-bucket rain gauges may experience calibration issues, potentially affecting the system’s accuracy. Kennedy et al. (2022) developed and designed an Internet of Things (IoT) rain detector device with GSM notification to address the problem of unexpected rainfall interfering with sun drying and irrigation activities. They used an Arduino microcontroller to process input from a rain sensor module that detected the presence and intensity of rain. Upon detection, the Arduino triggered a GSM module to send real-time SMS alerts to users. The findings showed that the device helped reduce water usage by up to 45% through timely irrigation management, while also enhancing user convenience by sending alerts to retrieve laundry or close windows before rainfall could cause damage. Beltran et al. (2021) developed an Arduino-based disaster management system equipped with multiple sensors, including rain sensors, to detect early signs of natural disasters and send real-time alerts to mobile devices. They utilized the Arduino Uno microcontroller as the central processing unit for the entire system and a GSM module to enable wireless communication for sending SMS alerts to users. The data proved reliable due to the consistency of the results. Their findings revealed the effectiveness and efficacy of the proposed system. Moreover, the study recommended further integration of the Internet of Things (IoT), artificial intelligence, and other innovative technologies to improve. Abella and Enriquez (2024) developed and implemented a flood alert system for Brgy. San Agustin, San Jose, Occidental Mindoro, emphasizing that rainfall is a critical environmental factor that requires close monitoring to mitigate disaster risks. They used an Arduino Uno microcontroller to receive input and process data from a water-level sensor. The GSM (Global System for Mobile Communications) module served as the communication component of the flood alert system, sending real-time SMS alerts to designated recipients, including local authorities and residents. The simulation program on an Arduino Uno demonstrated the system’s capability for real-time water-level detection and alert message sending. The GSM module's performance confirmed its ability to send flood alert messages based on water-level detection.

These three studies utilized Arduino Uno microcontrollers and GSM modules to enable real-time environmental monitoring and SMS alert systems. They shared a common goal of enhancing preparedness and response through automated, sensor-based data collection and real-time communication. Moreover, the integration of the Internet of Things (IoT), artificial intelligence, and innovative technology was recommended for further improvement.

Megantoro et al. (2021), Afandi et al. (2024), and Shaheen and Manasra (2022) utilized IoT technology for real-time weather or rainfall monitoring with online functionalities. Still, they faced challenges, including limited customization and temporary data storage. Rahmadani et al. (2023), Fajar et al. (2023), and Faisal et al. (2023) applied structured development methodologies to create IoT-based rainfall monitoring systems that enhanced system maintainability, modularity, and quality. Maddikera et al. (2023), Hudiono et al. (2021), and Mali

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14 Issue 12 , December - 2025

et al. (2022) used tipping-bucket rain gauges integrated with various sensors to measure rainfall. However, calibration issues may have affected data accuracy, especially in areas with variable rainfall. And the studies of Kennedy et al., Beltran et al., and Abella and Enriquez combined Arduino Uno microcontrollers and GSM modules for environmental monitoring, aiming to improve preparedness through real-time alerts. They used mobile networks and SMS communication for sending alerts, though they similarly relied on network coverage.

Synthesis of the Study

After reviewing the literature, a standard limitation across existing systems was the integration of real-time monitoring without dedicated platforms for data visualization and data storage management. On the hardware side, tipping bucket rain gauges were widely used. However, calibration challenges were reported, resulting in reduced rainfall measurement accuracy.

The proposed study designed and developed an Automated Rain Gauge Monitoring System that integrated hardware and software, using a calibrated Tipping Bucket Rain Gauge (TBRG) to measure rainfall and transmit data via an Arduino microcontroller and a GSM module for continuous operation. The data were displayed on a custom-built web-based platform using HTML for the front end, MySQL for database management, and an HTTP-based API for communication between the hardware and the software, enabling real-time monitoring, analysis, and reporting.

This study adopted the Agile methodology, a flexible and iterative approach suitable for developing both hardware and software systems. The development process was divided into seven key phases: planning, design, development, testing, deployment, review, and launch. Through continuous collaboration and iterative cycles, user requirements were gathered and refined. For hardware components, such as the Automated Rain Gauge (ARG) and sensor integration, Agile enabled incremental testing and adjustments during each sprint. The system was timely and relevant for the MDRRMO as it replaced manual rainfall monitoring with automated, real-time monitoring, data collection, reporting, and analysis. This helped reduce the need for manual intervention to check the rain gauge device outside every 15 minutes. It supported the organization's goal of enhancing disaster preparedness and public safety through accurate and efficient monitoring.

III. METHODOLOGY

The researchers presented the methods used to develop the Automated Rain Gauge Monitoring System using the Agile methodology. The study was divided into phases: planning, design, development, testing, deployment, review, and launch. Agile Methodology for Software Development in IoT projects offered a competitive edge by allowing organizations to adapt quickly to changes while fostering innovation and flexibility (Moedt et al., 2023).

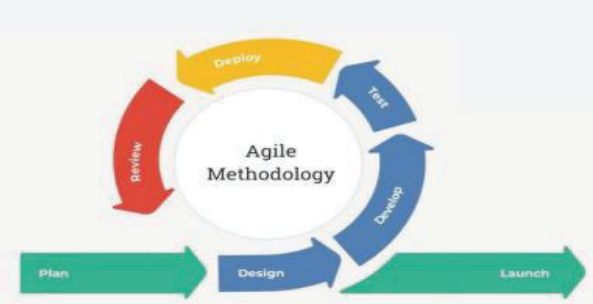


Figure 2 Agile methodology by Hu et al (2023)

Planning

In this phase, the researchers conducted interviews and follow-up discussions with the MDRRMO administrator in Tagoloan, Misamis Oriental, to identify gaps and limitations in the current manual rainfall monitoring process. These included the need for human intervention, manual data logging after device inspection, calculations, and data retrieval for reporting and analysis. Manual rainfall monitoring methods were prone to inaccuracies due to human errors and delays in data collection and recording (Saniel et al. 2023).

The interviews also revealed the hardware technologies in use: a tipping-bucket rain gauge for rainfall collection and a rain gauge display unit for displaying rainfall tips. The researchers defined the core functionalities needed for the project, utilizing both hardware and software components. These included automated data collection using a tipping bucket rain gauge (TBRG) as a key component, a weather instrument used to measure the amount of rainfall. The tipping bucket rain gauge was noted as one of the most direct methods for measuring rainfall (Danendra et al., 2023). Real-time data transmission was planned through a GSM module, a hardware device that allowed systems to send and receive data via SMS networks. Kennedy et al. (2022) found that real-time data transmission could be achieved using a GSM module, which enabled systems to send SMS alerts to users. A web-based dashboard is a dedicated platform used for real-time data visualization, analysis, report generation, and decision-making. A web-based dashboard enabled structured data visualization, report generation, and monitoring to support environmental decision-making. Afandi, et al., (2024)

The expected output of the planning phase was to define the project scope and core requirements of the Automated Rain Gauge Monitoring System. A structured plan was established to integrate hardware and software components for automated data collection and real-time, web-based visualization. This phase aimed to minimize human error, reduce delays, and enhance the accuracy and efficiency of rainfall monitoring. The outcomes of this phase also served as a foundation for guiding the hardware and software architecture design in the succeeding design phase. Fajar et al. (2023) found that functional specifications directly influenced the design phase, guiding the architecture and system structure of the IoT-based rainfall monitoring application.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14 Issue 12 , December - 2025

Design

In this phase, the researchers focused on designing the structure and interaction of both the software and hardware components of the proposed system. The administrator provided comments and feedback about the design, specifically on how the data was visualized in the system and what needed to be included. This was important because the data served as a guide for decision-making by the disaster response teams responsible for disaster preparedness.

A flowchart was used to visualize the hardware and software processes involved in the system's functionality. It presents the system's process flow, illustrating how data collected by the hardware is transmitted to the software and subsequently used by the user. The flowchart provides a clear representation of how data moves through the system and is effectively managed. This tool was essential for representing the flow of operation in embedded systems, especially in illustrating how different components communicated and functioned together (Batutay, 2023).

An Entity Relationship Diagram (ERD) was used as one of the primary tools to model the system's data structure. The ERD provided a visual representation of entities, attributes, and relationships, ensuring the database was correctly organized and normalized. By using the ERD, the researchers were able to translate system requirements into a structured data model, serving as a blueprint for database implementation and future scalability. ERD design builds on methodologies from prior research and is structured to support future system scalability by enabling the easy integration of additional devices and data types. Sobral et al. (2023).

The Use Case Diagram illustrated the interactions among the user, device, and software platform. It showed how the user engaged with the system through key actions. Use case diagrams are a type of Unified Modeling Language (UML) diagram that represent the interaction between actors (users or external systems) and a system under consideration to accomplish specific goals. They provided a high-level overview of the system's functionality by illustrating the various ways users could interact with it (GeeksforGeeks, 2025).

For the database, a Data Flow Diagram (DFD) was developed to model data flow within the Rain Gauge Monitoring System. The DFD illustrates how data flows between processes, data stores, and external entities, providing a clear visualization of the system's functional requirements. This tool ensured that the flow of information was well-structured and served as a guide for the subsequent development and implementation of the system. A Data Flow Diagram is a visual representation of the flow of data within the system. It helps to understand the flow of data throughout the system, from input to output, and how it gets transformed along the way. (GeeksforGeeks, 2025).

A physical database was structured to define how data is stored and managed in the system. It specifies the tables, fields, and relationships necessary for efficient data recording, retrieval, and analysis. A well-designed schema ensures data integrity, minimizes redundancy, and speeds up queries (via proper keys, constraints, and table relationships). Physical database design plays a critical role in database performance and scalability (2023).

The user interface (UI) was designed using Figma, a collaborative design tool for creating interfaces and wireframes. It ensured the system's UI was intuitive, user-friendly, and tailored to users' needs. The Design Thinking approach using Figma significantly improved user experience in disaster management by prioritizing usability and user feedback. Senoro et al. (2021).

The expected output of the design phase was the transformation of conceptual ideas into clear, functional, and scalable system components. By utilizing design tools such as Figma for UI design, flowcharts for process flow, use case diagrams for user interactions, entity-relationship diagrams (ERD) for database structure, data flow diagrams (DFD) for modeling data movement, and a physical database schema for data management, the researchers established a comprehensive technical blueprint. These outputs served as the foundation for guiding the development and implementation of the Rain Gauge Monitoring System.

Development

The development of the Automated Rain Gauge Monitoring System was based on a five-layer architecture the Data Acquisition Layer, Networking Layer, Presentation Layer, Application Layer, and Database Layer. This layered approach ensured clear separation of responsibilities, enhancing the system's scalability, reliability, and maintainability. Each layer was designed to fulfill a specific role in collecting, transmitting, processing, storing, and presenting rainfall data to support the Municipal Disaster Risk Reduction and Management Office. To build the prototype of the Rainfall Monitoring System, we first connected the tipping bucket rain gauge to the Arduino Uno, ensuring the magnetic sensor accurately detected each tip. The Arduino processed these signals and prepared the data for transmission. Next, we integrated the SIM-enabled GSM module to send rainfall data over mobile networks using HTTP requests, linking the hardware to the backend API. On the software side, we developed a dynamic web interface using HTML, CSS, JavaScript, React.js, and supporting libraries. Leaflet.js for visualizations and maps, and Alpine.js for toggles, dropdowns, and modals, enhancing user interaction without overcomplicating the code. Laravel handled the core application logic, while MySQL stored all the rainfall records, user data, and logs. Throughout the process, we tested and adjusted both hardware and software components to ensure reliable real-time monitoring and data visualization, resulting in a fully functional prototype that could accurately track rainfall and present it in an interactive dashboard.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14 Issue 12 , December - 2025

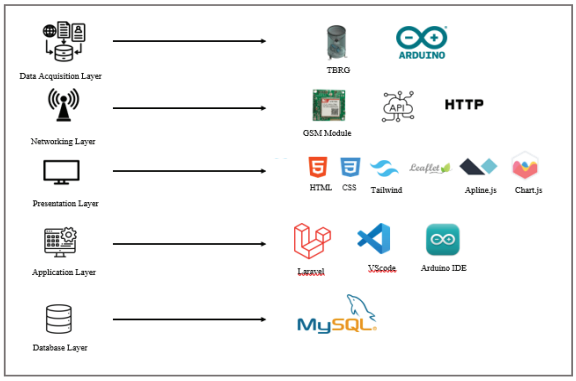


Figure 3. System Architecture Design

The Data Acquisition Layer consisted of a tipping bucket rain gauge (Figure 4) connected to an Arduino Uno microcontroller (Figure 5). This mechanism operated by tipping a small bucket once a specific amount of rainwater accumulated, which was then detected by a magnetic sensor. Each tip represented a measurable volume of rainfall, and the Arduino processed these signals to quantify the data. The microcontroller served as the core component for capturing accurate rainfall measurements in real time. As described by Hudiono et al. (2021), microcontroller-based systems like this are practical tools for environmental monitoring.

Figure 4 Tipping Bucket Rain Gauge

The figure shows the Tipping Bucket Rain Gauge (TBRG) used in the system as part of the Data Acquisition Layer**.** It serves as the primary sensor for collecting rainfall data, with each bucket tip corresponding to a measured rainfall amount. The TBRG provides accurate, real-time raw data, which is then transmitted to the microcontroller for further processing and integration into the monitoring system.

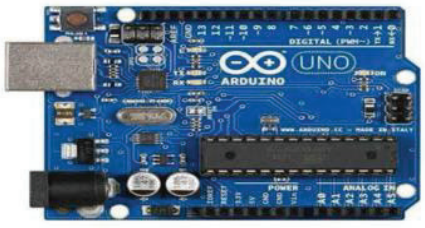


Figure 5 Arduino Uno Microcontroller

This figure shows the Arduino Uno microcontroller that processes signals from the rain gauge. It functions as the central processing unit of the Data Acquisition Layer, converting each tip into a digital rainfall record and transmitting the data to the successive layers of the system.

Once the data was captured, it was transmitted through the Networking Layer. As shown in Figure 6.0, a SIM-enabled GSM module powered by 4G and equipped with GPS was used to send rainfall data over mobile networks. The data was sent via an HTTP GET request to the API, ensuring standardized, secure communication between the hardware and the backend system. Through this process, the device transmitted rainfall values, timestamps, and device identifiers directly to the server for processing and storage. According to Agulto et al. (2022), an API enables both data transmission and access in a structured and efficient manner.

Figure 6. GSM Module

GSM module used in the system. It is powered by 4G and equipped with GPS. Its primary purpose is to transmit rainfall data from the Arduino to the server.

The Presentation Layer was responsible for the user interface. Built using HTML, CSS, and JavaScript. Tailwind CSS was used for a modern, consistent design, enabling visually appealing layouts with minimal custom CSS, providing a clear visualization of rainfall data over time. Leaflet.js, an open-source mapping library, enabled the display of geographic locations and rainfall measurements on interactive maps, making it easier to understand spatial patterns. Additionally, Alpine.js was used to implement interactive elements such as toggles, dropdowns, and modals, enhancing user interaction without overcomplicating the code. Together, these modern web technologies created an effective weather-monitoring dashboard that enabled users to monitor rainfall data in real time, view historical records, and generate meaningful visualizations, thereby improving user engagement, accessibility, and overall usability (Sharma et al., 2025).

Laravel powered the Application Layer. This layer handled the core business logic, including processing incoming API data, managing user authentication, generating reports, and integrating with the database. Laravel acted as the central coordinator, linking the frontend with the backend services. The system's code was developed using VSCode for Laravel and other backend/frontend components. In contrast, the Arduino code for data collection and sensor management was written and uploaded using the Arduino IDE. As noted by Shaheen and Manasra (2022), PHP frameworks such as Laravel are reliable for developing scalable, data-driven applications.

The Database Layer was implemented using MySQL. This relational database stored all rainfall data, user

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14 Issue 12 , December - 2025

information, and system logs. It played a crucial role in ensuring data persistence, integrity, and security. The structured nature of MySQL facilitated efficient data retrieval, supported historical analysis, and enabled the generation of comprehensive reports for disaster risk management and planning.

Testing

In this phase, the researchers focused on evaluating the system's performance, functionality, and reliability to ensure it met the specified objectives. The system underwent continuous, iterative testing, including comprehensive checks of all hardware and software components. This included functional evaluations, which verified that the system's features and operations, such as accurate rainfall measurement and data transmission, worked correctly in accordance with requirements, and non-functional assessments, which evaluated qualities such as performance, reliability, usability, and the system's behavior under real-world conditions. When issues were identified, the process returned to the development phase. Testing continued until the system operated as intended. This approach followed industry best practices, as Sharma and Sarje (2023) emphasized that testing was crucial for ensuring quality, reliability, and security in complex systems such as IoT applications.

Unit testing focused on verifying the individual components of the system to ensure they functioned as expected, including both hardware and software. Strandberg et al. (2020) and Cheddadi et al. (2022) found that unit testing played a critical role in verifying individual hardware and software components before system integration. This approach ensured that each part functioned correctly and helped reduce the risk of failures during later stages of development, especially in embedded systems.

By testing each component, the researchers confirmed that every part of the system worked correctly before integrating them into the complete system, minimizing the risk of issues later in the development process.

Integration testing ensured that the system's hardware and software components worked together. This included testing how the rain gauge communicated with the magnetic sensor, how the Arduino microcontroller processed the sensor data, and how the GSM module transmitted the data to the central server. Integration testing was critical to make sure that data flowed correctly between all components and that the entire system functioned as intended when all parts were connected.

Usability testing assessed how easy and intuitive the system was for MDRRMO personnel who interacted with it. Purnomo et al. (2020) emphasized that a sound information system should be easy for users to use and that its usefulness should be measured to ensure it meets user needs. The testing focused on ensuring that the interface was easy to navigate and that the displayed data was clear and understandable, allowing personnel to quickly interpret rainfall information, generate reports, and make informed decisions without difficulty. In addition to observational usability testing, the researchers used the System Usability Scale (SUS), a standardized questionnaire that assesses system usability from the user's perspective. The SUS consists of 10 items rated on a 5-point Likert scale that evaluate factors such as ease of use, interface consistency, and user confidence. This method provided a reliable, quantitative measure of usability, offering valuable insights into user satisfaction, acceptance, and areas for improvement (Lestari et al., 2023).

Acceptance testing verified that the system met the overall requirements and expectations of the MDRRMO unit, including functionality, performance, and usability. This ensured the solution aligned with stakeholder needs before deployment. Tong et al. (2022) described acceptance testing as a process of evaluating a system against predefined criteria to confirm it meets the client's requirements.

The expected output of the testing phase was a system that had undergone rigorous and comprehensive testing to ensure each component functioned as intended. This included verifying the system's functionality, reliability, performance, and integration under real-world conditions, ensuring it was fully ready for deployment.

Deployment

Acceptance testing verified that the system met the overall requirements and expectations of the MDRRMO unit, including functionality, performance, and usability. This ensured the solution aligned with stakeholder needs before deployment. Tong et al. (2022) described acceptance testing as a process of evaluating a system against predefined criteria to confirm it meets the client's requirements.

The expected output of the testing phase was a system that had undergone rigorous and comprehensive testing to ensure each component functioned as intended. This included verifying the system's functionality, reliability, performance, and integration under real-world conditions, ensuring it was fully ready for deployment.

The back-end was deployed on a live server so users could access the data anytime, anywhere with an internet connection. Using a live server made the system more reliable, with real-time updates and centralized data management. Authorized personnel could easily check rainfall statistics, generate reports, and monitor rainfall intensity in the area through a simple web interface. This setup made the system more accessible, efficient, and practical for the MDRRMO and other users who rely on timely weather information.

To support the transition, orientation and training sessions were conducted for MDRRMO personnel. These sessions introduced the technical components of the system, including how the rain gauge operated, how data was transmitted, and how to interpret the results displayed on the interface. Personnel were trained to access the server, generate reports, manage alerts, and maintain field hardware units. The training ensured that staff could confidently operate and manage the system.

Additionally, a user manual was provided as a reference for the MDRRMO team. This manual included step-by-step guides on operating the system, troubleshooting fundamental issues, and contacting the support team for technical assistance. It also contained diagrams of the

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14 Issue 12 , December - 2025

hardware setup, system functions, and explanations of the rainfall measurement process.

The expected outcome was the successful deployment of the system, with all hardware components installed and working correctly in the field. The system consistently collected and transmitted real-time rainfall data for monitoring and analysis. MDRRMO staff were able to access the data, track rainfall patterns, and generate reports through the back-end system. Overall, the system operated smoothly, providing reliable support for disaster risk management efforts.

Review and Launch

In this phase, the researchers evaluated the system's performance and reliability to determine whether the initial planning and development goals had been met. The assessment focused on hardware and software efficiency, data accuracy, system usability, and compliance with the MDRRMO's specific requirements. Tests confirmed that each component from the tipping bucket and magnetic sensor to the GSM module functioned as intended. Feedback from MDRRMO personnel was also gathered to ensure that the system's features, interface, and data visualizations were practical and met operational needs. Insights from this review guided future improvements and refinements, preparing the system for full-scale deployment. After completing this phase, the MDRRMO deemed the system ready for operational use.

The expected outcome was a fully functional system, enhanced by practical feedback and strategies to ensure its reliability, adaptability, and impact in real disaster management scenarios. Systems Engineering Technical Reviews and Audits (2020) emphasized the importance of conducting technical reviews at various stages of system development to evaluate significant achievements and assess technical maturity and risk.

IV. RESULTS AND DISCUSSION

This chapter presents the information gathered and documented in alignment with the research, following the software development life cycle guided by the Agile methodology**.** In the Planning phase, the researchers prepared a formal letter and a guided questionnaire for client interviews, identifying existing problems to establish a clear understanding of the client's needs and challenges. During the Design phase, appropriate tools were utilized to visualize the system processes and develop the user interface, providing a clear blueprint for implementation. Lastly, the Development phase focused on the actual creation and integration of system components, transforming the design into a functional Monitoring System.

Planning

The first step undertaken involved drafting a formal letter addressed to MDDRMO office, requesting access to relevant documents and permission to conduct interviews with designated personnel. A copy of the approved letter is included in Appendix A, while the guide questions are provided in Appendix B. This authorization granted official clearance for the researchers to collect data, ensuring that all information was obtained ethically and legitimately. Furthermore, the approved letter conferred credibility to the project and established a structured and transparent procedure for accessing the necessary records.

The second activity involved conducting interviews with administrator to gain a deeper understanding of the current manual monitoring process. During the interviews, the staff described the various device components used for data collection, including the tipping bucket rain gauge and display unit, as detailed in Appendix C.

Figure 7 illustrates the manual process of rainfall monitoring using a Swimlane diagram, detailing the sequence of tasks performed from data collection to reporting.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14 Issue 12 , December - 2025

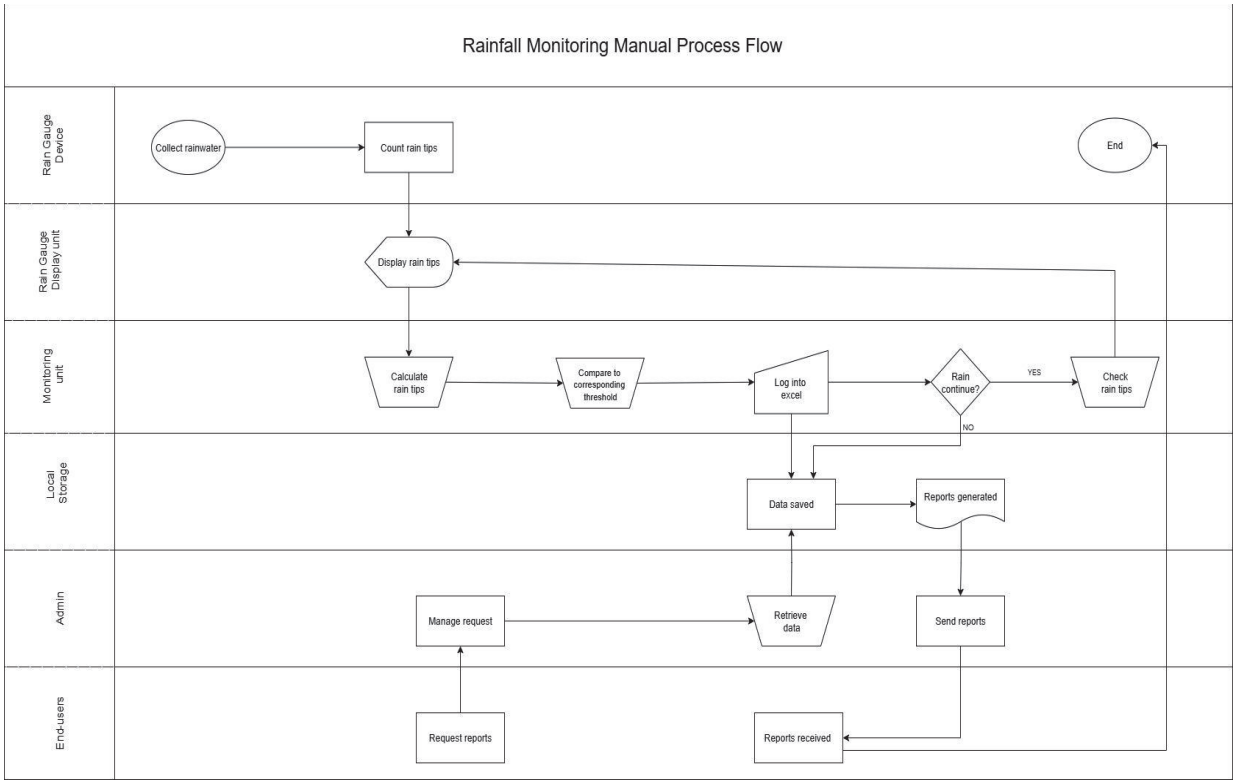


Figure 7 Swimlane Diagram of the Manual Monitoring Process

The process begins when the rain gauge device collects rainwater in its container or tipping bucket. Each time the bucket fills, it tips over, triggering a sensor that counts the tip. Every tip corresponds to a fixed amount of rainfall.

The rain gauge records rainfall in fixed amounts, and each measurement is displayed on the rain gauge display unit. This display allows users to visualize the rainfall tips and manually monitor the rain level at regular intervals, typically every 15 minutes.

After checking the rain gauge display unit, the fixed amount of rain per tip is set to 0.2 mm, and the corresponding thresholds are used to determine the rain intensity, such as light, moderate, intense, heavy, or torrential. Once the accumulated rainfall and its corresponding threshold are determined, the data is logged into a Microsoft Excel spreadsheet. If the rain is still falling, the process loops back, requiring staff to wait for the next interval and recheck the device's display unit before repeating the measurement.

After the data is logged into Microsoft Excel, it is stored in internal storage for analysis and report generation, allowing staff to track rainfall trends, evaluate rainfall intensity, and produce reports for monitoring and decision-making.

When end users request rainfall history records, the admin reviews and approves the request before manually retrieving and compiling the relevant data from internal storage. Rainfall records are generated from stored measurements and their corresponding thresholds. Once the records are prepared, they are sent to the end users who requested the data.

The reliance on manual checking and recording introduces risks of human error in both calculation and data entry. Additionally, the need for staff to physically monitor the device and update records delays the information, reducing its accuracy and timeliness. These inefficiencies highlight the need for an automated rainfall monitoring system that minimizes human intervention, ensures real-time accuracy, automates data collection, and improves decision-making.

Design

This phase presents the design of the system, which outlines the overall process flow and structure of the proposed solution. The design is described using multiple modeling tools that collectively illustrate the functional, logical, and physical aspects of the system.

Figure 8,9 and 10 presents the system process flow from hardware to software, illustrating how data collected by the hardware is transmitted to the software and subsequently utilized by the user**.** It provides a clear visualization of how data moves and is managed within the system.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14 Issue 12 , December - 2025

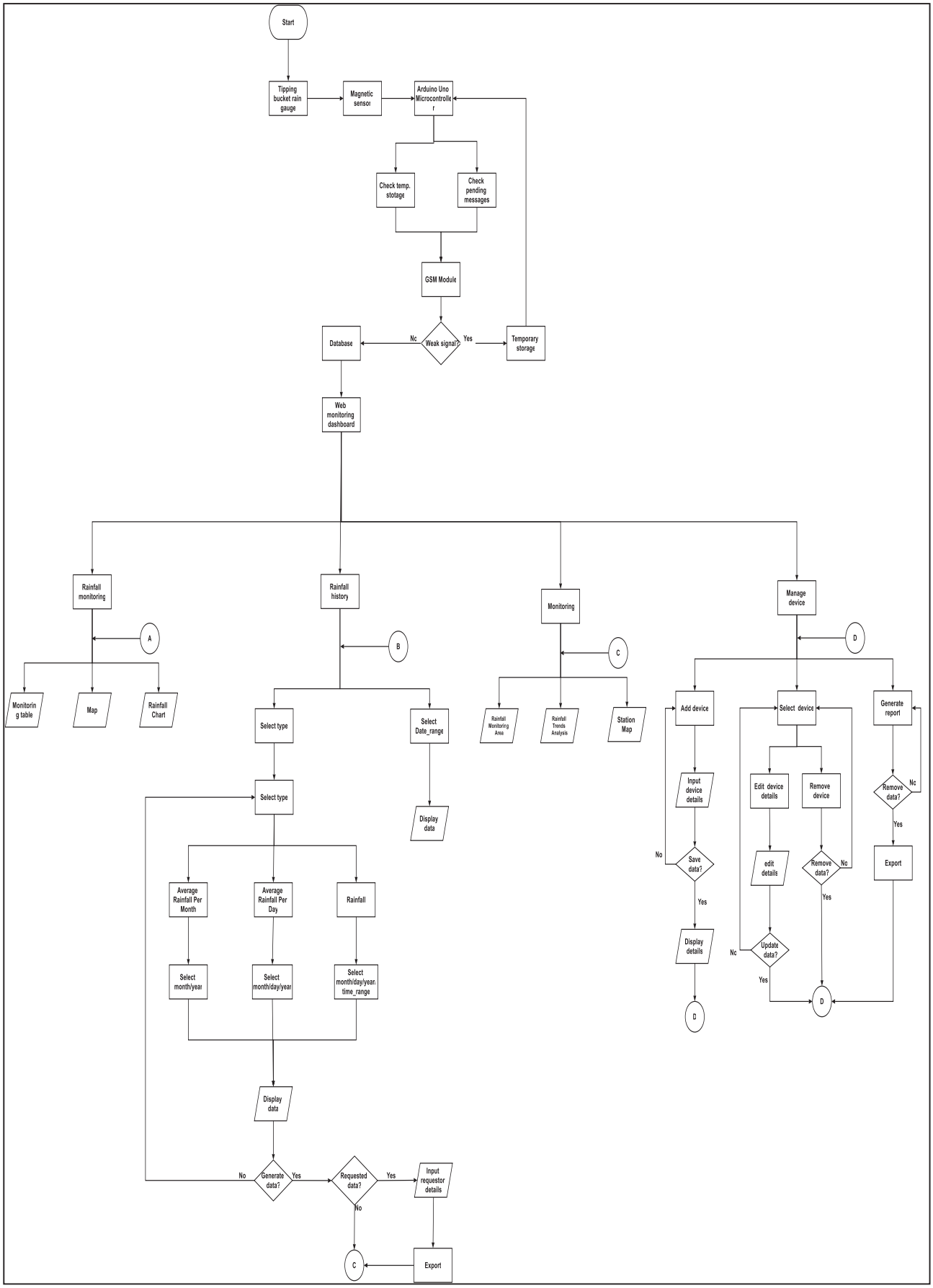


Figure 8 System process flow

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14 Issue 12 , December - 2025

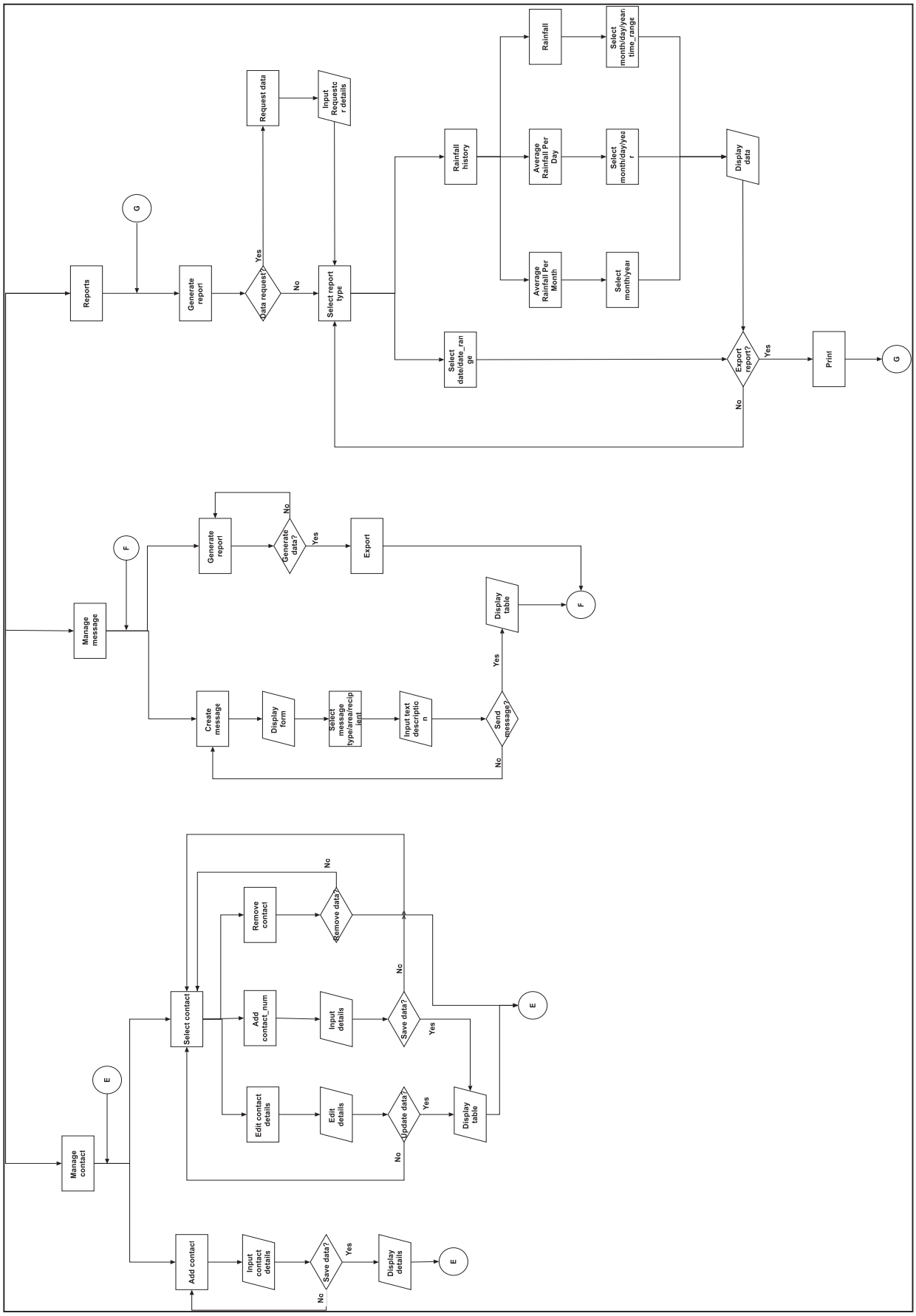


Figure 9 System process flow

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14 Issue 12 , December - 2025

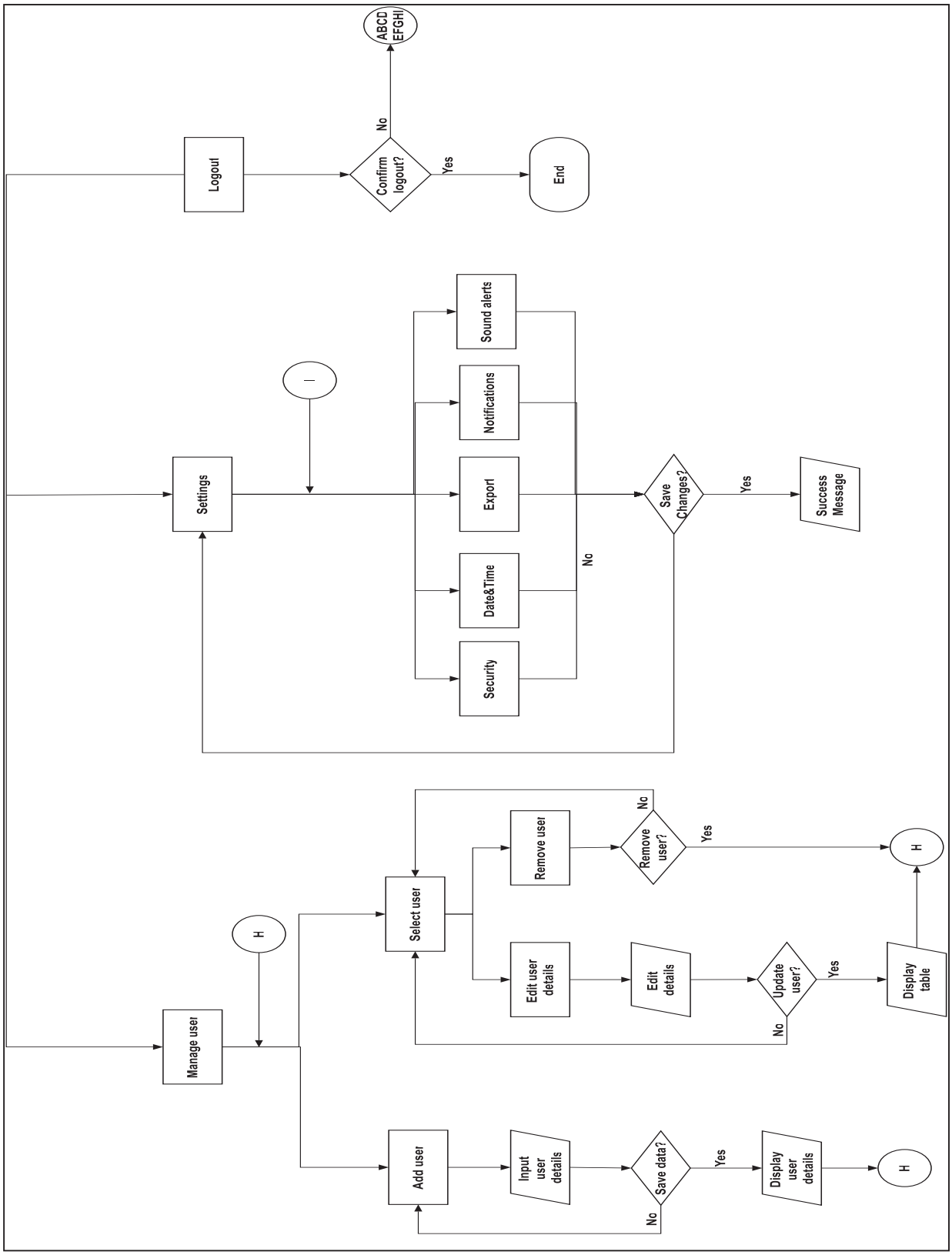


Figure 10 System process flow

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14 Issue 12 , December - 2025

The process starts with the Tipping Bucket Rain Gauge (TBRG), which measures rainfall by collecting water in a small bucket that tips once it fills to a specific volume. This tipping action is detected by a magnetic sensor, which sends signals to the Arduino Uno Microcontroller. The Arduino Uno acts as the main controller, processing the rainfall data.

Then checks the temporary storage to determine if there is any data waiting for transmission. The processed data is then forwarded to the GSM module for wireless transmission, where the system evaluates the GSM signal strength. If the signal is strong, the data is immediately sent to the Database for permanent storage. If the signal is weak, the data is stored temporarily to prevent loss. This mechanism ensures that data is preserved until a reliable connection is available, at which point it can be synchronized with the database. The process of checking temporary storage guarantees that previously unsent data is also retried and successfully transmitted once a stable connection is established.

The Check Pending Messages process allows the Arduino Uno microcontroller to verify if there are any messages composed by the admin through the web interface that are waiting to be sent. Once detected, the microcontroller forwards these messages to the GSM module for transmission to the designated contact. This ensures that communication through SMS notifications remains active and responsive, even when no new rainfall data is being processed. By separating message management from rainfall data handling, the system can efficiently manage both monitoring and communication functions simultaneously.

The Rainfall Monitoring module serves as the main feature of the system, allowing users to view and analyze real-time rainfall data efficiently. From the dashboard, users can access multiple functionalities that support informed decision-making and enhance disaster preparedness. Within this module, the system is composed of three primary components: The Monitoring Table, Map, and Rainfall Chart.

The Monitoring Table presents detailed rainfall information such as the date, time, and rainfall intensity, enabling accurate tracking and recording of data over time. The Map provides a geographical visualization of rainfall distribution and device locations, helping users identify which monitoring stations are currently collecting rainfall data. Meanwhile, the Rainfall Chart illustrates rainfall trends and patterns over a specific period through graphical representation, making it easier to interpret changes in rainfall intensity.

The Rainfall History Module allows users to access, view, and analyze recorded rainfall data in both tabular and graphical formats. The process begins when the user selects the desired type of rainfall information either Average Rainfall per Month, Average Rainfall per Day, *or* Rainfall and defines the date range or time frame for the data to be retrieved. Once the parameters are set, the system displays the rainfall data in a structured table format, allowing users to examine historical values in detail. For users who wish to analyze rainfall patterns and trends more visually, the system also provides an option to generate charts or graphs that illustrate variations over time. This dual representation tabular for detailed examination and graphical for trend analysis enables users to better understand rainfall behavior and make informed interpretations or decisions based on historical data.

The Monitoring module allow disaster response team to view clear picture of how rainfall conditions are changing. Users can focus on specific areas of interest and see detailed station maps, so they know exactly where data is coming from.

The Manage Device function enables system administrators to oversee connected devices. They can add a new device by inputting serial number and specific location, select an existing device to view or edit its specific location, or delete devices that are no longer in use. Each operation updates the database to keep records accurate. Additionally, this includes a Generate Report, allowing administrators to create and download device related reports for monitoring and documentation purposes.

The Manage Contact feature ensures that warning message are delivered to the right individuals efficiently. Users can add new contact by entering their details, including the ability to store multiple contact numbers for a single contact. They can also select contacts to view or update information, such as editing names, positions, or adding additional numbers as needed. If a contact is no longer part of the alert network, their record can be remove. And generate reports on stored contacts, making it easier for administrators to review, track, and document contact information.

The Manage User provides administrative control over who has access to the system. Administrators can add new users, input and save their details, select and update existing users, or remove users if access is no longer required. This supports security and accountability.

Staff members do not have access to the Manage User function to ensure proper accountability, and controlled system access. This restriction helps maintain the integrity of the system by preventing unauthorized modifications to user records.

Manage Message allows authorized users to create and send message. Users can create a new message, input the content, and then send the message to registered contacts. This functionality ensures that timely rainfall warnings or notifications are disseminated to the right stakeholders.

The Generate Reports module enables users to produce both comprehensive and specific rainfall reports. When a data request is made, the system prompts the user to input

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14 Issue 12 , December - 2025

their requester details before proceeding with the report generation. If no data request is made, the user can still continue by selecting the type of report they wish to generate.

Once the report type is selected, users have several options depending on their monitoring needs. They can select a specific date or date range to generate rainfall data within a defined period, view rainfall history for long-term analysis, or obtain average rainfall reports either per month or per day by specifying the desired month, day, or year. Additionally, users can generate a rainfall report for a particular time range to observe short-term rainfall patterns.

The Settings module allows users to configure and personalize various aspects of the system according to their preferences. Within this module, users can navigate through several configuration options, including Security, Date & Time, Export, Notifications, and Sound Alerts.

The Security setting enables users to manage account privacy and password updates to ensure system protection. The Date & Time setting allows users to adjust formats and time zones based on their local settings. The Export section provides options for managing report formats and data export preferences. Meanwhile, the Notifications setting lets users control alerts, reminders, and updates, while the Sound Alerts option allows customization of volume levels, and sound intervals.

Once all preferences are set, users can click Save Changes to apply the modifications. Upon successful saving, a Success Message is displayed, confirming that all configurations have been updated.

The Logout Process allows users to securely end their current session in the system. When the user selects the Logout option, the system triggers a confirmation prompt asking whether the user truly intends to log out. If the user confirms, the system immediately terminates the active session and returns to the login page, ensuring that no unauthorized access can occur after the user leaves. However, if the user cancels the logout request, the process is aborted and the user is redirected back to the previously active page, allowing them to continue their activities without interruption.

The Figure 11 below represents the hardware process flow inside the tipping bucket rain gauge. It shows how rainfall is captured, measured, and converted into data signals before being processed by the system.

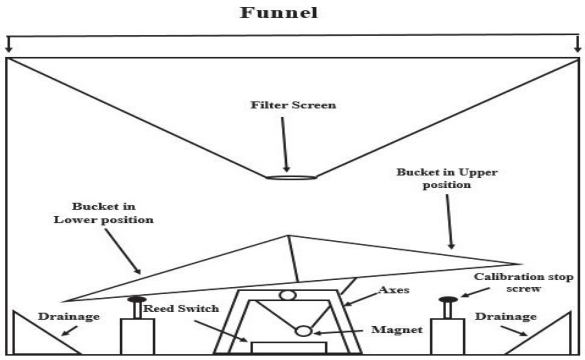


Figure 11 Process Flow inside the tipping bucket rain gauge

Rainwater first enters through the funnel, which directs it toward the filter screen. The filter ensures that only clean water flows into the measuring system, preventing clogging or errors caused by debris. The filtered water drips into the tipping bucket mechanism, which has two small buckets mounted on an axis. As one bucket fills to its calibrated capacity, it becomes heavy enough to tip, moving into the lower position while the opposite bucket shifts into the upper position to continue collecting rainfall. The water from the tipped bucket drains out through the drainage outlets, ensuring the system resets for the next measurement. During the tipping action, the magnet attached to the bucket assembly passes close to the reed switch, generating a magnetic pulse. Each pulse corresponds to a fixed rainfall volume, defined by the calibration stop screw, which ensures that tipping occurs only after the exact volume is reached. These pulses are then sent to the processing unit for conversion into rainfall data. This internal mechanism shown in the figure ensures a repetitive and accurate cycle of rainfall measurement, forming the first stage of the hardware-to-software rainfall monitoring system.

Figure 12 Presents the Entity Relationship Diagram of the Rain Gauge Monitoring System. The diagram consists of four major processes, each representing a critical activity in the system's workflow.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14 Issue 12 , December - 2025

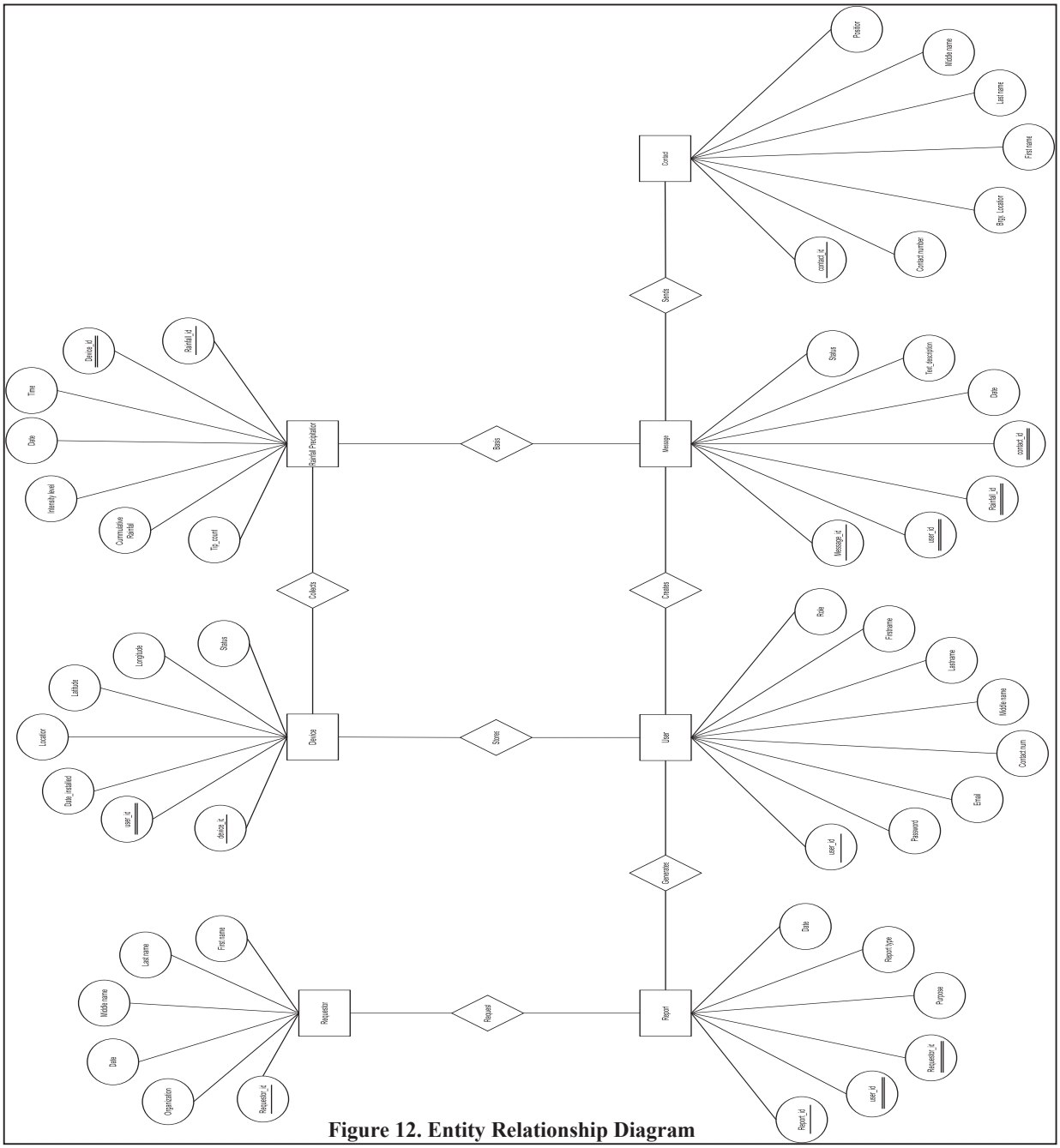


Figure 12. Entity Relationship Diagram

Each Device can generate multiple rainfall records, but every rainfall entry belongs to only one device. This establishes a one-to-many (1:M) relationship between the Device and Rainfall entities. The device, identified by Device_ID, represents the physical rainfall monitoring unit equipped with a tipping mechanism to capture rainfall data. This setup reflects real-world operations, where a single device continuously records several rainfall measurements during different rain events, ensuring accurate and continuous data collection.

Each User can create multiple messages, but every message is linked to only one user. This defines a one-to-many (1:M) relationship between the User and Message entities. The User entity, identified by User_ID, represents authorized personnel such as system operators or administrators who manage the rainfall monitoring platform. The Message entity, identified by Message_ID, stores the alerts or notifications that users create based on rainfall data. This structure establishes accountability by ensuring that each alert can be traced back to the user who generated it, while allowing one user to create multiple alerts over time.

Each Message is generated from exactly one rainfall record, but a single rainfall entry can generate multiple messages. This forms a one-to-many (1:M) relationship between the Rainfall and Message entities. It mirrors real-world monitoring conditions where a single rainfall observation can trigger several alert levels such as initial, warning, and critical messages. This relationship ensures that each alert has a verified rainfall source while enabling multiple alerts to be associated with one rainfall event as conditions change.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14 Issue 12 , December - 2025

One Message can be sent to multiple contacts, but every contact entry is tied to only one message at a time. This establishes a one-to-many (1:M) relationship between the Message and Contact entities. The Message, identified by Message_ID, represents the alert or notification generated from rainfall data, while the Contact entity, identified by Contact_ID, represents the recipients who receive these messages. This structure reflects the real-world process of information dissemination, where a single alert is broadcasted to several recipients to ensure timely communication and community safetyEach User can generate multiple reports, but every report is associated with only one user. This defines a one-to-many (1:M) relationship between the User and Report entities. The User, identified by User_ID, represents staff or administrators who are responsible for creating reports from the system's collected data. The Report, identified by Report_ID, includes details such as report type, requestor, purpose, and creation date. This relationship ensures that every generated report is properly attributed to the user who created it, promoting data traceability and accountability.

Each Requestor, identified by Requestor_ID, can produce multiple reports, and at the same time, each report can satisfy multiple requests. This establishes a many-to-many (M:M) relationship between the Requestor and Report entities. The Requestor represents individuals or organizations requesting rainfall data or analytical reports. This relationship captures real-world reporting scenarios, where one request may require multiple reports for analysis, and standardized reports may be reused to fulfill several different requests.

Figure 13 presents the Use Case Diagram of the system. The diagram highlights the interaction between the User

and the Device, showing how users access, monitor, and manage rainfall data through the system.

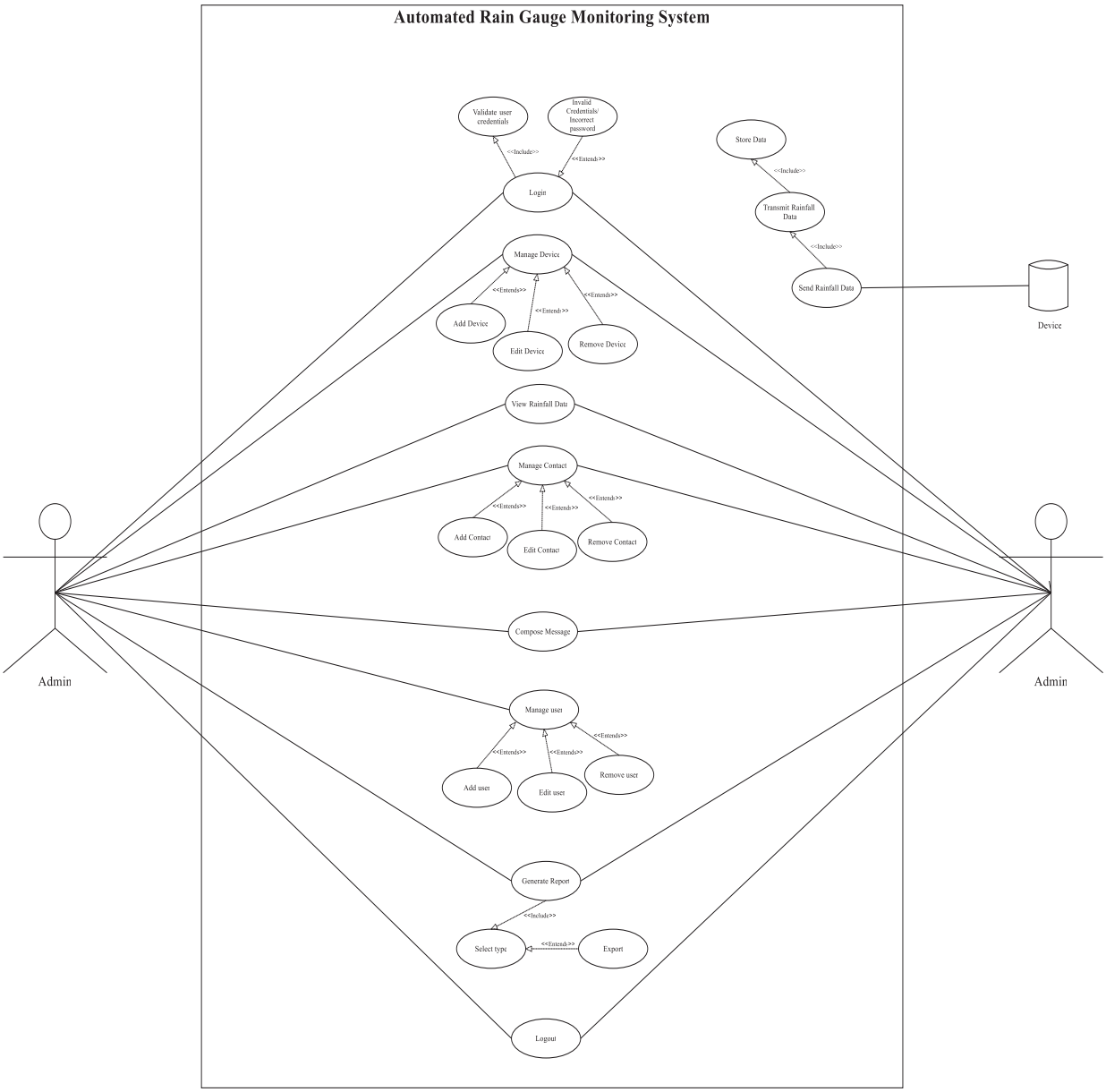


Figure 13 Use Case Diagram

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14 Issue 12 , December - 2025

User

The User actor represents the system administrator who has the highest level of access. The administrator logs into the system using valid credentials, ensuring secure authentication and preventing unauthorized entry. Once inside, the administrator can manage devices by adding, updating, or removing their details, ensuring that only operational and properly installed devices are active in the system. The administrator also has full access to view real-time rainfall data, which supports decision-making during rainfall events. Additionally, they can manage contact by registering, updating, or removing contact details, ensuring that rainfall alerts reach the correct stakeholders. Administrators can also compose messages based on rainfall data and send alerts to contact, guaranteeing timely and reliable communication. They have the authority to generate reports on rainfall history, device performance, and message information, which assists in documentation and evaluation. Importantly, administrators alone can manage user accounts, adding, editing, or removing access rights, which secures the system and ensures accountability. Finally, they logout to end their session securely, preventing unauthorized access. Compared to manual systems, these capabilities enhance efficiency, accuracy, and security.

Staff

The Staff actor represents regular users of the system who perform operational tasks but do not have the authority to manage other users. Staff log in securely using their credentials, protecting the system from unauthorized entry. They can view rainfall data collected by devices, providing timely and accurate updates without needing to manually check gauges. Staff can also manage devices by adding, updating, or removing device details, ensuring accurate monitoring records. They are responsible for managing contact, which involves adding, updating, or removing contact information to keep contact lists accurate and updated. Staff are also involved in composing messages based on rainfall data, customizing alerts depending on rainfall intensity or event status, and then sending those alerts to multiple contact. Beyond communication, staff can generate reports on rainfall events, message histories, and device data, as well as analyze rainfall trends by selecting specific date ranges. These functions support decision-making and planning for rainfall-related risks. Lastly, staff securely logout of the system to protect ongoing operations. Unlike administrators, staff cannot manage system users, which ensures stricter access control and maintains system integrity.

Device

The device automatically sends rainfall measurements to the system at regular intervals. This ensures continuous real-time monitoring. In the manual method, staff had to physically visit the rain gauge and write down readings, delaying data collection. The system stores all transmitted rainfall data securely in its database. This ensures that information is preserved for future reference, analysis, and reporting. The manual equivalent was maintaining handwritten logs or spreadsheets, which were harder to organize and more prone to loss.

Figure 14 presents the Data Flow Diagram (DFD) of the Monitoring System, which illustrates how data flows through the different processes.

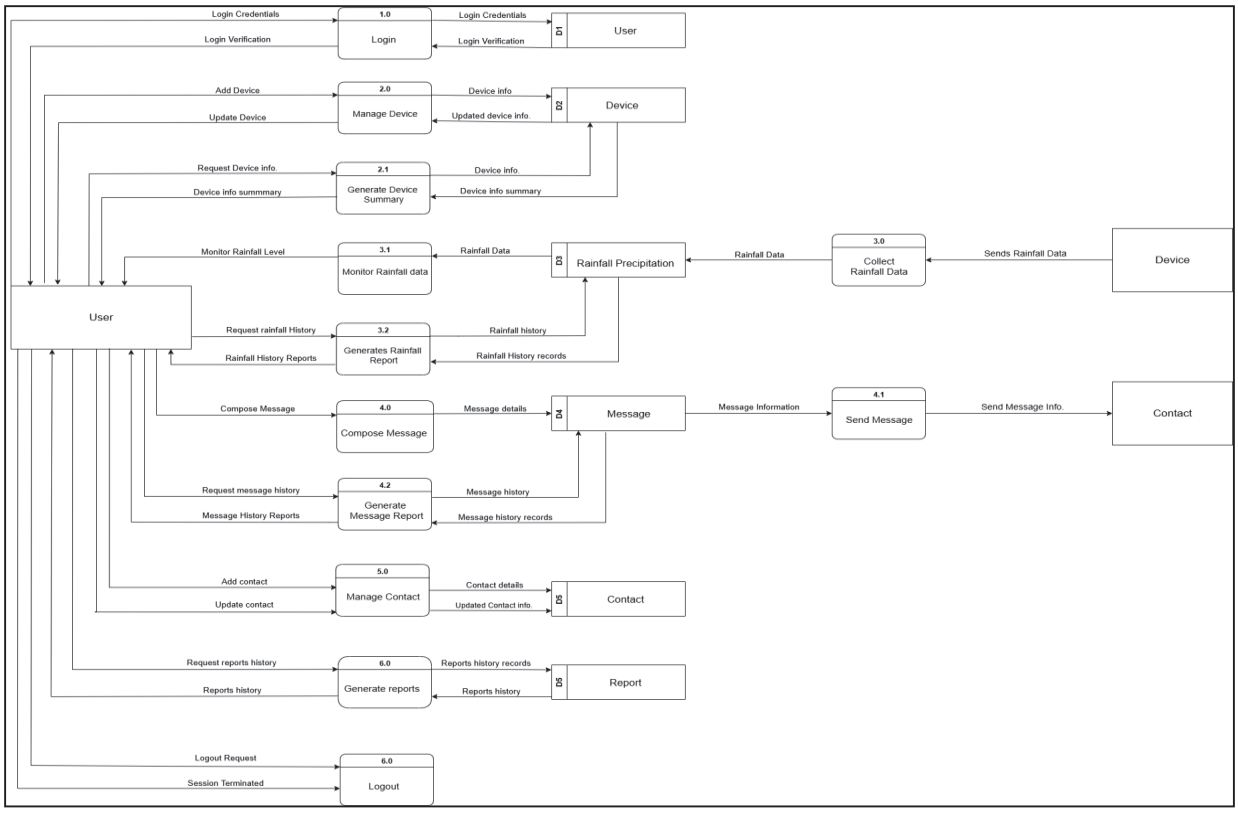


Figure 14 Data Flow Diagram

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14 Issue 12 , December - 2025

The user initiates the login process by entering their login credentials their username and password. The system verifies these credentials against stored data to confirm the user's identity. Upon successful verification, the user gains access to the system; if verification fails, access is denied, and the user is prompted to re-enter valid credentials. This ensures that only authorized users can access the platform.

Users can add or update devices within the system, with submitted device information either creating a new record or updating existing details. This information is stored in the Device Data Store, ensuring accurate and up-to-date records. The Generate Device Info process is when the user requests a device summary. The system then retrieves the stored data from the Device Data Store and produces a summarized report, which is sent back to the user. This setup allows users to efficiently manage and monitor all connected devices while eliminating the need for manual record-keeping.

The system automatically collects rainfall data from connected devices and records it in the rainfall precipitation data store, capturing details such as device ID, location, number of rain tips, and timestamp. In addition to storing raw values, the system also calculates the equivalent rainfall amount by multiplying the number of rain tips by 0.2 mm and determines the corresponding intensity level (light, moderate, heavy, intense, or torrential). This structured recording ensures that both raw measurements and meaningful classifications are available, supporting accurate monitoring, trend analysis, and timely decision-making.

The system continuously monitors incoming rainfall data from connected devices and makes the latest readings available to users in real time. Each record includes the device ID, location, timestamp, and equivalent rainfall amount with its corresponding intensity level. By instantly displaying rainfall levels as they are collected, the system provides users with immediate awareness of current conditions, enabling quick response to potential hazards. This automation eliminates the need for staff to manually check gauges at regular intervals, reducing delays and ensuring accurate, up-to-date information is always accessible.

When users request rainfall history, the system retrieves stored data from the rainfall precipitation database and processes it to generate summaries and detailed reports. These records may include daily, weekly, or monthly rainfall totals, as well as intensity patterns over time. By organizing historical information, the system allows users to analyze frequency, distribution, and trends in rainfall events. This replaces the manual practice of compiling handwritten logs or spreadsheets, minimizing errors and significantly improving efficiency, accuracy, and accessibility of rainfall records.

The system allows users to compose messages that will be sent to designated contact. When a message is created, the system captures and stores important details such as the unique message ID, the text description written by the user, related rainfall event information retrieved from connected devices, and the date and time the message was created. Once saved in the message data store, the message becomes available for distribution through the sending process to the intended contact. This structured workflow ensures that all alerts and notifications are properly documented, traceable, and directly linked to actual rainfall events, thereby enhancing accountability, accuracy, and the timely dissemination of critical information to the right individuals.

The system allows users to request message history, which triggers the retrieval of stored records from the message data store. Each record includes details such as the unique message ID, the content or text description of the message, the associated rainfall event information, the contact list, the sender's user ID, and the date and time the message was created and sent. The system then compiles this information into a message summary report that can be viewed or exported by the user. By organizing communication records in this way, the system ensures proper tracking of all alerts and notifications, supports accountability by linking messages to specific users and events, and enables administrators or staff to efficiently review, audit, and manage past communications. This replaces the manual process of recording messages or relying on memory, providing a reliable and accurate history that strengthens both operational efficiency and transparency.

The system allows users to add and update contact records to ensure that contact information is accurate and up to date. Each contact entry includes important details such as contact ID, full name, contact number and, assigned location or group. When users add a new contact, the information is captured and stored in the Contact Data Store, making the contact available for future alerts and notifications. If updates are made such as changes in phone numbers or contact preferences the system immediately reflects these modifications to maintain accuracy. This process guarantees that all messages are delivered to the correct contact without duplication or error. By centralizing and automating contact management, the system eliminates the need for maintaining separate manual lists, reduces the risk of outdated or missing information, and supports efficient, reliable communication during rainfall events.

The logout process is initiated when the user requests to exit the system and terminate the current session. Once the request is made, the system validates the action, clears any active session data, and confirms that the user has successfully logged out. This ensures that no further transactions or activities can be performed under the same session, protecting sensitive data and preventing unauthorized access if the device or terminal is left unattended. By securely ending the session, the logout process reinforces system security, maintains accountability for user activities, and ensures that only authenticated sessions remain active within the platform.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14 Issue 12 , December - 2025

Figure 15 below represents the physical entity diagram of the system. It illustrates how data is stored, organized, and linked across different tables.

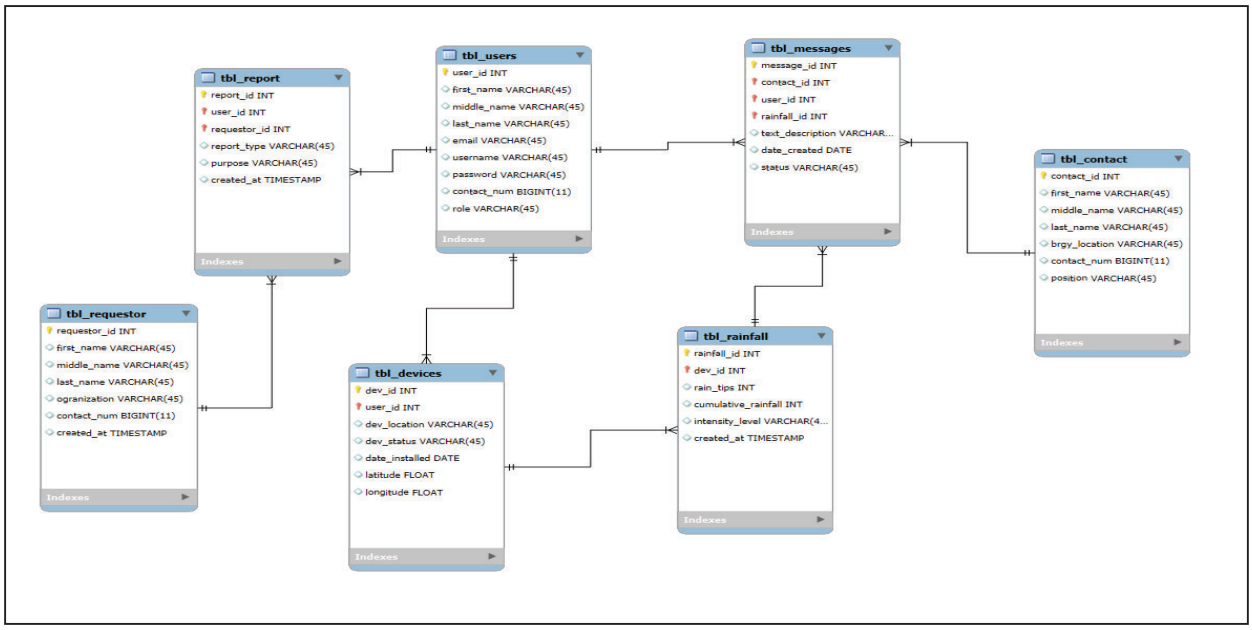


Figure 15 Physical Database Diagram

The Device Table stores information about each rainfall monitoring device installed in the field. Every device is uniquely identified by dev_id, which serves as the primary key. Key attributes such as dev_location, latitude, and longitude indicate the precise geographical placement of the device, ensuring accurate mapping of rainfall data. The date_installed field records when the device was deployed, while status reflects its operational condition (active,or inactive, This table is crucial for tracking and managing the system's physical monitoring units that continuously collect rainfall information.

The Rainfall Table holds the primary environmental data captured by each monitoring device. Each rainfall record is uniquely identified by rainfall_id and is linked to a specific device through the foreign key dev_id. The table stores temporal and quantitative attributes such as created_at (timestamp of data recording), rain_tips (number of bucket tips from the tipping bucket mechanism), cumulative_rainfall (total rainfall volume computed from the number of tips), and intensity_level (categorization of rainfall intensity such as light, moderate, or heavy). This table serves as the main data source for rainfall analytics, alerts, and reporting functions.

The User Table contains details of individuals authorized to access and manage the system. Each user is uniquely identified by user_id as the primary key. The table includes personal and contact attributes such as first_name, middle_name, last_name, contact_num, and email. The username and password fields secure system access, while the role field specifies the user's position, such as administrator or staff. This table ensures accountability by linking all major actions such as message creation, report generation, and device management to a specific user within the system.

The Message Table manages all notifications or alerts generated within the system. Each message is uniquely identified by message_id and contains foreign keys that link it to related entities: user_id (the user who created the message), rainfall_id (the rainfall event that triggered it), and contact_id (the recipient). The text_description field stores the message content, date_created logs when the message was generated, and *status* indicates the delivery state (sent, pending, or failed). This table functions as the bridge between rainfall data and communication, ensuring that alerts are appropriately recorded and traceable.

The Contact Table stores details of individuals or entities who receive rainfall alerts and system notifications. Each contact is identified by contact_id, the table's primary key. It includes attributes such as first_name, middle_name, last_name, and contact_num to personalize message delivery. Additional fields like brgy_location and position define the contact's location and role, ensuring that alerts reach the right recipients, such as barangay officials or disaster response personnel. This table plays a key role in ensuring effective dissemination of rainfall alerts.

The Requestor Table records information about individuals or organizations requesting rainfall reports. Each requestor is uniquely identified by requestor_id, which serves as the primary key. The table includes attributes such as first_name, middle_name, last_name, organization, and date, providing complete information about the requester and their affiliation. This table ensures that all report requests are properly documented,

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14 Issue 12 , December - 2025

attributed, and stored for monitoring and auditing purposes.

The Report Table maintains all reports generated within the system. Each report is uniquely identified by report_id as the primary key. The table includes attributes such as report_type (to specify the kind of report generated), purpose (to define the reason for its generation), and created_at (to record the date of creation). It also links to the user_id (who generated the report) and requestor_id (the requester of the report) through foreign keys. This table supports systematic recordkeeping and ensures that all generated reports are traceable and aligned with official requests or data needs.

User Interface

The lo-fi interface serves as the initial visual representation of the system's design, focusing on layout and functionality rather than detailed graphics. It provides a simple and clear overview of how users will interact with key features and navigation elements. This approach helps identify usability issues early and guides the development of the high-fidelity interface.

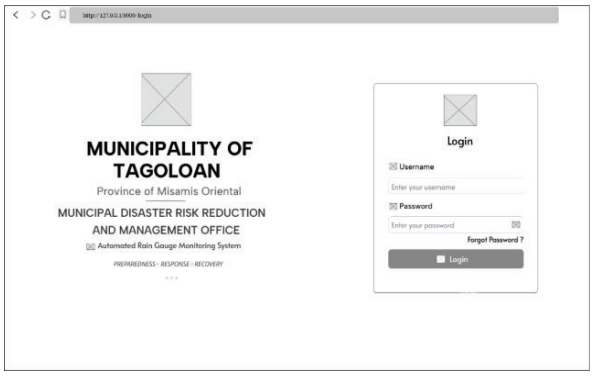


Figure 16 Low Fidelity for Login page

This login page contains two main input fields one for the and one Username for the Password, along with a Login button and a "Forgot Password?" link. After entering valid credentials and clicking the login button, the user will be navigated to the Dashboard page, where system data and monitoring details are displayed. If the user clicks "Forgot Password?", they will be directed to a password recovery form.

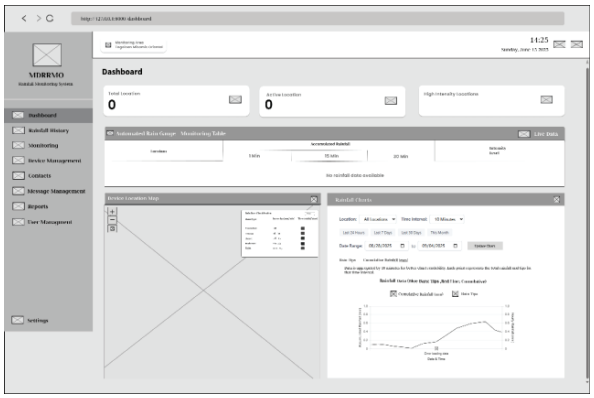


Figure 17 Low Fidelity for Dashboard

This Dashboard page contains three summary cards, a monitoring table, a device location map, and rainfall charts. It also features a sidebar navigation menu with links to different modules. The navigation flow after login directs the user here to view real-time rainfall data and summaries. From this page, users can easily navigate to other modules through the sidebar for further system operations.

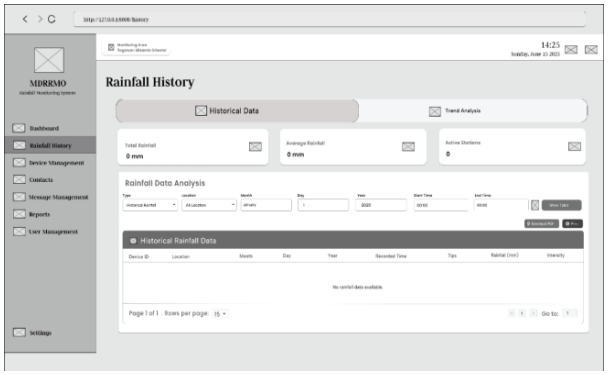


Figure 18 Low Fidelity for Rainfall History – Table

This Rainfall History page includes three summary cards, a Rainfall Data Analysis filter section, and a Historical Rainfall Data table and chart for data visualization. The navigation flow allows users to move here from the Dashboard or proceed to other sections.

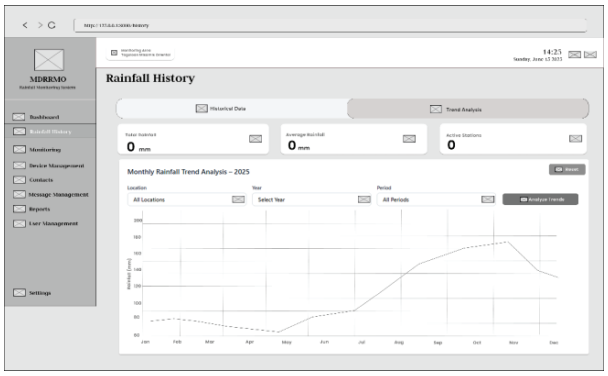


Figure 19 Low Fidelity for Rainfall History-Chart

The Rainfall History page presents three summary indicators to provide users with a quick understanding of past rainfall records. It features a Monthly Rainfall Trend Analysis section equipped with filters for selecting location, year, and period to refine the displayed data. The accompanying line chart visually illustrates these trends, enabling users to easily track rainfall patterns across the chosen timeframe.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14 Issue 12 , December - 2025

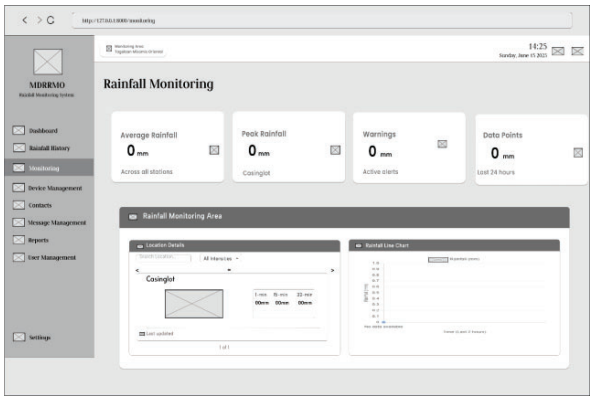


Figure 20 Low Fidelity for Monitoring

The Monitoring page presents key summary cards along with a Rainfall Monitoring Area that includes location details and a live rainfall line chart for quick data interpretation. It also provides a search and filter section that allows users to select specific locations or time intervals for more focused rainfall analysis. Below these components, users can access historical rainfall visuals and data, enabling deeper examination of trends and patterns across different monitoring periods.

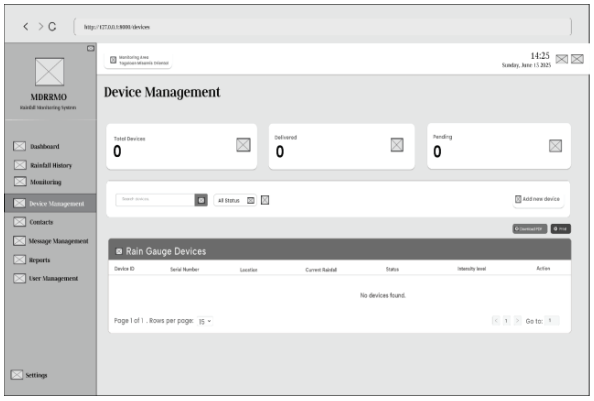


Figure 21 Low Fidelity for Device Management

This Device Management page includes three summary cards, a search bar, a filter dropdown, an “Add new device” button for add form modal below, and a Rain Gauge Devices table for listing device details. It also has a sidebar navigation menu for accessing other modules.

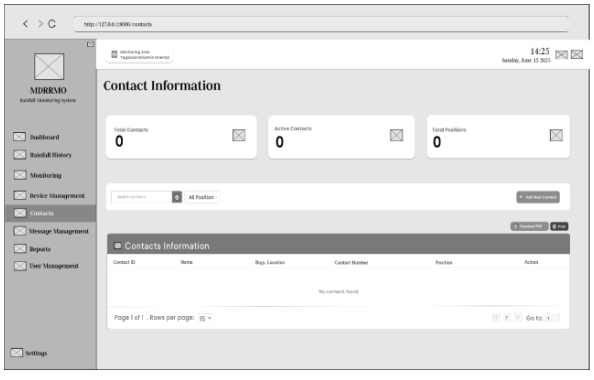


Figure 22 Low Fidelity for Contact Info. and Management

The Contact Info and Management page includes three summary cards for statistics, a search bar, a filter dropdown for filtering the contact list, an “Add new contact” button for the add form modal, and print and download buttons for contact list summaries reports or view. It also has a sidebar navigation menu for accessing other modules.

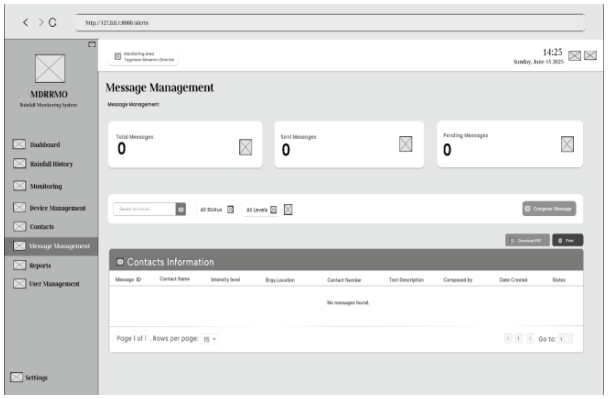


Figure 23 Low Fidelity for Message Management

The Message Management page includes three summary cards for statistics, a search bar, for filtering the contact list of message recipients, a “Compose Message” button for creating messages below. It also has status and level dropdowns to filter the messages.

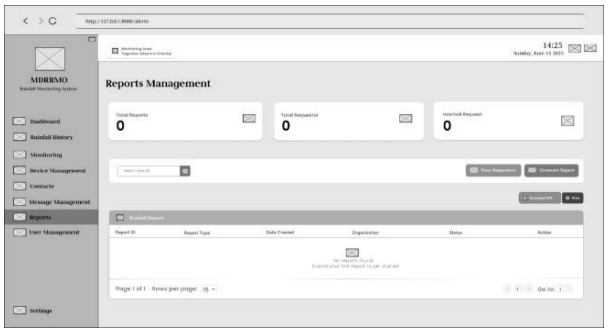


Figure 24 Low Fidelity for Reports Management

The Report Management page includes three summary cards for statistics, a search bar, for filtering the generated reports a “View requestor” button is for overview of list requestor and “Generate report” button for requesting data form. Download and print button for exporting data.

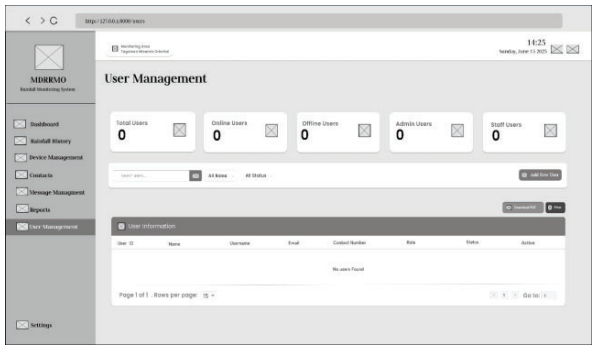


Figure 25 Low Fidelity for User Management

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14 Issue 12 , December - 2025

The User Management page includes five summary cards for statistics, a search bar, and dropdowns for roles and status to filter users. It also has an **"**Add User" button that opens an add modal form below for registering new users, as well as download and print buttons for exporting data.

Development Phase

The Automated Rain Gauge Monitoring System was developed using a five-layer architecture. Data Acquisition, Networking, Presentation, Application, and Database Layers. This structure ensures clear role separation, improving scalability, reliability, and maintainability while efficiently collecting, transmitting, processing, storing, and presenting rainfall data for the MDRRMO.

Table 2. Functional Requirements

Requirements Name	Description
Login	User needs to input username and password to access the system's user panel.
Dashboard	The system will allow the user to view real-time rainfall status and its intensity.
View Rainfall History	The system will allow the user to view previously collected rainfall data by date and time.
Manage Device	The system will allow the user to register, update, or remove connected rain gauge devices.
Manage Contact	The system will allow the user to add, edit, or remove emergency contact information for alert dissemination or information updates.
Message Management	The system will allow the user to compose and send custom messages to contacts (e.g., rainfall warnings).
Manage Users	The system will allow the user to create, update, or remove user accounts.
Manage Report	The system will allow the user to generate, view, and export rainfall reports in PDF format.
Settings	The system will allow the user to configure and update system preferences.

The login feature ensures that only authorized administrators can access the system. By requiring a username and password, it protects sensitive rainfall data and system configurations from unauthorized users. This is an important security measure that gives the user confidence that the system is safe to use.

The dashboard serves as the main monitoring hub of the system, giving the user a clear, real-time overview of the current rainfall status and its intensity. Displaying critical information in a simple and organized layout so the user can quickly understand rainfall conditions without confusion. Alongside rainfall readings, the dashboard includes a map that shows the exact location of the rain gauge device, ensuring the user knows where the data is coming from. It also provides visual aids like charts and graphs, which make it easier to analyze trends, compare rainfall data over time, and identify patterns that may be useful for decision-making. In short, the dashboard acts as the control center, bringing together all essential data in one accessible and easy-to-navigate space.

The View Rainfall History module enables the user to review rainfall data that has been automatically recorded and securely stored in the system, neatly organized by date and time. It also includes a search function that allows the user to easily locate and examine specific past rainfall events. This feature makes it possible to track and analyze how rainfall has varied across different periods, whether on a daily, monthly, or seasonal basis. To provide deeper insights, the system can generate comprehensive reports and present the data in trend analysis graphs, which clearly illustrate patterns such as heavy rainfall seasons, rainfall frequency, or unusual rain events. By maintaining this historical record, the module serves as a dependable reference point, ensuring that the user can revisit past conditions, compare them with real-time updates, and make informed decisions for effective monitoring, disaster preparedness.

The Manage Device provides the admin with full control over all devices connected to the rainfall monitoring system. In this module, the admin can register new devices by inputting important details such as the device's serial number, assigned location and installation date. Existing devices can also be updated to reflect changes like relocation. If a device is no longer in use, the admin can easily remove it from the system to keep records clean and accurate.

The Manage Contact enables the user to add, edit, or remove contacts who are responsible for receiving important alerts and warnings. These contacts may include members of community leaders, or local organizations. Having an updated and accurate list of contacts is essential to ensure that notifications reach the right people at the right time. This feature helps strengthen communication lines during emergencies, improving preparedness and response.

The Message Management allows the user to create and send customized notifications, such as rainfall alerts, safety warnings, or specific instructions, directly to selected contacts or groups. It also provides an overview of all message statuses sent, pending, and failed so the user can ensure that critical information is delivered reliably and take action if any issues arise.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14 Issue 12 , December - 2025

The Manage Users gives the admin full control over who can access the system by allowing them to create new user accounts, update existing ones, or remove accounts that are no longer needed. This feature helps define roles and responsibilities, ensuring accountability and security within the system. By restricting access only to authorized users, it safeguards data integrity and prevents unauthorized changes or misuse.

The system ensures that all data is protected and that only authorized users can access it. Users are required to create strong passwords, and the system incorporates security measures to prevent unauthorized access, safeguard sensitive rainfall information, and maintain data integrity.

The Manage Report allows the admin to generate, view, and export rainfall data reports, usually in PDF format, containing key details such as rainfall history, intensity, and trends. These reports make it easier to share accurate information with stakeholders, including local government units, disaster response teams, researchers or who request the data. By supporting documentation and long-term record-keeping, this module plays a vital role in planning, analysis, and policy-making.

The settings allow the admin to configure and adjust the system according to specific requirements. This includes updating notification preferences, modifying system parameters, and customizing other settings to ensure the system operates efficiently and meets the needs of its users.

Table 3. Non-Functional Requirements

Requirements Name	Description
Accessibility	The system is a web-based application which can be accessed through PC and laptop using a standard web browser.
Availability	The system will be available anytime and anywhere as long as there is a stable internet connection.
Reliability	The system is designed using software tools that are applicable and easy for everyone to use, ensuring dependable performance.
Security	The system data is secured, and accessing the system requires the user to create a strong password.

The system is designed as a web-based application, making it accessible through any standard web browser on a PC or laptop. This ensures that users can easily access the system without needing specialized hardware or software, allowing for flexibility and convenience in monitoring rainfall data from different locations.

The system is intended to be available anytime and anywhere, provided there is a stable internet connection. This ensures continuous access to real-time rainfall information, enabling the admin and other authorized users to monitor conditions and respond promptly to weather events without unnecessary downtime.

The system is built using software tools that are widely applicable and user-friendly, which guarantees consistent and dependable performance. Its design minimizes errors, supports smooth operation, and ensures that critical rainfall data is accurately collected and presented without interruption or data loss.

The system ensures that all data is protected and that only authorized users can access it. Users are required to create strong passwords, and the system incorporates security measures to prevent unauthorized access, safeguard sensitive rainfall information, and maintain data integrity.

Prototype Construction

A working prototype of the Automated Rain Gauge Monitoring System was developed in three versions to demonstrate its functionality, improvements over time, and the progressive enhancement.

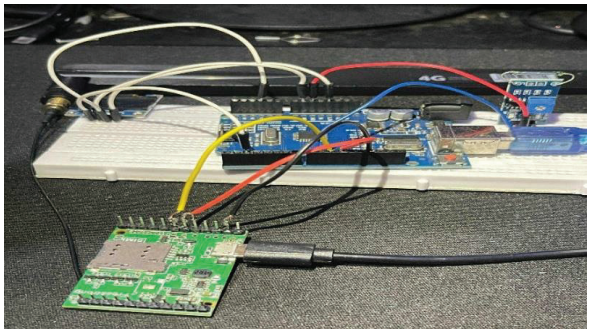


Figure 26 Version 1 Prototype

The figure shows the hardware setup of Version 1 of the Automated Rain Gauge Monitoring System. It features an Arduino Uno microcontroller connected to a GSM module for real-time data transmission. The setup includes wiring connections on a breadboard, which link the microcontroller to the sensor and communication module. This prototype demonstrates the basic configuration for collecting rainfall data and sending it to a web dashboard, serving as a proof-of-concept for the system's functionality.

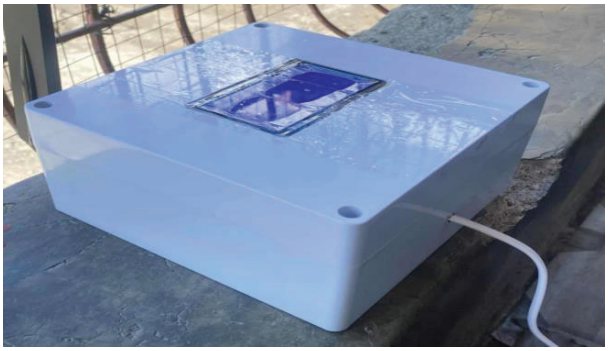


Figure 27 Version 2 Prototype (Outside)

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14 Issue 12 , December - 2025

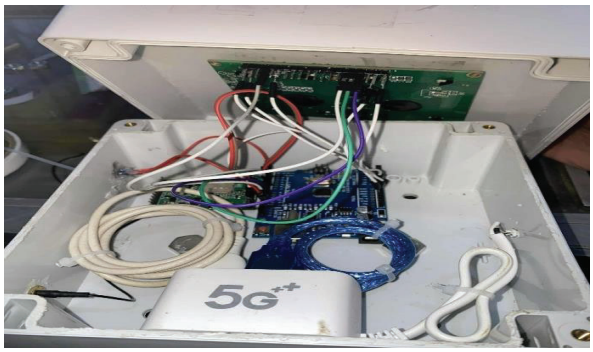


Figure 28 Version 2 Prototype (Inside)

The figure shows the hardware setup of the rain gauge device. The improved prototype features a compact and durable enclosure that securely houses the internal components, protecting them from external environmental factors. Inside, the components, including the Arduino microcontroller, GSM module, and connection interface, are neatly arranged within the plastic casing to ensure organized wiring and stable operation. Additionally, a 5G+ modem is integrated to enhance data transmission speed and reliability for real-time monitoring.

Materials Costing

Table 4. Version 1 Costing

Materials/Tools	Description	Price (₱)
Arduino Uno Microcontroller	A microcontroller board based on the ATmega328P, used for controlling sensors and communication modules in the rain gauge monitoring system.	349.00
SIM A7672s Module	A 4G LTE communication module that allows the system to transmit rainfall data to a server or dashboard using mobile networks.	1,363.00
Breadboard	A rectangular platform used for building and testing electronic circuits without soldering. Components and wires can be inserted for prototyping.	100
4G LTE Antenna	External antenna used to enhance the LTE signal reception of the SIM A7672s module for reliable data transmission.	130.00
Reed Sensor	A magnetic switch sensor that detects the opening and closing of contacts when exposed to a magnetic field. In a rain gauge, it is used to count the tipping motion of the bucket, generating pulses that correspond to rainfall measurement.	35.00

Table 5. Version 2 Costing

Materials/Tools	Description	Price (₱)
Arduino Uno Microcontroller	A microcontroller board based on the ATmega328P, used for controlling sensors and communication modules in the rain gauge monitoring system.	349.00
SIM A7672s Module	A 4G LTE communication module that allows the system to transmit rainfall data to a server or dashboard using mobile networks.	1,363.00
Reed Sensor	A magnetic switch sensor that detects the opening and closing of contacts when exposed to a magnetic field. In a rain gauge, it is used to count the tipping motion of the bucket, generating pulses that correspond to rainfall measurement	35.00
4G LTE Antenna	External antenna used to enhance the LTE signal reception of the SIM A7672s module for reliable data transmission.	130.00
LCD Module	A liquid crystal display used to show real-time data such as rainfall amount, device status, and system messages. It provides a simple interface for monitoring information directly from the device.	219.00
GPS Antenna	Antenna for GPS functionality, used to obtain accurate location data of the device installation site.	129.00

Table 6. Version 3 Costing

Materials/Tools	Description	Price (₱)
Arduino Uno Microcontroller	A microcontroller board based on the ATmega328P, used for controlling sensors and communication modules in the rain gauge monitoring system.	349.00
SIM A7672s Module	A 4G LTE communication module that allows the system to transmit rainfall data to a server or dashboard using mobile networks.	1,363.00
4G LTE Antenna	External antenna used to enhance the LTE signal reception of the SIM A7672s module for reliable data transmission.	130.00
Reed Sensor	A magnetic switch sensor that detects the opening and closing of contacts when exposed to a magnetic field. In a rain gauge, it is used to count the tipping motion of the bucket, generating pulses that correspond to rainfall measurement	35.00
LCD Module	A liquid crystal display used to show real-time data such as rainfall amount, device status, and system messages. It provides a simple interface for monitoring information directly from the device.	219.00
Steel Bar	A durable support structure used as a stand to hold and stabilize the rain gauge and its components, ensuring the device remains upright and securely fixed in place under outdoor conditions.	200
Junction Box	A weatherproof enclosure that houses and protects electronic components, wiring, and connections from environmental damage.	235.00

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14 Issue 12 , December - 2025

Major Functionalities

```php
public function store(Request $request)
{
    try {
        $result = $this->contactService->createContact($request->all());

        if ($request->wantsJson() || $request->ajax()) {
            return response()->json($result, $result['success'] ? 200 : 422);
        }

        return redirect()->route('contacts.index')
            ->with($result['success'] ? 'success' : 'error', $result['message']);
    } catch (\Exception $e) {
        $message = 'An unexpected error occurred: ' . $e->getMessage();

        if ($request->wantsJson() || $request->ajax()) {
            return response()->json(['success' => false, 'message' => $message], 500);
        }

        return redirect()->route('contacts.index')
            ->with('error', $message)
            ->withInput();
    }
}

/**
```

Figure 29 Code Snippet for Add Contact

This functionality is to allow users to register new contacts within the system. It specifically stores essential contact details such as name and phone number in the database form message.

```php
public function store(Request $request)
{
    // Basic validation rules based on requestor_type
    $rules = [
        'requestor_type' => 'required|in:no_requestor,old_requestor,new_requestor',
        'report_type' => 'required|string|in:rainfall_history,message_history,device_info,contacts_info,user_management,reports_summary,all_reports',
        'start_date' => 'nullable|date',
        'end_date' => 'nullable|date|after_or_equal:start_date',
    ];

    // Add requestor-specific fields based on requestor_type
    if ($request->requestor_type === 'old_requestor') {
        $rules['requestor_id'] = 'required|exists:requestors,requestor_id';
        $rules['purpose'] = 'required|string|min:10|max:500';
    } elseif ($request->requestor_type === 'new_requestor') {
        // Fields for new requestor
        $rules['first_name'] = 'required|string|max:255';
        $rules['middle_name'] = 'nullable|string|max:255';
        $rules['last_name'] = 'required|string|max:255';
        $rules['organization'] = 'required|string|max:255';
        $rules['purpose'] = 'required|string|min:10|max:500';
    } else {
        // For no_requestor, purpose is optional
        $rules['purpose'] = 'nullable|string|max:500';
    }

    $request->validate($rules);
```

Figure 30 Code Snippet for Add Contact

This functionality enables the system administrator to register new monitoring devices. It records device identification details and assigns them to specific locations for tracking purposes. The process ensures that data from each device is accurately mapped within the system's network for efficient communication and reporting.

```php
public function store(Request $request)
{
    try {
        $validated = $this->validateDeviceData($request);

        // Normalize the location
        $validated['dev_location'] = ucwords(strtolower($validated['dev_location']));

        DB::table('devices')->insert([
            'serial_number' => $validated['serial_number'],
            'dev_location' => $validated['dev_location'],
            'date_installed' => now(),
            'status' => 'pending',
            'added_by' => Auth::id(), // Add the authenticated user's ID
            'created_at' => now(),
            'updated_at' => now(),
        ]);

        return redirect()->route('devices.index')->with('success', 'Device added successfully.');

    } catch (\Illuminate\Validation\ValidationException $e) {
        return redirect()->back()->withErrors($e->errors())->withInput();
    } catch (\Exception $e) {
        return redirect()->back()
            ->with('error', 'Failed to add device: ' . $e->getMessage())
            ->withInput();
    }
}
```

Figure 31 Code Snippet for Generate Report

```php
try {
    DB::beginTransaction();

    $reportData = [
        'user_id' => Auth::id(),
        'requestor_type' => $request->requestor_type,
        'report_type' => $request->report_type,
        'start_date' => $request->start_date,
        'end_date' => $request->end_date,
        'purpose' => $request->purpose,
        'status' => 'pending',
    ];

    // Handle requestor based on type
    if ($request->requestor_type === 'old_requestor') {
        $reportData['requestor_id'] = $request->requestor_id;

        // Get organization from the existing requestor
        $requestor = Requestor::find($request->requestor_id);
        if ($requestor) {
            $reportData['organization'] = $requestor->organization;
        }
    } elseif ($request->requestor_type === 'new_requestor') {
        // Create new requestor
        $requestor = Requestor::create([
            'first_name' => $request->first_name ?? '',
            'middle_name' => $request->middle_name ?? '',
            'last_name' => $request->last_name ?? '',
            'organization' => $request->organization ?? 'N/A',
        ]);

        $reportData['requestor_id'] = $requestor->requestor_id;
        $reportData['organization'] = $request->organization;
    } else {
        // For 'no_requestor', set default organization
        $reportData['organization'] = 'MDDRMO';
    }

    $report = Report::create($reportData);
```

Figure 32 Code Snippet for Generate Report.

The Generate Reports function is designed to compile rainfall and system activity data into clear, readable reports. It collects stored information from the database and organizes it into structured tables or charts for easy interpretation. Through these reports, users can effectively analyze system performance and observe rainfall or weather trends over a specific period, supporting data-driven monitoring and decision-making.

```php
private function createMessages($contacts, array $data)
{
    $sentCount = 0;
    $batchSize = 10; // Process in batches to avoid memory issues

    Log::info('Starting message creation', ['total_contacts' => $contacts->count()]);

    foreach ($contacts->chunk($batchSize) as $contactBatch) {
        $messagesToInsert = [];

        foreach ($contactBatch as $contact) {
            $deviceIntensity = $this->getDeviceIntensity($contact->brgy_location);

            $messagesToInsert[] = [
                'intensity_level' => $deviceIntensity ?: 'moderate',
                'contact_id' => $contact->contact_id,
                'brgy_location' => $contact->brgy_location,
                'contact_num' => $contact->contact_num,
                'text_desc' => $data['message'],
                'status' => 'pending',
                'date_created' => now(),
                'user_id' => auth()->id(),
                'created_at' => now(),
                'updated_at' => now(),
            ];

            $sentCount++;
        }

        // Bulk insert for better performance
        if (!empty($messagesToInsert)) {
            Message::insert($messagesToInsert);
        }
    }

    Log::info('Message creation completed', ['sent_count' => $sentCount]);
    return $sentCount;
}
```

Figure 33 Code Snippet for Create Message.

This function is responsible for generating and storing alert messages for a list of contacts. The function processes the contacts in batches to prevent memory issues, logging the start of the message creation process and the total number of contacts involved. For each contact, it determines the rainfall intensity level based on the device's location and prepares message details such as contact information, message content, and timestamps. These message records are then inserted in bulk into the database to improve performance. Finally, the function logs the completion of the process and returns the total number of messages created.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14 Issue 12 , December - 2025

```
function validateFieldRealTime(fieldId) {
  const field = document.getElementById(fieldId);
  if (!field) return;

  const value = field.value.trim();
  const isEmpty = value === '';

  // Clear previous styling
  field.classList.remove('form-field-valid', 'form-field-invalid');
  clearFieldSuccess(fieldId);

  // Don't validate empty fields - only show success when there's actual valid content
  if (isEmpty) {
    clearFieldError(fieldId);
    return false;
  }

  if (fieldId === 'username') {
    if (!usernameRegex.test(value)) {
      field.classList.add('form-field-invalid');
      showFieldError(fieldId);
      return false;
    } else {
      field.classList.add('form-field-valid');
      clearFieldError(fieldId);
      showFieldSuccess(fieldId);
      return true;
    }
  }

  if (fieldId === 'password') {
    const passwordRegex = /^(?=.*[a-zA-Z])(?=.*\d).{8,}$/;
    if (!passwordRegex.test(value)) {
      field.classList.add('form-field-invalid');

      return false;
    } else {
      field.classList.add('form-field-valid');
      clearFieldError(fieldId);
```

Figure 34 Code Snippet for Login Validation.

This functionality serves as a security measure that allows only authorized users to access the system. It works by validating the credentials entered during login against the stored information in the database. Once the system confirms that the provided username and password match an existing record, the authentication process is considered successful. As a result, the user is granted access to the system's main dashboard and its various modules, ensuring that sensitive data and features remain protected from unauthorized access.

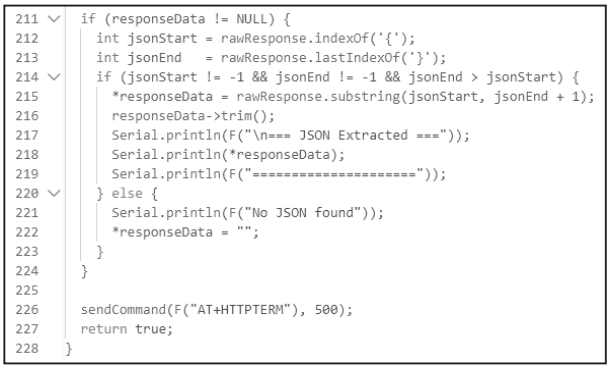
```
bool sendSMS(String number, String message) {
  Serial.print(F("Sending SMS to ")); Serial.println(number);

  sendCommand(F("AT+CMGF=1"), 1000);

  if (!number.startsWith(F("+63"))) number = "+63" + number;

  sim.print(F("AT+CMGS=\""));
  sim.print(number);
  sim.println("\"");
  delay(1500);

  sim.print(message);
  delay(500);

  sim.write(26); // CTRL+Z
  delay(9000);

  String modemResponse = "";
  unsigned long start = millis();
  while (millis() - start < 15000) {
    while (sim.available()) {
      char c = sim.read();
      Serial.write(c);
      modemResponse += c;
      start = millis();
    }
  }

  Serial.println(F("\n--- SMS Response ---"));
  Serial.println(modemResponse);

  if (modemResponse.indexOf(F("+CMGS")) != -1) {
    return true;
  } else {
    return false;
  }
}
```

Figure 35 Code Snippet for Send SMS.

This function enables the system to send SMS messages through a GSM module. It first prepares the recipient's number by ensuring it includes the "+63" country code and sets the module to SMS mode. The message is then transmitted using a series of AT commands, with a short delay to allow processing. And GSM module's response to confirm whether the SMS was successfully sent or failed.

```
bool httpGET(String targetURL, String* responseData) {
  targetURL.replace(" ", "%20");

  sendCommand(F("AT+CGATT=1"), 1000);
  sim.print(F("AT+CGDCONT=1,\"IP\",\""));
  sim.print((__FlashStringHelper*)apn);
  sim.println("\"");
  delay(1000);
  sendCommand(F("AT+CGACT=1,1"), 2000);

  sendCommand(F("AT+HTTPINIT"), 1000);
  sendCommand(F("AT+HTTPPARA=\"CID\",1"), 500);

  sim.print(F("AT+HTTPPARA=\"URL\",\""));
  sim.print(targetURL);
  sim.println("\"");
  delay(800);
  while (sim.available()) Serial.write(sim.read());

  sim.println(F("AT+HTTPACTION=0"));

  String actionResponse = "";
  unsigned long start = millis();
  while (millis() - start < 15000) {
    while (sim.available()) {
      char c = sim.read();
      actionResponse += c;
      Serial.write(c);
    }
    if (actionResponse.indexOf(F("+HTTPACTION:")) != -1 &&
        actionResponse.indexOf("\n", actionResponse.indexOf(F("+HTTPACTION:"))) != -1) {
      break;
    }
  }
```

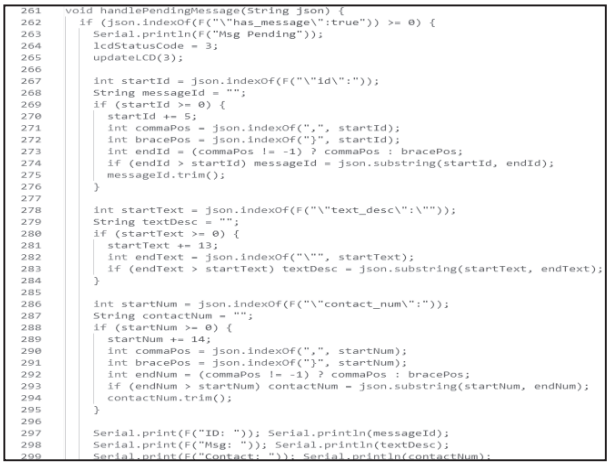
Figure 36 Code Snippet for HTTP GET



```
if (responseData != NULL) {
  int jsonStart = rawResponse.indexOf('{');
  int jsonEnd   = rawResponse.lastIndexOf('}');
  if (jsonStart != -1 && jsonEnd != -1 && jsonEnd > jsonStart) {
    *responseData = rawResponse.substring(jsonStart, jsonEnd + 1);
    responseData->trim();
    Serial.println(F("\n=== JSON Extracted ==="));
    Serial.println(*responseData);
    Serial.println(F("======================"));
  } else {
    Serial.println(F("No JSON found"));
    *responseData = "";
  }
}

sendCommand(F("AT+HTTPTERM"), 500);
return true;
}
```

Figure 37 Code Snippet for HTTP GET.

This function allows the system to request data from the web server. It retrieves updated message statuses or device readings via HTTP GET protocol. The retrieved data keeps the local display synchronized with the online database.



```
void handlePendingMessage(String json) {
  if (json.indexOf(F("\"has_message\":true")) >= 0) {
    Serial.println(F("Msg Pending"));
    lcdStatusCode = 3;
    updateLCD(3);

    int startId = json.indexOf(F("\"id\":"));
    String messageId = "";
    if (startId >= 0) {
      startId += 5;
      int commaPos = json.indexOf(",", startId);
      int bracePos = json.indexOf("}", startId);
      int endId = (commaPos != -1) ? commaPos : bracePos;
      if (endId > startId) messageId = json.substring(startId, endId);
      messageId.trim();
    }

    int startText = json.indexOf(F("\"text_desc\":\""));
    String textDesc = "";
    if (startText >= 0) {
      startText += 13;
      int endText = json.indexOf("\"", startText);
      if (endText > startText) textDesc = json.substring(startText, endText);
    }

    int startNum = json.indexOf(F("\"contact_num\":"));
    String contactNum = "";
    if (startNum >= 0) {
      startNum += 14;
      int commaPos = json.indexOf(",", startNum);
      int bracePos = json.indexOf("}", startNum);
      int endNum = (commaPos != -1) ? commaPos : bracePos;
      if (endNum > startNum) contactNum = json.substring(startNum, endNum);
      contactNum.trim();
    }

    Serial.print(F("ID: ")); Serial.println(messageId);
    Serial.print(F("Msg: ")); Serial.println(textDesc);
    Serial.print(F("Contact: ")); Serial.println(contactNum);
```

Figure 38 Code Snippet for Check Pending SMS

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14 Issue 12 , December - 2025

```
300
301  if (textDesc.length() > 0 && contactNum.length() > 0) {
302    bool smsSent = sendSMS(contactNum, textDesc);
303    if (smsSent) {
304      String url = String((__FlashStringHelper*)server) + F("/api/messages/") + messageId + F("/status?status=delivered");
305      Serial.println(F("SMS sent. Updating..."));
306      if (sendHTTPGETSimple(url)) Serial.println(F("Status updated!"));
307      else Serial.println(F("Update failed!"));
308    } else {
309      Serial.println(F("SMS failed"));
310    }
311  }
312  } else {
313    Serial.println(F("No messages"));
314  }
315 }
316
```

Figure 39 Code Snippet for HTTP GET

This function is responsible for monitoring and managing pending SMS messages within the system. It scans the database to locate unsent or queued messages that are awaiting transmission. Upon identifying these messages, the system processes them to ensure they are sent promptly.

```
328
329 void saveToBuffer(int tips) {
330   if (bufferCount < MAX_BUFFER) {
331     buffer[bufferTail].tips = tips;
332     buffer[bufferTail].timestamp = millis();
333     bufferTail = (bufferTail + 1) % MAX_BUFFER;
334     bufferCount++;
335     Serial.println(F("Data buffered"));
336     lcdStatusCode = 2;
337     updateLCD(2);
338   } else {
339     Serial.println(F("Buffer full!"));
340     lcdStatusCode = 4;
341     updateLCD(4);
342   }
343 }
344
```

Figure 40 Code Snippet for Store Temporary Storage.

The Data Buffer function temporarily stores unsent or unprocessed data before it is transmitted to the main database. It acts as a holding area that ensures all information is safely retained until a stable connection is available. This process prevents data loss during network interruptions.

```
345 bool sendBufferData() {
346   lcdStatusCode = 1;
347   updateLCD(1);
348
349   while (bufferCount > 0) {
350     RainRecord record = buffer[bufferHead];
351
352     String fullURL = String((__FlashStringHelper*)server)
353                      + F("/rain?rain_tips=") + String(record.tips)
354                      + F("&serial_number=") + String((__FlashStringHelper*)device_serial)
355                      + F("&timestamp=") + String(record.timestamp);
356
357     if (sendHTTPGETSimple(fullURL)) {
358       Serial.print(F("Buffer sent: "));
359       Serial.println(record.tips);
360       bufferHead = (bufferHead + 1) % MAX_BUFFER;
361       bufferCount--;
362       lcdStatusCode = 5;
363       updateLCD(5);
364     } else {
365       Serial.println(F("Buffer send fail"));
366       lcdStatusCode = 4;
367       updateLCD(4);
368       return false;
369     }
370   }
371   return true;
372 }
```

Figure 41 Code Snippet for Send Rainfall Data.

This functionality is to upload real-time rainfall measurements to the server. It packages rainfall counts and transmits them using HTTP or GSM protocols. This process ensures that the monitoring center receives up-to-date rainfall information.

```
44  public function store(UserFormRequest $request)
45  {
46      try {
47          $validated = $request->validated();
48
49          DB::transaction(function () use ($validated) {
50              $user = User::create([
51                  'username' => $validated['username'],
52                  'email' => $validated['email'],
53                  'password' => Hash::make($validated['password']),
54                  'role' => $validated['role'],
55                  'first_name' => $validated['first_name'],
56                  'middle_name' => $validated['middle_name'] ?? null,
57                  'last_name' => $validated['last_name'],
58                  'contact_num' => $validated['contact_num'],
59              ]);
60          });
61
62          return $this->jsonOrRedirect($request, 'User created successfully.', 'users');
63
64      } catch (ValidationException $e) {
65          return $this->handleValidationError($request, $e);
66      } catch (\Exception $e) {
67          return $this->handleGeneralError($request, 'An error occurred while creating the user: ' . $e->getMessage());
68      }
69  }
70
```

Figure 42 Code Snippet for Add User.

This function validates the request data and creates a new user inside a database transaction. It securely hashes the password and saves user details like username, email, and contact number. If successful, it redirects with a success message; if not, it handles validation or general errors.

SQL Report Queries

```
SELECT
        d.dev_id,
        r.dev_location,
        DATE(r.created_at) AS date,
        MONTH(r.created_at) AS month,
        DAY(r.created_at) AS day,
        YEAR(r.created_at) AS year,
        r.created_at,
        r.rain_tips AS tip_count,
        r.cumulative_rainfall,
        r.intensity_level
FROM
        tbl_rainfalls AS r
LEFT JOIN
        tbl_devices AS d ON d.dev_id = r.device_id
ORDER BY
        r.created_at DESC;
```

Figure 43 SQL Query for Historical Rainfall.

The purpose of this SQL query is to retrieve all recorded rainfall data over a defined historical period. It extracts date, time, and rainfall amount from the database. The report helps users analyze rainfall trends and variations over time for research or forecasting.

Table 7. Example Query Result for Historical Rainfall

Historical Rainfall Data

Device ID	Location	Month	Day	Year	Recorded Time	Tips	Rainfall (mm)	Intensity
1	Baluarte	Oct	9	2025	05:14	3	0.6	Light
1	Baluarte	Oct	9	2025	05:14	0	0	No Rain
1	Baluarte	Oct	9	2025	05:13	0	0	No Rain
1	Baluarte	Oct	9	2025	05:13	0	0	No Rain
1	Baluarte	Oct	9	2025	05:13	5	1	Light
1	Baluarte	Oct	9	2025	04:33	0	0	No Rain
1	Baluarte	Oct	9	2025	04:10	1	0.2	Light

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14 Issue 12 , December - 2025

The Historical Rainfall Table presents detailed rainfall records for Baluarte on October 9, 2025. It displays the exact time each measurement was recorded, along with the number of rain gauge tips, the corresponding rainfall amount in millimeters, and the categorized intensity level. Through this table, users can clearly observe the progression of rainfall throughout the day, distinguishing periods of light rain from intervals of no rainfall, which supports accurate data analysis and reporting.

```
SELECT
    d.dev_id,
    r.dev_location,
    DATE(r.created_at) AS date,
    MONTH(r.created_at) AS month,
    DAY(r.created_at) AS day,
    YEAR(r.created_at) AS year,
    ROUND(AVG(r.cumulative_rainfall), 2) AS average_rainfall,
    ROUND(SUM(r.rain_tips), 2) AS total_tips,
    ROUND(SUM(r.cumulative_rainfall), 2) AS total_rainfall
FROM
    rainfalls AS r
LEFT JOIN
    devices AS d ON d.dev_id = r.device_id
GROUP BY
    d.dev_id, r.dev_location, DATE(r.created_at), MONTH(r.created_at), DAY(r.created_at), YEAR(r.created_at)
ORDER BY
    DATE(r.created_at) DESC;
```

Figure 44 SQL Query for Daily Rainfall.

This query calculates the average rainfall amount for each day by retrieving summarized rainfall data grouped by date. It organizes the collected information into a clear daily summary that highlights variations in rainfall levels. The resulting report provides valuable insights into daily weather patterns and rainfall distribution over time.

Table 8. Example Query Result for Daily Rainfall.

Average Rainfall per Day

Device ID	Location	Month	Day	Year	Total Tips	Total Rainfall (mm)	Average Rainfall (mm)
1	Baluarte	October	14	2025	5.00	1	1
1	Baluarte	October	13	2025	20.00	4	4
1	Baluarte	October	12	2025	0.00	0	0
1	Baluarte	October	11	2025	15.00	3	3
1	Baluarte	October	10	2025	15.00	3	3
1	Baluarte	October	9	2025	40.00	8	0.67

The Average Rainfall per Day table shows the daily rainfall measurements recorded by a rain gauge device in Baluarte for October 2025. It includes the total rainfall in millimeters and the computed daily average based on the total tips collected.

```
SELECT
    d.dev_id,
    r.dev_location,
    DATE(r.created_at) AS date,
    MONTH(r.created_at) AS month,
    DAY(r.created_at) AS day,
    YEAR(r.created_at) AS year,
    ROUND(AVG(r.cumulative_rainfall), 2) AS average_rainfall,
    ROUND(SUM(r.rain_tips), 2) AS total_tips,
    ROUND(SUM(r.cumulative_rainfall), 2) AS total_rainfall
FROM
    rainfalls AS r
LEFT JOIN
    devices AS d ON d.dev_id = r.device_id
GROUP BY
    d.dev_id, r.dev_location, DATE(r.created_at), MONTH(r.created_at), DAY(r.created_at), YEAR(r.created_at)
ORDER BY
    DATE(r.created_at) DESC;
```

Figure 45 SQL Query for Monthly Rainfall.

This query generates the monthly average of recorded rainfall by extracting and computing the mean value of rainfall data for each month. It organizes the information to show overall rainfall trends and variations across different months. The resulting report is useful for analyzing long-term climate behavior and identifying seasonal changes in rainfall patterns.

Table 9. Example Query Result for Monthly Rainfall.

Average Rainfall per Month

Device ID	Location	Month	Year	Total Tips	Total Rainfall (mm)	Average Rainfall (mm)
1	Baluarte	February	2026	58	11.6	11.6
1	Baluarte	January	2026	35	7	7
1	Baluarte	December	2025	48	9.6	9.6

The Average Rainfall per Month table summarizes monthly rainfall data for the same location, displaying the total tips, total rainfall, and average rainfall for each month. It highlights variations in rainfall trends from October 2025 to February 2026.

User Interface

The high-fidelity user interface provides a detailed and refined visual representation of the system's overall layout and structure. It highlights the actual design elements, including colors, icons, typography, and interactive components that users will interact with. This interface demonstrates how various functions and features are seamlessly integrated to ensure smooth navigation and user interaction. It serves as a crucial reference in finalizing the system's design, ensuring usability, visual clarity, and an efficient user experience.

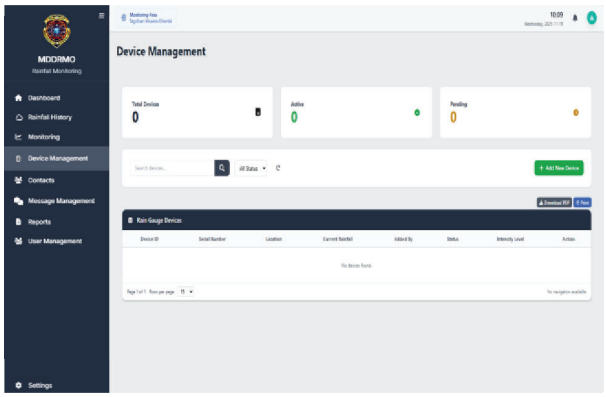


Figure 46 High Fidelity Login Page

The login page is for authorized users who have access to the system, whether they are admins or staff It helps protect sensitive information by ensuring only verified users can enter. This provides security and personalized access to the system's features.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14 Issue 12 , December - 2025

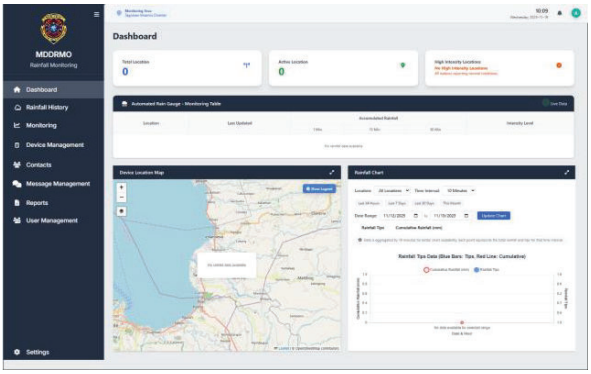


Figure 4 High Fidelity for Dashboard

The dashboard provides an overview of the system's generated data and monitoring. It displays key information and real-time updates in a clear and organized layout using tables, maps, and charts. This helps users easily track system performance and make informed decisions quickly.

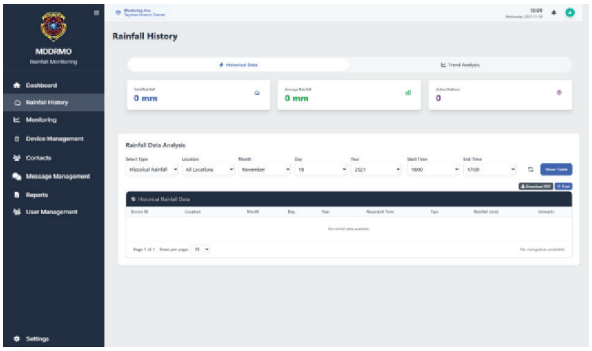


Figure 48 High Fidelity Rainfall History – Table

The Rainfall History Table is designed to store and display past rainfall data in an organized format. It allows users to review and analyze previous rainfall records for better understanding and decision-making. This feature helps in tracking patterns, supporting reports, and improving disaster preparedness.

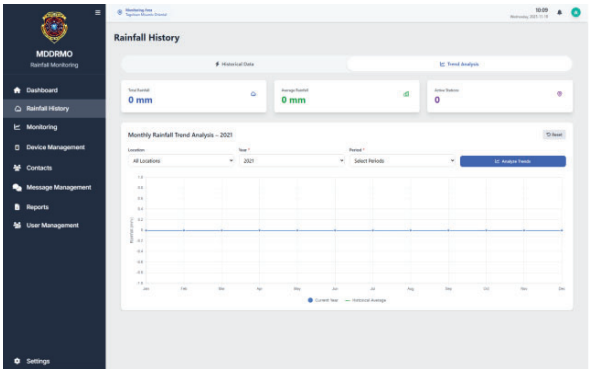


Figure 49 High Fidelity Rainfall History-Trend Analysis

The Rainfall History page is designed to provide users with a clear overview of previous rainfall data for effective monitoring and analysis. It helps users review long-term rainfall trends through visual charts and filters, supporting deeper understanding of seasonal or yearly patterns. This feature assists MDRRMO staff in making informed decisions, preparing reports, and strengthening disaster readiness based on historical rainfall behavior.

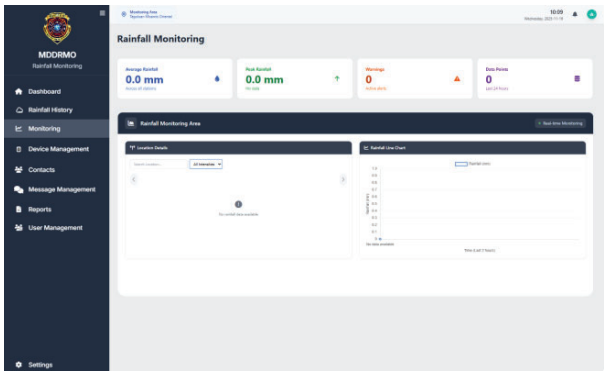


Figure 50 High Fidelity Monitoring

The Monitoring page provides real-time rainfall data from active stations, helping users keep track of weather conditions as they unfold. It displays up-to-date rainfall readings, alerts, and live trend visuals to support quick and informed decision-making. With this tool, MDRRMO staff can respond faster to changing weather situations, improving preparedness and ensuring community safety.

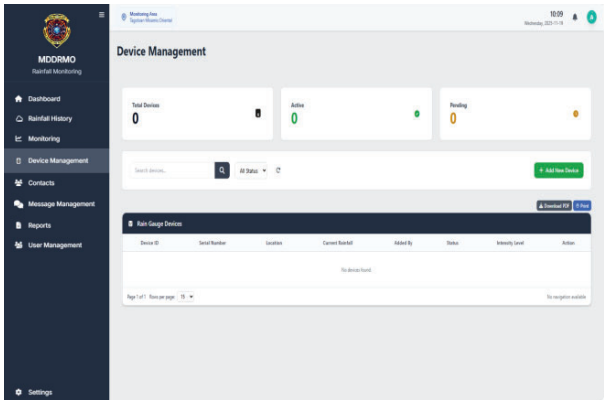


Figure 51 High Fidelity Device Management

The Device Management Interface provides an overview of all recorded devices in the system. It allows users to view detailed information about each device. Users can also edit or update specific device details to keep the records accurate and up to date.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14 Issue 12 , December - 2025

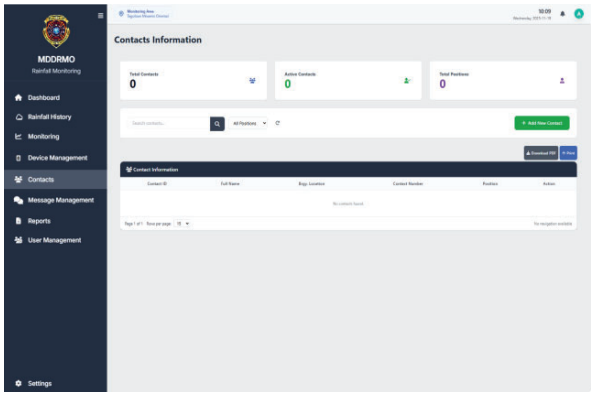


Figure 52 High Fidelity Contact Management

The Contact Management feature is designed to store and organize important contact information. It allows users to easily add, edit, and manage contact details. This ensures smooth communication and quick access to the right people, especially during disaster preparedness and response.

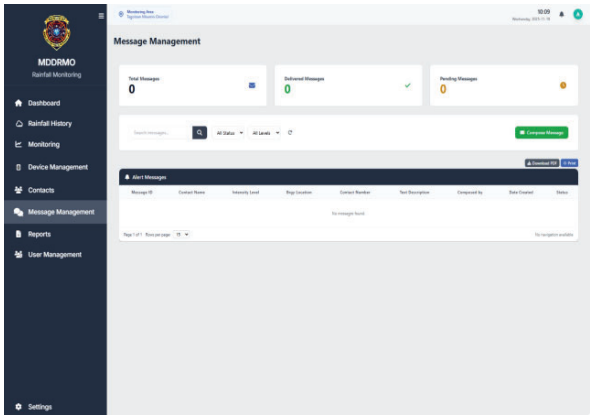


Figure 53 High Fidelity Message Management

The Message Management feature is designed to organize and manage all messages sent and received within the system. It allows users to view, send, and track messages efficiently. This ensures clear communication and faster response, especially during important operations or emergencies.

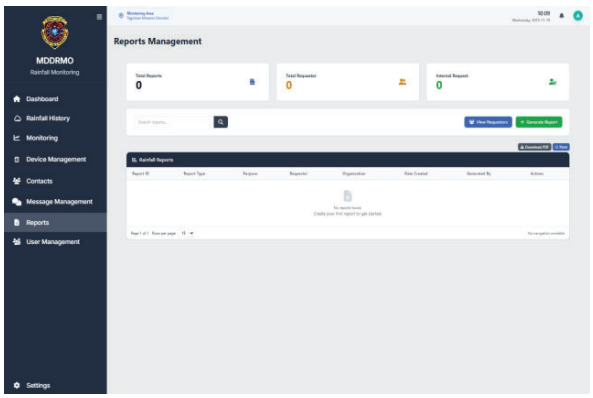


Figure 54 High Fidelity Report Management

Report Management feature is designed to store, organize, and manage generated reports within the system. It allows users to easily view, download, and track reports for analysis and documentation. This ensures better decision-making, transparency, and efficient record-keeping.

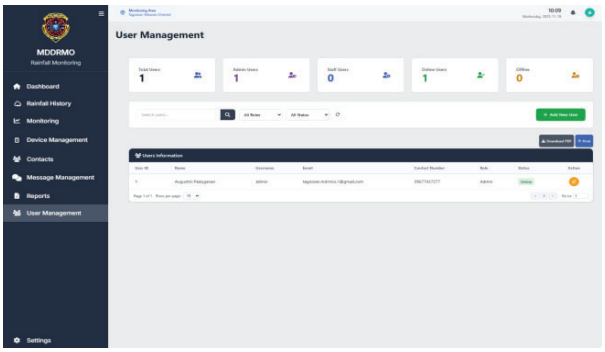


Figure 55 High Fidelity User Management

User Management interface is designed to manage all user accounts within the system. It allows admins to add, edit, update, or remove users as needed. This ensures proper access control, security, and smooth system operations.

Test

The functional testing aimed to verify whether each module of the system performed according to the specified requirements. It ensured that all features behaved correctly when used under both normal and boundary conditions. Developers participated in executing the test cases to observe real interactions with the system.

Table 10. Test Case Scenario

Test Case ID	Test Case Title	Expected Results	Actual Results	Criteria	Remarks
SCN FUNC-001	User Login with valid credentials.	This test verifies that users can successfully log in with correct credentials.	Users is authenticated and redirected to dashboard.	Users log in successfully using valid credentials.	Passed
SCN FUNC-002	Display Rainfall Data.	This test checks whether the system correctly displays real-time rainfall data from the sensor in the monitoring table.	Accumulated rainfall updates in real-time.	Data successfully displayed and updated.	Passed
SCN FUNC-003	User Add New Device (pending status change to active upon Rainfall Detection).	This test checks whether the system initially sets a newly added device to "Pending" status and automatically change the status to "Active" once the device detects rainfall data from its assigned location.	Device is added with pending status, Upon receiving first rainfall data status automatically updates to Active.	Device is added shown in the table. Upon triggered the device turn Active.	Passed
SCN FUNC-004	User Generate Rainfall Report for Specific Date Range	This test verifies whether the system can generate a downloadable rainfall report for a selected date range and location.	System generates and downloads a report containing rainfall data for the specified period.	Rainfall history generated successfully and ready to download	Passed
SCN FUNC-005	Admin add new contact	This test checks whether an admin can add a new contact to receive rainfall information.	Contact successfully added and appears in the contact list.	Contact added and shown in the contact table.	Passed
SCN FUNC-006	User login with Invalid Credentials.	This test checks whether the system prevents login with incorrect credentials.	Login fails with error message 'Invalid credentials'.	System show error message.	Passed
SCN FUNC-007	Admin Creates New User Account	This test verifies whether an admin can create new user accounts.	New user account is created and can log in.	New user added successfully and success message.	Passed

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14 Issue 12 , December - 2025

SCN FUNC-008	Admin Print an Overall reports	This test verifies whether the system allows the admin to print the overall reports covered all locations and rainfall data.	The system generates and prints the rainfall reports, showing complete, accurate.	The admin successfully prints the actual rainfall reports, displaying complete and accurate data for all monitored locations within the specified date range.	Passed
SCN FUNC-009	User customize profile	Admin customize profile.	The admin's profile should be successfully updated.	Profile updated successfully.	Passed
SCN FUNC-010	User changed date and time format	Verify that the user can successfully configure and save their preferred date and time format.	The system saves the selected date and time format and displays a success message.	Date and time updated 24hour format and date of "30/10/2025" .	Passed
SCN FUNC-011	Admin Updates Chart for Specific Location	This test checks whether the system allows the admin to update or refresh the rainfall chart for a specific monitoring location to reflect the most recent rainfall data.	Chart refreshes to display updated accumulated rainfall.	Updated chart are displayed.	Passed
SCN FUNC-012	Detection of High Rainfall Intensity	This test checks if the system detects and display a "High Intensity" alert exceeds a set threshold.	System displays "high Intensity".	The Intensity alert appear in the dashboard.	Passed
SCN FUNC-013	User Change Password	User changes account password.	The system successfully updates the user's passwords and displays a confirmation message.	The admin successfully change password.	Passed
SCN FUNC-014	User Analyze trend analysis	User views rainfall history and trend analysis data.	The system displays accurate rainfall data trends in chart and graphs.	History and trend analysis charts/graphs are display correctly with accurate data.	Passed
SCN FUNC-015	User Add contact number in existing contact name	Users adds contact number in existing contact's information.	The new contact number is added and displayed in the contact list table	Contact number successfully added to the table list.	Passed
SCN FUNC-016	User Log out	User logs out form the MDDRMO Rainfall Monitoring System.	System logs out the admin and redirects to the Login Page. The session in terminated.	The system immediately redirects Login Page.	Passed
SCN FUNC-017	Admin Updates Charts for Specific Location	This test checks whether the system allows the admin to update or refresh the rainfall chart for a specific monitoring location to reflect the most recent rainfall data.	Chart refreshes to display updated accumulated rainfall.	Updated chart are displayed.	Passed
SCN FUNC-018	Detection of High Rainfall Intensity	This test checks if the system detects and displays a "high intensity" alert when rainfall exceeds a set threshold.	System displays "High Intensity".	The Intensity alert appear in the dashboard	Passed
SCN FUNC-019	Send Notification When Rainfall Exceeds Threshold	This test verifies that the system automatically sends notifications to registered contacts when rainfall exceeds predefined threshold.	System notification to all registered contacts via configured channels.	The system send successfully send notification of a torrential level.	Passed
SCN FUNC-020	View Notification History	This test verifies whether users can view a log of all sent notifications.	System displays all sent notifications with timestamps and recipients.	Notification history displays complete and accurate records.	Passed
SCN FUNC-021	Admin Edits Device Information	This test verifies whether an admin can update device details such as location or serial number.	System displays all sent notifications with timestamps and recipients.	Notification history displays complete and accurate records.	Passed
SCN FUNC-022	Change password	User change account password.	System successfully updates the user password and display a confirmation message.	The admin successfully change password.	Passed
SCN FUNC-023	Configure Date and Time format	Verifies that the user can successfully configure and save their preferred date and time format in the system settings.	System saves the selected date and time format and displays a success message.	Date and time update to 24 hour format and date of "30/10/2025"	Passed
SCN FUNC-024	Sounds alerts	Users configures or test sound alerts.	Alerts sound configured to 50% of volume	The volume should be set to 50%	Passed
SCN FUNC-026	Arduino update message status to delivered.	This test verifies that the Arduino updated pending message to delivered.	Message status updates from "pending" to "sent"	The pending messages change to delivered.	Passed
SCN FUNC-027	Twitter Link Redirection	Verify that clicking the twitter icon redirects the user to the official Twitter Page.	Twitter page opens successfully in a new tab.	No page opens after clicking the Twitter Page.	Failed
SCN FUNC-028	User Facebook Link Redirection	Verify that clicking the Facebook icon redirects the user to the official Facebook page.	Facebook page opens in a new tab and loads successfully.	No page is opened after clicking the Facebook icon.	Failed
SCN FUNC-029	Forgot the Password via Email	Validate that the system sends a password rest link when a valid email is provided.	System displays confirmation message and sends reset link to registered email.	System displayed confirmation message, but no reset link was received in the email inbox.	Failed

Out of the 29 executed test cases, 26 passed while **3** failed during the initial testing phase. The failed test cases were specifically associated with Twitter and Facebook redirection features, as well as the email verification process for the Forgot Password function**.** These issues indicate that the external authentication connections and the email verification workflow require further debugging and configuration to ensure proper system functionality.

Most modules of the system including Login, Rainfall Data Display, Device Management, Report Generation, Notification Handling, Trend Analysis, and System Settings passed all their functional test scenarios. The system performed as expected across these features, with results matching the defined requirements. However, a few features, specifically the social media redirection functions such as the Twitter link, did not work as intended and were marked as failed, requiring further review and debugging.

These results show that the system's primary features such as rainfall monitoring, device management, data reporting, and notification services are functioning as intended and align with the functional requirements outlined in the study's specific objectives. The successful performance of these modules demonstrates that the system effectively supports accurate rainfall data collection, real-time monitoring, and efficient information dissemination, as originally targeted in the project goals.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14 Issue 12 , December - 2025

SYSTEM USABILITY STUDY RESULTS

Table 11. System Usability Scale (SUS) Raw Scale

System Usability Scale (SUS)	Participants				
	1	2	3	4	5
1. I think that I would like to use this system frequently.	5	5	4	4	4
2. I found the system unnecessarily complex.	1	2	2	1	2
3. I thought the system was easy to use.	5	4	5	4	3
4. I think that I would need the support of a technical person to be able to use this system.	1	2	2	1	2
5. I found the various functions in this system were well integrated.	5	4	5	4	5
6. I thought there was too much inconsistency in this system.	2	2	2	3	2
7. I would imagine that most people would learn to use this system very quickly.	5	5	5	4	4
8. I found the system very cumbersome to use.	2	2	2	2	2
9. I felt very confident using the system.	5	5	4	5	4
10. I needed to learn a lot of things before I could get going with this system.	2	2	1	2	2

The System Usability Scale (SUS) raw scores were collected from the participants to evaluate the usability of the Development of Automated Rain Gauge Monitoring System**.** The participants, consisting of the Administrator and staff responsible for rainfall monitoring, provided individual scores that formed the basis for the usability analysis.

The participants of the study were selected from the MDRRMO to ensure that feedback comes from personnel who are directly involved in operations and system use. Most participants are within the 25–44 age range, representing young to middle-aged adults who are actively engaged in their work roles. All participants are male, reflecting the composition of the personnel in the positions surveyed.

In terms of job roles, participants include Administrators, MCDH-1 officers, and Dispatch staff, all from the MDRRMO department, which ensures their familiarity with the systems and processes under study. Most have a Bachelor's degree, indicating a level of education suitable for understanding and effectively using the system.

Work experience varies widely, from less than 1 year to more than 10 years, providing perspectives from both new and experienced staff. Participants' technical proficiency is mostly intermediate, showing that they are comfortable with technology and use it regularly in their daily tasks.

All participants reported daily use of technology and daily interaction with systems similar to the one being tested. Their previous experience with similar systems is generally at the intermediate level, and they are comfortable providing feedback on system usability.

Table 12. System Usability Scale (SUS) Computed Score

System Usability Scale (SUS)	Participants				
	1	2	3	4	5
1. I think that I would like to use this system frequently.	4	4	3	3	3
2. I found the system unnecessarily complex.	4	3	3	4	3
3. I thought the system was easy to use.	4	3	4	3	2
4. I think that I would need the support of a technical person to be able to use this system.	3	3	3	2	3
5. I found the various functions in this system were well integrated	4	3	4	3	4
6. I thought there was too much inconsistency in this system.	3	3	3	2	3
7. I would imagine that most people would learn to use this system very quickly.	4	4	4	3	3
8. I found the system very cumbersome to use.	3	3	3	3	3
9. I felt very confident using the system.	4	4	3	4	3
10. I needed to learn a lot of things before I could get going with this system.	3	3	4	3	3
Total Score:	36	33	34	30	30

The System Usability Scale (SUS) is scored by adjusting each participant's answers according to a simple set of rules. For the odd-numbered questions (which are positively worded), the score is calculated by subtracting 1 from the participant's raw response. For the even-numbered questions (which are negatively worded), the score is found by subtracting the raw response from 5. This adjustment ensures that each question contributes a value between 0 and 4, making all items consistent in their impact on the final score. Once all ten items are adjusted, their scores are added together to get a total out of 40. To convert this into a more familiar 0–100 scale, the total is multiplied by 2.5. This standardized approach helps balance both positive and negative feedback, giving a clear and fair picture of how usable a system really is.

For Participant 1, the total score was 36, which translated into high adjusted values across most items, particularly on questions related to ease of use and system navigation. Participants 2 and 3, with raw totals of 33 and 34 respectively, also demonstrated favorable usability ratings, indicating that they generally found the system understandable and straightforward, while suggesting minor improvements for clearer data interpretation. On the other hand, Participants 4 and 5 both recorded raw totals of 30, and although these are slightly lower, their adjusted SUS scores still fall within an acceptable usability range. Their responses indicate that while the system functions well, they encountered a few areas that may require refinement, such as handling duplicate entries and providing clearer instructions to make it easier for users to understand and interpret data for alert purposes.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14 Issue 12 , December - 2025

Table 13. Overall Mean SUS Score

Participants	Total Score	SUS Score (2.5)
Participants 1	36	90
Participants 2	33	82.5
Participants 3	34	85
Participants 4	30	75
Participants 5	30	75
Total		407.5 / 5
Overall Mean SUS Score		81.5

The System Usability Scale (SUS) results show an overall average score of 81.5, which is considered "excellent" according to standard SUS guidelines. This means users generally found the Rain Gauge Monitoring System easy to use, intuitive, and efficient. The high score suggests that the system supports user tasks well and meets expectations for clarity and functionality. The high SUS score reflects that the system's interface and workflow align well with standard usability principles.

The individual SUS scores show that all five participants rated the system above 75, with Participant 1 giving the highest score of 90, followed by Participants 3 and 2 with scores of 85 and 82.5, respectively. These results indicate that the participants found the system understandable and experienced minimal difficulty in navigating its features. Meanwhile, Participants 4 and 5 both recorded a score of 75. Their ratings suggest that although the system is functioning well, they may have encountered certain areas that required clearer interpretation to understand the system well.

During the SUS activity, most participants found the system easy to navigate, and they were able to explore its features without difficulty. They also appreciated the system's consistency in design and workflow. However, some users experienced slight hesitation when inputting data, particularly when entering information for the same location. Since the system does not allow duplicate locations, this caused minor delays as participants had to review and adjust their inputs before proceeding.

The SUS results show that the Rain Gauge Monitoring System is not only functional but genuinely comfortable for users to work with. Participants clearly found it easy to move around the system, and most tasks felt straightforward and manageable.

5. SUMMARY, FINDINGS, CONCLUSION, AND RECOMMENDATIONS

Summary

This chapter provides a brief overview of the study, which focused on developing an Automated Rain Gauge Monitoring System to help the MDRRMO collect real-time rainfall data and maintain accurate historical records. The system combined hardware components such as the Arduino Uno, GSM module, and tipping bucket rain gauge with software developed using Laravel, VS Code, Arduino IDE, and tools like HTML, CSS, Tailwind, Alpine.js, Leaflet.js, and Chart.js, with MySQL as the database. The Agile methodology guided the development process, allowing the team to improve the system through iterative and repetitive cycles. The system evaluation shared results from both functional testing and the System Usability Scale (SUS), along with conclusions and recommendations. The way the system was built and tested provided a clear summary of what was learned and how the findings can be applied in real-world situations.

Findings

The system implementation and evaluation revealed several key findings. The overall mean SUS score was 81.5, which is classified as "excellent" and indicates that users found the system usable, intuitive, and efficient. All five participants scored above 75, with Participant1 achieving the highest rating of 90, while Participants 4 and 5 both scored 75, suggesting minor areas for improvement. The system's interface and workflow were consistently rated as clear and straightforward, supporting user tasks effectively. Most users reported that the system was easy to navigate, with minimal difficulty in performing core functions like logging in.

Patterns in the data show that the system's usability was strong across all user roles and experience levels, reflecting broad acceptance and ease of use. No major unexpected results were observed, and all feedback aligned with the expectation that the system would be accessible and functional for daily monitoring tasks. The system proved effective in addressing the need for efficient rainfall data management, with high usability scores and positive user comments on clarity and integration. Key metrics such as the SUS score (81.5), individual ratings, and user feedback collectively summarize the system's strong performance in usability and functionality.

Conclusion

The system was designed to work seamlessly with the Tipping Bucket Rain Gauge (TBRG), ensuring accurate rainfall measurements. Hardware and software components automatically recorded each bucket tip, capturing precise data without manual input. Testing confirmed consistent performance and reduced errors compared to traditional methods.

The web interface displays live charts, historical trends, and location maps, making data easy to understand. Usability feedback highlighted the interface as user-friendly and effective for daily operations. Real-time visualization supports transparency and enables faster decision-making for disaster preparedness.

Data collection and storage are fully automated, eliminating manual entry and calculations. Historical data

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14 Issue 12 , December - 2025

is securely saved in a database, allowing for trend analysis and reporting over time. Minor issues with duplicate location entries were resolved with system rules and clear instructions, ensuring smooth data management.

The system provides timely, accurate rainfall information to support decision-making, disaster preparedness, and response. Real-time data helps teams make informed choices during emergencies, improving accuracy and reducing delays. User feedback confirms the system's reliability and practical value in addressing operational challenges and enhancing community safety.

Recommendations

To keep the Rain Gauge Monitoring System working well in real-world conditions, a few key improvements can make a big difference. Focusing on better data accuracy, greater flexibility, and easier usability will help the system stay reliable and effective over time. By making these changes, the system can adapt to new needs, support users more efficiently, and deliver better information for making important decisions.

For long-term sustainability and further growth, the system should incorporate continuous improvements and integration options. This includes routine maintenance, updates for new features, and potential integration with other disaster management platforms.

In future updates, the system could be improved to reach more users and offer even more helpful features. Providing proper training, ongoing support, and clear guidelines will help staff feel confident using the system and get the most out of its tools. This way, the system can remain effective, adaptable, and valuable for disaster management teams as needs evolve.

Allowing the system to integrate with other sensors and platforms gives users a more complete picture of local conditions. This holistic approach supports smarter decisions and more resilient planning for both routine operations and emergency situations.

References

[1] Afandi, M., Rokhayadi, D., Budiarto, R., Hermawan, D., & Nugroho, Y. (2024). Rainfall monitoring using Aloptama Automatic Rain Gauge and the Network Development Life Cycle Method. Sinkron, 9(1), 2429–2435. https://jurnal.polgan.ac.id/index.php/sinkron/article/view/13908/2844

[2] Balaji, M., & Banupriya, P. (2023). IoT Based Rainfall Monitoring System. IRE Journals, 6(10), 232–236. Retrieved from https://www.irejournals.com/paper-details/1703807

[3] Danendra, D. P., Oktafandi, K. A., Mujiyanti, S. F., & Lasminto, U. (2023b). Integration of data acquisition system for rainfall measurement on IoT-based tipping bucket rain gauge. Journal of Physics Conference Series, 2673(1), 012033. https://doi.org/10.1088/1742-6596/2673/1/012033

[4] Faisal, M., Ramadhani, Z. M., Hidayat, S. A. M., Basrida, A. R., Fazrin, M. T., & University of Raharja. (2023). Prototype of water level and rainfall Detection system as flood warning based on Blynk IOT application. International Transactions on Education Technology (ITEE), 1–1, 1–10 https://journal.pandawan.id/itee/article/view/361/358

[5] Fajar, M., Chandra, F. B., & Arfandy, H. (2023). Applying use case 2.0 approach to the development of IoT-Based rainfall monitoring system. Jurnal Nasional Pendidikan Teknik Informatika (JANAPATI), 12(2), 244–252. https://doi.org/10.23887/janapati.v12i2.61529

[6] Fernandes, L. F., Soares, M., & Endo, P. T. (2020). Interoperability and integration testing methods for IoT systems: A systematic mapping study. arXiv preprint arXiv:2007.11308. https://arxiv.org/abs/207.11308

[7] Hu, J., Jothi, N., Rajagopal, H., Yeo, S. F., Institute of Computer Science and Digital Innovation, UCSI University, Faculty of Business, Multimedia University, & Graduate Business School, UCSI University. (2023). Real-Time Weather Data for Environment, Social, and Governance (ESG) Decision-Making. Journal of Robotics, Networking and Artificial Life, 10(1), 96–104. https://www.jstage.jst.go.jp/article/jrnal/10/1/10_15/_pdf

[8] Kennedy, G. C., Luckyn, B. J., Jaja, T. T., & Idahtonye, T. (2022). Design of an Internet of Things Rain Detector Device with GSM Notification. IRE Journals, 6(3), 133–134. https://www.irejournals.com/formatedpaper/1703807.pdf

[9] Kumari, R., Sah, D. K., Cengiz, K., Ivković, N., Gehlot, A., & Salah, B. (2023b). Acoustic signal-based indigenous real-time rainfall monitoring system for sustainable environment. Sustainable Energy Technologies and Assessments, 60, 103398. https://doi.org/10.1016/j.seta.2023.103398

[10] Maddikera, K., Araveti, S., Meghana, Vaishnavi Arun, T., Vineetha, S., Shanthi, K., & Student, U. (2023). Rainfall and Weather Monitoring System using Internet of Things (IoT). 22(03). https://doi.org/10.10543/f0299.2023.41543

[11] Mali, S., Gunjan, Dr. Prof. R., Shete, Dr. Prof. V. V., & Dr. Selva Balan. (2022). IOT based tipping Bucket rain Gauge Data Processing System. International Journal of Advances in Engineering and Management (IJAEM), 4(12), 513–517. https://doi.org/10.35629/5252-0412513517

[12] Megantoro, P., et al. (2021, June 1). Real-time monitoring system for weather and air pollutant measurement with HTML-based UI application. Bulletin of Electrical Engineering and Informatics. https://beei.org/index.php/EEI/article/view/3030/2211

[13] Moedt, W., Bernsteiner, R. C., Hall, M., & Fruhling, A. L. (2022). Enhancing IoT Project Success through Agile Best Practices. ACM Transactions on Internet of Things. https://doi.org/10.1145/3568170

[14] Mouha, R. A. (2021). Internet of Things (IoT). Journal of Data Analysis and Information Processing, 9(2), 77–101. https://www.scirp.org/journal/paperinformation.aspx?paperid=108574

[15] Ortega-Gonzalez, L., et al. (2021). Communication protocols evaluation for a wireless rainfall monitoring network in an urban area. Heliyon, 7(6), e07353. https://doi.org/10.1016/j.heliyon.2021.e07353

[16] Palestine Polytechnic University, Shaheen, N., & Manasra, K. (2022). IOT – based weather reporting system. https://scholar.ppu.edu/bitstream/handle/123456789/8956/IOT%20%E2%80%93%20based%20weather%20reporting%20system.pdf

[17] Panganiban, E. B. (2020). Rainfall measurement and flood warning systems: A review. International Journal of Scientific & Technology Research, 9(3), 244–254. https://www.researchgate.net/publication/363796595

[18] Rohmah, D. Z., et al. (2024). Development of an automatic portable calibrator for tipping bucket rain gauge (TBRG). Jurnal Penelitian Pendidikan IPA, 10(9), 6746–6755. https://doi.org/10.29303/jppipa.v10i9.7634

[19] Salcedo, E. (2024). Graph learning-based regional heavy rainfall prediction using low-cost rain gauges. Universidad Católica Boliviana "San Pablo". https://arxiv.org/pdf/2412.16842

[20] Saniel, A. L., et al. (2024). Development and Optimization of IoT-Based Weighing Rain Gauge. Chemical Engineering Transactions, 114, 679-684. https://doi.org/10.3303/CET24114114

[21] Shaheen, H., & Manasra, A. (2022). A comparison of PHP and Node.js. International Journal of Advanced Computer Science and Applications, 13(5), 190–196. https://doi.org/10.14569/IJACSA.2022.0130523

[22] Sharma, U., et al. (2025). Interactive weather forecasting system using OpenWeather API. International Journal of Research - GRANTHAALAYAH, 13(1), 130–135. https://www.granthaalayahpublication.org/journals/granthaalayah/article/view/6119/5917

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14 Issue 12 , December - 2025

[23] Strandberg, P. E., et al. (2020). Intermittently failing tests in the embedded systems domain. arXiv. https://arxiv.org/pdf/2005.06826

[24] Tan, J., Chen, Y., & Jiao, S. (2023). Visual Studio Code in introductory computer science course. arXiv preprint arXiv:2303.10174. https://arxiv.org/abs/2303.10174

[25] Tong, W., et al. (2022). A review of software acceptance testing techniques. International Journal of Advanced Computer Science and Applications, 13(3), 487–495. https://www.researchgate.net/publication/368491204

[26] View of an arduino-based disaster management system. (n.d.). JRC. https://journal.umy.ac.id/index.php/jrc/article/view/8711/5516

[27] View of Community-based flood alert system using long-range technology. (n.d.). MJSSDS. https://journal.omsc.edu.ph/index.php/mjssds/article/view/10/41

[28] View of INTERACTIVE WEATHER FORECASTING SYSTEM USING OPENWEATHER API. (n.d.). International Journal of Research -GRANTHAALAYAH. https://www.granthaalayahpublication.org/journals/granthaalayah/article/view/6119/5918

[29] View of IoT-Based rainfall monitoring system for Chili farming land. (n.d.). Brilliance Journal. https://jurnal.itscience.org/index.php/brilliance/article/view/3649/2863

[30] View of rainfall monitoring using Aloptama Automatic Rain Gauge. (n.d.). Sinkron Journal. https://jurnal.polgan.ac.id/index.php/sinkron/article/view/13908/2844

[31] View of telemetering of rainfall measurement results. (n.d.). Infotel Journal. https://ejournal.ittelkom-pwt.ac.id/index.php/infotel/article/view/603/341

[32] Zamarro, J., et al. (2025). Arduino-Based Rainfall and Flood Monitoring System. International Journal of Scientific Research & Engineering Trends, 11(1), 2395–2566. https://ijsret.com/wp-content/uploads/2025/01/IJSRET_V11_issue1_196.pdf

Book

[33] GeeksforGeeks. (2023, November 24). Use Case Diagram. GeeksforGeeks. https://www.geeksforgeeks.org/use-case-diagram/

[34] Jofel Batutay. (2023, December 23). Designing and Implementing Embedded Systems Using Flowchart-Based Algorithms: Microcontroller-based Project Examples. https://doi.org/10.13140/RG.2.2.33042.02246

[35] QUISPE, M. (2025, March 21). Software Engineering A Practitioners Approach by Roger Pressman, Bruce Maxim (1). Academia.edu. https://www.academia.edu/128349723/Software_Engineering_A_Practitioners_Approach_by_Roger_Pressman_Bruce_Maxim_1_?utm

[36] Systems Engineering Technical Reviews and Audits | www.dau.edu. (2025). Dau.edu. Systems Engineering Technical Reviews and Audits | www.dau.edu

[37] World Meteorological Organization. (2021). WMO atlas of mortality and economic losses from weather, climate and water extremes (1970–2019) (WMO-No. 1267). https://library.wmo.int/viewer/57564/download?file=1267_Atlas_of_Mortality_en.pdf&type=pdf&navigator=1

APPENDIX – A

Scanned Copy of the Received Letter of the Client

TAGOLOAN COMMUNITY COLLEGE
BALUARTE, TAGOLOAN, MISAMIS ORIENTAL
COLLEGE OF INFORMATION TECHNOLOGY

February 27, 2025

AUGUSTINI PADUGANAN
Engineer
MDRRMO
Tagoloan Misamis Oriental

Dear Sir:

Greetings of peace and blessings.

My name is **Harold Jay S. Variacion**, and I am a 3rd Year student at Tagoloan Community College, currently enrolled in the IT Capstone Project 01 course under Bachelor of Science in Information Technology. I am the Project Manager for **Stars Crew**, working with the following team members:

- Mark Anthony Bugabod
- Kaye Francine A. Tubio
- Jesuriel Rangas
- Nathaniel D. Tilbor

The working title of our Capstone Project is **Development of automated Rain Gauge Monitoring System** and I would like to ask if we may conduct a face-to-face interview with you in your most convenient time.

The purpose of this interview is to understand the organization/business' requirements, expectations, and any specific concerns regarding the project. Specifically, I aim to gather information on:

- Understanding your organization's/business, operations, and current IT usage.
- Identifying challenges, inefficiencies, and pain points in current processes.
- Determining the target users and beneficiaries of the system.
- Exploring possible IT-based solutions that align with your expectations.
- Defining the purpose and impact of the project.
- Clarifying the system's scope, features, constraints, and possible limitations.
- Understanding system workflow, data flow, key transactions, and interactions.
- Paper forms or documents involved and relevant in the process.

Rest assured that confidentiality will be strictly maintained during and after the interview. Should you have any questions please do not hesitate to contact me at **variacionharoldjay7@gmail.com** or **09526313803**. I look forward to your positive response with regards to this matter. Thank you and God bless!

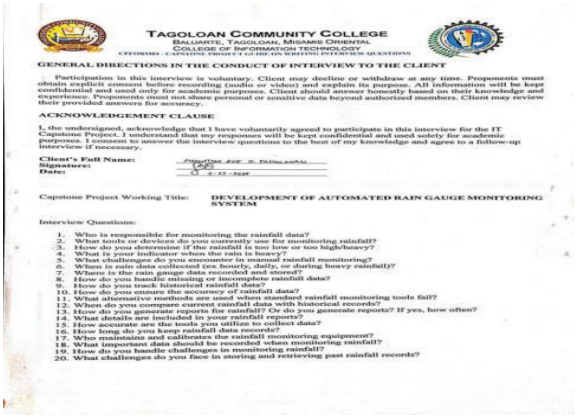
Always,

HAROLD JAY S. VARIACION
Project Manager

Noted by:

NEPTALE S. ROA III, MIT
Capstone Project 01 Coordinator / Instructor

Scanned Copy of the Interview Guide Questions



TAGOLOAN COMMUNITY COLLEGE
BALUARTE, TAGOLOAN, MISAMIS ORIENTAL
COLLEGE OF INFORMATION TECHNOLOGY

GENERAL DIRECTIONS IN THE CONDUCT OF INTERVIEW TO THE CLIENT

Participation in this interview is voluntary. Client may decline or withdraw at any time. Proponents must obtain explicit consent before recording (audio or video) and explain its purpose. All information will be kept confidential and used only for academic purposes. Client should answer honestly based on their knowledge and experience. Proponents must not share personal or sensitive data beyond authorized members. Client may review their provided answers for accuracy.

ACKNOWLEDGEMENT CLAUSE

I, the undersigned, acknowledge that I have voluntarily agreed to participate in this interview for the IT Capstone Project. I understand that my responses will be kept confidential and used solely for academic purposes. I consent to answer the interview questions to the best of my knowledge and agree to a follow-up interview if necessary.

Client's Full Name:
Signature:
Date:

Capstone Project Working Title: **DEVELOPMENT OF AUTOMATED RAIN GAUGE MONITORING SYSTEM**

Interview Questions:

1. Who is responsible for monitoring the rainfall data?
2. What tools or devices do you currently use for monitoring rainfall?
3. How do you determine if the rainfall is too low or too high/heavy?
4. What is your indicator when the rain is heavy?
5. What challenges do you encounter in manual rainfall monitoring?
6. When is rain data collected (ex. hourly, daily, or during heavy rainfall)?
7. Where is the rain gauge data recorded and stored?
8. How do you handle missing or incomplete rainfall data?
9. How do you track historical rainfall data?
10. How do you ensure the accuracy of rainfall data?
11. What alternative methods are used when standard rainfall monitoring tools fail?
12. How do you compare current rainfall data with historical records?
13. How often do you generate reports for rainfall? Or do you generate reports? If yes, how often?
14. What details are included in your rainfall reports?
15. How accurate are the tools you utilize to collect data?
16. How long do you keep rainfall data records?
17. Who maintains and calibrates the rainfall monitoring equipment?
18. What important data should be recorded when monitoring rainfall?
19. How do you handle challenges in monitoring rainfall?
20. What challenges do you face in storing and retrieving past rainfall records?

APPENDIX – B

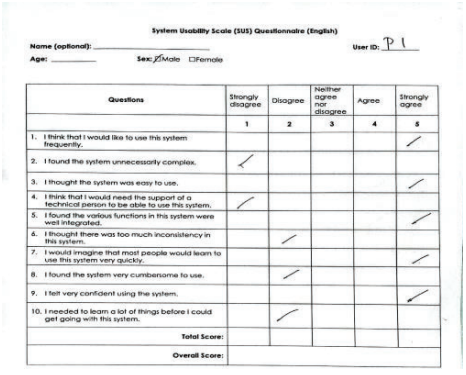
Figure of hardware technology used from Client

Tipping Bucket Rain Gauge (TBRG)

Rain Gauge Display unit

APPENDIX – C

System Usability Scale Forms



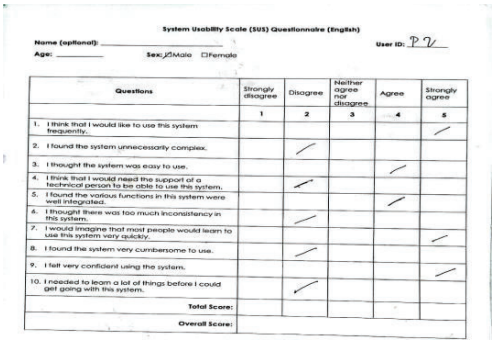
System Usability Scale (SUS) Questionnaire (English)

Name (optional): _____ User ID: P1
Age: _____ Sex: ☒Male ☐Female

Questions	Strongly disagree	Disagree	Neither agree nor disagree	Agree	Strongly agree
	1	2	3	4	5
1. I think that I would like to use this system frequently.					✓
2. I found the system unnecessarily complex.	✓				
3. I thought the system was easy to use.					✓
4. I think that I would need the support of a technical person to be able to use this system.	✓				
5. I found the various functions in this system were well integrated.					✓
6. I thought there was too much inconsistency in this system.		✓			
7. I would imagine that most people would learn to use this system very quickly.					✓
8. I found the system very cumbersome to use.		✓			
9. I felt very confident using the system.					✓
10. I needed to learn a lot of things before I could get going with this system.		✓			
Total Score:					
Overall Score:					

SUS-Participants 1

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14 Issue 12 , December - 2025



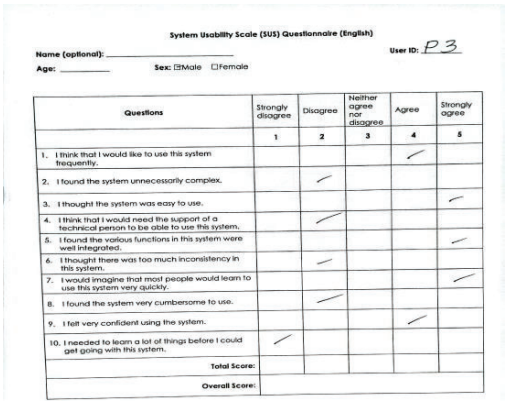
System Usability Scale (SUS) Questionnaire (English)

Name (optional): _____ User ID: P2

Age: _____ Sex: ☑Male ☐Female

Questions	Strongly disagree	Disagree	Neither agree nor disagree	Agree	Strongly agree
	1	2	3	4	5
1. I think that I would like to use this system frequently.					✓
2. I found the system unnecessarily complex.		✓			
3. I thought the system was easy to use.				✓	
4. I think that I would need the support of a technical person to be able to use this system.		✓			
5. I found the various functions in this system were well integrated.				✓	
6. I thought there was too much inconsistency in this system.		✓			
7. I would imagine that most people would learn to use this system very quickly.					✓
8. I found the system very cumbersome to use.		✓			
9. I felt very confident using the system.					✓
10. I needed to learn a lot of things before I could get going with this system.		✓			
Total Score:					
Overall Score:					

SUS-Participants 2



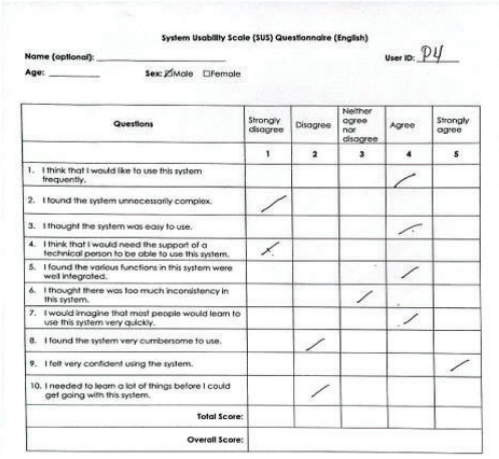
System Usability Scale (SUS) Questionnaire (English)

Name (optional): _____ User ID: P3

Age: _____ Sex: ☒Male ☐Female

Questions	Strongly disagree	Disagree	Neither agree nor disagree	Agree	Strongly agree
	1	2	3	4	5
1. I think that I would like to use this system frequently.				✓	
2. I found the system unnecessarily complex.		✓			
3. I thought the system was easy to use.					✓
4. I think that I would need the support of a technical person to be able to use this system.		✓			
5. I found the various functions in this system were well integrated.					✓
6. I thought there was too much inconsistency in this system.		✓			
7. I would imagine that most people would learn to use this system very quickly.					✓
8. I found the system very cumbersome to use.		✓			
9. I felt very confident using the system.				✓	
10. I needed to learn a lot of things before I could get going with this system.	✓				
Total Score:					
Overall Score:					

SUS-Participants 3



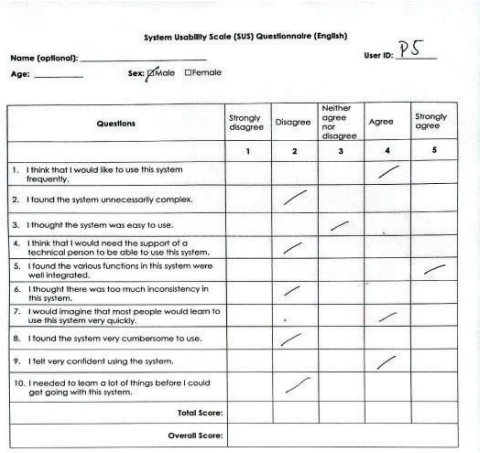
System Usability Scale (SUS) Questionnaire (English)

Name (optional): _____ User ID: P4

Age: _____ Sex: ☑Male ☐Female

Questions	Strongly disagree	Disagree	Neither agree nor disagree	Agree	Strongly agree
	1	2	3	4	5
1. I think that I would like to use this system frequently.				✓	
2. I found the system unnecessarily complex.	✓				
3. I thought the system was easy to use.				✓	
4. I think that I would need the support of a technical person to be able to use this system.	✓				
5. I found the various functions in this system were well integrated.				✓	
6. I thought there was too much inconsistency in this system.			✓		
7. I would imagine that most people would learn to use this system very quickly.				✓	
8. I found the system very cumbersome to use.		✓			
9. I felt very confident using the system.					✓
10. I needed to learn a lot of things before I could get going with this system.		✓			
Total Score:					
Overall Score:					

SUS-Participants 4



System Usability Scale (SUS) Questionnaire (English)

Name (optional): _____ User ID: P5

Age: _____ Sex: ☑Male ☐Female

Questions	Strongly disagree	Disagree	Neither agree nor disagree	Agree	Strongly agree
	1	2	3	4	5
1. I think that I would like to use this system frequently.				✓	
2. I found the system unnecessarily complex.		✓			
3. I thought the system was easy to use.			✓		
4. I think that I would need the support of a technical person to be able to use this system.		✓			
5. I found the various functions in this system were well integrated.					✓
6. I thought there was too much inconsistency in this system.		✓			
7. I would imagine that most people would learn to use this system very quickly.				✓	
8. I found the system very cumbersome to use.		✓			
9. I felt very confident using the system.				✓	
10. I needed to learn a lot of things before I could get going with this system.		✓			
Total Score:					
Overall Score:					

SUS-Participants 5