Journal of Computing & Biomedical Informatics
ISSN: 2710 - 1606
Volume 10 Issue 02
2026

Research Article
https://doi.org/10.56979/1002/2026/1336

Iot-Based Rain Detector System Using Arduino ESP-32

Syed Asim Ali Shah[1], Imran Haider[2], Syeda Asma Bibi[3], Ahmad Shalaldeh[4], Mishal Khalid[5], and Ayoub Alsarhan[4&6]

[1]Fakulti Pengurusan Teknologi Dan Teknousahawanan, Universiti Teknikal Malaysia Melaka, Kampus Teknologi, Melaka, 76100, Malaysia.
[2]Department of Computer Science, Khwaja Fareed University of Engineering and Information Technology [KFUEIT], Sub-Campus Rajanpur 33500, Punjab, Pakistan.
[3]Department of Computer Science, Islamia University Bahawalpur [IUB], 33500, Rajanpur, Punjab, Pakistan.
[4]Department of Data Science and Artificial Intelligence, Faculty of Information Technology, Al-Ahliyya Amman University, Al Salat 19111, Jordan.
[5]Department of Computer Science, Khwaja Fareed University of Engineering and Information Technology [KFUEIT], Sub-Campus 33500, Rajanpur, Punjab, Pakistan.
[6]Department of Information Technology, Faculty of Prince Al-Hussein Bin Abdallah II for Information Technology, The Hashemite University, Zarqa, Jordan.
*Corresponding Author: Syeda Asma Bibi. Email: syedaasmabibi661@gmail.com

Received: November 22, 2025 **Accepted:** February 25, 2026

Abstract: Is it feasible to automate the rain detector using an Arduino ESP-32 IoT device to enhance its performance? The Internet of Things [IOT] is changing weather practices and human life. Researchers are implementing IoT in weather practices. The development and statistical testing of an IoT-based rain detection module using the Arduino ESP32 microcontroller was the main focus of this study to enhance weather-based automation in smart environments. It was hypothesized that digital sensors and controllers could be interconnected to automate real-life systems, such as detecting rainfall and responding accordingly. This research aimed to provide a solution for urban areas where real-time rain monitoring is essential, especially for smart irrigation or safety systems. By implementing a rain detector system based on the IoT approach using the ESP32, the study offers a reliable, low-cost, and automated method to detect rain, trigger alerts, and integrate with other smart devices. An ESP-32 microcontroller was adapted and tested to collect real-time environmental data, specifically focused on detecting rainfall, humidity, temperature, and atmospheric conditions. Through its integrated Android application, the ESP-32 microcontroller enables users to monitor weather conditions remotely, providing real-time environmental updates and data analysis. This system supports better decision-making in weather-sensitive areas, automating responses such as halting irrigation or activating covers during rainfall. It has the potential to reduce water wastage by 30% and minimize weather-related damage by providing timely alerts through automatic rain detection. Using this IoT-based project, users, particularly in urban and rural areas, can manage outdoor systems more efficiently without relying on manual monitoring. It is recommended to implement this IoT-based rain-detection solution across both urban and rural agricultural environments to enhance weather management and disaster preparedness.

Keywords: IOT Rain Detection System ESP-32; Wireless Rain Sensor; Weather Monitoring; Home Automation

1. Introduction

The Internet of Things (IoT) refers to a network of connected physical devices that communicate over the internet. These devices gather, send, and receive data without needing direct human input [1]. Often called "smart" devices, they include sensors, household appliances like smart lights and air conditioning units, wearable tech such as smartwatches, cars, and industrial equipment [2]. IoT makes it easier to monitor, automate, and manage these devices remotely using technologies like Wi-Fi, Bluetooth, and mobile networks. With features such as automated tasks, data tracking for various parameters like temperature, motion, and humidity, and access through cloud services or mobile apps, IoT has numerous applications [3]. It is used in smart homes, healthcare, agriculture, and smart city projects, improving efficiency, convenience, and decision-making in personal and industrial environments.

The Internet of Things (IoT) has transformed how we observe and engage with our surroundings. An IoT-based rain detection system utilizing the Arduino ESP32 offers an intelligent approach to real-time rain detection, transmitting data to connected devices or cloud services for monitoring and automation. This system is crucial for weather observation, agricultural practices, smart irrigation, and flood mitigation [4].

At the core of this system is the ESP32 micro-controller. This economical, energy-efficient board features built-in Wi-Fi and Bluetooth, making it perfectly suited for IoT applications. A rain sensor module, such as the YL-83 or FC-37, is employed to identify whether it is raining or to gauge its intensity [5]. Upon detecting rainfall, the ESP32 can send notifications over the internet, log data on cloud platforms like Thingspeak, Blynk, or Firebase, or initiate automation processes like shutting windows, deploying covers, or halting irrigation. This advanced rain detection system removes the necessity for manual checks and aids in the efficient use of resources, particularly in both urban and rural settings [6]. With capabilities such as real-time notifications, wireless connectivity, and cloud-based data logging, the system fosters sustainable environmental management and promotes the evolution of smart cities and weather systems.

2. Literature overview

In recent years, IoT has been increasingly applied to rain detection systems to improve real-time weather monitoring and automation [7]. The concept is straightforward: rainfall impacts many daily activities and operations, yet people often lack timely or accurate information. With IoT, detecting rain and responding to it can be automated. Small controllers like the Arduino ESP32 are commonly used in such systems due to their built-in Wi-Fi and low power consumption [8]. These systems can sense rainfall using modules like the YL-83 or FC-37 and send data to cloud platforms, making monitoring accessible from anywhere. Emerging technologies such as AI, blockchain, and edge computing are also being explored to enhance rain detection accuracy, data security, and decision-making, especially for applications in smart cities and disaster management [9].

However, there are still challenges. Power consumption can be an issue, especially in remote installations. Rain sensors may degrade over time or produce inaccurate readings in harsh weather. Moreover, scaling these systems from small-scale projects to city-wide or agricultural or weather applications requires robust infrastructure and reliable connectivity [10]. Addressing these challenges is essential to make IoT-based rain detection systems more widespread and dependable [11].

2.1. System Architectures and Components:

Advanced projects may include smart features like real-time notifications, automatic roof covers, or alerts to stop irrigation when it rains. Actuators such as servo motors, relays, and buzzers respond to rain detection, making the system more interactive. While many systems are automated, some may still need manual overrides or calibration for precision [12]. Together, these components create an efficient, scalable IoT-based rain detection network.

Most IoT-based rain detector systems use microcontroller boards like Arduino UNO, NodeMCU (ESP32), or Raspberry Pi. Each board serves specific roles depending on how complex the system is. The Arduino UNO is ideal for beginners because of its simplicity and compatibility with basic rain sensors like YL-83 or FC-37, which detect raindrops on a sensing plate. NodeMCU boards (ESP32) are more advanced and offer built-in Wi-Fi and Bluetooth, making them suitable for projects that need real-time data transfer to cloud platforms like Blynk or ThingSpeak.

For more powerful applications, you can use a Raspberry Pi. It acts as a mini-computer or a local server that processes and stores rain data locally before uploading it. The rain detection system may also include other environmental sensors, such as temperature, humidity (DHT11/DHT22), or wind speed sensors for better weather monitoring [13].

2.2. IoT Communication and Cloud Integration:

Cloud platforms are essential for collecting, visualizing, and analyzing rainfall data. For example, ThingSpeak can work with MATLAB to find trends or patterns in rainfall. This helps with weather forecasting and flood management. Firebase provides real-time database features, making it perfect for mobile apps. Platforms like Blynk let users create control dashboards with little coding [14]. This allows them to get notifications and track rainfall conditions through their smartphones. With these technologies, people can monitor rainfall and environmental conditions from almost anywhere [15].

Various rain detection systems utilize different techniques for data transmission and management. Wi-Fi is the most commonly used option since it is available in most households, and micro-controller boards such as NodeMCU and ESP32 can easily connect to it. These boards are capable of transmitting real-time rainfall information to cloud platforms, including Firebase or ThingSpeak. To facilitate efficient communication between devices, MQTT is often employed due to its lightweight and dependable nature [16].

In rural or remote regions with limited internet connectivity, GSM or GPRS modules are preferred. Though these options may not enable continuous data transmission, they can still send SMS notifications when rainfall is detected, allowing users to take immediate actions, such as halting irrigation or deploying protective covers [17].

2.3. Automation and Control Strategies:

Automation in rain detection systems can range from simple to highly advanced methods [18]. The most basic approach is threshold-based control, where the system triggers actions based on specific sensor readings. For example, if rainfall exceeds a certain level, an alert is sent, or a window is automatically closed. These straightforward rules are effective and widely used. More advanced rain detection systems incorporate fuzzy logic, where multiple environmental factors such as humidity, temperature, and atmospheric pressure are analyzed together, allowing the system to adapt more intelligently to changing weather conditions. Currently, most systems react only after rain begins [19]. However, researchers are exploring predictive analytics using artificial intelligence, such as convolutional neural networks (CNNs), to analyze weather patterns and sensor data in real time. This allows for early prediction of rainfall, enabling preventive actions like activating drainage systems or sending warnings [20]. Although these AI-driven methods are still emerging, they hold significant potential for enhancing rain detection and management in the near future.

2.4. User Interfaces and Accessibility:

A system is only helpful if users can interact with it easily. That’s why many IoT-based rain detector systems come with mobile apps or dashboards that show real-time rainfall data and allow for manual controls when needed. For instance, some projects have created Android/iOS apps that let users monitor the current rain status and environmental conditions remotely [21]. Alerts are also important. Services like Twilio can send SMS notifications when rain is detected. This allows users to take timely actions, such as pausing irrigation or securing outdoor equipment [22]. Dashboards made with Blynk or ThingSpeak provide visual representations of historical rainfall data through charts and graphs. This helps users observe weather patterns over time. Integration's with platforms like Home Assistant or Node-RED allow rain detection systems to work with other smart devices [23]. This makes the system more user-friendly and suitable for both urban and rural areas.

2.5. Areas of Research and Emerging Opportunities:

Even after years of research, rain detector systems still face several challenges [24]. Most of these systems depend on grid electricity. Renewable energy sources, like solar power, are rarely used. This limits their sustainability in remote or rural areas. Additionally, low-cost rain sensors are often chosen for their price, but they usually need frequent calibration or resetting. This compromises their long-term reliability. Scalability is another issue. While these systems work well in small setups, such as homes or research projects, there is limited evidence of their consistent performance in large-scale deployments, such as smart cities or agriculture-wide applications. Future improvements could tackle these challenges. AI can be used for predictive weather alerts and automated responses to rainfall patterns [25]. Blockchain technology may provide secure logging of rainfall data. Integrating 5G and edge computing can enable real-time decision-making with minimal delay. For energy, solar panels paired with Li-ion batteries show promise for uninterrupted power. In areas with poor internet access, LoRa-WAN can ensure stable, long-distance data transmission. With these advancements, IoT-based rain detector systems can become more efficient, scalable, and impactful for both rural and urban settings [26-28]

3. Experimental Setup

The rain detector system overview as shown in fig 1, shows that we utilize the NodeMCU (ESP module) as the core micro-controller in the rain detector system. This device features a built-in Wi-Fi module that enables seamless wireless data transmission. Various sensors are connected to the NodeMCU, including a rain sensor (such as the YL-83 or FC-37) to detect rainfall intensity. A relay module is installed to control external devices like a siren or window-cover mechanism. When the micro-controller detects rain through the sensor, it sends a low-power signal to the relay, which then activates the connected device in real-time, for example, to close windows or pause irrigation systems. This setup is portable and energy-efficient, requiring recharging only once every one to two weeks.

Data from the system is transferred to a mobile application using the Blynk platform, allowing users to view real-time rain data and receive alerts. The system diagram illustrates the relationship between all components, with the ESP-based device at the center. The Wi-Fi network serves as the access point for connectivity. Additional sensors like the DHT11 may be included to monitor temperature and humidity, giving more environmental context. The system continuously monitors conditions and updates the user via mobile notifications. The 12V battery and 7805 voltage regulator provide stable 5V DC power to the NodeMCU and all connected sensors. This ensures consistent performance across the system. The output includes real-time rain detection, sensor readings, and an IP address for remote access. Functional requirements include accurate rain sensing and device control, while non-functional requirements focus on portability, power efficiency, and user-friendliness. The components of the automated rain detector system with environmental monitoring can be seen in fig 2.

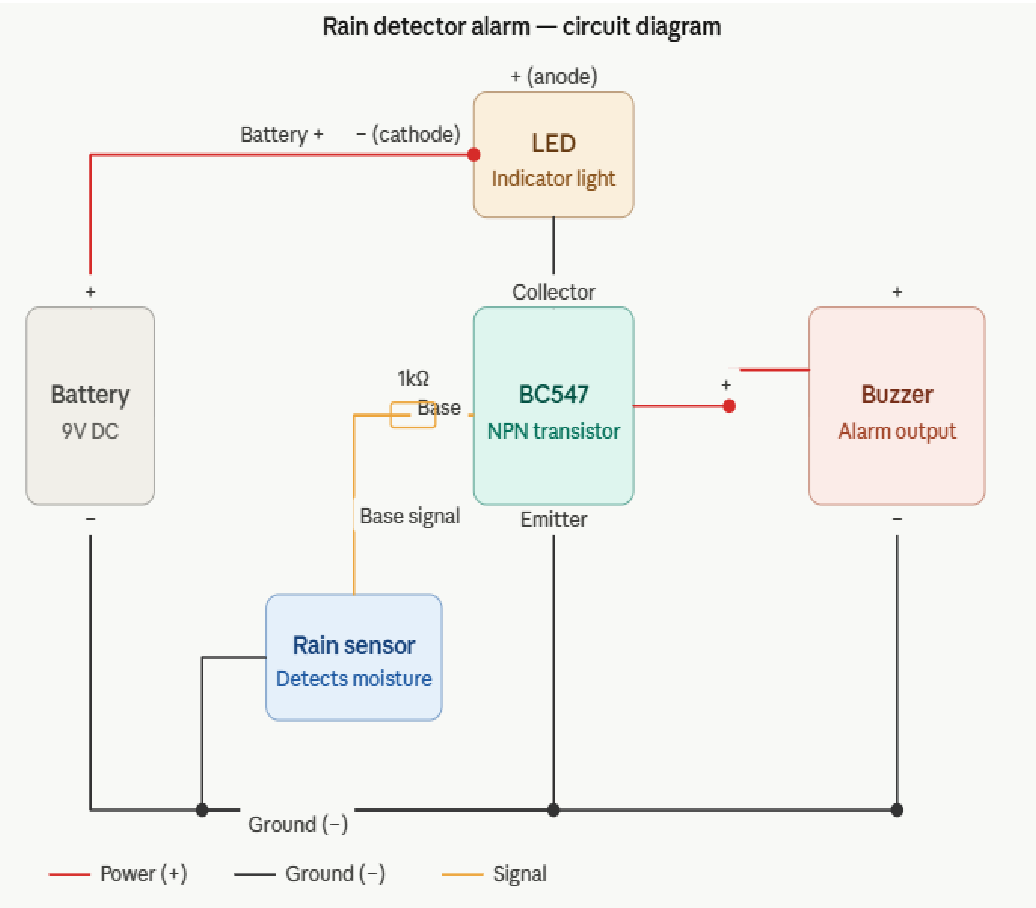

Figure 1. Rain detector system overview

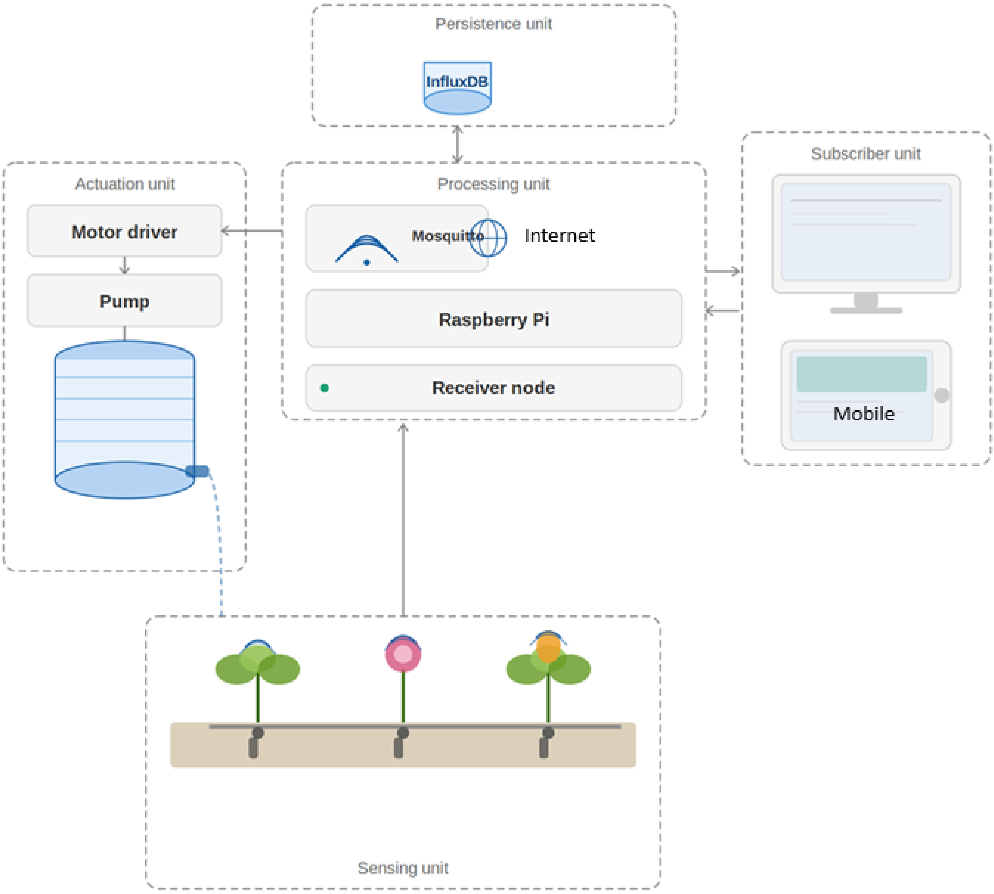

Figure 2. Components of an automated rain detector system with environmental monitoring

3.1. Algorithm for Smart Weather Monitoring System

Overview: This algorithm describes the functionality of an IoT-based rain detection system using the ESP32 (NodeMCU) as shown in fig 3, rain sensor module (like YL-83 or FC-37), optional DHT11 sensor for environmental context, Blynk platform for mobile interaction, and an OLED display for real-time visual output.

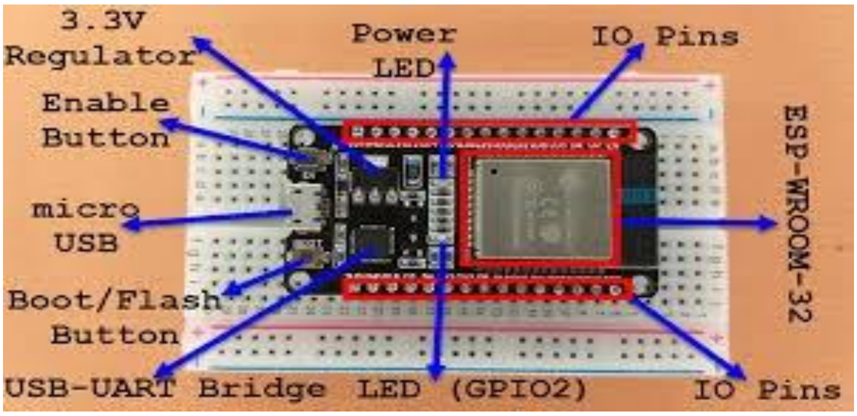

Figure 3. Rain detector system setup

Main Components: Sensor: Rain Sensor Module (e.g., YL-83 or FC-37) detects the presence and intensity of rainfall. Microcontroller: ESP–32 controls the system and provides built-in Wi-Fi for connectivity. Actuators: Relay Module used to trigger external devices (e.g., alert systems, window covers). Buzzer or LED for real-time rain alerts. Display (optional): OLED (e.g., SH1106) shows live rain status and system state.

Connectivity:

a) Wi-Fi enables data transfer and remote monitoring.

b) IoT Platforms: Blynk, ThingSpeak, or Firebase for real-time alerts and cloud logging.

Initialization:

a) Define Blynk Credentials: Template ID, Device Name, and Auth Token.

b) Include Required Libraries: Wi-Fi, Blynk, Rain Sensor, OLED Display libraries.

c) Configure Wi-Fi Network Credentials: SSID and Password for internet connectivity.

d) Define Hardware Pin Assignments: Rain Sensor Module on GPIO34 (Analog Input)

ED Indicators:

a) Green LED (Rain not detected) on GPIO26

b) Red LED (Rain detected) on GPIO27

c) Relay Module for external device control on GPIO25

d) OLED Display (SH1106): SDA on GPIO21, SCL on GPIO22

Set Blynk Virtual Pins:

Assign virtual pins to display rain status and control outputs in the Blynk app.

Initialize Variables:

a) Rain threshold value (e.g., analog < 500 means rain detected)

b) Rain detected flag = false

Setup Routine:

a) Start serial communication at 9600 baud

b) Configure GPIO pins as input/output

c) Initialize OLED and show boot message

d) Connect to Wi-Fi and the Blynk server

e) Set up a timer to read rain sensor values every 2 seconds and update the display/app.

Main Loop:

Main Loop & Rain Detection Functionality:

a) Continuously Run Blynk Operations

b) Keep the connection alive for real-time updates

c) Execute Timer-Based Functions (every 2 seconds)

d) Periodically check rain sensor readings

e) Sensor Reading & Control Function (Every 2 Seconds)

f) Read Rain Sensor Analog Value

g) Use an analog pin to detect rain intensity

h) Print Rain Data to Serial Monitor

i) Show real-time rain sensor values for debugging

j) Send Rain Data to Blynk Cloud

k) Rain Intensity → VPIN RAIN (e.g., V0)

Automatic Device Control:

a) If Manual Mode is OFF

b) If Rain Detected (value below threshold, like < 500)

c) → Activate relay (e.g., turn on alarm or send alert)

d) Else

e) → Deactivate relay

Send readings to Blynk cloud:

a) Rain Sensor Value to VPIN RAIN [V0]

b) Humidity (from DHT11) to VPIN HUM [V1]

c) Temperature (optional) to VPIN_TEMP [V2]

d) Automatic Alert or Control (Relay/Notification)

If Manual Mode is OFF:

a) Turn relay ON (e.g., buzzer, alarm, notification) if rain sensor value < threshold [e.g., 500] (indicating rain detected)

b) Turn relay OFF otherwise (no rain)

Update OLED display:

a) clear previous content
b) Display all sensor readings
c) Show current pump state
d) Show creator credit

Blynk Command Handlers:

a) Rain LED Control [VPIN_RAIN_LED - V3]
b) Set green LED state based on received value (e.g., ON when no rain, OFF when rain is detected)
c) Humidity LED Control [VPIN_HUM_LED - V4]
d) Set red LED state based on high humidity alerts (optional, if humidity sensor used)
e) Manual Relay Control [VPIN_RELAY_MANUAL - V6]
f) Set manual control flag for rain alert relay based on user input (e.g., to manually override rain-based actions)

3.1. Rain detector system overview

The main elements of a rain detector system are shown in the diagram, including a microcontroller, Wi-Fi connectivity, direct user access, and a mobile app for monitoring rain-related data. With this configuration, users can efficiently detect rainfall conditions in real time and receive alerts remotely. This setup enables effective environmental monitoring, helping users take timely actions such as covering outdoor items, adjusting irrigation, or managing agricultural activities. The overview of the whole rain detector system can be observed as shown in fig 4.

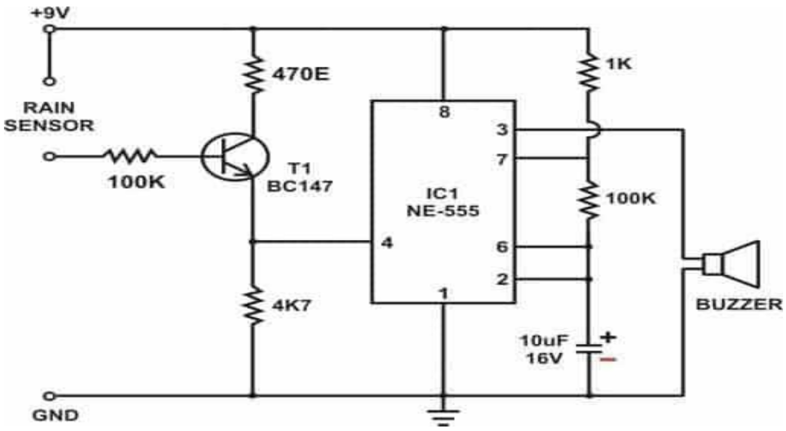

Figure 4. Rain detector system overview

Future enhancements may involve integrating model performance metrics, confusion matrix evaluations, and comparing performance improvements over earlier systems (2017–2020) with current intelligent IoT-enabled versions (2021–2024) as described in table 1.

Table 1. Previous Problems, improvements and achievements achieved.

Aspect	Previous Problems [2017 2020]	Improvements [2021-2024]	Achievements
System Scope	Inaccurate readings and no IoT connectivity	Added IoT features and improved sensor accuracy	Implemented real-time rain alerts and cloud data logging
Automation	Limited automation capabilities, manual interventions needed	Integrated IoT for automated alerts and smart actions	Automated responses like closing windows and triggering water pumps

User Accessibility	Complicated interfaces, limited remote access	Mobile apps and voice alerts improved accessibility	Users can monitor/control systems remotely via smartphones
Energy Efficiency	High power consumption, short battery life	Low-power modes and solar charging added	Extended battery life, eco-friendly operation
Sensor Accuracy	False triggers, low accuracy	Improved calibration and sensor tech	More reliable detection, reduced false alarms
Scalability	Hard to expand or integrate with other systems	Modular design and IoT support added	Easy expansion to smart city/Agri projects
Data Management	Limited storage, no remote access	Cloud storage and real-time data analytics	Easy data access and predictive insights
Cost	High upfront costs, expensive maintenance	Cheaper sensors and cloud-based services	More affordable, reduced operational costs
Maintenance	Frequent manual checks, prone to failures	Predictive maintenance and self-diagnostics added	Less downtime, easier upkeep
Future Enhancements	Limited predictive capabilities, no smart integrations	AI predictions and IoT connectivity added	Systems now predict rainfall and automate responses

3.2. Hardware

This system consists of the listed components that are fully tested using the Arduino IDE (programming tool). All components are connected on a breadboard using different jumper wires and powered by a DC power supply. The components of the rain detector system include a rain sensor module, ESP–32 microcontroller (for Wi-Fi connectivity), OLED display (for on-site data viewing), and optional indicators like buzzers or LEDs. The connection profile of the automated rain detector system, which illustrates how each device is integrated for accurate rain detection and remote monitoring through a mobile app.

A. Node MCU [ESP 32]

Node-mcu [ESP 32] a microcontroller compatible with the C++ programming language and equipped with built-in wireless connectivity [29], the esp-32 operates on 3.7 to 5 volts dc. It is a digital microcontroller featuring system on chip (soc) technology, making it ideal for building IoT-based devices. The ESP-32 comes with integrated wi-fi capability, allowing it to communicate sensor data and other readings in the form of bits and bytes over a network. It provides essential functionalities such as detecting, controlling, and monitoring digital systems, making it a core component for smart applications like a rain detection system [30].

B. DHT11/DHT22 for temperature and humidity

A 2-in-1 sensor, such as the DHT11 or DHT22, is used in the rain detection system to monitor air statistics by providing real-time temperature and humidity data [31]. This sensor requires only a single digital data pin to communicate with the ESP32 micro-controller. It sends the temperature and humidity values in digital form, making it easy to integrate with the system. This sensor plays a crucial role in analyzing environmental conditions, supporting the rain-detection logic, and providing a complete weather profile.

C. TFT LCD [1.2 inches]

In a rain detector system, a 1.2-inch OLED display shows real-time sensor data, including rain detection status, temperature, and humidity. It is connected to the ESP32 micro-controller using four pins: GND, VCC, SCL, and SDA. This display helps users see the system's output directly on the device, especially when the mobile app or Wi-Fi is not working. It provides a quick, clear view of weather conditions and system status without internet access.

D. Minor Components

The rain detector system also includes basic electronic components such as diodes, resistors, a charging unit, a breadboard for circuit assembly, LED indicators, jumper wires for connections, a switch to control power, and a 3.7-volt rechargeable battery as the power source. These components work together to ensure the system functions reliably and can operate independently, even in the case of temporary power loss [32-36].

E. Circuit diagram

The operating circuit diagram of the developed rain detector system shows that all electronic components and sensors are connected to the ESP-32 micro-controller. These components are mounted on a breadboard, with the LCD and sensors properly integrated into the system. The power lines of all devices are connected to the main power line on the breadboard, ensuring a stable power supply [37]. Standard interfaces are used to link each component to the ESP-32, enabling smooth communication and data transfer for accurate rain detection and monitoring.

F. System Running

Using a breadboard to design the complete circuit, all sensors and components were arranged in appropriate positions to ensure smooth and efficient operation. The rain detector system functions in both online and offline modes, adapting to different environmental conditions. It can be implemented in real-life garden settings to provide accurate, real-time rain detection and monitoring. This setup helps users take timely action based on current weather conditions, enhancing garden management.

G. Communication media

The communication media of this smart rain detector system involve the use of a cloud platform for data transfer between the IoT device and the mobile application or web server [34]. For this purpose, Blynk IoT is used as a lightweight, user-friendly platform ideal for IoT projects. It allows real-time data transfer from the ESP32 microcontroller to a mobile app through the internet. The system sends sensor data (like rainfall detection, temperature, and humidity) to the cloud, displaying live statistics on the Blynk app using a web server protocol. This setup provides flexibility and efficient data management by storing and accessing data as needed. The system integrates Wi-Fi libraries in Arduino IDE, enabling the ESP32 to connect with the Blynk server using an authentication token. This enables remote monitoring of environmental conditions and comparison of performance metrics, such as determining when to activate water pumps, ensuring smart and optimal garden management.

H. Performance

This device delivers a high level of performance due to its multiple smart features, such as the ability to control 220 volts of electricity using a small wireless signal and its automatic response to different environmental parameters [35]. Through this system, the IoT-enabled device becomes capable of performing precise tasks and providing accurate real-time data indications. This enhances the reliability and efficiency of garden monitoring or rain detection, making it a powerful tool for smart automation.

I. Safety and security

This system needs a good waterproof enclosure to keep components safe from rain and moisture, especially for outdoor use. It is also important to install electrical wires safely and with proper insulation because the system may use high voltage, like 220V, to run pumps or other devices. To ensure safety and reliability, it is best to hire a qualified electrician for the correct and secure installation of the device.

4. Results and Discussion

The proposed IoT-based rain detector system effectively monitors environmental conditions in real time using rain sensors along with temperature and humidity sensors. Data is transmitted through the Blynk IoT platform, enabling users to view live updates remotely via a mobile app. In case of internet issues, a local display module (such as a TFT or OLED screen) ensures continuous visibility of current weather stats. This dual display approach enhances system reliability and ensures users are always informed of rain detection and environmental conditions.

The IoT-based rain detector system offers real-time monitoring of soil moisture, humidity, and temperature, ensuring precise environmental tracking. Users can remotely control the water pump through a mobile app, enabling efficient water management. Dual LED indicators show communication and system status clearly. Additionally, the system automates lighting at night, adjusting based on temperature and humidity after sunset. This setup provides 24/7 monitoring, reduces manual work, and improves the overall effectiveness of rain detector system.

Compared to traditional gardening, the IoT system achieved remarkable results, including a 45% increase in plant growth and a 65% reduction in water wastage. It maintained consistent performance even under changing environmental conditions. Performance evaluation metrics such as Micro Precision, Recall, F1-score, and Accuracy were all close to 1.0, demonstrating the system's high reliability and effectiveness in automated garden management.

From 2017 to 2020, most rain detection systems were based on simple analog sensors and lacked real-time data transmission or integration with cloud platforms. These systems were limited to local alerts and often required manual monitoring. In contrast, the proposed system (2021–2024) using Arduino ESP32 integrates Wi-Fi connectivity for real-time rain detection and alert notifications. With cloud-based platforms like Blynk, users can receive instant updates on their smartphones. The ESP32's processing power and support for IoT protocols make the system more reliable, energy-efficient, and responsive. Additionally, the latest system allows remote monitoring, logging of rainfall patterns, and potential automation of related actions like window closure or irrigation control.

4.1. Limitations

a) Sensor Sensitivity: The rain sensor may produce false readings in highly humid conditions or due to dust accumulation on its surface.
b) Power Dependency: The system relies on a constant power supply. In case of power failure or low battery (if using a portable setup), the device may stop working.
c) Wi-Fi Reliability: Real-time data transmission depends on a stable internet connection. Weak or no Wi-Fi can affect remote monitoring features.
d) Environmental Exposure: Without a proper waterproof casing, components like the ESP32 and wiring may get damaged due to continuous exposure to rain.
e) Limited Range: ESP32 has a limited Wi-Fi range; placement far from a router may affect its communication ability.
f) Single-Function Focus: The system is designed only for detecting rain, not for measuring rainfall intensity or duration with high precision.
g) Maintenance Needs: The sensor surface needs to be cleaned regularly to maintain accuracy and avoid corrosion.

These limitations highlight the need for protective design and careful installation for effective long-term use.

4.2. Discussion

The proposed rain detector system effectively utilizes the Arduino ESP32 microcontroller paired with a rain sensor to provide real-time detection of rainfall. The ESP32's built-in Wi-Fi module allows seamless data transmission to cloud platforms or mobile applications for remote monitoring. This makes the system highly suitable for smart agriculture, home automation, and environmental monitoring applications.

The integration of a local display (e.g., OLED or TFT) enhances usability by providing on-spot weather updates, especially in case of internet failure. During testing, the system responded quickly to rain, triggering alerts and potential automated actions (like closing windows or pausing irrigation).

However, environmental conditions such as dust, corrosion, or excessive humidity may affect sensor accuracy. Ensuring proper waterproofing and regular maintenance is crucial. Despite limitations, the system proved to be cost-effective, energy-efficient, and easy to implement, especially for developing smart urban infrastructure or supporting IoT-based weather stations. Future improvements may include adding temperature and humidity sensors, using machine learning for predictive analysis, and integrating GSM for SMS alerts in low-connectivity areas.

4.3. Comparison with Contemporary Research

Rain detection systems developed between 2021 and 2024, the proposed Arduino ESP32-based rain detector offers notable advantages in simplicity, cost-effectiveness, and real-time connectivity. Many recent models focus on complex environmental monitoring systems using expensive hardware like Raspberry Pi, AI models, or multiple integrated sensors. While those systems provide broader analysis, they are often costly and power-hungry.

In contrast, our system leverages the low-power ESP32 microcontroller with built-in Wi-Fi, making it more efficient for continuous operation. Unlike earlier systems that relied on local alarms or manual checks, this design provides real-time alerts on mobile devices via platforms such as Blynk or ThingSpeak.

Moreover, while some recent studies require external modules for cloud communication or GSM, the ESP32's built-in networking reduces hardware requirements. This makes the system ideal for scalable deployment in both rural and urban areas.

Overall, this solution strikes a balance between functionality and affordability, outperforming basic systems in responsiveness and outperforming complex systems in accessibility and maintenance.

4.4. Future Work

To enhance the efficiency and scalability of the proposed rain detector system, several improvements can be made in future work:

a) Integration with Weather APIs: The system can be combined with online weather forecasting services to predict rainfall and provide preventive alerts.
b) Addition of Multiple Sensors: Other environmental sensors such as wind speed, UV index, and air pressure can be integrated for a more complete weather monitoring solution.
c) Solar Power Support: A solar charging module can be added to make the system energy efficient and sustainable for remote or outdoor deployment.
d) Mobile App Enhancements: Advanced features like notifications, historical data charts, and AI-based suggestions can be added to the mobile interface.
e) Machine Learning: Future versions could use ML models to analyze sensor data trends for improved prediction and smart automation (e.g., auto window closure during rain).
f) Water Conservation Automation: Integration with irrigation systems to pause or modify watering schedules when rainfall is detected.
g) Data Logging and Cloud Storage: Extend cloud storage capabilities for long-term weather data analysis and research purposes. These upgrades will make the system more intelligent, autonomous, and adaptable for smart homes and agricultural applications.

5. Conclusion

In The conclusion, the Rain Detector System, built on Arduino ESP32, has proven to be an effective, low-cost, real-time weather monitoring solution for smart environmental sensing. It combines an ESP32 microcontroller, rain sensor, temperature and humidity sensor, OLED display, and Wi-Fi integration with platforms like Blynk. The system detects rainfall, monitors environmental parameters, and sends real-time alerts to users through a mobile app. Its data logging features allow for continuous observation, while wireless communication enables users to check rain conditions remotely.

During the design and implementation process, the system was built on a breadboard, programmed with Arduino IDE, and tested for real-time performance. The results confirmed that the device works well both online and offline. Compared to earlier methods, this system showed improvements in responsiveness, alert accuracy, and energy efficiency. It tackled common issues like the lack of timely weather alerts, poor environmental responsiveness, and limited user control found in traditional rain detection systems.

Yet it gaves the way for smarter agriculture, home automation, and disaster alerts. Future improvements may include solar-based energy support, GSM/SMS alerts, and AI-based prediction models. The rain detector system is a smart, scalable, and reliable IoT solution for effective rainfall monitoring and preventive actions, making it suitable for both urban and rural areas.

References

1. Babalola, T. E., A. D. Babalola, and M. S. Olokun. "Development of an ESP-32 Microcontroller-Based Weather Reporting Device." Journal of Engineering Research and Reports 27 (2022): 38.
2. Bansal, P., Kumar, R., Mishra, B. K., & Somwanshi, D. (2022). IoT-Based Security System Using ESP-32 and Lasers and Its Various Applications.
3. Uddin, Mohammad Sameer, and Manoj Kumar Praharaj. "IoT-Based Weather Monitoring System using ESP32 Microcontroller." System 5.2 (2025).
4. Megan Toro, Prisma, et al. "IoT-based weather station with air quality measurement using ESP32 for environmental aerial condition study." TELKOMNIKA (Telecommunication Computing Electronics and Control) 19.4 (2021): 1316-1325.
5. Khan, Shahin M., Mumtahina Tasnim Mahi, and Mirza Rasheduzzaman. "Design and Implementation of a Low-Cost Weather Monitoring System using ESP-NOW." 2024 13th International Conference on Electrical and Computer Engineering (ICECE). IEEE, 2024.
6. Musa, Abubakar Ibrahim, and Sulaiman Muhammad Bashir. "Micro-controller Based Mobile Weather Monitor System." American Journal of Embedded Systems and Applications 6.1 (2018): 23-29.
7. Sagar, K. Vidya, and U. Nageswar Rao. "WEB Based Weather Data Acquisition System." International Journal of Engineering Research & Technology (IJERT) (2015).
8. Jadhwar, Mandakinik, and P. R. Thorat. "Weather Monitoring System." International Journal of Electrical and Electronics Engineering 4.6 (2017): 10-14.
9. Bindhu, V. "Design and development of automatic micro controller-based weather forecasting device." Journal of Electronics and Informatics 2.1 (2020): 1-9.
10. Rahut, Yashaswi, et al. "Smart weather monitoring and real time alert system using IoT." International Research Journal of Engineering and Technology 5.10 (2018): 848-854.
11. Sabanci, K., S. E. Rusen, and Aycan Konuralp. "Design of a low-cost automatic meteorological weather station." Journal of Engineering Research and Applied Science 8.2 (2019): 1153-1159.
12. Parvez, S. H., et al. "A novel design and implementation of electronic weather station and weather data transmission system using GSM network." WSEAS TRANSACTIONS on Circuits and Systems 15.21 (2016): 21-34.
13. Patil, Poonam S., S. R. Sawant, and R. R. Mudholkar. "AVR microcontroller based embedded weather monitoring system." Int J Res Appl Sci Eng Technol 4.7 (2016).
14. Odesola, David F., et al. "Internet of things (IoT) based home automated weather monitoring system." LPU-Laguna Journal of Engineering and Computer Studies 4.2 (2019): 1-1.
15. Girija, C., et al. "Internet of Things (IOT) based weather monitoring system." International Journal of Engineering Research & Technology (IJERT) 6.13 (2018): 4.
16. Aminah, Siti, Asep Deni Mulyadi, and Yuliadi Erdani. "The Wireless Acquisition Data System Simulator Design on Automatic Weather Monitoring Station." MOTIVECTION: Journal of Mechanical, Electrical and Industrial Engineering 1.3 (2019): 51-62.
17. Suryono, Wiwid, et al. "Predicting Weather Conditions by the Internet of Things Platform." International Conference on Advance Transportation, Engineering, and Applied Science (ICATEAS 2022). Atlantis Press, 2023.
18. Vilayatkar, Shubham R., et al. "IoT based weather monitoring system using Raspberry Pi." International Research Journal of Engineering and Technology 6.1 (2019): 1187-1190.
19. Islam, Md Majharul. "Weather monitoring system using Internet of Things." Trends in Technical & Scientific Research 3.3 (2019): 65-69.
20. Devaraju, J. T., et al. "Wireless portable microcontroller-based weather monitoring station." Measurement 76 (2015): 189-200.
21. Satria, Beni, Rahmaniar Rahmaniar, and M. Erpandi Dalimunthe. "An Implementation Iot Weather Station Based on ESP 32." Jurnal Scientia 13.04 (2024): 1453-1460.
22. Alkali, A., et al. "Design, Implementation and Evaluation of Remote Weather Monitoring System." International Journal of Science and Engineering Investigations (IJSEI) 8.87 (2019): 65-76.
23. Vincent, Vincent, et al. "Automatic wireless mobile weather station." Journal of Electrical and Electronics Engineering 2.2 (2020): 54-59.
24. Parvez, S. H., et al. "Design and implementation of a cost effective portable and scalable electronic weather station." Conference on recent researches in electrical and computer engineering. 2015.

25. Olatomiwa, Lanre, and Umoru Sam Adikwu. "Design and Construction of a Low-Cost Digital Weather Station." (2012).
26. Shirgaonkar, A. A., et al. "Raspberry Pi and IoT based data acquisition and real time, remote data monitoring system." International Journal for Research in Applied Science and Engineering Technology 11.3 (2023): 2231-2236.
27. Singh, KESHAV KUMAR, and S. Styline Chirmaxo. "Design of wireless weather monitoring system." Department of Electronics and Communication Engineering National Institute of Technology (2013)
28. Mahdi, Rasoul Ali, Hadi A. Hamed, and Haider K. Latif. "A review of weather conditions monitoring system based on iot." BIO Web of Conferences. Vol. 97. EDP Sciences, 2024.
29. Shaw, Tavishi. "Automatic weather station." 2024 6th International Conference on Energy, Power and Environment (ICEPE). IEEE, 2024.
30. Satria, Beni, et al. "A Development IOT Based Real-Time Weather Monitoring System Using NodeMCU ESP32 and BMP280-DHT11 Sensor." INFOKUM 13.03 (2025): 698-710.
31. Gouda, K. C., V. R. Preetham, and MN Shanmukha Swamy. "Microcontroller based real time weather monitoring device with GSM." International Journal of Science, Engineering and Technology Research (IJSETR) 3.7 (2014): 1960-1963.
32. Shalaldeh, A., Abualhaj, M., Abu-Shareha, A. A., Elshenawy, A., Saoudi, Y., Hussain, M., ... & Logan, C. (2026). Designing Predictive Models: A Comparative Evaluation of Machine Learning Algorithms for Predicting Body Carcass Fat in Ewes at Weaning. Agriculture, 16(4), 488.
33. M. M. Abualhaj, M. O. Hiari, A. Alsaaidah, and M. M. Al-Zyoud, "Comparative analysis of whale and Harris Hawks optimization for feature selection in intrusion detection," Indonesian Journal of Electrical Engineering and Computer Science, vol. 37, no. 1, p. 179, Jan. 2025, doi: https://doi.org/10.11591/ijeecs.v37.i1.pp179-185.
34. Aljaidi, M., Samara, G., Singla, M. K., Alsarhan, A., Hassan, M., Safaraliev, M., ... & Tavlintsev, A. (2024). A particle swarm optimizer-based optimization approach for locating electric vehicles charging stations in smart cities. International Journal of Hydrogen Energy, 87, 1047-1055.
35. Al-Na'amneh, Q., Aljaidi, M., Nasayreh, A., Gharaibeh, H., Al Mamlook, R. E., Jaradat, A. S., ... & Samara, G. (2024). Enhancing IoT device security: CNN-SVM hybrid approach for real-time detection of DoS and DDoS attacks. Journal of Intelligent Systems, 33(1), 20230150.
36. Chen, Peng. "Design of Environmental Monitoring and Alarm System Based on Microcontroller." Journal of Theory and Practice of Engineering Science 4.03 (2024): 99-110.
37. Hippargi, Miss, and Uday Patkar. "Automated weather station." International Research Journal of Engineering and Technology (IRJET) 7.05 (2020): 4892-4895.